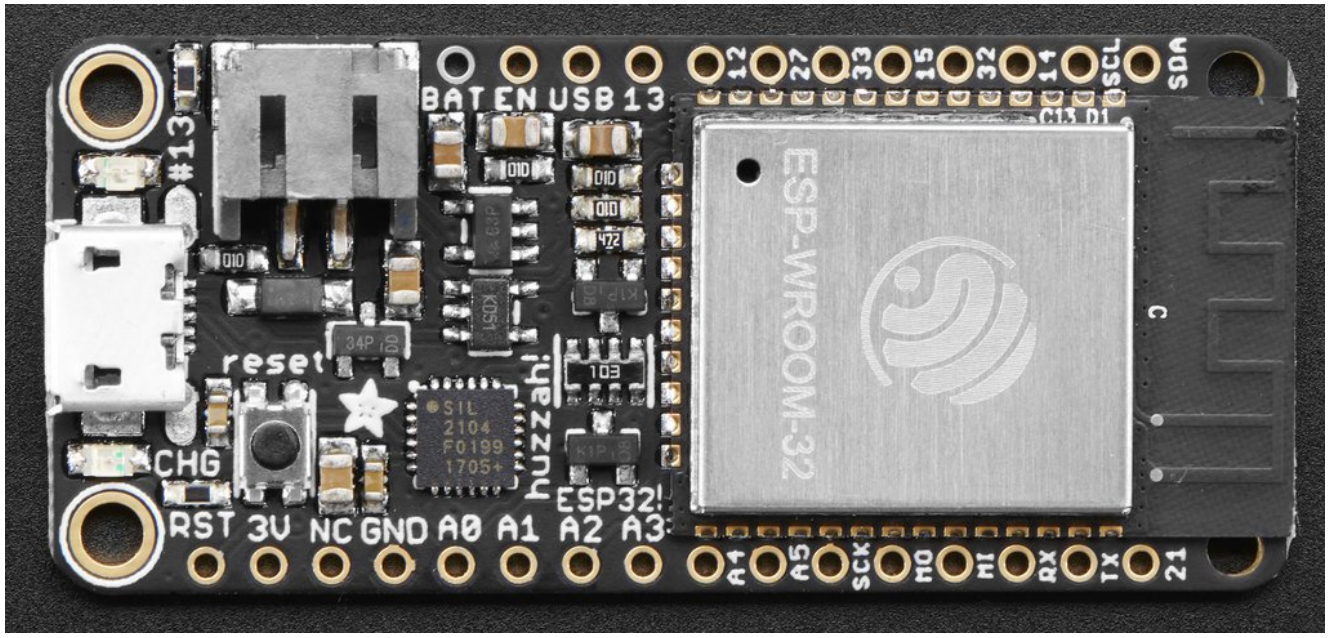
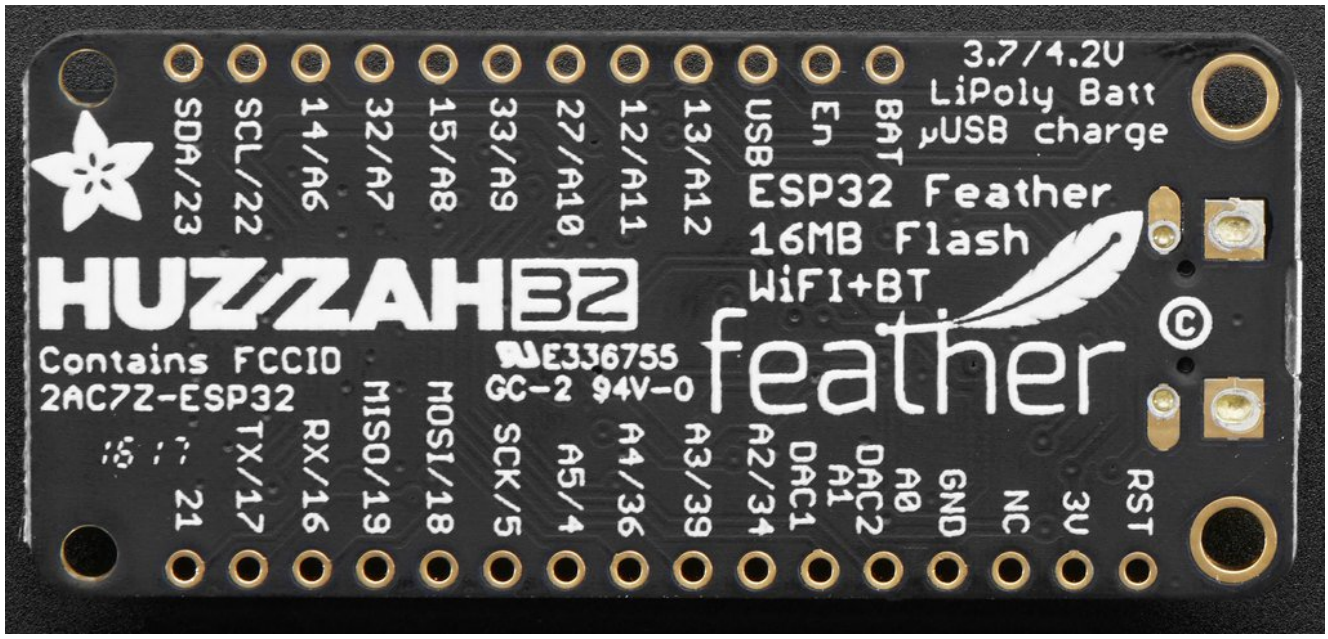
