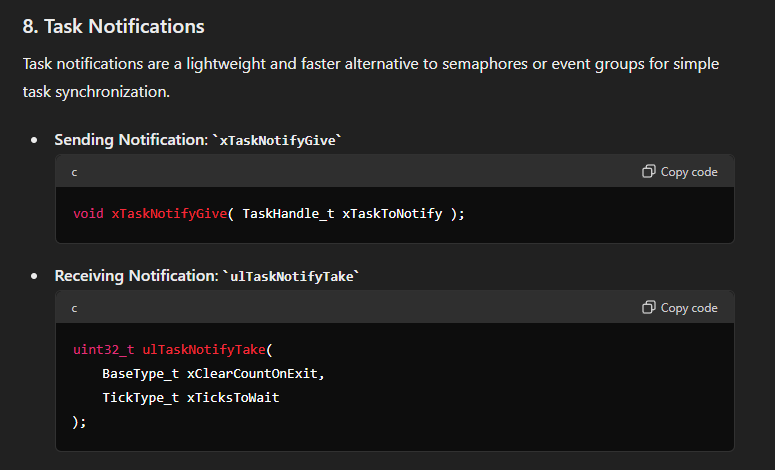
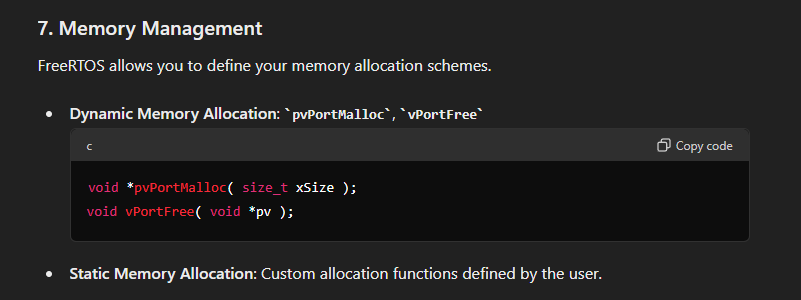
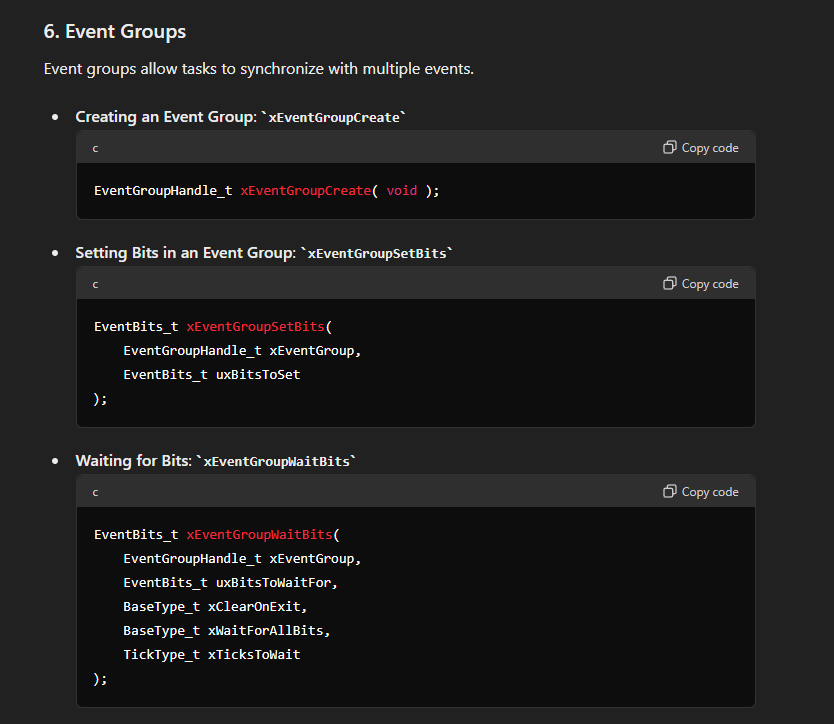
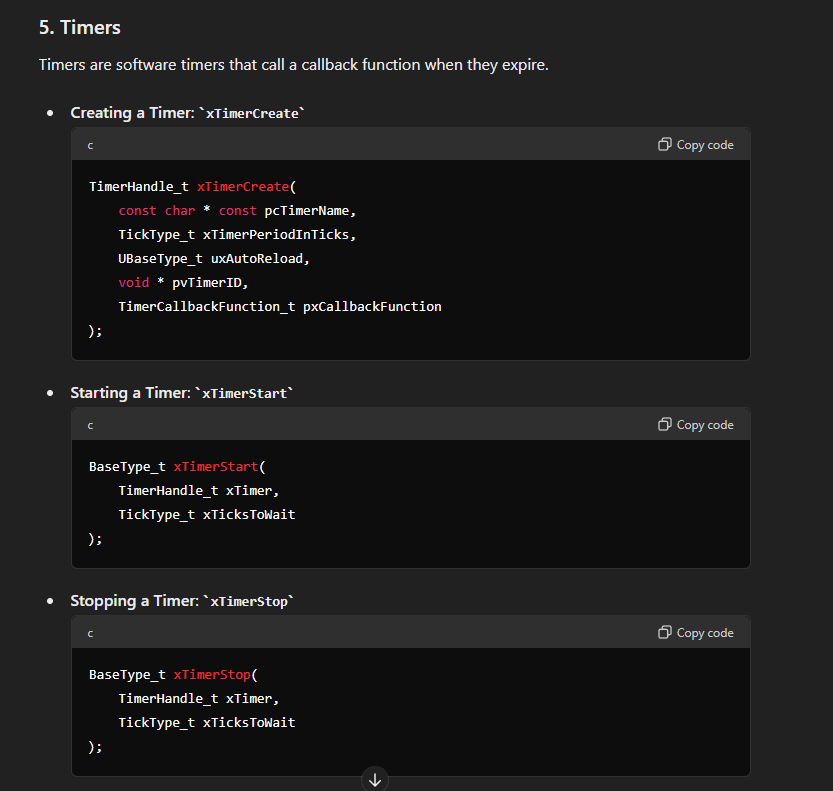
