

National Textile University, Faisalabad



Department of Computer Science

Name:	Muhammad Umar Mushtaq
Class:	BSCS_B 5 th Semester
Registration No:	23-NTU-CS-1077
Assignment:	01
Course Name:	Embedded Iot System
Submitted To:	Sir Nasir Mehmood
Submission Date:	25/10/2025

Description

Task_1

Multi-Device Control using ESP32 (LEDs, Buttons, OLED, and Buzzer)

Description:

In this task, an **ESP32 DevKit-C V4** is programmed to control **three LEDs**, **two push buttons**, a **buzzer**, and an **OLED display** using the **Adafruit SSD1306** library.

Working:

- Each **button press** performs a specific action — such as toggling LEDs or turning ON the buzzer.
- The **OLED display** shows clear text messages for user feedback like “LED ON”, “LED OFF”, or “Buzzer Active”.
- LEDs are connected to **GPIO 2, 4, and 5**, the buzzer to **GPIO 15**, and buttons to **GPIO 26 and 27** with internal pull-ups.
- The **I²C OLED** uses **GPIO 21 (SDA)** and **GPIO 22 (SCL)** with address 0x3C.
- Proper resistors (420Ω) are used with LEDs to limit current.
- The system was simulated on **Wokwi** to test circuit behavior.

Time & Execution:

On pressing a button, the ESP32 reads the input, updates the respective LED/buzzer state, and instantly displays the message on the OLED. The whole response time is **less than 1 second**, ensuring real-time feedback.

Objective:

To learn how to interface multiple input/output devices with ESP32 and display real-time feedback using an OLED display.

Task_2

This task demonstrates **button press duration detection** using an **ESP32**, differentiating between **short** and **long presses**.

The system includes a **push button**, **LED**, **buzzer**, and **OLED display** for feedback.

Working:

- When the button (GPIO 25) is pressed and released, the code calculates how long it was held using `millis()`.
- If the press duration is **less than 1.5 seconds**, it is treated as a **short press**, and the **LED (GPIO 5)** toggles its state.
- If the press is **longer than 1.5 seconds**, a **buzzer (GPIO 18)** activates for 0.5 seconds to indicate a **long press**.
- The **OLED (I²C: SDA 21, SCL 22)** displays messages like “Short Press → LED Toggle” or “Long Press → Buzzer”.
- The system resets automatically after each press for the next detection.

Time & Execution:

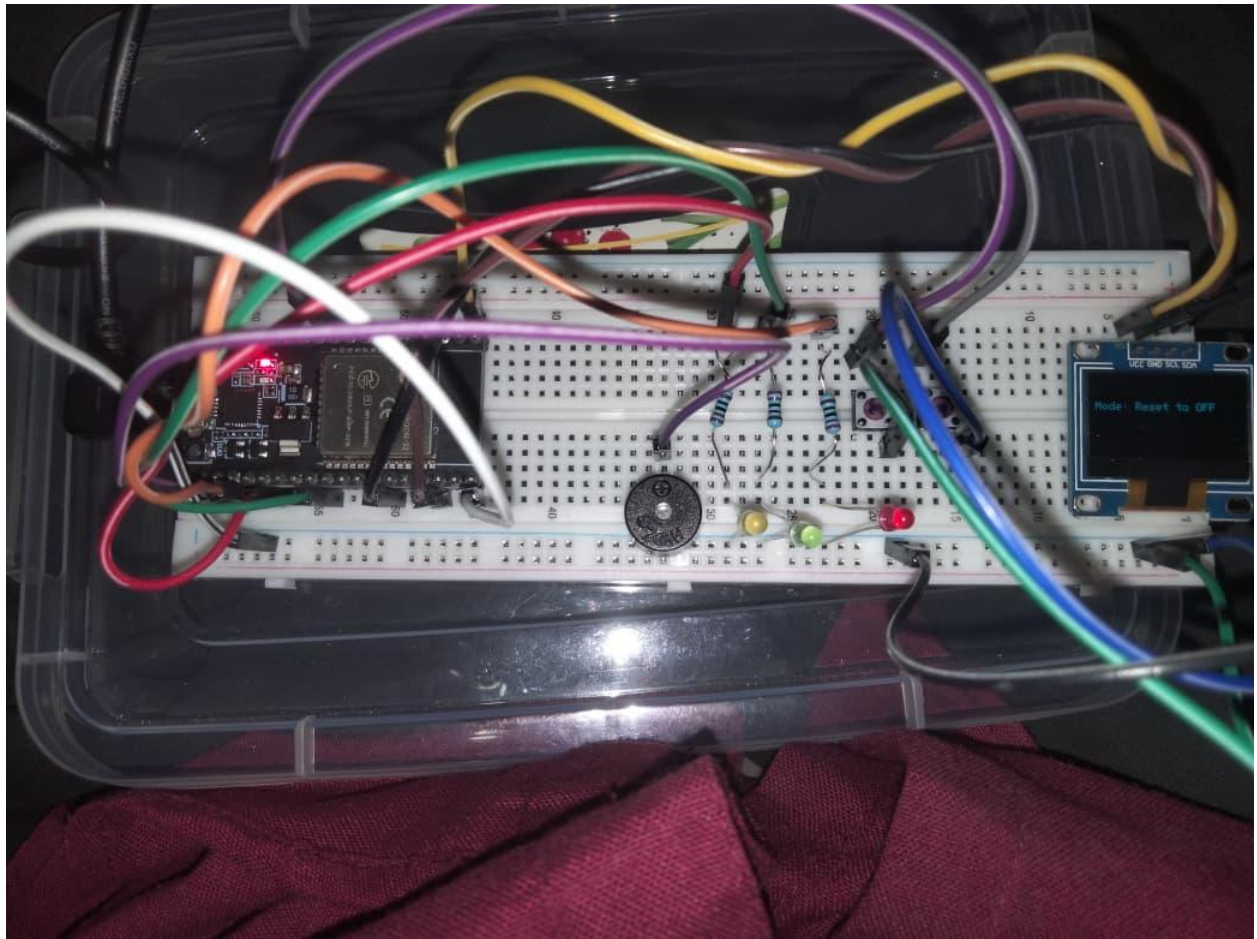
- Short press duration: **< 1.5 seconds**
- Long press duration: **> 1.5 seconds**
- OLED updates instantly after each event with clear text.

