The All New And Improved BBBS SysOp Manual, for v3.33
How

Kalle Soiha and Copyright

COLLABORATORS

	TITLE : The All New And Improved BBBS SysOp Manual, for v3.33 How		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Kalle Soiha and Copyright	February 15, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

The All New And Improved BBBS SysOp Manual, for v3.33 How

1.1 The BBBS System Operator Manual - Welcome!

```
                oooooooooooooooo    oooooooooooooooo    oooooooooooooooo            ↩
                   ooooooo o
 BBBBBB""""BBBBBBo   BBBBBB""""BBBBBBo   BBBBBB""""BBBBBBo    oBBBBB""""BBBo
 BBBBBB     BBBBB"   BBBBBB     BBBBB"   BBBBBB     BBBBB"   BBBBBB       "B
 BBBBBB   oBBBB""    BBBBBB   oBBBB""    BBBBBB   oBBBB""    BBBBBBBoo     "
 BBBBBBBBBBBBooo     BBBBBBBBBBBBooo     BBBBBBBBBBBBooo     "BBBBBBBBBBBoo
 BBBBBB"   "BBBBBo   BBBBBB"   "BBBBBo   BBBBBB"   "BBBBBo     "BBBBBBBBBBBBB
 BBBBBB      BBBBBB  BBBBBB      BBBBBB  BBBBBB      BBBBBB        """BBBBBBBB
 BBBBBB      BBBBBBB BBBBBB      BBBBBBB BBBBBB      BBBBBB Bo         BBBBBB"
 BBBBBB     oBBBBBB  BBBBBB     oBBBBBB  BBBBBB     oBBBBBB BBBo      oBBBBB"
 BBBBBBBooBBBBBBB"   BBBBBBBooBBBBBBB"   BBBBBBBooBBBBBBB"   oBBBBBBBBBBB""
"""""""""""""""     """""""""""""""     """""""""""""""          """"""

                                Version 3.33 How
             Copyright (c) 1990-1997 Kim Heino and Tapani T. Salmi

                            System Operator’s Manual

           Welcome to BBBS! Please select a topic you wish to view...


                First-time installation

                Table of Contents

                The BBBS License and contact info
```

1.2 The BBBS v3.33 How System Operator Manual - Table of Contents

```
                                                WELCOME TO BBBS!
                                ================

                First-time installation of BBBS
```

CONFIGURATION
=============

GENERAL STUFF
=============

THE BBBS SCRIPT LANGUAGE
========================

OPERATING SYSTEM CONSIDERATIONS
===============================

APPENDIDICES

==========

1.3 First-time installation of BBBS

First of all, thank you for choosing BBBS, the powerful, all-in-one BBS software solution!

Creating a working BBS system with BBBS is very simple. As you are reading this, you probably have already extracted the BBBS distribution package to a directory. This directory is called the "BBS directory". It is probably a good idea to name it something like c:/bbs, if not but for simplicity's sake.

To get a minimal configuration, you will need to run the
BCFG4 configuration program
. You can do a lot of configuration there, but for starters you will just have to configure the name of your BBS and your own name. Other options should be quite correct, but you can check them all if you like. If you run into trouble, you can at any time request help with the F1 key.

When you are finished with BCFG4, select exit and save and wait for BCFG4 to save the configuration you just created. Depending on how many conferences you created (and a lot of other things), the saving procedure can take a long time or a very long time. Remember that patience is a virtue!

When BCFG4 is finished, you have a working configuration. Now all you need to do is log in to the system once to register yourself as the system operator. BBBS can be run in local mode by executing it with the command line BBBS 0. The first user that will register into the system will be the main system operator or "user 0", with access to everything. After you have done this, you have a fully functional setup.

To get the most out of BBBS, though, you will need to do a bit more configuring. Just a few things you might like to check are if your
menu files

are OK, what kind of
file areas
you would like to have, if the
user access configuration
is ok and especially if
external programs
, such as packers work. But otherwise, BBBS is now installed. Have fun!

1.4 Configuring BBBS, an overview

BBBS, being a comprehensive, all-in-one system, has a lot of features and therefore there is also a lot to configure. The basic things are configured with the
BCFG4 Configuration Program
, the rest with ASCII-format text files residing in your BBS-directory, which can be edited with your favorite editor or with the BBBS's internal mg editor.

Most important one of these is the file
groups
. It is used to define user access groups. Groups are the basis of BBBS security, so it is very important that you have a thorough look into the file and how it is formed.

A set of important files are also the 'filedir*'-files. These are used to create
the file areas
in your BBBS system. The BBBS filesystem, File/4, is a very powerful "OS-like" filesystem, with tree-structure and file manipulation commands familiar from most operating system shells.

A lot of miscellaneous configuration is in the
external.bbb
-file. In it resides the configuration of external programs, such as packers and protocols. Also, a lot of configuration related to networking with other BBS systems can be found in this file.

The configuration of BBBS groupmail features resides in the
alias.bbb
-file. This is where things like mailing lists and comment receivers are defined.

BBBS has also a lot of Internet access tools. The behavior of BBBS in Internet and other TCP/IP-networks can be defined in the file
inet.bbb
.

The BBBS chat system
is also a very complicated, channel-based one. The available channels and feelings can be configured with the files in the feelings-directory, defined in
BCFG4
.

In addition to these configuration files, there are also a lot of other files that are used to do all sorts of miscellaneous configuration.

Examples of all configuration files came with the distribution package. In addition to the examples in this manual, do refer also to these files, as they are pretty straightforward and self-explainatory.

1.5 The BCFG4 configuration program

The BCFG4 configuration program is used to do most of the configuration in BBBS. The BCFG4 interface is very easy to use. If you run into trouble or do not understand something, the F1 key will give you immediate help.

The BCFG4 program is called bcfg4.exe. The calling convention of this program is simple:

```
bcfg4.exe [node]
```

Where node is the number of the BBBS node you want to configure. In BCFG4, the global configuration is always the same regardless of the node specified, the local configuration is specific for each node.

If you specify "bad" as the node-parameter, then BCFG4 will scan your badecho-directory for nonexistant echos and create them according to the rules in the badecho-section of external.bbb-file , if there is at least one "unnamed" conference in your conference configuration.

The things you'll probably want to check are:

- The general stuff
- The global toggles
- The conference configuration
- The modem setup for each node
- If you want to have networks, the FidoNet configuration

Otherwise, the default values should be pretty much OK. You can (and should) go through everything , though, just to make sure.

1.6 The GROUPS-file: defining user access

BBBS's powerful user access management is based on groups. Groups are collections of other groups with a distinctive name. The groups you define are global. That is, you can use them anywhere in the system. In every system you will most likely run into a situation where you need groups. Just one of them is when you need to create a "private" conference or a "news" conference. This is all very easy with groups. All you need to do is to type a few lines.

Defining general groups

Groups are defined in the file groups, residing in the BBS-directory. The format of the file is as follows; blank lines and lines beginning with a semicolon (';') are ignored:

```
name_of_group:group1{,group2{,...}}
```

As you can see, a group contains only other groups. Every user is a member of the groups called all and first.last, where first is the first name of the user and last is the last name. So, to define a group called bz with the users "Kim Heino" and "Tapani Salmi", you write:

```
; Creating a simple general group with two users:
;
bz:kim.heino,tapani.salmi
```

Then, to create a group simka with the members of the bz-group and also the users "Pertti Heikkinen" and "Rune Johansen", you would write:

```
; An example of creating a group called 'simka' with another group:
;
simka:bz,pertti.heikkinen,rune.johansen
```

Group names that include an exclamation mark ('!') have a special meaning: such groups include all members of the group left of the exclamation mark, except the members of the group at the right side of it. For example, the group fsysop !bz contains all users of the group fsysop, except the members of the group bz. A shortcut for all!foo is !foo. An example:

```
; Example of using exclusive groups:
;
simka:bz,pertti.heikkinen,tuomo.soini,rune.johansen
finnish:simka!rune.johansen
;
; Another example; group true has persons that are members of groups foo
; and bar:
;
foo=kim.heino,tapani.salmi,kalle.soiha,olli.trm,sami.virtanen
bar=rune.johansen,kim.heino,olli.trm,matti.luoma
temp=!foo,!bar
true=!temp
;
; So, the group true contains the users Kim Heino and Olli Trm.
;
```

Remember that if you try to create group foo with the line:

```
foo:all!kim.heino,all!tapani.salmi
```

The result is that everybody is a member of the group foo, since they are members of the first group or the second group and the group foo is the two put together. The correct way to do this is would be:

```
bz:kim.heino,tapani.salmi
foo:all!bz
; or foo:!bz
```

Remember also that the group all!all has no sense: it is an empty group.

Another group name that has a special meaning is any group with a string like @yymmdd added to it, where yymmdd is a date (yy specifies the year, mm the month and dd the day). Groups like this have effect in the group they are being added to only until the specified date is reached. 14 days before the date is reached, the members of the group receive a message that their access in a group will soon expire. When the date is reached, an exclamation mark is added to the group name. An example:

```
; Example of a group with a member that will exist only for a specified
; time:
;
kids:toni.saari,samuli.suominen,olli.trm@990903
;
; At 20.8. 1999, the line will change into:
;
; kids:toni.saari,samuli.suominen,olli.trm@990903!
;
; Finally, at 3.9. 1999, user Olli Trm will be removed from the group
; and the line will change into:
;
; kids:toni.saari,samuli.suominen
```

There are also so-called "age" groups. They are named <age and >age. Every user whose age is under age years, belongs into the group <age. Likewise, every user whose age is over age years, belongs into the group >age. An example:

```
; Example of a group with age definition:
;
kids2:samuli.suominen,<9
;
; In addition to the user "Samuli Suominen", everyone whose age is
; under nine years belongs into the group kids2.
```

Groups are also cumulative. That is, when you have created a group on one line, creating it "again" on an another line simply adds to the already existing group. The simka group defined above could have also been defined like this:

```
; Another example of creating a group called 'simka'.
;
simka:bz
simka:pertti.heikkinen,rune.johansen
```

This is good to remember as a single line in the groups file can only be 1023 characters in length. Remember also that the groups-file is read from top to bottom.

Restricting conference access with groups

For every conference you generate, there is also a set of groups, defining who can do what in the conference. Lets say you generate a conference called chat. For it, there are also four groups:

```
chat@   - members cannot read or write the messages in the conference.
chat@r  - members can only read messages in the conference.
chat@w  - members can both read and write messages in the conference.
chat@s  - members have SigOp access in the conference.
```

By default, all users are members of the chat@w-group, so they have full read and write access to the "chat"-conference. An example: to have only the members of the group simka and user "Kalle Soiha" be able to read the conference called "BBBS.Simka", you would write:

```
; An example of how to restrict conference access:
;
; First disallow everyone to read the bbbs.simka-conference:
;
bbbs.simka@:all
;
; Then allow group "simka" and the user "Kalle Soiha" to read and
; write into it:
;
bbbs.simka@w:simka,kalle.soiha
```

But what if you have a lot of network areas in your BBS and would prefer not to have the members of group newbies write into these conferences, only read? One way to do this would be:

```
; A bad example of limiting access from a lot of conferences:
;
sf.aloittelijat@w:all
sf.aloittelijat@r:newbies
sf.amiga@w:all
sf.amiga@r:newbies
; etc. ad nauseam
```

The BBBS group system offers a much quicker way to achieve this. You can create a group with a regular expression . There are four kinds of regexp groups:

```
name@=regexp   - Members of the group cannot read or write any messages
                 to conferences matching the regular expression
                 "regexp".
name@r=regexp  - Members of the group can only read messages from
                 conferences matching the regular expression "regexp".
name@w=regexp  - Members of the group can both read and write messages
                 to conferences matching the regular expression
                 "regexp".
name@s=regexp  - Members of the group have sigop access to conferences
                 matching the regular expression "regexp".
```

"name" can be any string you like, but it should be something that has something to do with the actual function of the group. Note, that "name" is NOT the name of the regular expression group. It is just an unique string that is part of the name. The real name of the regexp group name@r=regexp is name@r.

As you probably have already guessed, regexp groups can be very powerful indeed, though they can give you a major headache if you don’t know a lot about regular expression. It is very important to completely understand what regexp is. "Normal" OS-shell wildcards like ’?’ and ’*’ are part of regexp, but they have a completely different meaning in it.

As an example, let’s see how we can easily solve the FidoNet access problem with regexp groups:

```
; The correct way to limit access from a lot of conferences:
;
nonewbies@w=^sf\.:all
nonewbies@r=^sf\.:newbies
```

Notice the power of the regular expression. With just two lines, the write-access of group newbies is removed from all conferences beginning with the string "SF.". It doesn’t matter how many conferences there are, the regexp group automatically applies to all conferences matching the expression given.

Remember that the last specified conference group will take effect, not the most "powerful" one; that is, if you first give a sigop access to a certain group, and later on in the file give the same group a read-access, the group will have only read-access, NOT sigop-access. This is also true for regexp-conference groups.

List of all predefined groups that cannot be reassigned:

Group name	Members
all	all users, always.
firstname.lastname	user with the name "Firstname Lastname".
foo!bar	users who are members of the group foo, but not the group bar.
!foo	users who are not members of the group foo.
>age	users who are older than age years. (from the birthday-field of the "u a" command)
<age	users who are younger than age years. (from the birthday-field of the "u a" command)
account	users who have an account defined and have some credit/money left.
nomoney	users who have an account defined, but don’t have any credit/money left.
noaccount	users who don’t have an account defined.
limitx	users who have the limit value x.
sysx	users who have the sysop access x.

List of all groups that can be reassigned:

Group name	Who should be members?
conf@	users who have no read or write access to conference conf.

```
conf@r                   users who have only read access to the
                         conference conf.
conf@w                   users who have both read and write access to
                         the conferece conf.
conf@s                   users who have sigop access to the conference
                         conf.
conf@=regexp             users who have no read or write access to
                         conferences matching the regular expression
                         regexp.
conf@r=regexp            users who have only read access to the
                         conferences matching the regular expression
                         regexp.
conf@w=regexp            users who have read and write access to the
                         conferences matching the regular expression
                         regexp.
conf@s=regexp            users who have sigop access to the conferences
                         matching the regular expression 'regexp'.
nochat                   users to whom sysop should always be
                         unavailable for chat.
okchat                   users to whom sysop should always be available
                         for chat.
nonode                   users to whom members of other nodes should
                         always be unavailable (i.e. users who should
                         not be able to send node messages).
nofeel                   users who should not be able to use
                feelings
                .
okuser                   users who should be able to log in during a
                         "nouser"-event.
ftp                      users who should be allowed to use the main
                         menu command 'ftp' to FTP out.
rlogin                   users who should be allowed to use the main
                         menu command 'rlogin' to rlogin out.
telnet                   users who should be allowed to use the main
                         menu command 'telnet' to telnet out.
finger                   users who should be allowed to use the main
                         menu command 'finger' to finger out.
hunt                     users who should be allowed to use the main
                         menu command 'hunt' to play Hunt.
hack                     users who should be allowed to use the main
                         menu command 'hack' to play NetHack.
irc                      users who should be allowed to use the
                         BBBS groupchat to interface with the Internet Relay
                         Chat.
```

1.7 Defining file areas in BBBS

The BBBS filesystem, File/4, is a bit different than the filesystems on most systems you've seen. File/4 has a tree-like structure and is based on a command shell, so users familiar with most OS shells should feel pretty much at home.

The core of File/4 is the virtual directory structure defined in filedir*-files

. Virtual directory structure can but need not be the same as the real directory structure in your hard disk. Individual directories are in turn set up with
descript.ion-files
and the
BBBS group system
.

1.8 The directory structure: filedir*-files

The virtual directory tree for File/4 is created with filedir-files in your
BBS root directory
. This is the directory structure that the users see when browsing your file areas. Filedir-files have a running number as their extension. There are three kinds of filedir-files. For each type of filedir-file the extension has a different meaning. The different types are:

```
1) The filedirg-files (named filedirg.nnn, starting from 000). These
   are used to define the global file directories. Global file
   directories are always added to the directory structure. The
   extension number in filedirg-files has no special meaning, but you
   can have multiple filedirg-files, if you have large fileareas and
   want to keep different kinds of directories in different
   filedirg-files.

2) The filedirn-files (named filedirn.nnn, starting from 001). These
   are used to define directories specific for each node. For example,
   directories that should be available only for node 1 should be
   placed into the file filedirn.001. The first such line that
   the user can write to in the filedirn-file defines the "hold"-directory
   for that node and it must exist. Hold directory is used by the
   users to temporarily hold files which they want to download or otherwise
   handle. If you do not create a hold directory for a node, BBBS will
   complain "hold not found" when a user enters the fileareas. Therefore,
   you'll need a filedirn-file for each node you want to be able to access
   files. For things to be universal in all BBBS systems, please name your
   hold directory /tmp.

3) The filedirc-files (named filedirc.nnn, starting from 000). These
   are used to define directories specific for a certain conference.
   For example, the file filedirc.012 specifies directories that are
   added to the directory structure only when the user is in conference
   number 12.

The structure of all filedir-files is simple. Each line of the filedir-file specifies one directory entry. A single entry should consist of three fields:

    virtual_directory   real_directory   description_and_flags

Virtual directory is the name of the directory that the user sees when browsing the file areas. It should be written in lower case. Creating a directory like /Windows/Games will not work, it will only show the directory /windows and not its subdirectories. This is one of the
```

top-19 errors
.

Real directory is the actual directory on your hard disk or CD-ROM where the files belonging to this directory should be stored. You must create these directories.

It is very important that you use a forward slash ('/') as the directory separator in BOTH virtual directory and real directory fields and NOT a backslash ('\'). This is because backslash is used as an escape character in BBBS. Also, it is very recommended that you use only lowercase characters in virtual directory pathnames.

Description is simply a short description of what kind of files the directory contains.

Flags are very important: they are used to specify various things for the directory. One of these is the directory access. By default, everyone can read the files in every file area. To define which groups are allowed to read and/or write from/to the directory, the following flags can be used:

```
@r:group1{,group2{,...}}     (specified groups are given READ access to
                              the directory)
@w:group1{,group2{,...}}     (specified groups are given WRITE access to
                              the directory)
@rw:group1{,group2{,...}}    (specified groups are given both READ and
                              WRITE access to the directory)
(@wr flag is synonymous to the @rw flag)
```

READ access means that the users belonging to such a group can only read files. That is, they can type or download files in the directory. WRITE access allows users to also delete or move the files in the directory. Therefore, users should normally have only READ access to most directories.

The @e-flag, when added to the flags-field of the directory, is used to prevent new files scanning for that directory. This is useful for directories that don't change, like CD-ROMs.

By default, BBBS uses a file called descript.ion in the directory's real directory as the description file for the directory. You can change this with the @0 and @1-flags. The difference is that with @0, the file size and date are read from the disk, with @1 they are read from the descript.ion-file. This is useful with CD-ROMs.

For example, if you specify this in your filedir-file:

```
/                c:/root/            Root directory
/graph           c:/pictures/        Assorted pictures
/graph/people    c:/pictures/faces/  People @0c:/descs/faces.desc
```

BBBS would search the descriptions for files from the files
c:/descs/faces.desc, instead of c:/pictures/faces/descript.ion.

Note how the root directories for the directories /graph/people and /graph have to be specified. Otherwise they would not show up in the directory listing.

Also, the description output can be formatted with the @n and @@ flags. @n generates a single line feed to the description and the @@ creates one '@'-character.

If a line in a filedir-file begins with a dot, then the filename after the dot (separated with a space) must exist or the directory entries after the dot-line will not be included to the directory tree. This feature can be used with removable media, such as a CD-ROM. You can have one directory for all your CD-ROMs as long as you define correct dot-lines to ensure that the disk in the CD-ROM drive is autodetected.

1.9 Setup of directories: the descript.ion-files

Descriptions for files in a File/4 directory are stored in a 4DOS-compatible descript.ion file. They can have a "hidden" attribute, but not read only, as BBBS needs to rewrite the file from time to time. The format of a descript.ion file is simple :

```
filename  description_and_flags
```

Description is simply the description for the file. If there is a description for a file that does not exist on the disk, then it is displayed in the directory listing, but is marked "offline" by BBBS. In the description, some flags can be used:

@r:group1{,group2{,...}}
Give READ access to the specified groups for this file. Members of groups with READ access can read the file, that is, they can download or type it.

@w:group1{,group2{,...}}
Give WRITE access to the specified groups for this file. Members of groups with WRITE access can write to this file. That is, they can for example delete it, if this has been enabled in BCFG4. Having a WRITE-access to a directory overrides the read-only-access on a single file.

@p:group1{,group2{,...}}
Make the file PRIVATE for the specified groups. Only the members of these groups can see this file, except for users that have a WRITE access to the directory that the file is stored in.

@@
Write a single '@'-character to the description at this point.

@n
Insert a new line to the description at this point.

@f

Make this file free. Free files can be downloaded by all users if they have at least READ access to the directory. Free files can be downloaded even by users without download access. You should make especially useful files, like virus scanners, off-line mail programs and the BBBS, free so that all users can download them immedately.

@ltruefile

The file is a link: the real file is stored at the directory pointed to by truefile. The original file should still exist (use 0-byte files). Note that truefile can also be a link to some other file. If you use a lot of links, remember to run
BLINKFIX
daily.

@adate

The date date should be used as the file's date, specified in yymmdd-format. This is used in description files referred to via a @1-flag in the
filedirg
-files.

@ssize

The file should be marked as size bytes long, no matter what. This is also used with the @1-flag.

There are two special files that can be used to format the directory list output. These files are "." and "..". You can specify multiple entries for them. The description for the file "." is displayed in the directory list as-is, without the dot. Files before and after the "."-file are sorted separately by
BFILSORT
. The description of the file ".." is displayed right after user changes to this directory, and again, without the ".." prefix.

Example:

```
.. The current BBBS version is 3.33 How.
.
. The main files are:
.
bbbs_d.rar The BBBS executables for PC-DOS. @d223 @f
bbbs_2.rar The BBBS executables for IBM OS/2. @d543 @f
bbbs_gen.rar The shared parts of BBBS. @d666 @f
.
. Some other files:
.
bag.exe The AmigaGuide reader for PC-DOS. @d50 @f
bbbshelp The BBBS help file in English. @d72 @f
```

1.10 Configuring external things: the external.bbb-file

The external.bbb-file is used to configure most things that can be considered "external". This includes
packers
,
external protocols

and some networking configuration .

The external.bbb-file is divided into sections. A section begins with the name of the section in brackets ('[' and ']' characters) and continues until another section starts. Like in all BBBS configuration files, empty lines and lines beginning with a semi-colon (';') are ignored.

You can also implicitly force a certain section be used only if a certain version of BBBS is run. This can be accomplished by naming the section [section.os]. os can be any of the following, depending on the version of BBBS you wish to use those definitions with:

OS2	BBBS/2, the OS/2 version.
NT	BBBS/NT, the Windows NT version.
DOS	BBBS/D, the PC-DOS version.
SUNOS	BBBS/SUM and BBBS/SOS, the SunOS versions.
IRIX	BBBS/IRX, the Irix version.
HPUX	BBBS/HP, the HP-UX version.
ULTRIX	BBBS/U, the Ultrix version.
SCO	BBBS/SCO, the SCO UNIX version.
SOLARIS	BBBS/SOL, the Solaris version.
UNIXWARE	BBBS/UW, the UnixWare version.
LINUX	BBBS/L, the 80386 Linux version.
FREEBSD	BBBS/FBI, the 80386 FreeBSD version.

Note that the OS-specific sections (such as [af_pack.os2]) must be listed before "global" sections (such as [af_pack]), as BBBS will use the configurations from the first usable section it encounters.

1.11 external.bbb: configuring packers

The packers BBBS uses are configured in the external.bbb-file, in four sections. In all sections one line defines one packer: line #1 for packer #1, line #2 for packer #2 and so on. For each packer, all sections must be defined. The character - in any line means that packer is disabled.

The sections used are:

[af_ext]

In this section, the extensions used by the packers are configured. The extensions specified here are also used to identify the packer used when the user information is listed in the user editor.

[af_ident]

This section defines an unique "fingerprint" for each packer. You need to specify the offset of the fingerprint (starting from 0) and of course the indent bytes, in hexadecimal, separated with a comma. If the indent bytes are found from the specified offset in any file, then that file is assumed to be packed with the packer with the corresponding indent bytes.

[af_pack]

This section specifies the OS command string used to pack a single file into an archive.

[af_unpack]

This section specifies the OS command string used to unpack a file or a set of files from the packet to the current directory.

For the af_pack and af_unpack sections, you can use the standard external.bbb meta characters to specify various things, like the name of the archive.

If you add or remove packers, you should also remember to edit the line 359 in your bbbstxt-files to reflect your packer setup. Otherwise your users might select packers that are not available. In the bbbstxt-file, the packer names are separated with a slash ('/'). To add a packer, simply write an appropriate name at the end of the list. If you want to disable a packer, then just remove the appropriate packer's name, but NOT the slash, unless you remove it also from your external.bbb.

Note! In all command lines you should use the directory separator of your operating system!

Example:

```
; Example of a single packer definition:
;
[af_ext]
zip
;
[af_ident]
00,504b
;
[af_pack]
c:\bbs\pack\zip.exe -9 -j %p %F
;
[af_unpack]
c:\bbs\pack\unzip.exe -j -o -C -s -q %p %f
```

1.12 external.bbb: configuring protocols

The file transfer protocols are configured in the external.bbb-file. For each protocol, there are three sections that need to be configured. In each section, the first line is for protocol #1, the second line for protocol #2 and so on. The '-' character in any line means that protocol is disabled. BBBS can also handle protocol numbers 1 to 9 internally, but you can use an external program for them too if you like.

The sections used are:

```
[t_name]

This section simply defines the names of the protocols. The only place they are  ↩
   shown is the user editor to identify the used protocol.

[t_download]

This section defines the OS command strings to launch the protocol to download ( ↩
   send to the remote user) a set of files, the names of which are in a specified  ↩
   file. The
               meta-character
                '%f' can be used to insert the name of this list file.

[t_upload]

This section defines the OS command strings to launch the protocol to upload ( ↩
   receive from the remote user) files into the current directory.

As with
               packers
               , you can use the
               external.bbb meta characters
                in the t_download and t_upload-sections.

If you add or remove protocols, you should also remember to edit the line 358 in  ↩
   your
               bbbstxt
               -files to reflect your protocol setup. Otherwise your users might  ↩
                  select protocols that are not available. In the bbbstxt-file,  ↩
                  the protocol names are separated with a slash ('/'). To add a  ↩
                  protocol, simply write an appropriate name at the end of the  ↩
                  list. If you want to disable a protocol, then just remove the  ↩
                  appropriate protocol's name, but NOT the slash, unless you  ↩
                  remove it also from your external.bbb.

Note! In all command lines you should use the directory separator of your  ↩
   operating system!

Example:

; An example to use the GSZ program instead of the internal ZModem:
;
[t_name]
ZModem
;
[t_download]
c:\bbs\prot\gsz.exe portx %b,%i %h"handshake cts "sz @%f
;
[t_upload]
c:\bbs\prot\gsz.exe portx %b,%i %h"handshake cts "rz
```

1.13 external.bbb: Network configuration

In the external.bbb-file, there is also several sections related to FidoNet technology networking configuration. They are:

- The nodelist section (section nodelist)
- The FidoNet echomail area setup (section echomail)
- The FidoNet nodes setup (section nodes)
- The file-echo configuration sections (sections tick.adopt and ticks)
- The node and NetMail remappers (sections node_remap and NetMail_remap)
- Badecho configuration for BCFG4 (section badecho)

Also, an arcane AGNET/QWK-style network can be used with BBBS. Setup for such a network is done with the external.bbb-section agnet .

1.14 external.bbb: The nodelist-section

FidoNet technology networks use nodelists to work out the destinations of echomail messages. A nodelist is a list of all (or some) systems in the network. In BBBS, the nodelists are listed in the external.bbb-file, under the section nodelist. Under this section all nodelists that BNC should compile are listed. After each name, an additional wildcard-specification for a nodediff (used by BNDIFF) can be specified. If the second parameter starts with an exclamation mark ('!'), then the nodelist specified is replaced by the file specified by the second parameter. An ampersand character ('&') works like an exclamation mark, but the second parameter must specify a compressed list.

Remember that full nodelists should be uncompressed, whereas nodediffs should be in compressed form.

In all nodelist-section entries wildcards are allowed.

Example:

```
[nodelist]
; A normal nodelist for foonet:
```

```
c:/bbs/nodelist/foonet.list
; lukki.lst in c:/bbs/nodelist is replaced by lukki.list in c:/inbound
; when there is one...
c:/bbs/nodelist/lukki.lst !c:/inbound/lukki.lst
; barnet.list is replaced with the one from the compressed file barnet.zip
; in c:/inbound when there is one...
c:/bbs/nodelist/barnet.list &c:/inbound/barnet.zip
; Assuming all nodediff.* files in c:/bbs/inbound/ are diffs for the nodelist
; below...
c:/bbs/nodelist/nodelist.* c:/bbs/inbound/nodediff.*
```

1.15 external.bbb: The echomail-section

The FidoNet network echomail areas are defined under the section echomail, in the external.bbb-file. This section is automatically updated by
BCFG4
when it saves the conference configuration, but you can modify it also by hand if you like. Note that your 'NetMail' area should NOT be added to this section.

The format of the echomail-section is simple: one line defines one echomail area. A single entry consists of six fields, separated with spaces. The fields are (from left to right):

```
group        A single character group ID for this echomail area. Any
             character can be used. Areas from different networks should be
             configured under a differing group ID. If you are running a
             host or a hub, the accesses of your downlinks are defined with
             these group IDs.

areatag      The area tag for this echomail area. The echomail is
             distributed between systems under this name.

bbbs name    The name of the conference to which messages to this echomail
             area should be written to. If the conference name has spaces,
             enclose the name in quotes. If you want to write them a
             conference with the same name as the areatag of this echomail
             area, use a single '-' character here. A '!' character in this
             field makes this area a passthrough area. For passthrough areas
             at least one add-aka should be defined (see below).

add akas     The list of akas that should be added for this area. BOGUS also
             adds your default aka (defined in BCFG4) for this echomail area.
             Note that you can specify multiple akas here. You should not
             write the same aka twice. For passthrough-areas at least one
             aka should be defined in this field.

flags        Here you can specify one or more flags for this area. Just list
             the flags you want to activate one after another. If you don't
             want to use any flags, use the '-' character only. Allowed
             flags are:

             S  Secure: incoming mail to this area is accepted only if the
                the area's uplink is listed in the export list.
```

Whenever possible, use this flag.

V Visible: area is visible in areafix listings even to nodes who don’t have access to it.

D Don’t perform any dupe checking for this area.

export The list of nodenumbers this echomail area should be exported to. You can list as many nodes as you like. You can use only net/node if the node is in the same zone as the previous node, or only node if zone:net is the same. If you add a ’>’-character in front of a nodenumber, then it is marked only for export. If you add a ’<’-character in front of a nodenumber, then it is marked only for import.

As with tick file-echoes, you can also specify add a description for the echomail area after all fields by prefixing it with a slash ("/") character. If you don’t specify a description here, then the one defined in BCFG4 is used. This way you can add descriptions also for passthrough areas.

Example:

```
[echomail]
; An example echomail-section definition:
; gr areatag        bbbs name           add akas    flags export
; -- -------------- ------------------- ----------- ----- --------------
B BBBS.ENGLISH      -                   -           S     47:1000/101
G GENERAL           "MAIN BOARD"        40:765/151  S     40:765/765 49
P TESTAREA          !                   40:765/151  S     40:765/765 49
```

1.16 external.bbb: The nodes-section

The nodes in a FidoNet technology network you want to poll from or that poll you are defined under the section nodes in the external.bbb-file. The format of this section is simple: one line defines one node. In a single entry there are nine fields, separated with spaces. From left to right, the fields are:

node The FidoNet address of this node.

spass The session password that is used with BBBS’s internal BackDoor mailer. If you want to run a secure system, you must specify both mail session password and packet password for all nodes.

apass BRoboCop (areafix) requests from this node must have this password in the subject of the message.

ppass The "packet-password"; a password stored in outbound packets going to this node. This same password has to be also in inbound packets from this node to make sure the packets are really from that node.

tpass The password for TICK files.

```
gr          The
                echomail
                 area groups that are available for this node for
            both reading and writing. Groups that are listed after the '!'
            character, including the group '!', can not be disconnected,
            only connected.

st          Specifies the flags for this node, as well as the status of
            outbound mail packets for this node. List the flags one after
            another or use the '-' flag if the node is to have no flags
            and "normal" status (mail is transferred regardless of the
            poller). Normally the '-' flag is used. Available flags are:

            C  The mail packets for this node are automatically crashmail.
            H  The mail packets for this node are hold: they are not
               polled anywhere. Instead, the node must pick up them.
            !  Do not route NetMails directly to this node.
            B  Do not use HYDRA protocol with this node.
            R  Do not use the EMSI handshake with this node.
            X  Do not use the xHYDRA protocol with this node.
            D  Do not use FTS-0001 handshake with this node.

pa          Specifies the packer that should be used with this node. It
            should be a number, pointing to the appropriate
                packer entry
                 in
            the external.bbb-file.

route       The routings for this node. By default,
                BOGUS
                 automatically
            routes all NetMail to this node directly to it. With this field
            you can specify more. You can list as many routings as you like.
            One routing can be either zone:*, specifying all nodes in
            that zone, zone:net/*, specifying all nodes in that net or
            even a complete 4D-address (zone:net/node.point). Also, if you
            add an asterisk, '*', in front of a routing, then BOGUS will scan
            your nodelist for all downlinks of the hub or region specified
            and route also their mails via this node.
```

```
Example:

; An example nodes-section:
;
[nodes]
; node      spass   apass   ppass   tpass gr  st pa route
; --------- ------- ------- ------- ----- --- -- -- -------------------
47:1000/101 privat  passw   -       fub   BCD H  3  47:* 27:47/* 2:20/0
2:222/222   sesonly -       -       -     -   !  3
2:222/0     boot    boot    boot    boot  BD  !  3
2:222/70    boot    boot    boot    boot  C   -  3  1:* 2:* 3:* 4:* 5:* 6:*
2:222/71    boot    -       boot    -     -   !  3
; hub route:
2:22/222    session foopass pktpass tick  B   C  3  *2:22/222
; region route:
```

```
2:222/69    example foopass packet  tick  B   C  3  *2:22/0
```

1.17 external.bbb: FidoNet technology file-echo configuration

Adopting files

FidoNet technology TICK-style file-echoes are configured with two sections in the external.bbb-file: the tick.adopt section defines the files BTICK should generate a TICK file for. For each file, three sequential lines under tick.adopt should be listed. The first line should contain the TICK areaname that should be used with the file. The second the file name, wildcards are allowed. Finally, the third line should contain a description for the file.

Please be careful with the tick.adopt-section as there is no way to check who originally sent the file for you!

Defining TICK file-echoes

The TICK file-echo areas are defined under the ticks-section. Under the ticks-section, one line defines one file-echo. One entry consists of seven fields, separated with spaces. The fields are, from left to right:

```
gr      A single character group ID for this tick area. Any character can be
        used as the group ID. Different types of file-echo areas and
        especially file-echo areas of different networks should be grouped
        under a differing group ID. If you are running a host or a hub, the
        file-echo area accesses of your downlinks are specified in the
        nodes-section with these group IDs, so be clear with them.

tag     The name of this file-echo area.

path    The directory to which incoming files for this area should be moved
        to. This directory must exist.

aka     The nodenumber that should be used with this file-echo area.
        You can also list many akas.

afl     The area flags for this area, listed one after the another. You can
        use multiple flags. See below for export flags as you can use them
        in this field, too. You can also use the '-' flag to disable all
        flags, and usually you should. The available area flags are:

        V  The area is visible for all nodes.
        Z  file_id.diz should not be used in this area.
        R  Don't check CRC of the incoming file.
        !  Passthrough, do not update
                descript.ion-files
                 accordingly.
        A  Append info to the announce message.
```

```
export  The node numbers to which this file-echo area should be exported to.
        You can list as many node numbers as you like. You can also use one
        or more export flags in front of the node number. The available
        flags are:

        @  This node is unlinked, no files should be sent to it.
        +  Do not send a tick file to this node, just forward the file.
        <  Accept files from this node, but do not send any.
        >  Only send files to this node.
        H  Mark files as hold, so that the node in question must pick up
           them.
        C  Mark files as crash, so that you will poll them to the node.
        N  Mark files as normal (the default).
```

After these fields the slash character ("/") and the description for the file-echo can follow, but are not mandatory.

Note that the password used with tick-files is configured in the

nodes-section
of external.bbb!

Examples:

```
[tick.adopt]


[ticks]
; gr tag   path         aka        afl export         desc
; -- ----- ------------ ---------- --- -------------- -------------------
A BBBS     d:/pub/bbbs/ 2:22/222   AH  <2:222/222 @42 /BBBS and utilities
V LUKKI    d:/pub/txt/  40:765/151 HS  40:765/765     /LukkiVerkko
```

1.18 external.bbb: The remapping features

With BBBS it is also possible to remap NetMail destinations and ↩ nodelist entries. The sections node_remap and NetMail_remap in ↩ the external.bbb-file are used to accomplish these things.

[node_remap]

The section node_remap can be used to add or override a
nodelist
entry. This is useful for example points that need only one or ↩ few nodelist entries and not the whole nodelist. One line of ↩ node_remap-section defines one remapping. The syntax of one ↩ node remap entry is:

```
,4d_nodenumber,sitename,location,sysop,phonenumber,flags
```

All fields should be self-explainatory. The phonenumber-field can have multiple phone numbers, separated with colons (':'). When BackDoor tries to dial this system, one phone number is selected at random. Note that the phone number conversion in BCFG4 is done to these phone numbers as well!

The flags-field is the standard FidoNet flags that are active with this system, separated with commas.

Example:

```
[node_remap]
; A Foobar-site with three phonenumbers that can be dialled:
; Note that nodenumber is in 4D-format!
,12:34/5.0,Foobar-site,Foobar_City,Joe_Hacker,555-4242:555-6969:555-5555,300
```

[NetMail_remap]

The section NetMail_remap can be used to override the sender, receiver and their node numbers. One line under NetMail_remap-section defines one remapping. The syntax is original#new where original is a regular expression matching the string in the following format:

```
sender,senders_4d_fidonet_address,receiver,receivers_4d_fidonet_address
```

new is the sender, receiver and FidoNet addresses the message should be remapped to, in the same format. In new, an empty field means that the original value should be kept.

