IOT-Barcode checker

Project report

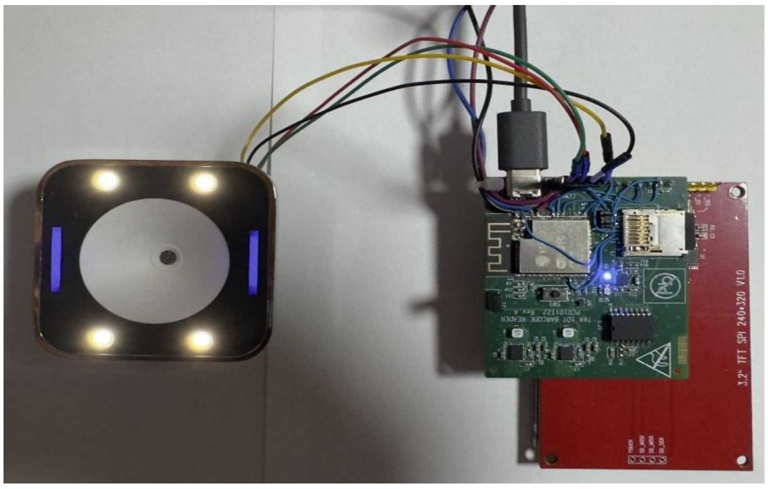
Students:

Muhammad Biadsy

Omar Sharafy

Supervised By:

Mony Orbach



Spring 2024

Contents

1. Overview

This project is a collaboration with Bnai Zion Medical Center aimed at addressing a common issue in hospital and medical center laboratories: the mismanagement of patient samples.

Currently, when samples are collected from patients, they are labeled with barcode stickers containing relevant patient information, such as name and ID. Upon arrival at the lab, these samples are checked against the patient's details in the computer system. Typically, staff label each sample with the patient's name/ID and a unique barcode specific to the patient.

However, problems arise when multiple samples from the same or different patients arrive simultaneously. Manual handling can lead to confusion and mix-ups, making it crucial to verify at every stage that the samples belong to the correct patient and are intended for the designated laboratory.

In this project, we aim to design a low-cost, reliable barcode checker device that ensures all samples labeled as belonging to a patient (via name/ID) are correctly labeled with the matching barcode unique to the patient (a.k.a. "Golden Barcode").

2.Introduction

2.1 Device Objective

The goal of this project is to develop a compact and portable device that ensures all samples labeled with a patient’s name/ID have barcodes matching the patient's unique "Golden barcode."

This device will feature a 2D barcode scanner capable of reading the barcodes on the samples, providing immediate feedback on its screen and via sound to indicate whether the barcode on the sample matches the patient’s Golden barcode.

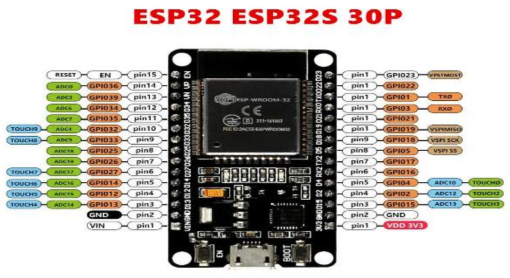
Additionally, the device will connect to a computer via Wi-Fi to display a list of the scanning results, ensuring comprehensive and accessible sample verification.

Designed to be user-friendly, cost-effective, and compatible with existing lab equipment and software, the device will be small, chargeable via USB-Type C & Showing results Via sound and Screen to avoid any confusion.

2.2 Device Components

For the objectives this project aims to achieve, we used a list of components to help with our design.

2.2.1 ESP-32



ESP32 is a series of low cost, low-power system on a chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth.

Cost: 1$

2.2.2 Display ILI9341

3.2-inch display that is used to display

important information about the barcode matching.

Resolution: 320X240 pixels

Comm. Protocol: SPI

Cost: 6.5$

2.2.3 Barcode scanner Grow Gm810

5.6 cm wide barcode scanner uses UART protocol.

Scans the barcode and provide us with important information about it.

Comm. Protocol: UART

Cost: 18$

2.2.4 RTC

A real time clock with high accurce, will be use to recored the time of each scanning and comparing operation and save it in order to allow backtrace.

Cost: 1$

2.2.5 SD Card

A high-accuracy real-time clock will be used to record the time of each scanning and comparison operation, ensuring precise timestamping. This data will be saved to facilitate backtracking and auditing of all activities.

Comm. Protocol: H-SPI.

Cost: 1$

2.2.6 Usb Charger – type C

A USB Type-C charger delivers the necessary power to efficiently charge the battery.

Cost: 0.5$

The overall cost of the components and sensor for the device sums up to about 28$ US Dollars, in addition to other components we will talk about and manufacturing cost, we can sum up to about 45$ US Dollars which is a noticeably a low price in comparison to other devices in the market.

2.3 Development Environments

2.3.1 Arduino IDE

Used to write and upload the code to ESP-32 control board.

2.3.1 OrCad

Used for designing the electric schematic and it’s blocks.

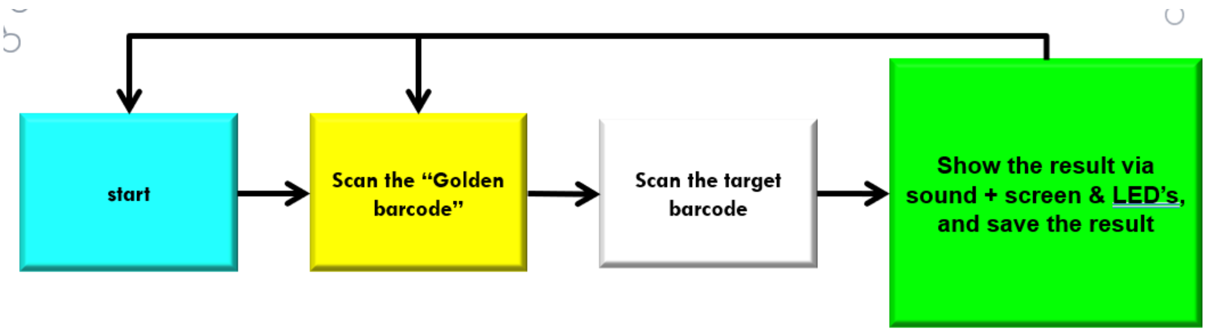
3.Flow Diagram

As described before, the device functionality starts by clicking a button to start the device.

When the user wants to start a series of comparing a pool of barcodes from sample with a "golden barcode", they must press the golden switch and scan the golden barcode.

