



MAX31855 Thermocouple

Created by lady ada



<https://learn.adafruit.com/thermocouple>

Last updated on 2023-08-29 02:07:45 PM EDT

Table of Contents

Overview	3
<hr/>	
<ul style="list-style-type: none">• Some Basic Stats• This is for a K-type thermocouple with glass overbraiding	
Wiring a Thermocouple	4
<hr/>	
Arduino Code	5
<hr/>	
<ul style="list-style-type: none">• Arduino Library• Adding a Display• 	
Python & CircuitPython	9
<hr/>	
<ul style="list-style-type: none">• CircuitPython Microcontroller Wiring• Python Computer Wiring• CircuitPython Installation of MAX31855 Library• Python Installation of MAX31855 Library• CircuitPython & Python Usage• Full Example Code	
Python Docs	12
<hr/>	
F.A.Q.	12
<hr/>	
Project Examples	14
<hr/>	
Downloads	17
<hr/>	
<ul style="list-style-type: none">• Datasheets & Files• Schematic• Fabrication Print	

Overview

A thermocouple is a kind of temperature sensor.

Unlike [semiconductor temperature sensors such as the TMP36 \(\)](#) , thermocouples have no electronics inside them, they are simply made by welding together two metal wires. Because of a physical effect of two joined metals, there is a slight but measurable voltage across the wires that increases with temperature. The type of metals used affect the voltage range, cost and sensitivity, which is why we have a few different kinds of thermocouples. The main improvement of using a thermocouple over a semiconductor sensor or thermistor is that the temperature range is very much increased. For example, the TMP36 can go from -50 to 150°C, after that the chip itself can be damaged. Common thermocouples on the other hand, can go from -200°C to 1350°C (K type) and there are ones that can go above 2300°C!



Thermocouples are often used in HVAC systems, heaters and boilers, kilns, etc. There are a few different kinds but this tutorial will discuss K type, which are very common and easier to interface with.

One difficulty in using them is that the voltage to be measured is very small, with changes of about 50 μV per $^{\circ}\text{C}$ (a μV is 1/1000000 Volts). While it is possible to read these voltages using a clean power supply and nice op-amps, there are other complications such as a non-linear response (its not always 50 $\mu\text{V}/^{\circ}\text{C}$) and cold-temperature compensation (the effect measured is only a differential and there must be a reference, just as ground is a reference for voltage). For that reason, we suggest only using an interface chip that will do the heavy lifting for you, allow you to easily

integrate the sensor without as much pain. In this tutorial we will use a MAX6675 K-thermocouple interface chip which doesn't even require an ADC, spitting out a nice digital data signal of the temperature.

Some Basic Stats

This is for a K-type thermocouple with glass overbraiding

- Size: 24 gauge, 1 meter long (you can cut it down if desired)
- Price: [\\$10 at the adafruit store \(\)](#)
- Temperature range: -100°C to 500°C / -150 to 900°F (After this the glass overbraiding may be damaged)
- Output range: -6 to +20mV
- Precision: +-2°C
- [Requires an amplifier such as MAX31855 \(http://adafru.it/269\)](#)
- Interface: [MAX6675 \(\)](#) (discontinued) [MAX31855 \(http://adafru.it/269\)](#), or [AD595 \(\)](#) (analog)
- [K Thermocouple Datasheet \(\)](#)
- [MAX6675 Datasheet \(\)](#)
- [MAX31855 Datasheet \(\)](#)

