

National Textile University, Faisalabad



Department of Computer Science

Name:	Samar Khalid
Class:	BSCS-5B
Registration No:	23-NTU-CS-1091
Assignment:	1 (Task-b)
Course Name:	Embedded IoT systems
Submitted To:	Sir Nasir Mahmood
Submission Date:	23-10-2025

Assignment 1

Task B

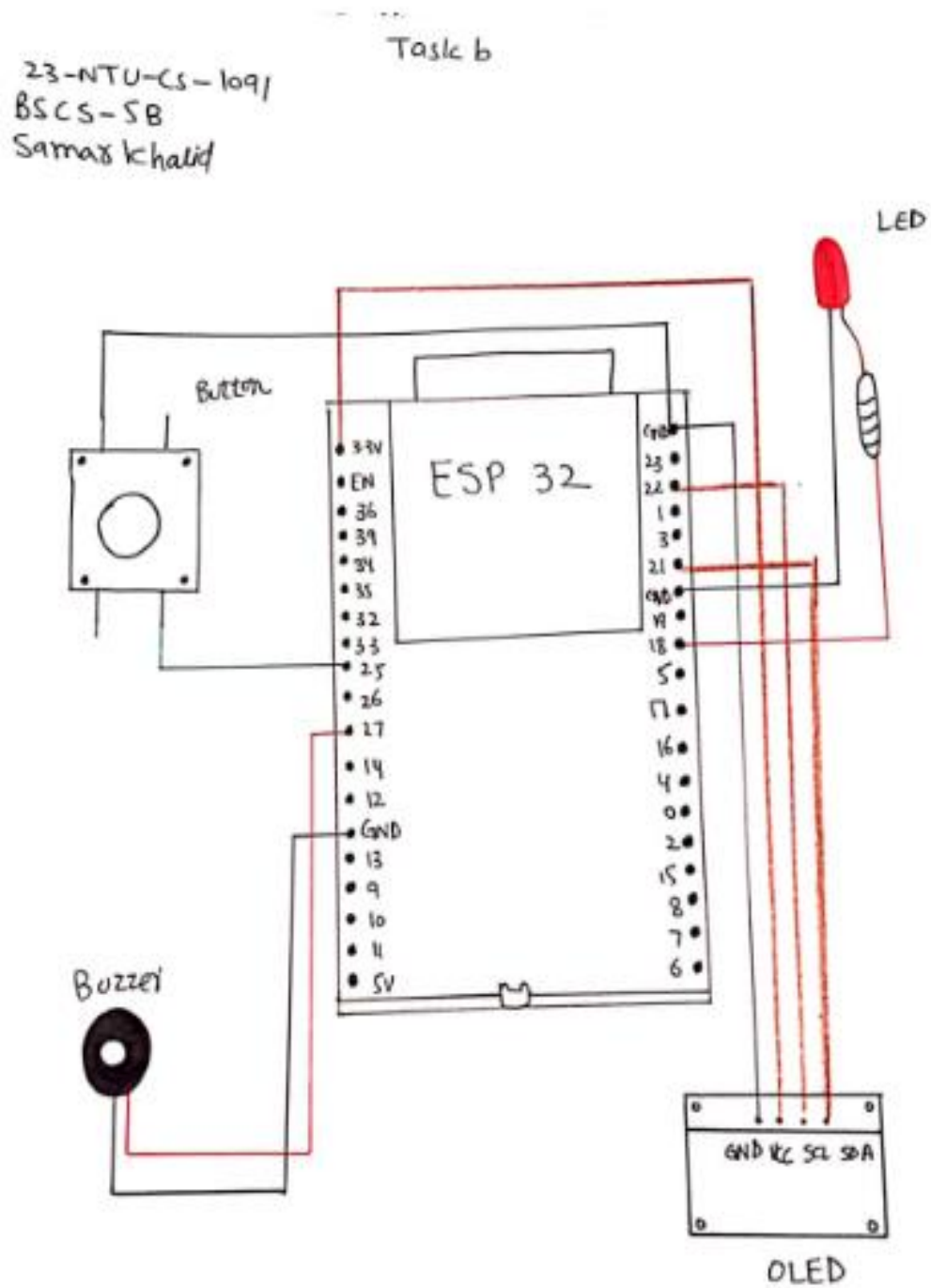
Task Explanation:

This code detects short and long button presses using an ESP32 equipped with a button, LED, buzzer, and OLED display. The LED turns ON or OFF and the screen displays "Short Press" when you press the button rapidly. The buzzer sounds and the message "Long Press" appears on the screen if you hold the button for longer than 1.5 seconds. Every action receives feedback from the OLED, and the buzzer or LED physically displays the outcome. When everything is functioning, it first sets up all the pins and displays "System Ready." In summary, the length of time you press the button effects how the program responds.

Pin Diagram:

Device Names	Device Pins	Esp-32 Pins
OLED	Vcc3	3.3 V
OLED	GND	GND
OLED	SCL	GPIO22
OLED	SDA	GPIO21
LED	Cathode (short leg)	GND
LED	Anode (Long leg)	GPIO18 (through resistor)
Buzzer	Cathode (Black)	GND
Buzzer	Anode (Red)	GPIO27
Button	Leg one	GND
Button	Other leg	GPIO25

Circuit Diagram:



Code screenshot:

```
task-b > src > main.cpp > loop()
1 //Assignmnet1 Task-b
2 //Long Press for buzzer and short press for led (ON)
3 //Embedded IOT system Fall-2025
4 //Samar Khalid //23-NTU-CS-1091
5 #include <Arduino.h>
6
7 #include <Wire.h>
8 #include <Adafruit_GFX.h>
9 #include <Adafruit_SSD1306.h>
10
11 //Pin Definitions
12 #define BUTTON_PIN 25 // Push button input
13 #define LED_PIN 18 // LED output
14 #define BUZZER_PIN 27 // Buzzer output
15 #define SDA_PIN 21 // I2C SDA
16 #define SCL_PIN 22 // I2C SCL
17
18 //OLED Setup
19 #define SCREEN_WIDTH 128
20 #define SCREEN_HEIGHT 64
21 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
22
23 //Global variables
24 bool ledState = false;
25 bool buttonPressed = false;
26 unsigned long pressStartTime = 0;
```

```
27 const unsigned long longPressDuration = 1500; // 1.5 seconds
28
29 //Display message
30 void showMessage(const String &msg) {
31     display.clearDisplay();
32     display.setTextSize(1);
33     display.setTextColor(SSD1306_WHITE);
34     display.setCursor(10, 20);
