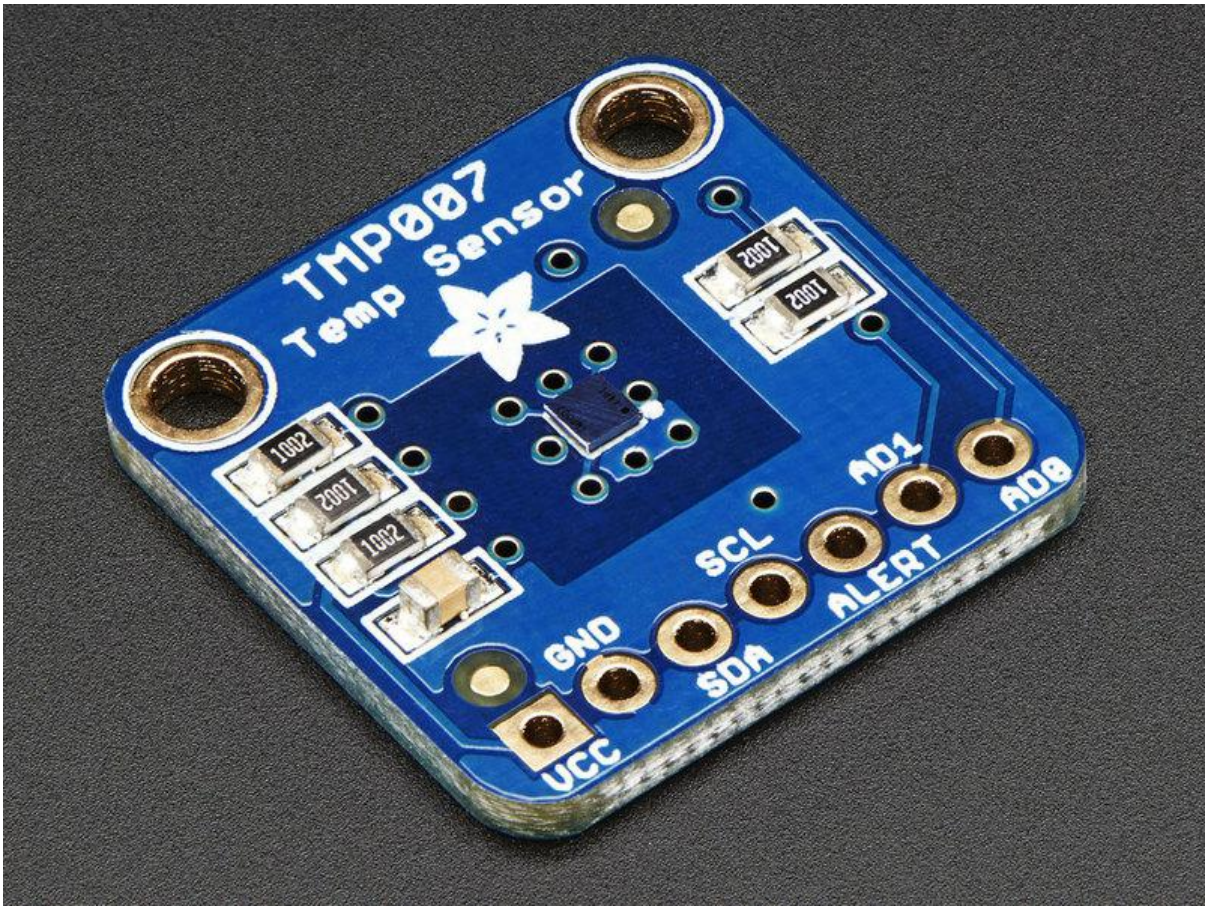
Adafruit TMP007 Sensor Breakout

Created by lady ada

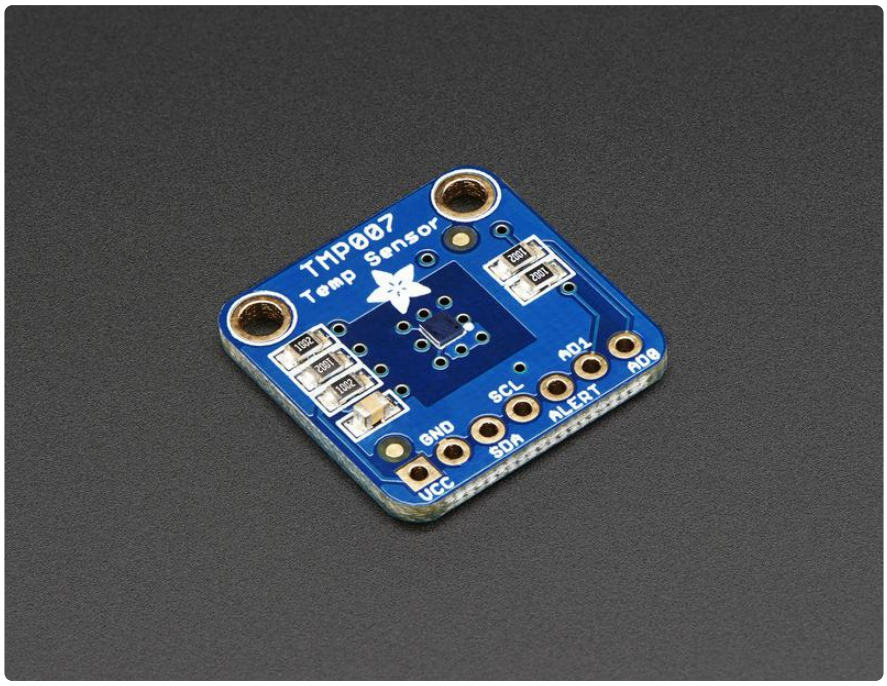


https://learn.adafruit.com/adafruit-tmp007-sensor-breakout

Last updated on 2022-12-01 02:14:47 PM EST

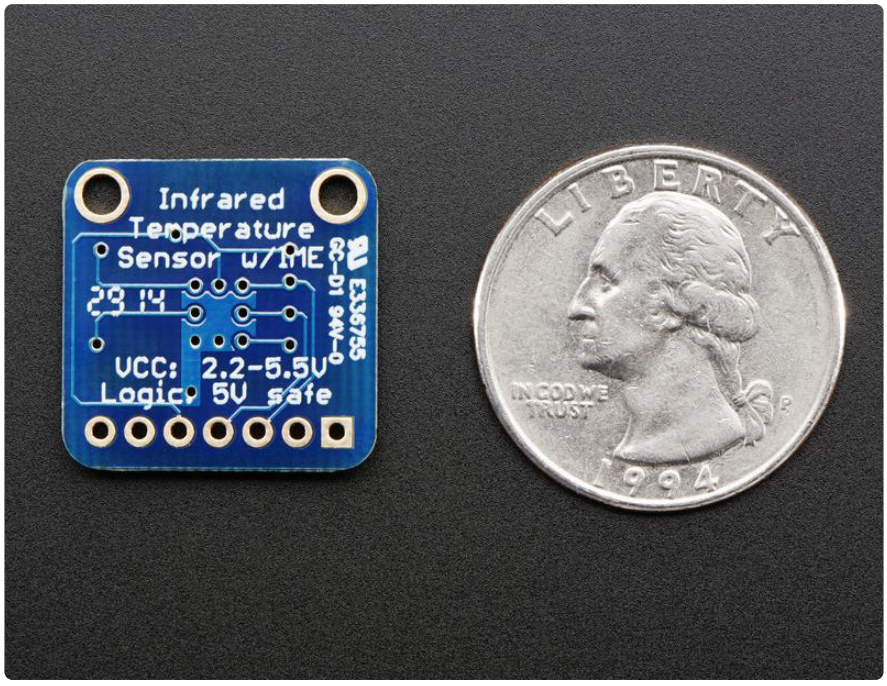
