Annexes

Sitographie

https://wiki.dfrobot.com/Gravity__Analog_Infrared_CO2_Sensor_For_Arduino_SKU__SEN0219

https://wiki.dfrobot.com/PM2.5_laser_dust_sensor_SKU_SEN0177

http://www.ac-lille.fr/hygienesecurite/site/documents/ressources/informations_diverses/rsd59.pdf

https://environnement.brussels/etat-de-lenvironnement/rapport-2011-2014/air/qualite-de-lair-concentration-en-particules-tres-fines

https://www.anses.fr/fr/system/files/AIR2012sa0093Ra.pdf

https://newscenter.lbl.gov/2012/10/17/elevated-indoor-carbon-dioxide-impairs-decision-making-performance/

Programme Arduino Capteur de particules

```
#include <Arduino.h>
#include <SoftwareSerial.h>
#include "rgb_lcd.h"
#include <SPI.h>
#include <SD.h>

#define LENG 31   //0x42 + 31 bytes equal to 32 bytes
unsigned char buf[LENG];
rgb_lcd lcd;
File myFile;

int PM01Value=0;          //define PM1.0 value of the air detector module
int PM2_5Value=0;         //define PM2.5 value of the air detector module
int PM10Value=0;         //define PM10 value of the air detector module
int Nb_sup_03=0;
int Nb_sup_05=0;
int Nb_sup_1=0;
int Nb_sup_2_5=0;
int Nb_sup_5=0;
int Nb_sup_10=0;

SoftwareSerial PMSerial(6, 7); // RX, TX

void setup()
{
  PMSerial.begin(9600);
  PMSerial.setTimeout(1500);
  Serial.begin(9600);
  lcd.begin(16, 2);
    Serial.print("Initializing SD card...");

  if (!SD.begin(4)) {
    Serial.println("initialization failed!");
    while (1);
  }
  Serial.println("initialization done.");


     myFile = SD.open("mesure.csv", FILE_WRITE);
     if (myFile) {
       myFile.print("PM 1.0");
       myFile.print(';');
       myFile.print("PM 2.5");
       myFile.print(';');
       myFile.print("PM 10");
       myFile.print(';');
       myFile.print("Nombre de particule supérieur à 0.3");
       myFile.print(';');
       myFile.print("Nombre de particule supérieur à 0.5");
       myFile.print(';');
       myFile.print("Nombre de particule supérieur à 1");
       myFile.print(';');
       myFile.print("Nombre de particule supérieur à 2.5");
       myFile.print(';');
       myFile.print("Nombre de particule supérieur à 5");
       myFile.print(';');
       myFile.println("Nombre de particule supérieur à 10");
```

```
        // close the file:
        myFile.close();
        }
       else {
     // if the file didn't open, print an error:
     Serial.println("error opening test.txt");
   }
}

void loop()
{
  if(PMSerial.find(0x42)){
    PMSerial.readBytes(buf,LENG);

    if(buf[0] == 0x4d){
      if(checkValue(buf,LENG)){
        PM01Value=transmitPM01(buf); //count PM1.0 value of the air detector module
        PM2_5Value=transmitPM2_5(buf);//count PM2.5 value of the air detector module
        PM10Value=transmitPM10(buf); //count PM10 value of the air detector module
        Nb_sup_03=transmitNb_sup_03(buf);
        Nb_sup_05=transmitNb_sup_05(buf);
        Nb_sup_1=transmitNb_sup_1(buf);
        Nb_sup_2_5=transmitNb_sup_2_5(buf);
        Nb_sup_5=transmitNb_sup_5(buf);
        Nb_sup_10=transmitNb_sup_10(buf);
      }
    }
  }

  static unsigned long OledTimer=millis();
    if (millis() - OledTimer >=3000)
    {
      OledTimer=millis();

      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("P1:");
      lcd.print(PM01Value);

      lcd.setCursor(7, 0);
      lcd.print("P2.5:");
      lcd.print(PM2_5Value);

