Adafruit ADXL375

Created by Kattni Rembor
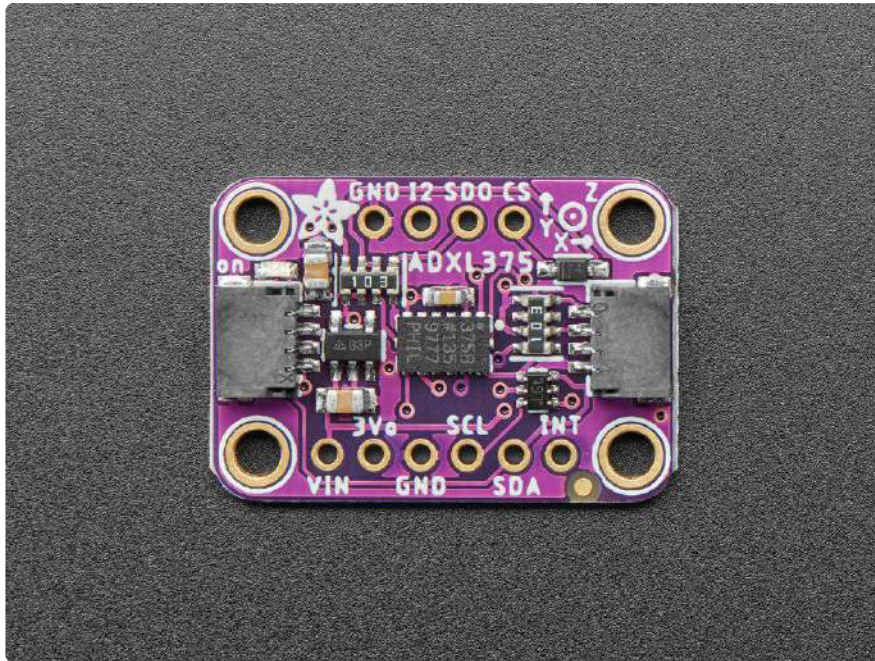

https://learn.adafruit