Prototypage de systèmes à base de capteurs

Intégration d'un capteur de qualité d'air

Sommaire:

Introduction

De nos jours, la pollution et sa régulation sont des enjeux majeurs de notre société. Que ce soit dans les domaines de l'industrie, de la chimie ou de l'automobile, tout ce que l'homme crée rejette une quantité plus ou moins importante de particules fines et d'autres gaz dangereux comme monoxyde de carbone, ammoniac, dioxyde de soufre, butane, méthane… Ces gaz peuvent contribuer à la détérioration de notre environnement mais aussi de notre santé

Les effets à court terme des TVOC (composés organiques volatils totaux) sur la santé peuvent durer des heures, voire des jours. Certains des symptômes peuvent inclure un essoufflement, une réaction cutanée allergique, des étourdissements, des nausées et de la fatigue. Les symptômes peuvent inclure des maux de tête chroniques et une perte de coordination.

Dans l'optique d'un contrôle et d'une régulation de la réjection de ces gaz dans l'environnement, nous souhaitons créer un capteur de qualité d'air retournant la concentration de gaz nocifs pour la santé.

Notre carte contient un capteur Arduino et deux capteurs non Arduino. Nous pouvons ainsi grâce au programme que nous avons développé comparer les valeurs retournées par le capteur Arduino et les capteurs non Arduino qui ont une bien meilleur précision et plage de détection. Ces capteurs retournent une concentration en ppm de polluants. En fonction du besoin de l'entreprise ou de l'usager, trois niveaux peuvent être définis correspondant à des niveaux de pollution faible, moyen ou élevé. En fonction de ces niveaux une LED sera allumée sur la carte électronique. De plus, les données seront envoyées par Bluetooth à une application sur smartphone pour tenir informé l'usager.

Pour réaliser l'étalonnage, nous disposons des courbes constructeurs reliant la tension de sortie à la concentration en ppm. Nous pourrons donc réaliser différentes mesures en appliquant en entrée un volume dilué dans de l'eau d'acétone. Nous nous concentrerons donc sur le polluant acétone, mais cette démarche peut être appliquée avec n'importe quel autre polluant.

Ainsi, tout au long de ce rapport nous présenterons le capteur Arduino et les deux capteurs non Arduino que nous avons intégrés et qui permettent ainsi de quantifier la pollution de l'air. De plus, nous analyserons chaque capteur pour mettre en avant les différences de chacun.

I - Étude et explication sur les capteurs choisis

Nous avons choisi le capteur Arduino le Grove - Air Quality Sensor v1.2. C'est un capteur relativement simple qui intègre un capteur Winsen MP503, le rendant compatible avec le Grove d'Arduino. En ce qui concerne le MP503, fabriqué par Winsen, sa documentation indique qu'il adopte une technologie de fabrication à film épais multicouche, où le chauffage et le matériau semi-conducteur à oxyde métallique sur le substrat céramique subminiature Al2O3 sont extraits par des électrodes de connexion, encapsulés dans un support et un capuchon métallique.

Pour le capteur non Arduino, nous avons choisi le TGS822 et le TGS2600 fabriqué par Figaro: l'élément sensible des capteurs de gaz Figaro est un semi-conducteur en dioxyde d'étain (SnO2) qui a une faible conductivité dans l'air pur.

Les trois capteurs ont le même principe de fonctionnement. L'élément sensible présent dans chacun des capteurs, lorsqu'il est chauffé, possède un ensemble d'électrons libres qui sont attirés par l'oxygène présent dans l'air pur. Cela réduit la conductivité du matériau et entraîne une impédance électrique plus élevée. Lorsqu'une certaine concentration de gaz polluants, tels que le méthane, le CO, le propane ou l'acétone, est détectée à proximité du capteur, ces gaz réagissent avec l'oxygène absorbé par le capteur et libèrent les électrons piégés, ce qui entraîne une résistance plus faible et un meilleur flux d'électrons à travers le capteur.

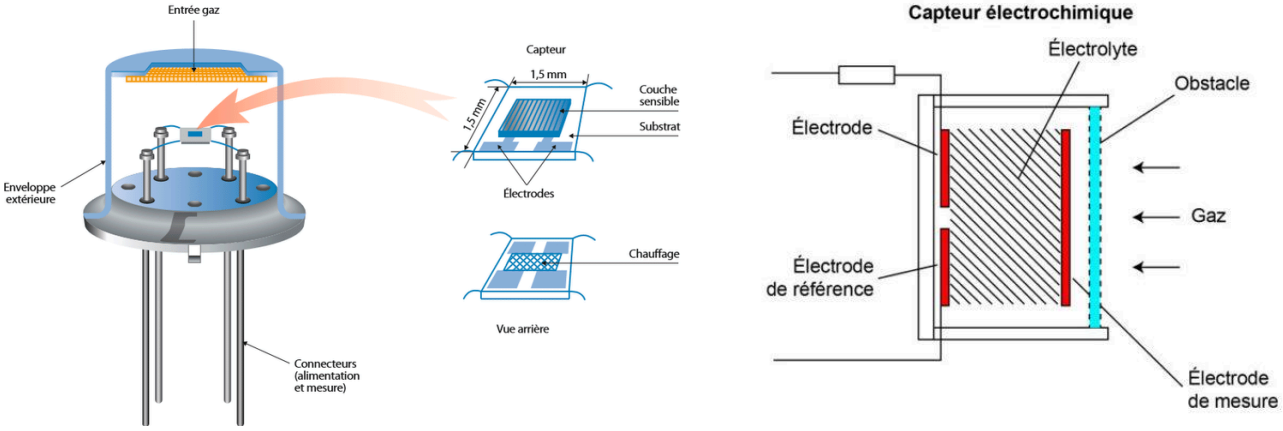


Figure 1 - Structure d'un capteur de gaz à technologie semi-conducteur

Malgré les similarités de fonctionnement, les deux capteurs non-Arduino présentent une plage de détection plus large que celle proposée par le capteur Arduino et sont assez

complémentaires au niveau des polluants détectés. Pour les 3 capteurs, le coefficient Rs/Ro représente le rapport de la résistance du capteur en milieu pollué / milieu air « pur ». De plus, ces deux capteurs sont bien plus sensibles aux variations de température affectant la mesure de Rs/R0. Finalement, la sensibilité du capteur Arduino est inférieur à 0,2 (Rs(50 ppm alcool) / Rs(air))alors que pour le TGS2600 ce rapport de sensibilité est de 0,3 - 0,6 et pour le TGS822 on a un coefficient de 0,4± 0,1.

