

Bluetooth and RFID door lock

Autor

- nume: Popescu Irina-Elena 331CB
- grupa: 331CB

Introducere

Door lock-ul se poate deschide atat prin RFID, folosindu-se de un tag, cat si prin modul de bluetooth, folosindu-se de o aplicatie de telefon. Exista doua functii: lock si unlock door. Cand se va apela una dintre aceste functii, se vor porni led-ul si buzzer-ul, si un servomotor se va roti 180 de grade, pentru a trage de un zavor. Am vrut sa fac un door lock pentru a mi-l pune la usa mea, deoarece daca imi inchid usa, mereu trebuie sa ma ridic sa o deschid pentru altcineva; dar acum, pot sa fac totul de pe telefon. 😊

Descriere generală

- Cand se va da lock, led-ul rosu se va aprinde de 2 ori, si buzzer-ul va suna tot de 2 ori, pe o anumita frecventa; la final, servomotorul se va roti cu 180 de grade.
- Cand se va da unlock, led-ul verde se va aprinde de 3 ori, si buzzer-ul va suna tot de 3 ori, pe o alta frecventa; la final, servomotorul se va roti cu 180 de grade, in sens opus.
- Daca tag-ul este gresit, led-ul albastru se va aprinde de 5 ori, si buzzer-ul va suna tot de 5 ori, pe o alta frecventa.
- Aplicatia de mobil a fost creata cu MIT App Inventor.

Pentru a putea da comenzile de lock/unlock din aplicatia de mobil, trebuie sa se introduca o parola in aplicatie.

Exista un unic tag care va porni functiile de lock/unlock folosind modulul de RFID, tag identificat prin numarul sau serial de 4 bytes.

Schema bloc:



Hardware Design

Lista de componente:

- Arduino UNO ATmega328p
- Breadboard
- Modul Bluetooth Master Slave HC-05
- Modul Tag RFID-RC522
- Micro servo motor SG90
- Led RGB cu catod comun
- Buzzer activ
- Fire tata-tata/mama-tata
- Rezistoare
- Tag RFID

Schema electrica:



Software Design

Am folosit [Arduino IDE](#) pentru a dezvolta programul. Bibliotecile folosite au fost:

- [AddicoreRFID.h](#) pentru modulul RFID-RC522.
- [SPI.h](#) pentru modulul RFID-RC522 - acest comunica prin SPI.
- [Servo.h](#) pentru micro servo motor SG90.

Variabilele globale sunt:

- `recv_bluetooth`: este 1 daca s-au primit date de la bluetooth, si 0 daca nu s-a primit nimic.
- `data_bluetooth`: byte-ul primit de la modulul de bluetooth.
- `door_locked`: reprezinta starea actuala a usii; ia valoarea 1/0 daca usa este inchisa/deschisa.
- `i`, `state`: variabile auxiliare.
- `checksumRFID`, `strRIFD`: variabile folosite in functiile pentru RFID.
- `myRFID`: obiectul de tip `AddicoreRFID`.
- `myServo`: obiectul de tip `Servo`.

Functii:

setPins()

Setez pinii pe care voi lega componentele de arduino pe **OUTPUT**:

- 7, 8, 9 pentru led-ul RGB.
- 2 pentru buzzer
- 5, 10 pentru RFID

si ii atasez obiectului de tip servo pinul PWM 3.

activatePins()

Scriu pe pinul 10 valoarea LOW si pe pinul 5 valoarea HIGH, pentru a activa RFID-ul.

setupComponents()

Initializez variabilele globale. Folosind *AddicoreRFID_Init()* initializez obiectul myRFID. Folosind *write* setez servo motorul la 0 grade, si variabila *door_locked* este 0, adica usa este deschisa. In plus, variabila *recv_bluetooth* este setata la 0 initial, pentru ca nu se primesc date de la bluetooth la setup.

setupBluetoothInterrupt()

Setez variabilele corespunzatoare pentru intreruperea pe USART, deoarece modulul de bluetooth comunica pe USART. Dezactivez intreruperile prin *cli()*. Setez *UBRR0* pe 103, pentru a configura baud rate-ul la 9600bps. Setez bitii *UCSZ01* si *UCSZ00* pe 1 in *UCSR0C* pentru a seta ca voi recepta/transmite date de 8 biti. Pentru a activa receptia/transmiterea, setez in *UCSR0B* biii *RXEN0* (activeaza receptia pe USART), *TXEN0*(activeaza transmiterea pe USART) si *RXCIE0* pe 1. La final, activez interuperile prin *sei()*.

setup()

Pornesc libraria SPI, deoarece este folosita de RFID. Apelez cele 3 functii descrise anterior si setez bitii corespunzatori pentru a activa interuperea pentru modulul de bluetooth.

set_RGB_led(int red_light, int green_light, int blue_light)

Pornesc led-ul RGB, folosind *analogWrite(pin, freq)*, unde *freq* = [0, 255], *pin* = {7, 8, 9}.

servo_lock(int startp, int endp)

Pornesc servo motorul, care face o rotire de la unghiul startp la endp, folosindu-se de functia *write*. Pentru a fi o miscare lina, folosesc un for, pentru a trece prin fiecare unghi de la startp la endp, si dupa fiecare miscare fac un *delay*.

