Pametno parkiralište

Hajduković, Matija

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

NASLOV ZAVRŠNOG RADA

Završni rad

Matija Hajduković

Osijek, 2020.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 07.08.2020.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime studenta:	Matija Hajduković
Studij, smjer:	Preddiplomski sveučilišni studij Elektrotehnika i informacijska tehnologija
Mat. br. studenta, godina upisa:	4447, 24.09.2019.
OIB studenta:	64191501123
Mentor:	Izv. prof. dr. sc. Davor Vinko
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	Pametno parkiralište
Znanstvena grana rada:	**Elektronika (zn. polje elektrotehnika)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	07.08.2020.
Datum potvrde ocjene Odbora:	09.09.2020.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 10.09.2020.

Ime i prezime studenta:	Matija Hajduković
Studij:	Preddiplomski sveučilišni studij Elektrotehnika i informacijska tehnologija
Mat. br. studenta, godina upisa:	4447, 24.09.2019.
Turnitin podudaranje [%]:	15

Ovom izjavom izjavljujem da je rad pod nazivom**: Pametno parkiralište**

izrađen pod vodstvom mentora Izv. prof. dr. sc. Davor Vinko

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA

o odobrenju za pohranu i objavu ocjenskog rada

kojom ja __Matija__Hajduković_____, OIB: __64191501123_____, student/ica Fakulteta elektrotehnike,
računarstva i informacijskih tehnologija Osijek na studiju _Preddiplomski sveučilišni studij Elektrotehnika i informacijska tehnologija_, kao autor/ica ocjenskog rada pod naslovom: _Pametno parkiralište**,**

dajem odobrenje da se, bez naknade, trajno pohrani moj ocjenski rad u javno dostupnom digitalnom repozitoriju ustanove Fakulteta elektrotehnike, računarstva i informacijskih tehnologija Osijek i Sveučilišta te u javnoj internetskoj bazi radova Nacionalne i sveučilišne knjižnice u Zagrebu, sukladno obvezi iz odredbe članka 83. stavka 11. *Zakona o znanstvenoj djelatnosti i visokom obrazovanju* (NN 123/03, 198/03, 105/04, 174/04, 02/07, 46/07, 45/09, 63/11, 94/13, 139/13, 101/14, 60/15).

Potvrđujem da je za pohranu dostavljena završna verzija obranjenog i dovršenog ocjenskog rada. Ovom izjavom, kao autor/ica ocjenskog rada dajem odobrenje i da se moj ocjenski rad, bez naknade, trajno javno objavi i besplatno učini dostupnim:

a) široj javnosti

b) studentima/icama i djelatnicima/ama ustanove

c) široj javnosti, ali nakon proteka 6 / 12 / 24 mjeseci (zaokružite odgovarajući broj mjeseci).

**U slučaju potrebe dodatnog ograničavanja pristupa Vašem ocjenskom radu, podnosi se obrazloženi zahtjev nadležnom tijelu Ustanove.*

__Osijek, 10.09.2020._____ _____

(mjesto i datum) (vlastoručni potpis studenta/ice)

SADRŽAJ:

1. UVOD

Zadatak ovog završnog rada je izrada sustava pametnog parkirališta pomoću arduino mikroupravljača. Parkiralište se sastoji od senzora koji detektiraju ulazak i izlazak vozila te se broj slobodnih mjesta bilježi na LCD zaslonu. Ulazak u parkiralište je napravljeno pomoću RFID kartica i aktuatora, a izlaz pomoću tipkala i aktuatora.

Kako bi cijeli sustav funkcionirao, korišteno je znanje stečeno na fakultetu iz kolegija „Arhitektura računala", „Programiranje 1" te „Elektronika 1". U nastavku će se prikazati specifikacija svakog pojedinog dijela te objasniti kôd napisan za mikroupravljač.

Rad se sastoji od 2 dijela: programskog, kojim mikroupravljač komunicira sa ostalim uređajima, te fizičkog, na kojemu su te komponente ugrađene.

1.1 Zadatak završnog rada

Zadatak završnog rada je izrada sustava pametnog parkiralište. Sustav se sastoji od senzora koji detektiraju ulazak i izlazak vozila, te bilježe ukupan broj vozila u memoriju i ispisuju ga na LCD zaslonu. Senzore, aktuatore i LCD zaslon potrebno je implementirati u sustav temeljen na mikroupravljaču.

2. PREGLED PODRUČJA TEME

Kao stvarni primjer pametnog parkirališta te kao aktualni praktični doseg može se uzeti Parklio™ [8] koja je samo jedna od tvrtki koja se bavi pametnim parkiralištima. Za razliku od makete koja je izrađna u sklopu ovog rada, Parklio™ parkirališta imaju i neke dodatne opcije koje se mogu dodatno implementirati na maketu. Od sustava video nadzora, aplikacija koje pomoću IoT-a (*eng. internet of things*) prate koliko je slobodnih mjesta te barijera koja omogućuju pojedinačno zatvaranje parkirnih mjesta. Ipak između makete i prakričnog parkirališta postoje sličnosti. Oboje prate koliko trenutno slobodnih mjesta ima te je li parkiralitše puno/prazno te to rade pomoću senzor. Također oboje omogućuju ulaz, odnosno izlaz, iz parkirališta pomoću rampi. Što se tiče upravljanja samog parkirališta ne može se sa sigrunošću reći na kojem sustavu je temeljen, ali je moguće sa velikom sigurnošću reći da se ne bazira na Arduino mikroupravljaču pošto su njegove specifikacije puno slabije te ne ispunjavaju minimalne uvjete za stvarnu primjenu.

Ako pogledamo drugi primjer pametnog parkirališta koje je kao koncept predstavila tvrtka Huawei[9] možemo vidjeti kako je dodatno proširena ideja IoT-a te kako je na već postojeću infrastrukturu senzora omogućeno praćenje broja slobodnih mjesta. Poveznica između ovog projekta i makete postoje, ali su vrlo male. Obije prate broj automobila koji se nalaze na parkiralištu putem senzora, ali maketa nije povezana s internetom te nema nikakvu aplikaciju koja pokazuje koliko trenutno ima slobodnih mjesta. No, to bi sigurno bila sljedeća stepenica u nadogradnji makete.

