AdafruitMAX31856 Library Documentation

Release 1.0

Bryan Siepert

Oct 21, 2019

Contents

1 Dependencies **3**

2 Usage Example **5**

3 Contributing **7**

4 Building locally **9**
4.1 Zip release files 9
4.2 Sphinx documentation 9

5 Table of Contents **11**
5.1 Simple test 11
5.2 `MAX31856` 11
5.2.1 Implementation Notes 12

6 Indices and tables **15**

Python Module Index **17**

Index **19**

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

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 3

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 4

Building locally

4.1 Zip release files

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-max31856 --
↪library_location .
```

4.2 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.

CHAPTER 5

Table of Contents

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/max31856_simpletest.py

```
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)
```

5.2 MAX31856

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

- Author(s): Bryan Siepert

5.2.1 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

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 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

adafruit_max31856, 11

Index

A

adafruit_max31856 (*module*), 11

F

fault (*adafruit_max31856.MAX31856 attribute*), 12

M

MAX31856 (*class in adafruit_max31856*), 12

R

reference_temperature (*adafruit_max31856.MAX31856 attribute*), 12

reference_temperature_thresholds (*adafruit_max31856.MAX31856 attribute*), 12

T

temperature (*adafruit_max31856.MAX31856 attribute*), 12

temperature_thresholds (*adafruit_max31856.MAX31856 attribute*), 12

ThermocoupleType (*class in adafruit_max31856*), 12