Wiring a Thermocouple

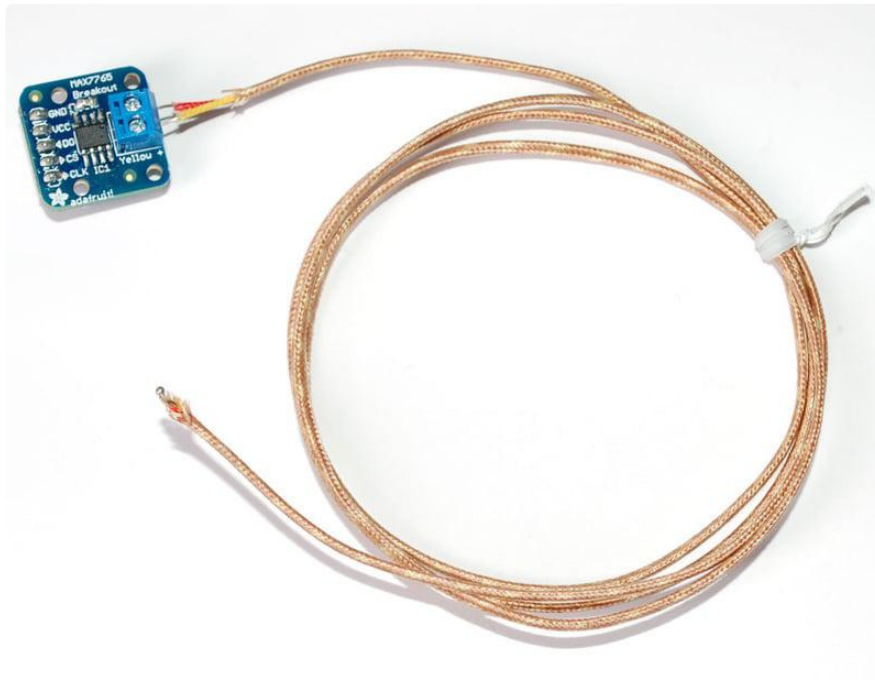
As we mentioned before, trying to actually measure the voltage across the wires will be very difficult for most people, which is why we strongly suggest using a thermocouple interface chip. The nicest one we've seen so far is the MAX6675 (and its replacement version called the MAX31855) which unfortunately is only available in SOIC package. While not too difficult to solder, [we nevertheless have in the shop a breakout board that is ready to go \(http://adafru.it/269\)](#) .

First thing to determine is which wire is which. As you recall, thermocouples are made by welding together two wires, the chip reads the voltage difference between the two. One is the negative (for K-type its made of Alumel) and the other positive (ditto, Chromel). Luckily the wires are color coded, and almost all of the time you'll find the Alumel is red and the Chromel is yellow.

Connect the leads as required to your amplifier:

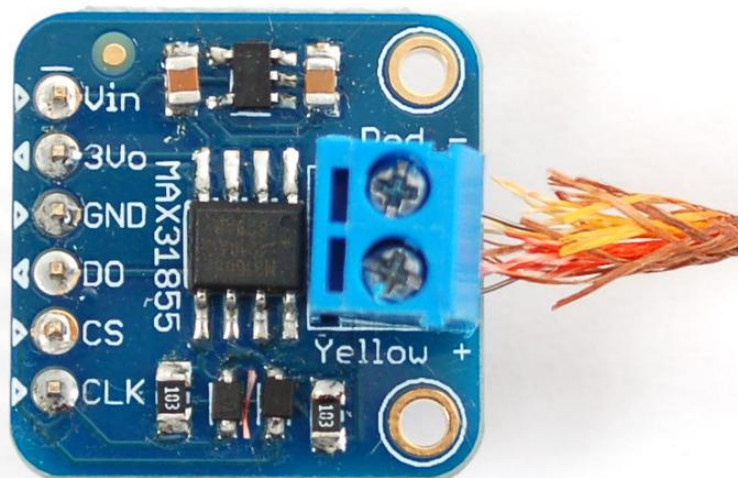
The MAX6675 and MAX31855 thermocouple amplifiers are not compatible with grounded thermocouples.

