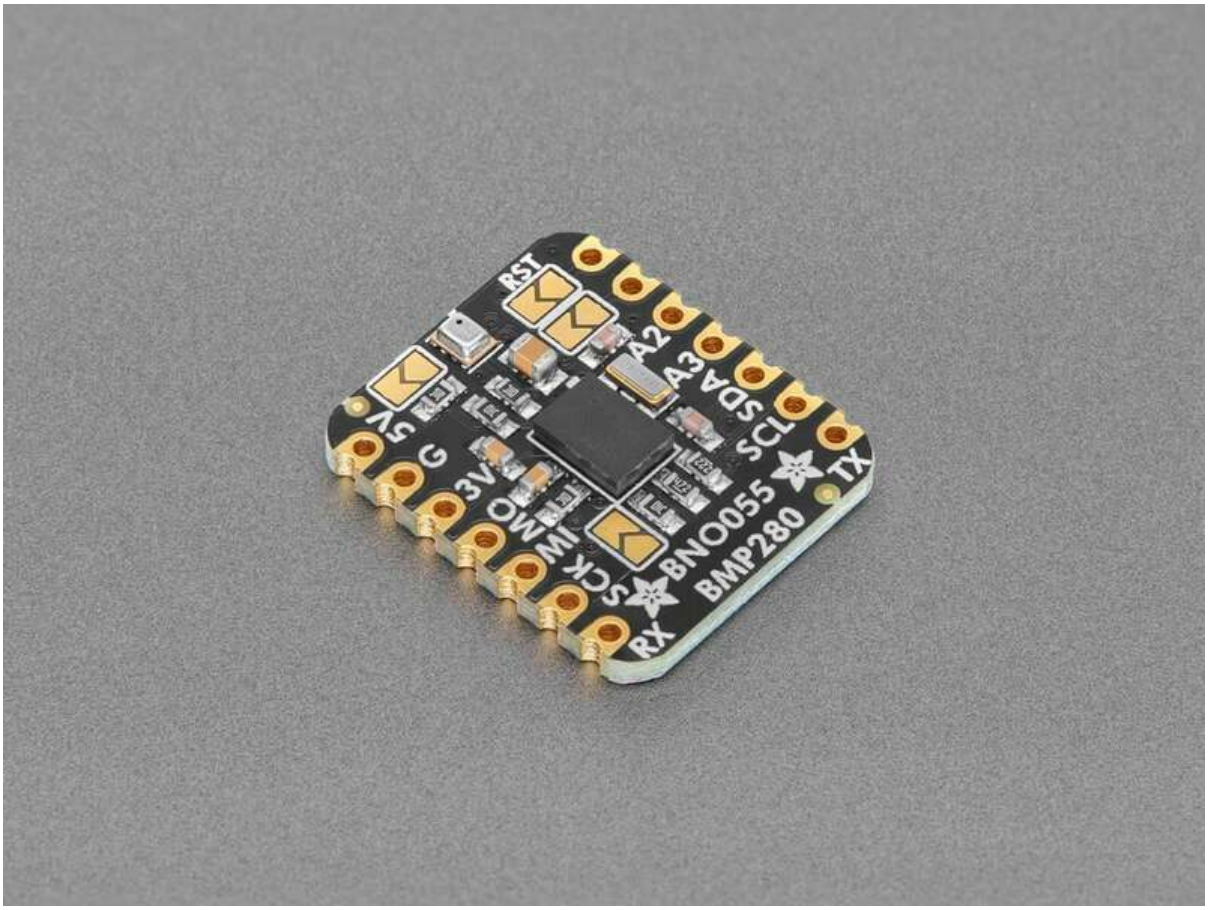
# Adafruit BNO055 + BMP280 BFF

Created by Liz Clark

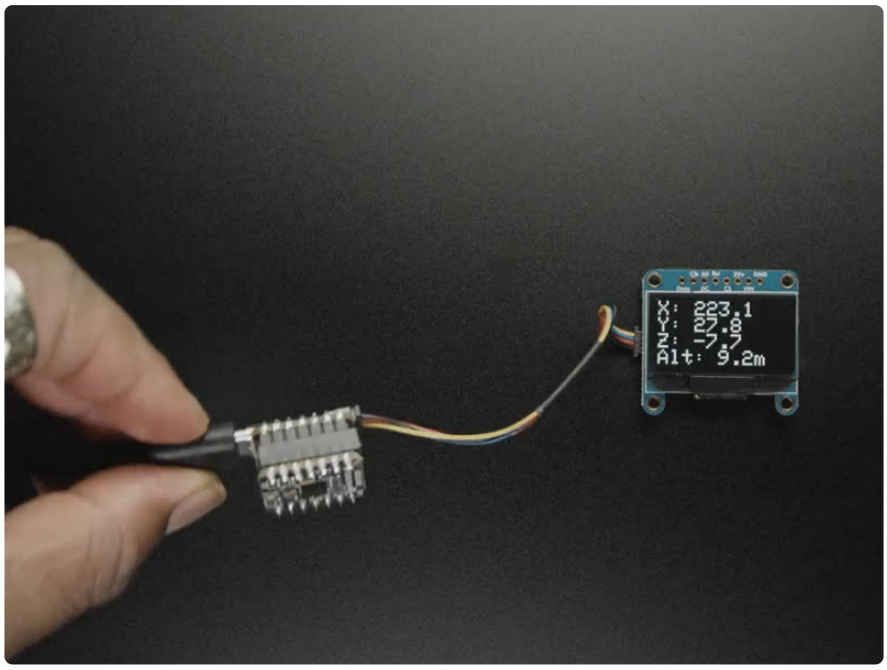



https://learn.adafruit.com/adafruit-bno055-bmp280-bff

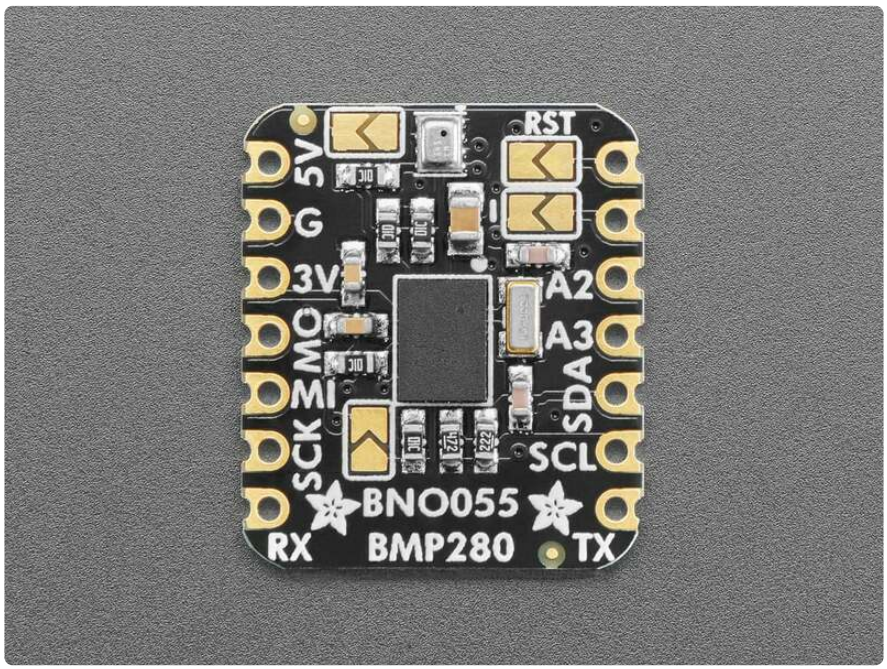
Last updated on 2024-06-03 04:00:24 PM EDT

# Table of Contents

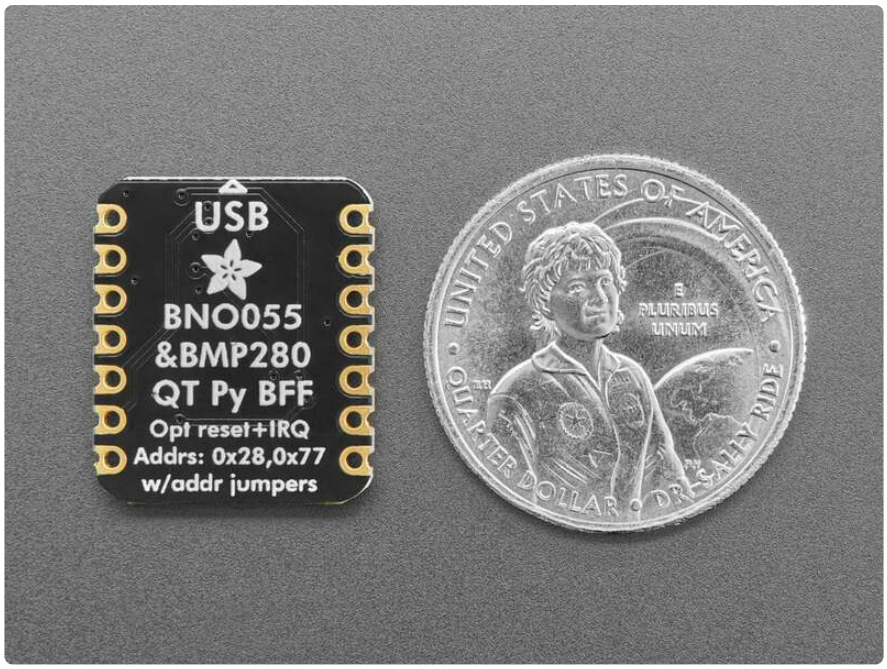
# Overview




Our QT Py boards are a great way to make very small microcontroller projects that pack a ton of power - and now we have a way for you to turn many QT Py boards into powerful 9 degree-of-freedom (9DoF) motion plus pressure/altitude sensing projects that are super small!

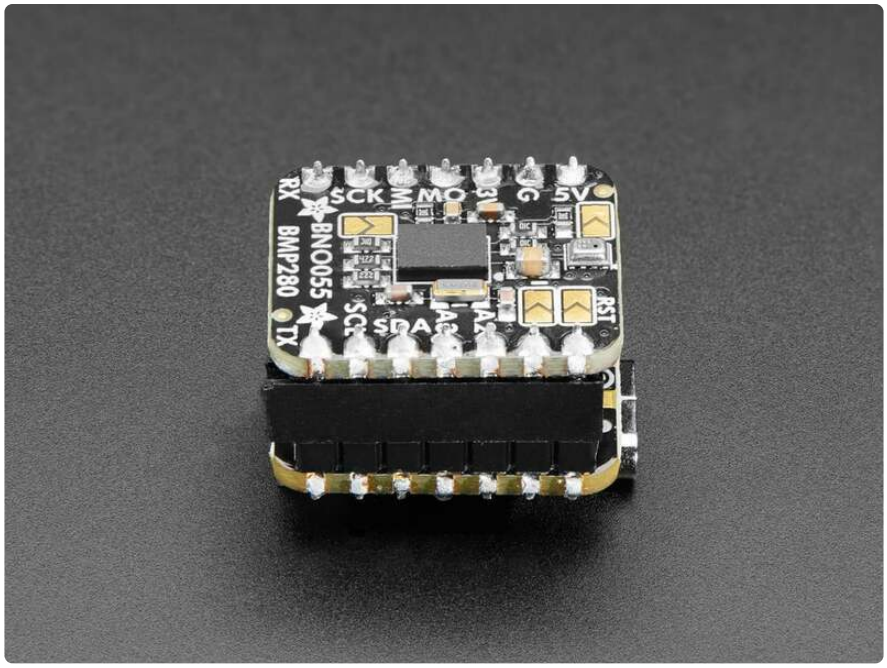



We call this the **Adafruit BNO055 + BMP280 BFF—a "Best Friend Forever."** When you were a kid, you may have learned about the "buddy" system; well, this product is kind of like that! It's a board that will watch your QT Py's back and give it more capabilities.




This PCB is designed to fit onto the back of any QT Py or Xiao board. It can be soldered into place or made removable using pin and socket headers.

Rather than spending weeks or months fiddling with algorithms of varying accuracy and complexity, you can have meaningful sensor data in minutes thanks to the BNO055 (http://adafru.it/2472). This smart 9-DOF sensor does the sensor fusion all on its own! You can read the data right over I2C and Bob's yer uncle.