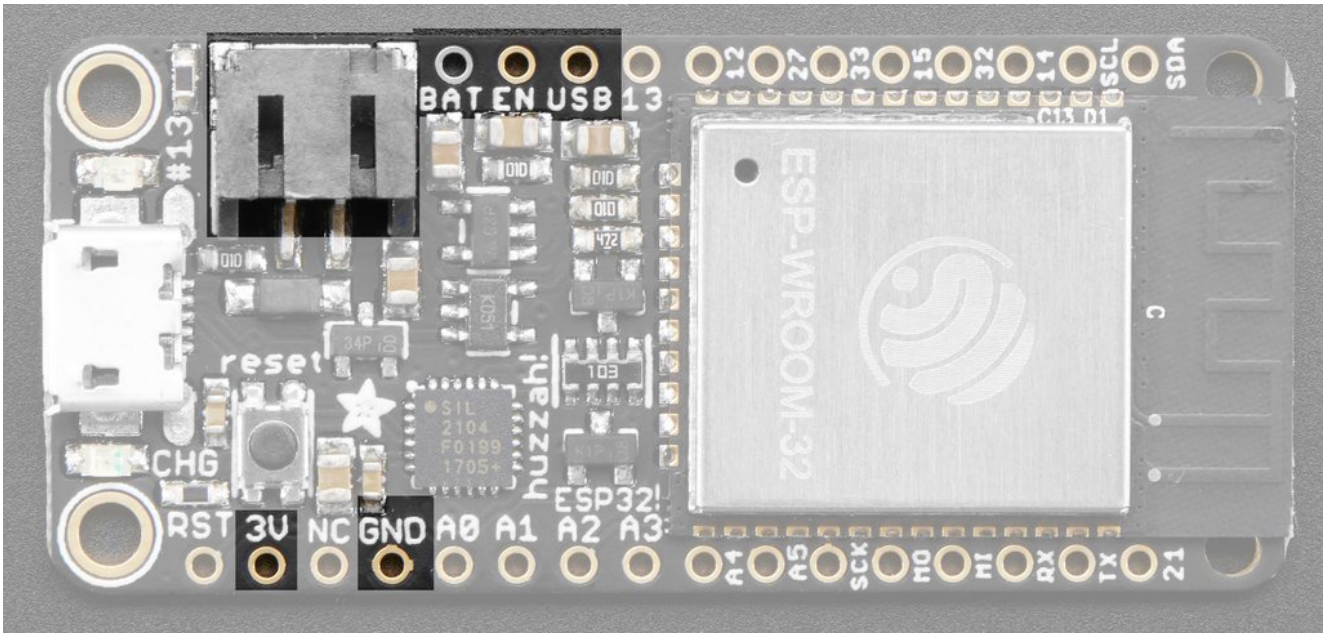
Adafruit HUZZAH32 - ESP32 Feather

