INSTITUTE OF TECHNOLOGY OF CAMBODIA

DEPARTMENT OF ELECTRICAL AND ENERGY ENGINEERING

Lab Part1 Report: TP-01 (ESP32)

PREPARED FOR:

Mr. TEP Sovichea

PREPARED BY:

HUT Hokkey	**e20180335**
CHHOUN Pichpisal	**e20180149**
HAK Menghour	**e20180239**

Engineering's Degree

Department of Electrical and Energy Engineering

Institute of Technology of Cambodia

2022-2023

Task

TP-01:

1, create the following task:

a, vButtonTask : handle button debounce and states using simple delay

b, vL1Task : blink L1 at the rate of 100ms

c, vL2Task : blink L2 at the rate of 500ms

2, Modify the code for the following function:

a, Long press B1 to delete/create vL1Task.

b, Press B2 to decrease L2 blink rate by 100ms. Reset back to 500ms when blink rate is equal to 0.

---

1. Create the following tasks:

a. vButton Task : handle button debounce and states using simple delay.

```
void vButtonTask(void *pvParameters)
{
    ButtonConfig_t button[2];
    for (int i = 0; i < 2; i++)
    {
        button[i] = BUTTON_CONFIG_DEFAULT;
    }

    button[0].gpio = GPIO_NUM_23;
    button[1].gpio = GPIO_NUM_22;


    gpio_config_t config;
    for  (int i = 0; i < 2; i++){
    config.pin_bit_mask = (1ULL << button[i].gpio);
    config.mode = GPIO_MODE_INPUT;
    config.pull_down_en = 0;
    config.pull_up_en = 1;
    config.intr_type = GPIO_INTR_DISABLE;
    gpio_config(&config);}

//  gpio_set_direction(BTN_GPIO, GPIO_MODE_INPUT);
//      //Disable pull up and down
//  gpio_set_pull_mode(BTN_GPIO, GPIO_FLOATING);
//  // Initialize pull-up input for B1 and B2
//  ESP_LOGI("vButtonTask", "Initializing GPIO....");

