Advanced Computer Products

Advanced Disc Toolkit
User Guide

NOTICE

Advanced Computer Products Limited reserves the right to make improvements to the product described in this manual at any time and without notice.

Advanced Computer Products Ltd
6 Ava House
High Street
Chobham
SURREY GU24 8LZ

Tel: (0276) 76545

Advanced Computer Products Limited cannot be held responsible for any loss of data or damage to equipment as a result of this product.

This product is not intended to operate in sideways RAM.

Tube, Electron and Econet are tradenames of Acorn Computers Limited.

CONTENTS

1 Introduction 4

2 **ADT** Commands 5

BACKUP 8
BFIND 9
BUILD 10
CATALL 11
DCOMP 12
DEX 13
DFIND 15
DIRALL 16
DUMP 17
ENVELOPE 18
FCOMP 19
FCOPY 20
FORM 21
FREE 23
FSN 24
KEYL 25
LIST 26
MAP 27
MDUMP 28
MENU 29
MEX 30
MFIND 32
MLOAD 33
MRUN 34
ROMS 35
SECTORS 36
SETADR 37
SPT 38
SWAP 39
TYPE 40
UNPLUG 41
VERIFY 42
XFER 43
VERIFY 44

3 Command Summary 45

4 Fitting **ADT** 46

1 INTRODUCTION

ADT is a utility Rom designed to be used in a BBC Computer or Acorn Electron fitted with the Disc Filing System, DFS or the Advanced Disc Filing System, ADFS.

It increases the computing power of the machine by adding new commands to the Machine's Operating System, MOS. These commands are instantly accessible from inside a program or typed in from the keyboard.

22 of the commands are disc utilities which use one of the Acorn disc filing systems. The other commands are general utilities which can be used in any filing system, such as a TAPE machine without a disc interface.

ADT has five utilities which are also in DFS. These are BACKUP, BUILD, DUMP, LIST and TYPE. These are in **ADT** for machines which only have ADFS as a disc filing system. This is because ADFS does not have these utilities in Rom, they are supplied as utilities on disc.

For information on the disc filing system, refer to the filing system's disc user guide.

Chapter 2 explains the parameters used by **ADT** commands followed by a description of each command, under the headings Purpose, Examples and Note. It is advisable not to just **read** the examples, but use them as well. This will often make the description of the command much clearer. In the examples **text appearing like this** should be typed into the computer.

Chapter 3 gives a brief description of each command.

Chapter 4 shows how to fit **ADT** into a BBC Computer or Electron.

2 ADT COMMANDS

Type ***HELP ADT**

This shows a list of all the commands. Their parameters are shown in brackets after the command. A parameter is a name, number, address or ad letter which the command needs to operate correctly. Some parameters are optional, meaning the command will supply its own number or address. Optional parameters are shown in round brackets, eg. (<start>), (<rom>), (<drive>), etc.

If a command is typed in with a parameter missing, a message will be displayed showing the correct syntax for the command, eg.
SYNTAX : MOVE <start> <end> <dest> (<rom>).

To avoid command names conflicting with other Roms, **ADT** commands can be preceded with the letter A, eg. *AMENU is the same as *MENU and *AFORM is the same as *FORM. For instance, DFS and **ADT** both have utilities *DUMP. If you wanted to be sure of using the *DUMP in **ADT** then you could use *ADUMP.

Commands can be abbreviated by adding a full stop to the abbreviated name, so long as the name remains unique to **ADT** and other MOS commands, eg. *XF. is the same as *XFER and *UN. is the same as *UNPLUG. *KEY is a MOS command and *KEYL is an **ADT** command. *KEYL cannot be abbreviated because it could be interpreted as *KEY by the MOS.

ADT commands will be accepted in both upper and lower case.

From now on, the BBC Microcomputer used with the second processor will be referred to as the I/O processor.

Parameter Definitions

In this manual, & printed in front of a number means the number is in hexadecimal (hex), base 16, otherwise the number is in decimal, base 10. When entering a hex number in an **ADT** command, & should not be printer in front of the number.

<b> - bytes, 0-255.

Determines how data is printed on the screen, and how much on each line. A byte can be printed as a hex number, an ASCII character, or part of a disassembled line. The list below shows how a byte is printed and the byte increments per line for different values of <b>.

<b> = 0	6502 disassembler
<b> = 1-99	ASCII + hex in <b> byte increments
<b> = 100	ASCII in 32 byte increments
<b> = 101-199	ASCII in <b> - 100 byte increments
<b> = 200	hex in 11 byte increments
<b> = 201-255	hex in <b> - 200 byte increments

<d.adr> - Disc sector address

A sector address is a hex number which refers to any one sector on a disc. The *INFO command gives the location of a file as a disc sector

address. To translate a track, sector address into a disc sector address and vice versa, use the following formula:

For DFS,

```
SECTOR ADDRESS = TRACK*10 + SECTOR
TRACK  = SECTOR ADDRESS DIV 10
SECTOR = SECTOR ADDRESS MOD 10
```

For ADFS

```
SECTOR ADDRESS = TRACK*16 + SECTOR
TRACK  = SECTOR ADDRESS DIV 16
SECTOR = SECTOR ADDRESS MOD 16
```

<dest> - Destination drive number.