Daca as face doar un *write(endp)*, s-ar misca prea repede.

servo_unlock(int startp, int endp)

Aplic aceeasi idee ca la functia anterioara, doar ca in sens opus.

door_routine(int redl, int greenl, int blucl, int delayt, int flickert, int freqb)

Rutina care se executa cand se da lock/unlock. De *flickert* ori, se executa:

- se porneste led-ul RGB
- se porneste buzzer-ul la o frecventa *freqb*, folosind functia *tone*
- delay
- se opreste led-ul RGB
- se opreste buzzer-ul
- delay

lock()

Funcția care se apelează când se închide ușa. Se apelează funcția *door_routine*, pentru a porni led-ul și buzzer-ul, și se porneste servo motorul.

unlock()

Funcția care se apelează când se deschide ușa. Se apelează funcția *door_routine*, pentru a porni led-ul și buzzer-ul, și se porneste servo motorul.

wrong_tag()

Dacă tag-ul pus la RFID este cel greșit, se apelează doar *door_routine*, dar nu se acționează și servo motorul.

check_tag()

Se verifică dacă tag-ul pus este cel corect. Se verifică fiecare byte din codul tag-ului cu numărul din tag-ul meu. În plus, ca o verificare extra, se verifică și checksum.

Reader-ul de RFID returnează un număr pe 5 bytes: primii 4 bytes reprezintă numărul unic al tag-ului, iar al 4-lea este checksum.

RFID_routine()

După ce s-a detectat un tag, dacă tag-ul este cel corect, se dă lock/unlock. Dacă tag-ul este greșit, se apelează funcția corespunzătoare. Dacă înainte se dă lock, se apelează unlock, și invers.

ISR(USART_RX_vect)

Rutina de tratare a interupției USART pentru bluetooth. Retin byte-ul primit prin *URD0*, și setez variabila *recv_bluetooth* pe 1, asta însemnând că am primit date de la telefon.

Variabila *recv_bluetoot* este **volatila** pentru a mă asigura că valoarea ei se schimbă imediat.

loop()

Se verifică dacă s-a găsit vreun tag, sau dacă se primesc date de la telefon pe bluetooth. Dacă s-a găsit tag, se verifică ce tag s-a găsit, și se verifică manual checksum. După ce se apelează funcția *RFID_routine()*, se dă *halt* la modul. Dacă s-au primit date de la telefon, se verifică ce funcție s-a primit (lock/unlock), se apelează funcția specifică, și se trimite înapoi la telefon valoarea corespunzătoare (1/0).

În rutina de întrerupere doar am setat variabila *recv_bluetooth* pentru că *lock()/unlock()* conțin **delay** și nu e bine să ai delay în interuperi.

Functionare:

Prin RFID, când se citește tag-ul bun, automat se dă lock(dacă starea anterioară a fost unlock), sau

unlock(daca starea anterioara a fost lock). Prin bluetooth(folosindu-se de o aplicatie de mobil pe Android), se poate alege daca se doreste lock sau unlock.

La bluetooth, daca se da lock, desi usa este locked, nu se intampla nimic. (la fel si pentru unlock).

Se retine starea usii printr-o varabila cu valorile posibile **0(unlocked)** sau **1(locked)**.

Indiferent daca s-a dat lock/unlock prin RFID sau bluetooth, se apeleaza aceeasi **rutina** pentru pornit led-ul, buzzer-ul si servo motorul.

Aplicatia de mobil pentru comunicarea cu modulul de bluetooth

Am folosit [MIT App Inventor](#) pentru a crea aplicatia de mobil ce comunica cu modulul de bluetooth.