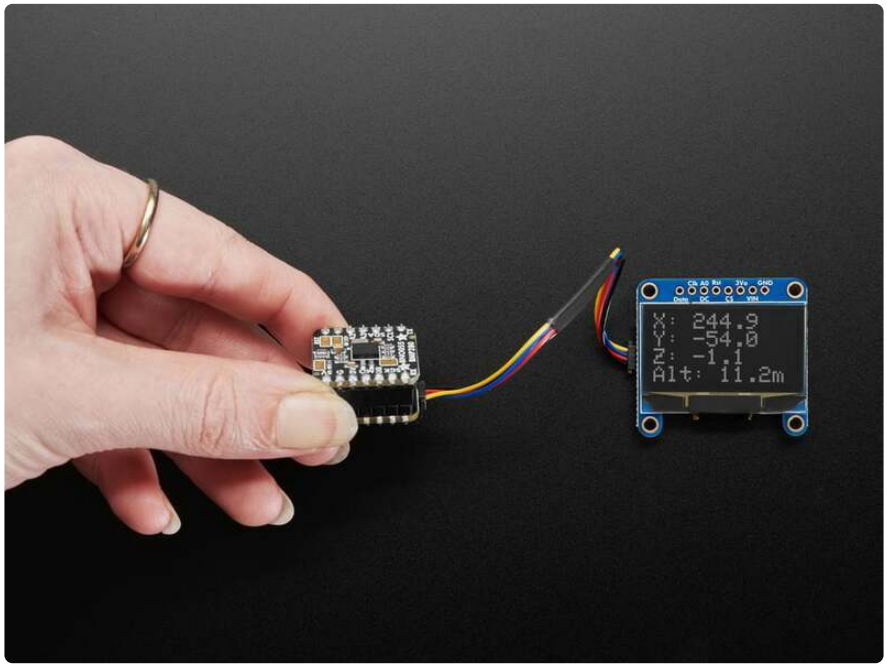



The BNO055 can output the following sensor data:

- **Absolute Orientation** (Euler Vector, 100Hz) Three axis orientation data based on a 360° sphere

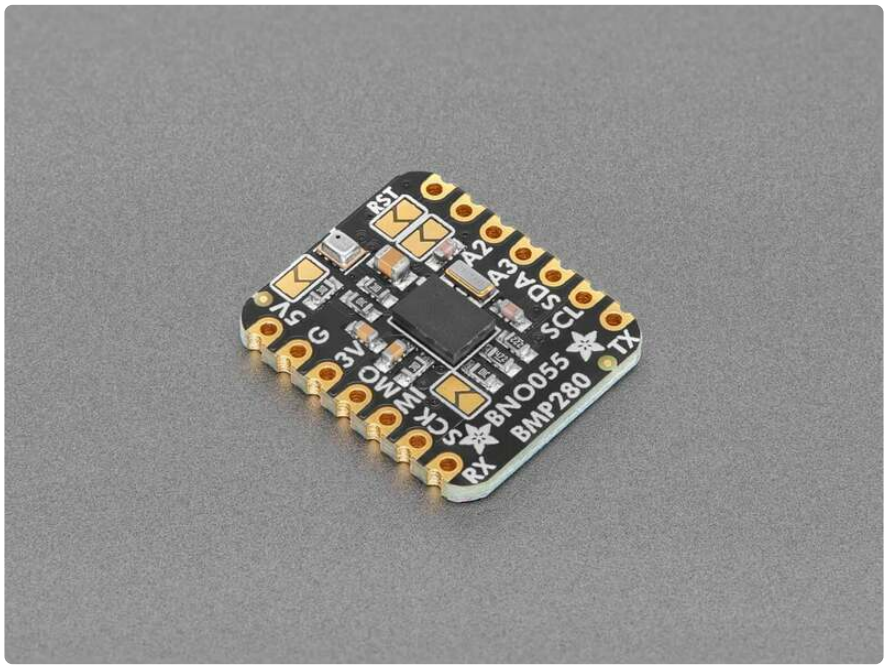
- **Absolute Orientation** (Quaternion, 100Hz) Four point quaternion output for more accurate data manipulation
- **Angular Velocity Vector** (100Hz) Three axis of 'rotation speed' in rad/s
- **Acceleration Vector** (100Hz) Three axis of acceleration (gravity + linear motion) in m/s^2
- **Magnetic Field Strength Vector** (20Hz) Three axis of magnetic field sensing in micro Tesla (uT)
- **Linear Acceleration Vector** (100Hz) Three axis of linear acceleration data (acceleration minus gravity) in m/s^2
- **Gravity Vector** (100Hz) Three axis of gravitational acceleration (minus any movement) in m/s^2
- **Temperature** (1Hz) Ambient temperature in degrees celsius

The only thing it doesn't have is barometric pressure, which can be used for altitude calculations. Aha, that's why there's also a BMP280 sensor (https://adafru.it/19Te) on board. This precision sensor from Bosch is the best low-cost, precision sensing solution for measuring barometric pressure with ±1 hPa absolute accuracy, and temperature with ±1.0°C accuracy. Because pressure changes with altitude and the pressure measurements are so good, you can also use it as an altimeter with ±1 meter accuracy.




Usage for the BFF is simple because both the BNO and BMP talk over I2C - so you'll just use the SDA/SCL pins. Note that some QT Py boards with a Stemma QT port have two I2C ports, so make sure you are using the port that's on the SDA/SCL pins, not the QT port!

There are optional Interrupt/Reset pins for the BNO055 if you want more advanced control, and each sensor also has address-changing jumpers in case you want to switch from the default addresses of BMP280 @ 0x77 and BNO055 @ 0x28.

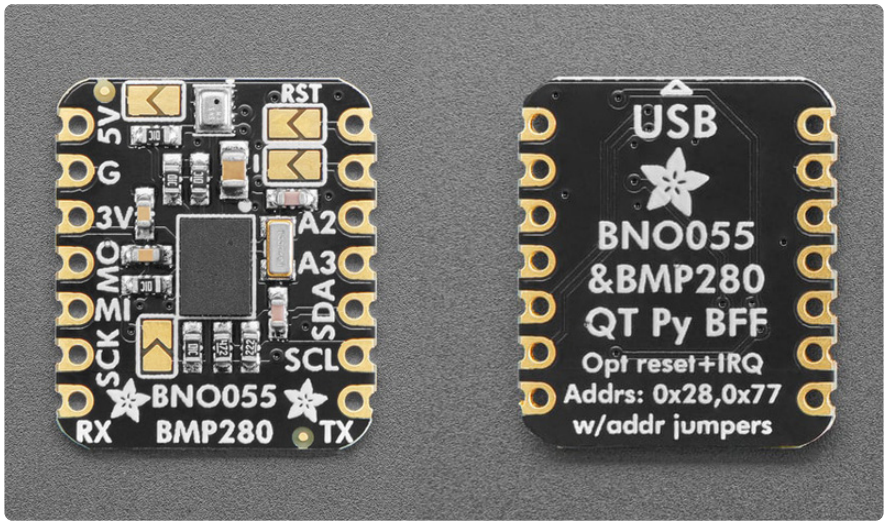



If you'd like, you can solder the BFF directly on the back of your Xiao/QT Py or use the included header to solder each side to your QT Py to make a sandwich. You can also pick up an Itsy Bitsy short female header kit to make it removable but compact (http://adafru.it/4174); you'll need to trim the headers to 7 pins long.

