SJ Research
File Server
User Manual

Draft Copy — January 1989 — (c) SJ Research

Contents

ECONET NETWORKS

In essence an Econet network is just a cable linking a collection of BBC computers together. This means networked computers can communicate with each other and also with resource servers. A resource server is a computer attached to the network which enables other networked computers to share devices such as disc drives and printers. Your Econet network is connected to an SJ Research file server which allows networked computers to share whatever disc drives and printers are attached to it. The advantage of this is that it is not necessary to equip every computer with its own disc drive and printer. In addition, software is available to all computers at all times.

Disc drives attached to the file server will be shared by all users of the network so the file server has to have some security features. These control which areas of discs users can access and also limit how much disc space they can use. The most fundamental security feature is the requirement that a user must *Log On* to the file server before it is possible to access the disc at all. Logging on consists of entering both a *User Id* and a *Password*. A "User Id" is the name by which you are known to the file server and is likely to be related to your real name, possibly your surname or your initials.

■LOGGING ON & PASSWORDS

After reading this section you should be able to:

- **Log on and off the file server**
- **Change your password**

Logging on

A computer which has just been switched on will have a message on the screen similar to the one below:

```
BBC Computer 32k

Econet Station nnn

BASIC

>
```

The number "nnn" will be some number between 1 and 254 and is called the *Station Number*. Every computer on the network has its own unique station number.

The simplest and most secure method of logging on is to perform a <SHIFT><BREAK>. This means holding down the <SHIFT> key while pressing and releasing the <BREAK> key. A "User Id :" prompt will appear on the screen at which you type your User Id and then press <RETURN>. A "Password:" prompt will then appear at which you type your password and then press <RETURN>.

For security reasons as you type your password asterisks, instead of letters, will appear on the screen. Should you make a mistake then the <DELETE> key can be used. If you manage to mistype your password then the "Password:" prompt will reappear so you can try typing your password again. The screen will end up looking as below:

```
User Id :FRED
Password:*******
```

Once you have logged on in this way you will be able to access the discs on the file server using normal commands such as "*CAT".

*To provide compatibility with Acorn systems it is also possible to log on to the file server with the "*I AM" command. This has the syntax: "*I AM <User Id> <Password>". Suppose your User Id is FRED and your password is CRICKET you would log on by typing:*

```
*I AM FRED CRICKET <RETURN>
```

This method of logging on is not ideal as anyone looking over your shoulder would be able to see your password. Another method of logging on which hides the password is available on some computers by pressing "<SPACE>:<RETURN>" before typing your password. A colon will appear on the line below and when you type your password no characters will appear on the screen. The screen would end up looking as below:

```
*I AM FRED :
:
```

N.B. The space between the "User Id" and the ":" is vital. Also, if you mistype your password then you may find that the <DELETE> key does not work correctly.

Logging off

When you finish using a networked computer it is vital that you *Log Off* from the file server. Simply switching off the computer you have been using is not sufficient to log off. The file server remains switched on and so remembers which user was on the computer. When the computer is switched back on again the file server will continue to allow that computer to access the user's files. To log off correctly type:

```
*LOGOFF   <RETURN>
```

You can check that this has worked by trying to access the file server disc, by using a command such as "*CAT". The file server will respond with the "Who are you?" error message indicating that access to the file server from your computer is now impossible until someone logs on again.

*For compatibility with Acorn systems it is also possible to log off from the file server using the "*BYE" command. This command should be avoided in preference for "*LOGOFF" as "*BYE" will not work under all circumstances.*

Changing your password

The best and most secure way to change your password is to type:

```
*NEWPASS   <RETURN>
```

You will first get the prompt "Old password:" at which you should type your current password and press <RETURN>. The password will appear on the screen as asterisks. You will then get the prompt "New password 1st time:" at which you should type your new password and press <RETURN>. The new password will also appear on the screen as asterisks. As you have not been able to see what you typed there is a possibility that you have made a typing mistake and so could be in danger of setting your password to something that even you do not know! To avoid this problem you are prompted to type the new password a second time. Only if you type exactly the same on both occasions will your password actually be changed.

Suppose FRED changes his password from CRICKET to SQUASH, the screen will end up looking as below:

```
Old Password :*******
New Password :******
New Password :******
```

If you do not type the same new password at both attempts then you will get the error message "Retype PW" and you will have to type "*NEWPASS" again.

Passwords must be no more than ten characters long and there are some characters you cannot use, in particular <SPACE>. It is simplest to stick to using only letters and numbers in your password and then you will have no problems.

