

**UNIVERSIDAD MODELO
INGENIERÍA EN
DESARROLLO DE TECNOLOGÍA Y SOFTWARE**



Semestre VI

Internet de las cosas (IoT)

Evaluación del Primer Parcial

Maestro:
Freddy Antonio Ix Andrade

Alumna:
Mendoza Díaz Ingrid

Fecha de entrega
24/02/2025

Introducción

Este documento presenta el desarrollo de un sistema IoT para la medición y monitoreo de temperatura en la nube, implementado con una Raspberry Pi Pico W y un sensor de temperatura LM35. El sistema permite capturar datos de temperatura, enviarlos a ThingSpeak y analizarlos mediante MathWorks.

A lo largo del desarrollo de este proyecto, adquirí y reforcé diversos conocimientos, entre ellos:

- Conexión y configuración de hardware IoT, integrando la Raspberry Pi Pico W con el sensor LM35.
- Programación en MicroPython, incluyendo la lectura de datos analógicos, conversión a temperatura y conexión a Wi-Fi.
- Comunicación con la nube, enviando datos a ThingSpeak para su almacenamiento y visualización.
- Análisis de datos en la nube, utilizando MathWorks para calcular el promedio de temperatura y generar alertas cuando se superen los 35°C.
- Visualización en tiempo real, configurando gráficos en ThingSpeak para monitorear los datos de manera efectiva.

Esta documentación que se mostrará a continuación describe cada fase del desarrollo, desde la configuración del hardware y la programación del microcontrolador hasta el análisis y la visualización de los datos en la nube.

Objetivo

Desarrollar un sistema de monitoreo de temperatura basado en Internet de las Cosas (IoT) utilizando una Raspberry Pi Pico W y un sensor LM35, con el propósito de capturar, procesar y visualizar datos en la nube mediante ThingSpeak. Además, se implementará MathWorks en la plataforma para realizar análisis de datos en tiempo real, como el cálculo del promedio de temperatura y la generación de alertas cuando los valores superen un umbral predefinido.

Desarrollo del proyecto

1. Configuración del hardware

Configuración del Hardware

Para el desarrollo de este sistema IoT de monitoreo de temperatura, se requiere la integración de la Raspberry Pi Pico W con el sensor de temperatura LM35. A continuación, se detallan los pasos para la correcta conexión y configuración del hardware.

1. Componentes Utilizados

- Raspberry Pi Pico W(microcontrolador con conectividad Wi-Fi).
- Sensor de temperatura LM35 (sensor analógico para medir temperatura en grados Celsius).
- Resistencias (opcional, según configuración del circuito).
- Cables de conexión (jumper cables).
- Fuente de alimentación USB para la Raspberry Pi Pico W.

2. Conexión del Sensor LM35 a la Raspberry Pi Pico W

El sensor LM35 es un dispositivo analógico que genera una salida de voltaje proporcional a la temperatura en grados Celsius. Para leer estos valores, se debe conectar el sensor a la Raspberry Pi Pico W de la siguiente manera:

Pin LM35	Conexión en Raspberry Pi Pico W
VCC (pin 1)	3.3V (pin 36 en la Pico W)
VOUT (pin 2)	GP26 (ADC0) (pin 31 en la Pico W)
GND (pin 3)	GND (pin 38 en la Pico W)

Con esta configuración, la Raspberry Pi Pico W estará lista para leer la temperatura desde el LM35 y enviarla a la nube a través de ThingSpeak.

