Year 2 Project

Door Control System

Department of Electrical Engineering and Electronics

16 March 2022

Abstract

In this project, we focus on building the door control system, aiming to achieve a door model with relatively high security level. In order to simulate a door control system model, a set of Arduino components such as LCD display and keypad is used and programming work is involved. The final result is a paper box door control system model which can protect the door through verifying PIN number, password and NFC card. The achievements and problems of the model is reflected, applications and future improvements also being considered.

Declaration

I confirm that I have read and understood the University's definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with any other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences of engaging in plagiarism and collusion as

Contents

Introduction:

In recent years, the problem of violation of property safety has been drawing great attention.

Experts are focusing on building solid access control system in order to change this situation. The security system is significant at homes, schools, offices and industries in real life [1]. Among these access control systems, RFID technology, which represents for radio frequency identification technology, is widely used in today's market to permit only authenticated and authorized personnel to enter secure spaces [2]. The RFID reader can read the information from ID card into current system to verify its identity. If the identity is authorized, the status of door lock will be unlocked. On the contrary, if the identity is not authorized, the door lock will be locked. The purpose of this project is to develop a smart security system using RFID technology and keypad based on Arduino UNO. This report will first introduce the materials and methods of the system, then make an evaluation of the experiment results and finally make a conclusion of this system.

Materials and Methods

Overview

The project used a whole set of Arduino-based components and tools such as keypad, LCD, motor and card reader to finish its functionalities and the purpose of the project is to use all these components to construct a door control system. In this project, the system is aimed to have function of password-after-PIN and card-after-PIN verification to control the door through programming and assembly skills. In detail, keypad is used for user input, LCD is used for displaying instructions to tell the user what to do, card reader is used to verify the card and motor is used as a lock.

Material/Equipment List

NO.	Materials	Type	Quantity
1.	Arduino Uno Main Board	A000066	1

2.	USB Cable	Cable50cm	1
3.	Breadboard	AD-102	1
4.	Keypad	VMA300	1
5.	LCD Display	WH1602B-YYH-JT	1
6.	RFID Control	364	1
7.	Stepper Motor	105990072	1
8.	Arduino IDE	--	--
9.	Paper Box	--	1
10.	Wire	--	some

Experiment Process

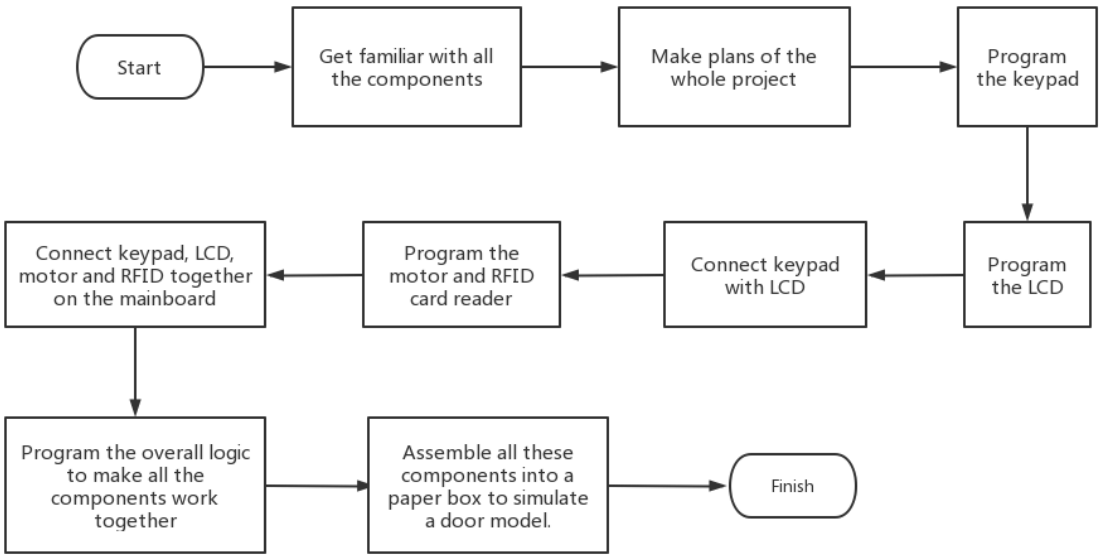

Figure 1 Flow chart of the whole procedure

As the figure above shows, figure 1 is a general flowchart of the project procedure.

