National Textile University, Faisalabad

Department of Computer Science

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

Assignment 1

Task-a

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

1. **All OFF**
2. **Alternate blink**
3. **All ON**
4. **PWM fade**

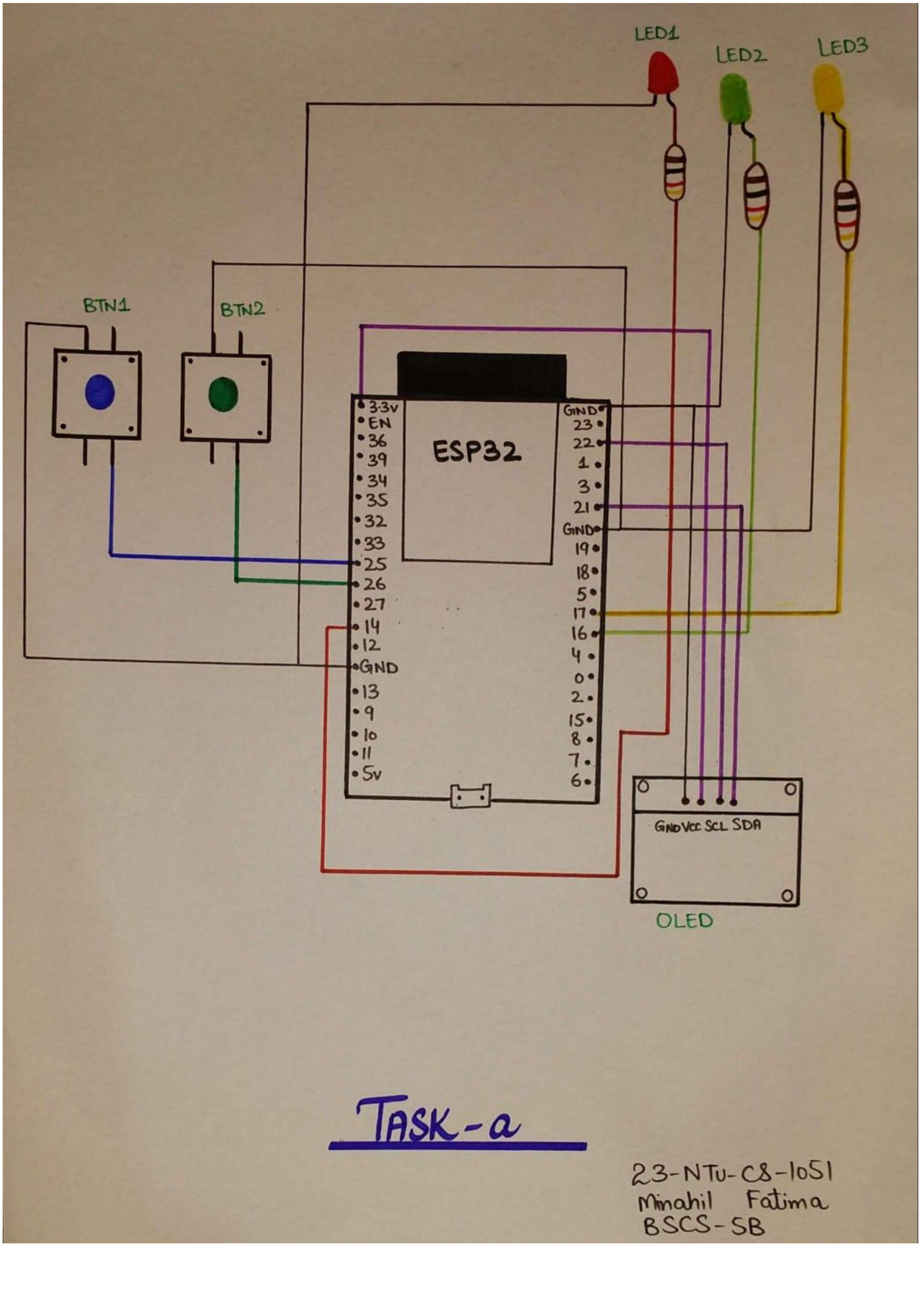
Use the second button to reset to OFF.

In this task, I used an ESP32 with three LEDs, two push buttons, and an OLED display. I programmed the first button to cycle through four LED modes: all off, alternate blink, all on, and PWM fade. The second button resets everything back to the "all off" mode. I also used the OLED display to show the current mode whenever it changes. To control the LED brightness, I used PWM channels, and for blinking effect, I used a hardware timer interrupt. I added debounce logic for the buttons to make sure they respond correctly.

Pin map:

Device name	Pin name	Pin number
OLED	VCC	3.3v
OLED	GND	GND
OLED	SCL	GPIO22
OLED	SDA	GPIO21
Button1	One side	GND
Button1	Other side	GPIO25
Button2	One side	GND
Button2	Other side	GPIO26
LED1	Cathode	GND
LED1	Anode	GPIO14 (through resistor)
LED2	Cathode	GND
LED2	Anode	GPIO16 (through resistor)
LED3	Cathode	GND
LED3	Anode	GPIO17 (through resistor)

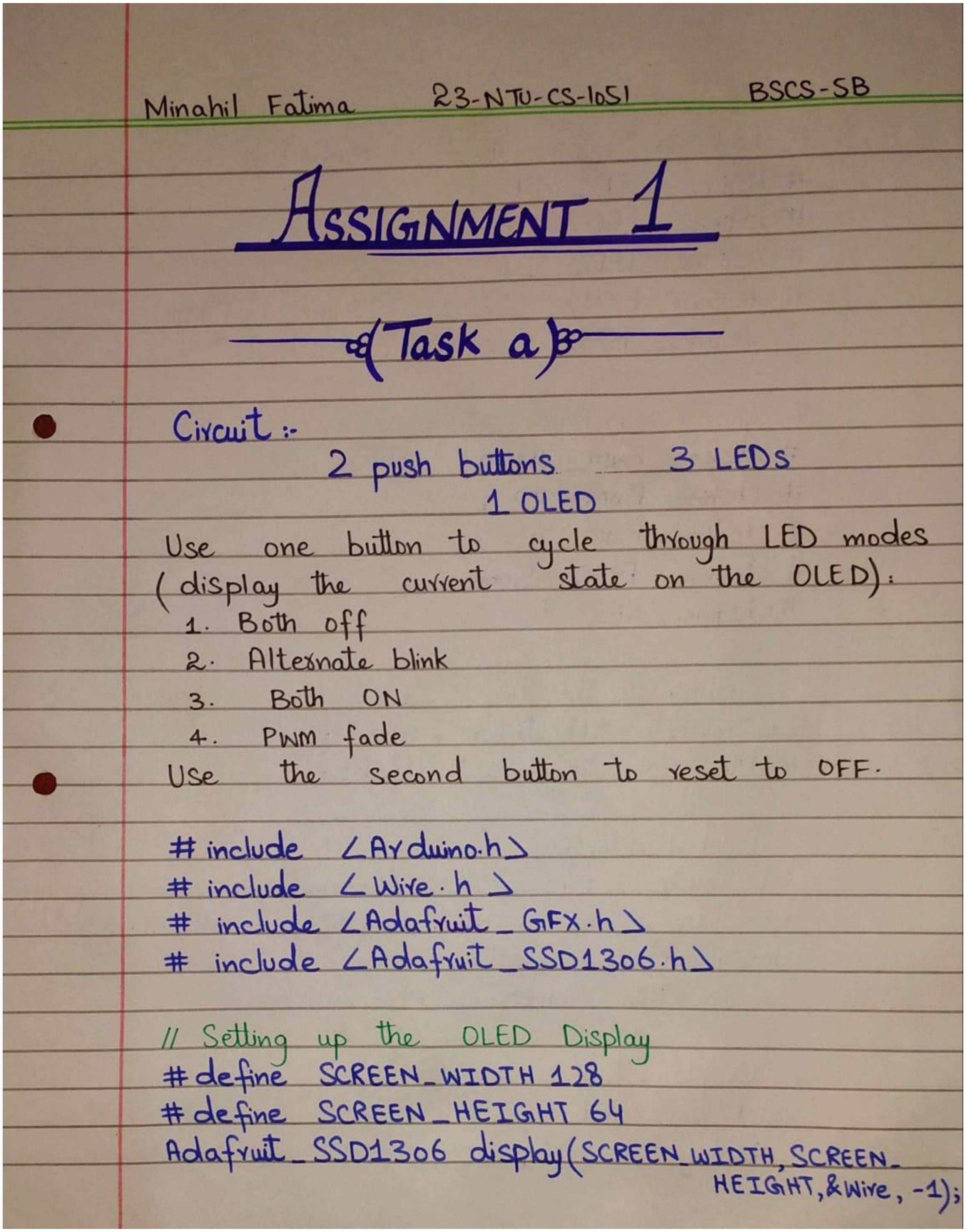
Circuit diagram:



TASK-a

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

Handwritten code pictures:



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

Assignment 1

(Task a)

