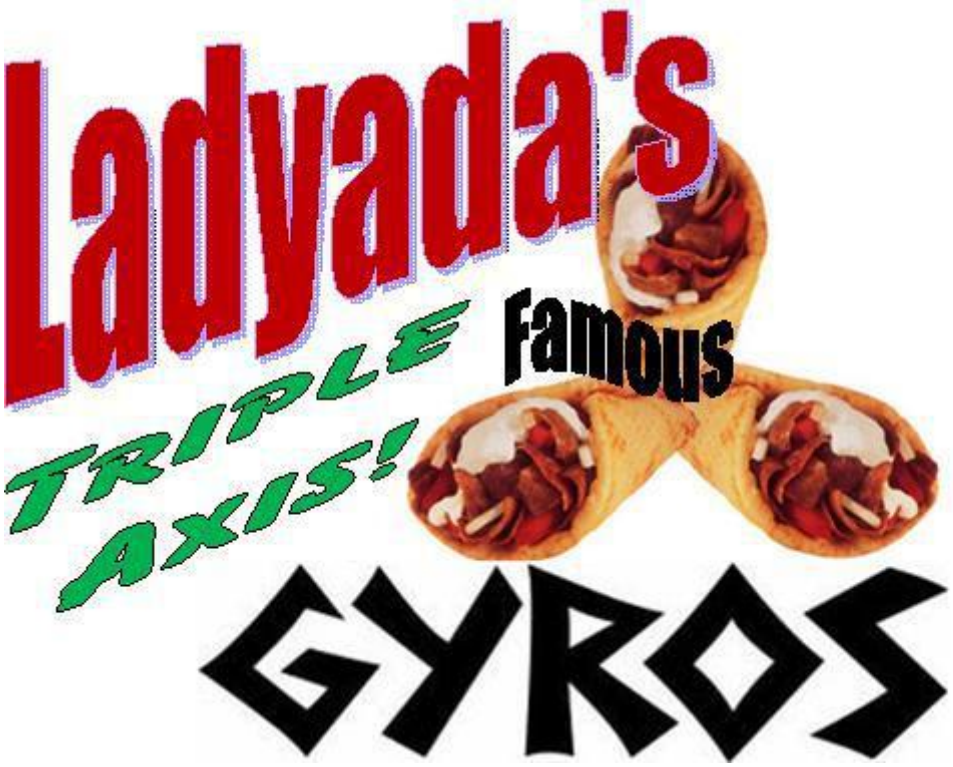
Adafruit Triple Axis Gyro Breakout

Created by Bill Earl



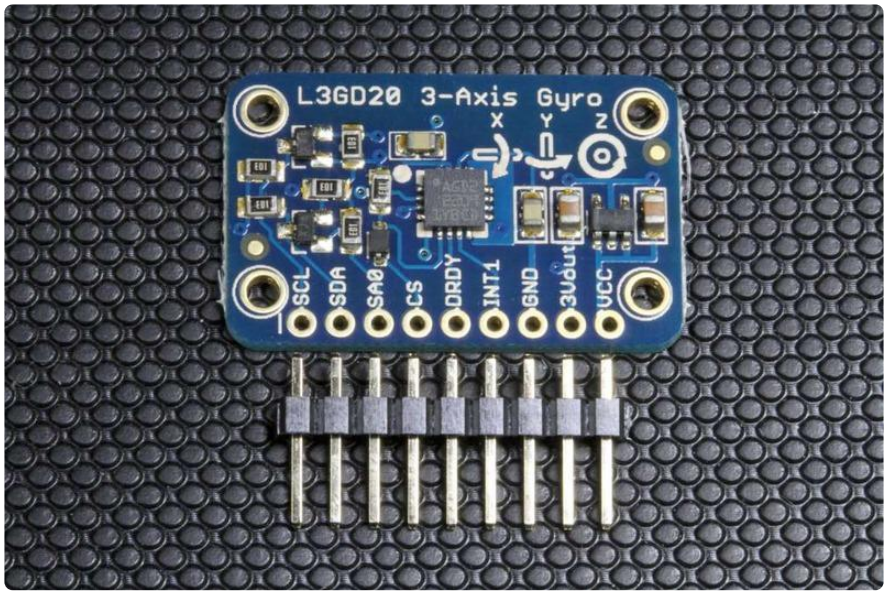
https://learn.adafruit.com/adafruit-triple-axis-gyro-breakout

Last updated on 2022-12-01 01:54:43 PM EST

Table of Contents

Overview

The Adafruit Triple Axis Gyro Breakout is based on the STMicro L3GD20 MEMS digital output gyroscope chip (). We include a 3.3v regulator on board for compatibility with 5v controllers like the Arduino. And there are 4 holes so that it can be rigidly mounted.



