MICRODRIVX

The psuedo microdrives that are created have a name MICRODRIVX, that the EMUL program looks for when trying to access them. When operating from the button, it is not possible to see the contents of these files using the CAT command.
Also if the MICRODRIVX file is renamed, the the emulator will not be able to find it. However this is useful if you wish to have a number of microdrive 1 psuedo cartridges.

MISSING CARTRIDGES

When the emulator cannot find a pseudo microdrive cartridge file, then the error message returned is **"% FILE NOT FOUND"**, This is a Swifd Disc error as identified by the leading %. It should not be confused with an Interface 1 file not found message which occurs if a file *within* a pseudo cartridge cannot be found, ie no leading %.

STREAMS & CHANNELS

For a description of stream and channels, please refer to the Swift Disc Operating Manual

USEAGE NOTES

Although the emulator software mimics Interface 1 as closely as possible, it is a SOFTWARE emulation, therefore the following 2 points should be noted:

1. Interface 1 has a HARDWARE trap in the middle of the main Spectrum ROM close routine to capture **CLOSE #** on microdrive channels. This cannot be emulated, therefore the allowed syntax of the emulator has been extended to allow **CLOSE#%#**, as already described.

2. Some machine code programs may directly interrogate the interface 1 hardware. If Interface 1 is not connected, then errors of the type **"Microdrive Not Present"** may occurr. There are 2 options that can be taken:-

a). Connect Interface 1 so that the hardware ports exist. Simple if you have Interface 1, but not really a long term solution.

b). Modify the machine code so that direct hardware control is removed, and if necessary, calls to the emulator made instead. Obviously this method requires knowledge of machine code, but provides a long term solution.

As we are made aware of programs that use direct hardware control, we can determine the necessary changes, and will be able to supply users with a list of patches to use. In this way you should be able to convert all your software such that you no longer need to have interface 1 connected.

Finally, if you wish to know more about microdrives, there are several good books widely available from high street stores which describe the workings of interface 1 in depth.

Microdrive Emulator Operations Manual

Author: I. Hunt
Date: 10 October 1987
Software Version: V1.0

Produced on behalf of : Sixword Ltd
26 Church Road
Warsash
Southampton
SO3 6GD
Tel : (0243) 771689

The information contained herin is the property of Sixword Ltd and is supplied without liability for errors or omissions and no part may be reproduced, used or disclosed except as authorised by contract or other written permission. The copyright and the foregoing restriction on reprodiction, use and disclosure extend to all media in which this information may be embodied, including magnetic or electronic storage, punch card, paper tape, computer printout, visual display etc.
Sixword Ltd reserve the right to make any changes to this manual, hardware, firmware and software at any time withou prior notice. In no circumstances will Sixword Ltd be liable for any direct, indirect incidental or consequenial damage or loss of use, data, profit or contracts which may arise from any error, defect or failure of the disc system hardware software or firmware.

Sixword Ltd have a policy of continuous improvement.

Introduction to the Microdrive emulator

The microdrive emulator is supplied on 3.5" diskette. On the diskette you will find two files.

EMUL - The microdrive emulator program its self
COPY - A program that will copy microdrives to psuedo microdrives.

The microdrive emulator is only supported with V3 base software and above

Background on the Emulator

The Microdrive emulator is loaded into the RAM of the Swift Disc Interface. This Loaded portion of the Emulator complements a resident portion that is in the PROM within the interface.
The function of the Microdrive Emulator is to perfectly emulate up to 4 microdrives simultaniously on a single diskette. The emulated microdrives are called **PSUEDO** microdrives and actually exist as files called MICRODRIVEX on the diskette. When using the emulator, you can only use disc drive 0.
The copy program, which copies blocks off a microdrive onto a psuedo microdrive, requires interface 1 to be present. In this case Interface 1 must be plugged into the Spectrum and the Swift Disc Interface must be plugged in behind it.

The emulator perfectly emulates up to 4 microdrives on a single diskette. The RS-232 output port is also emulated, but the RS-232 input port is not.
No atttempt has been made to emulate the network port, and any access to this device will return with an error code.

Hook codes are again fully supported, and the emulation is of version 1 Interface 1. However one of the Network hook codes has been assigned to put the interface in transparent mode, which is used by the copy program to enable interface 1 to be active whilst the emulator is loaded.

