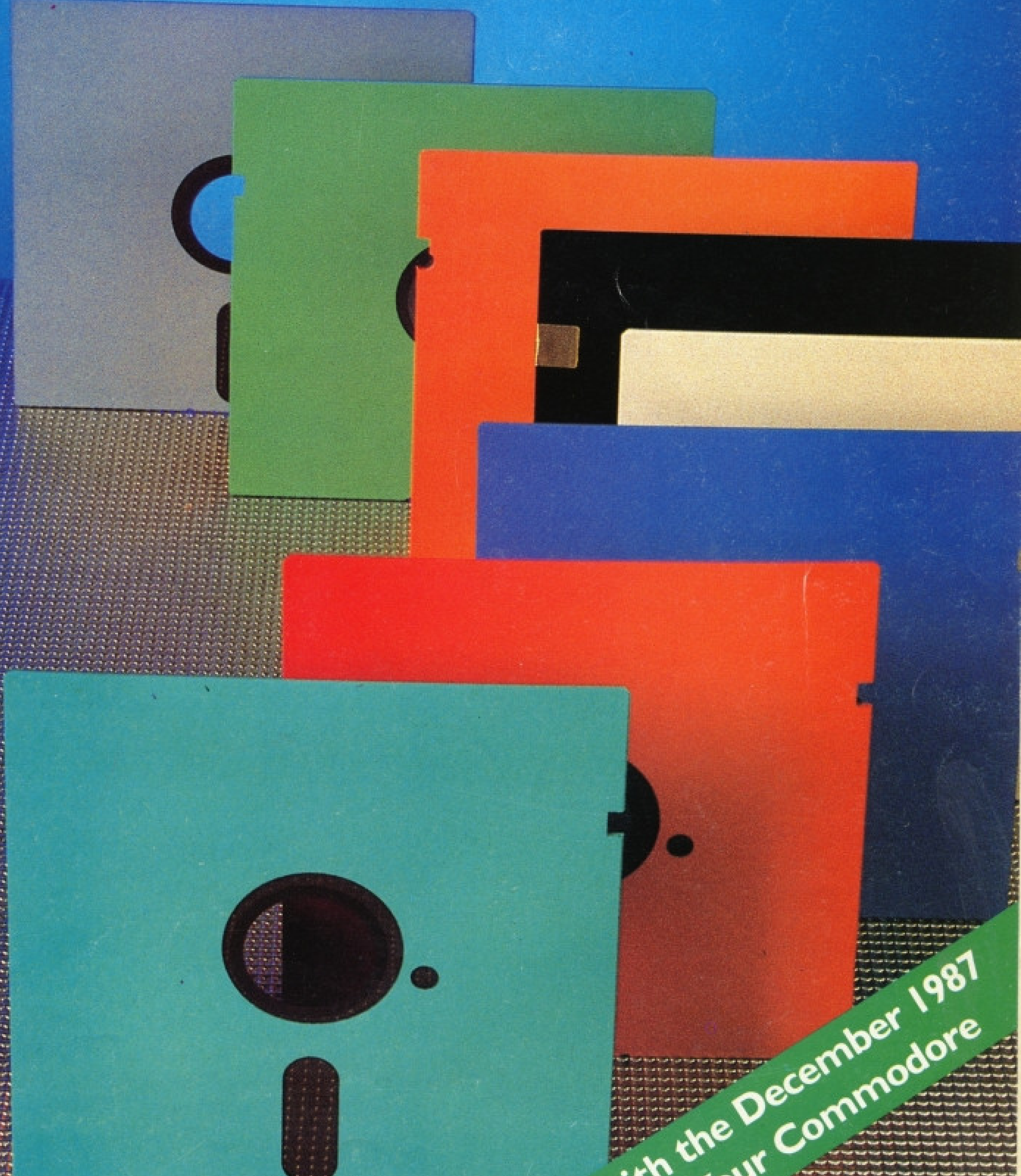
YOUR COMMODORE 1987

DISK USERS HANDBOOK

BEGINNERS GUIDE TO DISKS

PROGRAMMING WITH YOUR DISK DRIVE

USING A DISK EDITOR



FAST LOADER FOR 1541

C64 DISK EDITOR

PRINT YOUR OWN DISK SLEEVES

Free with the December 1987 issue of Your Commodore

Your Commodore Proudly Presents

On Sale 20th October 1987

Available from newsagents and by mail-order from
Infonet Ltd, 5 River Park Estate, Berkhamsted, Herts HP4 1HL. Tel: 04427 76661/4 (£1.50 plus 50p p&p.)

Editor: Stuart Cooke
Assistant Editor: Sue Joyce
Editorial Assistant: Kirk Rutter
Advertisement Manager: Stuart Taylor
Advertisement Copy Control: Laura Champion
Origination: Ebony Typesetting
Design: Argus Design Studio

Your Commodore incorporating Your 64 is a monthly magazine appearing on the first Friday of each month. Your Amiga is published every second month within the pages of Your Commodore. Argus Specialist Publications Limited Editorial & Advertisement Office, Your Commodore, No 1 Golden Square, London W1R 3AB. Telephone: 01-437 0626 Telex: 8811896.
Subscription rates upon application to Your Commodore Subscriptions Department, Infonet Ltd, 5 River Park Estate, Berkhamsted, Herts, HP4 1HL.

ARGUS PRESS GROUP

CONTENTS

The Your Commodore *Disk Users Handbook is packed full of vital information and programs for owners, and potential buyers, of all Commodore disk drives.*

If you are new to your disk drive then our beginners article will supplement the manual and help you discover the joys of using a disk drive. If you are a more advanced user, the article on disk commands will expand your knowledge so that you can talk directly to the drive. Learn how to read a directory from within Basic and much more.

For those readers wanting to go even further with their disk drive we give a detailed description of the disk structure and details of how to use this information with a disk editor, including how to resurrect scratched files.

The programs

As well as the articles already mentioned, this supplement also holds a variety of useful programs. Owners of the Commodore 1541 disk drive, which has often been described as a lumbering hippo, can speed loading up with our 1541 Fast Loader.

Should your appetite be whetted by our Disk Editing article then you can type in our Track/Sector Editor and give it a whirl yourself.

Should you have a large number of programs in your disk collection then our last two programs will be invaluable. The first, DIR cover, will produce your own disk library sleeves on your printer. The program is totally in Basic and can therefore easily be altered to suit any printer.

The second program is a C64 menu generator. This will place a menu of selected files on your disk. You can then use this menu to load any of the selected programs with ease.

All in all, whatever your technical ability, the Your Commodore *Disk Users Handbook will provide you with something to suit your needs.*

Beginners Start Here

If you're the proud owner of a disk drive, you'll already be discovering the benefits it can bring to your system. Read on for more info.

By Tony Hetherington

Congratulations!, if you've just bought a disk drive, then you're at last free of the frustration of waiting 15-20 minutes for a game to load. At last you can load and save your own programmes in seconds and can access the huge library of disk-based software. If you haven't already added a disk drive to your system but are wondering whether it would be worthwhile, then read on as we delve into the delights that lie ahead.

Apart from the considerable reductions in loading time, which is worthwhile on its own, a disk based system means you can now use bigger programs as although they can't use the whole of the disk storage space (about 170K) at the same time information can be loaded in as and when required. If you don't think 170K is enough you could run up to four disk drives from your C64 at any one time or store your data on more than one disk.

Finally, a disk based system is a lot more flexible than a cassette because as there is two-way communication between your C64 and the disk drive, any piece of information on a disk can be quickly read, altered and rewritten in a few seconds. This is why nearly all business software such as word processors, databases and spreadsheets are disk based.

What is a disk?

A disk is a flat disc of magnetic material made from a thicker version of the material used to make cassette tpes. The disk is then sandwiched between two sheets of special material that gently cleans any dirt off, as it spins in the disk drive. This is then sealed in a plastic cover to protect it from scratching, and any dirt, grime and grease that could damage the disk by handling it.

The plastic sleeve has several cutout sections which allows the disk drives head to read the information on the disk, slots to guide the disk to the coorrect place in the drive and a write-protect notch or hole. The drive senses the hole and allows new information to be written to the disk. Since this can mean writing over important data you can 'write protect' a disk by sticking a lable over the notch. This tell the drive to stop any commands that would write on the disk. Most blank disks are supplied with a sheet of write protect labels.

It's worth taking care of your disks as damaging part of one could ruin the whole disk. After all, 170k of data is a lot to lose! The following tips are worth following as they could save you a lot of time and spare you a lot of inconvenience.

1 Only touch the plastic sleeves and handle disks gently at all times. A bent disk is a ruined disk.
2 When you're not using a disk keep it stored in its cardboard (or heavy paper) sleeve and preferably in a plastic disk box.
3 Keep disks away from bright sunlight, cigarette smoke, coffee, dust, telephones, monitors and the top of the disk drive or other sources of magnetic fields.
4 Don't take a disk out of the drive when the red light is on as this means the drive is reading or writing information and could cause you horrendous problems.
5 Always ensure that you take disks out of your drive before you switch it off as you run the risk of losing everything on it.

Disks are supplied in a variety of forms and are labelled to show the amount of information that can be stored on them. All disks are manufactured to be 'double dided, double density' and are then tested for quality. If they fail these stringent quality control tests, they are then down graded to single-sided, double-density or double-sided single-density. The Commodore 1541 disk drive only requires single sided, single density (SS,SD) disks which means you don't have to waste money on extra quality you won't need.

As mentioned before all disks are originally manufactured to the double-sided and so you can buy a small device known as a disk notcher (for around £5) that will cut a second write protect notch into the disk so you can then use the other side! Obviously, there are no guarantees that this extra side will always read and write data perfectly but knowing it can be used is useful to know.

When you buy a disk it is a completely blank disk of magnetic material and so must be prepared for use with your C64. This is required since the same disk could have been bought by someone to use in a IBM or an Atari computer. Therefore, the first thing you must do is prepare or format it for use.

To format a blank disk ... place it in the disk drive and shut the door then type in the following command.

```
OPEN1,8,15,"N0:diskname,ID"
```

This command tells the processor inside the disk drive to open communications channel 1 (this can be any of 15) to device number eight (the disk drive). The 15 tells the drive that the rest of the command is an instruction for the whole disk and tells it to format the disk and give it the name diskname which is followed by the disk ID. The ID is a two letter or number identity code to distinguish the disk from other disks with the same name. For example, you could name a whole series of disks Tony,01, Tony,02, Tony,03 and so on. Therefore the command to name a disk Tony,01 would be..

```
OPEN1,8,15,"N0:Tony,01"
```

This should then be followed by CLOSE1 to close the number 1 command channel.

When this command is entered, the disk drive will whirr into action and the red light will flash on and off. This will take a few minutes as the drive has a lot to do. First of all, it creates 35 circular tracks on the disk and divides these into sectors or blocks. Because the circumference of a disk is wider at the outside than the inside there are more blocks on the outer than the inner tracks.

Each block can contain 256 characters of information although the first two characters are used by the drive to point to the next block where information is stored. Once each block has been created the drive tests it and then finally adds a directory in the centre of the disk which contains a list of all files or programmes stored on the disk and a Block Availability Map which helps the drive sllot new information into empty blocks.

When this process is completed the drive will stop and the disk will be ready to use.

As you might imagine, formatting a disk wipes all information that was stored on the disk so you should be careful that you don't format any disks that contain information you still need and NEVER format a disk containing a commercial program.

Device Numbers

The format and Load and Save commands include the device number 8. This tells the C64 which input or output device the information should be read or written to. The C64 uses the following device numbers.

1 – datasette
2 – keyboard (input only)
3 – Screen
4to7 – Printers (usually 4)
8to11 – disk drives.

Most people will only use one disk drive which is automatically set to device number 8. However, if you have a second drive (or third and fourth) and want to use it at the same time you will have to give it another device number (usually 9).

You can do this in two ways either by altering the hardware or a simpler way is to type in and run the program in the manual.

Loading and Saving

Now you have prepared or formatted a disk for use or have bought a commercial program you will want to load and save programs.

To load a program simply type...

```
LOAD"name",8
```

which loads the name program into memory. Then type RUN to start it.
OR type

```
Load "*",8
```

which loads the first program on the disk into memory.
OR

```
LOAD"*",8,1
```

loads the first program on the disk into the same memory locations it was saved from. This is the command you will use most for commercial programs which usually start automatically.
OR

```
LOAD"0:*",8,1
```

This ensures that the first program on the disk is loaded in. Occasionally Load"*",8,1, if used a few times will load in the next program on disk.

The command Load "$",8 loads in the disk directory that can be displayed by typing LIST which shows all the files that are stored on the disk.

To SAVE a program simply type

```
SAVE"0:name",8.
```

You will only need to use this if you're going to write and use your own programs as commercial programs have their own save routines but you will still need to ensure that you have a formatted disk ready for use.

The asterik (*) which can be used in loading commands replaces any number of characters. The command LOAD"*",8,1 loads in the first program on the disk as the * replaces the filename. You can also use * to save your typing finger and load in files further down the directory. For example, if there was a program called HOW TO USE THIS you could load it in by simply typing the command LOAD"HOW*",8,1 as long as there wasn't another program listed above this one in the directory called HOW I WON.

File Types

As mentioned above, typing LOAD"$",8 then LIST displays the disk directory on screen. As you can see from these examples there are four different types of disk file.

The program file which appears as PRG in the directory listing is probably the most common file that you will come across. A program file is exactly what its name suggest, a program that you have scored on disk.

The program file is stored on disk in exactly the same forat as it would be in the computers memory, i.e. it is tokenised.

A sequential file (SEQ in the directory listing) is essentially a file that contains a continuous string of characters. A sequential file could for example, be set up to contain data for a database. Let's say that we had two names in our database – Fred Bloggs and John Smith. In a sequential file the data would be stored as:
John Smith Fred Bloggs.

In other words as a continuous list. The problem with using this type of file to store data is that if you required, say the 50th entry of a database, the previous 49 entries would all have to be loaded in. This makes access to your data very slow.

A much better type of file to use for data storage is a relative (REL) file. this type of file allows you to select a specific record, delete a specific record. In other words you can access the information that you require from the file without having to read lots of unwanted data into memory. Perhaps the least used type of file is the user (USR) file. This is really just like a sequential file and is used in the same way.

Housekeeping

As you save and load files to a disk it will rapidly fill up with things you no longer need. The following commands allow you to tidy up your disks and so save disk space.

New

The new command will look familiar as one of its forms is the same as the format command.

OPEN1,8,15,"N0!diskname.ID"
(The N is short for NEW). This wipes the disk and marks out the sectors and tracks.

If the disk has been used before you can shorten this by leaving out the ID. This may bot sound a lot but the process is shorter as the drive doesn't have to recreate each block.

Initialise

If you are writing your own programmes and want to use a second disk then you must use the initialise command to tell the drive that you've swapped disks and instruct it to read in the new BAM.

Typing OPEN1,8,15,"I initialises the new disk ready for use.

Scratch

If you find you've a program on disk that you no longer need, such as an earlier version of an existing program then typing

OPEN 1,8,15,"S0:filename"will delete it.

You could use the * to delete everything, but when you're deleting files I find it best to type out the full name as you're less likely to make a mistake and delete a file you desperately needed.

Validate

Once you've saved and deleted a few files the blocks of each file will be spread about the disk. This won't stop the files from being read or written but it will slow down the process as the drive head must move over each block.

Typing OPEN1,8,15,"V0" will start the spring cleaning process. This can take some time but will be worth it as you'll be surprised at the saving in loading times.

Disk Commands Summary

The following commands are entered through the command channel. For example OPEN1,8,15 followed by...

NEW/Format – "N0:diskname,ID"
Scratch – "S0:filename"
Initialise – "I"
Validate – "V"

Disk Software

Now you have a disk drive you can use a variety of disk-based business packages, games and utilities.

A disk drive is essential if you plan to use one of the many business packages on the market. Although there are one or two tape based word processors, their disk based counterparts offer far more facilities such as different typefaces or fonts, the ability to include graphics in your text, quick loading and saving and even a spelling checker to correct any mistakes.

As well as create, merge, alter, load and save text files you can get your figures and budgets right with a spreadsheet or store thousands or records with a database program such as Superbase 64.

GEOS (Berkley Softworks now available through Microprose) brings icons and pull down menus to your C64 as well as a whole new disk operating system. With GEOS you dont't have to type in long commands you simply point at an icon instead. There is also a special GEOS word processor, a spreadsheet and database programmes.

If you want to be entertained then why not try a disk based game. These are often extended versios of the cassette gaes but have added features or more rooms, courses and options. As well as improving existing games you can also delve into the disk only world of the Infocom adventures. These are amazing games packed with mind boggling puzzles and text descriptions to fire your imagination. Whatever your particular interests you'll be able to find at least one Infocom adventure to explore. Fantasy fans can explore the amazing Zork trilogy and the worlds of Enchanter, Sorcerer and Spellcaster or Sci-fi buffs can step boldly into Planetfall and Stationfall or even enter the amazing Hitch Hikers Guide to the Galaxy!

SSI games are rarely converted onto tape and so you need a drive to command great battles in one of their many wargames or explore the roleplaying games such as Phantasie and Shard of Spring. Or why not try Rainbird's graphic adventures such as The Pawn and The Guild of Thieves or the amazing Ultima series of games. The next one to reach these shores will be Ultima V and will be so big that it will cover both sides of four disks! Try and get that on tape!

Just when you're thinking how much faster your disk drive is than the cassette player someone, somewhere develops a program that makes it even faster. Cartridges such as Quickdisc and the Expert will speed up your drive and also include a fast disk formatter but the most impressive device is Dolphin DOS from Evesham Micros. This is actually an entirely new disk operating system that replaces the existing DOS in your drive and will allow you to load programs in seconds and not minutes!

Disk Commands

Learn how to use your disk drive more efficiently.

By Stuart Cooke

Not only do Commodore disk drives provide the user with commands to format the disk, read the contents of the disk, LOAD programs etc, there is also a whole range of less documented commands that allow you to talk to the disk and disk drive directly. This range of commands is referred to as Direct Access commands. Once you understand the concept of these commands and how the disk drive works you can get the drive to do whatever you want.

Inside the 1541

Probably the most common of the whole range of Commodore drives is the 1541. For the sake of this article we will refer to this drive. Most of the information is the same for all of the other drives.

Before we take a close look at the direct access commands that are available it is about time we had a look at the inside of a 1541 disk drive. Figure 1 is a memory map of the disk drive. Before you can program the disk drive efficiently it is important that you know its inner workings.

Talking to the drive

Now that you've had a close look at what you can get at inside a drive it's time to move onto the direct access commands.

All communication between the disk and the user is made through a buffer. If you take a look at Figure 1 you will see that there are five buffers available. However, only four of these are free for use. Buffer four is normally reserved for holding an image of the disk BAM. When using SEQ and REL files at the same time, buffer number three is also not available because the directory uses it.

If you want to write information onto the disk or read information from it then the sector that you want to manipulate must be read into one of the buffers. When you wish to use a buffer, you first have to OPEN a channel and specify which buffer you wish to use. For example OPEN 1,8,2," # 2" would open the channel to buffer number 2. However, it is good practice not to specify the buffer number but let the DOS select it for you. You achieve this by not specifying a number after the '#' sign. For example:

```
OPEN1,8,2," # ".
```

If your selected buffer contains alphanumeric data, and is not over 88 characters in length you can use the INPUT # command to read in data from the buffer. Otherwise you will have to use the GET # command. Note that when using GET # it does not check for null characters. It is therefore advisable to have the following basic line, or something similar, inside a program that reads data from the disk with a GET # statement.

```
IF A$=""THEN A$=CHR$(0)
```

Obviously the chatacter read from the disk must be stored in A$.

Before we go any further there are four things that you should remember.

1 A PRINT # command to the command channel (secondary address of 15) send a direct access command to the DOS.
2 A PRINT # statement to any other channels (i.e. secondary address not 15) sends data into one of the buffers already mentioned.
3 An INPUT # or GET # statement to the command channel (secondary address of 15) returns any error messages.
4 An INPUT # OR GET # statement to any other channel reads data from one of the buffers.