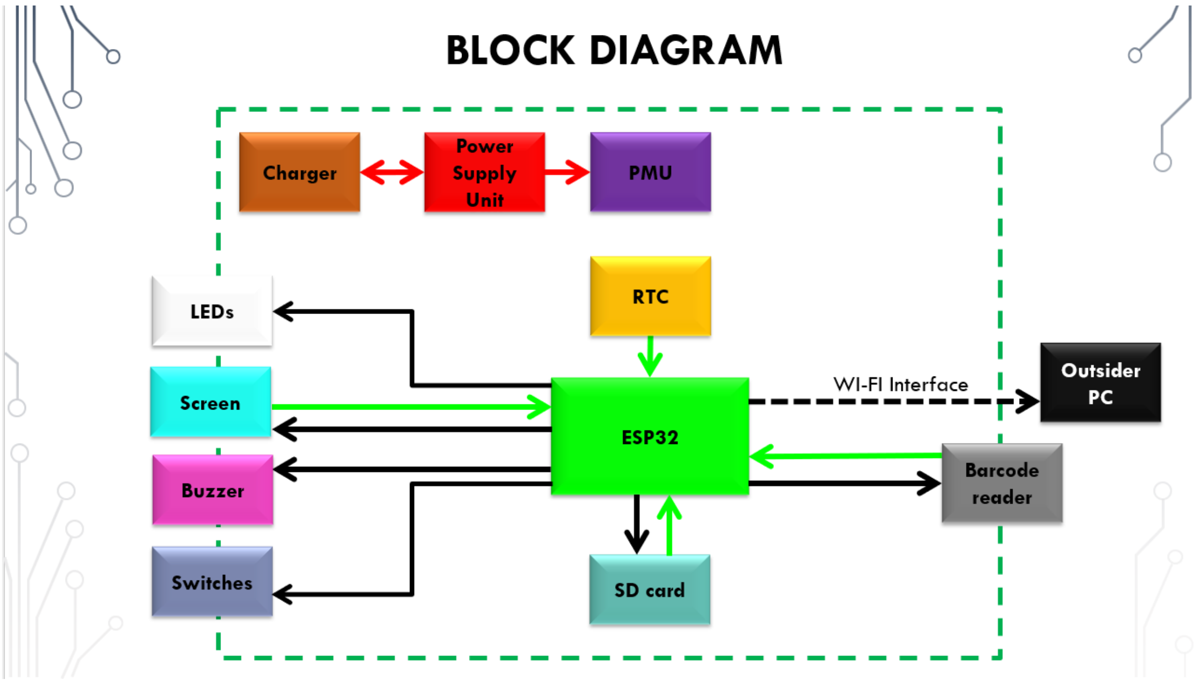
Then they can scan each barcode and get the results via screen and sound and save the result into the SD card.

When the user wants to change the golden barcode – he must press the golden switch again and scan the desired barcode.


start
Scan the "Golden barcode"
Scan the target barcode
Show the result via sound + screen & LED's, and save the result

4.Hardware Block Diagrams

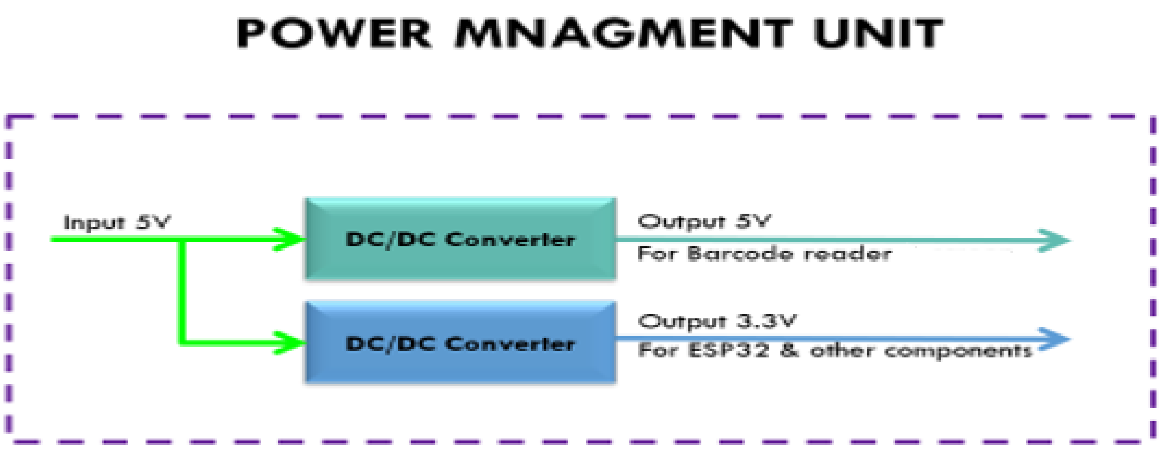
A top-view block diagram of the Bar-code checker components and connection both internally between the sensors and microprocessor, and externally with the outside world.



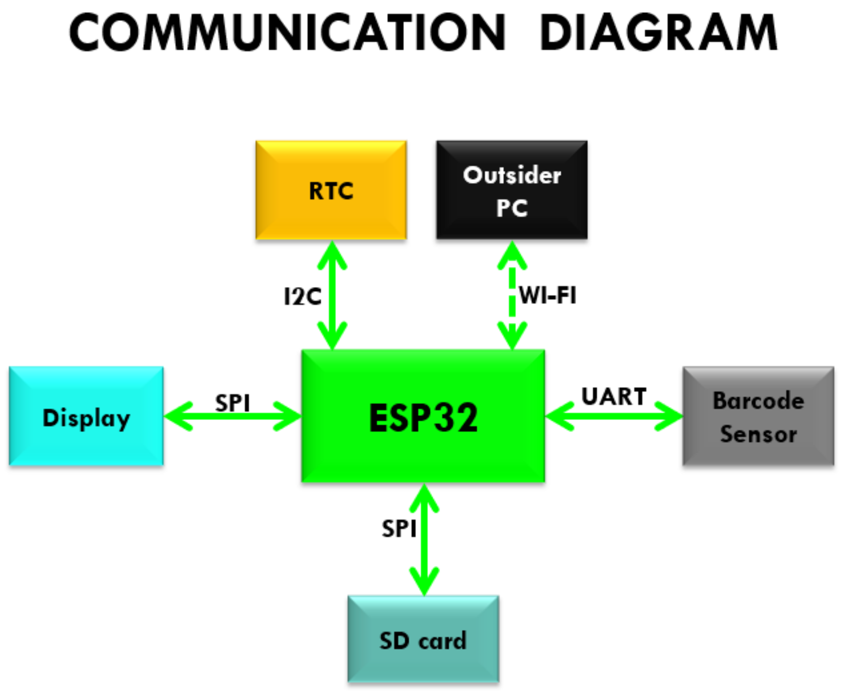
4.1 Power Management Unit

This unit is responsible for supplying the voltages for the different components of the design, It includes USB connection, The USB connection supplies power to the battery charger, which powers the system while simultaneously charging the battery. This feature enables the system to run without a battery or with defect battery.

Note that the Decision on whenever a certain component is working and gets its required voltage is managed in the Control Unit.

