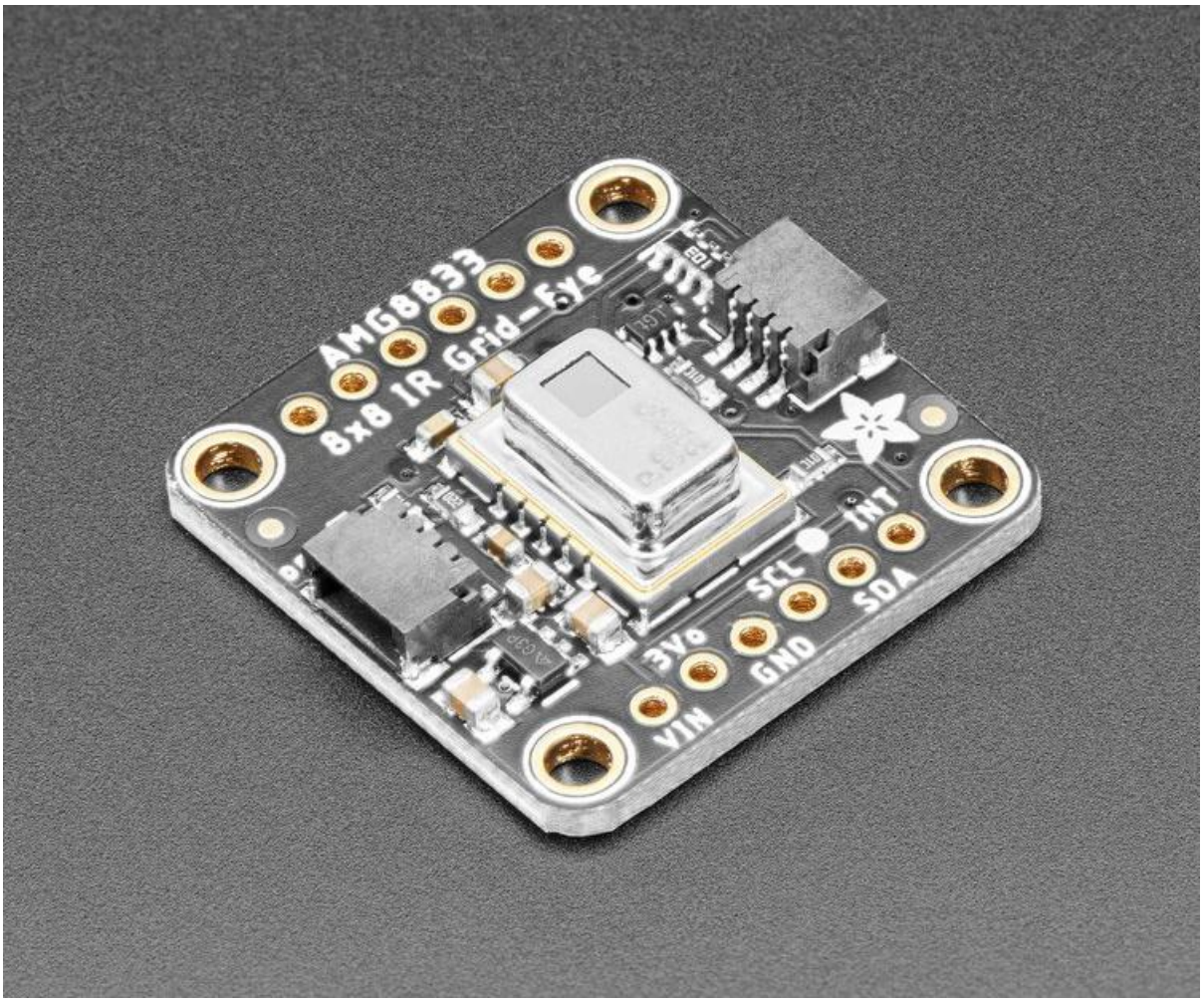
Adafruit AMG8833 8x8 Thermal Camera Sensor

Created by Abigail Torres

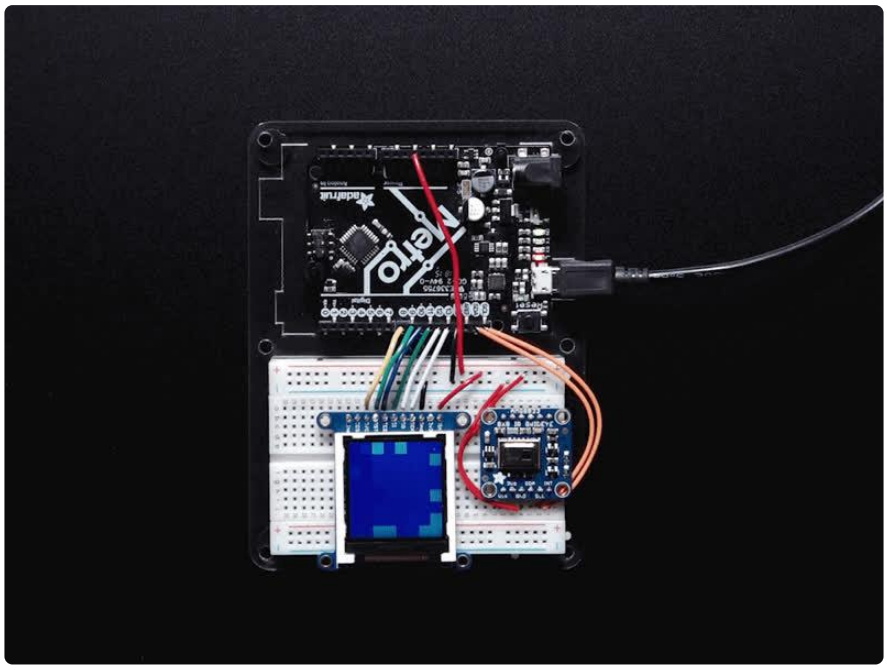


https://learn.adafruit.com/adafruit-amg8833-8x8-thermal-camera-sensor

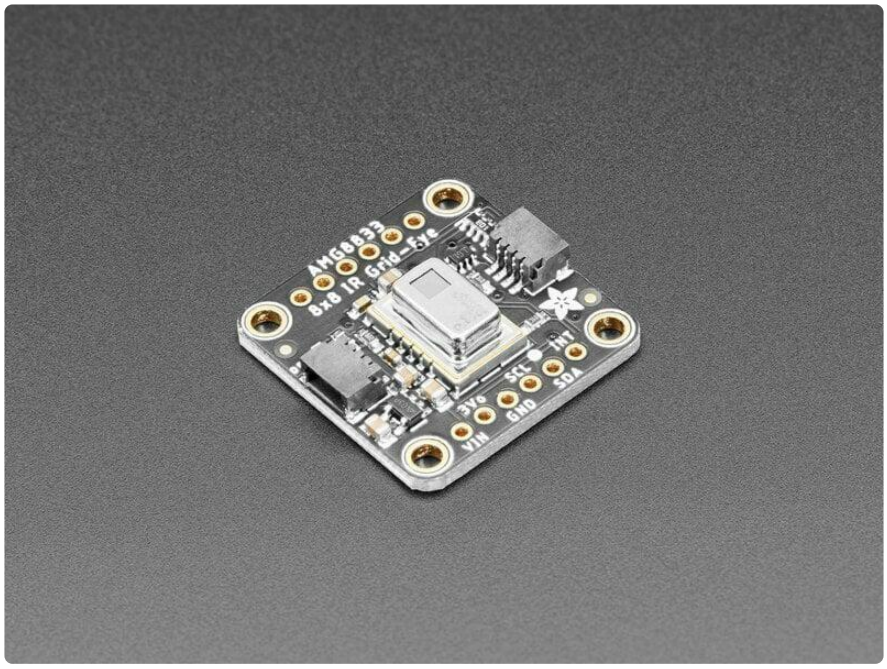
Last updated on 2022-12-01 02:57:31 PM EST

Table of Contents

Overview

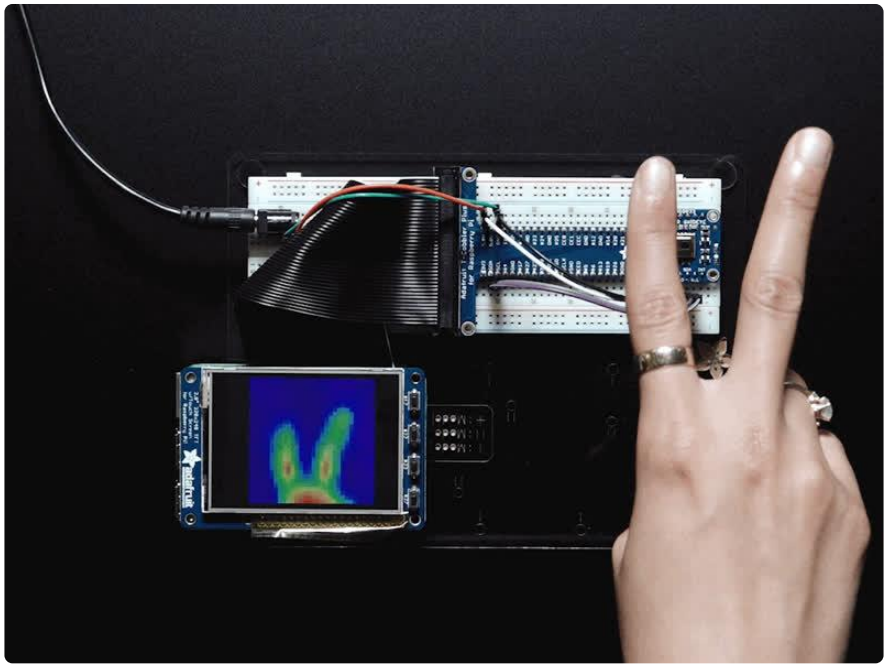


Add heat-vision to your project and with an Adafruit AMG8833 Grid-EYE Breakout! This sensor from Panasonic is an 8x8 array of IR thermal sensors. When connected to your microcontroller (or raspberry Pi) it will return an array of 64 individual infrared temperature readings over I2C. It's like those fancy thermal cameras, but compact and simple enough for easy integration.



