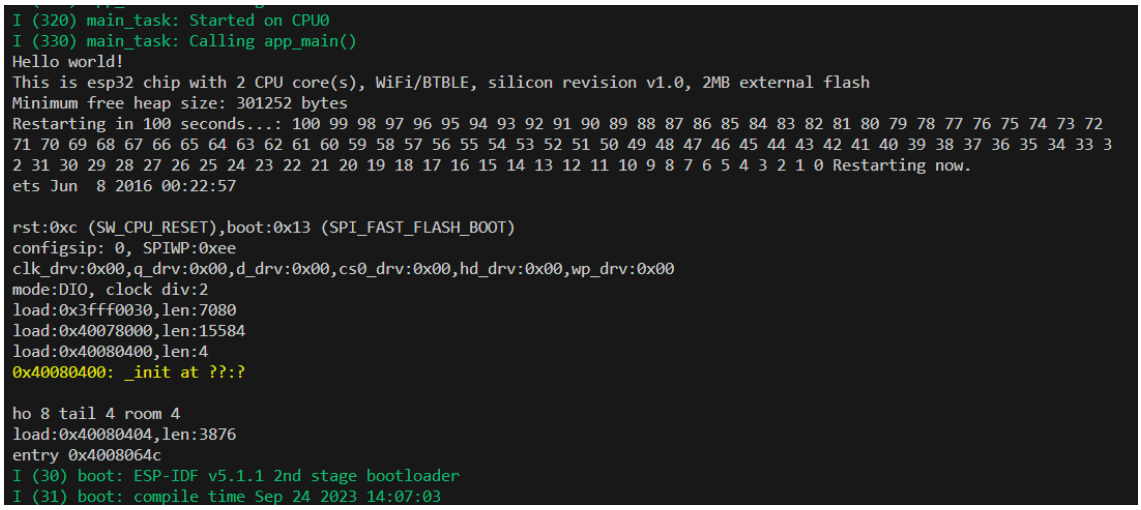
Practica 1

Entregable 1

Modifica el período de suspensión de la tarea para que sea mayor o menor, y comprueba que efectivamente esto modifica el comportamiento del firmware cargado.

Acciones realizadas:

1. Se crea la constante TIME_WAIT con valor 100 modificando el bucle dentro de app_main y al volcar a la placa examinamos el output en el monitor verificando el funcionamiento.

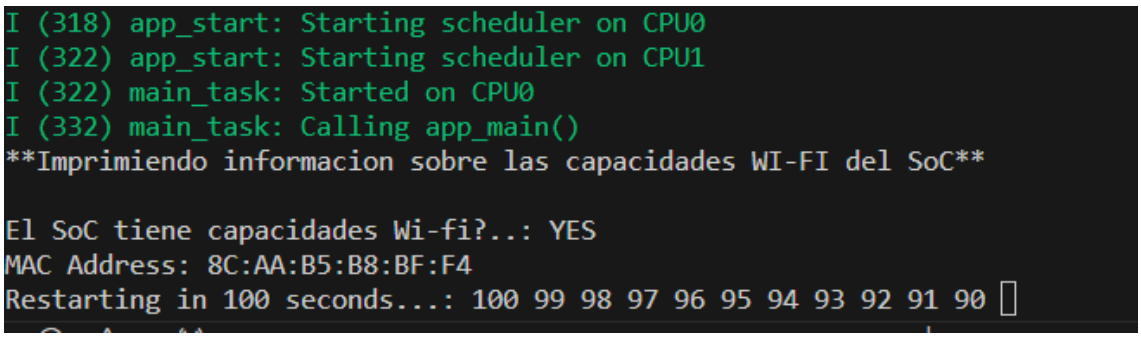


Entregable 2

Modifica el programa para que se compruebe debidamente si el SoC tiene capacidades WiFi y muestre la información correspondiente por la salida estándar (para ello, puedes consultar la siguiente página).

