Tutorials

- How To Use The Apollo GPIO Header To Control An LED Strip
- How to edit your Sensor's LUX update interval

How To Use The Apollo GPIO Header To Control An LED Strip

This tutorial will guide you through setting up one of our MSR-2 devices (works with any mezzanine port on any Apollo Device) with the optional $4.99 GPIO Header which adds pins for you to easily add functionality to your device! In this tutorial, however, we will be focusing on adding an LED strip to your Apollo device.

Materials Needed for tutorial:

- Apollo MSR-2, Apollo MTR-1, Apollo Air-1 and all other future Apollo Automation products with the mezzanine port.
- Apollo GPIO Header
- ws2812b aka neopixel RGB led strip or similar. sk6812 RGBW strip will also work.
- Optional DuPont Cables for GPIO Header but any DuPont cables will do.
- USB-C cable and power brick to power MSR-2

You are limited to 300mA of power output from the 5v port. You can either attach an external power supply and power the MSR-2 via 5v and gnd pins or work with the limited power output of the port

GPIO Pinout

Above is an image of the GPIO Header and its pinouts. We can use ports 2,4,6,7 for our data channel to an LED strip or multiple LED strips. We will also use the top two ports which are ground and 5v for power.

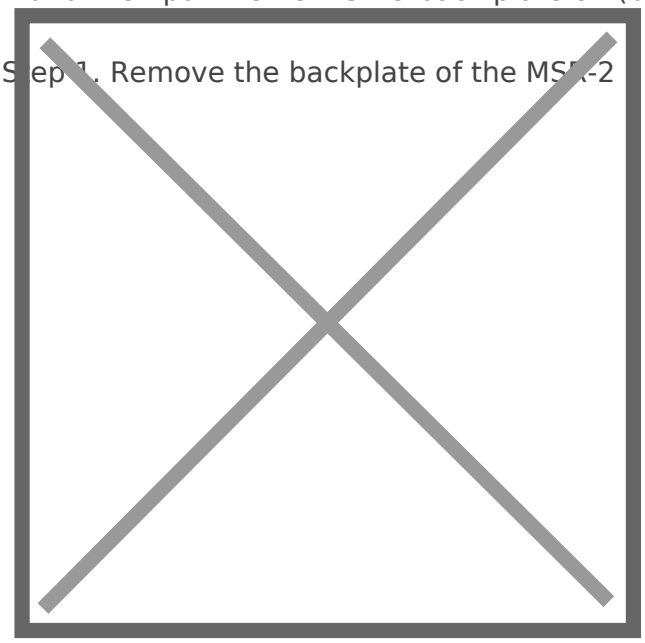
Did you know you can power the esp32 from the 5v and gnd pin? That means you can connect an external power supply and power it without the side USB port being used! This also allows for more power to be given to your LEDs!

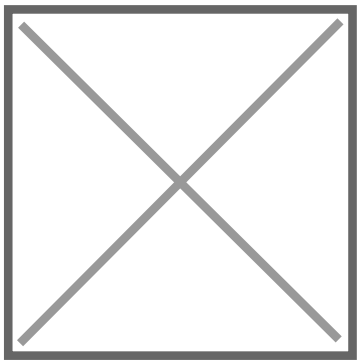
We cannot use the IO ports 0,1,18, or 19 for LEDs but you can use ports 0 and 1 for i2c sensors.

Connecting the GPIO Header to the MSR-2

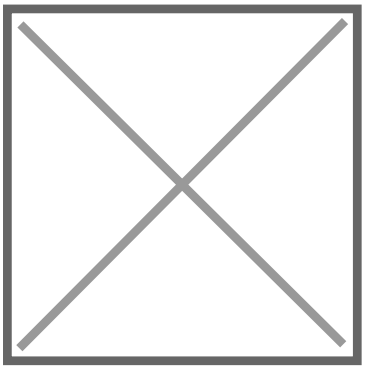
The first thing we will do is remove our MSR-2 back plate and connect our GPIO Header to our MSR-2 and then put the new GPIO back plate on (blue).