How it Works:

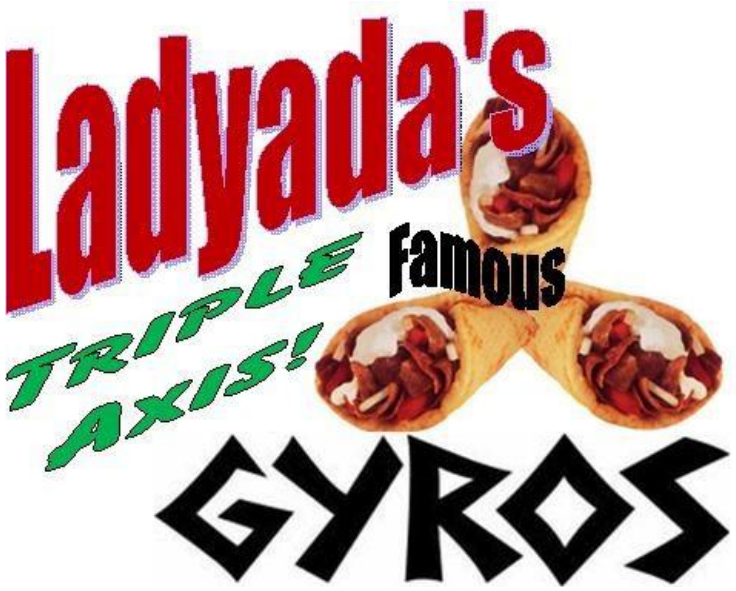
The triple-axis gyro sensor is a MEMS (Micro Electrical Mechanical System) device consisting of 3 micro-machined 'tuning fork' structures on a silicon wafer. These structures are designed to vibrate when stimulated by an electrical signal. When rotated about the axis of the tuning fork, the tines will deflect due to the Coriolis force (). This deflection is proportional to the speed of rotation.

The 3 MEMS structures are arranged orthogonally, on the X, Y and Z axis. Deflection on each tuning fork is detected as a change in capacitance between sensing plates built into the MEMS structure and converted to a degrees-per-second rotation rate for each of the three axis.

For a more detailed description of MEMS Gyros, see MEMS Gyroscopes and their applications (). and Everything about STMicroelectronics' 3-axis digital MEMS ()gyroscopes ().

What can it do?

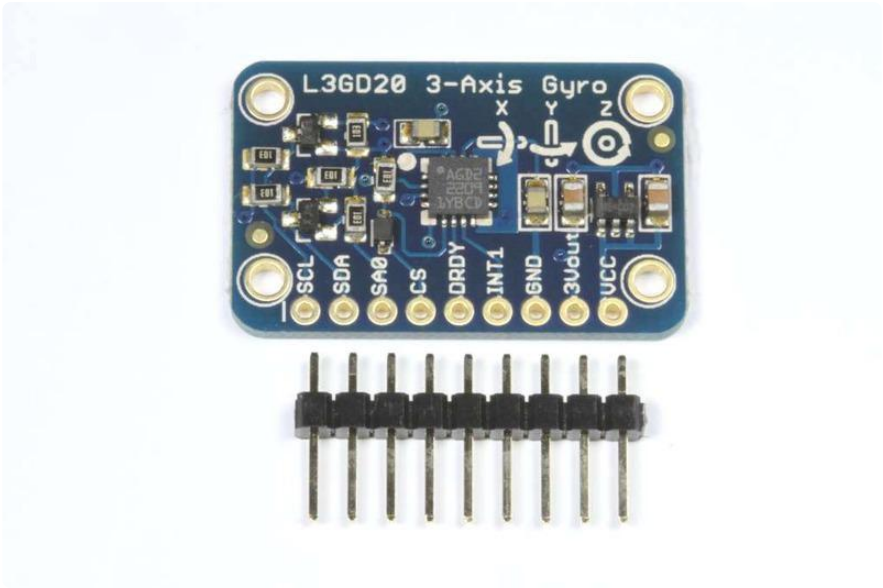
Gyroscopes are useful for many types of motion sensing applications. They are often paired with accelerometers () for inertial guidance systems, 3D motion capture and inverted pendulum (e.g. Segway) type applications. The L3GD20 is particularly versatile, with three full axes of sensing, selectable ±250, ±500, and ±2000 degree-per-second sensitivity ranges and built-in high/low pass filtering.



