
SunFounder PiPower

www.sunfounder.com

Oct 11, 2023

CONTENTS

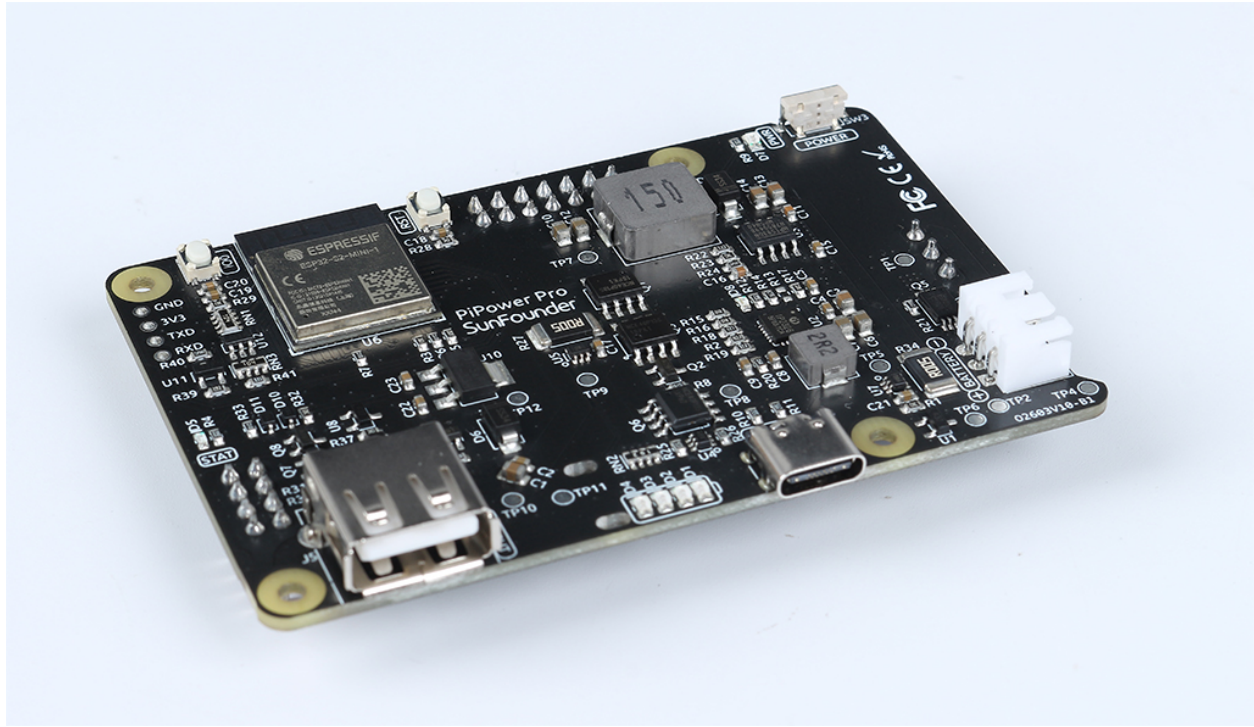
- 1 Component List 3**
- 2 Assemble the PiPower 5**
- 3 Features 13**
 - 3.1 Detailed Introduction 14
 - 3.2 Battery Indicators 17
- 4 Start to Play 19**
 - 4.1 Install the HassOS 19
 - 4.2 Add PiPower Pro in Home Assistant 29
 - 4.3 Configure Dashboard 35
 - 4.4 Add Card by Code Editor 37
 - 4.5 PiPower Pro Entity 44
 - 4.6 Setting up Safe Shutdown 45
 - 4.7 Coulomb Counter (Beta) 54
 - 4.8 Custom Development 55
 - 4.9 Multiple PiPower Pro Units 56
 - 4.10 IO Expansion 56
- 5 FAQ 59**
 - 5.1 PiPower Pro not working? 59
 - 5.2 Which single-board computers can PiPower Pro be used on? 59

Thanks for choosing our PiPower.

Note: This document is available in the following languages.

-
-
-

Please click on the respective links to access the document in your preferred language.



What a UPS Does?

If your Raspberry Pi project requires constant power, relying only on the main power system is not a viable option. Depending on your location, power drops and surges may occur frequently and often last for hours. Any power fluctuations can damage your Raspberry Pi, and a power outage will immediately shut it down. Consequently, it will not shut down safely, which can result in all data on the SD card being lost, increasing the chances of it being destroyed.

Therefore, the use of an uninterruptible power supply (UPS) is recommended.

With a UPS, if there is a power interruption from the mains (interruption = power outage), the battery or other power source will take over and continue to power the device without shutting it down. A UPS is often considered an emergency power source. After the main power source is repaired, the UPS will recharge and be ready to handle the next disaster.

About PiPower

That's why we designed PiPower in the first place. PiPower can be used as a second power source for the Raspberry Pi. A USB-C mains power supply plugged into the PiPower will directly power the Raspberry Pi and charge the battery at low current. PiPower can seamlessly power up a Raspberry Pi in the event of a power outage or disconnection of USB-C mains power.

PiPower can output 5V/3A power supply to meet various Raspberry Pi usage situation. It has 4 power indicators; each indicator represents 25% of the power, and is equipped with a power switch to turn on/off the power of the Raspberry

Pi without plugging or unplugging the power cord.

Warning: When you put the battery in for the first time or when the battery is unplugged and put in again, the battery will not work properly, at this time, you need to plug the Type C cable into the charging port to turn off the protection circuit, and the battery can be used normally.

About PiPower Pro

PiPower Pro builds upon PiPower, integrating an ESP32 S2 module that enables real-time monitoring of the module's battery voltage and current status, input/output voltage and current. It also features intelligent charging current adjustment and seamless switching between input and battery power, ensuring uninterrupted power output.

When integrated with Home Assistant, users can easily access and review all parameter data, as well as configure automation for smart device scenarios.

Additionally, PiPower Pro offers external IO interfaces for controlling the on/off state of sub-devices. With its open-source ESPHome configuration, users can customize IO functionalities and expand the system with more sensors.

If you have any questions while using our product, please send an email to service@sunfounder.com. We will respond as soon as possible.

About the display language

Note: In addition to English, we are working on other languages for this course. Please contact service@sunfounder.com if you are interested in helping, and we will give you a free product in return.

Currently the online tutorial supports English, German and Japanese. Please click the **Read the Docs** icon in the lower left corner of the page to change the display language.

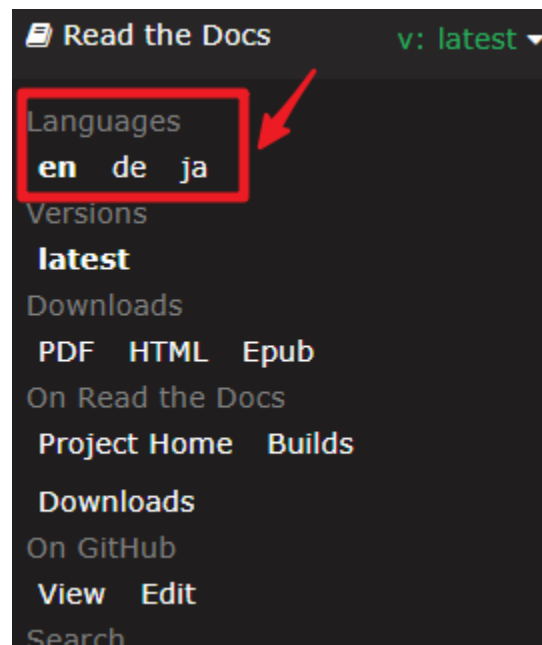
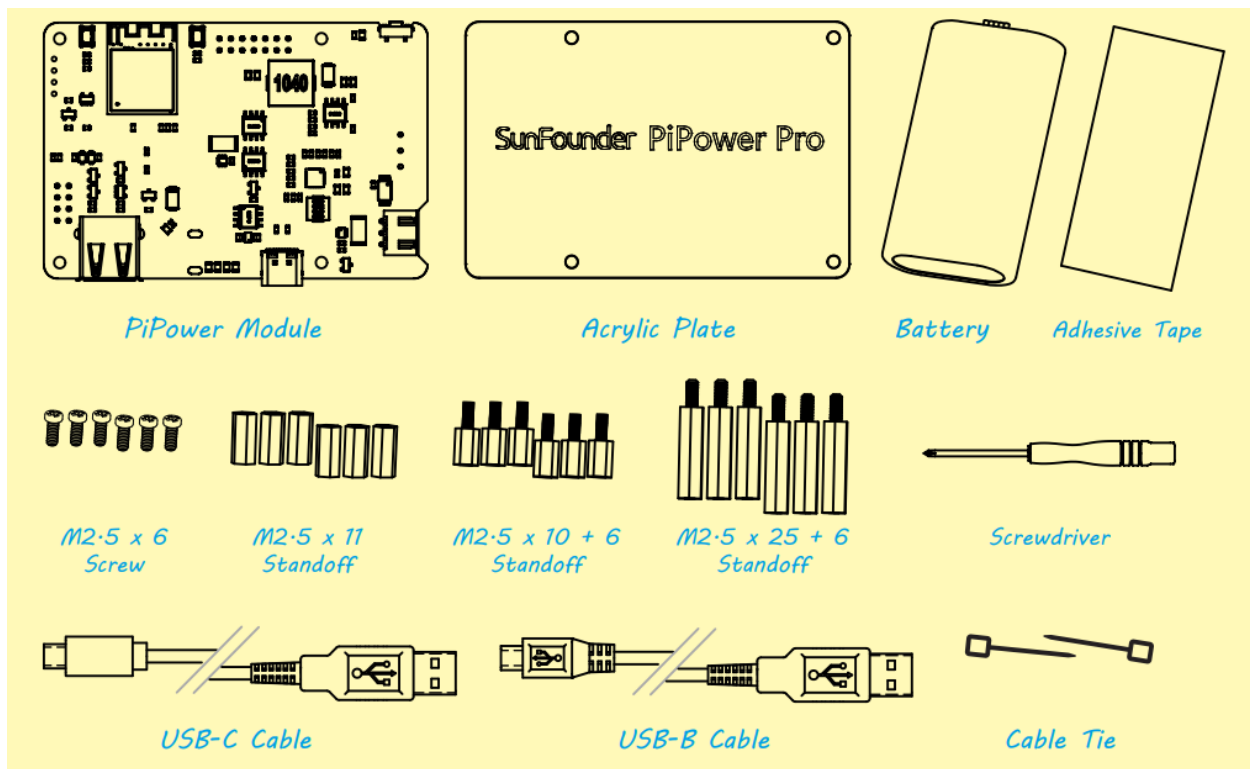


Table of contents

COMPONENT LIST

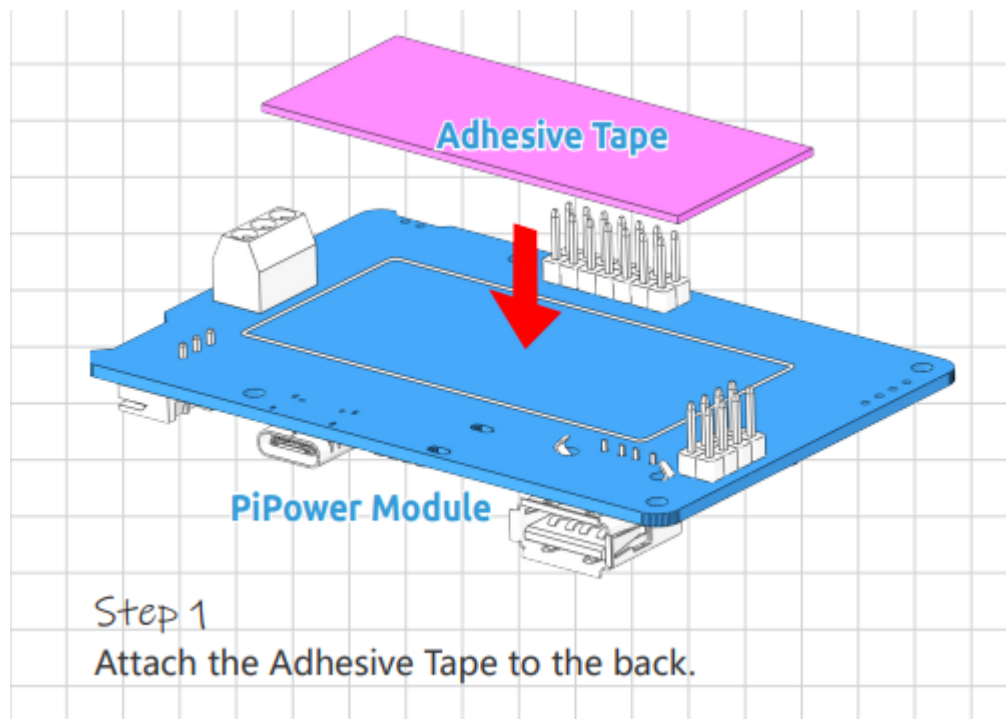


ASSEMBLE THE PIPOWER

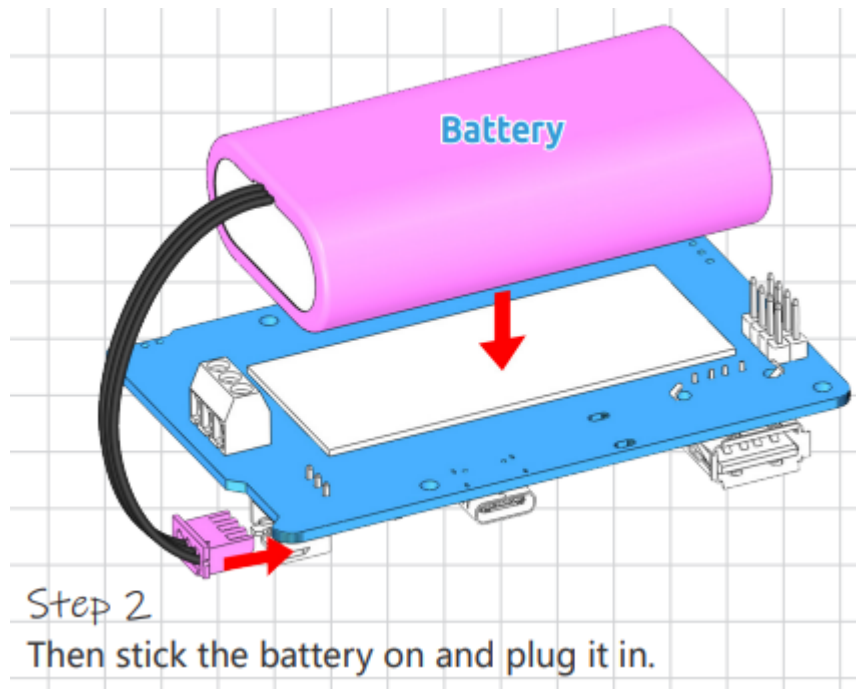
After getting familiar with the components in the package, we start to assemble PiPower.

In the next steps, there are a lot of details you need to notice, especially the assembly position of the battery and the clear acrylic back cover.

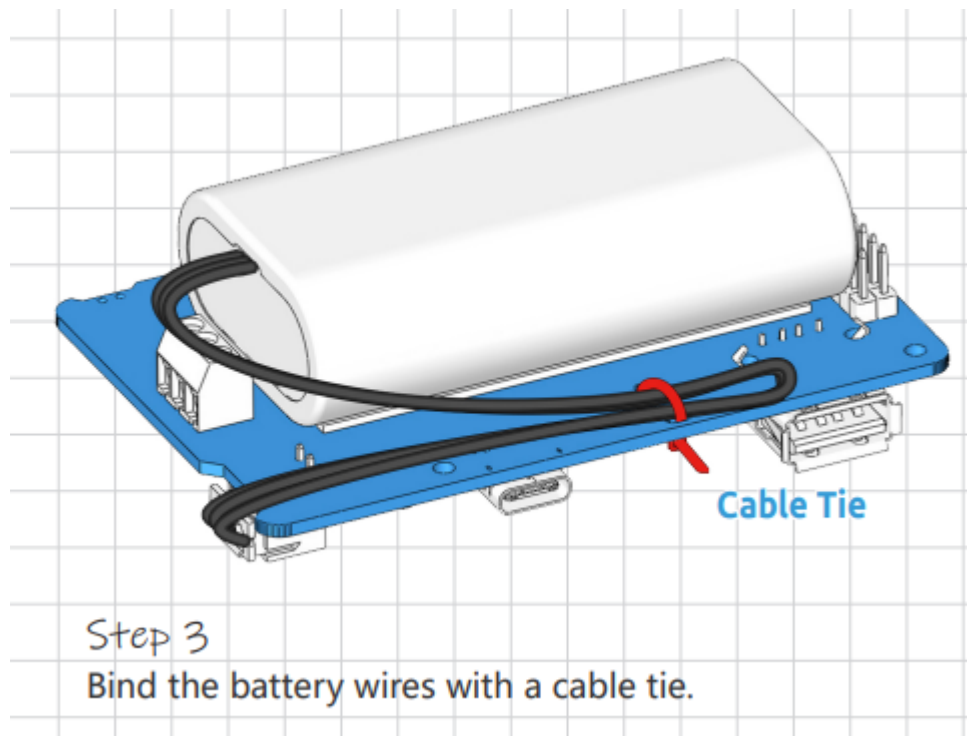
1. Attach the Adhesive Tape to the back.



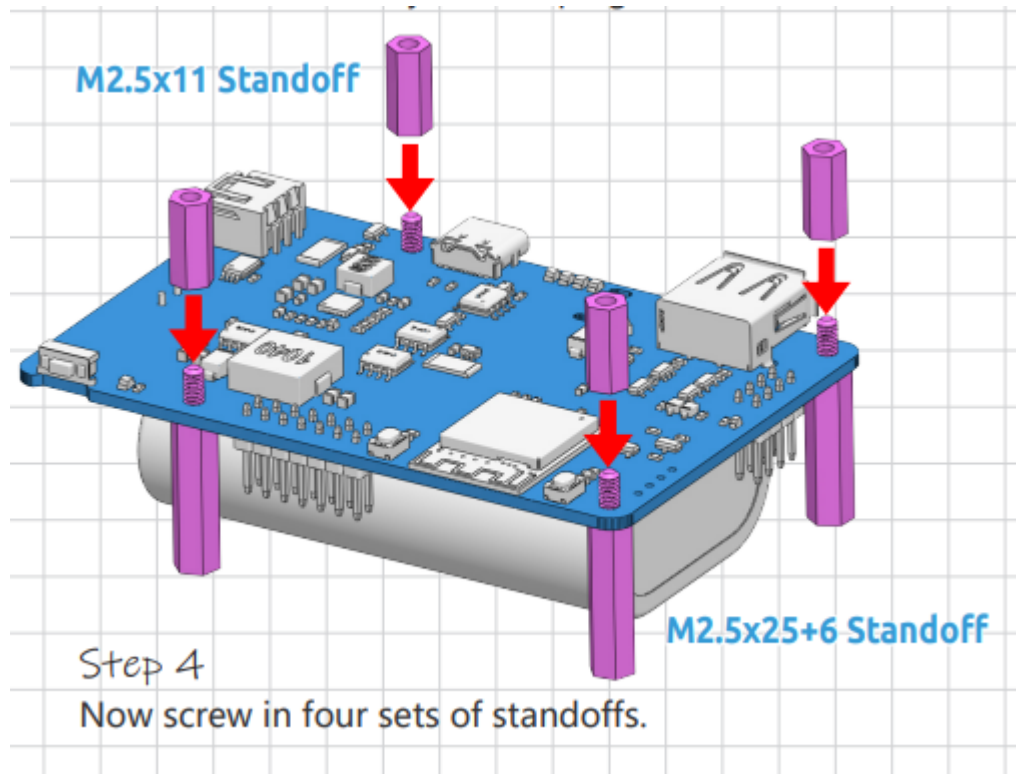
2. Then stick the battery on and plug it in.



