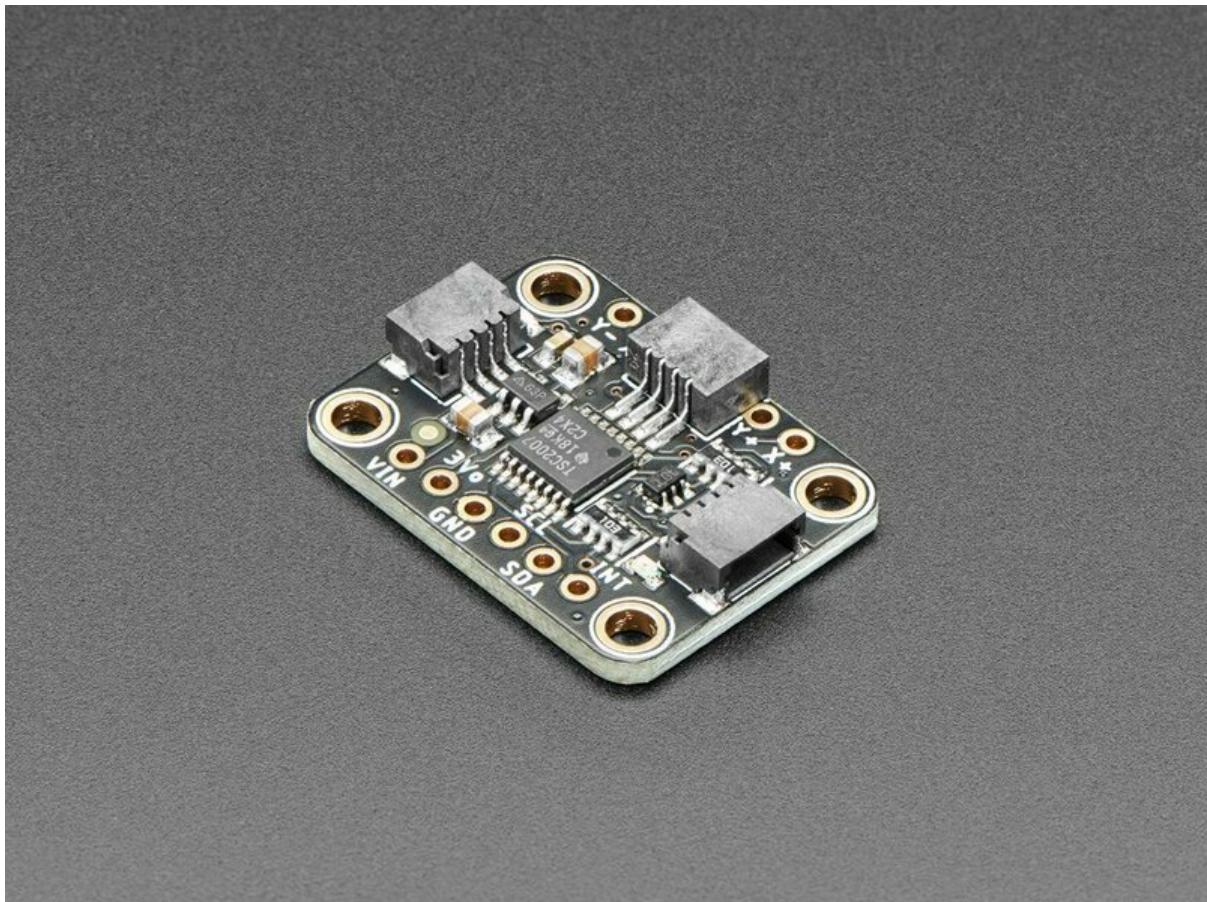




# Adafruit TSC2007 I2C Resistive Touch Screen Controller

Created by Liz Clark



<https://learn.adafruit.com/adafruit-tsc2007-i2c-resistive-touch-screen-controller>

