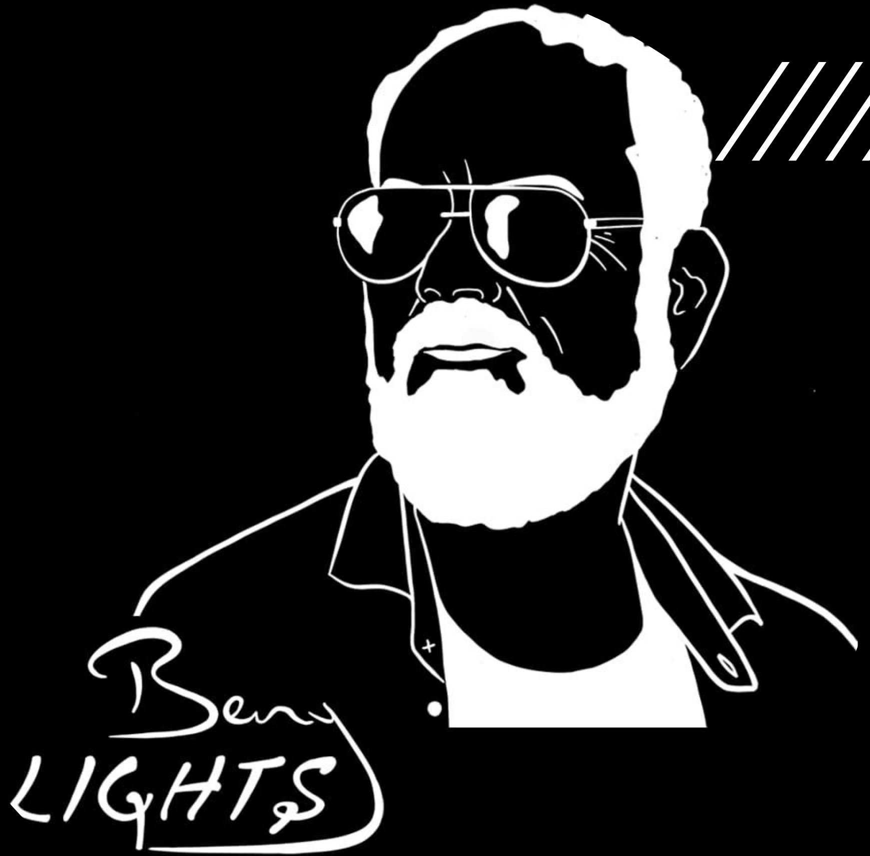
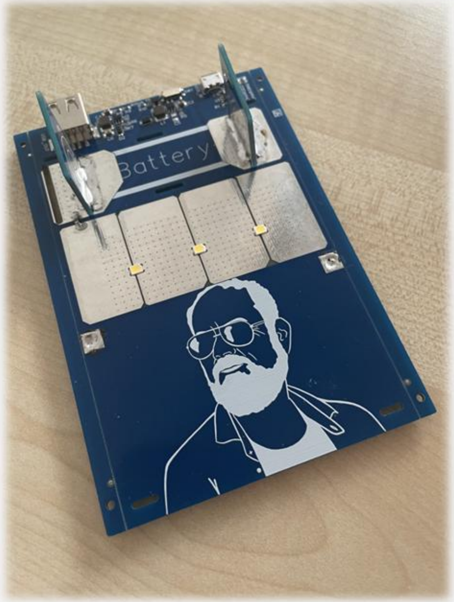
Beny LIGHTS

Jim O’Reilly, Lucy Munson,
Samuel Hinks, Leo Mills

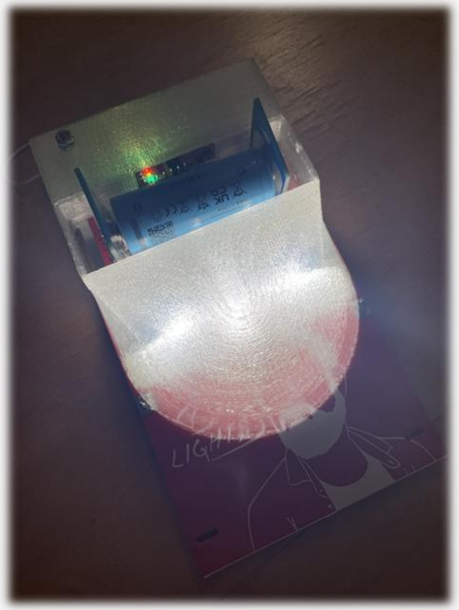


Beny LIGHTS: The Overview

The Problem



The Prototype



Our Output

The Future

Noam's Project

GM2 (Our Project)

Next Steps

The Problem

Kerosene lamps are the current lighting solution

- Inhaling fumes damages health
- Respiratory diseases decrease life expectancy
- Black carbon causes global warming

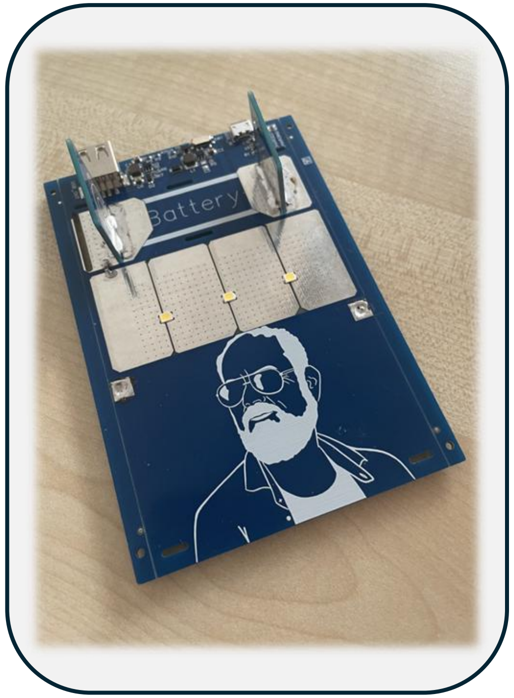
The Prototype

Board received from Noam had basic programming…..

- White LEDs increased in brightness on button press
- Coloured LEDs implemented simple counter

….but it was hard to use as a product

- No protective casing for the electronics
- LEDs blinding



Deliverable Aims

3D Printed Protective Case & Diffuser

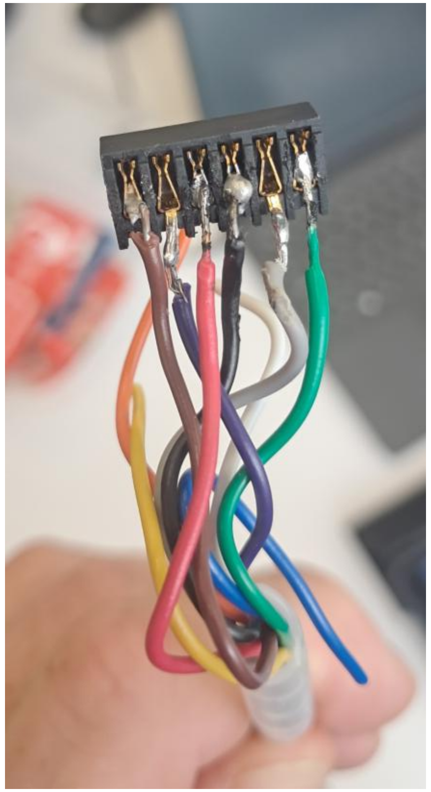
- Curved surface diffuses light
- Plastic protection over electronics
- Button press more reliable

Additional Programmed Features

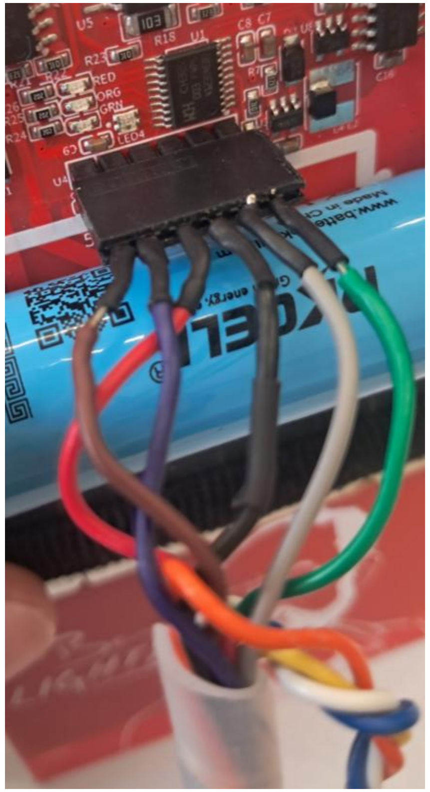
- Button press shows battery voltage
- Power conservation techniques
    - Sun detection
    - Sleep Mode
- Cost reduction options
    - RGB LED