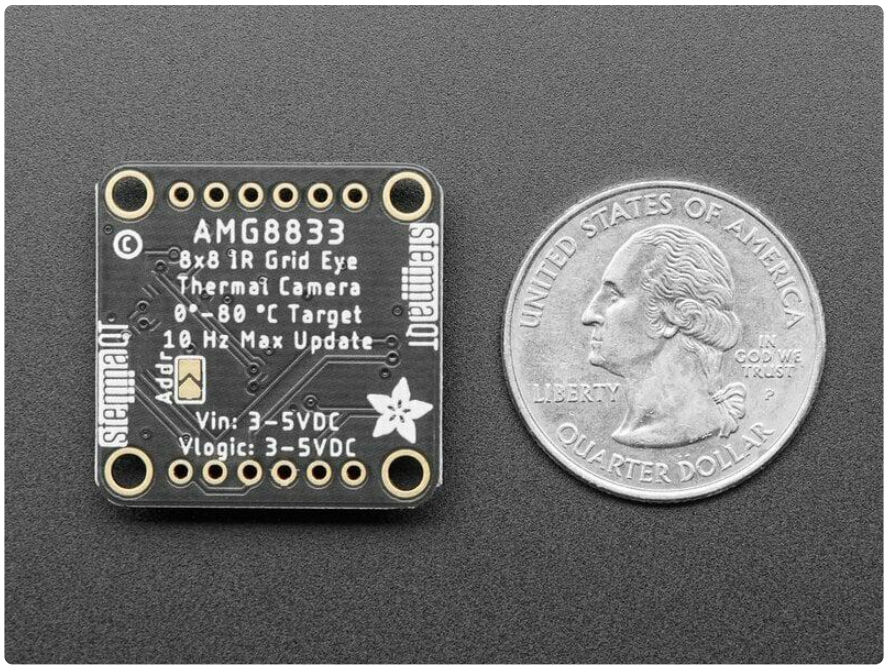
This part will measure temperatures ranging from 0°C to 80°C (32°F to 176°F) with an accuracy of +- 2.5°C (4.5°F). It can detect a human from a distance of up to 7 meters

(23) feet. With a maximum frame rate of 10Hz, It's perfect for creating your own human detector or mini thermal camera. We have code for using this breakout on an Arduino or compatible (the sensor communicates over I2C) or on a Raspberry Pi with Python. On the Pi, with a bit of image processing help from the SciPy python library we were able to interpolate the 8x8 grid and get some pretty nice results!

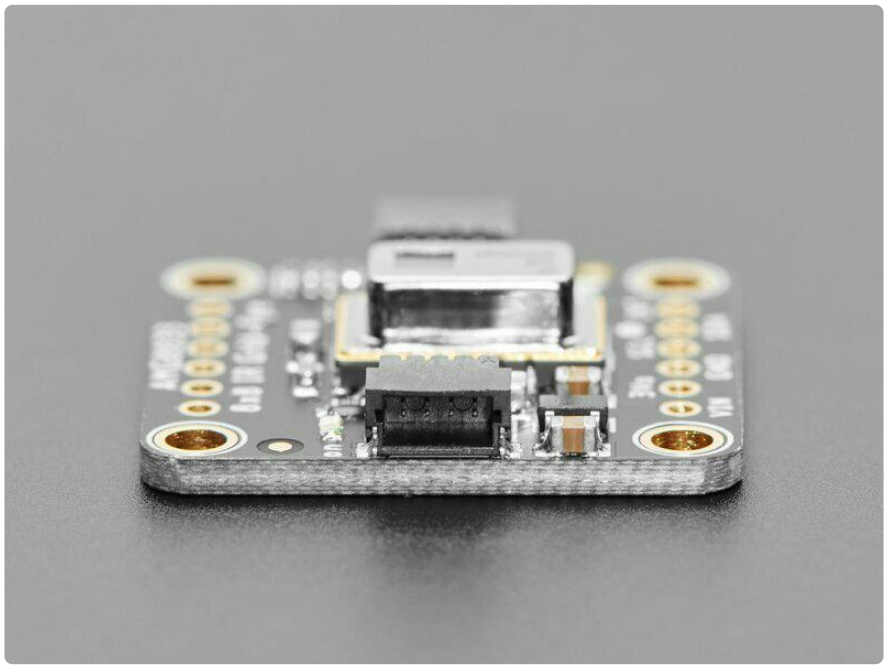


The AMG8833 is the next generation of 8x8 thermal IR sensors from Panasonic, and offers higher performance than it's predecessor the AMG8831. The sensor only supports I2C, and has a configurable interrupt pin that can fire when any individual pixel goes above or below a thresholds that you set.

To make it easy to use, we pick & placed it on a breakout board with a 3.3V regulator and level shifting. So you can use it with any 3V or 5V microcontroller or computer.

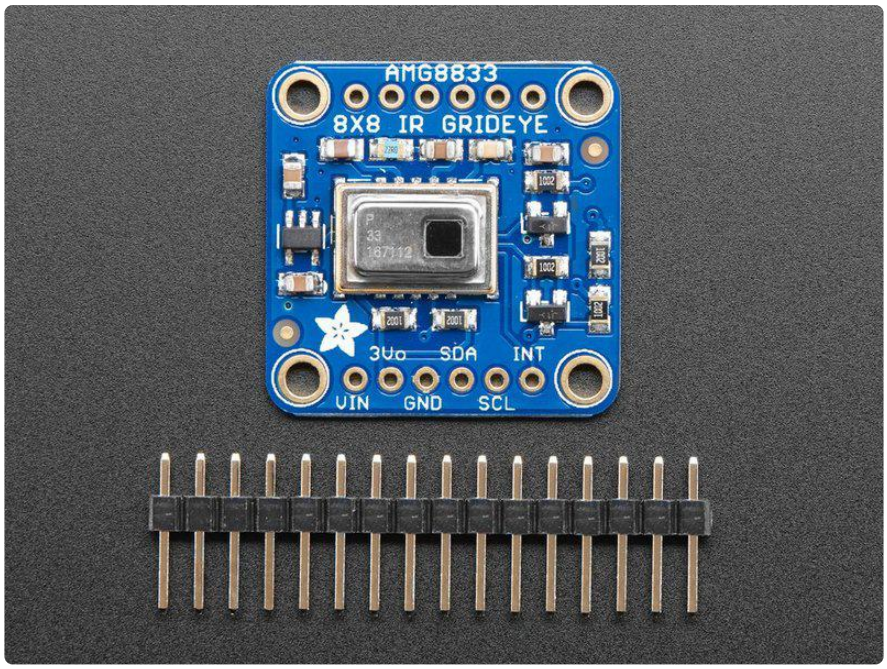


Should you wish to avoid soldering, we now also include our Stemma QT () connector s (SparkFun Qwiic () compatible). Using these handy connectors you can simply plug in the sensor, no soldering required! QT Cable is not included, but we have a variety in the shop ().

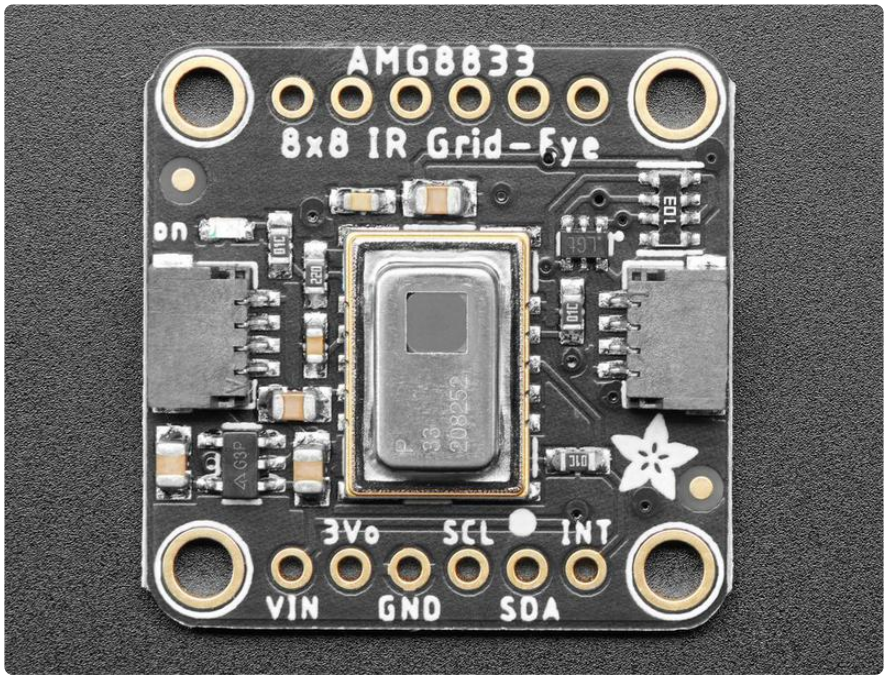


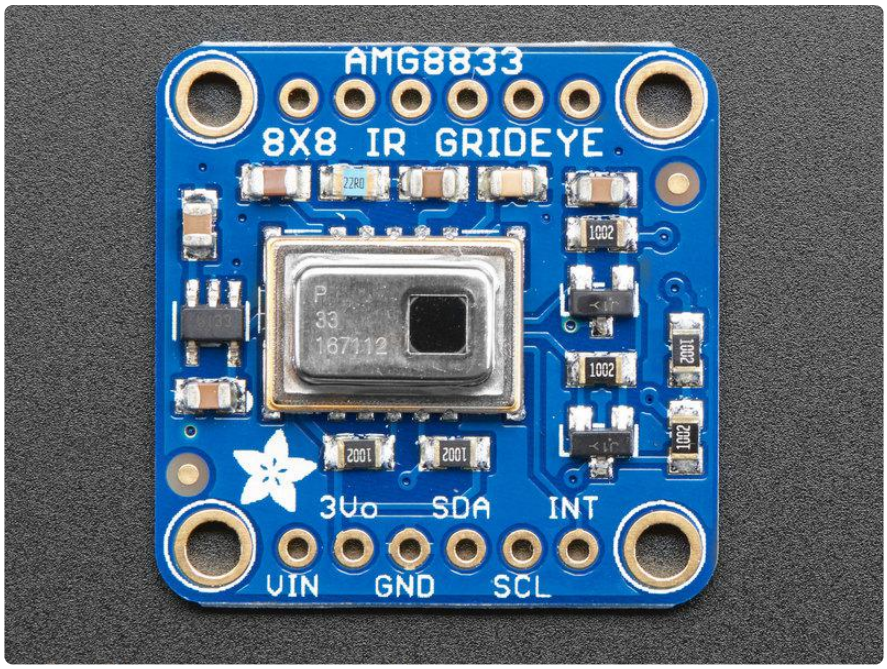
Even better - We've done all the hard work here, with example code and supporting software libraries to get you up in running in just a few lines of code!