.com/adafruit-adxl375
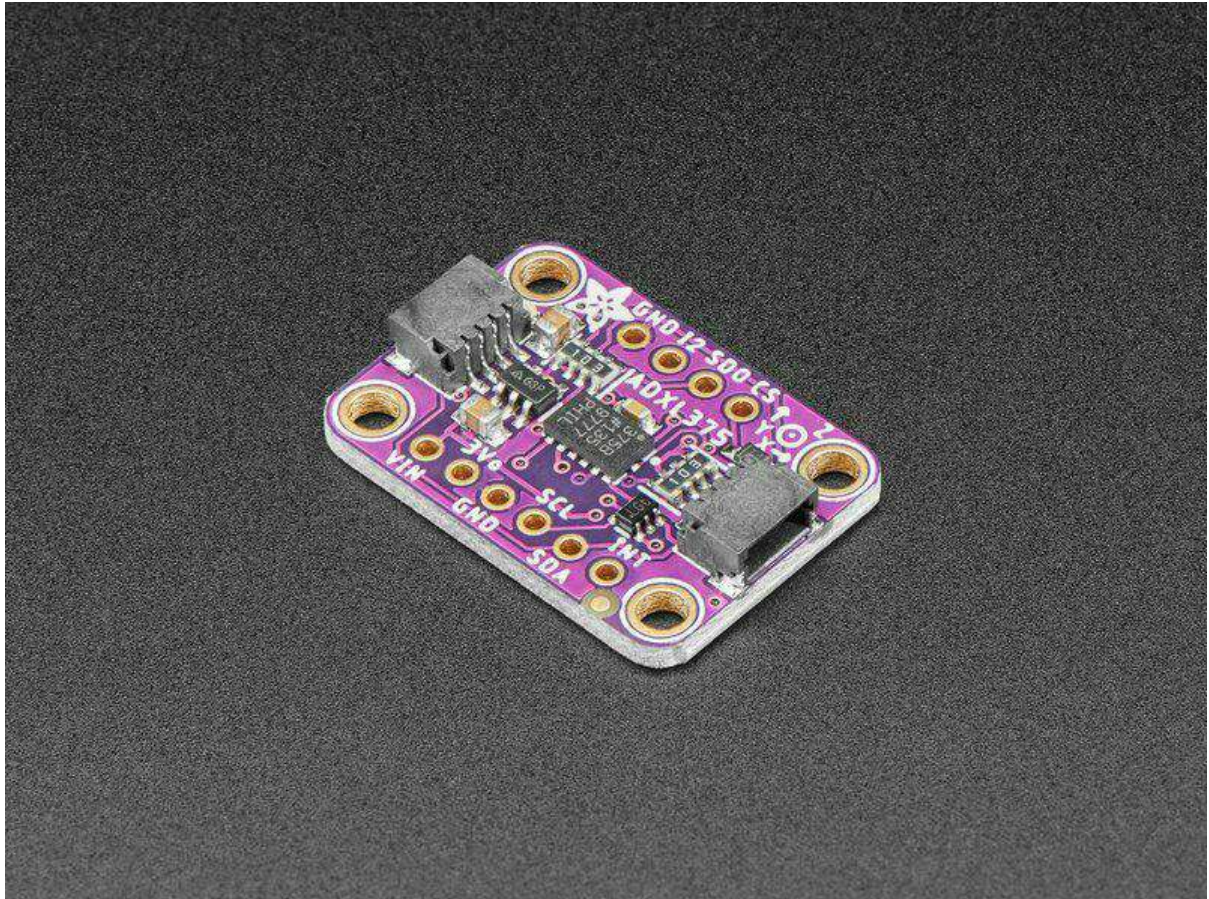
Last updated on 2022-03-17 12:15:20 PM EDT