Table of Contents

Overview

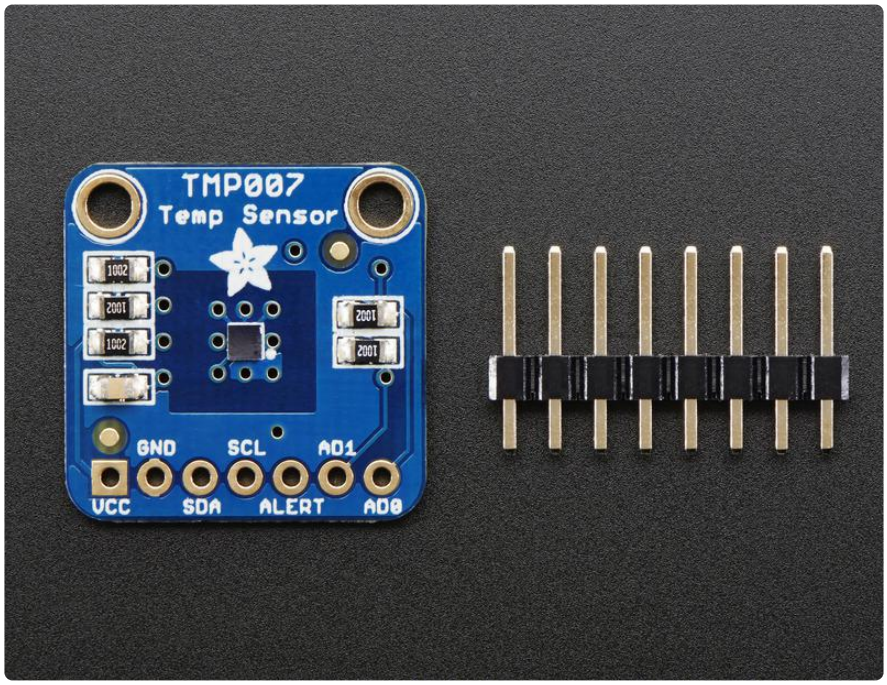


Unlike most of the other temperature sensors we have, this breakout has a really cool IR sensor from TI that can measure the temperature of an object without touching it.