Pour quantifier la concentration de polluants dans l'air, nous utilisons le ppm(partie par million), qui permet de savoir combien de molécules de polluant on trouve sur un million de molécules d'air (pour une concentration de 1 ppm, on a 1 gramme de soluté pour 1 000 000 de grammes de mélange).

Vous trouverez ci-dessous les courbes de sensibilité de chaque capteur.

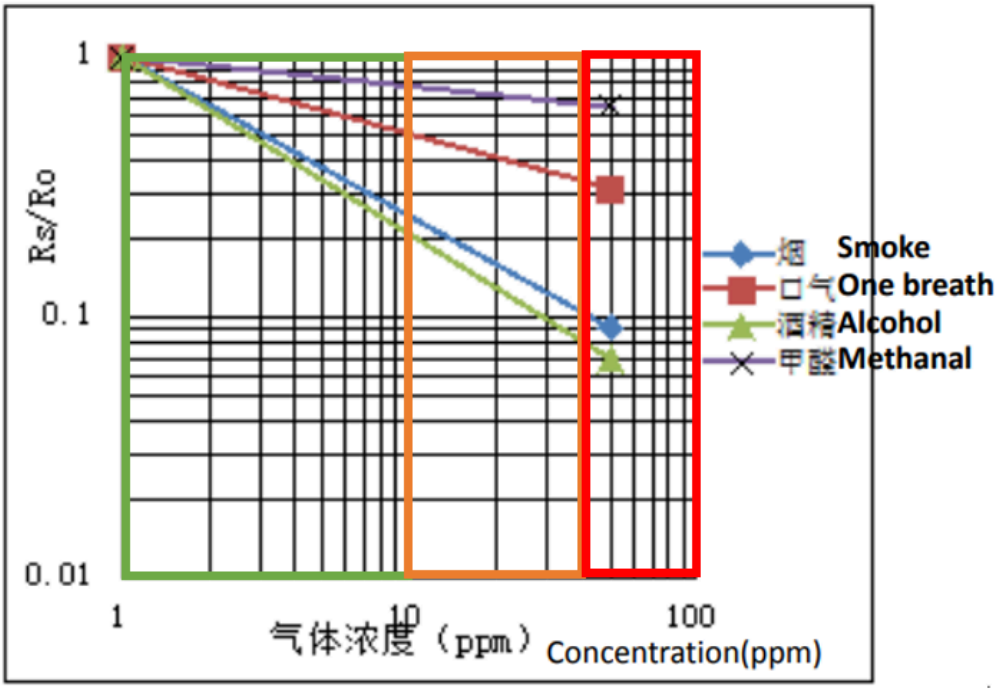


Figure 2 - Courbe de sensibilité capteur arduino MP503

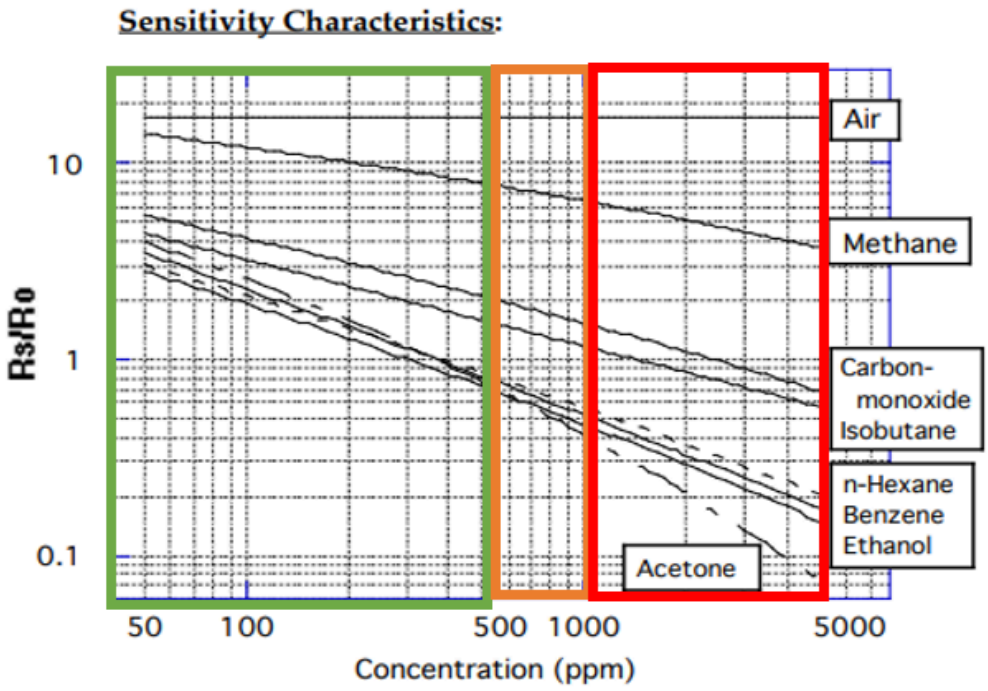


Figure 3 - Courbe de sensibilité capteur non-arduino TGS822

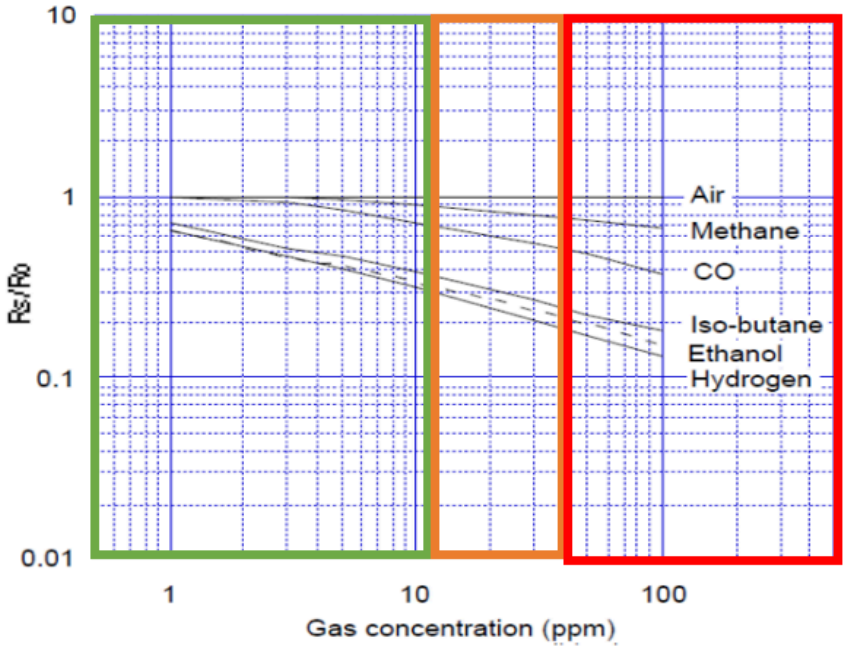


Figure 4 - Courbe de sensibilité capteur non-arduino TGS2600

Dans chacune de ces courbes, la valeur R0 correspond à la résistance Rs du capteur lorsqu'il est en contact avec 50 ppm d'alcool.

Afin de respecter certaines limites fixées, nous avons mis en place 3 zones selon la plage de mesure du capteur (concentration non nocive pour la santé, concentration peu courante et concentration nocive pour la santé). Ces bornes pourront être modifiées selon l'utilisation domestique (capteur TGS 2600) ou industrielle (TGS822)., et selon le type de polluants que l'on souhaite détecter.

En tenant compte de ces courbes, un circuit de mesure simple peut être utilisé, dans lequel l'un des capteurs fonctionne comme une résistance variable. Ce circuit génère une faible tension proportionnelle à la concentration des gaz polluants présents. Le schéma du circuit est présenté à la figure 5.

