MONITOREO DE TEMPERATURA CON RASPBERRY PI PICO W Y THINGSPEAK

Internet de las Cosas

28 DE FEBRERO DE 2025

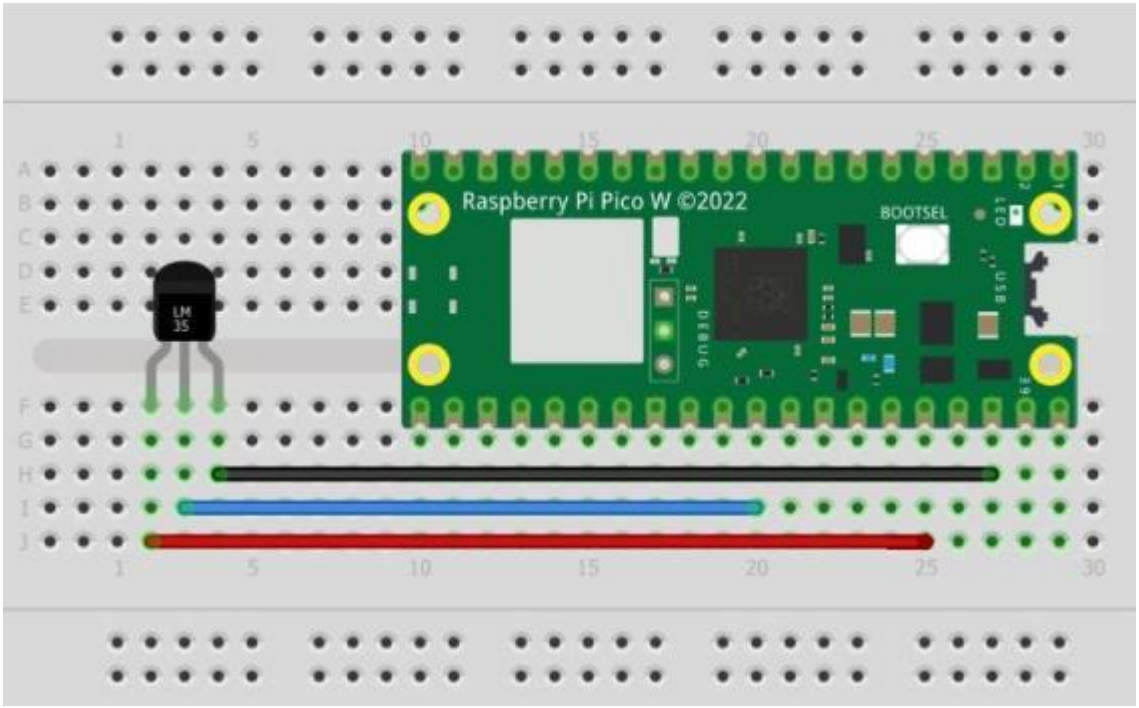
UNIVERSIDAD MODELO
FREDDY ANTONIO IX ANDRADE
Marco Antonio Moreno Sánchez - 21091535

Introducción

El objetivo de este proyecto es desarrollar un sistema de monitoreo de temperatura utilizando la Raspberry Pi Pico W y un sensor LM35, enfocado en la captura y visualización de datos a través de internet. El sistema estará conectado a ThingSpeak, para lograr almacenar y mostrar las lecturas de temperatura en tiempo real. Además, se integrará con MathWorks para realizar análisis de datos, como el cálculo del promedio de temperatura. El sistema también contará con una funcionalidad de alertas que se activarán cuando las temperaturas superen los 35 grados, mejorando así el control y la supervisión del entorno monitoreado.

Desarrollo del Proyecto

Diagrama de Conexión



La salida del LM35 genera un voltaje linealmente proporcional a la temperatura en grados Celsius.

A diferencia de otros sensores de temperatura, el LM35 no requiere una compensación adicional para medir la temperatura, ya que su escala es directamente compatible con la unidad **Celsius**.