*For compatibility with Acorn systems you can also change your password using the "*PASS" command. The syntax for this is "*PASS <Old Password> <New Password>". For example FRED could type:*

```
*PASS CRICKET SQUASH <RETURN>
```

*This method of changing your password suffers from the same problem as the "*I AM "command in that your passwords can be seen on the screen. On some computers you can hide the passwords by typing "*PASS : <RETURN>". As with "*I AM", another colon will appear on the line below and when you type the old and new passwords no characters will appear on the screen.*

LOADING & SAVING

After reading this section you should be able to:

- **Catalogue the file server disc**
- **Save files**
- **Load files**

Catalogues

Once you have logged on to the file server it is best to ask for a catalogue of the disc so you can see where you are and what programs are available. The command used to catalogue the disc is "*CAT" or "*." for short. The discs attached to the file server could be holding many thousands of programs and datafiles. It is impractical to display all of them together. For this reason the file server disc is divided in to sections called *Directories*.

A catalogue will look something like the one below:

```
*.
FRED        (002)       Owner
Work                    Option 00 (Off)
Dir. FRED               Lib. LIBRARY

PROGS       D/          SPELLING    WR/r
```

This is a catalogue of the directory you are currently in. This is not surprisingly known as your *Currently Selected Directory* (CSD). Note that entries in the catalogue are always organised to appear alphabetically.

The top three lines are the *Catalogue Header*. The name at the top left, "FRED", is the name of the directory you have catalogued. The word "Owner" indicates that you own this directory and so can save programs into it. Other information displayed in the header is not important at the moment. The contents of the directory you have catalogued appear below the header. The directory FRED has two entries: PROGS and SPELLING. The letters after an entry are called the *Access String* and give more information about directory entries. If you have not used the system before then you will probably find that your directory has no entries.

The directory you enter when you log on to the file server is called your *User Root Directory* (&).

The "D" in the access string of PROGS indicates that PROGS is itself a directory, ie a further section of the file server disc which may also contain programs and yet more directories. PROGS is called a *Sub-Directory* of the directory FRED. This arrangement of directories one inside another is called a directory hierarchy and is fundamental to the structure of the file server disc. In many ways you can imagine a directory hierarchy as being rather like a family tree. Indeed sometimes you refer to the directory which contains programs as the parent directory, and many properties of files you create are inherited from those of the parent directory.

Saving files

This is best illustrated by example. Assuming that your computer is in BASIC, enter the following short program:

```
10  PRINT"A small example program"
20  PRINT"saved on the file server."
30  END
```

This program can now be saved by typing:

```
SAVE"PROGRAM"  <RETURN>
```

The word "PROGRAM" is the *Filename*. This is the name which the program will be given when saved on to the disc. If you wish you can choose a different filename. Filenames must be ten characters or less in length and it is best if you use only letters and numbers. In particular, avoid using spaces.

Whenever you save a file on to the disc it is reassuring to then catalogue the directory to check that the file has appeared.

N.B. If you save another file with the same name as one already in your CSD, then the one on the disc will be overwritten. If the new file you are saving is simply a better version of the one on the disc then this is probably what you want. However, if this is not the case then you will lose the file on the disc. It is possible to prevent accidental loss of files from both overwriting and deletion by changing the access string of the file (see section on changing access strings, page 23).

Loading files

To load a program back from the disc you simple use the command:

```
LOAD"PROGRAM"   <RETURN>
```

It is wise to try saving a short program and then to load it back from the disc. This will give you a good feel for the speed of the network and will also check that you have been set up correctly by the System Manager.

■HELP

After reading this section you should be able to:

- **Obtain help on various topics**

Help topics

Rather than list detailed help here, there is a set of help files available on the file server so that you can have easy access to help at the times when you most need it. A command in the Library, "*DESCRIBE", is used to access the help files. The help available make use of the hierarchical directory structure to provide hierarchical help. To see a list of the topics on which help is available type:

```
*DESCRIBE  <RETURN>
```

You will obtain a list similar to the one below:

```
Errors
Utilities
```

To see what help on one of these topics type:

```
*DESCRIBE  <topic>  <RETURN>
```

and a list of items on which help is available will be displayed:

```
*DESCRIBE  Utilities  <RETURN>
```

To see the help available on a particular utility now type, for example:

```
*DESCRIBE  Utilities.PSLIST  <RETURN>
```

and a description of how the *PSLIST utility works will be displayed. In general the help information will fit on one screen, however if not, then you will need to press <SHIFT> to display more.