Chronological Steps

1. We started by learning about how each component works and how to make them work.

2. Then the general plan and schedule of the project is made: let each device works independently and then combine them together to form a functional system.

3. After formally launching the work, we first programmed the keypad. The core part is to enable it to output numbers correctly to the program. A serial monitor is used to check whether it ran right.

4. Next step we used a LCD display and connected it to the keypad, making is displays numbers correctly received from keypad input. The keypad and LCD is set up at first because they could be seen as the interface of the system which is accessed by users to interact with the system. The following steps is to let this interface truly works.

5. The remaining components we dealt with are the stepper motor and the RFID card reader. We enabled the motor to rotate a certain amount of angle to simulate a lock and we make the card reader to recognize cards that are authorized in programming.

6. The following part is to make them work together, so we connected all of the components to the main board and write code to enable them response properly at proper time.

7. Finally, we assembled all these into a paper box in order to simulate a real door control system, then it could work normally. The logic of the system is shown in Figure 2, which summarizes all the steps above.

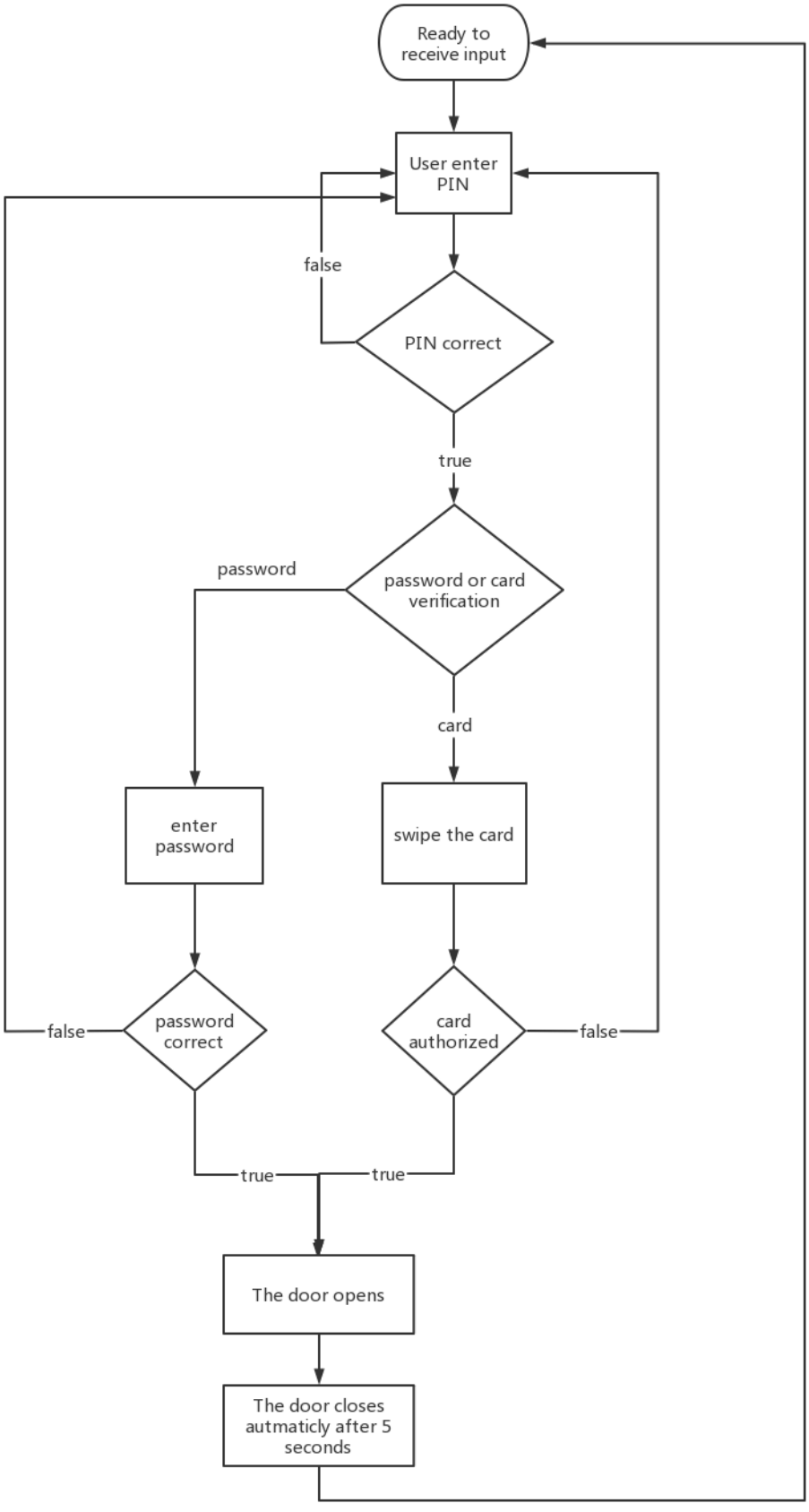

Figure 2 Logic of the system

Results

The final result of the project is shown as Figure 3 and Figure 4. It is expected to simulate a lock that can be controlled by password and card. The whole system that mainly consists of card reader, lock, keypad and LCD display is built on the Arduino platform. The first step to use the system is entering the PIN number through the keypad. All the input through the keypad like the number of pin or password are shown on the LCD display. In addition, the LCD display also shows some basic sentences for example "please enter your password", "the password is wrong", "approved" to tell users the current situation. And then, as the PIN number is correct, the user can choose to use password or card to open the door. When user choose to use password and enter the correct password, the door is unlocked. The function of lock is realized by a stepper