Block-Read

The block-read command tells the 1541 to read a sector from the disk into your open buffer – strictly speaking this is known as a direct access file. This command is shortened to "B-R" when talking to the drive or should you prefer to shorten the command even more, use the command "U1". An

1541 Memory Map

DRIVE ADDRESS HEX	DEC	Description
$0000	0	Command code for buffer 0
$0001	1	Command code for buffer 1
$0002	2	Command code for buffer 2
$0003	3	Command code for buffer 3
$0004	4	Command code for buffer 4
$0006-0007	6-7	Track and sector for buffer 0
$0008-0009	8-9	Track and sector for buffer 1
$000A-000B	10-11	Track and sector for buffer 2
$000C-000D	12-13	Track and sector for buffer 3
$000E-000F	14-15	Track and sector for buffer 4
$0012-0013	18-19	ID for drive 0
$0014-0015	20-21	ID for drive 1
$0016-0017	22-23	ID
$0020-0021	32-33	Flag for head transport
$0030-0031	48-49	Buffer pointer for disk controller
$0039	57	Constant 8, mark for beginning of data block header
$003A	58	Parity for data buffer
$003D	61	Drive number for disk controller
$003F	63	Buffer number for disk controller
$0043	67	Number of sectors per track for formatting
$0047	71	Constant 7, mark for beginning of data block header
$0049	73	Stack pointer
$004A	74	Step pointer for head transport
$0051	81	Actual track number for formatting
$0069	105	Step size for sector division (10)
$006A	106	Number of read attempts (5)
$006F-0070	111-112	Pointer to address for M and B cmds.
$0077	119	Device number plus $20 (32) for listen
$0078	120	Device number plus $40 (64) for talk
$0079	121	Flag for listen (I/O)
$007A	122	Flag for talk (I/O)
$007C	124	Flag for ATN from serial bus receiving
$007D	125	Flag for EOI from serial bus
$007F	127	Drive number
$0080	128	Track number
$0081	129	Sector number
$0082	130	Channel number
$0083	131	Secondary address
$0084	132	Secondary address
$0085	133	Data byte
$008B-008D	139-141	Work storage for division
$0094-0095	148-149	Actual buffer pointer
$0099-009A	153-154	Address of buffer 0 $0300
$009B-009C	155-156	Address of buffer 1 $0400
$009D-009E	157-158	Address of buffer 2 $0500
$009F-00A0	159-160	Address of buffer 3 $0600

$00A1-00A2	161-162	Address of buffer 4 $0700
$00A3-00A4	163-164	Pointer to input buffer $0200
$00A5-00A6	165-166	Pointer to buffer error message $02D5
$00B5-00BA	181-186	Record number lo, block number lo
$00BB-00C0	187-192	Record number hi, block number hi
$00C1-00C6	193-198	Write pointer for REL file
$00C7-00CC	199-204	Record length for REl file
$00D4	212	Pointer in record for REL file
$00D5	213	Side sector number
$00D6	214	Pointer to data block in side sector
$00D7	215	pointer to record in REL file
$00E7	231	File type
$00F9	249	Buffer number
$0100-0145	256-325	Stack
$0200-0228	512-552	Buffer for command string
$024A	586	File Type
$0258	600	Record length
$0259	601	Track side sector
$025A	602	Sector side sector
$0274	628	Length of input line
$0278	632	Number of filenames
$0297	663	File control method
$0280-0284	640-644	Track of a file
$0285-0289	645-649	Sector of a file
$02D5-02F9	725-761	Buffer for error messages
$02FA-02FC	762-764	Number of free blocks
$0300-03FF	768-1023	Buffer 0
$0400-04FF	1024-1279	Buffer 1
$0500-05FF	1280-1535	Buffer 2
$0600-06FF	1536-1791	Buffer 3
$0700-07FF	1792-2047	Buffer 4

Fig. 1

example of how to use the command is shown later. As a point to note, some Commodore drives have a bug in the B-R command and for this reason, it is always best to use the "U1" command.

Block-Write

The block-write command is the exact opposite to the block-read command. This takes the contents of the buffer in use and writes it into the specified sector. The format for this command is B-W or U2. Again a problem exists with B-W so use the U2 command.

Allocating Space

The Block-Allocate, or B-A, command allows the user to reserve blocks on the disk. The main purpose for this is to reserve areas of the disk for special usage. The Block-Allocate commands clear the necessary bits in the Block Availability Map after execution of this instruction.

The Buffer-Pointer command, shortened to B-P, tells the DOS just where you wish to start reading or writing data to or from in a buffer.

When using the direct access commands there are two formats for the command available. Either may be used depending upon your own preference. The first method is:

PRINT # 15,"U1:"channel-number; drive

the second method is:

PRINT # 15,"U1 channel-number drive"

Now that we've discussed what commands are available, let's take a closer look at them in use. The following examples should make the use of buffers and direct access commands much clearer.

Suppose you wished to follow a program through on disk by track and sector without actually reading in any data. To do this you need to follow the path of the "link" bytes. That is, the two bytes at the start of each block that tells you where the next track and sector of the specific program is.

The program in Figure 2 gives an example of how you would perform this task.

For our second example let's presume that we wish to read the diskette name from within a program. As you already know, (see article on disk editing), the name starts at position 144 of track 18 sector 0. Using a B-R command you would read the specified sector into the buffer. You

```
1 OPEN8,8,15 : REM OPEN THE
COMMAND CHANNEL
2 OPEN 4,8,4,"#" : REM OPEN
DIRECT ACCESS FILE
3 INPUT "TRACK AND SECTOR
PLEASE";TR,SE
4 PRINT#8,"U1:"4;0;TR;SE : REM
READ CONTENTS OF TRACK/SECTOR
INTO BUFFER
5 GET#4,T$,S$ : REM READ FIRST 2
BYTES INTO BUFFER
6 TR=ASC(T$+CHR$(0)):
SE=ASC(s$+CHR$(0)) : REM MAKE
SURE VALUE IS INTEGER
7 IF TR=OTHENCLOSE4:CLOSE8:END :
REM END OF LINKS
8 PRINT"TRACK NUMBER IS: ";TR,
"SECTOR NUMBER IS: ";SE
9 GOTO 4 : REM GET NEXT LINK
```

Fig. 2

```
1 OPEN8,8,15 : REM OPEN COMMAND
CHANNEL
2 OPEN4,8,4,"#" : REM OPEN DIRECT
ACCESS FILE
3 PRINT#8,"U1:"4;0;18;0 : REM
READ CONTENYTS OF DESIRED
TRACK/SECTOR
4 PRINT#8,"B-P:"4;144 : REM POINT
TO WHERE YOU WANT TO READ FROM
5 FORX=1TO16 : REM LENGTH OF
FILENAME
6 GET#4,X$:IFX$=CHR$(160)THEN8:
REM IF SHIFTED SPACE END
7 PRINTX$:NEXT : REM PRINT OUT
AND GET NEXT LETTER
8 CLOSE4:CLOSE8 : REM END
```

Fig. 3

would then have to read through all of the 143 bytes in the buffer until you get to byte 144, the start of the name. However there is a quicker way. The B-P command allows you to position the data pointer anywhere within the buffer. The bytes in the buffer are numbered from 0 to 255. The pointer is automatically reset to 0 after a "U1" command. Figure 3 illustrates our example.

The commands block-write and block-read are used in conjunction with each other. As previously mentioned block-write allows you to write the contents of a buffer to a specified track and sector,the command does not alter the contents of the buffer – you do this yourself. Figure 4 takes the program in figure 3 and expands it so that the disk name read in can be altered in the buffer and then re-written to the correct position, changing the disk name.

When using Program, Sequential or Relative files on disk, the BAM is being constantly updated as programs are written, scratches, etc. This prevents programs from being overwritten. However, when we use direct-access files the data that you write to the disk is not marked in the BAM.

This means that data you have put on the disk could be overwritten. To prevent this from happening we can use the Block-Allocate command. If you try to allocate a block that has already been marked as user, then you will get an error message 65. NO BLOCK,T,S ; T and S are the next higher numbered free blocks available.

The syntax for using the block allocate command is:

B-A drive track sector

The following example would mark track 17 sector 5 as in use:

```
1 OPEN 8,8,15
2 PRINT # 8,"B-A:"0;17;5
```

Freeing a Block

The Block-Free of B-F command is the opposite of the above command. This will set the specified bits in the BAM making the relevant tracks and sectors available for use.

Should we want to free the sector allocated in the above example you would do it as following:

```
1 OPEN 8,8,15
2 PRINT # 8,"B-F:"0;17;5
```

```
1 OPEN8,8,15 : REM OPEN COMMAND
CHANNEL
2 OPEN4,8,4,"#" : REM OPEN DIRECT
ACCESS FILE
3 PRINT#8,"U1:"4;0;18;0 : REM
READ CONTENTS OF DESIRED
TRACK/SECTOR
4 PRINT#8,"B-P:"4;144 : REM POINT
TO WHERE YOU WANT TO READ FROM
5 X$="NEW DISK NAME"
6
IFLEN(X$)<16THENX$=X$+CHR$(160):
GOTO6 : REM PAD OUT NAME
7 PRINT#4,X$; : REM CHANGE BUFFER
CONTENTS
8 PRINT#8,"U2:"4;0;18;0 : REM
WRITE BACK TO DISK
9 PRINT#8,"I":CLOSE4:CLOSE8:END
```

Fig. 4

Note–allocating and freeing blocks has an effect only on blocks that are used by PRG, SEQ and REL files by the DOS. The B-W and B-R commands do not check the BAM before overwriting blocks. Using these commands you can write to blocks marked as allocated in the BAM.

One use that has been made of this command in the past is to write a small menu program onto track 18, the directory track. This means that the MENU will not take up any of the normal disk space available.

Block-Execute

The Block-Execute command, shortened to B-E is an extremely powerful command. In essence, this command reads a sector from the disk into a previously opened buffer. The contents of that buffer are then executed as a machine code program within the buffer.

The syntax for the command is:

B-E channel drive track sector

When using the B-E command it is usual to specify the buffer to be used in the OPEN command, just in case the machine code program isn't relocatable.

The following program would read the contents of track 14 sector 6 into buffer number 2 and execute it.

```
1 OPEN 8,8,15
2 OPEN 4,8,4" # 2"
3 PRINT # 8,"B-E:"4;0;14;6
```

Talking Memory

Not only are Commodore disk drives provided with a wealth of commands that allow you to access the disk but commands also exist that allow you to gain access to the memory inside the disk drive.

There are three commands that we will detail here. They are Memory Read (M-R), Memory Write (M-W), and Memory Execute (M-E).

All of these commands require a knowledge of the inner workings of the DOS and a knowledge of 6502/6510. The memory map of the disk drive in figure 1 will be of invaluable use in this matter.

The syntax for the Memory Read command is:

M-R CHR$(LO) CHR$(HI) [CHR$(number)]

CHR$(LO) is the low byte of the address in DOS that is to be read.
CHR$(HI) is the high byte of the address in DOS that is to be read.
CHR$(number) is an optional extra parameter indicating how many bytes are to be read.

The figures 5 and 6 are used to illustrate the use of this command. The first example shows how to read from disk memory, how many free bytes there are on the current disk. The second example reads the disk name.

Memory Write is the complimentary command to Memory Read. Writing can only be performed to DOS RAM, page zero, stack and buffers. It is possible to send more than one byte to the disk drive with this command. The syntax is as follows:

M-W CHR$(LO) CHR$(HI) CHR$(number) CHR$(data) CHR$(data) etc etc...

Finally the Memory Execute command (M-E) will call and execute a machine code program that resides in DOS memory. The routine must end with a RTS instruction. The syntax for the command is as follows:

M-E CHR$(LO) CHR$(HI)

You can not only execute your own routine written with the use of the M-W command, but also the DOS routines.

Summary of Direct Access

Within the confines of this article I can obviously only gloss over the subject of programming your disk drive. The following table lists just a few ideas that spring to mind as tasks you could perform with your new-found knowledge.

1 You can manipulate the sectors and change the BAM
2 You can make changes to the directory.
3 You can make changes to files.
4 You can protect files from erasure.
5 You can close files that are OPENed.
6 You can prevent directories from being viewed.
7 You can recover lost or damaged files.
8 You can create data structures that the DOS would not normally recognise.

9 You could place a menu program within the directory – thus saving space.

10 You could put a simple form of protection on the disk.

Really the list is endless. Only your own knowledge and requirements are your constraints. However before you do try any of the commands that we have discussed out yourself, I must stress the importance of making sure that you only play around with old disks until you know what you are doing. After all, one simple mistake could wipe out a whole disk.

Figure 5

```
1 OPEN8,8,15
2 PRINT#8,"M-R"CHR$(250)CHR$(2)
3 GET#8,X$:IFX$=""THENX$=CHR$(0)
4 PRINT#8,"M-R"CHR$(252)CHR$(2)
5 GET#8,Y$:IFY$=""THENY$=CHR$(0)
6 PRINTASC(A$)+256*ASC(Y$)
7 CLOSE8
```

Figure 6

```
1 OPEN8,8,15
2 PRINT#8,"M-R"CHR$(144) CHR$(7)
CHR$(16)
3 INPUT#8,X$
4 PRINTX$
5 CLOSE8
```

DISK EDITING

You can rescue files and much more once you know how to use a disk editor.

By Stuart Cooke

How often have you scratched your latest programming masterpiece from your disk only to realise a few minutes later that you didn't have a backup?

No doubt, until now, the only option open to you was to re-type the whole program from the beginning.

A little more understanding about how a Commodore disk drive works will enable you to rescue most scratched programmes and make numerous other changes to your disk directories.

Before you can start playing with the contents of floppy disks it is important that you understand how the information is stored on them. If you don't understand and you start changing areas of a disk you can probably wave bye-bye to the contents of the whole thing.

In order to make any changes to a disk you will require access to some sort of disk editor program. There are a few available commercially. Disk Doctor from Precision being a good example, and we provide a listing for a good disk editor later in the supplement.

Disk Structure

You are no doubt aware, when you purchase a disk it is totally blank and of no use to you at all. Before the

Block Distribution By Track

Track Numbers		Range of Sectors		Total Sectors		Single Sided	Double Sided
HEX	DEC	HEX	DEC	HEX	DEC		
$01-$11	01-17	$00-$14	00-20	$15	21	YES	YES
$12-$18	18-24	$00-$12	00-18	$13	19	YES	YES
$19-$1E	25-30	$00-$11	00-17	$12	18	YES	YES
$1F-$23	31-35	$00-$10	00-16	$11	17	YES	YES
$24-$34	36-52	$00-$14	00-20	$15	21	NO	YES
$35-$3B	53-59	$00-$12	00-18	$13	19	NO	YES
$3C-$41	60-65	$00-$11	00-17	$12	18	NO	YES
$42-$46	66-70	$00-$10	00-16	$11	17	NO	YES

Fig. 1

computer/disk drive can make use of the disk it must be formatted.

Formatting a disk divides it into a number of rings called tracks. On all of the popular Commodore drives (1541 etc) except for the 1571, the disk is divided into 35 rings, on one side of the disk only. If you have a 1571 then the second side of the disk is also split into 35 rings or tracks.

Each of these circular tracks is then split up into a number of equal segments called sectors. Each track contains between 17 and 21 sectors. Figure 1 illustrates this more clearly. Note that the tracks on the second side of a 1571 are numbered from 36 to 70 and do not start from 1 again.

As Figure 1 clearly shows the number of sectors in each track gets smaller towards the centre of the disk. The reason for this is quite obvious when your ealise that the tracks are a lot shorter at the centre of the disk than they are on the outside.

How much room?

In the centre of the disk, side 1 for 1571 users, you will find the information track. Track 18 is used to keep all necessary information about programmes, where they are stored on the disk and how much room is free on the disk.

The first sector of track 18 is used to record which sectors of the disk have been used. This is called The Block Availability Map or BAM. Every time you make any changes to the contents of a disk the contents of the BAM are updated so the disk drive can find out which tracks and sectors on the drive are used.

Figure 2 shows the contents of the first 255 bytes of track 18 sector 0. As you can see from the figure this sector not only contains information about the BAM but is also used to store important information about the disk, such as the DOS type, the format type, the name, etc.

Returning to the BAM, figure 2 shows that bytes 4 to 143 of track 18 sector 0 holds the BAM. For consecutive bytes are used to represent the BAM for each track. Figure 3 gives a representation of the possible contents of sector 0 bytes 5 to 7, in other words the bytes that give an indication of which sectors on track 1 have been used.

As you are no doubt aware, a single byte can hold a number from 0 to 225. If we translate this from decimal to binary this means that the numbers held will range from 00000000 to 11111111. From the binary representation it can be seen that each byte can hold the information for eight sectors. Each 1 or 0 represents the status of the corresponding sector. A 0 tells the disk drive that the sector in question has been used while a 1 shows that it is still available.

If you take a second look at Figure 3 you can see that in our representation sectors 1 to 10 have all been used and sectors 11 to 23 are still available.

You may be wondering how the disk drive knows how many sectors are available on each track. If you refer back to Figure 2 you will see that the information about the BAM for each track is held in four consecutive bytes. We have already taken a close look at the BAM for track 1 above. As stated this information is stored in bytes 5 to 7 of track 18 sector 0. If you refer back to Figure 2 you will see that the previous byte (4) holds a number that represents the actual number of sectors available on track 1, in this case 23.

This sequence of four bytes is repeated for all tracks on the disk. 1571 users can see that the information about the second side of the disk is stored in the same way as the first side in bytes 221 to 255 of track 18 sector 0.

Disk Info

Bytes 144 to 255 of track 18 sector are used to hold specific information about the disk. Much of this is information that is printed at the top of each directory listing. If you refer back to Figure 2 you will see exactly what information is held in these bytes. Should you ever want to change the ID or the title of a disk then you can do it quite simply by using a disk editor to read the information on the disk into your computers, make the changes required and then re-write the information to the disk.

Directory Info

The sectors from one onwards on track 18 are used to hold information

BAM Format 1541 - Track 18 Sector 0

Number	Contents	Definition
0	18	Track of next directory block. Always 18.
1	1	Sector of next directory block Always 1.
2	65	ASCII character A indicating 1541/51/71/4040 format
3		Double sided flag. Ignored on 1541
4		Number of sectors available on track 1.
5		Track 1, Sector 0-7 BAM.
6		Track 1, Sector 8-16 BAM.
7		Track 1, Sector 17-23 BAM.
8		Number of sectors available on track 2.
9		Track 2, Sector 0-7 BAM.
10		Track 2, Sector 8-16 BAM.
11		Track 2, Sector 17-23 BAM.
...etc. Down To		
140		Number of sectors available on track 25.
141		Track 35, Sector 0-7 BAM.
142		Track 35, Sector 8-16 BAM.
143		Track 35, Sector 17-23 BAM.
144-159		Disk name padded with shifted spaces (CHR$ 160).
160-161	160	Shifted space.
162-163		Disk ID.
164	160	Shifted space.
165-166		ASCII "A which is the DOS version format type 1540/41/51/71/4040.
167-170	160	Shifted spaces.
171-255	0	Nulls, not used.

1571 Drive As Above Except :

3		Double sided flag: $80=Double Sided $00=Single Sided.
171-220	0	Nulls, not used.
221-237		Number of sectors available in
tracks 36-52.		
		Each sector by each byte.
Format as for 1541.		
238	0	Number of sectors in track 53
239-244		Number of sectors available tracks 54-59. Each track by each byte.
245-250		Number of sectors available tracks 60-65.
251-255		Number of sectors available tracks 66-70.

Fig. 2

BAM ALLOCATION

SECTORS 0-7	SECTORS 8-15	SECTORS 16-23
00000000	00011111	11111111

Fig. 3

relating to any program you have stored on the disk. Each sector is referred to as a directory block and will hold the information for around eight files. The first two bytes of each block are used to give the track and sector of the next directory block. Figure 4 shows the format of the directory on the disk. If there is no more information on the disk the first two bytes in the last sector will contain 0's. Each of the eight program entires in a directory block is made up of 30 bytes. These are the ones that hold the information about the type of program, where it is on the disk, etc. Figure 4 shows what information stored in the 30 bytes.

The first byte of each directory entry is used to hold information about the type of file that you are looking at. If you refer to figure 4 once more you will see that the file referred to can be one of five types. However, this isn't the only information that this byte gives.

Bits 0 to 2 of this byte are used to tell us what type of file we are looking at. Bit 7 is used to tell the drive if the file is correctly closed or not. A 1 in bit number 7 shows that the file is still open. This can be seen on a directory listing as a '*' following the filetype.

Bit 6 holds an extremely important piece of information which, unfortunately, a large number of people are unaware of. This bit is used to tell the disk drive whether or not the file is protected. Setting this bit to '1' will prevent deleting this file by normal methods. A protected file can be seen in a directory listing as it has a '<' following the filetype. If you have important files it is well worth going to the trouble of setting this bit to prevent accidental erasure.

Program Erasure

Whenever you delete a program from disk a number of changes are made to the disk. First of all, the sectors that the program occupied are marked as free in the BAM and secondly, the file type is changed to zero indicating that it has been deleted. The important thing to remember is that the program is still n the disk and will remain there until another program is saved over it, probably following the next SAVE operation.

If you delete a file by accident and realise before you have saved another to disk then it is a very simple matter to retrieve it. All you have to do is find the entry for the file in the directory block and change the filetype to whatever it was before. If, for example, the file type was a program you would store the number 02 in the relevant position. You will now be able to use your file.

Note, the BAM will not be updated and there is a chance that the next SAVE operation may overwrite your reserected file. It is therefore a good idea to make a new copy of any reserected file befoore doing anything else.