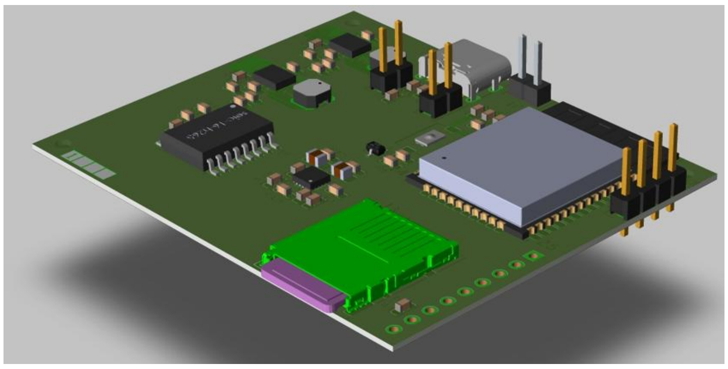


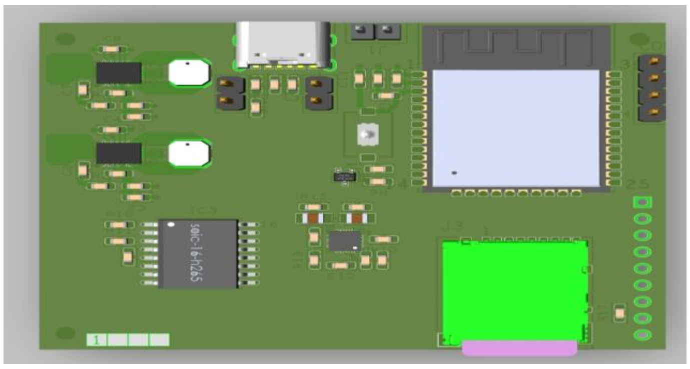
4.2 Communication Map



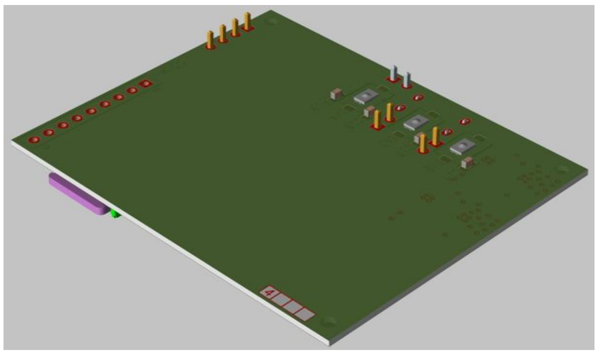
5.Layout

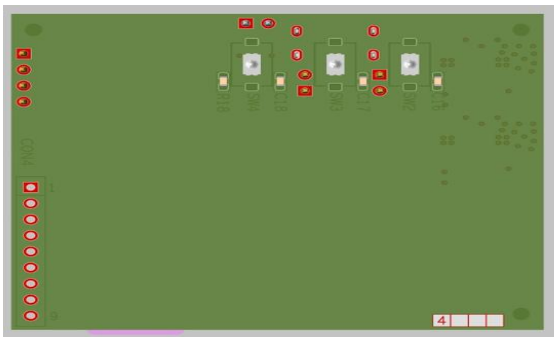
5.1 Top View





5.2 Bottom View





The top of the PCB contains Our ESP32, SD card, two DC to DC converters- TPS63020 and battery charger- BQ24072. It also contains a UART connector to the Barcode Reader.

The bottom of the PCB contains the Connector to the Screen, the LEDs and the 3 switches we will use.

5.3 Considerations taken in layout:

The size of the screen forced us to put it in the opposite size of most other components, to allow us to interact easily with the other components.

Top: the DC-to-DC converters contain inductors. This fact Forced us to place it far enough from the other components.

Then we placed the ESP32, which is the biggest component. The USB and battery connectors are close to the edge, as well as the SD card, so they can be connected easily.

Bottom: we placed the screen connector on the edge, in order to allow the user to interact with the LEDs and switches placed on the bottom.

6. Software

6.1 General description of the main program code

The file "Project_Arduino_code.ino" serves as the main program for the project. Its functionality revolves around continuously comparing scanned target barcodes to a golden barcode. The golden barcode can be updated by pressing a switch, which allows the program to enter the "scan golden barcode" mode. If an incorrect golden barcode is scanned or you wish to stop comparing target barcodes, pressing the switch again will allow you to update the golden barcode.

When switching back to the "scan golden barcode" screen, the display shows the battery percentage and the Wi-Fi Access Point's IP address. The IP address is updated each time a target barcode is scanned, along with the results. When the program starts running you will need to press the switch to scan the first golden barcode as you can see on the start screen.

At the start of the program, press the switch to scan the first golden barcode as prompted on the start screen.

To access the Wi-Fi page, connect a device (pc/phone & etc...) to the **"ESP32-Access-Point"** SSID using the password **"123456789"**. Open a web browser and enter the displayed IP address, which can be found at the bottom of the start screen or the "Scan Golden Barcode" screen.

The program provides visual feedback based on the barcode comparison:

- If the golden and target barcodes match, the screen displays a **green background** with the date, time, and barcodes, along with a **blue LED**.
- If they do not match, the screen shows a **red background** with the same details.

6.2 Project_Arduino_code.ino

Detailed description of the code:

- Lines 1-8: include the Arduino libraries.
- Lines 10-19: define the screen pins and creates instance of it.
- Lines 21-22: define the switch pin.
- Line 24-30: define the barcode pins and creates instance of it.
- Lines 32-37: define the real time clock pins and creates instance of it.
- Lines 39-43: define the AP WI-FI SSID, PW and create a web server that listen on port 80.
- Lines 45-46: define the blue LED pin.
- Lines 48-49: define the battery feedback pin.
- Lines 51-55: define global variables.
- Lines 57-101: “updateWifiPage” function that updates the WI-FI page(html) with the results[1].
- Lines 103-162: **setup function** initializes the screen, barcode, RTC, switch pin mode, LED pin mode, battery feedback pin mode, starts the ESP32 in an AP(access point) mode, and prints “welcome…” then the start page on the screen.
- Lines 164-188: “readBarcode” function that entire a while(1) loop until a data from the barcode received then returns the barcode, if the switch is pressed and the code not in the golden scan mode the function returns an empty barcode as a sign indicates the press of the switch.
- Lines 190-281: **loop function** that catch the switch press each iteration.
  - Lines 195-210: prints the scan golden barcode screen[1] and scan the golden barcode by calling the readBarcode function.
  - Lines 211-223: prints the scan target barcode screen[1].
  - Lines 225-279: entire while(1) loop, each iteration scans new target barcode, compare between the golden and the target barcodes, prints the identical barcodes screen[1] or different barcodes screen[1] based on the comparison results, update the WIFI page with the new comparison results and turn on the blue LED for half second if the barcodes are identical[1].

[1] Review 8.3 Device & working mode images section to see screen prints images and device modes.

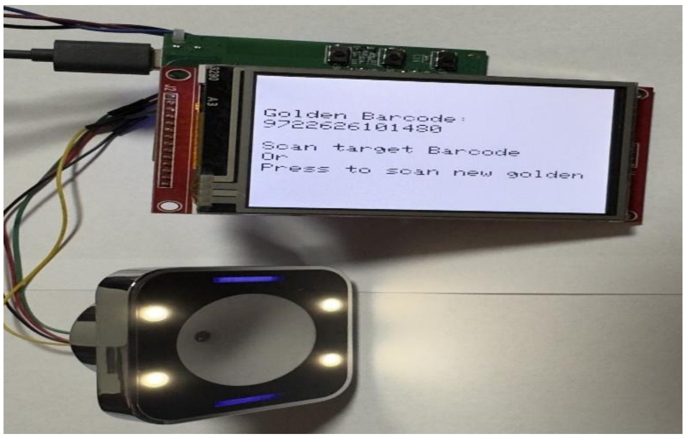
7. Summary and results

In this project, we were able to design a low cost & friendly user device with low cost & small size efficient components.

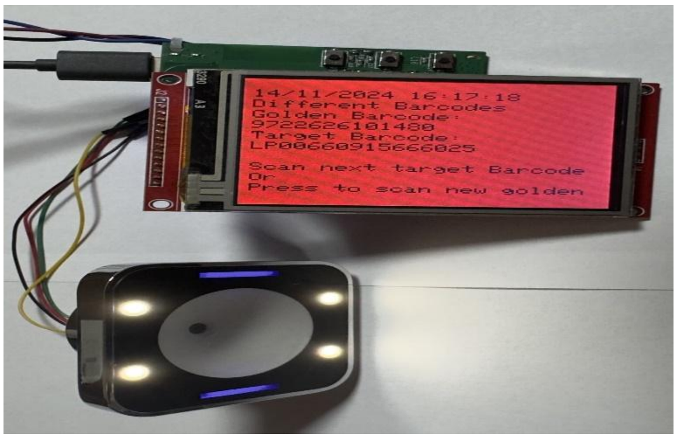
We tested each component alone with a "bring up" test, and then wrote the final code.

To test it, we acquired some barcodes with different lengths and values, decided which one's to use as "golden" and start scanning a series of barcodes, and comparing the given result to the desired result.