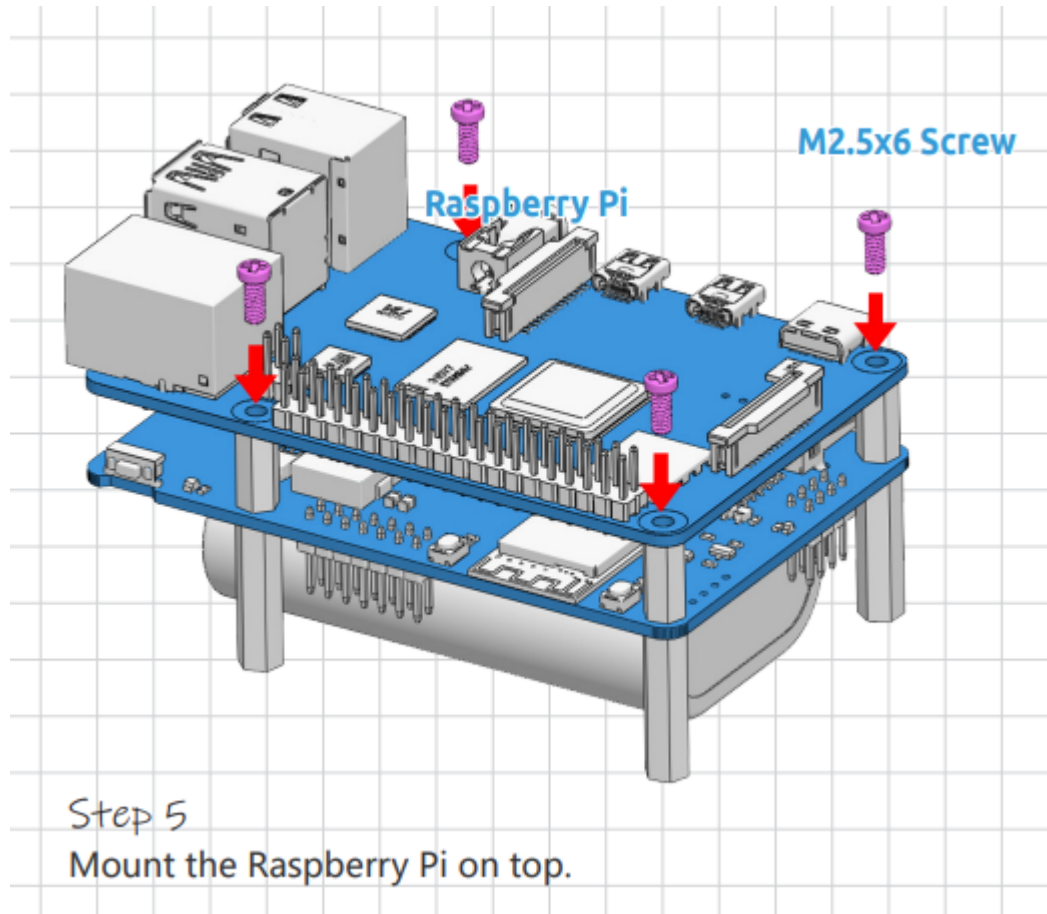
3. Bind the battery wires with a cable tie.



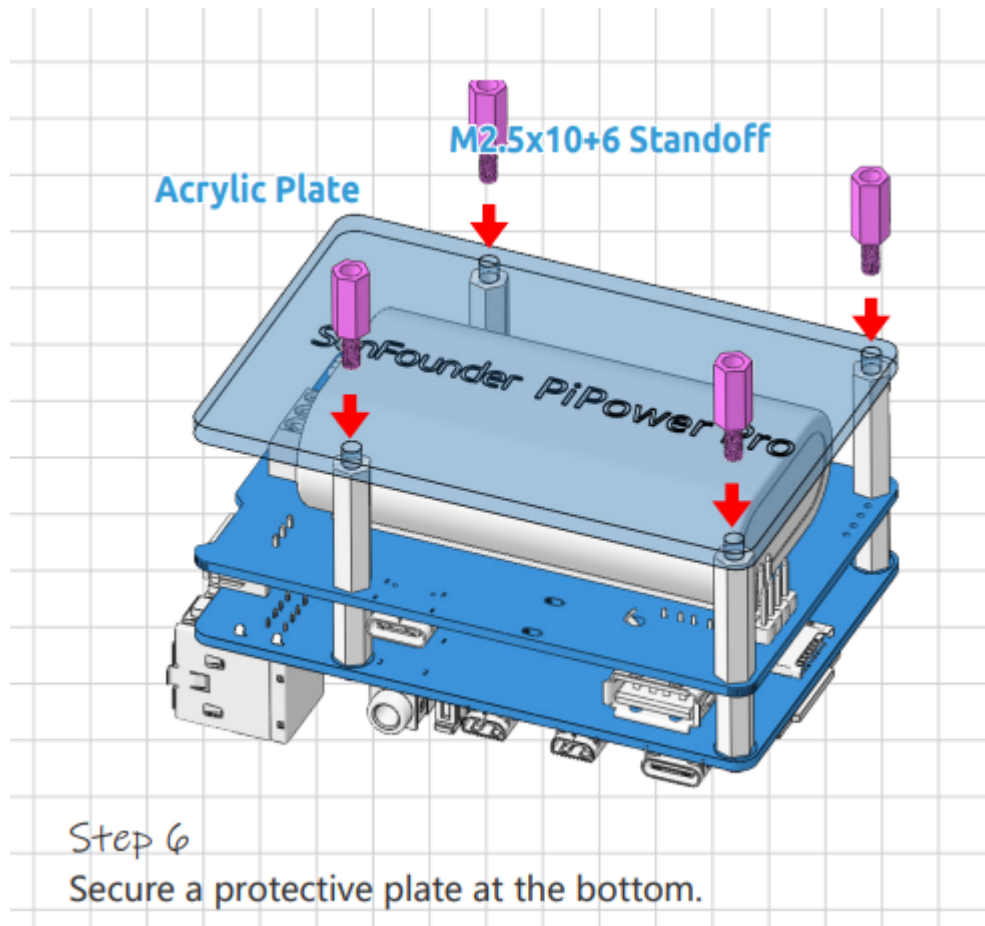
4. Now screw in four sets of standoffs.



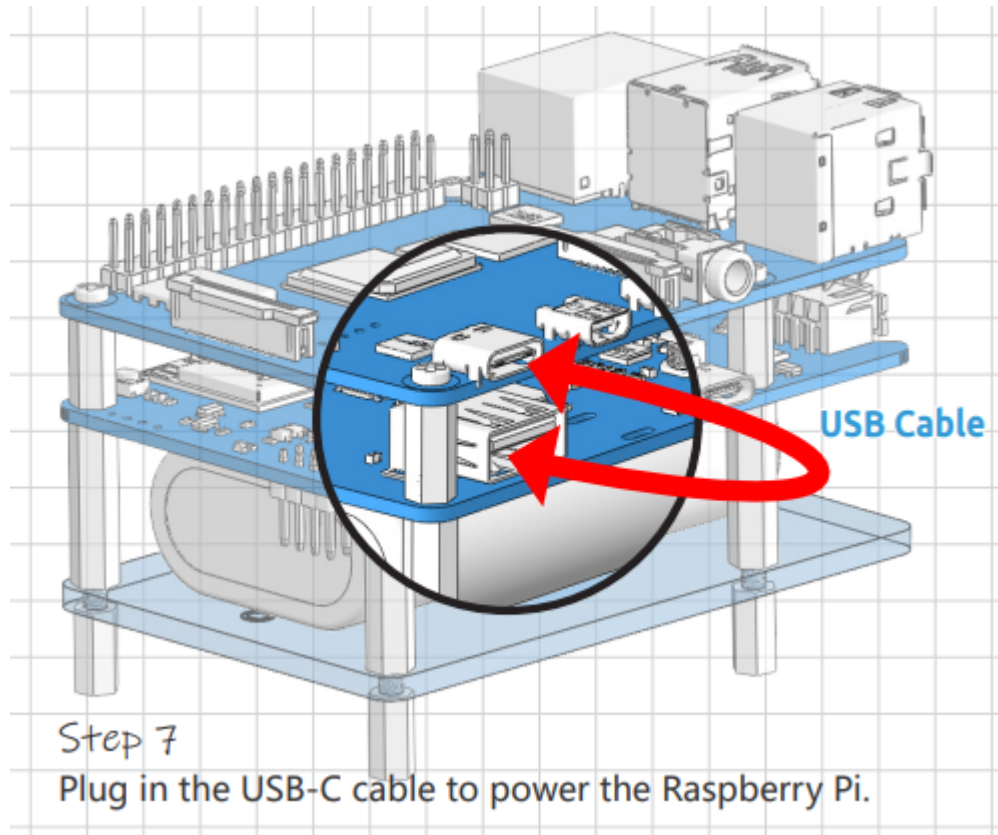
5. Mount the Raspberry Pi on top.



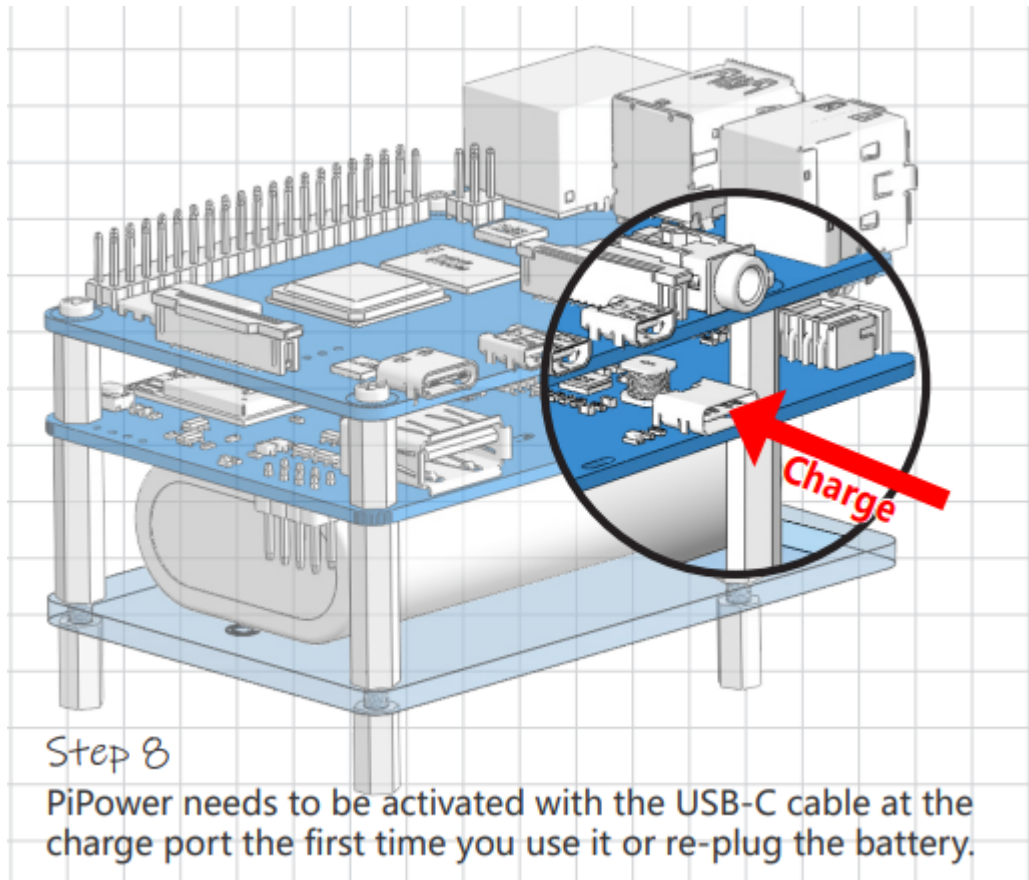
6. Secure a protective plate at the bottom.



7. Plug in the USB-C cable to power the Raspberry Pi.

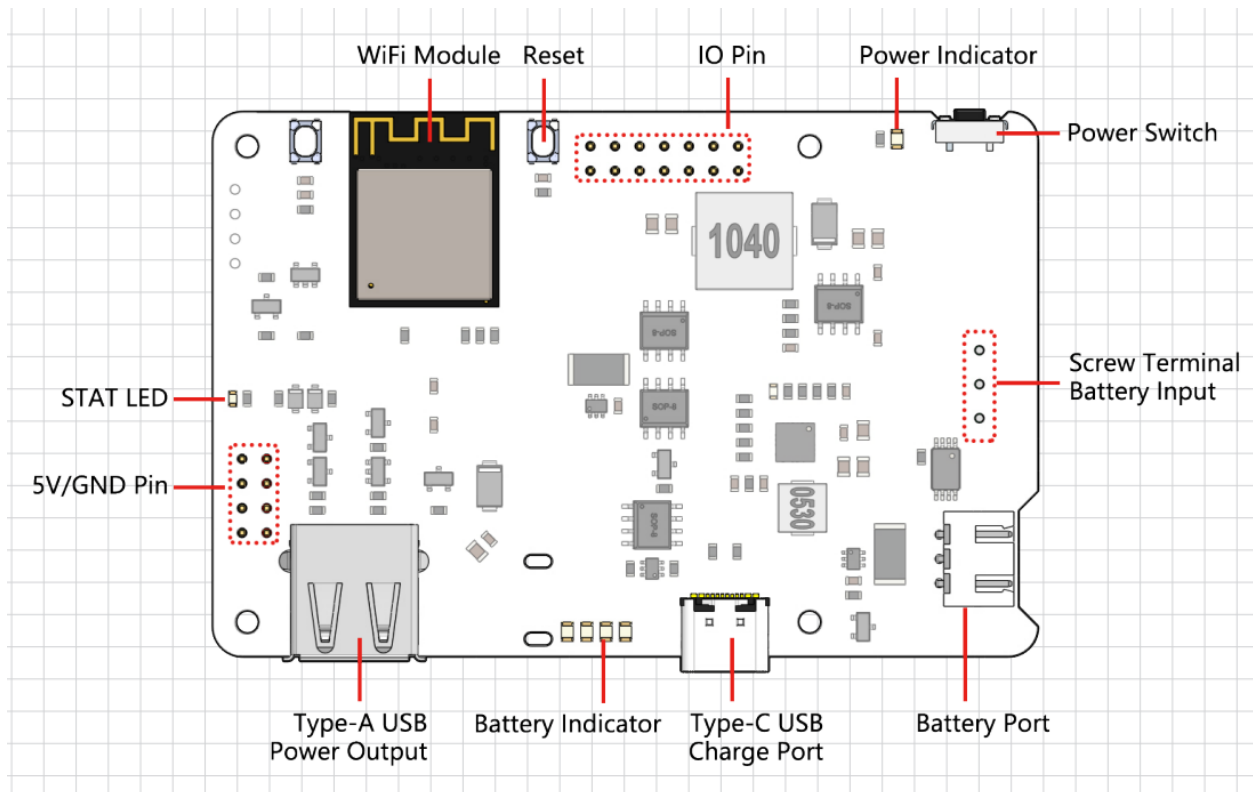


8. PiPower needs to be activated with the USB-C cable at the charge port the first time you use it or re-plug the battery



Warning: When you put the battery in for the first time or when the battery is unplugged and put in again, the battery will not work properly, at this time, you need to plug the Type C cable into the charging port to turn off the protection circuit, and the battery can be used normally.

FEATURES



- **Pass Through Charging**
- **Shutdown Current < 0.5mA**
- **Input:**
 - USB Type-C, 5V/3A
 - Battery Input
- **Output**
 - USB Type-A, 5V/3A
 - 2x4P P2.54 pin headers
- **Charging Power 5V/2A**
- **Equipped Battery**
 - Type: 3.7V Lithium-ion batteries x 2

- Capacity: 2000mAh
- Connector: PH2.0, 3P
- **Over Discharge Protection Voltage 6.0V**
- **Overcharge Protection Voltage 8.4V**
- **Dimension: 90mm x 60mm x 24.9mm**
- **On-board Indicators**
 - 1 x Charging Indicator (CHG)
 - 1 x Power Indicator (PWR)
 - 4 Battery Indicators (D4 ~ D7)
- **On-board Power Switch**
- **On-board MCU ESP32 S2**

3.1 Detailed Introduction

STAT LED

The STAT LED is the status indicator for the ESP32 S2.

- Off: ESP32 S2 is powered off.
- Slow blinking: ESP32 S2 is powered on, but Wi-Fi is not connected.
- Steady on: ESP32 S2 is powered on and Wi-Fi is connected.

Note: The so-called “ESP32 S2 powered off” state refers to the situation when the USB Type C power is connected. In this state, the ESP32 S2 is technically “powered off” but not completely shut down. The power LED still requires the ESP32 S2 to control its illumination, and some functions may remain operational. However, when you unplug the USB Type C power, the ESP32 S2 will shut down completely.

Switch Power Path

PiPower Pro integrates a power path function that automatically switches power paths to ensure maximum output protection.

1. When external power is connected, the 5V output is directly supplied through the external USB Type C and can be turned off using the switch. The external power source charges the battery with as much current as possible (see charging current) while ensuring the input voltage is greater than 4.6V.
2. In the moment of power disconnection, the system automatically switches to battery power output with seamless transition, ensuring the system can function properly during power interruptions.
3. If the external power is below 4.6V, the system automatically switches to battery backup power to prevent external devices from losing power.

Table 1: Output Power Logic

| Switch | External Power | Output Status |
|--------|---------------------------------|----------------|
| On | Plugged in | External Power |
| On | Unplugged or voltage below 4.6V | Battery Power |
| Off | Plugged in | Off |
| Off | Unplugged or voltage below 4.6V | Off |

Charging Power

Under the power-on state, the charging current will automatically adjust based on the input voltage.

Table 2: Charging Current Logic

| Switch | Charging Current |
|--------|---------------------------------|
| On | Adjusted based on input voltage |
| Off | 2A |

1. When the switch is in the “Off” state, PiPower Pro does not supply power externally, and the charging current reaches a maximum of 2A, allowing for fast charging. The charging time from 0% to 100% is approximately 2 hours and 10 minutes.
2. In the “On” state, since PiPower Pro needs to supply power externally, the external USB also needs to provide power to the battery. To ensure the voltage of the USB power supply remains stable, the charging current is adjusted based on the input voltage, ensuring the voltage stays above 4.6V.

Over-discharge Protection

When the single battery voltage is below 3V, the battery protection activates and the battery is no longer discharged.

When the battery is unplugged, due to the mechanism of the on-board over-discharge protection circuit, the voltage will be considered too low, thus activating the protection circuit; when you replug the battery into the PiPower, the battery will not work properly, at this time, you need to plug the Type C cable into the charging port to turn off the protection circuit, and the battery can be used normally.

Overcharge Protection

Charging ends when the total battery voltage reaches 8.4V.

Charge Balance

When the voltages of the two batteries are not equal, the charging current of the two batteries is automatically adjusted to balance the two batteries.