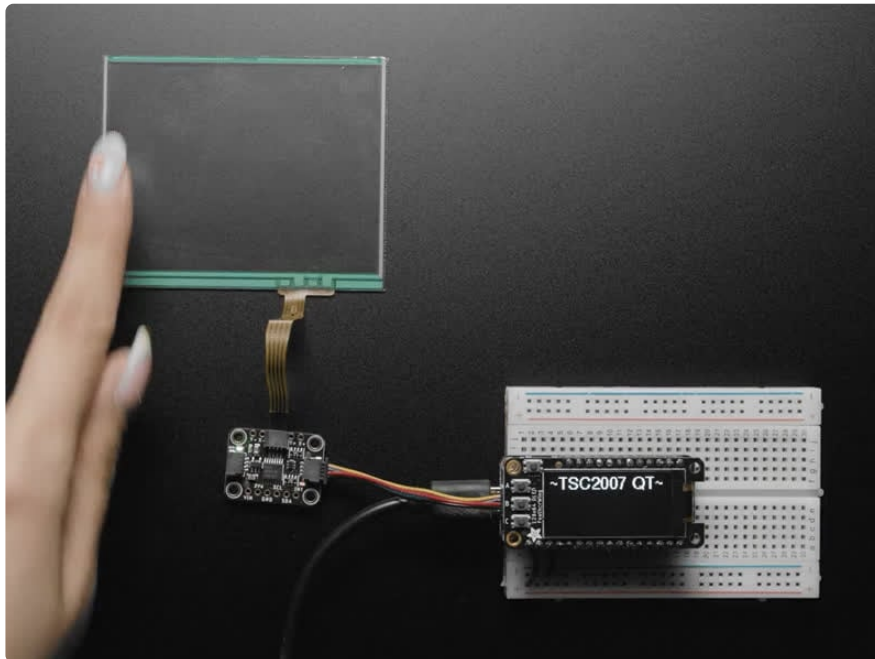
Last updated on 2025-03-21 11:01:11 PM EDT

# Table of Contents

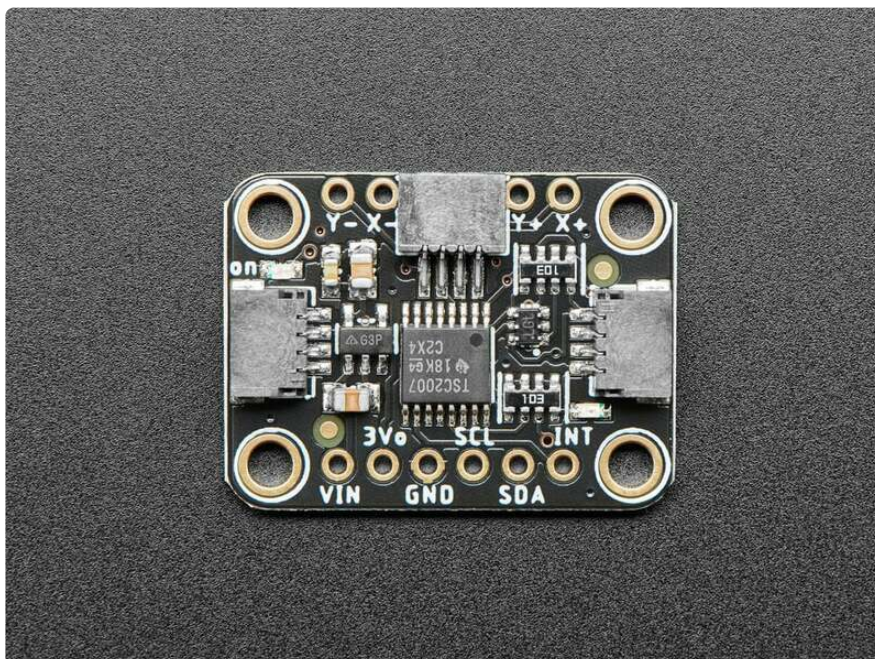
Overview	3
Pinouts	5
<ul style="list-style-type: none"><li>• Power Pins</li><li>• I2C Logic Pins</li><li>• Input Pins</li><li>• Interrupt Pins</li><li>• Address Pins</li><li>• Power LED and Jumper</li></ul>	
Python & CircuitPython	8
<ul style="list-style-type: none"><li>• CircuitPython Microcontroller Wiring</li><li>• Python Computer Wiring</li><li>• Python Installation of TSC2007 Library</li><li>• CircuitPython Usage</li><li>• Python Usage</li><li>• Example Code</li></ul>	
Python Docs	12
Arduino	12
<ul style="list-style-type: none"><li>• Wiring</li><li>• Library Installation</li><li>• Load Example</li></ul>	
Arduino Docs	15
Downloads	15
<ul style="list-style-type: none"><li>• Files</li><li>• Schematic and Fab Print</li></ul>	