Connector Cable

- Issues with unreliable connector
- Made a new connector
    - No longer crimping connector
    - Heat shrunk wires to prevent shorting



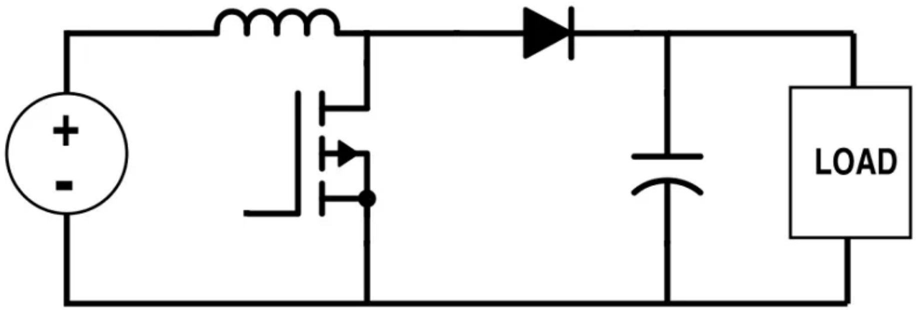
old

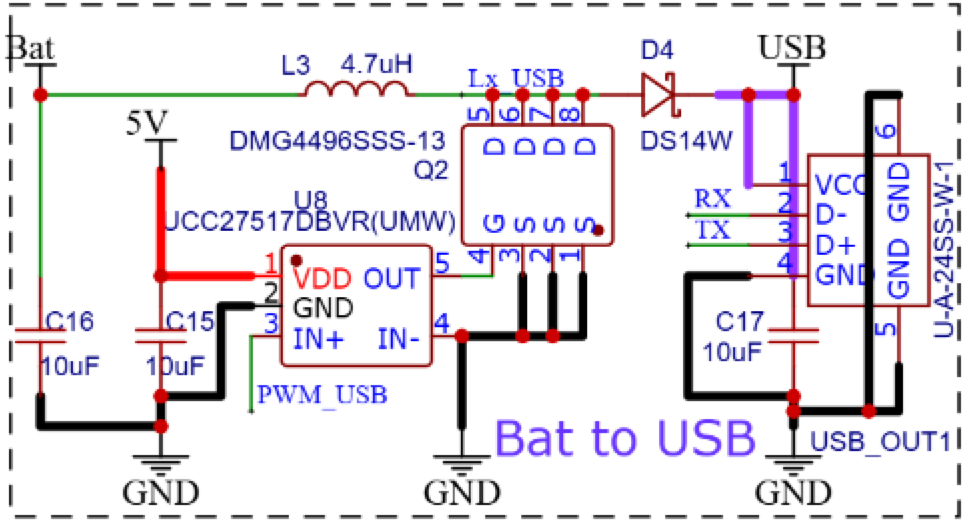


new

Reducing Overheating

- Inductor L3 was overheating, suggesting current PWM was faulty
- Lowered the ‘overvoltage’ parameter
- No longer overheats
- May need recalibrating to allow USB charging output





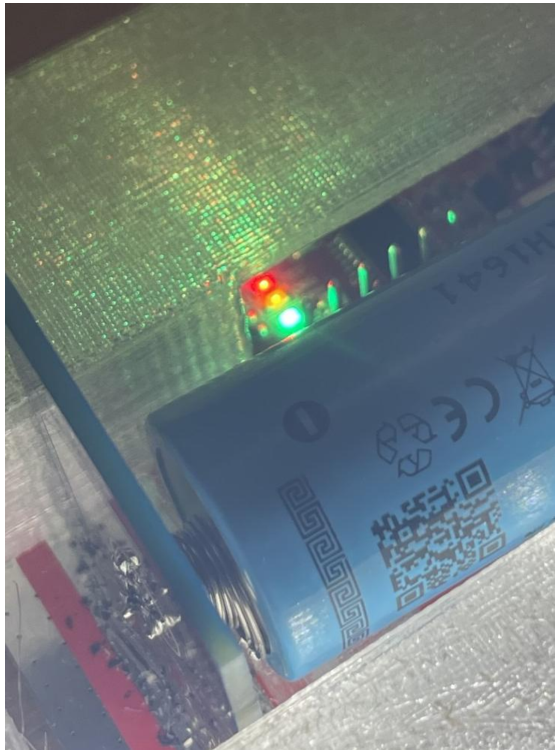
Boost converter circuit image from https://www.allaboutcircuits.com/technical-articles/understanding-the-operation-of-a-boost-converter/

Battery Voltage Displayed

- Button press lights up LEDs depending on charge levels

- LEDs are displayed for 10 seconds using a counter

Battery Voltage (V)	LEDs
Below 3.2	Red
3.2 - 3.4	Red + Orange
Above 3.4	Red + Orange + Green

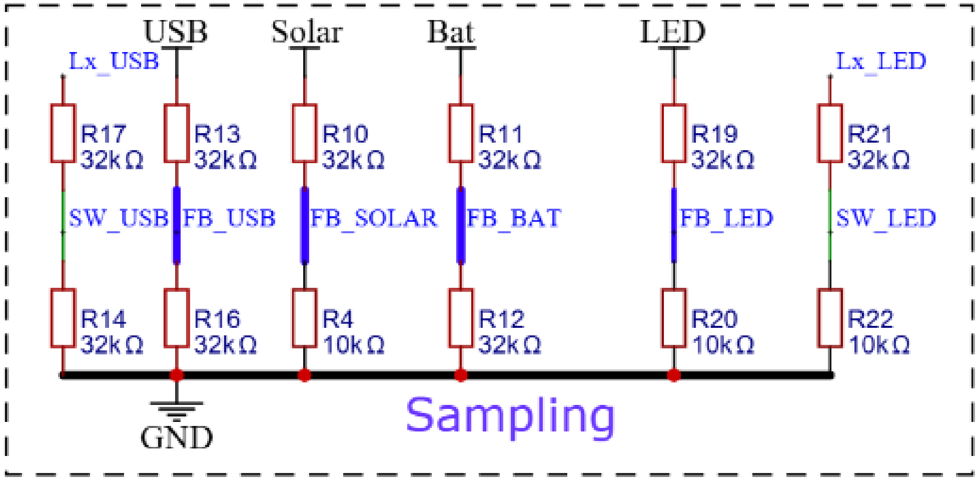


Multichannel ADC

Needed 4 samples shown in blue on the diagram

This is how each pin of the ADC works:

1. MCU reads voltage and records it as a 10 bit raw ADC value
2. Divide by 1024 to convert to value between 0 and 1
3. Convert to actual useful voltage using a multiplier

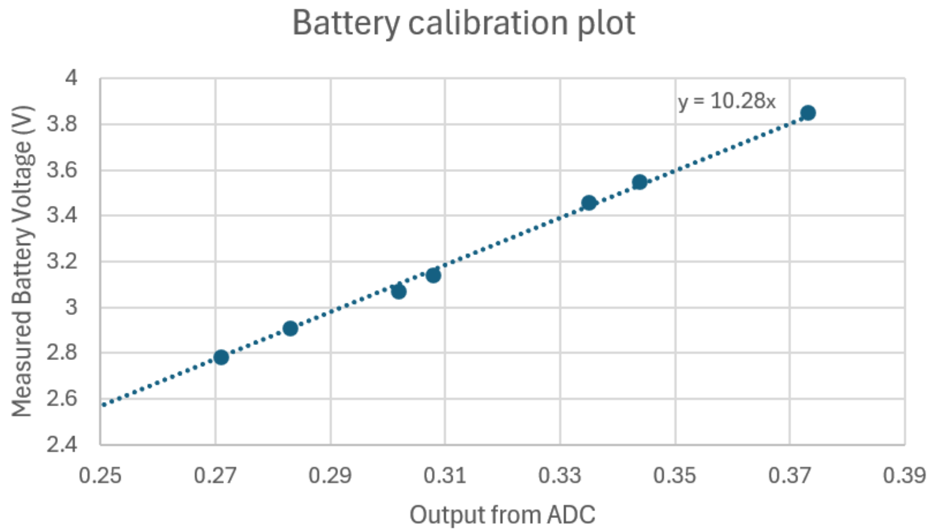


Calibration

- Linear relationship between the actual battery voltage and the value between 0 and 1

