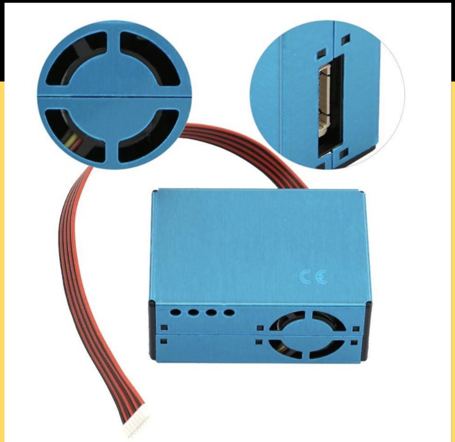
Plantower

Particulate Matter Sensor

By - Samriddhi, Jake, Michael, Nicholas

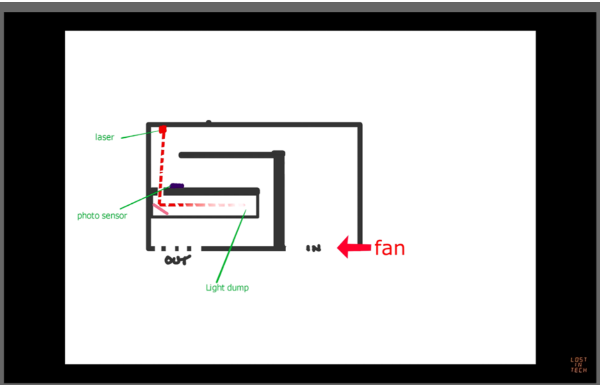
What is a Plantower?

PMS5003 is a kind of digital and universal particle concentration sensor, which can be used to obtain the number of suspended particles in the air and output them in the form of digital interface.



How Does it Work?

- The PMS5003 is a nephelometer, meaning that it records suspended particulates by shining a light beam into the cavity. The reflected light is incident upon a light detector set to one side (often 90°) of the source beam.
- The particle density is then a function of the light reflected into the detector and the particle mass is derived from it, assuming certain properties of the particles, such as shape, color and reflectivity, among others (Mie theory - EM scattering from spherical objects).



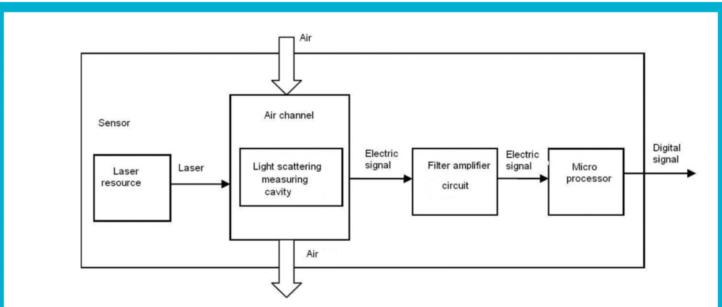
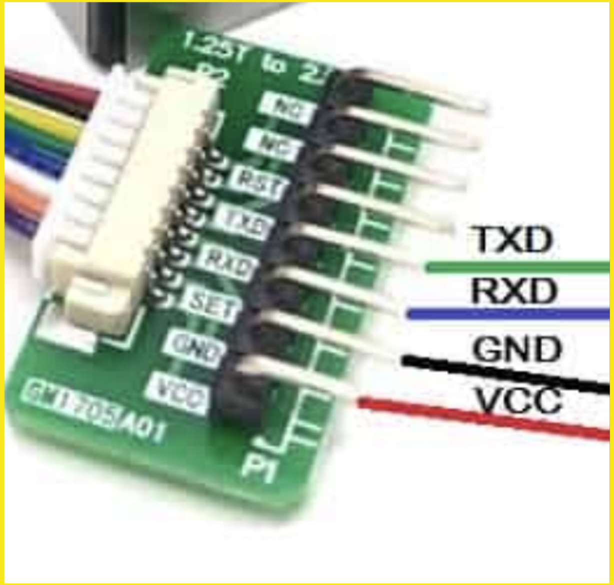


