Design and Development of a Modular Side-Illuminated Display

ab4647 lh2640 ss5250 sjk81
Bath University

April 2025

Abstract

This report outlines the design, implementation, and testing of a modular side-illuminated display using laser-cut acrylic sheets and an Arduino-controlled LED system. Inspired by Lixie displays, the project involved PCB design, embedded programming, and mechanical assembly. The final product displays weather data by retrieving real-time conditions from the OpenWeatherMap API [1], mapped onto etched symbols on acrylic sheets. This project showcases cross-disciplinary integration between electronics, CAD, and sustainability in a highly visual and data-driven product.

Contents

1 Introduction

The resurgence of interest in retro-futuristic displays, such as Nixie tubes, has inspired engineers to develop modern alternatives like Lixie displays. These use LED lighting and laser-etched acrylic sheets to recreate a similar aesthetic in a cost-effective and sustainable manner. This project was developed during Project Week and focused on the creation of a modular, side-illuminated display that visually communicates real-time weather data using custom PCB design, mechanical assembly, and embedded programming.

Acronyms Used

API	Application Programming Interface
CAD	Computer-Aided Design
ESP32	Embedded Serial Peripheral 32-bit Microcontroller
LED	Light Emitting Diode
PCB	Printed Circuit Board
WiFi	Wireless Fidelity

2 Methodology

2.1 System Design Overview

The modular display consists of six laser-etched acrylic sheets representing different weather symbols. Each sheet is side-lit by a designated LED controlled by an ESP32 microcontroller connected to a custom-designed PCB. The structure is housed in a laser-cut MDF frame that aligns each sheet precisely.

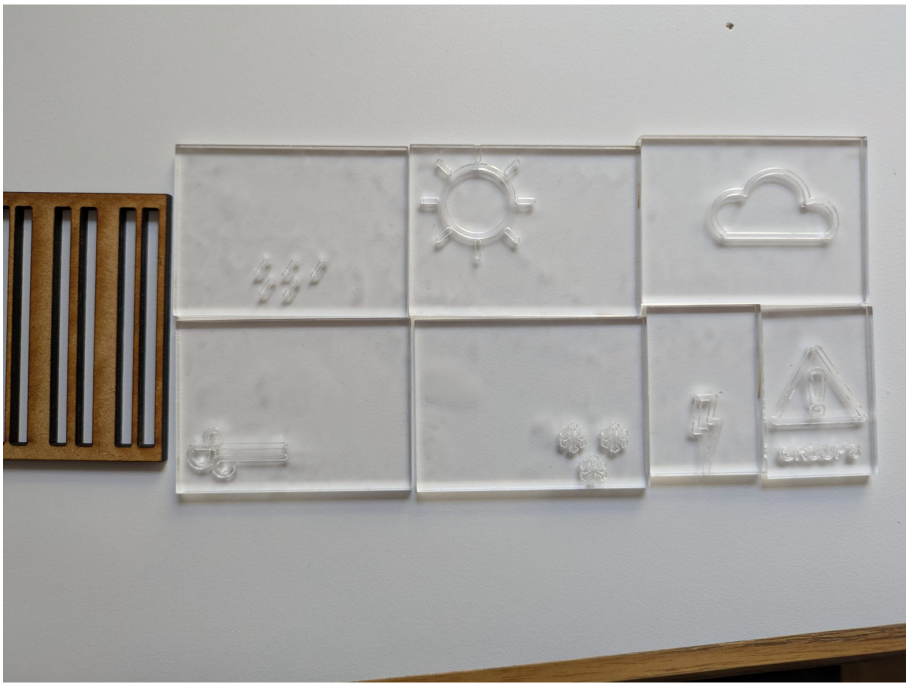

Figure 1: Laser-cut acrylic sheets with etched weather icons including sun, cloud, rain, wind, snow, lightning, and hazard symbols.

2.2 Hardware Components

- ESP32 development board with built-in WiFi [2]
- Custom two-layer PCB designed with KiCAD [3]

- Six side-illuminating LEDs
- Laser-cut 3mm acrylic sheets
- MDF frame for mechanical alignment
- Arduino IDE for embedded programming

2.3 Circuit and PCB Design

The PCB was created to route power and control signals to each LED. Design checks were performed using KiCAD's DRC and Gerber preview tools before manufacturing [3]. Once the board was received, components were soldered, and continuity tests confirmed functional connections.

