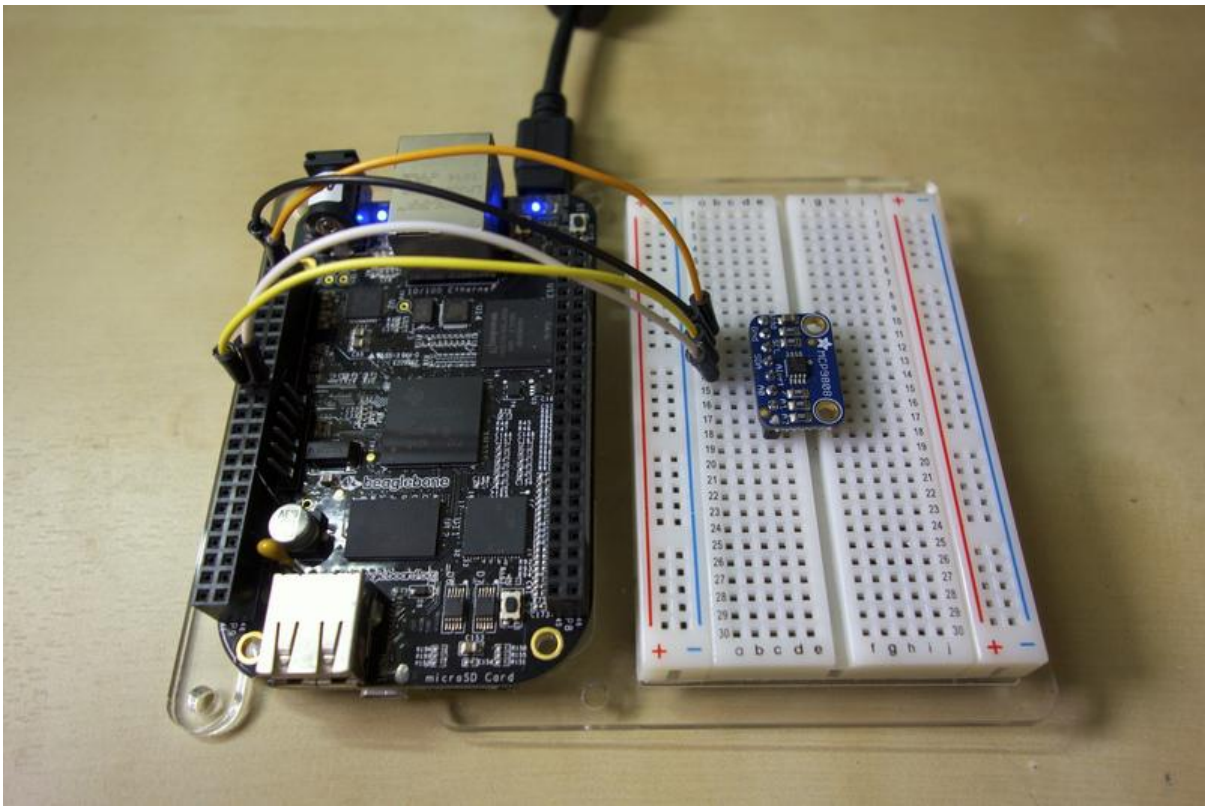
MCP9808 Temperature Sensor Python Library

Created by Tony DiCola



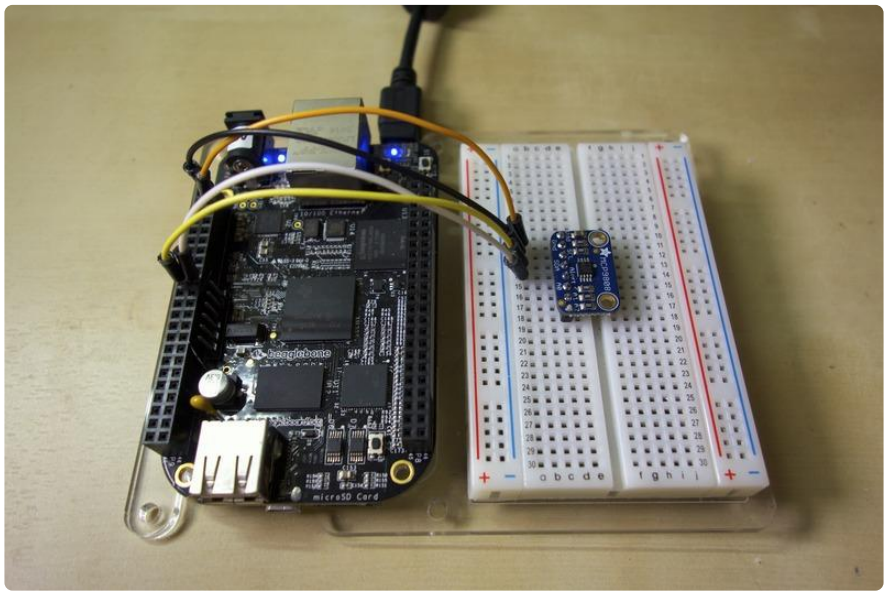
https://learn.adafruit.com/mcp9808-temperature-sensor-python-library