Ladyada's Triple Gyro Special! (http://adafru.it/1032)

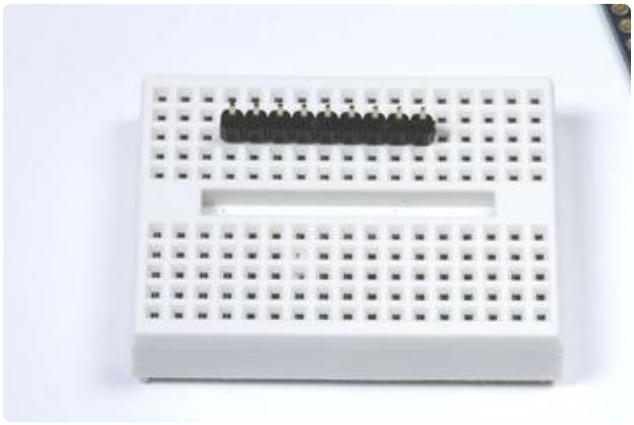
3 micro-machined gyros with I2C, SPI and a 3v regulator on a bite-sized breakout board.
(Tomato, onion and tzatziki sauce extra.)

Assembly and Wiring



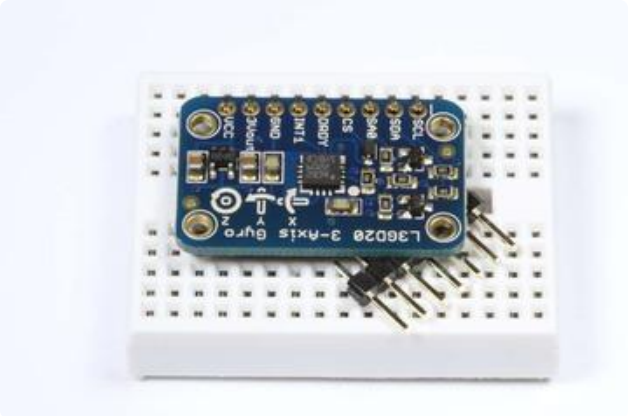
Assembly:

The board comes with all surface-mount components pre-soldered. The included header strip can be soldered on for convenient use on a breadboard or with 0.1" connectors. However, for applications subject to extreme accelerations, shock or vibration, locking connectors or direct soldering is advised.



Prepare the header strip:

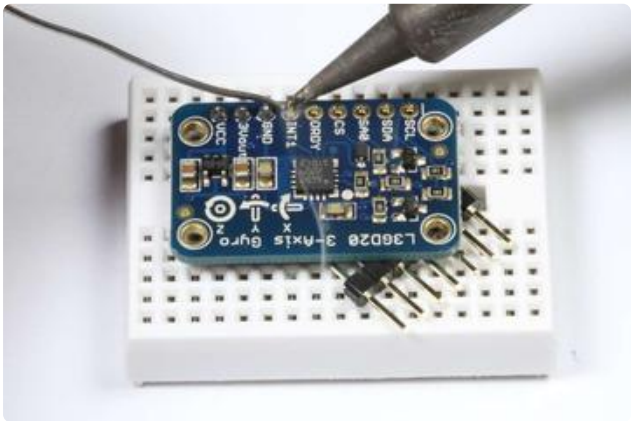
Cut the strip[to length if necessary. It will be easier to solder if you insert it into a breadboard - long pins down.



Add the breakout board:

Place the breakout board over the pins.

(You can prop up the free edge of the board with some extra pins to better align it for soldering.)



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 ()).

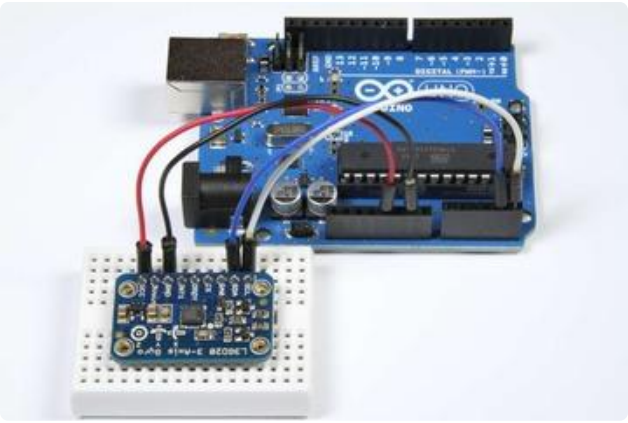
Wiring:

