Portable LED Lamp

by mattwach

***Note**: this page focuses on building the lamp. If you want to customize the firmware, hack on the 3D models, or read about more design details, see the associated github project.*

If you need light but don't have an outlet available, it's nice to have an efficient light that can accept a wide variety of batteries.

The lamp can accept a wide range of voltages - between 6 and 30V. This allows it to be **used with different batteries you may have lying around**, including 18650 packs, RC batteries, 12V lead acid, etc.

The lamp is designed to be **very efficient** and puts out very little heat, even at full power. A dimmer knob can be used to adjust the power consumption and light output. When "off" the lamp consumes less than 40 uA of power in its "sleep" state.

Voltage and current are monitored on a small OLED display. This allows you to estimate battery life at a given light level and make adjustments as needed for your situation.

The device has **automatic shutoff when the battery voltage gets too low**.

The device can autodetect the battery you are using with options for manual override.

Before we get too far, **a disclaimer**:

***Using batteries carries a risk.** Making mistakes can lead to batteries catching on fire. Sometimes batteries catch on fire even when everything is done correctly due to unknown issues during manufacturing or shipping. It is expected that you understand the risks associated with the battery you choose. Even though the hardware and software in on this page have been tested, you should **not assume** that your build and batteries will operate without problems. **I take no responsibility** for issues you may encounter while following this guide or using the provided software. Using a safety container for your battery is recommended.*

Supplies:

Needed Supplies

This is what I used. If you understand the design, you can swap out many of the parts below for alternatives.

1. **3.3V Arduino pro mini**: $10 (or a low cost clone $3)
2. **3 PicoBuck LED drivers**: $48
3. **10 3W leds (solds in packs of 5)**: $16 (or cheaper ones on ebay $5)
4. **128x64 I2C OLED (US shipping)**: $6 (or CN shipping $2)
5. **3.3V Regulator LM2936**: $2
6. **INA260 Voltage/Current monitor**: $10
7. **10k rotary potentiometer**: $1
8. **2x tactile buttons**: $1
9. **5.5mm jack for power**: $2
10. **P-Channel power MOSFET**. I used the FQP27P06 but many could work. It just has to have a Vdss > 30V

and an Id > 2A ($1)

11. **N-Channel MOSFET**. I used the BS170 most but anything with a Vdss >30V will work. (< $1)
12. **Resistors**: I used 3x 47k, 1x 10k and 1x 1k. Most of these values are not critical, ballpark will do. The one exception is the 1:5 ratio between the 47k and 10k should be similar.
13. **PCB boards**. I used a 30x70 and 50x70 prototype board. If you want to send the design to a fab, see the github files.
14. **M2 and M2.5 bolts** if you choose to use the 3D printed housing.
15. **Some hookup wire**. I think 22 gauge silicone wire is a good fit but you can use what you have.
16. Optional: **Connector** between input PCB and main PCB. I went with 8 pin JST XH connectors but many different options can be used.
17. Optional: **Some female pin headers** will allow you to plug and unplug the OLED, Power Monitor and Arduino. This can help with troubleshooting and protects the parts when soldering.

Price Range: $60 - $150, depending on where you get components, optional parts, shipping, etc.

Needed Tools

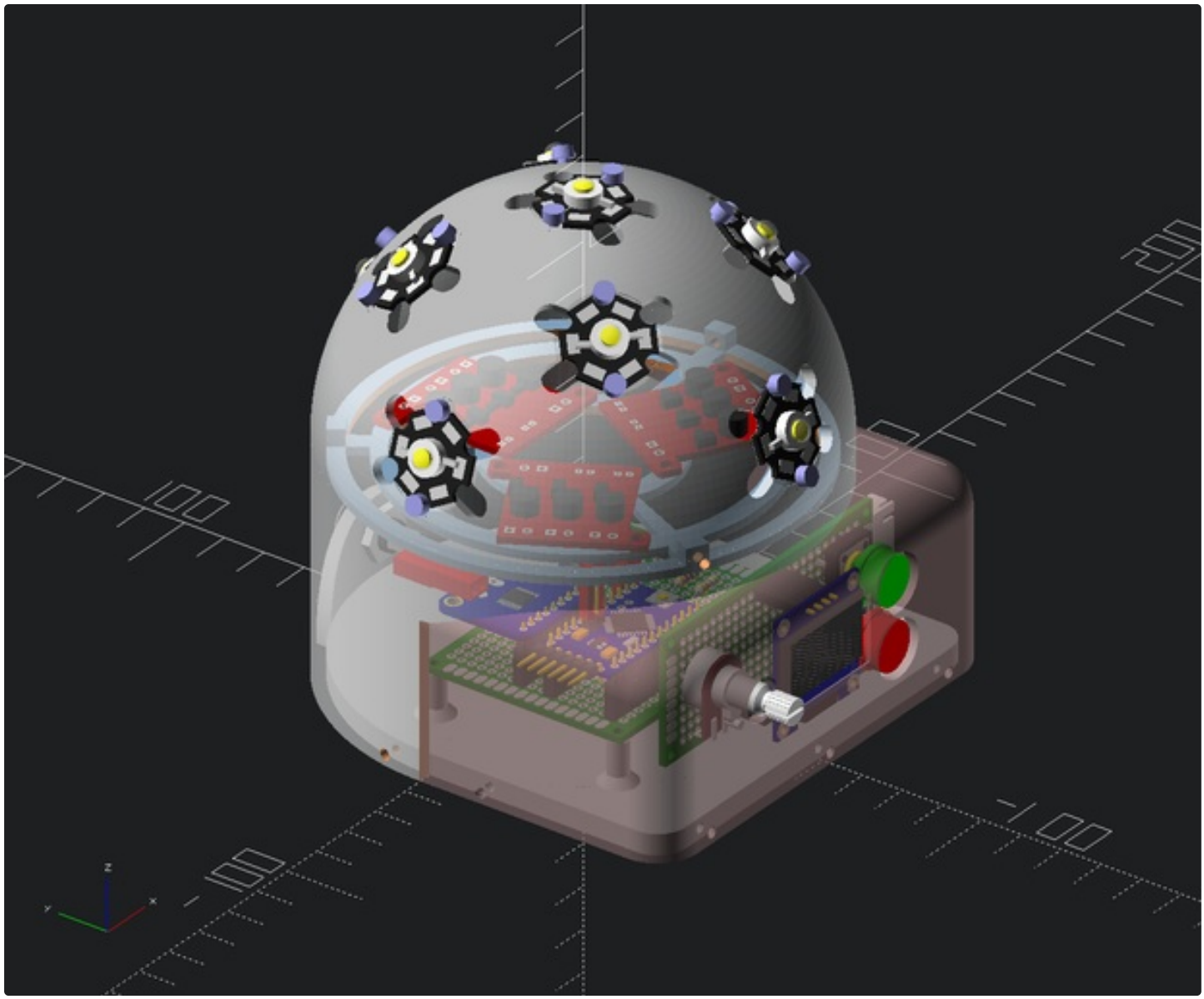
1. **A 3.3V FTDI programmer**. The linked one is $15 but there are many cheap alternatives.
2. A **soldering iron** and solder
3. **Hex wrenches** for assembly.
4. Optional: an **ESD wrist strap** is recommended when handling the MOSFETs.
5. Optional: a **breadboard** will be useful for incremental validation.

This project also includes .stl files for a **3D printed enclosure**. You can print it yourself, send the files to a service, or DIY your own case solution.

Needed Software

1. **AVRDude** for uploading firmware to the pro mini (free).
2. Optional: **AVR GCC tools** if you want to hack on the firmware or write your own. (free)
3. Optional: **OpenSCAD** if you want to hack on the 3D model files. (free)


183mA
G-PLUS SERIES



Step 1: Main Schematic Overview

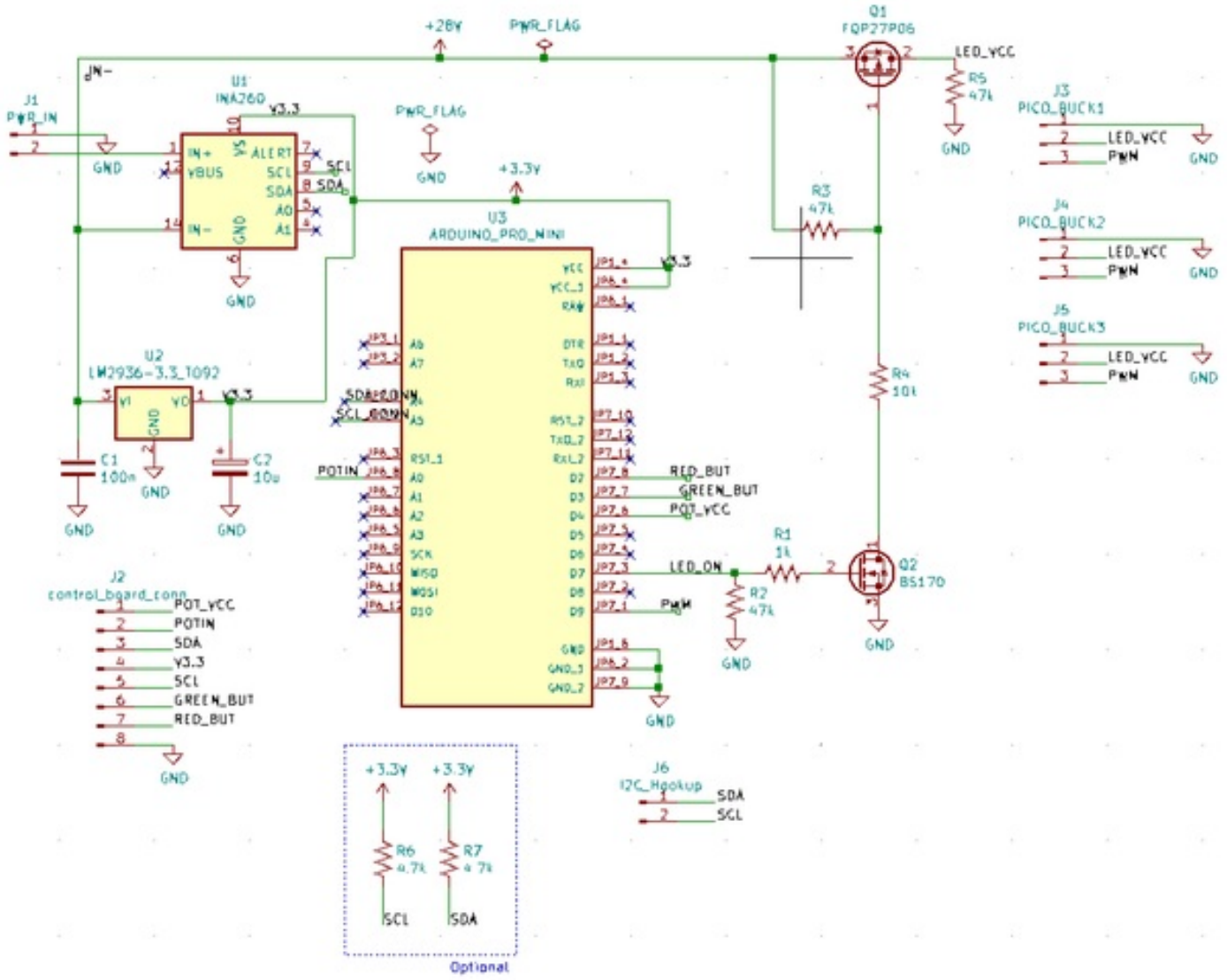
Let's start with an overview of where we are headed. The schematic shows the final circuit with everything connected. If this information looks overwhelming, **feel free to skip it** as we will be walking through the design in the upcoming steps:

- **Power comes in at the top left on J1**, Pin 2. It feeds directly into the V+ port of an INA260 power meter.
- The power flows through the V- port of the INA260. The INA260 will measure current that passes through V+ and V- and the voltage between V+ and GND
- The V- of the INA260 connects to 2 devices
    - One is the **LM2936 voltage regulator** on the left (U2). This regulator drops the voltage down to 3.3V which is needed by the Arduino pro mini, the INA260 and the OLED. Although the pro mini has a built in 3.3V regulator, this built in regulator can not handle the 30V maximum rating on V-. The pro mini regulator also has a higher "quiescent current" which means that it is less power-efficient than the LM2936.
    - The other device connected to V- is a **PFET (Q1)**. This is used to **cut power to all to the**

PicoBuck LED drivers. The 47k pull resistor (R3) pulls the gate to V- by default meaning that the PFET is off by default.

- The resistors and Q2 (NFET) are needed to switch the PFET on. The pro mini can not turn on the PFET directly because the voltage at the PFET gate is too high, thus an NFET (Q2) is used as a middle-man. The resistors R3 and R4 are used to keep the Vgs specification of the PFET in range. The R2 resistor keeps Q2 off by default. To turn on Q2, the arduino pro mini must drive D7 high. The R1 resistor consumes potential oscillations that can develop when switching Q2 on and off.

