Writeup

Pins

The first step is to look at the photographs, identify the components and use their documentation to work out the role of each logic analyser pin.

Documentation:

- Keyboard: https://docs.m5stack.com/en/unit/cardkb_1.1
- Display: https://github.com/pimoroni/inky/blob/776addf15d60ac720ac29893eeb902f0feea9108/library/inky/inky_ssd1608.py

Colour	Logic Pin	Pi Pin	Other Pin
White (KB: Black)	Ground	Ground	Ground
Grey (KB: White)	0	GPIO 2	I2C SDA (Keyboard/Display)
Purple (KB: Yellow)	1	GPIO 3	I2C SCL (Keyboard/Display)
Green	2	GPIO 17	Busy (Display)
Yellow	3	GPIO 27	Reset (Display)
Orange	4	GPIO 22	Data/Command (Display)
Red	5	GPIO 10	SPI MOSI (Display)
Brown	6	GPIO 11	SPI SCLK (Display)
Blue	7	GPIO 8	SPI CE (Display)

Keyboard

- Add an I2C analyser with SDA as Channel 0 and SCL as Channel 1.
- Copy the lines from the terminal view.
- Manipulate them to (see CyberChef recipe below):
    - Extract the data response from 0x5f address requests (see keyboard documentation).
    - Filter out 0x00 (null) responses.
    - Filter out 0x01 (glitch) responses.
    - Convert the hex to ASCII.

```
Regular_expression('User defined','read to 0x5F ack data:
0x(..)',true,true,false,false,false,false,'List capture groups')
Find_/_Replace({'option':'Simple string','string':'00'},'',true,false,true,false)
Find_/_Replace({'option':'Simple string','string':'01'},'',true,false,true,false)
Find_/_Replace({'option':'Extended (\\n, \\t,
\\x...)','string':'\\n'},'',true,false,true,false)
From_Hex('Auto')
```

Display

Identifying the Inky variant can be a little difficult:

- Look at how the variant is retrieved using I2C: https://github.com/pimoroni/inky/blob/776addf15d60ac720ac29893eeb902f0feea9108/library/inky/eeprom.py
- We get `width, height, color, pcb_variant, display_variant = unpack('<HHBBB', data)`.
- So the PCB variant is the 6th byte received (0x0c).
- Looking in the `DISPLAY_VARIANT` list, this equates to `Yellow pHAT (SSD1608)`.

So the SSD1608 file (https://github.com/pimoroni/inky/blob/776addf15d60ac720ac29893eeb902f0feea9108/library/inky/inky_ssd1608.py) is going to be our main point of reference.

After setting up an SPI analyser with MOSI as Channel 5, Clock as Channel 6 and Enable as Channel 7, we can copy the data from the terminal view and write a script to display the images (`solve2.py`).