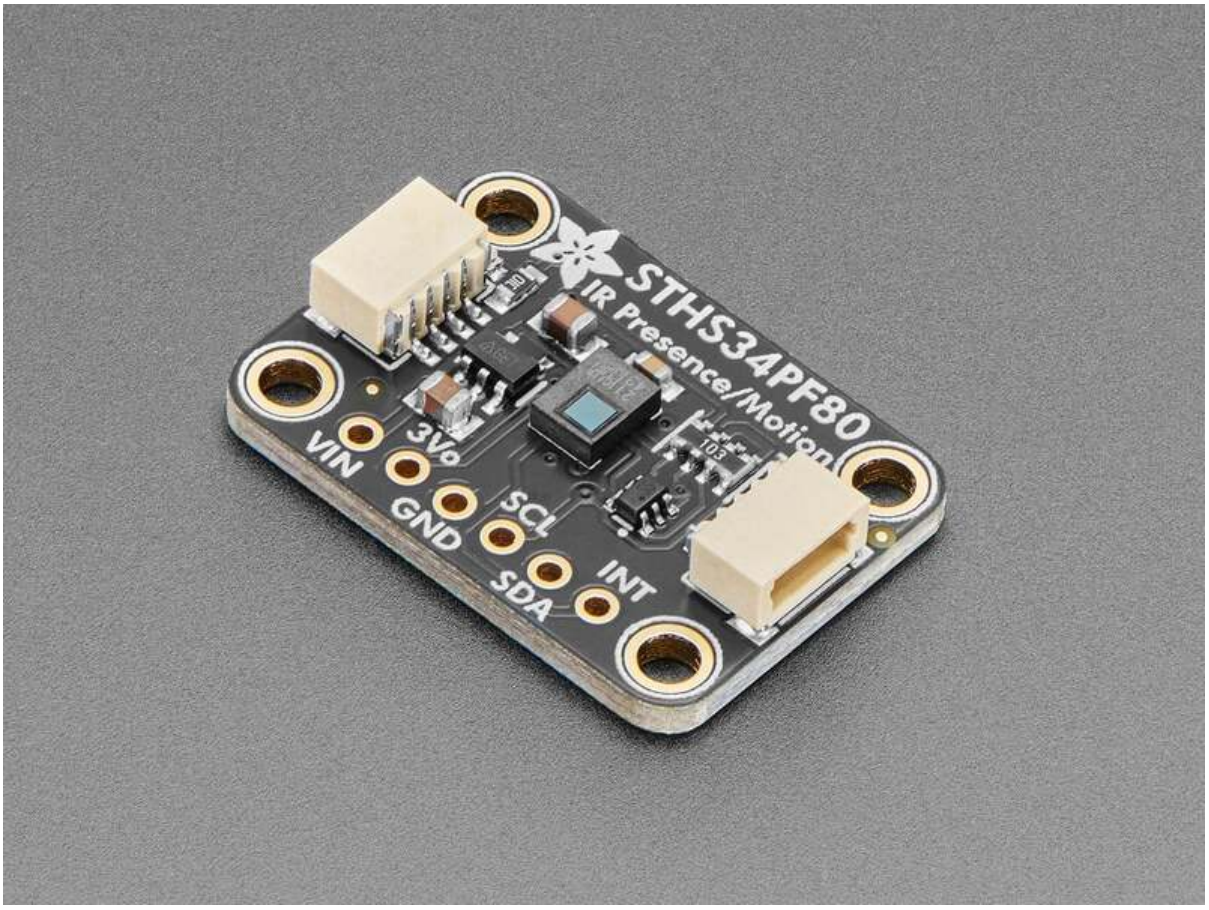
# Adafruit STHS34PF80 IR Presence / Motion Sensor

Created by Liz Clark




https://learn.adafruit.com/adafruit-sths34pf80-ir-presence-motion-sensor

Last updated on 2025-10-22 02:12:09 PM EDT

# Table of Contents


Overview 3

Pinouts 5

- Power Pins
- I2C Logic Pins
- Interrupt Pin
- Power LED and Jumper

CircuitPython and Python 6

- CircuitPython Microcontroller Wiring
- Python Computer Wiring
- Python Installation of STHS34PF80 Library
- CircuitPython Usage
- Python Usage
- Example Code

