# EGR 597: Internet of Things: Standards, Protocols, and Applications
# Lab 4: Thermistor and BME680 Temperature and Humidity Sensor

Central Michigan University
2018-2019

**Objective:** The objective of this lab is to learn the calibration process of thermistors and using digital temperature sensors.

In part 1 we will learn the fundamental principles of a negative temperature coefficient thermistor and how to calibrate them. In part 2 we will learn to use BME680, a digital temperature and humidity sensor.

**Materials Needed:**

1) Raspberry Pi 3
2) ADC chip (ADS1115)
3) Prototyping breadboard
4) Thermistor
5) 10KΩ resistor
6) Thermometer
7) Beaker
8) Hot plate
9) Ice
10) BME680
11) Jumper wires.

## Part 1: Thermistor

Thermistors are resistors that are thermally sensitive; their prime function is to exhibit a large, precise change in electrical resistance when subjected to a corresponding change in temperature. Temperature rise or fall changes the resistance of thermistors. There are two types of thermistors in terms of characteristics, namely PTC (Positive Temperature Coefficient) and NTC (Negative Temperature Coefficient) thermistors. In case of a NTC thermistor the resistance increases as the temperature decreases and vice-versa whereas in case of a PTC thermistor the resistance increases as the temperature increases and vice-versa. There are many kinds of thermistors depending on the application. Physically they might look very different but have the same operational principle. Fig. 2 shows a waterproof thermistor that we will be using during the lab. You can find more fundamental information about Thermistors at https://en.wikipedia.org/wiki/Thermistor

**Fig. 1.** Different types of thermistors

**Fig. 2.** Waterproof Thermistor

Now you know the thermistor changes its resistance depending on the temperature but how can we know the value of temperature from the resistance? Let's say for example, in room temperature an NTC thermistor's resistance is 1 kΩ and when we heat it up, its resistance drops to 500 Ω. However, we do not know the temperature of thermistor at this time. Accordingly, we need to calibrate the thermistor to identify the relationship between temperature and resistance of the thermistor. Calibration is an unavoidable process to keep your equipment or sensors accurate and precise over time. It is also used to characterize unknown sensors. The formal definition of calibration by the International Bureau of Weights and Measures (BIPM) is the following: "*Operation that, under specified conditions, in a first step, establishes a relation between the quantity values with measurement uncertainties provided by measurement standards and corresponding indications with associated measurement uncertainties (of the calibrated instrument or secondary standard) and, in a second step, uses this*

*information to establish a relation for obtaining a measurement result from an indication*." So we will calibrate our thermistor using a thermometer to establish a relation between temperature and the corresponding resistance of the thermistor.

In the next few steps we will take voltage measurements (the voltage across the thermistor directly corresponds to the changes in resistance of the thermistor) across a wide range of temperatures (from sub zero temperatures to 212℉), draw a characteristics curve in Excel and derive an equation from the curve. We can then use that equation to get the temperature from the resistance value of the thermistor.

**Step 1:** Take a 10KΩ resistor and make a voltage divider circuit with the thermistor and power it with 3.3V as shown in figure 2. Do note that thermistors don't have any polarity (similar to conventional resistors).

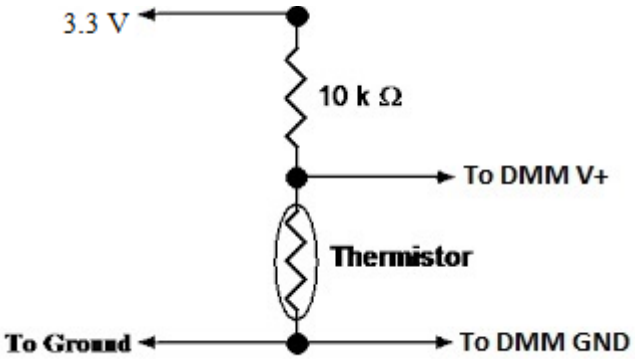


**Fig. 3.** Thermistor Calibration Circuit

**Step 2:** Put water and ice in a beaker, insert the thermistor in the water, and heat the water on a hot plate. Make sure the thermistor wire is not touching the hot plate surface. Take readings of the temperature of the water with a thermometer and the associated voltage. (If you do not have a multimeter, replace the LDR in lab-03 with thermistor, and record the digital output of ADC). Take about twenty readings as the water boils.