Figure 1 Functional block diagram of sensor

Breakout Pins



- VCC - 5V (fan)
- Ground
- TXD: UART/TTL Receive
- RXD: UART/TTL Transmit

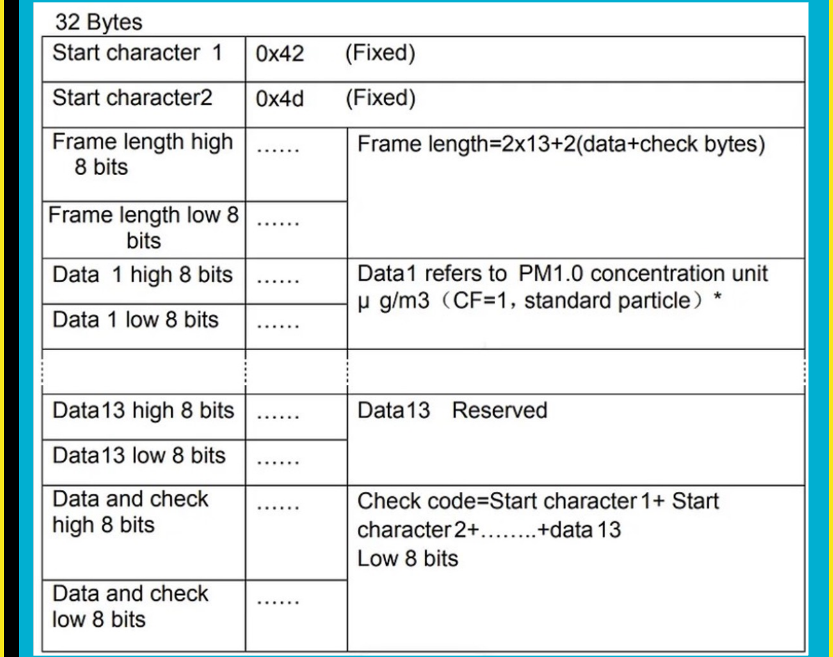
Data at 3.3V

What is UART?

Universal Asynchronous Receiver-transmitter

- A physical circuit in microprocessor
- Not synchronous
- Baud Rate needs to be same
- Uses only TX/RX

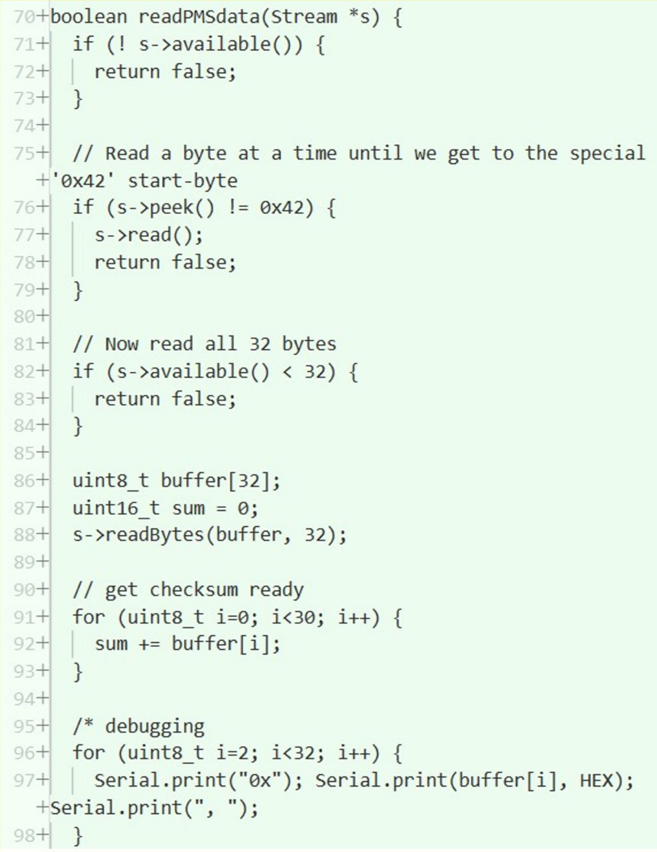
Superseded by USB, I2C, etc.