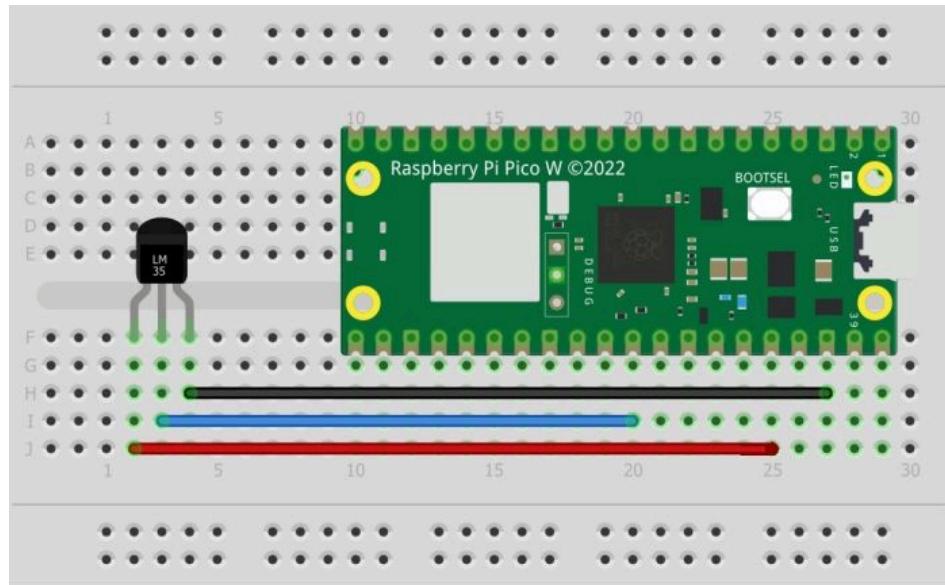
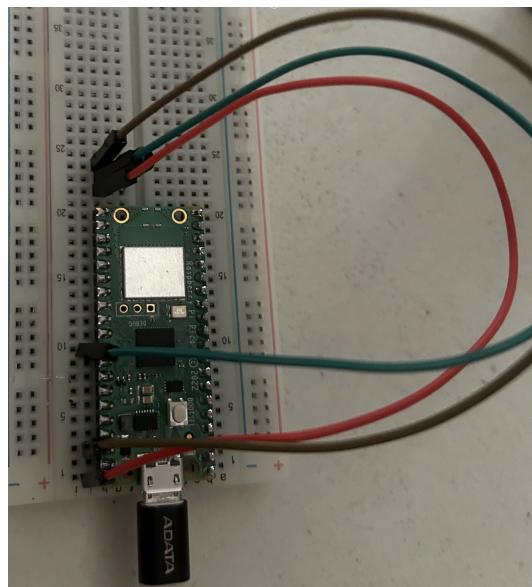


Diagrama de Conexión virtual



Dispositivo ya conectado

2. Programación en MicroPython

1. Importación de Módulos

El código comienza importando los módulos necesarios:

- network: Permite gestionar la conexión Wi-Fi de la Raspberry Pi Pico W.

- urequests: Módulo de MicroPython para realizar solicitudes HTTP (enviar datos a ThingSpeak).
- machine: Proporciona acceso a hardware, como pines y el convertidor analógico-digital (ADC).
- utime: Se usa para manejar pausas y medir el tiempo transcurrido.

2. Configuración de Wi-Fi

```
SSID = "Casa Mendoza"
```

```
PASSWORD = "9818211992"
```

Define el nombre de la red Wi-Fi (**SSID**) y su contraseña (**PASSWORD**) para establecer la conexión a Internet.

3. Configuración de ThingSpeak

```
THINGSPEAK_API_KEY = "2R7WW3XCBDEG304Q"
```

```
THINGSPEAK_URL = "https://api.thingspeak.com/update?api_key=" +  
    THINGSPEAK_API_KEY
```

Se define la clave de la API de **ThingSpeak**, que es necesaria para enviar datos. Luego, se crea la URL base para el envío de datos a la plataforma.

4. Configuración del ADC (Conversión Analógica-Digital)

```
adc = machine.ADC(26)
```

Se inicializa un objeto **ADC (Convertidor Analógico-Digital)** en el pin **GP26**, que leerá el voltaje del sensor **LM35**.

