Laboratorio de Estructuras Computacionales

Laboratorio 2

Profesor: Samuel Andres Cifuentes

Monitor: Tomas Jimenez Alvarez

Objetivos

- Comprender y medir el fenómeno del rebote en teclados de membrana, identificando el tiempo de rebote y sus oscilaciones al presionar las teclas.
- Proponer y evaluar técnicas de filtrado o debounce para mitigar el efecto del rebote.
- Explorar cómo la velocidad del reloj del bus I2C afecta los tiempos de comunicación.
- Evaluar y comprender el tiempo de comunicación I2C entre un microcontrolador y una pantalla SSD.
- Comparar y contrastar los paquetes de bytes transmitidos por UART y Telnet utilizando el módulo ESP8266 a través del protocolo WiFi.
- Analizar la sobrecarga de datos y la latencia en cada protocolo de comunicación.

Consulta previa

1. ¿En qué consiste el rebote y qué técnicas o circuitos de antirebote existen para mitigarlo?
2. ¿Cuáles son las principales diferencias entre los protocolos I2C y UART?
3. Realice un diagrama de bloques de cómo se transmiten los bytes desde un microcontrolador hasta hasta el protocolo WiFi?

Equipos a utilizar

- Tarjeta Núcleo-L476RG.
- Analizador lógico.
- Teclado de membrana.
- SSD1306.
- Protoboard.
- ESP8266.
- Capacitores (antirebote).
- Resistencias (antirebote).
- Cables para conexiones.

Procedimiento

Realizar la siguiente práctica siguiendo el debido procedimiento paso a paso en cada punto del laboratorio.

1. **Medición de oscilaciones con el teclado:** El efecto de rebote en switches al presionarlos, a menudo, suelen ser muy cansinos e incluso pueden llegar a perjudicar una aplicación. Es por esto, que en esta sección se identificarán este tipo de anormalidades, y así, aplicar una correcta implementación para este tipo de problemas.

   Como primer paso debemos iniciar un proyecto en el STM32CubeIDE:

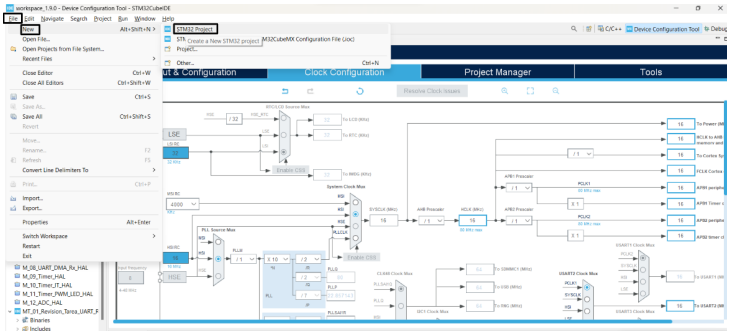

Figure 1: Iniciar proyecto en STM32CubeIDE.

Ahora debemos seleccionar la tarjeta que vamos a utilizar, que para nuestro caso será la tarjeta NUCLEO-L476RG:

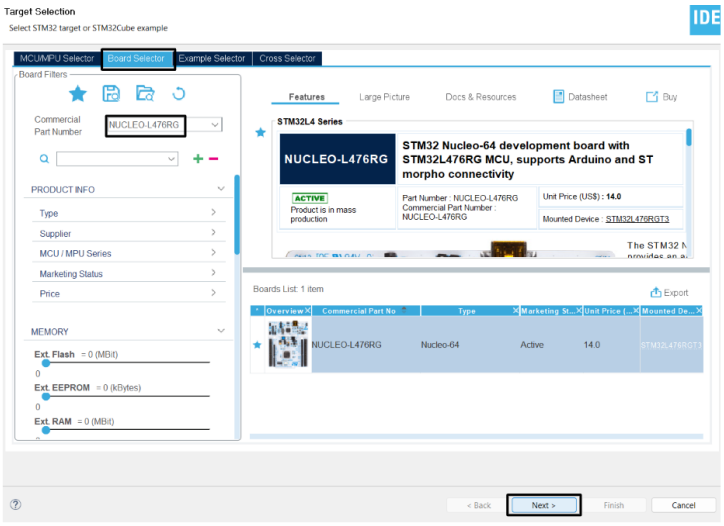

Figure 2: Selección de tarjeta.

