MFU

Version 4

User Guide

MOONSTONE
COMPUTING

MULTI - FORMAT UTILITY SOFTWARE

Version 4

USER GUIDE

AND

TECHNICAL REFERENCE MANUAL

COPYRIGHT NOTICE

Published by Moonstone Computing 1987

Printed in the U.K.

CONTENTS

1. INTRODUCTION

Welcome to the Users' Guide and Technical Reference Manual for Moonstone Computing's Multi-Format Utility (MFU). When you run MFU on an Amstrad PCW8256 or CPC464/664/6128 equipped with a 5¼" external disc drive, you instantly have one of the most powerful Multi-Format systems available anywhere - at a fraction of the multi-thousand-pound prices of other machines available to do a similar job.

If you have also bought one of our 5¼" disc drives, please read the other book in the package - the Hardware Installation Manual - before proceeding any further here. This will guide you through attaching and testing your disc drive and/or RAM upgrade. The Hardware Installation Manual is also available either separately or with cabling if you already have, or wish to acquire, a third-party disc drive.

This book is designed to guide you through the process of installing and using the MFU software. The next section deals with copying of your master disc to create a working copy, as well as covering installation and customisation to suit your own requirements. Following that, there is a general but comprehensive guide to firstly the concepts involved in disc formats, and secondly the CP/M operating system as it affects us.

Following this, we describe in detail the various ways MFU can be used, covering all the options from analysing unknown discs to formatting new ones or transferring data between your Amstrad and other computers.

At the end of the manual, there is an extremely comprehensive technical reference section, containing several detailed Appendices. These give in-depth information on several topics which were covered more generally in the main part of the manual for those who require this level of information. Topics include the way the Disc Controller chip writes information on a disc, CP/M's internal Disc Parameter data structures and disc Directory formats, as well as a full explanation of the various disc errors which can occur, and how to remedy them. Details of the hardware modifications required to use Single Density discs with a PCW8256/8512 are also given.

So what does MFU do?

Using the MFU system, you can read and write data on virtually **any** type of soft-sectored 5¼" floppy disc.

MFU's functions fall into three main groups :

1. Drive B can be set up to exactly emulate almost all types of CP/M formats. This means that you can use a disc (even run programs which are on it!) from another CP/M computer on your Amstrad. Once a format is set up, you can read, write and even format discs which the other computer will also be able to use.

   This instantly gives you access to the huge amount of existing CP/M software which is available, but not on Amstrad format discs! If you use other CP/M computers either at home or at work, you can use the **same discs** in them and your Amstrad.

   MFU includes a comprehensive library, containing over two dozen common CP/M formats (such as Superbrain, Wren, Osborne, Epson and DEC Rainbow), so **no** expert knowledge is required to emulate and use such a format. However, if the format for a computer isn't on the list, you can define it yourself and add it to the library for future use.

   Once a format has been loaded, **no** further software is required to use it - you can leave MFU and your computer will treat Drive B **exactly** as the other computer would. Any programs you want to run will know that Drive B is emulating the other format and will read and write data correctly.

2. It is not possible to emulate non-CP/M formats in the same way, unfortunately. Different computers use different ways of storing data on disc, and programs from one will not in general run on any other. However, MFU lets you transfer files between CP/M and a variety of other computers, including IBM-PCs and compatibles, and BBC and Master computers.

   Thus you can easily transfer your data around between different types of computers on disc, getting round the traditional problem of different computers being completely incompatible with each other.

   So, for example, you could write a letter on a BBC at home, use your Amstrad and MFU to transfer it to an IBM-PC at work for printing on the office laser printer - and then copy it onto a disc which your secretary can use in her Superbrain CP/M machine!

3. For the expert, MFU allows low-level analysis, inspection and alteration of the data and programs held on virtually any 5¼" disc, even those which MFU's higher-level emulation and file transfer functions cannot support.

   MFU has the ability to automatically analyse and determine the physical format of any disc, including density, number of sides, track pitch, number of sectors on a track and their size. Physical sectors can be read, edited and written in either hex or ASCII using the built-in sector editor.

   It is also possible to create physical copies of almost any type of disc, again including formats which the higher-level functions cannot handle.

MFU has been specially written for Amstrad computers to take full advantage of the capabilities of their Floppy Disc Controller (FDC) chip, and of Amstrad's CP/M Basic Input/Output System (BIOS). For example, when a 40 track format is installed and your drive has 80 tracks, MFU will automatically instruct the BIOS to "double-step" the heads - this means that you **don't** need a switchable drive to use both 40 and 80 track formats!