■ EXPLORING THE DISC

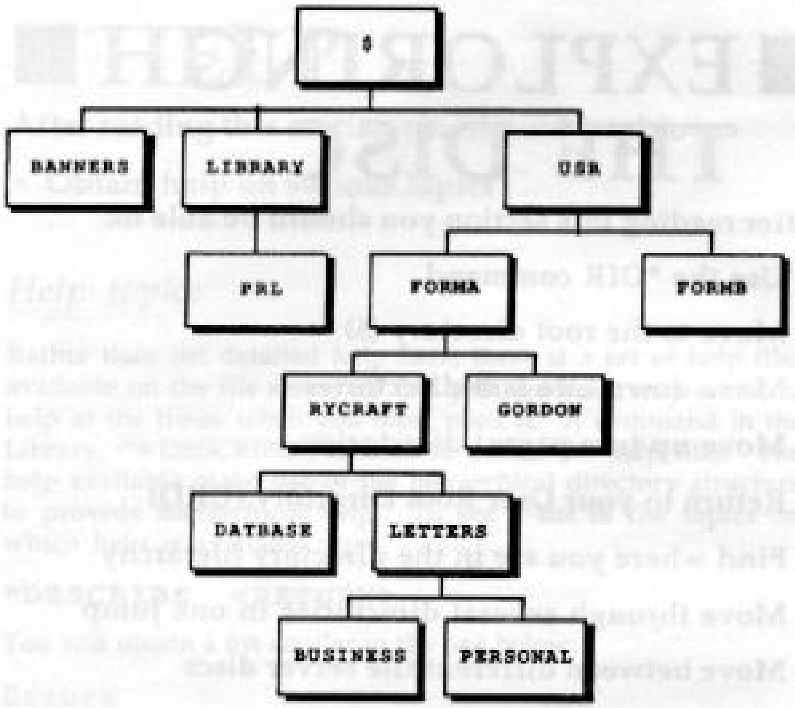
After reading this section you should be able to:

- **Use the *DIR command**
- **Move to the root directory ($)**
- **Move down into sub-directories**
- **Move up into parent directories**
- **Return to your User Root Directory (URD)**
- **Find where you are in the directory hierarchy**
- **Move through several directories in one jump**
- **Move between different file server discs**

*The Root Directory $ and the *DIR command*

Obviously there must be one directory from which the directory hierarchy starts. This directory is called "$" and is known as the *Root Directory*, the top of the family tree. You can enter this directory by using the "*DIR" command which is used for moving between directories:

```
*DIR  $  <RETURN>
```



The "*DIR"command simply moves you to the requested directory; it does not show you what is in it. You will need to type "*CAT" to get a catalogue.

When you catalogue the $ directory you will find that some entries are listed as "...Private". This is used to indicate an area of the disc which a user has decided to hide from other users of the system. You will also notice that where the catalogue header of your URD had "Owner", the root directory header has "This means that in this directory you only have public access and so will not be able to save any programs or data.

You can now try moving to another directory, LIBRARY for example, using "*DIR LIBRARY <RETURN>" and see what is there using "*CAT". One of the entries you will find there is another directory called PRL. You can go down into this using "*DIR PRL <RETURN>".

Having moved down into directories it is useful to be able to go back up again. You can do this using the command:

```
*DIR  ^  <RETURN>
```

This will take you into the parent directory of the directory you are currently in. If you type this in the directory PRL then you will get back to the LIBRARY directory. If you type it again you will get back to the $ directory.
If you know where you are heading in the directory structure then you can go in one step rather than by typing lots of separate "*DIR" commands. To go from the $ directory down into the PRL directory you could use the one DIR command below:

```
*DIR  LIBRARY.PRL  <RETURN>
```

The "." is used to separate the names of the directories. This process is called *Concatenation*. The "^" symbol can also be concatenated in order to move up through several directories in one go. You could go straight from the $.LIBRARY.PRL directory back to the $ directory by typing:

```
*DIR  ^.^  <RETURN>
```

Wandering around the directory structure it is quite possible to become lost! You can find out where you are at any time by typing:

```
*PATHNAME  <RETURN>
```

This will list the names of all the directories in the path from $ down to where you currently are. Alternatively you can always return to the $ directory by typing:

```
*DIR  $  <RETURN>
```

It is also possible to return to your URD, the directory in which you arrived when you logged on, by typing:

```
*DIR  <RETURN>
```

Files in general

Directory entries can be either further directories or files. A file is simply data which can be loaded into a computer's memory. Files can be of many types: BASIC programs, text from a word processor, or machine code programs, to name just a few. When exploring the disc it can be very unclear what sort of file a directory entry actually is. Loading a file of the wrong type can produce odd results. "LOAD" in BASIC assumes that the file being loaded is a BASIC program; if it is not then you will get the error message "Bad program". Similarly, attempting to load a BASIC or machine code program into a word processor will produce rubbish on the screen. Be prepared for such eventualities if you explore the file server and load files without knowing what type they are.

A convention which is used to help you distinguish between different type of file is:

filenames all in capitals — Machine code programs
filenames all in small letters — BASIC programs
any other filename — some sort of data file

Other file server discs

It is possible that your file server has more than one disc attached. Each disc will have a different name so that provided you know the names of the discs you can move between them. The disc name of the disc you are currently on can be found using the "*PATHNAME" command as it is the first name displayed after the colon.

