
Adafruit MAX31865 Library Documentation

Release 1.0

Tony DiCola

Dec 21, 2018

Contents

1	Dependencies	3
2	Usage Example	5
3	Contributing	7
4	Building locally	9
4.1	Sphinx documentation	9
5	Table of Contents	11
5.1	Simple test	11
5.2	adafruit_max31865	12
5.2.1	Implementation Notes	12
6	Indices and tables	15
	Python Module Index	17

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

Usage Example

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

CHAPTER 3

Contributing

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

CHAPTER 4

Building locally

To build this library locally you'll need to install the `circuitpython-build-tools` package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-max31865 --
↳library_location .
```

4.1 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the `index.html` in your browser to view them. It will also (due to `-W`) error out on any warning like Travis will. This is a good way to locally verify it will pass.

5.1 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)

```

5.2 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

5.2.1 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

class `adafruit_max31865.MAX31865` (*spi*, *cs*, *, *rtd_nominal*=100, *ref_resistor*=430.0, *wires*=2)
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 6

Indices and tables

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

a

[adafruit_max31865](#), 12

A

adafruit_max31865 (module), [12](#)
auto_convert (adafruit_max31865.MAX31865 attribute),
[12](#)

B

bias (adafruit_max31865.MAX31865 attribute), [12](#)

C

clear_faults() (adafruit_max31865.MAX31865 method),
[12](#)

F

fault (adafruit_max31865.MAX31865 attribute), [12](#)

M

MAX31865 (class in adafruit_max31865), [12](#)

R

read_rtd() (adafruit_max31865.MAX31865 method), [12](#)
resistance (adafruit_max31865.MAX31865 attribute), [12](#)

T

temperature (adafruit_max31865.MAX31865 attribute),
[13](#)