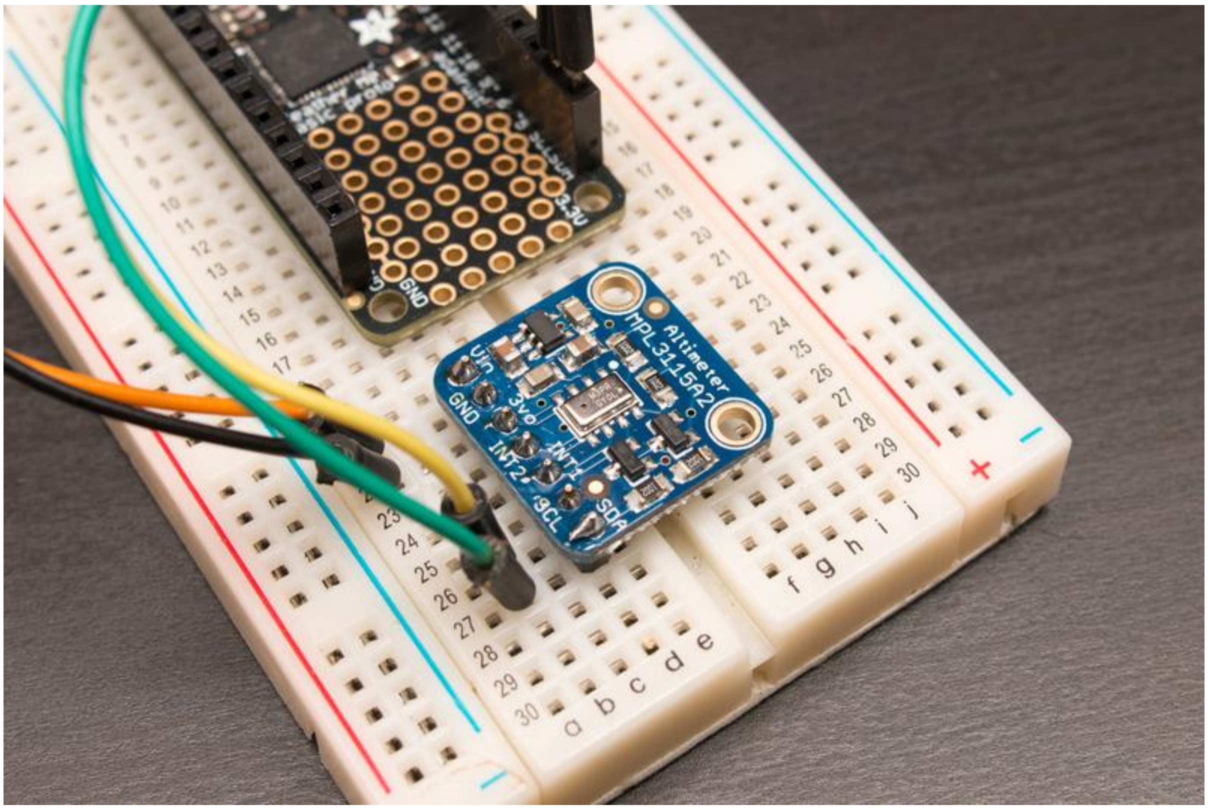
Using MPL3115A2 with CircuitPython

Created by Tony DiCola

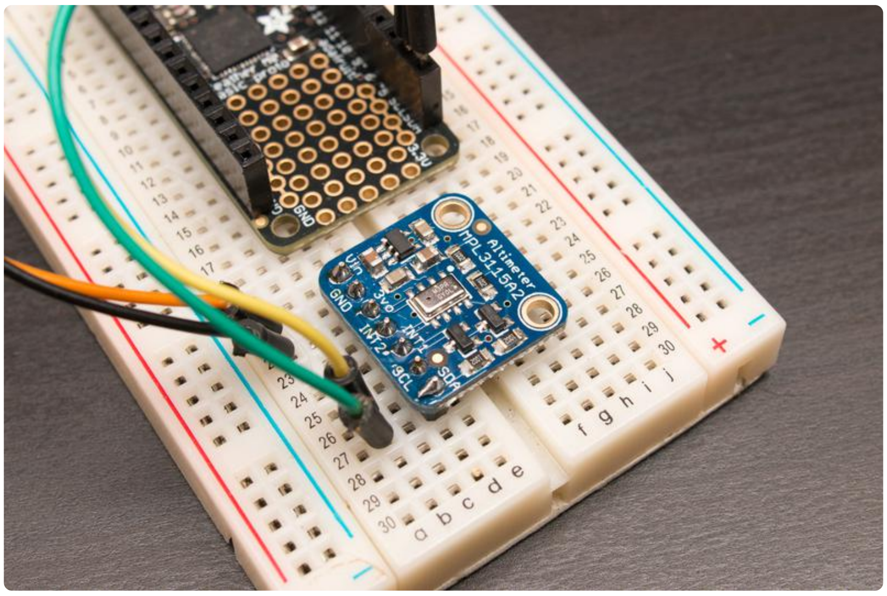


https://learn.adafruit.com/using-mpl3115a2-with-circuitpython

Last updated on 2023-08-29 03:40:16 PM EDT

Table of Contents

Overview



The MPL3115A2 () is a handy high precision barometric pressure that you can read from CircuitPython and Python code! This sensor provides pressure, temperature, and even calculates altitude (based on pressure) automatically for you. Best of all with a simple CircuitPython module you can read data from the sensor with ease. This guide shows how to connect a MPL3115A2 to a CircuitPython board and read data from it with Python code!

Hardware

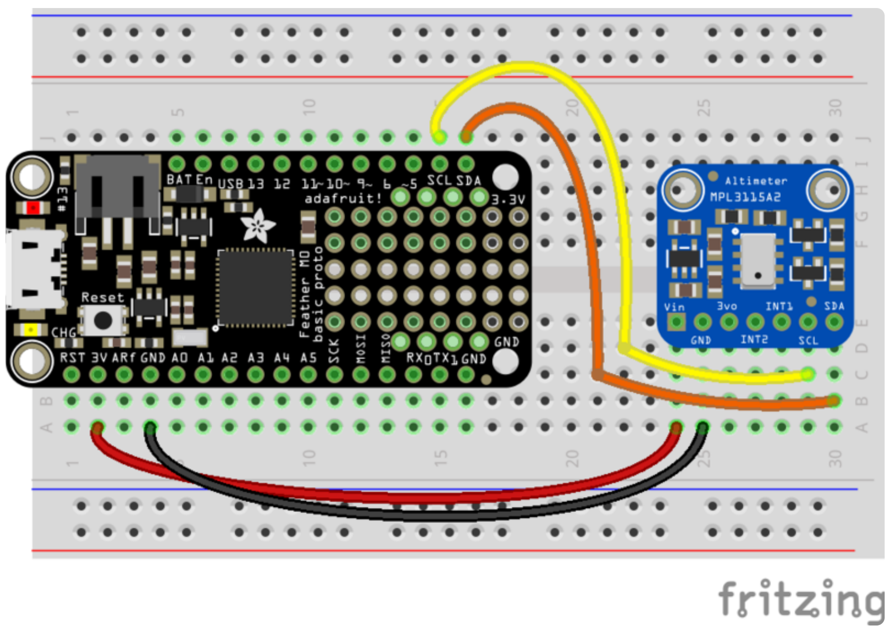
Hardware

To follow this guide you'll need the following parts:

- MPL3115A2 barometric pressure sensor. ()
- A board running CircuitPython. You'll need a board capable of running CircuitPython like the Feather M0 Express, Trinket M0 etc! ().
- Breadboard () and jumper wires (). You'll need these parts to connect components to your development board.

Wiring

Connect your MPL3115A2 to your development board using a standard I2C connection. Here's an example of wiring a MPL3115A2 to a Feather M0 board:



- Board 3.3V output to MPL3115A2 Vin
- Board GND/ground to MPL3115A2 GND
- Board SCL to MPL3115A2 SCL/I2C clock
- Board SDA to MPL3115A2 SDA/I2C data

CircuitPython

Installing Library

To use the MPL3115A2 you'll need to install the Adafruit CircuitPython MPL3115A2 () library on your CircuitPython board.

First make sure you are running the latest version of Adafruit CircuitPython () for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (). Our introduction guide has a great page on how to install the library bundle () for both express and non-express boards.

Remember for non-express boards like the, you'll need to manually install the necessary libraries from the bundle:

- adafruit_mpl3115a2.mpy
- adafruit_bus_device

You can also download the adafruit_mpl3115a2.mpy from its releases page on Github ().

Before continuing make sure your board's lib folder or root filesystem has the adafruit_mpl3115a2.mpy, and adafruit_bus_device files and folders copied over.

Next connect to the board's serial REPL ()so you are at the CircuitPython >>> prompt.