Table of Contents

Overview

Hey rocket man (burnin' out your fuse out there alone) ever wonder how fast you're rocketing? The Adafruit ADXL375 High G Accelerometer is an epic +-200g 3-axis accelerometer may be able to tell the answer.

You read that right, this accelerometer can sense up to 200 g's of force in three axes of measurements (X Y Z) and has pins that can be used either as I2C or SPI digital interfacing for easy integration into any fast project. Built-in motion detection features make shock detection easy to implement. There are two interrupt pins, and you can map any of the interrupts independently to either of them. Incredible! Not surprisingly, we couldn't say "no" to a breakout for this sensor.
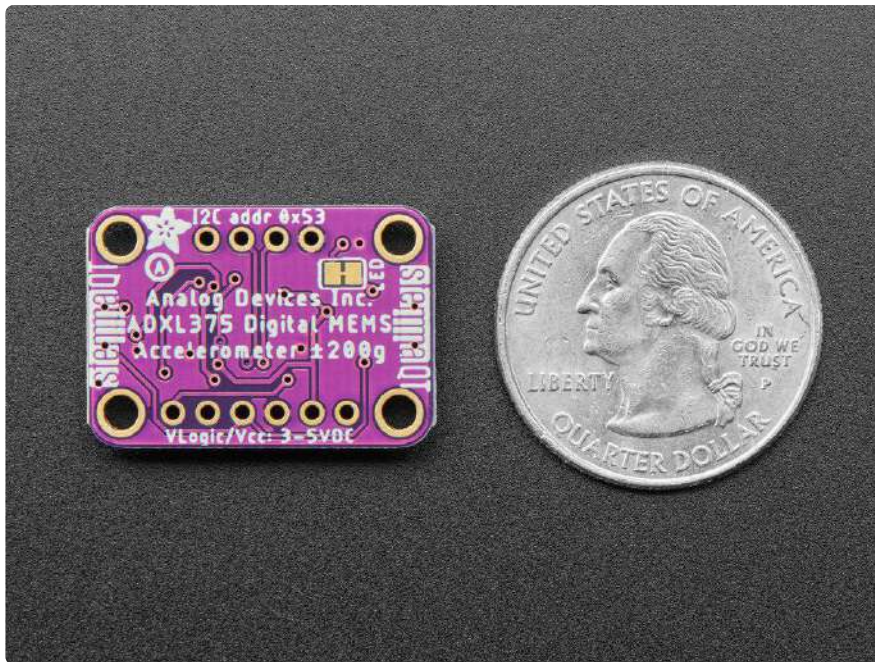
The ADXL375 looks and acts nearly identically in specifications to its little sisters, the ADXL345 and ADXL343. Those are only +-16g max, and have adjustable ranges. This sensor acts and looks the same, except that you cannot change the range, and it's fixed at 200g. Otherwise, existing library code will 'just work', so if you happen to be using something other than Arduino or CircuitPython, the port is pretty easy and code written for the '345/'343 will likely work on the '375 with just a scaling adjustment.
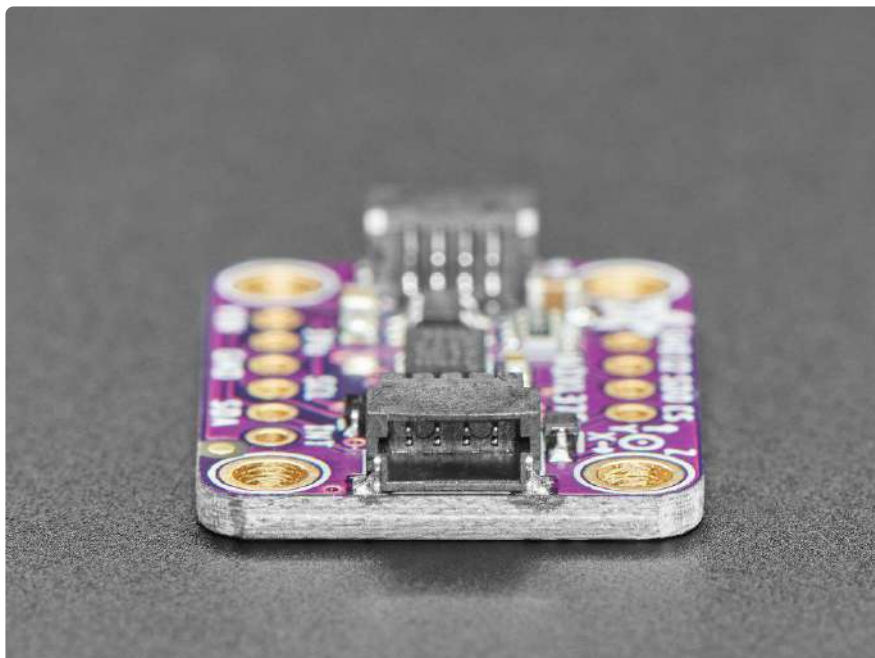
As with all Adafruit breakouts, we've done the work to make this handy accelerometer super easy to use. We've put it on a breakout board with the required support circuitry and connectors to make it easy to work with. Since I2C is supported, we've added SparkFun Qwiic (https://adafru.it/Fpw) compatible STEMMA QT (https://