motor and a stick attach to it. When the motor starts working, the stick rotating with it so the stick becomes a bolt that can fasten the door. When user choose to use card, the card reader can recognize the right card and unlock the door. The card reader system is based on an RFID controller, which is able to identify the ID of different cards through electromagnetic fields. Thus, the card reader can recognize the wrong card and the door will not open. Furthermore, an extra function that the door will close after opening for 5 seconds is added to the system.

However, there are several small problems in the result.

1. Due to the rough surface of paper box, the lock is likely to get stuck when it rotates.

2. Since the keypad only contains numbers, it isn't able to allow a strong password which includes both symbols, letters and numbers.

These problems could be solved with the improvement of equipment.

In general, the basic functions that are expected at the beginning are most achieved and the system works well.

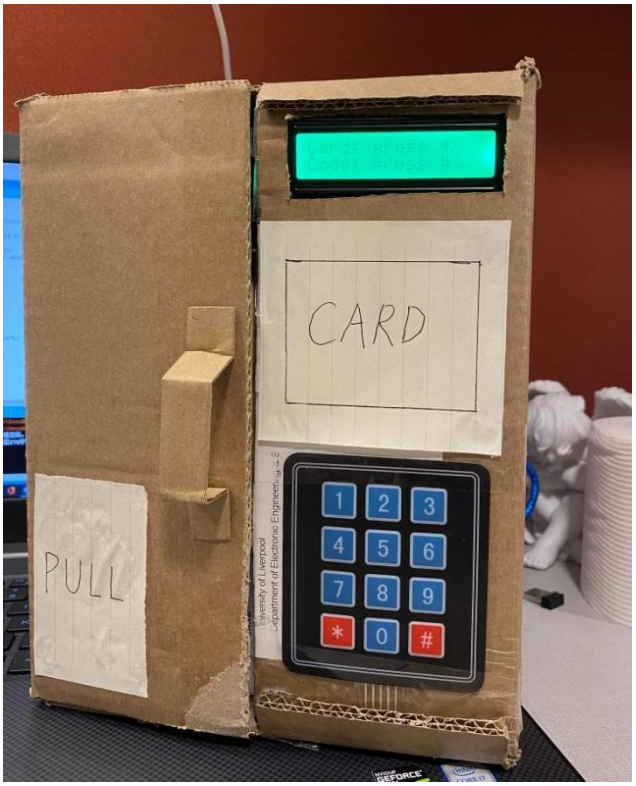

Figure 3 The model of the door control system

Figure 4 Internal view of the door control system

Discussion and Conclusion:

The project develops a simple RFID intelligent access control system based on Arduino. The system realizes the basic function of authentication and authorization, for example, can be used when building a study room. Since valuables like computers, tablets and charges may be left in the room when students go out for a while, it is necessary for the school to equip the room with access control system to guarantee the safety of students' properties. The system uses ID card or password to verify the users and control the status of the door lock accordingly. However, the system has its limitations. If the user choose the password to enter the system, it will be difficult to differentiate different users without facial recognition technology or fingerprint recognition technology. Considering that if someone use others' ID card to access the room, there should be a precondition before the step of swiping the ID card. As a result, different user has a unique pin code which should be entered before the password. Since the user should first enter their six-bit pin, then enter the password or swipe the ID card, it may be troublesome for the user to remember two code.

