DUTILS

The DUTILS commands replicate manual tasks such as inserting a disk into a drive.
A list of the commands can be displayed by entering `*HELP DUTILS` on the command line.

Parameters

<drv>   Drive number (0 to 3)
<dno>   Disk number (0 to a maximum of 510 dependent on the MMB image size)
<dsp>   Disk specification, i.e. disk title (not case sensitive)
<adsp> Ambiguous disk specification, i.e. disk title terminated with optional wildcard "*"

Parameters in brackets are optional, and "/" indicates "either/or".

Where the drive is optional, if no drive is given then the current drive (set using `*DRIVE`) is assumed.

Note: If disk titles contain spaces they must be enclosed in quotes.

Commands

DIN (<drv>) <dno>/<dsp>

Insert specified disk into drive.

Example: `*DIN 0 REVS`

Note: A disk cannot be in more than one drive at a time. E.g. if Disk 200 is in Drive 0, and the user enters `*DIN 2 200`, Disk 200 will then be in Drive 2, and "No Disk" will be in Drive 0.

DBOOT <dno>/<dsp>

Inserts specified disk into Drive 0 and boots it (if boot option of disk set).

Example: `*DBOOT Planetoid`

DCAT ((<from dno>) <to dno>) (<adsp>)

Lists disks in disk number order.

An optional disk number range and/or ambiguous disk title can be specified.

Example: `*DCAT 10 20 R*` displays all (formatted) disks with numbers in the range 10 to 20 whose disk title begins with "R" or "r":

```
 12 Reversi       P 13 Revs5Track    P
 19 Repton3       P
 3 disks found
```

Note: "**P**" after the disk title indicates that the disk is locked (read-only), and the number of disks found matching any specification is given at the bottom of the list.

DDISKS (<drv>)

Lists the drives and which disks are currently "inserted", plus their status.

```
Example:     *DDISKS
             0:  99 Elite       P
             1:   1 My Games
             2: 124 Meteors     P
             3: 499 Monaco      P
```

DLOCK <dno>/<dsp>

Locks a disk so that it is read-only.

```
Example:     *DLOCK REVS
```

DUNLOCK <dno>/<dsp>

Unlocks a disk so that it can be written to.

```
Example:     *DUNLOCK "My Data"
```

DFREE

Displays the number of unformatted disks, and the total number of formatted and unformatted disks.

```
Example:     *DFREE
             20 of 510 disks free (unformatted)
```

DKILL <dno>

Marks a disk as unformatted.

`*ENABLE` is required before this command can be used, and the user is asked for confirmation.

```
Example:     *ENABLE
             *DKILL 483
             Kill 483 Junk Disk      P : Y
```

NB: This command does not change any data on the disk "surface" and can be undone using **`*DRESTORE.`**

DRESTORE <dno>

Marks a previously unformatted disk as formatted.

```
Example:     *DRESTORE 483
```

DNEW (<drv>)

Finds the first unformatted disk, formats it and places it in specified drive.

Example: ***DNEW**
             Disk 100 in drive 0

DFORM <dno>

Formats the specified unformatted disk. This is similar to ***DRESTORE** but a blank catalogue is written to the disk "surface".

DONBOOT <drv> <dno>/<dsp>

Defines which disks are in Drives 0 to 3 when the machine is initially booted, or 'Z' is pressed with the 'Break' key.

Example: ***DONBOOT 0 EXILE**

DRECAT

Rebuilds the "Disk Table". The disk table contains a copy of the disk title, plus the disk status (read-only, read/write, unformatted or invalid).

The title in the disk table is updated when ***TITLE** is used, or a (new) disk is formatted using ***DFORM** or ***DNEW.** However, copying a disk using ***BACKUP** etc. will not update the "Disk Table". In this case ***DRECAT** can be used to refresh the disk table.

DROM (<rom>) <fsp>

This command loads a file into sideways ram. The optional 'rom' argument is a number between 0 and 15, which designates which sideways ram bank is to be used. If rom = 0 (the default) then the sideways ram with the highest rom number is used, if rom =1, then that with the second highest rom number is used, etc.

This command copies a small amount of code to Page &D, and uses Pages &E and &F as a buffer. Files must be exactly 1KB, 2KB, 4KB, 8KB or 16KB in size.

