TEC | Tecnológico de Costa Rica

Escuela de Física

Ingeniería Física

Instructivo de Laboratorio

Instrumentación I

30 de septiembre de 2025

Versión: 0.1

Realizado por: Juan J. Rojas

Índice general

Laboratorio 1

Medición de estímulos eléctricos

1.1. Objetivos

Al finalizar este laboratorio el estudiante estará en capacidad de:

- Configurar un sensor digital para lograr una medición eléctrica con la mayor resolución posible.
- Calcular el error asociado a la medición utilizando una multimetro digital de alta precisión.

1.2. Materiales y equipo

A suministrar por la Escuela:

- 1 ARDUINO UNO R4 MINIMA
- 1 sensor digital de voltaje, corriente y potencia (INA219)
- 1 mini breadboard
- Cables de interconexión (jumpers) macho-macho

1.3. Procedimiento

a. Realice las conexiones del circuito tal y como se indica en la Figura **??**.

b. Abra el programa Arduino IDE

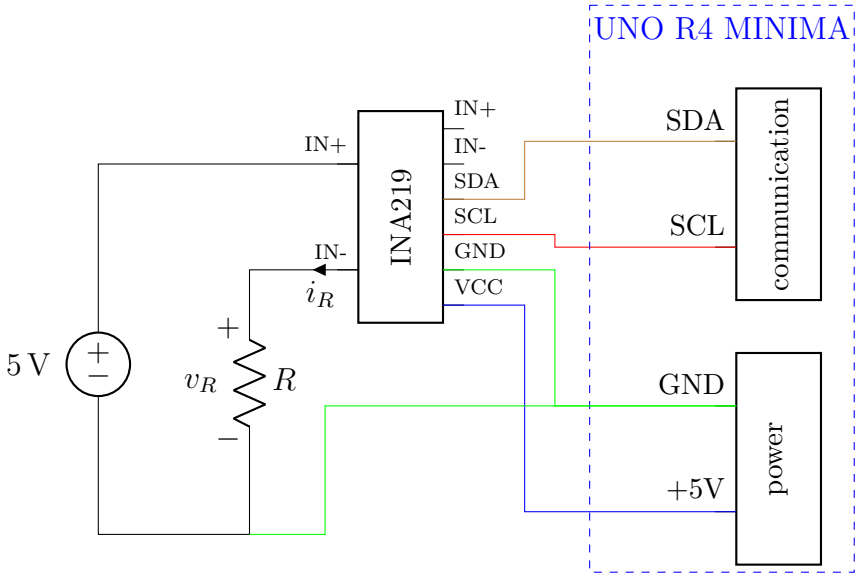

Figura 1.1: Conexión de sensor de corriente y voltaje INA219

c. Incluya el siguiente código en el Arduino IDE

```
#include <Wire.h>
#include <INA219.h>
INA219 ina219 (0x40);

//Instrucciones: Una vez subido, correr el monitor serial y presionar Enter para
    iniciar la medición, en caso de que no inicie automáticamente.
// Usar libreria INA219 (programada por Rob Tillart), de la lista de librerías.

void setup() {
  Serial.begin(115200);
  Wire.begin();
  delay(2000);
  if (!ina219.begin()) {
    Serial.println("INA219 not found");
    while (1){ delay(10); }}
  Serial.println("INA219 initialized");
  //Here we can set max current shunt value to calibrate the Calibration
   Register (8.5.1)
  ina219.setMaxCurrentShunt(1, 0.1);    //  adjust if needed
  Serial.print("Shunt value:\t");
  Serial.println(ina219.getShunt(), 4);
  Serial.print("Max current:\t");
  Serial.println(ina219.getMaxCurrent(), 4);
  //Set gain based on desired range
  ina219.setGain(1);
  Serial.print("Gain:\t");
  Serial.println(ina219.getGain());
  //Set bus voltage range
  Serial.print("Bus voltage range:\t");
  ina219.setBusVoltageRange(16);
  Serial.println(ina219.getBusVoltageRange());

  Serial.print("Calibrated?:\t");
```

```
  Serial.println(ina219.isCalibrated());
  Serial.print("CLSB:\t");
  Serial.println(ina219.getCurrentLSB());
  Serial.println("press any key");
  while (!Serial.available()) {
  }
}

float curre = 0;
float volta = 0;
float power = 0;
float shunt = 0;
int counter = 0;
// Here we set the oversampling
int samples = 100;
int delayTime = 1000/samples;

void loop() {
  counter++;
  curre += ina219.getCurrent_mA();
  volta += ina219.getBusVoltage();
  power += ina219.getPower_mW();
  shunt += ina219.getShuntVoltage_mV();
  if (counter >= samples) {
    Serial.print(curre/samples);
    Serial.print(",");
    Serial.print(volta/samples);
    Serial.print(",");
    Serial.print(power/samples);
    Serial.print(",");
    Serial.println(shunt/samples);
    counter = 0;
    curre = 0;
    volta = 0;
    power = 0;
    shunt = 0;
  }
  delay(delayTime);
}
```

