Instituto Politécnico Nacional

Escuela Superior de Cómputo

Práctica 8 Adquisición y visualización de datos con el ADS1115

Sistemas en chip
Profesora: Ana Luz Barrales Lopez

Luciano Hernández Jonathan
Rodriguez Morales Iñaki

3 de diciembre del 2024

1 Introducción

En esta práctica se desarrollará un sistema utilizando un Arduino Uno, enfocado en la lectura de datos ambientales y su visualización en un display LCD. Para ello, se integrarán un sensor BME280, que permite medir parámetros como temperatura y presión atmosférica, y un potenciómetro de 10 k, cuya salida analógica será digitalizada mediante un convertidor ADC ADS1115. Estos elementos se complementan con un LCD sin interfaz I2C para presentar los datos en tiempo real.

La práctica combina conceptos de electrónica analógica y digital, incluyendo el manejo de sensores, la conversión de señales y la visualización de datos. Además, se pondrán en práctica habilidades de programación para la configuración y sincronización de los dispositivos, fomentando una comprensión integral del proceso de adquisición y representación de información en sistemas en chips.

1.1 Objetivo

- Comprender el protocolo de comunicación.
- Configurar y leer datos de ADS1115
- Integración de la lectura de datos del sensor BMP280
- Visualización de los datos adquiridos en el monitor serie Arduino

1.2 Materiales

Los materiales utilizados en la práctica son los siguientes:

- Arduino Uno
- Arduino IDE
- Cable para conexiones
- Módulo ADS1115
- Módulo BMP280
- 2 Potenciometros 10k

2 Desarrollo de la Práctica

2.1 Conexiones

Para implementar esta práctica, el sensor BME280 debe conectarse con SDA al pin A4 y SCL al pin A5 del Arduino Uno, mientras que su VCC se conecta al pin de 3.3V y GND a GND. El módulo ADS1115 comparte los pines SDA y SCL con el BME280, tiene ADDR conectado a GND para usar la dirección por defecto, VCC conectado al pin de 5V del Arduino Uno y GND a GND, con la entrada A0 conectada al cursor del potenciómetro. El potenciómetro debe tener un terminal conectado a 5V, el cursor conectado al canal A0 del ADS1115 y el otro terminal a GND. Finalmente, el LCD sin I2C se conecta con

RS al pin 12, E al pin 11, D4 al pin 5, D5 al pin 4, D6 al pin 3, y D7 al pin 2, además de conectar su VCC al pin de 5V y GND a GND del Arduino Uno.

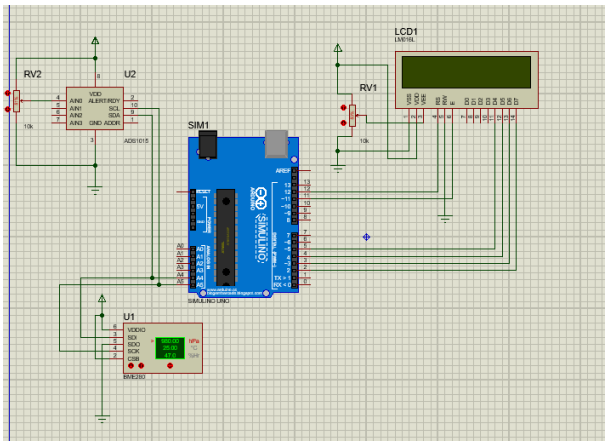

Figure 1: Diagrama de conexión

2.2 Desarrollo del código

2.2.1 Inicialización de Librerías y Componentes

Este bloque incluye las librerías necesarias para controlar el sensor BME280, el convertidor ADC ADS1115, y el display LCD sin I2C. También se declaran e inicializan las instancias de los componentes y las variables globales para manejar los datos de temperatura, presión y voltaje.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <Adafruit_ADS1X15.h>
#include <LiquidCrystal.h>

// Inicializar instancias
Adafruit_BME280 bme; // Sensor BME280
Adafruit_ADS1115 ads; // Convertidor ADS1115

// Pines LCD (ajusta si es necesario)
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// Variables globales
bool bmeConectado = false; // Bandera para validar si el BME280 est
    conectado
float voltaje;             // Voltaje le do del potenci metro
float temperatura;         // Temperatura del BME280
float presion;             // Presi n del BME280
```

2.2.2 Configuración Inicial del Sistema

En este bloque se inicializan la comunicación serial, el LCD, y los dispositivos externos (BME280 y ADS1115). Se verifica si el sensor BME280 está conectado y se configura el rango de ganancia del ADS1115 para medir voltajes entre ±6.144V. También se muestra un mensaje inicial en el LCD y en el monitor serial.