Throughout this manual, all topics will be presented in as simple and straightforward a manner as possible. MFU is designed to be simple and friendly to use, and you do **not** need any detailed knowledge of CP/M before you can install and use formats from the library. Obviously, however, some of MFU's functions (such as editing the format parameters, for example) **do** require a good working knowledge of the meanings and functions of these parameters, and readers are referred to appropriate parts of the Technical Appendices where these are relevant.

Note for CPC464/664/6128 owners :

Please note that throughout this manual, we refer to the **RETURN** and **EXIT** keys, as found on a PCW. For a CPC version of MFU, the corresponding keys are **ENTER** and **ESCape** respectively.

UPGRADE POLICY

We are continually improving and enhancing MFU, and it is our policy to issue periodic updates to existing customers. A small handling charge will be involved to cover the costs to us of manuals and postage, but upgrades will otherwise be **FREE** of charge if the previous master disc is returned to us. All existing customers are notified automatically by us when major upgrades are released, but if you find a bug or have a difficulty you can't understand please contact us - it may be that someone else has reported the same problem and a revision is available.

We are always willing to offer all reasonable support and assistance to our customers; if you have a problem or a special requirement please contact us. If you write, we would appreciate a stamped S.A.E for our reply.

2. INSTALLING MFU.

In general, **MFU requires no special installation before it can be run.**

However, for technical reasons it is only possible for us to support BIOS version 1.4 (for a PCW8256/8512) or 1.0 (for a CPC464/664/6128).

You can determine your BIOS version by looking at your computer's sign-on message when you boot.

If you have a different version of CP/M, MFU will NOT run.

If you find that you have an **earlier** version of CP/M, you should contact Amstrad themselves and request an upgrade of your CP/M and LocoScript disc to the current release; they will do this for you **free of charge. Please do NOT ask US for an updated CP/M - we CANNOT supply it!**

If you find that you have a **more recent** version of CP/M, please let us know - we'd like to hear about it!

IF you are using a switchable drive, please note the following:

If your version of MFU is set up for an 80 track drive (it WILL be unless you have requested a special version), ALWAYS have your drive set to "80 track" mode - EVEN when using a 40 track format. MFU will adjust the system to double-step automatically whenever required.

2.1 Distribution files

The current version of MFU has a number of files which are necessary for its full operation; these are as follows :

MFU.COM	- This is the main program; to run the system simply type "MFU" at the M> prompt.
MFU.DAT	- This is MFU's library file, which contains details of all its stored formats. This is distributed with a large selection of common formats, but you can alter these or add to them easily from within MFU.
MFU.000 to MFU.009	- These are machine code overlay files to which are required by MFU to perform its various functions.
MFUPATCH.COM	- This is a special "patcher" program which allows you to customise your working copy of MFU. You can alter any of the system defaults and specify printer and screen control codes.

Use CP/M's

A>**dir**

command on Side A of your distribution disc to make sure that all the files are present.

PCW versions ONLY :

With PCW versions of MFU **only**, we also supply a disc formatting utility called **XFORMAT**. This is a replacement for Amstrad's **DISCKIT** utility which formats blank discs both faster and to a much higher capacity! Using XFORMAT instead of DISCKIT will give you an extra 20K of space on a CF2 Drive A disc and an extra 80K on a CF2DD format disc on Drive B (including 5¼" versions, obviously!).

The XFORMAT program is contained in a single file, named

XFORMAT.COM

on Side B of the distribution disc.

Instructions for using XFORMAT will be found in a separate booklet.

2.2 Backing up your distribution disc.

Please, **before** you do anything else, make a working copy of your MFU distribution disc! Then put the original safely away to prevent it becoming corrupted or damaged in use. **NEVER** use the master copy of the software as the working version - sooner or later you will have a very frustrating (and expensive) disaster, as we cannot supply replacement discs in the event of such an accident.

PCW versions:

Please note that we now distribute PCW versions of MFU on **High Density** 200K 3" discs, formatted using our **XFORMAT** program. These discs are perfectly compatible with normal CF2 discs (except for their greater capacity) in all respects bar one - Amstrad's **DISCKIT** copying program **CANNOT** copy them!

Consequently, you must copy the distribution disc EITHER using a **FILE** copying program such as **PIP.COM** and **NOT** a **DISC** copying program such as **DISCKIT**, OR by using XFORMAT's **C**opy disc option. Details of the latter method are given in the XFORMAT User Manual.

To copy the MFU disc file-by-file, put your CP/M disc containing the PIP program into Drive A and type

A>pip

Now put your MFU distribution disc into Drive A and type

***m:=a:mfu*.*[v]**

This will copy all the files off the distribution disc into the RAMdisc (Drive M:). Note that Drive M should be **empty** before you start this operation or you may run out of room!