For example, if you specify the following under NetMail_remap:

```
^.*,.*,dino-maintainer,2:22/222.666$#,,Kalle Soiha,2:22/222.0
```

Then all NetMails to the user "dino-maintainer" at node 2:22/222.666 should be directed to the user Kalle Soiha at node 2:22/222.

NOTE! If you are not ABSOLUTELY SURE that you know what you are doing, do NOT use NetMail_remap. It can do you more harm than good and the results will very likely not be what you want.

1.19 external.bbb: AGNET/QWK-style network configuration

BBBS also supports the very arcane AGNET/QWK style of networking, in which the off-line QWK and REP-packets are transferred to and from BBS's according to certain rules. To set up this kind of network in BBBS, the section agnet in external.bbb is used. Under this section, each line defines one remote system and/or user name, in the following format:

```
name,areafile,suffix,route_regexp
```

name is the name of the AGNET-user that should execute the polling. The areafile-parameter should be the name of the file remapping the QWK numbers to local area names, see below for more information. suffix should be the AGNET suffix that should be used for this system. route_regexp should be a regular expression matching the systems that mail should be routed to.

For example:

```
BCG BOX,agareas,@BCG,@
```

This will make the AGNET username into "BCG BOX". Users writing messages from this system will be, for example "Joe Hacker@BCG". Also, all incoming AGNET mail to this system will be routed.

In the file specified by the areafile-parameter, the QWK index numbers should be matched against the correct local conferences. In the file, one line defines one conference in qwk_number,local_name-style format.

For example:

```
15,Net.Chat
16,Net.Computers
17,Net.BBBS
...
```

It is highly recommended that you do not use AGNET networking unless there really is need for it, as the FidoNet style of networking is much more advanced.

1.20 external.bbb: The badecho settings for BCFG4

BCFG4 can autocreate conferences for such echomail areas that have inbound mail, but have no conference configured. This can be accomplished by running the BCFG4 program with the "bad"-parameter.

The badecho-section can be used to control the behavior of the automatic conference creator of BCFG4. If the badecho-section exists, then only the areas matching the criterias defined in the section will be created. Remember also that you must have at least one "unnamed" conference in your conference configuration or no conferences are added.

Note that when you define badecho-entries and run BCFG4 with the "bad" parameter, your echomail-section in external.bbb will be updated automatically, along with the conference configuration in BCFG4.

In the badecho-section, one line defines one criteria in the following format (items in curly brackets are optional):

```
uplink area {aka_list} {in_char out_char origin} group {area_prefix} export
```

uplink is the nodenumber of your uplink the echomail area must originate.

area is a regular expression

matching the names of the areas that can be created.

aka_list is a list of your akas that should be added to the messages in this echomail area. The first aka is used in BCFG4, the rest in the echomail-section of external.bbb.

in_char is the name of the character set that should be used when importing messages.

out_char is the name of the character set that should be used when exporting messages. This must be specified if you have specified also the in_char-field.

origin is the number of the origin line that should be used on the area. This must be given if you have specified inbound and outbound character sets.

group is the single character group ID that should be applied to the area. If the group is '-', then the area will NOT be created.

area_prefix is the prefix that should be applied to the area name. If there is a '!' character in the prefix, then the area will be created as a passthrough area.

export is a list of akas that the area should be exported to. The uplink is automatically added to this list, so you do not need to list it.

Example:

```
[badecho]
; For areas that originate from 2:210/27: do not generate areas starting
; with the string "BBBS". Generate all the rest, with the aka 2:222/0,
; origin line 0, group F, prefix "INT.". Export the area to 2:22/10 and
; 2:222/151.666:
;
2:210/27 ^BBBS -
2:210/27 . 2:222/0 IBM IBM 0 F INT. 2:22/10 2:222/151.666
;
; All areas originating from 2:22/222, beginning with "FOO" should be
; created as passthrough areas, export them to 2:22/10:
;
2:22/222 ^FOO P ! 2:22/10
;
; For all areas from 2:22/222, beginning with "BAR" use the akas 50:123/5
; and 69:100/12, export to 2:222/151.666, using origin line 5:
;
2:22/222 ^BAR 50:123/5 69:100/12 IBM IBM 5 B 2:222/151.666
;
; For all areas from 2:22/222, beginning with "BUZ" use the akas 50:123/5
; and 69:100/12, export to 2:222/151.666, using origin line 0 and the ISO
; charset for inbound messages, and the MAC charset for outbound:
;
2:22/222 ^BUZ 50:123/5 69:100/12 ISO MAC 0 Z 2:222/151.666
```

1.21 The meta characters in external.bbb-file

You can use following meta chars when configuring packers and protocols in the external.bbb-file. They are replaced with corresponding string or number.

```
        %e              Character 27, ESC, ^[.
        %r              Character 13, CR, ^M.
        %%              Character 37, "%".
        %p              Packet name to/from which files should be (un)packed.
        %f              File that should be sent/received or file to be
                        (un)packed to/from a packet.
        %F              Filename (%f), but with "*.*" converted to "*".
        %d              Filename (%f) without its path.
        %D              Filename (%d), but with "*.* "converted to "*".
        %c              In PC-DOS, OS/2 and NT-versions this is the
                        value of either COMSPEC or SHELL variables, checked
                        in that order. If neither is defined, then it will be
                        "c:\os2\cmd.exe" (under OS/2) or "c:\command.com".
                        Under UNIX versions, the order is SHELL, COMSPEC.
                        If neither is defined, then "/bin/sh" is used.
        %n              Value of the "cfgl.newdir" (
                bl_newdir
                ) variable.
        %N              User's name.
        %L              The current node number.
        %o              Comport number.
        %b              Comport base address (in hex).
        %i              Comport IRQ.
        %s              Comport real baud rate.
        %l              Comport baud rate.
        %a              Comport handle.
        %A              Comport device.
        %h"text"        If RTS/CTS handshake is enabled, text "text",
                        otherwise nothing.
```

1.22 Configuring BBBS groupmail functions: the alias.bbb-file

The file alias.bbb is used for two purposes. Firstly, you can create mailing lists so that you can easily send the same message to multiple receivers. The format of a mailing list entry in alias.bbb is as follows:

```
listname:fidonet_number1, user_name1,
         fidonet_number2, user_name2,
         fidonet_number3, user_name3
         ...
```

Everything should be typed on a single line in the alias.bbb-file. You can also create aliases for users in this way: just create a mailing list with just one user. For email-aliases, the FidoNet number should be your own FidoNet number.

The second purpose of alias.bbb is to define the receivers for comments users write with the global command "com". They are defined in a very similar manner to mailing lists; to define a comment receiver, use the string COM_* as the mailing list name. Replace * with the corresponding key the user should press when asked for a comment receiver. After the colon, you should write the user name or mailing list that the comment should be sent to. If the first character after the colon is a '<'-character, then the file specified after the '<' is shown when the choice is selected.

When you define comment receivers, remember also to correctly list the comment receivers in the precom-file in the menus-directory .

Example:

```
; An example alias.bbb-file
; This one defines aliases 'B' and 'Z' for the users Kim Heino and
; Tapani Salmi at FidoNet address 2:22/222, a mailinglist called BZ,
; including the same users and three comment receivers:
;
B: 2:22/222, Kim Heino
Z: 2:22/222, Tapani Salmi
BZ: 2:22/222, Kim Heino, 2:22/222, Tapani Salmi
COM_B:Kim Heino
COM_Z:bz
COM_F:<c:/bbs/menus/access
```

1.23 Configuring Internet access in BBBS: the inet.bbb-file

All Internet access is configured in the file inet.bbb. Like the external.bbb-file , it is divided into different sections. The sections used are:

```
[telnet]
[finger]
[rlogin]
[hunt]
[ftp]
```

These five sections are used to configure the limitations for outgoing connections. The telnet-section is used for limitations concerning outgoing telnet-targets, the finger-section for finger targets and so on. Connections can only be made to the destinations specified in the appropriate inet.bbb-section. The syntax for an entry is:

```
destination:group1{,group2{,group3...}}
```

Before any user can use any Internet services, he must first be a member of the group with the name of the service: the group telnet for telnet-connections, the group ftp for ftp-connections and so on. Also, the user’s destination must be listed in that service’s inet.bbb-section, and be a member of one of the groups defined for that destination.

Only the groups listed can use the service specified in the section, and only to targets matching the regular expression destination.

For example, to allow only members of the group blood_men to make an ftp connection to the site ftp.pirasat.org, but allow everyone to ftp into sites with the .fi domain suffix, you would write into inet.bbb:

```
[ftp]
^ftp\.pirasat\.org$:blood_men
\.fi$:all
```

[aka]

This section is used to define shortcuts for destinations for other Internet services. The syntax is simple:

```
alias1:real1
alias2:real2
...
```

For example, the definition b:Kim.Heino@utu.fi would mean that the simple destination b would be expanded into Kim.Heino@utu.fi.

[aliasout]

This section is used to define outgoing Internet-aliases for SMTP and NNTP mail transfer and BOGUS import, when the gating option is enabled. The syntax is:

```
inet-address:original
@domain:@original
```

The first form is used to alias one local or remote user to a different inet-name. The second is used to alias a whole domain to a different name. The remote domain from BCFG4 is implied before checking for the entries in the aliasout-section.

An example:

```
; To define the address "b@fix.no" be an alias for the local user
; "Kim Heino":
b@fix.no:Kim Heino
; To direct all mail going to domain bcgbox.bbbs.eu.org to
; p0.f222.n22.z2.fidonet.org:
@bcgbox.bbbs.eu.org:@p0.f222.n22.z2.fidonet.org
```

[aliasin]

The aliasin-section works like the aliasout-section, but defines incoming Internet-aliases. For example:

```
; To define all mail sent to postmaster@fix.no to end up to the user
; Torfinn Nome:
postmaster@fix.no:Torfinn Nome
; To alias the whole domain bcgbox.bbbs.eu.org:
@bcgbox.bbbs.eu.org:@p0.f222.n22.z2.fidonet.org
```

[fingerin]

This section is used by the BBBS finger program, bbbs.fingerd.exe. In it the remote finger shortcuts are defined. The syntax is simple:

```
name:user name
name:<filename
```

The first form is a normal user alias, the second one can be used to show any text file to the person fingering the name.

An example:

```
; To make fingering of the user "B" to display the info of Kim Heino:
b:Kim Heino
; To display the file "c:/manual" to the person fingering the
; user "rtfm":
rtfm:<c:/manual
```

[listin]

This section can be used to allow remote users send email directly to a conference. The syntax is:

```
aliasfrom,aliasto:conference
```

All mail sent to aliasto by aliasfrom will be written to the conference named conference. Either of the fields can be empty, so just the other field is checked. For example, to write all mail sent to bbbs-list@fix.no to the conference BBBS.Chat, you would write:

```
,bbbs-list@fix.no:bbbs.chat
```

And, to write all mail coming from listserv@foo.edu to the conference foolist, you would write:

```
listserv@foo.edu,:foolist
```

These aliases are used when incoming SMTP messages are received and after the aliases in the aliasin-section have been processed. Remember that both aliases have to be full Internet addresses!

```
[bounce_from]
[bounce_to]
```

These two sections define the addresses that mail from/to should be bounced back to the sender. Under both sections, one entry can be either a full Internet address or just @domain. For example, to bounce all mail from coolcorp.com and all mail sent to nomail@mysite.net, you would write into inet.bbb:

```
[bounce_from]
@coolcorp.com

[bounce_to]
nomail@mysite.net
```

1.24 Configuring the BBBS Chat System

BBBS has a very powerful, channel -based chat system with feelings , familiar from conversation systems like IRC and different MUDs. Basically, chat channels are "discussion rooms". In every channel there can be a number of users chatting at the same time. All messages that are sent to a channel by a person are seen by all other members of the channel the message was sent to.

The channels can be either static or temporary. Static channels always exist, even if there aren't be any members on them. Temporary channels exist only as long as they have members. Static channels are defined in the file channels.bbb, residing in the feelings-directory specified in BCFG4 . Definitions for temporary channels reside in the file channel2.bbb, but you needn't worry about it, as BBBS will automatically update it when necessary.

1.25 Configuring chat channels

Each line of the channels.bbb-file defines one static channel, in the following format:

```
#channel_name channel_description
```

All channel names begin with the '#'-character. The description should be a simple description of the topic or topics that should be discussed in the channel. In the description, you can use some flags to define the behavior of the channel:

```
+a       Make the channel auto-invite. All users are automatically
         made members of a channel with this flag.
+igroup  Make the channel inaccessible to all but members of the
                group
                        group.
+ogroup  The group group is given channel operator access to this
         channel. Channel operators can use the /kick command to
         remove unwanted users from a channel and change the topic of
         channels that don't have a static topic.
+t       The topic of the channel is "locked" and can only be changed
         by channel operators. You don't need to specify this flag for
         static channels, as topic is always locked for them.
```

For example:

```
#chat +osysops +a General public chat
#simka +isimka Private simka channel
```

The bans for different channels are defined in the file banned.bbb in the feelings directory. The format of this file is simple: one line defines one ban in the following format:

```
#channel_name:user_name
```

So, for example, to ban the user "Samuli Suominen" from the channel #vanessa_rulez, you would write into banned.bbb:

```
#vanessa_rulez:Samuli Suominen
```

You can also have word bans for different channels. These are specified in the file wordban.bbb. In this file, each line is a regular expression that defines one word ban. Every time a user sends a chat message to a channel, a string in the form #channel:message is created, with channel being the channel the message is sent to and message the actual message. The string is then matched against all the definitions in the wordban.bbb-file. If any of the expressions in the file match the string, then the user sending the message is automatically banned from the channel the message was sent to.

For example, specifying ^#chat:Call.*BBS in wordban.bbb would autoban everybody sending a message starting with "Call" and containing the string "BBS" to channel #chat.

1.26 Configuring chat feelings

The BBBS chat has also the possibility to use various feelings, familiar from different MUDs. They can be used to give some "flavor" to the chat. There are a lot of feelings in the BBBS distribution package, but you might want to create your own as well. The feelings are contained in data files in the feelings-directory specified in BCFG4 . They are sorted into files by their first character, so that the feeling abduct is in the file a.dat, the feeling teleport in t.dat, and so on. Note that the data files are case sensitive!

There are three types of feelings: ones that require a focus (another user) for them to be used, ones that work without a focus and ones that can be used either with or without a focus. They are defined with the keywords wfocus, wofocus and wwofocus, respectively.

In the data files, the feelings are listed in blocks. Each block begins with a keyword, specifying what kind of a feeling will be defined next. Then a colon and the name that should be used to activate this feeling. In the name, the necessary part should written in upper case. That is, if you define a feeling called REMind, then it can be typed by writing just "rem". The feeling definition block consists of a starting curly bracket ’\{’, one or more sockets and a closing curly bracket ’\}’. Note that there has to be at least one space before the closing bracket. Empty lines are ignored, as well as lines beginning with the pipe character ’|’.

There are many sockets that can be used. Which ones are required vary with the type of the feeling. The generally required sockets are the message sockets:

```
selfwf: Message displayed to the person emitting the feeling, with
        focused feeling.
allwf:  Message displayed to all, except the person emitting the
        feeling and the focus, with focused feeling.
focus:  Message displayed to the focus.
selfof: Message displayed to the person emitting the feeling, without
        focus.
allof:  Message displayed to all, except the person emitting the
        feeling, without focus.
```

The message sockets contain the actual feeling message to be displayed. It can be up to 250 characters long and it will be wordwrapped automatically when printed to screen. You can use ansi-codes in the message to override the color-socket (see below), but it will screw up the wordwrapping.

Following variables can be used in the message string:

```
Var   Expands to:
=========================================================================
$N    The nick of the user emitting the feeling.
$F    The nick of the focus.
$A    The grade of the feeling. See below for more about
      grades. This should follow the verb to get "correct" English.
$B    The extension of the feeling. This should normally be at the end
      of the feeling.
```

The following depend on the user’s sex:

```
$G    "his" or "her"
$P    "him" or "her"
$E    "he" or "she"
```

In lower case ($g, $p, $e), they depend on the focus’ sex.

Other sockets that are not required in any feelings, but can still be defined, are:

author: Name of the person who added this feeling.
classes: The classes this feeling belongs to, see below for a list.
color: The default color of this feeling: yellow, green, blue, red, cyan or white. First letter is sufficient.
ext: The default extension of this feeling.
grade: The default grade of this feeling. See below for more on grades.
group: A
regexp
groupname, defaults to "all". Only members of the

group
specified here can use this feeling.
karma: "Karma" value of this feeling, from -10 (very evil) to +10.
standing: The political value of this feeling from -10 (extreme left) to +10 (extreme right).

List of feeling classes:

affectionate	(compassionate)
aggressive	
audible	(generates sound)
burgeois	(capitalistic)
friendly	
funny	
inferior	(the user of the feeling is inferior to others)
malignant	(evil, in other words)
negative	
positive	
red	(communist, remember also the standing-socket)
superior	(the user of the feeling is superior to others)
tactile	(physical, touching)
vulgar	
white	(monarchist or bourgoise, remember also "standing")

Grades

The grade of the feeling describes manner in which the feeling is executed. Such as shamelessly, gracefully etc. The users can add any grade they like to a feeling, but you can specify shortcuts for grades in the *.gra-files in the feelings-directory. Like the feeling data files, the grades are separated into files by their first letter: shamelessly goes into s.gra, gracefully into g.gra and so on.

The syntax of grade files is simple, first the name of the grade (with the necessary part written in upper case), then a colon and then what it should be expanded to. See below for an example.

Examples:

```
| Some example feelings.
|
wfocus: ABDuct
  {
    selfwf: You $A abduct $F $B.
    allwf: $N $A abducts $F $B.
    focus: $N $A abducts you $B.
    grade: shame
    group: sysops
    karma: -3
  }

wwofocus: AGRee
  {
    selfof: You $A agree $B.
    allof: $N $A agrees $B.
    selfwf: You agree $A with $F $B.
    allwf: $N agrees $A with $F $B.
    focus: $N agrees $A with you $B.
    grade: whole | wholeheartedly...
  }

wofocus: BLInk
  {
    selfof: You $A blink $B.
    allof: $N $A blinks $B.
  }


| An example f.gra:
|
FATherly:in a fatherly manner
FAIthfully:faithfully
FOOlishly:foolishly
FRUstration:in frustration
FONdly:fondly
FEVerently:feverently
FINally:finally
```

1.27 Setting up the color palette: the COLORS.BBB-file

The color schemes for the u pal-command are set up in the colors.bbb-file in the BBS-directory. In addition to the schemes defined here, users can create their own customized schemes with the palette editor.

The syntax of the file is simple: first the scheme name, then a colon and then 52 characters defining the colors to be used. The characters represent different items as follows:

```
Position Defines the color of...
==========================================================================
   1.    normal text in the message header.
   2.    user names in the message header.
   3.    subject lines in the message header.
   4.    normal text in messages.
   5.    quoted text in messages.
   6.    double-quoted text in messages.
   7.    net information (origin/tear/kludge lines) in messages.
   8.    the directory line in file listing.
   9.    the topmost header lines in file listing.
  10.    sub-directories in file listing.
  11.    normal files in file listing.
  12.    free files in file listing.
  13.    file links in file listing.
  14.    the first line of the file description in file listing.
  15.    the rest of the file description lines.
  16.    the numeric data (amount of downloads etc.) in file listing.
  17.    chat messages sent by the user him/herself.
  18.    incoming public chat messages.
  19.    incoming private messages.
  20.    informative messages (login/logout, entered message...) in chat.
  21.    the chat prefix string (the string displayed before the message).
  22.    incoming public chat messages addressed to the user.
  23.    the header help lines in "Join"-command.
  24.    the highlighted characters in "Join"-command.
  25.    a selected option in "Join"-command.
  26.    an unselected option in "Join"-command.
  27.    local public area names in "Join"-command.
  28.    public network area names in "Join"-command.
  29.    local private area names in "Join"-command.
  30.    network mail area names in "Join"-command.
  31.    area descriptions in "Join"-command.
  32.    header lines in "SHow"-command.
  33.    total separator line in "SHow"-command.
  34.    conference names in "SHow"-command.
  35.    numeric information in "SHow"-command.
  36.    number specifying the amount of personal messages in "SHow".
  37.    user info display header.
  38.    user info display texts.
  39.    user info display numeric information.
  40.    header of "Who"-display.
  41.    nodenumber for active nodes in "Who".
  42.    nodenumber for inactive nodes in "Who".
  43.    speed for active nodes in "Who".
  44.    speed for inactive nodes in "Who".
  45.    status text for active nodes in "Who".
  46.    status text for inactive nodes in "Who".
  47.    "when" time for active nodes in "Who".
  48.    "when" time for inactive nodes in "Who".
  49.    nickname for active nodes in "Who".
  50.    nickname for inactive nodes in "Who".
  51.    name of the caller for active nodes in "Who".
  52.    name of the caller for inactive nodes in "Who".

The character can be a number (0-9) or a letter from a to f. The characters
represent different ANSI attributes as follows:

  Char   Color
================================
   0     Black
   1     Red
   2     Green
```

3	Yellow
4	Blue
5	Magenta
6	Cyan
7	White
8	Black with bold
9	Red with bold
a	Green with bold
b	Yellow with bold
c	Blue with bold
d	Magenta with bold
e	Cyan with bold
f	White with bold

1.28 bbbstxt-files: language configuration

With BBBS, it is possible to use up to ten different language configurations. The language configuration is done with the bbbstxt-files in the BBS root directory. The extension of the file specifies the language number it describes. An extensionless bbbstxt-file contains the "default" language, usually English. With the distribution package, four languages are supplied: English (bbbstxt), Finnish (bbbstxt1), Swedish (bbbstxt2) and Norwegian (bbbstxt3).

To edit the supplied languages or create new ones, you can use your favorite ASCII-editor. Each bbbstxt-file should contain 713 lines. Each line describes one string that is displayed somewhere in the system. These are (of course) the same in every bbbstxt-file. When the strings are displayed should be pretty self-explainatory.

When you are editing the bbbstxt-files, note that the lines should always contain the original amount of modifiers, characters prefixed with a percent sign ('%'). The modifies are familiar from the C-language, so that '%u' specifies a number, '%s' a string and so on.

The bbbstxt-files also contain the names of the commands for the different menus. If you want to, for example, use the error-script to replace a command, then you should remove the original command from all your bbbstxt-files. The error.bz-script that came with the distribution package also offers an easier way to create new commands and replace old ones, the commands.bbb-file. See the example script for more details about it.

Remember that if you really want to have multiple languages, you should code also your scripts so that they support all your languages!

1.29 An overview of the files BBBS uses

Besides the most obivious configuration files , BBBS uses and creates a lot of other files that are used. The directory names given here are not mandatory, they can be anything you like, but you should have them like this for easier reference in case of trouble.

By default, BBBS creates the following directories and files to the BBBS root directory:

bad/	Badecho directory.
feelings/	Directory containing feelings and chat configuration.
grab1/	Grab directory for node one.
hold1/	Filearea "hold" directory for node one.
inbound/	Directory containing inbound FidoNet mail.
mail/	NetMail directory for file attaches.
main/	BBBS main directory, containing the messagebase etc.
menus/	BBBS menus directory, containing all the menus.
misc/	Miscellaneous files.
new1/	Temporary upload directory for node one.
outbound/	Directory containing outbound FidoNet mail.
pub/	Public file directory, used as an example in filedirg.000.
scripts/	Directory containing all the BBBS scripts.
secure/	Directory for storing secure incoming FidoNet messages.
tmpin/	Directory for temporarily storing inbound packets.
tmpout/	Directory for temporarily storing outbound packets.
voice/	Directory for storing voice messages.
work/	A general-purpose working directory.
alias.bbb	The file holding the BBBS groupmail configuration.
bbbscfg4.000	Global configuration used by all nodes
bbbscfg4.x	Local configuration for node x.
bcfg4.exe	The configuration program.
external.bbb	Configuration file for protocols, packers and some FidoNet stuff.
filedirg.000	The global file directories

```
                 .
filedirn.x        The local file directories for node x.
groups            The
                 group definition
                  file.
```

1.30 The files in the BBBS root directory

```
                 The following files should reside in your BBBS root directory:

alias.bbb           The file holding the
                 BBBS groupmail configuration
                 .
bag.exe             The AmigaGuide reader. bag.exe will have different
                    functions depending on what name it is run with. When run
                    as ag2html.exe, it will convert AmigaGuide files to HTML
                    format. As ag2doc.exe, it will convert AmigaGuide files
                    to a normal ASCII format, with emphasis characters. As
                    ag2txt.exe, it will convert AmigaGuide files to normal
                    ASCII text.
bbbs.001            A library file.
bbbs.002            Another library file.
bbbs.exe            The main executable.
bbbs.fingerd.exe    The BBBS finger daemon.
bbbs.key            Your personal registeration key. Keep good backups
                    on this one!
bbbs.smtpd.exe      The BBBS SMTP daemon for incoming Internet e-mail.
bbbs.telnetd.exe    The BBBS telnet daemon for telnet logins.
bbbscapi.dll        A dynamic link library (
                 OS/2 version
                  only).
bbbscfg4.000        Global configuration used by all nodes
bbbscfg4.x          Local configuration for node x.
bcfg4.exe
                 The configuration program
                 .
bbbstxt             Language file for language 0. The bbbstxt files hold most
                    of the text that BBBS will display to the screen. You can
                    modify it as you like.
bbbstxtn            Language file for language n.
bcfg4.exe           The
                 BCFG4 configuration program
                 .
brobo.wht           The Artificial Stupidity file for BRoboCop.
bterm.log           Log file created by
                 BTERM
                 .
bterm.pho           The BTERM phonebook.
colors.bbb          The
                 color scheme configuration
                  file.
external.bbb        Configuration file for
                 protocols
                 ,
                 packers
```

```
                  and some

                 FidoNet stuff
                 .
filedirc.x          The
                 file directories
                  for the conference x.
filedirg.000        The global file directories.
filedirn.x          The local file directories for node x.
groups              The
                 group definition
                  file.
ripc.exe            A program to compile RipScript files into BBBS RIP
                    menu files.
sysop.gui           What you are reading right now.
hello.zad           The default
                 voice greetings file
                 .
hippo.wht           When a user sends a message to Hippo, she will reply with
                    a random line from this file.
inet.bbb            File containing
                 Internet access configuration
                 .
jargon.txt          An optional online hacker's dictionary (The Jargon File).
                    The command q ja will need this.
up_scan.bat         For all uploads, BBBS will automatically run this file.
up_scan.cmd         Like up_scan.bat, but for
                 the OS/2 version
                 .
up_scan.sh          Like up_scan.bat, but for
                 UNIX versions
                 .
_mg                 The MG editor startup file. See the file mg.tex for MG
                    documentation.
```

1.31 The files in /bbs/main directory

```
                 The main directory contains the most important BBBS database files ↩
                     . You
should keep good backups on this directory!

00000000.hdr     Headers for the messages in conference 0.
00000000.txt     Message texts for conference 0.
0000000a.hdr     Headers for the messages in conference 10.
0000000a.txt     Message texts for conference 10.
accounts.dat     Account records for accounting.
badpoll.dat      Information on unsuccessful polls.
bbbsdown.dat     Records kept about up/downloaded files. If missing, BBBS
                 creates it again.
bbbshi.dat       Hippo messages for the users.
bbbsset.dat      Aliases and sets created by users.
bbbsstat.dat     Statistics file.
bbbsuser.dat     The main userfile. Backup this often!
bbbsuser.idx     Index for bbbsuser.dat. Autocreated by BBBS on startup.
btickube.dup     Duplicate file database for
```

```
                BTICK
                .
busypoll.dat    Information on polls on busy nodes.
bzll.dat        Library file.
chatlog.1       Logfile from last chat with local user on node 1.
confcfg4.dat    Conference setup.
confusr4.dat    Database containing the lastread-pointers and joined
                conferences for the users.
filedir*.idx    Index for fileareas. Can be recreated with
                BBBS BFILEIDX
                .
grabdupe.dup    Duplicate offline reply packet database. Used to check if
                user tries to upload same reply packet more than once.
log1            The log that is kept for each node. The name can be
                changed in BCFG4.
okfiles         Only files listed in this file may be uploaded.
okuser          Only users listed in this file may register as a new
                user.
nodelist.idx    Nodelist index, created by
                BNC
                .
statzero.dat    Statistics file.
sysnote         Notes that BBBS writes to the sysop (newusers, changed
                names, uploaded files, etc)
tossdupe.dup    Database to keep track of duplicate outgoing echomail.
trashcan        Opposite of okuser, users listed in this file may not
                register.
trashfil        Opposite of okfiles, files listed in this file may not be
                uploaded.
```

1.32 The files in /bbs/menus directory

The files in /bbs/menus directory can be recognized by their extension.

Extension	When user sees this file
<none>	Language 0, no ANSI, novice
.1	Language 1, no ANSI, novice
.gr	Language 0, ANSI, novice
.gr1	Language 1, ANSI, novice
.r	Language 0, RIP, novice
.r1	Language 1, RIP, novice
.x	Language 0, no ANSI, expert
.x1	Language 1, no ANSI, expert
.xg	Language 0, ANSI, expert
.xg1	Language 1, ANSI, expert
.xr	Language 0, RIP, expert
.xr1	Language 1, RIP, expert

```
areainfo.000    Displayed by "r ai" command on the conference zero.
bang            Russian roulette bang-file.
bbbshelp        File with the help displayed with "h".
bftpmenu        Menu used for FTP-commands.
```

```
birthday         The list of those who celebrate their birthday.
brobo_af         Header for BAllFix replies.
brobo_cf         Header for AreaFix replies and connection notifies.
brobo_ff         Header for FileFix (
                 BTICK
                 ) message.
brobo_fr         Header for file request replies.
brobo_nf         Header for NameFix replies.
bull             The bulletin menu.
bull0            Welcome-bulletin (displayed when the user logs in).
bull1            Bulletin number one.
c019b            Bulletin menu for conference 19.
c019b0           First-join-bulletin for conference 19.
c019b1           Bulletin one for conference 19.
c019j            Always-join-bulletin for conference 19.
charmenu         Menu used for character choice command.
chathelp         The help screen for groupchat.
chatmenu         Menu used for chat command.
colors01         The first screen of the palette editor.
colors02         The second screen of the palette editor.
confer           The list of the conferences. Delete it to use
                 BBBS internal comments.
doormenu         Menu used for door command.
edithelp         The help screen for the full-screen editor.
event            Displayed when user is thrown out after login because
                 of an event.
feelmens         Extended chat help, SysOp only.
feelmenu         Extended chat help.
fil4mens         Menu used for file command, SysOp only.
fil4menu         Menu used for file command.
fingerd          Header displayed before ’finger’ information.
flaghelp         Help file for file flagging.
formmenu         Menu used for choosing archiving method.
getlost          This file will be shown to user with "getlost" status.
globmenu         Menu  used  for  global commands, will be shown after the
                 listing of a main, read, file, utility or sysop menu.
grabdupe         Displayed when user has uploaded a reply packet
                 containing already sent messages.
grabmenu         Menu containing the different grab format choices.
hello            Hello-message for the new user.
helplink         Index file for bbbshelp.
huntmenu         Menu used for search command.
joinhelp         Help file for joining.
langmenu         Menu used for language choice.
lostpass         This file will be shown to user who has lost password.
mainmens         Menu used for main command, SysOp only.
mainmenu         Menu used for main command.
markmenu         Menu used for mark command.
outbmenu         Menu used for outbound manager.
postlog          The file will shown after the ‘g y‘ command.
postreg          The file will be listed after a user has registered.
prechat          The file will be shown before reason for SysOp chat is asked.
precom           The file will be shown before ‘comment‘ command.
precreg          The file will be shown before closed system password question.
predesc          Will be shown to user before asking descriptions for files.
predown          The file will be shown before a user downloads a file.
prefing          Displayed before asking a finger destination.
```

```
preftp          Displayed before asking a ftp destination.
pregrab         This file will be shown before the scratchpad is sent to the
                user using the grab command.
prehunt         Displayed before asking a Hunt server location.
prelog          The file that is listed before the user logs in.
prepreup        The file shown before asking the filename in upload command.
prereg          The file is shown before a user is registered.
prerlog         Displayed before asking a rlogin destination.
preteln         Displayed before asking a telnet destination.
preup           The file will be shown prior to an upload.
protmenu        Menu used for protocol choice.
readmens        Menu used for read command, SysOp only.
readmenu        Menu used for read command.
rmodmenu        Menu used for read mode selection.
termmenu        Menu used for terminal type selection.
utilmens        Menu used for utility command, SysOp only.
utilmenu        Menu used for utility command.
```

1.33 The files in /bbs/temp directory

```
                areatoss.0      Areatoss infofile for all nodes, created by BBBS  ↩
                   after
                user logs off. If he/she wrote messages to FidoNet
                areas, bit according to conference number will be
                set in this file. BBBS BOGUS A will read this
                to scan areas written to, for exporting mail.
areatoss.1      Areatoss infofile for node one, created by BBBS after
                user logs off. If he/she wrote messages to FidoNet
                areas, the number(s) of the conference(s) will be
                written into this file. For node two it has extension 2.
bbbsmsg.1       Nodemessage file for node one.
bbbsnode        Node status information for the system.
bbbsrun.1       Flagfile for node one, BBBS is now running on that node.
bbbsrun.a       Flagfile for
                BSMTP
                . When this file exists, a BSMTP session
                is running.
bbbsrun.b       Flagfile for
                BNNTP
                . When this file exists, a BNNTP session
                is running.
bbbsrun.c       Flagfile for
                BOGUS
                . When this file exists, BOGUS is running.
bbbsrun.d       Flagfile for
                BPOP
                . When this file exists, BPOP is running.
feelings.1      Feelings-file for node one.
transfer.1      Transfer queue for node one.
```

1.34 The files in /bbs/misc directory

The /bbs/misc directory contains various miscellaneous files:

bbbsdef.h The C-language header file containing the structure of all BBBS data files. If you want to do some programming, you can use this file as a reference.

mg.tex The complete documentation for the BBBS's internal MG editor, in LaTeX format.

msgview.c A C-language example on how to read the message text from the BBBS messagebase.

setpass.c A C-language source code for changing the SysOp's password - for emergencies.

1.35 The BBBS command line functions

The standard BBBS calling convention is:

bbbs.exe [com] [node]

Where node is the node number to start this BBBS session into, and com the comport number. This convention should suffice for most BBBS configurations. However, there are different versions of BBBS for various OS's. Depending on which of them you run, you can also use some other parameters or have a completely different calling convention.

BBBS offers also some other services that can be started from the command line. To use them, BBBS can also be invoked as:

bbbs.exe [command] {parameters}

Command can be any of the ones listed below. Parameters vary from command to command.

Command Function

==

BAADD
An account management tool.

BCONF
Display statistics on conferences.

BCSTAT
Display statistics on conference feeds.

BDATE
Update the birthday list.

BFILEIDX
Create an index file of the file directories.

BFILSORT
Sort the
file directories

```
.

BFTP
      Batch FTP.

BHATCH
    Hatch the specified file into the specified file-echo.

BKILL
     Kill users with criterias.

BLINKFIX
  Update the datestamps of file links.

BLIST
     Make an ASCII format file list.

BMKILL
    Kill messages with criterias.

BMSG
      Import/export FidoNet mail.

BMT
       A FidoNet message tracker.

BNC
       Compile the
nodelist
(s) specified in
external.bbb
.

BNDIFF
    Update a nodelist from a NODEDIFF.

BNEWF
     Make an ASCII format list of new files.

BNMSG
     Send a node message.

BNNTP
     Exchange netnews with NNTP servers.

BOGUS
     B's Original Garbage Unicast System (FidoNet mail import/ ↩
        export)

BOK
       Make an OKFILE.TXT file for use with the FrontDoor mailer.

BOM
       Command line interface to the BBBS outbound manager.

BPC
       Pack a conference with criterias.
```

```
BPOP
      Receive mail from POP3 server.

BPUS
      Pack the user database.

BRSA
      Generate new RSA keys.

BSETPACK
  Pack the environment variable settings file.

BSMTP
     Exchange email with SMTP servers.

BSTAT
     Show a statistics list.

BTICK
     Process he incoming TICK files from file-echoes.

BTIME
     Adjust the time with a timeserver.

BTXT2BBS
  Convert a text file to a message.

BWHO
      Show who is online at the moment.
```

1.36 BBBS BAADD - An account management tool

```
Usage: bbbs.exe baadd [amount] {account}
```

BAADD is used to edit the amount of money on different accounts. Amount is the value to add to the account(s): using negative values will deduct money from the specified account(s).

The account-parameter is a
regular expression
specifying the accounts
to change, default is all accounts. If it is not specified, then only accounts without credit will be changed.

1.37 BBBS BCONF - Display statistics on the conferences

```
Usage: bbbs.exe bconf
```

BCONF displays some status information on the active conferences. The output will look something like this:

```
  A# Name                                First  Last    kB MIFPN    Age    M#
--------------------------------------------------------------------------
   0 Netmail                                 1   270   159   FP  970410     1
   1 SysOp News                              1    12     2  I  N 961017    11
   2 Post Office                             1   228    67 MI P  970406    73
   3 Administrative                          1    36    10       961228     4
   4 Resume                                  1    48    50 MI  N 970307    73
   5 General Chat                            1   133    47  I    970411    68
```

```
Descriptions for the different fields:

A#     is the area number.
Name   is the area name.
First  is the number of the first active message in the conference.
Last   is the number of the last message in the conference.
kB     is the size of the message area, in kilobytes.
MIFPN  M - area is a "must": everyone must be a member.
       I - area is auto-invite: new users are automatically members.
       F - area is an echomail area.
       P - area is for private messages.
       N - replies are not allowed in the conference.
Age    is the datestamp of the last message in the conference.
M#     signifies how many members there are in the conference.
```

1.38 BBBS BCSTAT - Display statistics of conference feeds

```
                Usage: bbbs.exe bcstat [area_regexp] [date] {fido}
```

BCSTAT displays the amount of messages received to the areas matching the regular expression area_regexp since date. date can be any standard day, like 960831, or -10, meaning ten days before the current date. If the optional parameter "fido" is added, then the area names that are output are their FidoNet echomail names. By default, areas are listed with their local area names.

Examples:

```
bbbs.exe bcstat ^bbbs\. 962511
bbbs.exe bcstat ^bbbs\. -10 fido
```

1.39 BBBS BDATE - Update the birthday list.

```
                Usage: bbbs.exe bdate
```

BDATE will update the file birthday in your global menu directory , which is shown every time a user logs in. This command should be run every day.

1.40 BBBS BFILEIDX - Create an index file of the file directories

```
                Usage: bbbs.exe bfileidx {D} {O}
```

BFILEIDX scans through your file directories and creates an index file to speed up file searches and such. Also, it reports of errors in the file areas, such as too long description lines. For better performance, this command should be run every day. If you decide not to run it daily, then do not run it at all.

