Bitbanging is so 2017

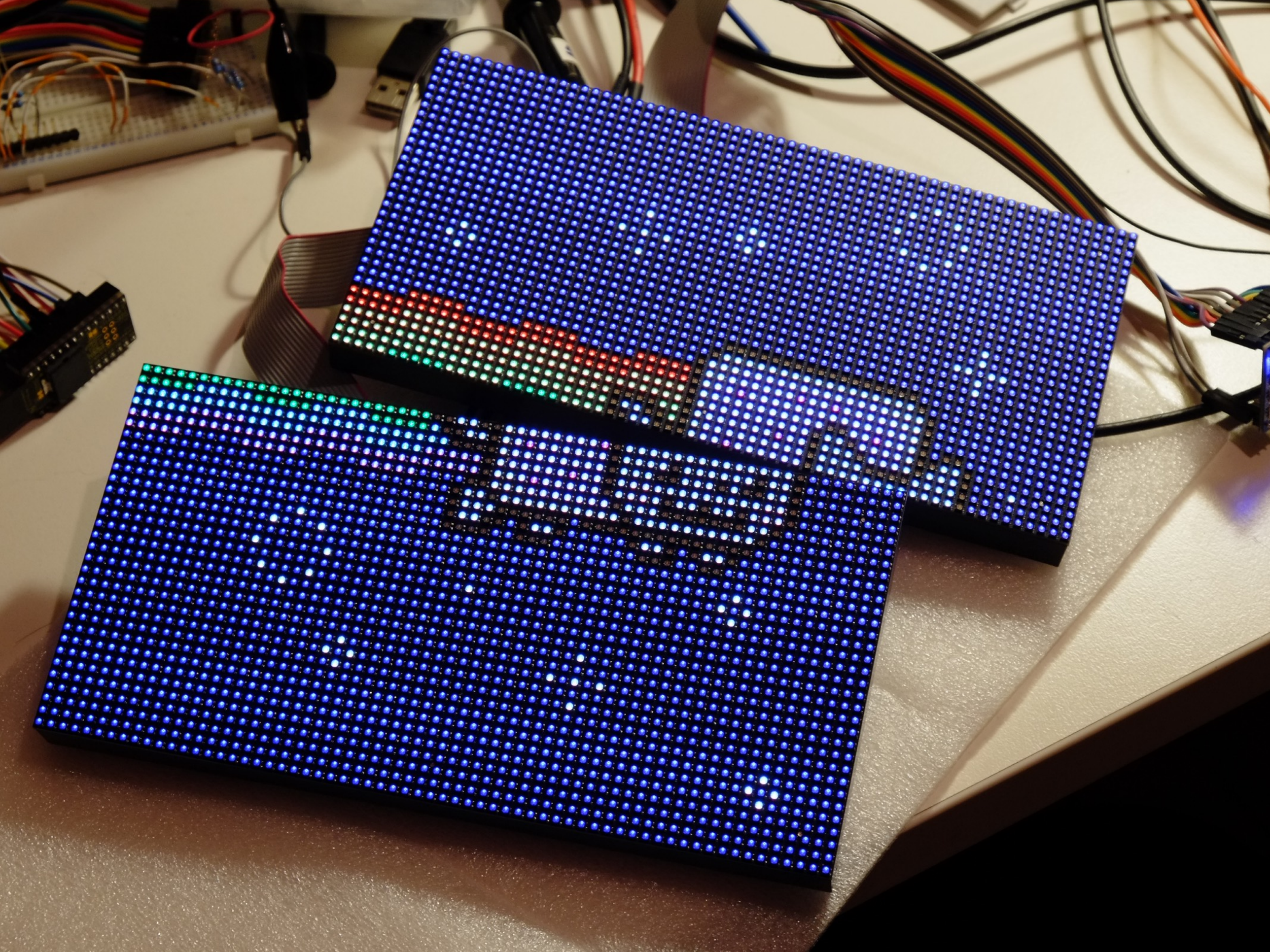
Fast peripheral control from Raspberry Pi and friends

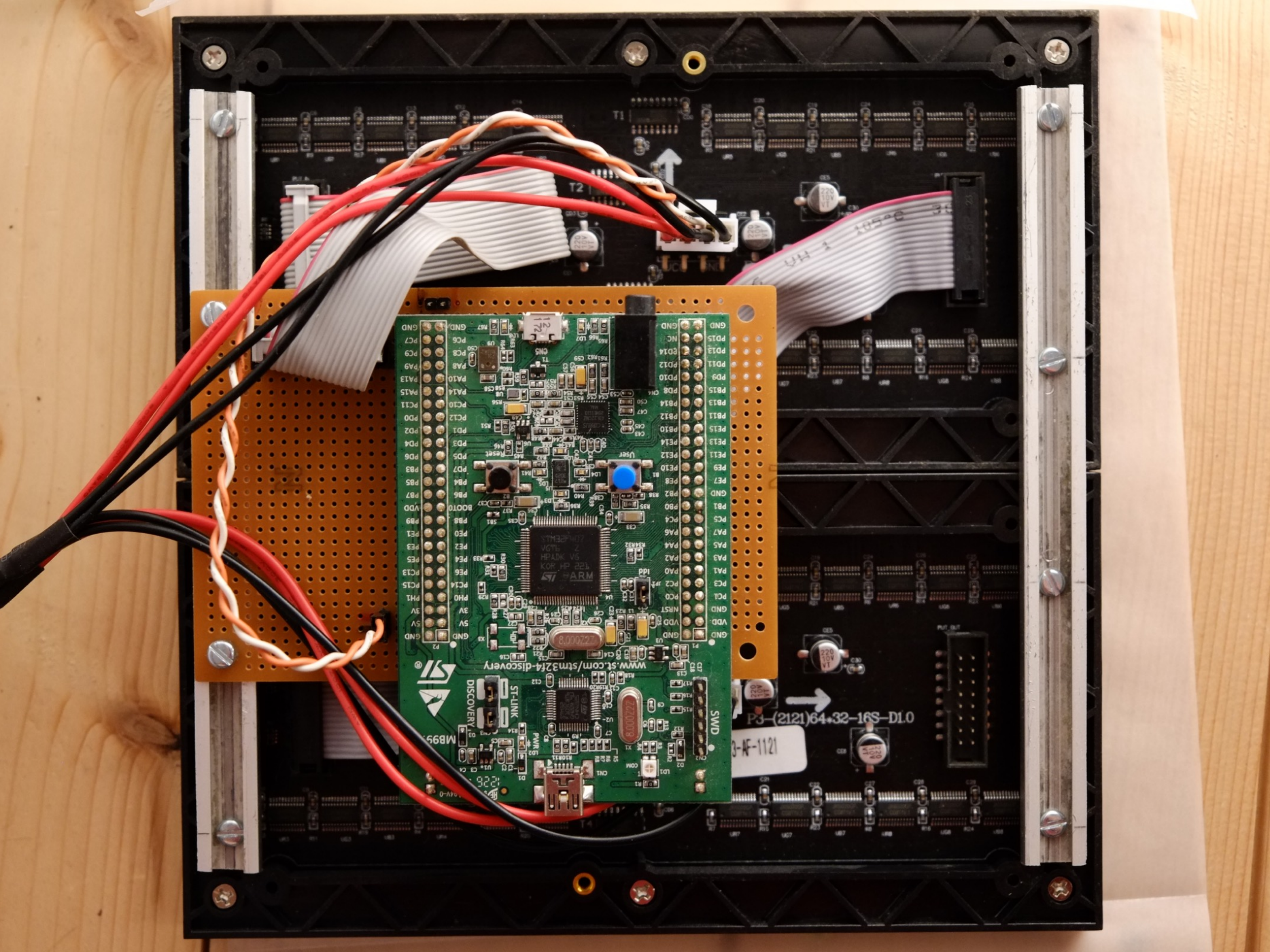
Matt Evans — Hackaday Belgrade 2018

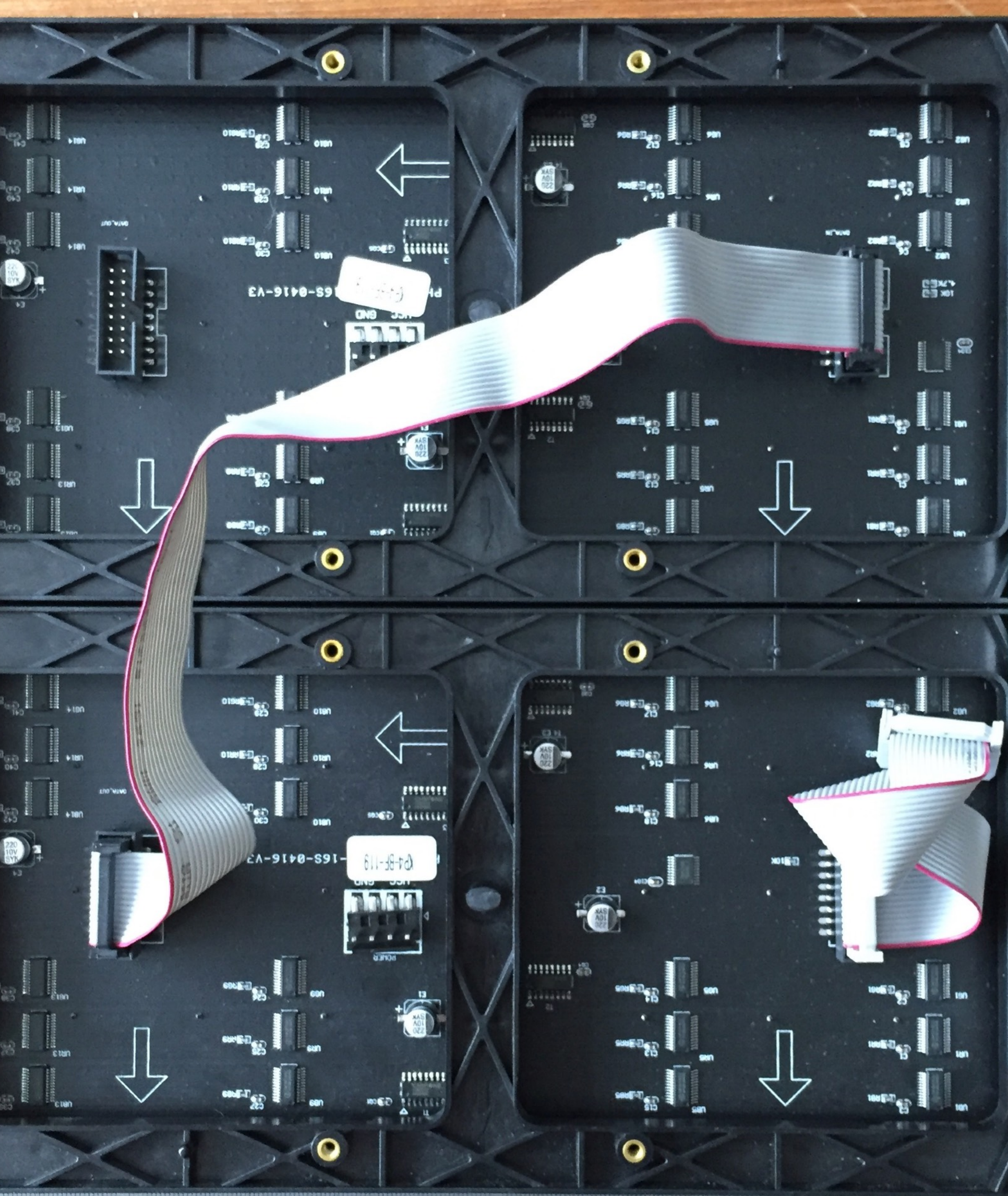
LED panels are amazeballs fun

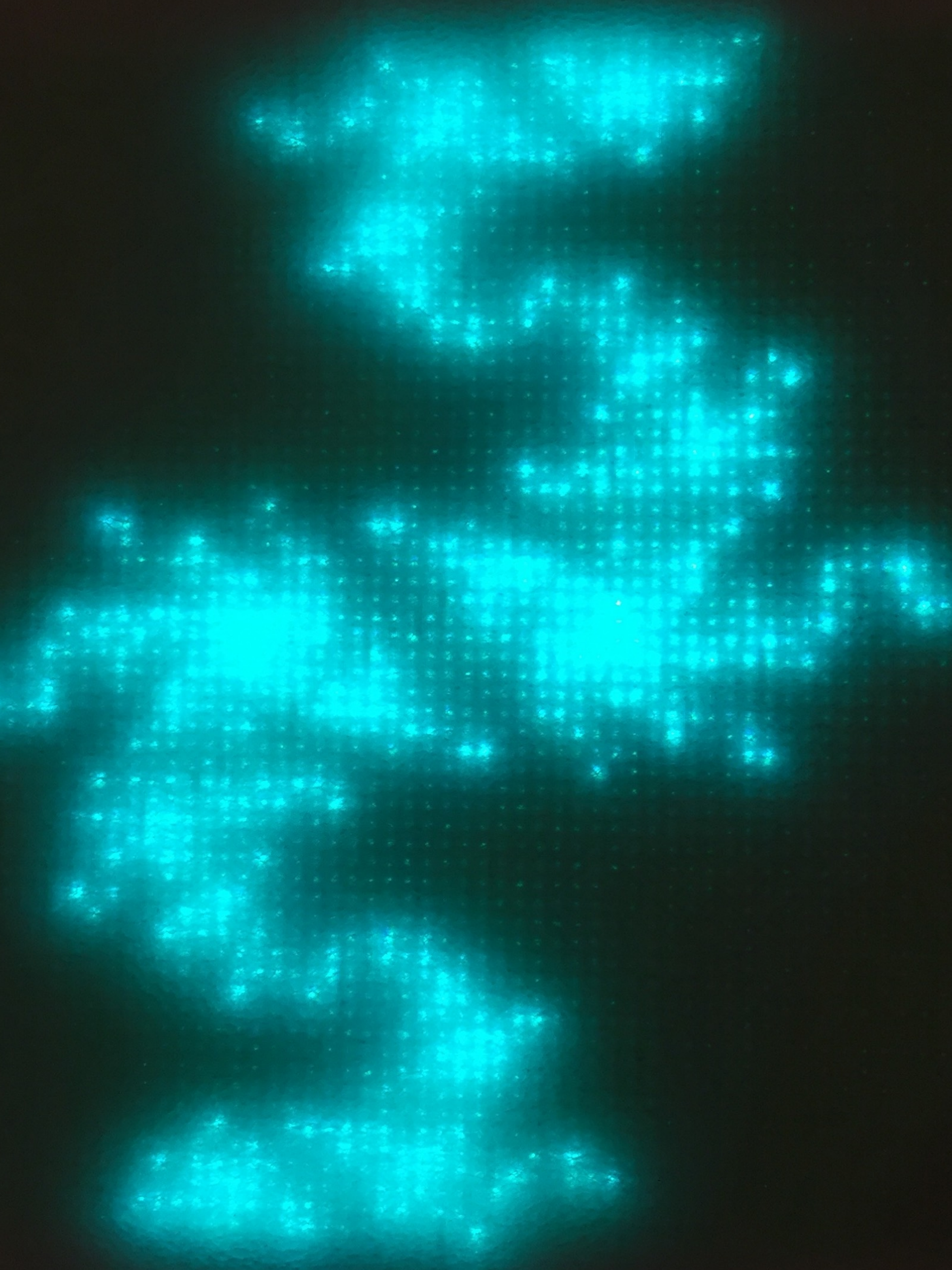
Many SMD RGB LEDs + constant-current driver