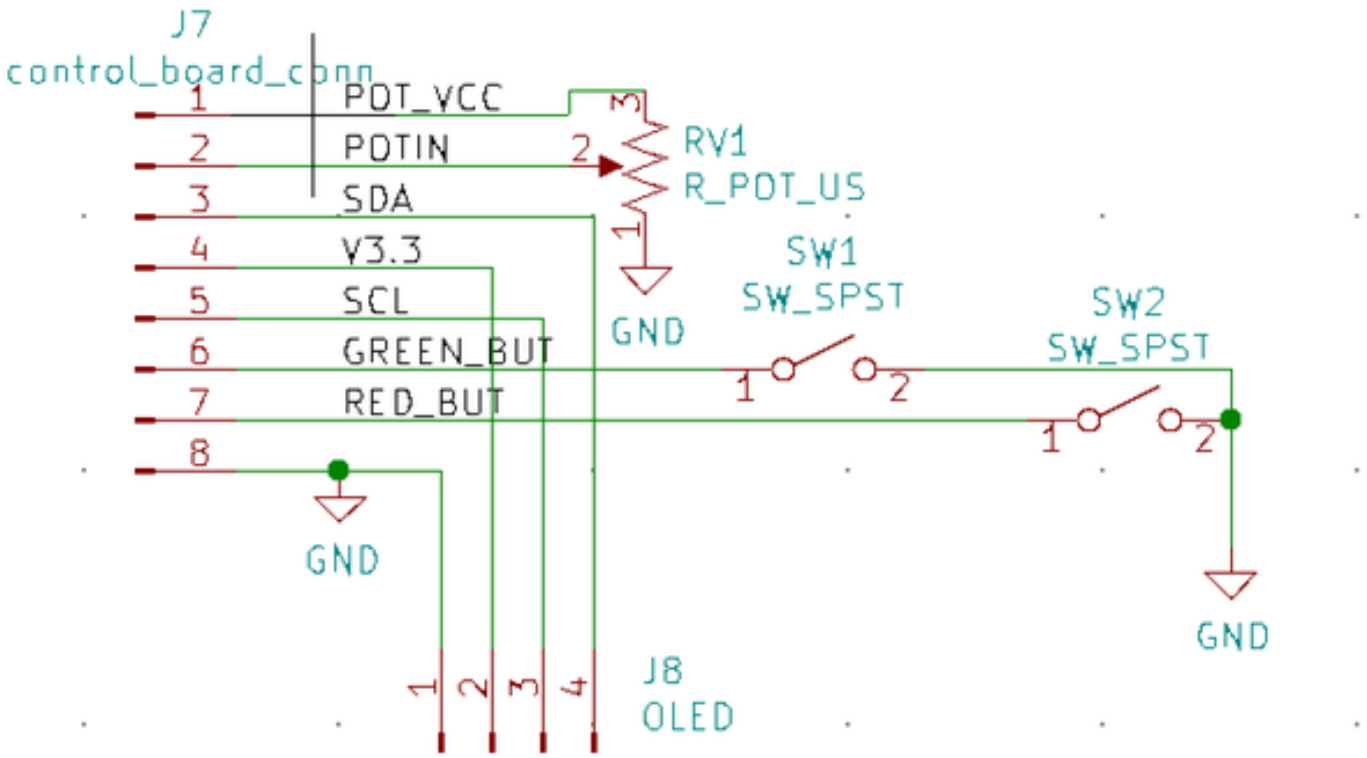
- Back to Q1, the PFET either blocks (nearly all) current or allows (nearly all) current, depending on it's state. The receivers of this current are 3 PicoBuck LED drivers, which are used to drive 9 3W LEDs.
- Let's focus on the pro mini's interface pins
    - **A4 (SDA), A5 (SCL)**: These are used to talk to both the INA260 and OLED via **I2C**. I2C is a bus architecture meaning that A4 connects to SDA of both the INA260 and OLED at the same time. Same for SCL.
    - **A0**: This connects to the center leg of a **potentiometer for dimming the LEDs**.
    - **D4**: This connects to the voltage (right) leg of the potentiometer. You could hook this leg directly to the 3.3V supply, but then the potentiometer would burn power even when the light is off. By using D4, the pro mini can cut the power to the potentiometer when the unit is off, saving 330 uA of current draw.
    - **D2**: This connects to the **"green" push button**. The other side of the button is connected to GND. This configuration is called "open drain". Basically a pullup resistor internal to the pro mini keeps D2 high at 3.3V but this voltage can easily be pulled to ground by something like a button - which the pro mini will detect as a button press.
    - **D3**: Same idea as D2 but for the **red button**
    - **D7**: The arduino drives this pin to 3.3V to turn on Q2, which in turn turns on Q1, which **turns on the LEDs**. To turn off Q2, we have the choice or driving D7 to GND or re-configuring it as an input (Hi-Z) and letting R2 pull the gate to ground. The firmware takes the later choice.
    - **D9**: This pins sends out a PWM signal. It tells the PicoBucks **how much to dim the LEDs**. The firmware makes this decision by considering the state of the lamp (is it on? is the voltage ok?) and by reading the potentiometer value
    - **3.3V VCC, GND**: Needed to power the chip



Step 2: Input Schematic Overview

The input PCB is wired on a separate board and connected via an interface cable to the main board. You don't have to use an interface cable if you are making your own case and want everything on one board.

Every component on this board was described in the previous step. If it's not clear what is going on, the upcoming breadboard examples should help.



Step 3: Initial OLED Hookup and Firmware Test

Note: Consider all of the breadboard steps optional exercise for fun, education and part verification. If you want to skip ahead to the final build, feel free (and good luck).

For this step, we are just going connect the Arduino Pro mini to the OLED and upload firmware to the pro mini.

I recommend using a breadboard for this step.

1. Add the pro mini and OLED to the breadboard
2. The wiring to the OLED is
    1. Pro Mini GND → OLED GND
    2. Pro Mini VCC → OLED VCC
    3. Pro Mini A4 → OLED SDA
    4. Pro Mini A5 → OLED SCL
3. Now attach your 3.3V FTDI programmer to the Arduino Pro Mini Interface pins as shown in the photo and connect the USB cable to your PC

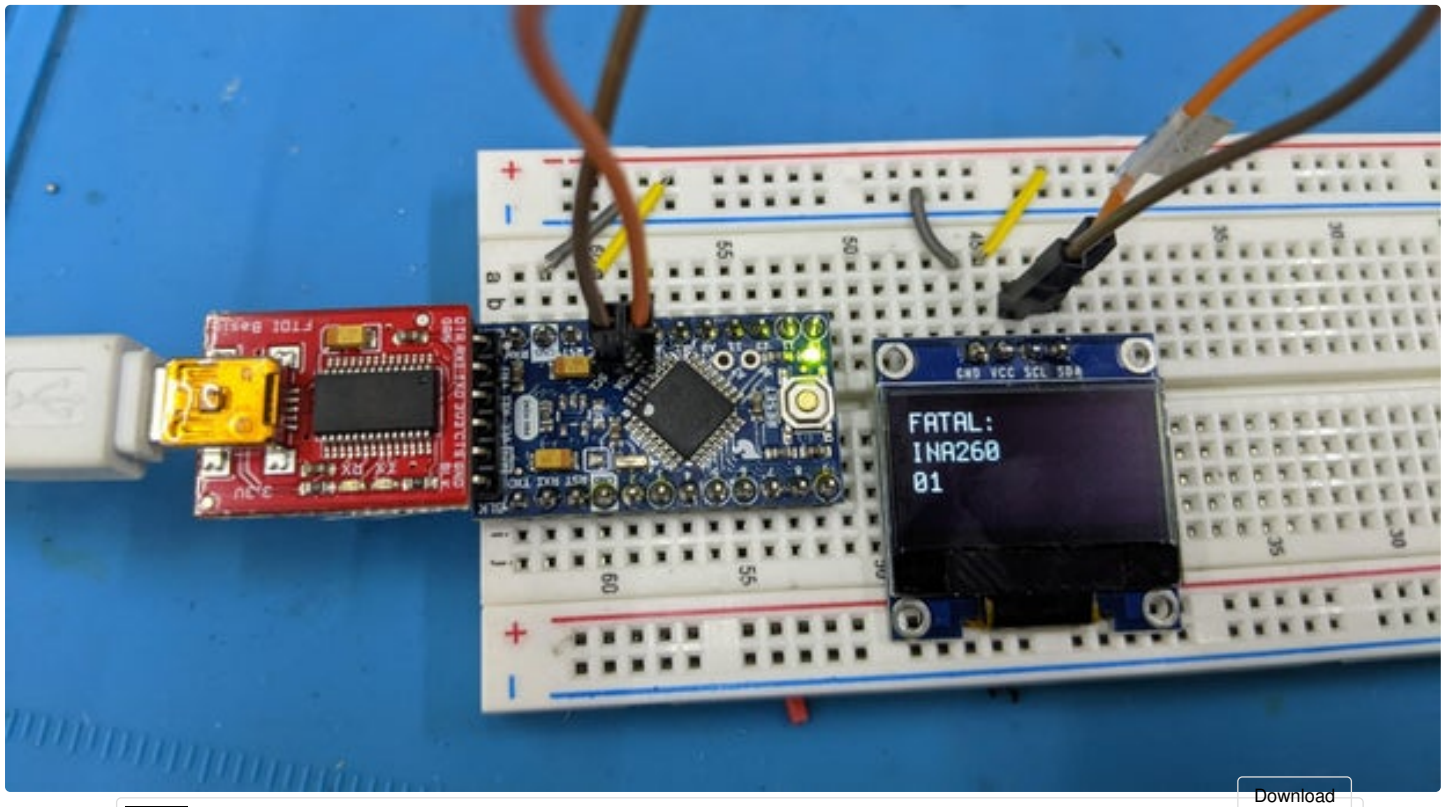
Now you need to upload the **led_lamp.hex** firmware file (link is below). This requires avrdude to be installed on your computer.

The command I used was:

```
<strong>avrdude -Cavrdude.txt -v -patmega328p -carduino -P/dev/ttyUSB0 -b57600  -D  -Uflash:w:led_lamp.hex:i</strong>
```

I tested this in Arch Linux and Debian Linux but avrdude works on Mac and Windows too - you will need to adjust (or omit) the -P setting to correctly point to your serial device. Here are some avrdude docs that might help with windows or mac.

You should see the error "FATAL: INA260 01". Although an error is being reported, **things are going well** if you see this message (as we have not added the INA260 yet).



Download

https://www.instructables.com/ORIG/FGZ/BCWE/KSQEIQ0H/FGZBCWEKSQEIQ0H.hex

Download

https://www.instructables.com/ORIG/FD0/RKZW/KSQEIQ0P/FD0RKZWKSQEIQ0P.txt

Step 4: INA260 Current Monitor Setup

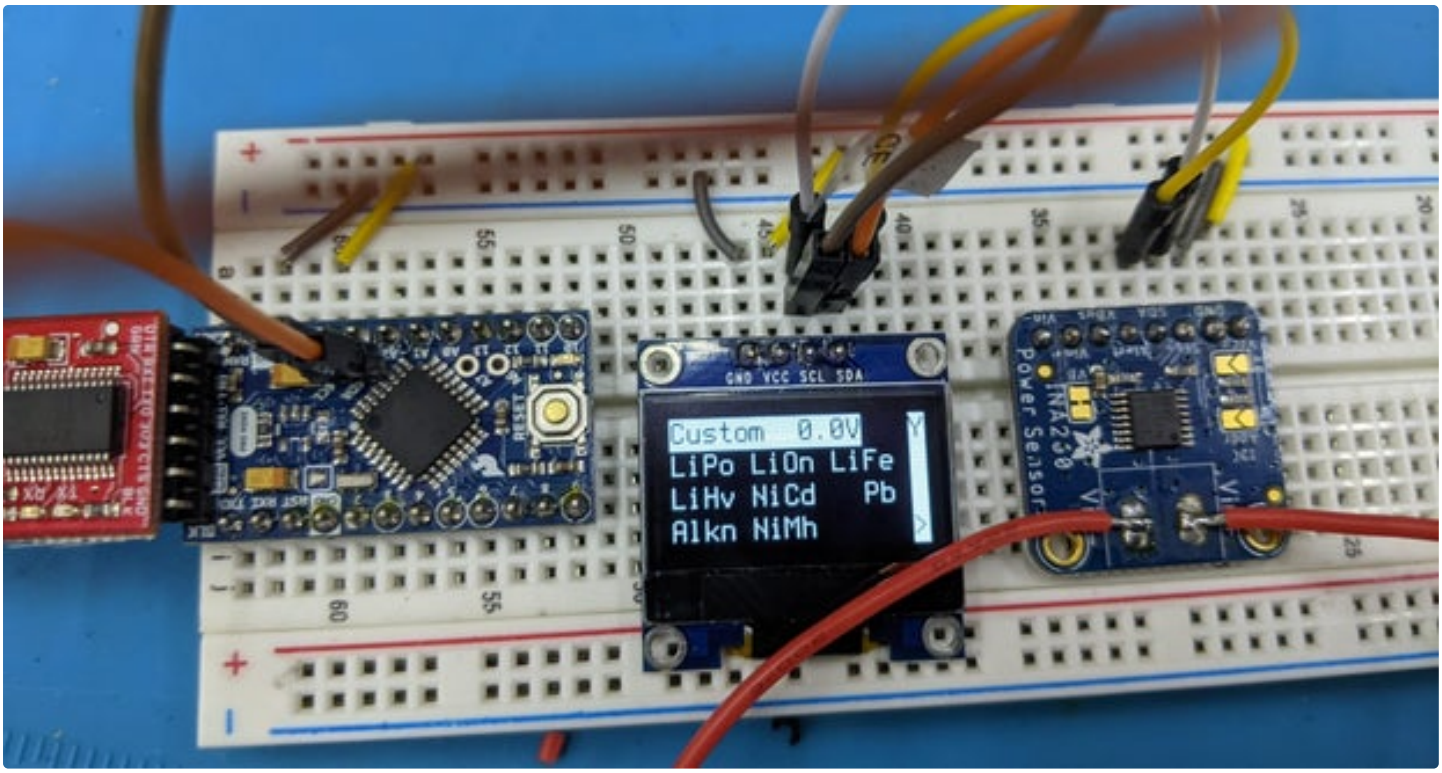
The INA260 current monitor is used to measure the voltage and current of the LED.

For now, we can just connect it onto the breadboard (the red wires in the photo are not connected to anything)

1. Connect Pro Mini GND → INA260 GND
2. Connect Pro Mini VCC → INA260 VCC
3. Connect Pro Mini A4 (SDA) → INA260 SDA
4. Connect Pro Mini A5 (SCL) → INA260 SDL

You will notice that all of the connections above are also connected to the OLED which is fine.

Power up the unit again and you should see it asking what type of battery you have as well as telling you you have zero volts.



