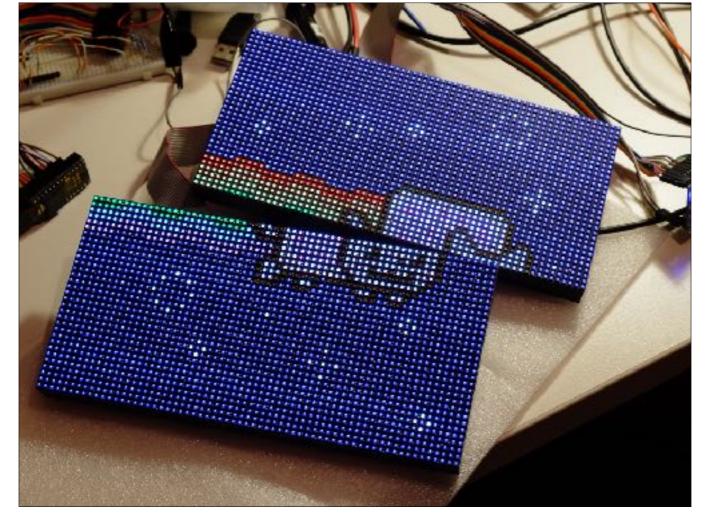


If you can read this, the projector setup is borked ;)



It all started with ... Shenzhen

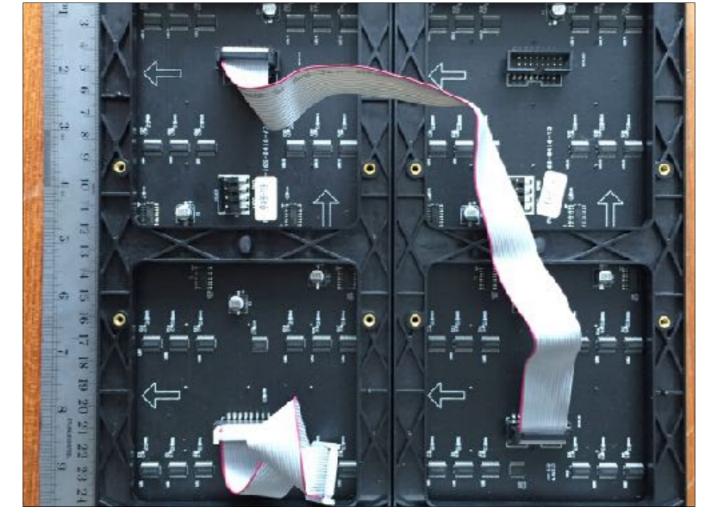
LED panels are cool — I find them abnormally fascinating :D **RGB** Lots of LEDs on a PCB, in modules with easy mounting/wiring Your driver circuit needs to scan video through these manually Manual BCM/PWM! High FPS crucial



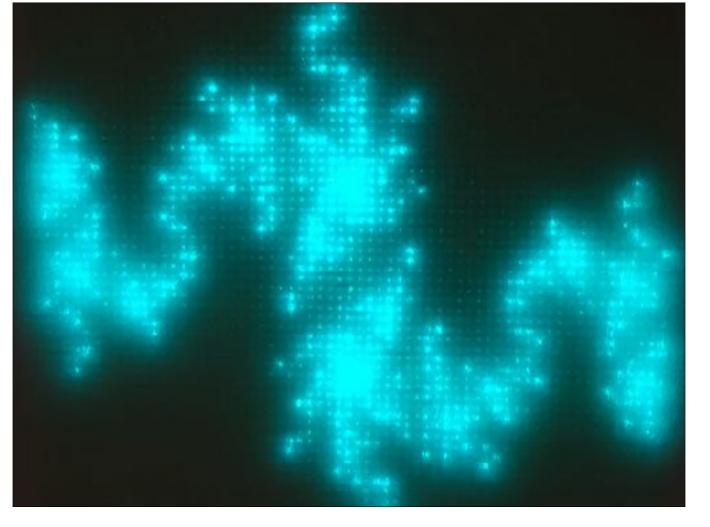
I got two 64x32, stuck together to make 64x64 — 4K display! My first test, purely **bitbanged** (haha) in software using an mbed — **EXAMPLE** Bitbanging is a fine way to try POC, test out an algorithm, etc.



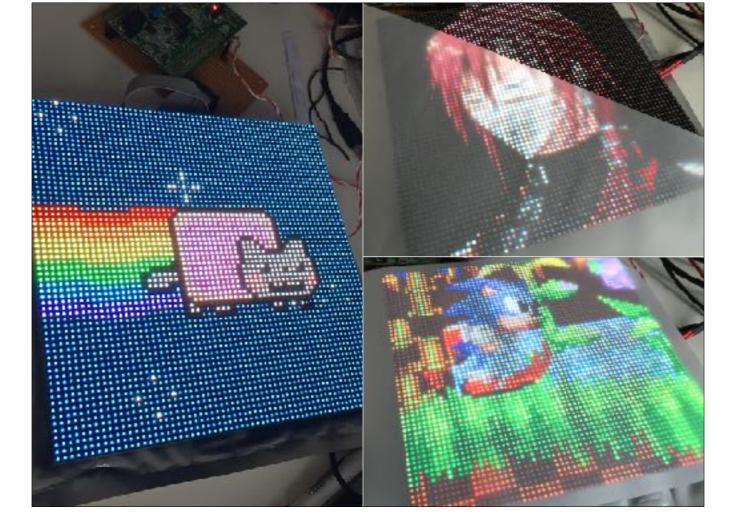
First proper go — driven from **STM32** Big shift register Clock data in for a whole row, then latch — example later Full-colour RGB — you have to do the PWM/brightness control by hand



Daisychain serial data out of one, into next



Got the panels running from my STM32, playing back **animations** Demo effects **plasma, blobs, fractals** 



Also MAME captures & GIFs.

**Timing** is key — **flicker** very noticeable! Will write about using **DMA controller**. Very **predictable timing**. Pleased with this ~170Hz 33BPP / 11BPC.