You'll need to power the breakout, you can power it from 3V-5VDC, connect ground to GND and VCC to our power supply (3-5V)

The L3GD20 breakout board supports both I2C and SPI communication. I2C requires the fewest connections, so we will start with that:

Wiring for I2C:

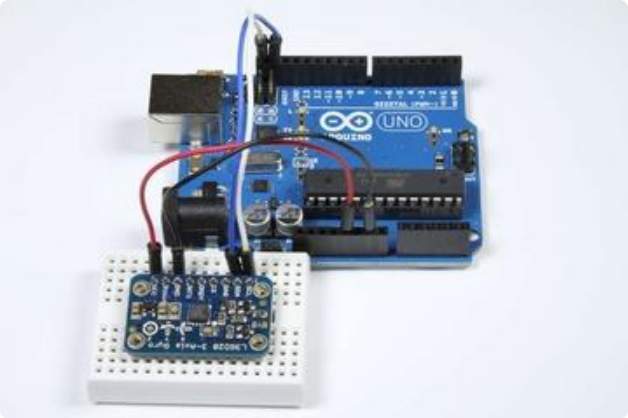
I2C requires only 2 pins (in addition to VCC and ground).



'Classic' Arduino Wiring:

On pre-R3 Arduinos, the I2C pins are:
SDA = Analog 4
SCL = Analog 5

For the Mega
SDA = Digital 20
SCL = Digital 21

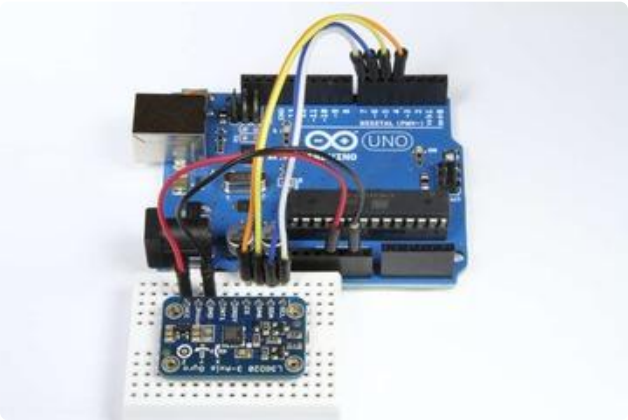


R3 and Later Arduino Wiring:

Although the 'classic' wiring will still work, All R3 and later Arduinos (including Mega, Due and Leonardo) have SDA and SCL pins on the extended header next to AREF for compatibility.

Wiring for SPI:

The SPI interface requires 4 wires (in addition to VCC and Ground).



SPI Wiring:

The library uses "software SPI" so the choice of pins is a little more flexible. What we show here is compatible with the example code included with the library:

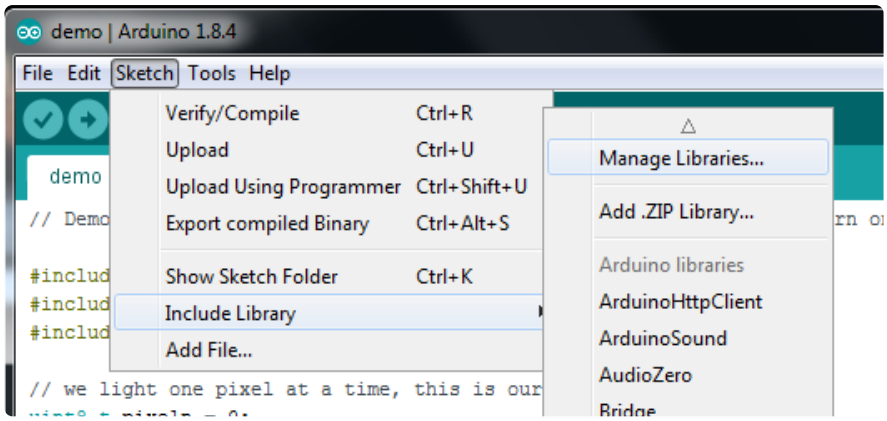
CS = Digital 4
SAO = Digital 5
SDA = Digital 6
SCL = Digital 7

Arduino

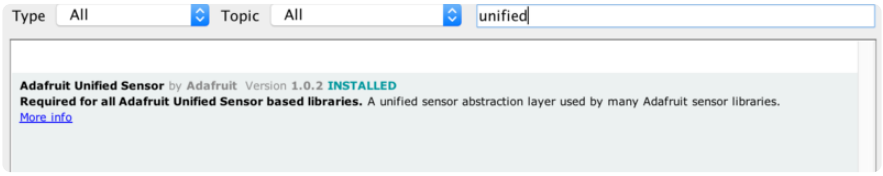
The Adafruit L3GD20 Library for the Arduino implements a convenient device class to handle the the low-level device communication with the Gyro module. The programming interface is described below:

Install Arduino Libraries

Before you can use the L3GD20, you'l need to install the required libraries using the Arduino Library Manager, which you can open via the menu entry shown below:



You will need to install the Adafruit Unified Sensor library ...



