



ELECTRON
USER
PORT AND
UPURS
v1.0E

USER GUIDE

User Ports and UPURS serial connection for the Acorn Electron | by Martin B

Contents

Introduction.....	5
What you need	5
The serial port.....	6
Serial Port assignments.....	6
Installing the UPURS suite in the Acorn Electron	7
Notes on using UPURS from disc.....	7
Installing UPURS to Sideways RAM	7
Getting up and running with UPURS and an FTDI USB to RS232 cable	8
Installing the FTDI drivers in Windows.....	8
Uninstalling older FTDI drivers	8
Installing the latest FTDI drivers for Windows	11
Installing the FTDI drivers in Linux	12
PC side software	13
Software for Windows	13
Configuring the settings.....	14
Software for Linux.....	15
Configuring the settings	16
Using UPURS	17
An overview	17
Creating a physical floppy disc on the Acorn Electron from a received disc image.....	18
Creating a disc image from a physical floppy disc.....	20
For Windows.....	20
For Linux	21
Advanced options	22
UPXSSD Error codes	24
Using *UPLOAD.....	25
*UPLOAD <addr>	26
*UPLOAD @	27
*UPLOAD R<id>.....	27
Extracting ROM images using *UPXROM	28
Usage	28
Listing current ROMs.....	28
Exporting a single ROM	28

Exporting multiple ROMs (Advanced users).....	29
UPCFS.....	30
About UEF files.....	30
Decompressing a UEF file.....	30
UPCFS Compatibility	30
Using *UPCFS.....	31
Supported commands with UPCFS.....	32
Loading a program from UEF on an Electron	32
Loading West	32
Loading Cylon Attack.....	32
Reporting issues with UEF files	33
Troubleshooting.....	34
General connectivity issues	34
Resolution	34
Clear to Send responds but data will not transfer correctly	34
Resolution	34
Electron reports “No Rx Data!”	34
Resolution	34
When receiving data, the screen went blank and now the computer is behaving oddly.....	35
Resolution	35
After receiving data, I press BREAK and see a “Bad Sum” error.....	35
Resolution	35
When listing the ROMs in my BBC Micro with UPXROM, I see duplicate entries and the listing differs from *ROMS	35
Resolution	35
When writing to a floppy I see a “Write error” message.	35
Resolution	35
When using UPDSD I see a “Write error” but UPSSD works just fine with the same disc.	36
Resolution	36
UPURS reports “Track No. >79” when I try to write an image to my floppy disc.	36
Resolution	36
My disc image transfers are losing bytes when transferring using UPXSSD or UPXDSD	36
Resolution	36
My disc image transfers are gaining bytes when transferring using UPXSSD or UPXDSD	37

Resolution	37
UPCFS reports “Chunk Type? XXY” when attempting to load a program.....	37
Resolution	37
UPCFS appears to work but then the cassette motor relay clicks	37
Resolution	37
Building the Electron User Port Expansion.....	38
Parts list	38
Optional Parts	38
Assembly of the Electron User Port expansion	39
Fitting the battery components	47
Programming for the Electron User Port	48
Technical differences between the BBC Micro and Electron User Ports	48
Schematics.....	49
Special Thanks	51

Introduction

The UPURS suite provides a set of tools for the transfer of data to and from the BBC Micro series of computers using a high speed serial link and now thanks to the Electron User Port expansion board, UPURS can be used with the Acorn Electron computer too. The combination of User Port expansion hardware, UPURS software, and a USB to RS-232 serial cable means that the implementation of the RS-232 link runs at an impressive 115,200 Baud both to and from the Acorn Electron.

Differing from the BBC Micro slightly, the high speed serial link is implemented directly into the Electron User Ports expansion board. However, the custom cable designed to connect to the BBC Microcomputers' User Port can also be used with the Electron User Port expansion board if preferred.

The accompanying software is designed to import and export single sided disc (ssd) and double sided disc (dsd) images that are used to archive software for the Acorn Electron. Advanced features of UPURS allow data to be loaded directly into the memory of the Electron and for ROM banks to be exported out to a connected computer.

The sections of the guide referring to the Linux operating system were created whilst testing UPURS using the Ubuntu 11.04 distribution (Kernel Version: 2.6.38-8), installed on an x36 (32-bit) PC. This specific Linux distribution and kernel version is not a prerequisite – it simply happens to be the version that the tests were conducted with for the transfer utilities. Other versions of Linux are likely to work just as well.

What you need

To operate the UPURS suite with the Electron, you need the following items:

1. A copy of the UPURS for Electron ROM image on EPROM.
2. An Acorn Electron with an Acorn Plus 1 or similar expansion unit.
3. An Electron User Port expansion board.
4. A PC with a suitable USB to RS-232 serial cable (FTDI chipset based cables are recommended)
5. A computer with MS Windows, Apple OS X, Linux or any other suitable operating system supporting the use of a USB to RS-232 serial cable with FTDI chipset.
6. Suitable PC side software to operate the serial interface.
7. If you wish to use the disc based utilities that UPURS provides, you need a disc interface that is compatible with the Acorn Electron supporting the DFS filing system.

The serial port

When using the UPURS port on the Electron User Port Expansion, the author only recommends USB based serial cables that use the FTDI USB RS-232 chipset.

Drivers for an FTDI chipset based USB to RS-232 cable supporting Windows, Linux and Mac OS X can be found on the FTDI website at the following address: <http://www.ftdichip.com/Drivers/VCP.htm>.

Suitable cables are currently available from Tronisoft Ltd and USBnow at the following address:

[0.9m cable from Tronisoft Ltd](#)

[1.8m cable from USBnow](#)

Serial Port assignments

With a USB to RS-232 serial adapter, the serial port “COM port” assignments are set at the time of installation by the operating system which assigns the first available “COM port” number to the adapter. This could be COM3, COM4, COM5 etc. On Linux, these USB RS232 serial ports are assigned slightly differently and as such, they can be assigned from 0 onwards as the device name differs from legacy serial ports e.g. /dev/ttyUSB0 as opposed to /dev/ttyS0 for a legacy port.

Installing the UPURS suite in the Acorn Electron

The UPURS suite can be used in two different ways on the Acorn Electron. Either as a ROM image loaded into Sideways RAM or as a permanent EPROM installed on one of the Expansion cards plugged into the Electron Plus 1 or directly into a Slogger ROM box in an available Sideways ROM socket.

Notes on using UPURS from disc

The Acorn Electron does not run fast enough to run the UPURS software in RAM and as such, the UPURS suite cannot run from disc.

Installing UPURS to Sideways RAM

To install the UPURS image from the UPURS utility disc into a suitable SRAM bank differs depending on the SRAM utilities each Acorn Electron has.

Please refer to the user manual for your SRAM utilities to find out how to load the ROM image into SRAM.

Getting up and running with UPURS and an FTDI USB to RS232 cable

Installing the FTDI drivers in Windows

If the FTDI drivers for MS Windows noted above are being used and their version number is 2.08.14 or earlier then they must be replaced by later versions of the drivers for reliable operation with the UPURS suite. The latest version of the drivers are identified as version 2.08.24.

The following instructions cover removing any existing FTDI drivers from a Windows PC and then installing version 2.08.24.

Uninstalling older FTDI drivers

Windows Vista and Windows 7

Due to the introduction of “Windows Resource Protection” with Windows Vista, the un-installation of the FTDI drivers must be done manually. The easiest method of un-installing the drivers is with the USB-RS232 cable plugged into the PC at the time and these instructions will cover that method of uninstalling the FTDI drivers.

To uninstall the drivers from Windows, they must be removed using the “Device Manager”. To access the Device Manager, open the Start Menu and right click on the “My Computer” or “Computer” entry and choose the “Manage” option.

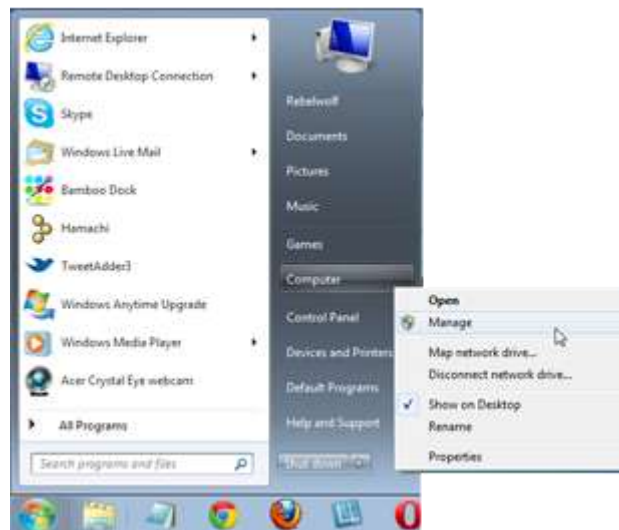


Fig.2. Opening the Computer Management console in Windows Vista and 7

The “Computer Management” console should open and the Device Manager can be found under the “System Tools” section.

Navigate through the device manager and expand the section covering “Ports (COM & LPT)”. If the USB-RS232 cable is plugged in, an entry should be listed as “USB Serial Port (COMx)” where x is the virtual COM port that has been created on the machine when the drivers were originally installed.

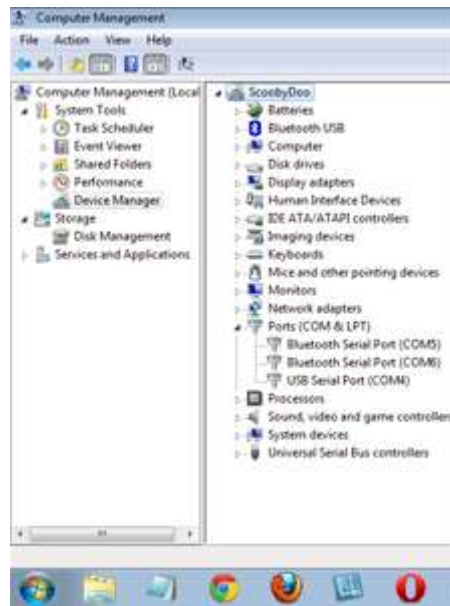


Fig.3 - Navigating through the device manager

To then uninstall the drivers, right click on the “USB Serial Port (COMx)” entry and choose “Uninstall” ensuring that the “Delete the driver software for this device” has been “checked” in the confirmation popup. Once uninstalled, unplug the USB-RS232 cable.

Windows XP

Uninstalling any previous versions of the FTDI drivers under Windows XP is relatively straight forward. A small utility called FTCClean is available from FTDI can be used to uninstall the drivers and clean the related registry entries, thereby ensuring a clean install of the latest drivers.

FTCClean can be downloaded from FTDI at the following address:

<http://www.ftdichip.com/Resources/Utilities/FTCClean.zip>

Once downloaded, the contents of the zip file should be extracted to the desktop. The software once used need not be kept and as such can be deleted after the successful installation of the new drivers.

Once the software is extracted, there should be a new folder on the desktop called FTCClean. Open the folder and double click on the file listed as FTCClean or FTCClean.exe.



Fig.4 - The Windows XP FTDI driver clean up tool

By clicking on the “Clean System” button, the utility will search for and uninstall the FTDI drivers and remove all related registry entries.

