Universidad Católica del Uruguay

Laboratorio 2

29 de Mayo de 2025

Diseño de IoT y Sistemas Embebidos

Integrantes del equipo:

Gonzalo Soto

Jerónimo Ventos

Santiago El Ters

Contenido

Primera Parte

2 - Lectura de botones mediante polling

a) Descripción del funcionamiento del TouchPad

Consigna:
Lea las referencias brindadas por Espressif para botones touch y haga una breve descripción de su funcionamiento.

El TouchPad del ESP32-S2 permite detectar toques sin usar botones físicos, usando sensores capacitivos. Estos sensores pueden detectar cuando un dedo se acerca o toca una superficie, sin necesidad de presionar nada.

El funcionamiento se basa en medir la capacitancia de un pin especial del microcontrolador. Cuando acercamos un dedo a ese pin, la capacitancia cambia, y ese cambio se puede usar para saber si hay un “toque” o no.

En total, el ESP32-S2 tiene 14 pines táctiles que se pueden usar como botones. Cada pin se puede configurar individualmente y se puede leer su valor para saber si fue tocado.

A nivel de código, se puede usar lectura por polling (leer en un bucle si fue tocado o no), o usar interrupciones para detectar automáticamente cuando se toca.

b) Esquemático de la placa base y TouchPad

Consigna:
Revise el esquemático de la placa base y el esquemático del TouchPad ¿Hay que modificar algo en la placa base para poder utilizar la placa de expansión?
¿Qué problemas (known issues) existen para esta placa de expansión?

¿Hay que modificar algo en la placa base para poder utilizar la placa de expansión?

Al revisar los esquemáticos de la placa base ESP32-S2-Kaluga-1 y del TouchPad (ESP-LyraP-TouchA), se puede observar que ambas están diseñadas para funcionar juntas sin necesidad de modificaciones. En el caso de la placa base, el conector que se utiliza para enlazar los sensores táctiles es el J9, mientras que en el TouchPad este conector es el J2. Ambos están alineados físicamente y comparten la misma disposición de pines, lo que permite una conexión directa y sencilla. Por lo tanto, no es necesario realizar ningún cambio en la Kaluga para que los botones táctiles funcionen correctamente.

Por otro lado, los botones del TouchPad están conectados a canales táctiles específicos del ESP32-S2, como ESP_TOUCH1, ESP_TOUCH2, hasta ESP_TOUCH14. Estos canales también están en el esquemático de la placa base y se conectan internamente al microcontrolador a través del conector J9, lo que confirma que el diseño ya contempla el uso de esta expansión.

¿Qué problemas (known issues) existen para esta placa de expansión?

A pesar de que existe una compatibilidad, también hay algunos detalles técnicos a tener en cuenta. Uno de ellos es que los sensores capacitivos pueden ser sensibles a interferencias o ruidos eléctricos, lo cual podría afectar la precisión de la lectura. En este caso, se identificó que los pines IO11 e IO6 necesitan no ser inicializados para que se puedan usar los pines del TouchPad.
También es importante activar en el hardware de la placa base, los switches del T1 al T14 para que puedan ser utilizados por el TouchPad.
Otro punto a considerar es que la lectura simultánea de múltiples toques no está implementada en la API básica de ESP-IDF, por lo que se recomienda leer los sensores de uno en uno, utilizando un pequeño delay entre cada lectura para mejorar la estabilidad.

c) Pseudocódigo

```
INICIO

    Inicializar Led
    Inicializar el TouchPad
    Configurar los pines táctiles que se van a usar

    Para cada pin táctil:
        Leer el valor en reposo
        Guardar ese valor para usarlo como referencia
    Fin Para

    Mientras verdadero:
        Para cada pin táctil:
            Si el valor actual es mayor que el del baseline (+300):
                Considerar que el botón está PRESIONADO
            Si no:
                Considerar que el botón está NO PRESIONADO
            Esperar unos milisegundos
        Fin Para
    Fin Mientras

FIN
```

Primero se inicializan los pines táctiles y se mide su valor base sin tocar. Luego, en un bucle, se leen los valores actuales y se comparan con ese baseline. Si el valor es mucho mayor(por ejemplo, baseline +300), se considera que el botón está presionado. El proceso se repite continuamente con una pequeña pausa para evitar lecturas duplicadas.

Segunda Parte

1 - Interfaz de red y loop de eventos

a) Análisis ESP-NETIF y Event Loop

Consigna:
Para poder llevar a cabo la implementación tanto del punto de acceso Wifi como la estación WiFi es necesario comprender que es el ESP-NETIF y que es el Event Loop.

