National Textile University, Faisalabad

Department of Computer Science

Name:	Minahil Fatima
Class:	BSCS-5B
Registration No:	23-NTU-CS-1051
Assignment:	Assignment 1 (Task-b)
Course Name:	Embedded IoT Systems
Submitted To:	Sir Nasir Mahmood
Submission Date:	23-October, 2025

Assignment 1

Task-b

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

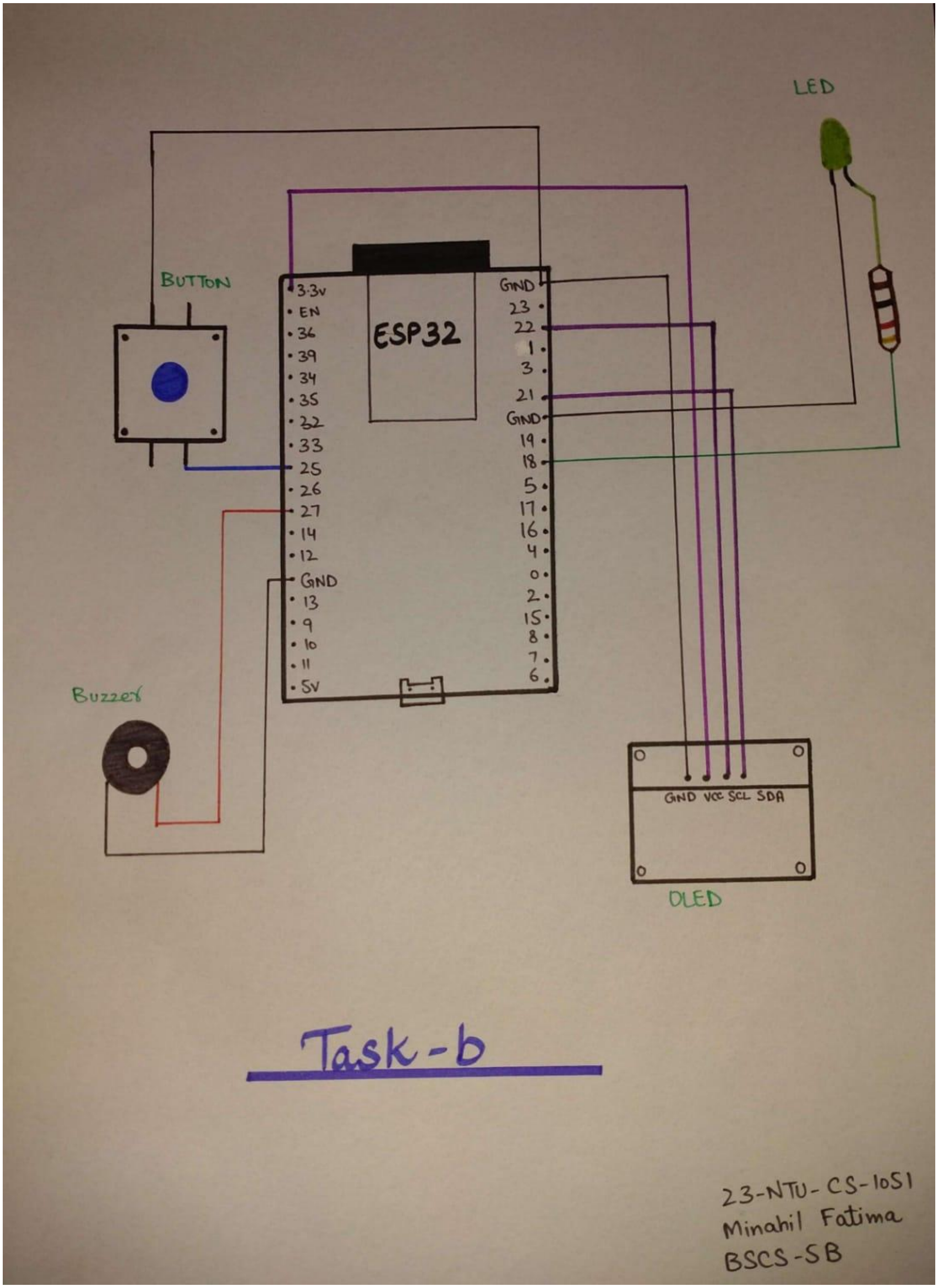
- **Short press: toggle LED**
- **Long press (> 1.5 s): play a buzzer tone**

In this task, I used an ESP32, a button, LED, buzzer, and OLED display. The button is used to detect short and long presses. A short press toggles the LED and shows “Short Press” on the screen, while holding the button for more than 1.5 seconds makes the buzzer sound and shows “Long Press”. The OLED shows messages using I2C.

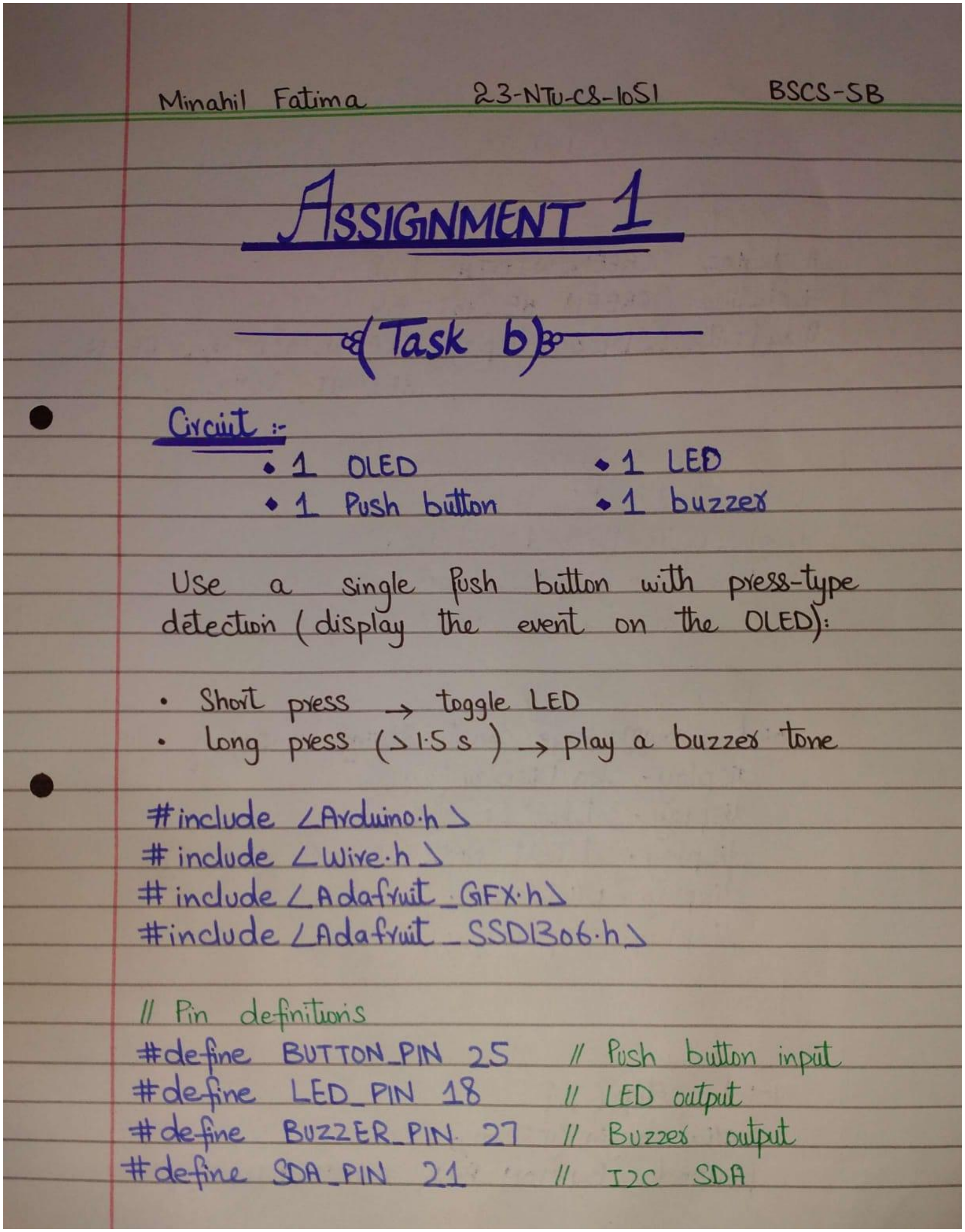
Pin map:

