

---

# **AdafruitMAX31856 Library Documentation**

***Release 1.0***

**Bryan Siepert**

**Oct 19, 2020**



---

## Contents

---

<b>1</b>	<b>Dependencies</b>	<b>3</b>
<b>2</b>	<b>Installing from PyPI</b>	<b>5</b>
<b>3</b>	<b>Usage Example</b>	<b>7</b>
<b>4</b>	<b>Contributing</b>	<b>9</b>
<b>5</b>	<b>Documentation</b>	<b>11</b>
<b>6</b>	<b>Table of Contents</b>	<b>13</b>
6.1	Simple test . . . . .	13
6.2	MAX31856 . . . . .	13
6.2.1	Implementation Notes . . . . .	14
<b>7</b>	<b>Indices and tables</b>	<b>17</b>
	<b>Python Module Index</b>	<b>19</b>
	<b>Index</b>	<b>21</b>



A CircuitPython driver for the MAX31856 Universal Thermocouple Amplifier



# CHAPTER 1

---

## Dependencies

---

This driver depends on:

- [Adafruit CircuitPython](#)
- [Bus Device](#)

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the [Adafruit library and driver bundle](#).





## CHAPTER 2

---

### Installing from PyPI

---

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally [from PyPI](#). To install for current user:

```
pip3 install adafruit-circuitpython-max31856
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-max31856
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-max31856
```



## CHAPTER 3

---

### Usage Example

---

```
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.D5)
cs.direction = digitalio.Direction.OUTPUT

# create a thermocouple object with the above
thermocouple = adafruit_max31856.MAX31856(spi, cs)

# print the temperature!
print(thermocouple.temperature)
```



## CHAPTER 4

---

### Contributing

---

Contributions are welcome! Please read our [Code of Conduct](#) before contributing to help this project stay welcoming.



## CHAPTER 5

---

### Documentation

---

For information on building library documentation, please check out [this guide](#).





### 6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/max31856\_simpletest.py

```
1 import board
2 import busio
3 import digitalio
4 import adafruit_max31856
5
6 # create a spi object
7 spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
8
9 # allocate a CS pin and set the direction
10 cs = digitalio.DigitalInOut(board.D5)
11 cs.direction = digitalio.Direction.OUTPUT
12
13 # create a thermocouple object with the above
14 thermocouple = adafruit_max31856.MAX31856(spi, cs)
15
16 # print the temperature!
17 print(thermocouple.temperature)
```

### 6.2 MAX31856

CircuitPython module for the MAX31856 Universal Thermocouple Amplifier. See examples/simpletest.py for an example of the usage.

- Author(s): Bryan Siepert

## 6.2.1 Implementation Notes

### Hardware:

- Adafruit [Universal Thermocouple Amplifier MAX31856 Breakout](#) (Product ID: 3263)

### Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: <https://github.com/adafruit/circuitpython/releases>
- Adafruit's Bus Device library: [https://github.com/adafruit/Adafruit\\_CircuitPython\\_BusDevice](https://github.com/adafruit/Adafruit_CircuitPython_BusDevice)

**class** adafruit\_max31856.**MAX31856** (*spi*, *cs*, *thermocouple\_type*=3)

Driver for the MAX31856 Universal Thermocouple Amplifier

#### Parameters

- **spi\_bus** (*SPI*) – The SPI bus the MAX31856 is connected to.
- **cs** (*Pin*) – The pin used for the CS signal.
- **thermocouple\_type** (*ThermocoupleType*) – The type of thermocouple. Default is Type K.

#### **fault**

A dictionary with the status of each fault type where the key is the fault type and the value is a bool if the fault is currently active

Key	Fault type
"cj_range"	Cold junction range fault
"tc_range"	Thermocouple range fault
"cj_high"	Cold junction high threshold fault
"cj_low"	Cold junction low threshold fault
"tc_high"	Thermocouple high threshold fault
"tc_low"	Thermocouple low threshold fault
"voltage"	Over/under voltage fault
"open_tc"	Thermocouple open circuit fault

#### **reference\_temperature**

The temperature of the cold junction in degrees celsius. (read-only)

#### **reference\_temperature\_thresholds**

The cold junction's low and high temperature thresholds as a (low\_temp, high\_temp) tuple

#### **temperature**

The temperature of the sensor and return its value in degrees celsius. (read-only)

#### **temperature\_thresholds**

The thermocouple's low and high temperature thresholds as a (low\_temp, high\_temp) tuple

**class** adafruit\_max31856.**ThermocoupleType**

An enum-like class representing the different types of thermocouples that the MAX31856 can use. The values can be referenced like `ThermocoupleType.K` or `ThermocoupleType.S`. Possible values are

- `ThermocoupleType.B`
- `ThermocoupleType.E`
- `ThermocoupleType.J`
- `ThermocoupleType.K`
- `ThermocoupleType.N`

- ThermocoupleType.R
- ThermocoupleType.S
- ThermocoupleType.T



## CHAPTER 7

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### a

adafruit\_max31856, [13](#)





## A

`adafruit_max31856` (*module*), [13](#)

## F

`fault` (*adafruit\_max31856.MAX31856 attribute*), [14](#)

## M

`MAX31856` (*class in adafruit\_max31856*), [14](#)

## R

`reference_temperature`  
(*adafruit\_max31856.MAX31856 attribute*),  
[14](#)

`reference_temperature_thresholds`  
(*adafruit\_max31856.MAX31856 attribute*),  
[14](#)

## T

`temperature` (*adafruit\_max31856.MAX31856 attribute*), [14](#)

`temperature_thresholds`  
(*adafruit\_max31856.MAX31856 attribute*),  
[14](#)

`ThermocoupleType` (*class in adafruit\_max31856*), [14](#)