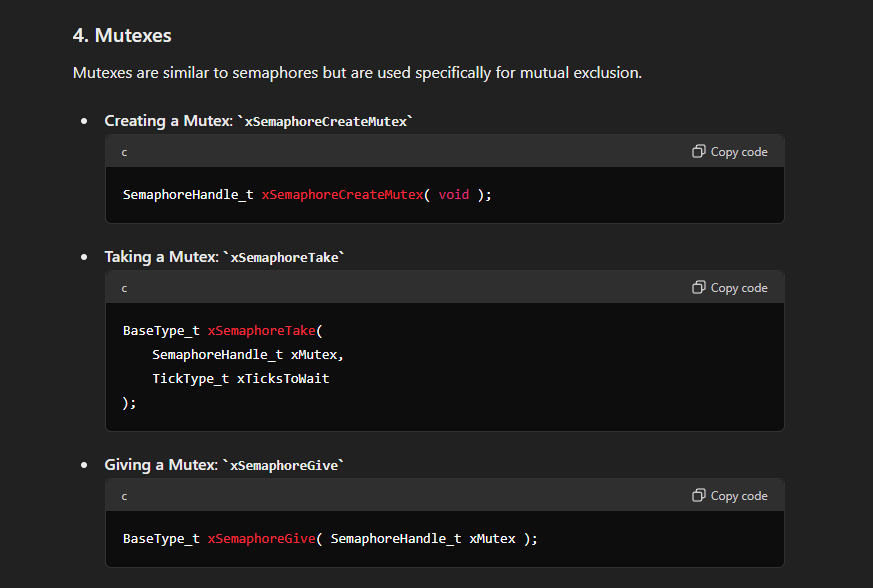
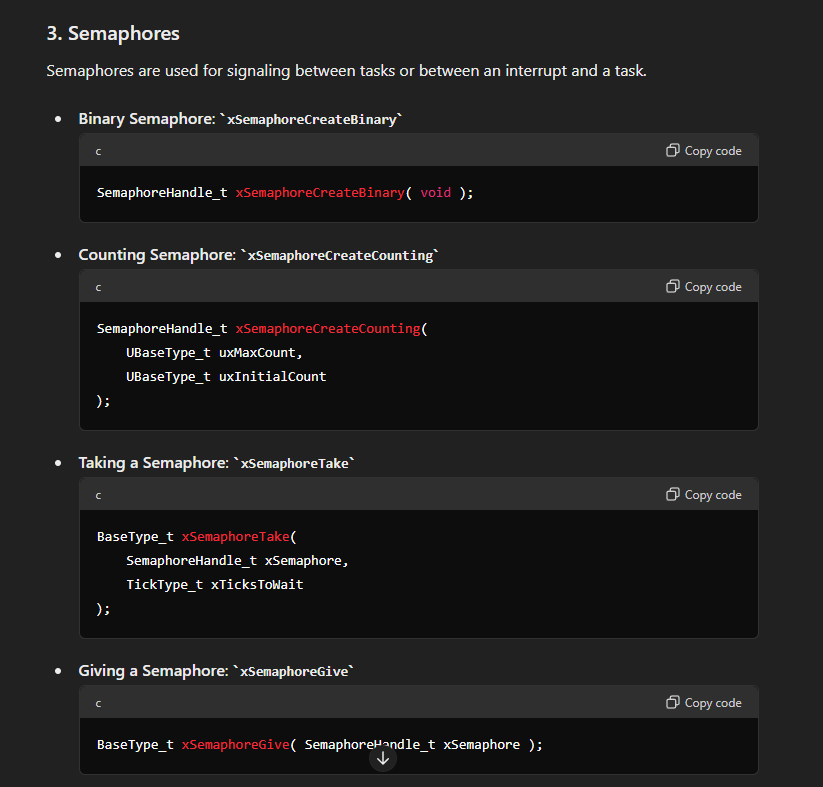
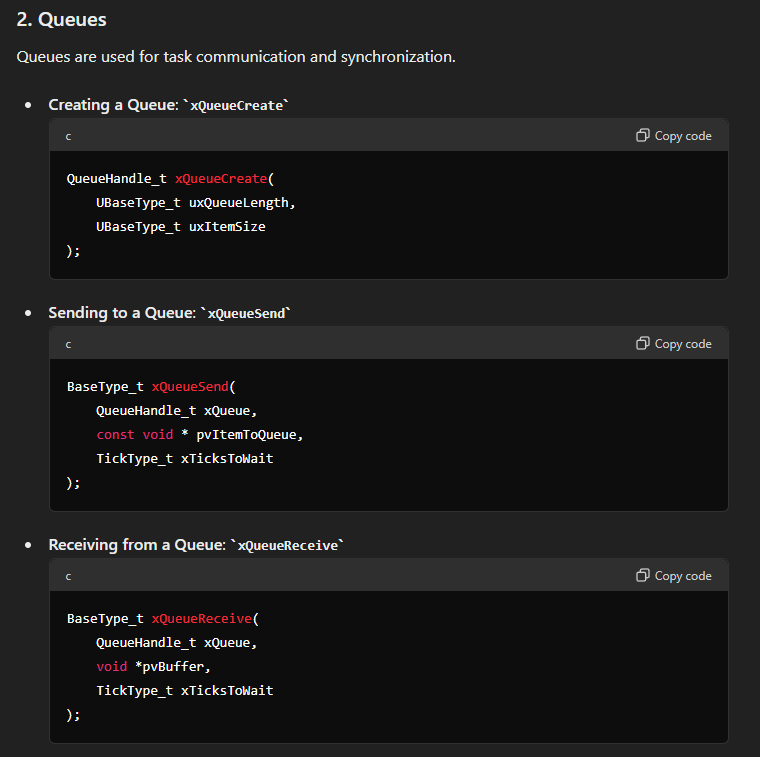
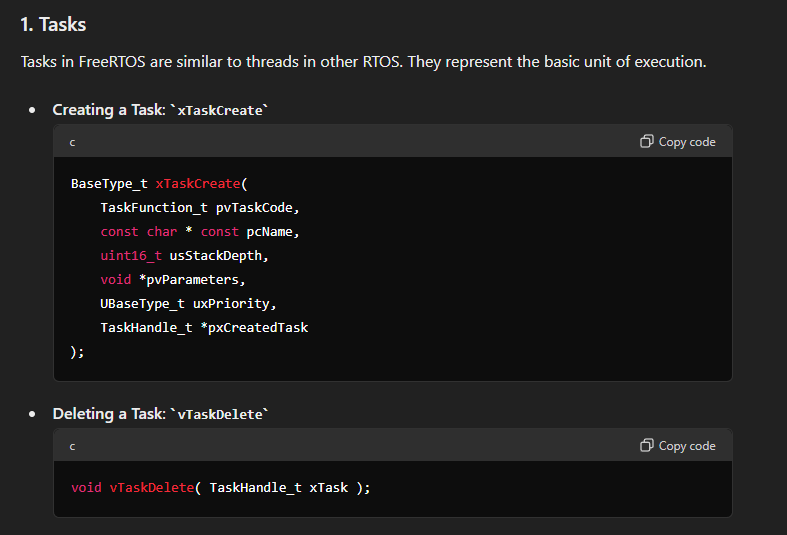
# FreeRTOS notes