WAIT – but WHY? For **lulz** 

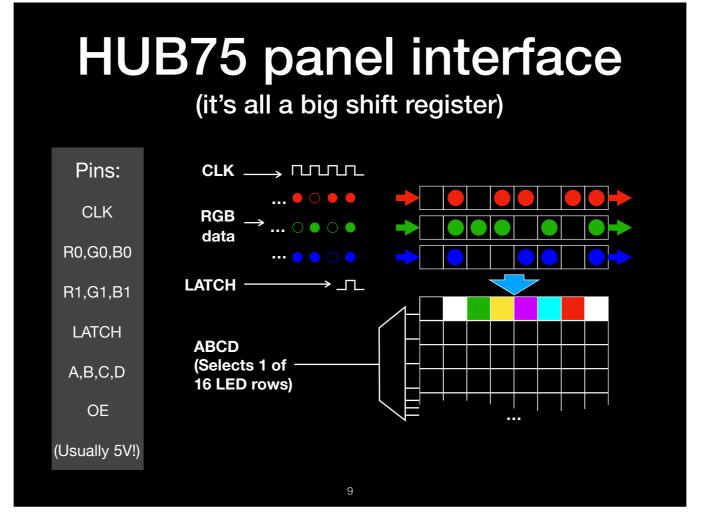
BUT... wanted **network**, stream **video from phone**. WHY? Haha for lulz

#### LED panels from a Raspberry Pi

- Great library for driving LED panels from Raspberry Pi:
  - <u>https://github.com/hzeller/rpi-rgb-led-matrix</u>
- 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"

Okay, so I wanted **networking** – **RPI zzz** but Linux is just too convenient to ignore. But **Pi Zero** – €5-10!
Existing libs. Adafruit.
Crux of my argument: Bitbanging – great for simple tasks. Rubbish for realtime.

Dedicate a 64-bit Cortex A53? Eww. PiZero only has one core. PROBLEM



What do I need? HUB75 is a common interface. 5V. Some variation:
R,G,B data bit + clock into 3 shift regs, for R/G/B along a row
+Latch = energise data onto a row
row selected by 4 bits A,B,C,D
Data needs to go in about 25-30MHz

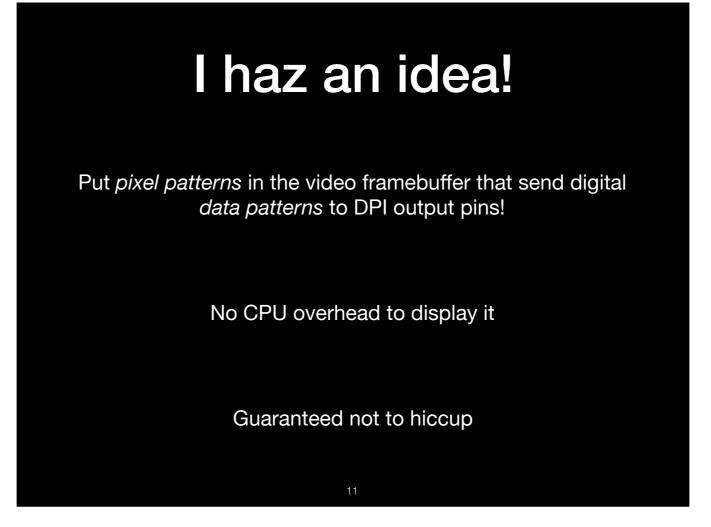
As fast as possible - higher refresh rate, higher colour depth

#### Display Parallel Interface (DPI)

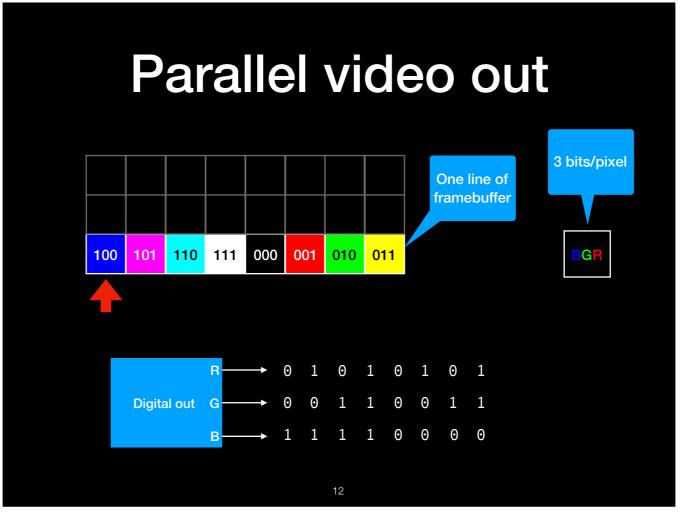
- Once upon a time, I was attaching an LCD to a RPi using DPI
- Parallel interface designed to drive TFT LCDs from BCM2835 alternative to HDMI
- 24-bit pixel output (+ pixel clock, + sync bits) high-speed digital output, 3.3V CMOS