Circuit :-

2 push buttons 3 LEDs

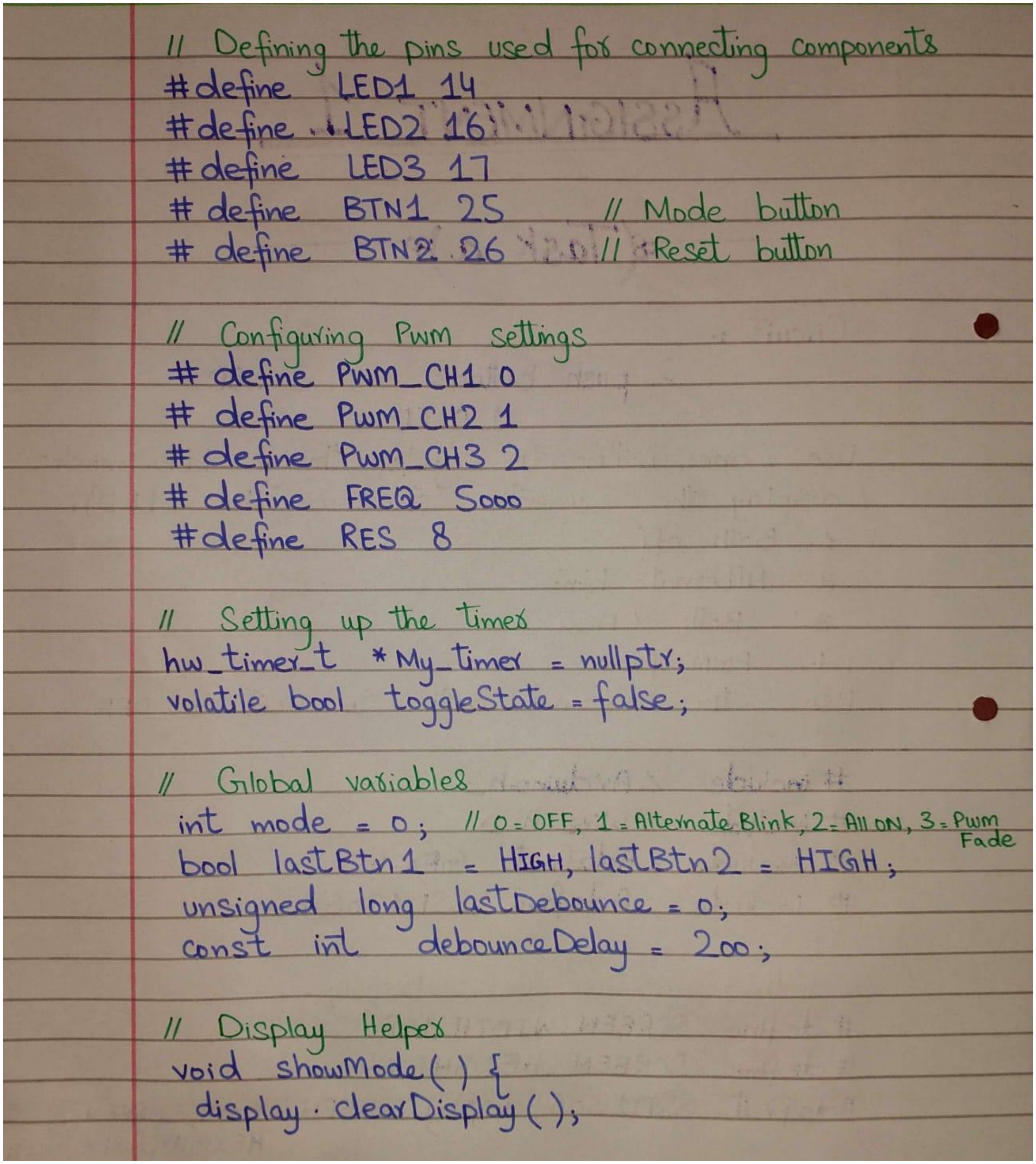
1 OLED

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

1. Both off
2. Alternate blink
3. Both ON
4. PWM fade

Use the second button to reset to OFF.

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

// Setting up the OLED Display
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_
                         HEIGHT, &Wire, -1);
```

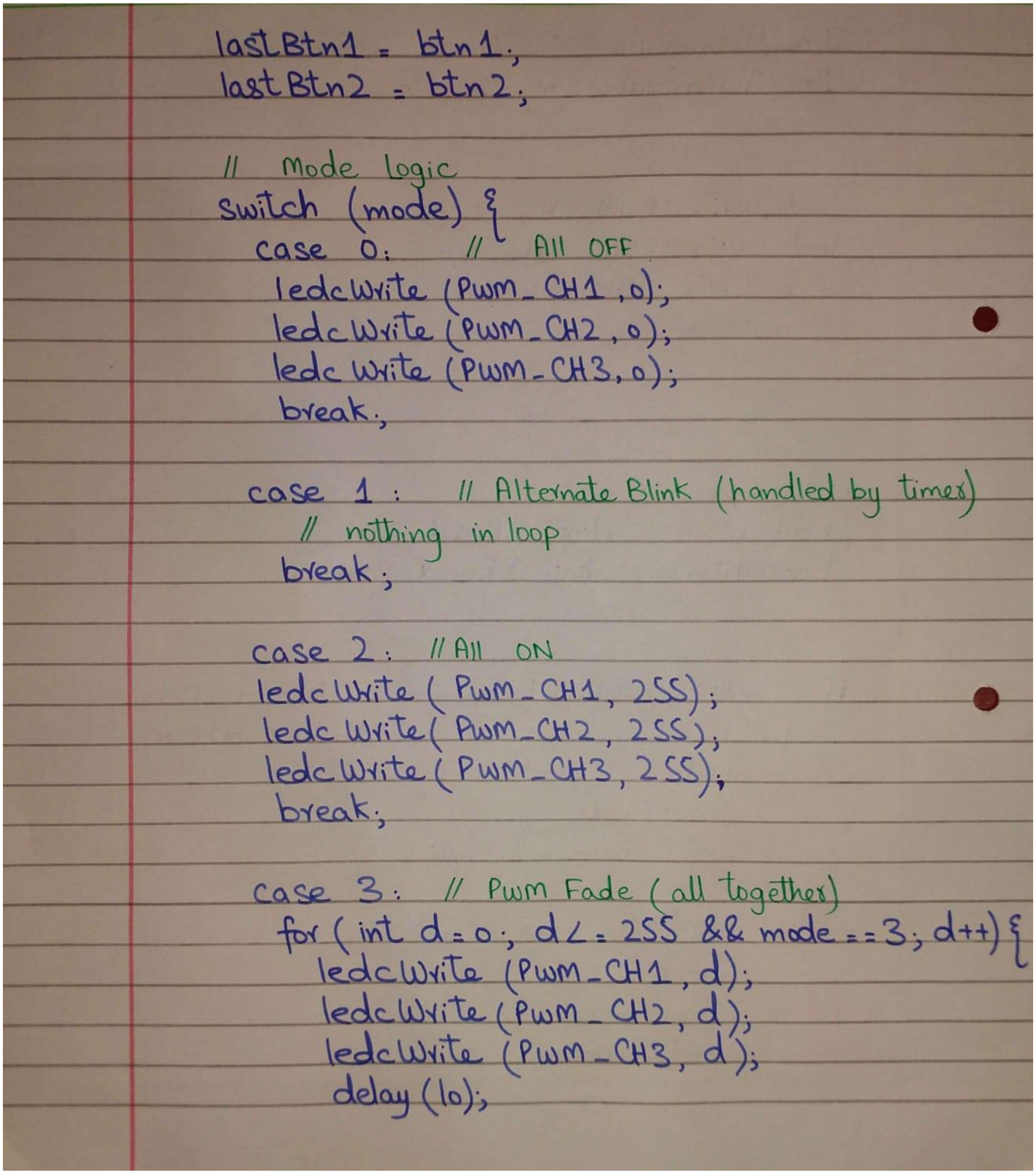


```
// Defining the pins used for connecting components
#define LED1 14
#define LED2 16
#define LED3 17
#define BTN1 25        // Mode button
#define BTN2 26        // Reset button

// Configuring Pwm settings
#define PwM_CH1 0
#define PwM_CH2 1
#define PwM_CH3 2
#define FREQ 5000
#define RES 8

// Setting up the timer
hw_timer_t *My_timer = nullptr;
volatile bool toggleState = false;

// Global variables
int mode = 0;   // 0 = OFF, 1 = Alternate Blink, 2 = All ON, 3 = PwM Fade
bool lastBtn1 = HIGH, lastBtn2 = HIGH;
unsigned long lastDebounce = 0;
const int debounceDelay = 200;

// Display Helper
void showMode() {
 display.clearDisplay();
```

```
    display.setTextSize(2);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(10, 20);
    switch (mode) {
        case 0: display.print("All OFF"); break;
        case 1: display.print("Alternate Blink"); break;
        case 2: display.print("All ON"); break;
        case 3: display.print("Pwm Fade"); break;
    }
    display.display();
}

// ISR Blinking
void IRAM_ATTR onTimer() {
    if (mode != 1) return;

    toggleState = !toggleState;

    ledcWrite(PWM_CH1, toggleState ? 255 : 0);
    ledcWrite(PWM_CH2, toggleState ?   0 : 255);
    ledcWrite(PWM_CH3, toggleState ? 255 : 0);
}

// Setup
void setup() {
    Serial.begin(115200);
```

```
pinMode(LED1, OUTPUT);
pinMode(LED2, OUTPUT);
pinMode (LED3, OUTPUT);
pinMode( BTN1, INPUT_PULLUP);
pinMode( BTN2, INPUT_PULLUP);

// OLED setup
if (!display.begin(SSD1306_SWITCHAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
}
display.clearDisplay();
display.display();

// Pwm setup
ledcSetup(PWM_CH1, FREQ, RES);
ledcSetup(PWM_CH2, FREQ, RES);
ledcSetup(PWM_CH3, FREQ, RES);
ledcAttachPin(LED1, PWM_CH1);
ledcAttachPin(LED2, PWM_CH2);
ledcAttachPin(LED3, PWM_CH3);

