BME680

BME680

Notes:

- Didn't need to download library Adafruit_Sensor, just needed to download Adafruit_BME680
- Data values changed under different circumstances (leaving it on the table vs talking vs blowing air)
- More resistance (higher KOhms value) → Worse air quality
    - Higher VOC concentrations lead to lower resistance
        - Lower KOhm value → bad air quality
- HVAC potentially impacting data?

Issues Encountered:

- Library Manager not loading libraries → saved file, closed Arduino IDE, reopen file
- Altitude: In OL → <1m but in Lab 3 ; inconsistent (can test outside, not too concerning)

Sample Data in OL:

```
Temperature = 24.05 *C

Pressure = 1013.22 hPa

Humidity = 60.34 %

Gas = 124.16 KOhms

Approx. Altitude = 0.17 m


Temperature = 24.05 *C

Pressure = 1013.22 hPa

Humidity = 60.32 %

Gas = 124.44 KOhms

Approx. Altitude = 0.25 m


Temperature = 24.04 *C

Pressure = 1013.22 hPa

Humidity = 60.32 %

Gas = 124.91 KOhms
```

Approx. Altitude = 0.33 m

Temperature = 24.04 *C

Pressure = 1013.21 hPa

Humidity = 60.34 %

Gas = 124.53 KOhms

Approx. Altitude = 0.33 m

Temperature = 24.04 *C

Pressure = 1013.21 hPa

Humidity = 60.35 %

Gas = 124.81 KOhms

Approx. Altitude = 0.25 m

Temperature = 24.04 *C

Pressure = 1013.22 hPa

Humidity = 60.35 %

Gas = 125.00 KOhms

Approx. Altitude = 0.33 m

Temperature = 24.03 *C

Pressure = 1013.20 hPa

Humidity = 60.34 %

Gas = 125.47 KOhms

Approx. Altitude = 0.33 m

Temperature = 24.03 *C

Pressure = 1013.20 hPa

Humidity = 60.35 %

Gas = 124.72 KOhms

Approx. Altitude = 0.33 m

Temperature = 24.03 *C

Pressure = 1013.20 hPa

Humidity = 60.35 %

Gas = 124.72 KOhms

Approx. Altitude = 0.33 m

Temperature = 24.03 *C

Pressure = 1013.20 hPa

Humidity = 60.36 %

Gas = 125.47 KOhms

Approx. Altitude = 0.33 m

Temperature = 24.03 *C

Pressure = 1013.21 hPa

Humidity = 60.37 %

Gas = 125.47 KOhms

Approx. Altitude = 0.25 m

Temperature = 24.03 *C

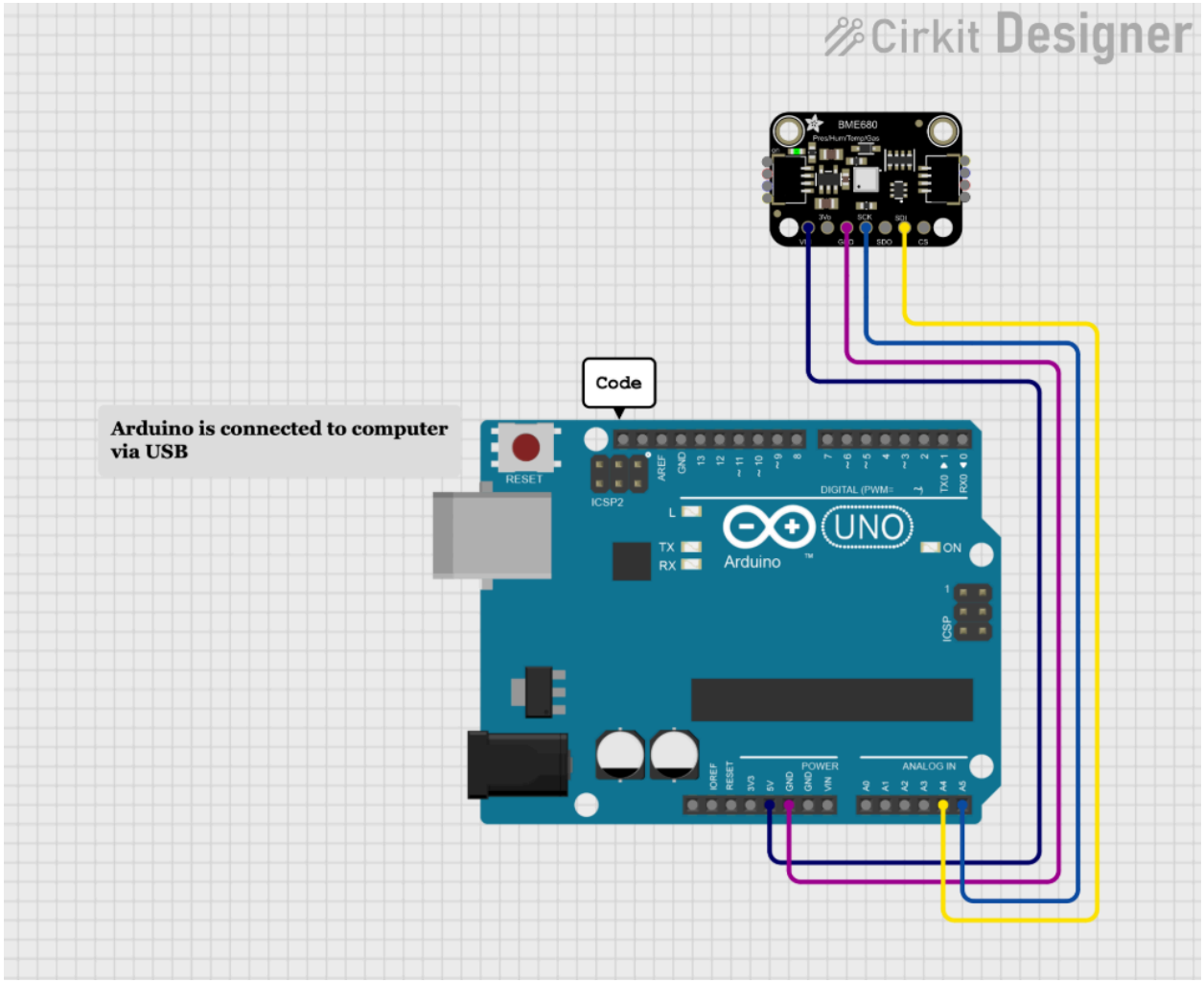
Pressure = 1013.22 hPa

Humidity = 60.36 %

```
Gas = 125.94 KOhms

Approx. Altitude = 0.25 m
```

Testing:

- Correct Configuration
    - VCC → 5V
    - GND → GND
    - SCL/SCK → A5
    - SDA/MOSI → A4
        - If using Arduino Mega, SCI goes to SCI pin and SDA goes to SDA pin, do not assign to analog pins
    - CS and ADDR/MISO → Not Used



Sample Data:

```
Temperature = 26.59 *C

Pressure = 1012.02 hPa

Humidity = 51.94 %

Gas = 121.98 KOhms

Approx. Altitude = 10.16 m

Temperature = 26.58 *C

Pressure = 1012.03 hPa

Humidity = 52.04 %

Gas = 121.81 KOhms

Approx. Altitude = 10.08 m
```

- If SCL/SCk in A4 and SDA/MOSI in A5 → Error message shows "Could not find a valid BME680 sensor, check wiring!"

Sample Data (OL 7/15)

```
Temperature = 24.73 *C

Pressure = 1016.04 hPa

Humidity = 60.56 %

Gas = 153.10 KOhms

Approx. Altitude = -23.20 m

Temperature = 24.74 *C

Pressure = 1016.04 hPa

Humidity = 60.62 %
```