Installing the latest FTDI drivers for Windows

Windows Vista and 7 installation procedure

1. Download the drivers.
2. Unzip the driver distribution to "C:\UPURSDrivers\" for example.
3. Open device manager as detailed in the "Uninstall" procedure for Windows Vista and 7
4. Plug in the USB-RS232 cable.
5. Choose "Skip obtaining driver software from Windows Update" in the Driver Software installation popup. (If this does not appear, check the installer icon near the clock in the task bar).
6. In "Device Manager" locate the "USB Serial Port" listed in "Other devices".
7. Right click on the "USB Serial Port" entry and choose "Update Driver Software"
8. Choose to "Browse" for driver software and locate the driver folder "C:\UPURSDrivers\" before clicking "Next"
9. The driver install should then proceed, please choose to accept any changes that the driver installation may require.
10. Once completed, the device "USB Serial Port (COMx)" will be listed in the "Ports (COM & LPT)" section of the device manager and be ready to use. Make a note of the COM port listed as you will require this information later.

Windows XP installation procedure

1. Download the drivers.
2. Unzip the driver distribution to "C:\UPURSDrivers\" for example.
3. Plug in the USB-RS232 cable.
4. In the "Found new Hardware Wizard" choose "No, not this time" when telling Windows to use "Windows Update" and click "Next".
5. At the next screen, choose "Install from a list or a specific location" and click "Next".
6. Choose "Don't search. I will choose the driver to install." and click "Next".
7. In the "Common hardware types" selection, scroll down and choose "Ports (COM & LPT)" and click "Next".
8. Choose "Have Disk" and navigate to where the drivers were unzipped e.g. "C:\UPURSDrivers\"
9. The device to be installed should be listed. Click "Next" followed by "Continue Anyway".
10. Once installed, a second device should be found, to install this second driver repeat the process from Step 4 onwards after which the driver installation should be complete.

Installing the FTDI drivers in Linux

From Kernel version 2.6.31 and later, the FTDI driver is already pre-installed in Linux distributions. If you are using a Linux kernel that predates this version, you will need to install the FTDI driver as a prerequisite step. The driver is available from the *Future Technologies Devices International Ltd.* website, which is listed at the beginning of this guide.

With the FTDI driver properly installed, connect the USB to Serial adaptor cable to a spare USB port on your Linux system. It should be automatically recognised and attached to a Virtual COM port within the operating system.

To confirm the FTDI cable is recognised by your system, type the following command in a *Terminal* window:

```
$ dmesg | grep FTDI
```

The system should respond with an output similar to figure 5.

```
martin@mga-linux:~$ dmesg | grep FTDI
[ 9440.662069] USB Serial support registered for FTDI USB Serial Device
[ 9440.662175] ftdi_sio 6-1:1.0: FTDI USB Serial Device converter detected
[ 9440.665200] usb 6-1: FTDI USB Serial Device converter now attached to ttyUSB0
[ 9440.665231] ftdi_sio: v1.6.0:USB FTDI Serial Converters Driver
martin@mga-linux:~$
```

Fig. 5 - The output from this example confirms that the operating system has detected the FTDI USB Serial adaptor and has assigned it to the Virtual Com Port **ttyUSB0**.

PC side software

Any suitable communications software capable of communicating over an RS-232 connection and logging received data to a file should be suitable for use with the UPURS suite.

Software for Windows

The author currently recommends the Hercules SETUP Utility which is available for free from HW group at the following address.

http://www.hw-group.com/products/hercules/index_en.html

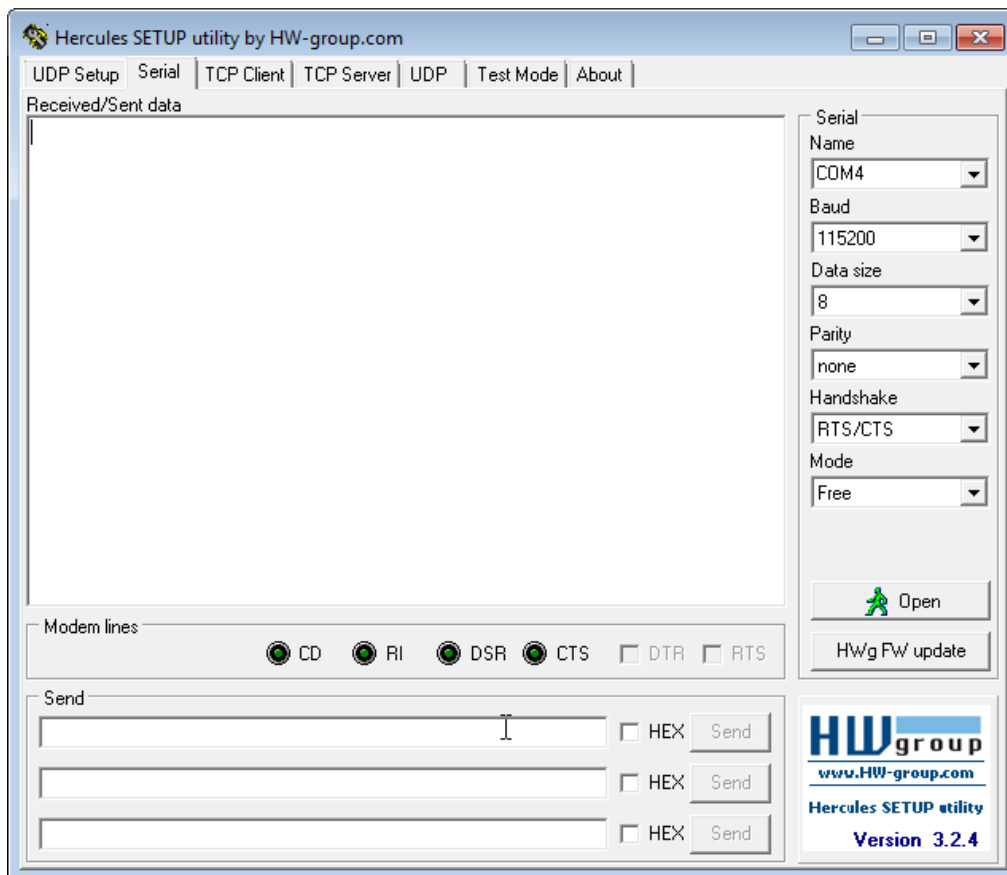


Fig. 6 - The Hercules Serial tab

The Serial tab of the Hercules utility can be used as a simple serial port terminal for the RS-232 standard. It's similar to the Hyperterminal application but with some extended functions.

Parameters :

- Set the parameters of the connection (speed, parity, handshake)
- Set the mode of operation (Free, PortStore test, Data, Setup)
- Set and display the value of the signal on each modem line (i.e. on each RS-232 pin)
- Send and display messages or commands to and from remote device
- Firmware update of device (N/A)

Other parameters (in [Right-click menu](#))

- Display special characters (in ASCII, HEX or DEC)
- Log the communication to a file
- Create and send a test file...

Configuring the settings

Name – Choose the appropriate RS-232 COM port (e.g. **COM4**)

Baud – Set this to 115200 Baud

Data size – Set this to 8 (bits)

Parity – Set this to “none”

Handshake – Set this to RTS/CTS

Mode – Set this to “Free”.

Open | Close button

Open and Close the serial connection to the Electron using the currently active parameters.

When all the correct settings have been selected, click the **Open** button to establish communications between the virtual COM port and the Electron.

Software for Linux

For the purposes of testing the suite with Linux, *Cutecom* was selected as it offers a Graphical User Interface and is very easy to configure. It may not be installed by default as part of your Linux distribution but can be obtained from the usual sources. (If using a Ubuntu distribution for example, the *Ubuntu Software Center* application can be used to locate, download and install it with a few mouse clicks).

Once Cutecom is installed, launch the application with the following command from a Terminal window:

```
$ sudo cutecom
```

You will be prompted for your password for authentication and once entered correctly, the Cutecom application will be opened with “*Super User*” privileges. A screen similar to figure 7 should appear:



Fig. 7 – The Cutecom GUI

Configuring the settings

Confirm the parameters are as shown in the above screenshot, specifically:

Device: Type the appropriate Virtual Com port name (e.g. `/dev/ttyUSB0`)¹

Baud : Set this to 115200 Baud

Data bits: Set this to 8 (bits)

Stop bits: Set this to 1

Parity: Set this to “none”

Handshaking: Tick the **Hardware** checkbox

Open for: Tick the **Reading** and **Writing** checkboxes

Tick the checkbox: **Apply settings when opening**

You should also:

- Uncheck the **Hex Output** box
- Ensure the file type for **Send File** is set to **Plain**
- Select **No Line End**
- Set **Char Delay** to **0ms**.

When all the correct settings have been selected, click the **Open Device** button to establish communications between the virtual COM port and the Electron.

¹ This is the device name assigned by the operating system and should match the device name returned by the `dmesg` command previously issued. (You will need to precede the device name assigned with “/dev/”. You may have to type the device name as the drop-down list may only contain physical RS232 ports found on your system.

Using UPURS

An overview

The UPURS suite has six commands allowing a user to image and create floppy discs and to load data received directly from the User Port into memory or Sideways RAM (SRAM) and also to extract images of Sideways ROM or RAM banks.

In addition to the disc based utilities, UPURS also provides a way of accessing UEF images of data cassettes to load and run programs and data directly from a PC into an Acorn Electron. UPURS accomplishes this feature by providing a pseudo file system that mimics the Electron's native Cassette File System (CFS).

- *UPSSD <drv> Creates a floppy disc with the contents of the SSD image being sent.
- *UPDSD <drv> Creates a floppy disc with the contents of the DSD image being sent.
- *UPXSSD <drv> [F] [I] Exports the image of a physical SSD to the connecting computer.
- *UPXSSD <drv> [I4|I8] Exports the image of a physical SSD to the connecting computer.
- *UPXDSD <drv> [F] [I] Exports the image of a physical DSD to the connecting computer.
- *UPXDSD <drv> [I4|I8] Exports the image of a physical DSD to the connecting computer.
- *UPLOAD <addr> Receives data and places it into the Electrons memory at *addr*.
- *UPLOAD @ Receives data and places it into the Electrons memory at *&0E00*.
- *UPLOAD R<id> Receives data and places it directly into the chosen SRAM bank *id*.
- *UPXROM ? Lists all currently installed ROMs
- *UPXROM <rom_id> [I] Exports the ROM located in the specified ROM bank.
- *UPXROM <low> <high> Exports an inclusive range of ROM banks from low to high.
- *UPCFS Prepares the Electron to accept UEF files by intercepting the CFS

Using the suite should involve a similar process on all connected operating systems, the main differences between interacting with different operating systems will be briefly described, in particular in how this relates to receiving and sending files using the *UPxSD and *UPXxSD utilities.

Creating a physical floppy disc on the Acorn Electron from a received disc image

*UPSSD <drv>

*UPDSD <drv>

There are two commands within the UPURS suite that are designed to create a floppy disc image from the data received over the RS-232 link called UPSSD and UPDSD. The process to send a disc image to the Electron is the same for both ssd and dsd disc images with the exception of the choice of command on the Electron itself.

This manual will only cover the commands used to create a Single Sided floppy disc. To create a double sided floppy disc, simply replace UPSSD with UPDSD in the command below.

To create a floppy disc from the contents of an SSD image on the connected computer, the following command must be issued at the Electron's command line:

*UPSSD <drv> where *drv* can be a value between 0 and 3.²

NOTE: When issuing this command, be careful to ensure that the floppy disc to be written to is pre-formatted. The disc maybe either blank or contain data, however any data on the disc will be overwritten.


```
Acorn Electron 
Slogger PEGASUS 400
BASIC
>*UPSSD 0
Purging link...
Insert disc, send PC file & hit <Space>
Receiving...
Transfer Ended.
>_
```

Fig. 8 - *UPSSD showing the process of receiving data from the connected computer.

Ensure that a floppy is inserted into the correct floppy drive on the Electron and then move across to the connected computer system.