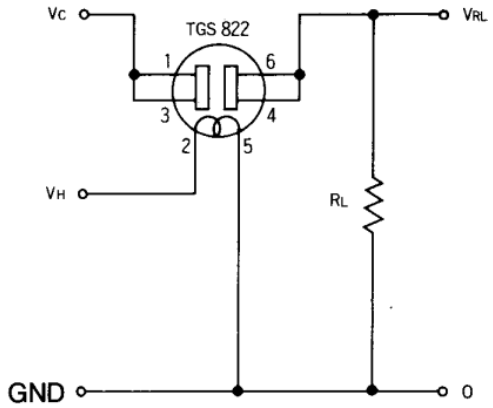


Figure 5 - Circuit de mesure de base qui utilise le TGS822 comme exemple

Dans la figure 5, Vc, Vh et RL représentent des valeurs fixes correspondant respectivement à la tension de source, la tension du chauffage et la résistance de charge. Vrl est la tension de sortie qui transmet les informations sur la concentration de polluants. On a ainsi la formule présentée à la figure 6, qui établit une relation entre la résistance variable du capteur Rs (résistance de chauffage) et la tension de sortie du circuit Vrl.

$$Rs = \left(\frac{Vc}{VRL} - 1\right) \times RL$$

Figure 6 - Équation descriptive du circuit de mesure

Avec cette formule et les courbes présentées aux figures 2, 3 et 4, si le gaz polluant analysé est déjà connu, il est possible de calculer directement sa concentration en fonction de la sortie du capteur d'air pour chacun des capteurs.

On peut aussi comparer les capteurs en différents paramètres techniques dans le tableau ci-dessous:

	Capteur Arduino (MP503)	**Capteurs Non-Arduino (TGS822 et TGS2600)**
Temps de Réponse	1 - 1.5 min	40 - 50 sec
Sensibilité Croisée	Élevée (CO_2, CO, COV, éthanol, acétone, H_2)	Faible à modérée (éthanol, CO, CH4)
Sensibilité (change of ratio)	0.6	0.3 - 0.4

- Effet de la Température et de l'humidité sur la performance des Capteurs:

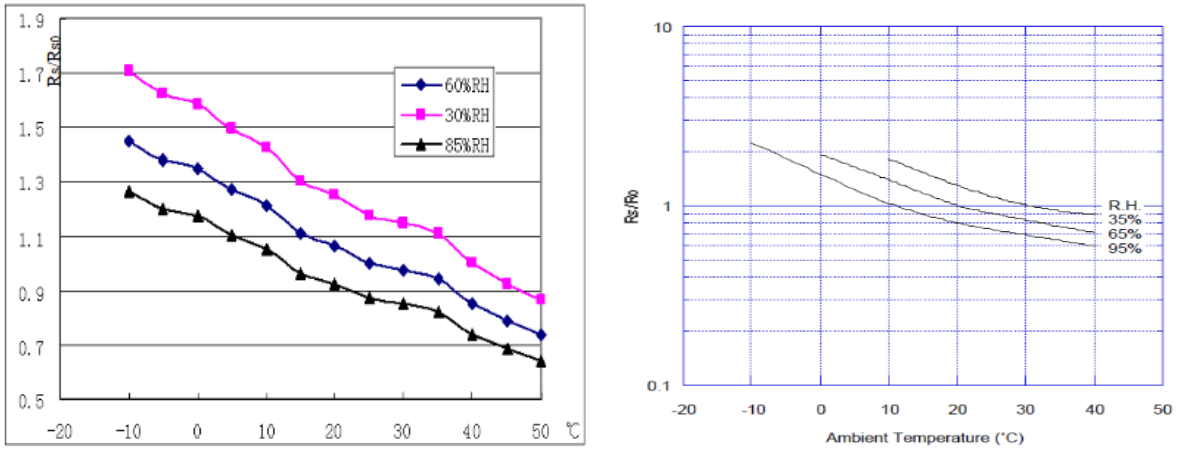


Figure 7 - Comparaison de la sensibilité du Capteur Arduino (à gauche) et les capteurs TGS (à droite)

II - Intégration des capteurs

L'objectif principal de l'intégration des trois capteurs est de permettre à l'utilisateur de choisir un capteur parmi les trois disponibles (capteur Arduino, TGS822 et TGS2600) pour analyser l'air ambiant à la recherche de substances chimiques toxiques, et de lui transmettre les résultats. Les messages de sortie du circuit ainsi que le choix du capteur effectué par l'utilisateur seront transmis via une connexion Bluetooth vers tout appareil capable d'accéder à un terminal Bluetooth.

Une fois le capteur approprié sélectionné, le circuit détectera la concentration de différents gaz polluants et, non seulement, enverra le niveau de concentration à l'utilisateur, mais également allumera trois LED de couleurs différentes (verte, jaune et rouge) pour indiquer respectivement la présence de polluants en faible, moyenne ou forte concentration.

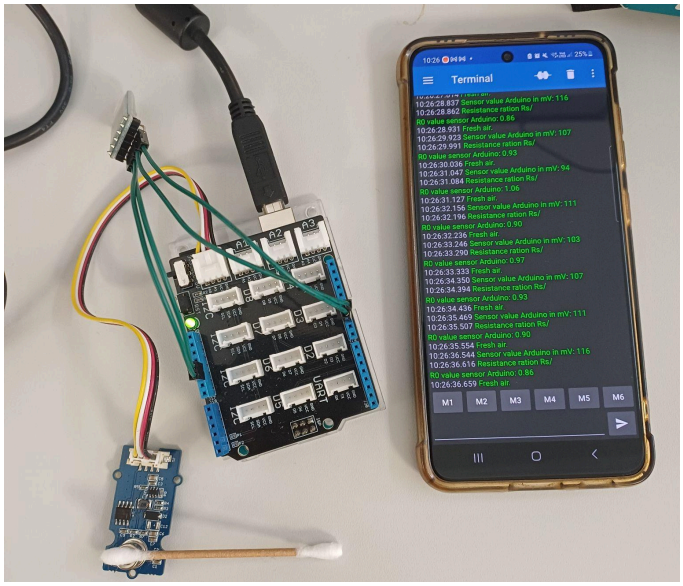


Figure 8 - Circuit Fonctionnel avec le capteur arduino et le grove base shield

Le code Arduino pour l'application du circuit et l'intégration des capteurs se trouve en **Annexe A**.

Dans le code, les trois capteurs sont attribués aux entrées de signal analogique de l'Arduino, et chaque capteur possède sa propre plage de limites, définissant les concentrations élevées, moyennes et faibles de gaz en fonction de la tension reçue en entrée. Une fois cette analyse effectuée, l'Arduino allume la LED correspondante et envoie les valeurs de sortie au terminal Bluetooth de l'utilisateur.

Le circuit mis en œuvre est une version adaptée du circuit de mesure de base présenté dans la Section I, où le TGS822 et le TGS2600 disposent chacun de versions distinctes mais identiques, tandis que le capteur Arduino peut être directement connecté à la carte Arduino comme d'habitude.

Les composants utilisés ont été soigneusement sélectionnés pour être compatibles en taille avec la carte Arduino UNO à notre disposition, en veillant à ce que les trous et les pastilles soient correctement dimensionnés afin de garantir un fonctionnement efficace du circuit. Le schéma électrique du circuit et le routage du PCB ont été réalisés à l'aide du logiciel KiCad et sont disponibles en **Annexe B**.

Nous avons décidé de placer un AOP en mode suiveur car il permet de ne pas tirer de courant sur l'entrée, tout en délivrant beaucoup de courant en sortie. Cela permet donc d'exploiter le signal fourni par le capteur sans le perturber (car le signal du capteur est assez instable).

Pour calibrer le TGS822, nous avons dû sélectionner un gaz détectable dans sa courbe de sensibilité présentant une variation de résistance très prononcée avec l'augmentation de la concentration, ainsi qu'une courbe distincte par rapport à d'autres gaz possibles. Pour ces raisons, nous avons choisi l'acétone.

Une fois le gaz sélectionné, le processus devient relativement simple : préparer des solutions de 100, 500 et 1000 ppm, exposer le capteur à ces solutions tout en notant les valeurs de sortie enregistrées, puis ajuster le capteur en fonction de la courbe de sensibilité indiquée dans la fiche technique. Enfin, la calibration du capteur Arduino et du TGS2600 est réalisée en comparant leurs valeurs avec celles calibrées pour le TGS822.