For further research, there are many recommendations for reference. As stated above, the system can use facial recognition or fingerprint recognition instead of pin code as a function of differentiating users. When the user enter the fingerprint or scan their face which are already stored in the access control system, the system will allow their access to the door. Furthermore, the system equipped with face or fingerprint recognition technology can be more reliable than other systems. What's more, the Iris recognition has become the wide-acceptable reliable form of biometric authentication.

References

[1]. N. N. S. Hlaing, S. S. Lwin, "Electronic Door Lock using RFID and Password Based on Arduino." International Journal of Trend in Scientific Research and Development, Mar. 2019.

[2] "Comparing RFID and NFC Access Control Systems", Kisi, [Online]. Available: https://www.getkisi.com/guides/rfid-access-control [Accessed by Mar. 12, 2022]

Appendices

Appendix A

Source Code (IDE: Arduino IDE)

```
#include <AccelStepper.h>
#include <MultiStepper.h>
#include <rgb_lcd.h>
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>
#include <Wire.h>
#include <Adafruit_PN532.h>

//--------------------------------------------------------------------------
//--------------------------------------------------------------------------
//size of the password
#define pin_size 7
#define password_size 7

//information
String pin[5];
String pswd[5];

//to hold password input
char inputPin[pin_size];
char inputPassword[password_size];

//the true password
char masterPassword[password_size] = "202122";

//number of input password character
int pinCount = 0;
int pswdCount = 0;

bool isLocked = false;

const byte rows = 4;
const byte columns = 3;
char hexaKeys[rows][columns] = {
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};
```

```
//pins of keypad
byte row_pins[rows] = {9, 8, 7, 6};
byte column_pins[columns] = {5, 4, 3};
Keypad keypad = Keypad( makeKeymap(hexaKeys), row_pins, column_pins, rows, co
lumns);

rgb_lcd lcd;

const int colourR = 0;
const int colourG = 255;
const int colourB = 0;


//------------------------------------------------------------------------
//------------------------------------------------------------------------

// If using the breakout or shield with I2C, define just the pins connected
// to the IRQ and reset lines.  Use the values below (2, 3) for the shield!
#define PN532_IRQ   (2)
#define PN532_RESET (3)  // Not connected by default on the NFC Shield

// Uncomment just _one_ line below depending on how your breakout or shield
// is connected to the Arduino:

// Use this line for a breakout with a SPI connection:
//Adafruit_PN532 nfc(PN532_SCK, PN532_MISO, PN532_MOSI, PN532_SS);

// Use this line for a breakout with a hardware SPI connection.  Note that
// the PN532 SCK, MOSI, and MISO pins need to be connected to the Arduino's
// hardware SPI SCK, MOSI, and MISO pins.  On an Arduino Uno these are
// SCK = 13, MOSI = 11, MISO = 12.  The SS line can be any digital IO pin.
//Adafruit_PN532 nfc(PN532_SS);

// Or use this line for a breakout or shield with an I2C connection:
Adafruit_PN532 nfc(PN532_IRQ, PN532_RESET);

#if defined(ARDUINO_ARCH_SAMD)
#define Serial SerialUSB
#endif

//------------------------------------------------------------------------
//------------------------------------------------------------------------

// Motor pin definitions:
```

```
#define motorPin1  10      // IN1 on the ULN2003 driver
#define motorPin2  11      // IN2 on the ULN2003 driver
#define motorPin3  12     // IN3 on the ULN2003 driver
#define motorPin4  13     // IN4 on the ULN2003 driver
// Define the AccelStepper interface type; 4 wire motor in half step mode:
#define MotorInterfaceType 8
// Initialize with pin sequence IN1-IN3-IN2-
IN4 for using the AccelStepper library with 28BYJ-48 stepper motor:
AccelStepper stepper = AccelStepper(MotorInterfaceType, motorPin1, motorPin3,
 motorPin2, motorPin4);


void setup() {

  //initialize keypad and lcd
  //-------------------------------------------------------------------------
  //-------------------------------------------------------------------------
  // Serial monitor
    Serial.begin(9600);
    //initPinAndPassword(pin,pswd);
    pin[0] = "201600";
    pswd[0] = "123456";