Differences between the MMB version and the original Acorn DFS 0.90

<u>Selection of DFS on boot or when pressing 'Break'</u>

A normal DFS is selected by pressing 'D' when booting (though normally this is required only if there is another filing system with a higher priority, e.g. the Network Filing System).

However, the MMB version is selected by pressing 'M' or 'Z'. In the later case the disks in the drives are reinitialised to those stated using the `*DONBOOT` command.

Pressing Shift (on its own or with 'M' or 'Z') when booting will cause the filing system to attempt to boot the disk in Drive 0.

<u>TUBE support</u>

The MMB version correctly implements the Tube Protocol, and copies the Tube Code to Page &4. The original did neither, which is the reason Acorn included the DNFS rom with the Second Processor.

However, the maximum file size which can be 'loaded' or 'saved' is 63.75KB, since only one byte is used to count the sectors.

<u>DFS 0.90 Bugs</u>

Some of the bugs identified in the "Advanced Disk User Guide" on page 426 have been fixed, namely:

- The catalogue is always read from the disk, e.g. on *CAT, or before being altered.
- The PTR# bug is fixed.
- CLOSE #0 appears to work correctly probably due to the first item in this list.

<u>Osword &7F</u>

Osword &7F is supported, but only three of the 8271 FDC's commands are recognised, all others being ignored. These are:

- Write FDC Output Port (Side Select) – Op Code &7A
- Read Sector(s) – Op Code &53
- Write Sector(s) – Op Code &4B

(Note: Acornsoft's game Acheton uses Osword &7F to read its data files!)

<u>Notes:</u>

- When the MMC card is being accessed, both the "CAPS LOCK" and "SHIFT LOCK" LED's are illuminated.

ZERO PAGE MEMORY USAGE

A0	**datptr%**
A1	
A2	**sec%**
A3	
A4	
A5	**seccount%**
A6	**skipsec% / attempts%**
A7	**byteslastsec%**

tempptr% = &B8
gdptr% / dfptr% = &F2 (OS text pointer)

WRITE BUFFER PAGE &E00

The DFS code assumes an MMC card
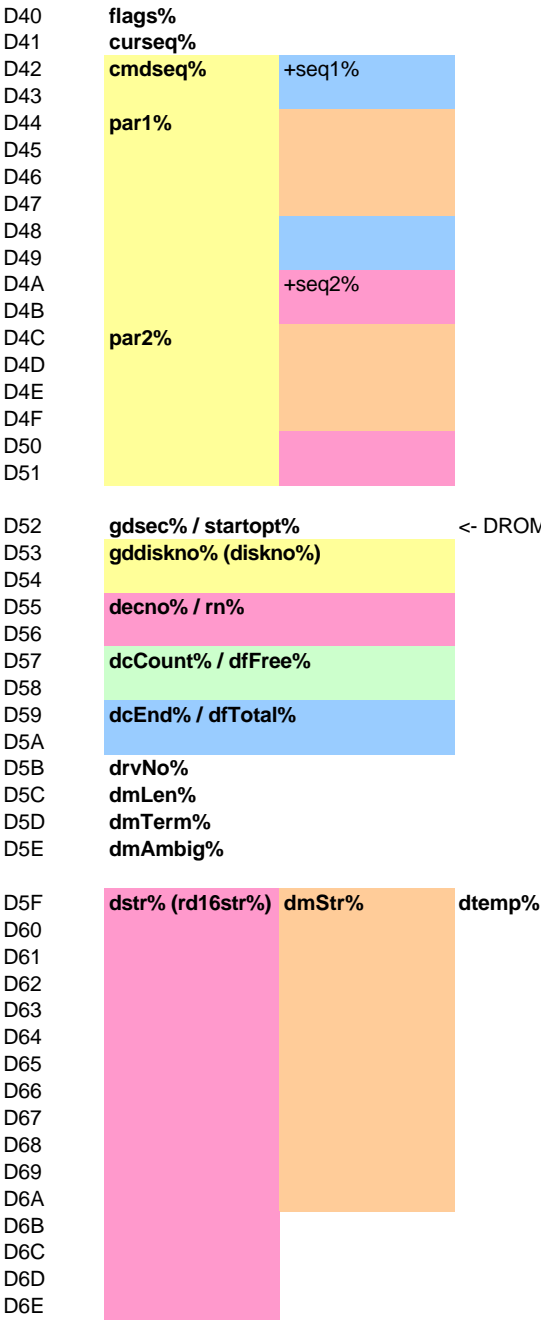only allows single block 512 byte writes (i.e. one MMC sector).

