Youtube Streamer

The Dream

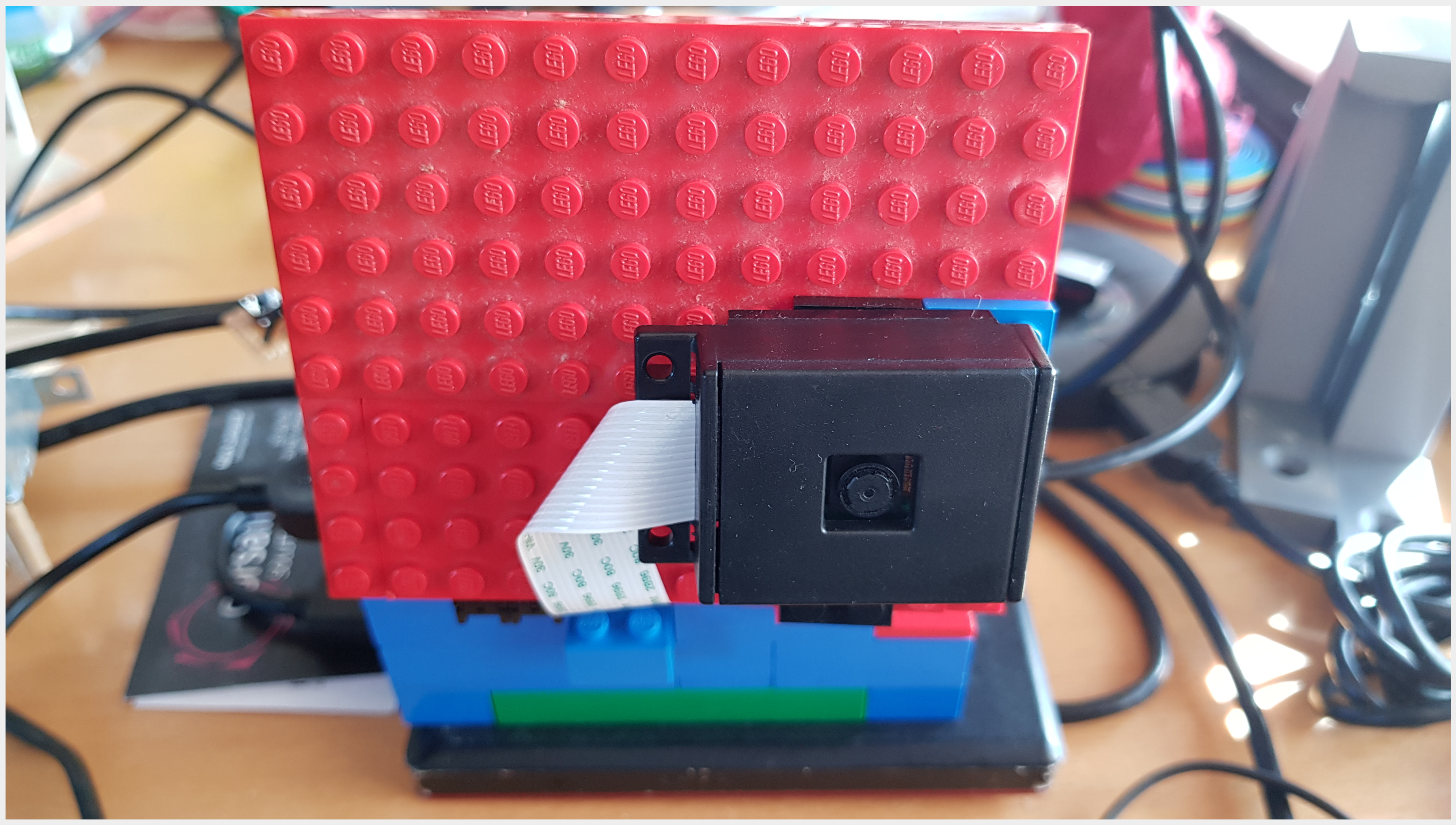
- Hit a button to stream talks to the greater community
- Small
- Quick
- No screen required

Setup

- Raspberry PI 3b+
- Simplecom UM350 microphone
- Raspbian Buster
- Used balenaEtcher to write the image
- install ffmpeg & screen

