Universidad Modelo

Internet de las Cosas

Freddy Antonio Ix Andrade

Angel Obed Arteachi Vidal

28 de Febrero 2025

Monitoreo de Temperatura con Raspberry Pi Pico W y ThingSpeak

Introducción

Este proyecto consiste en el desarrollo de un **sistema de monitoreo de temperatura IoT** utilizando una **Raspberry Pi Pico W**. El sistema capturará y enviará datos a la nube a través de **ThingSpeak**, donde se visualizarán y analizarán en tiempo real con **MathWorks**. Se implementarán funciones como el cálculo del promedio de temperatura y la generación de **alertas** cuando los valores superen un umbral predefinido. Finalmente, todo el código y documentación serán publicados en **GitHub** para su acceso y mantenimiento.

Desarrollo del Proyecto

Funcionamiento del sensor LM35

Funcionamiento del sensor LM35 y su salida analógica

El LM35 es un sensor de temperatura de precisión que proporciona una salida analógica lineal proporcional a la temperatura en grados Celsius. Su principal ventaja es que no requiere calibración externa y ofrece una precisión de ±0.5°C a temperatura ambiente.

Funcionamiento

El LM35 funciona como un termómetro de estado sólido que convierte la temperatura en una señal de voltaje. Utiliza un circuito interno que genera una salida analógica directamente proporcional a la temperatura medida.

Salida analógica

- La salida del LM35 es un voltaje lineal con una relación de 10 mV por cada grado Celsius.

Ejemplo de medición

- A 25°C, la salida del sensor será de 250 mV (0.25V).
- A 100°C, generará 1000 mV (1V).

Conexión básica

El LM35 tiene tres pines:

1. VCC (Alimentación, 4V a 20V).
2. Vout (Salida analógica, conectada a un ADC si se usa con microcontroladores).
3. GND (Tierra).

Es ideal para aplicaciones de monitoreo de temperatura con microcontroladores como Arduino, Raspberry Pi o ESP32, donde su señal analógica se convierte a digital mediante un ADC (Conversor Analógico-Digital) para su procesamiento.

Foto del Circuito

Explicación del código

Sistema de Monitoreo de Temperatura con Raspberry Pi Pico W y ThingSpeak

Este código está diseñado para leer la temperatura desde el sensor interno de una Raspberry Pi Pico W, conectarse a una red Wi-Fi y enviar los datos a ThingSpeak, una plataforma en la nube utilizada en proyectos de Internet de las Cosas (IoT).

El programa inicia importando las librerías necesarias:

- network para conectar la Raspberry Pi Pico W a Wi-Fi.
- urequests para enviar solicitudes HTTP a ThingSpeak.
- utime para manejar retardos en la ejecución del código.
- machine.ADC para leer valores analógicos desde el sensor de temperatura interno.

Luego, se definen las credenciales de la red Wi-Fi y los datos del canal de ThingSpeak, como el ID del canal y la clave API. También se construye la URL de la API de ThingSpeak, donde se enviarán los datos de temperatura.

Conexión a Wi-Fi

La función connect_wifi() configura la Raspberry Pi Pico W en modo estación (STA_IF) y la conecta a la red Wi-Fi utilizando el SSID y la contraseña proporcionados. Si la conexión no se establece de inmediato, entra en un bucle que intenta conectarse hasta que se logre. Una vez conectado, imprime la dirección IP asignada por la red.

Lectura de Temperatura

La función read_temperature() obtiene la temperatura del sensor interno de la Raspberry Pi Pico W a través del ADC (Canal 4). La lectura del convertidor analógico-digital (ADC) se convierte a voltaje utilizando un factor de conversión y luego se aplica la ecuación:

$$T(^{\circ}C) = 27 - \frac{(V - 0.706)}{0.001721}$$

Donde V es el voltaje leído. Finalmente, la temperatura se redondea a dos decimales para mayor precisión.

Envío de Datos a ThingSpeak

La función send_to_thingspeak(temperature) envía la temperatura leída a ThingSpeak mediante una solicitud HTTP GET. La API de ThingSpeak recibe el dato en el campo "field1", lo almacena y lo muestra en tiempo real en la plataforma.