3. IDEJNI PROJEKT

3.1 Tehnički opis

Rad je zamišljen kao maketa i namijenjen je u edukacijske svrhe. Čini ga jednoetažna zgrada, s jednim ulazom i izlazom, koja ima potpuno automatizirani proces praćenja broja automobila i podizanja te spuštanja rampi. Na samoj maketi se nalaze 4 parkirna mjesta te odvojeni ulaz i izlaz.

3.2 Pokazatelji ispravnosti tehničkog rješenja

Sustav pametnog parkirališta predstavlja nadogradnju već postojećeg parkirališta pomoću različith elektroničkih uređaja. Tehnički sustav sadrži elemente koji se električki napajaju, ali su smješteni unutar zaštitnog kućišta te su time izolirani od vanjskih uvjeta. Nadalje, na mjestima prolaska kabela kroz različite dijelove makete postavit će se zaštitna vodilica kako ne bi došlo do nepotrebnog izlaganja kabela te kako bi se dodatno zaštitilo od vanjskih utjecaja. Svi dijelovi koji su normalno pod naponom će se zaštititi od slučajnog dodira. Također, cijeli sustav upravlja mikroupravljač, što znači da ljudska prisutnost za kontroliranje rampe na samom mjestu nije potrebna, pa nema ni eventualne opasnosti po ljudski život. U sustavu nema dijelova koji negativno utječu na okoliš te se u slučaju eventualne zamjene svi elementi mogu odložiti na za to predviđeno mjesto. S obzirom na to da parkiralište prati i broj automobila koji se u tom trenutku tamo nalaze, korisnik koji eventualno dolazi na parking neće ni moći pristupiti parkingu ukoliko su sva mjesta zauzeta, a to će moći vidjeti na LCD modulu.

4. ALATI

4.1 Arduino

Upravljačka jedinica koja kontrolira cijeli process je Arduino Uno (Slika 4.1.), koji se temelji na ATmega32U4 mikroupravljaču. Ova je jedinica odabrana jer je vrlo jednostavna za upravljanje. Povezuje se s računalom putem USB kabela te se programira pomoću Arduino ISP razvojnog okruženja. Sama pločica se nalazi ispod makete parkirališta te je povezana sa svakim od izvršnih uređaja. Specifikacije upravljačke jedinice dostupne su u tablici 4.1. [1]

Slika 4.1.: Arduino Uno mikroupravljač[1]

Operativni napon	Broj D/A pinova	Operativna frekvencija	Način povezivanja	Veličina pločice [mm]
5V	20 (6 može proizvesti PWM singla)	16 MHz	USB tip B	68.8 / 53.4

Tablica 4.1.: Specifikacije Arduino Uno upravljačke jedinice

4.2 Servo motor

Na ulazu i izlazu samog parkirališta nalaze se dvije rampe koje su upravljane pomoću servo motora. Servo motor je posebna vrsta motora koja ima najčešće mogućnost zakretanja od 0° do 180°. Rotacijom motora se upravlja pomoću PWM signala (eng. *Pulse-width modulated signal*), odnosno *Duty Cycla*. U ovom radu korišteni su SG-90 servo motori prikazani na slici 4.2. Specifikacije SG-90 nalaze se u tablici 4.2., a način spajanja sa Arduinom objašnjen je u tablici 4.3. [3]

Slika 4.2.: Servo motor SG-90 [3]

Operativni napon	Okretni moment	Kut rotacije	Težina
5V	2.5 kg/cm	0° - 180°	9 g

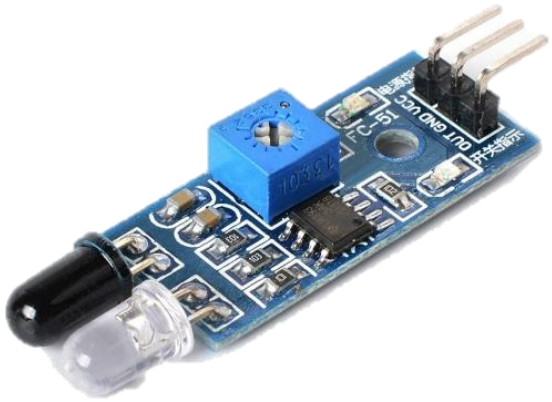
Tablica 4.2.: Specifikacije SG-90 servo motora

Smeđa žica	Povezati na uzemljenje (GND)
Crvena žica	Povezati na izvor napajanja (+5V)
Narančasta žica	Signal koji kontrolira motor(PWM signal)

Tablica 4.3.: Način spajanja SG-90 servo motora

4.3 Senzori

Kako bi se osiguralo da su auti zaista ušli ili izašli iz parkirališta korišteni su infracrveni senzori na ulazu/izlazu. IR senzor korišten u ovom radu zasniva se oko LM393 komparatora (Slika 3.3.). Način na koji senzor radi je da bijela LED-ica (eng. *Light Emmiting Diode*), koja služi kao odašiljač, zrači infracrvenu zraku koja se od predmeta odbija do crne LED-ice, koja služi kao prijemnik infracrvene zrake. Specifikacije uređaja dane su u tablici 4.4. [4], a način spajanja je prikazan u tablici 4.5.



Slika 4.3.: LM393 IR senzor[4]

Operativni napon	Udaljenost detekcije	Kut detekcije	Veličina pločice[cm]
3.3 do 5V	2 ~ 30cm	35°	3.2 / 1.4

Tablica 4.4.: Specifikacije senzora

Gnd pin	Povezati na uzemljenje (GND)
Vcc pin	Povezati na izvor napajanja (+5V)
Out pin	Povezati na analogni pin Arduina

Tablica 4.5.: Način spajanja senzora

4.4 RFID čitač

Na ulazu u parkiralište, prije rampe, nalazi se RFID (eng. *Radio-frequency identification*) čitač kako bi se osigurao ulazak samo onih koji imaju odgovarujuću RFID karticu. U ovom projektu korišten je RC522 modul (Slika 4.4.) zbog svoje jednostavnosti i relativno malih dimenzija. Pomoću RC522 modula moguće je očitati i pisati na RFID karticu. RC522 modul zrači elektromagnetskim poljem zbog kojeg kartica odašilje svoje identifikacijske podatke koje čitač zatim očita. Neke od karakteristika RC522 modula nalaze se u tablici 4.6. [5], a način spajanja na mikroupravljač opisan je u tablici 4.7.

