Seminario de Sensores (Sprint 1)

29 DE OCTUBRE DE 2025

EQUIPO 4 — PROYECTO ATMOS
Nerea Aguilar, Judit Espinoza, Alan Guevara, Santiago Fuenmayor y Alejandro Vázquez

Índice

Tabla de ilustraciones

1. Introducción

El objetivo de este trabajo es realizar la **calibración del sensor de ozono SPEC Sensors O_3**, obteniendo los parámetros de corrección necesarios (ganancia y desplazamiento) para ajustar las lecturas del sensor a los valores reales medidos con un sensor patrón.

Estas correcciones —**offset** y **ganancia**— se aplican posteriormente en el software del proyecto, dentro de la clase Medidor.h, para que las lecturas del sistema reflejen con precisión las concentraciones reales de ozono (ppm).

2. Datos experimentales y gráfica de calibración

Durante el proceso de calibración se tomaron varias medidas experimentales con el sensor de ozono conectado al módulo **ULPSM (Analog Sensor Developer Kit)** de SPEC Sensors.

Para cada medida se registraron:

- La concentración medida por nuestro sensor de ozono.
- Y la concentración real de ozono medida con un sensor patrón.

A partir de estos datos se obtuvo una tabla de correspondencia entre las concentraciones medidas por el sensor (C_sensor) y las concentraciones reales (C_real).

Con los valores recogidos, se construyó una gráfica de dispersión en Excel con los siguientes ejes:

- **Eje X:** concentración medida por el sensor (ppm)
- **Eje Y:** concentración real de ozono (ppm)

Sobre los puntos experimentales se trazó una **recta de ajuste lineal** que representa la relación entre ambas magnitudes:

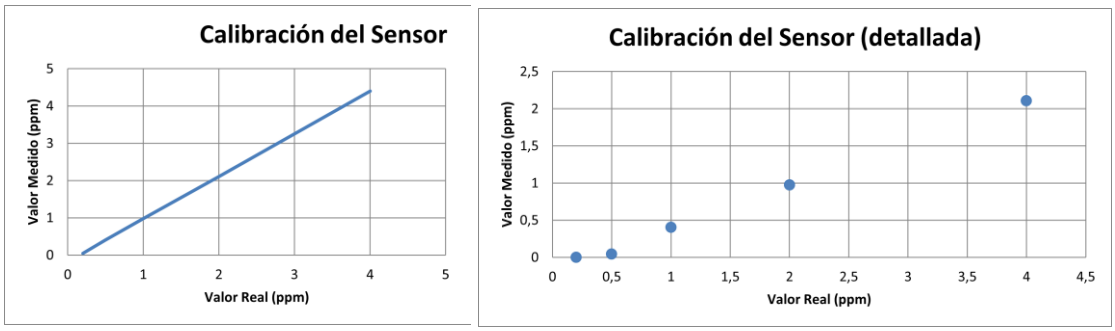

Ilustración 1: Recta calibración

La ecuación de dicha recta tiene la forma:

$$Creal = a \times Csensor + b$$

donde:

- a→ **pendiente o ganancia experimental**,
- b→ **intersección con el eje Y (offset)**.

3. Obtención de la pendiente y el offset en Excel

En un fichero Excel se calcularon los valores de a y b utilizando las funciones nativas de Excel, también se puede obtener matemáticamente con la fórmula de regresión lineal por mínimos cuadrados:

- **Pendiente (ganancia):** la fórmula utilizada fue **PENDIENTE(B2:B6;A2:A6)**, donde devuelve el coeficiente de inclinación de la recta que mejor se ajusta a los puntos experimentales.
  Este valor corresponde al factor de corrección de ganancia, que ajusta la escala de la medida del sensor.
- **Intersección con el eje (offset):** la fórmula utilizada fue **=INTERSECCION.EJE(B2:B6;A2:A6).** Donde devuelve el punto en el que la recta de ajuste corta el eje Y.
  Este valor representa el **desplazamiento** que hay que restar o sumar a las mediciones para que comiencen en cero real.