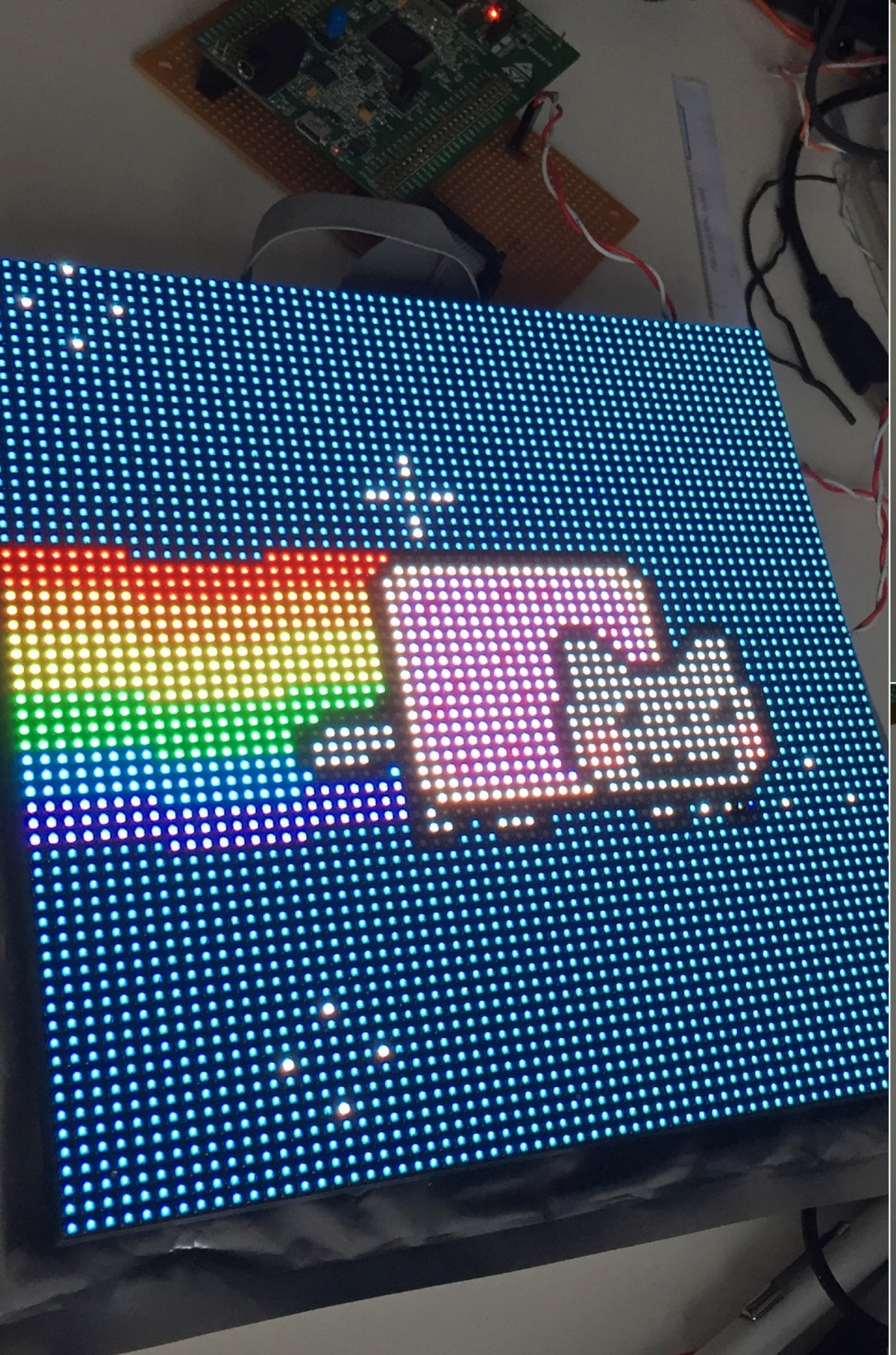
Available in many sizes, e.g. 32x16, 32x32, 64x32, 64x64, and pixel pitch




P3-(2121)64*32-16S-D1.0
www.st.com/stm32f4-discovery
STM32F407
VGT6
ARM
DISCOVERY
ST-LINK
SWD
User
Reset


STAINLESS STEEL





LED panels from a Raspberry Pi

- Great library for driving LED panels from Raspberry Pi:
  - https://github.com/hzeller/rpi-rgb-led-matrix
- But, doesn’t use DMA — ***it bitbangs, software loop***
- "The system needs constant CPU … roughly 30-40% of one core."
- To avoid flicker: “If you have a loaded system … you can ***reserve one core*** just for the refresh of the display” 😫

HUB75 panel interface

(it's all a big shift register)

Pins:

CLK

R0,G0,B0

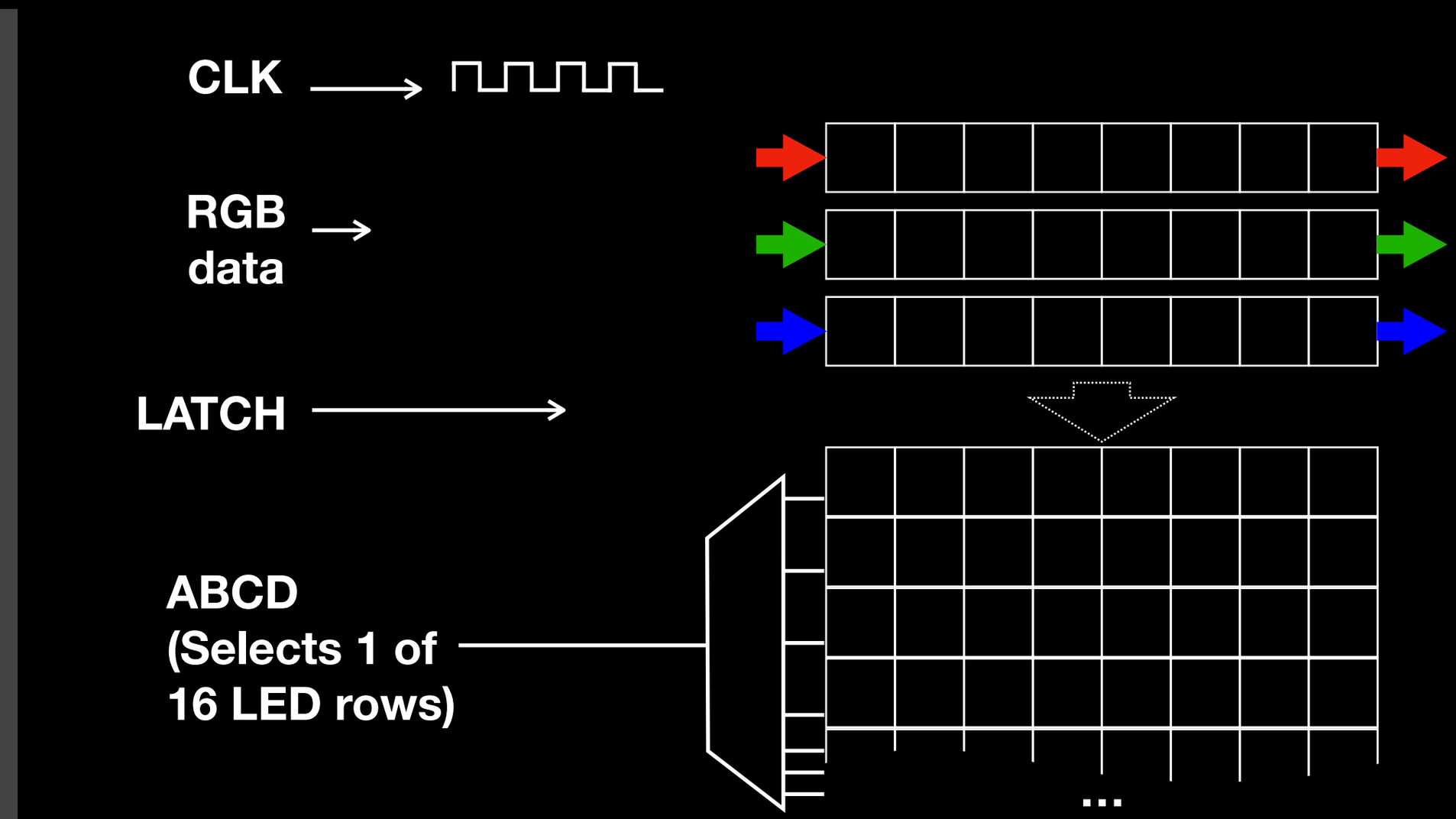
R1,G1,B1

LATCH

A,B,C,D

OE

(Usually 5V!)



HUB75 panel interface

(it's all a big shift register)

Pins:

CLK

R0,G0,B0

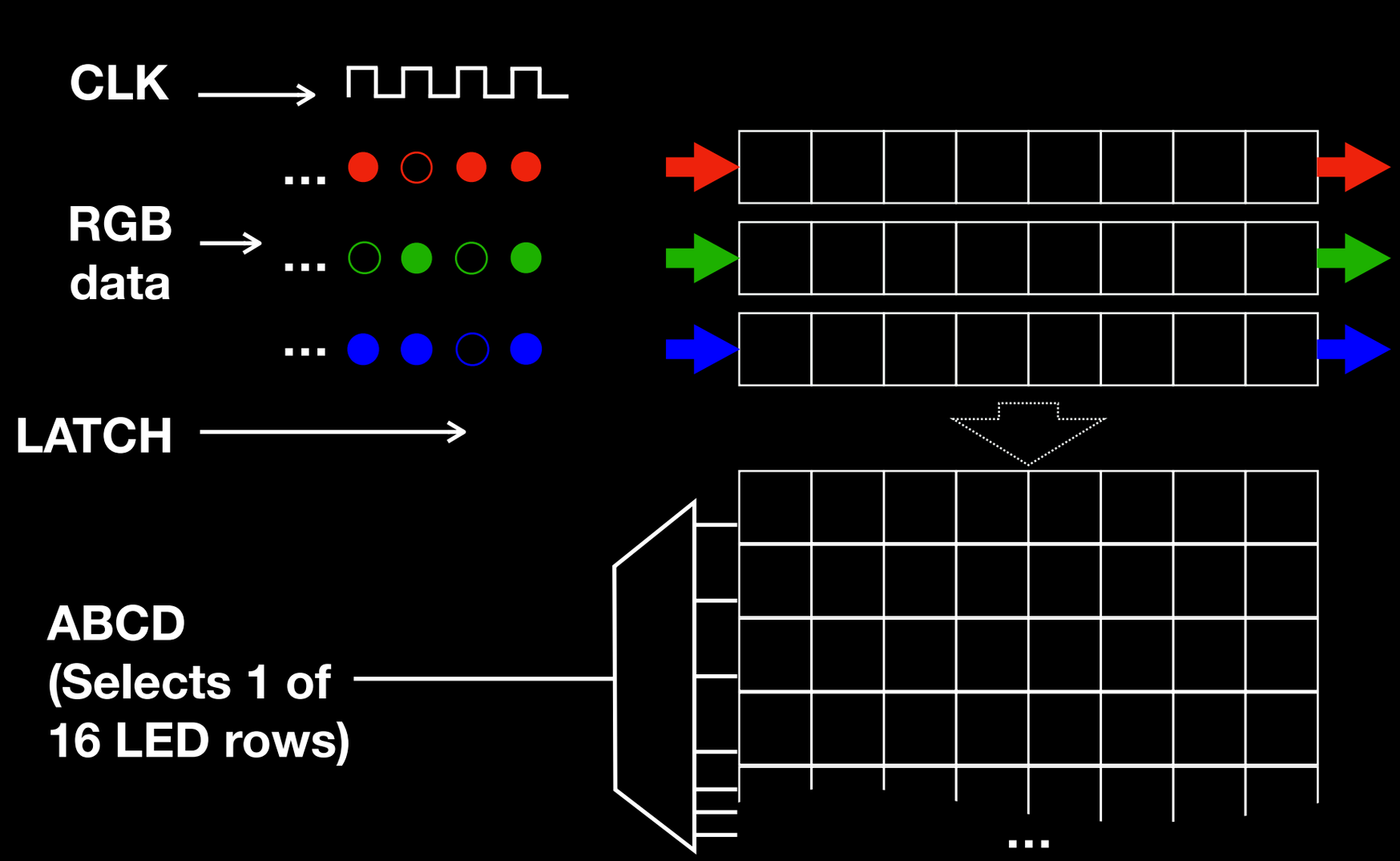
R1,G1,B1

LATCH

A,B,C,D

OE

(Usually 5V!)



HUB75 panel interface

(it's all a big shift register)

Pins:

CLK

R0,G0,B0

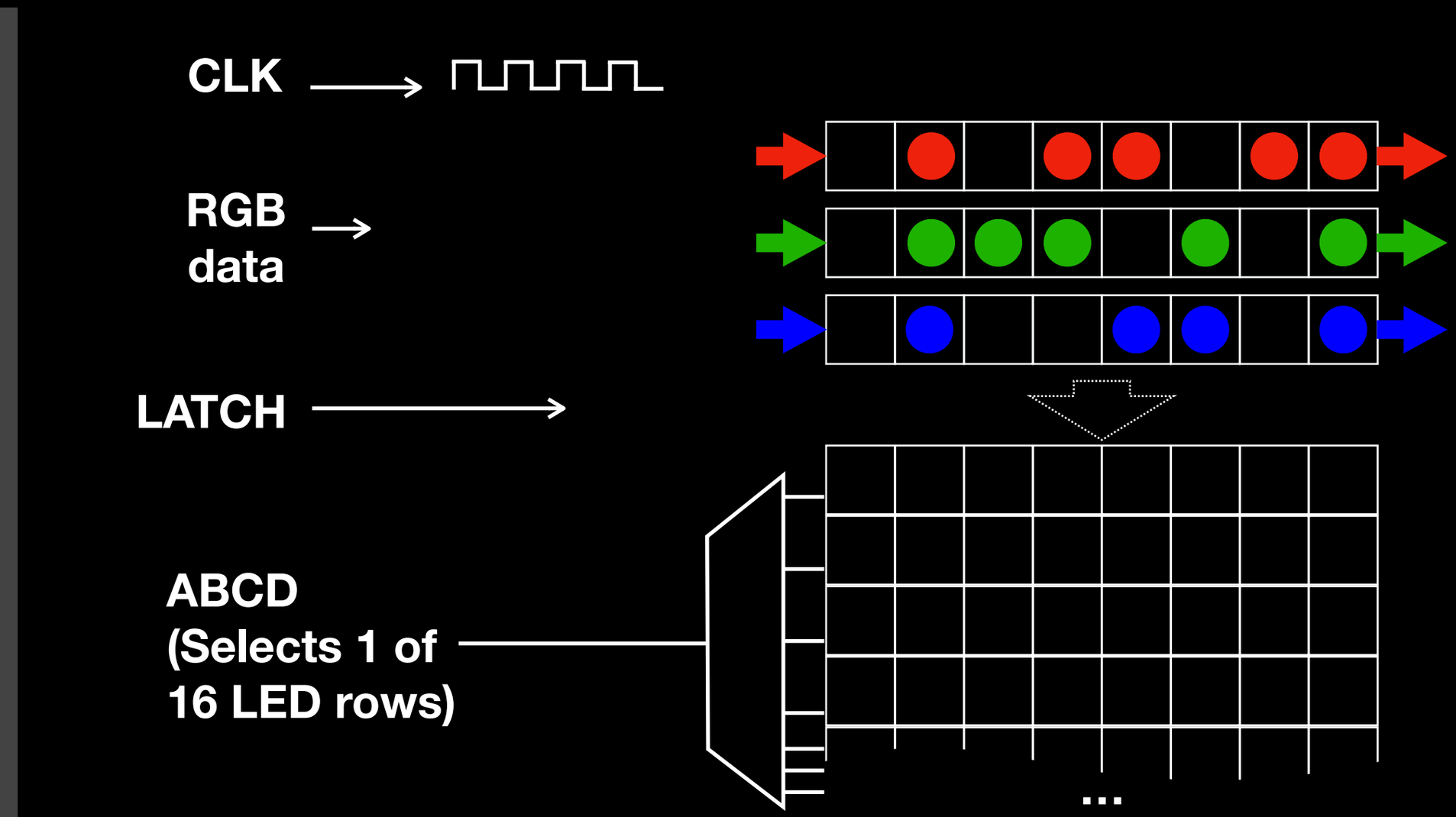
R1,G1,B1

LATCH

A,B,C,D

OE

(Usually 5V!)



