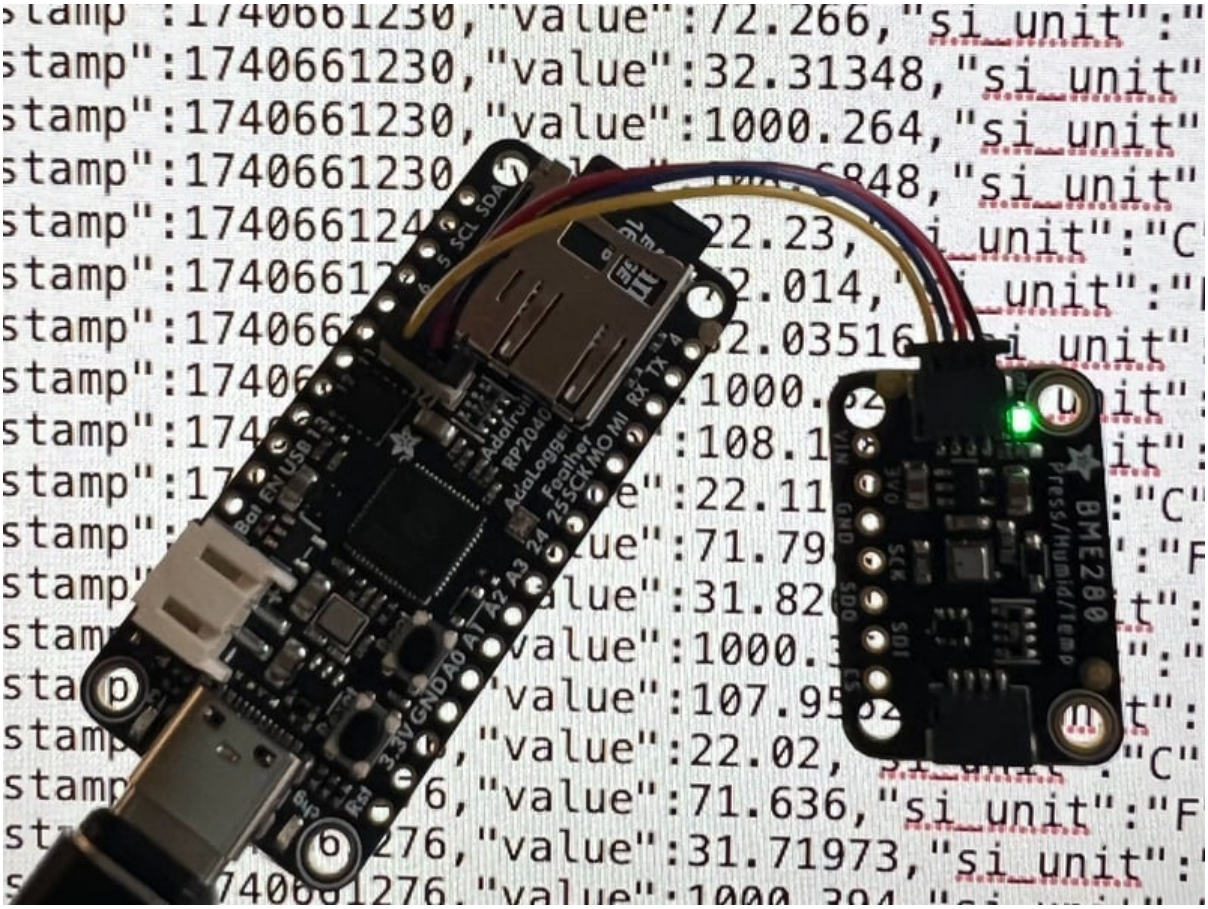
No-Code Offline Data Logger with WipperSnapper

Created by Brent Rubell



https://learn.adafruit.com/no-code-offline-data-logging-with-wippersnapper

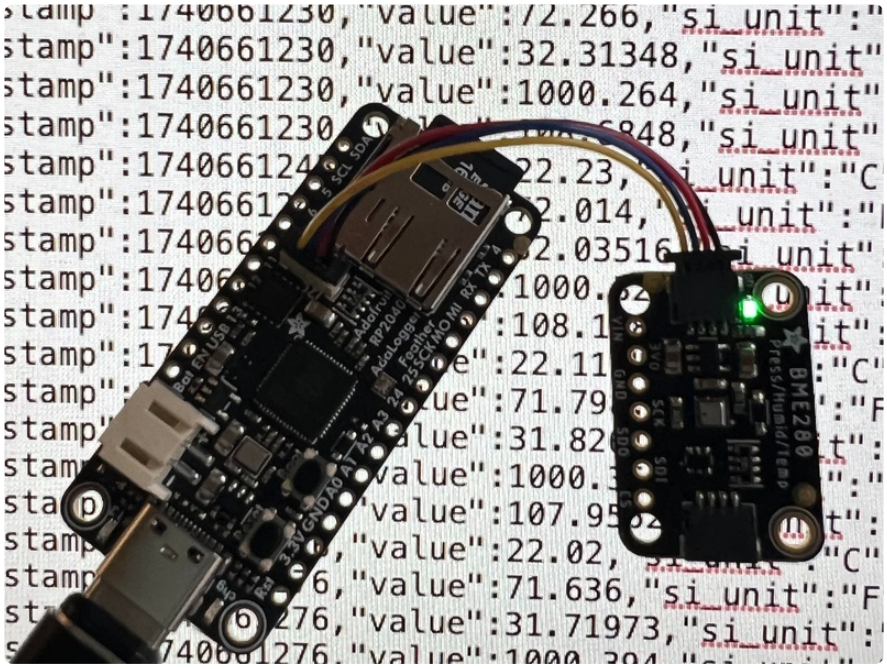
Last updated on 2025-03-05 11:39:43 AM EST

Table of Contents

Overview

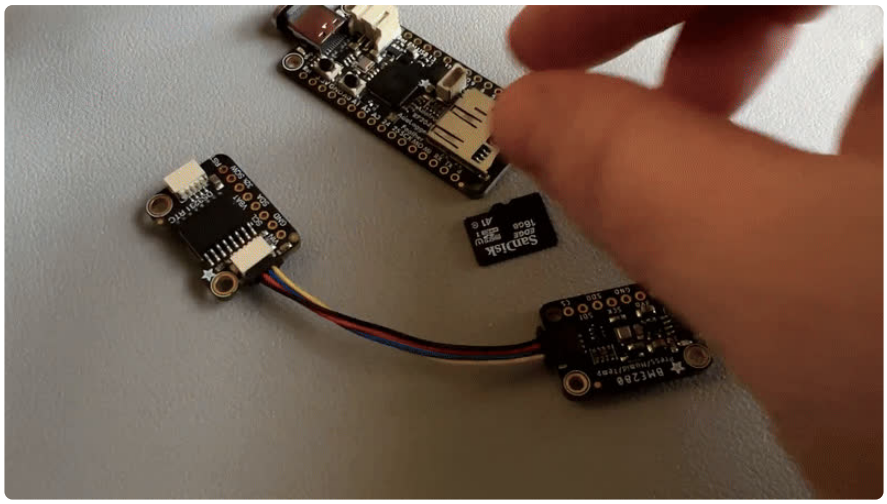
You have entered the Beta Zone! This guide along with WipperSnapper Firmware's Offline Mode is a work in progress.

