# National Textile University, Faisalabad


## Department of Computer Science

| | |
|---|---|
| **Name:** | MOHAMMAD ABU OBAIDA |
| **Class:** | BSCS 5$^{th}$B |
| **Registration No:** | 23-NTU-CS-1052 |
| **Course Name:** | Embedded IoT Systems |
| **Submitted To:** | Sir Nasir Mahmood |
| **Submission Date:** | 25 - OCT - 2025 |

# Question#3

## Task-1

**STATEMENT:** Use one button to cycle through LED modes (display the current state on the OLED):

**CODE:**

```
//Name : Abu Obaida
//REG NO : 23-NTU-CS-1052
//Class:BSCS-5th-B

#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>       // Graphics library for OLED
#include <Adafruit_SSD1306.h>   // OLED display driver library

// --- OLED display setup ---
#define W 128   // Screen width
#define H 64    // Screen height
Adafruit_SSD1306 scr(W, H, &Wire, -1);  // Create display object

// --- Pin connections ---
#define LED_Y 4         // Yellow LED pin
#define LED_G 0         // Green LED pin
#define LED_R 2         // Red LED pin
#define BTN_MODE 26     // Button 1 → used to change LED mode
#define BTN_RST 27      // Button 2 → used to reset to OFF mode

// --- PWM channel setup (ESP32 uses PWM channels for analog control) ---
#define CH_Y 0
#define CH_G 1
#define CH_R 2
#define FREQ 4000       // PWM frequency (Hz)
#define RES 8           // 8-bit resolution (0–255 duty range)

// --- Variables ---
hw_timer_t *blinkT = nullptr;  // Hardware timer for blinking
```

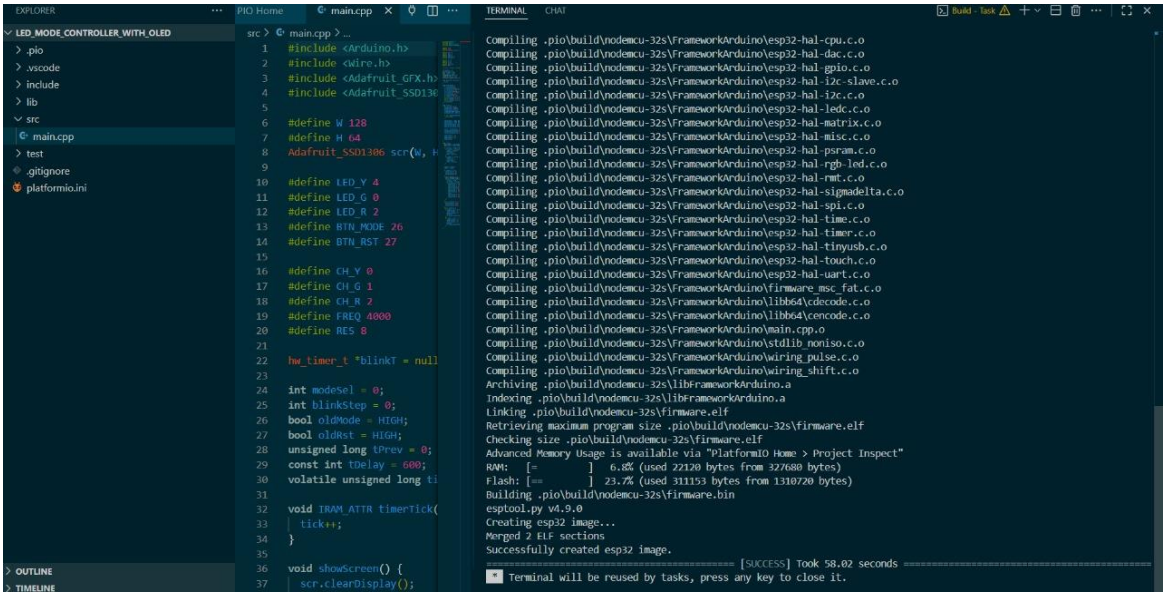
```
int modeSel = 0;               // Current LED mode (0=OFF,1=Blink,2=ON,3=PWM
fade)
int blinkStep = 0;             // Step for blinking colors
bool oldMode = HIGH;           // Stores previous button state
bool oldRst = HIGH;
unsigned long tPrev = 0;       // Stores last button press time
const int tDelay = 600;        // Debounce delay (ms)
volatile unsigned long tick = 0; // Counts timer ticks (used for blinking)

// --- Timer interrupt function ---
// Runs automatically every 1 second (in setup we set 1,000,000 µs = 1 sec)
void IRAM_ATTR timerTick() {
  tick++;  // Increase tick count each second
}

// --- Function to show current mode on OLED ---
void showScreen() {
  scr.clearDisplay();
  scr.setTextSize(2);
  scr.setTextColor(SSD1306_WHITE);
  scr.setCursor(15, 0);
  scr.println("LED PANEL");
  scr.drawLine(0, 20, 127, 20, SSD1306_WHITE);  // Divider line
  scr.setTextSize(1);
  scr.setCursor(10, 35);
  // Show text according to current mode
  if (modeSel == 0) scr.print("OFF");
  else if (modeSel == 1) scr.print("Blink");
  else if (modeSel == 2) scr.print("ON");
  else if (modeSel == 3) scr.print("PWM");
  scr.display();  // Update display
}

void setup() {
  Serial.begin(115200);

  // Set LED pins as output
  pinMode(LED_Y, OUTPUT);
  pinMode(LED_G, OUTPUT);
  pinMode(LED_R, OUTPUT);

  // Set button pins as input with pull-up resistors
  pinMode(BTN_MODE, INPUT_PULLUP);
  pinMode(BTN_RST, INPUT_PULLUP);
```

```
  // Initialize OLED display
  if (!scr.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    while (true);  // Stop if OLED fails
  }

  // Set up PWM for each LED
  ledcSetup(CH_Y, FREQ, RES);
  ledcSetup(CH_G, FREQ, RES);
  ledcSetup(CH_R, FREQ, RES);

  // Attach LEDs to PWM channels
  ledcAttachPin(LED_Y, CH_Y);
  ledcAttachPin(LED_G, CH_G);
  ledcAttachPin(LED_R, CH_R);

  // Set up hardware timer to trigger every 1 second
  blinkT = timerBegin(0, 80, true);          // Timer 0, prescaler 80 1 tick = 1
µs
  timerAttachInterrupt(blinkT, &timerTick, true);
  timerAlarmWrite(blinkT, 1000000, true);    // 1,000,000 µs = 1 second
  timerAlarmEnable(blinkT);

  // Start with all LEDs OFF
  ledcWrite(CH_Y, 0);
  ledcWrite(CH_G, 0);
  ledcWrite(CH_R, 0);

  showScreen();  // Display initial mode (OFF)
}

void loop() {
  bool nowMode = digitalRead(BTN_MODE);  // Read Mode button
  bool nowRst = digitalRead(BTN_RST);    // Read Reset button

  // --- Handle button presses (with debounce delay) ---
  if (millis() - tPrev > tDelay) {
    if (nowMode == LOW && oldMode == HIGH) {  // If Mode button pressed
      modeSel = (modeSel + 1) % 4;            // Go to next mode (0–3)
      blinkStep = 0;
      showScreen();                           // Update OLED
      tPrev = millis();
    }
    if (nowRst == LOW && oldRst == HIGH) {    // If Reset button pressed
      modeSel = 0;                            // Back to OFF
      blinkStep = 0;
```

