Appendix K

PRINTERS

Printers are categorized according to the method they use to produce print and how they interface to the computer. The BBC Model B microcomputer can be used with most present-day printers and a brief discussion of the more popular types and how they are used is given in this appendix.

DOT-MATRIX PRINTERS

The most popular type of printer is the dot-matrix printer which, as the name implies, has a print head with a matrix of pins, typically 5×7 although some are available with many more pins, giving a much better quality print. Producing characters as an array of dots provides flexibility in the types of characters produced, and a common facility is that of several, switch selectable, character sets, for use in different countries − UK, USA, Germany and Sweden for example. Another advantage is that some, but not all, dot-matrix printers can be used in graphics mode. This means that in conjunction with a suitable program often called a screen dump, they can be used to produce a hard copy of the contents of the screen when the computer is in graphics mode. Dot-matrix printers are relatively inexpensive.

DAISY-WHEEL PRINTERS

For high quality production of text, the more expensive daisy-wheel type of printer is superior. The characters are, of course, fixed by the daisy-wheel but the wheels can be interchanged to produce different size and type of characters, including Greek letters and mathematical symbols. Certain more expensive models are available with twin daisy wheels which are software selectable. Daisy-wheel printers cannot easily be used to produce a screen dump.

INTERFACING

A printer can be either a serial or parallel type, depending upon the way it interfaces to the computer. A serial type can normally be driven with just three

wires and because of the low-cost and compactness of the cable is suitable where the printer is to be stationed at some distance from the computer or is to be switched to serve a number of machines. Where the printer is to be used adjacent to the computer, then serial or parallel types are equally suitable although the latter may have a small advantage in terms of cost.

The BBC computer provides a parallel printer port, often referred to as a Centronics compatible port, implying that it satisfies prescribed standards for interconnections and protocol. It emerges via a 26-way socket accessible from the front underside of the computer case and is connected using a 26-way ribbon cable. For connection to a serial printer, the serial RS423 port is used. This is accessed from the rear via a 5-pin domino DIN socket.

USE OF PRINTERS

In order to send output from the computer to the printer it is necessary to give the computer a series of commands. On power-up the computer will automatically select the parallel port for printer output. If a serial printer is being used then the serial port can be selected by means of the command *FX5,2. The parallel port can be reselected by means of the command *FX5,1.

When printing, an additional optmn is to switch off the screen display. This is just one of the uses to which *FX3 can be put. If you would like to send output to the printer without it appearing on the screen, issue *FX3,3 before the output and *FX3,0 after output is completed. Note that *FX5,2 must also be issued if necessary. In principle, an alternative to both *FX commands is to simply issue *FX3,7, which disables the screen and enables the RS423 port simultaneously. However, this may not be advisable since it merely selects the RS423 socket as the output stream; it does not recognize it as a printer output stream so the *FX6 command, for instance, will be ineffective.

In the case of a serial printer the rate at which information is transmitted will have to be adjusted to match one which the printer is capable of receiving. Usually this would be the highest rate available, provided that the printer has ‘handshaking’ signal lines to stop the computer transmitting when it cannot receive any more information.

Information transmission rates are measured in baud with a typical value for printers being 300 baud.

Transmit baud rates can be changed with the following statements

***FX8,1**	75 baud
***FX8,2**	150 baud
***FX8,3**	300 baud
***FX8,4**	1200 baud

(Higher baud rates are possible, but are unlikely to be of use for printer output.)

Once the port has been selected and, in the case of the serial printer, the baud rate has been selected, the printer can be *enabled*. This means that the

computer is instructed that all output that goes to the screen is also to be sent to the printer. It can be done directly from the keyboard by means of CTRL-B. Similarly, the printer can be turned off by means of CTRL-C. To turn a printer on and off during execution of a program the statements VDU 2 and VDU 3 can be used respectively (or *FX 3,8 − see Appendix B).

In addition to the parallel and serial printer outputs, computers connected to an Econet network can use communal printer facilities provided by the network. The Econet output is selected by *FX5,4.

CHARACTER SUPPRESSION

Some printers automatically line feed when a carriage return is received. Therefore the effect of a carriage return and line feed being transmitted to this type of printer would be to produce a double line feed. This problem can be overcome by instructing the computer to suppress line feeds. The command *FX6,n will suppress the character with ASCII code n and the command *FX6,0 will cancel any previous suppressions. The ASCII code for line feed is 10 and therefore *FX6,10 will suppress line feeds. The default on the BBC computer is in fact to ignore line feeds on power-up. Therefore, if the printer does not automatically line feed, the command *FX6,0 must be used to enable line feeds.

If you do not need to suppress line feeds, you could use the *FX6 command to economize on paper during printer output from within a program by issuing *FX6,12, which will suppress any page feeds generated by CLS commands in the program.

The sequence of commands to list a program is therefore as follows

Parallel

***FX5,1**	(necessary only if serial port previously selected)
***FX6,0**	enable line feeds (if required)
CTRL-B or VDU 2	turn on
LIST	
CTRL-C or VDU 3	turn off

Serial

***FX5,2**	select serial port
***FX3,n**	n=1, 2, 3 or 4 according to baud rate
***FX6,0**	enable line feeds (if required)
CTRL-B or VDU 2	turn on
LIST	
CTRL-C or VDU 3	turn off

Econet

***FX5,4**	select Econet
***FX6,0**	enable line feeds (if required)
CTRL-B or VDU 2	turn on
LIST	

CTRL-C or VDU 3 turn off

DETECTING THE PRESENCE OF A PRINTER

If output is directed to a printer when no printer is connected, the computer will hang once the output buffer is filled. A solution is to send the first N characters to the printer (N must be in the range 2<=N<=63), then check the free space in the printer buffer by

```
X=ADVAL(-4)
IF X=64-N THEN PRINT CHR$(3);"NO PRINTER CONNECTED": GOTO...
```

If the free space is not greater than 64-N, then nothing has been transmitted and preventive action can be taken.

Appendix L

THE ECONET FILING SYSTEM

A computer network is a system for linking together a number of computers to permit commuriication between users, and also to enable them to share resources such as common disc storage and printer. It acts as an alternative filing system for the computers attached to the network.

The Econet is a network designed for use with all Acorn computers. The block diagram in Figure L. 1 shows a possible configuration for this network.

To connect computers to the Econet it is necessary to install an interface. This involves the use of a special ROM for software and the MC68B54 Advanced Data Link Controller which forms part of the necessary hardware. The ROM contains the software for a filing system called the Network Filing System (NFS) which is similar in some respects to the Disc Filing System (DFS) and the Cassette Filing System (CFS). The network filing system is selected by the command

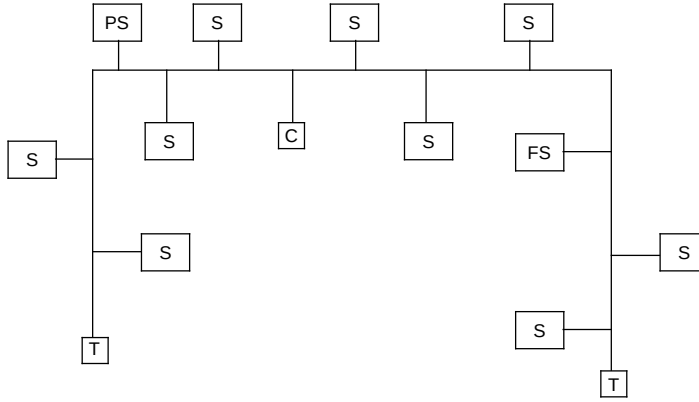
```
*NET
```

In an Econet system up to 254 BBC microcomputers can be connected together. Each microcomputer connected to the network has a station identity which is a number between 1 and 254. This number is set by a switch inside the computer case. No two computers on a network can have the same station identity. The diagram of the network shows that, in addition to the user stations, there are usually two special purpose computers: the file server.

Which together with a disc drive provides storage for the network and a printer server, which together with a printer, is available for program listings and output of data. A network may have more than one file server or printer server.

The printer server can be used as a normal station except when it is accessed by another user on the network to print a file. The printer server Heeds an additional special ROM containing the printer software. The Iletwork computers assume that the file server has a default identity of 254 and that the printer server has the default address (station identity) of 235. The computer (or computers) that acts as a file server is dedicated to that purpose and runs a special program supplied on a floppy disc. Two versions of the file

server software have been released and are in widespread use. These are referred to as level 1 and level 2.



Key
C Clock
FS file server
PS printer server
S station
T terminator

Figure L.1 The arrangement of a network.