HUB75 panel interface

(it's all a big shift register)

Pins:

CLK

R0,G0,B0

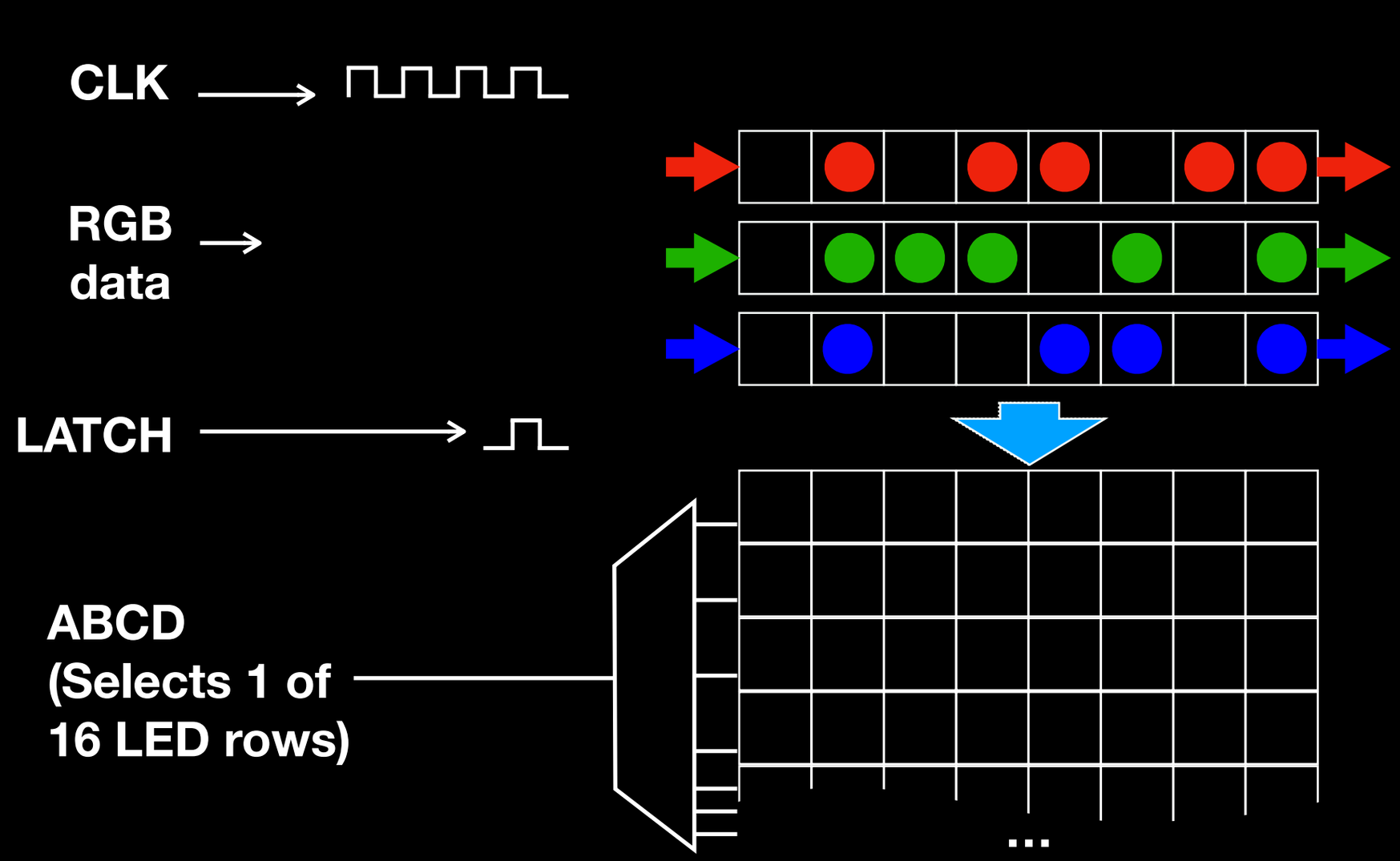
R1,G1,B1

LATCH

A,B,C,D

OE

(Usually 5V!)



HUB75 panel interface

(it's all a big shift register)

Pins:

CLK

R0,G0,B0

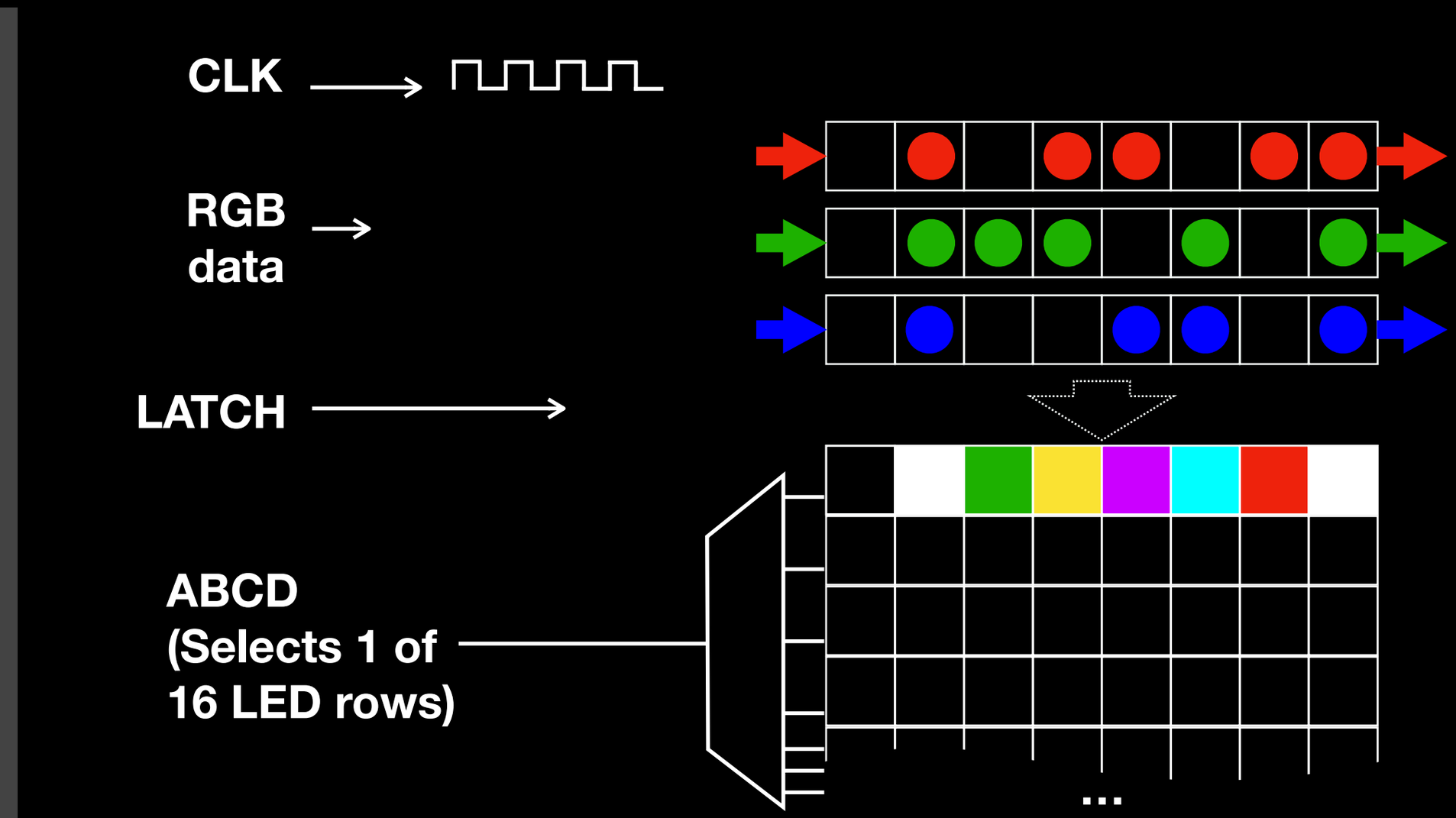
R1,G1,B1

LATCH

A,B,C,D

OE

(Usually 5V!)



Display Parallel Interface (DPI)

- Once upon a time, I was attaching an LCD to a RPi using DPI

- Parallel interface designed to drive TFT LCDs from BCM2835 — alternative to HDMI

- 24-bit pixel output (+ pixel clock, + sync bits) high-speed digital output, 3.3V CMOS

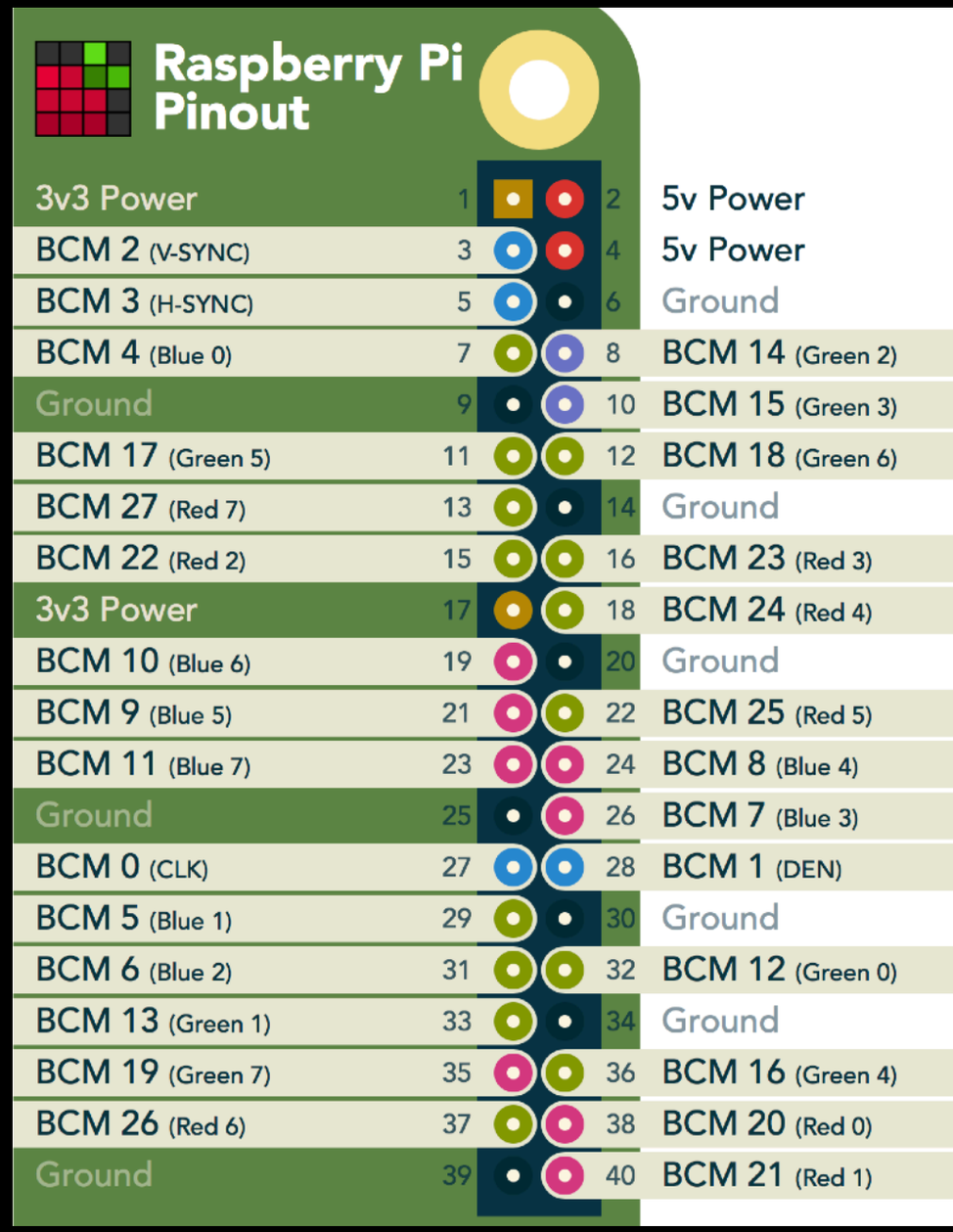


Image from pinouts.xyz

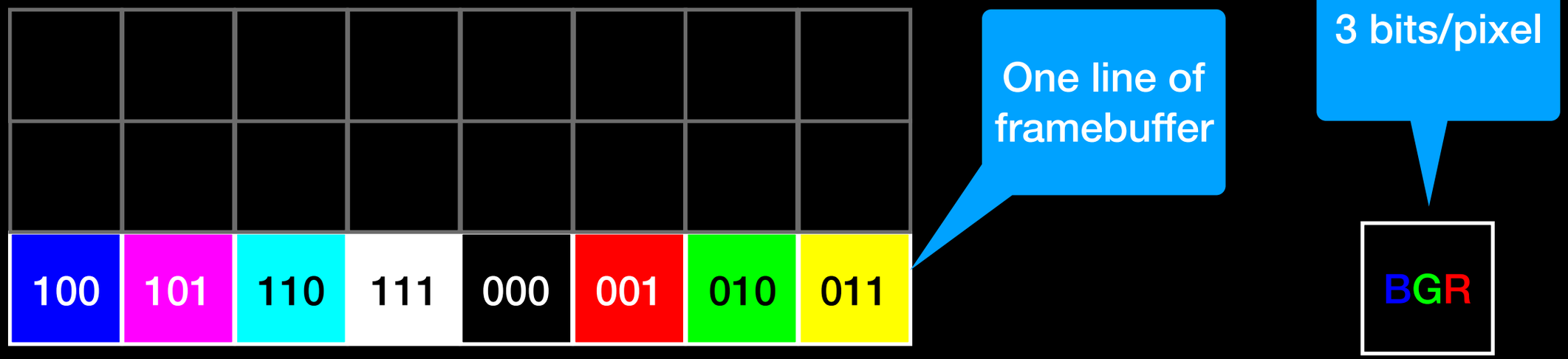
I haz an idea!