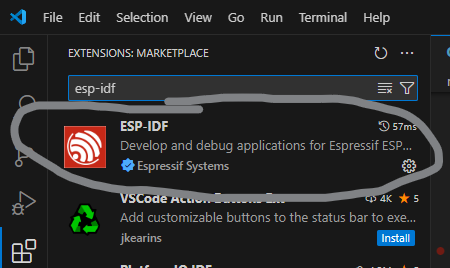


* xTaskCreate, xTaskDelete
* xQueueCreate, xQueueSend, xQueueReceive
* xSemaphoreCreateBinary, xSemaphoreCreateCounting, xSemaphoreTake, xSemaphoreReceive
* xSemaphoreCreateMutex, xSemaphoreTake, xSemaphoreGive
* xTimerCreate, xTimerStart, xTimerStop
* xEventGroupCreate, xEventGroupSetBits, xEventGroupWaitBits
* pvPortMalloc, vPortFree
* xTaskNotifyGive, ulTaskNotifyTake

# Setting up the ESP32 for flashing

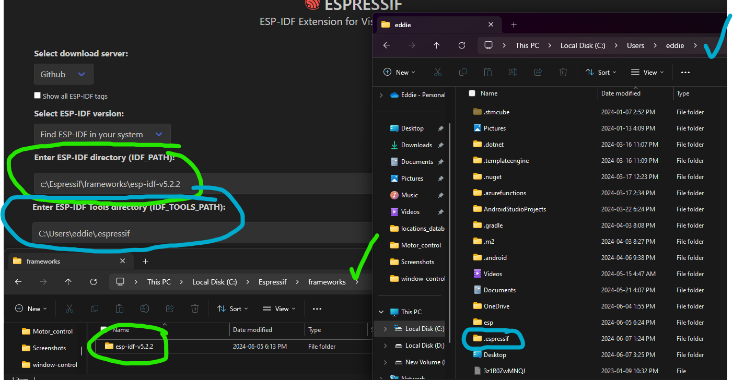


After the ESP-IDF extension is installed, on the ESP-IDF setup tab, double check your PATHs.

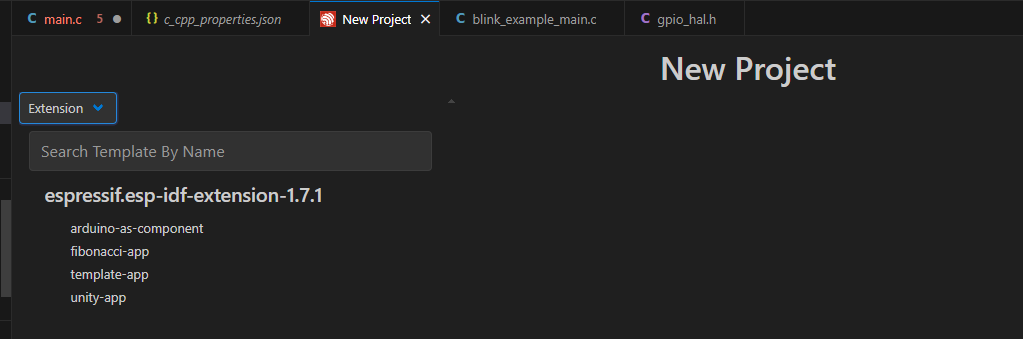


In order to get the ESP32 set up and functional, went to this website and installed ESP-IDF on vscode:

https://docs.espressif.com/projects/esp-idf/en/stable/esp32/get-started/windows-setup.html







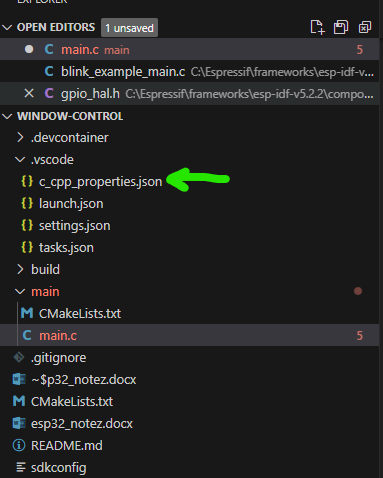
On VSCode when you press the “Install” button, it’ll ask what workspace you want to install to at the top of the screen, which in my case is the root project folder where my firmware, hardware, schematics etc. will be located.



After creating a new project and selecting the folder, I choose template-app on extension. This creates a blank project with no code other than app\_main



# Getting it to blink

I changed the code to this in c\_cpp\_properties.json

{

    "configurations": [

      {

        "name": "ESP-IDF",

        "cStandard": "c11",

        "cppStandard": "c++17",

        "includePath": [

          "${config:idf.espIdfPathWin}/components/\*\*",

          "${workspaceFolder}/\*\*"

        ],

        "browse": {

          "path": [

            "${config:idf.espIdfPathWin}/components",

            "${workspaceFolder}"

          ],

          "limitSymbolsToIncludedHeaders": false

        },

        "compilerPath": "${config:idf.toolsPathWin}\\tools\\xtensa-esp-elf\\esp-13.2.0\_20230928\\xtensa-esp-elf\\bin\\xtensa-esp32-elf-gcc.exe"

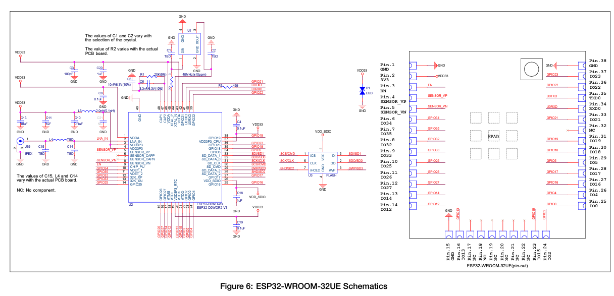
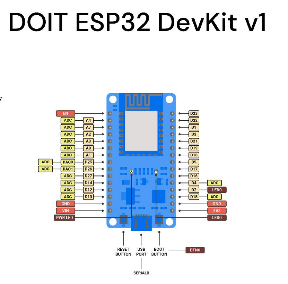
      }

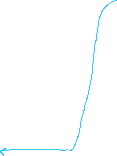
    ],

    "version": 4

  }

Getting ESP32 to work: LED0 on the board is connected to D2, or GPIO 2.