On the connected computer, using the RS-232 terminal software, the serial connection must be "OPENed" if it hasn't been already before choosing the option "Send a File..." or "Send File".

² Where UPDSD is used, the value of *drv* must be between 0 and 1.

When choosing to send a file, a dialogue box will appear, similar to those in figure 4 below, allowing you to select a file to send to the Electron. Navigate to the disk image file that you wish to send and click the **Open** button.

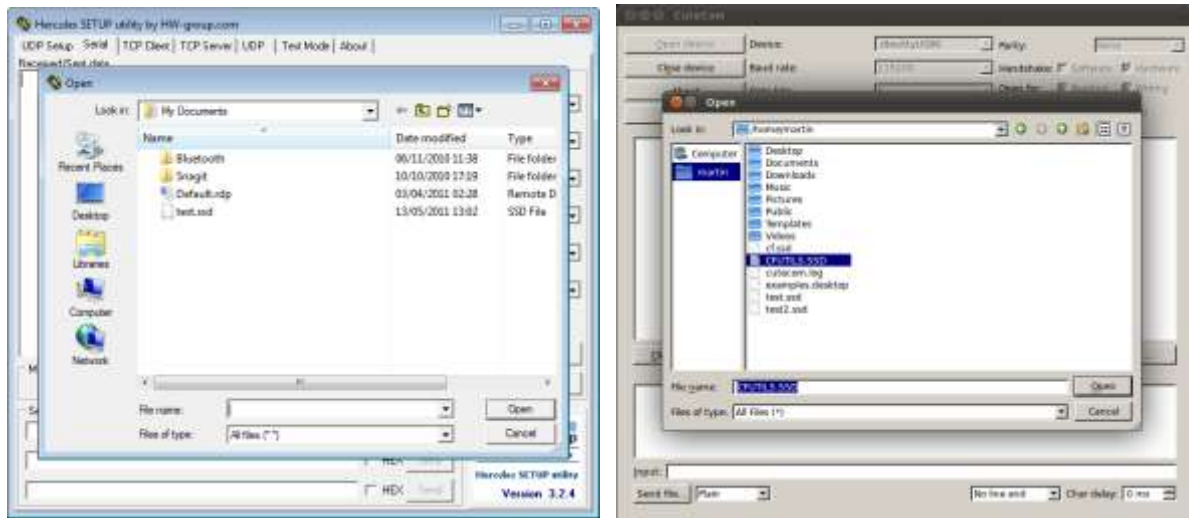


Fig.9 – The File selection screens for both Windows and Linux

Once a file has been selected, in order for the transfer to begin, press <SPACE> on the Electron and the progress of the transfer will be seen on the connecting computer.

When the entire file is transferred, it will be signalled on the Electron with the message “Transfer OK”. The disc should now be ready to use.

Creating a disc image from a physical floppy disc

*UPXSSD <drv> [F] |[I[4|8]]

*UPXDSD <drv> [F] |[I[4|8]]

Creating a disc image of a physical floppy disc on the Electron is achieved using one of two commands, UPXSSD and UPXDSD. The syntax is the same for UPXSSD and UPXDSD and the examples below will work for both commands.

To create a contiguous image of the used area of the source disc, at the command prompt on the Electron type the following command.

*UPXSSD <drv> where *drv* can be a value between 0 and 3.³

The Electron will wait for you to start the process by pressing the [SPACE BAR] before sending data to the connected computer. At this point, **do not** press the [SPACE BAR].

For Windows

Moving to the connected computer, using the data settings for the Hercules software outlined above, choose the “Log to File...” feature from the context menu available on the right click of your mouse. Choose a suitable name for the disc image e.g. test.ssd and click open.

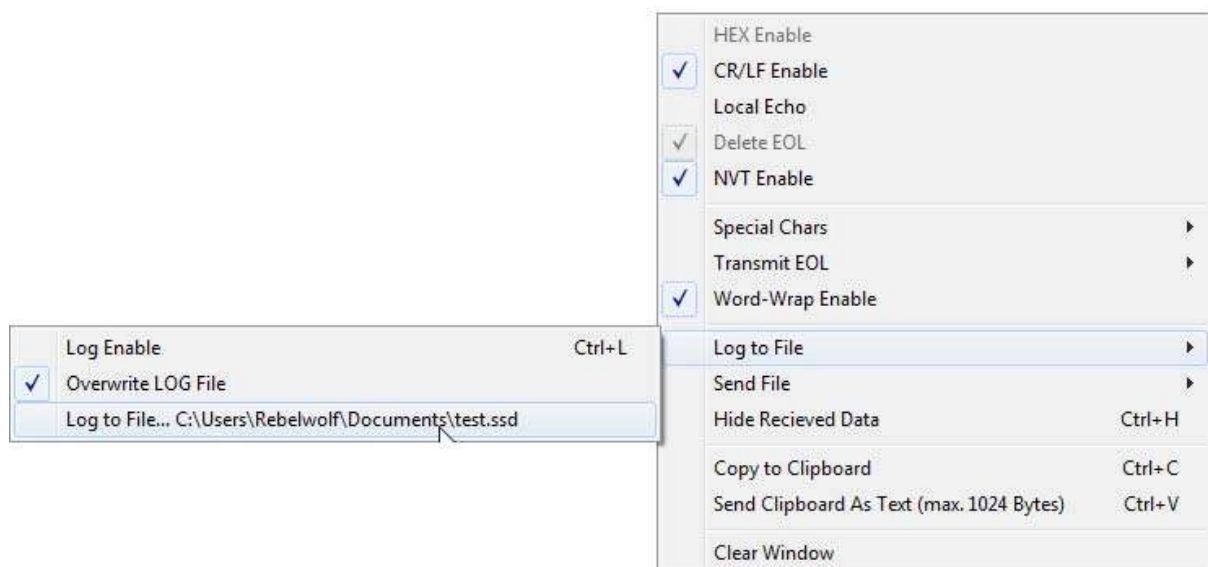


Fig. 10 - Configuring the Log to file options using the Hercules menu

Next, use the key combination of CTRL+L or the context menu option of “Log Enable” to begin logging any data received by the connected computer to the file ‘test.ssd’ then **Open** the Serial connection to the Electron before finally, moving back to the Electron, press the [SPACE BAR] to commence sending data to the connected computer.

³ Where UPXDSD is used, the value of *drv* must be between 0 and 1.

When the Electron completes the sending process, it is important to check that the number of bytes transmitted by the Electron is the same as the number received by the destination PC. Where the number of bytes are different, please refer to the troubleshooting guide for further information.

If the byte counts are the same, use CTRL+L key combination or the “Turn off Logging” option in the context menu to close the received file.

For Linux

On your Linux system, click the Cutecom Open File button (labelled “...”) and a file selection dialogue box will appear similar to figure 5 below. (You may need to uncheck the “Log to:” button to enable this selection).



Fig. 11 - Selecting a destination for the transferred data to be stored

Navigate to where you want to store your received file and enter a suitable filename and click “Save”. Ensure “Log to:” is selected in the drop-down box and tick the box to the left of it. Press the [SPACE BAR] on your Electron to initiate the transfer.

When the Electron has completed transferring the entire disk image, it will be signalled with the message “Image Complete”.

When the transfer is complete, double check the number of bytes sent reported by the Electron with the number of bytes received by the destination PC by checking the file size. If the byte counts are different, please refer to the trouble shooting section for more assistance.

Note: If you are to receive multiple files from the Electron, you will need to uncheck the “Log to:” box and specify a new name for each file you receive using the process described immediately above.

```
Acorn Electron #
Slogger PEGASUS 400
BASIC
>*UPXDSD 0

Insert disc,set PC to Rx,press <Space>
Imaging 62464 bytes...

Finished!

>_
```

Fig 12 - Showing the process of imaging a double sided floppy disc.

Advanced options

UPXSSD and UPXDSD both have several advanced command line options allowing for greater control over how a disc image is created including the recovery of data from corrupt or copy protected discs.

Parameters

No Parameters

Using UPXSSD with no parameters will create a contiguous image of the used area of the source disc as determined by the analysis of the DFS directory held in Track 0. Any read errors will immediately abort the process reporting 'Fatal error' for Track 0 errors or 'Read error' for any other track.

F - Creating a full disc image

The optional parameter “F” is used to force UPXSSD and UPXDSD to create a (F)ull image of the disc in the designated drive. This option may be useful where the directory structure on the disc may not reflect the true contents of the disc. Any read errors will immediately abort the process reporting 'Fatal error' for Track 0 errors or 'Read error' for any other track.

NOTE: The use of “F” is recommended when archiving original software unless you are confident that the disc has no 'hidden' data and the directory listing is a true reflection of the contents of the original disc.

I, I4 and I8 - Creating a full disc image from a corrupt or irregular floppy disc

The parameters "I", "I4" and "I8" are used to perform disc imaging, (I)gnoring any errors that may be encountered during the read process. This allows a disc with missing, out of sequence or invalid tracks to be imaged.

The use of "I" is the equivalent of using UPXSSD without any parameters producing a contiguous image of the used area of the floppy disc as determined by the inspection of a valid Track 0 directory list, "I4" and "I8" implicitly uses the "F" parameter thereby imaging an entire 40 or 80 track disc respectively.

If a read error is encountered when attempting to read Track 0 with option "I", the imaging process will fail whereas, "I4" and "I8" will image the entire disc regardless of the presence of a corrupt track 0.

Where a track cannot be read, valid track data will be sent to the receiving PC but will contain a contiguous block of bytes of the value &FA.

```
Acorn Electron  $\Delta$ 
Slogger PEGASUS 400
BASIC
>*UPXDSD 0 I

Insert disc,set PC to Rx,press <Space>
Imaging 62464 bytes...
06.8 0B.8
Finished!

>_
```

Fig. 13 - Use of the I parameter on a floppy disc with faulty tracks.

When UPXSSD encounters a fault on the floppy disc it is reading, if the "I", "I4" or "I8" parameter are used, the error is reported but the imaging process continues regardless in an effort to recover as much data from the disc as is possible. Errors are reported to the user in the format of "Track.Error" so "02.C" is reporting that track 2 has the error code "C" which represents "Sector Not Found".

When using UPXDSD, the errors are differentiated to provide a distinction between the two sides of the disc using the ‘.’ and ‘:’ characters. Errors reported with a ‘.’ are on the primary disc surface (i.e. 0 or 1), errors reported with a ‘:’ indicate an error in the secondary surface (i.e. 2 or 3)

For example, the error list “**05.C 05:C 1D:C 1E:C**” reports errors on track 5 on both primary and secondary surfaces and tracks 1D and 1E on the secondary surface.

UPXSSD Error codes

UPXSSD returns the actual floppy disc controller error code extracted from the fault byte. These codes differ from the Acorn DFS error codes and are as follows:

- 4 - Clock Error
- 6 - Sector ID CRC Error
- 7 - Data CRC Error
- 8 - Drive Not Ready
- C - Sector Not Found

Using *UPLOAD

*UPLOAD <addr>

*UPLOAD @

*UPLOAD R<id>

The *UPLOAD function is an advanced feature of the UPURS suite allowing the direct placing of data received from the User Port at any point in the Electron's memory map or the direct loading of a ROM image directly into a Sideways RAM bank.

The usage of *UPLOAD is identical for both Windows and Linux and only the file selection process differs between platforms both of which have been described in the section covering the use of the *UPxSD commands.

The process for *UPLOAD is similar to that of the *UPxSD commands except, where those commands write the received data directly to a floppy disc, *UPLOAD writes data to an area of RAM.

*UPLOAD <addr> and *UPLOAD @ do not support the writing of data across the TUBE to the Second Processor memory space.