Objective:

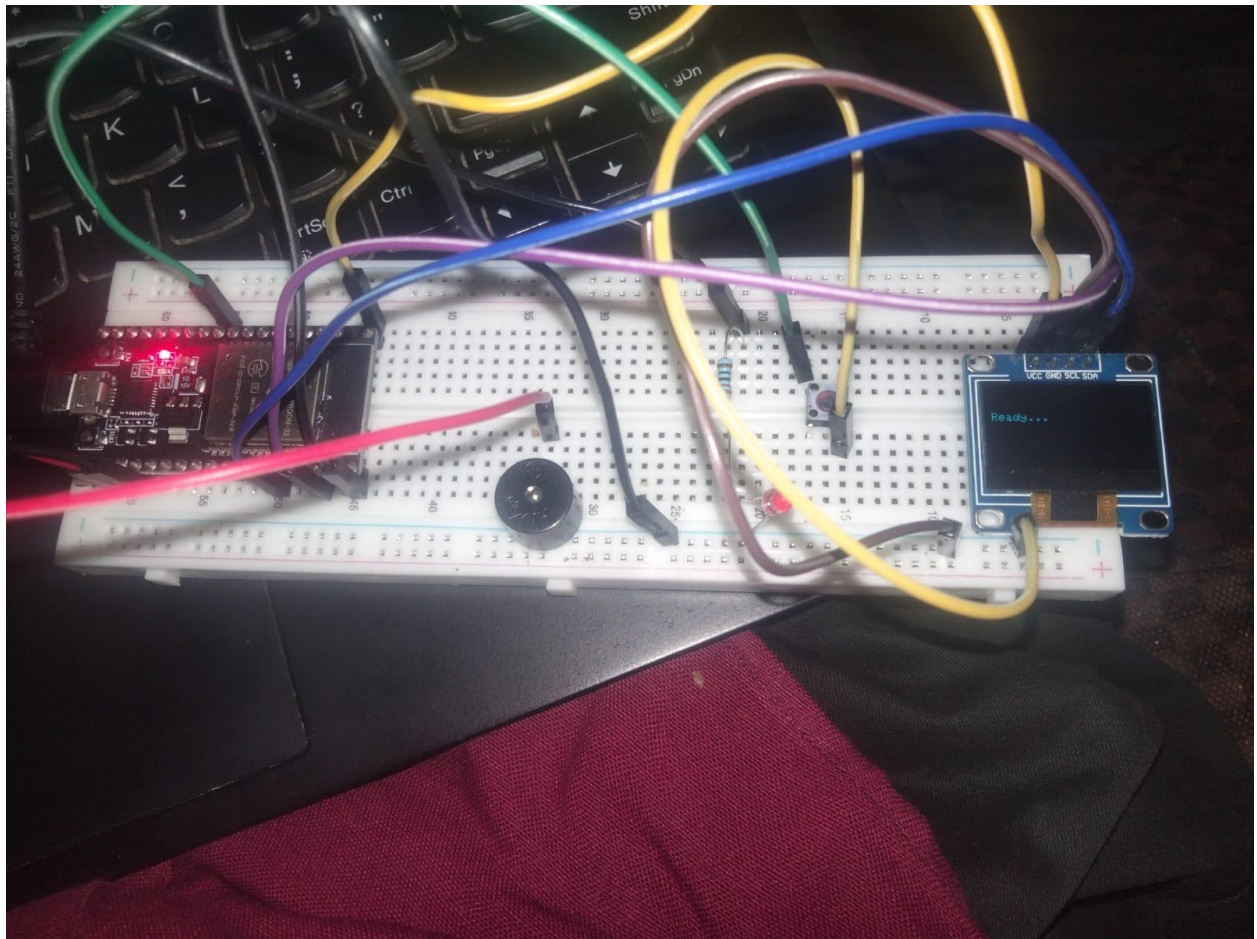
To understand how to measure input duration using `millis()`, differentiate user input types, and provide feedback through both **LEDs**, **buzzer**, and **display**.