Device name	Pin name	Pin number
OLED	VCC	3.3v
OLED	GND	GND
OLED	SCL	GPIO22
OLED	SDA	GPIO21
LED	Cathode	GND
LED	Anode	GPIO18 (through resistor)
Button	One side	GND
Button	Other side	GPIO25
Buzzer	Cathode	GND
Buzzer	Anode	GPIO27

Circuit diagram:



Handwritten code pictures:



Minahil Fatima   23-NTU-CS-1051   BSCS-5B

ASSIGNMENT 1

(Task b)

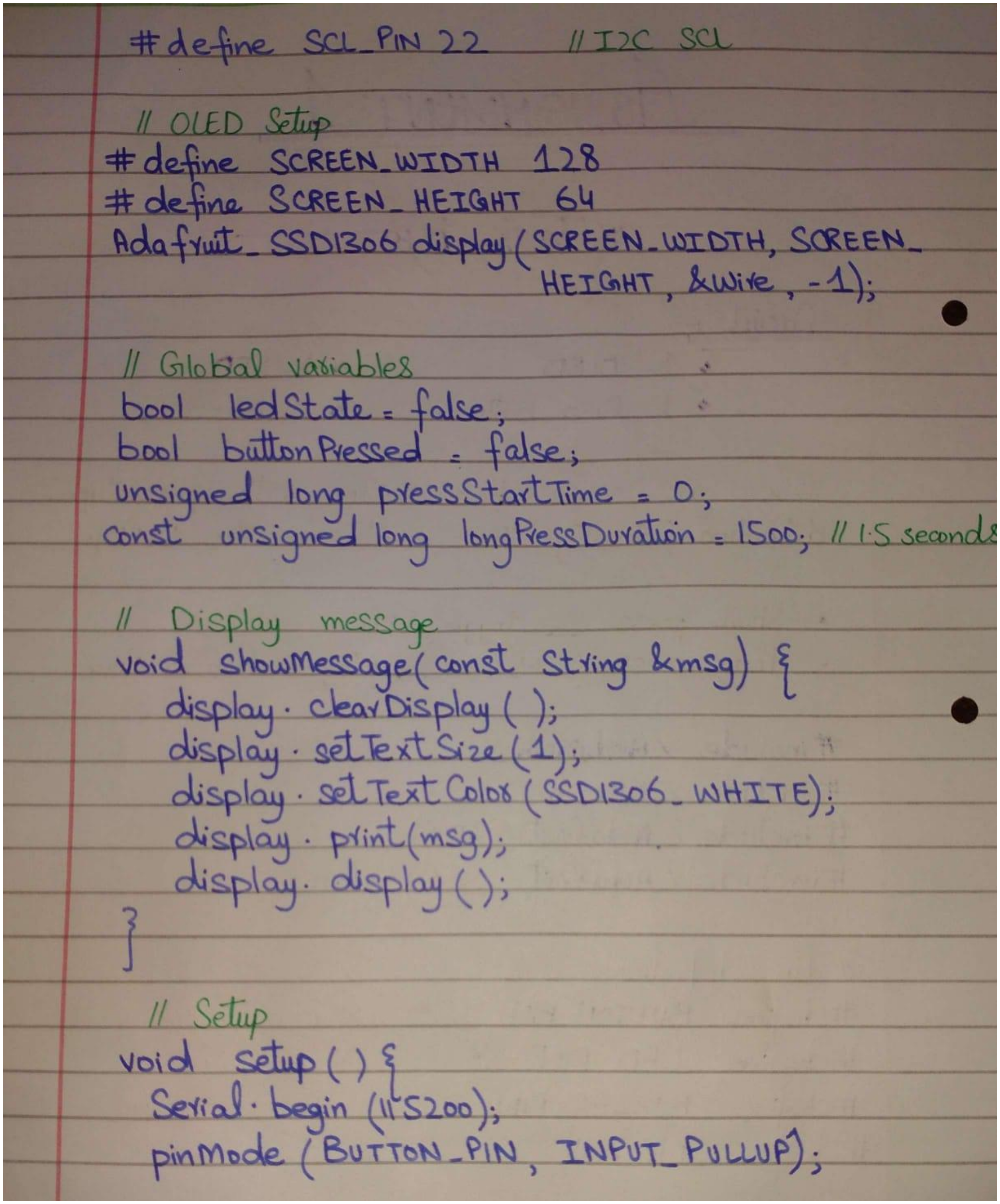
Circuit :-

- 1 OLED
- 1 Push button
- 1 LED
- 1 buzzer

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

- Short press → toggle LED
- Long press (>1.5 s) → play a buzzer tone

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

// Pin definitions
#define BUTTON_PIN 25     // Push button input
#define LED_PIN 18        // LED output
#define BUZZER_PIN 27     // Buzzer output
#define SDA_PIN 21        // I2C SDA
```

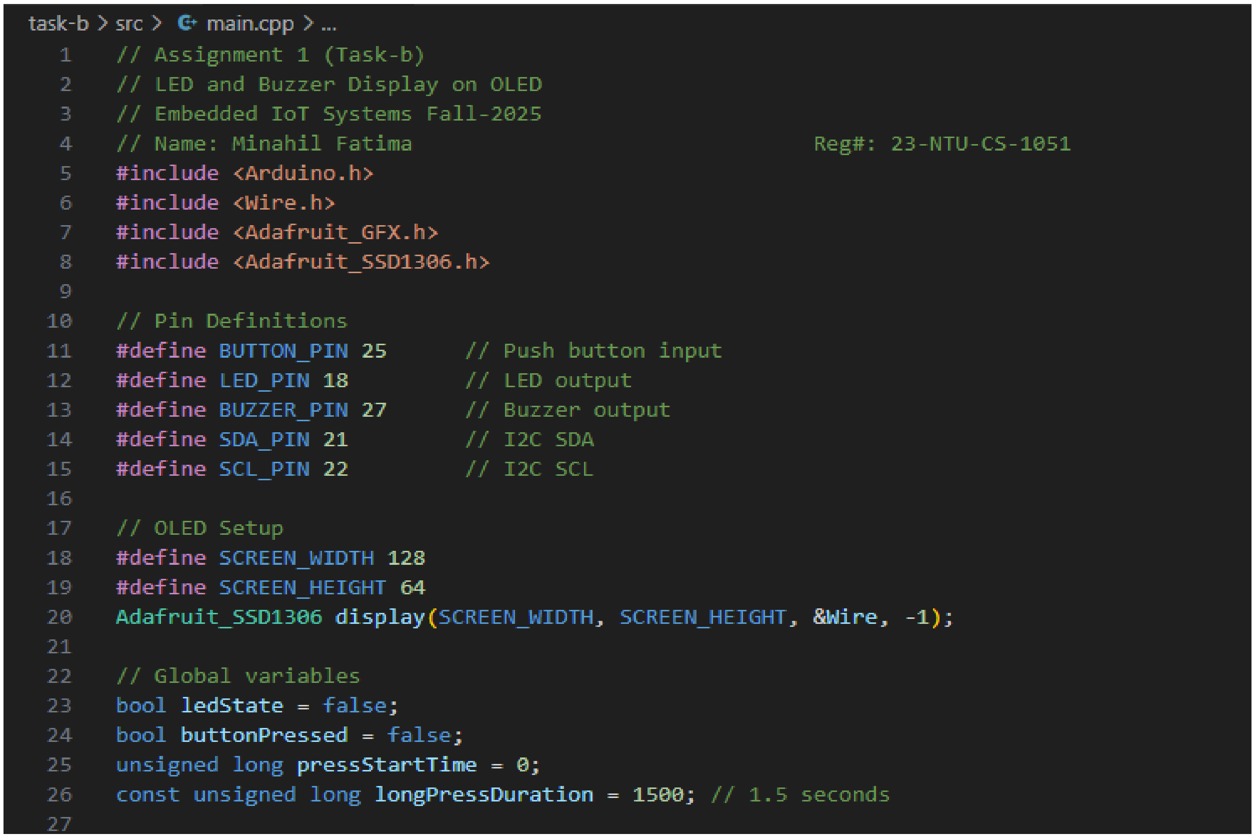


```
#define SCL_PIN 22     // I2C SCL

// 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 showMessage(const String &msg) {
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.print(msg);
  display.display();
}

// Setup
void setup() {
 Serial.begin(115200);
 pinMode(BUTTON_PIN, INPUT_PULLUP);
```

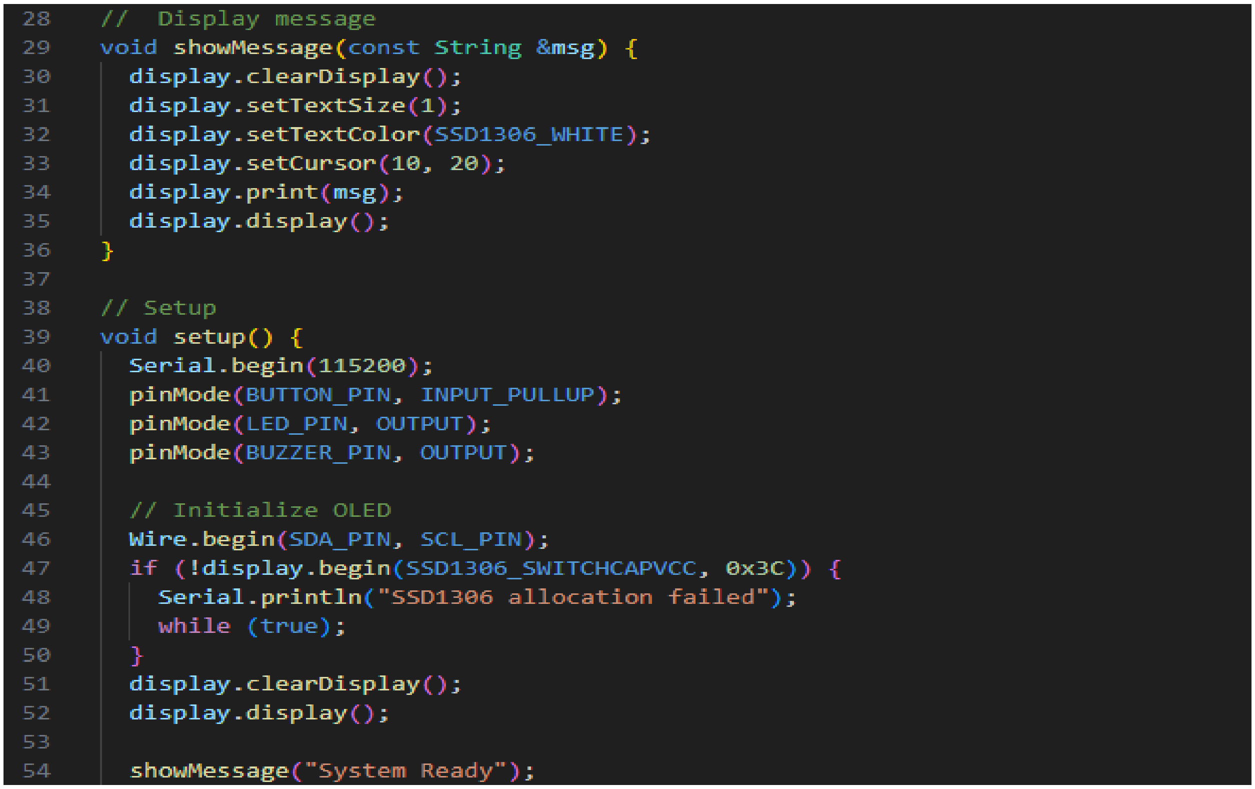
```
pinMode (LED_PIN, OUTPUT);
pinMode ( BUZZER_PIN, OUTPUT);

// Initialize OLED
Wire.begin (SDA_PIN, SCL_PIN);
if (!display.begin(SSD1306_SWITCHAPVCC, 0X3C)) {
  Serial.println("SSD1306 allocation failed");
  while (true);
}
 display.clearDisplay ();
 display.display ();

 ShowMessage ("System Ready");
 digital Write (LED_PIN, LOW);
 digital Write( BUZZER_PIN, LOW);
}