```
void setup() {
  Serial.begin(115200);
  lcd.begin(16, 2);  // Inicializa LCD de 16x2
  lcd.print("Iniciando...");

  // Inicializar BME280
  if (bme.begin(0x76)) {
    bmeConectado = true;
    Serial.println("BME280 conectado correctamente.");
  } else {
    bmeConectado = false;
    Serial.println("Advertencia: BME280 no conectado.");
  }

  // Inicializar ADS1115
  ads.setGain(GAIN_TWOTHIRDS); // Ganancia para rango  6 .144V
  ads.begin();
  Serial.println("ADS1115 inicializado correctamente.");

  delay(2000); // Pausa para mostrar mensajes iniciales
  lcd.clear();
}
```

2.2.3 Lectura de Datos y Visualización

Este bloque implementa el ciclo principal del programa. Se leen los valores del potenciómetro a través del ADS1115 y se convierten a voltaje. Si el BME280 está conectado, se obtienen los valores de temperatura y presión. Los datos se presentan tanto en el LCD como en el monitor serial. El programa se ejecuta en ciclos de un segundo.

```
void loop() {
  // Leer datos del potenci metro
  int16_t valorPot = ads.readADC_SingleEnded(0); // Canal A0
  voltaje = (valorPot * 6.144) / 32767.0; // Conversi n a voltaje

  // Leer datos del BME280 si est  conectado
  if (bmeConectado) {
    temperatura = bme.readTemperature();
    presion = bme.readPressure();
  } else {
    temperatura = 0.0; // Valores por defecto si no est  conectado
    presion = 0.0;
  }

  // Mostrar datos en el LCD
  lcd.setCursor(0, 0); // L nea superior
  lcd.print("T:");
  lcd.print(temperatura, 1);
  lcd.print("C,");
  lcd.print(presion, 0);
  lcd.print("Pa");

  lcd.setCursor(0, 1); // L nea inferior
  lcd.print("V:");
  lcd.print(voltaje, 2);
  lcd.print("V   ");

```