    pin[1] = "201508";
    pswd[1] = "097890";
    pin[2] = "201623";
    pswd[2] = "874345";
    pin[3] = "201232";
    pswd[3] = "223412";
    pin[4] = "201309";
    pswd[4] = "988721";
    lcd.begin(16, 2);
    lcd.setRGB(colourR, colourG, colourB);
    lcd.print("Protected Door");
    loading("Loading");
    lcd.clear();

    //initialize card reader
    //-------------------------------------------------------------------------
-
    //-------------------------------------------------------------------------
#ifndef ESP8266
  while (!Serial); // for Leonardo/Micro/Zero
#endif
```

```
  Serial.begin(115200);
  while (!Serial) delay(10); // for Leonardo/Micro/Zero
  Serial.println("Hello!");

  nfc.begin();
  uint32_t versiondata = nfc.getFirmwareVersion();
  if (! versiondata) {
    Serial.print("Didn't find PN53x board");
    while (1); // halt
  }
  nfc.setPassiveActivationRetries(0xFF);
  // configure board to read RFID tags
  nfc.SAMConfig();

  //initialize motor
  //-------------------------------------------------------------------------
  //-------------------------------------------------------------------------
  // Set the maximum steps per second:
  stepper.setMaxSpeed(1000);
}


//main procedure
//-------------------------------------------------------------------------
//-------------------------------------------------------------------------
void loop() {

  // receive the bottom value
  int pin_index = 0;
  lcd.setCursor(0, 0);
  lcd.print("Input your pin:");

  //pin(ID)
  char userKey = keypad.waitForKey();
  if (int(userKey) != 0) {
    inputPin[pinCount] = userKey;
    Serial.println(userKey);
    lcd.setCursor(pinCount, 1);
    lcd.print(userKey);
    pinCount++;
  }
  if (pinCount == pin_size - 1) {
    for (int i = 0; i < 5; i++) {
      const char *temp = pin[i].c_str();
```

```
      if (!strcmp(inputPin, temp)) {//if pin correct
        bool rightCard = true;
        while (true) {
          delay(1000);
          lcd.clear();
          lcd.print("Card: press *");
          lcd.setCursor(0, 1);
          lcd.print("Code: press #");
          delay(1000);
          char selectKey = keypad.waitForKey();
          if (int(selectKey) != 0) {
            if (selectKey == '*') {
              //-------------------------------------------------------------
  ---------------------------------
              //use card reader to enter
              lcd.clear();
              lcd.print("Swipe your card");
              rightCard = verifyCard();
              if (rightCard == true) {
                delay(300);
                lcd.clear();
                lcd.println("Approved.");
                counterClockWise();
                //clockWise();
                delay(1000);
                lcd.setCursor(0, 1);
                lcd.print("Door is open now.");
                //unLock();
                delay(2000);
                lcd.clear();
                lcd.print("Waiting...");
                //loading("waiting");
                delay(2000);
                lcd.clear();
                clockWise();
                //counterClockWise();
                lcd.print("Door is closing...");
                delay(2000);
                clearPin();
                break;
              } else {
                delay(1000);
                lcd.setCursor(0, 1);
                lcd.print("You are denied.");
```

```
                delay(2000);
                lcd.clear();
                clearPin();
                break;
              }

            }
            else if (selectKey == '#') {
              //-------------------------------------------------------------
  ---------------------------------
              //use password to enter
              pin_index = i;
              delay(1000);
              lcd.clear();
              lcd.print("Enter password:");
              bool flag = true;
              while (flag) {
                char pswdKey = keypad.getKey();
                if (int(pswdKey) != 0) {
                  inputPassword[pswdCount] = pswdKey;
                  Serial.println(pswdKey);
                  lcd.setCursor(pswdCount, 1);
                  lcd.print(pswdKey);
                  pswdCount++;
                }
                if (pswdCount == password_size - 1) {
                  flag = false;
