



Department of Electronic & Telecommunication Engineering
University of Moratuwa, Sri Lanka.

Design Document: RFID based asset tracking device for forklift

210610H Sirimanna N.T.W.

Submitted in partial fulfillment of the requirements for the module
EN 2160 Engineering Design Realization

06/07/2024

Overview

The design document serves as a comprehensive blueprint for the development of our RFID-based asset tracking system designed specifically for forklift operations. It outlines the market analysis, design specifications, system architecture, and implementation plan for the project. This document provides a detailed overview of the product, its functionalities, and the technical aspects of its design, including the RFID reader module, antenna design, power management, and data communication. It also includes testing and validation strategies, implementation timelines, and budget considerations. This document is intended to guide the development team throughout the project lifecycle, ensuring a systematic and efficient approach to delivering a high-quality asset tracking solution for forklifts.

Contents

1 General (10/02/2024)	3
1.1 Marketing Needs	3
1.1.1 Existing Product Analysis	3
1.1.2 User Profile	3
1.2 Market Segment Analysis	3
1.3 Final Specifications	3
2 RFID Reader Design	4
2.1 RFID chip Selection (24/02/2024)	4
2.1.1 NXP MFRC522	4
2.1.2 NXP PN532	5
2.1.3 Texas TRF7970A	6
2.1.4 Selected IC – NXP PN532	6
2.2 RFID Reader Circuit Diagram (26/02/2024)	7
2.3 Antenna and Matching Circuit Design (11/03/2024)	8
2.3.1 Equivalent circuit	9
2.3.2 Determination of parallel equivalent circuit	9
2.3.3 Circuit Diagram (Antenna, Matching Circuit, EMC Filter)	9
3 How to access WIFI connection, ESP32 WROOM 32 (25/03/2024)	10
3.1 SPI Communication with ESP32 WROOM 32	11
3.2 Programming ESP32 IC	12
3.3 Final Circuit Diagram - ESP32 WROOM 32	12
4 Power Circuit Design (02/04/2024)	12
4.1 Voltage Regulator - LM2596S-3.3	12
4.2 Power Circuit Diagram	13
5 Output Circuit Design (04/04/2024)	14
5.1 Buzzer Circuit	14
5.2 LED Indicator Circuit	14
6 Interconnection Between Schematics	15
7 PCB Design (05/04/2024)	15
8 Bill Of Materials	18
9 Firmware Implementation (12/04/2024)	19
9.1 Importance of Register Programming	19
9.2 Compatibility Issue Between the Adafruit PN532 library and ESP32 IC	25

10 Enclosure Design (08/03/2024)	26
10.1 Upper Housing Component	26
10.2 Lower Chassis Component	27
10.3 Enclosure Integration System	28
11 Other Components	29
11.1 ON OFF Switch	29
11.2 Power Input	29
12 PCB Manufacturing (03/05/2024)	30
12.1 PCB Specifications	30
13 Bare PCB	30
13.1 Soldered PCB	31
14 PCB Testing	31
15 Physically Build Enclosure	32
16 System Integration	34
17 Product Specifications	34
18 Future Improvements	35
19 Acknowledgement	35
20 Data sheets and References	36
21 Appendix - Daily Log Entries	37

1 General (10/02/2024)

1.1 Marketing Needs

The proposed RFID-based product for tracking assets loaded onto forklifts addresses a critical need in Sri Lanka's market. With the absence of major companies offering similar solutions, there is a clear opportunity to introduce a cost-effective product that enhances warehouse management efficiency. This solution will enable warehouse managers to easily locate assets within the warehouse, improving overall operational efficiency and asset tracking accuracy.

1.1.1 Existing Product Analysis

- [Link - A Case Study of Smart Forklift Integration with RFID](#)
- [Link - An RFID data-based forklift traffic management system by Research gate](#)

Solutions from industry leaders in forklift-mounted RFID technology, such as Honeywell and Impinj, were analyzed through product documentation, technical specifications, and online resources. Key features such as compatibility with forklift mounting, ruggedness for industrial environments, and integration capabilities with existing warehouse management systems were highlighted. Importing these products into Sri Lanka would incur significant costs due to shipping, taxes, and import duties, adding to the overall expense of implementing such solutions.

1.1.2 User Profile

The primary users of this device are warehouse managers, logistics supervisors, and inventory control personnel. These individuals are responsible for overseeing the efficient movement, storage, and tracking of assets within a warehouse or distribution center.

1.2 Market Segment Analysis

The market segment for the product is the logistics and warehouse management sector, which includes businesses seeking to optimize asset tracking and inventory management processes.

1.3 Final Specifications

The final specifications for the device outline the key requirements and features of the device.

1. Functionality: The device should be able to track assets loaded onto forklifts within a warehouse environment.
2. Technology: The device should utilize RFID technology for asset tracking.
3. Range: The device should have a sufficient range to track assets within a warehouse or distribution center.
4. Accuracy: The device should provide accurate asset tracking information to improve operational efficiency.
5. Integration: The device should integrate with existing warehouse management systems for seamless asset tracking and management.
6. User Interface: The device should have a user-friendly interface for warehouse managers, logistics supervisors, and inventory control personnel.
7. Durability: The device should be rugged and able to withstand the demands of an industrial environment.
8. Cost-Effectiveness: The device should be cost-effective to implement, considering the potential costs of importing similar solutions.

2 RFID Reader Design

2.1 RFID chip Selection (24/02/2024)

The selection of the RFID IC (Integrated Circuit) is a crucial aspect of the forklift asset tracking device project. RFID ICs play a vital role in enabling the device to accurately and efficiently track assets within a warehouse or distribution center environment. The choice of RFID IC impacts the device's performance, compatibility with existing systems, and overall cost-effectiveness. Therefore, careful consideration must be given to selecting the RFID IC that best meets the project's requirements and objectives.

First, I selected three alternatives as the IC.

- NXP MFRC522
- NXP PN532
- Texas TRF7970A

2.1.1 NXP MFRC522

The MFRC522 chip is a highly integrated contactless reader IC (Integrated Circuit) for 13.56 MHz RFID (Radio Frequency Identification) communication. It is commonly used in projects requiring RFID technology, such as access control systems, payment systems, and asset tracking devices. The MFRC522 chip supports ISO/IEC 14443 Type A and Type B protocols and MIFARE Classic RFID tags. It provides a simple and effective way to add RFID functionality to projects, offering features like anti-collision and 7-byte unique identifier (UID) reading. The MFRC522 chip's versatility, ease of use, and robust performance make it a popular choice for RFID applications.

1. Frequency Compatibility: The MFRC522 operates at 13.56 MHz, which is a standard frequency for RFID applications, including asset tracking. This ensures that it can communicate effectively with RFID tags used for asset identification.
2. Protocol Support: The MFRC522 chip supports ISO/IEC 14443 Type A and Type B protocols, which are widely used in RFID applications. This allows it to communicate with a variety of RFID tags, including MIFARE Classic tags, which are commonly used for asset tracking.
3. Versatility: The MFRC522 chip is versatile and can be easily integrated into my project. It provides features such as anti-collision and 7-byte UID reading, which are essential for reliable and efficient asset tracking.
4. Availability and Cost: The MFRC522 chip is widely available and cost-effective.



Figure 1: NXP MFRC522 IC

2.1.2 NXP PN532

The PN532 chip is a highly integrated NFC (Near Field Communication) controller that supports various modes of NFC communication, including ISO/IEC 14443 Type A and Type B, FeliCa, and NFCIP-1 (ISO/IEC 18092) standards. It is commonly used in applications such as access control, payment systems, and interactive devices.

The PN532 chip offers a range of features, including support for peer-to-peer communication, card emulation mode, and reader/writer mode. It operates at 13.56 MHz, which is a standard NFC operating frequency, and provides a flexible and easy-to-use interface for integrating NFC functionality into projects.

1. Frequency Compatibility:

- PN532: The PN532 chip operates at 13.56 MHz, which is a standard frequency for NFC communication.

2. Protocol Support:

- PN532: Supports ISO/IEC 14443 Type A and Type B, FeliCa, and NFCIP-1 standards, providing a wide range of compatibility.

3. Versatility:

- PN532: The PN532 chip offers a high level of versatility, supporting peer-to-peer communication, card emulation mode, and reader/writer mode.
- MFRC522: The MFRC522 chip is less versatile compared to the PN532, mainly focusing on RFID applications and lacking support for peer-to-peer communication and card emulation mode.

4. Availability and Cost:

- The PN532 chip is widely available and cost-effective, making it a practical choice for projects with budget constraints. However, it is more expensive than the MFRC522.



Figure 2: NXP PN532 IC

- [Link -PN532/C1 datasheet](#)
- [Link -PN532 and MFRC522 Antenna Design Guideline](#)

2.1.3 Texas TRF7970A

The Texas TRF7970A is a highly integrated RFID reader/writer IC designed for 13.56-MHz RFID systems. It supports various RFID standards, including ISO/IEC 14443A/B, ISO/IEC 15693, ISO/IEC 18000-3, and NFCIP-1, making it suitable for a wide range of RFID applications. The TRF7970A offers advanced features such as simultaneous protocol support, automatic modulation index adjustment, and selectable output power levels, providing flexibility and reliability in RFID communication. Its versatility and performance make it an excellent choice for implementing RFID functionality in my asset tracking device project.

1. Frequency Compatibility:

- **TRF7970A:** The TRF7970A operates at 13.56 MHz, which is compatible with NFC and RFID standards.
- **PN532:** The PN532 chip also operates at 13.56 MHz, making it compatible with the same NFC and RFID frequency.

2. Protocol Support:

- **TRF7970A:** Supports a variety of RFID standards, including ISO/IEC 14443A/B, ISO/IEC 15693, ISO/IEC 18000-3, and NFCIP-1.
- **PN532:** Supports ISO/IEC 14443 Type A and Type B, FeliCa, and NFCIP-1 standards.

3. Versatility:

- **TRF7970A:** The TRF7970A offers advanced features such as simultaneous protocol support, automatic modulation index adjustment, and selectable output power levels, providing flexibility in RFID communication.
- **PN532:** The PN532 chip also offers versatility, supporting peer-to-peer communication, card emulation mode, and reader/writer mode.

4. Availability and Cost:

- This chip is widely available, but it is more expensive than both other chips.

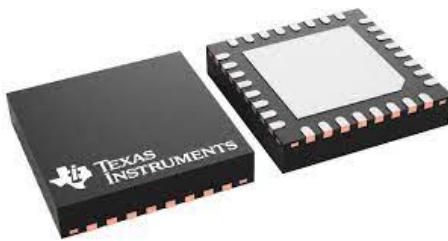


Figure 3: TEXAS TRF7970A IC

- [Link -TRF7970A Datasheet](#)
- [Link -TRF7970A Antenna Design Guide](#)

2.1.4 Selected IC – NXP PN532

The PN532's overall functionality can be separated into three functions:

- **Generate the RF field:** The generated magnetic field has to be maximized within the limits of the transmitter supply current and general emission limits.

- **Transmit data:** The coded and modulated data signal has to be transmitted in a way that every card and PN532 device is able to receive it. The signal shape and timing according to relevant standards have to be considered.
- **Receive data:** The response of a card or PN532 device has to be transferred to the receive input of the PN532 considering various limits, e.g., maximum voltage and receiver sensitivity.

2.2 RFID Reader Circuit Diagram (26/02/2024)

The PN532 (PN5321A3HN/C106) supports SPI (Serial Peripheral Interface) communication, allowing microcontrollers like the ESP32 to interact with RFID tags.

Here is the final circuit diagram for operating the PN532 chip for RFID card reading under SPI communication.

