Treball fi de Grau de Tecnologies de Telecomunicació

Disseny e implementació d'un sistema de calibració de sensors per Smart Citizen Kit

---

Autor: Lluís Anguera
Supervisor: Pere Tuset

12 Juny de 2015

Agraïments

Vull agrair al consultor d'aquest treball Pere Tuset per donar ànims i aconsellar en moments difícils d'aquest projecte.
També és just donar les gràcies a Alex Posada i Miquel Heras, responsables de hardware d'Smart Citizen que m'han ajudat amb els dubtes que anaven sorgint i amb els que ha sigut fàcil treballar.
Finalment vull dedicar aquest projecte a totes les persones que han estat al meu costat tots aquests anys i m'han donat suport quan l'he necessitat. En especial els meus pares Rafel i Antonia i el meu amic Zigor.

Resum

Aquest document descriu el Projecte Final de Grau de Tecnologies de Telecomunicacions de la Universitat Oberta de Catalunya (UOC), i es centra en l'àrea *d'Arduino.* Concretament, l'objectiu és desenvolupar i programar un algoritme compatible amb les plaques oferides per l'empresa col·laboradora *Smart Citizen*. Aquestes plaques contenen sensors ambientals per a mesurar la concentració de gasos NO2 i CO en l'atmosfera. Els sensors són de baix cost i són molt sensibles a la variabilitat de temperatura, humitat i a la seva pròpia degradació. Aquests canvis produeixen errors en les mesures, que es volen pal·liar a través d'una calibració que corregeixi aquestes diferències.

El primer pas serà oferir documentació sobre les característiques més importants de la placa de treball *Smart Citizen Kit (*SCK). A continuació es mostrarà una visió general sobre diferents mètodes de calibració amb sensors similars i es faran diverses proves. Finalment, es faran gràfiques a nivell matemàtic per veure quin és l'algoritme més òptim. Un cop sigui escollit crearem un codi compatible que quan s'introdueixin els valors de la placa que es vol calibrar, els compari amb els valors de referència i apliqui les correccions apropiades.

Paraules clau

Aquí mostrem les paraules clau que defineixen aquest projecte:

- Sensors ambientals.
- Arduino.
- Dades compartides.
- Smart Citizen
- Unitat de calibració i testeig.

Abstract

This document describes the Final Project of Telecommunications Technologies degree at Universitat Oberta de Catalunya (UOC) in the Arduino area. Specifically the objective is develop and program an algorithm for compatible Arduino boards offered by SmartCitizen company. These boards contains ambient sensors to mesure the concentration of CO and NO2 in the air. These sensors are low cost and very sensitive to temperature and humidity changes. We want to correct these changes with a good calibration. The first step of this project is show all information of our board SmartCitizen Kit (SCK). The second step is the study of different measuremnt methods with similar sensors. Finally we will test all options and we will create a lot of graphs to compare with references. With the best choice we will program a compatible code to correct read errors in the sensors.

Keywords

We show the different keywords of this project:

- *Environmental sensors.*
- *Arduino.*
- *Shared data.*
- *Smart Citizen*
- *Unity of testing and calibration.*

Índex de continguts

Llista de figures

1. Introducció

Abans de començar amb el desenvolupament d'aquest projecte es van dedicar dos mesos per estudiar i entendre les necessitats d'aquest treball. Una part molt important va ser la recollida d'informació complementària sobre calibració de sensors i sobre el comportament dels gasos. És un camp nou i amb molta manca de dades fiables. Per aquest motiu aquest estudi és una part essencial per a introduir a qualsevol usuari en aquest món i fer entendre el desenvolupament posterior.

1.1 Justificació

El projecte es basa en sensors ambientals que controlen les concentracions de gasos. Ajudar a tenir una millor consciència col·lectiva sobre l'aire que respirem és una molt bona motivació. Els sensors normalment resulten cars per a la majoria d'usuaris, aquests són de baix cost i si s'aconsegueix una bona calibració motivaria a molta gent a la seva compra i ús. Aquest fet impulsaria a investigar més i millorar les plaques i sensors existents. També l'interès que desperta treballar amb *hardware/software* lliure, en aquest cas amb la placa SCK basada en *Arduino.* A més, ser conscient de la importància que estan adquirint en l'actualitat els sensors, sobretot en el desenvolupament de les *smart cities* impulsa a voler adquirir nous coneixements i competències en aquesta matèria, que poder ser exportables a diversos camps.

1.2 Objectius principals

El projecte te dos parts diferenciades, la primera part implica:

- Estudiar i analitzar la documentació actual oferida en els repositoris d'Smart Citizen.
- Fer una recerca d'informació complementària per ampliar els nostres coneixements i trobar metodologies útils en el projecte.

La segona part es basa en dur a terme el desenvolupament:

- Desenvolupar un algoritme propi que pugui tornar tots els punts de calibració establerts d'una placa amb només dues dades.
- Aplicar aquest algoritme en un programa basat en la nostra plataforma de treball d'*Arduino.*
- Modificar la llibreria oficial SCK afegint el nostre codi al seus repositoris.
- Finalitzar el projecte i obtenir unes conclusions.

1.3 Beneficis

Aquest projecte ofereix diversos beneficis per als futurs usuaris de la SCK, els principals són:

- Adquisició de diverses dades ambientals a un baix cost.
- Compartir les dades recollides amb altres usuaris per obtenir mesures més precises i per ampliar la base de dades existent.
- Avaluar la qualitat de l'aire que respirem segons diversos indicadors, per exemple la concentració de gasos nocius. Aquesta avaluació es pot dur a terme tant dins com fora de casa i evitar ambients que puguin posar en risc la nostra salut.

1.4 Recursos bibliogràfics

A continuació es mostren cinc recursos que ens ajudaran a comprendre el concepte d'*Smart City* dels sensors ambientals i de tot allò que guarda relació amb el projecte.

1.-

• **Títol:** Sensor Networks for Ambient Intelligence

• **Autor/s:** Pauwels, E.J. and Salah, Albert A. And Tavenar,R.

• **Data:** 13 Octubre 2016.

• **Font d'informació:** http://ieeexplore.ieee.org/xpl/abstractReferences.jsp ?tp=&arnumber=4412806&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D4412806

• **Resum:**

Degut a la velocitat dels avenços tecnològics en la creació de xarxes i tecnologia basada en sensors, cada vegada més creix l'interès per crear xarxes amb sensors interconnectats entre ells i amb altres dispositius computacionals, que són capaços de processar senyals i analitzar dades en multimodal. Aquests treball té dues parts ben diferenciades, en la primera part es dóna una visió general de les novetats en l'àmbit
de les anomenades xarxes de sensors multimodals, concretant en les aplicacions d'intel·ligència ambiental.
En la segona part, es parla sobre com obtenir els patrons temporals emprant les dades obtingudes als sensors. Aquesta correlacions de senyals creuades permeten estudiar l'evolució d'algunes magnituds a través de l'espai i el temps.
Per exemple la sortida de gas a través d'una ciutat en diferents moments del dia.

• **Aspectes rellevants:**

Aquest article té informació rellevant pel nostre projecte, com per exemple la metodologia per obtenir dades concretes de diversos sensors. També es parla de la placa multimodal SCK, aquesta placa conté sensors de soroll, intensitat de la llum, de gasos CO2 i NO2, temperatura....
Saber quin és el millor patró de disseny tant per hardware com per software en les plaques de sensors ens pot ajudar a evitar errors.

El nostre projecte té el hardware dissenyat, però aquest article ens pot ajudar a detectar taules defectuoses que donen valors atípics en el software.

2.-

• **Títol:** An Incremental Approach to Unit Testing During Maintenance

• **Autor/s:** Harrold, MJ. And Souffa, M.L

• **Data:** Octubre de 1988

• **Font d'informació:** Proceedings of the Conference on Software Maintenance, 1988. IEEE.

• **Resum:**

En aquest article es descriu la metodologia que s'utilitzarà en paral·lel de la prova de testeig durant el manteniment del nostre projecte, les proves de testeig són integrades en un entorn de programació. És a dir, les proves no es realitzen només al final, sinó al llarg del projecte. Cada vegada que modifiquem alguna cosa o afegim una nova funció es crea un nou cas de prova al nostre marc de programació. Quan realitzem una modificació important, com una versió nova del programa, una persona haurà de fer un testeig per decidir quines parts resulten modificades o quines eliminades.

• **Aspectes rellevants:**

Un aspecte rellevant d'aquest article, és mostrar com realitzar un testeig continu en el nostre projecte. Cada vegada que hi hagi un canvi, el programa haurà de fer una prova de validesa. Aquest és un dels requisits mínims que es demanen en qualsevol software de components.

La metodologia de desenvolupament que es seguirà en aquest projecte, implica la creació de casos de proves per a qualsevol funcionalitat del sistema, la aplicació del codi per a la funcionalitat concreta i posteriorment si es passen les proves, la comissió del canvi en el repositori de la llibreria oficial.

3.-

• **Títol:** Danish Smart Cities: Sustainable Living in a Urban World

• **Autor/s:** Jonas Mortensen, Frederik Jonsbak Rohde, Klaus Rovsing

• **Data:** 19 Novembre de 2012

• **Font d'informació:** http://www.theinternetofthings.eu/sites/default/files/Danish%20Smart%20Cities%20-%20Sustainable%20living%20in%20an%20urban%20world.pdf

• **Resum:**

Aquest projecte parla sobre el concepte d'smart city i la creixent importància que està adquirint en les ciutats més importants. Cada setmana milions de persones es mouen de zones rurals a zones urbanes amb la intenció de trobar oportunitats laborals, aquest increment té molts efectes, però el més relacionat amb el nostre projecte és l'augment de gasos nocius per la salut.

És lògic per tant, que les ciutats més importants estiguin invertint molt de temps i diners en millorar les tecnologies basades en sensors (ambientals, de velocitat, de percepció...) per tenir una ciutat més eficient i neta.

• **Aspectes rellevants:**

Aquest article és rellevant pel nostre projecte ja que ens defineix objectivament que és una *smart city* i quins beneficis pot aportar als seus habitants. Tot i que aquest document no té un llenguatge tècnic, és útil per donar sentit a la importància de posar diverses plaques distribuïdes per la ciutat recollint dades. Dóna una visió diferent, que inspira als habitants a recollir dades ambientals per ells mateixos en diferents punts del país, d'aquesta manera tindríem dades fiables i fins i tot es podria utilitzar la informació recollida per pressionar al govern a millorar la qualitat de les nostres ciutats.

4.-

• **Títol:** An Introduction to I2C abd SPI protocols

• **Autor/s:** Leens, F.

• **Data:** Febrer del 2009

• **Font d'informació:** Instrumentation Measurament Magazine, IEEE

• **Resum:**

Aquest article parla sobre els dos protocols estàndard en la comunicació entre components electrònics. Concretament entre els circuits integrats (I2C) i el perifèrics amb interfície serial (SPI). Aquests dos protocols són els més utilitzats per la comunicació entre plaques.

El protocol I2C, és un protocol simple dissenyat per comunicar circuits integrats en una única placa. A dia d'avui aquest protocol pot treballar a una velocitat de fins a 3,4Mbps. El seu èxit radica en la seva simplicitat, ja que només requereix de dos línies de bus, és flexible i funciona amb una relació de *master/slave* entre tots els components.

L'altre protocol, l'*SPI*, pot transferir dades més ràpidament que I2C, concretament al doble de velocitat ja que els components es comuniquen entre si alhora. També segueix una relació de *master/slave* entre els seus components, però a diferència de I2C aquest no conté un dispositiu de direccionament.

• **Aspectes rellevants:**

Aquest article ens mostra com treballen els protocols I2C i SPI per aconseguir que els diferents circuits integrats de la mateixa placa es puguin comunicar. Evidentment, el nostre projecte consta de l'arquitectura SCK que funciona amb els mateixos protocols de comunicació pels diferents tipus de sensors que conté. En SCK, la majoria dels sensors es comuniquen com esclaus (slaves) del protocol mencionat I2C. Si Arduino IDE ofereix una llibreria estàndard per la comunicació I2C, els detalls de com iniciar la comunicació, o com dirigir-se serà necessari que estiguin a l'abast del coneixement del programador. El protocol SPI no s'utilitza actualment en els sensors SCK, però potser en el futur si, i per això és convenient tenir coneixement de la seva existència.

5.-

• **Títol:** The working principle of an Arduino

• **Autor/s:** Badamasi, Y.A.

• **Data:** 1 d'Octubre de 2014

• **Font d'informació:** http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6997578&queryText%3Darduino

• **Resum:**

Aquest article analitza el principi del funcionament d'Arduino. Arduino utilitza una versió simplificada de C++, que facilitza les coses als programadors, ja que es pot programar, esborrar i re-programar al mateix temps. Aquest article parla també sobre els components de la placa Arduino, el software que utilitza i una guia de com començar a fer els primers projectes amb un parell d'exemples. En general podem dir que es una guia d'inici d'Arduino uno.

• **Aspectes rellevants:**

Com ja sabem la placa SmartCitizen v1.2 està basada en Arduino, per tant aquest article ens ofereix un punt d'inici en el nostre projecte. Ens ofereix una visió aclaridora dels punts bàsics sobre Arduino, i amb els exemples podrem també conèixer millor la nostra placa. Tant a nivell de hardware, com de software.

1.5 Estructura de la memòria

Aquest document està seccionat en dos parts principals. La primera sintetitza tota la documentació estudiada de la SCK (tipus de placa, sensors que intervenen, funcions que utilitzen...), també informació sobre els efectes dels gasos mesurats i els diferents mètodes que existeixen per a regular gasos. La segona part es centra en el desenvolupament del projecte, és a dir defineix un algoritme de treball, es fan tests amb diferents valors de referència i després és passa tot a llenguatge de codi. Finalment es mostren les conclusions, els coneixements adquirits i les dificultats trobades durant els mesos d'investigació.

Primera part I

Estudi i anàlisis de la documentació

2. Estat de l'art

L'Smart Citizen Kit (SCK), consta de dues *PCBs*. La primera dedicada al processament , transmissió i emmagatzematge de dades (*Data Board*) i la segona als sensors i complements (*Ambient Sensor Board*). Tot el *firmware* està desenvolupat amb les llibreries pròpies d'*Arduino*, extraient les funcionalitats principals a llibreries en C++ natiu per simplificar el codi i permitir un major desenvolupament.

