UNIVERSIDAD DE LOS LLANOS
Facultad de Ciencias básicas e ingenierías
Departamento de Matemáticas y Física

INFORME
INSTRUMENTACIÓN
ELECTRÓNICA I

PROYECTO MULTÍMETRO

W. Avila Ducon [1] ,D. Gabriel Cruz [2], S. Pedroza Noriega [3],D. Gil López [4]

1. Cod: 161004502, Ing. Electrónica
2. Cod: 161004509, Ing. Electrónica
3. Cod: 161004530, Ing. Electrónica
4. Cod: 161004514, Ing. Electrónica

Facultad de Ciencias Básicas e Ingenierías.
Programa

1. Marco Teórico:

Un multímetro es un dispositivo portátil de naturaleza eléctrica empleado para la medición de magnitudes eléctricas activas, tales como corrientes y potenciales (tensiones), así como magnitudes pasivas que incluyen resistencias, capacidades, entre otras. Este instrumento exhibe la capacidad de medir directamente dichas magnitudes eléctricas en diversos rangos de medida, tanto en corriente continua como alterna. Los multímetros se presentan en formatos analógicos y digitales, compartiendo una funcionalidad común con ligeras variaciones.

Figura 1. Multímetro digital

La versatilidad del multímetro radica en su dependencia de un galvanómetro altamente sensible que se emplea en todas las determinaciones. Además del galvanómetro, este dispositivo consta de elementos adicionales, como una escala múltiple por la cual se desplaza una única aguja, y un conmutador que posibilita la alteración de la función del multímetro, permitiéndole actuar como medidor en todas sus versiones y márgenes de medida. La combinación de estos componentes confiere al multímetro una amplia aplicabilidad en el ámbito de las mediciones eléctricas.

Arduino es una plataforma de creación de electrónica de código abierto, fundamentada en hardware y software libre, caracterizada por su flexibilidad y facilidad de uso, dirigida a creadores y desarrolladores. Esta plataforma posibilita la creación de diversos tipos de microordenadores de una sola placa, otorgando a la comunidad de creadores la capacidad de aplicarlos de diversas maneras. Se define como una placa electrónica de hardware libre que emplea un microcontrolador reprogramable con una serie de pines que facilitan la conexión entre el controlador y distintos sensores. En esencia, actúa como el "cerebro" de un circuito o maquinaria.

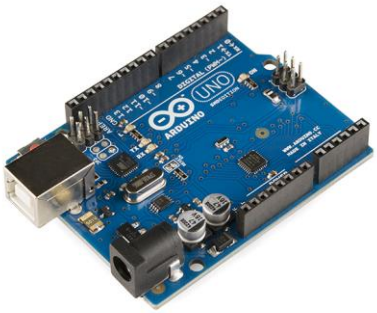


Figura 2. Tarjeta Arduino uno

La versatilidad de Arduino se manifiesta en su capacidad para ser programada con el propósito de llevar a cabo una amplia variedad de tareas, desde el control de luces y motores hasta la creación de robots y dispositivos de Internet de las cosas (IoT). Esta plataforma goza de una gran popularidad

entre los entusiastas de la electrónica y los fabricantes (makers), dado que su interfaz es accesible y permite la programación mediante diversos lenguajes de programación.

Una pantalla OLED, acrónimo de "Organic Light-Emitting Diode" (Diodo Orgánico Emisor de Luz, en inglés), constituye un tipo de pantalla que emplea diodos orgánicos emisores de luz para generar imágenes. A diferencia de las pantallas LCD, las OLED no dependen de una fuente de luz de fondo para iluminar los píxeles, ya que cada píxel emite su propia luz. Esta característica singular confiere a las pantallas OLED la capacidad de ofrecer negros más intensos y colores más vibrantes en comparación con las pantallas LCD convencionales.

Figura 3. Pantalla OLED de resolución 128x64

Estas pantallas se encuentran presentes en una amplia variedad de dispositivos electrónicos, tales como teléfonos inteligentes, televisores, relojes inteligentes y otros dispositivos portátiles. Además de sus excelentes propiedades visuales, las pantallas OLED destacan por su delgadez y flexibilidad, características que las vuelven idóneas para aplicaciones en las que se requiere una pantalla delgada y adaptable a diferentes formas y tamaños. Este avance tecnológico ha contribuido significativamente a la evolución y mejora en la calidad visual de numerosos dispositivos electrónicos.