This is used by *MOVE to mean a destination memory address.

<end> - End memory address. See <start>

Sometimes it is easier to express an end memory address as the number of hex bytes from the start memory address. This can be done by preceding the number with +, eg. +100, +500, etc.

<fs.afsp> - Filing system name.ambiguous file specification.

<fsp> - File specification.

<load> - Load memory address.

(L) - Large disc, ie. double sided 80 track. (ADFS only)

(M) - Medium sized disc, ie. single sided 80 track. (ADFS only)

<n> - Decimal number, 0-255.

<rom> - Sideways ROM number, 0-15.

<spt> - Sectors per track, 10-20.

If the Acorn DFS or ADFS is the current filing system, then this parameter need not be used. It is assumed DFS will use 10 sectors per track and ADFS 16 sectors per track.

Some DFS's use 16,17 or 18 sectors per track, in a double density mode. <spt> can be used to change the default number of sectors per track.

Note. This parameter relies on the fact that the DFS supports a call to access any sector on a track via OSWORD &7F.

<srce> - Source drive number.

<start> - Start memory address.

A memory address is a hex number from 0 to &FFFFFFFF, so called 32 bit addressing.

If a second processor is not connected, addresses 0 to &FFFEFFFF will refer to memory in the second processor and addresses &FFFF0000 to &FFFFFFFF will refer to memory in the I/O processor. To summarise:

I/O Processor Only

addresses 0-&FFFF - I/O processor memory

I/O Processor and Second Processor

addresses 0-&FFFEFFFF - second processor memory
addresses &FFFF0000-&FFFFFFFF - I/O processor memory

For more information on I/O and second processor memory addresses refer to chapter 9 'Distinguishing between memories' in the 6502 Second Processor User Guide.

<str> - String.

Strings which contain spaces must be enclosed within double quotation marks. The wildcard character # can be used to replace a single character in a string and represent all possible characters, eg. AB# could represent ABA, ABB, ABC, etc.

To enter a string as a hex number precede the number with &, eg. &20F4FF.

(T) - Tube

This parameter is only effective if a second processor is connected to the I/O processor, and turned on.

Because the second processor has more free memory than the I/O processor, which is not affected in size by screen mode or Rom software increasing the value of PAGE, commands which use the <T> option can make use of this extra memory. However anything in memory before the command is used will be overwritten. The main advantage of using the second processor is an increase in speed.

***BACKUP** <srce> <dest> (T)

Purpose

To copy the contents of one disc onto another. <srce> is the source drive from which sectors are read and <dest> is the destination drive to which sectors are written. If (T) is given then the second processor will be used.

A disc can be copied using a single drive, where <srce> and <dest> are both the source and destination drives, but swapping the disc will be necessary between reading and writing to the discs.

Before copying the disc a message is displayed to confirm that this is really what you want. Press **Y** to copy the disc, or any other key to exit the command.

Examples

***BACKUP 0 1**
Go (Y/N) ? **Y**

Copies the disc in drive 0 onto the disc in drive 1.

***BACKUP 0 1 T**
Go (Y/N) ? **Y**

Copies the disc in drive 0 onto the disc in drive 1, using the second processor.

Note

Because this program uses free memory it is advisable to save your program first. It is not possible to copy between discs of different sizes, eg. 40 track disc onto an 80 track disc, or of different formats, eg. DFS disc onto an ADFS disc. This command can only be used in a disc filing system.

***BFIND** <str>

Purpose

To search a BASIC program for every occurrence of the string <str> and print the line number and string where it is found. The rest of the BASIC line is not listed.

Examples

***BFIND HELLO**

Searches for the string "HELLO".

***BFIND P%**

Searches for the integer variable P%.

***BFIND JSR#######**

Searches for a string 10 characters long starting with "JSR".

***BFIND &F4**

Searches for the hex byte &F4, which is the token for the BASIC keyword REM. Any BASIC keyword can be searched for if the token of the keyword is known. These are listed on pages 483, 484 of the BBC User Guide.

***BUILD** <fsp>

Purpose

To create a text file called <fsp>, consisting of characters typed in from the keyboard, for the purpose of being *EXEC'd later. A line number is printed before each new line of text. Entry of characters into the file can be terminated by pressing <ESCAPE> at the start of a new line.

Example

```
*BUILD !BOOT
0001 MODE 3
0002 VDU19,0,4,0,0,0,0
0003 *AMENU $
0004 <ESCAPE>
```

Creates an EXEC file called !BOOT.

*CATALL

Purpose

To list the filenames in the current directory and the directories which are a member of the current directory. The directory name is printed first, followed by the filenames. Directories are shown with a D after their name. The list is identified by one space for each level of directory entered. This command is of most use in ADFS which uses a hierarchical 'tree' structure of directories, that can be nested to numerous levels.

Example

```
*CATALL
$
 !BOOT
 menu
 GAME      D
  BOARD     D
   board1
   board2
   board3
  ARCADE    D
   arcade1
   arcade2
   arcade3
 LIBRARY   D
  utility1
  utility2
  utility3
```

Lists the filenames in each directory starting from the current directory, in this example $.