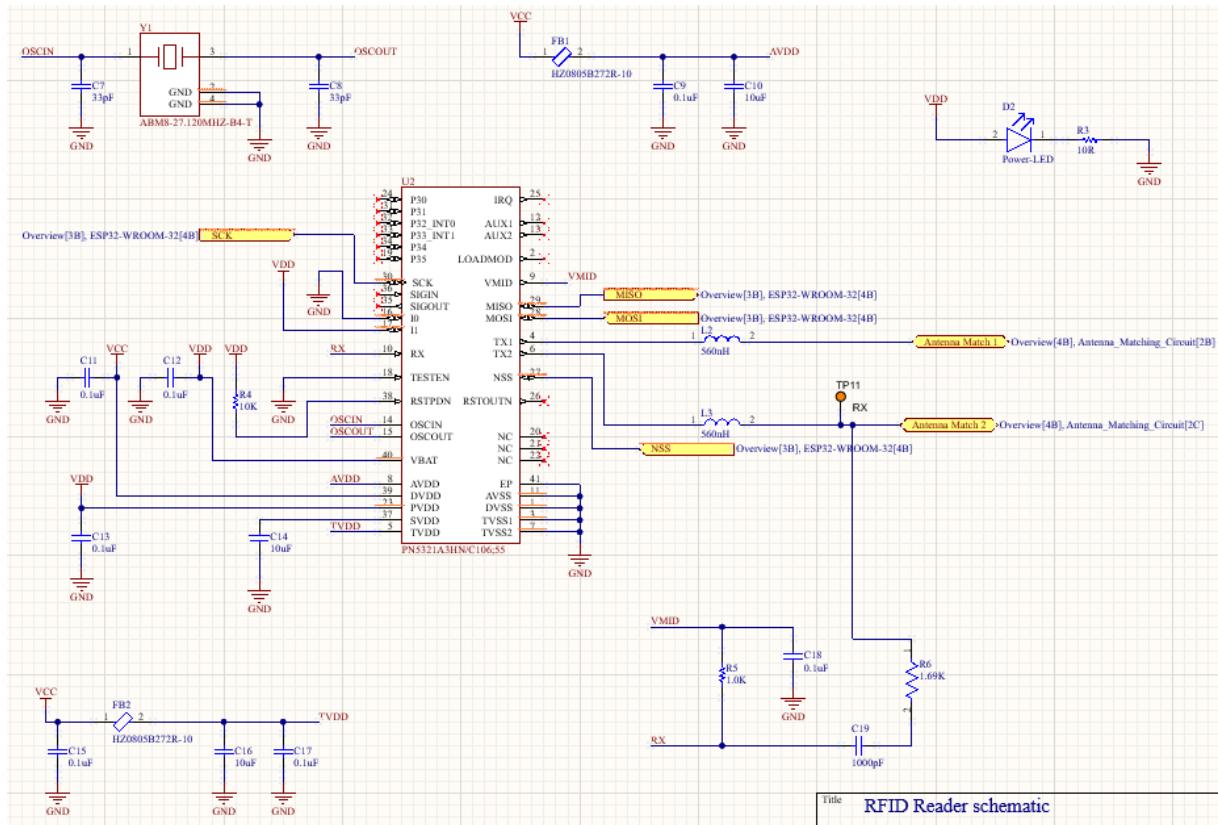


Figure 4: PN532 Circuit Diagram

To transmit data to the ESP32 using SPI, the following pins on the PN532 module should be connected to the corresponding pins on the ESP32:

- PN532 pin 30 (SCK) to ESP32 SCK pin
- PN532 pin 28 (MOSI) to ESP32 MOSI pin
- PN532 pin 29 (MISO) to ESP32 MISO pin
- PN532 pin 27 (SS) to ESP32 SS pin

The D1 LED turned on when the PN532 IC powered up. Also the PN532 IC operates using a 27.12MHz crystal oscillator for its internal timing requirements. This oscillator provides the necessary clock frequency for the IC to function correctly.

2.3 Antenna and Matching Circuit Design (11/03/2024)

The NXP Semiconductors PN532 chip, which enables contactless communication, is present in our PCB. When combined with the PCB antenna, it functions as an NFC reader for the ISO 14443 standard. The antenna has to match the PN532 chip in order to extend the reader's communication range. Aside from that, the signal that the tag reflected and the antenna detected needed to be optimized.

Since the PN532 chip outputs a 13.56MHz square wave, a low-pass filter is used to follow the output pins and produce a pure sine wave. The majority of power is sent in the reader-antenna configuration below when the antenna's impedance matches the chip's impedance.

The RF block diagram in Fig 7 shows a recommended circuitry design with all relevant components required to connect an antenna to the PN532. It also ensures the transmission of energy and data to the target device as well as the reception of a target device answer.

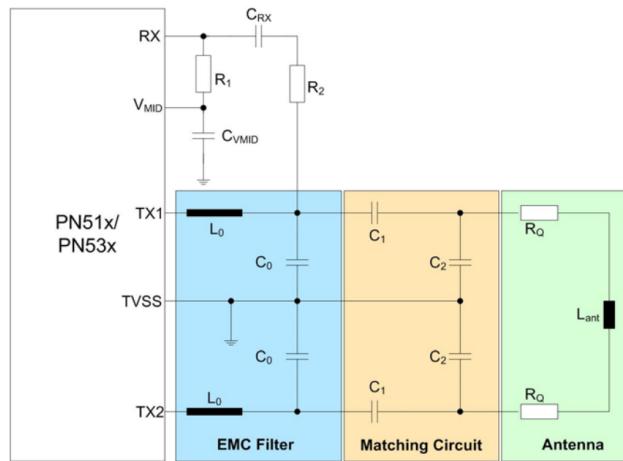


Figure 5: Block Diagram of complete RF part

- **EMC filter:** The EMC filter reduces 13.56 MHz harmonics and performs an impedance transformation.
- **Matching Circuit:** The Matching Circuit acts as an impedance transformation block.
- **Antenna coil:** The antenna coil itself generates the magnetic field.

The receiving part provides the received signal to the PN532 internal receiving stage.

Basically, this complete RF circuitry consists of at least 8 capacitors, 2 inductors, 2 resistors (the part size determines the maximum power which the resistor can withstand) and the symmetrical antenna coil as shown in Fig 1.

2.3.1 Equivalent circuit

There are two alternatives for equivalent circuit,

- **Series equivalent circuit:** The antenna loop has to be connected to an impedance or network analyzer to measure the series equivalent components.
- **Parallel equivalent circuit:** The parallel equivalent circuit of the antenna together with the added external damping resistor R_Q .

2.3.2 Determination of parallel equivalent circuit

The parallel equivalent circuit of the antenna together with the added external damping resistor R_Q has to be measured. The quality factor should be checked again to be sure to achieve the required value of $Q=35$.

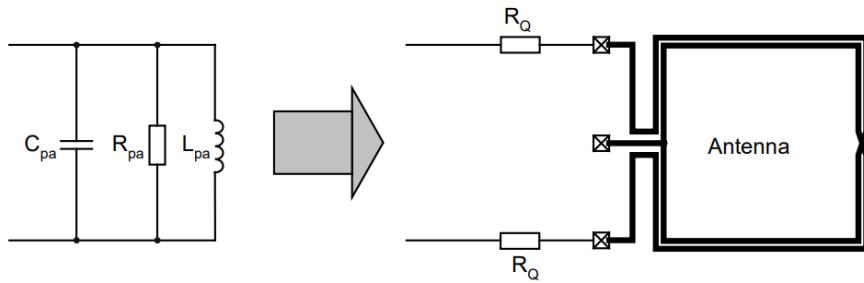


Figure 6: Parallel equivalent circuit

2.3.3 Circuit Diagram (Antenna, Matching Circuit, EMC Filter)

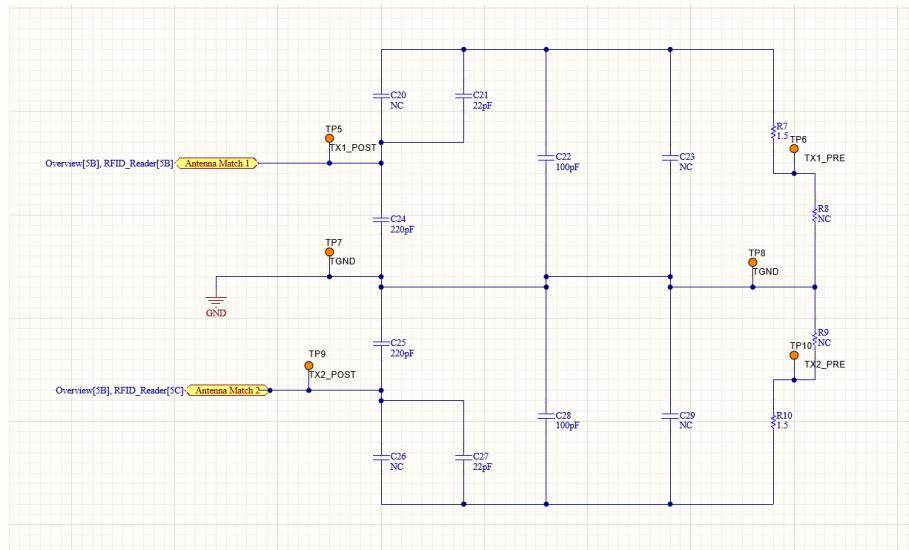


Figure 7: Circuit Diagram (Antenna, Matching Circuit, EMC Filter)

The circuit diagram for the PN532 RFID IC operating at 13.56MHz includes the antenna, matching circuit, and EMC filter. The antenna is designed to resonate at the operating frequency of 13.56MHz and is typically a coil or loop structure. The matching circuit is used to match the impedance of the antenna to the impedance of the PN532 IC for maximum power transfer. The EMC filter helps reduce electromagnetic interference (EMI) from the circuit, ensuring reliable operation and compliance with regulatory standards. This circuit diagram is crucial for understanding the RF front-end design of the RFID system using the PN532 IC.

According to the PN532 chip datasheet, the receive pin of the chip must not exceed AVDD +1 V, where AVDD is the analog supply voltage at 3.3V, resulting in a maximum allowed voltage of 4.3V on the RX pin. This limit applies when the chip is in continuous communication mode and the RF-field is continuously active. Even if the antenna is detuned by a tag, the voltage on the RX pin must not exceed 4.3V. The antenna transfers the most power when its input impedance matches 50Ω and the reactance is 0Ω at the resonance frequency.

To ensure accurate measurements, it's important to keep the ground connection short and close to the measurement point, as any ground wire within the electromagnetic field can induce voltage in the probe, leading to unreliable measurements.

3 How to access WIFI connection, ESP32 WROOM 32 (25/03/2024)

The selection of the ESP32 Wroom 32 chip for WiFi connectivity in our project was a strategic choice driven by several key factors. Firstly, the ESP32 Wroom 32 offers robust and reliable WiFi connectivity, allowing us to seamlessly upload data to cloud storage. This is crucial for our project, as it enables real-time monitoring and tracking of assets in the warehouse environment.

The ESP32 Wroom 32 is highly versatile and supports a wide range of applications, making it suitable for our asset tracking device. Its compatibility with popular development platforms such as Arduino and its extensive documentation and community support were also significant factors in our decision-making process.

Additionally, the ESP32 Wroom 32 offers low power consumption, which is essential for the project's requirements. Its dual-core processor and ample memory resources provide the necessary processing power for handling data uploads to the cloud efficiently. The ESP32 Wroom 32 also features three power modes (active, sleep, and deep sleep), allowing us to optimize power consumption based on the device's activity. This feature is crucial for our application, as it helps us maintain power efficiency and avoid high-speed discharge of the forklift battery, ensuring reliable operation over extended periods.



