

Adafruit MAX31856 Universal Thermocouple Amplifier Created by lady ada



Last updated on 2020-06-11 02:46:30 PM EDT

Overview





Thermocouples are very sensitive, requiring a good amplifier with a cold-compensation reference, as well as calculations to handle any non-linearities. For a long time we've suggested our MAX31855K breakout, which works great but is only for K-type thermocouples. Now we're happy to offer a great new thermocouple amplifier/converter that can handle just about *any* type of thermocouple, and even has the ability to give you notification when the temperature goes out of range, or a fault occurs. Very fancy!

This converter communicates over 4-wire SPI and can interface with any K, J, N, R, S, T, E, or B type thermocouple



This breakout does everything for you, and can be easily interfaced with any microcontroller, even one without an analog input. This breakout board has the chip itself, a 3.3V regulator and level shifting circuitry, all assembled and tested. Comes with a 2 pin terminal block (for connecting to the thermocouple) and pin header (to plug into any breadboard or perfboard). We even added inline resistors and a filter capacitor onboard for better stability, as recommended by Maxim. Goes great with our 1m K-type thermocouple (http://adafru.it/270) or any other thermocouple, really!

- Works with any K, J, N, R, S, T, E, or B type thermocouple
- -210°C to +1800°C output in 0.0078125° resolution note that many thermocouples have about ±2°C to ±6°C accuracy or worse depending on the temperature and type, so the resolution will be a lot better than the accuracy!
- Internal temperature reading
- 3.3 to 5v power supply and logic level compliant!
- SPI data requires any 4 digital I/O pins.





Pinouts



Power Pins:

- Vin this is the power pin. Since the sensor chip uses 3 VDC, we have included a voltage regulator on board 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

SPI Logic pins:

All pins going into the breakout have level shifting circuitry to make them 3-5V logic level safe. Use whatever logic level is on **Vin!**

- SCK This is the SPI Clock pin, its an input to the chip
- SDO this is the Serial Data Out / Microcontroller In Sensor Out pin, for data sent from the MAX31856 to your processor
- SDI this is the Serial Data In / Microcontroller Out Sensor In pin, for data sent from your processor to the MAX31856
- CS this is the Chip Select pin, drop it low to start an SPI transaction. Its an input to the chip

If you want to connect multiple MAX31856's to one microcontroller, have them share the SDI, SDO and SCK pins. Then assign each one a unique CS pin.

Additional Pins

There's two more pins that are available for advanced usage

• FLT - This is the Fault output. If you use the threshhold-notification capabilities of the MAX31856 you can monitor

this pin, when it goes low there's a fault!

• DRDY - This pin is used for advanced uses where you tell the sensor to begin a reading and then wait for this pin to go low. We don't use it in our library code because we keep it simple with a delay/wait, but it is available in case you need it!

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 5 pins for reliable electrical contact.

(For tips on soldering, be sure to check out ourGuide to Excellent Soldering (https://adafru.it/aTk))



Now you can do the terminal block, this is what you'll use to attach the thermocouple since you cannot solder to thermocouples

The terminal block goes on the top with the open ends pointing out



Solder the two pads as you did with the plain header. They're quite large and require a lot of solder



Insert the thermocouple wires and tighten down the clamps with a small Phillips or flat screwdriver

That's it! you are now ready to wire and test



Wiring & Test

You can easily wire this breakout to any microcontroller, we'll be using an Arduino. For another kind of microcontroller, as long as you have 4 available pins it is possible to 'bit-bang SPI' or you can use hardware SPI if you like. Just check out the library, then port the code.



SPI Wiring

Since this is a SPI-capable sensor, we can use hardware or 'software' SPI. To make wiring identical on all Arduinos, we'll begin with 'software' SPI. The following pins should be used:

- Connect Vin 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 SCK pin to Digital #13 but any pin can be used later
- Connect the SDO pin to Digital #12 but any pin can be used later
- Connect the **SDI** pin to **Digital #11** but any pin can be used later
- Connect the CS pin Digital #10 but any pin can be used later

Later on, once we get it working, we can adjust the library to use hardware SPI if you desire, or change the pins to other

Download Adafruit_MAX31856 library

To begin reading sensor data, you will need to install Adafruit MAX31856 from the Arduino library manager.