**QT Py is not included!**

---

# Pinouts



The default I2C address for the BNO055 is **0x28** and the default I2C address for the BMP280 is **0x77**.

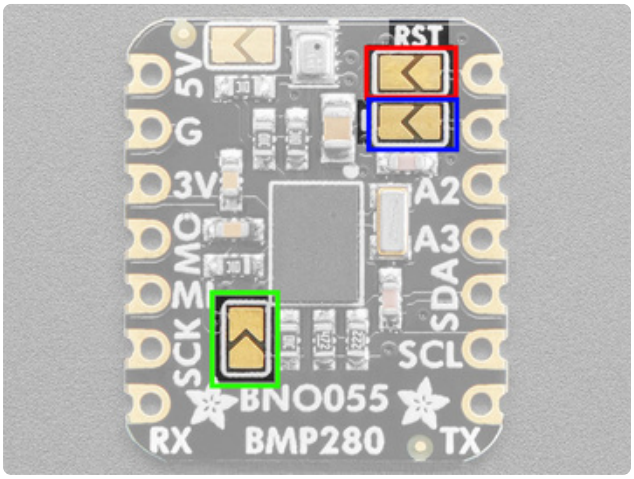
Some QT Py boards with a Stemma QT port have two I2C ports, so make sure you are using the port that's on the SDA/SCL pins, not the QT port in your code.

## BNO055 9-DoF Sensor

In the center of the board is the BNO055 9-DoF sensor. This sensor performs sensor fusion all on its own and you can read the data right over I2C on the SDA/SCL pins at default address **0x28**. You can access the following data from the BNO055:

- **Absolute Orientation** (Euler Vector, 100Hz) Three axis orientation data based on a 360° sphere
- **Absolute Orientation** (Quaternion, 100Hz) Four point quaternion output for more accurate data manipulation
- **Angular Velocity Vector** (100Hz) Three axis of 'rotation speed' in rad/s
- **Acceleration Vector** (100Hz) Three axis of acceleration (gravity + linear motion) in m/s^2
- **Magnetic Field Strength Vector** (20Hz) Three axis of magnetic field sensing in micro Tesla (uT)
- **Linear Acceleration Vector** (100Hz) Three axis of linear acceleration data (acceleration minus gravity) in m/s^2
- **Gravity Vector** (100Hz) Three axis of gravitational acceleration (minus any movement) in m/s^2
- **Temperature** (1Hz) Ambient temperature in degrees celsius

## BNO055 Jumpers




**RST** - The reset pin for the BNO055 is located next to pin **A0** (outlined in red in the board image). This jumper is open/not connected by default. If you solder the jumper closed, it will connect the reset pin to **A0**.

**I** - The interrupt pin for the BNO055 is located next to pin **A1** (outlined in blue in the board image). This jumper is open/not connected by default. If you solder the jumper closed, it will connect the interrupt pin to **A1**.

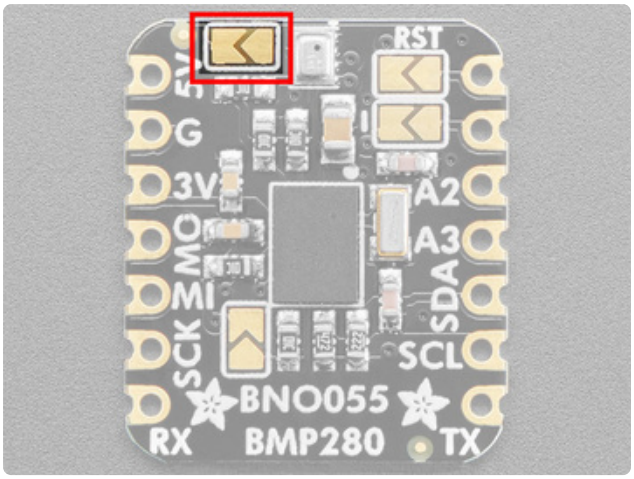
**Address jumper** - The I2C address changing jumper for the BNO055 is located to the left of the sensor (outlined in green in the board image). This jumper is open/not connected by default. If you solder the jumper closed, it will change the BNO055 I2C address from **0x28** to **0x29**.

## BMP280 Pressure Sensor

At the top edge of the board is the BMP280. You can read the BMP280 over I2C on the SDA/SCL pins at default address **0x77**. This sensor is used for:

- **Barometric pressure** (hPa) with ±1 hPa absolute accuracy
- **Temperature** (°C) with ±1.0°C accuracy
- **Altimeter** (meters) with ±1 meter accuracy

## BMP280 Address Jumper




**Address jumper** - The I2C address changing jumper for the BMP280 is located to the left of the sensor (outlined in red in the board image). This jumper is open/not connected by default. If you solder the jumper closed, it will change the BMP280 I2C address from **0x77** to **0x76**.

---

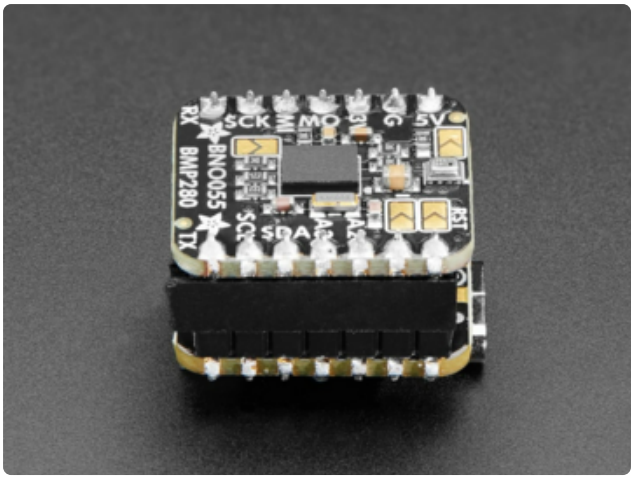
# CircuitPython

It's easy to use the **BNO055** + **BMP280 BFF** with CircuitPython and the Adafruit_CircuitPython_BNO055 (https://adafru.it/C43) and Adafruit_CircuitPython_BMP280 (https://adafru.it/C0x) modules. These modules allow you to easily write Python code to control the BNO055 9-DoF sensor and the BMP280 pressure sensor.