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







- **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

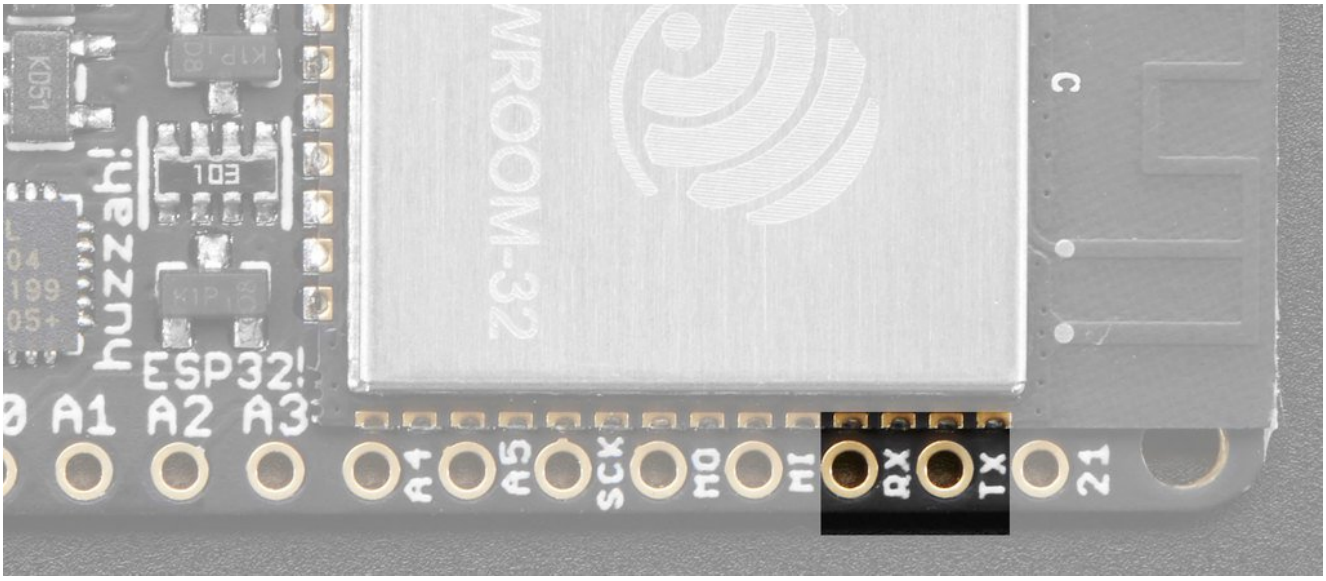
Logic pins

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

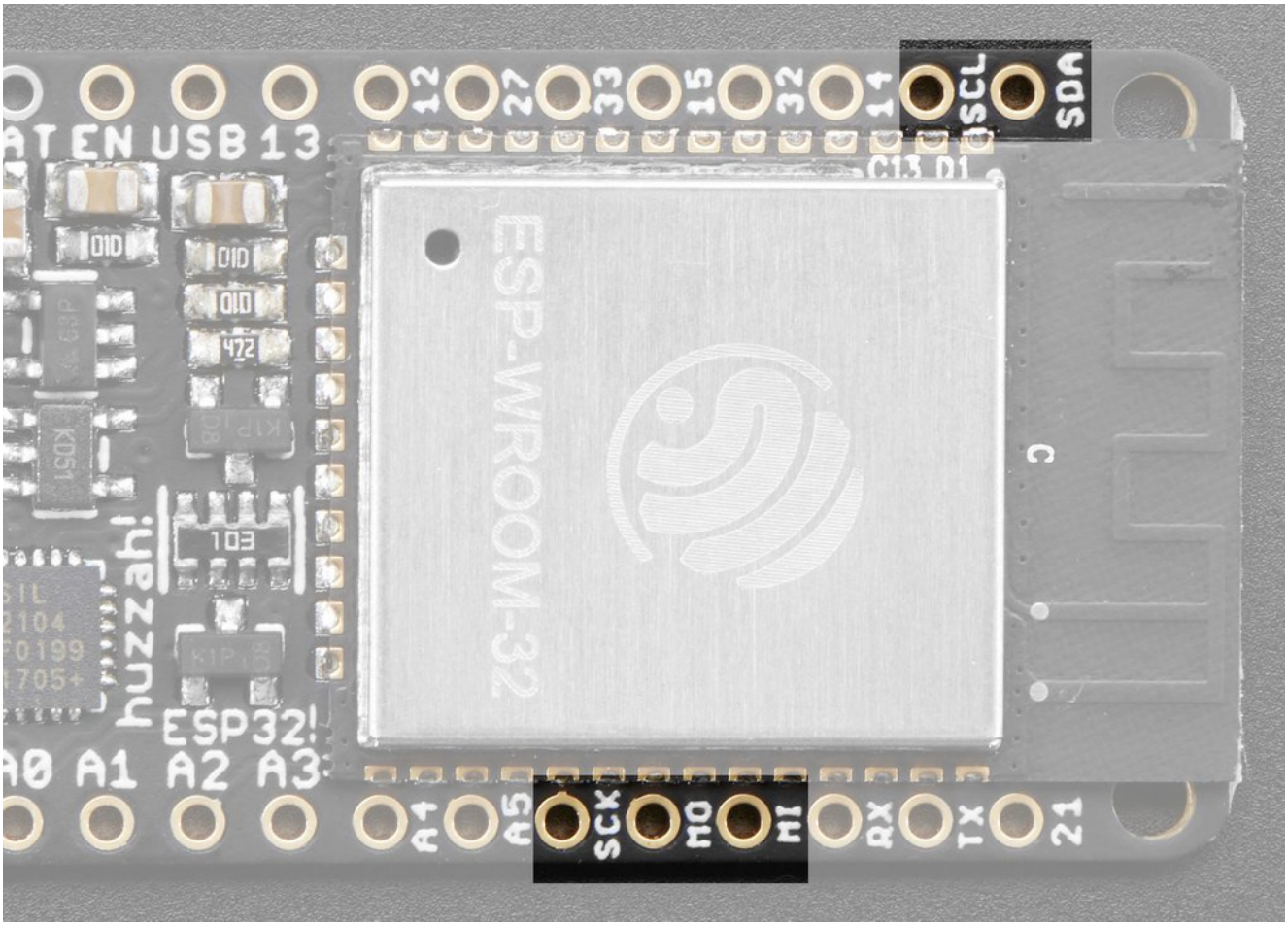
RX and **TX** are the additional Serial1 pins, and are *not* connected to the USB/Serial converter. That means you can use them to connect to UART-devices like GPS's, fingerprint sensors, etc.



