**RFID BASED INVENTORY CONTROL**

**1. Introduction**

The RFID-based inventory control system is a project that leverages RFID technology to automate inventory management processes. It utilizes an RFID scanner, an Arduino Uno microcontroller board, and various other components to enable efficient tracking and monitoring of inventory items. By reading RFID cards and retrieving customized data stored within them, the system provides real-time information about the quantity of items present in the inventory. This documentation outlines the materials used, the purpose of the project, how it works, and potential practical use case scenarios.

**2. Materials Used**

The following materials were used to construct the RFID-based inventory control system:

* Arduino Uno: The Arduino Uno is a microcontroller board based on the ATmega328P. It serves as the main control unit for the project, facilitating communication between the RFID scanner, LCD display, LEDs, and buzzer.
* RFID Scanner (RC522) : The RFID scanner is responsible for reading the RFID cards placed in its proximity. It communicates with the Arduino Uno to exchange data.
* RFID Cards: RFID cards are embedded with unique identification information and additional data storage capacity. The customized data at position 2 of each RFID card represents the quantity of an item in the inventory.
* LCD Display: The LCD (Liquid Crystal Display) module is connected to the Arduino Uno and provides a visual output for displaying inventory information.
* Green and Red LEDs: The green LED serves as a visual indicator of a successful inventory check, while the red LED indicates an unsuccessful check.
* Buzzer: The buzzer is an optional component that can be used to provide an auditory indication along with the red LED for an unsuccessful inventory check.
* Jumper Wires: Jumper wires are used to establish electrical connections between the components.
* Breadboard: The breadboard serves as a platform for prototyping and connecting the components securely.
* Resistors: Resistors may be required to limit the current flow through the LEDs or other components, depending on their specifications.
* Power Supply: The components are powered using a common ground (GND) and voltage (VCC) connection.