The BNO055 I2C implementation violates the I2C protocol in some circumstances. This causes it not to work well with certain chip families. It does not work well with Espressif ESP32, ESP32-S3, and NXP i.MX RT1011, and it does not work well with I2C multiplexers. Operation with SAMD51, RP2040, STM32F4, and nRF52840 is more reliable.

## CircuitPython Microcontroller Wiring

Plug a BNO055 + BMP280 BFF into your QT Py or Xiao form factor board exactly as shown below. Here's an example of connecting a QT Py RP2040 to the BFF.




Connect the QT Py RP2040 with plug headers into the BNO055 + BMP280 BFF with socket headers. They should be plugged in with the backs of the boards facing each other.

For more information on soldering socket headers, check out this Learn Guide (https://adafru.it/18gf).

**How to Solder Headers Learn Guide**

https://adafru.it/18gf

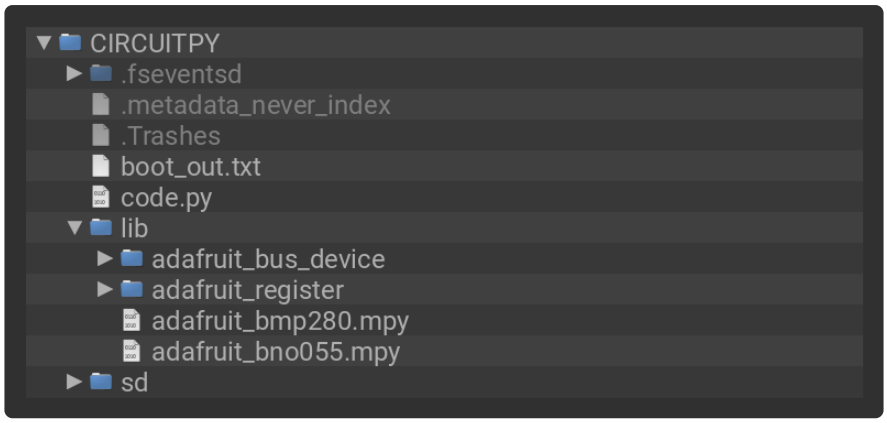
## CircuitPython Usage

To use with CircuitPython, you need to first install the **Adafruit_CircuitPython_BNO055** and **Adafruit_CircuitPython_BMP280** libraries, and the dependencies, into the **lib** folder on your **CIRCUITPY** drive. Then you need to update **code.py** with the example script.

Thankfully, we can do this in one go. In the example below, click the **Download Project Bundle** button below to download the necessary libraries and the **code.py** file in a zip file. Extract the contents of the zip file, and copy the **entire lib folder** and the **code.py** file to your **CIRCUITPY** drive.

Your **CIRCUITPY/lib** folder should contain the following folders and files:

- **adafruit_bus_device/**
- **adafruit_register/**
- **adafruit_bmp280.mpy**
- **adafruit_bno055.mpy**




## Example Code

Once everything is saved to the **CIRCUITPY** drive, connect to the serial console (https://adafru.it/Bec) to see the data printed out!

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT
#
# BNO055 + BMP280 BFF Demo

import time
import board
import adafruit_bno055
import adafruit_bmp280

i2c = board.I2C()  # uses board.SCL and board.SDA
bno055 = adafruit_bno055.BNO055_I2C(i2c)

bmp280 = adafruit_bmp280.Adafruit_BMP280_I2C(i2c)
bmp280.sea_level_pressure = 1013.25

while True:
    print(f"Temperature: {bmp280.temperature:0.1f} C")
    print(f"Pressure: {bmp280.pressure:0.1f} hPa")
    print(f"Altitude = {bmp280.altitude:0.2f} meters")
    print(f"Accelerometer (m/s^2): {bno055.acceleration}")
    print(f"Magnetometer (microteslas): {bno055.magnetic}")
    print(f"Gyroscope (rad/sec): {bno055.gyro}")
    print(f"Euler angle: {bno055.euler}")
    print(f"Quaternion: {bno055.quaternion}")
    print(f"Linear acceleration (m/s^2): {bno055.linear_acceleration}")
    print(f"Gravity (m/s^2): {bno055.gravity}")
    print()

    time.sleep(1)
```

First, both sensors are instantiated over I2C. In the loop, all of the available data parameters from both sensors are printed to the serial monitor every 2 seconds.

```
CircuitPython REPL

Temperature: 27.8 C
Pressure: 1012.6 hPa
Altitude = 5.08 meters
Accelerometer (m/s^2): (-4.56, -4.83, 5.15)
Magnetometer (microteslas): (23.0, 38.1875, -18.0)
Gyroscope (rad/sec): (1.25773, 3.52229, 3.93136)
Euler angle: (302.875, -21.125, 42.375)
Quaternion: (0.807007, -0.394592, 0.00878906, 0.43927)
Linear acceleration (m/s^2): (-0.75, 0.95, -2.49)
Gravity (m/s^2): (-3.99, -5.96, 6.68)