**Step 3:** Enter the data you collected into an excel sheet with the voltage readings in the first column and the temperature readings in the second. Click the insert tab, select the data set and make a graph with it by selecting the "scatter with smooth lines" option.

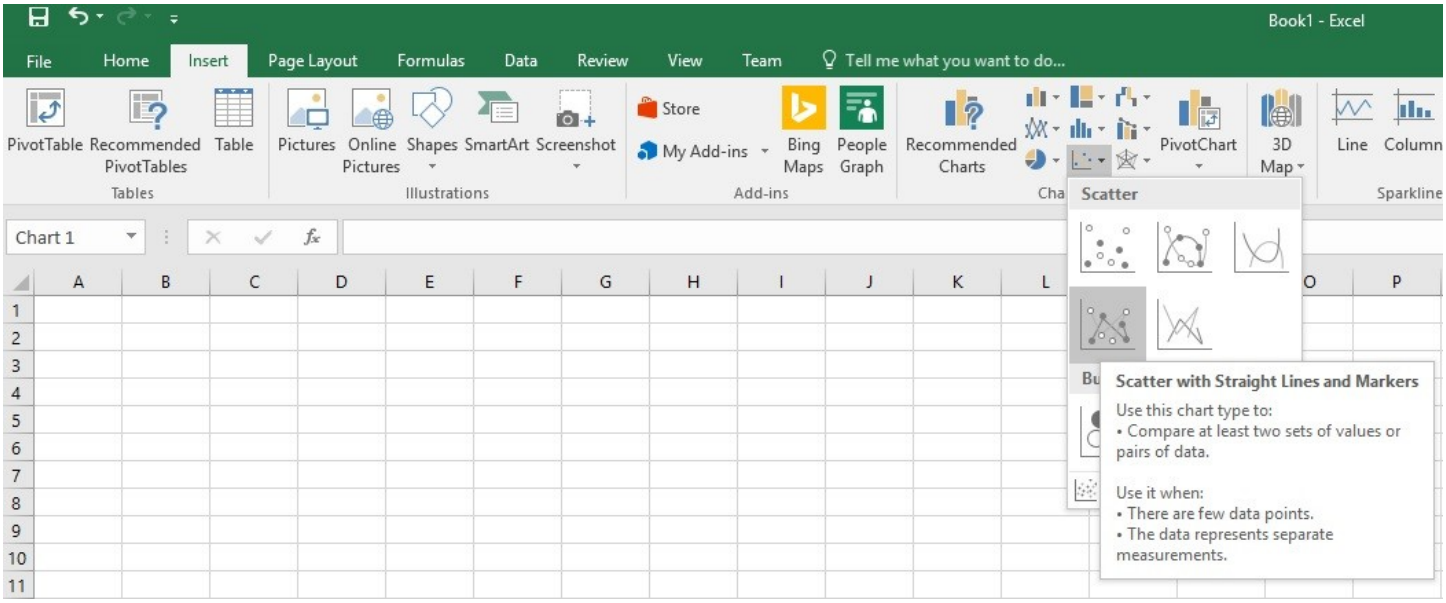


**Fig. 4:** Selecting “Scatter with Straight Lines and Markers”

You should see a graph drawn on the excel sheet. Acquire the equation from the curve by selecting the curve, right clicking, selecting “add trend line”, and then scrolling down in the menu and check the box next to “Display equation on chart” shown on Fig. 5.