5. Función para Conectar a Wi-Fi

```
def conectar_wifi():
```

```
wlan = network.WLAN(network.STA_IF) # Crea una interfaz Wi-Fi en modo estación
```

```
wlan.active(True) # Activa la interfaz Wi-Fi

wlan.connect(SSID, PASSWORD) # Intenta conectar con las credenciales definidas

print("Conectando a WiFi...")

tiempo_inicio = utime.time() # Guarda el tiempo de inicio

while not wlan.isconnected(): # Espera hasta que se conecte

    if utime.time() - tiempo_inicio > 10: # Si pasan más de 10 segundos, se cancela

        print("Error: No se pudo conectar a WiFi")

        return False

    utime.sleep(1) # Espera 1 segundo antes de volver a verificar

print("Conectado a WiFi:", wlan.ifconfig()) # Muestra los detalles de conexión

return True
```

Explicación:

- Activa la interfaz Wi-Fi en modo **estación** (cliente de una red).
- Intenta conectarse a la red definida con **SSID** y **PASSWORD**.
- Espera hasta 10 segundos para conectarse; si falla, muestra un mensaje de error.
- Si se conecta, imprime la configuración IP asignada.

6. Función para Leer la Temperatura del LM35

```
def leer_temperatura():

    valor_adc = adc.read_u16() # Lee el valor analógico del LM35 (0-65535)
```

```

voltaje = (valor_adc / 65535) * 3.3 # Convierte el valor en voltaje (0-3.3V)

temperatura_celsius = voltaje * 100 # Convierte el voltaje en temperatura (10mV/°C)

return temperatura_celsius

```

Explicación:

- El **LM35** entrega un voltaje proporcional a la temperatura (**10mV por cada °C**).
- Se lee el valor analógico (de 0 a 65535).
- Se convierte a voltaje usando la relación: Voltaje = ((Valor ADC) / 65535) × (3.3V)
- Se convierte a temperatura en grados Celsius usando la relación del LM35:
Temperatura=Voltaje × 100
- Se devuelve el valor de la temperatura.

7. Función para Enviar Datos a ThingSpeak

```

def enviar_a_thingspeak(valor):

    url = THINGSPEAK_URL + "&field1=" + str(valor) # Agrega el valor en field1

    try:

        respuesta = urequests.get(url) # Envía la solicitud HTTP GET

        print("Enviado a ThingSpeak:", respuesta.text) # Muestra la respuesta

        respuesta.close()

    except Exception as e:

        print("Error al enviar datos:", e) # Captura errores de conexión

```

Explicación:

- Se construye la **URL** con el dato de temperatura en **field1**.
- Se envía una solicitud HTTP **GET** a **ThingSpeak** con el valor de la temperatura.

- Se imprime la respuesta del servidor.
- Si ocurre un error en la conexión, se captura y muestra.

8. Bucle Principal

```
if conectar_wifi(): # Primero se intenta conectar a Wi-Fi

    while True: # Bucle infinito

        temperatura = leer_temperatura() # Lee la temperatura del sensor

        print("Temperatura LM35:", temperatura, "°C") # Muestra la temperatura en
        consola

        enviar_a_thingspeak(temperatura) # Envía el dato a ThingSpeak

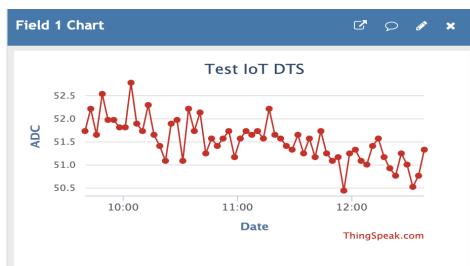
        utime.sleep(180) # Espera 3 minutos antes de la siguiente lectura
```

Explicación:

1. **Verifica la conexión Wi-Fi** antes de iniciar el monitoreo.
2. **Bucle infinito** para medir y enviar datos constantemente.
3. **Lee la temperatura** usando `leer_temperatura()`.
4. **Imprime la temperatura** en la consola.
5. **Envía el dato a ThingSpeak**.
6. **Espera 180 segundos (3 minutos)** antes de repetir el proceso.