SV3 Forver         1         0         2         3         V Power           BCM 2 (volne)         8         9         6         6         5         Power           BCM 3 (volne)         8         9         6         6         6         5         Power           BCM 3 (volne)         8         9         6         10         8         6         6         6         0         14         6         6         14         6         6         14         6         6         11         6         12         8         11         10         12         8         11         10         12         8         11         10         12         8         11         10         12         8         11         10         12         8         11         10         12         10         12         10         12         10         12         10         12         10         12         10         12         10         12         10         12         10         12         10         12         10         12         10         12         11         10         10         10         10         12         10	Raspberry Pinout	Pi (	D		
BCM 3 ()= SYNC         S         I         Ground           DCM 4 (base ()         7         6         8CM 14 (date: 2)           BCM 17 (Steel 3)         11         6         10         8CM 18 (date: 2)           BCM 17 (Steel 3)         11         6         12         8CM 18 (date: 2)           BCM 12 (base 3)         11         6         12         8CM 18 (date: 4)           BCM 22 (base 3)         12         6         16         8CM 28 (date: 4)           BCM 10 (base 4)         19         6         18         8CM 28 (date: 4)           BCM 10 (base 4)         19         6         18         8CM 28 (date: 4)           BCM 10 (base 4)         19         6         24         8CM 28 (date: 4)           BCM 10 (base 3)         21         6         26         8CM 28 (date: 4)           BCM 11 (base 7)         23         6         8CM 19 (date: 4)           BCM 11 (base 7)         23         6         8CM 1 (date: 4)           BCM 11 (base 7)         23         24         8CM 1 (date: 4)           BCM 11 (base 7)         23         24         8CM 1 (date: 4)           BCM 12 (base 7)         25         24         8CM 12 (date: 4)	3v3 Power	1	• •	2	5v Power
DCM 4 stars ()         7         6         BCM 14 (decer 2)           Eround         9         10         10         BCM 15 (decer 2)           BCM 17 (decer 3)         11         0         12         BCM 18 (decer 3)           BCM 27 (dec 7)         13         14         0         15         Genund           BCM 27 (dec 7)         13         14         0         15         Ground           BCM 27 (dec 7)         13         0         16         BCM 18 (decer 3)           BCM 27 (dec 7)         13         0         16         BCM 23 (decer 3)           BCM 22 (dec 7)         13         0         16         BCM 23 (decer 3)           BCM 10 (decer 3)         17         0         18         BCM 24 (decer 3)           BCM 10 (decer 3)         19         0         28         BCM 25 (decer 3)           BCM 17 (decer 3)         21         0         28         BCM 10 (decer 4)           BCM 17 (decer 3)         27         2         28         BCM 1 (decer 4)           BCM 25 (dec 3)         21         20         28         BCM 1 (decer 4)           BCM 5 (dec 3)         21         20         28         BCM 1 (deceer 4)           BCM 5 (	BCM 2 (vanic)	- 8 (	••	4	5v Power
Brown S         Image: S <thimage: s<="" th=""> <thimage: s<="" th=""> <t< td=""><td>BCM 3 (Horne)</td><td>5</td><td>- 0</td><td>ź.</td><td>Ground</td></t<></thimage:></thimage:>	BCM 3 (Horne)	5	- 0	ź.	Ground
BCM 17 (See 5)         11         10         12         BCM 18 (Geen 5)           BCM 27 (see 7)         12         14         Ground         BCM 23 (see 5)           BCM 22 (see 2)         15         16         BCM 23 (see 5)         BCM 23 (see 5)           BCM 10 (see 6)         17         18         BCM 24 (see 5)         BCM 23 (see 5)           BCM 10 (see 6)         19         18         BCM 24 (see 5)         BCM 25 (see 5)           BCM 10 (see 6)         21         20         22         BCM 25 (see 5)           BCM 11 (see 7)         23         24         BCM 25 (see 5)         BCM 1 (see 5)           BCM 0 (state 7)         27         28         BCM 1 (see 5)         BCM 1 (see 5)           BCM 0 (state 3)         27         28         BCM 1 (see 5)         BCM 1 (see 5)           BCM 5 (state 3)         21         25         28         BCM 1 (see 5)           BCM 5 (state 3)         21         25         28         BCM 1 (see 5)           BCM 13 (state 1)         27         28         BCM 12 (state 0)           BCM 13 (state 1)         26         26         26         BCM 16 (state 1)           BCM 19 (state 2)         26         26         26         BCM 16 (	DCM 4 plus ()	7.	00	8	BCM 14 (Green Z)
BCM 22 (mail ?)         12         14         Control           BCM 22 (mail ?)         15         16         BCM 23 (mail 3)           BCM 22 (mail ?)         15         16         BCM 23 (mail 3)           BCM 10 (mail ?)         17         18         BCM 24 (mail 9)           BCM 10 (mail ?)         17         18         BCM 24 (mail 9)           BCM 10 (mail ?)         17         18         BCM 25 (mail 9)           BCM 10 (mail ?)         21         10         22         BCM 25 (mail 9)           BCM 11 (mail ?)         23         12         24         BCM 3 (mail 9)           BCM 11 (mail ?)         23         12         24         BCM 3 (mail 9)           BCM 11 (mail ?)         23         12         24         BCM 1 (mail 9)           BCM 15 (mail 9)         27         24         BCM 1 (mail 9)           BCM 5 (mail 9)         28         27         24         BCM 1 (mail 9)           BCM 5 (mail 9)         28         27         24         BCM 1 (mail 9)           BCM 5 (mail 9)         28         27         24         BCM 1 (mail 9)           BCM 13 (mail 9)         28         27         24         BCM 1 (mail 9)           BCM 13		9	• •	10	BCM 15 (Green 2)
