Sveučilište u Rijeci, Tehnički fakultet

“RFID čitač kartica”

Završni projekt - Ugradbeni računalni sustavi

Dokumentacija projekta

Računarstvo

Tim:

Deni Vrban

Denis Južnić

Sven Celin

Mentor:

doc.dr.sc. Mladen Tomić

Asistent:

mag.ing.comp. Diego Sušanj

Sadržaj

1. Uvod

1.1. Pojmovnik i kratice

Kratica	Opis
RFID	Radio Frequency Identification

EEPROM	Electrically Erasable Programmable Read Only Memory
LED	Light Emitting Diode
LCD	Liquid Crystal Display
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory

1.2. Reference i materijali

- Atmel - ATMega16A datasheet
- MFRC522 Contactless reader IC datasheet -
  https://www.hobbytronics.co.uk/datasheets/sensors/MFRC522.pdf
  https://www.hobbytronics.co.uk/mfrc522-reader
  https://www.avrfreaks.net/forum/rfid-rc522-spi-and-atmega32a-initialisation
  https://randomnerdtutorials.com/security-access-using-mfrc522-rfid-reader-with-arduino/
  https://github.com/miguelbalboa/rfid
  https://github.com/asif-mahmud/MIFARE-RFID-with-AVR/tree/master/lib
- SPI - http://maxembedded.com/2013/11/the-spi-of-the-avr/
  http://avrbeginners.net/architecture/spi/spi.html
  https://www.electronicwings.com/avr-atmega/atmega1632-spi
  http://www.firmcodes.com/microcontrollers/avr/spi-interfacing-with-atmega16/

2. Hardware

2.1. Popis korištenih komponenti

Oprema	Funkcija
ATMega16a	Jezgra projekta, na njoj se izvršava software

LCD	Prikazuje izbornike i trenutna stanja
MFRC522	Učitava RFID kartice u sustav
Žice - razne	Povezuje hardware, služi za slanje i primanje signala
RFID kartice	Potiču promjene u sustavu
ISP	Programator kojim stavljamo kod na ATMegu16a
Breadboard	Koristi se za napajanje LCD-a, RFID readera itd.

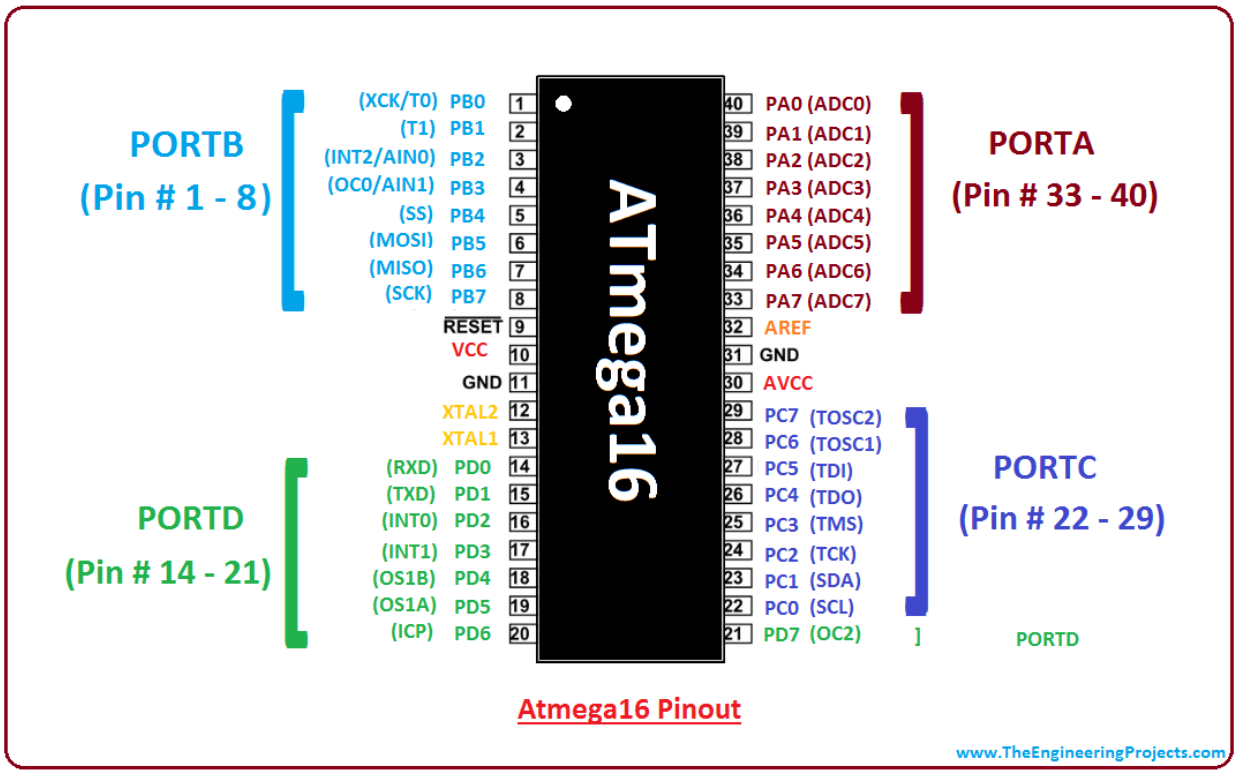
2.2. Opis komponenti

2.2.1. Atmega16a

Glavna uloga mikrokontrolera je povezivanje eksternih senzora tj. hardware-a sa napisanim kodom tj. software-om. Pomoću gumbi nazvanih KEY1, KEY2, KEY3 i KEY4, možemo aktivirati mogućnosti koje nudi ovaj sustav. Osim davanja mogućnosti pritiskom na gumbe pločica ima sekundarne funkcije. Napaja i postavlja potrebne pinove za korištenje eksternih senzora. Tako su na ATMegu spojeni LCD uz pomoć 10 pinova koji su povezani žičicama na PORT-u D te MFRC522 koji je povezan sa 6 žičica na PORT-u B. Sav kod se izvršava na pločici koja onda šalje signale ka senzorima odnosno ekranu nakon kojeg se ispisuju traženi rezultati na istome. Kao što smo prije naveli pritiskom na gumbe mijenjamo operacije same pločice koje mogu biti SCAN, ADD i DELETE uz koje smo dodali četvrtu opciju čišćenja EEPROM memorije radi resetiranja samog sustava.

