ElkSD64

Acorn Electron SD Card Interface &
64K RAM Expansion

www.ramtop-retro.uk

User Guide

Last updated 10/01/2020

1. Introduction

The ElkSD64 interface board is an expansion card for the Acorn Electron with the following features:

- **Load and Save software using standard SD cards up to 8GB**
- **Connects directly to the Electron's expansion port**
- **Needs no additional hardware (except the Electron of course!)**
- **Expands your Electron to 64K of RAM**
- **Enhanced compatibility, works with almost all Electron games**
- **Uses standard .SSD disk image files**
- **User configurable sideways memory slots; load ROM images to flash or sideways RAM**
- **3D printed protective case**

The purpose of this guide is to explain the inner workings of the ElkSD64 and provide guidance in regard to advanced uses such as directly using sideways ROM and RAM. The guide assumes you have read the installation instructions included with your ElkSD64, the interface is installed and working, and you are familiar with using disk images. A copy of the installation instructions is available at http://ramtop-retro.uk/elksd64.html

2. How the Interface Works

For many 8-bit home micros from the 80s adding support for removable solid state storage is as easy as attaching a floppy drive emulator, like the famous GoTEK, to the floppy drive port.

On Acorn's 8-bit systems floppy support has three components; the floppy drive mechanism itself, the drive controller chip, and a piece of system software called the *Disk Filing System* or DFS. Unfortunately, the Electron's budget origins mean it was shipped without any of these components included. Acorn provided them in the form of the Plus 3 disk peripheral, which is now rare and expensive to obtain.

As we want to use memory cards the floppy drive and controller chip are unnecessary. But we do need the DFS, or rather something that is compatible with it but actually talks to an SD card rather than a floppy drive. Such software exists in the form of MMFS, which is a DFS compatible package designed to work with memory cards.

The ElkSD64 card interfaces a standard SD memory card slot to the Electron and provides a copy of MMFS stored on a flash chip which enables the Electron to communicate with the SD card. It also contains 32KB of RAM,

used to expand the Electron to 64KB of memory via the 'sideways RAM' system. Some of the extra memory is used to enhance compatibility with games (detailed in section 5).

3. Understanding MMFS

MMFS works much like Acorn's DFS, but because a memory card can store potentially hundreds of disk images there are some additional considerations to deal with.

The most common disk image format for Acorn machines is .SSD (Single Sided Disk) format. MMFS uses this format, but rather than having many files scattered on a memory card it bundles them all into a single file called *beeb.mmb*, which is pre-prepared and copied to the memory card.

Each disk image is stored in a logical 'slot', identified by a number. The first disk image is slot 0, the second slot 1, etc.

On the Electron any given disk image can be 'inserted' – made available for use – by typing the command *DIN followed by the slot number. So, typing:

```
*DIN 3
```

will load the *fourth* disk image (fourth because the first slot is zero, not one). To see a list of files on the image you can type:

```
*CAT
```

The normal SAVE and LOAD commands work with MMFS just as they do with a tape drive or DFS disk system. Lots more information on MMFS commands can be found in the MMFS reference at the end of this guide, as well as a link to extensive documentation on the MMFS Wiki pages.

4. Going Sideways

To fully understand how the ElkSD64 and MMFS work, it's important to know about Acorn's 'sideways' memory system.

The Electron uses the 6502 processor, which can only access a total of 64KB of memory at one time. Acorn decided to split this 64KB 'address space' into three areas; 32KB of RAM for storing the screen display and user program code, 16KB for a ROM chip containing the computer's system software (MOS or Machine Operating System in Acorn nomenclature), and the final 16KB for a ROM containing the BBC Basic language.

In the interests of expandability, Acorn gave the Electron the ability to temporarily 'page out' the BBC Basic ROM and replace it with either another ROM chip or 16KB of additional RAM. These are known as 'sideways' ROM and RAM respectively.

A total of 16 sideways pages are supported by the Electron. Pages 8, 9, 10 and 11 are used by the BBC Basic ROM and the Electron's keyboard scanning system. The ElkSD64 adds 32KB of RAM to the Electron, 16KB each in pages 12 and 13, and 32KB of 'ROM' (physically a flash memory chip, not a real ROM) as pages 14 and 15.

The MMFS software is stored in page 14, and a suite of useful tools is stored in page 15. MMFS is critical to the operation of the interface and deleting it would 'brick' the card, so page 14 cannot normally be used for anything else. However, page 15 can be erased and loaded with a user selected ROM image file – see section 7 for details.

(it is worth pointing out here that just because the pages containing BASIC and MMFS cannot be visible to the 6502 processor at the same time does not prevent using the SD card from BASIC. The Electron is smart enough to switch out BASIC during card I/O operations, when access to MMFS is required, and switch BASIC back in afterward. This happens seamlessly.)

Similarly, only one of the two 16KB sideways RAM pages is normally available. On boot up MMFS copies itself from ROM page 14 to RAM page 12. This is done for compatibility reasons (see section 5). The sideways RAM in page 13 is unused and can be accessed by user programs or those games which support sideways RAM.