We've seen some K type thermocouples where the leads were marked incorrectly, so if you find that the thermocouple temperature goes down instead of up when heated, try swapping the red and yellow wires



Arduino Code

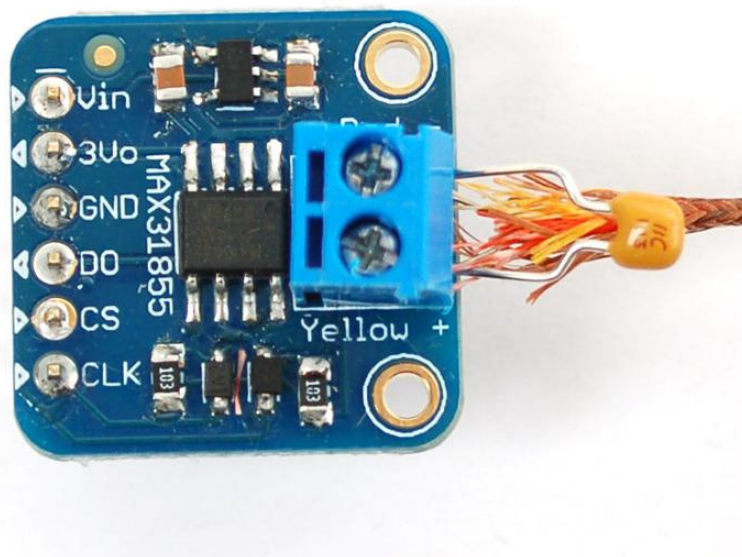
If you're using an AD595 interface chip, you can simply connect the voltage output to an analog input on your microcontroller and do some basic math to multiply the 10 mV/°C input into numerical output.



If you're planning to use the MAX6675/MAX31855, there's a little more work to be done. First off, Vin and GND must connect to a 3-5V supply. Then the three data pins must connect to digital IO pins:

- CLK (clock) is an input to the MAX6675/MAX31855 (output from microcontroller) which indicates when to present another bit of data
- DO (data out) is an output from the MAX6675/MAX31855 (input to the microcontroller) which carries each bit of data
- CS (chip select) is an input to the MAX6675/MAX31855 (output from the microcontroller) which tells the chip when its time to read the thermocouple and output more data.

In the beginning of our sketches, we define these pins. For our examples DO connects to digital 3, CS connects to digital 4, and CLK connects to pin 5



If you are using the MAX31855 v1.0 in a noisy environment, you may need to add a 0.01uF capacitor across the thermocouple leads.

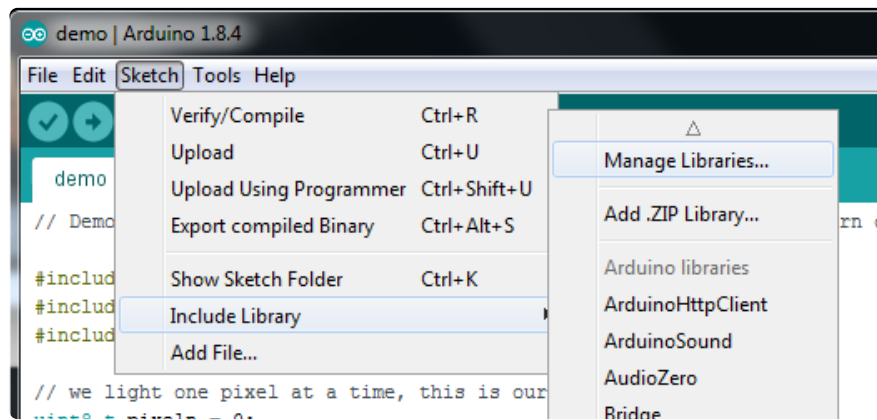
The MAX31855 does not support grounded thermocouples - if the sensor touches ground the chip will return an error

Arduino Library

If you have an older MAX6675 breakout, download the Adafruit MAX6675 library from the Arduino library manager.

If you have the newer MAX31855 breakout, download the Adafruit MAX31855 library from the Arduino library manager.