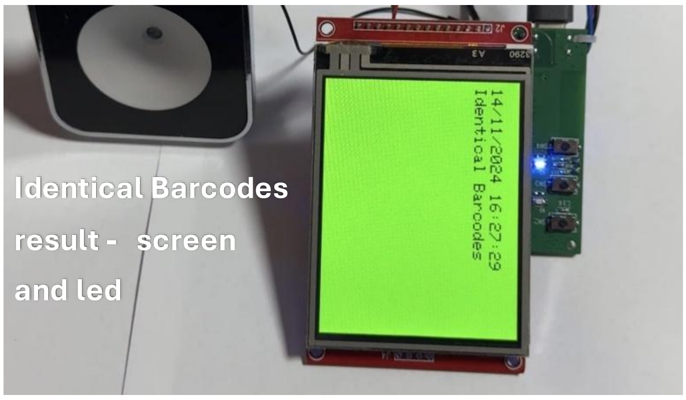
Our project was able to provide an accurate result when comparing Bar-codes, with a system of alarms that get the attention of the user if such a difference is deducted.



```
Golden Barcode:
9722626101480
Scan target Barcode
Or
Press to scan new golden
```

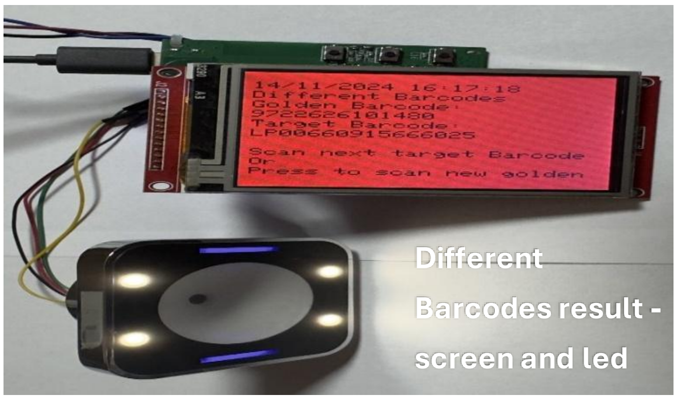


```
14/11/2024 16:17:18
Different Barcodes
Golden Barcode:
9722626101480
Target Barcode:
LP00660915666025
Scan next target Barcode
Or
Press to scan new golden
```



```
14/11/2024 16:27:29
Identical Barcodes
```

Identical Barcodes result - screen and led



```
14/11/2024 16:17:18
Different Barcodes
Golden Barcode:
9722626101480
Target Barcode:
LP00660915666025
Scan next target Barcode
Or
Press to scan new golden
```

Different Barcodes result - screen and led

Golden Barcode: 9722626101480

Target Barcode: LP00660915666025

Comparison: Different

Date: 14/11/2024

Time: 16:16:51

Different Barcodes result - WI-FI page

Golden Barcode: 9722626101480

Target Barcode: 9722626101480

Comparison: Identical

Date: 14/11/2024

Time: 16:16:21

7.1 Problems and solutions

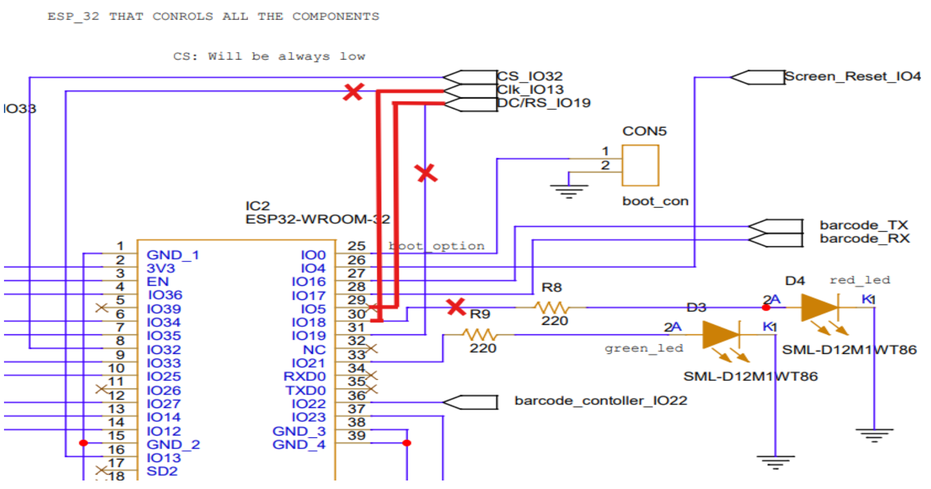
There were several components that didn't work as expected (SD card which does not save the data and the Screen)

We did use lot of components that needed “special” GPIO, mostly SPI connection, and due to human error, we ended up connecting wrong pins, and using some important SPI Pins to components that totally did not need them.

That caused most of our delays and re-attachment for connection, and caused us to lost some components (the red led).

When working on the screen, we miss placed the dc pin, which resulted on the screen being power, but not receiving any data from the esp32.

the connections and the fixes for the screen are shown below:



Also we had to abandonee the sd card usage.

As we had to reconnect the SD card to ensure its SPI connections matched those of the screen, except for the chip select (CS) pin. Initially, the connections were misplaced, which introduced added complexity. Managing the chip select signals to prevent both devices from being active simultaneously made it impossible to save data to the SD card while the screen was in use. This not only complicated the code but also caused operational issues with other PCB components. As a result, we were forced to abandon using the SD card, as it became impractical and unreliable to operate in this setup.

It would have been easier and more practical to use the other SPI IOs (VSPI). However, doing so would have required a complete redesign of the printed PCB, which would have added complexity, still we advice to do it when the project is re-done/ updated.

we attached an updated E.E Scheme with a separate SPI connection for the needed components, and we strongly advice to do a “first” setup with each component having its own connections, considering all the special connections each component had.

we identified this earlier, we could have adjusted the schematic, leading to a simpler design and more streamlined code.

7.2 Conclusions

We made full end to end process and achieved most of our goals. While working on the project we met a few difficulties.

We gained experience in design schematics and netlists, coding in Arduino and implementing layout.

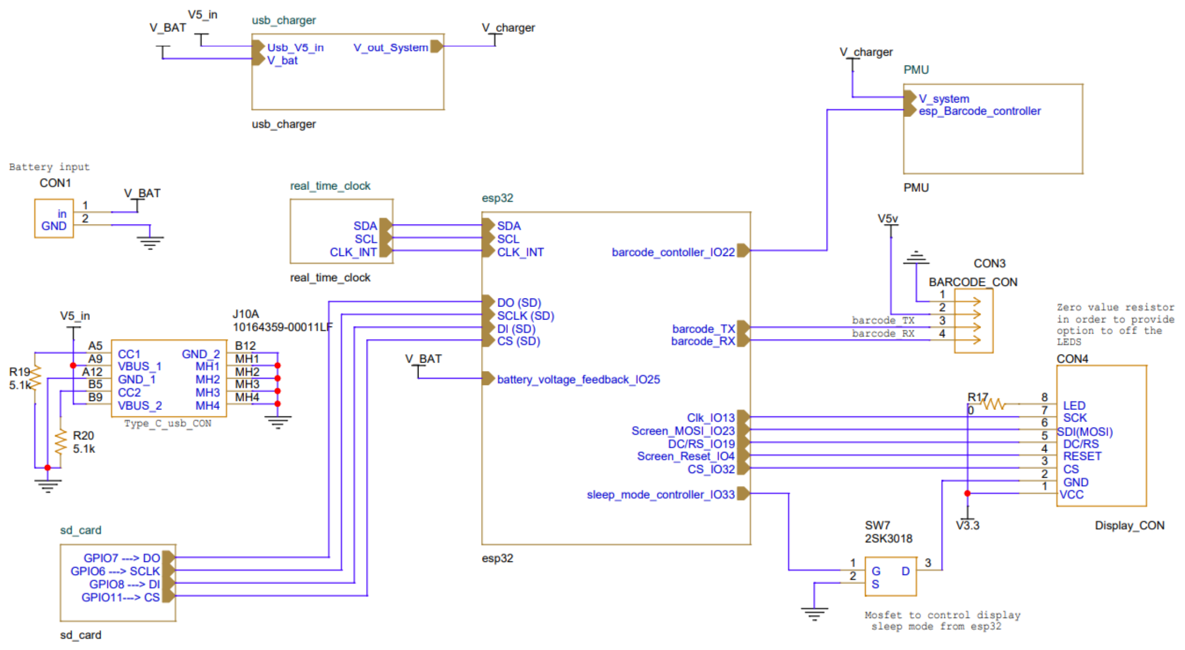
7.3 Future improvements

- Revise the electrical schematic to provide distinct connections for the SPI and DC/DL components.
- Add a buzzer.
- Add an option to remove the SD card, instead of overwriting its data when full.
- Make the device more user-friendly.

