

National Textile University, Faisalabad



Name:	Muhammad Zohaib Warrach
Class:	BS-CS(B)
Registration No:	23-NTU-CS-1080
Assignment 1:	Task B
Course Name:	IoT & Embedded Systems
Submitted To:	Dr. Nasir Mehmood

Document of Assignment 1

Task B

Code Screenshot:



```
1 // Name: Muhammad Zohaib Warraich
2 // Roll no: 23-NTU-CS-1080
3
4 #include <Wire.h>
5 #include <Adafruit_GFX.h>
6 #include <Adafruit_SSD1306.h>
7
8 #define SCREEN_WIDTH 128
9 #define SCREEN_HEIGHT 64
10 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
11
12
13 #define yellowLED 19 // Setting up GPIO pin for Yellow Led
14 #define blueButton 25 // Setting up GPIO pin for blue Button
15 #define buzzer 27 // Setting up GPIO pin for Buzzer
16
17 bool ledState = false; // Current state of the LED
18 unsigned long pressTime = 0; // Time when the button was pressed
19 bool buttonPressed = false; // Flag to track button press state
20 bool longPress = false; // Flag to track if long press was detected
21
22 const unsigned long longPressTime = 2000; // 2 seconds for long press
23
24 // Simple but visually better display message
25 void displayMessage(const char* line1, const char* line2 = "") {
26     display.clearDisplay();
27     // First line will be bold and large
28     display.setTextSize(2); // Large text size
29     display.setTextColor(SSD1306_WHITE); // White text
30     display.setCursor(15, 10); // Centered cursor
31     display.println(line1); // Print first line
32     // Second line will be smaller and placed below First line
33     if (line2[0] != '\0') { // If second line is not empty
34         display.setTextSize(1); // Smaller text size
35         display.setCursor(20, 40); // Centered cursor for second line
36         display.println(line2); // Print second line
37     }
38     display.display();
39 }
40
41 void setup() {
42     Serial.begin(115200);
43     pinMode(yellowLED, OUTPUT); // setting up blue LED pin for getting output
44     pinMode(buzzer, OUTPUT); // setting up buzzer pin for getting output
45     pinMode(blueButton, INPUT_PULLUP); // Initialize blue button pin for taking input
46     if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
47         Serial.println("OLED not found!"); // OLED not found
48         while (true); // Halt execution if OLED is not found
49     }
50     displayMessage("Ready", "Press the Button");
51 }
52
53 void loop() {
54     bool buttonState = digitalRead(blueButton); // Read button state
55
56     // Detect whether the button is pressed or released
57     if (buttonState == LOW && !buttonPressed) {
58         buttonPressed = true; // Button is pressed
59         pressTime = millis(); // Record the time when button was pressed
60     }
61     if (buttonState == HIGH && buttonPressed) {
62         buttonPressed = false; // Button is released
63     }
64     if (millis() - pressTime > longPressTime) {
65         longPress = true;
66     } else {
67         longPress = false;
68     }
69     if (longPress) {
70         if (buttonState == HIGH) {
71             displayMessage("Long Press", "Button Released");
72         } else {
73             displayMessage("Long Press", "Button Pressed");
74         }
75     }
76 }
77
78 void buttonHandler() {
79     if (buttonState == LOW && !buttonPressed) {
80         buttonPressed = true;
81         pressTime = millis();
82     }
83     if (buttonState == HIGH && buttonPressed) {
84         buttonPressed = false;
85     }
86     if (millis() - pressTime > longPressTime) {
87         longPress = true;
88     } else {
89         longPress = false;
90     }
91     if (longPress) {
92         if (buttonState == HIGH) {
93             displayMessage("Long Press", "Button Released");
94         } else {
95             displayMessage("Long Press", "Button Pressed");
96         }
97     }
98 }
```

```

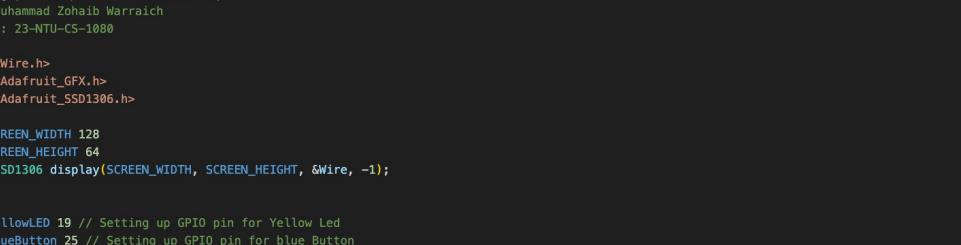
60     longPress = false; // Reset long press flag
61 }
62
63 // Check for long press
64 if (buttonState == LOW && buttonPressed) {
65     if ((millis() - pressTime > longPressTime)) { // If pressed for more than 2 seconds
66         displayMessage("Long Press", "Buzzer ON");
67         tone(buzzer, 1000); // It will play tone of 1 kHz for 1 second
68         delay(1000);
69         noTone(buzzer); // Stop the tone
70         displayMessage("Completed", "Buzzer OFF");
71         delay(500); // Brief delay to show message
72         longPress = true; // Set long press flag
73     }
74 }
75
76 // Check for short press
77 if (buttonState == HIGH && buttonPressed) {
78     if (!longPress) { // If it was not a long press
79         ledState = !ledState; // Toggle LED state
80         digitalWrite(yellowLED, ledState); // Update LED state
81         if (ledState==HIGH) { // If LED is on
82             displayMessage("LED ON", "Short Press");
83         } else {
84             displayMessage("LED OFF", "Short Press");
85         }
86     }
87
88     buttonPressed = false; // Reset button pressed flag
89     delay(300); // Debounce delay
90 }
91 }
92

