G E M I N I

M S - C O P Y

U S E R

M A N U A L

ISSUE 2

05-10-85

MSCOPY

MSCOPY is a program used to transfer files to and from CP/M and MS-DOS disks. (IBM PC-DOS disks can be considered as special cases of MS-DOS formats). The details of the MS-DOS formats required must have been previously entered into the FORMATS.DAT file. The program can also be used to initialise MS-DOS disks so that the directory and space allocation data is empty, to list the directories of both CP/M and MS-DOS disks and to erase files on both sorts of disk.

New MS-DOS formats can be added to the list (if the appropriate disk parameters have been determined) through the use of the SETUP program described earlier. SETUP will prompt for extra MS-DOS information if the Operating System type is specified as 1 instead of 0 for CP/M when entering format details.

Running MSCOPY

To run MSCOPY enter

MSCOPY <return>

MSCOPY will produce a menu of the form shown below.

```
-------------------------------------------------------------
MS-DOS <--> CP/M File Transfer Utility  Ver 3.2  5/10/1985
(C) TC Software. MSCOPY Utility for Gemini MFB Serial No: xxxx
-------------------------------------------------------------
   OPERATION    COMMAND
   Copy ........ COPY  Drive:=Drive:Filespec  [ options
   Directory ... DIR   Drive:Filespec         [ options
   Erase ....... ERA   Drive:Filespec         [ options
   Initialise... INIT  Drive:                 [ options
   Quit ........ <RETURN>
   Help ........ ? <RETURN>

 Option P       .... Directs output for this command to the printer
 Option W       .... Suppresses checks when over-writing Read/only data

=============================================================
A: RO202    B: RO202    C: GEMQDDS  D: ICLPC2   E: IBMPC8SM F: IBMPC8DM
G: IBMXT9DM
=============================================================
MSCOPY >>
=============================================================
```

Any formats which are MS-DOS ones will be shown in the format list as highlighted. The required command can then be entered after the prompt.

A printed copy of the output of any command can be produced by adding the option '[P' to the command line eg

```
DIR *.COM [P
```

Note that the trick of using <CNTL-P> to copy screen data to the printer is not effective with MS-COPY.

Copying files

The command for copying files follows a style similar to that of PIP, the CP/M copying program, except that the word 'COPY' is used instead of 'PIP'. The formal syntax is:

```
COPY <dest-drive>=<source-drive><file-ref>
```

where

<dest-drive>	is the drive to receive the data eg E:
<source-drive>	is the drive from which the data will be copied eg D:
<file-ref>	is a (possibly ambiguous) file reference eg *.BAS

For instance to copy all files from Drive A: to Drive D: the correct command would be:

```
COPY D:=A:*.*
```

where '*.*' means 'all files'.

MSCOPY will determine from the format information which disk is MS-DOS and take the appropriate action. Note that it is NOT possible to transfer files directly between 2 MS-DOS disks. The files would have to be transferred to a set of intermediate CP/M files to achieve this effect. You will also receive an error message if both disks are CP/M format, since such transfers should be done using PIP.

Disks to receive MS-DOS files must have been formatted and initialised before use. Initialisation is an extra step which is not required for most CP/M disks since the formatting usually has the required effect of initialising the directory.

When transferring files to CP/M the file length will be rounded up to the nearest 128 bytes, since all CP/M files have to be a multiple of 128 bytes in length. The padding will normally be with the character 1A Hex which is treated as an end of file mark by most CP/M programs.

The CP/M files will be transferred to or from the current default user number. MS-DOS disks do not have such a concept as user numbers. Files can only be transferred to or from the root directory of MS-DOS disks, that is to say, 'path-names' are not supported.