2.1 Hardware actual

La versió actual de la Data Board és la SCK 1.1 i està basada en un processador ATMEGA32U4 (AVR 8bits). El nom de la SCK és Lilypad USB i treballa a 8MHz. La versió anterior, era la 1.0 s'anomenava Leonardo i treballava al doble de freqüència i per tant tenia un consum major.
A continuació mostrem la *Ambient Sensor Board*, amb els diferents sensors:

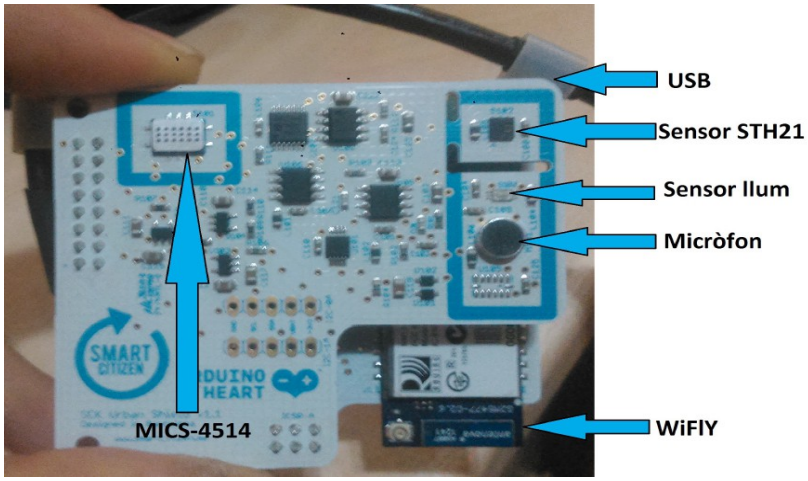

Fig. 1 Placa SCK 1.1

Com podem obervar en la Fig.1, tenim marcades les parts més importants:

- Micròfon → Ens permet detectar el nivell de soroll ambiental.
- Sensor de llum → Amb aquest sensor podem mesurar la quantitat de luminància que ens afecta.
- Sensor STH21 → Mesura la temperatura i la humitat ambiental.
- Sensor MICS-4514 → És l'encarregat de controlar les concentracions de gasos NO2 i CO.
- USB → La connexió es realitza amb un cable USB, que a més serveix d'alimentació. Una altra opció seria alimentar la placa amb una bateria i enviar les dades pel mòdul WiFi.
- Wifly → És un mòdul WiFi que ens permet visualitzar el número de xarxes WiFi disponibles i realitzar connexions inhalàmbriques,

Tots els sensors ambientals es troben en la part superior i es connecten amb la placa inferior (base) per treballar. És a dir la placa superior rep les dades ambientals i la base les processa.

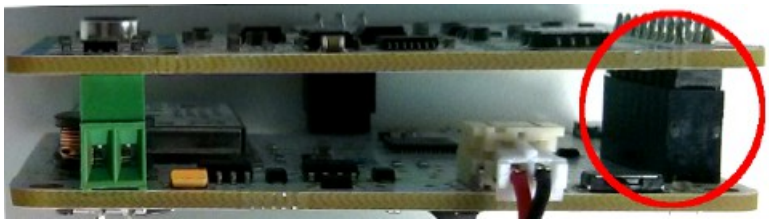

Fig. 2 Interconnexió de les dues plaques que formen la SCK

En la Fig.2 podem visualitzar com queden interconnectades les dues plaques amb els pins marcats.

2.2 Els sensors

En aquesta secció donarem informació tècnica i específica de tots els sensors i mòduls que formen la SCK.

2.2.1 Micròfon

El model de micròfon que utilitza la nostra SCK és el *Pro Signal ABM-705-RC.* És omnidireccional i pot funcionar amb voltatges de 2 fins a 10 volts. Aquest rang de valors s'ajusta als oferts per la placa perfectament. Consumeix només 0,5mA de corrent, això fa que sigui idoni per ser afegit en la SCK. Recordem que ens interessen sensors de baix cost que puguin seguir funcionant amb una bateria sense problemes.
El seu rang de freqüències d'àudio comprèn dels 50Hz fins als 16KHz i té una impedància de 2,2KOhms.
L'únic inconvenient és que a freqüències molt baixes, entre 20 i 50 Hz la nostra oïda detecta soroll, però el sensor no serà capaç de fer-ho. Tot i així la seva sensibilitat és suficient per a la majoria d'aplicacions.

En el firmware de la SCK el nivell de soroll captat pel sensor s'obté amb la funció *SCKAmbient :: getNoise ().* El micròfon està vinculat a un amplificador i a un filtre. La sortida del filtre es connecta a un convertidor d'analògic a digital (ADC) que transforma el nivell de voltatge analògic a un valor entre 0 i 1023 (10 bits). Aquest resultat s'enllaça amb el pin digital 4 (S4) i el podem llegir utilitzant la funció *analogRead(4).*
La fórmula que ens dóna el resultat respecte el voltatge d'entrada (Vcc) és:

$$L = \frac{(Valor\ S4)}{1023} * Vcc(mV)$$

Fig. 3 Micròfon *Pro Signal ABM-705-RC*

2.2.2 Sensor de llum (BH1730)

Per mesurar la quantitat de llum ambiental, la SCK fa ús del sensor BH1730. És un sensor molt comú i econòmic format per un foto-resistor o LDR (resistència que varia en funció de la quantitat de llum que rep) que treballa amb voltatges de 2.4 a 3.6 volts. Té una resolució de 16 bits que proporciona una bona qualitat. Es pot configurar quin rang de llum volem mesurar amb una comanda I2C (veure l'apartat 2.2 que explica aquest protocol). A més rebutja la llum de 50-60 Hz, aquest fet és molt important ja que és la típica freqüència de la xarxa elèctrica (llum artificial) i produiria interferències amb la llum natural.

La funció del *firmware* que ens permet visualitzar el nivell de llum és *SCKAmbient ::getLight().* A continuació s'inicialitza la comunicació I2C amb el dispositiu emprant una adreça fixa d'esclau 0101001. Rep dos bytes del bus I2C, anomenats DATA0 i DATA1, i amb la següent fórmula s'obté la intensitat en candeles (unitat del sistema internacional per definir el fluxos lluminosos):

$$Intensitat\ llum = \alpha * DATA0 - \beta\ DATA1$$

El valors $\alpha i \beta$ dependran de la relació entre DATA0 i DATA1.

Fig. 4 *Sensor de llum BH1730*

2.2.3 Sensor d'humitat i temperatura (STH21)

Aquest sensor ens proporciona informació sobre la temperatura i humitat relativa en l'entorn de la SCK. És un sensor que no cal calibrar, ja que dóna lectures molt fiables i no sofreix variacions com en el cas del sensor de gasos MICS-4514.

Consumeix molt poc, al voltant d'1mW. Té una resolució de 12 bits pels resultats d'humitat relativa i de 14 bits pels de temperatura i treballa amb temperatures de -10 fins a 80 graus Celsius.
Quan mesurem la humitat relativa, aquesta està fortament relacionada amb la temperatura, per tant si volem una mesura fiable el sensor que volem avaluar i el de referència han de tenir la mateixa temperatura per comparar la seva humitat relativa i veure si estan ben calibrats.
En el *firmware* de la SCK utilitzem la funció SCKAmbient ::getSHT21() per obtenir l'ultima temperatura i l'última humitat relativa. Rep dues variables anomenades *lastTemperature* i *LastHumidity*. Aquesta funció invoca a una altra anomenada SCKAmbient :: readSHT21 (uint8_t type) que rep com a paràmetres les següents comandes: SCKAmbient ::readSHT21(0xE3) i SCKAmbient ::readSHT21(0xE5) que mostren els valors de temperatura i humitat respectivament.

Les següent equació mostra la correcció que es fa internament pels valors rebuts d' humitat relativa (RH):

$$RH = -6 + 125\frac{(SRH)}{2^{12}}$$

El resultat és dóna en tant per cent (%). El '-6' (-6%), correspon a la mínima sortida digital que pot enviar el sensor i el '+125' correspon a la màxima. SRH és el valor mesurat pel sensor i 2^{12} és la nostra resolució.

De forma anàloga mostrem la fórmula de la temperatura:

$$T = -46{,}85 + 175{,}72\frac{(ST)}{2^{14}}$$

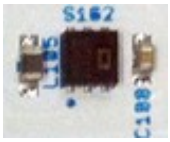

Fig. 5 Sensor de temperatura i humitat SHT21

2.2.4 Sensor de gasos NO2 i CO (MICS-4514)

La placa SCK pot mesurar els nivells de concentració dels gasos NO2 i CO amb el sensor MICS-4514. Els gasos generen electrons lliures en la superfície del detector. Aquests electrons generen un petit corrent, que si s'amplifica pot ser detectat.
Les principals característiques que defineixen aquest sensor són:

- Alta resistència als xocs i a les vibracions.
- Extens rang de detecció.
- Ampli rang de temperatures suportades.
- Alta sensibilitat.

A diferencia del sensor d'humitat i temperatura, aquest sensor s'ha d'anar re-calibrant , ja que degut a la exposició dels diferents gasos, els metalls que formen la superfície del detector canvien les seves propietats. Per fer una bona calibració necessitaríem disposar d'un laboratori on poguéssim fer mesures en una cambra amb *air zero* i després amb amb ampolles anar regulant diferents concentracions de gasos. Lògicament no disposem d'aquest material, tot i així el nostre objectiu és obtenir una calibració a partir d'unes dades simulades molt properes als valors reals que es mesurarien en una cambra. És per tant el sensor més important en el nostre projecte.

Aquest sensor pot ser vist com una resistència, que anomenarem Rs. El valor en ohms d'aquesta Rs disminueix quan la concentració de gas augmenta. Per tant Rs es una resistència variable i que estarà relacionada amb una altra resistència fixa de valor conegut que anomenarem Ro. Aquest valor Ro es troba analitzant la placa en una cambra amb *air zero* per que cap gas interfereixi en els resultats; És un valor que romandrà sempre igual i que depenent de la placa variarà. La relació entre aquests dos valors (Rs/Ro) donarà com a resultat una corba que ens mostrara la concentració de gas.
Internament MICS-4514 té dos parts ben diferenciades per a mesurar els dos gasos (NO2 i CO). A continuació mostrem l'esquema intern amb els diferents pins:

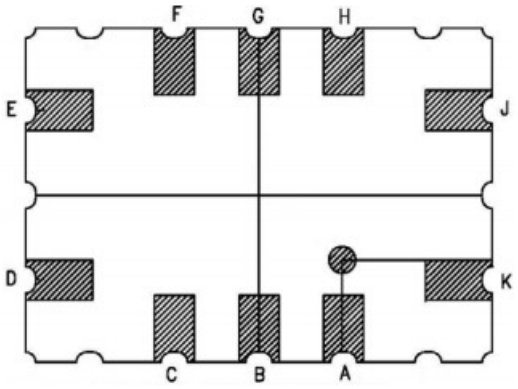

Fig. 6 Esquema intern MICS-4514

Pin	Connexió
A	Rh1 NO2
B	Rs1 NO2
C	Rh1 CO
D	Rs1 CO
E	NC
F	Rh2 CO
G	Rs2 CO
H	Rh2 NO2
J	Rs2 NO2
K	NC

Fig. 7 Taula assignació pins MICS-4514

En la fig. 6 i 7, observem la distribució dels pins. Per poder entendre millor el funcionament intern que té el sensor, observem en la Fig.8.

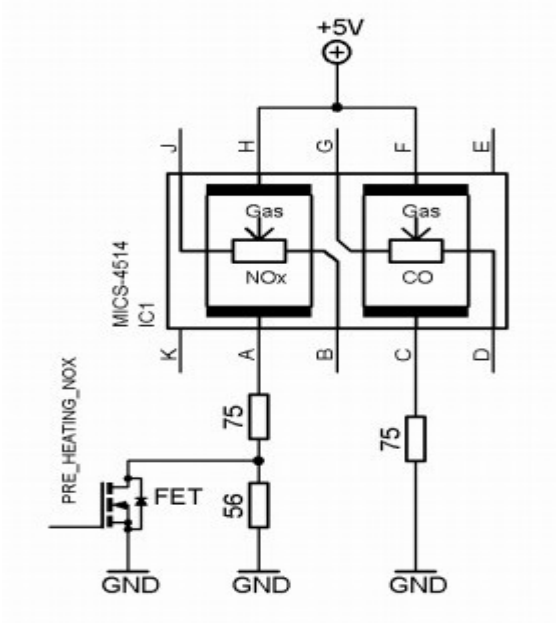

Fig. 8 Esquema intern detallat MICS-4514

A la Fig.8 podem veure que el valor de la resistència interna per a mesurar el CO és 75 Ohms i el de NO2 és 131 Ohms. Com hem dit anteriorment, quan més gran és la concentració d'un gas en l'ambient, més petita és la resistència interna del sensor. Les concentracions de CO és mesuren en parts per milió (ppm) i les de NO2 en parts per bilió (ppb), per tant és lògic que el resistor intern del detector tingui un valor més baix en la part de CO que en la de NO2.

La concentració de gas calculada, dependrà de la temperatura del sensor. Per aquesta raó abans de començar a mesurar, ha de passar una fase de *pre-heated.* Aquesta fase consisteix en augmentar significativament la temperatura interna del sensor (més de 200 ºC) per que no ens afecti la temperatura ambient.

Aquest fet provoca, que no es puguin dur a terme dues lectures consecutives en menys de deu minuts; Tot i així la SCK té una estratègia per possibilitar la lectura de dades dins d'aquests deu minuts de marge. Si el sistema detecta un valor dins d'aquest període, la API retorna l'últim valor llegit. D'aquesta manera es pot treballar de forma segura.

A continuació mostrem les corbes Rs/Ro de tots dos gasos oferides pel fabricant:

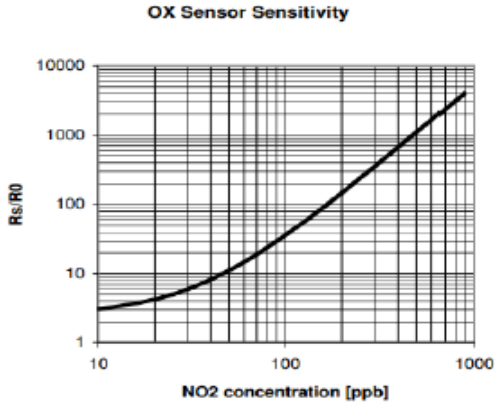

Fig. 9 Corba NO2 oferida pel fabricant

A la Fig.9, podem observar que la corba és creixent i exponencial. Quan més concentració de NO2 hi ha a l'ambient, més alt és el valor de Rs (Ro és fix).

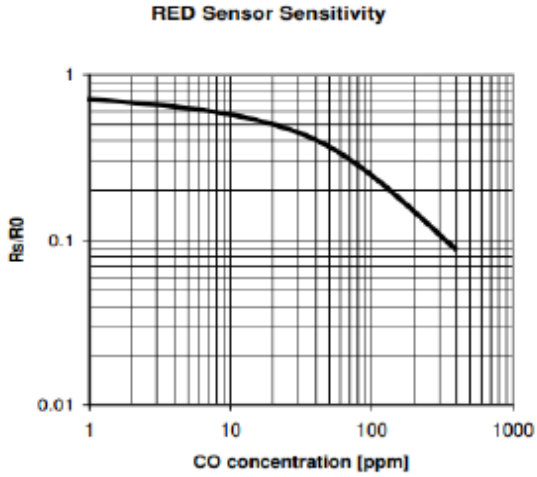

Fig. 10 Corba CO oferida pel fabricant

A la Fig.10, tenim el contrari. La corba de CO té una tendència decreixent, quant més acumulació de CO tenim en l'ambient més petit és el valor de Rs.
Cal dir, que aquestes dues gràfiques, només tenen un valor orientatiu, i en cap cas poden servir de referència per realitzar una bona calibració. Són resultats amb una humitat relativa fixada en un 40% i una temperatura de 25ºC.

En el *firmware* de la SCK la funció que permet llegir els nivells actuals de NO2 i CO és:

```
SCKAmbient :: getMICS() {
// Inicia la fase pre-heater per escalfar el sensor

heat (MICS_2710,26);  //En mA
heat (MICS_5525,32);  //En mA

// Llegeix els valors
RsNO2 = readMICS(MICS_2710);
RsCO = readMICS(MICS_5525);

}
```

En aquesta funció veiem que considera que hi ha dos sensors diferents (MICS_2710 i MICS_5525) enlloc del conegut MICS-4514. La raó és que MICS-4514 combina els dos sensors en un mateix paquet.