Note

In DFS this command will list the filenames in the current directory only, since DFS does not allow directories to be nested, ie. a directory cannot contain another directory.

***DCOMP** <srce> <dest> (<n>)

Purpose

To compare the source disc in drive <srce> with the destination disc in drive <dest> sector for sector, and print the sector address of the sectors which are not the same. The command is terminated after the first eight different sectors are found. If <n> is used, the command will be terminated after the first <n> different sectors. If <n> is zero all different sector addresses will be printed.

A single drive can be used to compare two disc, where <srce> and <dest> are both the source and destination drives, but swapping the discs will be necessary after reading each disc.

Examples

***DCOMP 0 1**

Compares the discs in drive 0 and 1 and terminates after the first eight different sectors.

***DCOMP 0 1 1**

Compares the discs in drive 0 and 1 and terminates after the first different sector.

Note

Comparing discs uses free memory, so it is advisable to save your program first before using this command. It is not possible to compare two discs of different sizes, eg. a 40 and 80 track disc, and discs of different type, eg. a DFS and ADFS disc. This command can only be used in a disc filing system.

***DEX** (<d.adr>) (<spt>)

Purpose

To examine and edit disc sectors. <d.adr> is the sector to be edited, by default 0. <spt> is the number of sectors per track, by default 10 for DFS and 16 for ADFS.

Each sector contains 256 bytes of which come or all can be displayed at once depending upon the screen mode. Modes 0 and 3 will display the whole sector. The format of the display is shown below. Moving the cursor is done using the four cursor keys. Editing a byte is done by overtyping the byte from the keyboard. The new byte will appear as hex inside the round brackets and as an ASCII character above the flashing cursor. To enter numbers as hex bytes, press the <COPY> key. The round brackets will change to square brackets. Now only hex numbers will be accepted, ie. 0-9 and A-F. Press <COPY> again to enter characters as ASCII. Moving forward or back a track or sector is done using the four <SHIFT> cursor keys. If the sector has been altered a message is displayed for saving the sector back onto the disc. Press **Y** to save the sector, or any other key not to save it.

Press <ESCAPE> to exit this command.

Format of *DEX in a 40 column screen mode.

```
DRIVE:0 TRACK:00 SECTOR:00 / 000000



00 (00)00 00 00 00 00 00 00 ........
08  21 42 4F 4F 54 20 20 24 !BOOT  $
10  00 00 00 00 00 00 00 00 ........
18  00 00 00 00 00 00 00 00 ........
20  00 00 00 00 00 00 00 00 ........
18  00 00 00 00 00 00 00 00 ........
30  00 00 00 00 00 00 00 00 ........
38  00 00 00 00 00 00 00 00 ........
etc.
```

Editing Keys

CURSOR keys	- cursor movement
SHIFT left cursor	- back one sector
SHIFT right cursor	- forward one sector
SHIFT down cursor	- forward one track
SHIFT up cursor	- back one track
COPY	- input type, hex or ASCII
CTRL P	- print screen
ESCAPE	- exit command

Examples

***DEX**

To edit the first sector of the disc in the current drive.

***DEX :1.30**

To edit sector &30 of the disc in drive 1.

***INFO !BOOT**
$.!BOOT   000000 FFFFFF 000010 127

This shows that the file !BOOT starts on sector 127.

***DEX 127**

To edit the first sector of !BOOT.

Note

A disc to be edited must be of the correct format for the current filing system, eg. if ADFS is the current filing system then this command can only edit ADFS discs, similarly if DFS is the current filing system then this command can only edit DFS discs. The current filing system can be identified with the *FSN command. Editing discs is only possible in a disc filing system, and in a screen mode which has 40 or 80 columns.

***DFIND** <str> (<d.adr>)(<d.adr>)

Purpose

To search a disc for every occurrence of the string <str> and print the sector, byte address and string where the string is found. The second parameter <d.adr> determines which drive and sector the search will start, by default 0 in the current drive, and the last parameter <d.adr> is the last sector, where the search will end.

Examples

***DFIND !BOOT**

Searches all sectors in the current drive for "!BOOT".

***DFIND "!BOOT $" 0 2**

Searches sectors 0 and 1 in the current drive for "!BOOT $".

***DFIND #BOOT :1.0+2**

Searches the first two sectors in drive 1 for a string five characters long, ending in "BOOT".

***DFIND &21424F4F54 :1**

Searches the disc in drive 1 for the sequence of bytes &21424F4F54.

***DFIND** ########

Displays all sectors in the current drive as ASCII characters.

Note

This command can only be used in a disc filing system.

*DIRALL

Purpose

To list all the directory names in the current directory and the directories which are a member of the current directory. The list is indented by one space for each level of directory entered. This command is of most use in ADFS which uses a hierarchical 'tree' structure of directories, that can be nested to numerous levels.

Example

```
*DIRALL
$
 GAME      D
  ARCADE    D
  BOARD     D
 LIBRARY   D
```

Lists all the directory names starting from the current directory, in this example $.

Note

In DFS this command will print the current directory name only, since DFS does not allow directories to be nested, ie. a directory cannot contain another directory.