// Timer setup for blinking
My_timer = timerBegin(0, 80, true);  // 1 tick = 1 us
                                         (80 MHz/80)
timerAttachInterrupt(My_timer, &onTimer, true);
```

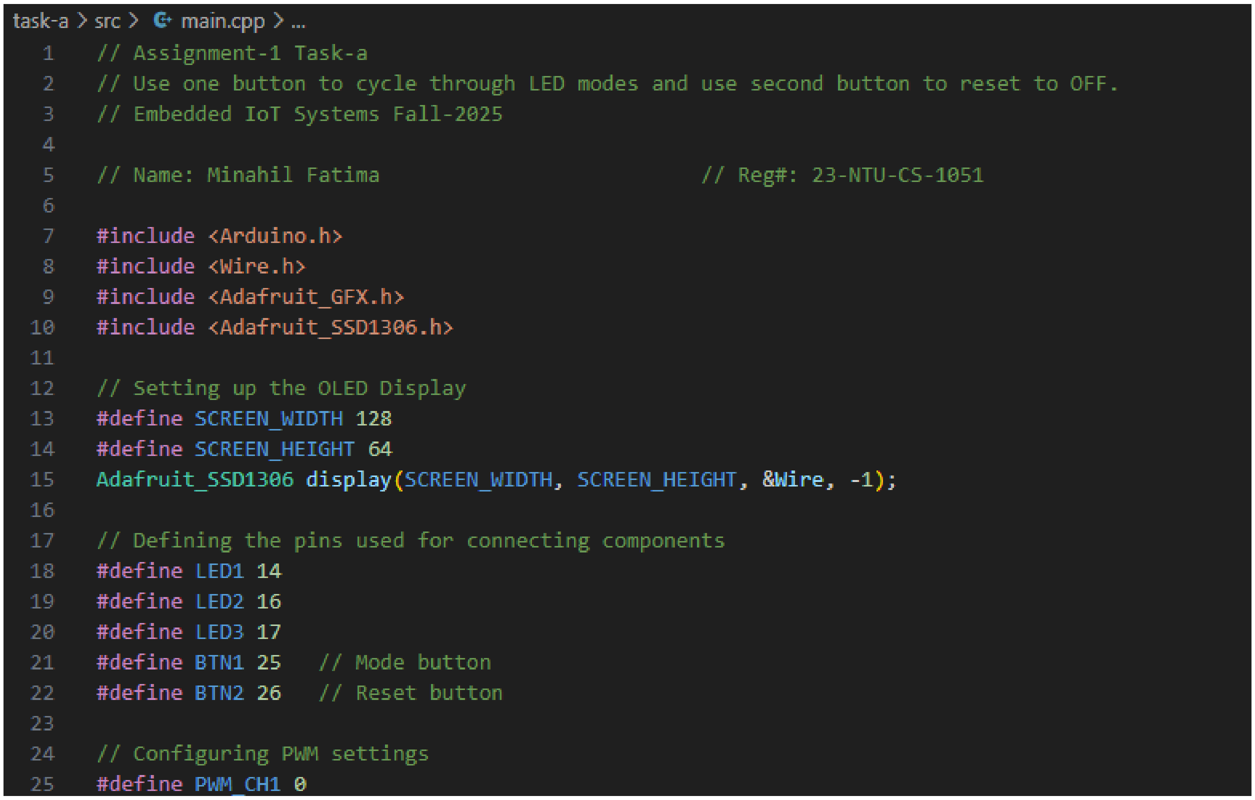
```
timerAlarmWrite(My_timer, 500000, true);   // Toggle every
                                                0.5 sec
timerAlarmEnable(my_timer);

// Initial State
 ledcWrite(PWM_CH1, 0);
 ledcWrite(PWM_CH2, 0);
 ledc write(PWM_CH3, 0);
 showMode();
}

// Loop
void loop() {
  bool btn1 = digitalRead(BTN1);
  bool btn2 = digitalRead(BTN2);

// Debounce logic
if (millis() - lastDebounce > debounceDelay) {
   if (btn1 == Low && lastBtn1 == HIGH) {
      mode = (mode + 1) % 4;
      show Mode();
      lastDebounce = millis();
   }
   if (btn2 == low && lastBtn2 == HIGH) {
      mode = 0;
      ShowMode();
      lastDebounce = millis();
   }
}
```



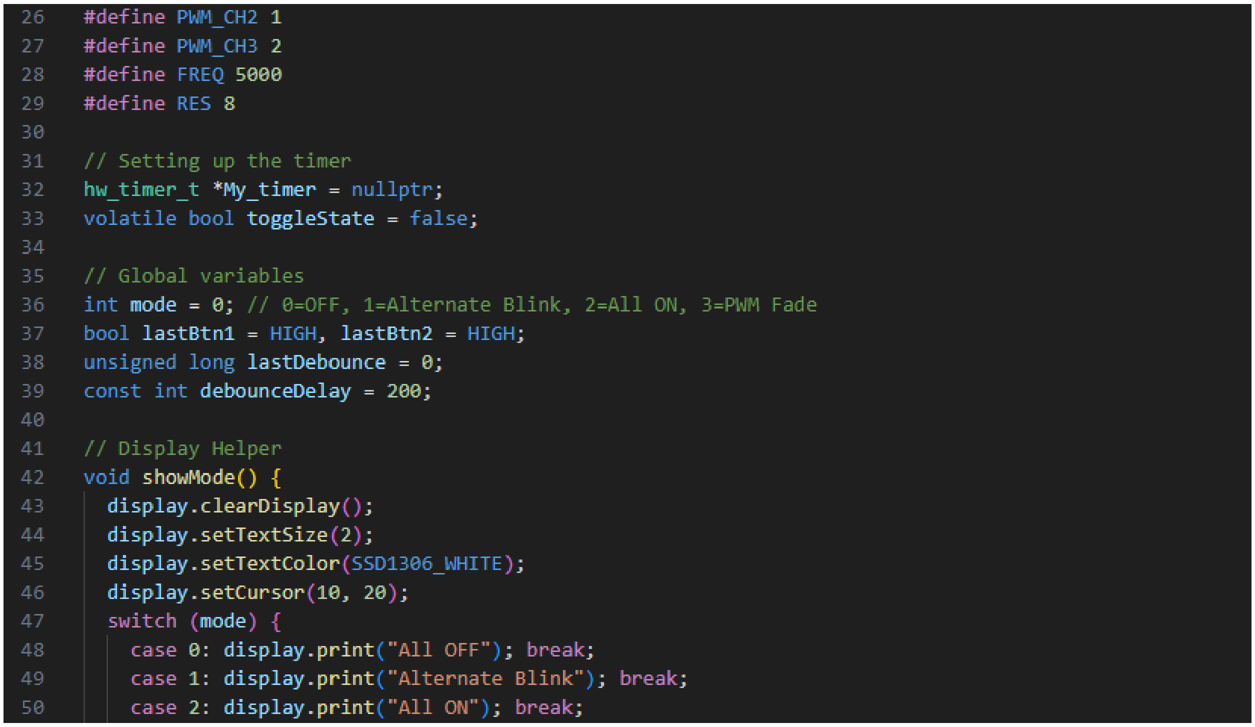
```
lastBtn1 = btn1;
lastBtn2 = btn2;

// Mode logic
switch (mode) {
  case 0:     // All OFF
    ledcWrite (PWM_CH1 ,0);
    ledcWrite (PWM_CH2 , 0);
    ledcWrite (PWM_CH3, 0);
    break;

  case 1 :   // Alternate Blink (handled by timer)
    // nothing in loop
    break;

  case 2 : //All ON
   ledcWrite ( PWM_CH1, 255);
   ledcWrite( PWM_CH2 , 255);
   ledcWrite ( PWM_CH3 , 255);
    break;

  case 3 : // PWM Fade (all together)
    for ( int d = 0; d<= 255 && mode == 3; d++) {
      ledcWrite (PWM_CH1 , d);
      ledcWrite (PWM_CH2, d);
      ledcWrite (PWM_CH3, d);
      delay (10);
```

```
     if (digitalRead (BTN1) == low || digitalRead (BTN2)
                                == low) return;
   }
  for (int d= 255; d>=0 && mode ==3 ; d--) {
     ledcWrite (PWM_CH1, d);
     ledcWrite (PWM_CH2, d);
     ledcWrite (PWM_CH3, d);
     delay (10);
     if (digitalRead (BTN1) == low || digitalRead
                           (BTN2) == low) return;
      }
     break;
  }
}
```

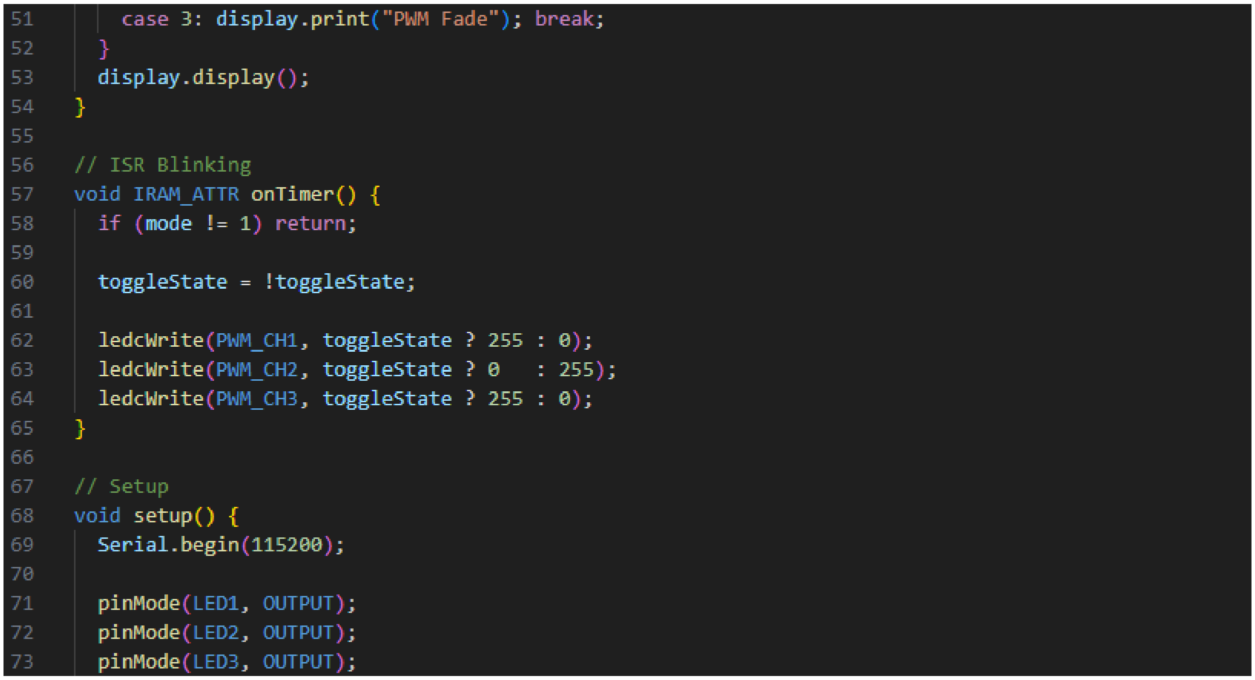
Code screenshots:



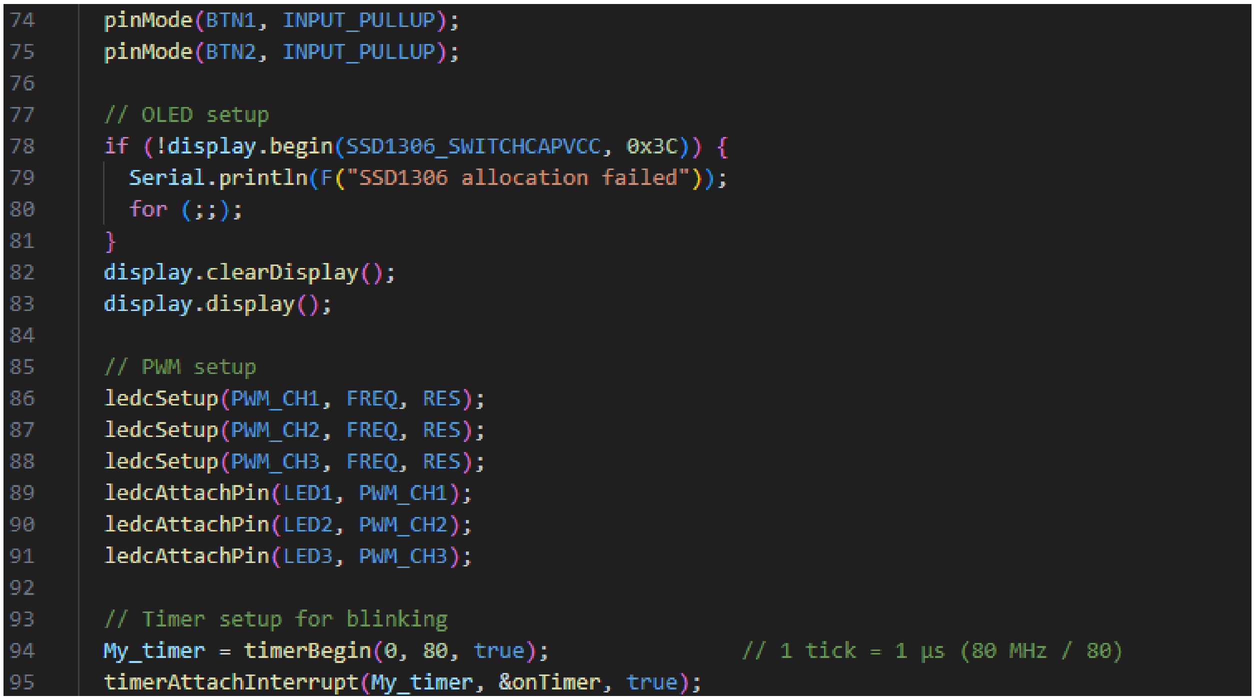
```cpp
// Assignment-1 Task-a
// Use one button to cycle through LED modes and use second button to reset to OFF.
// Embedded IoT Systems Fall-2025

