Adafruit HUZZAH32 - ESP32 Feather

Created by lady ada


https://learn.adafruit.com/adafruit-huzzah32-esp32-feather
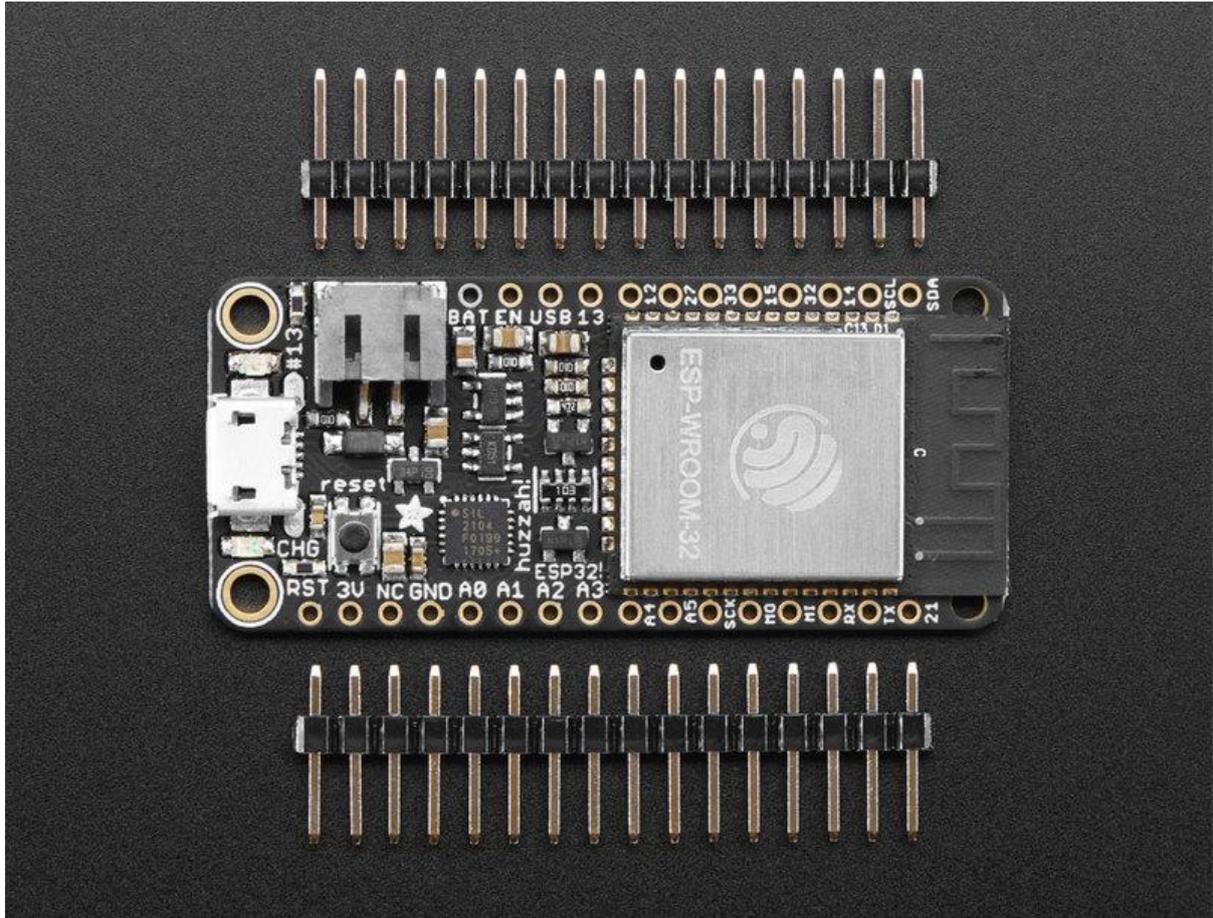
Last updated on 2022-07-26 02:35:21 PM EDT

Table of Contents

Overview

Aww yeah, it's the Feather you have been waiting for! The HUZZAH32 is our ESP32-based Feather, made with the official WROOM32 module. We packed everything you love about Feathers: built in USB-to-Serial converter, automatic bootloader reset, Lithium Ion/Polymer charger, and all the GPIO brought out so you can use it with any of our Feather Wings.

That module nestled in at the end of this Feather contains a dual-core ESP32 chip, 4 MB of SPI Flash, tuned antenna, and all the passives you need to take advantage of this powerful new processor. The ESP32 has both WiFi and Bluetooth Classic/LE support. That means it's perfect for just about any wireless or Internet-connected project.
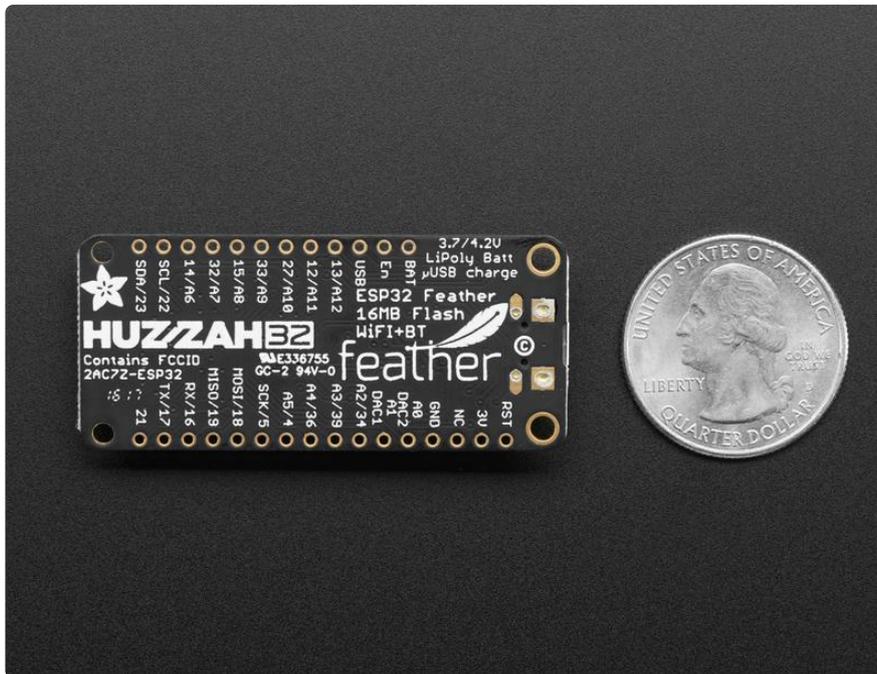
Because it is part of our Feather eco-system you can take advantage of the 50+ Wings (https://adafru.it/wev) that we've designed, to add all sorts of cool accessories

The ESP32 is a perfect upgrade from the ESP8266 that has been so popular. In comparison, the ESP32 has way more GPIO, plenty of analog inputs, two analog outputs, multiple extra peripherals (like a spare UART), two cores so you don't have to yield to the WiFi manager, much higher-speed processor, etc. etc! We think that as the ESP32 gets traction, we'll see more people move to this chip exclusively, as it is so full-featured.
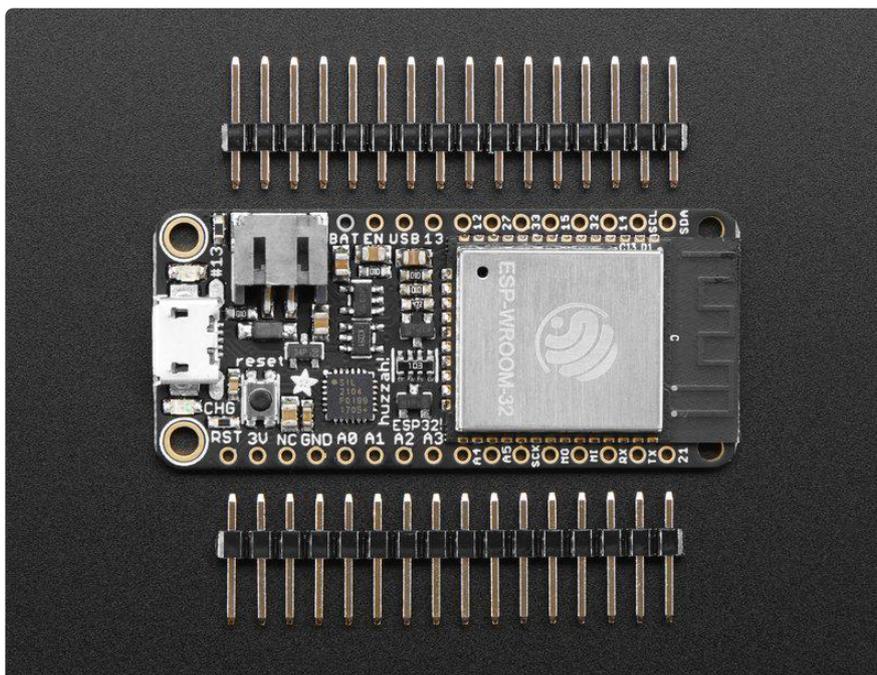
Please note: The ESP32 is still targeted to developers. Not all of the peripherals are fully documented with example code, and there are some bugs still being found and fixed. We got all of our Featherwings working under Arduino IDE, so you can expect things like I2C and SPI and analog reads to work. But other elements are still under development. For that reason, we recommend this Feather for makers who have some experience with microcontroller programming, and not as a first dev board.

Here are specifications from Espressif about the ESP32 (https://adafru.it/wew)

- 240 MHz dual core Tensilica LX6 microcontroller with 600 DMIPS
- Integrated 520 KB SRAM
- Integrated 802.11b/g/n HT40 Wi-Fi transceiver, baseband, stack and LWIP
- Integrated dual mode Bluetooth (classic and BLE)
- 4 MByte flash
- On-board PCB antenna
- Ultra-low noise analog amplifier
- Hall sensor
- 10x capacitive touch interface
- 32 kHz crystal oscillator
- 3 x UARTs (only two are configured by default in the Feather Arduino IDE support, one UART is used for bootloading/debug)
- 3 x SPI (only one is configured by default in the Feather Arduino IDE support)
- 2 x I2C (only one is configured by default in the Feather Arduino IDE support)
- 12 x ADC input channels
- 2 x I2S Audio
- 2 x DAC
- PWM/timer input/output available on every GPIO pin
- OpenOCD debug interface with 32 kB TRAX buffer
- SDIO main/secondary 50 MHz
- SD-card interface support
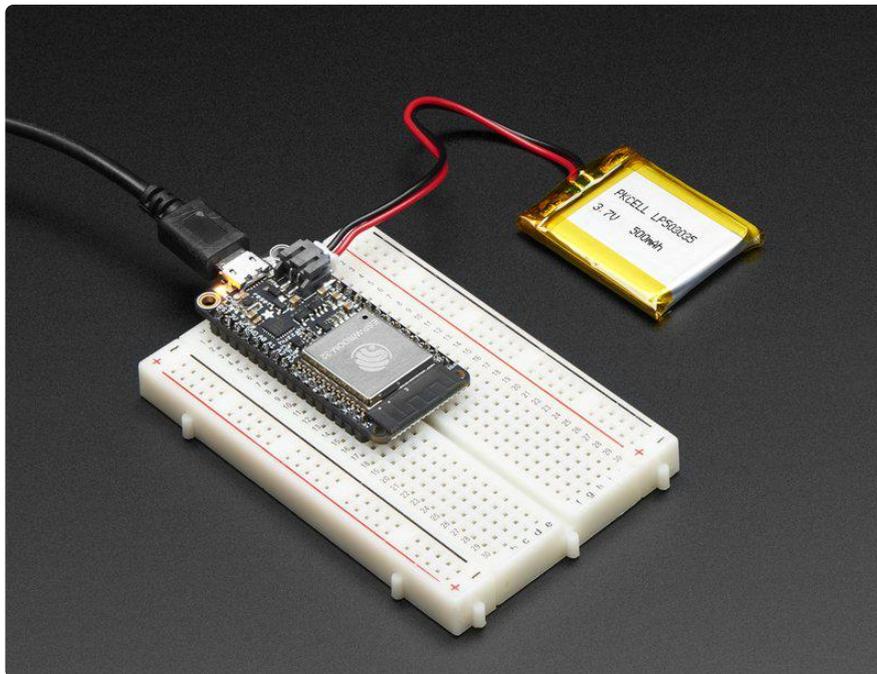
Comes fully assembled and tested, with a USB interface that lets you quickly use it with the Arduino IDE or the low-level ESP32 IDF. We also toss in some header so you can solder it in and plug into a solderless breadboard.
Lipoly battery and USB cable not included (but we do have lots of options in the shop if you'd like!)

