BBC MASTER AIV USER GUIDE

WARNING: THIS COMPUTER MUST BE EARTHED

Important

The wires in the mains lead for the computer are coloured in accordance with the following code:

Green and yellow	**Earth**
Blue	**Neutral**
Brown	**Live**

For United Kingdom users

The moulded plug must be used with the fuse and fuse carrier firmly in place. The fuse carrier is of the same basic colour (though not necessarily the same shade of that colour) as the coloured insert in the base of the plug. Different manufacturers' plugs and fuse carriers are not interchangeable. In the event of loss of the fuse carrier, the moulded plug MUST NOT be used. Either replace the moulded plug with another conventional plug wired as described below, or obtain a replacement fuse carrier from an Acorn Computers' authorised dealer. In the event of the fuse blowing it should be replaced, after clearing any faults, with a 3 amp fuse that is ASTA approved to BS1362.

For all users

If the socket outlet available is not suitable for the plug supplied, either a different lead should be obtained or the plug should be cut off and the appropriate plug fitted and wired as noted below. The moulded plug which was cut off must be disposed of as it would be a potential shock hazard if it were to be plugged in with the cut-off end of the mains cord exposed.

As the colours of the wires may not correspond with the coloured markings identifying the terminals in your plug, proceed as follows:

The wire which is coloured green and yellow must be connected to the terminal in the plug which is marked by the letter E, or by the safety earth symbol ⏚ or coloured green, or green and yellow.

The wire which is coloured blue must be connected to the terminal which is marked with the letter N, or coloured black.

The wire which is coloured brown must be connected to the terminal which is marked with the letter L, or coloured red.

Exposure

The computer should not be exposed to direct sunlight or moisture for long periods.

Ventilation

Do not block the ventilation slots in the case of any units supplied.

Within this publication the term 'BBC' is used as an abbreviation for 'British Broadcasting Corporation'.

The product described in this manual and products for use with it, are subject to continuous development and improvement. All information of a technical nature and particulars of the product and its use (including the information and particulars in this manual) are given by Acorn Computers in good faith. However, it is acknowledged that there may be errors or omissions in this manual. A list of details of any amendments or revisions to this manual can be obtained upon request from Acorn Computers Technical Enquiries. Acorn Computers welcome comments and suggestions relating to the product and this manual.

All correspondence should be addressed to:

Technical Enquiries
Acorn Computers Limited
Cambridge Technopark
Newmarket Road
CAMBRIDGE CB5 8PD

All maintenance and service on the product must be carried out by Acorn Computers' authorised dealers. Acorn Computers can accept no liability whatsoever for any loss or damage caused by service or maintenance by unauthorised personnel. This manual is intended only to assist the reader in the use of this product, and therefore Acorn Computers shall not be liable for any loss or damage whatsoever arising from the use of any information or particulars in, or any error or omission in, this manual, or any incorrect use of the product.

Written and computer typeset by Interaction Systems Limited, Cambridge

First published 1986
Issue 1 – November 1986
Published by Acorn Computers Limited

The BBC Master AIV

USER GUIDE

Part number 0461,001
Issue 1
October 1986

Contents

1. Introduction

The purpose of this guide is to describe the BBC Master Series Advanced Interactive Video (AIV) Microcomputer and its place in the BBC Advanced Interactive Video System. If you are using the equipment in conjunction with the Domesday videodiscs, you will also want to refer to the documentation supplied with them. This guide can be read as a supplement to the Master Series 'Welcome Guide' which is supplied with the computer in the AIV System package; an understanding of the operation of the Master Series computer is only necessary if you wish to use the AIV System to produce interactive video programmes of your own.

An Interactive Video system integrates the features of computers and video devices such as videodisc players to allow the creation of so-called delivery systems which play sequences of video material under the control of the computer, usually as a result of commands from the user. Applications for delivery systems include training aids, product marketing, information services and a host of others; the Domesday Project is one such application.

The BBC Master AIV is a sophisticated controller for Interactive Video applications, which may be programmed in a variety of programming languages (frequently BASIC), and contains most of the hardware and software necessary to provide a user-friendly environment for videodisc-based delivery systems; all that needs to be added to the package for it to be ready for use is a suitable videodisc player, discs and a display monitor. One example of such a package is the delivery system for the BBC's Domesday Project, in which the Master Series AIV Microcomputer is a major component.

This manual describes how to set up and operate the AIV System, and contains detailed information concerning the support for applications provided by the built-in software; the videodisc player and monitor are described in the documentation supplied with them.

Within this manual several typographical conventions are used to make the text easier to read:

- Text in ruled boxes, eg [SHIFT], denotes the single key on the keyboard with the corresponding legend, for example [SHIFT] means the SHIFT key.
- Where a number of keys need to be pressed simultaneously to have the desired effect, these are grouped together with the symbol '+', so [CTRL]+Q+[BREAK] means 'press the CTRL key, the letter Q and the BREAK key'. They should be released in the reverse order.

- `Text like this` is used for examples of commands, programs and computer output that you might see on the screen.
- **Bold** and *italic* text are used for emphasis.

2. Unpacking and set-up

The BBC Master Series AIV Microcomputer components are supplied in three boxes, which are themselves packed into one large carton. If you have the complete BBC AIV System you will also find a box containing the videodisc player, another containing the monitor and a fourth containing the Domesday videodiscs themselves.

The BBC Master Series AIV System package should contain the following.

In the first box:

- one copy of this guide
- one copy of the '65C102 Co-processor User Guide'
- one 65C102 co-processor support disc
- one RGB (Video) lead

In the second box:

- one trackerball

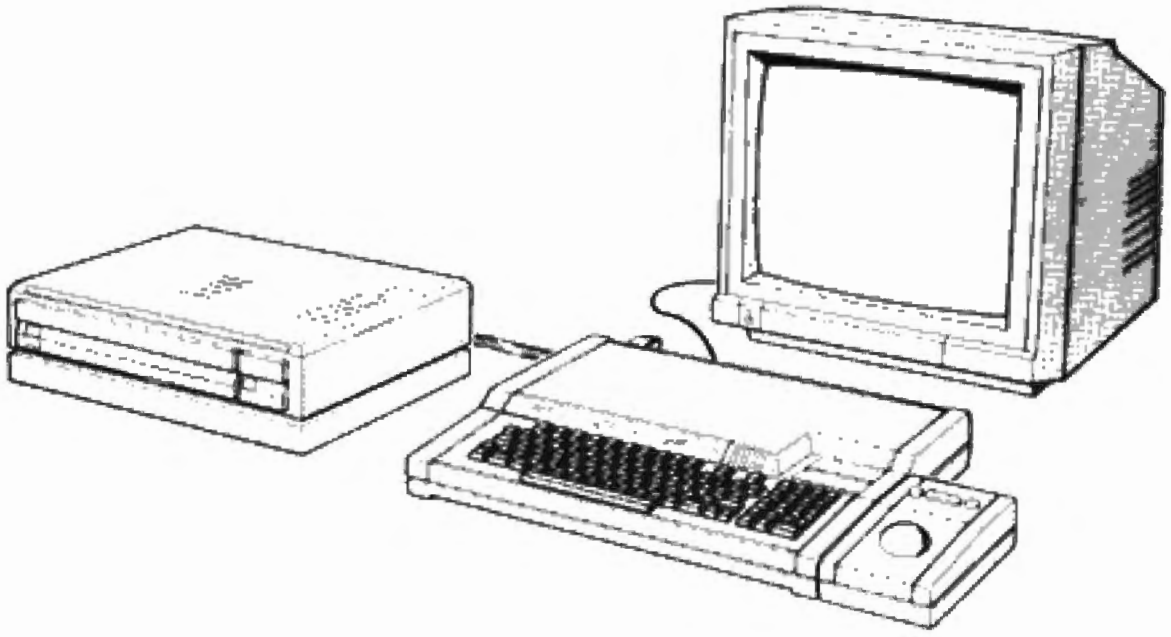
In the third box:

- one Master AIV computer

The computer has already been fitted with the Turbo co-processor card, a Small Computer Systems Interface (SCSI) and the Video Filing System (VFS) ROM chip. You should verify that all of the components listed above are present and contact your supplier as soon as possible if any item appears to be missing.

In order to set up the system you will need a good-sized horizontal surface near to three mains power outlets – it is best to allow an entire desk for the equipment to avoid having to move it around. The videodisc player in particular is quite heavy, so it's worth finding a stable base for it to protect it from vibration, etc. Do not put the monitor on top of the videodisc player as this obstructs the ventilation slots in the case.

Unpack everything and put the discs and manuals to one side, then arrange the equipment so that the keyboard is readily accessible and the monitor comfortably positioned for viewing from the keyboard position. A suggested arrangement is shown in the diagram on the next page:



Connecting the AIV System equipment

Once you have made the equipment ready as detailed above, you need to interconnect the various components. For this stage you will require the following cables:

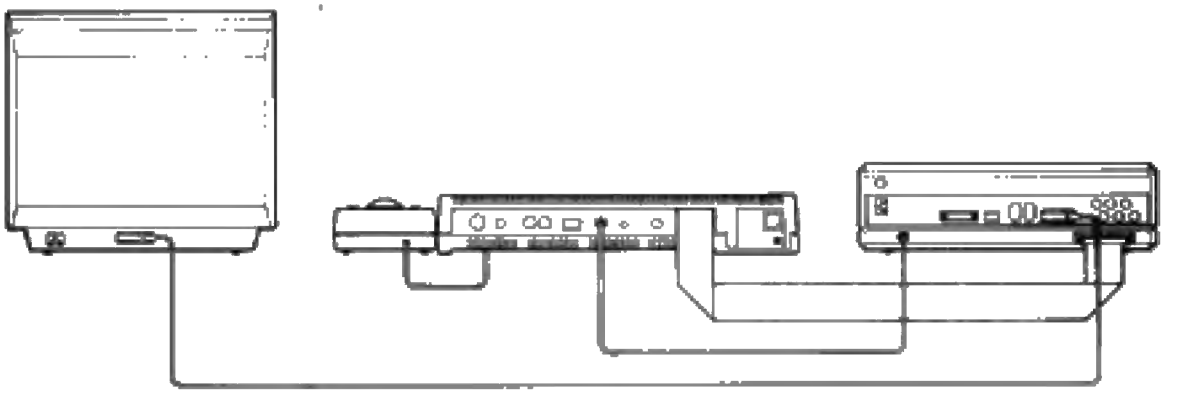
- RGB cable (supplied in the first box)
- SCSI cable (at the rear of the computer)
- Euroconnector cable (supplied with the videodisc player)
- trackerball cable (at the rear of the trackerball)

Additionally, you will need to identify the three mains power leads, one each from the computer, player and monitor. The computer's mains cable is captive at the computer end and has a mains plug fitted to it; the player and monitor may either have similar captive cables or free cables with a mains plug at one end and an IEC connector (which is roughly rectangular) at the other – consult the documentation supplied for further information. *Do not plug in the mains power leads at this stage.*

Connect the cables as indicated:

Cable	From	To
SCSI	(Fixed to computer)	Player SCSI socket at rear
RGB	Computer RGB socket at rear	Player RGB socket at rear
Trackerball	(Fixed to trackerball)	User Port socket (front underside of computer)
Euroconnector	Player Euroconnector socket at rear	Monitor Euroconnector socket at rear

When all of the cables are plugged in, connect the three mains power leads into mains outlets (and turn them on if the outlets are switched). The system is now ready to be activated.



Switching on

Begin by turning on all of the equipment. It is suggested, but by no means vital, that the monitor be turned on first, then the player and then the computer – this allows the monitor to warm up and the player to get the disc spinning at speed, but you may do it in any order you wish.

If you have been supplied with the Domesday videodiscs (they look like silver LP records), then insert the 'Community disc' into the player as follows. Ensure the videodisc player is on and press the 'Eject' button – the disc tray should slide open, allowing you to rest the disc in the depression in the tray. Now gently push the tray home into the player.

The final operation required to get the system running is to tell the computer that you want to start using the Video Filing System (VFS). Once the equipment is on, you should hold down [SHIFT]+Q+[CTRL]+[BREAK] (all at the same time), then release the keys in the order [BREAK], [CTRL], Q then [SHIFT] (this

sequence may be modified, see the appendix). If you have a disc in the videodisc player then it will automatically start playing, the screen will initially display:

```
Acorn TUBE 65C102 co-processor
Acorn VFS
```

and pictures will appear after 45 seconds. If you have no disc, then the screen will show an additional error message.

Once the disc has begun playing, you may press [ESCAPE] to begin using the information retrieval software which is encoded onto the disc. You will, therefore, need to refer to the documentation supplied for further information.

Fault-finding

The first (and most obvious) thing to check if your AIV System appears not to work is the mains power connections: ensure that the Master AIV Microcomputer, monitor and videodisc player all have their mains leads plugged into sockets (at both ends where appropriate), and that the sockets and the devices themselves are switched on.

Now go through the cabling chart above again. Make sure that each of the leads is connected to the appropriate socket and is firmly pushed home into the socket so that it cannot fall out.

Finally, check the symptoms you are experiencing against the list below:

There is no picture on the monitor at all.

- Is the monitor switched on?
- Is the videodisc player switched on?
- Is the Euroconnector cable properly connected between player and monitor?
- Is the computer displaying anything? Try restarting the computer by pressing [CTRL]+Q+[BREAK].
- Are the Brightness and/or Contrast controls set too low for the picture to be visible?

The player plays the videodisc but the computer seems not to work.

- Have you waited 15 seconds after resetting the system?
- Is the computer switched on? Does the 'Power On' indicator on the keyboard glow?
- Press [CTRL]+Q+[BREAK] on the keyboard. Unplug and plug the RGB cable. Does the computer's display appear?

– Turn off the player, unplug and re-plug the SCSI cable, then turn on the player again. Press [CTRL]+Q+[BREAK] on the keyboard. Does the computer's display appear?

– Check the computer status as described in the appendix.

Now try restarting the computer as described in the previous section.

Try using the remote control for the player; this should help you discover whether the player or the computer is at fault.