---

# Overview

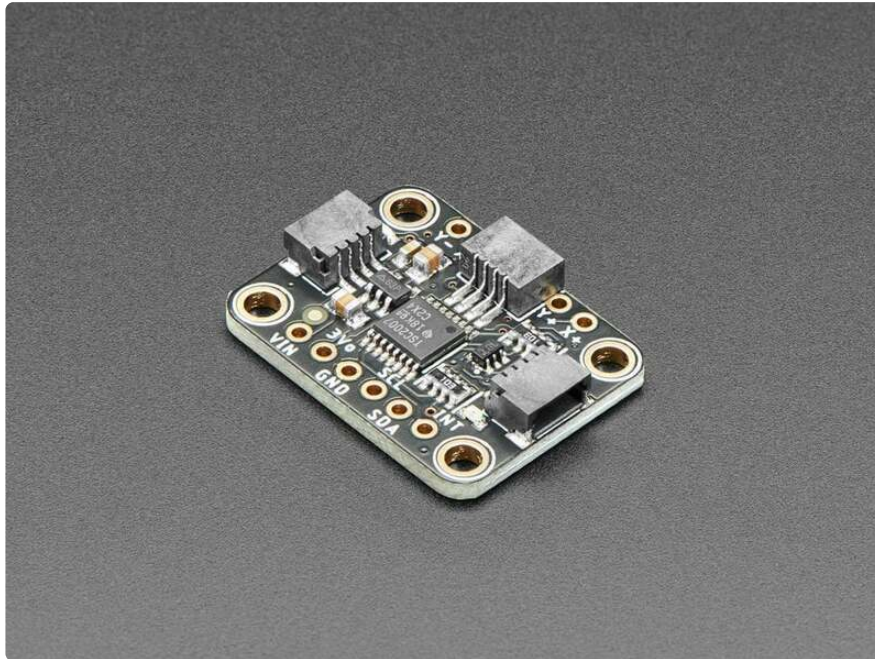


Getting touchy performance with your touch screen? Resistive touch screens are incredibly popular as overlays to TFT and LCD displays. The only problem is they require a bunch of analog pins and you have to keep polling them, since the overlays themselves are basically just big potentiometers. If your microcontroller doesn't have analog inputs, or maybe you want just a way more elegant controller, the TSC2007 is a nice way to solve that problem.



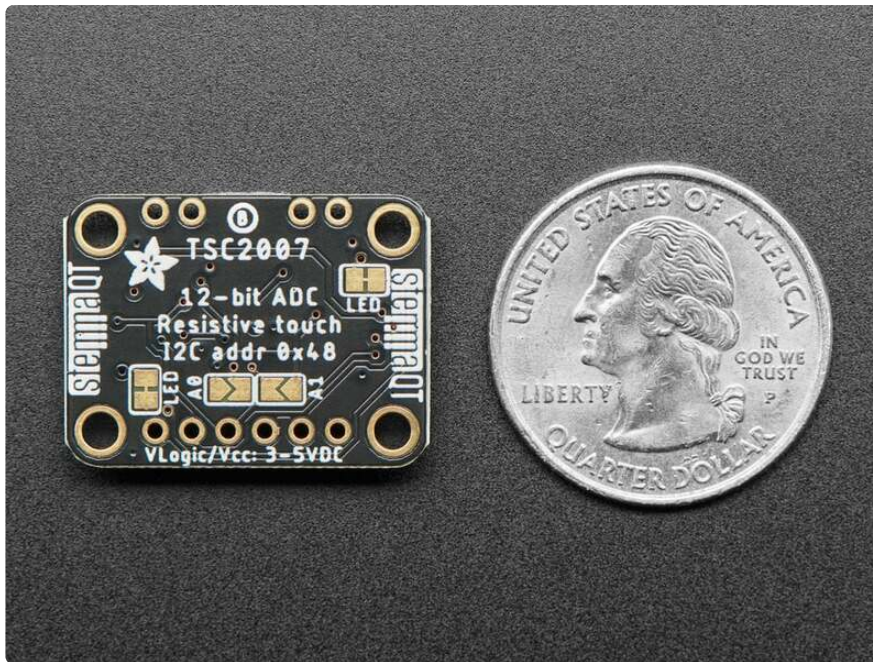
This breakout board features the TSC2007, which has an easy-to-use I2C interface. There is also an interrupt pin that you can use to indicate when a touch has been detected to your microcontroller or microcomputer.