Pinouts
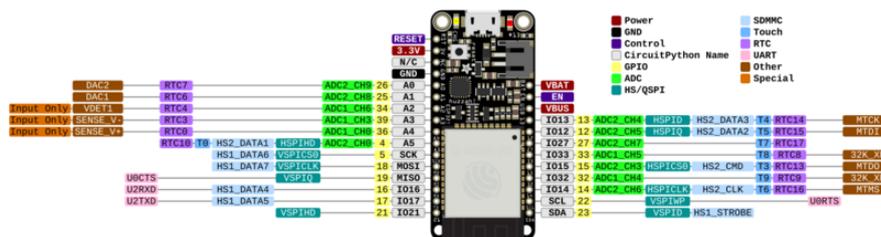
[Click here to view a PDF version of the pinout diagram](https://adafru.it/ZKA) (https://adafru.it/ZKA)

One of the great things about the ESP32 is that it has tons more GPIO than the ESP8266. You won't have to juggle or multiplex your IO pins! There's a few things to watch out for so please read through the pinouts carefully
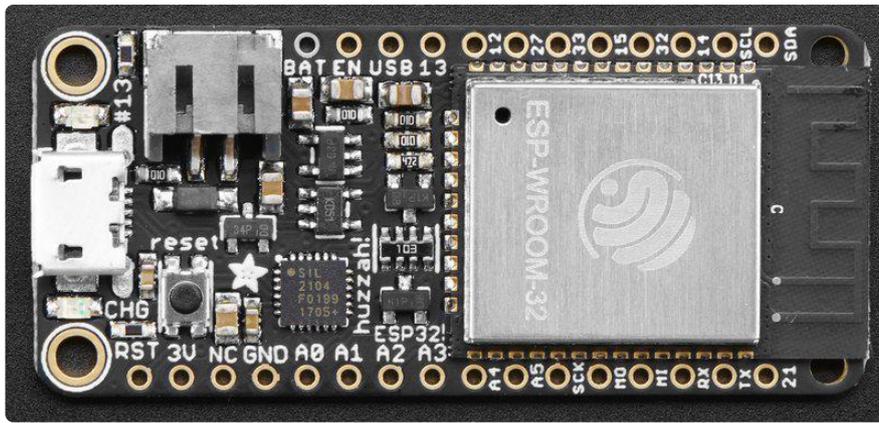
Power Pins
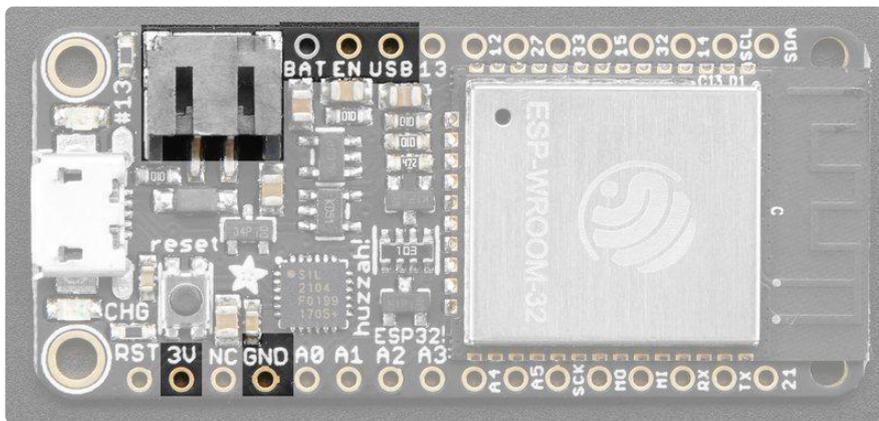
- GND - this is the common ground for all power and logic
- BAT - this is the positive voltage to/from the JST jack for the optional Lipoly battery
- USB - this is the positive voltage to/from the micro USB jack if connected
- EN - this is the 3.3V regulator's enable pin. It's pulled up, so connect to ground to disable the 3.3V regulator
- 3V - this is the output from the 3.3V regulator. The regulator can supply 500mA peak but half of that is drawn by the ESP32, and it's a fairly power-hungry chip.

So if you need a ton of power for stuff like LEDs, motors, etc. Use the USB or BAT pins, and an additional regulator

Logic pins

This is the general purpose I/O pin set for the microcontroller. All logic is 3.3V

The ESP32 runs on 3.3V power and logic, and unless otherwise specified, GPIO pins are not 5V safe!

Serial pins

RX and TX are the additional Serial1 pins, and are not connected to the USB/Serial converter. That means you can use them to connect to UART-devices like GPS's, fingerprint sensors, etc.
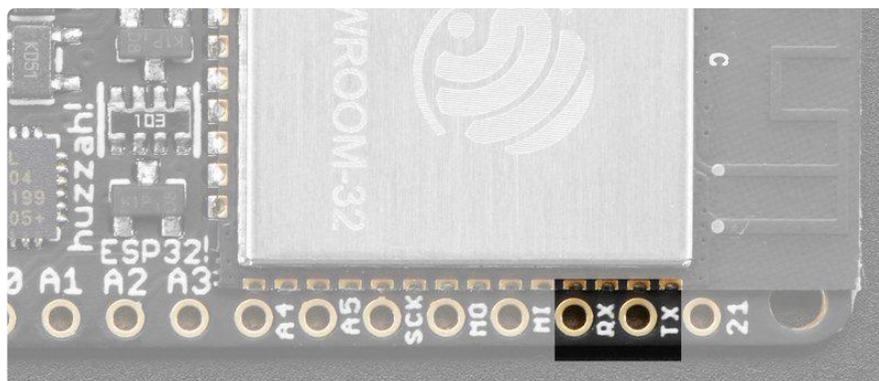
The TX pin is the output from the module. The RX pin is the input into the module. Both are 3.3V logic

I2C & SPI pins

You can use the ESP32 to control I2C and SPI devices, sensors, outputs, etc. If using with Arduino, the standard Wire and SPI devices work as you'd expect!
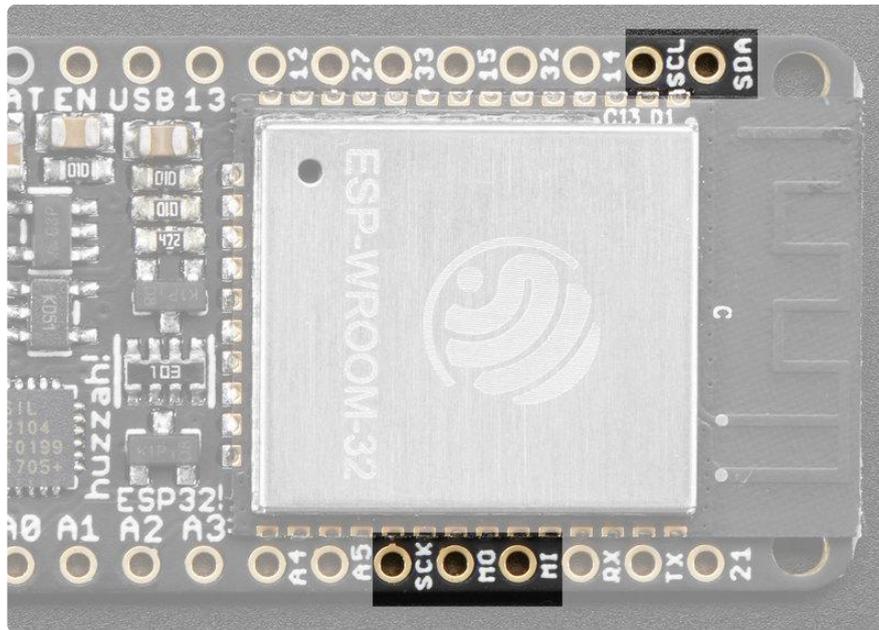
Note that the I2C pins do not have pullup resistors already! You must add them if you want to communicate with an I2C device

GPIO & Analog Pins
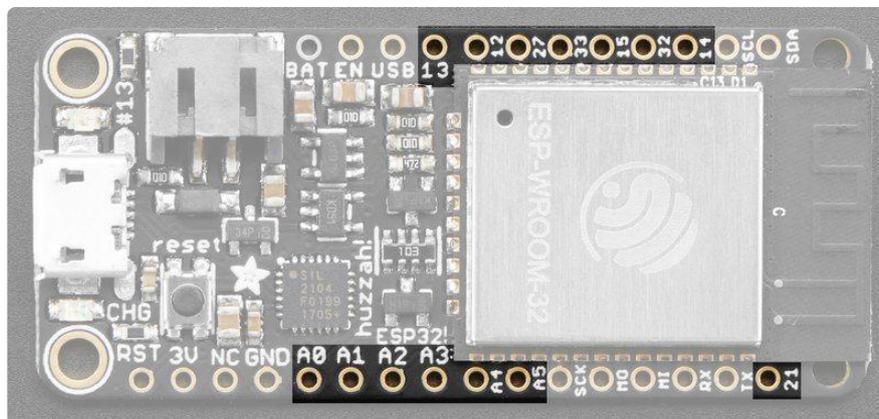
There are tons of GPIO and analog inputs available to you for connecting LEDs, buttons, switches, sensors, etc. Here's the remaining pins available.

Bottom row:

- A0 - this is an analog input A0 and also an analog output DAC2. It can also be used as a GPIO #26. It uses ADC #2
- A1 - this is an analog input A1 and also an analog output DAC1. It can also be used as a GPIO #25. It uses ADC #2
- A2 - this is an analog input A2 and also GPIO #34. Note it is not an output-capable pin! It uses ADC #1

- A3 - this is an analog input A3 and also GPIO #39. Note it is not an output-capable pin! It uses ADC #1
- A4 - this is an analog input A4 and also GPIO #36. Note it is not an output-capable pin! It uses ADC #1
- A5 - this is an analog input A5 and also GPIO #4. It uses ADC #2
- 21 - General purpose IO pin #21

Top row:

- 13 - This is GPIO #13 and also an analog input A12 on ADC #2. It's also connected to the red LED next to the USB port
- 12 - This is GPIO #12 and also an analog input A11 on ADC #2. This pin has a pull-down resistor built into it, we recommend using it as an output only, or making sure that the pull-down is not affected during boot.
- 27 - This is GPIO #27 and also an analog input A10 on ADC #2
- 33 - This is GPIO #33 and also an analog input A9 on ADC #1. It can also be used to connect a 32 KHz crystal.
- 15 - This is GPIO #15 and also an analog input A8 on ADC #2
- 32 - This is GPIO #32 and also an analog input A7 on ADC #1. It can also be used to connect a 32 KHz crystal.
- 14 - This is GPIO #14 and also an analog input A6 on ADC #2

There's also an external analog input

- A13 - This is general purpose input #35 and also an analog input A13, which is a resistor divider connected to the VBAT line

Note you can only read analog inputs on ADC #2 once WiFi has started as it is shared with the WiFi module.

Assembly

We ship Feathers fully tested but without headers attached - this gives you the most flexibility on choosing how to use and configure your Feather

Header Options!