- Značajke ATMege16a
    - 32 8 bitna registra opće namjene
    - 64 I/O registra
    - 1024B SRAM-a
    - EEPROM memorija od 512B

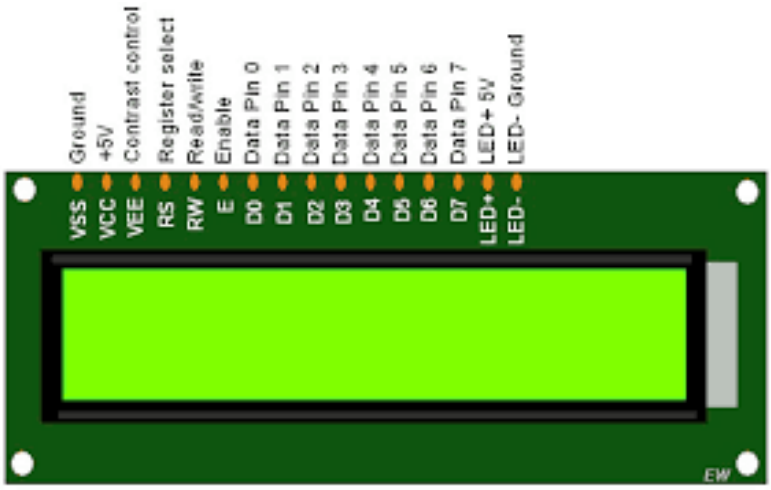
- Dva 8 bitna timera i jedan 16 bitni timer
- Radni napon 2.7V do 5.5V



2.2.2. LCM1602C LCD

LCD kao što samo ime govori, napravljen je na tehnologiji tekućih kristala. Ovaj tip ekrana koristi matricu od 16x2 znaka koja može ispisati odjednom. Također postoje pinovi koji nam mogu poslužiti za pozadinsko osvjetljenje ekrana ukoliko je mrak. Ovaj uređaj će nam omogućiti ispis opcija koje korisnik ima za korištenje.

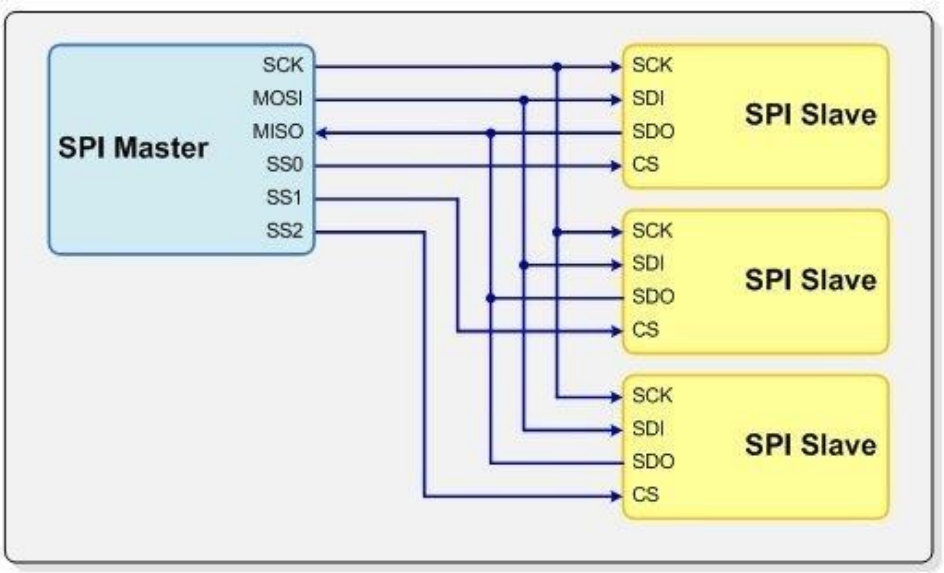
- Značajke LCD-a:
  - 80 pinski kontroler
  - 14 pinsko sučelje
  - 8 i 4 bitni način rada



2.2.3. RC522 RFID

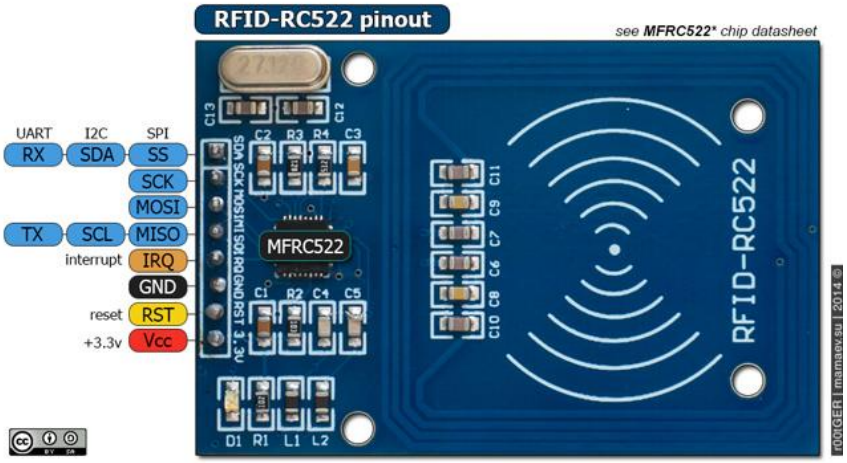
Uloga ovog senzora je čitanje podataka sa RFID kartice na određenim frekvencijama. Nakon očitavnja kartice, prosljeđuje taj signal na mikrokontroler e onda on obrađuje te podatke sukladno kodu.