NOTE: When using *UPLOAD, please make a careful note of the size of the file that is being sent to the Electron. If the file is too large to fit into the Electron's memory, the data may overwrite critical areas of the Electron's memory map and possibly even wrap around to &0000 causing system settings to be overwritten resulting in significant data corruption and requiring the Electron to be powered off and on again in order to restore normal function.

The maximum size of file that can be safely received is therefore dependent on the values of PAGE and HIMEM or if writing data directly to the screen, the values of PAGE and &7FFF.

***UPLOAD <addr>**

To load an image or file directly into an address within the Electron's memory, you can issue the *UPLOAD command with a hexadecimal address of the beginning of the memory space you wish to upload the data to.

For example to *UPLOAD a data file into MODE 2 screen memory, you can issue the commands:

MODE 2

*UPLOAD 3000

NB: The example screenshots below use MODE 2 with a graphics and text window defined in order to show the data loading process without overwriting the image data with output from *UPLOAD.

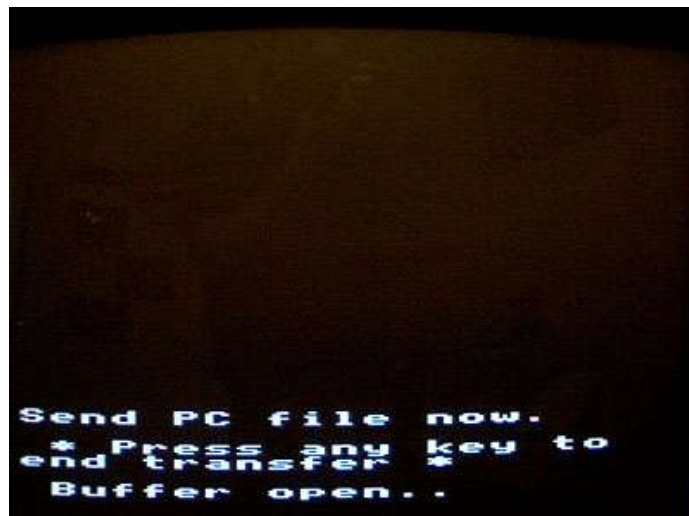


Fig. 14 - *UPLOAD 3000 command issued, waiting for data.



Fig. 15 - Data received and loaded between the values of &3000 and &6BFF.

NOTE: The address value passed to *UPLOAD must always be the address of a page boundary within the Electron's memory so must always be of the format &xx00. For example &0E00 and &3000 are both okay but &0E20 and &3040 cannot be used.

*UPLOAD @

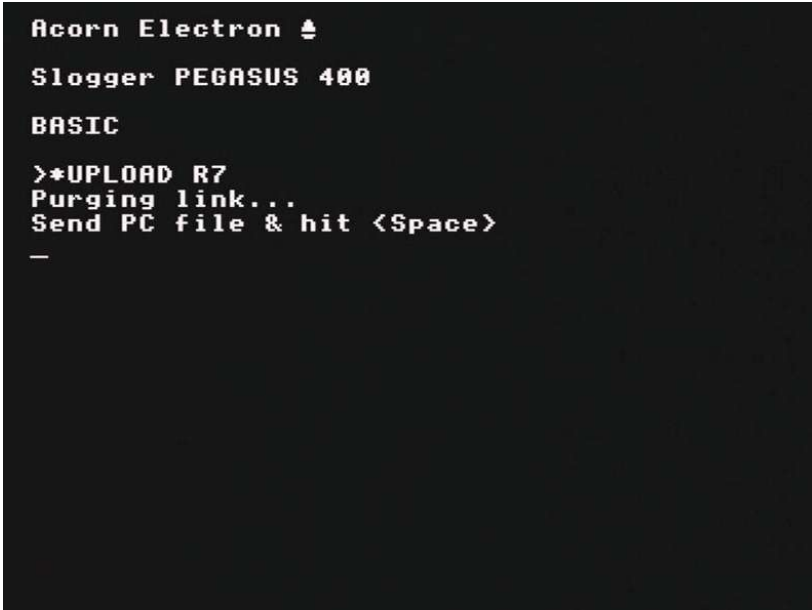
*UPLOAD @ is similar to *UPLOAD <addr> except it is fixed to the address of &0E00, allowing images of programs to be uploaded into memory on Electron's with a page of &0E00 which is common for DFS based file systems.

*UPLOAD R<id>

To load a ROM image directly into Sideways RAM begins by issuing the command:

```
*UPLOAD R<id>
```

The value of *id* must be between the hexadecimal values of 0 and &F and be a valid bank of Sideways RAM.




```
Acorn Electron   
Slogger PEGASUS 400  
BASIC  
>*UPLOAD R7  
Purging link...  
Send PC file & hit <Space>  
—
```

Fig. 16 - *UPLOAD awaiting data from the connected computer.

When you see the “Buffer open...” message, move to the connected computer and ensure that the Serial Port has been **OPENED** before then choosing the “Send a File...” option and browse to the ROM image file that you wish to load into Sideways RAM on the Electron.

Following the selection of the file, press the [SPACE BAR] on the Electron to start the transfer. The sending PC should indicate the progress with the software's “progress bar”. On success, the Electron should state “Transfer OK” and return to a prompt.

Once the transfer into a sideways RAM bank has been completed, use CTRL-BREAK to reset the Electron and initialise the newly loaded ROM image and type *HELP at the prompt to confirm that the ROM image has been loaded successfully.

NOTE: Uploading a ROM image to any other SRAM bank where it is already in use will result in the original ROM image in the designated bank being overwritten. *UPLOAD does not check to see if you are overwriting the SRAM bank that UPURS is currently occupying. If you attempt to *UPLOAD an alternate ROM image to the same bank that UPURS currently occupies, the transfer will hang and the SRAM bank will be corrupted.

Extracting ROM images using *UPXROM

Usage

*UPXROM ?

*UPXROM <rom_id> [\]

*UPXROM <rom_id_low> <rom_id_high>

*UPXROM supports the export of ROM images from designated ROM banks or sideways RAM banks. Individual ROMs can be exported as 8K or 16K files by use of the attribute “\” which signifies that the image should be written out as an 8K file. The banks to be exported are indicated by the *rom_id* which must be a hexadecimal numeric value between 0 and F.

Listing current ROMs

To list the current ROM images that are installed in the Electron, issue the command *UPXROM with the ‘?’ switch. The resulting list will identify the ROM/RAM banks in use and the name of the ROM image that is occupying each bank as shown in the listing below.

Banks that are empty are indicated by the use of the ‘?’, banks that are in use have the name of the ROM image listed, SRAM banks are indicated by the use of ‘w’ rather than ‘.’.

```
Slogger PEGASUS 400
BASIC
>*UPXROM ?
0 : PEGASUS 400 1.02
1 : T2PEG400 2.02
2 : UPURS 1.0E
3 : UPURS 1.0E
4 : ?
5 : ?
6 : ?
7 : ?
8 : ?
9 : ?
A : BASIC
B : BASIC
C : Electron Expansion 2.02
D : ?
E : ?
F : ?
>_
```

NOTE: On the Electron, the ROM slots 8 and 9 are always occupied by the keyboard and BASIC always occupies slots 10 and 11 (&0A and &0B).

Exporting a single ROM

When using *UPXROM, the process is similar to that when using the *UPXSSD and *UPXDSD and the same procedure should be used when choosing the file to which the ROM images should be saved on the connecting computer when preparing to receive the data from the Electron.

To export an 8K ROM (e.g. NFS 3.60) which resides in the sideways ROM bank 7 would use the following command:

```
*UPXROM 7 \
```

To export a 16K ROM (e.g. DNFS 1.20) which resides in the sideways ROM bank 11 would use the following command:

```
*UPXROM B
```

Once the connecting computer is ready to receive the ROM image(s), press the [SPACE BAR] to commence sending the ROM images.

Exporting multiple ROMs (Advanced users)

*UPXROM can be used to export multiple ROMs to a single file where all ROMs are exported out as 16K images regardless of their actual size and the use of “\” is ignored. The resulting exported file will be 16K * <number of roms exported> in size.

A file should be created on the connecting computer to which all the data sent from the Electron should be saved and the instructions for this can be found in the section covering *UPXSSD and *UPXDSD.

To export multiple ROMs, the following command line is used:

```
*UPXROM <rom_id_low> <rom_id_high>
```

The low and high ROM ids form a range signifying the beginning and end ROM banks to export.

Example: *UPXROM 1 A

This command will export **ALL ROM banks** from bank 1 to &A inclusive. **Including those ROM banks that are currently unoccupied.**

After issuing the command on the Electron, ensure that the connecting computer is ready to receive the data and logging is enabled before finally pressing the [SPACE BAR] on the Electron to initiate the send function.

A set of ROM banks exported in this way will require manual extraction of individual ROM images from the single contiguous file using a Hex editor such as HxD which has a file splitting tool built in or XVI32 which allows a selected block of data to be written directly to a file.

When extracting ROMs from a contiguous file in this way, each exported ROM bank starts at an offset of &4000 from the beginning of the file i.e. The first bank exported will be at address &0000, the second bank in the exported file will be at &4000, the third at &8000, fourth at &B000 etc.

REMEMBER: All ROMs are exported as 16K (&4000 byte) blocks regardless of their size which could be 8K (&2000 bytes) or 16K (&4000 bytes).

UPCFS

The use of UEF files with UPURS is intended to allow users to send UEF encapsulated data as contained on the original audio data cassettes that were produced in their thousands during the 1980's and 90's to the Electron for the purposes of "direct play".

The speeds at which UPCFS operate are far in excess of the original Acorn Cassette Filing System which allows users to read data as it was stored on cassette but at the speeds of a modern day system.

No more waiting 15 minutes for a cassette based game to load!

About UEF files

UEF stands for "Unified Emulator Format" and was originally developed to provide a way of emulating the original data storage formats for use with Emulator applications that recreate Acorn computer environments such as the Electron and BBC Micro on more modern PC technology.

The UEF format is a container format for the compressed storage of audio tapes, ROMs and floppy discs. The format can also be used to store machine state snapshots and the intention of the UEF format is to provide an accurate archive of the original media from which a program or data is taken.

When creating UEF files they are traditionally compressed using the gzip file compression algorithms before being packaged for download on the Internet as a Zip file.

UPCFS does not support compressed UEF files and they must be de-compressed before transferring them to an Electron.

Decompressing a UEF file

UEF files that can be found on the Internet are usually packaged as a Zip file. The UEF is then extracted and used with an emulator as it is. For use with UPCFS, a second decompression stage must be carried out. Tools such as Winzip can be used to accomplish this second decompression stage.

The process of preparing a UEF file for use with UPCFS is as follows.

1. Download the UEF zip file package from the Internet.
2. Extract the UEF **filename.uef** from the zip package.
3. Rename the extracted UEF file to **filename.uef.gz** by adding the **.gz** file extension.
4. Extract the gzip'd uef **filename.uef** from the **.gz** file.

Once the file is extracted and decompressed, it is ready for use with UPCFS.

UPCFS Compatibility

UPCFS aims to be as compatible as possible with the original Acorn CFS system however there are several programs that employ copy protection schemes or use particular techniques that can interfere with the UPCFS system rendering them incompatible.

At the time of writing, UPCFS for the Electron has not been fully tuned so compatibility may not be as good as it will be with future releases of the UPURS ROM image

Many of the failures are experienced when UPCFS is overridden by custom software loaders that re-engage the Acorn CFS and then attempt to load the next stage of the program. Typically this behaviour results in the Cassette Motor relay being enabled.