Before you go gung-ho on soldering, there's a few options to consider!
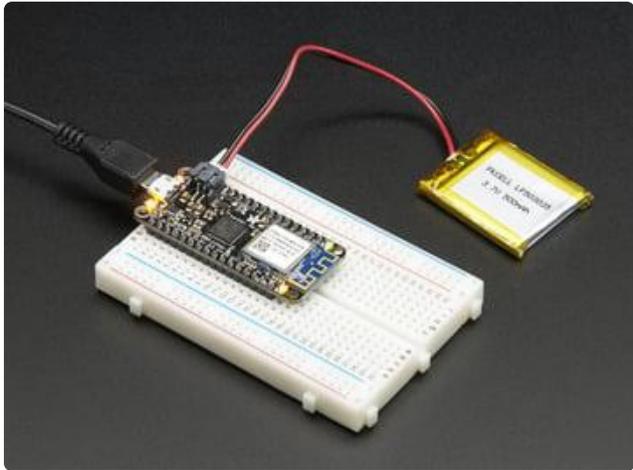
The first option is soldering in plain male headers, this lets you plug in the Feather into a solderless breadboard
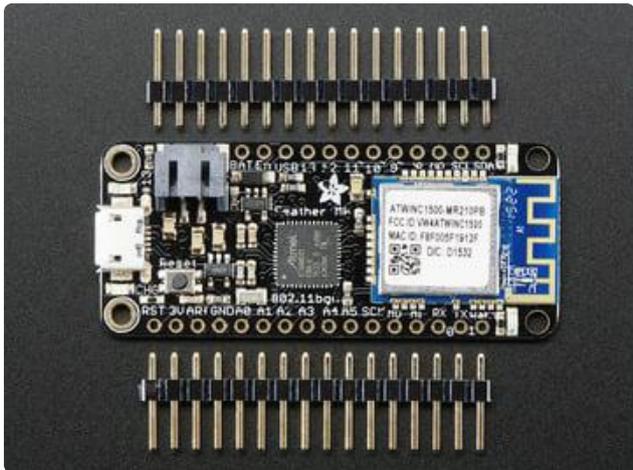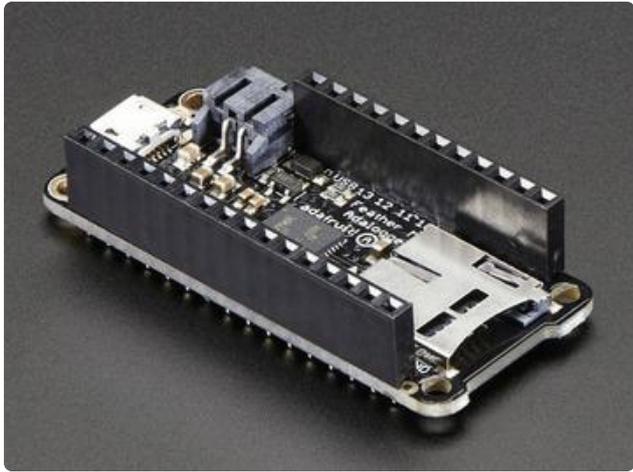
Another option is to go with socket female headers. This won't let you plug the Feather into a breadboard but it will let you attach featherwings very easily
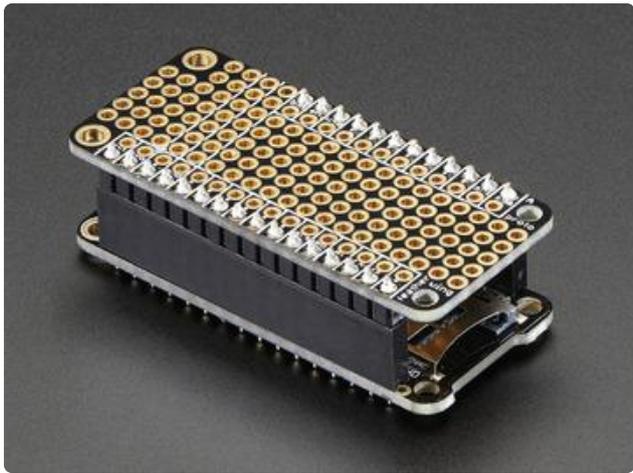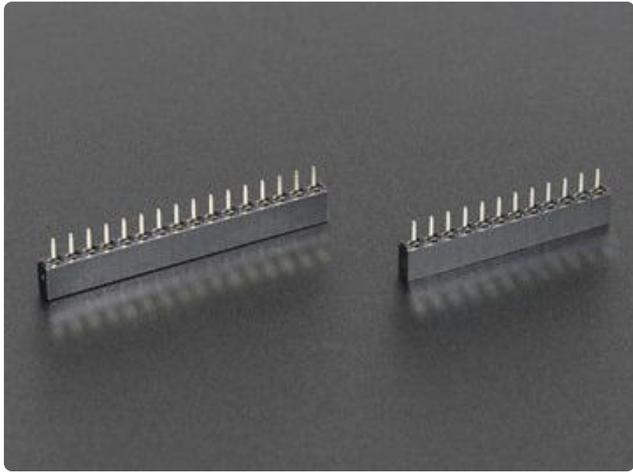
We also have 'slim' versions of the female headers, that are a little shorter and give a more compact shape
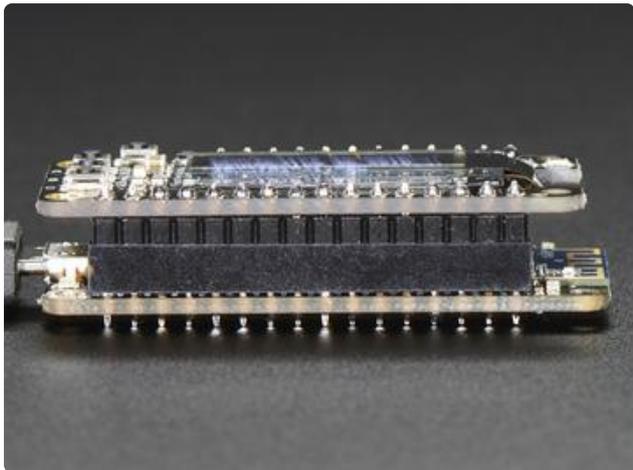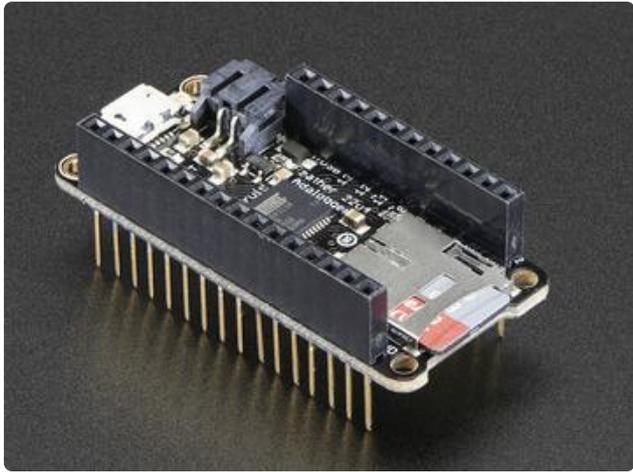
Finally, there's the "Stacking Header" option. This one is sort of the best-of-both-worlds. You get the ability to plug into a solderless breadboard and plug a featherwing on top. But its a little bulky
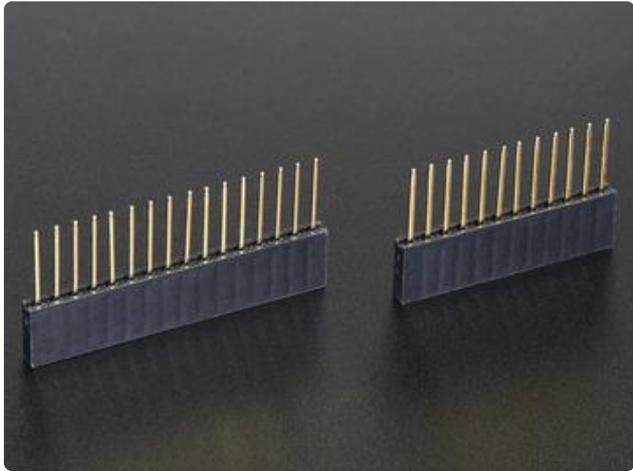
Soldering in Plain Headers
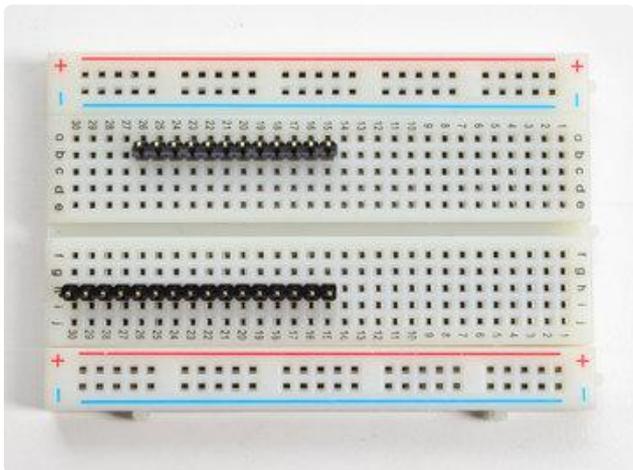
Prepare the header strip:
Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - long pins down
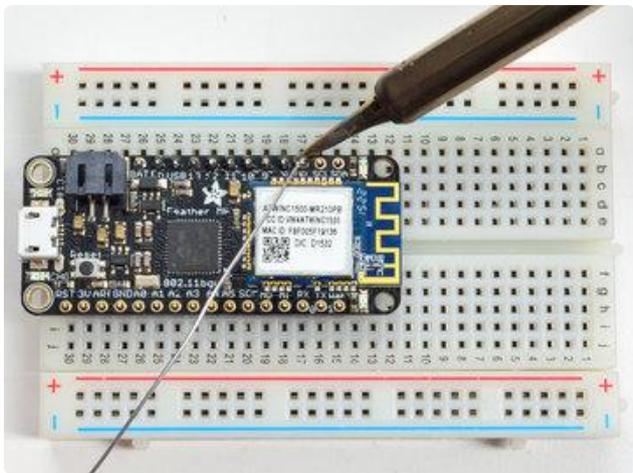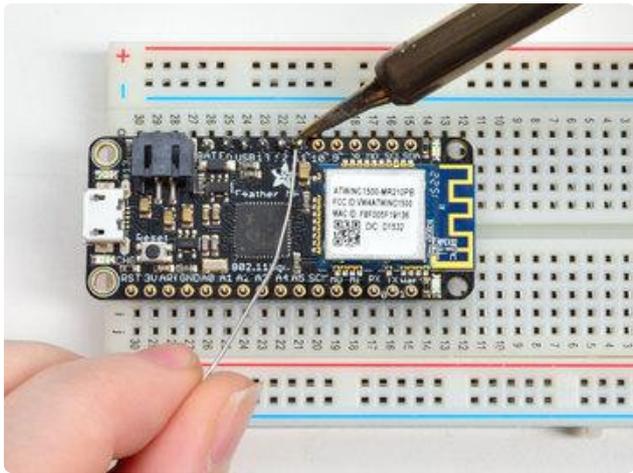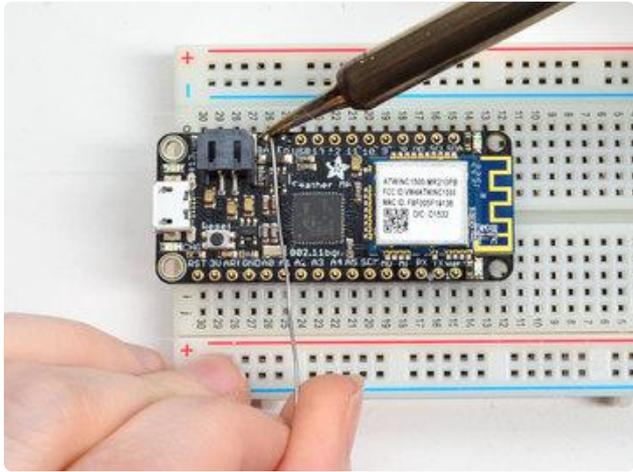
Add the breakout board:
Place the breakout board over the pins so that the short pins poke through the breakout pads

And Solder!
Be sure to solder all pins for reliable electrical contact.

(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk)).
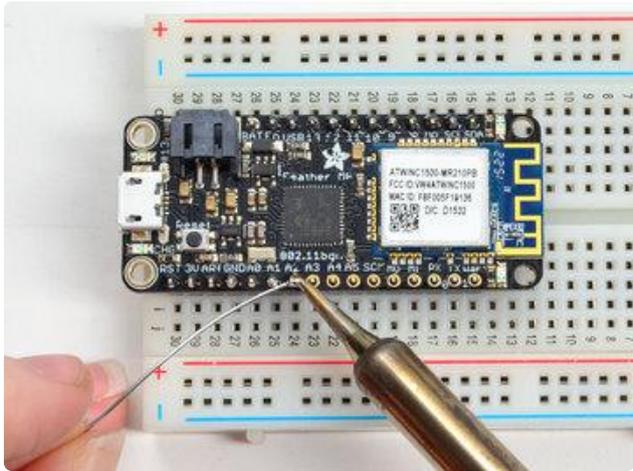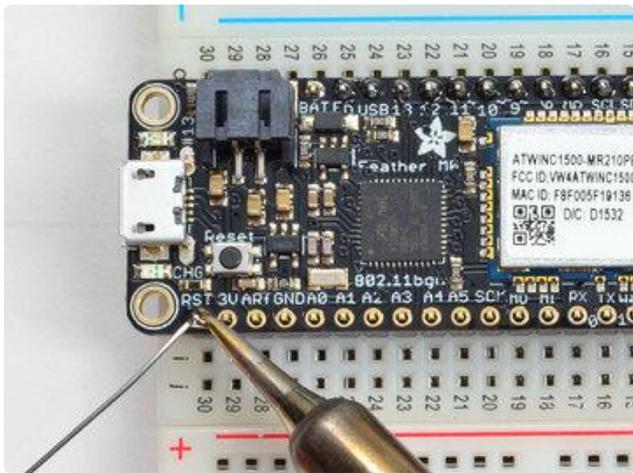
Solder the other strip as well.
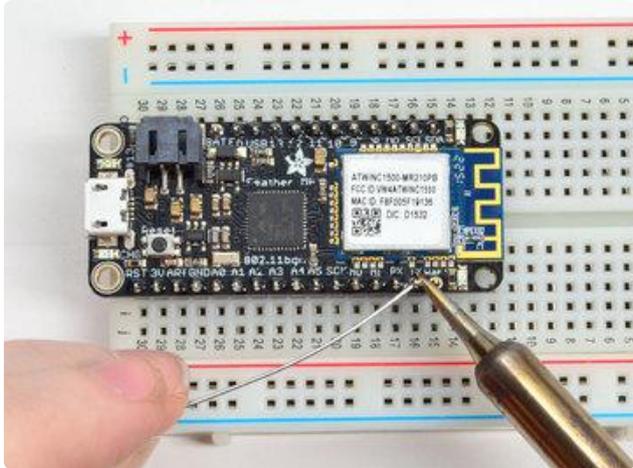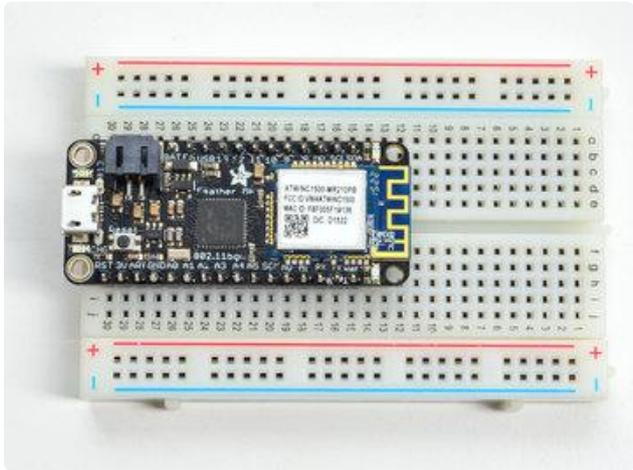
You're done! Check your solder joints visually and continue onto the next steps