2. Objetivo:

Diseñar un multímetro principalmente con un Arduino y aplicando criterios de diseño y medición, obtener un instrumento multiuso capaz de garantizar datos confiables y exactos.

Realizar 5 tipos de mediciones los cuales son frecuencia, voltaje, corrientes, capacitancia y resistencia.

3. Materiales:

- Multímetro
- Capacitores
- Potenciometros
- Trimmer
- Botones
- Resistencias
- Arduino Uno
- Pantalla Oled
- Amplificador Lm358
- Jumpers
- Caimanes
- Protoboard

4. Procedimiento:

La tarjeta usada para realizar el diseño del multímetro fue en el Arduino Uno, en el lenguaje y arquitectura de Arduino, siendo este un lenguaje de tipo C.

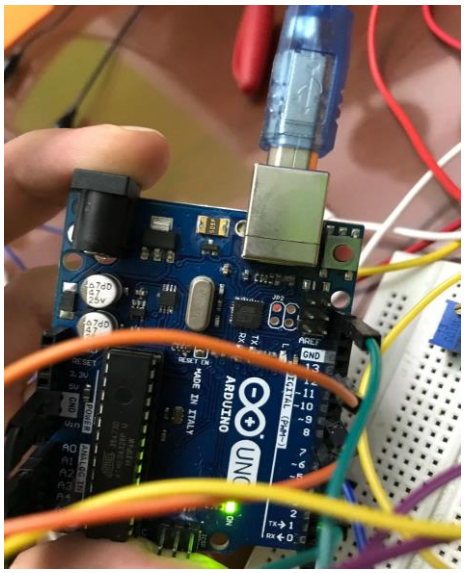


Figura 4: Tarjeta Arduino Uno

Para visualizar las lecturas se optó por usar una pantalla Oled el cual ocupa muy poco espacio en el protoboard.

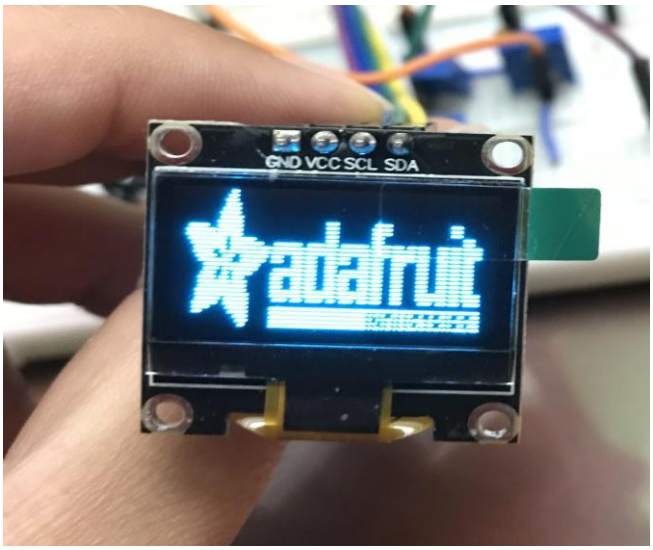


Figura 5: Pantalla Oled del multímetro.