Step 5: Adding the Voltage Regulator

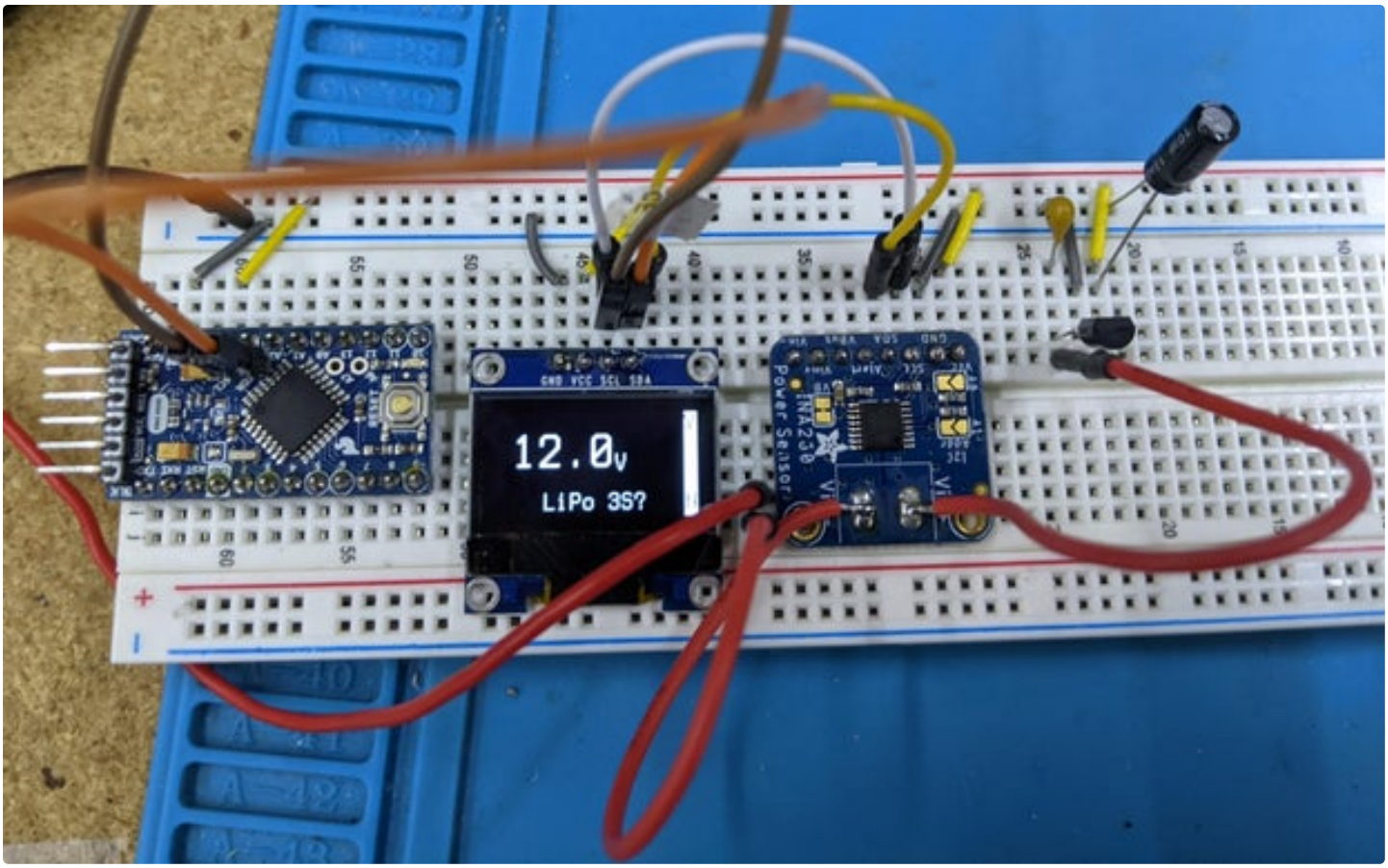
So far, the 3.3V has been provided by the FTDI programming board. For this step, we will install the voltage regulator so that we can run off arbitrary (e.g. 6-30V) input power. That allows us to see the INA260 working now and allows us to power real LEDs later.

Wiring errors here can burn up parts. One way to avoid this it to use a regulated power supply and set the maximum current low (20 mA should be enough for now). Another way is to put a small fuse in series with your battery. A third way is to verify that the 3.3V regulator is working as expected before connecting anything to it.

The basic wiring flow is as follows:

1. Battery + → V+ on the INA260
2. Battery - → GND
3. INA260 V- → Regulator Vin
4. Regulator Vout → 3.3V rail (which the pro mini, OLED and INA260 are all connected to)
5. Regulator GND → GND
6. Put the capacitors on the regulator as described by the datasheet.
    1. The LM2936 datasheet calls for Vin → 100nF → GND and Vout → 10uF → GND
    2. . I have confirmed with an oscilloscope that not using caps as suggested leads to unstable regulation with this part.

Your reward for getting everything correct is that the firmware will now show your voltage on startup. We still have no way to communicate with the unit but that will soon change.



Step 6: Add Buttons and the Potentiometer

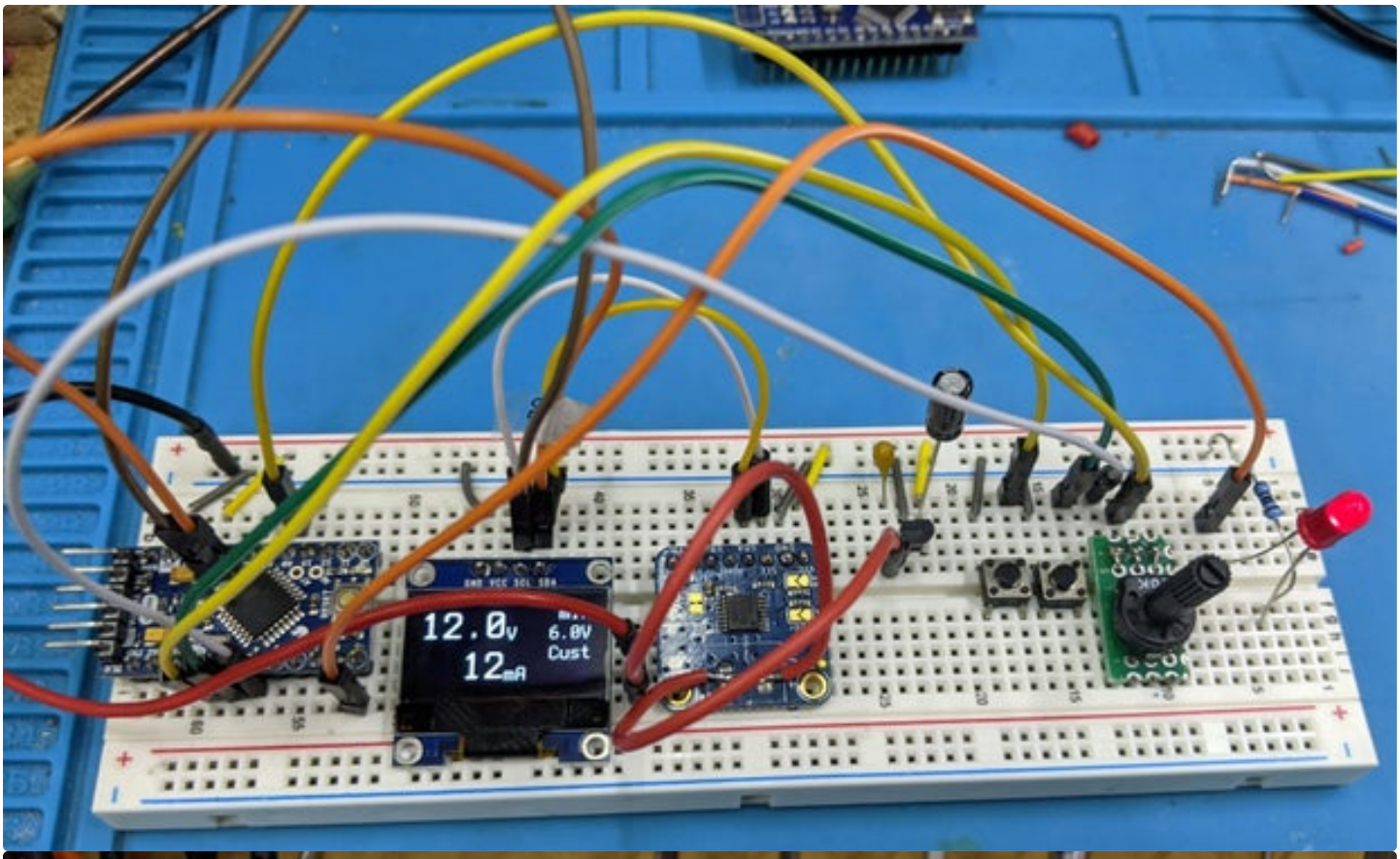
Here we add buttons and the potentiometer to the breadboard. In the next step, we build an input board. If you don't have good components for the breadboard, you can skip to the next step.

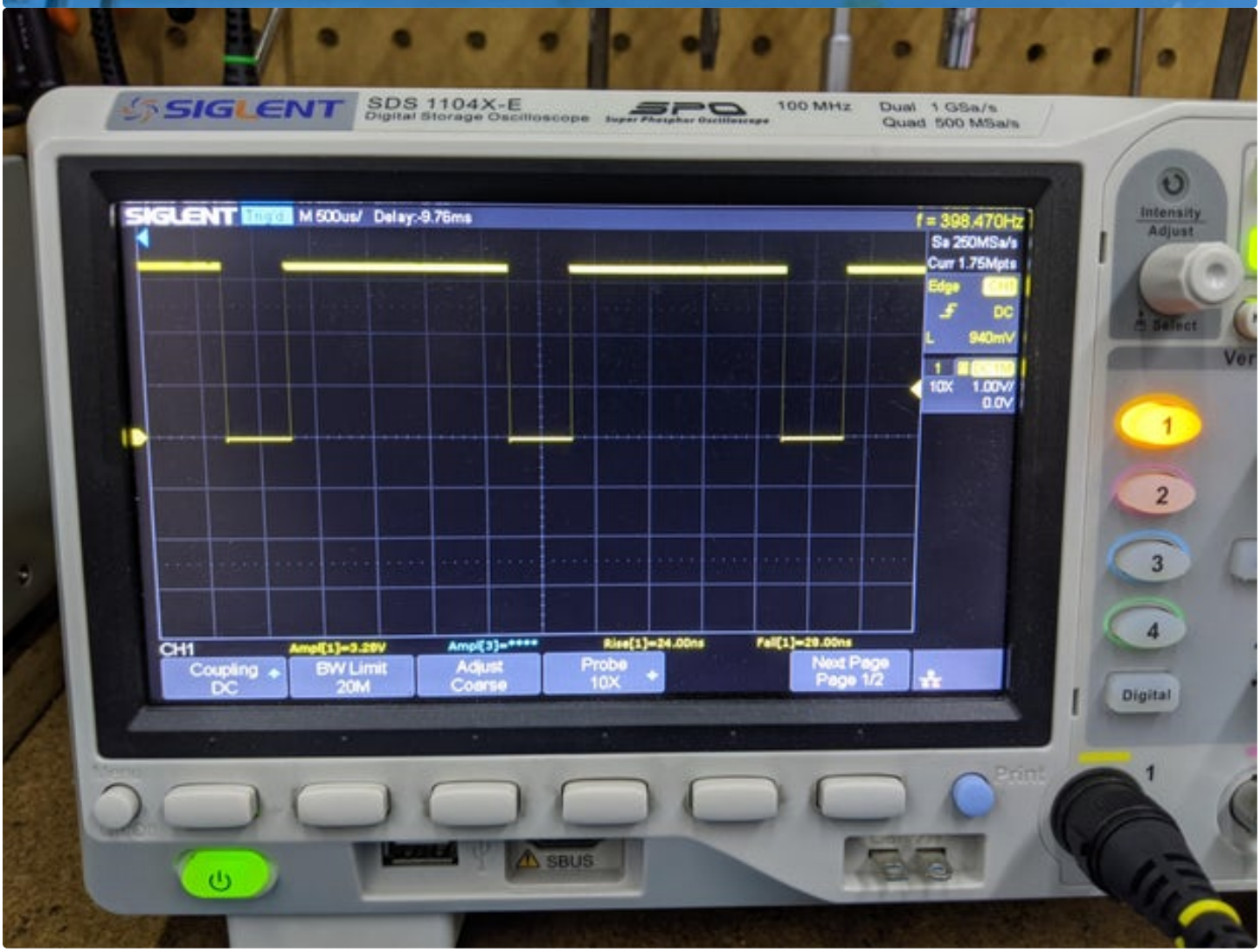
The buttons and potentiometer I used in this step are not the ones I'm using in the final design, just some from the parts box. Wiring is as follows:

- Pro Mini D2 → Red button (left side)
- Red button (right side) → GND
- Pro Mini D3 → Green button (left side)
- Green button (right side) → GND
- Potentiometer (left pin) → GND
- Pro Mini A0 → Potentiometer (center pin)
- Pro Mini D4 → Potentiometer (right pin)

Now you can control the unit. The unit will think it's controlling a real LED at this point and the controls will respond like the final design. In this design green is on, red is off and the potentiometer controls brightness.

Here I added a **temporary LED and resistor (1k)** between D9 and GND. D9 is the PWM output that will allow us to dim our LEDs. In the final design, D9 is just a signal to the Picobuck converters but here I use it to drive power through a small led for a direct demonstration. The oscilliscope signal shows the output of D9. As you turn the potentiometer to the right, the duty cycle of the square wave changes and is at 3.3V for more of each period. The buttons also change the state of the unit and turns the PWM on and off.