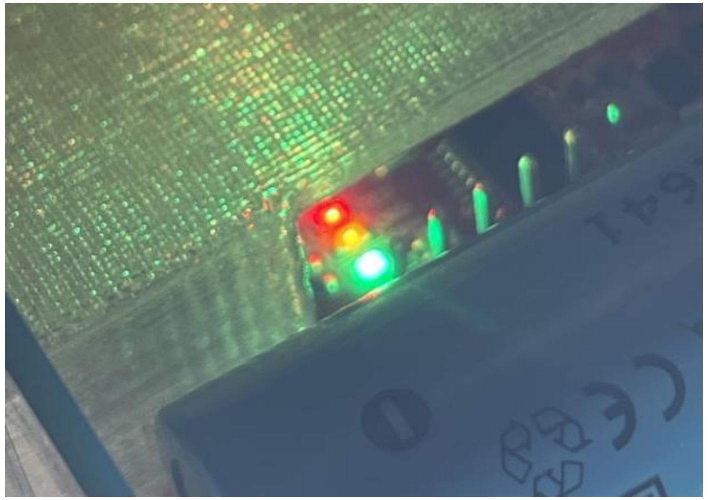
- Measurements were taken reading FB_BAT from serial and measuring battery voltage using multimeter

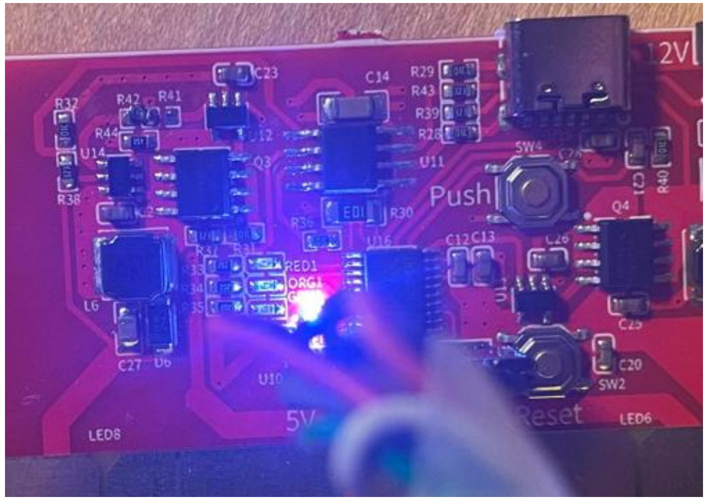
- Gradient of line shows BATmultiplier should be 10.28V



RGB LED

- Cost reduction option
- Sophisticated timing and lighting required
- We modified existing code to connect to correct pins
- Helpful for somebody progressing to the next stage of the project





Reduced Max Brightness

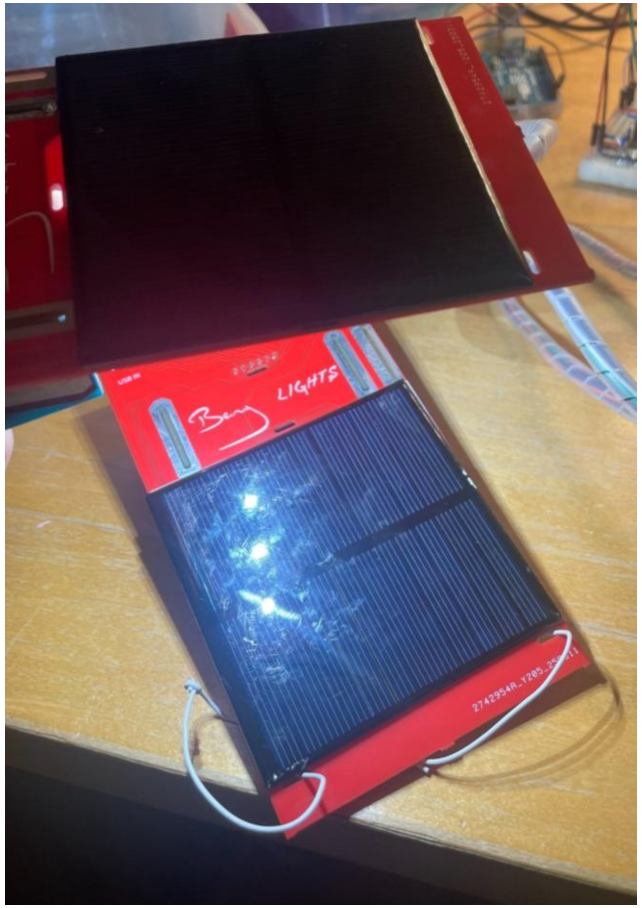
- Unexplained sudden shutdown
- ‘Browning out’ = current surge caused by battery overload
- Solution was reducing maximum brightness to limit sudden changes through trial and error

Original Sequence	New Sequence
[0, 25, 50, 75, 100]	[0, 10, 25, 45, 70]

- Additional benefit: reduced power usage

Solar Detection

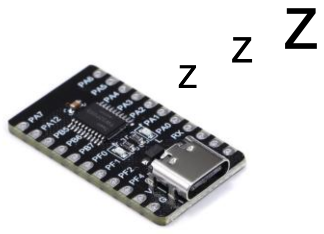
- On sun detection, white LEDs switch off
- User can overwrite by pressing the button
- Value of 0.27 was obtained from experimentation
- Should be tested in the actual environment



Sleep Mode & Interrupts

Microcontroller should enter sleep mode under the following conditions:

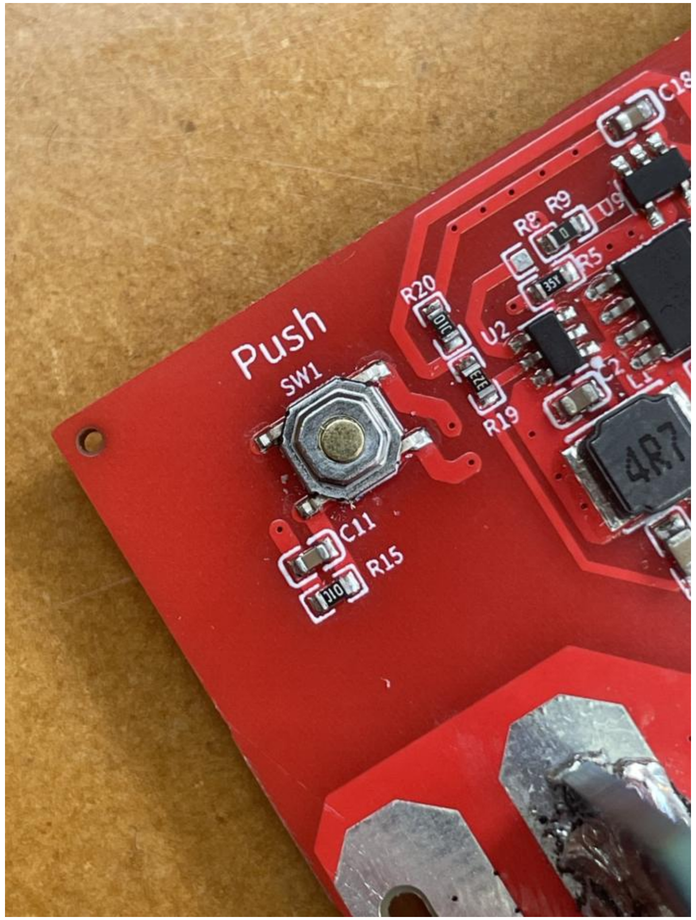
- Battery voltage drops below 3.2V
- The LED mode has been on 0 for over 2 minutes



How to wake up the MCU?	Challenge
Interrupt set on **push button** on PD0	• Board loses functionality: button function is overwritten by interrupt so the LEDs no longer switch on
Interrupt set on **different button**	• There is only one programmable button on our PCBs from Noam. • Reset button cannot be reprogrammed • Takes too long to print another board in this project timescale.
Periodic wakeup using interrupts at regular intervals	• Not as effective power conservation (uses power to wake up and go back to sleep) • The user has no way of forcing the MCU to wake up • Device becomes unresponsive and becomes impossible to wakeup

Extra Button

- Ideally introduce another button for interrupt
- This would enable the user to wake the MCU up from sleep mode without losing the original button functionality
- Sleep mode could actually be implemented usefully



