QR Code Scanner and ESP32-CAM Project Report

Submitted for: Electronic Design Workshop (ECECC404)

April 13, 2025

Contents

Chapter 1

The Idea

The original concept for this project was to design and implement a QR code scanner. However, as the project progressed, additional features were envisioned and integrated, including the ability to capture photos and toggle a flashlight on and off.

At the heart of our project lies the ESP32-CAM module, a compact yet powerful microcontroller with an integrated camera. This module is responsible for capturing frames, processing them to detect QR codes, and relaying data to a server if a QR code is detected.

Functional Overview

- **QR Detection:** The ESP32-CAM runs a detection algorithm which continuously analyzes incoming frames. If a QR code is found, it forwards the relevant frame to a server for decoding.
- **Server-Side Decoding:** The server, running custom code, takes the frame and attempts to decode the QR code using standard libraries. Once decoded, the extracted data is sent to the user.
- **Telegram Bot Integration:** The server communicates with a Telegram bot that sends the decoded data to the user's Telegram account. This makes the setup highly portable and easy to access.
- **Modes of Operation:** The system can be toggled between different modes — photo capture mode and QR scan mode. In the photo mode, users can capture and receive photos directly via Telegram.

This project showcases how low-cost, off-the-shelf microcontrollers can be leveraged to implement real-time computer vision tasks with additional cloud-based functionalities.

Chapter 2

Hardware Implementation

To power and support the ESP32-CAM, we designed a complete power delivery system comprising three main modules: an adapter module, a charging module, and a booster module. Each module is crucial for ensuring stability, portability, and efficient performance.

2.1 Adapter Module

The adapter is a basic AC to DC power supply circuit that converts 220V AC to a regulated 5V DC.

- A step-down transformer converts 220V AC to 9-12V AC.
- A Full-Wave Bridge Rectifier (FWBR) converts this to DC.
- An LM7805 voltage regulator regulates it to 5V at 1A.
- Initial filtering used a 0.33µF capacitor, but was later upgraded to 2200µF due to ripple issues and voltage dips under load.

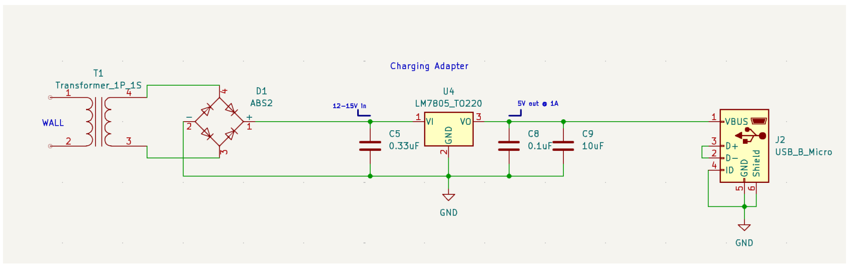

Figure 2.1: Schematic of the Adapter Module

2.2 Charging Module

The charging module allows the system to operate wirelessly with a LiPo battery.

- A USB-C connector feeds power into the module.
- The MCP738311/2 LiPo charging IC handles battery charging at a max of 500mA.

- PMOS switching circuitry automatically disables battery discharge during external power input.
- Once unplugged, battery resumes supplying power.

2.3 Booster Module

To drive the ESP32-CAM at stable voltage levels, we use the MT3608 booster module.

- It boosts the 3.7V LiPo battery to 5.1V with max output current of 1.5A.
- The battery used is rated at 1000mAh with a discharge capacity of 2C.
- This makes the module portable and supports moderate current draw without brownouts.

The system is designed to switch seamlessly between charging and discharging modes, ensuring stable operation throughout usage cycles.

