Instituto Tecnológico Superior de Huetamo

Ingeniería en Sistemas Computacionales

LISTA DE COTEJO PARA EVALUACIÓN DE INVESTIGACION DOCUMENTAL

Materia:	**Sistemas programables**	**Semestre**	**7mo**
Nombre del docente:	**Ing. Manuel Loeza González**	**Fecha**	**25/11/2024**
Nombre del alumno:	Efraid Jired Bahena, Haydee Barrera	**No. Control**	21070004,21070006
Evidencia:	**Reporte de investigación**		
Tema/Unidad;	**Puertos y buses de comunicación/Unidad 5**		

INSTRUCCIONES

1.-El docente llenara la rúbrica en función del desempeño mostrado por el (la/las/los) estudiante (es).
2.-Se escribirá "SI" o "NO" de acuerdo al desempeño de del aspecto a evaluar.
3.-Se marca el puntaje obtenido que corresponda con el desempeño dentro de "puntos" emitiendo retroalimentación.
4.-Deberá colocar el puntaje de cada ítem y realizar la suma.

No.	Indicador	Descripción	VALOR	IND	SI / NO	PUNTOS
1	**Formato**	Cumple con las partes y orden del formato para entrega de trabajos además de limpieza y entrega en tiempo y forma.	2	D	Si	2
2	**Índice**	Cuenta con un listado del contenido de información que forman el documento y esta paginado.	3		Si	3
3	**Introducción**	Cuenta con un escrito a manera de introducción del tema de que trata la investigación, mínimo media cuartilla.	5		Si	5
4	**Definiciones**	Cuenta con un índice de términos con sus definiciones o las especifica a lo largo del documento.	5	F	Si	5
5	**Desarrollo**	Incluye la información respectiva de los temas solicitados de forma clara, completa y bien organizada.	6	A	Si	6
6	**Conclusiones**	Presenta conclusiones congruentes y claras.	5	C	Si	5
7	**Referencias Bibliográficas**	Incluye bibliografía utilizada en formato APA	2		Si	2
8	**Ortografía**	No contiene faltas de ortografía.	2		Si	2
PUNTAJE MÁXIMO:		30	PUNTAJE OBTENIDO:			30

Retroalimentación:

Ing. Manuel Loeza González

Efraid Jired Bahena Cardenas

Haydee Barrera Santacruz

INSTITUTO TECNOLÓGICO SUPERIOR DE HUETAMO

INGENIERÍA EN SISTEMAS COMPUTACIONALES.

MATERIA:

SISTEMAS PROGRAMABLES.

REPORTE DE INVESTIGACIÓN.

TEMA:

PUERTOS Y BUSES DE COMUNICACIÓN PARA MICROCONTROLADORES.

DOCENTE:

ING. MANUEL LOEZA GONZALEZ.

ALUMNAS:

EFRAID JIRED BAHENA CARDENAS	21070004
HAYDEE BARRERA SANTACRUZ	21070006

LUGAR Y FECHA:

HUETAMO DE NUÑEZ A 25 DE NOVIEMBRE DEL 2024.

ÍNDICE

INTRODUCCIÓN

En este proyecto, se conectará el microcontrolador ESP32 a una red WiFi mediante MicroPython. El código cargado en el ESP32, a través de Thonny, establecerá las reglas de operación para dos sensores: uno de temperatura y otro de distancia. Estos sensores estarán monitoreando continuamente las condiciones del entorno, y el ESP32 enviará la información recopilada al Jupyter Notebook para su análisis y visualización.

Además, el ESP32 actuará como un servidor web, recibiendo comandos desde la computadora o dispositivo remoto y enviando respuestas con los datos que recolecta. Los dispositivos conectados al ESP32, como el motor paso a paso, el servomotor y la pantalla OLED, se controlarán de manera remota, permitiendo que se tomen decisiones y ajustes en tiempo real según la información de los sensores. La pantalla OLED, conectada al ESP32, mostrará información relevante basada en las condiciones de los sensores, proporcionando una interfaz visual de los datos y el estado de los dispositivos controlados.