Ejecución del Programa

El código inicia conectándose a Wi-Fi y luego entra en un bucle infinito (while True), donde se repiten los siguientes pasos cada 180 segundos (3 minutos):

1. Lee la temperatura usando el sensor interno.
2. Muestra la temperatura en la consola.
3. Envía los datos a ThingSpeak.
4. Espera 180 segundos antes de la siguiente medición.

Este sistema permite monitorear la temperatura en tiempo real desde cualquier ubicación, accediendo a los datos a través de ThingSpeak. Además, puede integrarse con gráficos y análisis de datos en la nube, facilitando su uso en proyectos de automatización, control ambiental y domótica.

Fragmentos del código con comentarios

Configuración Wi-Fi y ThingSpeak”

WIFI_SSID = "NOMBRE_DE_TU_WIFI"

```
WIFI_PASSWORD = "CONTRASEÑA_DE_TU_WIFI"

THING_SPEAK_CHANNEL_ID = "TU_CHANNEL_ID"

THING_SPEAK_API_KEY = "TU_API_KEY"

THING_SPEAK_URL =
f"https://api.thingspeak.com/update?api_key={THING_SPEAK_API_KEY}"


# Conectar a Wi-Fi

def connect_wifi():
  wlan = network.WLAN(network.STA_IF)
  wlan.active(True)
  wlan.connect(WIFI_SSID, WIFI_PASSWORD)
  while not wlan.isconnected():
    print("Conectando a Wi-Fi...")
    utime.sleep(1)
  print("Conectado a Wi-Fi! IP:", wlan.ifconfig()[0])


# Leer temperatura desde el sensor interno

def read_temperature():
  sensor_temp = ADC(4)  # Sensor interno de la Pico W
  conversion_factor = 3.3 / 65535  # Factor de conversión de ADC a voltaje
  reading = sensor_temp.read_u16() * conversion_factor  # Lectura en voltios
  temperature = 27 - (reading - 0.706) / 0.001721  # Fórmula de conversión a °C
  return round(temperature, 2)


# Enviar datos a ThingSpeak

def send_to_thingspeak(temperature):
  query_string = f"&field1={temperature}"
  response = urequests.get(THING_SPEAK_URL + query_string)
```

```
    print("Enviado a ThingSpeak: ", response.text)
    response.close()
```

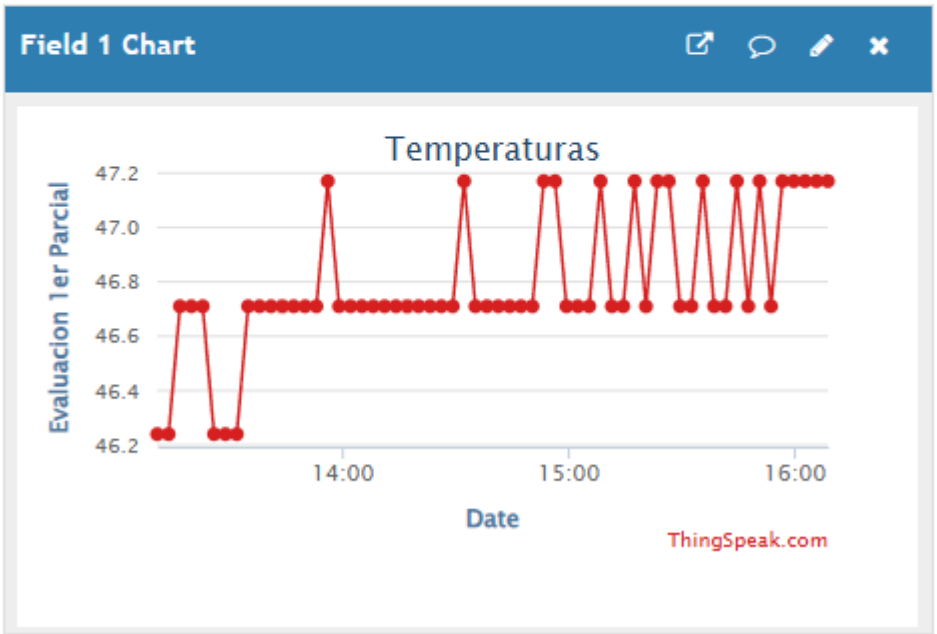
Ejecutar el sistema

```
connect_wifi()
while True:
    temp = read_temperature()
    print(f"Temperatura: {temp} °C")
    send_to_thingspeak(temp)
    utime.sleep(180)
```