A partir de los datos experimentales se obtuvieron los siguientes resultados:

$$a = 1.143019508$$
$$b = -0.17505004$$

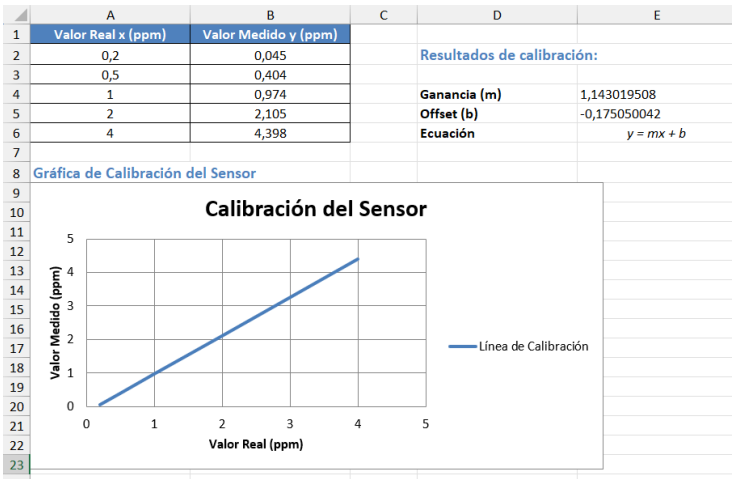

Ilustración 2: Excel

4. Desarrollo del proceso de corrección

Según el *Analog SDK User Manual* de SPEC Sensors (Analog-SDK-Users-Manual-v19), la concentración del gas se calcula mediante la expresión:

$$Cx = \frac{1}{M} * (Vgas - Vref)$$

donde:

- V_{gas} y V_{ref} son los voltajes medidos en las salidas correspondientes del sensor,
- M es el factor de conversión (V/ppm), calculado a partir de la sensibilidad y la ganancia TIA.

Para el sensor SPEC O_3 empleado en nuestro caso:

$$M = Sensibilidad \times Ganancia \times 10^{-6}$$

$$M = (-42.31) \times 499 \times 10^{-6} = -0.0211 \text{ V/ppm}$$

Posteriormente, a las medidas obtenidas se les aplica la **corrección experimental** derivada de la calibración:

$$Cx_{corregido} = \frac{Cx - b}{a}$$

5. Implementación de las correcciones en el código

El archivo Medidor.h implementa la clase Medidor, donde el método medirCO2() hasta ahora se encargaba de devolver un valor constante, con la conexión del sensor a la placa se encarga de leer los valores analógicos del sensor y calcular la concentración de ozono corregida.
En la sección de **variables privadas** se definen los parámetros eléctricos y los coeficientes de calibración, así como los pines:

```
private:

// ================================================
//     Variables Sensor de Ozono SPEC Sensors O3 (modo laboratorio)
//     Lectura con Vref, para recalibrar contra sensor patrón
// ================================================

const int pinVgas = A4;   // VGAS conectado al pin físico 28 (P0.28)
const int pinVref = A5;   // VREF conectado al pin físico 29 (P0.29)

// Parámetros del ADC y hardware
const float Vcc = 3.3;           // Voltaje de referencia ADC
const int resolucionADC = 4096;  // 12 bits -> 0 a 4095

// Parámetros de calibración base del sensor
const float sensibilidad = -42.31;  // nA/ppm (según etiqueta del sensor)
const float ganancianTIA = 499.0;    // kV/A (según manual)
const float M = sensibilidad * ganancianTIA * 1e-6;  // V/ppm

// Variables de corrección
float a = 1.143019508; // Factor de ganancia experimental (pendiente)
float b = --0.175050042;  // Offset experimental (intersección)
```

Ilustración 3: Variables privadas clase Medidor

El método principal de medida (**medirCO2**) —utilizado aquí para ozono— aplica la ecuación completa de cálculo y corrección:

```
int medirCO2() {
  float Vgas = leerVoltaje(pinVgas);
  float Vref = leerVoltaje(pinVref);
  float deltaV = Vgas - Vref;
  float Cx = deltaV / M;
  float Cx_corregido = (Cx - b) / a;

  // El sensor da valores negativos: usar valor absoluto ('fabs')
  int ppm = (int)(fabs(Cx_corregido) * 1000); // Escalado x1000, ya que beacon no admite decimales, luego lo dividimos por 1000.

  Serial.print("Vgas: "); Serial.print(Vgas, 4);
  Serial.print("  Vref: "); Serial.print(Vref, 4);
  Serial.print("  ΔV: "); Serial.print(deltaV, 6);
  Serial.print("  ppm: "); Serial.println(ppm);

return ppm;
} // ()
```

Ilustración 4: Función medirCO2()

A su vez, la fusión medirCO2(), llama a **leerVoltaje()** la cual convierte la lectura analógica de un pin (obtenida con analogRead) en su valor de voltaje real usando la tensión de referencia (Vcc) y la resolución del ADC:

```
// Función para leer voltaje desde el ADC
float leerVoltaje(int pin) {
  int codigo = analogRead(pin);
  float voltaje = (codigo * Vcc) / (resolucionADC - 1);
  return voltaje;
}
```

Ilustración 5: Función leerVoltaje()