Using *UPCFS

*UPCFS is different in operation to the other commands in the UPURS suite in that it is not in itself a command used to transfer data from the PC to the Electron. Instead, issuing the command *UPCFS configures the Electron in such a way that the traditional Cassette Filing System (CFS) commands are intercepted by UPURS and the Electron believes it is accepting data directly from the build in CFS.

When the *UPCFS command is issued, the Electron will initialise its memory settings and then pass control back to the user at the prompt.

```
Acorn Electron 🏠
Slogger PEGASUS 400
BASIC
>*UPCFS
UPURS UEF FS
Ver 1.0E 251112
>
```

The *UPCFS system will then recognise and intercept the standard suite of commands that are used by the CFS to load and run programs and data.

Supported commands with UPCFS

The UPCFS system is **read only** and supports the following commands.

- *RUN
- *LOAD
- */
- *CAT
- *OPT
- CHAIN
- LOAD
- OPENIN
- OPENUP (read only)
- BGET#
- CLOSE#
- INPUT#
- EOF#

Loading a program from UEF on an Electron

The loading of any particular program from a UEF is entirely dependent on the type of program that is contained by that UEF and often, the UEF filename may give a hint as to which command you should use to load the application.

e.g.

- West_RUN_E.uef uses *RUN to load the game WEST
- CylonAttack_E.uef uses CHAIN"" to load the game Cylon Attack.

The initial setup for transferring files from the PC to the Electron is the same as for other commands within the UPURS suite

Loading West

1. On the Electron, type *UPCFS to configure the UPURS suite.
2. On the PC, open the correct Serial Port to prepare to send the UEF file.
3. Select your decompressed UEF for WEST - **West_RUN_E.uef**.
4. The PC should then wait before sending the UEF file to the Electron.
5. On the Electron type *RUN followed by RETURN to initiate the loading sequence for WEST.
6. Play WEST.

Loading Cylon Attack

1. On the Electron type *UPCFS to configure the UPURS suite.
2. On the PC, open the correct Serial Port to prepare to send the UEF file.
3. Select your decompressed UEF for Cylon Attack – **CylonAttack_E.uef**.
4. The PC should then wait before sending the UEF file to the Electron.
5. On the Electron type CHAIN"" followed by RETURN to initiate the loading sequence for Cylon Attack.
6. Play Cylon Attack.

Reporting issues with UEF files

If you encounter repeatable, issues with a UEF file, you can submit your findings at the following address: <http://www.retro-kit.co.uk/UPURS/Report/>.

Each report is logged and imported into our tracking facilities for investigation so the more information you can provide, the easier it will be to answer your query.

Troubleshooting

General connectivity issues

Using Hercules on a Windows PC, open the COM port and then move to the Electron. Press the <BREAK> key then at the prompt enter:

```
?&FCB3=&7D <RETURN>
```

```
?&FCB1=&40 <RETURN>
```

The Clear to Send (CTS) indicator in Hercules should turn Green. Continue the diagnosis by entering:

```
?&FCB1=0 <RETURN>
```

The CTS indicator should go off again.

Resolution

If the CTS indicator does not respond as described, power off the Electron, re-check all connections, power on and try again.

If the CTS indicator responds correctly, the cable and connection should be operational so try to transfer data using one of the UPURS commands. If the CTS indicator continues not to respond after several attempts, please contact the author via <http://www.stardot.org.uk/forums>.

Clear to Send responds but data will not transfer correctly

If after carrying out the troubleshooting steps above, the CTS indicator responds correctly but data does not transfer correctly, this may indicate that the connecting computers serial port strictly implements the DTR/DSR part of the RS-232 communications protocol.

Resolution

Check the DTR box in Hercules. The Data Set Ready (DSR) indicator should “light up”. Attempt carrying out a data transfer using one of the UPURS commands.

Electron reports “No Rx Data!”

This is the Electron reporting that after issuing a Clear To Send (CTS) signal to the connecting computer, it did not receive a start bit for the first byte of data. If the connecting computer were to become disconnected from the Electron or the send was not initiated on the connecting computer before pressing the [SPACE BAR] on the Electron this error would be thrown.

Resolution

Power off the Electron and re-check all of the connections before powering on again, ensuring that all connections are good.

Ensure that the transfer procedure is done in the correct order. For more details, see the section covering *UPSSD and *UPDSD.

When receiving data, the screen went blank and now the computer is behaving oddly.

This can occur when a buffer overrun occurs where the Electron's memory is overwritten with NULL data thereby setting the system flags incorrectly. The buffer overrun can be triggered if the software on the connecting computer does not reset the RS-232 Tx Data line and it is left "high" rather than the correct idle state of "low".

Resolution

Power off the Electron and wait for a few seconds before turning it back on again; when power is restored, the Electron should behave correctly.

After receiving data, I press BREAK and see a "Bad Sum" error.

This can occur to the Electron when the value of PAGE on the computer is higher than &0E00. UPURS uses memory above &0E00 and this can be allocated as filing system workspace on some computers so when the BREAK key is pressed and the ROMs are initialised, they can report this error due to the fact that UPURS has corrupted their workspace.

Resolution

If you cannot lower the value of PAGE to &0E00 or below by disabling some ROMs, when you see the "Bad Sum" error, press CTRL+BREAK to hard reset your Electron.

When listing the ROMs in my BBC Micro with UPXROM, I see duplicate entries and the listing differs from *ROMS

This occurs in Electrons that do not have ROM expansion boards fitted and the ROM inspection code sees "ghosts" of ROMs it has previously encountered due to the way in which the Electron manages the ROM select lines in hardware.

Resolution

In general, the highest priority version of any particular ROM that is listed more than once is the actual ROM and the subsequent listings are ghosts and can be ignored.

When writing to a floppy I see a "Write error" message.

This occurs when a disc has not been formatted or there are corruptions on the disc.

Resolution

Format and verify the disc ensuring that both sides of the disc have been formatted. If the disc does not verify, try to write the same image using a known good disc. If writing the image to a known good disc fails, attempt to write a different image. If this fails, contact the author for further support.

When using UPDSD I see a “Write error” but UPSSD works just fine with the same disc.

This occurs when the second side of the disc has not been formatted. The double sided disc images are interleaved and as such, tracks are alternately written to the floppy disc 3 tracks at a time. When UPDSD attempts to write the first 3 tracks to the second side of the disc, it cannot and throws this error.

Resolution

Ensure both sides of the floppy disc to be written to have been correctly formatted before attempting to transfer data to the disc again.

UPURS reports “Track No. >79” when I try to write an image to my floppy disc.

This error occurs when the track count of an ssd or dsd image exceeds 80 tracks (0 to 79).

Resolution

Check that the SSD or DSD does not exceed 200KB or 400KB. If the image is larger than these maximum file sizes, the image is invalid and cannot be imported by UPURS.

My disc image transfers are losing bytes when transferring using UPXSSD or UPXDSD

There are two different scenarios where this can take place.

The first situation where this error can occur is if the transfer procedure is carried out out-of-sequence. For instance, pressing the space bar to initiate a send from the Electron before the PC has been configured to receive data and then subsequently opening the PC’s COM port. When the COM port is opened, any data on the COM port already will be flushed before the PC then issues an RTS command to receive data from the Electron. Typically, the data loss is at the beginning of the disc image file that is created.

The other situation where this error can occur when the receiving PC is running a version of the MS Windows operating system **AND** using an Intel chipset based motherboard running a variant of the 82801 Southbridge I/O controller hub **AND** the machine is under a high I/O or CPU load. Data loss can occur at any point through the transfer and appears random.

Resolution

To avoid data loss in the first situation, simply follow the correct procedure as this manual sets out.

In the second situation, when using UPURS with an Intel 82801 Southbridge I/O controller hub, it is essential that your PC is under as little load as possible. Perfect transfers are achievable if the I/O and CPU load are reduced to a minimum. To do this, simply close as many applications as possible and limit network and disk usage whilst transferring data to the PC.

My disc image transfers are gaining bytes when transferring using UPXSSD or UPXDSD

This issue can occur when using any terminal programs to either send or receive data with file logging enabled. In send mode, the terminal won't differentiate between typed characters and a spooled file and in receive or log mode, the log will capture all communications activity between the two devices whether sent or received and whether typed or via a file.

Resolution

When performing send or receive with terminal communications it is important to do nothing other than the required actions whilst a log (at either side) is in progress.

UPCFS reports “Chunk Type? XXYY” when attempting to load a program.

This issue can occur of a several different reasons. When using Serial Terminal software such as Hercules on the PC, every key press that is made whilst data is being transferred is incorporated into the data transfer process. This means you can intentionally cause a “Chunk Type?” error simply by pressing keys on the PC keyboard whilst a transfer is in progress.

The second reason for a “Chunk Type?” error is when UPCFS encounters a UEF chunk of a type that it was not expecting. The byte value in the error response from UPCFS indicates the value of the two bytes that describe the UEF Chunk type.

The third reason for the “Chunk Type?” error is usually due to the software that is being loaded using a cassette format that is encoded into the UEF. These custom encodings are usually down to copy protection methods that have been employed.

Resolution

Attempt to re-load the UEF in question. If the UEF loads without issue then it's likely that there was extraneous data on the serial port when the transfer began or a key was pressed on the keyboard during transfer.

Some software titles fail on the same block with the same “Chunk Type?” error and byte value. It's likely that the UEF is of a custom format or has copy protection techniques that are incompatible with the current version of UPCFS. Please report these errors using the online error reporting form:

<http://www.retro-kit.co.uk/UPCFS/Report/>

UPCFS appears to work but then the cassette motor relay clicks

Some software titles reset the system vectors that UPCFS intercepts to enable loading UEFs over the User Port causing the system to revert to the true Cassette Filing System. When a custom loader does this, and attempts to load the next section of the title, it engages the Electron's internal systems and the Cassette Motor relay engages.

Resolution

Titles that do this are currently incompatible with UPCFS. Please report the titles using the online error reporting form:

<http://www.retro-kit.co.uk/UPCFS/Report/>

Building the Electron User Port Expansion

Parts list

The parts list below details all of the components required to build the entire User Port Expansion board. Depending on the functionality required, some parts are optional.