Case

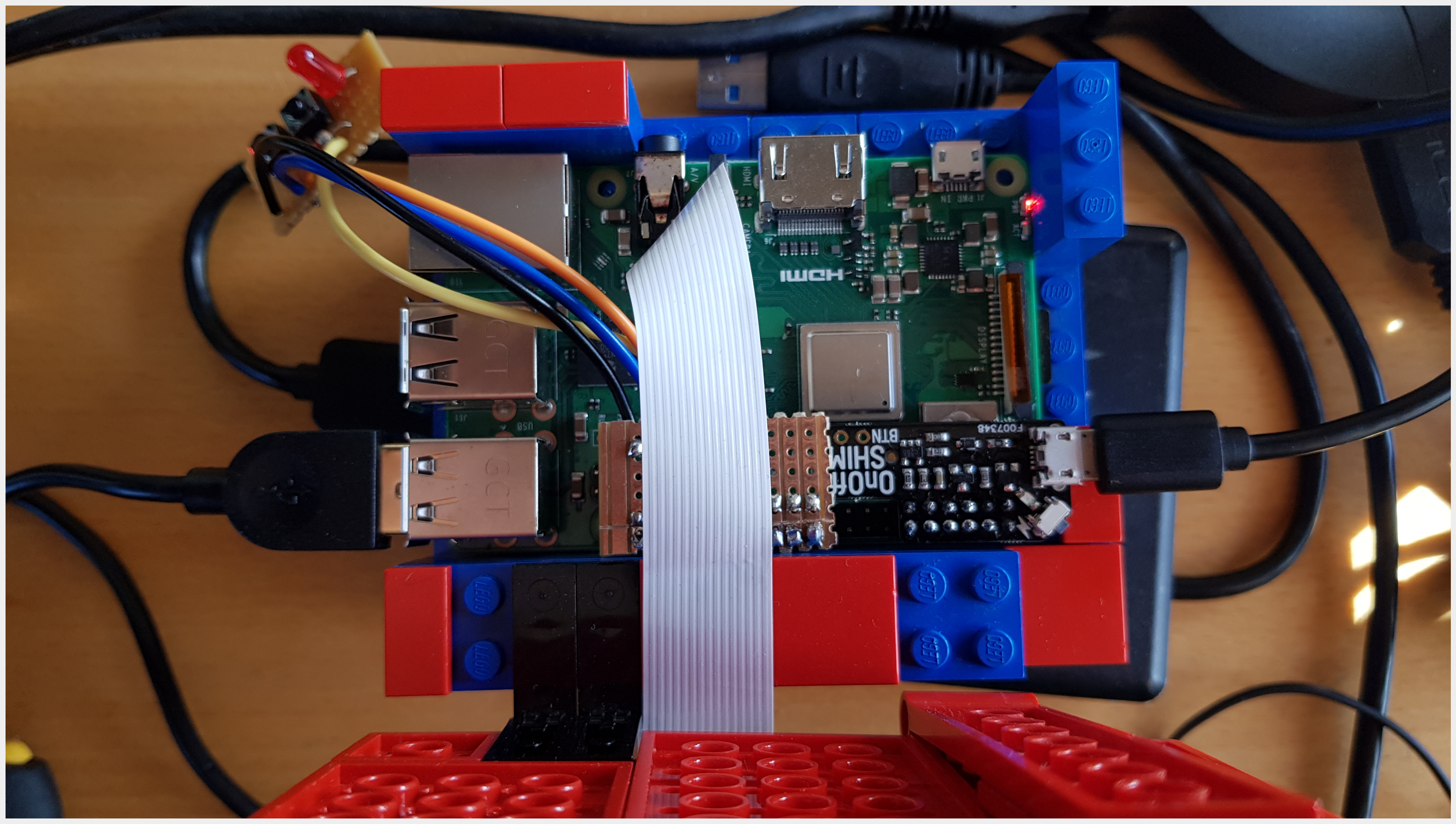
- Built out of my 1980's lego!


LEGO

Graceful shutdown

- Run without a screen
- Button turns on
- Hold for a few seconds to graceful shutdown

On/Off Shim



Setup

- Really easy to setup

  https://shop.pimoroni.com/products/onoff-shim

  Install software

  ```
  curl https://get.pimoroni.com/onoffshim | bash
  ```

  Now I can feed power to the OnOff Shim & hit the little button to boot.

Get IP address

Objective

- Flash LED

3 quick flashes	start
0.5 second flashes	number readout
long flash	zero
6 quick flashes	full stop

Script

The interesting bits...