For more info, please scroll down to the bottom of this page.



Looking to build an electronics project that logs data to a MicroSD card?

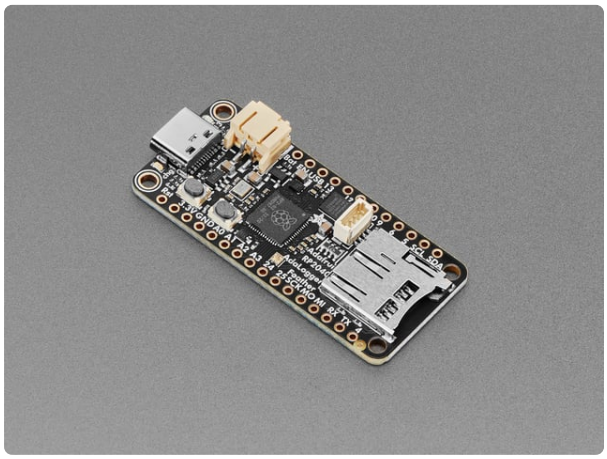
In this guide, we'll **build a simple datalogger that can log temperature, barometric pressure, and humidity to a microSD card, without programming**! You'll install WipperSnapper's offline mode firmware on an Adafruit Feather RP2040 Adalogger, format the microSD card to be used for storage, and add a configuration file to use the Feather RP2040 with the BME280 Humidity + Barometric Pressure + Temperature Sensor Breakout.



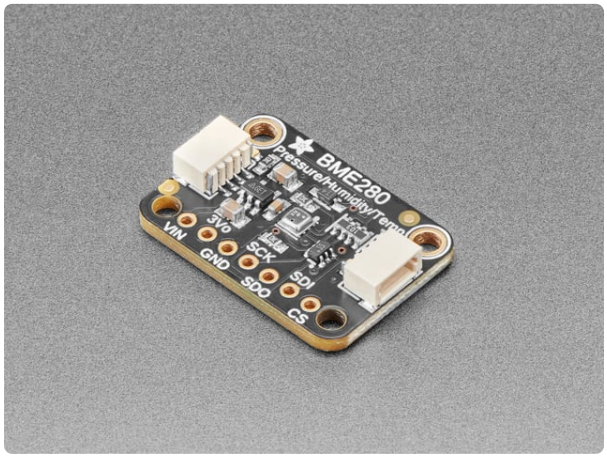
What's "WipperSnapper Offline Mode"?

Adafruit WipperSnapper (https://adafru.it/Vfd) firmware's offline mode enables you to **build and configure a data logger project without writing any code.** Just **install the WipperSnapper offline mode firmware on your board**, **add a configuration file** to the board's filesystem, **and reboot the board**. When the board reboots, **the board automatically configures its sensors and immediately begins logging data to a microSD card** and through a USB Serial connection.

Parts



Adafruit Feather RP2040 Adalogger - 8MB Flash with microSD Card
This is the Adafruit Feather RP2040 Adalogger - our take on an 'all-in-one' RP2040 data-logger (or data-reader) with built-in USB, battery charging,...
https://www.adafruit.com/product/5980



Adafruit BME280 I2C or SPI Temperature Humidity Pressure Sensor
Bosch has stepped up their game with their new BME280 sensor, an environmental sensor with temperature, barometric pressure and humidity! This sensor is great for all sorts...
https://www.adafruit.com/product/2652

SD/MicroSD Memory Card (8 GB SDHC)

1 x MicroSD Memory Card https://www.adafruit.com/product/1294

SD/MicroSD Memory Card (8 GB SDHC)

---

1 x STEMMA QT JST SH 4-pin Cable https://www.adafruit.com/product/4210

STEMMA QT / Qwiic JST SH 4-pin Cable - 100mm Long

---

1 x USB Cable USB Type A to Type C https://www.adafruit.com/product/5031

Right Angle USB Type C - Black 1 meter long

About WipperSnapper's Offline Mode Beta

Adafruit WipperSnapper's offline mode is currently in active development. While the firmware is ready for you to use, configuring the device isn't as magical and adafruit-y as we'd like...yet!

Our developers are building out a web-based workflow that will be accessible from Adafruit.IO to allow you to visually configure your device's sensors and export the resulting configuration. This will be similar to how you add sensors and configure a device from Adafruit IO.

We encourage you to try out the WipperSnapper offline firmware with the understanding that it is not final release software, is still in development, and will get easier to use over the coming months.

What skills do I need to complete this guide?

You should be comfortable creating and editing a JSON text file.

I read through the guide and it is unfortunately too advanced for me. What should I do?

We are releasing this guide before the web-based workflow to get feedback from the community. You may want to wait a few months for us to build out the web-based workflow for the offline mode.

I am having a trouble - My device is not working properly

If you encounter any bugs, glitches, or difficulties while following this guide, please file an issue on the WipperSnapper support forum (https://adafru.it/V6a).

Supported Hardware

The project example in this guide uses the Adafruit Feather RP2040 Adalogger (http://adafru.it/5980) to capture and log data and an Adafruit BME280 sensor breakout (http://adafru.it/2652) to read data.

However, WipperSnapper's offline mode is not limited to this one development board and breakout. There is support for more development boards, I2C sensors, 1-wire temperature sensors, analog input pins, digital input pins, and, real-time clock modules.

Development Boards

Offline Mode is currently supported by a limited number of development boards (we are hoping to expand this as we work on the web-based workflow). If you have one of these boards, you can follow along with the guide!

RP2040 / RP2350

- **Raspberry Pi Pico RP2040**
- **Raspberry Pi Pico 2 - RP2350**

ESP32-Sx

- **Adafruit Metro ESP32-S2**
- **Adafruit ESP32-S2 Feather - 4 MB Flash + 2 MB PSRAM**
- **Adafruit ESP32-S2 TFT Feather - 4MB Flash, 2MB PSRAM, STEMMA QT**
- **Adafruit ESP32-S2 Reverse TFT Feather**
- **Adafruit QT Py ESP32-S2 WiFi Dev Board with STEMMA QT**
- **Adafruit Metro ESP32-S3 with 16 MB Flash 8 MB PSRAM**
- **Adafruit ESP32-S3 Feather with 4MB Flash 2MB PSRAM**
- **Adafruit ESP32-S3 TFT Feather - 4MB Flash, 2MB PSRAM, STEMMA QT**
- **Adafruit ESP32-S3 Reverse TFT Feather**
- **Adafruit QT Py S3 with 2MB PSRAM WiFi Dev Board with STEMMA QT**

Compatible Analog and Digital Inputs/Sensors

WipperSnapper supports logging data from digital inputs (such as a switch (http://adafru.it/805) or button (http://adafru.it/1119)) and analog inputs (such as a potentiometer (http://adafru.it/356), photocell (http://adafru.it/161), or a force-sensitive resistor (http://adafru.it/166)). Digital inputs log data as booleans (on/off) while analog inputs can log either the raw value from the analog-to-digital converter or a voltage.