Soldering on Female Header

Tape In Place

For sockets you'll want to tape them in place so when you flip over the board they don't fall out
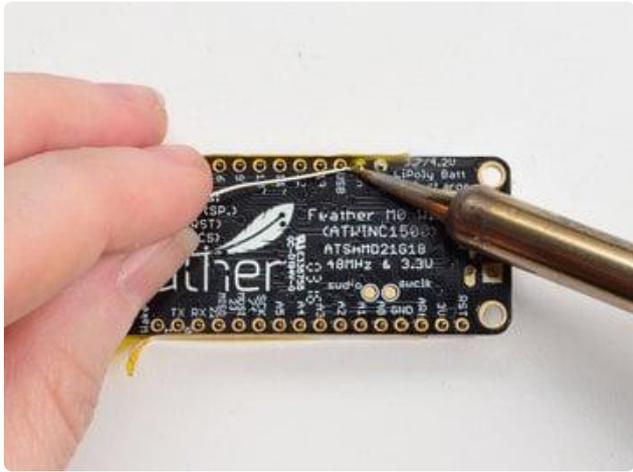
Flip & Tack Solder

After flipping over, solder one or two points on each strip, to 'tack' the header in place
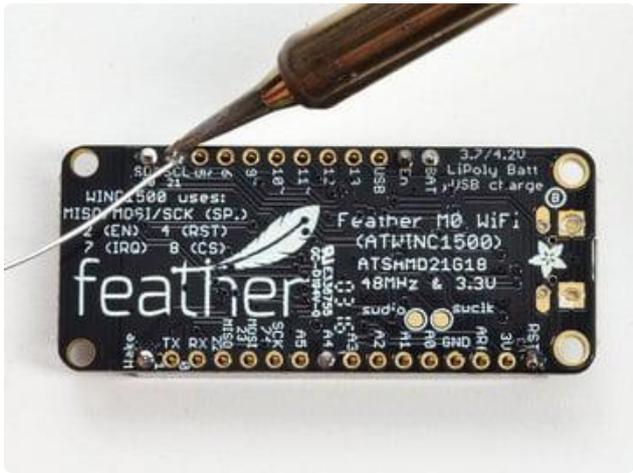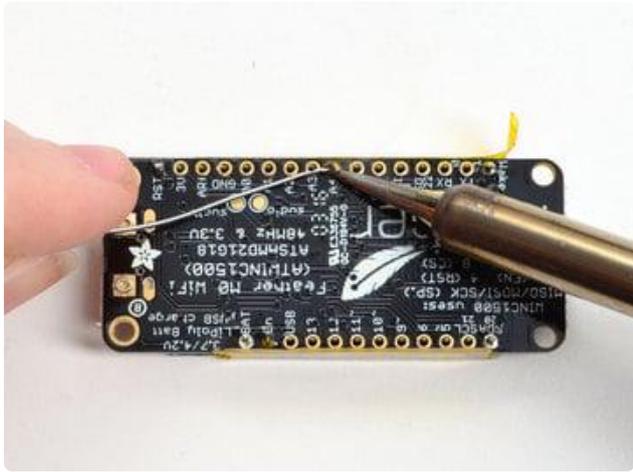
And Solder!

Be sure to solder all pins for reliable electrical contact.

(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk)).

You're done! Check your solder joints visually and continue onto the next steps


Power Management
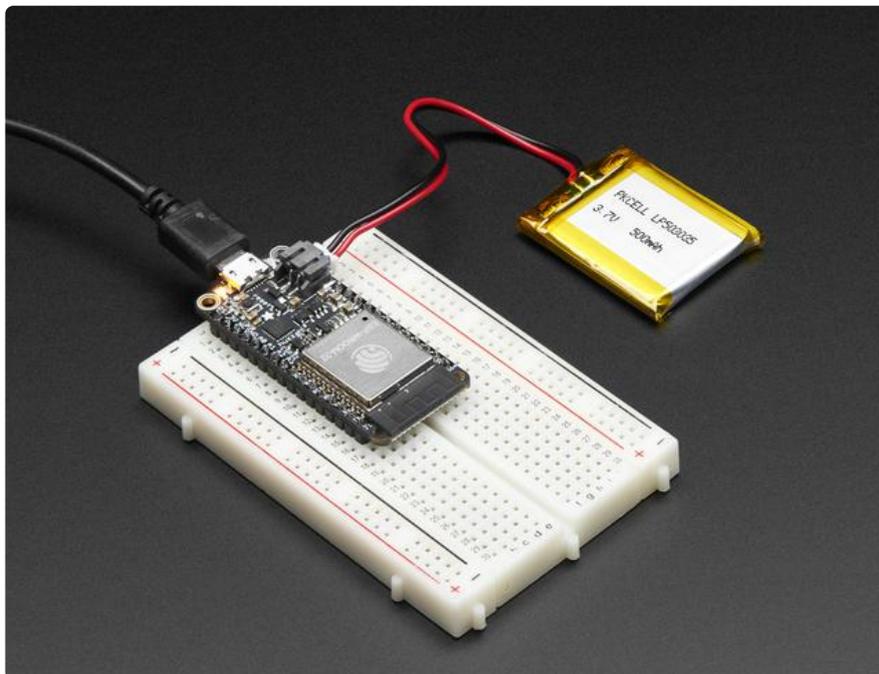
Battery + USB Power

We wanted to make our Feather boards easy to power both when connected to a computer as well as via battery.

There's two ways to power a Feather:

1. You can connect with a Micro USB cable (just plug into the jack) and the Feather will regulate the 5V USB down to 3.3V.
2. You can also connect a 4.2/3.7V Lithium Polymer (LiPo/LiPoly) or Lithium Ion (LiIon) battery to the JST jack. This will let the Feather run on a rechargeable battery.

When the USB power is powered, it will automatically switch over to USB for power, as well as start charging the battery (if attached). This happens 'hot-swap' style so you can always keep the LiPoly connected as a 'backup' power that will only get used when USB power is lost.

> The JST connector polarity is matched to Adafruit LiPoly batteries. Using wrong polarity batteries can destroy your Feather.
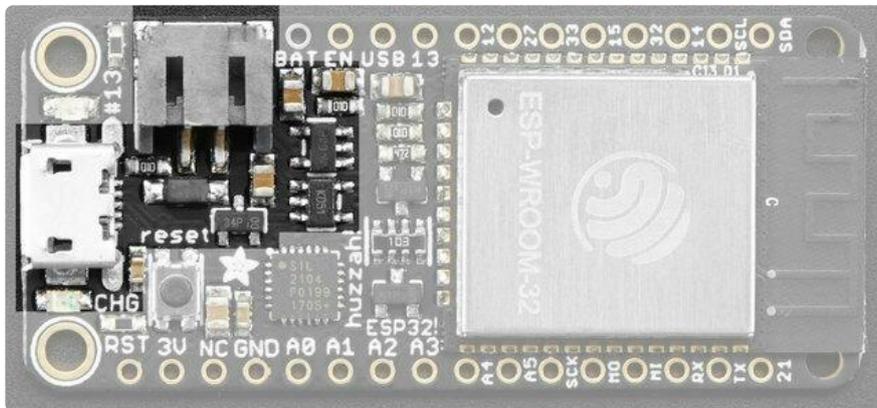
The above shows the Micro USB jack (left), LiPoly JST jack (top left), as well as the 3.3V regulator (to the right of the JST jack), changeover diode+transistor (below the JST jack) and the LiPoly charging circuitry (right below the regulator).

There's also a CHG LED next to the USB jack, which will light up while the battery is charging. This LED might also flicker if the battery is not connected, it's normal.

Power Supplies

You have a lot of power supply options here! We bring out the BAT pin, which is tied to the LiPoly JST connector, as well as USB which is the +5V from USB if connected. We also have the 3V pin which has the output from the 3.3V regulator. We use a 500mA peak regulator. While you can get 500mA from it, you can't do it continuously from 5V as it will overheat the regulator.

Please budget 250mA for the WROOM32 module. We use this to power the ESP32 which draws about 200mA continuous. The good news is you can put the ESP32 into sleep and low-power modes much easier.
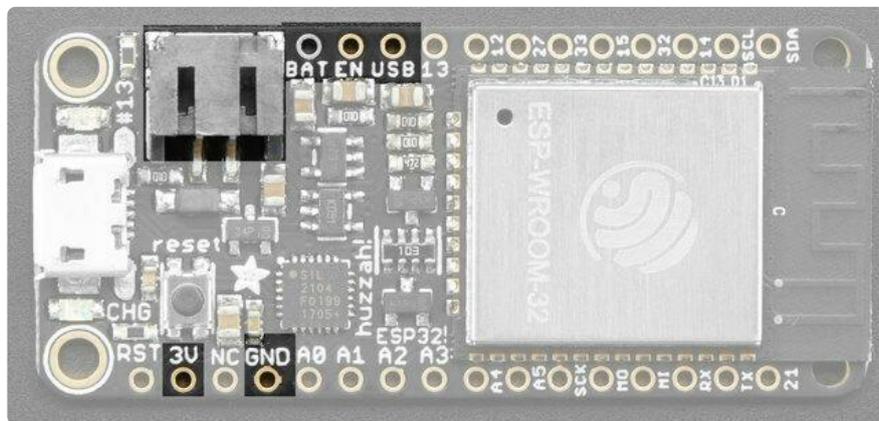
Measuring Battery

If you're running off of a battery, chances are you wanna know what the voltage is at! That way you can tell when the battery needs recharging. LiPoly batteries are 'maxed out' at 4.2V and stick around 3.7V for much of the battery life, then slowly sink down to 3.2V or so before the protection circuitry cuts it off. By measuring the voltage you can quickly tell when you're heading below 3.7V.

Since the ESP32 has tons of ADC pins, we 'sacrifice' one for Lipoly battery monitoring. You can read half of the battery voltage off of A13. Don't forget to double the voltage you read, since there is a divider.
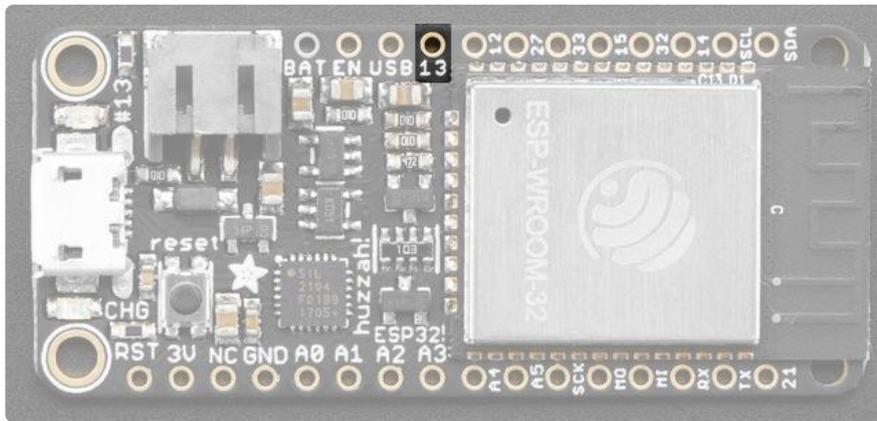
ENable pin

If you'd like to turn off the 3.3V regulator, you can do that with the EN(able) pin. Simply tie this pin to Ground and it will disable the 3V regulator. The BAT and USB pins will still be powered.