Once you have done this, put a blank, **FORMATTED** disc into Drive A and copy the files back onto it by typing

***a:=m:mfu*.*[v]**

Note that the **[v]** option tells PIP to verify each file after it has been copied to ensure none has been corrupted.

What to do if you run out of memory!

If you only have a 112K RAM-disc, you will **not** have enough space in it to hold **all** of the MFU files at once and the copying procedure we described above will abort with the message

ERROR: DISK WRITE NO DATA BLOCK - <filename>.$$$

when all of the space in Drive M has been filled.

If this happens to you, **don't panic!** Firstly, copy the files which **are** in Drive M out onto a blank 3" disc in Drive A as described above.

Once those files are all safely copied out, note down their names. Check against the list of files given above to determine which have not been copied.

Type the command

A>era m:*.*

followed by "**y**" when CP/M asks for confirmation. This will wipe Drive M clean again. Now type

A>pip

again and put your distribution disc back into Drive A.

You will now have to copy each of the files you **didn't** get the first time from Drive A into Drive M **individually** by giving a series of commands like

***m:=a:<filename>.<ext>[v]**

where **<filename>.<ext>** is the name of each file to be copied in turn.

This will get all the extra files into Drive M. Now put your copy disc into Drive A and again type

***a:=m:*.*[v]**

as before, to copy the remaining files out of Drive M onto the copy disc. Once this is done, press <RETURN> to get out of PIP and type

A>dir a:

to check that **all** the files are now present on your copy disc.

CPC versions :

If you have a version of MFU for a CPC464/664/6128, the procedure is much more straightforward! Just use Amstrad's **DISCKIT3** program to make a backup of the whole disc.

2.3 Customising MFU

Version 4 of MFU is pre-configured with several default options. These include parameters describing the type of disc drive supplied with MFU (or that with which it is expected to normally be used), and the speed tolerance of that drive as a percentage as well as terminal and printer control strings, and the designation of your RAMdisc, if you have one.

Some of these values can be altered temporarily from within MFU with the **S)** option; if you wish to alter them permanently (if you later buy a different drive, perhaps), then this can be done by patching MFU.COM itself using MFUPATCH.COM.

Full information on customising your copy of MFU is given in Appendix A.

3. USING MFU.

To run MFU, the computer must already be running CP/M. If you are still in AMSDOS on a CPC, or Locoscript on a PCW, then put your CP/M boot disc into Drive A and type **CPM** (on a CPC) or press the **SHIFT**, **EXTRA** and **EXIT** keys together to reset the computer (on a PCW). This will start CP/M running. When it is ready to receive a command, you will see the prompt

A>

To execute MFU, make sure you have all the files which make up the MFU system on the logged drive, then type **mfu** at the CP/M prompt.

You can also execute MFU from a SUBMIT file if you wish.

The following sections provide information on restrictions on the ways you can use MFU, as well as suggestions to make its use faster and easier.

3.1 Drives and Files

If you have a PCW8256, we suggest that you run MFU from the RAMdisc rather than from Drive A as this will speed up its operation greatly. All further discussion will assume that MFU and all its files are present in Drive M, and that M: is logged as the default drive (i.e. your CP/M prompt is **M>**).

CPC6128 owners **must** run MFU from Drive A unless they have a third-party RAMdisc upgrade - operation will be slower, but otherwise identical. Whichever drive contains MFU, it **must** be made the default drive before MFU is run or the program will fail to find its overlays. This can be altered by MFUPATCH; see Appendix B for details of this.

If you are running MFU from Drive A, **NEVER** remove the disc containing the program and its overlays from the drive while MFU is in use unless prompted to do so. Doing this will result in a non-recoverable error and MFU will abort.

NEVER attempt to run MFU from a disc in Drive B unless you are sure you know what you are doing; if you do this then change the format assigned to Drive B, MFU will not be able to find its library files on disc and will be unable to function fully.

If you are using a 40/80 track switchable disc drive, **always** keep it set to **80 track** mode, even if you are using a 40 track format. MFU will alter the BIOS to automatically compensate for the different track pitch.

The **E)**dit and **A)**nalyse functions both make use of the cursor keys to provide on-screen editing of parameters and data. For this to function properly, MFU sets the cursor keys up to supply WordStar-compatible codes.

If you want to restore CCP cursor keys on exit from MFU, we suggest you run MFU from a SUBmit file such as the following :

```
mfu
setkeys keys.ccp
```

We also suggest that you load the MFU system into the RAM-disc by including the necessary commands in your PROFILE.SUB file which the computer executes automatically when CP/M is started up.