Usage

To demonstrate the usage of the sensor we'll initialize it and read the pressure and other values from the Python REPL.

Run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```
import board
import adafruit_mpl3115a2
i2c = board.I2C()
sensor = adafruit_mpl3115a2.MPL3115A2(i2c)
```

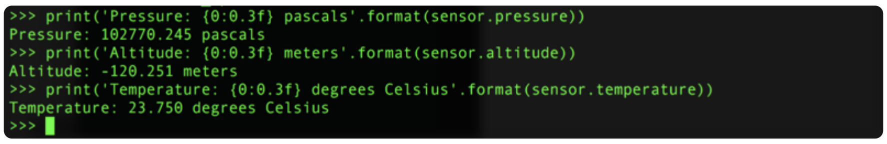
Remember if you're using a board that doesn't support hardware I2C (like the ESP8266) you need to use the bitbangio module instead:

```
import board
import bitbangio
import adafruit_mpl3115a2
i2c = bitbangio.I2C(board.SCL, board.SDA)
sensor = adafruit_mpl3115a2.MPL3115A2(i2c)
```

Now you're ready to read values from the sensor using any of these properties:

- pressure - The barometric pressure in Pascals.
- temperature - The temperature in degrees Celsius.
- altitude - The altitude as calculated using barometric pressure and returned in meters.

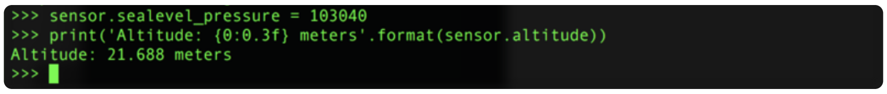
```
print('Pressure: {0:0.3f} pascals'.format(sensor.pressure))
print('Altitude: {0:0.3f} meters'.format(sensor.altitude))
print('Temperature: {0:0.3f} degrees Celsius'.format(sensor.temperature))
```



```
>>> print('Pressure: {0:0.3f} pascals'.format(sensor.pressure))
Pressure: 102770.245 pascals
>>> print('Altitude: {0:0.3f} meters'.format(sensor.altitude))
Altitude: -120.251 meters
>>> print('Temperature: {0:0.3f} degrees Celsius'.format(sensor.temperature))
Temperature: 23.750 degrees Celsius
>>>
```

To make the altitude calculation more accurate you'll want to set the pressure at sea level for your current location and weather conditions. Barometric pressure sensors can only calculate altitude based on the pressure of air they detect and that pressure changes daily and with weather conditions. Look up your local weather forecast and it typically includes the pressure at sea level. Set the sealevel_pressure property (to a value in Pascals) to tell the chip this information and get the most accurate altitude measurements:

```
sensor.sealevel_pressure = 103040
```



```
>>> sensor.sealevel_pressure = 103040
>>> print('Altitude: {0:0.3f} meters'.format(sensor.altitude))
Altitude: 21.688 meters
>>> 
```

That's all there is to using the MPL3115A2 with CircuitPython!

Here's a complete example that will print the pressure, altitude, and temperature every second. Save this as main.py on your board and open the REPL to view the output. Remember to switch to the bitbangio module if necessary for your board:

```
# SPDX-FileCopyrightText: 2019 Tony DiCola for Adafruit Industries
# SPDX-License-Identifier: MIT

# Simple demo of the MPL3115A2 sensor.
# Will read the pressure and temperature and print them out every second.
import time
import board
import adafruit_mpl3115a2


# Create sensor object, communicating over the board's default I2C bus
i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a
microcontroller

# Initialize the MPL3115A2.
sensor = adafruit_mpl3115a2.MPL3115A2(i2c)
# Alternatively you can specify a different I2C address for the device:
# sensor = adafruit_mpl3115a2.MPL3115A2(i2c, address=0x10)

# You can configure the pressure at sealevel to get better altitude estimates.
# This value has to be looked up from your local weather forecast or meteorological
# reports.  It will change day by day and even hour by hour with weather
# changes.  Remember altitude estimation from barometric pressure is not exact!
# Set this to a value in hectopascals:
sensor.sealevel_pressure = 1022.5

# Main loop to read the sensor values and print them every second.
while True:
    pressure = sensor.pressure
    print("Pressure: {0:0.3f} hectopascals".format(pressure))
    altitude = sensor.altitude
    print("Altitude: {0:0.3f} meters".format(altitude))
    temperature = sensor.temperature
    print("Temperature: {0:0.3f} Celsius".format(temperature))
    time.sleep(1.0)
```