DCM 22 (364 2)         15         16         BCM 23 (364 3)           3x3 Power         17         18         BCM 24 (set 4)           BCM 10 (stars)         17         18         BCM 24 (set 4)           BCM 10 (stars)         17         18         BCM 25 (rest 8)           BCM 11 (stars)         21         20         22         BCM 25 (rest 8)           BCM 11 (stars)         23         24         8CM 35 (stars)         4           BCM 11 (stars)         23         24         8CM 13 (stars)         4           BCM 11 (stars)         27         28         8CM 1 (stars)         8           BCM 5 (stars)         27         28         8CM 1 (stars)         8           BCM 5 (stars)         27         28         8CM 12 (stars)         8           BCM 5 (stars)         27         28         8CM 12 (stars)         8           BCM 5 (stars)         28         27         28         8         8           BCM 5 (stars)         28         27         28         8         8         9         9         9         8         9         9         9         9         9         9         9         9         9         9         9	BCM 17 (Green 2)	11	00	12	BCM18 (Green 4)
BCM 10         pice         12         18         BCM 224 (sad 4)           BCM 10         pice a)         19         28         Ground           BCM 10         pice a)         21         22         BCM 25 (red 5)           BCM 11         pice b)         23         24         BCM 25 (red 5)           BCM 11         pice b)         23         24         BCM 25 (red 5)           BCM 11         pice b)         23         24         BCM 25 (red 5)           BCM 11         pice b)         23         24         BCM 3 (red 5)           BCM 10         pice)         25         26         BCM 1 (relax)           BCM 5 pice 1)         27         24         BCM 12 (green c)           BCM 5 pice 2)         31         25         32         BCM 12 (green c)           BCM 13 (stress 1)         33         34         35         34           BCM 13 (stress 1)         33         34         35         34           BCM 13 (stress 1)         35         34         35         34           BCM 19 (stress 7)         35         34         35         36           BCM 26 (stress 7)         35         34         35         36	BCM 22 (Red 7)	12	<u>.</u>	14	Ground
BCM 10 (due k)         19         21         22         Ground           BCM 9 (due k)         21         22         BCM 25 (red 8)           BCM 11 (due 7)         23         24         8CM 25 (red 8)           BCM 11 (due 7)         23         24         8CM 25 (red 8)           BCM 11 (due 7)         23         24         8CM 25 (red 8)           BCM 11 (due 7)         23         24         8CM 18 (due 7)           BCM 1 (due 7)         25         26         8CM 1 (due 7)           BCM 5 (due 7)         31         32         33         8CM 12 (due 7)           BCM 5 (due 7)         33         34         35         36           BCM 13 (due 7)         35         34         37         36           BCM 19 (due 7)         35         34         36         37           BCM 13 (due 7)         35         34         36         37           BCM 19 (due 7)         35         34         36         37           BCM 26 (due 7)         35         36         37         36	BCM 22 (Red 2)	15	00	16	BCM 23 (8:43)
BCM 9 Star 11         21         21         22         BCM 25 (red 8)           BCM 11 Day 21         23         24         BCM 8 (red 8)           BCM 11 Day 21         23         24         BCM 8 (red 8)           BCM 11 Day 21         24         8 CM 8 (red 8)           BCM 0 (red 8)         26         8 CM 7 (red 8)           BCM 1 (refs)         27         28         8 CM 1 (refs)           BCM 5 (refs)         27         28         8 CM 1 (refs)           BCM 5 (refs)         27         28         8 CM 12 (refs)           BCM 6 (refs)         31         26         32         8 CM 12 (refs)           BCM 13 (refs)         33         34         35         34         8 CM 10 (refs)           BCM 19 (refs)         35         34         8 CM 16 (refs)         8 CM 16 (refs)         8 CM 16 (refs)           BCM 26 (refs)         37         35         34         8 CM 12 (refs)         8 CM 16 (refs)	3v3 Power	17	00	18	BCM 24 (Red 4)
BCM 11 Day 7         23         24         BCM 8 Day 4           Ground         25         26         BCM 7 (day 3)           BCM D (day 3)         27         28         8CM 1 (sex)           BCM 5 (day 3)         27         28         8CM 1 (sex)           BCM 5 (day 3)         27         28         8CM 1 (sex)           BCM 6 (day 3)         31         29         31         60         32         8CM 12 (days 2)           BCM 13 (days 3)         31         29         32         40         8CM 12 (days 2)         8CM 12 (days 2)<	BCM 10 (star e)	19	• •	30	Ground
Ground         Image: Control of the state of the s	DCM 9 also at	21	00	22	BCM 25 (hed b)
BCM D (rule)         27         6         38         BCM 1 (relies)           BCM 5 (star 1)         29         6         16         Ground           BCM 6 (star 2)         31         6         32         BCM 12 (case 1)           BCM 13 (stare 1)         32         6         33         Ground           BCM 13 (stare 1)         32         6         33         Ground           BCM 19 (stare 7)         36         6         32         BCM 16 (stare 1)           BCM 26 (stare 2)         37         0         38         BCM 20 (stare 3)	BCM 11 (the 7)	23	00	24	BCM 8 (thus 4
BCM 5 sterit         29         11         Ground           BCM 6 pterit;         81         22         8CM 12 (Greent;)           BCM 13 (Steent)         33         33         61         34           BCM 13 (Steent)         33         34         Ground         BCM 10 (Greent;)           BCM 19 (Steent)         35         36         36         BCM 10 (Greent;)           BCM 20 (Steent)         36         37         36         36           BCM 20 (Steent)         37         36         36         8CM 20 (Steent)	Ground	25	• 0	26	BCM 7 (the 3)
BCM 6 plus 3;         31         0         32         BCM 12 (Green 0)           BCM 13 (Streen 1)         33         0         34         Ground           BCM 19 (Streen 7)         35         0         34         Ground           BCM 19 (Streen 7)         35         0         34         BCM 16 (Streen 6)           BCM 20 (Streen 7)         35         0         34         BCM 16 (Streen 6)           BCM 20 (Streen 7)         35         0         34         BCM 16 (Streen 6)	BCM 0 years	27	00	38	BCM 1 (can)
BCM 13 (See 1)         33         34         Ground           BCM 19 (See 7)         35         36         36         BCM 16 (See 1)           BCM 26 (See 2)         37         36         36         BCM 10 (See 1)	BCM 5 the ti	29	• •	30	Ground
BCM 19 (2rease 7) 25 0 26 BCM 16 (2rease 1) BCM 26 (2rease 2) 27 0 28 BCM 20 (2read 2)	BCM 6 plus a)	81	00	52	BCM12 (Geen C)
BCM 26 static 27 00 28 BCM 20 rad st	BCM 13 (See 1)	33	۰ و	34	Ground
	BCM 19 cover 73	35	00	36	BCM 16 (Green to
Annual and the Annual and the Annual and the Annual A	BCM 26 and o	37	00	38	BCM 20 (red to
Cradna 20 40 BGM 21 (8d 1)	Ground	39	• 0	40	BCM 21 (Ref 1)