```

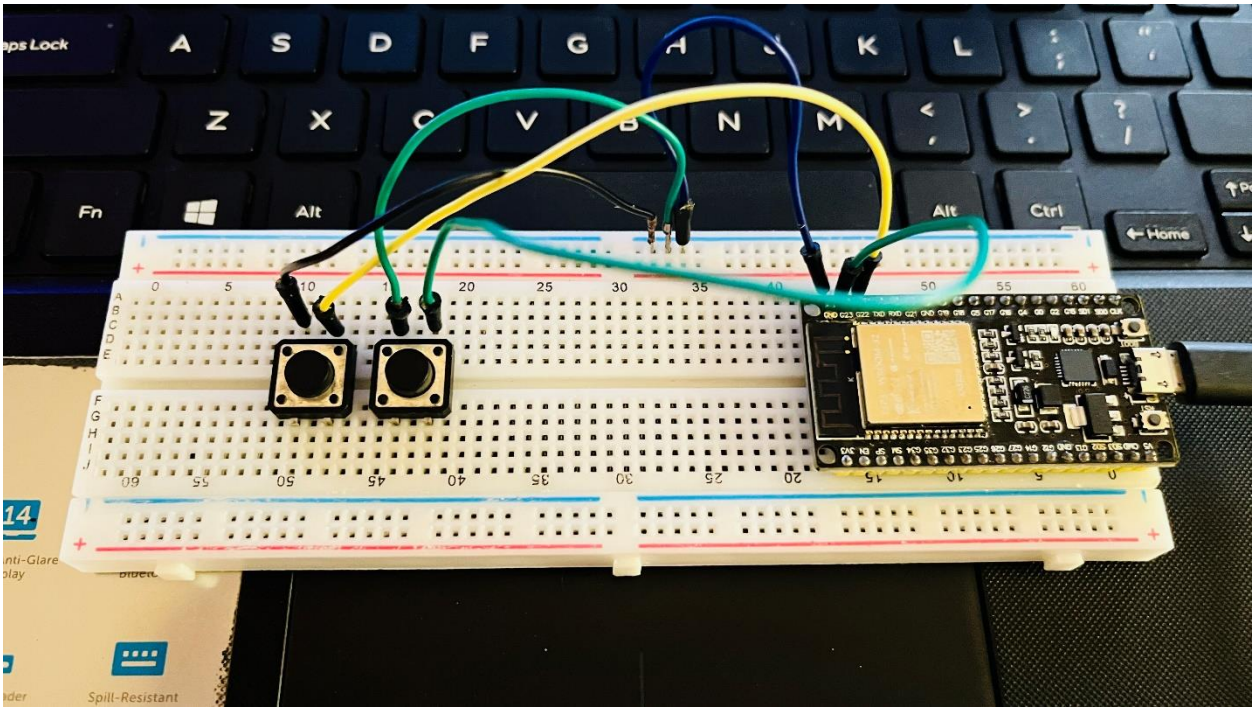

Figure 1: handle button debounce and states using simple delay.

b. vL1 Task: blink L1 at the rate of 100ms

```c
void vL1Task(void *pvParameters) {
    uint8_t state = 0;

    //Initializing LED1 GPIO as output push-pull
    gpio_config_t config;
    //For configure GPIO Pin
    //config.pin_bit_mask = (1ULL << LED1_GPIO) | (1ULL << 4)) | (1ULL << 5);
    config.pin_bit_mask = (1ULL << LED1_GPIO);
    config.mode = GPIO_MODE_OUTPUT;
    config.pull_down_en = GPIO_PULLDOWN_DISABLE;
    config.pull_up_en = GPIO_PULLUP_DISABLE;
    config.intr_type = GPIO_INTR_DISABLE;
    gpio_config(&config);

//  //simple GPIO configuration
//  gpio_set_direction(LED1_GPIO, GPIO_MODE_INPUT);
//        //Disable pull up and down
//  gpio_set_pull_mode(LED1_GPIO, GPIO_FLOATING);

    while (1) {
        gpio_set_level(LED1_GPIO, state);
        state = !state;
        vTaskDelay(pdMS_TO_TICKS(100));
    }

    vTaskDelete(NULL);

}
```