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

// Button pressed (active low)
if (! btn State && ! buttonPressed) {
  buttonPressed = true;
  pressStart Time = millis ();
}
```

```
  // Button held down
 if (buttonPressed && ! btnState) {
  unsigned long pressDuration = millis () - pressStartTime;
  if (pressDuration > longPressDuration) {
   // long press detected -> play buzzer continuously
   showMessage ("Long Press");
   tone (BUZZER_PIN, 2000); // 2 kHz tone (continuous)
   }
 }

 // Button released
 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);
 showMessage ("Short Press");
  }
}
}
```

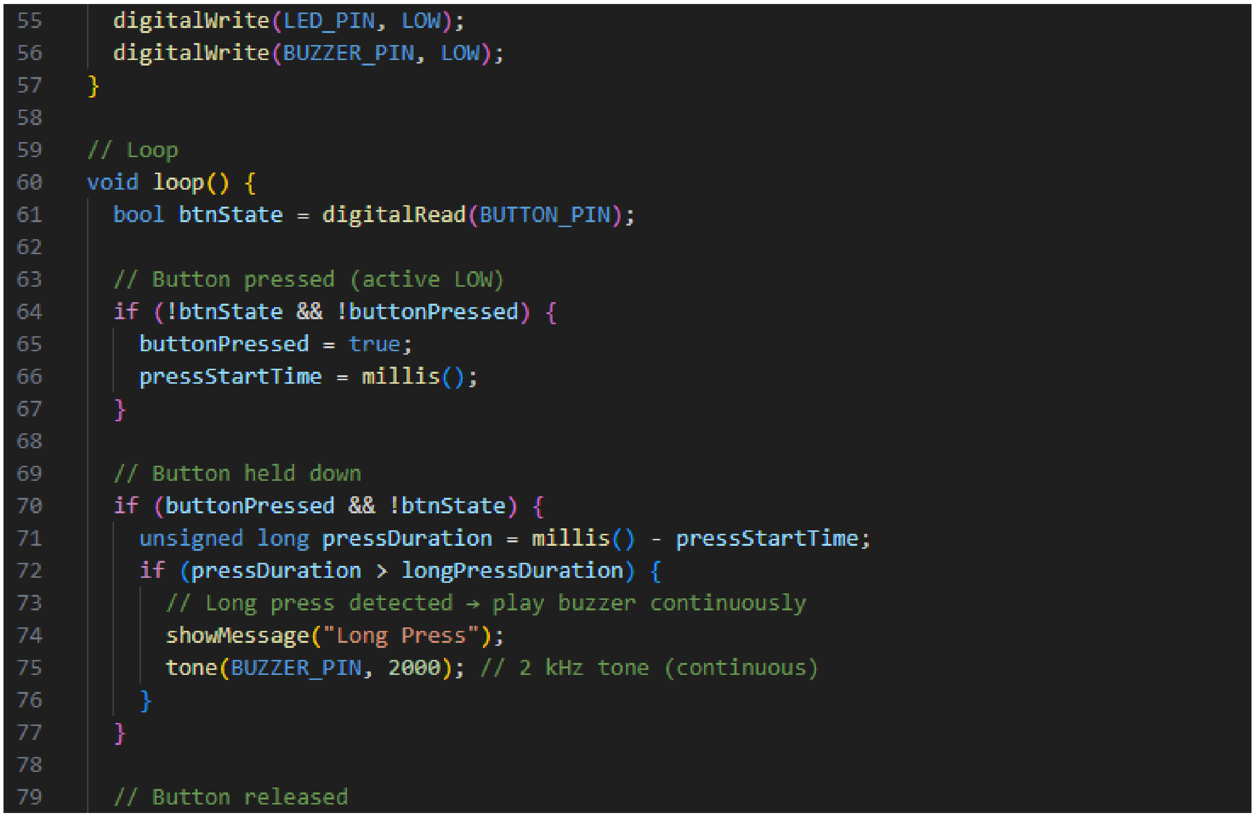
Code screenshots:



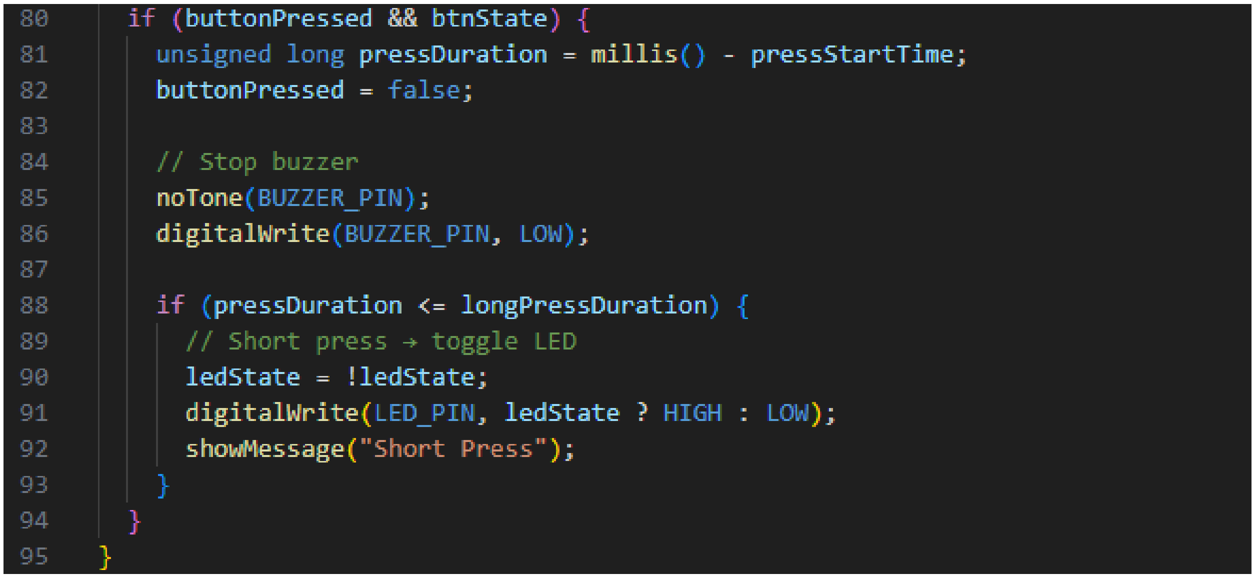
```cpp
// Assignment 1 (Task-b)
// LED and Buzzer Display on OLED
// Embedded IoT Systems Fall-2025
// Name: Minahil Fatima                               Reg#: 23-NTU-CS-1051
#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Pin Definitions
#define BUTTON_PIN 25      // Push button input
#define LED_PIN 18         // LED output
#define BUZZER_PIN 27      // Buzzer output
#define SDA_PIN 21         // I2C SDA
#define SCL_PIN 22         // I2C SCL