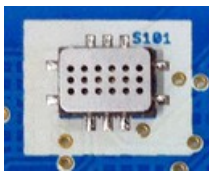

Fig. 11 Sensor MICS-4514

2.2.5 Mòdul WiFly

La SCK conté un mòdul *WiFi* model RN131. La seva funció principal es enviar les dades rebudes pels sensors cap a la plataforma d'*SmartCitizen*.
En aquest projecte no utilizem aquest mòdul, ja que la SCK està sempre connectada amb un cable USB i utilitza l'ordinador per enviar dades. Tot i així, explicarem una mica sobre aquest sensor, perquè pot resultar d'utilitat si en un futur es volen enviar dades ambientals exteriors sense la necessitat de portar un ordinador portàtil.

Les seves característiques principals són:

- Rang de freqüències de 2.4-2.5 GHz.
- Potència de sortida de fins a 18dBm.
- Consum en transmissió de dades de 210mA.
- Consum en recepció de dades de 40mA.
- Té l'opció de mantenir-se en mode *standby.*
- Conté una MAC per ésser identificat
- Utilitza l'encriptació AES128.
- Treballa amb el protocol 802.11 b/g.

La comunicació amb *WiFly* és serial, i *Arduino* ja conté una llibreria especifica amb diverses comandes per gestionar la xarxa. No utilitza el protocol I2C ni SPI (veure apartat 2.2), sinó un protocol propi de comunicació.
En el *firmware* per poder visualitzar el número de xarxes disponibles en un moment concret, fem servir la funció SCKBase::scan().

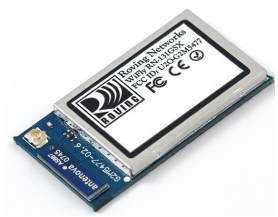

Fig. 12 Mòdul WiFly RN131

3. Efectes sobre l'organisme amb l'exposició de CO i NO2

En aquest apartat s'ha cregut convenient donar informació sobre els dos gasos que analitzarem (CO i NO2) per conèixer on es poden trobar amb més facilitat i els efectes que produeixen en el nostre organisme.

3.1 Característiques CO

El monòxid de carboni (CO), és un gas sense olor, ni color i format d'una composició més lleugera que l'aire. És altament tòxic en grans quantitats i es produeix quan no hi ha suficient oxigen per a produir diòxid de carboni (CO2). Les principals fonts de CO natural en el nostre medi ambient són els volcans, els incendis forestals i el fabricat per tots els éssers vius.
A continuació es mostren algunes concentracions de CO típiques i la seva font d'origen:

Concentració	Font
0,1 ppm	Nivell mitjà en ambient exterior
0,5-5 ppm	Interior de casa
10-25 ppm	Fum d'una cigarreta
100-200 ppm	Tub d'escapament d'un cotxe

Fig. 13 Nivell típics concentració CO

Finalment volem mostrar els efectes que té aquest gas en funció del nivell de concentració que respirem:

Concentració	Efecte	Temps d'exposició
8,7 ppm	Lleuger mal de cap	< 8 hores
52 ppm	Lleuger mal de cap i mareig	< 30 minuts
100 ppm	Nàusees, vertigen, mareig	< 15 min
400 ppm	Fort mal de cap, incoordinació muscular, debilitat, vòmits i colapse	3-5 hores
1600 ppm	Alta probabilitat de mort	< 1 hora

Fig. 14 Efectes d'exposició a CO

3.2 Característiques NO2

El diòxid de nitrogen (NO2) és un gas amb una forta olor molt desagradable. A temperatura ambient és un líquid incolor, però per a temperatures superiors als 21ºC és transforma en un gas de color marronós. Aquest gas el trobem en els tubs d'escapament dels cotxes de combustió i en els processos de combustió del carbó o el petroli. De la mateixa manera que el CO és un gas tòxic.En el cas de NO2 afecta al nostre sistema respiratori. A la següent figura podem observar els efectes que té:

Concentració	Efecte
0,01 ppm	Símptomes lleus respiratoris en nens
0,1 ppm	Augment significatiu de risc d'infeccions respiratòries
0,32 ppm	Broncopatia pulmonar quan es realitza exercici moderat o quan es tracta d'una persona asmàtica
10-20 ppm	Lleugerament irritant en adults sans
20 ppm	Perillós per a la salut
> 150 ppm	Alta probabilitat de mort

Fig. 15 Efectes d'exposició a NO2

3.3 Recomanacions

Per evitar els problemes derivats d'aquests dos gasos, s'ha de tractar evitar fer esport a l'aire lliure en els períodes més contaminants del dia, especialment la gent asmàtica i els nens. Els ambients d'interior mai han d'estar hermèticament tancats, sempre ha d'haver-hi ventilació. En cas d'estar atrapat en un lloc tancat amb qualsevol d'aquest dos gasos, tombar-se al terra i respirar molt lentament.

4. Protocols I2C i SPI de comunicació

Com qualsevol plataforma, *Arduino* utilitza els seus propis protocols de comunicació. En aquest cas són dos*: Inter-Integrated Circuit* (I2C) i *Serial Peripheral Interface* (SPI).
El protocol I2C va ser creat per Phillips en 1982 per connectar diferents dispositius. Per altra banda SPI va ser desenvolupat per Motorola en l'any 1985 amb el mateix objectiu. Tots dos s'han convertir en l'actualitat en estàndards.

El bus que produeix I2C conté quatre línies:

- SDA → Línia de transmissió de les dades.
- SCL → Línia amb el senyal per sincronitzar les transferències de dades.
- VCC → Alimentació.
- GND → Massa

És un protocol basat en mestre-esclau, on el mestre inicialitza la comunicació i l'esclau simplement l'accepta.
Cada dispositiu (mestre o esclau) té una única adreça que varia entre 7 i 10 bits. El funcionament del protocol és simple: primer de tot, el mestre posa l'adreça de l'esclau en el bus i especifica amb 1 bit si vol llegir o escriure dades. A continuació l'esclau respondrà al mestre i començarà l'intercanvi de dades, que dependrà en funció del tipus d'operació que s'executi. Aquesta transmissió és *serial*, és a dir bit a bit.
La línia SDA conté els bits reals d'informació, mentre que SCL és un senyal de rellotge generat pel mestre. La velocitat màxima de transmissió de dades en aquest protocol és de 5 Mbps, en un mode anomenat ultra ràpid (Ufm).
En la nostra SCK tots els sensors utilitzen aquest protocol per comunicar-se. Els avantatges que ofereix són:

- Mecanisme molt simple mestre-esclau, com hem explicat anteriorment.
- Hardware de fàcil implementació que només usa dues senyals (dades i rellotge).
- Possibilitat d'escollir cada dispositiu amb una única adreça, això permet afegir nous terminals sense modificar l'arquitectura del sistema.
- *Arduino* conté una llibreria oficial per suportar I2C, anomenada *Wire.h.*

Per altra banda, també té alguns inconvenients:

- Si tenim dispositius que suporten altes velocitats de transmissions i estan connectats amb altres que només suporten velocitats baixes, la velocitat final sempre serà la més baixa del sistema.
- Elèctricament parlant, I2C empra connexions anomenades *open-drain* , aquestes requereixen més corrent i això afecta al consum elèctric de la SCK.

SPI també funciona amb un sistema mestre-esclau tal com hem vist en I2C. La diferència entre un i l'altre, és que SPI no utilitza una adreça numèrica per seleccionar a cada dispositiu, enlloc d'això utilitza un senyal CS que indica quin dispositiu necessita comunicar-se amb el mestre. Quan aquest senyal està actiu, el mestre
selecciona als dispositius que la contenen i la comunicació pot establir-se. Aquest protocol treballa amb dos senyals més que indiquen si el mestre està escrivint i l'esclau llegint (MOSI) o a l'inrevés (MISO).

Mostrem alguns avantatges respecte I2C:

- Consum inferior. Aquest punt té especial rellevància si alimentem la SCK amb una bateria.
- No necessita adreces numèriques per a cada terminal, simplement envia un senyal per fer la selecció.

I també el principal problema que genera:

- No té un estàndard formal, i això provoca que quan afegim nous sensors a la SCK no tinguem cap garantia de que funcionin correctament.

5. Estructura bàsica Arduino

La SCK està programada amb codi *Arduino,* i aquest codi es pot dividir en tres parts.
La primera part correspon a la definició de les llibreries necessàries en el projecte. Un exemple de com es defineix una llibreria en Arduino seria el següent: #include <Arduino.h>

La segona part és el *setup* del sistema que s'inicia al mateix temps que arranca la placa o quan es fa un *reset*. La funció és void setup() i en el cas concret de la SCK, la classe *SCKAmbient* implementada en el *firmware* conté la funció ambient_ini() que s'encarrega de configurar el rellotge de la SCK i també d'inicialitzar els sensors de la placa.

Un cop s'ha executat void setup(), arriba la tercera part del codi que és la funció void loop(). Aquest bucle conté una funció anomenada execute() de la classe SCKAmbient, que observa les últimes mesures dels sensors i mira si ha passat un temps d'espera suficient entre mesura i mesura (10 minuts). El microcontrolador només necessita activar-se per llegir els sensors i per enviar dades, la resta del temps pot romandre en un estat *standby,* que li permet estalviar energia. Es pot forçar aquest mode canviant el pin amb valor AWAKE per HIGH. Si realitzem aquesta operació ja no executarà cap més instrucció o comanda introduïda. La única forma de sortir d'aquest mode d'espera és a través d'una interrupció del propi *hardware.* Una altra manera d'entrar en mode *standby* és canviant el valor del pin AWAKE per LOW, amb aquest canvi la SCK es manté en espera tot el temps, excepte quan ha d'enviar dades.

De forma resumida podem dir que l'estructura bàsica seria:

```
#include <llibreria.h>
void setup {
//Funcions que només s'executaran 1 cop.
}
void loop  {
//Funcions que s'executaran dins d'un bucle, normalment amb condicions
}
```

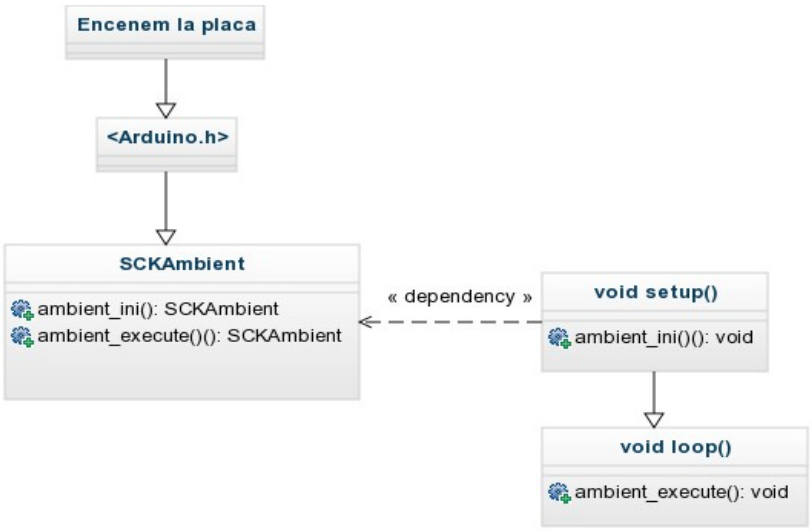

Fig. 16 Estructura bàsica codi Arduino

6. Mètodes de calibració

6.1 Introducció

La contaminació atmosfèrica en el medi urbà és una de les principals preocupacions en les grans ciutats i països creixents. Afecta considerablement a la salut i és responsable d'una gran varietat de malalties respiratòries, com per exemple l'asma i problemes ambientals com la pluja àcida.

Avui dia, la contaminació de l'aire es controla amb xarxes d'estacionament de mesura estàtica, també anomenades estacions fixes i són controlades per les autoritats oficials. Són molt fiables i poden mesurar amb una alta precisió una amplia gama de contaminants atmosfèrics utilitzant instruments analítics de mesura com espectròmetres de masses i cromatògrafs de gasos. Els principals inconvenients d'aquests aparells complexes de mesura són la seva gran mida, l'alt preu i un manteniment costós. Degut a això, varis grups d'investigació van començar a mesurar contaminants atmosfèrics amb sensors de gasos d'estat sòlid. Aquests sensors són
petits, tenen un baix cost i són adequats per controls mòbils. L'únic problema és que s'han de calibrar amb molta més freqüència que les estacions fixes i que la precisió és més baixa. Tot i així són molt recomanables en la majoria d'aplicacions.

6.2 Fases per realitzar una calibració

Els sensors de gasos han de ser calibrats periòdicament per assegurar la precisió del sensor i la integritat del sistema. És important instal·lar sensors estacionaris en llocs on es poden calibrar fàcilment. Els intervals de calibració poden ser diferents en cada sensor i generalment cada fabricant dóna un diferent. És una bona pràctica analitzar sempre qualsevol sensor els primers 30 dies després de la seva instal·lació per veure si en aquest període s'ha adaptat correctament o no. Si durant aquests primers 30 dies el seu funcionament és correcte, ens proporciona un alt grau de confiança per seguir treballant. En aquest període de prova, s'hauria de revisar setmanalment el sensor.

El mètode i el procediment per a calibrar qualsevol sensor s'ha d'establir immediatament, ha de ser simple, eficaç i fàcilment executat per qualsevol usuari.
La calibració d'un sensor de gas implica dues fases. La primera és establir una lectura d'aire z*ero* i la segona es mesurar el gas a partir d'una referència (*span calibration).*

- **Establir una lectura *zero*** → No existeix un norma establerta que defineixi l'aire zero. Molts dels procediments analítics utilitzen nitrogen o aire sintètic pur per establir aquest punt. La principal raó és que les ampolles de nitrogen i generar aire sintètic pur són fàcils d'aconseguir; En conseqüència popularment es creu que són bons mètodes per posar a zero un sensor de gasos.
  Per desgràcia, aquest procediment és incorrecte ja que l'aire que respirem conté diferents concentracions de gasos a part de nitrogen u oxigen. A més l'aire també conté un petit percentatge de vapor d'aigua.

  Per aquesta raó és molt més pràctic i realista calibrar el sensor en una zona que es consideri neta. Aquest punt pot ser difícil de trobar, però un exemple senzill seria un ambient d'oficina, ja que representaria l'aire zero en l'àrea de treball del sensor.

Tenint en compte factors com el tipus de sensor i la classe d'aplicació que es vol fer, es proposen dos mètodes de calibració d'aire zero:

A. En aplicacions on l'aire és normalment net i jutjant que no existeixi cap condició anormal que pugui afectar al resultat, una forma molt senzilla és utilitzar una bossa de plàstic que cobreixi el sensor per complet durant uns minuts. Aquest procediment tan simple s'utilitza a vegades per detectar errades en les lectures que no tenen lògica.

B. Utilitzar aire comprimit. Aquest té l'avantatge que és fàcilment regulable i està disponible en ampolles en establiments especialitzats. El problema és que aquestes ampolles contenen petites concentracions d'hidrocarburs i de diòxid de carbó, per aquesta raó l'aire comprimit té un nivell molt baix d'humitat. El nivell d'humitat en l'aire és molt important si utilitzem sensors de tipus sòlid o

PIDS, ja que necessiten el vapor d'aigua per funcionar. Si utilitzem per tant, un humidificador per pal·liar aquest problema, tenim un sistema molt fiable per mesurar aire zero. Un humidificador molt senzill seria afegir un mocador de paper mullat, i una altra opció seria afegir un tub de nafió. Tot dependrà dels recursos de l'usuari.