// Name: Minahil Fatima                          // Reg#: 23-NTU-CS-1051

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

// Setting up the OLED Display
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// Defining the pins used for connecting components
#define LED1 14
#define LED2 16
#define LED3 17
#define BTN1 25   // Mode button
#define BTN2 26   // Reset button

// Configuring PWM settings
#define PWM_CH1 0
```



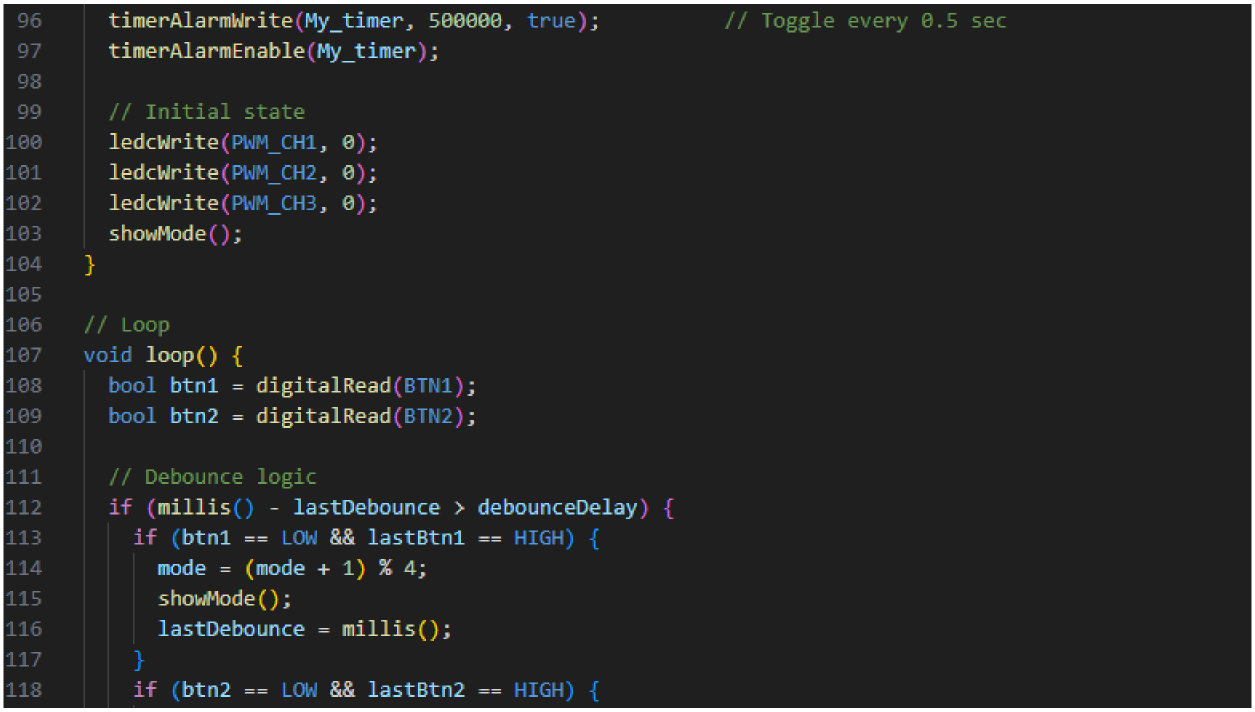
```cpp
#define PWM_CH2 1
#define PWM_CH3 2
#define FREQ 5000
#define RES 8

// Setting up the timer
hw_timer_t *My_timer = nullptr;
volatile bool toggleState = false;

// Global variables
int mode = 0; // 0=OFF, 1=Alternate Blink, 2=All ON, 3=PWM Fade
bool lastBtn1 = HIGH, lastBtn2 = HIGH;
unsigned long lastDebounce = 0;
const int debounceDelay = 200;

// Display Helper
void showMode() {
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 20);
  switch (mode) {
    case 0: display.print("All OFF"); break;
    case 1: display.print("Alternate Blink"); break;
    case 2: display.print("All ON"); break;
```



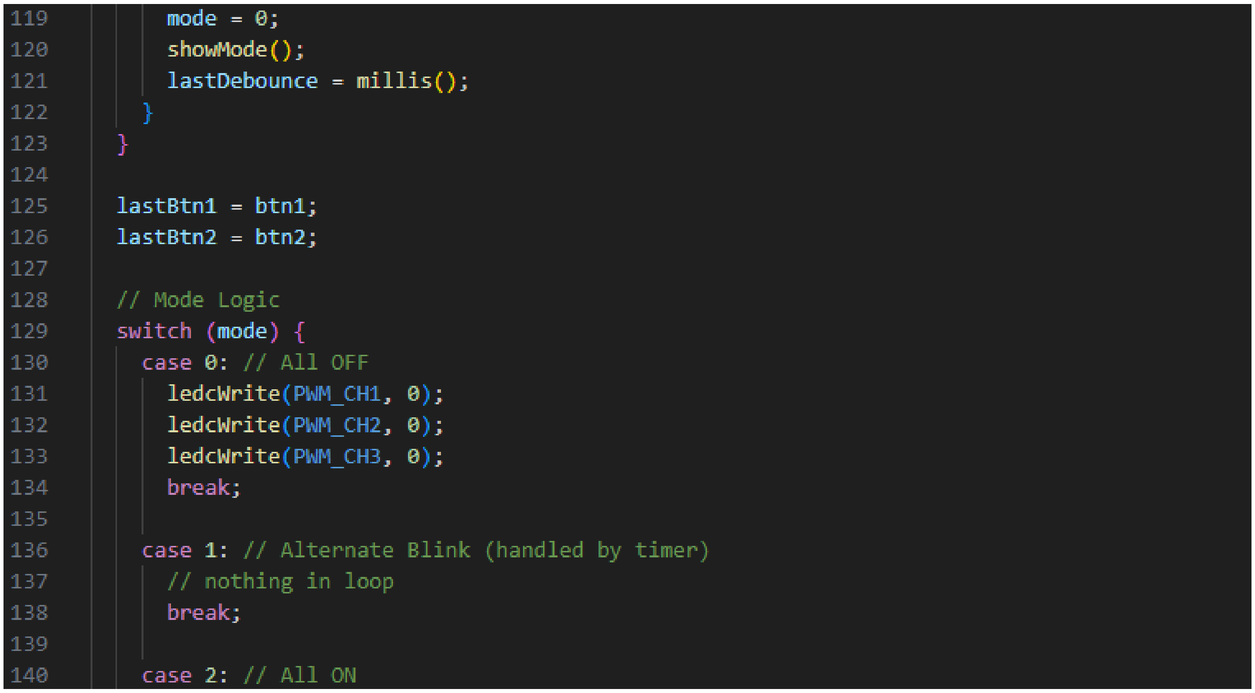
```
    case 3: display.print("PWM Fade"); break;
  }
  display.display();
}

// ISR Blinking
void IRAM_ATTR onTimer() {
  if (mode != 1) return;

  toggleState = !toggleState;

  ledcWrite(PWM_CH1, toggleState ? 255 : 0);
  ledcWrite(PWM_CH2, toggleState ? 0   : 255);
  ledcWrite(PWM_CH3, toggleState ? 255 : 0);
}

