INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION

INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
34100 TRIESTE (ITALY) - P.O.B. 586 - MIRAMARE - STRADA COSTIERA 11 - TELEPHONE: 2240-1
CABLE: CENTRATOM - TELEX 460392 - I

SECOND SCHOOL ON ADVANCED TECHNIQUES
IN COMPUTATIONAL PHYSICS
(18 January - 12 February 1988)

SMR.282/2

USING DOS 3.X

C. CURRAN
~~BEN SEGAL~~
CERN, DD Division, Geneva, Switzerland

CERN

DOS 3.x FOR PCs

CERN Operations Group/CO

21 December 1987

USING DOS 3.X

DD/CO

CONTENTS

1. INTRODUCTION 1

2. STARTING DOS 3.X ON YOUR MACHINE 2
- 2.1 Updating a running system to DOS 3.x 2
- 2.2 Starting up a new machine with DOS 3.x 2
- 2.3 CONFIG.SYS file 3
- 2.4 AUTOEXEC.BAT file and other .BAT command files 4
- 2.5 Device names in DOS 3.x 6
- 2.6 Special control combinations and DOS editing keys 6
- 2.7 DOS 3.x directory structure, PATH and filenames 6
- 2.8 Redirecting standard input, output, piping, filters. 7

3. DOS 3.X SUMMARY OF COMMANDS 9
- 3.1 ASSIGN 9
- 3.2 ATTRIB 9
- 3.3 BACKUP 9
- 3.4 Batch file commands 10
- 3.5 BREAK (control – break) 10
- 3.6 CD or CHDIR 10
- 3.7 CHKDSK 11
- 3.8 CLS (clear screen) 11
- 3.9 COMMAND 11
- 3.10 COMP 11
- 3.11 COPY 11
- 3.12 CTTY 12
- 3.13 DATE 12
- 3.14 DEL (delete file) 12
- 3.15 DIR (directory) 13
- 3.16 DISKCOMP (compare diskettes) 13
- 3.17 DISKCOPY 13
- 3.18 ECHO (batch subcommand) 13
- 3.19 ERASE 13
- 3.20 EXE2BIN 13
- 3.21 FDISK 14
- 3.22 FIND filter 14
- 3.23 FOR (batch subcommand) 14
- 3.24 FORMAT 14
- 3.25 GOTO (batch subcommand) 14
- 3.26 GRAFTABL (load graphics table) 15
- 3.27 GRAPHICS (screen print) 15
- 3.28 IF, IF EXIST, IF ERRORLEVEL (batch subcommand) 15
- 3.29 JOIN (join directories) 16
- 3.30 KEYBxx (load keyboard) 16
- 3.31 LABEL (volume label) 16
- 3.32 LINK (utility LINKer) 16
- 3.33 MD or MKDIR (make directory) 17
- 3.34 MODE 17
- 3.35 MORE filter 18
- 3.36 PATH (set search directory) 18
- 3.37 PAUSE (batch subcommand) 18
- 3.38 PRINT 18
- 3.39 PROMPT (set system prompt) 18
- 3.40 RECOVER 19

3.41 REM (batch subcommand) 19
3.42 RENAME or REN 19
3.43 RESTORE 19
3.44 RD or RMDIR (remove directory) 19
3.45 SELECT 20
3.46 SET (set environment) 20
3.47 SHARE 20
3.48 SHIFT (batch subcommand) 20
3.49 SORT filter 20
3.50 SUBST (substitute) 21
3.51 SYS (system) 21
3.52 TIME (set machine time) 21
3.53 TREE (sho all directories) 21
3.54 TYPE (list file contents) 21
3.55 VER (version) 21
3.56 VERIFY 22
3.57 VOL (Volume) 22

4. SOME UTILITIES WHICH ARE NOT INCLUDED IN DOS 3.X 23
4.1 BASCOM compiler (IBM/Microsoft) 23
4.2 Professional FORTRAN compiler (IBM/Ryan McFarland Corp) 24
4.3 LIB library maintenance utility 25

5. EDITING FILES 26
5.1 EDLIN line editor 26
5.2 BASICA as an editor adapted to BASIC 27

6. BASICA COMMANDS 28

7. COMMUNICATING WITH SERIAL DEVICES 32

8. PC NETWORKING 34

9. ASCII CHARACTER SET USED BY PCS 39

INDEX 41

1. INTRODUCTION

DOS, the Disk Operating System, is what most people use to run programs, games, and so on on IBM PC machines and 'look – alikes'. DOS is really very simple and easy to use. It is designed to be used by one person on his own machine, not a large group on some centralised processor. It cannot be **modified** by the normal user, which greatly simplifies things. It is now the most widespread operating system in the world. There is a good probability that anyone you meet who uses 'computers' has used (or still uses) DOS, so there is a great deal of experience and help to be found.

Those accustomed to UNIX systems will think DOS to be too simple, and lacking all the clever features they have grown to love and possibly even use. It does, however, offer 'redirection' of standard input and output, 'piping', and a UNIX type file structure. Users who are prepared to pay the price will will find interpreters/compilers for FORTRAN, PASCAL, C etc. available. Networking of various types is also available, again at a price!

A more serious limitation to DOS users has been the type of machines they had to use. The IBM PC – 1, XT and AT are all basically 16 – bit machines with a small memory size: up to 640K bytes. Much 'scientific' programming expects far more memory than this. Equally, without a 'co – processor' to speed floating point arithmetic operations, the computational speed of such machines is very limited.

Nevertheless, the steady drop in price of IBM machines and competition from look – alikes has made an AT – type machine with a co – processor and 20 Mbyte disk a practical 'personal' machine. With only one user, it is possible to envisage performing some scientific processing, using FORTRAN etc. The newer IBM PS2 is a 32 – bit machine, and this change should remove the memory size limitations. Even the cost of networking PCs (or PS2) machines is dropping fast, with connections to APPLETALK, Ethernet, DEC NET, IBM PC NET, IBM Token Ring, etc. costing from about 100$ (US) to 900$. The French Minitel system can be used to connect PCs to many different systems with an adapter costing less than 100$. However, you will still be using (most probably) DOS 3.x until IBM's next offering appears (later this year?), so here is an introduction to it.

In case you need to know what changed radically between previous DOS versions, DOS 1.0 was the first version. DOS 2.x started support of fixed disks, DOS 3.0/3.1 offered support of PC NET, and DOS 3.2 support for IBM Token Ring.

2. STARTING DOS 3.X ON YOUR MACHINE

2.1 Updating a running system to DOS 3.x

Switch off your machine, put the DOS 3.x diskette in the diskette drive (or a copy of it) and restart your machine. The system will eventually prompt you with:

```
A>
```

Enter the following commands to transfer all the new DOS 3.x files to your fixed disk, C, and to update your system:

```
A>SYS C:
A>COPY A:*.* C:*.*
```

2.2 Starting up a new machine with DOS 3.x

Put the diskette containing DOS 3.x (or a copy) in the diskette drive, A. Close the door and switch on your machine. The system will eventually prompt you with:

```
A>
```

The first utility you need to run from diskette A is **FDISK** designed to **partition** your fixed disk, C. Reply to the prompts in order to create a DOS partition, using the entire fixed disk. Next, you need to **FORMAT** the entire disk. This takes several minutes, during which the disk will be active. Enter the command:

```
FORMAT C:/S/V
```

FORMAT will prompt for a volume label (1 – 11 chars) for your disk. I suggest something related to your name, or to the planned function of the machine. You can change this label later with the **LABEL** command of DOS 3.x. Next, you should copy all the DOS 3.x files from the diskette to the hard disk, delete the FDISK command (this can destroy all data on your disk if used at a later time), and set the default drive to C:

```
A>COPY A:*.* C:
A>DEL C:FDISK.*
A>C:
C>
```

Your machine can now be restarted and will loading the system routines required from your newly labelled hard disk. Put the DOS 3.x diskette away in a safe place. Now, you should make a **CONFIG.SYS** and an **AUTOEXEC.BAT** file suitable for your hardware and for what you want to do with the machine.

Several partitions...for MINIX, perhaps..

Normally you will have only one partition, using all the space on your disk for DOS 3.x but other arrangements are quite possible. However, if you are using them, you should beware that FDISK can only create a single DOS partition. Thus, if you have some other partition already defined, but not occupying the whole disk, FDISK will allow the creation of a DOS partition occupying some (or all) of the remaining space. Should you already have a DOS partition on your disk, whether occupying the whole disk or not, FDISK will do nothing for you. Therefore, to create a MINIX partition:

- BACKUP your entire disk, delete all files and use FDISK to delete the single, DOS, partition.
- Run FDISK to create a new DOS partition. This should be large enough to be useful, but leave enough space for another DOS partition to RESTORE all your BACKUP files to. of the disk.
- Run DISKFIX (non – standard) to flag this new partition as 'non – DOS'
- Run FDISK again. This detects the 'non – DOS' partition, and will allow you now to create a DOS partition in the free cylinders. FDISK allows you to choose the active partition, and you can now restore the BACKUP files and run DOS again. In principal, you can use the 'non – DOS' partition to run MINIX, etc. I admit that I did not succeed!

2.3 CONFIG.SYS file

The CONFIG.SYS file is the first thing consulted by your operating system, before you have any chance to do anything. It declares such things as devices that you might be using, the style of date you like to use (USA or English), parameters describing your PC Net use (if any) etc **Note that information read from here is not refreshed until you restart your machine (power off – on or use Ctrl – Alt – Del).** You can create a CONFIG.SYS file with EDLIN (See later) or by typing one in as below, using a COPY from CON (the keyboard):

Simple example

```
C>COPY CON CONFIG.SYS
FILES=20
BUFFERS=20
FCBS=16,8
BREAK=ON
COUNTRY=044
DEVICE=ANSI.SYS
F6 Enter (the F6 function key, then Enter, closes input)
```

The FILES, BUFFERS and FCBS statements are used here (and in almost all machines using networking software) to support remote file access. We allow 20 FILES to be open at once (8 – 255, default is 8) and ask for 20 file BUFFERS also (1 – 99, default 2, each using 128 bytes of memory). We set BREAK to be ON. Normally it is OFF, allowing DOS to check for a user interrupt (Ctrl – Break key sequence) only during standard I/O, print or auxiliary operations. This can take some time to happen, so at first I suggest setting it ON. The date style will be dd/mm/yyyy, as the COUNTRY code of 044 (International trunk dialing code!) identifies the UK (default is 001, USA). DEVICE = ANSI.SYS allows you to use extended keyboard functions.

An example for a PC using TCP/IP over Ethernet

```
FILES = 20
BUFFERS = 20
FCBS = 16,8
LASTDRIVE = E
BREAK = ON
COUNTRY = 044
DEVICE = ANSI.SYS
DEVICE = C:\TCP\3C500.SYS
DEVICE = C:\TCP\IPCUST.SYS
```

In this example of a more complex CONFIG.SYS file, we show the one used in one of our machines which is normally running TCP/IP software, allowing other machines on our Ethernet to be accessed. The FILES, BUFFERS and FCBS statements are used here (and in almost all machines using networking software) to support remote file access. The LASTDRIVE command specifies the maximum number of drives we may use (default is E, allowing 6 drives). In a network machine we would often have A,B and C on our own machine already and declare D as the first drive we want to access on some other machine, E as the second etc. Two DEVICE definitions are used, which load software into memory for use later on (3C500.SYS and IPCUST.SYS here). If you are using the Token Ring, you will need the line

```
STACKS = 64,128
```

Two other commands exist for the CONFIG.SYS file. These are VDISK which defines a virtual disk resident in memory. Naturally you need a lot of memory for this to work, but it can greatly speed up library searches, etc. VDISK can also be very useful for speeding up execution of .BAT files (which contain commands such as YNPROMPT, MODE, etc). Such files are usually very small so it is possible to define a 64K byte virtual disk, copy the commands you need to it (and the .BAT files) and then make it the default drive. NO disk or diskette accesses are then needed until the batch file calls for something not on the virtual disk. Thus:

```
DEVICE = C:VDISK.SYS 64 128 50
```

installs VDISK when it is encountered in your CONFIG.SYS. The size is 64 Kbytes (small, but useful) in sectors of 128 bytes (default) and up to 50 directory entries. VDISK assigns the 'drive letter' it**self, using the first free letter.** Thus in a double diskette machine (A: and B:) it assigns C: to itself, and in a single fixed disk machine (A:, B: and C:) it assigns D:. Beware that you do not try to reuse this letter when your AUTOEXEC.BAT commands are issued. If you have an AT type machine, which can support memory extensions above 640K usable by VDISK amongst other things, you may also find this very useful. Finally there is SHELL, which allows you to run a top−level command processor of your own. I have never used it.