To find what other discs are available type:

```
*FREE  <RETURN>
```

This will display the names of all file server discs together with information about how much space is available.

To move to one of the other file server discs you can use the "*DIR" command. Suppose a disc is called WORK then you can move to the root directory of this disc by typing:

```
*DIR  $WORK
```

Alternatively you can use the "*SDISC" command by typing:

```
*SDISC  WORK
```

The "*DIR" command is perhaps more useful as you can concatenate other directory names on the end so that you can move straight to a directory other than the root directory $.

■ CREATING & DELETING

After reading this section you should be able to:

- **Create directories**
- **Delete files and directories**
- **Use concatenated filenames**
- **Use "&" to refer to your URD**

Creating sub-directories

As you start to save your own files on the file server it is possible for you to get confused over the type of your own files! You can largely avoid this by creating sub-directories in your own URD. eg a "BasicProgs" directory for BASIC programs and "Text" for word processor files. You can create sub-directories using the "*CDIR" command. The two example directories would be created using the commands:

```
*CDIR  BasicProgs  <RETURN>
*CDIR  Text  <RETURN>
```

Having created these directories you can move into them using the "*DIR" command in the same way as any other directory.

Creating sub-directories, and using them as a way of categorising the files that you save, is good practice and makes files much easier to find. A good general rule is that if when you catalogue a directory it is too large to fit on the screen then it is time to create a new sub-directory.

Deleting files and directories

If you have finished with a file then you can remove it from the disc using the "*DELETE" command. To delete the example program type:

```
*DELETE  PROGRAM  <RETURN>
```

You can also delete directories using the "*DELETE" command. For example:

```
*DELETE  BasicProgs  <RETURN>
```

It is not possible to delete a directory unless it is empty. Attempting to delete a non-empty directory will produce the "Directory not empty" error message.

Concatenated filenames

Just as you can use "." to link a series of directory names to you can also use the same technique to load and save files without having to move to the directory which contains them. For example:

```
LOAD"BasicProgs.PROGRAM"  <RETURN>
```

would load the file PROGRAM from the directory BasicProgs. This feature is particularly useful in conjunction with the special directory name "&" which is used to refer to your URD.

Often you will be running software packages in a directory which you do not own. This means that you will not be able to save any data into this directory. However if you use an "&" in the filename for your data then the data will be saved into your URD. For example you might use "&.GraphData" as a filename and this would save a file called GraphData into your URD.

■DISC SPACE

After reading this section you should be able to:

- **Find how much disc space you have left**
- **Find how much disc space you have used**
- **Find how much disc space is available on the file server**

Allocation of disc space

The SJ Research file server uses a system of *Account Numbers* for controlling how much disc space users can use. This system works very much like having a bank account. Whenever you save a file your balance will go down, and whenever you delete a file the balance will go back up again. If you do not have enough credit in your account for saving a file then you will get the "Account bankrupt" error message.

Finding how much disc space you have left

You can find out how much disc space you have left at any time by typing:

```
*BALANCE  <RETURN>
```

this will give a display similar to:

```
Account 1D8  Balance 37 K
```

This means that your account number is 1D8 and that you have enough space left for 37,000 characters - 1K is approximately 1000 characters. N.B. The minimum amount of space needed to store a file or directory is 1K and is always a multiple of 1K, so your balance will always be a whole number of K.

To see how much disc space is actually available on the file server disc as a whole you can type:

```
*FREE  <RETURN>
```

This will list both the amount of space used and the amount free on all discs attached to the file server.

If your file server has more than one disc, then "*BALANCE" tells you how much space you have available for the disc you are currently on. If your URD is not on this disc then your balance is likely to be zero.

"*BALANCE" always tells you about your own *Personal Account* which will be unique to you. You may, however, have access to other accounts which could be used to allow you to share an area of the disc with other users, or to control your access to various network printers. These extra accounts are called *Group Accounts* and are likely to be bankrupt. To find out about all the accounts you have you can type:

```
*STATEMENT  <RETURN>
```

which will give a display similar to

```
Account  Balance
   21    bankrupt       1D8       37k
```

The "*STATEMENT" command can be used to tell you about your account space on all file server discs:

```
*STATEMENT  A  <RETURN>
```

gives balances of all your accounts on all discs

```
*STATEMENT  PA  <RETURN>
```

gives balances of your personal account on all discs.

Finding how much disc space you have used

Finding how much disc space you have used is not straightforward because the file server only knows how much space you have left. The only way to find out is to examine all the files you have saved and see how big they are. A BASIC program is available for doing this; you can start it up by typing:

```
CHAIN"SIZER"  <RETURN>
```

If you simply press <RETURN> at the "Directory name:" prompt then the program will size your URD and all its sub-directories. The size of each file and the total size of directories is displayed, which makes it easy to spot which files can be deleted in order to release lots of disc space.