3. Visualización en ThingSpeak

Esta gráfica muestra los datos obtenidos en tiempo real cada 180s.



A continuación se mostrará el código que se utilizó para configurar el MATLAB Analysis y que se avisará por medio de un correo si la temperatura rebasaba los 35°C.

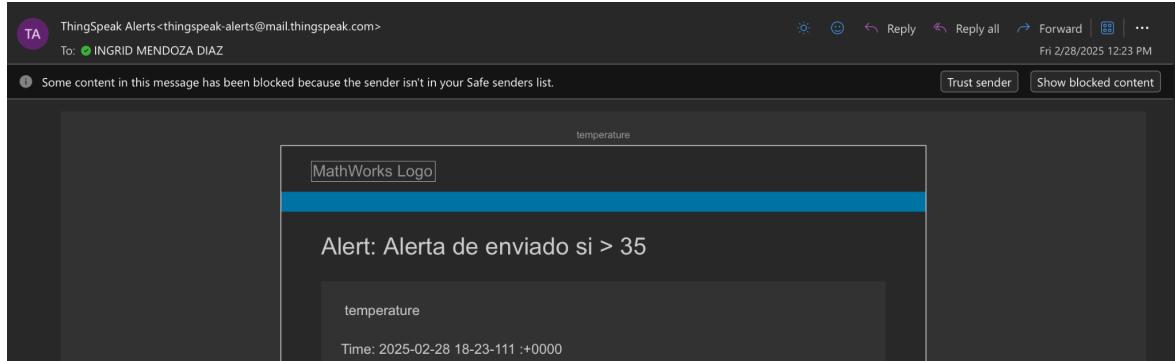
```
% Read the soil moisture channel data from the past two weeks.  
% Send an email and tell the user to add water if the value  
% is in the lowest 10 %.  
  
% Store the channel ID for the moisture sensor channel.  
channelID = 2841591;  
  
% Provide the ThingSpeak alerts API key. All alerts API keys start with  
TAK.  
alertApiKey = 'TAK0++7f+ejqnAyAvkQ';  
  
% Set the address for the HTTP call  
alertUrl="https://api.thingspeak.com/alerts/send";  
  
% webwrite uses weboptions to add required headers. Alerts needs a  
ThingSpeak-Alerts-API-Key header.  
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key",  
alertApiKey]);  
  
% Set the email subject.  
alertSubject = sprintf("Alerta de enviado si > 35");  
  
% Read the recent data.  
  
% Check to make sure the data was read correctly from the channel.  
  
alertBody = ' temperature';  
  
% Catch errors so the MATLAB code does not disable a TimeControl if it  
fails  
try  
    webwrite(alertUrl , "body", alertBody, "subject", alertSubject,  
options);
```

```

catch someException
    fprintf("Failed to send alert: %s\n", someException.message);
end

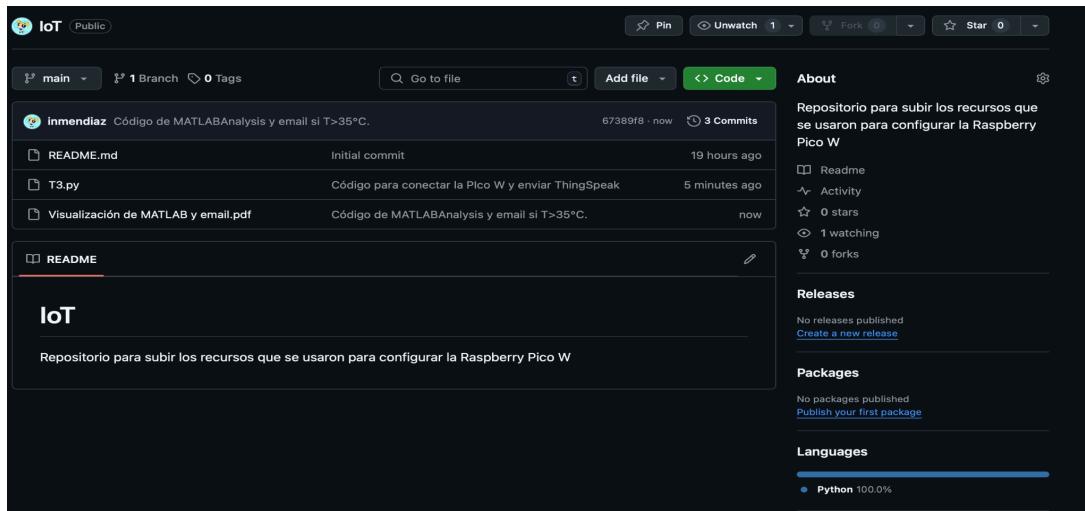
```