Código de desarrollo:

```
//#include <LiquidCrystal.h>

#include <Wire.h>
#include <SPI.h>
#include <Adafruit_SSD1306.h>

#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);
//const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2, Modo_Voltaje= 6;
const int Modo_Voltaje=6;
//LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

// Definir constantes
#define ANCHO_PANTALLA 128 // ancho pantalla OLED
#define ALTO_PANTALLA 64 // alto pantalla OLED
#define analogpin A1
#define VA A2
#define VB A3
#define chargepin 13
#define dischargepin 10
#define resistorValue 10000.0F
```

```
int VI = 5;
unsigned long startTime;
unsigned long elapsedTime;
float microFarads;
float nanoFarads;

float R1_Volt=1000000;
float R2_Volt=100000;
float R=1000;
float corriente=0.0;
int Voltaje_Active;
int contador = 0;

const int pulsePin = 8; // Input signal connected to Pin 8 of Arduino


int pulseHigh; // Integer variable to capture High time of the incoming pulse
int pulseLow; // Integer variable to capture Low time of the incoming pulse
float pulseTotal; // Float variable to capture Total time of the incoming pulse
float frequency; // Calculated Frequency
```

```
void setup() {
  // put your setup code here, to run once:
  //lcd.begin(16, 2);
  //lcd.print("Bienvenido");
  Serial.begin(19200);
  display.setCursor(0,2);
  display.print("Bienvenido");

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  //display.clearDisplay();
  //display.display();
  pinMode (Modo_Voltaje, INPUT);
  pinMode(chargepin, OUTPUT);
  digitalWrite(chargepin, LOW);
  //lcd.init();
  //lcd.backlight();
  pinMode(pulsePin,INPUT);
  display.display();
  delay(1000);

}
```

```
void loop() {
  // put your main code here, to run repeatedly:
  float Voltaje=(analogRead(0)*5.0)/1023.0;
  float Divisor_Volt=(Voltaje/(R2_Volt/(R1_Volt+R2_Volt))-0.3);
  Voltaje_Active = digitalRead(Modo_Voltaje);

  if (Voltaje_Active == 1 ){
      contador++;
      if(contador==6){
        contador = 1;
      }
  }

  if (contador == 1){

      display.setTextSize(1);
      display.clearDisplay();
      display.setTextColor(WHITE, BLACK);
      display.setCursor(0,1);
      display.print("Voltaje");
      //lcd.begin(16, 2);
      //lcd.print("Voltaje");
      if (Divisor_Volt<=0.10){
        display.setCursor(1, 10);
        display.print("0.0");
```

```
      if (Divisor_Volt<=0.10){
        display.setCursor(1, 10);
        display.print("0.0");
        Serial.print(0.0);
        Serial.print(" Voltios ");
        Serial.println();
      }
      else{
        display.setCursor(1,10);
        display.print(Divisor_Volt);
        display.print("V");
        Serial.print(Divisor_Volt);
        Serial.print(" Voltios ");
        Serial.println();
      }
  }
  if (contador == 2){
      display.setTextSize(1);
      display.clearDisplay();
      display.setTextColor(WHITE, BLACK);
      display.setCursor(0,1);
      display.print("Resistencia");
      float avgVI=0;
      float avgVA=0;
      float avgVB=0;
```

```
if (contador == 2){
    display.setTextSize(1);
    display.clearDisplay();
    display.setTextColor(WHITE, BLACK);
    display.setCursor(0,1);
    display.print("Resistencia");
    float avgVI=0;
    float avgVA=0;
    float avgVB=0;
    getAvgVoltajes(avgVI,avgVA,avgVB);
    Serial.print("VI: ");
    Serial.print(avgVI);
    Serial.print(" | VA: ");
    Serial.print(avgVA);
    Serial.print(" | VB: ");
    Serial.print(avgVB);
    Serial.println();

    float resistencia=getResistenciaValue(avgVI,avgVA,avgVB);
    display.setCursor(1,10);
    display.print("R: ");
    display.print(resistencia);
    display.print(" Ohm");
}
```

```
if(contador==3){
  display.clearDisplay();
  digitalWrite(chargepin, HIGH);
  startTime = micros();
  while(analogRead(analogpin)<648){
  }

  elapsedTime= micros() - startTime;
  microFarads=((float)elapsedTime / resistorValue);

  if(microFarads > 1){

    display.setTextSize(1);
    //display.clearDisplay();
    display.setTextColor(WHITE, BLACK);
    display.setCursor(0,1);
    display.print("Capacitancia: ");
    display.setCursor(1,10);
    display.print(microFarads);
    //display.setCursor(14,1);
    display.print("uF");
    delay(500);
    }
```

```
  else{
    nanoFarads = microFarads * 1000.0;
    display.setTextSize(1);
    //display.clearDisplay();
    display.setTextColor(WHITE, BLACK);
    display.setCursor(0,1);
    display.print("Capacitancia: ");
    display.setCursor(1,10);
    display.print(nanoFarads);
    //display.setCursor(14,1);
    display.print("nF");
    delay(500);
  }

  digitalWrite(chargepin, LOW);
  pinMode(dischargepin, OUTPUT);
  digitalWrite(dischargepin, LOW);
  while(analogRead(analogpin)>0){
    }

  pinMode(dischargepin, INPUT);

  //display.setCursor(1,10);

}
```

Medidor de voltaje

En el modo de voltaje, se implementó un circuito divisor de voltaje mediante un par de resistencias (1MΩ y 100kΩ). La conexión con la tarjeta Arduino se llevó a cabo de la siguiente manera: la primera resistencia (de 100kΩ) se conectó entre la tierra y el pin analógico designado, mientras que la segunda resistencia (de 1MΩ), encargada de recibir el voltaje de la fuente, se conectó entre la fuente y el mismo pin analógico.