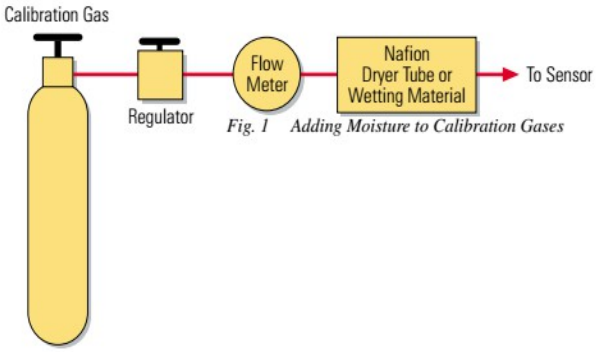

Fig. 17 Esquema calibració amb aire comprimit

- ***Span calibration*** → Aquesta calibració pot resultar o molt senzilla o molt complexe i costosa depenent del tipus de gas i dels rangs de concentració en que treballem. En principi, per aconseguir la millor precisió possible, s'ha d'afegir una quantitat equilibrada del gas que volem mesurar en aire ambiental net. Aquest procediment pot semblar fàcil d'implementar, però en realitat és una operació complexe que requereix certa habilitat. A continuació es mostren alguns mètodes d'*span calibration:*

  A. Gas de calibració prèviament barrejat. Aquest gas és comprimit i emmagatzemat sota pressió en una ampolla de gas. Aquestes ampolles estan disponibles en dues categories: a pressió baixa i a pressió alta. Les ampolles són lleugeres,retornables i normalment estan fetes d'alumini. Molts gasos es poden barrejar prèviament amb aire i ser emmagatzemats a baixa pressió, però d'altres només es poden barrejar amb un gas inert com el nitrogen.
  També existeixen casos concrets de gasos que requereixen ampolles especials i el temps de vida de cada gas abans de que expiri i sigui inutilitzable també varia.

  B. Dispositius permeables. Són recipients que contenen productes químics en vapor i fase líquida en perfecte equilibri. Un tub de permeabilització de gas emet de forma continua aquests productes químics, depenent de la pressió del recipient, la velocitat de permeabilització de cada gas pot variar. Per exemple, amb una pressió de vapor d'aigua molt alta els gasos penetraran més ràpidament, en canvi amb una pressió molt baixa els productes químics que conté el recipient fan que els gasos penetrin molt més lentament. Aquest mètode es poc comú i només s'utilitza en laboratoris i amb aplicacions molt analítiques.

  C. Calibració creuada. Aquest mètode aprofita les interferències de gasos externs en el sensor. Per exemple, per un sensor calibrat al 100% d'hexà, de forma general és més senzill utilitzar un 50% de gas metà. Això es degut, a que el gas hexà és líquid a temperatura ambient i té una baixa pressió de vapor. Per tant, és més difícil fer una barreja precisa i mantenir-ho en una pressió alta.

6.2.1 Algunes eines de calibració

Per dur a terme els procediments explicats anteriorment les següents eines són necessàries:

- Xeringa amb agulla → És l'eina més econòmica per mesurar la quantitat d'un gas. Existeixen micro-xeringues per volums molt petits de gasos i altres amb més capacitat. Aquestes últimes són més difícils d'aconseguir i normalment les més grans són d'uns 1000 centímetres cúbics (1000 cc). Per tant per grans volums de mesura no s'aconsellen.

Fig. 18 Xeringa de 1000 cc

- Bossa de calibració → La majoria de material utilitzats per l'emmagatzematge d'aliments són inerts, en cas contrari es contaminarien. Per aquesta raó aquests tipus de bosses poden ser emprades per mantenir la majoria de productes químics sempre que s'utilitzin en períodes curts. Aquest és un punt important a tenir en compte, ja que les molècules de gas es s'expandeixen lentament per les diverses capes de la bossa. L'exemple més clar és la bossa Tedlar; Està formada per flourur de polivinil i té una baixa absorció de molècules. Conté una vàlvula i un septe que s'utilitza com a injecció del gas.

Fig. 19 Bossa Tedlar

6.3 Factors a tenir en consideració

La temperatura de calibració d'un sensor és d'uns 25ºC (293 K). Quan aquesta temperatura augmenta o disminueix, el valor Rs varia i per tant obtindrem una lectura de concentració de gas errònia. Observem com afecten aquest canvis de temperatura al nostre sensor MICS-4514:

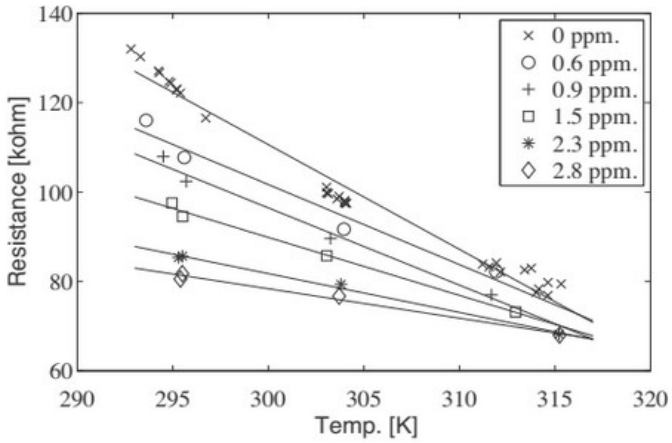

Fig. 20 Resposta en temperatura del sensor MICS-4514
a varies concentracions CO.

A la Fig. 17, podem veure fins a sis concentracions de CO diferents. Observem que quan més baixa és la concentració, més alt és el valor Rs i quant més alta és la concentració, menor és el valor. El punt d'inici de la gràfica és a 293 K (20º C) i conforme la temperatura augmenta, les resistències van disminuint de forma lineal.

Amb aquesta informació podem saber amb només dos punts (el extrems) quina serà a degradació que sofrirà el sensor amb diferents temperatures fent interpolació de punts lineal. La fórmula d'interpolació de punts és:

$$Y = Ya + (X - Xa)\frac{(Yb - Ya)}{Xb - Xa} \quad ;$$

Fem la prova en el punt Rs = 100 Kohms (Ya) ,Rs = 80 Kohms (Yb) i Temp = 293 K (Xa) , Temp = 312 K (Xb) corresponent a la concentració de 1.5 ppm de CO. Substituïm els valors a la fórmula o obtenim la següent equació:

$$Y = 408{,}42 - 1{,}05\,X \quad ;$$

Amb aquesta equació ja podem esbrinar el valor de Rs amb qualsevol temperatura per a una concentració d'1.5 ppm de CO.

De forma anàloga, si la temperatura ens afecta a la mesura de Rs també afecta al valor de Ro. Ro es calcula com hem explicat en apartats anteriors, amb aire zero i és un valor fixe a temperatura constant. Observem com afecten aquests canvis en tres sensor diferents:

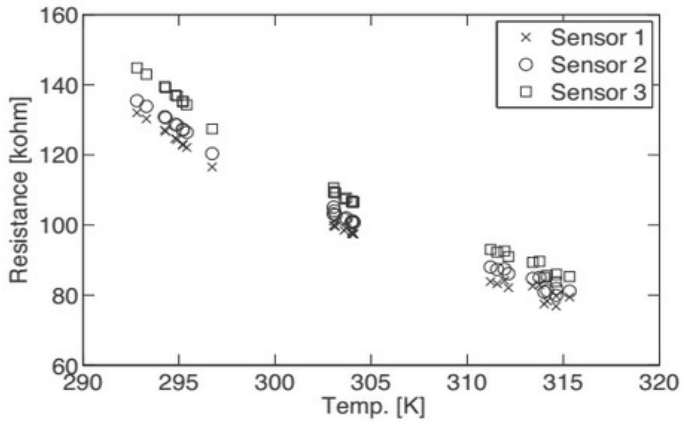

Fig. 21 Variabilitat de tres sensors amb concentracions d'aire zero sotmesos a diferents temperatures.

Podem visualitzar que passa el mateix que amb Rs. Cada sensor té el seu propi valor de Ro, però tots comparteixen una degradació lineal amb l'augment de la temperatura, que podríem trobar tornant a aplicar la fórmula d'interpolació de punts. Per a concentracions de NO2 succeeix el mateix ja que tots dos utilitzen el mateix sensor.

Un altre factor a tenir en compte és la degradació del propi sensor. Qualsevol aparell electrònic amb el pas del temps va perdent precisió en les seves mesures. A continuació mostrem una gràfica amb l'estudi de la degradació soferta pel sensor MICS-4514 al cap de més de 250 dies de treball.

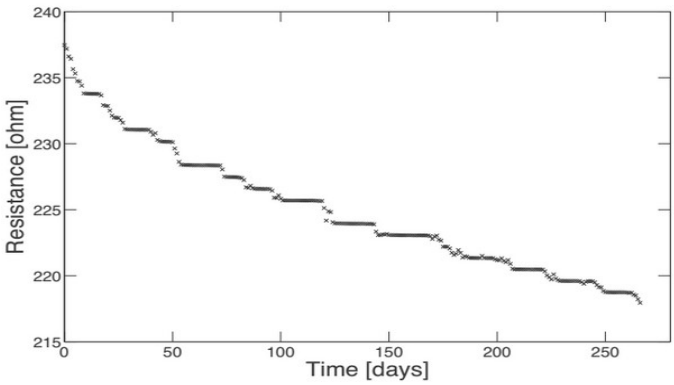

Fig. 22 Degradació temporal MICS-4514

De la Fig. 19, podem concloure que amb una calibració cada 15 dies és més que suficient, ja que no té pèrdues de valor significatiu fins passats els primers 50 dies.

6.4 Mètodes de calibració

Tres mètodes de calibració seran testejats: *linear regression (LR), multivariate linear regression (MLR)* i línia d'aproximació (LA).

6.4.1 Linear regression (LR)

Per a cada sensor s'estableix una funció de calibració assumint la linearitat de les seves respostes amb les de referència per a cada gas. La seva funció de calibració és del tipus: $Rs = aX + b$, on Rs respresenta la resposta donada pel sensor, X la mesura de referència i a/b són coeficients constants. Aquesta fórmula també la podem utilitzar de forma inversa, amb la finalitat de predir els nivells de contaminants de l'aire, tot dependrà de les nostres necessitats: X = (Rs-b)/a. Si ens fixem aquest equació correspon en realitat a una interpolació lineal. Per optimitzar aquest mètode s'haurien de tenir en compte diversos valors, per fer diverses interpolacions. D'aquesta manera la corba resultant seria molt més precisa.

6.4.2 Multivariate linear regression (MLR)

Aquest mètode és basa en LR, però té en consideració més variables. Concretament per mesurar el nivell de NO2, té en compte com a referències les lectures d'Ozó (O3) i d' òxid de nitrogen (NO). I també els valors de temperatura (T) i humitat relativa (RH).
De la mateixa manera que LR la seva funció de calibració consisteix en una equació del tipus: Rs = f(X,Yi), on f(X,Yi) és una funció amb múltiples valors de referència.

La fórmula és : $NO2 = \frac{(Rs - bO3 - cT - dRH - e)}{a}$;

On els valors a,b,c i d són coeficients de calibració amb un valor constant.
Aquest mètode és més precís i complex que LR, però si el particularitzem per al nostre sensor tindríem problemes ja que no mesura l'O3 i per tant no podríem obtindre una mesura fiable.

6.4.3 Línia d'aproximació (LA)

Aquest sistema de regulació consisteix en comparar dos punts d'un sensor amb dos punts de referència, amb les mateixes concentracions de gas. És té en compte un punt inicial i un punt final Rs/Ro. La seva funció seria: $C = \frac{A}{B} * X$;

On A= Rs/Ro de la placa a calibrar, B= Rs/Ro de la placa calibrada en el mateix punt de concentració que A i X = Punt Rs/Ro de A que es vol calcular a partir de B.

En conclusió, troba una desviació entre les dues entrades Rs/Ro i aplica aquesta desviació per a la resta de valors, obtenint una corba nova calibrada.

6.5 Resum

Abans de calibrar qualsevol gas, primer de tot hem de trobar el seu valor Ro amb aire zero. Aquest valor normalment el dóna el fabricant de la placa, però s'ha de tenir en compte que les cambres on es fan les proves d'aire zero estan a temperatures d'entre 20-25º C (293-298 K). Si la temperatura ambient és molt diferent, hem de tenir en compte la variació d'aquest valor i aplicar un corrector. Un cop l'obtenim, mesurem Rs.

Tenim moltes opcions per realitzar una calibració depenent dels nostres recursos i de la precisió que necessiti la nostra aplicació. Sempre treballarem amb uns valors de referència i s'ha de tenir en compte que mai tindrem uns resultats exactes. Tenint en compte la degradació temporal del sensor i els efectes que produeix la temperatura el més adient seria realitzar un control setmanal el primer mes, i a partir del segon realitzar un regulació cada quinze dies per garantir fiabilitat a les dades recollides.

Segona part II

Desenvolupament e implementació d'un algoritme de calibració per a la SCK

7. Dades de calibració

El primer pas es obtenir unes dades reals del sensor MICS-4514 en un laboratori, que ens serviran de referència. A la figura següent es mostren aquests valors enviats des d'Smart Citizen:

Punts corba NO2			KIT DEEP NO2 (simulacion)		
Rs/Ro	ppb		R0 (KOhm	RS (KOhm	ppb
3	10		10	10	0
4.5	20			30	10
8.5	40			50	20
20	70			80	40
39.7	100			150	70
180	200			300	100
700	400			450	200
1800	600			600	400
3000	800			750	600
4000	900			900	800

NO2 MEDICIONES (simulaciones)						
KIT 1		KIT 2		KIT 3		ppb (exposición en chamber)
R0 (KOhm)	RS (KOhm	R0 (KOhm	RS (KOhm	R0 (KOhm	RS (KOhm)	
10		7		6		0
	1450		1100		900	1000

Punts corba CO			KIT DEEP CO (simulacion)		
Rs/Ro	ppm		R0 (KOhm	RS (KOhm	ppm
0.72	1		5	5	0
0.68	2			4.3	1
0.63	5			3.8	2
0.58	10			3.2	5
0.5	20			2.5	10
0.35	50			1.5	20
0.25	100			1	50
0.16	200			0.8	100
0.11	300			0.6	200
0.09	400			0.5	300

ROJO: Niveles muy altos!

CO MEDICIONES (simulaciones)						
KIT 1		KIT 2		KIT 3		ppm (exposición en chamber)
R0 (KOhm)	RS (KOhm	R0 (KOhm	RS (KOhm	R0 (KOhm	RS (KOhm)	
12		9		7		0
	250		135		120	20

Fig. 23, Quadre amb punts de la corba de referència de CO i NO2 i amb 3 KITS de proves per a cada gas.