ESP-NETIF se trata de una biblioteca con funciones duales una que permite al programa crear una interfaz entre las funciones de TCP/IP y el código del usuario para facilitar su uso. A su vez que dar coherencia a las Apis que utilicen los mismos.

Mientras loop_event es una biblioteca que permite el creado rápido de un encolado de funciones o loop de eventos. El fuerte de esta biblioteca está en que permite la conexión de distintas partes del código sin recurrir a un uso mayor de recursos. Este tipo de estructura es útil para situaciones como el comprobado de conexión a una red.

Según el manual de ESP-IDF la función Loop_event se emplea de la siguiente manera:

1. El usuario define una función que debe ejecutarse cuando se envía un evento a un bucle. Esta función se denomina controlador de eventos y debe tener la misma firma que `esp_event_handler_t`.
2. Un bucle de eventos se crea usando `esp_event_loop_create()`, que genera un identificador para el bucle de tipo `esp_event_loop_handle_t`. Los bucles de eventos creados con esta API se denominan bucles de eventos de usuario. Sin embargo, existe un tipo especial de bucle de eventos llamado bucle de eventos predeterminado, que se describe en el bucle de eventos predeterminado .
3. Los componentes registran los controladores de eventos en el bucle mediante `esp_event_handler_register_with()`. Los controladores pueden registrarse con varios bucles; consulte las notas sobre el registro de controladores .
4. Las fuentes de eventos publican un evento en el bucle usando `esp_event_post_to()`.
5. Los componentes que desean eliminar sus controladores para que no sean llamados pueden hacerlo cancelando su registro en el bucle mediante `esp_event_handler_unregister_with()`.

6. Los bucles de eventos que ya no se necesitan se pueden eliminar usando `esp_event_loop_delete()`.

b) Análisis de la ‘esp_netif_init()‘

Consigna:
Comente que hace la función ‘esp_netif_init()‘ y si cree que es necesario para la implementación de redes WiFi.

La función esp_netif_init() es la que se encarga de que comience la pila de código que conecte con el dispositivo deseado.

c) Análisis de la esp_event_loop_create_default()‘

Consigna:
Comente qué hace la función ‘esp_event_loop_create_default()’ y qué utilidad podría tener para la implementación de redes WiFi.

La función esp_event_loop_create_default() empieza el loop de manera que el usuario no lo vea para operar las funciones de conexión a red.

Ambas son funciones necesarias para iniciar tanto el AP como STA. Una para iniciar la conexión a redes wifi y la otra para poder operar distintas funciones sin interrumpir el código con el que interactúa un usuario de la página web.

2 - Implementación AP

a) Análisis configuración API de WiFi

Consigna:
Comente con qué opciones de configuración cuenta el API de WiFi del equipo basado en la guía de referencia.

Primero estan las opciones de configuración del Kaluga. Lo que refiere a programar un Access point, punto de entrada a una red, Station, administrador de información en redes, o un programa híbrido que maneja ambas funciones.
Luego están las funciones que permiten configurar los protocolos de seguridad inalámbrica de conexión. que se definen como WPA (Wifi protected access) que tiene además tiene soporte de versión 2 y 3 entre otras alternativas.
Por último tiene un modo de escaneo activo y pasivo de puntos de acceso. A parte de un modo promiscuo que permite captar información de paquetes que circulan por el internet sin necesidad de conectarse a una red específica.

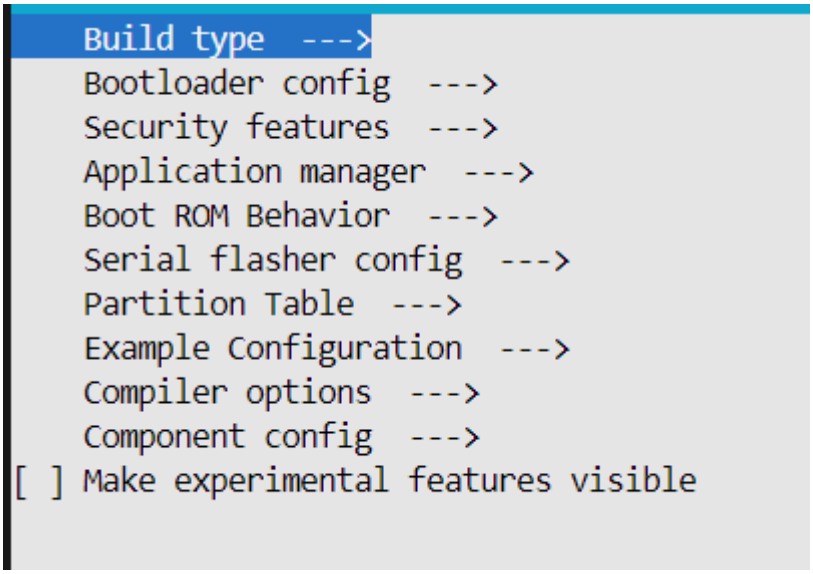
Todas estas son las funciones basicas que otorga la guia de ejemplos en el apartado de configuracion