It is not possible to change the name when the file is copied eg

```
COPY D:ABC.COM=A:XYZ.COM    .... is invalid
```

The display after a typical file transfer to a MS-DOS disk is shown below:

```
MSCOPY >> copy g:=a:*.sub

File(s) *.SUB being copied from A: (CP/M) to G: (MS-DOS)

                CP/M Files *.SUB

MSLOAD  .SUB  MEMLOAD .SUB  TORCHLD .SUB  BBCLOAD .SUB  $$$     .SUB
FLOPSAVE.SUB  GILL    .SUB
-------------------------------------------------------------------------

                Number of files to be transferred 7
                1 / 7  MSLOAD  .SUB has been created.
                2 / 7  MEMLOAD .SUB has been created.
                3 / 7  TORCHLD .SUB has been created.
                4 / 7  BBCLOAD .SUB has been created.
                5 / 7  $$$     .SUB has been created.
                6 / 7  FLOPSAVE.SUB has been created.
                7 / 7  GILL    .SUB has been created.
=========================================================================
A: RO202    B: RO202    C: GEMQDDS  D: ICLPC2   E: IBMPC8SM F: IBMPC8DM
G: IBMXT9DM
=========================================================================
MSCOPY >>
=========================================================================
```

The display for the same files transferred in the other direction is shown below:

```
MSCOPY >> copy a:=g:*.sub
═══════════════════════════════════════════════════════════════════

File(s) *.SUB being copied to A: (CP/M) from G: (MS-DOS)

-------------------------------------------------------------------
MSLOAD  .SUB length 512 ( 1 clusters) is being copied to A:MSLOAD.SUB
MEMLOAD .SUB length 384 ( 1 clusters) is being copied to A:MEMLOAD.SUB
TORCHLD .SUB length 384 ( 1 clusters) is being copied to A:TORCHLD.SUB
BBCLOAD .SUB length 256 ( 1 clusters) is being copied to A:BBCLOAD.SUB
$$$     .SUB length 128 ( 1 clusters) is being copied to A:$$$.SUB
FLOPSAVE.SUB length 256 ( 1 clusters) is being copied to A:FLOPSAVE.SUB
GILL    .SUB length 128 ( 1 clusters) is being copied to A:GILL.SUB

═══════════════════════════════════════════════════════════════════
A: RO202    B: RO202    C: GEMQDDS  D: ICLPC2   E: IBMPC8SM F: IBMPC8DM
G: IBMXT9DM
═══════════════════════════════════════════════════════════════════
MSCOPY >>
═══════════════════════════════════════════════════════════════════
```

The current copy operation will be aborted should the destination disk run out of space whilst the copy is taking place. All but the last file will be transferred normally.

Various checks are carried out before the transfer takes place. One check is that the Media Byte of the destination disk matches that specified in the format details. If it does not match then the copy will be aborted. The following message will be displayed in this case:

'Media byte of xxxxH does not match format details of zzzzH. Action not completed.'

This situation can be avoided either by specifying the correct format if you have made a mistake, or by entering a new set of format details which have a matching Media Byte, by the use of the SETUP command.

The time and date fields of the MS-DOS files created are completed with arbitrary values, which relate to the date on which the version of the MSCOPY program was created. These fields cannot be changed by the user. MS-DOS files created by MSCOPY are set to Read/Write and are visible in the directory. MSCOPY will prompt if you ask it to overwrite an existing file of the same name which has the READ ONLY attribute set on the MS-DOS disk. This prompt can be avoided if the '[W' option is specified on the command line.

Directory listing

MSCOPY can list both CP/M and MS-DOS directory data with its 'DIR' command. The syntax is the same as the CP/M 'DIR' command.

The display for an MS-DOS disk includes overall data on the space used on the disk. A typical directory listing is shown below:

```
MSCOPY >> dir g:

Disk is   2.0% full.          Space used          7K
Free clusters   346.          Free data space   346K

MSLOAD  .SUB Size    256      MEMLOAD .SUB Size   384
TORCHLD .SUB Size    256      BBCLOAD .SUB Size   256
$$$     .SUB Size    128      FLOPSAVE.SUB Size   256
GILL    .SUB Size    128

A: RO202    B: RO202    C: GEMQDDS  D: ICLPC2   E: IBMPC8SM F: IBMPC8DM
G: IBMXT9DM

MSCOPY >>
```

Note that the amount of space listed for free data may seem less than is normally stated for some MS-DOS disks, however the commonly quoted figures omit the space required for the directory and the File Allocation Table (FAT).