7. 1 Dades de referència i de test NO2

Per tenir una visió més aclaridora d'aquestes dades, farem gràfiques amb aquests punts. En l'eix Y tindrem Rs/Ro i en l'eix X la concentració del gas en ppb (NO2) o ppm (CO). Primer començarem pels punts de la corba NO2:

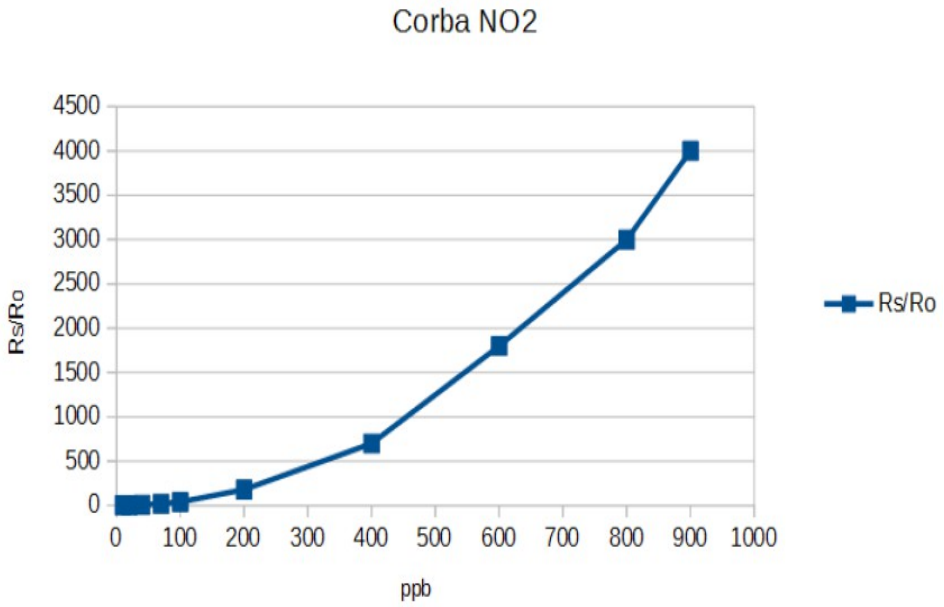

Fig. 24 Corba NO2 *datasheet*

Un cop hem introduït els primers punts del quadre anomenats com 'Punts corba NO2' obtenim la gràfica de la Fig.21 que es correspon a la oferta pel *datasheet*. Ens donen directament el valor Rs/Ro, no tenim forma de saber per tant quin és el valor Ro en que s'ha basat el fabricant per realitzar aquestes mesures. Aquest fet ja provoca que no siguin valors gaire fiables. Tot i així ens servirà per veure com varia respecte unes dades reals de mesura.

Ara fem el mateix pels punts 'KIT DEEP NO2' que és la referència que tindrem d'aquest gas. En aquesta ocasió tenim els valors Rs i Ro per separat, simplement els dividim per obtenir Rs/Ro en l'eix Y i obtenim la següent corba:

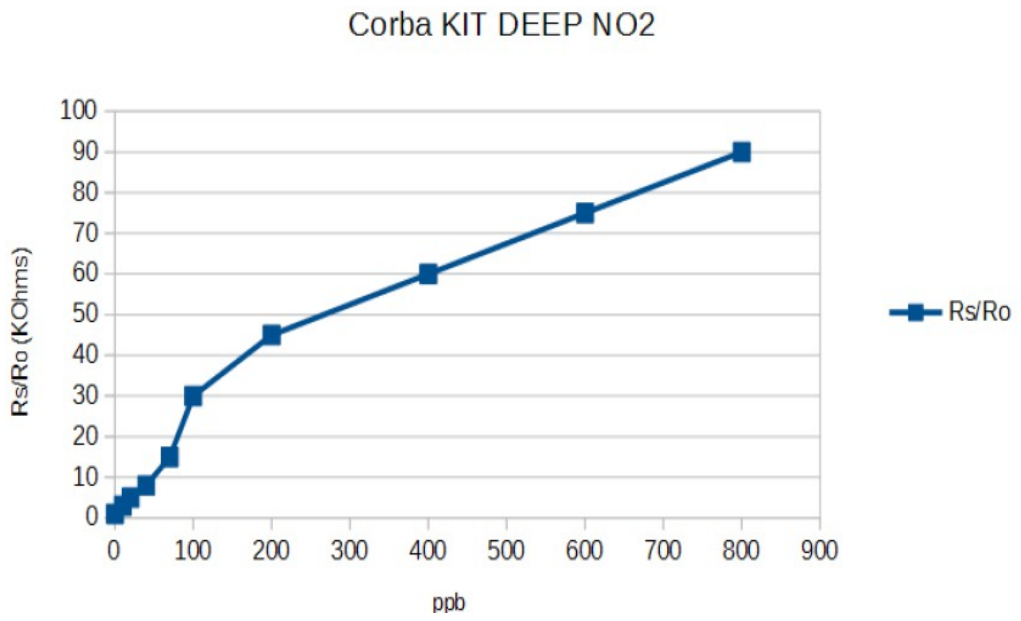

Fig. 25 Corba NO2 referència

Aquesta gràfica té un valor Ro= 10KOhms, és un valor diferent als altres KITS de mesura. El KIT 1, té la mateixa Ro, per tant els seus punts Rs/Ro han de ser gairebé idèntics als de referència. El KIT 2, té una Ro= 7 i el KIT 3 una R0 = 6, tots 3 KITS ofereixen diferents lectures de Rs amb la mateixa concentració de gas (1000 ppb).

La següent figura mostra una comparativa d'aquestes Ro i els valors Rs mesurats:

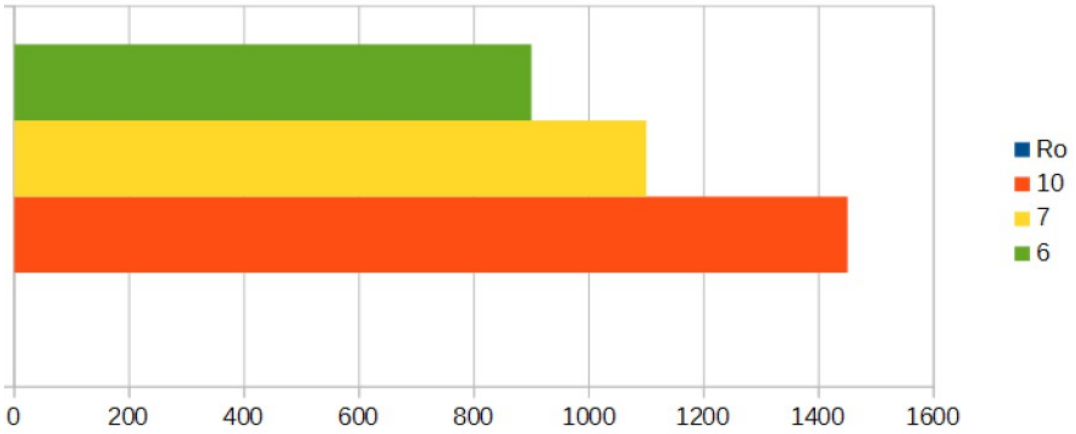

Fig. 26 Comparativa diferents valors Ro-Rs

La Fig. 23, ens indica que quan més gran és un valor de Ro, més gran serà el valor de Rs. Però això no vol dir que la relació Rs/Ro sigui més gran també. Els 3 KITS de proves podrien ser o bé 3 plaques diferents, o bé la mateixa placa mesurada amb diferents temperatures (veure apartat 4.3).

7.2 Dades de referència i de test CO

Repetim el mateix procés anterior, però ara amb les dades de CO. Primer mostrarem la gràfica del *datasheet* i a continuació la de referència d'aquest gas. Els valors Rs/Ro són molt més petits que en el cas anterior, però hem de recordar que els nivells de concentració estan en parts per milió (ppm), és a dir que aquest gas té una concentració molt més alta en el medi ambient.

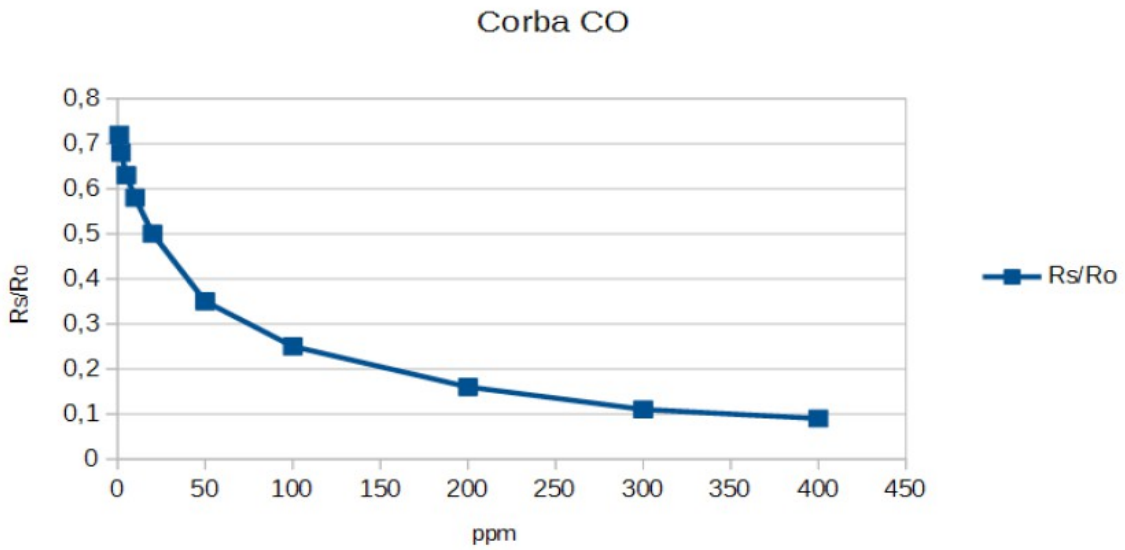

Fig. 27 Corba CO *datasheet*

A la Fig. 24 Observem la tendència que sofreix el nostre sensor quan està exposat a diverses concentracions de CO. Quan més alt és aquest valor de concentració, més baixa és la relació Rs/Ro. Segueix per tant una dinàmica decreixent, tot el contrari que NO2 que segueix una tendència creixent.

Cal afegir un apunt curiós, amb la degradació del sensor i amb l'augment de temperatura les lectures de NO2 seran més baixes que les reals. En canvi en el cas de CO provoca nivells de concentració més alts. Per corregir aquest problema, aplicaríem un *offset* per a cada gas. Aquest *offset* s'hauria de trobar mesurant com varia per cada grau centígrad (º C) la resposta del nostre sensor.

Ara mostrem la corba formada pels valors de 'KIT DEEP CO'. En aquest cas la nostra Ro val 5 KOhms i per tant haurem de dividir tots els valors de Rs entre aquest resultat.

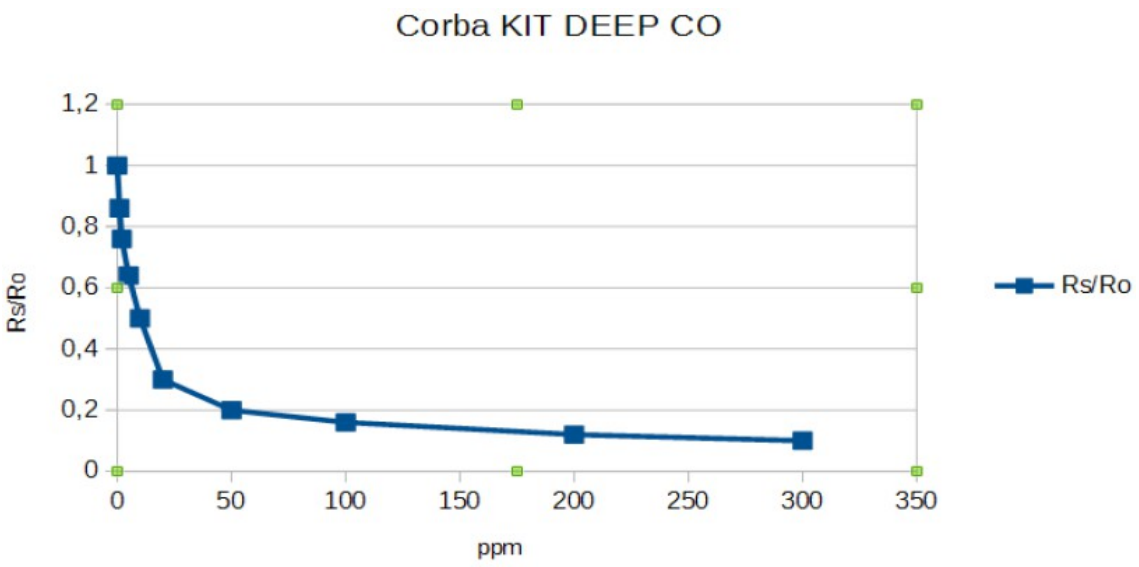

Fig. 28 Corba CO *referència*

Podem observar unes lleugeres diferencies amb la corba del *datasheet*. A diferència de la corba NO2, en aquest cas hem d'apuntar que els nivells de concentració més grans de 50 ppm, és consideren molt alts i poc comuns. Tot i així, s'ha cregut necessari afegir-los en la Fig. 25.

7.3 Algoritmes de test

En l'apartat 4.4, posàvem de manifest tres mètodes diferents per regular sensors (LR,MLR i LA). D'aquests tres, per al nostre projecte ens interessen dos. Concretament LR i LA. El procediment MLR ha quedat descartat, ja que és més complex i els requeriments que necessita el nostre sensor no els assoleix (no pot llegir O3). Amb les dades de test i de referència aplicarem aquests dos recursos per a cada gas i veurem els resultats.

7.3.1 Test Linear regression (LR)

Com hem mencionat, és tracta d'una interpolació lineal de punts (apartat 4.4.1). La fórmula d'interpolació lineal és: $Y = Yo + \frac{(Y1 - Yo)}{X1 - Xo}(X - Xo)$;

Yo = 1, és el punt d'inici establert (Rs/Ro) per a qualsevol gas amb aire zero.

Xo = 0, en condicions inicials el valor de concentració sempre és zero.

X1,Y1 és el punt conegut de mesura (Rs/Ro i concentració), o vist d'una altra manera és l'extrem de la corba que trobarem.

Substituïm els valors dels KITS 1,2,3 de NO2 i obtenim les següents equacions de la recta:

Y1 = 1 + 0,144X

Y2 = 1 + 0,156X

Y3 = 1 + 0,149X

Ara hem d'anar modificant el valor d'X pels diferents valors de concentració i obtindrem les respostes Rs/Ro en tots els punts. A continuació mostrem les 3 corbes superposades amb la referent:

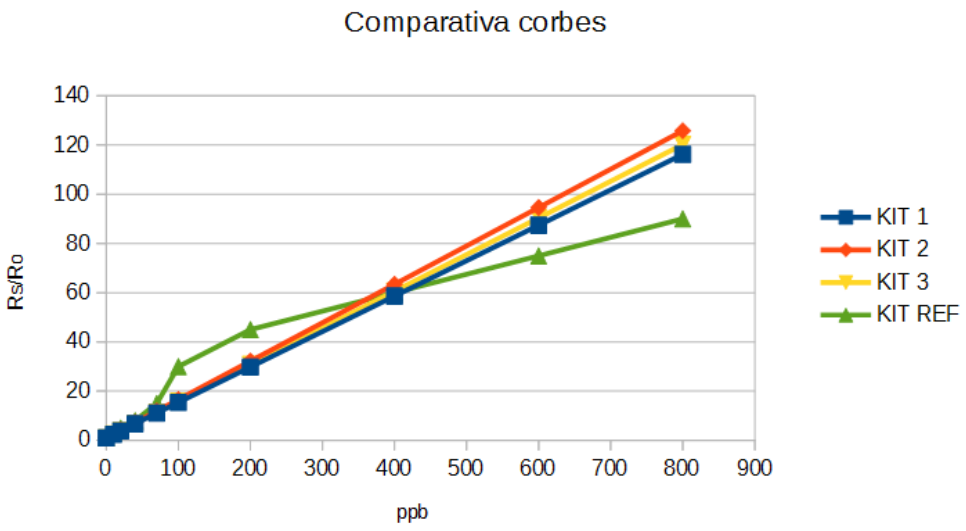

Fig. 29 Corbes dels KITS NO2 amb procediment LR

Observem que per a valors de concentració entre 300 i 500 ppb és un mètode vàlid. En canvi, per a concentracions baixes o altes ofereix un desviament important en les dades.