Programación en MicroPython

Como se puede ver en la Ilustración 1.1, la raspberry intenta conectarse a internet, para poder conectarse a la red es necesario que el código tenga el nombre de la red y su respectiva contraseña, por lo cual se lo agregamos tal y como se puede ver en la Ilustración 1.2

```
# Función para conectar al WiFi
def connect_wifi(ssid, password):
    wlan = network.WLAN(network.STA_IF)
    wlan.active(True)
    if not wlan.isconnected():
        print('Conectando al WiFi...')
        wlan.connect(ssid, password)
        while not wlan.isconnected():
            time.sleep(1)
    print('Conexión establecida. Dirección IP:', wlan.ifconfig()[0])
```

Ilustración 1.1 Código Wi-Fi

```
# Configuración de WiFi
ssid = 'IZZI-8798'
password = 'FCAE34B98798'

# Conectar a WiFi
connect_wifi(ssid, password)
```

Ilustración 1.2 Red y Contraseña

El sensor LM35 es un sensor de temperatura analógico. La Raspberry Pi Pico W tiene un conversor analógico-digital (ADC) que convierte la señal analógica del LM35 a una señal digital, le asignamos el PIN de entrada tal y como se observa en la Ilustración 2.1 y en la Ilustración 2 se puede ver el bucle principal del código el cual permite leer la temperatura y llegar un aviso si excedía el límite tal y como se puede observar en la Ilustración 2.2.

```
adc = ADC(Pin(26))
```

Ilustración 2.1 Asignación de Pin

```python
# Bucle principal
while True:
    # Leer la temperatura del LM35
    temperature = read_temperature()
    print('Temperatura: {:.2f} °C'.format(temperature))

    # Enviar la temperatura a ThingSpeak
    send_to_thingspeak(temperature)

    # Si la temperatura supera los 40°C, enviar un correo de alerta
    if temperature > 40:
        send_alert_email(
            subject="¡Alerta de Temperatura Alta!",
            body=f"La temperatura ha superado los 40°C. Temperatura actual: {temperature:.2f}°C"
        )

    # Esperar 60 segundos antes de enviar el siguiente dato
    time.sleep(60)
```

Ilustración 2.2 Bucle Principal

El LM35 produce una señal analógica que cambia según la temperatura. Para convertir esta señal analógica a una temperatura en grados Celsius, se utiliza una fórmula, dado que el LM35 entrega 10mV por cada grado Celsius, esta fórmula se puede ver en la Ilustración 3.1.

```python
def read_temperature():
    # El LM35 proporciona 10mV por grado Celsius, Acon 3.3V de referencia
    voltage = adc.read_u16() * (3.3 / 65535)  # Lectura de 16 bits (0-65535)
    temperature_celsius = voltage * 100  # Convertir el voltaje a temperatura (10mV/°C)
    return temperature_celsius
```

Ilustración 3.1 Conversor

Este código envía los datos de temperatura al canal de ThingSpeak utilizando su API (Ilustración 4.1).

```python
# Función para enviar los datos a ThingSpeak
def send_to_thingspeak(temperature):
    # Reemplaza 'YOUR_WRITE_API_KEY' con tu clave API de escritura de ThingSpeak
    api_key = 'EA6PPN2N46MRAXO0'
    url = f'https://api.thingspeak.com/update?api_key={api_key}&field1={temperature}'

    try:
        response = urequests.get(url)
        if response.status_code == 200:
            print("Datos enviados correctamente a ThingSpeak.")
        else:
            print("Error al enviar los datos.")
        response.close()
    except Exception as e:
        print("Error de conexión:", e)
```

Ilustración 4.1 Envío de datos

Visualización en ThingSpeak

El sensor trajo una falla y no es estable tal y como se puede ver en la Ilustración 5.1, si bien arrojaba a ratos datos mayores a 50 y menores a 10 grados, en el código implementé un limitante entre 40 y 25, si la temperatura era mayor a 40 o menor a 25 no manda los datos, se probó cambiar el lm35 pero seguía ocurriendo el mismo problema.

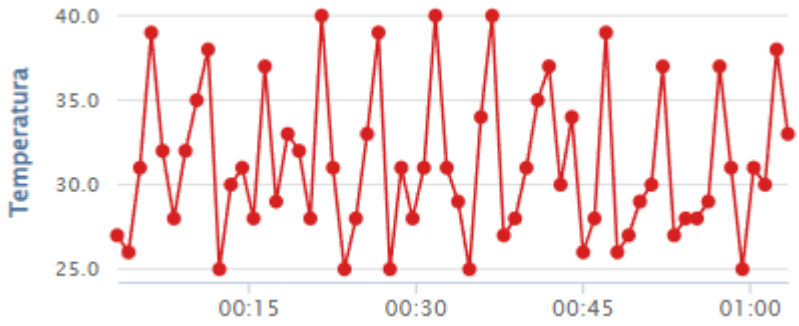


Ilustración 5.1 Medidas Sensor

Al promediar los 1500 datos en 10 en 10, nos da los resultados que se pueden observar en la Ilustración 5.2, al ningún promedio sobrepasar los 35 grados no se logró recibir el mensaje de alerta en ningún momento.