For more information on hookcodes for the emulator and the hook codes for the Swift Disc, Please contact Sixword Ltd.

There is no copy protection on the Emulator. It can be backed up or moved to any number of diskettes, and we would advise you to back it up as the first operation. If you are running a pirated copy of the emulator and you like the product, please purchase a genuine copy from us, as this gives us the ability to develop more products for your interface.

SETTING THE EMULATOR TO WORK

To Load the emulator either press the button and load it as if it were an image type file i.e

> L EMUL

or, as with image files it can be loaded from basic.

LOAD %0;"EMUL"

Your Spectrum will reset once the Emulator has been loaded from the button. Loading from Basic does not reset the Spectrum. Your Spectrum will now behave as though Interface 1 were present. Whilest the emulator is loaded, the interrupt button is disabled. The only way to re enable the button is to power off the Spectrum.

It should be noted that a **NEW** command from basic, or a spectrum crash will leave the emulator active, therefore there is no need to reload the emulator unless the reset button is pressed , or the power is turned off

For users unfamiliar with the interface 1 command syntax, the commands are described in the following pages. Also any restrictions or enhancements when using the emulator are detailed.

Note () means optionally included parameters

Command Name: Format

Command Syntax: FORMAT "m";x;"name"

Command Options:

x This is the number of the cartridge that you wish to format , between 1 and 8.
name This is the name of the cartridge. This is a maximum of 10 characters

Description:

The FORMAT command formats a psuedo microdrive on a diskette. This formatting process creates a 130K file called MICRODRIVEX, on disc drive 0. Formatting a psuedo microdrive erases all files that were on it. Up to 4 psuedo microdrive cartridges can exist on a single diskette. The file is only created the first time a particular pseodo cartridge is formatted. Subsequent formatting just resets the header blocks that describe the contents of the pseudo cartridge. Note the **keep** command from the button when applied to a pseudo cartridge causes the cartridge to be protected from formatting by the emulator.

Examples:

```
10 FORMAT "m";1;"md1"
20 REM Create a psuedo cartridge called MICRODRIVE1
30 FORMAT "m";4;"md4"
40 REM Create a second psuedo cartridge
```

Command Name: CAT

Command Syntax: CAT (#Z;)Y

Command Options :

Z is an optional stream number to send the CAT listing to.
Y is the psuedo microdrive number that you wish to CATalogue

Description:

The CAT command allows a listing of the files within a psuedo microdrive. The display shows the name of the cartridge that was assigned at format time, each file in alphabetical order and the amount of free space left on the psuedo microdrive.

Examples:

```
10 CAT 1
20 REM Get a listing of files on microdrive 1
30 CAT #3;1
40 REM Send the CAT listing to stream 3
```

Command Name: LOAD

Command Syntax: LOAD *"m";x;"name"(CODE (start(,len)))
(SCREEN$)
(DATA array)

Command Options:

x This is the number of the psuedo microdrive that you wish to load the file from
name This is the name of the file that you wish to load from the psuedo microdrive
start This is the start address where the code will be loaded. The default is the location where it was saved from.
len This is the length of code that will be loaded. The default is the whole file.
array This is the name of the array into which the data is loaded.

Description:

The LOAD command loads a file from the psuedo microdrive.

Examples:

```
10 LOAD *"m";1;"basicprog"
20 REM load a basic program from the microdrive
30 LOAD *"m";3;"codeload" CODE
40 REM Load a code file.
50 LOAD *"M";2;"dataload"  DATA b$()
60 REM load the data file into b$
```

Command Name: SAVE

Command Syntax: SAVE *"m";x;"name" (CODE start,len)
(SCREEN$)
(DATA array)
(LINE number)

Command Options:

x this is the psuedo microdrive number that you wish to save to.
name This is the name of the file that you wish to save.
start This is the decimal start address for a code save.
len This is the decimal length that you wish to save
array This is the name of an array that contains the data to be saved.
number This is the line number , from which a basic program will autorun when loaded

Description:

The SAVE command saves a block of memory on the psuedo microdrive in a given format.