Slika 4.4: RC522 RFID čitač[5]

Operativni napon	Udaljenost detekcije	Komunikacijski protokoli	Veličina pločice[mm]	Operativna frekvencija
2.5 do 3.3V	5cm	SPI,I2C i UART	59.44 / 39.89	13.56 MHz

Tablica 4.6.: Specifikacije RC522 modula

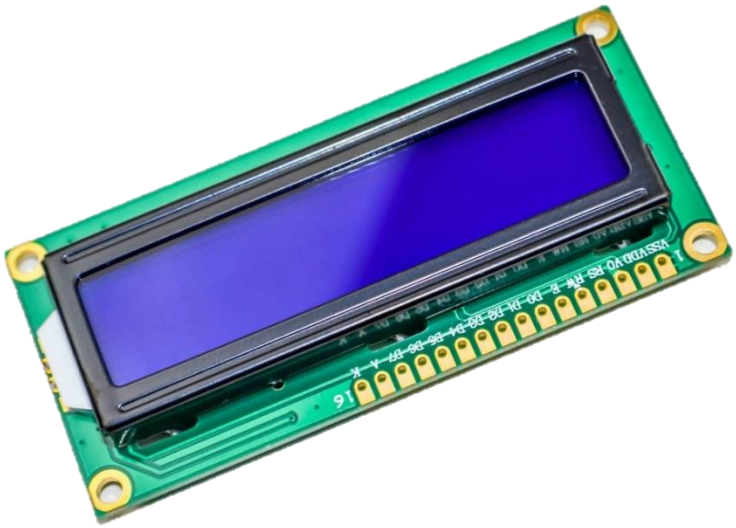
SDA	Digitalni pin 10
SCK	Digitalni pin 13
MOSI	Digitalni pin 11
MISO	Digitalni pin 12
IRQ	*Interrupt* pin, nije potrebno povezati
GND	Povezati na uzemljenje (GND)

RST	Povezati na PWM pin broj 9 Arduina
3.3V	Povezati na izvor napajanja (3.3V)

Tablica 4.7.: Način spajanja RC522 modula

4.5 LCD zaslon

Za razliku od običnog, pametno parkiralište prati koliko automobila se trenutačno nalazi na parkiralištu te ga može automatski zatvoriti. Sve te informacije (radi li parkiralište, koliko ima slobodnih mjesta) ispisuju se na LCD zaslonu. U ovom radu korišten je IC2 LCD zaslon (Slika 4.5.) koji ima mogućnost ispisa 32 znaka. Specifikacije zaslona nalaze se u tablici 4.8. [6], a način spajanja u tablici 4.9.



Slika 4.5: 16x2 IC2 LCD modul[6]

Operativni napon	Veličina	Veličina pločice[mm]	Boja znakova
5V	16 znakova u 2 reda	80/36/20	Bijela

Tablica 4.8.: Specifikacije I2C LCD zaslona

GND	Povezati na uzemljenje (GND)
Vcc	Povezati na izvor napajanja (+5V)
SDA	Analogni pin A5
SCL	Analogni pin A4

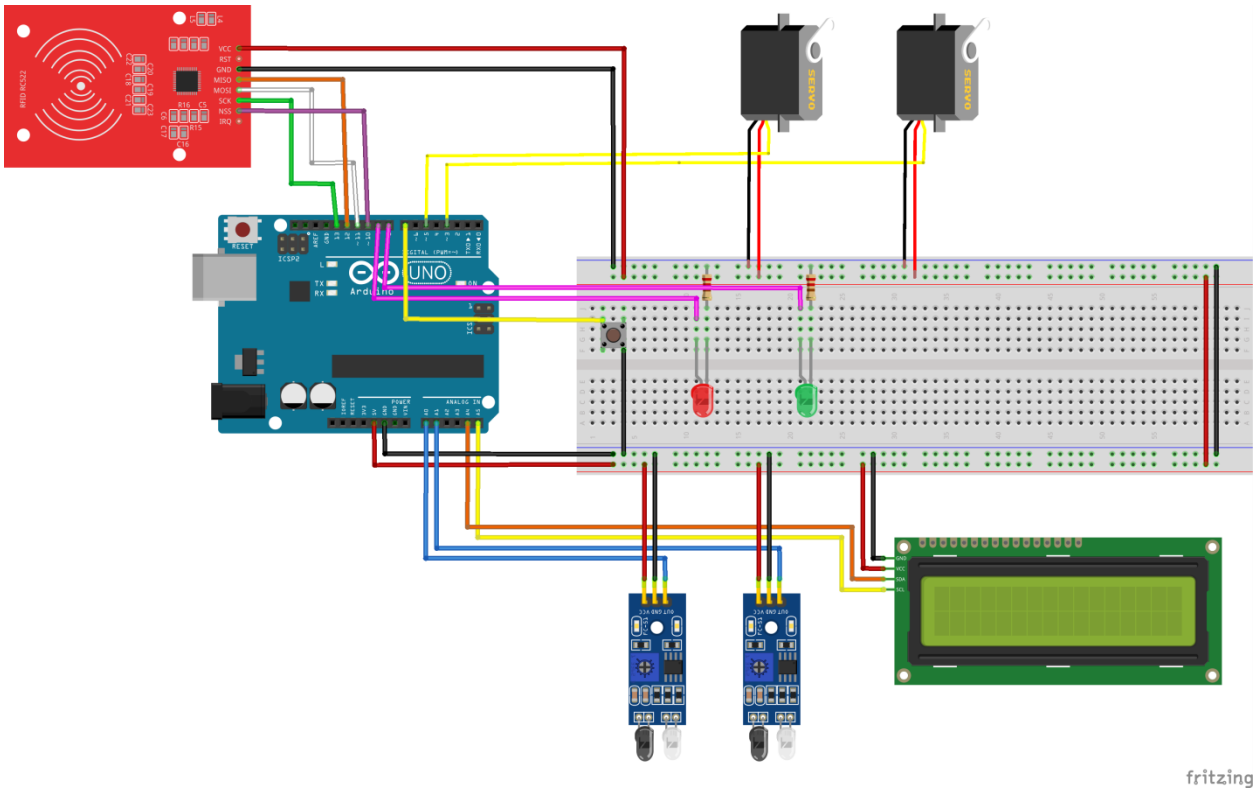
Tablica 4.9.: Način spajanja I2C LCD zaslona

4.6 Programska podrška

Kako bi se mikroupravljač programirao, prvo je potrebno povezati ga s računalom koje mora imati za to potreban program. Za njegovo programiranje korišten je Arduino IDE (eng. *Integrated Development Environment*). Programski jezik pomoću kojeg se programira Arduino je C++.