2.4 AUTOEXEC.BAT file and other .BAT command files

DOS 3.x will assume that it will find DOS commands in any file whose name ends in '.BAT'. It will try to execute these commands if you type the name of the file (with or without the '.BAT'). In particular, it always looks for commands to execute in a file AUTOEXEC.BAT as soon as it has scanned CONFIG.SYS. This AUTOEXEC.BAT file can like the CONFIG.SYS file be created by a COPY from CON (the keyboard) or with the editor EDLIN. Normally it will define a path to your files by using the PATH command or setting the current directory with the CHDIR or CD commands, load network software, set the date and time etc. before calling the program you normally use. For example, you might have these commands:

Simple example

```
DATE
TIME
NET START RDR FORT /NBC:4 /NBS:8K
NET USE D: /D
NET USE D: \\MST\MSTPUB
PATH C:;C:\FORTRAN;
CD\FORTRAN
```

We call the DATE and TIME programs to define a correct date and time. This may seem not very useful, but months later we might want to see which versions of files were created when! The default date and time, if not entered at power−on time, are always the same (usually 1 January 1980, 00:00 hrs) and this can lead to great confusion later on. We start the IBM PC Network software to define our machine as a 'redirector', called FORT, and ask for access to a server called MST (presumably running somewhere..) using a directory MSTPUB there as if it was a local drive D:. We set a PATH so that DOS 3.x will always look in these directories (the 'root', C: and a further directory C:\FORTRAN here) for programs with a '.COM' or '.EXE' extension. These are consulted in addition to any explicit path we specify as part of a 'filespec'. Lastly, we change the default directory to FORTRAN.

An AUTOEXEC.BAT for using DECNET – DOS and PC Net simultaneously

```
YNPROMPT Y N 39 Do you want to start PCNET...... (Y/N)?
IF ERRORLEVEL 1 GOTO B
:A
NET START SRV MST /NBC:8 /NBS:4K
NET SHARE MSTSYS = C:\MSTSYS /R
NET SHARE MSTPUB = C:\MSTPUB /RWC
:B
YNPROMPT Y N 39 Do you want to start DECNET..... (Y/N)?
IF ERRORLEVEL 1 GOTO C
CD\DECNET
SCH
DLL
DNP
NDU OPEN NODE VXCRNA NDISK a$b: < d > fname.ext;1 USER name/pwd DRIVE D
CD\MSTSYS
NET SHARE MSTVAX = D:\ /RWC
:C
BASICA
```

The question asked by YNPROMPT is shown twice: first, when YNPROMPT is called to ask it, and then when it asks it. This can be avoided by using ECHO ON/OFF as needed. The utility YNPROMPT is provided with the PC Net control program software. You specify two possible replies (Y, N here), the length of the 'prompt' (39 characters here) and then the prompt. Should you answer Y (Yes) above, the value of ERRORLEVEL is set to 0 and control passes to the label ':A' in the batch file. Should you answer N (No) ERRORLEVEL is set to 1 and control passes to label ':B'. In this latter case, the NET START commands etc. are not issued.

2.5 Device names in DOS 3.x

Certain names have a special meaning under DOS 3.x and they should not be be used as filenames, etc. The reserved names correspond to devices and are as follows:

CON	Keyboard (input) or screen (output). F6 is end of data..
COM1	Primary serial/parallel port
COM2	Alternate serial/parallel port
AUX	Alternate name for COM1:
LPT1	First parallel printer (output only)
LPT2,LPT3	Second, third parallel printers
PRN	Alternate name for LPT1
NUL	Null device, non existent, dummy for output

2.6 Special control combinations and DOS editing keys

Several keys or combinations of keys have specific meanings under DOS. You should know that when you type a line and press 'Enter', DOS puts a copy of this line in the 'input buffer'. You can subsequently use the contents of this buffer, change them etc. without typing the whole line again, by using the DOS editing keys. These keys and their meanings are listed below:

Enter	Enter the line now being displayed
Ctrl – break	End the current operation
Ctrl – Enter	Continue entering line on next display line
Ctrl – NumLock	Suspend operation; press any key to resume
Shift – PrtScr	Toggle: turn the printer on/off
ESC	Cancel current line
uparrow – PrtScr	Print the screen
leftarrow	Backspace and rubout character
Ctrl – ALT – DEL	Execute the 'AUTOEXEC.BAT' file
Ins	Insert characters in the line at cursor posn.
Del	Delete a character in line at cursor posn.
Esc	Cancel displayed command line (still in buffer)
F1	Display line buffer, char by char
F2 x	Display line buffer, up to character 'x'
F3	Display line buffer
F4 x	Delete all chars in buffer up to (but not) 'x'
F5	Put displayed line in buffer

2.7 DOS 3.x directory structure, PATH and filenames

DOS 3.x supports a tree – structure of directories. At the top of this tree is the 'root' directory which will normally contain all the DOS 3.x files. This root can contain pointers to further, lower levels of directory. These lower directories in turn may contain both files or pointers to lower levels of directory. To find a program or file, you need to specify unambiguously to DOS 3.x where to look. Such an unambiguous definition is a filename or 'filespec', and might look like this:

```
d:\level1\level2\....\leveln\filename.ext
```

Here, we specify a device 'd:' (diskette drive, disk or a disk shared by that machine via some network which has a letter A – Z as its name), and a descending path through a series of directories. Each level of directory is contained within '\' characters. Finally, having reached the directory where the file may be found, we name it as 'filename.ext'. A DOS 3.x filename is 1 – 8 characters, (usually A – Z

0 – 9) and the extension 'ext' is 1 – 3 characters (usually A – Z 0 – 9). If you are not sure about a filename or extension, you can use '?' to state that any single character will do in that position, or '*' to state that any series of characters may now follow. Thus, in the current directory:

```
*.*  means any file whatsoever
*.EXE any file with extension 'EXE'
PROG?.ASC any file PROGA.ASC, PROGB.ASC etc.
```

No matter what your current directory is, you can always access a file using the complete 'filespec'. Thus, to copy to the screen ('CON') a text file named 'mytext.asc' which is in second level directory 'MEMOS', pointed at from the root by the directory "TEXT', you could:

```
C>COPY C:\TEXT\MEMOS\mytext.asc CON:
```

However, you could very soon get tired of typing in long 'filespec's and getting them correct. You could, however, define a PATH using the DOS 3.x command. Thus, if we were to want to access many programs in a second level level directory as above, we could define a path to be looked at if the program can't be found in the current directory, whatever that might be:

```
C>PATH C:\TEXT\MEMOS
C>myprog
```

This will institute a search for 'myprog' in the current directory (probably but not necessarily the root) as well as all the directories named in the PATH comnand. However, only '.COM' or '.EXE' items will be found, so this is somewhat restricted. Note though that several different paths may be set separated by ';' as we show below:

```
C>PATH ;   (delete all previous settings)
C>PATH C:\;C:\TEXT\MEMOS;C:\FORTRAN;
```

Should the restriction that PATH **won't allow a search for files which do not have the '.EXE' or '.COM' extension** be a nuisance, you can always change the current directory. This is done with the CHDIR, or CD command. Thus to find any file in C:\TEXT\MEMOS, but still search the root and C:\FORTRAN for '.EXE' or '.COM' objects, you might:

```
C>PATH ;
C>PATH C:;C:\FORTRAN;
C>CD\TEXT\MEMOS
C>   (now current directory is C:\TEXT\MEMOS)
```

Several filename extensions have customary significance, being assumed as defaults by compilers, etc. Thus, a '.ASC' file is usually used to indicate a BASICA source SAVEd with the 'A' option, while '.BAS' is the default compressed format. A '.BAT' file will usually contain DOS commands to be executed. A '.COM' file will be a DOS command (such as DISKCOPY). A '.EXE' file will be an executable file created by compiling and linking a source program. FORTRAN source files are assumed to have a default extension of '.FOR'. The output from compilation is normally a '.OBJ' file. Device drivers normally are '.SYS' files (such as ANSI.SYS, the extended keyboard driver).

2.8 Redirecting standard input, output, piping, filters..

DOS 3.x does contain some features that UNIX users will recognise, and users of other operating systems will probably appreciate. These are the possibility of redirecting the standard input and output of programs (usually keyboard entries and screen output) to files, to automatically redirect the standard output of one program to be the standard input to another ('piping'), or performing 'filtering' with

MORE, SORT and FIND. For example, standard output is redirected from the command 'DIR' (or any other DOS command) to the file d:fname.ext by the following:

```
DIR > d:fname.ext
or
DIR > > d:fname.ext
```

The second form, using '> >' not '>', will add the output of DIR to the end of d:fname.ext, if it already exists. The first (more common) form will create a new file or replace an existing one. To redirect standard input to 'myprog' or SORT to be from a file, not the keyboard, you might use:

```
myprog < d:fname.ext
SORT < d:fname.ext
```

You need to be careful that all the expected standard input is to be found in the file! Piping exists to make the 'chaining' of the standard output of one program to the standard input of another very simple. The piping symbol is '|'. For example, to find all the files on your disk written in 1987, and sort the list to a file, you could use:

```
TREE C:/F | FIND "1987" | SORT > FILES.87
```

3. DOS 3.X SUMMARY OF COMMANDS

Commands I have never used

I have not needed to use COMP, EXE2BIN, GRAFTABL, JOIN, PROMPT, RECOVER, SELECT, SET, SHARE, SHIFT, VERIFY during the last five years!

However, all the commands are presented below, assuming that you are using a PC equipped with a disk drive. These commands can all be used (in a slightly different manner) if you have only one (or two) diskette drive(s). When 'filespec' appears in a command, enough information MUST be given to identify a file uniquely. This may imply changing the current directory, or giving the full path to the file as part of the 'filespec'. Note that parameters which are optional are enclosed in square brackets [..]; obligatory choice of an option is indicated by <x|y|...>. In many of the commands, the character '?' may be used to imply that any character may be accepted here in a filename or part of a filespec. Also, '*' can often be used to imply that any combination of characters following is acceptable.

3.1 ASSIGN

```
ASSIGN x=y [z=y] [..]
```

This tells DOS 3.x that whenever it finds a reference to the drive 'x', this must be applied to the drive 'y' instead. The usefulness of this is that many programs are compiled with use of diskette drives 'A' and/or 'B' in mind, and request these drives even on your XT with a hard disk, C. By using an ASSIGN A=C B=C command, you can reroute all requests for files on 'A' or 'B' to 'C'. **Beware that if you have entered ASSIGN A=C, if you then enter an ERASE A:*.* imagining that you are cleaning up a diskette, you will instead erase you hard disk!** To undo the effects of all previous assignments, enter:

```
ASSIGN
```

3.2 ATTRIB

```
ATTRIB [+R|-R][d:][path]filename[.ext]
```

This command allows you to set the read-only attribute of a file (use +R) or to remove it (use -R). Global filename characters (*, ?) are allowed in this command. If you omit both the +R and -R in the command, you are shown the **current attribute of the file.** This is a prefix 'R' to the result if the file is read-only. Beware that many software installation procedures (PC Network, for example!) make copies of existing files or move them, adding the read-only attribute. You can undo these effects with the '-R' option.