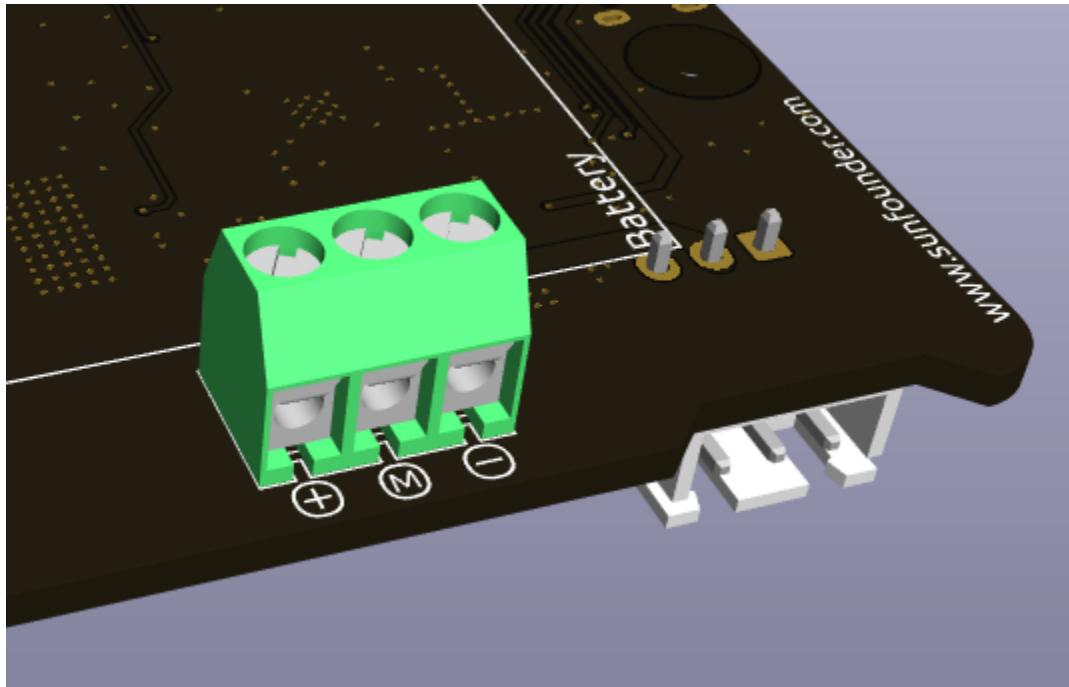
When a single battery exceeds 4.2V, the voltage divider resistor channel conducts and the battery charging current is reduced or even discharged.

Battery

The product comes with two 3.7V 18650 lithium-ion batteries in series, featuring an XH2.54 3P connector, with a nominal capacity of 2000mAh.

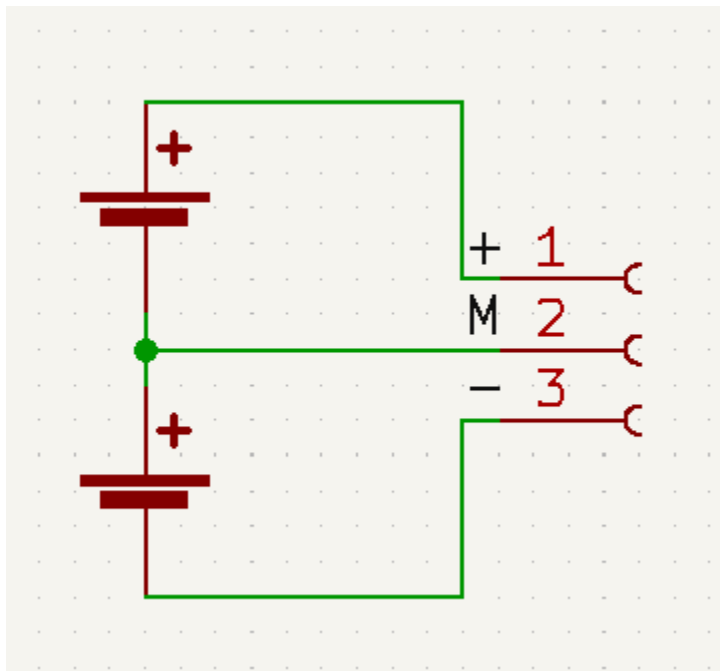
- Composition: Li-ion (Lithium-ion)
- Capacity: 2000mAh, 14.8Wh
- Weight: 90.8g
- Cells: 2
- Connector: XH2.54 3P
- Overcharge Protection Voltage: 4.2V per cell
- Over-discharge Protection: 3V

External Battery



You can connect your own battery using the Screw Terminal. The device only supports two 3.7V lithium-ion or lithium-polymer batteries. It's preferable for the batteries to have a protection board and ensure an output of more than 15W.

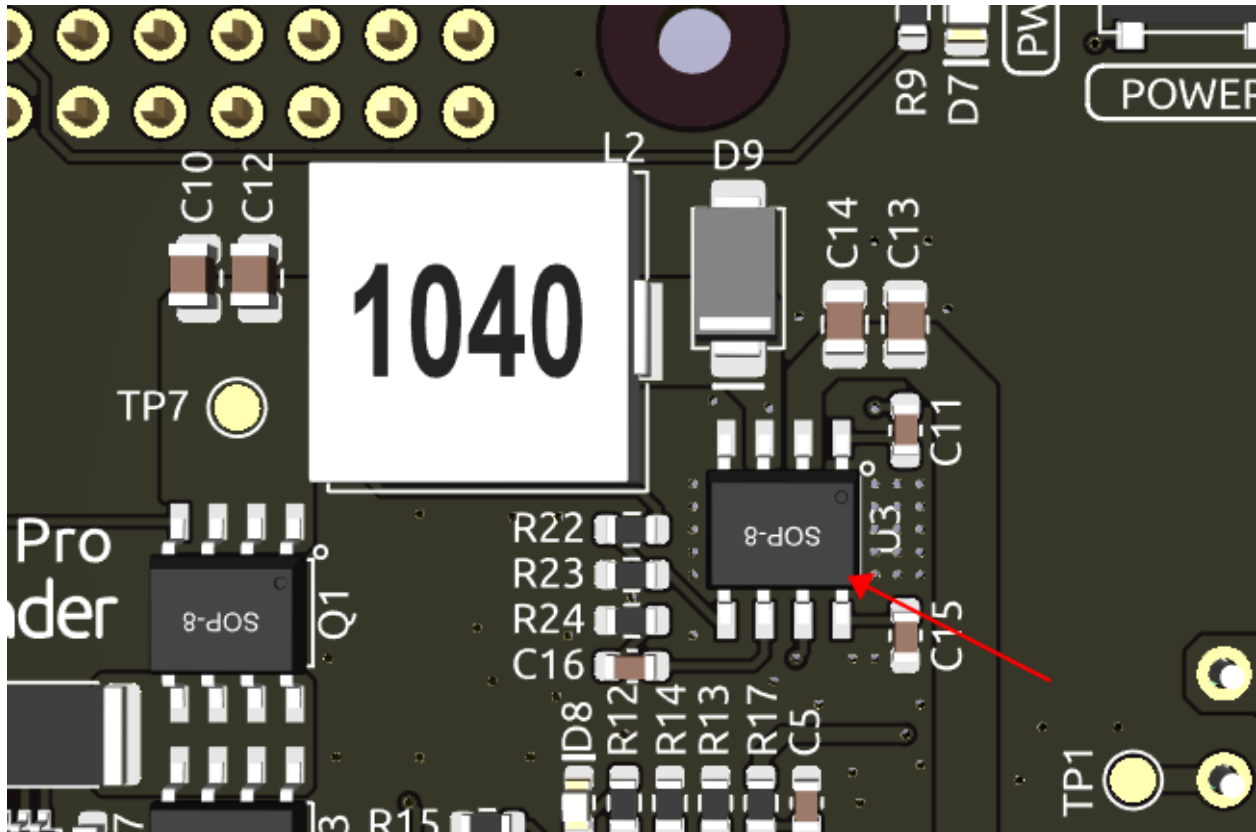
Warning: Do not connect the external battery and the included battery at the same time!



Temperature

When the output power reaches the maximum nominal 5V/3A, the temperature of DC-DC buck chip U1 will rise to about 70-80 degrees Celsius, so be careful not to touch it to prevent burns and keep ventilation. When the temperature

reaches the DC-DC protection temperature of 75 degrees Celsius, the DC-DC will shut down to prevent overheating damage.



D8 LED

The D8 LED is a charging status indicator provided by the IP2326 charging chip. Originally, this light was designed to indicate both the charging status and any abnormalities with the battery. However, it can only detect if there's current flow in the charging output. This output current can be routed through a DC-DC converter to output 5V. In simpler terms, when there's insufficient input power, the battery will supplement the power supply, and during this, the LED remains steadily lit, which can be misleading. However, the LED was retained as it can indicate if the battery is functioning normally (the LED will blink if the battery isn't inserted).

3.2 Battery Indicators

The relationship between the battery indicators and voltage is as follows:

- 4 LEDs all on: voltage > 7.7V
- 3 LEDs on: voltage > 7.2V
- 2 LEDs on: voltage > 6.7V
- 1 LED on: voltage > 6.4V
- 4 LEDs all off: voltage < 6V at this time batteries need to be charged.

START TO PLAY

PiPower Pro can be integrated into Home Assistant. To do this, you need to have a Raspberry Pi with HassOS installed. Please follow the link below for setup.

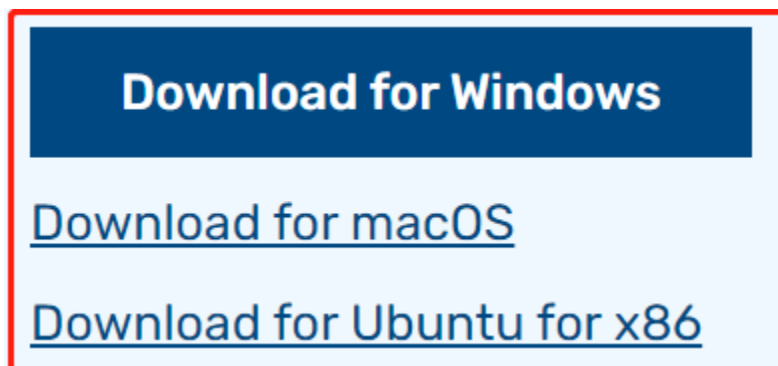
4.1 Install the HassOS

This section will guide you in installing the Home Assistant operating system on your Raspberry Pi. Please note that this process will result in the loss of all existing content on your Raspberry Pi system. It is important to backup your data before proceeding.

Step 1

Raspberry Pi have developed a graphical SD card writing tool that works on Mac OS, Ubuntu 18.04 and Windows, and is the easiest option for most users as it will download the image and install it automatically to the SD card.

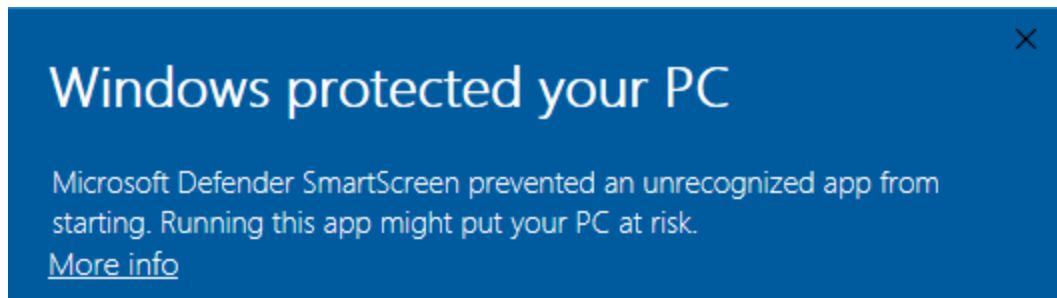
Visit the download page: <https://www.raspberrypi.org/software/>. Click on the link for the **Raspberry Pi Imager** that matches your operating system, when the download finishes, click it to launch the installer.



Step 2

When you launch the installer, your operating system may try to block you from running it. For example, on Windows I receive the following message:

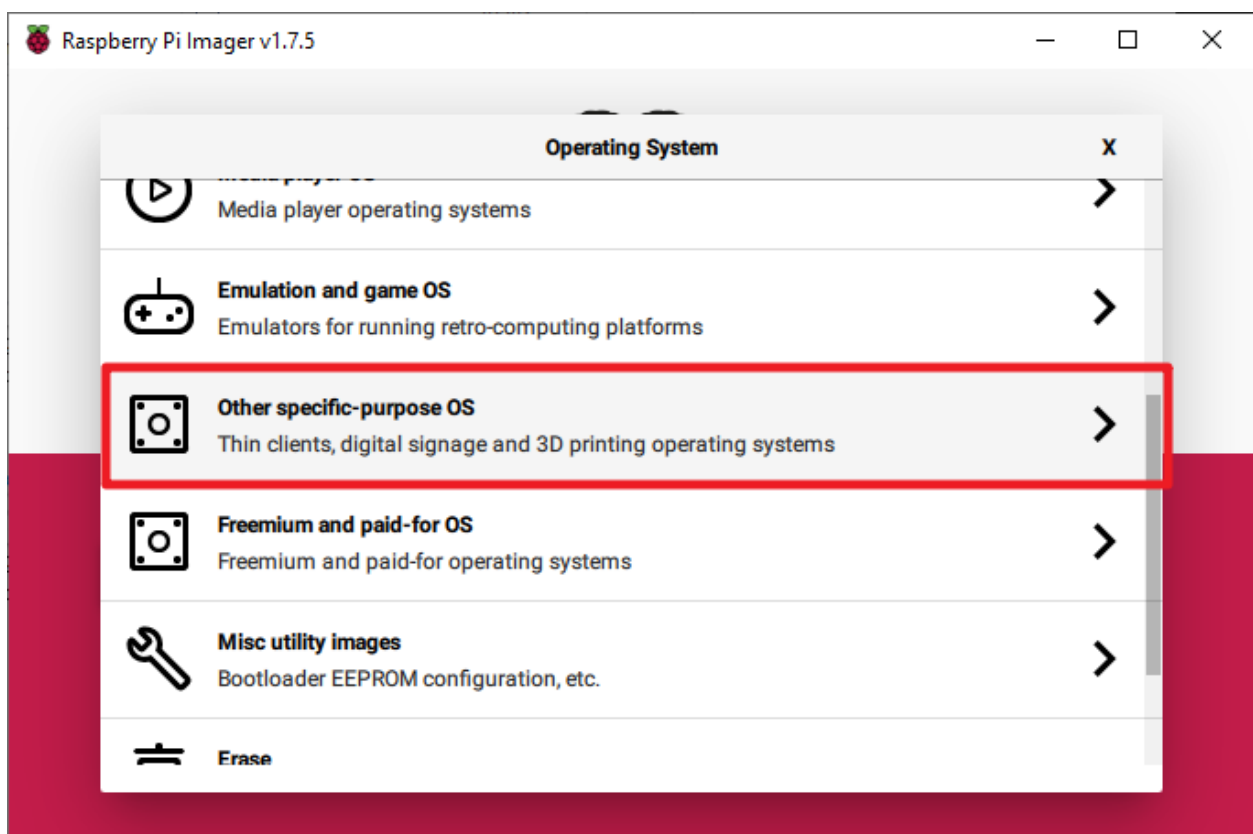
If this pops up, click on **More info** and then **Run anyway**, then follow the instructions to install the Raspberry Pi Imager.

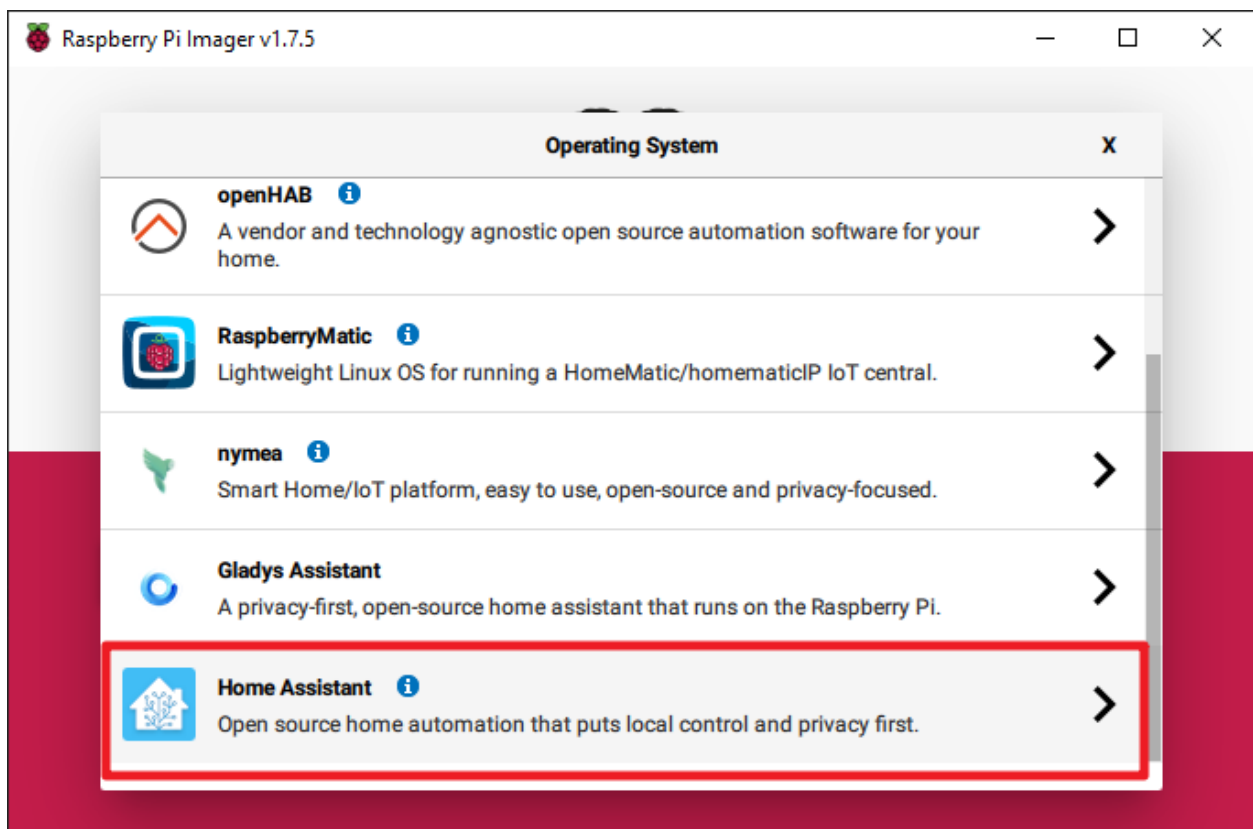
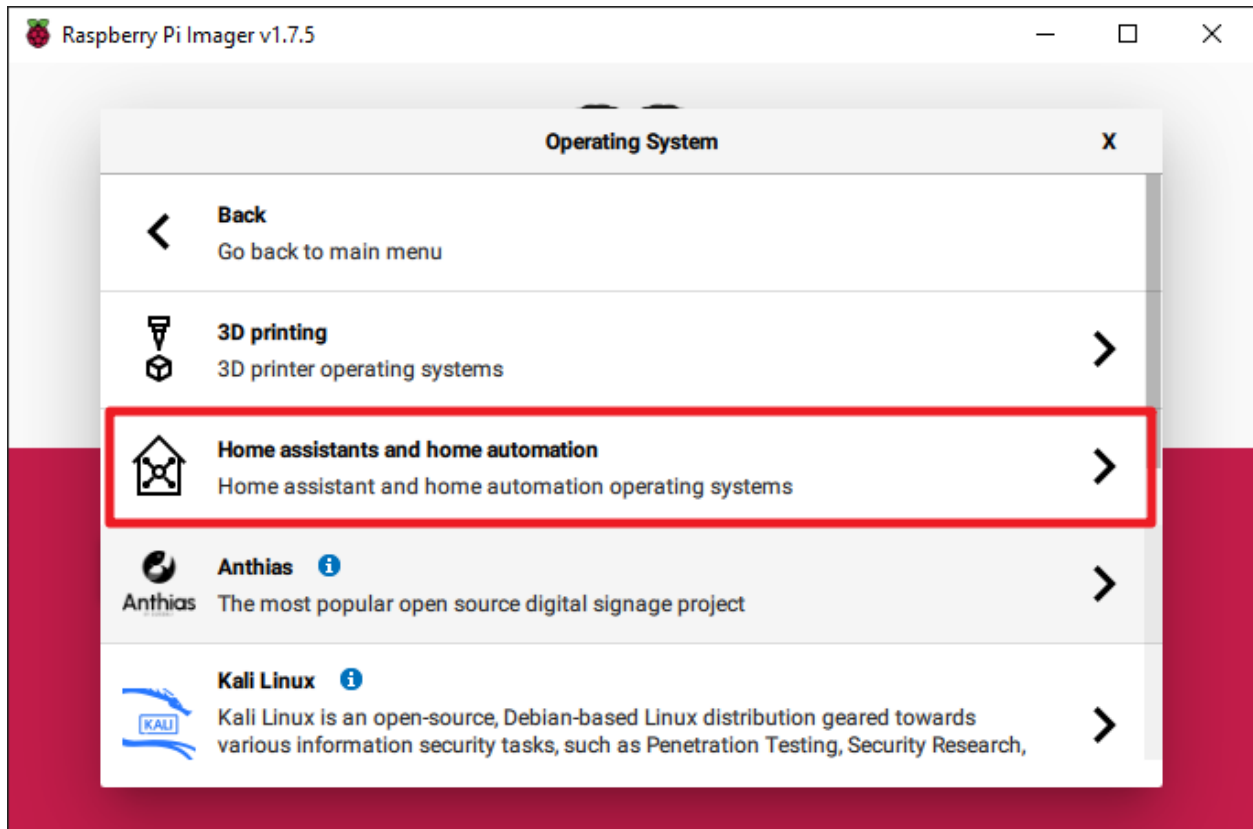
**Step 3**