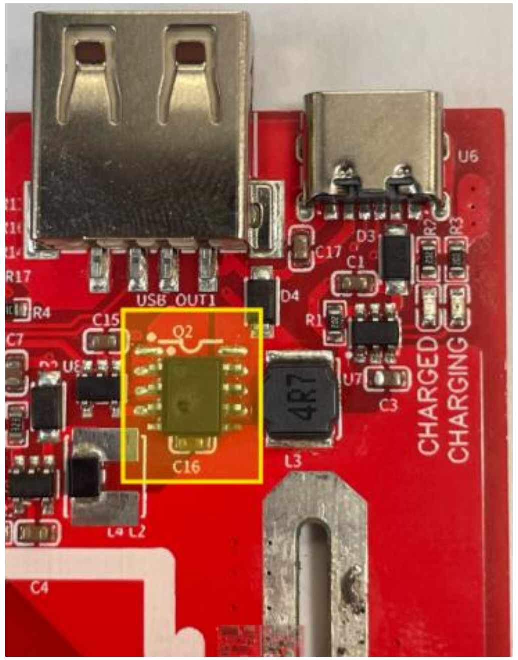
Averaging

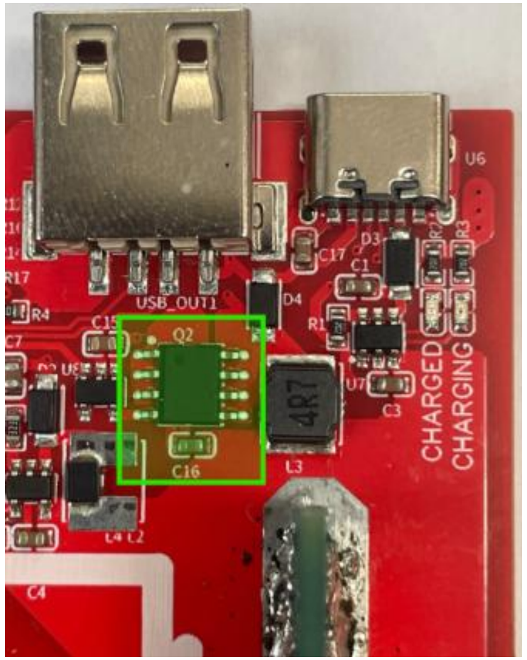
- Intermittent battery connection
- One incorrect reading of 0 V could cause the MCU to go to sleep
- Averaging smooths battery voltage
  - This code is implemented in the while loop to calculate avg_bat
- Same logic used for 4 channels
- **printFB_()** function allows floats to be printed in C

```
while(1)
{

for(x=0; x<samples; x++)
{
total_bat = total_bat + getFB_BAT();
}

avg_bat = total_bat/samples;

printFB_(avg_bat,"avg_bat");

total_bat = 0;


}
```

Check Soldering

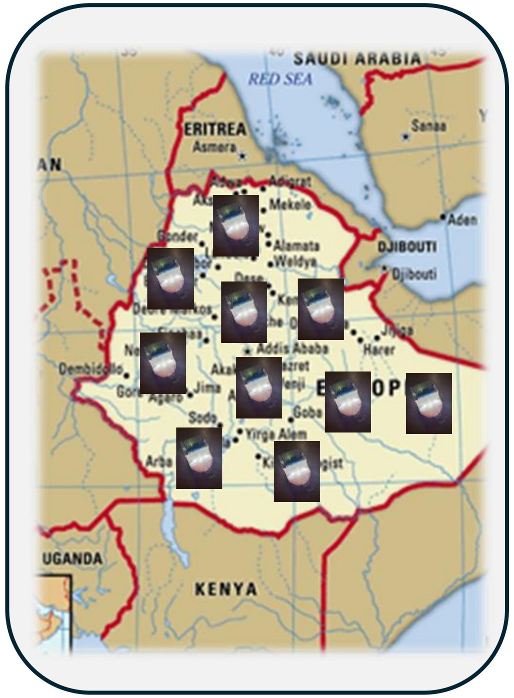
- One of the boards blew and started smoking
- One of the components was soldered wrong
- Safety concern and fire risk
- Need some checking system before distribution





Next Steps

- Introduce another button to the board so user can wake up from sleep mode
- Implement phone charging capability
- Cost reduction by replacing current battery voltage code with RGB LED



Deliverable Aims

3D Printed Protective Case & Diffuser

- Curved surface diffuses light
- Plastic protection over electronics
- Button press more reliable

Additional Programmed Features

- Button press shows battery voltage
- Power conservation techniques
  - Sun detection
  - Sleep Mode
- Cost reduction options
  - RGB LED

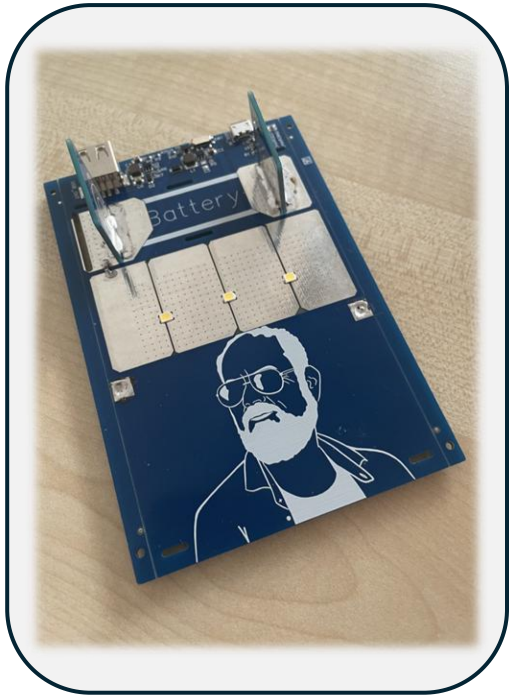
Mechanical: Case

Board was hard to use as a product...

- No protective casing for the electronics
- LEDs blinding to the naked eye

and had additional scope for functionality

- Kickstand for support when charging
- Holes to hang from roof
- Protection for ports



Software Choices

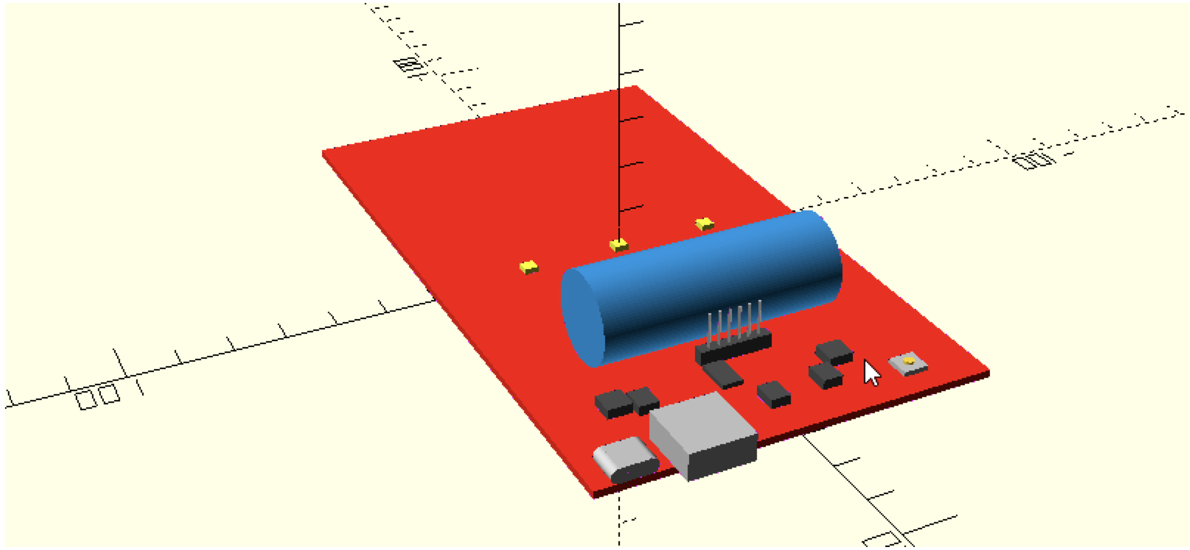
Selecting software for 3D modelling

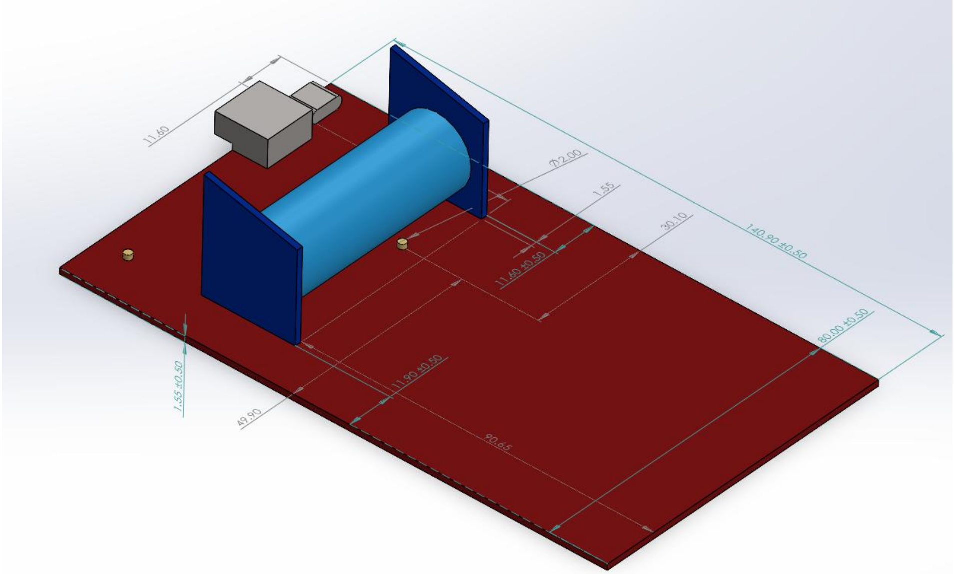
- Had experience in Solidworks modelling from earlier in course
- However, saw the arguments for the advantage of shareability of models over GitHub in OpenSCAD
- Wanted to be able to provide files to supervisor or construction teams that could be opened using open source software
- Decided to model initial designs in the familiar Solidworks and then replicate design in OpenSCAD once design was complete and familiarity with OpenSCAD had improved.