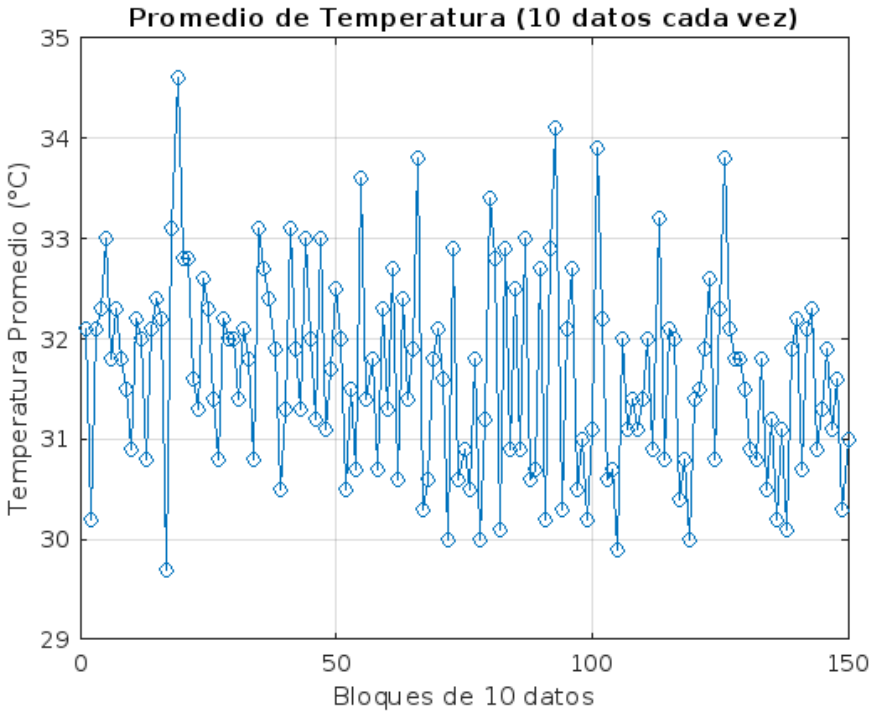


Ilustración 5.2 Gráfica Promedio

Análisis de Resultados

La temperatura nunca era estable, aunque por un lado llega a tener medio razón debido a que el lugar donde se situó es un lugar donde la temperatura cambia mucho y muy rápido, aun así no se llega a explicar el porque las fluctuaciones tan abruptas de temperatura. Al sacar el promedio en MathWorks se puede ver que este se estabiliza más.

Cumplimiento de los Objetivos del Proyecto

Se lograron todos los objetivos con la única excepción de el aviso que supere los 35 grados, si bien al sacar el promedio nunca supero esa barrera, al momento de "forzar" a que este superé los 35 grados no saltaba el mensaje, los demás se cumplieron tal y como se puede observar en el presente documento.

En resumen, los objetivos del proyecto fueron alcanzados a excepción del aviso. El sistema funciona correctamente al medir, enviar y procesar los datos de temperatura. Esto no solo permite el monitoreo en tiempo real, sino también la toma de decisiones basadas en las condiciones del entorno o los equipos monitoreados.

Conclusión y Reflexión

Aprendí lo útil que puede llegar a resultar conocer los datos de un sensor en tiempo real ya sea térmico, eléctrico, etc. Lo único que no logro entender no es el tema de la comunicación si no el tema de la detección del sensor, esto debido a que se probó con 2 lm35 pero aun así la temperatura nunca lograba estabilizarse. En un futuro este proyecto se puede mejorar conectándolo como por ejemplo a un motor y, en vez de utilizar un sensor de temperatura, un sensor de corriente, esto haría que el motor se moviese y al sobrepasar un cierto nivel de amperaje el motor haga una acción definida, en el caso del lm35 se podría comparar con un refrigerador para saber si este tiene el nivel de temperatura correcto.

En la industria esta aplicación puede resultar muy útil debido a que el circuito demuestra que funciona subiendo los datos a la Red y, en caso de que supere los datos deseados, este manda un aviso para que el encargado de área pueda apagar el sistema y arreglar el error del sistema.