Gas = 154.37 KOhms

Approx. Altitude = -23.20 m

Temperature = 24.79 *C

Pressure = 1016.05 hPa

Humidity = 61.13 %

Gas = 152.12 KOhms

Approx. Altitude = -23.37 m

Temperature = 24.88 *C

Pressure = 1016.06 hPa

Humidity = 60.95 %

Gas = 152.82 KOhms

Approx. Altitude = -23.37 m

Temperature = 24.94 *C

Pressure = 1016.06 hPa

Humidity = 60.65 %

Gas = 151.84 KOhms

Approx. Altitude = -23.37 m

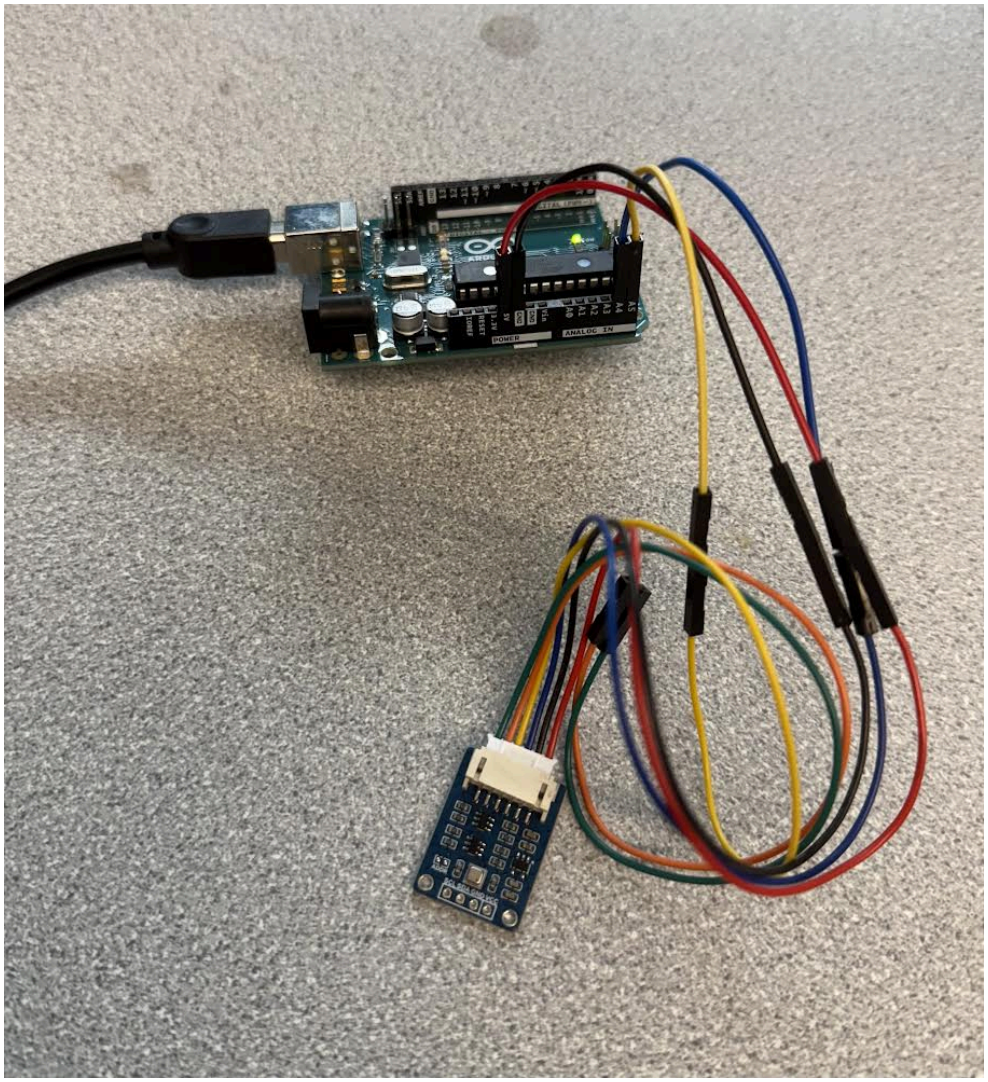
Temperature = 25.00 *C

Pressure = 1016.06 hPa

Humidity = 61.09 %

Gas = 150.61 KOhms

Approx. Altitude = -23.37 m


POWER
ANALOG IN

MQ135

MQ135

Notes:

- Measures VOC in ppm
- Gets quite hot (is this an issue?)

Issues Encountered:

- Could not compare readings with BME 680 due to unit issues (BME 680 measures in KOhms and MQ135 measures in ppm) (reference to convert? Kohms → PPM)

Sample Data:

- From 156 → 89; went down by increments of 1

```
90

90

90

90

90

90

90

90

90

90

89

90

90

90

89

89

89

89

89
```

89

89

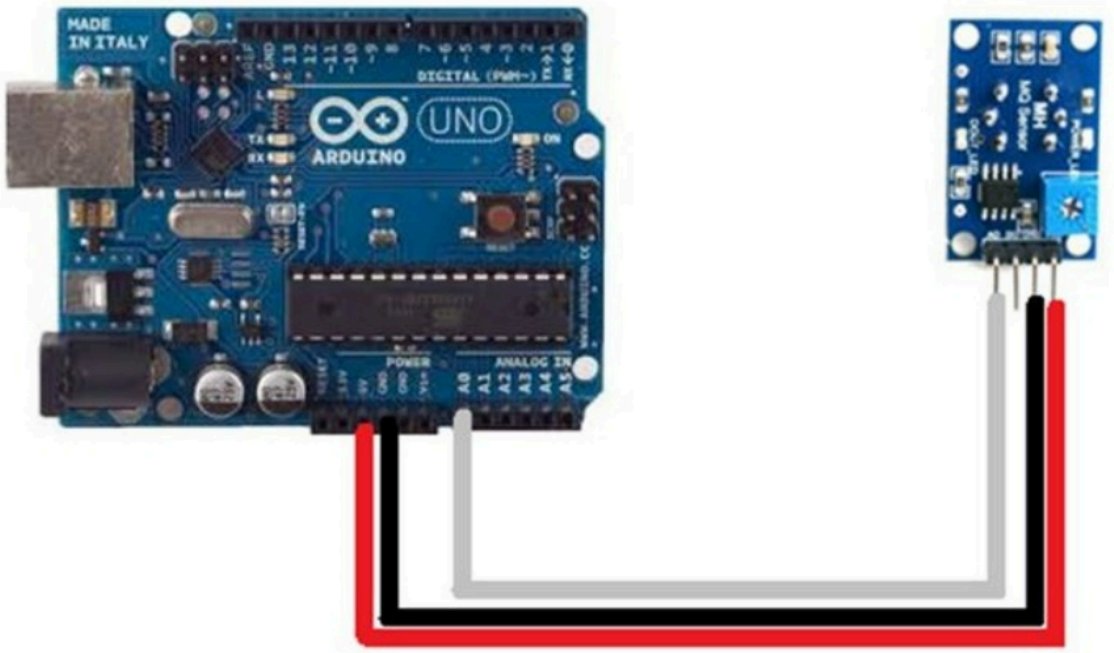
89

89

89

Correct Configuration:

- VCC → 5V
- GND → GND
- AO (Analog Output) → A0



DHT11 and DHT22

DHT11

Notes:

- Values (temp and humidity) match BME680
- No need for resistor (when resistor was in place, no values collected)

Issues Encountered:

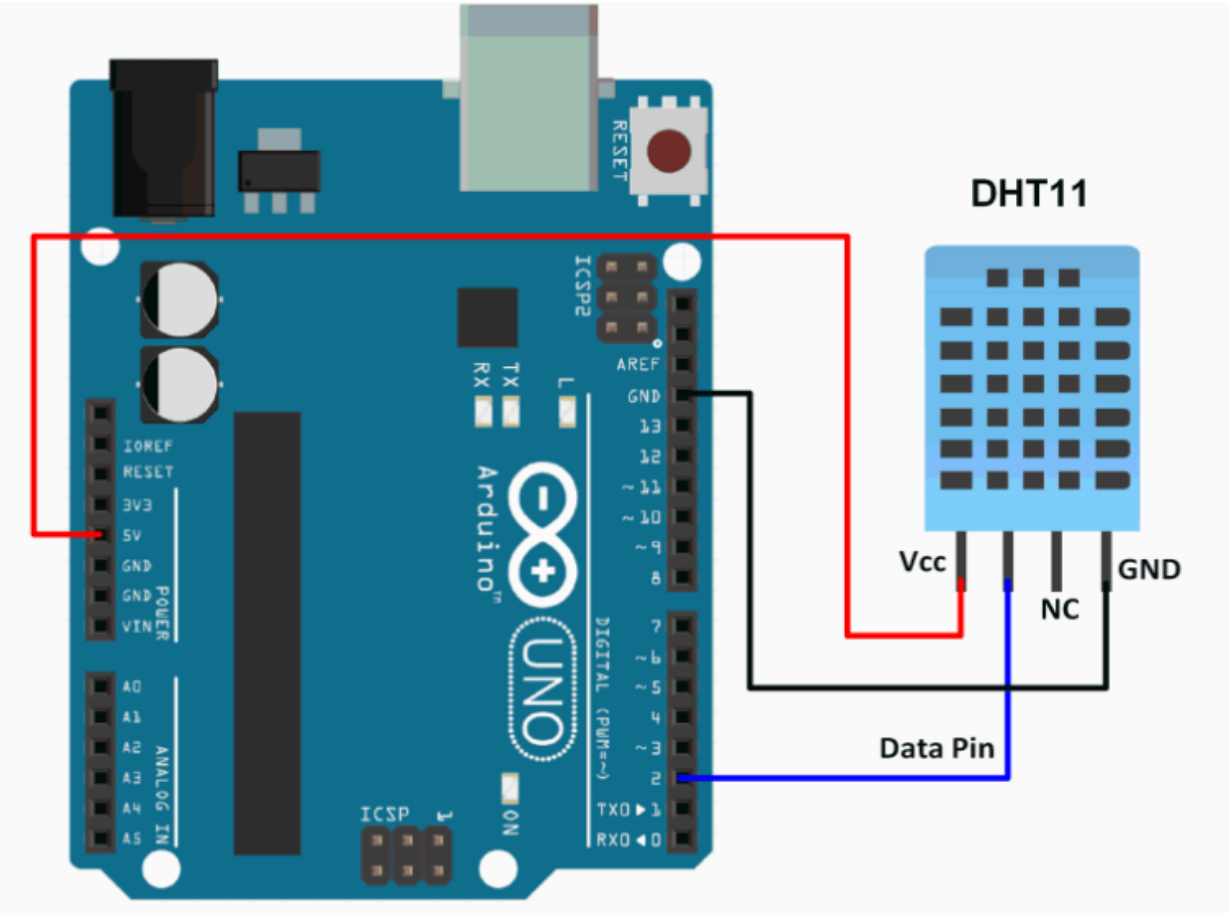
- Needed to download specific zip library (linked in our code); the dht library in library manager did not work
    - To import new library: download zip file of library → sketch → include library → add zip library
    - Link: https://projecthub.arduino.cc/arcaegecengiz/using-dht11-12f621

Sample Data:

```
Humidity (%): 59.00

Temperature  (C): 24.00

Humidity (%): 59.00

Temperature  (C): 24.00

Humidity (%): 59.00

Temperature  (C): 24.00

Humidity (%): 59.00

Temperature  (C): 24.00

Humidity (%): 59.00

Temperature  (C): 24.00

Humidity (%): 59.00

Temperature  (C): 24.00
```

Correct Configuration:

- VCC → 5V
- GND → GND
- Signal pin →Digital 2


DHT11
Vcc
NC
GND
Data Pin
RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
TX
RX
L
ON
ICSP
1
Arduino™
UNO
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN

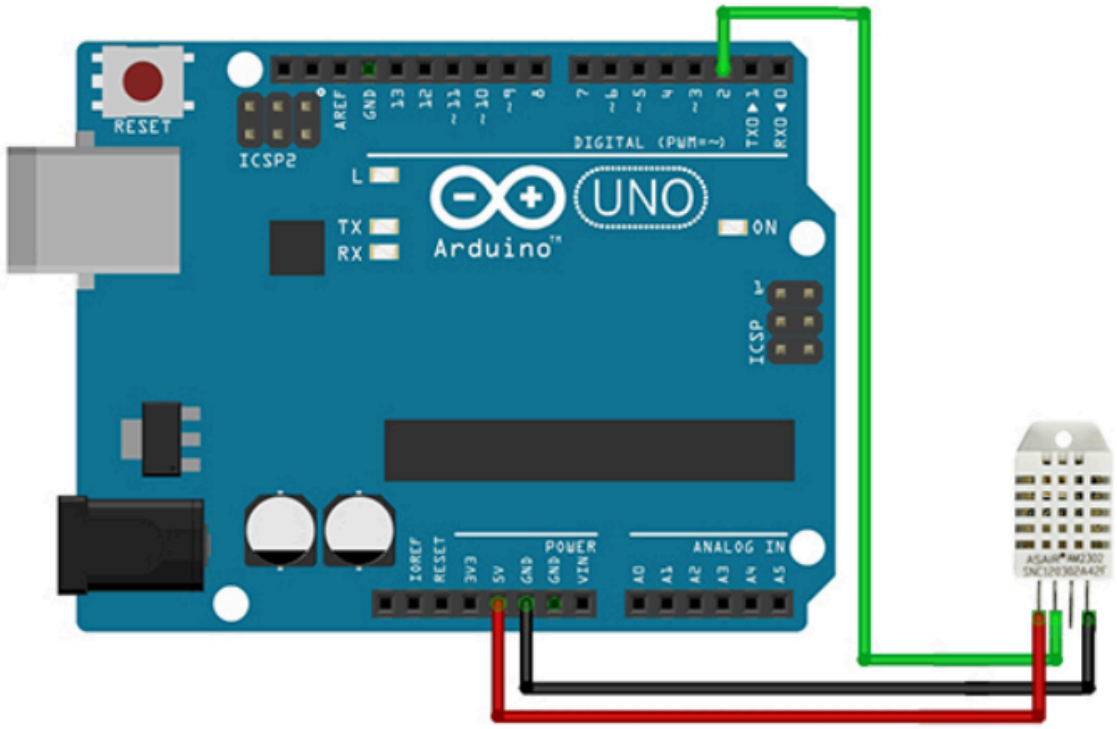
DHT22

Issues Encountered:

- Library would not upload (we zipped dht.h and dht.cpp files, did not work)

Correct Configuration (same as DHT11):

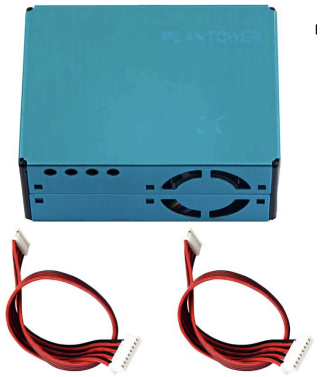
- VCC → Power
- GND → GND
- Signal Pin → Digital 2



PMS5003

PMS5003

- Missing Part?
    - The order came like this, with no clear way to plug into breadboard



    - Solution: I found the pin definitions, used a wire stripper to cut off one end of the wire that came with it, and exposed the wire to be able to plug into a breadboard

Pin Definition

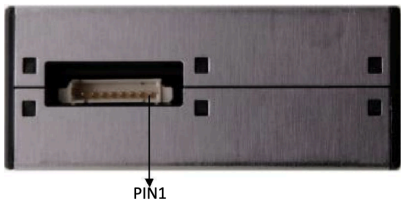
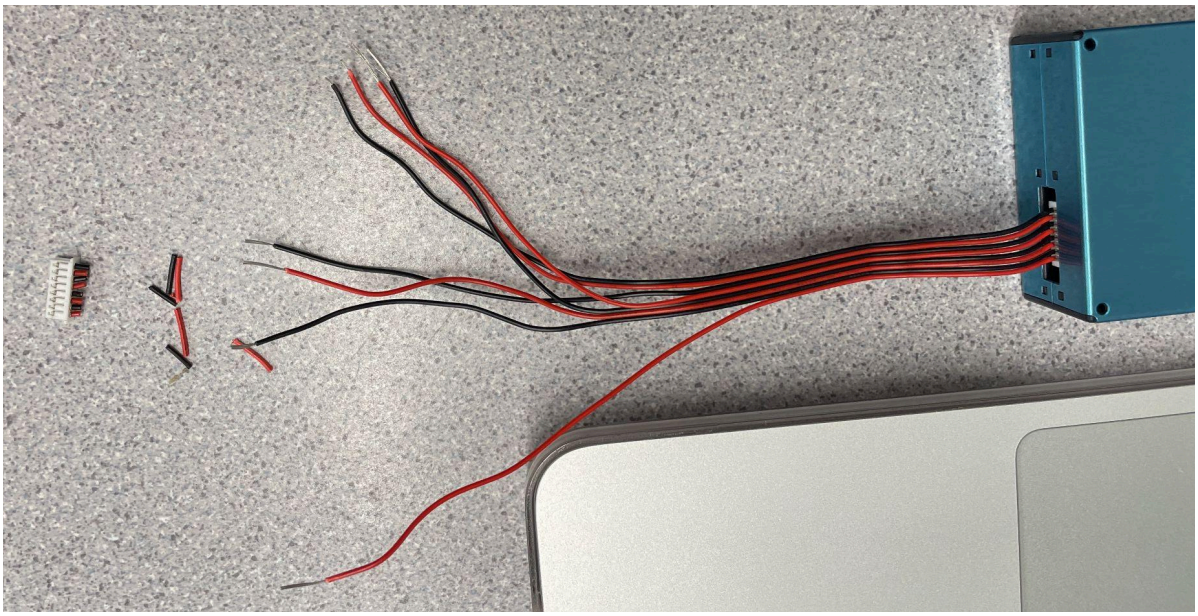
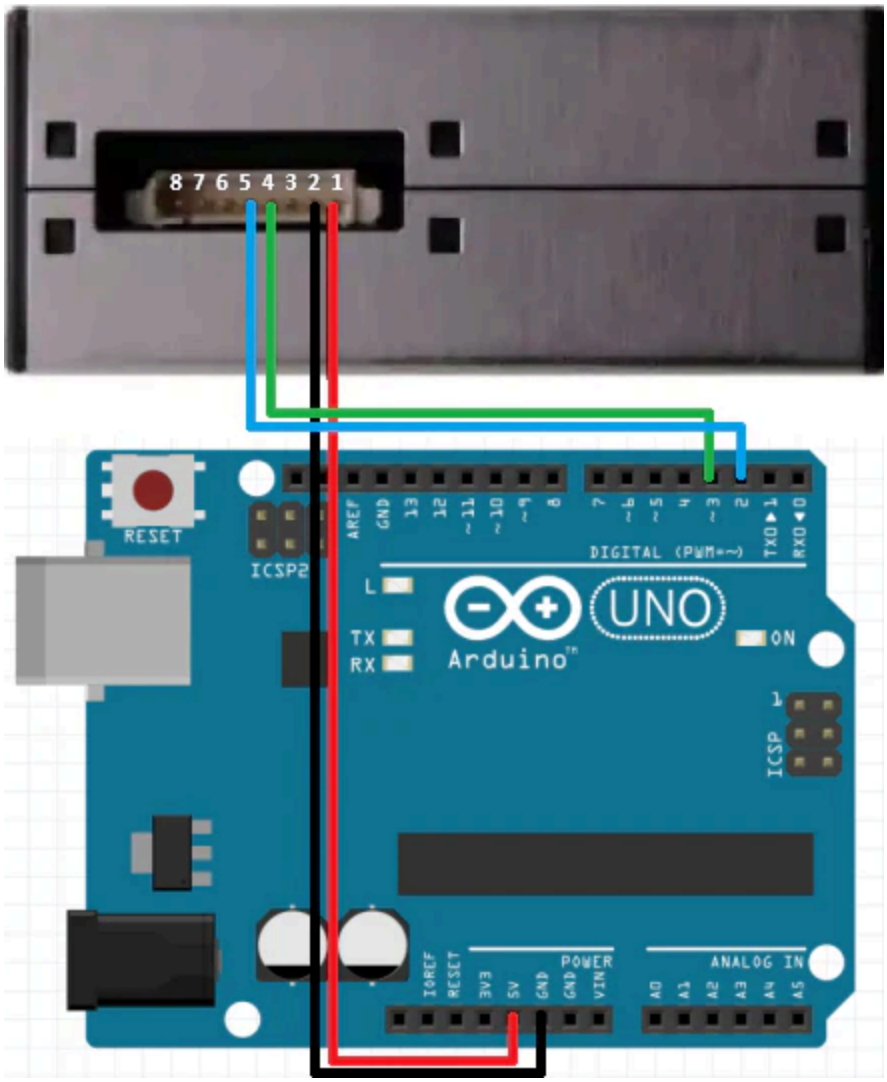