A program which is saved with a filename RUN on psuedo microdrive1 will autorun if RUN is the first command typed when EMUL is loaded. Note currently the emulator only supports up to 50 files on a single pseudo cartridge, giving a maximum of 200 files on a disk. This restriction may be removed on future versions of the emulator.

Examples:

```
10 SAVE *"m";1;"basicprog" LINE 100
20 REM save a basic program.
30 SAVE *"m";1;"codesave" CODE 20000,1000
40 REM save a code area
50 SAVE *"m";3;"numbers" DATA N()
60 REM save an array of numbers
```

Command Name: VERIFY

Command Syntax: VERIFY *"m";x;"name" (CODE start,len)
(SCREEN$)
(DATA array)

Command options:

x This is the pseudo cartridge you wish to verify with
name This is the name of the file you wish to verify
start This is the decimal start address for a code verify.
len This is the decimal length that you wish to verify
array This is the name of an array that contains the data to be verified.

Description:

The **VERIFY** command compares the contents of the file with the contents of memory. This command is not really required with a disc

system, as extensive error checking is performed by the disc hardware and firmware. The command is provided for compatibility with interface 1 programs.

Examples:

```
10 VERIFY *"m";1;"filename"
20 REM Verify  file with memory
```

Command Name: MERGE

Command Syntax: MERGE *"m";x;"name"

Command Options:

x This is the pseudo cartridge you wist to merge from
name This is the name of the program file you wish to merge

Description:

The **MERGE** command will load the program from the selected drive and merge it with the existing program and variables, replacing any existing ones with those on the pseudo cartridge. It is not possible to merge an auto-run program saved with **LINE xxxx**.

Examples:

```
10 MERGE *"M";1;"program"
20 REM Merge a program in memory with file
```

Command Name: OPEN #

Command Syntax: OPEN #stream;"m";x;"name"

Command Options:

stream This is the stream number that you wish the file to be attached to.
x This is the psuedo microdrive cartridge that the file will exist on.
name This is the name of the file on the psuedo microdrive.

Description:
This command links a stream to a file on a given psuedo microdrive. If the file does not exist then the file is created and the file is opened for write only. If the file exists then the file is opened for read only.

Examples:

```
10 OPEN #4;"m";1;"Printfile"
20 REM  open a file on drive 1 called printfile
30 OPEN #5;"m";4;"print4"
40 REM open a file on drive 4 called print4
```

Command Name: CLOSE # or **CLOSE #%#**
(See Description)

Command Syntax: CLOSE #stream *or* **CLOSE #%#stream**
(See Description)

Command Options:

stream This is the stream number that you wish to close. This is a number between 4 and 15.

Description:
This command closes the stream and empties all data on to the psuedo microdrive.
When using **CLOSE #**, If the stream was not open then the Spectrum will crash. If Interface 1 is present then the Spectrum will not crash as this this error is trapped within the hardware of interface 1 and then intepreted within the emulator.
If Interface 1 is not present, then the extended command **CLOSE #%#** should be used. Existing programs should be converted to this syntax if they are to be used exclusively with the emulator.

Examples:

```
10 CLOSE #5
20 REM Close stream 5 - Interface 1 connected
30 CLOSE #%#4
40 REM  Close stream 4 - No inteface 1
```

Command Name: PRINT # *and* **INPUT #**

Command Syntax: PRINT #stream;string
INPUT #stream;variable

Command Options:

stream This is the stream number that data is sent to.
string This is a "string" or a string variable.
variable This is a string variable.

Description:

The PRINT # command allows data to be directed into a stream and thus into a file. Sequential files are created using a sequence of PRINT # commands.

Examples:

```
10 OPEN #4;"m";1;"seqfile"
20 PRINT #4;"This is the SIXWORD Microdrive Emulator"
30 CLOSE #%#4
```

The above example creates a file and writes the PRINT text to that file and then CLOSES the file. The data in this file can then be read back by the following:-

```
40 OPEN #4;"m";1;"seqfile"
50 INPUT #4;A$
60 PRINT A$
70 CLOSE #%#4
```

Command Name: ERASE

Command Syntax: ERASE "m";x;"name"

Command Options:

x This is the psuedo microdrive cartridge that the file will exist on.
name This is the name of the file on the psuedo microdrive.