35     display.print(msg);
36     display.display();
37 }
38
39 //Setup
40 void setup() {
41     Serial.begin(115200);
42     pinMode(BUTTON_PIN, INPUT_PULLUP);
43     pinMode(LED_PIN, OUTPUT);
44     pinMode(BUZZER_PIN, OUTPUT);
45
46     //Initialize OLED
47     Wire.begin(SDA_PIN, SCL_PIN);
48     if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
49         Serial.println("SSD1306 allocation failed");
50         while (true);
51     }
```

```

51     }
52     display.clearDisplay();
53     display.display();
54
55     showMessage("System Ready");
56     digitalWrite(LED_PIN, LOW);
57     digitalWrite(BUZZER_PIN, LOW);
58 }
59
60 //Loop
61 void loop() {
62     bool btnState = digitalRead(BUTTON_PIN);
63
64     //Button pressed (active LOW)
65     if (!btnState && !buttonPressed) {
66         buttonPressed = true;
67         pressStartTime = millis();
68     }
69
70     //Button held down
71     if (buttonPressed && !btnState) {
72         unsigned long pressDuration = millis() - pressStartTime;
73         if (pressDuration > longPressDuration) {

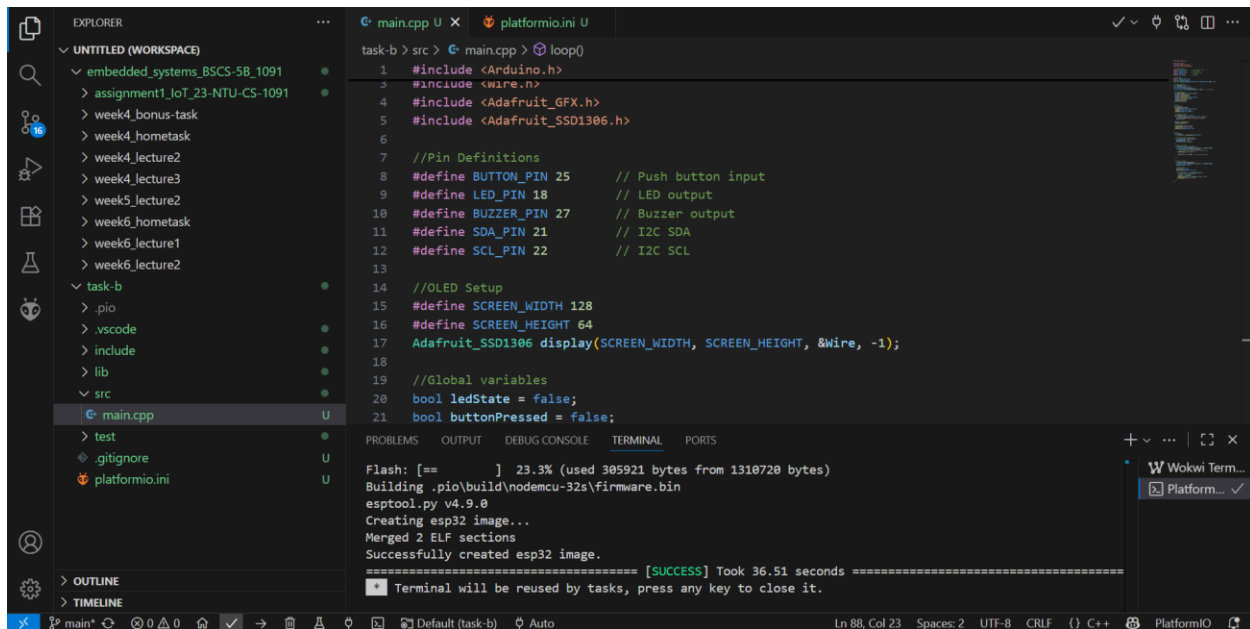
```

```

73             if (pressDuration > longPressDuration) {
74                 // Long press detected: play buzzer continuously
75                 showMessage("Long Press");
76
77                 tone(BUZZER_PIN, 2000); //2kHz tone(continuous)
78             }
79         }
80     }
81
82     //Button released
83     if (buttonPressed && btnState) {
84         unsigned long pressDuration = millis() - pressStartTime;
85         buttonPressed = false;
86
87         //Stop buzzer
88         noTone(BUZZER_PIN);
89         digitalWrite(BUZZER_PIN, LOW);
90
91         if (pressDuration <= longPressDuration) {
92             // Short press: toggle LED
93             ledState = !ledState;
94             digitalWrite(LED_PIN, ledState ? HIGH : LOW);
95             showMessage("Short Press");
96         }
97     }
98 }