The following sequence of commands will load all necessary files from a 3" disc in Drive A: into RAM-disc, change the logged drive to M: and execute MFU:

```
pip m:=a:mfu.*
m:
mfu
```

Add these to your PROFILE.SUB file using RPED (or your favourite text editor) if you want CP/M to execute the sequence for you automatically on boot.

Please note that the whole MFU system requires 140K of disc space; an unexpanded PCW8256 only has 112K available in Drive M and so on such a machine you will have to run the program from Drive A, with a corresponding reduction in speed of operation - or buy a RAM upgrade!

However, if you want to cut down on the space taken up by MFU on your working discs, the following list of files explains the function of each. If you are not intending to use one or more of the functions, the corresponding files need not be present.

MFU.COM	-	Main program; **must** be present.
MFU.000 to **MFU.004**	-	Main overlay files; **must** be present.
MFU.005	-	Only required by the **P)** option.
MFU.006, **MFU.007**	-	Only required by the **N)** option; used for CP/M <-> MS/PC-DOS file transfer.
MFU.008, **MFU.009**	-	Only required by the **N)** option; used for CP/M <-> BBC file transfer.
MFU.DAT	-	Default library data file; must be present if you want to use formats stored in it or for MFU to automatically display the format in use.

When MFU is first run (and at frequent points within the program) an attempt is made to identify the Drive B: format currently in place by referencing the Library file. For this facility to be of use, MFU.DAT **must** be on the logged drive. If the current format doesn't match any in the library then the message

Drive B: is currently set to an unknown format

is displayed at the bottom of the screen. Otherwise, the format is named.

Please note that the displayed format is that which is currently set up **in memory** for Drive B, and need **not** bear any relation to the **actual** format of a disc you may have put into Drive B. If you put a different type of disc into Drive B, you must also load the appropriate format from the library to allow MFU and CP/M to understand it!

If you don't know the format of disc, you can use the **A)**nalyse option from MFU's main menu. Section 6.1 explains how this option works.

3.2 Memory Restrictions

MFU reserves **almost all** of the TPA (from **&100** to **&EFFF** inclusive, to be exact) for its own use. This means that if you have any RSXs loaded **below** &F000, MFU may overwrite them and the BDOS will probably crash when any RSX-using program is run. MFU will attempt to determine the amount of memory available using standard legal BDOS calls, and will abort if the apparent TPA top is below &F000.

4. STRUCTURE OF A DISC

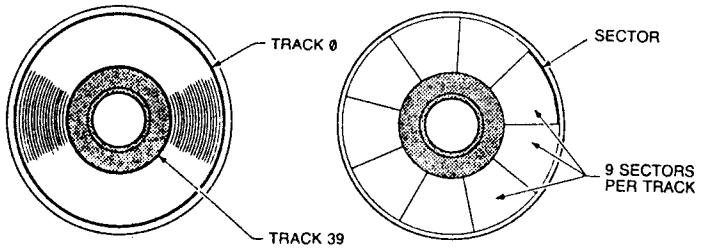
In this section, we will explain the basic facts behind the use of discs to store data on computers, and look at the restrictions physical disc structure puts on our format definitions. If you are already familiar with these concepts, you may wish to miss out this section. A much more detailed description of Physical Track formatting is given in Appendix B.

4.1 Basics

A floppy disc is simply a disc of plastic, coated in the same sort of magnetic material as an audio or video tape. Before a computer can use a disc, it must be **formatted** - this process writes a preset data pattern all over the disc, dividing it up into concentric circles, called **tracks**, within which our files will be stored.

This is what is meant when a disc is described as "40 track", or "80 track" - this simply describes the number of tracks of information which the disc can hold. Similarly, an "80 track" disc drive is a drive unit which is calibrated so as to be able to position its read/write heads over 80 concentric tracks. It is more accurate technically to use the term **96 tpi** (Tracks Per Inch) when referring to "80 track" drives and disc formats, as some drives and formats do not use exactly 80 tracks on the disc! Similarly, "40 track" drives and formats are better referred to as **48 tpi.**

All tracks on a disc are the same width and are spaced the same distance apart, so a disc drive can read or write any track on the disc by **stepping** its heads in or out across the disc by the appropriate amount.



Within each track, the formatting operation writes a rather complex pattern involving a header block to identify the start of the track followed by a number of (empty) **sectors.** These are simply a way of dividing the storage space on a disc into convenient little blocks, which the computer can keep track of. Each sector on a track also has a "hidden" block of information attached to it, which contains things like the sector's unique ID number and a checksum to verify any data which is stored in it.