Open up the Arduino library manager:

💿 demo	Arduino 1.8.4			
File Edit	Sketch Tools Help			
⊘ €	Verify/Compile Upload	Ctrl+R Ctrl+U	 Manage Libraries	
// Demo	Upload Using Programmer Export compiled Binary	Ctrl+Shift+U Ctrl+Alt+S	Add .ZIP Library	cn oi
#includ	Show Sketch Folder	Ctrl+K	Arduino libraries	
#includ	Include Library		ArduinoHttpClient	
#INCIUG	Add File		ArduinoSound	
// we li	ght one pixel at a time,	this is our	Bridge	

Search for the Adafruit MAX31856 library and install it

Type All	▼ Topic All	▼ adafruit max31856	
Adafruit M Library for MAX31856 More info	IAX31856 library by Adafruit r the Adafruit Thermocouple Amplifi	ier breakout with MAX31856 Library for the	Adafruit Thermocouple Amplifier breakout with
			Version 1.0.3 🔻 Install

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

Attach Thermocouple

You'll need to attach a thermocouple, for this demo we'll be using a K-type but you can adjust the demo if you do not have a K-type handy!



Open up File->Examples->Adafruit_MAX31856->max31856 and upload to your Arduino wired up to the sensor. Adjust the max.setThermocoupleType(MAX31856_TCTYPE_K) line if necessary.

😳 max31856 Arduino 1.6.9						
File	Edit Sketch	Tools Help				
	New	Ctrl+N				
	Open	Ctrl+0				
	Open Recent	I				
	Sketchbook	I	• .h>	•		1
	Examples	I		*		
	Close Save	Ctrl+W Ctrl+S		Adafruit_MAX21000 Adafruit_MAX21100 Adafruit_MAX30100	+ +	
	Save AS	Cur+Smit+S		Adafruit_MAX31856	- 1	max31856
	Page Setup Print	Ctrl+Shift+P Ctrl+P		Adafruit_MCP23008 Adafruit_MiniMLX9014) 	
	Preferences	Ctrl+Comma		Adafruit_MLX9014 Adafruit_MMA8451	+	
L_	Quit	Ctrl+Q		Adafruit_MPL115A2 Adafruit_MPU9150	+	

Upload to your Arduino and open up the serial console at 115200 baud to see a print out of the *cold junction temperature* (temperature of the microcontroller chip) and the *thermocouple temperature* (temperature detected at the end of the thermocouple probe

0	© COM68	
[Send	
MTCTCTCTCTC	AX31856 thermocouple test hermocouple type: K Type cold Junction Temp: 26.31 hermocouple Temp: 29.77 cold Junction Temp: 26.31 hermocouple Temp: 29.85 cold Junction Temp: 26.30 hermocouple Temp: 29.88 cold Junction Temp: 26.30 hermocouple Temp: 29.94 cold Junction Temp: 26.30 hermocouple Temp: 29.54 cold Junction Temp: 29.54 cold Junction Temp:	
[✓ Autoscroll Ø Autoscroll Both NL & CR → 115200 bauc	- 1

You can also see some of the faults that are detectable by say disconnecting one of the pins:

© COM68	
	Send
Thermocouple Temp: 26.16	
Cold Junction Temp: 26.19	
Thermocouple Temp: 26.05	
Cold Junction Temp: 26.19	
Thermocouple Temp: 1372.05	
Thermocouple Range Fault	
Thermocouple Open Fault	
Cold Junction Temp: 26.22	
Thermocouple Temp: 1372.05	
Thermocouple Range Fault	
Thermocouple Open Fault	
Cold Junction Temp: 26.22	
Thermocouple Temp: 1372.05	
Thermocouple Range Fault	E
Thermocouple Open Fault	
Cold Junction Temp: 26.23	-
Autoscroll	Both NL & CR 👻 115200 baud

Library Reference

You can start out by creating a MAX31856 object with either software SPI (where all four pins can be any I/O) using

```
// Use software SPI: CS, DI, DO, CLK
Adafruit_MAX31856 max = Adafruit_MAX31856(10, 11, 12, 13);
```