Picture of Kits

Task_01



Task_02



Handwritten Code

Task_01

Date: / 20

Embedded IoT sys.

Assignment #1

Task #1 code:

/* Project: LED Mode controller with OLED
and buzzer

Name: Umar Mushtaq

Reg No: 23-NTU-CS-1077

*/

```
#include <Arduino.h>
```

```
#include <Wire.h>
```

```
#include <Adafruit_GFX.h>
```

```
#include <Adafruit_SSD1306>
```

```
// --- Pin definitions ---
```

```
#define LED1 2
```

```
#define LED2 4
```

```
#define LED3 5
```

```
#define BTN_MODE 26
```

```
#define BTN_RESET 27
```

```
#define BUZZER 15
```

Date: 1 / 20

// --- object Display on OLED ---

Adafruit_SSD1306 display(128, 64, 8wire, -1)

// --- variables for mode control ---

int mode = 0;

unsigned long PreMillis = 0;

bool ledstate = false;

// --- Function: show message on OLED

void showmsg(string msg) {

display.clearDisplay();

display.setText(1);

display.setTextColor(white);

display.setCursor(0, 20);

display.print("Mode: ");

display.println(msg);

display.display();

}

// --- function beep sound ---

void beepBuzzer(int freq, int dur) {

tone(Buzzer, freq, dur);

delay(dur + 50);

noTone(Buzzer); }

Date: / 20

```
void setup() {
```

```
    pinMode(LED1, output)
```

```
    pinMode(LED2, output)
```

```
    pinMode(LED3, output)
```

```
    pinMode(BTN_Mode, output)
```

```
    pinMode(BTN_RESET, output)
```

```
    pinMode(Buzzer, output).
```

```
    display.begin(SS1306_SWITCHCAPVCC, 0x3C)
```

```
    display.clearDisplay();
```

```
    display.display();
```

```
    showmsg("Both off")
```

```
    void loop() {
```

```
        if (digitalRead(BTN_Mode) == LOW) {
```

```
            delay(200);
```

```
            mode++;
```

```
            if (mode > 4) mode = 1)
```

```
        switch (mode) {
```

```
            case 1:
```

```
                digitalWrite(LED1, LOW);
```

```
                digitalWrite(LED2, LOW);
```

```
                showmsg("Both OFF");
```

```
                beepBuzzer(800, 120);
```


break

case 2;

showMsg("Alternate Blink");

beep Buzzer(1000, 120);

break;

case 3;

digitalWrite(LED1, HIGH)

digitalWrite(LED2, HIGH)

showMsg("Both ON");

beep Buzzer(1200, 120);

break;

case 4;

showMsg("PWM Fade");

Beep Buzzer(1500, 120);

break;

}

}

if (digitalRead(BTN_RESET) == LOW)

delay(200);

mode = 1;

digitalWrite(LED1, LOW);

digitalWrite(LED2, LOW);

analogWrite(LED3, 0);