4.2 Track Pitch

As you may have realised, the tracks on a 48 tpi disc are spaced approximately twice as far apart as the tracks on an 96 tpi disc. This is not normally a problem if you have an 80 track disc drive; MFU can automatically "double step" the heads to compensate for the higher track spacing when using a 40 track disc. However, there is a theoretical problem inherent in using an 80 track drive to write data onto 40 track discs.

This is due to the fact that not only are the tracks on 48 tpi discs spaced further apart, but they are also **wider** than on 96tpi discs. In general, the read/write head on a 48tpi drive is larger than its 96tpi equivalent; this is largely just because they tend to be older and less advanced technologically.

This can have an effect when a 96tpi head writes data onto a track which **already** has 48tpi data recorded on it. It is possible that while the 96tpi drive will happily read its own data back again, the wider head on the 48tpi drive will **also** read some of the surrounding remnants of the old, wider data track with the result that the data read becomes garbled or corrupt.

While this must always be borne in mind when using a 48tpi disc in both types of drive, in practice it is rarely a problem. We ourselves have in fact yet to find a 48tpi drive which **does** have problems reading such a disc! We regularly use MFU and a variety of 96tpi drives to transfer data to and from 48tpi formats such as Wren and PC/MS-DOS; we have never experienced the slightest problem.

4.3 Sector Identification

Each sector on a track has a number assigned to it - normally, the first sector on a track has number 0 or 1 (depending on the preference of the manufacturer of the computer the disc is used in), and the rest of the sectors generally number consecutively from this. Thus we can identify a sector uniquely by specifying which track it is on, and which sector number it is within that track.

For reasons of their own, Amstrad designed the disc formats used by the CPC range of computers to have a first sector number of 65 ($41) or 193 ($C1) on each track for system and data format discs respectively. Their CF2 and CF2DD formats for the PCWs adopt the more normal practice of starting their sector numbering at 1.

Even stranger systems, which the computer cannot naturally handle, are sometimes used by commercial software distributors to physically copy protect discs. Such techniques often involve wildly different sector numbering schemes on different tracks, extra tracks which CP/M cannot recognise, or sectors within particular tracks which have deliberately had peculiar faults inserted into them. Others format some tracks with sectors of different sizes. Such discs cannot then be copied by any normal CP/M copy program.

Similar problems can sometimes occur with discs formatted on other computers for use on the Amstrad. Sometimes the Amstrad's Disc Controller chip is not able to read the physical format of part or all of a track on a disc from another computer. Often, however, you will find that both computers **can** happily use discs formatted on the Amstrad using MFU. One good example of this is the Wren - discs used to transfer data between one of these and an Amstrad **must** be formatted on the Amstrad by MFU, **not** by the Wren.

Each sector is tagged by the controller with a **Data Address Mark (DAM)** to help it locate and verify the data the sector holds. Exceptionally, a **Deleted DAM (DDAM)** may be used instead. This need **not** mean that the data in the sector is part of a deleted file, although that was the original meaning and function of the DDAM. This dates from the prehistory of computing; nowadays, most computers use only normal DAMs.

If a sector with a DDAM **is** present on a disc, the Amstrad will normally ignore it and fail to read it. MFU's **A**)nalysis function **can** however detect, read and display a sector with a DDAM. If wished, it can then be written back out with a normal DAM and other functions and programs will then be able to read it.

4.4 Data Capacities

Obviously, the maximum amount of data which we can store on a disc is limited by the number of tracks it has, the number of sectors per track, and the size of each of those sectors. For example, each side of a standard CF2 3" format disc as used in Drive A of a PCW8256 normally has the following physical format :

40 tracks
9 sectors per track
512 bytes per sector

Thus such a disc can hold 40 * 9 * 512 = 180K (KiloBytes) of data. Note that this may be a slightly higher figure than the **actual** maximum free data space on a disc, for reasons which we will go into in the next section!

Other computers use many different physical formats - the number of tracks on a disc can be 35, 40, 77 or 80; the number of sectors per track varies from 4 to 26 or more, and the size of sectors can be 128, 256, 512, 1024 or 2048 bytes! This is one major reason for the apparent total incompatibility of discs from different computers - until MFU was written!

Other factors which affect the capacity of a disc are the **number of sides** which are used to store data (either one or two), and the **density** of the recording. Amstrad 3" CF2 discs as used in Drive A of the PCW8256 (and on the CPC464/664/6128) are rather unusual in that they are double sided, but only one side can be used at a time. This is because the drive which reads them only has a single read/write head. 5¼" discs are **not** like this - single sided discs **cannot** be "turned over" like a 3" disc, while double sided ones are used in drives with two read/write heads and so both sides can be accessed without the disc being removed.