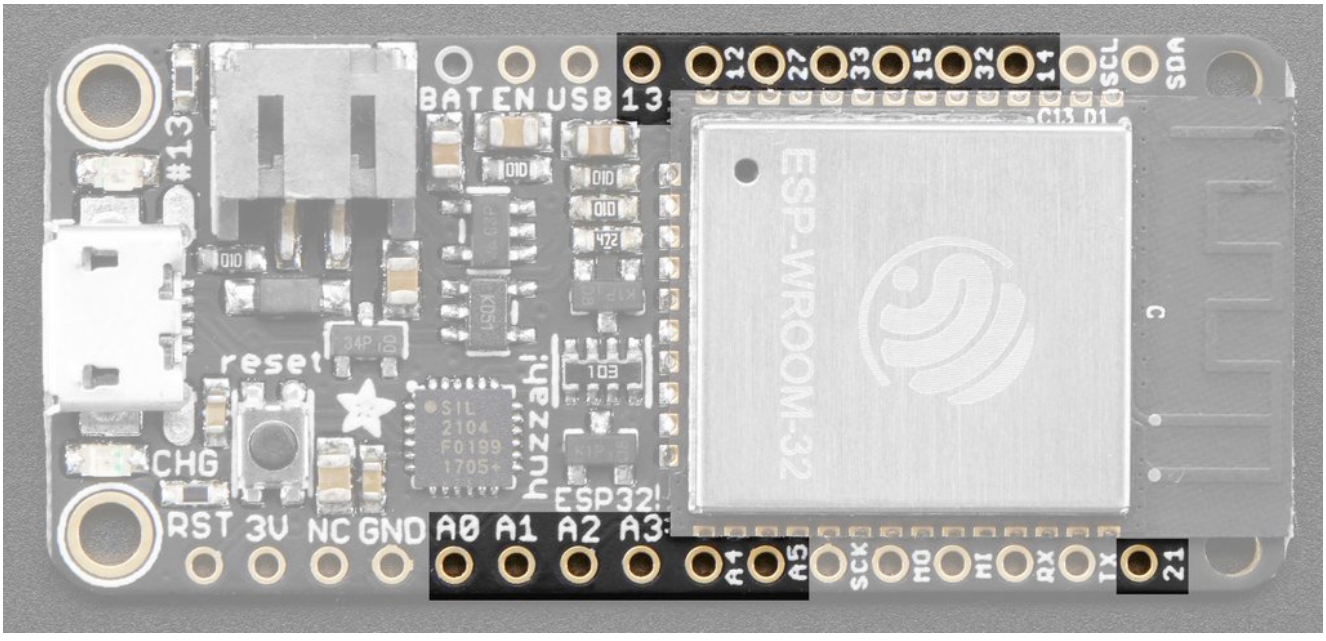
The **TX** pin is the output *from* the module. The **RX** pin is the input *into* the module. Both are 3.3V logic

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



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 GPI #34. Note it is *not* an output-capable pin! It uses ADC #1
- **A3** - this is an analog input A3 and also GPI #39. Note it is *not* an output-capable pin! It uses ADC #1
- **A4** - this is an analog input A4 and also GPI #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 #1. 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 #1** once WiFi has started

This guide was first published on May 10, 2017. It was last updated on May 10, 2017. This page (Pinouts) was last updated on Jun 15, 2018.