My Dot Matrix Display Clock

Chuck Rohs 2024-04

Chuck Rohs

- Comp Sci from U of Calgary
- Background in Compilers, Simulation, SCADA, Embedded, Security
- Retired for almost five years
- Interest in embedded hardware, software and pinball

Agenda

Clock Insanity

Products

Hub75 Interface

Hardware

Software

Data Files

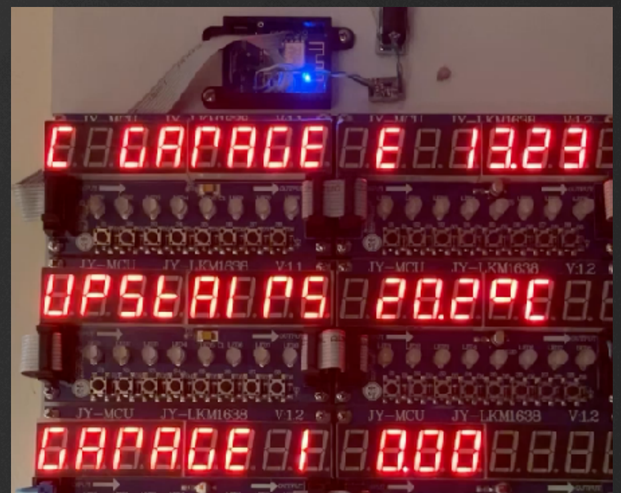
Hub75 with esp8266

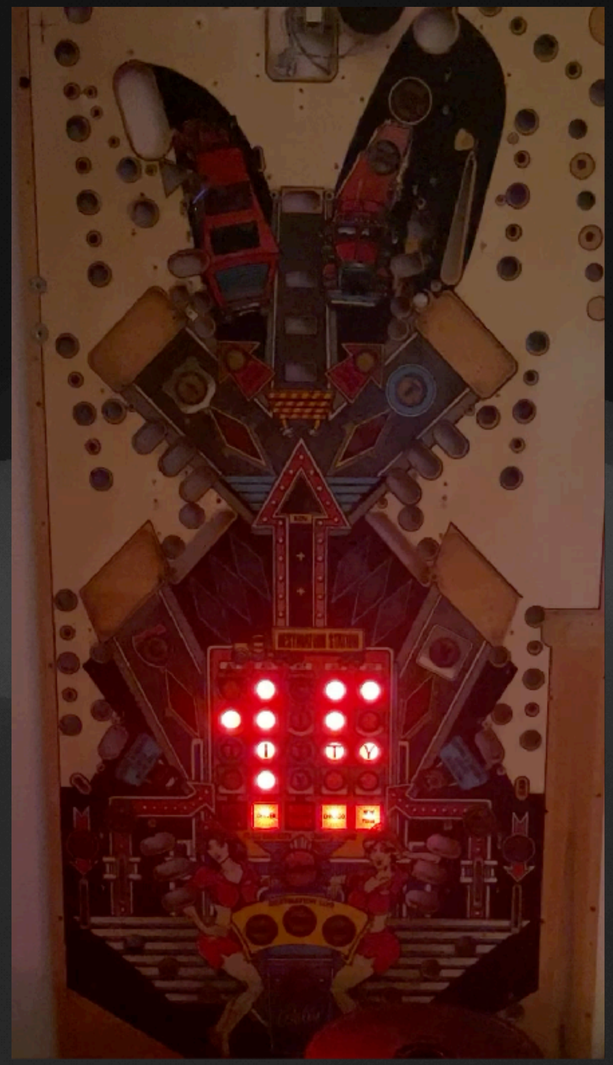
Background

- I like clocks.
- It seems to be “geek” thing
- Examples:
  - E-ink home assistant clock
  - Dot Matrix Heads up
  - DMD pinball clock
  - WS2812 strip clock

Background

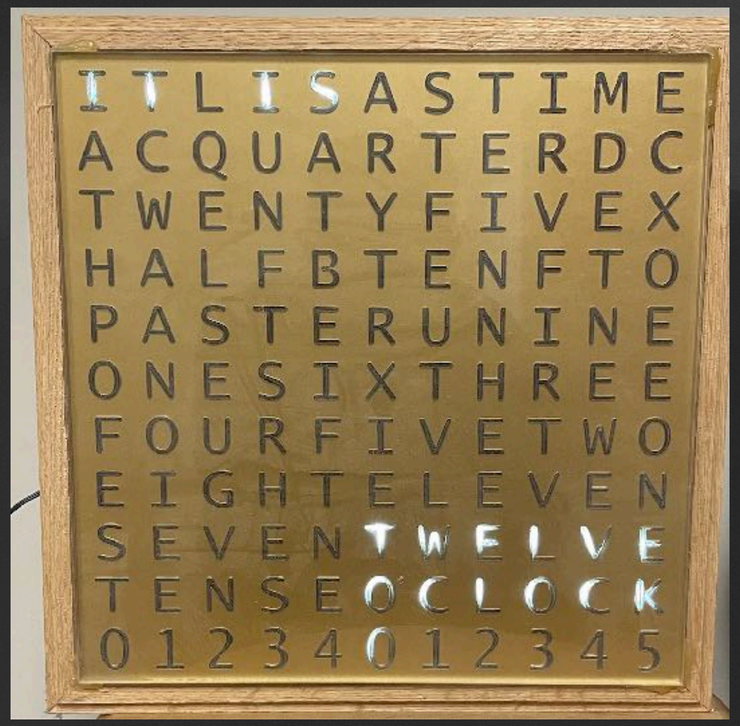
- More Examples:
  - Unreadable WS 2812 strip clock
  - Pinball playfield clock
  - Seven segment home assistant display

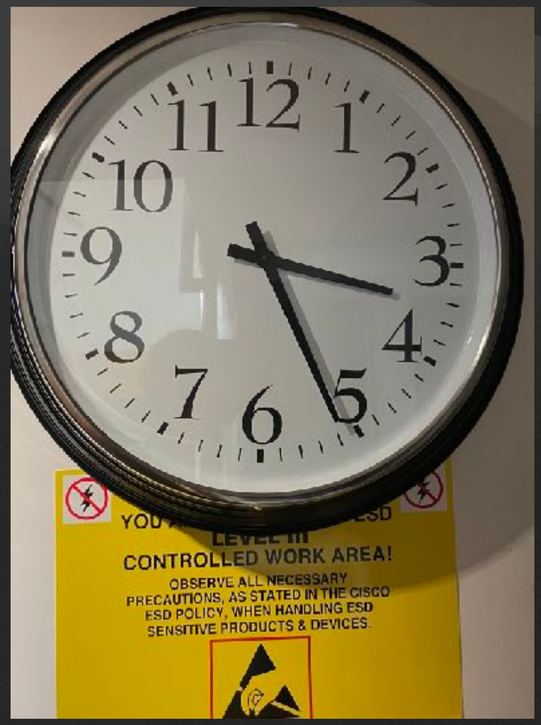




Background

- More Examples:
  - Failed UV CNC clock
  - Word Clock
  - Persistence of vision clock, bought parts for, but never built
  - Normal clock, that is actually legible, to show me the time in the lab



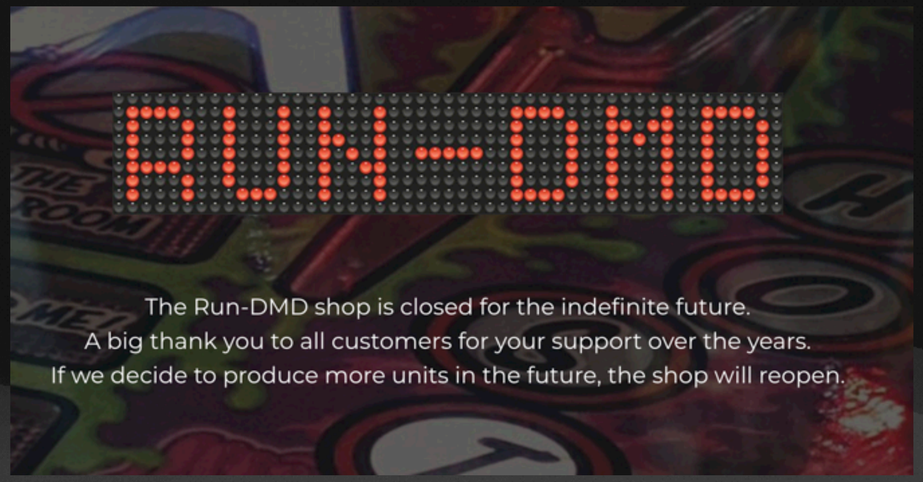


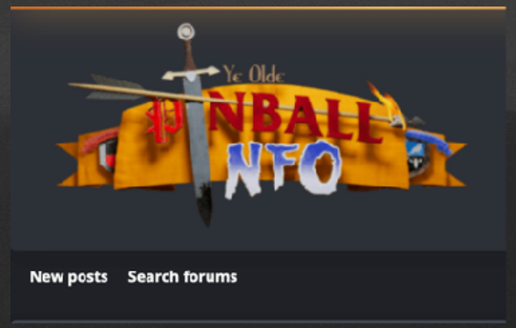
Overview

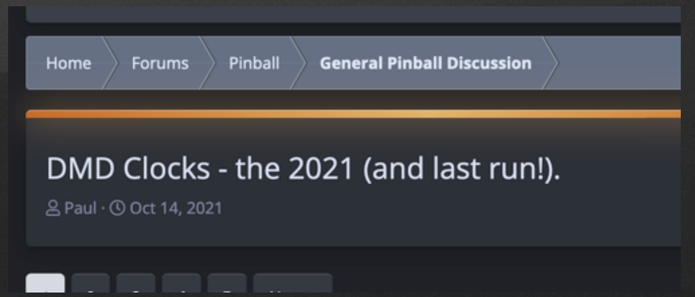
- 90s era pinball machines have 128 x 32 DM VF displays
- newer machine use led matrix displays
- There are a large number of animations made for this format of display