Repetirem el mateix procés per als KITS 1,2,3 de CO i obtenim les següents equacions:

Y1 = 1 - 0,0295X
Y2 = 1 - 0,031X
Y3 = 1 – 0,0325X

En aquesta ocasió les equacions tenen un signe negatiu, això és degut a que el valor que multiplica X, és inferior a 1 i per tant la corba serà negativa. El punts dels KITS estan mesurats amb 20 ppm, per tant no avarca tot l'espectre sinó que es centra en els valors més comuns que podem trobar en l'aire. Substituïm pels valors corresponents i comparem les corbes obtingudes amb la de referència:

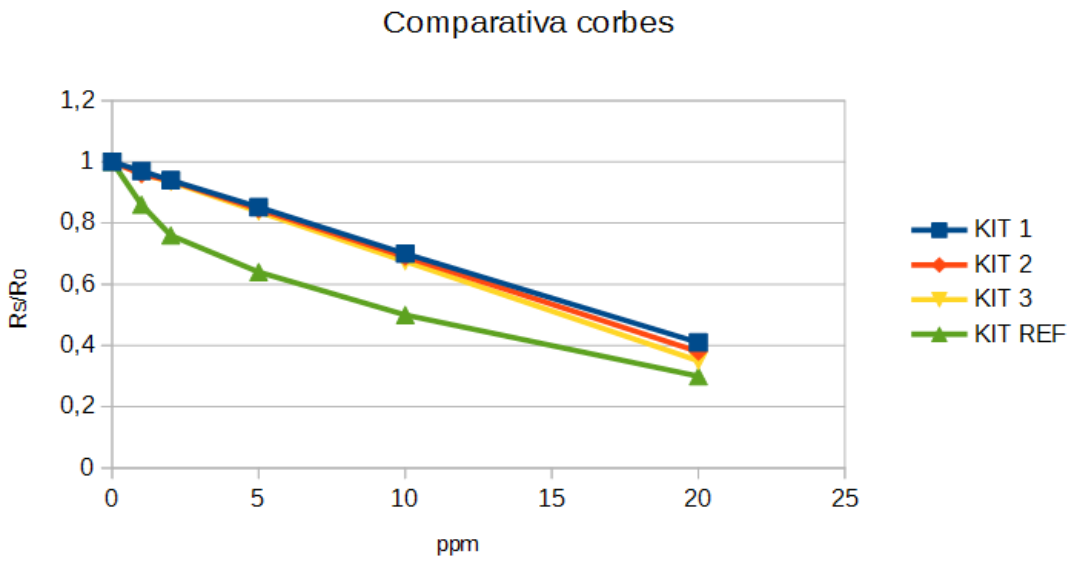

Fig. 30 Corbes dels KITS CO amb procediment LR

7.3.2 Test Línia d'adaptació (LA)

Recuperem la fórmula que es correspon a aquesta metologia (apartat 4.4.3):

$$C = \frac{A}{B} * X \quad ;$$

Substituïm pels valors dels KITS 2 i 3 de NO2 i obtenim:

C2 = 1,08X
C3 = 1,03X

El KIT 1 al tenir la mateixa Ro que les dades de referència, es converteix en el punt de referència per a concentracions de 1000 ppb. Això es degut a que no tenim dades reals d'aquesta concentració concreta i al ser un algoritme de comparació hem agafat el KIT 1 que es quasi idèntic.

Els resultats de C2 i C3 ens diuen que tenen una desviació d'un 8% i d'un 3% respectivament. I s'aplicarà un *offset* tenint en compte això. A la figura següent mostrem les dues corbes superposades amb la de referència:

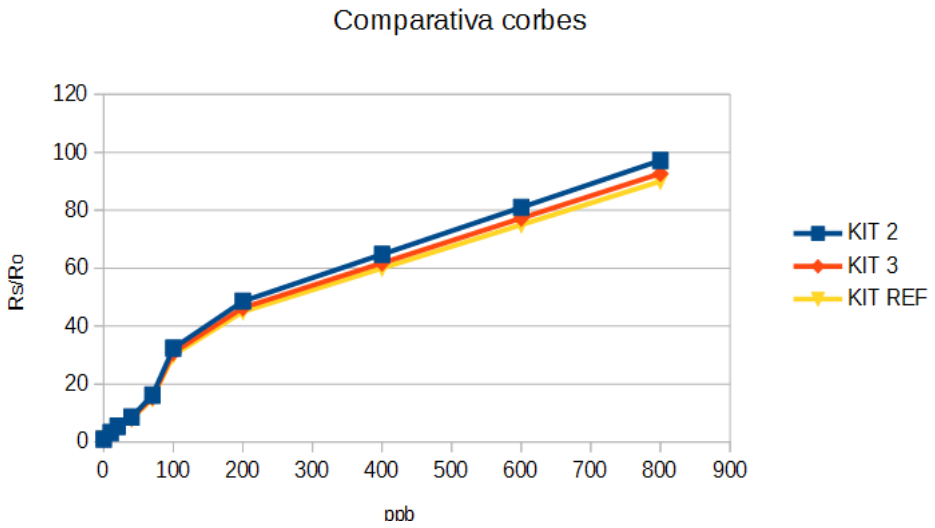

Fig. 31 Corbes dels KITS NO2 amb procediment LA

A primera vista podem observar que aquest mètode en el cas de mesures deNO2 s'aproxima més a la realitat que no pas l'altre. Ara comprovarem si manté la seva efectivitat amb CO:

Trobem les equacions que definiran els KITS 1,2 i 3 de CO:

C1 = 1,06X
C2 = 1,08X
C3 = 1,1X

En aquesta ocasió tenim desviacions d'un 6%, un 8% i un 10% en els KITS 1,2 i 3 respectivament. A continuació mostrem les tres corbes resultants superposades amb la de referència:

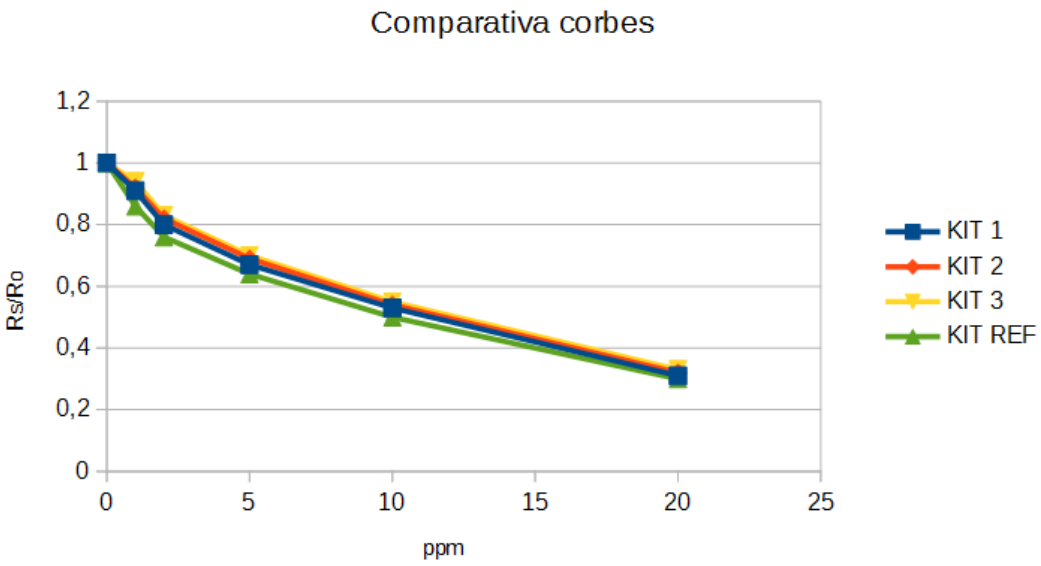

Fig. 32 Corbes dels KITS CO amb procediment LA

Un cop hem comparat els dos procediments amb els dos gasos, podem concloure que el mètode LA és mes precís que no pas LR. També cal dir, que en cas de no tindre una referència, el mètode LR seria una bona opció, ja que només requereix dos punts independentment de si hi ha o no dades fixades. En el nostre cas, aquests valors els tenim i aquest fet ens proporciona obtenir millors aproximacions.

8. Programació I

Un cop definida la nostra estratègia de calibració (LA) procedim a desenvolupar l'algoritme en codi. El primer que farem serà crear un arxiu de text (.txt) on guardarem les dades de referència dels dos gasos. Aquest arxiu serà creat en l'entorn *Arduino IDE* i un cop estigui creat és guardarà en una tarja microSD que afegirem a la SCK. A continuació podrem llegir pel *Serial* totes les dades.

Aquest programa té diverses funcions les principals són:

- Mostrar com gravar les dades recollides d'un sensor en una microSD.
- Obtenir un historial de mesures anteriors, poder fer comparacions i decidir quan és necessari recalibrar el sensor.

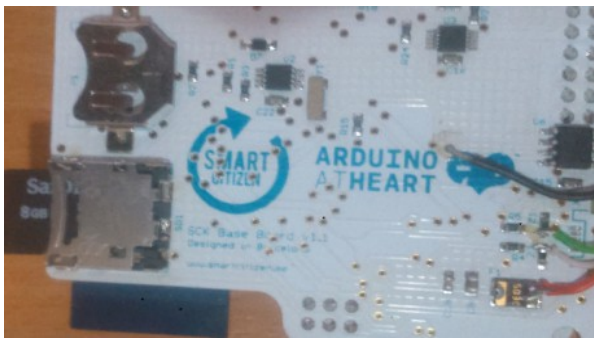

Fig. 33 Part posterior SCK, amb tarja microSD de 8 gB

Els passos previs abans de començar a programar la nostra SCK són:

- Formatejar la microSD amb un format FAT16 o FAT32. Això dependrà de la capacitat de la tarja. En el nostre cas serà FAT32. (Veure Annex 13.1).
- Descarregar *Arduino IDE*, a la seva pàgina oficial (Veure Annex 13.2): http://www.arduino.cc/en/Main/Software
- Conectar la SCK per USB a l'ordinador i assegurar-nos que en la consola de treball estigui seleccionada la nostra placa (LilyPad Arduino) i el port correcte (COM5). A la següent figura es mostra com asseverar-nos d'això.

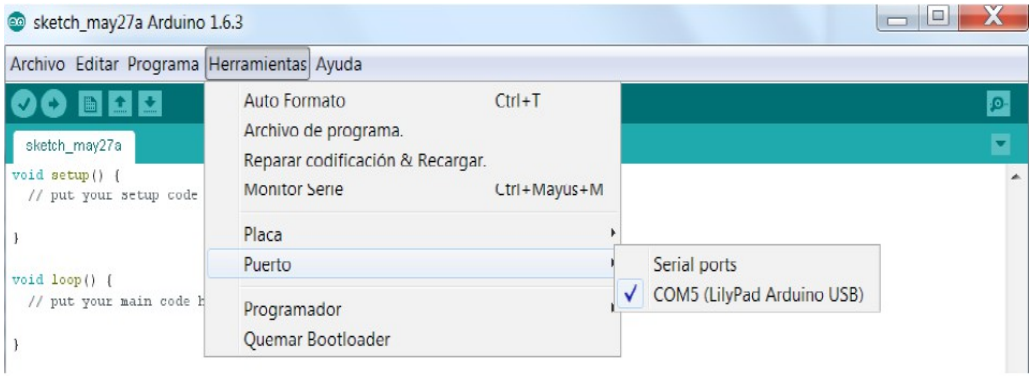

Fig. 34 Arduino IDE

En cas que la placa seleccionada sigui diferent a la connectada, pot haver-hi greus problemes. Un d'ells és que la placa deixi de funcionar indefinidament degut a que no reconeix el seu *firmware.* Un cop succeït això no és possible tornar a canviar la placa de detecció com veiem en la Fig.30, ja que per seguretat tot resta bloquejat i inoperatiu. Per tant abans de començar a compilar cap programa hem de comprovar quina placa tenim connectada. Per solucionar aquest problema, veure l'annex 13.3.

8.1 Programació II

Com hem explicat en l'apartat anterior, primer farem un codi per enregistrar les dades de referència a la SD. A continuació es mostra l'estructura d'aquest programa i s'explica el seu funcionament. S'ha programat de forma independent, per poder implementar-ho en un altre codi en cas necessari.
Es requereixen dues llibreries d'*Arduino* per a controlar la SD (SPI.h i SD.h).
La llibreria SD.h és dependent de la llibreria SPI.h i per tant han d'estar obligatòriament totes dues incloses. Al mateix temps SD.h està formada per dues classes:

- *SD class*
- *File class*

La primera s'encarrega de donar accés a la SD i de manipular els seus arxius o directoris. Conté les seves pròpies funcions que haurem de seleccionar segons les nostres necessitats.
La classe *File* conté mètodes per llegir o escriure i per verificar l'existència d'arxius en la SD.

El codi necessari per iniciar la *SD class* és:

Codi	Nota
#include <SPI.h> #include <SD.h>	Inclusió de les llibreries per a controlar el port SPI i el controlador SD.
File dades_ref;	Es defineix una variable del tipus File, que serà on s'emmagatzemaran les dades.
pinMode (10, OUTPUT); digitalWrite (10, HIGH);	Utilitza el pin 10 com a sortida CS.

If (!SD.begin(11) {
 	
// Codi error d'inicialització
}
 	
// Codi èxit d'inicialització
	La funció SD.begin() verifica si en el terminal 11 hi ha una microSD.

Fig. 35 Codi d'inicialització de la *SD class*

El codi emprat per a crear un arxiu dins del directori creat (dades_ref) i poder escriure:

Codi	Nota
dades_ref= SD.open('dades.txt', FILE_WRITE);	Obrim el fitxer dades_ref i creem l'arxiu dades.txt a dins.

If (dades_ref) {
 	
// Text que es vol gravar
 	
dades_ref.close():
 	
}
 	
// Error al obrir dades.txt
	Comprova si s'obre correctament el fitxer. En cas afirmatiu comença a escriure i quan acaba es tanca. Si no el pot obrir, retorna un error.

Fig. 36 Codi de creació i escriptura d'un arxiu

Finalment ens resta saber el codi necessari per a llegir l'arxiu creat.

Codi	Nota
dades_ref= SD.open('dades.txt');	Obrim el fitxer dades_ref

If (dades_ref) {
 	
Serial.println('dades.txt');
 	
while (dades_ref.available()) {
Serial.write(dades_ref.read()) ;
 	
}
	Comprova si el fitxer està disponible. En cas afirmatiu escriu dades a la SD i mostra el contingut del fitxer. Les dades guardades anteriorment apareixen com un historial en l'arxiu.

dades_ref.close(); }
 	
else {
 	
//Error al obrir dades.txt
 	
}
	Tanquem l'arxiu després de la seva lectura. I mostrem un error en cas que no es pugui obrir.

Fig. 37 Codi de lectura d'un arxiu

Ara compilem aquest programa abans d'executar-lo per veure que no hi hagi cap error d'escriptura o que no manca cap llibreria.

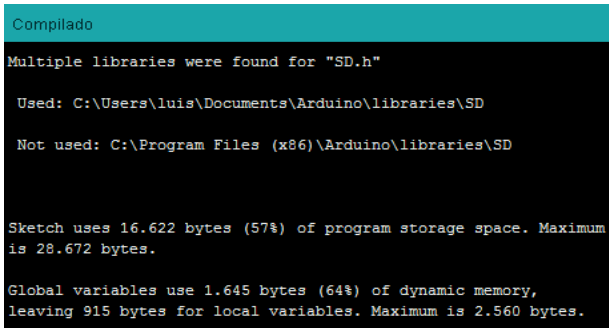

Fig. 38 SCK_1.1_SDcard compilat correctament

A la Fig. 31, tenim el resultat d'un programa sense errors de compilació. Ens dóna informació sobre la ubicació de les llibreries utilitzades i la memòria física i virtual que necessita. Quan compilem qualsevol programa és important fixar-nos en el valor de *'dynamic memory'* (veure Annex 13.3).