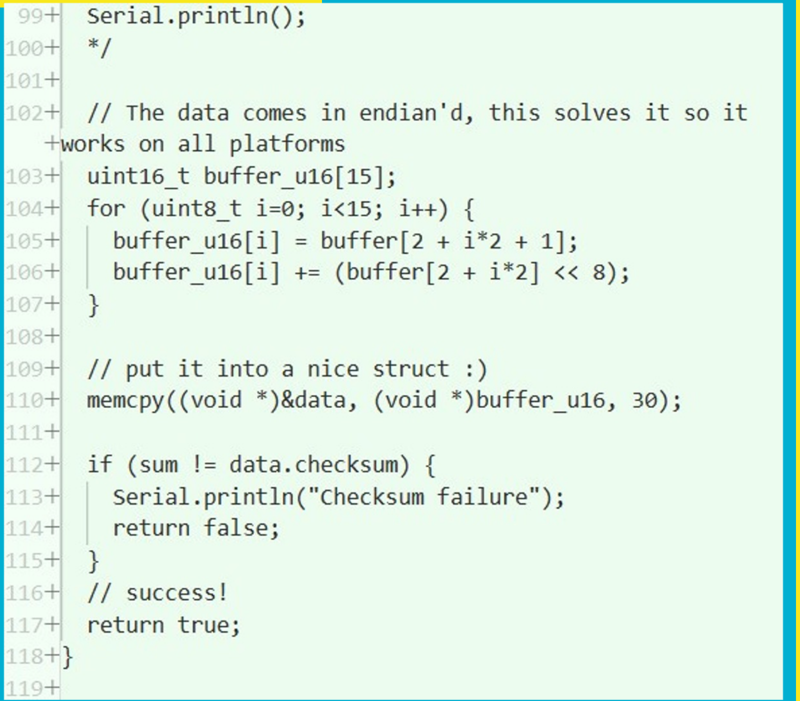
32 Bytes

Start character 1	0x42	(Fixed)
Start character2	0x4d	(Fixed)
Frame length high 8 bits	……	Frame length=2x13+2(data+check bytes)
Frame length low 8 bits	……	
Data 1 high 8 bits	……	Data1 refers to PM1.0 concentration unit μ g/m3（CF=1，standard particle）*
Data 1 low 8 bits	……	
Data13 high 8 bits	……	Data13 Reserved
Data13 low 8 bits	……	
Data and check high 8 bits	……	Check code=Start character1+ Start character2+……..+data 13 Low 8 bits
Data and check low 8 bits	……	

Why do we care?

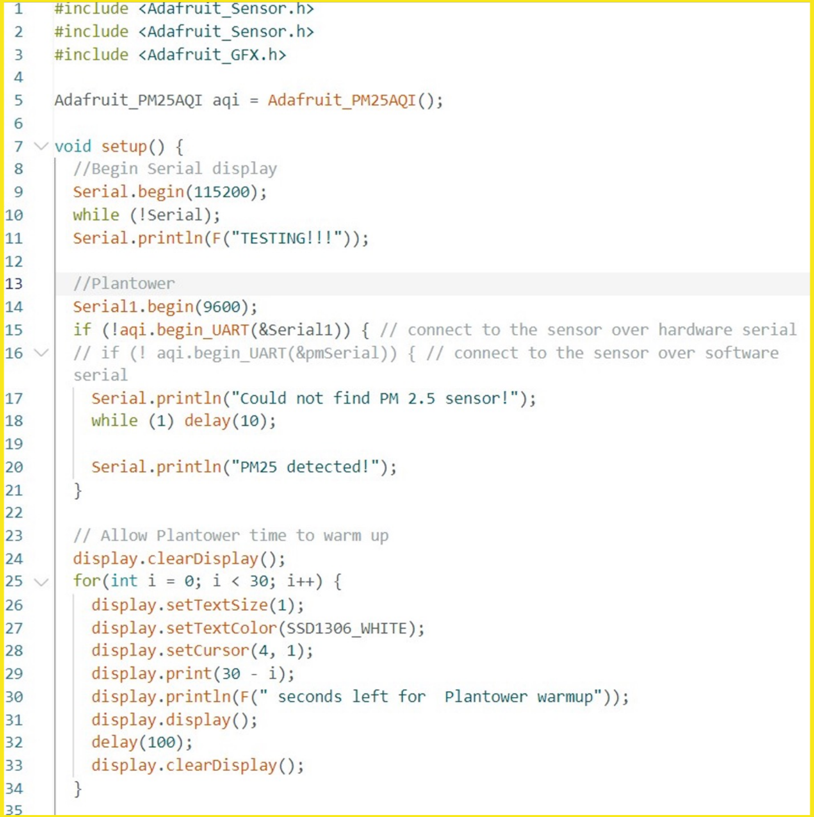


```
boolean readPMSdata(Stream *s) {
  if (! s->available()) {
    return false;
  }

  // Read a byte at a time until we get to the special '0x42' start-byte
  if (s->peek() != 0x42) {
    s->read();
    return false;
  }

  // Now read all 32 bytes
  if (s->available() < 32) {
    return false;
  }

  uint8_t buffer[32];
  uint16_t sum = 0;
  s->readBytes(buffer, 32);

  // get checksum ready
  for (uint8_t i=0; i<30; i++) {
    sum += buffer[i];
  }

  /* debugging
  for (uint8_t i=2; i<32; i++) {
    Serial.print("0x"); Serial.print(buffer[i], HEX); Serial.print(", ");
  }
```



```
  Serial.println();
  */

  // The data comes in endian'd, this solves it so it works on all platforms
  uint16_t buffer_u16[15];
  for (uint8_t i=0; i<15; i++) {
    buffer_u16[i] = buffer[2 + i*2 + 1];
    buffer_u16[i] += (buffer[2 + i*2] << 8);
  }

  // put it into a nice struct :)
  memcpy((void *)&data, (void *)buffer_u16, 30);

  if (sum != data.checksum) {
    Serial.println("Checksum failure");
    return false;
  }
  // success!
  return true;
}