Seleccionada la tarjeta asignar un nombre al proyecto, escoger el lenguaje en C, el tipo de proyecto objetivo como STM32Cube e inicializar todos los periféricos al seleccionar ”Finish”:

¡Listo! Deberíamos estar viendo la interfaz gráfica de STM32CubeIDE que nos ayudará a configurar las opciones que necesitamos.

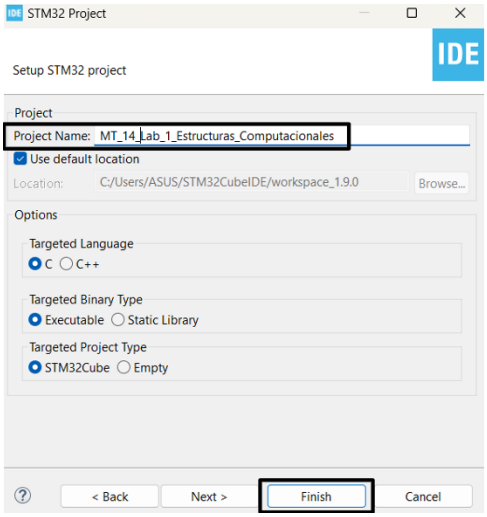

Figure 3: Configuración del proyecto.

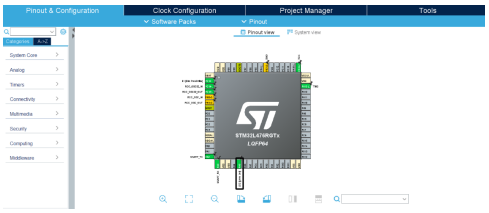

Figure 4: Interfaz gráfica de STM32.

Para esta aplicación se deben inicializar 4 GPIOs de entrada en modo interrupción que corresponderán a las columnas del teclado y, por otro lado, 4 salidas que serán las filas en el teclado.

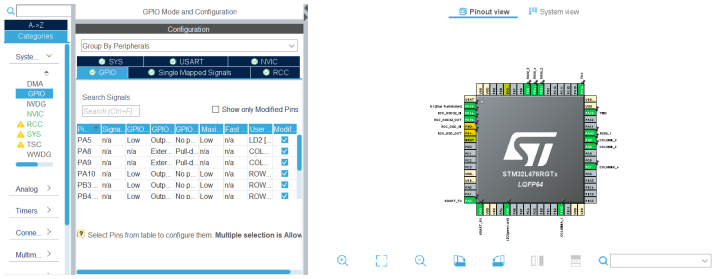

Figure 5: Configuración de puertos.

Por último, generaremos el código con las configuraciones realizadas por la HAL presionando sobre el ”Device Configuration Tool Code Generation”

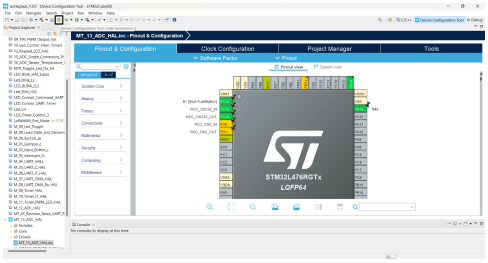

Figure 6: Generación de código.

Con cada uno de los pasos completados con éxito, se debió generar una plantilla de código en la que se debe lograr programar la captación de una tecla basándose en el siguiente repositorio https://github.com/Tjimenez1303/Monitoria_estructuras_Computacionales/tree/main/M_15_Membrane_Keyboard_HAL.

Una vez realizada la programación del teclado, asegúrese de estar recibiendo debidamente cada una de las teclas.

Figure 7: Impresión de teclas mediante UART.