Or you can use hardware SPI. With hardware SPI you *must* use the hardware SPI pins for your Arduino - and each arduino type has different pins! Check the SPI reference to see what pins to use. (https://adafru.it/d5h) In this case, you can use any CS pin, but the other three pins are fixed

```
// use hardware SPI, just pass in the CS pin
Adafruit_MAX31856 max = Adafruit_MAX31856(10);
```

Once started, you can initialize the sensor with

```
max.begin()
```

You'll also need to set the thermocouple type, remember there's a lot of options! Set the type with:

max.setThermocoupleType(MAX31856_TCTYPE_xxx)

Your options for the *TCTYPE* are:

- MAX31856_TCTYPE_B
- MAX31856_TCTYPE_E
- MAX31856_TCTYPE_J
- MAX31856_TCTYPE_K
- MAX31856_TCTYPE_N
- MAX31856_TCTYPE_R

- MAX31856_TCTYPE_S
- MAX31856_TCTYPE_T
- MAX31856_VMODE_G8
- MAX31856 VMODE G32

The last two are not thermocouple types, they're just 'plain' voltage readings (check the datasheet for more details, we don't use these modes in the library)

If you're ever not sure which mode you're in, query it with

max.getThermocoupleType()

Once that's set you can read the cold junction temperature, which will return a floating point Celsius reading. This is the temperature detected inside the MAX31856 chip ('ambient' temp)

max.readCJTemperature()

Or, of course, the temperature at the end/tip of the thermocouple, likewise a floating point #

max.readThermocoupleTemperature()

Faults

The MAX31856 has a wide-ranging fault mechanism that can alert you via pin or function when something is amiss. Don't forget to test this functionality before relying on it!

You can read faults with

```
max.readFault()
```

Which will return a uint8_t type with bits set for each of 8 different fault types. You can test for each one with this set of code:

```
uint8_t fault = max.readFault();
if (fault) {
    if (fault & MAX31856_FAULT_CJRANGE) Serial.println("Cold Junction Range Fault");
    if (fault & MAX31856_FAULT_TCRANGE) Serial.println("Thermocouple Range Fault");
    if (fault & MAX31856_FAULT_CJHIGH) Serial.println("Cold Junction High Fault");
    if (fault & MAX31856_FAULT_CJLOW) Serial.println("Cold Junction Low Fault");
    if (fault & MAX31856_FAULT_TCHIGH) Serial.println("Thermocouple High Fault");
    if (fault & MAX31856_FAULT_TCHIGH) Serial.println("Thermocouple High Fault");
    if (fault & MAX31856_FAULT_TCLOW) Serial.println("Thermocouple Low Fault");
    if (fault & MAX31856_FAULT_OVUV) Serial.println("Over/Under Voltage Fault");
    if (fault & MAX31856_FAULT_OPEN) Serial.println("Thermocouple Open Fault");
}
```

The last two faults are built in. For the low/high threshholds, you can set those with two functions.

For the cold junction (chip) temp, use:

Where *lowtemp* and *hightemp* range between -127 and +127 Centigrade (the chip wont function down to -127 but that's the lowest number you can put in.

For the thermocouple, use

setTempFaultThreshholds(lowtemp, hightemp)

Where lowtemp and hightemp are floating point numbers with a range of -4096 to +4096 and a resolution of 0.0625 degrees Centigrade





Python & CircuitPython

It's easy to use the MAX31856 sensor with Python and CircuitPython, and the Adafruit CircuitPython MAX31856 (https://adafru.it/Gao) 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 (https://adafru.it/BSN).

CircuitPython Microcontroller Wiring

First, wire up a MAX31856 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 VIN
- Board GND to sensor GND
- Board SCK to sensor SCK
- Board MISO to sensor SDO
- Board MOSI to sensor SDI
- Board D5 to sensor CS (or any other free digital I/O pin)

Once wired to the microcontroller, make sure you connect a thermocouple to the terminal on the breakout board.

Python Computer Wiring

Since there's *dozens* of Linux computers/boards you can use, this shows 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).