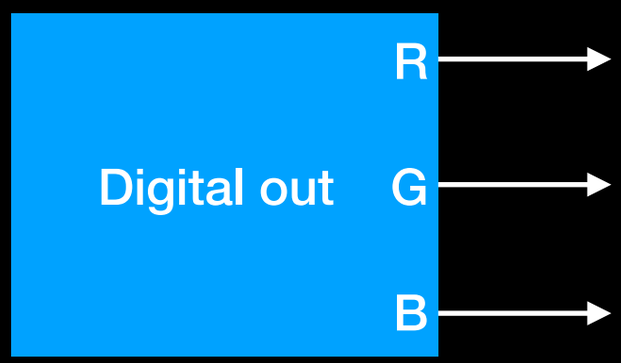
Put *pixel patterns* in the video framebuffer that send digital *data patterns* to DPI output pins!

No CPU overhead to display it

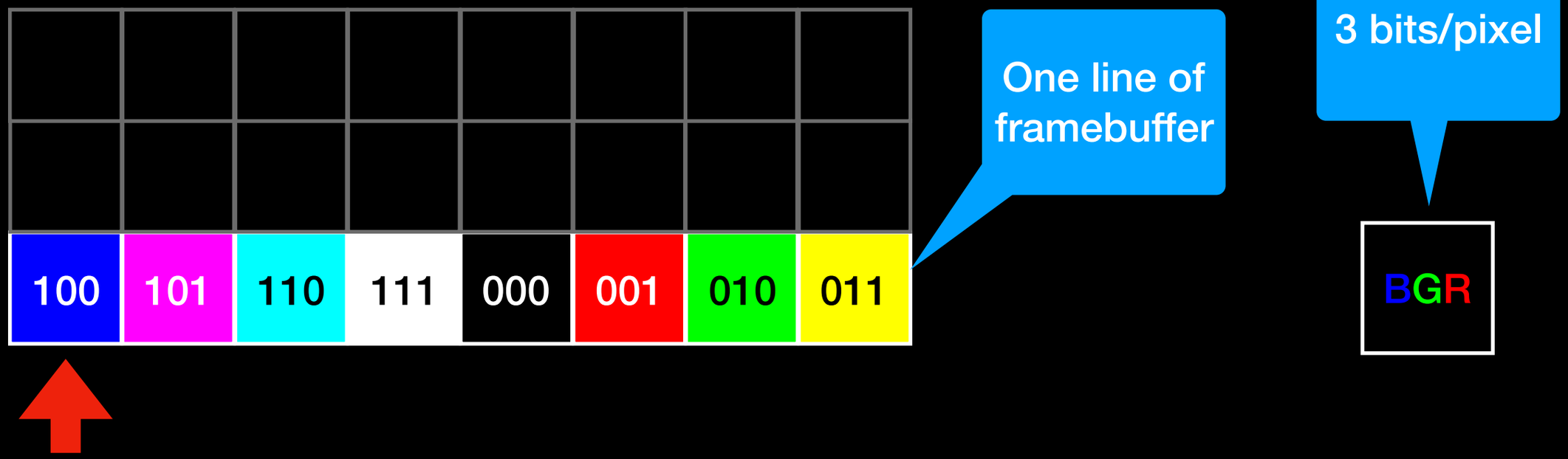
Guaranteed not to hiccup

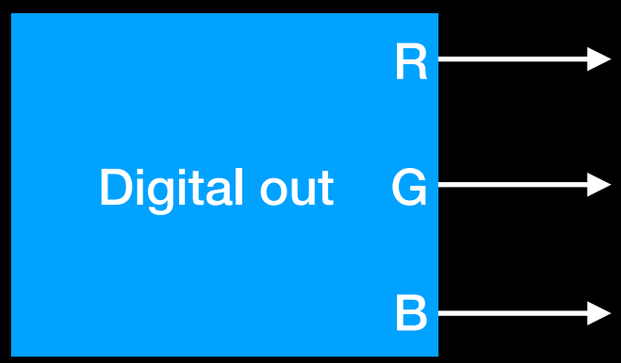
Parallel video out



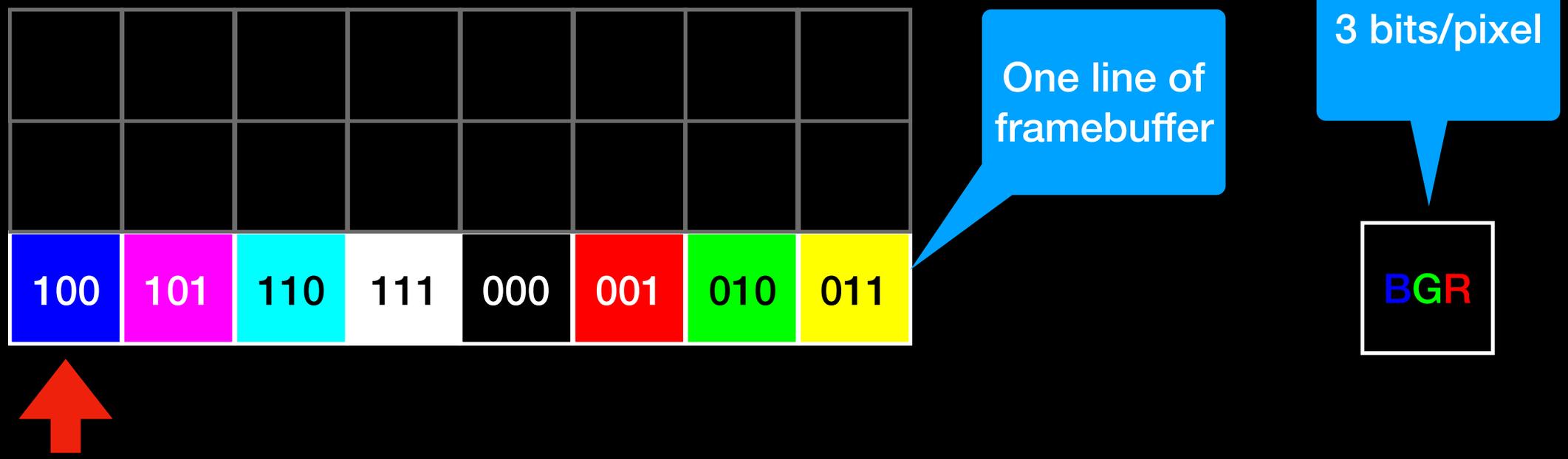


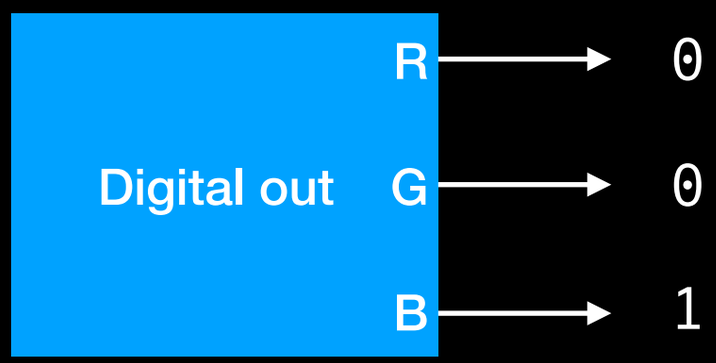
Parallel video out



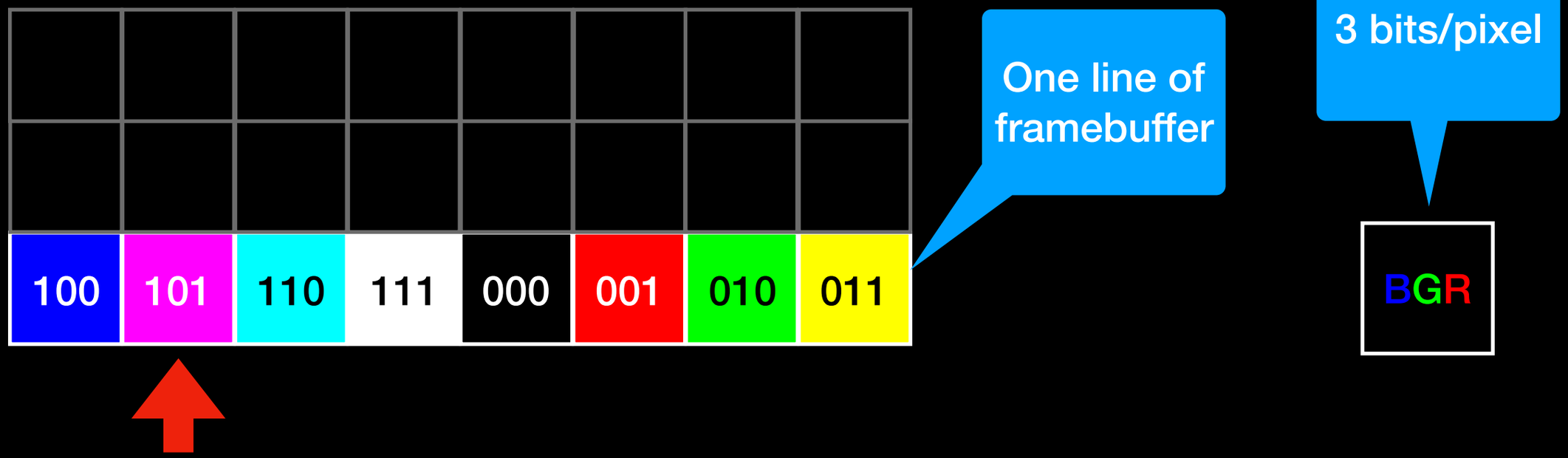


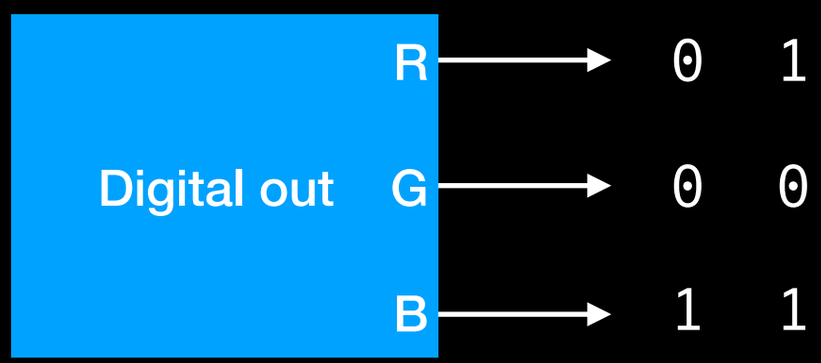
Parallel video out