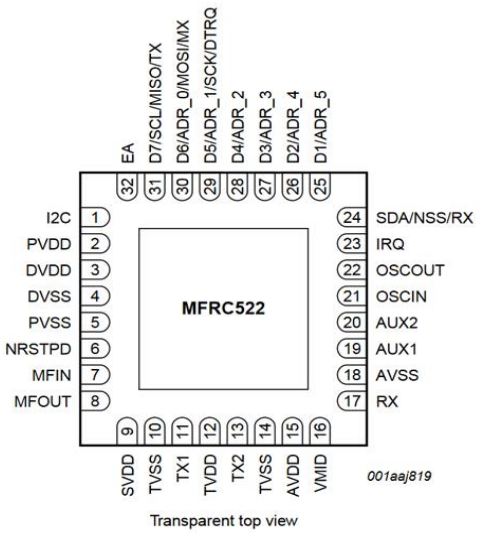
2.2.3.1. Serial Peripheral Interface



SPI je sinkrona serijalna komunikacija koja se najviše koristi za komunikaciju kratkog dometa, primarno za sigurnosne kartice. Glavna značajka SPI komunikacije je ta što uređaji komuniciraju u full duplex načinu rada, u master-slave arhitekturi, što znači da oba kraja mogu slati i primati poruke istovremeno (karakteristična primjena full duplexa je telefonska komunikacija). U master-slave arhitekturi uvijek postoji samo jedan master uređaj, koji definira okvire (frameove) u kojima se šalju podaci, dok slaveova može biti i više.

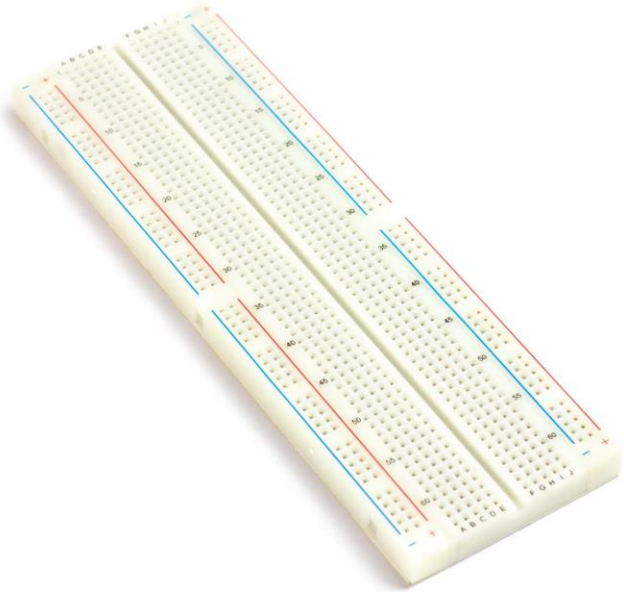
Da bi komunikacija započela, Master mora postaviti frekvenciju brojila na neku koju podržava slave, najčešće do nekoliko MHz. Zatim master odabere slave uređaj koji ima logičku nulu na Select liniji. Ako je potrebna analogno-digitalna pretvorba, master mora čekati minimalno toliko dugo koliko je potrebno za promjenu prije nego pokrene brojač. Tijekom svakog ciklusa brojača provodi se full duplex komunikacija. Master šalje bit na MOSI (Master Output Slave Input) pin kojega slave pročita te šalje odgovor na MISO (Master Input Slave Output) pin kojega master pročita. Ta sekvenca se održava čak i prilikom jednosmjerne komunikacije.

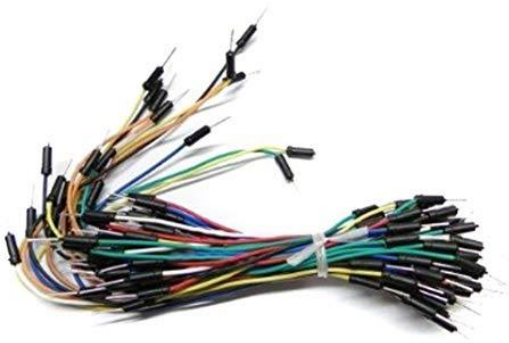




2.2.4. Ostalo

Preostali dijelovi koji su korišteni u ovom projektu su RFID kartice, koje koristimo za demonstriranje rada projekta, breadboard na koji smo postavili sve senzore (LCD i MFRC522), te žičice koje su muško-muške, muško-ženske te žensko-ženske.





2.3. Zahtjevi sustava

Osnovni uvjeti rada sustava:

- Normalna radna temperatura - 25℃
- Napon napajanja - 5V
- Navedeni dijelovi u prethodnom tekstu

Sigurnosni zahtjevi rada sustava:

- Ne izlagati vodenim površinama

- Ne izlagati visokom naponu
- Ne izlagati visokim temperaturama
- Ne izlagati jakim pritisnim silama
- Potrebno odvojiti dijelove jedno od drugih
- Izbjegavati oštećenja opreme