There are two versions of this board - the STEMMA QT version shown above, and the original header-only version shown below. Code works the same on both!


AMG8833
8X8 IR GRIDEYE
3Vo
SDA
INT
VIN
GND
SCL

Pinouts


AMG8833
8x8 IR Grid-Eye
3Vo
SCL
INT
VIN
GND
SDA



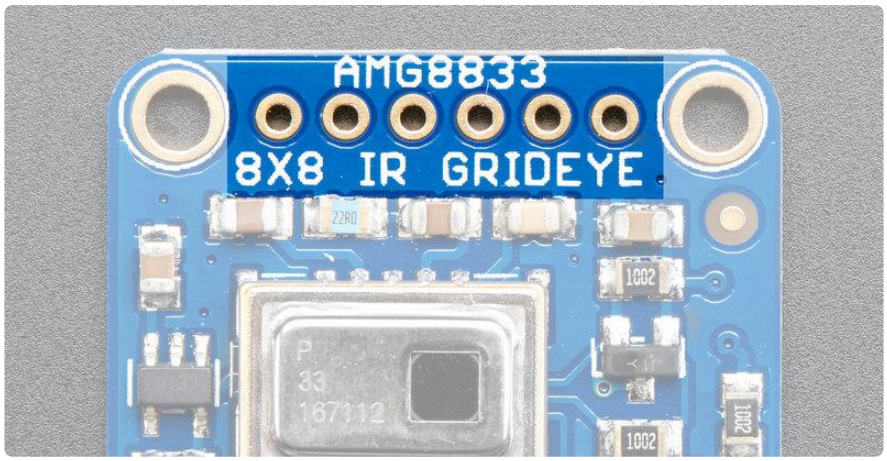
This camera has 4 mounting holes, and two header strips. Only the bottom strip is connected to the sensor. The top set of breakouts is there for mechanical stability only!

Power Pins:

- Vin - this is the power pin. Since the sensor uses 3.3V, we have included an onboard voltage regulator 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 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

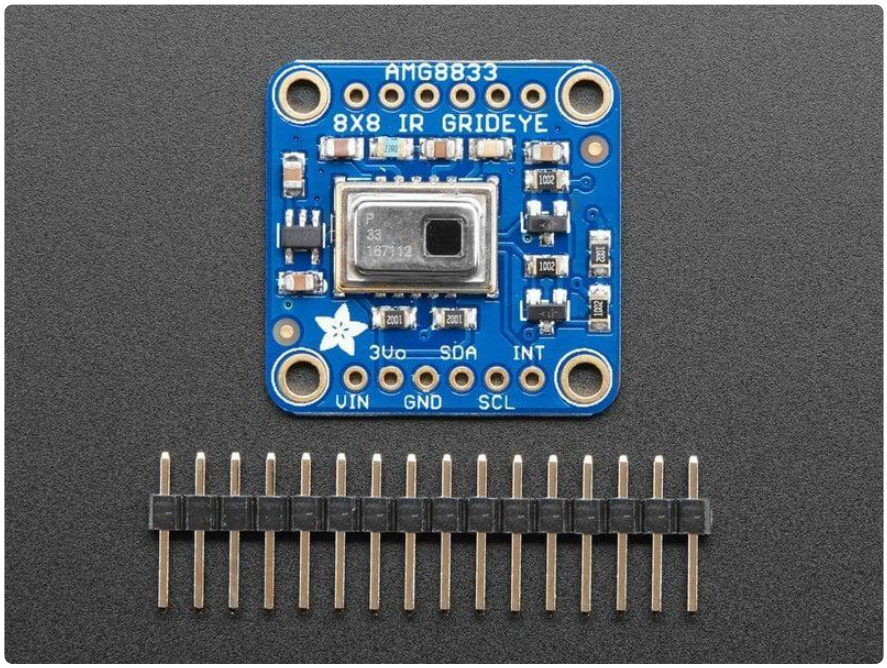
Logic pins:

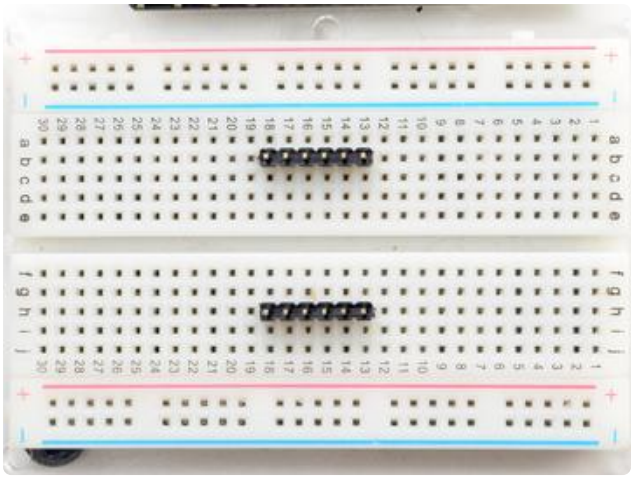
- SCL - this is the I2C clock pin, connect to your microcontrollers I2C clock line. There is a 10K pullup on this pin and it is level shifted so you can use 3 - 5VDC.
- SDA - this is the I2C data pin, connect to your microcontrollers I2C data line. There is a 10K pullup on this pin and it is level shifted so you can use 3 - 5VDC.
- INT - this is the interrupt-output pin. It is 3V logic and you can use it to detect when something moves or changes in the sensor vision path.
- STEMMA QT () - These connectors allow you to connectors to dev boards with S TEMMA QT connectors or to other things with various associated accessories ()



The 6 holes at the top of the board are provided for stability and are not connected to anything. Use these if you want your sensor to sit nice and flat on a breadboard or Perma-Proto.

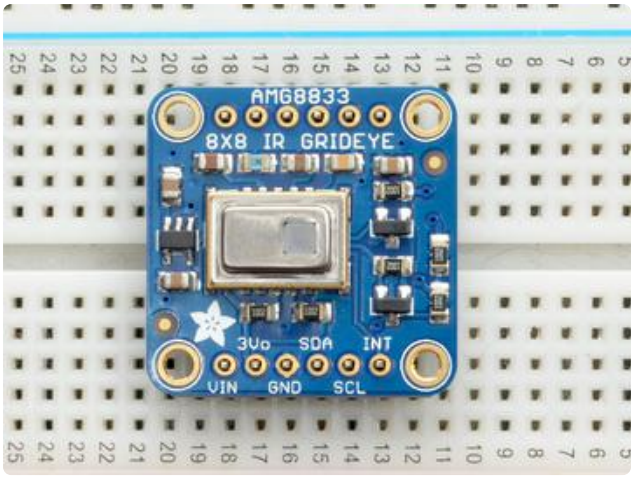
Assembly





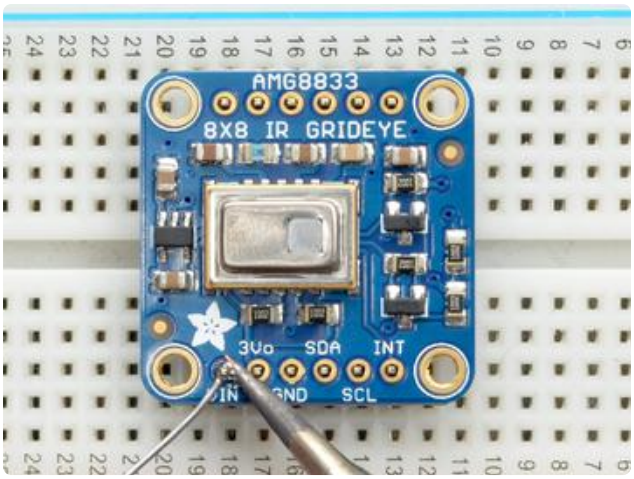
Prepare the header strips:

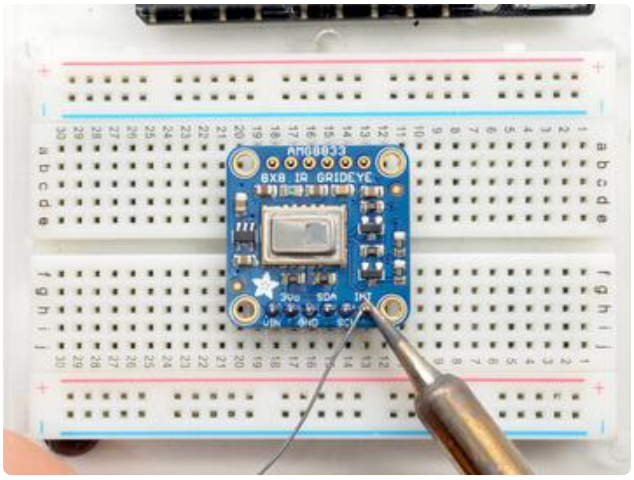
Cut the strips to length if necessary. It will be easier to solder if you insert it into a breadboard - long pins down

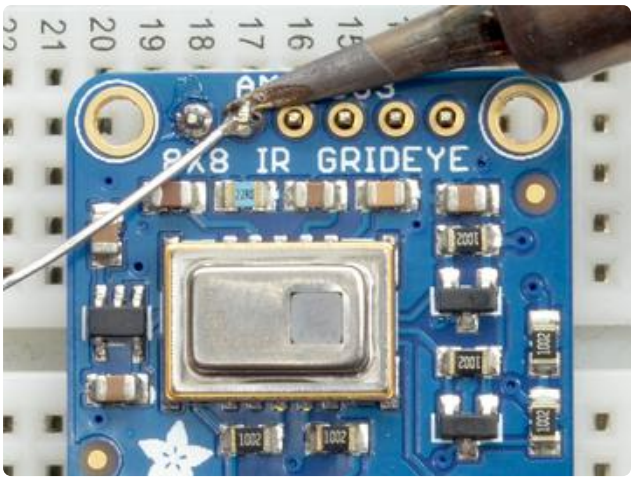


Add the breakout board:

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 ()).