■FILE INFO & OWNERSHIP

After reading this section you should be able to:

- **Determine whether you own a file**
- **Find the date and time when a file was saved**
- **Find the length of a file**

Finding out more information about a file

You can find out a lot more information about a file than simply the access string which is displayed when you catalogue a directory. The command for displaying further information is "*INFO". For example you could type:

```
*INFO  PROGRAM  <RETURN>
```

The output could look similar to that shown below:

```
PROGRAM      FFFF1200 FFFF802B      0010A7
WR/r     23may88 25oct88 16:55 1D8 (000)
```

The simplest parts of this to understand are detailed below:

"PROGRAM"	is the name of the file as displayed in an ordinary catalogue
"0010A7"	is the length of the file in bytes (number of characters)
"WR/r"	is the access string as displayed in an ordinary catalogue
"23may88"	is the date when the file was first created
"25oct88"	is the date when the file was last updated
"16:55"	is the time when the file was last updated

Sometimes you will find that instead of the date, "today" will be displayed. This makes it much easier for the eye to spot files which have been either created or updated today! This is a great help when you want to check that you really have saved a new version of your file.

The two numbers at the end of the information displayed are also of interest. "1D8" is the main account number attached to the file. "000 " is the auxiliary account number attached to the file

Every file and directory stored on the file server will have two such numbers attached to it. It is these two numbers which determine whether you own the file or directory. If either of the two account numbers attached to the file or directory match one of the account numbers displayed when you type "*STATEMENT" then you own that file or directory. Ownership of a file is important because, in particular, it means that you can delete it. Ownership of a directory is important because you need to own a directory in order to create new files within it.

The other information displayed is not of great interest unless you are a keen programmer: "FFFF1200" is the load address of the file and "FFFF802B" is its execute address.

Finding out more information about a directory

You can also use the "*INFO" command on a directory name, however the information returned is slightly different, for example you might type:

```
*INFO BasicProgs <RETURN>
```

to which the response might be:

```
BasicProgs   Entries=2   Default=WR/r
D/       19dec88 20dec88 14:25 1D8  (000)
```

Here the time stamping tells you the creation date of the directory and the last date and time at which its contents were changed in any way. The number of entries in the directory is displayed, which saves the effort of having to count them by hand from a catalogue. The access string displayed after "Default" is the access string which will be given to new files created in this directory.

It can sometimes be useful to display full information about all entries in directories at once. You can do this using the "*EX" command. The large amount of information displayed by this command is much easier to read if you are in an 80 column screen mode!

SETTING ACCESS

After reading this section you should be able to:

- **Change the access string of a file or directory**
- **Change the default access string of a directory**

Changing the access string of a file

The access string of a file can contain any of the following characters:

R	indicating ability to read the file
W	indicating ability to write in to the file
L	indicating that the file is Locked and so cannot be deleted or overwritten
P	indicating that the file is Private and is hidden from other users

The access string of a file has two parts separated by a slash ("/") . Characters on the left of this indicate what the owner of a file can do and characters on the right indicate what the public can do. The letters in capitals are the access letters which apply specifically to you. If you own the file then the characters on the left of the "/" will be capitals, if you do not own the file then the characters on the right of the "/" will be in capitals.

Changing the access string of a file or directory is one of the things you can only do if you own the file or directory. If you own a file then you can change its access string to any combination of letters you wish by using the "*ACCESS" command. For example:

```
*ACCESS  PROGRAM  LR/R  <RETURN>
```

would mean that PROGRAM was locked and both the owner and the public could load the program.

Rather than specify the whole of the new access string you would like, you can use "+" and "–" to add or remove letters from the access string. You are most likely to want to lock and unlock files or privatize and unprivatize them. You can do this using:

```
*ACCESS  PROGRAM  +L  <RETURN>
```

to lock the file,

```
*ACCESS  PROGRAM  -L  <RETURN>
```

to unlock the file,

```
*ACCESS  PROGRAM  +P  <RETURN>
```

to hide a file by making it private,

```
*ACCESS  PROGRAM  -P  <RETURN>
```

to make a file visible again.

Changing public access to a file

If you wish to alter the public part of the access string then you indicate this by using the "/" character. For example

```
*ACCESS  PROGRAM  -/wr  <RETURN>
```

would mean that the public would be unable to read from or write to this file.

Changing the access string of a directory

The only access letters which a user can apply to a directory are P and L. The D must obviously always be present. Other than this restriction, setting the access string of a directory is no different from setting that of a file.

Making a directory private means that public users will not be able to enter that directory. Whether files in that directory are private or not is unimportant. In this way it is very easy to hide large numbers of files by making the directory containing them private. One useful trick is to make your URD private, which this makes sure that no public users can see any of the files you have saved.