Python Docs 11

Arduino 11

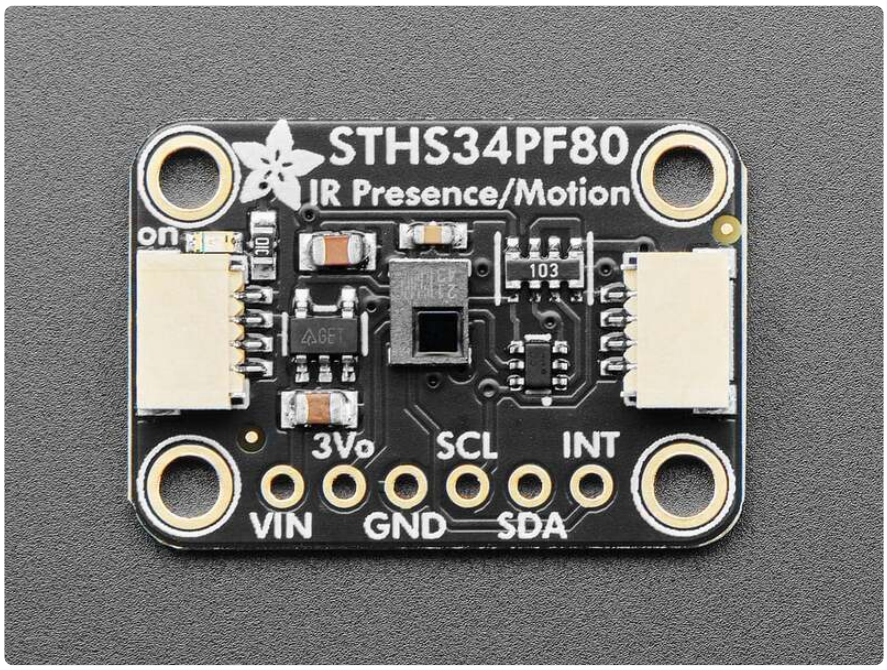
- Wiring
- Library Installation
- Example Code