**Interlude**: I was using DPI for another project (to drive an LCD as intended) Very **high speed** pixel output — up to >100MHz **Digital**: takes a pixel, sends it out



Patterns generate signals – **signals like HUB75**!



Taking an example, say we have 3 bits/pixel R,G,B framebuffer

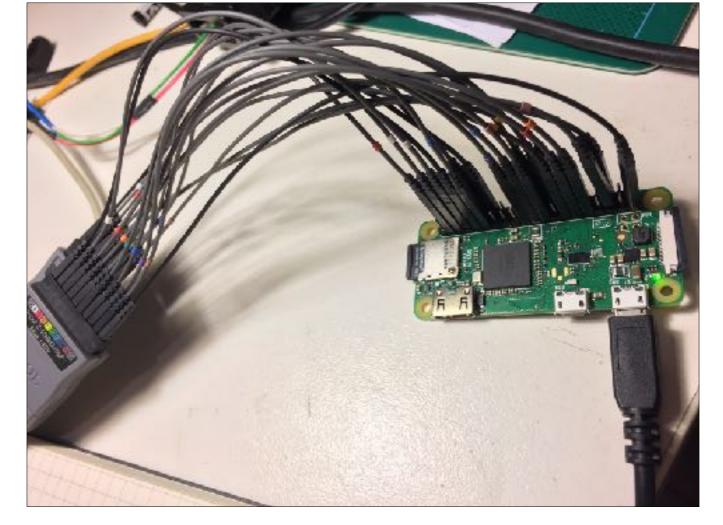
Colour bits output with regular timing

Pixel bits — as clock, or data, anything

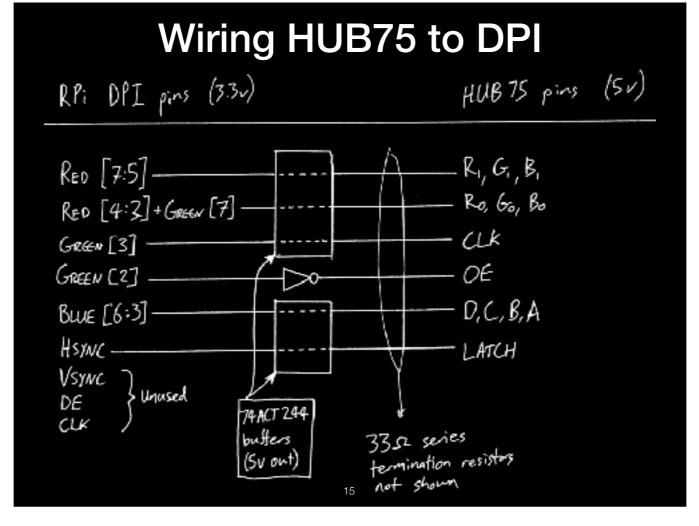
**HSYNC** at end of line can **latch** a block — a bit like the row **latch** in HUB75

## 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



LA experiments. DPI pretty flexible - program resolution, sync width etc., variable bit depth Just dump stuff into **/dev/fb0 framebuffer device Test program** to try different FB resolutions/**POC** 



Budget runs out for fancy diagrams. Prototype:

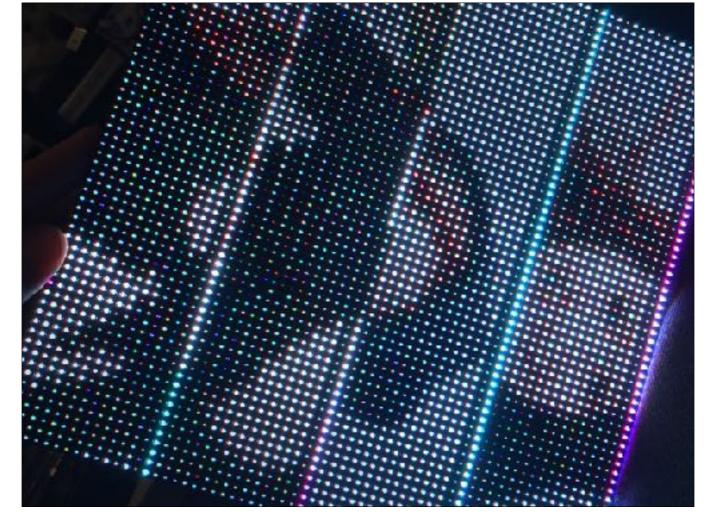
- 4 bits -> ABCD row select 1/16, 6 bits for RGB0/RGB1, 1 bit for CLK
- 1 bit for OE modulates brightness of current row
- HSYNC -> LATCH

Termination very important - wires long, fast signals

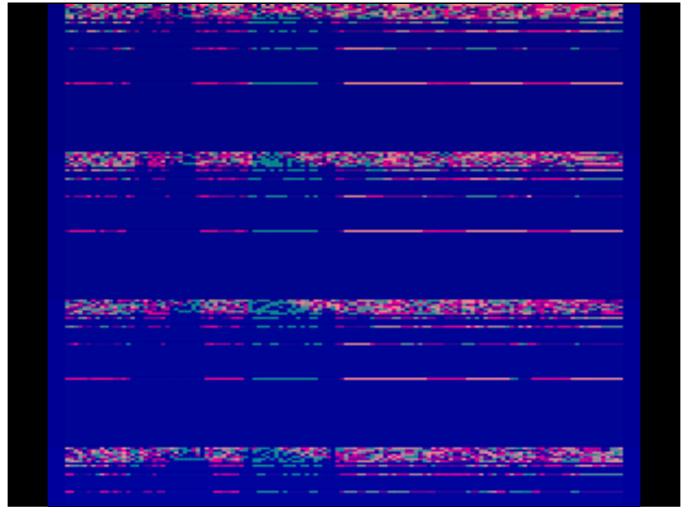
So I solder that up.



Little bit of protoboard — 74'245 to drive 5V Extremely cheap! Interface costs less than 1 beer Can see the acrylic case I started to build for it... **Glueing acrylic and trying to get it NEAT, OMG painful** 



ANNND... OK, more debug needed. But concept good:
Clocks one row worth of data every video line; HSYNC at end of line then latches that to drive LEDs.
For a given row, spends 68 lines driving different intensity levels of same pixels – BCM
Then, select new row – same again
Then, after 16 rows, frame done.