The TMP007 is the latest thermopile sensor from TI, and is an update of the TMP006 ( ). The internal math engine does all the temperature calculations so its easier to integrate - you can read the die and target temperatures directly over I2C. The TMP007 also has better transient management, so you don't get as much over/ undershoot when the temperature changes a lot.

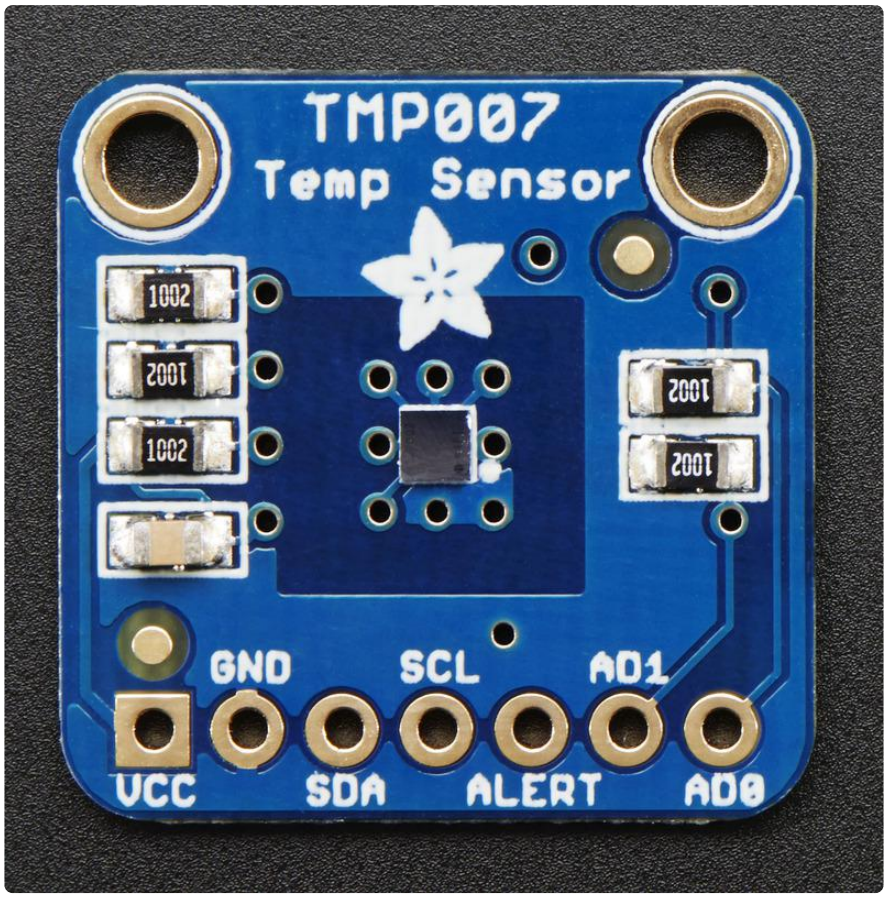


Simply point the sensor towards what you want to measure and it will detect the temperature by absorbing IR waves emitted. The embedded thermopile sensor generates a very very small voltage depending on how much IR there is, and using some math, that micro voltage can be used to calculate the temperature. It also takes the measurement over an area so it can be handy for determining the average temperature of something.



This sensor comes as a ultra-small 0.5mm pitch BGA, too hard to solder by hand. So we stuck it on an easy-to-work-with breakout board. The sensor works with 2.5V to 5V logic so it requires no logic level shifting. There are two address pins and using a funky method of connecting the pins you can have up to 8 TMP007's connected to one i2c bus. We also include a small piece of 0.1" breakaway header so you can easily solder to and use this sensor on a breadboard. Two mounting holes make it easy to attach to an enclosure.

Pinouts



The TMP007 is a very straight-forward sensor, lets go thru all the pins so you can understand what you need to connect to get started

Power Pins

- VCC - This is the positive power and logic level pin. It can be 2.2-5.5VDC, so fine for use with 3 or 5V logic. Power VCC with whatever logic level you plan to use on the i2c lines.
- GND - this is the ground power and logic reference pin.

I2C Data Pins

- SCL - this is the I2C clock pin. There's a 10K pull-up already on the board, so connect this directly to the i2c master clock pin on your microcontroller
- SDA - this is the I2C data pin. There's a 10K pull-up already on the board, so connect this directly to the i2c master data pin on your microcontroller

Optional Pins

These are pins you don't need to connect to unless you want to!