This will turn off the ESP32 processor as well as all onboard circuitry except the USB-Serial converter.
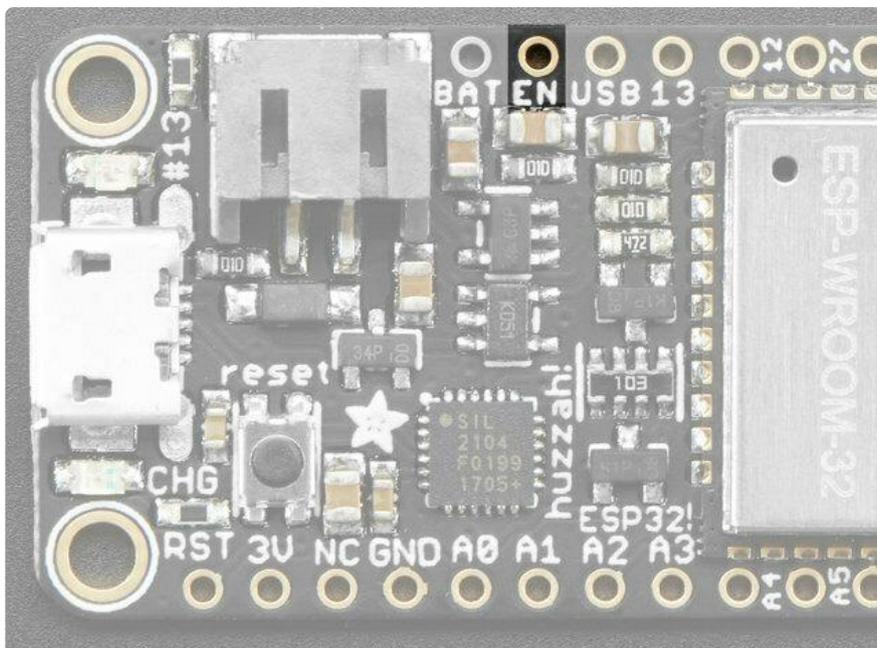
Alternative Power Options

The two primary ways for powering a feather are a 3.7/4.2V LiPo battery plugged into the JST port or a USB power cable.

If you need other ways to power the Feather, here's what we recommend:

- For permanent installations, a 5V 1A USB wall adapter (https://adafru.it/duP) will let you plug in a USB cable for reliable power
- For mobile use, where you don't want a LiPoly, use a USB battery pack! (https://adafru.it/e2q)
- If you have a higher voltage power supply, use a 5V buck converter (https://adafru.it/DHs) and wire it to a USB cable's 5V and GND input (https://adafru.it/DHu)

Here's what you cannot do:

- Do not use alkaline or NiMH batteries and connect to the battery port - this will destroy the LiPoly charger and there's no way to disable the charger
- Do not use 7.4V RC batteries on the battery port - this will destroy the board

The Feather is not designed for external power supplies - this is a design decision to make the board compact and low cost. It is not recommended, but technically possible:

- Connect an external 3.3V power supply to the 3V and GND pins. Not recommended, this may cause unexpected behavior and the EN pin will no longer. Also this doesn't provide power on BAT or USB and some Feathers/ Wings use those pins for high current usages. You may end up damaging your Feather.
- Connect an external 5V power supply to the USB and GND pins. Not recommended, this may cause unexpected behavior when plugging in the USB port because you will be back-powering the USB port, which could confuse or damage your computer.

Using with Arduino IDE

We primarily recommend using the ESP32 Feather with Arduino.

Check out the Espressif Arduino repository for details on how to install it (https://adafru.it/weF)

Don't forget you will also need to install the SiLabs CP2104 Driver

https://adafru.it/vrf

Once installed, use the Adafruit ESP32 Feather board in the dropdown

For Upload speed we've found 921600 baud works great.

> The ESP32 uses a fair amount of current, so if you're getting flakey behavior make sure you are plugging your console cable into either a motherboard USB port or a powered USB hub. Don't use the 'extra' USB port on your monitor or keyboard.
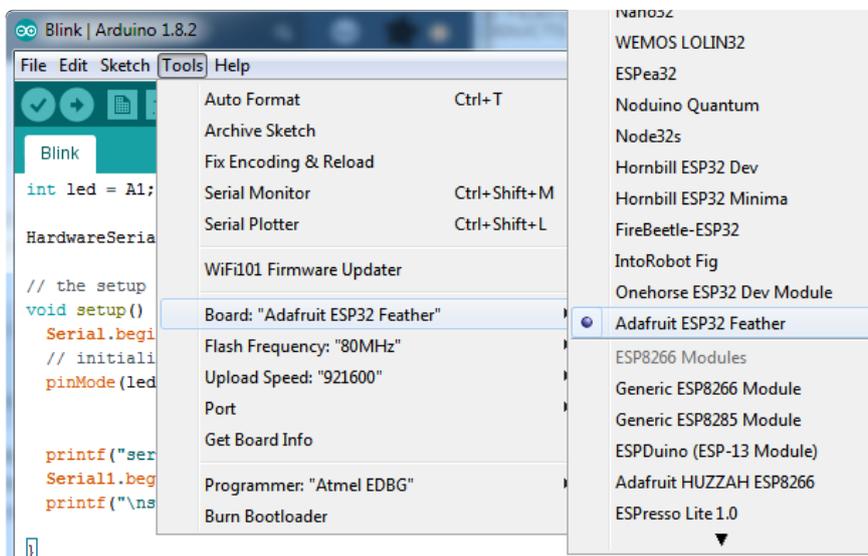
WipperSnapper Setup

> The WipperSnapper firmware and ecosystem are in BETA and are actively being developed to add functionality, more boards, more sensors, and fix bugs. We encourage you to try out WipperSnapper with the understanding that it is not final release software and is still in development.
>
> If you encounter any bugs, glitches, or difficulties during the beta period, or with this guide, please contact us via http://io.adafruit.com/support

What is WipperSnapper

WipperSnapper is a firmware designed to turn any WiFi-capable board into an Internet-of-Things device without programming a single line of code. WipperSnapper connects to Adafruit IO (https://adafru.it/fsU), a web platform designed (by Adafruit! (https://adafru.it/Bo5)) to display, respond, and interact with your project's data.

Simply load the WipperSnapper firmware onto your board, add credentials, and plug it into power. Your board will automatically register itself with your Adafruit IO account.

From there, you can add components to your board such as buttons, switches, potentiometers, sensors, and more! Components are dynamically added to hardware, so you can immediately start interacting, logging, and streaming the data your projects produce without writing code.

Sign up for Adafruit.io

You will need an Adafruit IO account to use WipperSnapper on your board. If you do not already have one, head over to io.adafruit.com (https://adafru.it/fsU) to create a free account.

Add a New Device to Adafruit IO

Log into your Adafruit IO (https://adafru.it/fsU) account. Click the New Device button at the top of the page.
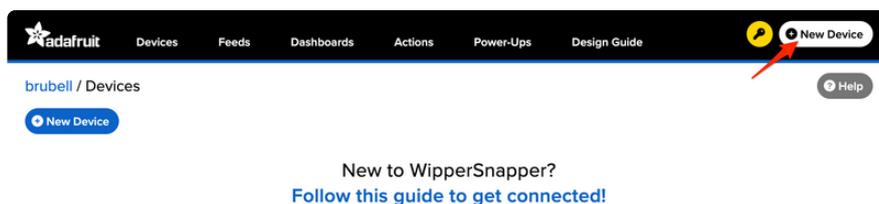
After clicking New Device, you should be on the board selector page. This page displays every board that is compatible with the WipperSnapper firmware.
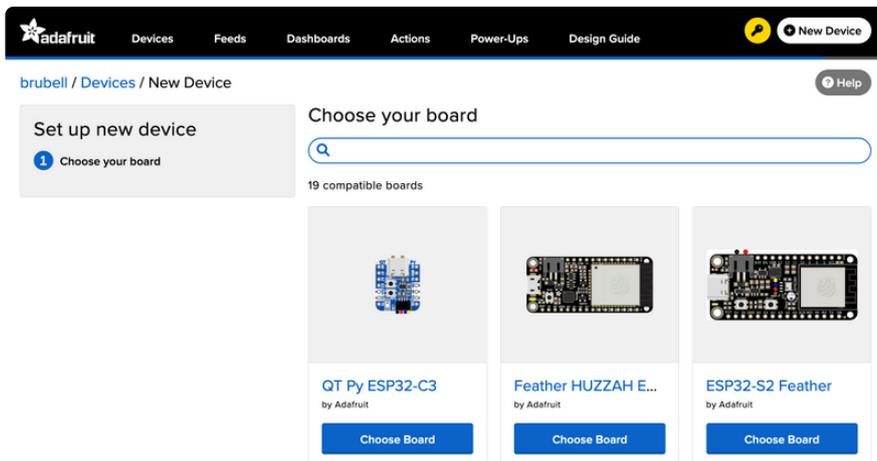
In the board selector page's search bar, search for the ESP32-S2 Feather. Once you've located the board you'd like to install WipperSnapper on, click the Choose Board button to bring you to the self-guided installation wizard.
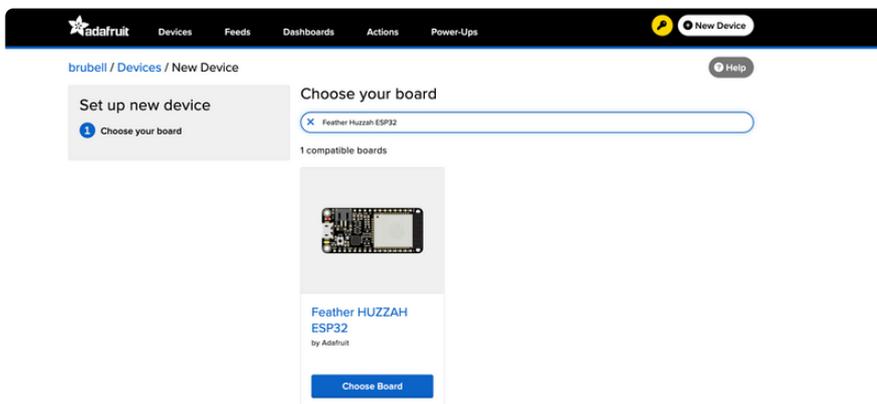
Follow the step-by-step instructions on the page to install Wippersnapper on your device and connect it to Adafruit IO.


If the installation was successful, a popover should appear displaying that your board has successfully been detected by Adafruit IO.

Give your board a name and click "Continue to Device Page".

New Device Detected!   ✕

You have successfully connected a new **feather-esp32** device to Adafruit IO. It is already set up and submitting data. You can name the device here, and set up components on the device page.
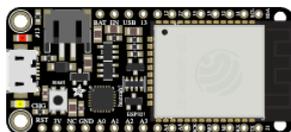
Device Name

Adafruit Feather HUZZAH ESP32

Firmware Version: ✅ v1.0.0-beta.38

Continue to Device Page ›

You should be brought to your board's device page. The next step, WipperSnapper Usage, will teach you how to configure and control your development board over the Internet.
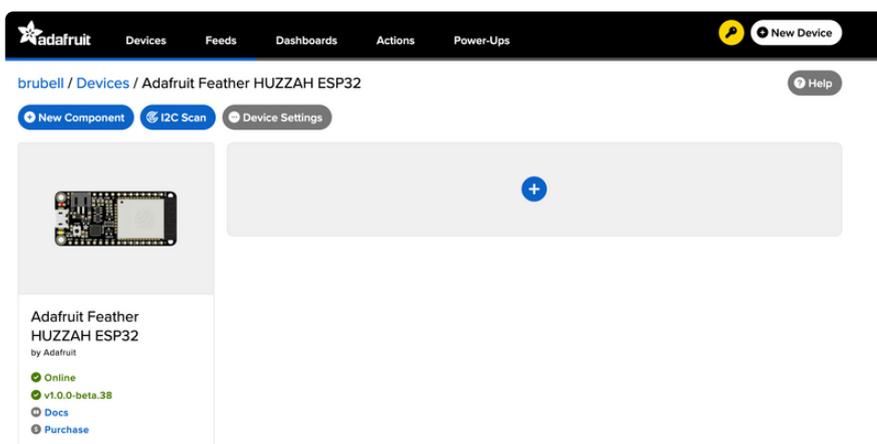
Feedback

Adafruit.io WipperSnapper is in beta and you can help improve it!

If you have suggestions or general feedback about the installation process - visit [https://io.adafruit.com/support](https://adafru.it/Sgb) (https://adafru.it/Sgb), click "Contact Adafruit IO Support" and select "I have feedback or suggestions for the WipperSnapper Beta".

Troubleshooting

If you encountered an issue during installation, please try the steps below first.

If you're still unable to resolve the issue, or if your issue is not listed below, get in touch with us directly at https://io.adafruit.com/support (https://adafru.it/Sgb). Make sure to click "Contact Adafruit IO Support" and select "There is an issue with WipperSnapper. Something is broken!"

I don't see my board on Adafruit IO, it is stuck connecting to WiFi

First, make sure that you selected the correct board on the board selector.

Next, please make sure that you entered your WiFi credentials properly, there are no spaces/special characters in either your network name (SSID) or password, and that you are connected to a 2.4GHz wireless network.

If you're still unable to connect your board to WiFi, please make a new post on the WipperSnapper technical support forum with the error you're experiencing, the LED colors which are blinking, and the board you're using. (https://adafru.it/V6a)

I don't see my board on Adafruit IO, it is stuck "Registering with Adafruit IO"

Try hard-resetting your board by unplugging it from USB power and plugging it back in.