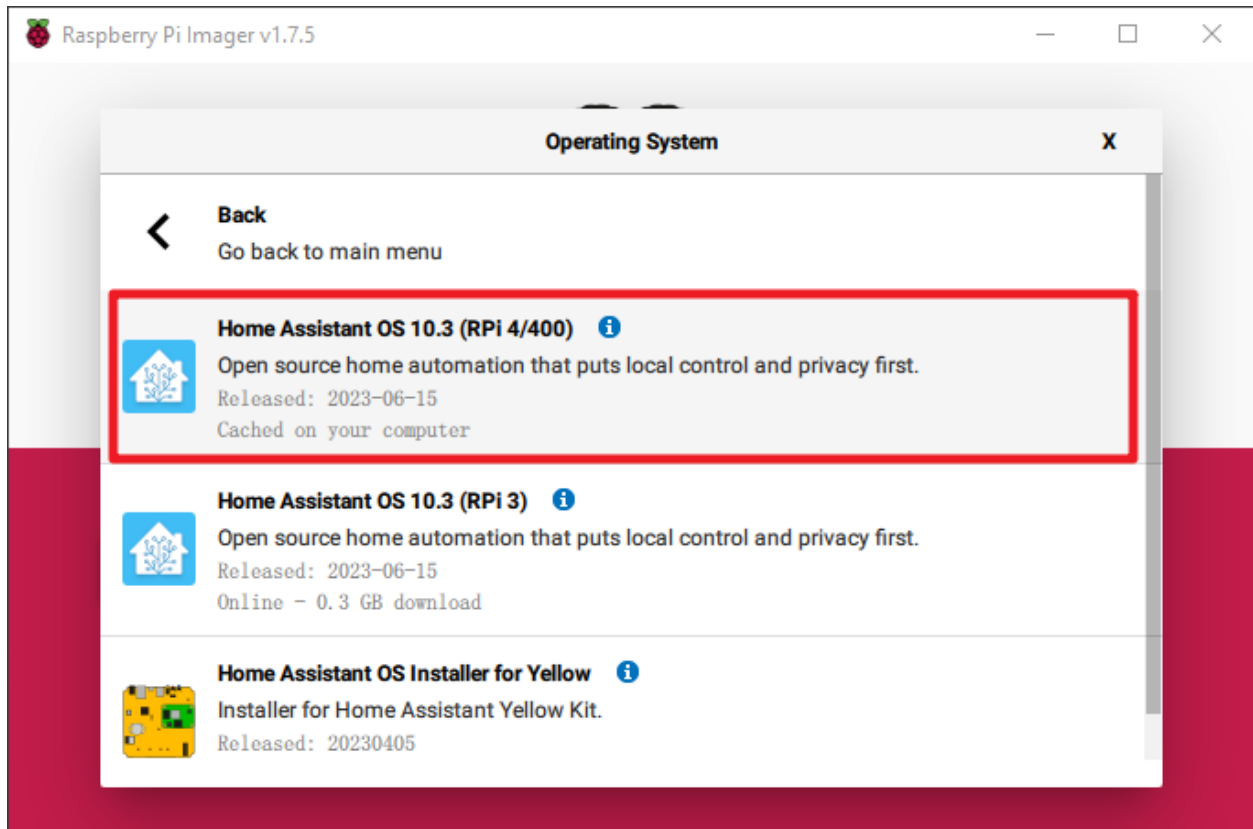
Insert your SD card into the computer or laptop SD card slot.

Step 4

In the Raspberry Pi Imager, select the OS that you want to install and the SD card you would like to install it on.

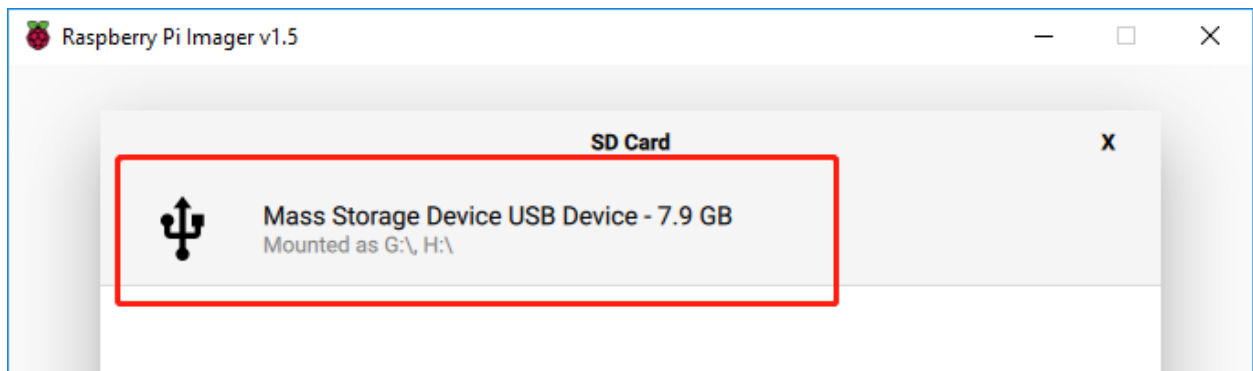






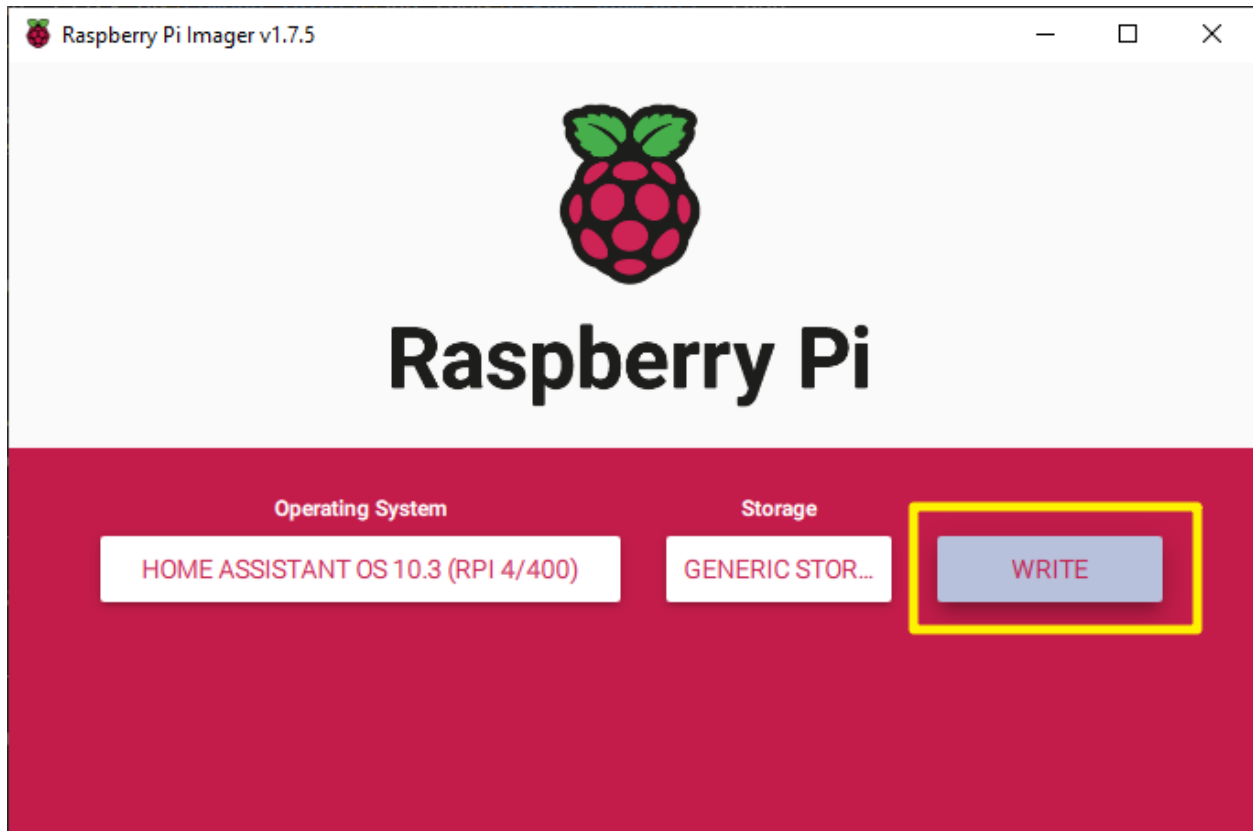
Step 5

Select the SD card you are using.



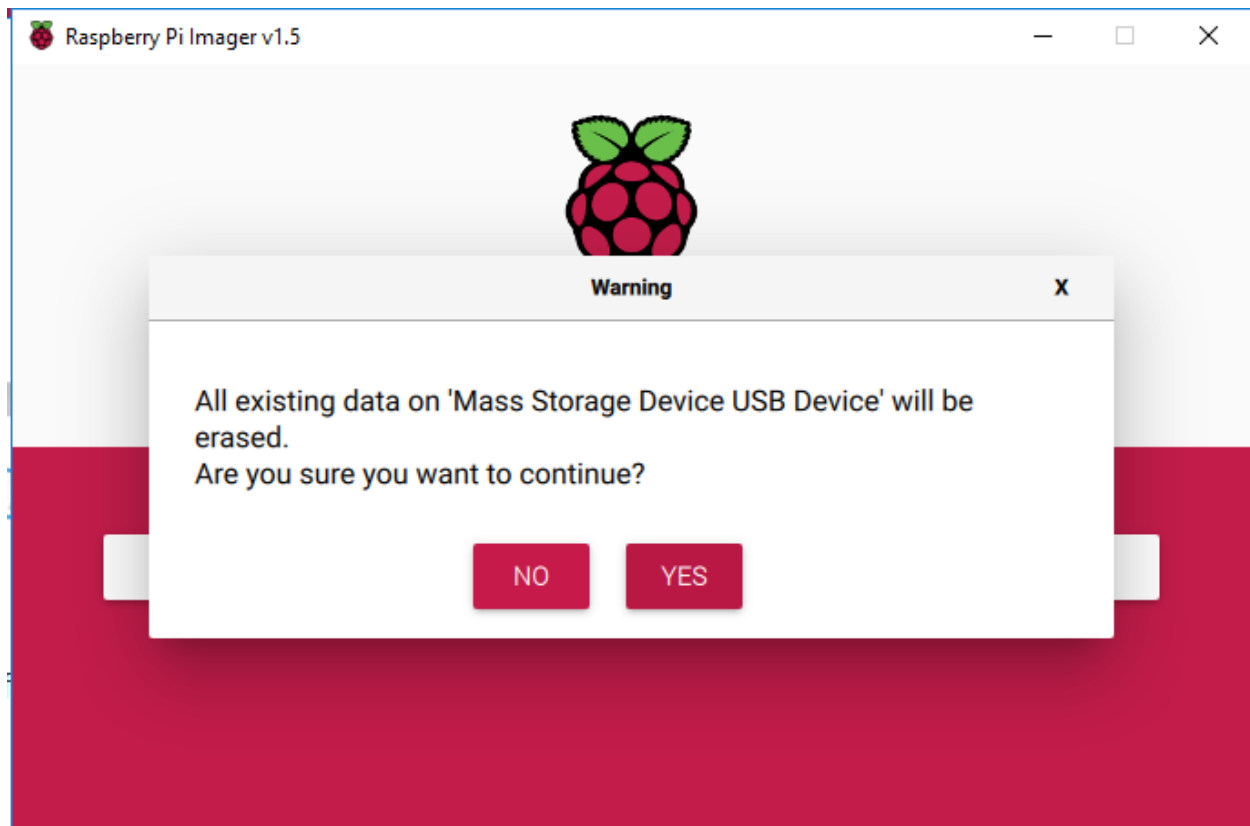
Step 6

Click the **WRITE** button.



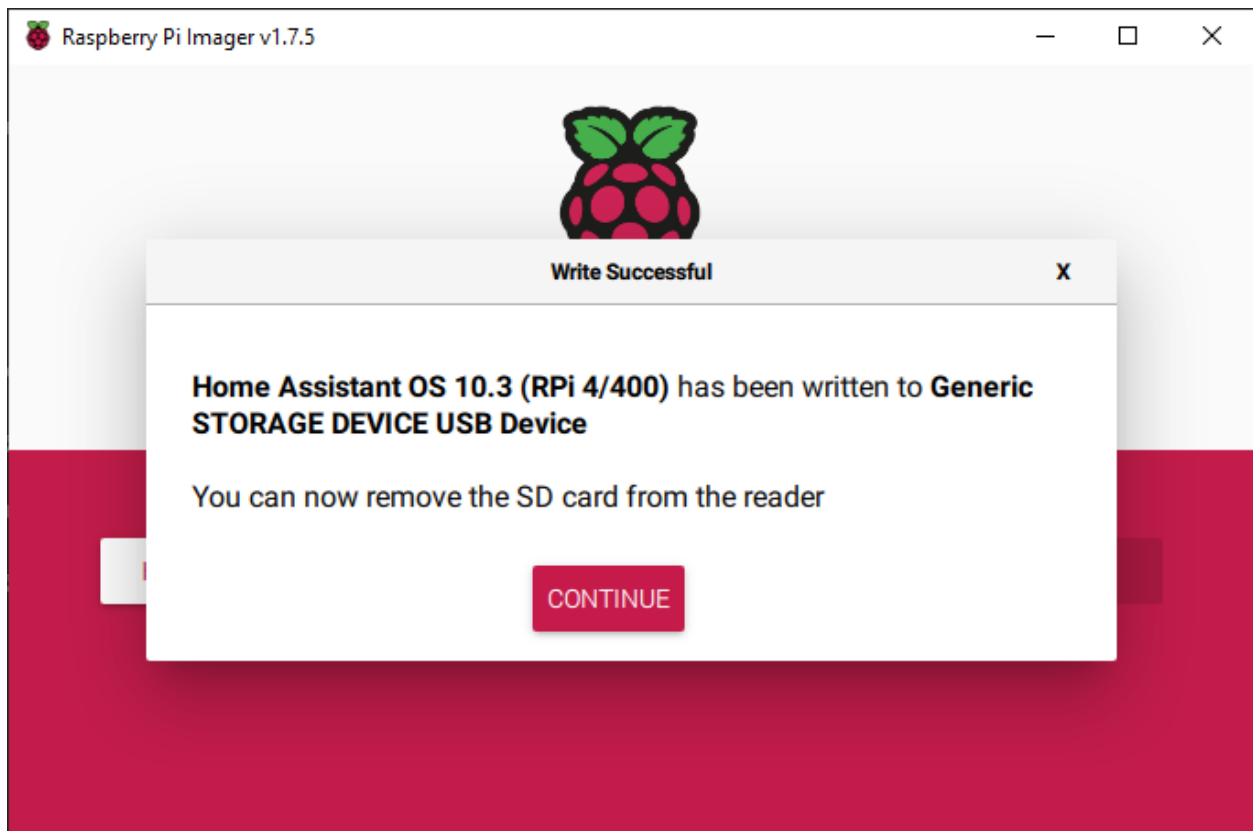
Step 7

If your SD card currently has any files on it, you may wish to back up these files first to prevent you from permanently losing them. If there is no file to be backed up, click **Yes**.



Step 8

After waiting for a period of time, the following window will appear to represent the completion of writing.

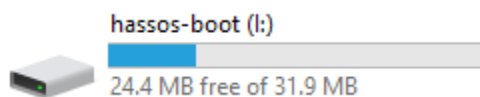


Step 9

Next, we will configure WiFi for Pironman.

Note: If you intend to use a wired connection for network access, you can skip this step.

Open File Explorer and access the SD card named Hassio-boot

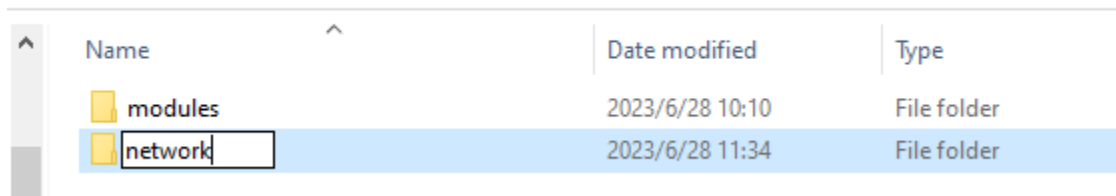


Create a new folder named CONFIG in the root partition.