Step 1. Remove the backplate of the MSR-2

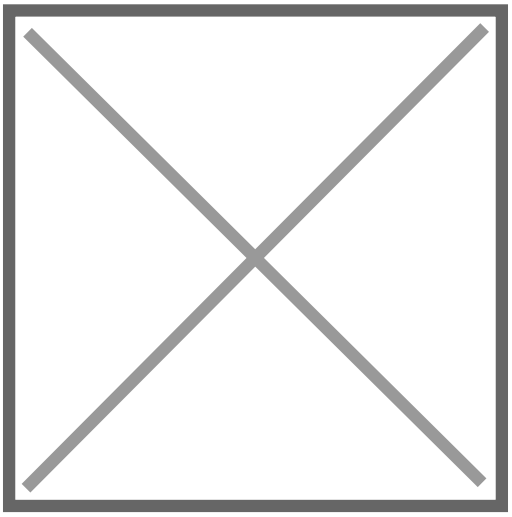




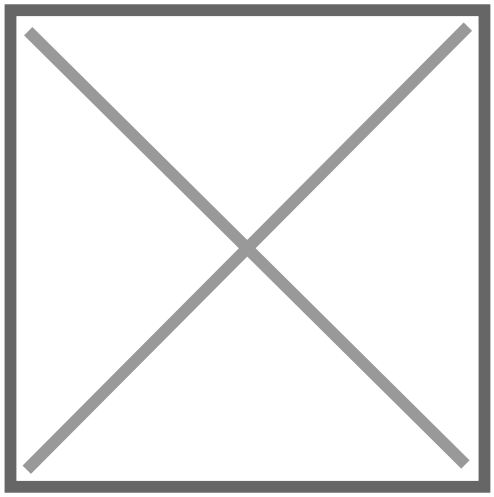
Step 2. Line up the Xs shown on the msr-2 and the GPIO Header. They should both be facing in the same direction as shown below.



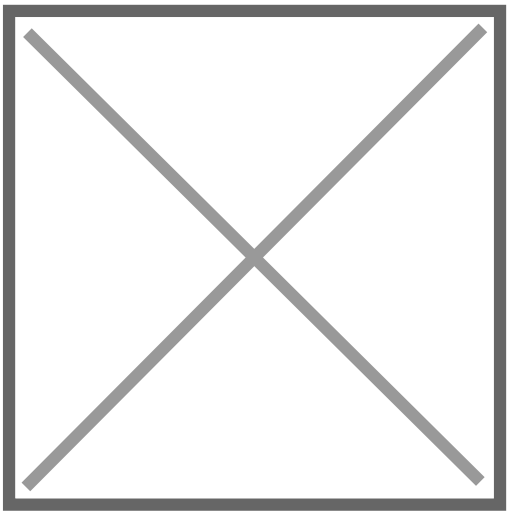
Step 3. Gently push down onto the GPIO Header as shown below:



Step 4. Confirm the GPIO Header is seated properly as shown below.



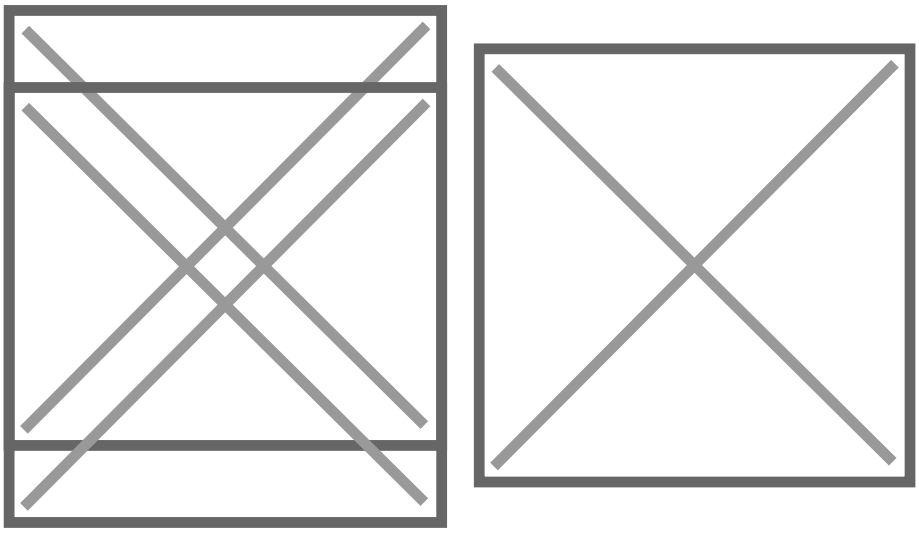
Step 5. Slide the GPIO Header back plate for the MSR-2 over your sensor and gently push down until it clicks into place.