The **density** of a disc is a description of the actual method used to record information on the magnetic coating. Early **Single Density** discs used the **Frequency Modulation (FM)** system, very similar to that used to record music on audio tapes. More modern **Double Density** discs use the **Modified FM (MFM)** system; as its name suggests, this technique packs information onto the disc at double the density of the older FM system, thus allowing twice as much data to be stored. All modern computers use Double Density discs; however, some older Single Density formats (such as Osborne and Acorn BBC) are still in fairly common use.

The PCW8256/8512 and CPC6128 **cannot** normally use Single Density formats due to hardware restrictions in Amstrad's Disc Controller circuitry. However, Appendix F details a simple series of hardware modifications to the PCW8256/8512's main circuit board which **will** then allow these computers to use both single and double density discs. Unfortunately, there are **no** equivalent simple modifications for the CPC6128.

Expanded CPC464 owners **can** use Single Density **without** any hardware modification.

We don't know whether CPC664s can handle Single Density or not, simply because we haven't been able to find one to test! If you own one of these rare machines, please contact us!

MFU is of course capable of setting up and using formats of **either** density.

4.5 Sector Skew

When a formatting program creates tracks and sectors on a disc initially, it may **not** number consecutive sectors within a track with consecutive numbers! For example, a disc with nine sectors, numbered 1 to 9, might actually have these written on each track in the order

1, 6, 2, 7, 3, 8, 4, 9, 5

It is important to realise that the **physical** order of sectors on a track has absolutely **no** effect on the order that data is read from the disc - it is **consecutively numbered** sectors which are read consecutively by CP/M, **not** consecutively **positioned** ones. So we can scramble the physical positions of sectors **without** scrambling the data they contain.

The reason for doing so is simple - it speeds up access to the disc. If you imagine CP/M (or a program) reading consecutively **numbered** sectors one after the other, then obviously each sector is read from the disc by the Disc Controller chip, transferred back to CP/M (which takes time) and then moved from CP/M's transfer buffer to its proper

destination in memory (which also takes time).

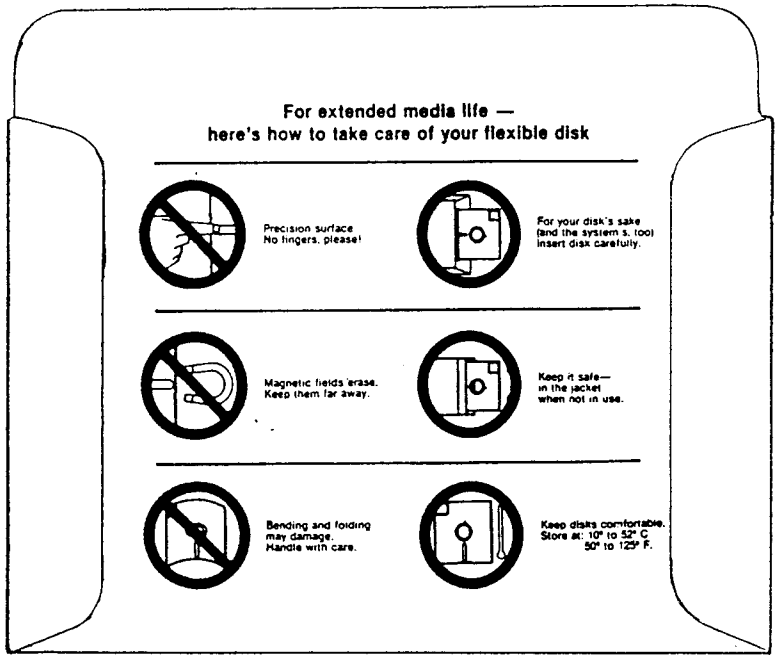
By the time CP/M asks the Disc Controller for the next sector, the next **contiguous** sector on disc will have slipped past the heads and (if it were also the next numerically), we would have to wait a full revolution of the disc (1/300th of a second, an age in computer terms!) for it to come round again. However, if we arrange the sectors physically in the order shown in the above example, then the sector which we have just missed is not the next one we want - instead, as if by magic, ours is the very next one to come along!

This system is known as **Physical Sector Skew**, and is implemented "once and for all" on any particular disc when that disc is formatted. If you want to change the **skew factor** (i.e. the number of sectors which physically come between two consecutively numbered ones) at a later date, you can only do this by reformatting the disc. Despite this restriction, physical skew is popular and widely used because it is simple to implement, effective and invisible to any operating system or program using the disc to read or write data. The alternative (and more complex) method of implementing skew - Logical Skew - is discussed in the next section.

Note that one of the features of physical skew is that different discs may be formatted with different skew factors and still used interchangeably - some will simply be more efficient (i.e. faster) in use; all will work perfectly.

