RFID Parking

Introducere

Aceasta lucrare presupune proiectarea unui sistem de parcare ce are la baza autentificarea folosind cartele/tag-uri RFID. Scopul autentificarii este pentru a securiza/restrictiona accesul in parcari, fie ca ar fi vorba de o parcare rezidentiala, comerciala, industriala etc.

Pentru functionalitatea de baza se vor folosi: 1x Arduino UNO, 1x Servomotor, 1x Modul RFID. La acestea pentru o utilizare mai intuitiva se vor adauga urmatoarele componte extra: 1x buzzer, 1x I2C LCD Display, 3x LEDs

Descriere generală

Utilizatorul prezinta cartela la senzorul RFID. Daca aceasta este gresita, buzzer-ul si semaforul vor atentiona utilizatorul de acest lucru. Daca este corecta, va fi actionat servomotorul, ridicand bariera. In tot acest timp semaforul se va afla in 3 stari, determinate de pozitia barierei: RED (CLOSED), YELLOW (AJAR = OPENING/CLOSING), GREEN (OPEN), iar pe display vor fi afisate mesaje corespunzatoare.

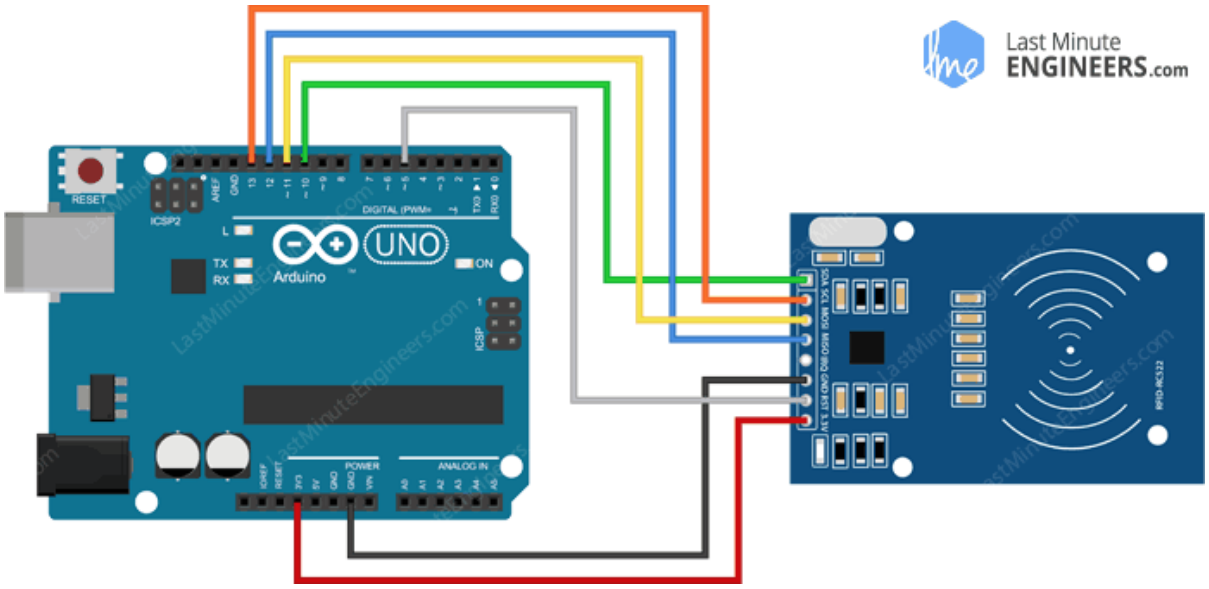
Schema bloc:

Hardware Design

Componente:

1. Arduino UNO
2. RFID RC522
3. Micro Servomotor SG90
4. LCD 1602 cu interfata I2C
5. Buzzer activ de 3V
6. LED 3 culori (Red, Yellow, Green)

Schema de wiring in Tinkercad. Deoarece in Tinkercad nu am gasit componentele de RFID si LCD 1602 I2C, acestea vor fi puse separat.



Software Design, Detalii Implementare

Am folosit bibliotecile

```
#include <SPI.h> - RFID communication
#include <MFRC522.h> - RFID
#include <Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <string.h>
```

Am pornit din exemplul **DumpInfo** din biblioteca MFRC522, care afiseaza datele unui tag detectat, din care am selectat UID-ul acestuia. De acolo am creat vectorul (de strings) de taguri acceptate, vectorul de masini in parcare, logica acceptarii in parcare, interactiunea/oferirea de informatii prin intermediul display-ului, buzzer-ului, semaforului.