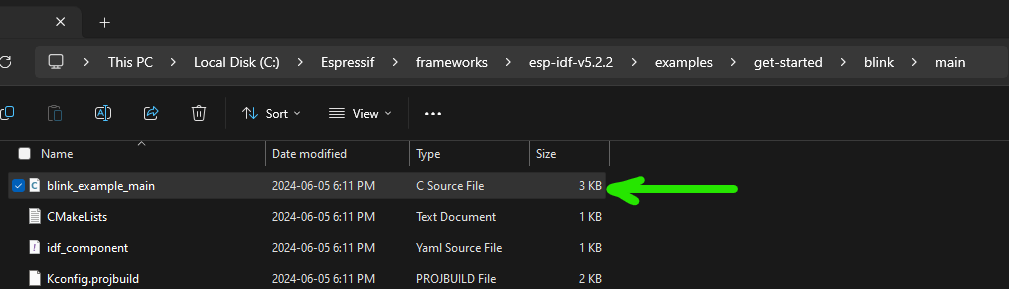




#include <stdio.h>

#include "freertos/FreeRTOS.h"

#include "freertos/task.h"

Note: this code is sort of based on their blink\_example\_main.c code. 

#include "driver/gpio.h"

#include "esp\_log.h"

#define CONFIG\_BLINK\_PERIOD 1000

//ESP32 controls window linear actuator to open and close

static const char\* TAG = "main";

static gpio\_num\_t LED\_PIN = 2;

static uint32\_t LED\_state = 0;

void configure\_led(void) {

    ESP\_LOGI(TAG, "configuring LED...");

    gpio\_reset\_pin(LED\_PIN);

    gpio\_mode\_t mode = GPIO\_MODE\_OUTPUT;

    gpio\_set\_direction(LED\_PIN, mode);

}



static void blink\_led(void) {

    gpio\_set\_level(LED\_PIN, LED\_state);



}

void app\_main(void)

{

    configure\_led();

    while(1) {

        ESP\_LOGI(TAG, "Turning the LED %s", LED\_state == true ? "ON": "OFF");

        blink\_led();

        LED\_state = !LED\_state;

        vTaskDelay(CONFIG\_BLINK\_PERIOD / portTICK\_PERIOD\_MS);

    }

}

# Blinking as Task

void vBlinkTask(void \*pvParameters) {

    while(1) {

        ESP\_LOGI(TAG, "Turning the LED %s", LED\_state == true ? "ON": "OFF");

        blink\_led();

        LED\_state = !LED\_state;

        //vTaskDelay(CONFIG\_BLINK\_PERIOD / portTICK\_PERIOD\_MS);

        vTaskDelay(pdMS\_TO\_TICKS(CONFIG\_BLINK\_PERIOD));

    }

}

extern "C" void app\_main(void)

{

    configure\_led();

    // LED blinking task

Stack Overflow Error, change to 4096 for sufficient stack size

    xTaskCreate(

        vBlinkTask,

        "Blink Task",

        configMINIMAL\_STACK\_SIZE,



        NULL,

        tskIDLE\_PRIORITY,

        NULL);

    while(1) {

    }

}

# Progress Notes

## WiFi

I (458) main: Turning the LED OFF



E (1078) mqtt\_client: Error parse uri = 192.168.77.237:1883

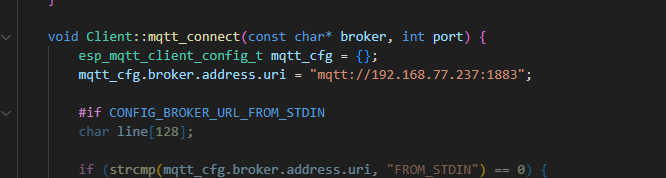
E (1078) mqtt\_client: Client was not initialized

I (1508) main: Turning the LED ON

I (2518) main: Turning the LED OFF

I (3088) main: Sending MQTT message...

I (3088) mqtt\_functionality: [APP] Startup..