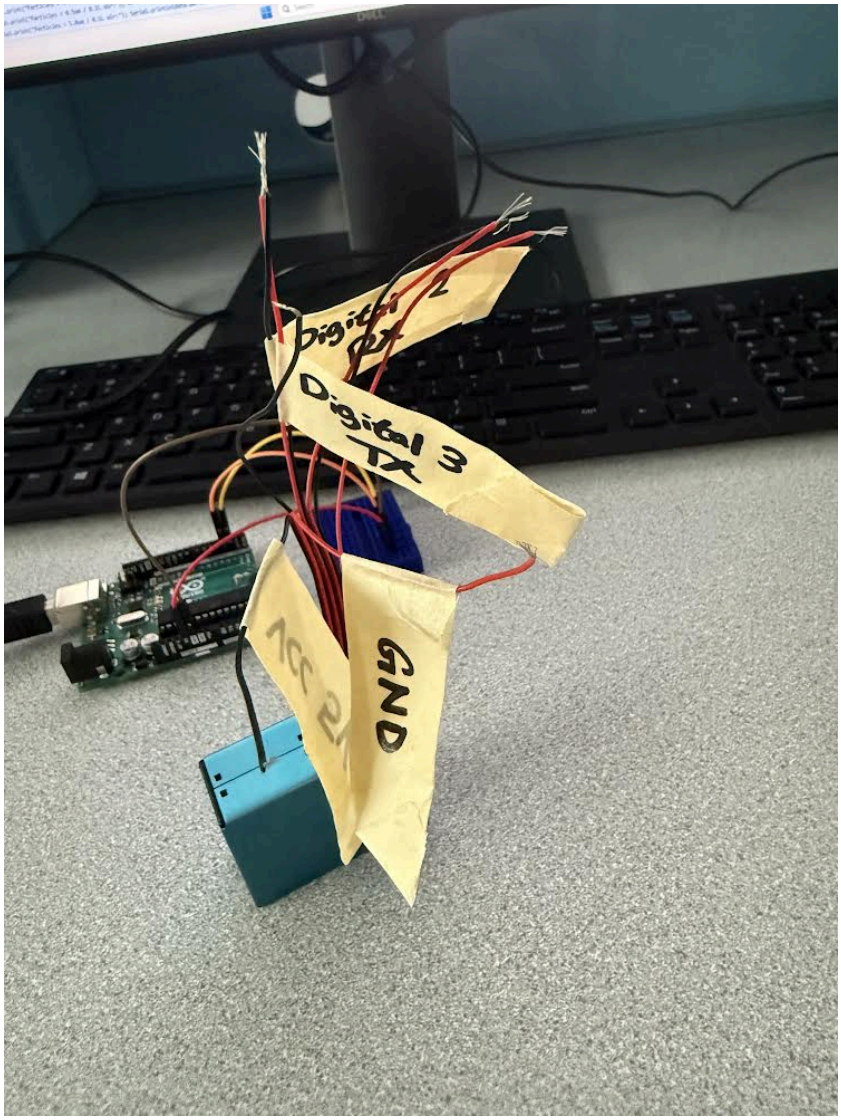
Figure 2 Connector Definition

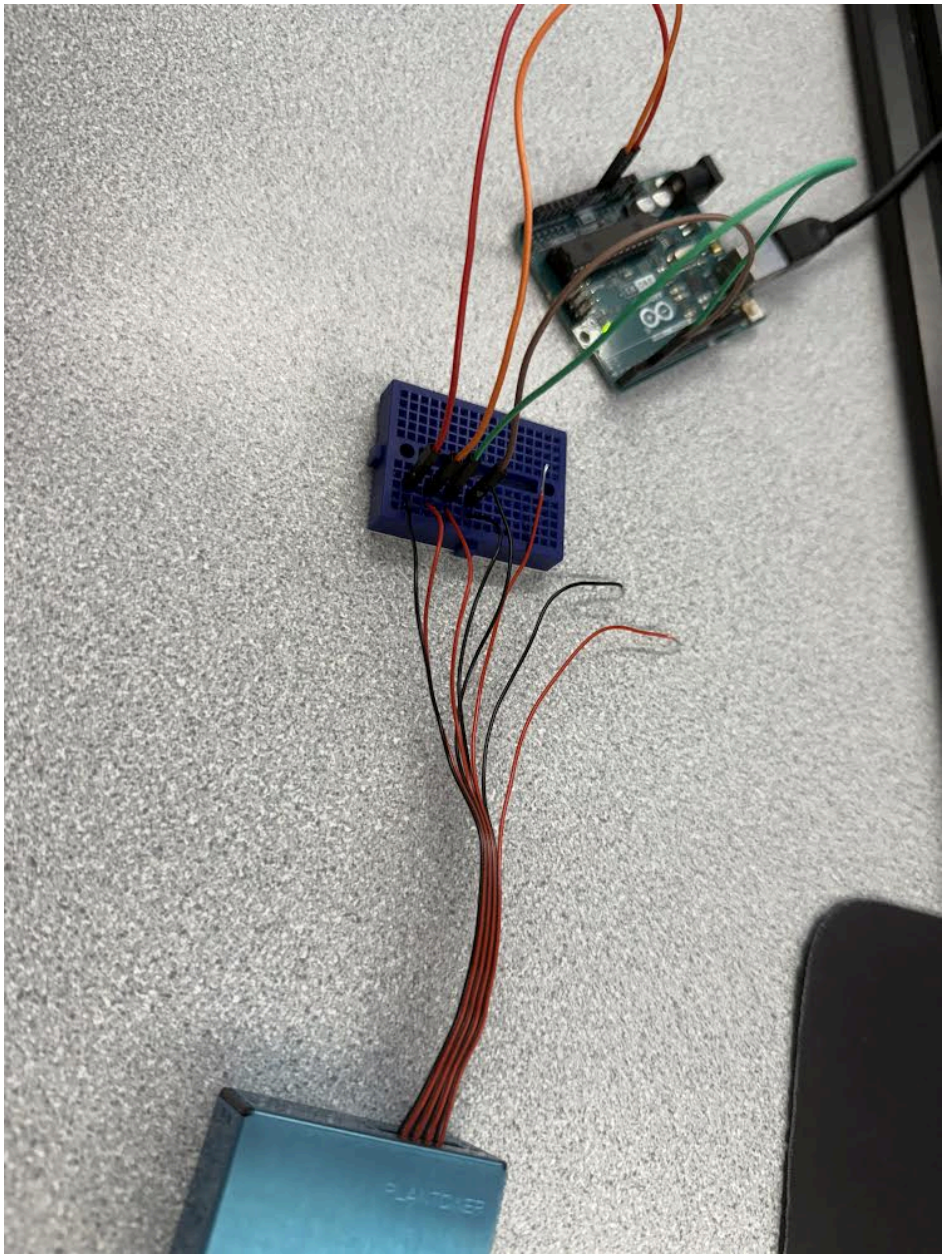
PIN1	VCC	Positive power 5V
PIN2	GND	Negative power
PIN3	SET	Set pin /TTL level@3.3V，high level or suspending is normal working status, while low level is sleeping mode.
PIN4	RX	Serial port receiving pin/TTL level@3.3V
PIN5	TX	Serial port sending pin/TTL level@3.3V
PIN6	RESET	Module reset signal /TTL level@3.3V，low reset.
PIN7/8	NC	