Arduino Docs 15

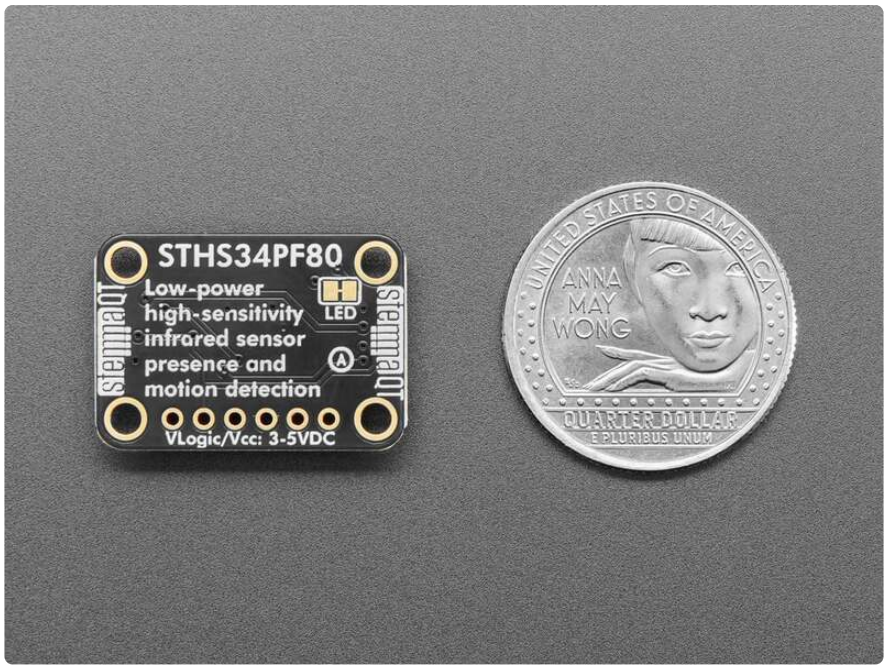
Downloads 15

- Files
- Schematic and Fab Print

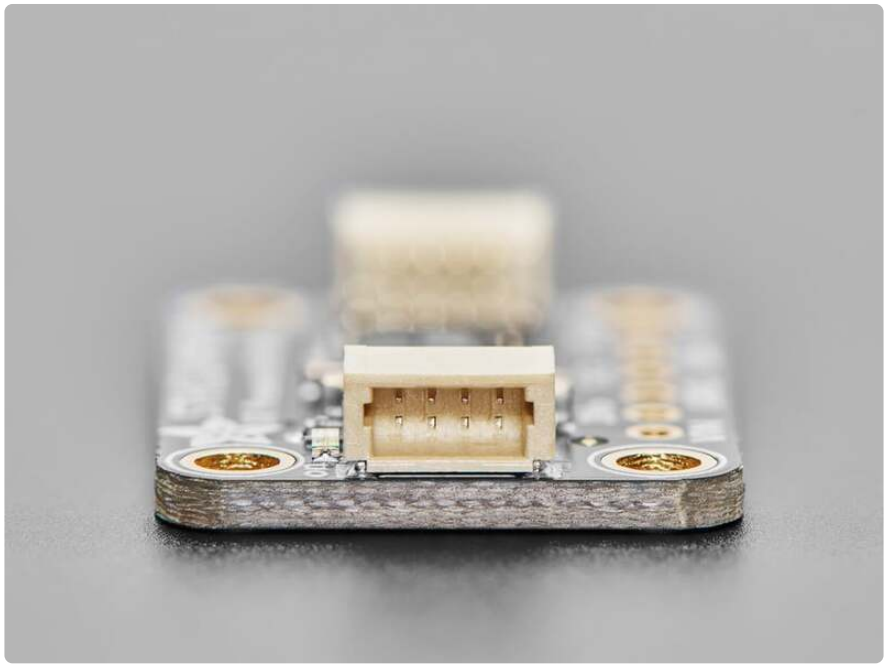
# Overview



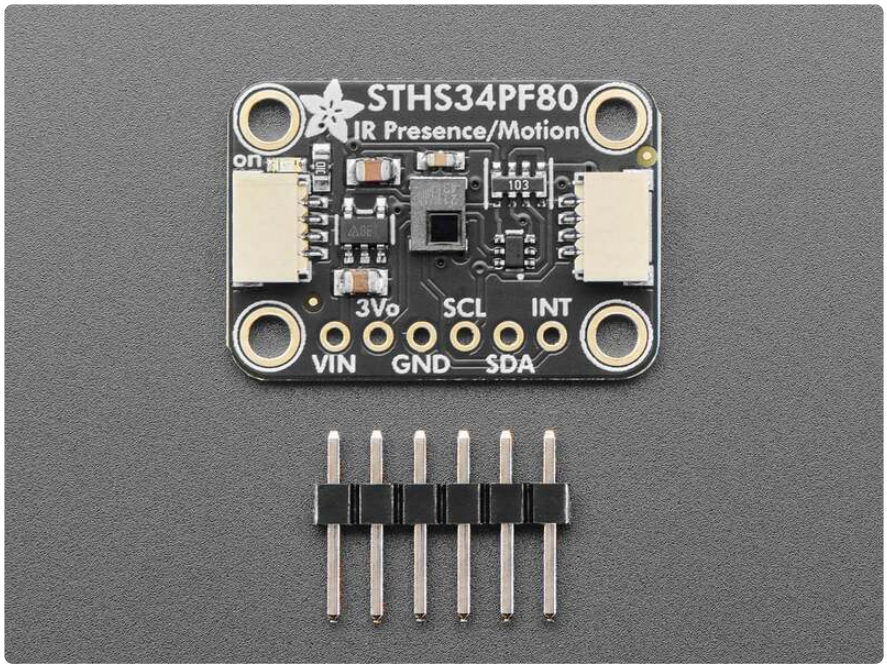

We stock many low cost "Passive IR" Presence / Motion sensors (https://adafru.it/1auu), and they're very popular for detecting motion from humanoids and pets 5 to 20 feet away. (They possibly work on zombies, not guaranteed, depends on how warm they are). The vast majority of PIR sensors use a larger infrared receiver, a processing chip (sometimes integrated into the receiver) and a lens. However, this makes them kinda chunky. Also most have a digital, or maybe analog signal output - which makes them very simple.




The **Adafruit STHS34PF80 IR Presence / Motion Sensor** is like a deluxe version of the classic PIR sensor in that it has an IR sensing element that works up to 4 meters away and doesn't require a lens, which makes it easier to integrate into small enclosures. It also comes with a full I2C interface so you can get detailed information about how much IR was detected, and whether there's presence, movement, or temperature changes.



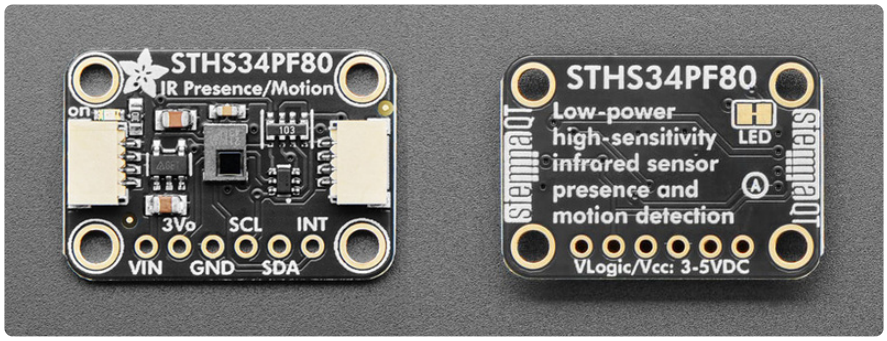
Nice sensor right? So we made it easy for you to get right into your next project. The surface-mount sensor is soldered onto a PCB and comes with a 3.3V regulator and level shifting so you can use it with a 3V or 5V logic microcontroller or single board computer without worry.




The custom made PCB is in the **STEMMA QT** form factor (https://adafru.it/LBQ), making them easy to interface with. The STEMMA QT connectors (https://adafru.it/JqB) on either side are compatible with the SparkFun Qwiic (https://adafru.it/Fpw) I2C connectors. This allows you to make solderless connections between your development board and the STHS34PF80 or to chain it with a wide range of other sensors and accessories using a **compatible cable** (https://adafru.it/JnB). QT Cable is not included, but we have a variety in the shop (https://adafru.it/17VE)

---

# Pinouts



The default I2C address is **0x5A**.

## Power Pins

- **VIN** - this is the power pin. It can be powered by 3V or 5V. Give it the same power as the logic level of your microcontroller - e.g. for a 5V micro 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** - common ground for power and logic.