```

Code Integration

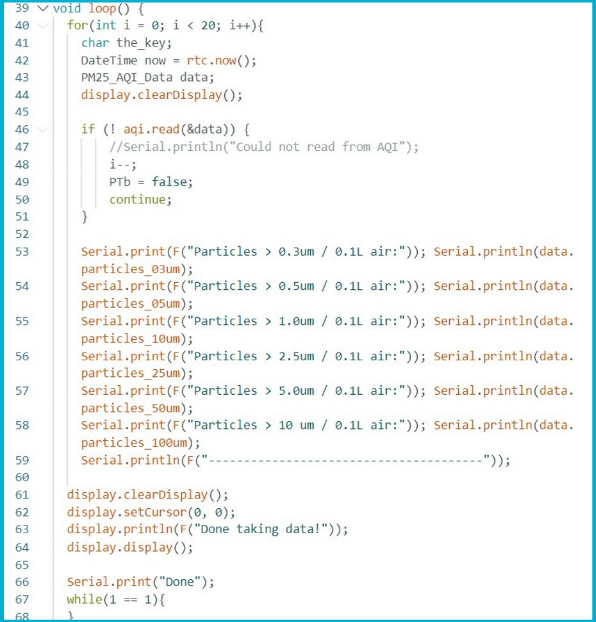


```
#include <Adafruit_Sensor.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_GFX.h>

Adafruit_PM25AQI aqi = Adafruit_PM25AQI();

void setup() {
  //Begin Serial display
  Serial.begin(115200);
  while (!Serial);
  Serial.println(F("TESTING!!!"));

  //Plantower
  Serial1.begin(9600);
  if (!aqi.begin_UART(&Serial1)) { // connect to the sensor over hardware serial
  // if (! aqi.begin_UART(&pmSerial)) { // connect to the sensor over software serial
    Serial.println("Could not find PM 2.5 sensor!");
    while (1) delay(10);

    Serial.println("PM25 detected!");
  }

  // Allow Plantower time to warm up
  display.clearDisplay();
  for(int i = 0; i < 30; i++) {
    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(4, 1);
    display.print(30 - i);
    display.println(F(" seconds left for  Plantower warmup"));
    display.display();
    delay(100);
    display.clearDisplay();
  }
```

Adafruit_PM25AQI library

- Default Library
- Not very stable
- Does not drain serial stream

Can also just use Hardware Serial



```
void loop() {
  for(int i = 0; i < 20; i++){
    char the_key;
    DateTime now = rtc.now();
    PM25_AQI_Data data;
    display.clearDisplay();

    if (! aqi.read(&data)) {
      //Serial.println("Could not read from AQI");
      i--;
      PTb = false;
      continue;
    }

    Serial.print(F("Particles > 0.3um / 0.1L air:")); Serial.println(data.particles_03um);
    Serial.print(F("Particles > 0.5um / 0.1L air:")); Serial.println(data.particles_05um);
    Serial.print(F("Particles > 1.0um / 0.1L air:")); Serial.println(data.particles_10um);
    Serial.print(F("Particles > 2.5um / 0.1L air:")); Serial.println(data.particles_25um);
    Serial.print(F("Particles > 5.0um / 0.1L air:")); Serial.println(data.particles_50um);
    Serial.print(F("Particles > 10 um / 0.1L air:")); Serial.println(data.particles_100um);
    Serial.println(F("------------------------------------"));

  display.clearDisplay();
  display.setCursor(0, 0);
  display.println(F("Done taking data!"));
  display.display();

  Serial.print("Done");
  while(1 == 1){
  }
```