En resumen, este proyecto permite la interacción remota con el ESP32 para la supervisión y control de dispositivos basados en datos de temperatura y distancia, utilizando Jupyter Notebook para visualizar y gestionar la información de manera eficiente.

CONTENIDO

Los **puertos** y **buses de comunicación** son componentes esenciales para que un microcontrolador interactúe con otros dispositivos

Puertos de Comunicación:

Son puntos de entrada/salida que permiten la conexión física y la transmisión de datos entre el microcontrolador y otros dispositivos.

Puertos GPIO (General Purpose Input/Output):

Funcionan como pines digitales configurables.

Usos: controlar LEDs, botones, motores, sensores, etc.

Ejemplo: en un ESP32, los pines GPIO pueden ser configurados como entrada o salida según se requiera.

Puertos analógicos (ADC/DAC):

ADC (Analog-to-Digital Converter): Convierte señales analógicas (voltajes) en valores digitales.

DAC (Digital-to-Analog Converter): Convierte valores digitales a señales analógicas (menos común).

Usos: leer sensores analógicos (temperatura, luz) o generar señales de audio.

Conexión WiFi y Servidor Web en el ESP32.

Conexión a WiFi:

El ESP32 se conecta a una red WiFi, permitiendo que otros dispositivos (como tu computadora o teléfono) puedan comunicarse con él a través de internet.

Servidor Web:

El ESP32 actúa como servidor web, donde puede recibir comandos desde un navegador o aplicación remota. Estos comandos pueden incluir peticiones de datos (como la temperatura o distancia) o instrucciones para controlar dispositivos conectados al ESP32, como motores y pantallas.

El ESP32 puede enviar respuestas con los datos recopilados por los sensores, que luego se pueden procesar y visualizar en Jupyter Notebook.

Interfaz de Visualización en Jupyter Notebook:

Jupyter Notebook:

Es una herramienta interactiva que permite ejecutar código en Python y mostrar resultados en tiempo real, como gráficos y datos tabulados.

En este caso jupyter interactua para visualizar la información que el ESP32 envía, como la temperatura y la distancia medida por los sensores.

Código Thonny.

El archivo main.py es el que el sistema de **MicroPython** busca y ejecuta automáticamente al iniciar el microcontrolador. Esto lo convierte en el lugar ideal para incluir el código principal del proyecto.

Este código en Thonny responde a una solicitud HTTP para obtener la temperatura, leyendo el valor desde un sensor DS18B20, ajustando el ángulo de un servomotor en función de esa temperatura y controlando un motor paso a paso de acuerdo a los límites de temperatura. Además, muestra la temperatura, el ángulo y el estado del motor en una pantalla OLED, y envía esta información como respuesta al cliente.

```
elif "GET /temperatura" in request:
    ds_sensor.convert_temp()
    temperatura = ds_sensor.read_temp(roms[0])
    b = int((temperatura - 2) * (98 / 250) + 22)
    servo.duty(b)

    motor_estado = ""
    if temperatura <= 15:
        pasos(512, True)
        oled.text("Motor: A Reloj", 0, 50)
        motor_estado = "A reloj"
    elif temperatura >= 30:
        pasos(512, False)
        oled.text("Motor: C. Reloj", 0, 50)
        motor_estado = "C. reloj"
    else:
        oled.text("Motor: STOP", 0, 50)
        motor_estado = "STOP"

    oled.text("Temperatura:", 0, 0)
    oled.text(f"{temperatura:.2f} C", 0, 10)
    oled.text(f"Angulo Servo: {b}", 0, 20)
    oled.show()
    oled.fill(0)

    response = f"Temperatura: {temperatura:.2f} C"
    response = f"Temperatura: {temperatura:.2f} C\nÁngulo Servo: {b}\nMotor: {motor_estado}"
```