To summarise, with the ElkSD64 fitted the Electron's sideways page layout looks like this:

15. ROM - RH Plus 1 Utilities
14. ROM - MMFS Bootstrap
13. 16KB Sideways RAM
12. 16KB Sideways RAM (used by MMFS)
11-08. BBC Basic & Electron Keyboard
09-00. Unused

5. Compatibility & Memory

As mentioned previously, at boot time MMFS copies itself from ROM to a sideways RAM page. This is done to address a compatibility issue that crops up when using disk-type storage systems on the Electron.

No program can run only from ROM; it will need some RAM to store data. Acorn's DFS gets the RAM it needs by commandeering a small section of the

Electron's main 32KB memory area. On 32KB Electrons MMFS works the same way, using around 2.5KB of main system RAM.

BBC Basic has a variable called PAGE, which lists the lowest byte of memory available for use by user programs. The higher PAGE is, the less memory is available. On an unexpanded Electron, with just a cassette player, PAGE will be set at byte 3845 ($E00 in hexadecimal). But with a disk system attached PAGE can rise to 6400 ($1900) or even 7424 ($1D00).

This poses no problems with games originally supplied on floppy disk, they expect DFS to use up some memory and make sure not to store anything there. But games written to load from tape will often expect to be able to use all of the Electron's 32KB, including the area reserved for DFS or MMFS.

Many of the Electron game disk images available for download have been converted straight from tape, and will just blindly trample all over the reserved memory area. The practical result is that loading such a game from a memory card will cause MMFS to crash or malfunction due to memory corruption. On a 32KB Electron there is nothing to be done about this other searching for a copy of the game that has been patched to work with PAGE at $1900.

However, there is a variant of MMFS, known as ZEMMFS, which is capable of using sideways RAM instead of main system memory. At boot time a small 'bootstrap' loader copies ZEMMFS from ROM to a sideways RAM page, and that page is used to store all the required data, meaning PAGE remains at $E00 and all the main system memory is available for games to use.

The ElkSD64 uses ZEMMFS to provide maximum compatibility.

6. A ROM of Your Own

There are many Electron ROM image files available on the internet. If you wish to use one the ElkSD64 provides two ways of doing this.

(it is important to note the terminology here; 'ROM' is sometimes sloppily used in retro gaming circles to refer to game code in an image file, no matter what the original storage medium. With the Electron a ROM image is just that, a copy of the information originally distributed in a physical ROM chip. ROM images are distinct from disk images and must be used differently.)

The easiest way is to load the ROM image into sideways RAM. This is temporary, the ROM will disappear when the Electron is switched off.

The first step is to download the Rom image and move it to the Electron via a .SSD disk image – see section 8 for details of how to get your own files into a disk image suitable for MMFS.

Once the file is available to the Electron, use *DIN to load the required disk image and *CAT to check the file is there. Next, type:

```
*SRLOAD "NAME" 13
```

replacing NAME with the name of the ROM image file. This will load the ROM image into sideways page 13. Press Ctrl-Break to reboot the Electron and the ROM will be available for use.

If you wish to play Electron games that were originally supplied on a plug-in cartridge, this is the way to do it. Load the ROM image to a sideways RAM bank. *Do NOT load such games into a sideways ROM bank, as the game may take over the Electron at boot time and prevent any other use whenever the ElkSD64 is connected, effectively bricking the interface card*.

7. Using Flash

If you make regular use of a ROM and want it available all the time without having to load it into sideways RAM, then programming it into the ElkSD64's flash chip is a good solution.

By default, sideways page 15 is used by the RH Plus 1 utility ROM, which provides extra commands to make using sideways memory easier. However, you can replace this with your own choice of ROM image.

To do this, download the flash tools disk image from the link below:

http://ramtop-retro.uk/elksd64.html

and copy your ROM image file(s) to the SSD (see section 8). Include the SSD in a beeb.mmb file. Use *DIN to load the correct disk image and type

```
*FLASH "NAME"
```