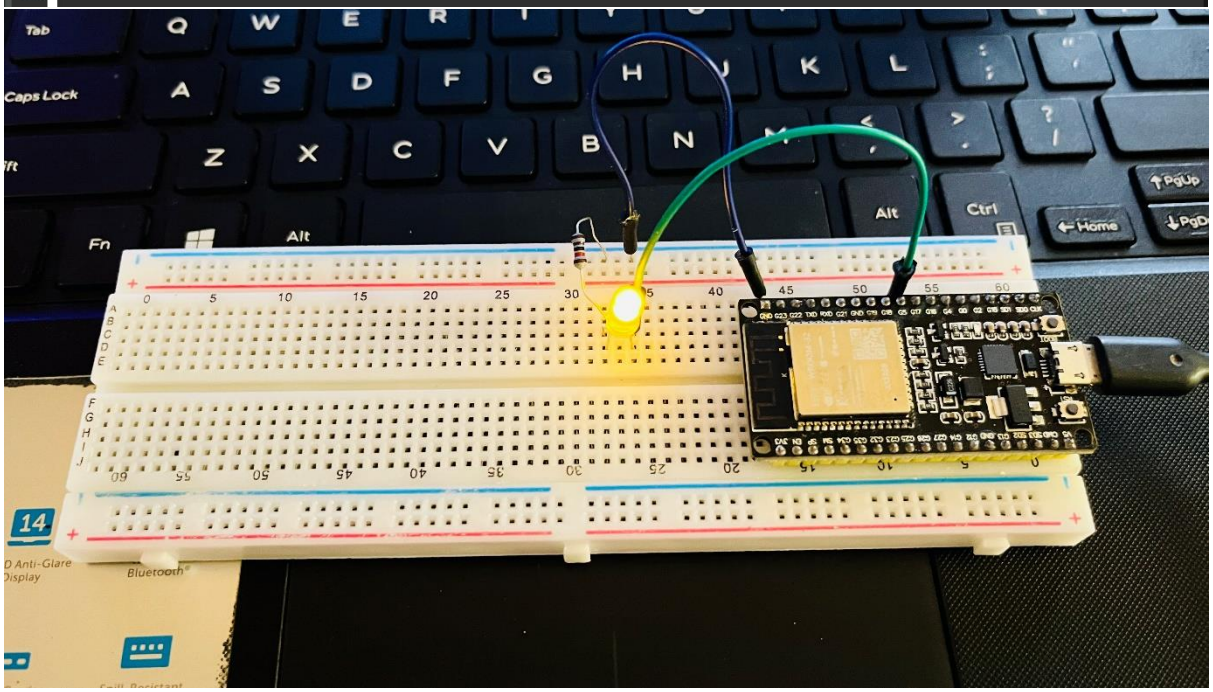

Figure 2: blink L1 at the rate of 100ms

c. vL2 Task: blink L2 at the rate of 500ms

```
void vL2Task(void *pvParameters) {


    uint8_t state = 0;

//  //simple GPIO configuration
//  gpio_set_direction(LED2_GPIO, GPIO_MODE_OUTPUT);
//  //Disable pull up and down
//  gpio_set_pull_mode(LED2_GPIO, GPIO_FLOATING);

    gpio_config_t config;
    config.pin_bit_mask = (1ULL << LED2_GPIO);
    config.mode = GPIO_MODE_OUTPUT;
    config.pull_down_en = GPIO_PULLDOWN_DISABLE;
    config.pull_up_en = GPIO_PULLUP_DISABLE;
    config.intr_type = GPIO_INTR_DISABLE;
    gpio_config(&config);

    while (1) {
        gpio_set_level(LED2_GPIO, state);
        state = !state;
        vTaskDelay(pdMS_TO_TICKS(old_delay));
    }

    vTaskDelete(NULL);
}
```