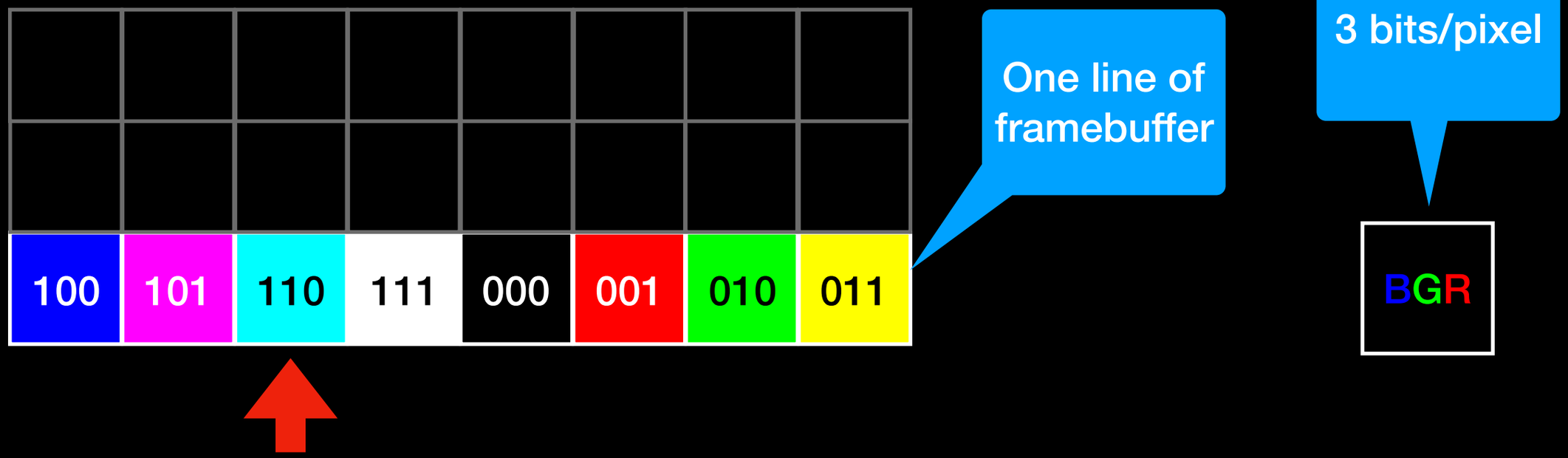


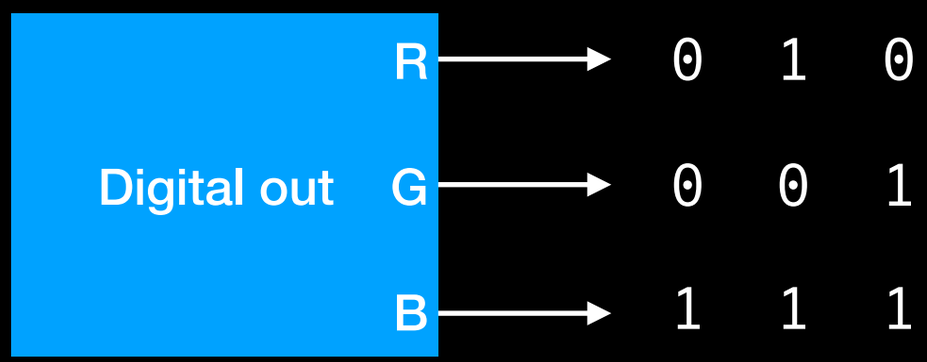
Parallel video out



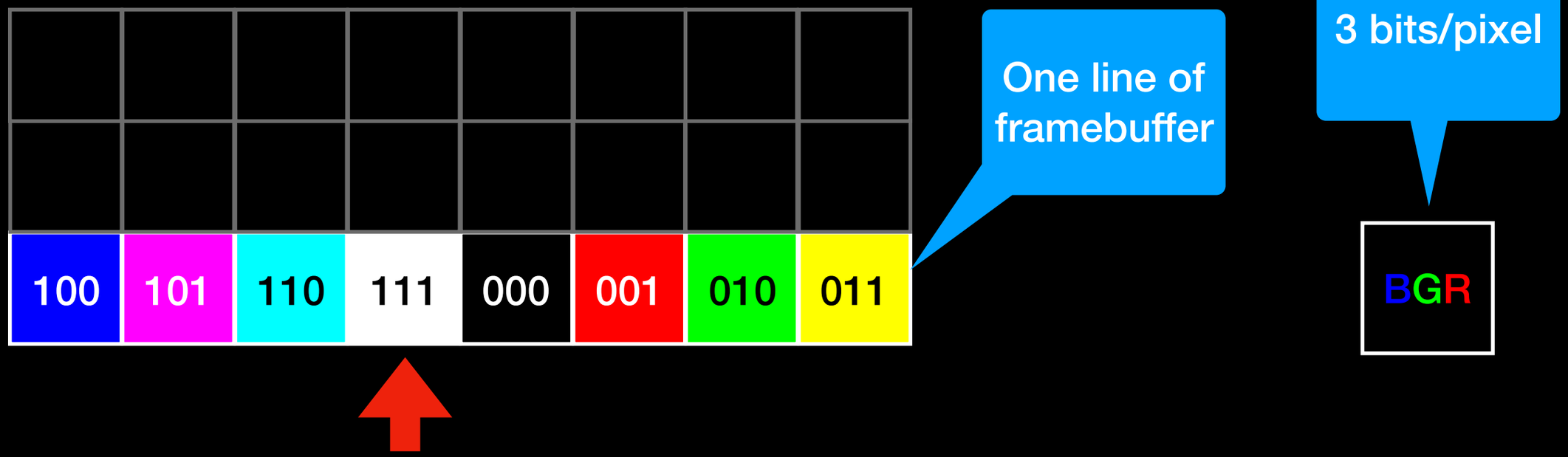


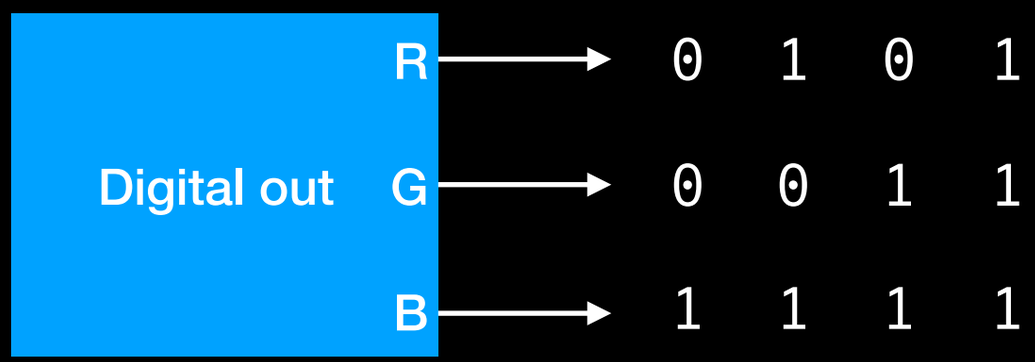
Parallel video out



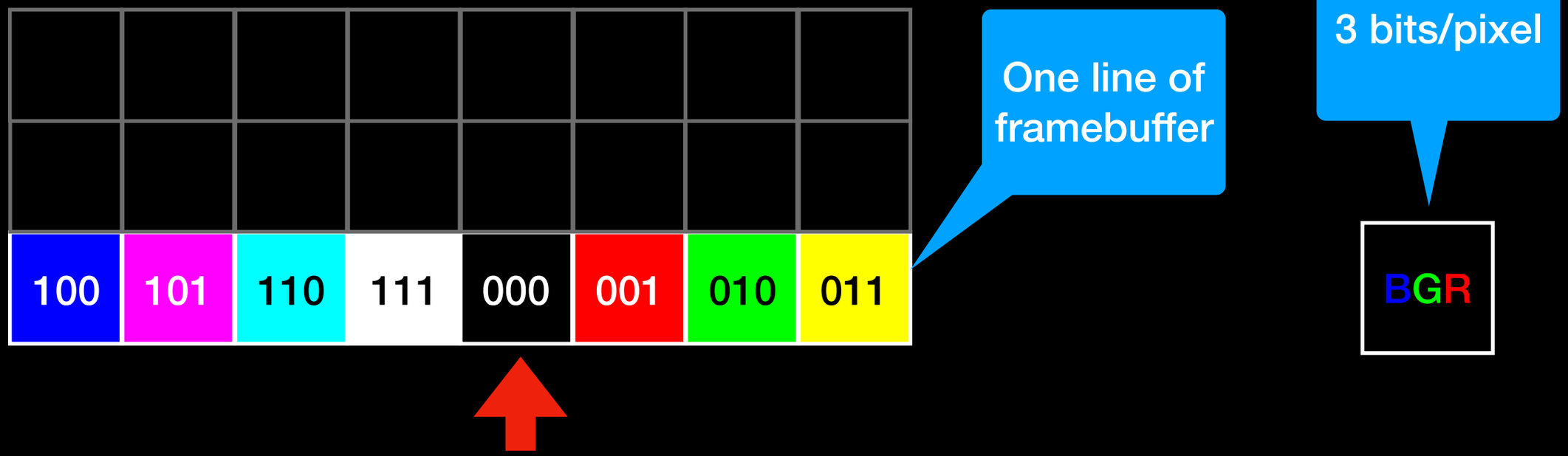


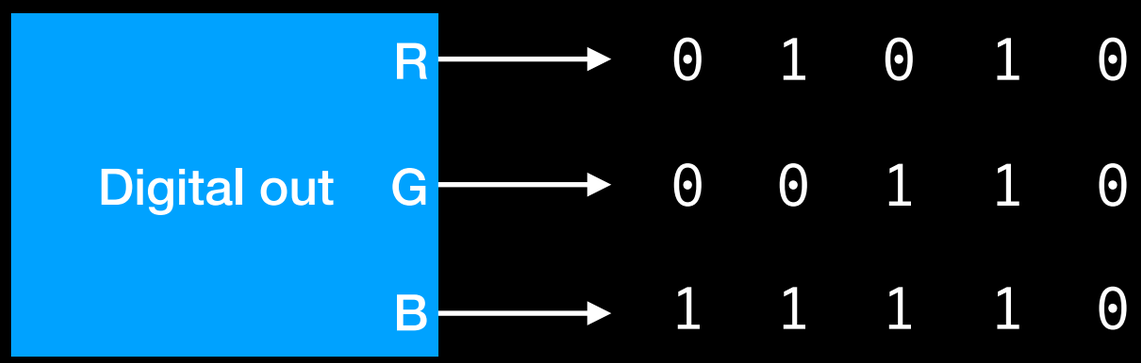
Parallel video out



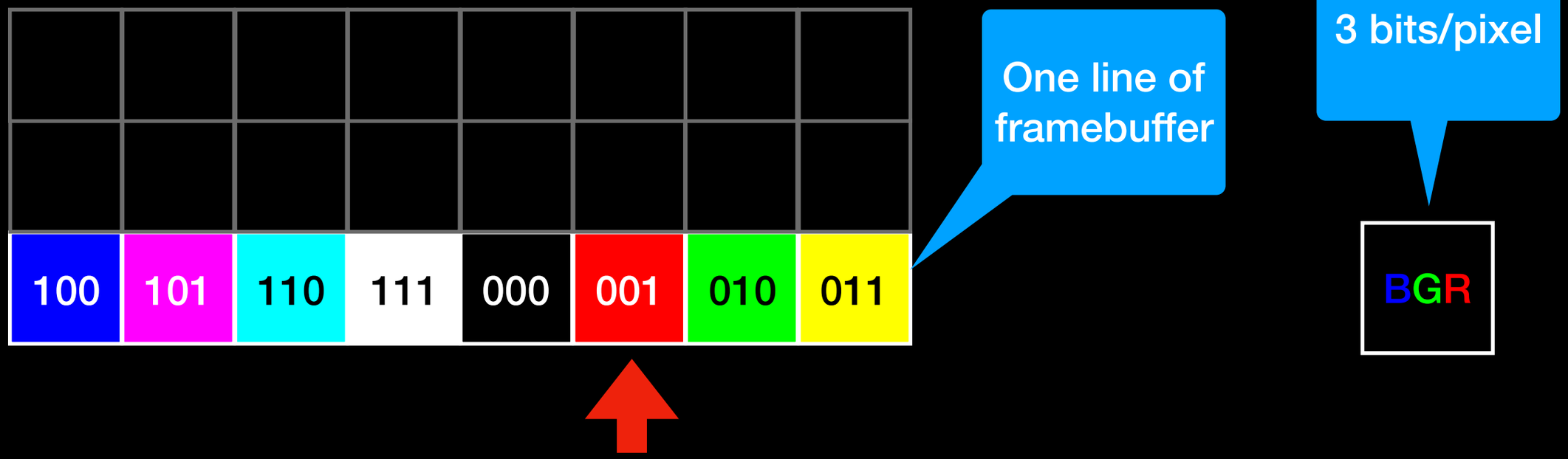


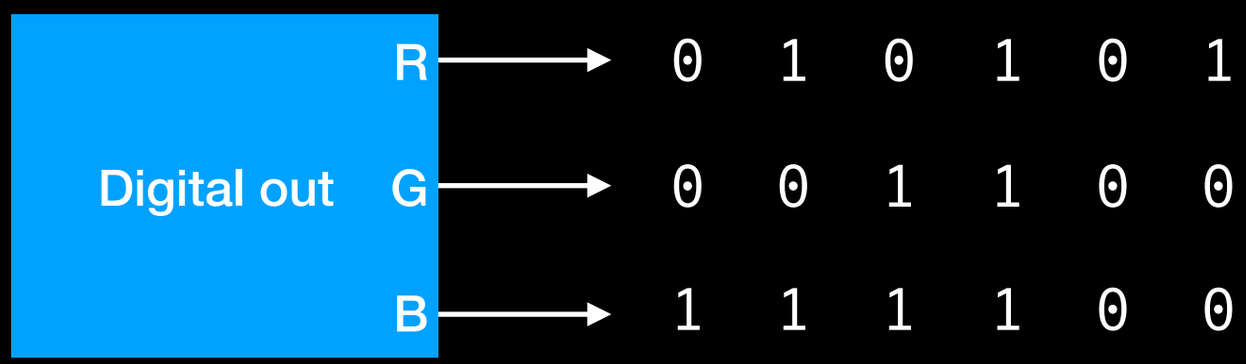
Parallel video out



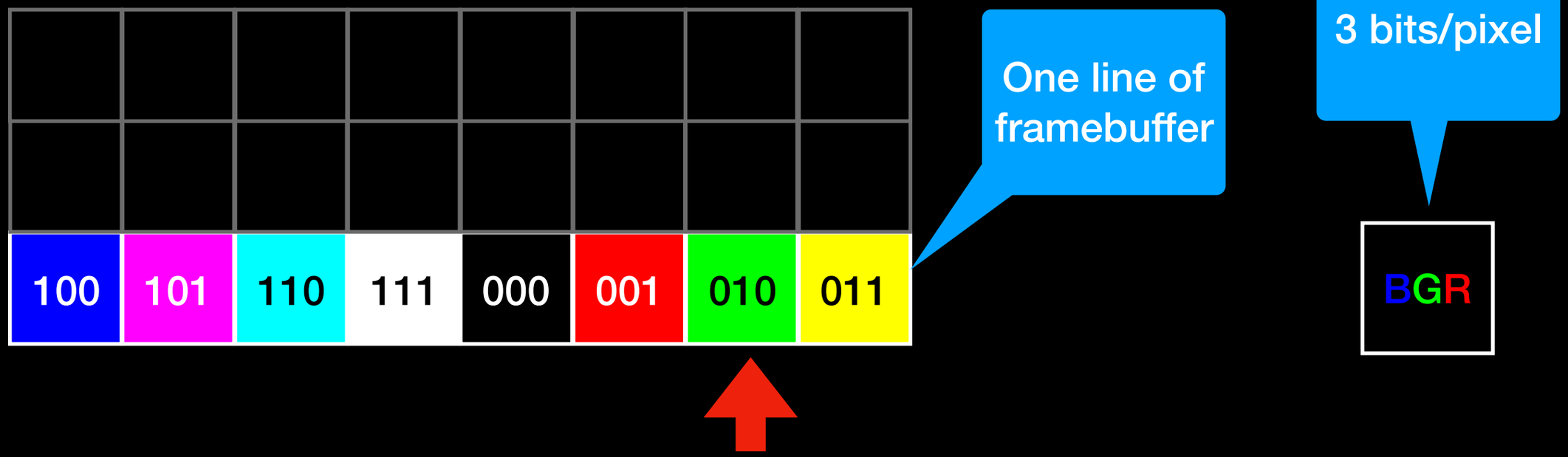


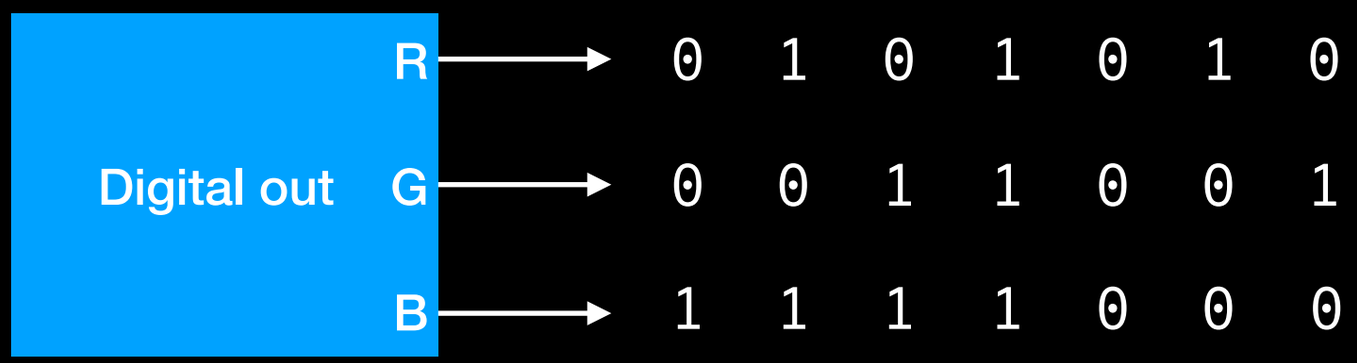