The trackerball does not seem to work.

– Is the trackerball plugged into the User Port socket? Is the plug pushed firmly home into the socket?

– Press [CTRL]+[BREAK] on the computer. Type in:

```
MODE 1[RETURN]
*TRACKERBALL[RETURN]
*POINTER[RETURN]
CLS[RETURN]
```

Now roll the ball on the trackerball – if an arrow-shaped pointer appears on the screen and moves about as you roll the ball, then the device works correctly. If no pointer appears, the trackerball or its cable is probably faulty.

Use without LaserVision player

The computer is also usable as a Master 128 without the LaserVision player. Simply switch the player off and connect the computer directly to your display device. (Some monitors may come ready equipped with a cable to allow you to do this. If one was not supplied consult your Acorn dealer.)

Refer to the Master Series 'Welcome Guide' for information on using the Master as a standalone computer.

3. System introduction

The BBC AIV System is a package of computer hardware and software which provides a basis upon which sophisticated Interactive Video (IV) systems may be built. In the jargon of Interactive Video technology it is a **delivery system**, and it is the generality of its design which makes it so well suited to this kind of work. The purpose of a delivery system is to retrieve information in any of several forms and display it to the user under software control – in this guide we use two alternative spellings to distinguish between the **program** which controls a computer and the video **programme** which is displayed by a delivery system.

Whether you are intending to use the AIV System for an application yet to be developed or to make use of one that already exists, it is useful to have some understanding of the way in which the components of the system interrelate. This chapter is devoted to a discussion of the workings of the AIV System, and it is written for readers who presently have little or no understanding of IV systems.

The videodisc

The videodisc is the database of an Interactive Video system because it contains all of the source material with which the system operates. A videodisc can store moving images (in the form of television-style video pictures), still pictures, speech, music and information which may be interpreted by computers as programs or data. This variety of media which the videodisc can store is fundamental to the range of uses to which IV may be put.

The videodisc itself may be single or double-sided with each side being made from a plastic disc which is pressed in a similar way to a vinyl record, coated in a reflective material and then encased between protective plastic sheets. The information stored on the disc is encoded in the form of millions of tiny pits in the reflective surface which are produced during the pressing process; because the disc is reflective, a spot of light directed at its surface will be reflected back whenever the spot hits a reflective part, and not reflected when it hits a pit. Electronic circuitry in the player reconstitutes these changes in reflected light into digital information which can be processed by a computer.

The BBC AIV System employs the LaserVision type of videodisc upon which both the video pictures and the computer information on the disc are stored as a number of concentric **tracks**. The laser mechanism which reads information from the disc can move in a straight line between the centre and the outside

edge and, as it does so, it passes many times over the tracks. Each track stores a still video image, called a **frame**, 25 of which go to make up one second of moving video picture. Each frame has a unique number, allowing individual frames to be selected by moving the reading mechanism to the appropriate part of the disc. There are 54,000 frames on each side of the disc, providing up to 36 minutes of moving video storage, as well as up to 324 million bytes of computer data stored on the audio tracks at six kilobytes per frame. To provide this data storage capacity the videodisc is played at Constant Angular Velocity (CAV) by the player.

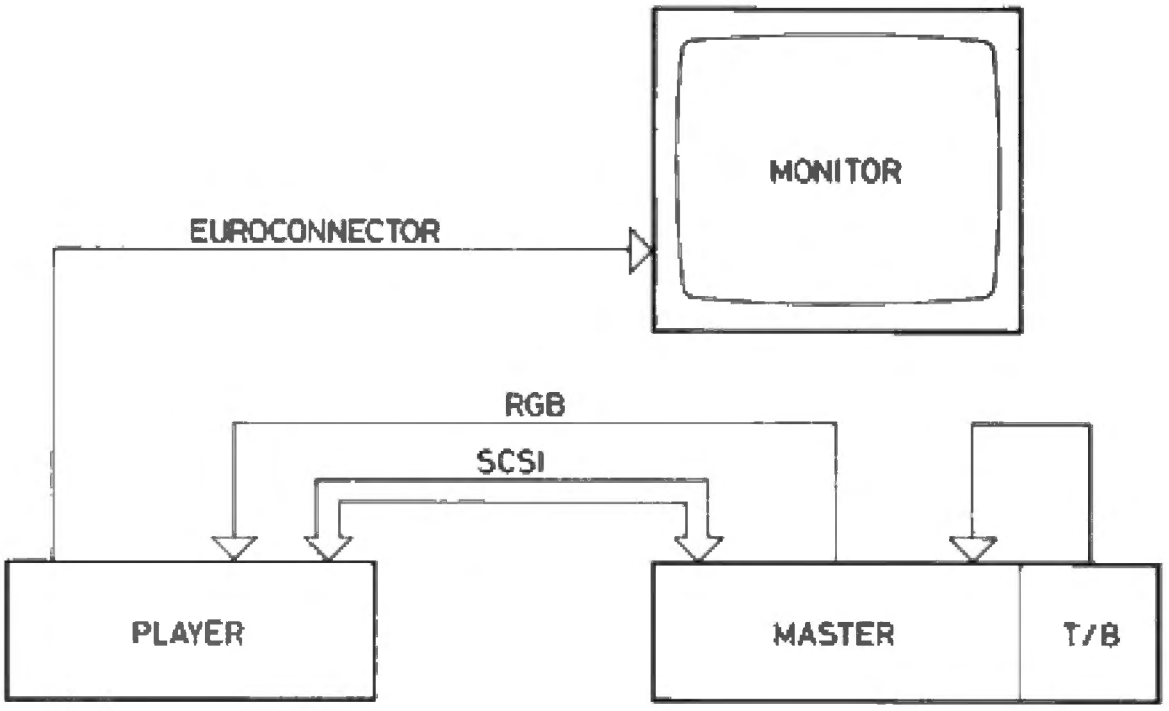
The videodisc player

The videodisc player is the mechanism which makes it possible for the computer to read information from the disc. It spins the disc at 1500 revolutions per minute and coordinates the movement of the **laser head** which actually detects the pits in the disc. The computer can send commands to the player asking it to search for a particular frame, at which point it may freeze the frame on the screen or start playing the disc as if it were normal audio-visual footage.

A second function of the videodisc player is to decode information on the disc which is intended to be used as computer data. This is read from the disc in the usual way and then passed back to the computer. The data is stored using a technique known as LV-ROM which is similar to the CD-ROM data storage system for compact discs. LV-ROM is a recent innovation, but the LV-ROM videodiscs may be played in any kind of videodisc player with the LV-ROM data being ignored by players not equipped to decode it. It is worth noting that LV-ROM data is stored on both of the audio tracks on the videodisc and therefore data will be heard as sound unless the sound output from the player is disabled.

Finally, the player also performs a number of video functions including synchronisation of the computer's picture, known as **genlocking**, and mixing of the videodisc picture and computer picture together to produce one final image. This feature is covered in more detail later in this guide.

The functions of the player are summarised in the diagram in section 5.14.



The computer

The role of the computer in IV systems is to control the progress of the programme according to the user's commands and responses. The computer reads programs from the disc, executes them, and then determines its subsequent actions on the basis of its instructions and the behaviour of the user. Users can employ either the keyboard or trackerball to interact with the system, which responds by selecting video footage, stills, soundtrack and special effects to be displayed on the monitor.

The BBC Master Series AIV microcomputer contains an SCSI circuit board which allows it to control the videodisc player and read data from the videodisc. This interface may also be used to control other SCSI-compatible devices, and its operation does not interfere with the use of other peripherals which are connected to the computer, so allowing more complex Interactive Video applications involving other devices such as floppy disc drives.

The AIV System can be programmed in a number of computer programming languages, offering a range of styles to suit most authors' needs. The VFS provides all of the commands needed to control the player, video mixer and trackerball and read programs and data from the disc so that the programme creator does not need to become involved in the details of the hardware itself.

4. The Domesday Project

The Domesday Project is one of the most comprehensive IV applications yet created; it is also the first to employ the technique of storing both source material (ie video, stills, data, etc) and the control programs together on the same videodisc – in the past the controlling computer has executed programs from its own, quite separate, storage medium.

The BBC, who initiated the Domesday Project, wished to compile a modern analogue of the 900-year-old Domesday Book which comprises the first, and best known, demographic survey of England. They realised that the material that they wished to include would come in a great diversity of forms – everything from tables of government statistics to satellite photographs. This posed a serious problem in terms of how to disseminate the information given the mixture of media, and it became clear that Interactive Video offered an excellent solution. By further including the control software itself on the videodisc, the problem was reduced to the collection and collation of the information, and the production of equipment suitable for its retrieval under the control of an untrained operator.

The Domesday discs

The Domesday Project information is stored on two LaserVision discs – the National disc and the Community disc. The National disc primarily contains text in the form of abstracts and essays, along with statistical data and other information on the country as a whole. The contents of the Community disc are split into two sides, each side covering roughly half of the country (the sides are 'North' and 'South'). Both discs combine still and moving video material with soundtrack or digital data in LV-ROM format; a computer-controlled videodisc player is used in conjunction with computer software read from the videodisc to effect the display of this information.

Retrieval software

The information on the National disc is indexed by **keyword**, allowing the user to choose a general topic from the four main classifications and refine it by choosing progressively more specific keywords. Once the relevant data has been found it may be displayed on the screen directly or, in the case of statistical data, in any of a number of user-selected graph and chart styles, utilising the computer's built-in graphics facilities.

The Community disc is primarily indexed by geographical location – a sizeable part of the database consists of Ordnance Survey maps. Users can either point to an area on the main UK map and enter a numerical grid reference or enter the name of the location they wish to examine (in which case the software endeavours to find the name given in a 250,000 name gazetteer). Furthermore, a keyword facility similar to that on the National disc is also available for searching by topic.

Once a location has been chosen, it and its surrounding area are initially displayed on the screen in the form of a large area map. The user can then point more specifically at a place of interest and the computer will enlarge the map scale where possible. A list of the written texts available for the location and a number of photos may also be displayed. The user may then select an item and have it displayed on the screen.

Conclusion

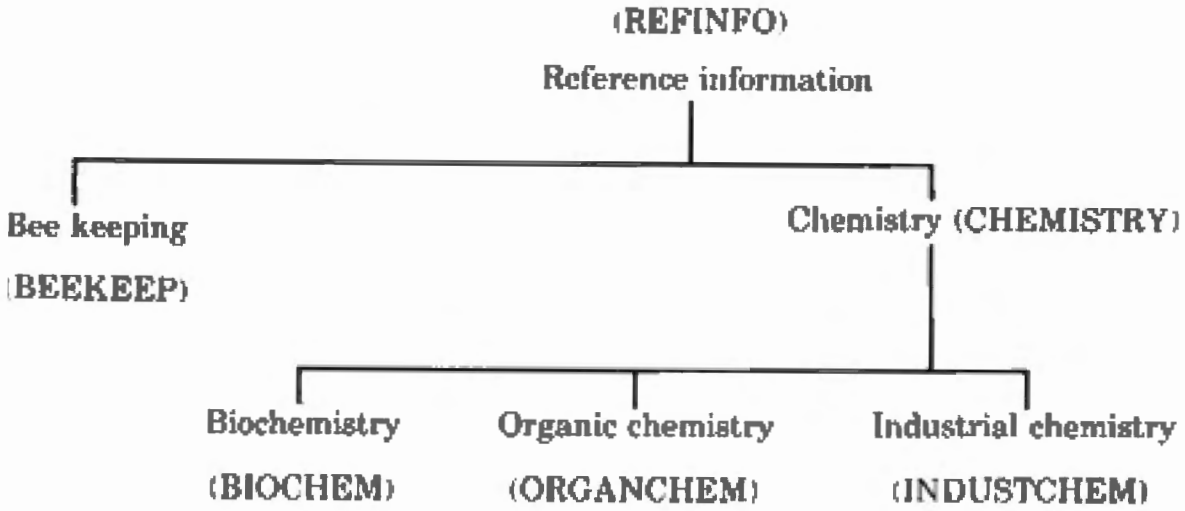
The Domesday Project is a complex information-retrieval system which has been implemented using readily-available equipment. The use of a videodisc player which is under computer control has enabled the designers of the system to offer the user a simple and yet versatile mechanism for displaying information which is located deep in the storage system – the delivery system does not expect or require the user to have any prior knowledge of computers.

The remainder of the material in this guide is of a technical nature and is provided for use by programmers creating specialised applications for the BBC Master Series AIV System. Readers who wish only to use the Domesday Project videodiscs need read no further – separate documentation is supplied with the videodiscs covering the operation of the retrieval software, and such readers are referred there.

5. The Video Filing System (VFS)

The VFS is a 16K byte program stored in a Read Only Memory (ROM) chip inside the BBC Microcomputer. The filing system controls the reading of information from the videodisc player and provides a number of useful facilities for maintaining that information. Because this filing system is based on the Acorn Advanced Disc Filing System (ADFS), readers may find it useful to refer to the Master 'Welcome Guide' or 'Reference Manual Part One' for supplementary information.

An important feature of the Video Filing System is that it is **hierarchical**, allowing information to be organised in tiers. At the bottom of the hierarchy is a file. Immediately above it is a collection of files, known as a **directory**. Directories enable files to be collected together into logical groups. Putting files into directories can be likened to a library, where books are put on shelves, each shelf or group of shelves having a name. Books about bee-keeping could probably all go on one shelf, whereas books on chemistry might need several shelves – one for industrial chemistry, say, one for biochemistry, one for organic chemistry, and so on (see below).



Suppose that this same information is stored, not in books in a library, but in files on a disc; the VFS allows us to organise information in the same way. The information files are collected together in directories, whose names (eg BIOCHEM) are shown in brackets. Notice that, just as in the library example the books about chemistry are on different shelves in the 'master' chemistry section, the information on the disc about chemistry is stored in different directories within the 'master' chemistry directory. Directory CHEMISTRY contains, not the information itself, but the addresses of the other directories (BIOCHEM, ORGANCHEM, INDUSTCHEM) and these contain the information. This illustrates a key feature of the hierarchical filing system provided by VFS: directories can contain not just files but other directories, which themselves can contain other directories and so on.