Last updated on 2021-11-15 06:18:19 PM EST

Table of Contents

Overview

This library and guide has been deprecated. We have a newer, better library and guide here https://learn.adafruit.com/adafruit-mcp9808-precision-i2c-temperature-sensor-guide/



Do you need accurate temperature measurements in a project? Consider using the MCP9808 precision temperature sensor (https://adafru.it/e06) with up to 0.25 degrees Celsius accuracy! The MCP9808 will read temperature and make it available over an I2C connection. With the MCP9808 Python library you can now use the MCP9808 precision temperature sensor with your Raspberry Pi or BeagleBone Black project!

Before you get started make sure your Raspberry Pi is running the latest Raspbian (https://adafru.it/dpb) or Occidentalis (https://adafru.it/dvg) operating system, and your BeagleBone Black is running the latest official Debian operating system (https://adafru.it/dUl). It will also help to familiarize yourself with the MCP9808 sensor by reading its Arduino guide (https://adafru.it/uan).

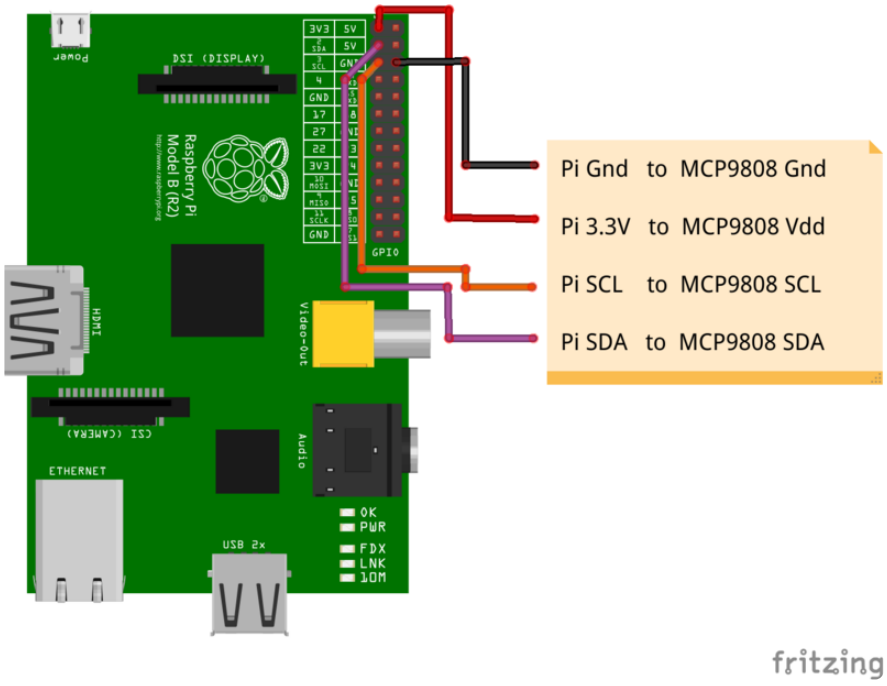
Hardware

This library and guide has been deprecated. We have a newer, better library and guide here https://learn.adafruit.com/adafruit-mcp9808-precision-i2c-temperature-sensor-guide/

Wiring the MCP9808 to a Raspberry Pi or BeagleBone Black is easy because the board only uses an I2C bus for communication.

Raspberry Pi

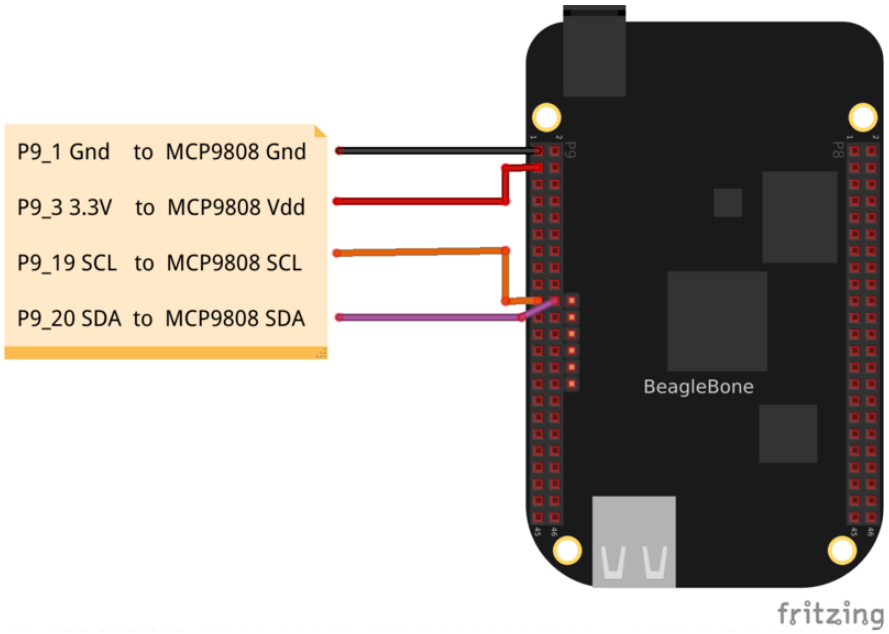
Connect the MCP9808 to a Raspberry Pi as follows. Note that you must have I2C enabled on your Raspberry Pi (https://adafru.it/dEO).



