Assignment 5

BMP 280 Sensor with ESP8266 / ESP 32

By Kumar Anmol

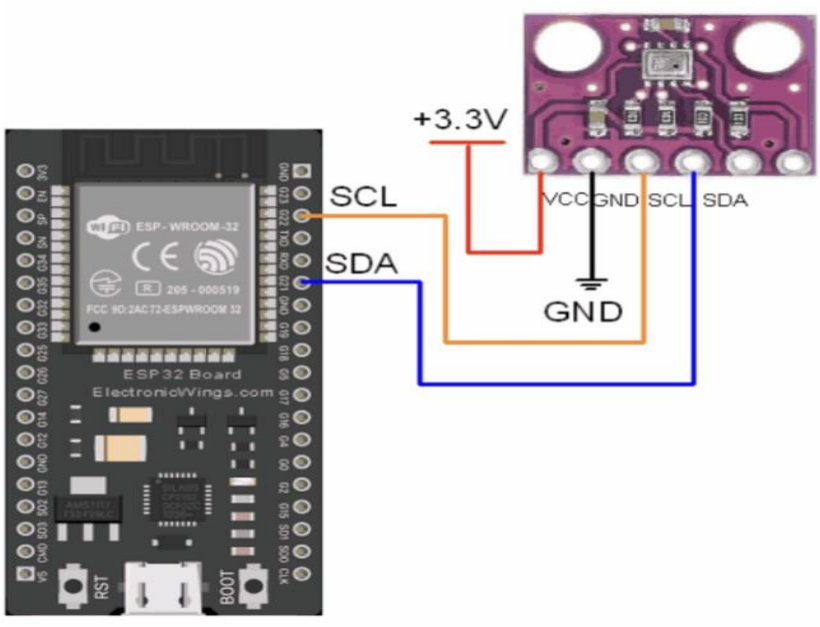
Aim: To interface a BMP280 Pressure, Temperature, and Altitude sensor with an ESP8266 / ESP32 module. This experiment demonstrates the capability of the IoT kit to collect temperature, pressure, and altitude data from the BMP280 sensor and transmit it to a Raspberry Pi for further processing or display.

Equipment:

1. Raspberry Pi 4 microprocessor
2. ESP32 module
3. BMP280 Temperature, Pressure, and Altitude sensor
4. Female-to-female connecting wires
5. HDMI cables
6. B-type connecting cable
7. Desktop

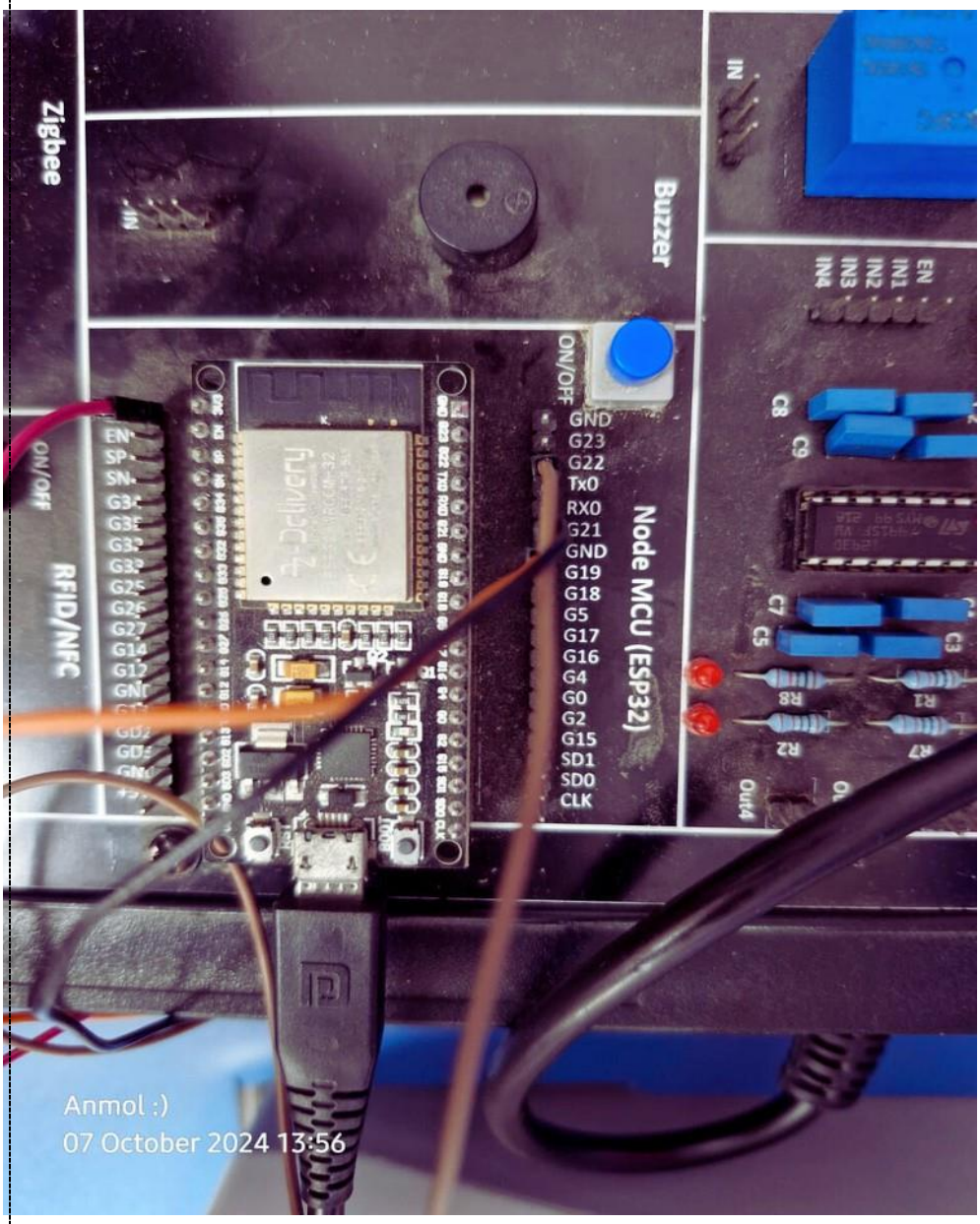
Methodology:

Connect the GND (ground) pin of the BMP280 sensor to the GND (ground) pin of the ESP32. Connect the VCC pin of the BMP280 sensor to the 3.3V (3V3) pin of the ESP32. The SCL pin of the BMP280 sensor should be connected to the G22 pin of the ESP32, and the SDA pin to the G21



pin of the ESP32. Use the B-type cable to connect the Raspberry Pi 4 microprocessor to the

ESP32 microcontroller board. The circuit is shown below:



Results:- The output we received are temperature, altitude and pressure data on the serial monitor of Arduino IDE on desktop connected to RPI 4 microporcessor.

Below is the drive link attached that contains the video of the circuit and output we received on serial monitor of Arduino IDE:-

Drive link:- Link to the drive

Conclusion:- We learnt about the connections to be installed between sensors like BMP-280 sensor with micro-controllers like ESP 32, also learnt about different libraries like Adafruit, BMP-280 library, that need to be installed on Arduino IDE, learnt about coding that need to be done to get data from sensor and also understood about the working principle of sensor like BMP-280.

Code:- Below is the code that we wrote in Arduino IDE:-

```
This is a library for the BMP280 humidity, temperature & pressure sensor

Designed specifically to work with the Adafruit BMP280 Breakout
----> http://www.adafruit.com/products/2651

These sensors use I2C or SPI to communicate, 2 or 4 pins are required
to interface.


#include <Wire.h>
#include <SPI.h>
```

```
#include <Adafruit_BMP280.h>

#define BMP_SCK  (13)
#define BMP_MISO (12)
#define BMP_MOSI (11)
#define BMP_CS   (10)

Adafruit_BMP280 bmp; // I2C
//Adafruit_BMP280 bmp(BMP_CS); // hardware SPI
//Adafruit_BMP280 bmp(BMP_CS, BMP_MOSI, BMP_MISO,  BMP_SCK);

void setup() {
 Serial.begin(9600);
 while ( !Serial ) delay(100);   // wait for native usb
 Serial.println(F("BMP280 test"));
 unsigned status;
 //status = bmp.begin(BMP280_ADDRESS_ALT, BMP280_CHIPID);
 status = bmp.begin(0x76);
 if (!status) {
  Serial.println(F("Could not find a valid BMP280 sensor, check wiring or "
                   "try a different address!"));
  Serial.print("SensorID was: 0x"); Serial.println(bmp.sensorID(),16);
  Serial.print("     ID of 0xFF probably means a bad address, a BMP 180 or BMP 085\n");
  Serial.print("  ID of 0x56-0x58 represents a BMP 280,\n");
  Serial.print(" ID of 0x60 represents a BME 280.\n");
  Serial.print(" ID of 0x61 represents a BME 680.\n");
  while (1) delay(10);
 }

 /* Default settings from datasheet. */
 bmp.setSampling(Adafruit_BMP280::MODE_NORMAL,  /* Operating Mode. */
                 Adafruit_BMP280::SAMPLING_X2, /* Temp. oversampling */
                 Adafruit_BMP280::SAMPLING_X16,   /* Pressure oversampling */
                 Adafruit_BMP280::FILTER_X16, /* Filtering. */
                 Adafruit_BMP280::STANDBY_MS_500); /* Standby time. */
}

void loop() {
  Serial.print(F("Temperature = "));
  Serial.print(bmp.readTemperature());
  Serial.println(" *C");

  Serial.print(F("Pressure = "));
  Serial.print(bmp.readPressure());
  Serial.println(" Pa");
```

```
  Serial.print(F("Approx altitude = "));
  Serial.print(bmp.readAltitude(1011.9)); /* Adjusted to local forecast! */
  Serial.println(" m");

  Serial.println();
  delay(2000);
}
```