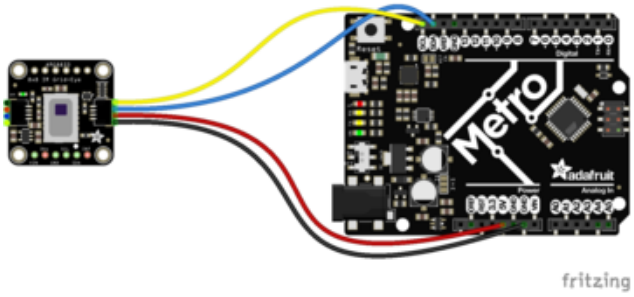
You're done! Check your solder joints visually and continue onto the next steps

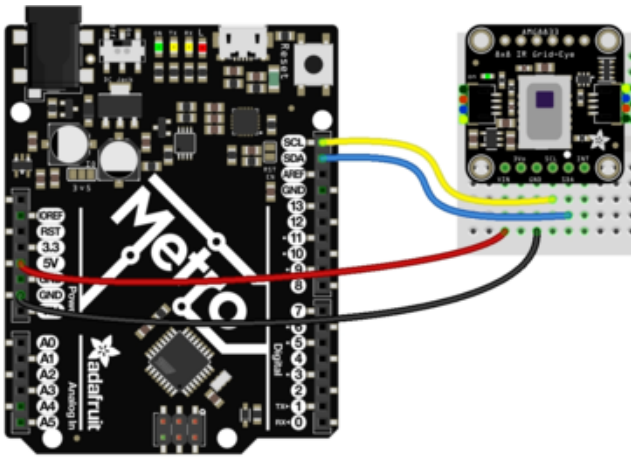
---

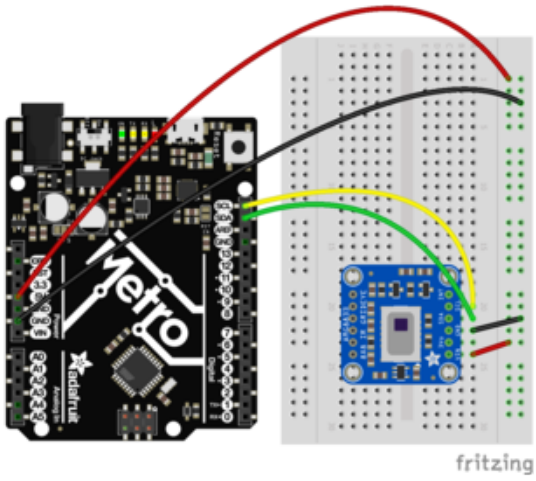
Arduino Wiring & Test

You can easily wire this breakout to any microcontroller, we'll be using an Arduino. You can use any other kind of microcontroller as well as long as it has I2C clock and I2C data lines.

I2C Wiring





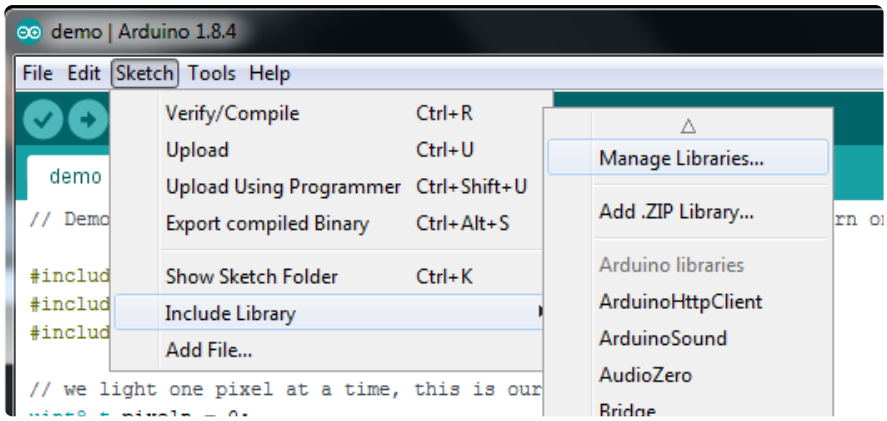


Connect Vin to the power supply, 3-5V is fine. Use the same voltage that the microcontroller logic is based off of. For most Arduinos, that is 5V (red wire on STEMMA QT version)
Connect GND to common power/data ground (black wire on STEMMA QT version)
Connect the SCL pin to the I2C clock SCL pin on your Arduino. (yellow wire on STEMMA QT version)
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. (blue wire on STEMMA QT version)
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
By default, the I2C address is 0x69. If you solder the jumper on the back of the board labeled "Addr", the address will change to 0x68.

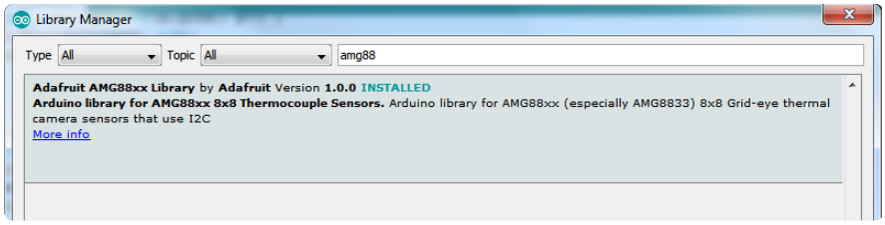
Download Adafruit_AMG88xx library

To begin reading sensor data, you will need to install the Adafruit_AMG88xx library ().

Start up the IDE and open the Library Manager:



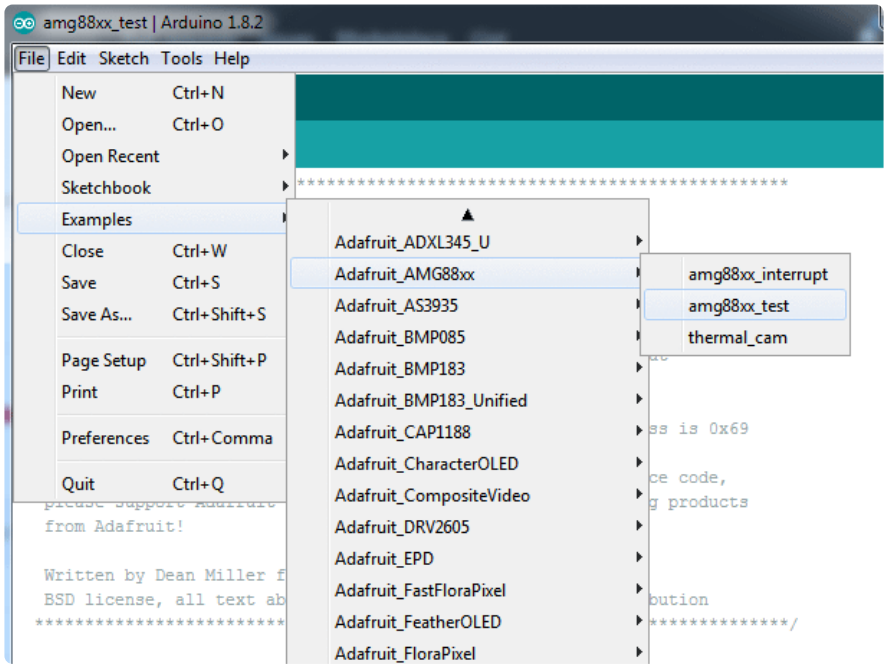
Type in AMG88xx until you see the Adafruit Library pop up. Click Install!



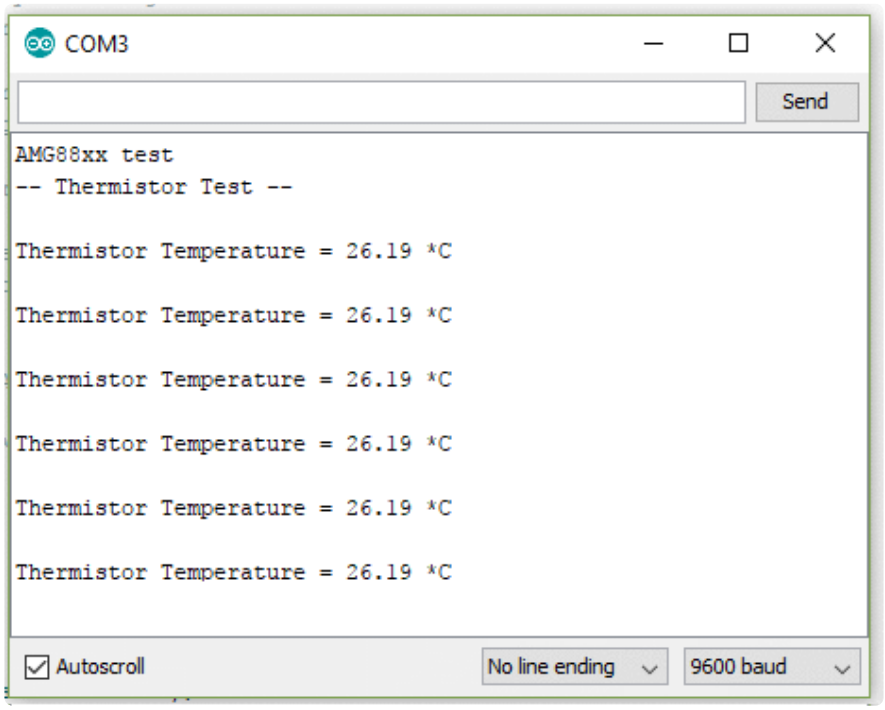
We also have a great tutorial on Arduino library installation at:
http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use ()

Load Thermistor Test

Open up File->Examples->Adafruit_AMG88xx->amg88xx_test and upload to your Arduino wired up to the sensor. This example just connects to the sensor and reads the internal thermistor to test your connections.



Once uploaded to your Arduino, open up the serial console at 9600 baud speed to see the internal thermistor reading. If you get a reading of ~26° degrees (room temperature) then everything is wired and working correctly!