Computers that have the Econet system installed will usually be arranged to default to the Econet filing system on switch-on and after a <BREAK>. It is possible to cause a <BREAK> or switch-on to select another filing system even though the Econet system is installed, as follows

<CTRL-D>+<BREAK> selects DFS (Disc Filing System)
<CTRL-T>+<BREAK> selects CFS (Cassette Filing System)
<CTRL-N>+<BREAK> selects NFS (Network Filing System)

Assuming that the file server has been started by the system manager, and the network is in operation, then each user station can access the Econet filing system.

After a switch-on with the network selected, a computer fitted with the Econet will respond thus

```
BBC Computer 32K
Econet station nnn
BASIC
>
```

where nnn is the station identity. The other filing systems can be selected in the usual way by *DISC or *TAPE without loss of program or data. Thus programs may be transferred between different filing systems.

LEVEL 1 ECONET

The range of facilities available to the network user, and the way in which they can be accessed by the Econet station user, depend upon which level of Econet the file server is running. A network can have file servers working with either level at the same time. The protocols for the transfer of data on the network are independent of the level of software. However the types of data that can be transferred on the network are limited by the level of Econet software.

On level 1 the user can select a file server by the command

```
*I AM nnn
```

where nnn is the station identity of the file server selected. As the default station number is 254 it is not necessary to select this after switch-on.

On level 1 the file server has a list of valid users identified by a single letter from A to Z, which is also the name of their directory on the file server. The station identity is paired with the user nameldirectory so that each station can only have access to one directory. The directories for different stations can all be different or some can have common directories. Thus a station cannot have more than one directory assigned to it, but a directory can have several stations assigned to it. The allocation of directories to stations is set up by the system manager whenever he starts the system up. This means that the users must normally use the same machine each time, in order to have access to their own files.

The level 1 file server software can be run on a Model B microcomputer With any type of disc drive and Econet interface. It does not require the 6502 Second processor, which is needed by level 2 file servers. The level I file Server is a BASIC program which uses the DFS filing system. This limits the total number of files available to the Econet system users to a maximum of about 120 on each file server and that only if a twin double-sided disc drive is being used.

In addition to the individual directories indicated by a letter, allocated to each network station, there is a library directory, &, which is available to all users on the network. The library cannot be written to while the system is running but since the file server discs are in normal disc filing system format they can be written to whilst the file server is operating as an ordinary DFS machine. The network computers automatically select the & directory on

drive 0 as the library. The & directory can only be accessed on drive 0; it can not be accessed on any other drive. Thus the library must be on drive 0. The library directory & is intended for the system utility programs which are used by all users, and for demonstration programs for the whole class to use. One of the advantages of the level 1 Econet system is that the discs can be prepared on an ordinary DFS machine, unlike level 2.

Another advantage of the Econet system is that it can provide a printer server. This means that a whole group of users can access a common printer. An Econet network can, however , have a printer server without having a file server.

The range of commands available on the level 1 Econet system is restricted and these are listed in Table L.1. The absence of data file handling commands is a severe limitation to the use of level 1. Another restriction is that, as the system is run by a file server without a second processor, the maximum length of file that can be transferred across the network is also limited. The latter restriction can be overcome by saving blocks of memory with *SA VE and using *LOAD to reassemble a file which would otherwise be too large.

Table L.1 Level 1 Econet system commands.

Command	*Argument*	*Explanation*
SAVE	**"XYZ3789"**	Save a BASIC program in file XYZ3789
LOAD	**"XYZ3789"**	Load the BASIC program XYZ3789
***CAT**		Catalogue the files of a particular user
***DELETE**	**<fsp>**	Delete a specified file.
***DRIVE n**		Select the disc drive surface in use (n=0 to 3).
***ACCESS**	**<fsp> L**	Lock (L) or unlock ( ) a file. Locked files cannot be deleted or overwritten
***I AM**	**nnn**	Select file server by stated identity.
***SAVE <fsp> start finish** ***SAVE <fsp> start+length**		Save an area of memory on disc giving start address and finish address or length
***LOAD**	**<fsp>**	Load file into memory.
***RUN**		Execute a machine code program
***INFO**	**<fsp>**	Displays information about a file
***<fsp>**		Execute a machine code program as a command
***/**	**<fsp>**	Allows <fsp> to take reserved names such as CAT so that you can write your own utilities.
***PS**	**nnn**	Select station nnn as print server
***FX 5,4**		Special *FX call to select Econet printer. Use CTRL-B and CTRL-C in the usual way to start and finish printing.

The file server disc drives may be single or dual, double-sided or single-sided drives. If two single-sided drives are used, then only drives 0 and 1 are available, whereas if double-sided drives are used, then drives 2 and 3 can also be used. The system and library files are stored on drive 0 which means that there is limited space on this drive. It is therefore advisable for users to save programs to one of the other drives. You can change drive by use of the command

```
*DRIVE n
```

where n=0, 1, 2 or 3, and subsequent LOAD or SAVE commands will refer to the selected drive. Alternatively, you can SAVE or LOAD from a different drive without changing the selected drive by preceding the filename with :n where n is the drive number.

The commands in Table L. 1 form part of the Econet software and are always available to the network user. A second group of commands or utilities would normally be made available to the user at the discretion of the system manager and these are given in Table L.2. A utility is a separate program that can be used in the same way as a command. The utility programs are stored in the & directory and are *RUN by the *<filename> command which will search the selected library if a file of the given name is not in the user's directory. The system manager can choose which utilities are available to the users by the range of program files he provides.

Table L.2 Level 1 network utilities.

Command	*Argument*	*Explanation*
***PROT**		Protect station from remote control.
***REMOTE**	**nnn**	Take remote control of station nnn.
***VIEW**	**nnn**	Take copy of VDU from station nnn.
***NOTIFY**	**nnn**	Send message to nnn.
***UNPROT**		Unprotect station.
***FS**	**nnn**	Change file server to station nnn.
***ROFF**		Remote off.

The Econet Level 1 system can generate a range of error conditions outside those met in a non-Econet environment. These are listed in Table L.3.

Table L.3 Additional Level I error messages.

Message	*Explanation*
`Directory full`	The number of files each user may store is restricted at the start-up of the system. You cannot exceed this limit.
`File not found`	The Econet system only looks on the selected drive for a requested file.
`Insufficient access`	The file server does not recognize you as a user station. The identity of the computer is not in the list given.
`Not listening`	Your command has not been accepted by the file server or print server.
`Too much data`	This arises when the file length limit is exceeded.
`Line jammed`	Information permanently present on the network.
`No clock`	The network clock signal is not reaching the station.
`No reply`	The file server is not responding to your command.

The last three error messages require the attention of whoever is in charge of the network. The procedure for starting the network is not discussed here since this will usually be the responsibility of a designated person and this text is intended for the general user and not the system manager.

LEVEL 2 ECONET

The level 1 Econet system provides a very rudimentary network capability. The level 2 system provides a full range of facilities so that individual station users have nearly the same range of facilities as an individual user with the DFS and a disc drive. In addition, the station user is able to work in an environment akin to that of a mainframe computer system in terms of access to the files of a wide range of other users.

The level 2 Econet system supports a hierarchical directory system and does not use the standard disc filing system (DFS), so several of its limitations are removed.

The root directory is the $ or system directory and SYST is a user who has access to this directory for all purposes. New directories and users can be created by SYST. Each user has a main directory which has the same name as his user name. Only SYST can create a user, but any user can create further directories. The system has a password facility so that only authorized people

can access a particular directory, usually their own user directory. The procedure for connecting to the network is called ‘logging on’ and it is done by use of the command

```
*I AM nnn USER3 PASS
```

where nnn is the file server station selected. if other than the default station 254, USER3 is the name of the user, and PASS is the password for this user.

Unlike level I, a user of the level 2 Econet system can use any of the stations on the network. It is not necessary to associate user names with the station identity as in level 1. The user name can consist of up to ten characters which must start with a letter. The password can be up to six characters long. The first time a user logs on to the system, the password will usually be blank.

If a user who is not known to the system tries to log on, the error message ‘User not known’ will be displayed. The system manager is responsible for creating new users, and for creating a user root directory for them. A user who does not have a directory can log on, but will not have any storage space allocated, and will only be able to read and use files on public access. At logon a user will normally have his own root directory as his selected directory. Extra security of passwords is provided in later versions of the Econet ROM in which the following sequence is permitted and will not reveal the password on the monitor screen

```
*I AM USER3 <RETURN>
PASS <RETURN>
```

In this case the second line will not be shown on the monitor screen. Use of the wrong password by a valid user will produce the error message ‘Wrong password’. The user will not be logged on by the system and he must try again. At the end of a session at a station on the Econet the command

```
*BYE
```

logs off the user. The *BYE command is issued automatically during a logon so that only one user can be logged on at a station at one time. Until a user has logged into the network the only Econet file server command accepted is *I AM <user> <password>. The manager sets the number of stations that are to be served by a file server, and if an attempt is made to log on to a file server which has its full complement of users, the message ‘Too many users’ is sent.