PCB ID	Part	Quantity
IC1	6522	1
IC2	74LS30	1
IC3	74LS04N	1
IC4	74LS02N	1
IC5	74LS08N	1
IC6	74LS393	1
IC7	EPROM/SRAM 27C128 or 27C256 or 62256	1
IC8	Spare – Not fitted	0
D1 - D2	1N4148	2
D3	1N5817	1
D4	BAT42	1
R1	220R	1
R2 - R4	4k7	3
C1	220pF	1
C2 - C9	100nF	8
B1	CR2032	1
B1h	Holder for CR2032	1
JP1 - JP3	Jumper header 3-way 0.1"	3
PL1 - PL2	R/A IDC 20-way	2
PL3	R/A 9-way D-type	1
IC1s	DIL IC socket 40-way	1
IC2s - IC6s	DIL IC socket 14-way	5
IC7s	DIL IC socket 28-way	1
IC8s	Spare – not fitted	0

Optional Parts

Battery backup

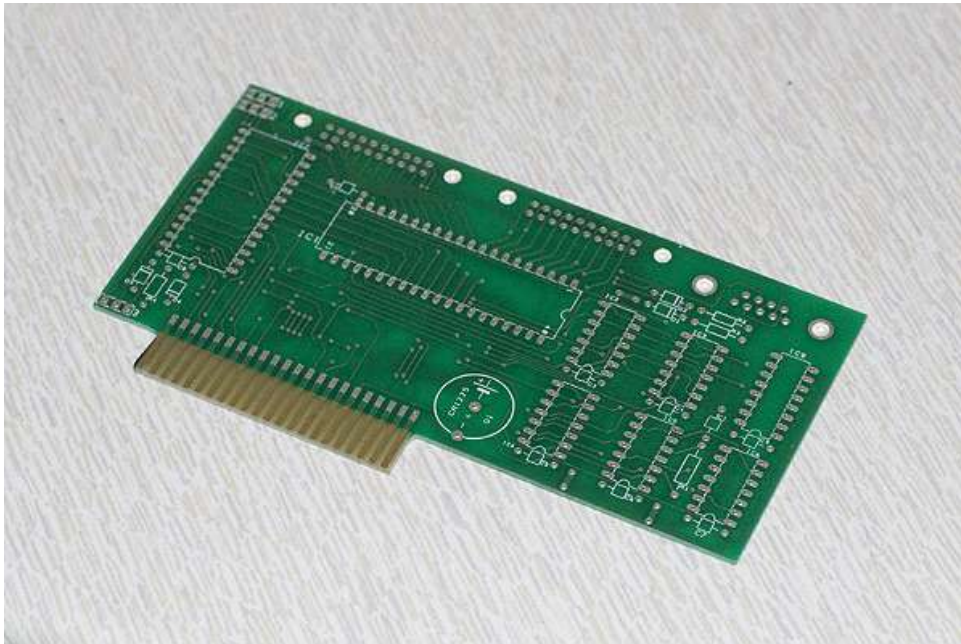
The following parts are optional and only required for battery backup: B1, B1h, D3, D4 and R4.

UPURS socket

The following parts are optional and only required for the use of the UPURS socket: D1, D2, R3, R4 and PL3

Assembly of the Electron User Port expansion

The first thing to do is ensure that the board is entirely clean and free of grease and other contaminants. A quick clean with Isopropyl Alcohol does the trick.

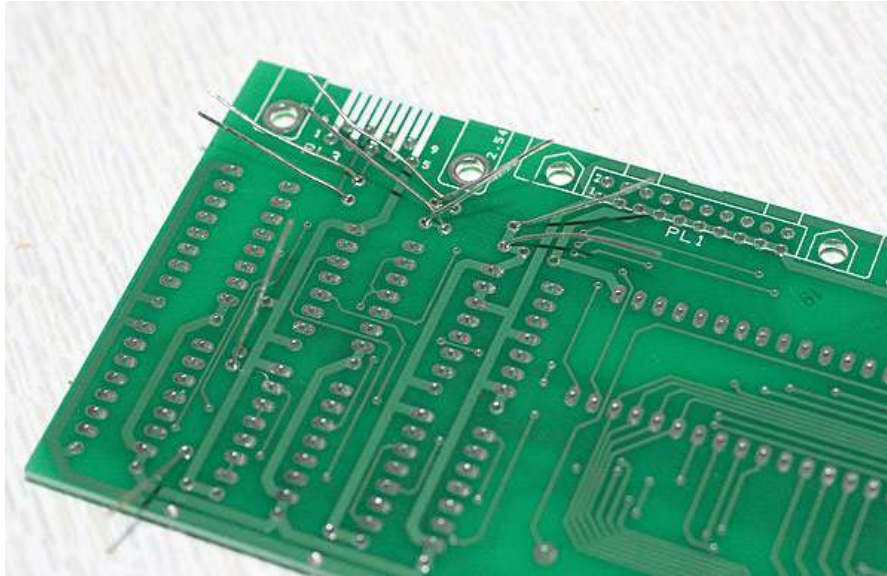


Begin populating the board with the passive electronic components. Normally, when building up a PCB, you start with the IC sockets and headers but on this occasion it is easier to start with the passive electronic components. The 1N5817 diodes are quite tricky to fit as their leads only just fit through the holes on the board. The leads must also be bent as close to the diodes body as possible for a neat fit.

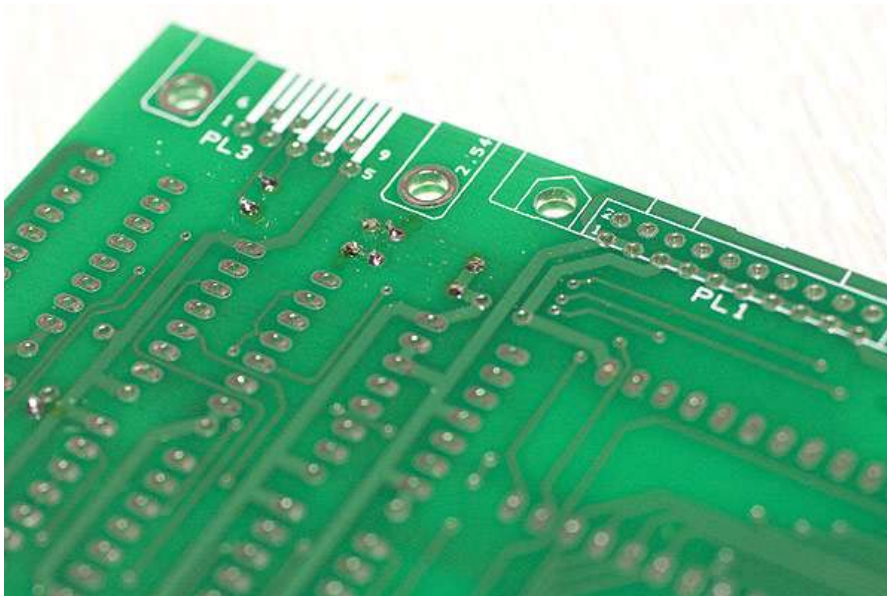
If you aren't intending on fitting the battery holder which is optional, the diode D3 does not need to be fitted and a tinned copper wire link can be used in its place.



By splaying the components leads, they're held in place ready for soldering. Solder each point and trim the leads off once completed.



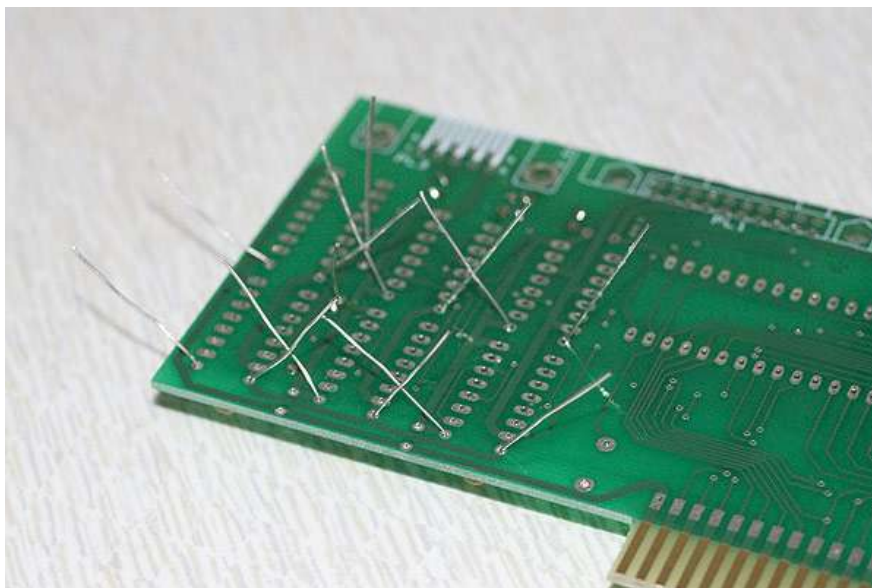
You can see the solder work of the passive components after they've been soldered and trimmed in the photo below.



The next stage is to fit all of the decoupling capacitors. If your capacitors have a narrower lead spacing than required, using a set of fine nose pliers manually bend the leads to the correct shape for fitting.



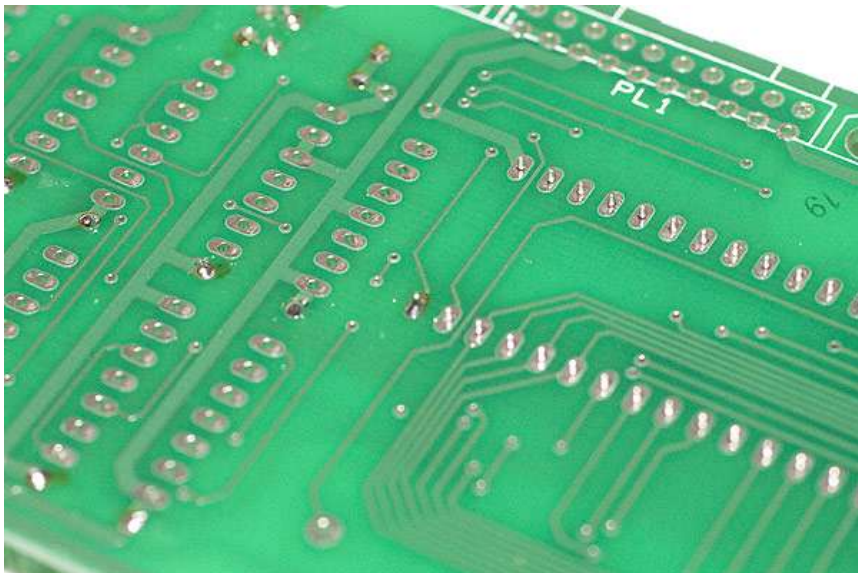
Again, splaying the leads on the solder side helps to retain the loose components in place prior to soldering. Solder and trim the capacitors in place like the resistors and diodes previously.



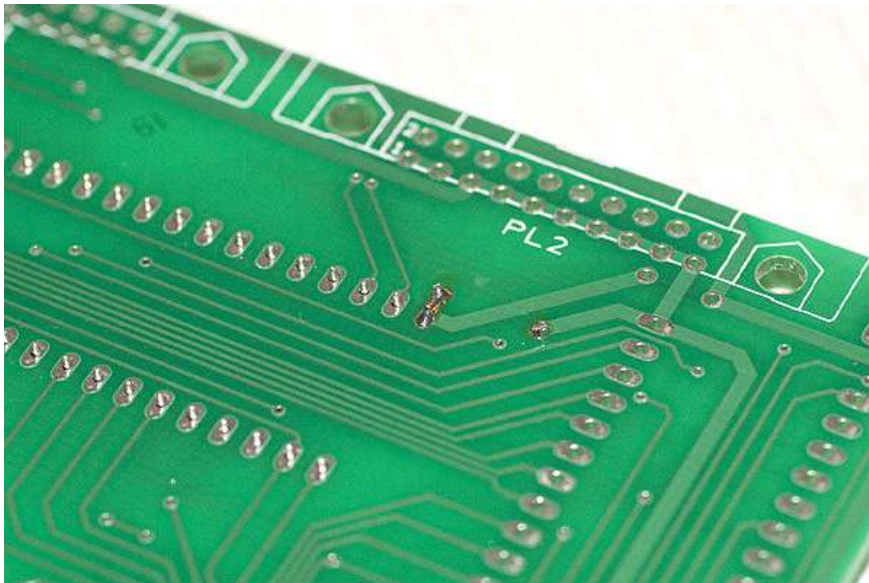
Use Blu-Tack to secure the sockets in place ready for soldering doing each socket in turn.