                }
              }
              if (pswdCount == password_size - 1) {
                //verify password
                const char *temp2 = pswd[pin_index].c_str();
                if (!strcmp(inputPassword, temp2)) {
                  delay(300);
                  lcd.clear();
                  lcd.println("Approved.");
                  counterClockWise();
                  delay(1000);
                  lcd.setCursor(0, 1);
                  lcd.print("Door is open now.");
                  delay(2000);
                  lcd.clear();
                  lcd.print("Waiting...");
                  //loading("waiting");
```

```
                    delay(1000);
                    lcd.clear();
                    clockWise();
                    lcd.print("Door is closing...");
                    delay(2000);
                  } else {
                    delay(1000);
                    Serial.println("Password wrong");
                    lcd.clear();
                    lcd.print("Password wrong");
                    delay(1000);
                    lcd.setCursor(0, 1);
                    lcd.print("You are denied.");
                    delay(3000);
                  }
                  lcd.clear();
                  clearPin();
                  clearPswd();
                  break;
                }
              } else {
                delay(1000);
                lcd.clear();
                lcd.print("Invalid input");
                delay(1000);
                lcd.clear();
              }
            }
          }
          return;
        }
    }
    clearPin();
    delay(1000);
    lcd.clear();
    lcd.print("Wrong pin.");
    delay(1500);
    lcd.clear();
    lcd.print("Enter again.");
    delay(1500);
    return;
  }
}
```

```
298.
299. //keypad and lcd method
300. //------------------------------------------------------------------------
301. //------------------------------------------------------------------------
302. void loading (char msg[]) {
303.   lcd.setCursor(0, 1);
304.   lcd.print(msg);
305.
306.   for (int i = 0; i < 9; i++) {
307.     delay(1000);
308.     lcd.print(".");
309.   }
310. }
311.
312. void clearPin() {
313.   while (pinCount != 0)
314.   {
315.     inputPin[pinCount--] = 0;
316.   }
317.   return;
318. }
319.
320. void clearPswd()
321. {
322.   while (pswdCount != 0)
323.   {
324.     inputPassword[pswdCount--] = 0;
325.   }
326.   return;
327. }
328.
329. //card verification
330. //------------------------------------------------------------------------
331. //------------------------------------------------------------------------
332.
333. bool verifyCard() {
334.   boolean success;
335.   uint8_t cardId[] = {4,244,162,1,20,49,3};
336.   uint8_t card2Id[]={4,61,140,114,238,109,128};
337.   uint8_t uid[] = { 0, 0, 0, 0, 0, 0, 0 };  // Buffer to store the returned
    UID
338.   uint8_t uidLength;        // Length of the UID (4 or 7 bytes depending on
    ISO14443A card type)
339.
```

```
  // Wait for an ISO14443A type cards (Mifare, etc.).  When one is found
  // 'uid' will be populated with the UID, and uidLength will indicate
  // if the uid is 4 bytes (Mifare Classic) or 7 bytes (Mifare Ultralight)

  success = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A, &uid[0], &uidLength);
  success = true;
  for(uint8_t i=0; i<uidLength; i++){
    Serial.print(" 0x");
    Serial.print(uid[i]);
    if(uid[i]!=cardId[i]&&uid[i]!=card2Id[i]){
      success = false;
    }
  }
  return success;
}

//motor operation
//------------------------------------------------------------------------
//------------------------------------------------------------------------

void clockWise() {
  // Set the current position to 0:
  stepper.setCurrentPosition(0);
  // Run the motor forward at 5000 steps/second until the motor reaches 4096 steps (1 revolution):
  while (stepper.currentPosition() != 2048) {
    stepper.setSpeed(5000);
    stepper.runSpeed();
  }
  delay(1000);
}

void counterClockWise() {
  // Reset the position to 0:
  stepper.setCurrentPosition(0);
  // Run the motor backwards at 5000 steps/second until the motor reaches -4096 steps (1 revolution):
  while (stepper.currentPosition() != -2048) {
    stepper.setSpeed(-5000);
    stepper.runSpeed();
  }
  delay(1000);
}
```