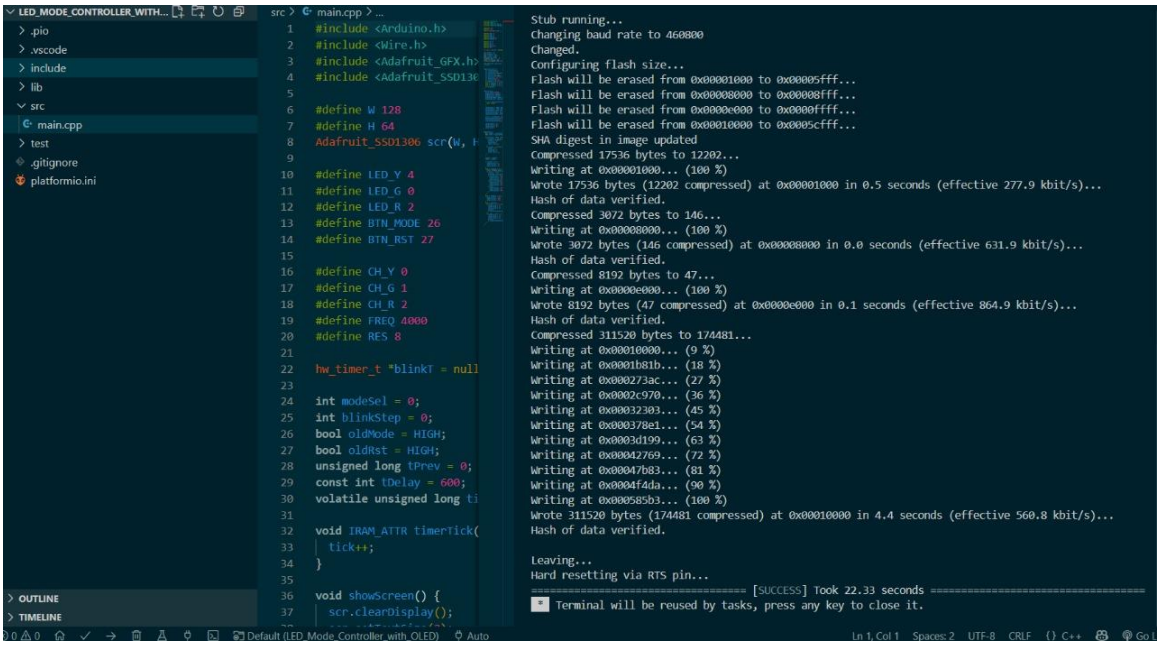
```
    showScreen();
    tPrev = millis();
  }
}

oldMode = nowMode;  // Save button states
oldRst = nowRst;

// --- LED behavior based on mode ---
if (modeSel == 0) {
  // Mode 0: All LEDs OFF
  ledcWrite(CH_Y, 0);
  ledcWrite(CH_G, 0);
  ledcWrite(CH_R, 0);
}
else if (modeSel == 1) {
  // Mode 1: Alternate blinking (changes every second)
  static unsigned long lastTick = 0;
  if (tick != lastTick) {         // Every time the timer ticks
    lastTick = tick;
    blinkStep = (blinkStep + 1) % 3;
    if (blinkStep == 0) {
      ledcWrite(CH_Y, 255);
      ledcWrite(CH_G, 0);
      ledcWrite(CH_R, 0);
    } else if (blinkStep == 1) {
      ledcWrite(CH_Y, 0);
      ledcWrite(CH_G, 255);
      ledcWrite(CH_R, 0);
    } else {
      ledcWrite(CH_Y, 0);
      ledcWrite(CH_G, 0);
      ledcWrite(CH_R, 255);
    }
  }
}
else if (modeSel == 2) {
  // Mode 2: All LEDs ON
  ledcWrite(CH_Y, 255);
  ledcWrite(CH_G, 255);
  ledcWrite(CH_R, 255);
}
else if (modeSel == 3) {
  // Mode 3: Smooth fading using PWM
  for (int i = 0; i <= 255 && modeSel == 3; i++) {
```

```
      ledcWrite(CH_Y, i);
      ledcWrite(CH_G, i);
      ledcWrite(CH_R, i);
      delay(5);
      if (digitalRead(BTN_MODE) == LOW || digitalRead(BTN_RST) == LOW) return; //
Stop fade if button pressed
    }
    for (int i = 255; i >= 0 && modeSel == 3; i--) {
      ledcWrite(CH_Y, i);
      ledcWrite(CH_G, i);
      ledcWrite(CH_R, i);
      delay(5);
      if (digitalRead(BTN_MODE) == LOW || digitalRead(BTN_RST) == LOW) return;
    }
  }
}
```