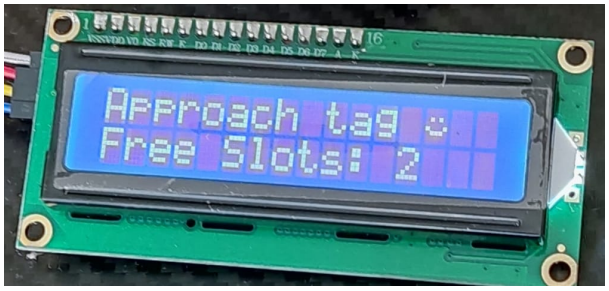
Functia de loop:

```
void loop()
{
  lcd.createChar(0, smiley);
  lcd.createChar(1, frown);
```

```
  if(getPresentCars() == CAPACITY )
    display_fullMessage();
  else
    display_WelcomeCounter();

  lightsRed(); // by default the gate is lowered

  // Look for new cards
  if ( ! mfrc522.PICC_IsNewCardPresent())
    return;

  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial())
    return;


  String tag = getTagID();

  // accepted tag and free slots in the parking area
  if (validTag(tag) && (getPresentCars() != CAPACITY || is_carPresent(tag))
)
  {

    if (!is_carPresent(tag))
      addCar(tag);
    else
      removeCar(tag);

    lightsYellow();

    tone(BUZZER_PIN, 6644, 250); // acceptance okay-ish sound
    raiseGate();
    lightsGreen();

    delay(2500);

    lightsYellow();
    lowerGate();
  }

  else   {
    lightsFail();
    buzzerFail();
    delay(3000);
  }
}
```

Implementarea functiilor din loop poate fi vazuta in fisiserul *parking.ino* din arhiva aflata in sectiunea **Download**.

- 1. Se verifica numarul de masini prezente in parcare si se afiseaza mesajul corespunzator



- 2. By default, bariera este data jos, deci si semaforul ar trebui sa aiba lumina rosie aprinsa
- 3. Se verifica daca exista un tag prezent si daca se poate citi. Daca nu este unul prezent se returneaza pentru optimizare
- 4. Daca exista, atunci se identifica UID-ul acestuia
  - 4.1. Daca este un card valid **(*)** ridicam bariera, scoatem un sunet de OK, schimbam culoarea semaforului. Dupa un timp se coboara bariera, se schimba culoarea semaforului iarasi si actualizam numarul de masini prezente din parcare, lucru ce va fi afisat si pe ecran.
  - 4.2. Daca nu este un card valid, semaforul va palpai rosu, iar buzzer-ul va scoate un sunet de eroare

***(*)** - In realitate nu se verifica doar daca este un tag valid. Verificam de asemenea si daca masina doreste sa iasa din parcare sau sa intre. Daca masina vrea sa iasa se va ridica bariera si aici se termina acest caz, dar daca masina vrea sa intre intai trebuie sa fie indeajuns de multe locuri disponibile in parcare, altfel se trece in rutina de fail precizata la 4.2.*

Rezultate Obţinute, Video Demo

Pentru vizualizarea unui exemplu de utilizare se poate apasa pe imagine (link spre youtube). In exemplu se poate observa rutina urmatoare:

1. tag invalid
2. primul tag valid
3. al doilea tag valid (parcarea este plina acum)
4. tag valid X(care nu este in parcare), dar parcarea este plina → fail
5. tag valid care este in parcare → parcarea nu mai este plina
6. tag valid X care poate intra acum (exista loc in parcare)

Concluzii

Desi pare simplist, proiectul poate avea o grad de utilitate destul de mare, prin parcarile private, rezidentiale etc. Pentru a fi folosite in aceste spatii ar fi nevoie totusi de un motor cu putere mult mai mare pentru a putea misca barierele/portile.

Download

smart-parking.rar

Bibliografie/Resurse

https://pages.mtu.edu/~suits/notefreqs.html

https://forum.arduino.cc/t/piezo-buzzer-win-and-fail-sound/133792

https://create.arduino.cc/projecthub/muhammad-aqib/rfid-and-keypad-door-lock-and-alert-system-using-arduino-60f050?ref=tag&ref_id=rfid&offset=0

https://www.youtube.com/watch?v=6gccSyp_uJQ

https://www.youtube.com/watch?v=Iz_iq9RwGkM

https://maxpromer.github.io/LCD-Character-Creator/ - for creating LCD smiley/frown face

Export to PDF