Date: / 20

```
void setup() {
```

```
    pinMode(BTN, INPUT_PULLUP);
```

```
    pinMode(LED, OUTPUT);
```

```
    pinMode(Buzzer, OUTPUT)
```

```
    display.begin(SSD1306_SWITCHCAPVCC, 0x3c)
```

```
    showText("Ready..."); }
```

```
void loop() {
```

```
    if (digitalRead(BTN) == LOW && !pressed) {
```

```
        pressed = true;
```

```
        presstime = millis(); }
```

```
    if (digitalRead(BTN) == HIGH && pressed) {
```

```
        unsigned long duration = millis() - presstime;
```

```
        pressed = false;
```

```
        if (duration > 1500) {
```

```
            tone(Buzzer, 1000, 500);
```

```
            showText("long press -> Buzzer");
```

```
        } else
```

```
            ledState = !ledState
```

```
            digitalWrite(LED, ledState);
```

```
            showText("short press -> LED  
toggle");
```

```
        }
```

```
    }
```


Task_02

Date: / / 20

Task #2 code

```
#include <Arduino.h>
```

```
#include <Wire.h>
```

```
#include <Adafruit_GFX.h>
```

```
#include <Adafruit_SSD1306.h>
```

```
#define BTN 27
```

```
#define LED 2
```

```
#define Buzzer 15
```

```
Adafruit_SSD1306 display(128,64,BW16,-1);
```

```
bool ledState = false;
```

```
unsigned long pressTime = 0;
```

```
bool pressed = false;
```

```
void showText(String msg) {
```

```
    display.clearDisplay();
```

```
    display.setTextSize(1);
```

```
    display.setColor(white);
```

```
    display.setCursor(0,20);
```

```
    display.println(msg);
```