Parallel video out



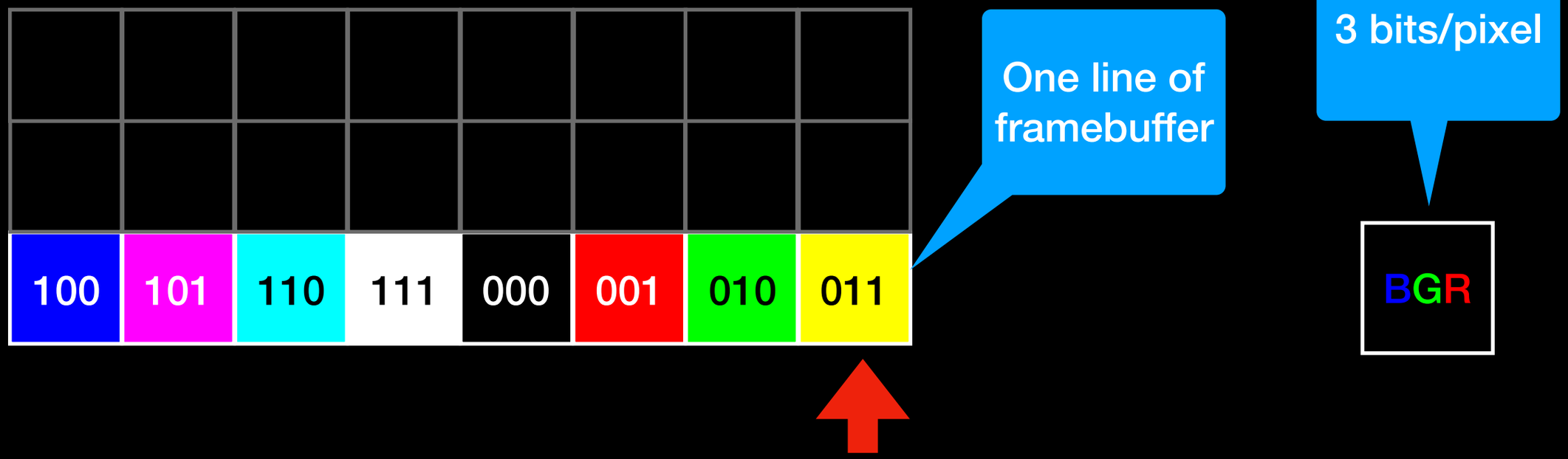


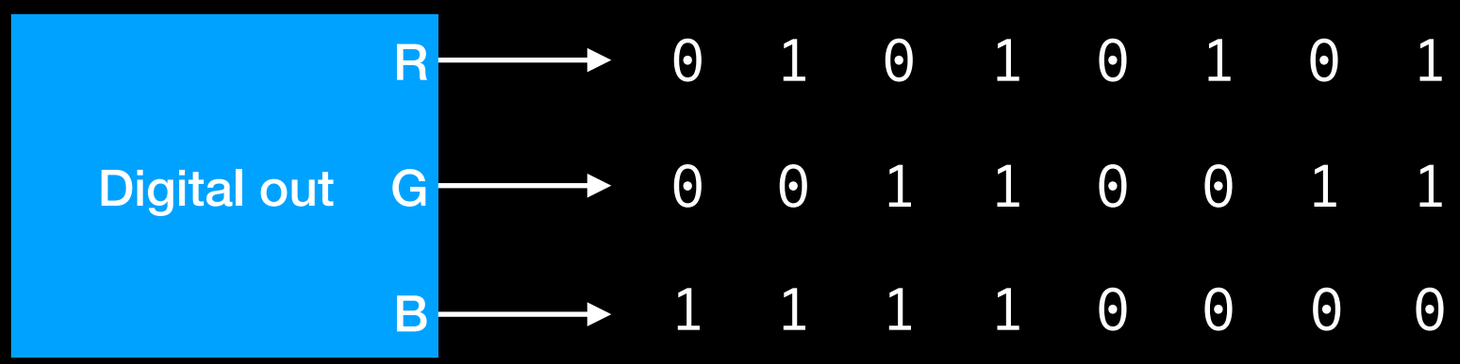
Parallel video out





Parallel video out

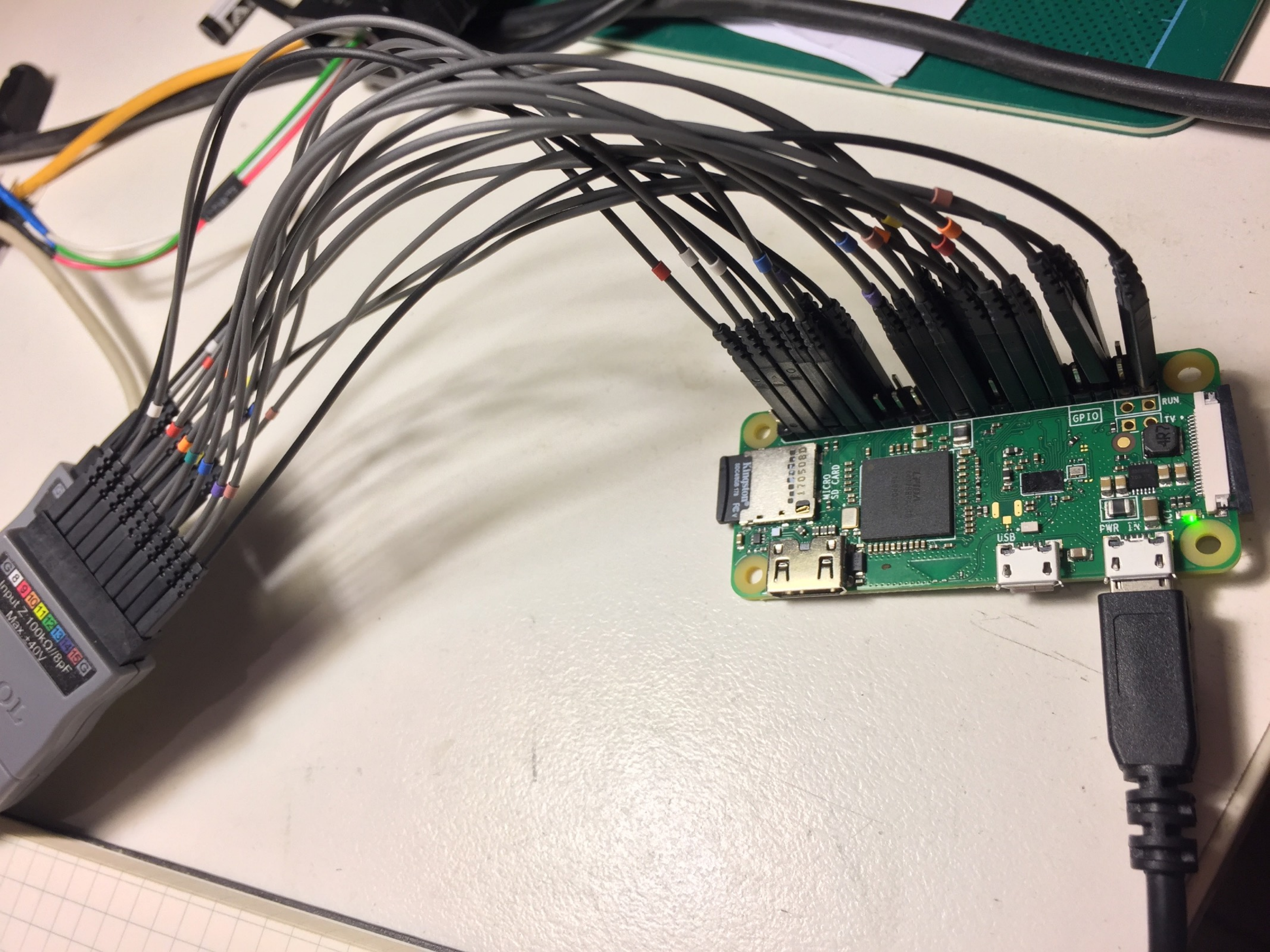




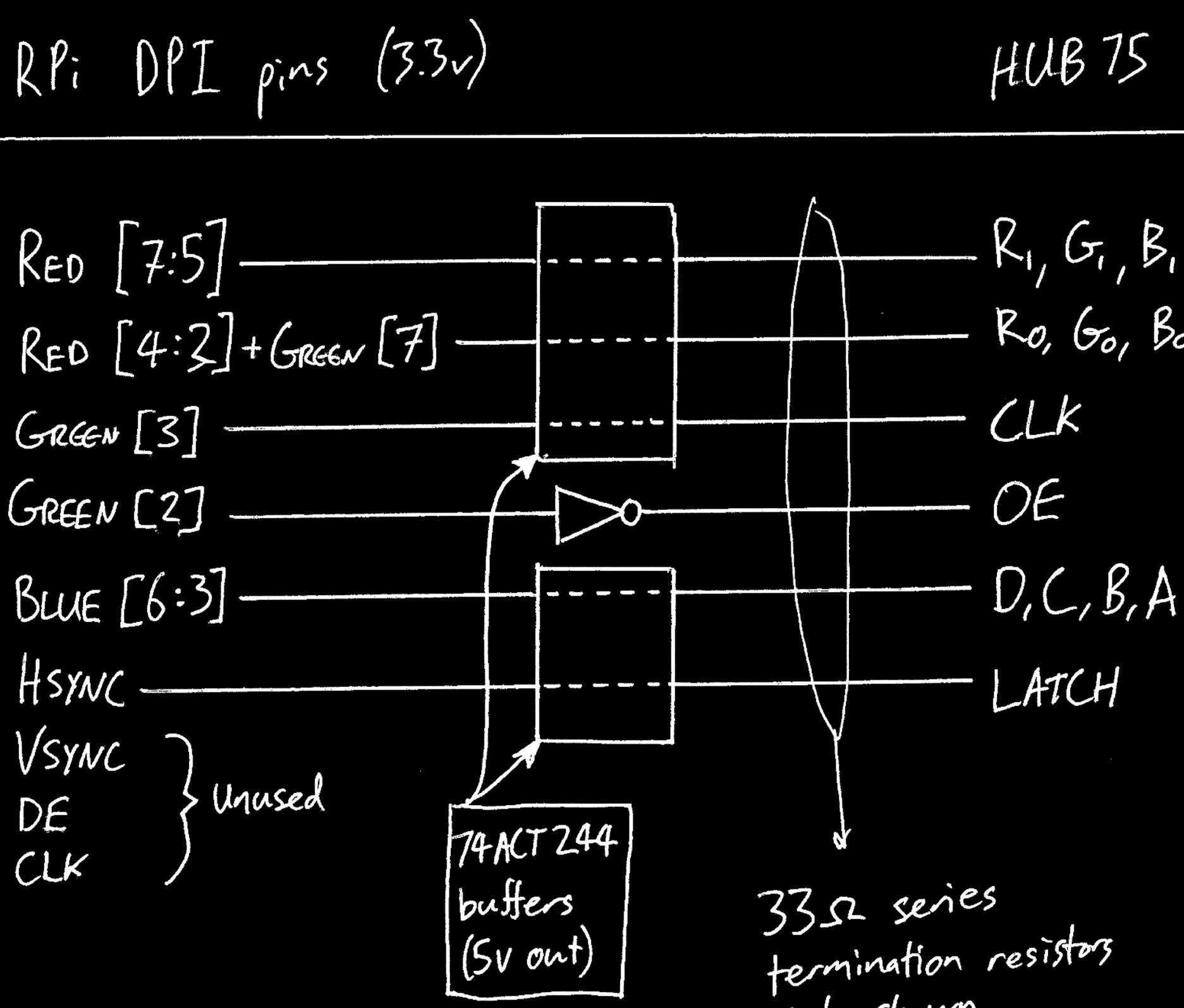
Misusing video outputs

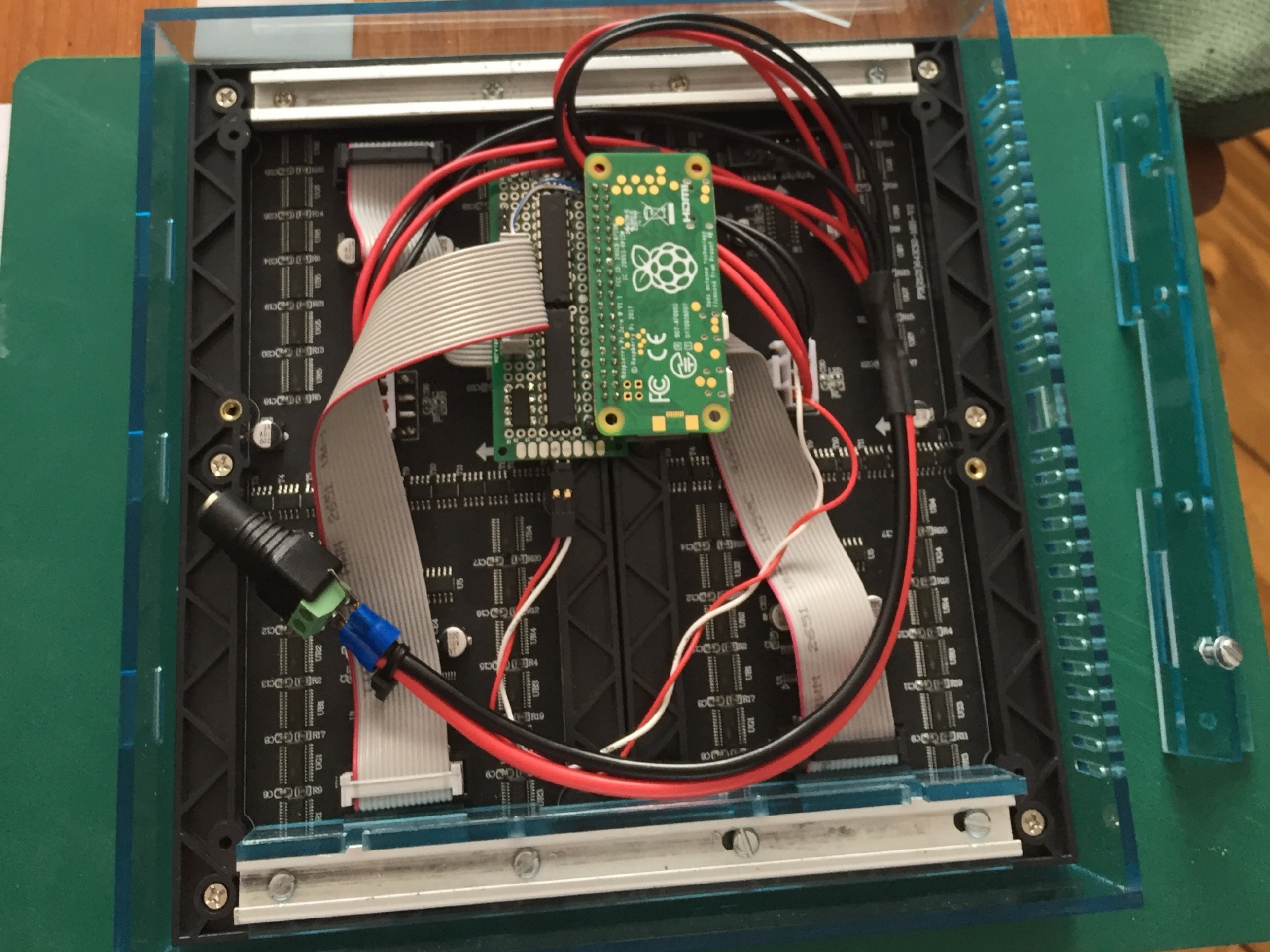
- You may have seen people using VGA for analog out:
  - Tempest for Eliza: AM radio transmitter
  - Fabrice Bellard's DVB-T transmitter
  - osmo-fl2k: Using FL2000 USB dongle as SDR transmitter