The CP/M directory listing is limited to the first 256 entries which match the file reference entered.

Erase command

The 'ERA' command is provided as a convenience to allow CP/M or MS-DOS files to be erased. It has the same syntax as the CP/M command of the same name.

Note that any attempt to erase a CP/M Read-only file will result in termination of the MSCOPY program.

Initialising MS-DOS disks

Initialising a MS-DOS disk clears the directory and the information held on which parts of the disk have been used to hold data. Disks can only be initialised after they have been formatted.

The 'INIT' command allows disks in most MS-DOS formats to be initialised according to the data specified in the FORMATS.DAT file.

To initialise a disk the command 'INIT' followed by the required drive is entered eg

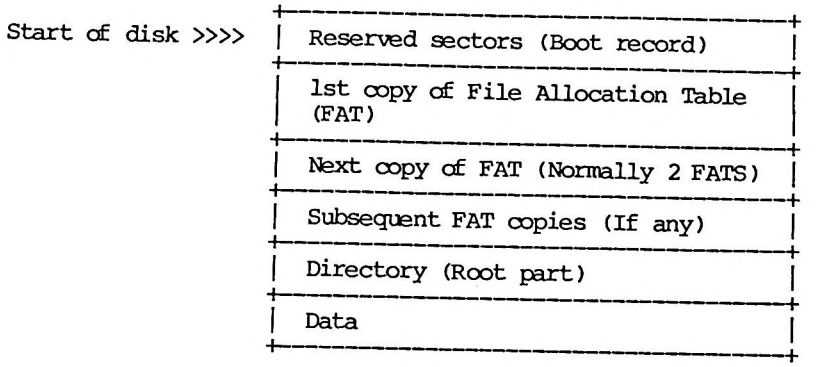
INIT G:

Any attempt to initialise a disk in a CP/M drive will be trapped. The check on whether a disk contains valid MS-DOS data in the current format on the specified drive can be by-passed by use of the '[W' option added at the end of the command line:

eg INIT G: [W

APPENDIX MS-DOS DISK STRUCTURE

MS-DOS disks have a structure as shown in the diagram below:



Boot Record

The boot record is the part of the disk which may contain information to start up the system. The boot record is not mandatory, some manufacturers omit it on disks which are to store data only. To read disks without a Boot record specify a Boot record size of 0 when entering the format details.

Microsoft have recommended that certain information at the start of the boot record should be standardised for MS-DOS 2.00 and later. Earlier systems used the FAT ID byte as a method of determining the disk type, but this was later found to be to restrictive.

Here is the recommended contents of the start of the boot record.

Offset Decimal	Contents
+0	3 BYTE JUMP to boot code
+3	8 BYTES OEM name and version
+11	WORD bytes per sector
+13	BYTE sectors per cluster
+14	WORD reserved sectors
+16	BYTE number of FATs
+17	WORD number of root dir entries
+19	WORD number of sectors in logical image
+21	BYTE media descriptor
+22	WORD number of sectors per FAT
+24	WORD sectors per track
+26	WORD number of heads
+28	WORD number of hidden sectors

Not all OEMS have implemented this recommendation. Some have completed some of the information, some were not sure what the terms above meant, for instance is the number of sectors per track the Physical number of sectors per track or the logical number of sectors per track? Hence this is an interesting and usually informative area in which to look for data about a new type of MS-DOS disk. The last three items of data are optional in the suggested standard.