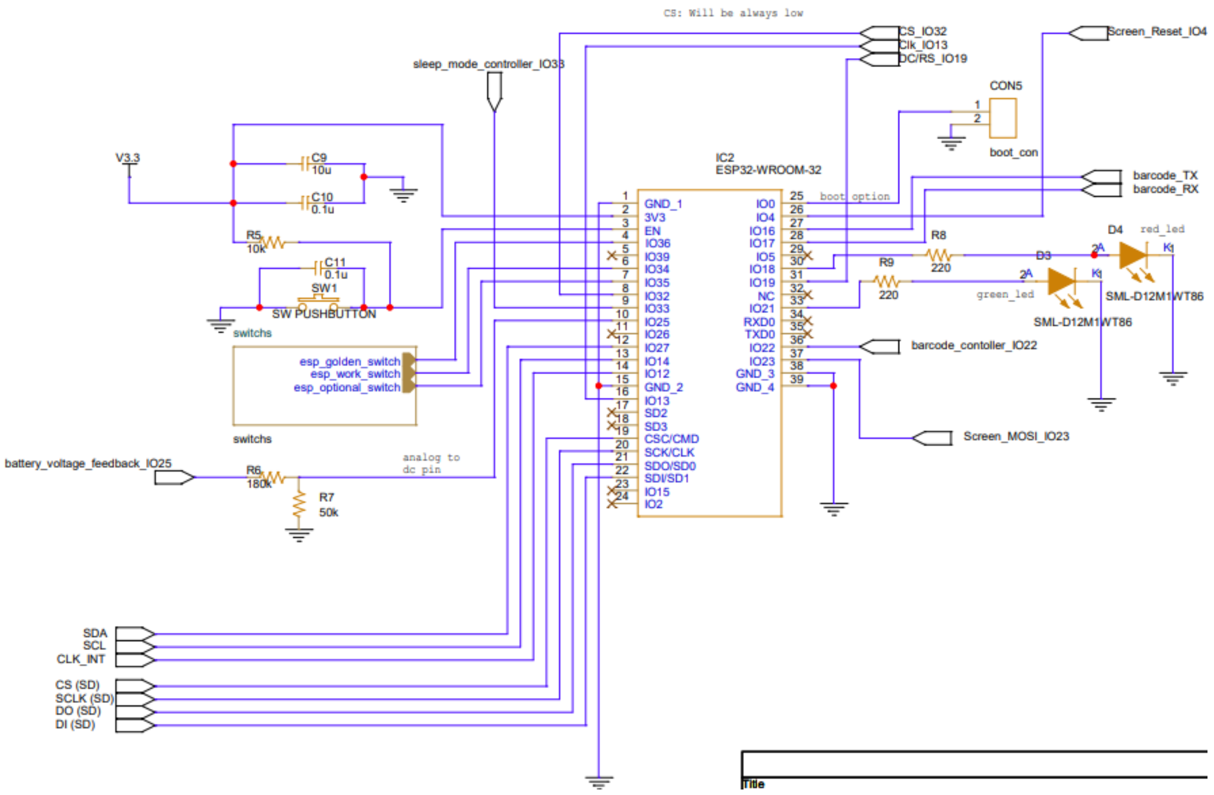
8.Appendix

8.1 Eelctrical schematic

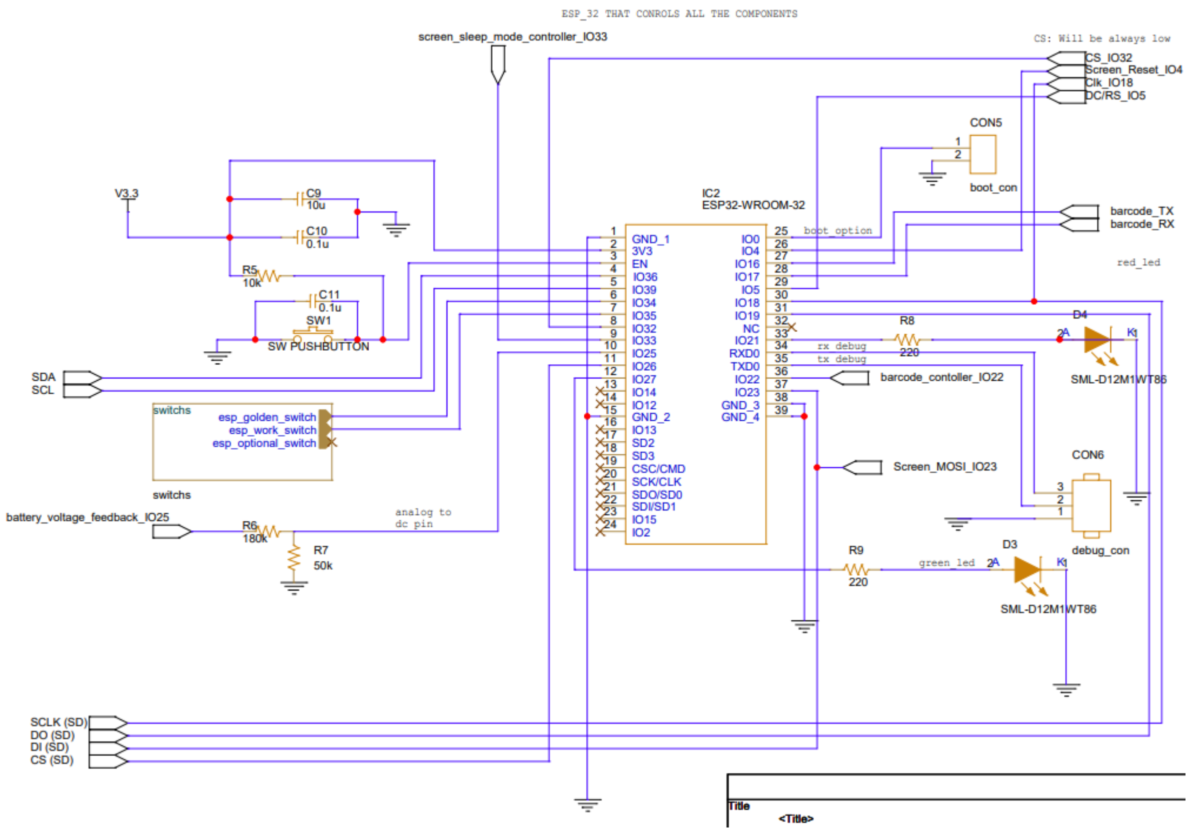
Top level:



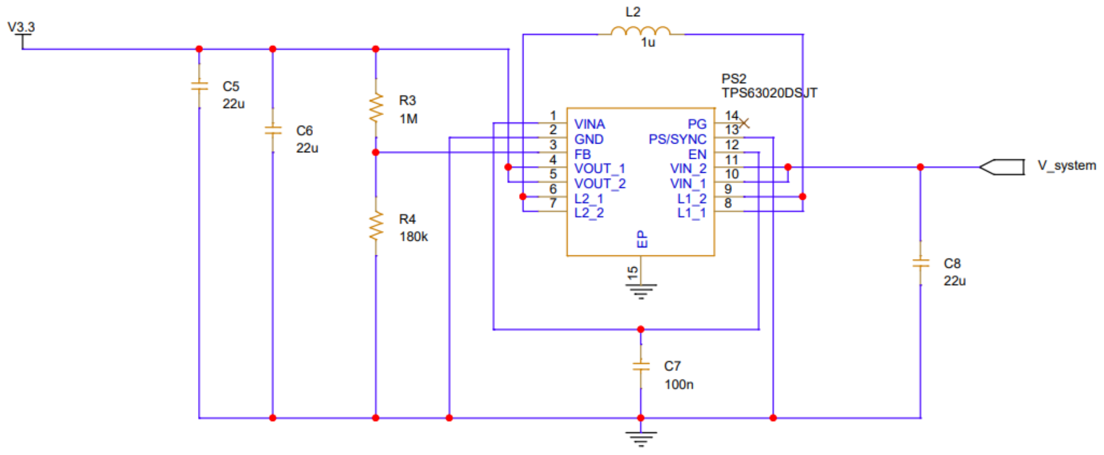
ESP32:



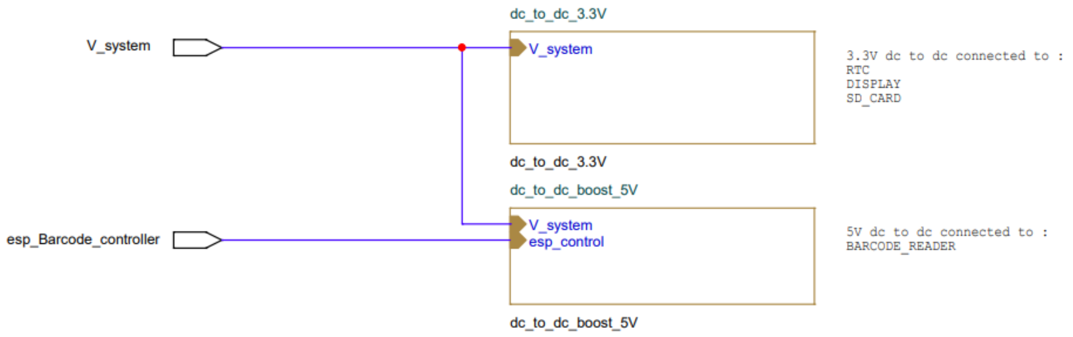
ESP32 re-wired:



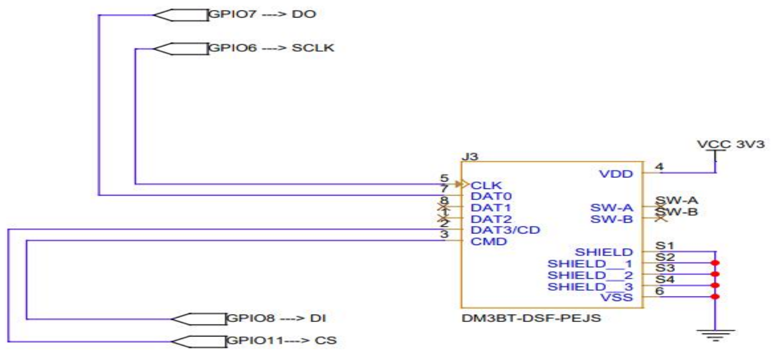
DC-TO-DC Converters (the 3.3v, the 5v is different slightly only in the resistor R3 value)



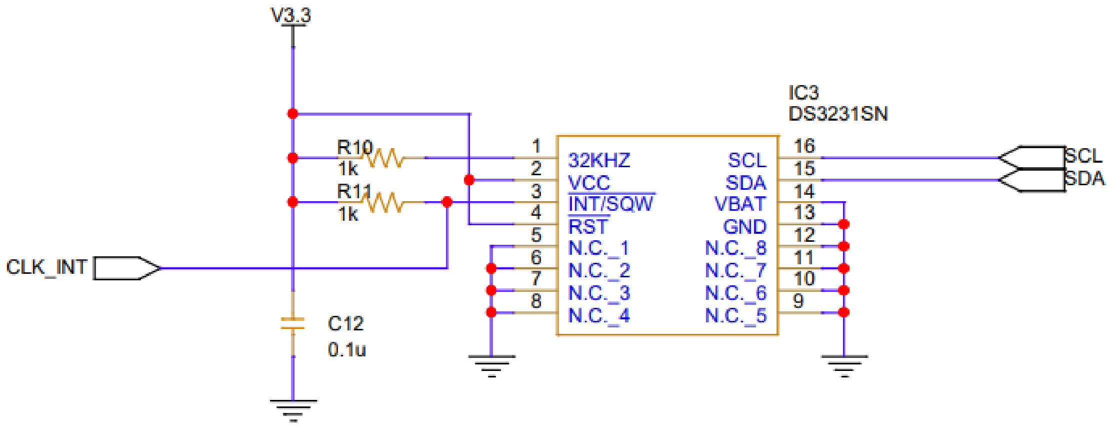
The Power Unit:



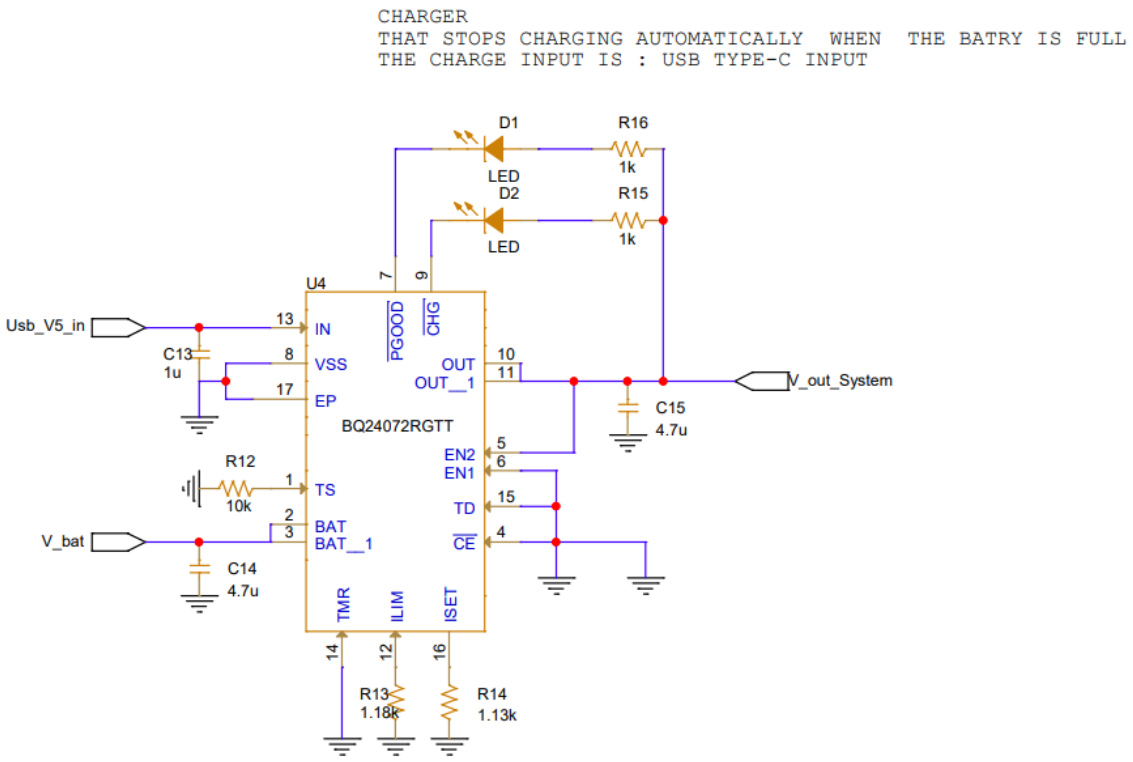
The SD Card slot:



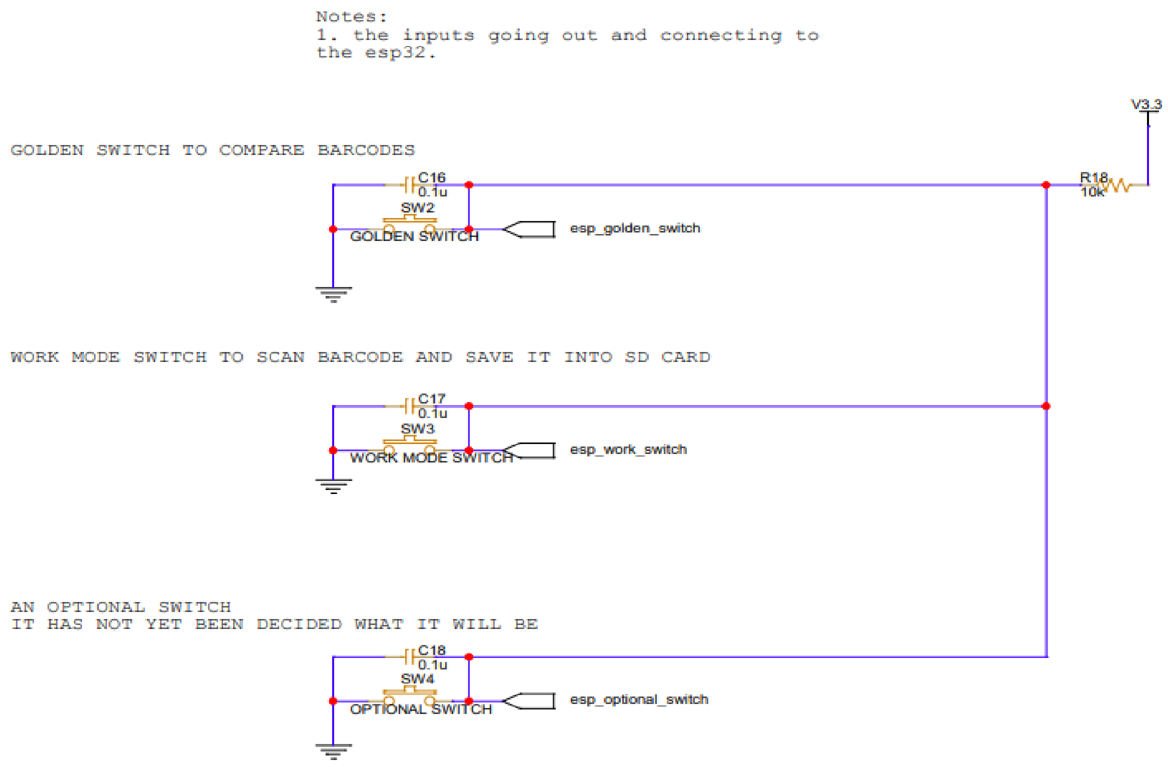
The RTC:



The USB Charger:

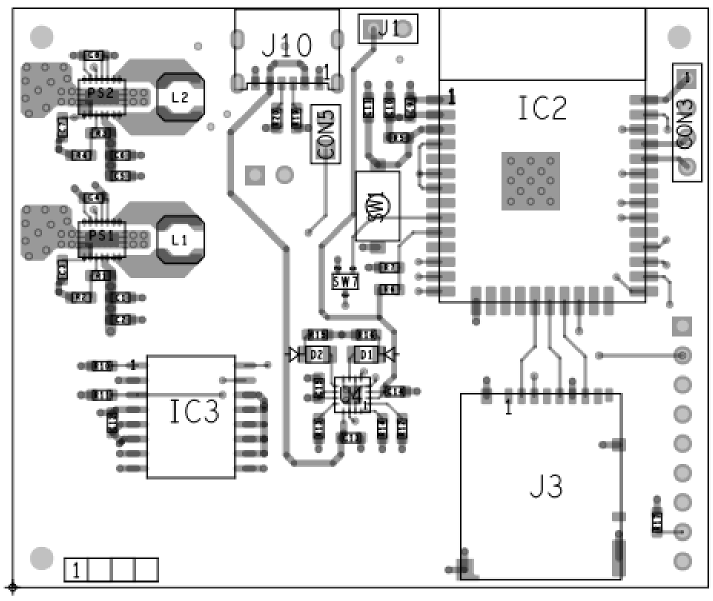


The switches:

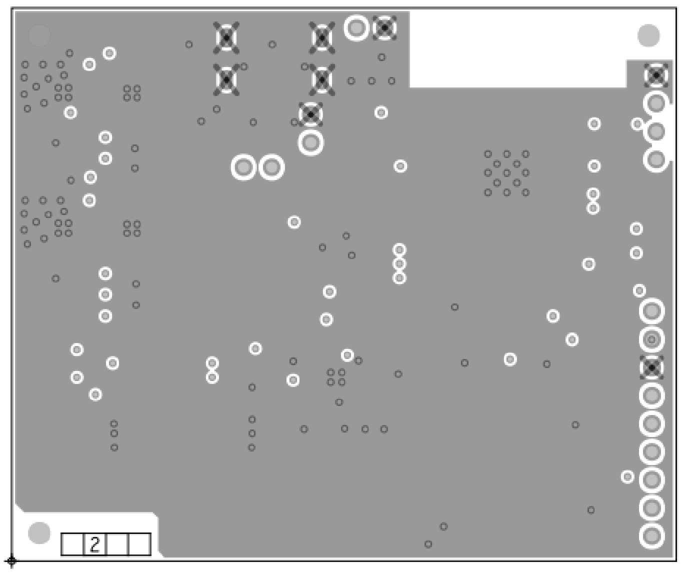

Notes:
1. the inputs going out and connecting to the esp32.
V3.3
R18
10k
GOLDEN SWITCH TO COMPARE BARCODES
C16
0.1u
SW2
GOLDEN SWITCH
esp_golden_switch
WORK MODE SWITCH TO SCAN BARCODE AND SAVE IT INTO SD CARD
C17
0.1u
SW3
WORK MODE SWITCH
esp_work_switch
AN OPTIONAL SWITCH
IT HAS NOT YET BEEN DECIDED WHAT IT WILL BE
C18
0.1u
SW4
OPTIONAL SWITCH
esp_optional_switch

8.2 Layout schematic

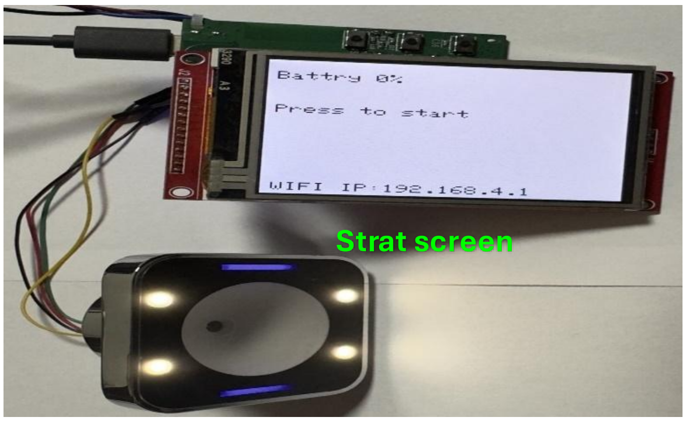
TOP:



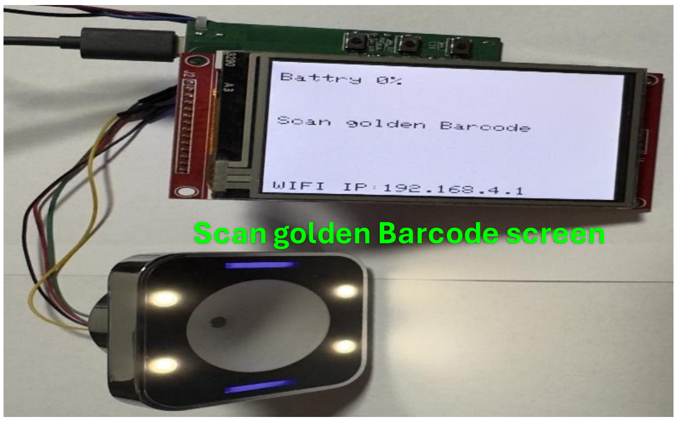
Bottom:



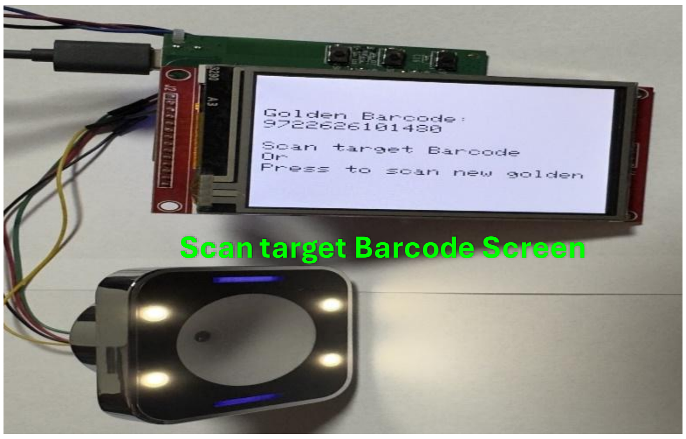
8.3 Device & working mode images



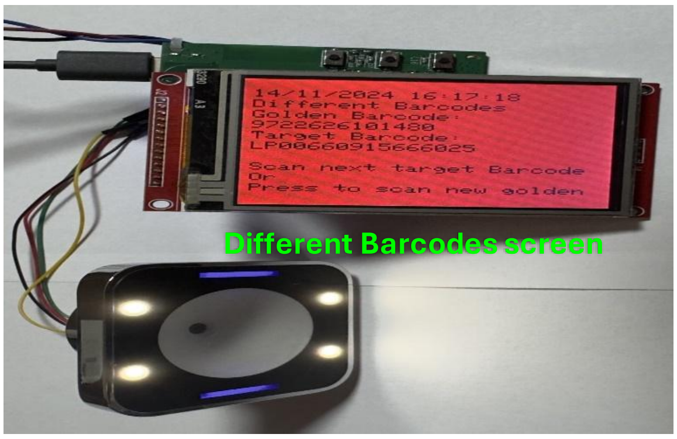
```
Battry 0%
Press to start
WIFI IP:192.168.4.1
```



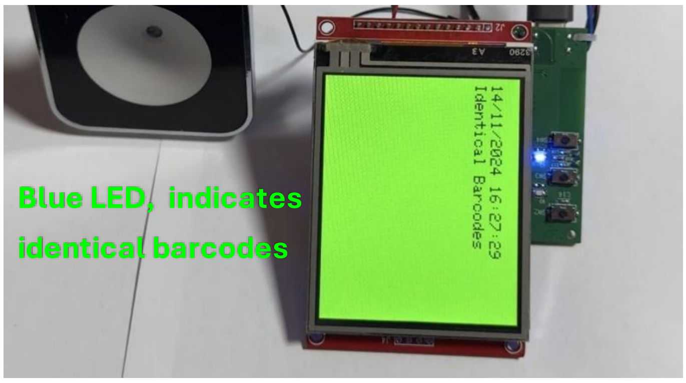
```
Battry 0%
Scan golden Barcode
WIFI IP:192.168.4.1
```



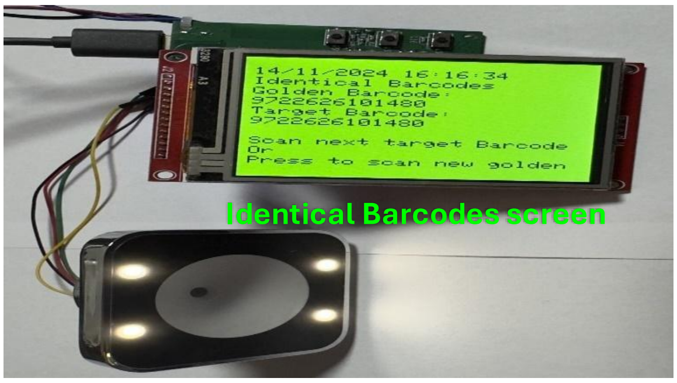
```
Golden Barcode:
9722626101480
Scan target Barcode
Or
Press to scan new golden
```



```
14/11/2024 16:17:18
Different Barcodes
Golden Barcode:
9722626101480
Target Barcode:
LP00660915666025
Scan next target Barcode
Or
Press to scan new golden
```



```
14/11/2024 16:27:29
Identical Barcodes
```



```
14/11/2024 16:16:34
Identical Barcodes
Golden Barcode:
9722626101480
Target Barcode:
9722626101480
Scan next target Barcode
Or
Press to scan new golden
```

Golden Barcode: 9722626101480

Target Barcode: LP00660915666025

Comparison: Different

Date: 14/11/2024

Time: 16:16:51

Different Barcodes WI-FI page

Golden Barcode: 9722626101480

Target Barcode: 9722626101480

Comparison: Identical

Date: 14/11/2024

Time: 16:16:21

Identical Barcodes WI-FI page

8.4 Refernces

We would like to express our gratitude to our supervisor, Mony, for his invaluable guidance and support. Without his assistance, we would not have completed this project. We have included some of the references Mony directed us to, which were essential for our progress.

We also extend our thanks to former HSDLS Lab students, whose reports provided valuable insights that helped us complete our project.

And a special thanks to all the lab staff.

Esp32 -
https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

Barcode -
https://www.dropbox.com/scl/fo/87hz5h82k25j3p9k5u603/AJfkL6iYDATRGkLYJjuhUJE?e=1&preview=GM810+Series+Barcode+reader+module+User+Manual-V1.2.1.pdf&rlkey=2fyvdir15kb1kj2ada1zkadqt&spm=a2g0o.detail.1000023.1.5cf0Orwf0rwfmn&st=x3ic3hkk&dl=0%E2%80%8B

Screen -
http://www.lcdwiki.com/2.8inch_SPI_Module_ILI9341_SKU:MSP2807

Real time clock -
https://www.mouser.co.il/datasheet/2/609/DS3231-3421123.pdf

USB type - C connector -
https://cdn.amphenol-cs.com/media/wysiwyg/files/drawing/10164359.pdf

Charger -
https://www.ti.com/lit/ds/symlink/bq24072.pdf?ts=1730900206995&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FBQ24072%252Fpart-details%252FBQ24072RGTT

SD slot -
https://www.mouser.co.il/datasheet/2/185/DM3BT_DSF_PEJS_CL0609_0029_9_00_2D_Drawing_00009199-1614465.pdf

Dc to dc -
https://www.ti.com/lit/ds/symlink/tps63020.pdf?ts=1730817589484&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTPS63020

MOSFET -
https://www.mouser.co.il/datasheet/2/258/2SK3018_SOT_323_-3364975.pdf