Compatible 1-Wire Digital Temperature Sensors (DS18B20)

WipperSnapper also supports logging data from 1-wire DS18B20 digital temperature sensors. Adafruit sells the bare sensor (http://adafru.it/374) and waterproof versions (http://adafru.it/642) in the shop.

The only caveat/limitation with the software is that you can only connect one DS18B20 to a pin. If you are using this sensor with WipperSnapper, select a board with a large number of GPIO pins.

Compatible I2C Sensors

WipperSnapper supports the following I2C sensors. You can connect sensors directly to your board's SCL/SDA pins or use an easy-to-use connector like Adafruit's STEMMA / STEMMA QT (https://adafru.it/YOE) or Sparkfun's Qwiic (https://adafru.it/Fpw).

Temperature

- ADT7410 (http://adafru.it/4089)
- MCP9808 (http://adafru.it/1782)
- PCT2075 (http://adafru.it/4369)
- TMP117 (http://adafru.it/4821)
- HTU21D (http://adafru.it/1899)

Humidity & Temperature

- AHT20 (http://adafru.it/4566)
- HTS221 (http://adafru.it/4535)
- SHT30 (http://adafru.it/4099)
- SHT40 (http://adafru.it/4885)
- SHTC3 (http://adafru.it/4636)
- Si7021 (http://adafru.it/3251)

Environmental

- BME280 (http://adafru.it/2652)
- BME680 (http://adafru.it/3660)
- BMP280 (http://adafru.it/2651)
- BMP388 (http://adafru.it/4816)
- BMP390 (http://adafru.it/4816)
- DPS310 (http://adafru.it/4494)
- MS8607 (http://adafru.it/4716)
- SEN50/SEN54/SEN55 using SEN54 or SEN55 Adapter Breakout (http://adafru.it/5964)

Air Quality

- ENS160 (http://adafru.it/5606)
- SCD30 (http://adafru.it/4867)
- SCD4X (http://adafru.it/5187)
- SGP40 (http://adafru.it/4829)
- PM2.5 (http://adafru.it/4632)

Light

- BH1750 (http://adafru.it/4681)
- LTR329/LTR303 (http://adafru.it/5591)
- LTR390 (http://adafru.it/4831)
- TSL2591 (http://adafru.it/1980)
- VEML7700 (http://adafru.it/4162)

Pressure

- LPS22HB (http://adafru.it/4633)
- LPS25HB (http://adafru.it/4530)
- LPS33HW (http://adafru.it/4414)
- LPS35HW (http://adafru.it/4258)
- MPL115A2 (http://adafru.it/992)
- MPRLS (http://adafru.it/3965)

Distance/Proximity

- VL53L0X (http://adafru.it/3317)
- VL53L1X (http://adafru.it/3967)

- VL53L4CD (http://adafru.it/5396)
- VL53L4CX (http://adafru.it/5425)
- VL6180X (http://adafru.it/3316)
- VNCL4020 (http://adafru.it/466)
- VNCL4040 (http://adafru.it/4161)

Power/Current Monitoring

- INA219 (http://adafru.it/904)
- LC709203F (http://adafru.it/4712)
- MAX1704X (http://adafru.it/5580)
- MCP3421 (http://adafru.it/5870)
- NAU7802 (http://adafru.it/4538)

Uncategorized and Miscellaneous

- DS2484 (http://adafru.it/5650)
- STEMMA Soil (http://adafru.it/4026)

Compatible I2C Multiplexers

If you want to connect more than one I2C sensor with the same address, WipperSnapper supports connecting an I2C multiplexer

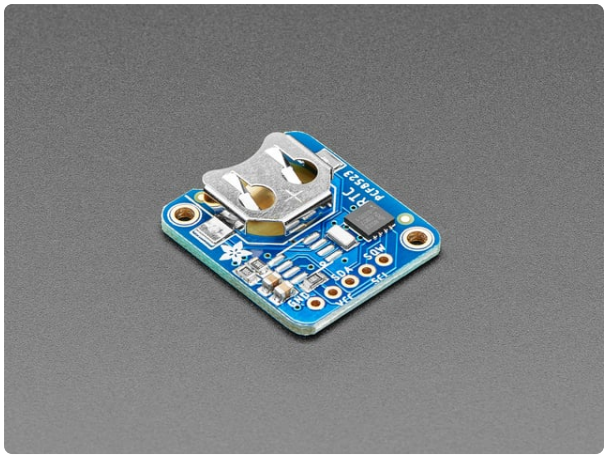
- PCA9546 (http://adafru.it/5664)
- PCA9548 (http://adafru.it/5626)

For more information about using an I2C multiplexer, check out this section of our "Working With I2C" guide (https://adafru.it/ZYe).

Compatible Real-Time-Clocks (RTC)

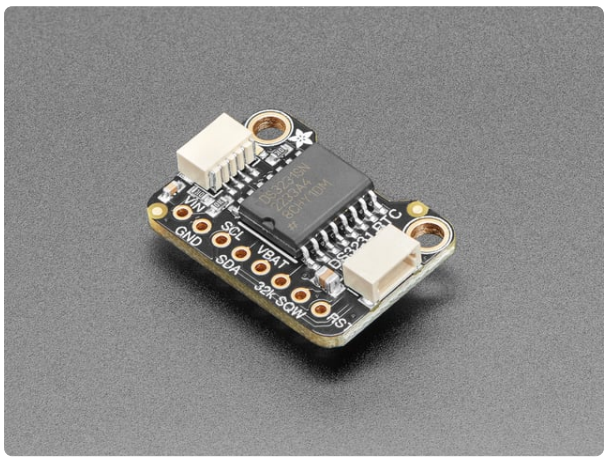
A real-time clock (RTC) allows your WipperSnapper project to keep track of time even if it is reprogrammed, or if it loses power. In WipperSnapper's Offline mode, the RTC provides a timestamp for every time new data is logged. This helps keep track of when measurements were taken.

However, if you're not using an Adalogger product, to use WipperSnapper's Offline Mode, a physical RTC is preferred (but not required) for data-logging projects as it provides higher precision and battery backup. If you do not have an RTC, the firmware can timestamp measurements using a "virtual" clock (also known a a "software" or "soft" RTC).



[Adafruit PCF8523 Real Time Clock Assembled Breakout Board](https://www.adafruit.com/product/3295)
This is a great battery-backed real time clock (RTC) that allows your microcontroller project to keep track of time even if it is reprogrammed, or if the power is lost. Perfect for...
https://www.adafruit.com/product/3295

The following real-time clock can also connect via STEMMA QT / Qwiic I2C connection for solder free use.