4.6 Disc corruption and damage

Discs can be corrupted quite easily if handled wrongly, often in ignorance. Most disc sleeves have a list of "DOs" and "DON'Ts" printed on their backs; a typical example is shown below. You should make sure that these rules are observed by anyone handling your discs. Most of the ways you can damage a disc are physical - someone spills coffee over it, or it warps under the heat from the sun if you leave it beside a window.



It is also possible, however, to corrupt or even destroy the information stored on a disc without damaging the disc itself. This can happen if you put a disc too near a source of strong electromagnetic radiation or magnetic field, such as a television, electric motor, microwave oven or even an old telephone with a bell!

If a disc has been corrupted in this way, the only thing to do is attempt to recover as much information from it as you can (by copying any files which are undamaged to another disc) and then reformat the disc.

Appendix E details the way the Disc Controller reports the various types of disc errors, as well as offering suggestions as to the likely causes and possible remedies.

5. INTRODUCTION TO CP/M

This section explains a bit about the CP/M Plus operating system as used on the PCW8256/8512 and CPC464/664/6128, and describes the way it uses a disc to store information. Again, if you are familiar with this, you may wish to go on to the next section.

It is not possible to describe CP/M in any depth in a manual of this sort; the number of large books and long magazine series which have appeared on the subject over the past year are evidence of this! We can, however, explain the aspects which relate to disc formats.

CP/M is simply a very specialised program which is used to interact with, and control, the computer at the lowest level. When the computer is switched on, the files on disc which make up CP/M are read into memory and stored away. Although we don't know it, CP/M is **always** present in our computer's memory, though it may not always be active.

The first interaction we have with CP/M is when it displays the well-known

A>

prompt on our screens. When we see this, it means that CP/M is ready to receive a command. Broadly speaking, such a command can be one of two types - **resident** or **external.** A **resident** command is one which is "built in" to CP/M, such as **DIR. External** commands are simply the names of programs stored on disc - when CP/M gets such a command, it loads the appropriate program from disc into memory and **executes** it. This simply means that control of the computer is turned over to the program - CP/M is no longer in full charge.

When the program has finished, it in turn gives control back to CP/M so that we can enter another command. When a program **crashes**, this process does not occur, and we are left with a computer which will not listen to any commands we give it! In this situation CP/M has lost control completely, and our only course of action is to reset the computer to restart CP/M from scratch. Sometimes even this will not work and we have to turn power off to the machine to "kill off" a rogue program!

CP/M itself is split into two main parts - the **Basic Input/Output System** (**BIOS**) and the **Basic Disc Operating System** (**BDOS**). We never need to bother with this unless we are writing CP/M programs, but it will help us understand the use of discs if we explain the distinctions between the two parts.

The BIOS is responsible for directly controlling the computer's hardware. It is the BIOS that scans the keyboard for input, sends data to the CRT Controller chip for display on the screen, and communicates with the Floppy Disc Controller (FDC) chip to read and write the tracks and sectors of data on the discs.

The BDOS, on the other hand, is above all this! It is the BDOS which programs use to communicate with the outside world - a program simply asks the BDOS to read a file from the disc, for example; it is left to the BDOS to work out where on the disc the file is physically

stored. It then asks the BIOS to recover the file's data from disc by sending the FDC the appropriate commands.

The third part of CP/M - called the **Console Command Program**, or CCP, is the command interpreter which we actually talk to. The CCP displays the

A>

prompt on our screens, and is responsible for making sense of what we type in! If we give the CCP an **internal** command such as DIR, it will carry out the operation itself and display the results (a directory listing of files on our disc, in this case). To do this, it will call the BDOS interface routines in exactly the same way as any other program. In fact, you can consider the CCP to actually be a program just like any other; the only real difference is that it is always hidden away somewhere in the computer's memory, ready to run. It does **not** need to be loaded from disc each time it is needed, as other programs must be.

The CCP is an example of a type of program called a **shell.** The concept of shells is not used very much in CP/M, unlike other operating systems such as PC/MS-DOS and Unix, where it has a much greater significance. It **is** possible to replace the standard CCP with a shell which you have written yourself, if you have a need for unusual commands or facilities, but this requires an expert knowledge of CP/M, and is not for the timid!

Although the division of labour between BIOS, BDOS and CCP may seem needlessly complex at first sight, it is in fact this which makes CP/M as useful and widespread as it is! The BDOS provides a standard interface for programs to use to perform all their input and output (whether it be to screen, printer, discs, keyboard or any other device attached to the computer). This interface is the **same** on **all** CP/M computers, of whatever make - thus a program which runs on one CP/M computer should run with little or no alteration on any other!