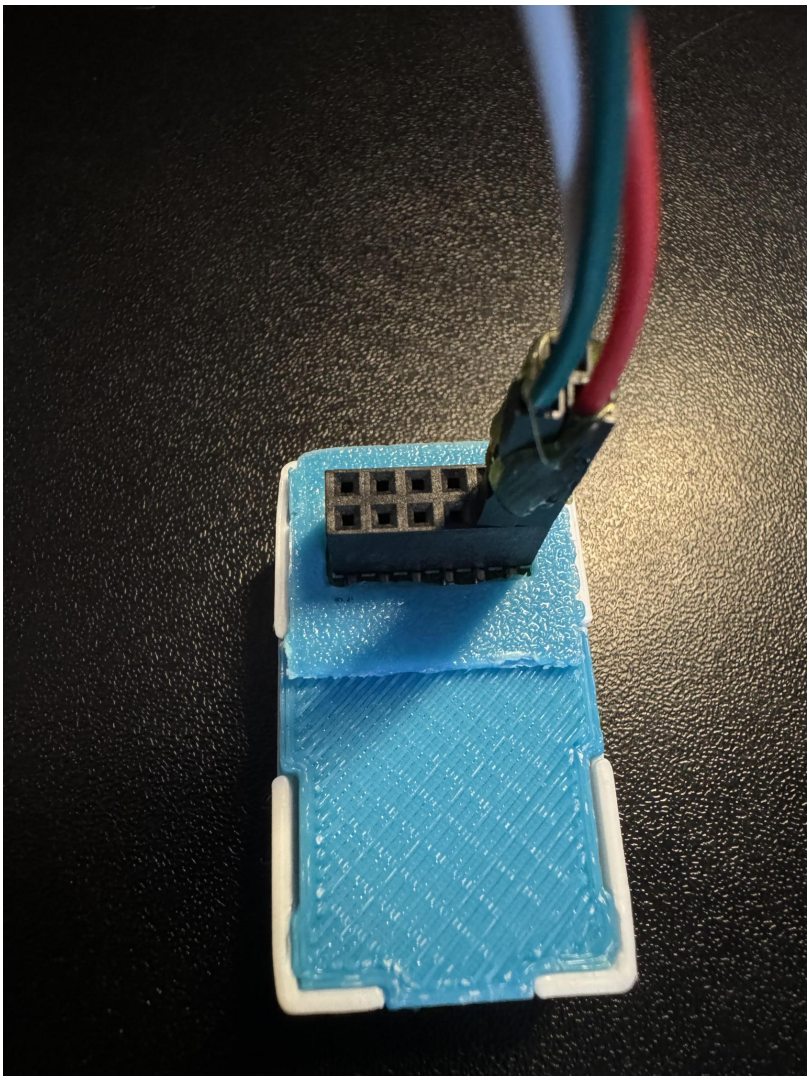
If the back plate does not gently go onto the sensor please investigate and confirm it is in the right orientation.

Connecting DuPont pins to proper GPIO ports

Now we need to reference the GPIO pinout we looked at above and then connect three wires. You will need three male-to-male DuPont wires included in your kit. I suggest using red for power aka 5v, White for ground aka GND, and green for data aka port IO7. Most LED strips will also have this same color scheme and it's easier to match like colors together.