Case Design

Modelling the PCB

- Done in both Solidworks and OpenSCAD
- Meant that we had an assembly that we could virtually test with
- Provided dimensions for future work on the case

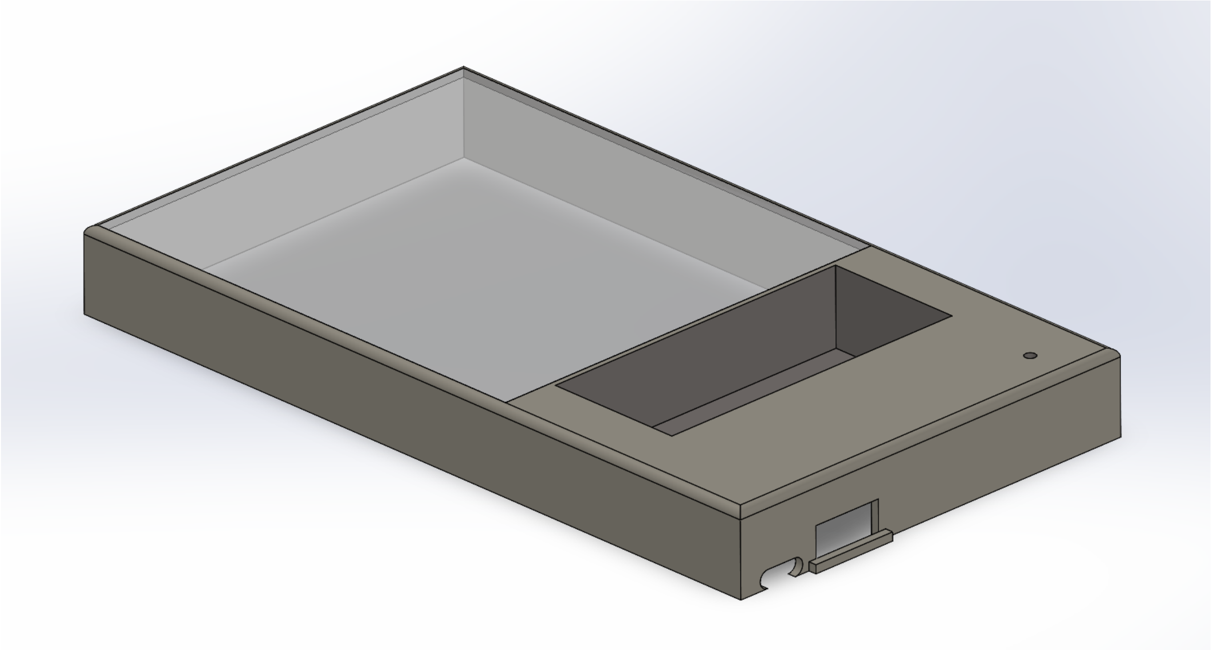




Case Design

Meeting basic requirements

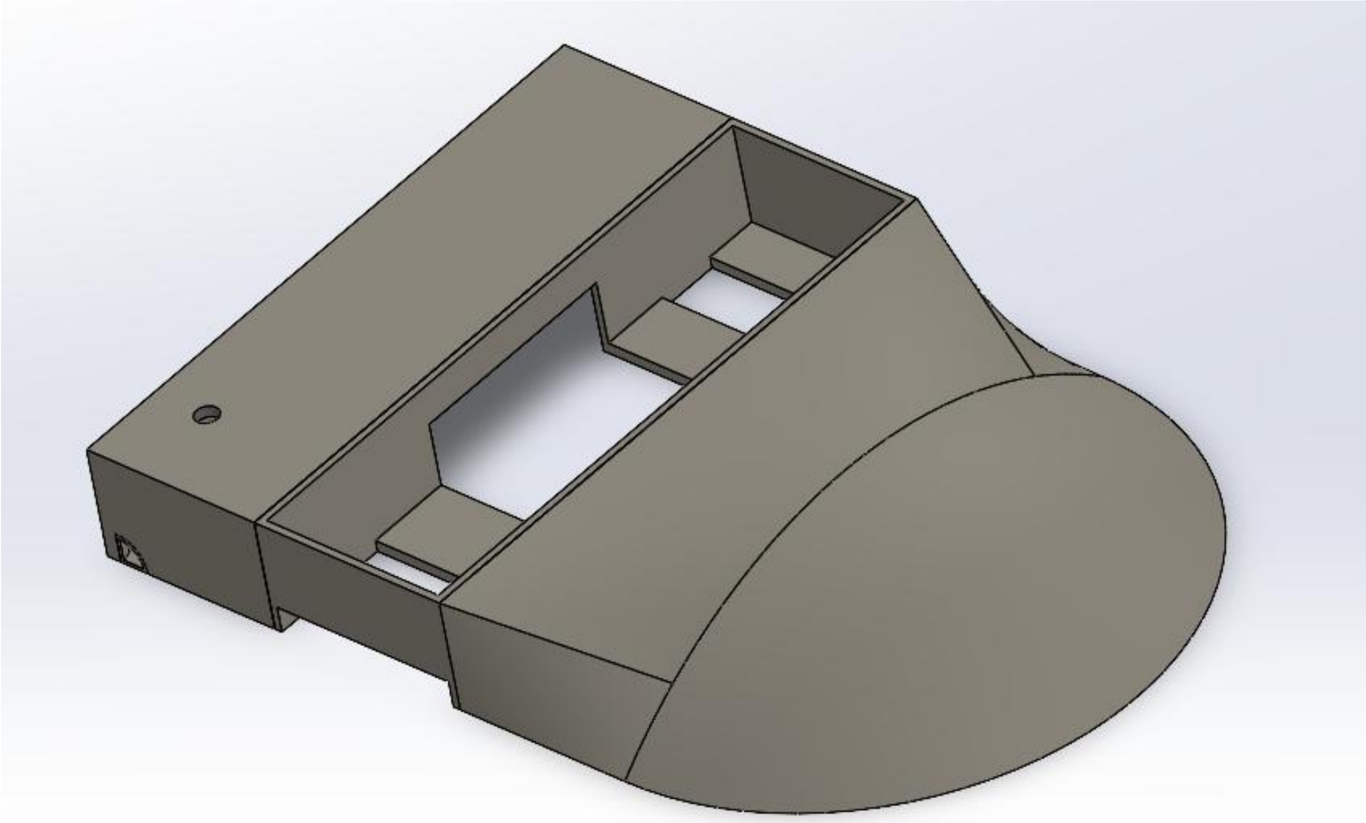
- Square case with simple cover, designed with a transparent cover over the LEDs