5. SHEMA SKLOPOVLJA

Elektroničko sklopovolje povezano je na način koji je opisan u 4. poglavlju te su korišteni kablovi za spajanja ekspermintalne pločice. Cjelokupna shema prikazana je na slici 5.1.



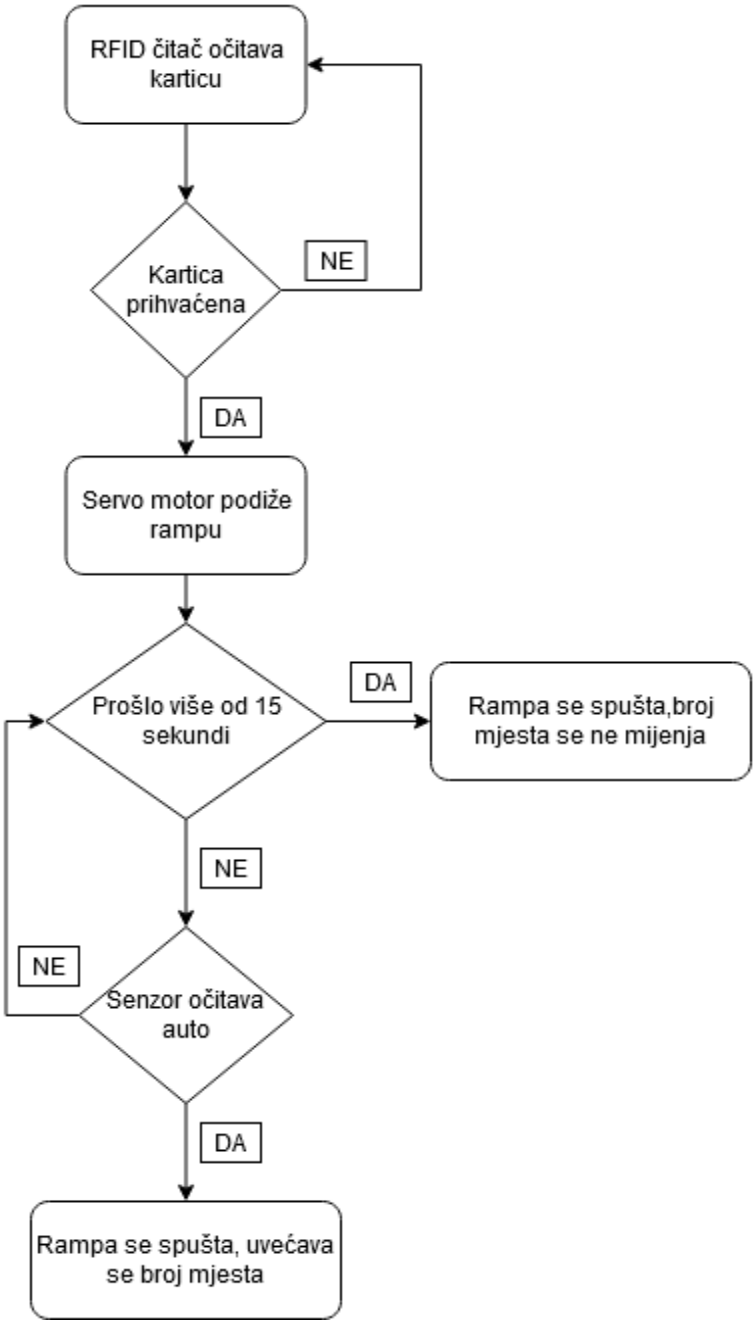
Slika 5.1.: Shema spajanja pametnog parkirališta izrađena u Fritzing programu

Ispred ulazne rampe se nalazi RFID čitač kartice koji nakon prislonjene, odgovarajuće, RFID kartice podiže rampu. Ulazni IR senzor očitava je li auto doista ušao te, ako je, smanjuje broj slobodnih mjesta na parkiralištu i spušta rampu. Broj slobodnih mjesta se prati na LCD zaslonu.

Ukoliko se popune sva mjesta (broj parkirnih mjesta je 4), servo motori više neće podizati rampu te se na LCD zaslonu ispisuje da je parkiralište puno.

6. KÔD

6.1 Ulaz u parkiralište



Slika 6.1.: Dijagram toka ulaska u parkiralište

Na slici 6.1. možemo vidjeti kako je prije svega potrebno imati odgovarajuću RFID karticu kako bi se rampa podigla. Nakon toga senzor očitava je li auto doista ušao u parkiralište te, ako je, uvećava broj mjesta. Na slici 6.2. se može vidjeti kôd koji je korišten za očitavanje RFID kartica.

```
void RFIDReader(){

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

  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }

  content.toUpperCase();
  if (content.substring(1) == "70 9D 0E A3")
  {
    delay(500);
    EntranceRampUp();

  }
  }
```

Slika 6.2.: RFIDReader() funkcija

U principu RFID čitač će neprestano očitavati kartice te će ovaj kod zapravo samo provjeravati je li to odgovarajuća kartica. Nakon što se kartica približi RFID čitaču, njezin identifikacijski broj se sprema u varijablu *content* te ako je odgovarajuća poziva se funkcija *EntranceRampUp().*Trenutno je samo jedna kartica odgovarajuća, ona kojoj je identifikacijski broj 70 9D 0E A3. *RFIDReader* funkcija se poziva u glavnom dijelu programa.

Slika 6.3. prikazuje *EntranceRampUp()* i *EntranceRampDown()* funckije su zadužene za dizanje i spuštanje rampi, unutar njih još nalaze dijelovi koji se odnose na senzore i LED diode. Servo motor će se pomoću *for* petlje podići za 90° u inkrementima od 5°. Unutar *while* petlje očitava se je li auto ušao ili nije te je li tipkalo na izlazu pritisnuto, u slučaju da auto želi napustiti

