UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS
PROYECTOS DIGITALES AVANZADOS

PRÁCTICA N°8: TRANSFERENCIA ENTRE COMPUTADORA Y PICO W

Profesor:

Pedro Rene
Cabrera

Bachiller:

Claudia Rodríguez
C.I: 27.943.668

Barcelona, diciembre de 2024

1. OBJETIVO DE LA PRÁCTICA

1.1. Investigar por cual método enviar la información del Pico a la computadora y de la computadora al Pico, y descargar la librería pertinente para manipular el puerto serial de la computadora.

1.2. Enviar un mensaje desde la computadora y recibirlo en el Pico.

1.3. Enviar un mensaje desde el Pico y recibirlo en la computadora.

1.4. Agregados:

1.4.1. Enviar archivos de computadora al Pico W y del Pico W a la computadora

1.4.2. Utilizar el modulo SD para los archivos que se encuentran en la memoria SD enviarlos a la computadora y también almacenarlos en la memoria SD.

1.4.3. Utilizar la pantalla OLED para mostrar la información recibida y enviada a través del Pico W.

2. DESARROLLO

2.1. Planteamiento del Problema

Anteriormente, se realizaron prácticas de envió y recepción de datos vía alámbrica e inalambricamente, pero eran operaciones realizadas entre Raspberry Pi Picos W. En esta ocasión, se esta enviando información de manera alámbrica pero desde el Raspberry Pi Pico W a la Computadora y viceversa. Debido a que no son el mismo tipo de dispositivo, ya la computadora posee un protocolo de comunicación establecido, por lo tanto, se tiene que recurrir a librerías externas para así poder manipular los puertos donde se conecta el Raspberry Pi Pico W. Para ello, se tuvo que investigar que complemento era necesario para así poder realizar la comunicación. Con respecto a la comunicación por parte del Pico W se podía reusar código de la práctica de UART. Por motivos de facilidad, la comunicación se debe de realizar de manera serial siguiendo así las normativas pertinentes y estableciendo tiempos de descanso para así enviar o recibir la información de manera completa y correcta.

2.2. SOLUCIÓN

2.2.1. Instalación de la librería PySerial: Instalar el pip python. Para ello, se siguieron los siguientes pasos:

1- Acudir a la página oficial para descargar el archivo: pyserial-3.5.tar.gz

2- Abrir el cmd y ejecutar las siguientes lineas:

python --m ensurepip --upgrade

python --m pip install --upgrade pip

python --m pip install setuptools

3- Una vez hecho el paso anterior y descargado el archivo, extraer la carpeta y buscar el archivo denominado setup.

4- Ejecutar dicho archivo desde el CMD con la siguiente línea: python setup.py install

5- Ya con lo anterior, se encuentra instalado la librería pyserial.

6- Ya los siguientes pasos es la realización del código y su solución, el cual, se encuentra explicado en el siguiente apartado.

2.2.2. Envío de datos desde la computadora al Pico: El envío de datos desde la computadora al Pico ocurre de la siguiente manera, (código de transferencia de la computadora al Pico):

1- Determinar cual es el puerto que usa el Pico W al momento de conectarse con la computadora.

2- Establecer parámetros de velocidad y puerto.

3- Ya con la librería instalada **PYSERIAL** crear el objeto que manipulará la conexión entre computadora y pico.

4- Almacenar en una variable de tipo cadena el mensaje que se va a enviar al Pico. Este mensaje esta almacenado en un archivo en la computadora anfitrión: "C:/Users/Claudia/Desktop/datos_pico.csv"

5- Mostrar en la pantalla de la computadora el mensaje que se va a enviar y posteriormente añadir un carácter delimitador (asterisco)

6- Utilizar el objeto creado con el comando **write** y la función **encode** para enviar el mensaje al Pico.

7- Colocar un sleep (de 5 a 10 segundos) para otorgar un tiempo de envió y no confundir la cadena que se va a enviar

8- Por ultimo, cerrar la conexión.