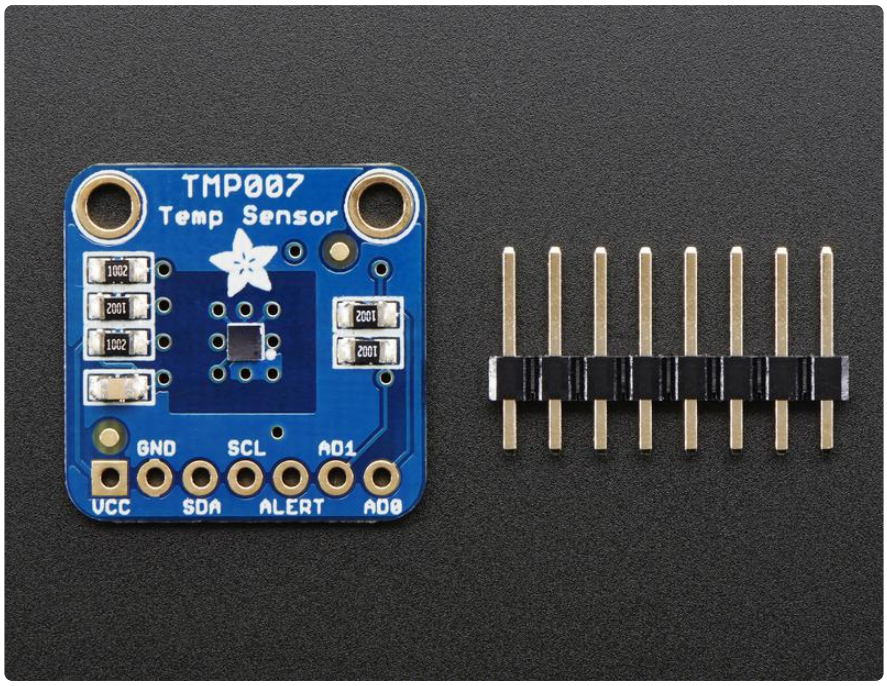
- Alert - This is the interrupt/alert pin from the TMP007. The chip has some capability to 'alert' you if the chip temperature goes above or below a set amount. This output can trigger to let you know. We don't have library support for this pin, so check the datasheet for more information.
- AD0 AD1 - These are the address select pins. Since you can only have one device with a given address on an i2c bus, there must be a way to adjust the address if you want to put more than one TMP on a shared i2c bus. The AD0/ AD1 pins set the bottom three pins of the i2c address. There are 10K pull-down resistors on the board.

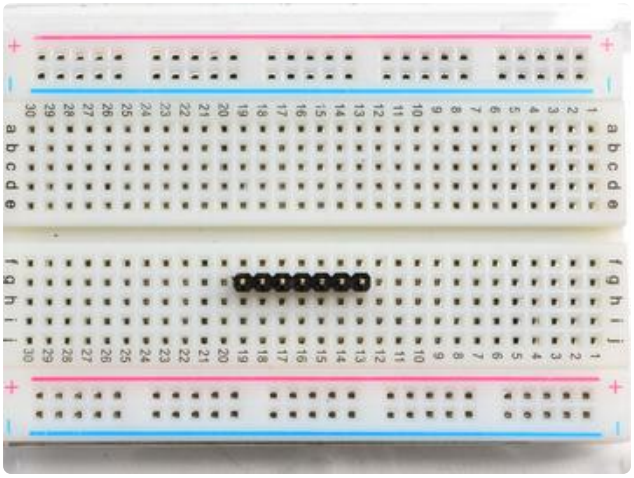
The default address of the TMP007 is 0x40. By connecting the address pins as in the following table, you can generate any address between 0x40 and 0x47

Table 2. Address Pins and Slave Addresses

ADR1	ADR0	SMBus ADDRESSES
0	0	1000000
0	1	1000001
0	SDA	1000010
0	SCL	1000011
1	0	1000100
1	1	1000101
1	SDA	1000110
1	SCL	1000111