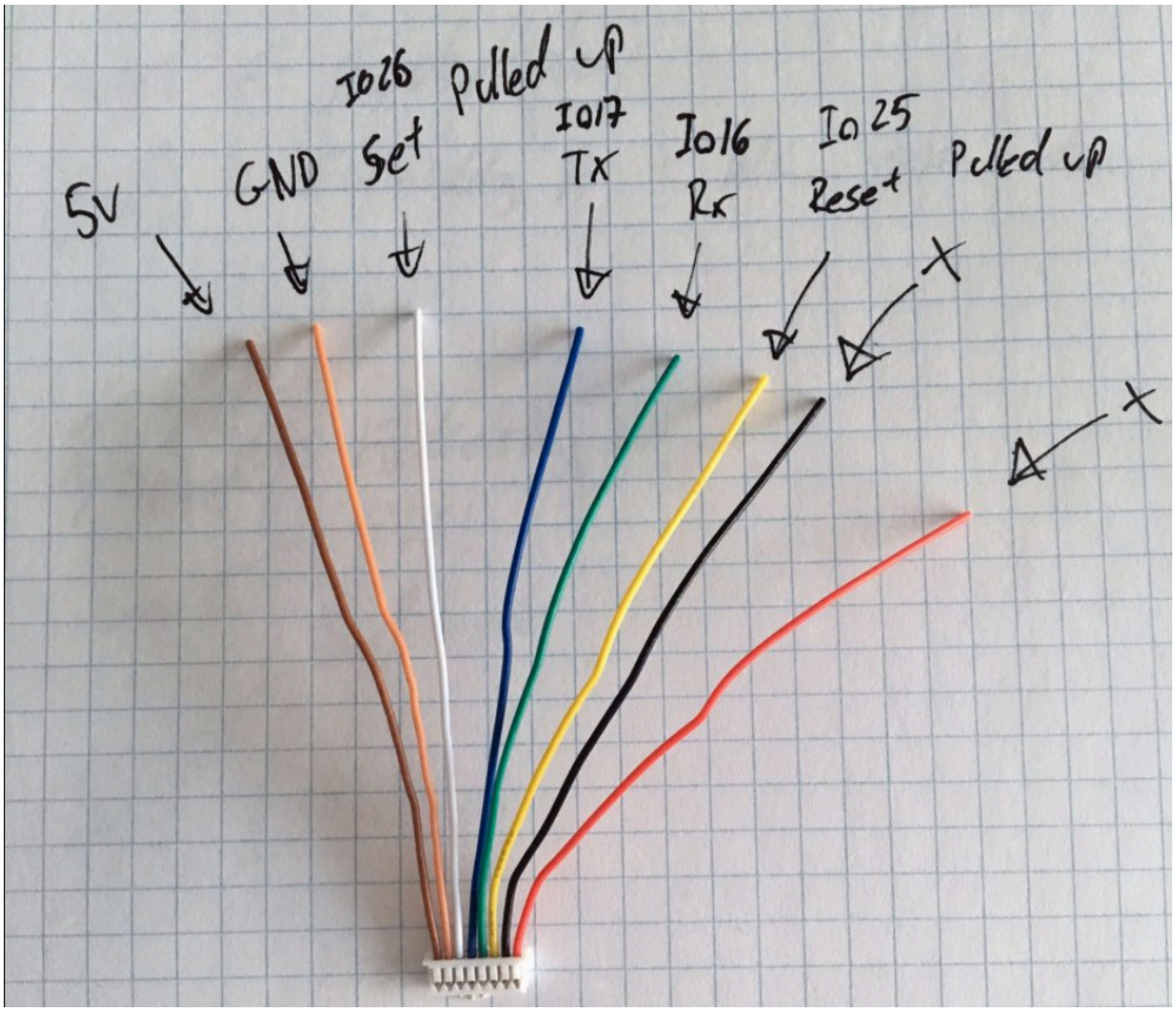


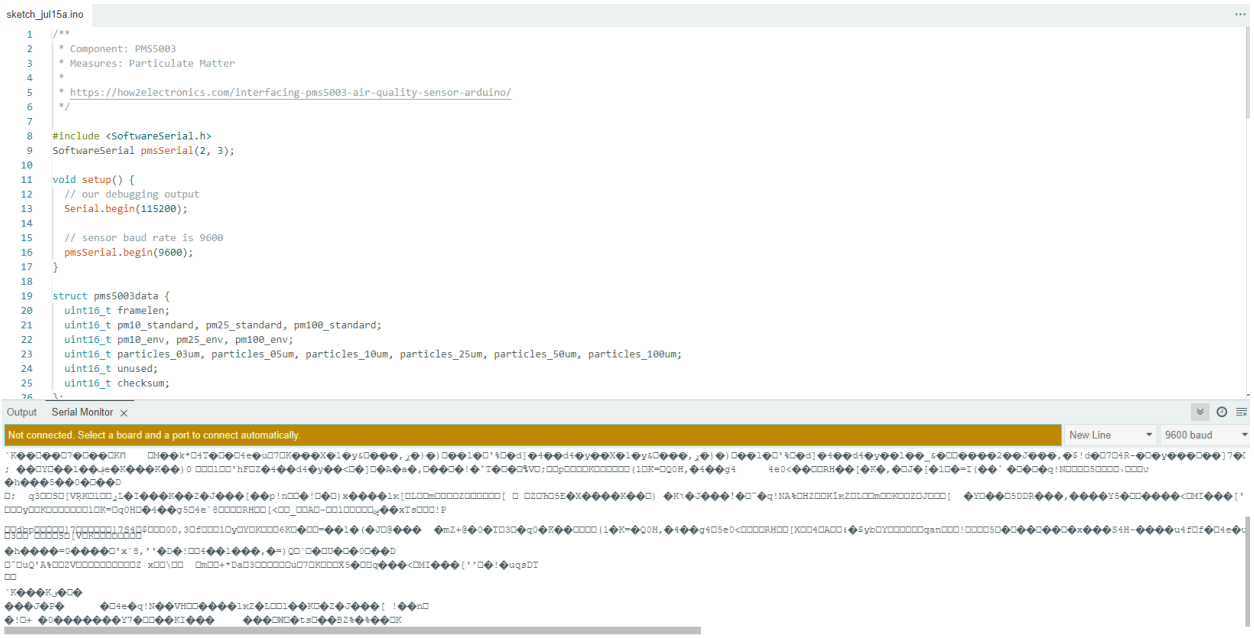
- New issue: After stripping the wire, they are frayed, so we need to twist or solder them to fix


8 7 6 5 4 3 2 1
RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
6
5
4
3
2
TX→1
RX←0
DIGITAL (PWM=~)
L
TX
RX
Arduino
UNO
ON
ICSP
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5

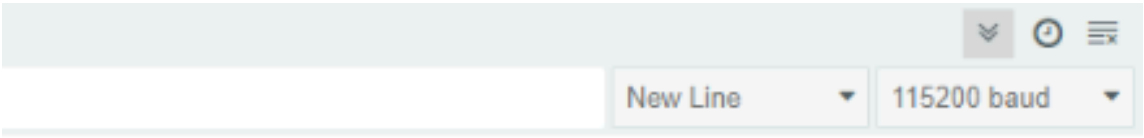

Digital 2
Rx
Digital 3
Tx
GND




5V
GND
IO26
Set
Pulled up
IO17
TX
IO16
Rx
IO25
Reset
Pulled up
X
X



- Change baud rate (change higher from 9600 baud to 115200 baud) → located on upper line of serial monitor



Sample Data:

```
---------------------------------------

Concentration Units (standard)

PM 1.0: 0           PM 2.5: 3           PM 10: 3

---------------------------------------

Concentration Units (environmental)

PM 1.0: 0           PM 2.5: 3           PM 10: 3
```

Particles > 0.3um / 0.1L air:258

Particles > 0.5um / 0.1L air:212

Particles > 1.0um / 0.1L air:28

Particles > 2.5um / 0.1L air:4

Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

Concentration Units (standard)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Concentration Units (environmental)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Particles > 0.3um / 0.1L air:272

Particles > 0.5um / 0.1L air:218

Particles > 1.0um / 0.1L air:36

Particles > 2.5um / 0.1L air:4

Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

Concentration Units (standard)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Concentration Units (environmental)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Particles > 0.3um / 0.1L air:272

Particles > 0.5um / 0.1L air:218

Particles > 1.0um / 0.1L air:36

Particles > 2.5um / 0.1L air:4

Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

Concentration Units (standard)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Concentration Units (environmental)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Particles > 0.3um / 0.1L air:262

Particles > 0.5um / 0.1L air:212

Particles > 1.0um / 0.1L air:34

Particles > 2.5um / 0.1L air:4

Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

Concentration Units (standard)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Concentration Units (environmental)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Particles > 0.3um / 0.1L air:282

Particles > 0.5um / 0.1L air:226

Particles > 1.0um / 0.1L air:44

Particles > 2.5um / 0.1L air:4

Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

Concentration Units (standard)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Concentration Units (environmental)

PM 1.0: 0 PM 2.5: 3 PM 10: 3

Particles > 0.3um / 0.1L air:276

Particles > 0.5um / 0.1L air:222

Particles > 1.0um / 0.1L air:42

Particles > 2.5um / 0.1L air:4

```
Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

--------------------------------------

Concentration Units (standard)

PM 1.0: 0          PM 2.5: 3          PM 10: 3

--------------------------------------

Concentration Units (environmental)

PM 1.0: 0          PM 2.5: 3          PM 10: 3

--------------------------------------

Particles > 0.3um / 0.1L air:286

Particles > 0.5um / 0.1L air:230

Particles > 1.0um / 0.1L air:40

Particles > 2.5um / 0.1L air:2

Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

--------------------------------------
```

Lab 3

```
--------------------------------------
Concentration Units (standard)
PM 1.0: 0          PM 2.5: 0          PM 10: 0

--------------------------------------

Concentration Units (environmental)

PM 1.0: 0          PM 2.5: 0          PM 10: 0

--------------------------------------
```

Particles > 0.3um / 0.1L air:12

Particles > 0.5um / 0.1L air:10

Particles > 1.0um / 0.1L air:4

Particles > 2.5um / 0.1L air:0

Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

Concentration Units (standard)

PM 1.0: 0 PM 2.5: 0 PM 10: 0

Concentration Units (environmental)