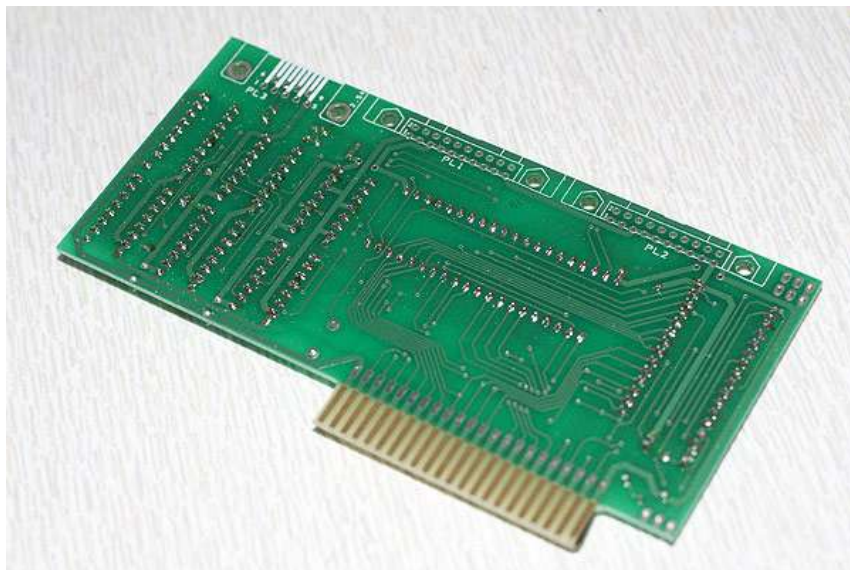
In order to speed the process of soldering all the sockets in place, using the Blu-Tack technique to hold one in place, simply "tack" each one by soldering one pin on opposite corners of the socket.



Tacking the opposite corner successfully fixes the socket in place ready for soldering all the other pins but don't solder all the pins until I've tacked every socket onto the board. This will make the job quicker to complete as you can solder all the remaining pins in one go.



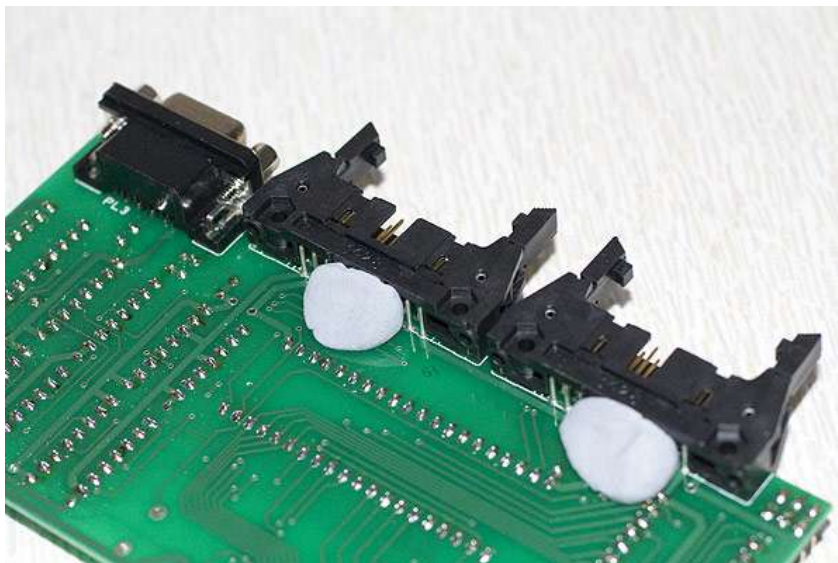
Once each socket has been tacked into place, revisit each socket in turn completing the job.



Using the same Blu-Tack technique, solder the headers to the board.



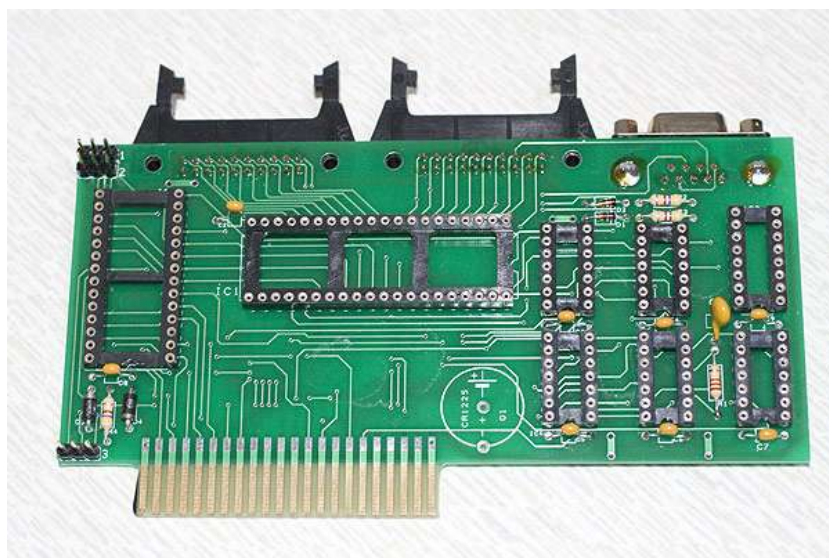
Finally, the User Port and UPURS sockets need to be fitted onto the solder side of the board, soldering them on the component side. Fitting the 9-pin D-type socket can be tricky as the pins can be slightly out of line and require some "gentle persuasion" in order to get them to fit through the holes. As the pins are offset, it's more difficult to fit than the two User Port sockets.



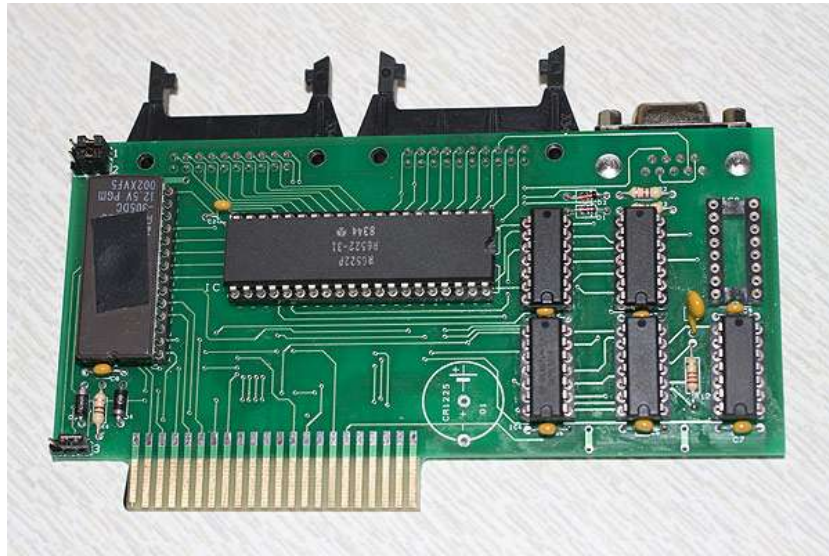
Finally, after soldering all the socket pins, use a large quantity of solder to ensure that there is no play in the mounting of the 9-pin D-type socket on the board.



Here's the completed board ready for cleaning with Isopropyl alcohol followed by population with the integrated circuits and testing.



The final result with all excess flux removed with IPA and the chips fitted. IC8 is not connected and could be used to add extra logic if required in the future.



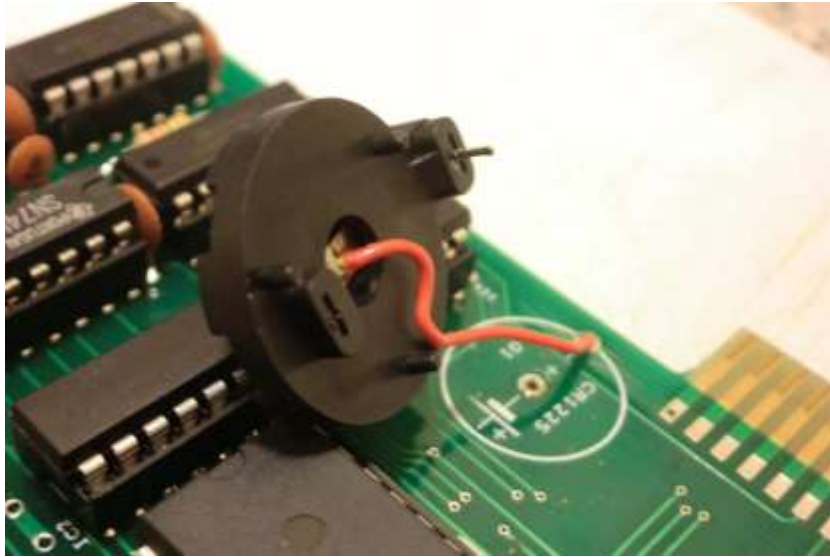
The jumper positions are in the default states and each jumper is configured as follows:

Jumper	East	West
JP1 RAM Write protect	Read/Write	Read Only
JP2 Disable EPROM	ROM Enabled	ROM Disabled
JP3	32K RAM chip and EPROMS 16KB or under	32KB EPROM

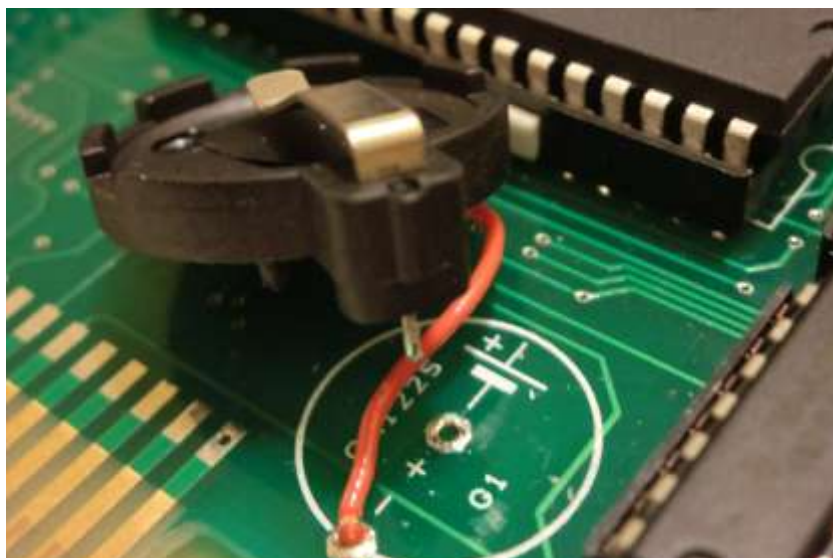
Fitting the battery components

With respect to fitting the battery components, there is a minor error in the PCB layout which must be corrected by fitting alternate battery components to those indicated on the PCB. Rather than fitting a CR1225 holder, a modified CR2032 battery holder must be fitted. The modification adjusts the pins to match the polarity of the holder to that of the PCB.

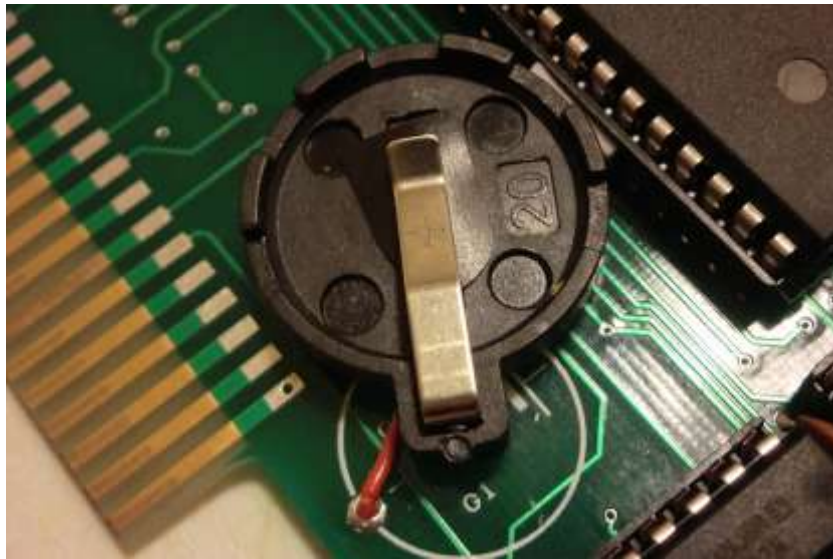
To achieve the modification, the negative pin must be cut off and a short piece of wire attached directly to the negative spring contact which can then be soldered to the negative terminal on the PCB.