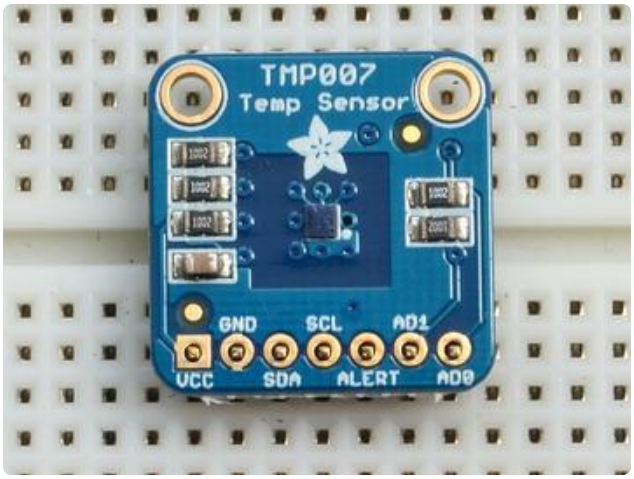
Assembly





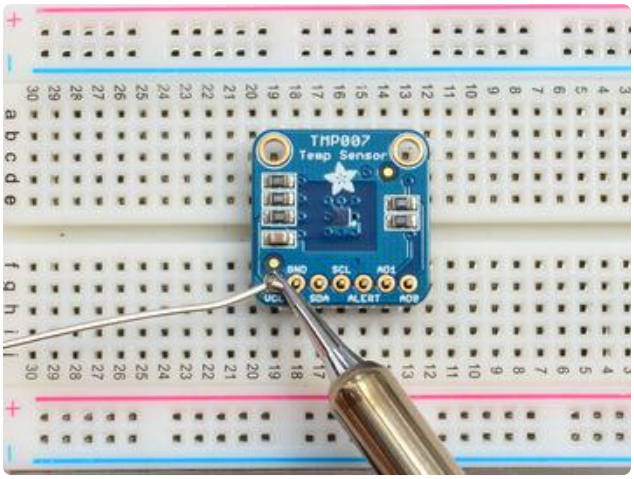
Prepare the header strip:

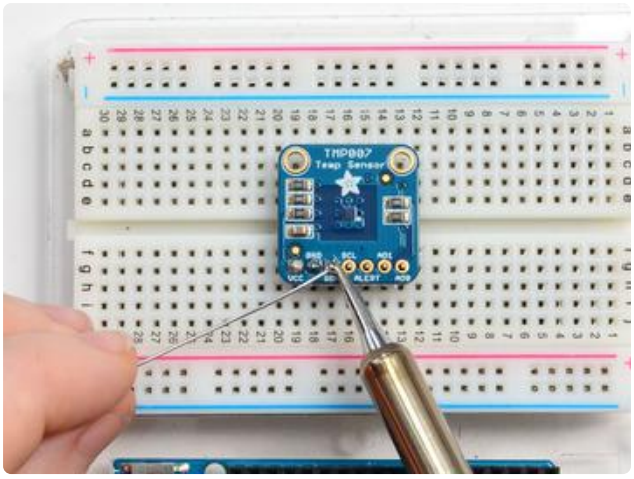
Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - long pins down

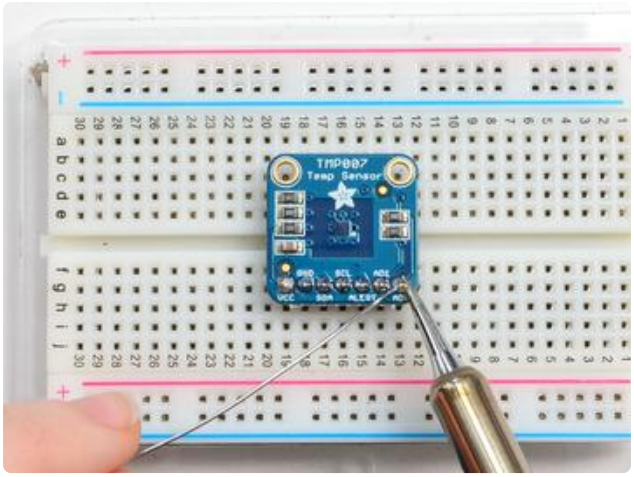


Add the breakout board:

Place the breakout board over the pins so that the short pins poke through the breakout pads







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 ()).

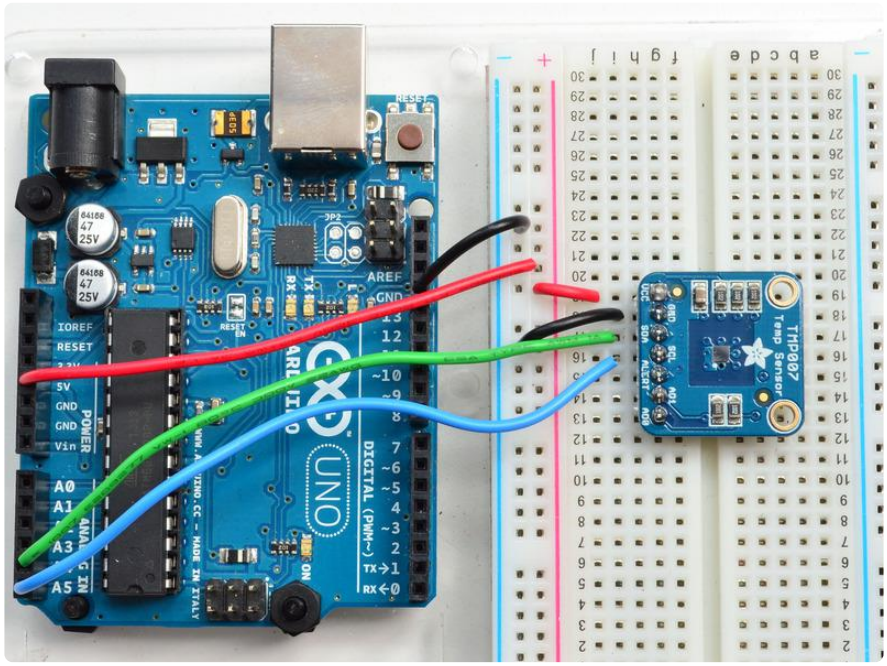
You're done! Check your solder joints visually and continue onto the next steps

