Operating System Laboratory

Course: DJ19ELEC6022

ESP32 - DevKitC

Task 1 – Blinking Internal on board led core 1 and hello world on core 0

Prof. in charge: Mayur Parulekar

Batch: B4

Name: Tushar Kumar

SAP ID: 60001190055

Third Year Dept. of Electronics

Electronics Engineering

Dwarkadas J. Sanghvi College of Engineering

Mumbai University

2020 - 2023

Task 1 – Blinking Internal on board led on core 1 & helloworld on core 0

Aim: Blinking an on board LED on core 1 & helloworld on core 0.

Producer:

Steps for working with ESP_IDF:

1. Launch Espressif IDE

2. click on file select ESP_IDF new project

3. All required Files are automatically loaded.

4. Select the project click on file to add ESP_IDF components then required files are loaded.

5. load your code in c file.

6. build project.

7. Select target esp32 and COM port.

8. Flash the code on esp32.

9. Open the monitor to see the desired output.

Code:

```
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/gpio.h"
#include "esp_log.h"
#include "led_strip.h"
#include "sdkconfig.h"

TaskHandle_t Task1;
TaskHandle_t Task2;

gpio_num_t led1 = 2;
gpio_num_t led2 = 4;

void Task1code(void *p)
{
   gpio_reset_pin(led1);
   gpio_set_direction(led1, GPIO_MODE_OUTPUT);
   printf("Task 1 is running on core:\n");
   printf("%d\n", xPortGetCoreID());

   while(1)
   {

      gpio_set_level(led1, 0);
      vTaskDelay(1000/portTICK_PERIOD_MS);
      gpio_set_level(led1, 1);
      vTaskDelay(1000/portTICK_PERIOD_MS);
   }
}
```

```
void Task2code(void *p)
{
   gpio_reset_pin(led2);
   gpio_set_direction(led2, GPIO_MODE_OUTPUT);
   printf("Task 2 is running on core:\n");
   printf("%d\n", xPortGetCoreID());

   while(1)
   {
      gpio_set_level(led2, 0);
      vTaskDelay(1000/portTICK_PERIOD_MS);
      gpio_set_level(led2, 1);
      vTaskDelay(1000/portTICK_PERIOD_MS);
   }
}

void app_main(void)
{
   xTaskCreatePinnedToCore(
               Task1code,   /* Task function. */
               "Task1",     /* name of task. */
               10000,       /* Stack size of task */
               NULL,        /* parameter of the task */
               1,           /* priority of the task */
               &Task1,      /* Task handle to keep track of created task */
               0);
   vTaskDelay(50/portTICK_PERIOD_MS);

   xTaskCreatePinnedToCore(
               Task2code,   /* Task function. */
               "Task2",     /* name of task. */
               10000,       /* Stack size of task */
               NULL,        /* parameter of the task */
               1,           /* priority of the task */
               &Task2,      /* Task handle to keep track of created task */
```

```
            1);
    vTaskDelay(50/portTICK_PERIOD_MS);
}
```

Conclusion: We have successfully separated the tasks into two cores

Operating System Laboratory

Course: DJ19ELEC6022

ESP32 - DevKitC

Task 2 – Interfacing 16*2 LCD and displaying Hello World.

Prof. in charge: Mayur Parulekar

Batch: B4

Name: Tushar Kumar

SAP ID: 60001190055

Third Year Dept. of Electronics

Electronics Engineering

Dwarkadas J. Sanghvi College of Engineering

Mumbai University

2020 - 2023

Task2 – Interfacing 16*2 LCD and displaying Hello World.

Aim: LCD interfacing and displaying of the HELLO WORLD text on display.

Producer:

Steps for working with ESP_IDF:

1. Launch Espressif IDE

2. click on file select ESP_IDF new project

3. All required Files are automatically loaded.

4. Select the project click on file to add ESP_IDF components then required files are loaded.

5. load your code in c file.