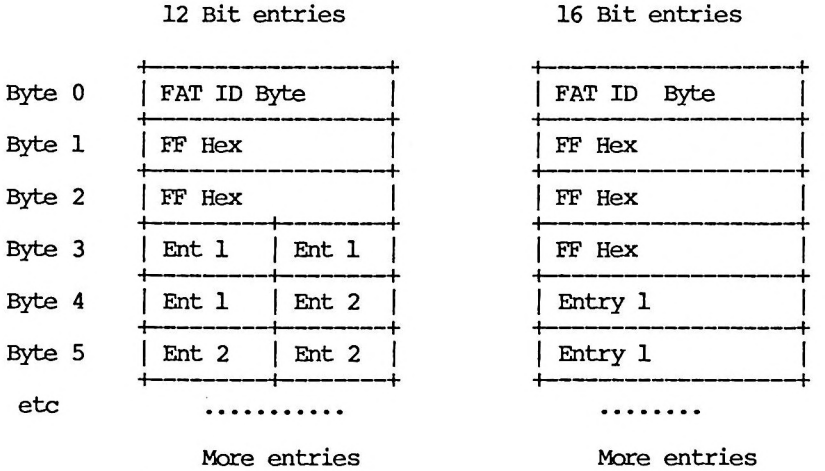
File Allocation Table (FAT)

The File Allocation Table (FAT) is used to hold the data concerning which parts of the disk have already been allocated to data. MSDOS disks allocate space in units called Clusters. The cluster size is one of the parameters which are specified to SETUP when a new MSDOS format is created.

Most implementations of MS-DOS have 2 copies of the FAT; some however have decided to have only one FAT. MSCOPY uses only the first copy of the FAT and duplicates it to any other FATS when re-writing the 1st FAT. One to four FATS are supported by SETUP, plus a large range of sizes of the file allocation table. The start of a FAT on an empty disk can usually be recognised by the Media byte followed by FFFFh and then zeroes. The size of a FAT can usually be deduced by finding the start of the next FAT in the same way, and then subtracting the sector offset.

The FAT consists of a number of entries each of which can be either 12 bit or 16 bit values. So far 16 bit values in the FAT have been confined to use on non-removable media on the IBM AT. MSCOPY does not support 16 bit values in the FAT at present.

The FAT structure is as shown below:



The FAT ID byte can be any value in the range F0 hex to FF hex, but the values in the table below have some degree of standardisation in that they apply to the IBM PC, XT and AT as follows:

(AT)	F9	- 2 side	512 bytes per sector	15 sectors per track
(XT)	FC	- 1 side	512 bytes per sector	9 sectors per track
(XT)	FD	- 2 side	512 bytes per sector	9 sectors per track
(PC)	FE	- 1 side	512 bytes per sector	8 sectors per track
(PC)	FF	- 2 side	512 bytes per sector	8 sectors per track

SETUP allows any value to be entered for the media byte.

The position of the entry in the FAT indicates which cluster it refers to. Apart from the special entries at the start of the FAT, the usage of entry values is as shown below:

0	-		Cluster is unused and available
(F)FF0	-	(F)FF7	Reserved or defective cluster
(F)FF8	-	(F)FFF	Only found as last cluster of file
(X)XXX	-		Cluster number of the next cluster in the file. (Only the first cluster number is stored in the directory entry).

Directory

Each entry in an MSDOS directory consists of 32 bytes, organised as shown below:

Offset Decimal	Contents
+0	File name. 1st byte multi-function
+8	File extension (3 bytes)
+11	Attributes (1 byte)
+12	Reserved (10 bytes)
+22	Time of last update (2 bytes)
+24	Date of last update (2 bytes)
+26	Starting cluster number (2 bytes)
+28	File size in bytes (4 bytes)

File name

The following special values of the first byte of the file-name indicates the file status as follows:

00H	End of used entries in the directory
05H	Special value indicates that file-name starts with E5h. Not supported by MSCOPY.
2EH	Entry for sub-directory name. Sub-directories are not supported by this release of MSCOPY.
E5H	Indicates erased file

Any other value is part of the filename.

