

---

# **Adafruit MAX31865 Library Documentation**

*Release 1.0*

**Tony DiCola**

**Apr 10, 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	adafruit_max31865 .....	14
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>



CircuitPython module for the MAX31865 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-max31865
```

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

```
sudo pip3 install adafruit-circuitpython-max31865
```

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-max31865
```



## CHAPTER 3

---

### Usage Example

---

See `examples/max31865_simpletest.py` for a demo of the usage.



## 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/max31865\_simpletest.py

```
1  # Simple demo of the MAX31865 thermocouple amplifier.
2  # Will print the temperature every second.
3  import time
4
5  import board
6  import busio
7  import digitalio
8
9  import adafruit_max31865
10
11
12 # Initialize SPI bus and sensor.
13 spi = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
14 cs = digitalio.DigitalInOut(board.D5) # Chip select of the MAX31865 board.
15 sensor = adafruit_max31865.MAX31865(spi, cs)
16 # Note you can optionally provide the thermocouple RTD nominal, the reference
17 # resistance, and the number of wires for the sensor (2 the default, 3, or 4)
18 # with keyword args:
19 # sensor = adafruit_max31865.MAX31865(spi, cs, rtd_nominal=100, ref_resistor=430.0,
20 # ↪wires=2)
21
22 # Main loop to print the temperature every second.
23 while True:
24     # Read temperature.
25     temp = sensor.temperature
26     # Print the value.
27     print("Temperature: {0:0.3f}C".format(temp))
```

(continues on next page)

```

27 # Delay for a second.
28 time.sleep(1.0)

```

## 6.2 adafruit\_max31865

CircuitPython module for the MAX31865 platinum RTD temperature sensor. See examples/simpletest.py for an example of the usage.

- Author(s): Tony DiCola

### 6.2.1 Implementation Notes

#### Hardware:

- Adafruit Universal Thermocouple Amplifier MAX31856 Breakout (Product ID: 3263)
- Adafruit PT100 RTD Temperature Sensor Amplifier - MAX31865 (Product ID: 3328)
- Adafruit PT1000 RTD Temperature Sensor Amplifier - MAX31865 (Product ID: 3648)

#### Software and Dependencies:

- Adafruit CircuitPython firmware for the ESP8622 and M0-based 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_max31865.MAX31865` (*spi, cs, \*, rtd\_nominal=100, ref\_resistor=430.0, wires=2, filter\_frequency=60*)

Driver for the MAX31865 thermocouple amplifier.

#### **auto\_convert**

The state of the sensor's automatic conversion mode (True/False).

#### **bias**

The state of the sensor's bias (True/False).

#### **clear\_faults ()**

Clear any fault state previously detected by the sensor.

#### **fault**

The fault state of the sensor. Use `clear_faults ()` to clear the fault state. Returns a 6-tuple of boolean values which indicate if any faults are present:

- HIGHTHRESH
- LOWTHRESH
- REFINLOW
- REFINHIGH
- RTDINLOW
- OVUV

#### **read\_rtd ()**

Perform a raw reading of the thermocouple and return its 15-bit value. You'll need to manually convert this to temperature using the nominal value of the resistance-to-digital conversion and some math. If you just want temperature use the temperature property instead.

**resistance**

Read the resistance of the RTD and return its value in Ohms.

**temperature**

Read the temperature of the sensor and return its value in degrees Celsius.



# CHAPTER 7

---

## Indices and tables

---

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



**a**

`adafruit_max31865`, 14





## A

`adafruit_max31865` (*module*), 14  
`auto_convert` (*adafruit\_max31865.MAX31865 attribute*), 14

## B

`bias` (*adafruit\_max31865.MAX31865 attribute*), 14

## C

`clear_faults()` (*adafruit\_max31865.MAX31865 method*), 14

## F

`fault` (*adafruit\_max31865.MAX31865 attribute*), 14

## M

`MAX31865` (*class in adafruit\_max31865*), 14

## R

`read_rtd()` (*adafruit\_max31865.MAX31865 method*), 14  
`resistance` (*adafruit\_max31865.MAX31865 attribute*), 15

## T

`temperature` (*adafruit\_max31865.MAX31865 attribute*), 15