

RFID Lock/Unlock

Autor: [Alexandru-Catalin Tache](#)

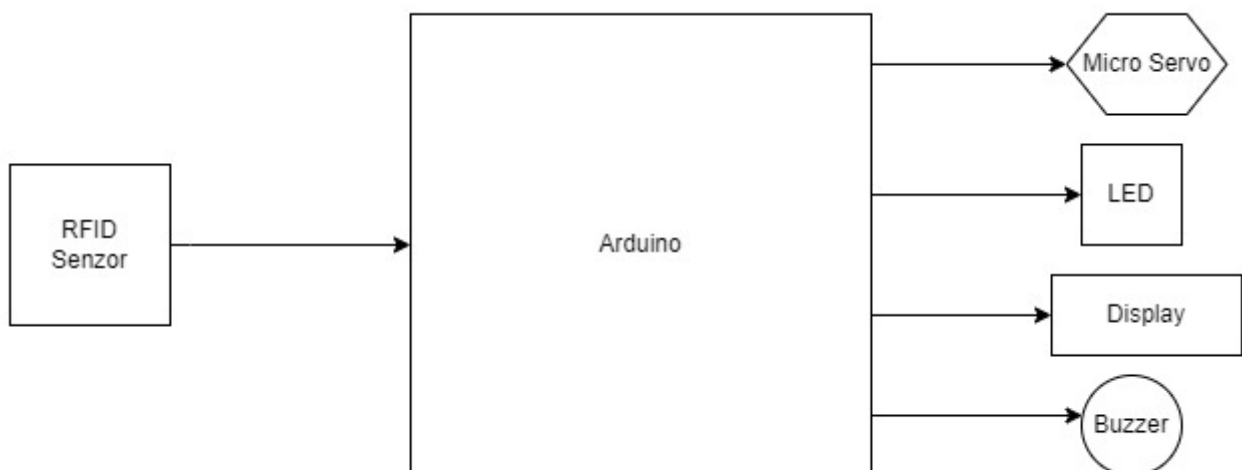
Introducere

Proiectul reprezinta un sistem de incuiat si descuiat usa, in baza unei cartele de tip RFID. Mod de functionare:

- la apropierea cartelei de senzor se va inchide/deschide o incuietoare
- semnalizarea vizuala va fi prin intermediul unor LED-uri (red + green)
- va fi incorporat si un semnal sonor provocat de un buzzer
- in caz ca nu va fi utilizata o cartela conforma, atunci se va oferi un mesaj de eroare.

Descriere generală

Schema bloc descrisa mai jos prezinta circuitul ce are ca principal senzor (parte de input) modulul RFID, iar prin placuta Arduino se ofera rezultate pe LED-uri, buzzer si servomotor (outputuri). Schema bloc:

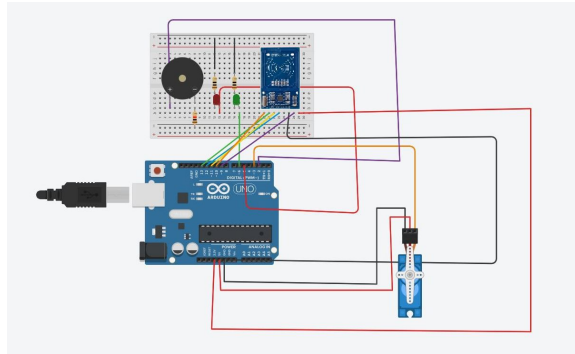


Hardware Design

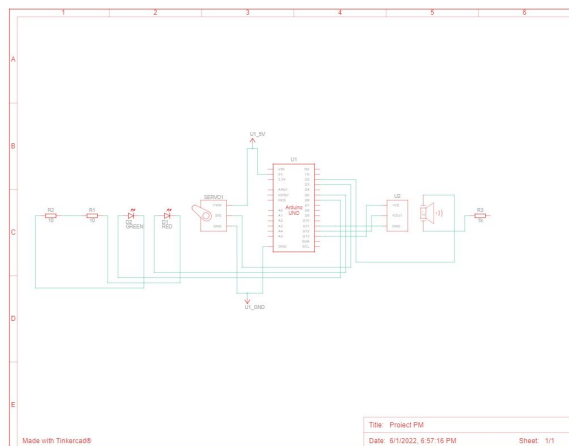
Lista de piese:

- Arduino

- Breadboard
- Jumpers
- RC522 RFID Sensor
- Micro Servo
- LED-uri
- Rezistori
- Buzzer



Schema electrica a circuitului a fost realizata utilizand tinkercad, iar designul prezinta modalitatea de conexiune la nivel intern a componentelor si disparea curentului electric. Aceasta este prezentata astfel:



Software Design

Scurta descriere generala a softului

Software-ul este la baza impartit in 2 bucati cea de realizare a citirii datelor cu ajutorul senzorului RFID si accesarea unor medii de output pentru afisarea unor rezultate. Pentru realizarea acestui proiect am utilizat o biblioteca specifica a RFID, ce contine implementarile functiilor de citire, halt, scriere sau selectTag (pentru a nu avea o citire redundanta). Pe aceste functii le-am corelat in cadrul functiilor prezentate mai jos.

Detalii implementare (functii utilizate)

Funcțiile utilizate sunt următoarele:

- `setup()`: initializare rfid, initializare si o scurta aprindere si oprire a LED-urilor, initializare scurta a buzzerului, transmiterea unui mesaj ca se poate citi cartela, initializare servomotor
- `loop()`: citire cod de pe cartela, apelare functie de verificat accesul utilizatorului si aplicare de functie `selectTag` pentru eliminarea redundantei
- `acces()`: verificare daca codul primit se afla in vectorul codurilor permise pentru blocare/deblocare, in caz afirmativ servomotorul isi va schimba pozitia curenta si va trece fie in cea de blocare fie de deblocare, iar permisiunea va fi validata si de buzzer, in timp ce in caz de folosinta a unui card nevalidabil se va oferi un raspuns negativ si ambele LED-uri vor lumina