b) Pseudocódigo AP

```
  Inicializar almacenamiento NVS
    Inicializar interfaz de red
    Crear ciclo de eventos por defecto

    Configurar WiFi en modo AP:
        Definir SSID, contraseña y canal
        Definir autenticación (WPA2 o WPA3)

        Si contraseña está vacía:
            Establecer autenticación abierta
        Fin Si

    Registrar manejador de eventos WiFi:
        Cuando un dispositivo se conecte:
            Registrar en log "dispositivo conectado"
        Cuando un dispositivo se desconecte:
            Registrar en log "dispositivo desconectado"
    Fin Registrar

    Iniciar WiFi en modo AP
    Registrar en log configuración completada

FIN
```

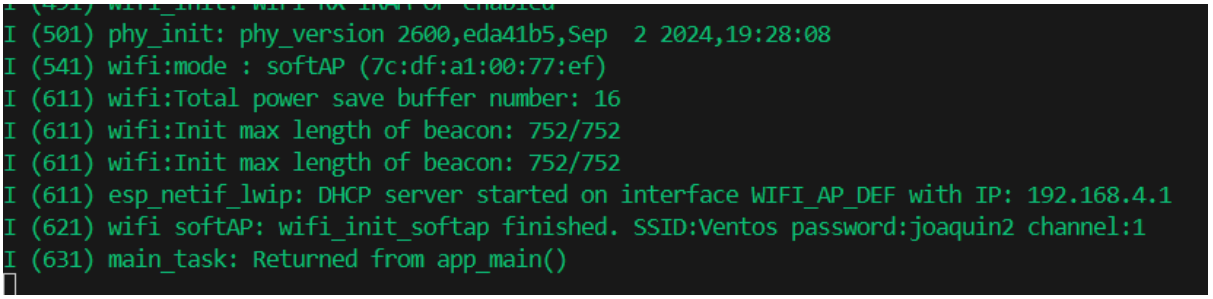
c) Análisis de datos en ESP32-S2

Consigna:
Lleve a cabo dicha implementación e intente conectar un dispositivo al ESP32-S2. ¿Qué datos del equipo que se conecta muestra el equipo? ¿Qué datos de red relevantes puede obtener en el monitor del idf?



Mientras que en las configuraciones del [Idf.py] se puede configurar como se buildea, flashea y ejecuta. También se requiere tener el SSDI para poder conectarse junto con la contraseña (Aunque esta no hace falta en una red abierta). Otra cosa que permite configurar es el canal al que va acceder (de 1 a 13 qué frecuencia va a usar). Y por último, a cuántas estaciones se le permite al dispositivo Kaluga conectarse (por defecto en 4).

Para finalizar, cuando el código se ejecuta se puede ver con que IP se conectó a la red aparte de datos generales como la hora de ejecución y nombre del programa.



c) Análisis de configuraciones WiFi

Consigna:
Pruebe distintas configuraciones (WiFi abierto/con contraseña, SSID oculto, etc.) y comente su resultado en la implementación.

Los datos relevantes que cambian es que en una red abierta es necesario no dejar una contraseña al conectarse, y la red oculta no permite que el SSDI de la red sea visible al ejecutar el programa.