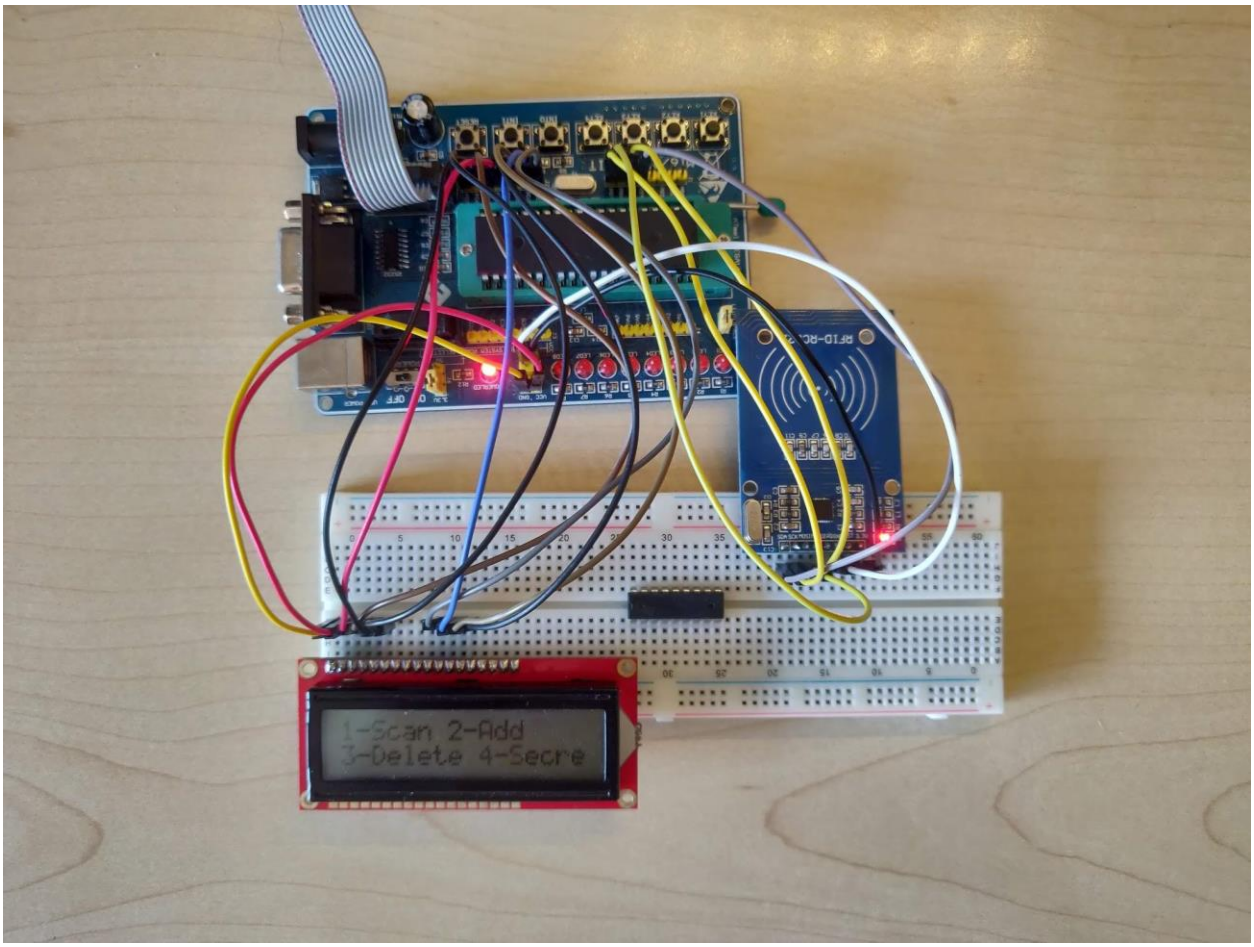
2.4. Ograničenja sustava

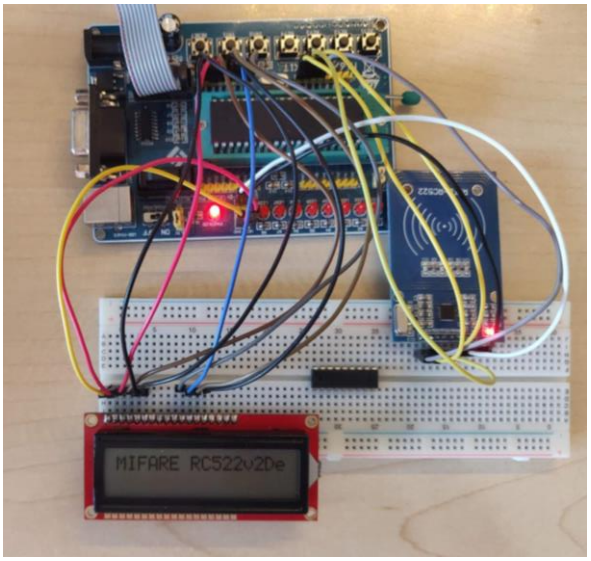
Naravno, kao i uvijek, i naš sustav ima određena ograničenja. Neki od portova nisu dostupni za eventualnae daljnje nadogradnje s obzirom na to da imaju određenu ulogu u funkcionalnosti ovog sustava. Naime, PORT D koristimo za upravljanje ekranom, tj. za upravljanje kontrastom LCD ekrana (pomoću timera Atmega mikrokontrolera), odabir registara (Register select, RS), odabir funkcije čitanja sa ekrana ili pisanja na LCD ekran (Read/Write Select, RW) te upravljanje pina za uključivanje ekrana. Ostatak PORT D (preostala četiri pina) koristi se za slanje podataka između ekrana i mikrokontrolera.

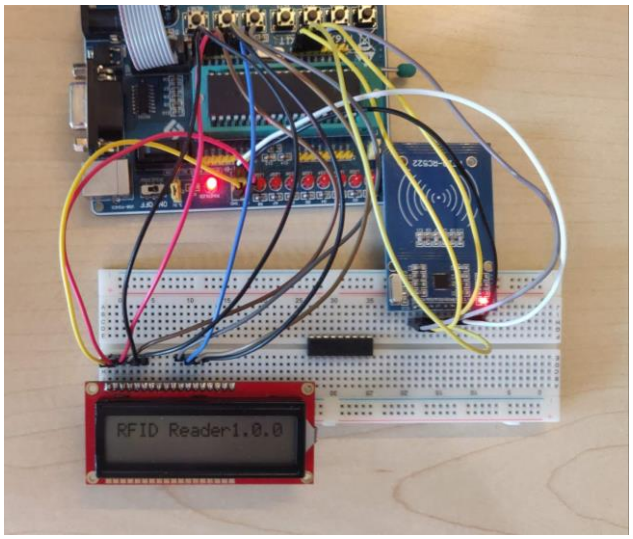
Drugo ograničenje javlja se kod korištenja PORT-a B, s obzirom na to da donji dio PORT B služi za upravljanje mikrokontrolerom putem gumba na pločici, a gornji dio nam služi za povezivanje RFID senzora s Atmega mikrokontrolerom. Stoga možemo zaključiti da nam je PORT B ključan za funkcionalnost ovog sustava - bez korištenja gumba ne možemo odabrati željenu radnju.

Još jedno ograničenje, možda i najznačajnije jest to da u EEPROM možemo spremiti samo tri kartice. To se, naravno, može i povećati, no za naše trenutne potrebe dovoljan nam je prostor za tri kartice.