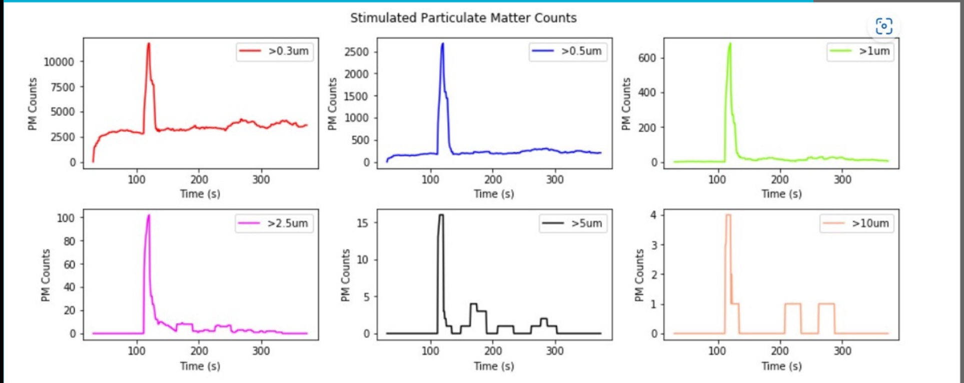
Sampling

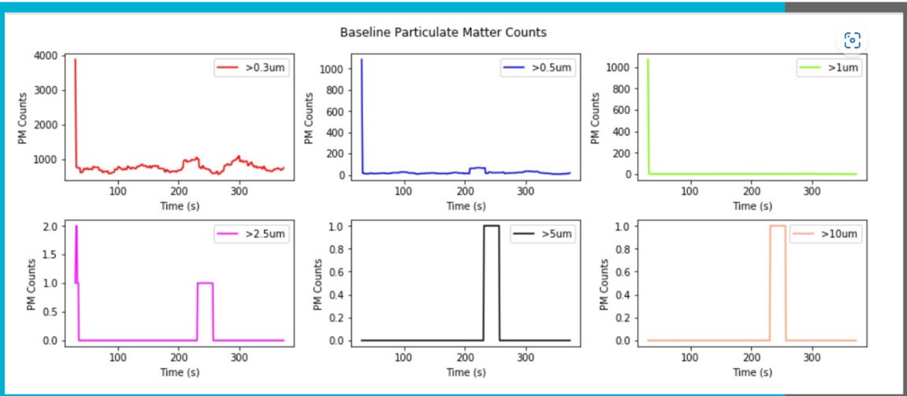
- Warm up time of 30 seconds.

- Active mode: Stable (2.3s) Fast(200-800 ms).

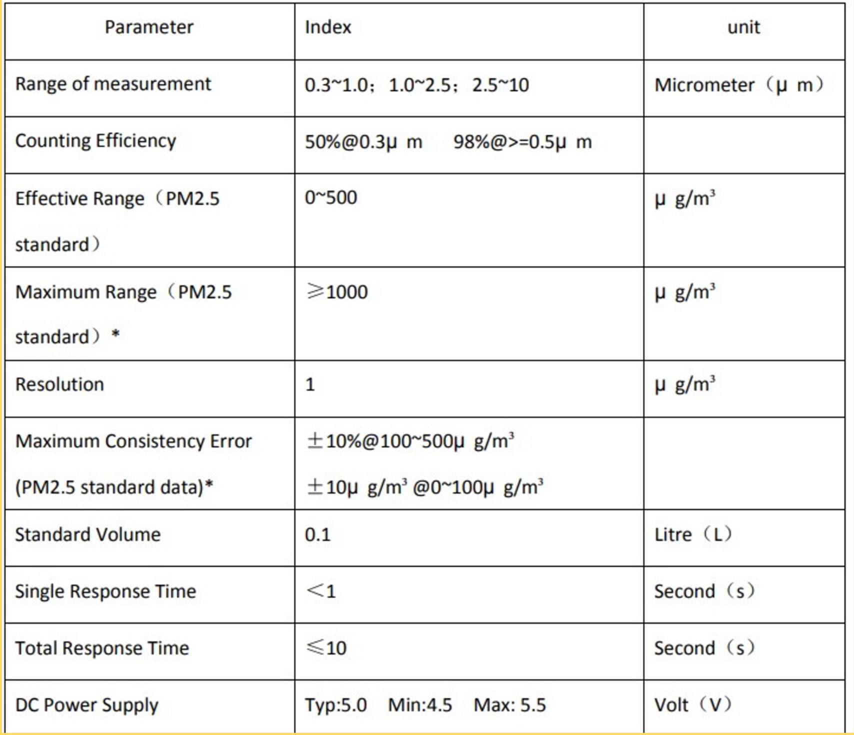
- Passive transmission: Upon request.