Comprobación de que si manda el correo cuando se registra que la temperatura está arriba de los 35°C.



Github

Podrás encontrar todos los códigos que se usaron para configurar todo el proyecto, en el repositorio con la siguiente dirección: <https://github.com/inmendiaz/IoT.git>.



En el Readme podrás visualizar las instrucciones que tienes que seguir para realizar este proyecto si alguna vez lo necesitas, deberías de sólo seguir los pasos, y te saldría de manera exitosa lo que se solicitó en este proyecto.

Conclusión

El desarrollo de este proyecto permitió la implementación de un sistema IoT funcional para la medición y monitoreo de temperatura en la nube, integrando hardware, programación en MicroPython y plataformas de almacenamiento y análisis de datos.

A lo largo del proceso, se configuró la Raspberry Pi Pico W para capturar la temperatura mediante el sensor LM35, procesar la señal analógica y enviar los datos a ThingSpeak mediante una conexión Wi-Fi. Además, se estableció la visualización de datos en tiempo real mediante gráficos y se implementaron herramientas de MathWorks para calcular promedios y generar alertas ante temperaturas elevadas.

Este proyecto reforzó conocimientos clave en IoT, electrónica, programación en MicroPython, transmisión de datos a la nube y análisis de información en tiempo real. La automatización del monitoreo de temperatura demuestra cómo los sistemas IoT pueden ser utilizados en aplicaciones prácticas como el control ambiental, la domótica y la supervisión de procesos industriales.

En general, este trabajo sirvió como una introducción sólida a las tecnologías IoT, proporcionando experiencia en la integración de hardware y software para el desarrollo de soluciones inteligentes y conectadas.

Reflexión Personal

El desarrollo de este sistema IoT para la medición y monitoreo de temperatura representó una experiencia enriquecedora en la integración de hardware y software para la transmisión de datos en la nube. A lo largo del proceso, adquirí conocimientos clave sobre el uso de la Raspberry Pi

Pico W, la lectura de sensores analógicos con el LM35, y la programación en MicroPython para procesar y enviar datos a ThingSpeak mediante una conexión Wi-Fi.

Además, comprendí la importancia de la visualización y análisis de datos en la nube, utilizando herramientas como MathWorks para calcular promedios y generar alertas ante temperaturas elevadas. Esta experiencia me permitió reforzar mis habilidades en IoT, electrónica, conectividad inalámbrica y manejo de plataformas en la nube.

Si bien el sistema desarrollado cumple con su función principal, hay varias áreas en las que se podría mejorar:

- Optimización del código: Se podrían implementar técnicas para reducir el consumo de energía y mejorar la estabilidad de la conexión Wi-Fi.

Mayor precisión en la medición: Usar filtros de software o hardware para reducir el ruido en la lectura del sensor.