3.3 BACKUP

```
BACKUP d:[path][filename[.ext]] d:[/S][/M][/A][/D:dd-mm-yy]
```

This command allows you to BACKUP files from one disk(ette) to **another** disk(ette), from which you may later RESTORE them. The first named disk(ette) is **the source disk(ette)**, from which file will be backed-up to the second named disk(ette), **the target disk(ette)**. Use /S to backup files in subdirectories as well as the specified or current directory. Use /M to backup files **modified since last backup**. Use /A to add the backed up file to those already present on the target disk(ette). Use /D:dd-m-yy to backup files **modified or created since the date given**. Note the format of date used depends on your COUNTRY command in force. Beware that BACKUP/RESTORE insists on retaining the directory structure: if you BACKUP a particular directory as below, it must be available for a subsequent RESTORE:

```
C>BACKUP C:\TCP\*.FOR A:
and later.... to restore *.FOR on a machine without these files..
C>MD C:\TCP
C>RESTORE A: C:\TCP\*.FOR
```

BACKUP will use roughly 25 diskettes to backup a 10 Mbyte fixed disk. Global characters (*, ?) are allowed in BACKUP commands. This command does not produce the same results as COPY on the target device; you need to use RESTORE to get files back.

3.4 Batch file commands

```
[d:]filespec [parameter]
```

This invokes batch processing of the contents of file 'filename.ext' on drive 'd:' (default drive, if not given). You need not specify the extension yourself in the call: it will be assumed to be '.BAT', '.COM' or '.EXE'. Note that '.BAT' files can call each other, if you wish to do so. You must use the COMMAND command if you wish to return correctly up the chain of .BATs, however.

3.5 BREAK (control-break)

```
BREAK [ON|OFF]
```

This requests DOS 3.x to check for ctrl-break whenever it is requested (BREAK=ON) or only when performing standard I/O, print or auxiliary operations (BREAK=OFF). Just typing BREAK shows you the current setting; the default is OFF. You can include this command in your CONFIG.SYS file. Remember that OFF can make it rather slow to interrupt some commands or programs.

3.6 CD or CHDIR

```
CD [d:][path]
CHDIR [d:][path]
```

CD and CHDIR are the same command. It changes the current directory of the specified or default drive, or displays it. Use just CD to display the current directory of the default drive, CD [d:] to display the current directory of drive d:. Specify path if you want to change the current setting. Note that to reset the current path to the root directory of the default drive, use:

```
CD \
```

You may created a directory with the MD (or MKDIR) command, and delete an empty directory with the RD (or RMDIR) command.

3.7 CHKDSK

```
CHKDSK [d:][path][filename[.ext]][/F][/V]
```

This analyses the directories, files and the File Allocation Table on the designated or default drive and produces a disk and memory status report. **This is a good way to see how much 'free' memory you have.** If errors are noted, you should should try the Diagnostics diskette supplied with your copy of DOS. If you specify /V, it will display all files and their paths on the specified drive. If you specify /F, it will be capable **of fixing up disk errors on your hard disk**. These are the so-called 'lost allocation units', which usually result from corrupted directories. With the /F option (and reply to other prompts) CHKDSK collects these together into files which it names as FILEnnnn.CHK (nnnn are sequential from 0000, starting with the first set of lost allocation units). These 'files' are created in the root directory; you can try editing them to see if they are useful or delete them as you wish. **Normally you should have very few or no lost allocation units on you fixed disk!**

3.8 CLS (clear screen)

```
CLS
```

This will clear the display on the standard output device. Normally, this is the screen.

3.9 COMMAND

```
COMMAND [d:][path][/P][/C string]
```

Here [d:][path] is used,if given, to find the command processor you want. COMMAND.COM should be found there, or in the path specified in you your environment (see PATH commands or PATH in your CONFIG.SYS file). This is used to allow you call batch files (.BAT) from each other and to return to the calling batch file. Specifying /P causes the new command processor to be permanent in memory, and the secondary command processor loaded by COMMAND will not return to the primary processor. The /C allows you to send a string to the secondary processor being loaded. This string is issued as if you had type it at the DOS prompt. Thus you might use the following sequence:

```
C>level1.BAT     .... and in this .BAT file,
C>COMMAND /C level2.BAT  .... which calls level2.BAT
C>  .... which returns to level1.BAT after execution
```

3.10 COMP

```
COMP [d:][path][filename[.ext]] [d:][path][filename[.ext]]
```

Compares files, byte for byte. After it has found 10 mis-matches, it will halt with a message 'BAD COMPARE'. GLobal characters (* and ?) are allowed in the primary and secondary filenames you give. If you give no parameters, you will be prompted. The file(s) to compare may be on the same or different drives, and in the same or different directories.

3.11 COPY

```
COPY filespec [d:][path][filename[.ext]][/V]
COPY filespec + filespec [d:][path][filename[.ext]][/V]
COPY filespec <LPT1:|LPT2:|LPT3:|CON:>
```

This can copy, concatenate or print/list files. The '/V' option has the same effect as the VERIFY ON command, forcing a check of all data written to disk. The '/A' and '/B' options may be used to determine the amount of data that COPY will treat. Option '/A' implies all files are treated as ASCII text files, and all data up to (but not including) the first ctrl-Z character (end-of-file) is copied but no subsequent data. Option '/B' causes the entire file to be copied. Note that '/A' is default when **concatenation is being performed**, and '/B' when it is not.

Global characters '*' and '?' may be used in the COPY command. Here are some examples of using COPY:

```
COPY file.ext newfile.ext (file copy, new name for copy)
COPY A:*.ASC C: (copy all files '.ASC' from A: to C:)
COPY A:*.* C: (copy all file from A: to C:)
COPY file1.asc+file2.asc both.asc (concatenate to 'both.asc')
COPY *.ASC all.ASC (concatenate all '.ASC' file to 'all.ASC')
COPY file.ext LPT1: (copy file to printer LPT1:)
```

3.12 CTTY

CTTY device-name

This allows you to change the standard input/output console for DOS 3.x function calls. The device names you may use are CON (default), AUX, COM1 and COM2. Note that not all programs use DOS function calls for standard input/output: BASICA, for example, does not. This can be used to specify a terminal as the console, if you have no display or are using it for some special purpose and do not wish to see DOS prompts, etc. on it:

```
C>MODE COM1:9600,N,8,1,P
C>CTTY COM1:
```

3.13 DATE

DATE [dd-mm-yy]

This allows you to specify the date to DOS 3.x. The format of date accepted and displayed (call DATE without any parameter) is decided by the COUNTRY in force. Normally, you will include the DATE command in your AUTOEXEC.BAT file. The default is usually something like 01 Jan 1980, unless your machine has a real time clock of some type.

3.14 DEL (delete file)

DEL [d:][path]filename[.ext]

This will delete the specified file. If you specify no drive d:, the default drive is assumed. Specify path to access a particular directory other than the current one. You may use the global characters (*, ?) in the 'filename', and beware that **by default '*.*' is assumed for 'filename'!** Use RD (RMDIR) to delete a directory (must be empty).

3.15 DIR (directory)

DIR [d:][path][filename[.ext]][/P][/W]

This will list directory entries for the specified files on device 'd:' (or default drive). Specify '/P' to pause when the screen is full; specify '/W' to use a wide (80 column) presentation. Dates shown are in a format depending on the COUNTRY in force. You may use global characters (*, ?) in the filename and the extension need not be given (.* will be assumed).

3.16 DISKCOMP (compare diskettes)

DISKCOMP [d: [d:]]

This will compare complete diskettes, rather than files as with the COMP command above. Always compare diskettes of similar capacity and format, to avoid confusion (i.e. 160K, 320K, 360 K and 1.2 Mb only with diskettes of 160K, 320K, 360 K and 1.2 Mb).

3.17 DISKCOPY

DISKCOPY [d: [d:]]

This will copy a diskette. If the target diskette is not formatted, it will be formatted automatically. Always copy diskettes to others of similar capacity and format, to avoid confusion (i.e. 160K, 320K, 360 K and 1.2 Mb only to diskettes of 160K, 320K, 360 K and 1.2 Mb).

3.18 ECHO (batch subcommand)

ECHO [ON|OFF|message]

This allows ('ON', default) or stops ('OFF') the display of commands executed from a batch (.BAT) file, including the REM command. It has no effect on messages **produced by the execution** of the commands themselves. Should you use the form 'ECHO message', 'message' will **always be displayed** regardless of whether ECHO is ON or OFF. ECHO with no parameters will display the status at the time.

3.19 ERASE

ERASE [d:][path]filename[.ext]

This will erase (delete) the specified file. If you specify no drive d:, the default drive is assumed. Specify path to access a particular directory other than the current one. You may use the global characters (*, ?) in the 'filename', and beware that **by default '*.*' is assumed for 'filename'!** Use RD (RMDIR) to erase (delete) a directory (must be empty).

3.20 EXE2BIN

EXE2BIN [d:][path]filename[.ext] [d:][path][filename[.ext]]

This converts '.EXE' file to '.BIN' or '.COM' file format. This is a memory image of your program, and can be called like a DOS '.COM' command. The input file must be a valid '.EXE' file, produced by the LINK program. I have never used this.

3.21 FDISK

```
FDISK
```

This is used when you prepare your fixed disk, normally **only once**. You should not keep this on your hard disk to avoid errors (type DELETE C:FDISK.• to get rid of it). If you have succeeded in making more than one disk partition, FDISK can change the 'active' partition' for you. See the section on 'Updating to DOS 3.x, or formatting the disk for the first time'.

3.22 FIND filter

```
FIND [/V][/C][/N]"string"[[d:][path]filename[.ext]]
```

This is one of the DOS 3.x 'filters' (see also SORT and MORE). It sends all lines from the specified filenames which contain "string" to the standard output device. If you use '/V', it sends lines **not containing "string"**. If you specify '/C', it gives a count of lines found containing "string" (you may not use '/C' with either '/V' or '/N'). Using '/N', you get a relative line number in the file for each line found (good pointers for EDLIN, etc.). Enclose the string as shown in double quotes ("). Case is significant in the "string". This command is commonly used with **piping** as in this example, to find all files '•.ASC' in the current directory and put their names in the file PROGS.LST:

```
DIR | FIND ".ASC" > PROGS.LST
```

3.23 FOR (batch subcommand)

```
FOR %%variable IN (set) DO command
```

This allows iterative execution of DOS commands. The %%variable is set to successive members of set, and command is evaluated and performed. A file may be used for set, but FOR may not be nested.

3.24 FORMAT

```
FORMAT [d:][/S][/1][/8][/V][/B][/4]
```

This will format a blank diskette. Formatting is necessary before other commands may be used, apart from DISKCOPY which will perform the formatting if necessary. The '/S' option allows you to format and place the principal system files on a diskette. The presence of these principal DOS files (IBMBIO.COM, IBMDOS.COM and COMMAND.COM) on your utility diskettes may reduce the amount of diskette manipulations you eventually require. If you specify B and not S, any version of these files from another DOS version may be put on the diskette by a later execution of the SYS command. 1 means one-sided format, 8 means use only 8 sectors per track of the default 9 or 15. I have never used 1 or 8! The '4' option is useful, as it allows you to perform a 360 Kb formatting in the 1.2 Mbyte type drive of an AT or AT look-alike, which can then be read by an older XT or PC-1.