History

- There are/were a number of these DMD clocks on the market
- Some have been shut down due to copyright violation
- Some “sort of” open source ones have come and gone https://gitlab.com/modernhackerspace/dmdclock

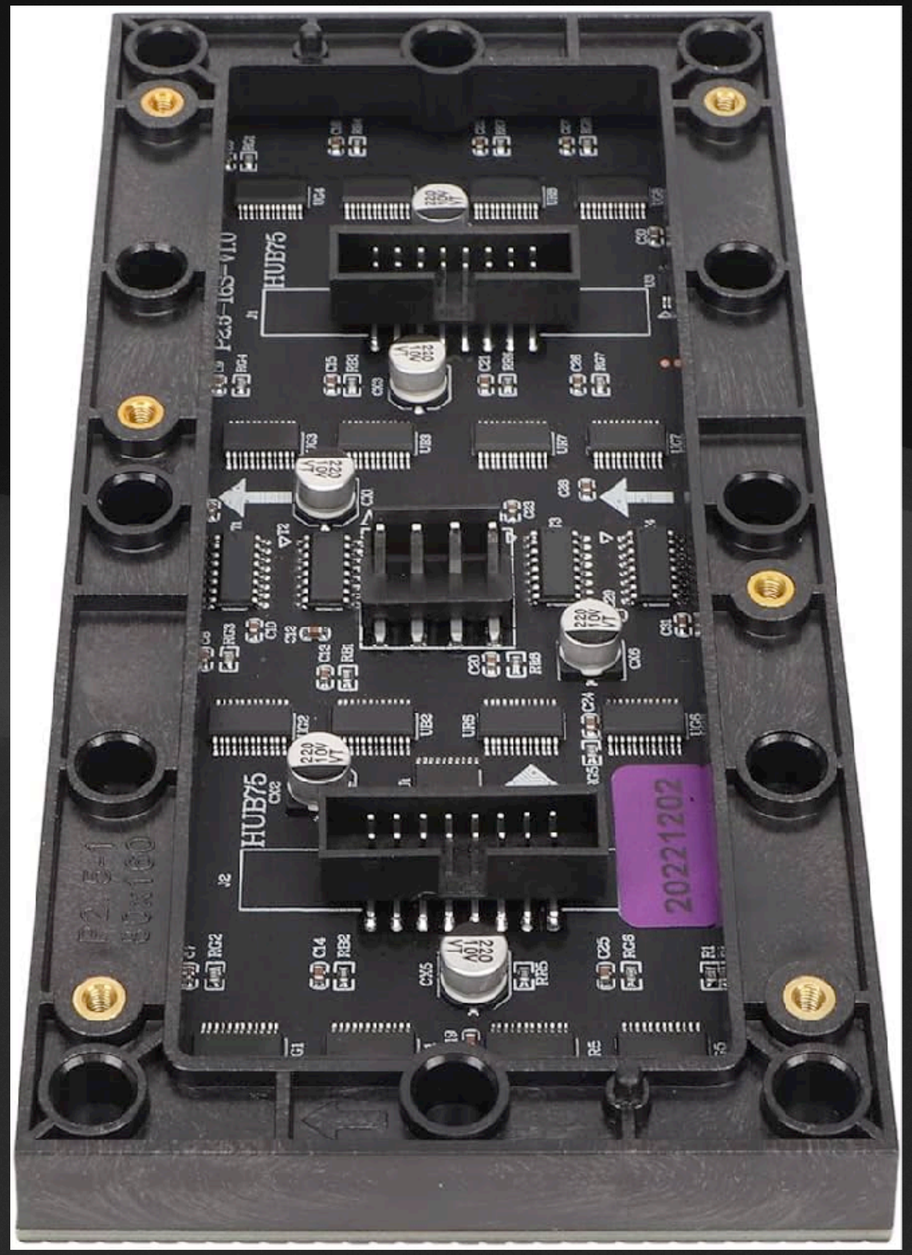






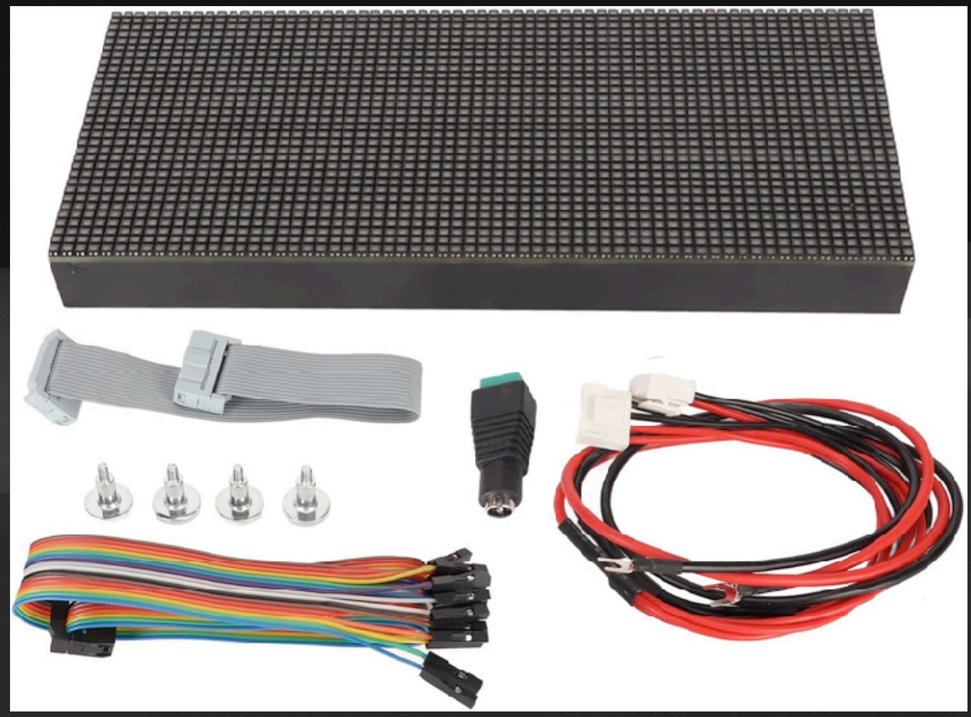
History

- Initially I was going to buy one, but they are all in the $500 range if you could find any.
- I was using these HUB75 displays in my pinball machine.
- I thought I'd build one.
- code to drive these from a raspberry pi has been around a while https://github.com/hzeller/rpi-rgb-led-matrix