Figure 8: ESP32 WROOM 32

- **Link -ESP32 WROOM 32 datasheet**

Overall, the ESP32 Wroom 32 chip stood out as the ideal choice for WiFi connectivity in this project, offering a combination of reliability, versatility, and performance that align perfectly with our project goals.

3.1 SPI Communication with ESP32 WROOM 32

The SPI Bus is a synchronous communication protocol used to transfer data between a microcontroller and other devices, such as sensors, displays, memories, and peripherals.

Unlike other protocols, the SPI Bus uses multiple communication lines, allowing for high transfer speeds and exceptional flexibility.

The SPI port is mainly used for its high data transmission capacity. For this reason, you will find it in devices that require a large amount of data per second, such as storage devices (memories, SD cards) and displays.

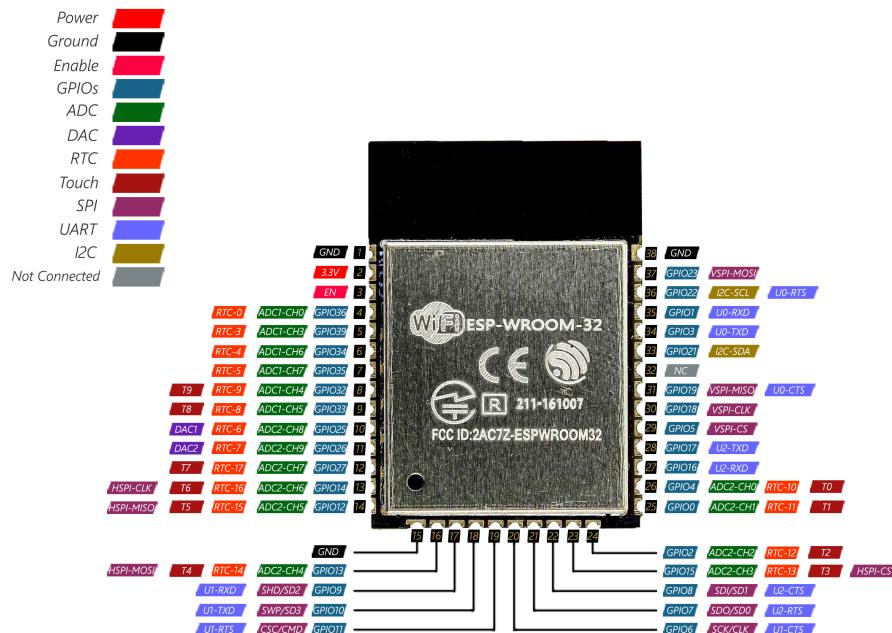


Figure 9: ESP32 WROOM 32 Pinout

The ESP32 WROOM 32 microcontroller supports two SPI (Serial Peripheral Interface) buses: VSPI (Virtual SPI) and HSPI (Hardware SPI). VSPI is a software-based SPI bus that can be mapped to any set of GPIO pins on the ESP32, providing flexibility in pin selection but may have slightly lower performance compared to HSPI. But in this case, performance is not a critical thing. So I am going to use VSPI communication.

- **VSPI MISO (Master In Slave Out):** GPIO 19
- **VSPI MOSI (Master Out Slave In):** GPIO 23
- **VSPI SCK (Serial Clock):** GPIO 18
- **VSPI CS (Chip Select):** GPIO 5

3.2 Programming ESP32 IC

Two important pins for programming the ESP32 chip are RX0 (Receive Data) and TX0 (Transmit Data). These pins are used for serial communication between the ESP32 and a programming device. RX0 is the input pin on the ESP32 for receiving data, while TX0 is the output pin for transmitting data. When programming the ESP32, these pins are typically connected to the corresponding RX and TX pins on the programming device. By using RX0 and TX0, developers can easily upload firmware and communicate with the ESP32 IC.

3.3 Final Circuit Diagram - ESP32 WROOM 32

Here is the final circuit diagram for ESP32 IC for connect with PN532 over VSPI communication.

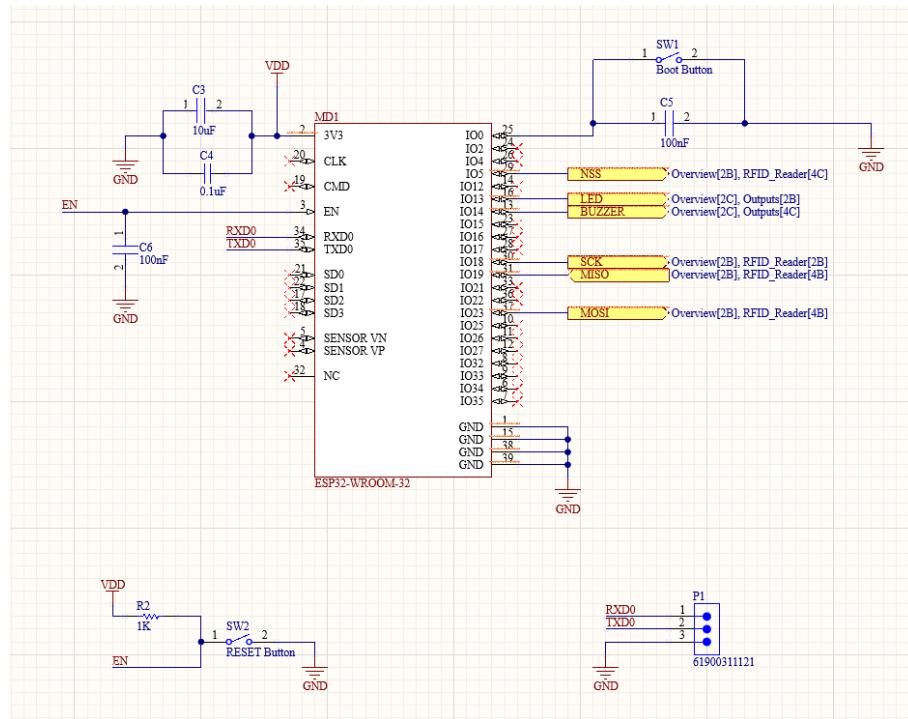


Figure 10: ESP32 WROOM 32 - Circuit Diagram

4 Power Circuit Design (02/04/2024)

In the power circuit design, the goal is to efficiently convert the 12V battery power from the forklift into a stable 3.3V output to power the components of the system. This section will detail the components and their connections required to achieve this voltage conversion, ensuring that the RFID reader and other electronics operate reliably. The design will focus on efficiency and reliability, considering factors such as voltage regulation, and current capacity.

4.1 Voltage Regulator - LM2596S-3.3

Selecting the right voltage regulator is crucial for ensuring stable and reliable power supply in electronic projects. For our project, which involves the PN532 and ESP32 WROOM 32 chips operating on 3.3V DC voltage, choosing the appropriate regulator was a critical decision. Factors such as efficiency, cost, thermal performance, and compatibility with the chips' voltage requirements were considered. After careful

evaluation, the LM2596S-3.3 regulator was selected due to its efficiency, cost-effectiveness, and compatibility with our project's needs. This selection process highlights the importance of choosing the right voltage regulator to meet the power supply requirements of electronic devices.

- **Link -LM2596S-3.3 datasheet**

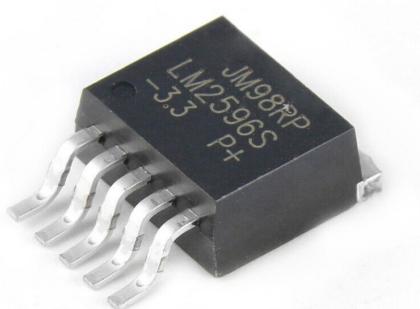


Figure 11: LM2596S-3.3 Regulator

4.2 Power Circuit Diagram

Here is the final power circuit diagram, showcasing the efficient conversion of the 12V battery power from the forklift into a stable 3.3V output.

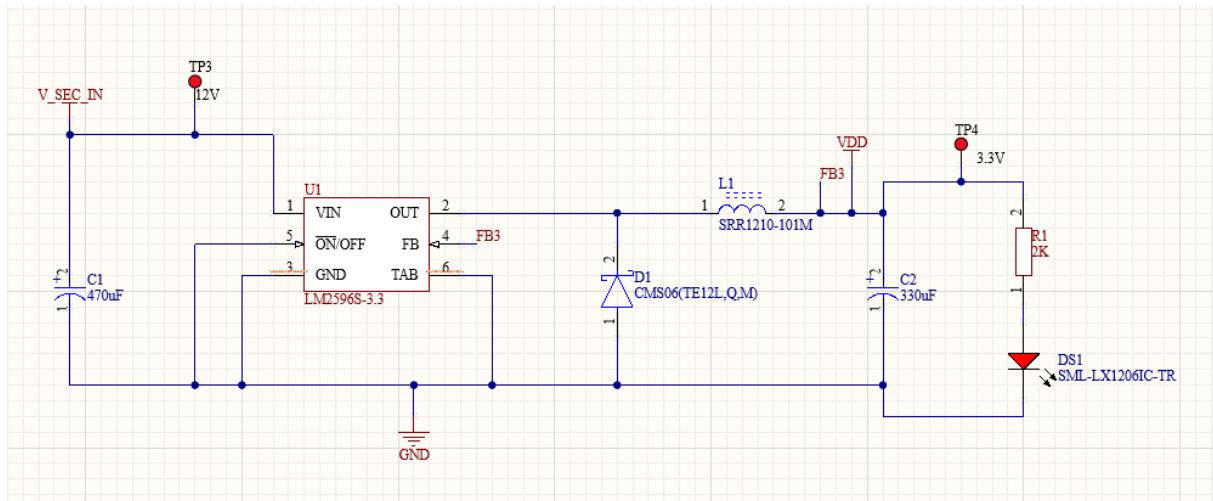


Figure 12: Power Circuit Diagram

- **D1 - CMS06 Schottky Diode:** The CMS06 is a compact and versatile power management module designed to provide efficient power conversion for various electronic devices. The CMS06 is selected for its compact size, high efficiency, and ability to handle the required voltage and current levels. Its inclusion ensures reliable and efficient power delivery, critical for the overall performance and functionality of the system.
- **DS1 - RED Led:** This indicator signifies the operational status of the device, providing a visual cue when power is actively supplied.

5 Output Circuit Design (04/04/2024)

The device integrates two indicators, a buzzer, and a green LED, to provide informative feedback to the user. These indicators play crucial roles in enhancing user interaction and awareness regarding asset detection:

- **Buzzer:** The buzzer serves as an auditory indicator, emitting a distinct beep sound when an asset is detected. This audible signal is useful in environments where visual cues may not be immediately noticeable, ensuring that users are promptly informed of detected assets.
- **Green LED:** The green LED functions as a visual indicator, illuminating to signal the detection of an asset. This visual feedback is valuable in situations where users need to quickly identify the presence of detected assets, providing a clear and intuitive indication.

The combination of these indicators offers users dual modes of feedback, accommodating different preferences and environmental conditions.

5.1 Buzzer Circuit

Presented below is the circuit diagram for connecting the buzzer, illustrating the necessary connections and components required for integrating the buzzer into the system.

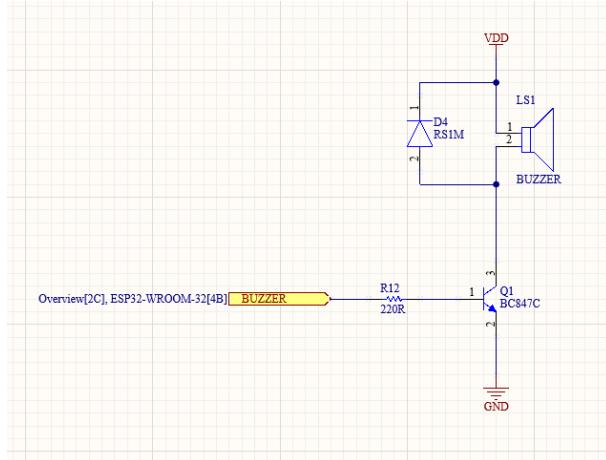


Figure 13: Circuit Diagram - Buzzer

5.2 LED Indicator Circuit

Presented below is the circuit diagram for connecting the LED, illustrating the necessary connections and components required for integrating the LED into the system.