3.25 GOTO (batch subcommand)

```
GOTO :label
..
..
:label
```

This construction in a batch file allow transfer of control to the line following the label line. The label consists of 1 to 8 significant characters. Labels are never echoed.

3.26 GRAFTABL (load graphics table)

```
GRAFTABL
```

This command permits the display of 'foreign' language characters on the colour graphics adapter, using ASCII values 128-255 for this purpose. If you use it, it should normally be placed in your AUTOEXEC.BAT file.

3.27 GRAPHICS (screen print)

```
GRAPHICS [printer type][/R][/B]
```

This allows the contents of a graphics display screen to be printed on an IBM PC printer. The printer type is one of GRAPHICS (default), COMPACT, COLOR1, COLOR4 or COLOR8. Option '/R' allows you to reverse black and white when copying the screen to the printer. Option '/B' allows you print the background colour if you have specified COLOR4 or COLOR8. The default is not to print the background colour.

3.28 IF, IF EXIST, IF ERRORLEVEL (batch subcommand)

```
IF ERRORLEVEL number command
IF EXIST [:d][path]filename[.ext] command
IF string1 = = string2 command
IF NOT [one of above conditions] command
```

This construction allows conditional execution of DOS commands within a batch file. You can set ERRORLEVEL yourself if you use the PC Net utility YNPROMPT to ask a question. BACKUP and RESTORE also set the value of ERRORLEVEL. The existence (or otherwise) of a file can be tested with the EXIST (or NOT EXIST) condition, also allowing transfer of control. Strings may be compared also (normally using the batch file **parameters** in the test, i.e. IF %1 = = C THEN MODE CO80 . Global characters '•' and '?' are allowed in the EXIST condition filename. For example, you might have an AUTOEXEC.BAT file like this:

```
YNPROMPT Y N 39 Do you want to run USER program. (Y/N)?
IF ERRORLEVEL 1 GOTO B
NET START RDR GKSAT /NBC:8 /NBS:4K
NET USE D: \\MST\MSTSYS
IF EXIST C:USER.NAM GOTO A
REM Get USER.NAM data from MST..
COPY D:USER.NAME C:USER.NAM
:A
USER.EXE
:B
```

3.29 JOIN (join directories)

```
JOIN
JOIN d: d:\directory
JOIN d: /D
```

This connects a drive to directory on another drive, to produce a single directory structure from two separate directories. JOIN on its own displays the directories and drives currently joined, and the '/D' option disconnects the specified join. You cannot use JOIN for a network drive.

3.30 KEYBxx (load keyboard)

```
KEYBxx
```

This loads keyboard program 'xx' where 'xx' may be UK, GR, FR, IT or SP. Normally you will not use this command, and use COUNTRY in your CONFIG.SYS file instead. You may only reset the keyboard once without restarting DOS 3.x. Beware that the keyboard transmits only 'scan codes' (essentially a physical key position code) to the PC. The interpretation of the fifth key along the second row as a visible character echoed on the screen, etc. depends on DOS (via KEYBxx) if the software you are running uses standard I/O. This may well not be true.. if results seem odd, you should try the assumption of a US English keyboard layout.

3.31 LABEL (volume label)

```
LABEL [:d][volume label]
```

This allows you to label, or re-label, the volume specified or the default drive volume. The label may be 1-11 characters.

3.32 LINK (utility LINKer)

```
LINK
```

This will invoke the LINK program. This allows you to produce a version of a compiled or assembled program, which contains all necessary code from the system to run by itself. The module produced is relocatable. LINK will combine separately produced object modules, and search library files to resolve external references. If you use the command LINK, it will prompt you for options etc. You may also call it with parameters defining the options you want: this is **command line mode**. You may also define an **automatic response file** which is consulted as if you were replying to prompts. This allows you to given very long replies, which are sometimes needed for linking with many libraries.

Command line mode

```
LINK objlist,runfile,mapfile,liblist [/param[/param[/....]]]
```

The 'objlist' is the object module to link (produced by BASCOM or PFORT, etc, or a list of separate object modules separated by spaces or '+' signs. The name you wish to give to the runfile is 'runfile'. The file to use for the linker map is 'mapfile': use NUL.LST to avoid one. The library to be consulted for external references is 'liblist', which may also be a list of libraries separated by spaces or '+' signs. Finally, you may specify any of the following parameters.

```
/D /H /L /M /P /S:nnnn /X /O
```

Of these, you will need /S = 5000 and /X = 1000 for linking Professional FORTRAN and the GKS libraries. Command line mode can be insufficient as the line length may not may not exceed 128 characters. Complex paths may soon use this use up this allowance!

Using an automatic response file with LINK

```
LINK @[:d][path]filename[.ext]
```

The text file given after the '@' sign must contain the replies you would have given to LINK in prompt mode, in the same order. You may use several lines within this file to reply to what would have been a prompt, by using a '+' sign to continue the response to the next line in your file.

3.33 MD or MKDIR (make directory)

```
MD [:d]path
```

This allows you to create a directory on the specified (or default) drive. The path specifies the directory name to be created. Empty directories may be removed with RD (or RMDIR), and the current directory changed by CD (or CHDIR).

3.34 MODE

```
MODE LPT1:[n][,m][,P]]
```

This sets the mode of operation of LPT1. If you have them installed, you may of course specify LPT2 or LPT3. The number of characters per line is n, either 80 or 132. The vertical spacing, m, is either 6 or 8 lines per inch. Power-on defaults are 80 characters and 6 lines per inch. If the printer is reset or initialised (for example BASICA does this) the defaults are set. You can retry 'timeouts' by specifying 'P'.

```
MODE n
MODE [n],m[,T]
```

This switches display adapters, or defines their mode of operation. Here 'n' may be any of 40, 80, BW40, BW80, CO40, CO80 or MONO. Using 40 or 80 sets the linelength of the colour monitor to 40 or 80 characters. BW means 'black and white mode' for the colour monitor (colour disabled), and CO enables colour. MONO enables the monochrome display, which is always 80 characters wide. Using the value L or R for 'm' will shift the display left or right 2 characters for legibility (especially if you use a video monitor and not an IBM display). T requests a test pattern to allow alignment of the display.

```
MODE COM1:baud[,parity[,databits[,stopbits[,P]]]]
```

This permits you to set the characteristics of the asynchronous communications adapter, COM1:. If you have the alternate adapter COM2: installed, you may of course specify it as well. The baud rate may be 110, 150, 300, 600, 1200, 2400, 4800 or 9600 baud. Telephone acoustic couplers usually run at 300 or 1200 baud. Parity is N (none, means 8 databits normally), O (odd) or E (even). Databits are 7 or 8, and stopbits 1 or 2. P allows you to retry timeouts, with use of a serial printer in mind, for example by:

```
MODE COM1:9600,N,8,1,P
MODE LPT1:=COM1
```

This would allows you to redirect output for LPT1: to the communications adapter COM1:, where a serial printer might be attached.

3.35 MORE filter

This DOS filter takes input from the standard input device, and sends one screen at a time to the standard output device. The prompt '– – More – – ' is then displayed, and another screen delivered should you press any key. Thus you might enter:

```
C>TREE | MORE
```

3.36 PATH (set search directory)

```
PATH [[d:]path[;[d:]path]...]]
PATH ;
PATH
```

This allows you to specify a list of drive and path names that will be searched for commands (.COM, .EXE or .BAT files) that cannot be found in the current directory. Several paths may be given, separated by the ';' sign. Typing just PATH displays the current path settings, and 'PATH ;' removes all path information and resets the path to null (i.e. search only the current directory).

3.37 PAUSE (batch subcommand)

```
PAUSE [remark]
```

This will pause the system with 'remark', and the message 'strike any key when ready' to resume. Pressing Ctrl – Break will end processing.

3.38 PRINT

```
PRINT [:d][path]filename[.ext] ...
```

This allows you to print the specified file (or list of file) on the printer. Once the command is entered, you may continue to perform other tasks without waiting for the printing to finish. See the DOS 3.x manual for details. You cannot use PRINT if your printer is shared on PC Net.

3.39 PROMPT (set system prompt)

```
PROMPT [prompt – text]
```

You may redefine the system prompt with this command. See the DOS 3.x manual for details, but remember that the prompt – text can contain meta – strings such as $t, $d, $p, $v and $g (among others) that will be displayed as the time, date, current directory of default drive, DOS version number and default drive letter.

3.40 RECOVER

```
RECOVER [d:][path]filename[.ext]
```

With this command you may attempt to recover the file specified from a disk which has a bad sector, although the contents of the bad sector will be lost. This only really makes sense for ASCII text files. Should you destroy your directory, you can try:

```
RECOVER d:
```

This reconstructs all the chains of allocated units it finds on drive d: into files called FILEnnnn.REC. The value of nnnn starts at 0001. It is up to you to try to guess what the files are: you could find it useful for corrupted ASCII text files.

3.41 REM (batch subcommand)

```
REM [remark]
```

This displays remarks in a batch file.

3.42 RENAME or REN

```
REN[AME] filespec filename[.ext]
```

This permits renaming of a file or files.

3.43 RESTORE

```
RESTORE f: [d:][path]filename[.ext][/S][/P]
```

This restores the file or files specified to from drive 'f', to which the said files were moved by BACKUP. Use '/S' to restore all files in subdirectories beyond the specified directory, as well as the files in the specified directory. Use '/P' to be prompted before RESTORE overwrites files that are still present, and are either 'read only' or modified since BACKUP moved them out. RESTORE sets ERRORLEVEL to 0 if it completes normally, 1 if no files were found to restore, 2 if some files were not restored due to sharing conflicts (PC Net), 3 if terminated by the user (Ctrl – Break or Esc), and 4 if terminated due to error. Beware that BACKUP/RESTORE insists on retaining the directory structure: if you BACKUP a particular directory as below, it must be available for a subsequent RESTORE:

```
C>BACKUP C:\TCP\*.FOR A:
and later.... to restore *.FOR on a machine without these files..
C>MD C:\TCP
C>RESTORE A: C:\TCP\*.FOR
```

3.44 RD or RMDIR (remove directory)

```
RD [:d]path
```

This allows you to delete an empty directory (i.e. one with no entries but '.' and '..'). You cannot remove directories that contain hidden files, or the root directory, or the current directory. You may create a directory with MD (or MKDIR) and change the current directory with CD (or CHDIR).

3.45 SELECT

```
SELECT xxx yy
```

This allows you to prepare a copy of your **DOS diskette** containing a CONFIG.SYS file with the appropriate COUNTRY=xxx command, and an AUTOEXEC.BAT file containing a relevant KEYBxx command. You are most unlikely to use this command if you use a disk machine.

3.46 SET (set environment)

```
SET [name=[parameter]]
```

This command inserts strings into the **command processor's environment**. What does this mean? It allows you to specify the kind of display you are using, etc. For example:

```
SET COMSPEC=D:
```

This allows all DOS utilities to know that they should look for COMMAND.COM on drive D: (perhaps a networked, or virtual (VDISK) device. On the other hand, if you use IBM's GKS system,

```
SET DISPLAY=VDIDY004.SYS /R
```

lets all DOS utilities, in particular VDI.SYS, the GKS Virtual Display Interface, know that DISPLAY is a standard 200 x 320 4 colour display, which is to be found resident. Omit 'parameter' to 'undefine' the name you give. Simply enter SET to display the environment. Note that lower case is converted to upper by SET. Any definitions of PROMPT or more importantly PATH are added by DOS to the environment as you redefine them. This can cause annoyance; if you have loaded a resident item of software (such as MODE, PRINT or GRAPHICS) DOS cannot expand the environment area (used to hold all these strings) to more than 127 bytes. For complex PATHs, this is a severe limit.