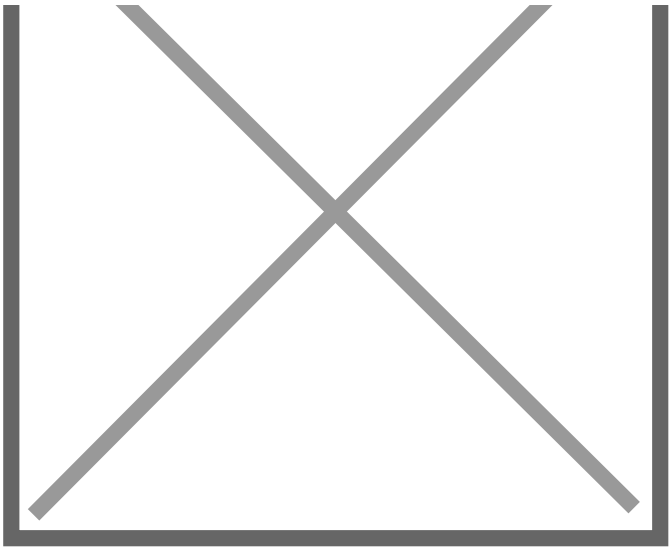


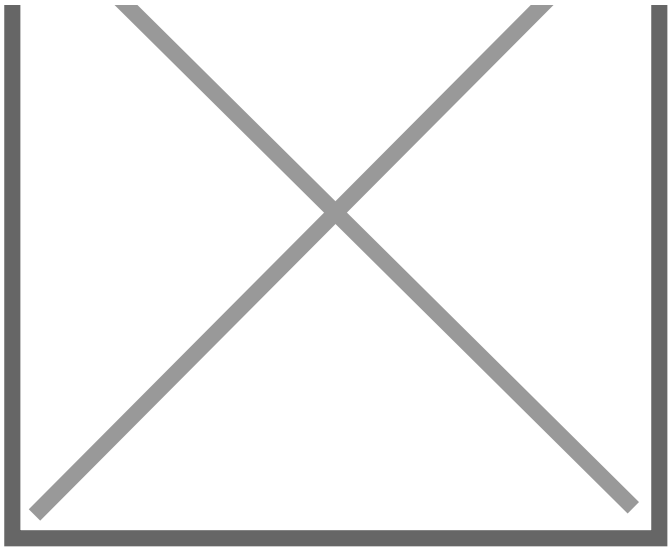
You can add a bit of hot glue to the Dupont wires to hold them together. DO NOT put hot glue into the GPIO Header's female pins that will ruin the addon. I am only suggesting that you can hot-glue the Dupont pins outer shell themselves together to stiffen them up.