Con la aplicación funcionando, es hora de ocasionar el rebote, y así medir sus oscilaciones. Comente el ”antirebote” por software que se encuentra en la función de ”keypad handler”.

```
/**
 * @brief  This function debounces and identify keypad events.
 * @param  column_to_evaluate: the column where the event happened.
 * @retval 0xFF -> invalid key. [0x00 - 0x0F] -> valid key.
 */
uint8_t keypad_handler(uint16_t column_to_evaluate)
{
    // Se inicializa como valores de error 0xFF -> 255
    uint8_t key_pressed = 0xFF; // Value to return
    uint8_t row = 0xFF;

    /*** Debounce the key press (remove noise in the key) ***/
//  #define KEY_DEBOUNCE_MS 0 /*!> Minimum time required for since last press */
//
//
//  static uint32_t last_pressed_tick = 0;
//  if (HAL_GetTick() <= (last_pressed_tick + KEY_DEBOUNCE_MS)) {
//      // less than KEY_DEBOUNCE_MS since last press. Probably noise
//      return key_pressed; // return 0xFF
//  }
//  last_pressed_tick = HAL_GetTick();

    /*** Check in which column the event happened ***/
    switch (column_to_evaluate) {
    case COLUMN_1_Pin:
        row = keypad_validate_column(COLUMN_1_GPIO_Port, COLUMN_1_Pin);
        if (row != 0xFF) {
            return keypad_map[row][0];
        }
      break;

    case COLUMN_2_Pin:
```

Figure 8: Función de Keypad Handler.

Una vez comentada, conecte a su analizador lógico a una fila,pero dejando conectada esa fila a la tarjeta, de la siguiente manera:

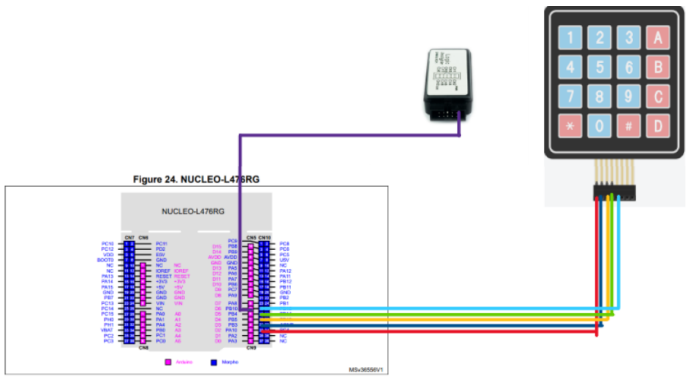

Figure 9: Conexión analizador lógico y Keypad.

¡Listo! Ahora se debe observar el rebote que se ocasiona al pulsar una tecla. Se debe abrir el programa *Pulse View* y conectar el analizador lógico con el programa (en caso de no tenerlo conectado). Para conocer el estado en que se encuentra, se debe verificar en la parte superior *No Device:* No está conectado,*Saleage Logic:* Conectado.

Figure 10: Conectividad analizador lógico.

En caso de no tener el analizador lógico se debe presionar sobre *No Device* y se desplegará una lista de opciones. Siga los pasos a continuación: 1. Escoger el Driver: fx2lafw (generic driver for FX2 based LAs) (fx2lafw). 2. Escanear los dispositivos configurados con el Driver seleccionado. 3. Escoger el dispositivo: Saleae Logic with 8 channels. 4. OK.