[Adafruit DS3231 Precision RTC - STEMMA QT](https://www.adafruit.com/product/5188)
The datasheet for the DS3231 explains that this part is an "Extremely Accurate I²C-Integrated RTC/TCXO/Crystal". And, hey, it does exactly...
https://www.adafruit.com/product/5188

Configure the MicroSD Card

A MicroSD card is required to be inserted into your board for offline mode. This applies even for cases where data is only intended to log to USB serial.

Which MicroSD Card Should I Get?

Any MicroSD Card with a capacity of less than 32GB should be good. When purchasing the card, be aware that some cheap cards are "fakes" and may cause headaches. The MicroSD cards in the Adafruit shop (http://adafru.it/1294) are guaranteed to work.

You will also need a way to read the logged data from the MicroSD Card. To do this, you will need an external SD Card reader to plug into your computer (http://adafru.it/5212).

Formatting under Windows/Mac

If you bought a MicroSD card, chances are it's already pre-formatted with a FAT filesystem. However, you may have problems with how the factory formats the card, or if it's an old card, it needs to be reformatted. The Arduino SD library used in the Wippersnapper software supports both **FAT16** and **FAT32** filesystems.

If you have a MicroSD card with a very small capacity, say 8-32 Megabytes you might find it is formatted **FAT12** which isn't supported by Offline Mode. You'll have to reformat these cards.

Either way, **it's always a good idea to format the card before using it, even if it's new!** Note that formatting will erase the card so save anything you want first.

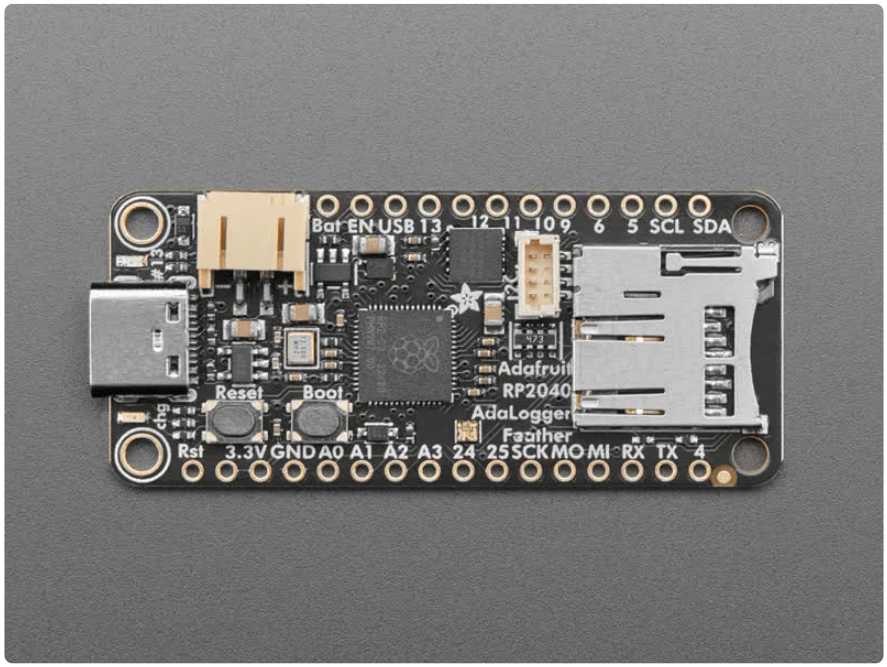
Before proceeding with the guide, we strongly recommend you use the official SD card formatted utility to format your MicroSD Card. This tool is written by the SD Association and it solves many problems that come with bad formatting. The link is below:

Download the Official SD Card Formatter for Windows/Mac

https://adafru.it/FKd

Insert the MicroSD Card

The Feather RP2040 Adalogger's **MicroSD card slot** is located at the end of the board, directly behind the STEMMA QT port. After formatting the MicroSD card, insert the card into the MicroSD card slot.



Install WipperSnapper Offline Firmware

I have a compatible board already running WipperSnapper but want to try out Offline Mode

If you wish to try Offline Mode and have a device running WipperSnapper, you must download the WipperSnapper Offline Mode beta firmware and install it on your device. This will replace the existing WipperSnapper (online) firmware on your board.

WipperSnapper (offline) is not permanently installed on your board, you can switch back to WipperSnapper (online) at any time.

I have a compatible board running an Arduino Sketch but want to try out Offline Mode

Installing WipperSnapper Offline will **completely erase the Arduino sketch that is currently running on your board**. Before installing WipperSnapper Offline, **ensure your computer has a backup of the sketch** running on the Arduino.

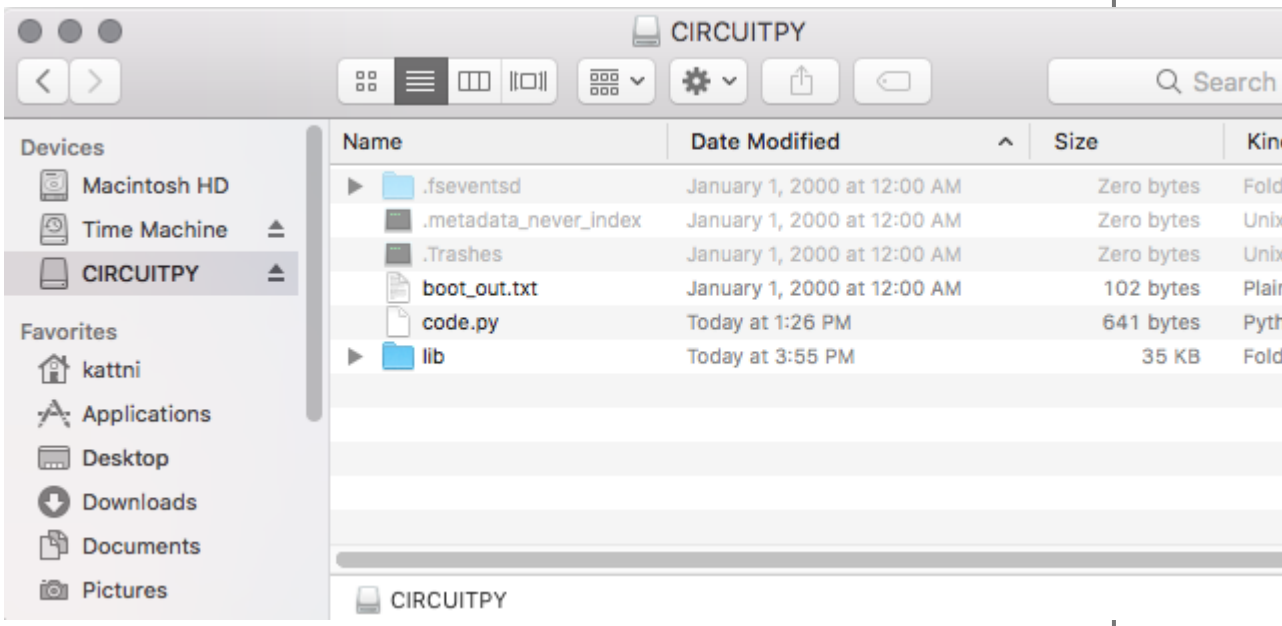
WipperSnapper is not permanently installed on your board, you can switch back to Arduino at any time.