```

VS code Build success:



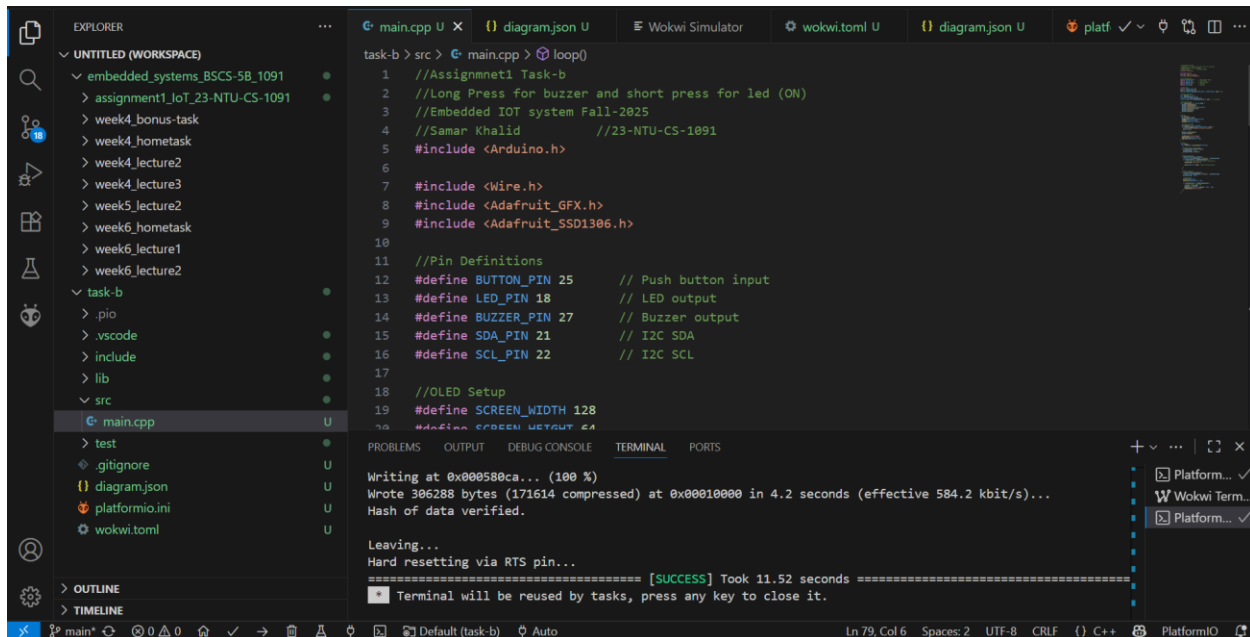
The screenshot shows the Visual Studio Code interface with a workspace named 'embedded_systems_BSCS-5B_1091'. The Explorer panel on the left shows a project structure with folders like 'src' and 'include'. The main editor displays 'main.cpp' with the following code:

```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <Adafruit_GFX.h>
4 #include <Adafruit_SSD1306.h>
5
6
7 //Pin Definitions
8 #define BUTTON_PIN 25 // Push button input
9 #define LED_PIN 18 // LED output
10 #define BUZZER_PIN 27 // Buzzer output
11 #define SDA_PIN 21 // I2C SDA
12 #define SCL_PIN 22 // I2C SCL
13
14 //OLED Setup
15 #define SCREEN_WIDTH 128
16 #define SCREEN_HEIGHT 64
17 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
18
19 //Global variables
20 bool ledState = false;
21 bool buttonPressed = false;
```

The TERMINAL panel at the bottom shows the build output:

```
Flash: [== ] 23.3% (used 305921 bytes from 1310720 bytes)
Building .pio/build/nodemcu-32s/firmware.bin
esptool.py v4.9.0
Creating esp32 image...
Merged 2 ELF sections
Successfully created esp32 image.
===== [SUCCESS] Took 36.51 seconds =====
Terminal will be reused by tasks, press any key to close it.
```

Upload on ESP-32 success:



The screenshot shows the Visual Studio Code interface with a workspace named 'embedded_systems_BSCS-5B_1091'. The Explorer panel on the left shows a project structure with folders like 'src' and 'include'. The main editor displays 'main.cpp' with the following code:

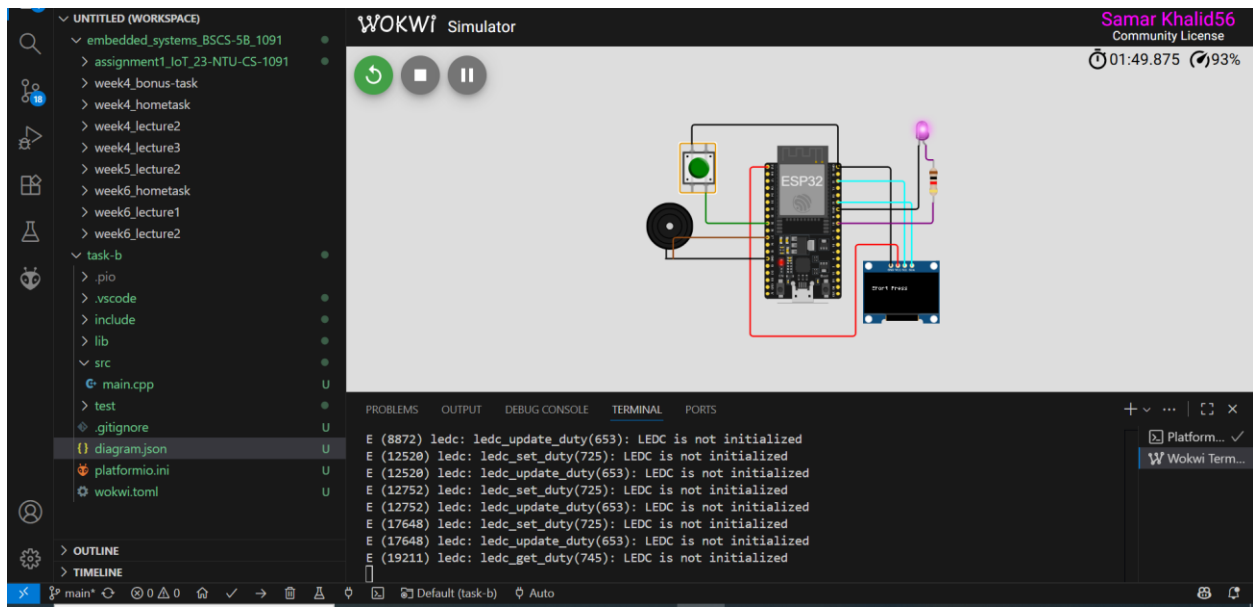
```
1 //Assignmnet1 Task-b
2 //Long Press for buzzer and short press for led (ON)
3 //Embedded IOT system Fall-2025
4 //Samar Khalid //23-NTU-CS-1091
5 #include <Arduino.h>
6
7 #include <Wire.h>
8 #include <Adafruit_GFX.h>
9 #include <Adafruit_SSD1306.h>
10
11 //Pin Definitions
12 #define BUTTON_PIN 25 // Push button input
13 #define LED_PIN 18 // LED output
14 #define BUZZER_PIN 27 // Buzzer output
15 #define SDA_PIN 21 // I2C SDA
16 #define SCL_PIN 22 // I2C SCL
17
18 //OLED Setup
19 #define SCREEN_WIDTH 128
20 #define SCREEN_HEIGHT 64
```

The TERMINAL panel at the bottom shows the upload output:

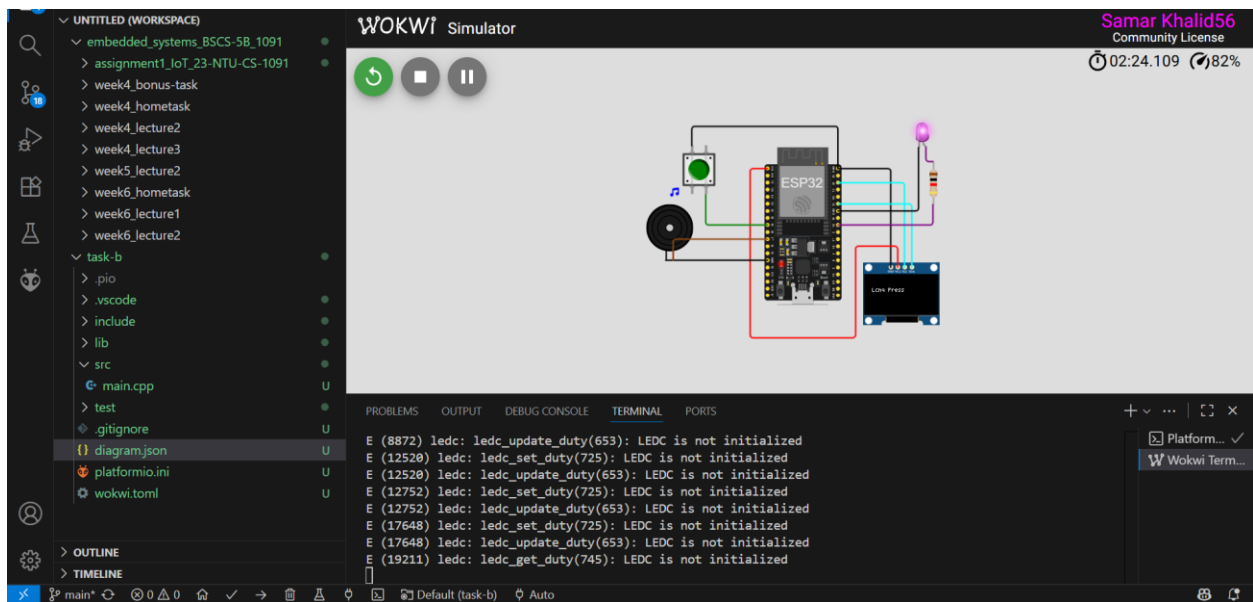
```
Writing at 0x000580ca... (100 %)
Wrote 306288 bytes (171614 compressed) at 0x00010000 in 4.2 seconds (effective 584.2 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
===== [SUCCESS] Took 11.52 seconds =====
Terminal will be reused by tasks, press any key to close it.
```