Temperature: 27.8 C
Pressure: 1012.6 hPa
Altitude = 5.04 meters
Accelerometer (m/s^2): (2.83, 7.94, 4.68)
Magnetometer (microteslas): (7.0625, -37.875, -32.25)
Gyroscope (rad/sec): (0.0829032, -0.0239983, -0.0119991)
Euler angle: (286.875, 18.625, -59.25)
Quaternion: (0.691711, 0.490906, 0.13269, 0.512756)
Linear acceleration (m/s^2): (-0.06, 0.01, 0.09)
Gravity (m/s^2): (3.13, 7.99, 4.72)
```

# BMP280 Python Docs

BMP280 Python Docs (https://adafru.it/19Tb)

# BNO055 Python Docs

BNO055 Python Docs (https://adafru.it/19Cg)

# Arduino

Using the BNO055 + BMP280 BFF with Arduino involves involves plugging the breakout into your Arduino-compatible QT Py or Xiao form factor board, installing the Adafruit_BNO055 (https://adafru.it/f0l) and Adafruit_BMP280 (https://adafru.it/flK) libraries, and running the provided example code.

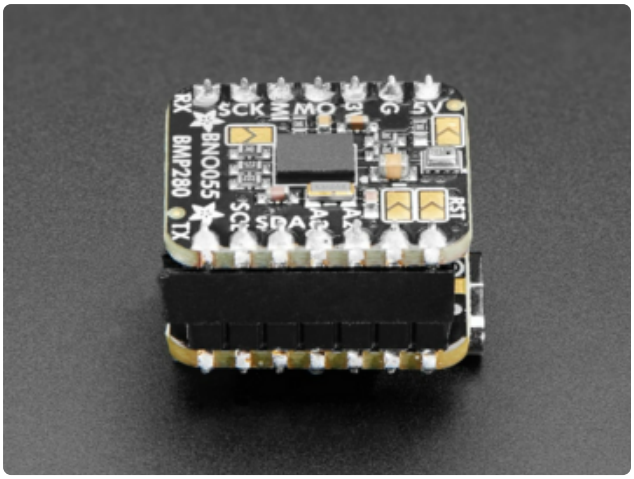
If you are using an Espressif board (ESP32/S2/S3) make sure you are using board support package 3.0.0 (https://adafru.it/19Uf) or newer to avoid issues with I2C clock-stretching.

The BNO055 I2C implementation violates the I2C protocol in some circumstances. This causes it not to work well with certain chip families. It does not work well with I2C multiplexers or Espressif chips with older board support

packages. Operation with SAMD51, RP2040, STM32F4, nRF52840 and Espressif with BSP 3.0.0 or newer are more reliable.

## Wiring

Plug a BNO055 + BMP280 BFF into your QT Py or Xiao form factor board exactly as shown below. Here's an example of connecting a QT Py RP2040 to the BFF.




Connect the QT Py RP2040 with plug headers into the BNO055 + BMP280 BFF with socket headers. They should be plugged in with the backs of the boards facing each other.

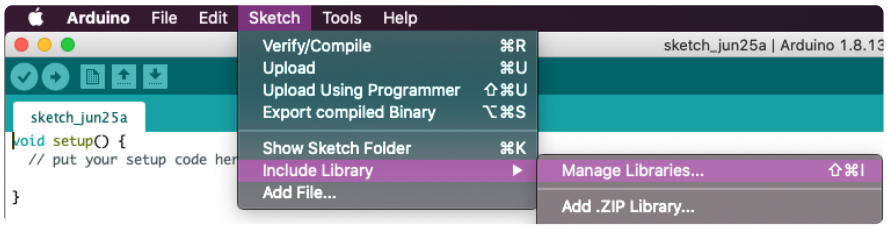
For more information on soldering socket headers, check out this Learn Guide (https://adafru.it/18gf).

**How to Solder Headers Learn Guide**

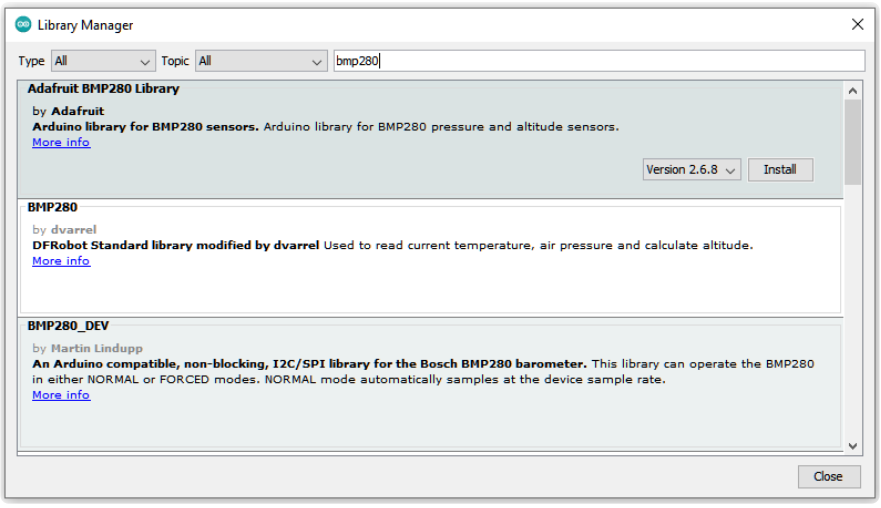
https://adafru.it/18gf

## Library Installation

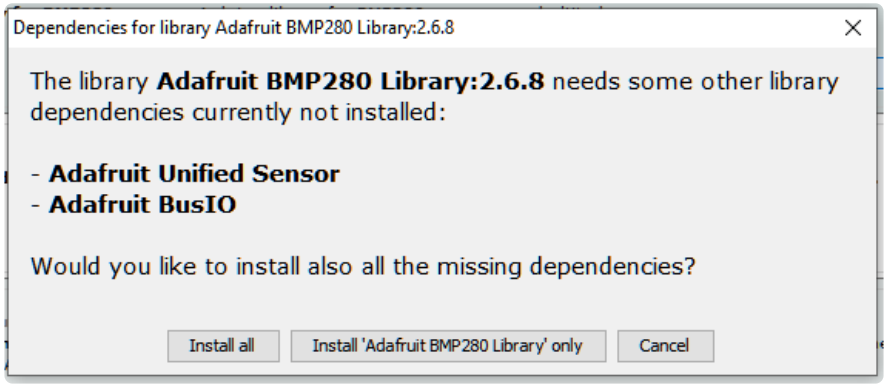
You can install the **Adafruit BNO055** and **Adafruit BMP280** libraries for Arduino using the Library Manager in the Arduino IDE.




Click the **Manage Libraries ...** menu item, search for **Adafruit BMP280** and select the **Adafruit BMP280** library:



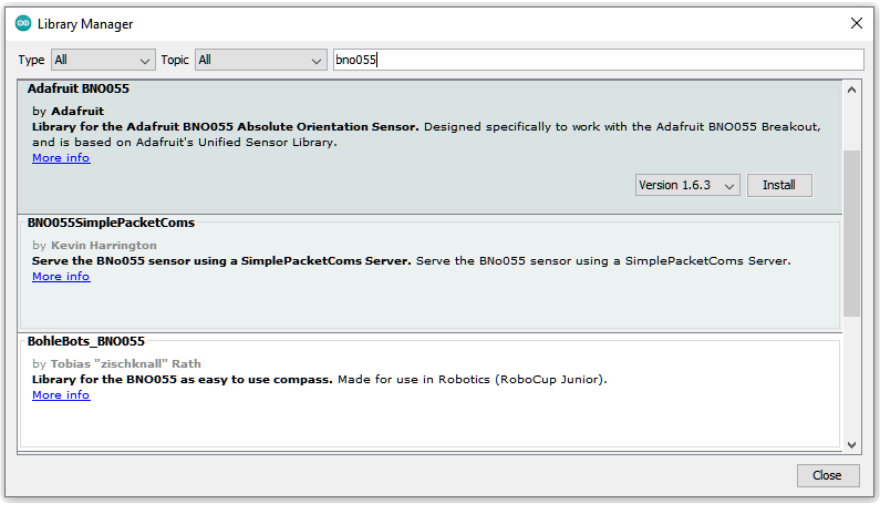

If asked about dependencies, click "Install all".




If the "Dependencies" window does not come up, then you already have the dependencies installed.

If the dependencies are already installed, you must make sure you update them through the Arduino Library Manager before loading the example!

Then, search for **Adafruit BNO055** and select the **Adafruit BNO055** library:




The BNO055 library has the same library dependencies as the BMP280.

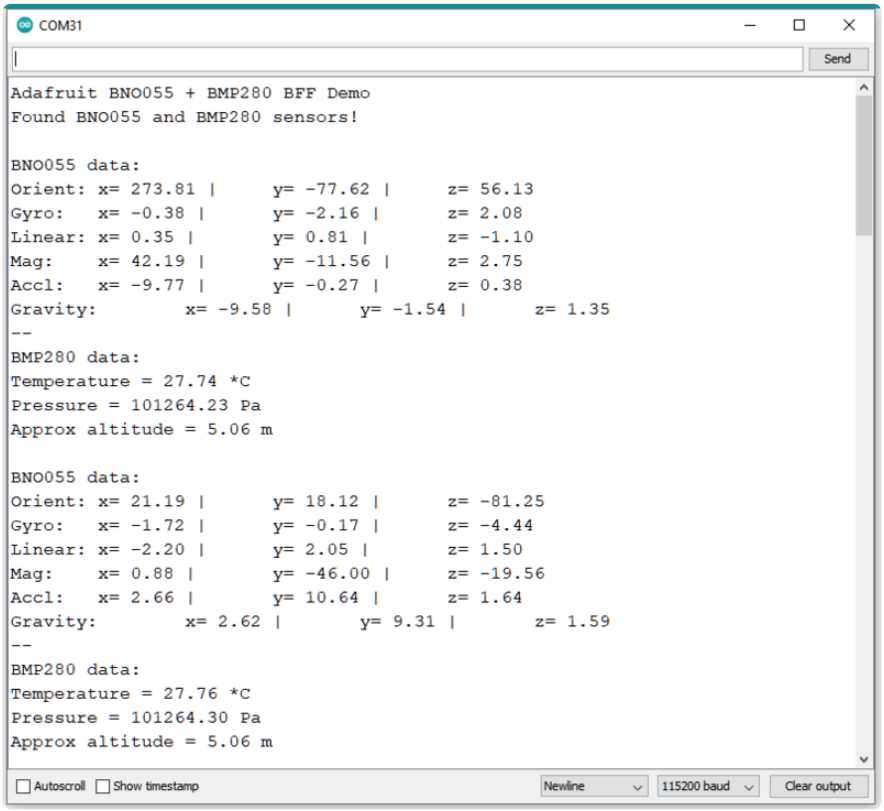
## Example Code