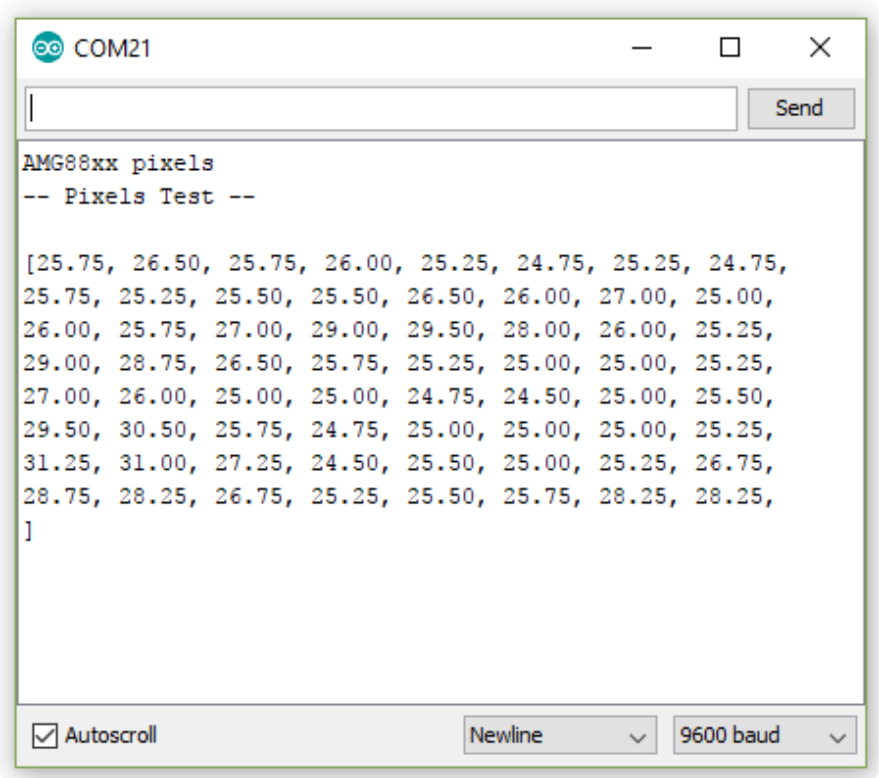


Pixel Array Output

OK now that we know the sensor is working, let's read actual thermal data. Load up Fi le -> Examples -> Adafruit_AMG88 -> pixels_test

Upload the code, and open the serial console at 9600 baud rate. You should see a printout of the array of readings every second. Each number is the detected temperature in Celsius, and in the 8x8 grid order that comes from the sensor

The numbers should increase if you put your hand or face above the sensor. They'll decrease if you hold up something cold in front of the sensor eye



Library Reference

To create the object, use

```
Adafruit_AMG88xx amg;
```

Initialize the sensor using

```
status = amg.begin();
    if (!status) {
        Serial.println("Could not find a valid AMG88xx sensor, check wiring!");
        while (1);
    }
```

to read the pixels you will need an array to place the readings into. Once you have one, you can call readPixels. Make sure the array you create is big enough by using the pre-defined AMG88xx_PIXEL_ARRAY_SIZE macro.

```
float pixels[AMG88xx_PIXEL_ARRAY_SIZE];
amg.readPixels(pixels);
```

Arduino Library Docs

Arduino Library Docs ()

Arduino Thermal Camera

To make your Arduino into a cool thermal camera, we can add a small display.

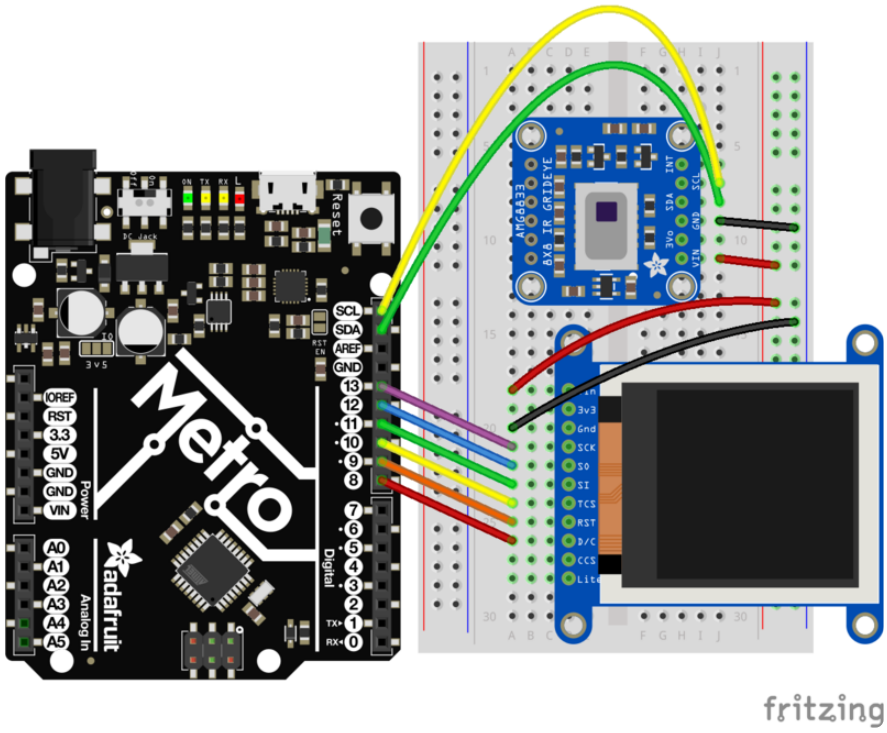
In this example we use an Adafruit 1.44" Color TFT. With some code changes, you can use other size displays but a color display is best of course.

Adafruit 1.44" Color TFT LCD Display with MicroSD Card breakout

This lovely little display breakout is the best way to add a small, colorful and bright display to any project. Since the display uses 4-wire SPI to communicate and has its own...

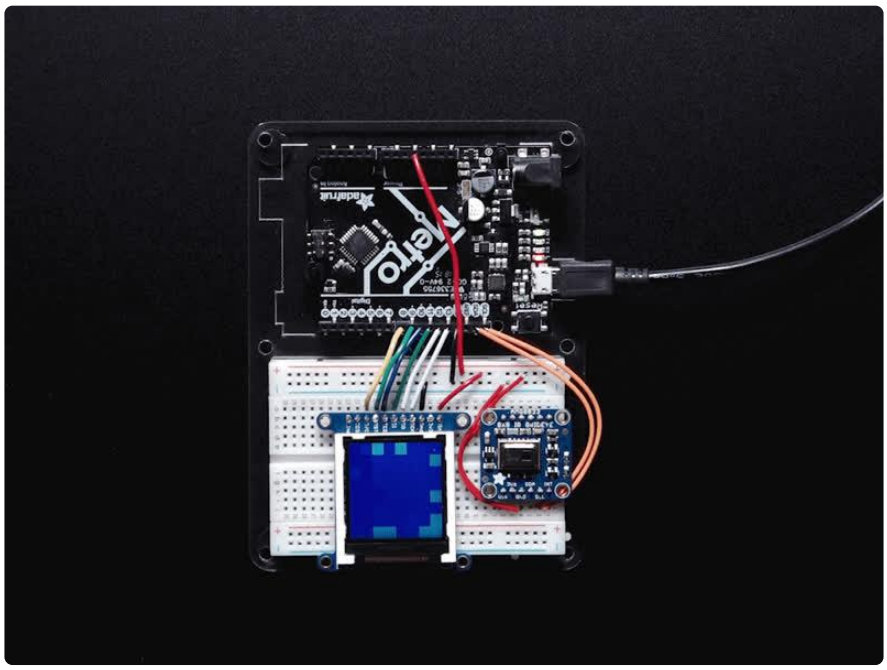
https://www.adafruit.com/product/2088

Keep your AMG8833 breakout wired as you already have it from the Wiring & Test section above, and add your TFT like this



Once everything is all wired up, load up File->Examples->Adafruit_AMG88xx->thermal_cam

Hit upload and you should have a simple thermal camera!



James DV has also sent over a version that is optimized if you want a faster display-update rate ()

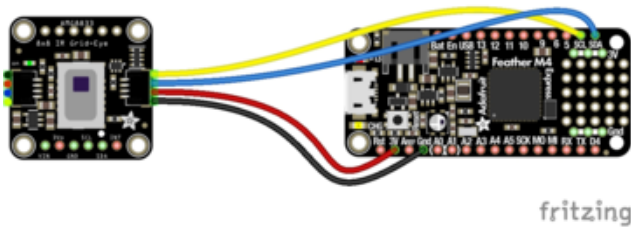
Python & CircuitPython

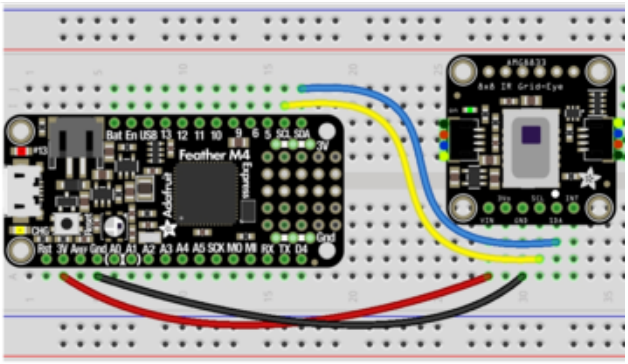
It's easy to use the AMG8833 sensor with Python or CircuitPython and the Adafruit CircuitPython AMG88xx () module. This module allows you to easily write Python code that reads thermal imaging data from the sensor.

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 ().