We wrapped up the chip with a 3V voltage regulator and level shifting so it's safe to use with 3V or 5V logic. It's a nicely designed chip, and has very stable, precise readings. We found it's also a lot faster than trying to do all the readings on an Arduino.



For the screens that have 1mm pitch FPC cables, you can plug the cable right into the connector. The majority of medium/large touchscreens have that kind of connector. If you have another kind of touch screen, the four X/Y contacts are available on 0.1" pitch breakouts so you can hand-solder or wire them.

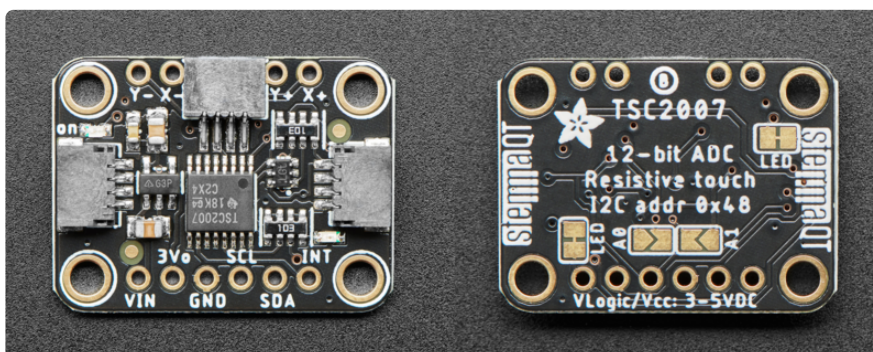




Getting started is super easy with our simple [TSC2007 Arduino library \(https://adafru.it/d4f\)](https://adafru.it/d4f) or [TSC2007 CircuitPython/Python library for microcontrollers or Raspberry Pi \(https://adafru.it/ZgD\)](https://adafru.it/ZgD). Plug any 1mm-pitch 4-wire resistive touchscreen to the on-board FPC connector, then use the library example to read touch points with X, Y and Z (pressure) results returned instantaneously.

There's an IRQ pin that will drop low when a touch is detected. You can use that to reduce the I2C polling. We also have a red LED on that line which can help debugging as it should light when the panel is touched.

## Pinouts



## Power Pins

- **VIN** - This is the power pin. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 3V microcontroller like a Feather M4, use 3V, or for a 5V microcontroller like Arduino, use 5V.
- **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

The default I2C address for the TSC2007 is **0x48**.

- **SCL** - I2C clock pin, connect to your microcontroller I2C clock line. There's a **10K pullup** on this pin.
- **SDA** - I2C data pin, connect to your microcontroller I2C data line. There's a **10K pullup** on this pin.
- **STEMMA QT** (<https://adafru.it/Ft4>) - These connectors allow you to connect to development boards with **STEMMA QT** / Qwiic connectors or to other things with [various associated accessories](https://adafru.it/JRA) (<https://adafru.it/JRA>).

## Input Pins

- **FPC Connector** - FPC socket for a 1mm-pitch 4-wire resistive touchscreen located at the top of the board. This reads the X/Y contacts directly from a 4-wire FPC cable.
- **Y-, X-, Y+, X+** - These are the X/Y contacts available on 0.1" pitch breakouts. If you have another kind of touch screen that does not have a 1mm pitch FPC cable then you can solder directly to these points.

## Interrupt Pins

- **INT** - This is the interrupt output pin. It will drop low when a touch is detected. You can use this to reduce the I2C polling.
- **Interrupt LED** - On the front of the board in the lower right corner, above the **INT** pin, is the interrupt LED. It is the red LED and turns on when a connected panel is touched.
- **Interrupt LED jumper** - In the lower right corner on the back of the board is a jumper for the interrupt LED. If you wish to disable the interrupt LED, simply cut the trace on this jumper.

Settings an alternate I2C address on the TSC2007 requires a resistor across the jumper pad. Using a Pull-Up between 2k - 10k ohms is ideal. An 0805 SMD fits well on the jumper pad.

## Address Pins

On the back of the board are **two address jumpers**, labeled **A0** and **A1**, below the **I2C Addr** label on the board silk. These jumpers allow you to chain up to 4 of these boards on the same pair of I2C clock and data pins. To do so, you solder the jumpers "closed" by connecting the two pads.

The default I2C address is **0x48**. The other address options can be calculated by "adding" the **A0/A1** to the base of **0x48**.