```
// SPDX-FileCopyrightText: 2024 Liz Clark for Adafruit Industries
//
// SPDX-License-Identifier: MIT
// BNO055 + BMP280 BFF Demo

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP280.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>

Adafruit_BNO055 bno = Adafruit_BNO055(55, 0x28, &Wire);
Adafruit_BMP280 bmp;

void setup(void)
{
  Serial.begin(115200);

  while (!Serial) delay(10);  // wait for serial port to open!

  Serial.println("Adafruit BNO055 + BMP280 BFF Demo");

  /* Initialise the sensor */
  if (!bno.begin())
  {
    /* There was a problem detecting the BNO055 ... check your connections */
    Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
    while (1);
  }
  if (!bmp.begin()) {
    Serial.print("Ooops, no BMP280 detected ... Check your wiring or I2C ADDR!");
    while (1);
  }
  bmp.setSampling(Adafruit_BMP280::MODE_NORMAL,     /* Operating Mode. */
                  Adafruit_BMP280::SAMPLING_X2,     /* Temp. oversampling */
                  Adafruit_BMP280::SAMPLING_X16,    /* Pressure oversampling */
```

```
                  Adafruit_BMP280::FILTER_X16,      /* Filtering. */
                  Adafruit_BMP280::STANDBY_MS_500); /* Standby time. */
  Serial.println("Found BNO055 and BMP280 sensors!");
  Serial.println();
  delay(1000);
}

void loop(void)
{
  //could add VECTOR_ACCELEROMETER, VECTOR_MAGNETOMETER,VECTOR_GRAVITY...
  sensors_event_t orientationData , angVelocityData , linearAccelData,
magnetometerData, accelerometerData, gravityData;
  bno.getEvent(&orientationData, Adafruit_BNO055::VECTOR_EULER);
  bno.getEvent(&angVelocityData, Adafruit_BNO055::VECTOR_GYROSCOPE);
  bno.getEvent(&linearAccelData, Adafruit_BNO055::VECTOR_LINEARACCEL);
  bno.getEvent(&magnetometerData, Adafruit_BNO055::VECTOR_MAGNETOMETER);
  bno.getEvent(&accelerometerData, Adafruit_BNO055::VECTOR_ACCELEROMETER);
  bno.getEvent(&gravityData, Adafruit_BNO055::VECTOR_GRAVITY);
  Serial.println("BNO055 data:");
  printEvent(&orientationData);
  printEvent(&angVelocityData);
  printEvent(&linearAccelData);
  printEvent(&magnetometerData);
  printEvent(&accelerometerData);
  printEvent(&gravityData);
  Serial.println("--");
  Serial.println("BMP280 data:");
  Serial.print(F("Temperature = "));
  Serial.print(bmp.readTemperature());
  Serial.println(" *C");

  Serial.print(F("Pressure = "));
  Serial.print(bmp.readPressure());
  Serial.println(" Pa");

  Serial.print(F("Approx altitude = "));
  Serial.print(bmp.readAltitude(1013.25)); /* Adjusted to local forecast! */
  Serial.println(" m");

  Serial.println();
  delay(2000);
}

void printEvent(sensors_event_t* event) {
  double x = -1000000, y = -1000000 , z = -1000000; //dumb values, easy to spot
problem
  if (event->type == SENSOR_TYPE_ACCELEROMETER) {
    Serial.print("Accl:");
    x = event->acceleration.x;
    y = event->acceleration.y;
    z = event->acceleration.z;
  }
  else if (event->type == SENSOR_TYPE_ORIENTATION) {
    Serial.print("Orient:");
    x = event->orientation.x;
    y = event->orientation.y;
    z = event->orientation.z;
  }
  else if (event->type == SENSOR_TYPE_MAGNETIC_FIELD) {
    Serial.print("Mag:");
    x = event->magnetic.x;
    y = event->magnetic.y;
    z = event->magnetic.z;
  }
  else if (event->type == SENSOR_TYPE_GYROSCOPE) {
    Serial.print("Gyro:");
    x = event->gyro.x;
    y = event->gyro.y;
    z = event->gyro.z;
```