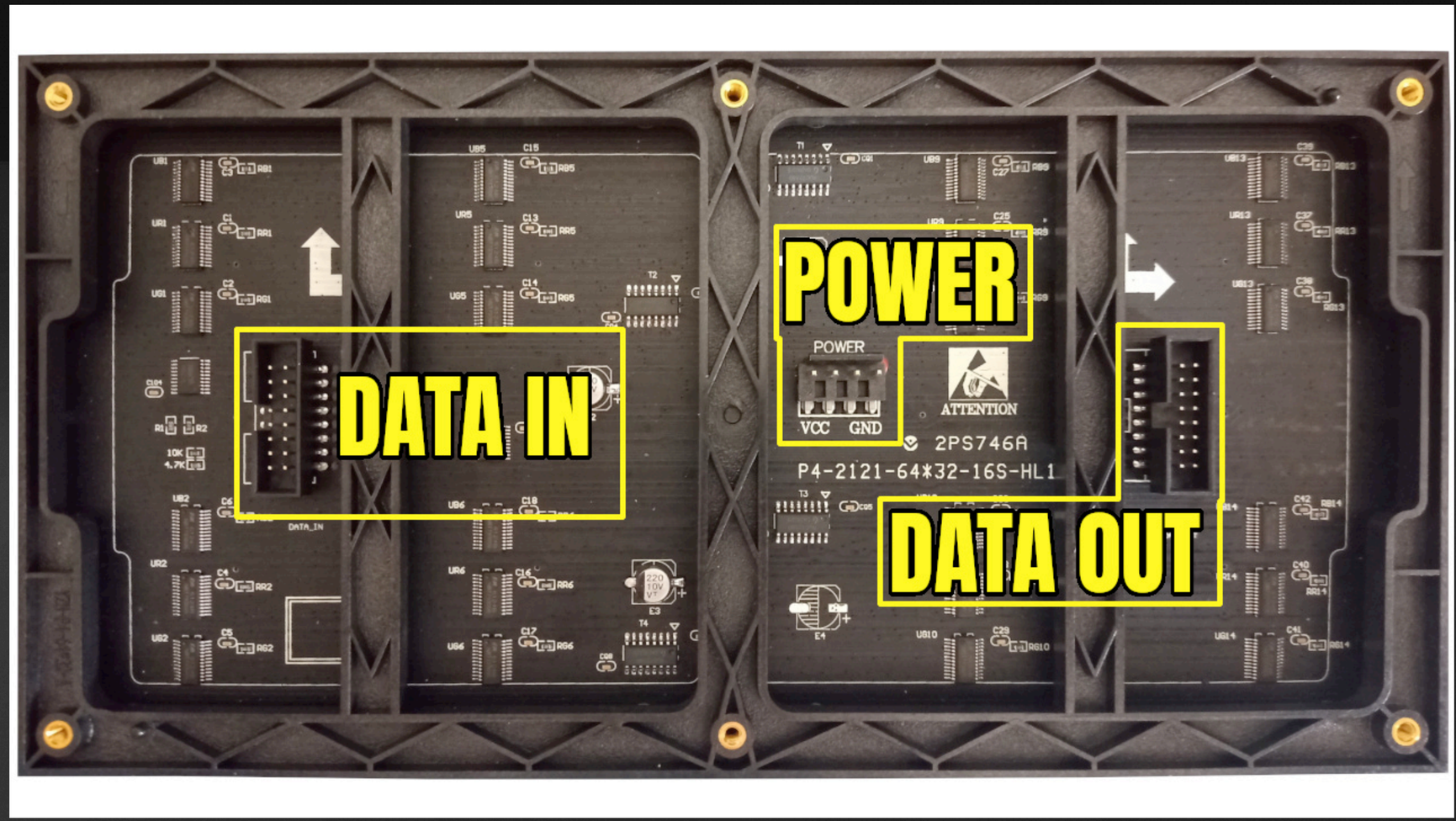
Hub75 Panels

- Used in signage
- Come in various dimensions: 64X32 64x64 32x8
- Panels will daisy chain
- They come in various sizes (pitches)
- p2.5 is the “pinball size”
- p5 are twice as big
- “p” is the spacing in mm



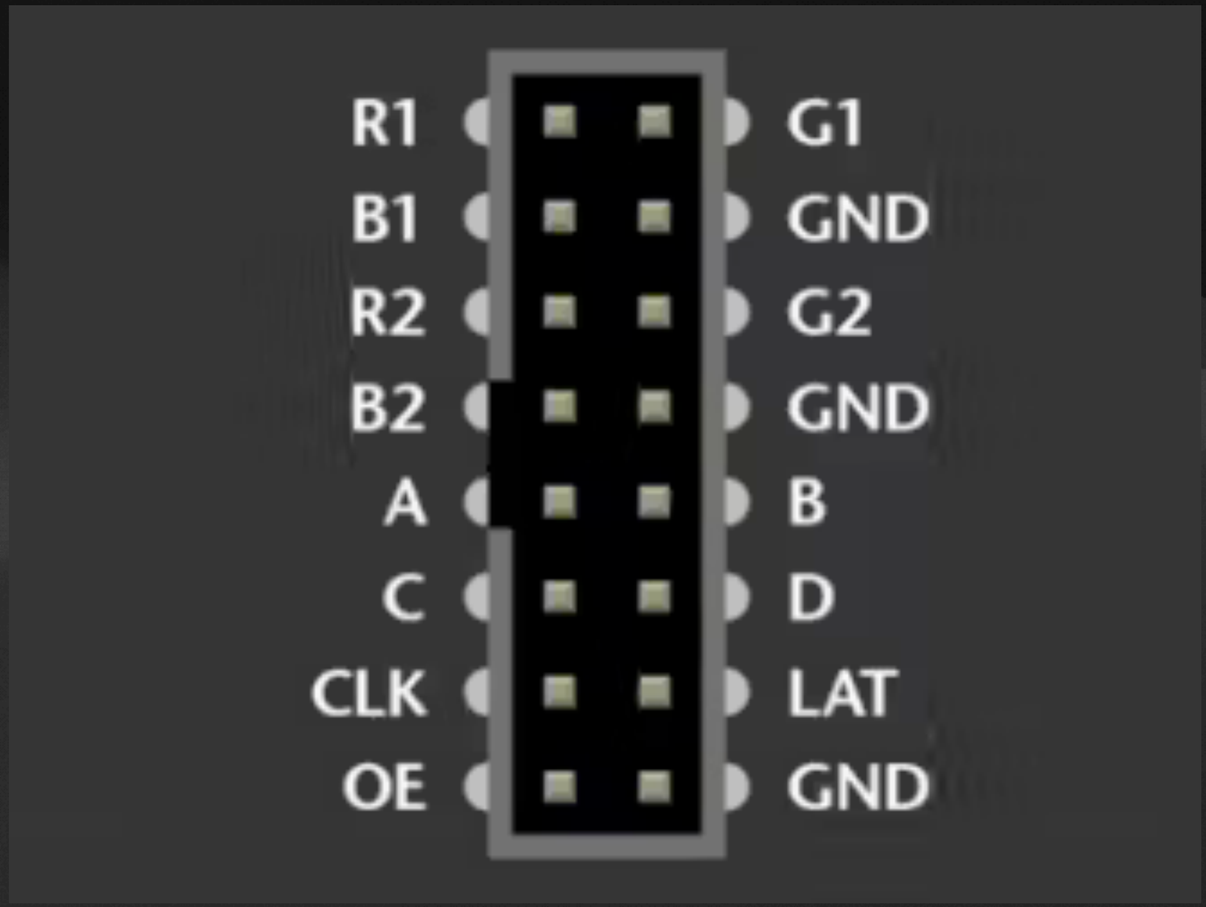
Hub75 Panels

- Hub75 connections
- Panels can be chained together
- Separate 5V power can draw many amps