... as well as Adafruit L3GD20 U:

Construction:

To use the L3GD20 in your sketch, you must first call a constructor to create a device object. There are two forms of the constructor:

- Adafruit_L3GD20(void);
- Adafruit_L3GD20(int8_t cs, int8_t mosi, int8_t miso, int8_t clk);

The first version takes no parameters and is used for I2C communication. The second version is for SPI communication and requires that you specify the pins to be used.

I2C Example: (use with I2C wiring ())

```
// No need to specify pins for I2C
Adafruit_L3GD20 gyro();
```

SPI Example: (use with SPI wiring ())

```
// Define the pins for SPI
#define GYRO_CS 4 // labeled CS
#define GYRO_DO 5 // labeled SA0
```

```
#define GYRO_DI 6  // labeled SDA
#define GYRO_CLK 7 // labeled SCL

Adafruit_L3GD20 gyro(GYRO_CS, GYRO_DO, GYRO_DI, GYRO_CLK);
```

Initialization:

Before using the device object you constructed, you must initialize it with the sensitivity range you want to use:

- bool begin(gyroRange_t rng);

where "rng" can be one of:

- L3DS20_RANGE_250DPS - for 250 degrees-per-second range (default)
- L3DS20_RANGE_500DPS - for 500 degrees-per-second range
- L3DS20_RANGE_2000DPS - for 2000 degrees-per-second range

Example:

```
void setup()
{
  Serial.begin(9600);

  // Try to initialise and warn if we couldn't detect the chip
  if (!gyro.begin(gyro.L3DS20_RANGE_250DPS))
  {
    Serial.println("Oops ... unable to initialize the L3GD20. Check your wiring!");
    while (1);
  }
}
```

Sensing Rotation:

To sense rotation, you must first call the "read()" function to take a reading:

- void read(void);

This function takes no parameters. After calling "read()". The raw x, y and z readings can be retrieved from the device object's "data" member.

- data.x - x-axis rotation rate in degrees-per-second
- data.y - y-axis rotation rate in degrees-per-second
- data.z - z-axis rotation rate in degrees-per-second

Example:

```
void loop()
{
  gyro.read();
  Serial.print("X: "); Serial.print((int)gyro.data.x);   Serial.print(" ");
  Serial.print("Y: "); Serial.print((int)gyro.data.y);   Serial.print(" ");
  Serial.print("Z: "); Serial.println((int)gyro.data.z); Serial.print(" ");
  delay(100);
}
```

Alternate Units:

The values reported by the read() function are in degrees-per-second (dps) For some calculations, it may be more convenient to work in radians. To convert dps to radians-per-second (rad/s), simply multiply by 0.017453293 as in the following code:

```
#define SENSORS_DPS_TO_RADS                 (0.017453293F)          /**&lt; Degrees/s
to rad/s multiplier */

void loop()
{
  gyro.read();
  Serial.print("X: "); Serial.print((int)gyro.data.x * SENSORS_DPS_TO_RADS);
Serial.print(" ");
  Serial.print("Y: "); Serial.print((int)gyro.data.y * SENSORS_DPS_TO_RADS);
Serial.print(" ");
  Serial.print("Z: "); Serial.println((int)gyro.data.z * SENSORS_DPS_TO_RADS);
Serial.print(" ");
  delay(100);
}
```

Calibration:

The L3GD20 is calibrated at the factory to close tolerances and will provide sufficient accuracy for most applications.

For critical applications where maximum accuracy is required, the gyro should be calibrated for zero-rate and sensitivity. For detailed information on how to calibrate a MEMS gyro, please refer to section 5.3 of this technical article ().

The L3GD20 includes an on-chip temperature sensor, but this sensor isn't intended to be used to measure ambient temperature. The register seems to be used to provide a temperature offset for the die temperature to account for variation across multiple gyros or over time, but the value this offset is relative to unfortunately isn't specified in the datasheet, so there is no obvious way to use it in a useful manner as-is.