## I2C Logic Pins

- **SCL** - I2C clock pin, connect to your microcontroller's I2C clock line. This pin 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 can use 3-5V logic, and 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/Ft6).

## Interrupt Pin

- **INT** - This is the interrupt pin. Its polarity and output type can be set in software over I2C.

## 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 a green LED.
- **LED jumper** - This jumper is located on the back of the board and is labeled **LED** on the board silk. Cut the trace on this jumper to cut power to the "**on**" LED.

---

# CircuitPython and Python

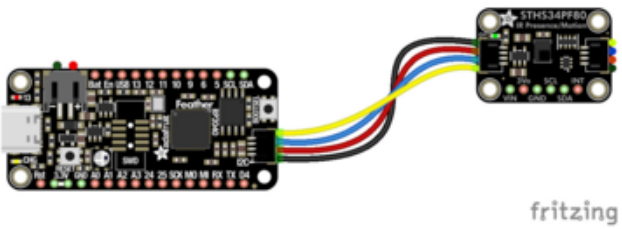
It's easy to use the **STHS34PF80 breakout** with Python or CircuitPython, and the Adafruit_CircuitPython_STHS34PF80 (https://adafru.it/1auv) module. This module

allows you to easily write Python code to read ambient and object temperatures with the sensor.

You can use this driver 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).

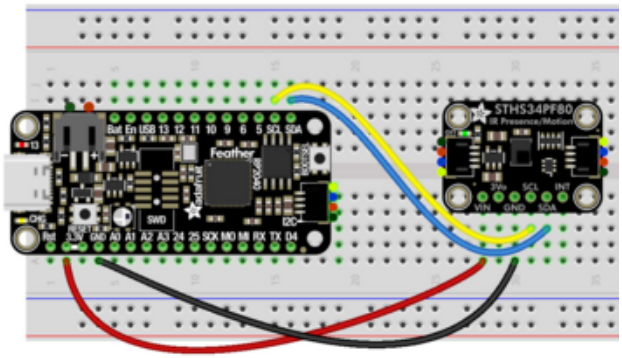
## CircuitPython Microcontroller Wiring

First wire up the breakout to your board exactly as follows. The following is the sensor wired to a Feather RP2040 using the STEMMA connector:




**Board STEMMA 3V** to **breakout STEMMA VIN (red wire)**
**Board STEMMA GND** to **breakout STEMMA GND (black wire)**
**Board STEMMA SCL** to **breakout STEMMA SCL (yellow wire)**
**Board STEMMA SDA** to **breakout STEMMA SDA (blue wire)**

The following is the breakout wired to a Feather RP2040 using a solderless breadboard:

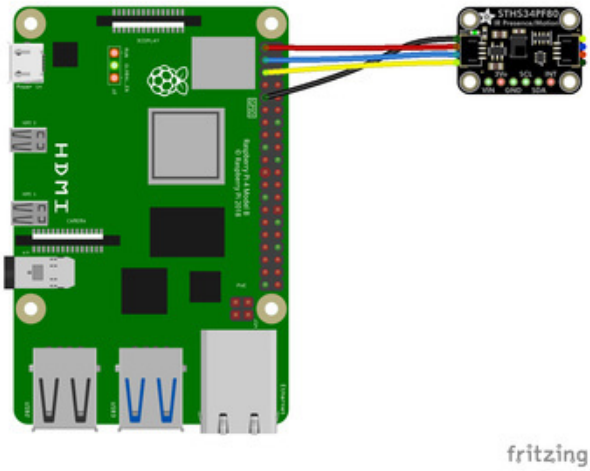


**Board 3V** to **breakout VIN (red wire)**
**Board GND** to **breakout GND (black wire)**
**Board SCL** to **breakout SCL (yellow wire)**
**Board SDA** to **breakout SDA (blue wire)**

## Python Computer Wiring

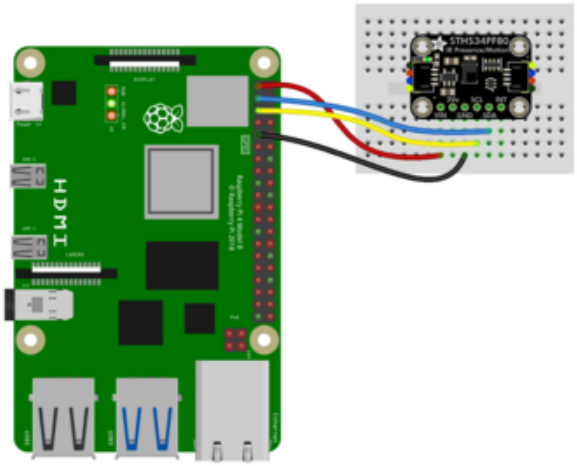
Since there are dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, [please visit the guide for CircuitPython on Linux to see whether your platform is supported](https://adafru.it/BSN) (https://adafru.it/BSN).