The BDOS in turn has a standard interface to any individual computer's BIOS; all a manufacturer has to do is write the code for the BIOS part of CP/M, which has to know about each computer's hardware design. Once this has been done, the rest of CP/M and any properly-written programs will run on that computer.

The CCP, in its turn, has the function of providing a completely standard "User Interface" - if you know how to use CP/M on one type of computer, you will feel equally at home on any other!

5.1 File Structure

CP/M (in common with all other computer operating systems) has the function of imposing and maintaining a **file structure** on a disc which physically is simply a vast and uncharted ocean of data! We (and our programs) can then store our data on a disc by putting it into **files** which each have unique names and so can be recalled at any time. Each file can be regarded as a high-tech pigeon hole; we can stuff any information we want to keep together (such as all the text in a letter, or all the lines of a program) into a single pigeon hole which we can then find again and look at whenever we want in the future.

To do this, CP/M lumps sectors together into **groups**, and allocates storage on the disc to these groups. It identifies data groups by number, and can calculate the track and sector position of the start of any group by using some of the values which are held in the **Disc Parameter Block** (DPB).

The DPB is a block of data maintained by the CP/M BIOS for use both by the rest of CP/M and by programs. It is not necessary to go into more detail about it here; Appendix D contains a full description of all the Amstrad's disc parameters.

CP/M stores the numbers of groups allocated to a particular **file** in a **directory** which it maintains at the start of the disc. A file is simply a collection of data groups which we have asked CP/M to keep track of for us when we created it - for example, it could be a program or a wordprocessed letter; CP/M neither knows its content nor cares. The directory is simply an indexed list of files stored on the disc which allows us to recover them and store others.

CP/M's group numbers are stored and used **entirely** internally to itself - we never need to know about them other than to understand the way information is stored on disc. Each directory entry contains the name we have given a file, information as to whether it has any special attributes such as Read-Only or System, and (most importantly) a list of group numbers which, in order, hold our file. One point to note is that the groups holding a file do **not** need to be contiguous - as CP/M hold a record of **every** group used by a file, it can allocate space which happens to be available anywhere on the disc.

It may be implied from the above that each file on a disc has a directory entry, and therefore we can store as many files on the disc as we have directory entries available. This is unfortunately not the case! Although their names are listed only once by a "DIR" command, many files have more than one entry in the directory; large files may have several. Again, we need never normally worry about this, but should be aware that it occurs in order to understand any "Directory Full" error message we may get from CP/M on a disc which may only have half the theoretical maximum number of files stored on it. The way CP/M allocates directory entries to files is complex, and we'll leave a fuller description to Appendices C and D.

5.2 Logical Disc Structure

CP/M discs may also have **Reserved Tracks**. These are tracks on the disc which are not available to us for data storage, and so discs with reserved tracks have a lower capacity than similar discs without. Reserved tracks are used to store the programs which make up part or all of CP/M itself; this code is read into memory from the disc and executed automatically when CP/M is booted.

In a banked-memory CP/M Plus system such as is implemented on Amstrads, only the disc which is booted from at the start of day need have reserved tracks; however to avoid confusion Amstrad decided that **all** discs used on PCWs would have one reserved track. CF2 and CF2DD discs therefore have a slightly lower capacity than is strictly necessary. On CPCs, SYSTEM format discs have a reserved track and thus a slightly lower capacity than DATA format discs.

Amstrad have made use of some of the wasted space on CF2 and CF2DD format discs to implement their "smart" auto-format detection system which gives PCW8512 owners a kind of limited multi-format ability - their otherwise incompatible Drive B can read but not write discs from Drive A!

The directory, group size and file structure that CP/M imposes on a disc is known as that disc's **Logical Format**. It is important to distinguish this form the disc's **Physical Format** - the logical format controls how and where files are stored and recovered, but is wholly a function of software; it does not (and cannot) affect the way data is actually recorded on the disc in physical tracks and sectors.

Nowhere does this cause more confusion than in the two quite different ways CP/M uses the word SECTOR! Early CP/M computers used discs whose physical sectors were 128 bytes in size, and so CP/M itself was designed to use 128 byte sectors in its interface with programs. However, times moved on and CP/M didn't!

Computers such as our Amstrads have much higher-capacity discs than the earlier types and so generally use physical sectors which are much larger than 128 bytes (512 bytes per sector, in our case). However, CP/M programs still "expect" to read and write 128 byte sectors when they call the BDOS Disc Service Routines.

To avoid confusion, we will always refer to one of CP/M's **logical**, 128 byte "sectors" as a **RECORD**, and leave the term **SECTOR** to refer only to the **physical** sector size used by a particular disc format.