UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE LAS CIENCIAS EXACTAS E INGENIERÍAS

INGENIERIA ROBOTICA

SEMINAR ON EMBEDDED SYSTEMS PROGRAMMING PROBLEMS

"ACTIVITY 5"

Materia: Seminar on Embedded Systems Programming Problems
Teacher: Daniel Martinez

Castillo Hernández Luis Fernando
Almanza Castañeda Favio Enrique
Lira Alamo Damian

FEBRUARY 25, 2025

INTRODUCTION:

This practice aims to demonstrate the process of reading an analog signal from a potentiometer using an ESP32/ESP8266, converting it to a voltage and angle, and then using the processed data to control the brightness of an LED via Pulse Width Modulation (PWM). The experiment illustrates fundamental concepts of Analog-to-Digital Conversion (ADC) and PWM signal generation.

MATERIALS USED:

- Microcontroller ESP32
- 10kΩ Potentiometer
- LED(Generic)
- **NPN transistors (2N2222)** (used to switch each digit individually)
- **330Ω resistors** (to limit LED current)
- Jumper wires and a breadboard

DEVOLPMEN:

The code defines two pins:

- POT_PIN (Pin 4): Connected to the potentiometer for ADC input.Segment control pins: Connected to the ESP32/ESP8266 to define digit segments (2, 4, 5, 18, 19, 21, 20).
- **LED_PIN (Pin 5):** Used to output a PWM signal that controls LED brightness.

Analog Reading and Data Conversion

- The potentiometer's analog signal is read using analogRead(POT_PIN), which returns a value between 0 and 8191 (since ESP32 has a 12-bit ADC resolution).
- The raw ADC value is converted to voltage using:

$$V = \frac{ADC\ Reading}{8191} * 3.3V$$

- The voltage is then mapped to an angle range (0° to 270°) assuming a linear potentiometer:

$$\theta = \frac{v}{3.3V} *\ 270°$$

PWM Output for LED Brightness

- The ADC reading is mapped to an 8-bit PWM value (0–255) using map(), ensuring that lower potentiometer values result in dimmer LED brightness and higher values increase brightness.
- The LED brightness is adjusted using analogWrite(LED_PIN, brillo)

Serial Output for Debugging

- The calculated voltage and angle are printed to the serial monitor using Serial.printf(), enabling real-time monitoring of the potentiometer's behavior.

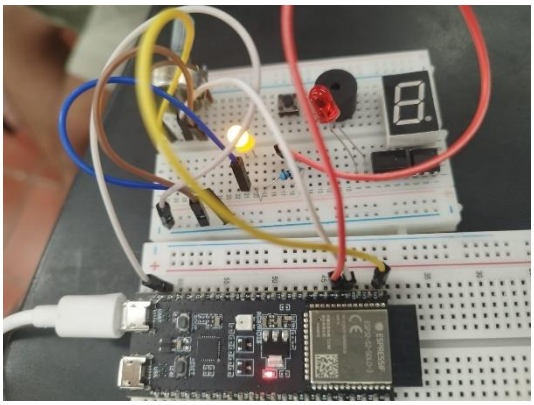
```
void loop() {
  // Leer el potenciómetro
  int lecturaADC = analogRead(POT_PIN); // 0 - 8191
  float voltage = lecturaADC / 8191.0 * 3.3;  // Convertir a voltaje
  float angulo = voltage / 3.3 * 270.0;  // Convertir a ángulo

  // Convertir a brillo PWM (0 - 255)
  int brillo = map(lecturaADC, 0, 8191, 0, 255);

  // Imprimir valores por serial
  Serial.printf("Voltaje: %.2fV, Ángulo: %.1f°\n", voltage, angulo);

  // Ajustar brillo del LED (sin usar ledcSetup)
  analogWrite(LED_PIN, brillo);

  delay(500);
}
```

RESULTS AND OBSERVATIONS:

- **Expected behavior:** The LED brightness smoothly increases as the potentiometer is rotated. The serial monitor correctly displays voltage and angle values.

Potential improvement:

- Use ledcWrite() instead of analogWrite() for better PWM control on ESP32.
- Implement **ADC calibration** to improve accuracy.
- Reduce delay(500) for smoother LED transitions.



CONCLUSIONS:

"This experiment effectively demonstrates ADC reading, voltage conversion, and PWM output in an embedded system. The ESP32/ESP8266's ADC resolution (12-bit) provides high precision, while PWM enables smooth LED brightness control. These principles are widely applicable in sensor-based automation and embedded control systems."