- PM concentration (standard and environmental) and raw counts.

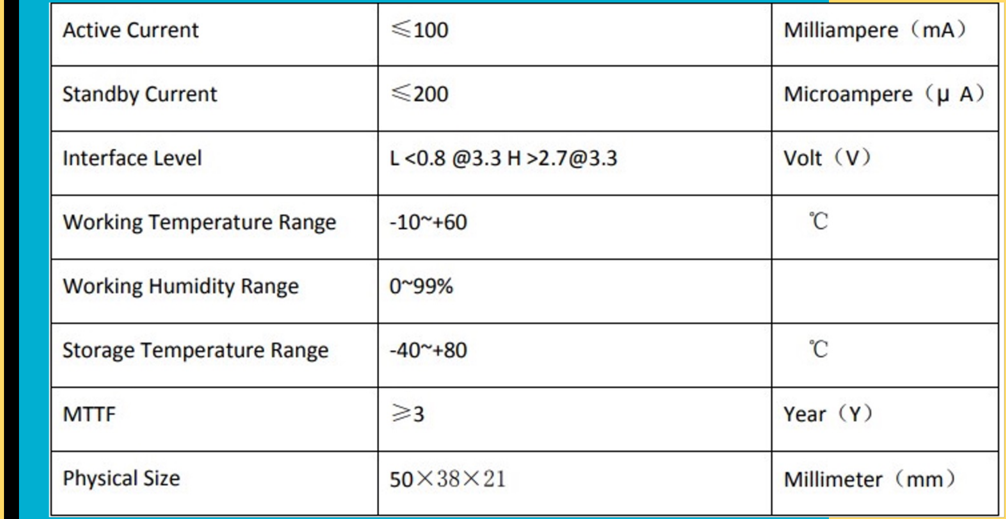




Technical Index



Parameter	Index	unit
Range of measurement	0.3~1.0；1.0~2.5；2.5~10	Micrometer（μ m）
Counting Efficiency	50%@0.3μ m　98%@>=0.5μ m	
Effective Range（PM2.5 standard）	0~500	μ g/m^3
Maximum Range（PM2.5 standard）*	⩾1000	μ g/m^3
Resolution	1	μ g/m^3
Maximum Consistency Error (PM2.5 standard data)*	±10%@100~500μ g/m^3 ±10μ g/m^3 @0~100μ g/m^3	
Standard Volume	0.1	Litre（L）
Single Response Time	＜1	Second（s）
Total Response Time	⩽10	Second（s）
DC Power Supply	Typ:5.0　Min:4.5　Max: 5.5	Volt（V）



Active Current	⩽100	Milliampere（mA）
Standby Current	⩽200	Microampere（μ A）
Interface Level	L <0.8 @3.3 H >2.7@3.3	Volt（V）
Working Temperature Range	-10~+60	℃
Working Humidity Range	0~99%	
Storage Temperature Range	-40~+80	℃
MTTF	⩾3	Year（Y）
Physical Size	50×38×21	Millimeter（mm）

Works Cited

https://docs.smartcitizen.me/Components/sensors/air/PM%20Sensors/#working-principle

https://www.aqmd.gov/docs/default-source/aq-spec/resources-page/plantower-pms5003-manual_v2-3.pdf

Debugging PM2.5 Sensor Library | by Kevin J. Walters | Nerd For Tech (medium.com)