adafru.it/Ft4) JST SH connectors that allow you to get going without needing to solder. Just use a STEMMA QT adapter cable (https://adafru.it/FA-), plug it into your favorite microcontroller or Blinka supported SBC and you're ready to rock! A QT Cable is not included, but we have a variety in the shop (https://adafru.it/JnB).
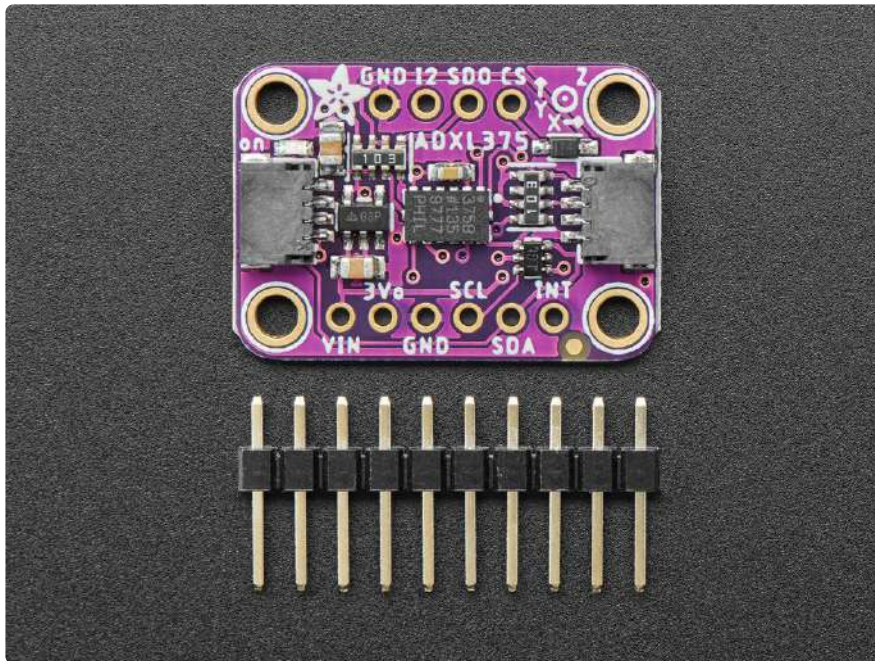
We've got both Arduino (C/C++) (https://adafru.it/Ynd)and CircuitPython (Python 3) libraries (https://adafru.it/E5S) available so you can use it with any microcontroller like Arduino, ESP8266, Metro, etc (https://adafru.it/Elp) or with Raspberry Pi or other Linux computers (https://adafru.it/Yne) thanks to Blinka (our CircuitPython library support helper).

Each order comes with a fully tested and assembled breakout and some header for soldering to a PCB or breadboard. It comes with a 9 pin 0.1" standard piece of header in case you want to use it with a breadboard or perfboard. It includes four 2.5mm (0.1") mounting holes for easy attachment. You'll be up and running in under 5 minutes!

Pinouts
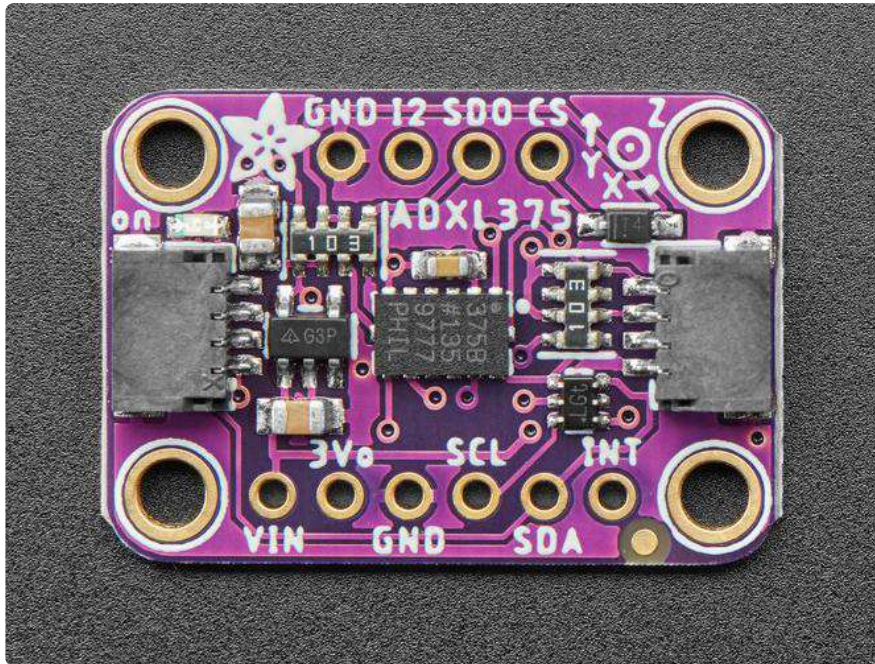
The default I2C address for this board is 0x53.