parkiralište. *While* petlja u sebi ima brojač koji ne dopušta da se petlja vrti više od 15 sekundi, time osiguravamo da sama rampa ne ostane otvorena duže od 15 sekundi. IR sentor daje vrijednosti od 0 do 1024. Što je vrijednost bliža nuli, to je predmet bliže senzoru. Vrijednost sa senzora se neprestano očitava te sprema u varijablu *x*. Ako je ona manja od 100, to znači da je auto prošao te se uvećava broj auta na parkiralištu. Nakon toga se poziva funkcija *EntranceRampDown()* koja spušta rampu. Također se vodi računa da *ParkingSpots* varijabla ne može prijeći broj 4 jer je toliko parkirnih mjesta na parkiralištu.

```
//Entrance Ramp-----------------------
void EntranceRampUp(){
  for (pos = 0; pos <= 90; pos += 5) {
    Entrance_servo.write(pos);
    delay(15);
  }

  int x;
  unsigned long starttime = millis();
  unsigned long endtime = starttime;
  while((endtime-starttime) <=15000){
    if(digitalRead(ExitButton)==LOW){
    ExitRampUp();
    }
    x = analogRead(A0);
    digitalWrite(GreenLED,HIGH);
    digitalWrite(RedLED,LOW);
    if(x < 100){
        ParkingSpots++;
        if(ParkingSpots>=4){
          ParkingSpots=4;
        }
        delay(100);
        break;
    }
    endtime = millis();
   }
 EntranceRampDown();
 }

void EntranceRampDown(){
  for (pos = 90; pos >= 0; pos -= 5) {
    Entrance_servo.write(pos);
    delay(15);
  }
  digitalWrite(GreenLED,LOW);}
```

Slika 6.3.: EntranceRamp funkcije za spuštanje/podizanje rampe

6.2 Ispis slobodnih mjesta

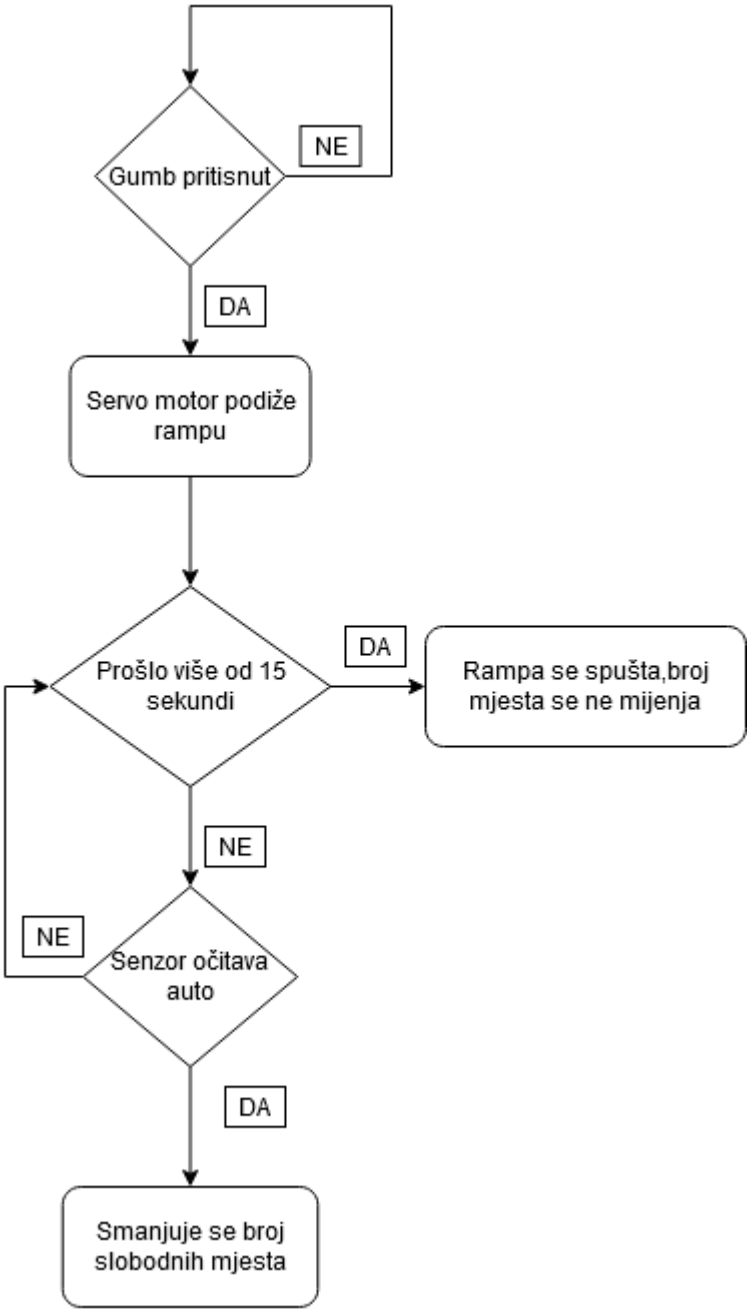
Sve informacije o stanju parkirališta (je li ono puno ili prazno, koliko ima slobodnih mjesta…) prikazuju se na LCD zaslonu.

```
CurrentSpots=4-ParkingSpots;
switch(CurrentSpots){

  case 4:
  lcd.clear();
  EmptyPark();
  break;

  case 3:
  lcd.setCursor(0,1);
  lcd.print(String(CurrentSpots));
  break;

  case 2:
  lcd.setCursor(0,1);
  lcd.print(String(CurrentSpots));
  break;

  case 1:
  lcd.setCursor(0,1);
  lcd.print(String(CurrentSpots));
  break;

  case 0:
  lcd.clear();
  ClosePark();
  break;

  default:

  break;
}
```

Slika 6.4.: Ispis na LCD zaslon

Prije svega potrebno je utvrditi koliko auta se doista nalazi na parkiralištu. To se odrađuje u prethodnom kodu koji je prikazan na slici 6.3. Varijabla *ParkingSpots* prati koliko je auta ušlo/izašlo, a *CurrentSpots* varijabla broji koliko je ostalo slobodnih mjesta te se ona ispisuje na LCD zaslon. Pomoću *switch case* petlje u ovisnosti o vrijednosti *CurrentSpots* varijable ispisuje se broj slobodnih parking mjest te se zatvaraju ulaz ili izlaz iz parkirališta.

6.3 Izlaz iz parkirališta



Slika 6.5.: Dijagram toka za izlaz iz parkirališta

Slika 6.5. prikazuje dijagram toka izlaza iz parkirališta. Za izlaz je potrebno pritisnuti dugme te se onda odrađuje ista provjera kao i kod ulaza.