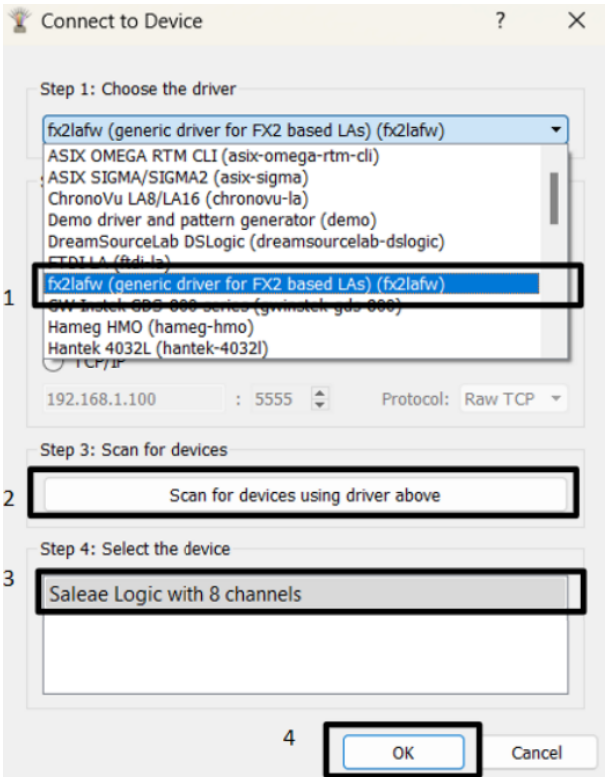

Figure 11: Conexión analizador lógico al Pulse View.

Con el analizador lógico conectado, deberás configurar el número de muestras que tomará y la frecuencia a la que el analizador lógico tomará cada una de estas.

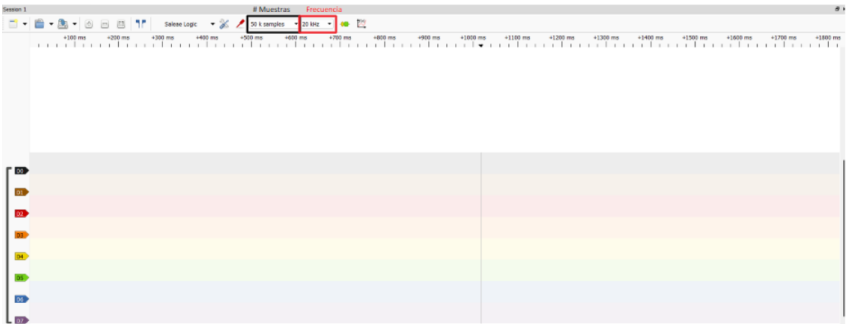

Figure 12: Configuración entorno Pulse View.

Nota: Tenga en cuenta que el antirebote ocurre en ocasiones y en tiempos muy cortos, por lo que deberá configurar una frecuencia de muestreo alta.

Realice las respectivas medidas con las que pueda sustentar en su informe de laboratorio.

Cuando finalice, implemente dos tipos de antirebote: un primer antirebote por software y un segundo por hardware. Realice las medidas necesarias, compare con las medidas tomadas anteriormente y analice cuál es la mejor alternativa.

2. **Medición de tiempo de comunicación I2C en una SSD:** El protocólo I2C (Inter-Integrated Circuit) es una comunicación serial utilizada para conectar microcontroladores, sensores, circuitos integrados y otros dispositivos en un sistema electrónico. para este caso, se utilizará este protocolo para conectar una SSD a la tarjeta Nucleo L476RG y visualizar en pantalla si un usuario tiene acceso o no al ingresar una clave secreta de 5 bytes.

   Al proyecto anteriormente creado para la captación de teclas, descargue la carpeta de drivers del siguiente repositorio: https://github.com/Tjimenez1303/Monitoria_estructuras_Computacionales/tree/main/External_Drivers/SSD1306. Realizado este paso, cree una carpeta fuente en su proyecto:

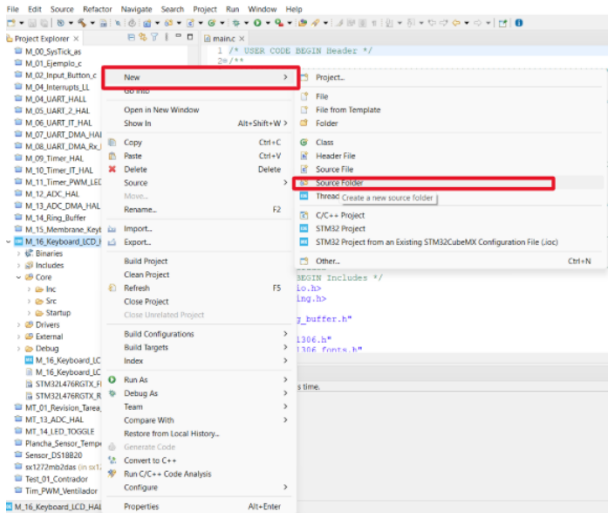

Figure 13: Crear carpeta fuente.