Get IP address

```
ip = Popen("ip -4 addr show wlan0 | grep inet | awk '{print $2}' | cut -d/ -f1", shell=True, stdout=PIPE).communicate()[0]
```

Setup LED & Button

```
led = 35
button=29
GPIO.setup(led, GPIO.OUT) ## Setup GPIO Pin 7 to OUT
GPIO.setup(button, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Button
```

Flash digits

```
for x in list(ip):
    time.sleep(2)
    if x.isdigit():
        if (int(x) == 0):
            GPIO.output(led,True)# long LED light indicating zerc
            time.sleep(3)
            GPIO.output(led,False)
        elif (int(x) != 0):
            for y in range(int(x)):
                GPIO.output(led,True)
                time.sleep(.5)
                GPIO.output(led,False)
                time.sleep(.5)
```

Six rapid blinks

```
elif (x == '.'):
    for x in range(6): #six rapid blinks indicate a .

        GPIO.output(led,True)
        time.sleep(.1)
        GPIO.output(led,False)
        time.sleep(.1)

    time.sleep(2)
```

Full Script

ipblink.py

WIFI Setup

- Use priority
- Mobile tether fallback

Configuration

/etc/wpa_supplicant/wpa_supplicant.conf

```
network={
  ssid="Mick"
  psk=<password>
  priority=2
  key_mgmt=WPA-PSK
}
network={
  ssid="GNH  WiFi"
  psk=<password>
  key_mgmt=WPA-PSK
}
```

NOTE: The higher the priority the

Microphone

Test with arecord

Test audio only recording

```
arecord -D hw:1,0 -c 1 -f S16_LE out.wav
```

Plugin headphones to the PI and turn up the sound

```
alsamixer
```

Test playback = works

```
sudo apt install mplayer
mplayer out.wav
```

Test with ffmpeg

Test by recording using ffmpeg (broken)

```
ffmpeg -f alsa -i hw:1 -t 30 out.wav
```

Testing using plughw (works but crap sounding)

```
ffmpeg -f alsa -ar 8000 -i plughw:1 -t 30 out.wav
```

Testing using plughw:1

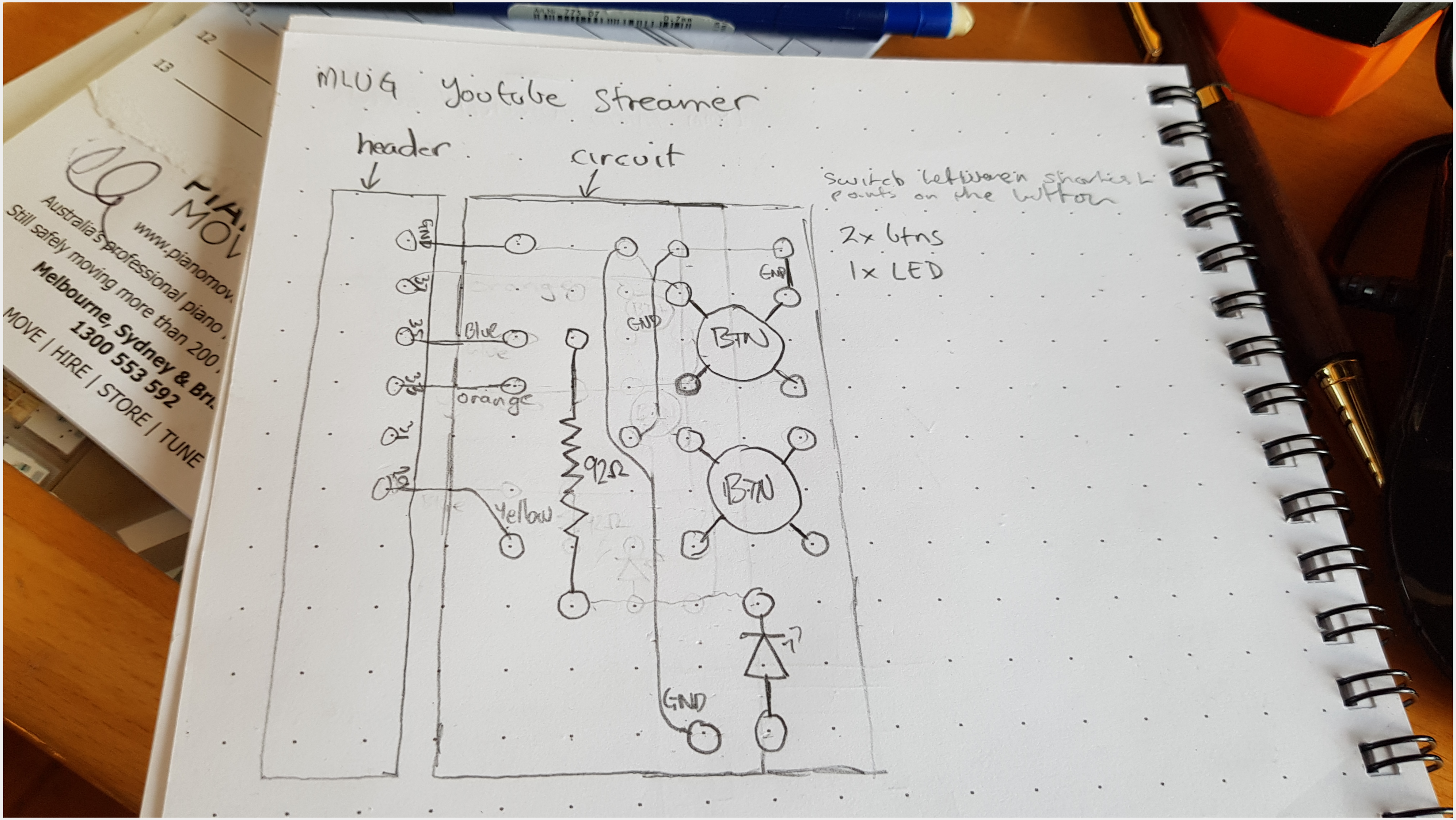
```
ffmpeg -f alsa -i plughw:1 -t 30 out.wav
```