Having taken a look at the way that a disk directory is stored on a Commodore disk it is probably worth looking at the format that files take. Figures 5 to 7 give details on all of

Dir File Format, Track 18 Sectors 1-19

```
Byte      Definition
0,1       Track and sector of next DIR block.
2-31      File Entry 1
34-63     File Entry 2
66-95     File Entry 3
98-127    File Entry 4
130-159   File Entry 5
162-191   File Entry 6
194-223   File Entry 7
226-255   File Entry 8
```

Structure Of Each Directory Entry

```
Byte      Contents    Definition

0         128+type    File type OR'ed with $80
to indicate closed file.
                      File type OR'ed with $C0
                      to indicate locked file.
                      Type: 0 = DELeted.
                            1 = SEQuential.
                            2 = PRoGram.
                            3 = USeR.
                            4 = RELative.
1-2                   Track and sector of first
                      data block.
3-18                  File name padded with shifted
                      spaces.
19-20                 REL file only. Track and
                      sector of 1st side sector.
21                    REL file only. Record length.
22-25                 UNUSED.
26-27                 Track and sector of replacement
                      during @SAVE or @OPEN.
28-29                 Number of blocks in file, stored
                      as a two-byte integer in normal
                      lo-byte hi-byte format.
```

Fig. 4

Program File Format

```
Byte    Definition

FIRST SECTOR
0,1     Track and sector of next block in file.
2,3     Load address of program.
4-255   Next 252 bytes of prg info stored
        tokenised as in computers memory.

REMAINING FULL SECTORS
0,1     Track and sector of next block in file.
2-255   next 254 bytes of prg info stored
        tokenised as in computers memory.

FINAL SECTOR
0,1     NULL ($00), followed by number of
        valid data bytes in sector.
2-???   Last bytes of program data.
        The end of a BASIC file is marked by
        3 zero bytes in a row.
```

Fig. 5

Sequential File Format

```
Byte    Definition

ALL BUT FINAL SECTOR
0,1     Track and Sector of next data block.
2-255   254 bytes of data.

FINAL SECTOR
0,1     NULL ($00), followed by number of
        bytes in sector.
2-???   Last bytes of data.
```

Fig. 6

the main file types. Careful examination of these figures should provide you with all of the information that you require to know. One important point which is worth a mention is that you can find out the start address of any program file by examining bytes 2 and 3 of the first sector of any program.

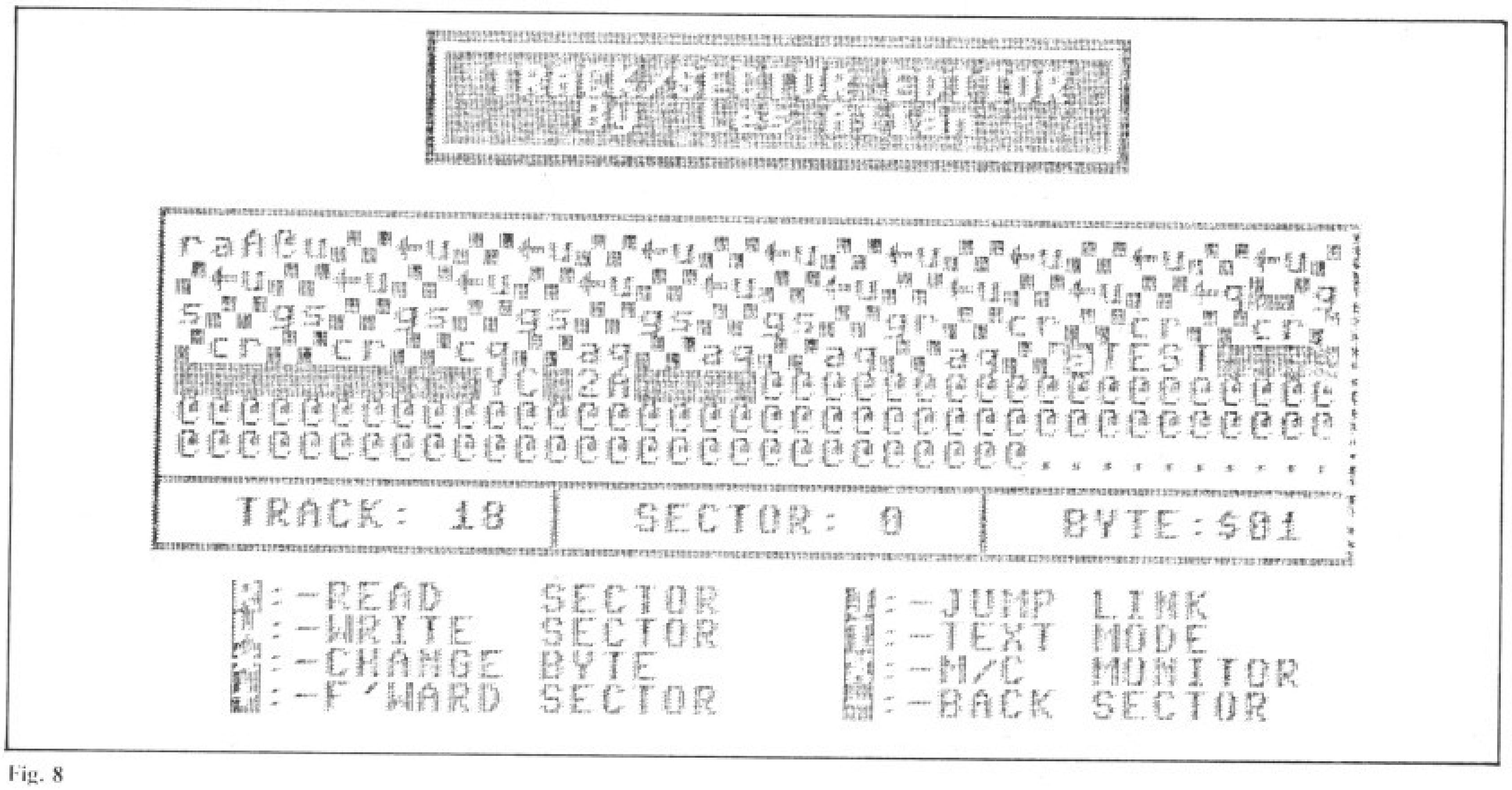
Give it a go

People say that the only way to find out if you have understood something is to give it a go. Presented here is a small tutoriall covering some of the aspects that we have looked at within this article. I have not referred to any specific Disk Editor, however, the figures presented here are from the one presented in this supplement.

Relative File Format

Byte	Definition
DATA BLOCK	
0,1	Track and Sector of next data block.
2-255	254 bytes of data. Empty Records contain $FF in the first byte followed by $00 to the end of the record. Partially filled records are padded with $00.
SIDE SECTOR BLOCK	
0,1	Track and Sector of next data block.
2	Side sector number (0-5).
3	record length.
4-5	Track and Sector of 1st side sector (0).
6-7	Track and Sector of 2nd side sector (1).
8-9	Track and Sector of 3rd side sector (2).
10-11	Track and Sector of 4th side sector (3).
12-13	Track and Sector of 5th side sector (4).
14-15	Track and Sector of 6th side sector (5).
16-255	Track and sector pointers to 120 data blocks.

Fig. 7



```
TRACK: 18      SECTOR: 0      BYTE:$01
```

Fig. 8

Firstly, you will need to format a blank disk. Make sure that it is blank and contains nothing that you require before going any further.

Put your disk in your drive and enter the following command:

```
OPEN,8,15,"N0:TEST,YC"
```

Next type the following small proogram and SAVE it onto your disk with the filename "ONE"

```
10 REM THIS IS A VERY
20 REM SHORT TEST
30 REM PROGRAM
40 REM
50 REM THE END
```

OK, so it's nothing stunning but it will serve our purpose very well.

Now LOAD your disk editor and examine the contents of track 18 sector 0. If you have a look at bytes 144 to 161 you will see that they hold the name of the disk. Figure 8 shows what you would see using our disk editor, the display may be slightly different with your editor.

Now we shall change the disk name. Change the letters of the filename to "DEMO". Figure 9 shows how your disk should look now.

Once you have done this write the sector back to the disk and your changes will have been made permanent. If you want to check this for yourself reset your machine and load in the disk directory, you will see that the name has been changed.

Rescue a file

Now we are going to delete a file and then recover it. Scratch the test file from your disk with the following command:

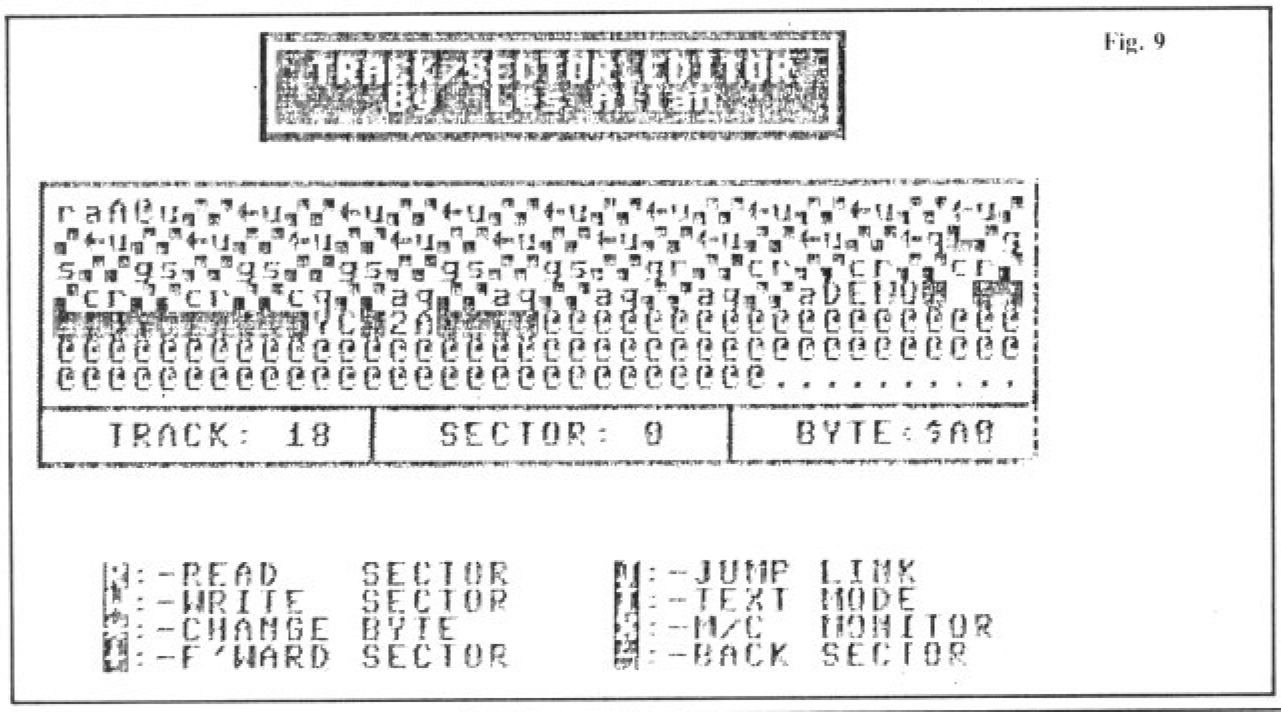
```
OPEN,8,15,"S0:ONE"
```

If you now try and LOAD the program"ONE" you will be unable to do so.

Load your disk editor into your computer and take a look at track 18 sector 1. Figure 10 shows something similar to what you should see. The 00 byte indicated on our picture shows that the file has been deleted. If you use the editor to change this to 82, ie a program file, and then write the sector back to disk you will be able to LOAD the program once again.

Obviously this article has only glossed over the area of disk structure and disk editing. If you have old disks that you no longer want don't format them straight away, play around with them using a disk editor till you feel sure that you know exactly what you are doing.

REMEMBER never edit a disk that has programmes on it that you require unless you are sure about what you're doing.



```
TRACK/SECTOR EDITOR
   BY LES ALLEN

TRACK: 18    SECTOR: 0    BYTE:$A0

R:-READ    SECTOR     J:-JUMP  LINK
W:-WRITE   SECTOR     T:-TEXT  MODE
C:-CHANGE  BYTE       X:-M/C   MONITOR
F:-F'WARD  SECTOR     B:-BACK  SECTOR
```

Fig. 9

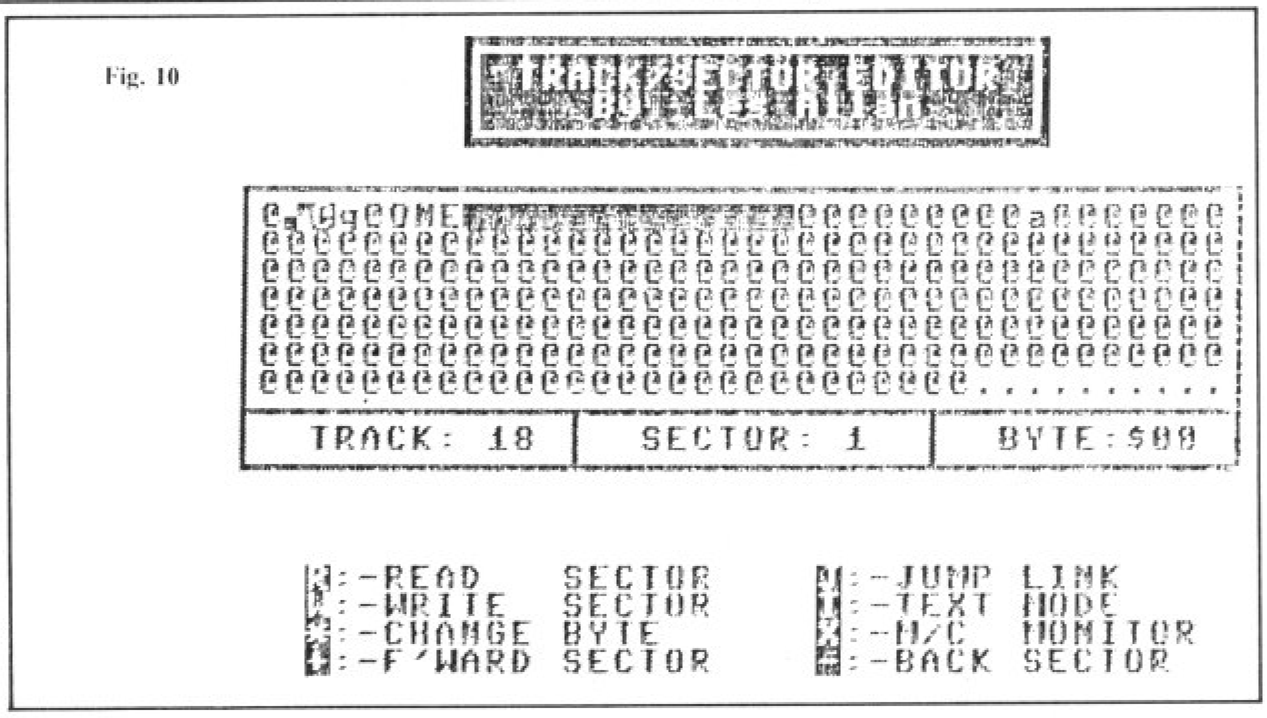


Fig. 10

```
TRACK/SECTOR EDITOR
   BY LES ALLEN

TRACK: 18    SECTOR: 1    BYTE:$00

R:-READ    SECTOR     J:-JUMP  LINK
W:-WRITE   SECTOR     T:-TEXT  MODE
C:-CHANGE  BYTE       X:-M/C   MONITOR
F:-F'WARD  SECTOR     B:-BACK  SECTOR
```

Track/Sector Editor for CBM 64/128

Hints on how to use the editor

By Les Allan

It is often useful to edit a section of memory either resident in the computer's memory or, as explained in this utility, by modifying the saved file directly on disk. It is extremely important that a 'backup' be made prior to making any alterations, so in the event of a mistake the user can always revert back to the original.

Boot up the TRACK/SECTOR EDITOR and you are directly into the READ SECTOR mode. Insert the disk to be read and simply press RETURN twice to select the BAM of the directory (default mode is TRACK 18 and SECTOR 0) or input the required TRACK and SECTOR as commanded by the cursor. The contents of that sector are now displayed in the viewing window with the cursor flashing at the top left hand corner – position 0. The first two bytes are highlighted in white to indicate the LINK to the location of the next track and sector of the saved file.

T - TEXT MODE

Use the cursor control keys to locate the cursor to the required position and PRESS T to enter TEXT mode. Simply type in from the keyboard the text required using the CTRL key to select lower case and press RETURN to end.

* - CHANGE BYTE

Use the cursor control keys to locate the cursor to the required position and PRESS the * key. The value of the byte to be changed can be entered either directly in decimal or in hex provided the number is preceded by the $ key.

W - WRITE SECTOR

To write the modified sector to the disk PRESS the W key and confirm your intention by pressing the Y key or quit with N key.

R - READ SECTOR

PRESS the R key at any time to select a

A C000 LDA $C100	assemble code at $C000
D C000	dis-assemble code at $C000
M C000 C100	monitor code between $C000 and $C100
F C000 C100 EE	fill with $EE between $C000 and $C100
S "SECTOR",08,C000,C100	save contents of sector to disk
L "SECTOR",08	load sector back to $C000
X	quit monitor and return to the editor

different track and sector to be read.

F1 - RE-START

PRESS the F1 key at any time during the edit mode to reset the registers to the start up configuration of TRACK 18 and SECTOR 0.

J - VIEW NEXT SECTOR

PRESS the HOME key to position the cursor to the LINK track and sector followed by the J key will cause the editor to Jump to the next sector of the saved file.

K - M/C MONITOR

Press the X key to enter the M/C monitor which will display the contents of the sector between $C000 and $C100.

Figure 1 gives the available commands.

Directory Entry Explanation

Boot up the TRACK/SECTOR EDITOR, insert diskette to be read, press RETURN twice to read BAM and J to jump to TRACK 18 and SECTOR 1 which is always the first sector of the directory. Figure 2 gives details.

Up to eight directory entries are saved per sector in exactly the same sequence. To view the contents of any saved file simply locate the cursor at byte 4 and press J to read the first track and sector.

The program as listed must be typed in exactly as written and saved prior to running. Error trap routines are included to ensure that the data as read is correct. The program when run stores the relevant code at a temporary address of $8000 and when prompted relocates the working program to basic ($0801) and saves the completed program to disk.

Fig. 2

byte 1	$12 location of next track of directory – $00 if end
byte 2	$04 location of next sector of directory – $FF if end
byte 3	$82 type of file saved (PRG) – change to $C2 to LOCK file
byte 4	$11 start track of saved file
byte 5	$00 start sector of saved file
byte 6 – 21	NAME of save file
byte 22 – 30	relative file data
byte 31	length of saved file in hex
byte 32	$00

```
program: disk editor boot

32  10 m1=32768:m2=38768:ch=0
97  11 printchr$(144)chr$(147)
c8  12 poke52,120:poke56,120:pok
    e53280,15:poke53281,12
94  13 print
8f  14 printspc(8)chr$(18)"[su,s
    *22,si]
26  15 printspc(8)chr$(18)"[s-,s
    pc22,s-]
e4  16 printspc(8)chr$(18)"[s-]
    track/sector  editor [s-]
3c  17 printspc(8)chr$(18)"[s-,s
    pc22,s-]
df  18 printspc(8)chr$(18)"[s-]
    for commodore 64/128 [s-]
4a  19 printspc(8)chr$(18)"[s-,s
    pc22,s-]
22  20 printspc(8)chr$(18)"[s-,s
    pc5]by les allan[spc5,s-]
50  21 printspc(8)chr$(18)"[s-,s
    pc22,s-]
6c  22 printspc(8)chr$(18)"[sj,s
    *22,sk]
a0  23 print:printchr$(155)
45  24 print" this routine enabl
    es a specified track"
80  25 print
43  26 print" and sector to be d
    isplayed and edited."
82  27 print
d4  28 print" simply follow the
    instructions on  the"
84  29 print
b0  30 print" screen to modify a
    ny track and sector."
59  31 print:print
d0  32 readcode$
bb  33 lb=asc(right$(code$,1))-4
    8:iflb>9thenlb=lb-7
cf  34 hb=asc(left$(code$,1))-48
    :ifhb>9thenhb=hb-7
32  35 printspc(5)chr$(5)"readin
    g memory block ...";m1:print
    chr$(145);
28  36 pokem1,hb*16+lb:m1=m1+1:c
    h=ch+hb+lb:ifm1<m2then32
ec  37 ifch=71422then39
13  38 print"check sum error - c
    heck data statements!":print
    chr$(19):end
ec  39 print"  hit return to sav
    e completed program"
d1  40 getkey$:ifkey$<>chr$(13)t
    hen40
68  41 sys38688
10  42 :
ce  43 ::::::::::::::::::: datum
    statements ::::::::::::::::::

16  44 :
ad  45 data 00,0b,08,00,00,9e,32
    ,30,36,31,00,00,00,20,44,e5
60  46 data a9,0c,8d,20,d0,8d,21
    ,d0,ea,a9,36,85,01,4c,3f,1e
a5  47 data 28,31,34,29,3a,83,22
    ,1d,22,2c,22,9d,22,2c,22,11
02  48 data 22,2c,22,91,22,2c,22
    ,13,22,2c,2b,2c,2d,2c,2a,2c
f1  49 data 57,2c,54,2c,52,2c,4a
    ,2c,58,2c,22,85,22,00,70,08
a1  50 data 04,00,81,5a,b2,31,a4
    ,31,34,3a,87,43,24,28,5a,29
9b  51 data 3a,82,3a,4e,b2,31,34
    ,3a,53,44,24,b2,22,30,22,00
69  52 data ba,08,05,00,54,24,b2
    ,22,20,27,e2,22,3a,50,b2,31
6d  53 data 33,30,35,3a,54,b2,31
    ,38,3a,53,b2,30,3a,8d,31,36
a4  54 data 34,3a,89,36,38,00,da
    ,08,0b,00,51,31,b2,51,31,aa
d8  55 data 31,32,38,3a,8b,51,31
    ,b1,32,35,35,a7,51,31,b2,51
7b  56 data 31,ab,32,35,36,00,fb
    ,08,0c,00,97,50,2c,51,31,3a
90  57 data 81,5a,b2,31,a4,31,30
    ,3a,a1,41,24,3a,8b,41,24,b3
1b  58 data b1,22,22,a7,31,36,00
    ,05,09,0e,00,82,3a,89,31,31
a5  59 data 00,26,09,10,00,49,b2
    ,30,3a,81,5a,b2,31,a4,31,34
0f  60 data 3a,8b,41,24,b3,b1,43
    ,24,28,5a,29,a7,82,3a,89,31
77  61 data 31,00,2f,09,11,00,97
    ,50,2c,51,00,62,09,12,00,91
7e  62 data 5a,89,31,39,2c,32,34
    ,2c,32,38,2c,33,32,2c,33,36
ef  63 data 2c,36,35,2c,37,32,2c
    ,34,32,2c,34,34,2c,35,32,2c
a2  64 data 36,38,2c,37,35,2c,31
    ,35,38,2c,35,20,20,00,75,09
6c  65 data 13,00,44,b2,33,37,3a
    ,8b,59,b2,36,a7,44,b2,32,37
50  66 data 00,88,09,15,00,8b,58
    ,b3,44,a7,58,b2,58,aa,31,3a
57  67 data 89,33,39,00,a4,09,16
    ,00,8b,58,b2,44,a7,58,b2,30
68  68 data 3a,8b,59,b3,36,a7,59
    ,b2,59,aa,31,3a,89,33,39,00
6b  69 data b0,09,17,00,59,b2,30
    ,3a,89,33,39,00,c3,09,18,00
6a  70 data 8b,58,b1,30,a7,58,b2
    ,58,ab,31,3a,89,33,39,00,db
94  71 data 09,1a,00,58,b2,33,37
    ,3a,8b,59,b1,30,a7,59,b2,59
d9  72 data ab,31,3a,89,33,39,00
    ,ec,09,1b,00,58,b2,32,37,3a
a5  73 data 59,b2,36,3a,89,33,39
    ,00,fe,09,1c,00,44,b2,36,3a
43  74 data 8b,58,b1,32,37,a7,44
    ,b2,35,00,11,0a,1e,00,8b,59
19  75 data b3,44,a7,59,b2,59,aa
    ,31,3a,89,33,39,00,1d,0a,1f
cd  76 data 00,59,b2,30,3a,89,33
    ,39,00,2f,0a,20,00,44,b2,36
20  77 data 3a,8b,58,b1,32,37,a7
    ,44,b2,35,00,42,0a,22,00,8b
35  78 data 59,b1,30,a7,59,b2,59
    ,ab,31,3a,89,33,39,00,4e,0a
8e  79 data 23,00,59,b2,44,3a,89
    ,33,39,00,5a,0a,24,00,58,b2
ac  80 data 30,3a,59,b2,30,00,69
```