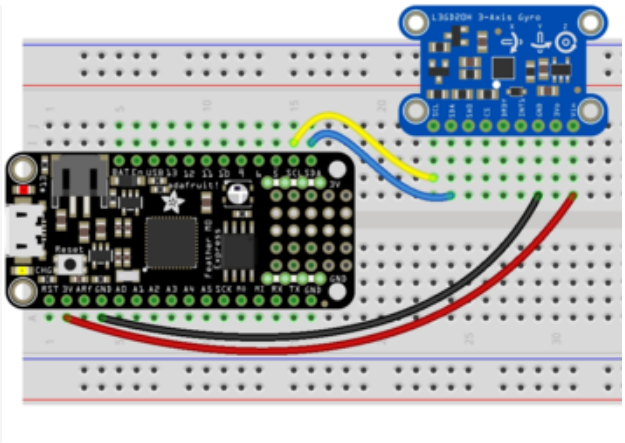
Python & CircuitPython

It's easy to use the L3GD20 sensor with Python or CircuitPython and the Adafruit CircuitPython L3GD20 () module. This module allows you to easily write Python code that reads the angular momentum from the sensor.

You can use this sensor with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library ().

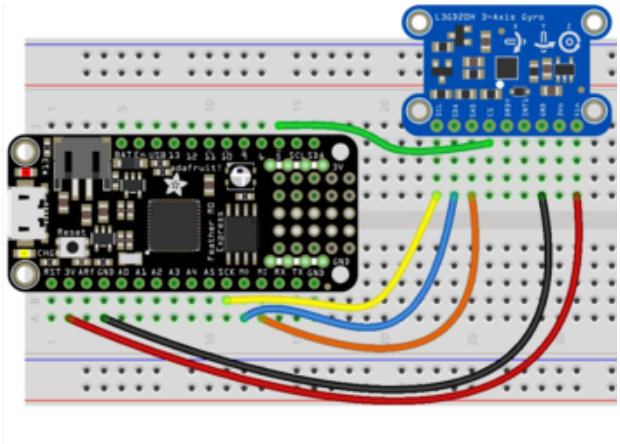
CircuitPython Microcontroller Wiring

First wire up a L3GD20 to your board exactly as shown on the previous pages for Arduino. You can use either I2C or SPI wiring, although it's recommended to use I2C for simplicity. Here's an example of wiring a Feather M0 to the sensor with I2C:



Board 3V to sensor Vin
Board GND to sensor GND
Board SCL to sensor SCL
Board SDA to sensor SDA

And an example of a Feather M0 wired with hardware SPI:

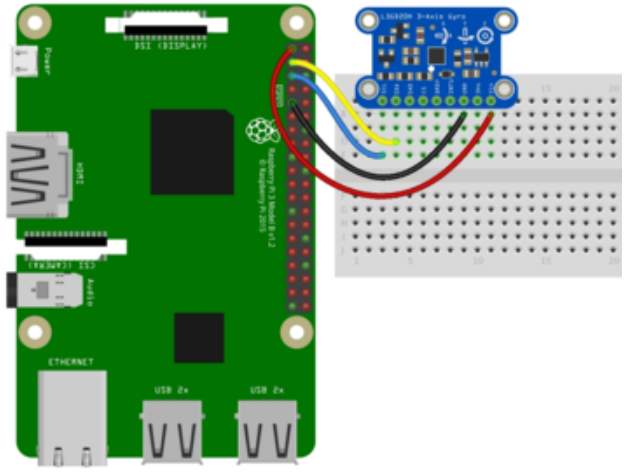


Board 3V to sensor Vin
Board GND to sensor GND
Board SCK to sensor SCL
Board MOSI to sensor SDA
Board MISO to sensor SA0
Board D5 to sensor CS (or use any other free digital I/O pin)

Python Computer Wiring

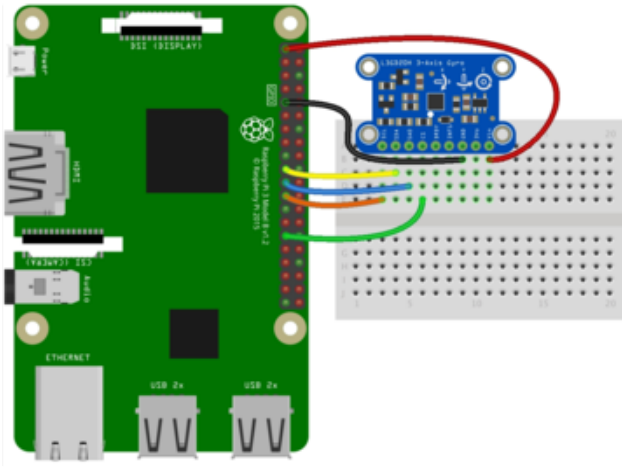
Since there's dozens of Linux computers/boards you can use we will show wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported ().