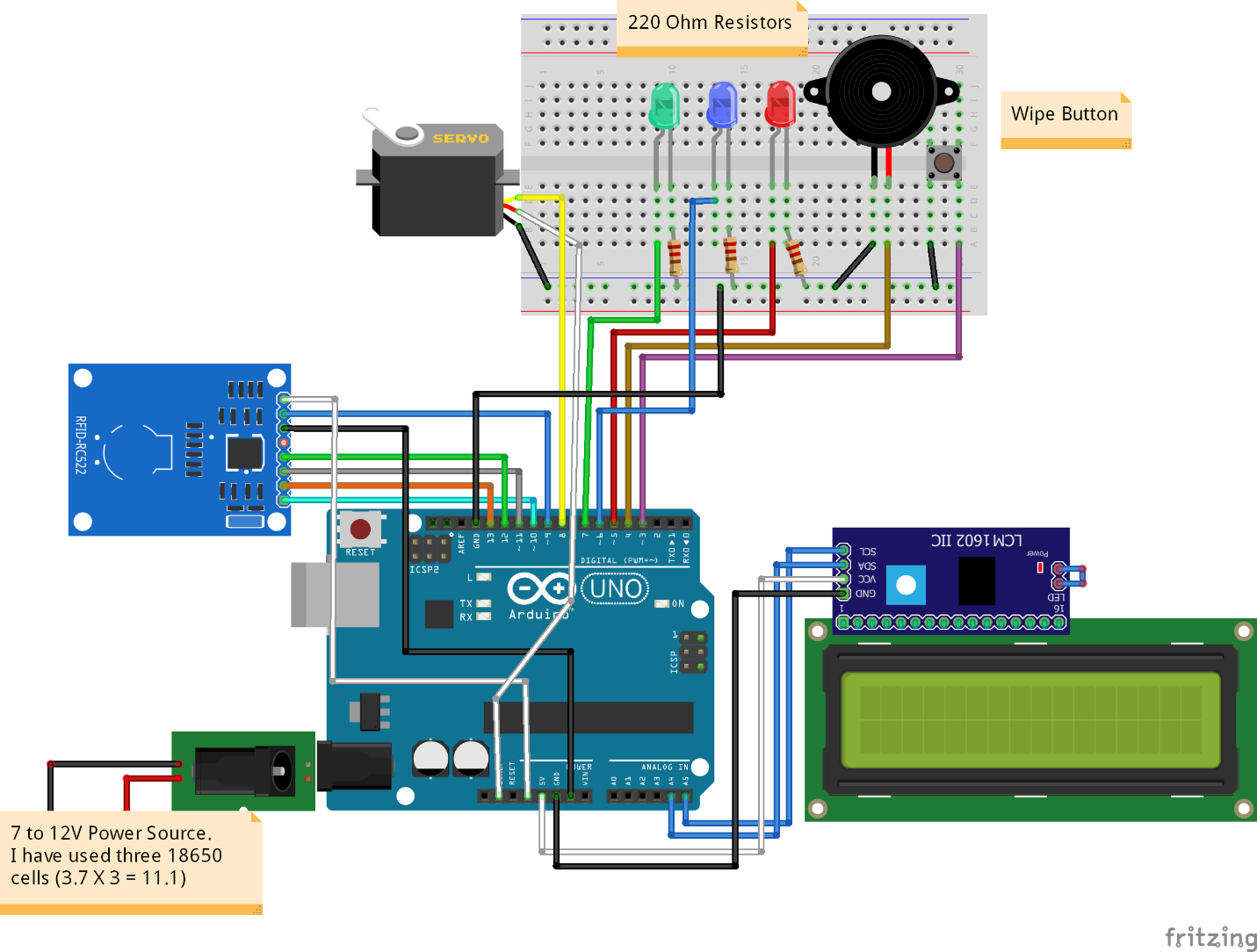
**3. Purpose**

The purpose of the RFID-based inventory control system is to automate and streamline the inventory management process. Traditional manual inventory tracking methods can be time-consuming and prone to errors. By employing RFID technology and an Arduino-based control system, this project aims to provide an efficient and accurate solution for inventory control. The system reads RFID cards, retrieves specific data representing the quantity of items in the inventory, displays the information on an LCD screen, and provides visual and auditory feedback to the user.

**4. How It Works**

The RFID-based inventory control system operates through the following steps:

1. RFID Card Scanning: The RFID scanner is placed in close proximity to an RFID card containing inventory information. The scanner emits radio waves that power the card and communicate with it.
2. Data Retrieval: The RFID scanner reads the data stored in position 2 of the RFID card. This data corresponds to the quantity of the item present in the inventory at a specific position.
3. Inventory Check: The obtained data is then processed by the Arduino Uno, which performs the necessary calculations to determine the quantity of the item in the inventory.
4. LCD Display: The Arduino Uno communicates with the LCD display module using the I2C protocol. The inventory information, such as the quantity of the item, is displayed on the LCD screen in a user-friendly format.
5. Visual and Auditory Indicators: Based on the inventory check result, the system provides visual and auditory feedback. If the check is successful and the item is available in the desired quantity, a green LED lights up. In the case of an unsuccessful check, indicating a shortage or absence of the item, a red LED turns on, accompanied by the activation of a buzzer (if present).
6. Pin Configuration: The various components are connected to specific digital pins on the Arduino Uno. The RFID scanner pins (SDA, SCK, MOSI, MISO, and RST) are connected to the corresponding Arduino pins (10, 13, 11, 12, and 5). The LCD display's SCL and SDA pins are connected to the Arduino's A5 and A4 pins, respectively. The green LED is connected to Pin 6, the red LED to Pin 7, and the buzzer to a designated pin on the Arduino (as per the specific module used).