If the error is still occurring, please make a new post on the WipperSnapper technical support forum with information about what you're experiencing, the LED colors which are blinking (if applicable), and the board you're using. (https://adafru.it/V6a)

"Uninstalling" WipperSnapper

WipperSnapper firmware is an application that is loaded onto your board. There is nothing to "uninstall". However, you may want to "move" your board from running WipperSnapper to running Arduino or CircuitPython. You also may need to restore your board to the state it was shipped to you from the Adafruit factory.

Moving from WipperSnapper to CircuitPython

Follow the steps on the Installing CircuitPython page (https://adafru.it/Amd) to install CircuitPython on your board running WipperSnapper.

- If you are unable to double-tap the RST button to enter the UF2 bootloader, follow the "Factory Resetting a WipperSnapper Board" instructions below.

Uploading this sketch will overwrite WipperSnapper. If you want to re-install WipperSnapper, follow the instructions at the top of this page.

Moving from WipperSnapper to Arduino

If you want to use your board with Arduino, you will use the Arduino IDE to load any sketch onto your board.

First, follow the page below to set up your Arduino IDE environment for use with your board.

<div align="center">

Setup Arduino IDE

https://adafru.it/AKr

</div>

Then, follow the page below to upload the "Arduino Blink" sketch to your board.

<div align="center">

Upload Arduino "Blink" Sketch

https://adafru.it/10aM

</div>

Uploading this sketch will overwrite WipperSnapper. If you want to re-install WipperSnapper, follow the instructions at the top of this page.

Factory Resetting a WipperSnapper Board

Sometimes, hardware gets into a state that requires it to be "restored" to the original state it shipped in. If you'd like to get your board back to its original factory state, follow the guide below.

WipperSnapper Usage

Now that you've installed WipperSnapper on your board - let's learn how to use Adafruit IO to interact with your board!

There's a large number of components (physical parts like buttons, switches, sensors, actuators) supported by the WipperSnapper firmware, this page will get you started with the core concepts to build an IoT project with Adafruit IO.

> This page assumes that you have installed WipperSnapper on your Feather and registered it with the Adafruit.io WipperSnapper web page. If you have not done this yet, please go back to the previous page in this guide and connect your Feather.

Blink a LED

One of the first programs you typically write to get used to embedded programming is a sketch that repeatedly blinks an LED. IoT projects are wireless so after completing this section, you'll be able to turn on (or off) the LED built into your Feather from anywhere in the world.

> In this demo, we show controlling an LED from Adafruit IO. But the same kind of control can be used for relays, lights, motors, or solenoids.

Navigate to the device page, io.adafruit.com/wippersnapper (https://adafru.it/TAu). You should see the Feather you just connected to Adafruit IO listed on this page.
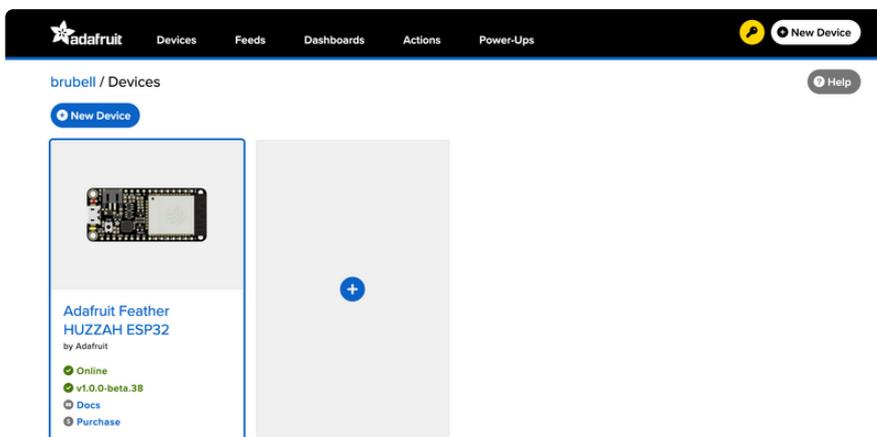
Verify that your Feather is online and ready to communicate with Adafruit IO by checking that the device tile says Online in green text.

- If the Feather appears offline on the website but was previously connected, press the Reset (RST) button to force the board to reboot.
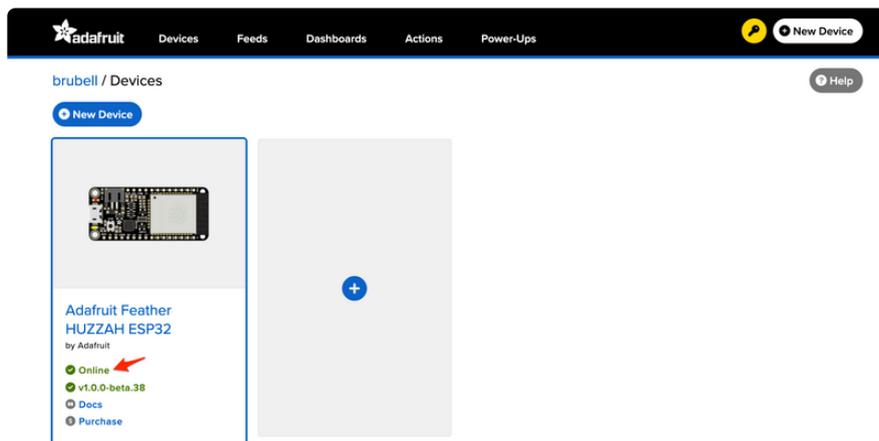
Once verified that the device is online, click the device tile to navigate to the device's interface page.
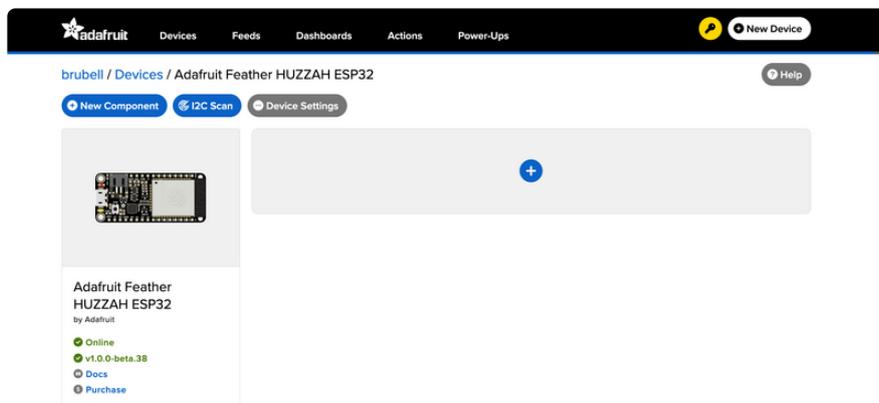

Add a new component to your Feather by clicking the + button or the + New Component button on the device interface.
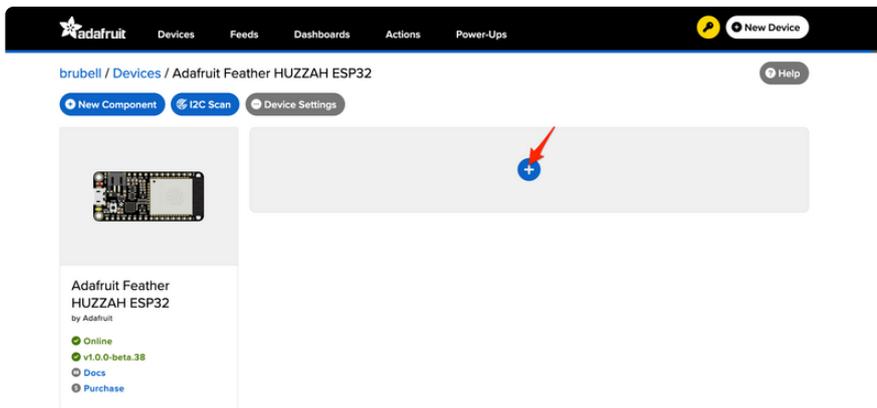
The Component Picker lists all the available components, sensors, and parts, which can be used with the WipperSnapper firmware. Your Feather board already has a LED built-in, so there's no wiring for you to configure.

Click the LED component.
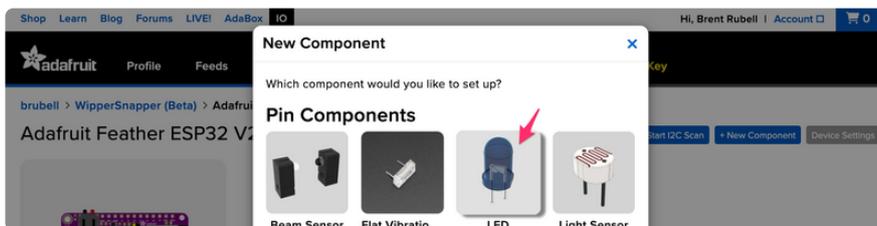
Feathers contain GPIO pins that can be configured either as an input or an output. The "Create LED Component" screen tells WipperSnapper to configure a general-purpose output pin connected to the LED on your Feather as a digital output so you can turn the LED on or off.

The Feather ESP32 has a built-in LED located at pin D13. Select this pin as the LED Pin and click Create Component

Create LED Component ✕

LED Name

LED

LED Pin

D13 LED ▾
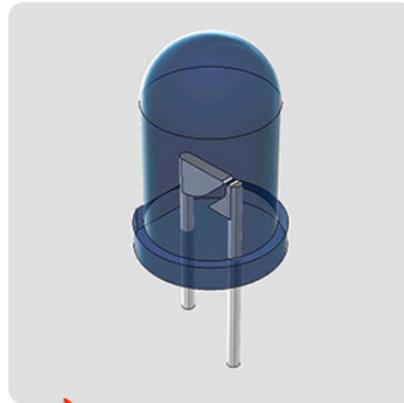
⟨ **Previous Step**   **Create Component**

Behind the scenes, Adafruit IO sends send a command to your board running WipperSnapper telling it to configure the GPIO pin as a digital output. Your Feather's interface shows the new LED component.
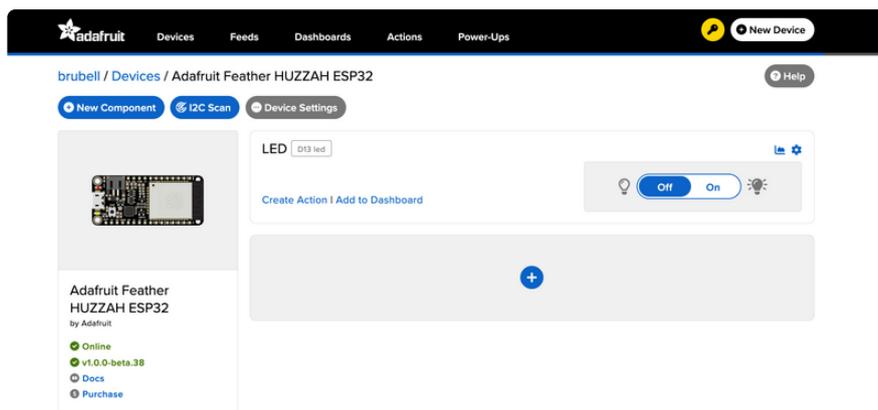
On the device interface, toggle the LED component by clicking the toggle switch. This should turn your Feather's built-in LED on or off.
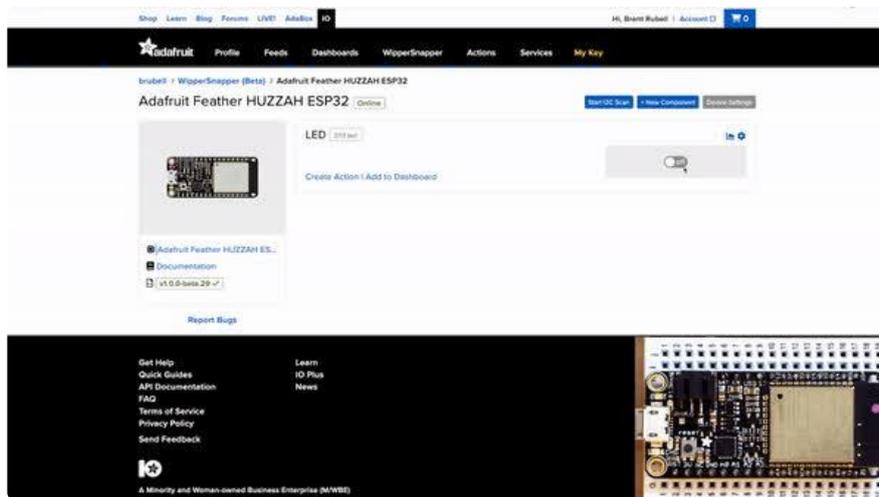
Read a Push-Button

You can also configure a board running WipperSnapper to wirelessly read data from standard input buttons, switches, or digital sensors, and send the value to Adafruit IO.

Let's wire up a push button to your Feather and configure it with Adafruit IO to publish a value to Adafruit IO when the button has been pressed or depressed.

> In this demo, we show reading the state of a push-button from WipperSnapper. But the same kind of control can be used for reading switches, break beam sensors, and other digital sensors.