```
  }
  else if (event->type == SENSOR_TYPE_ROTATION_VECTOR) {
    Serial.print("Rot:");
    x = event->gyro.x;
    y = event->gyro.y;
    z = event->gyro.z;
  }
  else if (event->type == SENSOR_TYPE_LINEAR_ACCELERATION) {
    Serial.print("Linear:");
    x = event->acceleration.x;
    y = event->acceleration.y;
    z = event->acceleration.z;
  }
  else if (event->type == SENSOR_TYPE_GRAVITY) {
    Serial.print("Gravity:");
    x = event->acceleration.x;
    y = event->acceleration.y;
    z = event->acceleration.z;
  }
  else {
    Serial.print("Unk:");
  }

  Serial.print("\tx= ");
  Serial.print(x);
  Serial.print(" |\ty= ");
  Serial.print(y);
  Serial.print(" |\tz= ");
  Serial.println(z);
}
```

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at 115200 baud. You'll see the BNO055 and BMP280 sensors recognized over I2C. Then, you'll see sensor data from the BNO055 and the BMP280 print out to the Serial Monitor.



# BMP280 Arduino Docs

BMP280 Arduino Docs (https://adafru.it/19Tc)

# BNO055 Arduino Docs

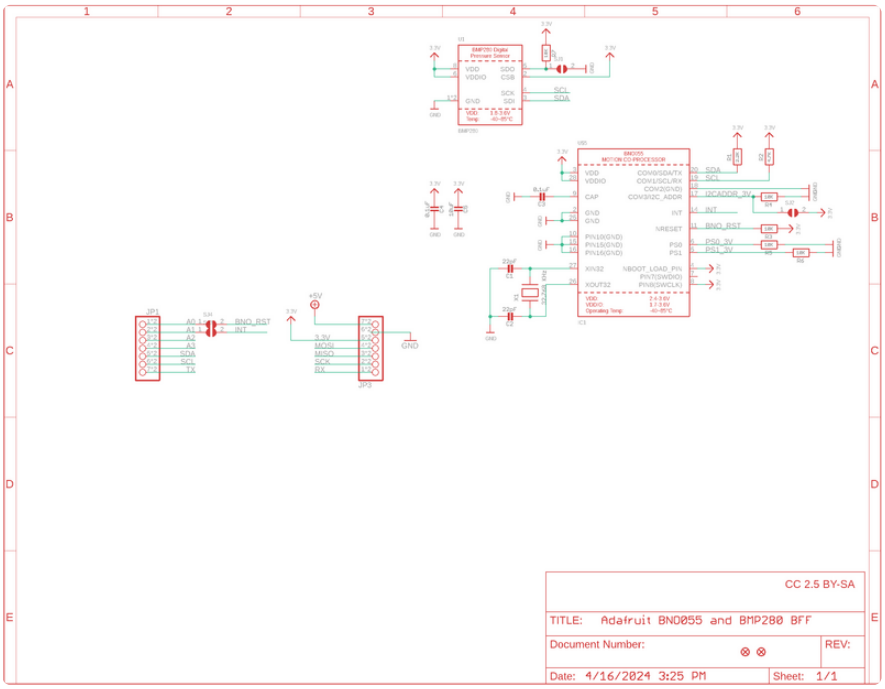
BNO055 Arduino Docs (https://adafru.it/19Td)

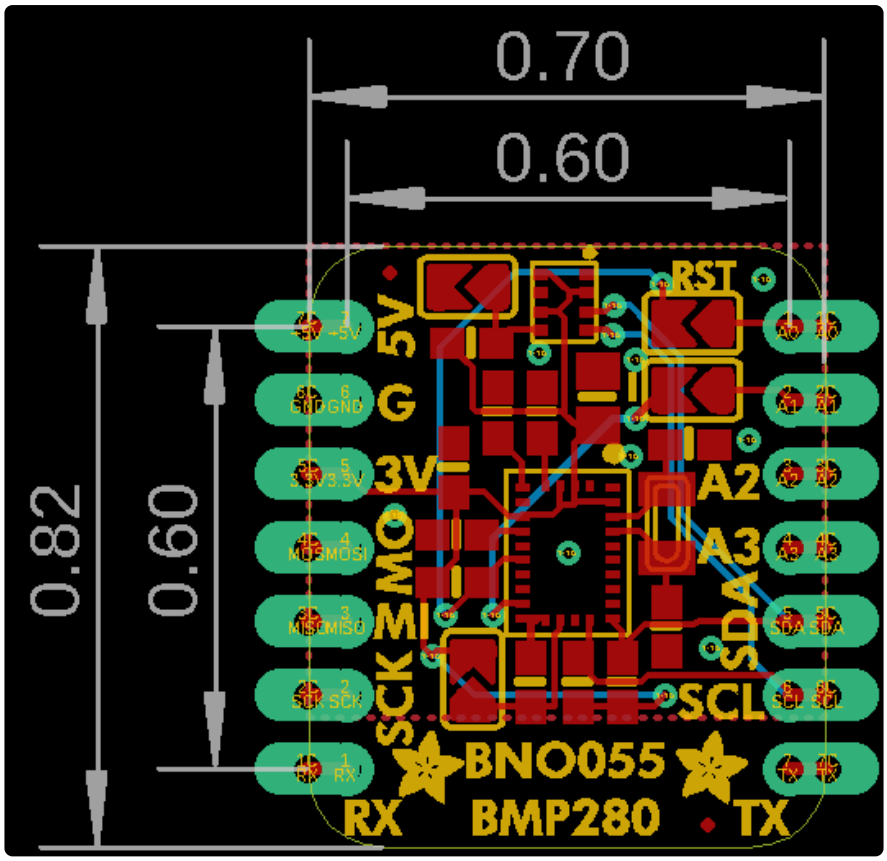
# Downloads

## Files

- BNO055 Datasheet (https://adafru.it/19Ub)
- BMP280 Datasheet (https://adafru.it/19Uc)
- EagleCAD PCB files on GitHub (https://adafru.it/19Ud)
- Fritzing object in the Adafruit Fritzing Library (https://adafru.it/19Ue)

## Schematic and Fab Print




0.70
0.60
0.82
0.60
RST
5V
G
3V
A2
A3
MO
MI
SCK
SDA
SCL
BNO055
BMP280
RX
TX