Este circuito resistivo permite medir hasta 50V en corriente continua. Sin embargo, dado que Arduino solo puede recibir hasta 5V, se emplea el divisor de voltaje para adaptar la señal. En la programación de la tarjeta Arduino, se crearon dos variables flotantes con los valores respectivos de las resistencias utilizadas.

Para que la tarjeta pueda interpretar el valor de la fuente, ya que no puede leer el voltaje directamente sino que proporciona valores analógicos ADC, se realiza un ajuste multiplicando el voltaje máximo (5V) y dividiéndolo entre el valor máximo que el microcontrolador entiende (1023). Este resultado se somete a la fórmula del divisor de voltaje con los valores resistivos mencionados anteriormente.

Finalmente, el valor calculado se envía a la pantalla OLED para su visualización.

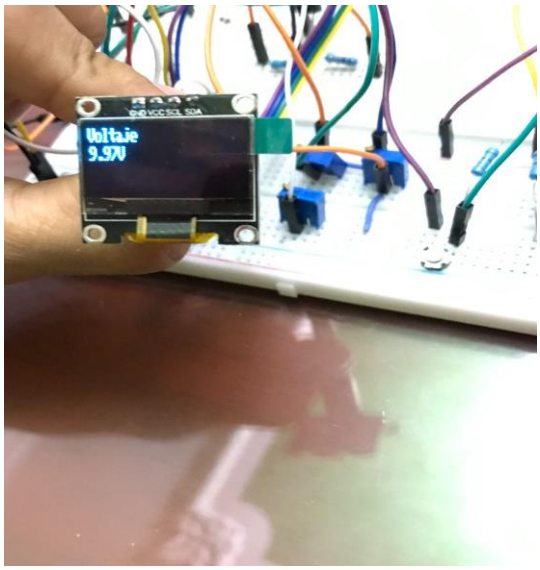


Figura 6: Evidencias fotográficas del multímetro en registrando voltaje.

Como se observó en la figura 6, Se midió el voltaje de una batería de 9v. Este valor fluctúa entre 9.97v a 10.13v.

Medidor de resistencias:

Para este modo, se realizó la implementación de un puente de wheatstone. con tres resistencias tipo trimmer de 10 kilohmios y una resistencia Rx, el cual va a ser la resistencia que se pondrá en la posición de Rx. Se realiza el respectivo analisis de un puente en equilibrio y se toman los voltaje de los nodos Va y Vb, y se tomo como voltaje de entrada los 5v del arduino.

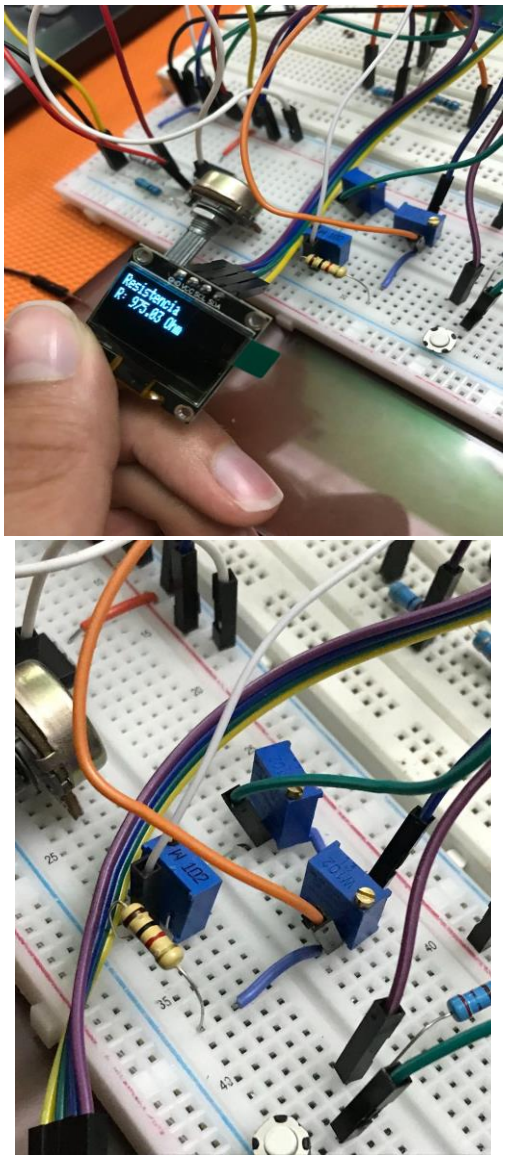


Figura 7: Evidencias fotográficas del puente de Wheatone y la resistencia registrada por el multímetro.