Since the system described above can result in directories containing files or other directories, or both, files and directories are collectively known as **objects**. A directory (assuming it is not empty) always contains objects, whether files and/or directories. The directory containing all the highest level objects is called the **root directory**.

5.1 Pathnames and object specifications

To refer to a file called (for instance) Memo1, it may not be enough to type, for example:

```
LOAD "Memo1"[RETURN]
```

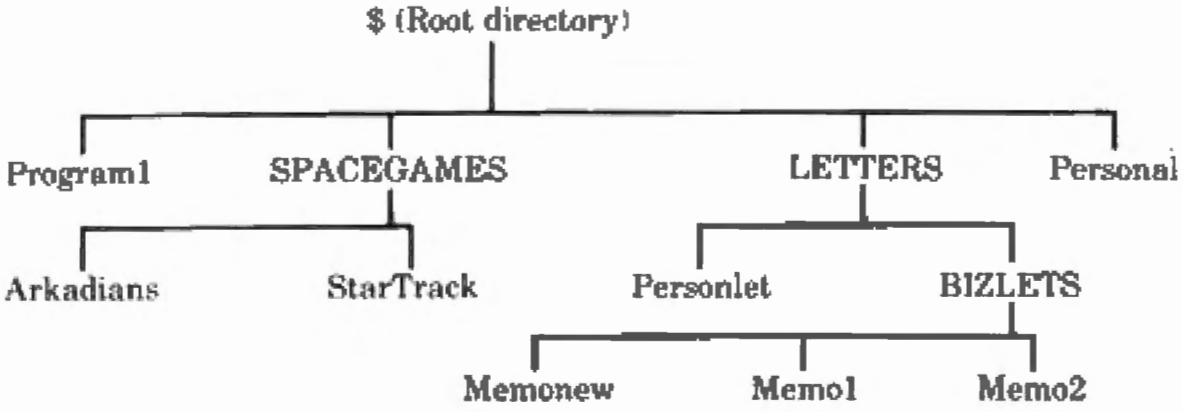
since Memo1 may be inside a directory, which may itself be in another directory and so on. The full specification for an object is called a **pathname**. From the root directory, the pathname for file Memo1 at the bottom right of the figure below is:

```
LETTERS.BIZLETS.Memo1
```

So, to load Memo1, type:

```
LOAD "LETTERS.BIZLETS.Memo1"[RETURN]
```

(Note that each part of the pathname is separated by a dot.) The above pathname is also called an **object specification**, since it defines how to find the object in question.



5.2 Directories

A directory is a collection of up to 47 objects. It may be part of another directory and may itself contain other directories. A directory name can be up to 10 characters long. Although it can contain any characters except:

`# * . : $ & @` and `^`

for the purpose of compatibility with other filing systems, it is suggested that only the following subset of characters is used:

`! ' ( ) 0-9 < = > ? A-Z _ £ a-z`

(This also applies to filenames.) The full stop character '.' is reserved for use in pathnames; the others have special uses which are explained later.

Different directories may contain identical filenames; although A.B.MYPROG and A.B1.MYPROG have the same filename, they are different files because they are in different directories.

Note: Although a directory may be regarded as 'containing' its constituent objects, in reality it merely contains a list of disc addresses (plus other information) relating to its objects.

The 'master' directory containing all the other directories (and perhaps files as well) is called the root directory. The $ (or &) symbol is used to refer to the root directory in pathnames. The root directory is located whenever the disc filing system is first entered. At this point it is known as the **Currently Selected Directory (CSD)**. Referring to the figure below, if the computer is asked to list all objects in the CSD (ie $), the list would consist of:

Program1
SPACEGAMES
LETTERS
Personal

(SPACEGAMES and LETTERS are directories, Program1 and Personal are files.) If LETTERS were to be made the CSD, and the computer asked to list all the objects in the CSD, the list would be:

Personlet
BIZLETS

(Here Personlet is a file, BIZLETS is a directory.) To refer to an object in the root directory from another directory, the object's pathname must begin with $; if already 'in' the root directory (ie if the root directory is the CSD), the $ can be omitted (see below for examples).

5.3 Object referencing

This section illustrates how to refer to objects within the VFS hierarchy. It should be read in conjunction with the figure on the previous page, and it is assumed that the VFS has just been entered, ie the root directory (directory $) is the currently selected directory (CSD). Notice that in the figure, directory names are shown all in upper case to distinguish them from filenames; this is for the purposes of illustration only; the VFS permits upper and lower case letters to be mixed in filenames.The file Program1 would be loaded by typing:

```
LOAD "Program1"[RETURN]
```

The file StarTrack would be loaded by typing:

```
LOAD "SPACEGAMES.StarTrack"[RETURN]
```

To get information about file Memo1, one might type:

```
*INFO LETTERS.BIZLETS.Memo1[RETURN]
```

(The *INFO command is detailed in section 5.11.)

Now suppose that by using the *DIR command (see section 5.11), the directory LETTERS is selected as the CSD. Information about file Memo1 would now be obtained with:

```
*INFO BIZLETS.Memo1[RETURN]
```

and for information about the file Personlet, only:

```
*INFO Personlet[RETURN]
```

would be necessary. However, to load file StarTrack one must type:

```
LOAD "$.SPACEGAMES.StarTrack"[RETURN]
```

since StarTrack is not in the CSD. Attempting to use:

```
LOAD "SPACEGAMES.StarTrack"[RETURN]
```

would result in the error message:

```
Not found
```

being displayed. The computer would be looking for a directory called SPACEGAMES within the directory LETTERS, but the directory SPACEGAMES exists only in the root directory. The above instructions would be correct if the root directory were to be reselected as the CSD.

5.4 Special characters

The circumflex character '^' represents the **parent directory**, ie the directory of which the CSD is a member. For example, if BIZLETS is the CSD, one might type:

```
*INFO ^.Personlet[RETURN]
```

instead of:

```
*INFO $.LETTERS.Personlet[RETURN]
```

Similarly, the '@' aymbol represents the CSD. It should be used in order to direct a command specifically to operate on the CSD.

5.5 The library directory

The VFS allows one directory to be specified as the **library**. This directory can contain frequently used utility files to which rapid access is needed. Machine code utility programs may be executed simply by typing their name as a '* command', for example:

```
*(utility name)[RETURN]
```

which is equivalent to typing:

```
*RUN (utility name)[RETURN]
```

In both cases the directory containing the utility is assumed to be the currently selected directory or the library. The filing system searches the CSD for the file and then, if it does not find it there, it searches the library. For example, if the library contains a machine code program MC1, then typing:

```
*MC1[RETURN]
```

causes the program to run regardless of whether or not the library is the CSD and whether or not (for systems with more than one drive) the library is on the currently selected drive.

Before the VFS library facility may be used, the library directory must be specified. The default library is set up as follows:

- If VFS is entered with CTRL+BREAK (or CTRL+Q+BREAK) and there is a directory in $ whose name begins with LIB, that directory is allocated as the library.
- If VFS is entered as above, but no directory exists with name beginning $.LIB, $ (ie the root directory) is allocated as the library.
- If a drive is accessed by the *MOUNT command, the library is not allocated at all (it is said to be 'unset') and has to be set using the *LIB filing system command.

The library directory may be altered by the *LIB command. It remains in force until the computer is switched off or reset by pressing CTRL+BREAK (or CTRL+Q+BREAK), or reset by another *LIB command.

5.6 Wildcard facilities

The VFS provides a means of abbreviating long object names and referring to several objects at once; this is the **wildcard** facility. Wildcards may only be used with certain commands, and in the reference section later in this chapter the filing system commands which can operate with wildcards are followed by the abbreviation <*obspec*> (meaning **wildcard object specification**) instead of <obspec> (meaning **object specification**).

As an example, consider *CAT which provides information about the contents of a named directory. Assuming the root directory is the currently selected directory, typing:

```
*CAT LETTERS.BIZLETS RETURN
```

will display information about the directory named BIZLETS in the directory named LETTERS.

To save typing LETTERS.BIZLETS in full, the wildcard characters * and # may be used in the object specification. The * character is understood by the VFS to mean 'substitute any number of characters up to ten', whilst # is a substitute for just one. The VFS checks the wildcard pathname and chooses the first object in alphabetical order which fits the bill. In the example hierarchy, LETTERS is the only directory in the root beginning with L, so typing:

```
*CAT L*.BIZLETS RETURN
```

could be used instead of the previous command. In fact, since BIZLETS is the only directory in LETTERS:

```
*CAT L*.* RETURN
```

would also work. Suppose catalogue information is required of a directory in LETTERS, but it is not known whether it is called BIZLETS or BIZLETZ, the # character can be used as part of an object specification, for example:

```
*CAT LETTERS.BIZLET#[RETURN]
```

5.7 Multi-object operations

Some filing system commands can operate on a number of objects instead of just one; in the programmer's guide (Chapter 6), these commands are all followed by the abbreviation <listspec> (short for list specification). *INFO, which provides information about a named object, is an example of such a command. If LETTERS is the CSD, then:

```
*INFO BIZLETS[RETURN]
```

will display information about the object named BIZLETS. A list specification may contain the same wildcard characters as the object specification discussed above, but with wider application. For example:

```
*INFO BIZLETS.*[RETURN]
```

would display information about all three files in BIZLETS. If information about just Memo1 and Memo2 is wanted, one might type:

```
*INFO BIZLETS.Memo#[RETURN]
```

A slightly quicker way is to type:

```
*INFO BIZLETS.#####[RETURN]
```

since ##### in this context means 'all names with five characters', or:

```
*INFO B*.*O#[RETURN]
```

where *O# means 'all names with O as the penultimate letter' (which is a little convoluted). A quick way of displaying information about file Memonew would be to type:

```
*INFO BIZLETS.*w*[RETURN]
```

since *w* in this context means 'all names with a "w" in them'.

Auto-boot facilities

Sometimes it is useful to load, run or execute a program or a file automatically when entering the filing system. This is achieved using a file named !BOOT. !BOOT is a special filename located by the filing system when [BREAK] and [SHIFT] are pressed simultaneously. If a file exists with the pathname:

```
$.!BOOT
```

the filing system will act as defined by the option set by the *OPT 4 (n) command (see below).

5.8 Resetting the system

The [BREAK] key resets the BBC Microcomputer. However, the VFS can preserve some of its status after [BREAK]. There are two types of [BREAK]:

- a 'hard [BREAK]' (sometimes called a 'cold boot') achieved by holding down the [CTRL] key and pressing [BREAK]; and
- a 'soft [BREAK]' (sometimes called a 'warm boot'), achieved by pressing the [BREAK] key.

If a hard [BREAK] does not give the VFS start-up message on the screen, then another filing system is the default filing system. VFS can be entered by pressing [BREAK] while holding down both [CTRL] and Q. (Alternatively, VFS can be 'warm booted' simply by pressing [BREAK] while holding down Q.) The *CONFIGURE command can be used to specify VFS as the default system.

The *CONFIGURE and *STATUS commands can take the parameters VFSDIR and VFSNODIR. These are the VFS equivalents of ADFS's DIR and NODIR, but use a different CMOS RAM location – which one depends on the number of the slot occupied by the VFS ROM; it should, therefore, always be put in the same slot.

As far as the VFS is concerned, hard [BREAK] is the same as switching the computer off and then on again. The currently selected directory is set to $, the library is set as previously described, and any open files are forgotten.

When performing a soft [BREAK], the VFS preserves its status, ie the CSD and the library remain the same, and open files remain open.

5.9 VFS commands

This section summarises the filing system commands, ie those words which the filing system program recognises and acts on. They can be typed directly onto the keyboard or embedded within a BASIC program. They are all prefixed with the * character which signals to the computer that a filing system (or operating system) command follows, and they must be followed by [RETURN].

The syntax abbreviations used are:

<obspec> = object specification
<*obspec*> = wildcard object specification
<listspec> = list specification
<drv> = drive number

Items appearing in brackets are optional.

5.10 Attributes

Since, unlike other Acorn filing systems peripherals, a LaserVision disc cannot be written to, the attributes of objects stored on it cannot be altered. However, when creating a disc, the files on it can be assigned attributes. These control access to the objects on the disc and distinguish files from directories. The attributes supported by the VFS are:

E – Execute only. This attribute is used to protect files containing machine code programs. If the E attribute is set, the file cannot be *LOADed, all OSFILE calls are prevented, and display of object information by the *EX and *INFO commands is prevented. The only commands which affect a file with the E attribute set are:

*RUN <filename> and *<filename>

R – Read access. This must be set for reading (including loading) to be allowed. Attribute R applies to files only; it has no meaning in relation to a directory.

D – Directory. This is set if the object is a directory.

If the R attribute is not set for a file, it cannot be read, and attempts to use the *LOAD command result in the error message:

```
Access violation
```

as would an attempt to use the OPENIN BASIC keyword (or the OSFIND assembly language call to open a file for reading).

Note: For ADFS compatibility, the full set of file attributes is supported. Although the locked bit is irrelevant for VFS, it should be set by any pre-mastering software, so that ADFS will not attempt to write to the file. VFS does not write to any files, regardless of attributes.

5.11 Command summary

*VFS

Purpose: To enter the VFS from another filing system.

*BACK

Purpose: To go back to the Previously Selected Directory (PSD). The directory selected before the last *DIR or *BACK command becomes current, and the PSD is set to the old CSD. Thus repeated *BACKs may be used to swap between two frequently used directories.

Examples:

*DIR A	Select A as CSD
*DIR B	Select B as CSD, A is PSD

```
*BACK       A is CSD again, B is PSD
*BACK       B is now CSD, A is PSD
```

Description: Makes the previously selected directory the currently selected directory.

Associated command: *DIR

*CAT (<*obspec*>)

Purpose: To display a 'catalogue' of the specified directory. This consists of directory 'header' information (see below) and a list of the objects in the directory. If no directory can be found which matches the <*obspec*>, an error occurs. If no <*obspec*> is supplied, the currently selected directory is catalogued. Objects in the directory are listed in alphabetical order with their attributes and sequence numbers.