I have a compatible board running CircuitPython but want to try out Offline Mode

WipperSnapper is a different application from CircuitPython. Because of this, installing WipperSnapper will **completely erase the CircuitPython installation and all files on your board.** WipperSnapper is not permanently installed on your board, you can switch back to CircuitPython at any time.

Before installing WipperSnapper, verify and back up your CircuitPython Installation by following the instructions below:

When you plug a CircuitPython board into your computer, your computer will see the board's flash memory as a USB flash drive where files can be stored. When the board is running CircuitPython, you'll see the **CIRCUITPY** drive.



If you see the CIRCUITPY drive appear as a USB flash drive:

- Copy all files to a safe location on your computer, like your desktop. **code.py** is the most important file on this drive as it contains your CircuitPython code.

If you do not see the CIRCUITPY drive and instead see a drive named boardnameBOOT (or similar such as one of these drive names (https://adafru.it/Amd)):

- Locate the reset button on your board. It's a small, black button, and on most of the boards, it will be the only button available. It is typically labeled **RESET** or **RST** on the board.
- Tap this button once
- If the drive changes from boardnameBOOT to **CIRCUITPY**, follow the instructions above to back up its contents
- If it stays as boardnameBOOT, the board does not have CircuitPython installed. You may proceed to the next step.

Once you've backed up your CIRCUITPY drive, proceed to the next steps to download and install the WipperSnapper firmware.

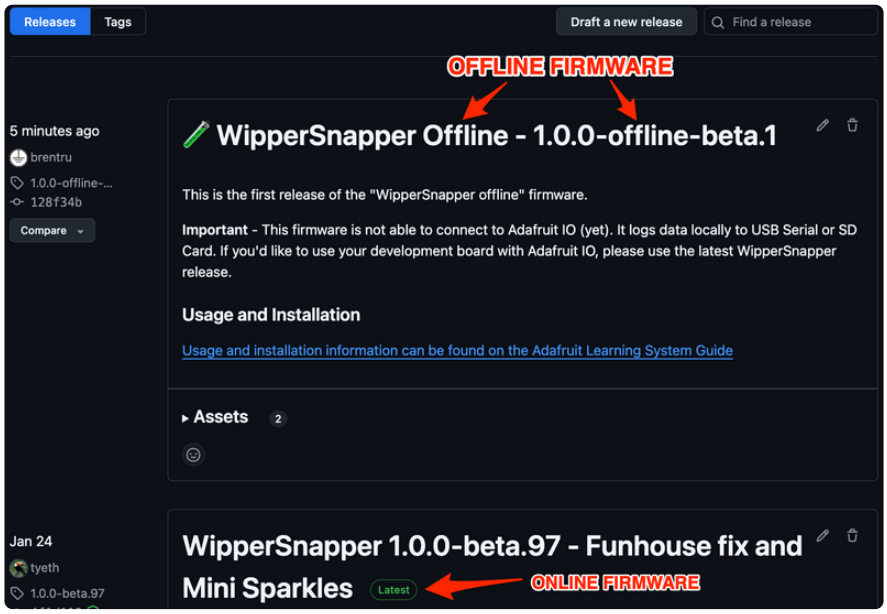
Download WipperSnapper Firmware with Offline Mode

WipperSnapper will ultimately be released as a single firmware file that supports online **and** offline modes. But, for now, WipperSnapper is split into two different files/firmware:

- WipperSnapper (Online) - Downloadable from io.adafruit.com/devices/new (https://adafru.it/192f)
- WipperSnapper (Offline) - Only downloadable from the WipperSnapper Arduino GitHub repository (https://adafru.it/ZYc)

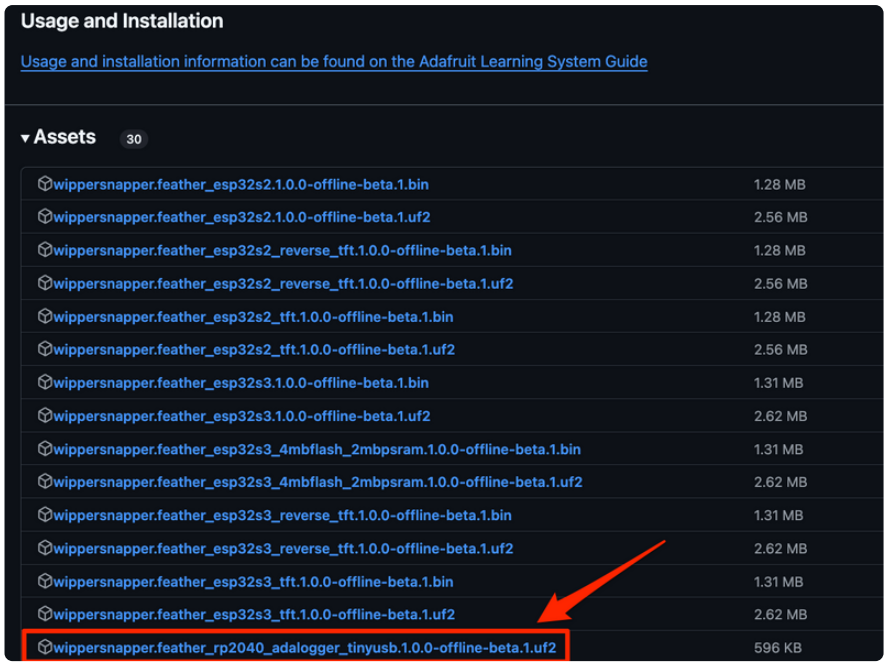
To download WipperSnapper with Offline Mode, navigate to the releases page for the WipperSnapper Arduino project. (https://adafru.it/V5D)

From the releases page, click the release for the **latest version of the offline firmware.** The offline firmware will have "offline" in the title.

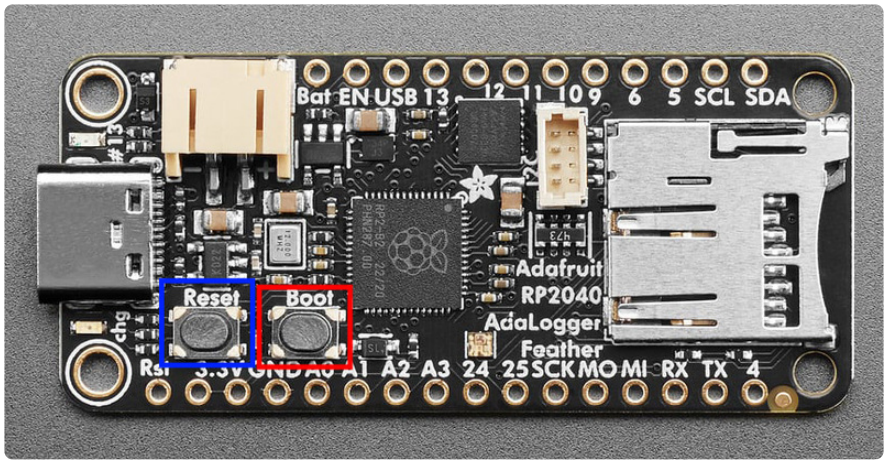


Under **Assets**, select the UF2 file for the Adafruit RP2040 FeatherWing. It should look something like **wippersnapper.feather_rp2040_adalogger_tinyusb.1.0.0-offline-beta.x.uf2.**

Save the file to your computer's desktop.