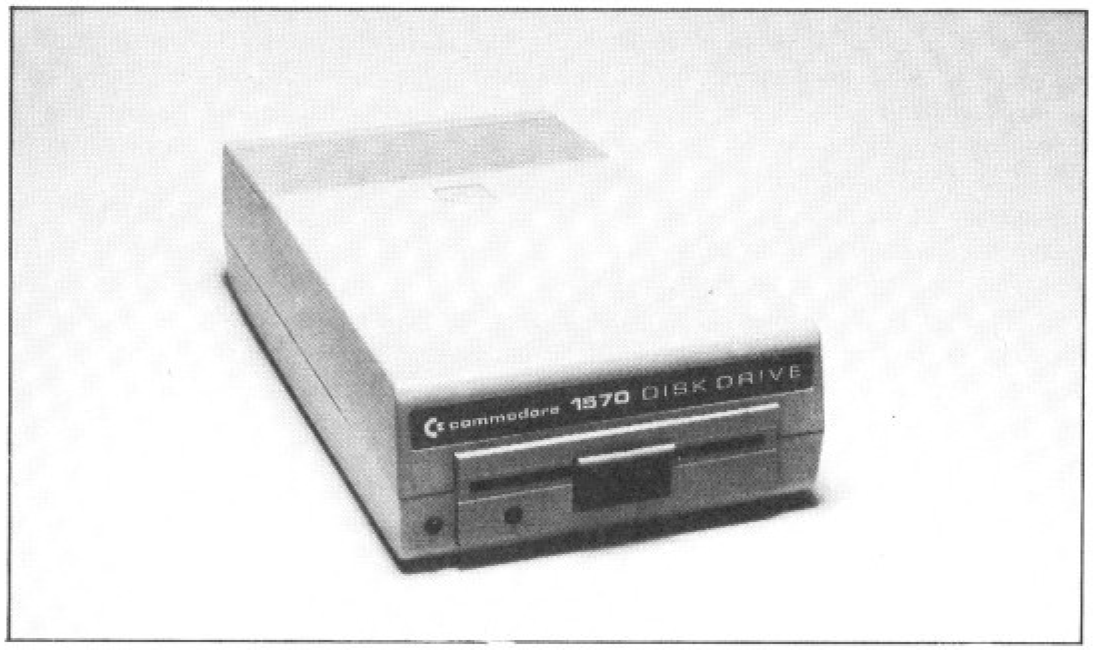
```
    ,0a,27,00,8b,51,b1,32,35,35
6a  81 data a7,31,30,35,00,8c,0a
    ,28,00,97,50,2c,51,3a,50,b2
34  82 data 31,33,30,35,aa,59,ac
    ,34,30,aa,58,3a,51,b2,c2,28
41  83 data 50,29,3a,51,31,b2,51
    ,00,b3,0a,29,00,97,32,2c,51
aa  84 data 3a,8d,31,30,03,e2,3a
    ,8d,31,35,37,3a,99,22,11,22
fb  85 data a6,33,35,29,22,9b,24
    ,22,57,24,3a,89,31,31,00,da
63  86 data 0a,2a,00,8d,31,35,37
    ,3a,99,22,11,22,a6,33,35,29
8f  87 data 3b,3a,56,b2,51,3a,4c
    ,b2,32,3a,8d,38,31,3a,51,b2
28  88 data 56,3a,89,33,39,00,e6
    ,0a,2c,00,8b,4f,50,a7,31,30
c4  89 data 35,00,1a,0b,2e,00,8d
    ,31,35,37,3a,99,a6,31,30,29
27  90 data 22,11,04,e2,c1,d2,c5
    ,a0,d9,cf,d5,a0,d3,d5,d2,c5
40  91 data 20,28,12,15,1c,d9,92
    ,05,2f,12,1c,ce,92,05,29,20
ad  92 data a4,9d,22,3b,00,32,0b
    ,2f,00,a1,41,24,3a,8b,41,24
d8  93 data b2,22,4e,22,a7,99,41
    ,24,3a,89,35,39,00,42,0b,30
42  94 data 00,8b,41,24,b3,b1,22
    ,59,22,a7,34,37,00,6c,0b,31
b0  95 data 00,99,41,24,3a,9f,31
    ,35,2c,38,2c,31,35,3a,9f,35
9c  96 data 2c,38,2c,35,2c,22,23
    ,22,3a,98,31,35,2c,22,42,2d
33  97 data 50,3a,35,2c,30,22,00
    ,a8,0b,32,00,9e,35,30,39,32
11  98 data 37,3a,98,31,35,2c,22
    ,55,32,3a,35,2c,22,53,44,24
ec  99 data 3b,54,3b,53,3a,84,31
    ,35,2c,41,24,2c,42,24,2c,43
21  100 data 24,2c,44,24,3a,8b,c
    5,28,41,24,29,b3,b1,30,a7,31
ec  101 data 30,32,00,bf,0b,33,00
    ,98,31,35,2c,22,49,22,3a,a0
d9  102 data 35,3a,a0,31,35,3a,89
    ,35,39,00,f1,0b,34,00,8d,31
9b  103 data 35,37,3a,99,a6,36,29
    ,22,11,04,e2,28,c8,c9,d4,20
e8  104 data d2,c5,d4,d5,d2,ce,20
    ,d4,cf,20,c5,ce,c4,20,d4,c5
0f  105 data d8,d4,20,cd,cf,c4,c5
    ,29,9b,22,00,11,0c,36,00,51
8b  106 data 31,b2,51,31,aa,31,32
    ,38,3a,8b,51,31,b1,32,35,35
67  107 data a7,51,31,b2,51,31,ab
    ,32,35,36,00,32,0c,37,00,97
7f  108 data 50,2c,51,31,3a,81,43
    ,b2,31,a4,35,35,3a,a1,41,24
cd  109 data 3a,8b,41,24,b3,b1,22
    ,22,a7,35,37,00,3c,0c,38,00
9a  110 data 82,3a,89,35,34,00,4e
    ,0c,39,00,8b,41,24,b3,b1,c7
80  111 data 28,31,33,29,a7,36,30
    ,00,57,0c,3a,00,97,50,2c,51
8d  112 data 00,64,0c,3b,00,8d,31
    ,35,35,3a,89,33,39,00,80,0c
55  113 data 3c,00,97,50,2c,c6,28
    ,41,24,29,3a,44,b2,33,37,3a
78  114 data 8b,59,b2,36,a7,44,b2
    ,32,37,00,93,0c,3d,00,8b,58
12  115 data b3,44,a7,58,b2,58,aa
    ,31,3a,89,36,34,00,af,0c,3e
ca  116 data 00,8b,58,b2,44,a7,58
    ,b2,30,3a,8b,59,b3,36,a7,59
f3  117 data b2,59,aa,31,3a,89,36
    ,34,00,b7,0c,3f,00,59,b2,30
bb  118 data 00,d9,0c,40,00,50,b2
    ,31,33,30,35,aa,59,ac,34,30
79  119 data aa,58,3a,51,b2,c2,28
    ,50,29,3a,51,31,b2,51,3a,89
76  120 data 35,34,00,fa,0c,41,00
    ,8d,31,35,37,3a,8b,4f,50,a7
22  121 data 4d,42,b2,4d,42,aa,31
    ,3a,9e,35,30,38,38,39,3a,89

10  122 data 31,33,34,00,08,0d,43
    ,00,53,b2,53,aa,31,3a,89,39
d2  123 data 39,00,18,0d,44,00,4f
    ,50,b2,30,3a,9e,35,30,38,38
5f  124 data 39,00,3b,0d,46,00,8d
    ,31,35,37,3a,99,22,11,22,a6
33  125 data 31,30,29,3b,3a,56,b2
    ,54,3a,4c,b2,31,3a,8d,38,31
46  126 data 3a,54,b2,56,00,77,0d
    ,47,00,8d,31,35,37,3a,99,22
93  127 data 11,22,a6,32,34,29,3b
    ,3a,56,b2,53,3a,4c,b2,31,3a
45  128 data 8d,38,31,3a,53,b2,56
    ,3a,9f,31,2c,38,2c,31,35,3a
e9  129 data 98,31,2c,22,49,22,53
    ,44,24,3a,a0,31,3a,89,39,39
1b  130 data 00,98,0d,48,00,8d,31
    ,35,37,3a,8b,4f,50,a7,4d,42
f3  131 data b2,4d,42,ab,31,3a,9e
    ,35,30,38,38,39,3a,89,31,33
a0  132 data 34,00,a6,0d,4a,00,53
    ,b2,53,ab,31,3a,89,39,39,00
d3  133 data b1,0d,4b,00,8b,4f,50
    ,a7,31,30,00,e3,0d,4d,00,8d
4f  134 data 31,35,37,3a,54,b2,c2
    ,28,50,29,3a,44,b2,33,37,3a
6f  135 data 8b,59,b2,36,a7,44,b2
    ,32,37,3a,8b,58,b2,44,a7,53
81  136 data b2,c2,28,31,33,30,35
    ,29,3a,89,39,39,00,f9,0d,4e
ad  137 data 00,8b,58,b2,44,a7,53
    ,b2,c2,28,50,aa,33,29,3a,89
26  138 data 39,39,00,0a,0e,4f,00
    ,53,b2,c2,28,50,aa,31,29,3a
0d  139 data 89,39,39,00,48,0e,51
    ,00,99,c8,28,22,9b,20,04,e2
16  140 data 22,2c,4c,aa,33,29,3b
    ,c8,28,22,9d,04,e2,22,2c,4c
1b  141 data aa,32,29,3b,3a,81,43
    ,b2,30,a4,31,3a,54,24,28,43
57  142 data 29,b2,22,22,3a,82,3a
    ,57,b2,30,3a,56,31,b2,30,00
eb  143 data 60,0e,52,00,99,22,92
    ,22,3b,3a,46,b2,a8,46,3a,8b
29  144 data 46,a7,99,22,12,22,3b
    ,00,8b,0e,53,00,99,22,20,9d
8e  145 data 22,3b,3a,81,43,b2,31
    ,a4,31,30,3a,a1,41,24,3a,8b
be  146 data 41,24,b3,b1,22,22,a7
    ,99,22,92,20,9d,22,3b,3a,89
4d  147 data 38,35,00,95,0e,54,00
    ,82,3a,89,38,32,00,bf,0e,55
81  148 data 00,8b,41,24,b2,c7,28
    ,32,30,29,a7,8b,57,b1,30,a7
7f  149 data 99,22,9d,20,9d,22,3b
    ,3a,57,b2,57,ab,31,3a,54,24
09  150 data 28,57,29,b2,22,22,00
    ,d0,0e,56,00,8b,41,24,b2,c7
c4  151 data 28,31,33,29,a7,39,32
    ,00,e3,0e,57,00,8b,41,24,b2
d8  152 data 22,24,22,af,57,b2,30
    ,a7,39,31,00,0b,0f,58,00,8b
95  153 data 28,41,24,b3,22,30,22
    ,b0,41,24,b1,22,39,22,29,af
09  154 data 28,41,24,b3,22,41,22
    ,b0,41,24,b1,22,46,22,29,a7
b2  155 data 38,32,00,27,0f,59,00
    ,8b,41,24,b1,22,40,22,a7,41
ed  156 data 24,b2,c7,28,c6,28,41
    ,24,29,aa,31,32,38,29,00,33
67  157 data 0f,5a,00,8b,57,b1,4c
    ,a7,38,32,00,52,0f,5b,00,54
b7  158 data 24,28,57,29,b2,41,24
    ,3a,57,b2,57,aa,31,3a,99,22
23  159 data 92,22,41,24,3b,3a,89
    ,38,32,00,62,0f,5c,00,8b,54
1b  160 data 24,28,30,29,b2,22,22
    ,a7,8e,00,87,0f,5d,00,56,b2
dc  161 data 30,3a,4c,b2,30,3a,8b
    ,54,24,28,30,29,b2,22,24,22
d7  162 data a7,54,24,28,30,29,b2
    ,22,22,3a,4c,b2,ab,31,00,c0
cc  163 data 0f,5e,00,41,24,b2,22
    ,22,3a,81,43,b2,30,a4,57,ab
08  164 data 31,3a,41,24,b2,41,24

    ,aa,54,24,28,43,29,3a,82,3a
dc  165 data 41,24,b2,c9,28,22,30
    ,04,e2,22,aa,41,24,2c,35,29
2e  166 data 3a,8b,4c,a7,39,36,00
    ,e9,0f,5f,00,81,43,b2,30,a4
af  167 data 33,3a,56,b2,56,aa,28
    ,31,30,ae,43,29,ac,c5,28,ca
cf  168 data 28,41,24,2c,35,ab,43
    ,2c,31,29,29,3a,82,3a,8e,00
cc  169 data 19,10,60,00,81,43,b2
    ,30,a4,31,3a,81,57,b2,30,a4
95  170 data 31,3a,97,32,aa,57,2c
    ,c6,28,ca,28,41,24,2c,28,34
68  171 data ab,28,43,ac,32,29,29
    ,aa,57,2c,31,29,29,3a,82,00
02  172 data 39,10,61,00,9e,35,31
    ,31,35,30,3a,56,b2,56,aa,28
e2  173 data 32,35,36,ae,43,29,ac
    ,c2,28,34,29,3a,82,3a,8e,00
dd  174 data 74,10,63,00,9e,35,30
    ,38,38,39,3a,8d,31,35,37,3a
8f  175 data 99,22,11,22,a6,39,29
    ,22,90,9b,20,03,e2,9d,03,e2
03  176 data 22,54,3a,8d,31,35,37
    ,3a,99,22,11,22,a6,32,33,29
17  177 data 22,20,03,e2,9d,03,e2
    ,9b,22,53,00,af,10,64,00,9f
0b  178 data 31,35,2c,38,2c,31,35
    ,3a,9f,35,2c,38,2c,35,2c,22
86  179 data 23,22,3a,98,31,35,2c
    ,22,55,31,3a,35,2c,22,53,44
dc  180 data 24,3b,54,3b,53,3a,84
    ,31,35,2c,41,24,2c,42,24,2c
e7  181 data 43,24,2c,44,24,00,e4
    ,10,65,00,8b,c5,28,41,24,29
51  182 data b2,30,a7,9e,35,30,38
    ,34,33,3a,50,b2,31,33,30,35
d4  183 data 3a,51,b2,c2,28,50,29
    ,3a,58,b2,30,3a,59,b2,30,3a
20  184 data a0,35,3a,a0,31,35,3a
    ,89,33,39,00,24,11,66,00,99
d9  185 data 22,13,22,3b,3a,81,58
    ,b2,31,a4,34,3a,99,22,20,28
26  186 data e2,22,3b,3a,82,00,59
    ,11,67,00,99,22,93,11,05,12
e7  187 data c4,c9,d3,cb,20,c5,d2
    ,d2,cf,d2,3a,22,3a,99,22,05
0c  188 data 11,22,41,24,22,2c,22
    ,42,24,22,2c,22,43,24,22,2c
f9  189 data 22,44,24,3a,a0,35,3a
    ,a0,31,35,00,6c,11,68,00,a1
05  190 data 41,24,3a,8b,41,24,b2
    ,22,22,a7,31,30,34,00,82,11
0a  191 data 69,00,54,b2,31,38,3a
    ,53,b2,30,3a,8d,31,36,34,3a
b5  192 data 89,36,38,00,8f,11,9b
    ,00,99,22,91,22,54,24,3a,8e
39  193 data 00,a7,11,9d,00,99,22
    ,13,11,0d,e2,22,3a,8e,00,b8
6c  194 data 11,9e,00,8d,31,35,37
    ,3a,99,22,11,04,e2,22,00,f0
46  195 data 11,9f,00,81,43,b2,31
    ,a4,35,3a,99,22,20,27,e2,22
ec  196 data 3a,82,00,16,12,a0,00
    ,97,32,2c,50,31,3a,8d,31,30
80  197 data 03,e2,3a,99,22,05,91
    ,91,20,44,20,c3,30,22,57,24
f0  198 data 22,91,05,e2,22,22,00
    ,52,12,a1,00,9e,35,31,30,33
17  199 data 35,3a,9e,35,31,31,38
    ,31,3a,8d,31,36,34,3a,9e,35
09  200 data 31,30,37,37,3a,50,b2
    ,31,33,30,35,3a,51,b2,c2,28
5c  201 data 50,29,3a,58,b2,30,3a
    ,59,b2,30,3a,8b,4f,50,a7,31
40  202 data 36,33,00,80,12,a2,00
    ,8d,31,35,37,3a,99,22,11,22
a3  203 data a6,39,29,22,9b,20,03
    ,e2,9d,03,e2,22,54,a6,31,30
96  204 data 29,22,90,20,03,e2,9d
    ,03,e2,9b,22,53,3a,89,33,39
b7  205 data 00,b6,12,a3,00,97,32
    ,2c,4d,42,3a,8d,31,3a,99,22
70  206 data 13,22,a3,31,30,29,22
    ,11,05,e2,20,20,05,cd,c5,cd
```

```
f2  207 data cf,d2,d9,20,c2,cc,cf
     ,c3,cb,3a,20,9b,24,22,57,24
17  208 data 3a,89,33,39,00,f3,12
     ,a4,00,97,35,33,32,38,30,2c
18  209 data 31,35,3a,97,35,33,32
     ,38,31,2c,31,32,3a,99,22,93
40  210 data 0e,22,a6,39,29,22,11
     ,12,90,b0,c0,15,e2,ae,22,00
4f  211 data 35,13,a5,00,99,a6,39
     ,29,22,12,dd,20,d4,d2,c1,c3
9e  212 data cb,2f,d3,c5,c3,d4,cf
     ,d2,20,c5,c4,c9,d4,cf,d2,20
91  213 data dd,22,3a,99,a6,39,29
     ,22,12,dd,20,04,e2,c2,59,20
b1  214 data 20,cc,45,53,20,c1,4c
     ,4c,41,4e,20,04,e2,dd,22,00
21  215 data 5b,13,a6,00,99,a6,39
     ,29,22,12,ad,c0,15,e2,bd,92
2c  216 data 22,3a,99,00,8d,13,a7
     ,00,99,22,05,b0,c0,26,e2,ae
11  217 data 22,3b,00,c3,13,a8,00
     ,99,22,dd,05,12,20,20,9b,20
29  218 data 24,e2,05,92,dd,22,3b
     ,00,01,14,a9,00,81,43,b2,31
8b  219 data a4,35,3a,99,22,dd,9b
     ,12,20,26,e2,05,92,dd,22,3b
45  220 data 3a,82,00,36,14,aa,00
     ,99,22,dd,9b,12,20,1c,e2,05
2f  221 data 92,2e,0a,e2,dd,22,3b
     ,00,66,14,ab,00,99,22,ab,c0
a7  222 data 0c,e2,b2,c0,0d,e2,b2
     ,c0,0b,e2,b3,22,00,9e,14,ac
62  223 data 00,99,22,91,dd,90,20
     ,20,d4,d2,c1,c3,cb,3a,20,04
bf  224 data e2,05,dd,90,20,20,d3
     ,c5,c3,d4,cf,d2,3a,20,04,e2
52  225 data 05,dd,90,20,20,c2,d9
     ,d4,c5,3a,20,04,e2,05,dd,22
84  226 data 3a,00,cf,14,ad,00,99
     ,22,91,ad,c0,0c,e2,b1,c0,0d
0c  227 data e2,b1,c0,0b,e2,bd,22
     ,00,06,15,ae,00,99,3a,99,3a
3f  228 data 99,22,20,03,e2,97,12
     ,d2,92,05,3a,2d,d2,c5,c1,c4
2d  229 data 20,03,e2,d3,c5,c3,d4
     ,cf,d2,20,04,e2,12,97,ca,92
08  230 data 05,3a,2d,ca,d5,cd,d0
     ,a0,cc,c9,ce,cb,22,00,39,15
2c  231 data af,00,99,22,20,03,e2
     ,97,12,d7,92,05,3a,2d,d7,d2
e4  232 data c9,d4,c5,a0,20,d3,c5
     ,c3,d4,cf,d2,20,04,e2,12,97
38  233 data d4,92,05,3a,2d,d4,c5
     ,d8,d4,a0,cd,cf,c4,c5,22,00
8d  234 data 6f,15,b0,00,99,22,20
     ,03,e2,97,12,2a,92,05,3a,2d
3a  235 data c3,c8,c1,ce,c7,c5,a0
     ,c2,d9,d4,c5,20,06,e2,12,97
2f  236 data d8,92,05,3a,2d,cd,2f
     ,c3,20,20,cd,cf,ce,c9,d4,cf
d0  237 data d2,22,00,a4,15,b1,00
     ,99,22,20,03,e2,97,12,2b,92
ab  238 data 05,3a,2d,c6,27,d7,c1
     ,d2,c4,a0,d3,c5,c3,d4,cf,d2
56  239 data 20,04,e2,12,97,2d,92
     ,05,3a,2d,c2,c1,c3,cb,a0,d3
96  240 data c5,c3,d4,cf,d2,22,00
     ,b1,15,b2,00,9e,35,30,38,38
88  241 data 39,3a,8e,00,d1,15,e8
     ,03,9e,35,31,03,e2,39,3a,57
a1  242 data 24,b2,c7,28,c2,28,33
     ,29,29,aa,c7,28,c2,28,34,29
aa  243 data 29,3a,8e,00,8e,00,ee
     ,ff,e2,ee,ff,e2,ee,ff,e2,ee
64  244 data ff,e2,ee,ff,e2,ee,ff
     ,e2,ee,ff,e2,ee,ff,e2,ee,ff
6d  245 data e2,ee,ff,e2,ee,ff,e2
     ,ee,08,e2,20,44,e5,a9,00,a8
ae  246 data 85,fb,85,fd,8d,20,d0
     ,8d,21,d0,a9,30,85,fc,a9,c6
f4  247 data 85,fe,b1,fb,91,fd,c8
     ,d0,f9,e6,fc,e6,fe,a5,fe,c9
a6  248 data d0,d0,ef,60,ee,ff,e2
     ,ee,ff,e2,ee,ff,e2,ee,ff,e2
31  249 data ee,ff,e2,ee,ff,e2,ee
     ,ff,e2,ee,ff,e2,ee,ff,e2,ee
12  250 data ff,e2,ee,ff,e2,ee,ff
     ,e2,ee,ff,e2,ee,ff,e2,ee,ff
49  251 data e2,ee,b0,e2,a2,19,a0
     ,05,86,fb,84,fc,a2,05,20,c6
ea  252 data ff,a0,00,84,03,a2,26