***DUMP** <fsp> (<b>)

Purpose

To display the contents of file <fsp>. <b> determines the format of the display, by default 8, which displays the file as hex bytes and ASCII characters in increments of 8 bytes (see page 5).

Example

***DUMP !**BOOT

Displays !BOOT in hex and ASCII.

***DUMP MCODE 0**

Displays MCODE in disassembler.

***DUMP LETTER 100**

Displays LETTER in ASCII only.

***DUMP NUMBERS 200**

Displays NUMBERS in hex only.

Note

Because DFS has a similar utility *DUMP, it might appear that this command is not working properly. To be sure of using the *DUMP in **ADT**, precede the command with the letter A, eg. *ADUMP.

*ENVELOPE (<n>) ...

Purpose

To list envelope definition <n>, by default envelope definitions 1 to 16. An explanation of the envelope parameters can be found on pages 182 and 245 of the BBC User Guide.

Examples

***ENVELOPE**

Lists envelope definitions 1 to 16.

***ENVELOPE 1**

Lists envelope definition 1.

***ENVELOPE 1 2 3**

Lists envelope definitions 1, 2 and 3.

***FCOMP** <fsp> <fsp> (<n>)

Purpose

To compare file <fsp> with file <fsp> byte for byte and print the address and bytes for the two files in hex and ASCII where they differ. The command is terminated after the first eight differences are found. If <n> is used the command is terminated after the first <n> differences. If <n> is zero, all differences will be printed.

Examples

***FCOMP LETTER LETTER2**

Compares LETTER with LETTER2 and terminates after the first eight different bytes.

***FCOMP LETTER LETTER2 1**

Compares LETTER with LETTER2 and terminates after the first different byte.

Note

A comparison of two files is only possible if they are the same length.

***FCOPY** <fsp> <fsp> (T)

Purpose

To create a copy of file <fsp> and give it a new name <fsp>. The file names must be different. If (T) is given then the second processor will be used. This is useful for copying large files.

Examples

***FCOPY LETTER LETTER2**

Creates a copy of LETTER called LETTER2.

***FCOPY :0.LETTER :1.LETTER2**

Creates a copy of LETTER in drive 0 called LETTER2 in drive 1.

***FORM 40/80** (<drive>)(M)(L)(C)...

Purpose

To initialise a new disc for reading and writing. The first parameter is the number of tracks to be formatted onto the disc. A 40 track drive will use 40 track discs, and an 80 track drive will use 80 track discs. <drive> is the drive number which contains the disc to be formatted. (M) and (L) only apply to formatting ADFS discs - use M to format a single sided 80 track disc, and L to format a double sided 80 track disc. (C) only applies to formatting a DFS disc. It is used to create a dual catalogue disc. The second and third parameters (<drive>) and (M)(L)(C) can be repeated in the command to format more than one disc in the same command. Before formatting the first disc a message is displayed to confirm that this is really what you want. Press **Y** to format the disc or any other key to exit the command.

Examples for DFS

*FORM80
Format which drive ? **0**
Go (Y/N) ? **Y**

Formats an 80 track disc in drive 0.

***FORM40**
Format which drive ? **0**
Go (Y/N) ? **Y**

Formats a 40 track disc in drive 0.

***FORM80 0 C 2 C**
Go (Y/N) ? **Y**

Formats both sides of an 80 track disc in drive 0 with dual catalogues.

Examples for ADFS

***FORM80**
Format which drive ? **0** (M) (L) ? **M**
Go (Y/N) ? **Y**

Formats a single sided 80 track disc in drive 0.

***FORM40 0**
Go (Y/N) ? **Y**

Formats a single sided 40 track disc in drive 0. It is not possible to format a double sided 40 track disc.

***FORM80 0 L 1 L**
Go (Y/N) ? **Y**

Formats a double sided 80 track disc in drive 0 and 1.

Note

DFS and ADFS use discs formatted to a different specification. The *FORM command will format a disc for use in the current disc filing system. For example, it is not possible to format a DFS disc whilst ADFS is the current filing system. Use the *FSN command to identify the current filing system. Formatting an ADFS disc uses free memory, so it is advisable to save your program first before using this command in ADFS.

Formatting an ADFS disc can cause parts of the screen to be overwritten in some screen modes. However, this will not affect the formatting command.

***FREE** (<drive>)

Purpose

To display the number of free and used files and disc space remaining and used on the disc in drive <drive>, by default the current drive. Free and used files are given in decimal. Disc space is given as sectors is hex and bytes in decimal.

Examples

***FREE**

Displays the free space on the current drive.

***FREE 2**

Displays the free space on drive 2.

Note

This command only operates in DFS on a DFS disc. However, ADFS has a similar command *FREE, which displays the amount of free space in the current drive.

***FSN** (<n>)

Purpose

To identify the current filing system by name. If <n> is used, the filing system name which has the filing system number <n> will be printed.

Examples

***FSN**
Disc filing system

DFS is the current filing system.

***FSN 3**
ROM filing system

Filing system number 3 is the ROM filing system.

*KEYL (<n>) ...

Purpose

To list function key definition <n>, by default definitions 0 to 15. A definition includes the *KEY syntax and key number so it can easily be edited using the cursor and copy keys. Function keys with no definitions are not listed.