I (3088) mqtt\_functionality: [APP] Free memory: 281560 bytes



I (3088) mqtt\_functionality: [APP] IDF version: v5.2.2-dirty



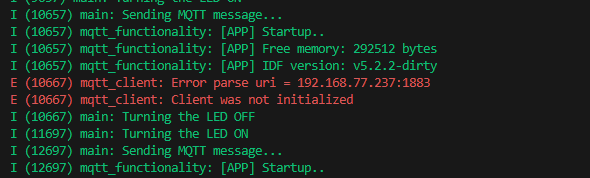
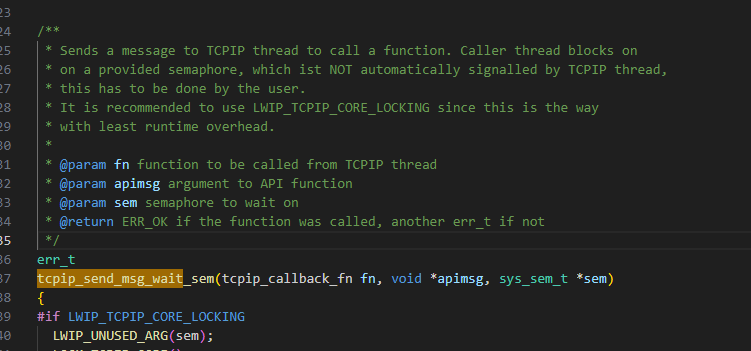
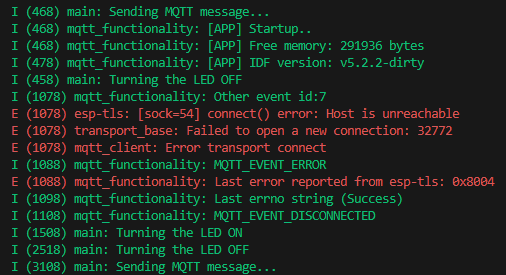
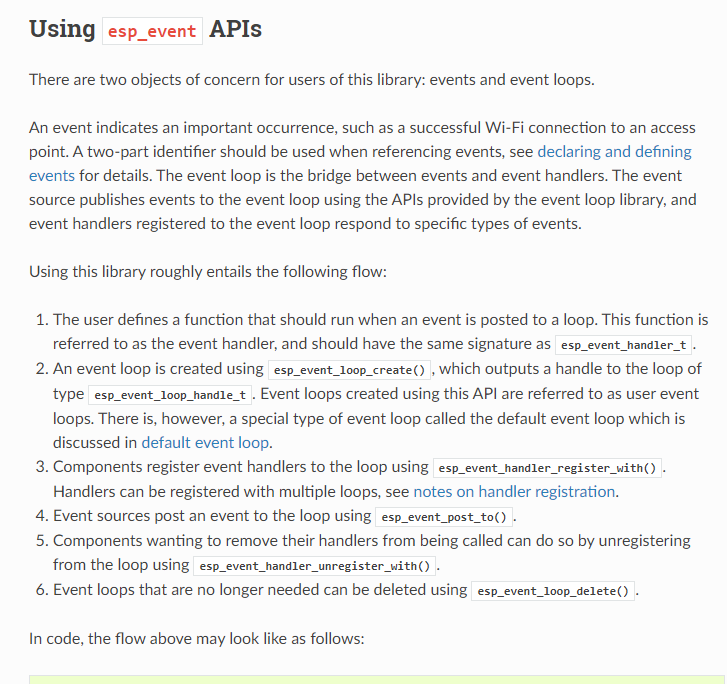
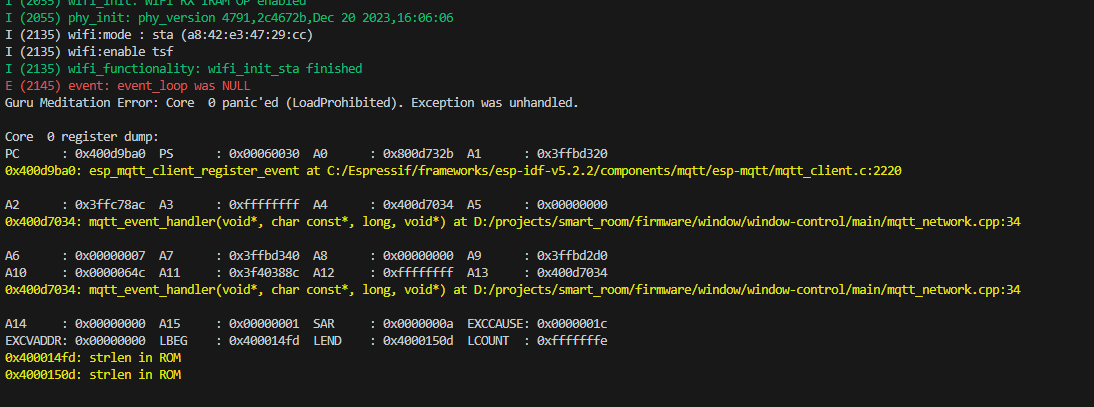
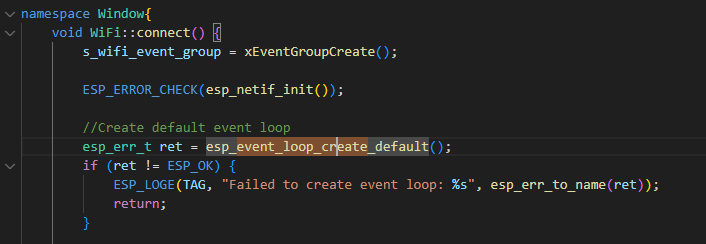