Example:

```
*CAT BusLet
Business Letters      (13)
Drive:0               Option 00 (Off)
Dir.BusLet            Lib.Library1

File1      R (08)   File2     R (09)   Glenn   R (00)   XDir    D (05)
Z-test-4   R (12)   Z-test-5  E (13)
```

Here the title of the directory is Business Letters. (Note: a directory title is distinct from a directory name – see section 5.12.)

Description: Displays the catalogue of a directory.

Associated commands: *DIR, *EX, *INFO

Note: The sequence numbers referred to above have been maintained for compatibility with ADFS. Although not adjustable, they are still useful for comparing discs that are ostensibly the same.

*CLOSE

Purpose: To close all sequential access files. *CLOSE is the same as the CLOSE#0 BASIC keyword.

Example:

```
*CLOSE
```

Description: Closes all sequential access files.

Associated command: BASIC's CLOSE#0

*DIR (<*obspec*>)

Purpose: To make a named directory the Currently Selected Directory (CSD). The object specified must be a directory. If no <*obspec*> is supplied, the root directory of the current drive is selected. When the system is first started, or after a hard reset, the CSD is the root directory of drive 0. The command is also used to change or reset the currently selected drive.

Examples:

```
*DIR Dir1
```

selects Dir1 as the CSD.

```
*DIR
```

selects the root of the current drive as the CSD.

```
*DIR *
```

selects the first directory in the CSD as the new CSD.

```
*DIR :0
```

changes the currently selected directory to the root of drive 0.

```
*DIR :0.Work
```

changes the currently selected drive to drive 0 (with Work in the root directory of drive 0 as the CSD).

```
*DIR^
```

selects the parent of the currently selected directory (ie the directory of which the CSD is a member) as the new CSD.

*DISMOUNT (<drv>)

Purpose: To unload the current LaserVision disc under software control.

Example:

```
*DISMOUNT
```

unloads the disc in the currently selected drive (ie puts the player into 'standby' mode).

Notes: *DISMOUNT only closes the files on the drive specified by <drv> (or the currently selected drive if <drv> is absent).

The *DISMOUNT command does not force a disc ejection. That is initiated by the video command *EJECT (see section 5.16).

If the CSD is on a *DISMOUNTed drive, the system 'unsets' the CSD and the

library. The next access to that drive will either produce the 'No directory' error message, or $ will be read from the drive and treated as the CSD, according to the nature of the access. If the LaserVision player is the currently selected drive, typing:

```
*DISMOUNT
*CAT
```

would produce (say):

```
$                    (19)
Drive:1              Option 00 (Off)
Dir. "Unset"         Lib. "Unset"

LIBPROG     DLR(22)  PROGX        WR (08)  PROGY        WR (19)
```

The CSD is not set, but is treated as being $ since $ is the default directory in the *CAT command. However, typing:

```
*DISMOUNT
LOAD "PROGX"
```

would produce the error message:

```
No directory
```

The CSD and the library can be set using the *DIR and *LIB commands.

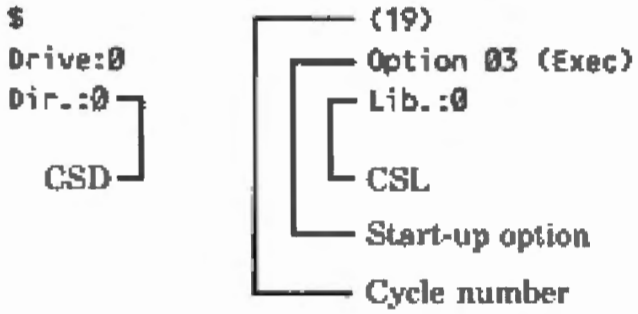
*EX (<*obspec*>)

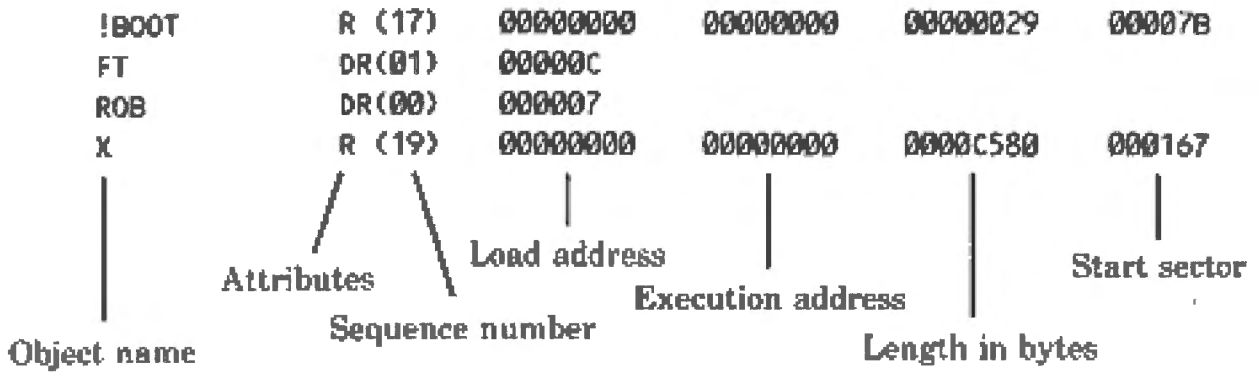
Purpose: To display information about the directory named in the <*obspec*>. It includes details not given by *CAT such as the length of a file and its location (both in hexadecimal). The information is displayed in the following order across the screen:

Example:

```
*EX $
```

might give:





(If no <*obspec*> is given, the currently selected directory is examined.) Note that if the object is a directory, only the start sector number is displayed (the other quantities have no meaning for a directory). For a file with the E attribute set, *EX will only display the attribute string and the generation number. For example, if file X had the E attribute, *EX would only give:

```
X                E (19)
```

Description: Displays directory contents information.

Associated commands: *CAT, *DIR, *INFO

*EXEC (<*obspec*>)

Purpose: This command reads, byte by byte, all the information in the specified file as if it were being typed in at the keyboard. Instead of typing in the same sequence of commands repeatedly, an EXEC file (a text file) can be created containing these commands, though not (of course) on a videodisc. Typing *EXEC <*obspec*> activates the sequence of commands.

Example:

```
*EXEC HELLO
```

reads the contents of file HELLO one character at a time as if it were being entered at the keyboard and acts upon any commands generated.

Description: Executes commands in a file as if typed in at the keyboard.

Notes: A useful application of this command is *EXEC !BOOT in conjunction with option 3 (described later in this section). If the file !BOOT contains the text CHAIN "<*obspec*>" where the object is a BASIC program, pressing BREAK while holding down SHIFT causes the program to load and run automatically.

If a file's execution address is &FFFFFFFF, typing:

*RUN <filename> (or just *<filename>)

will *EXEC the file rather than 'load and run' it.

*EXEC without a filename, when included in an EXEC file, will stop the file from executing. For example, an EXEC file could include the line:

```
IF <condition> THEN *EXEC
```

*HELP <keyword>

Purpose: Displays useful information. For the Videodisc Filing System the <keyword> is 'VFS'. The information displayed is a list of the filing system commands. If just *HELP is typed, the system produces a list of currently installed ROMs.

Examples:

```
*HELP

Videodisc FS 1.70
VFS, Video, Mouse, Trackerball

OS 3.20

*HELP VFS
```

displays a list of the VFS commands.

Likewise, *HELP VIDEO, *HELP TRACKERBALL, *HELP MOUSE all display information (including a command list).

Notes: *RUN, *EXEC, *OPT, *CAT and *LOAD are not included in the HELP information because they are Machine Operating System (MOS) commands which function outside the VFS. *HELP is a machine operating system command.

*INFO <listspec>

Purpose: Displays information about a list of objects. The information consists of the object name, attribute string and generation number, load address, execution address, length and disc sector address, and is the same as displayed by *EX.

Examples:

```
*INFO TEST*
```

displays information about all objects in the CSD with names beginning with TEST.

```
*INFO ADD1.*
```

displays information about all objects in directory ADD1 (identical to *EX ADD1).

Description: Displays detailed information about a set of objects.

Associated commands: *CAT, *DIR, *EX

*LCAT

Purpose: Catalogues the current library, as in *CAT.

Example:

```
*LCAT

$                    (96)
Drive:1              Option 03 (Exec)
Dir.WD1              Lib.$

!BOOT  R (29)   CINEMAS  R (96)  UTILS   DL (00)  WD1   DL (02)
WD2    DL (03)
```

Here the current library is the root directory, and the currently selected directory is WD1.

Associated command: *CAT

*LEX

Purpose: Examines the current library, as in *EX.

Example:

```
*LEX

$                    (96)
Drive:0              Option 03 (Exec)
Dir.$                Lib.$

!BOOT        R  (29)  00000000  00000000  00000029  000007
CINEMAS      R  (96)  00000000  FFFFFFFF  0000001E  000008
UTILS        DL (00)  000009
WD1          DL (02)  000062
WD2          DL (03)  000067
```

Here the current library and the currently selected directory are both $.

Associated commands: *EX, *LIB

*LIB (<*obspec*>)

Purpose: Sets the library to the specified drive and directory.

Example:

```
*LIB $.A
```

sets directory A in the root as the library. Typing after this:

*<filename>

causes a search of directory A for the named file; if found, the file is loaded and executed as would happen in response to:

*RUN A.<filename>

(Note that the library does not have to be the CSD.) In this example, directory A would not be retained as the library following a hard break or a *MOUNT command (see below for more details).

Description: Defines the directory that is to contain the library.

Associated commands: *DIR, *LCAT, *LEX, *RUN

Notes: When VFS is entered, the library directory is set to $, unless a directory exists with name beginning $.LIB, in which case the latter is allocated as the library. (If there were two or more such directories, they would be ordered alphabetically, with the first allocated as the library.) A directory will not be retained as the library following a hard break from VFS unless its name begins $.LIB. A *MOUNT operation does not adjust the library, unless it is on the current drive, where it will change it to 'unset'.

The library is usually used for files which contain machine code programs, such as games or utility programs.

*LOAD <*obspec*> (<address>)

Purpose: Reads a named file from the disc into computer memory starting at either a specified start address or the file's own load address.

Examples:

*LOAD "JOULES"

reads the file JOULES into memory starting at the load address of the file when it was saved.

*LOAD JOULES 3200

reads the file JOULES into memory starting at location 3200 (hexadecimal).

Description: Loads a file into memory.

Associated command: *RUN

Notes: The quotation marks shown above are optional (they are not treated as part of the filename). If they are used, they must be used in pairs. If the named file is not found, the error message:

```
Not found
```

is displayed.

*LVFS

Purpose: This command starts the VFS without accessing the disc (ie no disc information is loaded into RAM). The system is started with the CSD and the library 'unset', as with *DISMOUNT.

Example:

```
*LVFS
*CAT

$                    (19)
Drive:1              Option 00 (Off)
Dir."Unset"          Lib."Unset"

LIBPROG     DL (22)  PROGX        R (08)  PROGY          R (19)
```

Associated commands: *VFS, *DISMOUNT

Notes: Pressing [BREAK] while holding [CTRL] and L down together has the same effect as the *LVFS command except that it also sends a 'Start unit' command to the player. ADFS also uses this convention, but with the F instead of the L key.

*MOUNT (<drv>)

Purpose: The command initialises a drive (by forcing a 'hard reset' of the circuits within the player), starts the LaserVision disc, reads the free space map and stores the appropriate addresses in RAM. It is good practice to *MOUNT a drive after a disc error has occurred.

If *MOUNT is issued without a parameter, the current drive is assumed. The parameter can be a digit in the range from 0 to 7 or a letter from A to H in upper or lower case. The drive referenced by the parameter is mounted.

Example:

*MOUNT 0 or *MOUNT A

initialises the LaserVision drive.

Description: Initialises a drive.

Associated command: *DISMOUNT

Notes: Typing:

```
*DIR :<drv>
```

produces almost the same effect as typing:

```
*MOUNT <drv>
```

The only differences are that *MOUNT defines the library to be 'unset', even if there is a directory on the reselected drive with a name beginning with LIB, and that *MOUNT sends a 'Start unit' command to the player.

*OPT 1 (n)

Purpose: This command enables or disables a message system which displays a file's information (the same as *INFO). Every time a file on the disc is accessed the information is displayed. (n) takes the value 0 for disable or any number in the range from 1 to 99 to enable the feature.

Examples:

```
*OPT 1 1 or * OPT 1,1
```

enables messages;

```
*OPT 1 0 or *OPT 1,0
```

disables messages.

Description: Message system to display file information at every access.

Associated command: `*INFO`

Notes: A space or a comma between *OPT 1 and its argument (n) is essential.

*OPT 0 resets the *OPT 1 state to its default; ie *OPT 0 has the same effect as *OPT 1,0.

Options

Because the LaserVision disc system is a read-only medium, the auto-boot options recorded on the disc cannot be altered by VFS. In other Acorn filing systems *OPT 4 (n) changes these options. This section is intended to allow the interpretation of the auto-boot options which are recorded on LaserVision discs. They may be inspected by, for example, *CAT.

The option determines the action taken by the computer when [SHIFT] and [BREAK] are pressed. This action is on a file called !BOOT, which must be in the root directory, and is as follows:

0 does nothing
1 *LOADs file !BOOT automatically

2 *RUNs file !BOOT automatically
3 *EXECs file !BOOT automatically

If option 0 is set, the !BOOT file need not be present. With the other options the message:

```
Not found
```

is produced if !BOOT is not found after a [SHIFT]+[BREAK].

*RUN <*obspec*> (<optional parameters>)

Purpose: This command is used to run machine code programs. It loads a file into memory and jumps to its execution address, unless that address is &FFFFFFFF, in which case the file is *EXECed as a text file.

Example:

```
*RUN PROG
```

will cause a machine code program in the file PROG to be loaded and executed starting at the execution address of the file.

Description: Load and run a machine code file, or *EXEC the file if its execution address is &FFFFFFFF.

