# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>3</td>
</tr>
<tr>
<td>Pinouts</td>
<td>5</td>
</tr>
<tr>
<td>• Power Pins</td>
<td></td>
</tr>
<tr>
<td>• I2C Logic Pins</td>
<td></td>
</tr>
<tr>
<td>• SPI Logic pins:</td>
<td></td>
</tr>
<tr>
<td>• Other Pins</td>
<td></td>
</tr>
<tr>
<td>Arduino</td>
<td>6</td>
</tr>
<tr>
<td>• I2C Wiring</td>
<td></td>
</tr>
<tr>
<td>• SPI Wiring</td>
<td></td>
</tr>
<tr>
<td>• Library Installation</td>
<td></td>
</tr>
<tr>
<td>• Load Example</td>
<td></td>
</tr>
<tr>
<td>• Example Code</td>
<td></td>
</tr>
<tr>
<td>Arduino Docs</td>
<td>12</td>
</tr>
<tr>
<td>Python &amp; CircuitPython</td>
<td>13</td>
</tr>
<tr>
<td>• CircuitPython Microcontroller Wiring</td>
<td></td>
</tr>
<tr>
<td>• Python Computer Wiring</td>
<td></td>
</tr>
<tr>
<td>• CircuitPython Installation of LSM6DS Library</td>
<td></td>
</tr>
<tr>
<td>• Python Installation of LSM6DS Library</td>
<td></td>
</tr>
<tr>
<td>• CircuitPython &amp; Python Usage</td>
<td></td>
</tr>
<tr>
<td>• Example Code</td>
<td></td>
</tr>
<tr>
<td>Python Docs</td>
<td>16</td>
</tr>
<tr>
<td>Downloads</td>
<td>17</td>
</tr>
<tr>
<td>• Files</td>
<td></td>
</tr>
<tr>
<td>• Schematic</td>
<td></td>
</tr>
<tr>
<td>• Fab Print</td>
<td></td>
</tr>
</tbody>
</table>
Overview

Add motion and orientation sensing to your Arduino or CircuitPython project with this affordable 6 Degree of Freedom (6-DoF) sensor combo from ST. The board includes an LSM6DS33, a 6-DoF IMU accelerometer + gyroscope. The 3-axis accelerometer, can tell you which direction is down towards the Earth (by measuring gravity) or how fast the board is accelerating in 3D space. The 3-axis gyroscope that can measure spin and twist. This chip isn't the newest motion sensor, but it is well-established and comes at a great price.
To make getting started fast and easy, we placed the sensor on a compact breakout board with voltage regulation and level-shifted inputs. That way you can use it with 3V or 5V power/logic devices without worry. Both I2C and SPI interfaces are available, so you'll be able to use it with any hardware setup. The breakout comes fully assembled and tested, with some extra header so you can use it on a breadboard. Four mounting holes make for a secure connection.

This breakout does not include a magnetometer, often required for accurate orientation. We recommend the LIS3MDL 3-axis magnetometer to match up with this IMU. We even have a version that contains both!

Additionally, since it speaks I2C, you can easily connect it up with two wires (plus power and ground!). We’ve even included two SparkFun qwiic compatible STEMMA QT connectors for the I2C bus so you don't even need to solder! Just wire up to your favorite micro like the STM32F405 Feather with a plug-and-play cable to get 6 DoF data ASAP. You can change the I2C address on the back using the solder jumper, to allow you to have two of these sensor boards on one bus.

We also wrote libraries to help you get these sensors integrated with your project. This library contains an Arduino driver for the accel/gyro. For advanced Arduino usage, ST has their own fully-featured library that includes extras such as FIFO management and tap detection for the LSM6DS3.

If you prefer Python, or just want to get going extra quick with minimal hassle, you can use this CircuitPython library and its included examples.
Power Pins

- VIN - this is the power pin. Since the sensor chip uses 3 VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it down. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V microcontroller like Arduino, use 5V
- 3V - this is the 3.3V output from the voltage regulator, you can grab up to 100mA from this if you like
- GND - common ground for power and logic
I2C Logic Pins

- SCL - I2C clock pin, connect to your microcontroller I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- SDA - I2C data pin, connect to your microcontroller I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.

- STEMMA QT () - These connectors allow you to connectors to dev boards with STEMMA QT connectors or to other things with various associated accessories ()
- DO/AD0 Jumper - I2C Address pin. Pulling this pin high or bridging the solder jumper on the back will change the I2C address from 0x6A to 0x6B

SPI Logic pins:

- SCL - This is also the SPI Clock pin / SCK, it's an input to the chip
- SDA - this is also the Serial Data In / Microcontroller Out Sensor In / MOSI pin, for data sent from your processor to the LSM6DS33
- DO - this is the Data Out / Microcontroller In Sensor Out / MISO pin, for data sent from the LSM6DS33 to your processor.
- CS - this is the Chip Select pin, drop it low to start an SPI transaction. It's an input to the chip

If you want to connect multiple LSM6DS33's to one microcontroller, have them share the SCL, SDA, and DO pins. Then assign each one a unique CS pin.