Here's the Raspberry Pi wired with I2C using the STEMMA connector:




**Pi 3V** to **breakout STEMMA VIN (red wire)**
**Pi GND** to **breakout STEMMA GND (black wire)**
**Pi SCL** to **breakout STEMMA SCL (yellow wire)**
**Pi SDA** to **breakout STEMMA SDA (blue wire)**

Here's the Raspberry Pi wired with I2C using a solderless breadboard:



**Pi 3V** to **breakout VIN (red wire)**
**Pi GND** to **breakout GND (black wire)**
**Pi SCL** to **breakout SCL (yellow wire)**
**Pi SDA** to **breakout SDA (blue wire)**

## Python Installation of STHS34PF80 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-sths34pf80`

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 Usage

To use with CircuitPython, you need to first install the **Adafruit_CircuitPython_STHS34PF80** 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 folders and file:

- **adafruit_bus_device/**
- **adafruit_register/**
- **adafruit_sths34pf80.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`

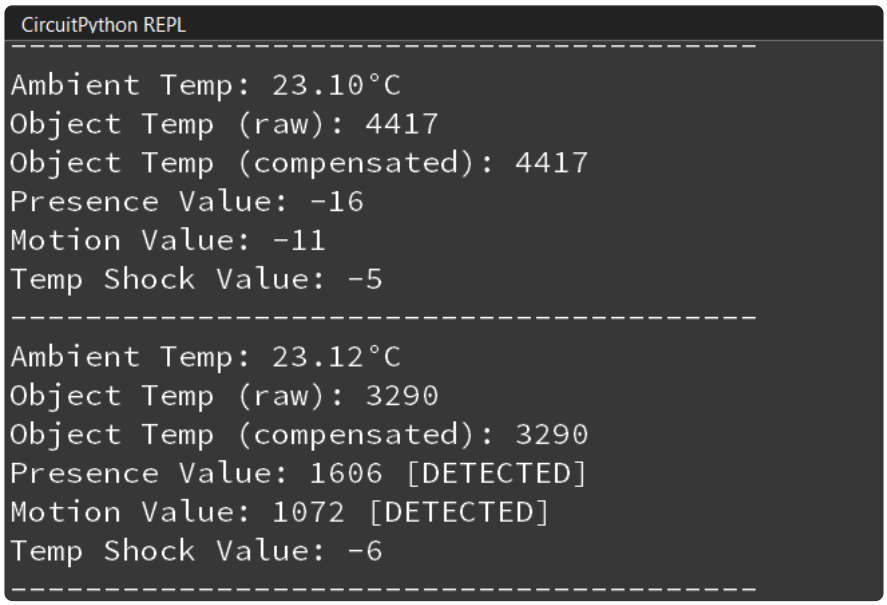
## Example Code

**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.

```
# SPDX-FileCopyrightText: Copyright (c) 2025 Liz Clark for Adafruit Industries
#
# SPDX-License-Identifier: MIT

"""
Simple test example for the STHS34PF80 IR presence and motion sensor
"""

import time

import board

import adafruit_sths34pf80

# Create I2C bus
i2c = board.I2C()

# Create sensor instance
sensor = adafruit_sths34pf80.STHS34PF80(i2c)

print("-" * 40)

while True:
    # wait for new data
    if sensor.data_ready:
        # temperature values
        ambient_temp = sensor.ambient_temperature
        object_temp = sensor.object_temperature
        comp_object_temp = sensor.compensated_object_temperature

        # detection values
        presence_val = sensor.presence_value
        motion_val = sensor.motion_value
        temp_shock_val = sensor.temperature_shock_value

        # detection true/false
        presence = sensor.presence
        motion = sensor.motion
        temp_shock = sensor.temperature_shock

        print(f"Ambient Temp: {ambient_temp:.2f}°C")
        print(f"Object Temp (raw): {object_temp}")
        print(f"Object Temp (compensated): {comp_object_temp}")
        print(f"Presence Value: {presence_val} {'[DETECTED]' if presence else ''}")
        print(f"Motion Value: {motion_val} {'[DETECTED]' if motion else ''}")
        print(f"Temp Shock Value: {temp_shock_val} {'[DETECTED]' if temp_shock else
''}")
        print("-" * 40)
```

First, the STHS34PF80 sensor is recognized over I2C. Then, in the loop, it prints out the ambient temperature, presence detection and motion detection data.




# Python Docs

Python Docs (https://adafru.it/1aus)

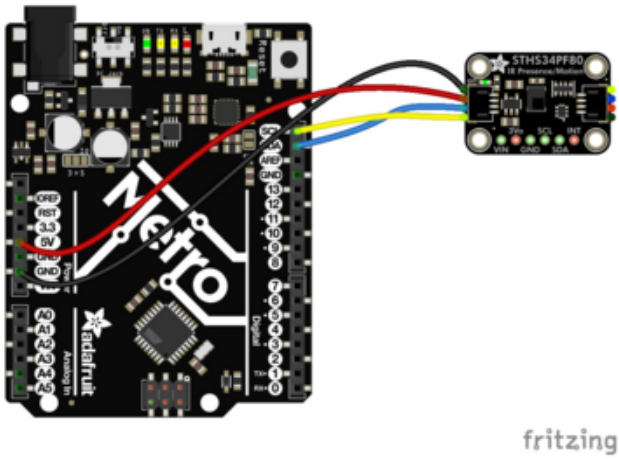
# Arduino

Using the STHS34PF80 breakout with Arduino involves wiring up the breakout to your Arduino-compatible microcontroller, installing the Adafruit_STHS34PF80 (https://adafru.it/1auw) library, and running the provided example code.