Les résultats mesurés peuvent être observés dans le graphique suivant en comparaison avec les valeurs attendues issues de la fiche technique :

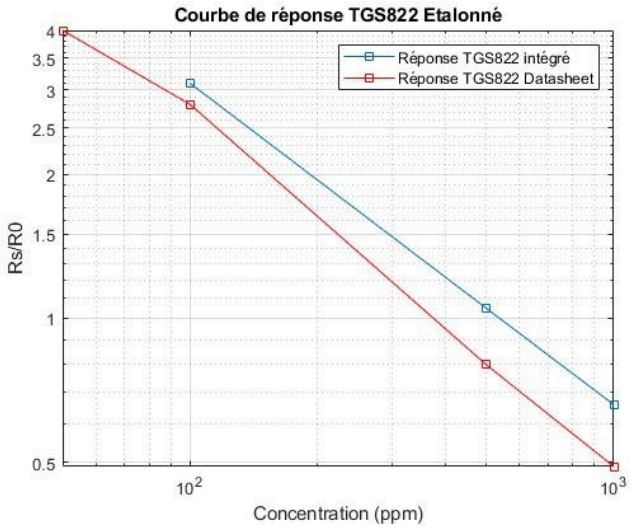
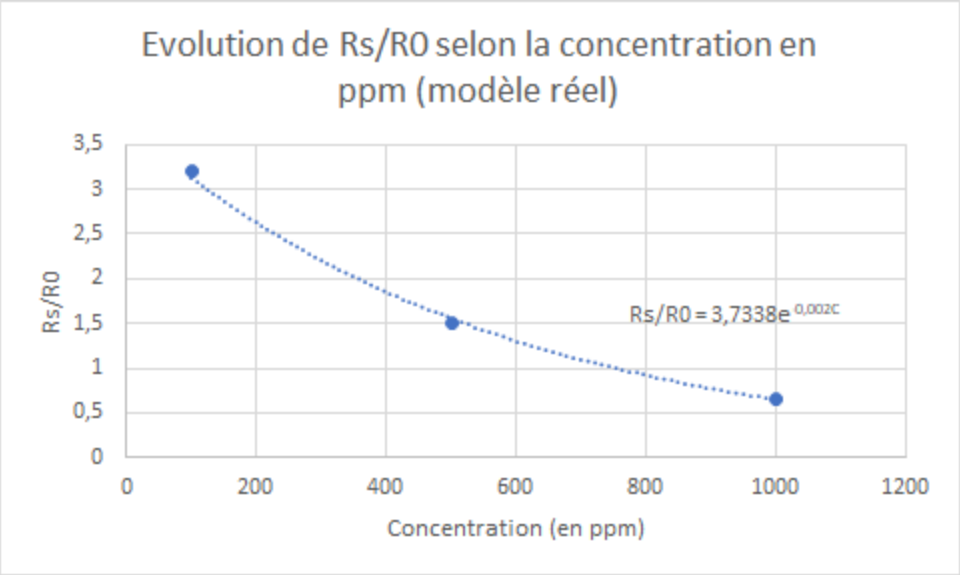


Figure 9 - Comparaison Réponse TGS822 Mesuré et Théorique

Nous pouvons constater que la réponse du capteur à l'augmentation de la concentration de gaz n'est pas linéaire. Ainsi, pour effectuer l'étalonnage, nous avons saisi les données mesurées dans Excel afin de réaliser un ajustement de courbe et de caractériser la réponse comme exponentielle (il a fallu prendre en compte la différence de ratio entre la courbe constructeur et la courbe que nous avons tracée). Enfin, la correction a été implémentée dans le code Arduino sous forme d'une conversion des données analogiques d'entrée afin d'obtenir la sortie correcte.



On a donc pu trouver une équation reliant la valeur de ratio Rs/R0 pour notre capteur, permettant ainsi de donner à l'utilisateur une valeur assez fiable de concentration en ppm d'acétone. On obtient C(en ppm) = ln(Rs/(R0*3,7338) / 0,002

III - Améliorations possibles

Bien que le circuit conçu pour ce projet fonctionne bien, certaines améliorations pourraient être envisagées lors de son passage à une utilisation réelle. Parmi celles-ci, on peut citer:

Réception de données de plusieurs capteurs différents simultanément: Chaque capteur devrait avoir ses propres courbes de sensibilité mais un même intervalle de mesure, permettant de croiser les valeurs de sortie. Cela faciliterait l'identification précise du gaz polluant détecté et la détermination de sa concentration.

Développement d'une application mobile améliorée: Cette application permettrait d'indiquer le polluant que le capteur est censé détecter et d'offrir une interface plus claire et intuitive pour l'utilisateur moyen.

Une autre amélioration serait de calibrer tous les capteurs pour tous les polluants et de proposer un modèle liant la concentration et le ratio Rs/R0, commun à tous les polluants, permettant de donner à tout instant une valeur de concentration à l'utilisateur.

Conclusion

Pour conclure, ce projet sur les capteurs nous a apporté de nombreux enseignements. Il nous a permis de comprendre en profondeur les différences entre divers types de capteurs et l'importance de sélectionner le capteur adapté à une application spécifique. Nous avons également eu l'occasion d'explorer l'intégration pratique des capteurs, ce qui nous a permis de mettre en œuvre concrètement les concepts théoriques étudiés en cours.

De plus, ce projet nous a offert une expérience pratique précieuse dans la conception de circuits imprimés (PCB) avec le logiciel Kicad et dans le soudage de composants CMS. Il a également contribué à développer des compétences en gestion de projet et de temps, tout en renforçant notre capacité à travailler en équipe et à répartir efficacement les tâches entre les membres.