Examples

***KEYL**

Lists key definitions 0 to 15.

***KEYL 5**

Lists key definition 5.

***KEYL 0 1**

Lists key definitions 0 and 1.

***LIST** <fsp>

Purpose

To list the text file <fsp> with line numbers.

Example

```
*LIST !BOOT
0001 MODE 3
0002 VDU19,0,4,0,0,0,0
0003 *AMENU $
```

Lists the text file !BOOT.

***MAP** (<drive>)

Purpose

To list a map of the free space on the disc in drive <drive>, by default the current drive. The map is printed as a list of disc addresses and lengths of free space in sectors, both in hex.

The free space map changes whenever a file is saved or deleted. This causes spaces to form between files, fragmenting the disc space. To eliminate the free spaces use the filing system command *COMPACT.

Examples

***MAP**

Lists the free space map on the current drive.

***MAP 1**

Lists the free space map on drive 1.

Note

This command only operates in DFS on a DFS disc. However, ADFS has a similar command *MAP, which lists the free space map on the current drive.

***MDUMP** <start> <end> (<b>) (<rom>)

Purpose

To display the contents of memory starting from address <start> and ending at address <end>. <b> determines the format of the display, by default 8, which displays memory as hex bytes and ASCII characters in increments of 8 bytes (see page 5). <rom> determines which Rom is used if memory is read from addresses &8000 to &BFFF in the I/O processor.

Examples

***MDUMP 1900 1A00**

Displays memory from &1900 to &1A00 in hex and ASCII.

***MDUMP 8000+4000 0 14**

Disassembles memory in Rom 14 from &8000 to &C000.

***MDUMP FFFF8000+4000 0** 14

Disassembles memory in Rom 14 from &8000 to &C000 in the I/O processor. Same as previous example but with a second processor.

MODE 0
***MDUMP 0 FFFF 172**

Displays memory in Mode 0 from 0 to &FFFF in ASCII.

MODE 0
***MDUMP 0 FFFF 18**

Displays memory in Mode 0 from 0 to &FFFF in hex and ASCII.

***MENU** (<dir>)

Purpose

To display the files in directory <dir>, by default the current directory, on the screen so that one can be selected for execution. The title of the directory is printed at the top of the screen and the entry names are printed below. An entry pointer => can be moved using the four cursor keys to point to one of the entries on the screen. To execute a program move the pointer opposite the filename and press <RETURN>. The program will be loaded into memory and RUN. To load a program only, press **L**. To enter directory $, press **$**.

The program is RUN according to the type of program it is, ie. BASIC, EXEC, or machine code using the *MRUN command. Alternatively, press **E** to *EXEC a file, **R** to *RUN a file or **C** to CHAIN a BASIC program.

In ADFS a directory entry can be a directory name. A directory name is shown with a D after it. To enter the directory, move the pointer opposite the directory's name and press **<RETURN>**. To move to a directory's parent, ie. one level back, press **^**. To move to the previous directory, press **B**. Using these keys it is possible to enter all directories on a disc and consequently to find any program.

Press **<ESCAPE>** to exit this command.

Key Definitions

CURSOR keys	- pointer movement
RETURN	- RUN program
L	- LOAD program
C	- CHAIN BASIC program
R	- *RUN program
E	- *EXEC program
$	- enter directory $
ESCAPE	- exit command

ADFS Key Definitions

^	- enter parent directory
&	- enter Root directory, same as $
B	- enter previous directory

Examples

***MENU**

Displays files from the current directory.

***MENU $**

Displays files from directory $.

Note

It is not possible to use this command in screen modes 2 and 5.

***MEX** (<start>) (<b>) (<rom>)

Purpose

To examine and edit the contents of memory. <start> is the start address from which memory is displayed, by default PAGE. <b> determines the format of the display, by default 8, which displays memory as hex bytes and ASCII characters, in increments of 8 bytes (see page 5). <rom> determines which Rom is used if memory is read from addresses &8000 to &BFFF in the I/O processor.

The cursor points to the current memory byte which is displayed as a hex byte surrounded by round brackets and an ASCII character above a flashing cursor. Moving the cursor is done using the four cursor keys. Editing is done by overtyping the current memory byte from the keyboard. The display will be updated to reflect the change in memory.

To enter numbers as hex bytes press **<COPY>**. The round brackets will change to square brackets. Now only hex numbers will be accepted, ie. 0-9 and A-F. Press **<COPY>** again to enter characters in ASCII.

Moving forward or back a screen is done using the **SHIFT** cursor keys. To move the current memory byte to the top left of the window press **CTRL ^** (**CTRL .** on an Electron). Whilst viewing memory from a Rom, press **CTRL R** to view memory from the next Rom number. To examine memory on the other side of the Tube, if a second processor is connected, press **T**.

If memory is viewed in disassembler, it is possible to follow the address of a JSR, JMP or branch instruction by pressing **<RETURN>** whilst the memory pointer is over the instruction opcode.

Press **<ESCAPE>** to exit this command.