Izlaz iz parkirališta napravljen je na identičan način kao i ulazak u parkiralište. Jedina razlika je što se varijabla *ParkingSpots* umanjuje i vodi se računa o tome da ne bude negativna te što se provjerava je li netko pokušava, pomoću RFID kartice, ući u parkiralište.

```
//Exit Ramp--------------------------------
void ExitRampUp(){
  for (pos2 = 180; pos2 >= 90; pos2 -= 5) {
    Exit_servo.write(pos2);
    delay(15);
  }

  int x;
  unsigned long starttime = millis();
  unsigned long endtime = starttime;
  while((endtime-starttime) <=15000){
    if (mfrc522.PICC_IsNewCardPresent()) {
     RFIDReader();
     }
    x = analogRead(A1);
    digitalWrite(GreenLED,HIGH);
    digitalWrite(RedLED,LOW);
    if(x < 120){
        ParkingSpots--;
        if(ParkingSpots<0){
           ParkingSpots=0;
        }
        delay(100);
        break;
    }
    endtime = millis();
   }
   ExitRampDown();
 }

void ExitRampDown(){
  for (pos2 = 90; pos2 <= 180; pos2 += 5) {
    Exit_servo.write(pos2);
    delay(15);
  }
  digitalWrite(GreenLED,LOW);}
```

Slika 6.6.: ExitRamp funkcije za spuštanje/podizanje rampe

6.4 Zatvaranje parkirališta

Ako je parkiralište prazno ili puno, potrebno je izlaz, odnosno, ulaz zatvoriti dok se stanje ne promjeni. Funkcije koje zatvaraju parkiralište pozivaju se iz *switch case* petlje koje je prikazana na slici 6.4., dok je kôd tih funkcija prikazan na slici 6.7.

```
//Closing / Opening park----------------------

void ClosePark(){
  while(0==CurrentSpots){
     Entrance_servo.write(0);
     lcd.setCursor(0,0);
     lcd.print("Park is full");

     if(digitalRead(ExitButton)==LOW){
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Available spots: ");
      ExitRampUp();
      break;
      }
   }

}

void EmptyPark(){
  while(4==CurrentSpots){
     Exit_servo.write(180);
     lcd.setCursor(0,0);
     lcd.print("Park is empty");

     if (  mfrc522.PICC_IsNewCardPresent()) {
     RFIDReader();
     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("Available spots: ");
     break;
     }

   }
}
```

Slika 6.7.: Funkcije za zatvaranje parkirališta

Funkcija *ClosePark()* zatvara ulaz tako da se ulazni servo motor ne podiže. Ako se pritisne tipka na izlazu, to znači da će se osloboditi mjesto te da parkiralište više nije puno.

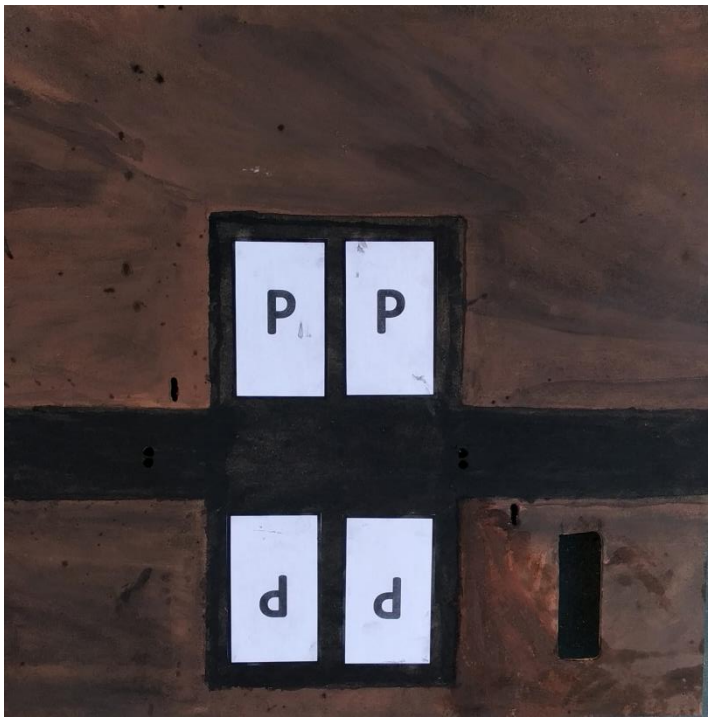
Funkcija *EmptyPark()* radi slično što i prethodna funckija, osim što ova zatvara izlaz iz parkirališta na isti princip. Ako se pojavi kartica na RFID čitaču, to znači da netko pokušava ući u parkiralište te ono više nije prazno, pa se poziva funkcija *RFIDReader()* koja će provjeriti je li ta kartica odgovarajuća.

7. IZRADA MAKETE

Izrada makete se odvijala u 3 faze:

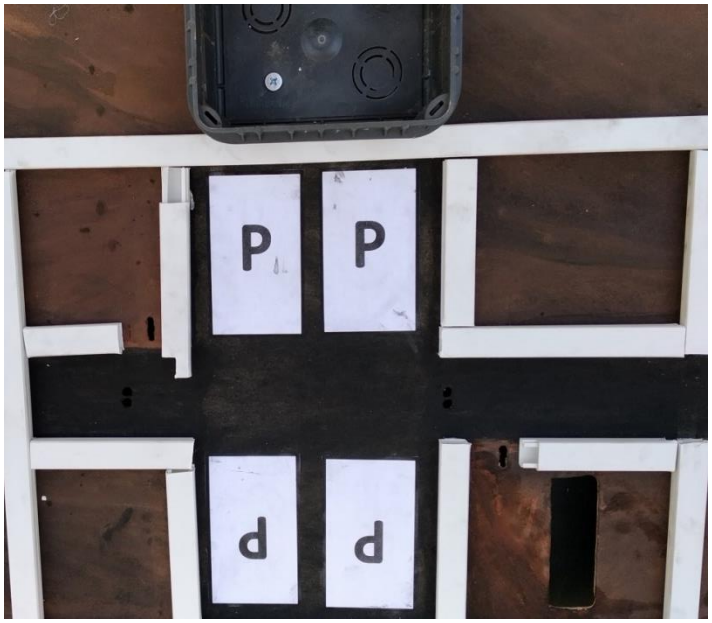
1. faza rezanje i bojanje,
2. faza postavljanje kanalica i razvodne kutije,
3. faza postavljanje elektronike te spajanje.

Cijela maketa je izrađena na drvenoj podlozi. Ploču je prvobitno trebalo izrezati na 40 cm * 40 cm pomoću pile, zatim izdubiti potrebne rupe kako bi se mogla postaviti sva elektronika, uključujući razvodnu kutiju u kojoj se nalazi Arduino. Nakon toga je uslijedilo bojanje makete.