El código anteriormente explicado, se debe de ejecutar desde el CMD. En paralelo, el código que se ejecutará en el Pico W se va iniciando, además se le coloco el nombre de "main.py" con el objetivo de que al conectar el Pico W a la computadora no exista un conflicto en el puerto. El código del Pico W se ejecuta de la siguiente manera:

1. Crear el objeto para manipular la pantalla OLED y esperar a recibir el mensaje.

2. Una vez recibido el mensaje se muestra en el OLED: “mensaje recibido. ”

3. Se procede a recibir el mensaje, el Pico W estará leyendo hasta que encuentre el carácter delimitador que se coloco al final del mensaje en la computadora.

4. En el momento que encuentre el carácter delimitador, almacena la cadena en un archivo denominado “**PC_PICO.csv”**

5. Como ultima acción, muestra en la pantalla OLED el mensaje que se recibió de la computadora y coloca la palabra “listo”.

2.2.3. Envío de datos desde el Pico a la computadora: El envío de datos desde el Pico a la computadora ocurre de la siguiente manera, (código de transferencia del Pico W a la computadora):

1. Se crean los objetos para poder manipular la pantalla OLED y el módulo de la memoria SD.
2. Se lee desde la tarjeta SD el archivo que se va a enviar (“**sd/lm35.txt”**) y se almacena la cadena en una variable de tipo cadena
3. Se invoca a la función “**enviar_info_pico”** y se le pasa de parámetros tanto el mensaje como el objeto para manipular la pantalla OLED
4. Se muestra en la pantalla del OLED el mensaje de “Transmisión de Pico a PC” y se utilizará el Protocolo UART debido a que causo menos problemas en prácticas anteriores.
5. Se muestra en la pantalla OLED “Mensaje a enviar: “ más el mensaje que se paso de parámetro.
6. Se le agrega de igual forma un carácter delimitador, es decir, el asterisco.
7. Se inicializa una variable bool denominada **enviado** que se usará en el bucle while, que hasta que no se envíe, esta no pasa a ser verdadera.
8. Por último, se muestra en la pantalla OLED “Mensaje enviado a la PC”

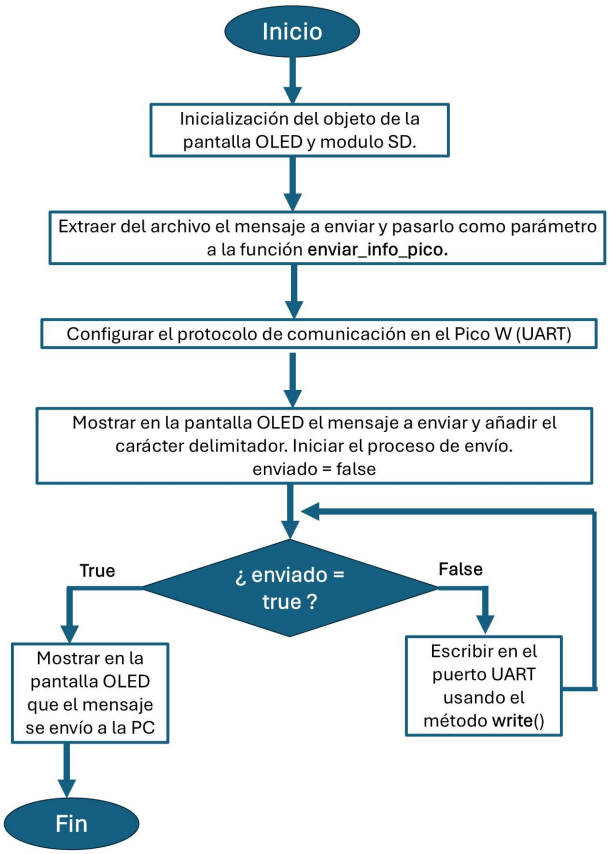
De la misma forma que se ejecuto el código cuando se realizo el envío de datos desde la computadora al Pico, se hace en esta transferencia**.** El archivo del código en el Pico se almacena como “main.py” para que cuando se conecte el Pico a la Computadora, se ejecute instantáneamente y no exista conflicto con el puerto COM. El código en la computadora se ejecuta de la siguiente manera:

1. Se imprime en pantalla “TRANSFERENCIA PICO A PC” y se establecen los parámetros para el uso del puerto, como la velocidad y cual puerto utiliza la computadora para comunicarse con el Pico W. Ademas, se inicializa una variable bool denominada recibido en False.
2. Observación: en la siguiente línea: `data = serial_connection.read(400)`, el número que se encuentra en paréntesis, es la cantidad de bytes que se van a recibir, este número es ajustable, debido a la número de caracteres que se van a enviar desde el Pico a la computadora.
3. Una vez recibido el mensaje, se almacena en una variable de tipo cadena y se separa donde encuentre el asterisco, ya que lo demás que se agrego, no esta incluido en el mensaje original.
4. Como última acción, muestra el mensaje en la pantalla de la computadora y lo almacena también en un archivo en el escritorio en la siguiente ruta: C:/Users/Claudia/Desktop/datos_pico.csv"

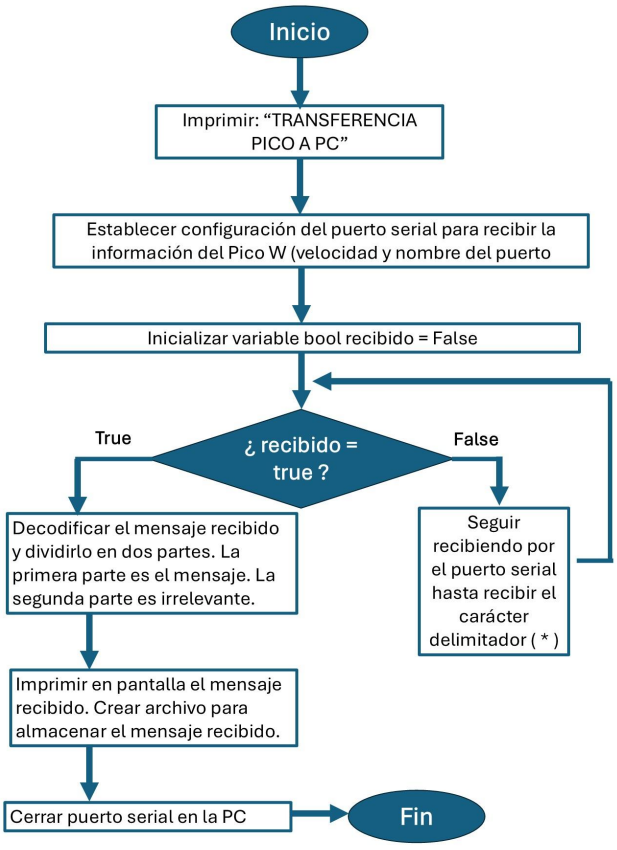
2.3. DIAGRAMAS DE FLUJO

2.3.1. Envío del Raspberry Pi Pico W a la Computadora

Algoritmo Pico W

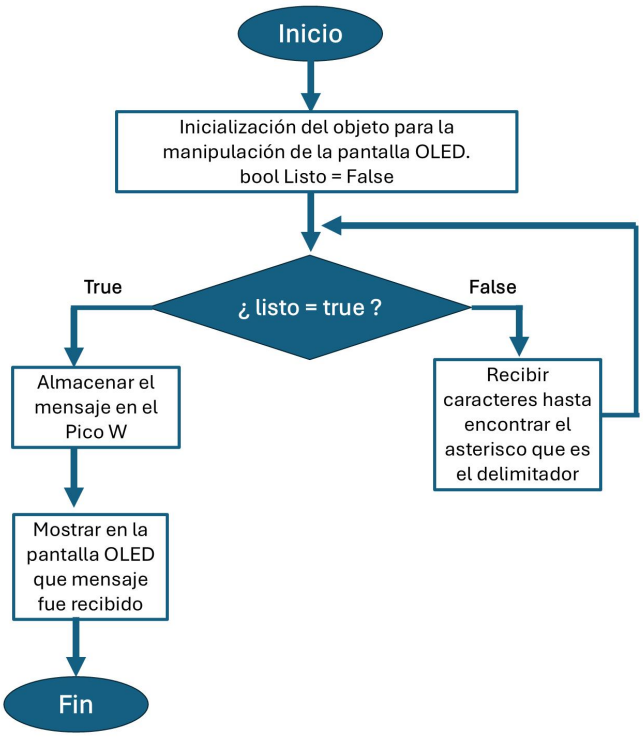


Algoritmo Computadora

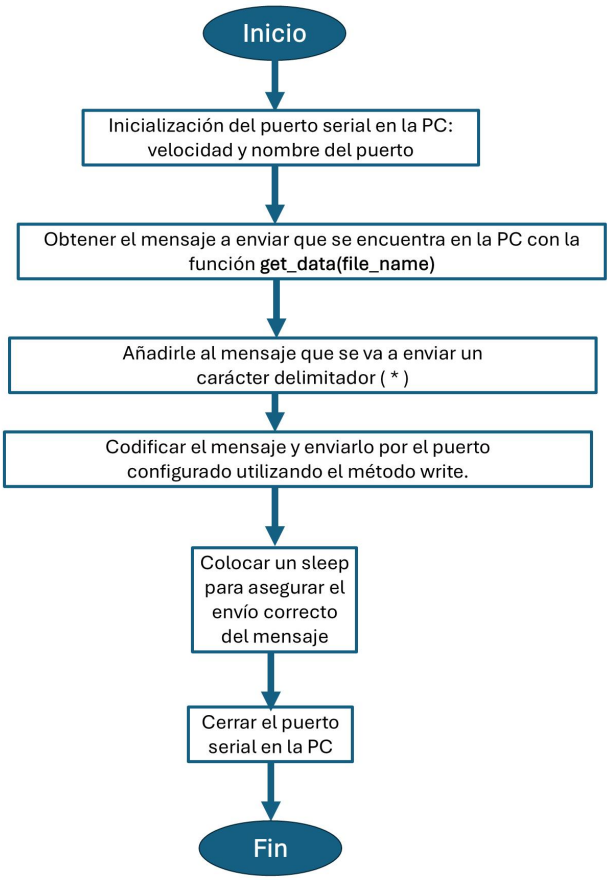


2.3.2. Envío de la Computadora al Raspberry Pi Pico W

Algoritmo Pico W



Algoritmo Computadora



2.4. COMANDOS UTILIZADOS

Librería	Método	Descripción
Serial	serial_connection = serial.Serial(port, baudrate)	Crea el objeto para la manipulación del puerto serial.
	serial_connection.write(cadena .encode())	Write(): Envía al puerto en seria la información en bytes Encode(): Convierte un string en bytes.
	serial_connection.close()	Close(): Cierra la conexión serial
	serial_connection.read(400)	Read(int n): Lee información del puerto serial. Su argumento es un número entero que representa el número máximo de bytes que se leerán.