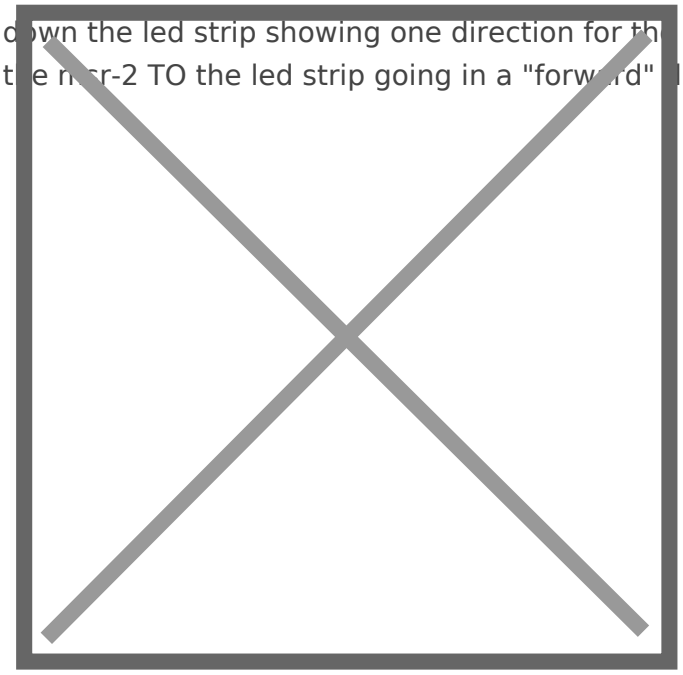
Connecting DuPont pins to LED Strip

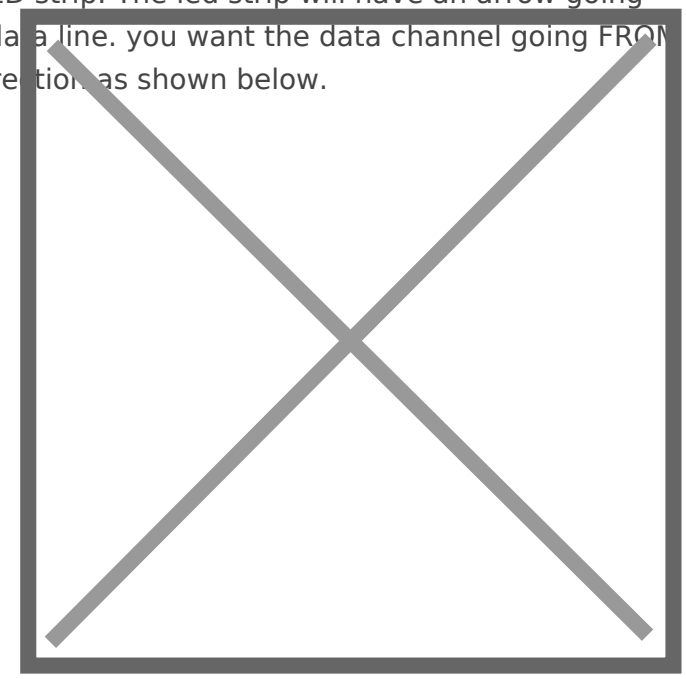
Next, we need to connect the other side of the Dupont pins to the LED strip. Most likely your LED strip will have a JST-SM connector which is a 3amp max connector with three wires connected: red for 5v, green for data, and white for gnd. We will be matching up our red, green, and white wires already attached to the GPIO add-on pins in the MSR-2 (using IO7 as the data pin for this tutorial)





Make sure to connect to the correct side of the LED strip. The led strip will have an arrow going down the led strip showing one direction for the data line. you want the data channel going FROM the msr-2 TO the led strip going in a "forward" direction as shown below.





Edit the YAML of your MSR-2 to let it know about your new LED strip

Finally, we need to tell the MSR-2 that we connected an LED strip. We need to tell it how many LEDs we have and we need to tell it that it's our second LED since the built-in LED is the first. This tutorial assumes you are comfortable with the ESPHome dashboard.

Step 1. Open ESPhome Dashboard and click edit to bring up the yaml your sensor is currently using.

You will see some YAML code here and you do NOT want to touch anything above line 20. If you need to, click your cursor at the end of wifi_password and hit enter to create a new line then make sure you backspace until you are "flush" with the line numbers like how wifi: is.