Open up the Arduino library manager:



If you have a MAX6675 breakout, search for the MAX6675 library and install it



If you have the MAX31855 breakout, search for the Adafruit MAX31855 library and install it



Open up the File->Examples->MAX6675/Adafruit_MAX31855->serialthermocouple sketch and upload it to your Arduino. Once uploaded, open up the serial port monitor to display the current temperatures in both Celsius and Fahrenheit.

We also have a great tutorial on Arduino library installation at:

<http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use> ()

For this wiring, we connected CLK to digital 3, CS to digital 4 and DO to digital 5. Once you get it working, you can change the pin connections in the sketch

We have an example sketch for this as well. [First get the LCD working by following our tutorial \(\)](#). Now load up the new sketch File->Examples->MAX31855>Lcdthermocouple and plug in the thermocouple module as we did in the serial thermocouple test, you'll see the internal temperature and the thermocouple temperature displayed in Celsius

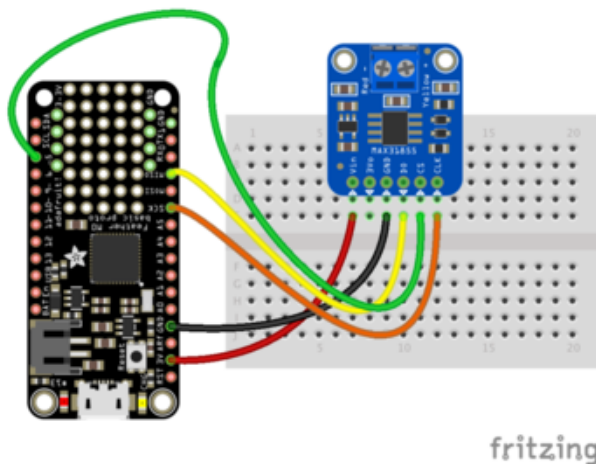
Python & CircuitPython

It's easy to use the MAX31855 sensor with Python and CircuitPython, and the [Adafruit CircuitPython MAX31855 \(\)](#) module. This module allows you to easily write Python code that reads the temperature from the thermocouple.

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 MAX31855 to your board exactly as shown on the previous pages for Arduino. Here's an example of wiring a Feather M0 to the sensor:

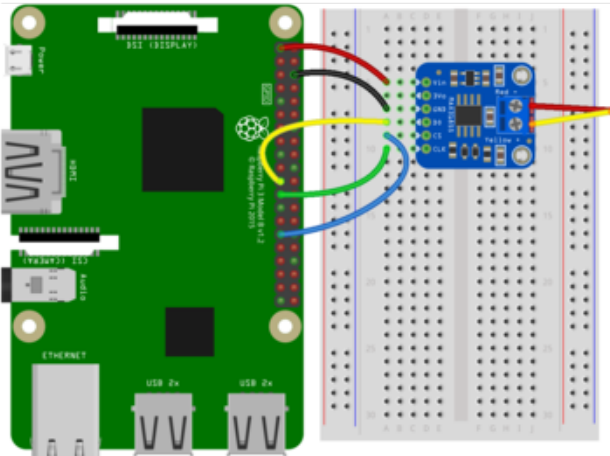


Board 3V to sensor Vdd
Board GND to sensor GND
Board SCK to sensor CLK
Board MISO to sensor DO
Board D5 to sensor CS (or any other free digital I/O pin)

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 SPI:



Pi 3.3V to sensor Vin
Pi GND to sensor GND
Pi SCLK to sensor CLK
Pi MISO to sensor DO
Pi GPIO 5 to sensor CS

CircuitPython Installation of MAX31855 Library

Next you'll need to install the [Adafruit CircuitPython MAX31855 \(\)](#) 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 \(\)](#). For example the Circuit Playground Express guide has [a great page on how to install the library bundle \(\)](#) for both express and non-express boards.