Ara és el moment de carregar aquest codi i veure per pantalla els resultats:

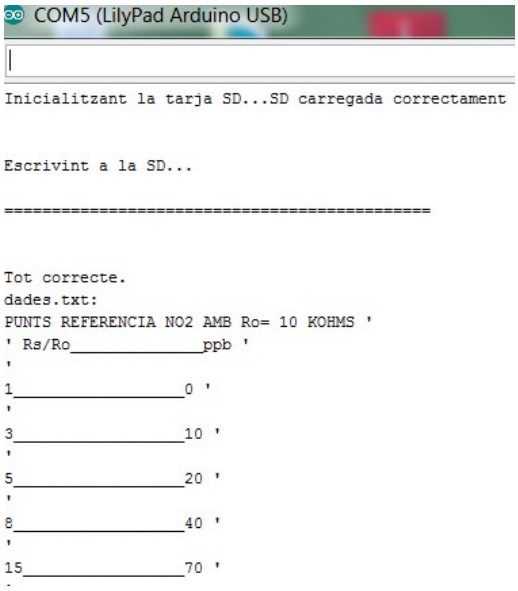

Fig. 39 Sortida de SCK_1.1_SDcard

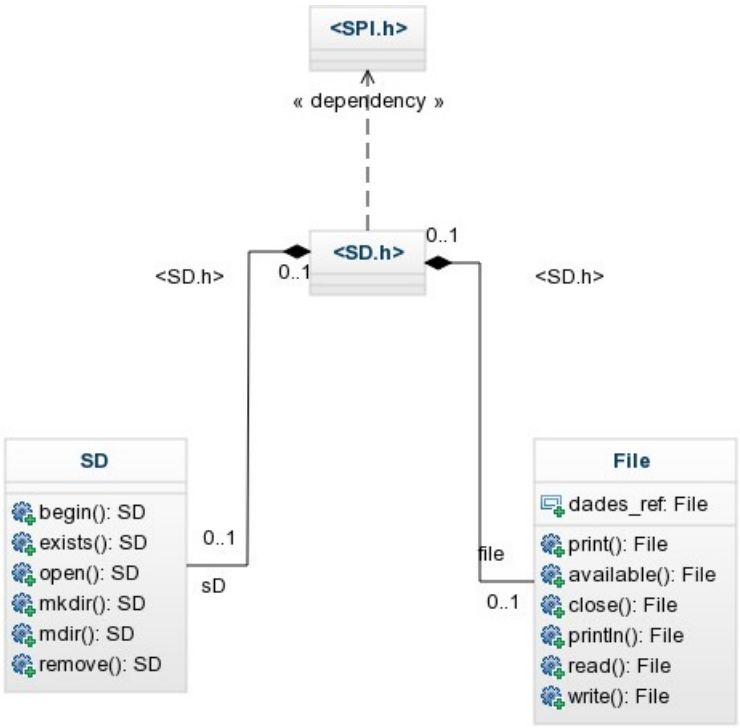

Fig. 40 Diagrama UML de SCK_1.1_SD

8.2 Programació III

Un cop tenim el codi per gravar i visualitzar dades necessitem aplicar l'algoritme LA. En aquesta ocasió hem d'introduir dades pel port *Serial.* Introduirem el valor de Rs/Ro de CO o de NO2 (Rs i Ro els coneixem) i després el programa calcularà els punts de la corba nova.

En aquest cas no requerirem cap llibreria ja que no cal en la comunicació *Serial. Serial* es pot considerar una classe pròpia de l'entorn *Arduino* amb les seves funcions. A continuació comentarem els punts més importants d'aquest codi

Definició de les variables i inici de la comunicació *Serial:*

Codi	Nota
float Byte0,Byte1,Byte2,Rs_Ro,difCO,a	Definim les variables com a decimals (float). El Byte0, Byte1 i Byte2 corresponen a les entrades que introduïm manualment (fins a 3 xifres). Rs_Ro conté el valor introduït de Rs/Ro manualment. A, és el valor de referència en un punt donat i difCO la diferència proporcional entre la mesura entrada i la referència.
Serial.begin(9600);	Inicia la comunicació a una velocitat de 9600 baudis.

Fig. 41 Codi inici comunicació Serial

Comprovació entrada de dades i aplicació de les condicions:

Codi	Nota

If (Serial.available()>0 ; {
 	
Byte0 = Serial.read();
Byte1 = Serial.read();
Byte2 = Serial.read();
 	
//Condicions
 	
}
	Comprovem si hi entren dades. En cas afirmatiu les guarda en les variables esmentades i comença a exposar les condicions de mesura i mostra els resultats per pantalla.

Fig. 42 Codi per introduir i guardar dades per Serial

S'ha decidit fer dos codis, un per trobar els punts de CO i un altre per NO2. La raó de fer-ho és que d'aquesta manera el codi és més curt i es poden detectar millor possibles errades. Són codis que treballen amb molestes condicions i es creu que és més còmode per entendre que fa cadascun.

Un cop executem qualsevol dels dos programes, haurem d'introduir una valor Rs/Ro pel port Serial. Aquest port només accepta 1 byte d'entrada, és a dir si es posa el número 123 només detecta el número 1. Per agafar les 3 xifres s'han de crear 3 entrades (Byte0, Byte1 i Byte2). Amb aquest canvi fem que ens detecti un 1 un 2 i un 3, però no un 123. La proposta de solució per aquest problema ha sigut (en l'exemple de 123) multiplicar la primera entrada per 100, sumar-li la segona multiplicada per 10 i finalment sumar-li també la tercera entrada multiplicada per 1. Aquesta solució és òptima pel cas de NO2 que empra valors Rs/Ro superiors a 1. En el cas de CO aquests valors són inferior a 1, i el port Serie no reconeix els decimals. Per solucionar-ho per introduir un valor decimal, per exemple 0.5, escriurem un 050 seguit i sense punt ni coma.

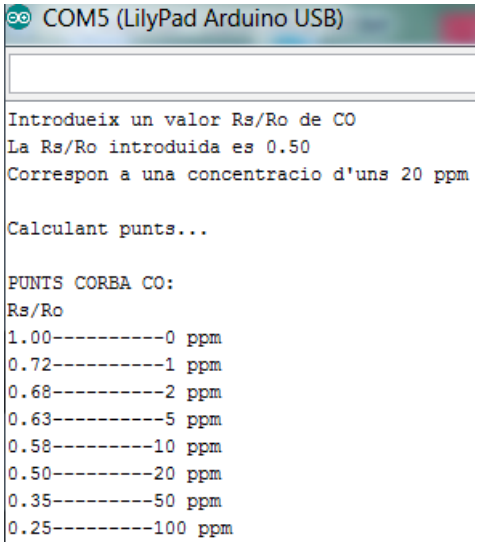

Fig. 43 Sortida de CO amb l'entrada d'exemple

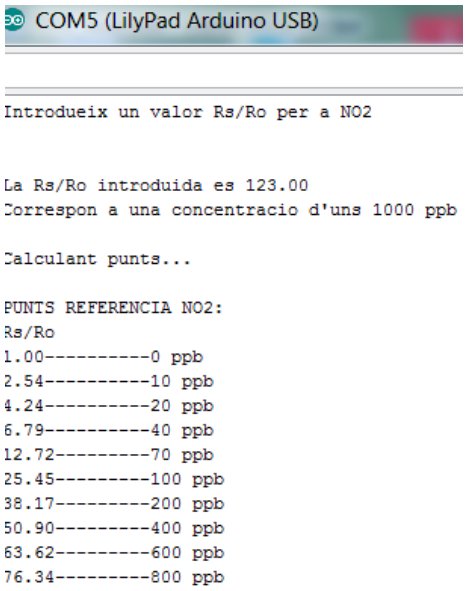

Fig. 44 Sortida de NO2 amb l'entrada d'exemple

A les figures 38 i 39 es veu que la sortida ens retorna la concentració en el punt introduït, això ho aconseguir amb condicions que comparen aquest valor amb el de referència i determina la concentració aproximada. Un cop la troba ja pot calcular la resta de punts de la corba.

Tot el codi explicat anteriorment el podem trobar a:
https://github.com/languerat/SCK_1.1_SDcard (Codi de la SD i dels gasos CO i NO2)
https://github.com/fablabbcn/Smart-Citizen-Kit/tree/master/sck_beta_v0_9 (Codi de l'apartat 9)

9. Escenari real de prova

Un cop tenim el programa desenvolupat s'han fet tests amb els valors simulats i funciona perfectament. Ara es vol fer una mesura real del nivell de CO en l'habitació en dos horaris diferents per comprovar si l'algoritme funciona també en aquest cas. S'ha escolli mesurar només CO perquè la concentració és més alta que NO2 i serà més senzill veure els resultats i comparar.

Recordem que la funció que ens permet llegir el nivell de CO (apartat 2.2.4) és:

```
SCKAmbient :: getMICS() {
// Inicia la fase pre-heater per augmentar la temperatura del sensor

heat (MICS_5525,32);  //En mA
// Llegeix els valors

RsCO = readMICS(MICS_5525);
// Emmagatzema aquests valors i amb un Serial.print(RsCO) els mostrem per pantalla.
}
```

Aquesta funció ens llegeix el valor Rs del sensor, per tant com que no sabem la Ro de la nostra placa, establim que Ro = 5 Kohms i ja tindrem el valor Rs/Ro. Hem escollit 5 valors de Rs (el valors donats estan en un bucle) i hem fet una mitjana. Ens dóna una Rs = 3590 Ohms.

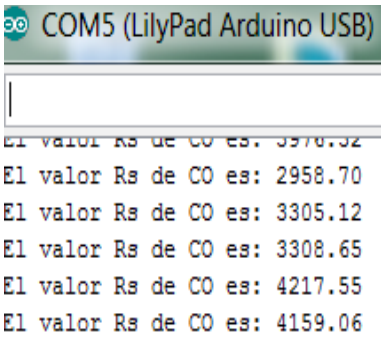

Fig. 45 Primera lectura Rs de CO en habitació

A la Fig.37 podem veure els 5 valors escollits per fer aquesta mitjana. Aquest resultat està en Ohms i la Ro la tenim en Kohms, per tant hem de fer un canvi d'unitats per trobar el nostre punt Rs/Ro. I obtenim:

$$Rs/Ro = \frac{(3590\,Ohms)}{5000\,Ohms} = 0{,}718$$

S'ha de dir, que aquesta mesura s'ha realitzat a les 13.00h amb una temperatura de 24,5º C (297,5 K) aproximadament. Ara introduïm aquest valor en el programa de càlcul de CO perquè ens retorni la concentració estimada i els seus punts de la corba:

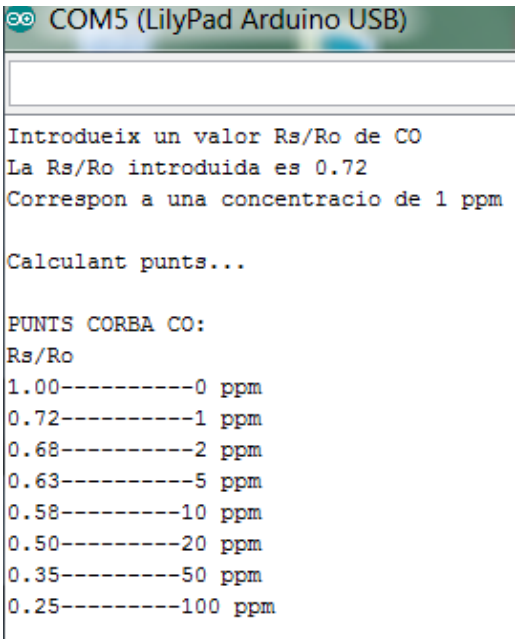

Fig. 46 Punts corba CO en habitació

La Fig. 38 ens dóna una concentració d'1 ppm aproximadament, un valor com s'esperava baix. Quan existeix un descens de la temperatura el valor de Rs augmenta (apartat 6.3), i un augment de Rs significa un descens en la lectura de concentració de CO. Per tant farem una segona mesura a una hora on hi hagi una temperatura inferior. En aquest cas s'ha realitzat un control a les 22.30h amb un temperatura de 21.3º C (294.3 K). Abans podem esbrinar fent una interpolació lineal amb el punt Rs trobat anteriorment quin ha de ser el valor aproximat de Rs a 21.3º C i podrem comprovar si l'algoritme ha calibrat correctament.

$$Rs = Rsa + (T - Ta)\frac{(Rsb - Rsa)}{Tb - Ta} \quad ;$$

- Rsa = Valor d'Rs en aire zero (punt d'inici), si sabem que en aquest punt Rs/Ro val 1, i Ro val 5, Rs en aquest punt val 5KOhms.
- Ta = Temperatura de referència, 20ºC (293 K).
- Rsb = Valor d'Rs obtingut en la primera mesura, 3.59KOhms.
- Tb = Temperatura de la primera mesura, 24,5º (297.5 K).

Substituïm i obtenim aquesta equació:
Rs = - 0.313T + 96,7

Ara substituïm T pel valor de la temperatura de la segona mostra (294.3 K) i obtenim un valor Rs aproximat de 4.58KOhms. Provem si el sensor ens mesura un valor similar:

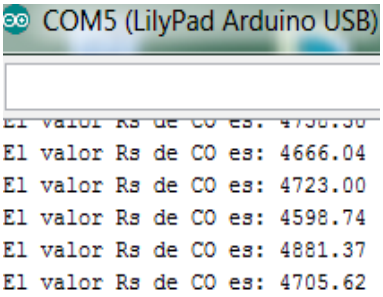

Fig. 47 Segona lectura Rs de CO en habitació

Tornem a agafar 5 valors per trobar un valor mitjà d'Rs (Fig.39) i obtenim Rs = 4.71 Kohms. És un resultat lleugerament superior a l'esperat però entra dins de la normalitat. El punt Rs/Ro és 0.94 i com podem observar és superior a l'anterior punt Rs/Ro (0.72). Si ho provem en el programa, ens retornarà una concentració d'1ppm ja que el programa no detecta valors més petits de 1, però podem assegurar que a la segona mesura tenim una concentració inferior a 1 ppm. L'experiment ha servit per comprovar que l'algoritme funciona i també per mostrar que la interpolació de punts es un bon mètode per veure els canvis que aplica la temperatura en les lectures.

10. Conclusions

En aquest apartat es comenten les conclusions finals obtingudes i si s'han assolit els objectius fixats al principi del projecte. També s'inclou un anàlisis amb possibles millores a nivell de *hardware* i una autoavaluació personal que descriu el punts crítics o que més problemes han donat durant el desenvolupament.

10. 1 Conclusió del projecte

Els objectius inicials del projecte es dividien en dos parts. La primera era:

- Estudiar i analitzar la documentació actual oferida en els repositoris d'Smart Citizen.
- Fer una recerca d'informació complementària per ampliar els nostres coneixements i trobar metodologies útils en el projecte.