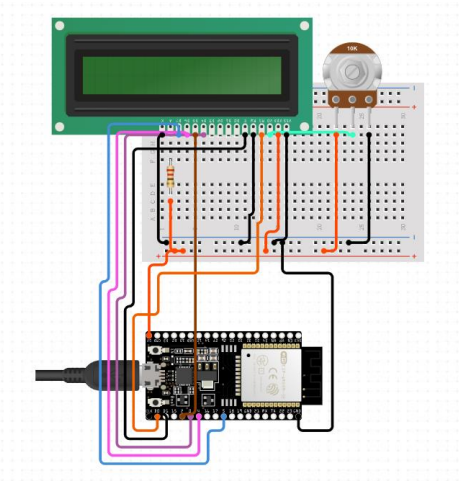
6. build project.

7. Select target esp32 and COM port.

8. Flash the code on esp32.

9. Open the monitor to see the desired output.

Circuit diagram:



Code:

```
void lcd_init();
void lcd_cmd(unsigned char);
void lcd_data(unsigned char);
void lcd_decode(unsigned char);
void lcd_string(unsigned char*);

int lcd_pins[11] = {12,13,14,18,25,26,27,32,33,5,19}; // ESP32 PINS
unsigned char msg[] = "Hello World";

void app_main(void)
{

printf("Code has started\n");
while(1)
{
   printf("printing msg on LCD\n");
   lcd_init();
}

}

void lcd_init()
{
   // initialize pins
   for(int i =0;i<11;i++)
   {
      gpio_pad_select_gpio(lcd_pins[i]);
      gpio_set_direction(lcd_pins[i],GPIO_MODE_OUTPUT);
   }

   lcd_cmd(0x38); // config lcd in 8-bit mode
   lcd_cmd(0x01); // clear display
   lcd_cmd(0x0E); // dispay cursor on and display on
   lcd_cmd(0x80); // set cursor to first line
```

```
    lcd_string(msg); // print this string on LCD
}

void lcd_decode(unsigned char info)
{
    unsigned char temp;
    for(int i =0;i<8;i++)
    {
        temp=pow(2,i);
        gpio_set_level(lcd_pins[i],(info&temp));
    }
}

void lcd_cmd(unsigned char cmd)
{
    lcd_decode(cmd);
    gpio_set_level(lcd_pins[8],0); //RS=0
    gpio_set_level(lcd_pins[9],0); //RW=0
    gpio_set_level(lcd_pins[10],1); //E=1
    vTaskDelay(10/portTICK_PERIOD_MS);
    gpio_set_level(lcd_pins[10],0);
    vTaskDelay(10/portTICK_PERIOD_MS);
}

void lcd_data(unsigned char data)
{
    lcd_decode(data);
    gpio_set_level(lcd_pins[8],1); //RS=1
    gpio_set_level(lcd_pins[9],0); //RW=0
    gpio_set_level(lcd_pins[10],1); //E=1
    vTaskDelay(10/portTICK_PERIOD_MS);
    gpio_set_level(lcd_pins[10],0);
    vTaskDelay(10/portTICK_PERIOD_MS);
}
```

```
void lcd_string(unsigned char *p)
{
   while(*p !='\0')
   {
      lcd_data(*p);
      p=p+1;
   }
}
```

Operating System Laboratory

Course: DJ19ELEC6022

ESP32 - DevKitC

Task 3 – Unicore LED Blinking an Display Hello World on console.

Prof. in charge: Mayur Parulekar

Batch: B4

Name: Tushar Kumar

SAP ID: 60001190055

Third Year Dept. of Electronics

Electronics Engineering

Dwarkadas J. Sanghvi College of Engineering

Mumbai University

2020 – 2023

Task 3 – Unicore LED Blinking an Display Hello World on console.

Aim: Unicore task in which led gets blinked when HELLO WORLD is displayed on the LCD

Producer:

Steps for working with ESP_IDF:

1. Launch Espressif IDE

2. click on file select ESP_IDF new project

3. All required Files are automatically loaded.

4. Select the project click on file to add ESP_IDF components then required files are loaded.

5. load your code in c file.

6. build project.