3.47 SHARE

This command loads support for file sharing in a PC Net system. However, the user himself (or his purchased software) must use the DOS 3.x calls for opening and closing files which are shared, provide his own record locking etc. DOS 3.x does not allow you simply to state that a given file (of commonly used data, etc.) **is to be shareable!**

3.48 SHIFT (batch subcommand)

This allows you to call a batch file with more than ten replaceable parameters, %0 to %9. Each time you call upon SHIFT in your batch file, the value of %0 is reset to that of %1, that of %1 by that of %2, etc.

3.49 SORT filter

```
SORT [/R][/+n]
```

This DOS filter command reads data from the standard input device, sorts it, and writes it to the standard output device. Use '/R' to reverse the sort order (i.e. Z before A). Use '+n' to specify where n is an integer to start sorting with column n. Note that a–z are the same as A–Z as far as SORT is concerned. For example, to sort your current directory by file size (given starting in column 14) you could say:

```
C>dir | SORT /+14
```

The DOS piping feature (|) is used here to send the output of DIR to SORT.

3.50 SUBST (substitute)

```
SUBST :d d:path
SUBST :d /D
SUBST
```

This command allows you to specify a 'dummy' drive letter instead of a combination of drive and path. This allows you to continue to run applications that do not support **paths** by using a 'dummy' drive letter, defined by your SUBST command. Use SUBST with '/D' to delete a SUBST substitution, and just SUBST to see what substitutions are in effect.

3.51 SYS (system)

```
SYS d:
```

This will put the operating system files IBMDOS.COM and IBMBIO.COM on diskette ':d' (a diskette formatted by FORMAT d:/S or FORMAT d:/B). Other files (such as COMMAND.COM) must be transferred explicitly.

3.52 TIME (set machine time)

```
TIME [hh[:mm[:ss[.xx]]]]
```

This allows the user to enter or change the time. The default (unless your machine has a real–time clock) is 00:00:00.

3.53 TREE (sho all directories)

```
TREE [:d][/F]
```

This command displays all of the directory paths found on the specified drive, and optionally (use /F) displays the names of the files in the root and subdirectories.

3.54 TYPE (list file contents)

```
TYPE [:d][path]filename[.ext]
```

This will display the contents of the file on the screen.

3.55 VER (version)

```
VER
```

This displays the DOS version number that you are working with on the standard output device (normally the screen).

3.56 VERIFY

VERIFY [ON|OFF]

VERIFY, if ON, requests DOS to perform a verify operation after **each disk operation**. This causes the system to run slower if many disk writes are being performed. You can check the status of VERIFY by typing it with no options.

3.57 VOL (Volume)

VOL [d:]

This will display the disk volume label of the specified (or default) drive.

4. SOME UTILITIES WHICH ARE NOT INCLUDED IN DOS 3.X

4.1 BASCOM compiler (IBM/Microsoft)

One thing that must be remembered before running programs in BASICA is that any program you run, unless you compile it, will be interpreted character by character throughout the program's execution. If we know that to interpret each character takes about 10 machine instructions we can soon see that more time is often spent interpreting than in useful execution. This (in many cases) is acceptable. If the program is small and maybe not often used, then the time needed to sort out this problem is not justified. However, if we compile and link our BASICA program then we have a program that will execute without any time wasted on interpretation. The bigger or more frequently used the program the greater the saving in execution time will be. To compile and link it we must first save the program in ASCII format, as the compiler cannot interpret 'compressed BASIC'. If this is not the case, go into BASICA, reload the file and make any final changes to it. This time save it as an ASCII file using the ',A' option of SAVE. Then enter either of the commands:

```
BASCOM [/option][/option]...[/option]
BASCOM srcfil,objfil,lisfil[/option][/option]...[/option]
```

Either will start the compiling process. The first will prompt for the name of your source file, the name of the object file to be produced, and where to put the source listing. You can specify these files directly in the second command line format. The object file produced should then be submitted to the LINK program. You may need to specify some of the compiler options when you call BASCOM. Options, and their significance, are as follows:

/E

The compiler will allow use of ON ERROR and RESUME constructions. These can be very useful, but slow down execution and increase the memory requirements. Extra code is generated for every Enter or GOSUB statement, and a line number address table is created so that errors can be related to line numbers. This requires 4 extra bytes for every line in the program.

/X

This performs all the functions of the /E option, with additional checks, and even larger overheads. You may then use RESUME, RESUME NEXT or RESUME 0 statements.

/V

This option is needed if you use event trapping itself (COM(n), KEY(n), PEN or STRIG(n) statements). One extra byte of code is generated for every statement.

/W

This is very similar to the /V option, generating an extra byte of code for event trapping at every line number. If you have only one statement on every line, it is the same as /V.

/C:nnnnn (where nnnnn bytes are allocated)

This option allows you to specify the size of the communications buffer. This must be made larger as your program becomes slower to give it a chance to keep up with processing of I/O. By default, only 256 bytes are allocated to the buffer, and this is shared by input and output for COM1: and COM2: .

/O

This option allows you to give copies of BASICA programs that you have written to others, in an executable form. If you compile BASICA without this option, you need to have the library BASRUN.EXE (or BASRUN20.EXE, depending on the version you use) available at run−time. However, this library is part of the BASCOM compiler software and you may not give a copy to another user. Happily, the following method has been provided for you to avoid this copyright problem:

```
C>BASCOM/...../O    (usual options, together with '/O')
C>LINK
```

When prompted by the BASCOM program for a library, **do not specify any library**. In reply to the prompts of the LINK program for the object code to link, give the name of your program and IBMCOM separated by a '+' sign: **myprog + IBMCOM** and specify **no .LIB files either**.

/D

This option allows the use of TRON and TROFF statements within a compiled program. Arithmetic overflow, array bounds and line number event trapping are automatic. Object code is larger and slower, of course!

/4, /T, /A, /N, /R, /S

See the BASIC Compiler Manual (6024003) for details of these (less useful) options.

4.2 Professional FORTRAN compiler (IBM/Ryan McFarland Corp)

```
PROFORT srcfil[.ext]/options
```

This compiler requires the presence on your machine of a co−processor. You must also specify at least BUFFERS = 10 and FILES = 10 in your CONFIG.SYS file. It assumes the extension '.FOR' if it is not given. Any path information may be used in 'srcfil'. The 'srcfil.OBJ' files produced by a successful compilation maybe be put into a library with the LIB utility. Multiple options can follow the '/', in any order, except for /C and /P. These are followed by specific values, and another '/' must precede the following option(s) if any. The most useful options are the following (consult the manual 6362003, Installation and Use, for more details).

/B

Assume that all adjustable or fixed−size arrays are **larger than 64 Kbytes. The default is 'less than 64 Kb', but** this may very often be insufficient for serious programs. You could, if pressed for speed, compile a separate version without '/B'.

/LE

Suppress source code listing but show error and warning messages. If you use just '/L', you still get errors and warnings but all the source list as well.

/I

Make all INTEGER data INTEGER∗2, unless specifically defined to be INTEGER∗4. This decreases program size, but may give precision problems for imported codes.

Note that FORTRAN WRITE(n) or READ(n) statements to UNIT n will attempt by default (except for the units 5 and 6 which default to the keyboard and screen) to link the datastream to a filename, FORTn. You can use the SET command of DOS to link and then unlink this name to another file, or, if you have given a filespec in your OPEN statement, to change this at time to be another file. For example:

```
SET FORT6=filespec
SET FORT6=
SET PROG6.DAT=other.DAT
```

You can use redirection of standard input or output (< and >) to redirect READ(5) or WRITE(6) statements to files or other devices.

4.3 LIB library maintenance utility

```
LIB
LIB[libfil[/P:n] operations [,[listfil]][,[newlib]][;]]
LIB @filespec
```

LIB will prompt you in the first format. In the second, 'libfil' is the name of a library file. You can specify the library pagesize to be 16, 32, 64, 128, 256 or 512 (default) bytes with '/P'. The 'operations' parameter is a list of filenames preceded by any of the operators + (add), − (delete), ∗ (extract object module named in a file with the name of the extracted item and extension .OBJ), − + (replace an object module with the contents of an object module with same name) or −∗ (remove named object module to .OBJ file and erase it. If you run out of space on one line to specify all your operations, put a '&' after the last one on the line and continue on another line. You may use an automatic response file if you wish (LIB @filespec). For example, to create a library MYFORT.LIB, adding MYSUB.OBJ, you could:

```
LIB MYFORT.LIB + MYSUB.OBJ
```

To subsequently replace BAD.OBJ in MYFORT.LIB, or to replace it with a newer corrected version, you could:

```
LIB MYFORT.LIB−BAD.OBJ
LIB MYFORT− +BAD  (.LIB and .OBJ assumed)
```

Should you wish to make some changes to MYFORT.LIB, but leave it as it was and put the revised library into a new file NEWFORT.LIB, you should:

```
LIB MYFORT+SUB1+SUB6−SUB7,,NEWFORT.LIB
```

5. EDITING FILES

An editor running under DOS 3.x normally allows you to to create, edit, and save versions of ASCII text files. There is a real editor called EDLIN, and there is BASICA itself. BASICA is not really an editor, but is very suited to adapting and testing BASICA programs.

5.1 EDLIN line editor

EDLIN performs its functions on a line-by-line basis, on any ASCII text file. To start it, you should type EDLIN filename where filename is the name of an existing file or the name of the file you wish to create. EDLIN will prompt with an asterisk, *. EDLIN lines are numbered from 1 to 65529. The symbol '.' is normally used to represent the current line. Also, '#' means the next line after the last line (in memory..).

EDLIN filespec

This will edit (or create, if it does not exist) a file called 'filespec'.

[line]

This will enter the line number to be edited.

[n] A

Append line(s) from disk(ette) to file in memory. This is only meaningful if the file is too large to be resident entirely in memory.

[line][,line] D

Delete the indicated line(s).

E

End EDLIN and save the edited file. The original file is then renamed to become 'filename.BAK' to try to avoid serious errors...

[line] I

Insert before indicated line. The following lines are 'pushed down'. You must enter Ctrl-Break to stop inserting lines once you have entered insert mode.

[line][,line] L

List the indicated line(s).

Q

Quit EDLIN without saving the edited file.

[line][,line][?]Rabc[F6xyz]

Replace the first string, 'abc', with the second string, 'xyz' in the indicated line(s). If '?' is is specified, you will be asked to verify each change.

[line][,line][?]Sabc

Search for the string 'abc' in the indicated line(s). If you specify '?', you are prompted at each occurrence of the string before the next is searched for. This must be one of the most bizarre editing commands ever devised! Note however that the strings need not be of the same length.

[n] W

Write n line(s) on disk(ette) from memory. This is only meaningful if the file is too large to be resident entirely within memory.

5.2 BASICA as an editor adapted to BASIC

By typing the command BASICA you will load the Advanced BASIC or BASICA program. You will see a title on the top of the screen and the number of free bytes available. On the bottom of the screen you will see the definition of the program function keys (F1 to F10) that are at the left of the keyboard. These can save you quite a lot of unnecessary typing. At this point, you can immediately start to create a file by typing linenumber expression. For example:

10 PRINT " HELLO"

This can be an expression in BASIC or just text of any type. To list what you have already created you can type LIST, or to look at a particular line n or lines n-m, LIST n or LIST n-m. If you start by using the linenumber 10 or 100 you can easily insert new lines. The numbers can be resequenced by using the RENUM command which in the case of a BASICA program will take care of references to other line numbers in such statements as GOSUB, GOTO etc. You can use the insert and delete editing keys, you can delete line with DELETE n or (for a range of lines) DELETE n-m. **The major missing element is a string search.**

Nevertheless, using BASICA as an editor for your BASIC programs has some distinct advantages. You may at once RUN your program to see if it works. Lines are interpreted and executed and the syntax is checked as this is done. You may trace the execution (with the TRON and TROFF function keys, or by using these as BASIC statements). At any time you can halt interpretation with Ctrl-Break, and interrogate the value of any variable. Thus if you have used N% as a FOR-NEXT loop index, you can just enter:

?N%

and you will be shown the value that N% currently posesses.