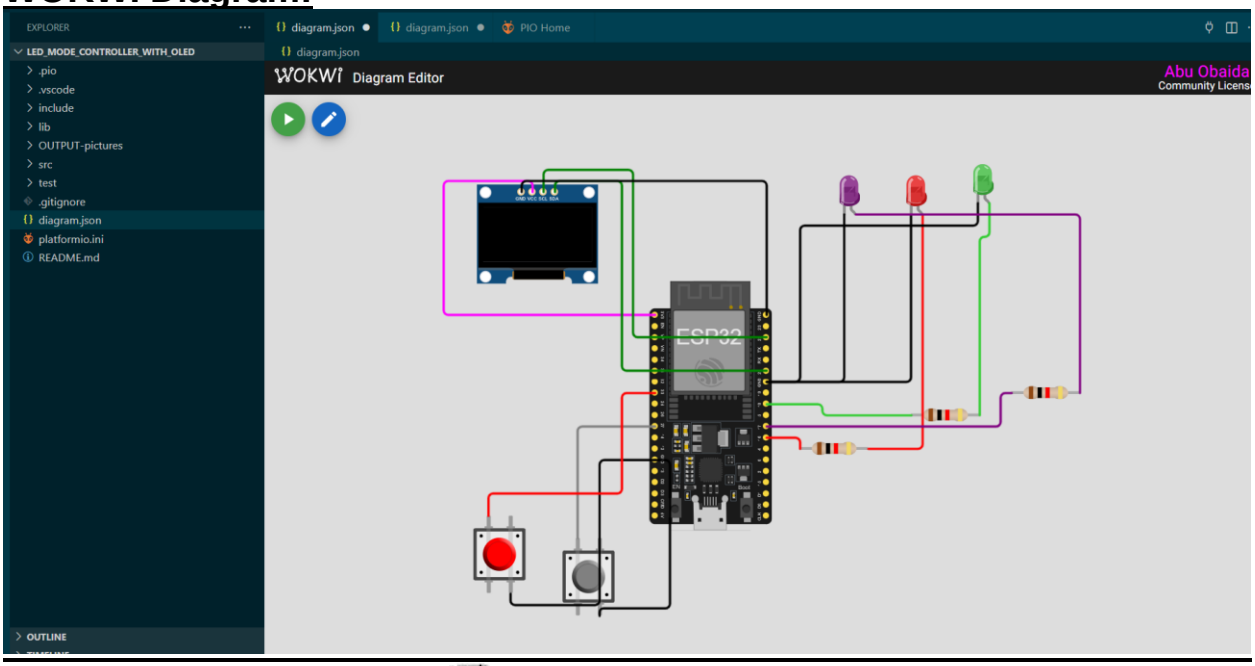
## BUILD:




## UPLOAD:



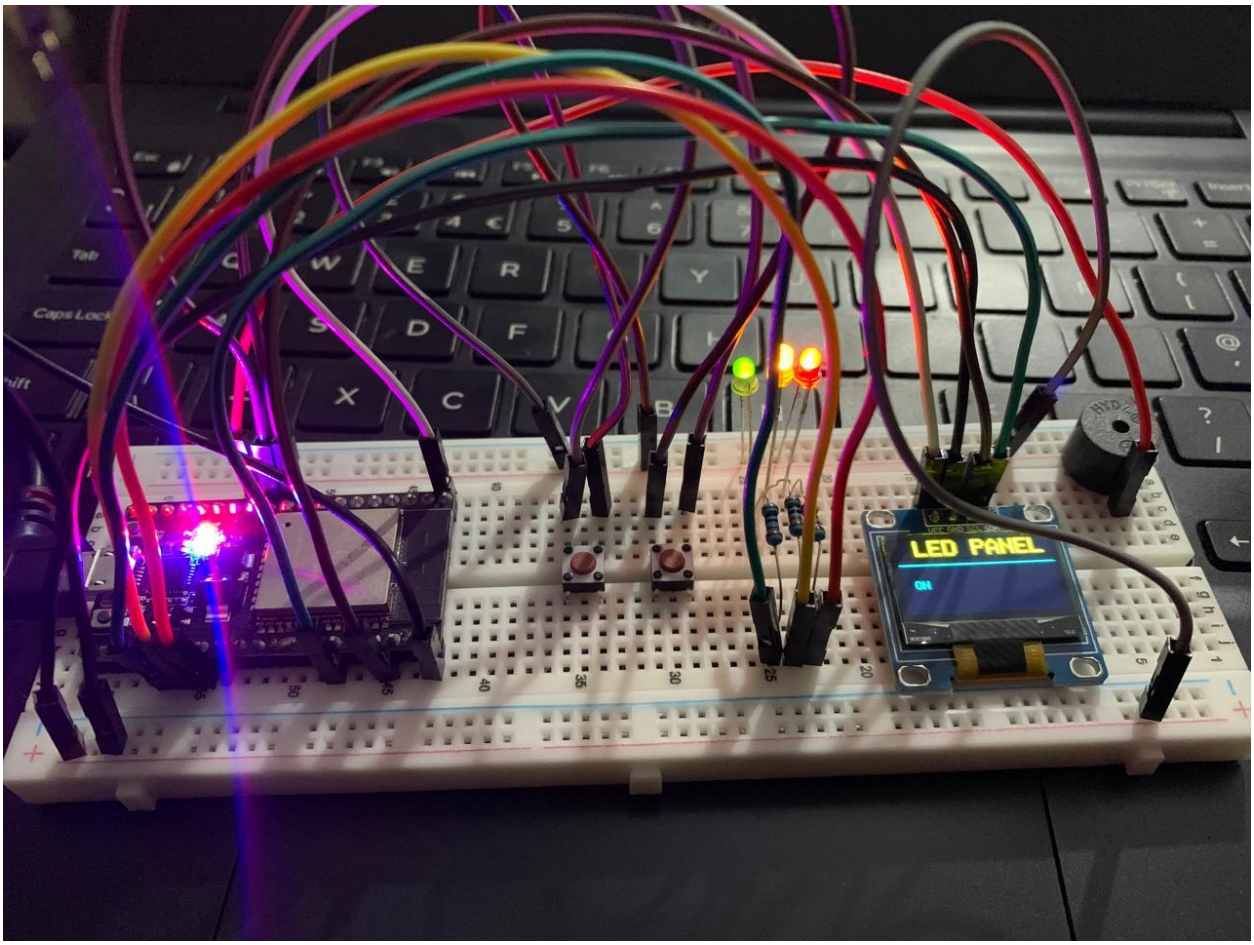

## WOKWI Diagram:

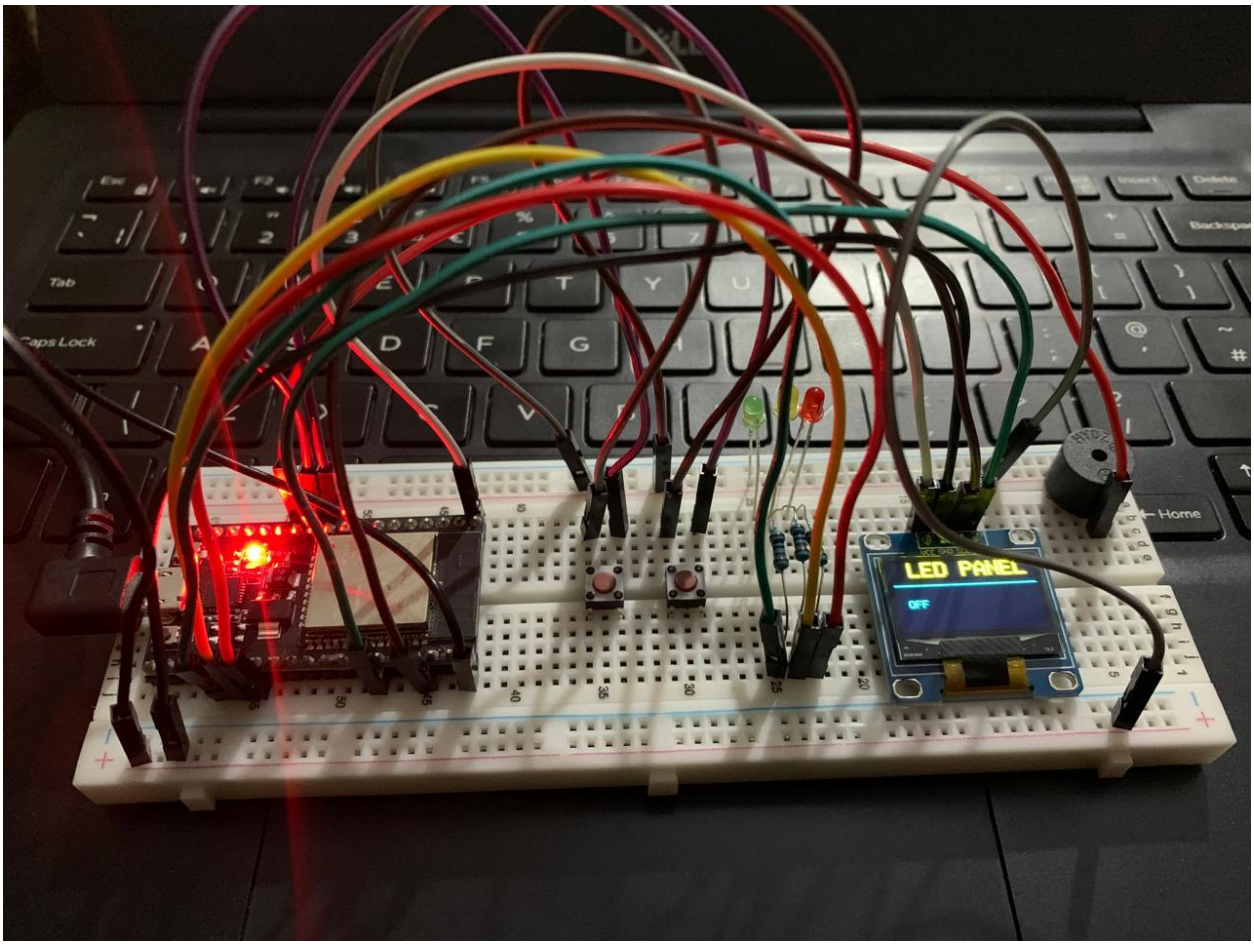



.mp4

vid1.mp4

## HARDWARE DEMO:


LED PANEL
ON




## Task-2

**STATEMENT:** Use a single button with press-type detection (display the event on the OLED):

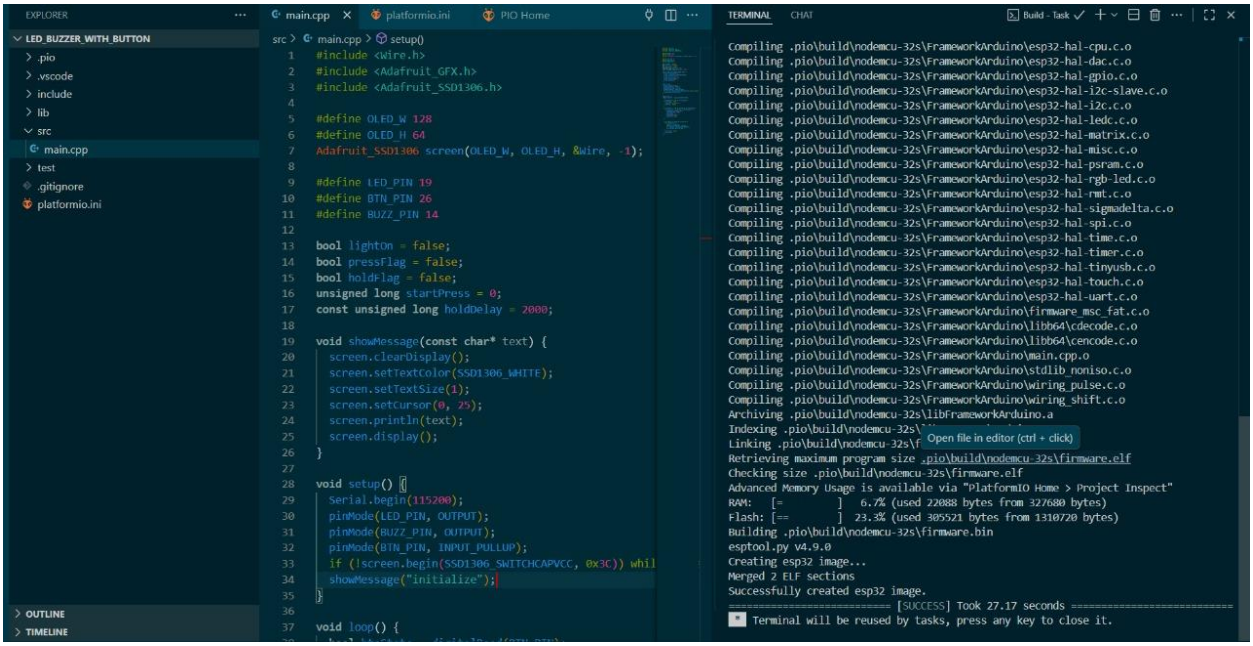
**CODE:**

```
//Name : Abu Obaida
//REG NO : 23-NTU-CS-1052
//Class:BSCS-5th-B

#include <Wire.h>                 // Library for I2C communication
#include <Adafruit_GFX.h>         // Graphics library for OLED display
#include <Adafruit_SSD1306.h>     // Library for controlling SSD1306 OLED

#define OLED_W 128                // OLED screen width in pixels
#define OLED_H 64                 // OLED screen height in pixels
Adafruit_SSD1306 screen(OLED_W, OLED_H, &Wire, -1);  // Create OLED object
```