The optional 'D' parameter gives a report of files with no description. The 'O' parameter gives a report of files which are "offline" (specified in the

```
                descript.ion files
```

, but do not exist in the disk).

1.41 BBBS BFILSORT - Sort the file directories

```
                Usage: bbbs.exe bfilsort
```

BFILSORT sorts your file directories (the

```
                descript.ion files
```

). The entries before and after the dot entries are sorted separately. If you want to have nicely sorted file directories, run this command every day.

1.42 BBBS BFTP - Batch FTP

```
Usage: bbbs.exe bftp [commandfile] [get-dir]
```

BFTP can be used to use BBBS's internal FTP routines in batch mode. The parameter commandfile should be the search path of a file containing the commands to execute, each on their own line. get-dir should be the directory the FTP command 'get' should store the transferred files to.

Note that the first line of commandfile should contain the address of the FTP server that should be connected to. Also, the last command should be "quit", otherwise BFTP will revert into interactive FTP mode.

An example of a commandfile:

```
ftp.bbbs.net
anonymous
myname@myhost
cd /pub/dist/bbbs
get *
quit
```

1.43 BBBS BHATCH - Hatch the specified file into the specified file-echo

```
Usage: bbbs.exe bhatch {-replace_file} [filename] [echoname] { ↩
   description}
```

BHATCH is used to hatch a new file into a TICK file-echo . The filename-parameter specifies the file to hatch. Echoname is the name of the file-echo the file should be hatched into, as entered in external.bbb . If you do not specify a description, then BHATCH will try to read it from your file area.

The replace_file-parameter is also optional. It specifies the file that should be replaced by this file. You cannot use wildcards, for obvious reasons.

1.44 BBBS BKILL - Kill users with criterias

```
Usage: bbbs.exe bkill {Tyyyymmdd} {Cx} {Lx} {Rx} {Ux} {Ix} {*}
```

BKILL can be used to kill users with certain criterias. You can specify one or more of the criterias. To be killed or listed, the user must match all the criterias.

Tyyyymmdd	All users who haven't called since yyyymmdd. Defaults into one year from the current date.
Cx	All users who have called less than x times.
Lx	All users who have written less than x messages.
Rx	All users who have read less than x messages.
Ux	All users who have uploaded less than x bytes.
Ix	All users who have limit x (default=don't care (256)).

By default BKILL only lists the users who match the specified criterias. If you are absolutely hundred percent positive that you want to kill the users matching the criterias, you need to add the *-parameter to the end of the command line.

1.45 BBBS BLINKFIX - Update the datestamps of file links

```
Usage: bbbs.exe blinkfix
```

BLINKFIX updates the dates of the file links to match the dates of the real files they are links to. This command should be run every day if you are using file links.

1.46 BBBS BLIST - Make an ASCII format file list

```
Usage: bbbs.exe blist [filename] {groups {starting_dir}}
```

BLIST can be used to build an ASCII-format file of all the files in your BBS. The list will be in the same format as the output of the "f s -r *"-command. To create a list of only new files, you can use the command BNEWF.

The only required parameter is the name of the file the resulting list should be written to. The group-parameter can be used to generate lists of files only for different groups. When used, only files for that group has read-access to are included into the list. The starting_dir-parameter simply defines the directory the list generation should be started from. By default, generation will start from the first directory defined in filedirg.000.

The starting_dir-parameter can only be used if the groups-parameter is also defined.

1.47 BBBS BMKILL - Kill messages with criterias

```
Usage: bbbs.exe bmkill [area] [from] [to] [subject] [after] [ ↩
   before] {*}
```

BMKILL can be used to kill messages according to certain criterias. The area, from, to and subject parameters are regular expressions that are matched against their relative fields. after and before parameters can be standard days; either in yymmdd-format or, for example "-10", for "ten days before the current date". For a message to be taken into account, it must match all criterias.

By default, BMKILL only lists the messages that match the specified criterias. If you are absolutely one hundred percent sure that you want to kill the messages, an additional * parameter has to be added.

Note that the BBBS BPC command has to be used to actually remove the killed messages from the message base.

1.48 BBBS BMSG - Import/export FidoNet mail

```
Usage: bbbs.exe bmsg [R|W|F]
```

The BMSG command can be used to interface BBBS with an external mail tosser. BMSG converts the messages in BBBS messagebase into FTS-0001 format and vice-versa.

The R-parameter (BBBS BMSG R) is used to collect all the new messages from BBBS and convert them into FTS-0001 format files and write them into directories specified in BCFG4 . After this command, a mail processor of some sort should be run to create outbound mail packets.

The W-parameter (BBBS BMSG W) is used to write all the *.MSG format messages to BBBS. Again, they are read from the directory specified in BCFG4.

The final parameter (BBBS BMSG F) requires an additional parameter; the file name of an areatoss file. This file should contain the names or numbers of the areas that should be scanned for new messages. BBBS automatically generates a file called areatoss.nnn into the temporary directory , where nnn is the node number.

It is highly recommended that you use the BBBS's internal BOGUS mail tosser instead of an external one.

1.49 BBBS BMT - A FidoNet message tracker

Usage: bbbs.exe bmt

BMT will scan your NetMail directory for messages with unlisted destination address. It also logs the sender, receiver and subject of the NetMail messages. This command can be run after NetMail is written into the message base.

1.50 BBBS BNC - Compile the nodelist(s)

Usage: bbbs.exe bnc {x}

BNC compiles the nodelist (s) defined in the external.bbb-file to be used with BBBS. This command should be run every time there is changes to a nodelist. If you give an additional parameter (can be anything), then the nodelist is complied even if it is up to date.

1.51 BBBS BNDIFF - Update a nodelist from a NODEDIFF

```
Usage: bbbs.exe bndiff
```

You should run BNDIFF every time you receive a nodediff. It will automagically update the nodelist to be consistent with the nodediff. The nodediff names are read from the nodelist-section of the external.bbb-file.

1.52 BBBS BNEWF - Make an ASCII format list of new files

```
Usage: bbbs.exe bnewf [filename] {groups} {starting_dir} {date}
```

BNEWF can be used to build an ASCII-format file of the new files in your BBS. The list will be in the same format as the output of the F S *-command. To create a list of all files in your system, you can use the command BLIST.

The only required parameter is the name of the file the resulting list should be written to. The group-parameter can be used to generate lists of files only for different groups. When used, only files for that group has read-access to are included into the list. The starting_dir-parameter simply defines the directory the list generation should be started from. By default, generation will start from the first directory defined in filedirg.000.

The date-parameter can be used to control the date after which files are considered to be "new". It can be any standard date, like 961125 (for the 25th of November, 1996) or, for example, -10, meaning the date ten days before the current day.

For example, to include the new files since the last ten days to the list in the file newfiles-10, issue the command:

```
bbbs.exe bnewf newfiles-10 all / -10
```

1.53 BBBS BNMSG - Send a node message.

```
Usage: bbbs.exe bnmsg [fromnode] {-fromnick} [tonode] [message]
```

BNMSG sends a node message to the BBBS. fromnode is the node the message should be coming from. The optional parameter fromnick defines the nick from which the message will seem to come from. tonode is the node that the message should be sent to. The message itself can be up to 250 characters long.

1.54 BBBS BNNTP - Exchange netnews with NNTP servers

```
Usage: bbbs.exe bnntp w [host0] {host1} {host2} {...}
       or bbbs.exe bnntp r [host0] {host1} {host2} {...}
       or bbbs.exe bnntp z [host0] {host1} {host2} {...}
       or bbbs.exe bnntp s [host] [localname] [n]
       or bbbs.exe bnntp c [host] [localname] [n]
       or bbbs.exe bnntp g [host] [filename]
```

BNNTP is used to exchange netnews with NNTP servers. The action-parameter defines whether netnews should be sent or received from the host(s) specified. The W action is used to receive, the R action to send news. The host list must be either names or IP-addresses of valid NNTP servers.

The Z action can be used to reset the want pointers to the last message number in the servers specified.

The actions S and C and be used to set the want pointers so that at maximum n messages are returned the next time news are received from the host specified, with the exception that with the C action, no duplicate articles (such articles that you have already received) will be transferred.

The G action can be used to get the list of available areas in host to the file filename.

1.55 BBBS BOGUS - The BBBS internal FidoNet mail processor

```
                Usage: bbbs.exe bogus w
       or bbbs.exe bogus b
       or bbbs.exe bogus r
       or bbbs.exe bogus f [filename]
       or bbbs.exe bogus a
       or bbbs.exe bogus n {nodes}
```

BOGUS is BBBS’s internal FidoNet mail processor. It can be used to import and export FidoNet mail. Instead of BOGUS, you can also use other, external mail processors with BBBS
BMSG
, but this is not
recommended, as BOGUS can directly manipulate the BBBS messagebase and is therefore a lot faster.

The behavior of BOGUS is controlled via the
FidoNet-configuration
in
the
external.bbb-file

.

There are six different actions that can be used. The W-action (bbbs bogus w) is used for toss/pack; BOGUS tosses all incoming mail packets in the FidoNet inbound directory, forwards mail to the downlinks of your node and imports the mail into the BBBS message base.

The B-action (bbbs bogus b) is almost the same as the W-action, with the exception that only the badecho-directory is checked for messages.

The R-action (bbbs bogus r) is used for scan/pack; BOGUS scans the BBBS message base for mail that is to be exported and packs it.

The F-action (bbbs bogus f) requires an additional parameter: a filename for an areatoss file that should list the area names or numbers that should be scanned for exportable mail. An areatoss-file is automatically generated by BBBS into the temporary directory , named as areatoss.nnn, where nnn is the nodenumber of the node that the areatoss-file is for. Otherwise, the F-action works like the R-action.

The A-action (bbbs bogus a) is like the R-action, with the exception that it uses internal quick scan information to check which areas contain exportable mail. When you use the A-action, the first run can be very slow, but subsequent runs are a lot faster than scans done with just the R-action.

The N-action also requires additional parameters. A list of downlinks or nodes should be specified. An area connection notify is then sent to the nodes/downlinks specified. For nodes that have a ! character in their st field in the nodes-section of external.bbb , the notify is sent only if the node has also the apass field defined.

The R-action should be used daily, even if F or A-actions are used on user logoff to ensure that all mail is actually exported.

1.56 BBBS BOK - Make an OKFILE.TXT file for use with FrontDoor mailer

Usage: bbbs.exe bok [path]

BOK is used to generate a list of files for the FrontDoor mailer. The path-parameter should specify the directory to which your FrontDoor is installed. FrontDoor can then receive file requests and answer to them. BOK will create two files: reqlist.txt, containing the directories from which files can be requested and okfile.txt, containing links to the files. You should specify these files also in your FrontDoor configuration.

It is highly recommended that instead of FrontDoor you use BBBS's internal BackDoor mailer.

1.57 BBBS BOM - Command line interface to the BBBS outbound manager

```
                Usage: bbbs.exe bom [command]

BOM can be used to interface with the BBBS’s outbound manager. The

With outbound manager you can manage your outbound mail. When you enter
into your outbound manager window, you will get list of our outbound mail.
List format is following:

Nodenumber         Age Flags     Bad Busy ArcMail Files
------------------ --- --------- --- ---- ------- -------
2:222/222.0          0 IC RFDE     0    0    45kB  1994kB
2:220/666.0          0   H FDE     0   12     3kB    10kB

Age tells you how old the oldest outgoing NetMail is, i.e. how long time
ago that node polled you.

Flags are:

Immidiate
Crash
Hold
File Request
File attach
Direct
Erase file when sent

Bad tells you number of many unsuccessfull mailsessions there is.
You can limit number of these from
                BCFG4
                .

Busy tells you number of busy calls. Busycalls area cleared every
night.

ArcMail tells you amount of outbound ArcMail.

Files tells you amount of outbound Files.

command-parameter can be any valid outbound manager command:

List {node#}                    List messages (for node)
CHange [message#|node#] {modif} Change message number/all messages for node
Create [node#] {modifiers}      Create message for node
Delete [message#]               Delete message
BAd [node#] [number]            Change number of bad polls for node
Busy [node#] [number]           Change number of busy polls for node
CLear                           Clear tickdir and attach messages
Quit                            Quit back to main menu

Modifiers are:

{New_to_Node#}
{+|-}Immediate  {+|-}Crash  {+|-}Hold  {+|-}Request  {+|-}File  {+|-}Direct
{+|-}Erase      {+|-}Trunc  {+|-}Lock
```

{New_Subject}

1.58 BBBS BPC - Pack a conference with criterias

```
                Usage: bbbs.exe bpc a [areaname] [min] [max]
       or bbbs.exe bpc b
       or bbbs.exe bpc s [areaname] [num]
       or bbbs.exe bpc r [areaname]
       or bbbs.exe bpc t
```

BPC is used to renumber the messages in the conferences. DO NOT RUN ANY BPC COMMANDS WHILE THERE ARE USERS LOGGED IN! This will cause very nasty effects you don't want to experience. Don't say we didn't warn you.

BPC A will delete messages from the conference areaname, so that at least min messages are left active, but no more than max.

BPC B works like BPC A, but it processes all the conferences, reading the min and max values from the
conference configuration
in
BCFG4
.

BPC S deletes all messages from the conference areaname that have a number smaller than num.

BPC R will repair the conference areaname, renumbering the messages so that the first active message will be the message number one.

BPC T scans for bad areas and fixes them, if any are found. It will also renumber the conference like BPC R, if there are more than 62000 active messages in the conference.

1.59 BBBS BPOP - Receive mail from POP3 server

```
                Usage: bbbs.exe bpop [host] [userid] [password]
```

BPOP receives and deletes all mail from userid's POP3 mailbox in host host, using the password password. BPOP first tries to make a secure APOP login and then falls back to normal login if that is unsuccessful. If you want to send your email out, you must of course use the
BBBS BSMTP
command.

POP3 cannot be reliably used with BBS's as the real receiver's name cannot be determined. Still, BBBS will try to figure it out from the "To:"-line. It is very recommended to use bbbs.smtpd.exe instead.

1.60 BBBS BPUS - Pack the user database

```
Usage: bbbs.exe bpus

BPUS will pack the user database by removing all killed users. DO NOT
USE THE BPUS COMMAND WHILE THERE ARE USERS LOGGED IN! This will cause
nasty effects you will not want to experience. Don’t say we didn’t
warn you.
```

1.61 BBBS BRSA - Generate new RSA keys

```
Usage: bbbs.exe brsa

BRSA generates new RSA keys for BZLink-Lite connections by making a new
bzll.dat-file to the main-directory. Please be patient, it might take a
long while to run.
```

1.62 BBBS BSETPACK - Pack the environment variable settings file

```
Usage: bbbs.exe bsetpack

BSETPACK packs the user settings file. This command should be run
every day. DO NOT RUN BSETPACK WHILE THERE ARE USERS LOGGED IN! This
will cause nasty effects you will not want to experience. Don’t say we
didn’t warn you.
```

1.63 BBBS BSMTP - Exchange email with SMTP servers

```
Usage: bbbs.exe bsmtp w [hostname]
       or bbbs.exe bsmtp r [hostname]

BSMTP is used to exchange email with SMTP servers. BSMTP W sends and
receives email to/from the specified host. Receiving email requires a
SMTP server that supports the ’turn’ command. This is unusual, as
the ’turn’ command in SMTP sessions creates a well-known security
hole. To receive email, you can use BBBS’s internal SMTP daemon,
bbbs.smtp.exe.

BSMTP R simply sends all outgoing email to the specified host.
```

1.64 BBBS BSTAT - Show a statistics list

```
Usage: bbbs.exe bstat

BSTAT displays some BBBS usage statistics for the last seven days. It
displays the total amount of online minutes, calls and messages left
to the system.
```

1.65 BBBS BTICK - Process he incoming TICK files from file-echoes

```
                Usage: bbbs.exe btick

BTICK is used to process the incoming TICK files from the

                file-echoes
                 and to place the files to correct file areas in your
BBBS. You should run this command every time you receive TICK files.
```

1.66 BBBS BTIME - Adjust the time with a timeserver

```
Usage: bbbs.exe btime [hostname] [socket_number] [max_diff] [add_secs]

BTIME can be used to adjust the clock of your computer according to the
time on a timeserver residing in the host specified by hostname, at the
socket specified by socket_number. Normally, the socket number is 37,
the standard time socket.

The max_diff parameter should be used to specify the maximum
difference (in seconds) that can exist. If the difference is greater than
max_diff seconds, then no adjusting is done.

add_secs specifies the amount of seconds to add to the time received from
the timeserver. It should be 0, unless the 'ping' time to the timeserver
host is very long.
```

1.67 BBBS BTXT2BBS - Convert a text file to a message

```
Usage: bbbs.exe btxt2bbs [areaname] [file] {/F from} {/T to} {/S subject}

BTXT2BBS is used to import a single text file to the BBBS message
base. areaname is the name of the area that the message should be
written to. file is the name of the file to be written. The /F, /T and
/S switches can be used to change the sender, receiver and subject of
the message, but they are optional. They default to "SysOp", "all" and
the name of the file being written.
```

1.68 BBBS BWHO - Show who is online at the moment

```
Usage: bbbs.exe bwho

BWHO shows the online status of the system at the moment, in the same
format as BBBS's internal who-command.
```

1.69 BBBS in TCP/IP networks

All versions of BBBS (except the PC-DOS version, BBBS/D) can be very easily interfaced with TCP/IP networks, such as the Internet. BBBS comes equipped with all necessary tools for TCP/IP network connections.

Access configuration for all Internet features is done with the

inet.bbb-file . If you want your users to be able to use the network features, be sure to check that this file is set up correctly.

The special executables (bbbs.telnetd.exe etc) are present only in BBBS/2 and BBBS/NT versions, as the UNIX versions communicate through stdin/stdout and are therefore simple to configure into using a TCP/IP network.

Outgoing telnet connections

If you want BBBS to be able to "dial" out with the standard telnet protocol, for example to do mail polls via the Internet, you must start BBBS with the TCPIP device. When doing so the com parameter is ignored.

Note that nodes that are started for TCP/IP dialling should have

slow protocols enabled in BCFG4 . Also, the "dial string" should be just ATD, not "ATDT". With outgoing telnet connections, the ATZ command is used to control the connection type: ATZ7 defaults to 7-bit connection, ATZ8 to an 8-bit one.

For a telnet poller, following nodelist format should be used:

```
;A With just an IP-address, using nonstandard port 666:
;A
,1000,A_Telnet_Poller,Cyberspace,Joe_Hacker,192.210.9.14:666,300,IP
;A
;A With a nameserver address, using default port 23:
;A
,1000,A_Telnet_Poller,Cyberspace,Joe_Hacker,bbs.telnetpoll.org,300,CM,IP
```

Incoming telnet connections

For incoming telnet communications, the program bbbs.telnetd.exe is provided. It will act as a telnet daemon and a gateway between the telnet connection and the BBBS. With bbbs.telnetd.exe, both user logins and mail polls are possible via telnet. Of course, for polling to be possible BackDoor must be enabled in BCFG4.

The calling convention for bbbs.telnetd.exe is:

```
bbbs.telnetd.exe <min_node> <max_node> <socket> [priority] ["identd"]
```

min_node is the first node to use for telnet connections, max_node is the last one. socket is the number of the socket bbbs.telnetd.exe should watch for incoming connections. 23 is the default telnet port, but you can specify any port you like. Priority defines the OS priority bbbs.telnetd.exe should run in.

The final parameter identd can be specified if you want to use identd with the telnet connections. When identd is used, the user id of the telnetter is reported, along with the IP address the user is telnetting from. Using identd is recommended only if you really need it.

When bbbs.telnetd.exe detects an incoming telnet connection for the port it was started into, it will spawn dotbbbs.cmd (dotbbbs.bat in BBBS/NT) in the current directory. This batch file should contain the necessary commands to start BBBS with the parameters given to the batch file.

Nodes that are configured for telnet connections should have the string "telnet" in their FD's DOBBS.BAT-field in BCFG4.

Incoming SMTP connections

BBBS can deal with incoming SMTP connections with the bbbs.smtpd.exe mail daemon program. It will keep an eye on a specific port and deal with incoming SMTP connections accordingly. The calling convention of bbbs.smtpd.exe is:

```
bbbs.smtpd.exe [socket_number] {priority}
```

socket_number is the number of the socket the mail daemon should be started to. The standard SMTP socket is 25, but you can start the daemon also to nonstandard ports. Priority-parameter defines the OS priority the daemon should run in.

You can send out email with the SMTP protocol with the BBBS command line command

BSMTP

.

BBBS also supports POP3 connections with for receiving email with

BBBS BPOP

. This is not recommended on multi-user systems, though.

Incoming finger connections

For incoming finger connections, the BBBS finger daemon bbbs.fingerd.exe is provided. It will keep an eye on a specific port and deal with incoming finger connections accordingly. The calling convention of bbbs.fingerd.exe is:

```
bbbs.fingerd.exe [socket_number] {priority}
```

socket_number is the number of the socket the finger daemon should be started to. The standard finger socket is 79, but you can start the daemon also to nonstandard ports. Priority-parameter defines the

OS priority

the

daemon should run in.

Remember to check you

inet.bbb

file for incoming finger configuration!

Example:

A sample dotbbbs.cmd file:

```
c:
cd \bbs
bbbs.exe %1 %2 %3 %4 %5 %6 %7
```

1.70 Waiting for a caller screen

When BBBS is waiting for a caller, it displays some statistics, like the name of the previous caller and the amount of messages entered. In this screen, there are some keys that you can use to invoke different options. They are listed below.

You can also select all these options from the pull-down menus, activated with the F1 key. Some of the options are available only if you have enabled Backdoor from

BCFG4

. They are marked with an asterisk ('*').

```
alt-A      Set SysOp available for chat requests.
alt-B      Set modem busy and exit BBBS.
alt-E      Exit with an errorlevel.
alt-F      Force modem to answer an incoming call now.
alt-G      Request a file from a remote system. *
alt-J      Start the
                BTERM Terminal Emulator
                .
alt-L      Make a local login to this node.
alt-N      Set SysOp not available for chat requests.
alt-O      Start the outbound manager. *
alt-P      Poll a remote system. *
alt-Q      Exit BBBS with errorlevel 0.
alt-S      Send a file to a remote system. *
alt-X      Reset modem.
alt-Z      Jump to OS shell.
space      Rescan outbound messages. *
```

1.71 Local SysOp keys

Listed below are the local sysop keys you can use when an user is logged into your BBBS system, assuming you have enabled them in

BCFG4
.

```
F10        Show information about the user that is online.
F6         Start full screen SysOp-to-user chat. Press ESC to end it.
F7         Start line-by-line SysOp-to-user chat. Press F7 again to
           end chat.
shift-F1   Reduce the online users' time by 5 minutes.
shift-F2   Reduce the online users' time by 1 minute.
shift-F3   Increase the online users' time by 1 minute.
shift-F4   Increase the online users' time by 5 minutes.
shift-F5   Reduce the online users' time limit by 5 minutes, permanently.
shift-F6   Reduce the online users' time limit by 1 minute, permanently.
shift-F7   Increase the online users' time limit by 5 minutes, permanently.
shift-F8   Increase the online users' time limit by 5 minutes, permanently.
alt-A      Set SysOp available for chat requests.
alt-B      Enters backscroll mode. Cursor keys up and down scroll, ESC
           key ends backscroll. Enter and space keys paste the middle
           line to the user screen . The backscroll is completely transparent;
           the remote user will not notice that you are in backscroll.
           Backscroll cannot be used while the user is downloading. Backscroll
           is available only in operating systems that have threads.
alt-C      Start full screen SysOp-to-User chat. Same as F6.
alt-E      Toggle whether the stuff sent to remote users' screen
           should be displayed in local screen.
alt-F      Grant file access to the user online. This is a permanent
           access change.
alt-K      Hangup this node now.
alt-L      Lock the remote keyboard. With this you can make sure that
           the remote user does not interfere with what you're doing on the
           node. Very useful when you grant temporarly SysOp access.
```

```
alt-N      Set SysOp not available for chat requests.
alt-Q      Take this node down after the current user logs out.
alt-S      Toggle temporary SysOp access #255 for the user
           online.
alt-U      Adds a file to the outgoing batch in HYDRA sessions. You
           should write the name of the file to be added to the chat
           window.
alt-Z      Drop to OS shell. The user online will get a message
           informing him that the SysOp has dropped into OS and that
           he should wait for the SysOp to return. Nasty.
alt-0      Run
                keyboard macro
                 0.
alt-1      Run keyboard macro 1.
alt-2      Run keyboard macro 2.
alt-3      Run keyboard macro 3.
alt-4      Run keyboard macro 4.
alt-5      Run keyboard macro 5.
alt-6      Run keyboard macro 6.
alt-7      Run keyboard macro 7.
alt-8      Run keyboard macro 8.
alt-9      Run keyboard macro 9.
```

1.72 BTERM, the small VT320 terminal emulator

BTERM, accessible from either the waiting for a caller-screen or from the command line by specifying the device BTERM (or by starting the BBBS executable named to bterm.exe), is a small VT320 terminal emulator. It allows you to call to other systems quickly, without having to start another program.

When BTERM is started from the command line, any additional parameters after the device will be interpreted as script names to be executed.

If BTERM is started with the ISDN device (OS/2 version only), then BTERM can interface the ISDN CAPI via a simple Hayes command emulation:

Command	Action
ATA	Answers incoming ISDN call.
ATDxxxx	Dials number xxxx.
ATI	Gives some information.
ATZ	Initializes the ISDN CAPI.
ATZx	Specifies the EAZ level x.

If BTERM is started with the TCPIP device, then BTERM can "dial" out using the standard TCP/IP telnet protocol. The ATD command can be used to achieve this. After the ATD command, simply insert the hostname or an absolute IP address. For example, to telnet to fix.no, just type "ATDfix.no". If you want to specify a different port than the standard telnet port (23), then you should separate it from the hostname or

```
IP address with the colon character (":"). For example, to telnet to
foobar.org, port 3000, type "ATDfoobar.org:3000". Also, the ATZ command can
be used to control the telnet connection type: ATZ7 defaults to 7-bit
connections, ATZ8 to 8-bit ones.

The following keys can be used in the terminal screen:

F1    Show this help.
F5    Toggle
                BZLink-Lite
                 on/off. When you are connected with
      BZLink-Lite, you cannot disable it.
F6    Set BZLink-Lite file transfer priority. Priority 1 is the
      slowest.
F7    Show some BZLink-Lite statistics: status of current
      download/upload, the amount of sent packets and the amount of
      resends for this session.
Alt-A Toggle audio bell on/off.
Alt-B Enter backscroll mode. Up/down arrow keys scroll, space and
      enter keys paste the middle line to the terminal.
Alt-C Select the
                character set
                 to use. By default, BTERM uses the
      ISO character set.
Alt-E Execute a
                BZ-script
                 in BTERM mode.
Alt-F Send a FAX.
Alt-G
                Get (download) a file.
                Alt-J Clear screen.
Alt-K Hang up now, without asking.
Alt-L Toggle the log file (bterm.log) on/off.
Alt-M Toggle newline/linefeed mode.
Alt-O Record a voice file.
Alt-P Enter the
                phonebook
                .
Alt-Q Quit BBBS, exit straight into the OS.
Alt-R Send a break signal.
Alt-S
                Send (upload) a file
                .
Alt-T Display the elapsed online time.
Alt-V Playback a voice file.
Alt-X Quit BTERM; exit into the caller waiting screen or into
      OS, depending on where BTERM was started from.
Alt-Z Shell into the OS.
Alt-0 Run
                keyboard macro
                 0.
Alt-1 Run keyboard macro 1.
Alt-2 Run keyboard macro 2.
Alt-3 Run keyboard macro 3.
Alt-4 Run keyboard macro 4.
Alt-5 Run keyboard macro 5.
Alt-6 Run keyboard macro 6.
```

```
Alt-7 Run keyboard macro 7.
Alt-8 Run keyboard macro 8.
Alt-9 Run keyboard macro 9.
```

1.73 The BTERM Phonebook

Pressing ALT-P in the BTERM terminal screen opens the phonebook.

The following keys can be used:

```
Up/Down  Move the selector.
Space    Mark the system under the selector.
Enter    Start dialing. If no systems are marked, the system under the
         selector is dialled. Otherwise, the systems are dialed in the
         order they were marked.
E        Edit the entry under the selector.
Del
                Delete
                 the entry under the selector.
Ins      Insert
                a new entry
                 to the selector position.
ESC      Exit phonebook to terminal screen.
```

BBBS uses the file bterm.pho to store the phonebook.

1.74 Really delete phonebook entry -dialog

When you delete entries, BTERM wants you to confirm that really want to delete the phonebook entry.

If you do, press Y. If you do not, press N.

1.75 BBS name -dialog

When inserting entries, you will be presented with two dialogboxes. To the BBS name-dialog you should enter the name of the system in question. After this, you will get the
BBS phonenumber -dialog

1.76 BBS phonenumber -dialog

Enter here the phonenumber(s) of the system. If the system has many nodes, just enter them all. When you have entered enough phonenumbers, press enter on an empty dialog. When dialling to a system, BTERM will call the numbers you entered sequentially.

Note that BTERM will do phone number conversion to these entries, too!

1.77 Download a file -dialog

Alt-G starts file downloading (gets a file from remote).

BTERM supports following file transfer protocols for downloading files:

HYDRA	bidirectional with chat
Slow-HYDRA	HYDRA for 7bit links
Zmodem	unidirectional full stream
Ymodem	unidirectional with block acking
ZedZap	Zmodem with 8kB blocks
Kermit	unidirectional, bit slow, but very universal

If you use HYDRA you can also upload files at the same time. To add files after you have started a download session, press Alt-C to enter the HYDRA chat window . Then type the filename with full pathname and press Alt-U to add it to the outgoing batch.

1.78 Filename to upload -dialog

When you start uploading, you will be presented with a dialog to which you should enter the file names that should be uploaded (wildcards are ok). Press Enter on an empty dialog to enter the file selector.

After this dialog, you will be presented with the upload protocol-dialog . In the selector you can move with the cursor keys, delete a file with the del-key and send the file under the selector with the space bar.

1.79 Upload protocol -dialog

BTERM supports following file transfer protocols for uploading files:

HYDRA	bidirectional with chat
Slow-HYDRA	HYDRA for 7bit links
Zmodem	unidirectional full stream
Ymodem	unidirectional with block acking
ZedZap	Zmodem with 8kB blocks
Xmodem	slow, but universal
Kermit	unidirectional, bit slow, but very universal
ASCII	just dump text file to remote

If you use HYDRA you can also download files at the same time. To add files after you have started a upload session, press Alt-C to enter the HYDRA chat window. Then type the filename with full pathname and press Alt-U to add it to the outgoing batch.

1.80 Really delete -dialog

When you press del in the file selection dialog, BTERM wants you to confirm that really want to delete this file.

If you do, press Y. If you do not, press N.

1.81 Select charset -dialog

Alt-C gives you a list of character sets that you can choose.

Possible choices are:

ISO Latin-1	8bit, used in most InterNet systems (default)
IBM PC	8bit, used in most BBS systems
Finnish	7bit, used in some BBS in Finland
Norwegian	7bit, used in some BBS in Norway
UK	7bit, no national charset conversion
MAC	8bit, used by MAC computers

1.82 The BZLink-Lite protocol

One of the most powerful features of BTERM and BBBS is the support of the BZLink-Lite protocol. There are many advantages from using BZLink-Lite. One of these features is the secure login service. Systems that can act as BZLink-Lite servers (like BBBS) autodetect BZLink-Lite clients (like BTERM) at login and make the login secure, so that no passwords are transferred between systems, only encrypted strings.

Another great feature of BZLink-Lite is that file transfers are completely transparent: you can start receiving a file and continue your business in the BBS in a normal way, with a negligible loss of response time or file transfer speed.

BTERM's BZLink-Lite client mode is activated with the F5 key in the terminal mode. Next login to a system with a BZLink-Lite server will then be a BZLink-Lite one.

The priority of BZLink-Lite file transfers can be used to control how much bandwidth is dedicated to the file transfer and how much to normal BBS usage. With priority 1, most of the time is dedicated to normal usage. Higher priority means faster file transfer, but slower response time for other activities. The (approximate) file transfer speed is priority*512 characters per second. Within BTERM, the priority can be changed with the F6 key.

1.83 BBBS Scripts: The BZ Programming Language

For heavy customization, BBBS has the possibility to run BZ49 processor binaries. All scripts that are executed by BBBS are BZ49 binaries. The bzc.exe compiler program that came with BBBS can be used to generate BZ49 binaries with the BZ programming language. The BZ language (hereafter referred to only as BZ) is a very powerful programming language, with features from C, Pascal and perl.

BZ source code files have the extension .BZ. BZ language header files have the extension .BZH. BZ49 binaries have the extension .BZB. BZ49 assembly source code files should have the extension .BZS.

BBBS will run the following scripts, residing in the script directory specified in BCFG4 :

Script	When it is run?
ascript	Always when a user logs in.
auscript	After a user has entered a description for a uploaded file.
brobo	When a user sends an unknown command to BRoboCop.
dscript	After a user has downloaded a file.
editor	When a user starts message editing and his editor is "script".
error	When a user enters an unknown command.
gascript	When a user logs out with the g a-command.
gscript	When a user logs out normally.
gotfax	When a FAX is received.
gotmail	When new FidoNet mail is received.
gotvoice	When a voice call is received.
hotlog	After a special hotlogin string from the user.
lscript	When the user logs out, after GSCRIPT and GASCRIPT.
msgfilt	After a user has saved a message in a message editor.
msgpfilt	Before the message editor is invoked.
newupass	Before asking the closed system password .
nodemsgf	When a user receives a node message (to show it).
notavail	After a chat request, if sysop is not available.
opendoor	When a user executes the main menu OPen command.
postchat	After sysop-to-user chat has ended.
rscript	After a new user has registered to the system.
script	When a user executes the main menu Answer-command.
snooze	When a user is about to be sleep disconnected.
uscript	After a user has uploaded a file.

Also, every time BBBS tries to print the string \2foo\2, where \2 is the ASCII character 2, the script foo is run. This can be used almost everywhere BBBS displays something; the bbbstxt-files, menus , bulletins...

Some BZ-scripts have parameters passed to their main()-function. See the examples that came with the distribution package to see which ones do, and what.

This manual documents BZ, but is not a guide to learn programming with. We assume that you have done programming with a structured programming language like C or Pascal before. If you haven't, then there are a lot of good books and tutorials out there that teach the concepts of structured programming. Reading a C tutorial/book will help tremendously as many BZ functions resemble their C counterparts or are exactly the same.

As the source code of BZ is pure ASCII, you can use any text editor to create your source. Remember that CASE IS IMPORTANT, like in all major languages. Whitespace (such as spaces, tabulators, carriage returns or line feeds) is not important, but using them makes the source look cleaner and it is easier for you and other people looking at your source to understand what you mean. The compiler, however, does not care how you arrange the source, so you can write it as you see fit.

In the distribution package, several examples of BZ scripts have been supplied. The best way to learn BZ is to study these files carefully and modify them to see what happens. Try it out! You really can't break anything!

Note that BBBS also supports the use of some other script languages .

1.84 BZ Language: Expressions

Every BZ command is an expression. A simple function call foobar() is an expression the value of which is the return value of the function foobar. The name of a variable is an expression with the value of the variable. Expressions can be grouped with operators to form more complex expressions. The evaluating of an expression means that the value of it is determined. Integer constants evaluate into the value the constant represents. String constants evaluate into 0 when they are used in an arithmetic context. Otherwise, they evaluate into the sequence of characters the string represents.

Every expression also evaluates into a boolean value; true or false. Expression is false if the value of it is 0, any other value means that the expression is true.

An assignment is an expression in which the value of an expression is placed into a variable. The value of an assignment expression is the value of the variable that is assigned to the other variable. Thus, the value of an expression like a=5 is 5. The value of the expression foo=2*500; is 1000, the value of the expression 2*500.

There is also the concept of a so-called conditional expression: the syntax of it is: expression?true:false. First, the expression is evaluated. If it evaluates to true, then the true-part of the conditional expression is evaluated next. Otherwise, the false-part is evaluated.

1.85 BZ Language: Functions and Variables

The structure of a BZ program is:

```
function {
   expression
   expression
   ...
}
```

Therefore, BZ is based on functions. The execution of a program starts from the function main and returning from it ends the program. "Execution" of a program means the evaluation of sequential
expressions
.

A function is defined with the keywords int or char. For example:

```
// An example function
/*
   Note how BZ uses C++-style comments. Everything between /* and */ are
   considered as comments. When there is a // in a line, everything after
   it is considered to be a comment.
 */
int example(int foo) {
  // the code goes here
}
```

A function can also have parameters passed to it. The parameters are listed in the parenthesis, separated with commas. This example function has one integer type of parameter called foo. Note how the block (the function code, in this case) is begun C-style. This is universal in BZ: blocks of code always start with curly brackets. There are a number of functions and constructs already defined in the
BZ Runtime Library
. They are the functions
you will mostly use.

A function is called by writing its name and the parameters that should be passed to it in parenthesis. For example, the function above could be called simply as example(). BZ functions always return something, normally 0. The special function return() can be used to return from a function with a value. Example:

```
// A function that always returns 42 (not very useful).
int return_42() {
  return(42);
}
```

The parameter of return can be any

expression

.

The int and char keywords are also used to declare variables in BZ. After int or char the variable name should be written. Variable name can consist of all letters and numbers and the underscore character '_'. A variable name cannot begin with a number. Multiple variables can be defined with a single keyword, they just need to be separated with commas. A variable is only effective in the block that it is defined in. To declare a global variable, declare it outside a function. In all cases, the variable can be used only after the point it has been defined.

The keyword int declares integer variables and char declares string variables. BZ does not have typing, so you can assign integer values to string variables and vice-versa. However, this is VERY bad style and might retailiate in the future, when stricter typing is added to BZ someday.

```
// An example of a variable declaration:
int example2() {
   int  number,second_number;
   char string,second_string;
}
```

When declaring variables, they can be assigned an initial value when they are defined:

```
// An example of variable declaration with initial value:
int example3() {
   int  bz=42;
   int  yebo=bz*49;
   char b="Kim Heino",z="Tapani Salmi",pjh="Durak";
}
```

Naturally, variables can be assigned values. This is done with the =

operator

.

```
// An example of an assignment
int example4() {
   int foo,bar,buz;

   foo=2;
   bar=5;
   buz=foo+bar;
}
```

Every variable in BZ is an index variable. An index is referred to as variable[index]. The name of the variable is actually just index 0 of it. Multiple indeces can also be given an initial value:

```
// An example of variable declaration with initial value on multiple indeces:
int example5() {
   char digits={
     "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"
   };
   digits[1];  // this contains the string constant "1".

}
```

1.86 bzc.exe: the BZ Language compiler

bzc.exe is the program that is used to compile BZ source codes into BZ49 binaries. The calling convention of BZC is very simple:

```
bzc.exe file {*}
```

The parameter file is the file name to be compiled. It is not necessary to write the extension .BZ. If you give the additional parameter, the asterisk character, then bzc.exe is run in debug mode: it displays the compiled BZ49 assembly source file after compilation. It is only of use for die-hard BZ49 gurus, so you should not bother.

bzc.exe also features a built-in preprocessor. The preprocessor first scans through the source file, before any compiling is done. It understands two compiler directives: #include and #define.