12.0V
6.0V
Cust
12mA

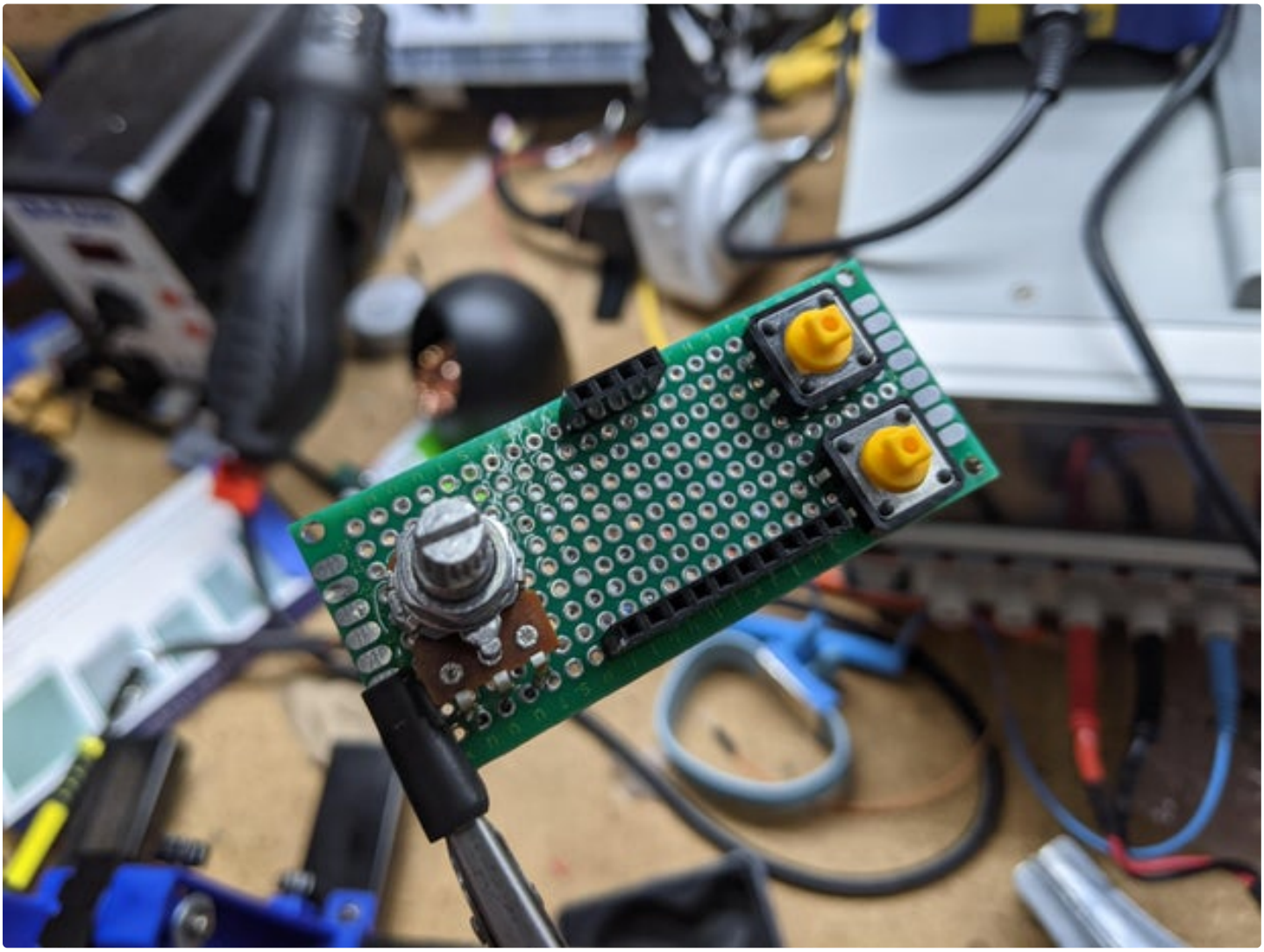

SIGLENT
SDS 1104X-E
Digital Storage Oscilloscope
100 MHz
Dual 1 GSa/s
Quad 500 MSa/s
M 500us/ Delay:-9.76ms
f = 398.470Hz
Sa 250MSa/s
Curr 1.75Mpts
Edge
DC
940mV
10X 1.00V/
0.0V
CH1
Coupling DC
BW Limit 20M
Adjust Coarse
Probe 10X
Next Page Page 1/2
Intensity Adjust
Select
1
2
3
4
Digital
SBUS

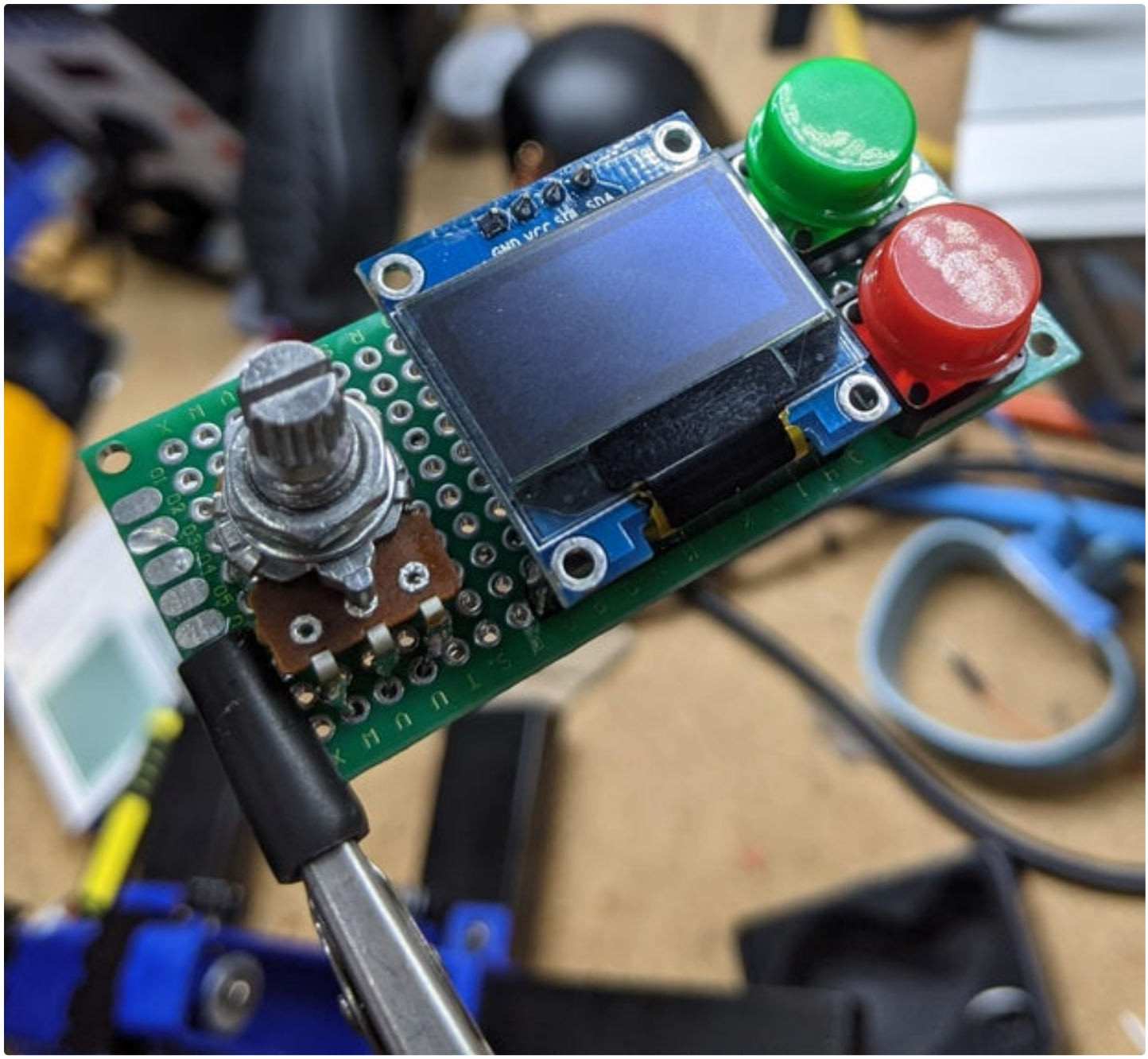
Step 7: Build the Input Board

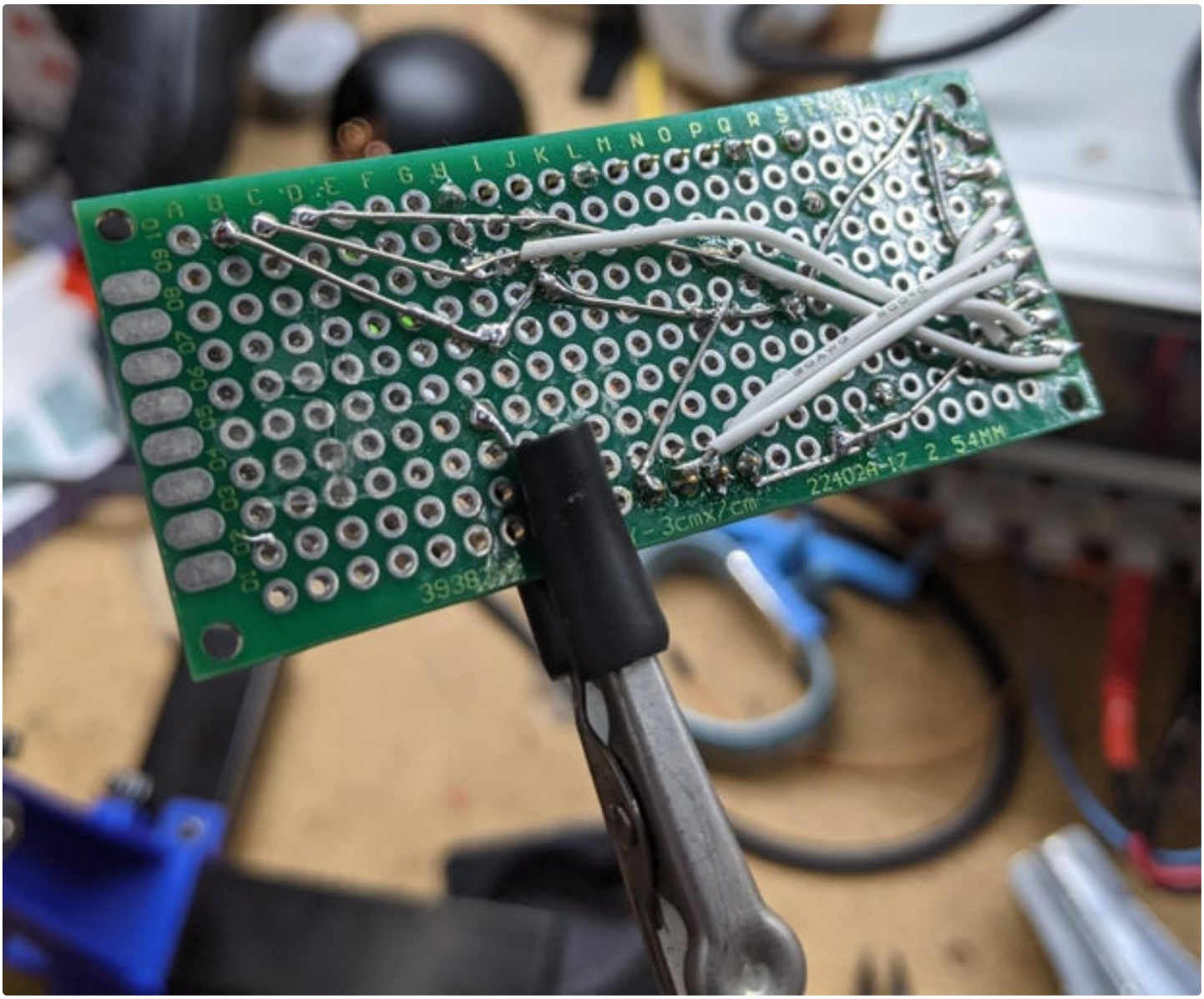
See the schematic in step 2 for the wiring. I used a 70x30mm prototype PCB for this. If you want the input board to fit nicely into the 3D print, pay close attention to the part placement. I suggest counting the PCB holes as a reference.

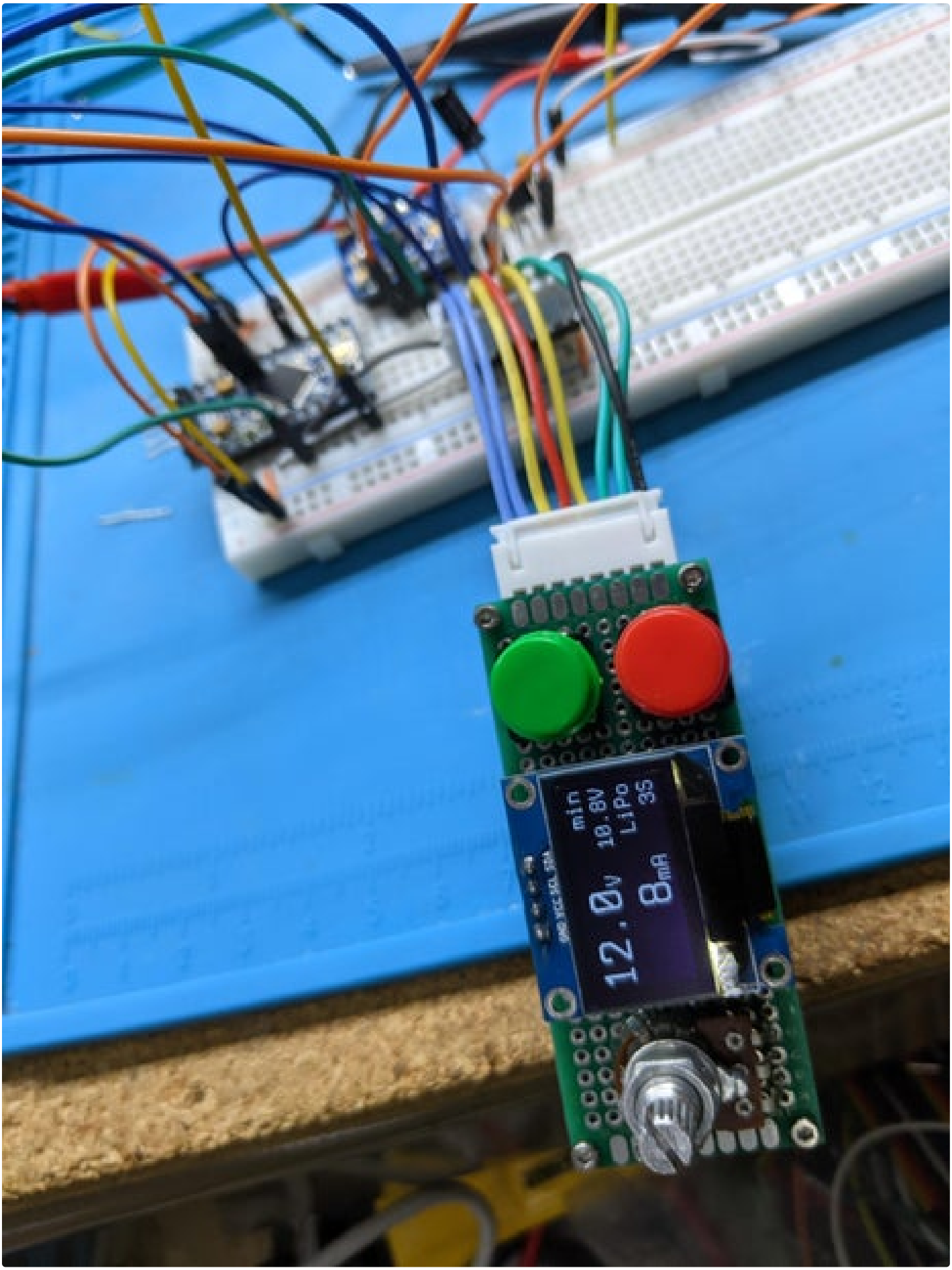
I glued the potentiometer to the board with epoxy.