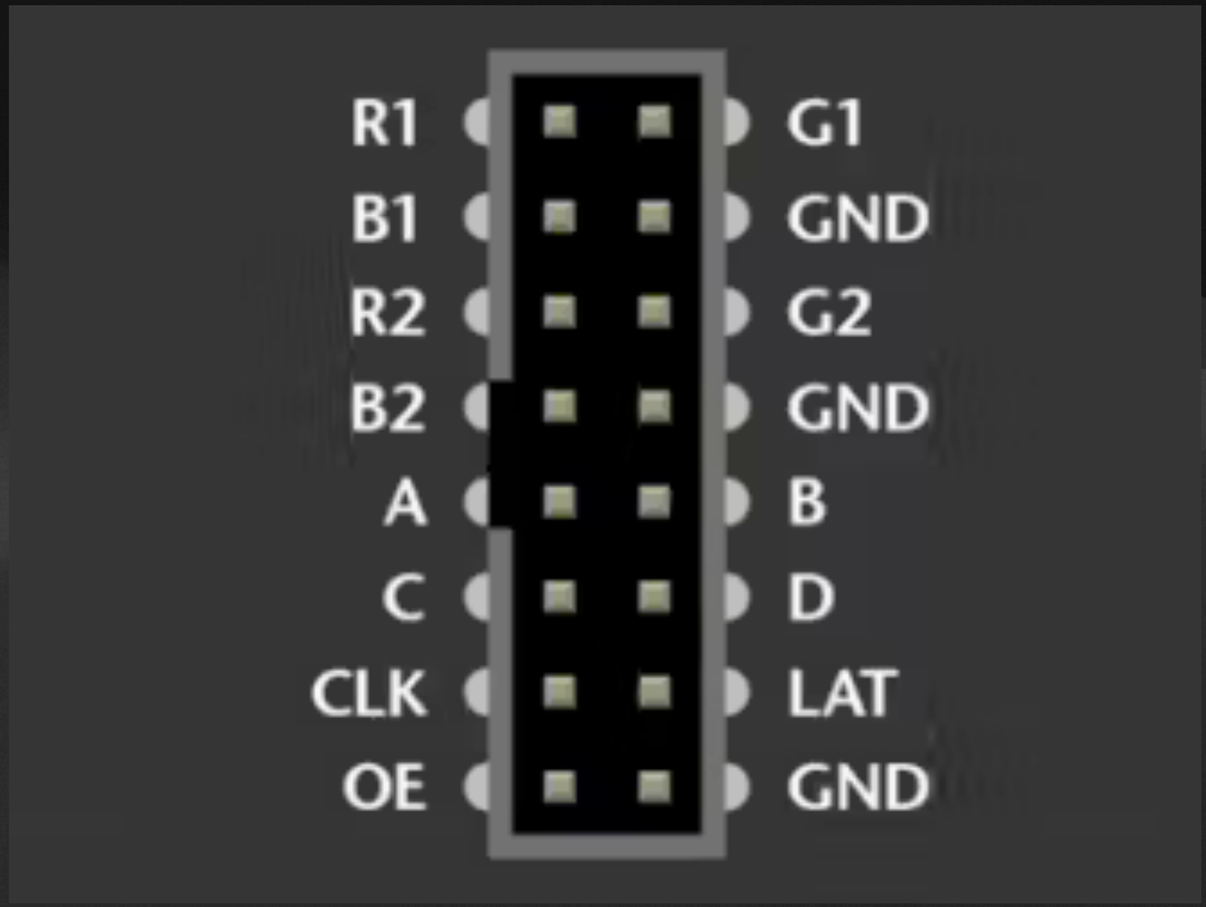
Hub75 Panels

- Separate 5V power (in a separate connector)
- In a 32 line panel, only 2 lines display at a time, one in the top half and one in the bottom
- r1 g1 b1 are data for the top line, r2 g2 b2 are data for bottom line
- 16 addresses required in this case - set by the four address lines: A,B,C,D
- CLK is the clock pin to clock in the serial R G B data
- OE is output enable, and LAT is Latch



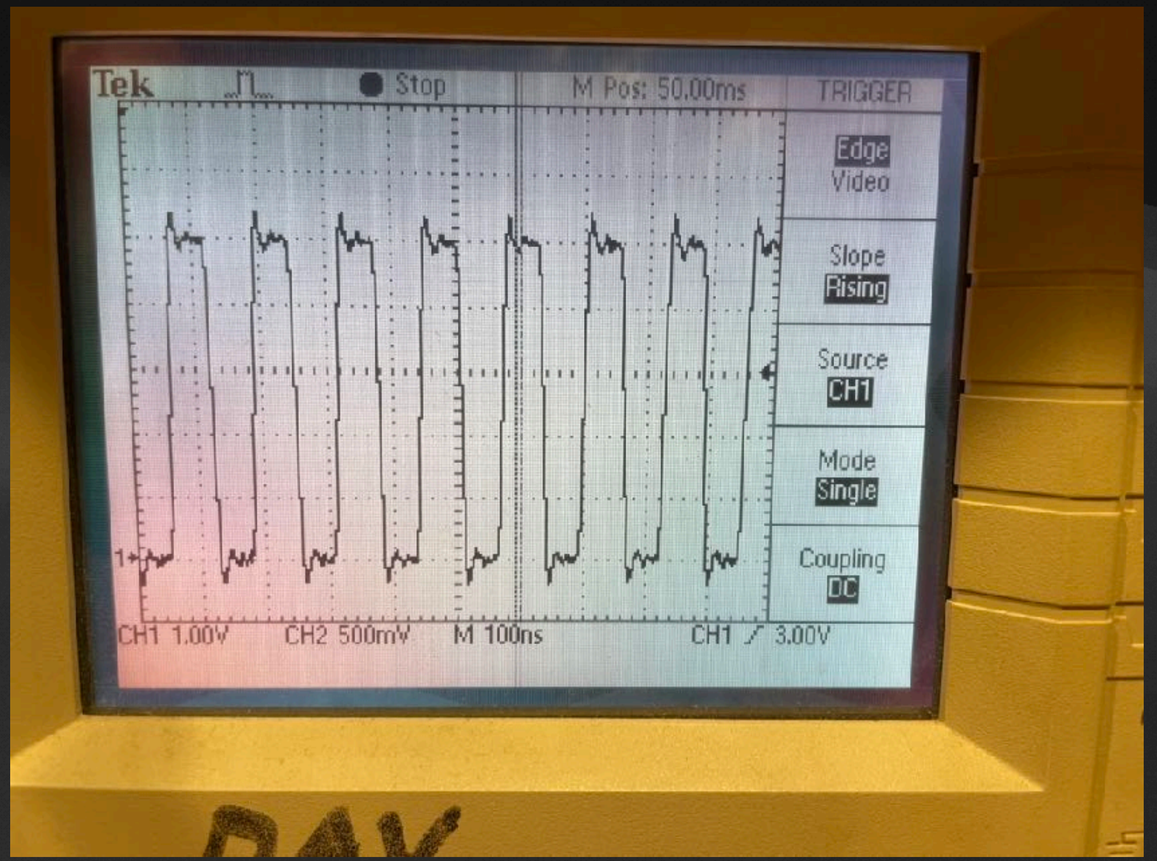
Hub75 Panels Continued

- Leds are on or off.
- To achieve any kind of intensity differences, it must be done using PWM in software.
- I.E. Super high frame rate.
- Eg 5Mhz / 2048 “pixels per field” = 2440 FPS 2400 / 30 = 80 updates per frame. ~6 bpp of color depth.
- Panel hardware can handle 75 MHz



Hub75 Panels Continued

- Scope on the pi pico clock



Driving the Displays

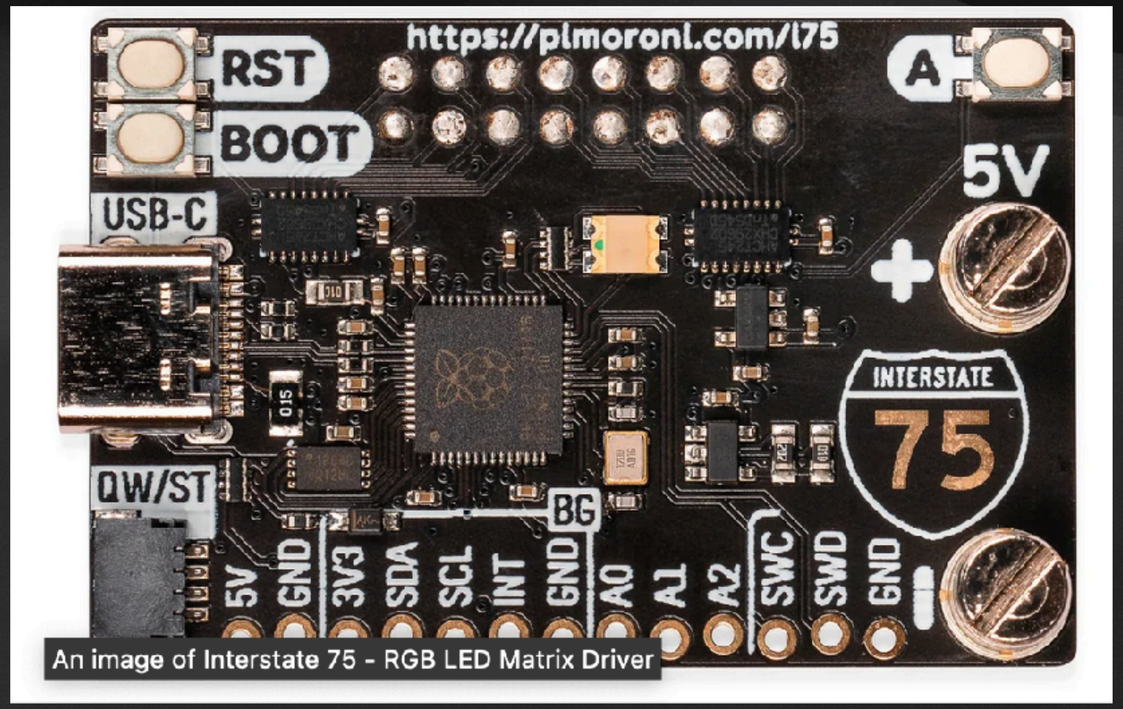
- Lots of options for capable microcontrollers/driverboards:
  - RaspberryPi - lots of software for this
  - Esp8266 - works but not enough I/O for effective framerate (PWM)
  - Teensy + smartled shield - more expensive works well
  - Pi Pico (Rp2040) - works great

Requirements:

- One of the things I needed was processing data at 5 M Baud from my pinball machine. (not required for the clock)
- This really limited me to the Teensy and the rp2040
- RP2040 has PIO that I wanted to play with, and allowed better control over PWM brightness.
- Also needed:
  - RTC
  - µSD
  - Buttons / rotary encoder for settings