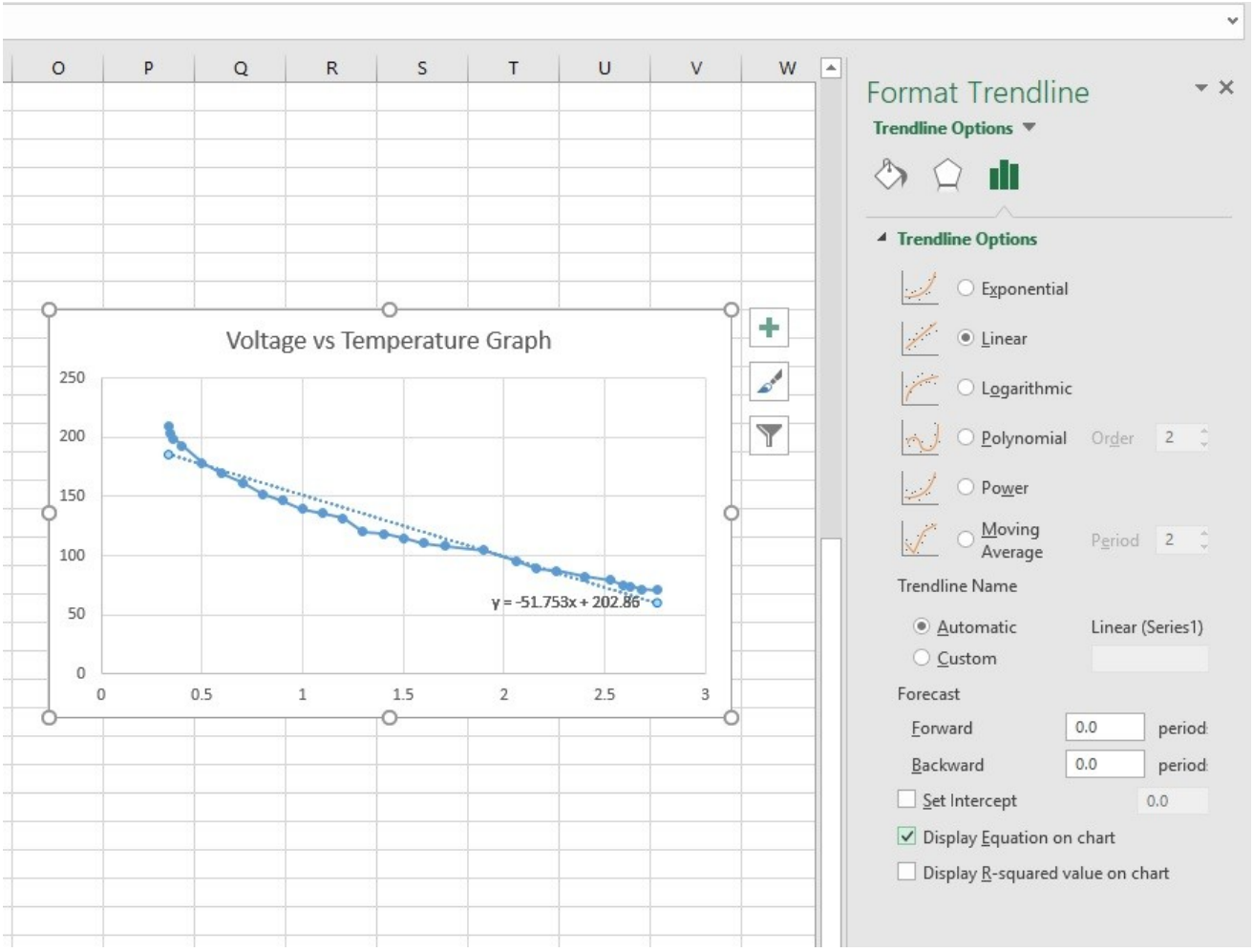


**Fig. 5.** Check “Display Equation on Chart”

Now you should see an equation next to the graph just like in Figure 6. Do note that the calibration equation could be different across thermistors.

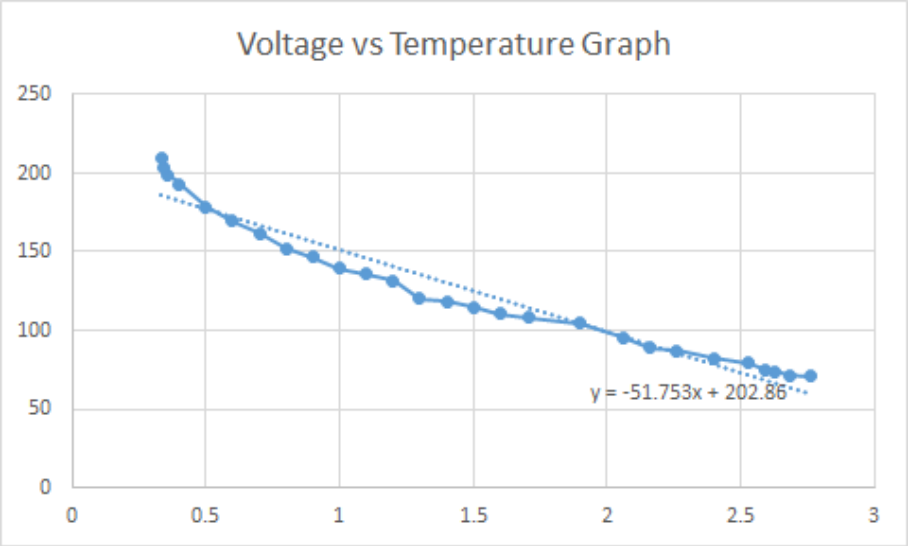


**Fig. 6.** Voltage vs Temperature Graph

Now that you have the equation, you can read the voltage across the thermistor with an ADC chip connected to the Pi and plug that into the equation to find current temperature.