The #include compiler directive can be used to insert another source file to a certain point in the source. It is simple to use; after the directive should come the name of file to include at that point, enclosed between "less than"- and "greater than"-characters or quotes. For style reasons, you should only include files that have the extension .BZH.

Examples:

```
#include <foo.bzh>     // includes foo.bzh
#include "foo.bzh"     // ...likewise...
```

The #define command can be used to create simple macros. The syntax of #define is:

```
#define OLD new
```

After a #define directive, whenever OLD is encountered in the source, it is replaced with new.

1.87 BZ Language constants

A string constant is a sequence of ASCII characters enclosed in quotes, for example, "Hello", "World", or "Hello World". Some character can, or must be, inserted by quoting it with '\' character.

Quote	Character	ASCII
\"	"	34
\\	\	92
\a	Bell	7

\b	Backspace	8
\e	Escape	27
\f	Form Feed	12
\n	Line Feed	10
\r	Carriage Return	13
\t	Horizontal Tabulator	9
\v	Vertical Tabulator	11

All other characters can be quoted by writing a backslash followed by character’s ASCII number in octal (base-8). For example ESC (\e) can also be written "\033".

An integer constant is a sequence of digits representing an integer value in the range -2147483648 to 2147483647. An integer constant must start with a digit from 1 to 9 or the negative sign ‘-’ followed by a digit. You can also use octal integers (base-8, starting with 0) and hexadecimals (base-16, starting with 0x).

1.88 BZ Language operators

Below are listed all the operators in BZ language:

Operator	(Un|Bin)ary	What it is/does
-	unary	Arithmetic negation
~	unary	Complement
!	unary	Logical NOT
++	unary	Increment by one
--	unary	Decrement by one
*	binary	Multiplication
/	binary	Division
%	binary	Remainder (Modulo)
+	binary	Addition
-	binary	Subtraction
<	binary	Is less than?
>	binary	Is greater than?
<=	binary	Is less than or equal to?
>=	binary	Is greater than or equal to?
==	binary	Is equal to?
!=	binary	Is inequal to?
&	binary	Bitwise AND
|	binary	Bitwise OR
^	binary	Bitwise Exclusive OR
&&	binary	Logical AND
||	binary	Logical OR
<<	binary	Bit shift left
>>	binary	Bit shift right
=	binary	Assignment
+=	binary	Assignment with addition
-=	binary	Assignment with substraction
*=	binary	Assignment with multiplication
/=	binary	Assignment with division
&=	binary	Assignment with bitwise AND
|=	binary	Assignment with bitwise OR
^=	binary	Assignment with bitwise Exclusive OR

<<=	binary	Assignment with bit shift left
>>=	binary	Assignment with bit shift right

1.89 REXX as the script language

With BBBS/2 it is possible to use REXX as the script language. When starting a script, BBBS/2 first checks for the existence of a BZ49 binary. If one is not found, then BBBS/2 will try to execute a CMD file with the same name. If one is found, it is then executed as a REXX script. Note that you must have REXX support installed, as BBBS REXX scripts are passed through the standard REXX interpreter.

BBBS/2 adds some functions to rexx:

Function	Usage
bbbs_say	Works like the normal REXX say function, except that output is done to both local and remote screens.
bbbs_kbhit	Works like the BZ Runtime Library kbhit() command.
bbbs_getch	Works like the BZ Runtime Library getch() command.
bbbs_input	Works like the BZ Runtime Library input() command.
bbbs_getvar	Gets a BZ Runtime Library variable. Requires the variable name and optinally the index (for such variables that are indexed) as parameters.
bbbs_setvar	Sets a BZ Runtime Library variable. Requires the variable name and the value to be set and optionally the index (for such variables that are indexed) as parameters.

Otherwise, there is nothing special about REXX scripts. They are just like any other REXX scripts, except that they are executed by BBBS.

1.90 The BZ Runtime Library by Categories

The BZ runtime library consists of the following predefined functions and constructs. See also the stdlib.bzh header file for constant definitions and some extra functions.

See also the alphabetic list of functions.

```
Constructs:

                asm

                break

                do...while

                for

                if...else

                return

                switch...case

                while
                General Functions:

                beep()

                breceive()

                bsend()

                delay()

                _delay()

                exit()

                help()

                tictactoe()

                yellsysop()
                Low-level Functions and System Calls:

                bbbs()

                bfile4()

                exec()

                getenv()

                getnodestatus()

                make_pcboard12sys()

                make_pcboard14sys()

                sendnode()

                setenv()

                spawn()
```

```
system()

time()
String Manipulation Functions:

copy()

delete()

parsecom()

pos()

regexp()

sprintf()

stlocase()

strcat()

strlen()

stupcase()
File I/O Functions:

closedir()

fclose()

feof()

fflush()

fgetc()

fgets()

fopen()

fprintf()

fputc()

fseek()

ftell()

getnodemessage()

opendir()

readdir()

remove()
```

```
rename()

rewinddir()
Standard I/O Functions:

getch()

_getch()

input()

kbhit()

printf()
User Data Functions:

freeuser()

loaduser()

seekforuser()
```

1.91 Alphabetic listing of the BZ Runtime Library

```
There is also the list of BZ Runtime Library commands
by category
.


  asm
 - directly program the BZ49-processor

  bbbs()
 - execute a BBBS-command

  bfile4()
 - enter the File/4 filesystem

  beep()
 - emit a beep sound

  break
 - end the execution of a block

  breceive()
 - receive file(s) from remote

  bsend()
 - send file(s) to remote

  closedir()
 - close a directory opened with opendir

  copy()
 - copy a part of string
  delay(), _delay()
 - halt program execution for a specified amount of time

  delete()
 - deletes a part of a string

  do..while
 - repeat a command until a condition is met

  exec()
 - execute another script

  exit()
 - quit the current script

  if..else
 - execute a statement based on an expression

  fclose()
 - close a file

  feof()
 - determine if a file pointer is at end of file

  fflush()
 - flush a file into the disk

  fgetc()
 - read a character from a file

  fgets()
 - read a string from a file

  fopen()
 - open a file

  for
 - repeat a loop until a condition is met

  fprintf()
 - write formatted text into a file

  fputc()
 - write a single byte into a file

  freeuser()
 - return normal bu-variable values

  fseek()
 - position the pointer of a file handle

  ftell()
 - return the file pointer position

  getch(), _getch()
 - read a keypress without waiting
```

```
 getenv()
- get the value of an environment variable

 getnodestatus()
- get a part of the status of a specified node

 getnodemessage()
- read a node message from a file

 help()
- execute the AmigaGuide help file reader

 input()
- read user input

 kbhit()
- check to see if a key has been pressed

 loaduser()
- adjust bu-variables to the values of another user

 make_pcboard12sys()
- make a v12 PCBOARD.SYS control file

 make_pcboard14sys()
- make a v14 PCBOARD.SYS control file

 opendir()
- opens a directory for reading

 parsecom()
- parse input for commands

 pos()
- search for a substring in a string

 printf()
- write formatted output into the screen

 readdir()
- read a single directory entry

 regexp()
- check to see if a string matches a regular expression

 remove
- delete a file

 rename()
- rename a file

 return
- return from a function

 rewinddir()
- move the directory pointer to the first entry
```

```
 seekforuser()
- find the user number of an user

 sendnode()
- send a node message

 setenv()
- set an environment variable

 spawn()
- execute another script

 sprintf()
- write formatted text into a string

 stlocase()
- convert a string into lower case

 strcat()
- join two strings

 strlen()
- calculate the length of a string

 stupcase()
- convert a string into upper case.

 switch...case
- create big if...else-loops with same condition

 system()
- execute an operating system command

 time()
- return seconds since 1.1.1970, 00:00 UTC

 tictactoe()
- execute the game of TicTacToe

 yellsysop()
- page the system operator

 while
- repeat until a condition is met
```

1.92 asm

```
CONSTRUCT  asm() - directly program the BZ49-processor

 SYNOPSIS  asm(command, argument);

           command  - a BZ49-processor command
           argument - an integer containing the command argument
 DESCRIPTION  asm() is used to control the BZ49-processor directly. command
              can be any valid BZ49-processor command. argument should be
              the argument of the command. This command should be used by
              die-hard BZ49-gurus only.
```

1.93 break

```
                   CONSTRUCT  break - end the execution of a block.

    SYNOPSIS  break;

 DESCRIPTION  break ends the execution of the current block, as if there was
              a closing bracket there.

     EXAMPLE  for(i=0;;++i) {
                if(i==3) break;
              }

    SEE ALSO
                for
                ,
                do..while
                ,
                switch..case
                ,
                while
```

1.94 do...while

```
                   CONSTRUCT  do...while - repeat a command until a condition is  ↩
                      met

    SYNOPSIS  do command while(condition);

 DESCRIPTION  do...while executes the command once and then repeatedly,
              until the
                expression
                 condition evaluates to false.

     EXAMPLE  do {
                ++i;
                printf("Still under 42...\n");
              } while(i<42);

    SEE ALSO
                while
                ,
                for
                ,
                break
```

1.95 for

CONSTRUCT for - repeat a loop until a condition is met

SYNOPSIS

```
for({expression{,...}}; {condition{,...}}; {expression{,...}})
statement;
```

DESCRIPTION

for is used to repeat the statement until the condition-expression evaluates to false. Before the loop is started, the expressions listed before the first semi-colon are evaluated. Every time the loop has been completed the third expression is evaluated. Normally, the first expression should be used to initialize a counter variable and the third expression to increment it.

EXAMPLE

```
for(t=10; t>0; t--) {
  printf("T minus %d and counting...\n",t);
  delay(10);
}
```

SEE ALSO

while , do...while

1.96 if...else

CONSTRUCT if...else - execute a statement based on an expression

SYNOPSIS

```
if(condition) command1; {else command2;}
```

DESCRIPTION

if is used when a command or group of commands should be executed only if a specified condition is met. command1 is executed only if the expression condition is true. The else-part is optional. If it is included, then command2 is executed if the expression condition is false.

EXAMPLE

```
if(foo != bar) {
  printf("Make the world equal!\n");
  foo=bar;
} else printf("Equality forever!\n");
```

SEE ALSO

switch...case

1.97 return

```
                  CONSTRUCT  return - return from a function

    SYNOPSIS  return({expression});

 DESCRIPTION  return is used to exit a
                BZ function
                 and optionally return a
              value, specified by expression. If expression is not specified,
              the value 0 will be returned.

     EXAMPLE  int addition(int first, int second) {
                int result;

                result=first+second;
                return(result);
              }

              int main() {
                printf("2 + 3 = %d\n",addition(2,3));
                return(0);
              }

    SEE ALSO
                break
                ,
                exit
```

1.98 switch...case

```
                  CONSTRUCT  switch...case - create big if...else-loops with same
              condition.

    SYNOPSIS  switch(condition) {
                case expression2: statement;
                case expression3: statement;
                ...
                default:statement
              }

 DESCRIPTION  switch...case is used to create big if...else loops with the
              same condition. First, the
                expression
                 condition is evaluated.
              Then all case-statements and their relative expressions are
              checked for equality. When an equality is found, all the
              following statements are executed. If none of the case
              expressions match, the optional default-statement is executed.
              Note that to execute only one case-statement, the statement
              has to end in break; call. Otherwise all case-statements below
              it are also executed.

     EXAMPLE  switch (bar) {
                 case 1:
                 case 2:{
```

```
                printf("1 or 2\n");
                break;
                }
                case 3:{
                   printf("3\n");
                   break;
                }
                case 4:printf("4 or ");
                case 5:{
                   printf("5\n");
                break;
                }
                default:{
                   printf("unknown\n");
                   break;
                }
             }

    SEE ALSO
               if...else
```

1.99 while

```
                  CONSTRUCT  while - repeat until a condition is met

    SYNOPSIS  while(condition) command;

 DESCRIPTION  while is used to loop continuously until a condition is met.
              Every time before executing command, the
                expression
                 condition
              is evaluated. If it evaluates into true, then the command is
              executed. Note that if the expression is false from the very
              beginning, the command will never be executed.

     EXAMPLE  while(t>0) {
                printf("T minus %d and counting...\n",t);
                delay(10);
                t--;
              }

    SEE ALSO
                do...while
                ,
                for
```

1.100 bbbs

```
    FUNCTION  bbbs() - execute a BBBS-command

    SYNOPSIS  bbbs(command);
```

```
              command - a string containing the command line to be executed

 DESCRIPTION  bbbs() executes the command line specified in command as if
              it were issued by the user.

     EXAMPLE  bbbs("f cd /pub/pictures get *");
              bbbs("g y");

RETURN VALUE  bbbs() returns always 0.
```

1.101 bfile4

```
                    FUNCTION  bfile4() - enter the File/4 filesystem

    SYNOPSIS  bfile4();

 DESCRIPTION  bfile4() enters the BBBS File/4 filesystem, as if the user
              had issued the BBBS command to enter the file menu. bfile4()
              exits when the user issues either the Quit or Goodbye command.
              When the user is using a file menu invoked with a call to
              bfile4() he/she cannot use any of the global commands or
              change the active menu.

RETURN VALUE  bfile4() returns always 0.

     EXAMPLE  int main() {
                printf("Entering filemenu...\n");
                bfile4();
                printf("Have a nice day!\n");
              }

    SEE ALSO
                bbbs()
```

1.102 beep

```
                    FUNCTION  beep() - emit a beep sound

    SYNOPSIS  beep();

 DESCRIPTION  beep() outputs a single beep to both the remote and local
              terminal if the user does not have the ’silent mode’ bit
              active in
                bu_utoggles
                .

RETURN VALUE  beep() returns always 0.
```

1.103 breceive

```
                    FUNCTION  breceive() - receive file(s) from remote

    SYNOPSIS  breceive(dir);

              dir - a string denoting the directory to which the file(s)
                    should be received to.

 DESCRIPTION  breceive() receives files to the directory specified in dir
              with the user's currently selected transfer protocol.

RETURN VALUE  breceive() returns always 0.

     EXAMPLE  int main() {
                char d;

                if(bu_access & 0xff) {
                  printf("Okay, sir boss, you can upload where you like.\n");
                  d=input("Enter directory: ",255,0);
                  printf("Ok, receiving...\n");
                  breceive(d);
                } else printf("You're not a sysop!\n");
              }

    SEE ALSO
                bsend()
```

1.104 bsend

```
                    FUNCTION  bsend() - send file(s) to remote

    SYNOPSIS  bsend(listfile);

              listfile - a string containing the name of the file in which
              the files to be sent are listed.

 DESCRIPTION  bsend() sends the files listed in listfile to the remote user
              with the user's currently selected transfer protocol. In the
              listfile, a single line should contain one file to be sent.

RETURN VALUE  bsend() returns true if all went OK, false otherwise.

     EXAMPLE  int main() {
                int f;

                printf("I'll send you my autoexec.bat and config.sys...\n");

                f=fopen("fubba.fil","wt");
                fprintf(f,"c:/autoexec.bat\n");
                fprintf(f,"c:/config.sys\n");
                fclose(f);

                bsend("fubba.fil");
              }
    SEE ALSO
                breceive()
```

1.105 closedir

```
                    FUNCTION  closedir() - close a directory opened with opendir

    SYNOPSIS  closedir();

 DESCRIPTION  closedir() closes the directory stream opened with a opendir
              call. After all necessary directory reads have been done to
              a opened directory, a closedir() call should be made to ensure
              that the memory allocated to the directory entries is freed.

RETURN VALUE  closedir() returns always 0.

     EXAMPLE  int main() {
                /* quick-and-dirty directory lister
                 *
                 * an example of opendir(), readdir(), closedir() and
                 * rewinddir().
                 */

                char spec,entry,file;
                int size,filecount;

                spec=input("Enter filespec: ",255,0);
                opendir(spec);

                printf("Directory of %s...\n\n",spec);

                do {
                  redisp=0; filecount=0;
                  while(strlen(entry=readdir())) {
                    file=copy(entry,1,pos(entry,"\2"));
                    delete(entry,1,pos("\2",entry));
                    size=copy(entry,1,pos(entry,"\2"));
                    printf("%.30s %i\n",file,size);
                    ++filecount;
                  }

                  printf("Total %u files.\n",filecount);
                  printf("Hit space to view the list again.\n");
                  while(!kbhit());
                  if(getch()==" ") {
                    rewinddir();
                    redisp=1;
                  }
                } while(redisp);

                printf("Have a nice day!\n");
                closedir();
              }

    SEE ALSO
```

```
                opendir
                ,
                readdir
                ,
                rewinddir
```

1.106 copy

```
                    FUNCTION  copy() - copy a part of string

    SYNOPSIS  copy(buf, start, len);

              buf   - the string to copy from
              start - an integer denoting the first character to start
                      copying from
              len   - an integer denoting the amount of characters to be
                      copied

 DESCRIPTION  copy() copies at maximum len characters from the string
              buf, starting from the startth character. The copying
              stops if the end of buf-string is reached.

RETURN VALUE  copy() returns the copied string.

    EXAMPLES  a=copy("foobar",1,3);
              (returns "foo")

              b=copy("It is true: BBBS rules, by far!",13,10);
              (returns "BBBS rules")

    SEE ALSO
                delete()
                ,
                pos()
```

1.107 delay, _delay

```
    FUNCTION  delay(), _delay() - halt program execution for a specified
              amount of time

    SYNOPSIS  delay(time);

              time - the time to delay, in tenths of second

              _delay(time);

              time - the time to delay, in milliseconds

 DESCRIPTION  Both delay() and _delay() completely halt the program
              execution for the specified amount of time. With _delay(),
              the time delayed is just an approximation and cannot be used
              for precise timing.
```

```
RETURN VALUE  Both delay() and _delay() return always 0.

     EXAMPLE  int main() {
                if(!bu_resume) {
                  printf("For not writing a resume, there will be a 5 ");
                  printf("second delay!\n");
                  delay(50);
                }
              }
```

1.108 delete

```
    FUNCTION  delete() - deletes a part of a string

    SYNOPSIS  delete(buf, start, len);

              buf   - the string to delete from
              start - an integer denoting the first character to start
                      deleting from
              len   - an integer denoting the amount of characters to
                      delete

 DESCRIPTION  delete() deletes at maximum len characters from the
              string buf, starting from character start. If the end
              of buf is reached, then the deleting will stop.

RETURN VALUE  delete() returns the operated string.

    EXAMPLES  a=delete("foobar",4,3);
              (returns "foo");

              b=delete("foobar",1,3);
              (returns "bar");

              z=delete("merry christmas, Mr. B!",7,11);
              (returns "merry Mr. B!")
```

1.109 exec

```
                    FUNCTION  exec() - execute another script

    SYNOPSIS  exec(file);

              file - a string containing the name of the script to run

 DESCRIPTION  exec() executes another script. file must be a valid
              script name. A call to exec() aborts the execution of the
              current script - there is no way to return to the script
              that invoked exec(). If this is what you want to do, use
              spawn() instead.
              It is also possible to pass parameters to the main-function
              of the script being called. This is accomplished by adding
              the parameters to the file-parameter of exec(), separated with
              the ASCII character 1.

RETURN VALUE  exec() does not return.

     EXAMPLE  exec("c:/bbs/bz/newscript");

    SEE ALSO
                spawn()
```

1.110 exit

```
    FUNCTION  exit() - quit the current script

    SYNOPSIS  exit(error);

              error - an integer containing the error code that should be
                      returned

 DESCRIPTION  exit() exits the current script with the error code
              specified in error. exit() does not do any checking for
              open files or loaded users. Always be sure that all files
              are closed and freeuser() called before calling exit().

RETURN VALUE  exit() does not return.

     EXAMPLE  exit(666);
```

1.111 fclose

```
    FUNCTION  fclose() - close a file

    SYNOPSIS  fclose(handle);

              handle - the filehandle that should be closed

 DESCRIPTION  fclose() closes the file associated to filehandle handle.
              All opened files should be closed via a call to fclose()
              before the script is exited.

RETURN VALUE  fclose() returns always 0.

     EXAMPLE  int main() {
                int f;

                if((f=fopen("fubba.fil","wt"))==-1) {
                  printf("Ugh!\n");
                  exit(-1);
                }
                fprintf(f,"foo!\n");
```

```
                  fprintf(f,"bar!\n");
                  fclose(f);
                  exit(1);
                }
```

1.112 feof

```
    FUNCTION  feof() - determine if a file pointer is at end of file

    SYNOPSIS  feof(handle);

              handle - the file handle of the file to check

 DESCRIPTION  feof() determines if end of file has been reached in file
              associated to the filehandle handle. If a file is opened
              in text mode (see fopen), then also control-z-character
              (normal end-of-file-character) is interpreted as end of
              file.

RETURN VALUE  feof() returns true, if end of file has been reached,
              otherwise it returns false.

     EXAMPLE  int main() {
                 int f;

                 if((f=fopen("fubba.fil","rt"))==-1) {
                   printf("Ugh!\n");
                   exit(-1);
                 }

                 /* display fubba.fil... */
                 while(!feof(f) && bv_carrier) printf("%s\n",fgets(f));
                 fclose(f);
                 exit(1);
              }
```

1.113 fflush

```
    FUNCTION  fflush() - flush a file into the disk

    SYNOPSIS  fflush(handle);

              handle - the file handle to flush

 DESCRIPTION  fflush() flushes all the buffers of the file associated
              with the file handle handle, causing all unwritten data
              be written to disk.

RETURN VALUE  fflush() returns always 0.

     EXAMPLE  f=fopen("testfile.tmp","wt");
              fputc(65,f);
              fputc(66,f);
              fputc(67,f);
              printf("fputc\n");
              getch();

              fflush(f);
              printf("fflush\n");
              getch();

              fclose(f);
              printf("fclose\n");
              getch();
              remove("testfile.tmp");
```

1.114 fgetc

```
                    FUNCTION  fgetc() - read a character from a file

    SYNOPSIS  fgetc(handle);

              handle - the file handle of the file to read the character
                       from

 DESCRIPTION  fgetc() reads a single character from the file associated
              to the file handle handle.

RETURN VALUE  fgetc() returns the ASCII value of the character that is
              read, or -1 if end of file has been reached.

     EXAMPLE  int main() {

                int i, f;
                char bar;

                f=fopen("c:/bbbs/bzc.exe","rb");
                i=fopen("c:/bbbs/test","wb");

                while ((bar=fgetc(f))!=-1) fputc(bar,i);

                fclose(i);
                fclose(f);

              }

    SEE ALSO
                fputc
                ,
                fgets
```

1.115 fgets

```
                 FUNCTION  fgets() - read a string from a file

    SYNOPSIS  fgets(handle);

              handle - the file handle of the file to read the string from

 DESCRIPTION  fgets() reads a single line of characters from the file
              associated to filehandle handle. A line is considered as
              a sequence of characters followed by a CR+LF-pair.

RETURN VALUE  fgets() returns the actual string read.

     EXAMPLE  fubba=fgets(f);

    SEE ALSO
                fgetc
                ,
                fputc
```

1.116 fopen

```
    FUNCTION  fopen() - open a file

    SYNOPSIS  fopen(filename, mode);

              filename - a string containing the name of the file to open
              mode     - a string containing the mode that the file
                         should be opened in

 DESCRIPTION  fopen() opens a file specified in filename. All files
              must be opened before any input or output can be made to or
              from them. The mode that the file is opened in depends on
              the string mode. Possible modes are:

              "r"     Open the file for reading. The file must exist.

              "w"     Open the file for writing only. If the file doesn't
              exist, it will be created. Otherwise, it will be truncated
              to 0 bytes.

              "a"     Open the file for appending. If the file doesn't
              exist, it will be created. Otherwise, the file pointer is
              set to the end of file. All writing will be done to the end
              of file.

              "r+"    Open the file for both reading and writing. The
              file must exist.

              "w+"    Open the file for both reading and writing. If the
              file exists, it will be truncated to 0 bytes, otherwise a
              new file will be created.

              "a+"    Open the file for reading and appending. If the
              file does not exist, it will be created. All writing will
              be done to the end of file.

              There are also two additional mode flags that need to be
              appended to the end of the mode-string; t for text mode and
              b for binary mode. In text mode CR characters are skipped
              on input and newline characters converted to CR/LF-pairs on
              output. If the last character of the file is a Ctrl-Z, then it
              and everything following it will be discarded on input. In
              binary mode, none of these transformations are used.

RETURN VALUE  fopen() returns a file handle if the open is successful,
              otherwise it returns -1.

     EXAMPLE  int main() {
                int f;

                if((f=fopen("fubba.fil","wt"))==-1) {
                  printf("Ugh!\n");
                  exit(-1);
                }
                fprintf(f,"foo!\n");
                fprintf(f,"bar!\n");
                fclose(f);
                exit(1);
              }
```

1.117 fprintf

```
    FUNCTION  fprintf() - write formatted text into a file

    SYNOPSIS  fprintf(handle, format {,argument{,argument...}});

              handle - the file handle to write the formatted text into.
              format - a format specifier, see printf() for details.

              fprintf() can have a number of additional arguments,
              depending on the format specifier. For every format
              specifier in format, an argument should be given.

 DESCRIPTION  fprintf() performs formatted output to the file specified by
              the file handle handle.

RETURN VALUE  fprintf() does not return anything.

     EXAMPLE  int main() {
                 int f;
                 char fubba;

                 f=fopen("fubba.fil","wt");
                 fubba=input("Fubba: ",10,0);
                 fprintf("3+3=%i  Fubba=%s\n",3+3,fubba);
                 fclose(f);
              }
    SEE ALSO  printf(), sprintf()
```

1.118 fputc

```
                   FUNCTION  fputc() - write a single byte into a file

    SYNOPSIS  fputc(byte, handle);

              byte   - the ASCII value of the byte to write.
              handle - the file handle to write the byte into.

 DESCRIPTION  fputc() writes the byte byte into the file associated
              with the file specified by file handle handle.

RETURN VALUE  fputc() returns always 0.

     EXAMPLE  int main() {

                int i, f;
                char bar;

                f=fopen("c:/bbbs/bzc.exe","rb");
                i=fopen("c:/bbbs/test","wb");

                while ((bar=fgetc(f))!=-1) fputc(bar,i);

                fclose(i);
                fclose(f);

              }

    SEE ALSO
                fgetc
```

1.119 freeuser

```
                   FUNCTION  freeuser() - return normal bu-variable values after  ↩
                      a
              loaduser() call.

    SYNOPSIS  freeuser();

 DESCRIPTION  freeuser() returns the bu-variable values to the ones they
              were before a call to loaduser(). It should always be
              called before exiting, if a loaduser() call has been made.

RETURN VALUE  freeuser() returns always 0.

     EXAMPLE  int main() {
                 int i;

                 for(i=0; !loaduser(i); ++i) {
```

```
                printf("User #%04u: %s\n",i,bu_name);
                freeuser();
              }
            }

    SEE ALSO
              loaduser()
```

1.120 fseek

```
                    FUNCTION  fseek() - position the pointer of a file handle

    SYNOPSIS  fseek(handle, offset, whence);

              handle - the file handle of the file the pointer of which
              should be positioned.
              offset - an integer determining the amount of bytes to move
              the file pointer.
              whence - an integer determining the start position to start
              the positioning:

              0 (SEEK_SET) - the file beginning
              1 (SEEK_CUR) - the current file pointer position
              2 (SEEK_END) - the end of file

 DESCRIPTION  fseek() moves the file pointer offset bytes, starting
              from the file position determined by whence. For text mode
              files, offset should be either 0, or a value returned by
              ftell().

RETURN VALUE  fseek() returns true, if an error is encountered.

    EXAMPLES  Seek to the 10th byte of the file:
              fseek(f, 10, 0);

              Seek 10 bytes forward from the current position:
              fseek(f, 10, 1);

              Seek 10 bytes backward from the end of file:
              fseek(f, -10, 2);

    SEE ALSO
                ftell()
```

1.121 ftell

```
                    FUNCTION  ftell() - return the file pointer position

    SYNOPSIS  ftell(handle);

              handle - the file handle of the file to return the file
              pointer position from.
```

DESCRIPTION ftell() determines and returns the current position of the file pointer and returns an integer determining the position as bytes from the beginning of the file, if the file is opened in binary mode. In text mode this is not so, but the value can nonetheless be used in subsequent calls to fseek().

RETURN VALUE ftell() returns an integer determining the current file pointer position.

EXAMPLE

```
fseek(f, 0, 2);
printf("The file is %u bytes long.\n",ftell(f));
```

SEE ALSO

fseek()

1.122 getch and _getch

FUNCTION getch(), _getch() - read a keypress without waiting

SYNOPSIS

```
getch();

_getch(whence);
```

whence - an integer denoting what kind of keypresses should be read, as follows:

whence	action
0 (GETCH_BOTH)	Read from both local and remote
1 (GETCH_LOCAL)	Read only from local keyboard
2 (GETCH_REMOTE)	Read only from remote keyboard

DESCRIPTION Both getch() and _getch() read a single character from the (specified) input buffer, if one is available. With _getch() the whence value 0 (GETCH_BOTH) should not be used. Instead, a getch() call should be made. Note that getch() nor _getch() will not wait for a character to become available. If that is what you want to do, you should use input() instead.

RETURN VALUE Both getch() and _getch() return the character that was read or the value (character) 255 if none is available.

EXAMPLES

```
printf("The key ’%c’ was pressed.\n",getch());
printf("User pressed the key ’%c’.\n",_getch(2));
```

SEE ALSO

input()
,
kbhit()

1.123 getenv

```
                    FUNCTION  getenv() - get the value of an environment variable.

    SYNOPSIS  getenv(env);

              env - a string containing the name of the environment variable
              to be read.

 DESCRIPTION  getenv() reads the value of the specified environment variable
              and returns it. Note that getenv() reads BBBS environment
              variables and has nothing to do with the environment variables
              set in the OS.

              The following environment variables are set internally by BBBS:

              Environment    Function
              ============================================================
              \01MEMBER      list of
                BBBS chat system
                 channels the user is a
                             member of, separated with commas.
              \01RESIGN      list of auto-join chat system channels the user
                             has resigned from, separated with commas.
              \01TARGET      the current chat system target of the user.
              \01COLORS      the number of the palette (in
                colors.bbb
                ) the
                             user is using, or "cxxxxx", where xxxxx are
                             the color letters, colors.bbb-style.

              (\01 is the ASCII character 1, not the actual string "\01")

RETURN VALUE  getenv() returns the value of the specified environment
              variable.

     EXAMPLE  printf("Your chat target is: %s\n",getenv("\1TARGET");

    SEE ALSO
                setenv()
```

1.124 getnodestatus

```
    FUNCTION  getnodestatus() - get a part of the status of a specified node.

    SYNOPSIS  getnodestatus(node, item);

              node - an integer specifying the node number to get status
              from.
              item - an integer specifying the type of information that
              should be returned.

 DESCRIPTION  getnodestatus() returns a single piece of information about
              the status of the specified node. The item-parameter value
```

is determined by what kind of information should be returned, as follows:

Item	Description	Type
0 (GNS_SPLIT)	Reserved	int
1 (GNS_BSTATUS)	bstatus-value (see below)	int
2 (GNS_ZSTATUS)	zstatus-value (see below)	int
3 (GNS_SPEED)	The bps rate of the node (0=local)	int
4 (GNS_TIME)	when-value in who (see below)	int
5 (GNS_ENDTIME)	when-value in who when downloading	int
6 (GNS_NICK)	The nickname of the user in node	char
7 (GNS_REALNAME)	The real name of the user in node	char
8 (GNS_IDLE)	The amount of idle time (see below)	int

The bstatus-value contains some toggles about the node. If bit 1 is set, then the node is in a mail session. If bit 2 is set, then the node is in groupchat via the HYDRA protocol. If bit 3 is set then the node has the "KEEPALIVE" environment set, meaning that it will not be sleep disconneted. If bit 4 is set, then the node has the "KEEPIDLE" environment set, meaning that is should be shown as "idle", even though it might not actually be so.

The zstatus-value defines the actual status of the node. It is a single integer, determining it as follows:

zstatus	Status of the node
0 (ZS_OFF)	Node is down.
1 (ZS_ACTIVE)	Node is "active".
2 (ZS_NOTAVAIL)	Node is not available.
3 (ZS_WRITING)	Node is entering a message.
4 (ZS_GRAB)	Node is grabbing (downloading messages).
5 (ZS_DOWNLOAD)	Node is downloading a file.
6 (ZS_UPLOAD)	Node is uploading a file.
7 (ZS_SYSCHAT)	Node is chatting with SysOp.
8 (ZS_DOOR)	Node has opened a door.
9 (ZS_GROUPCHAT)	Node is in groupchat.
10 (ZS_TELNET)	Node is using network (telnet, ftp...)
11 (ZS_SHELLED)	Node has shelled in to the OS.

The time value determines the login time of the current user. It is an integer, packed so that the "hour" value is multiplied with the value 256 (0x100) and the "minute" value is the added to the result.

The nick and realname values are simply strings containing the real name and the nickname of the user in the node.

The GNS_IDLE field is the UNIX time() timestamp of the moment when the node has started to be idle. Idle times of less than 120 seconds should be ignored.

RETURN VALUE getnodestatus() return value depends on the item parameter, see the table above.

```
     EXAMPLE  int main() {
                 int i;

                 for(i=1; i<bg_max_nodes+1; ++i) {
                   printf("Node #%02u: %s\n",i,getnodestatus(i,7));
                 }
              }
```

1.125 getnodemessage

```
                    FUNCTION  getnodemessage() - read a node message from a file

    SYNOPSIS  getnodemessage(nodefile);

              nodefile - the file handle of the file the node message should
              be read from.

 DESCRIPTION  getnodemessage() reads 340 bytes (a BBBS chatrec) from the
              file specified by the file handle nodefile and converts them
              into a formatted string with the message. Normally you should
              use getnodemessage() with the message.xxx-files in the BBBS

                temporary directory
                .

RETURN VALUE  getnodemessage() returns a formatted string containing the
              message from the chatrec read.

     EXAMPLE  int main() {
                int f;
                char s, fn;

                fn=sprintf("%sbbbsmsg.%u",bg_tempdir,bv_nodenumber);
                f=fopen(fn,"rb");

                while ((s=getnodemessage(f))!="") printf("%sn",s);

                fclose(f);
                fclose(fopen(fn,"wb"));
              }
```

1.126 help

```
    FUNCTION  help() - execute the AmigaGuide help file reader

    SYNOPSIS  help(file, node);

              file - a string containing the name of the help file to read.
              node - a string containing the name of the help file node to
              start reading from.

 DESCRIPTION  help() executes the BBBS's built-in AmigaGuide reader and
```

```
               loads into it the file specified in file and displays the
               node specified in node. If file is empty, then the
               standard help file (bbbshelp) is loaded.

RETURN VALUE   help() returns always 0.

     EXAMPLE   help("","Main");
               help("c:/bbs/sysop.guide","bz_lib_help");
```

1.127 input

```
                     FUNCTION  input() - read user input

    SYNOPSIS   input(prompt, length, multi);

               prompt - a string containing the prompt that should be
               displayed as a prompt.
               length - an integer determining the maximum amount of
               characters that should be accepted.
               multi  - an integer controlling the node messages and whether
               or not only a word should be read.

 DESCRIPTION   input() reads from the user at maximum length characters,
               first displaying the string prompt, using the BBBS input
               buffer. The parameter multi defines whether or not multiple
               words should be read; multiple words are read only if multi is
               false. The bit 7 of multi is an exception: if it is set, then
               no nodemessages are processed while user is typing his/her
               input.

RETURN VALUE   input() returns a string containing the user input, modified
               approriately.

    EXAMPLES   command=input("Your Command: ",50,1);
               foo=input("Enter string: ",50,0);

    SEE ALSO
                 getch()
```

1.128 kbhit

```
                     FUNCTION  kbhit() - check to see if a key has been pressed

    SYNOPSIS   kbhit();

 DESCRIPTION   kbhit() checks if a key has been pressed or not. Note that
               kbhit() will not wait for a user to press a key.

RETURN VALUE   kbhit() returns true if a key has been pressed.

     EXAMPLE   while(!kbhit()) printf("BBBS forever!\n");
    SEE ALSO
                getch()
```

1.129 loaduser

```
                    FUNCTION  loaduser() - adjust bu-variables to contain the  ↩
                        values of
              another user

    SYNOPSIS  loaduser(userno);

              userno - an integer determining the absolute user number of
              the user whose values should be loaded.

 DESCRIPTION  loaduser() adjusts the values of the bu-variables so that they
              contain the values of the user whose user number is userno.
              The bu-variables contain these values until a call to
              freeuser() is made.

RETURN VALUE  loaduser() returns a nonzero value if an error occurs,
              otherwise it returns 0.

     EXAMPLE  int main() {
                 int i;

                 for(i=0; !loaduser(i); ++i) {
                   printf("User #%04u: %s\n",i,bu_name);
                   freeuser();
                 }
              }

    SEE ALSO
                freeuser()
```

1.130 make_pcboard12sys

```
                    FUNCTION  make_pcboard12sys() - make a v12 PCBOARD.SYS control ↩
                         file

    SYNOPSIS  make_pcboard12sys(file);

              file - a string containing the name of the file that the
              control file should be created into.

 DESCRIPTION  make_pcboard12sys() creates a door control file PCBOARD.SYS,
              that is compliant with version 12. This can be used with
              various external door programs.

RETURN VALUE  make_pcboard12sys() returns always 0.

     EXAMPLE  make_pcboard12sys("pcboard.sys");
    SEE ALSO
                make_pcboard14sys()
```

1.131 make_pcboard14sys

```
                    FUNCTION  make_pcboard14sys() - make a v14 PCBOARD.SYS control ↩
                        file

    SYNOPSIS  make_pcboard14sys(file);

              file - a string containing the name of the file that the
              control file should be created into.

 DESCRIPTION  make_pcboard14sys() creates a door control file PCBOARD.SYS,
              that is compliant with version 14. This can be used with
              various external door programs.

RETURN VALUE  make_pcboard14sys() returns always 0.

     EXAMPLE  make_pcboard14sys("pcboard.sys");

    SEE ALSO
                make_pcboard12sys()
```

1.132 opendir

```
                    FUNCTION  opendir() - opens a directory for reading

    SYNOPSIS  opendir(filespec);

              filespec - a string containing the filespec of the files to
              read.

 DESCRIPTION  opendir() opens a directory for reading files that match the
              wild card expression in filespec. If you want to open a
              directory different than the current directory, simply list it
              before the wild card expression.

RETURN VALUE  opendir() returns true if there was no error, false otherwise.

     EXAMPLE  int main() {
                /* quick-and-dirty directory lister
                 *
                 * an example of opendir(), readdir(), closedir() and
                 * rewinddir().
                 */

                char spec,entry,file;
                int size,filecount;

                spec=input("Enter filespec: ",255,0);
                opendir(spec);
                printf("Directory of %s...\n\n",spec);

                do {
                  redisp=0; filecount=0;
                  while(strlen(entry=readdir())) {
                    file=copy(entry,1,pos(entry,"\2"));
                    delete(entry,1,pos("\2",entry));
                    size=copy(entry,1,pos(entry,"\2"));
                    printf("%.30s %i\n",file,size);
                    ++filecount;
                  }

                  printf("Total %u files.\n",filecount);
                  printf("Hit space to view the list again.\n");
                  while(!kbhit());
                  if(getch()==" ") {
                    rewinddir();
                    redisp=1;
                  }
                } while(redisp);

                printf("Have a nice day!\n");
                closedir();
              }
```

```
    SEE ALSO
                readdir()
                ,
                rewinddir()
                ,
                closedir()
```

1.133 parsecom

```
    FUNCTION  parsecom() - parse input for commands

    SYNOPSIS  parsecom(cmd_list, input);

              cmd_list - list of available commands, separated with slashes.
              input    - the input string.

 DESCRIPTION  parsecom() parses the input specified in input to see if it
              matches any of the commands specified in cmd_list. The
              commands in cmd_list should be separated with slashes. Also,
              they should be written in dual case, so that the necessary
              (unique) part of the command is in upper case, the rest in
              lower case. cmd_list should contain no more than 254 commands.

RETURN VALUE  parsecom() returns an integer denoting the number of the
              command that the input matches, otherwise it returns 255. If
              the input string is empty, parsecom() returns 0.

     EXAMPLE  cmdno=parsecom("Quit/Read/UGH",input("Com: ",40,1));
```

1.134 pos

```
                    FUNCTION  pos() - search for a substring in a string

    SYNOPSIS  pos(haystack, needle);

              haystack - the string to be scanned.
              needle   - the substring to be located.

 DESCRIPTION  pos() seeks the first occurance of needle in haystack.

RETURN VALUE  If needle is found in haystack, then the starting character
              number in haystack is returned. Otherwise, pos() returns 0.

     EXAMPLE  if(pos(bv_groups,"sysop")) printf("G'day sir boss!\n");

    SEE ALSO
                regexp()
```

1.135 printf

```
                    FUNCTION  printf() - write formatted output into the screen

    SYNOPSIS  printf(format {,argument{,argument...}});

              format   - a format specifier

 DESCRIPTION  printf() writes formatted output into the screen. Characters
              in format are written directly to the screen until a format
              specification is found. A format specification has the
              following format:

              %{flags}{width}{.precision}type

              where items in curly barckets are optional. For each format
              specification, an extra parameter has to be added to the
              printf() call. This parameter is then formatted according to
              the format specification and written to the screen.

              The flags field in format specification is used to modify
              the formatting. The following flags are available:

              Flag   Meaning
              ==============================================================
              -      Left-justify the result in the field. Otherwise, the
                     result is right-justified. The value will be padded
                     with blanks on the right. If - is not specified, then
                     the result is padded to the left with blanks or zeros,
                     depending on the type in the format specification.

              +      Always insert a sign (+ or -), when a number is being
                     output.

              0      If a number is being output, pad the field with '0'
```

characters instead of blanks. This flag is ignored if a string is being output or if the - flag is specified.