This means that when writing just one DFS 256 byte sector
the other 256 byte DFS sector contained in the MMC 512 byte
sector needs to be read in to a buffer, from which
it will be rewritten with the new sector.

Absolute Memory: NMI Routine (&D00-&D9E)

Addr	Label	Label2	Description
D00	RTI		
D01	**PrevNMI**		
D02	**mmcstatus%**		contains &54 if MMC initialised
D03	**mmcmode%**		
D04	**mmcfirstmode%**		
D05	**sec0set%**		contains &54 if set
D06	**sec0%**		
D07			
D08			
D09			= sec0% XOR &FFFFFF
D0A			(Used as a check)
D0B			
D0C	**drvidx%**		2 bytes for each drive
D0D	lsb's		containing 9 bit disk number
D0E			
D0F			If bit 15 set, then "no disk"
D10	msb's		
D11			
D12			
D13			
D14	lsb's		drvidx XORed with &FF
D15			
D16			
D17			
D18	msb's		
D19			
D1A			
D1B			
D1C	**drvwrite%**		1 byte for each drive
D1D			contains &54 if disk is writable
D1E			
D1F			
D20	**fdcdrv%**		
D21	**swap%**	**mmcstatus%**	COPY OF ALT VARS
D22		**mmcmode%**	FOR USE WITH *DSWAP
D23		**mmcfirstmode%**	
D24		**sec0set%**	Initialised to zero on 'real' boot
D25		**sec0%**	
D26			Var 130 = swapsize% = 31 (&1F)
D27			
D28			
D29			
D2A			
D2B		**drvidx%**	
D2C		lsb's	
D2D			
D2E			
D2F		msb's	
D30			
D31			
D32			
D33		lsb's	
D34			
D35			
D36			
D37		msb's	
D38			
D39			
D3A			
D3B		**drvwrite%**	
D3C			
D3D			
D3E			
D3F		**fdcdrv%**	

Addr	Name		
D40	**flags%**		
D41	**curseq%**		
D42	**cmdseq%**	+seq1%	
D43			
D44	**par1%**		
D45			
D46			
D47			
D48			
D49			
D4A		+seq2%	
D4B			
D4C	**par2%**		
D4D			
D4E			
D4F			
D50			
D51			
D52	**gdsec% / startopt%**	<- DROM code copied here (&1C bytes)	
D53	**gddiskno% (diskno%)**		
D54			
D55	**decno% / rn%**		
D56			
D57	**dcCount% / dfFree%**		
D58			
D59	**dcEnd% / dfTotal%**		
D5A			
D5B	**drvNo%**		
D5C	**dmLen%**		
D5D	**dmTerm%**		
D5E	**dmAmbig%**		
D5F	**dstr% (rd16str%)**	**dmStr%**	**dtemp%**
D60			
D61			
D62			
D63			
D64			
D65			
D66			
D67			
D68			
D69			
D6A			
D6B			
D6C			
D6D			
D6E			

MMB Structure

8 Kb	Disk Table (see below) (16 Sectors x 512 bytes)
200 Kb	First Disk Image (400 sectors x 512 bytes)
200 Kb	Second Disk Image, etc.

Disk Table Structure

16 x 512 byte sectors   **8 kb**

sector 0   +

0	On boot
1	disks
2	(as drive
3	index)
4	
5	
6	
7	
8	Not used
9	
A	
B	
C	
D	
E	
F	

Set using *DONBOOT

511 disk * 16 bytes per disk:
offset = (disk no + 1)*16

sector 0   + offset +

0	Disk title
1	
2	
3	
4	
5	
6	
7	
8	
9	
A	
B	
C	Not used
D	
E	
F	Status

Disk title is updated by
*TITLE, *DRECAT, *DNEW and *DFORM

Status:

1 byte for each disk
values:
00 - Read Only
0F - Read / Write
F0 - Unformatted
FF - Not Valid Disk No.