Changing the default access

It can be useful to provide newly saved files automatically with with a particular access. Every directory holds a *Default Access* which is given to new files saved. To see what this default access is you simply type:

```
*INFO  <RETURN>
```

within the directory you are interested in. You will get back information similar to that below:

```
FRED         Entries=2   Default=WR/r
D/       19dec88 20dec88 14:25 1D8 (000)
```

This clearly displays, along with other information, the default access. You can change the default access using the "*DEFACCESS" command. For example:

```
*DEFACCESS  PWR/r  <RETURN>
```

You can also use "+" and "–" in the same way as with "*ACCESS".

■ RENAMING & COPYING

After reading this section you should be able to:

- **Copy a file**
- **Rename a file**

Copying files

This can be done using the "*COPY" command. For example:

```
*COPY   PROGRAM   OLDPROG   <RETURN>
```

This would copy PROGRAM to a file called OLDPROG. Perhaps more sensibly, you would create a sub-directory called OLD in which you could archive the file PROGRAM. In this case you would use a concatenated filename as below:

```
*COPY   PROGRAM   OLD.PROGRAM   <RETURN>
```

To copy a file you only need to be able to read it, so it is possible to copy files which you do not own, but to which you have public read access. However, in order to create successfully the copy of the file you must own the directory in which it is going to be placed.

As copying involves creating a new file, the copy will inherit the account numbers and default access string of the directory in which it is created. It will also have its own creation date and last update date and not those of the original file.

Renaming files and directories

Unlike copying, you can only rename a file if you own it. You may wish to rename a file if you accidentally save it with the wrong name. For example:

```
*RENAME  PRIGRAM  PROGRAM  <RETURN>
```

You can also use "*RENAME" as a method of tidying directories which get too big. For example if you saved all your BASIC programs into one directory you may at a later stage decide to create two sub-directories ART and MATHS and then move your programs into the appropriate directory. For example, you could move PROGRAM into the sub-directory ART using:

```
*RENAME  PROGRAM  ART.PROGRAM  <RETURN>
```

As "*RENAME" is simply moving an existing file the account numbers and times attached to the file remain unchanged.

"*RENAME" can also be used on directories. This allows entire directories and their contents to be moved with just one command.

It is not possible to rename a locked file or directory. You must unlock the file or directory before trying to rename it.

It is also not possible to rename a file from one file server disc to another. If you wish to move a file between discs then the only way to do so is to copy the file and then delete the original.

■ PRINTING

After reading this section you should be able to:

- **Determine what printers are available on your network**
- **Select and use a network printer**
- **Understand the concept of print spooling**

Printer servers in general

One of the design features of Econet networks is to allow networked computers to share printers. Printers can be shared by attaching them to an SJ Research file server or to an ordinary BBC computer running special software. Any machine on the network which can accept printer data from another network computer, and print it out is called a *Printer Server*. The SJ Research file server may have two printers attached to it and so can be viewed as two printer servers and one file server all in the same box. The SJ Research printer server is an extension to the normal printer server in that it attaches names to printers which makes selecting a printer much more user-friendly.

Finding what printers are available

You can find out what printer servers are available on your network by typing:

```
*PSLIST  <RETURN>
```

You will get a list back similar to the one below:

```
000.254 Ready
    PARALL     SERIAL  x LASER
```

The "254" is the station number of the printer server and the "Ready" next to it indicates that it is ready to accept printer data. The three names PARALL, SERIAL and LASER are the names of the printers attached to this printer server. The "x" next to the name LASER means that although this printer name exists it is not possible for you to use it.

To select one of the printers for use the "*PS" command. For example, to use the PARALL printer you would type:

```
*/PS  PARALL  <RETURN>
```

You will get a message back:

```
Station 254 ready
Station 254 selected
```

The "/" has a special meaning and is required to make sure that your computer always behaves in the way you expect. See later section.

You can now use the normal commands for printing: VDU 2 and VDU 3 for starting and stopping printing within a BASIC program or CTRL-B and CTRL-C at the keyboard. If you are unfamiliar with SJ file servers then you will notice that, unlike other printers or printer servers, output does not appear on the printer straight away. In fact there will be no output until you turn the printer off with a VDU 3 or CTRL-C. This can be disconcerting at first but is because the SJ printer server exploits the fact that it is also a file server to store printer output on disc before printing it. This is called *Spooling*.

Print spooling

The problem with ordinary printer servers is that they can only cope with one user at a time. If the printer server is busy then another user who tries to use it will get an error message. Print spooling avoids this problem by temporarily storing each user's printer data in a separate file in a special directory on the file server's disc. As a directory can contain many files, this means many users can send printer data simultaneously. Only when a user ends printing with a VDU 3 or CTRL-C does the file server take the printer data from the temporary file, and send it to the printer. Obviously there is a fair queueing system and in fact items will come out on the printer in a 'first finished, first printed' manner!