     ,86,02,20,cf,ff,91,fb,c8,d0
5b  253 data 02,e6,fc,c6,03,f0,08
     ,c6,02,d0,ee,c8,c8,d0,e6,4c
81  254 data cc,ff,a2,19,a0,05,86
     ,fb,84,fc,a9,20,a0,00,84,03
bd  255 data a2,26,86,02,91,fb,c8
     ,d0,02,e6,fc,c6,03,f0,08,c6
eb  256 data 02,d0,f1,c8,c8,d0,e9
     ,60,a2,19,a0,05,86,fb,84,fc
11  257 data a2,05,20,c9,ff,a0,00
     ,84,03,a2,26,86,02,b1,fb,20
15  258 data d2,ff,c8,d0,02,e6,fc
     ,c6,03,f0,08,c6,02,d0,ee,c8
63  259 data c8,d0,e6,4c,cc,ff,a2
     ,05,20,c9,ff,a0,00,b9,4c,c7
b3  260 data 20,d2,ff,c8,c0,06,d0
     ,f5,a0,00,ae,51,c7,b1,fb,20
3b  261 data d2,ff,c8,ca,f0,0c,e6
     ,03,a5,03,c9,26,d0,ef,c8,c8
a6  262 data d0,eb,4c,cc,ff,4d,2d
     ,57,00,05,01,00,ff,08,e2,a2
28  263 data 19,a0,05,86,fb,84,fc
     ,a0,00,84,03,a2,00,a9,26,85
ad  264 data 02,b1,fb,9d,00,c0,e8
     ,c8,d0,02,e6,fc,c6,03,f0,08
78  265 data c6,02,d0,ed,c8,c8,d0
     ,e5,60,a2,19,a0,05,86,fb,84
f6  266 data fc,a0,00,84,03,a2,00
     ,a9,26,85,02,bd,00,c0,91,fb
f7  267 data e8,c8,d0,02,e6,fc,c6
     ,03,f0,08,c6,02,d0,ed,c8,c8
8e  268 data d0,e5,60,a5,02,48,29
     ,0f,20,c4,c7,85,04,68,4a,04
fc  269 data e2,20,c4,c7,85,03,60
     ,18,69,30,c9,3a,90,02,69,86
40  270 data 60,a5,02,20,e3,c7,0a
     ,04,e2,85,04,a5,03,20,e3,c7
7b  271 data 05,04,85,04,60,38,e9
     ,30,c9,0a,90,02,e9,87,60,ad
78  272 data e6,cf,8d,16,03,ad,e7
     ,cf,8d,17,03,a9,80,20,90,ff
84  273 data 00,d8,68,8d,3e,02,68
     ,8d,3d,02,68,8d,3c,02,68,8d
f9  274 data 3b,02,68,aa,68,a8,38
     ,8a,e9,02,8d,3a,02,98,e9,00
49  275 data 8d,39,02,ba,8e,3f,02
     ,20,57,cd,a2,42,a9,2a,20,57
ec  276 data ca,a9,52,d0,34,e6,c1
     ,d0,06,e6,c2,d0,02,e6,26,60
a3  277 data 20,cf,ff,c9,0d,d0,f8
     ,68,68,a9,07,20,d2,ff,a9,00
58  278 data 85,26,a2,0d,a9,2e,20
     ,57,ca,a9,07,20,d2,ff,20,3e
be  279 data c8,c9,2e,f0,f9,c9,20
     ,f0,f5,a2,0e,dd,b7,cf,d0,0c
eb  280 data 8a,0a,aa,bd,c7,cf,48
     ,bd,c6,cf,48,60,ca,10,ec,4c
8e  281 data ed,ca,a5,c1,8d,3a,02
     ,a5,c2,8d,39,02,60,a9,08,85
c6  282 data 1d,a0,00,20,54,cd,b1
     ,c1,20,48,ca,20,33,c8,c6,1d
8c  283 data d0,f1,60,20,88,ca,90
     ,0b,a2,00,81,c1,03,e2,f0,03
0a  284 data 4c,ed,ca,20,33,c8,c6
     ,1d,60,a9,3b,85,c1,a9,02,85
24  285 data c2,a9,05,60,98,48,20
     ,57,cd,68,a2,2e,4c,57,ca,a9
28  286 data 07,20,d2,ff,a2,00,bd
     ,ea,cf,20,d2,ff,e8,e0,16,d0
e6  287 data f5,a0,3b,20,c2,c8,ad
     ,39,02,20,48,ca,ad,3a,02,20
7e  288 data 48,ca,20,b7,c8,20,8d
     ,c8,f0,5c,20,3e,c8,20,79,ca
9c  289 data 90,33,20,69,ca,20,3e
     ,c8,20,79,ca,90,28,20,69,ca
d9  290 data a9,07,20,d2,ff,20,e1
     ,ff,f0,3c,a6,26,d0,38,a5,c3
7d  291 data c5,c1,a5,c4,e5,c2,90
     ,2e,a0,3a,20,c2,c8,20,41,ca
63  292 data 20,8b,c8,f0,e0,4c,ed
     ,ca,20,79,ca,90,03,20,80,c8
04  293 data 20,b7,c8,d0,07,20,79
     ,ca,90,eb,a9,08,85,1d,20,3e
f3  294 data c8,20,a1,c8,d0,f8,4c
     ,47,c8,20,cf,ff,c9,0d,f0,0c
28  295 data c9,20,d0,d1,20,79,ca
     ,90,03,20,80,c8,a9,07,20,d2
1c  296 data ff,ae,3f,02,9a,78,ad
     ,39,02,48,ad,3a,02,48,ad,3b
de  297 data 02,48,ad,3c,02,ae,3d
     ,02,ac,3e,02,40,a9,07,20,d2
a7  298 data ff,ae,3f,02,9a,60,ea
     ,ea,a0,01,84,ba,84,b9,88,84
1b  299 data b7,84,90,84,93,a9,40
     ,85,bb,a9,02,85,bc,20,cf,ff
ae  300 data c9,20,f0,f9,c9,0d,f0
     ,38,c9,22,d0,14,20,cf,ff,c9
5f  301 data 22,f0,10,c9,0d,f0,29
     ,91,bb,e6,b7,c8,c0,11,d0,ec
0a  302 data 4c,ed,ca,20,cf,ff,c9
     ,0d,f0,16,c9,2c,d0,dc,20,88
0a  303 data ca,29,0f,f0,e9,c9,03
     ,f0,e5,85,ba,20,cf,ff,c9,0d
94  304 data 60,6c,30,03,6c,32,03
     ,20,96,c9,d0,d4,a9,07,20,d2
5b  305 data ff,a9,00,20,ef,c9,a5
     ,90,29,10,d0,c4,4c,47,c8,20
c9  306 data 96,c9,c9,2c,d0,ba,20
     ,79,ca,20,69,ca,20,cf,ff,c9
14  307 data 2c,d0,ad,20,79,ca,a5
     ,c1,85,ae,a5,c2,85,af,20,69
8f  308 data ca,20,cf,ff,c9,0d,d0
     ,98,a9,07,20,d2,ff,20,f2,c9
11  309 data 4c,47,c8,a5,c2,20,48
     ,ca,a5,c1,48,4a,04,e2,20,60
2d  310 data ca,aa,68,29,0f,20,60
     ,ca,48,8a,20,d2,ff,68,4c,d2
4d  311 data ff,09,30,c9,3a,90,02
     ,69,06,60,a2,02,b5,c0,48,b5
08  312 data c2,95,c0,68,95,c2,ca
     ,d0,f3,60,20,88,ca,90,02,85
f1  313 data c2,20,88,ca,90,02,85
     ,c1,60,a9,00,85,2a,20,3e,c8
07  314 data c9,20,d0,09,20,3e,c8
     ,c9,20,d0,0e,18,60,20,af,ca
cc  315 data 0a,04,e2,85,2a,20,3e
     ,c8,20,af,ca,05,2a,38,60,c9
76  316 data 3a,90,02,69,08,29,0f
     ,60,a2,02,2c,a2,00,b4,c1,d0
73  317 data 08,b4,c2,d0,02,e6,26
     ,d6,c2,d6,c1,60,20,3e,c8,c9
be  318 data 20,f0,f9,60,a9,00,8d
     ,00,01,20,cc,ca,20,8f,ca,20
b6  319 data 7c,ca,90,09,60,20,3e
     ,c8,20,79,ca,b0,de,ae,3f,02
d0  320 data 9a,a9,07,20,d2,ff,a9
     ,3f,20,d2,ff,4c,47,c8,20,54
89  321 data cd,ca,d0,fa,60,e6,c3
     ,d0,02,e6,c4,60,a2,02,b5,c0
ff  322 data 48,b5,27,95,c0,68,95
     ,27,ca,d0,f3,60,a5,c3,a4,c4
1c  323 data 38,e9,02,b0,0e,88,90
     ,0b,a5,28,a4,29,4c,33,cb,a5
aa  324 data c3,a4,c4,38,e5,c1,85
     ,1e,98,e5,c2,a8,05,1e,60,20
8e  325 data d4,ca,20,69,ca,20,e5
     ,ca,20,0c,cb,20,e5,ca,20,2f
d9  326 data cb,20,69,ca,90,15,a6
     ,26,d0,64,20,28,cb,90,5f,a1
cc  327 data c1,81,c3,20,05,cb,20
     ,33,c8,d0,eb,20,28,cb,18,a5
32  328 data 1e,65,c3,85,c3,98,65
     ,c4,85,c4,20,0c,cb,a6,26,d0
e9  329 data 3d,a1,c1,81,c3,20,28
     ,cb,b0,34,20,b8,ca,20,bb,ca
6b  330 data 4c,7d,cb,20,d4,ca,20
     ,69,ca,20,e5,ca,20,69,ca,20
84  331 data 3e,c8,20,88,ca,90,14
     ,85,1d,a6,26,d0,11,20,2f,cb
bb  332 data 90,0c,a5,1d,81,c1,20
     ,33,c8,d0,ee,4c,ed,ca,4c,47
67  333 data c8,20,d4,ca,20,69,ca
     ,20,e5,ca,20,69,ca,20,3e,c8
00  334 data a2,00,20,3e,c8,c9,27
     ,d0,14,20,3e,c8,9d,10,02,e8
1e  335 data 20,cf,ff,c9,0d,f0,22
     ,e0,20,d0,f1,f0,1c,8e,00,01
```

```
dd  336 data 20,8f,ca,90,c6,9d,10
    ,02,e8,20,cf,ff,c9,0d,f0,09
5c  337 data 20,88,ca,90,b6,e0,20
    ,d0,ec,86,1c,a9,07,20,d2,ff
48  338 data 20,57,cd,a2,00,a0,00
    ,b1,c1,dd,10,02,d0,0c,c8,e8
4c  339 data e4,1c,d0,f3,20,41,ca
    ,20,54,cd,20,33,c8,a6,26,d0
ee  340 data 8d,20,2f,cb,b0,dd,4c
    ,47,c8,20,d4,ca,85,20,a5,c2
01  341 data 85,21,a2,00,86,28,a9
    ,93,20,d2,ff,a9,07,20,d2,ff
b6  342 data a9,16,85,1d,20,6a,cc
    ,20,ca,cc,85,c1,84,c2,c6,1d
06  343 data d0,f2,a9,91,20,d2,ff
    ,4c,47,c8,a0,2c,20,c2,c8,20
7b  344 data 54,cd,20,41,ca,20,54
    ,cd,a2,00,a1,c1,20,d9,cc,48
dc  345 data 20,1f,cd,68,20,35,cd
    ,a2,06,e0,03,d0,12,a4,1f,f0
2b  346 data 0e,a5,2a,c9,e8,b1,c1
    ,b0,1c,20,c2,cc,88,d0,f2,06
64  347 data 2a,90,0e,bd,2a,cf,20
    ,a5,cd,bd,30,cf,f0,03,20,a5
50  348 data cd,ca,d0,d5,60,20,cd
    ,cc,aa,e8,d0,01,c8,98,20,c2
e7  349 data cc,8a,86,1c,20,48,ca
    ,a6,1c,60,a5,1f,38,a4,c2,aa
48  350 data 10,01,88,65,c1,90,01
    ,c8,60,a8,4a,90,0b,4a,b0,17
e9  351 data c9,22,f0,13,29,07,09
    ,80,4a,aa,bd,d9,ce,b0,04,4a
7d  352 data 04,e2,29,0f,d0,04,a0
    ,80,a9,00,aa,bd,1d,cf,85,2a
57  353 data 29,03,85,1f,98,29,8f
    ,aa,98,a0,03,e0,8a,f0,0b,4a
21  354 data 90,08,4a,4a,09,20,88
    ,d0,fa,c8,88,d0,f2,60,b1,c1
73  355 data 20,c2,cc,a2,01,20,fe
    ,ca,c4,1f,c8,90,f1,a2,03,c0
ef  356 data 04,90,f2,60,a8,b9,37
    ,cf,85,28,b9,77,cf,85,29,a9
9d  357 data 00,a0,05,06,29,26,28
    ,2a,88,d0,f8,69,3f,20,d2,ff
dd  358 data ca,d0,ec,a9,20,2c,a9
    ,0d,4c,d2,ff,20,d4,ca,20,69
e6  359 data ca,20,e5,ca,20,69,ca
    ,a2,00,86,28,a9,07,20,d2,ff
ca  360 data 20,57,cd,20,72,cc,20
    ,ca,cc,85,c1,84,c2,20,e1,ff
d2  361 data f0,05,20,2f,cb,b0,e9
    ,4c,47,c8,20,d4,ca,a9,03,85
cc  362 data 1d,20,3e,c8,20,a1,c8
    ,d0,f8,a5,20,85,c1,a5,21,85
3e  363 data c2,4c,46,cc,c5,28,f0
    ,03,20,d2,ff,60,20,d4,ca,20
29  364 data 69,ca,8e,11,02,a2,03
    ,20,cc,ca,48,ca,d0,f9,a2,03
11  365 data 68,38,e9,3f,a0,05,4a
    ,6e,11,02,6e;10,02,88,d0,f6
6e  366 data ca,d0,ed,a2,02,20,cf
    ,ff,c9,0d,f0,1e,c9,20,f0,f5
85  367 data 20,d0,ce,b0,0f,20,9c
    ,ca,a4,c1,84,c2,85,c1,a9,30
79  368 data 9d,10,02,e8,9d,10,02
    ,e8,d0,db,86,28,a2,00,86,26
5a  369 data f0,04,e6,26,f0,75,a2
    ,00,86,1d,a5,26,20,d9,cc,a6
31  370 data 2a,86,29,aa,bc,37,cf
    ,bd,77,cf,20,b9,ce,d0,e3,a2
00  371 data 06,e0,03,d0,19,a4,1f
    ,f0,15,a5,2a,c9,e8,a9,30,b0
91  372 data 21,20,bf,ce,d0,cc,20
    ,c1,ce,d0,c7,88,d0,eb,06,2a
3a  373 data 90,0b,bc,30,cf,bd,2a
    ,cf,20,b9,ce,d0,b5,ca,d0,d1
84  374 data f0,0a,20,b8,ce,d0,ab
    ,20,b8,ce,d0,a6,a5,28,c5,1d
59  375 data d0,a0,20,69,ca,a4,1f
    ,f0,28,a5,29,c9,9d,d0,1a,20
29  376 data 1c,cb,90,0a,98,d0,04
    ,a5,1e,10,0a,4c,ed,ca,c8,d0
90  377 data fa,a5,1e,10,f6,a4,1f
    ,d0,03,b9,c2,00,91,c1,88,d0
e7  378 data f8,a5,26,91,c1,20,ca
    ,cc,85,c1,84,c2,a9,07,20,d2
dc  379 data ff,a0,41,20,c2,c8,20
    ,54,cd,20,41,ca,20,54,cd,a9
4c  380 data 07,20,d2,ff,4c,b0,cd
    ,a8,20,bf,ce,d0,11,98,f0,0e
97  381 data 86,1c,a6,1d,dd,10,02
    ,08,e8,86,1d,a6,1c,28,60,c9
56  382 data 30,90,03,c9,47,60,38
    ,60,40,02,45,03,d0,08,40,09
fa  383 data 30,22,45,33,d0,08,40
    ,09,40,02,45,33,d0,08,40,09
10  384 data 40,02,45,b3,d0,08,40
    ,09,00,22,44,33,d0,8c,44,00
7a  385 data 11,22,44,33,d0,8c,44
    ,9a,10,22,44,33,d0,08,40,09
e5  386 data 10,22,44,33,d0,08,40
    ,09,62,13,78,a9,00,21,81,82
f5  387 data 00,00,59,4d,91,92,86
    ,4a,85,9d,2c,29,2c,23,28,24
9d  388 data 59,00,58,24,24,00,1c
    ,8a,1c,23,5d,8b,1b,a1,9d,8a
cd  389 data 1d,23,9d,8b,1d,a1,00
    ,29,19,ae,69,a8,19,23,24,53
85  390 data 1b,23,24,53,19,a1,00
    ,1a,5b,5b,a5,69,24,24,ae,ae
1c  391 data a8,ad,29,00,7c,00,15
    ,9c,6d,9c,a5,69,29,53,84,13
48  392 data 34,11,a5,69,23,a0,d8
    ,62,5a,48,26,62,94,88,54,44
9e  393 data c8,54,68,44,e8,94,00
    ,b4,08,84,74,b4,28,6e,74,f4
ea  394 data cc,4a,72,f2,a4,8a,00
    ,aa,a2,a2,74,03,e2,72,44,68
e2  395 data b2,32,b2,00,22,00,1a
    ,1a,26,26,72,72,88,c8,c4,ca
46  396 data 26,48,44,44,a2,c8,3a
    ,3b,52,4d,47,58,4c,53,54,46
33  397 data 48,44,50,2c,41,42,c9
    ,35,c9,cc,c8,f7,c8,56,c9,89
f2  398 data c9,f4,c9,0c,ca,3e,cb
    ,92,cb,c0,cb,38,cc,5b,cd,8a
a3  399 data cd,ac,cd,46,c8,ff,c7
    ,ed,c7,0d,20,03,e2,50,43,20
d7  400 data 20,53,52,20,41,43,20
    ,58,52,20,59,52,20,53,ee,a2
87  401 data 00,bd,81,1e,9d,10,01
    ,e8,d0,f7,a2,34,bd,e8,1e,9d
0e  402 data 00,03,e8,d0,f7,ea,ea
    ,a2,ff,9a,a9,00,85,2d,38,e9
c3  403 data 01,85,fe,a9,3a,85,2e
    ,e9,00,85,ff,a9,3e,85,fc,a9
73  404 data 1e,85,fd,ad,21,d0,85
    ,fb,4c,52,01,00,00,00,00,00
da  405 data 00,00,19,08,02,00,8b
    ,c2,28,35,31,36,30,33,29,b3
d7  406 data b1,39,36,a7,9e,38,34
    ,30,30,00,4e,08,03,00,86,43
e6  407 data 24,48,c6,fc,a5,fc,c9
    ,ff,d0,02,c6,fd,68,60,48,ad
d2  408 data 20,d0,49,03,8d,20,d0
    ,c6,fe,a5,fe,c9,ff,d0,02,c6
05  409 data ff,68,60,a0,00,b1,fc
    ,c9,e2,d0,18,ea,ea,ea,20,30
52  410 data 01,b1,fc,aa,20,30,01
    ,b1,fc,91,fe,20,3d,01,ca,d0
0b  411 data f8,f0,05,91,fe,20,3d
    ,01,a6,fc,ca,86,fc,e0,ff,d0
30  412 data 02,c6,fd,a5,fe,c9,1f
    ,a5,ff,e9,08,b0,c8,a5,fb,8d
8f  413 data 20,d0,a9,37,85,01,20
    ,60,a6,20,8e,a6,a2,1f,bd,10
b4  414 data 01,9d,00,08,ca,d0,f7
    ,a9,e2,4c,34,03,4c,ae,a7,00
3d  415 data a9,00,a8,85,fb,85,fd
    ,a9,80,85,fc,a9,08,85,fe,b1
6f  416 data fb,91,fd,c8,d0,f9,e6
    ,fc,e6,fe,a5,fc,c9,a0,d0,ef
83  417 data a5,ba,aa,a8,20,ba,ff
    ,a9,0c,a2,64,a0,97,20,bd,ff
b7  418 data a9,01,85,2b,a9,08,85
    ,2c,a9,2b,a2,b2,a0,1f,20,d8
d1  419 data ff,4c,66,fe,54,52,41
    ,43,4b,20,45,44,49,54,4f,52
9f  420 :
9c  421 :
27  422 ::::::::::::::::::::::::::
    ::::::::::::
cc  423 :
                :
d7  424 : track/sector editor  fo
    r cbm 64    :
ca  425 :
                :
87  426 :       written by les all
    an          :
d0  427 :
                :
e5  428 ::::::::::::::::::::::::::
    ::::::::::::
```



