

# ESP32 Morse Code trainer

## Documentation

### Summary

This circuit is a Morse Code Trainer based on the ESP32 microcontroller. It includes an OLED display for visual feedback, multiple pushbuttons for user input, a potentiometer for adjusting settings, a buzzer for audio feedback, and an LED for visual signaling. The circuit is designed to help users learn, send, and decode Morse code through a combination of hardware and software interfaces, including a web interface for additional control and feedback.

### Component List

1. **ESP32 (38-pin Type-C CP2102)**
  - Description: A powerful microcontroller with WiFi and Bluetooth capabilities, used as the main processing unit.
  - Pins: 5V, CMD, SD3, SD2, G13, GND, G12, G14, G27, G26, G25, G23, G32, G35, G34, SN, SP, EN, 3V3, CLK, SD0, SD1, G15, G2, G0, G4, G16, G17, G5, G18, G19, G21, RXD, TXD, G22
2. **OLED Display**
  - Description: A small display module used to show text and graphics.
  - Pins: GND, VCC, SCL, SDA
3. **Pushbuttons (5 units)**
  - Description: Momentary switches used for user input.
  - Pins: Pin 1, Pin 2, Pin 3, Pin 4
4. **Potentiometer**
  - Description: A variable resistor used to adjust the speed of Morse code transmission.
  - Pins: GND, Output, VCC
5. **Buzzer**
  - Description: An audio output device used to emit sound for Morse code signals.
  - Pins: +, -
6. **LED (Two Pin, Green)**
  - Description: A light-emitting diode used for visual signaling of Morse code.
  - Pins: Cathode, Anode
7. **Comment V2**
  - Description: A placeholder for comments or notes in the circuit design.
  - Pins: None

### Wiring Details

#### ESP32 (38-pin Type-C CP2102)

- **CMD:** Connected to OLED GND, Pushbutton Pin 1 (all), Potentiometer GND, Buzzer -, LED Cathode
- **5V:** Connected to Potentiometer VCC
- **G21:** SDA oled
- **G22:** SCL oled
- **G32:** Connected to Pushbutton Pin 4 (first button)
- **G35:** Connected to Pushbutton Pin 4 (second button)
- **G25:** Connected to Pushbutton Pin 4 (third button)
- **G26:** Connected to Pushbutton Pin 4 (fourth button)
- **G27:** Connected to Pushbutton Pin 4 (fifth button)
- **G34:** Connected to Potentiometer Output
- **G19:** Connected to Buzzer +
- **G23:** Connected to LED Anode

## OLED Display

- **GND:** Connected to ESP32 CMD
- **VCC:** Not connected
- **SCL:** Not connected
- **SDA:** Not connected

## Pushbuttons

- **Pin 1:** Connected to ESP32 CMD
- **Pin 4:** Connected to respective ESP32 GPIO pins (G32, G35, G25, G26, G27)

## Potentiometer

- **GND:** Connected to ESP32 CMD
- **Output:** Connected to ESP32 G34
- **VCC:** Connected to ESP32 5V

## Buzzer

- **-:** Connected to ESP32 CMD
- **+:** Connected to ESP32 G19

## LED (Two Pin, Green)

- **Cathode:** Connected to ESP32 CMD
- **Anode:** Connected to ESP32 G23

## Code Documentation

The code for this circuit is written in C++ and is designed to run on the ESP32 microcontroller. It includes the following key features:

- **OLED Display Initialization:** The OLED display is initialized using the Adafruit\_SSD1306

library, and it is used to display various UI elements and messages.

- **Button Handling:** The code includes functions to read the state of the pushbuttons and handle user input for navigating menus and entering Morse code.
- **Morse Code Transmission:** The code can encode text into Morse code and control the buzzer and LED to transmit the encoded signals.
- **WiFi and Web Server:** The ESP32 is configured as a WiFi access point, and a web server is set up to provide a web interface for sending messages and adjusting settings.
- **Settings and State Management:** The code manages various settings such as words per minute (WPM) for Morse code transmission, and it maintains the current state of the UI and transmission.
- **Practice Mode:** A practice mode is implemented to help users learn Morse code by providing feedback on their input.

The code is structured into several sections, including setup, main loop, and various helper functions for handling specific tasks. The use of libraries such as WiFi, WebServer, Wire, and Adafruit\_GFX simplifies the implementation of complex features like networking and graphics.