3 - Implementación STA

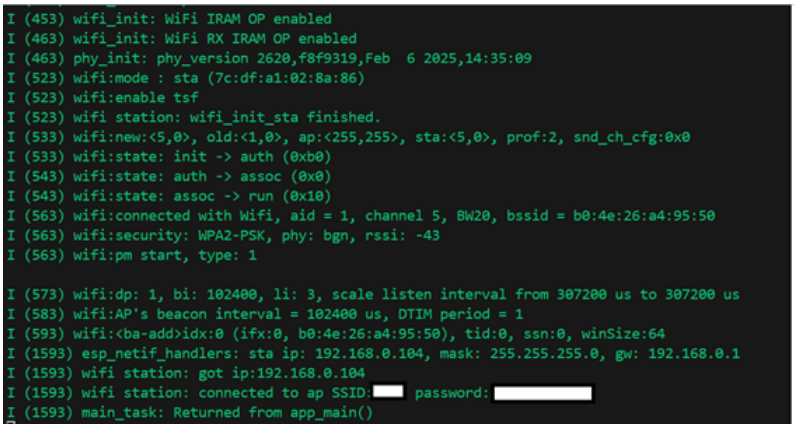
a) Pseudocódigo STA

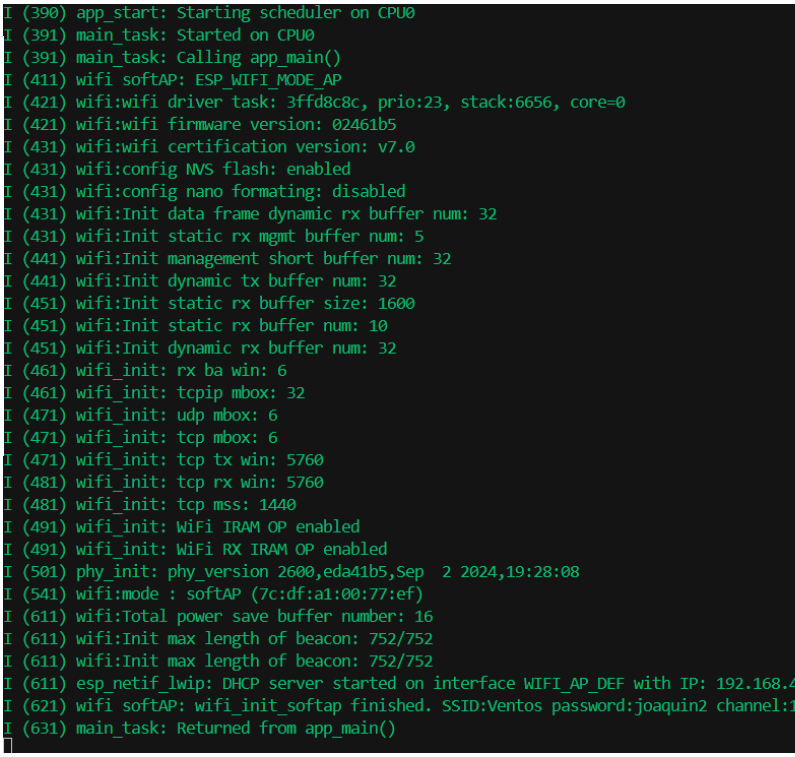
```
INICIO

    Inicializar NVS

    Inicializar la pila de red (netif)

    Crear el loop de eventos por defecto

    Crear interfaz Wi-Fi en modo ESTACIÓN (STA)

    Inicializar el driver Wi-Fi


    Para cada manejador de evento:

        Si evento == WIFI_EVENT_STA_START:

            Llamar a esp_wifi_connect()

        Si evento == WIFI_EVENT_STA_DISCONNECTED:

            Si reintentos < máximo:

                Incrementar contador de reintentos

                Llamar a esp_wifi_connect()

            Si no:

                Marcar fallo de conexión

        Si evento == IP_EVENT_STA_GOT_IP:

            Guardar dirección IP recibida

            Reiniciar contador de reintentos

            Marcar éxito de conexión

    Fin Para
```

```
    Configurar credenciales de la red:
        SSID ← CONFIG_ESP_WIFI_SSID
        Contraseña ← CONFIG_ESP_WIFI_PASSWORD
        Umbral de authmode y parámetros SAE

    Establecer modo Wi-Fi → STA
    Aplicar configuración y arrancar Wi-Fi

    Mientras verdadero:
        Si conexión exitosa:
            Mostrar “Conectado a SSID y contraseña”
            Salir del bucle
        Si fallo de conexión:
            Mostrar “Falló la conexión a SSID y contraseña”
            Salir del bucle
        Esperar unos milisegundos
    Fin Mientras

FIN
```

b) Análisis conexión a red

Consigna:
Lleve a cabo dicha implementación e intente conectar el equipo a una red. ¿Qué datos de red relevantes puede obtener en el monitor del idf? ¿Qué diferencias hay comparado con la implementación del AP?