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

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
```



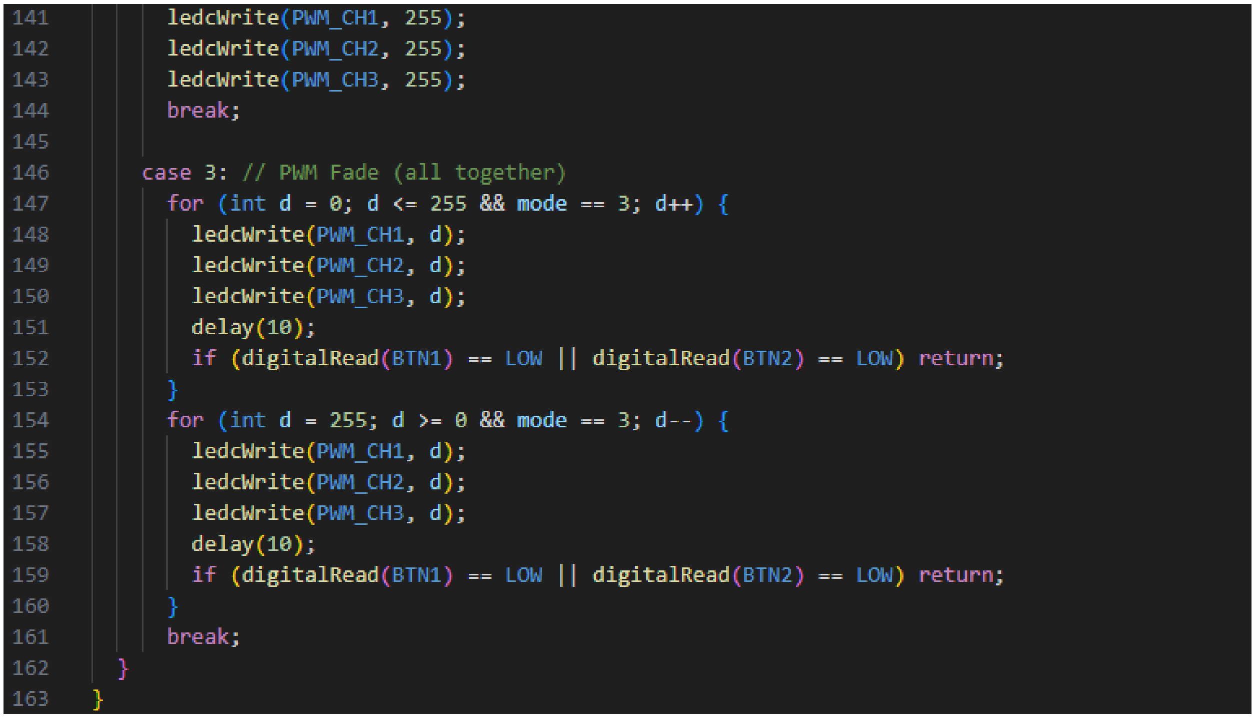
```
  pinMode(BTN1, INPUT_PULLUP);
  pinMode(BTN2, INPUT_PULLUP);

  // OLED setup
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
  }
  display.clearDisplay();
  display.display();

  // PWM setup
  ledcSetup(PWM_CH1, FREQ, RES);
  ledcSetup(PWM_CH2, FREQ, RES);
  ledcSetup(PWM_CH3, FREQ, RES);
  ledcAttachPin(LED1, PWM_CH1);
  ledcAttachPin(LED2, PWM_CH2);
  ledcAttachPin(LED3, PWM_CH3);

  // Timer setup for blinking
  My_timer = timerBegin(0, 80, true);                // 1 tick = 1 µs (80 MHz / 80)
  timerAttachInterrupt(My_timer, &onTimer, true);
```



```
  timerAlarmWrite(My_timer, 500000, true);          // Toggle every 0.5 sec
  timerAlarmEnable(My_timer);

  // Initial state
  ledcWrite(PWM_CH1, 0);
  ledcWrite(PWM_CH2, 0);
  ledcWrite(PWM_CH3, 0);
  showMode();
}

// Loop
void loop() {
  bool btn1 = digitalRead(BTN1);
  bool btn2 = digitalRead(BTN2);

  // Debounce logic
  if (millis() - lastDebounce > debounceDelay) {
    if (btn1 == LOW && lastBtn1 == HIGH) {
      mode = (mode + 1) % 4;
      showMode();
      lastDebounce = millis();
    }
    if (btn2 == LOW && lastBtn2 == HIGH) {
```

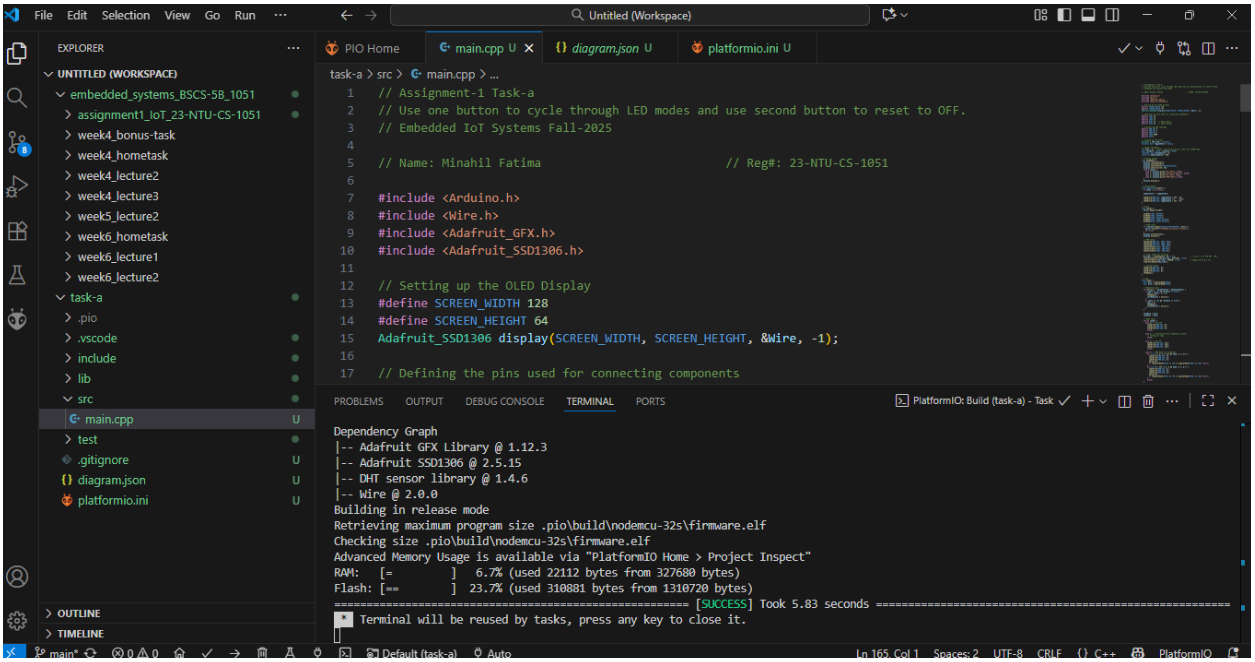


```
      mode = 0;
      showMode();
      lastDebounce = millis();
    }
  }

  lastBtn1 = btn1;
  lastBtn2 = btn2;

  // Mode Logic
  switch (mode) {
    case 0: // All OFF
      ledcWrite(PWM_CH1, 0);
      ledcWrite(PWM_CH2, 0);
      ledcWrite(PWM_CH3, 0);
      break;

    case 1: // Alternate Blink (handled by timer)
      // nothing in loop
      break;

    case 2: // All ON
```



```
    ledcWrite(PWM_CH1, 255);
    ledcWrite(PWM_CH2, 255);
    ledcWrite(PWM_CH3, 255);
    break;

   case 3: // PWM Fade (all together)
    for (int d = 0; d <= 255 && mode == 3; d++) {
     ledcWrite(PWM_CH1, d);
     ledcWrite(PWM_CH2, d);
     ledcWrite(PWM_CH3, d);
     delay(10);
     if (digitalRead(BTN1) == LOW || digitalRead(BTN2) == LOW) return;
    }
    for (int d = 255; d >= 0 && mode == 3; d--) {
     ledcWrite(PWM_CH1, d);
     ledcWrite(PWM_CH2, d);
     ledcWrite(PWM_CH3, d);
     delay(10);
     if (digitalRead(BTN1) == LOW || digitalRead(BTN2) == LOW) return;
    }
    break;
  }
 }
```

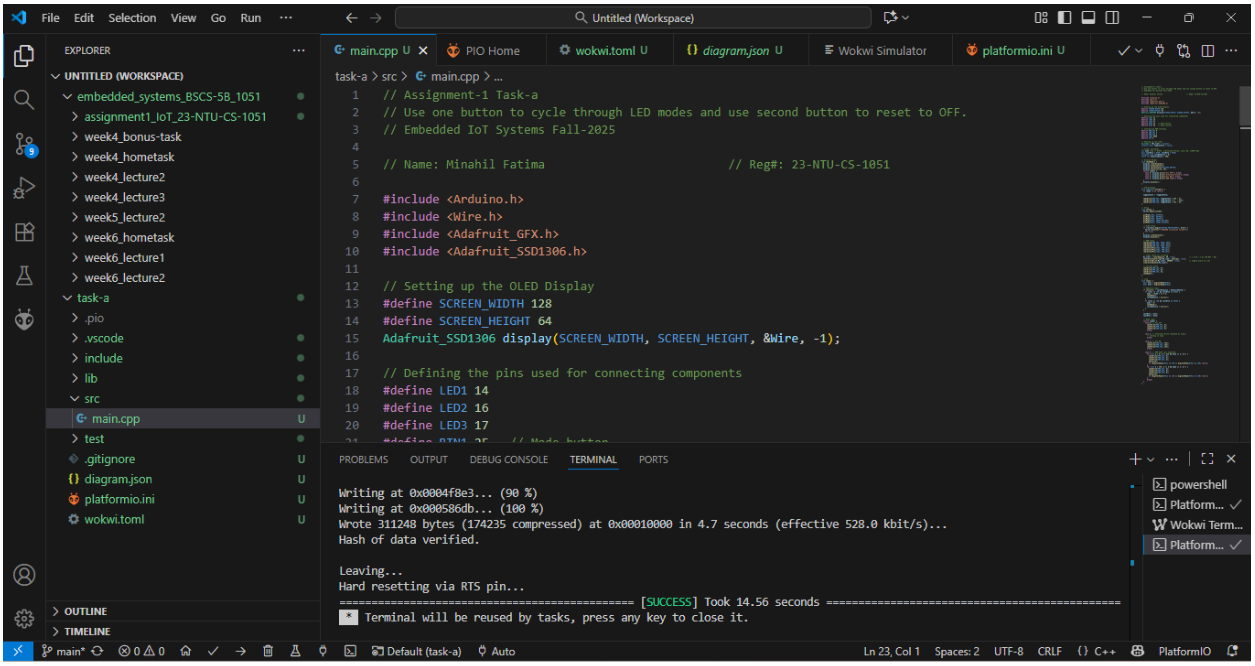
Code build success:



```
// Assignment-1 Task-a
// Use one button to cycle through LED modes and use second button to reset to OFF.
// 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>

// Setting up the OLED Display
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// Defining the pins used for connecting components
```

```
Dependency Graph
|-- Adafruit GFX Library @ 1.12.3
|-- Adafruit SSD1306 @ 2.5.15
|-- DHT sensor library @ 1.4.6
|-- Wire @ 2.0.0
Building in release mode
Retrieving maximum program size .pio\build\nodemcu-32s\firmware.elf
Checking size .pio\build\nodemcu-32s\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM:   [=         ]   6.7% (used 22112 bytes from 327680 bytes)
Flash: [==        ]  23.7% (used 310881 bytes from 1310720 bytes)
=========================================================== [SUCCESS] Took 5.83 seconds ===========================================================
 * Terminal will be reused by tasks, press any key to close it.
```

Code upload success:

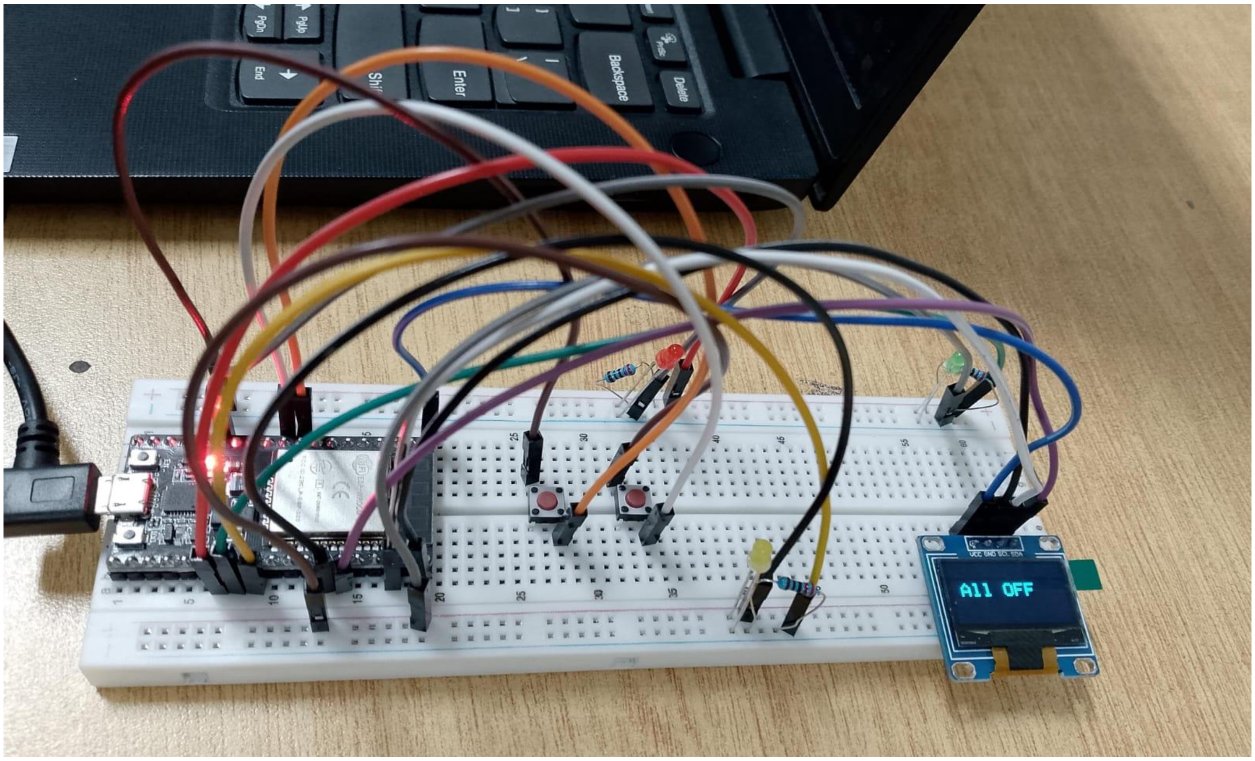


```cpp
// Assignment-1 Task-a
// Use one button to cycle through LED modes and use second button to reset to OFF.
// Embedded IoT Systems Fall-2025