Case Design

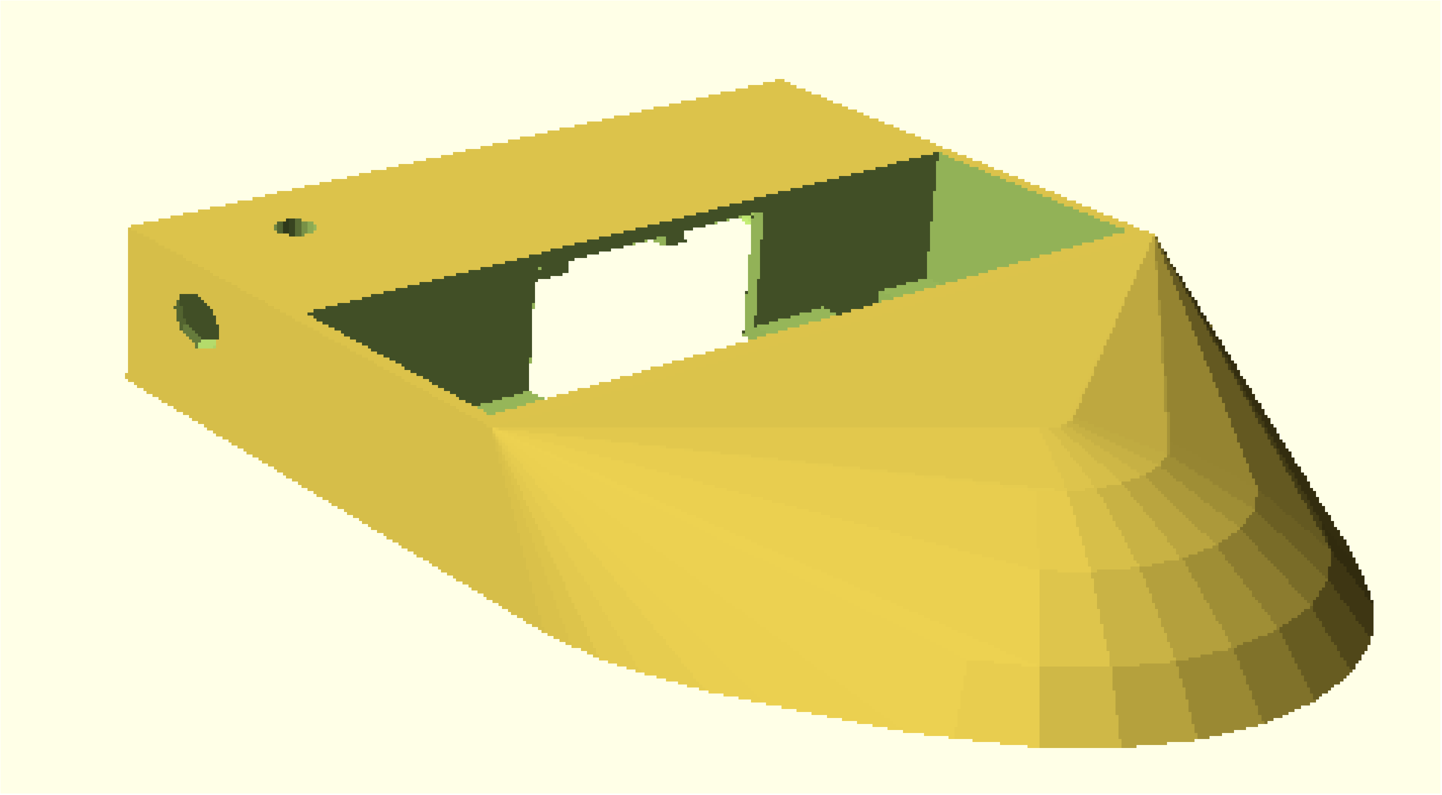
Improving diffusion of light

- Curved diffuser to improve refraction of light



Replicate in OpenSCAD

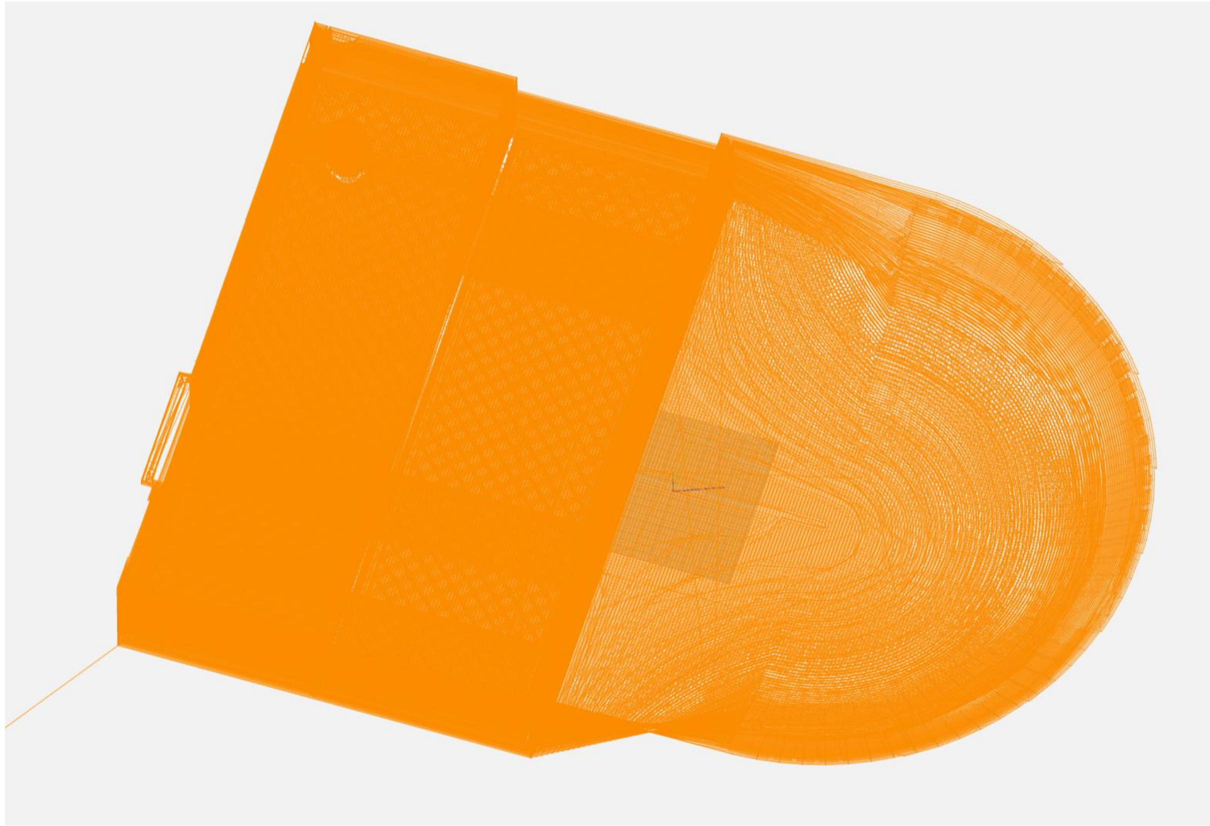
Simplify curves and structure while maintaining essence