Una vez creada, debe agregar la carpeta al PATH del proyecto para que así, el compilador pueda incluirla al compilar:

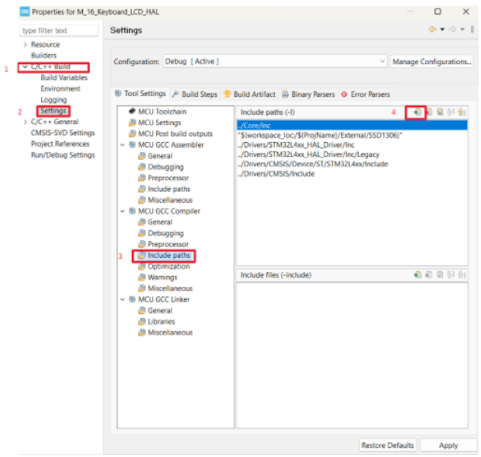

Figure 14: Agregar PATH al proyecto.

Seleccione la carpeta fuente que acaba de agregar al proyecto y aplique los cambios:

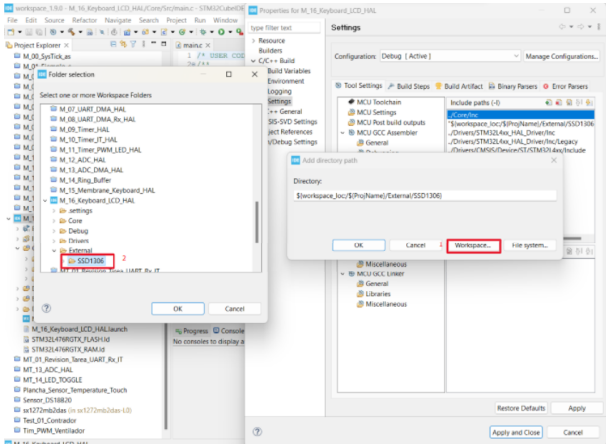

Figure 15: Selección de la carpeta al PATH.

Realizados estos pasos, asegúrese de que el programa está funcionando correctamente y que los paquetes se están enviando a la SSD.

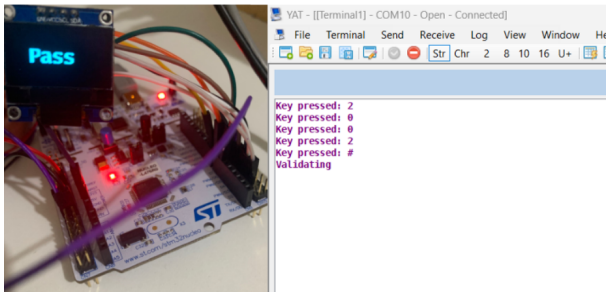

Figure 16: Funcionamiento d el programa.

Seguidamente, deberá abrir el programa *Pulse View* siguiendo los pasos anteriores y, además configurando el protocolo I2C

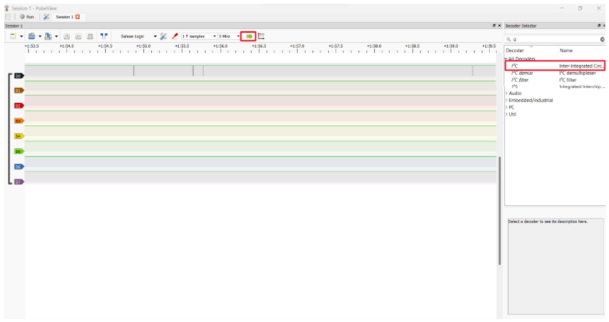

Figure 17: Configuración protocolo Pulse View.

Por último, se deben modificar los parámetros del I2C de la sesión. Esto se realiza pulsando sobre I2C, acto seguido, se desplegará un menú de configuración en donde se deberá ajustar cada uno de los parámetros acorde a los de el entorno STM32CubeIDE.