// 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

```



```cpp
//  Display message
void showMessage(const String &msg) {
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(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, 0x3C)) {
    Serial.println("SSD1306 allocation failed");
    while (true);
  }
  display.clearDisplay();
  display.display();

  showMessage("System Ready");
```

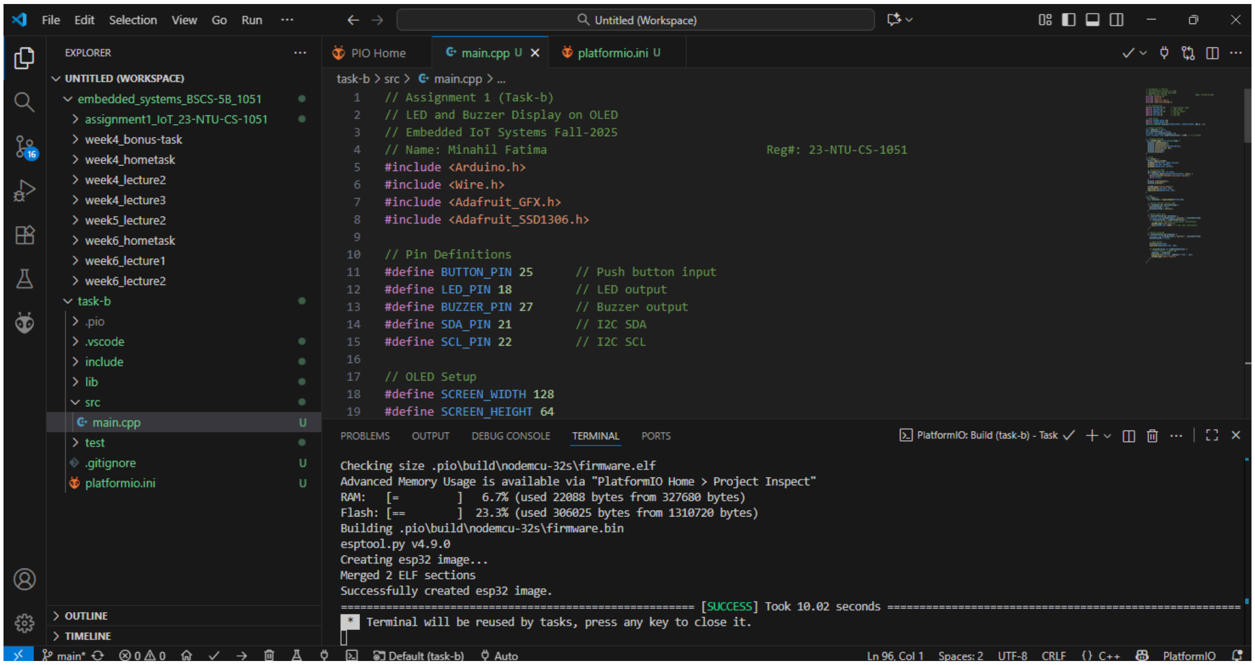


```
  digitalWrite(LED_PIN, LOW);
  digitalWrite(BUZZER_PIN, LOW);
}

// 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;
    if (pressDuration > longPressDuration) {
      // Long press detected → play buzzer continuously
      showMessage("Long Press");
      tone(BUZZER_PIN, 2000); // 2 kHz tone (continuous)
    }
  }

  // Button released
```



```
  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);
      showMessage("Short Press");
    }
  }
}
```

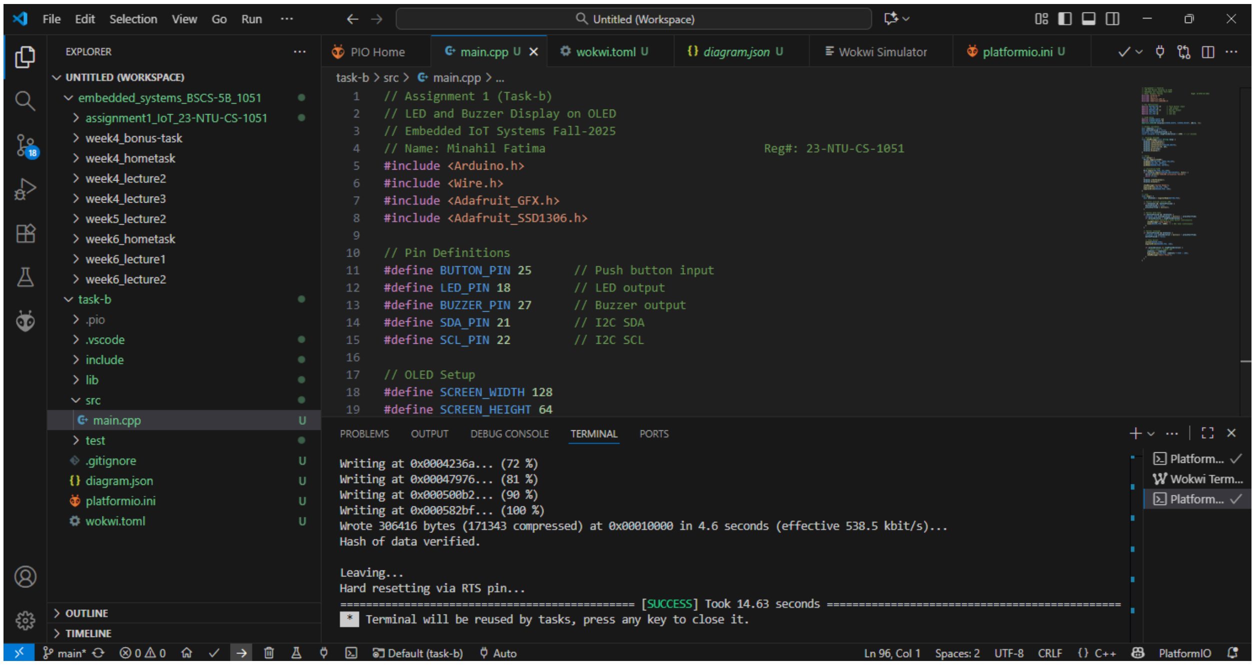
Code build success:



```cpp
// Assignment 1 (Task-b)
// LED and Buzzer Display on OLED
// Embedded IoT Systems Fall-2025
// Name: Minahil Fatima                              Reg#: 23-NTU-CS-1051
#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Pin Definitions
#define BUTTON_PIN 25      // Push button input
#define LED_PIN 18         // LED output
#define BUZZER_PIN 27      // Buzzer output
#define SDA_PIN 21         // I2C SDA
#define SCL_PIN 22         // I2C SCL

// OLED Setup
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

```
Checking size .pio\build\nodemcu-32s\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 306025 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 10.02 seconds ===================================================
 *  Terminal will be reused by tasks, press any key to close it.
```

Code upload success:

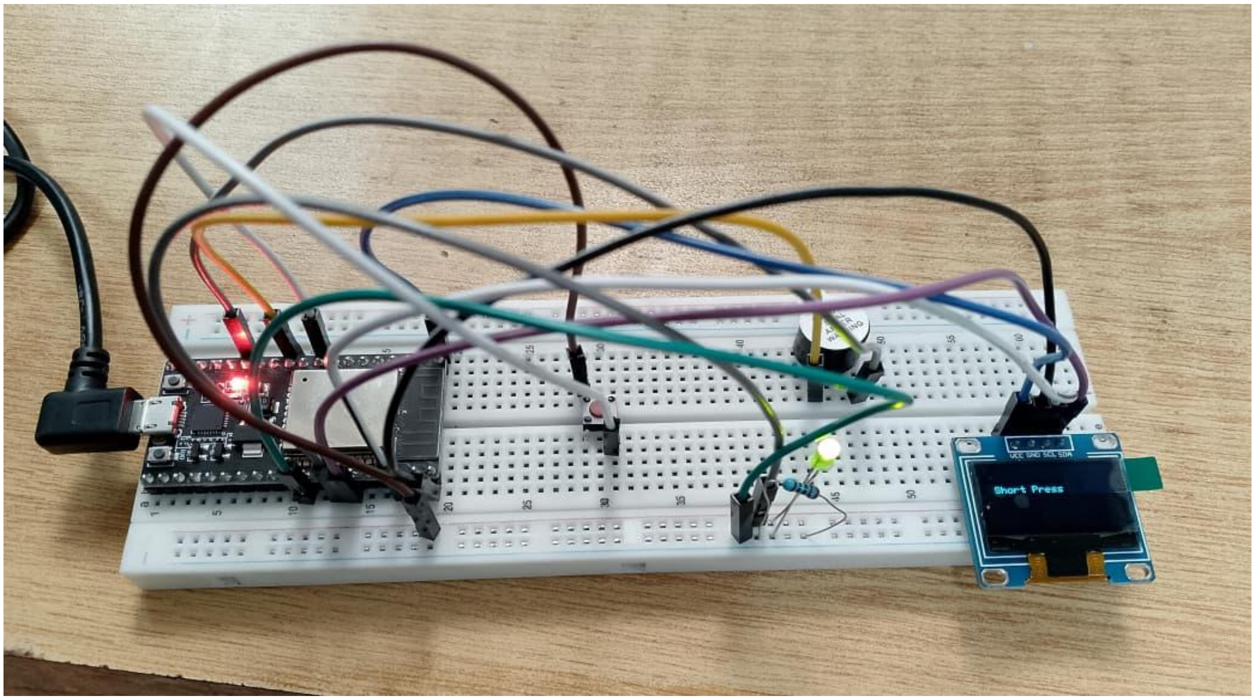


```cpp
// Assignment 1 (Task-b)
// LED and Buzzer Display on OLED
// Embedded IoT Systems Fall-2025
// Name: Minahil Fatima                              Reg#: 23-NTU-CS-1051
#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Pin Definitions
#define BUTTON_PIN 25      // Push button input
#define LED_PIN 18         // LED output
#define BUZZER_PIN 27      // Buzzer output
#define SDA_PIN 21         // I2C SDA
#define SCL_PIN 22         // I2C SCL

// OLED Setup
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

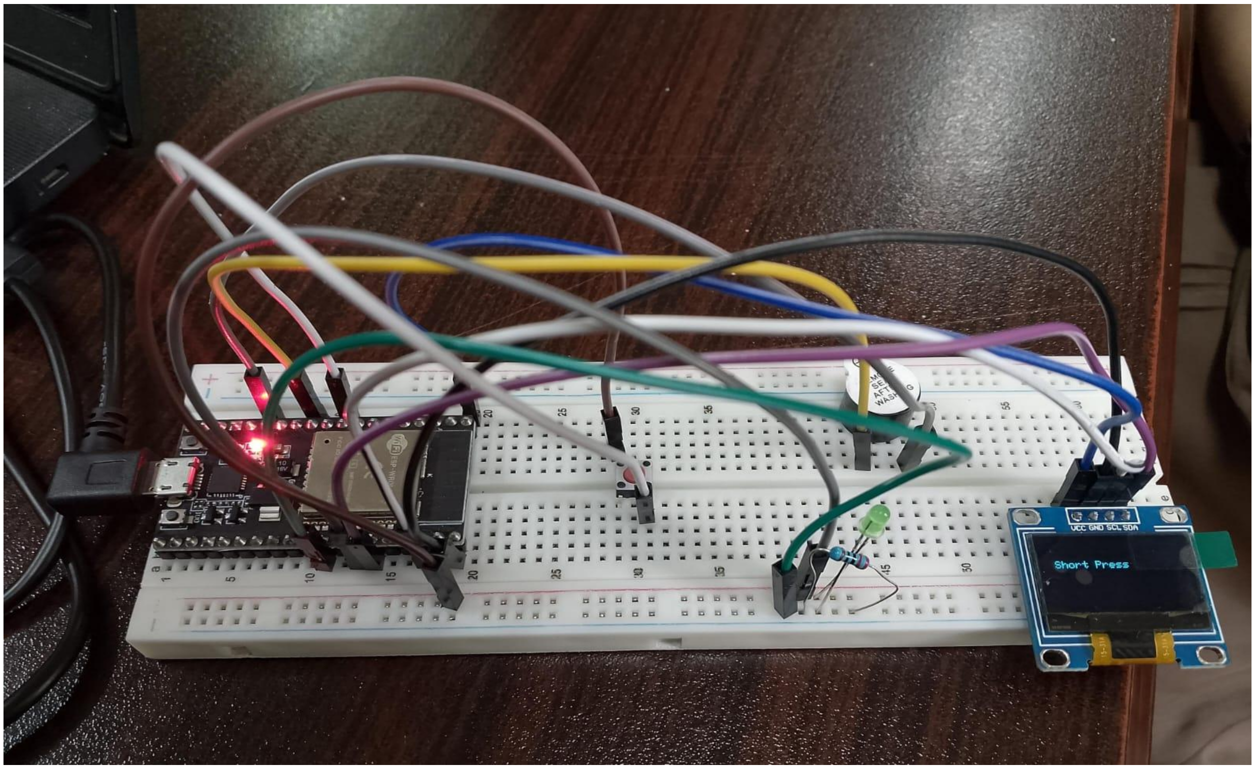
```
Writing at 0x0004236a... (72 %)
Writing at 0x00047976... (81 %)