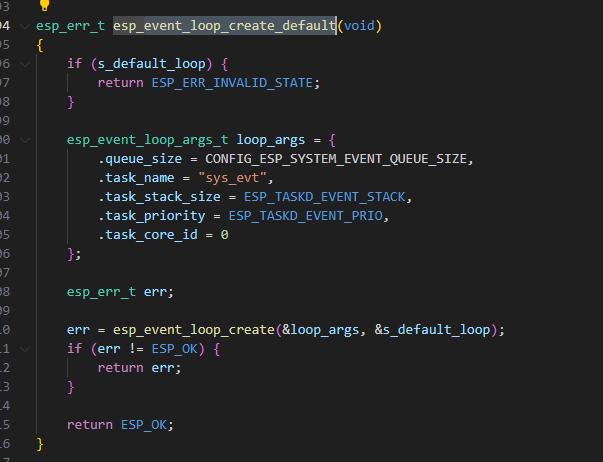
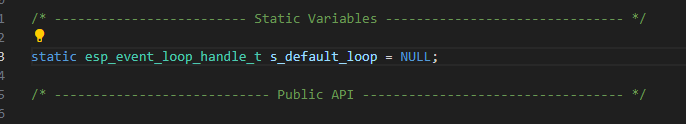
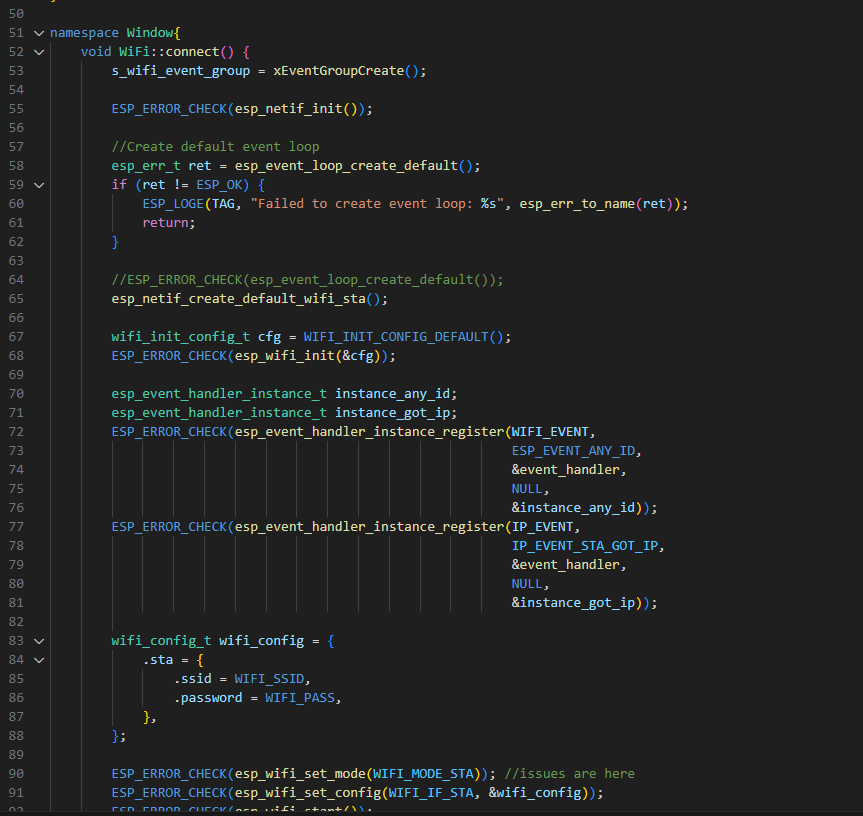
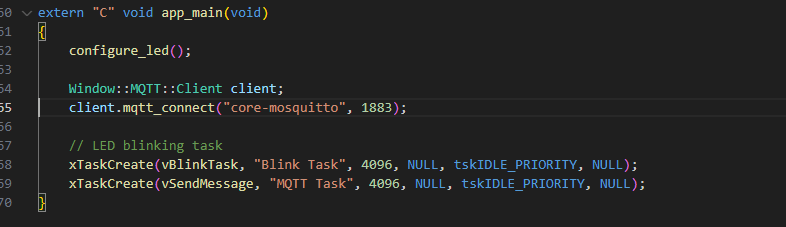
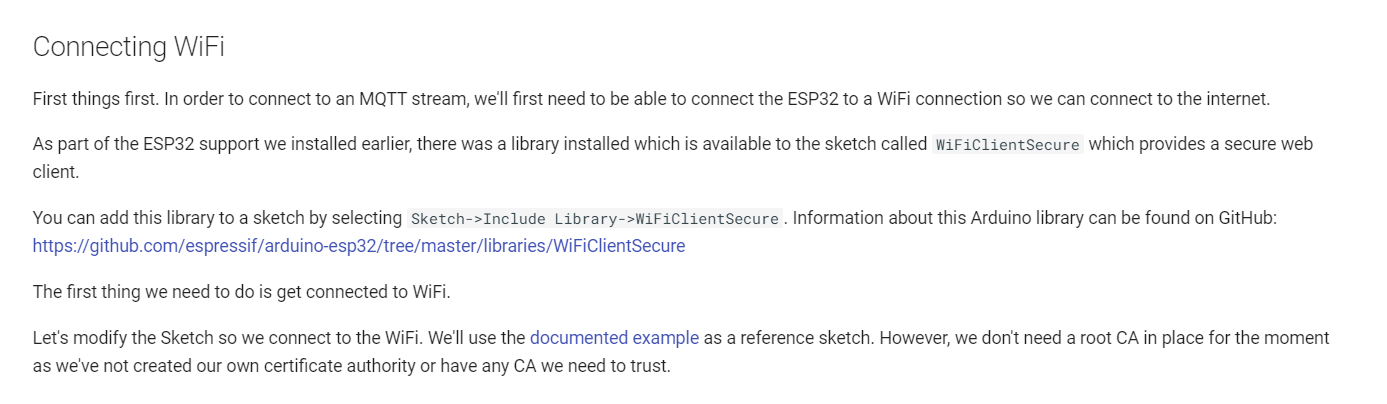
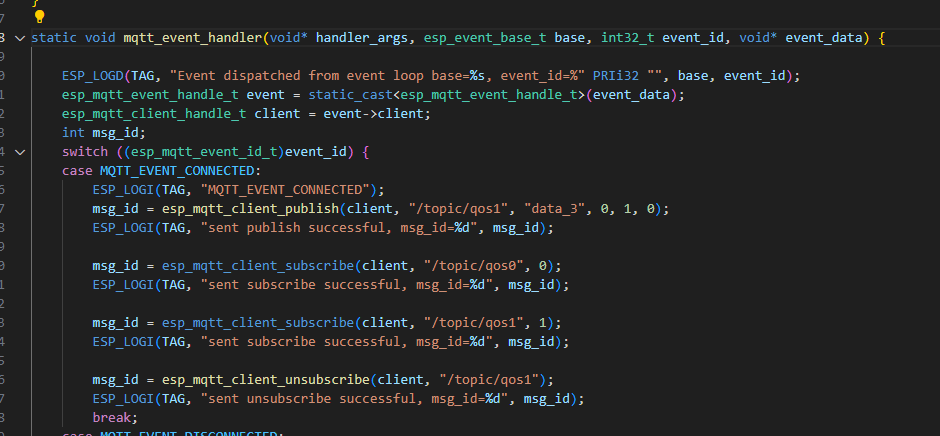
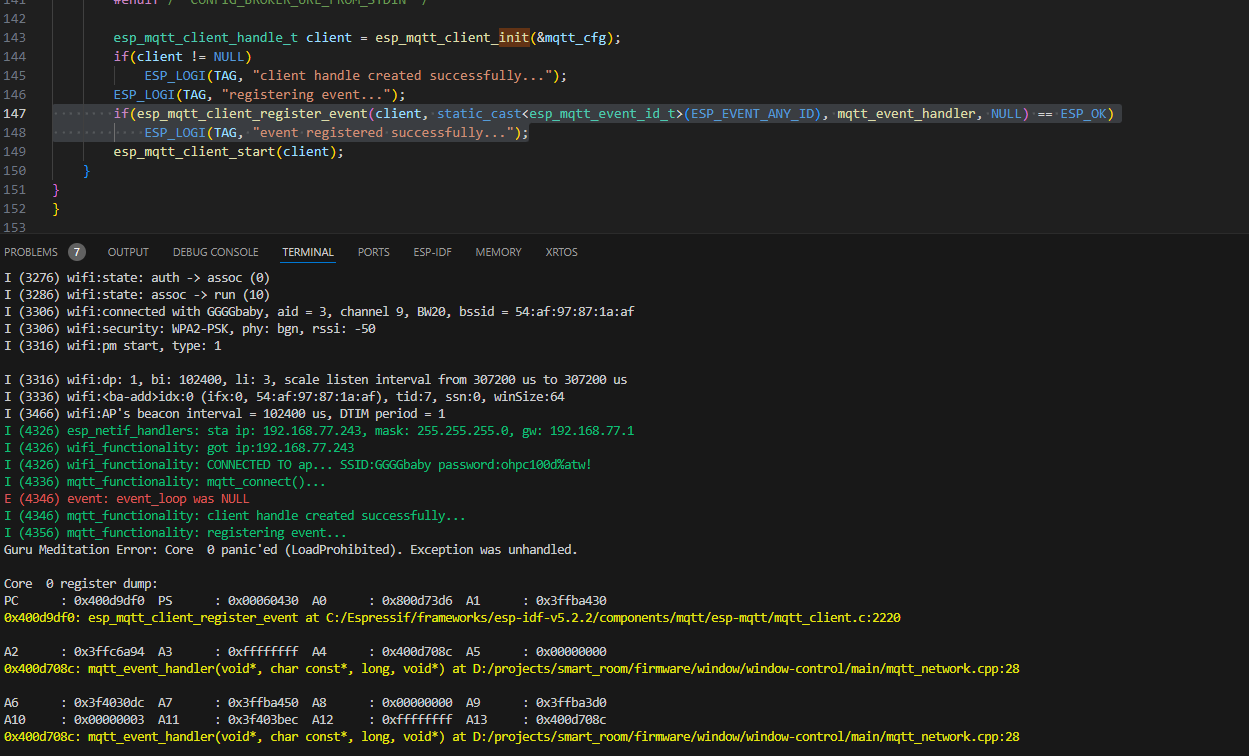
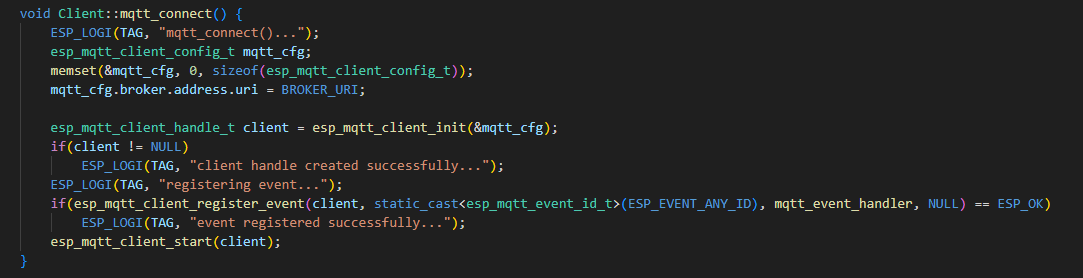
ESP\_ERROR\_CHECK failed: esp\_err\_t 0x103 (ESP\_ERR\_INVALID\_STATE) at 0x400d74eb