The network stations on the Econet may include stations with DFS and disc drives. The range of standard DFS commands available on the Econet to non-DFS machines is nearly as extensive as those available on DFS machines. The commands on the DFS and Econet filing system (NFS) are as nearly compatible as possible. The DFS commands with no equivalent are *BACKUP, *BUILD, *COMPACT, *COPY, *DESTROY, *DRIVE, *DUMP, *ENABLE, *LIST and *TYPE, but a system can have library utilities to provide some of these facilities, for example *BUILD, *DUMP and *TYPE.

(See, for instance, *Networking with the BBC Microcomputer* by R G Napier, published by Prentice-i-iall International.)

The commands that are different or modified are * ACCESS, *CAT, *DIR and *INFO. In addition to the DFS commands and those discussed for level 1, level 2 has the following additional commands available

```
*BYE  *CDIR  *DISCS  *EX  *LIB  *PASS  *SDISC
```

Table L.4 gives a summary of these commands and the modified DFS commands.

Table L.4 Level 2 Econet system commands

Command	*Argument*	*Explanation*
***BYE**		Logs the user off
***CDIR**	**<dir name>**	Creates a directory.
***DISCS**		Displays the names of the discs currently in use.
***EX**	**<dir name>**	Gives extensive information about files in directory <dir name>
***I AM**	**nnn <name>**	The logon command to station nnn by user <name>.
***INFO**	**<fsp>**	Gives information about a file. It cannot be used with wildcards as in the DFS.
***LIB**	**<dir name>**	Selects <dir name> as the library directory.
***PASS**	**"" <pass>**	Sets user password to <pass>.
***PASS**	**<old> <new>**	Changes user password from <old> to <new>.
***PS**	**<nnn>**	Selects the printer server to be a station other than the default station 235. It is assumed the selected station is suitably equipped.
***SDISC <disc name>**		Changes the disc available to the user.

All the commands in Table L.1, except *DRIVE, are available to the level 2 user. The format for the commands *ACCESS, *INFO and *I AM are modified. The arguments of the file handling commands and *CAT are modified by the use of pathnames.

The majority of the modified commands that differ from their DFS counterparts can, in general be used as though there were no difference. The *ACCESS command is the most changed command due to the wider range of facilities offered by the network environment. The format for this command in the Econet situation is

```
*ACCESS <fsp> <owner access>/<public access> L
```

The presence of L indicates that the file is locked, to prevent its deletion. The owner access can be R for read and/or W for write, and the public access may be nil or read R. Thus

```
*ACCESS SAMPLE R/ L
```

sets the file in the currently selected directory called SAMPLE for read access by the owner, and at the same time prevents its deletion. Another format might be

```
*ACCESS MINEONLY RW/ L
```

or

```
*ACCESS EVERYONE RW/R L
```

The default setting of file access on creation of a file is RW/ . The access control is limited to either public access or no public access. There is no way of limiting access to a group of users.

The *PASS command is to enable a user to set a new value for a password. The format is

```
*PASS <oldpass> <newpass>
```

or, in the case of the initial setting

```
*PASS "" <pass>
```

The quotation marks are used to identify the empty string.

The *DISCS command enables a user to find out which discs the file server has loaded. A file server disc has a name to identify it. A user who needs a disc not currently loaded can then ask the system manager to load it, in place of one of those currently available. The command *SDISC allows a user to select the disc in the other drive for use, thus

```
*SDISC <name>
```

The user will need a directory, created by the system manager, on each disc on

which he wishes to store programs or data.

The selection of a current library, by use of

```
*LIB <directory name>
```

is restricted to the disc selected by *SDISC, and so does not have the power of the similar DFS command. The library must be reselected after using *SDISC. The default library directory selected at logon and on change of disc by use of the command *SDISC is the directory 'Library'. 'Library' contains the system utilities given in Table L.2.

The *DIR command is used to select a directory on the currently selected disc. After a change of disc the user has his root directory as the currently selected directory. If any other is required then the command

```
*DIR NEWDIR
```

has to be used to make NEWDIR the currently selected directory. In order to select a directory such as 'Library' or that of another user, it is necessary to ascend to the root directory and then progress to the directory required by the appropriate route. The return to the user's own directory is made by

***DIR** <RETURN>

without specifying the name or root. It will be found that on return to the user's directory by

```
*DIR $.<username>
```

that the user only has public access rights. To regain the normal rights of a user logged into their own directory it is necessary to use *DIR alone, logon again or to use the *SDISC command as though to change the selected disc.

A similar restriction and limitation to that for *LIB also applies to *CAT. Only directories on the selected disc can be catalogued and, unlike the DFS, a directory must be created before it is catalogued. A directory is created by

```
*CDIR <name>
```

This command can be issued by a user to create a sub-directory. Figure L.2 illustrates the hierarchical structure of the directories.

The format for the catalogue command is

```
*CAT <name>
```

If no name is given, the user's currently selected directory is catalogued. If the catalogue of another directory is required, then either that directory must be made the selected directory or the pathname of that directory must be given.

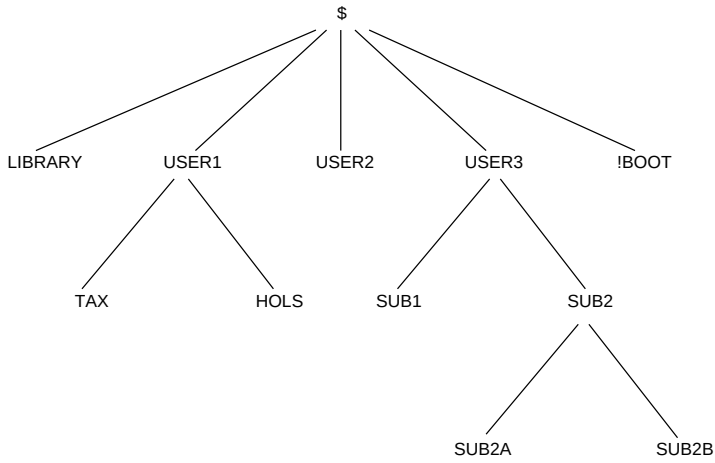


Figure L.2 The directory structure for level 2

For instance

```
*CAT $.Library
```

will catalogue the public access files in the directory ‘Library’ and

```
*CAT $.USER3.SUB2.SUB2A
```

will catalogue any files in USER3’s sub-directory SUB2A which are on public access to any other user issuing the command.

The files may be referred to by pathnames in which the directory route to − locate a file is given by a list of directory names, separated by the full stop. For example, the library file LSQ could be loaded by the command

```
LOAD "$.Library.LSQ"
```

without the need to change the currently selected directory (provided, of Course, that the access control for public read is set). It is generally advisable − to use the pathname, rather than to change directory to use a file in another directory, as it is then no longer necessary to use *DIR <RETURN> to regain owner rights in your own directory. The use of pathnames with files provides a convenient means to refer to files not in the currently selected directory. All the file handling commands can be used with pathnames as a prefix provided

that the access sought is allowed. (A useful tip if you are using a prefix frequently is to program it into a function key.)

The *INFO command is less flexible than that available with the DFS as it is no longer possible to use the wildcards * and # to give information about a group of files. It is limited to information about one file only, the first one found to fit the description. In the NFS environment, this command can also give further information about a directory. The level 2 Econet command *EX compensates for the loss of the wildcard facility to some degree, as

```
*EX <directory name>
```

is equivalent to issuing a *INFO command for each of the files in the directory named.

The information displayed about files with *EX and *INFO is as follows for each file

```
NETFILE   FFFF1200   FFFF8023   000043
 WR/      14:4:85    000147
```

where

NETFILE	is the filename
FFFF1200	is the reload address
FFFF8023	is the execution address
WR/	is the access currently set
000043	is the file size
14:4:85	is the most recent date on which the file was written to
000147	is the System Internal Name (SIN)

The date is set each day by the system manager when the level 2 system is started.

The filenames on level 2 Econet can be up to ten characters long, and can contain any combination of letters and numbers. The following symbols may also be used

```
! % & = - ~ ^ | @ {£ _ + ; ) ] < > ? /
```

but not the space character. No distinction is made between upper- and lower-case letters.

The *NOTIFY command can either be used with station identity or with user name. Thus both of the following messages, sent from station 121, are valid

```
*NOTIFY USER3 It is time for lunch <RETURN>
*NOTIFY 235 Ptease witch the printer on <RETURN>
```

The station at which USER3 is located will ‘bleep’ and display the message thus

```
--121: It is time for lunch
```

The message is not followed by a return so that the receiving user can delete it and continue with his own activity. The message at station 235 is

```
--121: Please switch printer on
```

The number prefixing the message gives the station identity of the source of the message.