Bibliographie

Grove - Air Quality Sensor v1.3 | Seeed Studio Wiki

Figaro Capteur de gaz TGS-822 Pour type de gaz: monoxyde de carbone, ammoniac, dioxyde de soufre, alcool, essence (Ø x H - Conrad Electronic France

https://www.conrad.fr/fr/p/figaro-capteur-de-gaz-tgs-2600-o-x-h-9-2-mm-x-7-8-mm-183304.html#productDownloads

TGS822 footprint & symbol by Figaro | SnapMagic Search

https://www.snapeda.com/parts/TGS2600/Figaro/view-part/?ref=search&t=TGS2600

LA QUALITÉ DE L'AIR INTÉRIEUR

Calibrating Sensors

Annexe A : Codes Arduino

- Code C pour le module température et bluetooth :

```
// Déclaration des librairies

#include <TM1637Display.h>
#include <DHT.h>
#include <SoftwareSerial.h>


// Déclaration des entrées / sorties

//Capteur
#define DHTPIN 2 // la borne data du capteur est branchée sur la broche 2
#define DHTTYPE DHT11 //#define DHTTYPE DHT22 (pour un capteur DHT22)
DHT dht(DHTPIN, DHTTYPE); // on indique la broche et le type de capteur

//Bouton
# define button 4

//afficheur 7 segments
#define CLK 7
#define DIO 8
TM1637Display display(CLK,DIO);
const uint8_t SEG_DONE[] = {
  SEG_B | SEG_C | SEG_D | SEG_E | SEG_G,
  SEG_A | SEG_B | SEG_C | SEG_D | SEG_E | SEG_F,
  SEG_C | SEG_E | SEG_G,
  SEG_A | SEG_D | SEG_E | SEG_F | SEG_G
  };

//Bluetooth
#define rxPin 11 // Broche 11 en tant que RX, à raccorder sur TX du HC-05
#define txPin 10 // Broche 10 en tant que TX, à raccorder sur RX du HC-05
SoftwareSerial mySerial(rxPin, txPin);
```

```
// datas pour le programme loop
float data ;
int changement=0;

void setup() {
  pinMode(button,INPUT);
  display.setBrightness(4); // luminosité de 0 à 7
  // define pin modes for tx, rx pins:
  pinMode(rxPin, INPUT);
  pinMode(txPin, OUTPUT);
  mySerial.begin(9600);
  Serial.begin(9600);
  dht.begin(); // initialisation du capteur
}

void loop()
{
  int appui= digitalRead(button);

  //si appui bouton alors changment mode température/humidité
  if (appui==1){
    if (changement == 0){
      changement = 1;
    }
    else if ( changement == 1){
      changement=0;
    }
  }

  if ( changement == 0){
    display.clear();
    data = dht.readHumidity(); // lecture de la valeur de l'humidité
    display.showNumberDec(data, true, 2, 0);
    delay(500);
    // en Bluetooth de la valeur d'humidité
    mySerial.print("Humidity = ");
```

```
    mySerial.print(data);
    mySerial.print("\n");
  }

  else if (changement == 1){
    display.clear();
    data = dht.readTemperature(); // lecture de la valeur de la
température
    display.showNumberDec(data, true, 2, 0);
    delay(500);
    mySerial.print("Temperature = ");
    mySerial.print(data);
    mySerial.print("\n");
  }

  delay(50);

}
```

- code C pour notre système de qualité d'air :

```
/*
    Grove_Air_Quality_Sensor.ino
*/

#include "Air_Quality_Sensor.h"
#include <SoftwareSerial.h>

//Bluetooth
#define rxPin 11 // Broche 11 en tant que RX, à raccorder sur TX du HC-05
#define txPin 13 // Broche 10 en tant que TX, à raccorder sur RX du HC-05
SoftwareSerial mySerial =  SoftwareSerial(rxPin, txPin); // Bluetooth

//LED couleur
#define rouge 5
#define jaune 7
```

```
#define vert 9

AirQualitySensor sensor(A0);//capteur Arduino
#define  sensor_non_arduino_home A1//capteur TGS2600
#define sensor_non_arduino_industry A2//capteur TGS822

/*Variable utilisées pour les calculs de qualité d'air*/
int Vrl0; //tension en mV dans l'air pur
float R0; //resistance du capteur dans air pur
float Rs; // resistance lors de la mesure de la quélité d'air
float ratio; // raport Rs/R0
float concentration;
int analog_value; // valeur analogique entre 0 et 1024 (10bits) retournée
par le capteur
int mV_value; // valeur convertit en mV retournée par le capteur


//choice of the non Arduino sensor to use
String choix_sensor = "arduino";

void setup(void) {
  pinMode(rxPin, INPUT);
  pinMode(txPin, OUTPUT);

  pinMode(rouge, OUTPUT);
  pinMode(jaune, OUTPUT);
  pinMode(vert, OUTPUT);

  pinMode(sensor_non_arduino_home, INPUT);
  pinMode(sensor_non_arduino_industry, INPUT);

  mySerial.begin(9600);

  while ((!mySerial)and(mySerial.available() <= 0)) ;
  mySerial.println("Waiting sensor to init and choice sensor...");
  delay(3000);
```

```
  if (sensor.init()) {
    mySerial.println("Sensor ready.");
  } else {
    mySerial.println("Sensor ERROR!");
  }
}

void loop(void) {

  if (mySerial.available() > 0) {
    choix_sensor = mySerial.readStringUntil('\n');//on lit tous les
caractères envoyés jusqu'au caractère sentinelle
    choix_sensor.trim();//'netoie' la chaine de caractères des caractères
parasites
  }

  delay(500);

  if (choix_sensor == "arduino") { /*Arduino Sensor*/

    Vrl0 = 140.0;  //reference value in fresh air in mV for arduino groove
sensor

    int quality = sensor.slope();
    analog_value = sensor.getValue();
    mV_value = (float) map(analog_value, 0, 1024, 150, 4400);

    R0 = ((5000.0 / Vrl0) - 1.0) * 10000.0;
    Rs = (((5000.0 / mV_value) - 1.0) * 10000.0);
    ratio = (Rs / R0);

    mySerial.println("arduino");
    mySerial.println("Sensor value in mV: ");
    mySerial.print(mV_value);
    mySerial.println("");
    mySerial.print("Value Rs on R0: ");
    mySerial.print(ratio);
```