Install WipperSnapper Offline Firmware

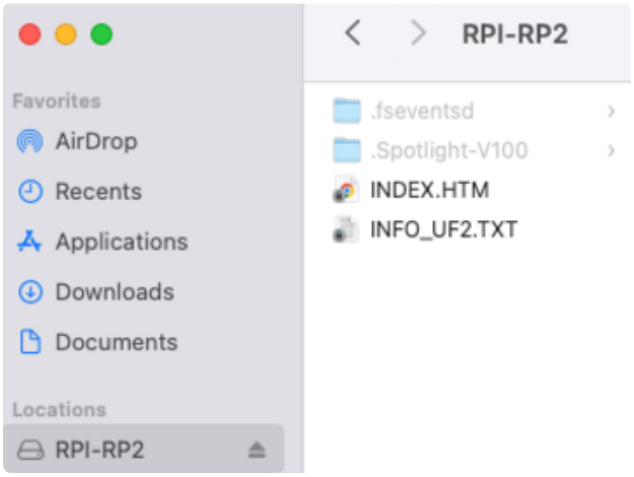


To enter the bootloader, hold down the **BOOT/BOOTSEL button** (highlighted in red above), and while continuing to hold it (don't let go!), press and release the **reset button** (highlighted in red or blue above). **Continue to hold the BOOT/BOOTSEL button until the RPI-RP2 drive appears!**

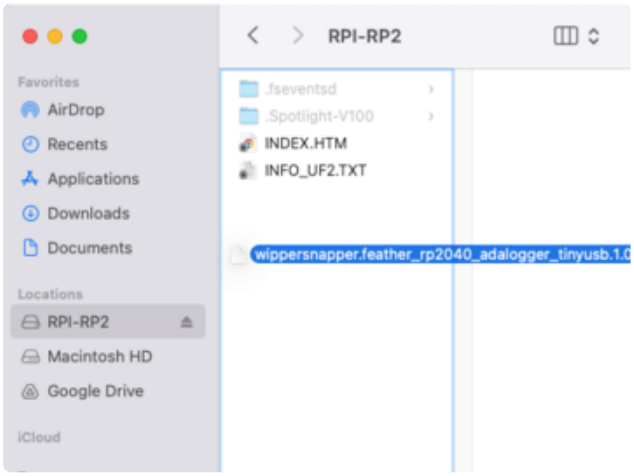
If the drive does not appear, release all the buttons, and then repeat the process above.

You can also start with your board unplugged from USB, press and hold the BOOTSEL button (highlighted in red above), continue to hold it while plugging it into USB, and wait for the drive to appear before releasing the button.

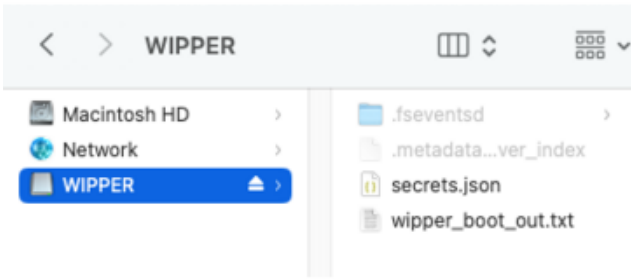
A lot of people end up using charge-only USB cables and it is very frustrating! **Make sure you have a USB cable you know is good for data sync.**



You will see a new disk drive appear called **RPI-RP2**.



Drag the **wippersnapper.feather_rp2040_adalogger_tinyusb.1.0.0-offline-beta.1.uf2** file to **RPI-RP2.**



The **RPI-RP2** drive will disappear and a new disk drive called **WIPPER** will appear.

That's it, you're done! :)

Flash Resetting UF2

If your board ever gets into a really weird state and the **WIPPER** doesn't show up as a disk drive after installing WipperSnapper, try loading this 'nuke' UF2 to RPI-RP2. which will do a 'deep clean' on your Flash Memory. **You will lose all the files on the board**, but at least you'll be able to revive it!

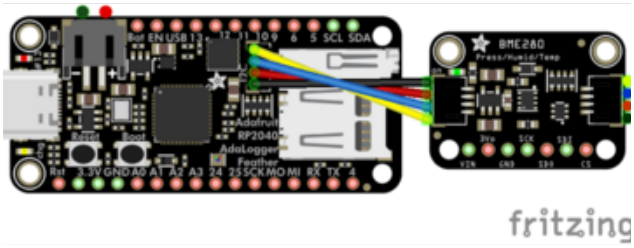
After loading this UF2, follow the steps above to re-install WipperSnapper.

Download Flash "Nuke"/Reset UF2 file for RP2040

https://adafru.it/1afi

Usage

Wiring



Connect the BME280 STEMMA QT board to the Feather RP2040 Adalogger's STEMMA QT port with a STEMMA QT cable.

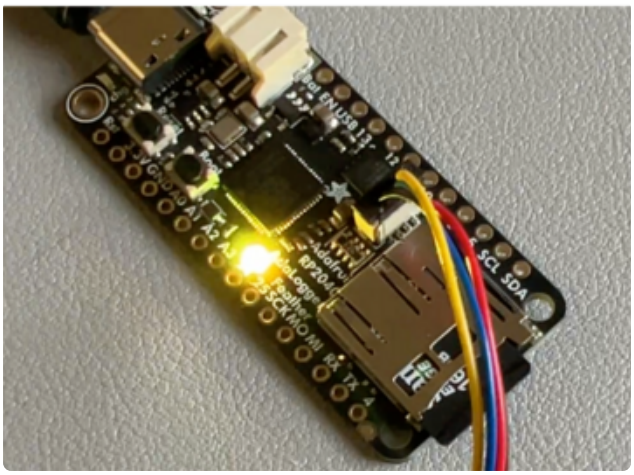
Add the Configuration File

Offline mode relies on a configuration file called **config.json.** This file contains information about the hardware and what's connected to it.