The range of network utilities available can be controlled by the network manager since he can change the access rights of the users to the files in the directory ‘Library’, which is the equivalent of the & directory in level 1. The utilities are programs that can be *RUN by the use of the abbreviation *<filename>. The user’s own directory is scanned first for the file <filename>. If the file is not found in the owner’s directory the ‘library directory’ is then scanned and if it is found then <filename> is *RUN. The library directory is either that last set by the *LIB command or if no change has been made it will be the directory ‘Library’ which is the default library directory set by the system during logon using the *I AM command. The system manager may choose to limit the range of utilities available to general users but this does not prevent their use by privileged users. This option has clear applications in class situations where a teacher may not want pupils to have the same access to the computers of other users that he can have.

The additional error messages of the level 2 Econet system are given in Table L.5.

Table L.5 Level 2 Econet error messages.

Message	*Explanation*
`Bad attribute`	*ACCESS has been used with an inappropriate access string.
`Bad password`	Whilst using *PASS, the new password is not acceptable.
`Bad string`	The file or pathname is too long or, the " symbol has been omitted.
`Broken dir`	The directory has been damaged. The system manager is responsible for keeping backup copies.
`Channel`	Corrupted memory (RAM) in user station. Log on again.
`Dir. full`	The directory on the selected disc is full.

Message	Explanation
Dir . not empty	An attempt has been made to delete a non-empty directory.
Disc changed	A change in file server disc has occured
Entry locked	It is necessary to use *ACCESS to unlock a file.
EOF	You have tried to read after the End Of File.
Insufficient access	You have tried to get access to a file for which you do not have sufficient access rights.
Insufficient privilege	There are certain commands only available to the system manager.
Is a dir.	An attempt to make incorrect use of a directory.
Not found	The file or directory specified is not where you expected.
Not listening	This can occur if you try to use the printer while it is busy.
Not logged on	The only command available to a user who is not logged on, is *I AM.
Not open for update	You are trying to write to a read-only file.
Outside file	Occurs with OPENIN in BASIC I. An attempt has been made to put the random access pointer past the file end.
PW file not found	This message may occur at logon if there is no password file on the file server
Rename across discs	The *RENAME command does not allow a file to be moved between different directories on different discs.
Too many open files	In general data file access, you have used all the channels available.
Too many users	The system manager sets the maximum number of users for the file server.
Types don't match	**You have tried to save a file to a directory, or similar.**
User not known	The system manager creates users.

Message	*Explanation*
`Who are you?`	Reminder that *I AM is the only command for non-logged on users.
`Wrong password`	An attempt to log on with an incorrect password.

This table does not include errors common to level 1 and the DFS

Appendix M

USING THE DISCZAP PROGRAM

Before discussing the DISCZAP option in the ZAP program, a word of warning should be given. When writing sectors directly to a disc, it is very easy to corrupt an entire disc irretrievably. Before working on a disc, therefore, *it is essential that you make a backup copy of the disc*.

When the DISCZAP option is chosen from program ZAP (Section 10.5), the command P(age) is replaced by three other commands: Rt,s -read track t, sector s; Wt,s -write track t, sector s; and Vn -select drive n. The read and write commands must be issued in the form

```
R15,9 and W25,0
```

with track and sector numbers in *decimal* and separated by a comma because the numbers are collected by a two-variable INPUT command.

A sector is split into two 128-byte display pages, which can be examined using the cursor keys, with or without SHIFT. The program always begins by reading track 0, sector 0 − the first of the two catalogue sectors. Adjacent sectors can be read simply by scrolling the cursor off the top or bottom of the two pages of display. Larger jumps can be made with the R command. A sector can only be written back to disc with the W command, and the program checks first that you really do intend to write a sector.

The current track and sector is displayed at the top of the display, but you can write a sector back to a different place on the disc if you have a reason to do so. Initially the current drive remains in force, and only when you issue a command such as V1 is the drive changed and the value displayed at the top of the screen.

The remaining commands are the same as for MEMZAP.

HOW THE DISCZAP PROGRAM WORKS

Reading a sector from disc, and writing it back, is carried out by OSWORD calls &7F at lines 6490 and 6720. These involve calls to address &FFF1 with

&7F in the accumulator (via resident integer variable A%). X% and Y% contain low and high bytes of the address of a parameter block in memory -in this case the section of zero page starting at &70.

The parameter block is set up partly on line 160 and partly on lines 6450 or 6680. The parameters are, in order: the drive number; the address in memory for the source or destination of the sector (4 bytes, low byte first); 3; &53 for read or &4B for write; track; sector; &21; error number on return from the call (0 for a successful call).

The program uses page &7B of memory (immediately below the Mode 7 screen display area) for storing the sector and sets HIMEM to address &7B00, below this buffer area.

The R and W commands set the seventh byte of the parameter block to &53 or &4B respectively, track and sector are set via the variables TRACK and SECTOR, and then the call to &FFF1 is made, checking afterwards that there is no error from the call. If an error is detected the procedures try up to ten times before giving up.

The V command to change drive simply alters variable DRIVE which is used in the parameter list. Remaining commands work as they do for MEMZAP, moving the cursor or altering the specific page of memory being used as the disc buffer. Obviously the Page command is inappropriate for DISCZAP. When the cursor runs off either end of the page of memory representing the sector, this is trapped and the preceding or following sector is read from disc by a call to the 'read sector' subroutine.

THE STRUCTURE OF THE DFS CATALOGUE SECTORS

In order to make use of the DISCZAP utility, it is necessary to be able to find your way around the sectors of a disc, and the key to this is the two sectors containing the catalogue information. These are sectors 0 and 1 on track 0. Sector 0 is split up into 8 byte segments, and each segment except the first may contain the name of a file on the disc (hence the 31 file limit for the DFS). In corresponding positions in sector 1 is the catalogue information for each file. This is broadly similar to the information given by the *INFO command.

The precise use of the two catalogue sectors is as follows

Sector 0

bytes 0 to 7	First 8 characters of the disc title
bytes 8 to 14	Name of last file on disc (maximum 7 characters)
byte 15	File directory letter and lock status
bytes 16 to 22	Name of penultimate file
...	
...	
...	

Sector 1

bytes 0 to 3	Last 4 characters of disc title
byte 4	Number of times the disc has been accessed for writing
byte 5	8×number of entries in catalogue
byte 6 (high nibble)	Startup option set by *OPT 4,n
byte 6 (low nibble) and byte 7	Number of sectors on disc (&190 for 40 track discs, &320 for 80 track)
bytes 8 to 15	Information on the last file on the disc, as follows
bytes 8 and 9	Load address of file
bytes 10 and 11	Execution address of file (&801F or &8023 for BASIC programs)
bytes 12 and 13	File length in bytes
byte 14	High bits of each number
byte 15	Sector address of start of file (in hex)
bytes 16 to 23	Information on the penultimate file
...	
...	
...	

Three of the items require further explanation. The eighth byte of each file name contains the directory letter for the file in the first seven bits, while the high bit is set to indicate a file that is locked or zero if the file is not locked.

In sector 1 the eighth byte for each file contains the sector address of the file, that is the number of sectors from the start of the disc. To make use of this figure, it will be necessary to convert it into decimal and hence calculate the track and sector values before using DISCZAP.

The seventh byte contains spillover bits from the other four items. This is because the addresses and file length are stored as 18-bit numbers, and the sector address as a 10-bit number. The two high bits of each number are stored in byte seven as follows: bits 0 and 1 -sector address; bits 2 and 3 -load address; bits 4 and 5 -file length; bits 6 and 7 -execution address. These bits will usually be zero except for the sector address, so the byte will effectively act as the high byte of the sector address.

Figure M. 1, taken from DISCZAP, shows sectors 0 and I of the BBC utility disc, and many of the features mentioned above can be seen.