Associated commands: *LIB, *LOAD

Notes: This command will not run a BASIC program.

Typing *<*obspec*> or */<*obspec*> is interpreted as *RUN <obspec> (clearly, the object in this case must be a file).

Typing *<filename> results in the file being loaded and executed if it is in the currently selected directory or the library.

The <optional parameters> referred to above are those which are optional to the machine code program whose filename is in the command.

If a file's load address is &FFFFFFFF, *RUN <filename> produces the error message:

```
No!
```

5.12 Titles

Once again, because the LaserVision disc system is a read-only medium, the *TITLE command supported by other Acorn filing systems is not supported by VFS. This section, however, specifies the form of the directory titles which may be inspected using *CAT, etc.

The title may be up to 19 characters long. Note that a directory title is distinct from a directory name. The title has no meaning to the computer; it is only used to enable the user to give a directory a 'readable' identity.

5.13 MOS write commands

Commands built into the MOS, which expect to be able to write to the disc (such as *SAVE and *DELETE), produce the error message:

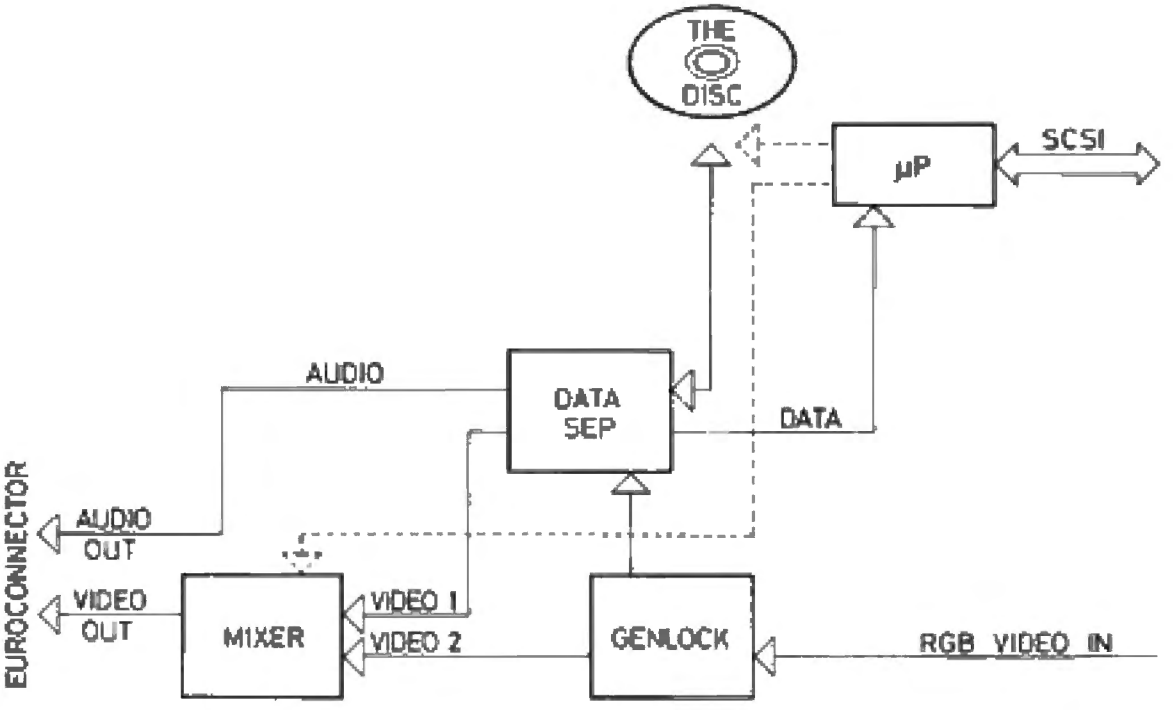
```
Disc read only
```

Commands built into the ADFS such as *ACCESS are not supported by the VFS; their use results in the error message:

```
Bad command
```

5.14 Videodisc player internal operation

The videodisc player used with the BBC AIV System is an enhancement of a standard videodisc player, and it has been designed specifically for advanced Interactive Video use. Below is an internal block diagram of the player circuitry:

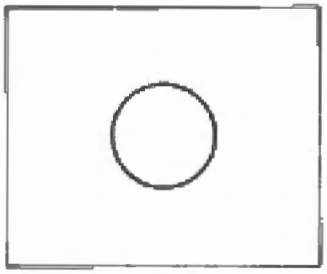


The player is connected to the computer by two cables: the RGB video output from the computer and the SCSI cable. Referring to the diagram, we can see that the RGB video signals from the computer are used to synchronise or 'genlock' the player's picture to that of the computer. This synchronisation is necessary before the video mixing stage can take place; commands from the computer determine which of the five video mixing modes is used, and the mixed signal is then passed out of the player to the monitor through the SCART connector.

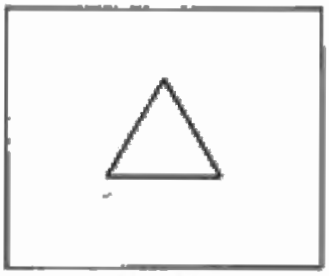
The SCSI allows the computer to take control of the player by passing it commands known as **F-codes**. Upon receipt of an F-code the player's microprocessor decodes the command and performs the appropriate operation. Data recovered from the disc is decoded into two streams: the video data is passed directly into the mixing circuitry for display on the monitor, whilst the audio data is treated according to the current mode of operation. Because computer data is encoded in LV-ROM format in one of the audio data tracks, the audio output is usually disabled and the data decoded and fed to the computer. However, when sound is selected by the appropriate F-code, the player passes it out to the SCART connector for amplification in the monitor.

5.15 Video mixing modes

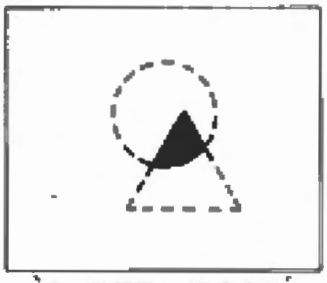
The video mixer built into the player has five modes of operation. These provide a set of useful ways of selecting and overlaying the images from one source and/or the other. The first two modes are fairly self-explanatory: *VOCOMPUTER displays only the image generated by the computer, whilst *VODISC displays only the image that the player is retrieving from the disc. *VOSUPERIMPOSE displays the image generated by the computer on top of that produced by the player by blanking out that part of the player's picture which is occupied by the computer's output (ie non-black). *VOTRANSPARENT simply mixes the two signals together, producing an effect rather like laying two sheets of tracing paper on top of one another. Finally, *VOHIGHLIGHT dims down the player's picture *except* where the computer's image is on (ie non-black), producing a 'brightup' effect. These modes are summarised in the diagram on the next page.



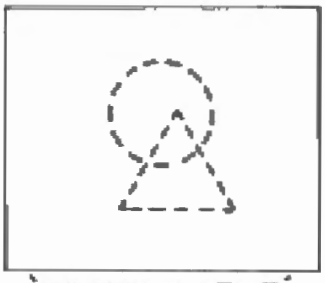
COMPUTER



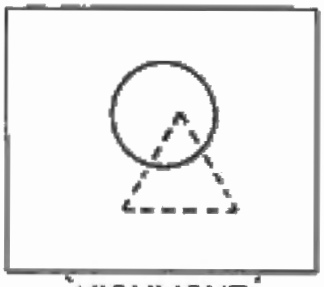
PLAYER



SUPERIMPOSE



TRANSPARENT



HIGHLIGHT

In order to facilitate these mixing modes, the player's video output needs to be synchronised very accurately with that of the computer. To perform this function the player contains circuits which 'genlock' the player's picture. This is apparent each time the screen mode is changed in the form of a gentle wavering of the display for a short period of time; it does not take long for the genlock circuitry to synchronise, or 'lock', the two video pictures together and they will remain locked until the next change of screen mode. It is important to understand that the genlock circuitry *requires* the video interlace to be turned on, so you should *not* issue *TV commands with a second parameter of 1 (eg *TV 0,1).

5.16 Videodisc player commands

The VFS is configured to control a BBC AIV System videodisc player, which contains not only the videodisc access control circuitry, but also the video mixer for merging the player and computer video signals. Below is a summary of the commands which the VFS understands and which may therefore be used to control the player from programs:

Command	Meaning	
*VOCOMPUTER	or *VP 2	Video from computer only
*VODISC	or *VP 1	Video from LaserVision only
*VOSUPERIMPOSE	or *VP 3	Computer overlaid on LaserVision
*VOTRANSPARENT	or *VP 4	Both computer and LaserVision mixed
*VOHIGHLIGHT	or *VP 5	LaserVision enhanced by computer

*FRAME <frame no.>	Go to frame and still
*STILL (<frame no.>)	Go to frame and still
*SEARCH <frame no.>	As *FRAME, but return control immediately
*PLAY (<start>(,<end>))	Go to start frame and play to end frame Default start = current position Default end = end of disc Note: works backwards as well as forwards
*RESET	Send start unit command to player
*AUDIO <channels>	Switch relevant sound channel(s) on/off
*STEP (<mode>)	Stop player and, if mode<>0, move up to 50 frames forwards (mode=1) or backwards (mode=255). <mode> is a signed 8-bit number. Default mode = 0
*FCODE <F-Code string>	Send the string direct to the player
*SLOW <speed>,<direction>	Move at given speed in given direction
*FAST <direction>	Move at 3 * play speed in direction
*EJECT	Stop the player and eject the disc
*NOEJECT	Disables the 'Eject' button on the player
*CHAPTER <digits>,<opt char>	Play chapter sequence
*VP <n>	Send LaserVision video mix command <n>

Notes

<channels>	0 Both audio channels off 1 Channel 1 on, channel 2 off 2 Channel 2 on, channel 1 off 3 Both channels on
<direction>	0-127: forwards 128-255: backwards
<speed>	5 is slowest; 253 fastest (in accordance with Philips standards, only numbers in the range from 5 to 253 are valid)

The default LaserVision video mix command is VP3, so that LaserVision video is seen overlaid by computer video.

To prevent command clashes with other ROMs, an optional prefix of 'LV' can be added to any command, thereby ensuring that the VFS gets the command.

Command descriptions

Below is a description of the F-codes which the VFS sends to the player to perform each of the 'star commands' listed above. A summary of F-codes appears in Chapter 6.

*FRAME issues a *SEARCH command, and then polls the player for command completion. This can be useful when viewing a sequence of 'stills'. After issuing *SEARCH, no subsequent video command will be executed until the search has been completed, as the player has no buffering of commands. Otherwise sending another command before the first one has finished would result in the loss of the second one. The *SEARCH command sends F-code FxxxxxR. Polling with *FRAME causes a wait for an 'A0' response from the player. If *SEARCH doesn't receive a reply within about 12 seconds, a timeout is caused, thus enabling further commands.

*STILL without a parameter is interpreted as *STEP 0, which halts the video with picture display on. *STILL with a frame number as its argument has the same effect as *FRAME <frame no.>.

*PLAY determines which of its two parameters is the larger, and then sends a *FRAME command with the first parameter, a load auto-stop register of the second (F-code FxxxxxS), and finally a normal play forward (F-code N) or reverse (F-code O).

If *PLAY is issued without parameters, the player plays from the currently displayed frame to the end of the disc (though the user can enter *SLOW, *STILL, etc). *PLAY with just one parameter causes a 'Bad number' error.

Note that *PLAY returns control to the user once it has instructed the LaserVision player, ie as soon as the player starts playing the sequence, not when the sequence ends. User programs wishing to wait for the sequence to end must poll the player for an 'A2' return code (ie repeatedly read F-codes until 'A2' is detected). See OSWORD &64.

Note that, since the player's LVDOS controller turns the video off after data access, *FRAME, *PLAY and *STILL turn the video on (F-code E1) before commencing action.

*RESET sends the SCSI 'start unit' command (SCSI code &1B). See Chapter 6 for the SCSI command list.

*AUDIO sends F-codes A and B each followed by 1 to turn the sound channel on or 0 to turn it off. Thus *AUDIO 3 results in F-codes A1,B1 being sent, and *AUDIO 2 sends A0,B1, and so on. Note that only certain areas of a disc should have audio signals on, since the audio tracks in other areas of the disc hold LV-ROM data.

STEP sends one of five F-codes: '' for *STEP 0, 'L' for *STEP 1, 'M' for *STEP 255, '+xx' for *STEP 2 to 50 and '−xx' for *STEP 254 to 206.

*FCODE sends any arguments to the LV player, using the F-code protocol. As this allows the full set of MOS special characters to be used (" ", |M, and so on), all ASCII characters can be sent to the player. A full list of F-codes is given in Chapter 6. The parameter may be optionally enclosed within double-quotes.

*SLOW sends F-code Sxxx to set the speed according to the second parameter. However, it first inverts the number, so that low values are slow, and high values are fast. It then sends 'U' (for slow motion forwards) or 'V' (for slow motion backwards).

*FAST sends either 'W' or 'Z', depending upon direction ('W' is fast forwards, and 'Z' is fast reverse).

*EJECT sends " ' " to eject the disc from the player. Note that *EJECT can be disabled with *CONFIGURE NOEJECT and enabled by *CONFIGURE EJECT. This configuration happens immediately.

*CHAPTER sends 'QxxyyzzS' by default so that the player will play a sequence of chapters. (The sequence may be just one chapter.) The string xxyyzz is that supplied by the user.

The optional second parameter may be R or N. If present, the F-code string sent to the player includes that parameter instead of 'S'. 'N' asks for a single chapter to be played (though chapter sequence can be used for this), and R causes the player to go to the chapter and halt (*CHAPTER xx,R). See Chapter 6 for further information on F-codes.

*VP enables use of the full range of VP commands that have been specially provided by Philips for their SCSI-compatible player. VP 1 to VP 5 have the synonyms *VODISC, *VOCOMPUTER, etc, as listed above. VP x asks the player for the current video mixing mode. See OSWORD &64 for details of how to read this.