PM 1.0: 0 PM 2.5: 0 PM 10: 0

Particles > 0.3um / 0.1L air:14

Particles > 0.5um / 0.1L air:10

Particles > 1.0um / 0.1L air:4

Particles > 2.5um / 0.1L air:0

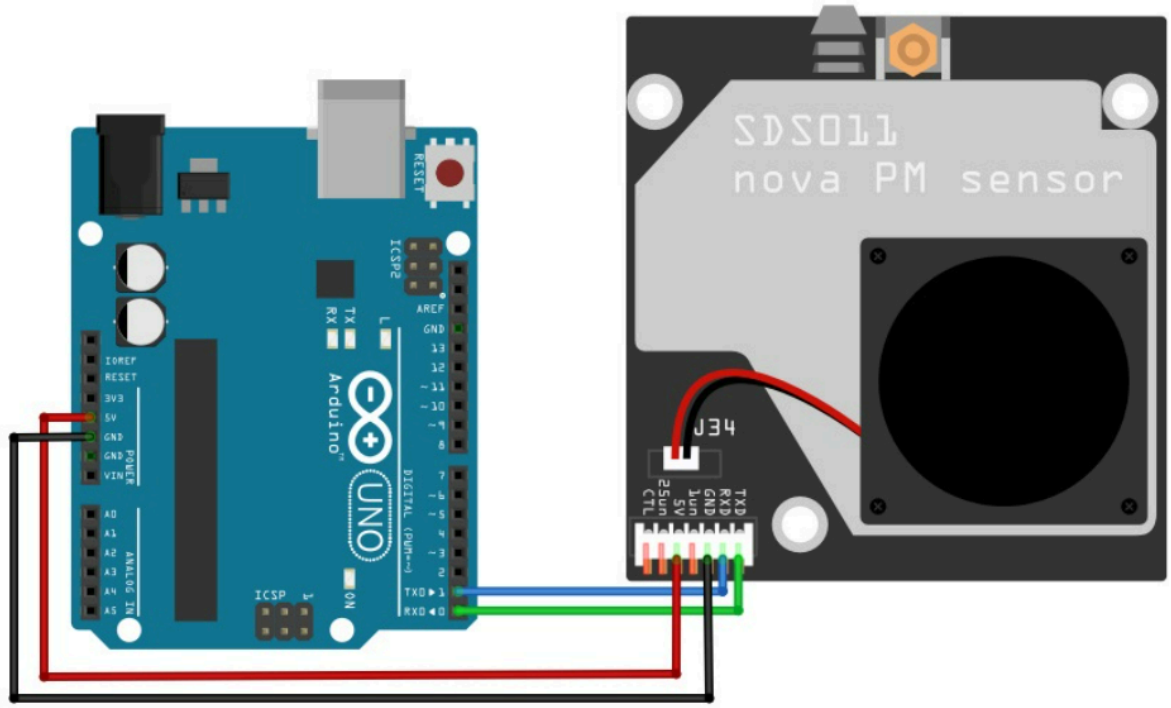
Particles > 5.0um / 0.1L air:0

Particles > 10.0 um / 0.1L air:0

SDS011

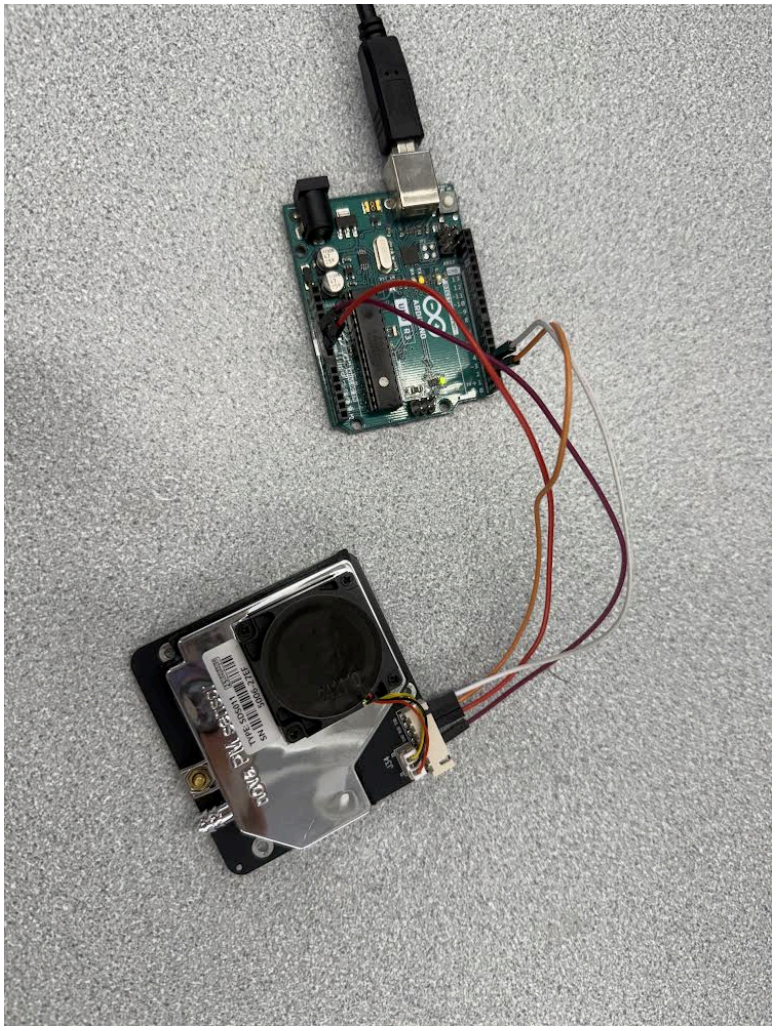
SDS011

- USB to serial converter? The sensor package includes a USB serial adapter, which serves to provide power to the sensor and enable data communication between the sensor and the Arduino.



Issues:

- Inconsistent timing for testing → changed to every 30 seconds
- Sensor put to sleep → deleted sleep code



```
Mode: query

SDS011 dust sensor

SDS011 Firmware version [year.month.day]: 23.6.26

Mode: query

----------------SDS-------------------------

PM2.5 = 0.20, PM10 = 0.40

----------------SDS-------------------------

----------------SDS-------------------------

PM2.5 = 0.20, PM10 = 0.40

----------------SDS-------------------------
```

----------SDS----------

PM2.5 = 0.20, PM10 = 0.40

----------SDS----------

----------SDS----------

PM2.5 = 0.50, PM10 = 0.80

----------SDS----------

----------SDS----------

PM2.5 = 0.30, PM10 = 0.40

----------SDS----------

----------SDS----------

PM2.5 = 0.30, PM10 = 0.70

----------SDS----------
----------SDS----------

PM2.5 = 0.30, PM10 = 1.10

----------SDS----------

----------SDS----------

PM2.5 = 0.20, PM10 = 0.70

----------SDS----------
----------SDS----------

PM2.5 = 0.20, PM10 = 0.20

----------SDS----------

----------SDS----------

PM2.5 = 0.30, PM10 = 1.50

----------SDS----------

```
---------------SDS-------------------------

PM2.5 = 0.20, PM10 = 0.60

---------------SDS-------------------------

---------------SDS-------------------------

PM2.5 = 0.30, PM10 = 1.30

---------------SDS-------------------------

---------------SDS-------------------------

PM2.5 = 0.20, PM10 = 0.40

---------------SDS-------------------------

---------------SDS-------------------------

PM2.5 = 0.30, PM10 = 0.50

---------------SDS-------------------------
```

```
#include <Arduino.h>
#include <SdsDustSensor.h>
#include <SoftwareSerial.h>

/* SDS011 Dust Sensor */
const int SDS_RX_PIN = 3; // D3 -> SDS011 TX pin
const int SDS_TX_PIN = 4; // D4 -> SDS011 TX pin
SoftwareSerial softwareSerial(SDS_RX_PIN, SDS_TX_PIN);
SdsDustSensor sds(softwareSerial); //  additional parameters: retryDelayMs
and maxRetriesNotAvailable
const int MINUTE = 60000;
const int WAKEUP_WORKING_TIME = 30000; // 30 seconds.
const int MEASUREMENT_INTERVAL = 1 * MINUTE;


void setup() {
  Serial.begin(115200);
  Serial.println("SDS011 dust sensor");
  delay(500);
```

```
  /* SDS011 Dust Sensor */
  sds.begin();
  // Prints SDS011 firmware version:
  Serial.print("SDS011 ");
  Serial.println(sds.queryFirmwareVersion().toString());
  // Ensures SDS011 is in 'query' reporting mode:
  Serial.println(sds.setQueryReportingMode().toString());

}

void loop() {
// Wake up SDS011
sds.wakeup();
delay(WAKEUP_WORKING_TIME);
// Get data from SDS011
PmResult pm = sds.queryPm();
  if (pm.isOk()) {
    Serial.println("---------------SDS------------------------");
    Serial.print("PM2.5 = ");
    Serial.print(pm.pm25); // float, µg/m3
    Serial.print(", PM10 = ");
    Serial.println(pm.pm10);
    Serial.println("---------------SDS------------------------");
  } else {
    Serial.print("Could not read values from sensor, reason: ");
    Serial.println(pm.statusToString());
  }

}
```

BME680 + DHT11

BME680 + DHT11
*tested using arduino mega
Issue:

- DHT11 Readings kept showing “DHT Read Error, Code: -1”

Solution

- Instead of using the dht11 library (#include <dht11.h>), use Adafruit’s DHT library (#include “DHT.h”)
    - Changing libraries → pin set up is different

Test #1

```
----- Reading Sensors -----

BME680 Temperature = 26.55 *C

BME680 Pressure = 1022.33 hPa

BME680 Humidity = 50.63 %

BME680 Gas Resistance = 26.63 KOhms

BME680 Approx. Altitude = -75.24 m

DHT11 Temperature = 23.00 *C

DHT11 Humidity = 59.00 %

----- Reading Sensors -----

BME680 Temperature = 26.46 *C

BME680 Pressure = 1022.32 hPa

BME680 Humidity = 50.79 %

BME680 Gas Resistance = 29.49 KOhms

BME680 Approx. Altitude = -75.16 m

DHT11 Read Error, Code: -1

----- Reading Sensors -----

BME680 Temperature = 26.09 *C

BME680 Pressure = 1022.31 hPa
```

```
BME680 Humidity = 51.50 %

BME680 Gas Resistance = 43.10 KOhms

BME680 Approx. Altitude = -75.24 m

DHT11 Temperature = 23.00 *C

DHT11 Humidity = 60.00 %

----- Reading Sensors -----

BME680 Temperature = 26.03 *C

BME680 Pressure = 1022.31 hPa

BME680 Humidity = 51.65 %

BME680 Gas Resistance = 45.39 KOhms

BME680 Approx. Altitude = -75.08 m

DHT11 Read Error, Code: -1

----- Reading Sensors -----

BME680 Temperature = 25.97 *C

BME680 Pressure = 1022.31 hPa

BME680 Humidity = 51.78 %

BME680 Gas Resistance = 48.00 KOhms

BME680 Approx. Altitude = -75.16 m

DHT11 Read Error, Code: -1

----- Reading Sensors -----

BME680 Temperature = 25.91 *C

BME680 Pressure = 1022.31 hPa

BME680 Humidity = 51.90 %

BME680 Gas Resistance = 50.40 KOhms

BME680 Approx. Altitude = -75.16 m
```

```
DHT11 Read Error, Code: -1
```

Test #2

```
----- Reading Sensors -----

BME680 Temperature = 24.56 *C

BME680 Pressure = 1022.32 hPa

BME680 Humidity = 56.26 %

BME680 Gas Resistance = 110.73 KOhms

BME680 Approx. Altitude = -75.33 m

DHT11 Read Error, Code: -1

----- Reading Sensors -----

BME680 Temperature = 24.37 *C

BME680 Pressure = 1022.32 hPa

BME680 Humidity = 56.35 %

BME680 Gas Resistance = 112.83 KOhms

BME680 Approx. Altitude = -75.33 m

DHT11 Temperature = 24.00 *C

DHT11 Humidity = 57.00 %
```

Test #3

```
----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.88 %

BME680 Gas Resistance: 172.93 KOhms

BME680 Altitude: -74.17 m
```

DHT11 Temp: 23.50 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.87 %

BME680 Gas Resistance: 173.11 KOhms

BME680 Altitude: -74.08 m

DHT11 Temp: 23.10 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.19 hPa

BME680 Humidity: 57.89 %

BME680 Gas Resistance: 172.75 KOhms

BME680 Altitude: -74.08 m

DHT11 Temp: 23.90 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.89 %

BME680 Gas Resistance: 173.11 KOhms

BME680 Altitude: -74.17 m

DHT11 Temp: 23.60 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.89 %

BME680 Gas Resistance: 173.65 KOhms

BME680 Altitude: -74.08 m

DHT11 Temp: 23.70 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.90 %

BME680 Gas Resistance: 172.39 KOhms

BME680 Altitude: -74.08 m

DHT11 Temp: 23.10 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.88 %

BME680 Gas Resistance: 173.11 KOhms

BME680 Altitude: -74.08 m

DHT11 Temp: 23.20 *C

```
DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.60 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.91 %

BME680 Gas Resistance: 173.47 KOhms

BME680 Altitude: -74.17 m

DHT11 Temp: 23.20 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.17 hPa

BME680 Humidity: 57.91 %

BME680 Gas Resistance: 172.75 KOhms

BME680 Altitude: -74.08 m

DHT11 Temp: 23.30 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.92 %

BME680 Gas Resistance: 173.83 KOhms

BME680 Altitude: -74.17 m

DHT11 Temp: 23.30 *C

DHT11 Humidity: 59.00 %
```