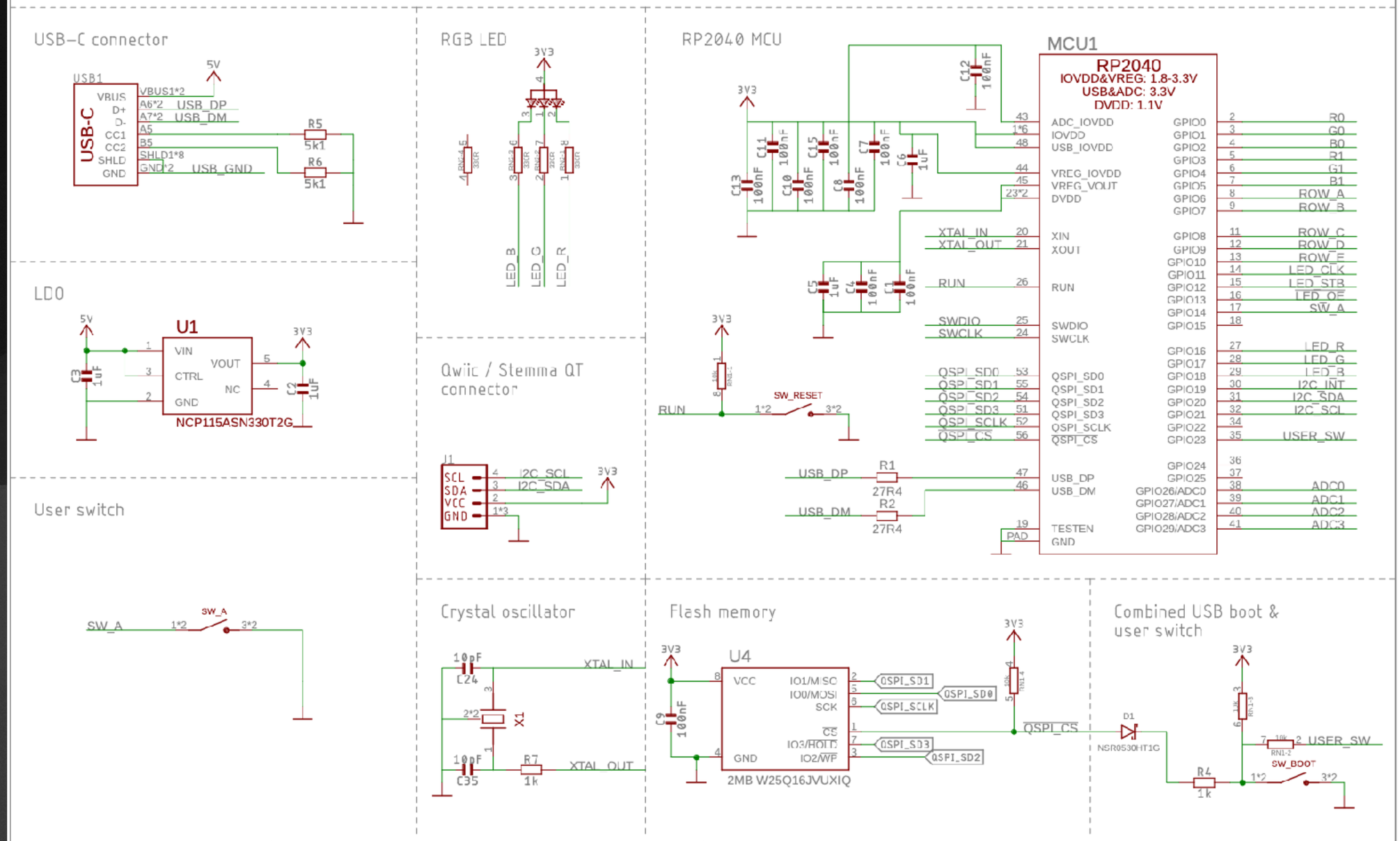
Hardware:

- Pimoroni already had decent hardware and software for this purpose - interstate75
- This was a good starting point for the design



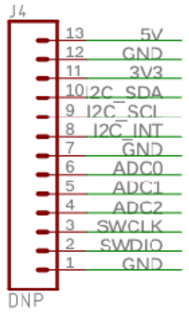
An image of Interstate 75 - RGB LED Matrix Driver

Interstate 75 (HUB75 LED Matrix Driver), PIM584

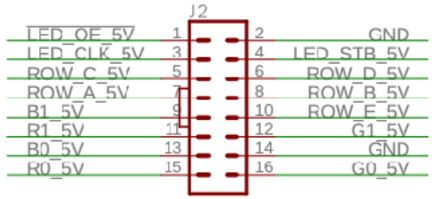


Interstate 75 (HUB75 LED Matrix Driver), PIM584

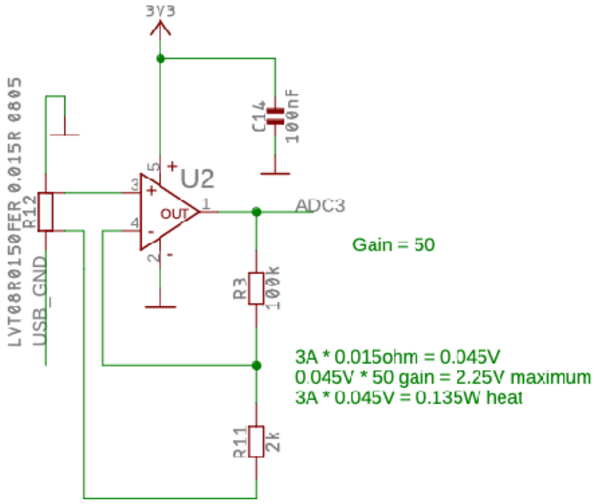
Expansion header



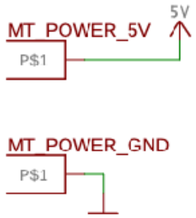
LED Matrix header
(top view of driver side connector)



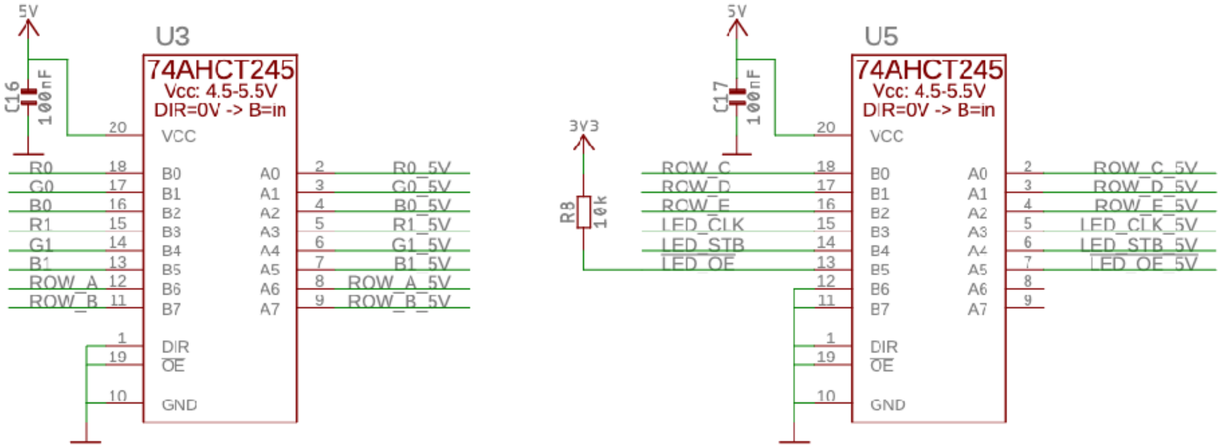
Current sense amplifier



5V screw terminals

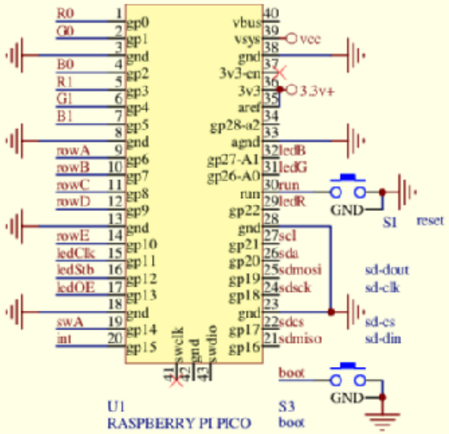


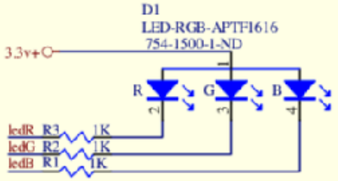
5V level shifters



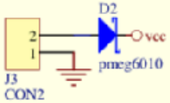
Chucks-I-75matrixBoardV1.3

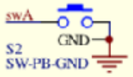
V1.0 uses R Pi Pico to drive a Hub-75 DMD
V1.1 adds uSD card in socket 3M5607CT-ND
V1.2 adds MCP7940 RTC on i2c bu
V1.3 fixes D1 footprint and rotates SD by 180 degrees, swap uSD miso mosi