What the Pi's framebuffer actually looks like (e.g. if you connected HDMI)

First four of 16 rows

128 clocks horizontally, for each driving RGB\*2 into LED rows

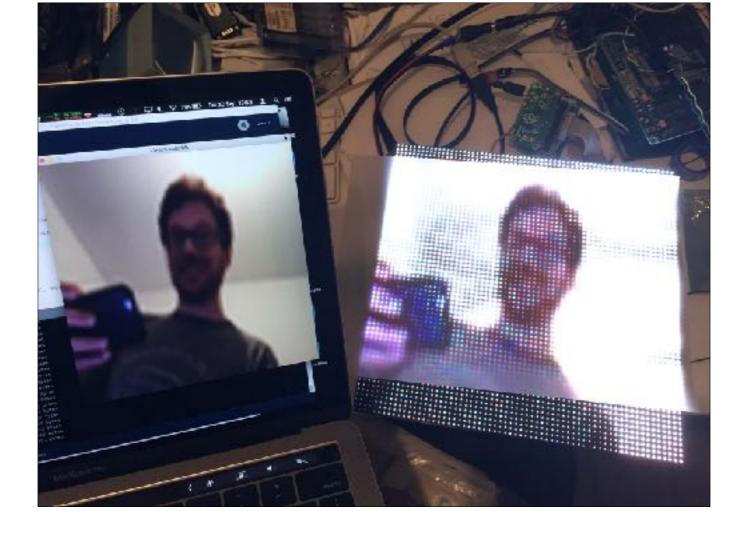
Look at the **gap** — this is the timing for the intensity levels — higher bits in colour/intensity are left on for a longer time BLUE background - dark to light (**ABCD row sel**)



Works really well — **guaranteed no flickering** Simple **Linux library** that takes 64x64 32BPP RGB and does **massive bit-shifting**, writing to /dev/fb0 to send data out line by line. App simply sees a **flat RGB** frame-buffer Plasma, fractal animations



The **realtime Julia set fractal** was pretty smooth on the STM32, but super smooth on 1GHz CPU I haven't hacked **DOOM** to use it yet :) 177Hz, 11BPC/33BPP — **entirely flicker free** no matter how heavily loaded network/CPU is! **0% CPU overhead** 



I got my **video streaming** wish...

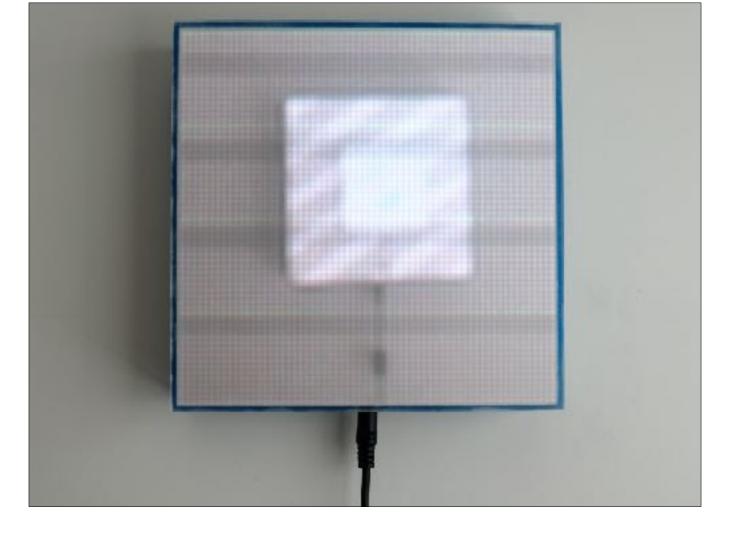
2:47 🏾 🕸 🕏 70% 🔳 -	
testpi.house	
Sending video	
+	
0	
	testpi house

...wrote a little **app for my phone** to send image snapshots or **stream video** 



Pointless fun thing for the wall

Also resolution so low that software MPEG2 player is only a few % CPU ;-)

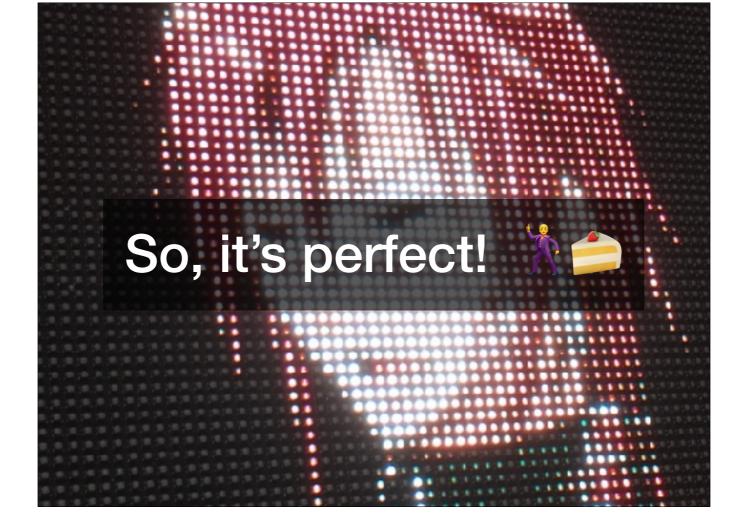


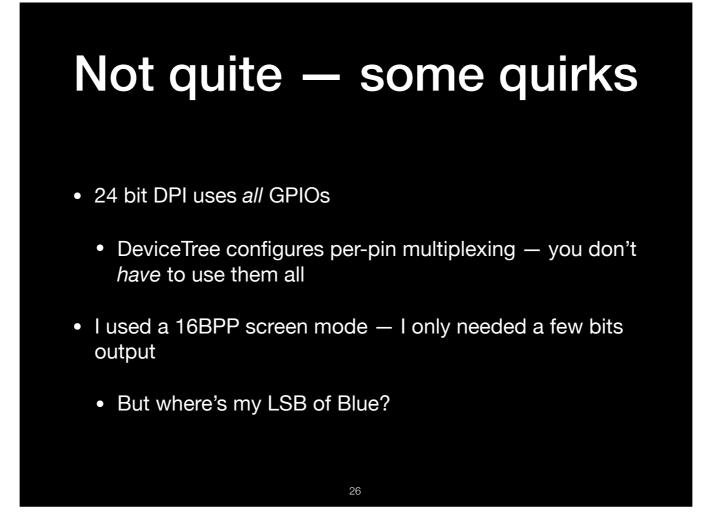
Recursive...