```
28   // Mostrar datos en el monitor serial
29   Serial.print("Temperatura: ");
30   Serial.print(temperatura);
31   Serial.print("  C , Presi n: ");
32   Serial.print(presion);
33   Serial.print(" Pa, Voltaje: ");
34   Serial.print(voltaje, 2);
35   Serial.println(" V");
36
37   delay(1000); // Pausa de 1 segundo
38 }
```

2.3 Simulación

A continuación se muestra el funcionamiento del sistema realizando las conexiones necesarias para la practica, las cuales fueron mostradas en el primer esquema.

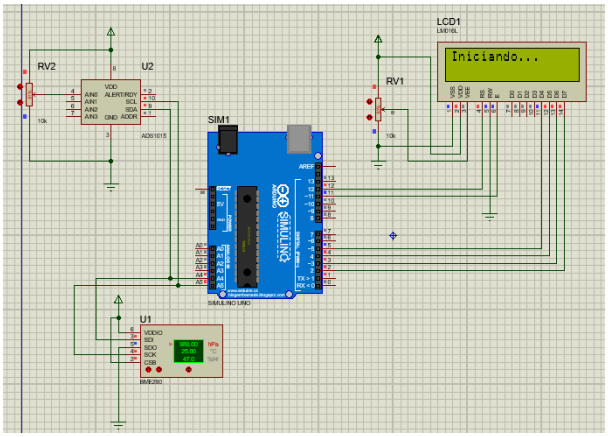

Figure 2: Funcionamiento del circuito

Primera imagen donde se puede observar como se inicializa el programa, mostrando dicho mensaje en el lcd, esperando a leer los datos, tanto de voltaje en el ads115, como en del sensor BME280.

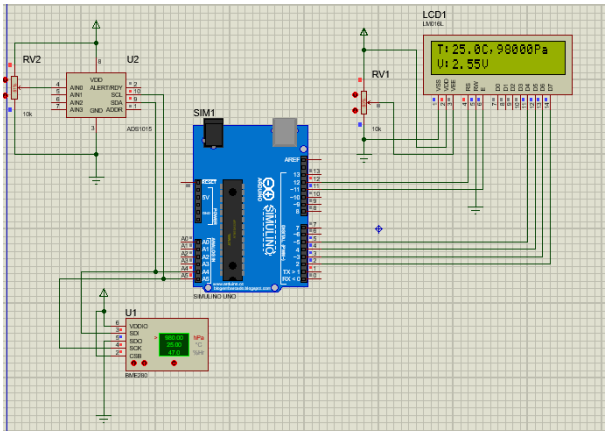

Figure 3: Simulación lectura de datos

En esta segunda imagen se puede observar que una vez recibido la lectura de ambas señales (analogica y digital), muestra los datos en el display LCD.

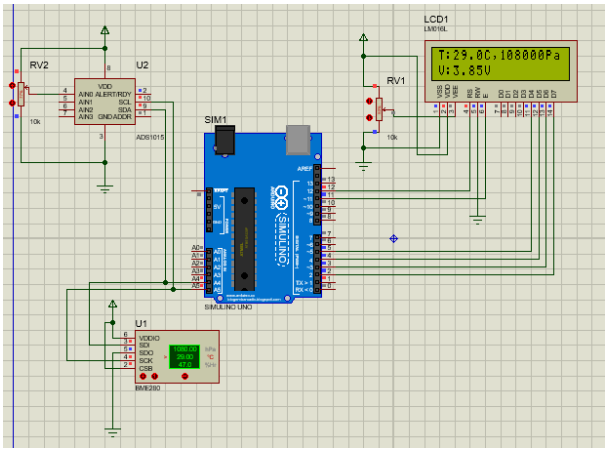

Figure 4: Simulación actualización de datos

Finalmente, en esta simulación se muestra como al variar los valores leidos en ambas señales, esta se actualiza y muestra los datos actualizados en el panel lcd.

2.4 Implementación

Una vez mostrada la simulación del sistema utilizando proteus, se procede a mostrar el funcionamiento del sistema utilizando componentes reales y el mismo equema que se motro al principio de la práctica.

A continuación se muestran las fotografías de su correcto funcionamiento.

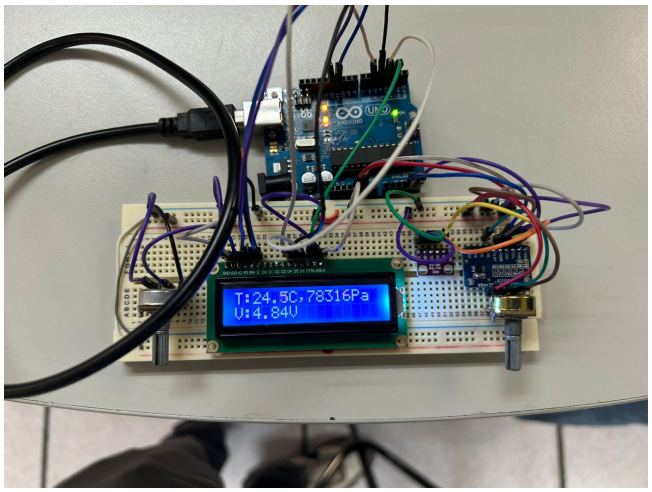

Figure 5: Funcionamiento del cricuito

Primera fotografía donde se puede observar el funcionamiento corrcto del sistema al inicializarse. Mostrando en el LCD los valores recogidos por el sensor y la lectura del voltaje del potenciometro.

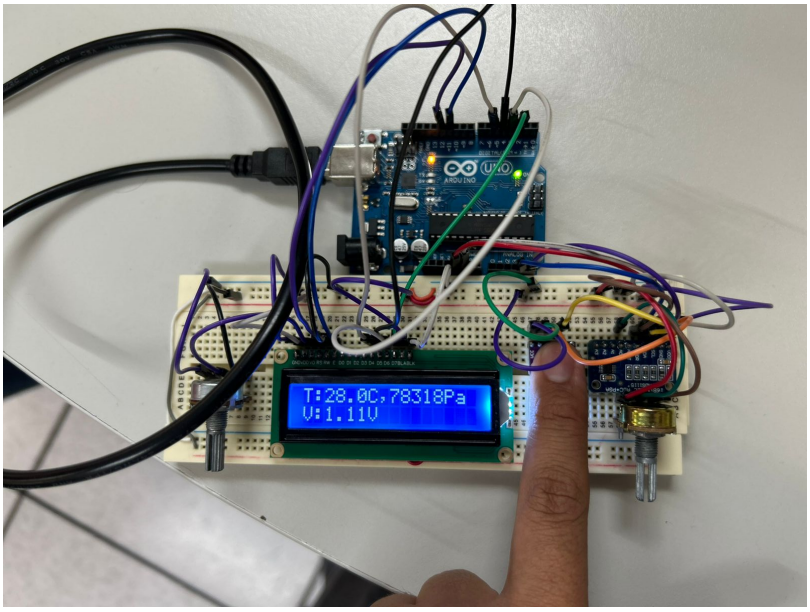

Figure 6: Funcionamiento aumento de temperatura

En esta segunda fotografía se puede observar que el sensor detecta un aumento de temperatura, por lo que manda los datos actualizados al sensor, mostrandolos de manera correcta.

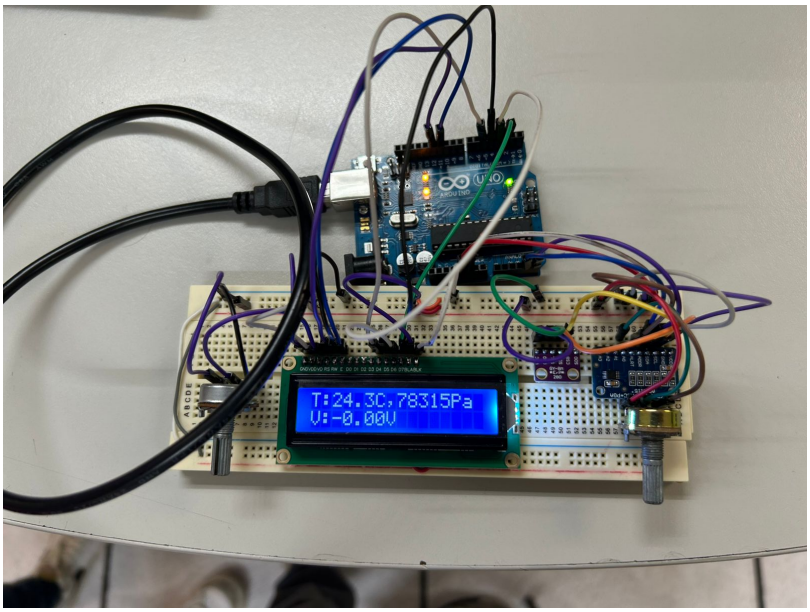

Figure 7: Funcionamiento actualización de voltaje