If the length of the result is less than the width specified in the format specifier, then it is padded with blanks or '0' characters. If the - flag is used, then the field is padded to the right, otherwise to the left. If the length of the result is greater than the width-field in format specifier, then the field is NOT truncated.

The precision-field can only be used with strings. It determines the maximum length of the string for this field. If the string is longer, then it will be truncated to be the correct length.

The type-field is a single character determining the type of conversion done the the argument matching this format specification. It can be one of the following:

Type	Output and the type of the required extra argument
%	The '%' character. This type does not need an extra argument.
c	A single character.
d	A signed decimal integer.
i	A signed decimal integer.
o	An unsigned octal integer.
s	A string
u	An unsigned decimal integer
x	An unsigned hexadecimal integer, using lower case.
X	An unsigned hexadecimal integer, using upper case.

RETURN VALUE printf() returns the number of characters displayed.

EXAMPLES

```
printf("Foobar!\n");
printf("Hello, %s!\n",bu_name);
printf("49=0x%04X=%4oo",49,49);
printf("%-33s%s",bu_name,bu_city);
printf("%4.4s","123456");
```

SEE ALSO

fprintf()
,
sprintf()

1.136 readdir

FUNCTION readdir() - read a single directory entry

SYNOPSIS readdir();

DESCRIPTION readdir() reads the next directory entry matching the filespec given when opendir() was called. If opendir() has not been called, readdir() will always fail.

```
RETURN VALUE  If successful, readdir() returns a string containing the
              filename, file size and the file date, separated with the
              ASCII character number 2. Otherwise, readdir() returns an
              empty string.

     EXAMPLE  int main() {
                /* quick-and-dirty directory lister
                 *
                 * an example of opendir(), readdir(), closedir() and
                 * rewinddir().
                 */

                char spec,entry,file;
                int size,filecount;

                spec=input("Enter filespec: ",255,0);
                opendir(spec);

                printf("Directory of %s...\n\n",spec);

                do {
                  redisp=0; filecount=0;
                  while(strlen(entry=readdir())) {
                    file=copy(entry,1,pos(entry,"\2"));
                    delete(entry,1,pos("\2",entry));
                    size=copy(entry,1,pos(entry,"\2"));
                    printf("%.30s %i\n",file,size);
                    ++filecount;
                  }

                  printf("Total %u files.\n",filecount);
                  printf("Hit space to view the list again.\n");
                  while(!kbhit());
                  if(getch()==" ") {
                    rewinddir();
                    redisp=1;
                  }
                } while(redisp);

                printf("Have a nice day!\n");
                closedir();
              }

    SEE ALSO
                opendir()
                ,
                rewinddir()
                ,
                closedir()
```

1.137 regexp

```
                    FUNCTION  regexp() - check to see if a string matches a  ↩
                       regular
              expression
```

```
    SYNOPSIS  regexp(exp, str);

              exp - a string containing the regular expression to match.
              str - a string containing the string to check.

 DESCRIPTION  regexp() executes
                regular expression
                 matching, using exp as
              the regular expression and str as the string to check. exp has
              to be a valid regular expression.

RETURN VALUE  regexp() returns true, if str matches the expression exp,
              otherwise it returns false.

     EXAMPLE  int main() {
                // a quick-and-dirty grep program:

                int f,c=0;
                char fn,exp,line;

                exp=input("Exp : ",255,0);
                fn=input("File:",255,1);

                if((f=fopen(fn,"rt"))==-1) {
                  printf("grep: no such file '%s'\n",fn);
                  exit(-1);
                }

                while(!feof(f)) {
                  ++c;
                  if(regexp(exp, line=fgets(f)))
                    printf("%i: %s\n",c,line);
                }
              }

    SEE ALSO
                pos()
```

1.138 remove

```
                    FUNCTION  remove - delete a file

    SYNOPSIS  remove(file);

              file - a string containing the name and path of the file
              to delete.

 DESCRIPTION  remove() deletes the file specified by the parameter file.
              The file-parameter cannot contain
                wildcards
                . If you want
              to delete multiple files, you should do this with the OS
              shell by using the system() function.
```

```
RETURN VALUE  remove() returns always 0.

     EXAMPLE  remove("c:/bbs/fubba.fil");

    SEE ALSO
                system
                ,
                rename
```

1.139 rename

```
                    FUNCTION  rename() - rename a file

    SYNOPSIS  rename(old, new);

              old - a string containing the name of the file to be renamed.
              new - a string containing the name that the file should be
              renamed to.

 DESCRIPTION  rename() renames the file old to new.

RETURN VALUE  rename() returns 0 if renaming was successful, nonzero if
              error occurred.

     EXAMPLE  rename("fubba.fil","foobar.fil");

    SEE ALSO
                remove()
```

1.140 rewinddir

```
                    FUNCTION  rewinddir() - move the directory pointer to the  ↩
                        first
              directory entry

    SYNOPSIS  rewinddir();

 DESCRIPTION  rewinddir() adjusts the directory pointer so that it points to
              the first file in the directory matching the filespec given to
              opendir(). rewinddir() should not be called until after
              opendir() is called to open a directory.

RETURN VALUE  rewinddir() returns always 0.

     EXAMPLE  int main() {
                /* quick-and-dirty directory lister
                 *
                 * an example of opendir(), readdir(), closedir() and
                 * rewinddir().
                 */

                char spec,entry,file;
                int size,filecount;

                spec=input("Enter filespec: ",255,0);
                opendir(spec);

                printf("Directory of %s...\n\n",spec);

                do {
                  redisp=0; filecount=0;
                  while(strlen(entry=readdir())) {
                    file=copy(entry,1,pos(entry,"\2"));
                    delete(entry,1,pos("\2",entry));
                    size=copy(entry,1,pos(entry,"\2"));
                    printf("%.30s %i\n",file,size);
                    ++filecount;
                  }

                  printf("Total %u files.\n",filecount);
                  printf("Hit space to view the list again.\n");
                  while(!kbhit());
                  if(getch()==" ") {
                    rewinddir();
                    redisp=1;
                  }
                } while(redisp);

                printf("Have a nice day!\n");
                closedir();
              }

    SEE ALSO
                opendir()
                ,
                readdir()
                ,
                closedir()
```

1.141 seekforuser

```
                    FUNCTION  seekforuser() - find the user number of an user

    SYNOPSIS  seekforuser(name);

              name - a string containing the name of the user whose user
              number should be searched for.

 DESCRIPTION  seekforuser() finds the absolute user number of the user name.
              This number can be used in subsequent calls to loaduser().

RETURN VALUE  seekforuser() returns the user number of name, or 65535 if the
              user is not found.

     EXAMPLE  loaduser(seekforuser("Gleb Butcher"));

    SEE ALSO
```

```
                loaduser()
```

1.142 sendnode

```
                    FUNCTION  sendnode() - send a node message

    SYNOPSIS  sendnode(from, to, message, filter, prefix);

              from    - an integer denoting the node from which the node
              message should originate.
              to      - an integer denoting the node to which the node
              message should be sent to.
              message - a string containing the actual message text.
              filter  - an integer containing the filter level of the
              message.
              prefix  - a string containing the prefix of the message.

 DESCRIPTION  sendnode() sends a node message from node from to the node to.
              The bits in filter define what type of node message it is. If
              the bit is unset, it means that it can be interpreted to be a
              message of that filter level. Therefore, messages with filter
              level 255 should not be used, as they can not be filtered out.

              Bit  Level
              ========================================
              0    Chat feeling
              1    Login / Logout information
              2    "New message entered"-information
              3    Public chat message
              4    Private chat message

              prefix determines what kind of prefix the message should have
              when BBBS formats it into a message. Available prefixes are:

              Prefix               Message type
              ============================================================
              #[* ] :nick{ color}  A feeling with focus nick, displayed
                                   with color color.
              #channel:nick        A chat message in channel #channel,
                                   written by user with nickname nick.
              nick                 A private message from user with nickname
                                   nick.
              *channel:nick        An informative message to/about channel
                                   #channel form user with nickname nick.

              There are some node messages that have a special meaning:

              Message      Action taken when a node receives this message
              ==============================================================
              \02CLICK     Sends back the message "\02HUM"
              \02UDATA     Re-reads the user record
              \02RGROUP    Re-reads the
                group control file
                 'groups'.
              \02RCHAT     Re-reads the
```

```
                chat channels
                 file 'channels.bbb'.

              (In the above, \02 is the ASCII character number 2, not the
              absolute string "\02".)

RETURN VALUE  sendnode() returns always 0.

     EXAMPLE  sendnode(bv_nodenumber,1,"Hello!",255,bn_nick);

    SEE ALSO
                getnodemessage()
```

1.143 setenv

```
                    FUNCTION  setenv() - set an environment variable

    SYNOPSIS  setenv(env, value);

              env   - a string containing the name of the environment variable
              to set.
              value - a string containing the value that the environment
              variable should be set to.

 DESCRIPTION  setenv() changes the value of the environment variable env to
              be contain value. If the environment variable does not exist,
              then it will be created. Note that like getenv(), setenv()
              deals with BBBS's internal environment variables, and has
              nothing to do with the environment variables of the OS.

              If you do not wish to have a environment variable listed with
              the "u set" command, then use the ASCII character 1 as the
              first character of the name.

RETURN VALUE  setenv() returns always 0

    EXAMPLES  setenv("FUBBA","t");
              setenv("\1INVISIBLE","foo!");

    SEE ALSO
                getenv()
```

1.144 spawn

```
                    FUNCTION  spawn() - execute another script.

    SYNOPSIS  spawn(file);

              file - a string containing the name of the script to execute.

 DESCRIPTION  spawn() executes another script. The execution of the
              program invoking spawn() will continue when the executed
              program exits.

              It is also possible to pass parameters to the main-function
              of the script being called. This is accomplished by adding
              the parameters to the file-parameter of spawn(), separated with
              the ASCII character 1.

RETURN VALUE  spawn() returns always 0.

     EXAMPLE  spawn("c:/bbs/bz/newscript");

    SEE ALSO
                exec()
```

1.145 sprintf

```
                    FUNCTION  sprintf() - write formatted text into a string.

    SYNOPSIS  sprintf(format {,argument{,argument...}});

              format - a format specifier, see printf() for details

              sprintf() can have a number of additional arguments,
              depending on the format specifier. For every format
              specifier in format, an argument should be given.

 DESCRIPTION  sprintf() performs formatted output and returns it.

RETURN VALUE  sprintf() returns the formatted string.

     EXAMPLE  bar=sprintf("49=0x%04X=%oo",49,49);

    SEE ALSO
                printf()
                ,
                fprintf()
```

1.146 stlocase

```
                    FUNCTION  stlocase() - convert a string into lower case

    SYNOPSIS  stlocase(str);

              str - the string to be converted

 DESCRIPTION  stlocase() converts a string into lower case, taking into
              account the character set that the user is using.

RETURN VALUE  stlocase() returns str in lower case.

     EXAMPLE  foo=stlocase("eLiTeS SuCK!");
    SEE ALSO
                stupcase()
```

1.147 strcat

```
                    FUNCTION  strcat() - join two strings

    SYNOPSIS  strcat(str1, str2);

              str1 - a string
              str2 - the string that should be joined to str1

 DESCRIPTION  strcat() joins str2 into str1.

RETURN VALUE  strcat() returns a string consisting of str1 and str2.

     EXAMPLE  foo=input("What is your name? ",50,1);
              printf("%s!\n",strcat("Hello, ",foo));

    SEE ALSO
                sprintf()
```

1.148 strlen

```
    FUNCTION  strlen() - calculate the length of a string

    SYNOPSIS  strlen(str);

              str - the string the length of which should determined.

 DESCRIPTION  strlen() calculates the length of str and returns it.

RETURN VALUE  strlen() returns an integer determining the length of str in
              characters.

     EXAMPLE  printf("You are %u chars long!\n",strlen(bu_name));
```

1.149 stupcase

```
                    FUNCTION  stupcase() - convert a string into upper case.

    SYNOPSIS  stupcase(str);

              str - the string to be converted

 DESCRIPTION  stupcase() converts a string into upper case, taking into
              account the character set that the user is using.

RETURN VALUE  stupcase() returns str in upper case.
```

```
     EXAMPLE  foo=stupcase("i want to yell!");

    SEE ALSO
                stlocase()
```

1.150 system

```
                    FUNCTION  system() - execute an operating system command

    SYNOPSIS  system(command, swap);

              command - a string containing the OS command to be executed.
              swap    - an integer determining if swapping should be done.

 DESCRIPTION  system() drops into OS, executes the OS command command and
              returns back to the program. If swap is true, then BBBS will
              be swapped to disk or EMS/XMS to save memory for the OS
              command execution in such environments in which this is
              possible to do.

RETURN VALUE  system() returns an integer determining the errorlevel value
              as returned by the OS.

     EXAMPLE  printf("Errorlevel=%u",system("foo.exe",1));

    SEE ALSO
                bbbs()
                ,
                exec()
                ,
                spawn()
```

1.151 time

```
    FUNCTION  time() - return seconds since 1.1.1970, 00:00 UTC

    SYNOPSIS  time();

 DESCRIPTION  time() returns the amount of seconds that have passed
              since 1.1.1970 at 00:00 UTC.

RETURN VALUE  time() returns an integer determining the amount of
              seconds that have passed.
```

1.152 tictactoe

```
    FUNCTION  tictactoe() - execute the game of TicTacToe

    SYNOPSIS  tictactoe(mode, mark);
```

```
             mode - An integer determining the mode of play.
             mark - The mark to be used.

 DESCRIPTION tictactoe() executes the BBBS's internal TicTacToe game. The
             game is played in mode mode, with the mark setup mark. mode 1
             and mark 0 is the normal local versus remote-tictactoe.

RETURN VALUE tictactoe() returns an integer from 1 to 5, determining the
             result of the game, as follows:
             1 - Local user quit the game.
             2 - Remote user quit the game.
             3 - Local user won.
             4 - Remote user won.
             5 - Carrier was dropped.

    EXAMPLES tictactoe(1,0);
             tictactoe(2,1);
             tictactoe(2,2);
```

1.153 yellsysop

```
    FUNCTION yellsysop() - page the system operator

    SYNOPSIS yellsysop(ask);

             ask - an integer denoting whether or not a reason should be
             asked.

 DESCRIPTION yellsysop() pages the system operator. If ask is true, then a
             chat reason will be asked.

RETURN VALUE yellsysop() returns true if the SysOp answered to the page
             request.

     EXAMPLE yellsysop(1);
```

1.154 BBBS variables

```
                The following global variables are defined in BZ run-time library:

        Variable name           Type         Index   Constant
        ========================================================

        Conference variables:

                 bc_count_of_co
                       int                  *

                 bc_post_conf
                       int                  *

                 bc_resume_conf
```

```
               int                  *

         bc_fileinfo_co
               int                  *

         bc_lastread
               int          *       *

         bc_status
               int          *       *

         bc_confname
               char         *       *

         bc_description
               char         *       *

         bc_ulastread
               int          *

         bc_ustatus
               int          *

         bc_uaccess
               int          *

Function variables:

         bf_timleft
               int                  *

         bf_hour
               int                  *

         bf_min
               int                  *

         bf_sec
               int                  *

         bf_year
               int                  *

         bf_month
               int                  *

         bf_day
               int                  *

Global config:

         bg_bbbs_name
               char                 *

         bg_sysop_name
               char                 *
```

bg_closed_pass	char	*
bg_grabfile	char	*
bg_maindir	char	*
bg_updir	char	
bg_tempdir	char	*
bg_menudir	char	*
bg_tickdir	char	*
bg_NetMail	char	*
bg_inbound	char	*
bg_feelingsdir	char	*
bg_scriptdir	char	*
bg_max_nodes	int	*
bg_bankmax	int	*
bg_newu_time	int	*
bg_bankrate	int	*
bg_msgrate	int	*
bg_newu_access	int	*
bg_gtoggles	int	*
bg_htoggles	int	*

```
         bg_zone
               int          *       *

         bg_net
               int          *       *

         bg_node
               int          *       *

         bg_point
               int          *       *

Local config:

         bl_modem_init1
               char                 *

         bl_modem_init2
               char                 *

         bl_modem_init3
               char                 *

         bl_modem_hangu
               char                 *

         bl_modem_busy_
               char                 *

         bl_modem_answe
               char                 *

         bl_fd_dobbs
               char                 *

         bl_logfile
               char                 *

         bl_loginlog
               char                 *

         bl_grabdir
               char                 *

         bl_menudir
               char                 *

         bl_newdir
               char                 *

         bl_spyfile
               char                 *

         bl_start_speed
               int                  *

         bl_min_speed
```

```
               int                  *

         bl_base_addres
               int                  *

         bl_irq
               int                  *

         bl_pollrate
               int                  *

         bl_answer_ring
               int                  *

         bl_ltoggles
               int                  *

Node variables:

         bn_split
               int

         bn_bstatus
               int

         bn_zstatus
               int

         bn_speed
               int

         bn_time
               int

         bn_nick
               char

         bn_realname
               char

User variables:

         bu_name
               char                 *

         bu_address
               char

         bu_city
               char

         bu_phone
               char

         bu_birth
               char
```

```
bu_ok2login
      int

bu_termcap
      int

bu_pagelength
      int

bu_charset
      int

bu_language
      int

bu_readmode
      int

bu_packtype
      int

bu_protocol
      int

bu_nodemsgfilt
      int

bu_timelimit
      int

bu_timeleft
      int

bu_timeson
      int

bu_msgleft
      int

bu_uploaded
      int

bu_downloaded
      int

bu_pmsgleft
      int

bu_puploaded
      int

bu_pdownloaded
      int

bu_ptimeson
      int
```

```
         bu_pmsgdumped
               int

         bu_pmsgread
               int

         bu_pkbup
               int

         bu_pkbdown
               int

         bu_fchecked
               int

         bu_timebank
               int

         bu_resume
               int

         bu_limits
               int

         bu_access
               int

         bu_utoggles
               int

         bu_firsttime
               int

         bu_lasttime
               int

         bu_msgread
               int

         bu_msgdumped
               int

         bu_kbup
               int

         bu_kbdown
               int

         bu_todaydown
               int

         bu_userbits
               int

Other variables:

         bv_filna
```

```
      char

bv_local_buffe
      char

bv_comhandle
      int                  *

bv_comdevice
      char                 *

bv_comstring
      char

bv_interface
      char

bv_crrdir
      char

bv_holdreal
      char

bv_hddesc
      char

bv_serna
      char                 *

bv_tempsys
      int

bv_unum
      int                  *

bv_confnro
      int                  *

bv_baud
      int

bv_realbaud
      int

bv_loginhour
      int

bv_loginmin
      int

bv_lastmsg
      int

bv_curmsg
      int

bv_firstmsg
```

```
      int

bv_lastreaded
      int

bv_newavail
      int

bv_foryou
      int

bv_serno
      int                  *

bv_vername
      char                 *

bv_ver
      char                 *

bv_brobot
      int

bv_tomenu
      int

bv_nodenumber
      int                  *

bv_reduced
      int

bv_nextevent
      int

bv_temptime
      int

bv_com
      int                  *

bv_carrier
      int

bv_outputstopp
      int

bv_quicklogin
      int

bv_paged
      int

bv_groups
      char

bv_txt
```

```
                       char          *       *
```

1.155 bc_count_of_co

```
   VARIABLE   bc_count_of_co

       TYPE   int (constant)

DESCRIPTION   Returns the total number of conferences.
```

1.156 bc_post_conf

```
   VARIABLE   bc_post_conf

       TYPE   int (constant)

DESCRIPTION   Conference number for the defined post conference.
              If no post conference is defined this will contain the
              number 65535.
```

1.157 bc_resume_conf

```
   VARIABLE   bc_resume_conf

       TYPE   int (constant)

DESCRIPTION   Conference number for the defined resume conference.
              If no resume conference is defined this will contain the
              number 65535.
```

1.158 bc_fileinfo_co

```
   VARIABLE   bc_fileinfo_co

       TYPE   int (constant)

DESCRIPTION   Conference number for the defined fileinfo conference.
              If no fileinfo conference is defined this will contain the
              number 65535.
```

1.159 bc_lastread

VARIABLE bc_lastread[x]

TYPE int (constant), indexed

DESCRIPTION Returns the amout of messages automatically put to scan when user joins to conference #x.

EXAMPLE

```
printf("User gets %u messages when joining to %s\n",
       bc_lastread[3],bc_confname[3]);
```

1.160 bc_status

VARIABLE bc_status[x]

TYPE int (constant), indexed

DESCRIPTION Returns a number which indicates what status conference #x has.

The bits are:

1	must
2	member
4	invite
8	fidoarea
16	postarea
32	allowpriv
64	nomarks
128	noreply
256	nostrip
512	allfix
1024	namefix
2048	alias
4096	allowtag
8192	agnet
16384	moderated
32768	nntp

1.161 bc_confname

VARIABLE bc_confname[x]

TYPE char (constant), indexed

DESCRIPTION Returns the name of conference #x.

1.162 bc_description

VARIABLE bc_description[x]

TYPE char (constant), indexed

DESCRIPTION Returns the description of conference #x.

1.163 bc_ulastead

VARIABLE bc_ulastread[x]

TYPE int (constant), indexed

DESCRIPTION Returns the user’s lastread pointer for conference #x.

1.164 bc_ustatus

VARIABLE bc_ustatus[x]

TYPE int (constant), indexed

DESCRIPTION Returns the user’s status bits for conference #x.

1.165 bc_uaccess

VARIABLE bc_uaccess[x]

TYPE int (constant), indexed

DESCRIPTION Returns the user’s access bits for conference #x.

1.166 bf_timleft

VARIABLE bf_timleft

TYPE int (constant)

DESCRIPTION Returns minutes left for this call for the current user.

EXAMPLE CODE

```
int main() {
  printf("You have %u min. left for this call.\n",bf_timleft);
}
```

1.167 bf_hour

VARIABLE bf_hour

TYPE int (constant)

```
DESCRIPTION   This is the hour part of the time. It returns the
              current hour.
```

1.168 bf_min

```
   VARIABLE   bf_min

       TYPE   int (constant)

DESCRIPTION   This is the minute part of the time. It returns the
              current minutes.
```

1.169 bf_sec

```
   VARIABLE   bf_sec

       TYPE   int (constant)

DESCRIPTION   This is the seconds part of the time. It returns the
              current seconds.
```

1.170 bf_year

```
   VARIABLE   bf_year

       TYPE   int (constant)

DESCRIPTION   This is year part of the date. It returns the
              current year.
```

1.171 bf_month

```
   VARIABLE   bf_month

       TYPE   int (constant)

DESCRIPTION   This is month part of the date. It returns the
              current month.
```

1.172 bf_day

```
   VARIABLE   bf_day

       TYPE   int (constant)
```

DESCRIPTION This is the day part of the date. It returns the current day.

1.173 bg_bbbs_name

VARIABLE bg_bbbs_name

TYPE char (constant)

DESCRIPTION Returns the name of the board.

1.174 bg_sysop_name

VARIABLE bg_sysop_name

TYPE char (constant)

DESCRIPTION Returns the name of User #0, which is the "real" SysOp.

1.175 bg_closed_pass

VARIABLE bg_closed_pass

TYPE char (constant)

DESCRIPTION Returns the password for accessing the system if you run a closed system. This string is crypted.

1.176 bg_grabfile

VARIABLE bg_grabfile

TYPE char (constant)

DESCRIPTION Returns the name of the grabfile as you have defined it in BCFG/4.

1.177 bg_maindir

VARIABLE bg_maindir

TYPE char (constant)

DESCRIPTION Returns the path to your main dir including a trailing slash.

1.178 bg_updir

```
   VARIABLE   bg_updir

       TYPE   char

DESCRIPTION   Returns the path to your upload directory including
              a trailing slash.

    Remarks   If you changes this variable remember to change it back to
              original value too!
```

1.179 bg_tempdir

```
   VARIABLE   bg_tempdir

       TYPE   char (constant)

DESCRIPTION   Returns the path to your work directory including
              a trailing slash.
```

1.180 bg_menudir

```
   VARIABLE   bg_menudir

       TYPE   char (constant)

DESCRIPTION   Returns the path to your main menu directory
              including a trailing slash.
```

1.181 bg_tickdir

```
   VARIABLE   bg_tickdir

       TYPE   char (constant)

DESCRIPTION   Returns the path to your tick files directory
              including a trailing slash.
```

1.182 bg_NetMail

```
   VARIABLE   bg_NetMail

       TYPE   char (constant)

DESCRIPTION   Returns the path to your NetMail directory
              including a trailing slash.
```

1.183 bg_inbound

```
   VARIABLE   bg_inbound

       TYPE   char (constant)

DESCRIPTION   Returns the path to your inbound files directory
              including a trailing slash.
```

1.184 bg_feelingsdir

```
   VARIABLE   bg_feelingsdir

       TYPE   char (constant)

DESCRIPTION   Returns the path to your feelings directory
              including a trailing slash.
```

1.185 bg_scriptdir

```
   VARIABLE   bg_scriptdir

       TYPE   char (constant)

DESCRIPTION   Returns the path to your scripts directory
              including a trailing slash. All of your scripts should
              be in this directory.
```

1.186 bg_max_nodes

```
   VARIABLE   bg_max_nodes

       TYPE   int (constant)

DESCRIPTION   Returns the number of nodes configured in your system.
```

1.187 bg_bankmax

```
   VARIABLE   bg_bankmax

       TYPE   int (constant)

DESCRIPTION   Returns the maximum amount of time allowed to store
              in the Time Bank.
```

1.188 bg_newu_time

```
   VARIABLE   bg_newu_time

       TYPE   int (constant)

DESCRIPTION   Returns the default timelimit for new users.
```

1.189 bg_bankrate

```
   VARIABLE   bg_bankrate

       TYPE   int (constant)

DESCRIPTION   Returns the ratio divider used for calculating the
              Time Bank ratio.
```

1.190 bg_msgrate

```
   VARIABLE   bg_msgrate

       TYPE   int (constant)

DESCRIPTION   Returns the ratio divider used for calculating the
              number of messages a user must write to get Download access
              from BRoboCop.
```

1.191 bg_newu_access

```
   VARIABLE   bg_newu_access

       TYPE   int (constant)

DESCRIPTION   Returns a number which sets several different user
              settings:

              0x40000000        New user has download access
              0x80000000        New user has upload access
```

1.192 bg_gtoggles

```
                   VARIABLE   bg_gtoggles

      TYPE    int (constant)

DESCRIPTION   Returns a number containg different global toggles.
              (See also
                 bg_htoggles
                 )

               Bit Meaning
               === ==========================
                0  up_down_check
                1  show_privates
                2  brobocop
                3  hippo
                4  show_empty
                5  whotext
                6  pack_messages
                7  fido_hydra
                8  fido_zedzap
                9  fido_tranx
               10  fido_unlistnode
               11  fido_unlistpoint
               12  fido_unprotnode
               13  fido_freq_answering
               14  fido_freq_calling
               15  show_sysop_in_stats
               16  bmt_check_destination
               17  disable_newu_address
               18  disable_newu_birthday
               19  users_are_hidden
               20  upload_scan
               21  poll_all_crashes
               22  delete_dup_uploads
               23  savebad
               24  savesecure
               25  bmt_log_headers
               27  no_remote_sysop
               28  bogus_save_NetMail
               29  kb_day_relative
               30  global_download
               31  nntp_gateway
               32  smtp_gateway
```

1.193 bg_htoggles

```
                    VARIABLE   bg_htoggles

      TYPE     int (constant)

DESCRIPTION    Returns a number containg more different global toggles.
               (See also
                 bg_gtoggles
                 )

               Bit Meaning
               === ==========================
                0  nntp_save_headers
                1  smtp_save_headers
                2  chat_uses_time
                3  sysnote_msg
                4  eom_to_uucp
                5  use_nodenum_not_nick
                6  allow_all_names
                7  not used
                8  grab_is_free
                9  uploader_owns_file
```

1.194 bg_zone

```
   VARIABLE   bg_zone[x]

      TYPE    int (constant), indexed

DESCRIPTION   Returns the zone part of FidoNet AKA #x.
```

1.195 bg_net

```
   VARIABLE   bg_net[x]

      TYPE    int (constant), indexed

DESCRIPTION   Returns the net part of FidoNet AKA #x.
```

1.196 bg_node

```
   VARIABLE   bg_node[x]

      TYPE    int (constant), indexed

DESCRIPTION   Returns the node part of FidoNet AKA #x.
```

1.197 bg_point

```
   VARIABLE   bg_point[x]

      TYPE    int (constant), indexed

DESCRIPTION   Returns the point part of FidoNet AKA #x.
```

1.198 bl_modem_init1

```
   VARIABLE   bl_modem_init1

       TYPE   char (constant)

DESCRIPTION   Returns the first of the three line Init string from BCFG/4.
```

1.199 bl_modem_init2

VARIABLE bl_modem_init2

TYPE char (constant)

DESCRIPTION Returns the second of the three line Init string from BCFG/4.

1.200 bl_modem_init3

VARIABLE bl_modem_init3

TYPE char (constant)

DESCRIPTION Returns the third of the three line Init string from BCFG/4.

1.201 bl_modem_hangu

VARIABLE bl_modem_hangu

TYPE char (constant)

DESCRIPTION Returns the configured string to force the modem to hang up.

1.202 bl_modem_busy_

VARIABLE bl_modem_busy_

TYPE char (constant)

DESCRIPTION Returns the configured string to set the modem busy from BCFG/4.

1.203 bl_modem_answe

VARIABLE bl_modem_answe

TYPE char (constant)

DESCRIPTION Returns the configured string to tell the modem to answer an incoming call. This is configured in BCFG/4.

1.204 bl_fd_dobbs

```
   VARIABLE   bl_fd_dobbs

       TYPE   char (constant)

DESCRIPTION   Returns the path and name of the DOBBS.BAT file used for
              FrontDoor if one is defined in BCFG/4. If none is defined, the
              string will be empty.
```

1.205 bl_logfile

```
   VARIABLE   bl_logfile

       TYPE   char (constant)

DESCRIPTION   Returns the path and the filename of the current
              node’s log file.
```

1.206 bl_loginlog

```
   VARIABLE   bl_loginlog

       TYPE   char (constant)

DESCRIPTION   Returns the path and the filename of the current node’s
              login file.
```

1.207 bl_grabdir

```
   VARIABLE   bl_grabdir

       TYPE   char (constant)

DESCRIPTION   Returns the path including a trailing slash for the
              current node’s grab directory.
```

1.208 bl_menudir

```
   VARIABLE   bl_menudir

       TYPE   char (constant)

DESCRIPTION   Returns the path including a trailing slash for the
              current node’s menu directory.
```

1.209 bl_newdir

```
   VARIABLE   bl_newdir

       TYPE   char (constant)

DESCRIPTION   Returns the path including a trailing slash for the
              current node’s temporarly upload directory. (The directory that
              BBBS uses to store files users upload while the upload is in
              process.)
```

1.210 bl_spyfile

```
   VARIABLE   bl_spyfile

       TYPE   char (constant)

DESCRIPTION   Returns the filename for the  current node’s temporarly
              spyfile. Empty if none defined.
```

1.211 bl_start_speed

```
   VARIABLE   bl_start_speed

       TYPE   int (constant)

DESCRIPTION   Returns the configured start speed from BCFG/4.
```

1.212 bl_min_speed

```
   VARIABLE   bl_min_speed

       TYPE   int (constant)

DESCRIPTION   Returns the configured minimum speed to log in from BCFG/4.
```

1.213 bl_base_addres

```
   VARIABLE   bl_base_addres

       TYPE   int (constant)

DESCRIPTION   Returns the configured base I/O address for the
              communications port for the current node.
```

1.214 bl_irq

```
   VARIABLE   bl_irq

       TYPE   int (constant)

DESCRIPTION   Returns the configured IRQ address for the communications
              port for the current node.
```

1.215 bl_pollrate

```
   VARIABLE   bl_pollrate

       TYPE   int (constant)

DESCRIPTION   Returns the configured polling rate for current node.
```

1.216 bl_answer_ring

```
   VARIABLE   bl_answer_ring

       TYPE   int (constant)

DESCRIPTION   Returns the configured number of rings to answer on for
              the current node.
```

1.217 bl_ltoggles

```
   VARIABLE   bl_ltoggles

       TYPE   int (constant)

DESCRIPTION   Returns a number containing several different settings
              that are unique to each node. The numbers are:

              Bit Meaning
              === ======================
               0  local_bell
               1  rts_cts
               2  reset_speed
               3  hangup_at_exit
               4  set_16550
               5  local_sysop_keys
               6  local_echo
               7  save_screen
               8  null_modem_login
               9  send_crashmail
              10  backdoor
              11  fast_fax
```

```
                 12  dont_check_carrier
                 13  nocarrier_is_busy
                 14  show_shell_output
                 15  allow_shell_break
                 16  slow_protocols
                 17  fax_receive_revbit
                 18  fax_send_revbit
                 19  buffered_output
                 20  rockwell_kludge
                 21  fix_rar_bug
```

1.218 bn_split

```
   VARIABLE   bn_split

       TYPE   int

DESCRIPTION   Currently unused, should be zero.
```

1.219 bn_bstatus

```
   VARIABLE   bn_bstatus

       TYPE   int

DESCRIPTION   If bit 0 is set, user has error correcting modem.
              If bit 1 is set, this is a mail session.
```

1.220 bn_zstatus

```
   VARIABLE   bn_zstatus

       TYPE   int

DESCRIPTION   Enumed activity:

               0 = logged off
               1 = active
               2 = not active
               3 = writing
               4 = grab
               5 = down
               6 = up
               7 = chat
               8 = door
               9 = groupchat
              10 = net activity (telnet, ftp, etc...)
```

1.221 bn_speed

```
   VARIABLE   bn_speed

       TYPE   int

DESCRIPTION   Line connect speed, 0 if local.
```

1.222 bn_time

```
   VARIABLE   bn_time

       TYPE   int

DESCRIPTION   Login time, hours*256+minutes.
```

1.223 bn_nick

```
   VARIABLE   bn_nick

       TYPE   char

DESCRIPTION   Returns the nick used in the Who command. This can be
              set by the user using U SET "NICK".
```

1.224 bn_realname

```
   VARIABLE   bn_realname

       TYPE   char

DESCRIPTION   Returns the realname of current user.
```

1.225 bu_name

```
   VARIABLE   bu_name

       TYPE   char (constant)

DESCRIPTION   Returns the users name as written to BBBS in uppercase.

EXAMPLE CODE  printf("%s\n",bu_name);
```

1.226 bu_address

VARIABLE bu_address

TYPE char

DESCRIPTION Returns the users address as written to BBBS.

EXAMPLE CODE `printf("%s\n",bu_address);`

1.227 bu_city

VARIABLE bu_city

TYPE char

DESCRIPTION Returns the users city as written to BBBS.