Power Pins

The sensor on the breakout requires between a 2.0V and 3.6V, but it can be easily used with most microcontrollers from an Arduino to a Feather or something else.

- VIN - This is the power pin. To power the board, give it the same power as the logic level of your microcontroller - i.e. for a 5V micro like Arduino, use 5V, or for a 3V micro like a Feather, use 3V.
- 3Vo - This is the 3.3V output from the voltage regulator. You can grab up to 100mA from this if you like.
- GND - This is common ground for power and logic.

I2C Logic Pins

- SCL - I2C clock pin, connect to your microcontroller's 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's I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- CS - This is the Chip Select pin. It is tied to 3V by default to enable I2C mode.

- **STEMMA QT** (https://adafru.it/Ft4) - These connectors allow you to connect to development boards with STEMMA QT connectors, or to other things, with [various associated accessories](https://adafru.it/JRA) (https://adafru.it/JRA).

SPI Logic Pins

- SCL - This is the SPI Clock pin. It's an input to the chip.
- SDA - This is the Serial Data In / Microcontroller Out Sensor In pin. It is for data sent from your microcontroller to the accelerometer.
- SDO - This is the Serial Data Out / Microcontroller In Sensor Out pin. It is for data sent from the accelerometer to your microcontroller.
- CS - This is the Chip Select pin. When in SPI mode, it is controlled by the SPI bus. Drop this pin low to start a SPI transaction. It's an input to the chip.

Other Pins

- INT / I2 - These are the two interrupt pins. You can configure the interrupt to trigger on multiple things such as threshold detection, shock detection and data ready. You can map any of the interrupts independently to each pin.

On LED Jumper

- LED jumper - On the back of the board is a jumper for the power LED. If you wish to disable the power LED, simply cut the trace on this jumper.


Python & CircuitPython

It's easy to use the Adafruit ADXL375 with Python or CircuitPython, and the [Adafruit CircuitPython ADXL37x](https://adafru.it/YsF) (https://adafru.it/YsF) module. This module allows you to easily write Python code that reads the acceleration and more 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](https://adafru.it/BSN) (https://adafru.it/BSN).

CircuitPython Microcontroller Wiring

First wire up a ADXL375 to your board exactly as shown below. Here's an example of wiring a Feather M4 to the sensor with I2C using one of the handy STEMMA QT (https://adafru.it/Ft4) connectors:
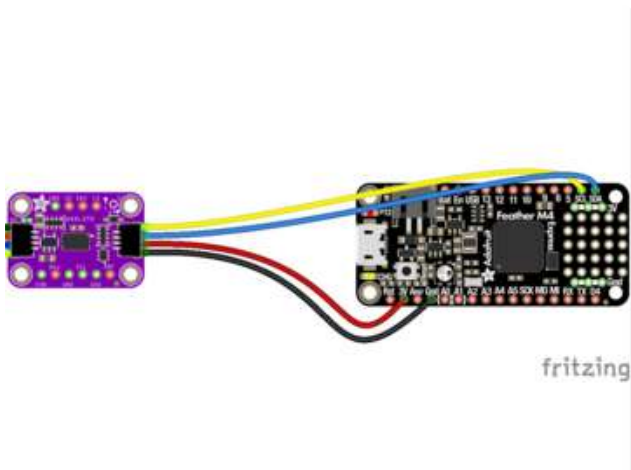
Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

You can also use the standard 0.100" pitch headers to wire it up on a breadboard:
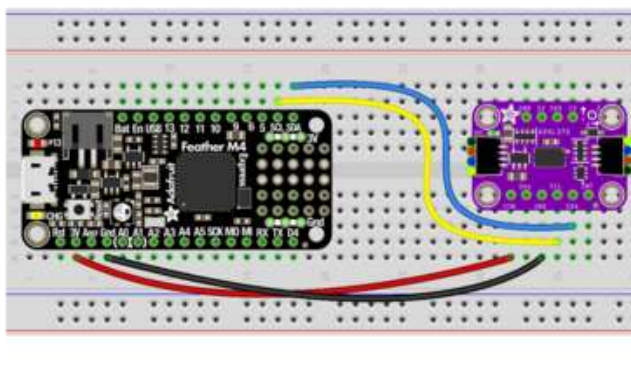
Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

Python Computer Wiring

Since there'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 (https://adafru.it/BSN).