5.17 Trackerball commands

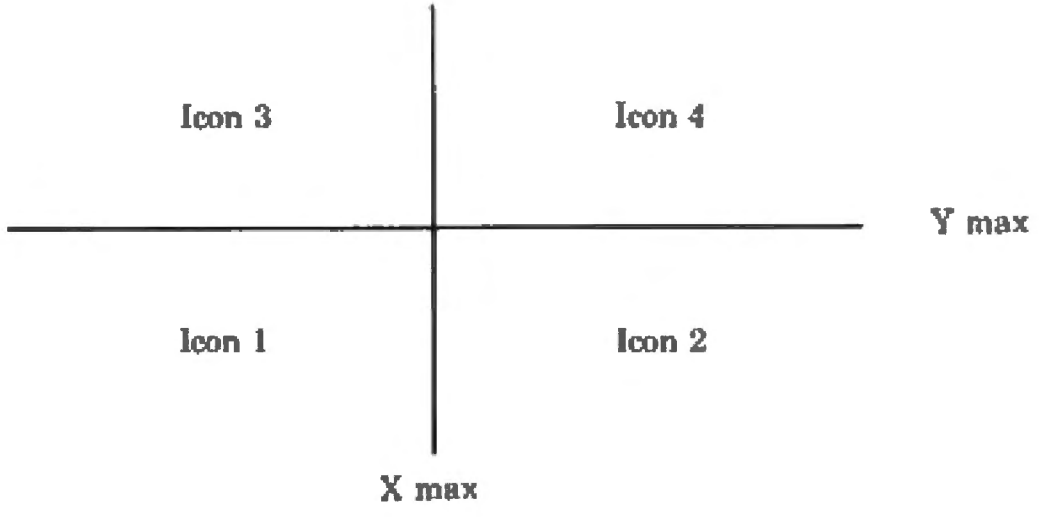
The VFS supports the connection of a pointing device to the computer, in the form of a Trackerball or a Mouse. Trackerball and Mouse commands are identical and interchangeable. Both forms of pointing device are wired to the User Port, using either Marconi Trackerball or AMX Mouse de facto wiring standards. The system, on receiving a *TRACKERBALL or *MOUSE command, automatically determines which device is attached, thus obviating the need for user programs to know which device is fitted; the two commands are otherwise synonymous. The Trackerball position is returned by BASIC's ADVAL function and a pointer is also maintained on the screen if selected.

Four different types of pointer icon may be selected and it is also possible to use different icons in different parts of the screen.

Below is a summary of the VFS commands for controlling pointing devices. Reference to the 'Master Series User Guide' may be useful in understanding their function:

Command	Meaning
*TRACKERBALL 0/1	Set up Trackerball software if the argument is 1. The pointer is off unless switched on by *POINTER. If argument is 0, the Trackerball software is disabled and the pointer is switched off. The default argument is 1.
*MOUSE 0/1	Synonymous with above.
*TSET x,y	Set pointer to x,y. Default is centre of screen.
*TMAX x,y	Set boundaries where pointer will change (see below).
*POINTER 0/1/2/3	Erase/Display/Erase and halt/Display and halt pointer. The default is 1. Only works if device is on (ie after a *TRACKERBALL 1 command).
OSBYTE 128 X	X takes the value: 0-4 – normal ADVAL results 5 – maximum X is returned 6 – maximum Y is returned 7 – X coordinate 8 – Y coordinate 9 – buttons; bottom three bits of X hold button settings (L to R) These values are returned in XY (low, high).

The boundaries at which the pointer changes (defined by *TMAX) work in the following manner:



Icon 1 is a cross, icon 2 is a magnifying glass, icon 3 is a NW-facing arrow, and icon 4 is a NE-facing arrow.

The default boundaries are below, and to the right of, the screen, so that only icon 3 is displayed. Note that the coordinates for the pointer are always absolute with point (0,0) at the bottom left of the screen. This means that the coordinates are in the standard screen format, and are not affected by VDU 29 or VDU 24 commands (define origin and define graphics window).

The pointer only works in screen modes 0, 1 and 2 (with shadow as an option). This is not an overriding limitation, since the mouse positions can still be used, and users can write their own pointer plotter for a desired application, if a screen mode other than those mentioned above is deemed necessary.

All pointers are in logical colour 1. They are therefore red in MODEs 1 and 2, and white in MODE 0, but can be changed using the palette.

Each pointer is plotted with regard to its 'active point'. This is at the tips of the arrows, the centre of the cross, and the middle of the glass in the magnifying glass. Hence the coordinates returned by ADVAL(7) and ADVAL(8) are at the active point of the current icon.

The three buttons on the Trackerball/Mouse mimic keys on the keyboard. The left-hand button equates to the RETURN key, the centre button to TAB, and the right-hand button returns code 192 in the keyboard buffer. Note that these keys do not auto-repeat.

The pointer can be moved near the edge of the screen with impunity, as it is automatically 'clipped' to display only the visible bits of the pointer. In pointer mode 1 the pointer will follow the movements of the Trackerball/Mouse. In pointer mode 3 any such movement will have no effect on the pointer.

Technical notes

Since the Trackerball pointer code uses the 10ms polling semaphore provided on all Master machines, it follows the ball very accurately. As the pointer is only updated if the screen coordinates have changed, leaving the Trackerball alone does not cause the processor to do any unnecessary plotting.

It should be noted that the Trackerball software is filing system independent. Hence swapping filing systems in and out does not affect it.

The command *TRACKERBALL 1 (or *MOUSE 1) positions the coordinates in the centre of the screen (640,512). *TSET can, of course, then be used to move them elsewhere.

The pointer software also uses zero page locations &84 to &8F inclusive. However, as it preserves them on the stack, and restores them after each plot, this is not normally noticeable.

Note that the screen should not be scrolled with the pointer displayed, and that the pointer code assumes the screen memory map is in a default (MODE or CLS) state. Any limitations in the pointer code arise from the interests of fast executable code, and should not prove excessive.

Although the pointer always preserves the state of any memory underneath, it cannot cope with remembering any text/graphics written on top of it, nor can it clear the screen or change mode while active. If one of these actions is required, it is important to turn the pointer off (preferably using *POINTER 2 so that the user cannot move the Trackerball without the pointer), carry out the action (write to the screen or change mode), and then turn the pointer back on (using *POINTER 1).

Bytes &D92 to &D9E are used by the Trackerball routines. Within these, bytes &D97 and &D98 hold the old osbyte vector.

Furthermore, some locations at the start of page 9 are used. These are preserved, though, and thus will not affect software unless it has an NMI routine which uses these locations. This limitation applies to locations &84 to &8F in language workspace.

The purpose of the *POINTER 3 mode divorcing itself from the Trackerball/Mouse movement is to allow users to write their own device drivers but continue to use existing pointer software by issuing *TSET commands to move the pointer.

6. Programmer's guide

The information in this chapter is for programmers who wish to drive the Master Series A1V microcomputer directly from assembly language. Sections 'O', 'P' and 'Q' of the 'Master Series Reference Manual' are essential reading for anyone wanting to write assembler programs for the BBC Microcomputer Series. Most of the necessary information for using the filing system in assembly language is presented there. In this chapter the main points are summarised and particular uses of OSWORD are described in detail.

6.1 MOS routines – general principles

There are a number of operating system routines available to handle disc input/output. All the routines must be called with a machine-code JSR instruction and with the decimal flag clear; the BASIC instructions CALL and USR are usually sufficient for this purpose. These routines are called in address range from &FF00 to &FFFF. Each routine, when called, calls an internal (OS ROM resident) routine whose address is stored in RAM between &0200 and &02FF. (The OS routine is said to 'indirect' through one of these addresses.) The internal routine addresses vary according to the filing system in use. For example, the routine OSFIND to open or close a file is entered at &FFCE, and is indirected via &021C. &021C and &021D contain the address of (or the 'vector' to) the executable routine in the filing system ROM. (Note that the foregoing is somewhat simplified; a detailed description of the situation for sideways ROMs is beyond the scope of this manual.) See the 'Reference Manual Part One' for further information.

Using the available routines all necessary functions relating to disc files may be performed. The relevant routines together with their entry points are summarised below (the third column gives the names of the vectors to the internal routines):

OSFIND	&FFCE	FINDV	&021C	Open or close a file for byte access
OSFILE	&FFDD	FILEV	&0212	Load a complete file
OSARGS	&FFDA	ARGSV	&0214	Load data about an open file
OSGBPB	&FFD1	GBPBV	&021A	Load a number of bytes
OSBGET	&FFD7	BGETV	&0216	Read one byte from an open file
OSWORD	&FFF1	WORDV	&020C	Load a number of sectors (and other functions)

6.2 OSFIND

Call address &FFCE (indirects through &021C).

OSFIND is used to open and close files. Opening a file declares a file requiring byte access to the filing system. Closing a file declares that byte access is complete. A file must be opened prior to using OSARGS, OSBGET, OSBPUT or OSGBPB with it.

A=0 Causes a file (or files) to be closed
A=&40 Causes a file to be opened for input only (reading)
A=&C0 Causes a file to be opened for input only (reading)

If A=0, a file, or all files, will be closed depending on the value of Y. Y=0 closes all files, otherwise the file whose channel number is given in Y is closed.

If A=&40 or &C0, Y (high byte) and X (low byte) must contain the address in memory of the filename, terminated by a carriage return (ASCII &0D). On exit, A will contain the channel number allocated to the file for all future operations (the file 'handle'). If on exit A=0, the operating system was unable to open the file.

On exit from OSFIND, X and Y are preserved, C, N, V and Z are undefined and D=0. The interrupt state is preserved, but interrupts may be enabled during the operation.

Note that there is no difference between A=&40 and A=&C0; the latter is provided for compatibility with other filing systems.

6.3 OSFILE

Call address &FFDD (indirects through &212).

This routine performs actions on whole files, namely loading a file into memory and loading file 'catalogue information' (see A=5 below).

On entry A indicates the function to be performed. X (low byte) and Y (high byte) point to an 18-byte parameter block, structured as shown below (the left hand column shows addresses relative to the base address given by X and Y).

00	Address of filename, which must end with a carriage return	LSB
01		MSB
02	Load address of file	LSB
03		
04		
05		MSB
06	Flag for using disc address or bytes 02-05	LSB

07	Execution address on return from OSFILE	
08		
09		MSB
0A	Length of file	LSB
0B		
0C		
0D		MSB
0E	File attributes	LSB
0F		
10		
11		MSB

The following functions are performed by OSFILE according to the value held in A:

A=5 – Read a named file's catalogue information (ie load address, execution address, length, type) from the file's entry in the directory. The object type (see below) is returned in A, the other information being written to the parameter block. (If the object is a directory, default values are returned for the catalogue information.)

A=&FF – Load the named file. The address to which the file is loaded is determined by byte 06 in the parameter block. If this is zero, the address given in the parameter block is used, otherwise the file's own load address is used. The parameter block is then filled in as for A=5.

Object attributes are stored in the last four bytes of the parameter block. The most significant three bytes are undefined; the least significant eight bits, when set, have the following meanings:

Bit	Meaning
0	The file is readable by you
1	Undefined
2	Undefined
3	Undefined
4	The file is readable by others
5	Undefined
6	Undefined
7	Undefined

In VFS, bits 4-7 are always identical to bits 0-3. If the object is a directory, bit 0 is ignored. Note that 'others' in the above context means other users of, say, the Econet filing system.

Object types returned in the accumulator are:

0 Nothing found
1 File found
2 Directory found
FF Protected file ('E' access set)

On exit X and Y are preserved, A contains the object type, C, N, V and Z are undefined. The interrupt status is preserved, but may be enabled during the call.

6.4 OSARGS

Call address &FFDA (indirects through &0214).

This routine reads an open file's arguments (as defined below), or returns the type of filing system in use, according to the value held in Y on entry. In either case, X (on entry) must point to four bytes in page zero.

If Y is non-zero, A determines the function to be carried out on the file whose handle is in Y:

A=0 Read file's sequential pointer (BASIC PTR#)
A=1 Set file's sequential pointer
A=2 Read file's length (BASIC EXT#)

(The file length and sequential pointer are sometimes collectively referred to as the file 'arguments'.)

If Y is zero, the following operations are carried out according to the value in A:

A=FF Ensures all files.
In the VFS this has no effect as all files are read-only and will always return A=0.

A=0 Returns the type of filing system in A:

A=0 – No filing system currently selected
A=1 – 1200 baud cassette
A=2 – 300 baud cassette
A=3 – ROM Filing System
A=4 – DFS Disc Filing System
A=5 – Econet
A=6 – Telesoftware
A=7 – IEEE
A=8 – ADFS Advanced Disc Filing System
A=9 – Host Filing System
A=10 – VFS Videodisc Filing System
A=16 – Acacia RAM Filing System

A=1 Returns address of rest of command line after a star command in the zero page command block

On exit X and Y are preserved and A is set to zero unless otherwise stated. The flags may have been corrupted.

6.5 OSGBPB

Call address &FFD1 (indirects through &021A).

This routine can be used to transfer a number of bytes from an open file, or to transfer filing system information. If single bytes are transferred using the OSBGET routine, the 'overhead' incurred for each transfer has a marked effect on performance times. The greater the number of bytes that can be read together, the more efficient the transfer; a single OSGBPB call can replace an OSARGS call, and a large number of OSBGET calls.

On entry X (low byte) and Y (high byte) point to a control block in memory. A defines the operation to be performed. The control block format is shown below (the left-hand column shows addresses relative to the base address given by X and Y):

00	File handle	
01	Pointer to memory area used to transfer data to	LSB
02		
03		
04		MSB
05	Number of bytes to transfer	LSB
06		
07		
08		MSB
09	Sequential pointer value to be used for transfer (if used)	LSB
0A		
0B		
0C		MSB

The sequential pointer value given replaces the old sequential pointer value.