Aquí se controla un motor paso a paso y actualiza una pantalla OLED según la distancia medida por el sensor y las condiciones que se le establecen. Cambia la dirección del motor o lo detiene según el valor de la distancia, muestra esta información junto con el ángulo del servomotor en la pantalla, y prepara una respuesta con esos datos para enviarla al cliente.

```
  # Control del motor paso a paso
motor_estado = ""
if distancia <= 5:
    pasos(512, True)
    oled.text("Motor: A Reloj", 0, 50)
    motor_estado = "A reloj"
elif distancia >= 15:
    pasos(512, False)
    oled.text("Motor: C. Reloj", 0, 50)
    motor_estado = "C. reloj"
else:
    oled.text("Motor: STOP", 0, 50)
    motor_estado = "STOP"

oled.text("Distancia:", 0, 0)
oled.text(f"{distancia:.2f} cm", 0, 10)
oled.text(f"Angulo Servo: {b}", 0, 20)
oled.show()
oled.fill(0)

response = f"Distancia: {distancia:.2f} cm\nÁngulo Servo: {b}\nMotor: {motor_estado}"
```

Código Jupyter.

Función para manejar el botón y obtener la distancia del sensor y luego muestra el resultado en una ventana de la interfaz de usuario.

```
def obtener_distancia(_):
    resultado = enviar_comando("distancia")
    mostrar_ventana("Datos del Sensor", resultado)
```

Función para manejar el botón y obtener la temperatura del sensor y luego muestra el resultado en una ventana de la interfaz de usuario.

```
def obtener_temperatura(_):
    resultado = enviar_comando("temperatura")
    mostrar_ventana("Datos del Sensor", resultado)
```

Crea dos botones interactivos que, al ser clickeados, ejecutan las funciones obtener_distancia y obtener_temperatura, respectivamente, para obtener y mostrar la distancia o la temperatura del sensor.

```
boton_distancia = widgets.Button(description="Obtener Distancia", button_style="primary")
```

```
boton_distancia.on_click(obtener_distancia)

boton_temperatura = widgets.Button(description="Obtener Temperatura",
button_style="success")

boton_temperatura.on_click(obtener_temperatura)
```

Aquí se crea un botón llamado "Cerrar" con un estilo de color rojo, y al hacer clic en él, ejecuta la función cerrar_ventana. Luego, agrega este botón al contenido de un popup o ventana emergente.

```
boton_cerrar = widgets.Button(description="Cerrar", button_style="danger")

boton_cerrar.on_click(cerrar_ventana)

popup.children += (boton_cerrar,)
```

Este código muestra la interfaz con los botones.

```
display(widgets.HBox([boton_distancia, boton_temperatura]))

display(popup)
```

CONCLUSIÓN

Este proyecto muestra cómo usar un microcontrolador ESP32 conectado a WiFi para trabajar con sensores y motores, controlándolos de manera remota. Gracias a MicroPython y Jupyter Notebook, logramos leer datos como temperatura y distancia, y enviar comandos para que el ESP32 realice acciones, como mover el motor paso a paso, servomotor y mostrar información en la pantalla OLED.

El ESP32 funciona como un servidor que conecta el mundo físico con la computadora, permitiendo un control en tiempo real. Este sistema es muy útil para proyectos simples de automatización o dispositivos inteligentes.

BIBLIOGRAFÍA

Hernandez, D. (07 de abril de 2019). *MICROPYTHON ESP32 – Conexión WiFi – STATION*. Obtenido de esploradores.com: https://www.esploradores.com/micropython_conexwifi_station/

Pulido, R. U. (08 de febrero de 2022). ESP32: Configurar Wifi con MicroPython. Obtenido de *maquinasvirtuales.eu*: https://www.maquinasvirtuales.eu/esp32-configurar-wifi-con-micropython/