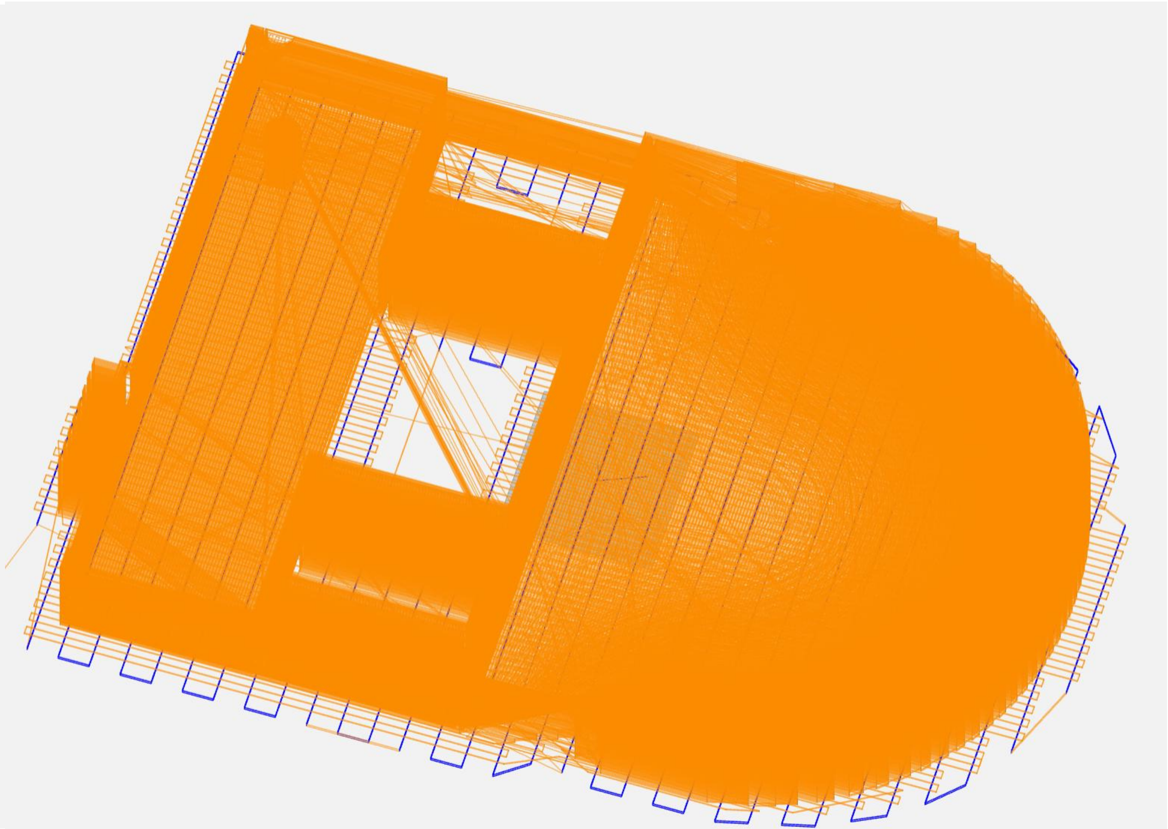


3D Printing

Issues with GCode

- Double layering of STL – fixed by editing pin cutout

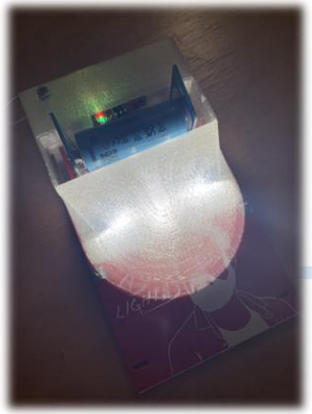




3D Printing

Trial Prints

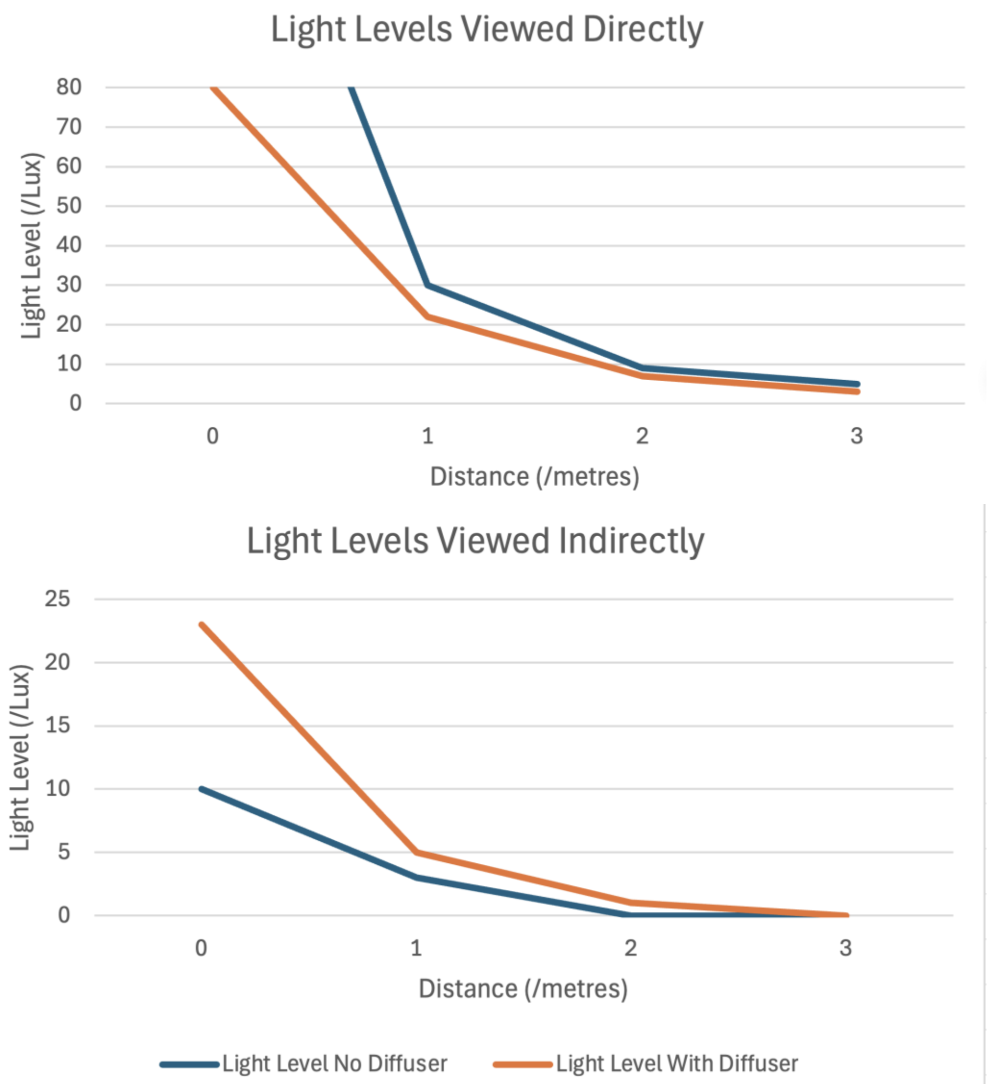
- Transparent filament prints translucent in reality
- Several prints were made to test fit on case
- To save weight, the case height was reduced slightly – however this leaves the top of the case more exposed



Case Testing

Aim 1 – Improve Light Diffusion

- Took case to the SST lab to test how well light was diffused.
- Measured light levels directly and indirectly
- Successfully increased diffusion



Case Testing

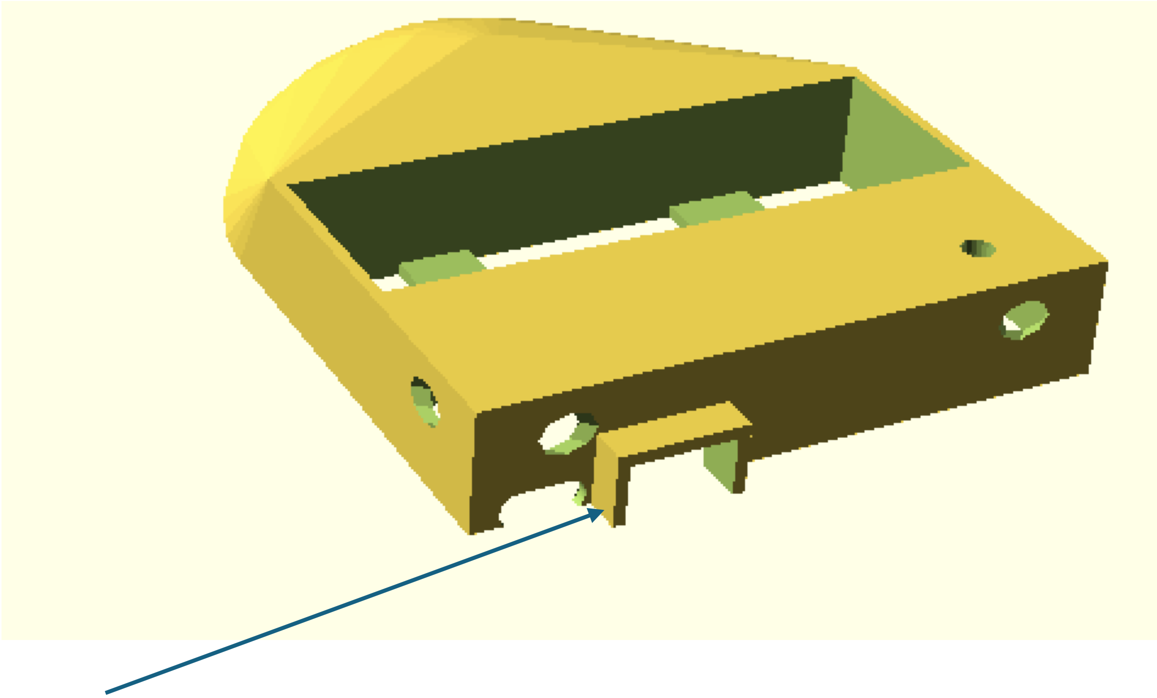
Aim 2 – Improve Survivability

- Took case outside for 1m drop test
- Dropped on flat and rocky surfaces
- Aimed to find out primary causes of failure
- Found that battery supports (vid 1) and USB-A port (vid 2) were the primary causes of failure – the case itself withstood impacts

Modifications

Improved Strength

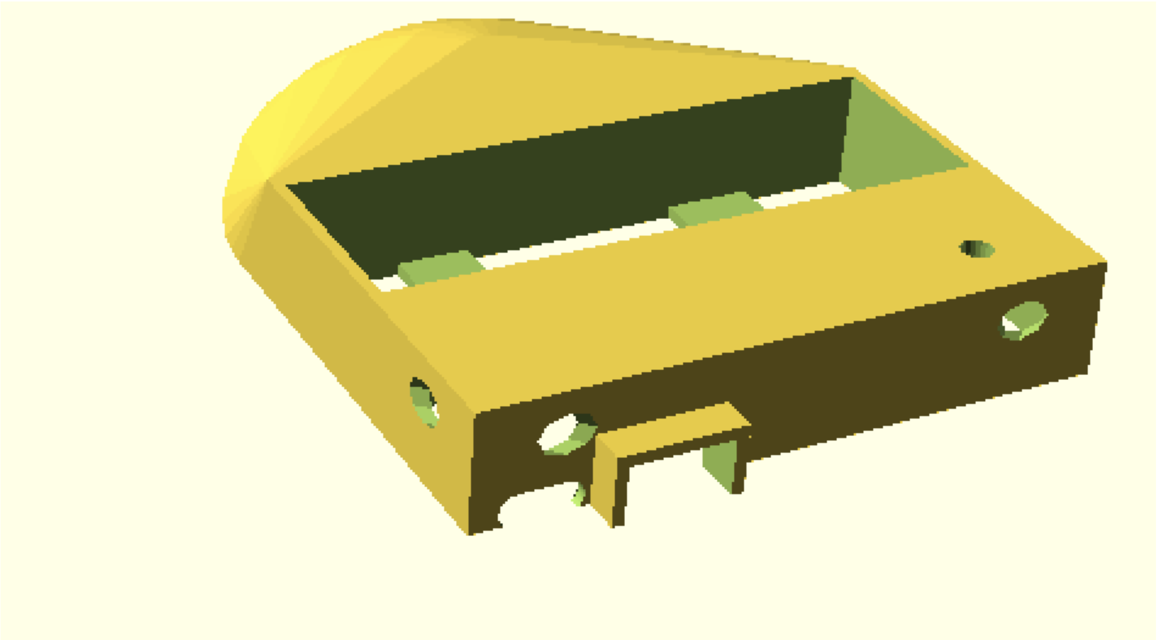
- Added extra support around USB-A port to protect it against impacts
- Would recommend considering adding extra material around battery supports, depending on strength of soldering (industrially this may be better than our homemade attempt)