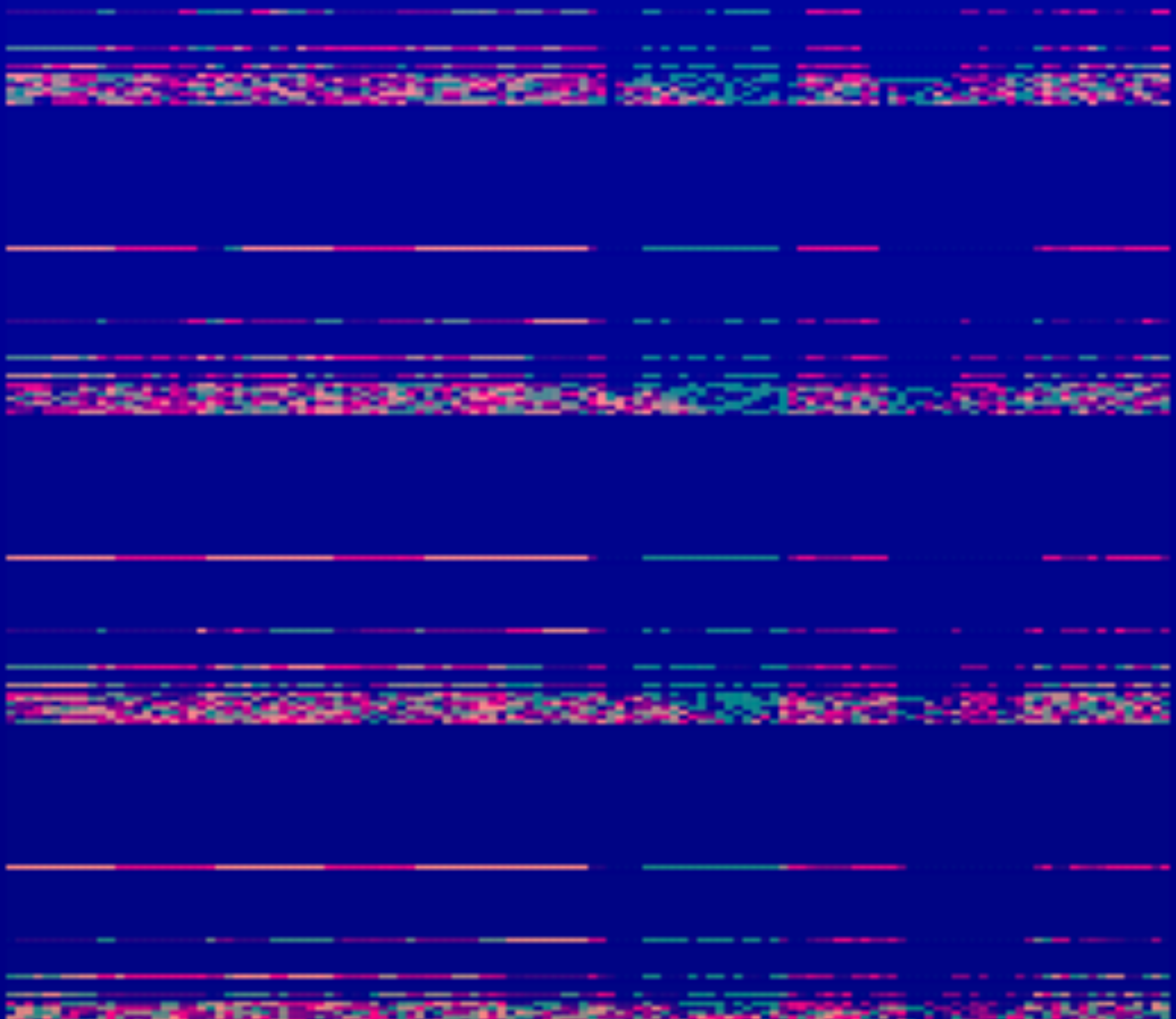
- Haven't found any projects using *digital* video out for other things

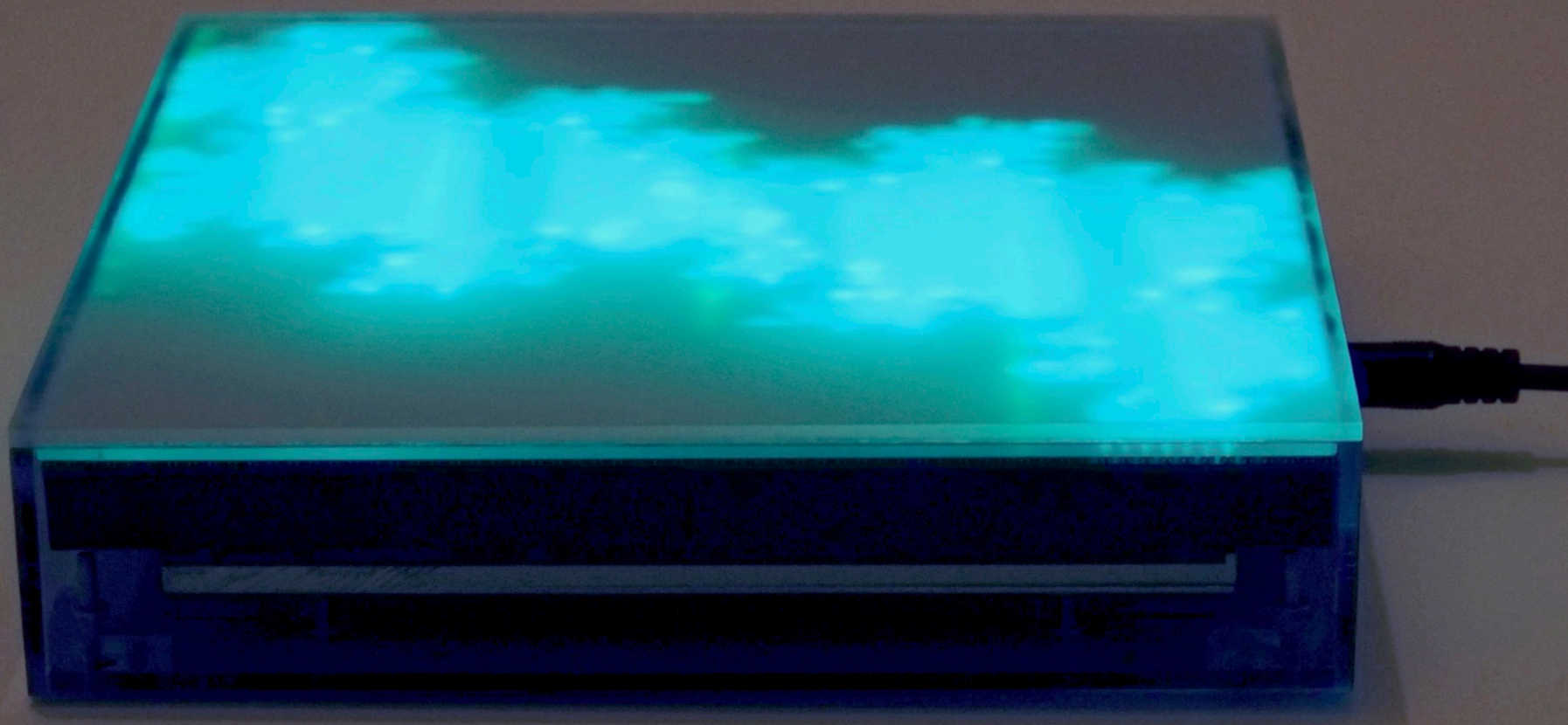

GPIO
RUN
TV
4R7
Kingston
MICRO SD CARD
USB
PWR IN
Input Z: 100kΩ//8pF
Max: ±40V

Wiring HUB75 to DPI









Tx: 56.1KB/s
Rx: 17.4KB/s
73%
Sat 30 Sep 19:53
VideoSenderME | Processing 3.3.6
Java
VideoSenderME
MacBook Pro
delete

giffgaff 22:47 70%

Connected to
10.10.10.128

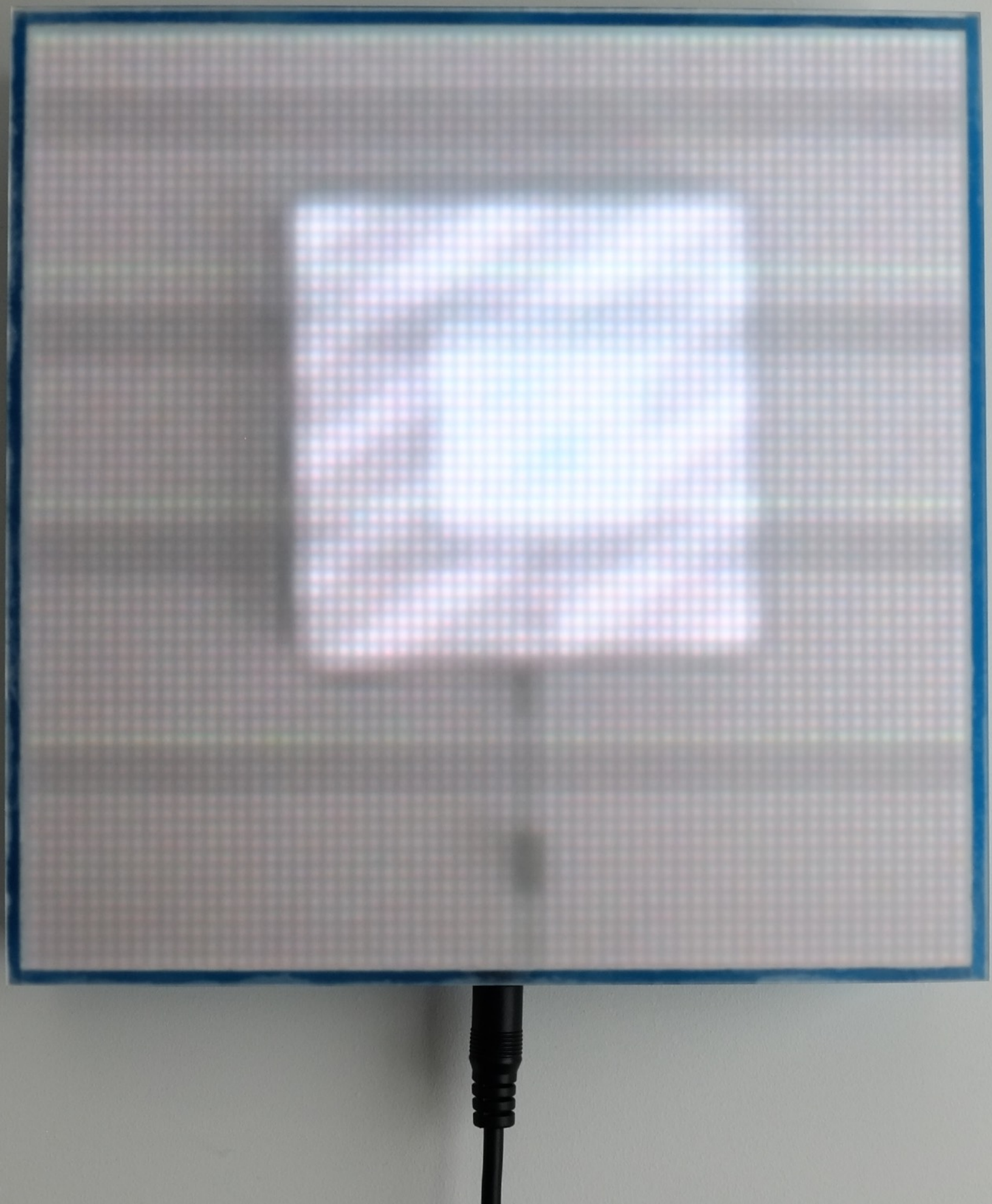
testpi.house

Snapshot

Sending video

Effect

Gamma

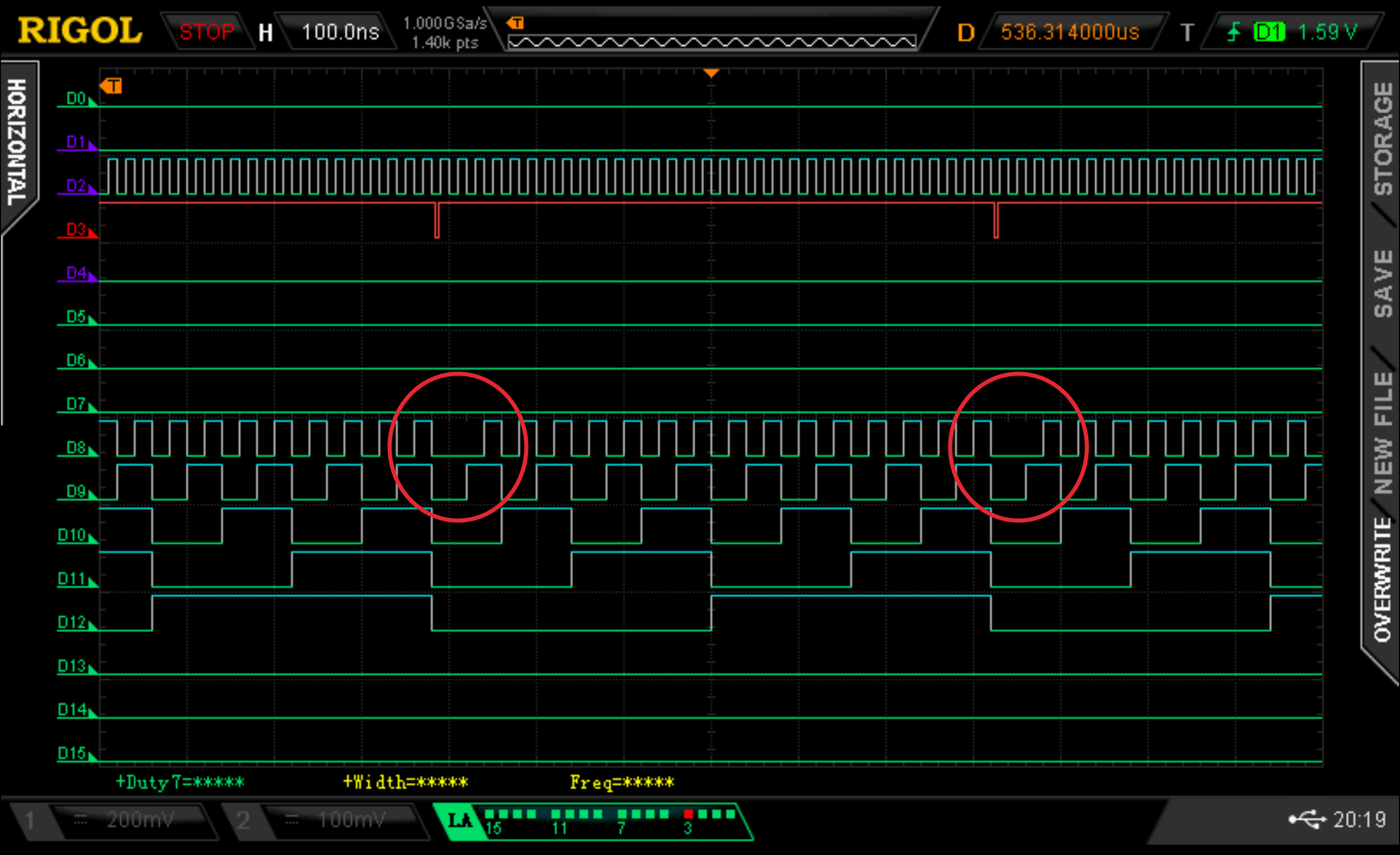


So, it’s perfect! 