Hem trobat en el repositori d'*Smart Citizen* informació sobre la placa SCK 1.1, sobretot a nivell de *hardware*. Els components que la formen, els tipus de sensors, les seves freqüències de treball, etc. La placa és una creació pròpia de l'empresa i només *Smart Citizen* pot proporcionar aquesta informació. Aquest coneixement ha suposat un important punt de partida.
El següent pas va ser consultar investigacions amb els mateixos sensors, (la majoria d'EEUU i UK). Amb aquestes fonts d'informació vam descobrir l'abast del projecte, la diversitat de mètodes de calibració i hem pogut aprendre com intentar solucionar el problema plantejat. A més hem afegit nova informació que resultarà rellevant en futures investigacions. Per tant els objectius d'aquesta primera part es consideren tots complerts.

La segona part es basava en dur a terme el desenvolupament:

- Desenvolupar un algoritme propi que pugui tornar tots els punts de calibració establerts d'una placa amb només dues dades.
- Aplicar aquest algoritme en un programa basat en la nostra plataforma de treball d'*Arduino* i afegir aquest resultat a la llibreria oficial d'SCK

En aquesta punt, bàsicament havíem de trobar el millor sistema de calibració possible pel nostre sensor. Després d'estudiar diversos mètodes, es va definir un que complia les nostres expectatives (LA). Aquest algoritme treballa amb dues dades (Rs i Ro) i per tant compleix el primer objectiu d'aquesta fase. L'últim punt era passar aquest algoritme a codi *Arduino*. Vam decidir amb alguns enginyers d'*Smart Citizen* que es volia un programa que treballés amb dades introduïdes manualment pel *Serial* i retornés els punts Rs/Ro amb la seva concentració de gas associada. Aquesta decisió va ser la correcta, ja que tot i poder fer lectures reals com s'ha vist en l'apartat 9, no es podria trobar una bona corba de calibració amb diversitat de punts.

10.2 Possibles millores

Hem establert que el sensor de gas MICS-4514 que utilitza la SCK s'ha de calibrar cada 15 dies. El problema és que passats uns mesos la nostra calibració no serveix, ja que és tanta la diferència entre el valor de referència i el mesurat que donaria moltes errades i no seria gens fiable. S'hauria d'enviar a un laboratori i tornar a calibrar bé. Per pal·liar aquest problema s'hauria d'invertir una mica de pressupost en trobar sensors més resistents al pas del temps i a les condicions ambientals o esperar noves versions.

Els programes de CO i NO2 retornen uns punts de corba. La forma ideal seria que retornes un gràfic, però no existeix cap funció a *Arduino* que ho permeti. En el futur s'hauria de buscar un programa que directament llegís els resultats i formés un gràfic.

En aquest projecte s'ha treballat sempre amb la SCK connectada a l'ordinador, però té la possibilitat d'oferir una comunicació sense fils amb una bateria que alimenta el seu mòdul WiFi RN131. El principal problema és el consum d'aquest mòdul que com hem explicat en l'apartat 2.2.5 té un consum energètic d'uns 210mA i limita molt els temps de vida de la SCK. Per tant una millora futura seria substituir-ho per un altre amb un consum inferior. Un exemple seria el mòdul *WizFi210* de l'empresa *Anatronic*.

10.3 Autoavaluació personal

A continuació es descriuen els problemes que han anat apareixent durant el desenvolupament d'aquest projecte:

- Retard en l'enviament de la placa de treball → L'enviament va tenir un retràs d'uns 2 mesos que va afectar al ritme de treball, ja que sense la placa no es podien realitzar proves i el treball només podia avançar a nivell teòric.
- Retard en rebre les dades de referència → Aquest dades van arribar passats 2 mesos de l'inici del projecte. Sense aquestes dades no es podia trobar cap algoritme, ni tan sols a nivell matemàtic.
- Problema amb la placa → La placa va quedar bloquejada e inoperativa. Es va perdre una setmana en portar-la al servei tècnic i fer un *reset* de fàbrica. Es desconeix exactament perquè va succeir.
- Dificultats amb l'entrada de dades pel *Serial* → *Arduino* només accepta un byte d'entrada per *Serial*, aquest ens ha portat a buscar una solució amb enginy per solucionar-ho.
- Dificultat en realitzar la prova real → Inicialment es volia realitzar una prova mesurant la concentració de gas en un tub d'escapament. Han sorgit problemes per activar i treballar amb el mòdul WiFi i tampoc es disponia d'un ordinador portàtil per connectar la SCK. Per aquesta raó s'ha decidit fer mesures en l'habitació.

Personalment ha estat un projecte motivador que ha resultat tot un repte per la desconeixença de conceptes referents a gasos, sensors i calibracions. El fet de treballar amb una empresa ens ha donat un *feedback* que ens ha ajudat a avançar tot i la manca de temps per dur a terme el desenvolupament. S'han ampliat els coneixements que es tenien sobre la matèria i s'ha aconseguit donar més facilitats per aconseguir apropar aquesta placa amb els seus sensors a la majoria d'usuaris.

11. Glosari

ADC: Bloc conversor d'analògic a digital.

Arduino: Plataforma de hardware lliure basada en un microcontrolador que té un entorn de desenvolupament propi que facilita la creació de projectes electrònics.

CO: Monòxid de carboni.

I2C: *Inter-Integrated Circuit,* és un bus de comunicacions en sèrie.

MAC: *Media Acces Control,* és una direcció física per identificar un aparell electrònic.

NO2: Diòxid de nitrogen.

Ppm: Parts per milió, s'utilitza per a mesurar concentracions de gasos.

Ppb: Parts per bilió, s'utilitza per a mesurar concentracions de gasos.

Ro: Resistència d'una placa amb aire zero.

Rs: Resistència d'una placa quan detecta un gas.

SCK: *Smart Citizen Kit*, la placa Arduino de treball.

Standby: Mode d'espera. En aquest mode s'estalvia energia.

SPI: *Serial Peripheral Interface,* és un estàndard de transmissió de dades.

Ufm: *Ultra Fast Mode,* Mode ultra ràpid de transmissió de dades amb el protocol I2C.

Vcc: *Voltage Continuous Current,* és el voltatge d'entrada d'un circuit amb corrent continua.

Wifi: *Wireless Fidelity,* sistema de comunicació sense fils.

12. Bibliografia

[1] Plataforma Smart Citizen :
https://smartcitizen.me/ , Març 2015.

[2] Software per realitzar diagrames de Gantt:
http://www.ganttproject.biz/ , Març,2015

[3] Introducció del projecte proposat i datasheets dels sensors:
https://docs.google.com/document/d/11VG_-jEGQcqvF3jQzM-kAP_gzdBm6W2YHF1EQiT38yc/edit# , Abril 2015.

[4] Documentació d'investigacions de calibració amb sensors similars:
https://drive.google.com/folderview?id=0B147uznMqCE4c2hpczZCQk9ENWs&usp=sharing , Abril 2015.

[5] Protocols de comunicació SPI i I2C:
http://es.slideshare.net/JonathanRuizdeGaribay/09bcomunicacin-i2-cyspi-9769471
Abril 2015.

[6] Efectes sobre l'organisme de CO. Estudi sobre els perill a l'exposició d'aquest gas:
http://www.murciasalud.es/pagina.php?id=180398&idsec=1573 , Abril 2015.

[7] Efectes sobre l'organisme de NO2. Estudi sobre els perill a l'exposició d'aquest gas:
http://www.murciasalud.es/pagina.php?id=180252&idsec=1573 , Abril 2015.

[8] Plataforma Arduino 'Arduino IDE':
http://www.arduino.cc/ , Abril 2015.

[9] Exemples de codi Arduino amb sensors de temperatura i humitat: http://www.ajpdsoft.com/modules.php?name=News&file=print&sid=572 , Maig 2015.

[10] Serial Arduino. Explicació del concepte Serial i del tipus de comunicació que comporta: http://www.arduino.cc/en/pmwiki.php?n=Reference/Serial , Maig 2015.

[11] Sensors amb entrades analògiques Arduino: http://diymakers.es/sensores-en-entradas-analogicas-de-arduino/ , Maig 2015.

[12] Interpolació lineal de punts. Explicació i fórmula d'aquest mètode: http://es.wikipedia.org/wiki/Interpolaci%C3%B3n , Juny 2015.

[13] Software per a realitzar UML: https://dashboard.genmymodel.com/ , Juny 2015.

[14] Consells per a la redacció de la memòria final i per a la estructura del TFG: http://es.slideshare.net/BibliotecaCampusGandiaUPV/cmo-redactar-la-bibliografa-el-resumen-y-las-palabras-clave-tfg-tfm-2014 , Juny 2015

[15] Model de classes UML. Tutorial sobre com visualitzar les relacions entre classes involucrades en un sistema: https://dashboard.genmymodel.com/ , Juny 2015.

13. Annexes

13.1 Formatejar tarja micro SD

Abans de començar a utilitzar la tarja de memòria l'hem de formatejar. Per fer-ho, necessitem un lector extern de targes connectat amb USB a l'ordinador. També pot ser que l'ordinador tingui integrat un lector de SD (el nostre cas) i per tant utilitzarem un adaptador de micro SD a SD, ja que les targes micro SD són molt petites i no es poden llegir directament en el PC.

Fig. 48 Adaptador micro SD-SD

La introduïm com mostra la Fig. 36 i a continuació la llegim amb l'ordinador. Busquem la unitat a que correspon la nostra tarja (en el nostre cas F:) fem un click amb el botó dret i escollim la opció formatejar.

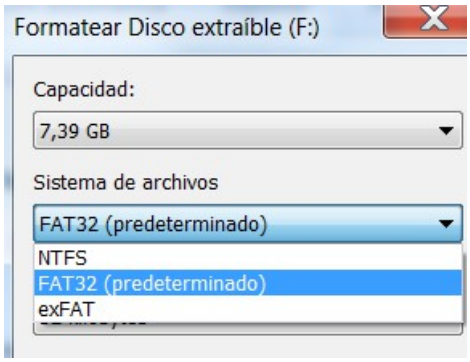

Fig. 49 Quadre formateig micro SD

En aquest quadre podem observar que només ens deixa fer un format amb tres sistemes diferents d'arxius (NTFS, FAT32 i exFAT). Per el nostre projecte ens interessa que estigui en format FAT, a ser possible FAT16. Però aquest format antic només és possible amb targes de fins a 2gB de capacitat i nosaltres utilitzem una de 8gB, per tant FAT32. Seleccionem aquesta opció i donem a acceptar. Un cop fet això totes les dades que hi havia quedaran esborrades i la tarja ja estarà llesta per començar a treballar.

13.2 Entorn Arduino

Arduino posseeix un entorn de programació propi que li permet editar, depurar, compilar i posteriorment pujar el programa o *sketch* a la placa emprada. A més permet la comunicació amb el port sèrie a nivell de lectura, escriptura i monitorització. El seu editor té una llista de llibreries incloses, tot i que podem crear les nostres pròpies o importar-les.
Un cop estigui descarregat e instal·lat en el nostre sistema, mostrem les passes necessàries per a realitzar una programació, compilació i posterior càrrega del codi a la placa:

- Obrir el programa Arduino IDE

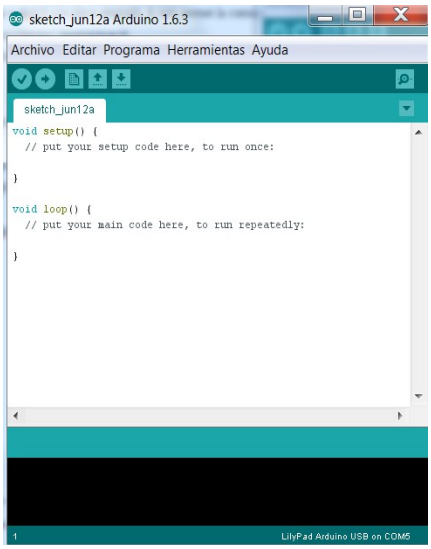

Fig. 50 Editor Arduino IDE

- Editar el codi tenint en compte el void.setup() i el void loop(). Dóna la opció de copiar i enganxar textos d'altres fonts.
- Un cop estigui editat es pressiona el botó de verificació:

- Si no retorna cap error de compilació, connectem la placa per USB i seleccionem el port corresponent en el menú 'Herramientas'.
- Carreguem el codi:

- En aquest punt s'estan enviant les dades del codi via *Serial* .Per comprovar el resultat per pantalla i veure si el programa treballa correctament pressionem el següent botó:

13.3 Solucions a problemes típics

Quan compilem un programa, obtenim informació sobre la memòria que necessita per executar-se. Aquesta informació és més important del que pot semblar. Quan s'edita un programa molt extens aquest ocupa un % de la memòria dinàmica , si aquest valor super el 80% podria haver-hi un desbordament de dades i això provocaria el bloqueig indefinit de la placa. Per tant sempre s'ha de vigilar no superar aquest valor.

Un altre problema típic de bloqueig és carregar un *sketch* amb una placa mal seleccionada, aquest fet provoca que *Arduino* sigui incapaç de reconèixer la llicència i per tant per seguretat resta bloquejat. Per solucionar qualsevol d'aquests dos problemes s'ha d'anar al servei tècnic *d'SmartCitizen* per fer un *reset* de fàbrica.

Està ubicat a: Hangar.org c/ Emilia Coranty 16, 08018, Barcelona.

Aquest procés el podria realitzar qualsevol usuari, sempre que disposés de dues plaques iguals fent un *flash* de la memòria. Tot i així no és recomanable ja que si es realitza incorrectament podrien restar bloquejades les dues plaques.
Per detectar si tenim la placa bloquejada per qualsevol dels dos motius mencionats, hem de fixar-nos en la llum dels leds. Si hi ha una llum taronja fixa, indica que no hi ha transmissió de dades. Una altre forma de detecció més evident , és que el nostre ordinador no reconeix l'USB de la SCK i per tant es com si no estigués connectada.

13.3 Diagrama de Gantt del desenvolupament del projectes

En aquest apartat s'adjunta un diagrama amb la distribució temporal de les tasques realitzades al llarg d'aquest projecte:

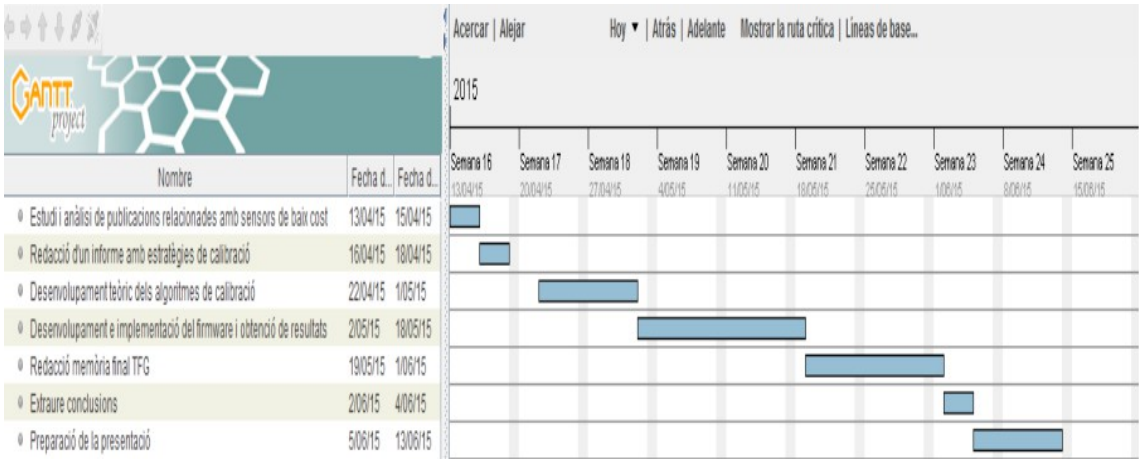

Fig. 51 Diagrama de Gantt del projecte