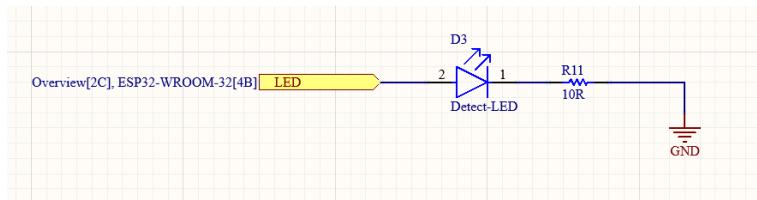


Figure 14: Circuit Diagram - LED Indicator

6 Interconnection Between Schematics

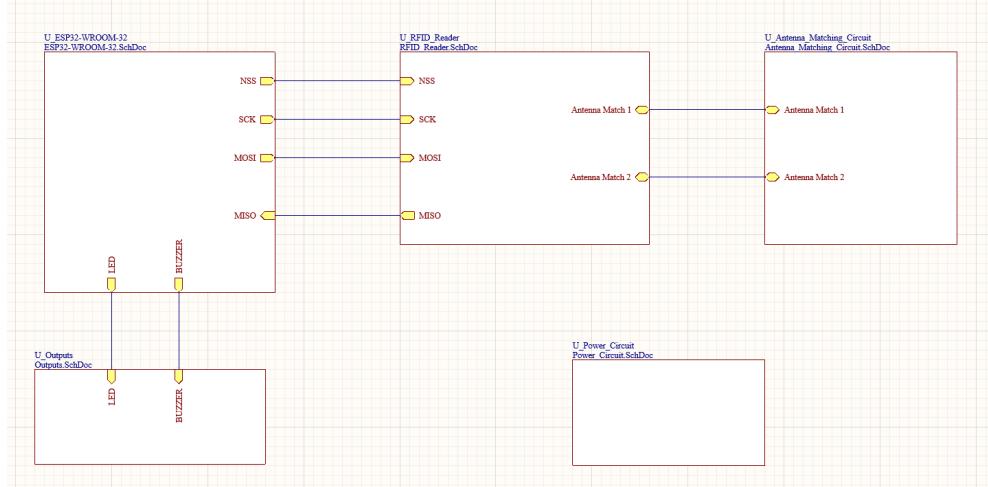


Figure 15: Overview

7 PCB Design (05/04/2024)

Within this subsection, detailed images of the PCB design are presented, showcasing the layout, component placement, and routing strategies employed in the development of the RFID-based asset tracking device. These images provide a visual representation of the circuit design and serve to illustrate the meticulous planning and implementation of the project.

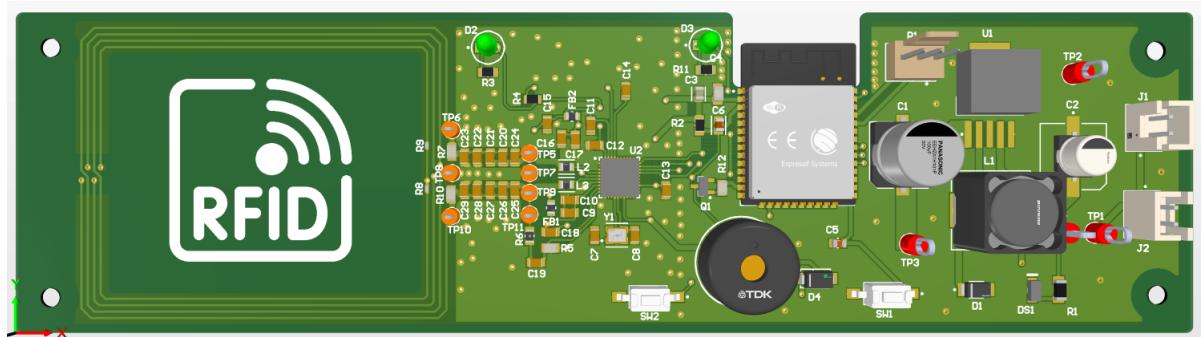
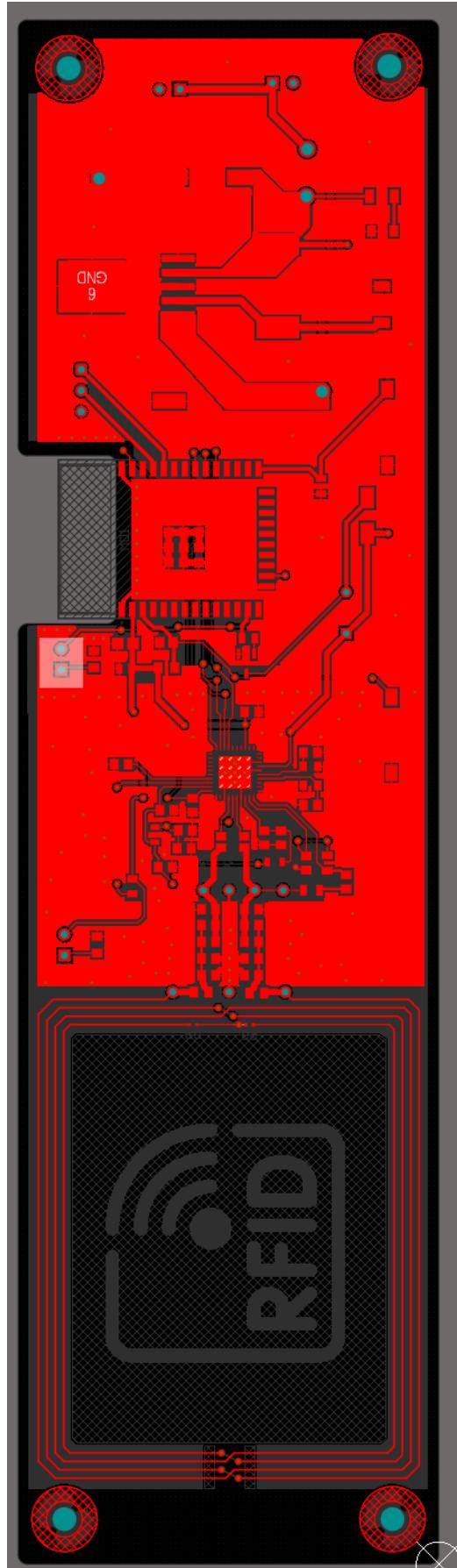
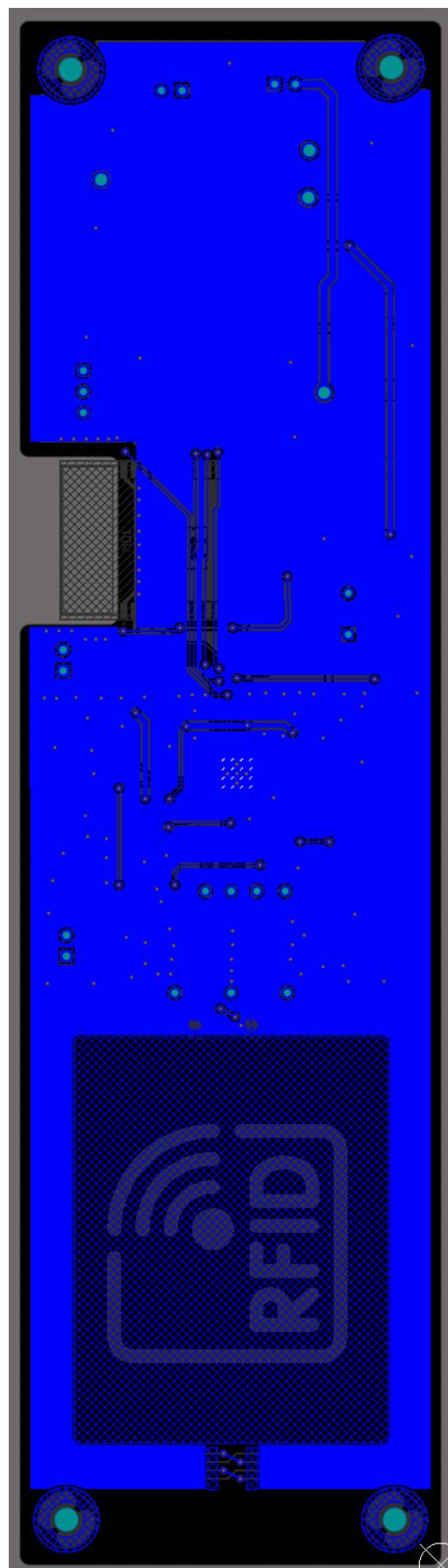