Not quite — some quirks

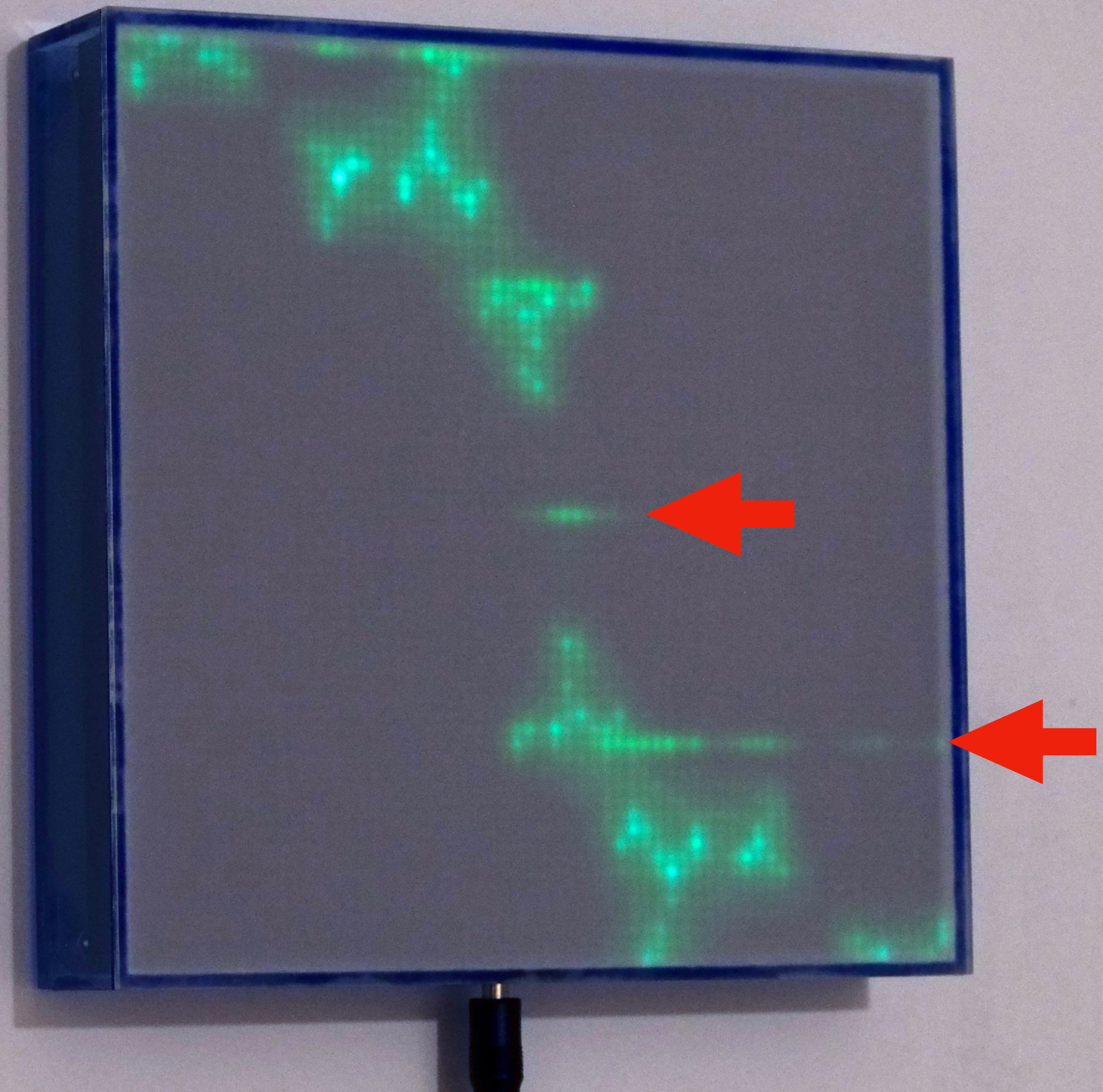
- 24 bit DPI uses *all* GPIOs
  - DeviceTree configures per-pin multiplexing — you don't *have* to use them all
- I used a 16BPP screen mode — I only needed a few bits output
  - But where's my LSB of Blue?

Lost blue bit

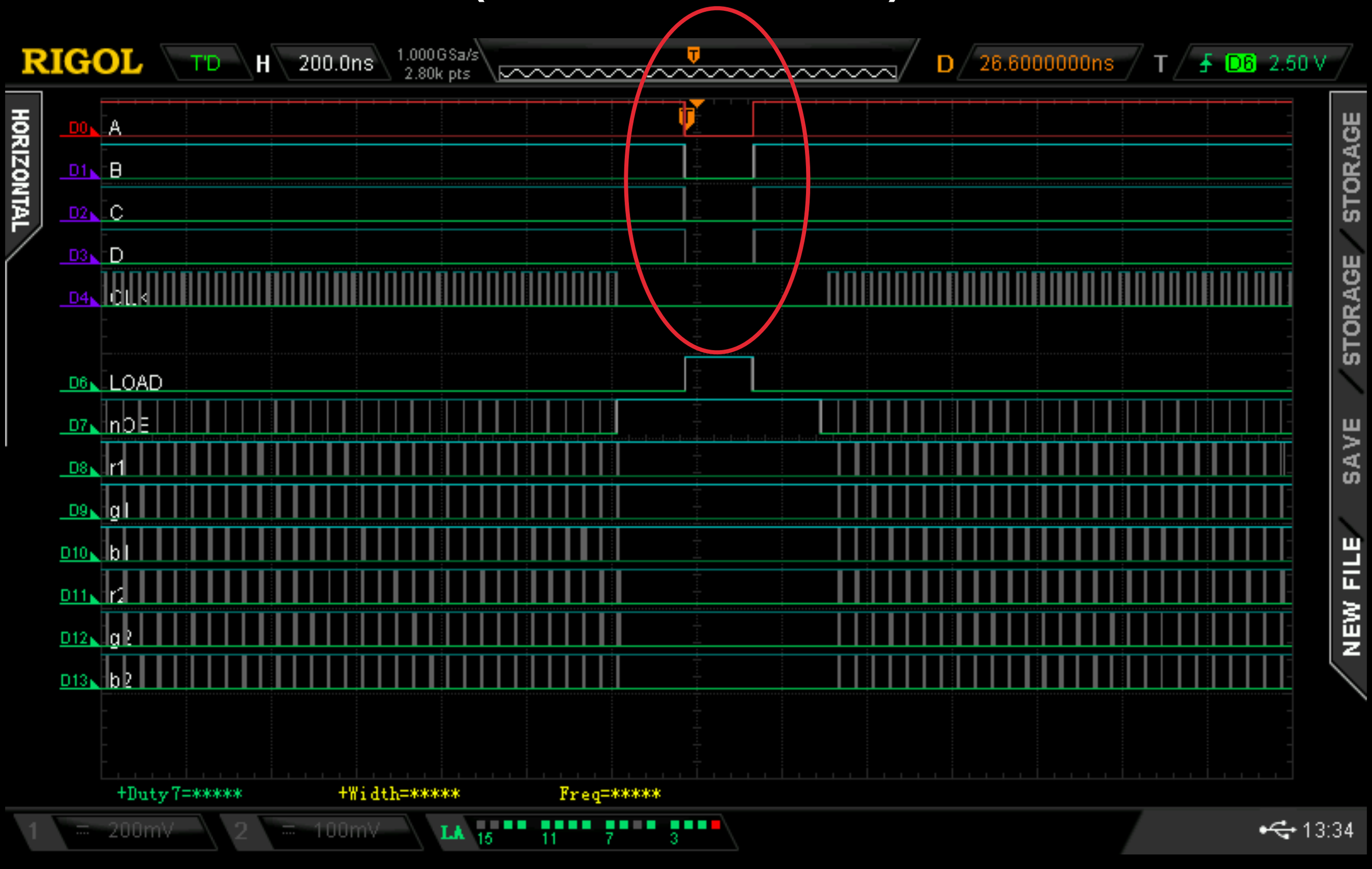


Lost blue bit

- My theory: display controller doing post-framebuffer dithering/colour correction in 16BPP modes
- This does not occur in 32BPP modes — I recommend just using 32BPP!

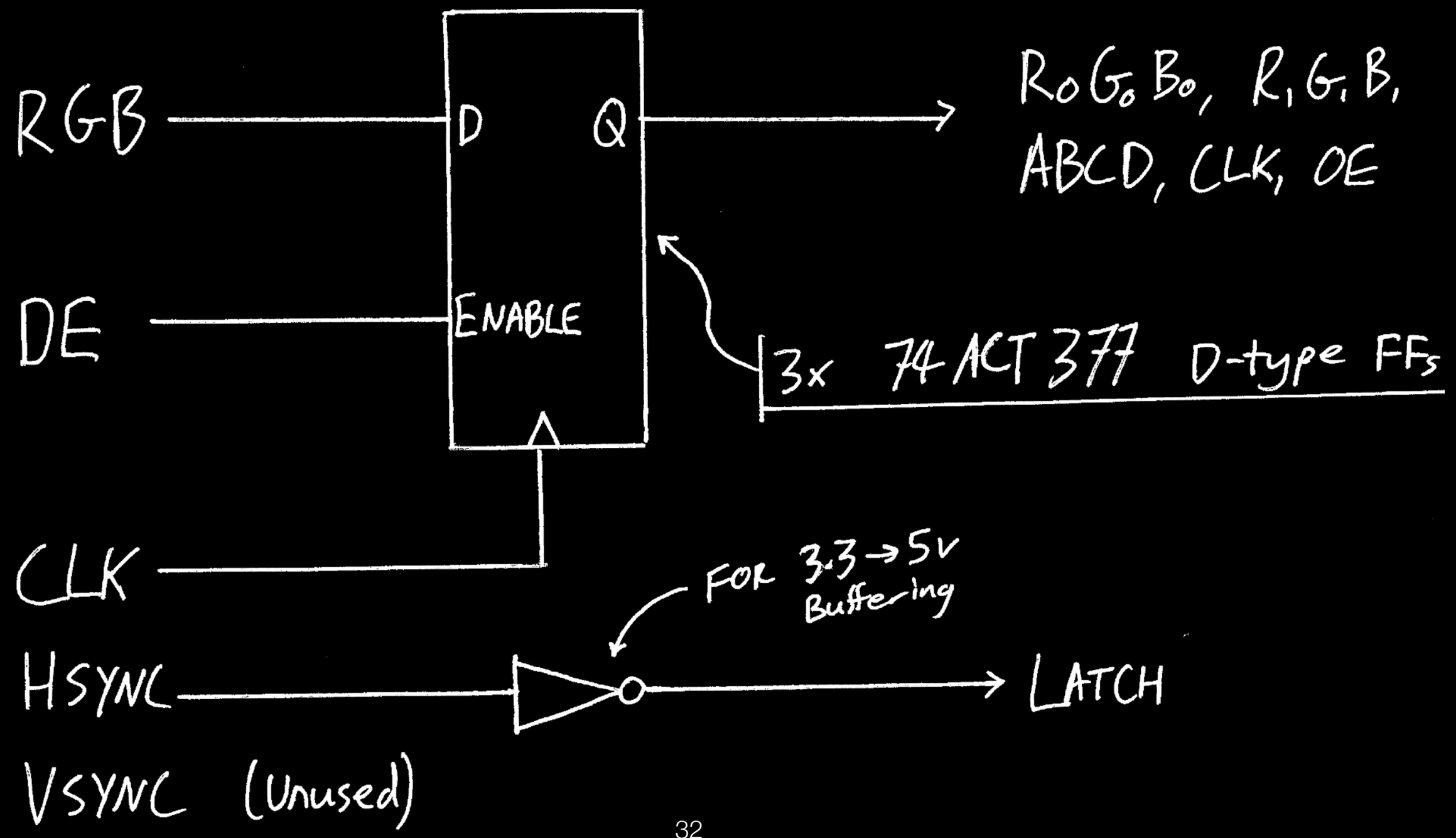


Signals not (all) held across Hsync (but some are...)



v2 circuit

RPi DPI (3.3v) HUB75 (5v)



Hub75Hat r0

GPIO

Matt Evans
290318

R,T,-

UART

Pi
Me

AUX

NI

HUB75 1:16

DISP1

DISP0

5VDC
(+) (-)

RN10 RN12 RN9 RN3

RN1 RN11 RN4 RN2

R2 L1 C1 C2 IC1 C8 L2 L3 C10 C9 C6 C5 C4 C3

Enabling DPI

- OK Matt, DPI sounds *amazing*, how do I use it?
- https://www.raspberrypi.org/documentation/hardware/raspberrypi/dpi/README.md
- Enabled through /boot/config.txt
- Then, just write the display framebuffer as usual

```
dtoverlay=dpi18
enable_dpi_lcd=1
display_default_lcd=1
framebuffer_depth=16
# HS/VS phase (0) polarity (0) control (7), nCLK/DE (2),
# RGB order(1), 565 mode 3 (3):
dpi_output_format=0x007213
# Custom timings
dpi_group=2
dpi_mode=87
# hdmi_timings=<h_active_pixels> <h_sync_polarity> <h_front_porch>
#  <h_sync_pulse> <h_back_porch>
#  <v_active_lines> <v_sync_polarity> <v_front_porch> <v_sync_pulse>
#  <v_back_porch> <v_sync_offset_a> <v_sync_offset_b>
#  <pixel_rep> <frame_rate> <interlaced> <pixel_freq> <aspect_ratio>
hdmi_timings=272 0 0 8 0  1103 0 1 1 100 0 0  0 170 0 60000000 1
```

BCM2835 DPI capabilities

Pixel clock	min 32MHz, 105MHz tested OK Maximum ~150MHz?
X pixels	min 8 max 1920
Y pixels	min 8 max 1280
Sync widths	min 1 pixel for HS min 1 line for VS
Misc	Didn’t check max sync widths or lowest frame rate (theoretically 14Hz)

Many other computers support similar LCD video output

- Common for SBCs to have parallel output from LCD controller!
  - Beagleboard, various cheap Allwinner/sunxi boards
- I like this technique because:
  - Often faster than GPIO
  - Realtime, zero CPU overhead
  - Much easier to get started/debug than using DMA controllers
  - Can do this from **userspace**, or even python
- Not as nice as a Beaglebone PRU ;-) (But $$$/complicated!)

Aside from VGA and LCDs and LEDs, what is it good for?

- Supply data to FPGA/CPLD — pattern/signal generator?
- Motors — 24 servos!
  - Steppers?
- Drive 20 SPI LCDs at once
- Or 24 strings of WS2812s — 24x1024 at >30fps!
  - Only 1.2kW & about €2000 🧐

Goodbye

- Some ideas for cool things to do with LEDs?
- Try using DPI/TFT controllers for unusual purposes?
- Look at peripherals on your boards — any opportunity for creative misuse?

Thank you!

More hax!

http://axio.ms

https://github.com/evansm7