Here's the Raspberry Pi wired to the sensor using I2C and a STEMMA QT (https://adafru.it/Ft4) connector:
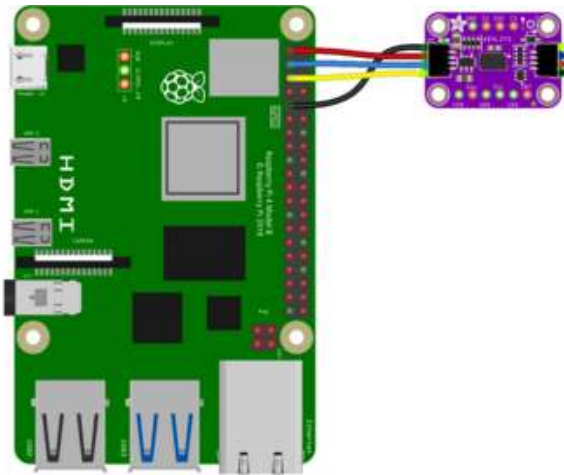
Pi 3V to sensor VIN (red wire)
Pi GND to sensor GND (black wire)
Pi SCL to sensor SCL (yellow wire)
Pi SDA to sensor SDA (blue wire)

Finally here is an example of how to wire up a Raspberry Pi to the sensor using a solderless breadboard:
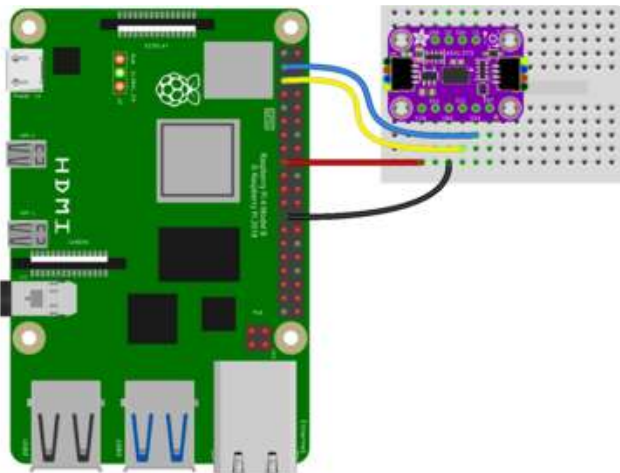
Pi 3V to sensor VIN (red wire)
Pi GND to sensor GND (black wire)
Pi SCL to sensor SCL (yellow wire)
Pi SDA to sensor SDA (blue wire)

Python Installation of ADXL37x 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 (https://adafru.it/BSN)!

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

- `pip3 install adafruit-circuitpython-adxl37x`

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

CircuitPython Usage

To use with CircuitPython, you need to first install the ADXL37x library, and its 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 folder and file:

- adafruit_bus_device/
- adafruit_adxl34x.mpy
- adafruit_adxl37x.mpy

Python Usage

Once you have the library `pip3` installed on your computer, copy or download the following example to your computer, and run the following, replacing code.py with whatever you named the file:

`python3 code.py`

Example Code

```
# SPDX-FileCopyrightText: Copyright (c) 2022 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: Unlicense
import time
import board
import adafruit_adxl37x
```

```
i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a
microcontroller
accelerometer = adafruit_adxl37x.ADXL375(i2c)

while True:
    print("%f %f %f m/s^2" % accelerometer.acceleration)
    time.sleep(0.2)
```

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

If running Python: The console output will appear wherever you are running Python.

Try moving the breakout to see the values change! Acceleration is in m/s^2

First you import the necessary modules and libraries. Then you instantiate the sensor on I2C.

Then you're ready to read data from the sensor. Inside the loop, you check the acceleration every 0.2 seconds.