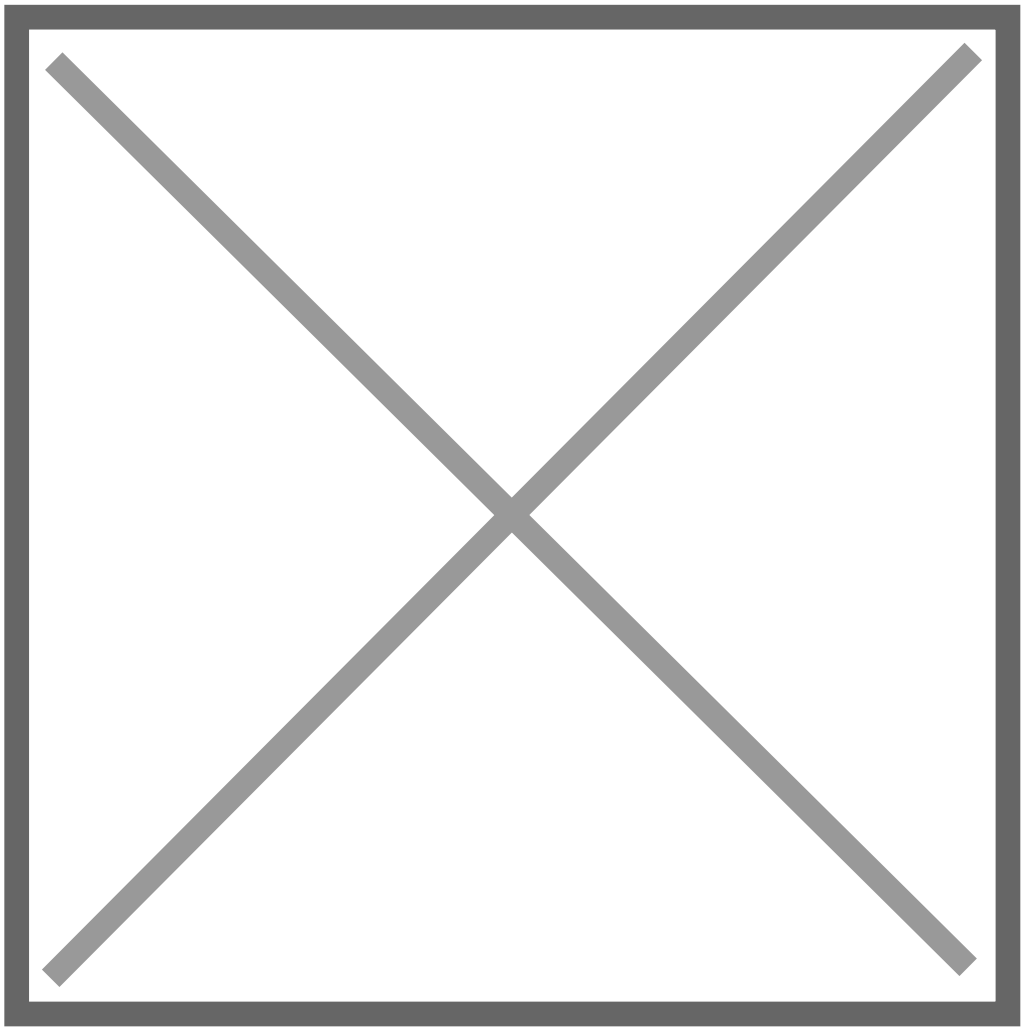
Step 2. Copy the code below and paste it to line 20 in your ESPHome yaml for this device.

```
light:
  - platform: esp32_rmt_led_strip
    id: bed_led
    name: "Bed LED"
    pin: GPIO7
    rmt_channel: 1
    default_transition_length: 0s
    chipset: WS2812
    num_leds: 60
    rgb_order: grb
    effects:
      - pulse:
          name: "Slow Pulse"
          transition_length: 1000ms
          update_interval: 1000ms
          min_brightness: 50%
          max_brightness: 100%
      - pulse:
          name: "Fast Pulse"
          transition_length: 100ms
          update_interval: 100ms
          min_brightness: 50%
          max_brightness: 100%
      - addressable_rainbow:
```

This is where you can change your number of LEDs as well as the GPIO pin used for the LED data!

Make sure to check out https://esphome.io/components/light/index.html#light-effects for all the effects supported such as addressable scan effect!