**A0** sets the lowest bit with a value of **1**, **A1** sets the next bit with a value of **2**. The final address is **0x48 + A1 + A0** which would be **0x4B**.

If only **A0** is soldered closed, the address is **0x48 + 1 = 0x49**

If only **A1** is soldered closed, the address is **0x48 + 2 = 0x4A**

The table below shows all possible addresses, and whether the pin(s) should be high (closed) or low (open).

ADDR	A0	A1
0x48	L	L
0x49	H	L
0x4A	L	H
0x4B	H	H

## Power LED and Jumper

- **Power LED** - In the upper left corner, above the STEMMA connector, on the front of the board, is the power LED, labeled **on**. It is the green LED.
- **LED jumper** - In the upper right corner 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.

---

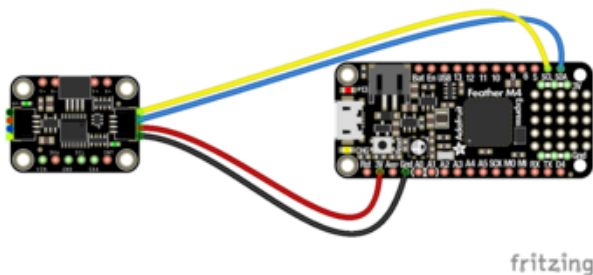
# Python & CircuitPython

It's easy to use the **TSC2007** with Python or CircuitPython, and the [Adafruit CircuitPython TSC2007 \(https://adafruit.it/ZgD\)](https://adafruit.it/ZgD) module. This module allows you to easily write Python code that reads the distance from the **TSC2007** 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 \(https://adafruit.it/BSN\)](https://adafruit.it/BSN).

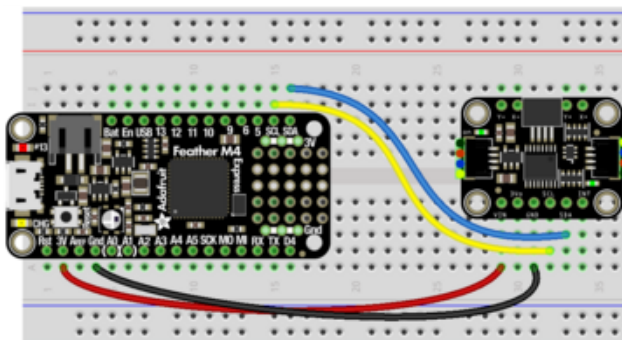
## CircuitPython Microcontroller Wiring

First, wire up a TSC2007 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 \(https://adafruit.it/Ft4\)](https://adafruit.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:



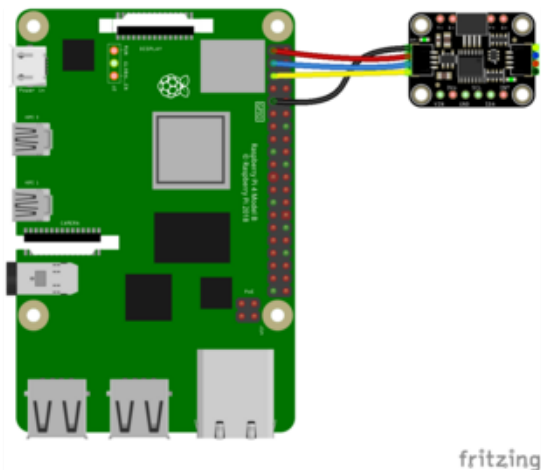
Board 3V to sensor VIN (red wire)  
Board GND to sensor GND (black wire)  
Board SCL to sensor SCL (yellow wire)  
Board SDA to sensor SDA (blue wire)



# Python Computer Wiring

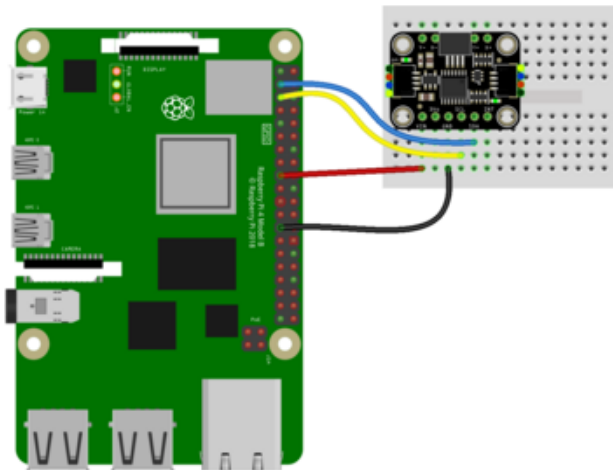
Since there's dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, [please visit the guide for CircuitPython on Linux to see whether your platform is supported](https://adafru.it/BSN) (<https://adafru.it/BSN>).