Other Pins

- INT1 -This is the primary interrupt pin. You can setup the LSM6DS33 to pull this low when certain conditions are met such as new measurement data being available. Consult the datasheet () for usage
- I2/INT2 -This is the secondary interrupt pin. You can setup the LSM6DS33 to pull this low when certain conditions are met such as new measurement data being available. Consult the datasheet () for usage

Arduino

I2C Wiring

Use this wiring if you want to connect via I2C interface
By default, the I2C address is 0x6A. If you add a jumper from DO to 3.3V the address will change to 0x6B.

Connect board VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.
Connect board GND (black wire) to Arduino GND
Connect board SCL (yellow wire) to Arduino SCL
Connect board SDA (blue wire) to Arduino SDA

The final results should resemble the illustration above, showing an Adafruit Metro development board.

**SPI Wiring**

Since this is a SPI-capable sensor, we can use hardware or 'software' SPI. To make wiring identical on all microcontrollers, we'll begin with 'software' SPI. The following pins should be used:
Connect Vin to the power supply, 3V or 5V is fine. Use the same voltage that the microcontroller logic is based off of.

Connect GND to common power/data ground

Connect the SCK pin to Digital #13 but any pin can be used later

Connect the DO pin to Digital #12 but any pin can be used later

Connect the SDA pin to Digital #11 but any pin can be used later

Connect the CS pin Digital #10 but any pin can be used later

Later on, once we get it working, we can adjust the library to use hardware SPI if you desire, or change the pins to others.

**Library Installation**

You can install the Adafruit LSM6DS Library for Arduino using the Library Manager in the Arduino IDE.

Click the Manage Libraries ... menu item, search for Adafruit LSM6DS, and select the Adafruit LSM6DS library:

Then follow the same process for the Adafruit BusIO library.
Finally follow the same process for the Adafruit Unified Sensor library:

Load Example

Open up File -> Examples -> Adafruit LSM6DS -> adafruit_lsm6ds33_test and upload to your Arduino wired up to the sensor.

Depending on whether you are using I2C or SPI, change the pin names and comment or uncomment the following lines.

```cpp
if (!lsm6ds33.begin_I2C()) {
    // if (!lsm6ds33.begin_SPI(LSM_CS)) {
    // if (!lsm6ds33.begin_SPI(LSM_CS, LSM_SCK, LSM_MISO, LSM_MOSI)) {
```

Once you upload the code and open the Serial Monitor (Tools->Serial Monitor) at 115200 baud, you will see the current configuration printed, followed by the accelerometer, gyro, and temperature measurements. You should see something similar to this:

```
Adafruit LSM6DS33 test!
LSM6DS33 Found!
Accelerometer range set to: ±2G
Gyro range set to: 250 degrees/s
Accelerometer data rate set to: 104 Hz
Gyro data rate set to: 104 Hz
Temperature 24.50 deg C
  Accel X: -4.44  Y: 1.53  Z: 9.42 m/s²
  Gyro X: 0.13  Y: -0.21  Z: -0.80 radians/s
Temperature 24.52 deg C
  Accel X: -3.60  Y: 5.21  Z: 6.90 m/s²
  Gyro X: 3.73  Y: -1.21  Z: 1.56 radians/s
Temperature 24.52 deg C
  Accel X: -0.63  Y: 6.54  Z: 4.43 m/s²
  Gyro X: -0.75  Y: -2.57  Z: 1.64 radians/s
```

Give the sensor a wiggle or a spin and watch how the measurements change!