Codul realizat

```
#include <SPI.h>
#include <Servo.h>

#ifndef RFID_h
#define RFID_h

#include <Arduino.h>

#define MAX_LEN 16
#define BUZZ 2

#define PCD_IDLE          0x00
#define PCD_AUTHENT       0x0E
#define PCD_RECEIVE       0x08
#define PCD_TRANSMIT      0x04
#define PCD_TRANSCEIVE    0x0C
#define PCD_RESETPHASE    0x0F
#define PCD_CALCCRC       0x03

#define PICC_REQIDL       0x26
#define PICC_REQALL       0x52
#define PICC_ANTICOLL     0x93
#define PICC_SELECTTAG    0x93
#define PICC_AUTHENT1A    0x60
#define PICC_AUTHENT1B    0x61
#define PICC_READ          0x30
#define PICC_WRITE         0xA0
#define PICC_DECREMENT     0xC0
#define PICC_INCREMENT     0xC1
#define PICC_RESTORE       0xC2
#define PICC_TRANSFER      0xB0
#define PICC_HALT          0x50

#define MI_OK              0
#define MI_NOTAGERR       1
#define MI_ERR             2

#define Reserved00        0x00
#define CommandReg        0x01
#define CommIEnReg        0x02
#define DivlEnReg         0x03
#define CommIrqReg        0x04
#define DivIrqReg         0x05
#define ErrorReg          0x06
#define Status1Reg        0x07
#define Status2Reg        0x08
#define FIFODataReg       0x09
#define FIFOLevelReg      0x0A
#define WaterLevelReg     0x0B
#define ControlReg        0x0C
#define BitFramingReg     0x0D
#define CollReg           0x0E
#define Reserved01        0x0F

#define Reserved10        0x10
#define ModeReg           0x11
#define TxModeReg         0x12
#define RxModeReg         0x13
#define TxControlReg      0x14
#define TxAutoReg         0x15
#define TxSelReg          0x16
```

```

#define RxCS1Reg 0x17
#define RxFreqHoldReg 0x18
#define DownReg 0x19
#define Reserved11 0x1A
#define Reserved12 0x1B
#define HIFreqReg 0x1C
#define Reserved13 0x1D
#define Reserved14 0x1E
#define SerialSpeedReg 0x1F
#define Reserved18 0x20
#define CKResu11Reg 0x21
#define CKResu12Reg 0x22
#define Reserved21 0x23
#define ModuLiftReg 0x24
#define Reserved22 0x25
#define RxCFSReg 0x26
#define GUSReg 0x27
#define CUSReg 0x28
#define HUSReg 0x29
#define THUSReg 0x2A
#define TTRVcalerReg 0x2B
#define TloadReg 0x2C
#define TloadReg 0x2D
#define TCounterValueReg 0x2E
#define TCounterValueReg 0x2F
#define Reserved30 0x30
#define TestSelReg 0x31
#define TestSelReg 0x32
#define TestPinReg 0x33
#define TestPinValueReg 0x34
#define TestBusReg 0x35
#define AutoTestReg 0x36
#define VersionReg 0x37
#define AnalogTestReg 0x38
#define TestDCReg 0x39
#define TestDACReg 0x3A
#define TestADCReg 0x3B
#define Reserved31 0x3C
#define Reserved32 0x3D
#define Reserved33 0x3E
#define Reserved34 0x3F

class RFID
{
public:
    RFID(int chipSelectPin, int MRSFPD);
    unsigned char *serNum[5];
    void init();
    void reset();

    void setBitMask(unsigned char reg, unsigned char mask);
    void clearBitMask(unsigned char reg, unsigned char mask);
    void antennaOff(void);
    void antennaOn(void);
    void calculateCRC(unsigned char *pData, unsigned char len, unsigned char *pOutData);
    void writeRF6222(unsigned char addr, unsigned char val);

    unsigned char readRF6222(unsigned char addr);
    unsigned char findCard(unsigned char reqPos, unsigned char *tagType);
    unsigned char RF6222Card(unsigned char cmd, unsigned char *sendData, unsigned char *recvData, unsigned char *backData, unsigned int *backLen);
    unsigned char anticoll(unsigned char *serNum);

    unsigned char auth(unsigned char authMode, unsigned char blockAddr, unsigned char *SectorKey, unsigned char *serNum);
    unsigned char read(unsigned char blockAddr, unsigned char *recvData);
    unsigned char write(unsigned char blockAddr, unsigned char *writeData);
    unsigned char selectTag(unsigned char *serNum);

    void halt();

private:
    int chipSelectPin;
    int MRSFPD;
};

#endif

RFID rfid(18, 9);
unsigned char status;
unsigned char str[MAX_LEN]; //lungime cod cartela
String coduri_acceptate[2] = {"9334151514", "912315814410"}; //coduri cartela acceptate
int nr_coduri_acceptate = 2;

Servo lockServo;
int lockPos = 15;
int unlockPos = 280;
boolean locked = true;

int redLEDPin = 5;
int greenLEDPin = 6;

void setup()
{
    Serial.begin(9600);
    SPI.begin();
    rfid.init(); //initializare RFID
    pinMode(redLEDPin, OUTPUT);
    pinMode(greenLEDPin, OUTPUT);
    digitalWrite(redLEDPin, HIGH);
    delay(200);
    digitalWrite(greenLEDPin, HIGH);
    delay(200);
    digitalWrite(redLEDPin, LOW);
    delay(200);
    digitalWrite(greenLEDPin, LOW);
    lockServo.attach(3);
    lockServo.write(lockPos); //Miscare servomotor
    tone(BUZZ, 500);
    delay(500);
    noTone(BUZZ);
    Serial.println("Astept cartela...");
}

void loop()
{
    if (rfid.findCard(PICC_REQIDL, str) == MI_OK) //verificare daca se citeste o cartela
    {
        Serial.println("Card gasit");
        String cod = "";
        if (rfid.anticoll(str) == MI_OK) //detectare anticollisione
        {
            Serial.println("ID-ul cardului este : ");
            for (int i = 0; i < 4; i++)
            {
                {
                    cod = cod + (0x0F & (str[i] >> 4));
                    cod = cod + (0x0F & str[i]);
                }
                Serial.println(cod);
                acces(cod);
            }
            rfid.selectTag(str); //pentru a nu avea o citire redundanta
        }
        rfid.halt();
    }

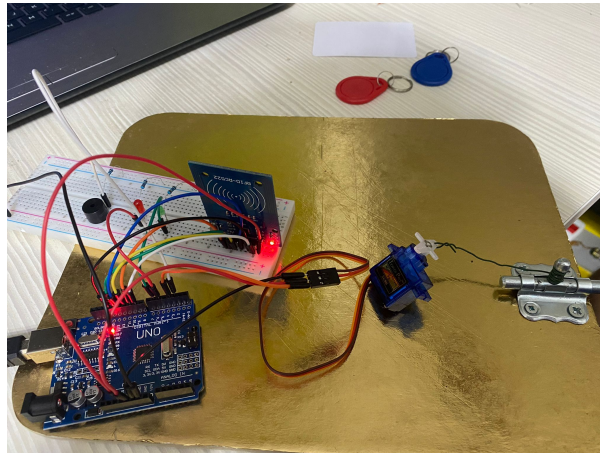
    void acces(String cod) //functie ce permite accesul sau nu
    {
        int ok = 0; //pentru permiterea accesului sau nu
        for (int i=0; i <= (nr_coduri_acceptate - 1); i++) //cautare cod citit in vectorul de coduri
        {
            if(coduri_acceptate[i] == cod)
            {
                Serial.println ("Acces Permis");
                ok = 1;
                if(locked == true)
                {
                    lockServo.write(unlockPos);
                    locked = false;
                }
                else if(locked == false)
                {
                    lockServo.write(lockPos);
                    locked = true;
                }
            }
            break;
        }
    }

    if (ok == 0)
    {
        Serial.println ("Acces Respins");
        digitalWrite(redLEDPin, HIGH);
        delay(200);
        digitalWrite(redLEDPin, LOW);
        delay(200);
        digitalWrite(redLEDPin, HIGH);
        delay(200);
        digitalWrite(redLEDPin, LOW);
        delay(200);
    }

    if(ok == 1) {
        digitalWrite(greenLEDPin, HIGH);
        delay(200);
        digitalWrite(greenLEDPin, LOW);
        tone(BUZZ, 500);
        delay(200);
        noTone(BUZZ);
    }
}