With an understanding of the way in which catalogue information is stored, you should be able to find your way around the sectors on a disc, and read or modify files direct on disc. Perhaps most valuable of all, used with care, is the editing of the catalogue sectors themselves. To give just one example, it is easy with the first version of *Wordwise* to wipe out a file by

'saving' it with nothing in memory when you intended to load it. All that will actually have happened on the disc is that the length will be marked as zero in the catalogue sectors, and the first sector may have been overwritten. The rest of the old file can be tracked down on the disc, and recovered by altering the length in the catalogue back to its full extent.

```
Sector 0
00 55 74 69 6C 69 74 79 00 UtiLity.
08 4D 45 4D 4F 44 20 20 A4 MEMOD  $
10 44 43 4F 4E 56 20 20 A4 DCONV  $
18 56 45 52 49 46 59 20 A4 VERIFY $
20 46 4F 52 4D 38 30 20 A4 F0RM80 $
28 46 4F 52 40 34 30 20 A4 F0RM40 $
30 21 42 20 20 20 20 20 24 !B     $
38 42 49 4F 52 54 48 4D 57 BIORTHMW
40 50 48 4F 54 4F 20 20 57 PHOTO  W
48 50 4F 45 4D 20 20 20 57 POEM   W
50 21 42 4F 4F 54 20 20 24 !BOOT  $

Sector 1

00 00 00 00 00 06 D8 31 90 .....X1.
08 00 00 00 00 F1 20 01 4C ....q .L
10 00 19 1F 80 C9 01 01 4A ....I..J
18 00 28 00 28 00 02 01 48 .(.(...H
20 00 28 04 28 00 03 01 45 .(.(...E
28 00 28 00 28 00 03 01 42 .(.(...B
30 00 19 1F 80 BA 02 01 3F ....:..?
38 00 19 1F 80 1D 19 01 25 .......%
40 00 19 00 19 00 10 01 15 ........
48 00 19 1F 80 B1 25 00 EF ....1%.o
50 00 30 00 30 0B 00 00 BB .0.0...;
```

Figure M.1 DISCZAP display for the BBC utility disc.

(The alternative disaster under Wordwise, obliterating the text in memory by 'loading' a nonexistent file when you meant to save it, can be recovered with MEMZAP, by tracking down the text which will still exist somewhere in memory, and saving it with *SAVE.)

Appendix N

ANSWERS TO EXERCISES

Exercise 2.1

```
10 INPUT "WHAT IS THE CAPITAL OF FRANCE" ,A$
15 M=0
20 IF A$="PARIS" THEN GOTO 60
25 M=M+1
27 IF M=2 THEN 80
30 PRINT "WRONG, TRY AGAIN"
40 INPUT A$
50 GOTO 20
60 PRINT "CORRECT"
70 END
80 PRINT "NO, THE ANSWER IS PARIS"
90 END
```

Exercise 2.2

```
10 REM SUM OF THE FIRST N WHOLE NUMBERS
20 INPUT "HOW MANY NUMBERS",N
30 SUM=0
40 FOR J%=1 TO N
50 SUM=SUM+J%
60 NEXT J%
70 PRINT "SUM OF THE FIRST ";N;" WHOLE NUMBERS =";SUM
80 END
```

Exercise 2.3

```
10 REM MASTERMIND
20 X=RND(10)
30 PRINT "WHAT NUMBER AM I THINKING OF?"' "(BETWEEN 1 AND 10)"
40 REPEAT
```

```
50 INPUT GUESS
60 UNTIL GUESS=X
70 PRINT "CORRECT"
```

Exercise 2.4

```
 10 REM GUESS THE NUMBER
 20 X=RND(1000)
 30 PRINT "WHAT NUMBER AM I THINKING OF?"' "(BETWEEN 1 AND 1000)"
 35 I=0
 40 REPEAT
 42 IF I=40 THEN PRINT "GIVE UP! THE ANSWER IS ";X: GOTO 80
 45 I=I+1
 50 INPUT GUESS
 52 IF GUESS>X THEN PRINT "TOO HIGH"
 54 IF GUESS<X THEN PRINT "TOO LOW"
 60 UNTIL GUESS=X
 70 PRINT "CORRECT"
 80 PRINT "YOU TOOK ";I;" GUESSES"
 90 IF I>10 THEN PRINT "YOU HAVEN'T DISCOVERED THE SYSTEM YET"
100 IF I<=10 THEN PRINT "CONGRATULATIONS"
110 END
```

Exercise 2.5

```
*KEY0 LISTO O |M |O |M
*KEY1 LISTO 7 |M |N LIST |M
```

Exercise 2.6

```
 10 REM PRODUCT OF 10 NUMBERS
 20 DIM NUM (10)
 30 FOR J=1 TO 10
 40 READ NUM(J)
 50 NEXT J
 60 PRODUCT=1
 70 FOR J=1 TO 10
 80 PRODUCT=PRODUCT*NUM(J)
 90 NEXT J
100 PRINT "PRODUCT = ";PRODUCT
110 END
120 DATA 2,5,3,7,1,4,6,9,10,8
```

Exercise 3.1

```
 10 REM HYPERBOLIC SIN AND COS
 20 REPEAT
 30 INPUT "VALUE OF X (USE 0 TO FINISH)",X
```

```
 40 C=FN_hcs(X)
 50 S=FN_hsn(X)
 60 Y=C*C-S*S: REM SHOULD EQUAL ONE
 70 PRINT "COSH(X)="C
 80 PRINT "SINH(X)="S
 90 PRINT "COSH SQUARED - SINH SQUARED ="Y
 95 PRINT
100 UNTIL X=0
105 END
110 DEF FN_hcs(X)=(EXP(X)+EXP(-X))/2
120 DEF FN_hsn(X)=(EXP(X)-EXP(-X))/2
```

Alternatively, we can print the results in columns

```
 10 REM HYPERBOLIC SIN AND COS
 15 @%=&20509: REM @% IS EXPLAINED IN FULL IN CHAPTER 11. THIS
    IS SETTING THE FORMAT TO 5 DECIMAL PLACES SO THAT THE
    OUTPUT IS IN NEAT COLUMNS
 18 PRINT: PRINT " X COSH SINH TEST=1?"
 19 PRINT
 20 FOR X=0 TO 2.01 STEP 0.1
 30 C=FN_hcs(X)
 40 S=FN_hsn(X)
 50 Y=C*C-S*S: REM SHOULD EQUAL ONE
 60 PRINT ;X,C,S,Y
 70 NEXT X
 80 END
 90 DEF FN_hcs(X)=(EXP(X)+EXP(-X))/2
100 DEF FN_hsn(X)=(EXP(X)-EXP(-X))/2
```

Exercise 3.2

```
 10 REM FAHRENHEIT/CENTIGRADE CONVERSION
 20 INPUT "GIVE A TEMPERATURE IN FAHRENHEIT",TE
 30 TC=FN_CENT(TF)
 40 PRINT "TEMPERATURE IN CENTIGRADE ="TC
 50 INPUT "GIVE A TEMPERATURE IN CENTIGRADE",TC
 60 TF=FN_FAHR(TC)
 70 PRINT "TEMPERATURE IN FAHRENHEIT="TF
 80 END
 90 DEF FN_CENT(T)
100 =(T-32)/1.8
110 DEF FN_FAHR(T)
120 =T*1.8+32
```

Exercise 3.3

Modify Example 3.3 as follows

```
 110 DEF PR0C_delay(CENTISEC)
 160 ENDPROC
 190 PROC_delay(RND(150))
```

Further modifications to eliminate the formal parameter are

```
110 DEF PROC_delay
150 UNTIL (100+START)<TIME
160 ENDPROC
190 PROC_delay
```

Exercise 3.4

A suitable procedure is

```
100 DEF PROC_LOWHIGH
110 REM finds the Largest and smaLLest numbers in array NOS of
    10 numbers
120 LOW=NOS(1): HIGH=NOS(1)
130 FOR J=2 TO 10
140 IF NOS(J)>HIGH THEN HIGH=NOS(J)
150 IF NOS(J)<LOW THEN LOW =NOS(J)
160 NEXT J
170 ENDPROC
```

which can be tested with the following program

```
 10 REM PROGRAM TO TEST PROC_LOWHIGH
 20 DIM NOS(10)
 25 PRINT "TYPE IN 10 NUMBERS, ONE PER LINE"
 30 FOR J=1 TO 10
 40 INPUT NOS(J)
 50 NEXT J
 60 PROC_LOWHIGH
 70 PRINT
 80 PRINT "LOWEST="LOW
 90 PRINT "HIGHEST="HIGH
 99 END
```

Exercise 3.5A

A suitable procedure is

```
100 DEF PROC_PRIME(RANGE)
110 PRIMES(1)=1
120 LOCAL N,X,Y,Z
130 N=1
140 Y=2
150 REPEAT
160 N=N+1
```

```
170 PRIMES(N)=Y
180 REPEAT
190 Y=Y+1
200 X=1
210 REPEAT
220 X=X+1
230 Z=INT(Y/X)
240 UNTIL (Z*X=Y OR X*X>Y)
250 UNTIL X*X>Y
260 UNTIL N=RANGE
270 ENDPROC
```

Exercise 3.5B

A suitable program to test the procedure is

```
10 INPUT "No. of prime numbers ",N
20 DIM PRIMES(N)
30 PROC_PRIME(N)
40 SUM=0
50 FOR J=1 TO N
60 SUM=SUM+PRIMES(J)
70 NEXT J
80 PRINT "Sum of first ";N;" primes = ";SUM
90 END
```