Wiring

We'll be using the Feather's internal pull-up resistors instead of a physical resistor.
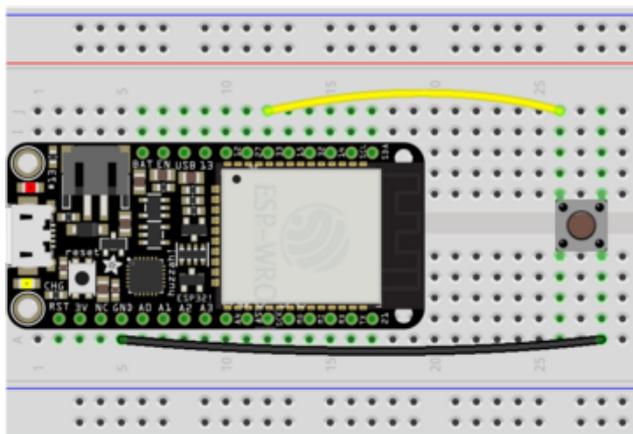
Feather GND to Push Button
Feather GPIO 33 to Push Button

Usage

On the device interface, add a new component to your Feather by clicking the + button or the + New Component button on the device interface.
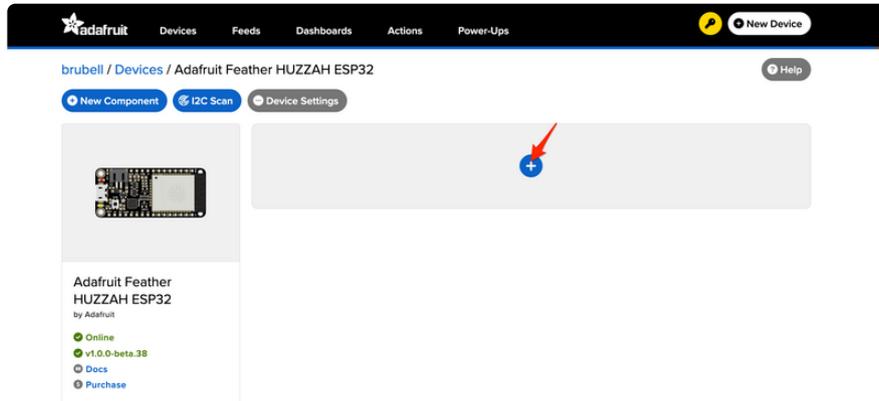
From the component picker, select the Push Button.
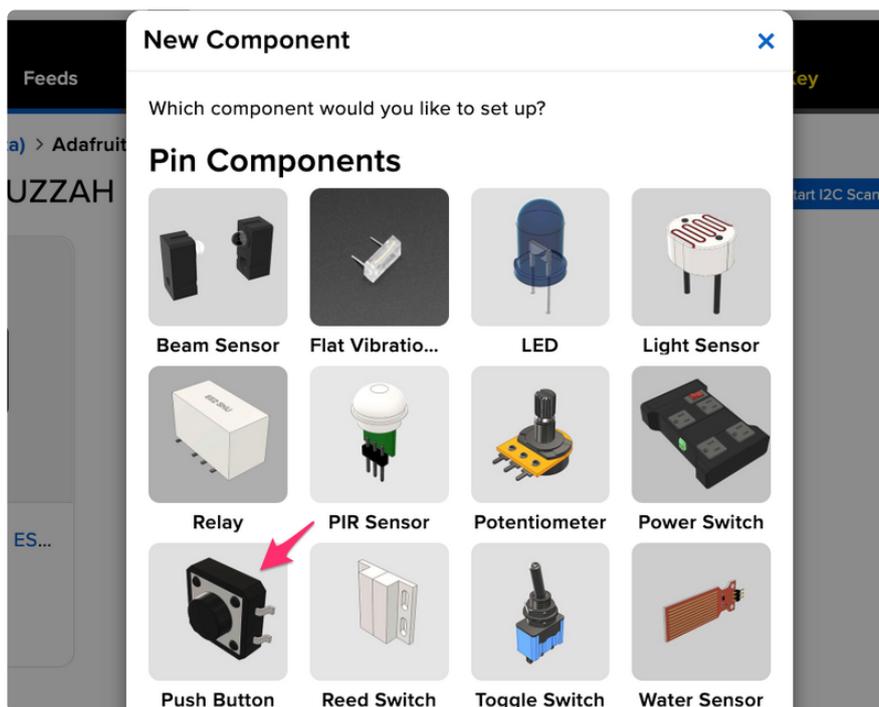
The next screen presents you with options for configuring the push button. Start by selecting the Feather's digital pin you connected to the push button.
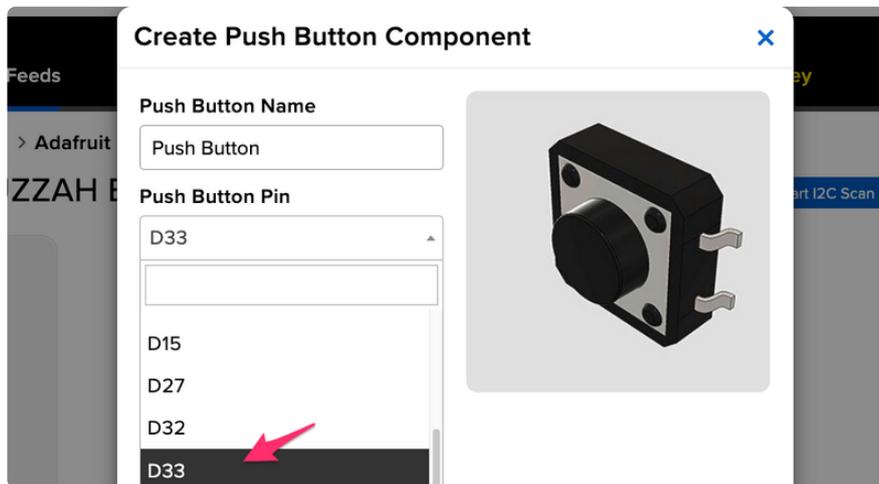
The Return Interval dictates how frequently the value of the push-button will be sent from the Feather to Adafruit IO. For this example, the push-button's value should only be sent when its pressed.

Select On Change
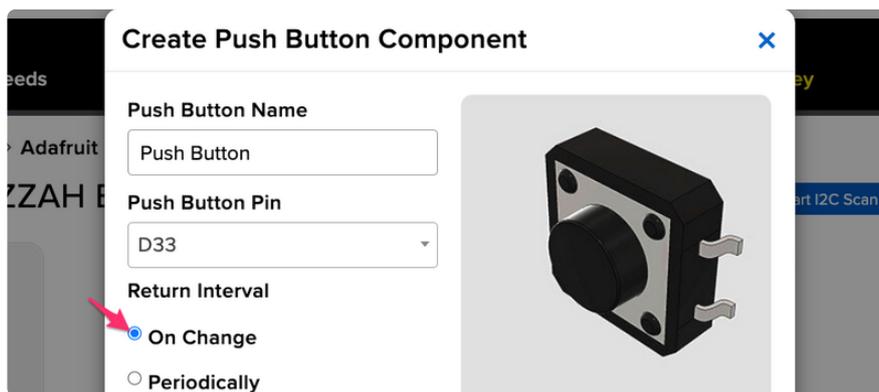
Finally, check the Specify Pin Pull Direction checkbox and select Pull Up to turn on the Feather's internal pullup resistor.
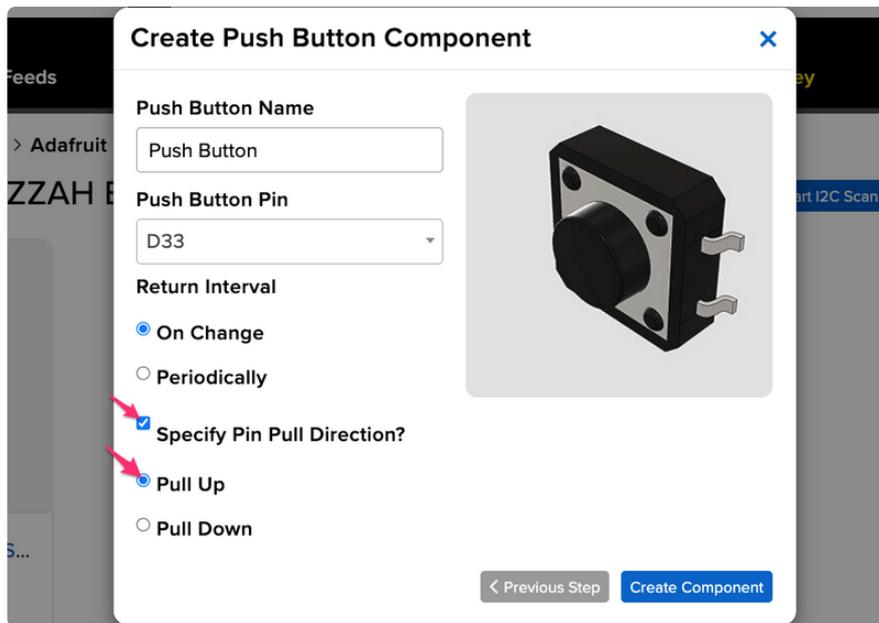
Make sure the form's settings look like the following screenshot. Then, click Create Component.
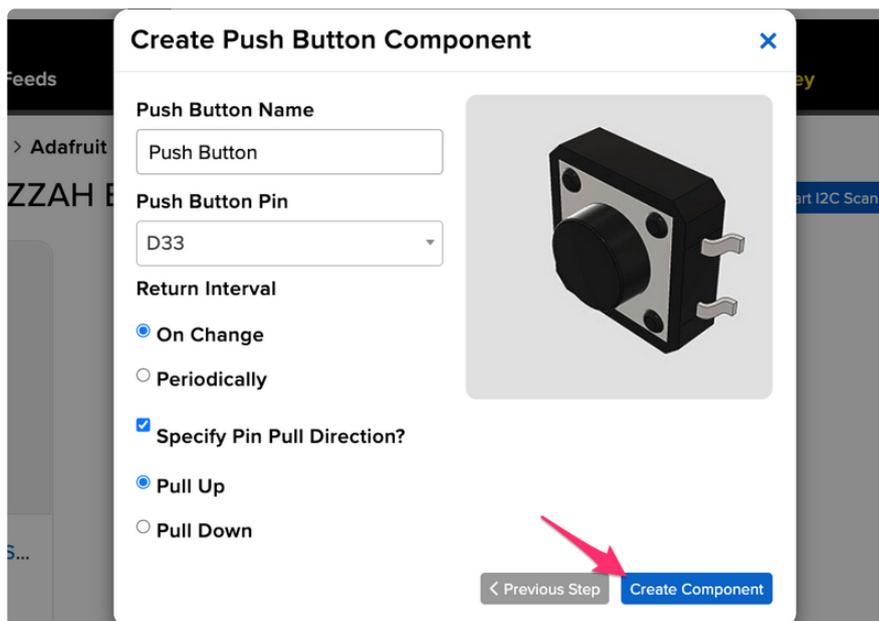
Adafruit IO should send a command to your board (running WipperSnapper), telling it to configure the GPIO pin you selected to behave as a digital input pin and to enable it to pull up the internal resistor. Your Feather's interface should also show the new push-button component.
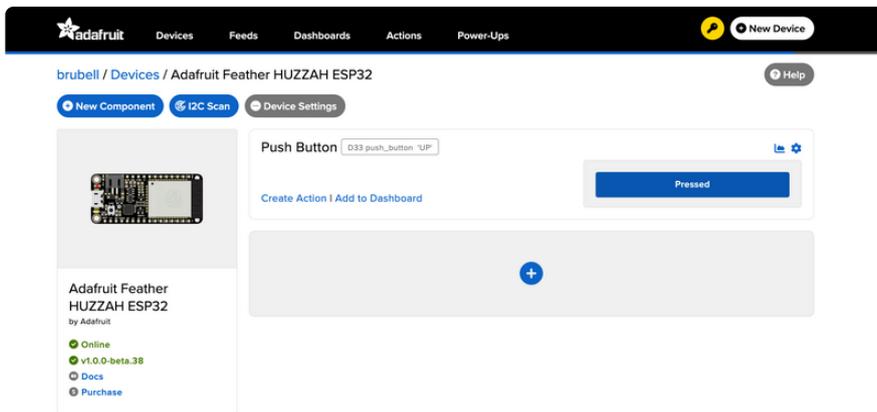
Press the button to change the value of the push button component on Adafruit IO.
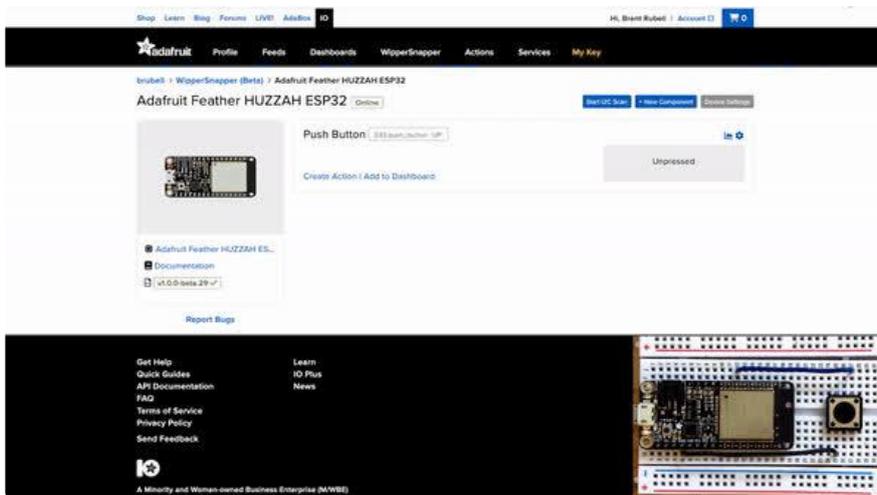
Read an I2C Sensor

Inter-Integrate Circuit, aka I2C, is a two-wire protocol for connecting sensors and "devices" to a microcontroller. A large number of sensors, including the ones sold by Adafruit, use I2C to communicate.