**5. Practical Use Case Scenarios**

The RFID-based inventory control system has several practical use case scenarios, including but not limited to:

1. Retail Shops: The system can be implemented in retail shops to keep track of products in real-time. It ensures accurate inventory management by providing immediate information about the availability of items and their quantities. This helps shop owners restock products efficiently and avoid stockouts.
2. Warehouses: Implementing the system in warehouses enables efficient inventory tracking. It allows warehouse managers to monitor the quantity and location of items within the warehouse, facilitating streamlined logistics and reducing manual effort.
3. Libraries: Libraries can utilize the system to automate the tracking of books or other media items. By scanning RFID cards embedded in library materials, librarians can easily keep track of borrowed and returned items, simplifying inventory management and enhancing the overall library experience.
4. Manufacturing Facilities: The system can be deployed in manufacturing facilities to monitor the inventory of components or raw materials. It helps ensure that an adequate quantity of materials is available for production, minimizing production delays and optimizing supply chain management.

**6. Pin Configuration**

The following table outlines the pin configuration for the RFID-based inventory control system:

| **Equipment** | **Pin Connection** |
| --- | --- |
| RFID SDA | Arduino Pin 10 |
| RFID SCK | Arduino Pin 13 |
| RFID MOSI | Arduino Pin 11 |
| RFID MISO | Arduino Pin 12 |
| RFID RST | Arduino Pin 5 |
| LCD SCL | Arduino Pin A5 |
| LCD SDA | Arduino Pin A4 |
| Green LED | Arduino Pin 6 |
| Red LED | Arduino Pin 7 |
| Buzzer | Arduino Pin (if applicable) |

Note: Ensure that the GND (Ground) and VCC (Voltage Common Collector) connections are made common among all the equipment to establish a common reference voltage. Additionally, consider using appropriate resistors to limit the current flow through the LEDs or other components, if required.

By following the guidelines and understanding the functionality of each component, you can successfully replicate and implement the RFID-based inventory control system for practical inventory management purposes.

**CODE :**

#include <SPI.h>

#include <MFRC522.h>

#include <LiquidCrystal.h>

#include <LiquidCrystal\_I2C.h>

#include <LiquidCrystal.h>

LiquidCrystal\_I2C lcd(0x27, 16, 2);

 int led1=6;

 int led2=7;

#define RST\_PIN 9

#define SS\_PIN 10

byte readCard[4];

String MasterTag = "FCF1A132";

String user1 = "D4CD7C89";

String user2 = "90567989";

String user3 = "E67C7889";

String user4 = "E9D42A8C";

String user5 = "F7D87689";// REPLACE this Tag ID with your Tag ID!!!

String tagID = "";

// Create instances

MFRC522 mfrc522(SS\_PIN, RST\_PIN);

void setup()

{

  lcd.init();

  lcd.backlight();

  SPI.begin(); // SPI bus

  mfrc522.PCD\_Init(); // MFRC522

   pinMode(led1,OUTPUT);

   pinMode(led2,OUTPUT);

  lcd.clear();

  lcd.print("Inventory Manage");

  lcd.setCursor(0, 1);

  lcd.print("Scan Product>>");

}

 int M2Mpieces = 7;

 int LCDpieces = 9;

 int Arduino = 12;

 int M2Fpieces = 76;

 int RFID = 88;

 int RELAY = 92;

void loop()

{

    Serial.begin(9600);

  //Wait until new tag is available

  while (getID())

  {

    lcd.clear();

    lcd.setCursor(0, 0);

    if (tagID == MasterTag || tagID == user1 || tagID == user2 || tagID == user3 || tagID == user4 || tagID == user5)

    {

      if(tagID == MasterTag)

      {

        lcd.print("  ARDUINO UNO  ");

        lcd.setCursor(0, 1);

          if(Arduino == 0)

          {

            lcd.print("Out of Stock");

                 digitalWrite(led2,HIGH);

       digitalWrite(led1,LOW);

       delay(200);

          }

          else

          {

          Arduino = Arduino - 1;

        lcd.print(Arduino);

         lcd.print(" Pieces ");

         digitalWrite(led1,HIGH);

          digitalWrite(led2,LOW);

          delay(2000);

          }

      }

      else if(tagID == user2)

      {

        lcd.print("    LCD    ");

        lcd.setCursor(0, 1);

        if(LCDpieces == 0)

        {

          lcd.print("Out of Stock");

          digitalWrite(led2,HIGH);

       digitalWrite(led1,LOW);

       delay(200);