Here's the Raspberry Pi wired to the sensor using I2C and a [STEMMA QT](https://adafru.it/Ft4) (<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 TSC2007 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) (<https://adafru.it/BSN>)!

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

- `pip3 install adafruit-circuitpython-tsc2007`

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 TSC2007 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\_tsc2007.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 ladyada for Adafruit Industries
#
# SPDX-License-Identifier: Unlicense
```

```

import board
import adafruit_tsc2007

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

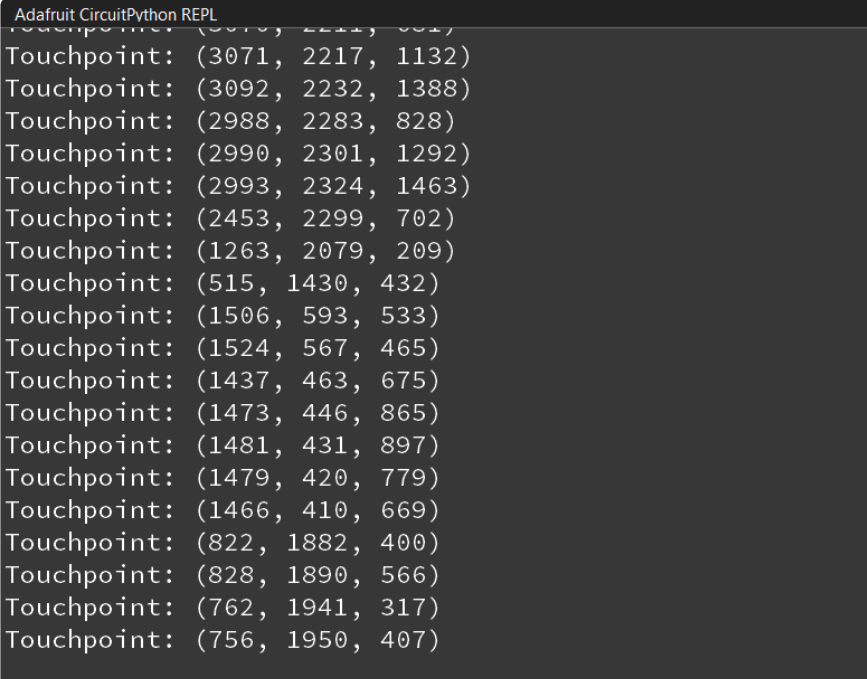
irq_dio = None # don't use an irq pin by default
# uncomment for optional irq input pin so we don't continuously poll the I2C for
touches
# irq_dio = digitalio.DigitalInOut(board.A0)
tsc = adafruit_tsc2007.TSC2007(i2c, irq=irq_dio)

while True:
    if tsc.touched:
        point = tsc.touch
        if point["pressure"] < 100: # ignore touches with no 'pressure' as false
            continue
        print("Touchpoint: (%d, %d, %d)" % (point["x"], point["y"],
point["pressure"]))

```

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

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