Attributes

The attribute byte is used to set special file characteristics as shown below:

Bits set	Meaning
....1	Read-only
.... ..2.	Hidden file
.... .4..	System file
.... 8...	Volume label
...1	Sub-directory
..2.	Archive bit
.4..	Not used
8...	Not used

Cluster number

Valid cluster numbers start at 2. A cluster number of 0 indicates an empty file.

File size

The file size is stored in two words. The first word is the low value; the second is the high value. Each word is stored with the low byte first.

APPENDIX 2. Using MSCHECK to check format details

The format details of unknown MS-DOS disks can be investigated using a combination of utilities supplied with each MFB. Firstly the physical details of the disk can be examined by using the ANALYSE utility, documented earlier in the system manual.

You should then use SETUP to enter the details found into a trial format which should include :

Tracks per inch

Single or double sided

(If double sided then most MS-DOS disks will need the 'C' option for the side type, not 'T' or 'S')

Physical sector size (Usually 512 bytes/sector)

Physical sectors per track (Usually 8 or 9)

Physical sector skew (Normally none i.e. 0)

Starting sector number (Normally 1)

Index marks (Usually yes)

Gap sizes (used in formatting the disk)

Don't forget that if you change the details of a format using SETUP that it will not become effective until you constuct a new BIOS using the SETUP menu. The disk type should be set to MS-DOS if you intend to use MS-CHECK. Put in the MS-DOS details which match those of the IBM Doubled Sided 9 sectors/track where the required logical data is not known. It should be noted that MSCHECK will not work if the physical details of the format are incorrectly set up. It is recommended that the Boot Size be set to 0 sectors, since this information can usually be easily deduced by MS-CHECK.

Now run MSCHECK by typing

MSCHECK <RETURN>

The program will prompt for the required drive identity which must be set up with an MS-DOS format.

MS-CHECK first reads the boot sector, and as discussed in the Appendix on the MS-DOS disk structure this often contains much useful information on the format details. If this appears to contain valid data then MS-CHECK displays the data in a useable form.

Next MS-CHECK attempts to locate the File Allocation Tables. If successful it will display the number of Boot Sectors, ie the number of sectors before the 1st FAT, the FAT size and the FAT ID byte. It expects FATs to begin with a byte of Fx Hex followed by FFFF Hex, where x is the any value 0-ffh and is the media byte. It starts looking at for the first FAT at the number of reserved sectors which you specified to SETUP. If that sector does not contain a valid looking FAT header then it starts at the beginning of the disk, and will search for up to 256 sectors for the first FAT.

MS-CHECK then attempts to locate a 2nd FAT (nearly all MS-DOS disks have 2 FATs). It looks sequentially through the disk until it comes across a 128-byte sector which matches the start of the first FAT. When it finds a match it assumes that this is the start of the 2nd FAT. By seeing how far this was from the 1st FAT the number of sectors in each FAT can be determined.

If MS-CHECK decides the FAT begins too early, then you could try setting the number of reserved sectors higher.

MS-CHECK compares the values it deduces or read from Boot sector and warns of any changes to the format details that it thinks are necessary to match the data. A typical set of outputs is shown below.