Description:
This command erases a file from a given psuedo microdrive. Once a file has been erased, it cannot be recovered without a recovery utility.

Examples:

```
10 ERASE "m";1;"myfile"
```

Command Name: MOVE

Command Syntax:

```
MOVE "m";x;"name" TO #stream
                     "m";y;"name2"
```

Command Options:

x This is the pseudo microdrive number you want to move from
name This is the name of the file you want to move from.
stream This is the stream you want to move the contents of the file to
y This is the pseudo microdrive cartridge you want to move the file to
name2 This is the the filename of the file to be created.

Description:

The **MOVE** command allows the contents of data files to be moved to a channel such as a printer or the screen, or to another file. The **MOVE** command can also be used to append data to an existing data file, by opening a new file, moving the contents of the old file to the new file, then writing the new data onto the end of the new file.

Examples:

```
10 MOVE "m";1;"datafile" TO "m";2;"copyfile"
20 REM Copy a file from one cartridge to another
30 MOVE "m";3;"datafile" TO #2
40 REM List the data on the screen
50 OPEN #4;"m";1;"newfile"
60 MOVE "m";1;"oldfile" TO #4
70 PRINT #4;"Data appended to end of file"
80 CLOSE #%#4
90 REM Append data onto the end of an existing file
```

Command Name: RS-232 Port Handling

Command Syntax:

```
FORMAT "T";baud
           B

OPEN #stream;"T"
             B
```

Command Options:

stream This is the number of the stream that you wish to format
T This is a request for a TEXT channel. All characters are interpreted. Used for listings
B This is the BINARY channel and characters are not intepreted.
baud This is the baud rate of the printer. See Swift Disc Operating Manual for Baud Rates

Description:

To print data to the RS-232 port, the port firstly has to be formatted and then opened.

Example:

```
10 FORMAT "T";1200
20 OPEN #3;"T"
30 LLIST
```

FRAGMENTATION

If you are intending to use the microdrive emulator a great deal, then it is worth formatting the psudo microdrives on a clean diskette. This will increase the speed of operation as the psuedo microdrive will be in contiguous sectors on contiguous tracks. If however you create a psuedo microdrive on a diskette that is already being used , then the speed of access will depreciate considerably.

MICRODRIVX

The psuedo microdrives that are created have a name MICRODRIVX, that the EMUL program looks for when trying to access them. When operating from the button, it is not possible to see the contents of these files using the CAT command.
Also if the MICRODRIVX file is renamed, the the emulator will not be able to find it. However this is useful if you wish to have a number of microdrive 1 psuedo cartridges.

MISSING CARTRIDGES

When the emulator cannot find a pseudo microdrive cartridge file, then the error message returned is **"% FILE NOT FOUND"**, This is a Swifd Disc error as identified by the leading %. It should not be confused with an Interface 1 file not found message which occurs if a file *within* a pseudo cartridge cannot be found, ie no leading %.

STREAMS & CHANNELS

For a description of stream and channels, please refer to the Swift Disc Operating Manual

USEAGE NOTES

Although the emulator software mimics Interface 1 as closely as possible, it is a SOFTWARE emulation, therefore the following 2 points should be noted:

1. Interface 1 has a HARDWARE trap in the middle of the main Spectrum ROM close routine to capture **CLOSE #** on microdrive channels. This cannot be emulated, therefore the allowed syntax of the emulator has been extended to allow **CLOSE#%#**, as already described.

2. Some machine code programs may directly interrogate the interface 1 hardware. If Interface 1 is not connected, then errors of the type **"Microdrive Not Present"** may occurr. There are 2 options that can be taken:-

a). Connect Interface 1 so that the hardware ports exist. Simple if you have Interface 1, but not really a long term solution.

b). Modify the machine code so that direct hardware control is removed, and if necessary, calls to the emulator made instead. Obviously this method requires knowledge of machine code, but provides a long term solution.

As we are made aware of programs that use direct hardware control, we can determine the necessary changes, and will be able to supply users with a list of patches to use. In this way you should be able to convert all your software such that you no longer need to have interface 1 connected.

Finally, if you wish to know more about microdrives, there are several good books widely available from high street stores which describe the workings of interface 1 in depth.