## Part 2: BME680 Temperature, Pressure and Humidity Sensor

Now we are going to learn how to interface BME680 with our Raspberry Pi. Let's first learn what this sensor is. Pictured in Fig. 7 is a BME680 sensor.

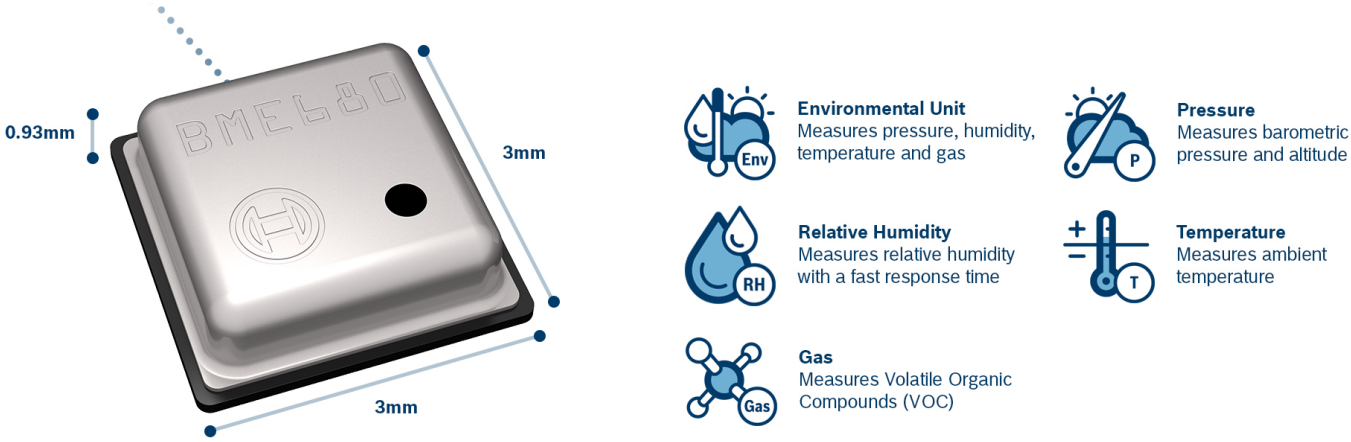


**Fig. 7.** BME680 Sensor

BME680 is an environmental sensor made by Bosch Sensortec. It can sense temperature, humidity, barometric pressure, and VOC (Volatile Organic Compounds) gas, all integrated into one low power, 8 pin, monolithic chip. It can communicate via SPI or I2C. If you don't change the communication method, it defaults to I2C. The sensor is fairly accurate and since pressure changes with altitude, you can also use this sensor as an altimeter. If you want to know more about this sensor, here's a link to its datasheet: https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BME680-DS001-00.pdf

For our experiments, we will use a sensor module by Adafruit as in Fig. 8 which handles the supply voltage requirements and communication level shiftings so that we can use this sensor without worrying about all that.

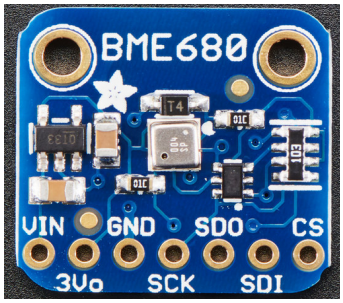


**Fig. 8.** BME680 Sensor Module

There are 7 pins on the module. Namely,

VIN - Voltage input pin
3Vo - 3.3V output
GND - Ground pin
SCK - SPI Clock pin, also the clock pin for I2C
SDO - Serial Data Out pin
SDI - Serial Data In pin, also the data pin for I2C
CS - Chip Select Pin for SPI communication

For our experiment we will use I2C communication, so we just need to VIN, GND, SCK and SDI pins. We can leave the remaining pins disconnected. Now let’s connect this sensor with the Raspberry Pi as in Fig. 9.