d. Tomando en cuenta el valor de la resistencia que se le asignó, calcule el valor de la corriete esperada en el circuito utilizando la Ley de Ohm.

e. Investigue en el datasheet que significa el parametro PGA y como se relaciona con el rango del sensor. Ver 8.5.1

f. Configure el sensor para que el rango de medición sea el más adecuado para la resistencia que se le asignó, suba el código modificado al Arduino.

g. Realice dos mediciones de 10s de duración con una frecuencia de 1Hz al mismo tiempo que mide la corriente con un multímetro digital de alta precisión.

h. Calibre el sensor utilizando un multímetro digital de alta precisión y cambie nuevamente el código para que el valor medido por el sensor sea lo más cercano posible al valor medido por el multímetro. Ver 8.5.2.1. Suba el código modificado al Arduino.

i. Realice dos mediciones de 10s de duración con una frecuencia de 1Hz al mismo tiempo que mide la corriente con un multímetro digital de alta precisión.

j. Modifique el código para que tener oversampling de 100 muestras por segundo y suba el código modificado al Arduino.

k. Realice dos mediciones de 10s de duración con una frecuencia de 100Hz al mismo tiempo que mide la corriente con un multímetro digital de alta precisión.

l. Para cada ser de medicines calcule lo siguiente e incluya en una tabla:

- El valor promedio, $\overline{i_R}$
- La desviación estándar, σ
- El valor de la incertidumbre estándar, $\sigma_x = \sigma/\sqrt{n}$
- La exactitud
- La precisión
- La repetibilidad

m. Tome en cuenta lo siguiente:

- La exactitud se calculará como el mayor error relativo obtenido en cualquier punto de los datos en todas las corridas de medición.
- La precisión se calculará como la mayor desviación estándar entre todas las corridas de medición.
- La repetibilidad se calculará de la siguiente manera:

$$\text{repetibilidad} = \sqrt{\frac{\sum_{i=1}^{n}(\text{error absoluto})^2}{n}}$$

1.4. Preguntas para el analisis de resultados

- ¿Cómo afecta el valor de la resistencia al rango de medición del sensor?
- ¿Como afecta el valor de PGA al rango de medición del sensor?
- ¿En que consiste el oversampling y cómo afecta la resolución de la medición?
- ¿Que es lo que más afecta la precisión de la medición?
- ¿Qué es lo que más afecta la exactitud de la medición?

Laboratorio 2

Mediciones de temperatura

2.1. Objetivos

Al finalizar este laboratorio el estudiante estará en capacidad de:

- Utilizar un microcontrolador para realizar mediciones de temperatura.
- Realizar un algoritmo básico para obtener mediciones e imprimirlas en la terminal.

2.2. Materiales y equipo

A suministrar por la Escuela:

- 1 Arduino UNO R4 MINIMA
- 1 termistor de 10 kΩ NTC marca TDK
- 1 breadboard
- Cables de conexión macho-macho

2.3. Procedimiento

a. Arme el circuito tal como se muestra en la Figura 2.1.