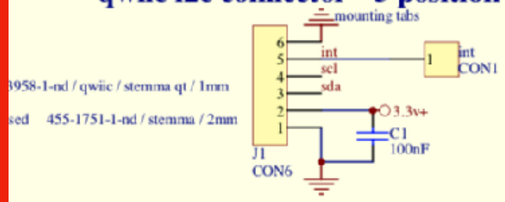


5V power

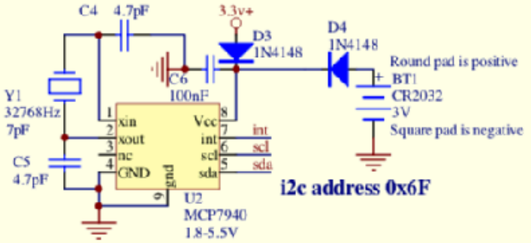




qwiic i2c connector - 5 position

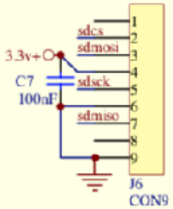


RTC

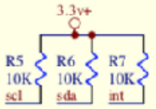


i2c address 0x6F

uSD socket

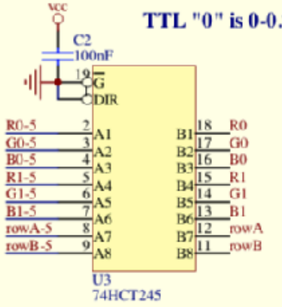


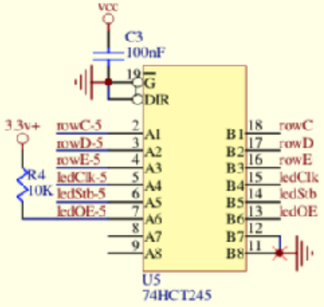
3M5607CT-ND



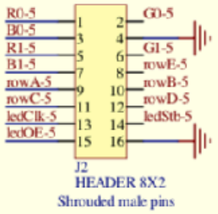
5V HCT logic will switch properly with a CMOS 3.3V drive

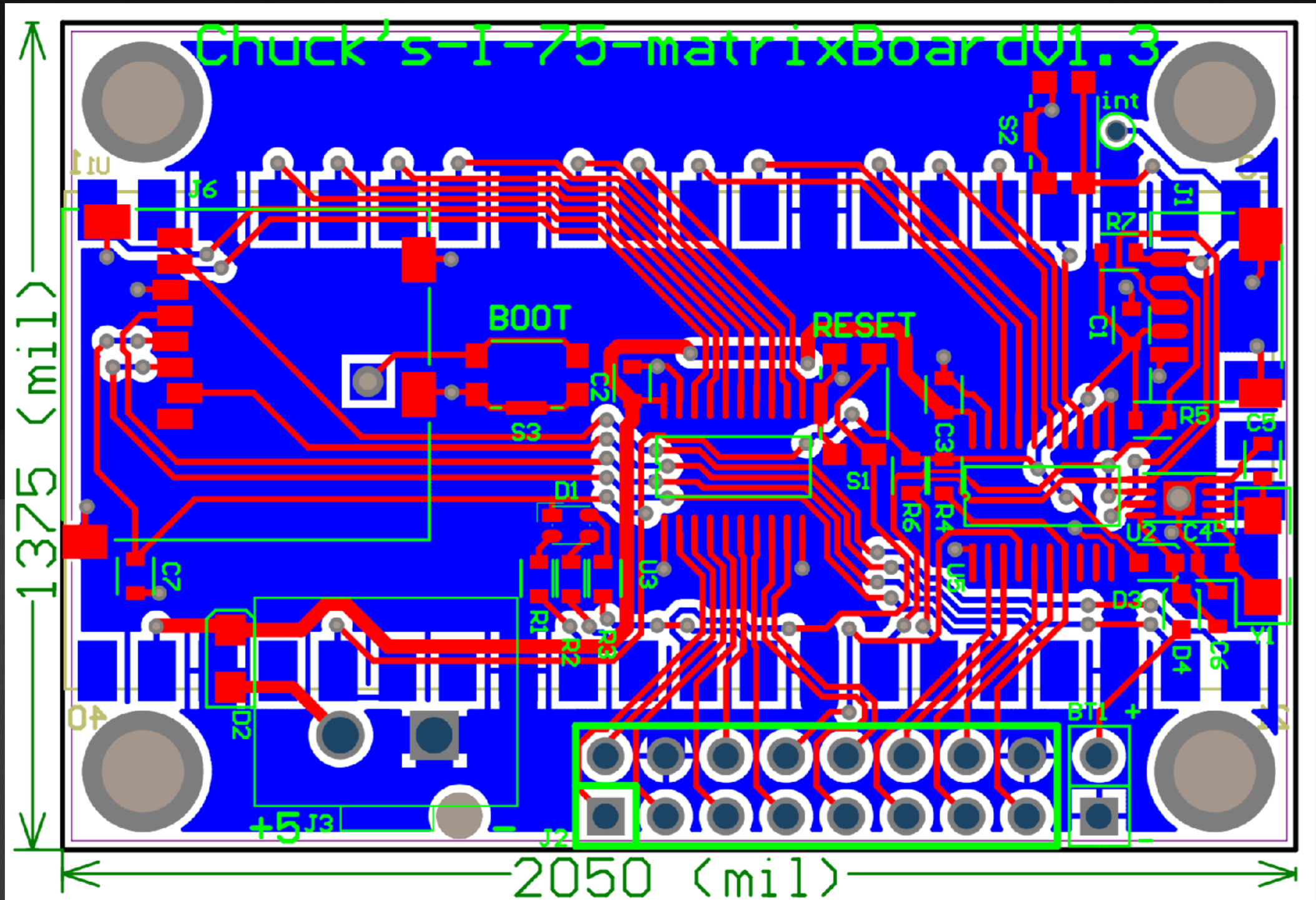
TTL "0" is 0-0.7V, "1" is 2.0-5V





To HUB-75




Chuck's-I-75-matrixBoardV1.3
1375 (mil)
2050 (mil)
BOOT
RESET
S3
S1
S2
int
J6
J3
J2
+5
BT1
U1
U2
U3
U4
U5
C1
C2
C3
C4
C5
C6
C7
R1
R2
R3
R4
R5
R6
R7
D1
D2
D3
D4
Q1
Y1