Editing Keys

CURSOR keys	- cursor movement
SHIFT left cursor	- move cursor to left margin
SHIFT right cursor	- move cursor to right margin
SHIFT down cursor	- move one page down
SHIFT up cursor	- move one page up
COPY	- swap cursor, hex/ASCII
TAB (CTRL I on Electron)	- cycle display format
CTRL ^ (CTRL . on Electron)	- home current memory byte
CTRL P	- print screen
CTRL R	- increment Rom number
CTRL T	- examine other side of Tube
ESCAPE	- exit command

Disassembler Only

RETURN	- follow JMP, JSR or Branch
CTRL X	- return from JSR

Examples

***MEX**

To examine memory at PAGE, in hex and ASCII.

***MEX 8000 0 14**

To disassemble memory from &8000 in Rom 14.

MODE 0
***MEX 8000 172 14**

To examine memory in Mode 0 from &8000 in Rom 14 in ASCII.

MODE 3
***MEX 8000 18 14**

To examine memory in Mode 3 from &8000 in Rom 14 in hex and ASCII.

***MFIND** <str>(<start>)(<end>)(<rom>)

Purpose

To search screen memory for the string <str>, starting from address <start>, by default 0, and ending at address <end>, by default &FFFF. <rom> is the Rom number to read if the search includes memory from &8000 to &BFFF, in the I/O processor. The address of the string in hex and the string is printed for every occurrence of the string.

Examples

***MFIND BASIC**

Searches memory for the string "BASIC".

***MFIND BASIC D000 DFFF**

Searches memory from &D000 to &DFFF, for the string "BASIC".

***MFIND &4241534943**

Searches all of memory for the bytes &4241534943.

***MFIND** ################

Displays all of memory in ASCII, in increments of 16 bytes.

Note

Using this command in screen mode 7 can produce some surprising results, when the search includes screen memory, from &7C00 to &7FFF in the I/O processor. Every occurrence of the string is printed on the screen, which in so doing produces another occurrence, rapidly filling the screen with strings. To solve this, confine the search to memory above or below screen memory, or change to another screen mode.

***MLOAD** <fsp> (<load>)

Purpose

To load file <fsp> into memory and move it to the address <load>. If <load> is omitted from the command, the file's load address is used instead.

This command is useful for loading a program from disc which is designed to run at an address below PAGE.

Example

***MLOAD GAME**

Loads GAME and then moves it to its load address.

***MLOAD GAME E00**

Loads GAME and then moves it to &E00.

Note

If a program is moved to an address below OSHWM, normally PAGE, the TAPE filing system is selected.

***MOVE** <start> <end> <dest> (<rom>)

Purpose

To move a block of memory starting at address <start>, ending at address <end>, and moved to address <dest>. <rom> is the Rom number used if start is an address between &8000 and &BFFF, in the I/O processor.

Examples

***MOVE 1900 1A00 1B00**

Moves memory from &1900 to &1A00 to address &1B00.

***MOVE 8000+4000 2000 15**

Moves contents of Rom 15 to address &2000.

***MOVE FFFF8000+4000 800 15**

Moves contents of Rom 15 to address &800. If a second processor was connected, memory would be copied from the I/O processor to the second processor.

***MRUN** <fsp> (<load>) (<exec>)

Purpose

To load file <fsp>, move it to the address <load>, and start execution of the program at address <exec>. If <load> and/or <exec> are omitted from the command the files load and/or execution addresses, respectively, are used instead. This command is useful for executing a BASIC or machine code program which is designed to run at an address below PAGE.

This command makes a sensible guess about a program's language ie. BASIC, EXEC or machine code and RUNS the program accordingly. It reads the file's execution address to determine its language. The list below shows which execution addresses can be used to identify a program. A file's addresses can be changed with the *SETADR command.

Language	**Execution Address**
BASIC	- &8023
"	- &801F
"	- &B823
"	- &B82B
EXEC	- &00000000
"	- &FFFFFFFF
machine code	- any other address

Examples

***MRUN GAME**

Runs GAME using the file's own load and execution addresses.

***MRUN GAME E00 E00**

Runs GAME at &E00

Note

If a program is loaded below OSHWM, normally PAGE, the TAPE filing system is selected.

***ROMS** (<rom>) ...

Purpose

To catalogue sideways Rom <rom>, by default Roms 0 to 15. <rom> can be the title of the Rom as an alternative to the Rom number. The catalogue of a Rom shows its Rom socket number, the type of program in the Rom, the title of the Rom and its version number, if it has one. A Rom can be a language shown as (L), a service Rom (like **ADT**) shown as (S), or both a language and a service Rom shown as (SL). A Rom which has been unplugged will appear as (**). See *UNPLUG.

Examples

***ROMS**

Catalogues Roms 0 to 15.

***ROMS 15**

Catalogues Rom socket 15.

***ROMS BASIC**

Catalogues the Rom socket containing BASIC.

***ROMS 12 13 14 15**

Catalogues Rom sockets 12, 13, 14 and 15.

***SECTORS** <d.adr><d.adr><start> R/W

Purpose

To read or write sectors from or to a disc. The first parameter specifies the drive and first sector to read or write. If a drive is not specified the current drive is used. The second parameter specifies the last sector to read or write. <start> is the memory address to which the sectors are read or written. The last parameter, by default R, determines whether sectors are used from disc into memory, R or written onto disc from memory, W.