To save the program just entered, use the SAVE command followed by the name you wish to give to the file. For example:

SAVE "TEST.ASC",A

This will create a file on disk called TEST.ASC. By using the option ',A' of SAVE, your file is always saved as an ASCII file. This is essential if you want to look at the file later without using BASICA. BASICA by default saves its data in 'compressed BASIC' format, where a very few characters are used to represent the line numbers and 'commands' (such as PRINT) of BASICA. This compressed format saves space, but makes the files thus saved unintelligible except to BASICA. You may leave BASICA (hopefully after saving your program) by entering the command SYSTEM. You will then see the DOS prompt again. To reload you file later into BASICA for additions or corrections, enter the BASICA command:

LOAD "TEST.ASC"

6. BASICA COMMANDS

The following is a summary of the BASICA commands, which fall into one of three categories, Command, Function or Statement. The list is not exhaustive, but should contain all the most frequently required commands.

(C) = Command (F) = Function (S) = Statement.

ABS (F)	v = ABS(x) returns the absolute value of x
ASC (F)	v = ASC(x$) returns ASCII code for 1st char of string x
ATN (F)	v = ATN(x) returns the arctangent of x
AUTO (C)	AUTO[number][,[increment]] generates a line number automatically
BEEP (S)	BEEP beeps the speaker.(800hz for 1/4 sec.)
BLOAD (C)	BLOAD filespec [,offset] Loads a memory image file into memory.
BSAVE (C)	BSAVE filespec,offset,length saves a portion of memory .
CALL (S)	CALL numvar [(variable[,variable]..)] calls a m/c language subroutine
CDBL (F)	v = CDBL(x) converts x to a double – precision number
CHAIN (S)	CHAIN[MERGE]filespec[,[line][,[ALL][,DELETE range]]] transfers control to another program and passes variables from it to the current program.
CHR$ (F)	v$ = CHR$(n) converts an ASCII code to its character equivalent.
CINT (F)	v = CINT(x) converts x to an integer
CIRCLE (S)	CIRCLE(x,y),r[,colour[,start,end[,aspect]]] (graphics only) draws an ellipse on the screen with center (x,y) and radius r
CLEAR (C)	CLEAR[,[n][,m]] sets all numeric values to zero and all string variables to null. Options set the end of memory and amount of stack space.
CLOSE (S)	CLOSE[[#]filenum[,[#]filenum]...] concludes I/O to a device or file.
CLS (S)	CLS clears the screen
COLOR (S)	COLOR[foreground][,[background][,border]] sets the colours for the screen
COM(n) (S)	COM(n) ON enables or disables trapping of communications activity to the specified communications adapter. COM(n) STOP
COMMON (S)	COMMON variable[,variable]... passes variables to a chained program
CONT (C)	CONT resumes execution after a break.
COS (F)	v = COS(x) returns the trigonometric cosine function of x
CSNG (F)	v = CSNG(x) Converts x to a single – precision number
CSRLIN (V)	v = CSRLIN returns as a variable the vertical coordinate of the cursor
CVI,CVS,CVD (F)	v = CVI(2 – byte string) converts a string to an integer variable type, while v = CVS(4 – byte string) converts to a single precision type and v = CVD(8 – byte string) converts to a double precision type.
DATA	DATA constant[,constant] stores the numeric and string constants that are accessed by the program's READ statements.
DATE($) (V) & (S)	v$ = DATE$ retrieves the date, while DATE$ = v$ (US format,i.e. mm/dd/yy) sets it.
DEF FN (S)	DEF FNname[(arg[,arg]...)] = expression defines and names a function that you write
DEF SEG (S)	DEF SEG[= address] defines the current "segment" of storage. Subsequent BLOAD,BSAVE,CALL,PEEK,POKE or USR will define the actual physical address of its operation as an offset into this segment.
DEFtype (S)	DEFtype letter[– letter][,letter[– letter]].. defines variable types
DEF USR (S)	DEF USR[n] = offset specifies the starting address of a m/c language subroutine
DELETE (C)	DELETE[line1][– line2] deletes program lines

DIM (S)	DIM variable specifies the maximum values for an array
DRAW (S)	DRAW string draws an object as specified by string.
EDIT (C)	EDIT line displays a line for editing
END (S)	END terminates program execution
EOF (F)	v = EOF(filenum) indicates an end of file condition
ERASE (S)	ERASE arrayname[,arrayname]... eliminates arrays from a program.
ERR and ERL	variables return the error code and associated line number
ERROR (S)	ERROR(n) simulates the occurrence of a BASIC error or allows you to define your own error codes.
EXP (F)	v = EXP(x) calculates the exponential function.
FIELD (S)	FIELD[#]filenum,width AS stringvar[,width AS stringvar]... allocates space for variables in a random file buffer.
FILES (C)	FILES(filespec) displays the names of files on a diskette
FIX (F)	v = FIX(x) truncates x to an integer.
FOR and NEXT	(S) FOR variable = x TO y [STEP z] performs a series of NEXT [variable][,variable] instructions in a loop a given number of times.
FRE (F)	v = FRE(x) returns the number of bytes in memory not being used by BASIC, excluding the size of portion reserve for interpreter.
GET (S)	GET(x1,y1) – (x2,y2),arrayname reads points from an area in the screen.
GOSUB (S)	GOSUB line branches to and
ENTER (S)	ENTER returns from a subroutine.
GOTO (S)	GOTO line branches unconditionally out of a program sequence to a specific line number.
HEX$ (F)	v$ = HEX$(n) returns a string representing the hexadecimal value of a decimal argument.
IF (S)	IF expression[,]THEN clause[ELSE clause] makes a decision regarding program flow based on the result of an expression.
INKEY$ (V)	v$ = INKEY$ reads a character from the keyboard.
INPUT (S)	INPUT [;]["prompt";]variable[,variable] receives input from the keyboard during program execution.
INPUT # (S)	INPUT #filenum,variable[,variable] reads data from a sequential device or file
INPUT$ (F)	v$ = INPUT$(n[#]filenum]) returns a string of n characters from the keyboard or from file number filenum
INSTR (F)	v = INSTR([n,]x$,y$) searches for the 1st occurrence of y$ in x$, starting at optional offset n
INT (F)	v = INT(x) returns largest integer less than or equal to x
KEY (S)	KEY n,x$ sets or displays the soft keys. KEY LIST lists all 10 keys on the screen, with all 15 associated characters. KEY ON displays the keys on line 25, and KEY OFF erases the line 25 display.
KILL (C)	KILL filespec deletes a file from a diskette
LEFT$ (F)	v$ = LEFT$(x$,n) returns the leftmost n characters of x$
LEN (F)	v = LEN(x$) returns the number of characters in x$
LET (S)	[LET] variable = expression assigns the value of an expression to a variable
LINE (S)	LINE[(x1,y1)] – (x2,y2)[,[color][,B[F]]] a line or box on the screen
LINE INPUT (S)	LINE INPUT[;]["prompt";]stringvar reads an entire line from the keyboard into a string variable
LINE INPUT #	LINE INPUT #filenum,stringvar reads an entire line from a sequential file
LIST (C)	LIST[line1][– [line2]][,filespec] lists the program currently in memory
LLIST (C)	LList[line1][– [line2]] lists all or part of the program currently in memory on the printer(LPT1:)
LOAD (C)	LOAD filespec[,R] loads, and optionally runs, a program in memory

LOC (F)	v = LOC(filenum) returns the current position in the file
LOCATE (S)	LOCATE[row][,[col][,[cursor][,[start][,stop]]]] position the cursor
LOF (F)	v = LOF(filenum) returns the number of bytes assigned to the file (length of file)
LOG (F)	v = LOG(x) returns the natural logarithm of x
LPOS (F)	v = LPOS(n) returns the current position of the print head within the printer buffer for LPT1:.
LPRINT (S)	LPRINT[list of expressions][;] prints data on the printer (LPT1:)
LSET and RSET (S)	LSET stringvar = x$ moves data into a random file buffer (in preparation for a PUT (file) statement).
MERGE (C)	MERGE filespec merges the lines from an ASCII program file into the program currently in memory.
MID$ (F), (S)	v$ = MID$(x$n[,m]) returns the requested part of a string. When used as a MID$(v$,n[,m]) = y$ statement, it replaces a portion of one string with another.
NAME (C)	NAME filespec AS filename changes the name of a diskette file
NEW (C)	NEW deletes the program currently in memory and clears all variables.
OCT$ (F)	v$ = OCT$(n) returns a string representing the octal value of the decimal.
ON COM(n) (S)	ON COM(n) GOSUB line sets up line number for BASICA to trap to when there is information coming into the communications buffer.
ON ERROR (S)	ON ERROR GOTO line enables error trapping and specifies 1st. line of the error handling subroutine
ON..GOSUB or GOTO (S)	ON n GOTO line[,line] branches to ohe of several lines depending on the value of an expression.
ON KEY(n) (S)	ON KEY(n) GOSUB line sets up a line number for BASICA to trap when the specified function key is pressed
OPEN (S)	OPEN filespec[FOR mode] AS [#] filenum [LEN = lrecl] allows I/O to a file or device.
OPEN"COM..	OPEN"COMn:optionlist"AS[#]filenum [LEN = number]. This option list can include speed, parity, databits, stopbits (in that order) as well as RS, CS[n], DS[n], CD[n] and LF after them. It opens a communications file
OPTION BASE (S)	OPTION BASE n declares the minimum value for array subscripts
OUT (S)	OUT n,m sends a byte to a m/c output port.
PAINT (S)	PAINT(x,y)[,paint[,boundry]] fills in an area on the screen with the desired colour.
PEEK (F)	v = PEEK(n) returns the byte read from the indicated memory position
PLAY (S)	PLAY string plays music as specified by string
POINT (F)	v = POINT(x,y) returns the colour of the specified point on the screen
POKE (S)	POKEn,m writes a byte into a memory location.
POS (F)	v = POS(n) returns the current cursor column position
PRINT (S)	PRINT[list of expressions][;] displays data on the screen
PRINT USING (S)	PRINT USING v$;list of expressions[;] prints strings using a specified format.
PRINT # [USING]	writes data sequentially to a file.
PSET, PRESET (S)	PSET(x,y)[,colour] draws a point at the specified position on the screen
PUT (S)(Files)	PUT[#]filenum[,number] writes a record from a random buffer to a random file
PUT (S)	(Graphics) PUT(x,y),array[,action] writes colours onto a specific area of the screen
RANDOMIZE (S)	RANDOMIZE[n] reseeds the random number generator.
READ (S)	reads values from a DATA statement and assigns them to variables.
REM (S)	REM remark inserts explanatory remarks into a program
RENUM (C)	RENUM [newnum][,[oldnum][,increment]] renumbers program lines
RESET (C)	RESET closes all diskette files and clears the system buffer