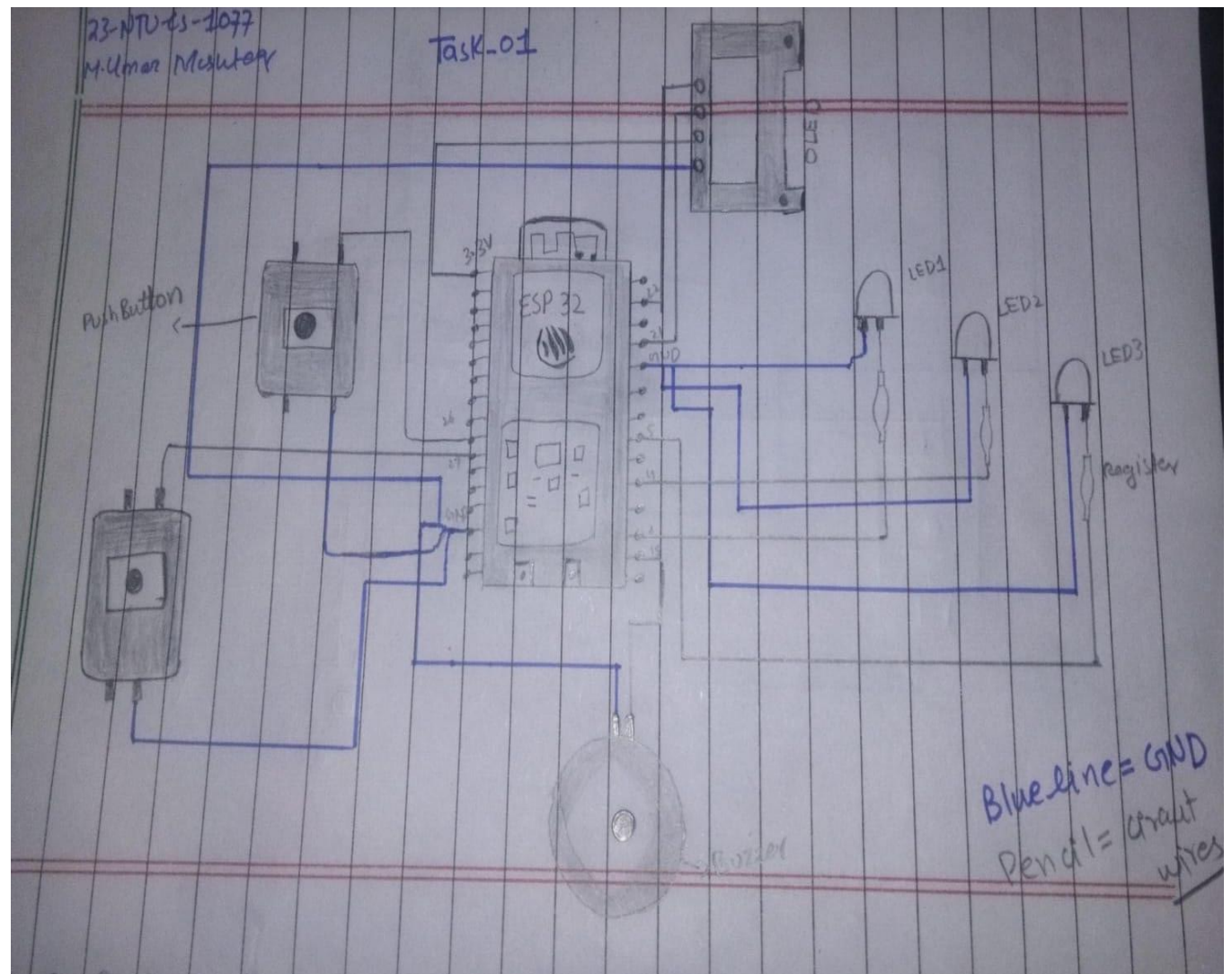
```
    display.display(); }
```

Date: / 20

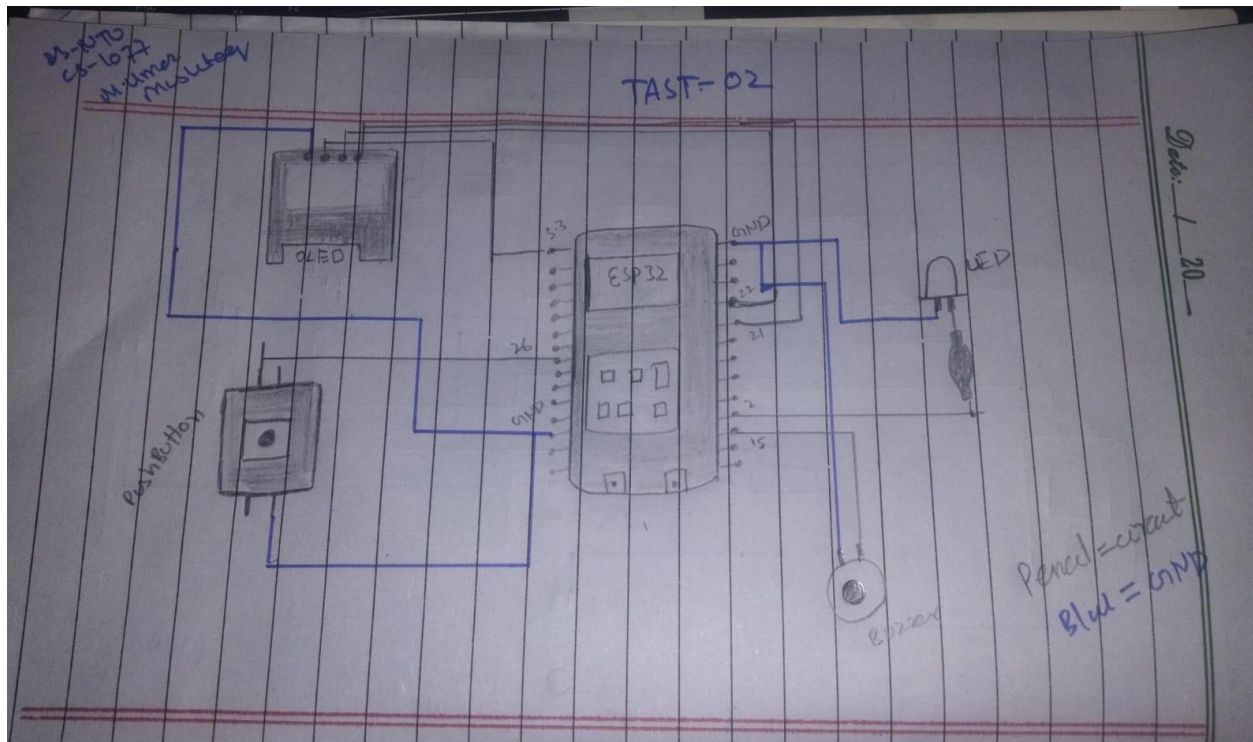
```
showMsg("Reset to OFF");  
keepBuzzer(400, 200); }  
if (mode == 2) {  
    if (millis() - preMillis >= 500) {  
        preMillis = millis();  
        ledstate = !ledstate;  
        digitalWrite(LED1, ledstate);  
        digitalWrite(LED2, !ledstate); }  
}  
if (mode == 4) {  
    for (int i = 0; i <= 255; i++) {  
        analogWrite(LED3, i);  
        delay(15);  
    }  
    for (int i = 0; i <= 255; i--) {  
        analogWrite(LED3, i);  
        delay(15);  
    }  
}  
}
```


Handmade Diagram

Task_01



Task_02



Wokwi Link

Task_01

<https://wokwi.com/projects/445223337931397121>

Task_02

<https://wokwi.com/projects/445224971914989569>

Github Repo Link

<https://github.com/noniiiiiiiiiiiiiiii/1077-embedded-iot-system-cs-b.git>