0x400d74eb: MQTT::Client::Client() at D:/projects/smart\_room/firmware/window/window-control/main/mqtt\_network.cpp:105 (discriminator 1)



file: "./main/mqtt\_network.cpp" line 105

func: MQTT::Client::Client()

expression: esp\_event\_loop\_create\_default()



LEARNED:

* find specific point where program fails, do this by printing and running error checks
* memset 0 for structs, to clear potential garbage data



Esp\_event\_loop\_create\_default

Returns ESP\_OK….

Default\_event\_loop.c

Default\_event\_loop.c

Wifi\_network.cpp



Event\_loop NULL means the event loop has not even been created.



I (468) main: Sending MQTT message...

I (468) mqtt\_functionality: [APP] Startup..

I (468) mqtt\_functionality: [APP] Free memory: 291936 bytes

I (478) mqtt\_functionality: [APP] IDF version: v5.2.2-dirty

I (458) main: Turning the LED OFF

I (1078) mqtt\_functionality: Other event id:7

E (1078) esp-tls: [sock=54] connect() error: Host is unreachable

E (1078) transport\_base: Failed to open a new connection: 32772

E (1078) mqtt\_client: Error transport connect

I (1088) mqtt\_functionality: MQTT\_EVENT\_ERROR

E (1088) mqtt\_functionality: Last error reported from esp-tls: 0x8004

I (1098) mqtt\_functionality: Last errno string (Success)

I (1108) mqtt\_functionality: MQTT\_EVENT\_DISCONNECTED

I (1508) main: Turning the LED ON

I (2518) main: Turning the LED OFF

I (3108) main: Sending MQTT message...

assert failed: tcpip\_send\_msg\_wait\_sem /IDF/components/lwip/lwip/src/api/tcpip.c:449 (Invalid mbox)

Tcpip.c below… possible hint?