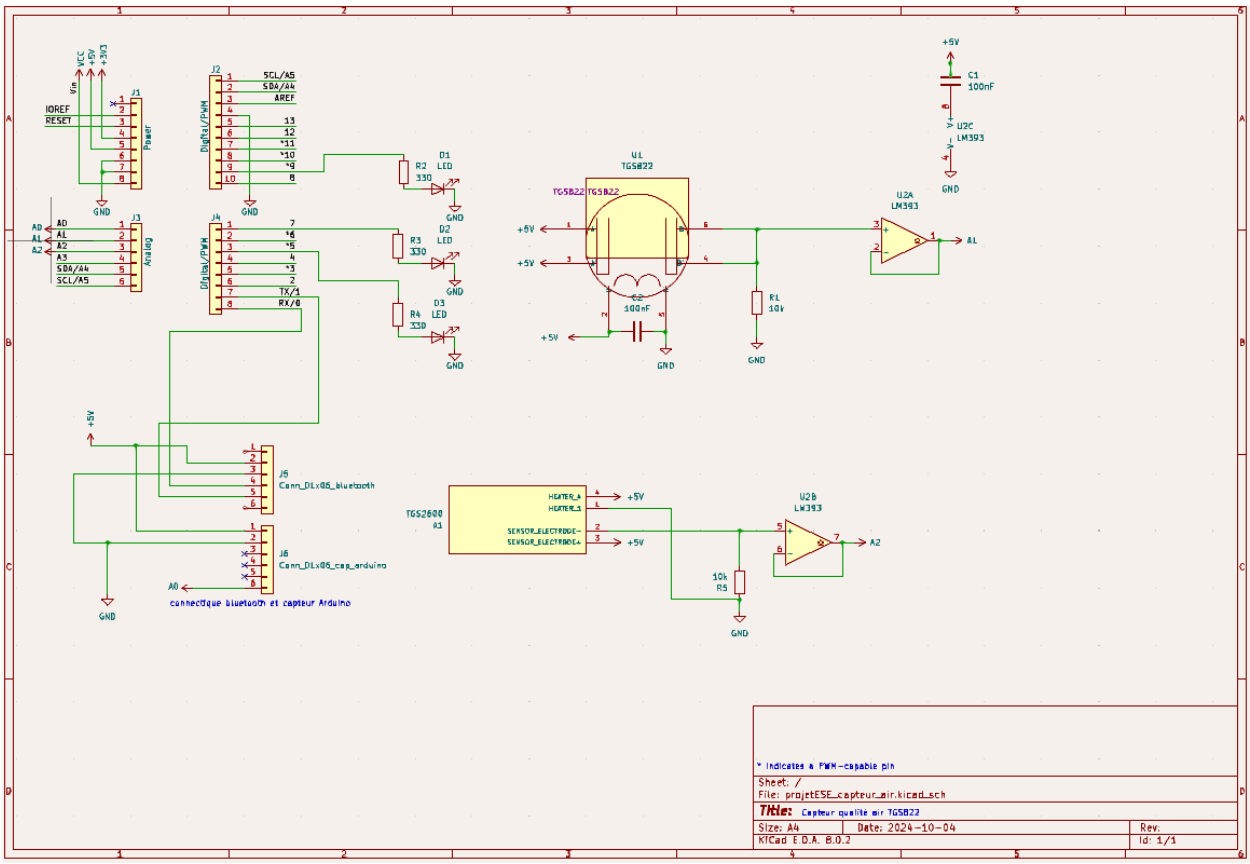
```
    mySerial.println("");


    //The arduino gaz sensor module provides a method that send a message
wether the pollution is high or low
    if (quality == AirQualitySensor::HIGH_POLLUTION) {
      mySerial.println("High pollution!");
      digitalWrite(vert, LOW);
      digitalWrite(jaune, LOW);
      digitalWrite(rouge, HIGH);
    } else if (quality == AirQualitySensor::LOW_POLLUTION) {
      mySerial.println("Low pollution!");
      digitalWrite(vert, LOW);
      digitalWrite(jaune, HIGH);
      digitalWrite(rouge, LOW);
    } else if (quality == AirQualitySensor::FRESH_AIR) {
      mySerial.println("Fresh air.");
      digitalWrite(vert, HIGH);
      digitalWrite(jaune, LOW);
      digitalWrite(rouge, LOW);
    }

  }

  else if (choix_sensor == "TGS822") { /* Non -arduino sensor for home*/

    R0 = 37654.3;  //reference value in fresh air in mV for TGS822
calculated with Rs,air/R0=10.8

    analog_value = analogRead(sensor_non_arduino_industry);
    mV_value = (float) map(analog_value, 0, 1024, 70, 4000); //Value in mV

    Rs = (((5000.0 / mV_value) - 1.0) * 10000.0);
    ratio = (Rs / R0);

    concentration = log(ratio/3.7338) / 0.002; // modèle déduit lors de la
calibration
```