Output on wokwi:

Short Press:



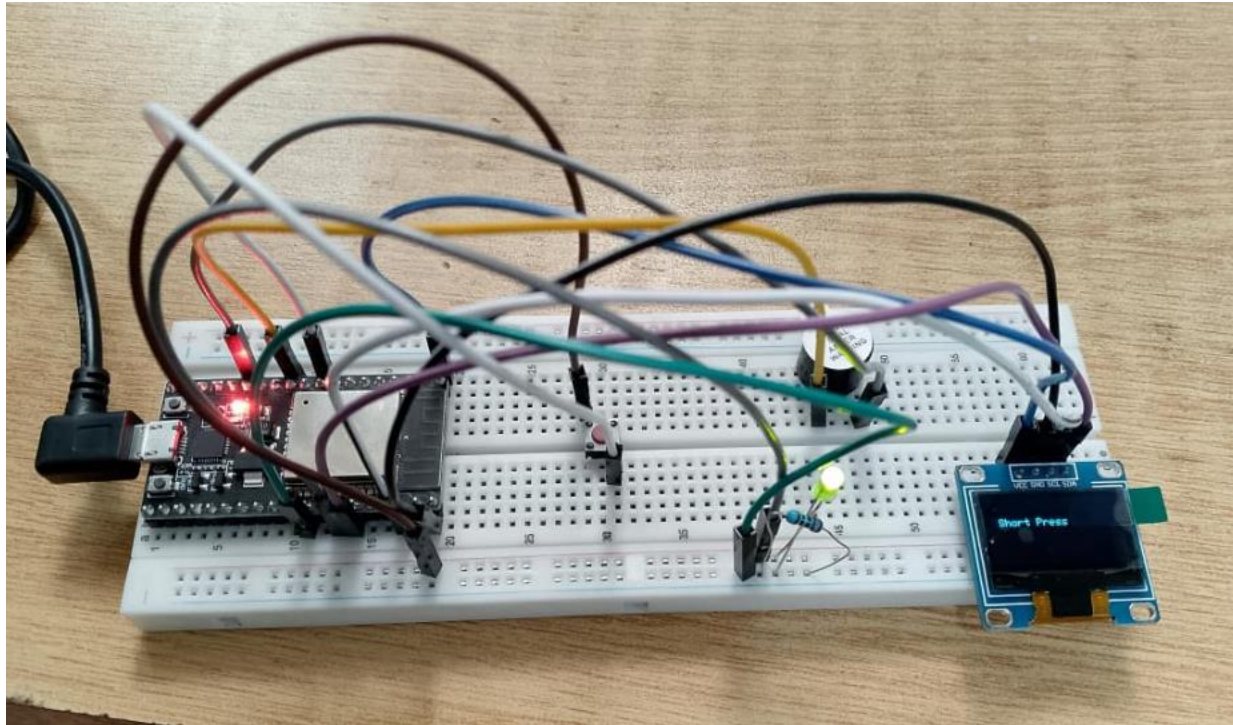
Long Press:



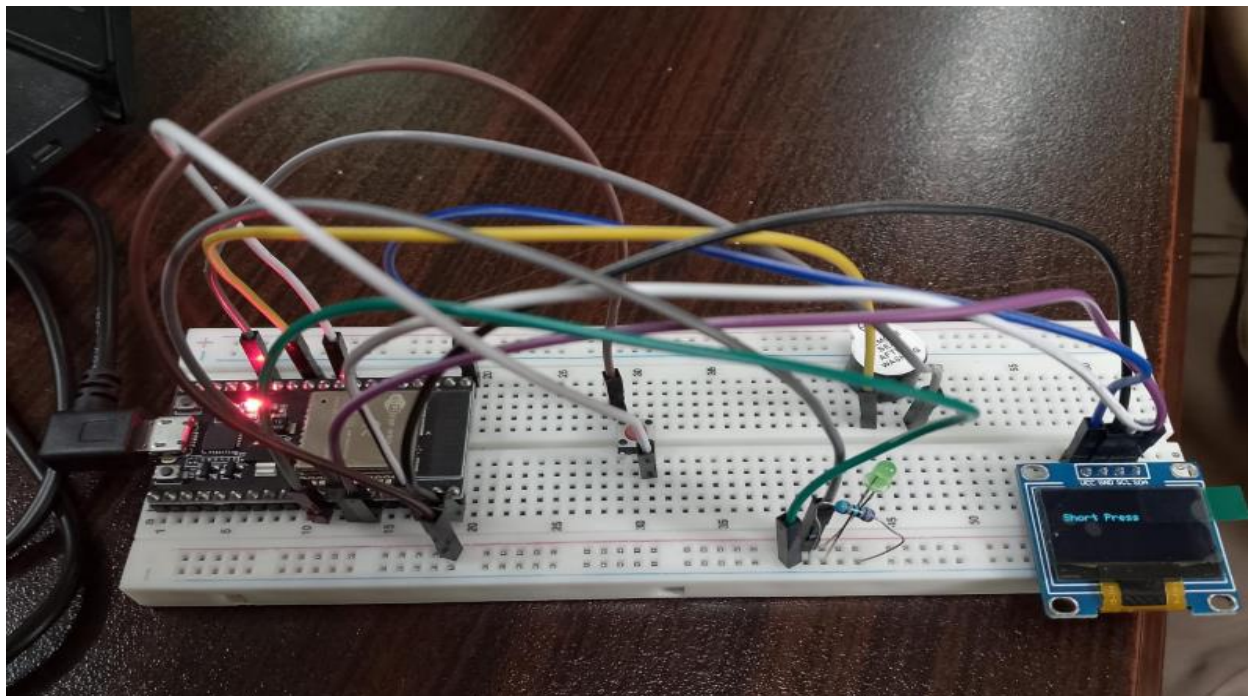
Output on Kit:

Short press:

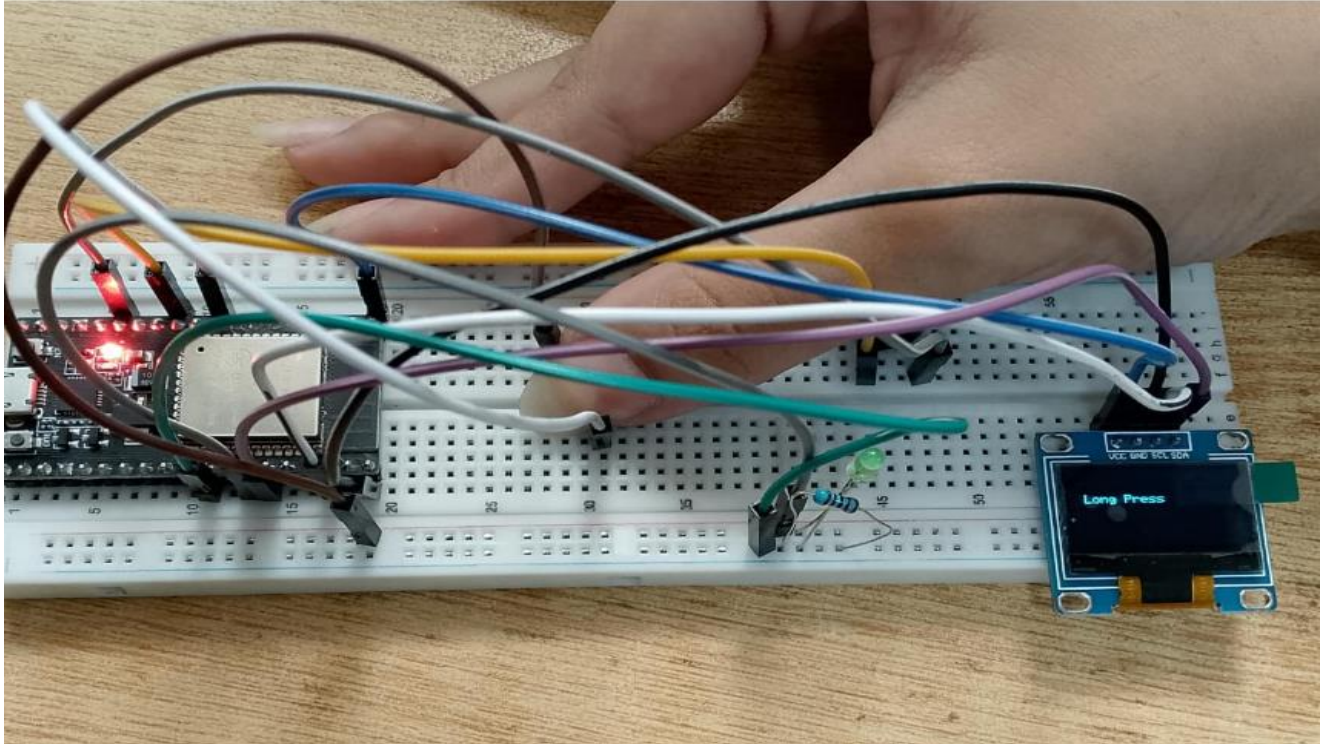
LED ON:



LED OFF:



Long Press:



Wokwi link:

<https://wokwi.com/projects/445578718745765889>

Handwritten Code:

Assignment 1

(Task b)

```
#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

// Pin Definitions

```
#define BUTTON_PIN 25
#define LED_PIN 18
#define BUZZER_PIN 27
#define SDA_PIN 21
#define SCL_PIN 22
```

// OLED Setup

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
```

// Global variables

```
bool ledstate = false;
bool buttonPressed = false;
unsigned long pressStartTime = 0;
const unsigned long longPressDuration = 1500; // 1.5 seconds
```

```
// Display message  
void show Message (const String &msg) {
```

```
    display.clearDisplay();  
    display.setTextSize(1);  
    display.setTextColour(SSD1306_WHITE);  
    display.setCursor(10, 20);  
    display.print(msg);  
    display.display();  
}
```

```
// setup  
void setup() {
```

```
    Serial.begin(115200);  
    pinMode(BUTTON_PIN, INPUT_PULLUP);  
    pinMode(LED_PIN, OUTPUT);  
    pinMode(BUZZER_PIN, OUTPUT);
```

```
// Initialize OLED
```

```
Wire.begin(SDA_PIN, SCL_PIN);  
if (!display.begin(SSD1306_SWITCHCAPVCC,  
                    0x32)) {  
    Serial.println("SSD1306 allocation failed");  
    while(true);  
}  
display.clearDisplay();  
display.display();
```



```
showMessage ("system Ready");  
digitalWrite (LED_PIN, LOW);  
digitalWrite (BUZZER_PIN, LOW);  
}
```

```
// Loop  
void loop () {  
    bool btnState = digitalRead (BUTTON_PIN);
```

```
    // Button pressed (active low)  
    if (!btnState && !buttonPressed) {  
        buttonPressed = true;  
        pressStartTime = millis();  
    }
```

```
    // Button held down
```

```
    if (buttonPressed && btnState) {  
        unsigned long pressDuration = millis() - pressStartTime;  
        buttonPressed = false;
```

```
    // stop buzzer  
    noTone (BUZZER_PIN);  
    digitalWrite (BUZZER_PIN, LOW);
```

```
    if (pressDuration <= longPressDuration) {  
        // short press -> toggle LED
```

```
ledstate = ! ledstate;  
digitalWrite ( LED-PIN, ledState ? HIGH : LOW;  
  show Message ("short Press");  
}  
}  
}
```