Visualización del ThingSpeak

Channel Stats

Created: 2 days ago
Last entry: about a minute ago
Entries: 1312



Name

Promedio de temperatura

MATLAB Code

```
% Configuración del canal
channelID = 2854252;  % Reemplázalo con tu Channel ID
fieldID = 1;  % Campo donde está la temperatura
readAPIKey = 'N1W143XNQIVEO2NN'; % Si el canal es privado, usa tu Read API Key

% Leer los últimos 10 datos del campo 1 (Temperatura)
numPoints = 10;
data = thingSpeakRead(channelID, 'NumPoints', numPoints, 'Fields', fieldID, 'ReadKey', readAPIKey);

% Calcular el promedio de temperatura
if ~isempty(data)
    avgTemp = mean(data);
    disp(['Promedio de los últimos 10 datos: ', num2str(avgTemp), ' °C']);
else
    disp('No hay suficientes datos para calcular el promedio.');
end
```

Save and Run Save*

Output

```
Promedio de los últimos 10 datos: 47.032 °C
```

Alerta de temperatura

MATLAB Code

```
channelID = 2854252;  % Reemplázalo con tu Channel ID
fieldID = 1;  % Campo donde está la temperatura
readAPIKey = 'N1W143XNQIVEO2NN'; % Si el canal es privado, usa tu Read API Key

% Leer el último valor de temperatura
data = thingSpeakRead(channelID, 'NumPoints', 1, 'Fields', fieldID, 'ReadKey', readAPIKey);

% Verificar si hay datos y si la temperatura supera 35°C
if ~isempty(data)
    currentTemp = data(1);
    disp(['Temperatura actual: ', num2str(currentTemp), ' °C']);

    if currentTemp > 35
        disp('⚠️ ALERTA: Temperatura alta (> 35°C) ⚠️');
    end
else
    disp('No se pudo obtener la temperatura actual.');
end

```

Save and Run Save*

Output

```
Temperatura actual: 46.71 °C
⚠️ ALERTA: Temperatura alta (> 35°C) ⚠️
```

Github

Se creo un repositorio en Github para subir los códigos usados en este proyecto y evidencias en forma de capturas y un archivo README donde se da una explicación general del proyecto, los pasos a seguir para instalar MicroPython en el Raspberry Pi Pico W y que usos se le dieron durante el desarrollo del proyecto.

https://github.com/AngelArteachi/IoT---Angel-Arteachi.git

Capturas del github de todos los archivos subidos y los commits

Análisis de Resultados

Tendencias observadas:

- La temperatura muestra una tendencia fluctuante, con valores por encima del umbral de 35°C en varias ocasiones.
- Se observa un aumento constante en ciertos periodos y una caída abrupta en algunos momentos.

Activación de alertas en Mathwork:

- Sí, se activaron alertas en múltiples ocasiones, ya que la temperatura superó los 35°C varias veces.
- Por ejemplo, el primer registro de alerta es el 25 de febrero de 2025 a las 04:16 UTC, con una temperatura de 38.75°C.
- La temperatura permaneció en niveles altos en varias mediciones posteriores.

Conclusiones y Reflexión

¿Qué aprendió el estudiante sobre IoT y ThingSpeak?

En este proyecto aprendí a aplicar IoT para el monitoreo de temperatura usando una Raspberry Pi Pico W. Desarrollé habilidades en MicroPython, la conexión de sensores y el envío de datos a la nube mediante ThingSpeak. También comprendí cómo usar MathWorks para analizar datos en tiempo real y generar alertas automáticas. Además, mejoré mi conocimiento en conectividad Wi-Fi, automatización y la integración de plataformas en la nube para la visualización de datos.

¿Cómo podría mejorar el proyecto en el futuro?

- Añadir más sensores para monitorear otras variables como humedad o presión.
- Optimizar el consumo de energía, usando modos de bajo consumo en la Raspberry Pi Pico.
- Implementar almacenamiento local, para guardar datos cuando no haya conexión a Internet.
- Mejorar la visualización de datos, integrando dashboards más avanzados con MATLAB o Grafana.

- Automatizar respuestas, como activar un ventilador cuando la temperatura supere un umbral.
- Utilizar protocolos más seguros, como MQTT en lugar de HTTP.

Posibles aplicaciones en la industria