Here's the Raspberry Pi wired with I2C:



Pi 3V3 to sensor Vin
Pi GND to sensor GND
Pi SCL to sensor SCL
Pi SDA to sensor SDA

And an example on the Raspberry Pi 3 Model B wired with SPI:



Pi 3V3 to sensor Vin
Pi GND to sensor GND
Pi MOSI to sensor SDA
Pi MISO to sensor SA0
Pi SCLK to sensor SCL
Pi #5 to sensor CS (or use any other free GPIO pin)

CircuitPython Installation of L3GD20 Library

You'll need to install the Adafruit CircuitPython L3GD20 () 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 CircuitPython starter guide has a great page on how to install the library bundle ().

For non-express boards like the Trinket M0 or Gemma M0, you'll need to manually install the necessary libraries from the bundle:

- adafruit_l3gd20.mpy
- adafruit_bus_device

Before continuing make sure your board's lib folder or root filesystem has the adafruit_l3gd20.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.

Python Installation of L3GD20 Library

You'll need to install the Adafruit_Blinka library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready ()!

Once that's done, from your command line run the following command:

- `sudo pip3 install adafruit-circuitpython-l3gd20`

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!

CircuitPython & Python Usage

To demonstrate the usage of the sensor we'll initialize it and read the angular momentum values from the board's Python REPL.

If you're using an I2C connection run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```
import time
import board
import busio
import adafruit_l3gd20
I2C = busio.I2C(board.SCL, board.SDA)
SENSOR = adafruit_l3gd20.L3GD20_I2C(I2C)
```

Or if you're using a SPI connection run this code instead to setup the SPI connection and sensor:

```
import time
import board
import busio
import digitalio
import adafruit_l3gd20
CS = digitalio.DigitalInOut(board.D5)
SPIB = busio.SPI(board.SCK, board.MOSI, board.MISO)
SENSOR = adafruit_l3gd20.L3GD20_SPI(SPIB, CS)
```

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

- gyro - The x, y, z angular momentum tuple floats, rescaled appropriately for range selected.

For example to print angular momentum:

```
print('Angular momentum (rad/s): {}'.format(SENSOR.gyro))
```

```
>>> print('Angular Momentum (rad/s): {}'.format(SENSOR.gyro))
Angular Momentum (rad/s): (-2.09125, -3.78, 0.2975)
```

That's all there is to using the L3GD20 sensor with CircuitPython!

Full Example Code

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
import adafruit_l3gd20

# Hardware I2C setup:
I2C = board.I2C()  # uses board.SCL and board.SDA
# I2C = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a
microcontroller
# Initializes L3GD20 object using default range, 250dps
SENSOR = adafruit_l3gd20.L3GD20_I2C(I2C)
# Initialize L3GD20 object using a custom range and output data rate (ODR).
# SENSOR = adafruit_l3gd20.L3GD20_I2C(
#     I2C, rng=adafruit_l3gd20.L3DS20_RANGE_500DPS,
```

```
rate=adafruit_l3gd20.L3DS20_RATE_200HZ
# )

# Possible values for rng are:
# adafruit_l3gd20.L3DS20_Range_250DPS, 250 degrees per second. Default range
# adafruit_l3gd20.L3DS20_Range_500DPS, 500 degrees per second
# adafruit_l3gd20.L3DS20_Range_2000DPS, 2000 degrees per second

# Possible values for rate are:
# adafruit_l3gd20.L3DS20_RATE_100HZ, 100Hz data rate. Default data rate
# adafruit_l3gd20.L3DS20_RATE_200HZ, 200Hz data rate
# adafruit_l3gd20.L3DS20_RATE_400HZ, 400Hz data rate
# adafruit_l3gd20.L3DS20_RATE_800HZ, 800Hz data rate

# Hardware SPI setup:
# import digitalio
# CS = digitalio.DigitalInOut(board.D5)
# SPIB = board.SPI()
# SENSOR = adafruit_l3gd20.L3GD20_SPI(SPIB, CS)
# SENSOR = adafruit_l3gd20.L3GD20_I2C(
#    SPIB,
#    CS,
#    rng=adafruit_l3gd20.L3DS20_RANGE_500DPS,
#    rate=adafruit_l3gd20.L3DS20_RATE_200HZ,
# )

while True:
    print("Angular Velocity (rad/s): {}".format(SENSOR.gyro))
    print()
    time.sleep(1)
```