Acciones realizadas:

1. Para verificar si el chip soporta wifi realizamos las siguientes acciones:
    a. Recuperamos la información del chip llamando a la función esp_chip_info con una estructura de tipo esp_chip_info_t.
    b. Realizamos una operación AND entre la sección features de la estructura obtenida y la MACRO CHIP_FEATURE_WIFI_BGN.
2. Adicionalmente mostramos información de la MAC del chip WIFI:
    a. Obtenemos la dirección base de la MAC programada de fabrica con la función esp_efuse_mac_get_default
    b. Leemos la dirección MAC con la función esp_read_mac
    c. Mostramos el array obtenido con la función print_mac



Entregable 3

Implementa una modificación del programa hello_world que implemente y planifique dos tareas independientes con distinta funcionalidad (en este caso, es suficiente con mostrar por pantalla algún mensaje) y distintos tiempos de suspensión. Comprueba que, efectivamente, ambas tareas se ejecutan concurrentemente.

Acciones realizadas:

1. Se crean dos tareas:
    a. Tarea 1 muestra cada 4 segundos la información de los cores y capacidades del chip
    b. Tarea 2 muestra cada 7 información del chip y su versión, capacidad y tipo de la memoria flash y la disponibilidad de memoria de pila.
    c. En ambas tareas, la información se obtiene localmente usando la función esp_chip_info.

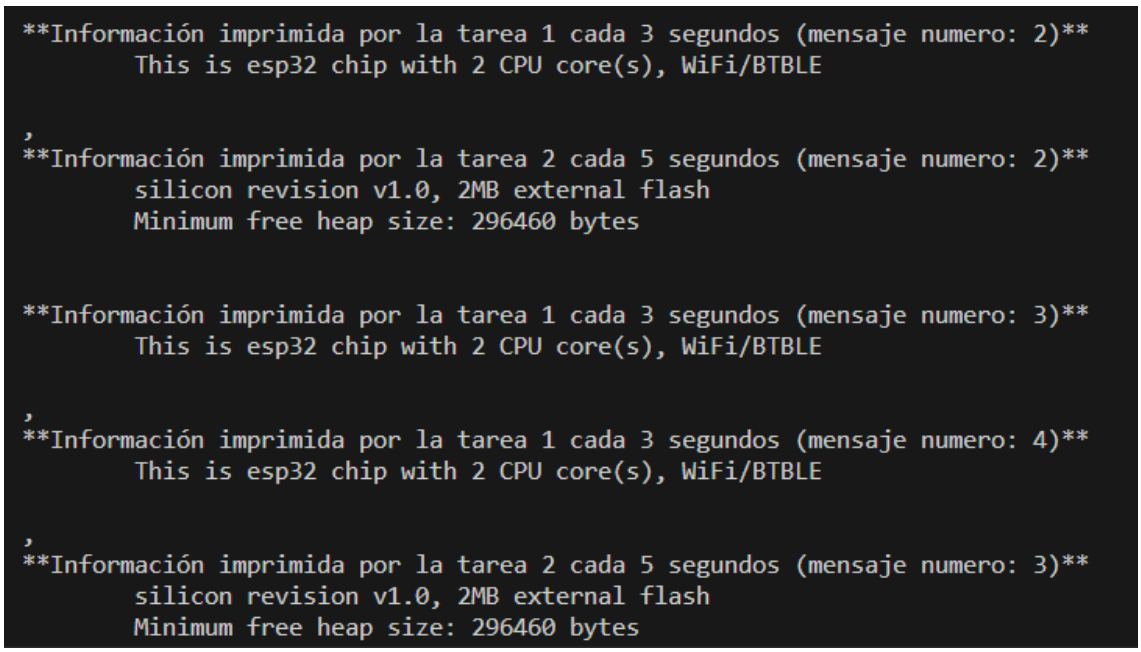
```
**Información imprimida por la tarea 1 cada 4 segundos (mensaje numero: 2)**
        This is esp32 chip with 2 CPU core(s), WiFi/BTBLE

,
**Información imprimida por la tarea 2 cada 7 segundos (mensaje numero: 2)**
        silicon revision v1.0, 2MB external flash
        Minimum free heap size: 296460 bytes


**Información imprimida por la tarea 1 cada 4 segundos (mensaje numero: 3)**
        This is esp32 chip with 2 CPU core(s), WiFi/BTBLE

,
**Información imprimida por la tarea 1 cada 4 segundos (mensaje numero: 4)**
        This is esp32 chip with 2 CPU core(s), WiFi/BTBLE

,
**Información imprimida por la tarea 2 cada 7 segundos (mensaje numero: 3)**
        silicon revision v1.0, 2MB external flash
        Minimum free heap size: 296460 bytes
```