The positive terminal of the battery holder can then be soldered into place to the positive connection on the PCB. For extra stability, attach a piece of double-sided adhesive foam between the 6522 socket and where the CR2032 battery holder will fit.



Once soldered into place, use a piece of insulation tape to prevent short circuits when a battery is not fitted in the holder.



Programming for the Electron User Port

The Electron User Port expansion board provides two full user ports that behave identically to the User Port implementation on the BBC Micro and it is recommended that you reference the extensive information about programming the User Port in general from sources such as the BBC Micro Advanced User Guide.

Technical differences between the BBC Micro and Electron User Ports

The technical difference between the BBC Micro and Electron User Ports are simply down to the base addresses of the hardware being different to fit in with the respective memory maps of the machines.

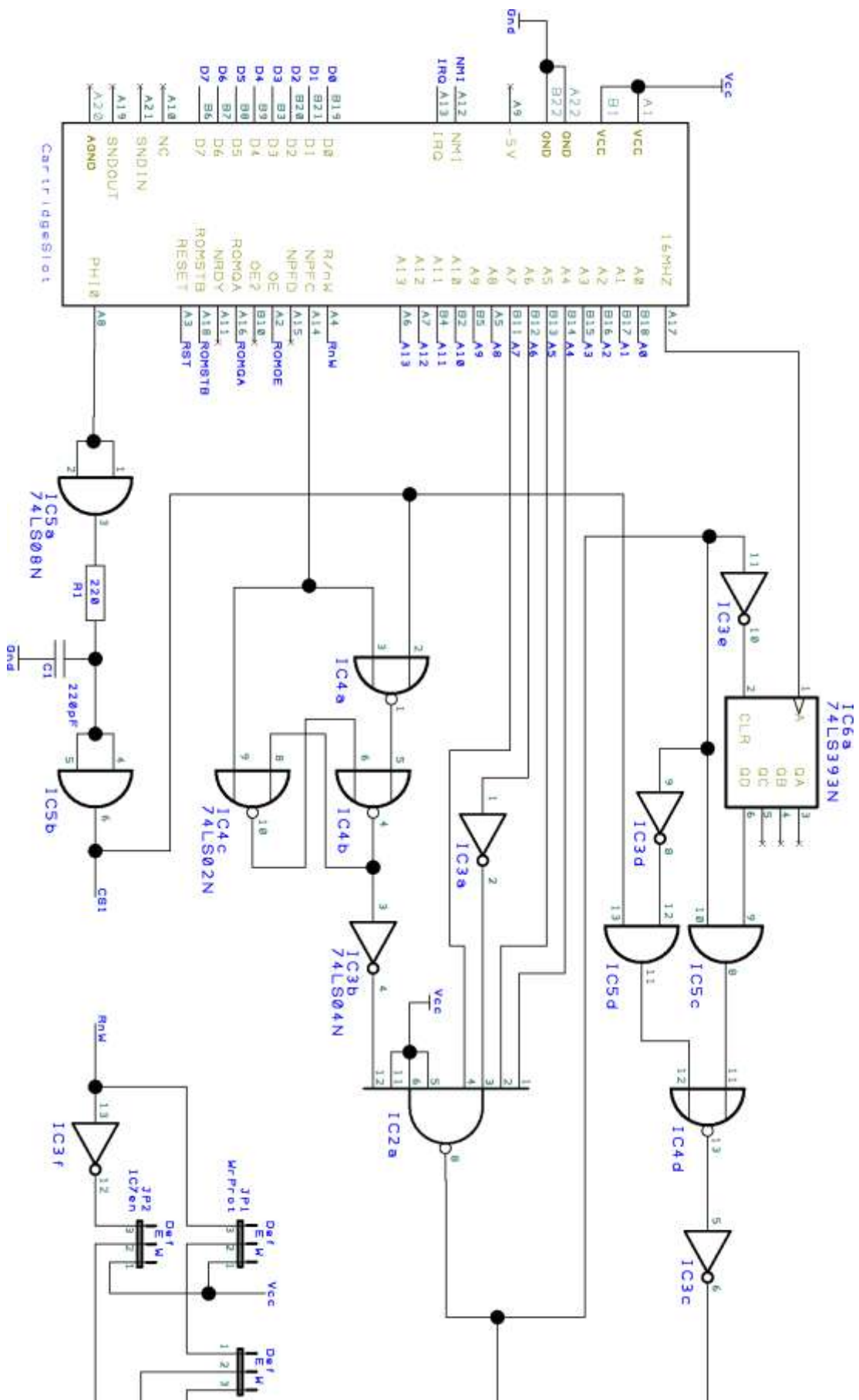
Where the BBC Micro has a base address of &FE60 for the User VIA which controls the User and Printer Port, the Electron User Ports have a base address of &FCB0.

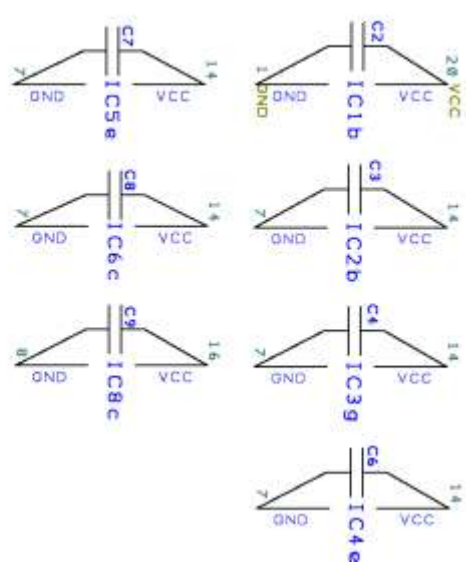
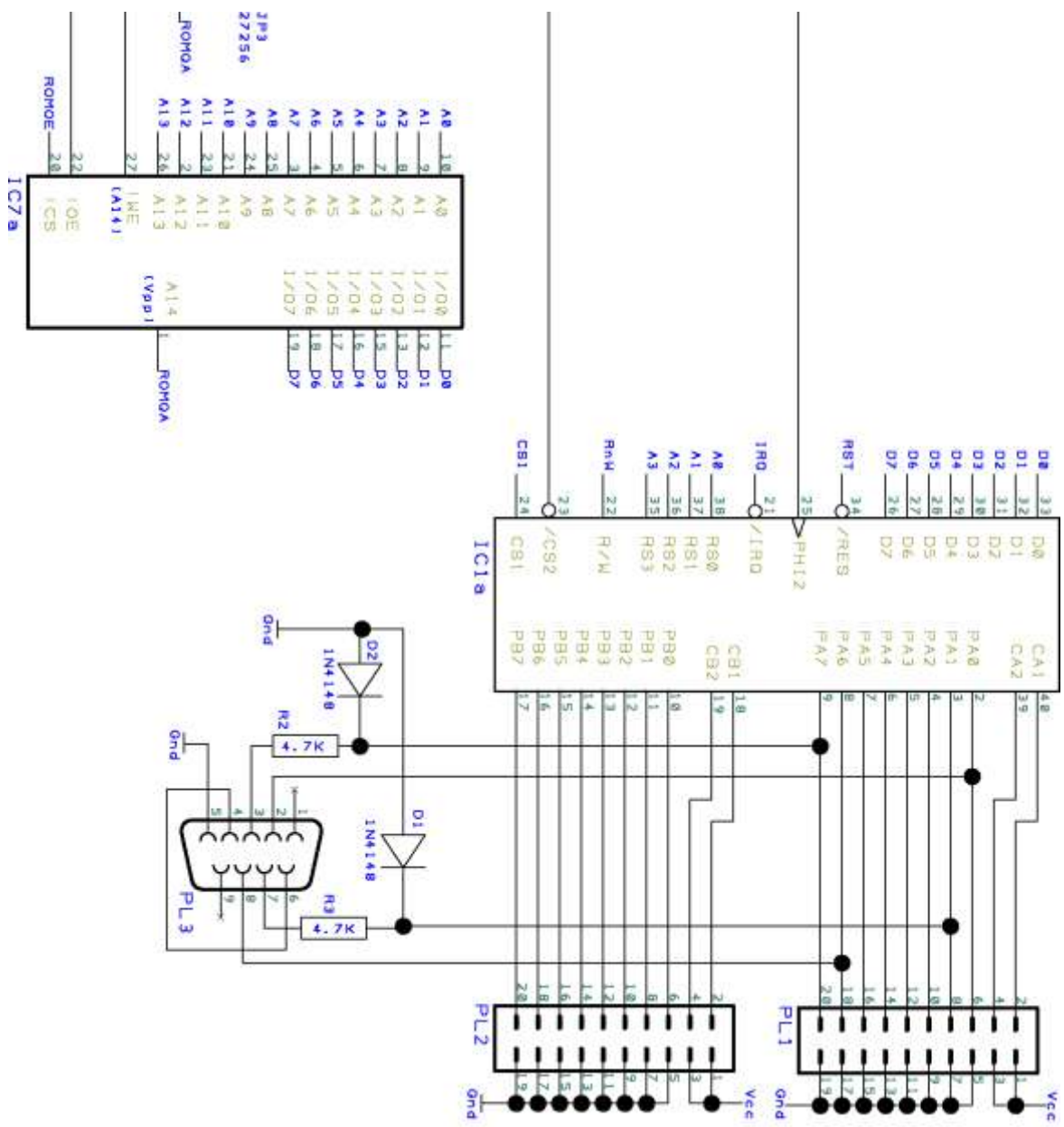
As both the A and B ports of the 6522 VIA are available to the user in full, the following addresses should provide enough information to get started with programming them.

	Port A and UPURS socket	Port B
I/O port	&FCB1	&FCB0
Data Direction Register	&FCB3	&FCB2

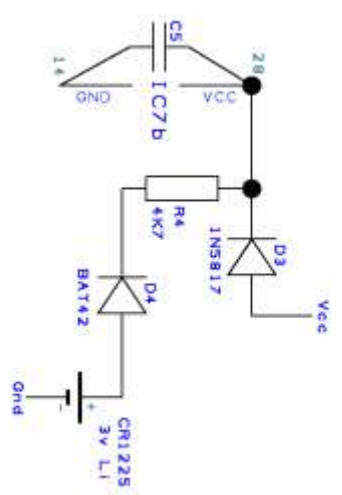
The Electron Plus 1 expansion already has a Printer port on the rear, for which the standard base address is &FC70. Please refer to the Electron Plus 1 user manual for more details.

Schematics





C2-C8 0.1uF



Special Thanks

The unfortunately named UPURS transfer system has been an enjoyable and rewarding project for the author to work on and especially so due to the tireless and generous assistance provided by the other three members of the UPURS development team, namely Paul Vernon, Martin Anderson and Hugh Vance. (paulv, mga1103 and Advance on the stardot.org.uk forums.) Without their help, UPURS would have been a mere shadow of it's now excellent self.

Extra thanks must go to Paul for his amazing contributions supporting & documenting UPURS through his retro-kit.co.uk website. Burning my appreciation into the UPURS ROM therefore seems wholly appropriate ;o) Thanks also to the many enthusiastic Stardot forum members who assisted along the way with UPURS and its Electron expansion host, EUP - you know who you are ;o)