DIR Cover

It can sometimes be a real pain when trying to find one specific program that's hidden away on one of over 100 disks. DIR COVER will make life much easier as it will produce, a disk cover on a printer, that you can cut out and make. A total list of all the programs found on the disk will be printed on the cover, together with lots more useful information about the programs.

By Elizamac Mackenzie

Dir cover is designed for the Gemini 10X printer, although slight modifications can be made for other printers. The program is simple to use, as it's just a case of answering the prompts. It will not accept wrong answers (within reason, though it will accept anything between 01-12 for the month or 01-31 for the day), when entering the date.

If you choose the first prompt, (to screen) then you may view the directory as it will be printed before choosing your print option. As well as directory or plain disk covers, you can now print a directory to the right or left column, (saving paper) or in double width, if it is only listings which are needed, (with or without the start addresses or just some).

It is very handy (and can save hair pulling) to have this as well as the usual information, plus track, sector, date, disk name and number, and also blocks free on the cover.

Further information can be added, when saving a program to disk. You press shift/space after the filename. Then type one or more letters to indicate the information you may need, before typing the closing quotes. For example:

SAVE"MC.PROG(SH/SP)1S".

When the directory is listed the '1S' is outside the quotes, you do not include this with the filename when loading, it just tells you to load with ,8,1 and SYS, the SYS number being normally the start address which could be on the cover.

Print the covers on different coloured paper and you'll have a rainbow library (as I have), with a different colour for different types of files. Also you can see at a glance which disks are games or utilities, etc.

If you do have printing problems, refer to your printer manual. The lines that may need altering are: (130 REMOVE REM), (1530 and 2030 TIGHT PRINT. You could remove the REM from line 1650), (1540, 1550 and 2040 2050 SET LEFT MARGIN). For someone with a non-graphics printer, the graphics could be changed in lines 670-680, 2050-2080, 2350-2380. The other graphic lines are obvious. Use '!', '*' and the dash '-'.

The 87 in line 410 will need to be changed each year if you don't want a rude answer.

The 75 in lines 2330, 2470, 2480, can be altered if you have more than 75 files on your disk, and it will depend on paper length, (you can get it cut to the length you wnat). More files will then be printed to the back flap and can be turned in. Paring the sides of the flap allows easy access.

PROGRAM: DIRECTORY COVER

```
37  10 rem *********************
9b  20 rem *                   *
52  30 rem * directory printer *
67  40 rem *                   *
6e  50 rem *    by elizamac    *
73  60 rem *                   *
b9  70 rem *   september /86   *
df  80 rem *                   *
87  90 rem *********************
5e  100 :
8e  110 print chr$(147):poke 532
    80,11:poke 53281,15
9b  120 gn$=""
73  130 rem gn$=chr$(8):rem for
    some printers
b6  140 :
c1  150 dim ft$(5):fori=0to5:rea
    dx$:ft$(i)=x$:nexti
6f  160 dim f$(144,5)
e0  170 di=0:fb=664:sd$=chr$(15)

82  180 data del,seq,prg,usr,rel
    ,del
84  190 :
4f  200 gosub 3000:rem introduct
    ion
52  210 gosub 2700:rem scr/print
     speed
47  220 print chr$(147)chr$(149)
    "[down5]";spc(7);:input"[rvs
    on]disk number[rvsoff]";di
90  230 print chr$(17)spc(7);:in
    put"[rvson]list to screen [y
    /n][rvsoff]";sp$
0a  240 if sp$<>"n"and sp$<>"y"t
    hen print"[up2]";:goto 230
a9  250 if sp$="y"then dv=3:goto
     720
3f  260 :
c7  270 dv=4:gosub 300:gosub 700
    :if x$="n"then 220
89  280 goto 720
19  290 :
9e  300 dc$="":lr$="":ds$="":if
    sd$=chr$(14)then sd$=chr$(15
    )
30  310 print chr$(145)spc(7);:i
    nput"[rvson]print disk cover
    [spc3][y/n][rvsoff]";d$
33  320 if d$<>"y"and d$<>"n"the
    n print chr$(145):goto 310
46  330 if d$="n"then 500
f5  340 print chr$(17)spc(7);:in
    put"[rvson]directory on cove
    r [y/n][rvsoff]";dc$
91  350 if dc$<>"y"and dc$<>"n"t
    hen print"[up2]";:goto 340
d7  360 print chr$(17)spc(7):inp
    ut"[rvson]jacket name[rvsoff
    ]";jn$
40  370 print chr$(17)spc(7);"[r
    vson]enter today's date :[rv
    soff]"
83  380 print chr$(17)spc(7);:in
    put"[rvson]dy/mo/yr[rvsoff]"
    ;dt$
1a  390 if val(mid$(dt$,1,2))<1
    or val(mid$(dt$,1,2))>31 the
    n 420
a7  400 if val(mid$(dt$,4,2))<1
    or val(mid$(dt$,4,2))>12 the
    n 420
ee  410 if val(mid$(dt$,7,2))=87
     then 440
82  420 print chr$(17)spc(10);"[
    rvson]dummy[rvsoff]":fori=1t
    o500:nexti
e9  430 print chr$(145);"[spc15]
    ":print"[up5]":goto 380
8e  435 :
ad  440 y$=left$(in$,(len(jn$))/
    2):z$=right$(jn$,(len(jn$))/
    2)
45  450 dn$=sd$+"[s-,spc6,s-,sz]
    ":dj$="[sz,s-,spc6,s-]"+gn$
08  460 lf$=dn$+chr$(14)+"[sz5]"
    :rf$="[sz5]"+chr$(15)+dj$
8b  470 jb$=sd$+"[spc7,s-,spc10]
    ":bj$="[spc10,s-]"+gn$
6f  480 return
d1  490 :
32  500 print chr$(145)spc(7);:i
    nput"[rvson]list double widt
    h  [y/n][rvsoff,spc3,left3]"
    ;ds$
73  510 ifds$="y"then sd$=chr$(1
    4):printspc(8)"[down4,rvson]
    to printer double width[rvso
    ff]":return
6e  520 if ds$<>"n"then print ch
    r$(145):goto 500
2e  530 :
a7  540 print chr$(17)spc(7);:in
    put"[rvson]list left column[
    spc3][y/n][rvsoff]";lr$
92  550 if lr$<>"n"and lr$<>"y"t
    hen print"[up3]":goto 540
dc  560 if lr$="y"then print spc
    (8);"[down4,rvson]to printer
      left column[rvsoff]":retur
    n
1c  570 print spc(8);"[down4,rvs
    on]to printer right column[r
    vsoff]":return
7c  580 :
1d  590 print chr$(28)spc(12);"[
    down4,rvson]is printer set ?
    [rvsoff]"chr$(149)
16  600 print spc(8);"[down3,rvs
    on]press any key to continue
    [rvsoff]"
b3  610 wait 198,1:get a$:if a$=
    ""then 610
f0  620 if a$="q"then 1920
f6  630 return
c4  640 wait 198,1:get x$:if x$<
    >"y"and x$<>"n"and x$<>"q"th
    en 640
c9  650 if x$="q"then 1920
18  660 return
db  670 print#4,sd$;"[s-]";:forl
    =1to70:print#4,jl$;:nextl:pr
    int#4,"[s-]";gn$:return
9f  675 :
fe  680 print#4,sd$;spc(7);:forl
    =1to58:print#4,jl$;:nextl:pr
    int#4,gn$;"fold"
3a  690 return
83  695 :
24  700 print spc(8);"[down2,rvs
    on]is this correct [y/n] ?[r
    vsoff]"
7f  710 gosub 640:return
f7  715 :
c1  720 print chr$(147)spc(11);"
    [down6,rvson]insert correct
    disk[rvsoff]"
58  730 gosub 600:rem keypress
9a  740 print chr$(147)spc(9);"[
    down6,rvson]please be patien
    t....[rvsoff]"
e8  750 print chr$(17)spc(11);"[
    rvson]reading directory[rvso
    ff]"
c0  760 :
44  770 open15,8,15:print#15,"i0
    ":gosub 1960:rem error chann
    el
09  780 open8,8,8,"$0,s,r":rem r
    ead directory
b8  790 gosub 1960
'19 800 :
5a  810 fori=1to142:get#8,x$:nex
    ti
f3  820 fori=143to160:get#8,x$:h
    $=h$+x$:nexti
03  830 fori=161to162:get#8,x$:i
    d$=id$+x$:nexti
f3  840 get#8,x$:fori=164to165:g
    et#8,x$:tf$=tf$+x$:nexti
72  850 fori=166to254:get#8,x$:n
    exti
0a  860 ab=8
36  870 bc=bc+1
99  880 if ab=8then ab=1:goto 91
    0
74  890 ab=ab+1:get#8,x$,x$:fi=s
    t
ae  900 if fi<>0 then 1090
12  910 get#8,x$:if x$=""then x$
    =chr$(133)
f4  920 fi=st:if fi<>0 then 1090

73  930 ty$=ft$((asc(x$)and191)-
    128)
69  940 get#8,x$:if x$=""then x$
    =chr$(0)
df  950 tr$=right$("  "+str$(asc
    (x$)),2)
bd  960 get#8,x$:if x$=""then x$
    =chr$(0)
83  970 se$=right$("  "+str$(asc
    (x$)),2)
27  980 fi$="":fori=3to18:get#8,
    x$:fi$=fi$+x$:nexti
b7  990 fori=19to27:get#8,x$:nex
    ti
45  1000 get#8,lb$,hb$
5e  1010 bl=asc(lb$+chr$(0))+256
    *asc(hb$+chr$(0))
f4  1020 if ty$<>"del"then fb=fb
    -bl
4a  1030 bl$=right$("[spc5]"+str
    $(bl),3)
94  1040 if tr$=" 0"then 1090
82  1050 f$(bc,0)=fi$:f$(bc,1)=t
    r$:f$(bc,2)=se$:f$(bc,3)=bl$
    :f$(bc,4)=ty$
99  1060 f$(bc,5)="[spc5]"
cc  1070 if ty$="prg"then f$(bc,
    5)="-----"
14  1080 goto 870
1e  1090 close8
6a  1100 :
f9  1110 gosub 1960
6d  1120 if f$(bc,0)=""then bc=b
    c-1:goto 1120
6a  1130 fb$=right$("[spc4]"+str
    $(fb),3)
50  1140 di$=right$("[spc4]"+str
    $(di),3)
b8  1150 :
cf  1160 if dc$="n"then 1450
e9  1170 print chr$(147)spc(8);"
    [down6,rvson]the start addre
    ss [y/n] ?[rvsoff]"
01  1180 gosub 640:if dv=3 and x
    $="n"then print chr$(147):go
    to 1500
85  1190 if dv=4 and x$="n"then
    1450
8e  1200 :
ab  1210 print chr$(17)spc(15);"
    [1] all"
35  1220 print chr$(17)spc(15);"
    [2] some"
```

```
97  1230 wait 198,1:get a$:if a$
    <>"1"and a$<>"2"then 1230
99  1240 if a$="2"then print chr
    $(147)chr$(17):goto 1270
eb  1250 if a$="1"then print chr
    $(147)spc(12);"[down6,rvson]
    just a moment...[rvsoff]"
ca  1260 :
06  1270 fori=1tobc
64  1280 if f$(i,4)<>"prg"then 1
    410
4f  1290 if a$="1"then 1340
23  1300 :
cc  1310 print spc(3);f$(i,0);"
    [y/n]"
6e  1320 gosub 640:if x$="n"then
     print"[up2]":goto 1410
0d  1330 :
71  1340 sa$=f$(i,0)
4c  1350 open8,8,8,"0:"+sa$+",p,
    r"
64  1360 gosub 1960
31  1370 get#8,lb$,hb$
96  1380 sa=asc(lb$+chr$(0))+256
    *asc(hb$+chr$(0))
0b  1390 close8
c2  1400 f$(i,5)=right$("[spc6]"
    +str$(sa),5)
4e  1410 nexti
ab  1420 :
d1  1430 if dv=3 then gosub 600:
    print chr$(147):goto 1500
0f  1440 print chr$(147)chr$(28)
    spc(3);"[down5,rvson]wait un
    til the drive light goes off
    [rvsoff]"
80  1450 gosub 590
e0  1460 if gn$=""then open6,4,6
    :print#6,chr$(21):close6
f9  1470 :
f6  1480 if d$="y"then 2020
ed  1490 :
2a  1500 if dv=3then print spc(7
    );"[down,rvson]press 's' slo
    w ' ' normal[rvsoff]":wait 1
    98,1:get s$
91  1510 if dv=3 and s$="s"then
    poke 251,10:rem 10=speed (25
    5 slowest)
ca  1515 :
a0  1520 open4,dv
b8  1530 remif dv=4 then print#4
    ,chr$(15)chr$(27)chr$(48)
7a  1540 if(lr$="y"or ds$="y")th
    en print#4,chr$(27)chr$(77)c
    hr$(1)
25  1550 if lr$="n"then print#4,
    chr$(27)chr$(77)chr$(42)
88  1560 print#4,sd$;"[ca,s*18,c
    r,s*2,cr,s*2,cr,s*11,cs]";gn
    $
e7  1570 print#4,sd$;"[s-]";h$;"
    [s-]";id$;"[s-]";tf$;"[s-]f/
    blks: ";fb$;"[s-]";gn$
22  1580 print#4,sd$ "[cq,s*16,c
    r,s*,ce,cr,s*,ce,cr,s*,ce,s*
    ,cr,s*3,cr,s*5,cw]";gn$
8b  1590 print#4,sd$;"[s-]files:
      disk/";di$;"[s-]tr[s-]se[s
    -]blk[s-]typ[s-]start[s-]";g
    n$
00  1600 print#4,sd$ "[cq,s*16,s
    +,s*2,s+,s*2,s+,s*3,s+,s*3,s
    +,s*5,cw]";gn$
00  1610 for i=1 to bc
c4  1620 fi$=f$(i,0):tr$=f$(i,1)
    :se$=f$(i,2):bl$=f$(i,3):ty$
    =f$(i,4)
17  1630 if ty$="del"then 1660
b0  1640 print#4,sd$;"[s-]";fi$;
    "[s-]";tr$;"[s-]";se$;"[s-]"
    ;bl$;"[s-]";ty$;"[s-]";f$(i,
    5);"[s-]";gn$
ec  1650 rem print#4,sd$;"[s-,sp
    c16,s-]  [s-]  [s-,spc3,s-,s
    pc3,s-,spc5,s-]";gn$
49  1660 nexti
c7  1670 print#4,sd$;"[cz,s*16,c
    e,s*2,ce,s*2,ce,s*3,ce,s*3,c
    e,s*5,cx]";gn$
c2  1680 print#4:close4:close15
72  1690 if s$="s"then s$="":pok
    e 251,0:rem 0=normal speed
9f  1695 :
62  1700 if dv=4 then print chr$
    (147)spc(10);"[down6,rvson]p
    rint again [y/n] ?[rvsoff]"
68  1710 if dv=3 then print chr$
    (17)spc(11);"[rvson]view aga
    in [y/n] ?[rvsoff]"
65  1720 gosub 640:if dv=3 and x
    $="n"then 1780
a5  1730 if dv=3 and x$="y"then
    print chr$(147):goto 1500
b8  1740 if dv=4 and x$="n"then
    1840
9c  1750 gosub 590:if d$="y"then
     2020
f3  1760 goto 1500
ca  1770 :
87  1780 print chr$(145)spc(9);"
    [rvson]send to printer [y/n]
     ?[rvsoff]";gosub 640
aa  1790 if x$="n"then 1900
47  1800 dv=4
48  1810 print chr$(147)"[down5]
    ";:gosub 300:gosub 700:if x$
    ="n"then 1810
ba  1820 print chr$(147)"[down5]
    ":goto 1450
17  1830 :
5e  1840 print chr$(145)spc(6);"
    [rvson]change print option [
    y/n] ?[rvsoff]":gosub 640
d5  1850 if x$="n"then 1900
4b  1860 print chr$(147)"[down5]
    ";:gosub 300:gosub 700:if x$
    ="n"then 1860
fd  1870 print chr$(147):gosub 5
    90:if d$="n"then 1500
f6  1880 goto 2020
53  1890 :
b0  1900 print,chr$(145)spc(6);"
    [spc3,rvson]new directory [y
    /n] ?[rvsoff,spc5]"
59  1910 gosub 640:if x$="y"then
     run
b3  1920 print chr$(28)spc(12);"
    [down2,rvson]are you sure ?[
    rvsoff]":gosub 640:if x$="n"
    then run
e7  1930 print spc(8);"[down,rvs
    on]remember to reset printer
    [rvsoff]":for i=1 to 500:nex
    t i
9b  1940 sys 64738
9f  1950 :
03  1960 input#15,ea,eb$,ec,ed:i
    f ea=0 then return
2e  1970 print chr$(18)ea;eb$;ec
    ;ed
a5  1980 close8:close15
4d  1990 end
ed  2000 :
04  2010 rem disk cover
9e  2020 open4,dv
e5  2030 remprint#4,chr$(15)chr$
    (27)chr$(48)
c8  2040 remprint#4,chr$(27)chr$
    (77)chr$(1)
e8  2050 jl$="[s*]":print#4,sd$;
    spc(4);chr$(206);:fori=1to62
    :print#4,jl$;:nexti
f4  2060 print#4,chr$(205);gn$;"
    cut"
76  2070 fori=1to2:print#4,chr$(
    10):nexti:rem line feed
c7  2080 print#4,spc(1);chr$(206
    );spc(60);"fold";spc(4);chr$
    (205)
08  2090 :
20  2100 gosub 670:rem fold
36  2110 print#4,dn$;spc(54);dj$

9e  2120 print#4,lf$;:forq=1to(1
    7-len(jn$))/2:print#4," ";:n
    extq:print#4,jn$;
da. 2130 if(len(y$)+len(z$))<len
    (jn$)then print#4,spc(q-1);r
    f$:goto 2160
c3  2140 print#4,spc(q);rf$
54  2150 :
a7  2160 print#4,dn$;spc(1);"dis
    k/";di$;spc(36)dt$;spc(1);dj
    $
4d  2170 print#4,dn$;spc(9);h$;"
    [s-]";id$;"[s-]";tf$;"[s-]f/
    blks: ";fb$;spc(9);dj$
b6  2180 :
85  2190 if dc$="n"then goto 252
    0
9a  2200 :
91  2210 print#4,dn$;spc(9);"[s*
    16,cr,s*,ce,cr,s*,ce,cr,s*,c
    e,s*,cr,s*3,cr,s*5]";spc(9);
    dj$
61  2220 print#4,dn$;spc(9);"fil
    enames:[spc6,s-]tr[s-]se[s-]
    blk[s-]typ[s-]start";spc(9);
    dj$
84  2230 :
b9  2240 print#4,dn$;spc(9);"[sz
    16,s+,sz2,s+,sz2,s+,sz3,s+,s
    z3,s+,sz5]";spc(9);dj$
96  2250 cc=0
86  2260 fori=1to bc
a0  2270 fi$=f$(i,0):tr$=f$(i,1)
    :se$=f$(i,2):bl$=f$(i,3):ty$
    =f$(i,4)
c1  2280 if ty$="del"then goto 2
    400
c0  2290 :
98  2300 if cc=26 then gosub 670
    :goto 2380
98  2310 if cc=66 then gosub 680
    :goto 2380
af  2320 if cc>26 then 2380
a1  2330 if cc=75 then 2590
17  2340 :
94  2350 print#4,dn$;spc(9);fi$;
    "[s-]";tr$;"[s-]";se$;"[s-]"
    ;bl$;"[s-]";ty$;"[s-]";f$(i,
    5);
4e  2360 print#4,spc(9);dj$
71  2370 :
aa  2380 ifcc=>26thenprint#4,jb$
    ;fi$;"[s-]";tr$;"[s-]";se$;"
    [s-]";bl$;"[s-]";ty$;"[s-]";
    f$(i,5);bj$
2c  2390 cc=cc+1
34  2400 nexti
49  2410 :
5a  2420 if bc<26 then fori=1to2
    6-cc:print#4,dn$;spc(54);dj$
    :nexti:goto 2540
2f  2430 if bc=26 then 2540
ab  2440 :
c8  2450 if bc<66then fori=1to66
    -cc:print#4,jb$;spc(36);bj$:
    nexti:goto 2560
c8  2460 if bc=66 then goto 2560

90  2470 if bc>66and bc<75then f
    ori=1to75-cc:print#4,jb$;spc
    (36);bj$:nexti:goto 2590
84  2480 if bc=75 then 2590
f9  2490 :
4a  2500 rem jacket unlisted
ed  2510 :
```

```
14  2520 print#4,dn$;spc(9);"[sz
    18,ce,sz2,ce,sz2,ce,sz11]";s
    pc(9);dj$
37  2530 fori=1to28:print#4,dn$;
    spc(54);dj$:nexti
bd  2540 gosub 670:rem fold
31  2550 fori=1to40:print#4,jb$;
    spc(36);bj$:nexti
e9  2560 gosub 680:rem fold
1d  2570 fori=1to9:print#4,jb$;s
    pc(36);bj$:nexti
24  2580 :
4a  2590 print#4:close4
5f  2600 close15:goto 1700
06  2610 :
aa  2680 rem scr/print speed
b6  2690 :
c2  2700 ck=0
64  2710 for j=679to703:reada:ck
    =ck+a:poke j,a:next
20  2720 if ck<>3615then print"c
    heck data":end
b6  2730 data 72,138,72,152,72,8
    ,166,251,240,8,160,255,136,2
    08,253,202,208,248,40
32  2740 data 104,168,104,170,10
    4,76
7c  2750 if peek(807)<>2then pok
    e 704,peek(806):poke 705,pee
    k(807)
bb  2760 poke 806,167:poke 807,2

52  2770 return
dc  2780 :
6e  3000 rem introduction
ce  3010 print chr$(142)chr$(31)
    "[clr]"::fori=0to40:print"[c
    b]";:nexti:print spc(38);"[c
    b2]";
f6  3020 printspc(10);"for some
    printers"spc(11)"[cb2]";:pri
    nt spc(38);"[cb2]";
9c  3030 printchr$(28)spc(10);"[
    rvson]remember line 130[rvso
    ff]";spc(11)chr$(31);"[cb2]"
    ;
46  3040 print spc(38);"[cb2]";
f5  3050 print"[spc6]**** the op
    tions are ****[spc7,cb2]";:p
    rint spc(38);"[cb2]";
de  3060 print"[spc10]print disk
     covers[spc11,cb2]";:print s
    pc(38);"[cb2]";
20  3070 print"[spc6]with or wit
    hout directory[spc7,cb2]";:p
    rint spc(38);"[cb2]";
61  3080 print"  (with no start
    address,all - some)  [cb2]";
    :print spc(38);"[cb2]";
74  3090 print"  track & sector,
    disk number & date.  [cb2]";
    :print spc(38);"[cb2]";
a5  3100 print"[spc5]list direct
    ory double width[spc6,cb2]";
    :print spc(38);"[cb2]";
61  3110 print"[spc6]or to right
     - left column[spc7,cb2]";:p
    rint spc(38);"[cb2]";
91  3120 print"[spc4]you can vie
    w before printing.[spc5,cb2]
    ";:print spc(38);"[cb2]";
eb  3130 print"[spc6,rvson]press
     any key to continue[rvsoff,
    spc7,cb2]";:print spc(38);"[
    cb2]";
51  3140 fori=1to38:print"[cb]";
    :nexti:print"[left]"chr$(148
    )"[cb]";
c2  3150 gosub 610
ea  3160 return
```