EXAMPLE CODE `printf("%s\n",bu_city);`

1.228 bu_phone

VARIABLE bu_phone

TYPE char

DESCRIPTION Returns the users phonenumber as written to BBBS.

EXAMPLE CODE `printf("%s\n",bu_phone);`

1.229 bu_birth

VARIABLE bu_birth

TYPE char

DESCRIPTION Returns the users birthday as written to BBBS.

EXAMPLE CODE `printf("%s\n",bu_birth);`

1.230 bu_ok2login

VARIABLE bu_ok2login

TYPE int

DESCRIPTION Contains users login status. There are three different

```
              types:

              0 = yes       The user can log in.
              1 = getlost   The user is shown the getlost file and then
                            logged off.
              2 = killed    The user name is removed, but a new user with
                            that name can log in as a new user.

EXAMPLE CODE  printf("Login status: %u\n",bu_ok2login);
```

1.231 bu_termcap

```
   VARIABLE   bu_termcap

       TYPE   int

DESCRIPTION   This variable contains terminal related information.

              It is divided into two bitwise parts:

              Bit 0-3 - terminal types:
              0=TTY
              1=ANSI
              2=AVATAR
              3=VT320

              Bit 4-7 - editors:
              0=Line
              1=FSE
              2=MG
```

1.232 bu_pagelength

```
   VARIABLE   bu_pagelength

       TYPE   int

DESCRIPTION   Sets number of lines per page for the current user.

EXAMPLE CODE  int main() {
                int p;
                p=bu_pagelength;
                bu_pagelength=0;
                /* do your stuff that requires no --more-- prompt */
                bu_pagelength=p;
              }
```

1.233 bu_charset

```
   VARIABLE   bu_charset

       TYPE   int

DESCRIPTION   Returns a number representing the character set the
              user is using. The number for the different character sets are:

               0 IBM
               1 SF7
               2 ISO
               3 IBN
               4 US7
               5 GE7
               6 NO7
               7 FR7
               8 IT7
               9 SP7
              10 MAC
```

1.234 bu_language

```
   VARIABLE   bu_language

       TYPE   int

DESCRIPTION   Returns the users language. Values are 0 through 9.
              (0=English,1=Suomi,2=Svenska,3=Norsk)
```

1.235 bu_readmode

```
   VARIABLE   bu_readmode

       TYPE   int

DESCRIPTION   This contains information about read modes for the
              user and preferred format for grab collection. It is divieded
              into two bitwise parts:

              Bit 0-3 - read modes:
              0=Marked
              1=Reference
              2=Forward

              Bit 4-7 - offline format:
              0=Text
              1=Hippo
              2=Hippo2
              3=OMEN
              4=QWK
              5=Blue Wave
```

1.236 bu_packtype

```
   VARIABLE   bu_packtype

       TYPE   int

DESCRIPTION   Contains a number corresponding to a packer type.

              The different types are:

              0=text
              1=arc
              2=zip
              3=lzh
              4=arj
              5=zoo
              6=hpk
              7=rar

              Additional packers will just be given the next number in range.
```

1.237 bu_protocol

```
   VARIABLE   bu_protocol

       TYPE   int

DESCRIPTION   Returns a number corresponding to a protocol.

              The different types are:

              0=Zmodem
              1=Ymodem
              2=Xmodem
              3=Slow-HYDRA
              4=Xmodem CRC
              5=Ymodem Batch
              6=Slow--Zmodem
              7=Hydra
              8=ZedZap

              Additional protocols will just be given the next number in range.
```

1.238 bu_nodemsgfilt

```
   VARIABLE   bu_nodemsgfilt

       TYPE   int

DESCRIPTION   Returns the node message filter level.

              Different levels are:
                0 nothing
               10 login / logout
               20 entered message
               30 joined/exited group chat
               49 messages in group chat, node messages
               50 everything
```

1.239 bu_timelimit

```
   VARIABLE   bu_timelimit

       TYPE   int

DESCRIPTION   Returns the daily timelimit of the user. 0 means the
              user has unlimited time.
```

1.240 bu_timeleft

```
                   VARIABLE   bu_timeleft

       TYPE   int

DESCRIPTION   Returns the amount of minutes that the caller had
              available when the call started. At logout, this value
              will contain the value of
                bf_timleft
                .
```

1.241 bu_timeson

```
   VARIABLE   bu_timeson

       TYPE   int

DESCRIPTION   Returns the number of logins for all time for the
              current user.
```

1.242 bu_msgleft

```
   VARIABLE   bu_msgleft

       TYPE   int

DESCRIPTION   Returns the number of messages written for the current
              user for all time.
```

1.243 bu_uploaded

```
   VARIABLE   bu_uploaded

       TYPE   int

DESCRIPTION   Returns the number of files uploaded by the current user.
```

1.244 bu_downloaded

```
   VARIABLE   bu_downloaded

       TYPE   int

DESCRIPTION   Returns the number of files dowloaded by the current user.
```

1.245 bu_pmsgleft

```
   VARIABLE   bu_pmsgleft

       TYPE   int

DESCRIPTION   Returns the number of messages left since the last reset
              for the current user.
```

1.246 bu_puploaded

```
   VARIABLE   bu_puploaded

       TYPE   int

DESCRIPTION   Returns the number of files upload since the last reset
              for the current user.
```

1.247 bu_pdownloaded

```
   VARIABLE   bu_pdownloaded

       TYPE   int

DESCRIPTION   Returns the number of files downloaded since the last
              reset for the current user.
```

1.248 bu_ptimeson

```
   VARIABLE   bu_ptimeson

       TYPE   int

DESCRIPTION   Returns the number of calls since the last reset for the
              current user.
```

1.249 bu_pmsgdumped

```
   VARIABLE   bu_pmsgdumped

       TYPE   int

DESCRIPTION   Returns the number of messages dumped since the last reset
              for the current user.
```

1.250 bu_pmsgread

```
   VARIABLE   bu_pmsgread

       TYPE   int

DESCRIPTION   Returns the number of messages read since the last reset for
              the current user.
```

1.251 bu_pkbup

```
   VARIABLE   bu_pkbup

       TYPE   int

DESCRIPTION   Returns the number of kilobytes uploaded since the last
              reset for the current user.
```

1.252 bu_pkbdown

```
   VARIABLE   bu_pkbdown

       TYPE   int

DESCRIPTION   Returns the number of kilobytes downloaded since the last
              reset for the current user.
```

1.253 bu_fchecked

```
   VARIABLE   bu_fchecked

       TYPE   int

DESCRIPTION   Returns a datecode for the date the current user last
              checked for new files.

              Day is lower 5 bits, month is next 4 bits and year (since 1980)
              is the upper 7 bits.
```

1.254 bu_timebank

```
   VARIABLE   bu_timebank

       TYPE   int

DESCRIPTION   Returns the number of minutes the current user has
              stored in his/hers timebank.
```

1.255 bu_resume

```
   VARIABLE   bu_resume

       TYPE   int

DESCRIPTION   Returns the message number in the User Info conference
              that contains the current users userresume.
```

1.256 bu_limits

```
   VARIABLE   bu_limits

       TYPE   int

DESCRIPTION   Returns a number containing different user limits,
              corresponds to "u limit" command.
```

1.257 bu_access

```
   VARIABLE   bu_access

       TYPE   int

DESCRIPTION   Returns a number containing differnt user acces bits.
```

```
               The bits are:
                 0 DOS
                 1 Conferences
                 2 Files
                 3 Private messages
                 4 Passwords
                30 Download
                31 Upload
               255 Sysop
```

1.258 bu_utoggles

```
   VARIABLE   bu_utoggles

       TYPE   int

DESCRIPTION   Returns a number containing several different user toggles.

              The bits are:
               0 Insert mode
               1 Indent mode
               2 XY Display in editor
               3 Don't flash your name
               4 Conference status at login
               5 Expert mode
               6 Not used
               7 Colors
               8 Review own messages
               9 Real VT100 keyboard
              10 Quote messages
              11 Silent mode
              12 Return to read menu with enter-key
```

1.259 bu_firsttime

```
   VARIABLE   bu_firsttime

       TYPE   int

DESCRIPTION   Returns a datecode for the first time the current user
              logged on to the board.

              Bits  Meaning
              ====  ====================
              0-4   Seconds/2
              5-10  Minutes
              11-15 Hours
              16-20 Day
              21-24 Month
              25-31 Year-1980
```

1.260 bu_lasttime

```
   VARIABLE   bu_lasttime

       TYPE   int

DESCRIPTION   Returns a datecode for the last time the current user
              logged on to the board.

              Bits  Meaning
              ====  ====================
              0-4   Seconds/2
              5-10  Minutes
              11-15 Hours
              16-20 Day
              21-24 Month
              25-31 Year-1980
```

1.261 bu_msgread

```
   VARIABLE   bu_msgread

       TYPE   int

DESCRIPTION   Returns the number of total messages read for the
              current user.
```

1.262 bu_msgdumped

```
   VARIABLE   bu_msgdumped

       TYPE   int

DESCRIPTION   Returns the number of total messages dumped for the
              current user.
```

1.263 bu_kbup

```
   VARIABLE   bu_kbup

       TYPE   int

DESCRIPTION   Returns the total number of kilobytes uploaded for the
              current user.
```

1.264 bu_kbdown

```
   VARIABLE   bu_kbdown

       TYPE   int

DESCRIPTION   Returns the total number of kilobytes downloaded for the
              current user.
```

1.265 bu_todaydown

```
   VARIABLE   bu_todaydown

       TYPE   int

DESCRIPTION   Returns the amount of bytes the current user has
              downloaded today.
```

1.266 bu_userbits

```
   VARIABLE   bu_userbits

       TYPE   int

DESCRIPTION   Returns a number contianing userbits. This is not used
              by BBBS so feel free to use this variable as you want, but
              remember that others might use it as well.
```

1.267 bv_filna

```
   VARIABLE   bv_filna

       TYPE   char

DESCRIPTION   Contains the name of the last processed file.

EXAMPLE CODE  int main() {
                printf("Last processed file: %s\n",bv_filna);
              }
```

1.268 bv_local_buffe

```
   VARIABLE   bv_local_buffe

       TYPE   char

DESCRIPTION   This is the keyboard buffer used by BBBS in all input
```

```
             routines.

EXAMPLE CODE int main() {
               bv_local_buffe = "G;Y;";
             }
```

1.269 bv_comhandle

```
   VARIABLE   bv_comhandle

       TYPE   int (constant)

DESCRIPTION   Has communication device handle.

EXAMPLE CODE  int main() {
                system(sprintf("foo.exe %u %s",bv_comhandle,bv_comdevice),0);
              }
```

1.270 bv_comdevice

```
   VARIABLE   bv_comdevice

       TYPE   char (constant)

DESCRIPTION   Has communication device name.

EXAMPLE CODE  int main() {
                system(sprintf("foo.exe %u %s",bv_comhandle,bv_comdevice),0);
              }
```

1.271 bv_comstring

```
   VARIABLE   bv_comstring

       TYPE   char

DESCRIPTION   Contains the command queue used by BBBS.

EXAMPLE CODE  int main() {
                if (pos("TEST",bv_comstring)) // if queue contains TEST
                  bv_comstring = "";          // then delete queue
              }
```

1.272 bv_interface

```
   VARIABLE   bv_interface

       TYPE   int

DESCRIPTION   Is used as a indicator which identifies b-mode or not.

EXAMPLE CODE  int main() {
                if (!bv_interface)
                  printf("Standard mode is active\n");
                else
                  printf("You are currently in b-mode\n");
              }
```

1.273 bv_crrdir

```
                   VARIABLE   bv_crrdir

       TYPE   char

DESCRIPTION   Returns the current virtual dir in BBBS’s internal File/4
              system.

EXAMPLE CODE  int main() {
                printf("Current dir: %s\n",bv_crrdir);
              }

   See Also
                bv_realdir
```

1.274 bv_realdir

```
                   VARIABLE   bv_realdir

       TYPE   char

DESCRIPTION   Returns the current real dir in BBBS’s internal File/4
              system.

EXAMPLE CODE  int main() {
                printf("Current dir: %s\n",bv_realdir);
              }

   See Also
                bv_crrdir
```

1.275 bv_holdreal

```
   VARIABLE   bv_holdreal
```

TYPE char

DESCRIPTION Returns the path to current node’s hold directory.

EXAMPLE CODE

```
int main() {
  int f;
  if ((f=fopen(strcat(bv_holdreal,"windows.bin"),"rb"))!=-1) {
    fclose(f);
    printf("DANGER! Will Robinson DANGER! Alien lifeform!\n");
  }
}
```

1.276 bv_hddesc

VARIABLE bv_hddesc

TYPE char

DESCRIPTION Returns the path and filename of the current node’s descript.ion file containing the descriptions for files in hold.

EXAMPLE CODE

```
int main() {
  int f;
  if ((f=fopen(bv_hddesc,"rt"))!=-1) {
    if (fgets(f)!="")
      printf("You have files in your hold-area!\n");
    fclose(f);
  }
}
```

1.277 bv_serna

VARIABLE bv_serna

TYPE char

DESCRIPTION Returns the registered BBBS Name from BBBS.KEY

EXAMPLE CODE

```
int main() {
  printf("You have reached %s\n",bv_serna)
}
```

1.278 bv_tempsys

VARIABLE bv_tempsys

TYPE int

DESCRIPTION Change the temporary SysOp level for the current user.

SysOp's access is a bitfield integer. You can use values from 0 to 255, as following:

1	Can shell to OS and execute OS commands
2	Full access to all conferences
4	Full access to all files
8	May read private messages from all conferences
16	May change passwords
32	May edit user's status (kill, status)
64	May change Netmail message attributes

To give a certain access just add the numbers. For example, if you want a user to have access to all conferences and private messages, the value is 10 (2+8).

EXAMPLE CODE

```
int main() {
  char name, passw;
  char old_sys;

  if (bu_name == "FOO USER") {
    printf("You may change the password on a user.\n");
    if ((name = input("User        : ",80,0))!="") {
      passw = input("New password: ",10,1);
      old_sys = bv_tempsys;
      bv_tempsys = 16;
      bbbs(sprintf("q;q;u;find;\"%s\";p;%s;%s;;",name,
                                                 passw,
                                                 name);
      bv_tempsys = old_sys;
    }
  }
}
```

1.279 bv_unum

VARIABLE bv_unum

TYPE int

DESCRIPTION Returns the current users user number from the user database.

EXAMPLE CODE

```
int main() {
  printf("You are user number %u\n",bv_unum);
}
```

1.280 bv_confnro

VARIABLE bv_confnro

TYPE int

```
DESCRIPTION   Returns the number of the current conference.

EXAMPLE CODE  int main() {
                printf("You are in conference number %u\n",bv_confnro);
              }
```

1.281 bv_baud

```
   VARIABLE   bv_baud

       TYPE   int

DESCRIPTION   Returns the baudrate for the current call. If it is a
              local login, this number will be 9600.
```

1.282 bv_realbaud

```
   VARIABLE   bv_realbaud

       TYPE   int

DESCRIPTION   Returns the real baudrate for the current call. If it
              is a local login, this number will be 0.
```

1.283 bv_loginhour

```
                   VARIABLE   bv_loginhour

       TYPE   int

DESCRIPTION   Returns the hour part of the time when the user
              logged in.

EXAMPLE CODE  int main() {
                printf("You logged in at %02u:%02u\n",bv_loginhour,

                bv_loginmin
                );
              }
```

1.284 bv_loginmin

```
                   VARIABLE   bv_loginmin

       TYPE   int

DESCRIPTION   Returns the minute part of the time when the user
```

```
              logged in.

EXAMPLE CODE  int main() {
                printf("You logged in at %02u:%02u\n",
                bv_loginhour
                ,
                                                      bv_loginmin);
              }
```

1.285 bv_lastmsg

```
   VARIABLE   bv_lastmsg

       TYPE   int

DESCRIPTION   Returns the number of last message read in current
              conference.

EXAMPLE CODE  int main() {
                printf("Last message read: %u\n",bv_lastmsg);
              }
```

1.286 bv_curmsg

```
   VARIABLE   bv_curmsg

       TYPE   int

DESCRIPTION   Returns current message number.

EXAMPLE CODE  int main() {
                printf("Current message number is: %u\n",bv_curmsg);
              }
```

1.287 bv_firstmsg

```
   VARIABLE   bv_firstmsg

       TYPE   int

DESCRIPTION   Returns number of first available message in current
              conference.

EXAMPLE CODE  int main() {
                printf("First message number is: %u\n",bv_firstmsg);
              }
```

1.288 bv_lastreaded

```
   VARIABLE   bv_lastreaded

       TYPE   int

DESCRIPTION   Returns number of last read message in current
              conference.

EXAMPLE CODE  int main() {
                printf("Last message read is: %u\n",bv_lastreaded);
              }
```

1.289 bv_newavail

```
   VARIABLE   bv_newavail

       TYPE   int

DESCRIPTION   Returns a number indicating how many new messages the
              current conference contains.

EXAMPLE CODE  int main() {
                printf("You have %u new messages.\n",bv_newavail);
              }
```

1.290 bv_foryou

```
   VARIABLE   bv_foryou

       TYPE   int

DESCRIPTION   Returns a number indicating how many messages in
              current conference is addressed to you.

EXAMPLE CODE  int main() {
                printf("You have %u personal messages.\n",bv_foryou);
              }
```

1.291 bv_serno

```
   VARIABLE   bv_serno

       TYPE   int

DESCRIPTION   Returns the BBBS registration number from BBBS.KEY.

EXAMPLE CODE  int main() {
                printf("Serial number: %u\n",bv_serno);
              }
```

1.292 bv_vername

```
   VARIABLE   bv_vername

       TYPE   char

DESCRIPTION   Returns the BBBS version name.

EXAMPLE CODE  int main() {
                printf("OS: %s\n",copy(bv_vername,6,3));
              }
```

1.293 bv_ver

```
   VARIABLE   bv_ver

       TYPE   char

DESCRIPTION   Returns the BBBS version number.

EXAMPLE CODE  int main() {
                printf("%s v%s\n",bv_vername,bv_ver);
              }
```

1.294 bv_tomenu

```
   VARIABLE   bv_tomenu

       TYPE   int

DESCRIPTION   Returns the menu number where user is.
                     1=read
                     2=main
                     3=util
                     4=file
                     5=chat
                     6=outb
                     7=ftp

EXAMPLE CODE  int main() {
                if (bv_tomenu==2) printf("main menu\n");
              }
```

1.295 bv_brobot

```
   VARIABLE   bv_brobot

       TYPE   int
```

```
DESCRIPTION   Returns how many times user has asked for download access
              from BRoboCop in this call.

EXAMPLE CODE  int main() {
                printf("Download access requests: %u\n",bv_brobot);
              }
```

1.296 bv_nodenumber

```
   VARIABLE   bv_nodenumber

       TYPE   int

DESCRIPTION   Returns the nodenumber for the current node.

EXAMPLE CODE  int main() {
                printf("You are connected to node %u\n",bv_nodenumber);
              }
```

1.297 bv_reduced

```
   VARIABLE   bv_reduced

       TYPE   int

DESCRIPTION   Returns the number of minutes the current users
              timelimit has been reduced by due to an event.

EXAMPLE CODE  int main() {
                printf("Timelimit is reduced by %u minutes\n",bv_reduced);
              }
```

1.298 bv_nextevent

```
   VARIABLE   bv_nextevent

       TYPE   int

DESCRIPTION   Returns the number of minutes left to the next event.

EXAMPLE CODE  int main() {
                printf("Next event is due in %u minutes\n",bv_nextevent);
              }
```

1.299 bv_temptime

```
                   VARIABLE   bv_temptime

       TYPE   int

DESCRIPTION   Returns the time limit of the user for the current
              call. At the start of the call, this is assigned the value
              of
                bu_timeleft
                .

              If you want to modify the user's timelimit for the current
              call, modify this variable.
```

1.300 bv_com

```
   VARIABLE   bv_com

       TYPE   int

DESCRIPTION   Returns the COM port number of current node.

EXAMPLE CODE  int main() {
                printf("You are running on COM port %u\n",bv_com);
              }
```

1.301 bv_carrier

```
   VARIABLE   bv_carrier

       TYPE   int

DESCRIPTION   Inidicates the state of modem carrier

              0 inactive
              1 active

              Can be used to decide if the user hangs up or not.
              SHOULD be used when running loops in scripts,
              BBBS will not abort the script even if user hangs up.

EXAMPLE CODE  int main() {
                printf("Hit any key (or hangup)!!!\n");
                while (!kbhit() && bv_carrier);
              }
```

1.302 bv_outputstopp

```
                   VARIABLE   bv_outputstopp
```

TYPE int

DESCRIPTION (Dis)enables output to the user/local screen. Can be used when you want to do something which the user shouldn't see.

BBBS will set bit 0 to 1 if user presses 'N' to --more-- and resets it when doing `input()` next time. You probably want to alter this variable after showfile() or similar functions. BBBS resets only bits 0-2 and saves 3-7.

Bit	Meaning
1	User pressed 'N' to "--more--"
16	'N' to "--more--" is disabled
32	Show log entries to screen too
64	Local screen output is disabled
128	Remote output is disabled

EXAMPLE CODE

```
int main() {
  printf("Please wait while joining some conferences..\n");
  bv_outputstopp = bv_outputstopp | 1;
  bbbs("j;bbbs.english;");
  bbbs("j;bbbs.util;");
  bv_outputstopp = bv_outputstopp & 0xF8;
}
```

1.303 bv_quicklogin

VARIABLE bv_quicklogin

TYPE int

DESCRIPTION Toggles whether quicklogin was used or not.

0 not used
1 used

1.304 bv_paged

VARIABLE bv_paged

TYPE int

DESCRIPTION Number of times user has requested chat with sysop.

1.305 bv_groups

```
   VARIABLE   bv_groups

       TYPE   char

DESCRIPTION   Returns a string with all the groups the current user
              are a member of. Groups are seperated by a comma (,).
```

1.306 bv_txt

```
   VARIABLE   bv_txt[linenumber]

       TYPE   char (constant), indexed

DESCRIPTION   Returns the text at the linenumber you supply from
              BBBSTXTx where x is nothing for english or the language code
              the current user is using.

EXAMPLE CODE  Using english language the following command:
              printf("%s\n",bv_txt[13]);
              will output:
              Sorry, no resume info in this system.
```

1.307 Other script languages

Along with the BZ language , BBBS also supports Java and perl as script languages. When a perl or Java script is executed, all input and output from/to stdin/stdout is redirected accordingly, so you can use the standard input/output commands of the language.

If you want to use perl or Java as a script language, you will have to specify the BBBSJAVA (for Java) or BBBSPERL (for perl)

environment variable to contain the path to the parser for the appropriate language.

If you are using BBBS/2 , it is also possible to use REXX as the script language.

If you use multiple script languages, remember that BBBS first tries to run BZ programs (.bz), then REXX scripts (.cmd), then Java scripts (.class) and finally perl scripts (.pl).

1.308 BBBS/D, BBBS for PC-DOS

BBBS/D is the PC-DOS specific version of BBBS.

BBBS should work on all IBM PC compatible computers running PC-DOS v3.1 (or compatible). BBBS/D can be run under DESQview or other similar multitasking environment. BBBS/D requires about 370kB of free memory and can use your EMS/XMS memory for swapping in DOS shells (where it uses only 3.5kB of your conventional memory). You may not load any memory resident (TSR) programs when you are in DOS shell. You can run BBBS on Local Area Networks. Using a real disk caching program (not SmartDrive) will speed things up dramatically. Optionally, a FOSSIL communications driver (revision level 5 or higher) can be used with BBBS/D, but is not required.

There are some of features, like the Internet-support, that do not work under the PC-DOS version, for obvious reasons.

You can use the following DESQview settings with BBBS/D:

```
        Memory Size (in K)................: 370
        Writes text directly to screen....: [Y]
        Displays graphics information.....: [N]
        Virtualize text/graphics (Y,N,T)..: [T]
        Maximum Program Memory Size (in K): 640
        Maximum EMS/XMS/VCPI/DPMI (in K)..:
        Uses its own colors...............: [Y]
        Runs in background (Y,N,blank)....: [Y]
        Uses math coprocessor.............: [N]
```

On multinode systems it is highly recommended to use IBM OS/2, not DESQview, even with BBBS/D. The DOS-version of BBBS automatically detects OS/2 and DESQview, but some advanced timesliding options are only available under OS/2 (in a VDM with BBBS/D or native BBBS/2). When running BBBS/D in a VDM you should use VX00 (look for a package called SIO*) or some other OS/2 FOSSIL driver. If you are running BBBS/D in a VDM, run remote nodes in full screen sessions, not in a WPS window.

BBBS supports all standard FOSSIL drivers (revision level 5 or higher), such as X00 or BNU. However, if you don't want to use a FOSSIL driver, BBBS offers you reliable, high-speed internal communication routines. On startup BBBS will automatically detect whether or not a FOSSIL driver is loaded and act accordingly. Using a FOSSIL driver is recommended, though. Remember to load the FOSSIL driver before DESQview.

Usually you can set the receive and transmit buffer sizes when loading the FOSSIL driver. Using 4kB buffers will give you good performance and will speed up your system. It will also give you more reliable file transfer with some internal and external protocols . Of course, bigger buffers will allocate more memory so if you are very tight on memory you can use smaller transmit and receive buffers, like 512 bytes (usually FOSSIL driver's default settings).

For more information, read your FOSSIL user's guide.

1.309 BBBS/2, BBBS for IBM OS/2

BBBS/2 is the IBM OS/2 2+ specific version of BBBS.

To run BBBS/2, you need a system that is capable of running OS/2, version 2 or higher. Thus, you will need at least 4MB of memory, but 8MB is highly recommended. If you are running tight on memory, you should consider running BBBS under a shell other than WPS. MShell and FileBar are known to be good replacement shells for running BBBS.

BBBS/2 will work on a FAT drive, but it is highly recommended to run BBBS only on HPFS drives. Using HPFS drives will speed up your system dramatically. At least your file areas should be on a HPFS drive to have real file names.

The BBBS/2 calling convention is:

```
bbbs.exe [com] [node] {device} {*} {handle}
```

Device is the comport device, handle is the filehandle number of an already opened comport. Device should be specified if you are using some special hardware, like DigiBoard. With the device-parameter BBBS will support up to 255 comports.

Users of DigiBoard should note that the DigiBoard OS/2 drivers (and also some other drivers that have too small output buffers) have trouble coping with high-speed bidirectional transfers with protocols such as HYDRA. To get BBBS working correctly with these kind of setups, the BCFG4 option

Local: OS: Buffered output
should be turned on to get better performance.

Device must also be specified if you want to specify a handle, used with external mailers and other such programs. Using the special handle BTERM you can start
BTERM
automatically. When started like this, exiting BTERM will cause a return to the OS.

BBBS/2 is able to do communications via named pipes and shared memory. To use pipes, you must specify a valid pipe name as the device, for example \PIPE\BBBSpipe or \\SERVER\PIPE\BBBSpipe. To use shared memory, the device must be \SHAREMEM\memfile, where memfile is an appropriate shared memory handle.

The asterisk ('*') should be added after the device name if you want BBBS to act as a client for named pipe/shared memory communications (by default, BBBS will act as a server).

BBBS/2 also supports ISDN CAPI; by starting BBBS with an additional ISDN device, it is possible to dial out and take incoming calls with ISDN equipment that support 16bit or 32bit CAPI (version 1.1) under OS/2.

An ISDN node is started normally, with an ISDN as the device. The com-parameter should be the adapter number, with 1 being adapter number 0 and so on.

BBBS accomplishes dialling out with ISDN by doing some simple Hayes emulation on the ISDN device via the ISDN CAPI. The EAZ of incoming and outgoing calls is done with the ATZin,out command, where in is the incoming EAZ and out the outgoing EAZ. For incoming EAZ the parameter is a bitmapped value (bits 0 to 9), where bit 0 specifies global calls and the rest of the bits specify each their respective EAZ. For example, to specify global call and EAZ 1, the value for in should be 3 (000000011b). The out parameter, on the other hand, is simply the EAZ number (not bitmapped) to use.

You should also configure the modem init string in BCFG4 appropriately. For ISDN dialling, the correct one is "ATD".

Examples:

Start node 5 and use modem in comport 7:

```
        bbbs.exe 1 5 COM7
```

Start node 2 and use async pipe so that it can be used via network from the server "SERVER01" (start a named pipe server):

```
        bbbs.exe 1 2 \PIPE\BBSNODE2
```

Start node 1 from workstation (as a named pipe client) so that it uses BBBS node 2 via async pipe and network from "SERVER01":

```
        bbbs.exe 1 1 \\SERVER01\PIPE\BBSNODE2 *
```

Start node 5 for use with ISDN connections, using adapter number 0:

```
        bbbs.exe 1 5 ISDN
```

Start BTERM, use modem in comport 2:

```
        bbbs.exe 1 1 COM2 BTERM
```

1.310 BBBS/NT, BBBS for Microsoft Windows NT

BBBS/NT is the Microsoft Windows NT specific version of BBBS.

To run BBBS/NT, you will need a system that is capable of running

Microsoft Windows NT. A 386/33 with 16MB of memory can run BBBS/NT with acceptable performance, but 32MB of memory is highly recommended. More memory means better performance.

BBBS/NT will work from a FAT drive, but using NTFS drives is recommended. At least your file areas should be placed on a NTFS drive for maximum performance.

The BBBS/NT calling convention is:

```
bbbs.exe [com] [node] {device} {*} {handle}
```

Device is the comport device, handle is the filehandle number of an already opened comport. Device should be specified if you are using some special hardware, like DigiBoard. With device-parameter, BBBS will support up to 255 comports.

Device must also be specified if you want to specify a handle, used with external mailers and other such programs. Using the special handle BTERM you can start BTERM automatically. When started like this, exiting BTERM will cause a return to the OS.

BBBS/NT is able to do communications via named pipes and shared memory. To use pipes, you must specify a valid pipe name as the device, for example \\.\PIPE\BBBSpipe. To use shared memory, the device must be MAPFILE:memfile, where memfile is an appropriate shared memory name.

The asterisk ('*') should be added after the device name if you want BBBS to act as a client for named pipe/shared memory communications (by default, BBBS will act as a server).

Examples:

Start node 5 and use modem in comport 7:

```
        bbbs.exe 1 5 COM7
```

Start node 2 and use async pipe so that it can be used via network from the server "SERVER01" (start a named pipe server):

```
        bbbs.exe 1 2 \\.\PIPE\BBSNODE2
```

Start node 1 from workstation (as a named pipe client) so that it uses BBBS node 2 via async pipe and network from "SERVER01":

```
        bbbs.exe 1 1 \\SERVER01\PIPE\BBSNODE2 *
```

Start BTERM, use modem in comport 2:

```
        bbbs.exe 1 1 COM2 BTERM
```

1.311 The UNIX versions of BBBS

There is a version of BBBS for the following UNIX clones:

BBBS/L	Linux, 80386
BBBS/SoS	Solaris 2.X, Sparc
BBBS/UW	UnixWare, 80386
BBBS/SuM	SunOS 4.X, Motorola
BBBS/SuS	SunOS 4.X, Sparc
BBBS/IrS	IRIX, SGI
BBBS/UlD	Ultrix, DEC
BBBS/HP	HP-UX, PA
BBBS/Sco	SCO, 80386
BBBS/FBI	FreeBSD, 80386

The UNIX versions of BBBS can use stdin/stdout to communicate without support of the local screen. This means that BBBS can be used as a shell and as a getty. When BBBS is executed with the name bbbs_an.exe, it will automatically find the first free node and use stdin/stdout.

If you would like to run BBBS as a shell for the users, you can create a special user account called bbbs, the shell of which is /home/bbbs/bbbs_an.exe. To use BBBS as a getty, you must first disable the real getty from that port and then start bbbs_an.exe and redirect the input and output to the correct port. If you want to use BBBS via network, you should patch your telnet daemon to start bbbs_an.exe and not login.

If you desperately want to run BBBS with local screen support, you should specify the comport device BBBS should use. It is very much advised that you use redirection, though, as it is much, much faster. When you are using local screen, you should at least disable local echo from BCFG4.

BBBS/L supports also ISDN cards, if you have compiled proper support to your kernel. If you use the ISDN device, then be sure to set the modem init strings in BCFG4 according to the ISDN driver in your kernel. Remember that your kernel must do proper Hayes emulation for the ISDN card, as BBBS will see the card as a normal modem.

The BTERM handle can be used to start the BTERM terminal emulator automatically. When started this way, exiting BTERM will cause a return to the OS.

Examples:

Start BBBS for node 1, using the device /dev/ttyS2, with local screen support:

```
        bbbs.exe 2 1 /dev/ttyS2
```

Start BBBS for node 1, using normal redirection:

```
        bbbs.exe 2 1 </dev/ttyS2 >/dev/ttyS2
```

Start
BTERM
, use the device /dev/ttyS1 to communicate:

```
        bbbs.exe 1 1 /dev/ttyS1 BTERM
```

1.312 OS environment variables

The following OS environment variables can be set to control the behavior
of BBBS:

```
Variable        Available values/what it is
===========================================================================
BBBS            FOSSIL - with
                BBBS/D
                , use always FOSSIL for communication
                BCOM   - with BBBS/D, use always internal high speed async
                         communication routines. This overrides FOSSIL.
                INT14  - with BBBS/D, use always BIOS routines for
                         communication. This is slow. Do not use it,
                         unless you really need to.
                DEBUG  - show some debugging information while running.
                         This can be added to any of the above with a
                         comma. For example BCOM,DEBUG.

BZLL            When this environment variable is set (that is, it contains
                any string) the
                BZLink-Lite
                 is automatically activated in

                BTERM terminal emulator
                .

BBBSJAVA        This environment variable determines the search path of the
                Java parser used to run Java scripts (.class).

BBBSPERL        This environment variable determines the search path of the
                perl parser used to run perl scripts (.pl).

BBBSHACK        This environment variable should point to a NetHack
                playground, if you want your users to use the 'q hack'-command
                to play NetHack.
```

BBBS also supports the standard timezone variables TZ and TZUTC. It is very recommended to set both to get correct time values in BBBS. The TZ variable should contain the timezone, the TZUTC variable the difference from UTC time. Remember that the variables should be the same for all nodes, otherwise you will see very big (errorneous) values in, for example, the who-command output.

```
For Finland, in winter time, the correct timezone is EET-2. Thus, TZUTC
should be +0200. For Norway, the timezone is CET-1 and the difference from
UTC is +0100. For summer time, one hour should be added to these values,
resulting into EET-3 (+0300) for Finland and CET-2 (+0200) for Norway.
```

1.313 Top-19 errors

```
                1Q:     BBBS reports "Directory not found." when I try to enter  ↩
                   the file
        areas.
1A:     Your "
                filedirg.000
                " file is missing root-directory, '/'.

2Q:     BBBS does not find subdirectories in the file area.
2A:     Write directories in lower case, not upper. Use '/' as directory
        separator, not '\'.

3Q:     BBBS does not initialize modem.
3A:     Your initialization string misses '|'. Also remember to use it
        in your busy and answer strings.

4Q:     BBBS reports mysterious disk errors.
4A:     You are using SmartDrive or some other non-working disk utility.
        Please  remove  them  and try again. You should also specify big
        value for files-setting in your config.sys.

5Q:     I get "Access denied." when I try to DEScribe or delete a file.
5A:     Give yourself a write access to file directories with '@w' flag.
        Remember to edit "groups" file too.

6Q:     BBBS reports "Hold not found." every time somebody logs in.
6A:     You have not specified correct hold directory in filedirn.???
        file(s). Remember to give all users a write access to the hold.

7Q:     BBBS reports swap error when shelling to external program?
7A:     You have wrong path in external.bbb file or the filename is
        missing extension.

8Q:     The commands are all messed up: who brings up bulletin menu
        and stuff like that.
8A:     You haven't updated your bbbstxt-files or have messed them up.
        Update them from the distribution package.

9Q:     BBBS does not work.
9A:     Use a soft, lint-free cloth to clean your hard disk. The
        magnetic dust interferes with high-fidelity applications. Also
        delete your copy of Microsoft Windows.

10Q:    When will the next version be available?
10A:    Dunno.

11Q:    "Download", "MOve" and "COpy" commands does not work in the file
        area.
```

11A: You have too long description for some file(s) in the current directory. Maximum length is about 750 characters.

12Q: When starting BBBS it just returns to OS. It works fine with FD.
12A: You have defined local/general/dobbs.bat in BCFG4. Clear it. You should also read help for that option.

13Q: BBBS does not find the phonenumber of system listed in nodelist.
13A: Remember to compile nodelists every day by running BNC.

14Q: Sometimes I see descriptions like "-Q81028001e000001c83acc0" for the files in my filearea.
14A: You've been running QPEG with "write descriptions" option on. That's a error/problem in QPEG. Turn the option off.

15Q: BackDoor doesn't poll to some system
15A: The system has been busy too many times or there has been too many errors while calling it (see: maindir/busypoll.dat and maindir/badpoll.dat). You should also check that node is CM or you have correct event running and your nodelist index is up to date. "Send crashmail" and "Backdoor" toggles must also be enabled.

16Q: BBBS reports "Can't open the program" when executing it.
16A: Your shell does not provide full filename for "bbbs.exe". Try to start it with command "bbbs.exe", not "bbbs".

17Q: I have noticed that some messages are disappearing faster than others. Why is that and who decides?
17A: Its the size of the message. You see, the messages are stored on disk. A message is encoded in things known as 'bits' which are written on the disk. A disk is a rotating platter. As anyone knows centrifugal force will force anything off of a rotating surface.

As time goes on, the message moves closer and closer to the edge of the disk, and finally, it flies right off. Of course, the larger messages (more bits == more weight) tend to fly off faster.

18Q: Clock in OS is OK, but in BBBS it's wrong by one hour.
18A: Your TZ environment variable is wrong. You haven't set it at all, or it has/misses DST suffix.

19Q: OS/2 reports that comport is already in use when I try to start external program.
19A: If you try to start OS/2 program you must give it comport handle (%a in external.bbb), not number or device (%o, %A). If you try to start a DOS program you must use SIO driver and it's "-"-parameter (in config.sys: "(port,base irq,-)").

1.314 About regular expression

Regular Expression is a standard way to scan for a text. RegExp has specific syntax for wildcards which differs from wildcard scan used for files. All texts in BBBS are scanned with RegExps, the scan in not case sensitive.

RegExp	What it does
foz	text "foz" as it is
^	beginning of the line
$	end of the line
.	any character
[foz]	character ‘f’, ‘o’ or ‘z’
[^foz]	any other character than ‘f’, ‘o’ or ‘z’
[f-j]	equal to command "[fghij]"
[^f-j]	equal to commnad "[^fghij]"
(foo|bar)	text "foo" or "bar"
x?	equal to command "(x|)"
x+	one or more ‘x’
x*	zero or more ‘x’
\x	character ‘x’, used finding f.ex. ‘*’

NOTE

Following RegExps are valid, but the result might not be what you think you asked for.