Step 3. Confirm you do not have any red lines showing errors in your code

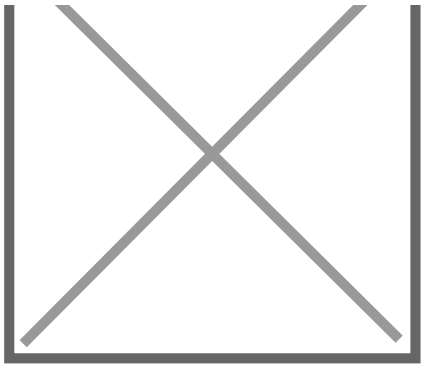


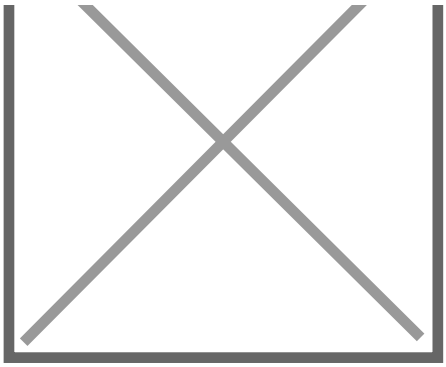
You change the rmt_channel to 1 because 0 is being used by the built-in LED of the MSR-2.

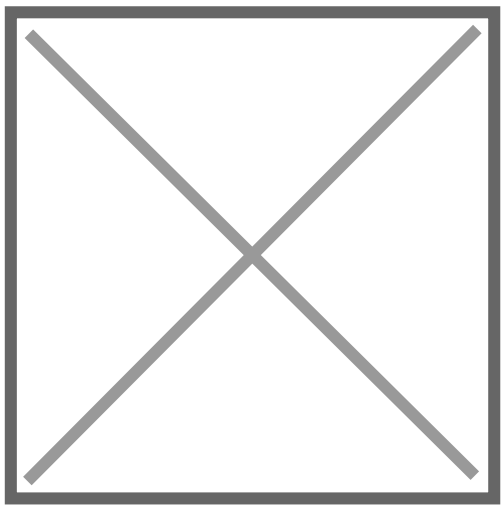
Step 4. Hit save and then install in the top right. It should have a popup where you select "wirelessly" then it will begin compiling the firmware and finally installing the compiled firmware to your MSR-2.

Step 5. Go into home assistant and confirm you now have a new light entity called Bed LED

Step 6. Click on the name "Bed LED" circled and it will pop up a color picker. You can then choose the color wheel option to pick any color of the rainbow, or select "effect" and choose an effect.







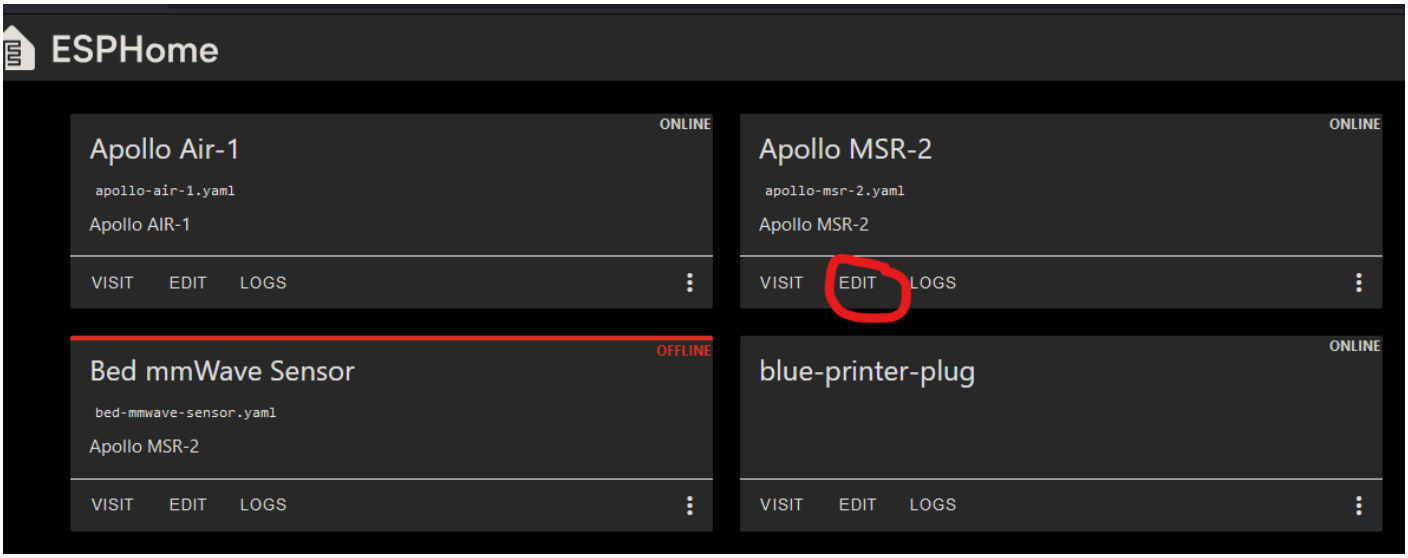
That's all folks! Thanks to Smart Home Sellout for putting this tutorial together!