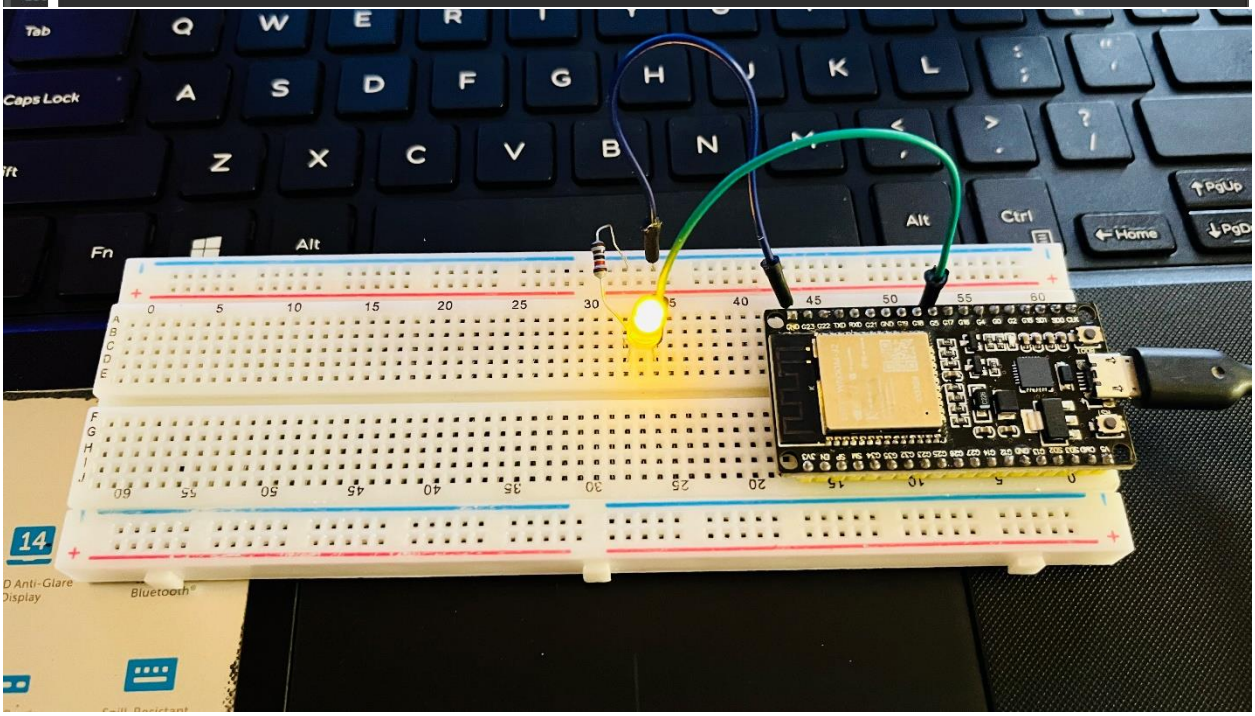

Figure 3: vL2 Task: blink L2 at the rate of 500ms

2. Modify the code for the following functions

a. Long press B1 to delete / create vLTask

```c
typedef struct {
    uint8_t gpio;
    uint8_t value;
    uint8_t longPressCounter;
    uint16_t longPressDuration;
    eButtonState_t nextState;
    eButtonState_t currentState;
} ButtonConfig_t;

ButtonConfig_t BUTTON_CONFIG_DEFAULT = {
        .gpio = GPIO_NUM_MAX,
        .value = 1,
        .longPressCounter = 0,
        .longPressDuration = 60, // duration * delay = 60 * 50 = 3000 millisecond
        .nextState = BUTTON_RELEASED,
        .currentState = BUTTON_RELEASED,
};

void vL1Task(void *pvParameters); //prototype function when use with callback function
static void buttonPressCallback(int btn_idx, eButtonState_t btn_state) {
    // handle button event here

    if (btn_idx == 0 && btn_state == BUTTON_LONG_PRESSED) {

        if (TaskHandle_L1 != NULL){
            vTaskSuspend(TaskHandle_L1);
            printf("Suspended");
            vTaskDelete(TaskHandle_L1);
        }
        else {
            vTaskResume(TaskHandle_L1);
            printf("Resumed");
        }


    }
```

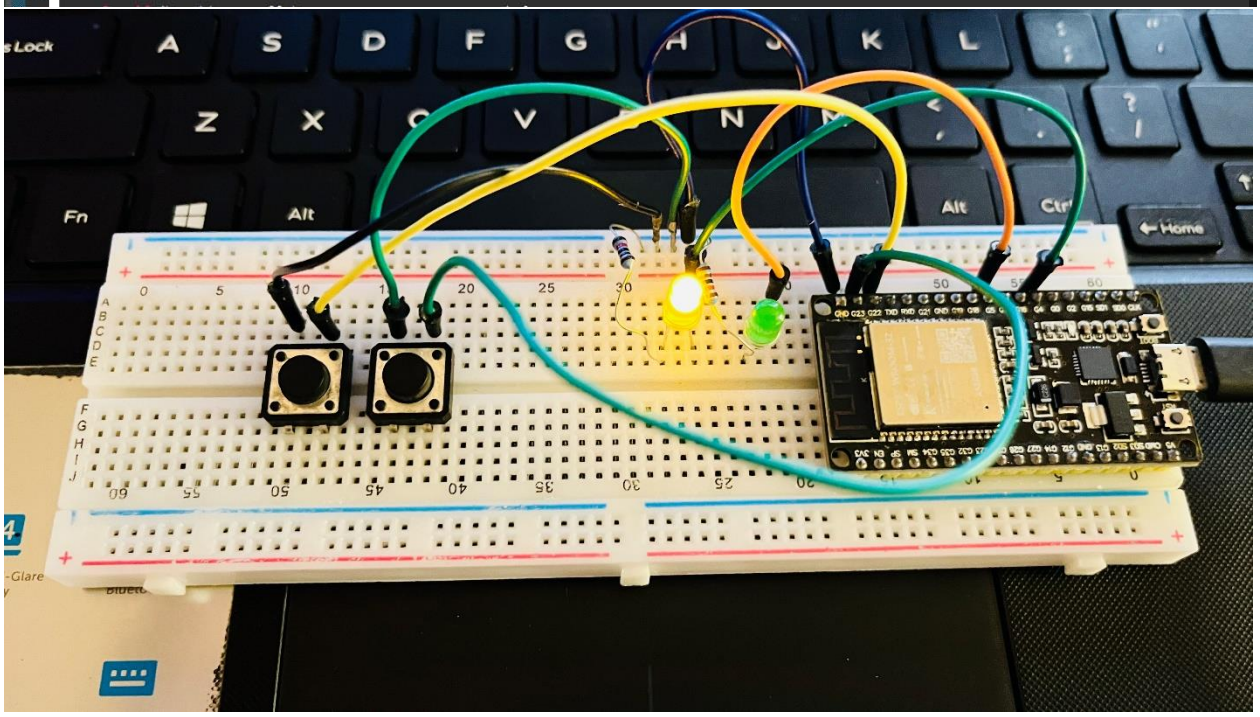

Figure 4: Long press B1 to delete / create vLTask.