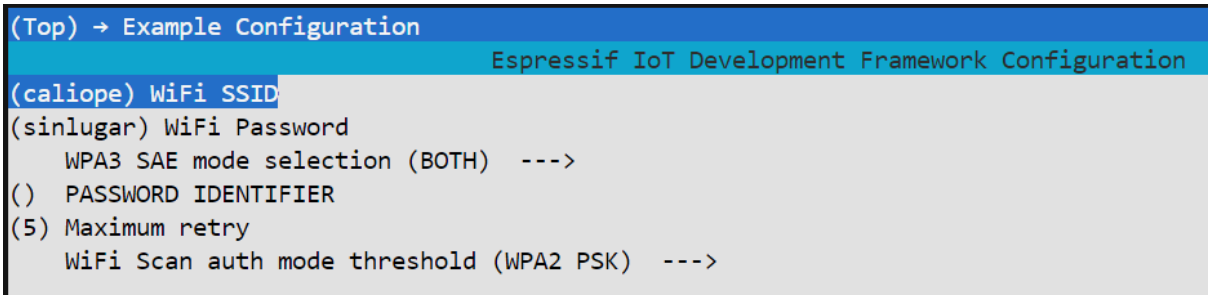
Además de hora de iniciación del programa:

Concepto	Station (STA)	SoftAP (AP)
IP	Se le asigna un router vía DHCP (cliente DHCP).	IP estática (192.168.4.1) y se levanta DHCP server.
Roles de DHCP	Cliente	Servidor
Información impresa	BSSID, RSSI, seguridad, estado de asociación, IP.	SSID, password, buffer/beacon config, clients.
driver	Más “ligero” (solo estados y netif).	Más “pesado”: buffers, mailboxes, firmware, NVS.
Eventos de cliente	No tiene.	Aparecen al detectar clientes (si usas esos handlers).

c) Análisis de distintas configuraciones

Consigna:
Pruebe distintas configuraciones y comente su efecto en la implementación.

Hay varias configuraciones del protocolo 802.11, y en las configuraciones es donde se configura el wifi objetivo (ssid y password del wifi)



Estos parámetros son usados por esta función wifi_config.

1. `threshold.authmode` filtra en el scan y en la conexión:
    - Si lo pones muy alto (p.ej. WPA3), no accedes ni ves redes menos seguras.
    - Si lo dejas OPEN, ves todos los APs, incluso los sin cifrar.
2. `sae_pwe_h2e` **(WPA3-SAE)** te deja forzar o desactivar SAE:
    - “BOTH” es lo más flexible (usa SAE cuando puede, PSK en WPA2).
    - “SAE only” te aísla de las WPA2 clásicas; “PSK only” te aparta de las WPA3.
3. `max_retry` controla la resiliencia a contraseñas malas o APs caídos: pocos retries = reacción rápida; muchos = paciencia a redes inestables.

4. **SSID/Password en Kconfig**: cada vez que quieras apuntar a otro AP, debes cambiar ahí y recompilar — no hay “over-the-air” dinámico sin reflash.

Con estos cuatro knobs puedes ajustar desde “solo conectó a lo más fuerte (SAE/WPA3)” hasta “acepto todo y me quedo con WEP/Open”, así como la tolerancia a fallos de red.

Tercera Parte

a) Pseudocódigo Webserver

Consigna:
Recordando las funciones provistas en la guía de referencia, implemente en pseudocódigo (se recomienda investigar bien los distintos ejemplos para llevar a cabo dicha implementación) un servidor web y que al acceder despliegue el mensaje de “Hola mundo” en el navegador.

```
INICIO

      Inicializar almacenamiento NVS
      Inicializar pila de red (TCP/IP)
      Crear y configurar la red WiFi
      Puede ser como cliente (STA) o como punto de acceso (AP)
      Iniciar la conexión WiFi

      Definir función para responder a solicitudes web:
      Cuando se acceda a la ruta "/", mostrar el texto "Hola mundo"

      Crear servidor web
      Registrar la función definida para la ruta "/"
      Iniciar el servidor web

      Mantener el servidor activo (esperando conexiones)

FIN
```

Primero se inicializan los componentes necesarios para red y almacenamiento. Luego se configura la red WiFi: el equipo puede actuar como cliente (conectándose a una red) o como punto de acceso (creando su propia red).
Después se crea una función que responde a los accesos desde el navegador, mostrando simplemente el texto "Hola mundo".
Finalmente, se enciende el servidor web y queda funcionando, esperando que alguien acceda desde un navegador.