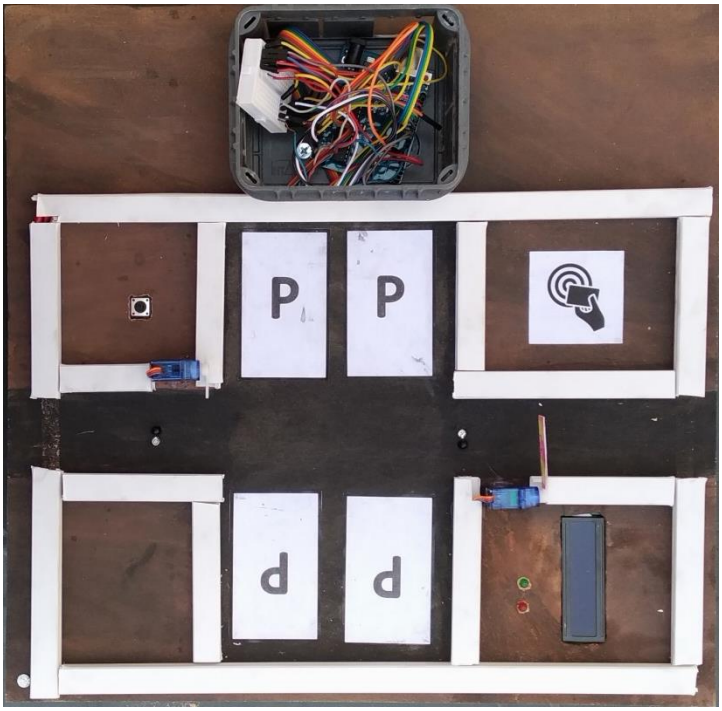
Slika 7.1.: 1. faza izrade makete

Potom je uslijedilo postavljanje kanalica kroz koje će se provoditi određeni kablovi te razvodna kutija u kojoj se nalazi Arduino. Također su napravljene oznake parkirnih mjesta.



Slika 7.2.: 2. faza izrade makete

Na kraju je još bilo potrebno montirati svu elektroniku te ju povezati putem 0.1 mm kablova s mikroupravljačem.



Slika 7.3.: 3. faza izrade makete

8. ZAKLJUČAK

Cilj ovog rada bio je prikazati rad pametnog parkirališta, napraviti funkcionalnu maketu koristeći Arduino mikroupravljač te potrebne elektroničke uređaje koji su s njim kompatibilni. Pokazano je što je sve potrebno za njegovu izradu te koje sve dijelove treba sadržavati.

S obzirom da je ovo ipak samo maketa, ne može se reći da je ona dobar primjer stvarnog pametnog parkirališta jer postoji još puno parametara kod same izrade parkirališta koji nisu obrađeni u ovom radu. Također, pravo fizičko parkiralište ne može biti bazirano na Arduino mikroupravljaču nego se ipak mora koristiti neki jači mikroupravljač. Ipak, ovaj rad je dobar pokazatelj što bi pametno parkiralište trebalo predstavljati te na koji način ono funkcionira.

Također, rad je moguće i dodatno unaprijediti. Moguća je izrada mobilne ili web aplikacije koja prati koliko slobodnih mjesta trenutno ima na parkiralištu, što bi omogućilo vozaču da zna može li ući u parkiralište i prije nego što dođe do njega. Također, moguće je ugraditi videonadzor kojim će se nadgledati cijelo parkirlište, a moguća je i izgradnja dodatnog kata kako bi se povećao broj parkirnih mjesta.

LITERATURA

[1]. Arduino Uno: https://store.arduino.cc/arduino-uno-rev3 (20.7.2020.)

[2]. Programska podrška, Arduino ISP: https://www.arduino.cc/en/Tutorial/ArduinoISP (20.7.2020.)

[3]. SG-90 servo motor: https://components101.com/servo-motor-basics-pinout-datasheet (20.7.2020.)

[4]. Infracrveni sensor:http://qqtrading.com.my/ir-infrared-obstacle-detaction-sensor-module-fc-5 (20.7.2020.)

[5]. RC522 RFID čitač kartica: https://components101.com/wireless/rc522-rfid-module (20.7.2020.)

[6]. Specifikacije LCD zaslona: https://e-radionica.com/blog/2015/08/19/kkm-lcd-16x2/ (20.7.2020.)

[7]. Program za izradu diagram toka: https://app.diagrams.net/ (20.7.2020.)

[8]. Parklio™: https://parklio.com/hr/#home (29.7.2020.)

[9]. Huawei Smart Parking - https://www.huawei.com/minisite/iot/en/smart-parking.html (29.7.2020.)

SAŽETAK

Ovaj rad bazirao se na izradi pametnog parkirališta koristeći Arduino mikroupravljač. Za izradu makete parkirališta korišteni su: Arduino mikroupravljač, aktuatori, infracrveni senzori, RFID čitač kartice, LCD zaslon te dugme. Mikroupravljač je isprogramiran u Arduino IDE razvojnom okružju. Kôd se sastoji od više dijelova: 1. RFID čitač kartice koji registrira dolazak auta, 2. dio koji kontrolira rampe, prati koliko se automobile trenutno nalazi na parkiralištu i potvrđuje je li automobil doista ušao/izašao iz parkirališta i 3. dio koji provjerava je li parkiralište puno te, ako je, zatvara ga. Također je pomoću drvene ploče izrađena fizička maketa parkirališta na koju su ugrađeni svi elektronički dijelovi. Sve u svemu, pametno parkiralište je testirano te obavlja sve funckije koje su zamišljene.

Ključne riječi: Pametno parkiralište, infracrveni senzor, LCD zaslon, RFID čitač, Arduino, Arduino IDE

SUMMARY

The aim of this bachelor paper is to present the making of a smart parking lot using Arduino microcontroller. For our model we used an Arduino microcontroller, servo motors, infrared sensors, RFID card reader, LCD screen, and a button. The microcontroller is programmed using Arduino IDE software. Its code consists of multiple parts: the first part is for the RFID reader that registers vehicles, the second is for the servos that control the ramp as well as the sensor that confirms that the car has entered/exited the lot and keeps track of how many cars are in the lot, and the third part that closes the parking lot if it is full. In addition, the model of the park is made using woodcarving tool and plywood. All in all, the parking lot has been tested and has preformed all functions intended.