1541 Fast Loader

The 1541 disk drive has been described as the 'lumbering Hippo' of disk drives. Speed it up with this fast loader.

By Paul Eves

The Commodore 1541 disk drive is notorious for being one of the slowest disk drives available for any computer. It may seem silly but some of the cassette fast loading systems for the C64 are actually faster than this disk drive.

The fast loader program presented here patches itself into the C64's memory and improves on the speed of loading.

All that you need to do is LOAD and RUN the program "FAST LOADER" and the changes to disk loading speed will become very obvious.

It is worth pointing out at this stage that the fast loader does occupy some of the C64's memory. It is therefore possible for some programmes to corrupt the fast loader preventing it from working.

Getting it all in

The program is presented here as a Basic loader and should be typed in as a normal Basic program. When you have finished typing it in save it do disk with the name "FAST LOAD BAS".

Type the following line and press RETURN

POKE 43,0,POKE 44,16:NEW

Now LOAD the program "FAST LOAD BAS" from your disk and RUN it.

When you want to use the fast loader simply load it into your C64 and RUN it. You will told when it is patched into the computers operating system.

Getting it in

FAST LOADER

1) Type in the BASIC program presented here.
2) SAVE the program onto disc.
3) Type NEW.
4) Enter the following:

```
   POKE 43,0:POKE 44,16:NEW
```

5) LOAD and RUN the program saved in 2.
6) When finished enter the following to SAVE the program :

```
   POKE43,1:POKE44,8:POKE45,192:
POKE46,12:SAVE"FASTLOAD",8
```

7) The program will now be on disk.

PROGRAM: FAST LOAD.BAS

```
ready.

84   1 rem ***********************

bd   2 rem * program to set up   *

a7   3 rem * fast loader         *

dd   4 rem * in memory           *

9c   5 rem * remember to enter   *

4f   6 rem * poke's before       *

a8   7 rem * loading and running *

a7   8 rem * this program        *

bc   9 rem ***********************

6e   10 bl=75    :ln=50    :sa=2049

5b   20 for l=0 to bl:cx=0:for d=0
      to 15:read a:cx=cx+a:poke sa
     +l*16+d,a:next d
a5   30 read a:if a><cx thenprint"
     error in line";ln+(l*10):stop
40   40 next l:end
2b   50 data 11,8,51.8.158,50,48,5
     7,57.0,0,0,0,147,89,67,751
ba   60 data 32,70,65,83,84,45,76,
     79,65,68,32,83,89,83,84,69,11
     07
c2   70 data 77,13,65,67,84,73,86,
     65,84,69,68,46,13,0,32,32,874

6a   80 data 32,32,169,6,141,33,20
     8,162,0,189,14,8,240,6,32,210
     ,1482
ea   90 data 255,232,208,245,120,1
     60,0,132,251,169,224,133,252,
     177,251,145,2954
d9   100 data 251,200,208,249,230,
     252,208,245,169,248,133,252,1
     69,191,162,8,3175
37   110 data 133,253,134,254,177,
     253,145,251,200,208,249,230,2
     54,230,252,165,3388
04   120 data 252,201,252,144,239,
     169,229,141,214,253,162,34,18
     9,156,8,157,2800
e7   130 data 192,2,202,16,247,32,
     191,8,141,76,253,142,77,253,1
     69,219,2220
ca   140 data 162,2,141,35,229,142
     ,40,229,88,96,0,72,169,53,133
     ,1,1592
9e   150 data 104,32,111,248,72,16
     9,72,141,143,2,169,235,141,14
     4,2,169,1954
0f   160 data 55,133,1,104,96,0,16
     9,53,133,1,76,72,235,0,169,19
     2,1489
47   170 data 162,2,141,48,3,142,4
     9,3,96,120,169,39,141,0,221,4
     4,1380
73   180 data 0,221,80,251,169,3,1
     41,0,221,162,9,202,208,253,16
     2,4,2086
04   190 data 173,0,221,10,8,10,38
     ,251,40,38,251,202,208,242,18
     1,251,2124
57   200 data 145,174,200,208,233,
     169,23,141,0,221,165,251,96,1
     20,169,39,2354
eb   210 data 141,0,221,44,0,221,8
     0,251,169,3,141,0,221,162,8,2
     02,1864
36   220 data 208,253,162,4,173,0,
     221,10,8,10,38,251,40,38,251,
     202,1869
67   230 data 208,242,169,23,141,0
     ,221,234,234,234,165,251,96,1
     33,147,169,2667
64   240 data 0,133,144,165,186,20
     1,8,240,3,76,171,244,164,183,
     208,3,2129
8e   250 data 76,16,247,140,230,25
     1,160,0,177,187,153,231,251,1
     92,0,208,2519
d6   260 data 4,201,36,240,228,200
     ,196,183,144,238,32,175,245,1
     73,24,3,2322
dc   270 data 72,173,25,3,72,169,1
     93,162,254,141,24,3,142,25,3,
     169,1630
e8   280 data 130,141,13,221,169,1
     ,141,6,221,169,0,141,7,221,16
     9,25,1775
29   290 data 141,15,221,169,8,141
     ,15,221,104,141,25,3,104,141,
     24,3,1476
83   300 data 173,21,208,133,254,1
     69,0,141,21,208,169,19,162,25
     0,133,3,2064
fe   310 data 134,4,162,0,169,3,13
     4,5,133,6,169,8,32,12,237,169
     ,1377
28   320 data 111,32,185,237,165,1
     44,16,7,169,128,133,253,76,22
     0,249,169,2294
09   330 data 77,32,221,237,169,45
     ,32,221,237,169,87,32,221,237
     .165 5,2187
6b   340 data 32,221,237,165,6,32,
     221,237,169,29,32,221,237,160
     ,0,177,2176
77   350 data 3,32,221,237,200,192
     ,29,144,246,32,254,237,24,165
     ,3,105,2124
bd   360 data 29,133,3,144,3,230,4
     ,24,165,5,166,6,105,29,133,5,
     1184
02   370 data 144,3,232,230,6,224,
     4,144,161,201,228,144,157,173
     ,17,208,2276
28   380 data 41,239,141,17,208,16
     9,8,32,12,237,169,111,32,185,
     237,169,2007
04   390 data 77,32,221,237,169,45
     ,32,221,237,169,69,32,221,237
     ,169,3,2171
17   400 data 32,221,237,169,3,32,
     221,237,32,254,237,169,7,141,
     0,221,2213
36   410 data 162,0,202,208,253,13
     4,253,32,63,248,201,255,240,9
     0,160,2,2503
38   420 data 166,253,208,23,72,32
     ,63,248,168,32,63,248,166,185
     ,208,4,2139
5e   430 data 164,195,165,196,132,
     174,133,175,160,4,104,201,0,2
     40,20,132,2195
84   440 data 253,56,165,174,229,2
     53,133,174,176,2,198,175,32,1
     1,248,230,2509
77   450 data 175,208,196,32,63,24
     8,133,253,160,0,165,253,201,2
     ,144,10,2243
ab   460 data 32,63,248,145,174,20
     0,198,253,208,240,169,255,133
     ,253,152,24,2747
85   470 data 101,174,133,174,144,
     2,230,175,173,17,208,9,16,141
     ,17,208,1922
32   480 data 165,254,141,21,208,1
     73,13,221,169,127,141,13,221,
     88,165,253,2373
b7   490 data 208,3,76,4,247,201,1
     28,208,3,76,7,247,76,169,245,
     0,1898
8f   500 data 0,76,8,4,169,8,141,0
     ,24,76,126,3,162,1,88,138,102
     4
8f   510 data 44,0,24,240,251,120,
     169,0,141,0,24,138,44,0,24,20
     8,1427
f9   520 data 251,234,162,4,177,10
     ,73,255,133,20,169,0,6,20,42,
     10,1566
b4   530 data 6,20,42,10,141,0,24,
     202,208,240,234,234,234,200,2
     08,226,2229
02   540 data 234,234,234,169,8,14
     1,0,24,96,73,255,88,133,20,16
     2,1,1872
2a   550 data 138,44,0,24,240,251,
     120,169,0,141,0,24,138,44,0,2
     4,1357
f7   560 data 208,251,162,4,169,0,
     6,20,42,10,6,20,42,10,141,0,1
     091
74   570 data 24,202,208,240,162,3
     ,202,208,253,169,8,141,0,24,9
     6,32,1972
09   580 data 24,193,169,0,162,6,1
     33,10,134,11,133,14,169,6,133
     ,249,1546
fe   590 data 169,2,133,106,169,18
     ,133,6,169,1,133,7,32,119,4,1
     60,1361
24   600 data 35,201,1,208,80,160,
     0,185,2,6,41,135,201,130,208,
     53,1646
f6   610 data 162,0,240,26,189,212
     ,4,217,5,6,240,11,201,63,208,
     37,1821
cc   620 data 185,5,6,201,160,240.
     30,232,200,236,211,4,176,9,18
     9,212,2296
04   630 data 4,201,42,240,59,208,
     221,152,41,31,201,16,176,50,1
     85,5,1832
a5   640 data 6,201,160,240,43,152
     ,41,224,24,105,32,168,144,185
     ,173,0,1898
16   650 data 6,208,16,160,98,169,
     255,32,72,3,169,0,141,0,24,15
     2,1505
5b   660 data 76,200,193,173,1,6,7
     6,153,3,169,6,133,49,76,209,2
     44,1767
dc   670 data 152,41,224,168,185,3
     ,6,133,6,185,4,6,133,7,32,119
     ,1404
51   680 data 4,160,35,201,1,208,2
     06,173,0,6,133,6,32,72,3,165,
     1405
65   690 data 14,208,18,230,14,173
     ,2,6,32,72,3,173,3,6,32,72,10
     58
67   700 data 3,160,4,208,2,160,2,
     165,6,240,11,32,11,3,173,1,11
     81
1a   710 data 6,133,7,76,29,4,173,
     1,6,32,72,3,136,204,1,6,889
49   720 data 176,10,200,185,0,6,3
     2,72,3,76,92,4,169,0,141,0,11
     66
43   730 data 24,169,1,133,28,76,1
     48,193,162,0,134,15,134,12,16
     6,28,1423
fc   740 data 240,9,169,0,133,28,1
     69,176,32,189,4,169,224,32,18
     9,4,1767
b0   750 data 201,2,208,41,165,12,
     208,37,230,12,169,192,32,189,
     4,169,1871
bd   760 data 176,32,189,4,201,1,2
     08,21,76,138,4,201,3,208,14,1
     65,1641
15   770 data 15,208,10,230,15,169
     ,192,32,189,4,76,138,4,96,141
     ,91,1610
10   780 data 2,141,77,2,133,0,169
     ,255,141,152,2,162,0,88,32,16
     6,1522
73   790 data 213,176,251,96,165,2
     53,208,3,76,4,247,201,128,208
     ,3,76,2308
e4   800 data 7,247,76,169,245,73,
     1,169,245,0,0,0,0,255,0,0,148
     7
```

Menu Maker

Make the loading and running of files much easier with this handy menu program.

By Tony Crowther

When loading a program from disk it can sometimes be quite difficult to remember exactly how a program should be loaded. Three months after writing your all-sing, all-dancing utility, the chances of you remembering whether it was loaded and RUN as a Basic program, or loaded as machine code program or started with SYS 49152, or was it 32768?

The menu program presented here will make life much easier. This program will produce a menu on your disk which when loaded and RUN as a Basic file will offer you a menu of the programs on the disk. Pressing the letter next to the program that you require will cause the program to be loaded into the computers memory and then executed as required.

Using the program

When RUN the MENU program will read the filenames off the disk that is in the drive when requested. The user can then select which programmes he/she wants to appear in the menu. If you don't want a certain file in the menu just press 'N' when prompted. If you require a file to be present in the menu then pressing 'Y' will give you further options, asking for the type of file, etc.

The file type can either be Basic, press 'B' when prompted, or machine code 'M'. If you select Basic then the menu generator will move onto the next program on the disk. Selecting a file marked with a 'B' will cause the program to be loaded and RUN just as you would with a normal program.

Should you press 'M' when prompted for the file type you will then be asked for the start address of the machine code program. You can give the start address either in decimal (e.g. 49152), or hexadecimal by prefixing the number with a dollar ($) sign (e.g. $C000).

When you have been through all of the program on the disk the menu generator will save a program called "MENU"onto the disk. Loading and running this file will produce a menu on screen which you can load the required file from simply by pressing the relevant letter.

If you have a directory designer it is quite useful to move the program MENU so that it is the first in the directory. This means that you can load it into your computer with a simple LOAD "*",8 command.

Other options

As well as allowing you to create a MENU program the menu generator also allows you to specify a colour for the word MENU when it appears on the screen. The option to add a line of descriptive text to the menu also exists. Should you ever require to check that the disk in the drive is the one that you want to add a menu to, the main menu of the generator program offers the facility of printing a directory listing to the screen.

