MK14 Pico uploader PCB Rev 4

PCB corrections & Option

- P2-1 and P2-2 connections are reversed, to correct this carefully cut the 2 tracks between the P2-1 and P2-2 header connections (on back of PCB) then solder wires to cross over the connections.
- The only jumper option that needs to be soldered on the PCB is are the back light connections for the LCD, as these can very between LCD modules. For anode on pin 15 and cathode on pin 16 jumper straight across - as pictured, cross over jumpers is anode and cathode connections are reversed.

Soldering

No significant challenges on when it comes to soldering the parts on the board but worth keeping an eye on the following.

- **Contrast adjustment pot**, the cases on some pots are quite large and can overlap with the edge of the LCD PCB. Worth doing a trial fit before you start soldering.
- **Sockets,** I Only socketed the Pico what you do here is down to user preference.
- **Edge connector,** this is mirrored on both sides and there is an option to use a single inline header, again a trial fit to get the best alignment up is recommended before you start soldering. The edge connector pads could be a bit closer to the board edge but still seem ok
- **Resistor networks,** check these are the correct way around, pin 1's toward should point towards each other.

Setting up the Pico [Note, s/w has only been tested with a basic Pico not the Pico W]

A basic understanding of how-to setup a Pico is assumed, for more info check the Raspberry website

Importantly the Uploader s/w is written in CircuitPython, this requires a specific from UT2 file - i.e. not the one from the RaspberryPi website.

1. Download the latest stable CircuitPython UT2 file - https://circuitpython.org/board/raspberry_pi_pico/
2. Flash this onto the Pico
3. Confirm you can see a new drive titled 'CIRCUITPY'
4. Download the adfruit-circuitpython-bundle, making sure get the correct one for your UT2 file (currently version 8.x) - https://circuitpython.org/libraries
5. In the lib folder of the Zip find the 'adafruit_charater_lcd' folder extract this and copy the complete folder to the lib folder on the CIRCUITPY drive - you may have to create the lib folder
6. Download the latest Pico uploader application and associated files from my git hub - XXXXXXXXX
    i. code.py - the main application (CircuitPython autoruns code.py at startup)
    ii. dev.txt - this text file points to a development file name [currently set to dev.hex]
    iii. menust.txt - this text file is where the software setup is stored, see description below.
    iv. any initial Intel hex files you want to be able to upload. Keep titles below 15 characters so they fit on the LCD display. Note the software is relatively dumb, no validation of the Intel hex files is performed.
7. Reset the Pico and it should run the Uploader code, some debug information is sent to the terminal (assuming you are running Thorney or MU on the connected computer). You can also opt run the code manually by renaming it from code.py to anything else.

Menutx.txt

This a simple text file containing 3 integers (000), no space, no separators

- First digit, speed - legal values 0,1,2 which map to 4.7MHz, 4.0Mhz and 'slow'
- Second Digit, backlight - legal values 0,1 which map to 'off' and 'on'
- Third digit, autorun - legal values 0,1 which map to 'off' and 'on', if set to off auto run address if supplied will be ignored.

- A basis syntax check is performed on this file, if an error is detected default values (211) are loaded and entry to setup mode is locked out -this is to manage a circuitpython 'feature' which causes the app to crash otherwise. To fix manually edit the menust.txt file using notepad or equivalent non formatting text editor.

Normal Operation

- App will auto start - no PC connection required.
- Select the desired .hex files from the available list using UK and DOWN keys.
- Hit SELECT to upload the chosen file.
- If set it program will auto run when the upload completes
- To load or delete .hex files from the Pico simply drag and drop to the CIRCUITPY drive.
    - If this causes a write error, the most likely reason it that the uploader application has control of the file system. CircuitPython only allows one 'thing' to write. By default this is the USB app, but to update it's setup the uploader 'grabs' control. This should go back to the USB app following the next reset.
- Each time a file is added or deleted the Uploader App will re-start, this is mildly annoying and hopefully may be fixable in the future - again it's down the setup update process.
- The DEV key offers a short cut to upload a named file (specified in the dev.txt). Hold DEV and SELECT down at the same time to upload this file.

Setup Operation

- Hold down RESET and SELECT, release RESET while keeping SELECT down, until LCD display indicates Pico is in setup mode. Note if an error in the menust.txt file is detected setup mode is inhibited - see above.
- Use the UP and DOWN keys to select the various setup parameters and SELECT to change the options.
- When changes to setup are complete hit DEV to save them, the app will then re-start