The second parameter <d.adr> can be given as the number of bytes in hex to read or write if preceded by +. This makes it easy to read a file into memory using a file's sector address and length.

Examples

***SECTORS 0 2 1900**

Reads sectors 0 and 1 from the current drive to address &1900.

***SECTORS :1.0 2 1900**

Reads sector 0 and 1 from drive 1 to address &1900.

***INFO !BOOT**
$.!BOOT 000000 FFFFFF 000010 027

This shows the first sector of !BOOT is on sector &27.

***SECTORS 27+10 1900**

Reads !BOOT at sector &27 or address &1900.

Note

This command can only be used in a disc filing system.

***SETADR** <fsp> <load> (<exec>)

Purpose

To set the load address of file <fsp> to <load> and the execution address to <exec>. If <exec> is omitted, the file's execution address will remain unaltered. Use the filing system's *INFO command to display a file's addresses.

Examples

***SETADR GAME 1900**

Changes GAME's load address to &1900.

***SETADR GAME 1900 2300**

Changes GAME's load address to &1900 and execution address to &2300.

***SETADR GAME 1900 8023**

Changes GAME's load address to &1900 and execution address to &8023. (The execution address &8023 is used by the *MRUN command to identify a BASIC program.)

*SPT (<n>)

Purpose

To change the default number of sectors per track to <n>. The default is 10 as used by the Acorn DFS. Some Acorn compatible DFS's can operate in a double density mode which use up to 18 sectors per track. This command can be used so that **ADT** commands like DEX, DFIND, SECTORS will work on discs which have up to 18 sectors per track.

Examples

***SPT 18**

Changes the number of sectors per track to 18.

***SPT**

Changes the number of sectors per track to 10.

Note

This command can only be used in the Disc Filing System. Pressing **<BREAK>** resets the number of sectors per track to 10. This command will accept a decimal number between 10 and 31 inclusive. Any other number will be treated as 10.

***SWAP** <drive>

Purpose

To swap the two catalogues on a dual catalogue disc in the current drive. See the *FORM command.

Examples

***SWAP**

Swaps the catalogues on the disc in current drive.

***SWAP 1**

Swaps the catalogues on the disc in drive 1.

Note

This command can only be used in the Disc Filing System on a dual catalogue DFS disc, formatted by the FORM command. All DFS commands should operate normally on a dual catalogue disc, with the exception of *BACKUP. It is possible to backup a dual catalogue disc with the *BACKUP command providing the right catalogue n the disc in the current catalogue. The right catalogue is the one which has the most free and used disc space as displayed by the *FREE command. One of the two catalogues will have twice as much free and used disc space as the other. Ensure this is the current catalogue before using the *BACKUP command.

***TYPE** <fsp>

Purpose

To list text file <fsp> without line numbers. See *LIST,

Example

```
*TYPE !BOOT
MODE 3
VDU 19,0,4,0,0,0,0
*AMEND $
```

Types the text file !BOOT.

***UNPLUG** (<rom>) ...*

Purpose

To 'unplug' sideways Rom <rom>. This has the effect of turning a Rom off, without having to physically remove it from its socket. <rom> can be the title of the Rom as an alternative to the Rom number.

If <rom> is omitted from the command, each Rom title is displayed followed by :. Press **Y** to 'unplug' the Rom or any other key to leave the Rom intact. If a Rom which has already been 'unplugged', is displayed and the response to the prompt is **N**, the Rom will be 'plugged' back in.

Examples

***UNPLUG 14**

To unplug the Rom in socket 14.

***UNPLUG DFS**

To unplug DFS. The value of PAGE will not be effected until **<BREAK>** or **<CTRL><BREAK>** is pressed.

Note

This command should be used with caution. Some Roms can respond unpredictably to this command, and not produce the intended effect. Pressing **<BREAK>** or **<CTRL><BREAK>** after unplugging a Rom may or may not plug it back in. **Type ***HELP to see if the Rom has been plugged back in. To recover any 'lost' Roms turn the machine off and then on, or use ***FX200,2** and then press **<CTRL><BREAK>**.

***VERIFY** (<drive>)

Purpose

To verify all sectors of the disc in drive <drive> for legibility. If <drive> is omitted from the command a message is displayed for a drive to verify. Press the number of the drive to verify. <drive> can be repeated in the command to verify more than one disc in the same command. A track number is printed in hex as each track is verified.

If verifying a track fails on the first attempt a ? is printed after the track number. This could mean temporary corruption of the disc, caused by dust, lint or other foreign matter adhering to the surface of the disc.

If verifying a track fails after six attempts the command is terminated and a disc fault is printed. This could mean permanent corruption of the disc, caused by physical damage to the surface of the disc.

Examples

***VERIFY**

Verifies the disc in drive 0.

***VERIFY 0 1**

Verifies the discs in drive 0 and 1.

Note

This command will not overwrite a program in memory, and can only be used in a disc filing system.

***XFER** <fs.fsp> <fs.fsp> (T)

Purpose

To copy a file from one filing system to another. The first parameter specifies the source filing system and filename. The second parameter specifies the destination filing system and filename. The filing system should be given as a name. eg. TAPE, DISC, ADFS, ETC. If (T) is given then the second processor will be used. This is useful for transferring large files.