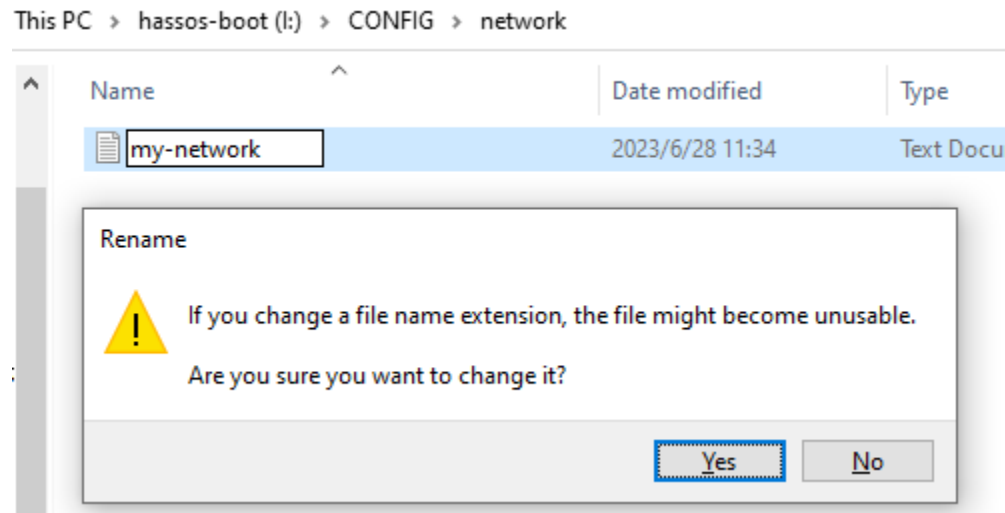


Create a folder named network inside the CONFIG folder.

This PC > hassos-boot (I:) > CONFIG >



Inside the network folder, create a new text file named `my-network` (without extension).



In the `my-network` file, write the following text, replacing `MY_SSID` and `MY_WLAN_SECRET_KEY` with your own network's SSID and password:

```
[connection]
id=my-network
uuid=72111c67-4a5d-4d5c-925e-f8ee26efb3c3
type=802-11-wireless

[802-11-wireless]
mode=infrastructure
ssid=MY_SSID
# Uncomment below if your SSID is not broadcasted
#hidden=true

[802-11-wireless-security]
auth-alg=open
key-mgmt=wpa-psk
psk=MY_WLAN_SECRET_KEY

[ipv4]
method=auto

[ipv6]
addr-gen-mode=stable-privacy
method=auto
```

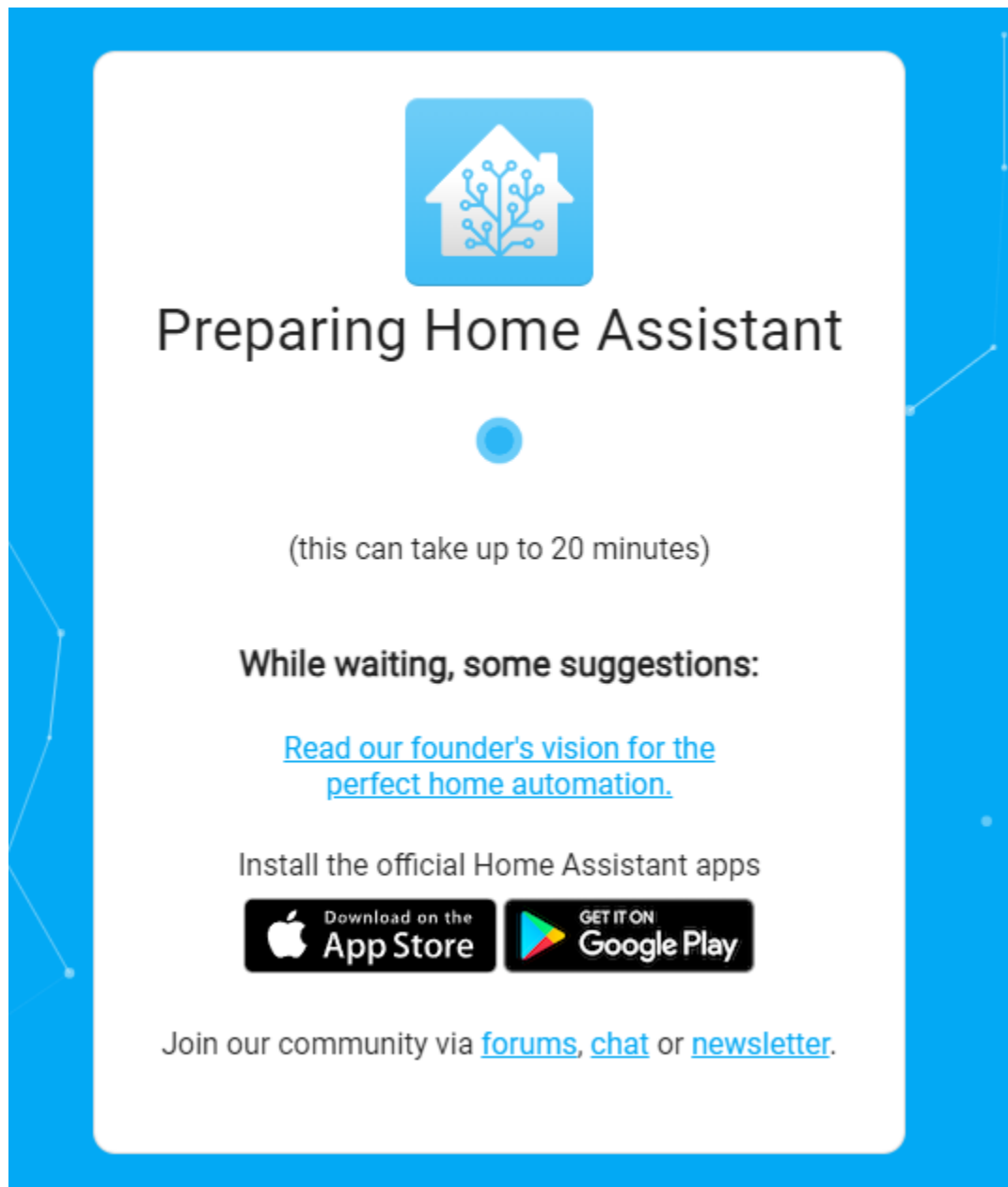
Save and exit the file.

Step 10

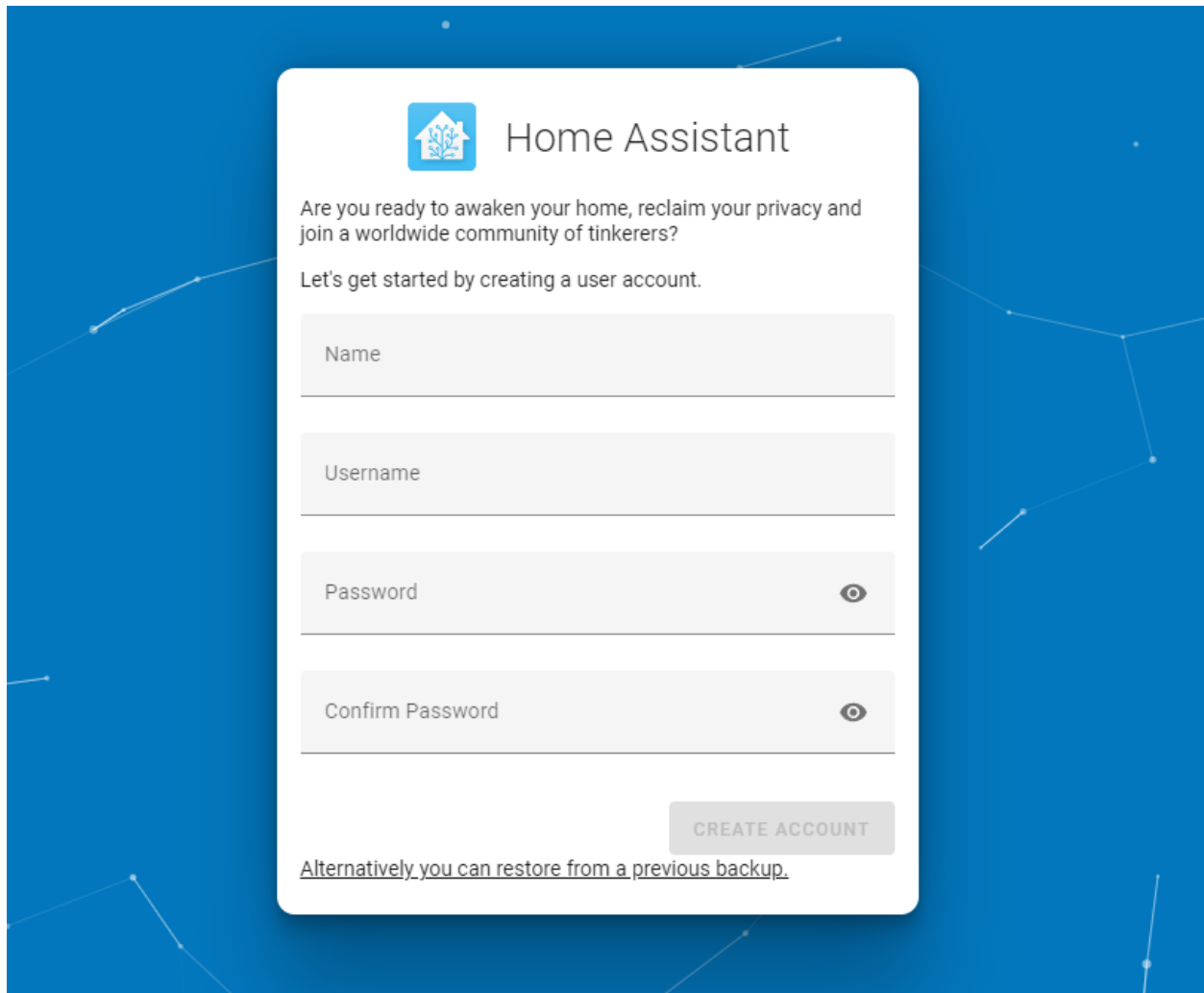
Remove the microSD card from your computer and insert it into the Raspberry Pi. Then, connect the power (and Ethernet cable if needed).


Go back to your computer and navigate to `homeassistant.local:8123`, or if that doesn't work, you can find the IP address by checking your router.

During the first use of Home Assistant, you may need to wait for some time as it performs initial setup.

**Step 11**

Next, you will be prompted to create the first account.

The image shows the Home Assistant account creation interface. It features a blue background with a white central card. At the top of the card is the Home Assistant logo (a house with a plant) and the text "Home Assistant". Below this, a message asks if the user is ready to awaken their home and join a community of tinkerers, followed by a prompt to create a user account. There are four input fields: "Name", "Username", "Password", and "Confirm Password". The "Password" and "Confirm Password" fields have eye icons to toggle visibility. A "CREATE ACCOUNT" button is located below the "Confirm Password" field. At the bottom, there is a link that says "Alternatively you can restore from a previous backup.".


 Home Assistant


Are you ready to awaken your home, reclaim your privacy and join a worldwide community of tinkerers?

Let's get started by creating a user account.

Name

Username

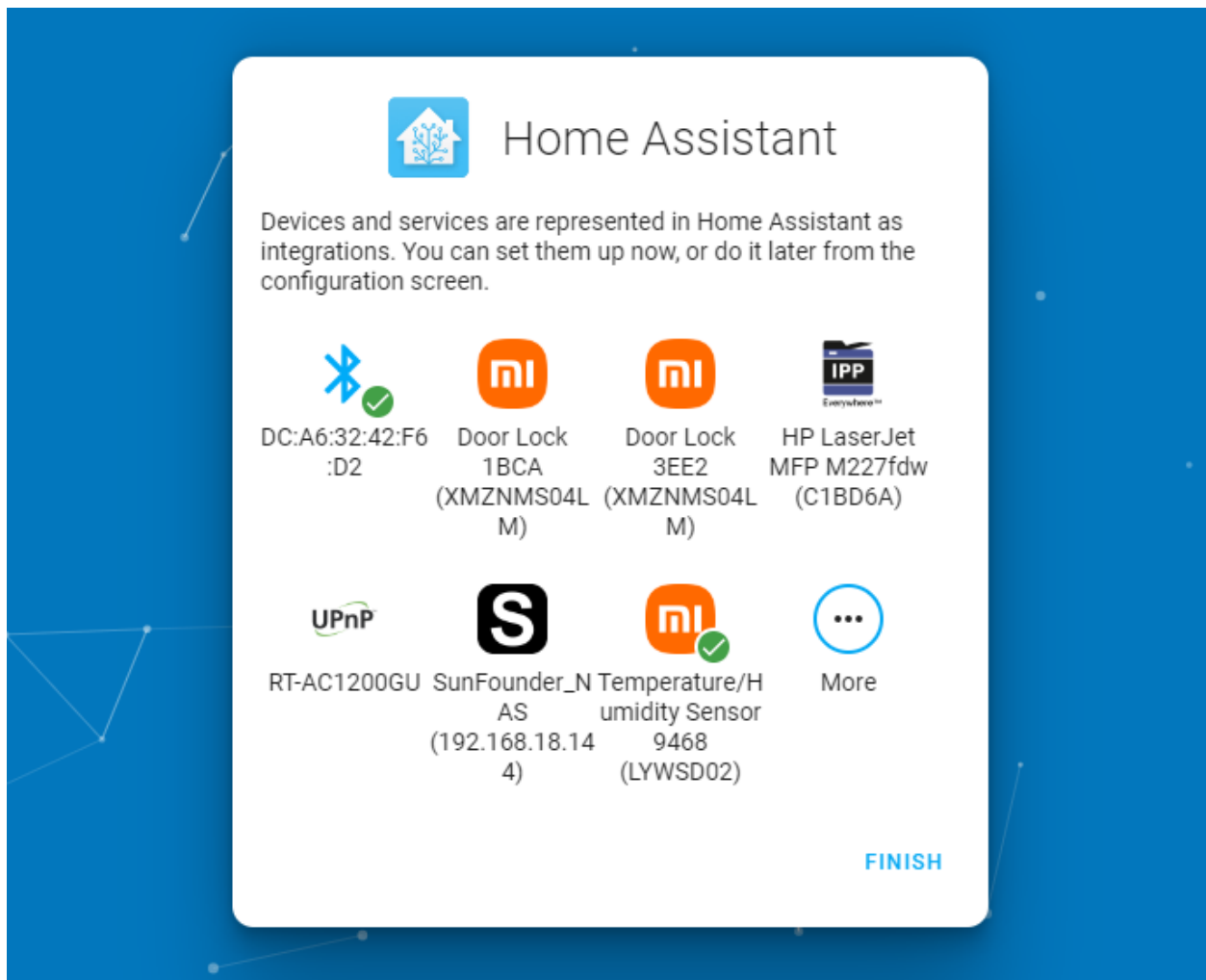
Password 

Confirm Password 

[Alternatively you can restore from a previous backup.](#)

CREATE ACCOUNT

The system will prompt you to install some detected devices, but for now, you can skip this by clicking FINISH.



Now you've set up Home Assistant.

Note: If you already have Home Assistant, please ignore.

Setup Your PiPower-Pro Card in Home Assistant:

4.2 Add PiPower Pro in Home Assistant

Step 1

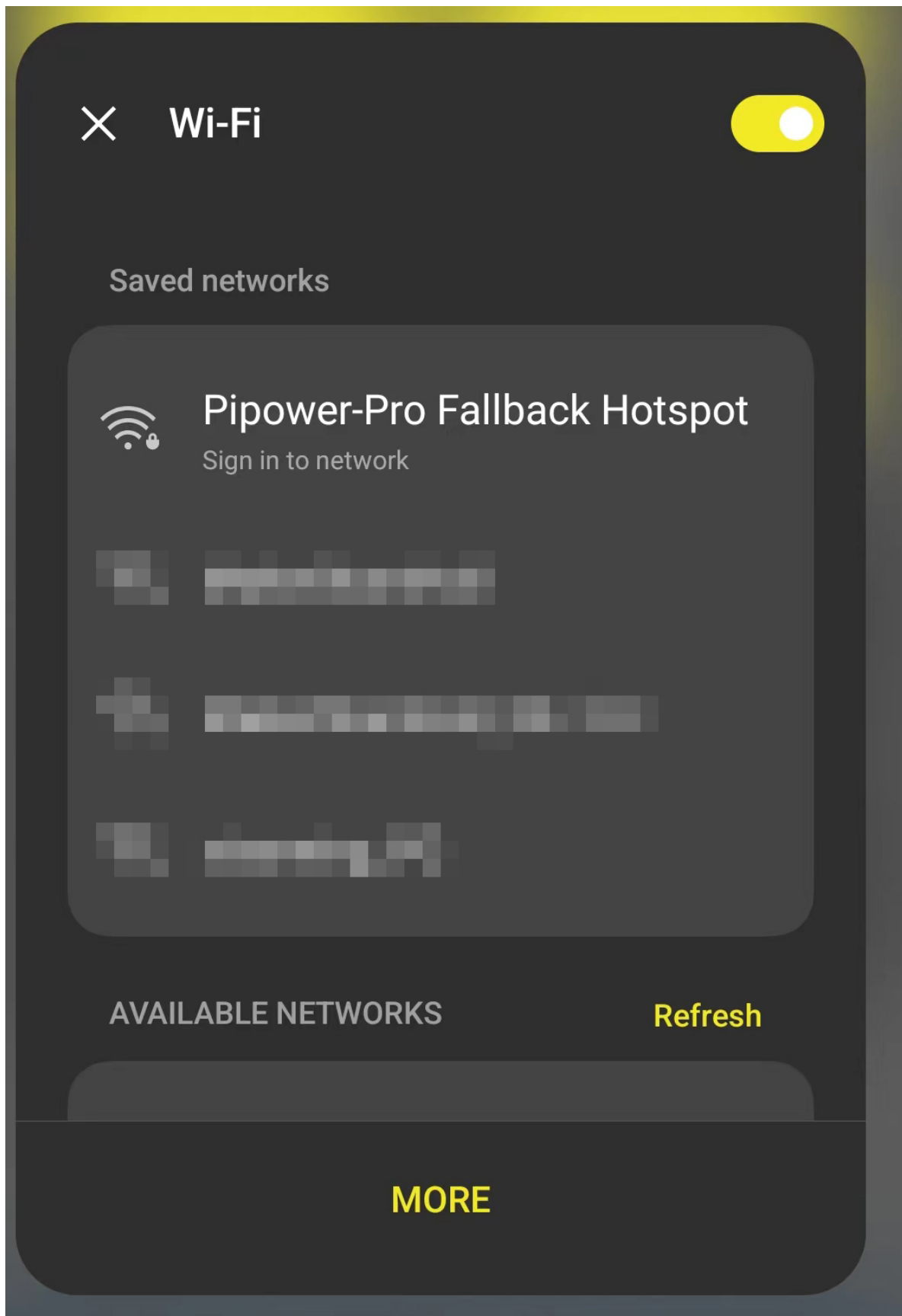
Install the battery.

Step 2

Connect the USB-C charger until all four battery indicators are lit (this means the battery is fully charged). Press the power button to turn it on.

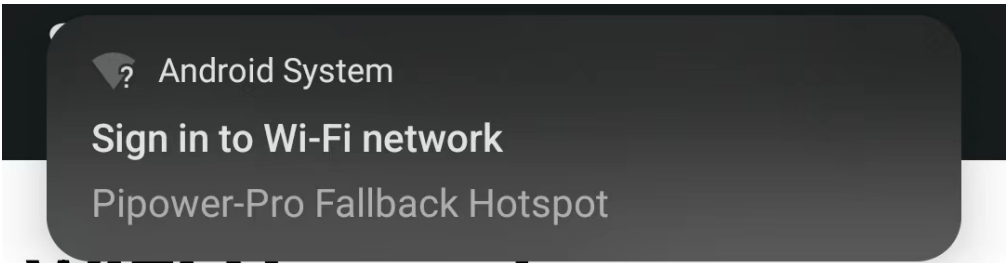
Step 3

Configure the network for PiPower Pro. Search for Wi-Fi on your phone (or other device) and connect to PiPower Pro Fallback Hotspot. The password is 12345678.