On the software side, I mentioned 0% CPU to hold the image flicker free

To change the image there's the massive bit-shifting exercise - low but not free, 1.5ms: at 60changes/sec it's 9% CPU

-> Not optimised — expect can improve

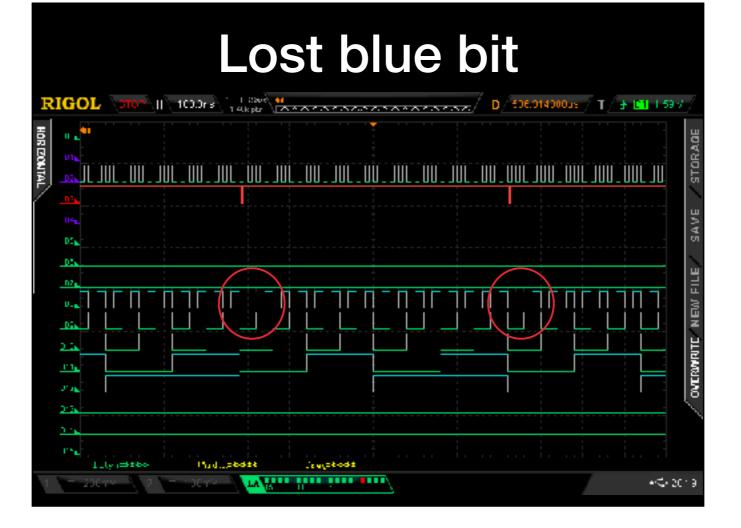




Use custom DT to use UART - max 22 bits

In 16BPP noticed odd/intermittent disappearance of b0 in SOME situations

Can re-create by drawing a gradient ramp in blue (R=G=0)



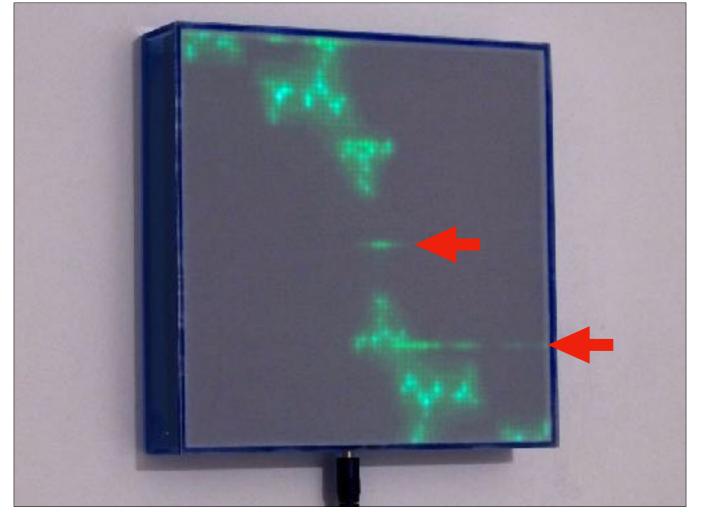
LA plot showing **intensity ramp in blue** Binary counting up 5 bits of blue See the gaps? B=0b00000 and 0b00001 are both output as zero

### Lost blue bit

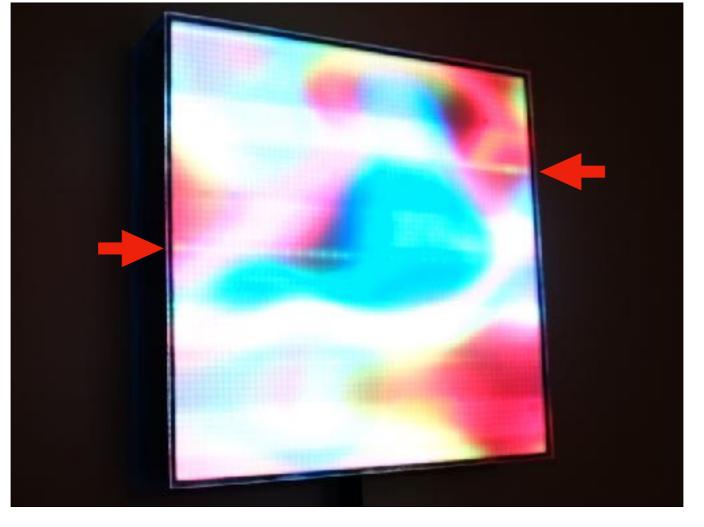
- My theory: display controller doing post-framebuffer dithering/colour correction in 16BPP modes
- This does not occur in 32BPP modes I recommend just using 32BPP!

Just use 32BPP mode.

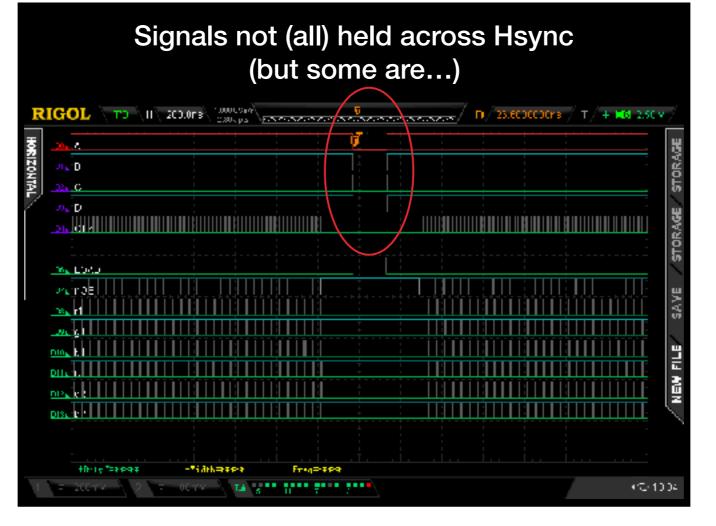
OK anything else?