Examples

***XFER TAPE DISC**

To transfer all files from TAPE to DFS. Filenames will be concatenated to seven characters.

***XFER TAPE.MUSIC DISC**

To transfer MUSIC from TAPE to DFS. The destination filename is MUSIC.

***XFER DISC.:1.$.MUSIC ADFS.:0.$.LIBRARY.SOUNDS**

To transfer $.MUSIC in drive 1 to DFS to LIBRARY.SOUNDS in drive 0 in ADFS.

***XFER DISC ADFS T**

To transfer all files in the current directory in DFS to the current directory in ADFS using the second processor.

Note

This command will transfer files between filing systems which support the commands to read and write files. It is not possible to transfer a file from DISC to ROM. Copying a large file from TAPE might cause part of the file to overwrite the screen in some modes, and consequently display a 'No room' error. This might be avoided by selecting Mode 7 (Mode 6 on Electron) before transferring the file or alternatively, using the (T) option is a second processor is connected.

3 COMMAND SUMMARY

*BACKUP - copy one disc onto another
*BFIND - search a BASIC program for a string
*BUILD - create a text file
*CATALL - list filenames from the current directory
*DCOMP - compare two discs
*DEX - disc sector editor
*DFIND - search a disc for string
*DIRALL - list directories from the current directory
*DUMP - display the contents of a file
*ENVELOPE - list the envelope definitions
*FCOMP - compare two files
*FCOPY - create a copy of a file
*FORM - initialise a new disc for reading and writing
*FREE - display the amount of free space on a disc
*FSN - identify the current filing system
*KEYL - list the function key definitions
*LIST - list a text file with line numbers
*MAP - display a map of the free space on a disc
*MDUMP - display the contents of memory
*MENU - select a program from a menu for execution
*MEX - memory editor
*MFIND - search memory for a string
*MLOAD - load a program to a specific memory address
*MOVE - move a block of memory
*MRUN - run a program at a specific memory address
*ROMS - catalogue the sideways Rom sockets
*SECTORS - read or write sectors onto a disc
*SETADR - change a file's load and execution addresses
*SPT - change the default sectors per track
*SWAP - swap catalogues on a dual catalogue disc
*TYPE - list a text file without line numbers
*UNPLUG - turn off a Rom
*VERIFY - verify a disc for legibility
*XFER - transfer files between two filing systems

4 FITTING ADT

For the BBC Computer

After turning off the power, take off the top cover of the computer by removing the four fixing screws located at the rear of the computer and on the underside near the front. Remove the four screws securing the keyboard to the computer and move carefully to one side just enough to expose the four Rom sockets located on the right side of the board. Insert **ADT** into one of the empty Rom sockets, with the little notch on the chip having the back of the computer. Take care not to bend any of the legs on the chip. Replace the keyboard and top cover.

To check the Rom has been properly inserted into the socket turn on the power and type ***HELP ADT**. A list of commands should appear on the screen. If no list appears it is likely the Rom is not correctly in position. Check that all the legs of the Rom are in place and the Rom is facing the right way.

For the Electron

For some **ADT** disc utilities to operate correctly in screen modes 0,1,2 and 3 a small link, LK1 must be joined inside the Plus 3. This may invalidate the warranty on the Plus 3. If necessary see your dealer.

Checking LK1

The link in the Plus 3 may already be joined. This can be checked by performing the following procedure. First, fit the Rom cartridge into the Plus 1 as described below. With the computer turned on, select screen mode 0, by typing **MODE 0**, and then try verifying a disc, by inserting a formatted disc into the disc drive and typing ***VERIFY**. If the disc verifies OK then the link has already been joined, and nothing more need be done. If none of the tracks verify then the link has not been joined. If the disc verifies OK you will have noticed a 'snow storm' effect on the screen while the disc drive is spinning. This will occur when some **ADT** disc utilities are used in screen modes 0, 1, 2 and 3 and is confirmation that LK1 has been joined.

Joining LK1

After turning off the power, isolate the Plus 3 by disconnecting the Plus 1 and Electron. Remove the top cover of the Plus 3, by unscrewing the nine fixing screws on the bottom of the Plus 3.

Locate LK1 situated to the top left of the disc drive between IC16 and IC10. If the link is not joined make the connection with a small piece of wire, ideally soldering it to the board. Replace the cover to the Plus 3 and re-connect to the Electron and Plus 1. Fit the Rom cartridge to the Plus 1 as described below. To check the link has been properly joined, follow the procedure above, 'Checking LK1'.

Fitting the Rom cartridge

Turn off the power and insert the Rom cartridge into one of the slots in the top of the Plus 1. Ensure the label on the cartridge faces the

keyboard. If in doubt refer to 'Using cartridges' on page 16 of the Plus 1 User Guide. To check that the cartridge has been properly inserted turn on the power to the computer and type ***HELP ADT**. A list of commands should appear on the screen. If no list appears, it is likely the Rom cartridge is not correctly positioned in the Plus 1.

ACP, 6 Ava House, Chobham, SURREY. Tel: 0276 76545