Project: Button Press Detection (Short/Long Press)

The system distinguishes between short and long button presses, triggering different outputs: a short press toggles an LED, while a long press (>1.5 seconds) activates a buzzer. The system status is displayed on an OLED screen

---

Working of the System

The system is designed to detect button presses and perform actions based on the duration of the press. Here's a step-by-step explanation of how it works:

1. **Initialization (Setup)**:

 - The ESP32 initializes the button (pin 34) as an input with an internal pull-up resistor, the LED (pin 18) as an output, and the buzzer (pin 26) as an output.

 - The OLED display (SSD1306, 128x64 pixels) is initialized over I2C with address `0x3C` and displays "System Ready" on startup.

2. **Button Press Detection**:

 - The button is active-low (pressed = LOW, released = HIGH due to the pull-up resistor).

 - When the button is pressed (`LOW`), the system records the start time using `millis()` and sets a flag (`buttonPressed`) to track the press.

3. **Button Release Handling**:

 - When the button is released (`HIGH`), the system calculates the press duration by subtracting the start time from the current `millis()` value.

 - **Short Press** (<1.5 seconds): Toggles the LED state (ON to OFF or OFF to ON) and displays "Short Press: LED Toggled" on the OLED.

- **Long Press** (>1.5 seconds): Activates the buzzer at 1000 Hz for 0.5 seconds and displays "Long Press: Buzzer ON" on the OLED.

4. **Display and Feedback**:

- The OLED screen updates with relevant messages for each action (system ready, short press, or long press).

- The LED and buzzer provide visual and audible feedback based on the button press type.

5. **Continuous Monitoring**:

- The system continuously monitors the button state in the `loop()` function, ensuring real-time detection of presses without debouncing issues (handled implicitly by the state-based logic).

Pin Mapping

The project uses an ESP32 DevKit-C V4 board connected to various components. Below is the pin mapping based on the provided Wokwi schematic and code:

- **Pushbutton (BTN)** → GPIO 34: Green pushbutton, input with pull-up resistor, active-low (LOW when pressed).

- **LED (led1)** → GPIO 18: Red LED with 1kΩ resistor, output, toggles ON/OFF on short press.

- **Buzzer (bz1)** → GPIO 26: Buzzer, output, plays 1000 Hz tone for 0.5s on long press.

- **OLED Display (SSD1306)**:

  - SDA → GPIO 21: I2C data line for display.

  - SCL → GPIO 22: I2C clock line for display.

  - VCC → 3V3: 3.3V power for display.

  - GND → GND: Ground for display.

- **Ground Connections** → GND: Shared ground for LED, buzzer, button, and OLED

- **Libraries Used**:

  - `Arduino.h`: Core Arduino functions.

  - `Wire.h`: For I2C communication with the OLED.