Exercise 3.6

```
 10 REM ON...GOTO DEMONSTRATION
 20 REPEA T
 30 INPUT "Give the time on a 24 hour cLock" ,ZTIME
 40 ON ZTIME G0TO50,50,50,50 ,50,50,50,50,70, 70,70,70,90,90,
    90,90,90,110,110,1 10,110,130,130,130 ELSE 150
 50 GS$="MORNING"
 60 GOTO 140
 70 GS$="DAY"
 80 GOTO 140
 90 GS$="AFTERNOON"
100 GOTO 140
110 GS$="EVENING"
120 GOTO 140
130 GS$="NIGHT"
140 PRINT "GOOD ";GS$
150 UNTIL TIME>24
160 PRINT "GOODBYE!"
```

Exercise 3.7

```
10 X=TIME
20 TS=INT(X/100)
```

```
30 TM=INT(TS/60)
40 TH=INT(TM/60)
50 TD=INT(TH/24)
60 PRINT "Time ";TD;" days ";TH;" hours ";TM MOD 60; " minutes.
   ";TS MOD 60;" second"
```

Exercise 3.8A

```
 10 REM PERPETUAL CALENDAR
 20 DIM MON$(12),DAY(12)
 30 MODE 0
 40 INPUT "For which year do you want a calendar ",YEAR
 50 IF YEAR<1800 THEN CLS: PRINT "Remember the year must be
    post 1800": GOTO 40
 60 FOR I=1 TO 12: READ MON$(I),DAY(I): NEXT
 70 DAY1=(((YEAR-1800)*365+(YEAR-1800) DIV 4)-(YEAR DIV 100 -
    YEAR DIV 400 -14)+3) MOD 7
 80 IF ((YEAR DIV 4)*4=YEAR) AND ((YEAR DIV 400)*400 <>YEAR)
    THEN DAY(2)=29: DAY1=(DAY1+6) MOD 7 ELSE DAY(2)=28
100 CLS
110 PRINT TAB(38,0);YEAR
120 FOR J=0 TO 3
125 I=J*3+1: K=J*8+1: L=I
130 PRINT ;TAB(11,K);MON$(I);TAB(36,K);MON$(I+1);TAB(61,K);
    MON$(I+2)
140 PRINT STRING$(3,"     Su Mo Tu We Th Fr Sa")
150 REPEAT
160 PROC_DATES(L,J,DAY1)
170 DAY1=(DAY1+DAY(L)) MOD 7
180 L=L+1
190 UNTIL L=I+3
200 NEXT
210 PRINT TAB(0,31);
400 END
500 DEF PROC_DATES(LD,JD,SDAY)
520 DATE=1
530 CDAY=SDAY
540 REPEAT
550 PRINT ;TAB(((LD-1) MOD 3)*25+(CDAY MOD 7)*3+5,JD*8+(CDAY
    DIV 7) MOD 5 +3);DATE;
560 DATE=DATE+1
570 CDAY=CDAY+1
580 UNTIL DATE-1=DAY(LD)
600 ENDPROC
900 DATA January,31,February,28,March,31,April,30,May,31,
    June,30,July ,31,August,31,September,30,October,31,
    November,30,December,31
```

Exercise 3.8B

This modified solution uses an array to produce the screen display. This modification allows the calendar to be sent to a printer, if after use of the appropriate printer calls the section of program at the end is implemented by the use of GOTO 2000 in immediate mode.

```
 10 REM PERPETUAL CALENDAR
 20 DIM MON$(12),DAY(12),SCRPO$(20,19)
 25 FOR I=0 TO 19: FOR J=0 TO 20: SCRPO$(J,I)=" ": NEXT: NEXT
 30 MODE 0
 40 INPUT "For which year do you want a calendar ",YEAR
 50 IF YEAR<1800 THEN CLS: PRINT "Remember the year must be
    post 1800": GOTO 40
 60 FOR I=1 TO 12: READ MON$(I) ,DAY(I): NEXT
 70 DAY1=(((YEAR-1800)*365+(YEAR-1800) DIV 4)-(YEAR DIV 100 -
    YEAR DIV 400 -14)+3) MOD 7
 80 IF ((YEAR DIV 4)*4=YEAR) AND ((YEAR DIV 400)*400 <>YEAR)
    THEN DAY(2)=29: DAY1=(DAY1+6) MOD 7 ELSE DAY(2)=28
100 CLS
110 PRINT TAB(38,0);YEAR
120 FOR J =0 TO 3
125 I=J*3+1: K=J*8+1: L=I
130 PRINT ;TAB(11,K);MON$(I); TAB(36,K);MON$(I+1);TAB(61,K);
    MON$(I+2)
140 PRINT STRING$(3,"     Su Mo Tu We Th Fr Sa")
150 REPEAT
160 PROC_DATES(L,J,DAY1)
170 DAY1=(DAY1+DAY(L)) MOD 7
180 L=L+1
190 UNTIL L=I+3
200 NEXT
400 END
500 DEF PROC_DATES(LD,JD,SDAY)
520 DATE= 1
530 CDAY=SDAY
540 REPEAT
545 X=((LD-1) MOD 3)*7+(CDAY MOD 7): Y=JD*5+(CDAY DIV 7) MOD 5
547 IF DATE<10 THEN SCRPO$(X,Y)=" "+STR$(DATE) ELSE
    SCRPO$(X,Y)=STR$(DATE)
550 PRINT ;TAB(((LD-1) MOD 3)*25+(CDAY MOD 7)*3+5,JD*8+(CDAY
    DIV 7) MOD 5 +3);DATE;
560 DATE=DATE+1
570 CDAY=CDAY+1
580 UNTIL DATE-1=DAY(LD)
600 ENDPROC
900 DATA January,31,February,28,March,31,April,30,May,31,
    June,30,July,31,August,31,September,30,October,31,
```

```
    November,30,December,31
2000 PRINT ;TAB(38);YEAR: PRINT
2010 FOR K=0 TO 3: M=K*3+1
2020 PRINT;TAB(11);MON$(M);TAB(36);MON$(M+1);TAB(61);MON$(M+2)
2030 PRINT STRING$(3,"     Su Ma Tu We Th Fr Sa")
2040 FOR I=0 TO 4: I=K*5+L
2050 FOR J=0 TO 20
2060 IF (J MOD 7)=0 THEN PRINT ;"    ";
2070 PRINT ;" ";SCRPO$(J,I);
2080 NEXT
2090 PRINT
2100 NEXT
2110 PRINT
2120 NEXT
```

Exercise 3.9A

The simple solution is to modify Example 3.11 as shown below

```
 10 DIM N(6),M(3)
 20 FOR L = 1 TO 20
 30 FOR I = 1 TO 6: N(I)=I: NEXT
 40 PROC perm (3 )
 50 PRINT ;L;"   ";M(1);"   ";M(2);"   ";M(3)
 60 NEXT L
 70 END
 80 DEF PROC_perm(I)
 90 IF I<>1 THEN J=RND(I+3): M(I)=N(J): N(J)=N(I+3):
    PROC_perm(I-1) ELSE M(I)-N(RND(I+3))
100 ENDPROC
```

Exercise 3.9B

A more widely applicable solution is the following program, that can be used to select a random set of N numbers from a given range R.

```
  5 INPUT "Combination of N from R give N and R ",N,R
 10 DIM N(R),M(N)
 20 FOR L = 1 TO 20
 30 FOR I = 1 TO R: N(I)=I: NEXT
 40 PROC_perm(N)
 50 PRINT ;L;
 55 FOR J=1 TO N: PRINT" ";M(J);: NEXT
 57 PRINT
 60 NEXT L
 70 END
 80 DEF PROC_perm(I)
```

```
 90 IF I<>1 THEN J=RND(I+R-N): M(I)=N(J): N(J)=N(I+R-N):
    PROCperm(I-1) ELSE M(I)=N(RND(I+R-N))
100 ENDPROC
```

Exercise 4.6

```
10 PRINT "INPUT ANY STRING"
20 INPUT A$
30 L=LEN(A$)
40 B$=LEFT$(A$,INT(L/2))
50 C$=RIGHT$(A$,L-INT(L/2))
60 PRINT B$+ " EXTRA " +C$
70 REM OR PRINT B$; " EXTRA " ;C$ WOULD DO
```

Exercise 4.7

```
10 PRINT "INPUT ANY STRING"
20 INPUT A$
30 FOR J=LEN(A$) TO 1 STEP -1
40 PRINT MID$(A$,J,1);
50 NEXT J
60 PRINT
```