Other printer servers

If your network has a BBC with a printer attached and it is running the Acorn printer server software then it will also appear in the list of printers supplied by "*PSLIST". For example:

```
000.254 Ready
    PARALL  *   SERIAL
000.235 Ready
```

The Acorn software does not allow the printer to be named so if you wish to use this printer server you will need to specify the station number instead ,using:

```
*/PS  235  <RETURN>
```

Only one user at a time can access Acorn printer servers because they do not support print spooling.

■ WILDCARDS

After reading this section you should be able to:

- **Use the wildcards * and # in file names and directory names**
- **Delete using wildcards**
- **Use *ENABLE**

Wildcards in general

Wildcards are special characters which can be used to replace some or all of the characters in a filename. This can save you typing the whole of a filename, or allow, you to perform operations on several files with a single command. Files which will be affected by the command you type are said to *match* the wildcard name.

Wildcards can be used as part of filenames in any circumstance where you would normally enter a full directory or file name. You can use wildcards with "*ACCESS", "*DIR", "LOAD", "*INFO", "*RENAME, "*DELETE" amongst others. These operations fall into one of two categories; those which will perform the relevant operation on all files which match, and those which will perform the operation on one file only. In the latter case, if there is more than one match then the first name found which matches the name specified will be the one selected (remember that catalogues are always ordered alphabetically).

First matching name	All matching names
*DIR	*ACCESS
*INFO	*RENAME
LOAD	*DELETE

Deciding which category any particular command falls in to is usually fairly straight forward. It makes no sense to select several directories simultaneously with a "*DIR" command or to load several files with "LOAD".

The # wildcard

The # wildcard is used to replace any other single character in a file or directory name. For example, rather than type

```
LOAD"PROG1"   <RETURN>
```

you could type

```
LOAD"PROG#"   <RETURN>
```

or

```
LOAD"#ROG1"   <RETURN>
```

or

```
LOAD"##OG#"   <RETURN>
```

subject of course to PROG1 being the first match found for all these possibilities.

*The * wildcard*

This is a much more powerful wildcard as it will match any number of characters. Use of * in long pathnames can save a lot of typing. Suppose you wanted to move to the directory $.SOFTWARE.PHYSICS.ALEVEL.MECHANICS. Rather than type this in full, you might use the command:

```
*DIR  $.S*.P*.A*.M*  <RETURN>
```

This again assumes that you know enough about the file server directory structure to be sure that the first match for this name is actually the one you want. If there was a directory called SATELLITE in $ then, as this before SOFTWARE in the catalogue, this would be selected and you would almost certainly get an error message. To avoid selecting SATELLITE you could type instead:

```
*DIR  $.SO*.P*.A*.M*  <RETURN>
```

You should never use wildcards for accessing files or moving directories within a program as creation of other files in the future could mean that your wildcard names no longer match what you were expecting.

The * is very useful for matching all directory entries. You could make every directory entry private with the single command:

```
*ACCESS  *  +P  <RETURN>
```

Deleting with wildcards

This is a dangerous operation as it will delete several files with a single command. For this reason it is likely that your system manager will have provided you with some protection against accidental deletion using wildcards. This protection takes the form of having to type:

```
*ENABLE  <RETURN>
```

before it is possible to delete using wildcards. The *ENABLE stays in force until such time as you log on again or type:

```
*DISABLE  <RETURN>
```

Suppose you had a directory containing PROG, PROG1, PROG12, PROG2 and PROGa

```
*DELETE  PROG#  <RETURN>
```

would delete all these programs except PROG and PROG12, ie all five character filenames starting with the letters PROG.

```
*DELETE  P*  <RETURN>
```

would delete all the programs, ie names of any length which start with P. In particular note that a file P would also be deleted so the * has matched no characters in that case.

■BOOT FILES

After reading this section you should be able to:

- **Set an appropriate Boot Option**
- **Write a short Boot File**

Boot files in general

Booting a computer system is a method of making it run a program automatically when it is started up. On the network this means users can start a program running automatically when they log on. The sequence of events which will happen when you log on is determined by the contents of the file called "!BOOT" in your URD together with your *Boot Option* setting. The boot option setting tells the computer what exactly to do with the !BOOT file.

Setting your boot option

This is done using the "*OPT 4 , n " command when "n" is in the range 0 to 3. The values of "n" have the following meanings:

0	-	Off	-	Ignore any !BOOT file
1	-	Load	-	*LOAD the !BOOT file
2	-	Run	-	*RUN the !BOOT file
3	-	Exec	-	*EXEC the !BOOT file

The easiest way of finding out what your boot option is currently set to, is to display a catalogue. The current boot option setting is always displayed as part of the catalogue header. Option 1 is seldom appropriate and option 2 only works if the !BOOT file is machine code. Unless you are an experienced programmer the only options you are likely to want are 0 and 3.