RESTORE (S)	RESTORE line allows DATA statements to be reread from a specified line
RESUME (S)	RESUME or RESUME[0] continues program execution after an error recovery procedure has been performed, restarting at the same line.. RESUME NEXT and RESUME line will restart at the next, or specified, line.
ENTER (S)	ENTER[line] to bring you back from a subroutine.
RIGHT$ (F)	v$ = RIGHT$(x$,n) returns the rightmost n characters from string x$
RND (F)	v = RND[(x)] returns a random number between 0 and 1
RUN (C)	RUN[line] begins execution of a program as does RUN filespec[,R]
SAVE (C)	SAVEfilespec[,A] saves a BASICA program file on a diskette (SAVEfilespec[,P] A = ASCII format, P = Encoded binary format).
SCREEN (F)	v = SCREEN(row,col[,z]) returns the ASCII code for the character on the active screen at the indicated position.
SCREEN (S)	SCREEN[mode][,burst][,[apage][,vpage]]] sets the screen attributes to be used by subsequent statements
SGN (F)	v = SGN(x) returns the sign of x
SIN (F)	v = SIN(x) returns the trigonometric sine function
SOUND (S)	SOUND freq,duration generates sound through the speaker.
SPACE$ (F)	v$ = SPACE$(n) returns a string consisting of n spaces
SPC (F)	PRINT SPC(n) skips n spaces in a print statement
SQR (F)	v = SQR(x) returns the square root of x
STOP (S)	STOP terminates program execution and returns to command level.
STR$ (F)	v$ = STR$(x) returns a string representation of the value of x
STRING$ (F)	v$ = STRING$(n,m) returns a string of length n whose characters all have the v$ = STRING$(n,x$) ASCII value m or the 1st. character of x$.
SWAP (S)	SWAP variable1,variable2 swaps the values of two variables
SYSTEM (C)	SYSTEM exits from BASICA and returns to DOS
TAB (F)	PRINT TAB(n) tabs to position n
TAN (F)	v = TAN(x) returns the trigonometric tangent of x
TIME$ (V) and (S)	v$ = TIME$ sets or retrieves the current time (TIME$ = v$).
TRON, TROFF (C)	TRON traces the execution of program statements and TROFF stops this
USR (F)	v = USR[n](arg) calls the indicated machine language program with the argument arg
VAL (F)	v = VAL(x$) returns the numerical value of string, x$
VARPTR (F)	v = VARPTR(variable) returns the address in memory of the variable or file control block
WAIT (S)	WAIT port,n[,m] suspends program execution while monitoring the status of a machine input port
WHILE, WEND (S)	WHILE expression executes a series of statements in a loop as long as a given condition is true
WIDTH (S)	WIDTH size sets the output line width in number of characters
WRITE (S)	WRITE[list of expressions] outputs data on the screen
WRITE #	WRITE #filenum,list of expressions writes data to a sequential file

7. COMMUNICATING WITH SERIAL DEVICES

Let us now look at how to communicate with other computers or devices using an asynchronous adapters. Such an adapter provides RS–232C I/O possibilities. Two such adapters, referred to as the primary and (if installed) the secondary adapters, may be installed on an IBM PC. DOS 3.x commands refer to them as COM1: and COM2: respectively. Each connects to the outside world via a standard 25–pin connector (PC–1, XT) or a 9 pin connector and adapter lead, fitted with a similar 25 pin connector (AT). The pins are identified as follows:

1 – Chassis Ground.
2 – Transmit Data
3 – Receive Data
4 – Request to Send
5 – Clear to Send
6 – Data Set Ready
7 – Signal Ground
8 – Carrier Detect
9 to 18 are less frequently used
19 – Ready
20 – Data Terminal Ready
21 to 25 are rather infrequently used.

The most commonly used are 1 – 8 and 20. Some manufacturers try to simplify their lives by using 'spare' pins for their own non standard purposes, occasionally for example to supply power to a device. This can lead to difficulties. The best way to avoid problems is to make use of as few pins as possible and keep the interchange of control information as basic as possible. For example, to send data to a serial peripheral (terminal, etc.) you require only:

PC		Terminal or peripheral
1	–	1 (chassis ground)
2	–	3 (transmit data to receive data)
7	–	7 (signal ground.. not really needed!)

Then, you need to create a small program. Here is an example using BASICA to open a communications file, at 4800 baud, with no parity checking, 8 data bits and 2 stop bits as file #1 to address the primary asynchronous adapter, COM1: .

```
10 CLOSE #1
20 OPEN "COM1:4800,N,8,2" AS #1
30 INPUT "Please type a line, or END",B$
40 IF B$ = "" THEN GOTO 30
50 IF B$ = "END" THEN GOTO 100
60 IF B$ = "end" THEN GOTO 100
70 PRINT #1,B$
80 GOTO 30
90 '
100 CLOSE #1
110 SYSTEM
```

Note that we wait for and entire line to be entered (INPUT waits for the Enter key to be typed) before anything is sent to the terminal by the PRINT function. We allow the program to stop by looking for the text END being typed (but we are careful to allow the user to type it in lower case as well!). However, there is no possibility of replying to the PC from the terminal. This requires a 'return' path for data to flow from the peripheral to the PC:

PC		Terminal or peripheral
1	–	1 (chassis ground)
2	–	3 (transmit data to receive data)
3	–	2 (receive data to transmit data)
7	–	7 (signal ground.. not really needed!)

You can now perform a simple dialogue, for example using two PCs connected as above, each running the following BASICA program. Already you can see that it is more elaborate:

```
 10 OPEN "COM1:4800,N,8,2" AS #1
 20 CR$ = CHR$(13)
 30 LF$ = CHR$(10)
 40 SEND$ = ""
 50 K$ = INKEY$
 60 IF K$ = "" THEN GOTO 50
 70 IF ASC(K$) > 31 THEN PRINT K$;
 80 SEND$ = SEND$ + K$
 90 IF K$ < > CR$ THEN GOTO 50
100 PRINT #1,SEND$ + CR$ + LF$
110 SEND$ = ""
120 EDITED$ = ""
130 WHILE NOT EOF(1)
140 REPLY$ = INPUT$(LOC(1),#1)
150 LBIT% = LEN(REPLY$)
160 IF LBIT% = 0 GOTO 230
170 FOR N% = 1 TO LBIT%
180 CH = ASC(MID$(REPLY$,N%))
190 IF CH > 127 THEN CH = CH - 128
200 IF CH < > 0 AND CH < > 22 THEN EDITED$ = EDITED$ + CHR$(CH)
210 NEXT N%
220 PRINT EDITED$;
230 WEND
240 GOTO 40
```

To connect to other types of serial peripherals, you may need a 'null modem' crossover cable with the following connections: 1 – 1, 2 – 3, 3 – 2, 4 – 5, 5 – 4, 6 – 20, 20 – 6, 7 – 7. Without much difficulty you could use a modified version of the last program to connect a PC to another host machine. One of the drawbacks of running such programs when connecting to a host is that they are somewhat slow and under normal conditions can easily overflow the standard communications buffers. By using the BASICA /C:20000 option, some of the problems disappear but for greater performance it is really necessary to compile programs. This can improve speed by a factor of 10 or more. Another problem with such programs is that, while offering a certain dialogue, they may be extremely limited in their actual terminal emulation, may not allow file transfer, etc. To perform all of these functions you normally need to use professional network software (PC NET, DECNET DOS, TCP/IP, etc.).

8. PC NETWORKING

Any PC user who has the necessary adapters and software can try to use one of the following networks: PC Network, IBM Token Ring, DECNET or Ethernet/Cheapernet. Generally, such networks allow the sharing or using of relatively expensive resources (large disks, cartridge backup devices, printers), transmission of messages, sharing files, remote login to host machines running VM (IBM) or VMS (DEC). **The features you might consider before choosing your networking hardware/software** are:

1. **Is the equipment and software easy to obtain?**
2. **Is the equipment and software easy to install? Does it conflict with other equipment?**
3. **What sort of 'services' can the user expect, if any?**

PC Network

This is the network that we still use for most of our own IBM PCs. From 1 to 72 machines may be connected to a PC Network. Machines may (depending on the 'kit' they are connected to) be from 3 to 300 metres form the Network Translator Unit. To build such a network you require:

PC Network Translator Unit
PC Network Base expander (if > 8 PCs)
PC Network short/medium/long distance kits (60/120/240 m to 1 – 8 PCs)
PC Network 15/30/60 m cables as needed
DOS 3.0 or higher for each PC (3.2 is current)
PC Network control program (1.0 is current)
PC Network Adapter per PC

The cable is RG – 11 coaxial cable, fitted with 'F' type connectors. It is rather clumsy but can be fitted into the telephone ducts, etc. The network allows a machine configured as a 'server' (NET START SRV..) to share its disks, files and printers with other machines and to send and receive messages. This may use up to 360 Kb of memory. A machine that is fitted only with diskette drives may instead be configured as a 'redirector' and use the shared disk of a server. This is more economical with memory needed only 128 Kb (NET START RDR...).

The user can add or remove machines at any time provided they are not 'servers', i.e. machines which are sharing their disks or printers. It is not possible to 'inquire' on the network to see what other machines exist,so one person normally allocates machines names, etc. Documentation is clear and complete. Hardware is easy to install. **However, I suggest that you do not use the IBM supplied installation procedure. Instead, just copy all the files on the PC Network Control program diskette to your root directory:**

```
COPY A:*.* C:/*.*
```

You should add the following statements in your CONFIG.SYS file, which would be sufficient for a server or redirector:

```
FILES = 20
BUFFERS = 20
FCBS = 16,8
LASTDRIVE = E
```

You should add the following statements to your AUTOEXEC.BAT to become a server called MST, sharing your directory MSTSYS for read only and MSTPUB for read/write. You might for example keep all of DOS 3.x in MSTSYS, sharing it for others to use as a 'system' disk, while allowing use of MSTPUB for a user's other files:

```
NET START SRV MST /NBC:4 /NBS:8K
NET SHARE MSTSYS = C:\MSTSYS /R
NET SHARE MSTPUB = C:\MSTPUB /RWC
```

You could add the following statements to your AUTOEXEC.BAT of another machine on this network, to become a redirector called FORTRAN, using the above shared directories of MST as if they were 'disks'. The same statements as for the server above are used in CONFIG.SYS. It is assumed that you have only diskette drives, A and B:

```
NET START RDR FORTRAN /NBC:4 /NBS:8K
NET USE C: /D
NET USE C: \\MST\MSTSYS
NET USE D: /D
NET USE D: \\MST\MSTPUB
C:
```

IBM Token Ring

Many more than 72 machines may be connected to a Token Ring. Machines may (depending on the configuration of the equipment) be from 3 to 1000 metres from each other. To build such a network you require:

Token Ring Network PC adapter cable 5160 9088
DOS 3.2 or higher for each PC (3.2 is current)
PC Network control program (1.1 is current)
Token Ring Network Adapter per PC 5160 9100
NETBIOS software for each PC

The cabling is two twisted pairs. It is again rather clumsy but can be fitted into the telephone ducts, etc. The IBM Token Ring uses the same PC Network control program as above. A machine may be configured as a server or redirector just as above. It is, as above, not possible to 'inquire' on the network. One person normally allocates machines names, etc. Documentation is clear and complete. Hardware is easy to install. **However, I suggest that you do not use the IBM supplied installation procedure. Instead, just copy all the files on the PC Network Control program diskette and the NETBIOS program diskette (61X3825) to your root directory:**

```
COPY A:*.* C:/*.*
```

PC Network and Token Ring can coexist in a machine running as a dedicated 'bridge' (5669 – 041, Token Ring – PC Network Interconnect software). You can connect Token Ring to VM but this seems to require another dedicated server. You then may run 3278 emulation software on other PCs on the Token Ring. Similar changes are needed in CONFIG.SYS and AUTOEXEC.BAT. They should contain the following statements:

```
FILES = 20
BUFFERS = 20
FCBS = 16,8
LASTDRIVE = E
STACKS = 64,128

TOKREUI
NETBEUI
NET START SRV MST /NBC:4 /NBS:8K
NET SHARE MSTSYS=C:\MSTSYS /R
NET SHARE MSTPUB=C:\MSTPUB /RWC
```

DECNET

This is the network that DEC supports. Both hardware and software are readily obtained. Very many machines are already connected to DECNET. Machines may (depending on the configuration of the equipment) be at almost any distance from each other. To connect a PC to this network you require:

3-Com IE4 (1221) or IE2 (34-0780) Ethernet adapter
or
MICOM Interlan NI5010-1 (rev XD) or NI5010-2 adapter

DECNET DOS 1.0 or higher
DELNI port or Ethernet transceiver

The cabling is by 50 ohm coax in the case of the internal transceiver or by 15-pin cable if connecting to for example a DELNI. The adapter can conflict with the 'alternate' RS232 adapter. The software is easy to install, as is the hardware. DECNET DOS allows connected machines to share disks, files and printers and to send and receive messages. VT100 emulation is provided. The user has to obtain a valid DECNET node number and/or name. Care should be taken in connecting a machine to a public Ethernet; the situation is simpler for a DELNI. It is possible to 'inquire' on the network. You can use NFT (Network File Transfer) or SETHOST (login to VMS) etc. PC Network and DECNET DOS can coexist in a single non-dedicated machine. Documentation is clear and complete.