Finalmente en esta ultima fotografía se puede observar que al manipular el potenciometro, este actualiza los datos de lectura recibidos por el ads1115, y mostrandose manera correcta la actualización en el LCD.

3 Conclusiones

- **Morales Rodriguez Iñaki** La práctica permitió integrar múltiples dispositivos electrónicos, como el sensor BME280, el módulo ADS1115, un potenciómetro y un LCD sin I2C, demostrando cómo gestionar diversos protocolos y señales analógicas en un sistema embebido. La correcta configuración del protocolo I2C para el sensor BME280 y el ADS1115 facilitó la comunicación con el Arduino Uno, destacando la importancia de comprender la interacción entre componentes digitales y analógicos. Asimismo, la visualización de datos en un LCD sin I2C reafirmó la necesidad de manejar eficientemente los recursos del microcontrolador para implementar soluciones que combinen lectura de sensores, cálculos y despliegue de resultados.

- **Luciano Hernández Jonathan** Esta práctica fue un ejemplo integral de adquisición, procesamiento y visualización de datos en un sistema basado en Arduino. La conexión del sensor BME280 permitió medir parámetros ambientales críticos como temperatura y presión, mientras que el potenciómetro, acoplado al ADS1115, proporcionó un ejemplo de lectura y conversión de señales analógicas. La integración con el LCD sin I2C permitió desplegar los datos en tiempo real, lo que demostró la versatilidad del Arduino Uno para manejar diversos dispositivos simultáneamente. En conjunto, esta experiencia resaltó la importancia de planificar adecuadamente las conexiones y utilizar estrategias de programación para optimizar el desempeño del sistema.

4 Referencias

1 LvRfxGTUEpE, “Video relacionado,” YouTube, Oct. 25, 2023. [Online]. Available: https://www.youtube.com/watch?v=LvRfxGTUEpE. [Accessed: Nov. 21, 2024].

2 Programación de microcontroladores y electrónica. Introducción a la Programación de Microcontroladores Básicos 1. YouTube, 20 mar. 2024. [En línea]. Disponible en: https://www.youtube.com/watch?v=QqIsp6bjAwA. [Accedido: 2 dic. 2024].

3 M. Moreudev, “Simulando hardware en Proteus - Tutorial,” YouTube, Oct. 2024. [Online]. Available: https://www.youtube.com/watch?v=ExQGweONqkU. [Accessed: Dec. 2, 2024].