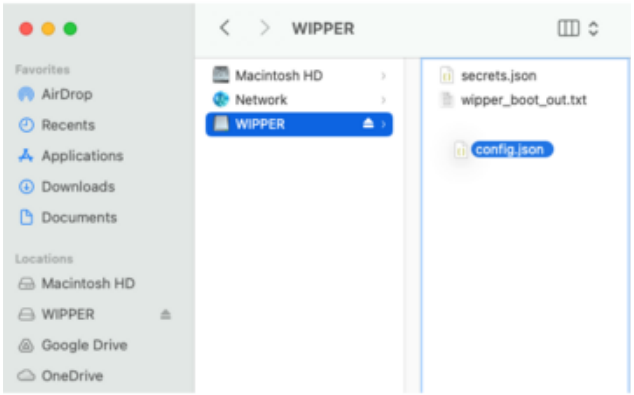
Below is an example configuration file for an RP2040 AdaLogger connected to a BME280 sensor. Save this file to your computer as **config.json**.

```
{
    "exportedFromDevice": {
        "referenceVoltage": 3.3,
        "totalGPIOPins": 18,
        "totalAnalogPins": 4,
        "sd_cs_pin": 23,
        "statusLEDBrightness": 0.5
    },
    "components": [
        {
            "name": "BME280 Sensor",
            "componentAPI": "i2c",
            "i2cDeviceName": "bme280",
            "period": 15,
            "i2cDeviceAddress": "0x77",
            "i2cDeviceSensorTypes": [
                {"type": "relative-humidity"},
                {"type": "ambient-temp"},
                {"type": "ambient-temp-fahrenheit"},
                {"type": "pressure"},
                {"type": "altitude"}
            ]
        }
```

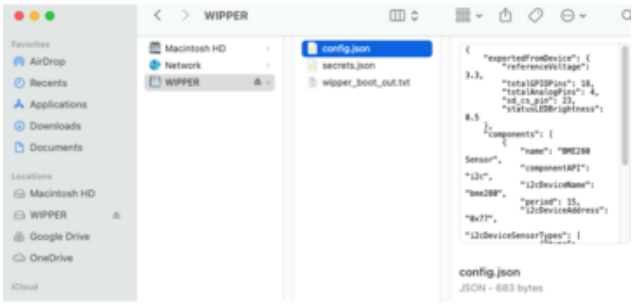
```
    ]
}
```



Connect the RP2040 Adalogger Feather to your computer's USB. You should see the Adalogger Feather's pixel glow yellow.



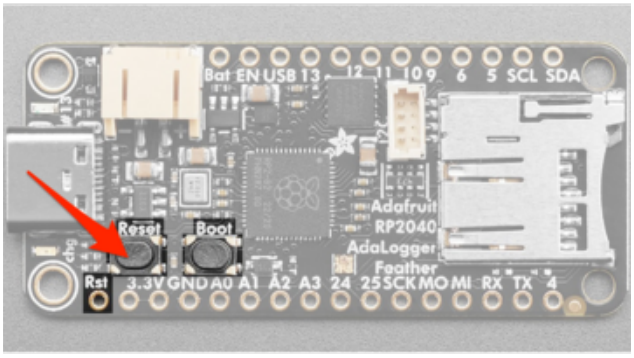
On your computer, **drag and drop the config.json file to the WIPPER drive**.



Verify that WIPPER drive contains the following files:

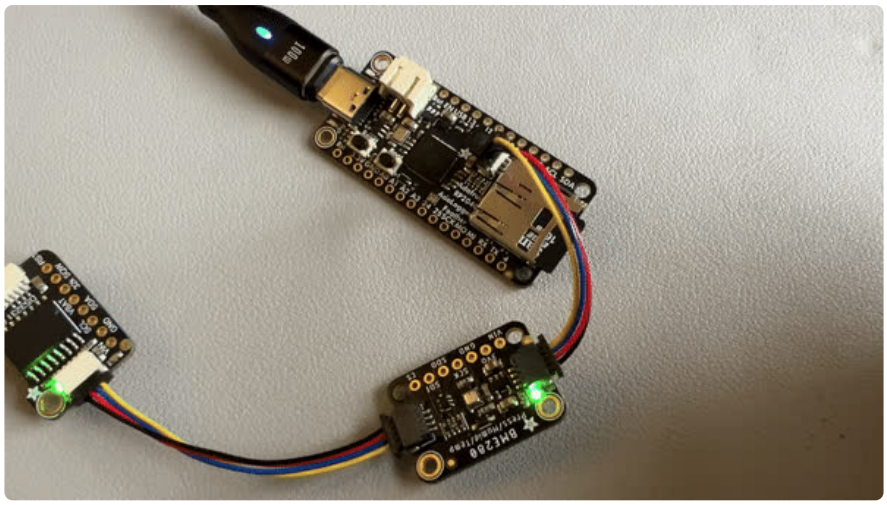
config.json
secrets.json
wipper_boot_out.txt

Then, **press the RESET button on your board.** The board should restart and "load" the new **config.json** file.



Once the Feather reboots, its status pixel should briefly blink the color green to indicate that the hardware has been successfully configured and data is logging into the MicroSD card.

A new **.log** file is created on the microSD card each time the Feather is rebooted.



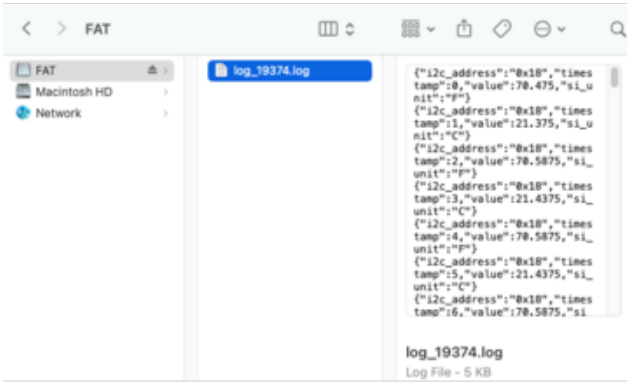
Every 15 seconds, the Feather reads the BME280 sensor's data and writes this data to the MicroSD card. The Feather's LED blinks green every few minutes to indicate that it is still logging data.

Do not remove the microSD card while the Feather is connected to power.

Viewing the microSD Card's Data

When you are ready to view the logged data, first unplug the Feather from USB power. Then, remove the microSD card and insert it into your computer using a microSD card reader (http://adafru.it/5212).

ALWAYS disconnect power from the Feather first, before removing the microSD card!



The SD card contains a .log file (or multiple log files, if you've rebooted your Feather).

About the Log File

Each .log file uses the JSON Lines text file format. (https://adafru.it/1aft) It's a great format for storing log files!

We opted to use JSONL instead of CSV files because if power is removed from the logger while it's logging, the file may become corrupted. Attempting to rescue a corrupted CSV file is very difficult (https://adafru.it/1afu) and often requires guesswork to reconstruct the file. On the other hand, JSONL files are easy to read, each new line in the file is a JSON-encoded line of text. Each log line follows the previous line formatting, making it easy to see an error in a potentially corrupted file and reconstruct the line.

Reading the Log File

Each line of the log file corresponds to a different component within the **config.json** file.

For example, the **config.json** file on the board defines a BME280 sensor reading specific data.

```
...
    "components": [
        {
            "name": "BME280 Sensor",
            "componentAPI": "i2c",
            "i2cDeviceName": "bme280",
            "period": 15,
            "i2cDeviceAddress": "0x77",
            "i2cDeviceSensorTypes": [
                {"type": "ambient-temp"},
                {"type": "ambient-temp-fahrenheit"},
                {"type": "relative-humidity"},
                {"type": "pressure"},
                {"type": "altitude"}
            ]
 ...
```

While a few sensor `type`s are listed above, a lot more are supported by WipperSnapper depending on what sensor you are using.