Writing at 0x000500b2... (90 %)
Writing at 0x000582bf... (100 %)
Wrote 306416 bytes (171343 compressed) at 0x00010000 in 4.6 seconds (effective 538.5 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
============================================== [SUCCESS] Took 14.63 seconds ==============================================
 *  Terminal will be reused by tasks, press any key to close it.
```

Kit output pictures:

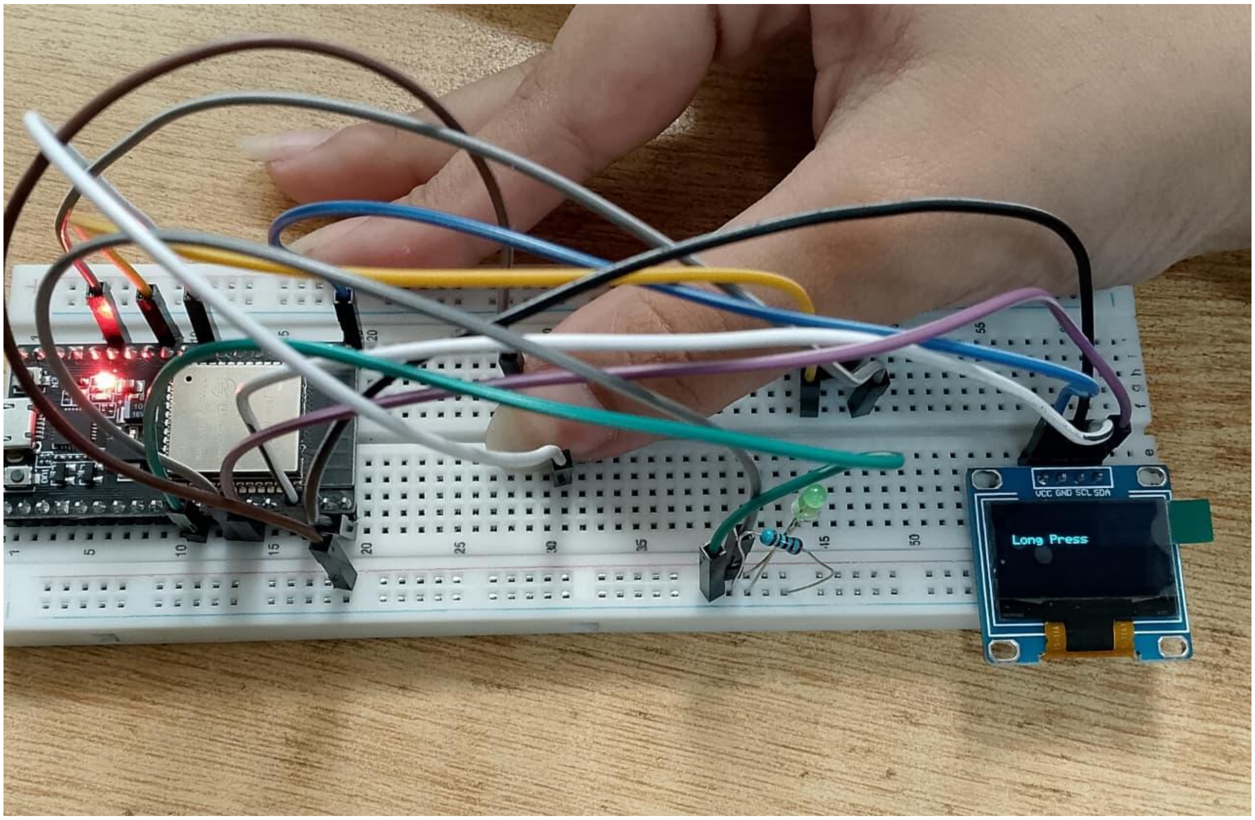
Short press LED ON:



Short press LED OFF:

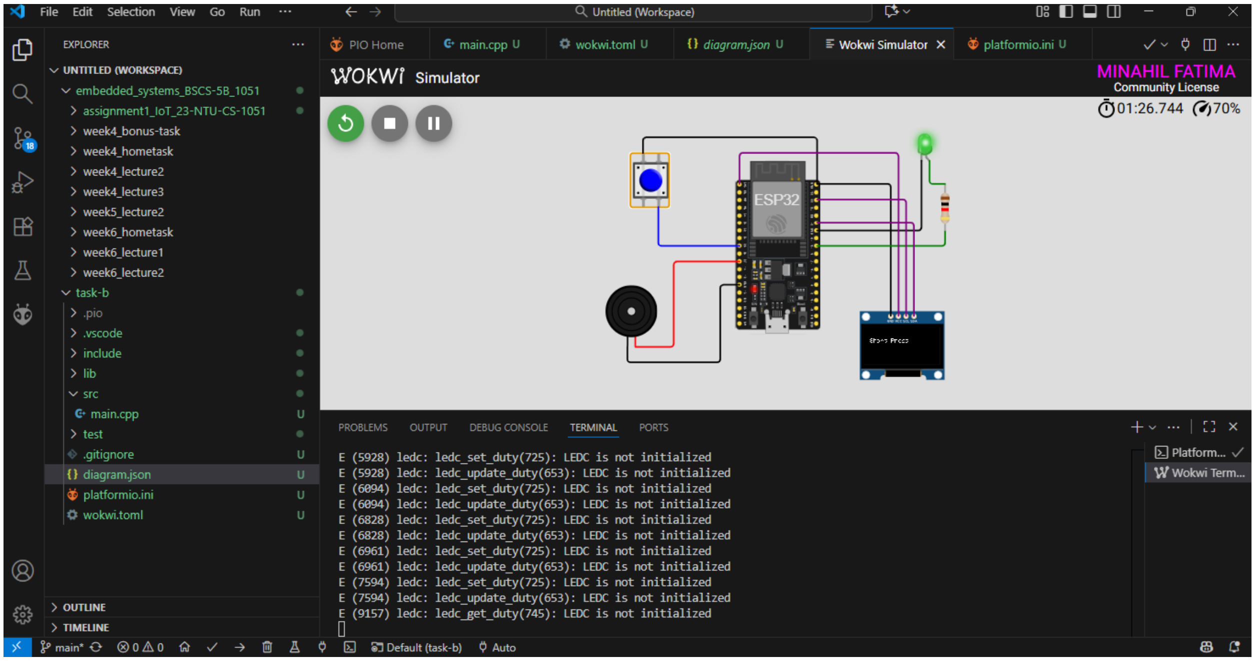


Long press buzzer:



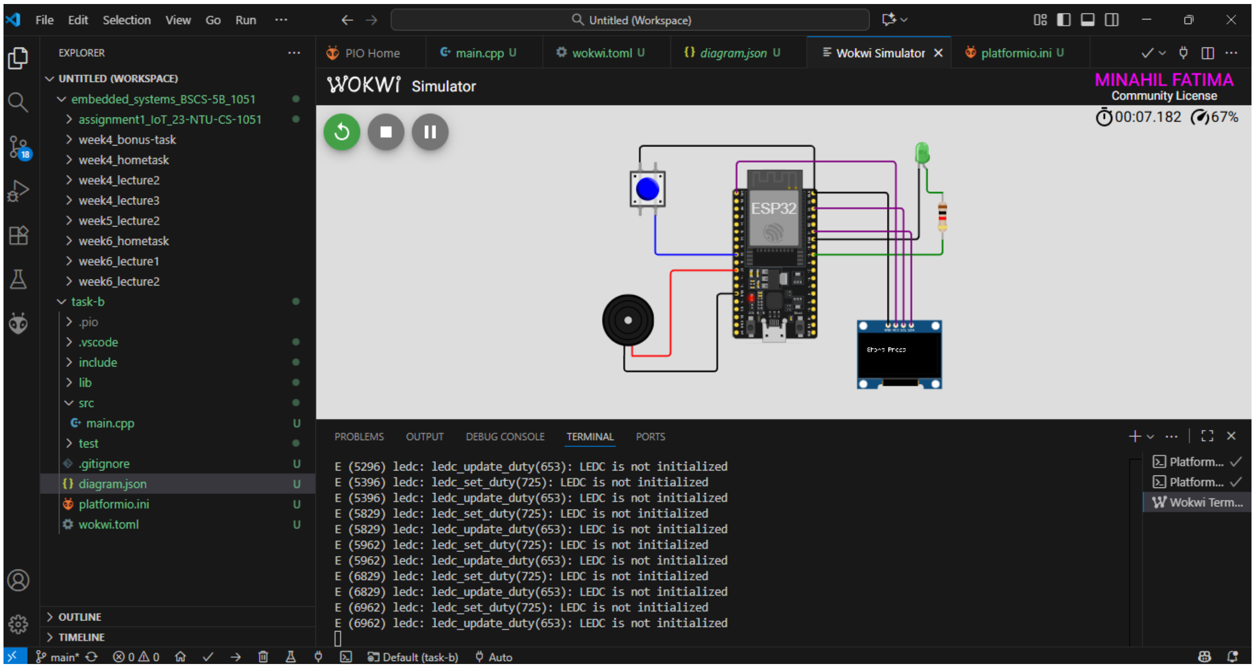
Wokwi output pictures:

Short press LED ON:



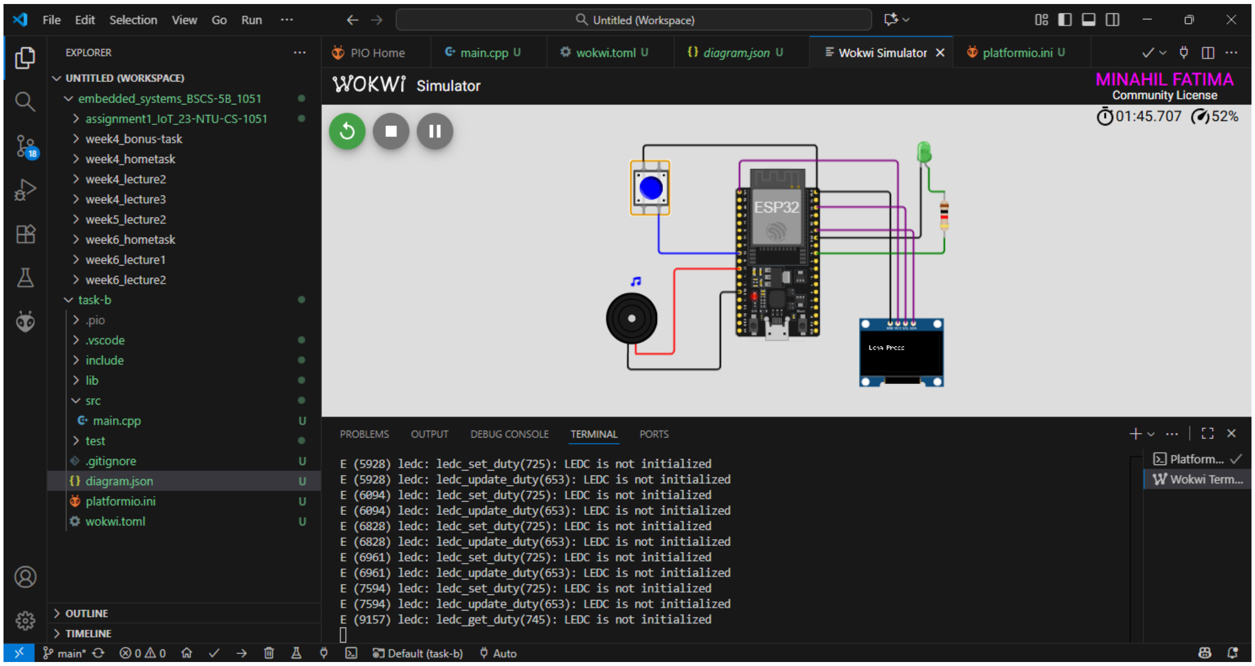
```
E (5928) ledc: ledc_set_duty(725): LEDC is not initialized
E (5928) ledc: ledc_update_duty(653): LEDC is not initialized
E (6094) ledc: ledc_set_duty(725): LEDC is not initialized
E (6094) ledc: ledc_update_duty(653): LEDC is not initialized
E (6828) ledc: ledc_set_duty(725): LEDC is not initialized
E (6828) ledc: ledc_update_duty(653): LEDC is not initialized
E (6961) ledc: ledc_set_duty(725): LEDC is not initialized
E (6961) ledc: ledc_update_duty(653): LEDC is not initialized
E (7594) ledc: ledc_set_duty(725): LEDC is not initialized
E (7594) ledc: ledc_update_duty(653): LEDC is not initialized
E (9157) ledc: ledc_get_duty(745): LEDC is not initialized
```

Short press LED OFF:



```
E (5296) ledc: ledc_update_duty(653): LEDC is not initialized
E (5396) ledc: ledc_set_duty(725): LEDC is not initialized
E (5396) ledc: ledc_update_duty(653): LEDC is not initialized
E (5829) ledc: ledc_set_duty(725): LEDC is not initialized
E (5829) ledc: ledc_update_duty(653): LEDC is not initialized
E (5962) ledc: ledc_set_duty(725): LEDC is not initialized
E (5962) ledc: ledc_update_duty(653): LEDC is not initialized
E (6829) ledc: ledc_set_duty(725): LEDC is not initialized
E (6829) ledc: ledc_update_duty(653): LEDC is not initialized
E (6962) ledc: ledc_set_duty(725): LEDC is not initialized
E (6962) ledc: ledc_update_duty(653): LEDC is not initialized
```

Long press buzzer:



```
E (5928) ledc: ledc_set_duty(725): LEDC is not initialized
E (5928) ledc: ledc_update_duty(653): LEDC is not initialized
E (6094) ledc: ledc_set_duty(725): LEDC is not initialized
E (6094) ledc: ledc_update_duty(653): LEDC is not initialized
E (6828) ledc: ledc_set_duty(725): LEDC is not initialized
E (6828) ledc: ledc_update_duty(653): LEDC is not initialized
E (6961) ledc: ledc_set_duty(725): LEDC is not initialized
E (6961) ledc: ledc_update_duty(653): LEDC is not initialized
E (7594) ledc: ledc_set_duty(725): LEDC is not initialized
E (7594) ledc: ledc_update_duty(653): LEDC is not initialized
E (9157) ledc: ledc_get_duty(745): LEDC is not initialized
```

Wokwi link:

https://wokwi.com/projects/445578717475938305