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.18 hPa

BME680 Humidity: 57.91 %

BME680 Gas Resistance: 172.93 KOhms

BME680 Altitude: -74.08 m

DHT11 Temp: 23.50 *C

DHT11 Humidity: 59.00 %

----- Sensor Readings -----

BME680 Temp: 23.59 *C

BME680 Pressure: 1022.17 hPa

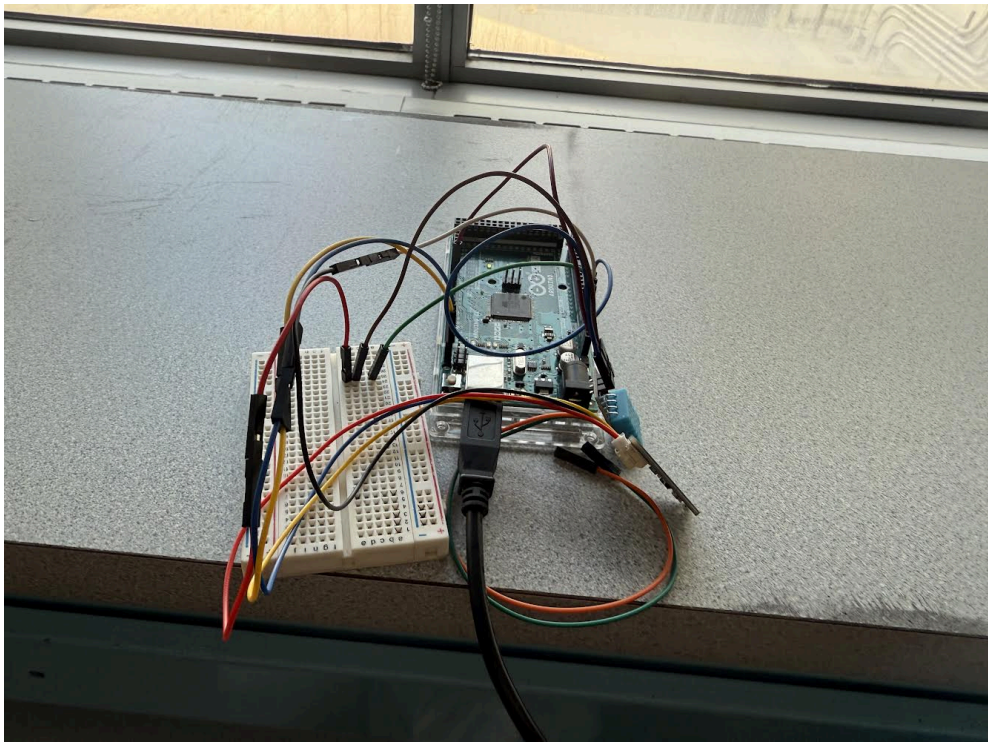
BME680 Humidity: 57.90 %

BME680 Gas Resistance: 173.83 KOhms

BME680 Altitude: -74.00 m

DHT11 Temp: 23.20 *C

DHT11 Humidity: 59.00 %



Code:

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"
#include <DHT.h>

// ----- BME680 Setup -----
#define SEALEVELPRESSURE_HPA (1013.25)
Adafruit_BME680 bme;  // BME680 using I2C

// ----- DHT11 Setup -----
#define DHTPIN 2          // DHT11 data pin
#define DHTTYPE DHT11     // DHT sensor type
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  while (!Serial);
```

```
  Serial.println(F("Starting BME680 + DHT11 sensor readings"));

  // Initialize DHT11
  dht.begin();

  // Initialize BME680
  if (!bme.begin()) {
    Serial.println("Could not find a valid BME680 sensor, check wiring!");
    while (1);
  }

  // Configure BME680 oversampling and filter
  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setIIRFilterSize(BME680_FILTER_SIZE_3);
  bme.setGasHeater(320, 150); // 320°C for 150 ms
}

void loop() {
  Serial.println(F("----- Sensor Readings -----"));

  // ---- BME680 Reading ----
  if (bme.performReading()) {
    Serial.print("BME680 Temp: ");
    Serial.print(bme.temperature);
    Serial.println(" *C");

    Serial.print("BME680 Pressure: ");
    Serial.print(bme.pressure / 100.0);
    Serial.println(" hPa");

    Serial.print("BME680 Humidity: ");
    Serial.print(bme.humidity);
    Serial.println(" %");

    Serial.print("BME680 Gas Resistance: ");
    Serial.print(bme.gas_resistance / 1000.0);
    Serial.println(" KOhms");
```

```
    Serial.print("BME680 Altitude: ");
    Serial.print(bme.readAltitude(SEALEVELPRESSURE_HPA));
    Serial.println(" m");
  } else {
    Serial.println("Failed to perform BME680 reading :(");
  }

  // ---- DHT11 Reading ----
  float dhtTemp = dht.readTemperature();
  float dhtHum = dht.readHumidity();

  if (isnan(dhtTemp) || isnan(dhtHum)) {
    Serial.println("Failed to read from DHT11 sensor!");
  } else {
    Serial.print("DHT11 Temp: ");
    Serial.print(dhtTemp);
    Serial.println(" *C");

    Serial.print("DHT11 Humidity: ");
    Serial.print(dhtHum);
    Serial.println(" %");
  }

  Serial.println(); // Space between readings
  delay(2000);      // Wait 2 seconds before next loop
}
```

BME680 + PMS5003

BME680 + PMS5003 (need to fix baud rate issue)

Issues:

- Only prints BME680 values

```
---- BME680 Readings ----

Temperature: 24.06 °C

Pressure: 1017.59 hPa

Humidity: 57.97 %

Gas Resistance: 101.20 KOhms

--------------------------

---- BME680 Readings ----

Temperature: 24.06 °C

Pressure: 1017.59 hPa

Humidity: 58.05 %

Gas Resistance: 101.32 KOhms

--------------------------

---- BME680 Readings ----

Temperature: 24.05 °C

Pressure: 1017.58 hPa

Humidity: 58.04 %

Gas Resistance: 101.75 KOhms

--------------------------
```

Testing 7/31:

```
---- SENSOR READINGS ----

[BME680]

Temp: 24.79 °C
```

```
Pressure: 1011.45 hPa

Humidity: 56.77 %

Gas Resistance: 127.19 KOhms

[PMS5004] Waiting for data...

---- SENSOR READINGS ----

[BME680]

Temp: 24.79 °C

Pressure: 1011.45 hPa

Humidity: 56.84 %

Gas Resistance: 126.90 KOhms

[PMS5004] Waiting for data...
```

Testing 2 7/31: IT WORKS!

```
---- SENSOR READINGS ----

[BME680]

Temp: 24.94 °C

Pressure: 1011.32 hPa

Humidity: 55.30 %

Gas Resistance: 137.57 KOhms

[PMS5003]

PM1.0: 5 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

---- SENSOR READINGS ----

[BME680]
```

Temp: 24.94 °C

Pressure: 1011.32 hPa

Humidity: 55.32 %

Gas Resistance: 138.03 KOhms

[PMS5003]

PM1.0: 6 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.94 °C

Pressure: 1011.32 hPa

Humidity: 55.28 %

Gas Resistance: 137.57 KOhms

[PMS5003]

PM1.0: 6 µg/m3

PM2.5: 6 µg/m3

PM10: 3 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.95 °C

Pressure: 1011.32 hPa

Humidity: 55.26 %

Gas Resistance: 137.69 KOhms

[PMS5003]

PM1.0: 5 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.94 °C

Pressure: 1011.32 hPa

Humidity: 55.24 %

Gas Resistance: 137.80 KOhms

[PMS5003]

PM1.0: 7 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.94 °C

Pressure: 1011.33 hPa

Humidity: 55.24 %

Gas Resistance: 137.12 KOhms

[PMS5003]

PM1.0: 7 µg/m3

PM2.5: 7 µg/m3

PM10: 2 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.94 °C

Pressure: 1011.33 hPa

Humidity: 55.24 %

Gas Resistance: 137.91 KOhms

[PMS5003]

PM1.0: 7 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.93 °C

Pressure: 1011.34 hPa

Humidity: 55.24 %

Gas Resistance: 137.69 KOhms

[PMS5003]

PM1.0: 6 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.93 °C

Pressure: 1011.34 hPa

Humidity: 55.24 %

Gas Resistance: 138.03 KOhms

[PMS5003]

PM1.0: 6 µg/m3

PM2.5: 6 µg/m3

PM10: 2 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.93 °C

Pressure: 1011.35 hPa

Humidity: 55.25 %

Gas Resistance: 137.57 KOhms

[PMS5003]

PM1.0: 6 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

---- SENSOR READINGS ----

[BME680]

Temp: 24.93 °C

Pressure: 1011.35 hPa

Humidity: 55.26 %

Gas Resistance: 137.46 KOhms

[PMS5003]

PM1.0: 5 µg/m3

PM2.5: 66 µg/m3

PM10: 19712 µg/m3

```
---- SENSOR READINGS ----

[BME680]

Temp: 24.93 °C

Pressure: 1011.34 hPa

Humidity: 55.26 %

Gas Resistance: 137.46 KOhms

[PMS5003]

PM1.0: 4 µg/m3

PM2.5: 4 µg/m3

PM10: 1 µg/m3
```

Code:

```
#include <Wire.h>
#include <SoftwareSerial.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME680.h>

// Initialize BME680 sensor on I2C
Adafruit_BME680 bme;