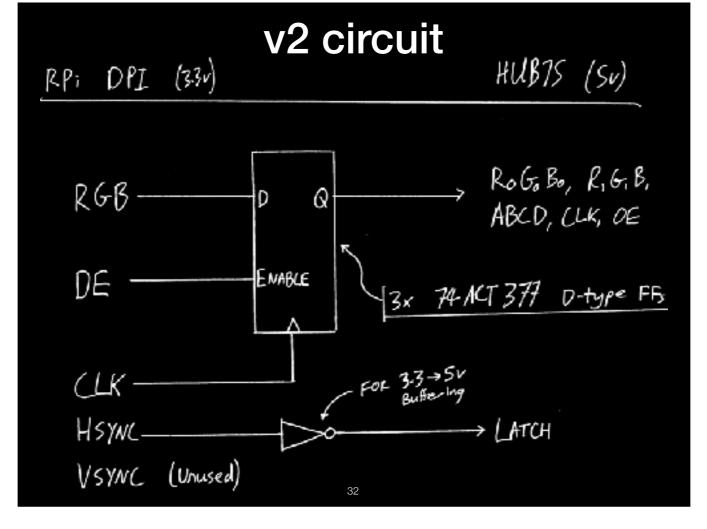
Another quirk: These lines aren't supposed to have that shadowing of other stuff lower down the image



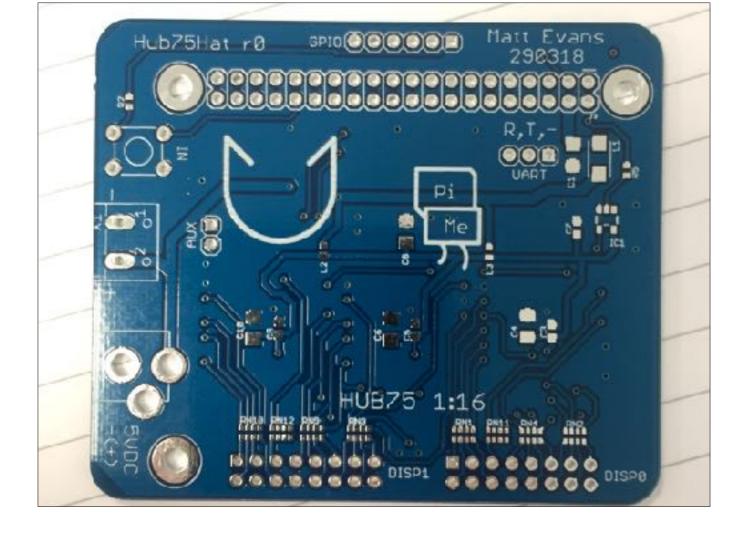
These lines correspond to row 1 out of 16 for each segment of the panel (4)



LA plot: You can see the **ABCD/row** lines all going to **0 across HSYNC** (LOAD). (ALSO **B0** loss) That means that **first row is getting selected across HSYNC** — displaying whatever was latched Remember OE? Improved this by **de-asserting OE** before HSYNC **Wait time** isn't long enough — more padding bad



Decided to take a **different approach** Added **flip-flips** — latch output pixel Hold it across HSYNC (so current row held) — **predictable** Buffer to 5V. Still **cheap** interface — <1 beer



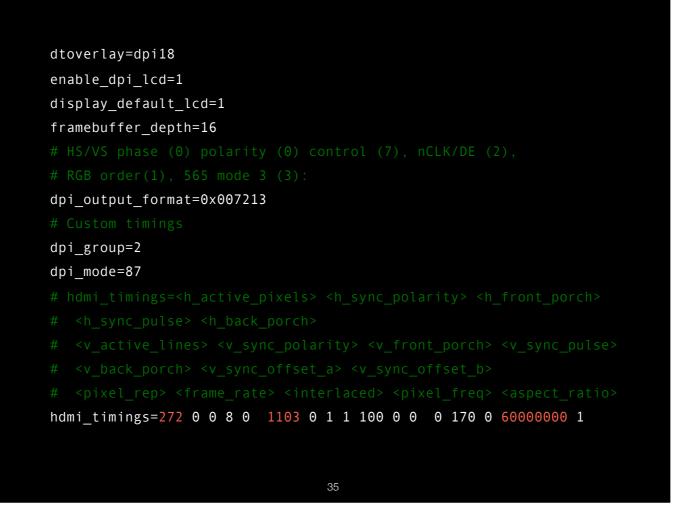
WORKS GREAT! No shadowing.

## **Enabling DPI**

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

Using **SDL** or similar makes it easier to gain **control of the framebuffer** 

- Disables the **cursor** and **debug messages** trampling on your image



Example working config

If the Pi doesn't like your configuration, it will silently:

- choose one it prefers
- fail to boot
- boot but keep **DPI disabled**

#### **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)	
	36	

Remember don't NEED PCLK — can clock external stuff from a pixel colour bit My scope isn't high enough BW to see how good o/p @140MHz is, but it enabled FB dimensions; max/min/alignment — **fit your data** around this SYNC— :( **Can't get completely unbroken** stream of data out

## 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!)

37

I used a Pi here but this **technique** applies to many other machines.

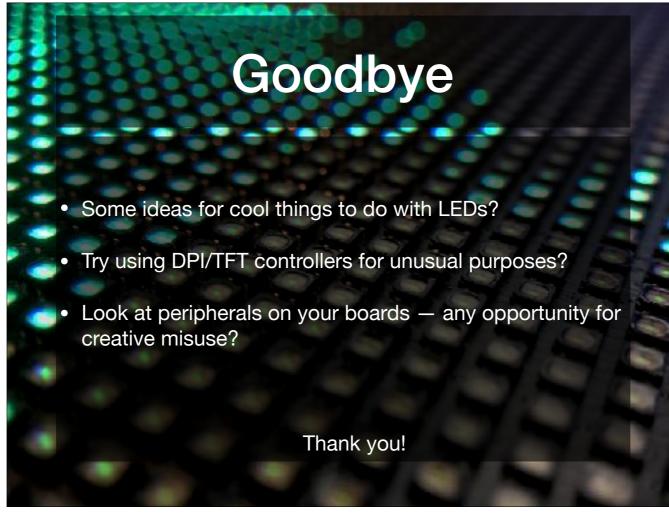
# Aside from VGA and LCDs and LEDs, what is it good for?

- 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

Wanted to do sig gen - but hard w/o unbroken stream of data (VS/HS gaps) - FPGA retime

**Stepper motors** might work if they can deal with the VS/HS gaps?

Have been playing with some very cheap 2" TFT modules (0.5 beer) — SPI stream, could drive 20 of these in parallel from one Pi. 20 tiny displays, cool! ... all from userspace, all 0% CPU overhead



OK, tour has come to an end. Cheers, seeya. :D