Remember for non-express boards like the Trinket M0, Gemma M0, and Feather/Metro M0 basic you'll need to manually install the necessary libraries from the bundle:

- adafruit_max31855.mpy
- adafruit_bus_device

Before continuing make sure your board's lib folder or root filesystem has the adafruit_max31855.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 MAX31855 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-max31855`

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 temperature. First initialize the SPI connection and library by running:

```
import board
import digitalio
import adafruit_max31855
spi = board.SPI()
cs = digitalio.DigitalInOut(board.D5)
max31855 = adafruit_max31855.MAX31855(spi, cs)
```

Now you can read the temperature property to retrieve the temperature from the sensor in degrees Celsius:

```
print('Temperature: {} degrees C'.format(max31855.temperature))
```

```
>>> print('Temperature: {} degrees C'.format(max31855.temperature))
Temperature: 21.5 degrees C
>>> █
```

That's all there is to reading temperature with the MAX31855 and CircuitPython code!

Full Example Code

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

import time
import board
import digitalio
import adafruit_max31855

spi = board.SPI()
cs = digitalio.DigitalInOut(board.D5)

max31855 = adafruit_max31855.MAX31855(spi, cs)
while True:
    tempC = max31855.temperature
    tempF = tempC * 9 / 5 + 32
    print("Temperature: {} C {} F ".format(tempC, tempF))
    time.sleep(2.0)
```