## Wiring

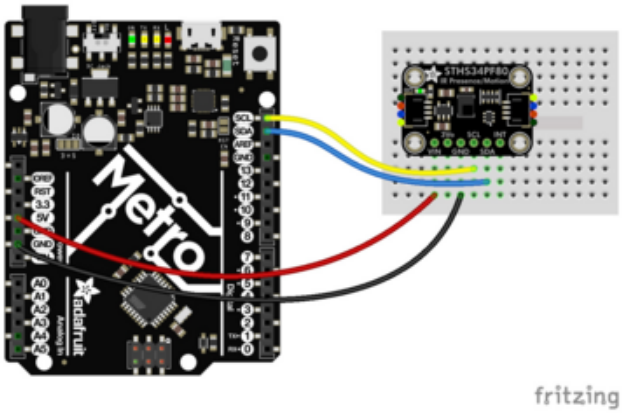
Wire as shown for a **5V** board like an Uno. If you are using a **3V** board, like an Adafruit Feather, wire the board's 3V pin to the sensor VIN.

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




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

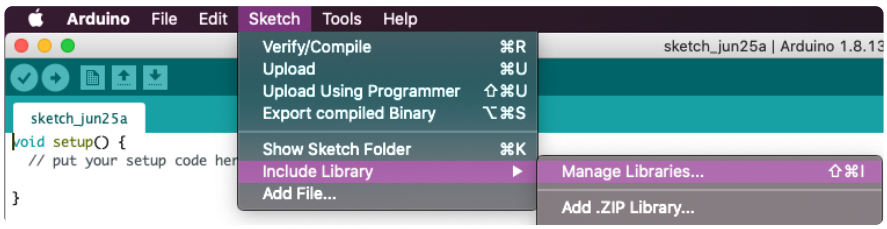
Here is an Adafruit Metro wired up using a solderless breadboard:



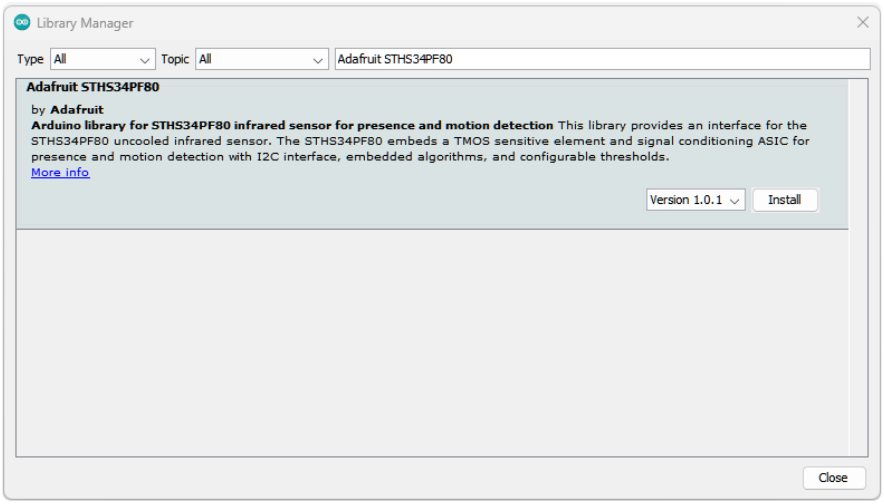

**Board 5V** to **breakout VIN (red wire)**
**Board GND** to **breakout GND (black wire)**
**Board SCL** to **breakout SCL (yellow wire)**
**Board SDA** to **breakout SDA (blue wire)**

## Library Installation

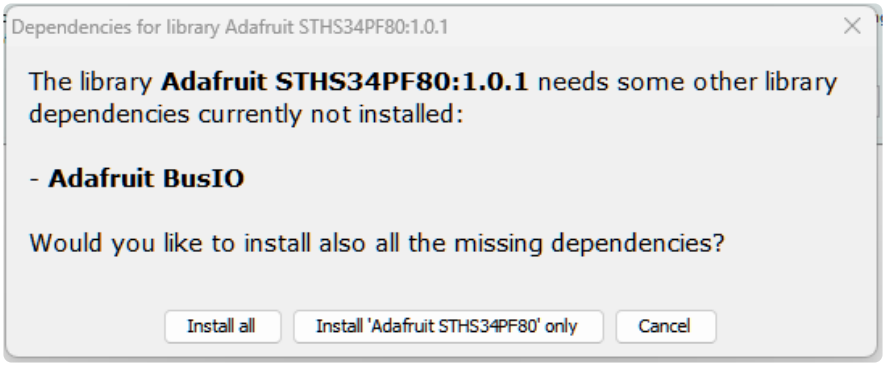
You can install the **Adafruit_STHS34PF80** library for Arduino using the Library Manager in the Arduino IDE.