Partea de frontend:

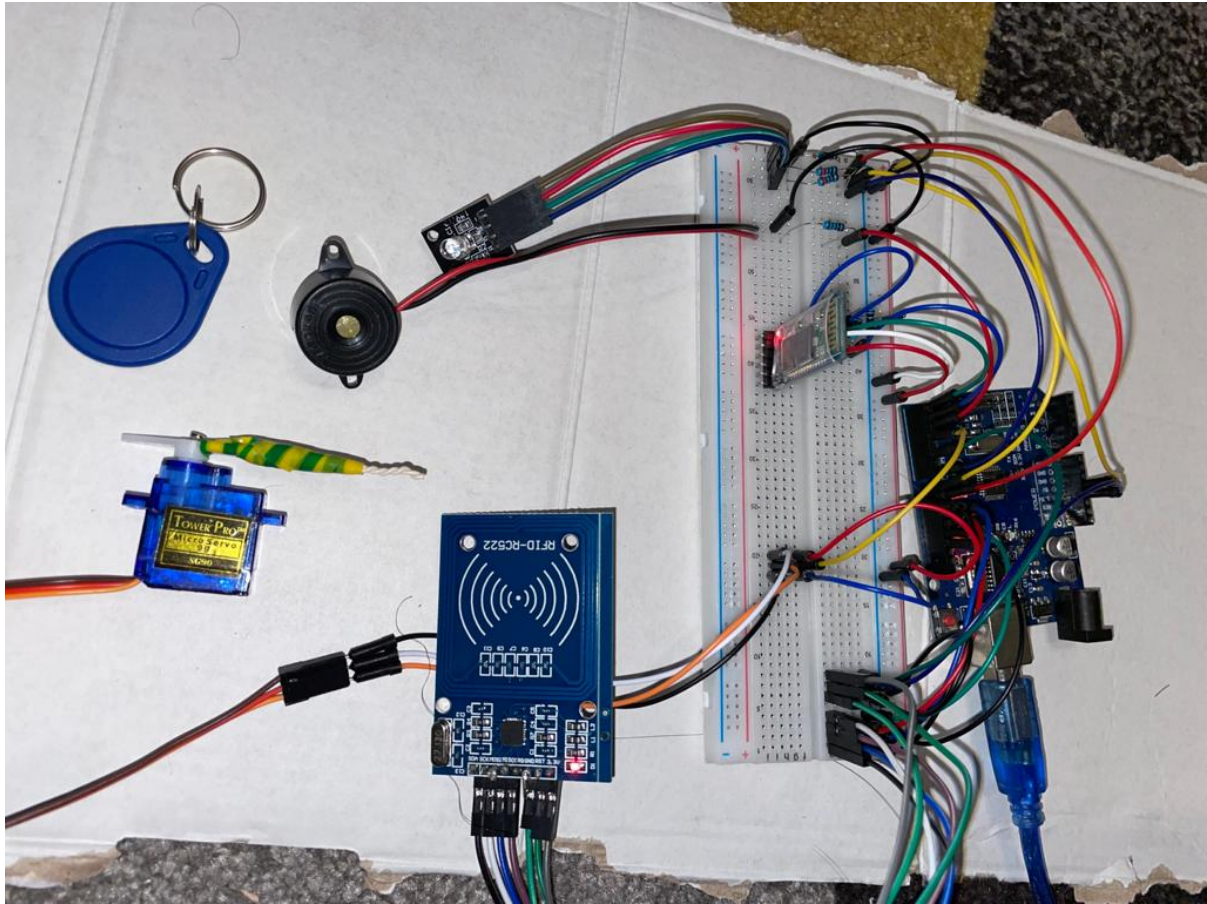


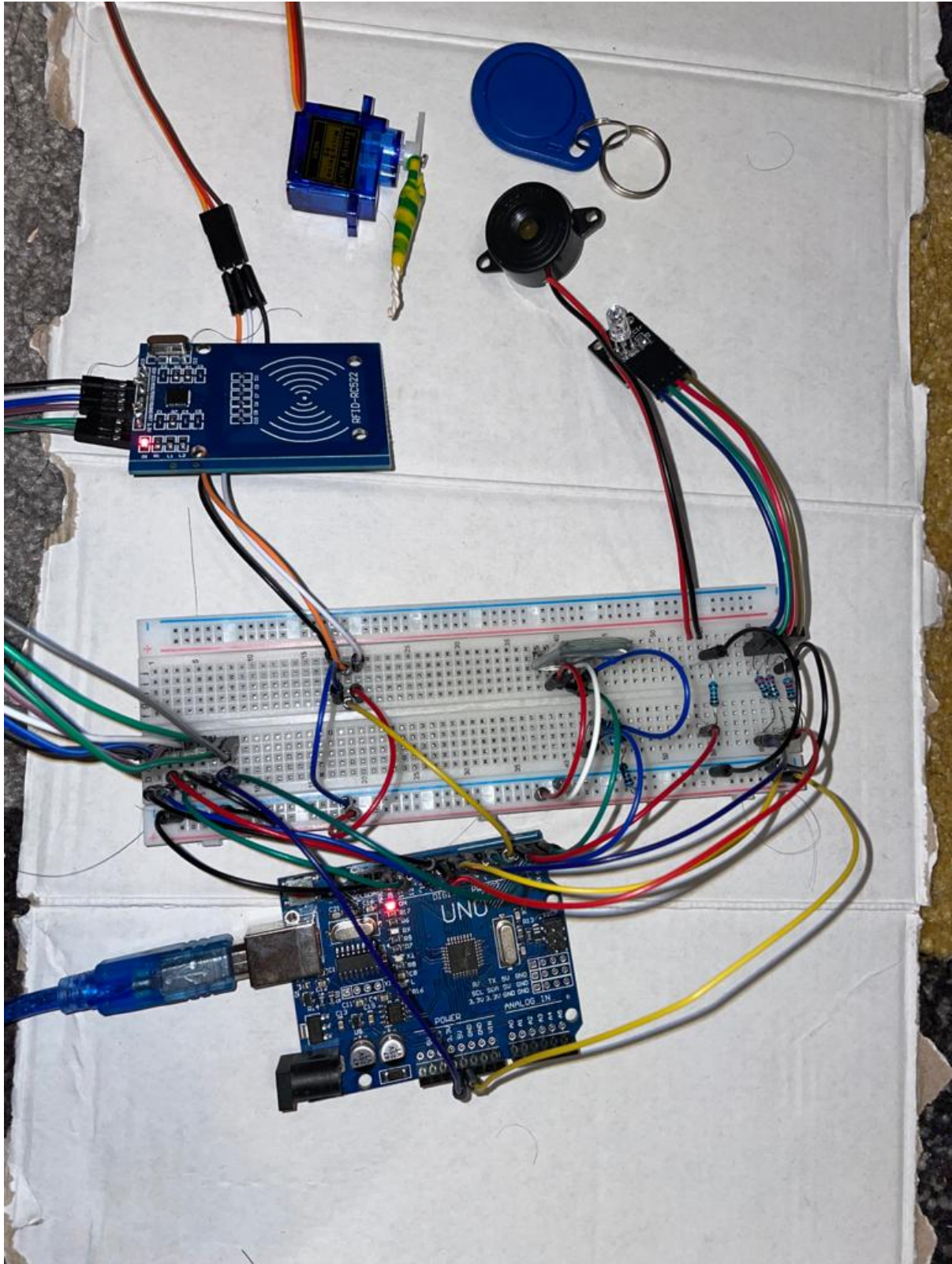
Partea de backend:



Rezultate Obținute

Asa arata proiectul, cu toate componentele:





Demo-ul care arata cum functioneaza proiectul:[demo](#)

Concluzii

REUSITE

- Am reusit sa implementez tot ce mi-am propus, si a iesit exact cum mi-am dorit.
- Am reusit sa fac o aplicatie de mobil functionala.

PROBLEME INTAMPINATE

- Am avut probleme la a face interuperea pentru USART pentru bluetooth. In rutina de interuperea apelam si lock()/unlock(), care contin **delay()**, si din cauza asta, nu se executau corect.
- Nu am reusit inca sa fac un log in care sa retin pe aplicatia de mobil cand s-a deschis/inchis usa mereu. Am creat o baza de date in aplicatie, dar nu le afisez inca cum vreau eu.

CE AM INVATAT DIN PROIECT?

- Am invatat sa folosesc modulul de bluetooth, si in acelasi timp, am inteles mai bine cum se face o intrerupere.
- Am invatat sa citesc mai bine datasheet-ul pentru placa Arduino UNO.
- Am invatat sa fac o logica pentru un proiect pe Arduino ce contine mai multe module.

Download

Arhiva cu codul sursa si README este: [arhiva_cod](#)

Jurnal

- 25.04.2021: am ales proiectul si am creat pagina.
- 03.05.2021 - 26.05.2021: lucrat la proiect.
- 26.05.2021: update la pagina + adaugat schema electrica.
- 27.05.2021: modificat functiile din software design + concluzii.
- 30.05.2021: adaugat link demo + arhiva cod + bibliografie

Bibliografie/Resurse

- [PM-laborator 0](#) - scrieri/citiri digitale
- [PM-laborator 1](#) - Intrerupere
- [PM-laborator 2](#) - USART
- [PM-laborator 3](#) - PMW - led RGB, servomotor
- [LED RGB](#)
- [BUZZER](#)
- [Servomotor](#)
- [RFID](#)
- [Bluetooth HC-05](#)
- [PDF](#)

From:

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

Permanent link:

http://ocw.cs.pub.ro/courses/pm/prj2021/alazar/bluetooth_door_lock



Last update: **2021/05/29 23:09**