Exercise 4.9

Change line 130 of Exercise 4.8 to

```
130 UNTIL J=0 OR (LEN(A$)-LEN(B$))<(L-1)
```

Exercise 4.11

```
 10 PRINT "INPUT A NUMBER IN FIXED POINT OR EXPONENT FORM"
 20 INPUT A$
 30 L=LEN(A$): D=0
 40 FOR J = 1 TO L
 50 IF MID$(A$,J,1)="E" THEN GOTO 110
 60 IF MID$(A$,J,1)="." THEN D=J
 70 NEXT J
 80 B$=RIGHT$(A$,L+1-1))
 90 B=VAL(B$)
100 IF D>0 THEN PRINT 'B
110 PRINT '"END"
```

Exercise 4.13

CHR$(7) produces a bleep since this is the BELL character
CHR$(8) moves the cursor to the left one space
CHR$(10) moves the cursor down one line
CHR$(12) clears the screen and homes the cursor to top left

Exercise 4.14

The following program carries out all the tasks set in Exercise 4. 14

```
 10 PRINT "INPUT ANY CHARACTER"
 20 INPUT A$
 30 PRINT "ASCII CODE OF ";A$;" IS ";ASC(A$)
 40 PRINT
 50 PRINT "INPUT ANY ASCII CODE (33 TO 255)"
 60 INPUT A
 70 PRINT "THE CHARACTER ";CHR$(A);" HAS ASCII CODE ";A
 80 PRINT
 90 TIME=0: REPEAT: UNTIL TIME>=300
100 FOR J=33 TO 255
110 PRINT J;"  ";CHR$(J)
120 TIME=0: REPEAT: UNTIL TIME>=20
130 NEXT J
```

Exercise 4.15

Modify Exercise 4.7 by

```
25 DIM A$(LEN(A$))
30 FOR J=1 TO LEN (A$)
40 A$(J)=MID$(A$,J,1)
60 FOR J=LEN(A$) TO 1 STEP -1: PRINT A$(J);: NEXT J
70 PRINT
```

Exercise 5 .2

```
10 PRINT "TYPE YOUR EXPRESSION AS A FUNCTION"
20 INPUT "OF X: ",EX$
30 PRINT "TYPE UPPER AND LOWER LIMITS"
40 INPUT "OF X: ",A,B
50 FOR X=A TO B STEP (B-A)/10
60 PRINT X;"  ";EVAL(EX$)
70 NEXT X
```

Exercise 5.3

```
10 INPUT "TYPE THE AMOUNT IN PENCE: "AMT%
20 PDS%=AMT% DIV 100
30 PENCE%=AMT% MOD 100
40 PRINT "AMOUNT IS £";PDS%;" ";PENCE%;"p"
```

Exercise 5.5

Line 170 can be rewritten as

```
170 SCORE(PLAYER)=SCORE(PLAYER)-10*(ACE(PLAYER)>0 AND
    SCORE(PLAYER)<=11)
```

with a similar line for 260.

Exercise 6.5

In Example 6.4, change line 70 to

```
70 VDU 29,640;608;
```

and delete (or replace by zero) X0 and YO from lines 80, 90, 130 and 170. Proceed similarly for the other programs.

Exercise 6.6

Change line 30 to, say

```
30 VDU 28,3,9,16,2
```

Exercise 6.8

A greek phi can be created by

```
VDU 23,225,8,8,28,42,28,8,8,0
```

Exercise 6.9

The following VDU statements define, in order, an open and closed square, an open and close triangle and an open and closed circle

```
VDU 23,230,248,136,136,136,248,0,0,0
VDU 23,231,248,248,248,248,248,0,0,0
VDU 23,232,32,80,80,136,248,0,0,0
VDU 23,233,32,112,112,248,248,0,0,0
VDU 23,234,112,136,136,136,112,0,0,0
VDU 23,235,112,248,248,248,112,0,0,0
```

Exercise 7.1

The program contains the following errors

line 20	should be DIM V(3)
line 40	should not be a REM
line 200	should not contain a minus sign
line 220	closing bracket omitted
line 230	should be T, not T$
line 230	missing colon
line 390	must return to line 30, not 20

Exercise 9.3

```
10 INPUT "TYPE THE NUMBER OF CARS: ",N
20 CH%=OPENOUT ("CARFILE")
30 FOR J = 1 TO N
```

```
40 PRINT "TYPE THE NAME OF THE CAR"
50 INPUT CAR$
60 PRINT#CH%,CAR$
70 NEXT J
80 CLOSE#CH%
```

Exercise 9.4

```
10 PRINT "LIST OF CARS": PRINT
20 CH%=OPENIN ("CARFILE")
30 REPEAT
40 INPUT#CH%,CAR$
50 PRINT CAR$
60 UNTIL EOF#CH%
70 CLOSE#CH%
```

Exercise 9.5

Modify Example 9.5 by the following addition

```
65 IF LEN(ITNM$)+LEN(TPE$)+9>32 THEN PRINT "YOUR RECORD IS TOO
   LONG: RE-INPUT": GOTO 40
```

Exercise 9.6

Example program 9.7 requires the following addition

```
1015 IF RECNO*32>EXT#CH% THEN PRINT "THAT RECORD NUMBER DOES
     NOT EXIST": GOTO 1080
```

Exercise 9.7

Modify Example 9.7 further, as follows

```
1075 PROC_modify(RECNO)
3000 DEF PROC_modify(RECNO)
3010 INPUT "DO YOU WANT TO CHANGE THE QUANTITY (Y/N)",YN$
3020 IF LEFT$(YN$,1)="N" THEN ENDPROC
3030 INPUT "TYPE THE NEW QUANTITY: "QU%
3040 PTR#CH%=RECNO*32
3050 PRINT#CH%,ITNM$,TPE$,QU%
3060 ENDPROC
```

Exercise 9.8

```
10 CH%=OPENIN( "INVENT")
20 C=0
30 REPEA T
40 C=C+1
```

```
50 PRINT ;'BGET#CH%;" ";
60 IF C=8*INT(C/8) THEN PRINT
70 UNTIL EOF#CH%
80 CLOSE#CH%
```

Exercise 9.9

Modify the program from Exercise 9.3 as follows

```
52 L%=LEN(CAR$)
54 IF L%>10 THEN PRINT "NOT MORE THAN 10 CHARACTERS" :GOTO 50
56 CAR$=CAR$+STRING$(10-L%," ")
```

Exercise 9.10

```
10 CH%=OPENIN( "CARFILE")
20 PRINT "WHICH CAR DO YOU REQUIRE (1,2,3,...)"
30 INPUT N
40 PTR#CH%=(N-1)*12
50 INPUT#CH%,CAR$
60 PRINT "CAR ";N;" IS ";CAR$
70 CLOSE#CH%
```

Exercise 9.11

The ASCII file should have the form

```
*KEY 0 RUN |M
*KEY 1 LIST |M
*KEY 2 AUTO |M
*KEY 3 RENUMBER |M
*KEY 4 *RENAME
*KEY 5 *DELETE
```

Exercise 9.13

Delete lines 10 to 310 of Example 4.10, and add or modify the remaining lines as follows

```
300 DEF PROC_textprint(L)
310 LOCAL J,K,M
385 M=0
390 REPEAT
395 M=M+1
405 IF T$(M+K)=CHR$(13) THEN J=M
410 UNTIL M>=J-1
460 ENDPROC
```

Then renumber the resulting procedure to start at 2000 instead of 300. Append the procedure to Example 9.10, and make further changes to the program, as follows

```
525 DIM T$(3500)
700
760 L=L+1
770 T$(L)=CH$
780 UNTIL EOF#CH% OR L>=3500
784 IF (OP% AND 4)=4 THEN VDU 2
786 PROC_textprint(L)
```

A minor problem with this program is that, because it now stores the text as single characters in the string array T$, it is very wasteful of memory. Much larger files could be read if the data was poked directly into memory with the indirection operator instead of into an array. This can be achieved by the following further changes to the program above. As a further refinement, a choice of page widths is offered, to accommodate output to a printer, or use in Mode 0 or Mode 3.

```
525 BA=&2500
682 PRINT "What page width do you require?"
684 INPUT PGW
770 ?(BA+L)=ASC(CH$)
780 UNTIL EOF#CH% OR L>&5000: REM REDUCE VALUE OF &5000 IF
    NOT IN MODE 7
782 ?(BA+L+1)=0
784 IF (OP% AND 4)=4 THEN VDU 2
786 PROC_textprint(L,BA)
2000 DEF PROC textprint(L,BA)
2010 LOCAL J,K,M,CH$
2040 J = PGW+2
2060 J = J-1: CH$=CHR$(?(BA+J+K))
2070 UNTIL CH$ = " " OR CH$ = CHR$(0) OR (CH$ = "-" AND
     J<PGW+1) OR J=0
2080 IF J = 0 THEN J = PGW: REM WORD TOO LONG TO FIT
2115 CH$=CHR$(?(BA+M+K))
2120 PRINT CH$ ;
2130 IF CH$=CHR$(13) THEN J=M
2145 CH$=CHR$(?(BA+J+K))
2150 IF CH$ <> " " THEN PRINT CH$;
2160 IF J<PGW OR CHR$(?(BA+PGW+K)) = " " OR CHR$(?(BA+PGW+K))
     = "" THEN PRINT
```

Exercise 10.1

```
 10 FOR J=7 TO 0 STEP -1
 20 MODE J
 30 PRINT "PAGE = ";PAGE
 40 PRINT "TOP = ";TOP
 50 PRINT "HIMEM = ";HIMEM
 60 PRINT "LOMEM = ";LOMEM
```

```
 70 PRINT "PROGRAM SIZE = ";(TOP-PAGE)
 80 PRINT "FREE SPACE = ";(HIMEM-TOP)
 90 PRINT "PRESS ANY KEY TO CONTINUE"
100 A$=GET$
110 NEXT J
```

Exercise 10.2

PAGE should have the value &2300.