2.5. Ograničenja sustava

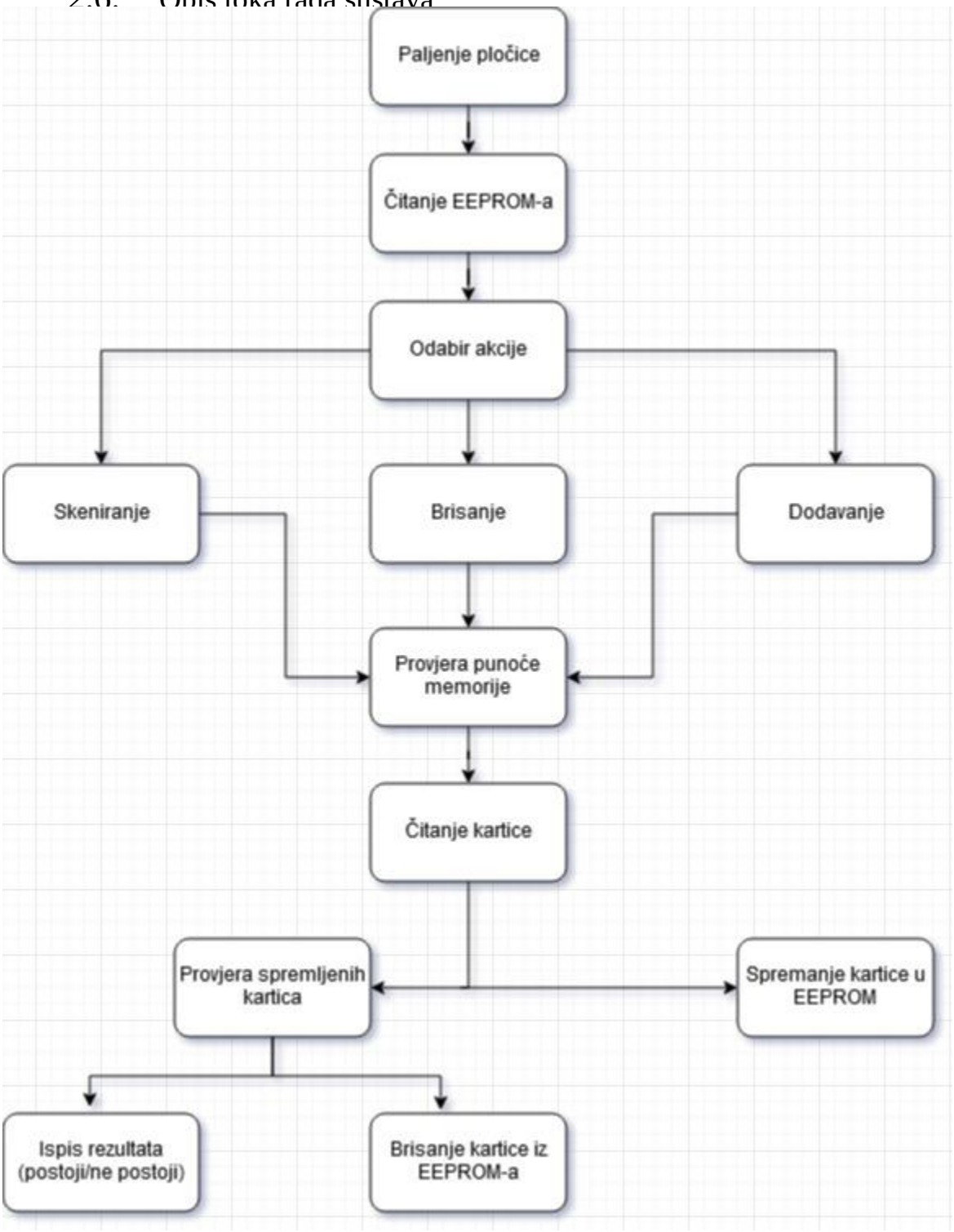






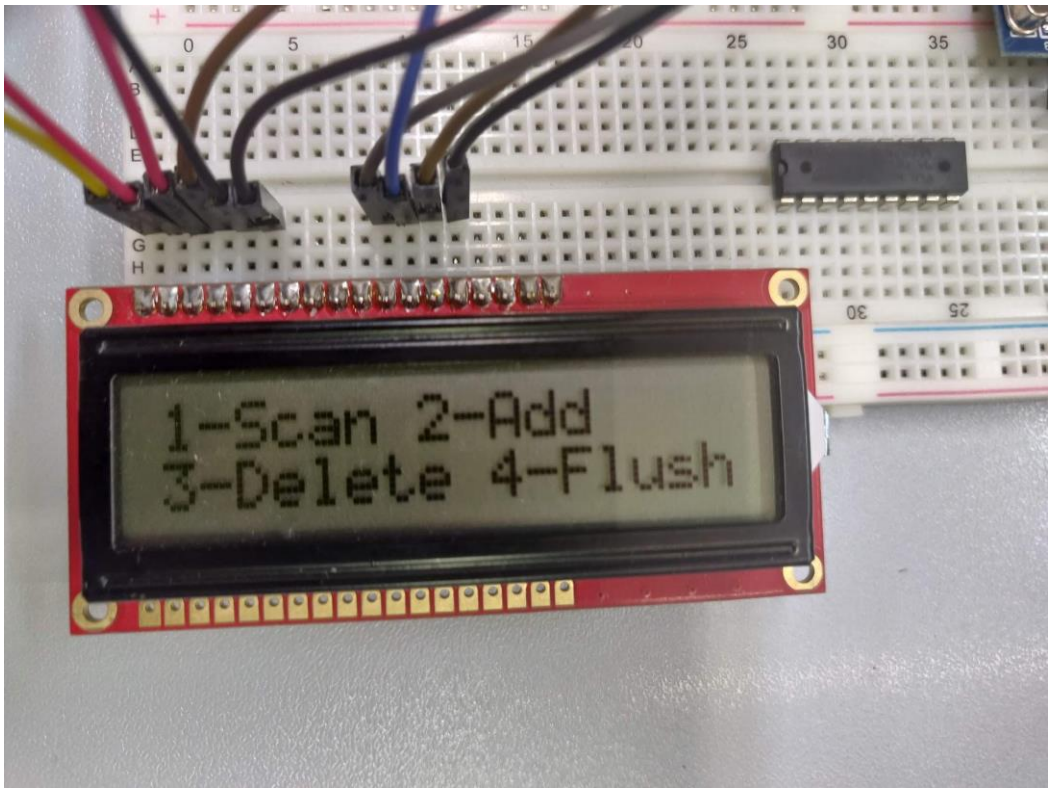
Prilikom pokretanja prikazuje se verzija čitača koja se koristi.