        }

        else

        {

          LCDpieces = LCDpieces - 1;

        lcd.print(LCDpieces);

        lcd.print(" Pieces ");

         digitalWrite(led1,HIGH);

          digitalWrite(led2,LOW);

          delay(2000);

           }

      }

      else if(tagID == user3)

      {

        lcd.print("Jumper Wire M2M");

        lcd.setCursor(0, 1);

        if(M2Mpieces == 0)

        {

          lcd.print(" OUT of STOCK ");

               digitalWrite(led2,HIGH);

       digitalWrite(led1,LOW);

       delay(200);

        }

        else

        {

        M2Mpieces = M2Mpieces - 1;

        lcd.print(M2Mpieces);

        lcd.print(" Pieces ");

         digitalWrite(led1,HIGH);

          digitalWrite(led2,LOW);

          delay(2000);

        }

      }

      else if(tagID == user4)

      {

        lcd.print("Jumper Wire M2F");

        lcd.setCursor(0, 1);

        if(M2Fpieces == 0)

        {

             lcd.print(" OUT of STOCK ");

                  digitalWrite(led2,HIGH);

       digitalWrite(led1,LOW);

       delay(200);

        }

        else

        {

          M2Fpieces = M2Fpieces - 1;

        lcd.print(M2Fpieces);

        lcd.print(" Pieces ");

         digitalWrite(led1,HIGH);

          digitalWrite(led2,LOW);

          delay(2000);

        }

      }

      else if(tagID == user5)

      {

        lcd.print("  RFID Scanner  ");

        lcd.setCursor(0, 1);

        if(RFID == 0)

        {

          lcd.print(" OUT of STOCK ");

               digitalWrite(led2,HIGH);

       digitalWrite(led1,LOW);

       delay(200);

        }

        else{

          RFID = RFID - 1;

        lcd.print(RFID);

        lcd.print(" Pieces");

        digitalWrite(led1,HIGH);

        digitalWrite(led2,LOW);

        delay(2000);

        }

      }

      else

      {

          lcd.print("    Relay    ");

          lcd.setCursor(0, 1);

          if(RELAY == 0)

          {

            lcd.print(" OUT of STOCK ");

                   digitalWrite(led2,HIGH);

       digitalWrite(led1,LOW);

       delay(200);

          }

          else

          {

          RELAY = RELAY - 1;

          lcd.print(RELAY);

        lcd.print(" Pieces");

       digitalWrite(led1,HIGH);

       digitalWrite(led2,LOW);

       delay(2000);

          }

      // You can write any code here like opening doors, switching on a relay, lighting up an LED, or anything else you can think of.

      }

    }

    else

    {

       digitalWrite(led2,HIGH);

       digitalWrite(led1,LOW);

      lcd.print(" UNAVAIlable!");

       delay(2000);

    }

      // lcd.setCursor(0, 1);

      // lcd.print(" ID : ");

      // lcd.print(tagID);

     digitalWrite(led2,LOW);

     digitalWrite(led1,LOW);

    lcd.clear();

    lcd.print("Inventory Manage");

    lcd.setCursor(0, 1);

    lcd.print("Scan Product>>");

  }

}

//Read new tag if available

boolean getID()

{

  // Getting ready for Reading PICCs

  if ( ! mfrc522.PICC\_IsNewCardPresent()) { //If a new PICC placed to RFID reader continue

  return false;

  }

  if ( ! mfrc522.PICC\_ReadCardSerial()) { //Since a PICC placed get Serial and continue

  return false;

  }

  tagID = "";

  for ( uint8\_t i = 0; i < 4; i++) { // The MIFARE PICCs that we use have 4 byte UID

  //readCard[i] = mfrc522.uid.uidByte[i];

  tagID.concat(String(mfrc522.uid.uidByte[i], HEX)); // Adds the 4 bytes in a single String variable

  }

  tagID.toUpperCase();

  mfrc522.PICC\_HaltA(); // Stop reading

  return true;

}