---

Arduino

Arduino Wiring

You can easily wire this sensor to any microcontroller, we'll be using an Arduino



- Connect Vdd to the power supply, 3V or 5V is fine. Use the same voltage that the microcontroller logic is based off of. For most Arduinos, that is 5V
- Connect GND to common power/data ground
- Connect the SCL pin to the I2C clock SCL pin on your Arduino. On an UNO & '328 based Arduino, this is also known as A5, on a Mega it is also known as digital 21 and on a Leonardo/Micro, digital 3

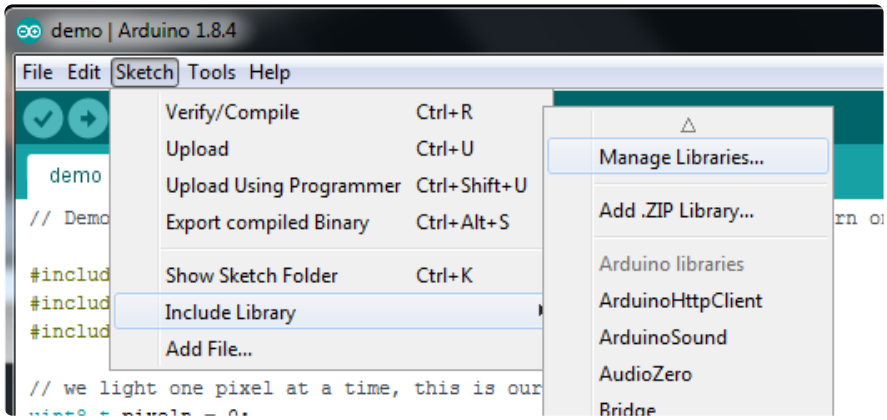
- Connect the SDA pin to the I2C data SDA pin on your Arduino. On an UNO & '328 based Arduino, this is also known as A4, on a Mega it is also known as digital 20 and on a Leonardo/Micro, digital 2

The TMP007 has a default I2C address of 0x40 but you can set the address to any of 8 values between 0x40 and 0x47 so you can have up to 8 of these sensors all sharing the same SCL/SDA pins.

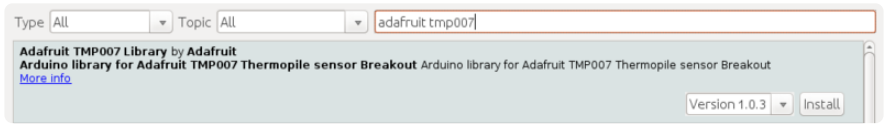
Download Adafruit_TMP007

To begin reading sensor data, you will need to download the Adafruit TMP007 library from the Arduino library manager.

Open up the Arduino library manager:



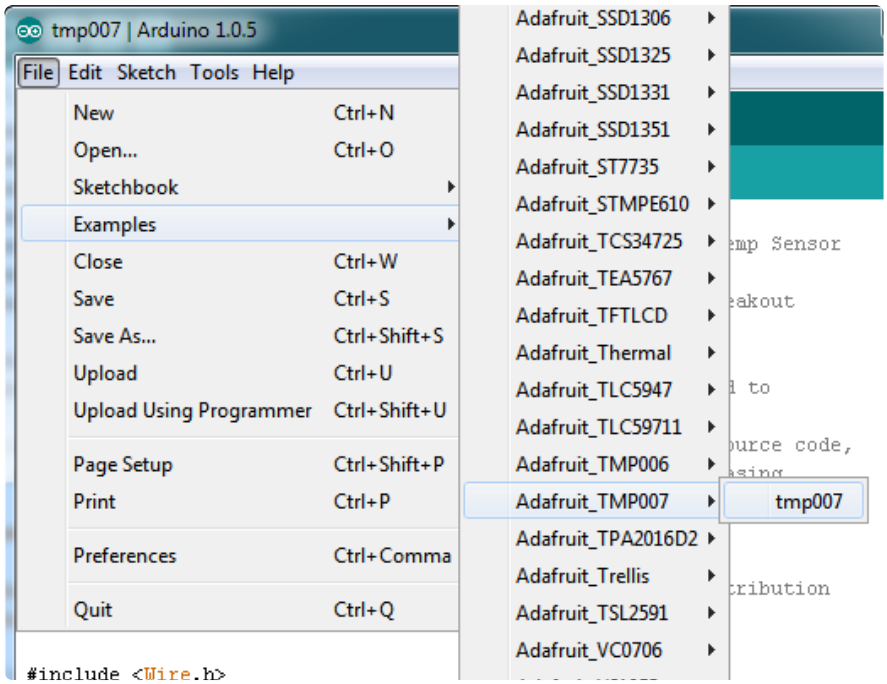
Search for the Adafruit TMP007 library and install it