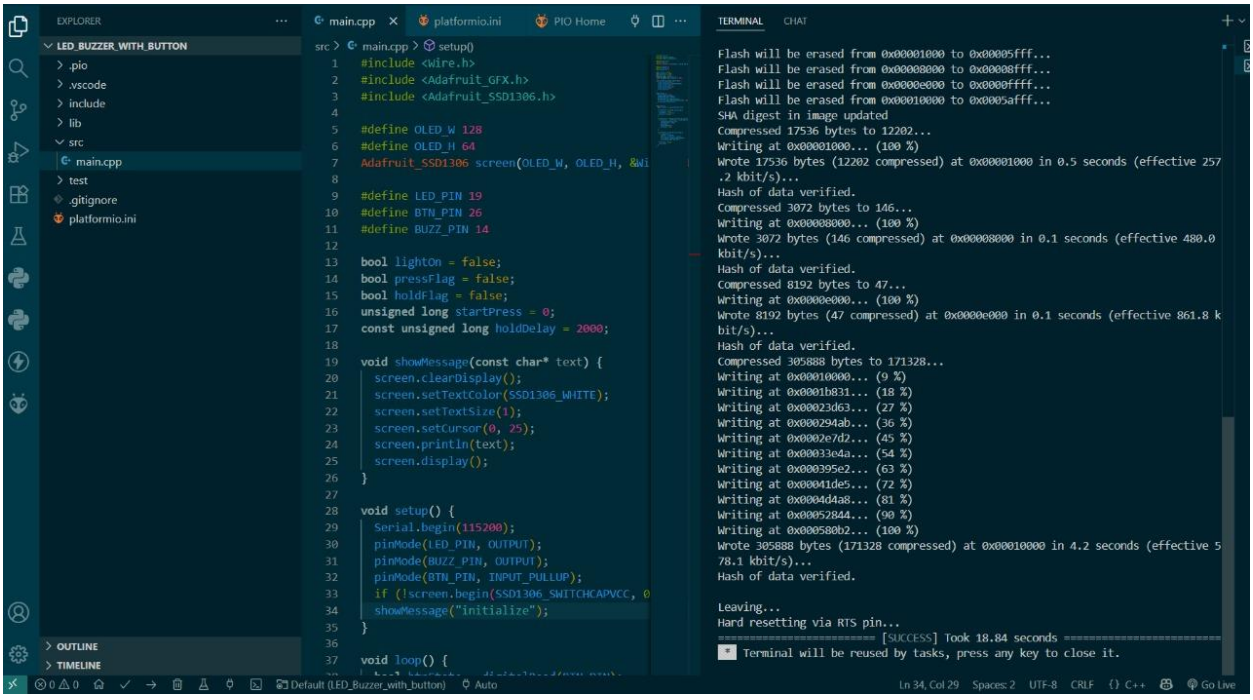
```
#define LED_PIN 19                // LED connected to pin 19
#define BTN_PIN 26                // Button connected to pin 26
#define BUZZ_PIN 14               // Buzzer connected to pin 14

bool lightOn = false;             // Keeps track if LED is ON or OFF
bool pressFlag = false;           // True when button is pressed
bool holdFlag = false;            // True when button is held for long
unsigned long startPress = 0;     // Stores the time when button was first
pressed
const unsigned long holdDelay = 2000;  // 2 seconds hold time to trigger buzzer

// Function to show a message on the OLED screen
void showMessage(const char* text) {
  screen.clearDisplay();          // Clear the screen
  screen.setTextColor(SSD1306_WHITE);  // Set text color to white
  screen.setTextSize(1);          // Set text size to small
  screen.setCursor(0, 25);        // Set text position (x=0, y=25)
  screen.println(text);           // Print the given text
  screen.display();               // Display it on the screen
}

void setup() {
  Serial.begin(115200);           // Start serial communication
  pinMode(LED_PIN, OUTPUT);       // Set LED pin as output
  pinMode(BUZZ_PIN, OUTPUT);      // Set buzzer pin as output
  pinMode(BTN_PIN, INPUT_PULLUP); // Set button pin as input with pull-up
resistor

  // Initialize the OLED display
  if (!screen.begin(SSD1306_SWITCHCAPVCC, 0x3C))
    while (true);  // Stop if the display is not found

  showMessage("initialize");      // Show startup message
}

void loop() {
  bool btnState = digitalRead(BTN_PIN);  // Read button state

  // When button is first pressed
  if (btnState == LOW && !pressFlag) {
    pressFlag = true;             // Mark that button is pressed
    startPress = millis();        // Record the time of press
    holdFlag = false;             // Reset hold flag
  }
```

```
  // Check if button is being held down
  if (btnState == LOW && pressFlag && !holdFlag) {
    if (millis() - startPress >= holdDelay) {  // Held for more than 2 seconds
      showMessage("Buzzer ON");    // Show message on screen
      tone(BUZZ_PIN, 1500);        // Turn buzzer ON (1500Hz tone)
      delay(500);                  // Wait for half a second
      noTone(BUZZ_PIN);            // Turn buzzer OFF
      holdFlag = true;             // Mark that long press was handled
    }
  }

  // When button is released
  if (btnState == HIGH && pressFlag) {
    if (!holdFlag) {               // If it was a short press
      lightOn = !lightOn;          // Toggle LED state
      digitalWrite(LED_PIN, lightOn); // Turn LED ON or OFF

      // Show LED status on screen
      if (lightOn) showMessage("LED ON");
      else showMessage("LED OFF");
    }
    pressFlag = false;             // Reset button press flag
    delay(250);                    // Small delay to avoid bouncing
  }
}
```