Python Docs

[Python Docs \(\)](#)

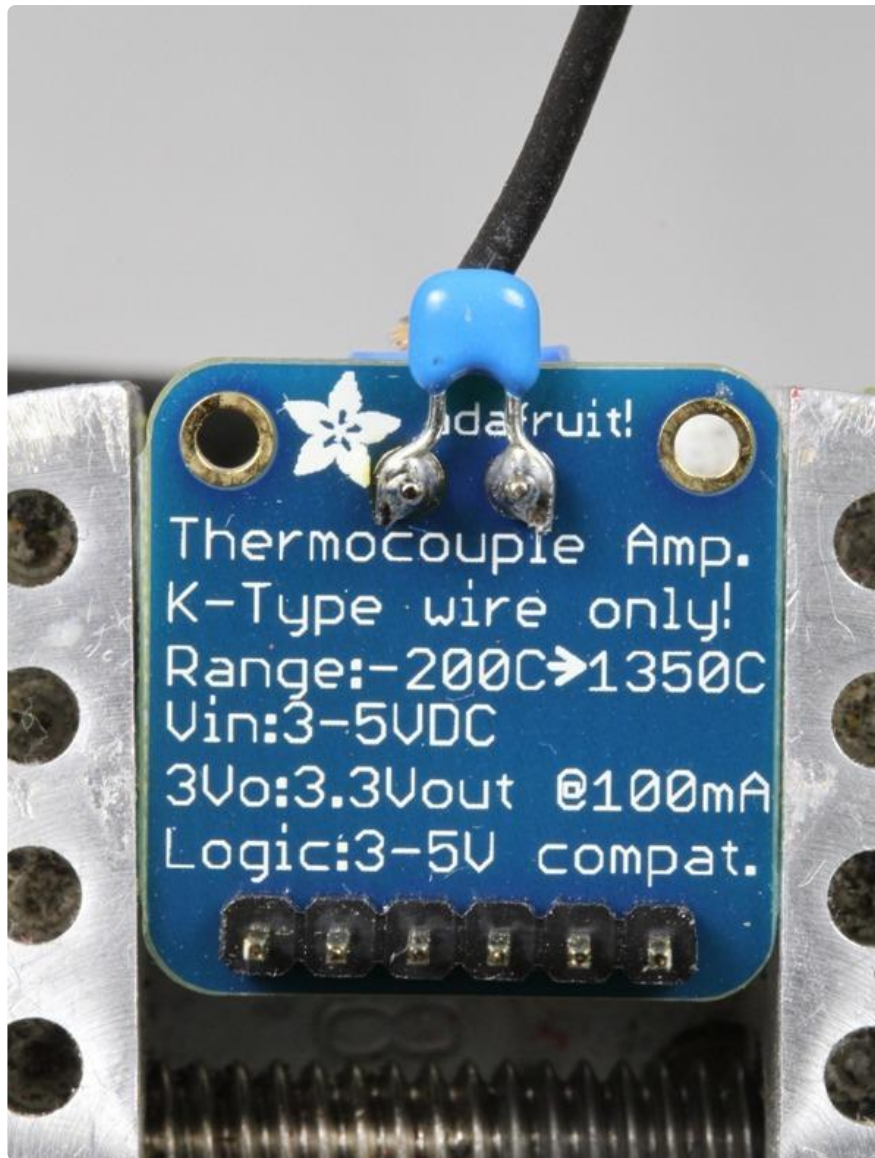
F.A.Q.

The temperature of my thermocouple seems to be backwards! If I heat up the probe the reported temperature goes down

This is likely caused by the thermocouple wires being labled incorrectly. Try swapping the two thermocouple leads, even if yellow and red wires are in the right slots - we've seen some thermocouple where the wire colors are wrong.

My MAX31855 output is really unstable and noisy - If I touch or move the probe the temperature output goes crazy

The MAX31855 is surprisingly sensitive, we've found a good way to fix this is to place a 0.01uF to 0.1uF capacitor across the thermocouple leads (that is, place the capacitor into the blue terminal block, or solder to the bottom as shown below).



My thermocouple output has an offset / I have multiple thermocouples and they don't 'agree' on a temperature

K thermocouples are not precision temperature measurement devices! There will be offsets & differences between thermocouples. Most thermocouple thermometers have the offset corrected in software which is what we suggest. See this guide for tips on calibration:

[Sensor Calibration \(\)](#)

For precision temperature measurement, we suggest a 1% Thermistor.

How can I connect multiple thermocouples?

You can connect as many MAX31855's as you have pins. Simply share the CLK and DO pins of all the breakouts and have a unique CS pin for each one

Then you can create new thermocouples using the following style:

```
Adafruit_MAX31855 thermocouple1(thermoCLK, thermoCS1, thermoDO);  
Adafruit_MAX31855 thermocouple2(thermoCLK, thermoCS2,  
thermoDO);  
Adafruit_MAX31855 thermocouple3(thermoCLK, thermoCS3,  
thermoDO);
```

You can also try having same CS and CLK pins but all different DO pins

```
Adafruit_MAX31855 thermocouple1(thermoCLK, thermoCS, thermoDO1);  
Adafruit_MAX31855 thermocouple2(thermoCLK, thermoCS,  
thermoDO2);  
Adafruit_MAX31855 thermocouple3(thermoCLK, thermoCS,  
thermoDO3);
```

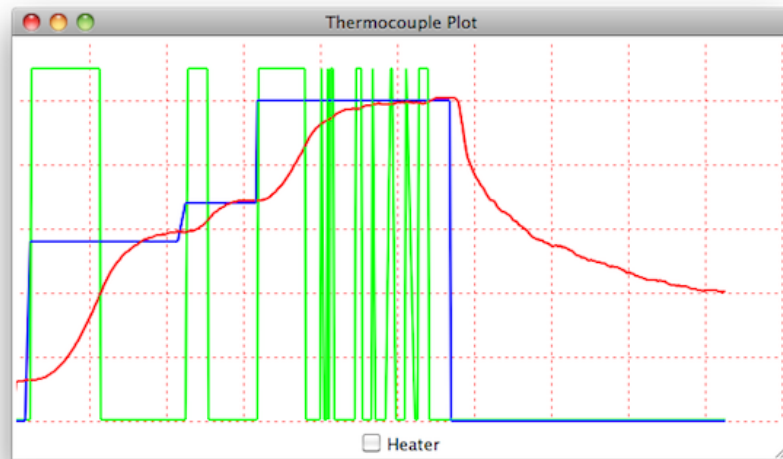
At extremely high or low temperatures, the measurements are not correct

The 31855 chip handles the linear range of the K-type thermocouples very well. It does not provide correction for the non-linearities that occur at the extremes of the measurement range. Thermocouple linearization for temperature extremes requires some curve fitting. See this guide for more information and example code:

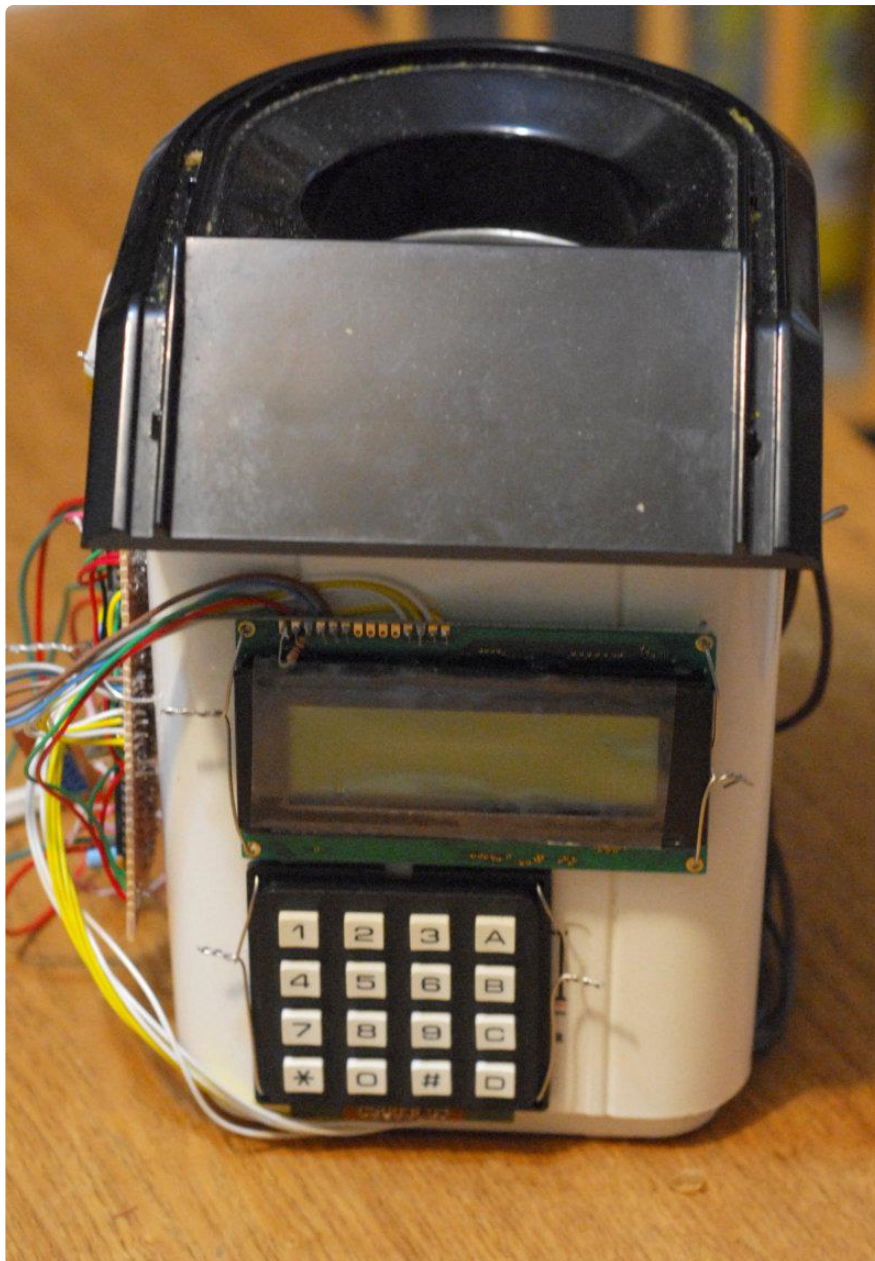
[Thermocouple Linearization \(\)](#)

Project Examples

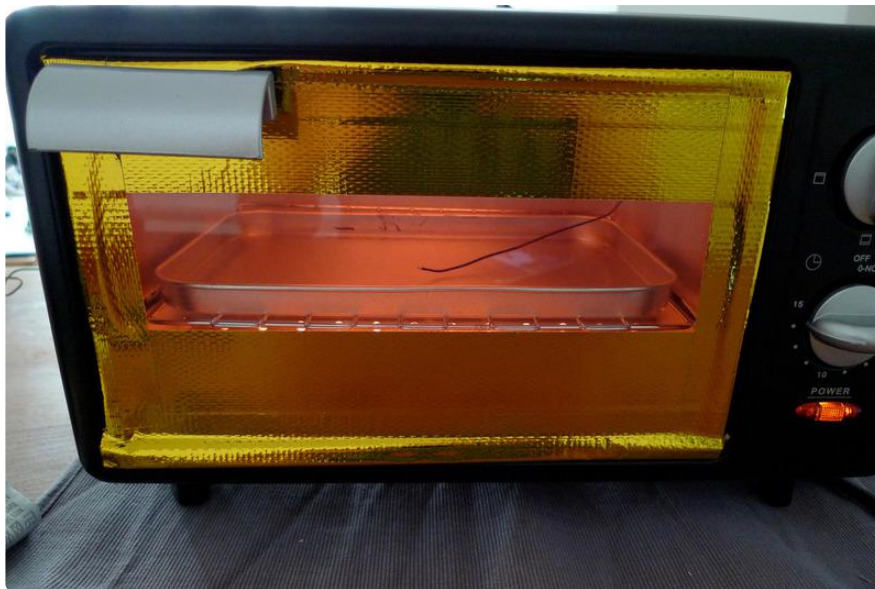
Need ideas? Check out these projects!



[Jeelabs has a detailed walkthrough for a reflow controller \(uses an AD595-type chip\) \(](#)
)



[Terran's PI controlled coffee roaster \(\)](#)



[RocketNumberNine's reflow toaster project \(\)](#)

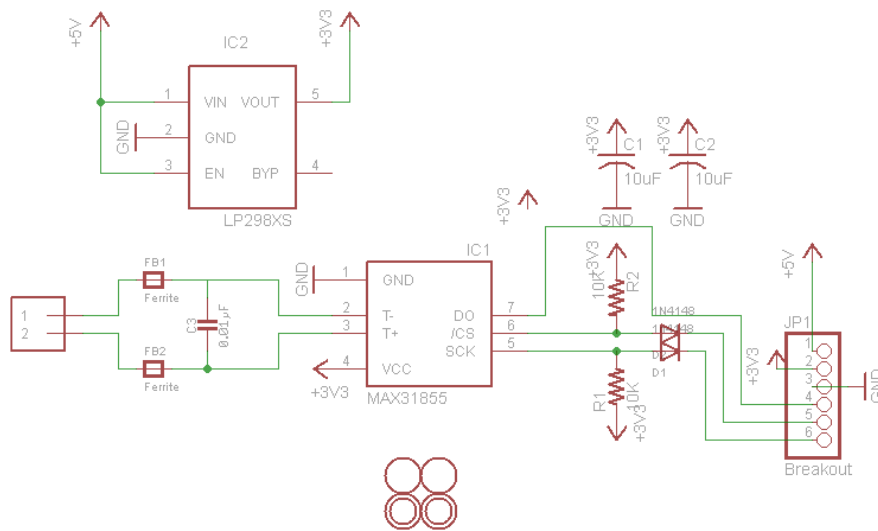
Downloads

Datasheets & Files

- [MAX31855 Datasheet \(\)](#)
- [MAX6675 Schematic and layout files can be found at GitHub \(\)](#)
- [MAX31855 Schematic and layout files can be found at GitHub \(\)](#)
- [Fritzing objects for both in the Adafruit Fritzing library \(\)](#)

Schematic

Click to embiggen



Fabrication Print

Dimensions in Inches