```

Short explanation about code:

This program is designed for an ESP32 (or Arduino) that uses a yellow LED, a buzzer, and an OLED display. When powered on, the display shows a “Ready” message, indicating that the system is waiting for user input. The blue button controls the actions, if it’s pressed briefly, the LED toggles between ON and OFF, and the display updates to show the LED’s status. However, if the button is held down for more than two seconds, it’s recognized as a long press, causing the buzzer to sound for one second while displaying a “Buzzer ON” message. Afterward, the screen confirms that the buzzer has turned off. Overall, the code provides both visual and sound feedback based on the type of button press.

Build Output:



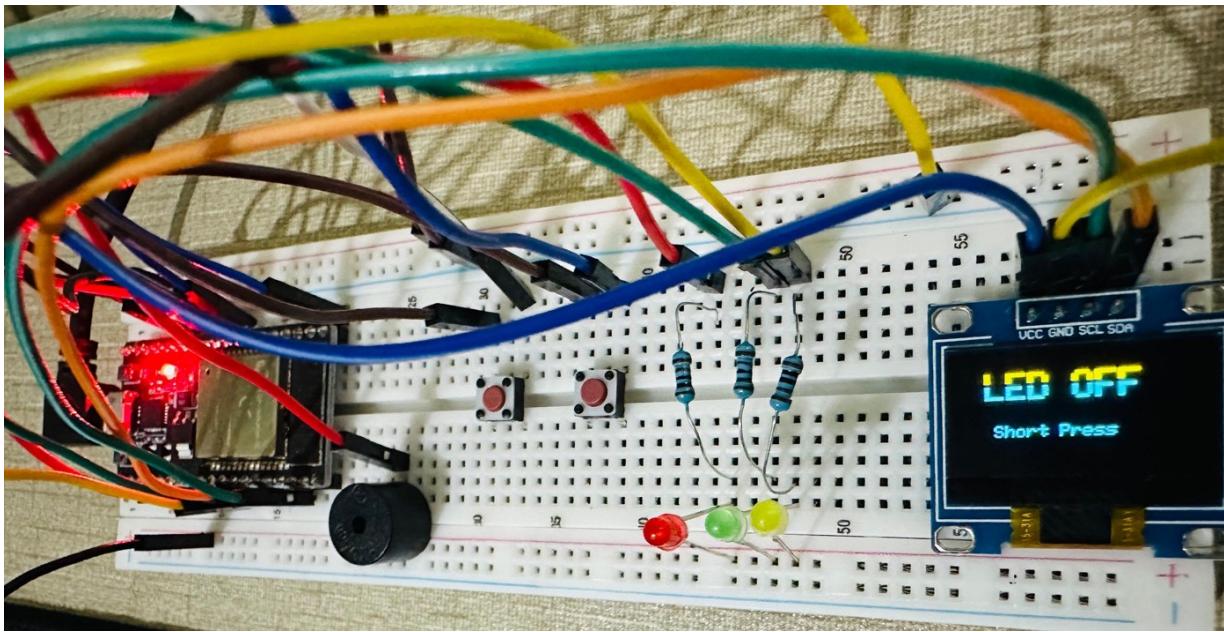
```
src > C++ main.cpp > buzzer
You, 1 minute ago | 2 authors (zohaibwarrach1 and one other)
1 // Name: Muhammad Zohaib Warrach
2 // Roll no: 23-NTU-CS-1080
3
4 #include <Wire.h>
5 #include <Adafruit_GFX.h>
6 #include <Adafruit_SSD1306.h>
7
8 #define SCREEN_WIDTH 128
9 #define SCREEN_HEIGHT 64
10 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