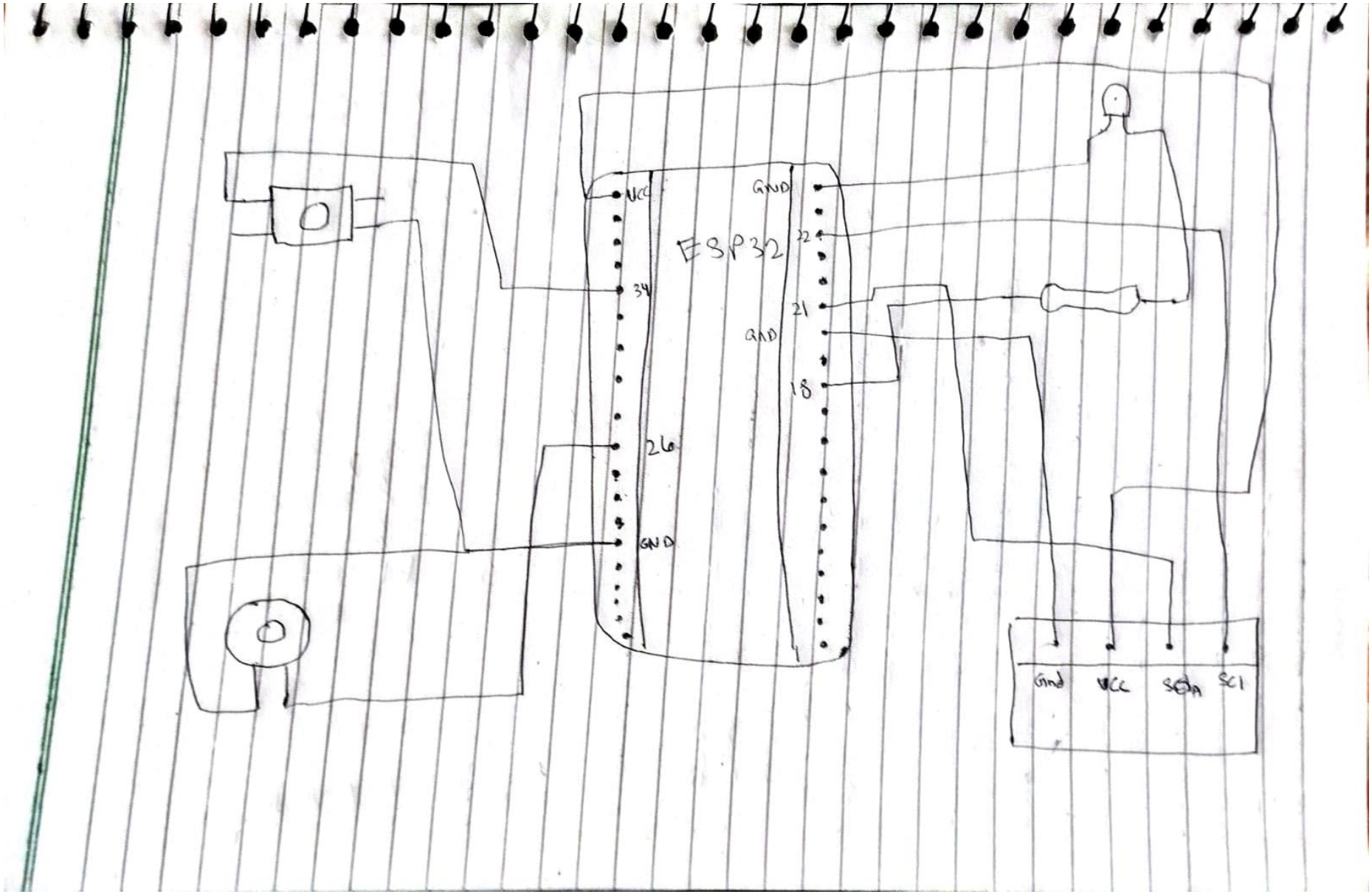
  - `Adafruit_GFX.h` and `Adafruit_SSD1306.h`: For controlling the OLED display.

-

Project Button Press detection (short / long Press)

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

#define BTN 34
#define LED 18
#define Buzzer 26

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

Bool ledon = false;
bool buttonPressed = false;
unsigned long pressstart = 0;

void showtext ( string text ) {
    display.clearDisplay ();
    display.setTextsize (1);
    display.setTextcolor (White);
    display.setcursor ( 0, 20);
    display.println (text)
    display.display ();
}

void setup () {

    PinMode ( BTN, INPUT_PULLUP);
    pinMode ( LED, Output);
    PinMode ( Buzzer, Output);
```

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
showText("system ready");

void loop(){

  int buttonState = digitalRead(BTN);

  if (buttonState == LOW && !buttonPressed){

      buttonPressed = true;
      pressStart = millis();
    }

  if (buttonState == HIGH && buttonPressed){
      buttonPressed = false;
      unsigned long pressTime = millis() - pressStart;

  if (pressTime > 1500){

      tone(BUZZER, 1000, 500);
      showText("long Press Buzzer ON");

  else {

      ledON = !ledON;
      digitalWrite(LED, ledON);
      showText("short Press led toggled");

    }
  }
}
```


VCC
GND
ESP32
22
34
21
GND
18
26
GND
GND
VCC
SCL