// Name: Minahil Fatima                          // Reg#: 23-NTU-CS-1051

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

// Setting up the OLED Display
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// Defining the pins used for connecting components
#define LED1 14
#define LED2 16
#define LED3 17
```

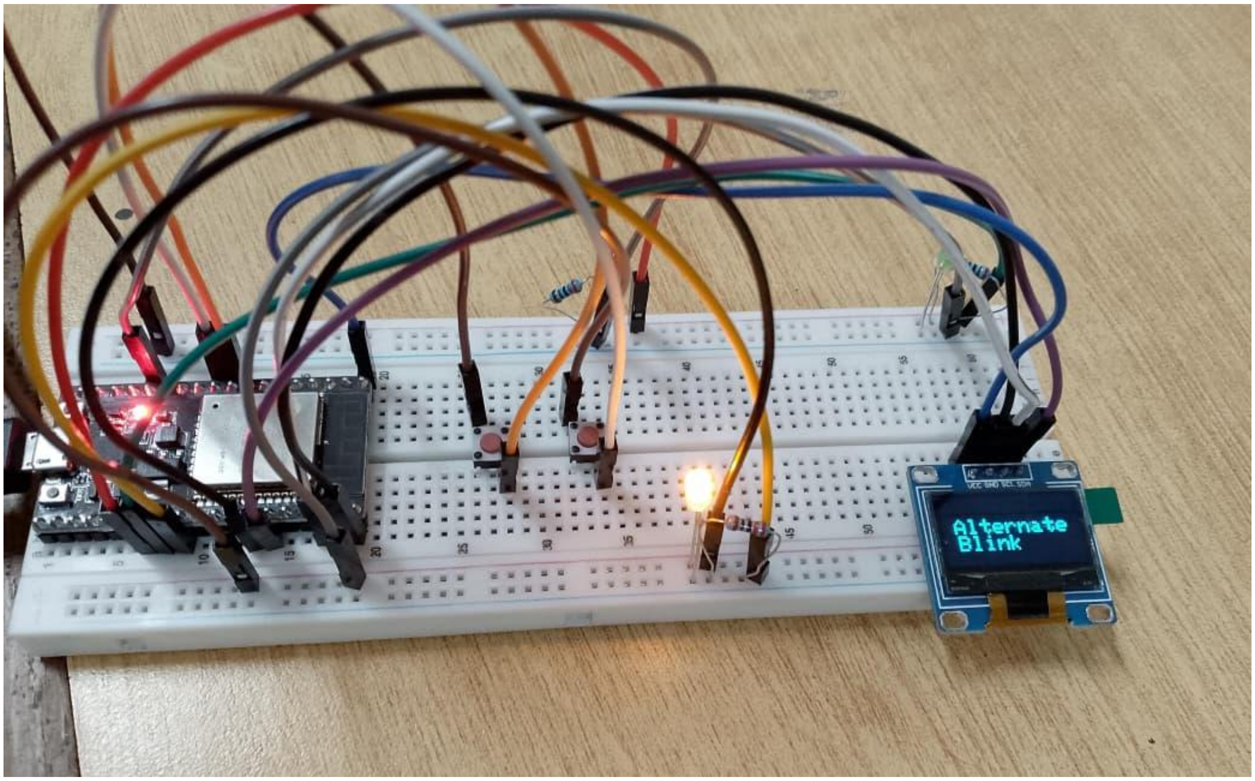
```
Writing at 0x0004f8e3... (90 %)
Writing at 0x000586db... (100 %)
Wrote 311248 bytes (174235 compressed) at 0x00010000 in 4.7 seconds (effective 528.0 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
============================================= [SUCCESS] Took 14.56 seconds =============================================
 * Terminal will be reused by tasks, press any key to close it.
```