Pi 3V3 ☑ sensor VIN
Pi GND ☑ sensor GND
Pi SCL ☑ sensor SCK
Pi SDA ☑ sensor SDI

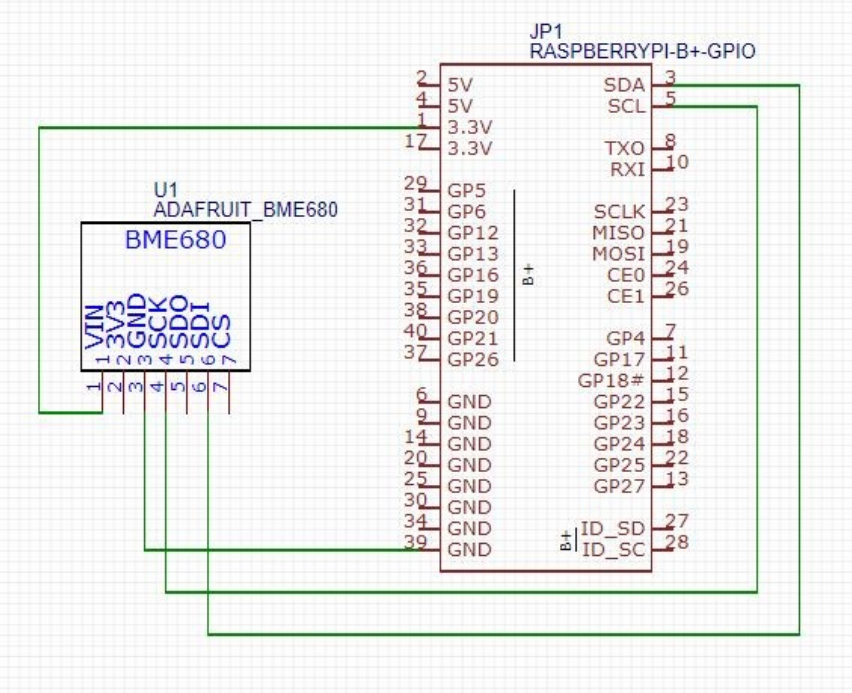


**Fig. 9.** Wiring schematic for BME680 and Raspberry Pi

**The Software:**

We are going to use a python library from *Pimoroni*. To install this library open a terminal window and enter the following command. Press enter after every command.

*git clone https://github.com/pimoroni/bme680-python.git*

*cd bme680-python/library*

*sudo python setup.py install*

*cd ~*

Now we have the library installed on our Pi. Let's try out the sensor with a simple code that will give us the ambient temperature in degree fahrenheit, the atmospheric pressure in hPa and the relative humidity. Open a new text file and save it with a name "bme680_test.py" and paste the following code.

**The Code:**

```
#!/usr/bin/env python
import bme680
import time
```

```
print("""Display Temperature, Pressure and Humidity

Press Ctrl+C to exit

""")

try:
    sensor = bme680.BME680(bme680.I2C_ADDR_PRIMARY)
except IOError:
    sensor = bme680.BME680(bme680.I2C_ADDR_SECONDARY)

sensor.set_humidity_oversample(bme680.OS_2X)
sensor.set_pressure_oversample(bme680.OS_4X)
sensor.set_temperature_oversample(bme680.OS_8X)
sensor.set_filter(bme680.FILTER_SIZE_3)

try:
    while True:
        if sensor.get_sensor_data():

            output = '{0:.2f} F, {1:.2f} hPa, {2:.3f} %RH'.format(
              (1.8* sensor.data.temperature) + 32.0,
               sensor.data.pressure,
               sensor.data.humidity)

            print(output)

        time.sleep(2)

except KeyboardInterrupt:
    pass
```

Type the following command in the terminal and you should start seeing temperature, pressure and humidity values popping up on the screen every two seconds. Press Ctrl + C to quit the program.

***sudo python bme680_test.py***

**Lab Report:**

Include the following in your lab report

1. Lessons learned

2. Schematics and hookups.

3. Does condensation have any effect on the humidity readings? If so, what can be done to minimize this effect?

4. What are the different interfacing modes for BME 680? What mode was utilized in the lab, and how would you update the code to interface in a different mode?

5. What is the operating range of the BME 680 for temperature, pressure, and humidity?

6. What is the accuracy of the BME 680 for temperature, pressure, and humidity?

7. Name three different applications where a combination of any two parameters (temperature, pressure, humidity) are necessary for efficient operation.

8. The datasheet mentions a specification of “Response time to complete 63% of step.” Identify the significance of this specification.

9. Based on information in the datasheet, calculate (show work) the power consumption of the sensor operating at 3.3V at a rate of 10Hz. Also, how would you estimate the energy consumed?