2.6. Opis toka rada sustava



S dijagrama možemo vidjeti osnovni tijek rada sustava. Nakon paljenja pločice, pokreće se program, a prva radnja koja se odvija je čitanje memorije EEPROM-a. U EEPROM spremamo sve dodane kartice, kako bi i nakon gašenja pločice imali spremljene sve kartice u memoriji.

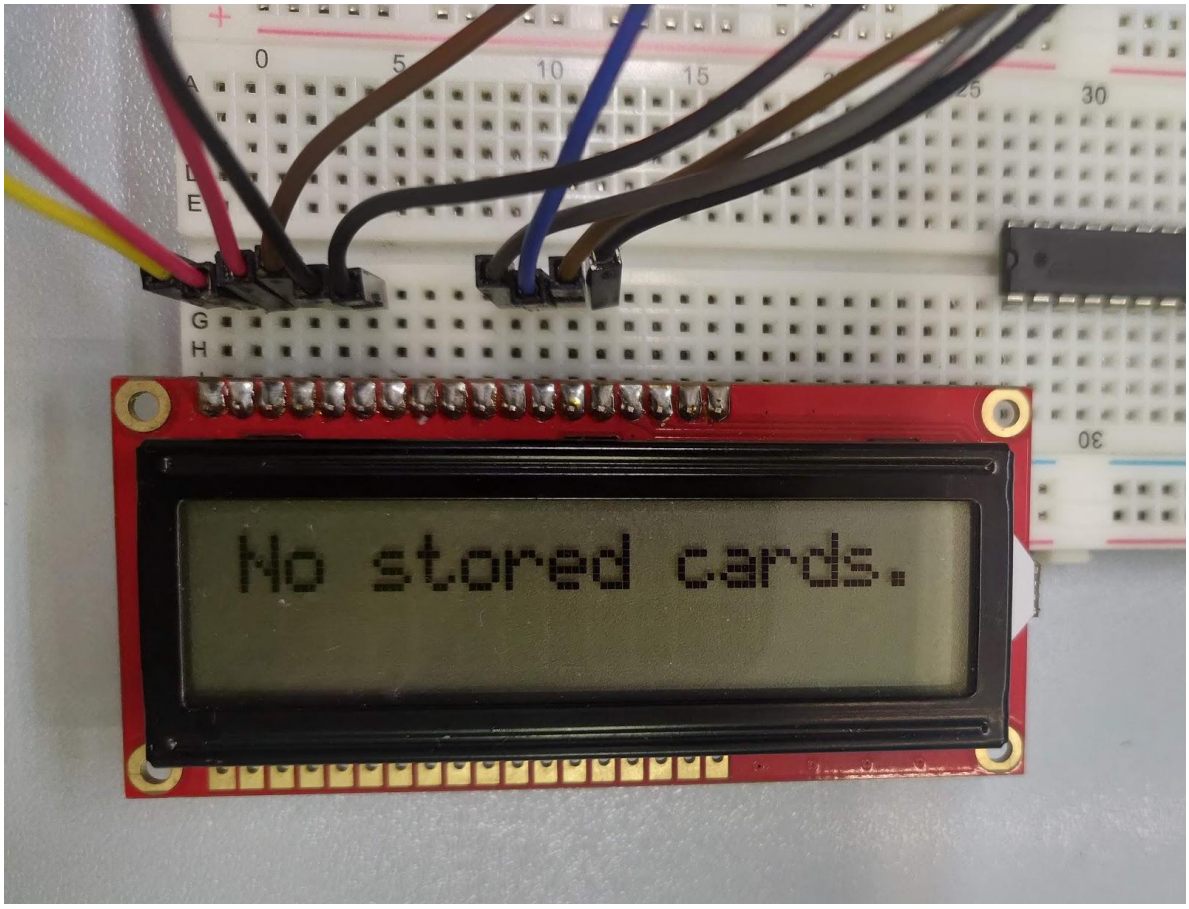
Nakon čitanja memorije EEPROM-a, čeka se odabir jedne od 3 mogućih akcija - skeniranje, brisanje ili dodavanje kartice, popraćene porukom (slika) s označenim akcijama i pripadajućim brojem koji odgovara gumbu koji je potrebno pritisnuti. Točnije, gumb broj 1 služi za skeniranje kartice i provjeru sa već postojećima u memoriji, gumb broj 2 za dodavanje kartice u memoriju, dok gumb broj 3 služi za brisanje nekih od već postojećih kartica.