- Connect Pi GND to MCP9808 Gnd.
- Connect Pi 3.3V to MCP9808 Vdd.
- Connect Pi SCL to MCP9808 SCL.
- Connect Pi SDA to MCP9808 SDA.

BeagleBone Black

Connect the MCP9808 to a BeagleBone Black as follows. If you aren't familiar with how GPIO pins are numbered, check out this guide on BeagleBone Black GPIO (https://adafru.it/dCl).



- Connect BeagleBone Black P9_1 DGND to MCP9808 Gnd.
- Connect BeagleBone Black P9_3 3.3V to MCP9808 Vdd.
- Connect BeagleBone Black P9_19 I2C SCL to MCP9808 SCL.
- Connect BeagleBone Black P9_20 I2C SDA to MCP9808 SDA.

Software

This library and guide has been deprecated. We have a newer, better library and guide here https://learn.adafruit.com/adafruit-mcp9808-precision-i2c-temperature-sensor-guide/

To install and use the MCP9808 Python library (https://adafru.it/e08) follow the steps below.

Before you get started make sure your board is connected to the internet through an ethernet or wireless connection so you can download dependencies.

You'll also want to be familiar with connecting to a Raspberry Pi (https://adafru.it/jvB) o r BeagleBone Black (https://adafru.it/ne9) terminal with SSH.

Dependencies

First install dependencies by executing in a terminal:

```
sudo apt-get update
sudo apt-get install build-essential python-dev python-pip python-smbus git
```

You can ignore warnings about dependencies which are already installed.

Raspberry Pi

On a Raspberry Pi execute the following to make sure the RPi.GPIO library is installed:

```
sudo pip install RPi.GPIO
```

BeagleBone Black

On a BeagleBone Black execute the following to make sure the Adafruit_BBIO library is installed:

```
sudo pip install Adafruit_BBIO
```

Library Install

Next download the MCP9808 Python library to your home directory and install it by executing:

```
cd ~
git clone https://github.com/adafruit/Adafruit_Python_MCP9808.git
cd Adafruit_Python_MCP9808
sudo python setup.py install
```

That's all you need to do to install the Python library!

Usage

To learn how to use the MCP9808 Python library you can run and review an example program included with the library. To run the example navigate to the examples directory and run the simpletest.py script by executing:

```
cd examples
sudo python simpletest.py
```

If everything goes well you should see the sensor's temperature displayed every second:

```
Press Ctrl-C to quit.
Temperature: 23.750*C / 74.750*F
Temperature: 25.688*C / 78.237*F
Temperature: 27.000*C / 80.600*F
...
```

If you see an error make sure you're running the program as root with the sudo command, and that the dependencies & library were installed successfully.

To understand the usage, open simpletest.py in a text editor and follow along with the description of the code below.

```
import Adafruit_MCP9808.MCP9808 as MCP9808
```

First the MCP9808 module is imported with a Python import statement.

```
# Default constructor will use the default I2C address (0x18) and pick a default
I2C bus.
#
# For the Raspberry Pi this means you should hook up to the only exposed I2C bus
# from the main GPIO header and the library will figure out the bus number based
# on the Pi's revision.
#
# For the Beaglebone Black the library will assume bus 1 by default, which is
# exposed with SCL = P9_19 and SDA = P9_20.
sensor = MCP9808.MCP9808()

# Optionally you can override the address and/or bus number:
#sensor = MCP9808.MCP9808(address=0x20, busnum=2)
```

Next an instance of the MCP9808 class is created.

Notice that if nothing is passed in to the initializer function the library will pick a default I2C device address (0x18) and bus number. If you need to specify the I2C address or bus number you can do so by specifying them as optional parameters in the MCP9808 initializer.

```
# Initialize communication with the sensor.
sensor.begin()
```

After creating the MCP9808 class instance the begin() function is called to initialize communication with the device.

```
# Loop printing measurements every second.
print 'Press Ctrl-C to quit.'
```

```
while True:
        temp = sensor.readTempC()
        print 'Temperature: {0:0.3F}*C / {1:0.3F}*F'.format(temp, c_to_f(temp))
        time.sleep(1.0)
```

Finally the example enters a loop where it reads the sensor's temperature and prints it out every second. The important thing to see is the temperature measurement function readTempC(). The readTempC() function will read the sensor temperature and return its value in Celsius.

That's all there is to use the MCP9808 Python library! If you run into issues or wish to contribute, feel free to followup on the library's Github page (https://adafru.it/e08).