Se usó de ejemplo una resistencia de 1 kilo ohmio y se registró una resistencia de 975 ohmios

Medidor de Capacitancia:

Para llevar a cabo la medición de capacitancia, se emplea un enfoque basado en el método de carga y descarga, utilizando dos resistencias. Este método implica medir el tiempo que un condensador tarda en cargarse a través de un resistor conocido. Posteriormente, el código utiliza esta información para realizar una estimación precisa de la capacitancia del condensador.

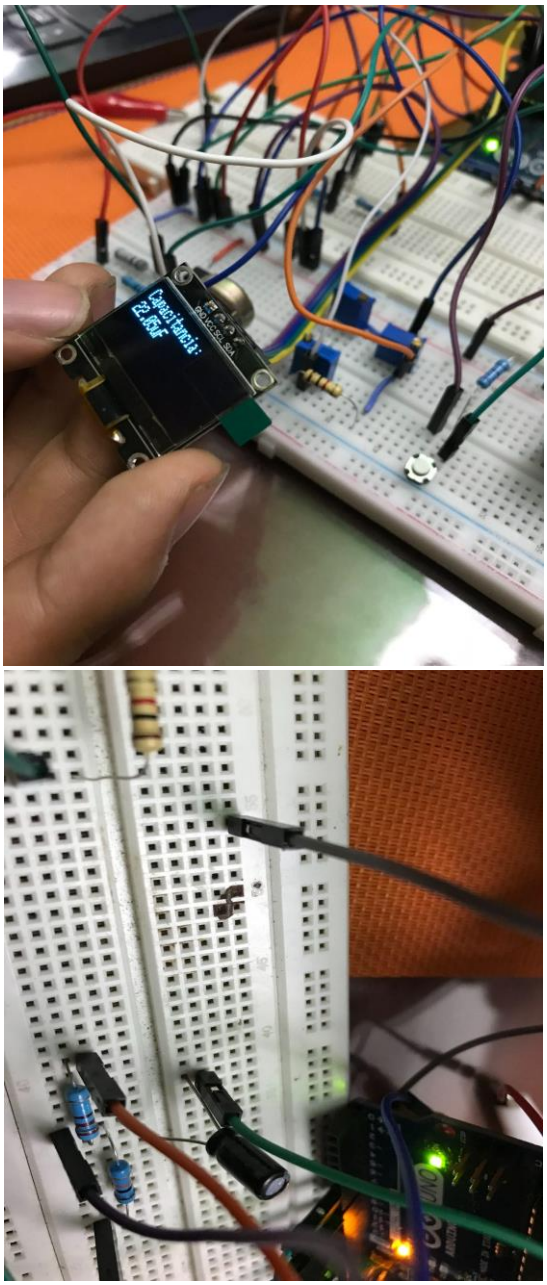


Figura 8: Evidencias fotográficas del método de carga y descarga.

Medidor de Corriente:

Para realizar esta medición se uso un amplificador LM358

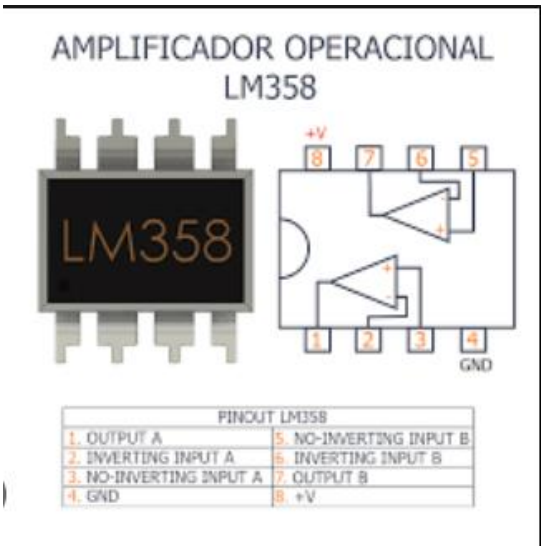
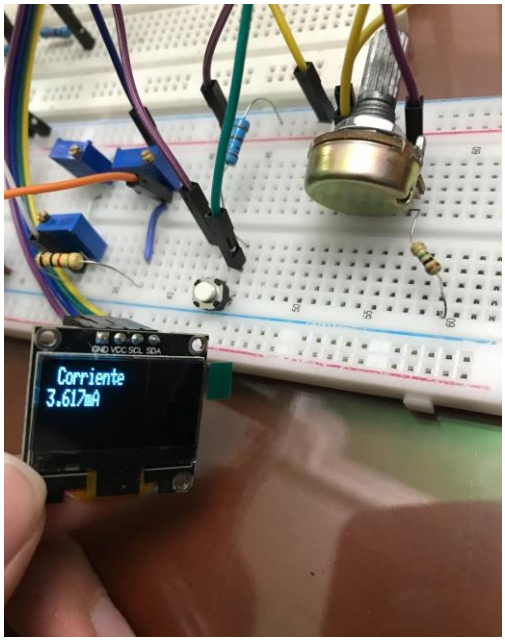


Figura 9: Datasheet del LM350.

Para medir corriente con un Arduino y un amplificador operacional como el LM358, se puede utilizar un resistor de bajo valor (resistencia de carga) en serie con el circuito a medir y luego amplificar la caída de voltaje a través de ese resistor con el amplificador operacional.



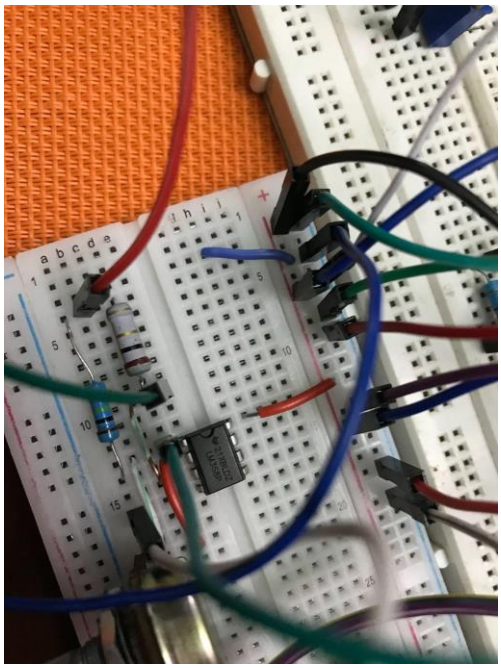


Figura 10: Evidencias fotográficas del montaje del circuito para medir corriente.

5. Conclusiones:

- Se determinó que debido a que la tarjeta Arduino solo acepta valores en formato analógico (ADC), fue necesario aplicar un circuito específico. Este circuito, ingresado como fórmula en el código, permitió ajustar los valores provenientes de las mediciones. De este modo, se logró visualizar en la tarjeta un valor ajustado que representara de manera efectiva la magnitud real medida.

- La medición de voltaje y capacitancia demostró ser la menos demandante en términos de componentes físicos, ya que ambas solo necesitaron dos resistencias, además de la fuente de voltaje en el primer caso o el condensador en el segundo.

- La implementación del puente de Wheatstone mejoró significativamente la precisión al obtener el valor resistivo de la resistencia. Su alta sensibilidad a cualquier variación en la resistencia contribuye directamente al equilibrio del puente, lo que se refleja en una mayor exactitud en las mediciones.

6. Bibliografía

1. “3296 Trimmer Potentiometer”. Components101. [En línea]. Disponible: https://components101.com/resistors/3296-trimmer-potentiometer-pinout-datasheet

2. Pallas Areny, Ramón. Sensores y Acondicionadores de Señal. Prácticas. Ed. Marcombo. México. 4ta edición. 2008.

3. A. D. Helfrick y W. D. Cooper, "Instrumentación Electrónica Moderna y Técnicas de Medición," Pearson Education. México.

4. J. P. Bentley, "Principles of Measurement Systems," 4th ed. Pearson, 2005.

5. AD620 Datasheet(PDF) - Analog Devices. (s.f.). ALLDATASHEET.COM - Electronic Parts Datasheet Search. https://www.alldatasheet.com/datasheet-pdf/pdf/48090/AD/AD620.html