Example Code

```cpp
// Basic demo for accelerometer/gyro readings from Adafruit LSM6DS33
```
```c
#include <Adafruit_LSM6DS33.h>

// For SPI mode, we need a CS pin
#define LSM_CS 10
// For software-SPI mode we need SCK/MOSI/MISO pins
#define LSM_SCK 13
#define LSM_MISO 12
#define LSM_MOSI 11

Adafruit_LSM6DS33 lsm6ds33;

void setup(void) {
  Serial.begin(115200);
  while (!Serial) delay(10); // will pause Zero, Leonardo, etc until serial console opens

  Serial.println("Adafruit LSM6DS33 test!");

  if (!lsm6ds33.begin_I2C()) {
    // if (!lsm6ds33.begin_SPI(LSM_CS)) {
    // if (!lsm6ds33.begin_SPI(LSM_CS, LSM_SCK, LSM_MISO, LSM_MOSI)) {
      Serial.println("Failed to find LSM6DS33 chip");
      while (1) {
        delay(10);
      }
    }
  }

  Serial.println("LSM6DS33 Found!");

  // lsm6ds33.setAccelRange(LSM6DS_ACCEL_RANGE_2_G);
  Serial.print("Accelerometer range set to: ");
  switch (lsm6ds33.getAccelRange()) {
    case LSM6DS_ACCEL_RANGE_2_G:
      Serial.println("+-2G");
      break;
    case LSM6DS_ACCEL_RANGE_4_G:
      Serial.println("+-4G");
      break;
    case LSM6DS_ACCEL_RANGE_8_G:
      Serial.println("+-8G");
      break;
    case LSM6DS_ACCEL_RANGE_16_G:
      Serial.println("+-16G");
      break;
  }

  // lsm6ds33.setGyroRange(LSM6DS_GYRO_RANGE_250_DPS);
  Serial.print("Gyro range set to: ");
  switch (lsm6ds33.getGyroRange()) {
    case LSM6DS_GYRO_RANGE_125_DPS:
      Serial.println("125 degrees/s");
      break;
    case LSM6DS_GYRO_RANGE_250_DPS:
      Serial.println("250 degrees/s");
      break;
    case LSM6DS_GYRO_RANGE_500_DPS:
      Serial.println("500 degrees/s");
      break;
    case LSM6DS_GYRO_RANGE_1000_DPS:
      Serial.println("1000 degrees/s");
      break;
    case LSM6DS_GYRO_RANGE_2000_DPS:
      Serial.println("2000 degrees/s");
      break;
    case ISM330DHCX_GYRO_RANGE_4000_DPS:
      Serial.println("4000 degrees/s");
      break; // unsupported range for the DS33
  }

  // lsm6ds33.setAccelDataRate(LSM6DS_RATE_12_5_HZ);
  Serial.print("Accelerometer data rate set to: ");
```

©Adafruit Industries
switch (lsm6ds33.getAccelDataRate()) {
  case LSM6DS_RATE_SHUTDOWN:
    Serial.println("0 Hz");
    break;
  case LSM6DS_RATE_12_5_HZ:
    Serial.println("12.5 Hz");
    break;
  case LSM6DS_RATE_26_HZ:
    Serial.println("26 Hz");
    break;
  case LSM6DS_RATE_52_HZ:
    Serial.println("52 Hz");
    break;
  case LSM6DS_RATE_104_HZ:
    Serial.println("104 Hz");
    break;
  case LSM6DS_RATE_208_HZ:
    Serial.println("208 Hz");
    break;
  case LSM6DS_RATE_416_HZ:
    Serial.println("416 Hz");
    break;
  case LSM6DS_RATE_833_HZ:
    Serial.println("833 Hz");
    break;
  case LSM6DS_RATE_1_66K_HZ:
    Serial.println("1.66 KHz");
    break;
  case LSM6DS_RATE_3_33K_HZ:
    Serial.println("3.33 KHz");
    break;
  case LSM6DS_RATE_6_66K_HZ:
    Serial.println("6.66 KHz");
    break;
}

// lsm6ds33.setGyroDataRate(LSM6DS_RATE_12_5_HZ);
Serial.print("Gyro data rate set to: ");
switch (lsm6ds33.getGyroDataRate()) {
  case LSM6DS_RATE_SHUTDOWN:
    Serial.println("0 Hz");
    break;
  case LSM6DS_RATE_12_5_HZ:
    Serial.println("12.5 Hz");
    break;
  case LSM6DS_RATE_26_HZ:
    Serial.println("26 Hz");
    break;
  case LSM6DS_RATE_52_HZ:
    Serial.println("52 Hz");
    break;
  case LSM6DS_RATE_104_HZ:
    Serial.println("104 Hz");
    break;
  case LSM6DS_RATE_208_HZ:
    Serial.println("208 Hz");
    break;
  case LSM6DS_RATE_416_HZ:
    Serial.println("416 Hz");
    break;
  case LSM6DS_RATE_833_HZ:
    Serial.println("833 Hz");
    break;
  case LSM6DS_RATE_1_66K_HZ:
    Serial.println("1.66 KHz");
    break;
  case LSM6DS_RATE_3_33K_HZ:
    Serial.println("3.33 KHz");
    break;
  case LSM6DS_RATE_6_66K_HZ:
    Serial.println("6.66 KHz");
    break;
}
case LSM6DS_RATE_6_66K_HZ:
    Serial.println("6.66 KHz");
    break;
}

lsm6ds33.configInt1(false, false, true); // accelerometer DRDY on INT1
lsm6ds33.configInt2(false, true, false); // gyro DRDY on INT2

void loop() {
    /* Get a new normalized sensor event */
    sensors_event_t accel;
    sensors_event_t gyro;
    sensors_event_t temp;
    lsm6ds33.getEvent(&accel, &gyro, &temp);