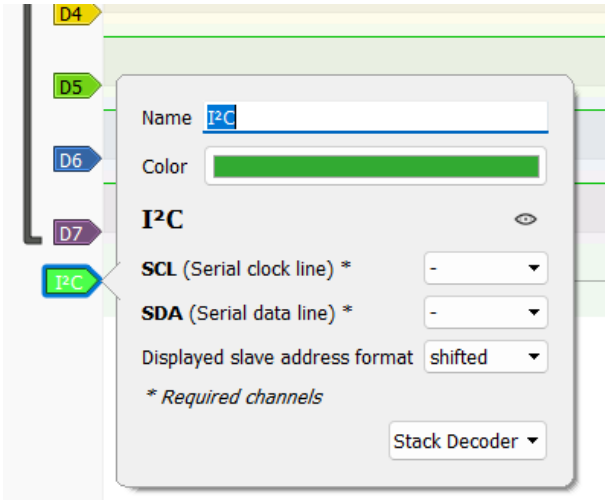

Figure 18: Configuración I2C Pulse View.

Tome el tiempo en que se tarda en enviar un paquete por I2C a la pantalla SSD1306 y analice los resultados.

3. **ESP8266:** Para este ítem, se compararán los paquetes de datos enviados entre UART y Telnet usando ESP8266. Es por esto, que se explorarán y analizarán las diferencias en la transmisión de paquetes de datos entre los protocolos UART y Telnet utilizando el módulo ESP8266 a través de una conexión WiFi. Para el correcto funcionamiento del litreal, deberá seguir los siguientes pasos:

   En primera instancia, acceda al siguiente link: `https://www.espressif.com/en/support/download/other-tools` de la página oficial de Espressif para descargar la herramienta de flasheo.

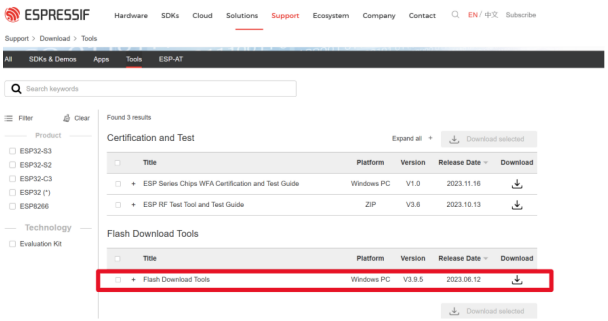

Figure 19: Espressif Tool Flash.

Una vez instlado, debe programar el ESP8266. Para esto acceda al siguiente repositorio de github `https://github.com/jeelabs/esp-link/releases/tag/v2.2.3`, continúe bajando hasta llegar a los Assets (ver figura 20). Una vez allí encontrará 3 archivos y deberá descargar el primero (esp-link-v2.2.3.tgz).

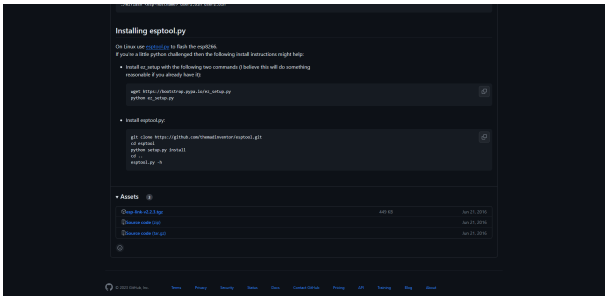

Figure 20: Github Assets.

Ubicados los archivos en su entorno de trabajo, proceda a abrir la herramienta de Espressif y cargue los archivos descargados en el orden en que se observan en la figura 21, además cerciórese de vincular cada archivo al espacio de memoria que se observa y marcar las opciones que se muestran pues de lo contrario la herramienta tendrá un error.