Typically, using I2C with a microcontroller involves programming. Adafruit IO lets you configure a microcontroller to read data from an I2C sensor and publish that data to the internet without writing code.

The WipperSnapper firmware supports a number of I2C sensors, viewable in list format here (https://adafru.it/Zbq). If you do not see the I2C sensor you're attempting to use with WipperSnapper - we have a guide on adding a component to Adafruit IO WipperSnapper here (https://adafru.it/Zbr).

The process for adding an I2C component to your board running WipperSnapper is similar to most sensors. For this section, we're using the Adafruit AHT20 (https://

adafru.it/RKe), an inexpensive sensor that can read ambient temperature and humidity.

Wiring

First, wire up an AHT20 sensor to your board exactly as follows. Here is an example of the AHT20 wired to a Feather using I2C with a STEMMA QT cable (no soldering required) (https://adafru.it/FA-).
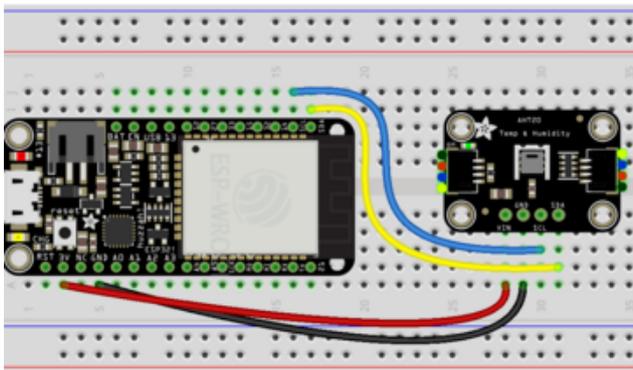

Board 3V to sensor VIN (red wire on STEMMA QT)
Board GND to sensor GND (black wire on STEMMA QT)
Board SCL to sensor SCL (yellow wire on STEMMA QT)
Board SDA to sensor SDA (blue wire on STEMMA QT)

Scan I2C Bus

First, ensure that you've correctly wired the AHT20 sensor to your Feather by performing an I2C scan to detect the I2C device on the bus.

On the upper right-hand corner of the device interface, click Start I2C Scan.

- If you do not see this button, double-check that your Feather shows as Online.
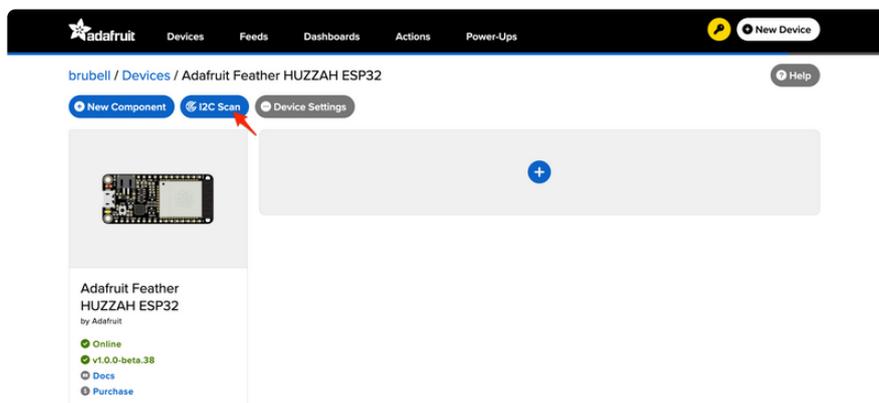
You should see a new pop-up showing a list of the I2C addresses detected by an I2C scan. If wired correctly, the AHT20's default I2C address of `0x38` appear in the I2C scan list.

## I don't see the I2C sensor's address in the list

First, double-check the connection and/or wiring between the sensor and the board.

Then, reset the board and let it re-connect to Adafruit IO WipperSnapper.

Create the Sensor Component

Now that you know the sensor can be detected by the Feather, it's time to configure and create the sensor on Adafruit IO.

On the device interface, add a new component to your Feather by clicking the + button or the + New Component button on the device interface.
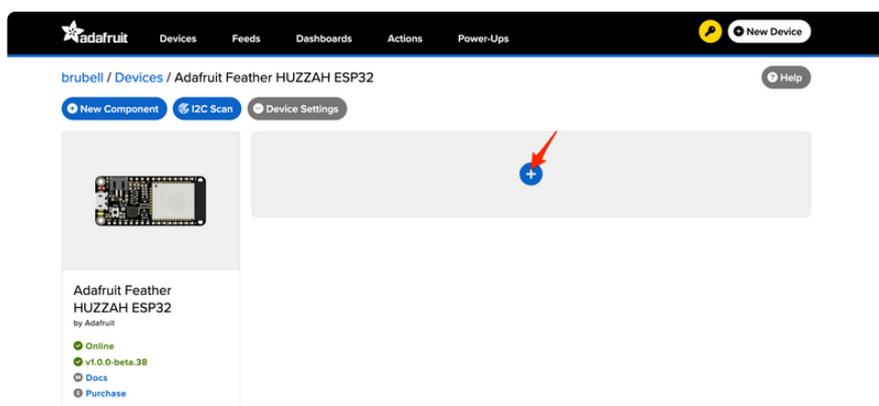
The Component Picker lists all the available components, sensors, and parts that can be used with WipperSnapper.

Under the I2C Components header, click AHT20.
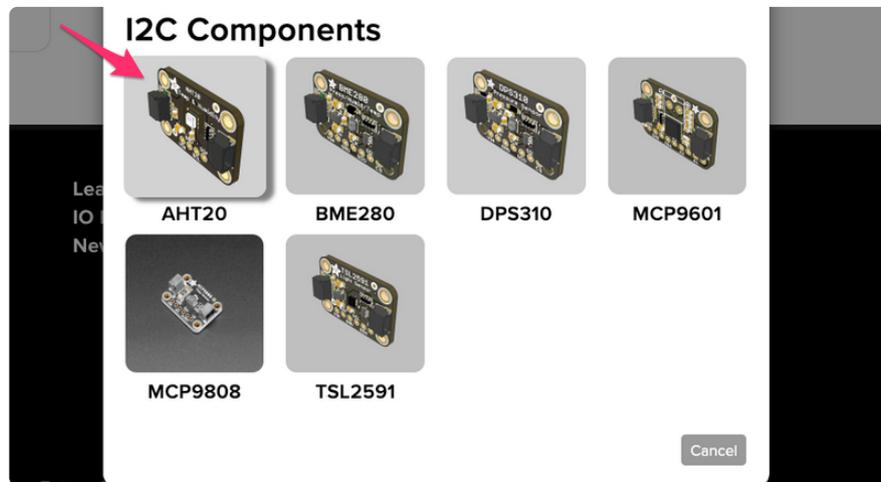
On the component configuration page, the AHT20's I2C sensor address should be listed along with the sensor's settings.

The AHT20 sensor can measure ambient temperature and relative humidity. This page has individual options for reading the temperature and the relative humidity. This comes in handy in the case where you have multiple I2C sensors connected to your Feather which read the same value. You may choose to enable or disable a specific sensor type on this page.

The Send Every option is specific to each sensor measurements . This option will tell the Feather how often it should read from the AHT20 sensor and send the data to Adafruit IO. Measurements can range from every 30 seconds to every 24 hours.

For this example, set the Send Every interval for both seconds to Every 30 seconds and click Create Component.
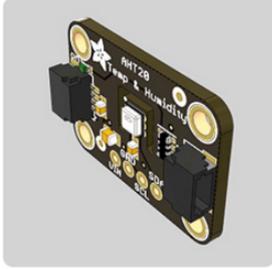
The device interface should now show the AHT20 component you created.

After the interval you configured elapses, WipperSnapper automatically reads values from the sensor and sends them to Adafruit IO.

Going Further

Want to learn more about Adafruit IO WipperSnapper? We have more complex projects on the Adafruit Learning System (https://adafru.it/Yyd).

ESP32 F.A.Q

Some pins are special about the ESP32 - here's a list of 'notorious' pins to watch for!

- A2 / I34 - this pin is an input only! You can use it as an analog input so we suggest keeping it for that purpose
- A3 / I39 - this pin is an input only! You can use it as an analog input so we suggest keeping it for that purpose
- IO12 - this pin has an internal pulldown, and is used for booting up. We recommend not using it or if you do use it, as an output only so that nothing interferes with the pulldown when the board resets
- A13 / I35 - this pin is not exposed, it is used only for measuring the voltage on the battery. The voltage is divided by 2 so be sure to double it once you've done the analog reading

Why does the yellow CHARGE LED blink while USB powered?

The charging circuit will flash when there is no LiPoly battery plugged in. It's harmless and doesn't mean anything. When a LiPoly battery is connect it will stabilize the charger and will stop flashing

Why can I not read analog inputs once WiFi is initialized?

Due to the design of the ESP32, you can only read analog inputs on ADC #1 once WiFi has started. That means pins on ADC 2 (check the pinouts page) can't be used as analog inputs

Why is Serial.read() not working as expected on ESP32 Breakout?

This is a minor design issue with the initial version of the Breakout (does not apply to Feather version). If you are having issues similar to the discussion here (https://adafru.it/ven), try the trick of enabling the internal pull-up as described here (https://adafru.it/ven).

Downloads

Files

- ESP32 WROOM32 Datasheet (https://adafru.it/EVE)
- ESP32 Technical Manual (https://adafru.it/weC)
- Don't forget to visit esp32.com for the latest and greatest in ESP32 news, software and gossip! (https://adafru.it/weD)
- EagleCAD PCB files on github (https://adafru.it/weE)
- Fritzing object in the Adafruit Fritzing library (https://adafru.it/aP3)
- 3D Models on GitHub (https://adafru.it/GAA)
- PDF for Huzzah32 ESP32 Feather Board Diagram on GitHub (https://adafru.it/ZKA)

SVG for Huzzah32 ESP32 Feather Board Diagram

https://adafru.it/ZKB
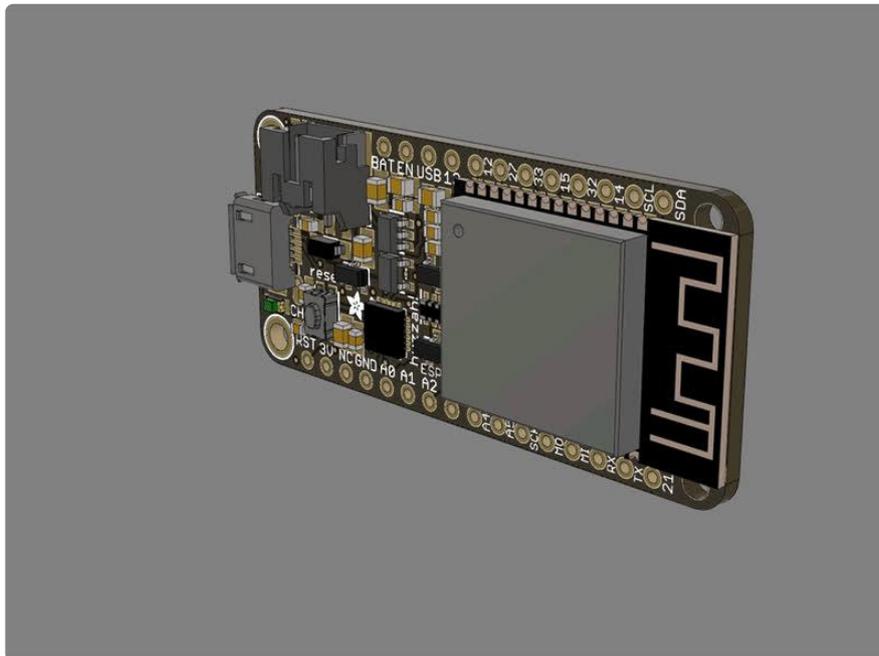
Schematic and Fabrication print