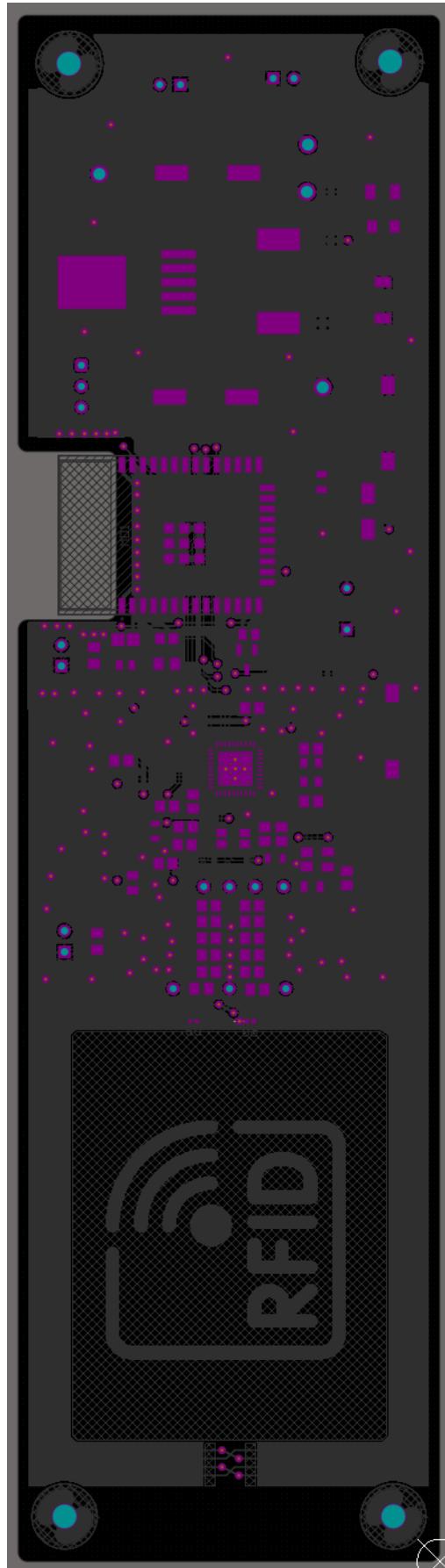
Figure 16: Top View



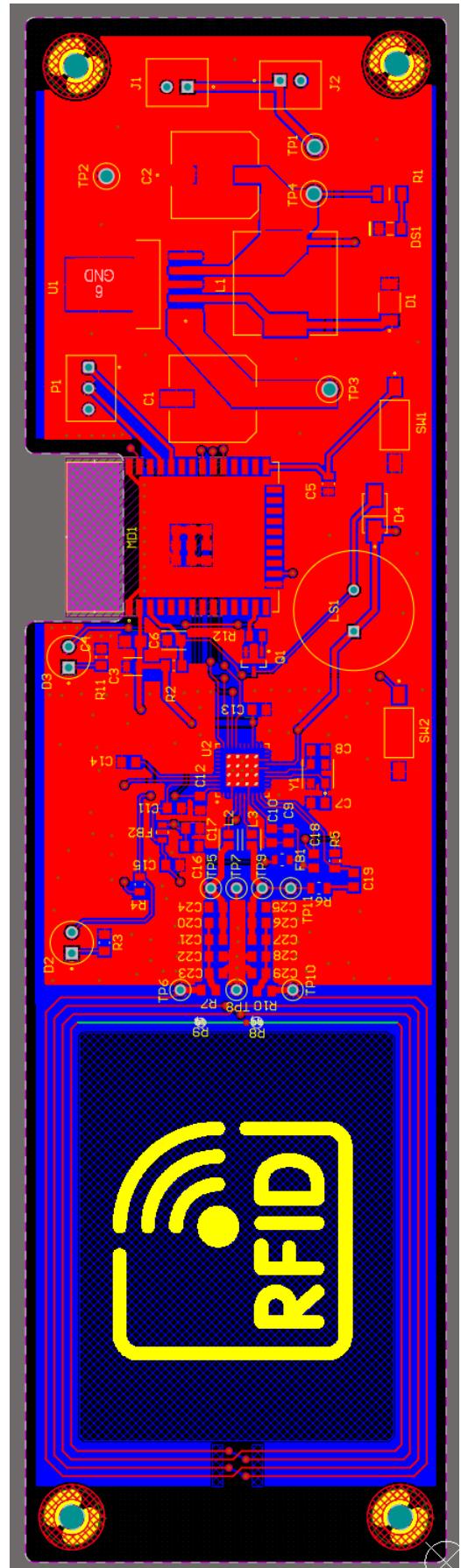
((a)). Top Layer



((b)). Bottom Layer



((a)). Soldermask



((b)). PCB View

8 Bill Of Materials

The Bill of Materials (BOM) is a detailed inventory of all the components, parts, and materials needed for the construction and assembly of the RFID-based asset tracking device for forklifts. Each item listed in the BOM includes a unique designator, a description of the component, and the required quantity. The BOM ensures that all necessary components are accounted for, facilitating efficient procurement and assembly processes. The following table outlines the BOM for this project:

Designator	Description	Quantity
C1	0805 SMD 470uF Capacitor	1
C2	0805 SMD 330uF Capacitor	1
C3, C10, C14	0805 SMD 10uF Capacitor	3
C4, C5, C6, C9, C11, C12, C13	0805 SMD 0.1uF Capacitor	7
C7, C8	0805 SMD 33pF Capacitor	2
C19	0805 SMD 1nF Capacitor	1
C22, C28	0805 SMD 100pF Capacitor	2
C24, C25	0805 SMD 220pF Capacitor	2
C21, C27	0805 SMD 22pF Capacitor	2
R1, R2, R5	0805 SMD 1K Resistor	3
R3, R11	0805 SMD 10R Resistor	2
R4	0805 SMD 10K Resistor	1
R6	0805 SMD 1.69K Resistor	1
R12	0805 SMD 220R Resistor	1
R7, R10	0805 SMD 1.5R Resistor	2
D1	CMS06 - DIODE SCHOTTKY 30V 2A SMD	1
D2	SMD 0805 Blue LED	1
D3	Green through hole LED	1
D4	RS1M Zener Diode SMD	1
DS1	Red through hole LED	1
U1	LM2596S-3.3 SMD	1
U2	PM5321A3HW SMD	1
MD1	ESP32 - WROOM 32	1
Y1	27.12MHz Crystal Oscillator SMD	1
Q1	BC847C Shockley Diode SMD	1
LS1	Buzzer	1
L1	SMD 0805 100uH Inductor	1
L2, L3	SMD 0805 560nH Inductor	2
FB1, FB2	Ferrite Bead 2.7K	2
SW1, SW2	SMD Push Buttons	2
-	2 pin JST	2
-	Female Headers	3

9 Firmware Implementation (12/04/2024)

Firmware implementation is a crucial aspect of embedded systems development, involving the creation and deployment of software that operates directly on hardware devices like microcontrollers or system-on-chips (SoCs). In the context of this introduction, we will focus on the implementation of firmware using the ESP-IDF framework for the ESP32-WROOM32 IC, leveraging UART communication for uploading firmware.

The ESP32-WROOM32 IC is based on the Xtensa LX6 architecture developed by Tensilica (now owned by Cadence Design Systems). The Xtensa architecture is known for its flexibility and configurability, allowing for efficient customization of the processor's instruction set and hardware features to meet specific application requirements.

ESP-IDF (Espressif IoT Development Framework) is a comprehensive platform for developing applications for ESP32 and ESP32-S series SoCs. It provides a rich set of libraries, tools, and APIs tailored for efficient and scalable firmware development.

9.1 Importance of Register Programming

In firmware development, register programming plays a vital role in configuring and controlling hardware peripherals directly at the register level. This approach offers several key advantages:

- **Performance Optimization:** Register programming allows developers to fine-tune hardware settings for optimal performance, often achieving faster data processing and reduced latency compared to higher-level APIs.
- **Resource Efficiency:** By accessing hardware registers directly, firmware can conserve system resources such as memory and processing power, crucial in embedded systems where resource constraints are common.
- **Low-Level Access:** It facilitates precise control over timing, interrupts, and low-level hardware features, essential for tasks like real-time data acquisition, signal processing, and device synchronization.

```
1 #include <stdio.h>
2 #include <string.h>
3 #include "freertos/FreeRTOS.h"
4 #include "freertos/task.h"
5 #include "freertos/event_groups.h"
6 #include "nvs_flash.h"
7 #include "esp_wifi.h"
8 #include "esp_event.h"
9 #include "mqtt_client.h"
10 #include "driver/spi_master.h"
11 #include "pn532.h"
12
13 #define PN532_SCK 5
14 #define PN532_MISO 19
15 #define PN532_MOSI 23
16 #define PN532_SS 18
17
18 #define BUZZER_PIN 14
19 #define INDICATOR_PIN 13
20
21 #define WIFI_SSID "NTWS-4G"
22 #define WIFI_PASS "XXXXXXXXXXXX"
```

```

23 #define MQTT_BROKER "mqtt://test.mosquitto.org"
24
25 static const char *TAG = "RFID_TRACKER";
26
27 static EventGroupHandle_t wifi_event_group;
28 const int WIFI_CONNECTED_BIT = BIT0;
29
30 esp_mqtt_client_handle_t mqtt_client;
31 pn532_t pn532;
32
33 char details[50];
34 char details_long[200];
35 char idcard[50];
36 char idcard_old[50];
37
38 // Function prototypes
39 static void wifi_event_handler(void *arg, esp_event_base_t event_base,
40                               int32_t event_id, void *event_data);
41 static void wifi_init(void);
42 static void mqtt_event_handler(void *handler_args, esp_event_base_t base,
43                               int32_t event_id, void *event_data);
44 static void mqtt_app_start(void);
45 static void setup_pn532(void);
46 static void read_pn532(void);
47 static void indicate(void);
48
49 // Macros for direct register access
50 #define REG_WRITE(addr, val) (*((volatile uint32_t *) (addr))) = (val)
51 #define REG_SET_BIT(addr, bit) REG_WRITE((addr), (REG_READ(addr) | (1ULL
52     << (bit))))
53 #define REG_CLEAR_BIT(addr, bit) REG_WRITE((addr), (REG_READ(addr) & ~(1
54     ULL << (bit))))
55 #define REG_READ(addr) (*((volatile uint32_t *) (addr)))
56
57 // GPIO register addresses and bit masks for ESP32
58 #define GPIO_OUT_REG(pin) (GPIO_REG_BASE + 0x0C + (4 * ((pin) % 32)))
59 #define GPIO_ENABLE_W1TC_REG(pin) (GPIO_REG_BASE + 0x88 + (4 * ((pin) /
60     32)))
61 #define GPIO_ENABLE_W1TS_REG(pin) (GPIO_REG_BASE + 0x78 + (4 * ((pin) /
62     32)))
63
64 // Function to set GPIO pin level
65 static inline void gpio_set_level(uint32_t pin, uint32_t level) {
66     if (level)
67         REG_SET_BIT(GPIO_OUT_REG(pin), pin % 32);
68     else
69         REG_CLEAR_BIT(GPIO_OUT_REG(pin), pin % 32);
70 }
71
72 // Function to set GPIO pin direction
73 static inline void gpio_set_direction(uint32_t pin, uint32_t mode) {
74     if (mode == GPIO_MODE_OUTPUT) {
75         REG_SET_BIT(GPIO_ENABLE_W1TS_REG(pin), pin % 32);
76     } else {
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
837
838
839
839
840
841
842
843
844
845
845
846
847
847
848
849
849
850
851
852
853
853
854
855
855
856
857
857
858
859
859
860
861
861
862
863
863
864
865
865
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1580
1581
1581
1582
1582
1583
1583
1584
1584
1585
1585
1586
1586
1587
1587
1588
1588
1589
1589
1590
1590
1591
1591
1592
1592
1593
1593
1594
1594
1595
1595
1596
1596
1597
1597
1598
1598
1599
1599
1600
1600
1601
1601
1602
1602
1603
1603
1604
1604
1605
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1610
1611
1611
1612
1612
1613
1613
1614
1614
1615
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1620
1621
1621
1622
1622
1623
1623
1624
1624
1625
1625
1626
1626
1627
1627
1628
1628
1629
1629
1630
1630
1631
1631
1632
1632
1633
1633
1634
1634
1635
1635
1636
1636
1637
1637
1638
1638
1639
1639
1640
1640
1641
1641
1642
1642
1643
1643
1644
1644
1645
1645
1646
1646
1647
1647
1648
1648
1649
1649
1650
1650
1651
1651
1652
1652
1653
1653
1654
1654
1655
1655
1656
1656
1657
1657
1658
1658
1659
1659
1660
1660
1661
1661
1662
1662
1663
1663
1664
1664
1665
1665
1666
1666
1667
1667
1668
1668
1669
1669
1670
1670
1671
1671
1672
1672
1673
1673
1674
1674
1675
1675
1676
1676
1677
1677
1678
1678
1679
1679
1680
1680
1681
1681
1682
1682
1683
1683
1684
1684
1685
1685
1686
1686
1687
1687
1688
1688
1689
1689
1690
1690
1691
1691
1692
1692
1693
1693
1694
1694
1695
1695
1696
1696
1697
1697
1698
1698
1699
1699
1700
1700
1701
1701
1702
1702
1703
1703
1704
1704
1705
1705
1706
1706
1707
1707
1708
1708
1709
1709
1710
1710
1711
1711
1712
1712
1713
1713
1714
1714
1715
1715
1716
1716
1717
1717
1718
1718
1719
1719
1720
1720
1721
1721
1722
1722
1723
1723
1724
1724
1725
1725
1726
1726
1727
1727
1728
1728
1729
1729
1730
1730
1731
1731
1732
1732
1733
1733
1734
1734
1735
1735
1736
1736
1737
1737
1738
1738
1739
1739
1740
1740
1741
1741
1
```

```

73     REG_CLEAR_BIT(GPIO_ENABLE_W1TC_REG(pin), pin % 32);
74 }
75 }
76
77 // WiFi event handler
78 static void wifi_event_handler(void *arg, esp_event_base_t event_base,
79                               int32_t event_id, void *event_data) {
80     if (event_base == WIFI_EVENT && event_id == WIFI_EVENT_STA_START) {
81         esp_wifi_connect();
82     } else if (event_base == WIFI_EVENT && event_id ==
83                WIFI_EVENT_STA_DISCONNECTED) {
84         esp_wifi_connect();
85         xEventGroupClearBits(wifi_event_group, WIFI_CONNECTED_BIT);
86     } else if (event_base == IP_EVENT && event_id == IP_EVENT_STA_GOT_IP)
87     {
88         xEventGroupSetBits(wifi_event_group, WIFI_CONNECTED_BIT);
89     }
90 }
91
92 // Initialize WiFi
93 static void wifi_init(void) {
94     esp_err_t ret = nvs_flash_init();
95     if (ret == ESP_ERR_NVS_NO_FREE_PAGES || ret ==
96         ESP_ERR_NVS_NEW_VERSION_FOUND) {
97         ESP_ERROR_CHECK(nvs_flash_erase());
98         ret = nvs_flash_init();
99     }
100    ESP_ERROR_CHECK(ret);

101    wifi_event_group = xEventGroupCreate();
102
103    ESP_ERROR_CHECK(esp_netif_init());
104    ESP_ERROR_CHECK(esp_event_loop_create_default());
105    esp_netif_create_default_wifi_sta();

106    wifi_init_config_t cfg = WIFI_INIT_CONFIG_DEFAULT();
107    ESP_ERROR_CHECK(esp_wifi_init(&cfg));

108    esp_event_handler_instance_t instance_any_id;
109    esp_event_handler_instance_t instance_got_ip;
110    ESP_ERROR_CHECK(esp_event_handler_instance_register(WIFI_EVENT,
111                                                       ESP_EVENT_ANY_ID,
112                                                       &
113                                                       wifi_event_handler,
114                                                       NULL, &
115                                                       instance_any_id));
116
117    ESP_ERROR_CHECK(esp_event_handler_instance_register(IP_EVENT,
118                                                       IP_EVENT_STA_GOT_IP,
119                                                       &
120                                                       wifi_event_handler,
121                                                       NULL, &
122                                                       instance_got_ip));
123