Librería	Método	Descripción
Uselect y stdin	uselect.select([stdin], [], [], 0) stdin: Indica que se debe comprobar si hay datos disponibles para lectura []`, []: Listas vacías para escritura y excepciones, respectivamente. 0: Indica un timeout de 0 segundos.	Es una función que realiza una operación de selección I/O múltiple
stdin	stdin.read(1)	Lee un solo byte (1)
machine	uart = machine.UART(0, baudrate=115200)	Crea el objeto de tipo UART para su posterior manipulaión
	uos.dupterm(uart)	Para monitoreo de errores, se utiliza para imprimir que errores se dan con el puerto UART.
	uart.write(frase)	Envía información al puerto UART

3. ANEXOS

3.1. Códigos

3.1.1. Envío del Raspberry Pi Pico W a la Computadora

Código Pico W	Código Computadora

import machine
import time
from machine **import** Pin, UART, I2C
import uos
from ssd1306 **import** SSD1306_I2C
from time **import** sleep
import utime
import sdcard
import uos
 	
def **microsd():**# <-inicializacion microsd
cs = machine.Pin(1, machine.Pin.OUT)
spi = machine.SPI(0, baudrate=1000000, polarity=0, phase=0, bits=8, firstbit=machine.SPI.MSB, sck=machine.Pin(2), mosi=machine.Pin(3),
	import serial
	
	def **get_data(file_name)**:
	with open(file_name, 'r') as file:
	data = file.read()
	return data
	
	def **recibir_info_pc():**
	print("TRANSFERENCIA PICO A PC")
	port = "COM6" #
	configuracion puerto COM
	baudrate = 115200 # velocidad
	serial_connection = serial.Serial(port, baudrate) # crear objeto
	print(f"Configurando parametros..Puerto:{port} Velocidad: {baudrate}")

```
miso=machine.Pin(4))
    sd = sdcard.SDCard(spi, cs)
    vfs = uos.VfsFat(sd)
    uos.mount(vfs, "/sd")
    return sd                    # retorna sd

def pantalla():
        # <-inicializacion de oled
    WIDTH = 128
    HEIGHT = 64
    i2c = I2C(0, scl=Pin(17), sda=Pin(16), freq=200000)
    oled = SSD1306_I2C(WIDTH, HEIGHT, i2c)
    return oled

def mostrar_oled(oled, message):
    oled.fill(0)
    ancho_caracter = 7
    max_columna = 120
    fila = 0
    columna = 0
    palabras = message.split()
    for palabra in palabras:
        ancho_palabra = len(palabra) * ancho_caracter
        if columna + ancho_palabra > max_columna:
            fila += 16
            columna = 0
        if fila >= 50:
            oled.show()
            sleep(1)
            oled.fill(0)
            fila = 0
            columna = 0
        oled.text(palabra, columna, fila)
        columna = columna + 7
        columna += ancho_palabra + ancho_caracter
    oled.show()
    sleep(5)

def enviar_info_pico(oled, data):
    cadena = "TRANSMISION DE PICO A PC"
    mostrar_oled(oled, cadena)
    uart = machine.UART(0, baudrate=115200)
# uart del pico
    uart.init(115200, bits=8, parity=None, stop=1, tx=Pin(0), rx=Pin(1))
    recibido = False             # bandera para detener bucle while
    while not recibido:
        data = serial_connection.read(400)
        if data == b"EOF":
            break
        #print(data)
        recibido = True

    # recibir info y dividir la parte que nos importa
    data = data.decode('utf-8')
    pos = data.find('*')
    if pos != -1:
        parte_filtrada = data[:pos]
    else:
        parte_filtrada = data  # si no eencueentra asterisco, mantener la cadena original
    partes = data.split('*')
    data = partes[0]
    print("Mensaje recibido del pico: ", data)

    # ruta donde va a guardar el archivo
    with open("C:/Users/Claudia/Desktop/datos_pico.csv", "wb") as destination_file:
        destination_file.write(partes[0].encode())
    serial_connection.close() # cierra conex

def main():
    recibir_info_pc()

if __name__ == "__main__":
    main()
```

```
    uos.dupterm(uart)
    cadena = "Mensaje a enviar: " + data
    mostrar_oled(oled, cadena)
    frase = cadena + "*"
    enviado = False
    while not enviado:
        uart.write(frase)  # ... envia frase
        print(frase.strip())    # <- verificas que
frase envias
        enviado = True
        break
    cadena = "Mensaje enviado a PC"
    mostrar_oled(oled, cadena)

def get_data(file_name): # obtener info del
archivo
    with open(file_name, 'r') as file:
        data = file.read()
    return data

def main():
    sd = microsd()
    oled = pantalla()
    with open("/sd/lm35.txt", "r") as file:
        data = file.read()
    enviar_info_pico(oled, data)

if __name__ == "__main__":     main()
```