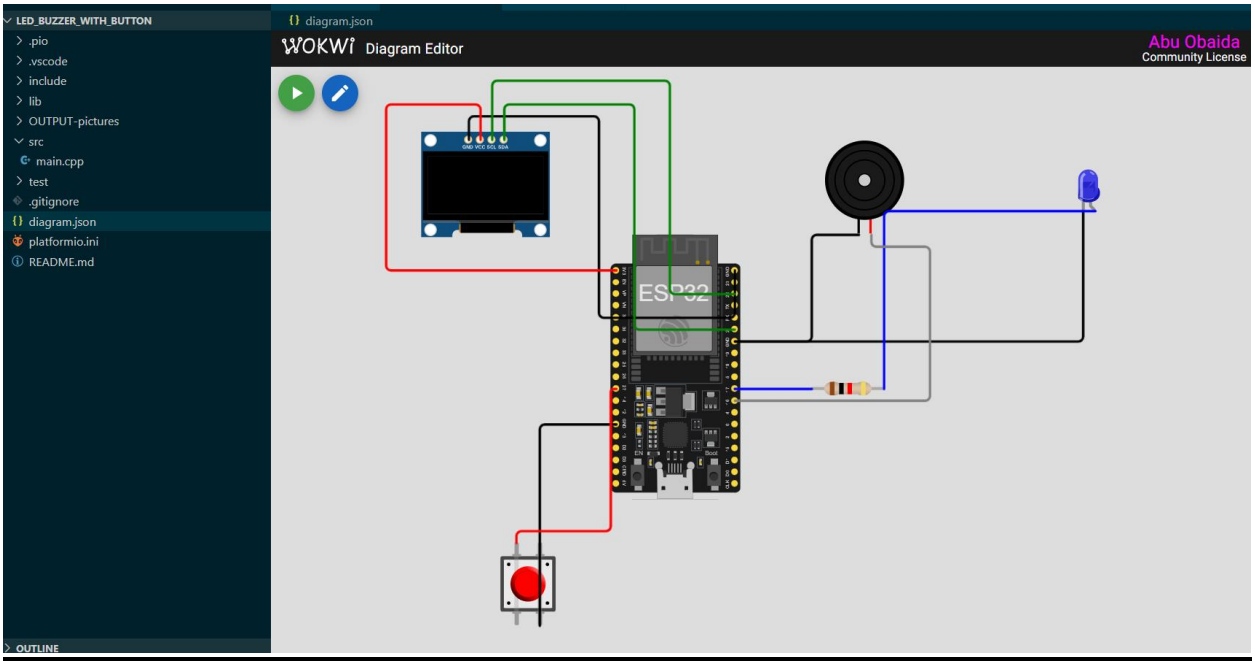
## BUILD:



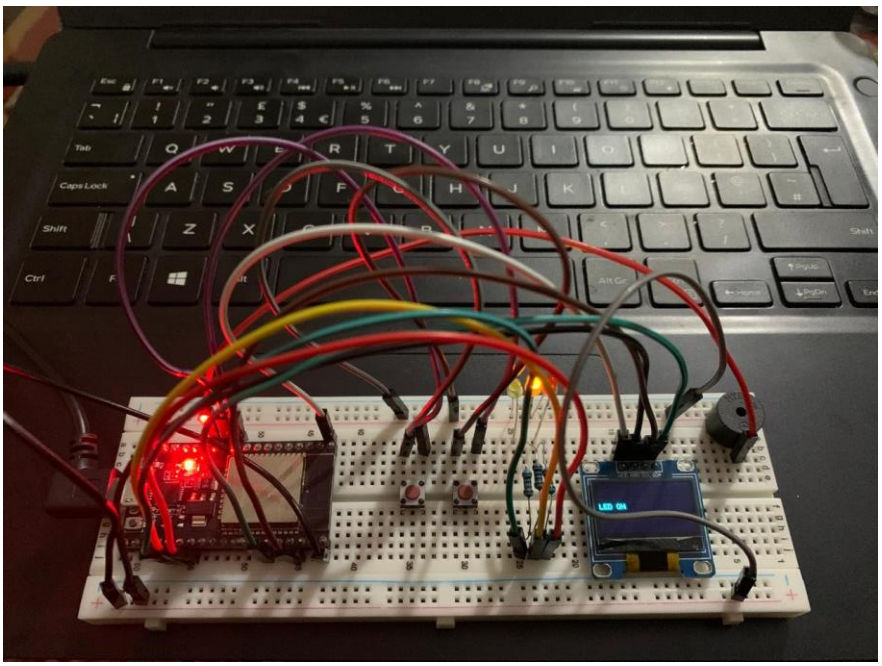

## UPLOAD:

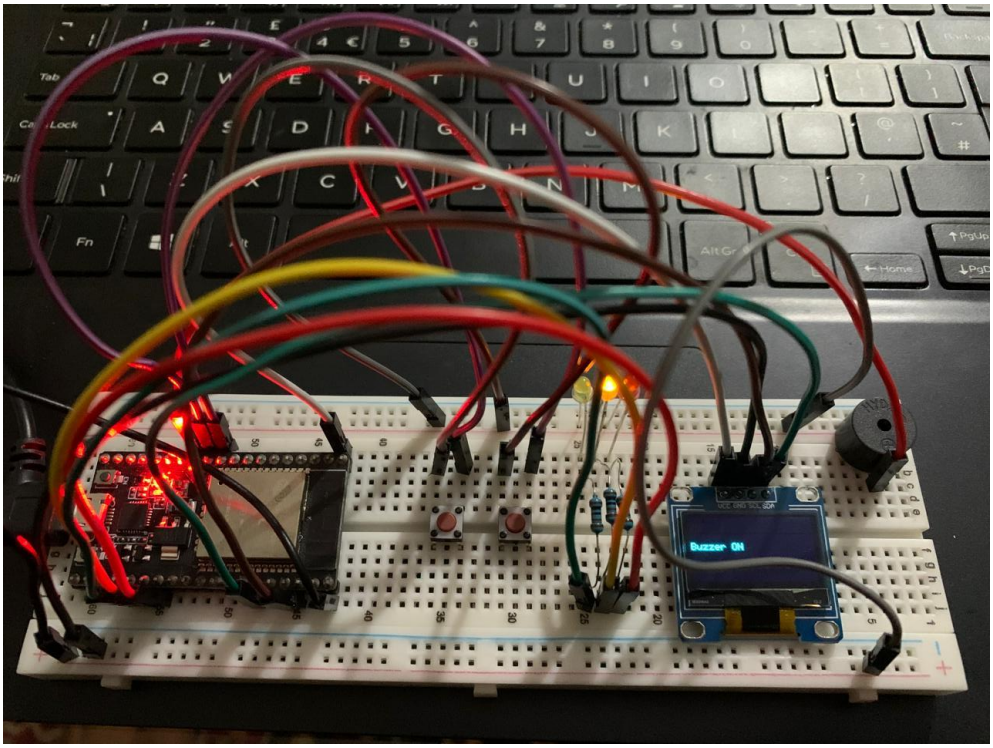



## WOKWI Diagram:

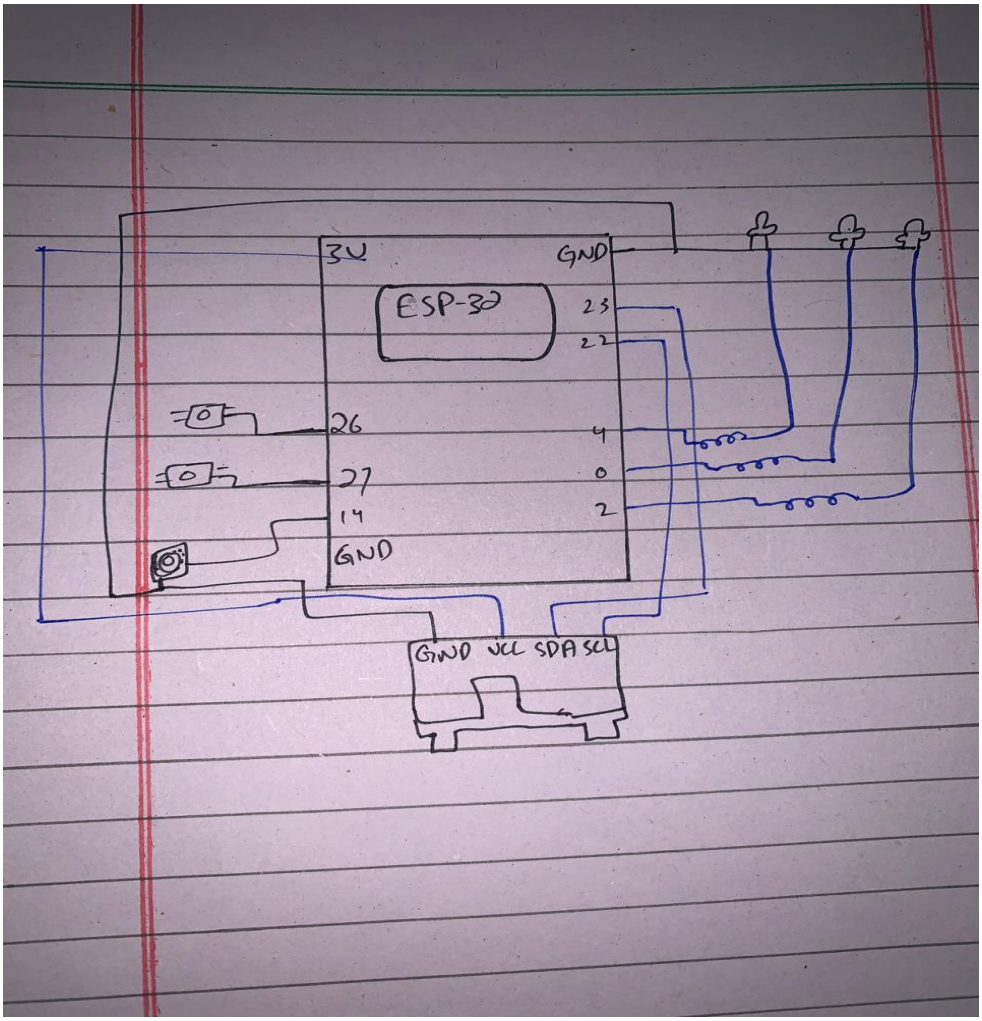



## HARDWARE OUTPUT:




## HAND DRAWN CIRCUIT




**PIN DIAGRAM:**

| Pin No | Name | Function | Use Case |
|---|---|---|---|
| GND | Ground | Common Ground | For all LEDs, Buzzer, Buttons, OLED |
| 15 | GPIO 26 | Pin for mode Button | Output for Button (Modebtn) |
| 16 | GPIO 27 | Pin for reset Button | Output for Button (Resetbtn) |
| 17 | GPIO 14 | Pin for Buzzer | Output for Buzzer |
| 3v3 | Power | 3.3V Output Power | OLED VCC |
| 36 | GPIO 22 | I2C SCL | OLED SCL |

| Pin No | Name | Function | Use Case |
|---|---|---|---|
| 39 | GPIO 23 | I2C SDA | OLED SDA |
| 24 | GPIO 4 | Pin for Yellow LED | Output for Yellow LED |
| 23 | GPIO 0 | Pin for Green LED | Output for Green LED |
| 22 | GPIO 2 | Pin for Red LED | Output for Red LED |