```

```

114     wifi_config_t wifi_config = {
115         .sta = {
116             .ssid = WIFI_SSID,
117             .password = WIFI_PASS,
118         },
119     };
120 }
121 ESP_ERROR_CHECK(esp_wifi_set_mode(WIFI_MODE_STA));
122 ESP_ERROR_CHECK(esp_wifi_set_config(ESP_IF_WIFI_STA, &wifi_config));
123 ESP_ERROR_CHECK(esp_wifi_start());
124
125 xEventGroupWaitBits(wifi_event_group, WIFI_CONNECTED_BIT, pdFALSE,
126                     pdTRUE, portMAX_DELAY);
127
128 // MQTT event handler
129 static void mqtt_event_handler(void *handler_args, esp_event_base_t base,
130                               int32_t event_id, void *event_data) {
131     esp_mqtt_event_handle_t event = event_data;
132     switch (event->event_id) {
133         case MQTT_EVENT_CONNECTED:
134             esp_mqtt_client_subscribe(mqtt_client, "210610H-CardDetails",
135                                         0);
136             break;
137         case MQTT_EVENT_DISCONNECTED:
138             break;
139         default:
140             break;
141     }
142 }
143
144 // Start MQTT client
145 static void mqtt_app_start(void) {
146     esp_mqtt_client_config_t mqtt_cfg = {
147         .uri = MQTT_BROKER,
148     };
149     mqtt_client = esp_mqtt_client_init(&mqtt_cfg);
150     esp_mqtt_client_register_event(mqtt_client, ESP_EVENT_ANY_ID,
151                                     mqtt_event_handler, NULL);
152     esp_mqtt_client_start(mqtt_client);
153 }
154
155 // Setup PN532 module
156 static void setup_pn532(void) {
157     spi_bus_config_t buscfg = {
158         .misio_io_num = PN532_MISO,
159         .mosi_io_num = PN532_MOSI,
160         .sclk_io_num = PN532_SCK,
161         .quadwp_io_num = -1,
162         .quadhd_io_num = -1,
163         .max_transfer_sz = 0,
164     };
165     spi_device_interface_config_t devcfg = {
166         .clock_speed_hz = 1 * 1000 * 1000, // 1 MHz

```

```

165     .mode = 0,
166     .spics_io_num = PN532_SS,
167     .queue_size = 7,
168 };
169
170 ESP_ERROR_CHECK(spi_bus_initialize(HSPI_HOST, &buscfg, 1));
171 ESP_ERROR_CHECK(spi_bus_add_device(HSPI_HOST, &devcfg, &pn532.
172     spi_handle));
173
174 pn532_init(&pn532);
175 uint32_t versiondata = pn532_get_firmware_version(&pn532);
176 if (!versiondata) {
177     ESP_LOGE(TAG, "Didn't find PN53x board");
178     while (1);
179 }
180
181 pn532_set_passive_activation_retries(&pn532, 0xFF);
182 ESP_LOGI(TAG, "PN532 setup complete");
183 }
184
185 // Read data from PN532
186 static void read_pn532(void) {
187     uint8_t success;
188     uint8_t uid[7]; // Buffer to store the returned UID
189     uint8_t uidLength;
190     success = pn532_read_passive_target(&pn532, PN532_MIFARE_IS014443A,
191         uid, &uidLength, 0);
192     if (success) {
193         idcard_old[0] = '\0';
194         strcpy(idcard_old, idcard);
195
196         idcard[0] = '\0';
197         for (uint8_t i = 0; i < uidLength; i++) {
198             char hex[3];
199             sprintf(hex, "%02X", uid[i]);
200             strcat(idcard, hex);
201         }
202
203         if (strcmp(idcard, idcard_old) != 0) {
204             snprintf(details, sizeof(details), "%s", idcard);
205             esp_mqtt_client_publish(mqtt_client, "210610H-CardDetails",
206                 details, 0, 1, 0);
207             indicate();
208         }
209     }
210 }
211
212 // Indicate action with LEDs and buzzer
213 static void indicate(void) {
214     gpio_set_level(BUZZER_PIN, 1);
215     gpio_set_level(INDICATOR_PIN, 1);
216     vTaskDelay(400 / portTICK_PERIOD_MS);
217     gpio_set_level(BUZZER_PIN, 0);
218     gpio_set_level(INDICATOR_PIN, 0);

```

```
216 }
217
218 // Main application entry point
219 void app_main(void) {
220     esp_log_level_set("*", ESP_LOG_INFO);
221
222     // Initialize GPIO pins using direct register access
223     gpio_set_direction(BUZZER_PIN, GPIO_MODE_OUTPUT);
224     gpio_set_direction(INDICATOR_PIN, GPIO_MODE_OUTPUT);
225
226     // Initialize WiFi, MQTT, and PN532 module
227     wifi_init();
228     mqtt_app_start();
229     setup_pn532();
230
231     // Main loop to read from PN532 and handle MQTT communication
232     while (1) {
233         read_pn532();
234         vTaskDelay(1000 / portTICK_PERIOD_MS);
235     }
236 }
```

Listing 1: C Code for programming ESP32 IC - main.c

The CMakeLists.txt file is a configuration file used by CMake, a cross-platform build system generator. CMake automates the process of configuring build environments, making it easier to manage large projects and their dependencies. When working with the ESP-IDF (Espressif IoT Development Framework) for ESP32, CMakeLists.txt is used to specify how the project should be built.

```
# In main/CMakeLists.txt

idf_component_register(SRCS "main.c" "pn532.c"
                      INCLUDE_DIRS "."
                      REQUIRES esp_netif esp_wifi mqtt spi_master)

1           /your_project
2               CMakeLists.txt
3               sdkconfig
4               main
5                   CMakeLists.txt
6                   main.c
7                   pn532.c
8                   pn532.h
9               components
10              Adafruit-PN532
11                  Adafruit_PN532.cpp
12                  Adafruit_PN532.h
13                  ...
14                  library.properties
```

Listing 2: Project Directory Structure in ESP-IDF

1. `setupWifi()`: This function is responsible for setting up the WiFi connection. It attempts to connect to a WiFi network with the SSID "NTWS-4G" and the password "XXXXXXXXXXXXXX". It waits in a loop until the connection is established.
2. `setupMqtt()`: Sets up the MQTT client with the MQTT broker's address and port. It initializes the `PubSubClient` object `mqttClient` to communicate with the MQTT broker at "test.mosquitto.org" on port 1883.
3. `setupPn532()`: Initializes the PN532 NFC module. It starts the PN532 module and checks the firmware version. If the firmware version cannot be retrieved, the function enters an infinite loop, halting further execution.
4. `readPn532()`: Reads the NFC tag using the PN532 module. It checks if a card is present and reads its unique identifier (UID). If a new card is detected (UID is different from the previous one), it converts the UID to a string and publishes it to the MQTT topic "210610H-CardDetails". It also calls the `Indicate()` function to indicate a successful read.
5. `connectToBroker()`: Attempts to connect to the MQTT broker. If the connection is successful, it publishes a message to the MQTT topic "210610H-Details" indicating that the RFID-based asset tracking device is connected. If the connection fails, it retries after a delay of 5 seconds.
6. `Indicate()`: This function indicates a successful NFC tag read. It activates a buzzer and LED for a short duration to provide visual and auditory feedback.

9.2 Compatibility Issue Between the Adafruit PN532 library and ESP32 IC

When implementing the software component, I encountered an issue where the PN532 board was not being properly detected by the ESP32 board, despite correct wiring. Upon investigation, I identified this as a problem stemming from the Adafruit PN532 library.

The specific issue with the Adafruit PN532 library pertains to the communication bitrate configuration when used with an ESP32 board. By default, the library sets the bitrate to 1000000 (1 MHz), which typically functions adequately. However, when the ESP32 is operating at a clock speed of 240 MHz, communication becomes erratic and prone to glitches.

To mitigate this issue, it is necessary to adjust the bitrate in the Adafruit PN532 library to 100000 (100 kHz) when the ESP32 is running at 240 MHz. This lower bitrate stabilizes communication and prevents glitches from occurring. This indicates a potential need for optimization or adjustment in the library to ensure compatibility with higher clock speeds.

So you need to add the following code to the Adafruit_PN532 library to resolve that problem.

```
1 Adafruit_PN532::Adafruit_PN532(uint8_t ss) {
2
3     #ifdef ESP32
4         uint32_t frequency = 100000;
5     #else
6         uint32_t frequency = 1000000;
7     #endif
8
9     spi_dev = new Adafruit_SPIDevice(ss, frequency, SPI_BITORDER_LSBFIRST,
10                                     SPI_MODE0);
11 }
```

Listing 3: Additional code for Resolve the issue

10 Enclosure Design (08/03/2024)

This section showcases the SolidWorks designs of the enclosure for the RFID-based asset tracking device, presenting detailed images of the top, bottom, and assembly views. These images highlight the design's aesthetics, functionality, and compatibility with the forklift, providing insights into the enclosure's robustness and suitability for industrial use.