```

Adafruit CircuitPython REPL
Touchpoint: (3071, 2217, 1132)
Touchpoint: (3092, 2232, 1388)
Touchpoint: (2988, 2283, 828)
Touchpoint: (2990, 2301, 1292)
Touchpoint: (2993, 2324, 1463)
Touchpoint: (2453, 2299, 702)
Touchpoint: (1263, 2079, 209)
Touchpoint: (515, 1430, 432)
Touchpoint: (1506, 593, 533)
Touchpoint: (1524, 567, 465)
Touchpoint: (1437, 463, 675)
Touchpoint: (1473, 446, 865)
Touchpoint: (1481, 431, 897)
Touchpoint: (1479, 420, 779)
Touchpoint: (1466, 410, 669)
Touchpoint: (822, 1882, 400)
Touchpoint: (828, 1890, 566)
Touchpoint: (762, 1941, 317)
Touchpoint: (756, 1950, 407)

```

Attach a resistive touch screen to the TSC2007 with the FPC connector or the four X/Y contacts. Now try touching different spots on your attached resistive touchscreen to see the values change!

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, including the initial information printed to the serial console.

Finally, inside the loop, you check the coordinates where pressure is detected.

The first number is the X coordinate, the second number is the Y coordinate and the last number is the Z "pressure" coordinates that can tell you how hard the touch pad is being pressed

That's all there is to using the TSC2007 with CircuitPython!

---

## Python Docs

[Python Docs \(https://adafru.it/ZgF\)](https://adafru.it/ZgF)

---

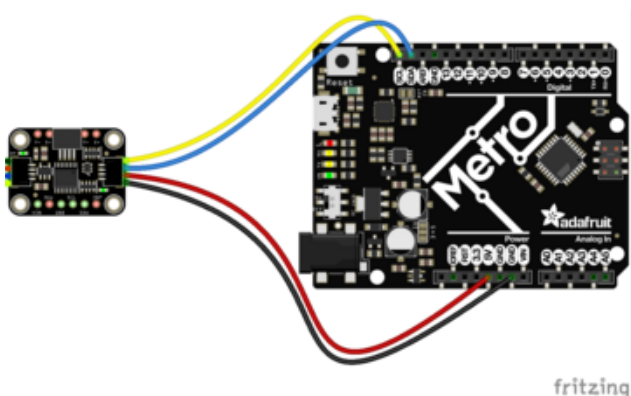
## Arduino

Using the TSC2007 with Arduino involves wiring up the expander to your Arduino-compatible microcontroller, installing the [Adafruit TSC2007 \(https://adafru.it/Zha\)](https://adafru.it/Zha) 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 TSC2007 VIN.

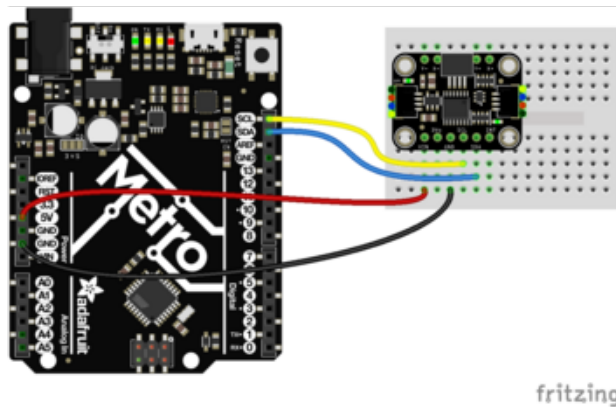
Here is an Adafruit Metro wired up to the TSC2007 using the STEMMA QT connector:



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:

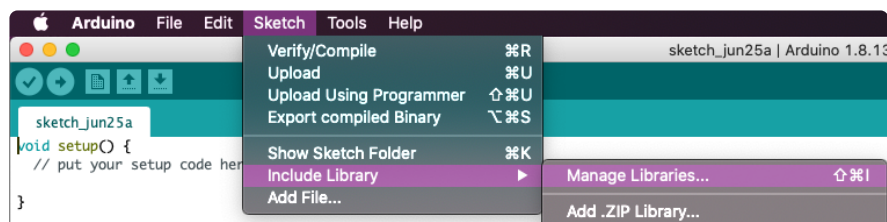




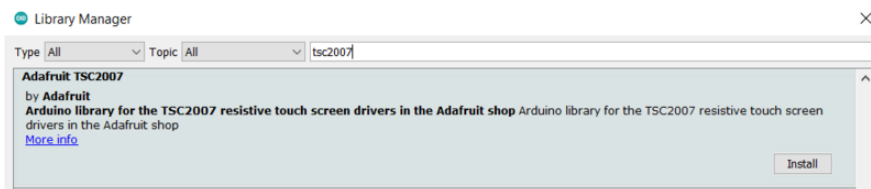
Board 5V to sensor VIN (red wire)  
 Board GND to sensor GND (black wire)  
 Board SCL to sensor SCL (yellow wire)  
 Board SDA to sensor SDA (blue wire)

## Library Installation

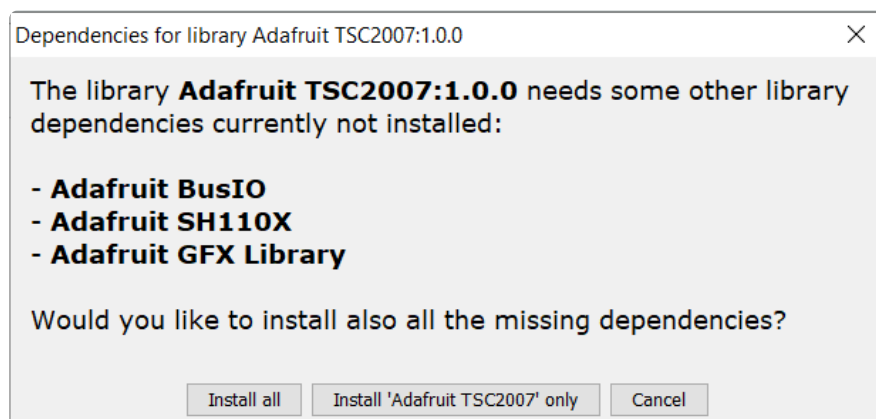
You can install the **TSC2007** library for Arduino using the Library Manager in the Arduino IDE.



Click the **Manage Libraries ...** menu item, search for **TSC2007**, and select the **Adafruit TSC2007** 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!

## Load Example

Open up **File -> Examples -> Adafruit TSC2007 -> tsc2007\_demo** and upload to your Arduino wired to the sensor.