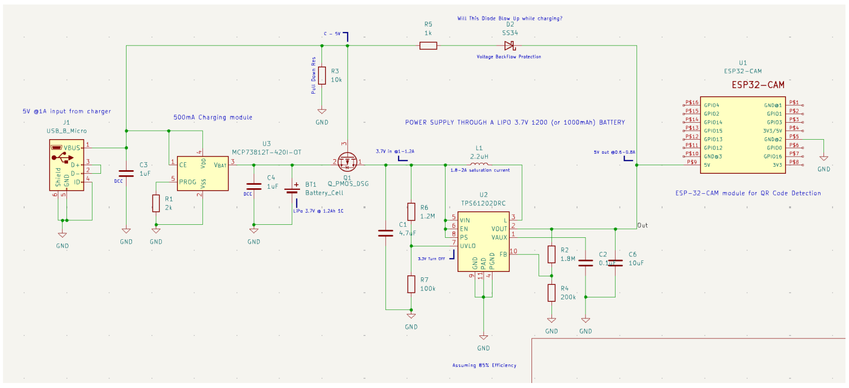

Figure 2.2: Schematic of the Multiplexed Charger and Booster Module

2.4 ESP32 Cam Scanner Modue

This is the Last module which uses the ESP32-CAM to Detect QR codes and take photos and send it to the user!

- Camera Module - **MODULE NAME**
- Supply Pin Used was 5V one, as evident from the above module.
- The ESP Send the image and QR data to the user via a telegram bot for ease of access and minimal setup!

Chapter 3

Challenges and Issues Faced

3.1 Adapter Module Power Constraints

We aimed to extract 5W of power at 5V (1A) using the LM7805 regulator. However, the regulator's inefficiency meant we needed 10W input power. At 7V input, this translated to a current draw of 1.42A, which practically shorted the Input of the 7805. The input resistance appeared too low, leading to significant ripple and voltage drops.

Solution: We increased the input capacitor from 0.33µF to a much larger 2200µF, which effectively dampened voltage ripple and ensured smoother regulation.

3.2 Component Unavailability

Despite extensive searching across local markets in Delhi, we encountered repeated unavailability of key components:

- **TPS61202 IC** (intended for boosting): Not found. Replaced with an MT3608 breakout.
- **MCP73831/2 Charger IC**: Eventually ordered online after many failed attempts to find it in stores.

3.3 ESP32 Flashing Issues

The ESP32 initially refused to flash firmware. After investigation, we realized the user needed permissions to access USB devices. Adding the user to the correct system group ('dialout' or similar on Linux) resolved the problem.

3.4 QR Code Decoding Errors

While the ESP32 detected QR codes reliably, the actual decoding sometimes failed due to ECC (Error Correction Code) issues. Despite tweaking buffer sizes and memory allocations (using PSRAM), results were inconsistent.

Workaround: We moved decoding to the server. This improved performance and also allowed decoding of stylized QR codes — a feature that wouldn't be feasible on the ESP itself.

Chapter 4

Remaining Work and Planning Ahead

4.1 Charger Module Completion

Although the charger circuit is designed and the MCP73831 IC is ordered, delivery delays mean we might not be able to complete this part before the final deadline. We plan to discuss this with our faculty and request appropriate accommodations.

4.2 Enclosure Design

The plan was to make the system wearable — possibly like smart glasses. However, due to uncertainty in circuit dimensions and size, we may pivot to a more conventional box-style enclosure.

New Plan:

- Compact rectangular enclosure with buttons and ports exposed.
- Increased safety by keeping LiPo battery enclosed and away from the user's face.
- Easier to prototype and manufacture quickly using 3D printing or acrylic fabrication.

Chapter 5

Conclusion

The QR Code Scanner project was not only a learning opportunity but also a creative engineering endeavor. It encompassed embedded systems, electronics hardware design, real-time computer vision, and web-server integration — making it a well-rounded interdisciplinary project.

Key takeaways include:

- Understanding the real-world limitations of voltage regulators and power electronics.
- Working around hardware availability constraints by creatively adapting modules.
- Integrating a full-stack system: from camera sensing on an ESP32, to server-side decoding, to real-time messaging via Telegram.
- Dealing with bugs, flashing issues, and decoding limitations — and learning to design around those constraints.

While the project is not perfect, it has shown us the value of adaptability and the power of embedded computing when paired with cloud integration.

We hope that with a few final tweaks — especially to the charger and the enclosure — our QR scanner will be a working, demo-ready product. Regardless, the process has already taught us far more than a simple classroom assignment ever could.