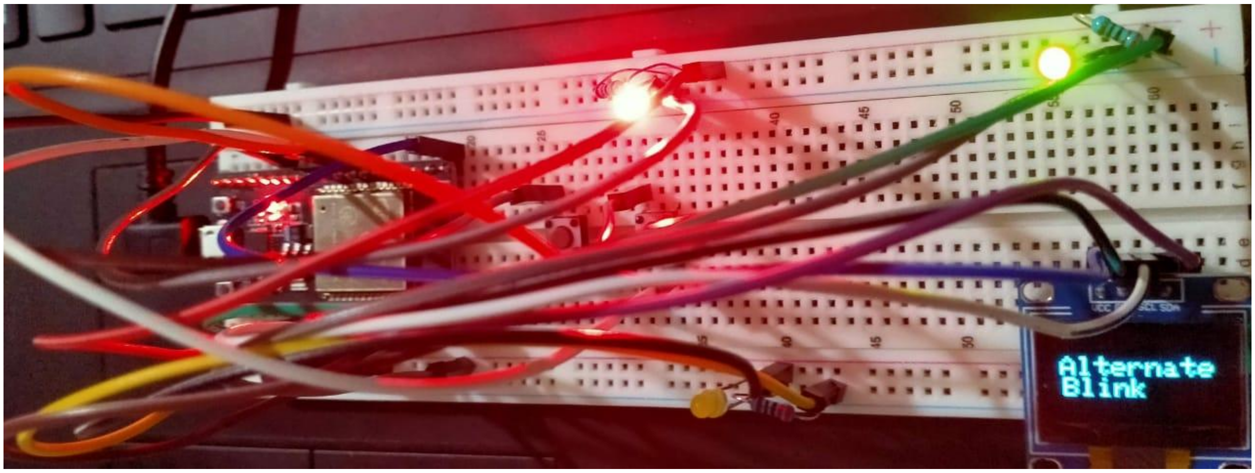
Kit output pictures:

All OFF

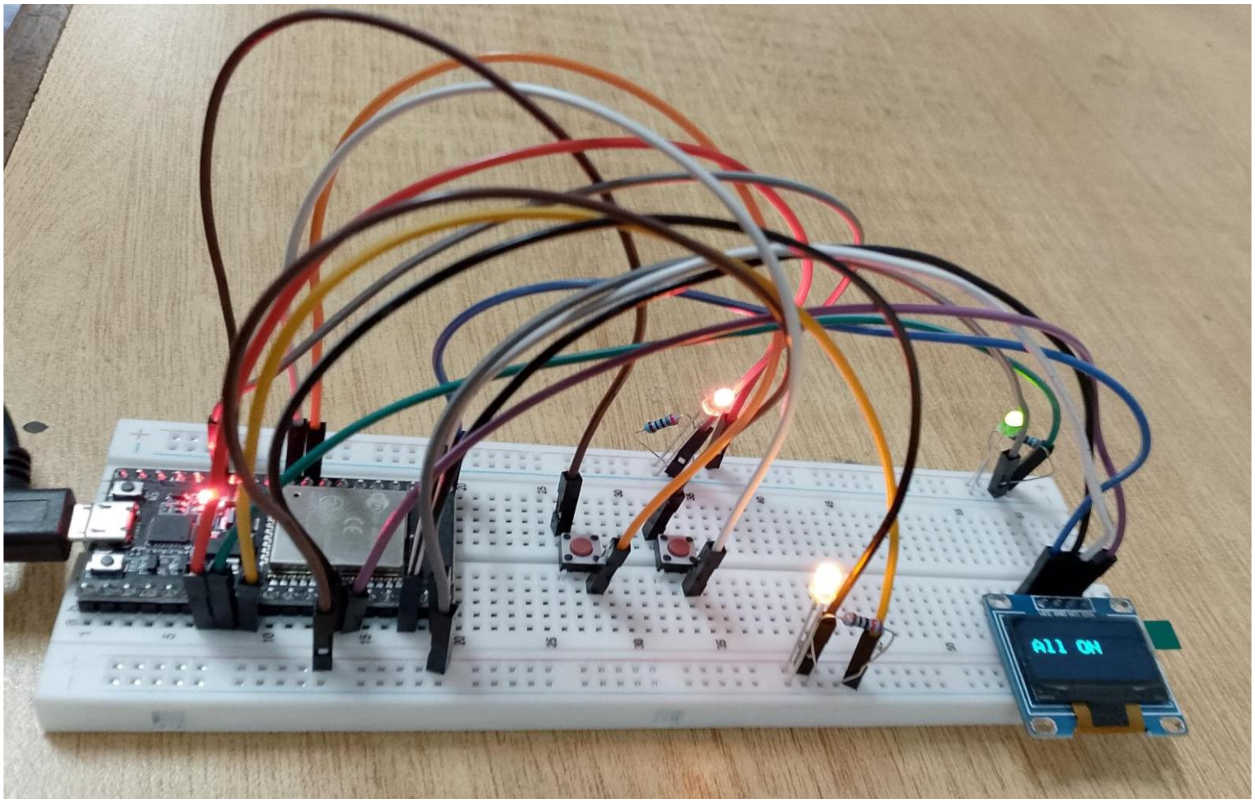


Alternate Blink

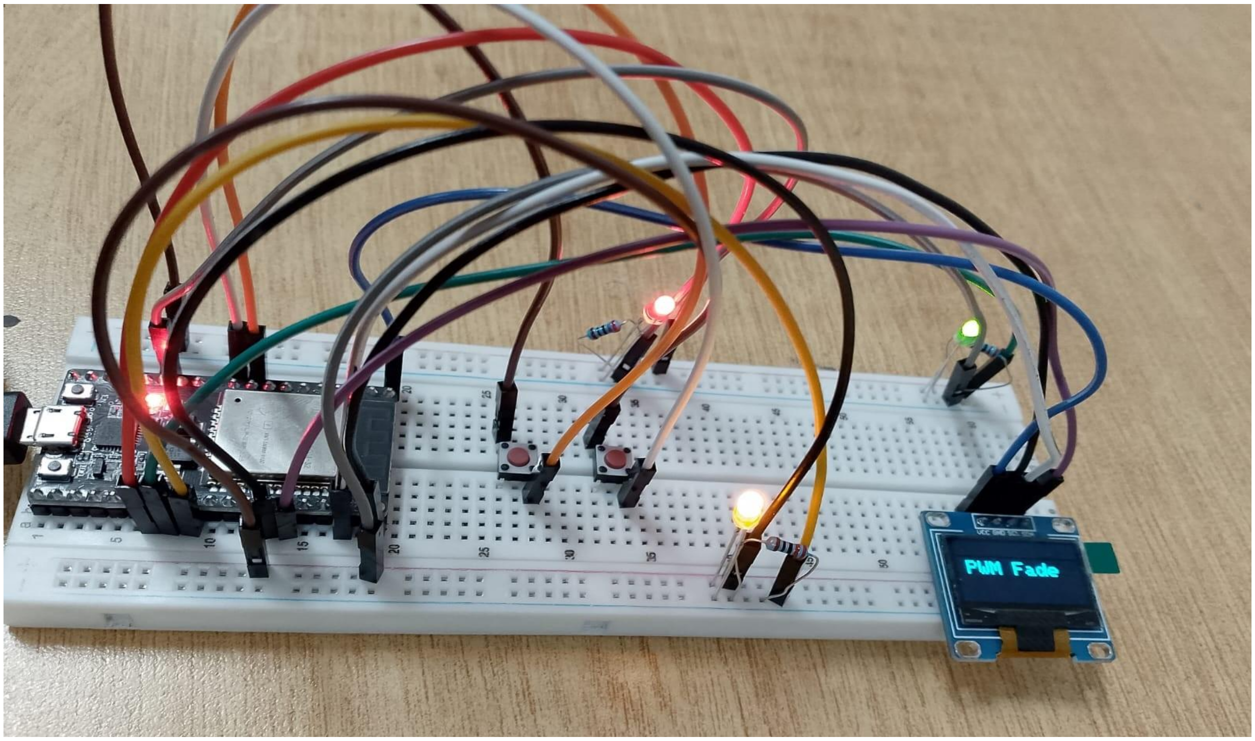


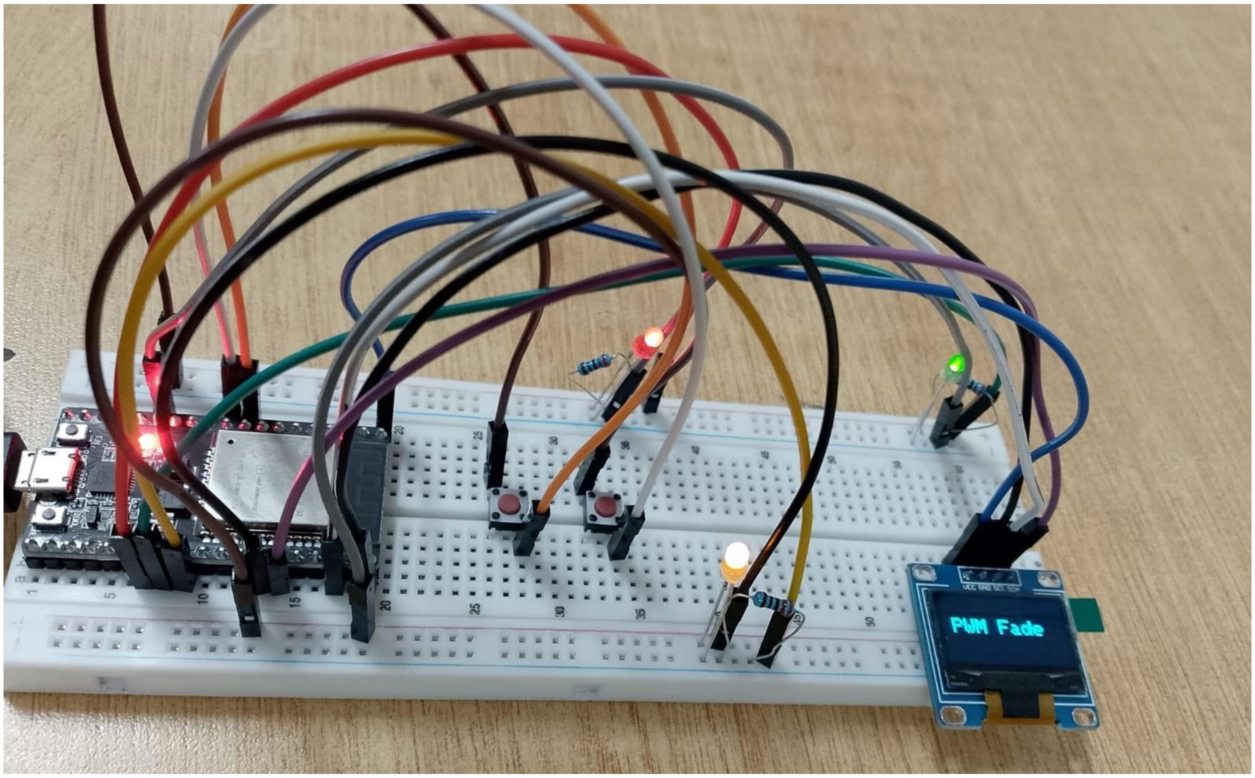


All ON



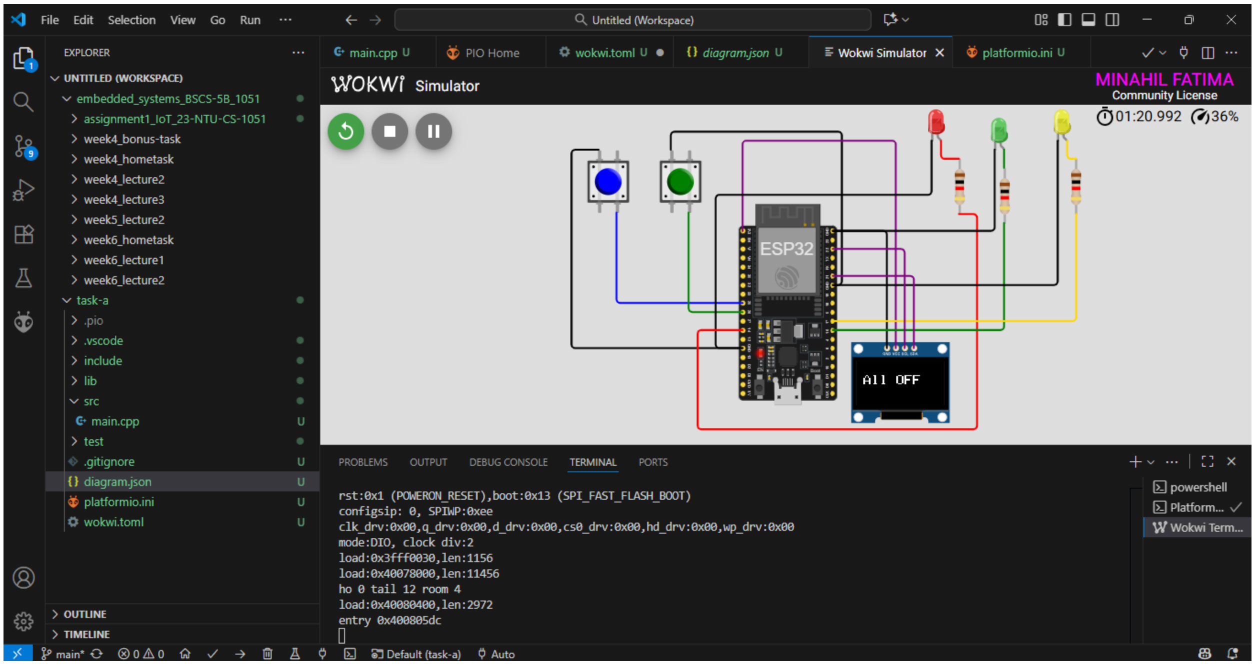
PWM Fade





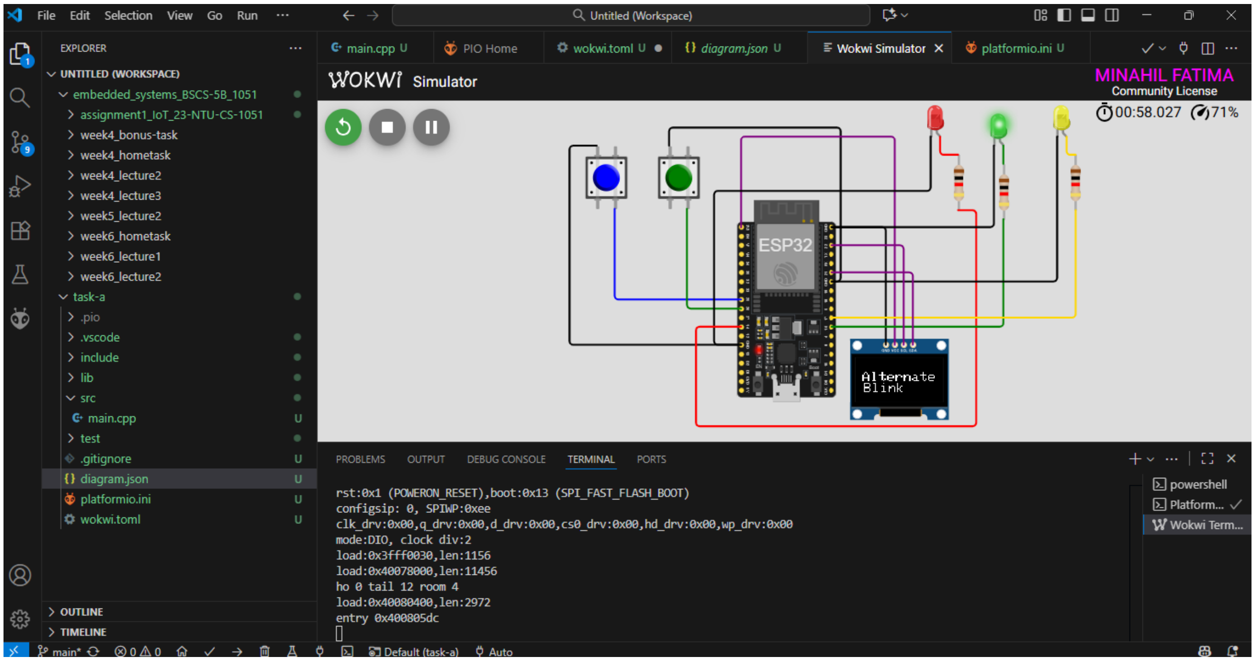
Wokwi output pictures:

All OFF



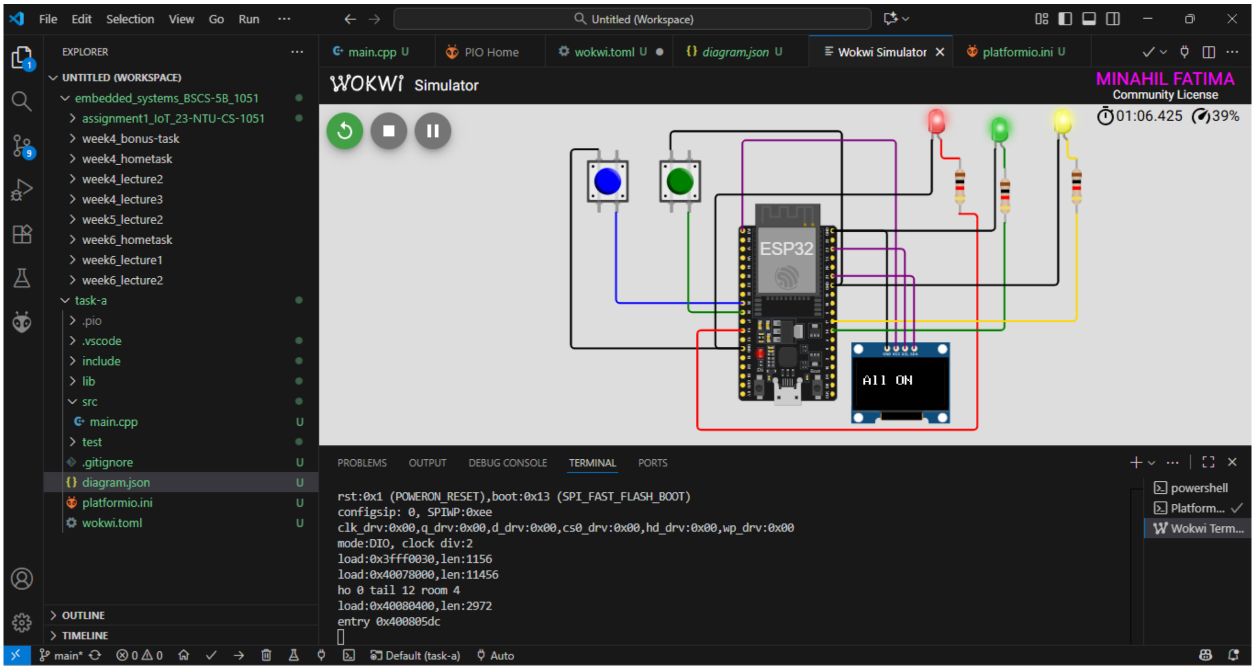
```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
```

Alternate blink



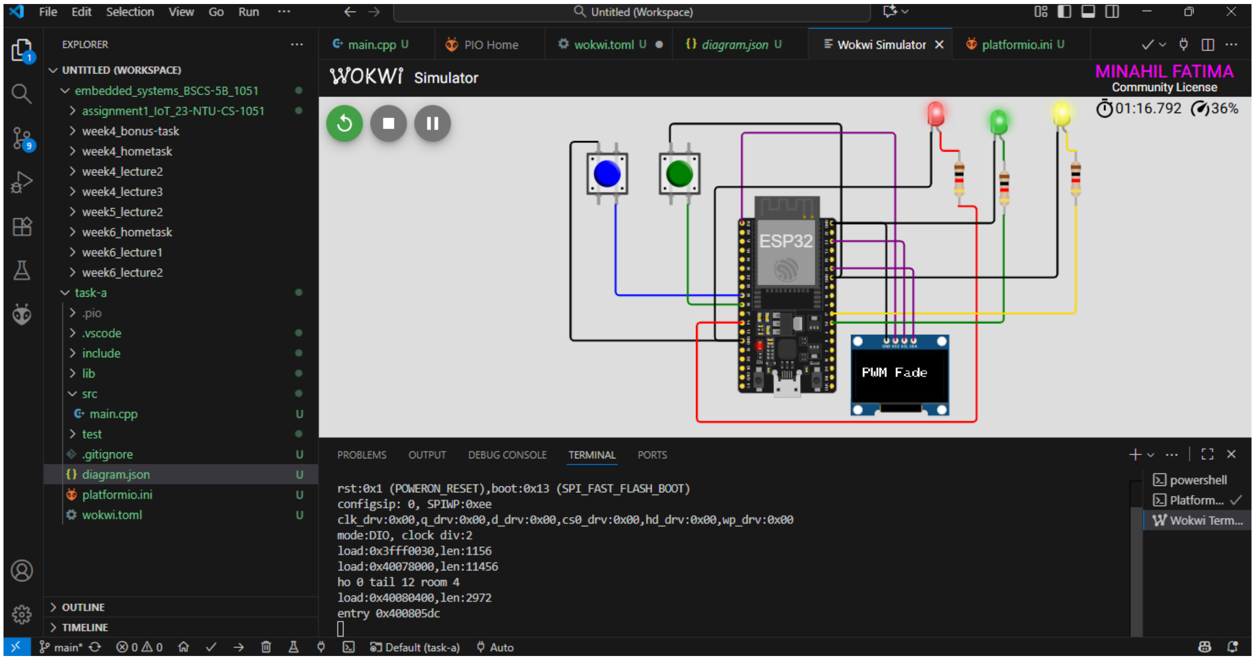
```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
```

All ON



```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
```

PWM fade



```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
```

Wokwi link:

https://wokwi.com/projects/445509042415788033