MS-CHECK Analysis of a disk where some parameters need changing

```
--- Data from the Boot SECTOR for Drive E  Format Name TRIAL2 ---
OEM Name/Version..: Gem1001
Bytes-per-sector..: 512
Sectors/cluster...: 8  (Calc as 128 Byte Sectors) **** Change required ****
Sectors/FAT.......: 8  (Calc as 128 Byte Sectors)
Directory size....: 128  **** Change required ****
Boot size (Sect)..: 4  (Calc as 128 Byte Sectors) **** Change required ****
Disk capacity.....: 720 K
Media byte .......: F1 **** Change required ****
Number of FATS....: 2
---------- Details from the optional system data area --------
NOTE: The ANALYSE utility will give more accurate data than this.
Sectors-per-track.: 9  May be physical or logical count.
Surfaces-per-disk.: 2
Hidden Sectors....: 0  Not normally used.
------------- Processing File Allocation Table (FAT ) ---------
Found   1st FAT after 4 Reserved Boot Sectors.
Found   2nd FAT after 12 Sectors.
-------- Data deduced from FAT position and contents ----------
Boot size (sect)..: 4  (Calc as 128 Byte Sectors) **** Change required ****
FAT ID Byte ......: F2 **** Change required ****
Sectors/FAT ......: 8  (Calc as 128 Byte Sectors)
-----------------------------------------------------------------
Enter drive reference for MS-DOS Disk  :
```

Using the output from this process it is a simple matter to set up a new MS-DOS format. The analysis below shows no errors.

```
--- Data from the Boot SECTOR for Drive G  Format Name TRIAL ---
OEM Name/Version..: Gem1001
Bytes-per-sector..: 512
Sectors/cluster...: 8  (Calc as 128 Byte Sectors)
Sectors/FAT.......: 8  (Calc as 128 Byte Sectors)
Directory size....: 112
Boot size (Sect)..: 4  (Calc as 128 Byte Sectors)
Disk capacity.....: 360 K
Media byte .......: FD
Number of FATS....: 2
---------- Details from the optional system data area --------
NOTE: The ANALYSE utility will give more accurate data than this.
Sectors-per-track.: 9  May be physical or logical count.
Surfaces-per-disk.: 2
Hidden Sectors....: 0  Not normally used.
------------- Processing File Allocation Table (FAT ) ---------
Number of reserved sectors confirmed. FAT found.
Found   2nd FAT after 12 Sectors.
-------- Data deduced from FAT position and contents ----------
Boot size (sect)..: 4  (Calc as 128 Byte Sectors)
FAT ID Byte ......: FD
Sectors/FAT ......: 8  (Calc as 128 Byte Sectors)
-----------------------------------------------------------------
```

Disks which do not have the Boot Sector Implemented

Some manufacturers have not implemented the boot sector recommendation from Microsoft. In these cases MS-CHECK will display the first sector as a Hex dump, and then attempt to locate the FAT(s). It will then be necessary to use a disk utility such as DU to examine the disk structure to investigate issues like directory sizing, cluster sizing etc. An example of output from MS-CHECK in this situation is shown below:

```
═══════════════════════════════════════════════════════════
--- Data from the Boot SECTOR for Drive G:  Format Name IBMXT9DM ---
OEM has not implemeted MS-DOS Boot Sector.
Dump of 1st Sector.
  EB1C0001 00020201  00027000 80020908 k.......  ..p.....
  0A000103 000118B0  0000A00F 881C0E1F  .......0  ..  .....
  B800148E C0B9003C  BE00048B FEFCF3A4  8...@9.<  >...~|s$
  EA350400 148CC88E  D833DB8E C3268E06  j5....H.  X3[.C&..
  FE03268A 1E090089  1E170426 8A1E5A00  ~.&.....  ...&..Z.
  081E1604 8EC0B008  E6FC8B1E 13048A0E  .....@0.  f|......
  0E04D3E3 F6061604  04751380 3E120402  ..Scv...  .u..>...
  7D0CE85D 01A80874  05800E15 040CB90B  }.h].(.t  ......9.
-----------------------------------------------------------
Number of reserved sectors confirmed. FAT found.
Found   2nd FAT after 8 Sectors.
--------- Data deduced from FAT position and contents ----------
Boot size (sect)..: 4  (Calc as 128 Byte Sectors)
FAT ID Byte ......: FF **** Change required ****
Sectors/FAT ......: 4  (Calc as 128 Byte Sectors)
-----------------------------------------------------------
═══════════════════════════════════════════════════════════
```

END OF APPENDIX