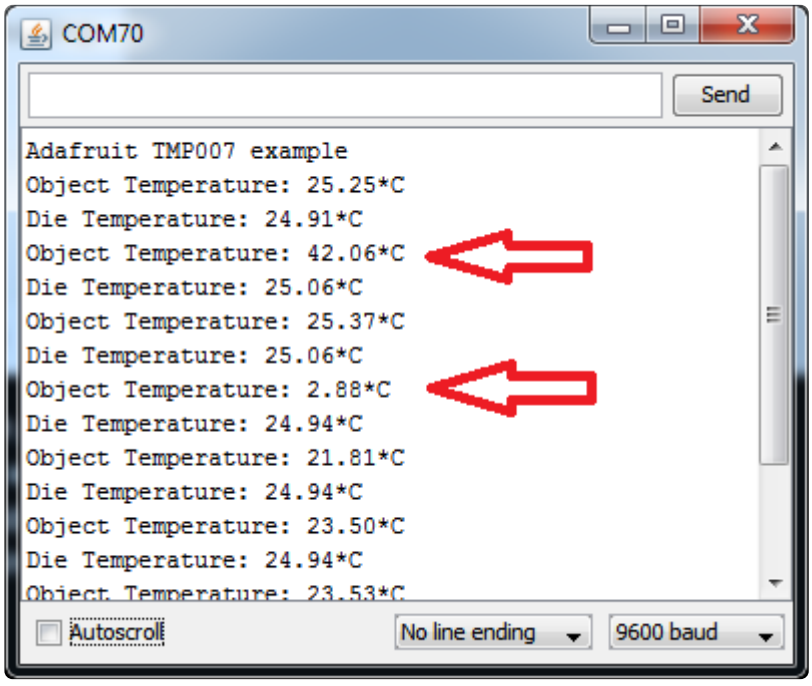
We also have a great tutorial on Arduino library installation at:
http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use ()

Load Demo

Open up File->Examples->Adafruit_TMP007->tmp007 and upload to your Arduino wired up to the sensor



Thats it! Now open up the serial terminal window at 9600 speed to see the temperature in real time. You can try putting your hand or a cold glass of water over the sensor (not touching) to see the thermopile sensor adjust!



Library Reference

The TMP007 library is pretty straight forward! Start by creating the Adafruit_TMP007 object with:

```
Adafruit_TMP007 tmp007;
```

Or with an i2c address assigned

```
Adafruit_TMP007 tmp007(0x41);  // start with a diferent i2c address!
```

Then you can initialize & configure

```
tmp007.begin()
```

or, to set the samples/reading with:

```
tmp007.begin(TMP007_CFG_1SAMPLE)
```

We suggest the default, 16 samples for best accuracy. begin() will return true or false based on whether it found the sensor, check it using an if statment like so:

```
  if (! tmp007.begin()) {
    Serial.println("No sensor found");
    while (1);
  }
```

Now you can read the Die temperature (the temperature of the physical sensor itself) and Object temperature (the temperature of the stuff in front of the sensor)

```
tmp007.readObjTempC();
tmp007.readDieTempC();
```

The readings are floating point values, in degrees C.

Then wait 4 seconds between readings to get a new reading!

```
delay(4000); // 4 seconds per reading for 16 samples per reading
```

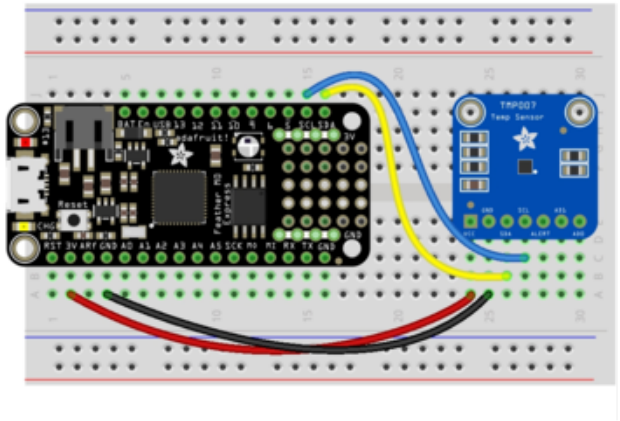
Python & CircuitPython

It's easy to use the TMP007 sensor with Python or CircuitPython and the Adafruit CircuitPython TMP007 () module. This module allows you to easily write Python code that reads the temperature 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 TMP007 to your board exactly as shown in the previous pages for Arduino. Here's an example of wiring a Feather M0 to the sensor with I2C:

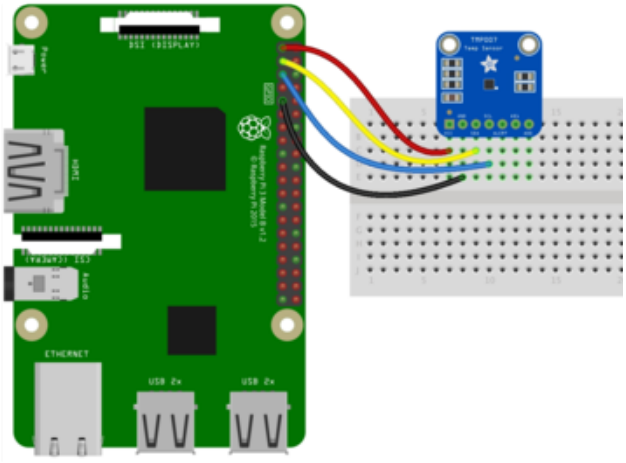


Board 3V to sensor VCC
Board GND to sensor G
Board SCL to sensor SCL
Board SDA to sensor SDA

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 SCK
Pi SDA to sensor SDA

CircuitPython Installation of TMP007 Library

You'll need to install the Adafruit CircuitPython TMP007 () 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 Welcome to CircuitPython 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_tmp007.mpy
- adafruit_bus_device

Before continuing make sure your board's lib folder or root filesystem has the adafruit_tmp007.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 TMP007 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-tmp007`

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 die temperature and object temperature from the board's Python REPL.

First, run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```
import board
import busio
import adafruit_tmp007
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_tmp007.TMP007(i2c)
```

Now you're ready to setup the sensor and read the values using these properties:

- die_temperature - reads sensor die temperature and return its value in degrees Celsius.
- temperature - reads the object temperature and returns it's value in degrees Celcius.

For example, to print the die temperature and object temperature:

```
print('Die temperature: {0:0.3F}*C'.format(sensor.die_temperature))
print('Object temperature: {0:0.3F}*C'.format(sensor.temperature))
```

```
>>> print('Die temperature: {0:0.3F}*C'.format(sensor.die_temperature))
Die temperature: 23.375*C
>>> print('Object temperature: {0:0.3F}*C'.format(sensor.temperature))
Object temperature: 28.313*C
```

That's all there is to using the TMP007 with CircuitPython!

Full Example Code

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

#!/usr/bin/python
# Author: Adapted to CircuitPython by Jerry Needell
#     Adafruit_Python_TMP example by Tony DiCola
#

import time
import board
import busio
```

```
import adafruit_tmp007


# Define a function to convert celsius to fahrenheit.
def c_to_f(c):
    return c * 9.0 / 5.0 + 32.0


# Create library object using our Bus I2C port
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_tmp007.TMP007(i2c)


# Initialize communication with the sensor, using the default 16 samples per
conversion.
# This is the best accuracy but a little slower at reacting to changes.
# The first sample will be meaningless
while True:
    die_temp = sensor.die_temperature
    print(
        "   Die temperature: {0:0.3F}*C / {1:0.3F}*F".format(die_temp,
c_to_f(die_temp))
    )
    obj_temp = sensor.temperature
    print(
        "Object temperature: {0:0.3F}*C / {1:0.3F}*F".format(obj_temp,
c_to_f(obj_temp))
    )
    time.sleep(5.0)
```

Python Docs

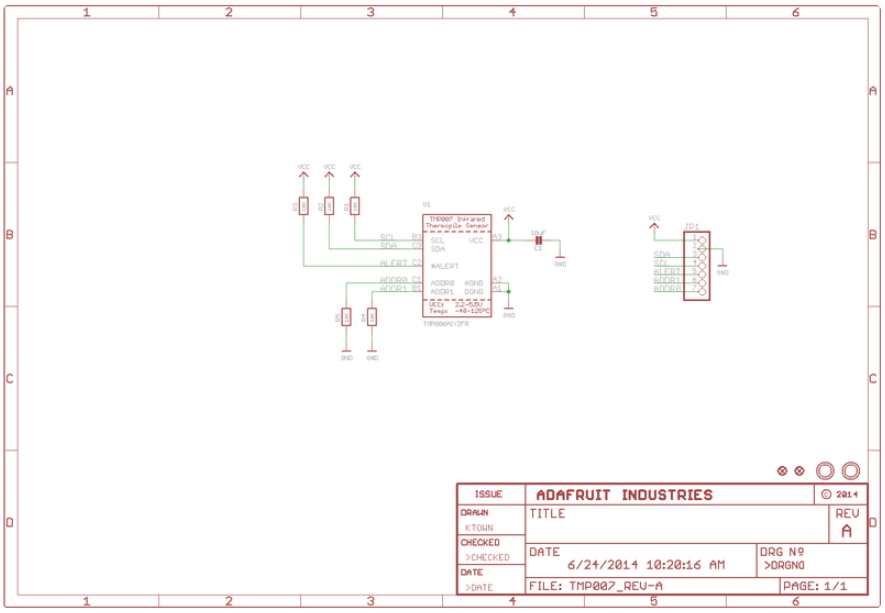
Python Docs ()

Downloads

Datasheets & Files

- TMP007 datasheet ()
- TMP007 product page at TI ()
- Fritzing object in Adafruit Fritzing library ()
- EagleCAD PCB files on GitHub ()

Schematic



PCB Fabrication Print

Dimensions in Inches