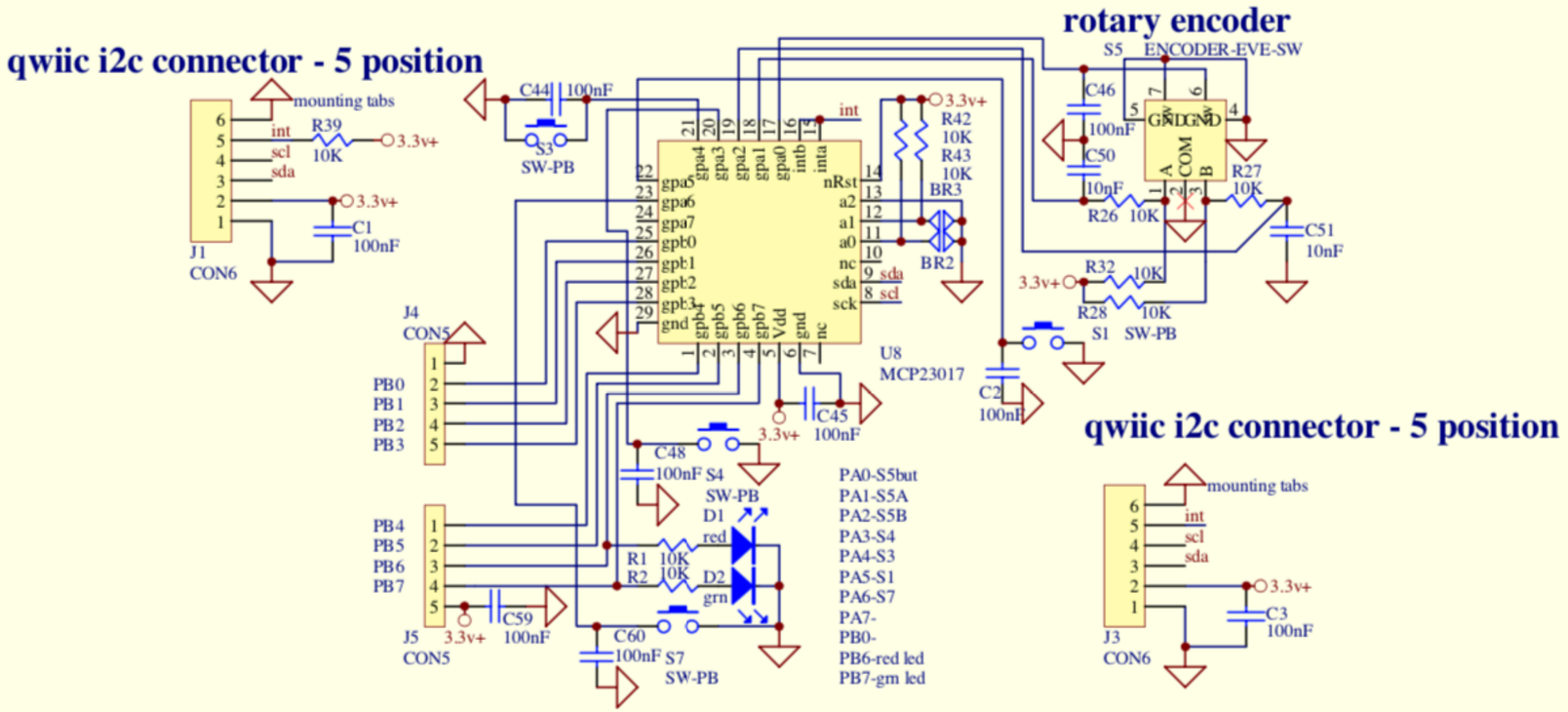
i2cButtonsV1

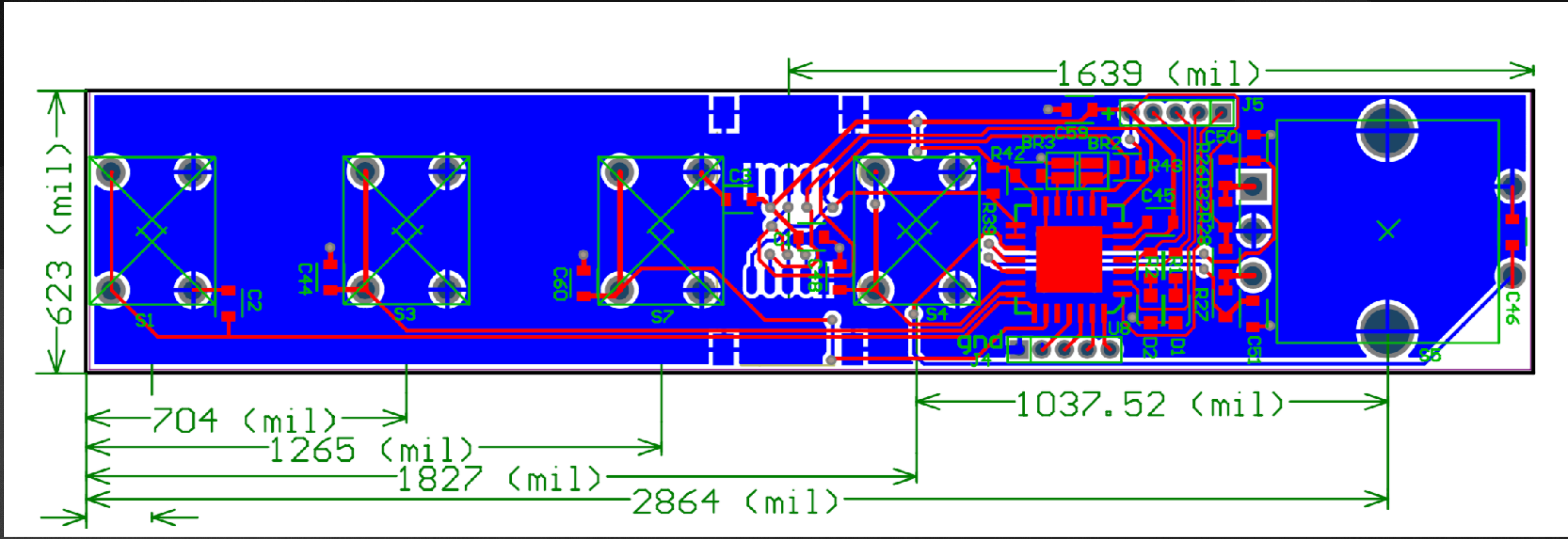
by David Viberg. Feb, 2023

MIT License

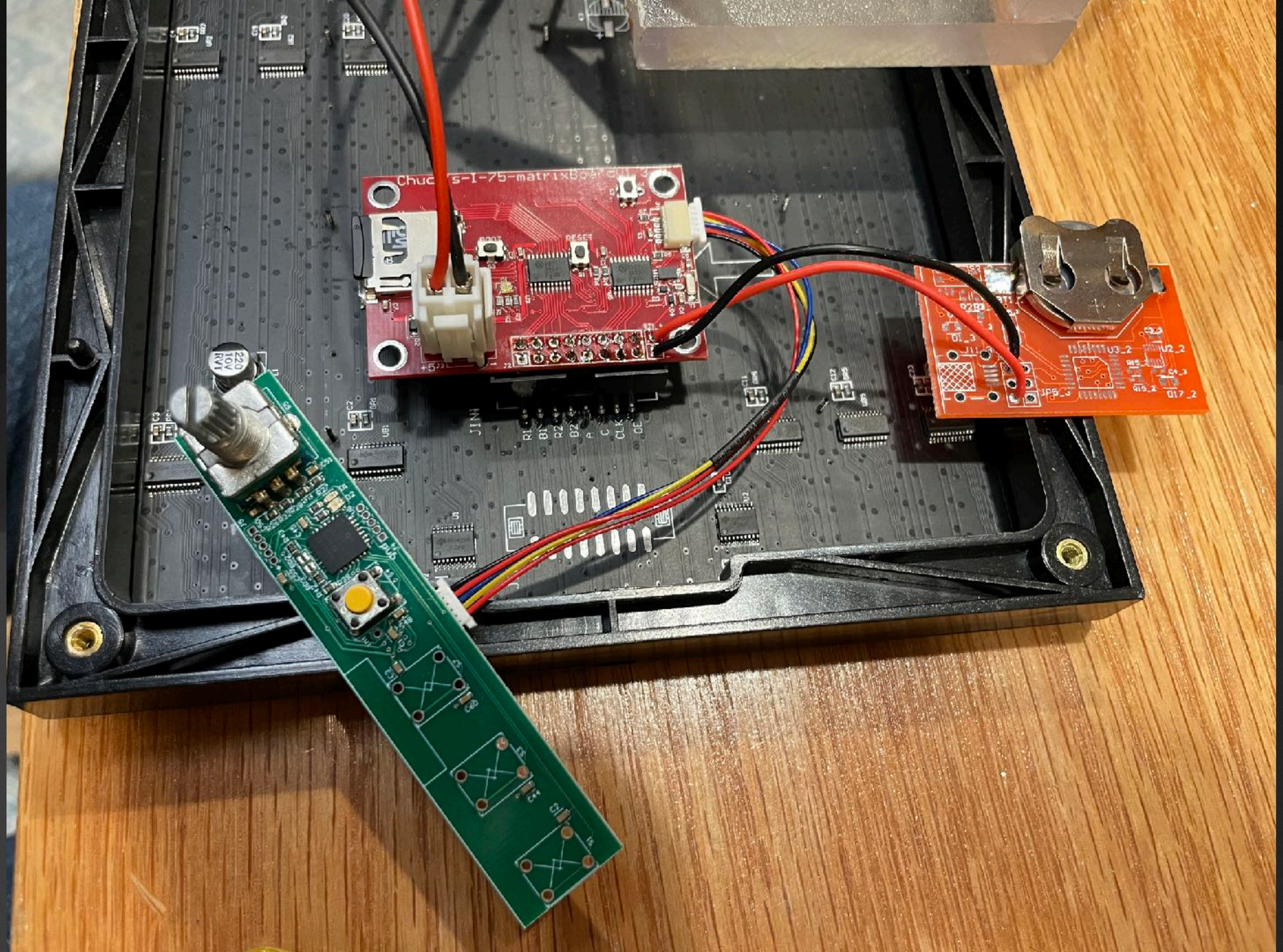
In collaboration with Chuck Rohs / Zethus Information Systems