Python Docs

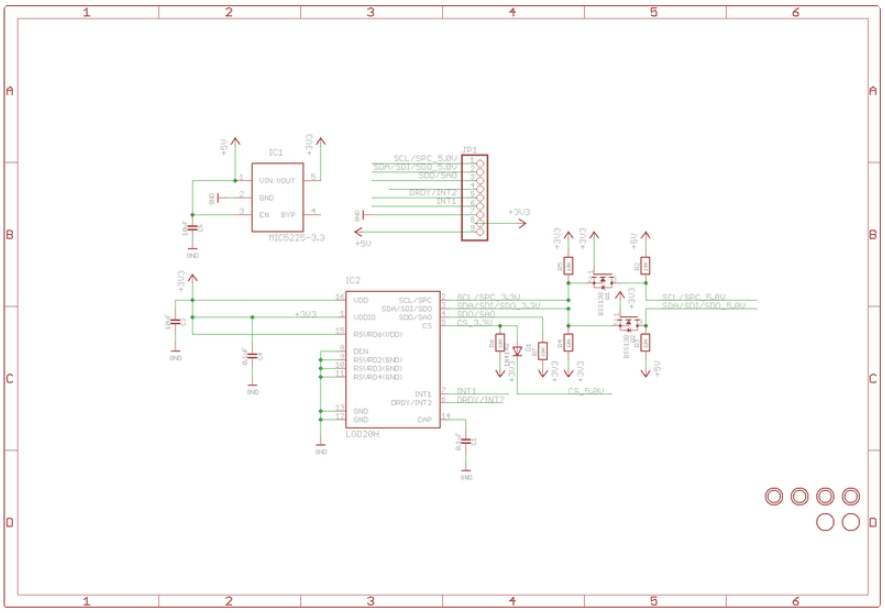
Python Docs ()

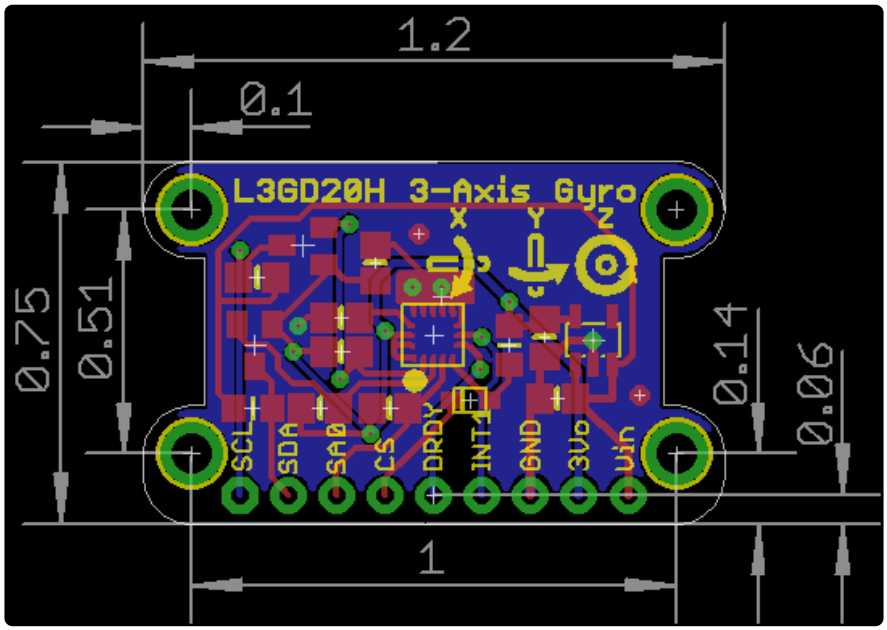
Downloads

Files

- L3GD20 Data Sheet ()
- Technical Article: Everything about STMicroelectronics' 3-axis digital MEMS gyroscopes ()
- Adafruit L3GD20 Library driver for Arduino ()
- Adafruit L3GD20 Library driver for CircuitPython ()
- EagleCAD PCB files on GitHub ()
- Fritzing object in Fritzing library ()

Schematic & Fabrication Print





F.A.Q.