Click the **Manage Libraries ...** menu item, search for **Adafruit STHS34PF80**, and select the **Adafruit STHS34PF80** library:




If asked about dependencies, click "Install all".




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

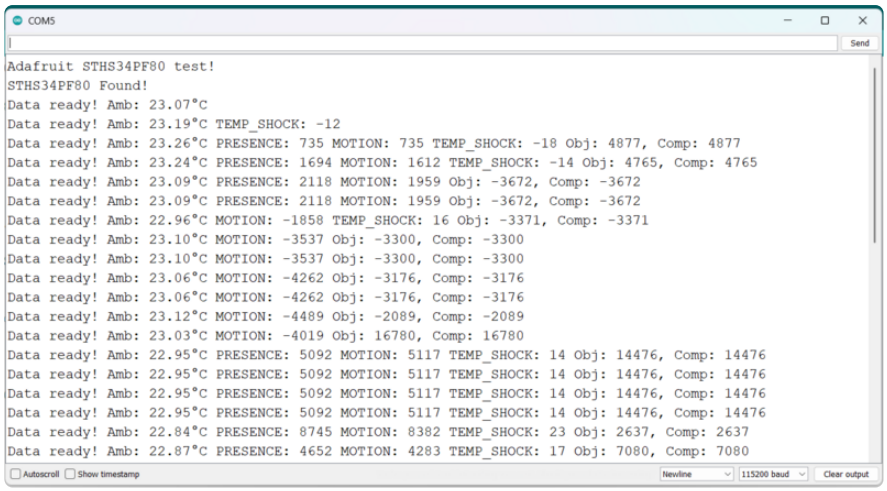
If the dependencies are already installed, you must make sure you update them through the Arduino Library Manager before loading the example!

## Example Code

```
// Basic test for STHS34PF80 infrared sensor

#include "Adafruit_STHS34PF80.h"

Adafruit_STHS34PF80 sths;
```

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

  Serial.println("Adafruit STHS34PF80 test!");

  if (!sths.begin()) {
    Serial.println("Could not find a valid STHS34PF80 sensor, check wiring!");
    while (1) delay(10);
  }

  Serial.println("STHS34PF80 Found!");
}

void loop() {
  if (!sths.isDataReady()) {
    delay(10);
    return;
  }

  Serial.print("Data ready! ");

  // Read temperature data
  Serial.print("Amb: ");
  Serial.print(sths.readAmbientTemperature(), 2);
  Serial.print("°C");

  // Check for presence and show data if detected
  if (sths.isPresence()) {
    Serial.print(" PRESENCE: ");
    Serial.print(sths.readPresence());
  }

  // Check for motion and show data if detected
  if (sths.isMotion()) {
    Serial.print(" MOTION: ");
    Serial.print(sths.readMotion());
  }

  // Check for temperature shock and show data if detected
  if (sths.isTempShock()) {
    Serial.print(" TEMP_SHOCK: ");
    Serial.print(sths.readTempShock());
  }

  // Show object temperatures only if presence or motion detected
  if (sths.isPresence() || sths.isMotion()) {
    Serial.print(" Obj: ");
    Serial.print(sths.readObjectTemperature());
    Serial.print(", Comp: ");
    Serial.print(sths.readCompensatedObjectTemperature());
  }