b. Press B2 to decrease L2 blink rate by 100ms. Reset back to 500ms when blink rate is equal to 0.

```
    else if (btn_idx == 1 && btn_state == BUTTON_PRESSED) {

        if ( old_delay == 500) {
            new_delay = old_delay - 100;
            old_delay = new_delay;
            printf("blink rate is 400ms");
        }

        else if (old_delay == 400) {
            new_delay = old_delay - 100;
            old_delay = new_delay;
            printf("Blink rate is 300ms");
        }

        else if (old_delay == 300) {
                new_delay = old_delay - 100;
                old_delay = new_delay;
                printf("Blink rate is 200ms");
        }

        else if (old_delay == 200) {
                new_delay = old_delay - 100;
                old_delay = new_delay;
                printf("Blink rate is 100ms");
        }

        else if (old_delay == 100) {
                new_delay = old_delay - 100; //new_delay = 0 so blink rate is 0 too
                old_delay = 500;
                printf("Reset to 500ms");
        }

    }
}
```

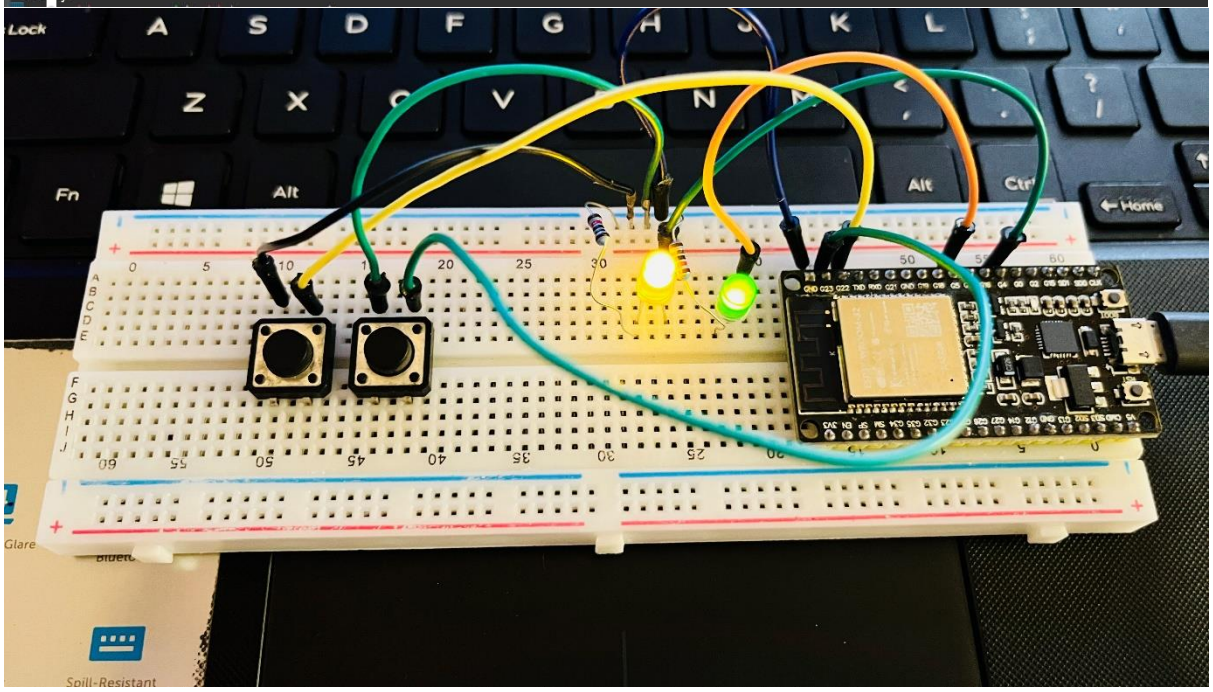

Figure 5: Press B2 to decrease L2 blink rate by 100ms. Reset back to 500ms when blink rate is equal to 0.

Conclusion

After finish TP-01, we understood the process of using **Espressif-IDE** and coding in ESP32. We are also able to configuration the button to handle LED with delay and many more task by coding in **Espressif-IDE**.