*Writing a boot file for *EXECing*

*EXECing a !BOOT file reads commands from the file and executes them as if they had been typed in at the keyboard. The way to create such a !BOOT file is to use the "*BUILD" command as follows:

```
*BUILD  !BOOT  <RETURN>
```

You will get a "1" prompt back indicating the computer is ready to accept the first line of your sequence of commands. You can now enter your sequence of commands pressing <RETURN> at the end of each line and finally pressing <ESCAPE> on a blank line to finish. A typical !BOOT file might contain the following:

```
1 *BASIC
2 PAGE=&7000
3 CHAIN"MENU"
4
```

Setting PAGE to &7000 is a wise precaution to avoid losing any program that you have in memory as a result of logging on.

■PROTECTION

After reading this section you should be able to:

- **Protect and unprotect your computer**

Protection in general

Because the network cable links all the computers on the network together, it is possible for any two computers to communicate with each other. Such communications fall into two categories;

1. Communication which requires both computers to be actively participating in the exchange by running appropriate software.

2. Communication which does not require the active participation of the second computer but which momentarily interrupts whatever it was doing.

The second type of communication can be quite irritating, as other users of the system may send you unwanted messages which suddenly appear in the middle of whatever you are doing. They may also copy whatever is currently on your screen on to their screen and in this way see what you are typing. This is not desirable if you are working on something private!

Setting protection

You can stop other computers from using the second type of communication by typing:

```
*PROT   <RETURN>
```

This protects your computer against interference from other users until such time as you press <CTRL><BREAK> or decide to remove protection using "*UNPROT". It is normally sensible to protect your computer unless you have some specific reason for not so doing. When you start using a computer you cannot be sure whether it is protected or not, so users will often include a "*PROT" command in their boot sequence.

Removing protection

You can remove protection by typing:

```
*UNPROT   <RETURN>
```

SPECIAL DIRECTORY NAMES

After reading this section you should be able to:

- **Access various special directories using a single character**

Special directory names

Some directories on the file server are frequently used, and for this reason have been given an alternative single character name which provides a quick way of accessing these directories and entries within them. The special directories are:

LIB - %
CSD - @
URD - &
Parent Directory - ^

All these symbols can be used in filenames and can be concatenated.

■STAR COMMANDS & THE LIBRARY

After reading this section you should be able to:

- **Access programs in the Library**
- **Change your Library**

STAR commands in general

Typing a ✱ command starts a piece of machine code of that name running. This might happen in any one of three ways.

1. Code inside one of the ROMs in your computer is run. Eg ✱I AM
2. Code inside the file server is run. Eg ✱ACCESS
3. Code is loaded from the file server disc into your computer and run. Eg ✱USERS

It is possible that the name of the command you type could be run in any combination of these ways. The command is offered in the order above and only the first which responds will actually occur.

The Library

There are many star commands which are so useful that the file server has a method of making them available no matter what directory you are in. The way the file server achieves this is by allowing the user to have a *Library* directory. If a command name is not found in your CSD the file server will then search the library directory as well. By default the library directory is $.LIBRARY on disc 0 and this will almost certainly have the largest number of entries of all the directories on your file server.

It is worth cataloguing $.LIBRARY as you will find that a large number of the star commands which you will use frequently are actually machine code programs in the library.

The library directory can be changed using the "`*LIB`" command. For example you could change the library directory to $.BEEBTEL by typing:

```
*LIB  $.BEEBTEL  <RETURN>
```

This is not something you will wish to do often as it makes all the useful commands in $.LIBRARY much more difficult to access. The name of your current library directory is always displayed as part of the catalogue header.

If you are used to Acorn file servers then you should note that the SJ file server differs in that it is also searches the library for loads and opens. If you wish you can disable this extra feature and have an Acorn style of library by using the command:

```
*DISABLE  LIBRARY  <RETURN>
```

A full library search can be restored by using the command:

```
*ENABLE  LIBRARY  <RETURN>
```

Forcing execution of the right piece of machine code

There are unfortunately occasions when the same name is attached to three possible different types of machine code described above. Under these circumstances because of the order in which the name gets offered you will not get the piece of code you wanted. The worst, and most confusing, case is when a name exists in all three guises. To get round this the special symbols "\" and "/" can be used which work in the following way:

```
*name    <RETURN>
```

to execute code in a ROM inside your computer

```
*\name    <RETURN>
```

to execute code in the file server

```
*/name    <RETURN>
```

to load and execute code from the file server disc.

SJ Research
File Server
User Manual

Draft Copy January 1989 [illegible] SJ Research