Exercise 10.6

$M=A$ means 'load the string stored in variable A$ into memory starting from address M'.
M$=$A means 'set variable M$ equal to the string (hopefully) stored in memory starting from address A'.

```
10 PRINT "TYPE THE STARTING ADDRESS FOR STRING"
20 INPUT ADDR$
25 ADDR=EVAL(ADDR$)
30 PRINT "TYPE THE STRING"
40 INPUT A$
50 FOR J=0 TO LEN(A$)-1
60 ADDR?J=ASC(MID$(A$,J+1,1))
70 NEXT J
75 J=LEN(A$)
80 ADDR?J=13
90 PRINT "STRING WAS: ";$ADDR
```

Exercise 10.17

If the line P%=start% is not inside the FOR...NEXT loop, the second assembly will start at the end of the first assembly. Note the different assembly addresses for the two passes.

Exercise 11.1

The program from Exercise 4.14 can be modified as follows

```
20 A$=GET$
```

Exercise 11.2

```
 10 MODE 6
 20 VDU 23,1,0;0;0;0;
 30 *FX 4 , 1
 40 ST$="*"
 50 CH=20: CV=12
 60 PRINT TAB(CH,CV);ST$;
 70 REPEAT
```

```
 80 A=ASC(GET$)-135
 90 PRINT TAB(CH,CV);" ";
100 IF A>0 AND A<3 THEN CH=CH+2*A-3
110 IF A>2 AND A<5 THEN CV=CV-2*A+7
120 IF CV>24 THEN CV=24
130 IF CV<0 THEN CV=0
140 IF CH>39 THEN CH=39
150 IF CH<0 THEN CH=0
160 PRINT TAB(CH,CV);ST$;
170 UNTIL A=0
```

Exercise 11.3

Modify the program from Example 11.1 as follows

```
 30 PRINT "PRESS THE KEY SPECIFIED"
105 K=64+RND(26)
120 PRINT TAB(15,12);"PRESS THE KEY: ";CHR$(K)
160 UNTIL INKEY(0)=K
```

Exercise 11.4

Modify the program from Example 11.2 as follows

```
  70 IF INKEY(-99) THEN PR0C_wait
1030 UNTIL NOT INKEY(-99)
```

Exercise 11 .8

A suitable procedure is

```
1010 DEF PROC_table
1020 LOCAL J,K
1030 FOR J=1 TO 5
1040 PRINT TAB(7*J-7);HEAD$(J);
1050 NEXT J
1060 PRINT ''
1070 FOR K=1 TO 10
1080 FOR J=1 TO 5
1090 PRINT TAB(7*J-7);A$(K,J);
1100 NEXT J
1110 PRINT
1120 NEXT K
1130 ENDPROC
```

Exercise 11.9

The following procedure will provide the required display

```
1000 DEF PROC_clock(HOUR1,SEC1,HOUR2,SEC2)
1010 PRINT TAB(0,7);SPC(16);
1020 PROC_display(HOUR1,MIN1,SEC1)
1030 PRINT SPC(16)
1040 PRINT TAB(0,15) ;SPC(16);
1050 PROC_display(HOUR2,MIN2,SEC2)
1060 PRINT SPC(16)
1070 ENDPROC
2000 DEF PROC_display(HOUR,MIN,SEC)
2010 LOCAL HOUR$,MIN$,SEC$
2020 HOUR$=STR$(H0UR): IF HOUR<10 THEN HOUR$="0"+H0UR$
2030 MIN$=STR$(MIN): IF MIN<10 THEN MIN$="0"+MIN$
2040 SEC$=STR$(SEC): IF SEC<10 THEN SEC$="0"+SEC$
2050 PRINT HOUR$; ":";MIN$;":";SEC$;
2060 ENDPROC
```

Exercise 11.10

The following program should be used in conjunction with PROC_formprint from Example 11.8.

```
  5 MODE 0
 10 FOR J = 1 TO 20
 20 I=RND(1000)
 30 R1=1E5*RND(1)
 40 R2=1E5*RND(1)
 50 D1=RND(1)
 60 D2=RND(1)
 70 PROC_formprint(I,0,5)
 80 PROC_formprint(R1,2,10)
 90 PROC_formprint(R2,2,10)
100 PROC_formprint(D1,3,7)
110 PROC_formprint(D2,3,7)
120 PRINT
130 NEXT J
140 END
```

Exercise 11.12

The ASCII file should contain the following lines

```
*CAT
NEW
10 PRINT
20 INPUT "WHICH PROGRAM DO YOU WANT TO RUN",P$
30 CHA I N P$
RUN
```

Exercise 11.13

```
 10 REPEAT
 15 CLS
 20 PRINT ''
 30 PRINT "WHICH MATHEMATICAL OPERATION WOULD YOU LIKE?"
 50 PRINT '''1. RECIPROCAL OF N"
 60 PRINT '"2. 10 TO POWER N"
 70 PRINT '"3. FACTORIAL OF N"
 80 PRINT '"4. PI TO N DECIMAL POINTS"
 90 PRINT '"5. N RANDOM NUMBERS"
100 PRINT '"6. END"
120 PRINT ''''TYPE THE NUMBER OF YOUR CHOICE"
125 REPEAT
130 A=VAL(GET$)
135 UNTIL A>0 AND A<7
140 CLS
150 IF A=6 THEN END
160 PRINT "TYPE N: ";
170 N=VAL(GET$)
180 ON A GOSUB 200,300,400,500,600 ELSE END
190 PRINT '''PRESS ANY KEY TO CONTINUE" :C$=GET$
195 UNTIL FALSE
200 PRINT '"RECIPROCAL OF ";N;" IS ";1/N
210 RETURN
300 PRINT '"10 ^ ";N;" IS ";10^N
310 RETURN
400 PROD=1
410 FOR J=1 TO N
420 PROD=PROD*J
430 NEXT J
440 PRINT '"FACTORIAL ";N;" IS ";PROD
450 RETURN
500 PW=1
510 FOR J=1 TO N: PW=PW*10: NEXT J
520 P=INT(PI*PW)/PW
530 PRINT '"PI TO ";N;" DECIMAL POINTS IS ";P
540 RETURN
600 PRINT 'N;" RANDOM NUMBERS:"'
610 FOR J=1 TO N: PRINT RND(1): NEXT J
620 RETURN
```

Appendix O

THE SUPPLEMENTARY DISC AND TAPE

A supplementary disc or tape is available to accompany this book. Each contains all the example programs and the longer answer programs. Example programs are stored on the tape as the example number with a prefix EX, and answer programs prefixed by A, so Example 1.1 would be EX1.1 and the solution to Exercise 1.1 would be A1.1.

On the disc, because of the limitation of 31 files in the disc directory, programs are stored in sets, by stacking them up in memory and then setting PAGE to the particular program wanted. The example and answer programs are therefore accessed via a two-level menu system, initiated by pressing SHIFT-BREAK. All programs, once accessed, can be freely SAVEd to a new disc, or the whole disc can be copied with *BACKUP.

Purchasers of either disc or tape are entitled to make one working copy either of individual programs or entire tape or disc.

The supplementary disc and tape are distributed by Lostock Software, and each is priced at £6.95 including postage and packing. Please write, enclosing cheque or postal order and specifying whether you require tape, 40 track disc or 80 track disc, to

Lostock Software
13 Cranborne Close
Lostock
Bolton
Lancs BL6 4JG

Please apply directly to Lostock Software, not to Addison-Wesley Publishers Ltd.