NOTE: By using **plughw** ffmpeg will automatically detect all the stuff and work, yay!

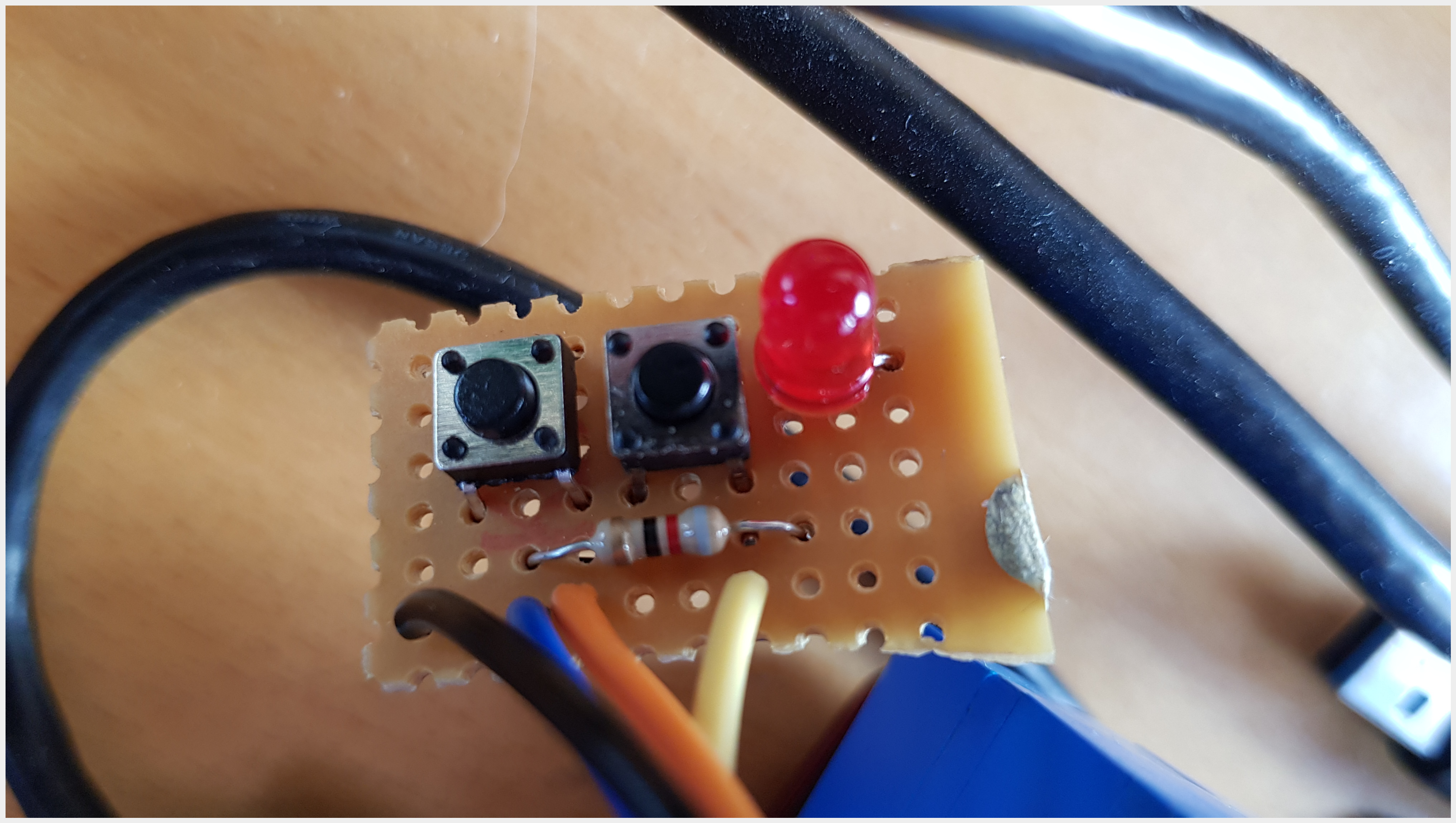
Circuit

- One LED
- IP button
- Stream button (start/finish)

Circuit design



Circuit finished



ffmpeg

Stream without microphone

```
raspivid -o - -t 0 -vf -hf -fps 30 -b 6000000 | ffmpeg
-re -ar 44100 -ac 2 -acodec pcm_s16le -f s16le -ac 2
-i /dev/zero -f h264 -i - -vcodec copy -acodec aac
-ab 128k -g 50 -strict experimental
-f flv rtmp://a.rtmp.youtube.com/live2/<your key>
```

-acodec	fake audio for Youtube (remove if using a Mic)
-f	File format (FLV is best with youtube)

Stream with microphone

```
raspivid -o - -t 0 -fps 25 -b 750000 | ffmpeg
-use_wallclock_as_timestamps 1 -f alsa -i plughw:1
-f h264 -i - -vcodec copy -acodec aac -ab 128k
-g 50 -strict experimental
-f flv rtmp://a.rtmp.youtube.com/live2/$KEY
```

raspivid switches

```
raspivid -o - -t 0 -fps 25 -b 750000
```

-o -	Output to the PIPE
-t 0	Record time 0=forever
-fps 25	25 Frame per second
-b 750000	bitrate 750Kbps

Audio ffmpeg switches

-use$_{\text{wallclockastimestamps}}$ 1	Sync audio with video using timestamps
-f alsa	Format for microphone
-i plughw:1	Microphone hardware port
-acodec aac -ab 128k	codec to use for the audio & bitrate
-strict experimental	Add this option to use aac in ffmpeg