  Serial.println();
  delay(100);
}
```

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at 115200 baud. You'll see the STHS34PF80 recognized over I2C. Then, the ambient temperature, presence detection and motion detection are printed to the Serial Monitor.




# Arduino Docs

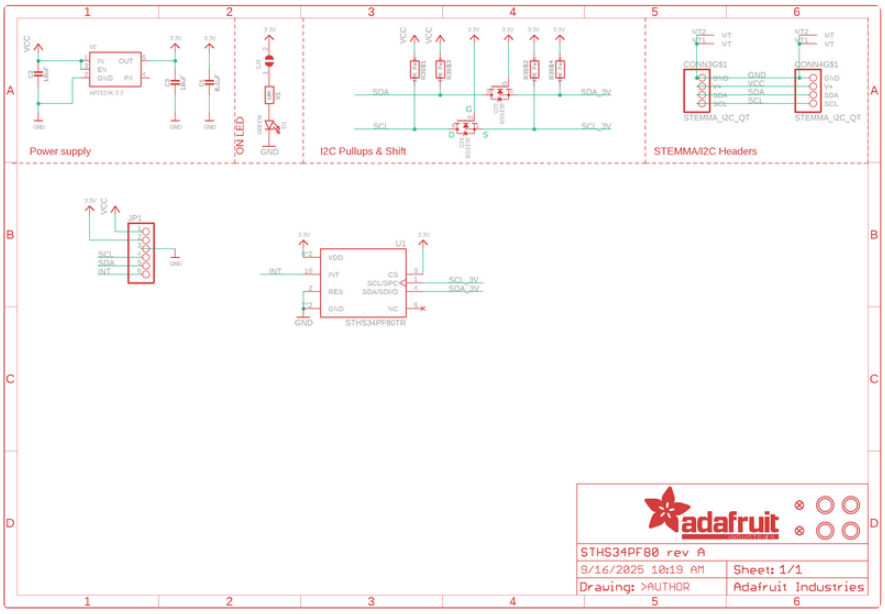
Arduino Docs (https://adafru.it/1aut)

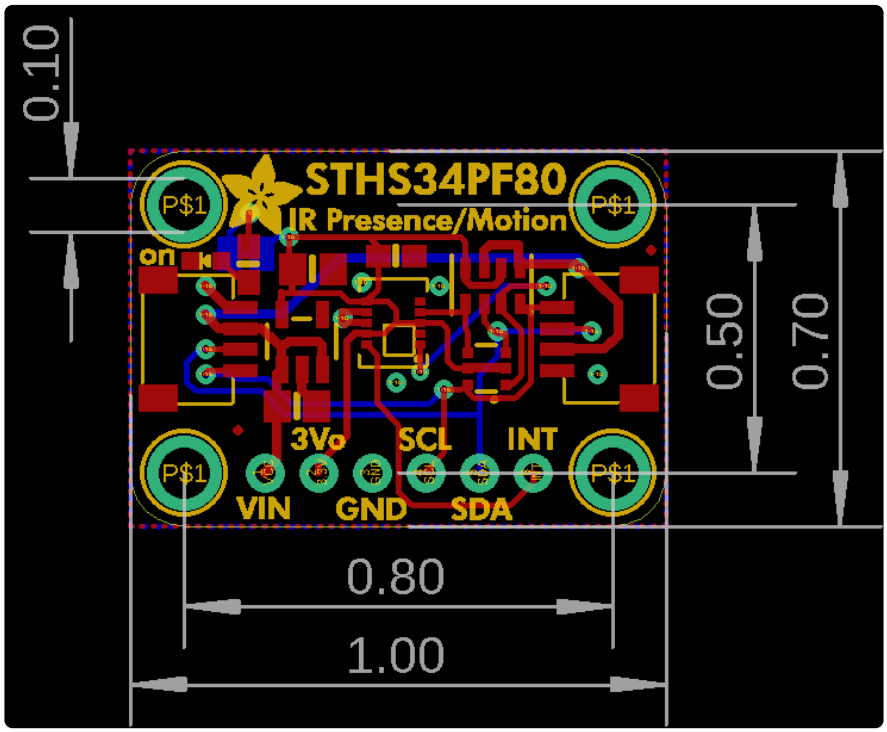
# Downloads

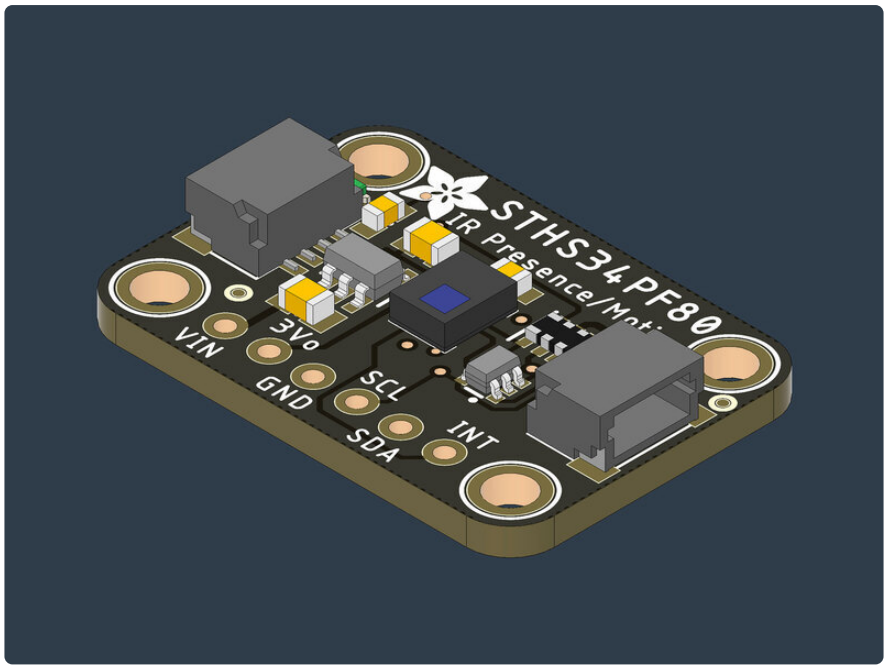
## Files

- STHS34PF80 Datasheet (http://adafru.it/64263480)
- EagleCAD PCB files on GitHub (https://adafru.it/1aux)
- 3D models on GitHub (https://adafru.it/1auy)
- Fritzing object in the Adafruit Fritzing Library (https://adafru.it/1auz)

# Schematic and Fab Print




STHS34PF80
VIN
3Vo
GND
SCL
SDA
INT