That's all there is to using the ADXL375 with CircuitPython!

Python Docs

Python Docs (https://adafru.it/Yta)

# Arduino

Using the ADXL375 with Arduino involves wiring up the sensor to your Arduino-compatible microcontroller, installing the Adafruit ADXL375 (https://adafru.it/Ynd) library and running the provided example code.

Wiring

Wire as shown for a 5V board like an Uno. If you are using a 3V board, like an Adafruit Feather, wire the board's 3V pin to the ADXL374 VIN.

Here is an Adafruit Metro wired up to the ADXL375 using the STEMMA QT connector:
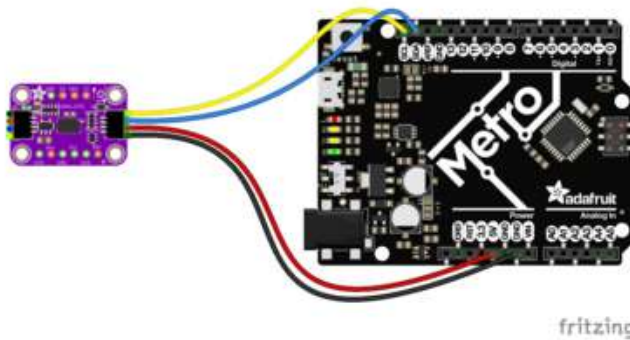
Board 5V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

Here is an Adafruit Metro wired up using a solderless breadboard:
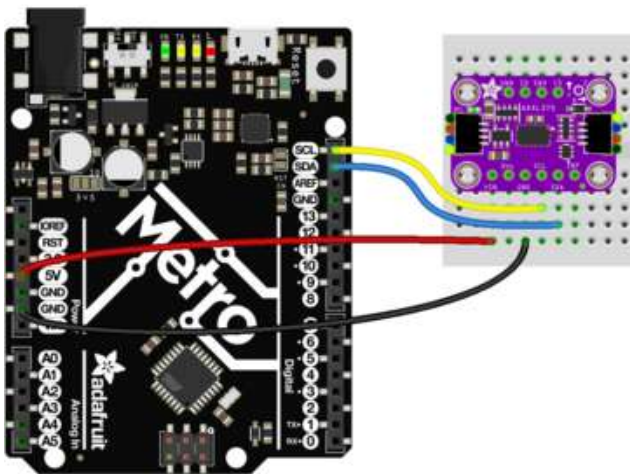
Board 5V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

Library Installation

You can install the ADXL375 library for Arduino using the Library Manager in the Arduino IDE.
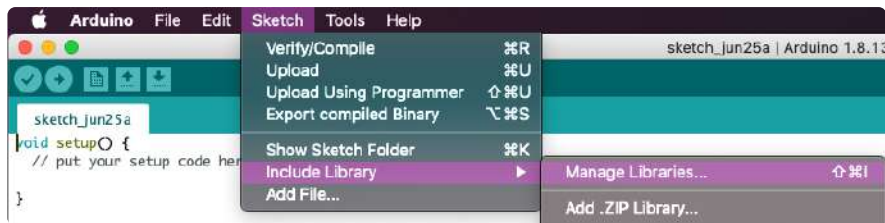
Click the Manage Libraries ... menu item, search for ADXL375 , and select the Adafruit ADXL375 library:
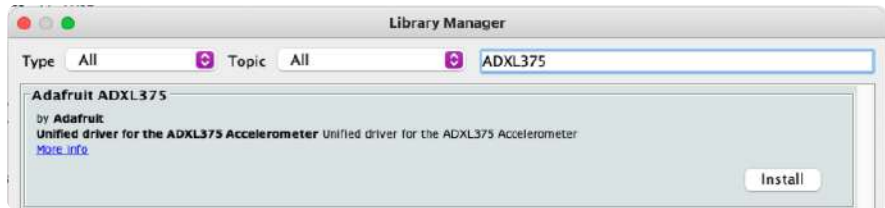
When asked about dependencies, click "Install all".