Here's the Raspberry Pi wired with SPI:



- Pi 3V to sensor VIN
- Pi GND to sensor GND
- Pi SCK to sensor SCK
- Pi MISO to sensor SDO
- Pi MOSI to sensor SDI
- Pi D5 to sensor CS (or any other free digital I/O pin)

Once wired to the Raspberry Pi, make sure you connect a thermocouple to the terminal on the breakout board.

CircuitPython Installation of MAX31856 Library

Next, you'll need to install the Adafruit CircuitPython MAX31856 (https://adafru.it/Gao) library on your CircuitPython board

First, make sure you are running the latest version of Adafruit CircuitPython (https://adafru.it/tBa) 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 (https://adafru.it/ENC). For example the Circuit Playground Express guide has a great page on how to install the library bundle (https://adafru.it/Bf2) for both Express and non-Express boards.

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

- adafruit_max31856.mpy
- adafruit_bus_device

Before continuing, make sure your board's lib folder has the adafruit_max31856.mpy, and adafruit_bus_device files and folders copied over.

Next connect to the board's serial REPL (https://adafru.it/Awz) so you are at the CircuitPython >>> prompt.

Python Installation of MAX31856 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)!

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

• sudo pip3 install adafruit-circuitpython-max31856

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, initialize it and read the temperature. Remember to attach a thermocouple to the terminal on the breakout!

First initialize the SPI connection and library by running:

```
import board
import busio
import digitalio
import adafruit_max31856
spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
cs = digitalio.DigitalInOut(board.D5)
cs.direction = digitalio.Direction.OUTPUT
thermocouple = adafruit_max31856.MAX31856(spi,cs)
```

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

print(thermocouple.temperature)

>>> print(thermocouple.temperature)
25,125

That's all there is to reading temperature with the MAX31856 and CircuitPython code!

Full Example Code

```
import board
import busio
import digitalio
import adafruit_max31856
# create a spi object
spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
# allocate a CS pin and set the direction
cs = digitalio.DigitalInOut(board.D0)
cs.direction = digitalio.Direction.OUTPUT
# create a thermocouple object with the above
thermocouple = adafruit_max31856.MAX31856(spi, cs)
# print the temperature!
print(thermocouple.temperature)
```

Python Docs

Python Docs (https://adafru.it/G7C)





Downloads

Files

- Fritzing object in Adafruit Fritzing library (https://adafru.it/c7M)
- EagleCAD PCB files on GitHub (https://adafru.it/rAN)
- Library on GitHub (https://adafru.it/rAL)
- MAX31856 Datasheet (https://adafru.it/rAO)

Schematic



Fabrication Print







Общество с ограниченной ответственностью «МосЧип» ИНН 7719860671 / КПП 771901001 Адрес: 105318, г.Москва, ул.Щербаковская д.З, офис 1107

Данный компонент на территории Российской Федерации

Вы можете приобрести в компании MosChip.

Для оперативного оформления запроса Вам необходимо перейти по данной ссылке:

http://moschip.ru/get-element

Вы можете разместить у нас заказ для любого Вашего проекта, будь то серийное производство или разработка единичного прибора.

В нашем ассортименте представлены ведущие мировые производители активных и пассивных электронных компонентов.

Нашей специализацией является поставка электронной компонентной базы двойного назначения, продукции таких производителей как XILINX, Intel (ex.ALTERA), Vicor, Microchip, Texas Instruments, Analog Devices, Mini-Circuits, Amphenol, Glenair.

Сотрудничество с глобальными дистрибьюторами электронных компонентов, предоставляет возможность заказывать и получать с международных складов практически любой перечень компонентов в оптимальные для Вас сроки.

На всех этапах разработки и производства наши партнеры могут получить квалифицированную поддержку опытных инженеров.

Система менеджмента качества компании отвечает требованиям в соответствии с ГОСТ Р ИСО 9001, ГОСТ РВ 0015-002 и ЭС РД 009

Офис по работе с юридическими лицами:

105318, г.Москва, ул.Щербаковская д.З, офис 1107, 1118, ДЦ «Щербаковский»

Телефон: +7 495 668-12-70 (многоканальный)

Факс: +7 495 668-12-70 (доб.304)

E-mail: info@moschip.ru

Skype отдела продаж: moschip.ru moschip.ru_4

moschip.ru_6 moschip.ru_9