The value held in A determines the type of operation:

A=1 or 2-Return 'Disc read only' error for compatibility with other filing systems
A=3 Read bytes from disc, using new sequential pointer value
A=4 Read bytes from disc, using old sequential pointer value
A=5 Read currently selected directory title, boot up option, and drive number

The data returned are:

- Single byte giving the length of the title
- The title in ASCII character values
- Single byte giving the start option
- Single byte giving the drive number

A=6 Read currently selected directory name. The data returned are:

- 1 (length of drive number)
- ASCII-coded drive number
- Single byte giving the length of the name
- The name in ASCII character values
- A zero byte for compatibility with the NFS (owner access)

A=7 Read currently selected library name. The data returned are:

- 1 (length of drive number)
- ASCII-coded library drive number
- Single byte giving the length of the name
- The name in ASCII character values
- A zero byte for compatibility with the NFS (owner access)

A=8 Read filenames from the CSD. The control block is:

00	CSD master sequence number returned here	
01	Pointer to memory area used to transfer filenames to	LSB
02		
03		
04		MSB
05	Number of filenames to read	LSB
06		
07		
08		MSB
09	File counter (search begins with first file if this is zero)	LSB
0A		
0B		
0C		MSB

The data returned are:

- Length of filename 1
- Filename 1
- Length of filename 2
- Filename 2 . . .

On exit X, Y and the accumulator are preserved, N, V and Z are undefined, and C is set if the transfer could not be completed. The interrupt state is preserved, but may be enabled during the call.

A requested transfer cannot be completed if the end of the file has been reached, or if there are no more bytes (or filenames) to be transferred. C is set on exit and the number of bytes (or filenames) which have not been transferred is written to the control block (bytes 05-08). The address field (bytes 01-04) is always adjusted to point to the next byte/filename to be transferred, and the sequential pointer always points to the next entry in the file to be transferred.

6.6 OSBGET

Call address &FFD7 (indirects through &0216).

This routine reads a single byte from an open file.

On entry Y contains the file handle, as set up by OSFIND. The byte is read from the point in the file designated by the sequential pointer, as set up by OSARGS.

On exit, X and Y are preserved, A contains the byte read, N, V and Z are undefined. C is set if the end of the file has been reached, indicating that the byte obtained is invalid. The interrupt state is preserved, but may be enabled during the call. The sequential pointer is incremented.

6.7 OSWORD

Call address &FFF1 (indirects through &020C).

The VFS recognises four OSWORD calls. All four require X (low byte) and Y (high byte) to point to an area of memory containing a control block or where results are to be placed.

OSWORD with A=&60 – Read the master sequence number and the status byte.

The master sequence number of the currently selected directory is placed in the location pointed to by YX. It is in binary coded decimal form in the range 0-99 inclusive. YX+1 contains a status byte, structured thus:

Bit number	Meaning if set
0	File ensuring in progress (IRQ pending) [Should never happen]
1	Bad free space map
2	*OPT 1,x flag – set if messages on
3	(Undefined)
4	(Undefined)
5	LVROM controller present
6	The Tube is currently in use by VFS
7	The Tube is present

OSWORD with A=&62 – Access the LVROM disc controller (reads blocks of bytes from the disc or accesses the video frames).

The control block is:

00	0	
01	Start address in memory of data source or destination	LSB
02		
03		
04		MSB
05	Command block to disc controller (see below)	
06		
07		
08		
09		
0A		
0B	Data length in bytes	LSB
0C		
0D		
0E		MSB

If the value in byte 00 above is 0, the controller number defaults to 1. As well as this control block, various status bytes in the VFS workspace are used (eg a byte for the current drive number), and so this OSWORD call only works if VFS is the currently selected filing system (the call should not be made otherwise). If an error of any kind occurs during the execution of the command, the error number is returned in byte 00 of the control block (0 will be returned otherwise). Error codes are detailed below.

The command block is structured thus:

	Bit							
Byte	7	6	5	4	3	2	1	0
00	Function code							
01	X	X	X	Disc address (MSB)				
02								
03	Disc address (LSB)							
04	Sector count							
05	Unused (must be set to 0)							

The three bits marked X X X in byte 01 are ORed with the current drive number to give the drive number to use. For a single drive these bits should all be zero. The function code field can take the values:

Value	Meaning
&00	Test drive ready
&01	Restore (ie move laser head to sector 0)
&03	Request sense status (useful after an error to get more information)
&08	Read
&1B	Start/stop unit
&C8	Read F-code result from LVdos
&CA	Transmit F-code to LVdos

For functions 0 and 1, the disc address (in the command block) should be set to zero.

Note that for function 3 the player will do this automatically and since the player will clear the error after an &03 call this is not very useful to the user. See OSWORD &63.

Data length should be set to zero for all functions except &08, &C8 and &CA. VFS looks at the number of bytes as well as the number of sectors when handling SCSI codes &08 and &C8.

If byte 04 in the command block is non-zero, it is used as a sector count and the data length parameter (bytes 0B to 0E of the main control block) is ignored. Note that for call &1b (Start/Stop unit) bit 0 of the sector count is 1 for Start and 0 for Stop.

Example: To read &1234 bytes starting from sector number &002345 of the current drive, loading into memory at location &FFFF3000 (high bytes FFFF indicating the host machine), the required control block is:

Byte	Value	Meaning
00	&00	Controller number
01	&00	Load address (LS byte)
02	&30	
03	&FF	
04	&FF	Load address (MS byte)
05	&08	Read command
06	&00	Disc address (MS byte)
07	&23	
08	&45	Disc address (LS byte)
09	&00	
0A	&00	
0B	&34	Data length (LS byte)
0C	&12	
0D	&00	
0E	&00	Data length (MS byte)

Example: To send F-code &4F (Normal Play Reverse) to the LV player:

Byte	Value	Meaning
00	&00	Controller number
01	&00	Load address (LS byte)
02	&30	
03	&FF	
04	&FF	Load address (MS byte)
05	&CA	Transmit F-code command
06	&00	Disc address (MS byte)
07	&00	
08	&00	Disc address (LS byte)
09	&01	Transmit 1 F-code
0A	&00	
0B	&00	Data length (LS byte)
0C	&00	As some F-codes are longer than 1 byte, alter byte &0B to suit
0D	&00	
0E	&00	Data length (MS byte)

The F-code to be sent is put (in this case) at &FFFF3000, ie location &3000 in the IO processor. So &3000 should be set to (in this case) &4F and &3001 to &OD before calling the OSWORD. Note also that F-codes should be padded with 0 bytes so locations &3002 to &30FF should be set to 0.

OSWORD with A=&63 – Read last error information

This call, if made immediately after a disc error of some kind (including a data error in sequential filing) returns error information (in the control block) as follows:

Byte		
00	Disc address where error occurred,	(LSB)
01	including drive number in three most	
02	significant bits of byte 02	(MSB)
03	Disc error number, top bit set=valid address	
04	Channel number of file where error occurred	

Only one of bytes 03 and 04 will be valid, depending on the type of error.

OSWORD with A=&64 – Read current F-code

On entry, X and Y point to a 16 byte parameter block. The F-code will be left in this on exit. For example:

```
10 DIM block 16
20 AX=&64
```

```
30 XX=block
40 YX=block DIV 256
50 CALL &FFF1
60 PRINT $block
```

6.8 Trackerball connections

The Trackerball connections to the User Port are:

Pin no.	Name	Marconi	AMX
1	+5		
2	CB1	X1	X1
3	+5		
4	CB2	Y1	Y1
5	0		
6	PB0	Button	X2
7	0		
8	PB1	Button	
9	0		
10	PB2	Button	Y2
11	0		
12	PB3	X2	
13	0		
14	PB4	Y2	
15	0		
16	PB5		Button
17	0		
18	PB6		Button
19	0		
20	PB7		Button

6.9 F-code command list

This table lists the necessary codes to be sent by the computer to the player in order to perform each function.

Dec = Decimal code
Hex = Hexidecimal code
Char = Character

Dec	Hex	Char	Function required
33	21	!xy	Sound insert (beep)
35	23	#xy	RC-5 command out via A/V EUROCONNECTOR

36	24	$0	Replay switch disable
		$1	Replay switch enable (default)
39	27	'	Eject (open the frontloader tray)
41	29	)0	Transmission delay off (default)
		)1	Transmission delay on
42	2A	*	Halt (still mode)
		*xxxxx+yy	Repetitive halt and jump forward
		*xxxxx−yy	Repetitive halt and jump backward
43	2B	+yy	Instant jump forward yy tracks (max 50)
44	2C	,0	Standby (unload)
		,1	On (load)
45	2D	−yy	Instant jump backward yy tracks (max 50)
47	2F	/	Pause (halt + all muted)
58	3A	:	Reset to default values
63	3F	?F	Picture number request
		?C	Chapter number request
		?D	Disc program status request
		?P	Player status request
		?U	User code request
		?=	Revision level request
65	41	A0	Audio-1 off
		A1	Audio-1 on (default)
66	42	B0	Audio-2 off
		B1	Audio-2 on (default)
67	43	C0	Chapter number display off (default)
		C1	Chapter number display on
68	44	D0	Picture number/time code display off (default)
		D1	Picture number/time code display on
69	45	E0	Video off
		E1	Video on (default)
70	46	FxxxxxI	Load picture number information register
		FxxxxxS	Load picture number stop register
		FxxxxxR	Goto picture number then Still mode
		FxxxxxN	Goto picture number then normal play forward
		FxxxxxQ	Goto picture number and continue previous play mode
72	48	H0	Remote control not routed to computer (default)
		H1	Remote control routed to computer
73	49	I0	Local front-panel buttons disabled
		I1	Local front-panel buttons enabled (default)
74	4A	J0	Remote control disabled for player control
		J1	Remote control enabled for player control (default)
76	4C	L	Still forward
77	4D	M	Still reverse

78	4E	N	Normal play forward
		Nxxxxx+yy	Repetitive play forward and jump forward
		Nxxxxx−yy	Repetitive play forward and jump backward
79	4F	O	Play reverse
		Oxxxxx+yy	Play reverse and jump forward
		Oxxxxx−yy	Play reverse and jump reverse
81	51	QxxR	Goto chapter and halt
		QxxN	Goto chapter and play
		QxxyyzzS	Goto chapter (sequence) and halt
83	53	SxxxF	Set fast speed value, 2-40
		SxxxS	Set slow speed value, 2-250
84	54	TxxyyN	Goto time code xx=min, yy=sec (yy=opt)
		TxxyyI	Load time code info register (yy=opt)
85	55	U	Slow motion forward
86	56	V	Slow motion reverse
		VPy	Video overlay (VP1 is default)
87	57	W	Fast forward
88	58	X	Clear
90	5A	Z	Fast reverse
91	5B	[0	Audio-1 from internal (default)
		[1	Audio-1 from external
92	5C	\0	Video from internal (default)
		\1	Video from external
93	5D	]0	Audio-2 from internal (default)
		]1	Audio-2 from external
95	5F	_0	Teletext from disc off
		_1	Teletext from disc on (default)

Notes:
1. Each command must be terminated by a carriage return (CR).
2. Digits (x,y,z) must be in ASCII; leading zeros are optional.

6.10 Acknowledgements back to external computer

On some F-code commands, the player will return a response code to the host computer. These are summarised below.

Dec	Hex	Response syntax (ASCII)	Description
79	4F	O	Returned when disc-tray is opened on '(Eject) command, or when disc-tray is open and a command which expects a response is received
83	53	S	Ackn on ON command when disc reaches correct speed
61	3D	= x1 x2 x3 x4 x5	Returned after revision level request (?=)

70	46	F x1 x2 x3 x4 x5	Returned after frame number request command (?F)
67	43	C x1 x2	Returned after chapter number request command (?C)
68	44	D x1 x2 x3 x4 x5	Returned after disc status request command (?D)
80	50	P x1 x2 x3 x4 x5	Returned after player status request command (?P)
85	55	U x1 x2 x3 x4 x5	Returned after user code request command (?U)
86	56	VP1...VP5	Returned after video mode request command (VPX)
88	58	X	Returned after ?F,?C,?D or ?U when the information is not available
65	41	A0	Acknowledgement on FxxxxxR or FxxxxxQ when completed
		A1	Acknowledgement on FxxxxxN when completed
		A2	Acknowledgement on FxxxxxS when stopped
		A3	Acknowledgement on FxxxxxT when paseed
		A6	Acknowledgement on QxxN or QxxR when completed
		A7	Acknowledgement on QxxS when completed
		A8	Acknowledgement on TxxN when completed
		A9	Acknowledgement on Txx1 when passed
		AN	Negative acknowledgement: picture number, chapter number or time code in error

Notes:
1. Each response is terminated by a carriage return (CR).
2. All response characters, including leading zeros, are sent.
3. Digits (x1...x5) are in ASCII.

6.11 Memory usage

The term 'sector' here refers to SCSI 'blocks'.

Sectors 0 and 1 on each drive contain the total number of sectors on the drive, the boot option number, and the free sector gap list. Sectors 2 to 6 inclusive are the root directory.

6.12 The free space map

The free space map (FSM) is stored in sectors 0 and 1 on each drive. The format is:

Sector 0

0 Disc address of first free space (LS byte)
1 Disc address of first free space
2 Disc address of first free space (MS byte)
3 Disc address of second free space (LS byte)
4 Disc address of second free space
5 Disc address of second free space (MS byte)
6 Disc address of third free space (LS byte)
:
:
:
etc for all other free space up to 82 entries
:
:
246 Reserved
247 Reserved
248 Reserved
249 Reserved
250 Reserved
251 Reserved
252 Total number of sectors on disc (LS byte)
253 Total number of sectors on disc
254 Total number of sectors on disc (MS byte)
255 Checksum on free space map, sector 0

Sector 1

0 Length of first free space (LS byte)
1 Length of first free space
2 Length of first free space (MS byte)
3 Length of second free space (LS byte)
4 Length of second free space
5 Length of second free space (MS byte)
6 Length of third free space (LS byte)
:
:
:
etc for all other free space up to 82 entries
:
:

246 Reserved
247 Reserved
248 Reserved
249 Reserved
250 Reserved
251 Disc identifier
252 Disc identifier
253 Boot option number
254 Pointer to end of free space list
255 Checksum on free space map, sector 1

The disc addresses and lengths are in sectors. The free space map is stored in the 'secret' RAM at &C000.

