Adafruit MCP9808 Library Documentation

Release 1.0

Phiilip Moyer

May 21, 2021

Contents

The MCP9808 is an awesome, high accuracy temperature sensor that communicates over I2C. Its available on Adafruit as a breakout.

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-mcp9808
```

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

```
sudo pip3 install adafruit-circuitpython-mcp9808
```

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-mcp9808
```

CHAPTER 3

Usage Notes

Getting the temperature in Celsius is easy! First, import all of the pins from the board, board.I2C() for native I2C communication and the thermometer library itself.

```
import board
import adafruit_mcp9808
```

Next, initialize the I2C bus in a with statement so it always gets shut down ok. Then, construct the thermometer class:

```
# Do one reading
with i2c = board.I2C() as i2c:
    t = adafruit_mcp9808.MCP9808(i2c)

    # Finally, read the temperature property and print it out
    print(t.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.

CHAPTER 6

Table of Contents

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/mcp9808_simpletest.py

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
import adafruit_mcp9808

i2c = board.I2C()  # uses board.SCL and board.SDA

# To initialise using the default address:
mcp = adafruit_mcp9808.MCP9808(i2c)

# To initialise using a specified address:
# Necessary when, for example, connecting A0 to VDD to make address=0x19
# mcp = adafruit_mcp9808.MCP9808(i2c_bus, address=0x19)

while True:
    tempC = mcp.temperature
    tempF = tempC * 9 / 5 + 32
    print("Temperature: {} C {} F ".format(tempC, tempF))
    time.sleep(2)
```

6.2 Temperature Limit test

Show the MCP9808 to setup different temperature values

Listing 2: examples/mcp9808_temperature_limits.py

```
# SPDX-FileCopyrightText: 2021 Jose David M.
# SPDX-License-Identifier: MIT

"""
Show the MCP9808 to setup different temperature values
"""

import time
import board
import adafruit_mcp9808

i2c = board.I2C()  # uses board.SCL and board.SDA
mcp = adafruit_mcp9808.MCP9808(i2c)

# Change the values according to the desired values
print("Setting Temperature Limits")
mcp.upper_temperature = 23
mcp.lower_temperature = 10
mcp.critical_temperature = 100

# To verify the limits we need to read the temperature value
print(mcp.temperature)
time.sleep(0.3)  # This is the time temperature conversion at maximum resolution

# Showing temperature Limits
while True:
    if mcp.below_lt:
        print("too cold!")
    if mcp.above_ut:
        print("getting hot!")
    if mcp.above_ct:
        print("Above critical temp!")
```

6.3 adafruit_mcp9808

CircuitPython library to support MCP9808 high accuracy temperature sensor.

- Author(s): Scott Shawcroft, Jose David M.

6.3.1 Implementation Notes

Hardware:

- Adafruit MCP9808 High Accuracy I2C Temperature Sensor Breakout (Product ID: 1782)

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
- Adafruit's Register library: https://github.com/adafruit/Adafruit_CircuitPython_Register

Notes:

1. Datasheet: http://www.adafruit.com/datasheets/MCP9808.pdf

class adafruit_mcp9808.**MCP9808**(*i2c_bus*, *address=24*)
Interface to the MCP9808 temperature sensor.

Parameters

- **i2c_bus** (*I2C*) – The I2C bus the MCP9808 is connected to.
- **address** (*int*) – The I2C address of the device. Defaults to `0x18`

MCP9808 Settings You could set the MCP9808 with different temperature limits and compare them with the ambient temperature Ta

- above_ct this value will be set to `True` when Ta is above this limit
- above_ut: this value will be set to `True` when Ta is above this limit
- below_lt: this value will be set to `True` when Ta is below this limit

To get this value, you will need to read the temperature, and then access the attribute

Quickstart: Importing and using the MCP9808

Here is an example of using the *MCP9808* class. First you will need to import the libraries to use the sensor

```
import board
import adafruit_mcp9808
```

Once this is done you can define your `board.I2C` object and define your sensor object

```
i2c = board.I2C()   # uses board.SCL and board.SDA
mcp = adafruit_mcp9808.MCP9808(i2c_bus)
```

Now you have access to the change in temperature using the *temperature* attribute. This temperature is in Celsius.

```
temperature = mcp.temperature
```

above_critical
True when the temperature is above the currently set critical temperature. False Otherwise

above_upper
True when the temperature is above the currently set high temperature. False Otherwise

below_lower
True when the temperature is below the currently set lower temperature. False Otherwise

critical_temperature
Critical alarm temperature in Celsius

lower_temperature
Lower alarm temperature in Celsius

resolution
Temperature Resolution in Celsius

Value	Resolution	Reading Time
0	0.5°C	30 ms
1	0.25°C	65 ms
2	0.125°C	130 ms
3	0.0625°C	250 ms

temperature
Temperature in Celsius. Read-only.

upper_temperature
Upper alarm temperature in Celsius

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index