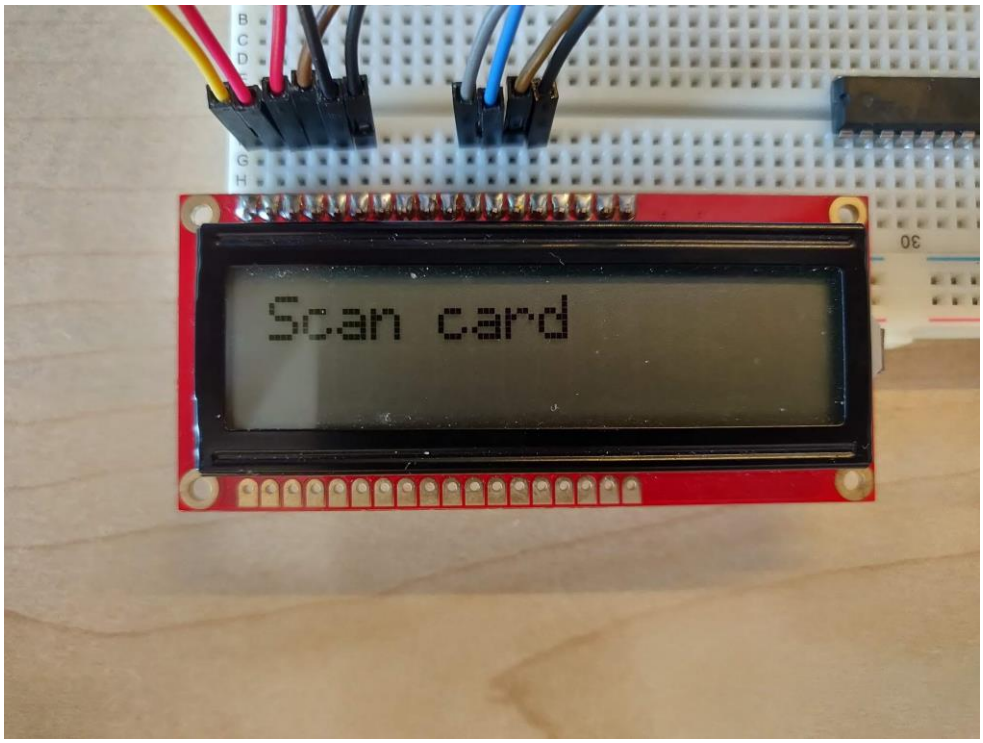
Nakon odabira akcije, moguća su daljnja tri slijeda kojim će se program izvršavati:

1. Skeniranje

Za skeniranje potrebno je, kao i kod druge dve akcije, raditi provjeru punoće memorije. Razlog tomu je kako bi preskočili cijeli postupak skeniranja ukoliko je memorija prazna. Naime, ako nema spremljenih kartica u memoriji - skeniranje nije potrebno raditi, jer nemamo s čime uspoređivati jer spremljenih kartica nema. Ako je došlo do takve situacije, na ekranu se ispisuje poruka “No stored cards.”, te se vraćamo u početno stanje čekanja odabira akcije.



Ukoliko smo prošli provjeru memorije, na ekranu se ispisuje “Scan card” te čekamo korisnika da prisloni karticu na RFID čitač kako bi očitali broj kartice.

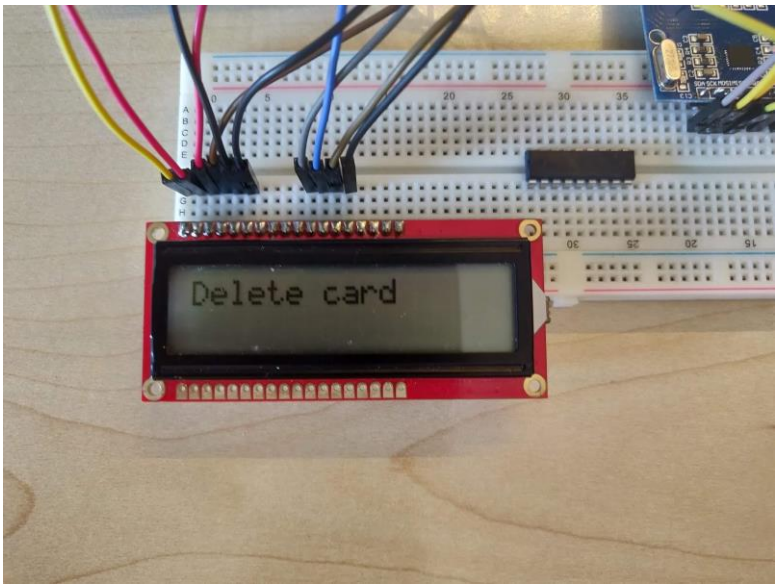


Nakon očitanog broja kartice, idemo u provjeru s već spremljenim karticama. Drugim riječima, dobiveni broj kartice potrebno je usporediti s prethodno spremljenim karticama u memoriju. Nakon usporedbe s 3 moguće kartice, na ekranu će se ispisati poruka “Card OK.” ukoliko smo utvrdili da je kartica već spremljena u memoriji, dok se u suprotnom ispisuje “Unknown card” ukoliko niti jedna od spremljenih kartica ne odgovara onoj očitanoj.

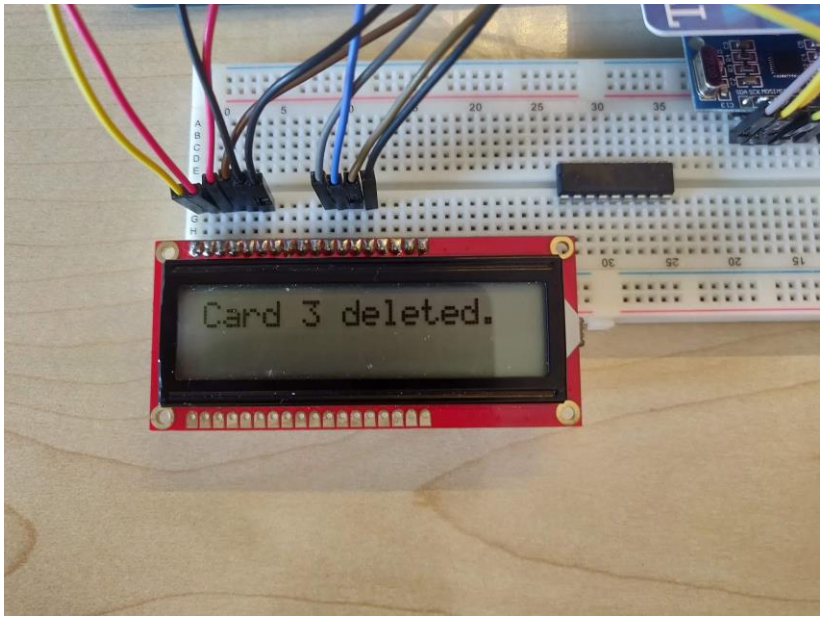
2. Brisanje

Brisanje je akcija kojom očitanu karticu, ukoliko je već spremljena u memoriji, brišemo iz memorije. Nakon odabira akcije, ulazimo u provjeru punoću memorije kako bi provjerili je li memorija prazna. Ukoliko je memorija prazna, nemamo što za brisati iz memorije i tako preskačemo cijele daljnje operacije skeniranja i pokušaja brisanja kartice, te na ekranu ispisujemo poruku “No card saved.”. Vraćamo se u početno stanje čekanja odabira akcije.