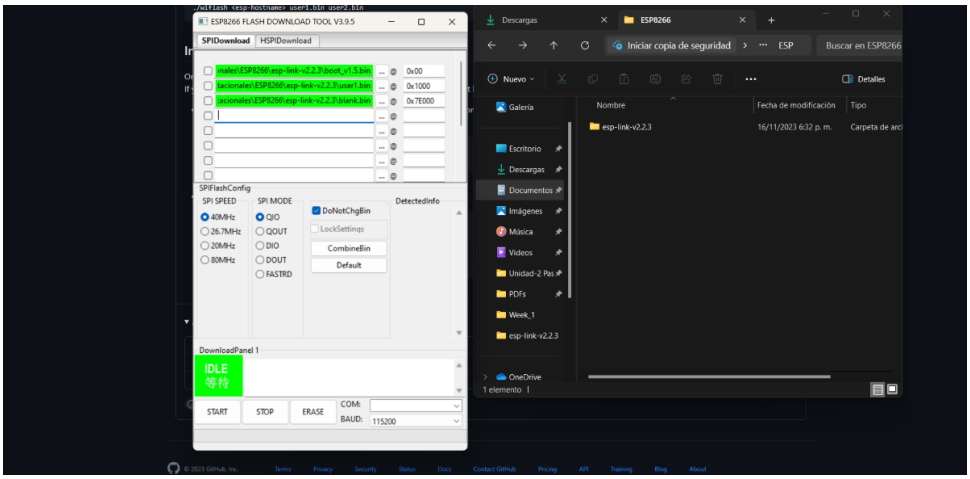

Figure 21: Configuración de archivos ESP8266.

Ahora se configurará el hardware, es así que deberá conectar a su computadora un dispositivo capaz de traducir de USB a UART.

Figure 22: Dispositivo USB a UART.

Conecte el ESP8266 a 3V3 y GND en sus respectivos pines. Además deberá conectar los pines RST e IO0 a GND. Por último, conecte los pines de Rx y Tx de su dispositivo al USB/UART.

Figure 23: Dispositivo USB a UART.

Ahora diríjase a la herramienta de flasheo y presione sobre Start, inmediatamente deberá desconectar IO0 del pin GND para que el dispositivo entre en modo programación y pueda programar los archivos descargados.

Como paso siguiente, deberá conectarse a la red de su ESP y colocar en su navegador la siguiente dirección IP: 192.168.4.1. Una vez allí, deberá cambiar el modo para ponerlo en AP+STATION y conectarlo a su WIFI, verificando la dirección IP que arroja al conectarse (ver figura 24).

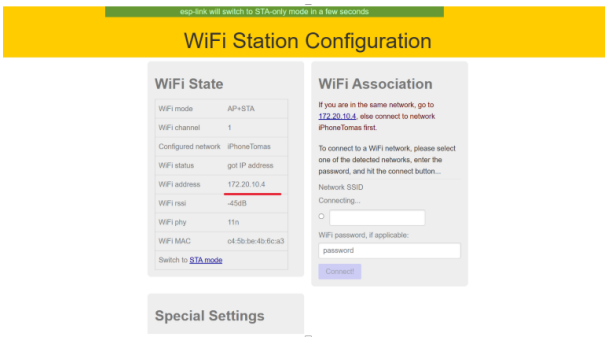

Figure 24: Conexión WiFi ESP8266.

Descargue el siguiente proyecto https://github.com/saacifuentesmu/4100901-uart-k para comparar los tiempos en que se envían por Telnet Y UART en LL. Además deberá descargar el siguiente script de Python https://github.com/Tjimenez1303/Monitoria_estructuras_Computacionales/blob/main/External_Drivers/Scripts_Telnet/com_port_plus_telnet.py. Este script enviará al mismo tiempo un buffer al ESP8266 y por UART al puerto de la tarjeta Nucleo L476RG. Con estos dos archivos deberá medir el tiempo que tarda en enviarsen estos paquetes por UART y por WiFi.

Ejecute este script 10 veces, mida los tiempos que se imprimen y saque sus conclusiones.

Cuestionario

1. Presentar un informe de manera organizada con los resultados de la práctica dos de estructuras computacionales. El informe se estructura en las siguientes secciones: introducción, marco teórico, análisis de resultados, que incluye capturas de pantalla del analizador lógico, así como tablas pertinentes y pantallazos. Además, deberá escribir unas conclusiones (mínimo dos conclusiones).”