Step 4

Once connected, a configuration page will pop up on your phone. Complete the Wi-Fi configuration for PiPower here.



WiFi Networks: pipower-pro

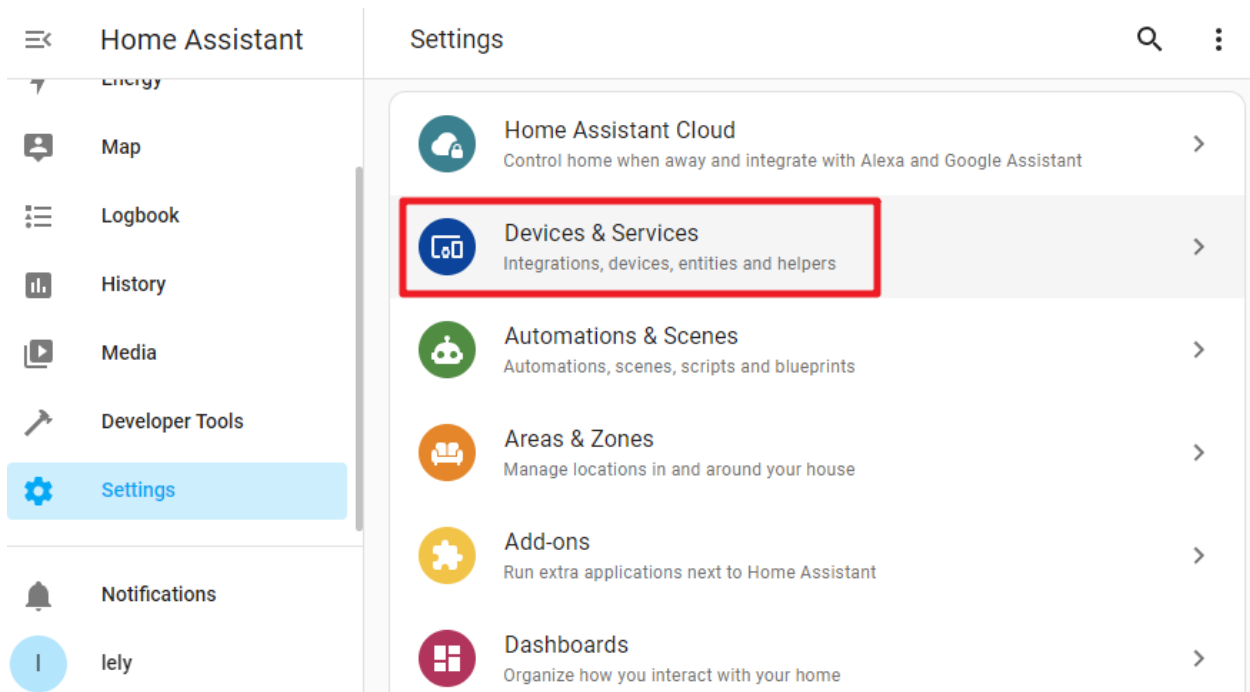


WiFi Settings

If the configuration page doesn't automatically pop up, you can open a browser and visit *pipower-pro.local*.

Step 5

Open your Home Assistant page, select Configuration from the left sidebar, then select Devices and Services.



Step 6

Click on + ADD INTEGRATION at the bottom right.



Step 7

Select ESPHome.

Select brand

Search for a brand name
ESPHome

ESPHome

**Step 8**

Enter `pipower-pro.local` and submit.

ESPHome



Please enter connection settings of your [ESPHome](#) node.

Host*

pipower-pro.local

Port

6053

SUBMIT**Step 9**

Choose an area for it and complete the setup.

Success!



Created configuration for PiPower-Pro.

We found the following devices:

PiPower-Pro
PiPower-Pro (SunFounder)

Area ▾

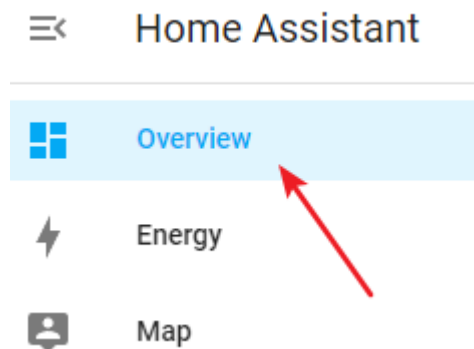
FINISH

Step 10

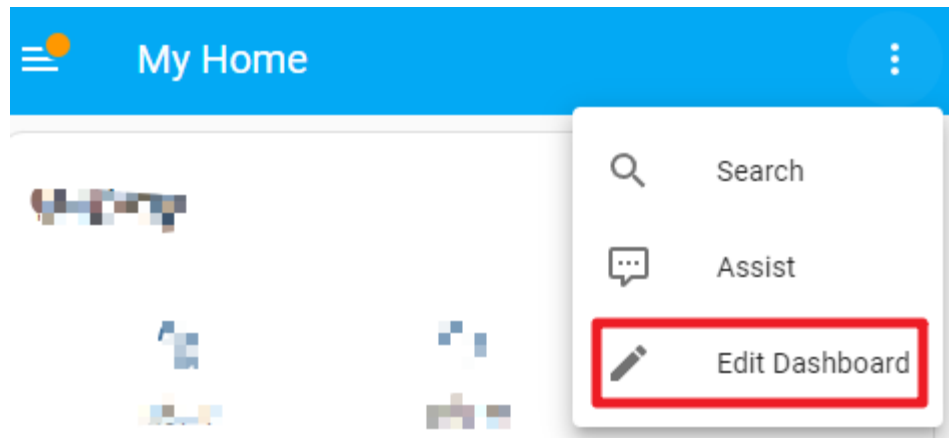
You have now successfully added PiPower Pro. You can add the PiPower Pro configurations you need on the dashboard.

4.3 Configure Dashboard

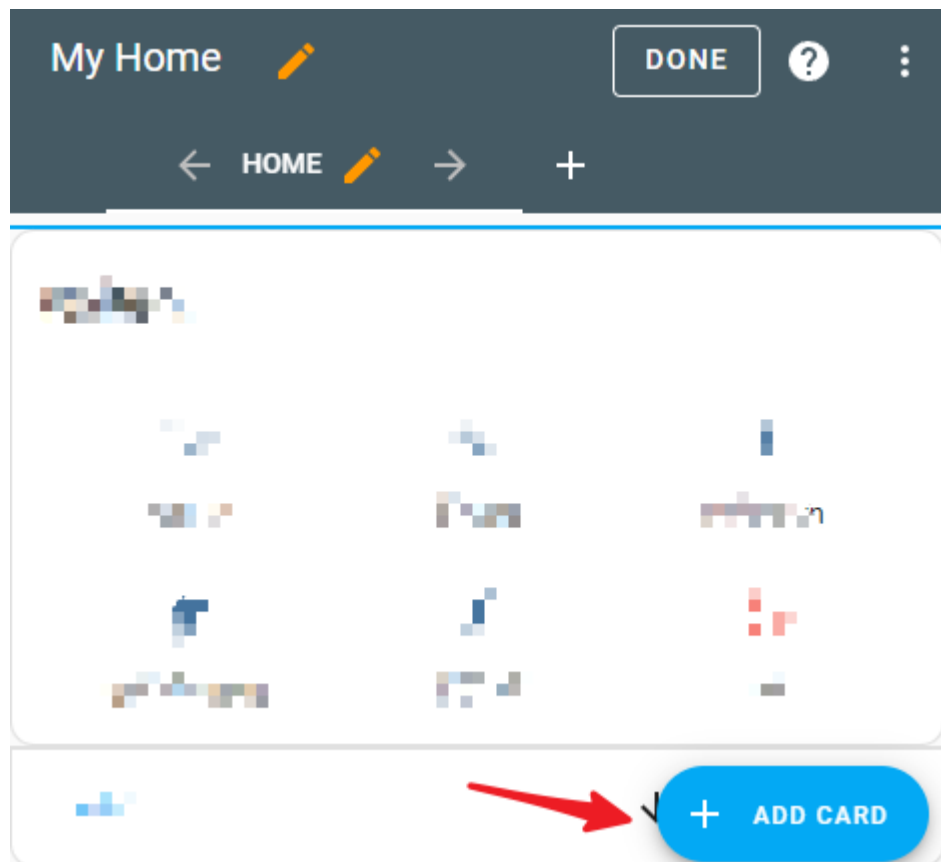
1. On the Home Assistant page, click **Overview** on the left sidebar, which leads you to the control page.



2. Click the menu button, then select **Edit Dashboard**.



3. Click on the bottom right to add a card, select the desired card from the options, configure it as needed, then save.



4.4 Add Card by Code Editor

1. After adding any card, you can manually edit the card's yaml file. Click on **SHOW CODE EDITOR** on the card editing page.

× Alarm Panel Card Configuration ?

Entity*
No matching entities found

Name
Theme (optional)

Available States

- ☒ Arm home
- ☒ Arm away
- ☐ Arm night
- ☐ Arm vacation
- ☐ Custom bypass

[SHOW CODE EDITOR](#) [CANCEL](#) [SAVE](#)

Invalid configuration

```
type: alarm-panel
states:
  - arm_home
  - arm_away
entity: ''
```

2. Then directly modify the yaml file. We provide some useful PiPower Pro configurations. Please copy the following yaml code directly into the box.

× Alarm Panel Card Configuration ?

```
1 type: alarm-panel
2 states:
3   - arm_home
4   - arm_away
5 entity: ''
6
```

Invalid configuration

```
type: alarm-panel
states:
  - arm_home
  - arm_away
entity: ''
```

[SHOW VISUAL EDITOR](#) [CANCEL](#) [SAVE](#)

× Vertical Stack Card Configuration



```

1 type: vertical-stack
2 cards:
3   - type: entities
4     entities:
5       - entity: switch.pipower_pro_output_switch
6       - entity: sensor.pipower_pro_output_source
7       - entity: binary_sensor.pipower_pro_external_power
8       - entity: sensor.pipower_pro_battery_voltage
9       - entity: sensor.pipower_pro_output_voltage
10    title: PiPower Pro
11    show_header_toggle: true
12    state_color: true
13  - square: true
14    type: grid
15    cards:
16      - type: gauge
17        entity: sensor.pipower_pro_battery_current
18        min: -2
19        max: 2
20        severity:
21          green: 0
22          yellow: 2
23          red: 2
24        needle: true
25        name: Battery Current
26      - type: gauge
27        entity: sensor.pipower_pro_output_current
28        min: 0
29        max: 3
30        severity:
31          green: 0
32          yellow: 2
33          red: 2.5
34        needle: true
35        name: Output Current

```

PiPower Pro

PiPower-Pro Output Switch

PiPower-Pro Output Source

PiPower-Pro External Power

PiPower-Pro Battery Voltage

PiPower-Pro Output Voltage

☒

☒

Battery

Plugged in

5.87 V

4.11 V

0.00 A

Battery Current

0.08 A

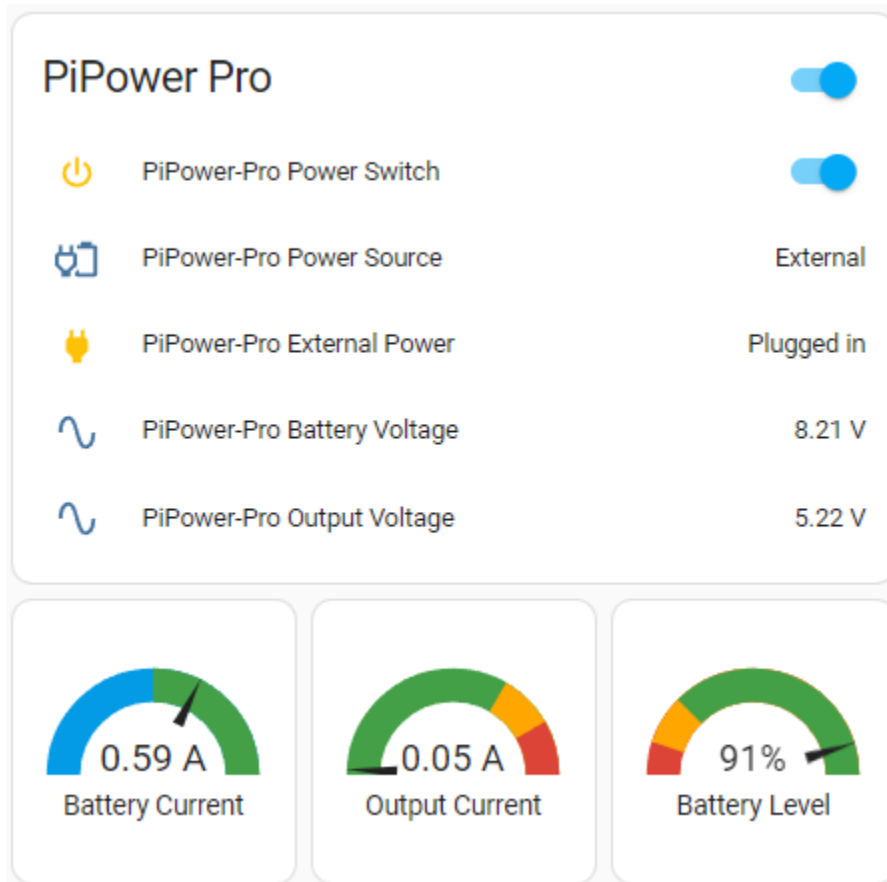
Output Current

0%

Battery Level

SHOW VISUAL EDITOR
CANCEL SAVE

Quick Overview



```
type: vertical-stack
cards:
  - type: entities
    entities:
      - entity: switch.pipower_pro_output_switch
      - entity: sensor.pipower_pro_output_source
      - entity: binary_sensor.pipower_pro_external_power
      - entity: sensor.pipower_pro_battery_voltage
      - entity: sensor.pipower_pro_output_voltage
    title: PiPower Pro
    show_header_toggle: true
    state_color: true
  - square: true
    type: grid
    cards:
      - type: gauge
        entity: sensor.pipower_pro_battery_current
        min: -2
        max: 2
        severity:
          green: 0
          yellow: 2
          red: 2
        needle: true
```

(continues on next page)

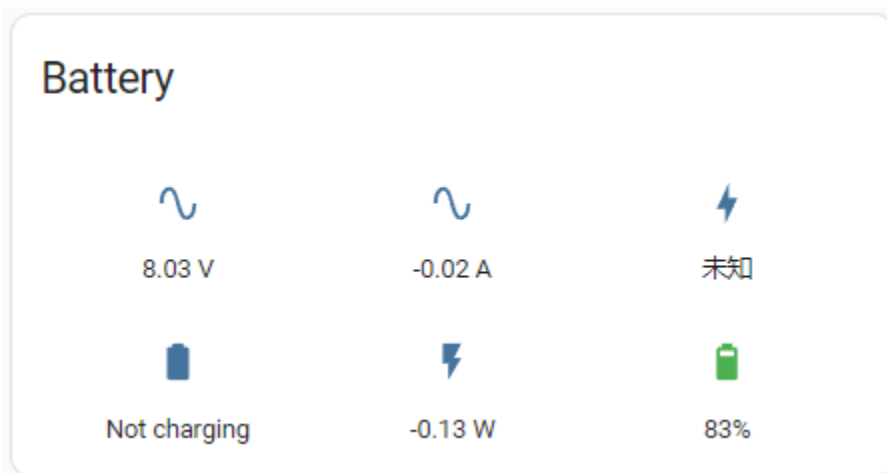
(continued from previous page)

```

    name: Battery Current
  - type: gauge
    entity: sensor.pipower_pro_output_current
    min: 0
    max: 3
    severity:
    green: 0
    yellow: 2
    red: 2.5
    needle: true
    name: Output Current
  - type: gauge
    entity: sensor.pipower_pro_battery_level
    name: Battery Level
    min: 0
    max: 100
    severity:
    green: 25
    yellow: 10
    red: 0
    needle: true
columns: 3

```

Battery Information



```

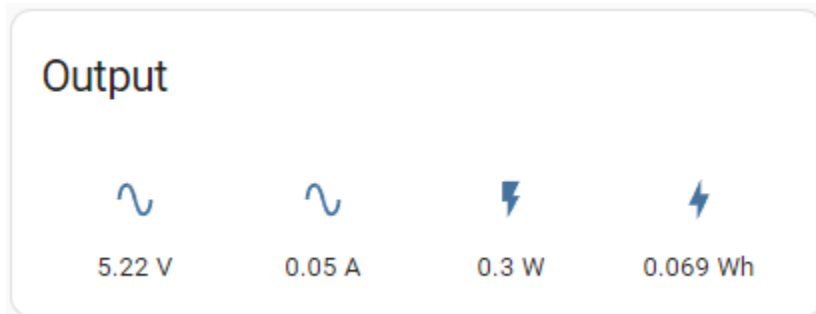
show_name: false
show_icon: true
show_state: true
type: glance
entities:
  - entity: sensor.pipower_pro_battery_voltage
  - entity: sensor.pipower_pro_battery_current
  - entity: sensor.pipower_pro_battery_capacity
  - entity: binary_sensor.pipower_pro_is_charging
  - entity: sensor.pipower_pro_battery_power
  - entity: sensor.pipower_pro_battery_level
title: Battery

```

(continues on next page)

(continued from previous page)

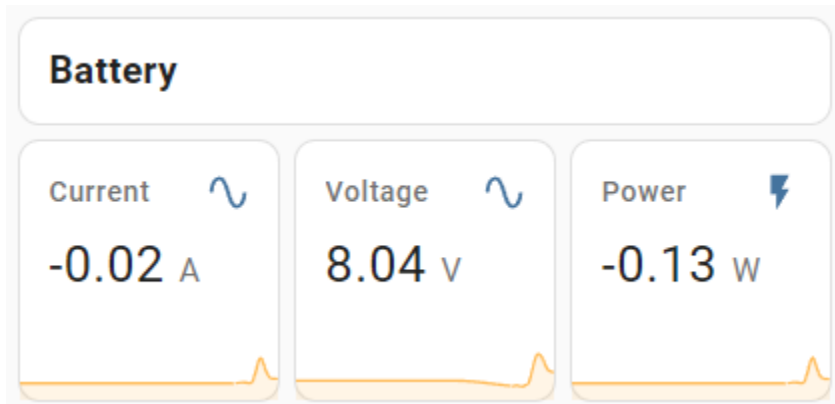
columns: 3

Output Information

```

show_name: false
show_icon: true
show_state: true
type: glance
entities:
  - entity: sensor.pipower_pro_output_voltage
  - entity: sensor.pipower_pro_output_current
  - entity: sensor.pipower_pro_output_power
  - entity: sensor.pipower_pro_output_energy
title: Output

```

Battery Chart

```

type: vertical-stack
cards:
  - type: markdown
    content: '## Battery'
  - square: true
    columns: 3
    type: grid
    cards:
      - hours_to_show: 12
        graph: line
        type: sensor

```

(continues on next page)