3.1.2. Envío de la Computadora al Raspberry Pi Pico W

Código Pico W	Código Computadora

import time
from sys **import** stdin
import uselect
from machine **import** Pin, I2C
from ssd1306 **import** SSD1306_I2C
from time **import** sleep
import utime
 	
csv_filename = "data.csv"
 	
def **pantalla():**
WIDTH = 128
HEIGHT = 64
i2c = I2C(0, scl=Pin(17), sda=Pin(16), freq=200000)
oled = SSD1306_I2C(WIDTH, HEIGHT, i2c)
	import serial
	import time
	
	def **get_data(file_name):** # obt mensaje
	with open(file_name, 'r') as file:
	data = file.read()
	return data
	
	def **main():**
	port = "COM6" # acuerdate revisarlo si usas el otro pico
	baudrate = 115200
	serial_connection = serial.Serial(port, baudrate)
	cadena = get_data("C:/Users/Claudia/Desktop/datos_pico.csv")
	print("Mensaje a enviar: ", cadena)
	cadena = cadena + " *" # añades el caracter

```
    return oled

def mostrar_oled(oled, message):
    oled.fill(0)
    ancho_caracter = 7
    max_columna = 120
    fila = 0
    columna = 0
    palabras = message.split()
    for palabra in palabras:
        ancho_palabra = len(palabra) * ancho_caracter
        if columna + ancho_palabra > max_columna:
            fila += 16
            columna = 0
        if fila >= 50:
            oled.show()
            sleep(1)
            oled.fill(0)
            fila = 0
            columna = 0
        oled.text(palabra, columna, fila)
        columna = columna + 7
        columna += ancho_palabra + ancho_caracter
    oled.show()
    sleep(5)

def save_to_csv(data):
    with open(csv_filename, "w") as f:
        f.write(data + "\n")

def main():
    oled = pantalla()
    mostrar_oled(oled, "Mensaje recibido: ")
    listo = False
    while not listo:
        select_result = uselect.select([stdin], [], [], 0)
        buffer = ''
        while select_result[0]:
            input_character = stdin.read(1)
            # caracter delimitador d ela frase = *
            if input_character == '*':
                # si lo encuentra, entonces ya guarda en el archivo
delimitador
    serial_connection.write(cadena.encode())  # codificas y envias
    # colocas tiempos de esperar por si acaso
    time.sleep(0.01)
    time.sleep(10)
    # cierras conexion
    serial_connection.close()

if __name__ == "__main__":
    main()
```

```
            if buffer:
                save_to_csv(buffer)
                mostrar_oled(oled,
buffer)
                listo = True
            #buffer = ''   # Limpiar el
buffer
            break

        buffer += input_character # va
contenando en buffer
                        # hasta que
sea necesario
        select_result                =
uselect.select([stdin], [], [], 0)
    mostrar_oled(oled, "Listo")


if __name__ == "__main__":
    main()
```

3.2. Bibliografía

pyserial 3.5. Enlace: https://pypi.org/project/pyserial/

Instalación de pySerial en python 3. Enlace: https://www.youtube.com/watch?v=dYeaaW2NvHY

How to Transfer Data/Files from Raspberry Pi Pico to Local Computer (Programmatically) - Part 1. Enlace: https://www.youtube.com/watch?v=OfJ5Y1FlW94&t=7s

How to Transfer Data/Files from Local Computer to Raspberry Pi Pico (Programmatically) - Part 2. Enlace: https://www.youtube.com/watch?v=mi9UT7fTDZw&t=688s

3.3. Diagrama circuital