6.13 Directory information

A directory consists of five contiguous sectors on the disc drive with a maximum of 47 entries, each entry comprising 26 bytes as follows:

Name and access string	10 bytes
Load address	4 bytes
Execution address	4 bytes
Length in bytes	4 bytes
Start sector on drive	3 bytes
Sequence number	1 byte
Total	26 bytes

The remaining 58 bytes in the directory are one zero byte, one byte which is the directory master sequence number, 19 bytes of directory title, three bytes for the parent pointer (ie the disc address of ^), a directory name string, and a directory identity string. The master sequence number is incremented every time the directory is rewritten. When an entry is made or changed in the directory the entry's sequence number is set to the directory master sequence number.

The currently selected directory is stored in RAM from &C400 to &C8FF when VFS is selected. The attributes are stored in the top bit of the first four characters of the entry name, so the R attribute of the first entry is in bit 7 of &C405, W in bit 7 of &C406, L in &C407 bit 7, D in &C408 bit 7. R of the second entry is in bit 7 of &C41F and so on. The end of the list of entries is denoted by a 0 in the first character position of the first unused entry, hence the 0 before the directory name. The store map of locations &C400 to &C8FF is:

C400 Master sequence number
C401 Start of text to identify the directory ('Hugo')

C404 End of text
C405 First directory entry
C41E Last byte of 1st directory entry
C41F Second directory entry

```
:      End of last directory entry
:      0 Last entry marker
:
:            ('garbage')
:
```

C8CB 0 Last entry marker ('dummy')
C8CC Directory name
C8D5 End of directory name
C8D6 1st byte of parent pointer (LSB)
C8D8 Last byte of parent pointer (MSB)
C8D9 Directory title
C8F9 Last byte of directory title
C8FA Master sequence number
C8FB 1st byte of text to identify the directory ('Hugo')
C8FE Last byte of text
C8FF Reserved

Location &C400 in the above example contains byte 0 of the first sector of the directory (sector 2 for directory $). Location &C8CB contains byte &CB of the fifth sector of the directory (sector 6 for directory $).

The four bytes which spell 'Hugo' are included for ADFS compatibility.

6.14 Checksum calculation for VFS

Assembler:

```
    LDA #&FF
    TAX
    CLC
.loop
    ADC blockcache-1,X
    DEX
    BNE loop
    STA blockcache+255
```

The above code is executed for the free space map (held on the first two sectors of the disc). It is included here for those who wish to provide VFS compatible discs.

6.15 Error messages

This section lists all the VFS error messages, and their appropriate error code. Although error codes are not displayed, they are included here so users can include error handling sections in any programs which include VFS commands. Section 6.16 details errors in alphabetical order of error names; section 6.17 lists errors in numerical order of error numbers.

6.16 List of error messages

&BD Access violation. An attempt has been made to read or load a file with the R attribute not set.

&AA Bad checksum. Corrupted RAM is preventing VFS from closing or reading a file. The system must be restarted by a hard break.

&FE Bad command. The command given was neither recognised by the VFS nor found as a utility in the CSD or the current library.

&A9 Bad FS map. Either RAM or disc sector 0 or 1 is not able to be loaded. The system must be restarted by a hard break.

&AD Bad mode. An attempt was made to turn the pointer on with the screen not in mode 0, 1 or 2.

&CC Bad name. An illegal filename was used, ie one including $ (dollar) or : (colon) outside the context of a root specification, or with a zero length component of a pathname, or other special characters in the wrong context. For example:

```
*EX $$
*DIR FILE:ONE
*DIR DIR..XDIR 1
*EX A@B
*EX A^B
```

&DC Bad number. A numeric argument to a star command was out of range, non-existent or consisted of non-numeric data. Searches for illegal frame numbers are trapped with this error.

&CB Bad opt. An invalid argument has been assigned to a *OPT command.

&FF Bad parameter. An illegal mode has been specified. For example, for *CHAPTER.

&A8 Broken directory. An attempt has been made to access a directory which is in some way corrupt and as such should not be accessed. This error implies that the disc is in an inconsistent state.

&DE Channel <nn>. A sequential file operation has been attempted with an illegal or unassigned file handle. <nn> is decimal.

&CA Data lost on channel <nn>. A disc error occurred while accessing a sector from the disc in an attempt to read an open file. Caused by memory being illegally overwritten (or hardware problems). <nn> is hexadecimal.

&C7 Disc error <nn> at :<drv>/<sector number>. A fault on the disc was detected by the controller during the last operation. <nn> is the error code, <drv> is the drive number, <sector number> is the sector number (in hexadecimal) where the error was discovered (if appropriate). Some error codes are:

00 – Controller error (should never happen)
02 – Not ready (drive door open)
03 – Media error (disc dirty)
05 – Malformed SCSI command

(The error codes are hexadecimal values, the same as would be returned by an OSWORD &63 call.)

&C9 Disc read only. An attempt has been made to write to the disc.

&CD Drive not ready. Synonymous with disc error 02.

&93 Door open. The player thinks the disc is not ready to be played.

&DF EOF on channel <nn>. End of file. This error occurs if two consecutive attempts have been made to read from a file whose end has been reached. The failure of the first attempt will have been flagged by the contents of the C flag following an OSBGET or OSGBPB command (see Chapter 6). <nn> is decimal.

&D6 Not found. The object referred to was not found.

&C1 Not open for update on channel <nn>. An attempt has been made to write to a random access file which is only open for reading. <nn> is decimal.

&B7 Outside file on channel <nn>. An attempt has been made to set the pointer of a file to a value beyond the end of the file. <nn> is decimal.

&C0 Too many open files. An attempt has been made to open an eleventh file. Only ten files may be open at once.

&AD Turn interlace on. An attempt has been made to change the video mix mode with interlace turned off – this results in the LaserVision video being unable to lock with the BBC video, causing the BBC video to move up the mixed output slowly.

&93 No! An attempt has been made to *RUN a file whose load address is &FFFFFFFF.

&94 No eject. Configuration has been set to disallow eject.

&95 IRQ already indirected. The trackerball software has noticed that someone else has already grabbed IRQ1V and so it doesn't. You should 'unplug' the offending ROMS and restart the system.

6.17 Error codes – numerically ordered list

Note that this list includes ADFS errors which are never generated by the VFS, but are included for compatibility.

Hex	Decimal	Message
&92	146	Aborted
&93	147	No!
&93	147	Door open
&94	148	Bad parameters
&94	148	No eject
&95	149	Too many defects
&95	149	IRQ already indirected
&A8	168	Broken directory
&A9	169	Bad FS map
&AA	170	Bad checksum
&AD	173	Bad mode
&AD	173	Turn interlace on
&B7	183	Outside file on channel <nn>
&BD	189	Access violation
&C0	192	Too many open files
&C1	193	Not open for update on channel <nn>
&C2	194	Already open
&C7	199	Disc error <nn> at: <drv>/<sector number>
&C9	201	Disc read only
&CA	202	Data lost on channel <nn>
&CB	203	Bad opt
&CC	204	Bad name
&CD	205	Drive not ready
&D6	214	Not found
&DC	220	Bad number
&DE	222	Channel on channel <nn>
&DF	223	EOF on channel <nn>
&FD	253	Wildcards
&FE	254	Bad command
&FF	255	Bad parameter

6.18 Responses to computer on commands from remote control handset

Player commands from remote control handset when routed to host computer, after H1 command (RC to computer on), are of the form:

Dec	Hex	Syntax
76	4C	Lx

where x is given by the following codes:

STANDBY	'
DISPLAY	!
NEXT	*
CLEAR	X
ENTER	P
START/REPEAT	F
AUDIO1	A
AUDIO2	B
CNR	R
PNR	D
CORR	C
GOTO	K
FAST▶	W
FAST◀	Z
SLOW▶	T
SLOW◀	U
SPEED+	H
SPEED−	G
TXT	Y
PAUSE	V
SEARCH▶	>
SEARCH◀	<
STILL▶	L
STILL◀	M
PLAY▶	N
PLAY◀	O

Similarly, when an H1 command routes RC commands to the host computer, the numeric keys of the remote control handset will give a response of the form:

Dec	Hex	Syntax
86	56	Vx

where x is the key value in ASCII:

DIGIT0	0
DIGIT1	1
DIGIT2	2
DIGIT3	3
DIGIT4	4
DIGIT5	5
DIGIT6	6
DIGIT7	7
DIGIT8	8
DIGIT9	9

Note: Each response is terminated by a carriage return (CR).

7. The Small Computer Systems Interface (SCSI)

The videodisc player used in the BBC AIV System is controlled by means of a high speed parallel channel, the Small Computer Systems Interface (SCSI), which connects to the Master Series 1MHz bus. SCSI is an industry-standard data bus, corresponding to an ANSI specification. It is a general-purpose peripheral bus which makes few assumptions about the devices which are connected to it, and both this and its potentially high bandwidth make it well suited to controlling the videodisc player.

SCSI communications protocol

The SCSI standard defines a sophisticated protocol for data transfers between multiple hosts and multiple peripherals on the bus. The Master AIV computer supports the subset of the full SCSI standard which is required by the application. A standard handshaking sequence for data transfer goes along these lines:

Before beginning a transaction over the SCSI bus, the SCSI standard requires the device which is starting the transaction (known as the 'Initiator') to begin an **arbitration phase** in order to gain control of the bus. The BBC AIV System does not support this phase, and it is mentioned only for the sake of completeness.

When the computer wishes to gain the attention of a device (the 'Target'), it does so by starting the **selection phase** of the SCSI protocol, in which it outputs the **SCSI address** of the device on the bus (a unique number in the range from zero to seven) and waits for a response.

Assuming the selection phase is successful, the host computer must then indicate to the player what it wishes to do. This is called the **command phase**, in which the host computer sends a **command descriptor block** containing the command that it wishes the player to carry out and any additional information needed. The videodisc player's commands are known as 'F-codes' and they are listed in section 6.9.

The command phase is followed, where neccessary, by a **data in** or **data out phase**, in which the information is transferred from player to computer, or vice versa. When reading data from the player, there may be a delay of up to several seconds before the data is sent, because of the mechanical limitations on the speed of the player's mechanism.

The data phase is followed by a **status phase**, during which the player returns a single byte to the computer indicating the success or failure of the operation. In the latter case, the computer may request further information on the failure with the **request sense status** command. Status bytes are summarised in section 6.10

Lastly, the player issues a **command complete message** to the computer and then releases the bus. This is a requirement of the SCSI standard, but serves no purpose for LVDOS since all its operations are synchronous (ie completed before the status is returned).

Logical units

The SCSI standard defines the logical subdivision of devices connected to the SCSI bus as a hierarchy; at the top are the devices themselves which may be addressed as several distinct 'logical units'; these logical units in turn contain blocks of data, each block comprising a number of bytes. One of the values in the SCSI command descriptor block is the **logical unit number (LUN)**, in the range from zero to seven. This is provided by the SCSI standard to address different logical units within the same device.

The data part of the videodisc is divided into **volumes** of data, each of which may be related to a LUN by means of the 'open' and 'close' F-codes. Normally, these relations are initialised at power-on by the player reading defaults from the disc. Currently the VFS can access six logical units. They are numbered zero to five, with number six reserved for future use. There is also a seventh unit which addresses the entire videodisc. The Domesday videodiscs do not make use of the volume/LUN facilities and as a result the entire disc responds as volume zero.

In the future, a more versatile mechanism may be introduced to allow many more volumes on the disc by letting the user select the desired volume from a list by name. The current VFS will work with discs that have default volume to LUN mappings, but will not cope with more than six volumes. Chapter 6 contains more information on the *MOUNT command, which is responsible for volume selection.

The videodisc player supports a similar, but quite unrelated, subdivision of the disc's *video* contents whereby the disc itself is subdivided into 'chapters', each of which comprises an arbitrary number of video frames. This feature may be exploited through the use of F-codes described in section 6.9.

Appendix

Default status

The BBC Master AIV Microcomputer should be configured with the default status listed below.

Note that some of the parameters are especially important to the AIV System, whereas others are optional. Those of particular importance are marked thus †.

```
  Baud              4
  No Boot
  Caps
  Data              4
  Delay             50
  No Directory
† Internal Tube
  FDrive            0
  File              15
  Hard
  Ignore            10
  Lang              12
  Mode              7
† Tube
  Loud
  Print             1
  Repeat            8
  Scroll
† TV                0,0
† VFSDir
  Eject
```

Configuring for auto-boot

Use *CONFIGURE (see page 192 of the 'Welcome Guide') as follows:

```
BASIC
    > *CO. BOOT[RETURN]
    > *CO. FILE 8[RETURN]
    > *CO. VFSDIR[RETURN]
```

then press [CTRL]+[BREAK].

From now on the system will auto-boot, provided there is an appropriate LV-ROM LaserVision disc in the player, whenever the computer is switched on.

'Warm boot' can be initiated by pressing [BREAK] and complete reboot is initiated by pressing [CTRL]+[BREAK].

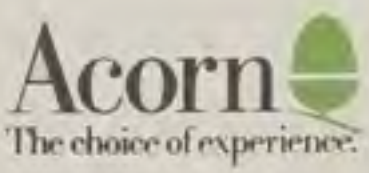
To return to BASIC press [SHIFT]+space bar+[CTRL]+[BREAK] and release in reverse order.

Booting from VFS

When the auto-boot feature has not been configured, the VFS may be invoked by pressing [CTRL]+Q+[BREAK]. Provided that a suitable LV-ROM LaserVision disc is in the player, the computer will eventually return the BASIC prompt. When this occurs, the user may issue VFS * commands from the keyboard (see section 5.16). Alternatively the retrieval software can be booted by pressing [SHIFT]+[BREAK].

If no disc is present, an error message will be displayed. In either case the computer may be used in standalone mode (ie without necessarily invoking the laser disc-based software). However, if the LaserVision player is switched off, the display will be blank.

Index



Acorn Computers Limited
Fulbourn Road
Cherry Hinton
Cambridge CB1 4JN
England