To connect the input board to the main board (and breadboard), I went with a male 8 pin JST XH connector (2.54 mm pin pitch). You can go with whatever connector you want, including directly soldering on the wires.






A B C D E F G H I J K L M N O P Q R S
2.54MM

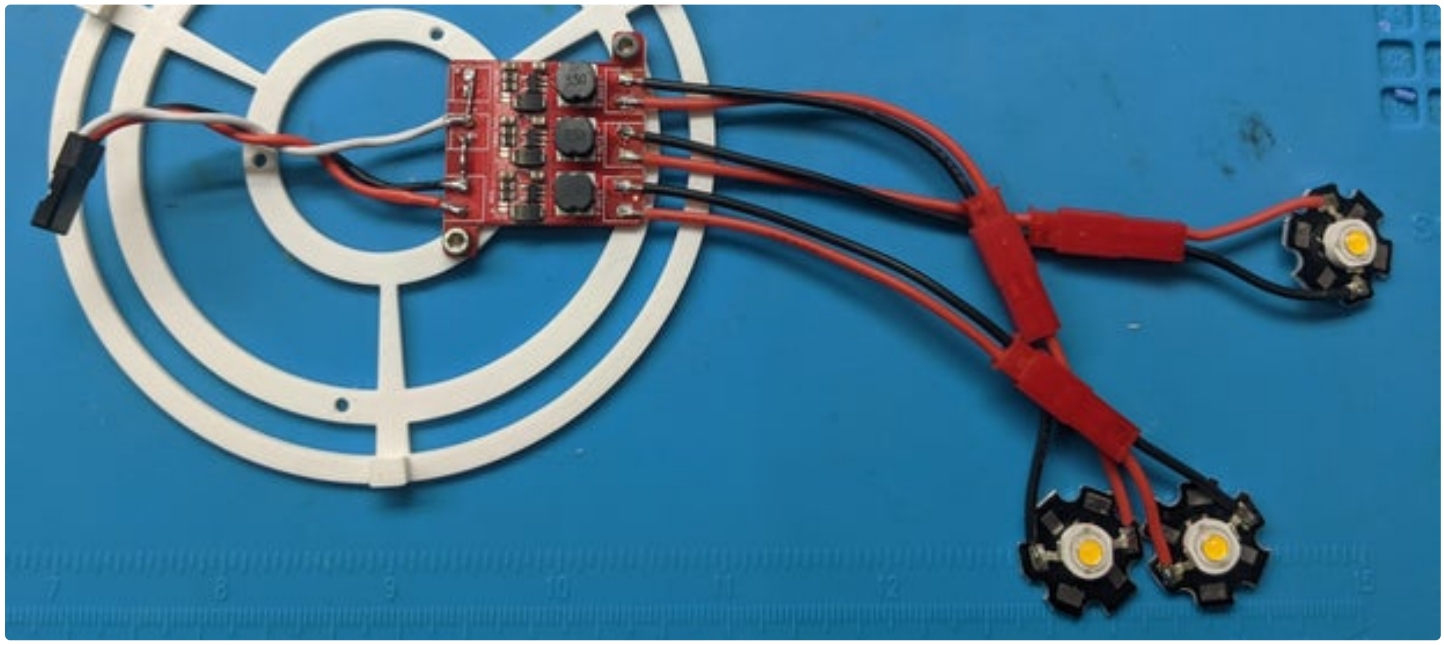

min
10.8V
LiPo
3S
12.0V
8mA

Step 8: Prepare a PicoBuck and 3 LEDs

The photo shows the PicoBuck on the 3D printed assembly but you do not need the assembly yet.

For the PicoBuck, you can short all of the PWM inputs together as shown in the photo. You can also short together the signal and LED GND on the board (assuming you go with the high-side PFET switch used in this guide. If you opt for a "low side" switch, you would not short them).

I used JST connections between the LEDs and picobuck. It's probably not necessary. As long as your wires are 3.5-4 inches (90-100mm), you should be able to use the 3D printed dome without the connectors.

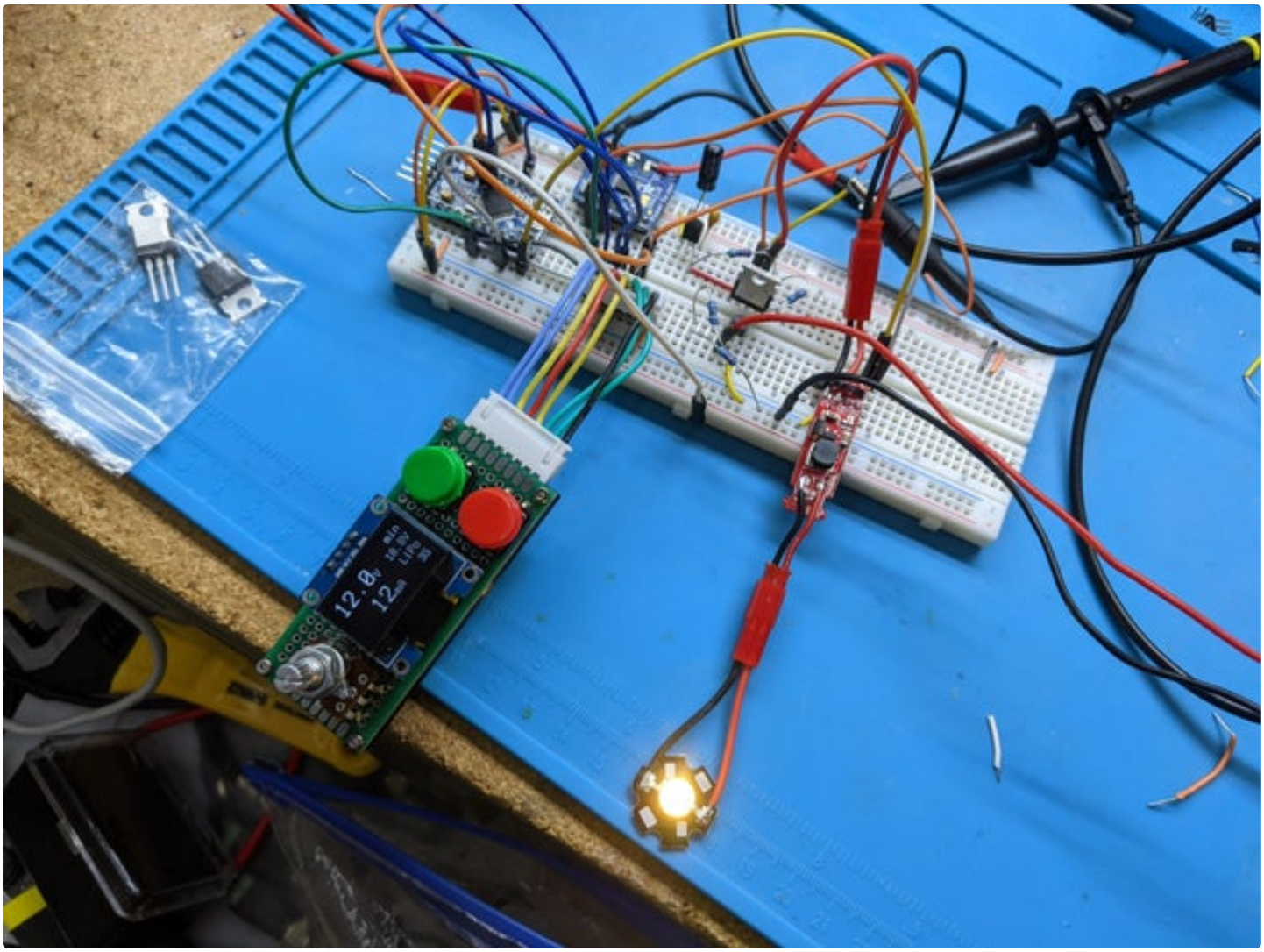


Step 9: Wire Up and Test the PFET Switch

The schematic is in step 1. You basically need a power PFET, any NFET and 4 resistors. The 47k pulldown from the PFET to the PicoBuck is optional - it will allow the voltage to drop to zero when the PFET is off.

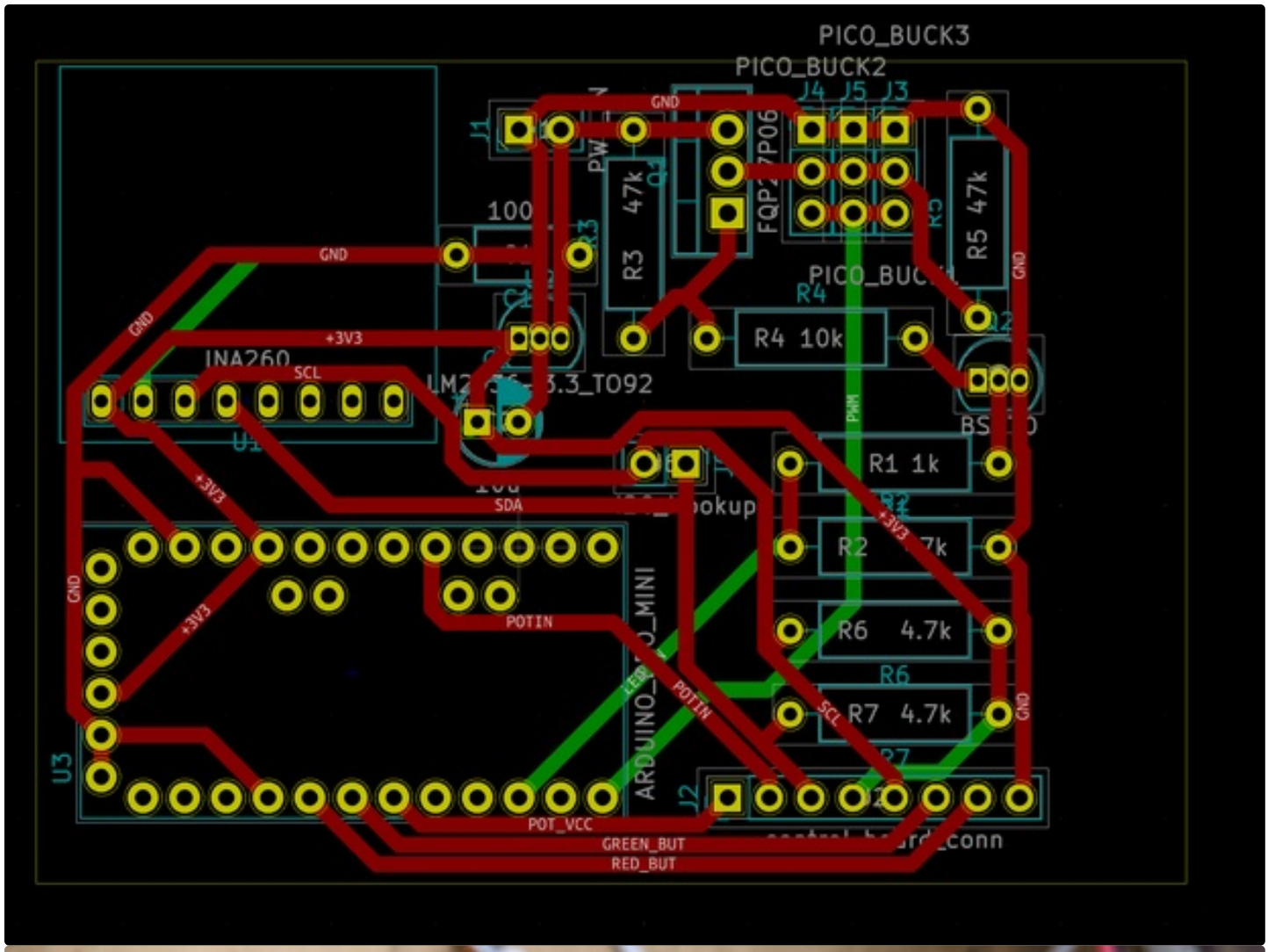
The photo shows a FemToBuck instead of a PicoBuck. They can be used interchangeably.

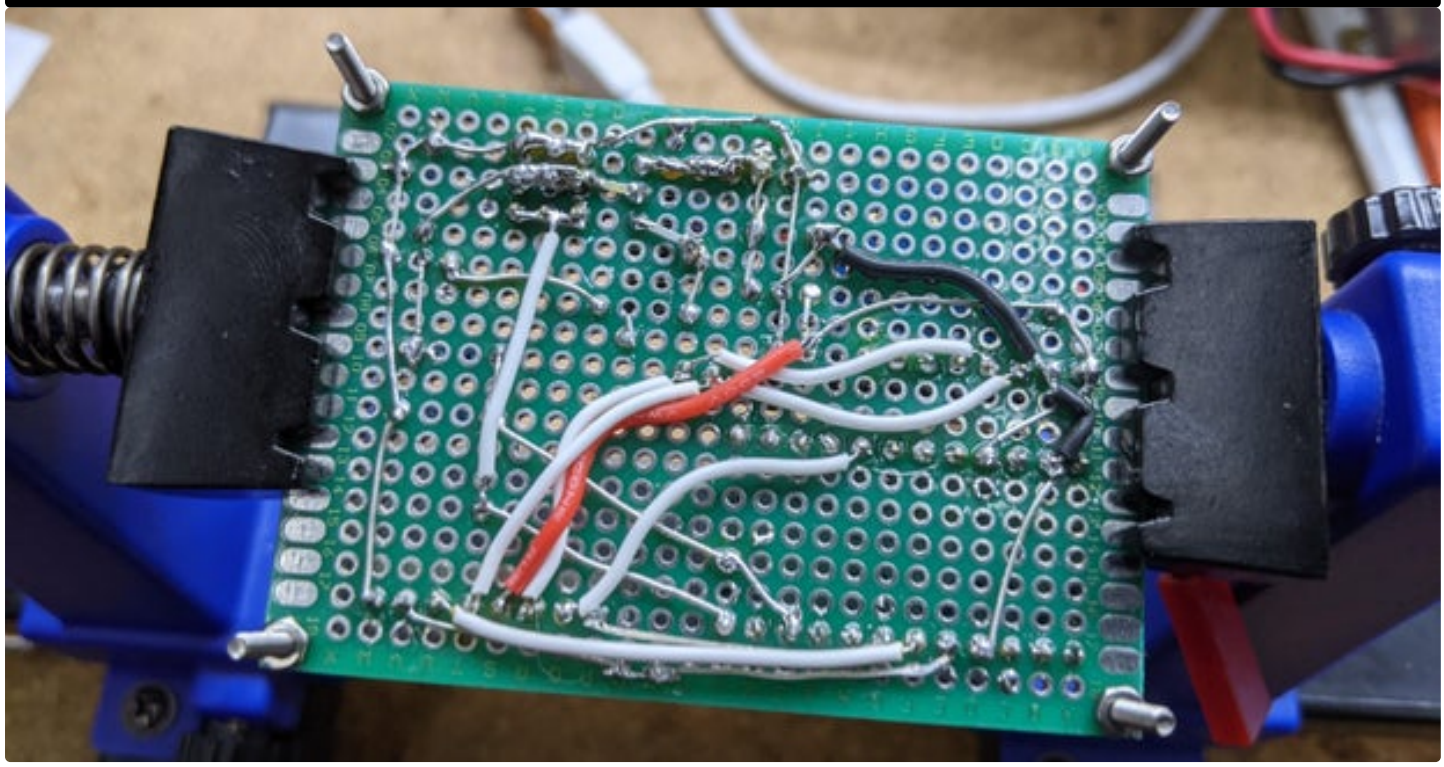
Note: you also have the option of omitting the PFET/NFET switch, plugging V- from the INA260 directly into the PicoBuck power input. Things will still work as the "off" position of the firmware also flatlines the PWM signal at D9. The downside is that each PicoBuck uses about 1 mA in this configuration for a total waste of 3mA, even when the unit appears off. Using the PFET avoids this 3 mA parasitic drain.