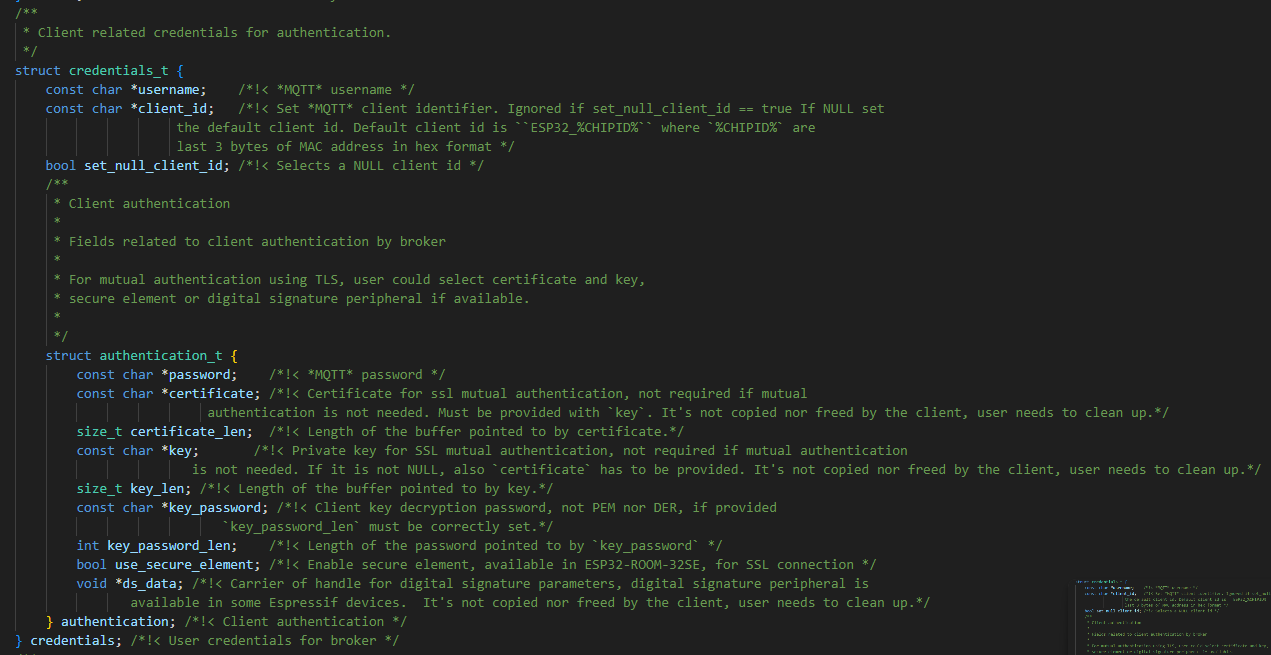


## MQTT

### address, broker

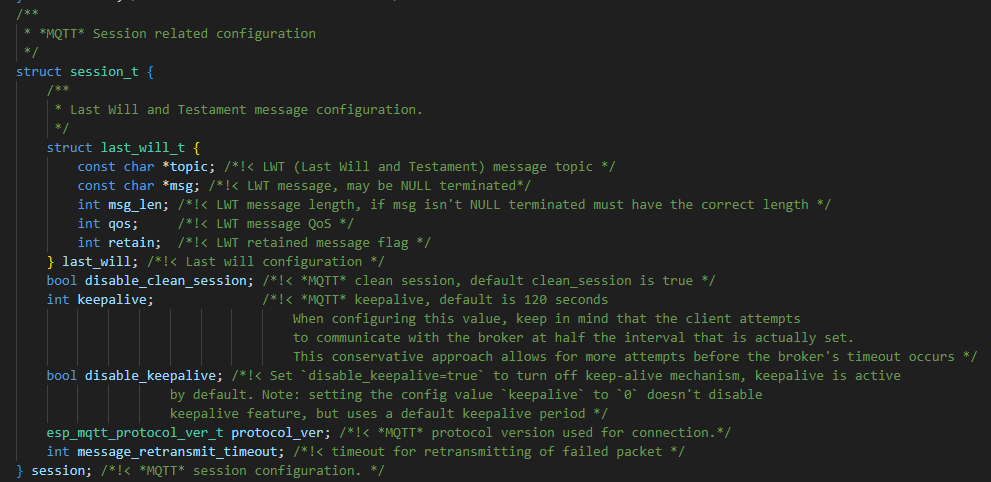


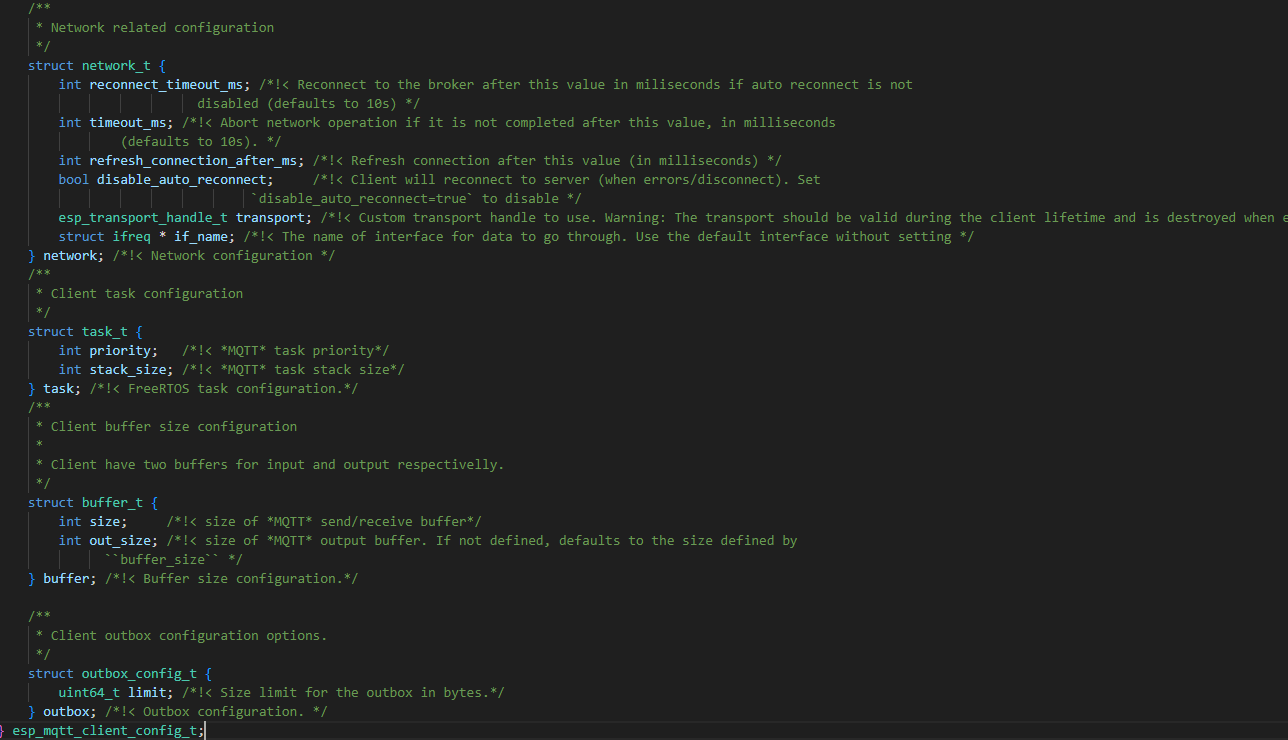
### Credentials





Session



Network, task, buffer, outbox