// Define SoftwareSerial for PMS5003 sensor
SoftwareSerial pmsSerial(10, 11);  // RX, TX pins for PMS5003

const int PMS_FRAME_SIZE = 32;  // PMS5003 sends 32-byte data frames

void setup() {
  Serial.begin(115200);      // Serial monitor for debugging
  pmsSerial.begin(9600);     // PMS5003 communicates at 9600 baud

  // Initialize BME680 sensor
  if (!bme.begin()) {
    Serial.println("Could not find BME680 sensor!");
    while (1);  // Halt if sensor not found
  }
```

```
  // Configure BME680 oversampling and heater settings
  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setGasHeater(320, 150);  // 320°C for 150 ms

  Serial.println("Sensors initialized.");
}

void loop() {
  Serial.println("---- SENSOR READINGS ----");

  // Read and print BME680 data
  if (bme.performReading()) {
    Serial.println("[BME680]");
    Serial.print("Temp: "); Serial.print(bme.temperature);
Serial.println(" °C");
    Serial.print("Pressure: "); Serial.print(bme.pressure / 100.0);
Serial.println(" hPa");
    Serial.print("Humidity: "); Serial.print(bme.humidity);
Serial.println(" %");
    Serial.print("Gas Resistance: "); Serial.print(bme.gas_resistance /
1000.0); Serial.println(" KOhms");
  } else {
    Serial.println("[BME680] Reading failed.");
  }

  // Read and print PMS5003 data
  readPMS5003();

  Serial.println();
  delay(3000);  // Wait before next reading
}

void readPMS5003() {
  // Check if enough bytes are available for a full frame
  if (pmsSerial.available() >= PMS_FRAME_SIZE) {

    // FRAME SYNC: Look for the first byte of the frame header (0x42)
    while (pmsSerial.read() != 0x42) {
```

```
      // If not enough bytes left to complete a frame, exit early
      if (pmsSerial.available() < PMS_FRAME_SIZE) {
        Serial.println("[PMS5003] Frame sync failed.");
        return;
      }
    }

    // Confirm second header byte (should be 0x4D)
    if (pmsSerial.read() != 0x4D) {
      Serial.println("[PMS5003] Invalid second header byte.");
      return;
    }

    // Read remaining 30 bytes of the frame (we already read 2)
    uint8_t buffer[PMS_FRAME_SIZE - 2];
    pmsSerial.readBytes(buffer, PMS_FRAME_SIZE - 2);

    // Extract PM values from buffer (standard frame format)
    uint16_t pm1_0 = buffer[4] << 8 | buffer[5];
    uint16_t pm2_5 = buffer[6] << 8 | buffer[7];
    uint16_t pm10  = buffer[8] << 8 | buffer[9];

    // Print PM readings
    Serial.println("[PMS5003]");
    Serial.print("PM1.0: "); Serial.print(pm1_0); Serial.println("
µg/m3");
    Serial.print("PM2.5: "); Serial.print(pm2_5); Serial.println("
µg/m3");
    Serial.print("PM10: ");  Serial.print(pm10);  Serial.println("
µg/m3");

  } else {
    // Not enough data yet — wait for next loop

    Serial.println("[PMS5003] Waiting for data...");
  }
}
```

BME680 + MQ135

BME680 + MQ135

Issues:

- Tried to use breadboard and put both ground pins into 1 on the the Arduino → ended up shorting the Arduino
    - Solution: Separate Arduino GND pin for each sensor (only use breadboard for both VCC → 5V)

Testing:

```
-----------------------

Temperature = 25.01 *C

Pressure = 1021.23 hPa

Humidity = 53.79 %

Gas Resistance = 175.66 KOhms

Approx. Altitude = -66.23 m

MQ135 Analog Value = 66

MQ135 Digital Value = 0

-----------------------

Temperature = 25.00 *C

Pressure = 1021.22 hPa

Humidity = 53.80 %

Gas Resistance = 176.59 KOhms

Approx. Altitude = -66.23 m

MQ135 Analog Value = 65

MQ135 Digital Value = 0

-----------------------

Temperature = 25.00 *C

Pressure = 1021.22 hPa
```

Humidity = 53.81 %

Gas Resistance = 175.66 KOhms

Approx. Altitude = -66.31 m

MQ135 Analog Value = 66

MQ135 Digital Value = 0

Temperature = 25.00 *C

Pressure = 1021.22 hPa

Humidity = 53.81 %

Gas Resistance = 176.03 KOhms

Approx. Altitude = -66.31 m

MQ135 Analog Value = 65

MQ135 Digital Value = 0

Temperature = 24.99 *C

Pressure = 1021.24 hPa

Humidity = 53.84 %

Gas Resistance = 176.21 KOhms

Approx. Altitude = -66.40 m

MQ135 Analog Value = 66

MQ135 Digital Value = 0

Temperature = 24.99 *C

Pressure = 1021.25 hPa

Humidity = 53.85 %

Gas Resistance = 176.03 KOhms

Approx. Altitude = -66.40 m

MQ135 Analog Value = 65

MQ135 Digital Value = 0

Temperature = 24.98 *C

Pressure = 1021.25 hPa

Humidity = 53.87 %

Gas Resistance = 176.40 KOhms

Approx. Altitude = -66.48 m

MQ135 Analog Value = 65

MQ135 Digital Value = 0

Temperature = 24.98 *C

Pressure = 1021.26 hPa

Humidity = 53.88 %

Gas Resistance = 176.77 KOhms

Approx. Altitude = -66.48 m

MQ135 Analog Value = 65

MQ135 Digital Value = 0

Temperature = 24.98 *C

Pressure = 1021.25 hPa

Humidity = 53.92 %

```
Gas Resistance = 176.40 KOhms

Approx. Altitude = -66.56 m

MQ135 Analog Value = 65

MQ135 Digital Value = 0

-----------------------

Temperature = 24.97 *C

Pressure = 1021.26 hPa

Humidity = 53.94 %

Gas Resistance = 176.40 KOhms

Approx. Altitude = -66.56 m

MQ135 Analog Value = 65

MQ135 Digital Value = 0
```

Code:

```
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"

#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10

#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME680 bme; // I2C

int mq135AnalogPin = A0;
int mq135DigitalPin = 2;

void setup() {
```

```
  Serial.begin(9600);
  while (!Serial);
  Serial.println(F("Combined BME680 + MQ135 Test"));

  // BME680 initialization
  if (!bme.begin()) {
    Serial.println("Could not find a valid BME680 sensor, check wiring!");
    while (1);
  }

  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setIIRFilterSize(BME680_FILTER_SIZE_3);
  bme.setGasHeater(320, 150); // 320*C for 150 ms

  // MQ135 pin setup
  pinMode(mq135DigitalPin, INPUT);
}

void loop() {
  // Read BME680 sensor data
  if (!bme.performReading()) {
    Serial.println("Failed to perform BME680 reading :(");
  } else {
    Serial.print("Temperature = ");
    Serial.print(bme.temperature);
    Serial.println(" *C");

    Serial.print("Pressure = ");
    Serial.print(bme.pressure / 100.0);
    Serial.println(" hPa");

    Serial.print("Humidity = ");
    Serial.print(bme.humidity);
    Serial.println(" %");

    Serial.print("Gas Resistance = ");
    Serial.print(bme.gas_resistance / 1000.0);
    Serial.println(" KOhms");
```

```
    Serial.print("Approx. Altitude = ");
    Serial.print(bme.readAltitude(SEALEVELPRESSURE_HPA));
    Serial.println(" m");
  }

  Serial.println();

  // Read MQ135 sensor data
  int sensorValue = analogRead(mq135AnalogPin); // analog reading
  int digitalValue = digitalRead(mq135DigitalPin); // digital reading

  Serial.print("MQ135 Analog Value = ");
  Serial.println(sensorValue);

  Serial.print("MQ135 Digital Value = ");
  Serial.println(digitalValue);

  Serial.println("-----------------------");

  delay(2000); // 2 seconds delay between readings
}
```

MQ135 + DHT11

MQ135 + DHT11

Testing:

```
---- MQ135 Sensor ----

Analog Value: 59

---- DHT11 Sensor ----

Humidity (%): 58.00

Temperature (°C): 24.00

----------------------

---- MQ135 Sensor ----

Analog Value: 59

---- DHT11 Sensor ----

Humidity (%): 58.00

Temperature (°C): 24.00

----------------------

---- MQ135 Sensor ----

Analog Value: 59

---- DHT11 Sensor ----

Humidity (%): 58.00

Temperature (°C): 24.0

---- MQ135 Sensor ----

Analog Value: 59

---- DHT11 Sensor ----

Humidity (%): 58.00

Temperature (°C): 24.00
```

```
----------------------

---- MQ135 Sensor ----

Analog Value: 59

---- DHT11 Sensor ----

Humidity (%): 58.00

Temperature (°C): 24.00

----------------------

---- MQ135 Sensor ----

Analog Value: 59

---- DHT11 Sensor ----

Humidity (%): 58.00

Temperature (°C): 24.00

----------------------
```

Code:

```
/**
 * Components: MQ135 and DHT11
 * MQ135 Measures: CO2, benzyne (analog pin A0)
 * DHT11 Measures: Temperature and Humidity (digital pin 4)
 */

#include <dht11.h>

#define MQ135_ANALOG_PIN A0
#define DHT11_PIN 4

int mq135AnalogValue;
dht11 DHT11;

void setup() {
  Serial.begin(9600);
```

```
}

void loop() {
  // MQ135 Analog Reading
  mq135AnalogValue = analogRead(MQ135_ANALOG_PIN);

  Serial.println("---- MQ135 Sensor ----");
  Serial.print("Analog Value: ");
  Serial.println(mq135AnalogValue);

  // DHT11 Reading
  int chk = DHT11.read(DHT11_PIN);

  Serial.println("---- DHT11 Sensor ----");
  Serial.print("Humidity (%): ");
  Serial.println((float)DHT11.humidity, 2);

  Serial.print("Temperature (°C): ");
  Serial.println((float)DHT11.temperature, 2);

  Serial.println("----------------------\n");

  delay(2000); // Wait 2 seconds before the next reading
}
```