    Serial.print("\t\tTemperature ");
    Serial.print(temp.temperature);
    Serial.println(" deg C");

    /* Display the results (acceleration is measured in m/s^2) */
    Serial.print("\t\tAccel X: ");
    Serial.print(accel.acceleration.x);
    Serial.print(" Y: ");
    Serial.print(accel.acceleration.y);
    Serial.print(" Z: ");
    Serial.print(accel.acceleration.z);
    Serial.println(" m/s^2 ");

    /* Display the results (rotation is measured in rad/s) */
    Serial.print("\t\tGyro X: ");
    Serial.print(gyro.gyro.x);
    Serial.print(" Y: ");
    Serial.print(gyro.gyro.y);
    Serial.print(" Z: ");
    Serial.print(gyro.gyro.z);
    Serial.println(" radians/s ");
    Serial.println();
    delay(100);

    // serial plotter friendly format
    Serial.print(temp.temperature);
    Serial.print(accel.acceleration.x);
    Serial.print(accel.acceleration.y);
    Serial.print(accel.acceleration.z);
    Serial.print(gyro.gyro.x);
    Serial.print(gyro.gyro.y);
    Serial.print(gyro.gyro.z);
    Serial.println();
    delayMicroseconds(10000);
}
Python & CircuitPython

It's easy to use the LSM6DS33 sensor with Python and CircuitPython, and the Adafruit CircuitPython LSM6DS module. This module allows you to easily write Python code that reads measurements from the accelerometer and gyro.

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 LSM6DS33 to your board for an I2C connection, exactly as shown below. Here's an example of wiring a Feather M4 to the sensor with I2C:

Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)
Python Computer Wiring

Since there are 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:

![Raspberry Pi wired with I2C](image)

- Pi 3V to sensor VCC (red wire)
- Pi GND to sensor GND (black wire)
- Pi SCL to sensor SCL (yellow wire)
- Pi SDA to sensor SDA (blue wire)

CircuitPython Installation of LSM6DS Library

You'll need to install the Adafruit CircuitPython LSM6DS library on your CircuitPython board. This library works with the LSM6DSOX, LSM6DS33, or ISM330DHCX

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_lsm6ds.mpy
- adafruit_bus_device
- adafruit_register

Before continuing, make sure your board's lib folder has the adafruit_lsm6ds.mpy, adafruit_bus_device, and adafruit_register files and folders copied over.

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

Python Installation of LSM6DS 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:

```bash
sudo pip3 install adafruit-circuitpython-lsm6ds
```

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 acceleration and rotation measurements from the board's Python REPL.

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

```python
import time
import board
from adafruit_lsm6ds.lsm6ds33 import LSM6DS33
```
import time
import board
from adafruit_lsm6ds.lsm6ds33 import LSM6DS33

time.sleep(1)

i2c = board.I2C()

sensor = LSM6DS33(i2c)

Now you’re ready to read values from the sensor using these properties:

- acceleration - The acceleration forces in the X, Y, and Z axes in m/s^2
- gyro - The rotation measurement on the X, Y, and Z axes in degrees/sec

For example, to print out the acceleration and gyro measurements use this code:

```python
print("Acceleration: X:%.2f, Y: %.2f, Z: %.2f m/s^2" % (sensor.acceleration))
print("Gyro X:%.2f, Y: %.2f, Z: %.2f degrees/s" % (sensor.gyro))
```

That's all it takes! Now you can use the sensor with a project where you need to know how something is moving and the direction it's pointing

Example Code

```python
# SPDX-FileCopyrightText: Copyright (c) 2020 Bryan Siepert for Adafruit Industries
# # SPDX-License-Identifier: MIT
import time
import board
from adafruit_lsm6ds.lsm6ds33 import LSM6DS33

i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a microcontroller

sensor = LSM6DS33(i2c)

while True:
    print("Acceleration: X:%.2f, Y: %.2f, Z: %.2f m/s^2" % (sensor.acceleration))
    print("Gyro X:%.2f, Y: %.2f, Z: %.2f degrees/s" % (sensor.gyro))
    print("
    time.sleep(0.5)
```

Python Docs

Python Docs
Downloads

Files

- LSM6DS33 Datasheet ()
- EagleCAD files on GitHub ()
- Fritzing object in Adafruit Fritzing Library ()

Schematic