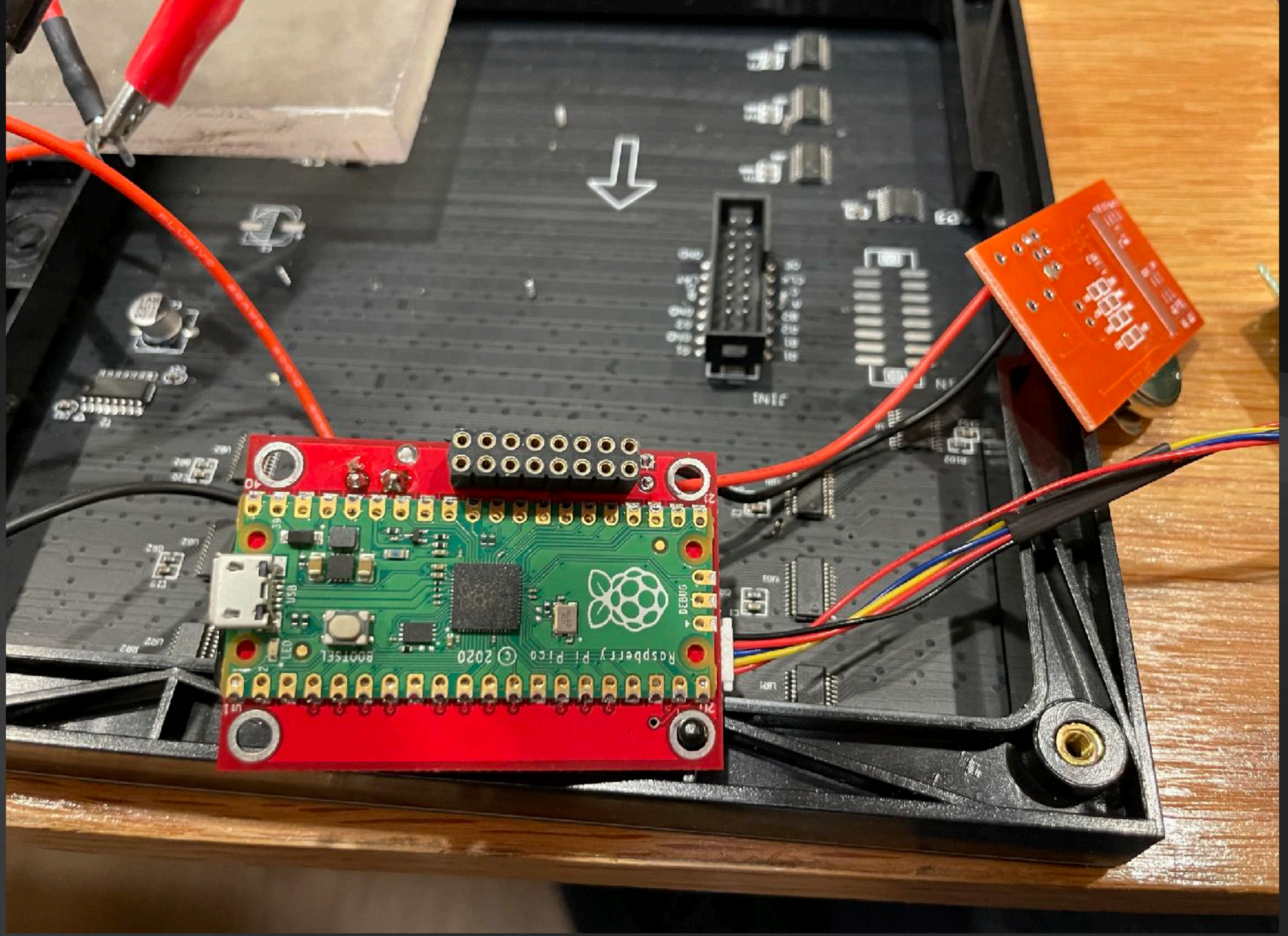
4 push buttons and an encoder,
all are hardware debounced
program pullups for PA0, PA3, PA4, PA5 and PA6





1639 (mil)
623 (mil)
704 (mil)
1265 (mil)
1827 (mil)
2864 (mil)
1037.52 (mil)
S1
S3
S7
S4
S5
J5
J4
C2
C44
C60
C3
C59
C45
C50
C46
C51
R42
R43
BR3
BR2
U8
D1
D2
gnd




Raspberry Pi Pico © 2020
BOOTSEL
LED
USB
DEBUG

Software

- At a high level this software is a mashup of:
  - Teensey pixelmatrix code from mission pinball - for serial port “framebuffer” streaming
  - Interstate 75 code for pi pico pio/hub75 driver
  - custom code for:
    - ROM image decode
    - Real Time Clock
    - Menus,Time Display, Second indicator, Font handling

Dev Environment

- VSCode
- CMake
- Programming is done by coping a “UF2” file to the pico in bootloader mode.It appears as a mass storage device over USB.

Animations

- I investigated a few sources for these
- Original roms and other software images had proprietary run length encoding
- The RUN-DMD image had “raw” images, but the frame sequencing and timing, needed to be reverse engineered, with a lot of “burn and learn”
- Animations are broken down by machine, and each machine animation can be turned on or off in the menu.

Animations

- I spent lots of time in HexFiend trying to decode their Roms for my V1

Animations

- Each animation has a list of frames
- Each frame points to an “image” and has a delay.
- There is a mask for clock transparency
- Lots of pinball animations are just two or three frames cycling over and over

Animations

- After doing all the reverse engineering, I stumbled on this:

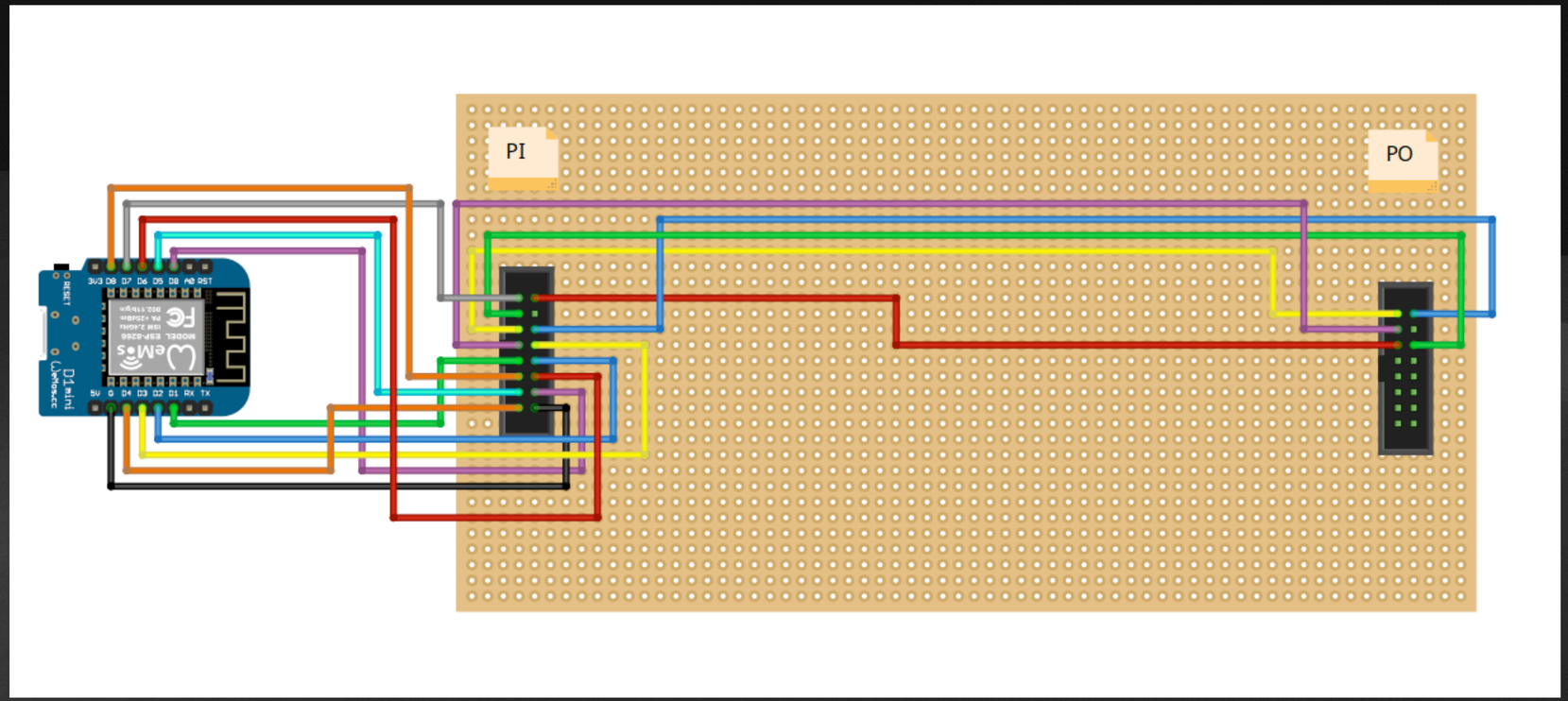
  https://github.com/sigmafx/DotClk-Resources
- ~2000 animations in separate files
- Makes it easy to add your own
- Has tools to extract animations from ROMs

- I rewrote my parser and menu code at this point.

ESP8266 and HUB 75

If you don’t have enough I/O you can daisy chain the outputs back into the inputs

In this example 1 data line is used to drive all six (RGB1 and RGB2) datelines



Demo

Demo video here

Thanks for your interest!

Questions?