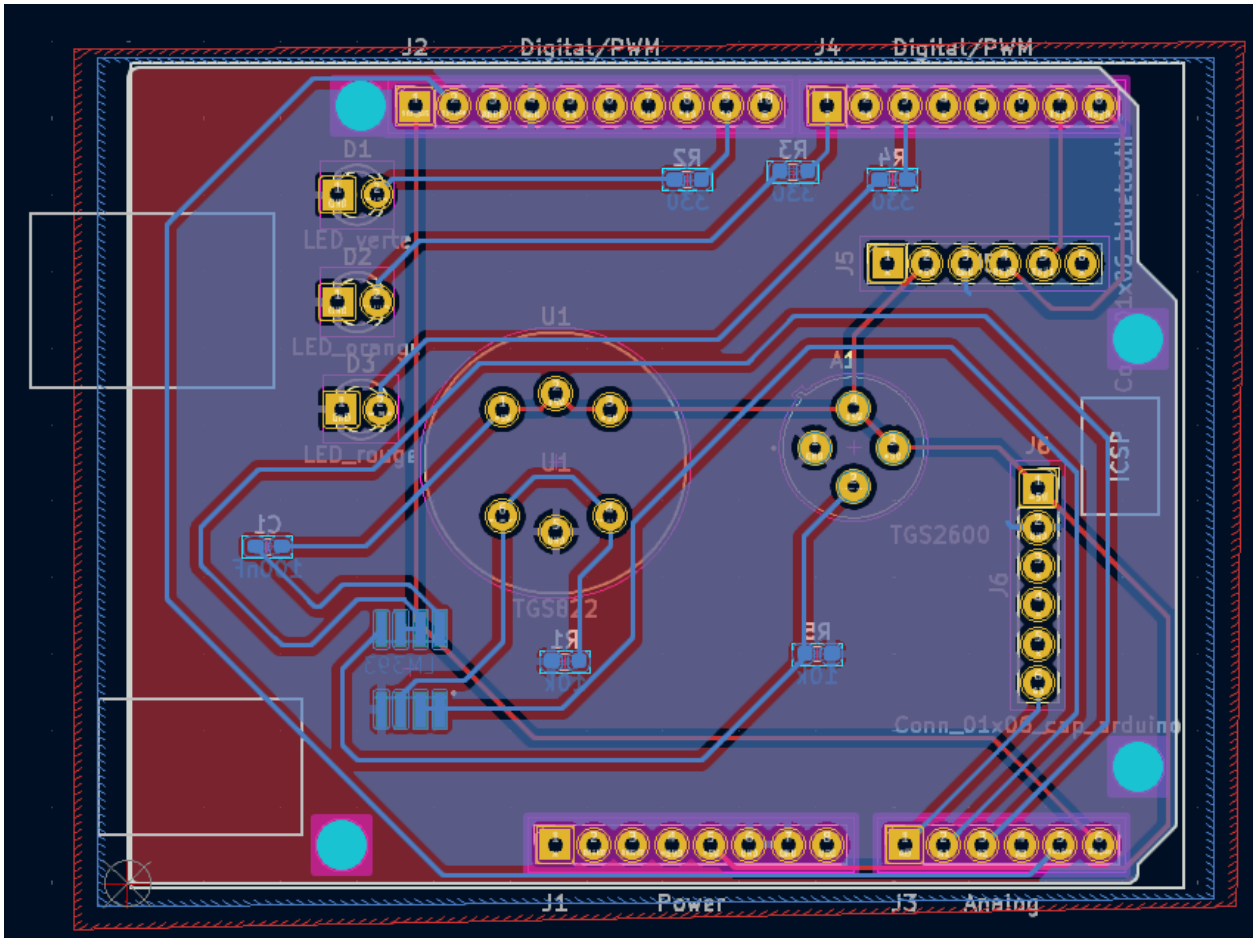
```
    mySerial.println("TGS822");
    mySerial.println("Sensor value in mV: ");
    mySerial.print(mV_value);
    mySerial.println("");
    mySerial.print("Value Rs on R0: ");
    mySerial.print(ratio);
    mySerial.println("");
    mySerial.print("Concentration (ppm) : ");
    mySerial.print(concentration);
    mySerial.println("");

    if (ratio > 2) {
      digitalWrite(vert, HIGH);
      digitalWrite(jaune, LOW);
      digitalWrite(rouge, LOW);
    }
    else if (ratio<2 and ratio>0.7) {
      digitalWrite(jaune, HIGH);
      digitalWrite(vert, LOW);
      digitalWrite(rouge, LOW);
    }
    else if (ratio < 0.7) {
      digitalWrite(rouge, HIGH);
      digitalWrite(jaune, LOW);
      digitalWrite(vert, LOW);
    }

  }


  else if  (choix_sensor == "TGS2600") { /* Non -arduino sensor for
industries*/

    Vrl0 = 100.0;  //reference value in fresh air in mV for TS2600
```

```
    analog_value = analogRead(sensor_non_arduino_home);
    mV_value = (float) map(analog_value, 0, 1024, 100, 5000); //Value in
mV

    R0 = ((5000.0 / Vrl0) - 1.0) * 10000.0;
    Rs = (((5000.0 / mV_value) - 1.0) * 10000.0);
    ratio = (Rs / R0);


    mySerial.println("TGS2600");
    mySerial.println("Sensor value in mV: ");
    mySerial.print(mV_value);
    mySerial.println("");
    mySerial.print("Value Rs on R0: ");
    mySerial.print(ratio);
    mySerial.println("");

    if (ratio > 0.4) {
      digitalWrite(vert, HIGH);
      digitalWrite(jaune, LOW);
      digitalWrite(rouge, LOW);
    }
    else if (ratio<0.4 and ratio>0.1) {
      digitalWrite(jaune, HIGH);
      digitalWrite(vert, LOW);
      digitalWrite(rouge, LOW);
    }
    else if (ratio < 0.1) {
      digitalWrite(rouge, HIGH);
      digitalWrite(jaune, LOW);
      digitalWrite(vert, LOW);
    }
  }

  else {
    mySerial.println("Waiting for sensor choice...");
  }
```

```
  delay(1000);
}
```

Annexe B : Schéma Kicad




J2
Digital/PWM
J4
Digital/PWM
D1
LED_verte
D2
LED_orange
D3
LED_rouge
U1
A1
J5
J6
ICSP
TGS2600
TGS822
J1
Power
J3
Analog