11
12
13 #define yellowLED 19 // Setting up GPIO pin for Yellow Led
14 #define blueButton 25 // Setting up GPIO pin for blue Button
15 #define buzzer 27 // Setting up GPIO pin for Buzzer
16
17 bool ledState = false; // Current state of the LED
18 unsigned long pressTime = 0; // Time when the button was pressed
19 bool buttonPressed = false; // Flag to track button press state
20 bool longPress = false; // Flag to track if long press was detected

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
Build - Task ✓ + × [ ] x
Dependency Graph
|--- Adafruit GFX Library @ 1.12.3
|--- Adafruit SSD1306 @ 2.5.15
|--- Wire @ 2.0.0
Building in release mode
Retrieving maximum program size .pio/build/esp32dev/firmware.elf
Checking size .pio/build/esp32dev/firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [=          ] 6.7% (used 22088 bytes from 327680 bytes)
Flash: [==        ] 23.3% (used 305873 bytes from 1310720 bytes)
===== [SUCCESS] Took 2.55 seconds =====
* Terminal will be reused by tasks, press any key to close it.
```

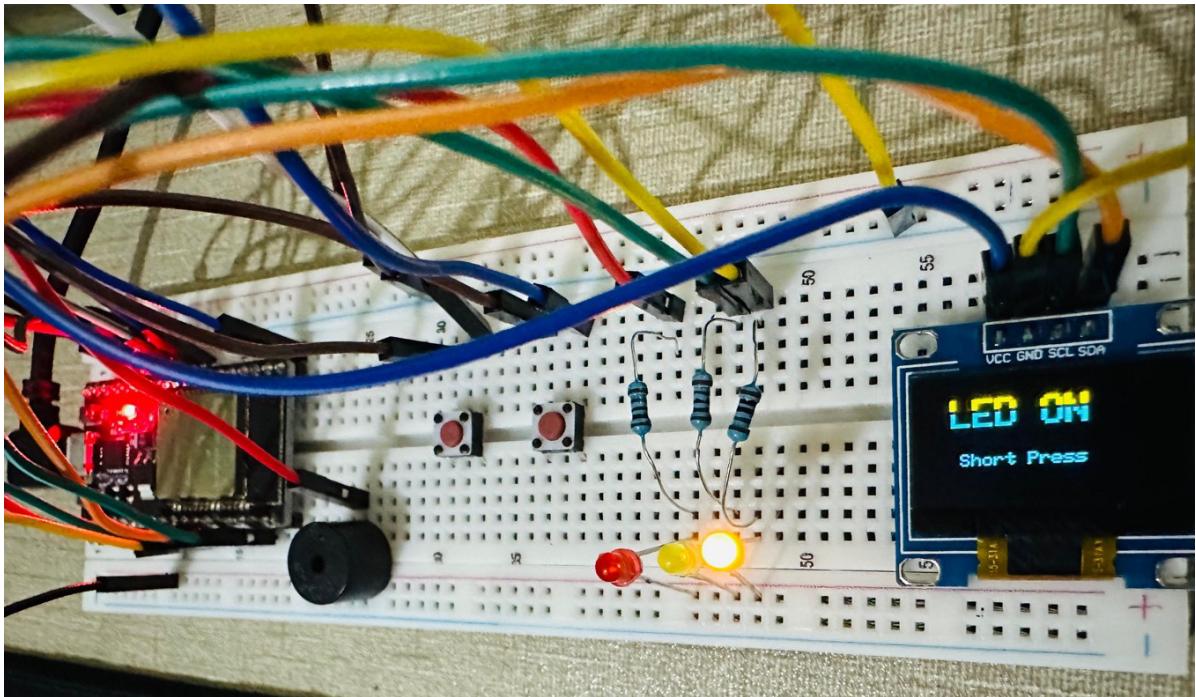
Upload Output:

Hardware Output:

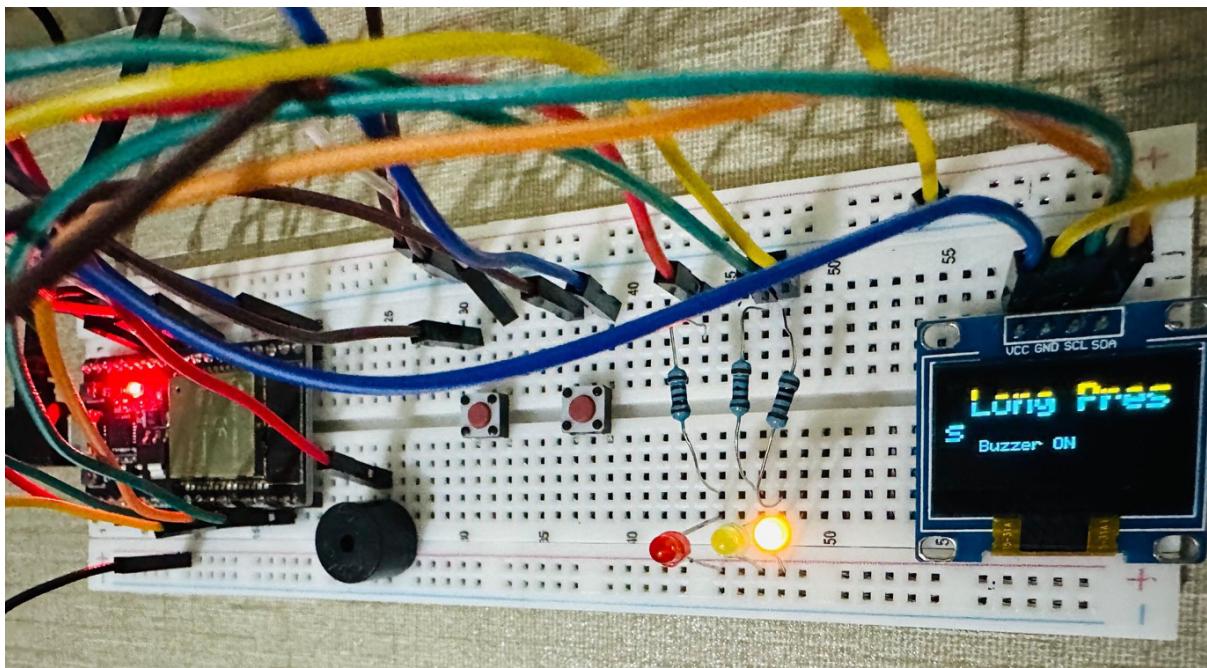
Short Press LED OFF:



Short Press LED ON:



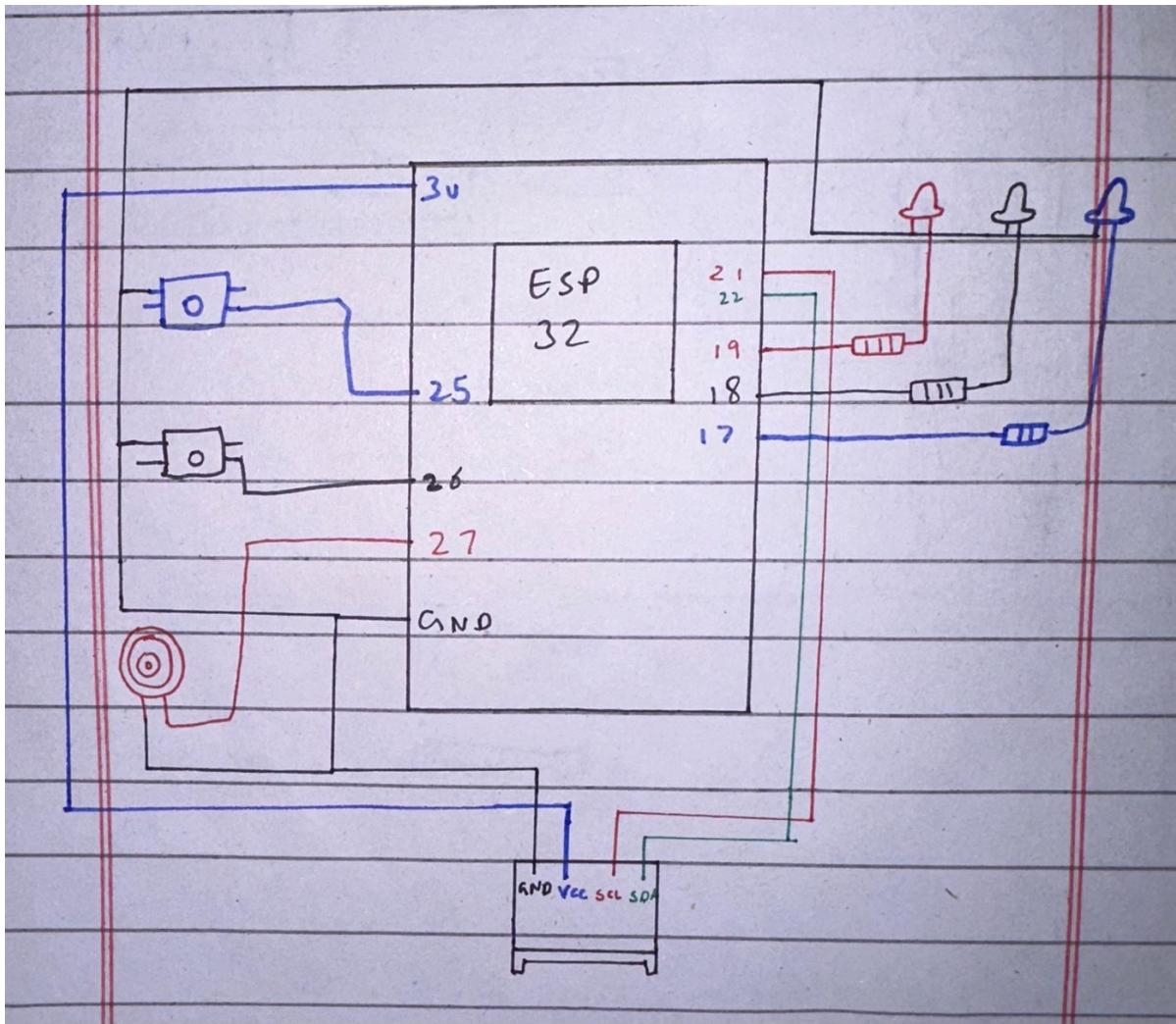
Long Press:



Wokwi Project Link:

<https://wokwi.com/projects/445165207310341121>

Circuit Diagram:



Pin Map Diagram:

Pin No	Name	Function	Use Case
GND .2	Ground	Common Ground	For all LEDs, Buzzer, Buttons, OLED
25	GPIO 25	Pin for Blue Button	Output for Blue Button (Modebtn)
26	GPIO 26	Pin for White Button	Output for White Button (Resetbtn)
27	GPIO 27	Pin for Buzzer	Output for Buzzer
3v3	Power	3.3V output power	OLED VCC
22	GPIO 22	I2C SCL	OLED SCL
21	GPIO 21	I2C SDA	OLED SDA
19	GPIO 19	Pin for Yellow LED	Output for Yellow LED
18	GPIO 18	Pin for Green LED	Output for Green LED
17	GPIO 17	Pin for Red LED	Output for Red LED