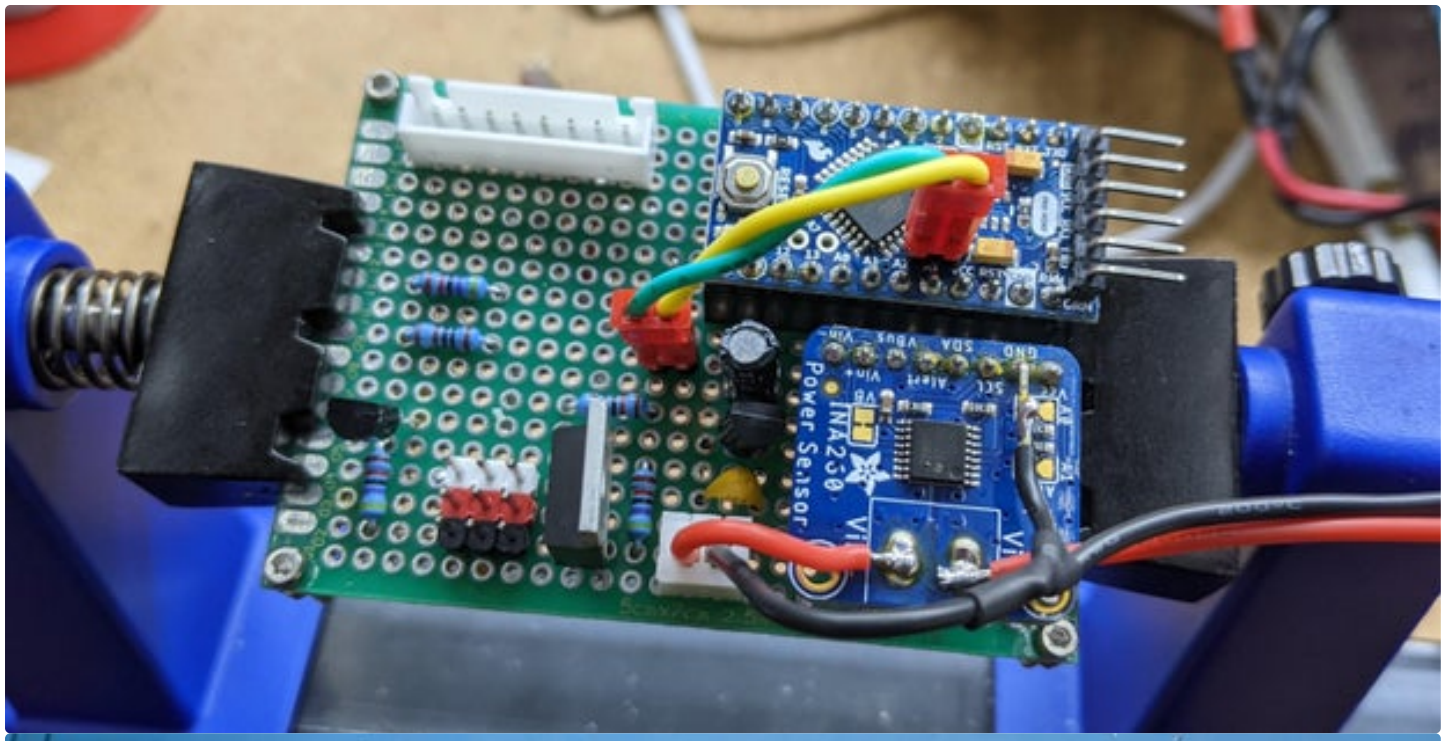


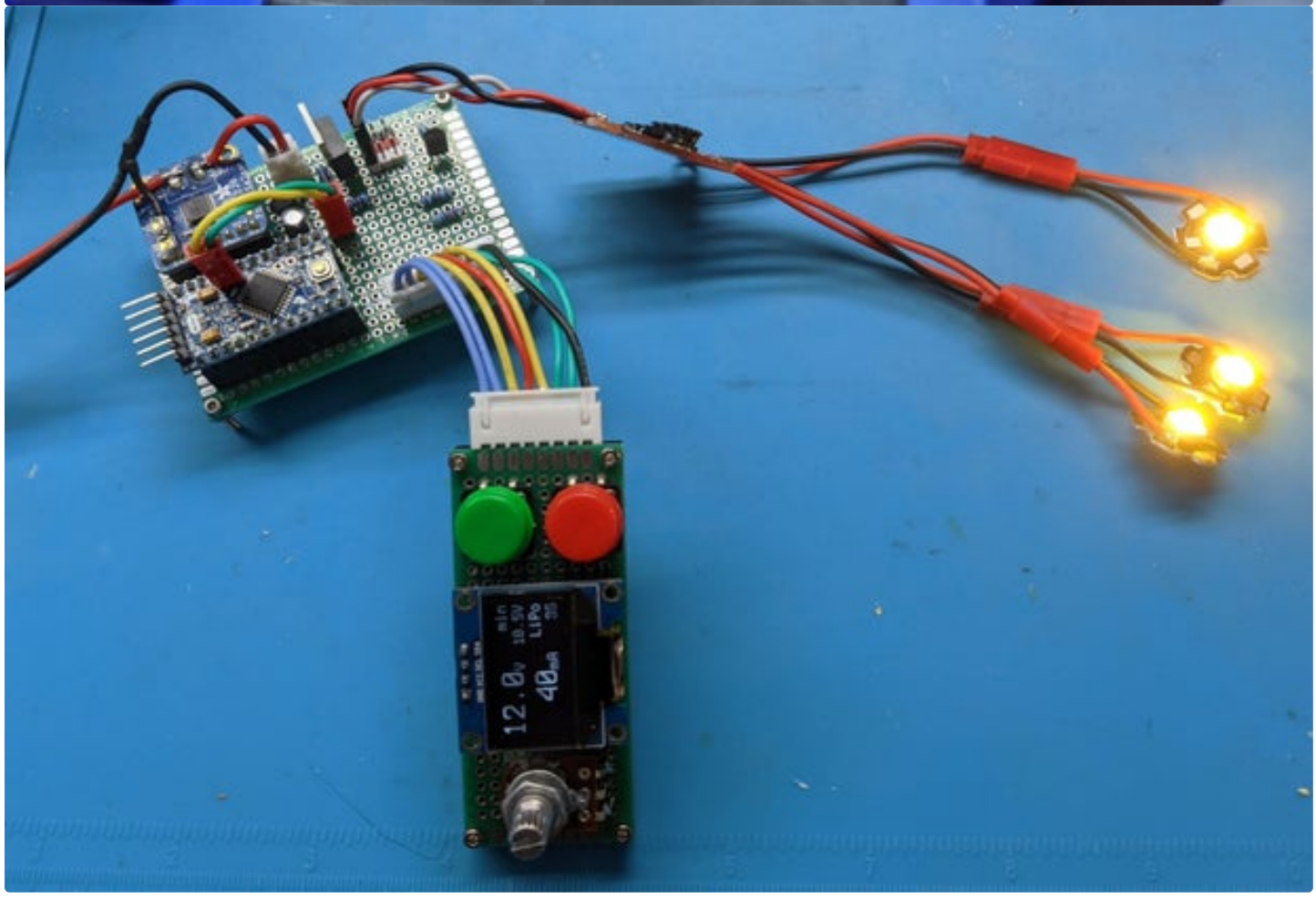
Step 10: Build Master PCB

Now we transfer the breadboard circuit to a PCB. I used a 50x70mm PCB which is also what the 3D print is designed to use. I used kicad to help layout the components (files here) before wiring up the board. Note that the kicad schematic contains 4.7k pullup resistors for SDA and SCL. The INA260 and OLED already have pullup resistors onboard and the 4.7k pullups were never needed in testing, so I did not include them in my actual wiring.


PICO_BUCK3
PICO_BUCK2
J4 J5 J3
GND
J1
R3 47k
R5 47k
R4
R4 10k
GND
+3V3
INA260
SCL
SDA
PWM
R1 1k
R2
R6 4.7k
R7 4.7k
POTIN
POT_VCC
GREEN_BUT
RED_BUT
U1
U3
J2







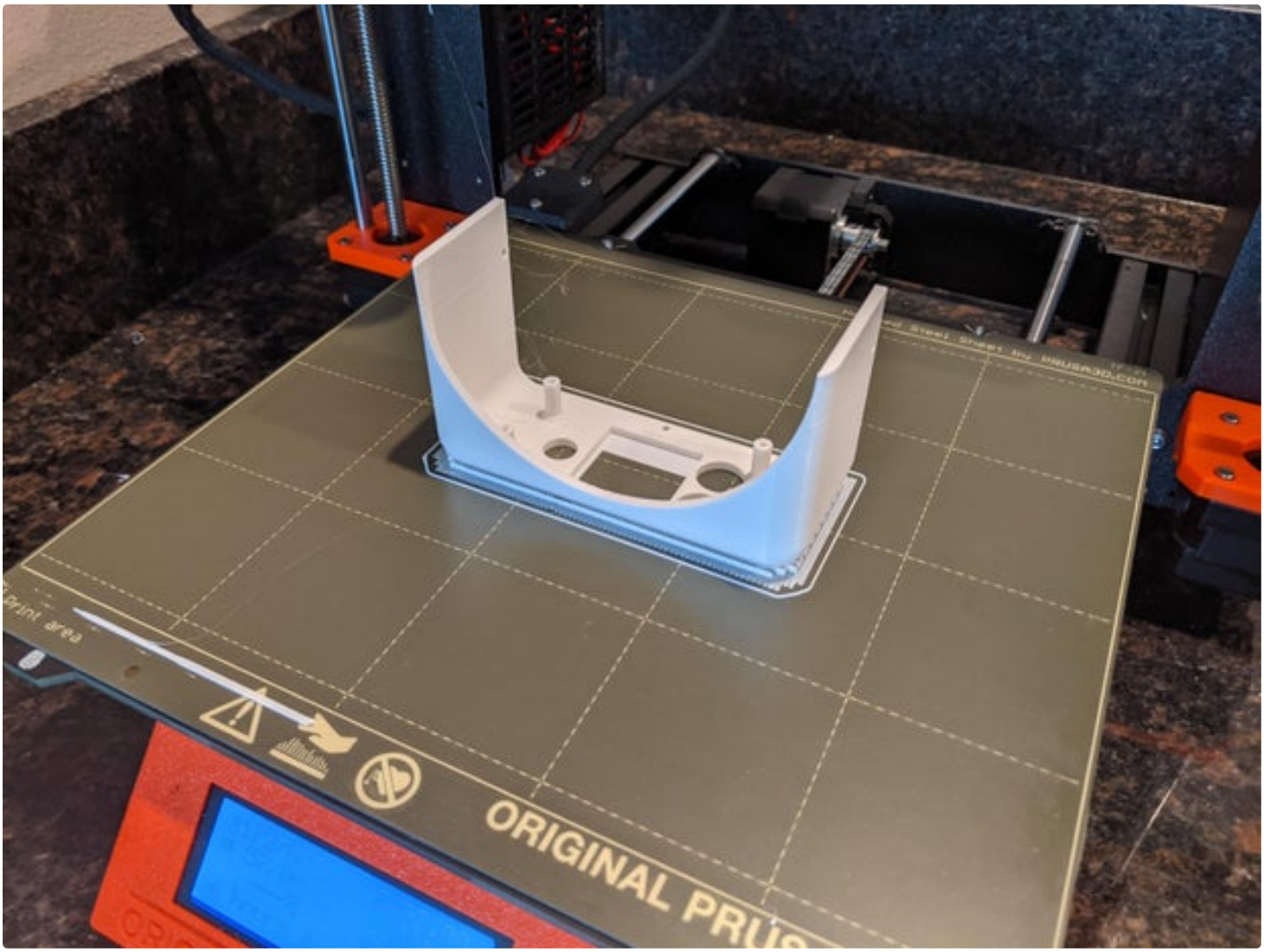
Step 11: 3D Print and Assemble Front and Base Plate

