Table of Contents

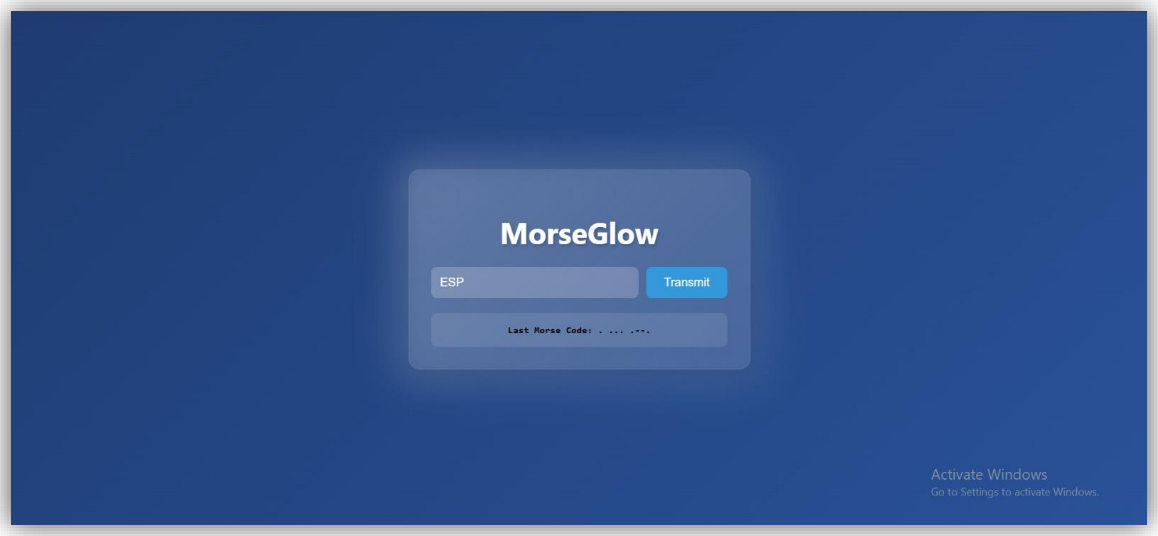
Abstract

This project report presents an innovative IoT application titled **MorseGlow,** developed using the ESP32 microcontroller. The project allows text input via a web interface and converts it to **Morse Code**, which is displayed on an **OLED screen** and represented visually through a **NeoPixel LED**. The system also provides Wi-Fi connectivity through **STA and AP modes**, making it accessible both through an external network and direct device connection.

1. Introduction

The ESP32 microcontroller is a versatile platform for developing IoT-based projects due to its integrated wireless communication and peripheral support. This project, MorseGlow, demonstrates the ESP32's ability to convert text messages into Morse Code, displayed via an OLED screen and visualized using a **NeoPixel LED**.

The application aims to enhance educational tools and creative projects where visual representation of text messages is desired. Additionally, the use of both STA and AP modes ensures flexibility in device connectivity.



```
Last Morse Code: . ... .--.
```

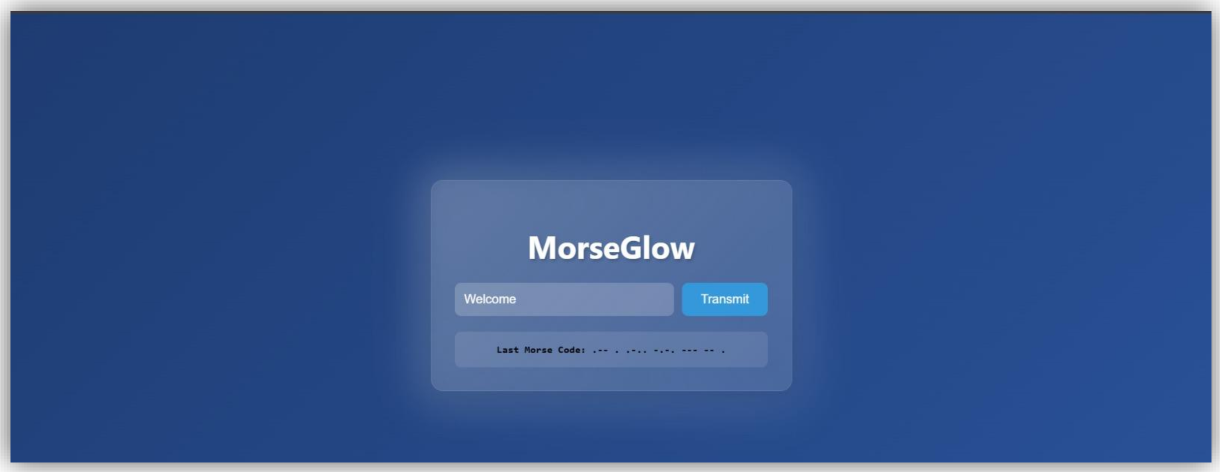
2. System Architecture

The MorseGlow System is composed of the following components:

- ESP32 Microcontroller
- NeoPixel LED(RGB LED)
- SSD1306 OLED Display
- Wi-Fi Connectivity (STA & AP modes)
- Socket Programming for Web Serve

2.1 Network Configuration

- STA Mode: Connects to a local Wi-Fi network for internet access.
- AP Mode: Provides a local access point for direct connection to the ESP32.



```
Last Morse Code: .-- . .-.. -.-. --- -- .
```

2.2 Hardware Integration

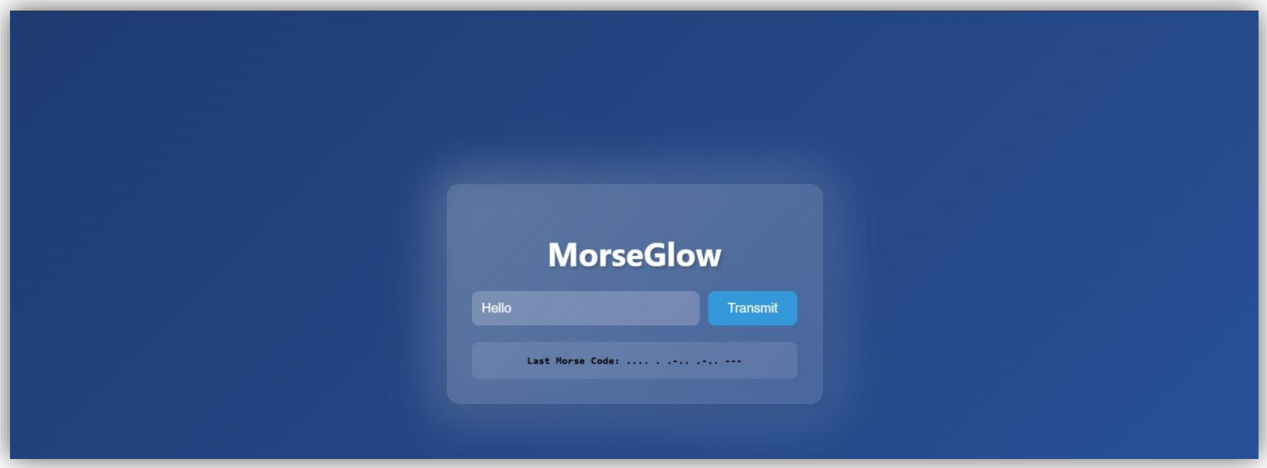
- NeoPixel Control: Uses color coding for dots (red) and dashes (blue) in Morse Code.
- OLED Display: Displays the original text and its corresponding Morse code.

3. Implementation Details

The implementation involves three major aspects:

- ***3.1 Web Server Development***
  A socket-based HTTP server is hosted on the ESP32, enabling communication with clients over the network. The server listens on port 80 and serves a simple HTML interface for text input. The server processes the request, converts the text to Morse code, and displays the result via OLED and NeoPixel.

- ***3.2 Morse Code Generation***
  The text input from the user is converted to Morse Code using a predefined dictionary of Morse symbols. The Morse code is displayed on the OLED and visualized using the NeoPixel LED.



```
Last Morse Code: .... . .-.. .-.. ---
```

- ***3.3 UI Design***

  The web interface is designed to provide a clean and intuitive user experience. It features a text box for input and displays the last transmitted Morse code message.

4. Results & Discussion

The MorseGlow System successfully integrates various components, allowing:

- Real-time text-to-Morse code conversion.

- Clear visual representation via OLED and NeoPixel LED.

- Seamless communication using the ESP32's Wi-Fi capabilities.

- ***4.1 Technical Challenges***
    - Synchronizing visual representation between OLED and NeoPixel.
    - Ensuring compatibility with multiple devices through AP and STA modes.
- ***4.2 Solutions***
    - Efficient message handling techniques to prevent data loss.
    - Error handling mechanisms for improved stability.

5. Conclusion

The MorseGlow System demonstrates the ESP32's potential to handle complex tasks involving user interaction, data processing, and visual representation. The system can be further expanded to include audio feedback and wireless data logging.

6. References

- Socket Programming Fundamentals

- W3Schools for Web Page Design

- ESP32 Documentation