The sensor types we support are:

Environmental Measurements

- ambient-temp
- ambient-temp-fahrenheit
- pressure
- relative-humidity
- altitude
- co2

- eco2
- tvoc
- gas-resistance
- voc-index
- nox-index

Particulate Matter

- pm10-std
- pm25-std
- pm100-std
- pm10-env
- pm25-env
- pm100-env

Temperature

- object-temp
- object-temp-fahrenheit

Electrical

- voltage
- current

Motion & Position

- accelerometer
- gyroscope
- gravity
- orientation
- magnetic-field
- linear-acceleration
- rotation-vector
- proximity

Light & Color

- light
- lux
- color

Data Types

- raw
- unitless-percent

- bytes
- boolean

The corresponding log file for the **config.json** file above looks like the following:

```
{"i2c_address":"0x77","timestamp":0,"value":21.84,"si_unit":"C"}
{"i2c_address":"0x77","timestamp":1,"value":71.312,"si_unit":"F"}
{"i2c_address":"0x77","timestamp":2,"value":31.1377,"si_unit":"%"}
{"i2c_address":"0x77","timestamp":3,"value":1001.109,"si_unit":"hPa"}
{"i2c_address":"0x77","timestamp":4,"value":101.5793,"si_unit":"m"}
```

To visualize the data produced - you can convert the JSONLines file to a CSV (https://adafru.it/1afv) or manually parse it in your favorite spreadsheet program (Excel, Google Sheets) or numeric computing program (MATLAB, Jupyter, etc.).

NOTE: This website only works for JSONL files <= 5MB.

Viewing USB-Serial Output

In addition to logging data to a microSD card, WipperSnapper also streams data over USB-serial. The data streamed over USB also uses the JSON Lines text file format (https://adafru.it/1aft).

To view the output, use a serial terminal client (such as PuTTY (https://adafru.it/1afw), TerraTerm (https://adafru.it/1afx), or even the Arduino IDE's serial monitor) to connect to the USB device, and **set the terminal's baud rate to 115200 baud**.

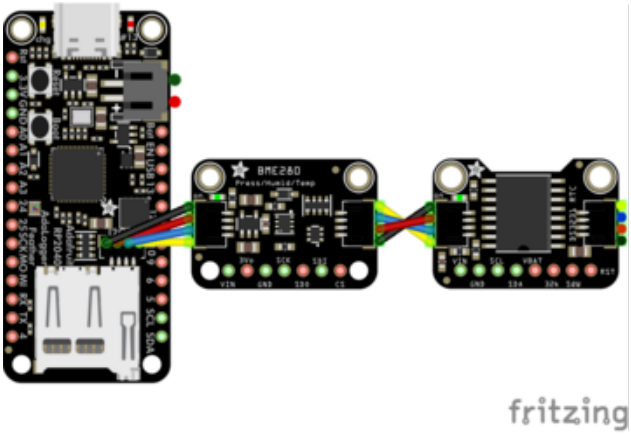
In your serial terminal, you should see the output begin with debugging info and then begin to record data:

```
TERMINAL    SERIAL MONITOR    COMMENTS
RTC Type: DS3231
Begin DS3231 init
Parsing components array...
[SD] I2C component found, decoding JSON to PB...
[Offline] Attempting to configure hardware...
[SD] Created new log file on SD card: log_1740665666.lo
g
-> I2C Device Add/Replace Message Type
[i2c] Decoding I2cDeviceAddOrReplace message...
Creating a new I2C driver
[i2c] Driver initialized and added to controller:
bme280
[Offline] Hardware configured, skipping network setup a
nd running app...
{"i2c_address":"0x77","timestamp":1740665667,"value":21
.26,"si_unit":"C"}
{"i2c_address":"0x77","timestamp":1740665667,"value":70
.268,"si_unit":"F"}
{"i2c_address":"0x77","timestamp":1740665667,"value":32
.09961,"si_unit":"%"}
```

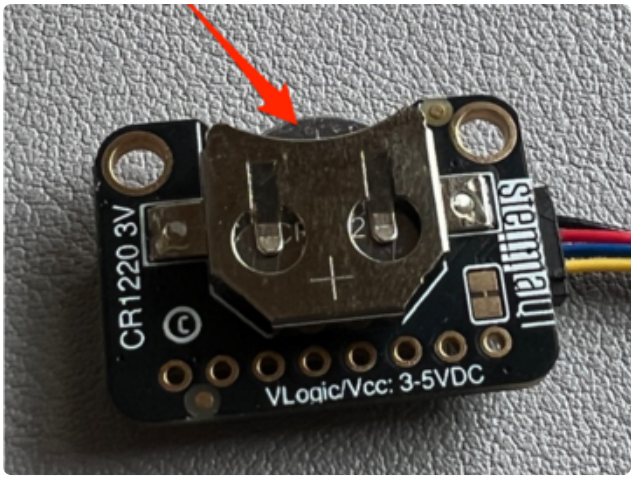
Adding a Real-Time Clock (RTC)

The `timestamp` field in the JSON log increments with each measurement taken, it does not reflect the "actual" time the measurement was taken. To get that, you'll need to connect an RTC module like the DS3231 Precision RTC (http://adafru.it/5188) to your board.

Once an RTC is connected and defined within **config.json**, the log file will automatically include a timestamp (in unix timestamp format (https://adafru.it/1afy)) with the precise time each measurement was read and logged to the microSD card.



Connect the BME280's STEMMA QT port to the DS3231's STEMMA QT port using a STEMMA QT cable.



Don't forget that a Real Time Clock requires a battery backup to operate. A CR1220 size battery goes in the back (http://adafru.it/380), make sure the + symbol on the battery is visible when you insert it!

Connect the Adalogger to your computer, the **WIPPER** drive should appear.

Open the **config.json** file in a text editor and make the following modification to the `exportedFromDevice` array to add a `rtc` key/value pair.

```
{
    "exportedFromDevice": {
        "referenceVoltage": 3.3,
        "totalGPIOPins": 18,
        "totalAnalogPins": 4,
        "sd_cs_pin": 23,
        "statusLEDBrightness": 0.5,
        "rtc": "DS3231"
    },
    ...
```

Save the configuration file. Then, **press the Adalogger's RESET button**.

When the Adalogger boots back up, each line in the log file's timestamp will now reflect the actual time (as a UNIX timestamp (https://adafru.it/1afy)):

```
{"i2c_address":"0x77","timestamp":1740661260,"value":22.11,"si_unit":"C"}
{"i2c_address":"0x77","timestamp":1740661260,"value":71.798,"si_unit":"F"}
{"i2c_address":"0x77","timestamp":1740661260,"value":31.82617,"si_unit":"%"}
{"i2c_address":"0x77","timestamp":1740661260,"value":1000.35,"si_unit":"hPa"}
{"i2c_address":"0x77","timestamp":1740661260,"value":107.9562,"si_unit":"m"}
```