```
#include "Adafruit_TSC2007.h"

Adafruit_TSC2007 touch;

void setup() {
  Serial.begin(115200);
  while (!Serial) delay(10);

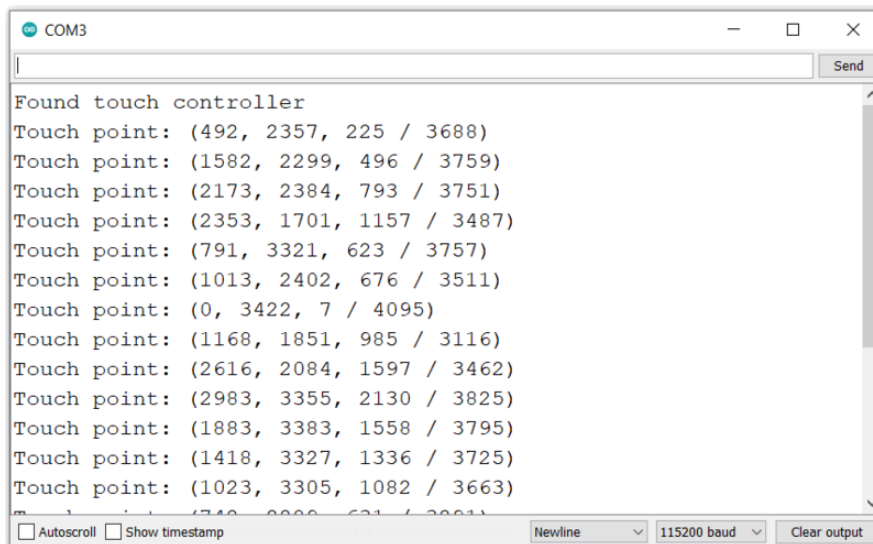
  if (!touch.begin()) {
    Serial.println("Couldn't find touch controller");
    while (1) delay(10);
  }
  Serial.println("Found touch controller");
}

void loop() {
  uint16_t x, y, z1, z2;
  if (touch.read_touch(&x, &y, &z1, &z2)) {
    Serial.print("Touch point: (");
    Serial.print(x); Serial.print(", ");
    Serial.print(y); Serial.print(", ");
    Serial.print(z1); Serial.print(" / ");
    Serial.print(z2); Serial.println(")");
  }

  delay(100);
}
```

Connect a resistive touchscreen to the TSC2007. Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at 115200 baud. You should see the values from the sensor being printed out.

The first number is the X coordinate, the second number is the Y coordinate and the last two numbers are Z "pressure" coordinates that can tell you how hard the touch pad is being pressed



## Arduino Docs

[Arduino Docs \(https://adafru.it/Zha\)](https://adafru.it/Zha)

## Downloads

### Files

- [TSC2007 Datasheet \(https://adafru.it/Zic\)](https://adafru.it/Zic)
- [EagleCAD PCB files on GitHub \(https://adafru.it/Zid\)](https://adafru.it/Zid)
- [3D Models on GitHub \(https://adafru.it/11dH\)](https://adafru.it/11dH)
- [Fritzing object in the Adafruit Fritzing Library \(https://adafru.it/Zie\)](https://adafru.it/Zie)

## Schematic and Fab Print