To install the software, I suggest that you copy all the diskettes to a new directory C:\DECNET (there are rather too many to conveniently put in the root directory, and ensure that the following statements are in your CONFIG.SYS and AUTOEXEC.BAT files:

```
FILES=20
BUFFERS=20
FCBS=16,8
DEVICE=C:\DECNET\NDDRV.SYS
DEVICE=C:\DECNET\NPDRV.SYS

PATH C:\;C\DECNET
CD C:\DECNET
SCH
DLL
DNP
NDU OPEN NODE rvax NDISK a$b:[dir]fname.ext;1 USER name/pwd DRIVE D
CD C:\
```

There is a configuration program, DIP, that you can run. It will select the software components you need (perhaps changing the names a little). SCH is the real tme scheduler, DLL the Data Link Layer process and DNP the DECNET DOS Network process. Other components that might be selected are TTT, Transparent Task to Task, TFA, Transparent File Access, and FAL, File Access Listener.

The NDU OPEN statement is useful if you have defined a remote disk on a VAX 'rvax', to be found in a$b:[dir]fname.ext;1 belonging to USER name/pwd. It will be drive D on the PC. To create such a disk, you need to call NDU (Network Device Utility) and give the command:

```
NDU>CREATE DRIVE D NODE rvax NDISK a$b:[dir]fname.ext;1 USER name/pwd
```

Once created, such a drive may be accessed each if an AUTOEXEC.BAT like the above is called to define it. DECNET DOS supports a wide variety of DECNET functions. You can connect a terminal session (VT100) to a remote machine 'rvax' with:

```
SETHOST rvax
```

You may transfer files to and from a remote machine 'rvax' by using NFT, Network File Transfer. To send a PC file 'fname.ext' to a remote VAX, 'rvax', to a directory a$b:[dir] assuming your login name and password are 'name pwd', you could call:

```
NFT COPY fname.ext rvax"name pwd"::a$b:[dir]
```

Most important DECNET parameters are kept (in the case that something goes wrong) in a file NCP.TXT, which the Network Control Program, NCP, can consult. One of ours for a server machine alos connected to DECNET contained in editable text the following:

```
SET ECHO ON
SET EXECUTOR IDENTIFICATION "DECnet-DIS V1.1,PC-DOS V3.10"
DEFINE EXECUTOR MAXIMUM LINKS 2
DEFINE EXECUTOR MAXIMUM BUFFERS 12
DEFINE EXECUTOR NAME PCOPS1 ADDRESS 12.34
SET EXECUTOR NAME PCOPS1 ADDRESS 12.34
SET EXECUTOR STATE ON
SET NODE 12.34 NAME PCOPS1 USER "TECHSUPPORT"
SET NODE 12.35 NAME VAX1
SET NODE 12.99 NAME CERNVM
DEFINE LINE STATE ON
SET LINE STATE ON
EXIT
```

Cheapernet/TCP-IP from FTP Software Inc.

This combination of hardware and software is becoming quite popular. Both hardware and software are readily obtained. Very many machines are already connected to networks supporting TCP/IP. Machines may (depending on the configuration of the network) be at almost any distance from each other. To connect a PC to this network you require:

3-COM Ethernet or BICC 4201 adapter, etc
TCP/IP for PCs (FTP Software Inc. etc)
DELNI port or Ethernet transceiver, etc.

The cabling is by 50 ohm coax in the case of the internal transceiver or by 15-pin cable if connecting to for example a DELNI. The 3-COM adapter can conflict with the 'alternate' RS232 adapter. The software is easy to install, as is the hardware. TCP/IP allows connected machines to share disks, files and printers and to send and receive messages, etc. However, a 'server' which shares resources must be a dedicated machine. VT100 emulation is provided. The user has to obtain a valid network node number and/or name. Care should be taken in connecting a machine to a public network. It is possible to 'inquire' on the network. You can use FTP (File Transfer Protocol) or TNVT100, TN3270 (login to VMS, VM) etc. Documentation is in many cases obscure, but for skilled interpreters it is quite complete.

To install the software, I suggest that you copy all the diskettes to a new directory C:\TCP (there are once again rather too many to conveniently put in the root directory) and ensure that the following statements are in your CONFIG.SYS and AUTOEXEC.BAT files:

```
FILES=20
BUFFERS=20
```

```
FCBS=16,8
DEVICE=C:\TCP\3C501.SYS
DEVICE=C:\TCP\IPCUST.SYS

PATH C:\;C\TCP
CD C:\TCP
TNVT100 rvax (etc.)
```

No changes are needed in the AUTOEXEC.BAT file. You can call upon the services of TCP/IP when you want them. To obtain or send a file from a VAX or VM host, you would call FTP. In reply to the subsequent prompts, you should select VM or VMS:

```
GET fname.fm   (for a VM file on the 'A' disk)
or
CD vmnam.234
GET fname.fm   (for a VM file on disk 234 of user 'vmnam')

GET fname.fm   (for a VMS file in current VMS directory)
or
CD newdir
GET fname.fm   (for a VMS file in VMS directory newdir)
```

To send a file, you can do this easily to a VMS system. For VM, as you might expect, things are more awkward. You need to define a read and write password for each disk you wish to write to. This should be the same as you LOGIN password, or you will need to use the ACCT option of FTP to set this.

```
PUT fname.fm   (for a VM file to the current disk)
PUT fname.fm   (for a VMS file in current VMS directory)
```

9. ASCII CHARACTER SET USED BY PCS

PC uses ASCII for their internal character set. The BASICA user can easily move back and forward between the numerical representation of a character and a string representation by using the functions CHR$(number) and ASC(string). The numerical values of the characters are as follows (for 0–127, the non–graphic ones):

Value		Character	
Decimal	Hex		
0	0	null	
1	1	SOH	
2	2	STX	
3	3	ETX	
4	4	EOT	
5	5	ENQ	
6	6	ACK	
7	7	BEL	
8	8	BS	(backspace)
9	9	HT	(horizontal tab)
10	A	LF	(line feed)
11	B	VT	(home)
12	C	FF	(form feed)
13	D	CR	(carriage return)
14	E	SO	
15	F	SI	
16	10	DLE	
17	11	DC1	
18	12	DC2	
19	13	DC3	
20	14	DC4	
21	15	NAK	
22	16	SYN	
23	17	ETB	
24	18	CAN	
25	19	EM	
26	1A	SUB	
27	1B	ESC	
28	1C	FS	(cursor right on PC)
29	1D	GS	(cursor left on PC)
30	1E	RS	(cursor up on PC)
31	1F	US	(cursor down on PC)
32	20	space	
33	21	!	
34	22	"	(quotes)
35	23		(numeric sign)
36	24	$	
37	25	%	
38	26	&	
39	27	'	(apostrophe)
40	28	(	
41	29	)	
42	2A	*	
43	2B	+	(plus)
44	2C	,	(comma)

45	2D	−	(minus)
46	2E	.	(full stop)
47	2F	/	
48 − 57	30 − 39	0 to 9	
58	3A	:	(colon)
59	3B	;	(semicolon)
60	3C	<	
61	3D	=	
62	3E	>	
63	3F	?	
64	40	@	
65 − 90	41 − 59	A − Z	
91	5B	[	
92	5C	\	
93	5D	]	
94	5E	-	(circumflex)
95	5F	_	(underscore)
96	60	,	(grave accent)
97 − 122	61 − 7A	a − z	
123	7B	{	
124	7C	|	(vertical bar)
125	7D	}	
126	7E	¬	(tilde)
127	7F		

INDEX

* character in a filename.ext ... 7
* character in commands ... 9
* prompt from EDLIN ... 26

? character in a filename.ext ... 7
? character in commands ... 9

ASC function ... 39
ASCII files from BASICA ... 27
ASSIGN drive requests to another drive ... 9
asynchronous adapters ... 32
AUTOEXEC.BAT file, sample one ... 2
automatic response file for LINK ... 16
AUX alternate name for COM1 ... 6

BASICA as an editor ... 26
batch processing in DOS ... 10

CD or CHDIR change directory command ... 7
changing the default directory ... 10
CHDIR or CD change directory command ... 7
CHKDSK command in DOS ... 11
CHR$ function ... 39
COMMAND needed to chain .BATs ... 10
COMP command in DOS ... 11
compiling BASIC ... 23
compressed BASIC, avoiding of ... 27
COM1 primary serial/parallel port ... 6
COM1: and COM2: cards ... 32
COM2 alternate serial/parallel port ... 6
CON as the keyboard/screen ... 6
CONFIG.SYS file, sample ... 2
creating a file ... 26
Ctrl − ALT − DEL key combination in DOS ... 6
Ctrl − break key combination in DOS ... 6
Ctrl − Enter key combination in DOS ... 6
Ctrl − Numlock key combination in DOS ... 6
Ctrl − PrtScr key combination in DOS ... 6

DATE command in DOS ... 12
diagnostics diskette ... 11
DIR command in DOS ... 11, 13
DISKCOMP command in DOS ... 13
DISKCOPY command in DOS ... 13

editing a file ... 26
EDLIN editor ... 26
Enter key in DOS ... 6
ESC key in DOS ... 6
EXE2BIN command in DOS ... 13

FCBS command, for network CONFIG.SYS ... 4
FDISK to set up your DOS 3.x partition ... 2
FORMAT command in DOS ... 14

LABEL command of DOS 3.x ... 2
LASTDRIVE command, for network CONFIG.SYS ... 4
leftarrow key in DOS ... 6
LINK command in DOS ... 16
LPT1 as first parallel printer ... 6
LPT2,LPT3 as second, third parallel printers ... 6

NUL, dummy output device ... 6

PATH command to find EXE or COM files ... 7
PAUSE batch subcommand under DOS ... 18
PRN as first parallel printer ... 6

REM batch subcommand under DOS ... 19
RENAME command under DOS ... 19
RENUM command of BASIC ... 27
renumbering lines with the RENUM command of BASIC ... 27

SAVE option A for ASCII in BASICA ... 27
saving a file ... 26
set or remove the read − only attribute ... 9
SHELL command ... 4
shift − PrtScr key combination in DOS ... 6
SYS command of DOS 3.x ... 2
SYS command under DOS ... 21

TIME command under DOS ... 21
TYPE command under DOS ... 21

VDISK command, useful for AT ... 4

YNPROMPT example ... 5