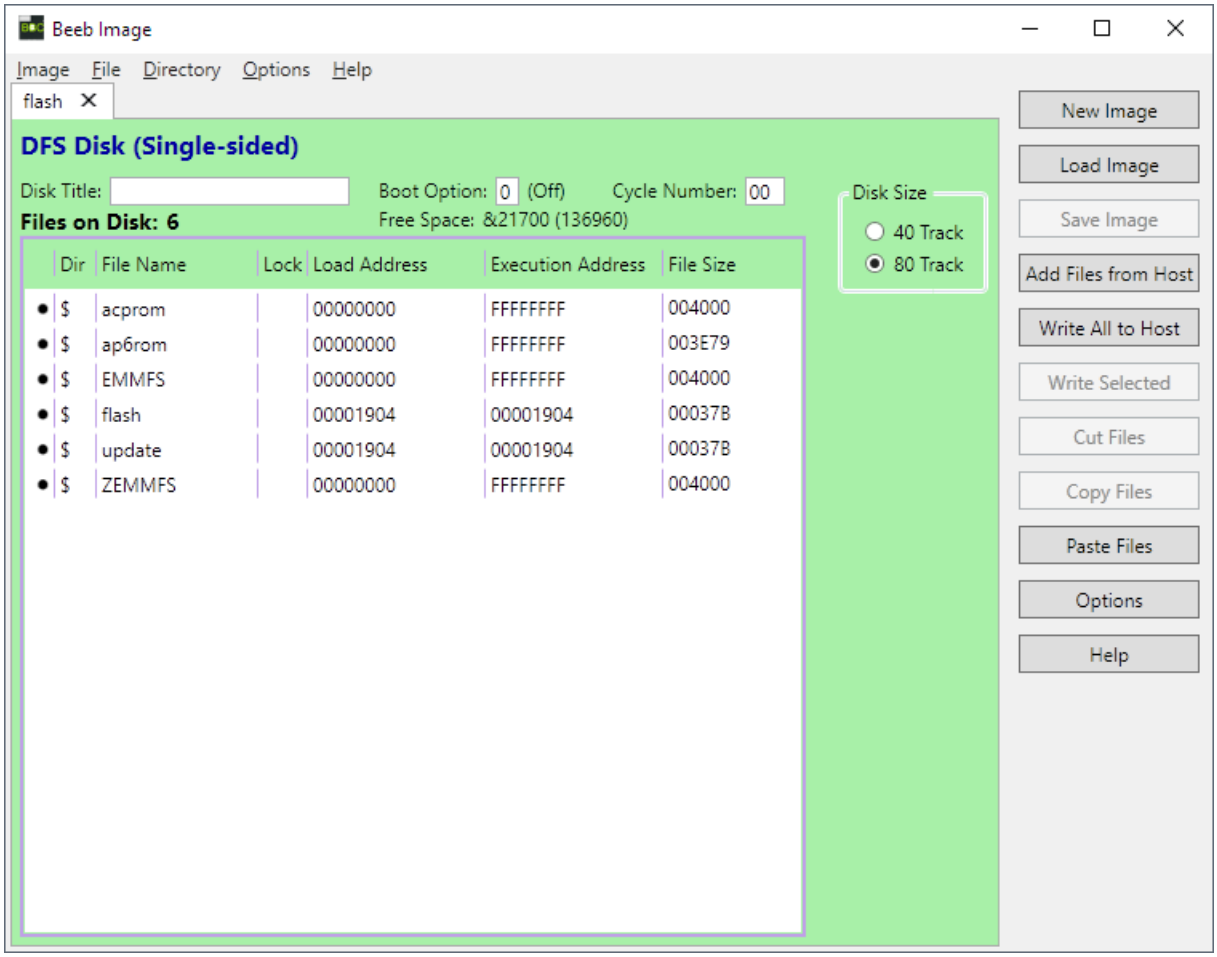
replacing NAME with the name of the ROM image file. The word 'complete' will appear once the flashing process has completed. Press CRTL-BREAK to reboot the Electron.

8. Adding files to a .SSD Disk Image

To add your own files to an existing disk image, a program called Beeb Image is used. You can download a copy here:

http://www.cowsarenotpurple.co.uk/beeb-image.html

Open Beeb Image and click 'Load Image' to select the SSD file. Then click 'Add files from host' and select the files you wish to add. Finally, click the image menu and 'Save Image' to save the modified SSD.



9. Adding SSD images to a beeb.mmb file

To create your own beeb.mmb file or to add .SSD files to an existing one you will need to download the MMBImager software from

https://github.com/dandelion-labs/MMBImager

If you are creating a new beeb.mmb file, click 'File' then 'New Image', and select the root directory of your SD card as the destination. When prompted enter 'beeb' as the filename.

To use a pre-existing beeb.mmb, select 'File' then 'Open Image' and choose the beeb.mmb file you would like to use.

Select a disk image slot in the main window and click 'SSD Image' then 'Load Image' to insert an SSD image in to that slot. When you have loaded all images, click 'File' then 'Close Image'.

Note: on some PCs MMBImager may not run due to an error that references the '**MSCOMCTL.OCX**' file. To fix this problem follow the steps below:

- Download the archive containing MSCOMCTL.OCX from Microsoft by clicking here : https://www.microsoft.com/en-us/download/details.aspx?id=10019
- Run the installer you just downloaded
- If you have a 64-bit Windows system, open a command prompt (CMD.EXE) and type: **copy c:\windows\system32\mscomctl.ocx c:\windows\syswow64**
- If you have a 32-bit Windows system, open a command prompt (CMD.EXE) and type: **copy c:\windows\system32\mscomctl.ocx c:\windows\syswow32**
- Then finally, type : **regsvr32 mscomctl.ocx**
- Reboot the PC

10. MMFS Command Reference

See https://github.com/hoglet67/MMFS/wiki/Command-Reference

11. The Case of the Phantom Joystick

In some games which support analogue joysticks you may experience a 'phantom' joystick that overrides keyboard control. This can be prevented by typing:

```
*UNPLUG F
```

And then resetting the Electron by pressing CTRL and BREAK. This setting will persist until the computer is switched off.

12. Acknowledgements

The ElkSD64 uses the MMFS software, originally developed by Martin Mather and maintained by David Banks. Many thanks to these gentlemen, without their work this device would not have been possible.