7. Select target esp32 and COM port.

8. Flash the code on esp32.

9. Open the monitor to see the desired output.

Code:

```
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/gpio.h"
#include "esp_log.h"
#include "led_strip.h"
#include "sdkconfig.h"



TaskHandle_t Task1;
TaskHandle_t Task2;

gpio_num_t led1 = 2;
gpio_num_t led2 = 4;

void Task1code(void *p)
{
   gpio_reset_pin(led1);
   gpio_set_direction(led1, GPIO_MODE_OUTPUT);
   printf("Task 1 is running on core:\n");
   printf("%d\n", xPortGetCoreID());

   while(1)
   {

      gpio_set_level(led1, 0);
      vTaskDelay(1000/portTICK_PERIOD_MS);
      gpio_set_level(led1, 1);
      vTaskDelay(1000/portTICK_PERIOD_MS);
   }
}

void Task2code(void *p)
```

```
{
   printf("Hello world");
}

void app_main(void)
{
   xTaskCreatePinnedToCore(
               Task1code,   /* Task function. */
               "Task1",     /* name of task. */
               10000,       /* Stack size of task */
               NULL,        /* parameter of the task */
               1,           /* priority of the task */
               &Task1,      /* Task handle to keep track of created task */
               0);
   vTaskDelay(50/portTICK_PERIOD_MS);

   xTaskCreatePinnedToCore(
               Task2code,   /* Task function. */
               "Task2",     /* name of task. */
               10000,       /* Stack size of task */
               NULL,        /* parameter of the task */
               1,           /* priority of the task */
               &Task2,      /* Task handle to keep track of created task */
               1);
   vTaskDelay(50/portTICK_PERIOD_MS);
}
```

Conclusion:

Programming on the ESP32 IDF is very similar to Arduino programming, and prior Arduino experience will undoubtedly aid in better understanding the code. Our programming environment has been successfully set up, and the Hello World and Blinking LED programmes have been implemented.

Operating System Laboratory

Course: DJ19ELEC6022

ESP32 – DevKitC

Task 4 – WIFI scanning code in ESP_IDF

Prof. in charge: Mayur Parulekar

Batch: B4

Name: Tushar Kumar

SAP ID: 60001190055

Third Year Dept. of Electronics

Electronics Engineering

Dwarkadas J. Sanghvi College of Engineering

Mumbai University

2020 – 2023

Task 4 – WIFI Scanning code in ESP_IDF

Aim: Wifi Scanning code in esp_idf

Producer:

Steps for working with ESP_IDF:

1. Launch Espressif IDE

2. click on file select ESP_IDF new project

3. All required Files are automatically loaded.

4. Select the project click on file to add ESP_IDF components then required files are loaded.

5. load your code in c file.

6. build project.

7. Select target esp32 and COM port.

8. Flash the code on esp32.

9. Open the monitor to see the desired output.

Code :

```
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "esp_event_loop.h"
#include "esp_wifi.h"
#include "nvs_flash.h"

#define STA_SSID "maayaz"
#define STA_PASSWORD "67156715"

void scann(){

  wifi_scan_config_t scan_config = {
    .ssid = 0,
    .bssid = 0,
    .channel = 0,
    .show_hidden = true
    };
  printf("Start scanning...");
  ESP_ERROR_CHECK(esp_wifi_scan_start(&scan_config, true));
  printf(" completed!\n");


  uint16_t ap_num;
  wifi_ap_record_t ap_records[20];
  ESP_ERROR_CHECK(esp_wifi_scan_get_ap_num(&ap_num));
  ESP_ERROR_CHECK(esp_wifi_scan_get_ap_records(&ap_num, ap_records));

  printf("Found %d access points:\n", ap_num);

  printf("               SSID              | Channel | RSSI |   MAC \n\n");
```

```
 for(int i = 0; i < ap_num; i++)

   printf("%32s | %7d | %4d %2x:%2x:%2x:%2x:%2x:%2x \n", ap_records[i].ssid, ap_records[i].primary, ap_records[i].rssi , *ap_records[i].bssid, *(ap_records[i].bssid+1), *(ap_records[i].bssid+2), *(ap_records[i].bssid+3), *(ap_records[i].bssid+4), *(ap_records[i].bssid+5));


}



void app_main()
{
 ESP_ERROR_CHECK(nvs_flash_init());
 tcpip_adapter_init();
 wifi_init_config_t wifi_config = WIFI_INIT_CONFIG_DEFAULT();
 ESP_ERROR_CHECK(esp_wifi_init(&wifi_config));
 ESP_ERROR_CHECK(esp_wifi_set_mode(WIFI_MODE_STA));

 wifi_config_t sta_config = {
   .sta = {
     .ssid = STA_SSID,
     .password = STA_PASSWORD
   },
 };

 ESP_ERROR_CHECK(esp_wifi_set_config(ESP_IF_WIFI_STA, &sta_config));



 ESP_ERROR_CHECK(esp_wifi_start());
 //ESP_ERROR_CHECK(esp_wifi_connect());
 esp_err_t check = esp_wifi_connect();
 if(check == ESP_OK)
 {
  printf("FINALLY MEIN CONNECT HO GAYA\n");
 }

 while(1)
```

```
  {
      vTaskDelay(3000 / portTICK_RATE_MS);
      scann();
  }
}
```