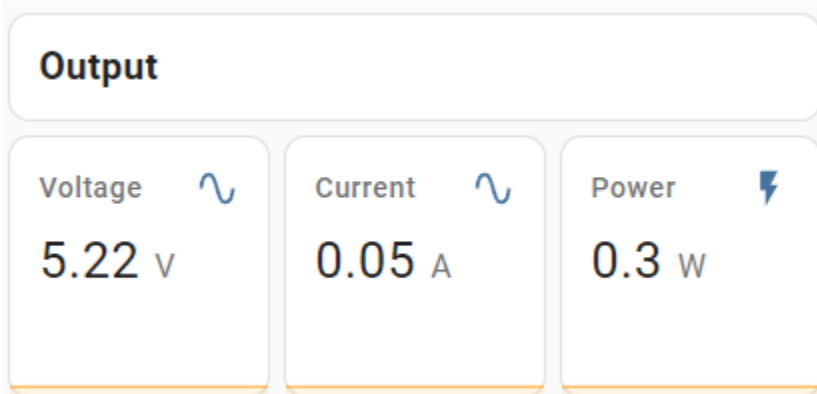
(continued from previous page)

```

    entity: sensor.pipower_pro_battery_current
    detail: 2
    name: Current
  - hours_to_show: 12
    graph: line
    type: sensor
    entity: sensor.pipower_pro_battery_voltage
    detail: 2
    name: Voltage
  - hours_to_show: 12
    graph: line
    type: sensor
    entity: sensor.pipower_pro_battery_power
    detail: 2
    name: Power

```

Output Chart



```

type: vertical-stack
cards:
  - type: markdown
    content: '## Output'
  - square: true
    columns: 3
    type: grid
    cards:
      - hours_to_show: 12
        graph: line
        type: sensor
        entity: sensor.pipower_pro_output_voltage
        detail: 2
        name: Voltage
      - hours_to_show: 12
        graph: line
        type: sensor
        entity: sensor.pipower_pro_output_current
        detail: 2
        name: Current
      - hours_to_show: 12

```

(continues on next page)

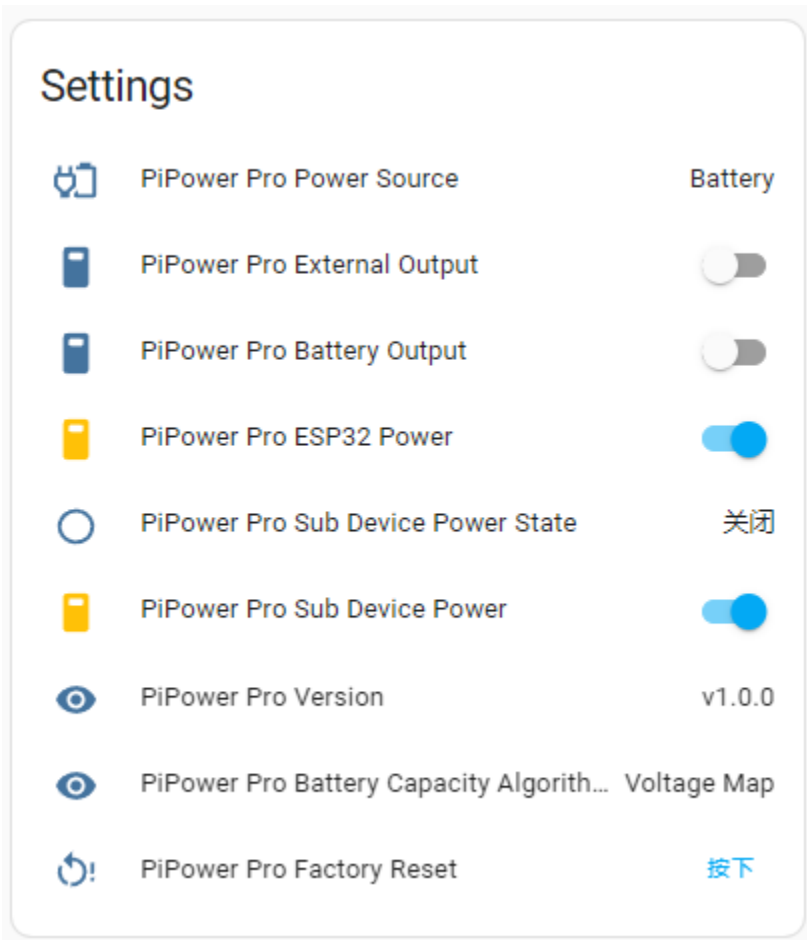
(continued from previous page)

```

graph: line
type: sensor
entity: sensor.pipower_pro_output_power
detail: 2
name: Power

```

Settings



```

type: entities
entities:
- entity: sensor.pipower_pro_input_voltage
- entity: sensor.pipower_pro_output_source
- entity: switch.pipower_pro_external_output
- entity: switch.pipower_pro_battery_output
- entity: switch.pipower_pro_esp32_power
- entity: binary_sensor.pipower_pro_sub_device_power_state
- entity: switch.pipower_pro_sub_device_power
- entity: sensor.pipower_pro_version
- entity: sensor.pipower_pro_battery_capacity_algorithm
- entity: button.pipower_pro_factory_reset
title: Settings
show_header_toggle: false

```

(continues on next page)

(continued from previous page)

```
state_color: true
```

4.5 PiPower Pro Entity

If you are familiar with Home Assistant and want to customize the Card yourself, here is a list of PiPower Pro entities you can use.

Basic Information

- `binary_sensor.pipower_pro_battery_low` - Battery low status (bool)
- `binary_sensor.pipower_pro_is_charging` - Charging status (V)

Switches

- `switch.pipower_pro_battery_output` - Battery output switch (bool)
- `switch.pipower_pro_esp32_power` - ESP32 power switch (bool)
- `switch.pipower_pro_external_output` - External output switch (bool)

Output

- `sensor.pipower_pro_output_voltage` - Output voltage (V)
- `sensor.pipower_pro_output_current` - Output current (A)
- `sensor.pipower_pro_output_power` - Output power (W)
- `sensor.pipower_pro_output_energy` - Output energy (Wh) used for calculating total output energy, can be reset via services, see all services for details

Battery

- `sensor.pipower_pro_battery_voltage` - Battery voltage (V)
- `sensor.pipower_pro_battery_current` - Battery current, positive for charging, negative for discharging (A)
- `sensor.pipower_pro_battery_power` - Battery output power (W)
- `sensor.pipower_pro_battery_capacity` - Battery capacity (mAh)
- `sensor.pipower_pro_battery_level` - Battery level (%)

Input

- `sensor.pipower_pro_input_voltage` - External input voltage (V)

Sub-device Control

- `switch.pipower_pro_sub_device_power` - Sub-device power control signal (bool)
- `binary_sensor.pipower_pro_sub_device_power_state` - Sub-device power state (bool)

Others

- `sensor.pipower_pro_battery_capacity_algorithm` - Battery capacity algorithm (String)
- `sensor.pipower_pro_power_source` - Current output source: Battery/External (String)
- `sensor.pipower_pro_battery_factory_capacity` - Battery factory nominal capacity (mAh)
- `binary_sensor.pipower_pro_external_power` - External input status (bool)
- `button.pipower_pro_factory_reset` - Factory reset button (bool)

- `update.pipower_pro_firmware` - Update firmware
- `switch.pipower_pro_power_switch` - Output switch (bool)
- `sensor.pipower_pro_version` - PiPower Pro version (String)

All Services

- `set_battery_factory_capacity` - Modify battery factory nominal capacity (capacity: int, mAh), default 2000
- `enable_coulomb_count_beta` - Enable Coulomb counting algorithm (enable: bool), default false
- `reset_capacity` - Reset current capacity to factory nominal capacity
- `reset_output_energy` - Reset output energy to 0
- `set_edv2` - Set End of Discharge Voltage 2, voltage for end-of-discharge calibration 2, default 6.8. See Coulomb count for details
- `set_edv1` - Set End of Discharge Voltage 1, voltage for end-of-discharge calibration 1, default 6.5. See Coulomb count for details
- `set_edv0` - Set End of Discharge Voltage 0, voltage for end-of-discharge calibration 0, default 6.2. See Coulomb count for details
- `set_rcv` - Set Reset Calibrate Voltage, voltage for reset calibration status, default 8.0. See Coulomb count for details
- `simulate_low_power` - Simulate low power for testing low power trigger scenarios

The Advanced Usage:

4.6 Setting up Safe Shutdown

PiPower Pro has two pins that are pre-configured to monitor the power status (referred to as sub-devices below) of connected devices, enabling remote power on, power off, and automatic safe shutdown when the battery is low.

Note: If the host running HassOS is set as a sub-device of PiPower Pro, it will also lose its functionality when the host shuts down, and remote power-on will not be possible.

- Pin 42 and sensor entity `binary_sensor.pipower_pro_sub_device_power_state` read the current state of the device.
- Pin 41 and entity `switch.pipower_pro_sub_device_power` control the sub-device power.

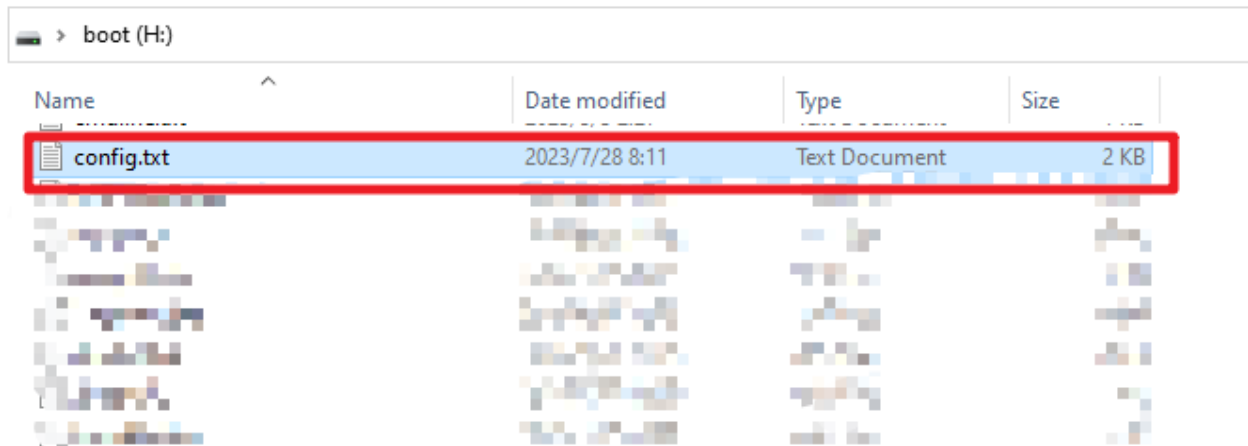
For example, let's use PiPower Pro as a UPS power source for a Raspberry Pi, monitoring its status and automatically shutting it down safely when external power is lost and the battery level is low.

Step 1

Configure the Raspberry Pi.

Set the two Raspberry Pi pins to the **Power Status Signal Pin** and **Shutdown Signal Pin** respectively. This can be done through devicetree.

Insert the SD card with the Raspberry Pi system into your computer. In the root directory of the boot partition, find `config.txt`.



| Name | Date modified | Type | Size |
|------------|----------------|---------------|------|
| config.txt | 2023/7/28 8:11 | Text Document | 2 KB |

Open it and add the following two lines at the end under [all].

```
dtoverlay=gpio-poweroff,gpiopin=17  
dtoverlay=gpio-shutdown,gpio_pin=18
```

- `gpio-poweroff` is the Raspberry Pi's power on/off status. After successful configuration, the Raspberry Pi will set this pin high when powered on and pull it low when powered off.
- `gpio-shutdown` controls the signal for shutting down the Raspberry Pi. After successful configuration, pulling this pin low will trigger the Raspberry Pi to shut down.

Step 2

- Connect PiPower Pro's pin 42 to the Raspberry Pi's `gpio-poweroff` pin, here using pin 17.
- Connect PiPower Pro's pin 41 to the Raspberry Pi's `gpio-shutdown` pin, here using pin 18.

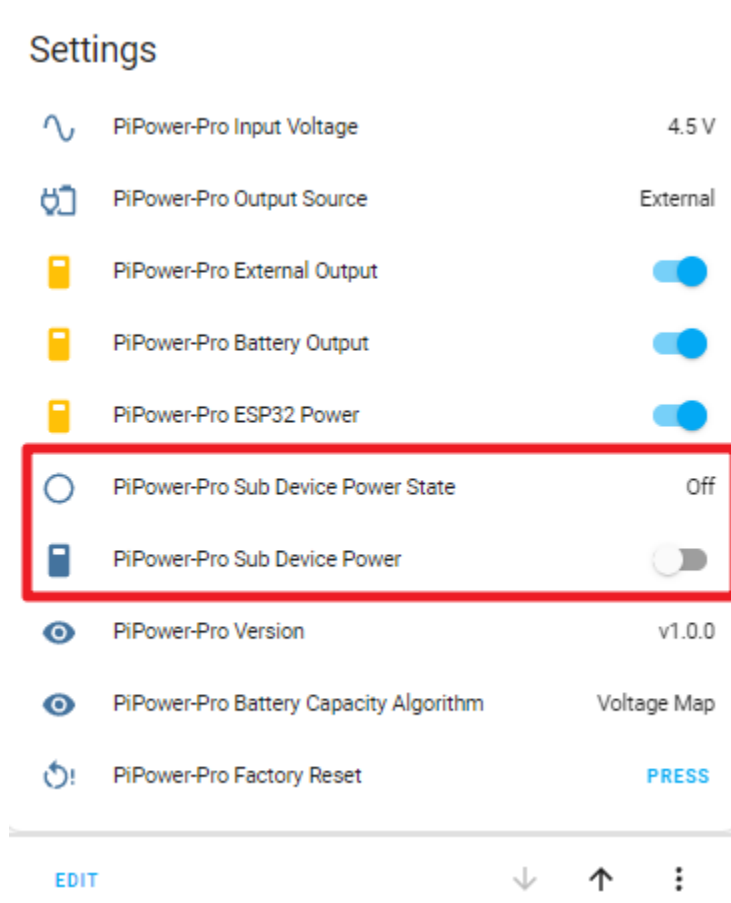
Step 3

Now test if the above two signals are working properly.

Add two entities to the Dashboard:

- `binary_sensor.pipower_pro_sub_device_power_state`
- `switch.pipower_pro_sub_device_power`

If you add the **Settings** Card (see [Add Card by Code Editor](#) for instructions on adding cards), these two entities will be included, labeled as PiPower-Pro Sub Device Power State and PiPower-Pro Sub Device Power.



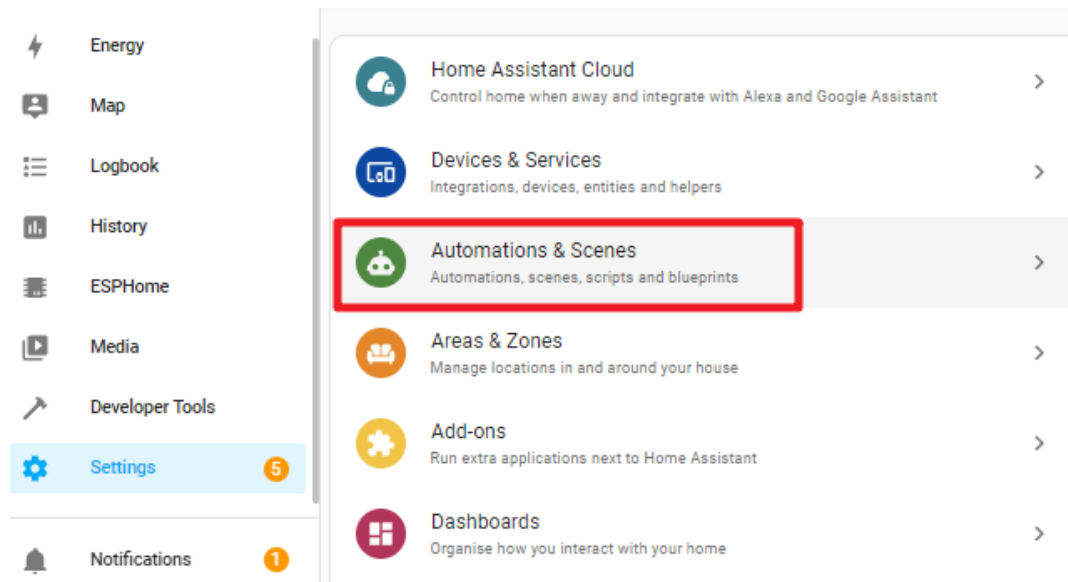
You can use the former to check if the Raspberry Pi is working and the latter to power off the Raspberry Pi.

Note: PiPower-Pro Sub Device Power can only power off the Raspberry Pi. To power it on, you still need to supply power to the Raspberry Pi (i.e., turn on the main switch on the **PiPower Pro Card**).

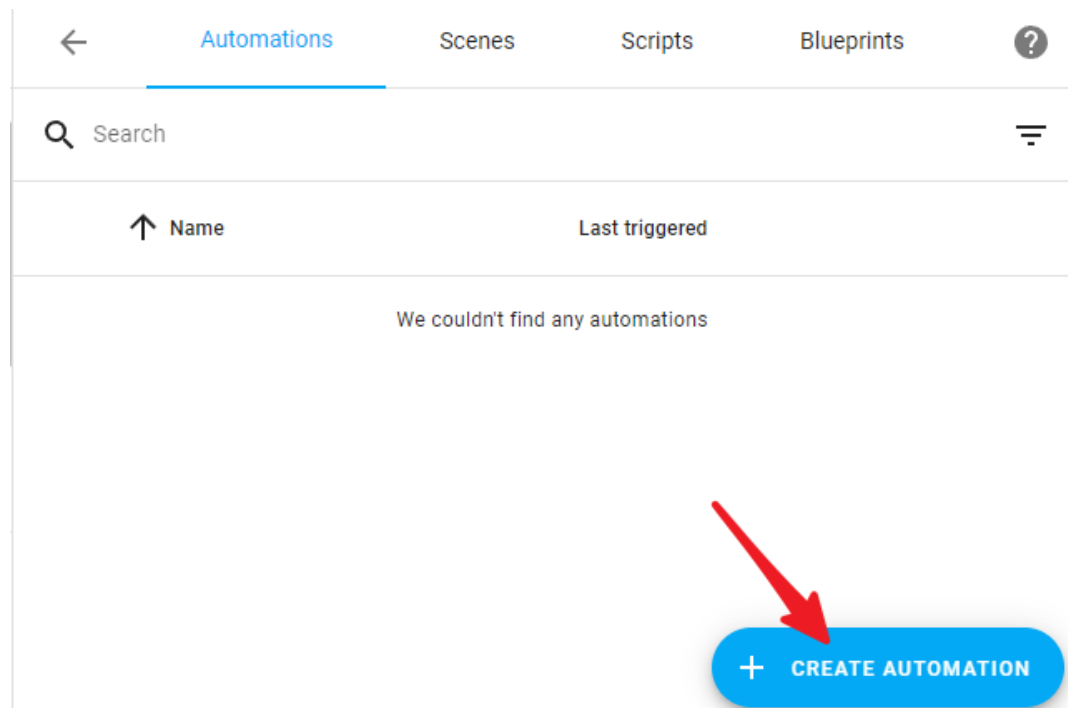
Step 4

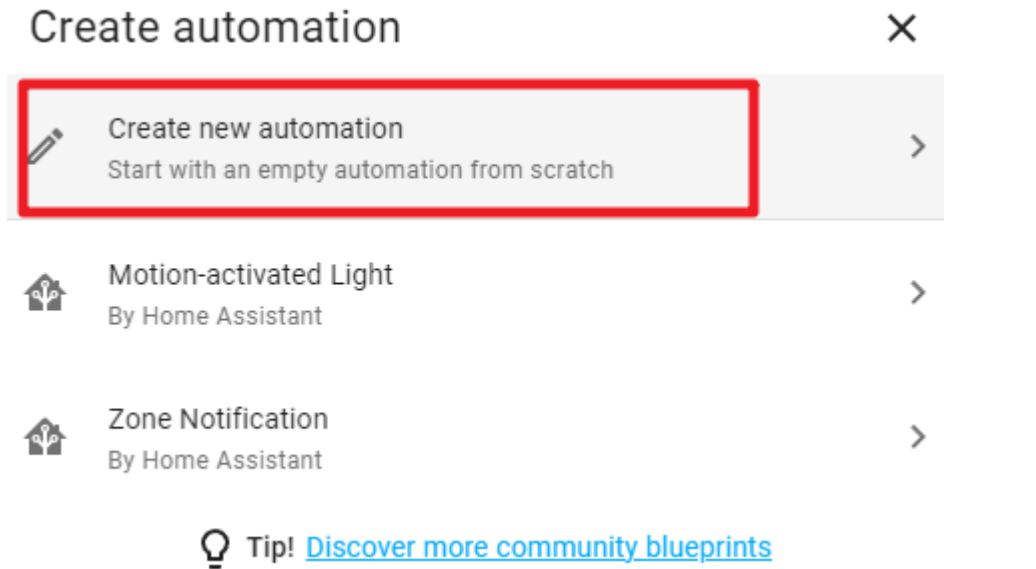
Next, configure automations to enable PiPower Pro to safely shut down the Raspberry Pi:

1. Open the Home Assistant configuration page, click on “Settings” on the left sidebar, and choose “Automations.”

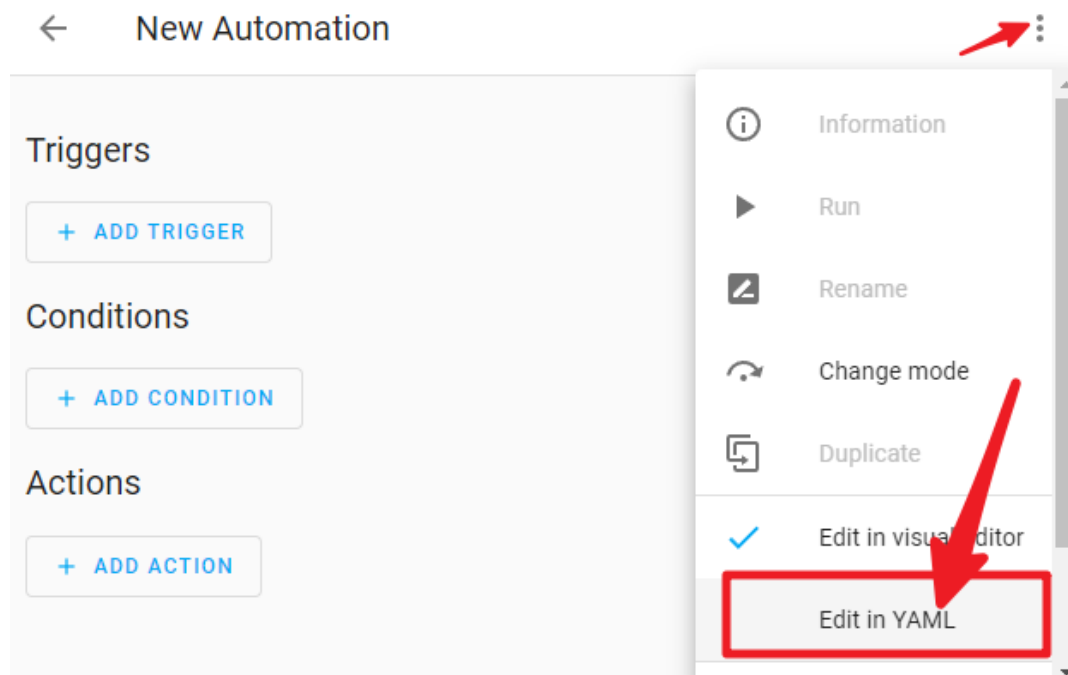


2. Create a new automation.





3. Click “Edit in YAML.”



4. Replace the existing code with the following code.

```
alias: Safe shutdown RPi
description: Turn off Raspberry Pi if no external power plug in and battery
  ↳ low
trigger:
  - platform: state
    entity_id:
      - binary_sensor.pipower_pro_external_power
    from: "on"
```

(continues on next page)

(continued from previous page)

```

    to: "off"
  - platform: numeric_state
    entity_id: sensor.pipower_pro_a03846_battery_level
    below: 25
condition:
  - condition: and
    conditions:
      - condition: state
        entity_id: binary_sensor.pipower_pro_a03846_external_power
        state: "off"
  - condition: and
    conditions:
      - condition: state
        entity_id: switch.pipower_pro_sub_device_power
        state: "on"
action:
  - type: turn_off
    device_id: a0ee4e356c85c4f69f765ed72baad129
    entity_id: switch.pipower_pro_sub_device_power
    domain: switch
mode: single

```

5. Click "Save."



6. Click "rename."

Rename



Name*

Safe shutdown RPi

Description

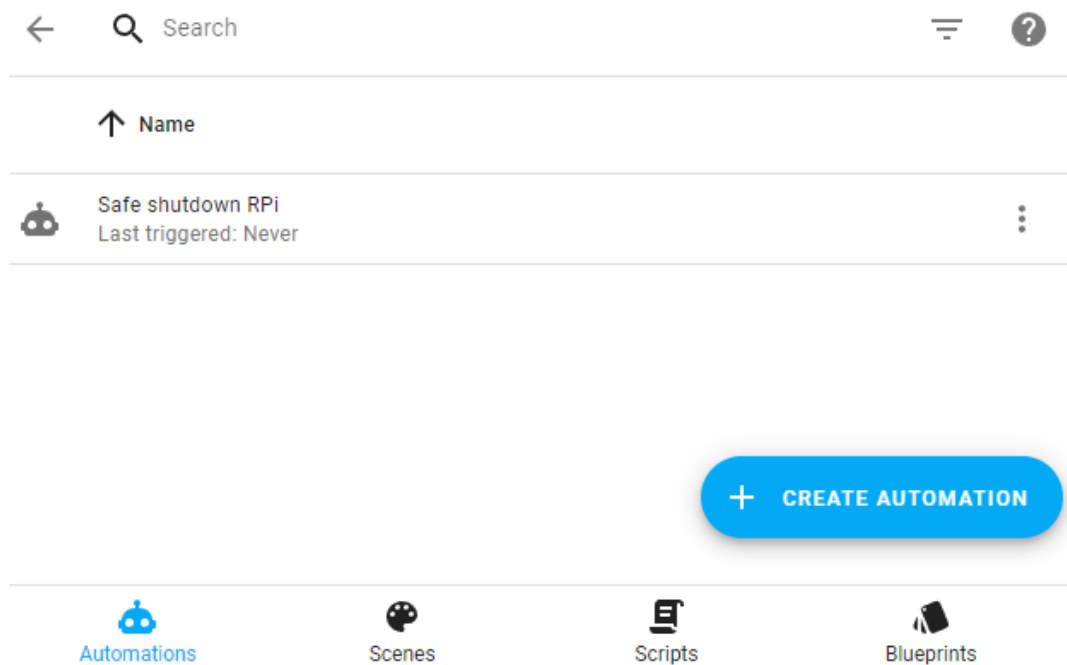
Turn off Raspberry Pi if no external power plug in and battery low

CANCEL

RENAME



7. Go back one level. Now you should see the newly set automation.



Note: We need to create a few more automations. Refer to the previous steps to complete them all.

Save Power

```
alias: Save Power
description: Turn off if raspberry pi power off
```

(continues on next page)

(continued from previous page)

```

trigger:
  - platform: state
    entity_id:
      - binary_sensor.pipower_pro_sub_device_power_state
    from: "on"
    to: "off"
condition:
  - condition: state
    entity_id: switch.pipower_pro_sub_device_power
    state: "off"
action:
  - delay:
    hours: 0
    minutes: 0
    seconds: 2
    milliseconds: 0
  - type: turn_off
    device_id: a0ee4e356c85c4f69f765ed72baad129
    entity_id: switch.pipower_pro_a03846_power_switch
    domain: switch
  - type: turn_off
    device_id: a0ee4e356c85c4f69f765ed72baad129
    entity_id: switch.pipower_pro_a03846_esp32_power
    domain: switch
mode: single

```

Sync Power Off RPi

```

alias: Sync Power Off RPi
description: Power Off Raspberry Pi is Switch Off
trigger:
  - platform: state
    entity_id:
      - switch.pipower_pro_a03846_power_switch
    from: "on"
    to: "off"
condition: []
action:
  - type: turn_off
    device_id: a0ee4e356c85c4f69f765ed72baad129
    entity_id: switch.pipower_pro_sub_device_power
    domain: switch
mode: single

```

Sync Power On RPi

```

alias: Sync Power On RPi
description: Power On Raspberry Pi is Switch On
trigger:
  - platform: state
    entity_id:
      - switch.pipower_pro_a03846_power_switch

```

(continues on next page)

(continued from previous page)

```

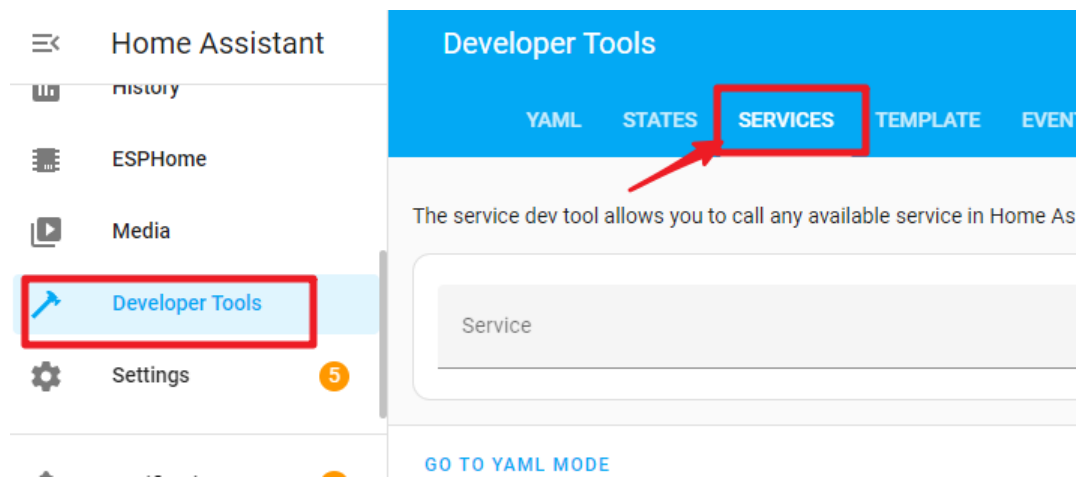
    from: "off"
    to: "on"
condition: []
action:
  - type: turn_on
    device_id: a0ee4e356c85c4f69f765ed72baad129
    entity_id: switch.pipower_pro_sub_device_power
    domain: switch
mode: single

```

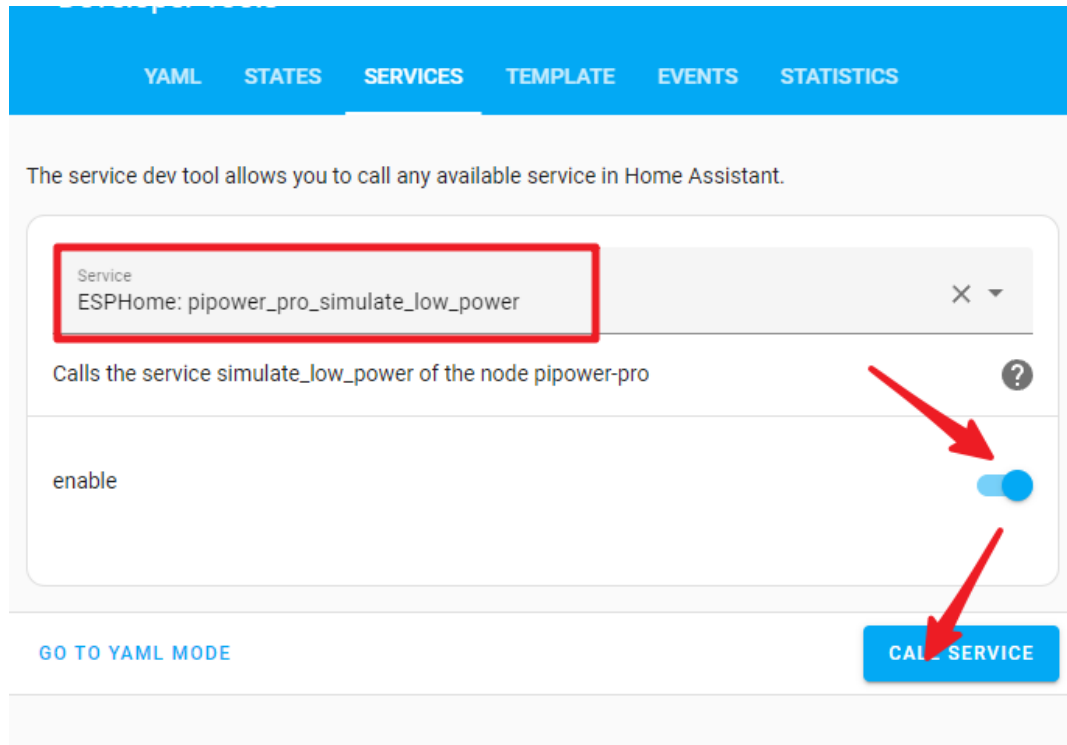
Step 5

We use a simulated low-power state to trigger the test:

1. Open the SERVICES interface in Developer Tools.



2. Find ESPHome: `pipower_pro_simulate_low_power`, enable it, and click the “Call Service” button.



You will see the PiPower battery light go off, and the battery level drop to 10% in the overview.

The Raspberry Pi will then shut down, and 2 seconds after it completes the shutdown, the PiPower Pro will power off, and the PWR light will go off.

4.7 Coulomb Counter (Beta)

The Coulomb Counter algorithm can improve the accuracy of battery capacity calculation, but it is currently in the beta stage and may result in serious inaccuracies. Please use it with caution.

Enable the Coulomb Counter

1. Go to the Home Assistant page and click on “Developer Tools” on the left sidebar.
2. In the Developer Tools page, select the “Services” tab.
3. In the list of services, choose ESPHome: `pipower_pro_enable_coulomb_count_beta`.
4. Turn on the switch for `enable_coulomb_count_beta`.
5. Click the **Call Service** button below.
6. You can check the currently selected battery capacity algorithm in the entity sensor. `pipower_pro_battery_capacity_algorithm`.

Algorithm

The Coulomb Counter algorithm calculates the energy by integrating the current and voltage measurements of the battery every second.

Capacity += Voltage * Current

Matching

The capacity calculated by this integration is only the charge/discharge energy from the current moment. To associate it with the actual capacity of the battery, a matching process is needed. The matching method here is simple. PiPower Pro's default battery capacity is the nominal capacity of the battery, which is 2000mAh. The actual battery capacity will be less than this value. As long as the battery is charged, the capacity will be set to the maximum of 2000mAh (can be changed using the service `set_battery_factory_capacity`), so when the battery is fully charged, the capacity value matches the actual battery capacity of 2000mAh, and the integration calculation value matches the actual battery capacity value.

Auto Calibration

Integration can accumulate errors, and the battery capacity will decrease as the battery is used over time, which may not reach the nominal 2000mAh capacity. Therefore, some calibration methods need to be used to calibrate the battery capacity.

Here, the Compensated End of Discharge Voltage (CEDV) calibration method is used. The principle of the CEDV calibration method is that the voltage at the end of the battery discharge is relatively accurate, and the voltage curve at this time is also the steepest. Using this voltage as a calibration point is more appropriate. So here we set 3 EDV points: `edv2` (7%), `edv1` (3%), and `edv0` (0%).

After setting these 3 calibration voltages, when the battery is discharged to these 3 points, PiPower Pro will calibrate the battery: $\text{MaxCapacity} = \text{MaxCapacity} - \text{Capacity} + \text{MaxCapacity} * 7\%$ To avoid unlimited calibration at the same point due to voltage fluctuations, calibration is limited to once before charging reaches RCV (Reset Calibration Voltage, default 8.0V). Both `edv2`, `edv1`, `edv0`, and `rcv` can be configured in the service `Service`, see [PiPower Pro Entity](#) for details.

Indicator

When the Coulomb Counter algorithm is enabled, the battery indicator will also switch to the Coulomb Counter mode. However, there is a small chance of incorrect battery level readings or even the battery level resetting.

The relationship between the battery indicators and power is as follows:

- 4 LEDs all on: 75%
- 3 LEDs on: 50%
- 2 LEDs on: 25%
- 1 LED on: 10%
- 4 LEDs all off: 0%, batteries need to be charged.

4.8 Custom Development

If you find that the basic functionality of PiPower Pro is not enough for your needs, you can perform custom development on PiPower Pro.

All software for PiPower Pro is open source. Below is the basic tutorial and preparation for custom development.

1. **Open the developer mode of Home Assistant.**
 - a. Open the Home Assistant management page.
 - b. Select "Configuration" in the lower-left corner.
2. **Install ESPHome.**
 - a. Open the Home Assistant management page.
 - b. Select "Configuration" in the lower-left corner.
 - c. Select "Add-ons."

- d. Click “Add” button.
 - e. Search for “esphome.”
 - f. Click “Install.”
 - g. After installation, click “Start.”
 - h. Select “Add to Sidebar.”
3. **Create a new device.**
 - a. Click “ESPHome” in the sidebar to enter the ESPHome management page.
 - b. Select “New Device.”
 - c. Enter the device name, such as “PiPower Pro.”
 - d. For the first configuration, you also need to enter the Wi-Fi account and password.
 - e. Select “ESP32 S2.”
 - f. Confirm and skip the installation.
4. **Configure the new device.**
 - a. Select the device you just created and click “Edit” to enter the YAML editing page.
 - b. At the bottom, add the PiPower Pro template:

```
packages:
  remote_package: github://sunfounder/pipower-pro/pipower-pro-
    template.yaml@main
```

- c. Click “Install” in the upper right corner to install it on PiPower Pro.

4.9 Multiple PiPower Pro Units

If you have multiple PiPower Pro units to use in the same Home Assistant environment, you need to modify the YAML settings. Add `name_add_mac_suffix: true` under “esphome.”

```
esphome:
  name: pipower-pro
  friendly_name: PiPower-Pro
  name_add_mac_suffix: true
```

4.10 IO Expansion

J4 is used for expansion. The IO comes from ESP32 S2.

Table 1: IO Expansion

| Functions | Pin | Pin | Functions |
|-----------------|-----|-----|------------------|
| 5V | 5V | 3V3 | 3V3 |
| ADC,Touch,GPIO8 | 8 | GND | Ground |
| ADC,Touch,GPIO9 | 9 | 10 | GPIO10,Touch,ADC |
| ADC,DAC,GPIO18 | 18 | 36 | GPIO36 |
| GPIO37 | 37 | 38 | GPIO38 |
| GPIO39 | 39 | 40 | GPIO40 |
| GPIO41 | 41 | 42 | GPIO42 |

5.1 PiPower Pro not working?

When you put the battery in for the first time or when the battery is unplugged and put in again, the battery will not work properly.

This is because when the battery is removed, due to the mechanism of the on-board over-discharge protection circuit, the voltage will be considered too low, thus activating the protection circuit;

At this time, you need to plug the **Type C** cable into the charging port to release the protection circuit, and the battery can be used normally.

5.2 Which single-board computers can PiPower Pro be used on?

PiPower Pro compatible single-board computers are shown below.

Note: Functionally Compatible means that it can be powered by PiPower Pro normally.