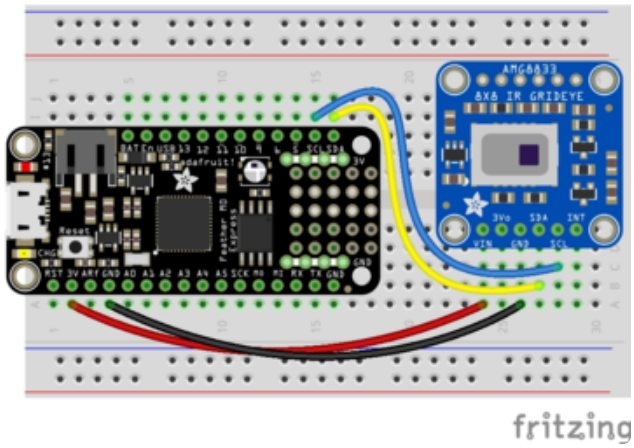
CircuitPython Microcontroller Wiring

First wire up a AMG8833 to your board exactly as shown on the previous pages for Arduino. Here's an example of wiring a Feather M0 to the sensor with I2C:





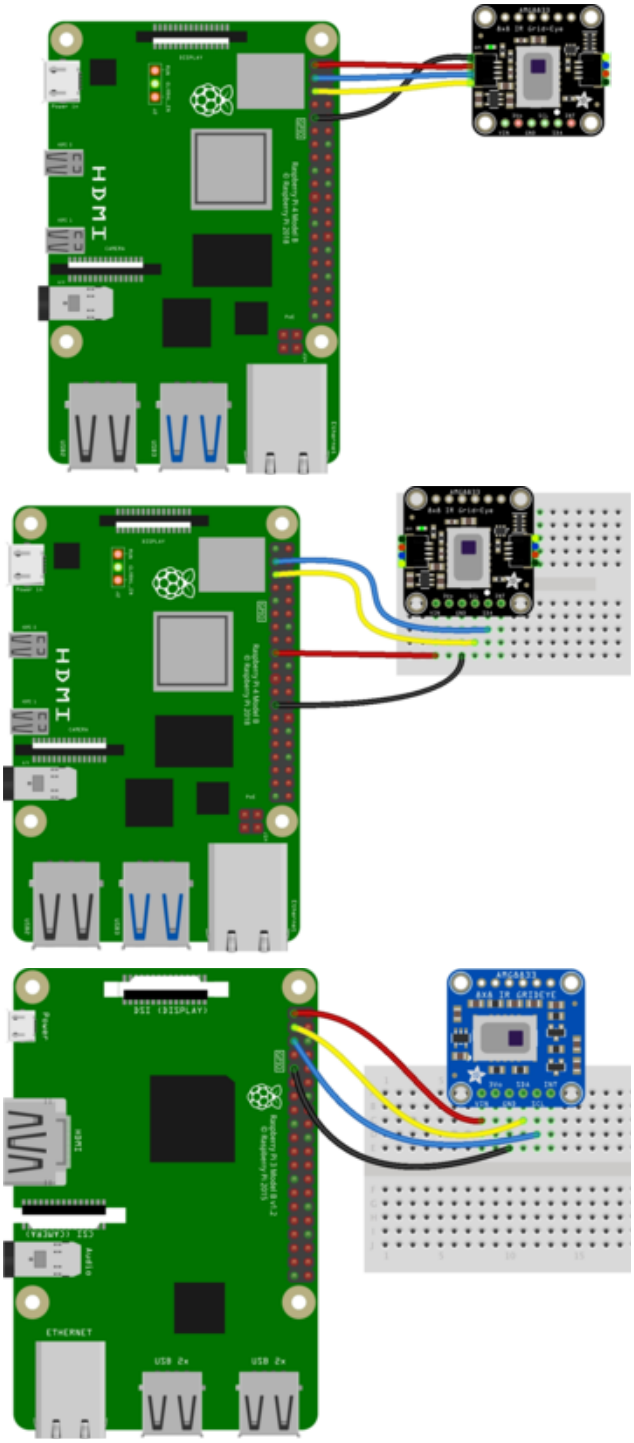
Board 3V to sensor VIN (red wire on STEMMA QT version)
Board GND to sensor GND (black wire on STEMMA QT version)
Board SCL to sensor SCL (yellow wire on STEMMA QT version)
Board SDA to sensor SDA (blue wire on STEMMA QT version)



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 (red wire on STEMMA QT version)
Pi GND to sensor GND (black wire on STEMMA QT version)
Pi SCL to sensor SCL (yellow wire on STEMMA QT version)
Pi SDA to sensor SDA (blue wire on STEMMA QT version)

CircuitPython Installation of AMG88xx Library

You'll need to install the Adafruit CircuitPython AMG88xx () 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 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_amg88xx.mpy
- adafruit_bus_device

Before continuing make sure your board's lib folder or root filesystem has the adafruit_amg88xx.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 AMG88xx 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-amg88xx`

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 thermal imaging data the board's Python REPL.

If you're using an I2C connection run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```
import time
import busio
import board
import adafruit_amg88xx
i2c = busio.I2C(board.SCL, board.SDA)
amg = adafruit_amg88xx.AMG88XX(i2c)
```

Now you're ready to read values from the sensor using any of these properties:

- temperature - The sensor temperature in degrees Celsius.
- pixels - Temperature of each pixel across the sensor in Celsius. Temperatures are stored in a two dimensional list where the first index is the row and the second is the column. The first row is on the side closest to the writing on the sensor.

For example, to print the pixel temp once every second until you tell it to stop:

```
while True:
    for row in amg.pixels:
        print(['{0:.1f}'.format(temp) for temp in row])
        print("")
    print("\n")
    time.sleep(1)
```

```
['27.8', '26.7', '26.7', '26.5', '26.7', '26.7', '26.7', '27.3']
['27.8', '27.0', '26.0', '26.5', '26.5', '26.3', '26.5', '26.7']
['27.0', '25.8', '26.5', '26.0', '26.3', '26.7', '26.5', '26.5']


['26.3', '26.3', '25.8', '26.5', '26.5', '27.0', '26.5', '26.0']
['26.3', '26.3', '26.3', '26.7', '26.5', '26.7', '26.5', '27.0']
['26.0', '26.0', '26.5', '26.3', '26.7', '27.0', '27.5', '26.7']
['26.5', '26.5', '26.3', '26.7', '26.3', '26.5', '27.0', '27.0']
['27.3', '26.7', '26.5', '26.5', '26.7', '26.5', '26.3', '27.0']
['27.8', '26.0', '26.5', '26.3', '26.3', '27.0', '26.5', '26.5']
['26.7', '26.5', '26.5', '26.5', '26.7', '26.5', '26.0', '26.3']
['26.3', '26.3', '26.0', '26.5', '25.8', '25.8', '26.3', '26.5']
```

That's all there is to using AMG88ss with CircuitPython!

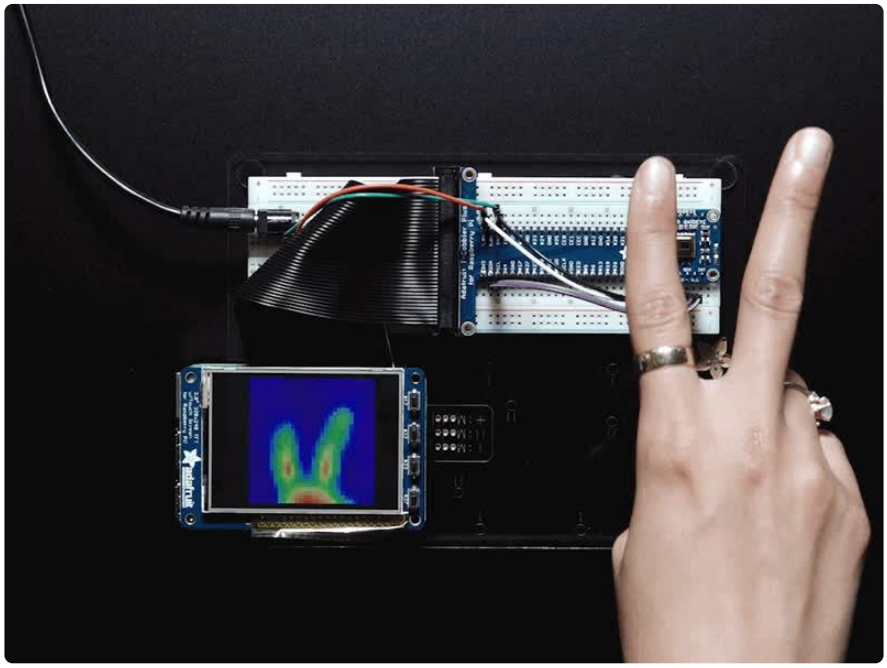
Full Example Code

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

import time
import busio
import board
import adafruit_amg88xx

i2c = busio.I2C(board.SCL, board.SDA)
amg = adafruit_amg88xx.AMG88XX(i2c)

while True:
    for row in amg.pixels:
        # Pad to 1 decimal place
        print(["{0:.1f}".format(temp) for temp in row])
        print("")
    print("\n")
    time.sleep(1)
```

Python Docs

Python Docs ()

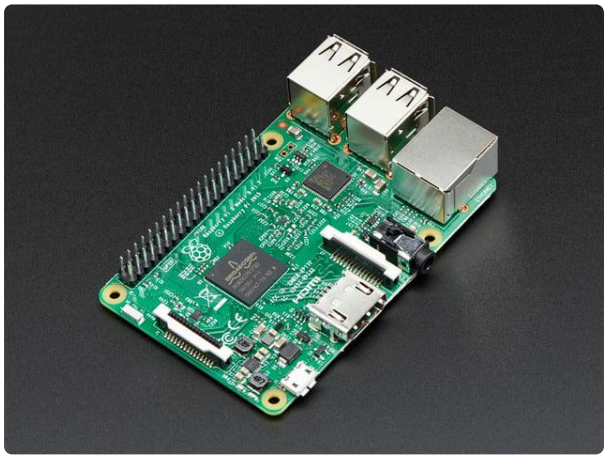
Raspberry Pi Thermal Camera



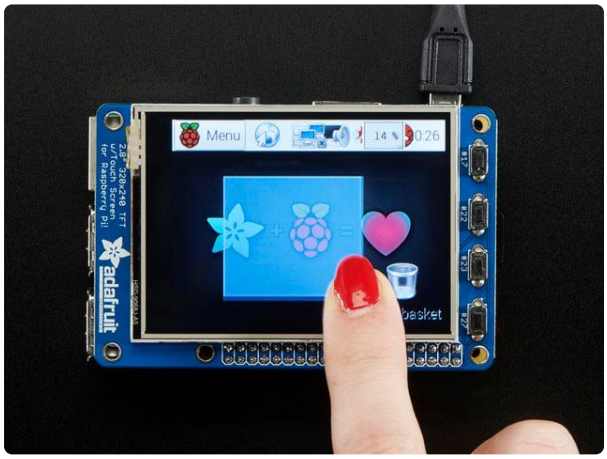
The Raspberry Pi also has an i2c interface, and even better has processing capability to interpolate and filter the sensor output. By adding processing power, you can 'turn' the 8x8 output into what appears to be a higher-resolution display.

We're using a PiTFT 2.8" and a Pi Cobbler but the code can be adapted to output to the HDMI display - we're using pygame to draw to the framebuffer.