Entregable 4

Modifica el proyecto hello_world para que defina dos opciones de configuración que permitan definir el tiempo de espera de cada una de las dos tareas que hayas definido en tu anterior solución. Haz uso de ellas en tu código y comprueba que efectivamente su modificación a través del sistema de menús permite una personalización del comportamiento de tus códigos.

Acciones realizadas:

1. Dentro del fichero Kconfig.projbuild se han definido las entradas TIME_WAIT_TASK1 & TIME_WAIT_TASK2 obteniendo su valor a partir de CONFIG_TIME_WAIT_TASK1 y CONFIG_TIME_WAIT_TASK1 respectivamente.
2. Se muestra la misma información que en el entregable 3.



Entregable 5 - Escaneado de redes WiFi

Compila, flashea y monitoriza el ejemplo scan situado en el directorio examples/wifi/scan. Crea un nuevo proyecto a partir de este ejemplo y amplia el número máximo de redes a escanear a 20 a través del menú de configuración del ejemplo. Crea un punto de acceso WiFi con tu teléfono móvil y observa que, efectivamente, es escaneado por el ejemplo.

Analiza el código de la función wifi_scan (tarea principal). Céntrate especialmente en las líneas que permiten activar y configurar el escaneado de redes. Intenta entender el funcionamiento general del programa, consultando y apuntando el cometido de cada línea, con especial interés a aquellas funciones con prefijo esp_wifi_*. Si tienes dudas puede consultar la documentación oficial de ESP-IDF.

Acciones realizadas:

1. Se compila, flashea y monitoriza el ejemplo scan.
2. Comentadas las llamadas a las funciones dentro de wifi_scan.

```
I (804) wifi:mode : sta (8c:aa:b5:b8:bf:f4)
I (804) wifi:enable tsf
I (3314) scan: Total APs scanned = 7
I (3314) scan: SSID              UCM-CONGRESO
I (3314) scan: RSSI              -54
I (3314) scan: Authmode          WIFI_AUTH_WPA2_PSK
I (3314) scan: Pairwise Cipher   WIFI_CIPHER_TYPE_CCMP
I (3324) scan: Group Cipher      WIFI_CIPHER_TYPE_CCMP
I (3324) scan: Channel           13

I (3334) scan: SSID              UCMOT
I (3334) scan: RSSI              -54
I (3334) scan: Authmode          WIFI_AUTH_WPA2_PSK
I (3344) scan: Pairwise Cipher   WIFI_CIPHER_TYPE_CCMP
I (3344) scan: Group Cipher      WIFI_CIPHER_TYPE_CCMP
I (3354) scan: Channel           13

I (3354) scan: SSID              UCM
I (3364) scan: RSSI              -54
I (3364) scan: Authmode          WIFI_AUTH_OPEN
I (3364) scan: Pairwise Cipher   WIFI_CIPHER_TYPE_NONE
I (3374) scan: Group Cipher      WIFI_CIPHER_TYPE_NONE
I (3384) scan: Channel           13

I (3384) scan: SSID              eduroam
I (3384) scan: RSSI              -55
I (3394) scan: Authmode          WIFI_AUTH_WPA2_ENTERPRISE
I (3394) scan: Pairwise Cipher   WIFI_CIPHER_TYPE_CCMP
I (3404) scan: Group Cipher      WIFI_CIPHER_TYPE_CCMP
I (3404) scan: Channel           13
```

Entregable 6 - Gestión de eventos de red

Crea un proyecto a partir del ejemplo station situado en el directorio examples/wifi/getting_started. Compilalo, flashealo y monitoriza su salida estándar. Acuérdate de modificar el SSID de la red al que conectará, así como la contraseña elegida a través del sistema de menús de configuración.