RegExp	What it does
.*	matches everything
.	matches everything except an empty line
fub*	matches lines with "fu"
file*.*	matches lines with "fil"

EXAMPLE

RegExp	What it does
foobar	scan for text "foobar" anywhere in the line
^foobar	scan for a line starting with "foobar"
fo.bar	scan for a line with text fo<any character>bar
*	scan for character "*" anywhere in the line
foo(bar|ugh)*buz$	scan for a line with text "foo" followed by zero or more "bar" or "ugh" and followed by "buz" at the end of the line. For example lines "foobuz" and "junkfoobarughbarbuz" matches this RegExp

SEE ALSO

Wildcards

1.315 Wildcards

DESCRIPTION

Wildcard-filescan is a standard way to scan for files with user-defined pattern. The pattern must match whole filename.

Wildcard	What it does
foz	text "foz" as it is
*	zero or more characters (any)
?	any character
[foz]	character ‘f’, ‘o’ or ‘z’
[!foz]	any other character than ‘f’, ‘o’ or ‘z’
[f-j]	equal to command "[fghij]"
[!f-j]	equal to commnad "[!fghij]"
\x	character ‘x’, used finding f.ex. ‘[’

NOTE

Following wildcards are valid, but the result might not be what you think you asked for.

Wildcard	What it does
.	matches filenames with "." in it, not all files
foo*.*	matches filenames starting with foo and having "." in it

EXAMPLE

Wildcard	What it does
*	all files
foo	filenames with "foo" in it
foo*bar[1-4x]	filenames starting with "foo" and ending with "bar" followed by a number from 1 to 4 or "x"

SEE ALSO

Regular Expression

1.316 The BBBS License

Read the terms and conditions of this license agreement carefully before using the software. If you for any reason, whatsoever, cannot accept the conditions in this agreement, you are not permitted to use BBBS.

BBBS is a proprietary product of Kim Heino and Tapani T. Salmi, hereafter "the authors", protected by applicable copyright laws and international treaty provisions.

BBBS is not, nor has ever been, public domain or free software. You must register after the 30-day evaluation period. If you decide to use this software, then you are under both legal and moral obligations to register it with the author. Registration entitles you continue using BBBS.

In the event that you are in violation of this license agreement, you agree and accept that the authors may cancel your registration and any rights to use BBBS that you may have.

BBBS is provided "as is", without warranty of any kind or fitness for a particular purpose, either expressed or implied, all of which are hereby explicitly disclaimed. The authors only guarantees that BBBS will occupy disk space. The authors liability resulting from your use or inability to use BBBS is limited to the amount that the affected party has paid for it.

The registered version of BBBS may not be duplicated other than for backup purposes. In a non registered version there is user’s timelimit limited to 31 minutes. That limitation will be removed when registering the program.

Some BBBS versions are distributed with object code files. These objects may only be used to rebuild (link) BBBS executables, all other uses are strictly forbidden.

There are two versions of BBBS, with and without RSA crypting. BBBS with RSA may not be imported to some countries, especially the USA. Please check your local import laws first. The RSA library used is fully coded in Finland.

There are two different licenses for BBBS: COMMERCIAL and NONCOMMERCIAL. You may use NONCOMMERCIAL license only if your BBS is free for all users (new and old ones) and it is public for everybody to log in. You must use COMMERCIAL license if you use BBBS in a company for inner or external mail or file transfer purposes, or you run a support BBS for a company or it’s products.

One license for BBBS is valid only in one system, which may contain one or more computers physically connected to each other all the time via LAN, WAN or similar. You may not use one license in two physically separated systems, i.e. in two different offices of one company.

You can order with the
registeration form
.

In doubt contact the authors.

Contact Information:

```
Kim Heino                                BarCode BBS
Paavolankatu 3 D 34                      v/Rune Johansen
FIN-20240  TURKU                         Ospelia 23 B
Finland                                  N-1481 Hagan
Internet: Kim.Heino@utu.fi, b@bbbs.net  Norway
BBS/FAX: +358 2 240 4036 (BCG-Box)       Internet: rune@cinet.no, rune@bbbs.net
         +358 2 240 7755                 BBS/FAX: +47-67 06 00 07 (BarCode)
         +358 2 240 1413                          +47-67 06 10 44
         +358 2 240 4410                 Fidonet: 2:210/20
FidoNet: 2:22/222
```

1.317 BBBS prices and order form

Remember to read and understand the
BBBS license
carefully before ordering
to avoid unnecessary hassle! If you have any questions, contact the authors.

The BBBS node license is divided in two; phone nodes and local nodes.
A local node can not be used with modem or ISDN (or modem emulating driver), it can only be used for local/telnet/PIPE (LAN) connections.
A phone node can be used for all purposes. The minimum is 2 phone nodes and 0 local nodes. See the table below for current prices and available node amounts.

Current prices in Finnish Marks (FIM) and Norwegian Crowns (NKR) are:

	Noncommercial		Commercial	
Phone nodes	FIM	NKR	FIM	NKR
2	300	500	900	1500
5	500	800	1500	2400
10	1000	1500	3000	4500
20	2000	3000	6000	9000
21 and over	ask	ask	ask	ask

	Noncommercial		Commercial	
Local nodes	FIM	NKR	FIM	NKR
0	0	0	0	0
5	100	180	300	500
10	200	350	600	1000
20	400	650	1200	1900
21 and over	ask	ask	ask	ask

```
!!!!!!!!!!!!!!!!!!!!!!!!!
!!                     !!
!!  Remember:          !!
!!                     !!
!!  The minimum is     !!
!!  2 phone nodes and  !!
!!  0 local nodes.     !!
!!                     !!
!!!!!!!!!!!!!!!!!!!!!!!!!
```

The change of the number of nodes or the license type will cost the difference * 1.1. Changes made within a week will be charged for difference only. I.e. if you want to upgrade from 2 phone node noncommercial license to 5 phone node commercial license, the price will be (1500-300)*1.1, 1320 FIM. Fill in new registeration form when you want to upgrade. When upgrading you can only increase your node amounts, you can not decrease them.

When registering world-wide, send the registration form and money to:

Address:
Kim Heino
Paavolankatu 3 D 34
FIN-20240 TURKU
Finland

Bank-account:
Turun Seudun Osuuspankki, Finland
571236-58198

When registering in Norway or with credit card, send the registration form and money to:

Address:
BarCode BBS
v/Rune Johansen

Bank-account: 1607 57 54098
Post-account: 0536 20 82139

```
 Ospelia 23 B
 N-1481 Hagan
 Norway

Do not use "Telegiro" for payment in Norway, since the receipt for your
payment is not sent out to the receiver. It is only noticable on monthly
statement, and could very easily be overlooked.

For registering by credit card (VISA or American Express), please fill
out the credit card information in the bottom of this form too, and FAX
to BarCode BBS at +47 22 64 41 22, or send printed version to the
address above by regular mail.

                Don't forget to sign at the bottom line!!!


                         BBBS Registration Form           Date: __/__/__


      Your name/company: ____________________________________________

         Street address: ____________________________________________

            Postal code: ____________________________________________

                Country: ____________________________________________

            Voice phone: ____________________________________________

 BBS name (max 18 char): _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

  BBS phones/open hours: ____________________________________________

      Network addresses: ____________________________________________

         Other BBS info: ____________________________________________

        License type:
        [_] Noncommercial license
        [_] Commercial license
        [_] Upgrade, current BBS name is __________________________,
            license number _______ and number of phone nodes ______,
            and number of local nodes ______.

        Number of phone nodes: __________
                                             Total price: __________
        Number of local nodes: __________

        Method of payment:
        [_] Money included to letter
        [_] Credit card
        [_] Paid to bank account number _________________, date: _____________

        Method of sending registration key to you:
        [_] by mail, in 3.5" HD disk
        [_] by modem from BCG-Box, username:
        [_] by modem from BarCode, username: _________________________________
```

What kind of hardware and software are you using? (Name and model, hard disk, amount of RAM, modem, operating system, LAN, etc.)

____

____

Other comments, ideas, etc:

____

____

I have read the BBBS license agreement and fully agree to obey it.

Signature: ________________________________

For registration by credit card only:

Due to the fact that credit card registrations are done in Norway, the prices applied are in Norwegian Crowns. 1 US$ was approx. 6,50 NKR in August, 1996.

Remember that the address and name in the registration has to be the same as the name and address of the card holder.

Prices for BBBS registration:

Noncommercial		Commercial	
Phone nodes	NKR	Phone nodes	NKR
2	500	2	1500
5	800	5	2400
10	1500	10	4500
20	3000	20	9000

Noncommercial		Commercial	
Local nodes	NKR	Local nodes	NKR
0	0	0	0
5	180	5	500
10	350	10	1000
20	650	20	1900

I wish to withdraw the sum of __________ NKR from my credit card account for the payment of registration of BBBS.

Credit card type (Mark with X): ( ) American Express ( ) VISA

Card number: ______ ______ ______ ______

Expiry date: ___/___

```
                SIGNATURE: _____________________________________________

  Name in capital letters: _____________________________________________
```

1.318 How to use the help system

```
        The help system commands:

        Contents  - Shows the contents (the main help menu)
        Index     - Shows an alpateical index
        Help      - Shows you this help
        Retrace   - Shows the last helpscreen you looked at
        Browse <  - Shows the previous helpscreen in the helpfile
        Browse >  - Shows the next helpscreen in the helpfile
        Search    - Keyword search
        Quit      - Quits back to BCFG4

        When you are in the helpsystem you can move up and down with
        arrow keys. The keys Ctrl-R and Ctrl-C will scroll one page Up
        or Down.

        Often you will see words or sentences that are marked with a
        different background color. These are links to other related
        information. Just press enter when you have selected a link to
        get more information about that subject.
```

1.319 References

```
See Also (if you want to know more about):

Byte, March 1991, page 309, Lossless Data Compression, by Steve Apiki
        This article describes Huffman coding and LZW coding.

Davis, Stephen R.: DESQview
        A Guide to Programming the DESQview Multitasking Environment.
        ISBN 1-55851-028-1

fts-0001
        A Basic FidoNet(r) Technical Standard.

fts-0005
        The distribution nodelist.

fts-0004
        EchoMail Specification

fts-0006
        YOOHOO and YOOHOO/2U2, The NetMail handshake used by Opus-CBCS
        and other intelligent FidoNet mail handling packages.
```

```
fts-0007
        An Enhanced FidoNet(r) Technical Standard

fts-0009
        Message identification and reply linkage.

fsc-0011
        Some thoughts on fsc-0001.

fsc-0015
        FOSSIL 5.0 Documentation.

fsc-0016
        FidoNet Mail Session Startup.

fsc-0025
        AVATAR Video Spec.

fsc-0028
        A Collection of Notes on Moving Files in FidoNet.

fsc-0037
        AVATAR 0+ Video Spec.

fsc-0039
        A Type-2 Packet Extension Proposal.

fsc-0045
        A Proposed Type-2 Packet Extension.

fsc-0048
        A Proposed Type-2 Packet Extension.

fsc-0050
        A Character Set Identifier For FidoNet Message Editors.

fsc-0053
        Specifications for the ^aFLAGS field.

fsc-0054
        A System-Independent Way of Transferring Special Characters,
        Character Sets and Style Information in FIDO Messages.

fsc-0056
        EMSI/IEMSI Protocol Definitions.

fsc-0057
        Conference Managers - Specifications for Requests

fsc-0062
        A Proposed Nodelist flag indicating Online Times of a Node

fsc-0068
        A Proposed Replacement For FTS-0004.

fsc-0072
```

The HYDRA file transfer protocol.

grep(1) manual page
Regular expression.

COMMON ISDN API
Standard Interface between Application Programs and ISDN Adapters.

Murphy’s Law
Why BBBS does not work?

National Semiconductor: Data Communications, Local Area Networks, UARTs
What the heck is this NS16550AFN anyway? Also describes other NS’s UARTs with great details.

PC Magazine, May 26, 1992, page 361, Lab Notes, by Douglas Boling
Putting Serial-Port Technology in Perspective, Part 2. This article describes serial port communication with normal and FIFO UARTs and IBM Type 3 UART.

rfc0821
Simple Mail Transfer Protocol.

rfc0822
Standard for the format of ARPA InterNet text messages.

rfc0850
Standard for Interchange of USENET Messages

rfc0854
Telnet protocol specification

rfc0855
Telnet option specifications

rfc0856
Telnet binary transmission

rfc0857
Telnet echo option

rfc0858
Telnet suppress go ahead option

rfc0859
Telnet status option

rfc0860
Telnet timing mark option

rfc0861
Telnet extended options - list option

rfc0959
File Transfer Protocol (FTP)

rfc0977
Network News Transfer Protocol.

rfc1282
BSD Rlogin

rfc1321
The MD5 Message-Digest Algorithm.

rfc1345
Character Mnemonics & Character Sets.

rfc1437
The Extension of MIME Content-Types to a New Medium. The matter-transport/sentient-life-form MIME type is intended to facilitate the wider interoperation of electronic mail messages that include entire sentient life forms, such as human beings.

rfc1459
Internet Relay Chat Protocol

rfc1521
MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies.

rfc1522
MIME (Multipurpose Internet Mail Extensions) Part Two: Message Header Extensions for Non-ASCII Text.

rfc1563
The text/enriched MIME Content-type.

RC96AC and RC144AC Modem Designer's Guide
All the possible info about modems based on Rockwell chip.

Tremblay, Sorenson: The Theory and Practice of Compiler Writing
How to write your own BZC replacement. ISBN 0-07-066616-4

Your Modem Reference Guide
How to set up your modem and initialization strings.

1.320 BCFG4 configuration program, table of contents

THE HITCHHIKERS' GUIDE TO BCFG4
================================

TABLE OF CONTENTS:

```
 FidoNet

 Confs
        Local

 General

 Modem

 Dial

 Hotlogins

 Macros

 Rush Hour

 Events

 OS
        Other

 Exit/Don’t save

 Exit/Save
```

1.321 Global General

```
SUBTOPICS:

 BBBS’s name

 BTERM down dir

 FAX receive dir

 Grabfile name

 Main directory

 Menus directory

 NewUser password

 SysOp’s name

 Work directory

 Upload directory

 Internet log
```

IRC Server

1.322 Global Toggles

SUBTOPICS:

BRoboCop

Check for duplicate uploads

Check similar filenames in upload

Don’t ask address from new user

Don’t ask birthday from new user

Hide userlist

Hippo

Pack messages

Show SysOp in statistics

Show empty nodes on Who command

Show private messages to CoSysOps

Upload/download check

Grab is free

Uploader owns file

1.323 Global Numbers

SUBTOPICS:

Byte Limits

Download

FAX error

File Limits

Max. desc lines

Message rate

NewUser timelimit

```
          Timebank maximum

          Timebank rate

          Total nodes

          Upload

          WhoDown size
```

1.324 Global FidoNet

```
         SUBTOPICS:
 General

          AKA Matching

          Flags

          Location

          Nodenumbers

          Phone

          Show AKAs

          Site name

          Speed
                 Sessions

          Check destination (BMT)

          HYDRA

          Inbound

          Mail errorlevel

          Mail from unlisted nodes

          Mail from unlisted points

          Mail from unprotected nodes

          NetMail

          Poll all crashmail

          Rescan time

          Tickdir

          Tranx node
          Tranx toggle

          User mail errorlevel

          ZedZap
                 FREQ

          Magic files

          Maximum 100 kB's

          Maximum files

          Maximum minutes

          Minimum baud

          Normal dirs

          Remote FREQ when answering

          Remote FREQ when calling
                 Dial

          Convert from

          Convert to

          Delay when busy

          Number of tries
```

1.325 Global Conferences

```
SUBTOPICS:

 Alias allowed

 AllFix

 Allow private

 BPC max

 BPC min

 Conf Browser

 Desc

 Export

 Fileinfo
 Import

 Invite users

 Moderator

 Msg scan

 Must for all

 Name

 NameFix

 Net area

 Netpath

 No Fido strip

 No mark reset

 No reply

 Nodenumber

 Origin

 Post conf.

 Post

 Resume

 Total

 UUCPpath
```

1.326 Local General

```
SUBTOPICS:

 Audio bell

 BackDoor

 Blackout timer

 Don’t poll flags

 FAX baud

 FD’s DOBBS.BAT

 Grab directory
 Local SysOp keys

 Local screen echo

 Logfile

 Login.Log

 Menus directory

 Min. login baud

 Nodemsg poll rate

 Save screen on shell

 Send crashmail

 Sleep disconnect

 Uptemp directory

 Lockup password

 Lockup timeout
```

1.327 Local Modem

```
SUBTOPICS:

 Busy string

 Fast FAX

 Hangup at logout

 Hangup string

 IRQ

 Init string

 NS16550 FIFO buffer

 No carrier check

 Null modem login

 RTS/CTS handshake

 Reset to connect speed

 Ring count
```

```
                 Start baud
```

1.328 Global: Conferences: Conference Browser

Move with cursor keys: up, down, home, end, pgup, pgdn. Select current conference with Enter or ESC.

Press SPACE to toggle dragging. With dragging you can reorder the conferences. All required updates are done automatically when you'll save the config. DO NOT change order when BBBS is active!

delete moves current conference to the end of the list.
insert moves last conference to the current position.
s-key sorts all the next conferences having the same prefix than the current one.

1.329 Main: Global: General

Global options are same for all nodes.

You can move in this menu by pressing up and down keys. Select the item to configure by pressing enter.

You can also use Emacs keys to move around:

```
  C-n   down
  C-p   up
  C-f   right
  C-b   left
  C-v   page down / next screen
  C-u   page up   / previous screen
  C-q   F1 / Help
  C-w   F2
  C-e   F3
  C-r   F4
  C-t   F5
  C-y   F6
```

1.330 Main: Global: Toggles

You can move in this menu by pressing up and down keys. Select the item to configure by pressing enter.

1.331 Main: Global: Numbers

You can move in this menu by pressing up and down keys. Select the item to configure by pressing enter.

1.332 Main: Global: FidoNet

```
New submenu pops up here. Move with cursor keys and press enter to select new subtopic to enter.

General:        General FidoNet settings, AKAs and such stuff.
Sessions:       Receive directories, errorlevels, etc.
Origins:        Origin lines.
FREQ:           FREQ limits.
Dial:           Dial conversion.
BOGUS:          BOGUS and BMT settings.
```

1.333 Main: Global: Confs

```
You can move in this menu by pressing up and down  keys. Select the item to configure by pressing enter.

In this window you can press F2 to browse conferences, F3 to edit your nodenumber setup and F4 to edit your origin lines.

By pressing F5 you can add multiple conferences at one batch.

By pressing F6 BCFG4 will automatically check your badecho directory for new conferences.
```

1.334 Main: Local: General

```
Local options are only for this node.

You can move in this menu by pressing up and down  keys. Select the item to configure by pressing enter.
```

1.335 Main: Local: Modem

```
You can move in this menu by pressing up and down  keys. Select the item to configure by pressing enter.
```

1.336 Main: Local: Hotlogins

```
You can move in this menu by pressing up and down  keys. Select the item to configure by pressing enter.
```

1.337 Main: Local: Macros

```
You can move in this menu by pressing up and down  keys. Select the item to configure by pressing enter.
```

1.338 Main: Local: Rush Hour

You can move in this menu by pressing up and down keys. Select the item to configure by pressing enter.

1.339 Main: Local: Events

You can move in this menu by pressing up and down keys. Select the item to configure by pressing enter.

1.340 Main: Exit, don’t save

Quit, but do not save changes. This will abort all the changes you have made during configuration.

See Also:

Exit and save

1.341 Main: Exit and save

Quit and save changes to the disks. The changes you have just made will be written to the disk.

See Also:

Exit, don’t save

1.342 Global: General: BBBS’s name

The BBS name is the name you wish to identify your system with.

1.343 Global: General: SysOp’s name

When user enters message to "SYSOP" or a comment, message will be addressed to this name. This is useful when you are for example going to have a holiday and you want make someone else a temporary SysOp.

NOTE! the sysnote-file will still be shown only to user number 0.

1.344 Global: General: NewUser password

NewUser password will be asked when a new user logs in and he/she wants to register as an user. If empty, no password will be asked (except his/her personal password).

If you are running public BBS you should leave this empty.

1.345 Global: General: NewUser account

The account name for new user. Empty is none, "*" is user’s name, others are the name of the account. Accounts can be used to charge money from users.

If you are running free BBS you should leave this empty.

1.346 Global: General: Grabfile name

This is the name of the grabfile users will get from you with command MD. Do not write ‘.’ to the name!

NOTE! With OMEN packets only two first letters are used. That’s because of OMEN standard.

1.347 Global: General: Main directory

This is the most important directory of BBBS. This directory includes messagebase, userbase and other important files. You should backup it often.

1.348 Global: General: Menus directory

Common menus for all nodes are here. You can have different menus for all nodes with local menu directory setting.

1.349 Global: General: Upload directory

Uploaded files will be copied to this directory.

1.350 Global: General: FidoNet log

BOGUS, BMT and BTICK will log their activities here.

1.351 Global: General: CD-ROM paths

You can specify here the real paths of your CD-ROM drives. When user wants to download a file from these paths BBBS will copy the file to your hard disk (hold/cdtemp.dir) first. It will speed up downloading and allows you to have CD-jukebox.

Use regexp. For example, if you want to specify paths d:, e: and f:, you can use "^[def]:".

1.352 Global: General: Work directory

Small temporary files needed by BBBS are stored here. BBBS creates and deletes them automatically. Ramdisk is recommended!

Size requirements is nodes*size_of_long_message, about nodes*64kB.

1.353 Global: General: Feelings directory

Chat configuration and feelings files are here.

1.354 Global: General: Script directory

BBBS will look for your scripts only from this directory.

1.355 Global: General: FAX receive dir

Incoming faxes will be stored here. BBBS can receive FAXes if you have group 3 class 2 FAXmodem and you have configured it’s adaptive answer correctly. FAXes will be saved as raw Group 3 FAX files. You can use the utility called g3topbm.exe to convert a 1D Huffman encoded FAX file a to PBM file (Portable BitMap). There is also a utility called g3togif.exe that converts a raw G3 file directly to GIF (Graphics Interchange Format).

Example: c:/bbs/fax

1.356 Global: General: BTERM down dir

The directory where BTERM puts your downloads.

Example: c:/bbs/bterm

1.357 Global: General: Internet log

The name of the file where all logs on Internet activities should be stored, in case you want to separate them from
FidoNet logs
.

1.358 Global: General: IRC Server

BBBS can interface its groupchat with the Internet Relay Chat. If you want this, you should specify here the IRC server to which BBBS should connect to when IRC features are in use. Syntax is server:port, for example "irc.funet.fi:6667". If you leave this field empty, the IRC features are disabled.

1.359 Global: Toggles: Show private messages to CoSysOps

If enabled, CoSysOp(s) can read all the private messages in the conferences. Main SysOp (user 0) has always this privilege.

1.360 Global: Toggles: Upload/download check

If enabled, BBBS will perform upload/download check for user's statistic every time he/she logs in.

1.361 Global: Toggles: Show empty nodes on Who command

Normally all nodes will be shown in Who's online command. If you disable this option only nodes with active user will be shown. This might be good to turn off if you have more than 10 nodes.

1.362 Global: Toggles: BRoboCop

Enables node-BRoboCop. BRoboCop is a automatic cop in your system, users can ask download access from it, play russian roulette, etc. BRoboCop also has control of uploaded message packets when user is requesting something like resign from conference.

BRoboCop's artificial stupidity is controlled via the brobo.wht-file. The format of the file is:

```
>regexp
text
text
text
```

[...]

’regexp’ is a

regular expression

. If the message the user sends to BRoboCop matches it, then a random text line under that regexp is sent back as a reply. This feature of BRoboCop is programmed to the brobo.bz script file.

If enabled, BRoboCop will also control if the users flood (repeat the same messages over and over) on a BBBS chat channel. When BRoboCop detects excessive flooding by a user it will react as if the user had sent a message containing only "FLOOD" in it and throw the user out.

1.363 Global: Toggles: Hippo

Enables node-Hippo. Hippo is a nice guy to talk with. It also enables you to send Hippo-messages to the users with ‘HIPpo‘ command in mainmenu.

When user send a node message to Hippo-node, Hippo will look file hippo.wht to what to reply.

1.364 Global: Toggles: Pack messages

By default, BBBS will compress all messages to smaller space, thus saving your disk space about 50%. The compression used is very fast and transparent, disabling it wont speed up anything.

The only reason why you would like to disable it is because you are using an external disk compressing utility, like Stacker.

You can enable and disable this at any time as it only effects to new messages.

1.365 Global: Toggles: Show SysOp in statistics

When enabled SysOp (user #0) will be included and counted to statistics.

1.366 Global: Toggles: Don’t ask address from new user

When enabled system won’t ask address or phone number from new user.

1.367 Global: Toggles: Don’t ask birthday from new user

When enabled system won’t ask birthday from new user.

1.368 Global: Toggles: Hide userlist

When enabled system won't allow users to see userlist or statistics. Users are also logged in in "hide" mode.

1.369 Global: Toggles: Check similar filenames in upload

When enabled, BBBS will scan for similar filenames according to the filename user just gave the system in upload.

1.370 Global: Toggles: Check for duplicate uploads

When enabled, BBBS will scan your fileareas for exactly matching filename according to the filename user just uploaded. If found, the uploaded file will be deleted.

If you run LAN or (brain-dead) FAT using a big trashfil file is better idea for performance reasons. Just write names of all the files in your file areas to trashfil file (ls -R > trashfil from your favourite shell).

1.371 Global: Toggles: Download

Does user get download access in first call or not.

1.372 Global: Toggles: Upload

Does user get upload access in first call or not.

1.373 Global: Toggles: kB/Day limit relative to 9600 bps

Normally (this toggle is off) users are allowed to download only x kB's per each day. If you toggle this on, the limit will be relative to connect speed.

Example:

Limit is 3000kB/day, connect speed is 19200bps, user is allowed to download 6000kB's.

1.374 Global: Toggles: Global Download command

Should be disabled for better operation.

1.375 Global: Toggles: Chat uses time

Normally BBBS does not reduce user’s time while chatting with SysOp. If this toggle is on, time will be reduced.

1.376 Global: Toggles: SysOp Notes message

It this toggles is on SysOp notes will be written to post area as a message when user #0 logs in. If off, it will only be shown to user #0 at login.

1.377 Global: Toggles: Don’t allow remote SysOp logins

You should not turn this toggle on unless you are absolutely sure you will not try to login to your system by calling it. If it is on, only local logins will be allowed for user #0.

1.378 Global: Toggles: Email-O-Magic: Send all to UUCP

Forces Email-O-Magic to send all outgoing messages as to UUCP, even if the receivers name fits. If you want to use Email-O-Magic against some other than BBBS gate you must turn this toggle on.

To define Email-O-Magic conference turn postarea and fidoarea toggles on and define gate’s address as moderator.

1.379 Global: Toggles: Use nodenumber, not nick

Uses (old) nodenumber style nodemessages, not nick.

This toggle forces a static nick for users, "1" for user in node 1, and so on.

1.380 Global: Toggles: Allow all names

If you turn this toggle on users are not restricted to two word names. You should also remove "FIRST"-word from bbbstxt lines 17 and 63.

1.381 Global: Toggles: Grab is free

When this option is enabled, grabbing will not reduce the user’s available time for the day.

1.382 Global: Toggles: Uploader owns file

When this option is enabled, users can use the 'rm' and 'des' file menu commands on files they have uploaded.

1.383 Global: Numbers: Total nodes

Total number of nodes in your system. BRoboCop and Hippo are automatically added to it if enabled.

1.384 Global: Numbers: NewUser timelimit

When a new user registers, their daily time limit on the system is set in their user profile. If you don't want to use time limits then just set the new user time limit to zero and the system will not check this for you. Otherwise new users will get the value set here. You can upgrade it later, of course.

1.385 Global: Numbers: Message rate

BRoboCop doesn't give download access before:

```
        100 * Number_of_written_messages
        -------------------------------- > Message_rate
              Number_of_calls
```

In other words:

If message rate is 0, user has to write no messages.
If message rate is 50, user has to write one message per two calls.
If message rate is 100, user has to write one message per one calls.

1.386 Global: Numbers: Timebank maximum

Maximum amount of time user can save to timebank.

1.387 Global: Numbers: Timebank rate

For one minute in timebank BBBS requires timebank_rate minutes from user.

For example: User stores 10 minutes, rate is 5. He/she gets 2 minutes to the bank.

1.388 Global: Numbers: HYDRA tx

Size of HYDRA transmit window. Do not change this unless you know what you are doing.

0=stream, 1=2kB, 2=4kB, ..., 8=16kB

1.389 Global: Numbers: HYDRA rx

Size of HYDRA receive window. Do not change this unless you know what you are doing.

0=stream, 1=2kB, 2=4kB, ..., 8=16kB

1.390 Global: Numbers: Flood count

Determines how many identical chat messages can be sent to a channel before the sender is kicked out by BRoboCop.

Entering a value of 0 here disables flood checking completely.

1.391 Global: Limits: Byte Limits

User can download limit bytes for each byte he/she uploads. These numbers are used with BRoboCop.

1.392 Global: Limits: File Limits

User can download limit files for each file he/she uploads. These numbers are used with BRoboCop.

1.393 Global: Limits: kB/Day

User can download limit kB's for each day. 0 is unlimited.

1.394 Global: Limits: Cost

If you have enabled accounts each call will be changed by these numbers. Cost/Call is charged for each call and Cost/Min is changed for each minute user is online. Cost/Hour is charged for each starting hour of online time.

Example:

```
Cost/Call   = 56
Cost/Min    = 2
Cost/Hour   = 100
Online time = 13 minutes
Total cost  = 56 + 2 * 13 + 100 = 182
```

1.395 Global: Numbers: Max. desc lines

This is the maximum amount of lines user can use to describe a file in the file area (does not effect to file_id.diz or TICK import).

There is also a limitation of 4096 bytes (768 for PC-DOS version) for descriptions, which cannot be changed.

1.396 Global: Numbers: WhoDown size

By default, BBBS will keep track of all uploads and downloads. Here you can specify how many entries will be saved. Users can view and scan this log with `f wd` command.

Entering zero here will disable this feature.

1.397 Global: Numbers: FAX error

If there are new received faxes after a completed FAX session, BBBS exits with this errorlevel. Remember that you can also use the gotfax-script.

1.398 Global: FidoNet: General: Site name

This is your system’s name that will be exchanged in EMSI with mailers that poll you, and that you poll. Usually your BBS name and the same as in Global: General: BBBS’s name.

1.399 Global: FidoNet: General: Location

Location of your system. Example: Turku, Finland. Will be exchanged in EMSI.

1.400 Global: FidoNet: General: Phone

Telephone number of your system. Will be exchanged in EMSI.

1.401 Global: FidoNet: General: Speed

```
Your maximum modem speed in bps. Will be exchanged in EMSI.
```

1.402 Global: FidoNet: General: Flags

```
The nodelist flags that are valid for your site. Will be exchanged in EMSI. Example: CM,V32B,V42B,XX

   Action flags
   ============
   CM        Node accepts mail 24 hours a day
   MO        Node is "mail only", no BBS.

   Modem feature flags
   ===================
   V22       ITU-T V22     1200 bps full duplex
   V32       ITU-T V32     9600 bps full duplex
   V32B      ITU-T V32bis 14400 bps full duplex
   V34       ITU-T V34    28800 bps full duplex
   V42       LAP-M error correction, also MNP 1-4
   V42B      LAP-M error correction, also MNP 1-5
   MNP       Microcom Networking Protocol error correction
   HST       USR Courier HST
   H14       USR Courier HST up to 14.4Kbps
   H16       USR Courier HST up to 16.8Kbps
   V32T      V.32 Terbo (includes  V.32Bis)
   VFC       Rockwell V.Fast Class
   ZYX       Zyxel 16.8/19.2 Kbps (includes V42B and V32B)

   Echomail flags
   ==============

   |-----------------------------------------------------------|
   |      |           Bark            |          WaZOO         |
   |      |-------------------------|-------------------------|
   |      |    File     |  Update     |    File     |  Update    |
   | Flag |  Requests   |  Requests   |  Requests   |  Requests  |
   |------|------------|------------|------------|------------|
   | XA   |     Yes     |     Yes     |     Yes     |     Yes    |
   | XB   |     Yes     |     Yes     |     Yes     |     No     |
   | XC   |     Yes     |     No      |     Yes     |     Yes    |
   | XP   |     Yes     |     Yes     |     No      |     No     |
   | XR   |     Yes     |     No      |     Yes     |     No     |
   | XW   |     No      |     No      |     Yes     |     No     |
   | XX   |     No      |     No      |     Yes     |     Yes    |
   |-----------------------------------------------------------|

   If you have file requests enabled, the correct file request flag
   for BackDoor is XX.

   Mailer open time flags (if not 24h/day)
   ======================
   Txy       (See 'xy' from the table below)
```

NOTE! All times are UTC-times (for example, Finland’s summer time is three hours ahead of UTC)

Char	Time	Char	Time	Char	Time	Char	Time	Char	Time
A	0000	F	0500	K	1000	P	1500	U	2000
a	0030	f	0530	k	1030	p	1530	u	2030
B	0100	G	0600	L	1100	Q	1600	V	2100
b	0130	g	0630	l	1130	q	1630	v	2130
C	0200	H	0700	M	1200	R	1700	W	2200
c	0230	h	0730	m	1230	r	1730	w	2230
D	0300	I	0800	N	1300	S	1800	X	2300
d	0330	i	0830	n	1330	s	1830	x	2330
E	0400	J	0900	O	1400	T	1900		
e	0430	j	0930	o	1430	t	1930		

For example, if a node has a T-flag "TsF", meaning that it is open 18:30-05:00 UTC-time (that is 21:30-08:00 Finland’s summer time).

ISDN flags
==========

V110L	ITU-T V.110 19k2 asynchronous
V110H	ITU-T V.110 38k4 asynchronous
V120L	ITU-T V.120 56k asynchronous
V120H	ITU-T V.120 64k asynchronous
X75	ITU-T X.75 single link procedure, 64kbit/s B-channel
ISDN	"Others". Should be used only if none of the above are correct.

1.403 Global: FidoNet: General: Show AKAs

Show only # of first akas to remote. If you specify 99 here, all of your akas will be transmitted to remote in EMSI session (this should be used). Specifying zero will only send your matching aka to remote, no other akas. Using value of 3 will present remote your matching aka and akas 0-2.

1.404 Global: FidoNet: General: Nodenumbers

Your nodenumbers, in 4D-format (zone:net/node.point). First nodenumber (#0) is your main address, others are AKAs (Also Known As).

Example: 2:22/222.0

1.405 Global: FidoNet: General: AKA Matching

Match this AKA for following regexp nodenumbers.

Example:

```
^[1-6]:                 Will use this nodenumber with zones 1-6
^14:15[12]0/            Will use this nodenumber with zone 14 and nets
                        1510 and 1520.
```

1.406 Global: FidoNet: Sessions: Inbound

Where your incoming FidoNet files are stored. BTICK will look for this directory for new files and move received files to right directory.

1.407 Global: FidoNet: Sessions: NetMail

Your FTS-0001 format NetMail directory for file attaches.

1.408 Global: FidoNet: Sessions: Tickdir

Where outgoing tick files are stored to be sent from your system. It is a good idea to use your (BOGUS's) outbound directory for these too.

1.409 Global: FidoNet: Sessions: Tranx node

Tranx your clock with this node.

1.410 Global: FidoNet: Sessions: Rescan time

This setting tells how often BBBS should check your Netmail-area for changes. Unit is minutes.

1.411 Global: FidoNet: Sessions: Mail errorlevel

If there is new received mail after a completed mailsession, BBBS exits with this errorlevel. In addition to exiting, the gotmail-script is always run when mail is received.

Entering 0 here disables this feature.

1.412 Global: FidoNet: Sessions: User mail errorlevel

If user writes new message(s) to area(s) which are declared as 'FidoNet area' in the message area configuration, BBBS exits with this errorlevel after user logs off.

Entering 0 will disable this function.

1.413 Global: FidoNet: Sessions: HYDRA

If checked, internal HYDRA protocol is enabled in EMSI mailsessions.

1.414 Global: FidoNet: Sessions: ZedZap

If checked, internal ZedZap (ZModem variant) protocol is enabled in EMSI mailsessions.

1.415 Global: FidoNet: Sessions: Tranx toggle

If enabled BBBS tries to Tranx your clock to time with system address configured in
Global: FidoNet: Sessions
.

1.416 Global: FidoNet: Sessions: Mail from unlisted nodes

If checked, accept mail also from nodes not listed on the nodelist.

1.417 Global: FidoNet: Sessions: Mail from unlisted points

If checked, accept mail also from points not listed on the nodelist.

1.418 Global: FidoNet: Sessions: Mail from unprotected nodes

Accept mail from nodes which doesn't have a session password specified.

1.419 Global: FidoNet: Sessions: Poll all crashmail

Normally BackDoor will only poll crashmail originally from your akas. If you enable this option BackDoor will poll all messages having crash bit set.

If you enable this, everybody can route NetMails to you and thus force you to poll long distance calls.

1.420 Global: FidoNet: Origins

BBBS will add origin line to echomail messages when exporting messages from BBBS with
BMSG
 or
BOGUS
. Origin line contains information about node where the message came from. You could e.g. place your BBS name here.

NOTE! Your nodenumber will be added automatically so DO NOT write it on the origin line. For example, if you specify "BCG-Box" as your origin line, the last line added to your messages will look something like this " * Origin: BCG-Box (2:22/222)".

1.421 Global: FidoNet: FREQ: Remote FREQ when answering

When this option is enabled, remote systems can call you and request files from your system. Usually enabled.

1.422 Global: FidoNet: FREQ: Remote FREQ when calling

When this option is enabled, remote system can request files from your system when you call them. Usually disabled.

NOTE! When you call, you also pay for the transfer. If you check this option, be sure you have some kind of limits set in FREQ configuration.

1.423 Global: FidoNet: FREQ: Maximum files

The maximum number of files your system allows for one file request session.

Normal: Maximum number of files for normal nodes
Secure: Maximum number of files for password secured sessions
Points: Maximum number of files for points

1.424 Global: FidoNet: FREQ: Maximum 100 kB's

The maximum number of 100 kilobytes your system allows for one file request session.