Operating System Laboratory

Course: DJ19ELEC6022

ESP32 - DevKitC

Task 5 – Component

Prof. in charge: Mayur Parulekar

Batch: B4

Name: Tushar Kumar

SAP ID: 60001190055

Third Year Dept. of Electronics

Electronics Engineering

Dwarkadas J. Sanghvi College of Engineering

Mumbai University

2020 – 2023

Task 5 – Component

Producer:

Steps for working with ESP_IDF:

1. Launch Espressif IDE

2. click on file select ESP_IDF new project

3. All required Files are automatically loaded.

4. Select the project click on file to add ESP_IDF components then required files are loaded.

5. load your code in c file.

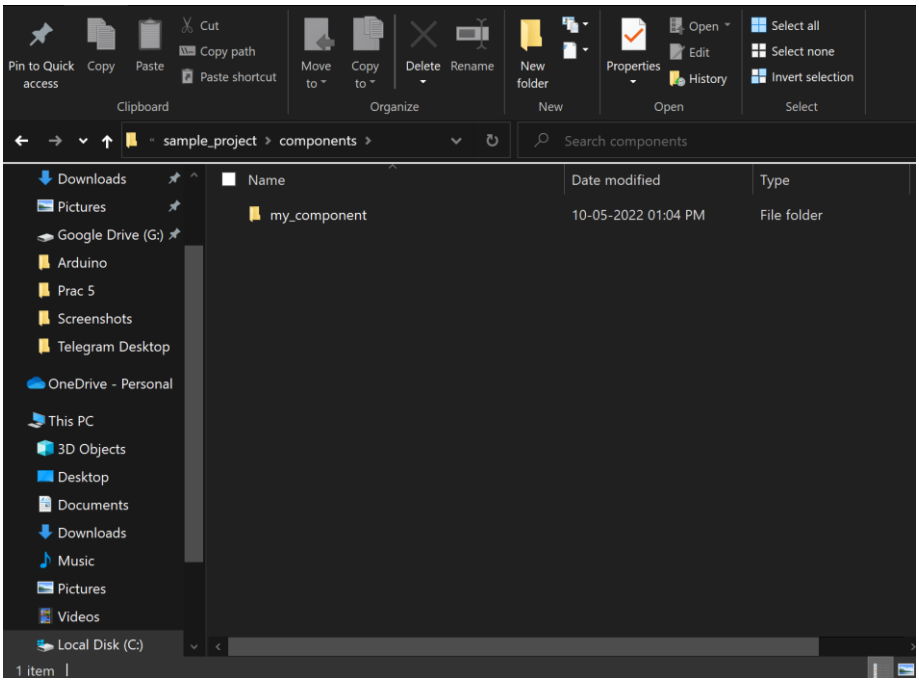
6. build project.

7. Select target esp32 and COM port.

8. Flash the code on esp32.

9. Open the monitor to see the desired output.

```
C:\Espressif\frameworks\esp-idf-v4.4\examples\get-started\sample_project>idf.py -C components create-component my_component
Executing action: create-component
The component was created in c:\espressif\frameworks\esp-idf-v4.4\examples\get-started\sample_project\components\my_component
```



Conclusion: We were able to build and flash the project but while monitoring we weren't able to see the output.