      lcd.setCursor(0, 1);
      lcd.print("P10:");
      lcd.print(PM10Value);
      lcd.setCursor(7, 1);
      lcd.print("en ");
      lcd.print((char)228);
      lcd.print("g/m3");

      myFile = SD.open("mesure.csv", FILE_WRITE);
      if (myFile) {
        myFile.print(PM01Value);
        myFile.print(';');
        myFile.print(PM2_5Value);
        myFile.print(';');
        myFile.print(PM10Value);
        myFile.print(';');
        myFile.print(Nb_sup_03);
        myFile.print(';');
        myFile.print(Nb_sup_05);
        myFile.print(';');
```

```
        myFile.print(Nb_sup_1);
        myFile.print(';');
        myFile.print(Nb_sup_2_5);
        myFile.print(';');
        myFile.print(Nb_sup_5);
        myFile.print(';');
        myFile.println(Nb_sup_10);


        // close the file:
        myFile.close();
        }
      else {
    // if the file didn't open, print an error:
    Serial.println("error opening test.txt");
  }


      Serial.print("PM1.0: ");
      Serial.print(PM01Value);
      Serial.println("  ug/m3");

      Serial.print("PM2.5: ");
      Serial.print(PM2_5Value);
      Serial.println("  ug/m3");

      Serial.print("PM1 0: ");
      Serial.print(PM10Value);
      Serial.println("  ug/m3");
      Serial.println();
    }

}
char checkValue(unsigned char *thebuf, char leng)
{
  char receiveflag=0;
  int receiveSum=0;

  for(int i=0; i<(leng-2); i++){
  receiveSum=receiveSum+thebuf[i];
  }
  receiveSum=receiveSum + 0x42;

  if(receiveSum == ((thebuf[leng-2]<<8)+thebuf[leng-1]))  //check the serial data
  {
    receiveSum = 0;
    receiveflag = 1;
  }
  return receiveflag;
}

int transmitPM01(unsigned char *thebuf)
{
  int PM01Val;
  PM01Val=((thebuf[3]<<8) + thebuf[4]); //count PM1.0 value of the air detector module
  return PM01Val;
}

//transmit PM Value to PC
int transmitPM2_5(unsigned char *thebuf)
{
  int PM2_5Val;
  PM2_5Val=((thebuf[5]<<8) + thebuf[6]);//count PM2.5 value of the air detector module
```

```
  return PM2_5Val;
  }

//transmit PM Value to PC
int transmitPM10(unsigned char *thebuf)
{
  int PM10Val;
  PM10Val=((thebuf[7]<<8) + thebuf[8]); //count PM10 value of the air detector module
  return PM10Val;
}

int transmitNb_sup_03(unsigned char *thebuf)
{
  int Nb_sup_03;
  Nb_sup_03=((thebuf[15]<<8) + thebuf[16]); //nombre de particule superieur a 0.3
  return Nb_sup_03;
}

int transmitNb_sup_05(unsigned char *thebuf)
{
  int Nb_sup_05;
  Nb_sup_05=((thebuf[17]<<8) + thebuf[18]); //nombre de particule superieur a 0.5
  return Nb_sup_05;
}

int transmitNb_sup_1(unsigned char *thebuf)
{
  int Nb_sup_1;
  Nb_sup_1=((thebuf[19]<<8) + thebuf[20]); //nombre de particule superieur a 1
  return Nb_sup_1;
}

int transmitNb_sup_2_5(unsigned char *thebuf)
{
  int Nb_sup_2_5;
  Nb_sup_2_5=((thebuf[21]<<8) + thebuf[22]); //nombre de particule superieur a 2.5
  return Nb_sup_2_5;
}

int transmitNb_sup_5(unsigned char *thebuf)
{
  int Nb_sup_5;
  Nb_sup_5=((thebuf[23]<<8) + thebuf[24]); //nombre de particule superieur a 5
  return Nb_sup_5;
}

int transmitNb_sup_10(unsigned char *thebuf)
{
  int Nb_sup_10;
  Nb_sup_10=((thebuf[25]<<8) + thebuf[26]); //nombre de particule superieur a 10
  return Nb_sup_10;
}
```

Programme capteur de CO_2

Héberger dans :

https://github.com/lloxis/Olympiades/blob/master/CapteurCO2_SD_CSV.ino