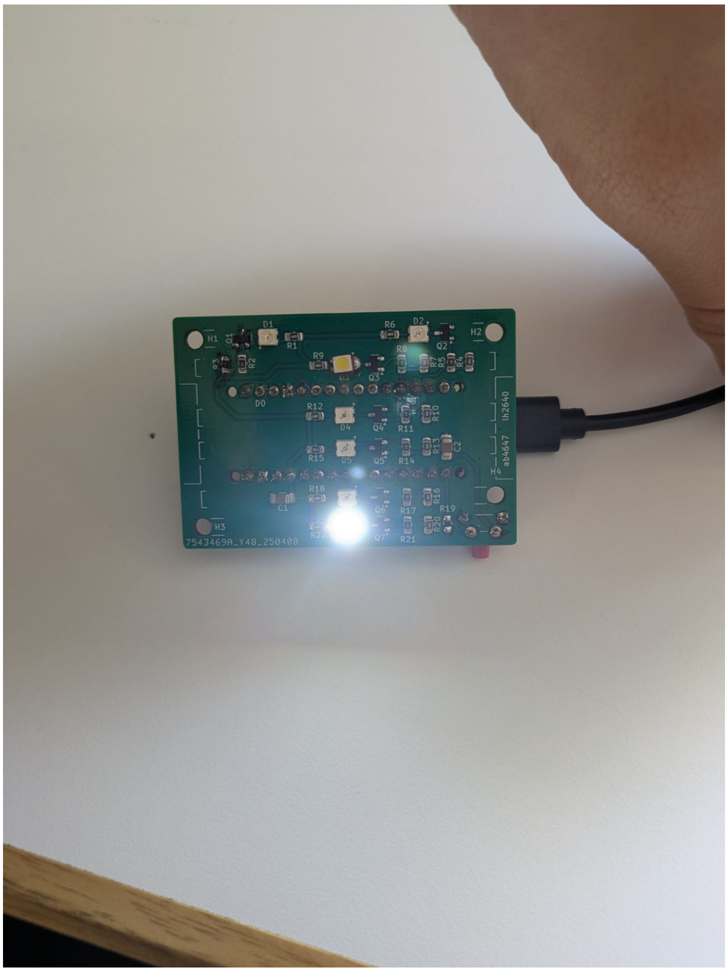

Figure 2: Powered PCB with an illuminated LED, demonstrating functionality and confirming correct assembly.

2.4 Software and Firmware Implementation

The ESP32 connects to the Bath-IoT network and periodically fetches weather data from OpenWeatherMap using HTTP requests [1]. Data is parsed using the Arduino_JSON library, and weather conditions are mapped to the corresponding LED. The WiFi library from Arduino was used to manage network communications [?].

1. Connect to WiFi
2. Fetch and parse weather data
3. Identify primary condition (e.g. Rain, Clear, Clouds)
4. Illuminate corresponding LED

3 Results and Analysis

3.1 System Performance

The system performed reliably, with LEDs responding correctly to parsed weather conditions. The modular nature allowed fast sheet replacement, enabling customization or extension with new symbols.

Figure 3: Fully assembled working prototype with symbol overlay illuminated, representing combined weather states.

3.2 Verification and Validation

- All acrylic sheets fit within tolerance (±0.2 mm)
- LED alignment achieved via test jigs and visual inspection
- API data correctly mapped and parsed
- WiFi connectivity established within 5 seconds on average

4 Conclusion

This modular display project successfully integrates disciplines across electronics, mechanical design, and networked systems. It not only mimics vintage aesthetic appeal but delivers real-time functionality for educational and prototyping use cases. The modularity supports future expansion into more detailed weather states or alphanumeric displays.

5 Sustainability and Future Improvements

Future iterations of this project can significantly enhance interactivity, energy efficiency, and scalability:

- **Button-Activated Modes:** Introduce input via push buttons to allow users to cycle through LED modes, or manually override weather inputs for display testing or fun visual effects.
- **Energy Saving Features:** Implement auto-off modes and wake-up on button press. Use Arduino's low-power modes to conserve energy when idle [5].
- **Task Scheduling:** Incorporate the TaskScheduler library to handle timed events like API polling, button handling, and sleep modes without blocking other processes [6].
- **Arduino IoT Cloud Integration:** Add remote access through the Arduino IoT Cloud to view and control LED states, update settings remotely, and visualize historic weather data [5].
- **Modular Expansion:** Allow multiple units to daisy-chain for larger displays or more symbols. Introduce support for textual data or animated transitions.

References

[1] OpenWeatherMap API Documentation, Available: `https://openweathermap.org/api`

[2] Arduino ESP32 WiFi Library Reference, Available: `https://www.arduino.cc/en/Reference/WiFi`

[3] KiCAD Documentation, Available: `https://docs.kicad.org/`

[4] MAX9814 Datasheet, Available: `https://cdn.sparkfun.com/datasheets/BreakoutBoards/MAX9814.pdf`

[5] Arduino IoT Cloud Documentation, Available: `https://docs.arduino.cc/cloud/iot-cloud/`

[6] TaskScheduler GitHub Repository, Available: `https://github.com/arkhipenko/TaskScheduler`