Keywords: Smart parking lot, infrared sensor, RFID reader, LCD screen, Arduino, Arduino IDE

ŽIVOTOPIS

Matija Hajduković rođen je 15. lipnja 1998. u Osijeku. Provodi svoj život u rodnom gradu gdje 2012. godine završava Osnovnu školu Frana Krste Frankopana. Kasnije te iste godine upisuje Elektrotehničku i prometnu školu u Osijeku koju nakon 4 godine završava te stječe zvanje tehničara za elektroniku. Nakon završetka srednje škole upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, smjer elektrotehnika, ali se na 2. godini studija opredjeljuje za smjer Komunikacije i informatika.

PRILOG

```
#include <SPI.h>

#include <MFRC522.h>

#include <Servo.h>

#include <Wire.h>

#include <LiquidCrystal_I2C.h>


#define SS_PIN 10

#define RST_PIN 9

#define servoPin1 3

#define servoPin2 5


MFRC522 mfrc522(SS_PIN, RST_PIN);

Servo Entrance_servo;

Servo Exit_servo;

LiquidCrystal_I2C lcd(0x27, 16, 2);



int ExitButton = 7;

int ParkingSpots = 0;

int CurrentSpots;
```

```
int pos = 0;

int pos2 = 180;

int GreenLED = 8;

int RedLED = 9;

void setup()

{

  Serial.begin(9600);   // Initiate a serial communication

  SPI.begin();          // Initiate  SPI bus

  mfrc522.PCD_Init();   // Initiate MFRC522



  //LED and button initialization

  pinMode(ExitButton, INPUT_PULLUP);

  pinMode(GreenLED, OUTPUT);

  pinMode(RedLED, OUTPUT);


  //Servo initialization

  Entrance_servo.attach(servoPin1);

  Entrance_servo.write(pos);

  Exit_servo.attach(servoPin2);

  Exit_servo.write(pos2);
```

```
  //LCD initialization
  lcd.init();
  lcd.backlight();
  lcd.begin(16,2);
  lcd.setCursor(0,0);
  lcd.print("Available spots: ");

}

//Entrance Ramp-------------------------------------------------------------------------------
void EntranceRampUp(){
  for (pos = 0; pos <= 90; pos += 5) { // goes from 0 degrees to 90 degrees in steps of 5 degree
    Entrance_servo.write(pos);         // tell servo to go to position in variable 'pos'
    delay(15);                         // waits 15ms for the servo to reach the position
  }

  int x;
  unsigned long starttime = millis();   //time since the function started
  unsigned long endtime = starttime;
```

```
 while((endtime-starttime) <=15000){
  if(digitalRead(ExitButton)==LOW){
  ExitRampUp();
  }
  x = analogRead(A0);
  digitalWrite(GreenLED,HIGH);
  digitalWrite(RedLED,LOW);
  if(x < 100){
     ParkingSpots++;
     if(ParkingSpots>=4){
       ParkingSpots=4;
     }
     delay(100);
     break;
  }
  endtime = millis();              //time since loop started
 }
EntranceRampDown();
}

void EntranceRampDown(){
```

```
  for (pos = 90; pos >= 0; pos -= 5) { // goes from 90 degrees to 0 degrees
    Entrance_servo.write(pos);        // tell servo to go to position in variable 'pos'
    delay(15);                        // waits 15ms for the servo to reach the position
  }
  digitalWrite(GreenLED,LOW);
}



//Exit Ramp-----------------------------------------------------------------------------------
void ExitRampUp(){
  for (pos2 = 180; pos2 >= 90; pos2 -= 5) { // goes from 180 degrees to 90 degrees in steps of 5 degree
    Exit_servo.write(pos2);               // tell servo to go to position in variable 'pos'
    delay(15);                            // waits 15ms for the servo to reach the position
  }


  int x;
  unsigned long starttime = millis();
  unsigned long endtime = starttime;
  while((endtime-starttime) <=15000){           //while loop will run for 15s, after that ramp is down
    if (mfrc522.PICC_IsNewCardPresent()) {
```

```
    RFIDReader();
    }

   x = analogRead(A1);
   digitalWrite(GreenLED,HIGH);
   digitalWrite(RedLED,LOW);
   if(x < 120){

      ParkingSpots--;
      if(ParkingSpots<0){
        ParkingSpots=0;
      }
      delay(100);
      break;
    }
   endtime = millis();
  }
  ExitRampDown();
 }

void ExitRampDown(){
```

```
  for (pos2 = 90; pos2 <= 180; pos2 += 5) {  // goes from 90 degrees to 180 degrees
    Exit_servo.write(pos2);                 // tell servo to go to position in variable 'pos'
    delay(15);                              // waits 15ms for the servo to reach the position
  }
  digitalWrite(GreenLED,LOW);
}



//Closing / Opening park--------------------------------------------------------------------------
void ClosePark(){
  while(0==CurrentSpots){
     Entrance_servo.write(0);
     lcd.setCursor(0,0);
     lcd.print("Park is full");

     if(digitalRead(ExitButton)==LOW){
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Available spots: ");
      ExitRampUp();
      break;
```

```
      }
    }

}

void EmptyPark(){
  while(4==CurrentSpots){
     Exit_servo.write(180);
     lcd.setCursor(0,0);
     lcd.print("Park is empty");

     if (  mfrc522.PICC_IsNewCardPresent()) {
     RFIDReader();
     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("Available spots: ");
     break;
     }

    }
}
```

```
//RFID reader function----------------------------------------------------------------------
void RFIDReader(){
  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial())
  {
    return;
  }

  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }

  content.toUpperCase();
  if (content.substring(1) == "70 9D 0E A3")
  {
```

```
    delay(500);
    EntranceRampUp();

  }
  }

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

void loop()
{
  digitalWrite(RedLED,HIGH);

  CurrentSpots=4-ParkingSpots;
  switch(CurrentSpots){

    case 4:
    lcd.clear();
    EmptyPark();
```

```
break;

case 3:
lcd.setCursor(0,1);
lcd.print(String(CurrentSpots));
break;

case 2:
lcd.setCursor(0,1);
lcd.print(String(CurrentSpots));
break;

case 1:
lcd.setCursor(0,1);
lcd.print(String(CurrentSpots));
break;

case 0:
lcd.clear();
ClosePark();
break;
```

```
    default:


    break;
  }
//-------------------------------------------------------


  if(digitalRead(ExitButton)==LOW){
  ExitRampUp();
}


//-------------------------------------------------------
  // Look for new cards
if (  mfrc522.PICC_IsNewCardPresent()) {
     RFIDReader();
     }
}
```