Normal: Maximum number of 100kB's for normal nodes
Secure: Maximum number of 100kB's for password secured sessions
Points: Maximum number of 100kB's for points

1.425 Global: FidoNet: FREQ: Maximum minutes

The maximum number of minutes your system allows for one file request session.

Normal: Maximum number of minutes for normal nodes
Secure: Maximum number of minutes for password secured sessions
Points: Maximum number of minutes for points

1.426 Global: FidoNet: FREQ: Minimum baud

The minimum baud rate to allow requests. If the current connection is at lower speed, requests will not be honored.

Normal: Minimum baud rate to allow requests for normal nodes
Secure: Minimum baud rate to allow requests for secured sessions
Points: Minimum baud rate to allow requests for points

1.427 Global: FidoNet: Dial: Delay when busy

How long BBBS will wait before dialing again the number that was busy on the previous call. Unit is minutes.

1.428 Global: FidoNet: Dial: Number of tries when busy

How many times BBBS will try to dial and get connect to a specific number before giving up, when the node is busy.

NOTE! It's not good for your health to specify a HIGH number here. Good value is 30. It tries 30 times before giving up.

1.429 Global: FidoNet: Dial: Number of tries when bad

How many times BBBS will try to dial and get connect to a specific number before giving up, when there was some kind of error (carrier lost or such problems).

NOTE! It's not good for your health to specify a LOW number here. Good value is 3. It tries 3 times before giving up.

1.430 Global: FidoNet: Dial: Convert from

Convert phonenumbers from. See example below:

```
    Convert from                  Convert to
 0:                            -> 990-
    : Adds 990- string to before every phonenumber.
    : 358-2-2404036 will be translated to 990-358-2-2404036
 1: 990-358-                   -> 0
    : Converts 990-358- string to 0
    : 990-358-2-2404036 will be translated to 02-2404036
 2: 02-                        ->
    : Takes 02- string away
    : 02-2404036 will be translated to 2404036
```

1.431 Global: FidoNet: Dial: Convert to

Convert phonenumbers to. See example below:

```
    Convert from                  Convert to
 0:                            -> 990-
    : Adds 990- string to before every phonenumber.
    : 358-2-2404036 will be translated to 990-358-2-2404036
 1: 990-358-                   -> 0
    : Converts 990-358- string to 0
    : 990-358-2-2404036 will be translated to 02-2404036
 2: 02-                        ->
    : Takes 02- string away
    : 02-2404036 will be translated to 2404036
```

1.432 Global: FidoNet: FREQ: Magic files

Path to the "magic" file list. In it you can enter magic names that can be used when remote system is requesting files. Most of the systems have magic file called "files" (your filelist) and "about" (generic info about your system). If the file name begins with a '@'-character, then a script file name should be written after it. This script will be run, when a file with it's magic name is FREQ'd.

```
--- magic.lst begin ---
files   c:/bbs/files/bcgbox.fil
about   c:/bbs/bcgbox.txt
bbbs    d:/pub/bbbs/bbbs_d.lzh d:/pub/bbbs/bbbs_gen.lzh
secret  @script
--- magic.lst end ---
```

1.433 Global: FidoNet: FREQ: Normal dirs

Path to normal directory list in which you can enter directories that remote systems can request files from. Your file areas are scanned automatically first.

```
--- normdir.lst begin ---
c:/bbs/files/
d:/pub/
d:/pub/bbbs/
d:/pub/txt/faq/
e:/tmp/
--- normdir.lst end ---
```

1.434 Global: Conferences: Total

Total number of conferences in your system.

1.435 Global: Conferences: Post

Areanumber for your post conference. Messages entered with COMment command are stored here. This is not your NetMail area.

1.436 Global: Conferences: Resume

Areanumber for your resume conference (the conference where users can write information about themselves). The command in BBBS is ‘U RES’.

65535 = disabled

1.437 Global: Conferences: Fileinfo

Areanumber for your fileinfo conference (the conference where information of the uploaded files will be written).

65535 = disabled

1.438 Global: Conferences: Name

Name of current conference. Some more-or-less standard names are "News" for news from SysOp to users, "Post Office" (or Post) for private messages, "NetMail" for private FidoNet messages, "Users" (or Resume) for users resumes, "Fileinfo" for fileinfo messages, etc.

By pressing F5 you can add multiple conferences at one batch from fidonet.na type text file.

By pressing F6 BCFG4 will automatically check your badecho directory for new conferences.

1.439 Global: Conferences: Description

Conference description, users will see this description when joining to conferences with the JOIN command.

For example:

BBBS.Chat	BBBS support echo, Finnish
BBBS.English	BBBS support echo, English

1.440 Global: Conferences: *.MSG path

Path to FTS-0001 format (MSG) messages.

When exporting messages, BMSG places messages in this directory. After export you should run your message tosser (e.g. GEcho) in SCAN mode. Tosser scans your message areas for new messages and creates outbound mail packets. When your tosser has performed the operation you can remove all files from this directory.

1.441 Global: Conferences: NNTP

These should be empty (zero) if you are not polling messages from NNTP hosts.

Name is the real name of this newsgroup in NNTP server. Host number is NNTP server number, given in BNNTP command line.

Newsgroup name "-" defaults to the name of the conference.

For email conference set newsgroup name to "-" and toggle postarea flag on.

1.442 Global: Conferences: Msg scan

When user joins to a new conference there are usually lots of unread messages. This option is used to limit the first time new message scan. E.g. you have 600 messages in the general chat area and a new user joins that area. Now you could have been specified that only 50 newest messages are scanned automatically.

1.443 Global: Conferences: Nodenumber

Which nodenumber is used, if this conference is linked to some sort of a FidoNet. Nodenumbers can be specified in Global:FidoNet part of the configuration.

With local message areas you don’t need to set up this option correctly!

You can press F3 to see/edit your nodenumbers.

1.444 Global: Conferences: Origin

Origin line used in this conference.

You can press F4 to see/edit your origin lines.

1.445 Global: Conferences: Moderator

If this conference is premoderated, you should specify moderator’s FidoNet address here. If it’s not moderated, use 0:0/0.0 in this field.

BBBS uses PISKI moderating. The message is saved as public message to conference, but when sending it to other FidoNet nodes, it is moved to NetMail as a private message to moderator’s node. It’s moderators job to move it again to conference or delete the message.

Usually you should just use "0:0/0.0" here. Use "2:22/222.0" in BBBS.SYSOP.

1.446 Global: Conferences: BPC min

Minimum amount of messages for BPC.

When your conference reaches to total BPC_max messages old messages are deleted so that there will be BPC_min messages left.

Let’s take couple of examples:

# messages	BPC_min	BPC_max	Action
0	100	200	Nothing happens
50	100	200	Nothing happens
100	100	200	Nothing happens
150	100	200	Nothing happens
200	100	200	Nothing happens
250	100	200	Conf. is packed, 100 messages are left
300	100	200	Conf. is packed, 100 messages are left

1.447 Global: Conferences: BPC max

```
                Maximum amount of messages for BPC.

See Also:
                BPC min
```

1.448 Global: Conferences: Must for all

```
Don't allow users to resign.
```

1.449 Global: Conferences: Invite users

```
Make new user member of this conference automatically on his first call.
```

1.450 Global: Conferences: Post conf.

```
Post only conference. Enable for your Post, NetMail and EMail areas.
```

1.451 Global: Conferences: Allow private

```
Allow both public and private messages.
```

1.452 Global: Conferences: No reply

```
Do not allow replies in this conference.
```

1.453 Global: Conferences: No mark reset

```
If enabled, user won't be able to mark messages as read on this area, he/she has ↩
   to read them all.
```

1.454 Global: Conferences: Fido area

```
Enable this to force BBBS handle this area as FidoNet area.
```

1.455 Global: Conferences: AGNET area

Enable this to cause this conference to function as an AGNET area.

1.456 Global: Conferences: No Fido strip

With BOGUS this will record SEEN-BY and PATH lines as kludges of the message. You can use normal "u set" command to view kludges.

With BMSG this will record all the kludges and SEEN-BY’s as visible lines of the message.

1.457 Global: Conferences: AllFix

Enables BBBS to answer incoming AllFix-requests in this conference. Currently local requests are ignored.

Do not enable this on your NetMail area!

AllFix request is a simple way to find a specific file(s) from remote systems by using wildcards or keyscan. The structure of a message to AllFix is very simple. The message must be addressed to AllFix. The subject line must contain a list of file specifications or keywords. Keywords must be enclosed within double quotes and/or started with dash character. You can put more than one word within the quotes.

For example, in the following message, Super Luser is looking for the latest version of BBBS and a help file.

```
-------------------------------------------------------
AllFix-area message #42 from SUPER LUSER to ALLFIX.
   Entered on 18th November, 1993 at 12:41, 1 lines.

Subject: bbbs* "help file"
===========================

Just looking for...
-------------------------------------------------------
```

The message text (body) is ignored. BBBS recognizes messages addressed to ALLFIX, BROBOCOP, FILEMGR and ARCHIE.

Please note that when user is looking for file snip* and keyword snip, the same file will be reported twice.

To disable AllFix scan for certain file directory use "@r:all" flag in the filedir* file. AllFix does not belong to the "all" group!

If you enable both AllFix and NameFix in the same area and BBBS receives a message addressed to BRoboCop, both files and users are checked.

See Also:

NameFix

1.458 Global: Conferences: NameFix

Enables BBBS to answer incoming NameFix-requests in this conference. Currently local requests are ignored.

Do not enable this on your NetMail area!

NameFix request is a simple way to find a specific user from remote systems. The structure of a message to NameFix is very simple. The message must be addressed to NameFix. The subject line must contain name of the user you are looking for.

For example, in the following message, Super Luser is looking for user called Test User.

```
-----------------------------------------------------
NameFix-area message #42 from SUPER LUSER to NAMEFIX.
   Entered on 18th November, 1993 at 12:41, 1 lines.

Subject: Test User
==================

Where is she?
-----------------------------------------------------
```

The message text (body) is ignored. BBBS recognizes messages addressed to NAMEFIX, BROBOCOP, WHOIS and FINGER.

If you enable both AllFix and NameFix in the same area and BBBS receives a message addressed to BRoboCop, both files and users are checked.

See Also:

AllFix

1.459 Global: Conferences: Alias allowed

Allow users to write message with alias names. The command in BBBS to set an alias is ‘U SET ALIAS’.

1.460 Global: Conferences: Import

Charset used when importing msg-files to BBBS. Usually this might be ISO (ISO Latin-1), IBM or your national charset (SF7 for Finland, NO7 for Norway).

You can use hotkeys 1234567890+ to select, or toggle with space.

1.461 Global: Conferences: Export

```
Charset used when exporting msg-files from BBBS. Usually this might be ISO (ISO ↩
    Latin-1), IBM or your national charset (SF7 for Finland, NO7 for Norway).

You can use hotkeys 1234567890+ to select, or toggle with space.
```

1.462 Global: Conferences: Fido name

```
Network name of the conference. "-" if same as the real name.
```

1.463 Global: Conferences: Fido export

```
Nodenumbers where this area will be exported.
```

1.464 Global: Conferences: Fido group

```
The one character group for this conference.
```

1.465 Global: Conferences: Fido flags

```
Flags for this area:

        S       Secure
        V       Visible
        D       No dupe check
        >       Export only
        <       Import only
```

1.466 Global: Conferences: Multiadd: Listfile

```
Full filename to fidonet.na type text file with conference names and descriptions. ↩
    Press ESC to start updating.

Adding many conferences can take several minutes. Patience is a virtue.
```

1.467 Global: Conferences: Multiadd: Prefix

```
Prefix string to be added to conference names.
```

1.468 Local: General: Logfile

All kinds of information about user's activities are written to this file. For example, connection speed, login time and date, downloaded files, written messages, time used and so on. Here is example of one visit:

```
02:42 -------------------------
02:42 CONNECT 14400
02:42 1st January, 1994
02:42 Login: SUPER LUSER
02:42 Invited: Flame
02:48 Marketplace message 17 entered
02:49 Joined: LV/Markkinat
02:50 LV/Markkinat message 984 entered
02:57 New files checked
02:57 Logout: SUPER LUSER
02:57  * Messages dumped:  0
02:57  * Messages read:    31
02:57  * Messages entered: 2
02:57  * Time used today:  15
```

1.469 Local: General: Login.Log

You can leave this entry empty.

You may use a separate logfile to log user logins and logouts. User address, Phone number, Times On and Last On -information is also written to this file.

1.470 Local: General: Spy file

You should leave this entry empty.

If you want to keep very detailed logfile of your users actions you can specify a filename here. All the commands user enters are logged here. Please note that this will slow the system down a bit. See also "q spy" command.

1.471 Local: General: FD's DOBBS.BAT

If you want to use BBBS's internal answer mode this field MUST be left empty.

Here you can specify path to FrontDoor style DOBBS.BAT file. When user logs in, BBBS looks this file for user's baud rate and some other information.

Please note, that BBBS can read only FrontDoor v2.02's DOBBS.BAT. Other versions write different file, which will not be processed correctly.

If filename is specified, it should contain one line in following format:

```
x s x n i
```

```
x = anything, may not contain spaces
s = connect speed
n = time for next event, or "x", if not specified
i = other connection information from modem (may contain spaces)

Example: x 14400 x 720 /ARQ/V32/LAPM/V42BIS/Caller ID=1234567
```

1.472 Local: General: Grab directory

Temporary grabfiles are stored here.

MUST BE EMPTY DIRECTORY!

1.473 Local: General: Menus directory

Local menus for this node are here. You can have common menus for all nodes with global menu directory setting.

BBBS will first look this directory for menus and if the menu file is not found here the common menu directory will be searched. Some files, like global bulletins are only looked from common menu directory.

1.474 Local: General: Uptemp directory

Uploads are temporarily stored here. After user has described uploaded files, they are moved to the upload directory (defined in Global: General: Upload directory).

MUST BE EMPTY DIRECTORY!

1.475 Local: General: Min. login baud

If you wish, you may specify a minimum speed for callers so as to lock out users using slower modems from this particular node. If you leave the option set at zero all users will be allowed to connect.

1.476 Local: General: Nodemsg poll rate

How often to poll nodemessages. Smaller values are faster. One unit is about two seconds.

1.477 Local: General: Blackout timer

Blank screen when blackout timer (in minutes) has expired with no activity on internal answer mode. Use 0 to disable.

1.478 Local: General: Local screen echo

Normally you can see exactly what is happening when a remote user is logged in. This can use quite a bit of resources on a loaded system, and if you don't really need to see what is going on then you can use this option to tell the system not to update the local screen while remote users are using the system.

You can toggle the local screen echo back on at any time by pressing the F10 key for the rest of the current user session. You can also press Alt-E to turn it on/off temporarily.

1.479 Local: General: Save screen on shell

Restore original screen after external program.

1.480 Local: General: Audio bell

Enables local audio bell. If bell is disabled, you won't hear if someone is yelling you.

1.481 Local: General: Local SysOp keys

Allow all local users to use local sysop keys. If turned off then the user for example cannot use the sysop local-keys to give temporary sysop privileges to the user logged in. This might be useful if you are running a local system in order to prevent the users from giving themselves sysop rights or change the time limit. You can see help for the keys by pressing shift-F10. You can use F10 to get information about the caller for the node.

1.482 Local: General: Sleep disconnect

Users who are logged into the system have to type something to keep the system alive. If it looks as they have fallen asleep then we just hangup on them. The sleep disconnect is the time in minutes we wait. Set it to zero to disable this feature.

Values are not exactly correct maximum idle times, for example value 4 means the system waits from minimum 3 to maximum 4 minutes before hangup.

Remember also that before sleep disconnect, the snooze-script is run.

1.483 Local: General: Send crashmail

Enables crashmail polling for this node. All crash mail will be sent immediately.

1.484 Local: General: BackDoor

Enables BBBS's internal FidoNet mailer, BackDoor, for this node. Normal human callers will not notice anything if enabled. Must be enabled for allowing other BBSes to transfer mail and files with you.

1.485 Local: General: FAX baud

New baud rate when receiving FAXes. Some modems want it to be 19200, ZyXEL and Nokia do not. If no change is required, use 0.

1.486 Local: General: Lockup password

The password that has to be entered before any of the local sysop keys can be used. Local sysop keys will be locked when the lockup timeout expires.

1.487 Local: General: Lockup timeout

The time that has to pass before the local sysop keys are locked with the lockup password . Units are seconds in the range 1-65500. Using value 0 here disables the locking of the sysop keys.

1.488 Local: General: Slow protocols

Normally this option is off.

When turned on, BBBS will use Slow-HYDRA and Slow-Zmodem instead of HYDRA and Zmodem in EMSI sessions of this node. Slow versions should be used on non-8bit links (as telnet nodes).

1.489 Local: General: Poll flags

BackDoor will not poll crashmail to nodes whose nodelist entry matches "don't poll " regexp and will poll only to nodes whose nodelist entry matches "ok poll" regexp. These options should be used on multinode systems to make sure BackDoor will use your modemx to poll remotes modemx and your modemy on another node to poll remotes modemy. You can also use this option to hold all outgoing mail to certain systems or systems who only have V22 modem.

1.490 Local: General: Revbits

Some fax modems use reversed bit order in send and/or receive. For most modems these options should be off (direct bit order), ZyXEL uses direct for receive and reversed for send, USR Sportster uses reversed for both.

See also AT+FBOR (Class 2) and AT+FBO (Class 2.0) commands from your modem's manual.

1.491 Local: Modem: Init string

String(s) to be sent for the modem upon initializing.

Here is couple of examples for different modems:

```
Modem:          Init-1                    Init-2
=================================================================
Asta 2400E      ATZ|                      ATM0E0|
GVC             ATZ|                      ATM0E0|
Nokia ECM FAST  ATZ|
Supra 14400     AT&F2M0E0S95=45W2|
Supra 2400      ATZ|                      ATM0E0X3|
USR V.34        AT&F1S0=0M0X4E0|
Zoom            ATZ|                      AT%E2M0E0|
ZyXEL           AT&F2M0E0|
```

Following meta-chars are available:

```
|      Enter, Cr, ^M
<      Lower DTR (hangup)
>      Raise DTR
~      0.5s delay
```

If you want to be able to receive FAXes, you must turn on your modem's adaptive answer. Here is couple of examples:

```
Nokia ECM FAST:  AT+FCLASS=2+FCLASS=0+FLID="fax_id"|
                 AT+FAA=1|
Supra 14400:     AT+FCLASS=0;+FAA=1;+FCR=1;+FDCC=1,5,0,2,0,0,0,0|
                 AT+FLID="fax_id";&K3|
USR V.34:        Init1: AT+FCLASS=2.0|
                 Init2: AT+FLI="fax_id"+FBO=1+FNR=1,1,1|
                 BTERM Init: AT+FCLASS=0|
                 Answer: ATS2=255+FAA=1;A|
ZyXEL:           AT#B0#V1#T0#L2+FLID=fax_id|
                 AT+FCLASS=0|
Terton ET:       Init1: ATZ|
                 Init2: AT+FCLASS=1;+FCLASS=0;+FLID="fax_id"|
                 Answer: ATS2=255+FAA=1;A|
Generic:         AT+FCLASS=0;+FCR=1;+FDCC=1,5|
Generic answer:  AT+FAA=1;A|
```

BBBS requires line speed (DCE) in CONNECT-response from your modem, not the speed between your computer and your modem (DTE).

1.492 Local: Modem: BTERM init string

String(s) to be sent for the modem when entering BTERM.

Following meta-chars are available:

```
|      Enter, Cr, ^M
<      Lower DTR (hangup)
>      Raise DTR
~      0.5s delay
```

For all modems, there should be the command ATS5=127 in this field to change
the del character from Ctrl-H to del, which is correct for BTERM.

1.493 Local: Modem: Hangup string

String to sent for the modem to hang up the phone.

Following meta-chars are available:

```
|      Enter, Cr, ^M
<      Lower DTR (hangup)
>      Raise DTR
~      0.5s delay
```

1.494 Local: Modem: Busy string

String to sent for the modem to make line busy.

Following meta-chars are available:

```
|      Enter, Cr, ^M
<      Lower DTR (hangup)
>      Raise DTR
~      0.5s delay
```

1.495 Local: Modem: Answer string

When a caller has dialed in the modem should send the text "RING" to the system. When the system sees this it will tell the modem to answer the phone. The modem answer string does this, usually with "ATS2=255A".

If you enable voice-options, you could use:

```
ZyXEL: AT+FCLASS=8;S2=255A|
Rockwell: AT#CLS=8;S2=255A|
```

Following meta-chars are available:

```
|      Enter, Cr, ^M
<      Lower DTR (hangup)
>      Raise DTR
~      0.5s delay
```

1.496 Local: Modem: Aftercall string

This string is sent to the modem after caller has disconnected. It can be used for debugging purposes. Usually you don’t want to use this, but if you do, remember to specify aftercall lines too. For example ZyXEL supports "ATI2" which will dump line info to screen.

Following meta-chars are available:

```
|      Enter, Cr, ^M
<      Lower DTR (hangup)
>      Raise DTR
~      0.5s delay
```

1.497 Local: Modem: Aftercall lines

BBBS will log this many lines after aftercall string. There is also 5s timeout.

1.498 Local: Modem: Start baud

Initialize modem with this baud rate. Usually the maximum baud rate your modem can handle.

This is not the maximum baud rate your modem can transfer data with another modem (like 14400 bps, DCE speed), but the maximum baud rate your modem and and your computer can talk (like 38400 or 57600 bps, DTE speed).

BBBS/D supports only baud rates up to 115200.

In BBBS/L you should also give "setserial /dev/ttyS? spd_vhi" command for correct tty before starting BBBS.

1.499 Local: Modem: Base address

Base address (port) for your comport entered as a hexadecimal number.

The default comport setup is (you can specify different, of course):

Port	Base	IRQ
COM1	03f8	4
COM2	02f8	3
COM3	03e8	4
COM4	02e8	3

This option is used only in BBBS/D.

1.500 Local: Modem: IRQ

IRQ for your comport.

The default comport setup is (you can specify different, of course):

Port	Base	IRQ
COM1	03f8	4
COM2	02f8	3
COM3	03e8	4
COM4	02e8	3

This option is used only in BBBS/D.

1.501 Local: Modem: Ring count

Answer to countth call, usually 1. If you want the caller id information to be recorded this must be set to 2.

1.502 Local: Modem: Ringing count

Remote is considered as down after this many number of RINGING messages.

1.503 Local: Modem: Reset to connect speed

If you want to run at constant speed between your modem and the computer (and have set the modem up correctly) you must turn off this option (and most probably RTS/CTS handshake on to ensure that data is not lost).

1.504 Local: Modem: Hangup at logout

Hang up the phone after user logs off.

1.505 Local: Modem: NS16550 FIFO buffer

Enables NS16550 FIFO buffering. If you don’t know what it is, don’t.

1.506 Local: Modem: RTS/CTS handshake

RTS/CTS handshake is used in certain cases to ensure that the modem does not feed BBBS faster than it can handle, or more likely, especially when using a constant speed interface, that BBBS does not feed the modem faster than it can handle. If you are using speed reseted to connect speed you probably don’t need handshaking.

1.507 Local: Modem: Null modem login

If this node is connected to other computer via null-modem cable, you can enable this option to allow users to log in by just pressing enter.

Normally this option should be disabled.

1.508 Local: Modem: No carrier check

Don’t check carrier while user is logged in. If you enable this, BBBS does not notice if user hang ups. You should only use it for null modem nodes and if your modem does not present DCD pin correctly.

1.509 Local: Modem: NO CARRIER is BUSY

If your modem does not detect BUSY signal correctly turn this option on. If turned off, NO CARRIER means error in mail sessions.

1.510 Local: Modem: Fast FAX

Enable this option if you have a FAXmodem with "fast" negation, ie. ignores "FAX" result and waits for "+FCON". Most V.34 modems should use this option, for example Nokia ECM FAST requires it.

1.511 Local: Macros: Keyboard Macros

Macro for local key Alt-\{num\}.

You can use caret to simulate control-keys and run script. For example:

```
^M              Enter
^^              Caret
^@script@       Run script called "script"
```

1.512 Local: Macros: Hotlogins

Hot string logins can be used to identify mail sessions not supported by BBBS. When hotlogin string is received from modem at login state, the hotlog-script is run with string number as a parameter. You should use long enough hotlogin string for users not to type them by accident.

1.513 Local: Rush Hour: Rush Hour Timelimits

If you have a very busy system, then you may wish to limit the amount of time users are permitted to use the system at certain times during the day. This section allows you to set the maximum time a user will be allowed to use the system, depending upon what time they logged in. Users who have less time left than is specified here will of course only be allowed the actual time they have left. For periods during the day where you don’t want an extra time limit to apply, just enter a zero.

1.514 Local: Events: Sunday

Run this event on Sunday.

1.515 Local: Events: Monday

Run this event on Monday.

1.516 Local: Events: Tuesday

Run this event on Tuesday.

1.517 Local: Events: Wednesday

Run this event on Wednesday.

1.518 Local: Events: Thursday

Run this event on Thursday.

1.519 Local: Events: Friday

Run this event on Friday.

1.520 Local: Events: Saturday

Run this event on Saturday.

1.521 Local: Events: Start hour

The start hour for this event.

See Also:
Dial-event

1.522 Local: Events: Start min

The start minute for this event.

See Also:
Dial-event

1.523 Local: Events: End hour

The end hour for this event. Event may not cross day boundary!

See Also:
Dial-event

1.524 Local: Events: End min

The end minute for this event.

See Also:
Dial-event

1.525 Local: Events: Errorlevel

Shell to OS with this errorlevel. BBBS does not start any BATch file itself, you should start BBBS from BATch and check these errorlevels.

0 Don’t shell

See Also:
Dial-event

1.526 Local: Events: Don’t allow users during event

Don’t allow users to log in when this event in running.

1.527 Local: Events: Flexible even

Event can "slide" if users logs in.

1.528 Local: Events: Event is a ’must’

Force event to be run once a day, even if the time has already elapsed.

1.529 Local: Events: Don’t allow incoming mail calls

Don’t allow remote systems to call you and send mail to you during the event.

1.530 Local: Events: Don’t allow mail pickup

Send remote’s mail to them when they call you during this event. If you enable this and allow incoming mail calls, system is in receive only mode.

1.531 Local: Events: Don’t send crashmail to CM systems

Don’t try to poll crashmail to CM systems during this event.

1.532 Local: Events: Send crashmail to all systems

Try to poll crashmail to all systems during this event.

1.533 Local: Events: Don't allow file requests

Don't allow remote systems to requests files from you during this event.

1.534 Local: Events: Dial to node

Dials to this node in this event.

Don't use overlapped events. You can define both Dial to node and Errorlevel for one event. First BBBS exits with errorlevel and then dials.

First BBBS tries to run event #1, then #2, and so on... While (dial)event is running, other events can force exit with errorlevel, but not polling (except if new event has smaller number than current event).

1.535 Local: OS: OS/2 Priority

BBBS/2 can change it's priority according the activities of user. Here you can specify priorities for different places in BBBS.

Name	Activity	Default
Modem thread	Routine handling comport activity	FOREGROUNDSERVER 0
Protocols	Upload and download	FOREGROUNDSERVER 0
Arcers, grab	(Un)Packing files and grabbing	REGULAR 0
Message search	Mark and Search menus	REGULAR 0
File search	Keyword, Wildcard, New files	REGULAR 0
List search	Jargon, Nodelist browser, Whodown	REGULAR 0
Idle	Waiting for a caller	REGULAR 0
Normal	All others	REGULAR 0

The value you should type in can be calculated as following:

value = class * 64 + delta + 32,

where delta is a number between -31 .. 31 and class is one the followings:

0 = IDLETIME
1 = REGULAR
2 = TIMECRITICAL
3 = FOREGROUNDSERVER

For example, if you want to specify priority TIMECRITICAL 16, the value is 2 * 64 + 16 + 32 = 176.

Windows NT users should use the
NT priority
-field insted.

1.536 Local: OS: NT Priority

BBBS/NT can change it's priority according the activities of user. Here you can specify priorities for different places in BBBS.

Name	Activity	Default
Modem thread	Routine handling comport activity	HIGHEST
Protocols	Upload and download	HIGHEST
Arcers, grab	(Un)Packing files and grabbing	NORMAL
Message search	Mark and Search menus	NORMAL
File search	Keyword, Wildcard, New files	NORMAL
List search	Jargon, Nodelist browser, Whodown	NORMAL
Idle	Waiting for a caller	NORMAL
Normal	All others	NORMAL

The value you should type in can be calculated as following:

value = delta + 15,

where delta is a number between -15 .. 15.

```
-15 = IDLE
 -2 = LOWEST
 -1 = BELOW_NORMAL
  0 = NORMAL
  1 = ABOVE_NORMAL
  2 = HIGHEST
 15 = TIME_CRITICAL
```

For example, if you want to specify priority NORMAL, the value is 0 + 15 = 15.

OS/2 users should use the
OS/2 priority
-field insted.

1.537 Local: OS: Show shell output

If this toggle is on, BBBS will redirect outputs of external (un)packers to user. BBBS can redirect only stdout and stderr streams, not direct screen writes.

1.538 Local: OS: Allow break in shell

If this toggle is on, BBBS will check carrier and monitors incoming characters while spawning to the external (un)packer. If carrier is drop or C-c pressed, BBBS tries to abort (un)packer.

1.539 Local: OS: Buffered output

This setting should be disabled.

If you have DigiBoard or other similar serial communication device with braindead driver with very small buffers you should turn this option on.

1.540 Local: OS: Rockwell modem

Some Rockwell chipset based modems have very poor performance on full duplex high speed transfers, like with HYDRA. If your modem is Rockwell based (like Best or Well), you should turn this option on.

This option will slow down BBBS's modem routines and add echo delay (not very notable, though). If you are sure your modem doesn't need it, turn this option off.

1.541 Local: OS: Fix RAR's 'feature'

The packing program RAR has an evil feature: it always displays its header (containing registeration info) and the absolute directories of the archives it processes. This is a major security issue. You should enable this option, if you are using RAR. It will disable packer output for the packer number 7 defined in external.bbb.

1.542 Global: General: Organization

This can be empty if you are not sending messages to NNTP/SMTP hosts.

This is one line organization name included to outgoing NNTP/SMTP message header. Usually it tells the name of your BBBS.

1.543 Global: General: Hostname

This can be empty if you are not sending messages to NNTP/SMTP hosts.

This is your computers InterNet hostname. Outgoing NNTP/SMTP messages are addressed from User.Name@hostname.

1.544 Global: General: Remote domain

This can be empty if you are not sending messages to NNTP/SMTP hosts or if you are not NNTP/SMTP gateway.

This is a format string for remote user names for messages sent via your gateway system. BBBS can recognize following incoming addresses:

```
        %d@p%p.f%f.n%n.z%z.fidonet.org
        %d!%z.%n.%f.%p@myhost.mynet.myorg
```

Metastrings:

```
        %%      '%'
        %d      Senders name in "User.Name" format
        %z      Origin zone number
        %n      Origin net number
        %f      Origin node number
        %p      Origin point number
        x       'x'
```

1.545 Global: FidoNet: BOGUS: NNTP gateway

Act as a NNTP (news) gateway, convert remotely (net) enter messages to remote domain form.

1.546 Global: FidoNet: BOGUS: SMTP gateway

Act as a SMTP (email) gateway. If this toggle is on, net users may send and receive email messages via your system.

Remote users must sent the email messages to you as a normal NetMail message, receiver must be "receiver@remote.host".

1.547 Global: FidoNet: BOGUS: NNTP save headers

If this toggle is on all the headers in incoming NNTP (news) messages are saved to message as kludges.

Should be used for debugging purposes only.

1.548 Global: FidoNet: BOGUS: SMTP save headers

If this toggle is on all the headers in incoming SMTP (email) messages are saved to message as kludges.

Should be used for debugging purposes only.

1.549 Local: Dial

When dialing to the remote system, BBBS tries to match remote’s nodelist entry to case sensitive regexp given here. If none matches, it uses #1.

Matching predial string will be sent to the modem before the number and postdial right after the number.

Normal command strings modifiers can be used (see Local: Modem)

1.550 Global: FidoNet: BOGUS: Temporary in pkt

When processing incoming mail bundles BOGUS unpacks them here.

1.551 Global: FidoNet: BOGUS: Temporary out pkt

Outgoing packets are temporarily created here.

1.552 Global: FidoNet: BOGUS: Outbound

Outgoing mail bundles are stored here.

1.553 Global: FidoNet: BOGUS: Badecho: area

Incoming messages to unknown area are stored here in FTS-0001 format.

1.554 Global: FidoNet: BOGUS: Badecho: secure

Incoming messages to secure area with access are stored here in FTS-0001 format.

1.555 Global: FidoNet: BOGUS: Max open files

Maximum number of files BOGUS may open. Bigger number is faster.

In BBBS/D this number must be limited to 7.
In BBBS/2 this number should be limited to about 20, but can be as big as 70. If you increase this number, your unpackers must can handle many files too!
In BBBS/L good number is 20.

1.556 Global: FidoNet: BOGUS: BOGUS dupes

BOGUS will remember n*1024 last messages for dupe checking. 0 disables this feature. Bigger number is slower, uses more memory and disk space, but also will prevent duplicates better.

In BBBS/D this number must be limited to 15.

1.557 Global: FidoNet: BOGUS: Max. SMTP message size

Here you can limit the maximum size of the message in kilobytes that bbbs.smtpd.exe will import. This option can be used to disable flood-bombs and binary file transfers.

0 disables this feature.

1.558 Global: FidoNet: BOGUS: Max. out pkt size

Maximum size for outgoing pkt file in kilobytes. 0 is unlimited. Normally this should be about 512.

1.559 Global: FidoNet: BOGUS: Max. out bundle size

Maximum size for outgoing bundle file in kilobytes. 0 is unlimited. Normally this should be about 2048.

1.560 Global: FidoNet: BOGUS: Save badecho: area

Toggle for saving messages to unknown area.

1.561 Global: FidoNet: BOGUS: Save badecho: secure

Toggle for saving messages violating secure setting.

1.562 Global: FidoNet: BOGUS: Save all NetMail msgs

Toggle for saving all NetMail messages passing through your system.

1.563 Global: FidoNet: BOGUS: Log headers(BMT)

If this option is enabled, BMT will log info about messages passing your system.

1.564 Global: FidoNet: BOGUS: Check destination (BMT)

If this option is enabled, BMT will check that destination node is listed in your nodelist. If not it will be returned to the sender. If sender is not listed, the message will be deleted.

Don't enable this unless you compile full nodelists.

1.565 Local: Voice

You can move in this menu by pressing up and down keys. Select the item to configure by pressing enter.

1.566 Local: Voice: Data/fax answer string

Modem command string to change from voice mode to data/fax mode.

```
ZyXEL: ATS2=255+FCLASS=0A|
Rockwell: AT#CLS=0;A|
```

Refer to your voice modem manual for more information.

1.567 Local: Voice: Init string

Modem command string to be sent before record and playback.

It must contain two "%d" substrings, first one for compression and second one for device.

```
ZyXEL: AT+VSM=%d;+VLS=%d;+FLO=2;S39.7=0;+VIT=45;+VSD=16,70|
Rockwell: AT#VBS=%d;#VLS=%d;#VSD=1|
```

Refer to your voice modem manual for more information.

1.568 Local: Voice: Beep string

Modem command string which produces small beep sound.

```
ZyXEL: AT+VTS=[800,0,42]|
Rockwell: AT#VTS=[800,0,4]|
```

Refer to your voice modem manual for more information.

1.569 Local: Voice: Playback string

```
Modem command string to start voice playbacking.

ZyXEL: AT+VTX|
Rockwell: AT#VTX|

Refer to your voice modem manual for more information.
```

1.570 Local: Voice: Record string

```
Modem command string to start voice recording.

ZyXEL: AT+VRX|
Rockwell: AT#VRX|

Refer to your voice modem manual for more information.
```

1.571 Local: Voice: Modem device

```
Device number for telephone line.

ZyXEL: 2
Rockwell: 0

Refer to your voice modem manual for more information.
```

1.572 Local: Voice: Speaker device

```
Device number for internal/external speaker.

ZyXEL: 16
Rockwell: 2

Refer to your voice modem manual for more information.
```

1.573 Local: Voice: Mic device

```
Device number for internal/external microphone.

ZyXEL: 8
Rockwell: 3

Refer to your voice modem manual for more information.
```

1.574 Local: Voice: Go voice mode

```
Modem command string to enter voice mode.

ZyXEL: AT+FCLASS=8|
Rockwell: AT#CLS=8|

Refer to your voice modem manual for more information.
```

1.575 Local: Voice: Go data mode

```
Modem command string to enter data mode.

ZyXEL: AT+FCLASS=0|
Rockwell: AT#CLS=0|

Refer to your voice modem manual for more information.
```

1.576 Local: Voice: Compression method

```
Compression method number to be used to record voice calls.

ZyXEL: 3
Rockwell: 4

Refer to your voice modem manual for more information.
```

1.577 Local: Voice: Minimum filesize to keep

```
The minimum size of a recorded voice file to keep. You can use this to avoid ←
   storing "empty" messages. Units are in kilobytes.
```

1.578 Local: Voice: Voice receive dir

```
Directory where to store recorded voice calls.
```

1.579 Local: Voice: Greetings file

```
Greetings file to be played to called.
```

1.580 Local: Voice: Remote password

```
Remote password (DTMF) for listening messages.
```

1.581 Main: CD-ROM installer

You can move in this menu by pressing up and down keys. Select the item to configure by pressing enter.

1.582 CD-ROM installer: OK to process

CD-ROM installer can be used to install a CD-ROM disk to your file areas easily.

After you have checked these questions and you are ready to process with CD-ROM install, turn this toggle on and press ESC.

1.583 CD-ROM installer: CD-ROM path

OS path for your CD-ROM drive.

1.584 CD-ROM installer: Description path in CD-ROM

Path in the CD-ROM to PCBoard-type description files (dirname.DIR files). May be left empty.

1.585 CD-ROM installer: Virtual base directory in BBBS

Virtual base directory in BBBS where to install these files.

1.586 CD-ROM installer: Filedirg name

Name of filedirg file. See sysop.gui for more information about these files.

1.587 CD-ROM installer: Description save directory

Path where to store descriptions in your hard disk. It is a good idea to store descriptions of one CD-ROM in it’s own unique directory.

1.588 CD-ROM installer: Description file extension

Extension for description files. May be left empty.

1.589 CD-ROM installer: Lower directory names

```
Turn case of directories into lower case.
```

1.590 CD-ROM installer: Lower file names

```
Turn case of files into lower case.
```

1.591 CD-ROM installer: Convert all chars

```
Don’t turn this toggle off, unless you want to keep @ chars in descs as they are ( ←
   not recommended).
```