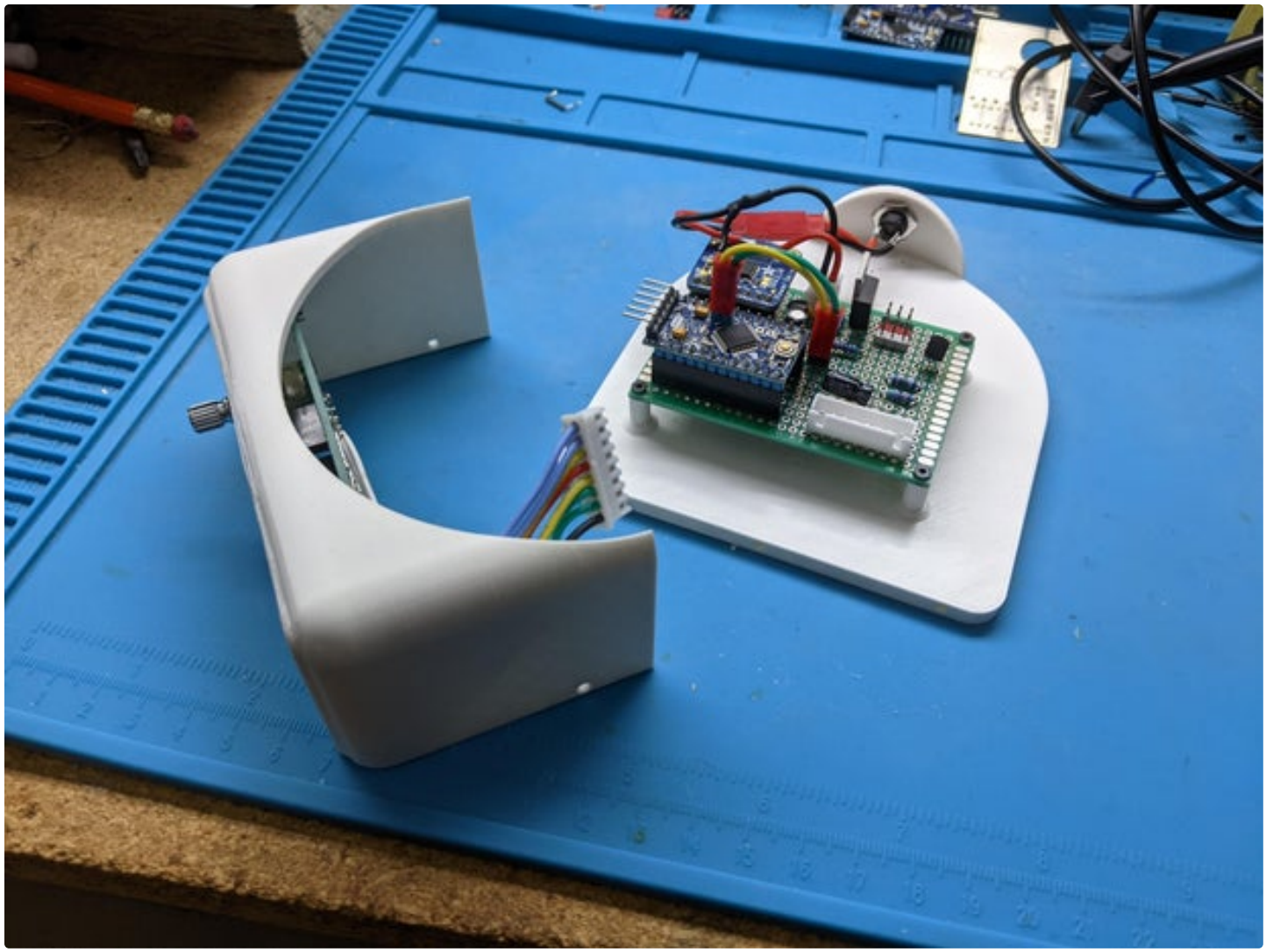
My OpenSCAD design files are here, but you can just use the .stl files below if you don't need any design changes.

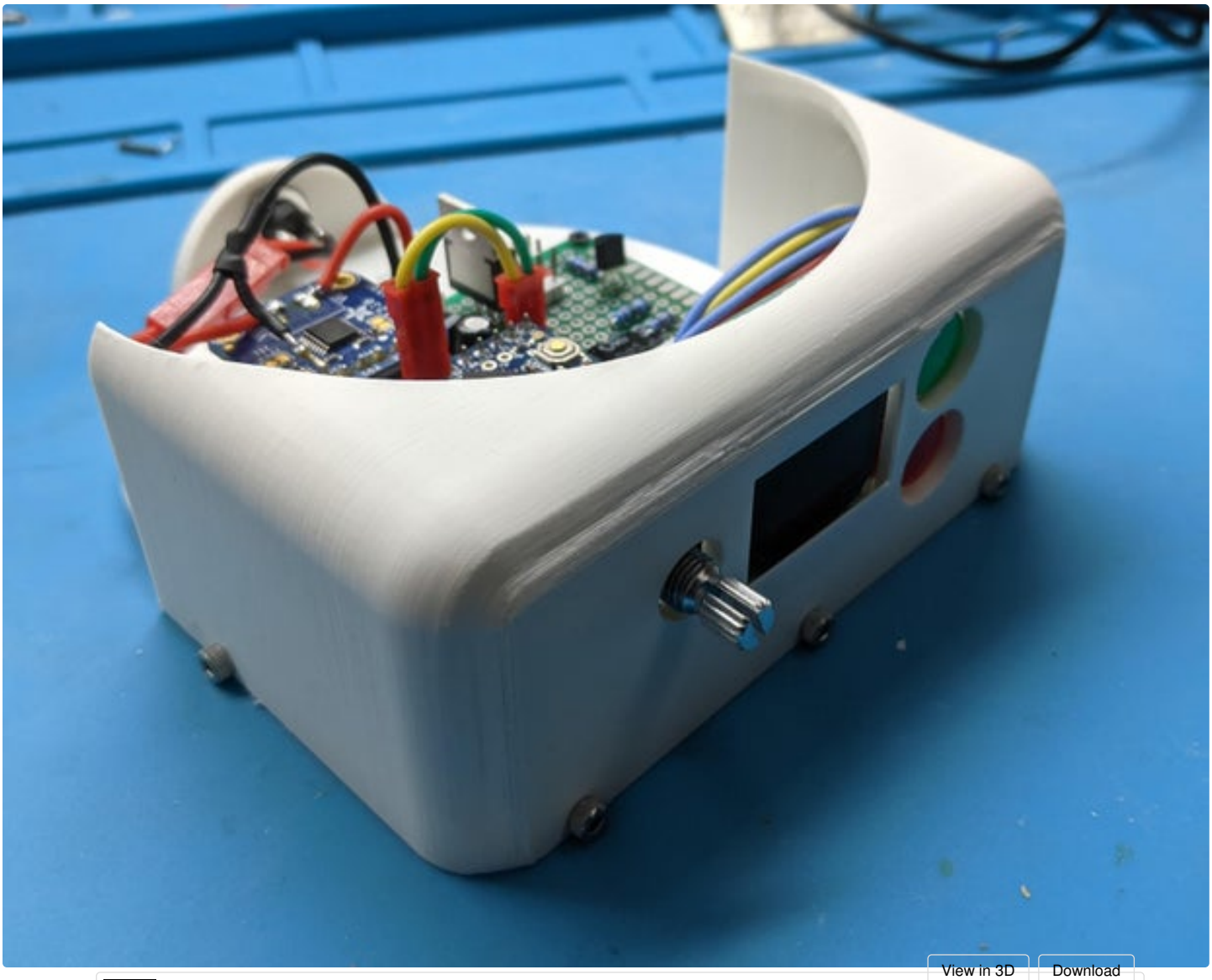
I used 0.15mm layer height in PLA. A 0.2mm height could also be used. PETG or other filaments can be used in place of PLA. I printed the front face *with supports* for the curved surfaces - they might not be necessary.

There is a hex nut hole in the middle of the base. It can be fitted with a 1/4-20 hex nut which will then allow the lamp to be attached to a standard tripod mount.

The front attaches to the base with M2.5 bolts. Anything between 6mm and 16mm should work. Be careful not to over-tighten the bolts or they will strip plastic.


ORIGINAL PRU





View in 3D Download

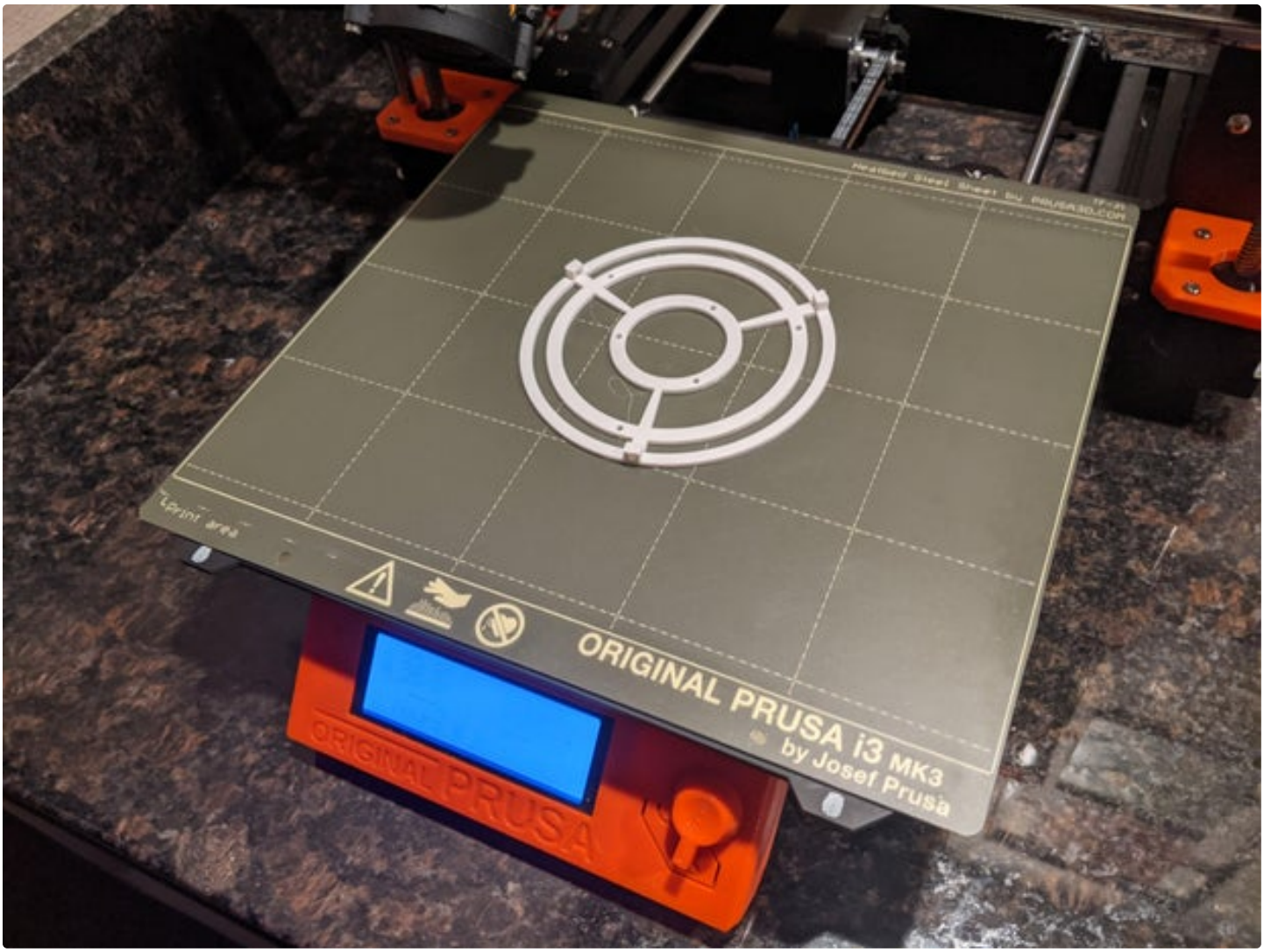
https://www.instructables.com/ORIG/FBI/O692/KST8YWEX/FBIO692KST8YWEX.stl

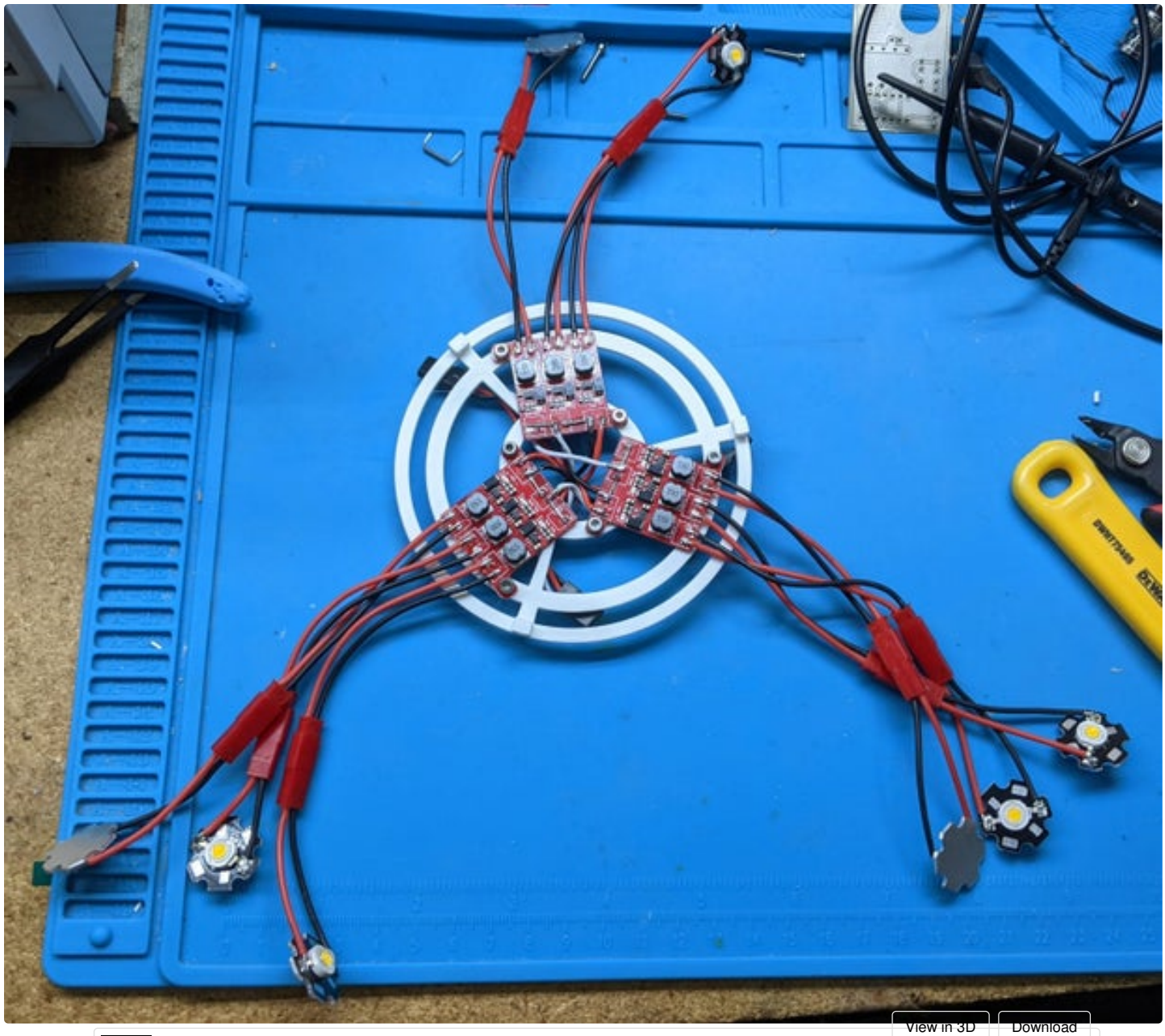
View in 3D Download

https://www.instructables.com/ORIG/FXQ/J1PD/KST8YWEY/FXQJ1PDKST8YWEY.stl

Step 12: 3D Print the Picobuck Mount and Attach the Picobucks

The PicoBuck drivers attach with M2.5 bolts. Anything over 6mm should be ok. The bolts do not need to be very tight and tightening them too much will strip plastic.