Modifiability

Comments and notes

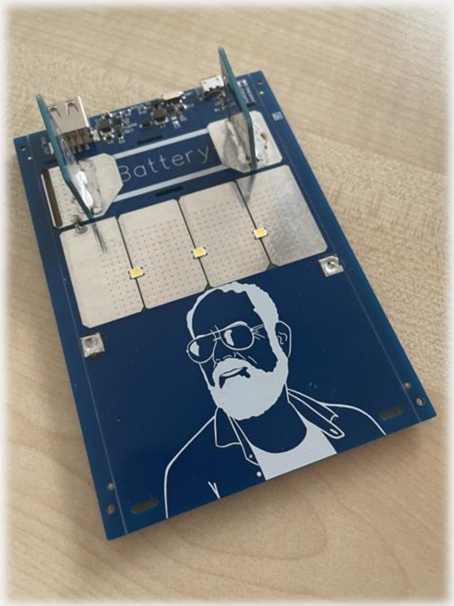
- OpenSCAD file modified to make it easier to interpret and modify where needed



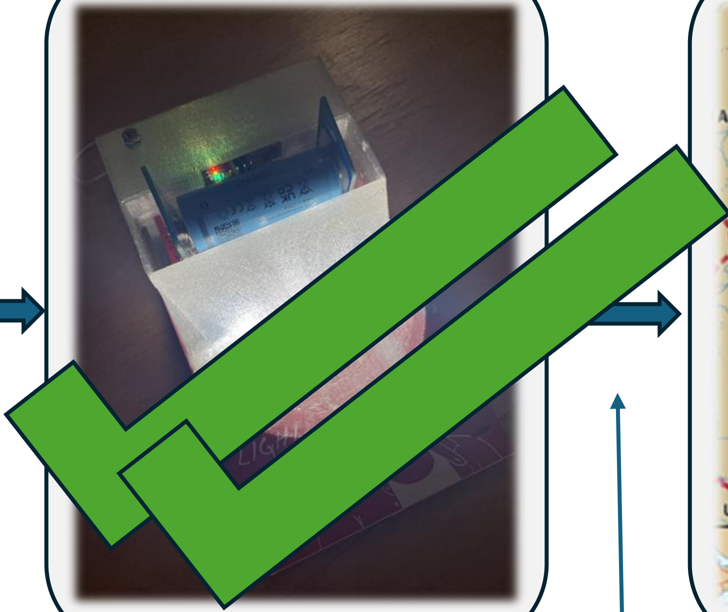
```
                //Case Design

      //Key dimensions

//Note: Width: x-axis  Length/len: y-axis  Depth: z-axis     all measurements in millimetres

    //Relevant PCB dimensions
plate_thickness = 1.55;
plate_y = 140;
LED_length = 2.8;
LED_y=74-LED_length/2; //Distance from top of plate to center of LED

    //Primary case dimensions
case_height = 15;
case_width = 80;
box_length = 51.75;
shell = 1.5;

    //Primary void spaces
chips_cutout_len=24.6-shell*2;
batt_slot_len = 25.15;

    //Other holes and cuts
//Ties linking under the battery, numbered left to right looking down on the case from the USB port end
tie_link_0 = 8-shell;
tie_hole_1 = 8;
tie_link_1 = 9;
tie_hole_2 = 53-25;
tie_hole_3 = 16-shell;
//Data transfer pin slot
pin_cut_depth = case_height-shell;
pin_cut_width = case_width-25-27;
//Hole for button push pin
push_pin_dia = 4;
push_depth = 3.36-plate_thickness+3;
```

Beny LIGHTS: The Overview

The Problem



The Prototype



Our Output

The Future

Noam's Project

GM2 (Our Project)

Next Steps

Next Steps

Commercial Printing

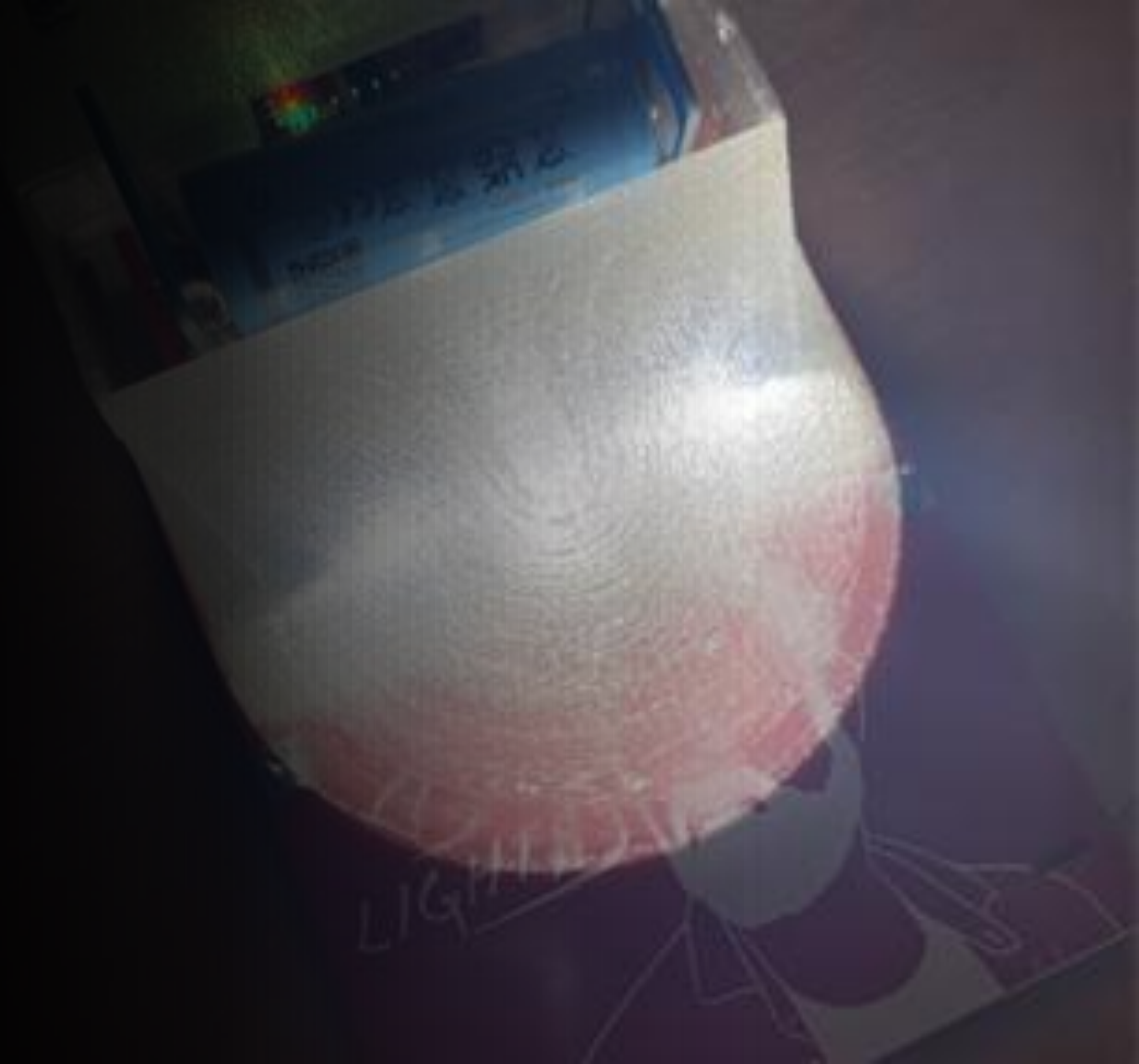
- In the ideal situation, we will have a printer in-situ producing these cases. They can be low quality – in fact, the diffusion benefits from it – so home printing is an advantage on both cost and time.

Adaptability

- With the efforts made to make the code accessible, our hope is that this basic case design will be modifiable even if design requirements or the PCB shapes change

Summary

- **Product meets expectations and Noam’s requirements**
- **Board is programmed correctly**
- **Case protects electronics and diffuses light**
- **All clearly documented and ready for handover in our GitHub**



Beny LIGHTS