Other options

As well as allowing you to create a MENU program the menu generator also allows you to specify a colour for the word MENU when it appears on the screen. The option to add a line of descriptive text to the menu also exists. should you ever require to check that the disk in the drive is the one that you want to add a menu to, the main menu of the generator program offers the facility of printing a directory listing to the screen.

```
Getting it in

MENU MAKER

1) Type in the BASIC program
presented here.
2) SAVE the program onto disc.
3) Type NEW.
4) Enter the following:

   POKE 43,0:POKE 44,18:NEW

5) LOAD and RUN the program saved
in 2.
6) When finished enter the
following to SAVE the program :

   POKE43,1:POKE44,8:POKE45,164:

   POKE46,17:SAVE"MENUMAKER",8

7) The program will now be on
disk.
```

```
PROGRAM: MENU MAKER.BAS

ae   1 rem ***********************
     *
e1   2 rem * program to set up
     *
45   3 rem * menu maker in memory
     *
ab   4 rem *
     *
c0   5 rem * remember to enter
     *
13   6 rem * poke commands before
     *
cc   7 rem * loading and running
     *
d7   8 rem * this program
     *
e6   9 rem ***********************
     *
64   10 bl=154  :ln=50   :sa=2049

5b   20 for l=0 to bl:cx=0:for d=0
      to 15:read a:cx=cx+a:poke sa
     +l*16+d,a:next d
a5   30 read a:if a><cx thenprint"
     error in line":ln+(l*10):stop

40   40 next l:end
03   50 data 22,8,195,7,158,50,48,
     56,48,58,143,34,20,20,20,20,9
     07
91   60 data 82,65,84,84,0,0,0,0,0
     ,0,0,0,0,0,0,162,477
52   70 data 0,189,115,8,157,0,1,2
     32,224,147,208,245,162,0,189,
     10,1887
e7   80 data 9,157,0,2,232,224,84,
     208,245,169,62,133,252,169,3,
     133,2082
a0   90 data 253,173,6,9,133,250,1
     73,7,9,133,251,173,8,9,133,24
     8,1968
15   100 data 173,9,9,133,249,165,
     248,56,229,250,133,254,165,24
     9,229,251,2802
cf   110 data 133,255,165,254,24,1
     05,61,133,254,165,255,105,3,1
     33,255,76,2376
1d   120 data 0,1,120,165,1,133,24
     7,169,0,133,1,160,0,177,250,1
     45,1702
7c   130 data 252,230,250,208,2,23
     0,251,230,252,208,2,230,253,1
     65,251,197,3211
ae   140 data 249,144,234,208,6,16
     5,250,197,248,144,226,162,0,1
     89,0,2,2424
96   150 data 24,125,2,2,133,250,1
     89,1,2,125,3,2,133,251,165,25
     0,1657
44   160 data 56,233,1,133,250,165
     ,251,233,0,133,251,189,2,2,13
     3,252,2284
08   170 data 189,3,2,133,253,160,
     0,177,254,145,250,165,254,56,
     233,1,2275
18   180 data 133,254,165,255,233,
     0,133,255,165,250,56,233,1,13
     3,250,165,2681
27   190 data 251,233,0,133,251,16
     5,252,56,233,1,133,252,165,25
     3,233,0,2611
73   200 data 133,253,5,252,208,20
     9,138,56,233,4,170,16,160,165
     ,247,133,2382
00   210 data 1,88,76,0,16,16,9,16
     4,17,0,16,148,8,0,0,186,745
0b   220 data 142,98,23,169,147,32
     ,210,255,169,0,141,32,208,141
     ,33,208,2008
```

```
6d   230 data 169,147,133,44,169,2
     4,133,45,169,11,141,134,2,169
     ,0,32,1522
d9   240 data 38,20,32,228,255,201
     ,49,144,249,201,52,176,245,20
     1,49,240,2380
6f   250 data 10,201,50,240,3,76,1
     13,16,76,250,16,169,0,141,91,
     16,1468
ff   260 data 32,97,17,160,0,185,9
     2,16,240,6,32,210,255,200,208
     ,245,1995
58   270 data 32,228,255,201,0,240
     ,249,76,4,16,0,32,32,32,32,32
     ,1461
03   280 data 32,32,32,32,80,82,69
     ,83,83,32,65,32,75,69,89,0,88
     7
c0   290 data 169,1,141,91,16,32,9
     7,17,169,1,32,38,20,160,0,32,
     1016
e8   300 data 207,255,201,13,240,8
     ,153,228,23,200,192,30,144,24
     1,169,2,2306
7f   310 data 32,38,20,32,132,18,1
     73,144,19,141,152,22,169,6,16
     2,244,1504
fc   320 data 160,16,32,189,255,16
     9,8,162,8,160,1,32,186,255,32
     ,192,1857
02   330 data 255,162,8,32,201,255
     ,169,52,133,250,169,21,133,25
     1,169,38,2298
b7   340 data 32,210,255,169,3,32,
     210,255,160,0,177,250,32,210,
     255,230,2480
c3   350 data 250,208,2,230,251,16
     5,251,197,45,144,237,208,6,16
     5,250,197,2806
25   360 data 44,144,229,169,255,3
     2,210,255,32,204,255,169,8,32
     ,195,255,2488
bb   370 data 76,4,16,64,58,77,69,
     78,85,160,0,169,13,32,210,255
     ,1366
0d   380 data 32,210,255,169,32,32
     ,210,255,32,210,255,32,210,25
     5,32,210,2431
db   390 data 255,169,64,32,210,25
     5,32,207,255,201,13,240,8,153
     ,57,17,2168
e0   400 data 200,192,32,144,241,1
     52,162,57,160,17,32,189,255,1
     69,13,32,2047
f7   410 data 210,255,32,232,17,76
     ,68,16,234,234,234,234,234,23
     4,234,234,2778
5c   420 data 234,234,234,234,234,
     234,234,234,234,234,234,234,2
     34,234,234,234,3744
c6   430 data 234,234,234,234,234,
     234,234,234,234,234,234,234,2
     34,234,234,234,3744
8f   440 data 169,147,32,210,255,1
     69,0,133,144,169,1,162,156,16
     0,19,32,1958
38   450 data 189,255,169,8,162,8,
     160,0,32,186,255,32,192,255,1
     62,8,2073
51   460 data 32,198,255,166,144,2
     08,73,32,207,255,32,207,255,3
     2,207,255,2558
52   470 data 32,207,255,32,207,25
     5,166,144,208,54,133,251,32,2
     07,255,133,2571
61   480 data 252,32,167,19,169,32
     ,32,210,255,32,207,255,166,14
     4,208,32,2212
46   490 data 170,240,6,32,210,255
     ,76,170,17,169,13,32,210,255,
     165,197,2217
75   500 data 201,63,240,12,173,91
     ,16,240,3,32,12,18,160,4,208,
```

```
     189,1662
1f   510 data 169,13,32,210,255,32
     ,204,255,169,8,32,195,255,169
     ,15,32,2045
2d   520 data 195,255,169,0,32,189
     ,255,169,15,162,8,160,15,32,1
     86,255,2097
a4   530 data 32,192,255,162,15,32
     ,198,255,164,144,208,9,32,207
     ,255,32,2192
e1   540 data 210,255,76,249,17,32
     ,204,255,32,231,255,169,145,3
     2,210,255,2627
e0   550 data 160,5,177,209,201,34
     ,240,3,76,110,18,169,29,133,2
     11,169,1944
59   560 data 63,32,210,255,32,204
     ,255,32,228,255,201,3,240,16,
     201,89,2316
48   570 data 240,19,201,78,208,24
     1,169,3,32,38,20,76,105,18,17
     4,98,1720
b9   580 data 23,154,76,4,16,169,4
     ,32,38,20,32,228,255,201,77,2
     40,1569
a0   590 data 33,201,66,208,245,16
     9,5,32,38,20,169,0,141,145,19
     ,141,1632
d9   600 data 144,19,32,14,19,76,1
     05,18,162,8,32,198,255,169,13
     ,76,1340
28   610 data 210,255,169,6,32,38,
     20,32,132,18,32,14,19,32,100,
     19,1128
9a   620 data 76,105,18,160,0,32,2
     07,255,201,13,240,8,153,124,1
     9,200,1811
54   630 data 192,5,144,241,173,12
     4,19,201,36,240,53,169,0,141,
     144,19,1901
3b   640 data 141,145,19,170,136,1
     85,124,19,72,201,49,144,26,17
     3,144,19,1767
01   650 data 24,125,134,19,141,14
     4,19,173,145,19,125,135,19,14
     1,145,19,1527
45   660 data 104,56,233,1,76,169,
     18,104,232,232,136,16,216,76,
     13,19,1701
aa   670 data 136,169,0,141,144,19
     ,141,145,19,162,0,185,124,19,
     201,59,1664
3c   680 data 144,3,56,233,7,72,20
     1,49,144,26,173,144,19,24,125
     ,148,1568
e8   690 data 19,141,144,19,173,14
     5,19,125,149,19,141,145,19,10
     4,56,233,1651
e6   700 data 1,76,230,18,136,104,
     232,232,192,1,176,207,96,160,
     0,140,2001
3b   710 data 147,19,177,209,200,2
     01,34,208,249,140,146,19,172,
     146,19,177,2263
36   720 data 209,201,34,240,21,20
     1,32,176,3,24,105,64,172,147,
     19,145,1793
38   730 data 44,238,146,19,238,14
     7,19,76,29,19,169,0,172,147,1
     9,145,1627
35   740 data 44,238,147,19,238,14
     7,19,238,147,19,200,173,144,1
     9,145,44,1981
dc   750 data 200,173,145,19,145,4
     4,165,44,24,109,147,19,133,44
     ,144,2,1557
93   760 data 230,45,96,169,29,133
     ,211,32,238,19,169,32,32,210,
     255,32,1932
95   770 data 210,255,32,210,255,3
     2,210,255,76,210,255,234,234,
     234,234,234,3170
67   780 data 234,234,234,234,234,
     1,0,10,0,100,0,232,3,16,39,0,
     1571
33   790 data 0,0,0,1,0,16,0,0,1,0
     ,16,36,16,39,232,3,360
b3   800 data 100,0,10,0,1,0,160,0
     ,140,37,20,162,0,165,251,232,
     1278
58   810 data 56,249,157,19,133,25
     1,165,252,249,158,19,133,252,
     16,238,165,2512
99   820 data 251,202,24,121,157,1
     9,133,251,165,252,121,158,19,
     133,252,138,2396
24   830 data 24,105,48,201,48,208
     ,9,192,8,240,5,174,37,20,240,
     6,1565
6f   840 data 238,37,20,32,210,255
     ,200,200,192,10,144,191,96,16
     2,0,142,2129
61   850 data 143,24,160,8,162,0,1
     73,144,19,56,249,134,19,141,1
     44,19,1595
14   860 data 173,145,19,249,135,1
     9,144,6,141,145,19,232,16,232
     ,173,144,1992
ca   870 data 19,24,121,134,19,141
     ,144,19,138,24,105,48,32,210,
     255,136,1569
72   880 data 136,16,209,96,0,10,1
     70,189,66,20,141,55,20,189,67
     ,20,1404
05   890 data 141,56,20,160,0,185,
     255,255,240,6,32,210,255,200,
     208,245,2468
6f   900 data 96,80,20,224,20,4,21
     ,24,21,28,21,34,21,44,21,32,7
     11
91   910 data 32,32,32,32,32,32,32
     ,32,32,77,69,78,85,32,77,65,7
     71
08   920 data 75,69,82,13,13,32,32
     ,32,32,32,32,32,32,32,70,79,6
     89
aa   930 data 82,32,68,73,83,75,32
     ,85,83,69,82,13,32,32,32,32,9
     05
4e   940 data 32,32,32,32,32,40,67
     ,41,32,49,57,56,55,32,82,65,7
     36
ce   950 data 84,84,13,13,13,13,32
     ,32,32,32,32,32,32,32,32,40,5
     48
12   960 data 49,41,32,68,73,82,69
     ,67,84,79,82,89,13,13,32,32,9
     05
ed   970 data 32,32,32,32,32,32,32
     ,40,50,41,32,68,79,83,32,67,7
     16
95   980 data 79,77,77,65,78,68,13
     ,13,32,32,32,32,32,32,32,32,7
     26
ca   990 data 32,40,51,41,32,77,65
     ,75,69,32,77,69,78,85,0,13,83
     6
5b   1000 data 73,78,80,85,84,32,7
     7,69,83,83,65,71,69,32,63,32,
     1076
6b   1010 data 68,73,83,75,32,85,8
     3,69,82,157,157,157,157,157,1
     57,157,1749
4c   1020 data 157,157,0,13,73,78,
     80,85,84,32,67,79,76,79,85,82
     ,1227
29   1030 data 32,32,63,32,50,157,
     0,20,78,79,0,20,66,58,77,63,8
     27
72   1040 data 0,20,20,20,20,66,65
     ,83,73,67,0,20,20,20,20,65,57
     9
f9   1050 data 63,32,0,54,3,237,24
     6,62,241,47,243,102,254,165,2
     44,237,2230
54   1060 data 245,120,169,120,169
     ,202,141,38,3,169,241,141,39,
     3,88,169,2057
b0   1070 data 0,133,250,169,32,13
     3,251,169,103,133,252,169,3,1
     33,253,160,2343
8d   1080 data 0,162,7,177,252,145
     ,250,200,208,249,230,251,230,
     253,202,16,2832
bd   1090 data 242,76,0,32,169,0,1
     41,32,208,141,33,208,141,0,8,
     133,1564
d6   1100 data 198,104,104,104,104
     ,104,104,162,0,169,32,157,0,4
     ,157,250,1753
fa   1110 data 4,157,244,5,157,238
     ,6,169,3,157,0,216,157,250,21
     6,157,2136
14   1120 data 244,217,157,238,218
     ,202,208,225,32,32,33,32,120,
     33,32,238,2261
cb   1130 data 33,32,153,34,120,16
     9,54,141,20,3,169,33,141,21,3
     ,88,1214
ab   1140 data 32,228,255,240,251,
     56,233,65,141,27,35,32,229,34
     ,176,240,2274
d9   1150 data 120,169,49,141,20,3
     ,169,234,141,21,3,88,169,255,
     133,157,1872
c5   1160 data 166,252,164,253,173
     ,228,34,32,189,255,169,8,162,
     8,160,1,2254
20   1170 data 32,186,255,173,228,
     34,24,105,1,168,177,252,141,1
     92,32,200,2200
2e   1180 data 177,252,141,193,32,
     13,192,32,208,5,169,44,141,19
     1,32,162,1984
fd   1190 data 45,189,174,32,157,6
     4,3,202,16,247,169,147,32,210
     ,255,76,2018
25   1200 data 64,3,169,0,162,255,
     160,255,32,213,255,134,45,134
     ,47,132,2060
80   1210 data 46,132,48,76,255,25
     5,169,82,141,119,2,169,85,141
     ,120,2,1842
60   1220 data 169,78,141,121,2,16
     9,13,141,122,2,169,4,133,198,
     96,169,1727
f6   1230 data 236,141,0,4,169,251
     ,141,39,4,169,226,162,37,157,
     1,4,1741
4c   1240 data 202,16,250,169,40,1
     33,250,169,4,133,251,162,4,16
     0,0,169,2112
4a   1250 data 97,145,250,160,39,1
     69,225,145,250,32,216,34,202,
     16,238,162,2380
41   1260 data 37,169,98,157,241,4
     ,202,16,250,169,252,141,240,4
     ,169,254,2403
eb   1270 data 141,23,5,96,162,0,1
     69,2,157,0,216,157,24,216,169
     ,32,1569
a0   1280 data 157,0,4,157,24,4,20
     2,208,237,96,173,236,33,238,2
     36,33,2038
e4   1290 data 41,63,208,6,32,219,
     32,76,77,33,201,32,208,3,32,8
     0,1343
25   1300 data 33,76,49,234,169,40
     ,133,250,169,4,133,251,162,39
     ,169,32,1943
c4   1310 data 157,0,4,157,240,4,2
     02,16,247,162,4,169,32,160,0,
     145,1699
f8   1320 data 250,160,39,145,250,
     32,216,34,202,16,240,96,169,2
     16,141,149,2355
bb   1330 data 33,169,33,141,150,3
     3,160,0,169,0,72,169,45,133,2
     50,169,1726
31   1340 data 4,133,251,169,0,72,
     162,7,173,255,255,61,208,33,2
     40,4,2027
05   1350 data 169,160,145,250,200
     ,202,16,240,32,216,34,152,56,
     233,8,168,2281
2e   1360 data 173,149,33,24,105,1
     ,141,149,33,144,3,238,150,33,
     104,24,1504
4b   1370 data 105,1,201,5,144,207
     ,152,24,105,7,168,104,24,105,
     1,201,1554
53   1380 data 4,144,183,96,1,2,4,
     8,16,32,64,128,198,238,254,21
     4,1586
a0   1390 data 198,126,96,120,96,1
     26,102,118,126,110,102,102,10
     2,102,102,60,1788
a5   1400 data 0,0,169,64,162,39,1
     57,24,5,157,104,5,157,152,7,2
     02,1404
86   1410 data 16,244,160,0,185,11
     1,34,240,9,41,63,153,64,5,200
     ,76,1601
64   1420 data 0,34,162,12,169,144
     ,133,250,169,5,133,251,169,1,
     141,152,1925
d3   1430 data 34,160,0,169,93,145
     ,250,160,20,145,250,160,19,14
     5,250,160,2160
d4   1440 data 39,145,250,160,2,16
     9,58,145,250,160,22,145,250,1
     73,152,34,2154
ba   1450 data 160,1,145,250,24,10
     5,13,160,21,145,250,238,152,3
     4,32,216,1946
56   1460 data 34,202,16,205,169,1
     14,141,104,5,141,123,5,141,12
     4,5,141,1670
76   1470 data 143,5,169,74,141,15
     2,7,141,172,7,169,75,141,171,
     7,141,1715
12   1480 data 191,7,96,32,32,32,3
     2,32,32,32,32,32,32,32,32,32,
     710
b4   1490 data 32,32,32,32,32,32,3
     2,32,32,32,32,32,32,32,32,40,
     520
62   1500 data 67,41,49,57,56,55,3
     2,82,65,84,84,0,0,169,147,133
     ,1121
e7   1510 data 250,169,5,133,251,1
     69,0,141,27,35,32,229,34,176,
     44,160,1855
6e   1520 data 0,177,252,41,63,145
     ,250,200,204,228,34,144,244,3
     2,216,34,2264
9e   1530 data 238,27,35,173,27,35
     ,201,13,144,224,208,11,169,16
     7,133,250,2055
de   1540 data 169,5,133,251,173,2
     7,35,201,26,144,207,96,165,25
     0,24,105,2011
f1   1550 data 40,133,250,144,2,23
     0,251,96,0,169,30,133,252,169
     ,35,133,2067
aa   1560 data 253,162,0,160,0,177
     ,252,201,255,240,31,200,201,0
     ,208,245,2585
b4   1570 data 136,140,228,34,236,
     27,35,240,19,152,24,105,3,24,
     101,252,1756
1b   1580 data 133,252,144,2,230,2
     53,232,76,239,34,56,96,24,96,
     0,0,1867
94   1590 data 148,24,0,0,0,0,0,0,
     0,0,0,0,0,0,0,0,172
```

Disk Command Summary

To send a command to the disk drive use :

OPEN 1,8,15,"command":CLOSE 15

LOAD

LOAD "file",8	LOAD to start of Basic.
LOAD "file",8,1	LOAD file to address which it was saved from.
DLOAD "file"	LOAD basic file in Basic 7.0.
BLOAD "file",Bbank,Pstart address	Load file to different address. Basic 7.0 only.
BOOT "file"	Load and execute file (Basic 7)

SAVE

SAVE "file",8	Save a Basic file.
DSAVE "file"	Save a Basic file in Basic 7.0.
BSAVE "file",Bbank,Ps TO Pe	Save code in Basic 7.0 where: s = Start address; e = End address.

FILE ACCESS

OPENfile-no,8,cannel-no,"0: file-name,file-type,direction"	Open disk file where: File-type = P,S,R etc. Direction = R for read or W for write.
DOPEN#file-no,"file-name",	Open disk file (Basic 7.0)
Ddrive-no,Uunit-no,W	NB. W only needed for write.
CLOSEfile-no	Close open file
DCLOSE#file-no	Close file (Basic 7.0)
PRINT#file-no,data	Send data to file.
GET#file-no,variable	Get data from file.
INPUT#file-no,variable	Input data from file.

DIRECT ACCESS

"B-A";0;track-no;sector-no	Mark track/sector as used.
"B-F";0;track-no;sector-no	Mark track/sector as free.
"B-E";channel-no;0;track-no; sector-no	Execute code at track/sector.
"B-P";channel-no;byte	Move to byte in disk buffer.
"U1";channel-no;0;track-no; sector-no	Read track/sector into buffer.
"U2";channel-no;0;track-no; sector-no	Write buffer to track/sector.
"M-R"CHR$(address-lo)CHR$ (address-hi)CHR$(no of bytes)	Read disk memory at address.
"M-W"CHR$(address-lo)CHR$ (address-hi)CHR$(no of bytes) CHR$(data)CHR$(data)etc.	Write to disk memory at address
"M-E"CHR$(address-lo)CHR$ (address-hi)	Execute machine code in drive at address.

TRYING TO USE YOUR COMPUTER?...

YOUR
COMMODORE

CAN HELP.

GREAT NEW RATES

12 issues UK £13.20
12 issues Middle East £21.20
12 issues Far East £24.00
12 issues Rest of World £21.60

Send your remittance to:
INFONET LTD., 5 River Park Estate, Berkhamsted, Herts. HP4 1HL.

Access VISA

Please begin my subscription(s) to YOUR COMMODORE with the .... issue.
I enclose my cheque/postal order for £.................. made payable to Argus Specialist Publications Ltd.
or debit £.................. from my Access/Barclaycard No.
valid from .... to ....
NAME (Mr/Mrs/Miss) ....
ADDRESS ....
Postcode ....
Signature ....
Date ....

Please use BLOCK CAPITALS and include post codes

BE IN ON THE ACTION . . . FROM THE VERY START

COMMODORE DISK USER is a lot more than just another computer magazine. Every issue carries a diskette containing more than £30 worth of software ranging from serious programming utilities to arcade games. There are plenty of Commodore magazines on the market, but we believe that this is the *first* to cater for disk users of all ages and tastes.

The MEGAVALUE first issue will include on disc:

- ★ **3 into 1 Plus – A superb all-purpose graphics editor**
- ★ **Mobsters – An extremely addictive arcade strategy game**
- ★ **Disk Designer – Revamp your disk director, any way you like it.**
- ★ **Ski Run – A compulsive 3D winter sports simulation. Break a leg!**

. . . and lots more.

The magazine will include full details of how to use the programs, plus a wealth of reviews and features. This month we interview the US software houses poised to invade the UK, Compunet tells us all its secrets, and you get the chance to win a selection of Microprose software. Plus news, reviews, and our mind-bending puzzle page.

COMMODORE DISK USER is what you have been waiting for – take out a subscription TODAY!

SUBSCRIPTION RATES

£15.00 for 6 issues U.K.
£18.00 for 6 issues EUROPE
£18.20 for 6 issues MIDDLE EAST
£19.30 for 6 issues FAR EAST
£18.40 for 6 issues REST OF WORLD
Airmail Subscription Rates on Request

Send your remittance to:
INFONET LTD., 5 River Park Estate, Berkhamsted, Herts. HP4 1HL.

Please begin my subscription(s) to COMMODORE DISK USER with the .... issue
I enclose my cheque/postal order for £............ made payable to Argus Specialist Publications Ltd.
or debit £............ from my Access/Barclaycard No. ....
valid from .... to ....
NAME (Mr/Mrs/Miss) ....
ADDRESS ....
....
Postcode ....
Signature ....
Date ....

Please use BLOCK CAPITALS and use post codes.