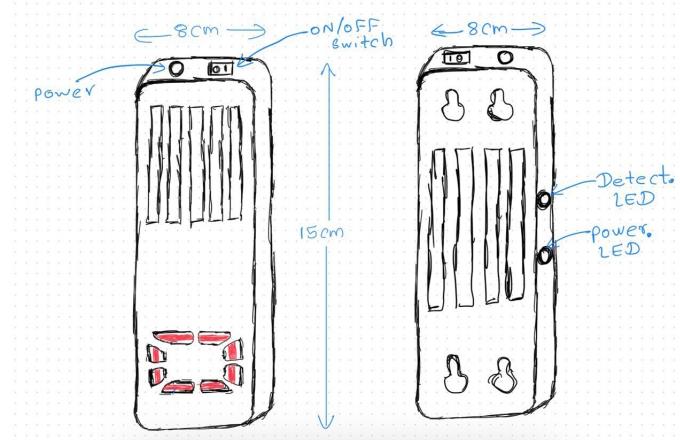


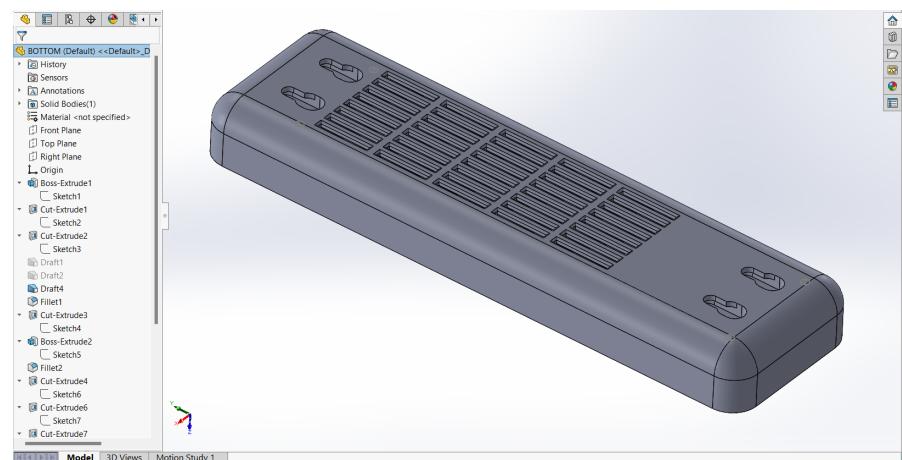
Figure 19: Handsketch

10.1 Upper Housing Component





10.2 Lower Chassis Component



10.3 Enclosure Integration System



11 Other Components

11.1 ON OFF Switch

To enhance the functionality and user experience of the device, an on/off switch has been integrated into the design. This switch allows users to easily power the device on or off.



lizardleds

Figure 20: ON OFF Switch

11.2 Power Input

A DC barrel female socket has been integrated into the device as a power input enhancement. This feature enables direct input of 12V from a forklift battery, ensuring a dependable and uninterrupted power supply. Users are required to utilize a compatible barrel connector for this purpose.



Figure 21: Base Barrel Connector

12 PCB Manufacturing (03/05/2024)

PCB was manufactured by JLC PCB.

12.1 PCB Specifications

- **Layers:** 2
- **Dimension:** 186.8 mm x 50.8 mm
- **Product Type:** Industrial/Consumer Electronics
- **PCB Thickness:** 0.6 mm
- **Material Type:** FR4-Standard TG 135-140
- **Via Covering:** Tented
- **Surface Finish:** HASL (with lead)
- **Outer Copper Weight:** 1 oz

13 Bare PCB

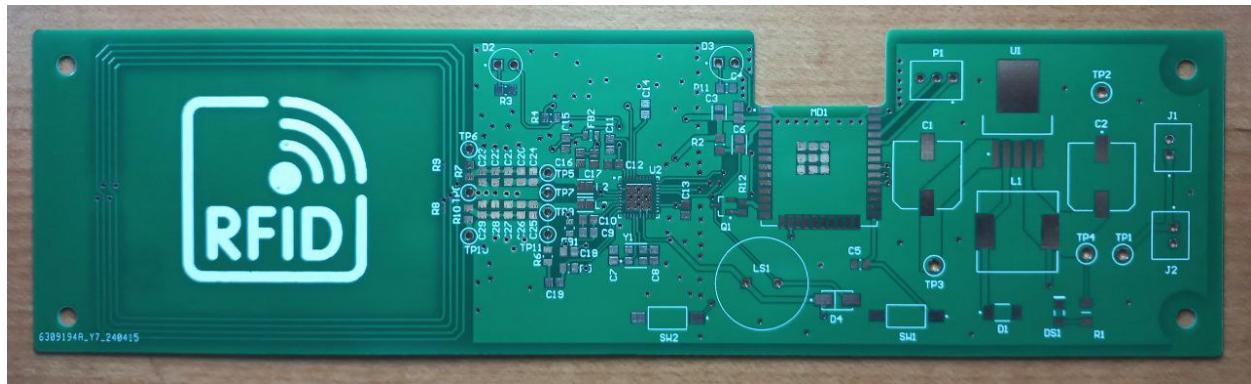


Figure 22: Upside View

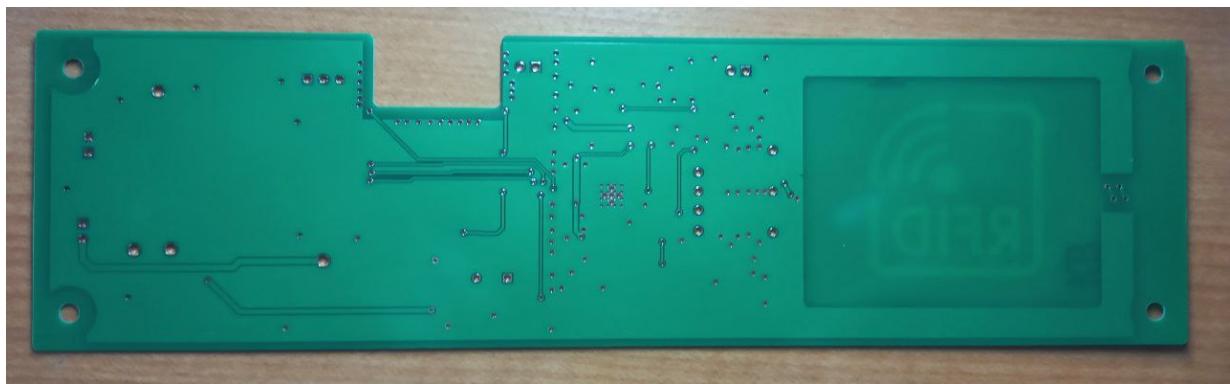


Figure 23: Downside View

13.1 Soldered PCB



Figure 24: Upside View

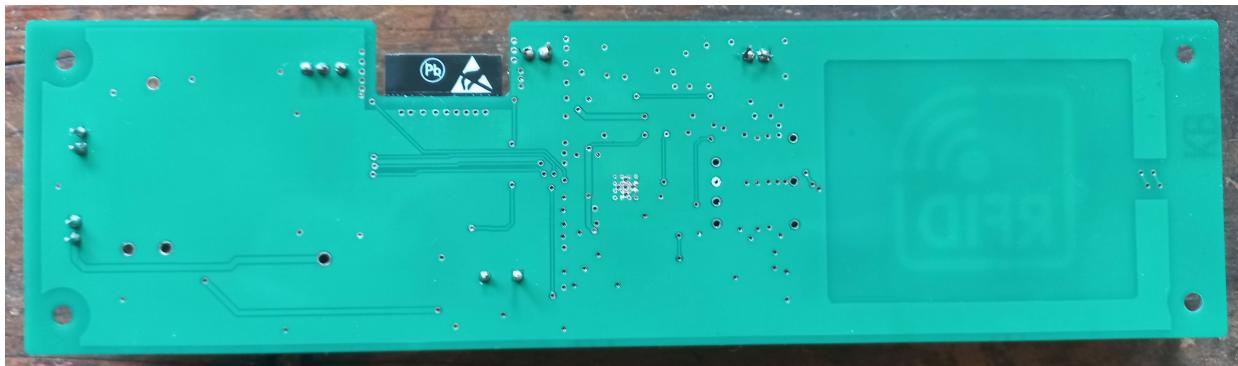


Figure 25: Downside View

14 PCB Testing

This task has not yet been completed due to the inability to use the workshop since it is closed due to the non-academic staff strike. This will be completed, and the relevant documentation will be updated once it is done.



15 Physically Build Enclosure





Figure 28: Printed Enclosure

16 System Integration

To be done.

17 Product Specifications

- **Input Voltage:** 12 V (From Forklift Battery)
- **Operating Voltage:** 3.3 V
- **Reading Distance (max):** (To be tested)
- **Maximum Power:** (To be tested)
- **RFID Operating Frequency:** 13.56MHz
- **Minimum Buzzer Sound Pressure Level:** (To be tested)
- **Operating Ambient Temperature:** (To be tested)
- **Wi-Fi Connectivity:** Wi-Fi 2.4GHz
- **Product Dimensions:** 210 × 52 × 48 mm

18 Future Improvements

In future iterations of my RFID-based asset tracking device, I plan to integrate more advanced RFID readers with enhanced capabilities. These readers will support higher read ranges and faster processing speeds, enabling more efficient asset tracking even in larger and more complex warehouse environments. Additionally, I aim to implement multi-frequency RFID readers that can simultaneously operate in different RFID frequency bands, providing greater flexibility and compatibility with a wider range of RFID tags. This enhancement will significantly improve the overall performance and reliability of the asset tracking system.

To further leverage the data collected by my RFID system, I plan to develop advanced data analytics and reporting features. By incorporating machine learning algorithms and data visualization tools, my system will be able to provide deeper insights into asset usage patterns, inventory levels, and operational efficiency. Users will benefit from customizable dashboards and real-time alerts, allowing them to make informed decisions and optimize their warehouse operations. These features will not only enhance the utility of the asset tracking device but also provide actionable intelligence for better resource management and cost reduction.

As part of my vision for future developments, I aim to integrate my RFID-based asset tracking device with broader IoT and Industry 4.0 ecosystems. By connecting my device to other smart sensors and automation systems within the warehouse, I can create a more cohesive and intelligent asset management solution. This integration will enable seamless communication and interoperability between different systems, leading to improved automation, predictive maintenance, and enhanced operational efficiency. Ultimately, my goal is to transform the asset tracking device into a key component of a fully integrated, smart warehouse environment.

19 Acknowledgement

I deeply appreciate and am grateful to everyone who has played a significant role in the development of this exceptional product. The successful completion of this design document would not have been possible without the collective efforts and support from all contributors.

Initially, I would like to extend my sincere thanks to my lecturer for his unwavering encouragement and support throughout this project. His dedication to innovation and strategic guidance has been instrumental in shaping my ideas and driving me toward success.

We also thank the members of our main project group for their support. Their continuous encouragement, rigorous testing, and meticulous attention to detail have ensured that our product meets the highest standards of reliability and performance.

20 Data sheets and References

1. Case study of smart forklift integration with RFID - (<https://www.turck.nl/static/media/downloads/more22254e.pdf>).
2. An RFID data-based forklift traffic management system by Research gate - (https://www.researchgate.net/figure/An-RFID-data-based-forklift-traffic-management-system_fig6_228657522).
3. PN532/C1 datasheet - (https://www.nxp.com/docs/en/nxp/data-sheets/PN532_C1.pdf).
4. PN532 and MFRC522 Antenna Design Guideline - (<https://community.nxp.com/pwmxy87654/attachments/pwmxy87654/nfc/3055/1/AN1445-2010%20-%20Antenna%20design%20guide%20for%20MFRC52x,%20PN51x,%20PN53x.pdf>).
5. TRF79xxA datasheet - (<https://pdf1.alldatasheet.com/datasheet-pdf/download/1244246/TI1/TRF79XXA.html>).
6. Antenna Design Guide for the TRF79xxA - (<https://www.ti.com/lit/pdf/sloa241>).
7. ESP32 WROOM 32 datasheet - (https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf).
8. LM2596S-3.3 datasheet - (<https://www.ti.com/lit/ds/symlink/lm2596.pdf>).