```

Rezultate Obținute



Concluzii

Consider ca proiectul a fost util pentru intelegerea lucrului cu placuta Arduino si folosirea unor componente suplimentare precum RFID, servomotor, buzzer sau LED uri. Si inca o parte mai interesanta, dar in acelasi timp cea mai grea a proiectului lipirea pinilor de conexiune la breadboard a senzorului de citire a cardurilor/tagurilor RFID. :)

Download

[tache_alexandru-catalin_335ca_pm_rfid.zip](#)

O arhivă (sau mai multe dacă este cazul) cu fișierele obținute în urma realizării proiectului: surse, scheme, etc. Un fișier README, un ChangeLog, un script de compilare și copiere automată pe uC crează întotdeauna o impresie bună 😊.

Fișierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fișierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul).
Exemplu: Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

<https://medium.com/autonomous-robotics/an-introduction-to-rfid-dc6228767691>

<https://docs.arduino.cc/learn/electronics/servo-motors>

<https://www.the-diy-life.com/arduino-based-rfid-door-lock-make-your-own/>

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** și **Resurse Hardware**.

[Export to PDF](#)

From:

<http://ocw.cs.pub.ro/courses/> - **CS Open CourseWare**

Permanent link:

<http://ocw.cs.pub.ro/courses/pm/prj2022/arosca/rfid-lock>



Last update: **2022/06/02 07:16**