Video ffmpeg switches

-f h264 -i -	Take h264 from raspivid input
-vcodec copy	Copy the video codec
-g 50	Fix keyframes to 50

Stream ffmpeg switches

-f flv	Format FLV for youtube
rtmp://a.rtmp.youtube.com/live2/$KEY	Send stream to Youtube

Quality settings

Bytes	Mbps upload	Works	Quality
4000000	4.3	y	good
2000000	2.1	y	good
1000000	1.2	y	good
750000	1.0	y	medium
500000	0.6	y	poor

Scripting

- Store keys in /etc/environment
- Use a bash script for the stream command
- Use python for button & LED control

Preview mode

- For setting up the angle before stream

  ~/bin/stream$_{vlc}$

  ```
  raspivid -o - -t 0 -n -w 1440 -h 1080 -fps 24 | cvlc -vvv str
  ```

  Then in VLC on the computer only type

  ```
  rtsp://10.1.1.207:8554/
  ```

Add to startup

Add to startup in /etc/rc.local just before the exit

```
/home/pi/bin/ip_blink.py &
/home/pi/bin/button.py &
```

What's next

- Picture in Picture for slides
- 3d printed case
- Auto posting of youtube link on IRC & email once **live**

References

- Making a LED blink using the Raspberry Pi and Python | Raspberry Pi HQ
- https://www.youtube.com/watch?v=BWYy3qZ315U
- Setting WiFi up via the command line - Raspberry Pi Documentation
- Capture/ALSA – FFmpeg

Questions

Email	map7777@gmail.com
Twitter	@map7
Github	github: map7