21 Appendix - Daily Log Entries

- **29 January - 4 February** At the outset of the Electronic Design Realization module, my first task was to select a project. After thorough research into various industrial project scopes, I finalized three potential projects. Following a presentation to our lecturer, I were pleased to be accepted for my project based on RFID technology. This marked the beginning of my journey into RFID technology, where I sought to understand its basic principles and real-world applications. I explored RFID's role in warehouse management, gaining valuable insights into its potential uses and benefits.
- **5 February - 9 February** This week, I focused on deepening our understanding of RFID technology, particularly its applications in warehouse management. Our lecturer provided us with a unique opportunity to visit operational warehouses that use RFID devices for managing goods and activities. This firsthand experience inspired us to delve deeper into the technology, studying existing RFID devices and systems used in warehouses. Despite the widespread use of RFID in foreign warehouses, we observed a scarcity of RFID technology in Sri Lanka. This realization motivated us to explore the potential for introducing RFID technology in local warehouse management systems.
- **10 February - 23 February** As part of a main group tasked with developing a comprehensive RFID solution for warehouse management, we divided ourselves into three specialized teams: forklift reader, portal reader, and handheld reader. I were assigned to develop the RFID based forklift reader. My focus shifted towards researching existing forklift RFID readers, analyzing their features, strengths, and weaknesses. This analysis provided valuable insights into the improvements and innovations needed in forklift RFID reader technology.
- **24 February - 1 April** During this period, I focused on several key aspects of my RFID Reader Design 3 project.
 - **RFID Chip Selection (24/02/2024):**
I researched various RFID chips, including the NXP MFRC522, NXP PN532, and Texas TRF7970A, evaluating their features and specifications. After careful consideration, I selected the NXP PN532 as the most suitable chip for my project due to its compatibility with different RFID frequencies and its robust performance.
 - **RFID Reader Circuit Diagram (26/02/2024):**
I created a detailed circuit diagram for the RFID reader, outlining the connections and components required for the project. This diagram served as a crucial reference point for the assembly and testing phases of the project.
 - **Enclosure Design (08/03/2024):**
I worked on the design of the enclosure for the RFID reader, focusing on the upper housing component and the lower chassis component. Additionally, I developed an enclosure integration system to ensure the components fit together securely and efficiently.
 - **Antenna and Matching Circuit Design (11/03/2024):**
I designed the antenna and matching circuit for the RFID reader, ensuring optimal performance and compatibility with the chosen RFID chip. This involved determining the equivalent circuit and creating a circuit diagram that integrated the antenna, matching circuit, and EMC filter.
 - **How to Access WiFi Connection, ESP32 WROOM 32 (25/03/2024):**
I researched and implemented methods to access the WiFi connection using the ESP32 WROOM 32. This involved establishing SPI communication with the ESP32 and programming the IC to enable WiFi connectivity. I also finalized the circuit diagram for the ESP32 WROOM 32, ensuring seamless integration with the RFID reader circuit.

These details are included on the following pages:

- RFID Chip Selection: Page 3
- RFID Reader Circuit Diagram: Page 6

- Antenna and Matching Circuit Design: Page 7
- How to Access WiFi Connection, ESP32 WROOM 32: Page 9
- Enclosure Design: Pages 19-20

- **2 April - 26 April** During this period, I focused on the power circuit design and output circuit design for my RFID Reader Design for forklift project.

- **Power Circuit Design (02/04/2024):**

I designed the power circuit for the RFID reader, incorporating a voltage regulator LM2596S-3.3 to ensure stable power supply to the components. This design was crucial for maintaining the proper functioning of the RFID reader.

- **Output Circuit Design (04/04/2024):**

I designed the output circuit for the RFID reader, including circuits for a buzzer and LED indicator. These circuits were essential for providing feedback to the user regarding the RFID reading process, ensuring smooth operation of the device.

- **Software Implementation (12/04/2024):**

I encountered a compatibility issue between the Adafruit PN532 library and the ESP32 IC during the software implementation phase. This required me to troubleshoot the issue and find a workaround to ensure proper communication between the RFID reader and the ESP32 IC.

The details for each section are included on the following pages:

- Power Circuit Design: Page 11
- Output Circuit Design: Page 13
- Software Implementation: Page 16

- **3 July - 7 July** Initially, I developed the firmware for our project using the Arduino IDE. The Arduino IDE is known for its user-friendly interface, making it popular among hobbyists and for rapid prototyping. However, after consulting with our professor, we realized that while the Arduino IDE simplifies coding, it is not considered a professional tool for engineering-grade projects. It abstracts many lower-level operations, which can limit the control and optimization necessary for more complex and high-performance applications.

ESP-IDF provides a more robust development environment, allowing for finer control over the hardware through register programming. This approach not only enhances performance but also aligns with best practices in embedded systems engineering.

Previously, my programming approach did not involve direct register manipulation, relying instead on high-level functions provided by the Arduino framework. With the switch to ESP-IDF, I have learned to engage in register-level programming, which provides greater flexibility and efficiency, crucial for our project's requirements.

For reference, I have included the initial code implemented using the Arduino IDE in the appendix below.

```
1 #include <SPI.h>
2 #include <Adafruit_PN532.h>
3 #include <WiFi.h>
4 #include <PubSubClient.h>
5
6 #define PN532_SCK  (5) // SCK (Serial Clock)
7 #define PN532_MISO (19) // MISO (Master In Slave Out)
```

```

8 #define PN532_MOSI (23) // MOSI (Master Out Slave In)
9 #define PN532_SS (18) // SS (Slave Select)
10
11 #define Buzzer 14
12 #define Indicator 13
13
14 int size1 = 50;
15 int size2 = 200;
16
17 char details[40];
18 char detailsLong[250];
19
20 WiFiClient espClient;
21 PubSubClient mqttClient(espClient);
22
23 Adafruit_PN532 nfc(PN532_SCK, PN532_MISO, PN532_MOSI, PN532_SS);
24 String idcard;
25 String idcardOld;
26
27 void setup() {
28   Serial.begin(115200);
29
30   setupWifi();
31
32   setupMqtt();
33
34   setupPn532();
35
36   pinMode(Buzzer, OUTPUT);
37   digitalWrite(Buzzer, LOW);
38
39   pinMode(Indicator, OUTPUT);
40   digitalWrite(Indicator, LOW);
41 }
42
43 void loop() {
44
45   if (!mqttClient.connected()){
46     connectToBroker();
47   }
48
49   mqttClient.loop();
50
51   readPn532();
52 }
53
54 void setupWifi() {
55
56   WiFi.begin("NTWS-4G", "XXXXXXXXXXXXXX");
57
58   while (WiFi.status() != WL_CONNECTED) {
59     delay(500);
60   }
61 }
```

```

62
63 void setupMqtt() {
64     mqttClient.setServer("test.mosquitto.org", 1883);
65 }
66
67 void setupPn532(){
68     nfc.begin();
69     uint32_t versiondata = nfc.getFirmwareVersion();
70     if (!versiondata) {
71         while (1); // halt
72     }
73
74     nfc.setPassiveActivationRetries(0xFF);
75 }
76
77 void readPn532(){
78     uint8_t success;
79     uint8_t uid[] = { 0, 0, 0, 0, 0, 0, 0, 0 }; // Buffer to store the
80     // returned UID
81     uint8_t uidLength;
82     success = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A, uid, &
83     uidLength);
84     if (success) {
85         // Display some basic information about the card
86
87         idcardOld = idcard;
88
89         idcard = "";
90         for (byte i = 0; i <= uidLength - 1; i++) {
91             idcard += (uid[i] < 0x10 ? "0" : "") + String(uid[i], HEX);
92         }
93
94         if (idcard != idcardOld){
95             String(idcard).toCharArray(details, size1);
96             mqttClient.publish("210610H-CardDetails", details);
97             Indicate();
98         }
99     }
100 }
101
102 void connectToBroker(){
103     while (!mqttClient.connected()){
104
105         if (mqttClient.connect("210610H-forklift")){
106             String("RFID based asset tracking device CONNECTED").toCharArray
107                 (details, size2);
108             mqttClient.publish("210610H-Details", details);
109         }
110     }
111 }
112

```

```
113 void Indicate(){  
114     tone(Buzzer, 400);  
115     digitalWrite(Indicator, HIGH);  
116     delay(400);  
117  
118     noTone(Buzzer);  
119     digitalWrite(Indicator, LOW);  
120 }
```

Listing 4: Arduino Code for program ESP32 IC

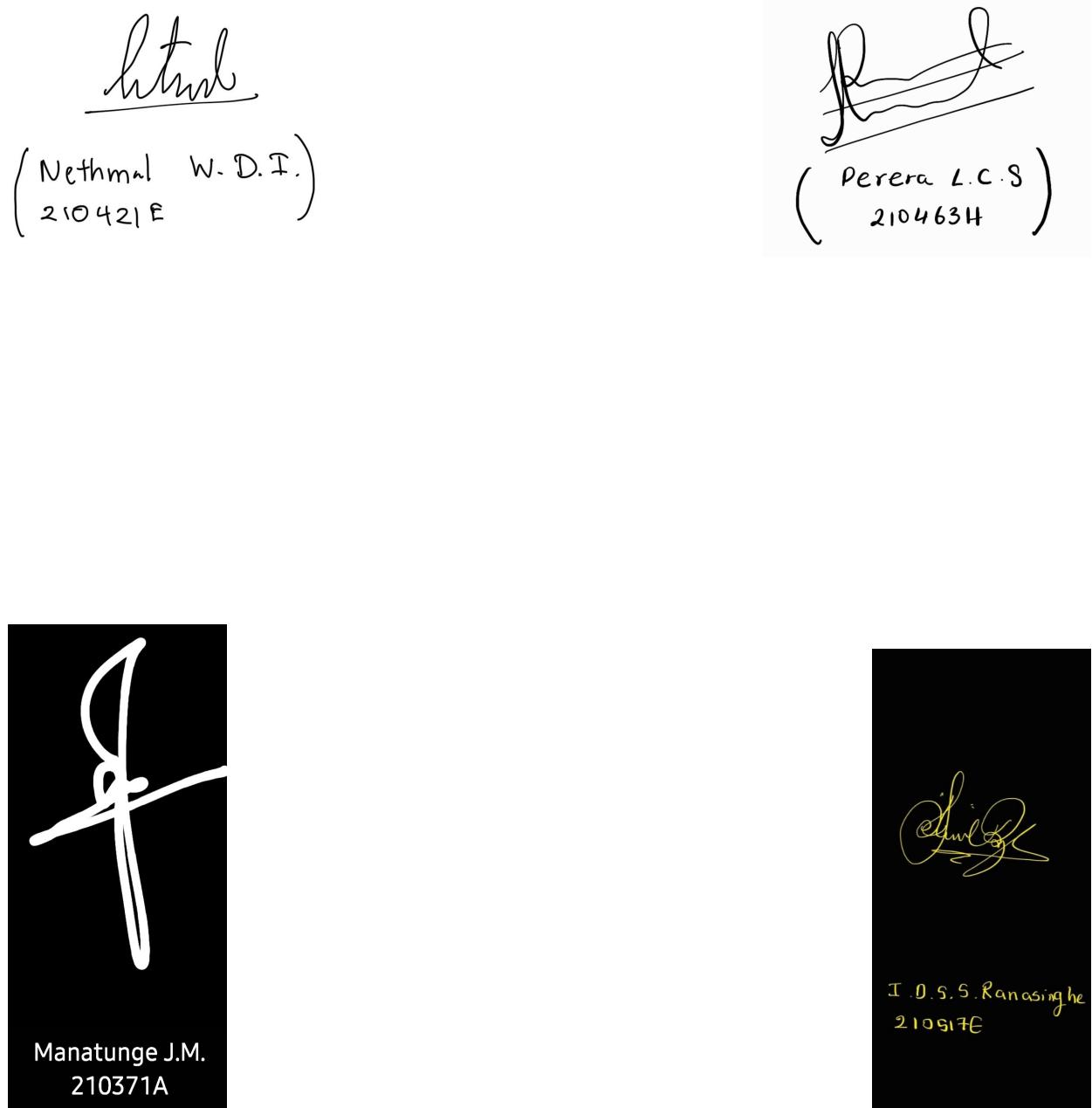


Figure 30: Signs Of Other Members Of Main Group