You can use any Raspberry Pi computer, from Pi A+ to Pi 3 or even a Pi Zero, but we happen to have a Pi 3 on our desk set up already so we're using that.



Raspberry Pi 3 - Model B - ARMv8 with 1G RAM
NOTE: Due to stock limitations we may only be able to offer refunds or store credit for Pis that are defective, damaged or lost in transit.Did you really...
https://www.adafruit.com/product/3055



PiTFT Plus Assembled 320x240 2.8" TFT + Resistive Touchscreen
Is this not the cutest little display for the Raspberry Pi? It features a 2.8" display with 320x240 16-bit color pixels and a resistive touch overlay. The plate uses the high...
https://www.adafruit.com/product/2298

Assembled Pi T-Cobbler Plus - GPIO Breakout
This is the assembled version of the Pi T-Cobbler Plus. It only works with the Raspberry Pi Model Zero, A+, B+, Pi 2, Pi 3 & Pi 4! (Any Pi with 2x20...
https://www.adafruit.com/product/2028

Setup PiTFT

If you have not done so already, the first thing you will need to do is setup your PiTFT. Instructions on how to do so can be found in this guide ().

Install Python Software

Once your PiTFT is all set up, and you have Internet access set up go back to this page and install the Python software for the AMG8833 so you can read data from the sensor. ()

Finally, install both pygame and scipy. Pygame lets us draw easily to a screen using python, we'll use that to make the display work. Scipy is a powerful scientific/data processing library that we can use to magically turn the 8x8 = 64 pixel array into something that looks more like a 32x32 = 1024 pixel array. Wow, isn't digital signal processing cool?

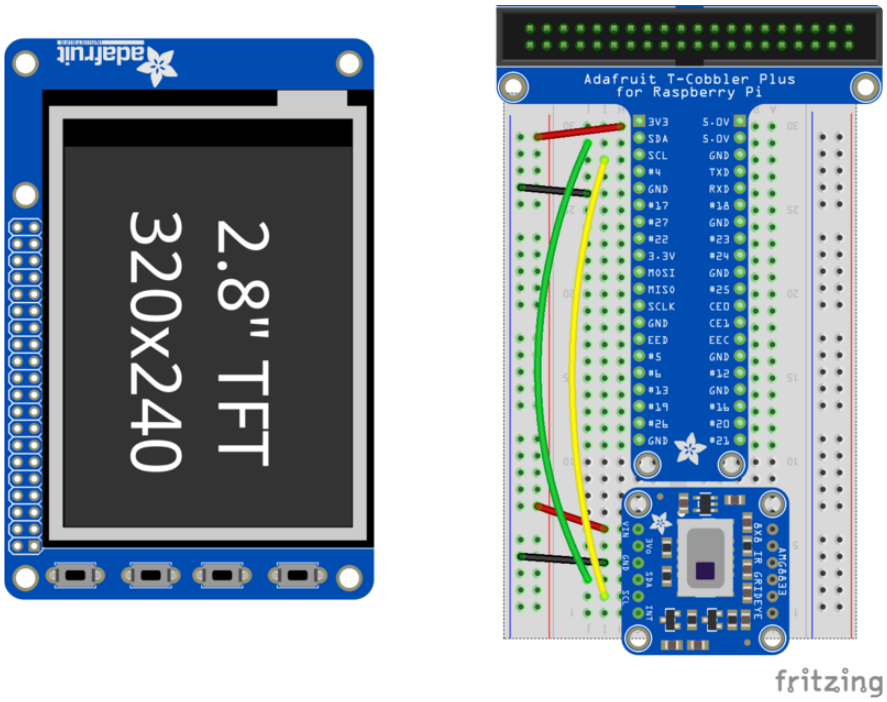
```
sudo apt-get install -y python-scipy python-pygame
sudo pip3 install colour
```

Wiring Up Sensor

With the Pi powered off, we can wire up the sensor to the Pi Cobbler like this:

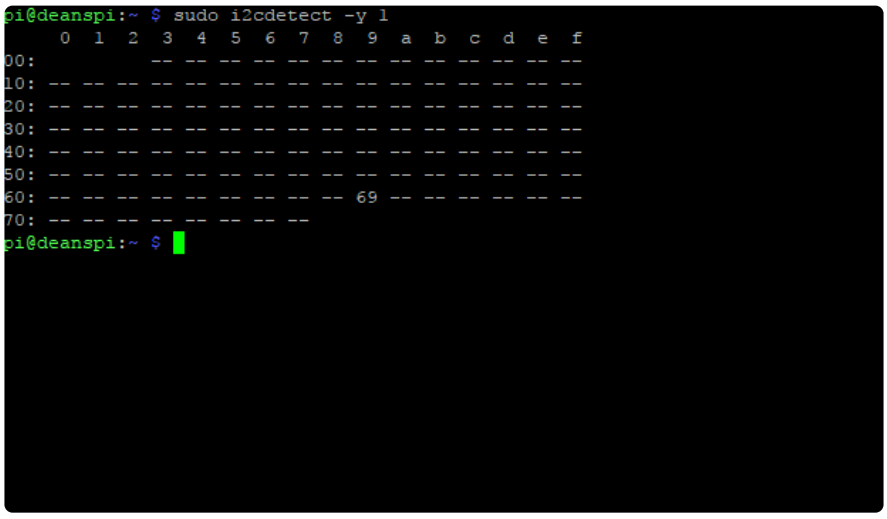
- Connect Vin to the 3V or 5V power supply (either is fine)
- Connect GND to the ground pin on the Cobbler
- Connect SDA to SDA on the Cobbler
- Connect SCL to SCL on the Cobbler

You can also use direct wires, we happen to have a Cobbler ready. remember you can plug the cobbler into the bottom of the PiTFT to get access to all the pins!



Now you should be able to verify that the sensor is wired up correctly by asking the Pi to detect what addresses it can see on the I2C bus:

```
sudo i2cdetect -y 1
```



It should show up under its default address (0x69). If you don't see 0x69, check your wiring, did you install I2C support, etc?

Run example code

At long last, we are finally ready to run our example code

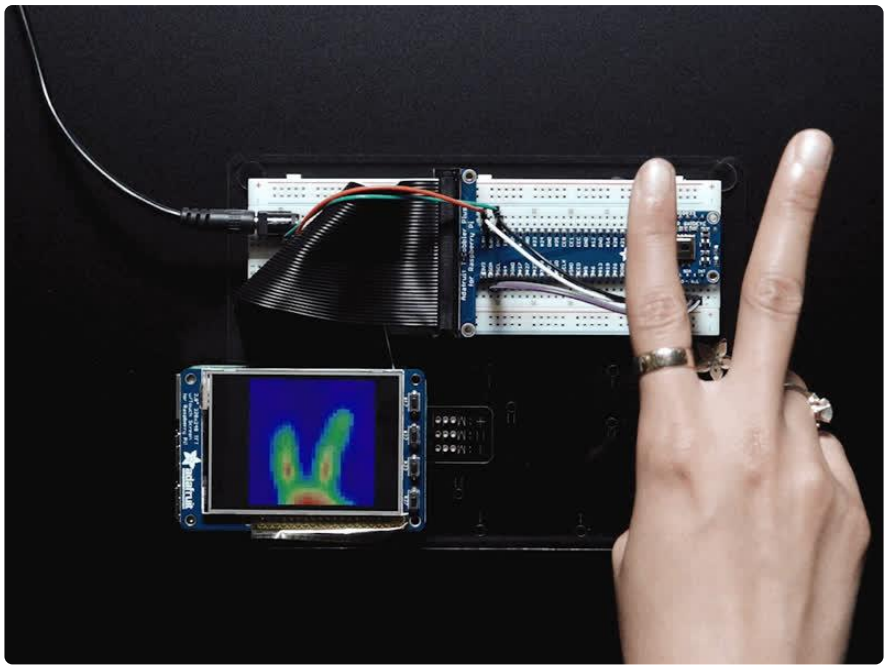
```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

"""This example is for Raspberry Pi (Linux) only!
   It will not work on microcontrollers running CircuitPython!"""

import os
import math
import time

import numpy as np
import pygame
import busio
import board

from scipy.interpolate import griddata

from colour import Color

import adafruit_amg88xx

i2c_bus = busio.I2C(board.SCL, board.SDA)

# low range of the sensor (this will be blue on the screen)
MINTEMP = 26.0

# high range of the sensor (this will be red on the screen)
MAXTEMP = 32.0

# how many color values we can have
COLORDEPTH = 1024

os.putenv("SDL_FBDEV", "/dev/fb1")
# pylint: disable=no-member
pygame.init()
# pylint: enable=no-member

# initialize the sensor
sensor = adafruit_amg88xx.AMG88XX(i2c_bus)

# pylint: disable=invalid-slice-index
points = [(math.floor(ix / 8), (ix % 8)) for ix in range(0, 64)]
grid_x, grid_y = np.mgrid[0:7:32j, 0:7:32j]
# pylint: enable=invalid-slice-index

# sensor is an 8x8 grid so lets do a square
height = 240
width = 240

# the list of colors we can choose from
blue = Color("indigo")
colors = list(blue.range_to(Color("red"), COLORDEPTH))

# create the array of colors
colors = [(int(c.red * 255), int(c.green * 255), int(c.blue * 255)) for c in colors]

displayPixelWidth = width / 30
displayPixelHeight = height / 30

lcd = pygame.display.set_mode((width, height))

lcd.fill((255, 0, 0))

pygame.display.update()
pygame.mouse.set_visible(False)

lcd.fill((0, 0, 0))
pygame.display.update()
```

```
# some utility functions
def constrain(val, min_val, max_val):
    return min(max_val, max(min_val, val))


def map_value(x, in_min, in_max, out_min, out_max):
    return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min


# let the sensor initialize
time.sleep(0.1)

while True:

    # read the pixels
    pixels = []
    for row in sensor.pixels:
        pixels = pixels + row
    pixels = [map_value(p, MINTEMP, MAXTEMP, 0, COLORDEPTH - 1) for p in pixels]

    # perform interpolation
    bicubic = griddata(points, pixels, (grid_x, grid_y), method="cubic")

    # draw everything
    for ix, row in enumerate(bicubic):
        for jx, pixel in enumerate(row):
            pygame.draw.rect(
                lcd,
                colors[constrain(int(pixel), 0, COLORDEPTH - 1)],
                (
                    displayPixelHeight * ix,
                    displayPixelWidth * jx,
                    displayPixelHeight,
                    displayPixelWidth,
                ),
            )

    pygame.display.update()
```

If you have everything installed and wired up correctly, you should see a nice thermal camera image. Cool tones (blue and purple) are cooler temperatures, and warmer tones (yellow, red) are warmer temperatures.