ORIGINAL PRUSA i3 MK3
by Josef Prusa
ORIGINAL PRUSA



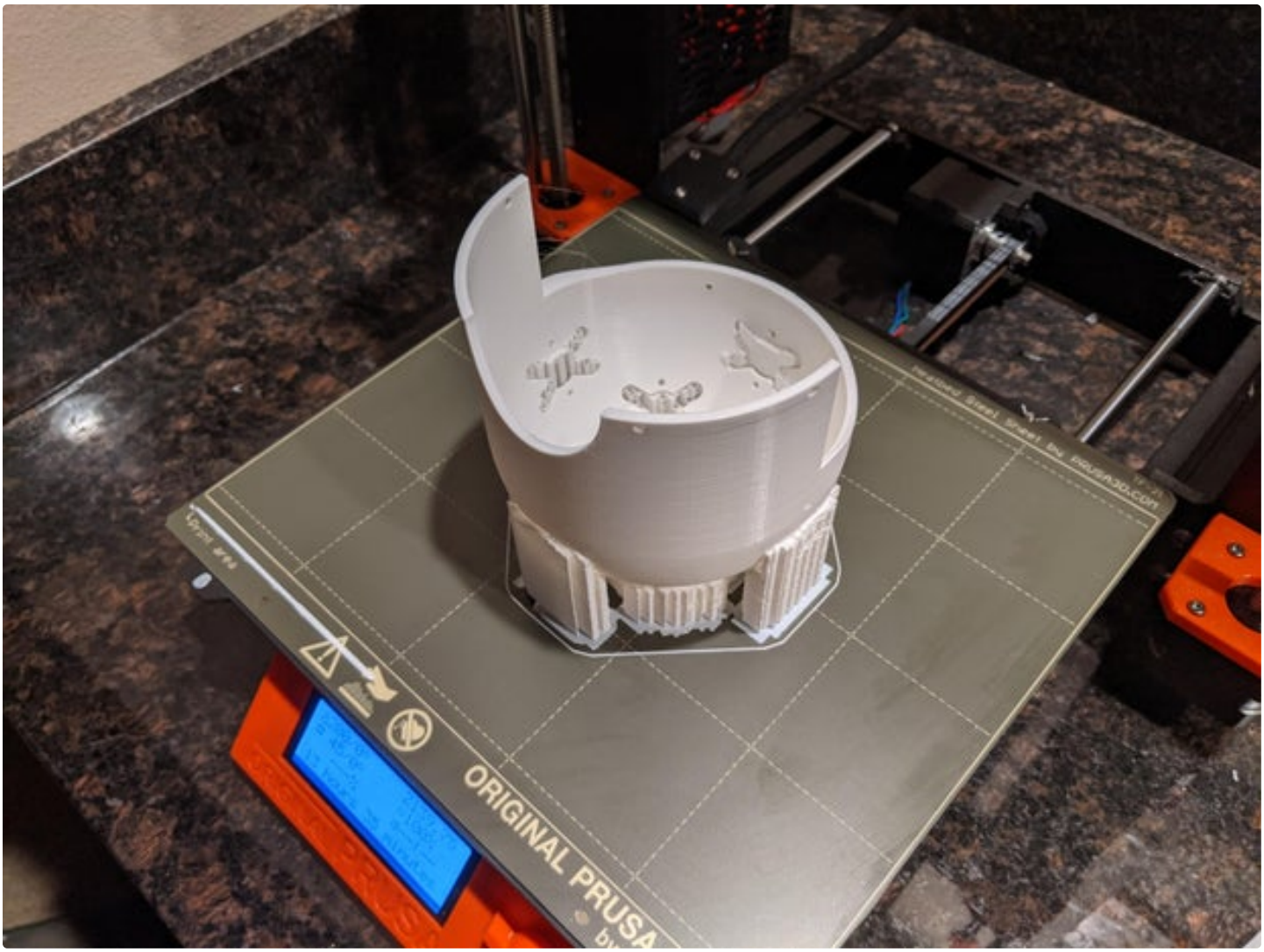
View in 3D Download

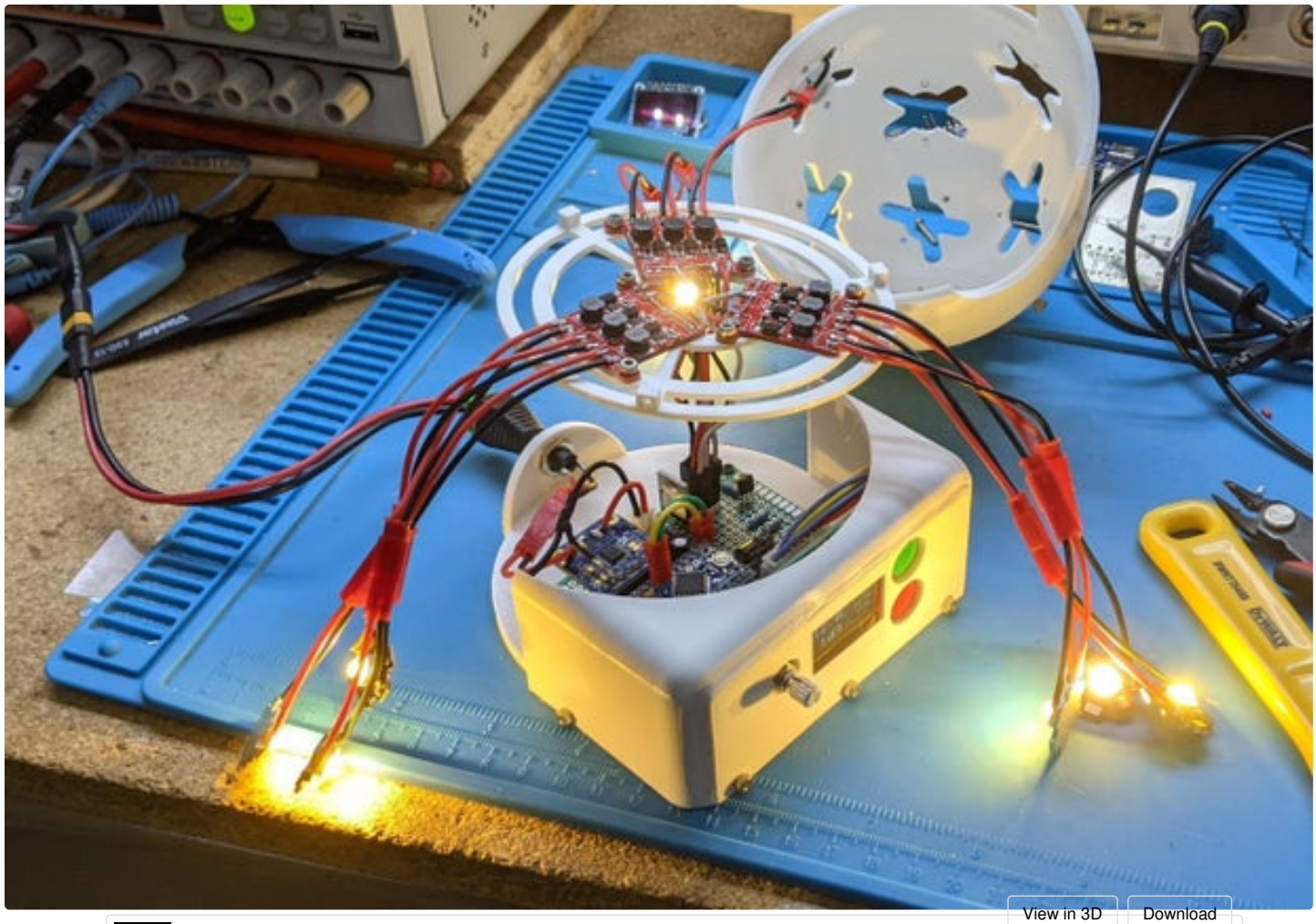
https://www.instructables.com/ORIG/FA2/G5UJ/KST8YX16/FA2G5UJKST8YX16.stl

Step 13: Print Led Dome and Do Final Pretest

I used both a raft and support for this print - the raft is helpful because the top of the dome has small details which do not make for a good first layer.

Before continuing with final assembly, plug the PicoBucks into the PWM ports on the main board and assert that the light works as expected


ORIGINAL PRUSA



View in 3D Download

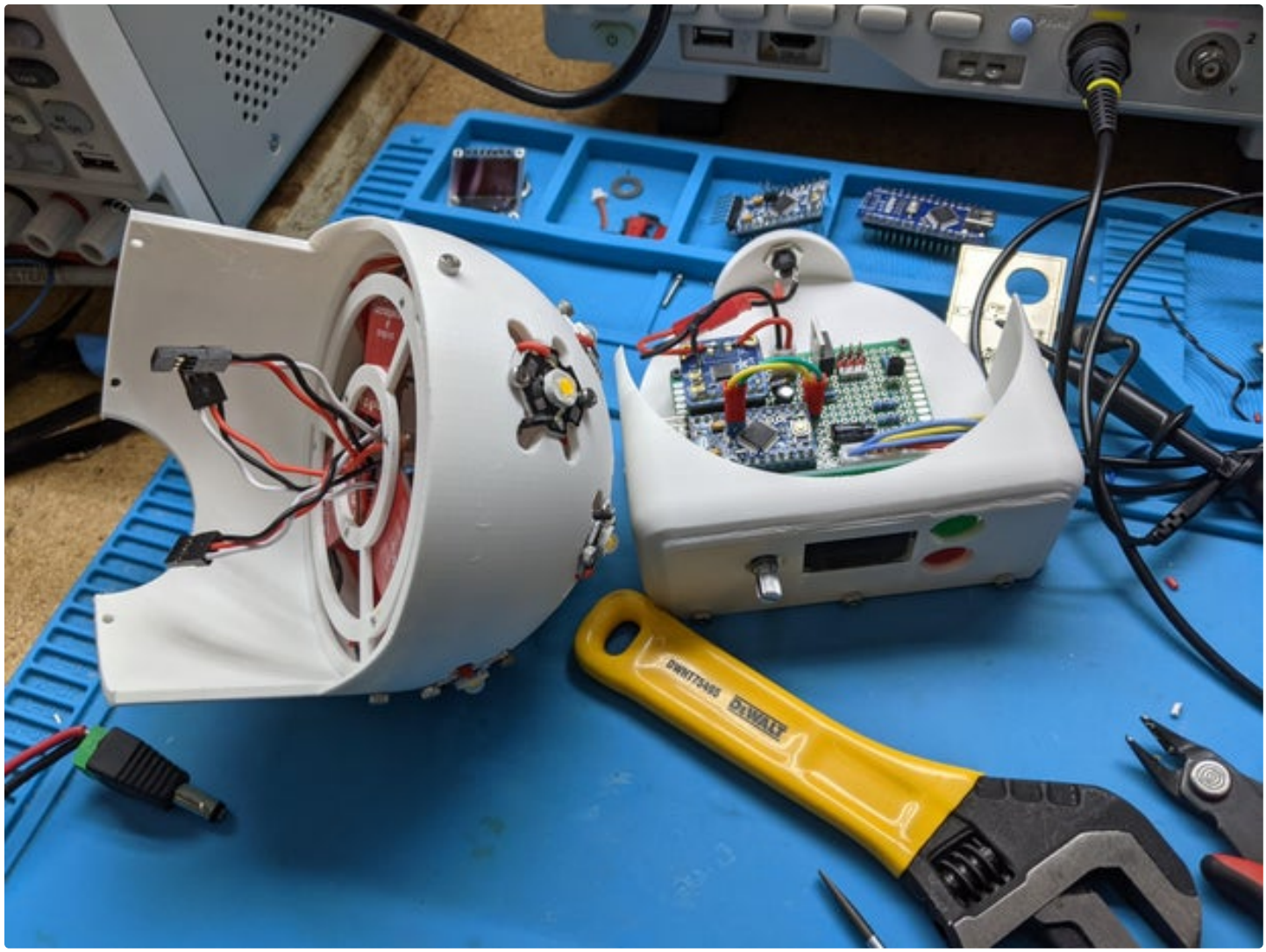
https://www.instructables.com/ORIG/F20/A32K/KT37UYTM/F20A32KKT37UYTM.stl

Step 14: Final Assembly and Notes

Attach the dome to the base with M2.5 bolts that are between 6mm and 16mm long. Attached below is a 3D printable dimmer knob that you can push onto the potentiometer to finalize the build.

Some notes for future designs:

- The unit could be modularized into two components. One provides power, PWM and monitoring. The other is just the needed buck (or boost) converters and LEDs. This would allow for different LED modules, such as a flat/directional one. Or LEDs with different color temperatures
- Other sensors could be added, such as a photo resistor or motion detector.


DEWALT

View in 3D Download

https://www.instructables.com/ORIG/FGM/7F11/KST8YXNU/FGM7F11KST8YXNU.stl