How to edit your Sensor's LUX update interval

This guide will show you how to edit the lux updates down to 3-5 seconds.

Your sensor is defaulting to 60 seconds for updates to the state of the lux sensor. This is the default for all esphome devices using lux because it uses less Wi-Fi airtime fairness which means it is less chatty to your Wi-Fi AP or router. I would not personally go below 5 seconds.

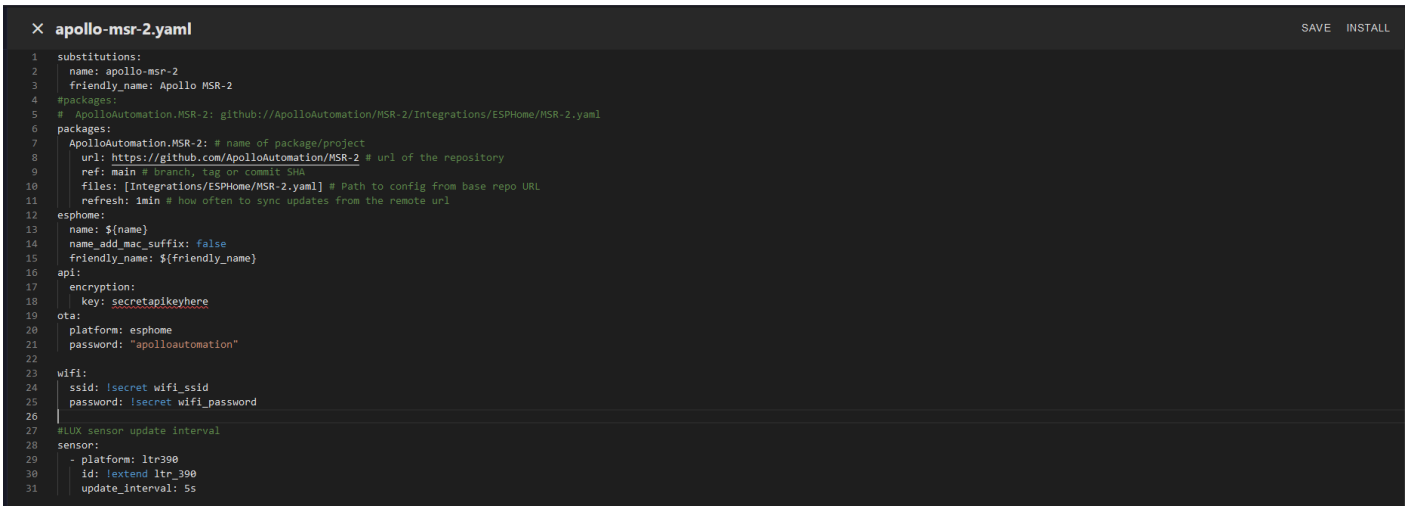
1. Open the Esphome dashboard and click "edit" under the device you want to edit.



2. Copy this code and enter it just like shown in the next step. Make sure there are no extra spaces or any other characters it needs to look just like the example in the next step.

```
#LUX sensor update interval
sensor:
  - platform: ltr390
    id: !extend ltr_390
    update_interval: 5s
```

3. Paste the code you copied in step 2 below your sensor's existing yaml as shown below.



4. Click "SAVE" and then click "INSTALL" as shown in the image above. Once that is finished your sensor should now be reporting at your new update_interval such as 5 seconds!