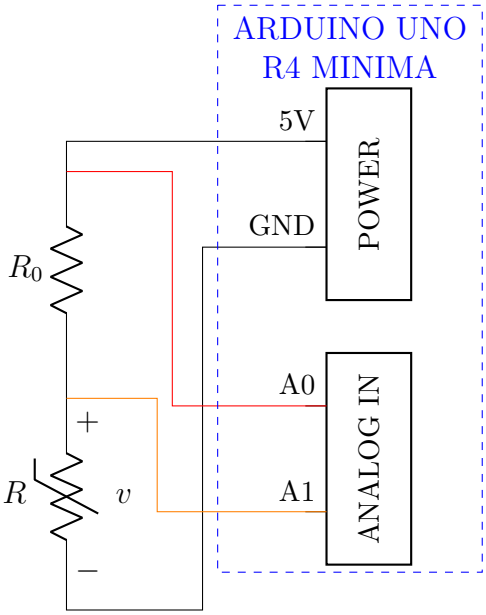

Figura 2.1: Conexión de circuito para el Laboratorio 3

b. Usando la formula del divisor de voltaje y simplificando se llega a que:

$$\frac{R_0}{R} = \frac{5}{v} - 1$$

c. El valor de la temperatura se puede aproximar usando la siguiente ecuación:

$$\frac{1}{T} = \frac{1}{T_R} + \frac{1}{B_{25/100}} \ln\left(\frac{R}{R_R}\right)$$

donde T es la temperatura del termistor en K, T_R es la temperatura de referencia en K y R_R es la resistencia de referencia en Ω

d. Según la hoja de datos del termistor el valor de $B_{25/100}$ es igual a 4300.

e. El siguiente sketch de Arduino IDE implementa las ecuaciones anteriores en dos funciones, una que calcula la resistencia (R) y otra que calcula la temperatura usando el valor de $B_{25/100}$ (T_1).

```
float V = 0;
float Vin = 0;
float vR = 0;
float R = 0;
float T1 = 0;
float T2 = 0;
int counter = 0;
int samples = 100;
int delayTime = 1000/samples;

#define vR0 4587.0
#define vBETA 4300
#define vTR 298.15
#define vRR 10000
```

```
#define vZero 273.15

float calc_R (float V, float Vin, float R0) {
  float R = (Vin / V) - 1;
  R = R0 / R;
  return R;
}

float calc_T1 (float R, int RR, float TR, float Zero, int BETA) {
  float T1 = BETA + TR * log(R/RR);
  T1 = (BETA * TR) / T1;
  T1 = T1 - Zero;
  return T1;
}

void setup() {
  // initialize serial communication at 9600 bits per second
  Serial.begin(9600) ;

}

void loop() {
  counter++;
  V = (analogRead(A1) / 1023.0) * 5;
  Vin = (analogRead(A0) / 1023.0) * 5;
  vR = calc_R(V, Vin, vR0);
  R = R + vR;
  T1 += calc_T1(vR, vRR, vTR, vZero, vBETA);
  if (counter >= samples) {
    Serial.print(R/samples);
    Serial.print(",");
    Serial.print(T1/samples);
    Serial.print(",");
    Serial.println(T2/samples);
    counter = 0;
    R = 0;
    T1 = 0;
    T2 = 0;
  }
  delay(delayTime);
}
```

f. Suponga que su sensor será utilizado para medir la temperatura de la piel (20 °C a 40 °C), para reducir el error en la medición utilice lo mostrado en la sección 2.3 de este documento para determinar $C0$, $C1$ y $C3$.

g. Realice una función en el mismo sketch en Arduino IDE que permita calcular el valor de T_2 a partir de los valores de C_0, C_1, C_3 y R usando la ecuación de Steinhart-Hart

h. Imprima en la terminal los valores de R, T_1 y T_2 y léalos usando un script de Python. Realice tres mediciones de 20 segundos cada una en las que toque el sensor para provocar cambios en la temperatura, deje enfiar entre cada medición.

i. Usando el código del sensor (B57164K0103) genere una tabla de valores de resistencia usando este software, use $\Delta R/R$ de 5 % y 1 °C de resolución, exporte los datos.

j. Utilizando Python y los datos obtenidos en las tres mediciones calcule T_3 para cada valor de R usando interpolación lineal entre los puntos obtenidos por medio del software

k. Para cada una de las mediciones realice una gráfica que muestre T_1, T_2 y T_3

2.4. Análisis

- ¿Cual le parece que es el mejor método de aproximación? ¿Porqué?
- ¿Que tan importante es la precisión de la resistencia R_0?
- ¿Que tan importante es la precisión de la referencia del ADC?