Load Example

Open up File -> Examples -> Adafruit ADXL375 -> sensortest and upload to your Arduino wired to the sensor.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL375.h>

#define ADXL375_SCK 13
#define ADXL375_MISO 12
#define ADXL375_MOSI 11
#define ADXL375_CS 10

/* Assign a unique ID to this sensor at the same time */
/* Uncomment following line for default Wire bus      */
Adafruit_ADXL375 accel = Adafruit_ADXL375(12345);

/* Uncomment for SPI */
//Adafruit_ADXL375 accel = Adafruit_ADXL375(ADXL375_SCK, ADXL375_MISO,
```

```cpp
ADXL375_MOSI, ADXL375_CS, 12345);
void displayDataRate(void)
{
  Serial.print  ("Data Rate:    ");

  switch(accel.getDataRate())
  {
    case ADXL343_DATARATE_3200_HZ:
      Serial.print  ("3200 ");
      break;
    case ADXL343_DATARATE_1600_HZ:
      Serial.print  ("1600 ");
      break;
    case ADXL343_DATARATE_800_HZ:
      Serial.print  ("800 ");
      break;
    case ADXL343_DATARATE_400_HZ:
      Serial.print  ("400 ");
      break;
    case ADXL343_DATARATE_200_HZ:
      Serial.print  ("200 ");
      break;
    case ADXL343_DATARATE_100_HZ:
      Serial.print  ("100 ");
      break;
    case ADXL343_DATARATE_50_HZ:
      Serial.print  ("50 ");
      break;
    case ADXL343_DATARATE_25_HZ:
      Serial.print  ("25 ");
      break;
    case ADXL343_DATARATE_12_5_HZ:
      Serial.print  ("12.5 ");
      break;
    case ADXL343_DATARATE_6_25HZ:
      Serial.print  ("6.25 ");
      break;
    case ADXL343_DATARATE_3_13_HZ:
      Serial.print  ("3.13 ");
      break;
    case ADXL343_DATARATE_1_56_HZ:
      Serial.print  ("1.56 ");
      break;
    case ADXL343_DATARATE_0_78_HZ:
      Serial.print  ("0.78 ");
      break;
    case ADXL343_DATARATE_0_39_HZ:
      Serial.print  ("0.39 ");
      break;
    case ADXL343_DATARATE_0_20_HZ:
      Serial.print  ("0.20 ");
      break;
    case ADXL343_DATARATE_0_10_HZ:
      Serial.print  ("0.10 ");
      break;
    default:
      Serial.print  ("???? ");
      break;
  }
  Serial.println(" Hz");
}

void setup(void)
{
  Serial.begin(115200);
  while (!Serial);
  Serial.println("ADXL375 Accelerometer Test"); Serial.println("");
```

```
    /* Initialise the sensor */
    if(!accel.begin())
    {
        /* There was a problem detecting the ADXL375 ... check your connections */
        Serial.println("Ooops, no ADXL375 detected ... Check your wiring!");
        while(1);
    }

    // Range is fixed at +-200g

    /* Display some basic information on this sensor */
    accel.printSensorDetails();
    displayDataRate();
    Serial.println("");
}

void loop(void)
{
    /* Get a new sensor event */
    sensors_event_t event;
    accel.getEvent(&event);

    /* Display the results (acceleration is measured in m/s^2) */
    Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print("  ");
    Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print("  ");
    Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.print("  
");Serial.println("m/s^2 ");
    delay(500);
}
```

Upload the sketch to your board and open up the Serial Monitor (Tools -> Serial Monitor) at 115200 baud. You should see the the values from the senor being printed out.

Arduino Docs

Arduino Docs (https://adafru.it/YwA)

Downloads

Files:

- ADXL375 Datasheet (https://adafru.it/Yqa)
- EagleCAD PCB Files on GitHub (https://adafru.it/Yqb)
- Fritzing object in the Adafruit Fritzing Library (https://adafru.it/Yqc)

Schematic and Fab Print for ADXL375