Acciones realizadas:

1. Parametrizada la wifi del móvil con valores para testing
2. Al generar el proyecto desde examples/wifi/getting_started station, se configuran parámetros de entrada desde menuconfig: Wifi SSID / Wifi Password.

```
I (700) wifi_init: WiFi IRAM OP enabled
I (700) wifi_init: WiFi RX IRAM OP enabled
I (710) phy_init: phy_version 4670,719f9f6,Feb 18 2021,17:07:07
I (820) wifi:mode : sta (8c:aa:b5:b8:bf:f4)
I (820) wifi:enable tsf
I (820) wifi station: wifi_init_sta finished.
I (830) wifi:new:<6,0>, old:<1,0>, ap:<255,255>, sta:<6,0>, prof:1
I (830) wifi:state: init -> auth (b0)
I (840) wifi:state: auth -> assoc (0)
I (850) wifi:state: assoc -> run (10)
I (980) wifi:state: run -> init (2c0)
I (990) wifi:new:<6,0>, old:<6,0>, ap:<255,255>, sta:<6,0>, prof:1
I (990) wifi station: retry to connect to the AP
I (990) wifi station: connect to the AP fail
I (3400) wifi station: retry to connect to the AP
I (3400) wifi station: connect to the AP fail
I (3410) wifi:new:<6,0>, old:<6,0>, ap:<255,255>, sta:<6,0>, prof:1
I (3410) wifi:state: init -> auth (b0)
I (3420) wifi:state: auth -> assoc (0)
I (3430) wifi:state: assoc -> run (10)
I (3570) wifi:connected with TESTAP, aid = 1, channel 6, BW20, bssid = 02:5b:c6:a1:9e:a7
I (3570) wifi:security: WPA2-PSK, phy: bgn, rssi: -49
I (3580) wifi:pm start, type: 1

I (3590) wifi:<ba-add>idx:0 (ifx:0, 02:5b:c6:a1:9e:a7), tid:0, ssn:0, winSize:64
I (3650) wifi:AP's beacon interval = 102400 us, DTIM period = 2
I (4580) esp_netif_handlers: sta ip: 192.168.41.86, mask: 255.255.255.0, gw: 192.168.41.114
I (4580) wifi station: got ip:192.168.41.86
I (4580) wifi station: connected to ap SSID:TESTAP password:Testpwd123?
I (4590) main_task: Returned from app_main()
```

Responde a la siguiente pregunta de forma razonada: ¿Qué eventos se asocian a la ejecución de qué función en el firmware que estás estudiando?

Los eventos WIFI_EVENT e IP_EVENT se asocian a la función event_handler.

- Ante un evento WIFI_EVENT dentro de la función se intenta la conexión llamando a la función esp_wifi_connect();
- En caso de un evento IP_EVENT se muestra la ip obtenida al conectarse a la red wifi.

Entregable 7

Modifica el firmware para que el handler de tratamiento de la obtención de una dirección IP sea independiente del tratamiento del resto de eventos del sistema WiFi que ya se están considerando. Comprueba que, efectivamente sigue observándose la salida asociada a dicho evento, aun cuando ambas funciones sean independientes. Entrega el código modificado.

Acciones realizadas:

1. Se generan dos funciones distintas para cada evento: event_handler_WIFIevent y event_handler_IPevent.
2. Se verifica la información en el monitor.