If your image seems to be flipped on the screen, try changing the orientation of the AMG8833 breakout on the breadboard.

If you're interested int he details, and want to know more about how we made 64 pixels look like many more, it's called bicubic interpolation () (hat tip to OSHpark for the idea ()!)

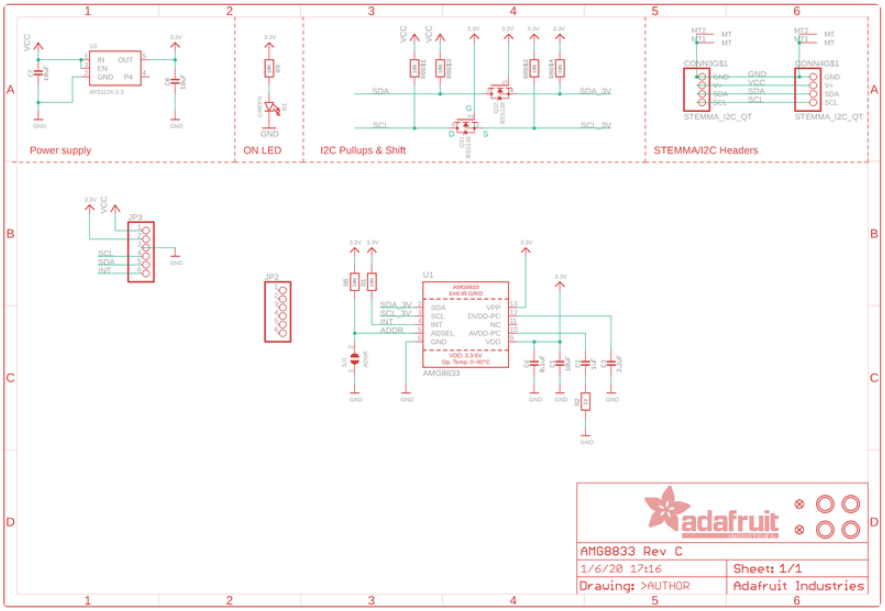


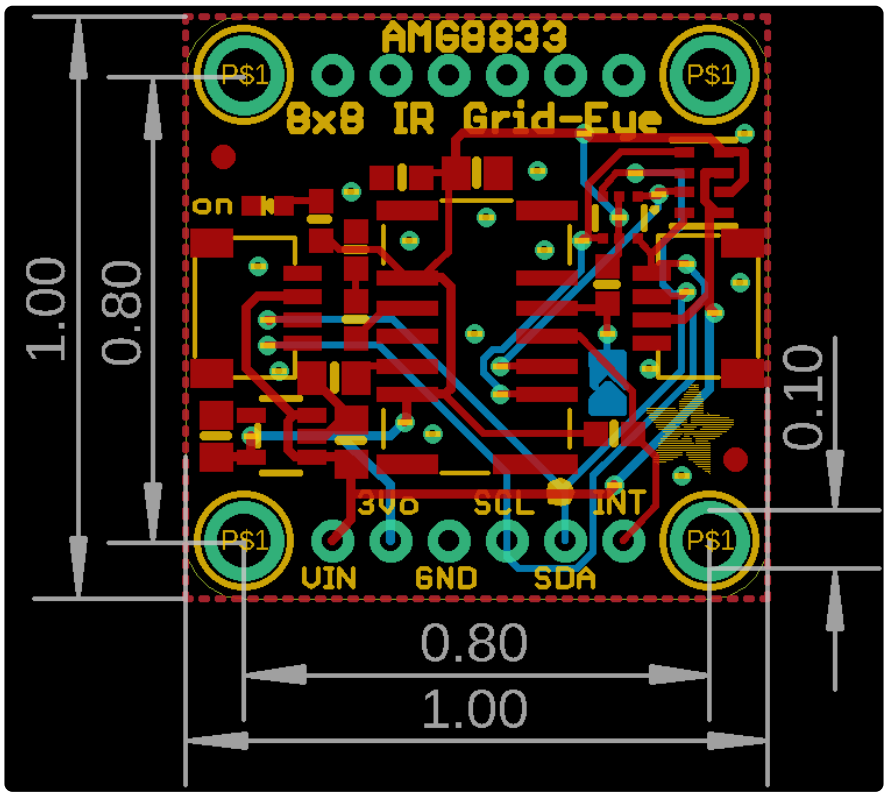
Downloads

Documents

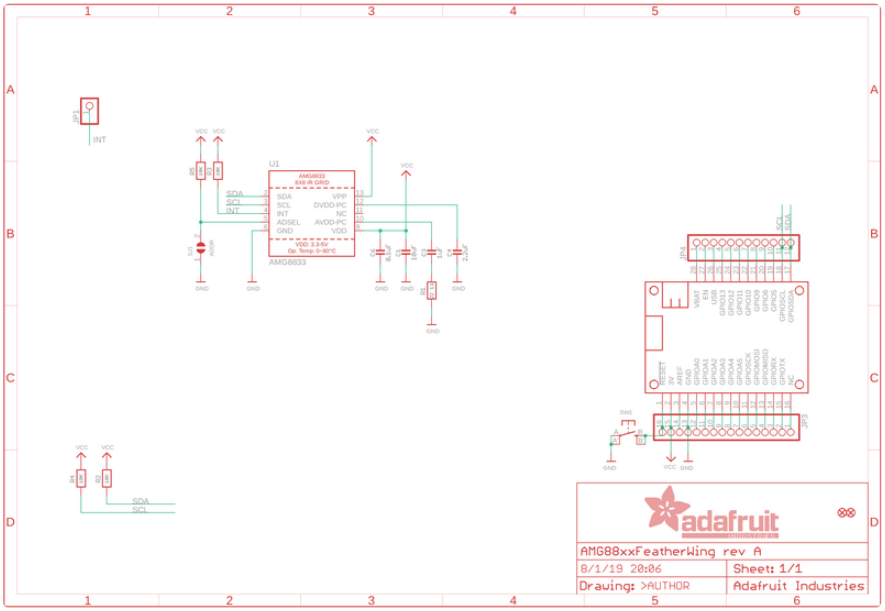
- AMG8833 datasheet ()
- AMG8833 Arduino Driver ()
- Fritzing object for STEMMA QT version in Adafruit Fritzing Library ()
- Fritzing object in the Adafruit Fritzing library ()
- AMG8833 CircuitPython Driver ()
- AMG8833 breakout PCB files (EAGLE format) ()

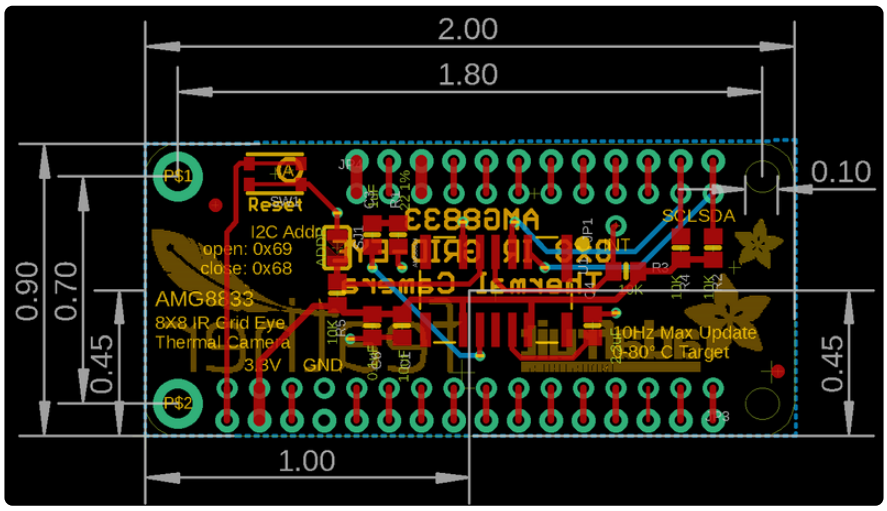
Schematic and Fab Print STEMMA QT Version





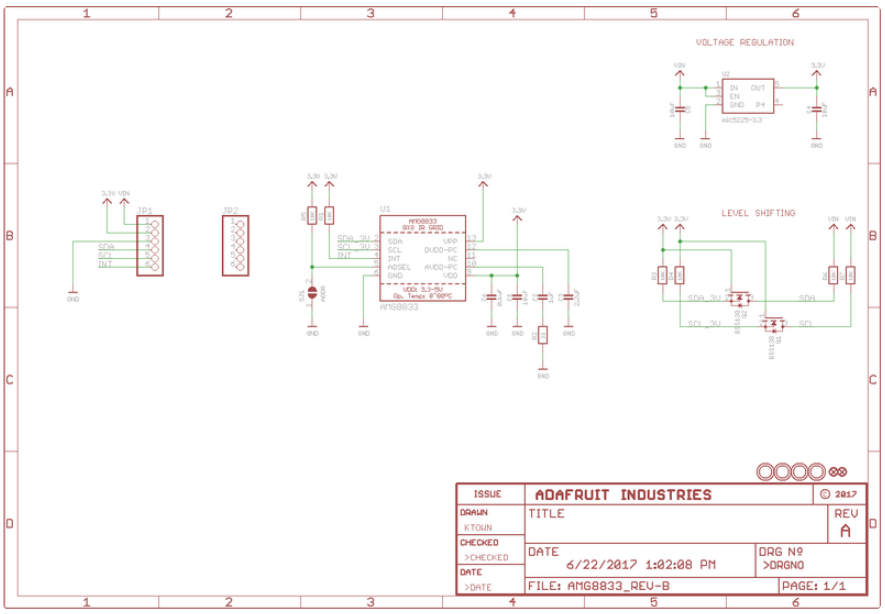
Schematic and Fab Print FeatherWing Version





Schematic Original Breakout Version

click to enlarge



Dimensions Original Breakout Version

in inches. Click to enlarge