Ukoliko smo prošli provjeru punoće memorije, na ekranu se ispisuje “Delete card”, te čekamo korisnika da očita karticu.



Nakon što je korisnik očitao karticu, kao i kod operacije skeniranja, radimo usporedbu očitanog broja kartice sa onim već postojećim u memoriji. Ukoliko je kartica pronađena, radimo brisanje iz memorije tako da u EEPROM zapisujemo vrijednost kartice u obliku svih nula. Na ekranu se ispisuje poruka “Card n deleted”, dok n označuje broj kartice (1, 2 ili 3).

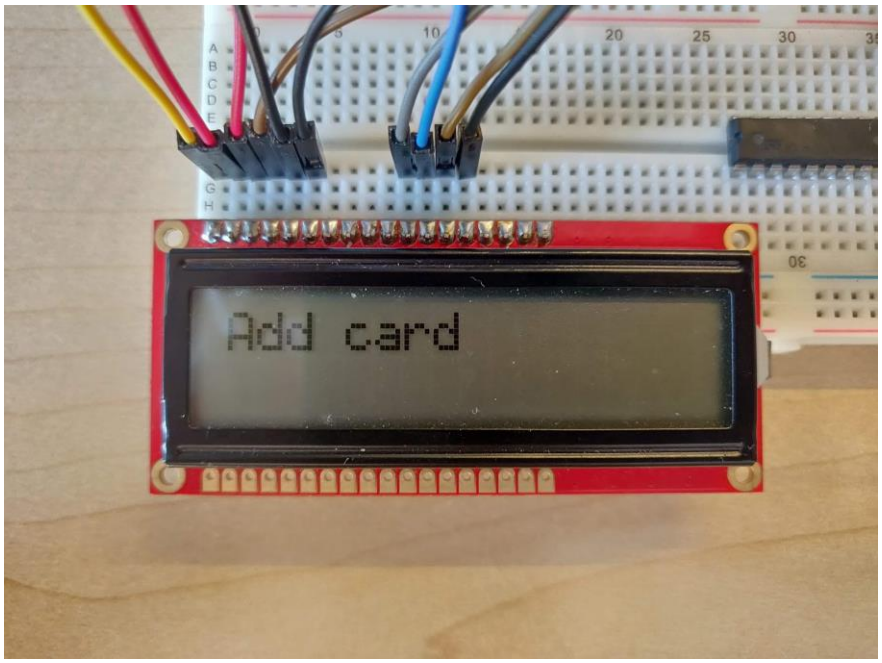


S druge strane, ukoliko kartica nije pronađena usporedbom, na ekranu se ispisuje poruka “Unknown card”. Nakon bilo koja od ova dva koraka, vraćamo se u početno stanje glavnog izbornika i čekanja na odabir akcije.

3. Dodavanje

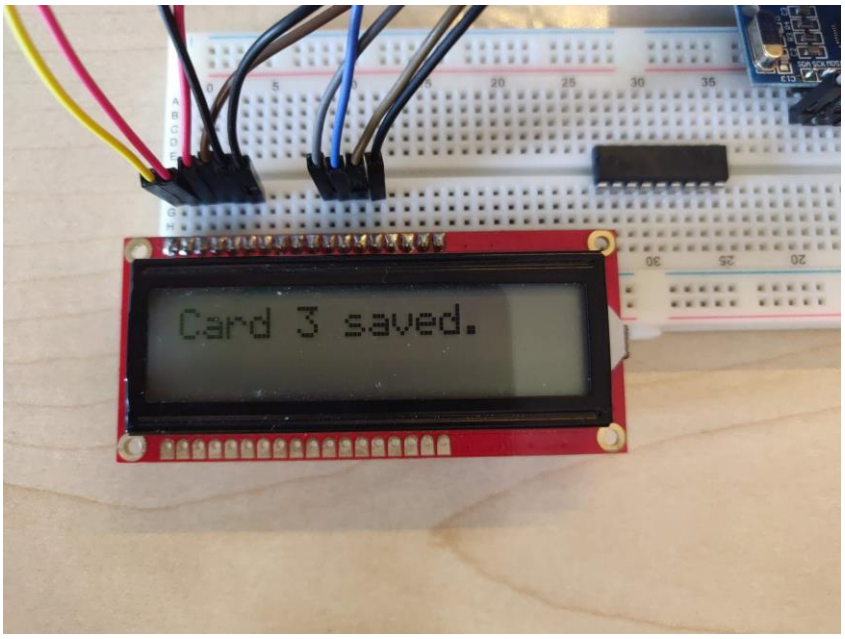
Dodavanje je akcija kojom spremamo očitanu karticu u EEPROM memoriju. Kao i kod druge dvije akcije, radimo provjeru punoće memorije kako bi utvrdili hoćemo li uopće preći u stanje očitavanja kartice. No, u ovom slučaju ne provjeravamo je li memorija prazna, već je li ona puna - ako je memorija puna, nećemo moći spremiti karticu, te tako preskačemo čitanje kartice i vraćamo se na glavni izbornik.

Ukoliko smo prošli provjeru memorije, na ekranu se ispisuje poruka “Add card”, te čekamo korisnika da očita karticu kako bi mogli nastaviti sa izvršavanjem programa.



Nakon što je korisnik očitao karticu, broj kartice prolazi provjeru slobodnih mjesta. Odnosno, provjerava se u koje mjesto je moguće spremiti karticu - počinjemo od prvog mjesta, ukoliko je ono prazno, spremamo u njega, ako ukoliko nije, idemo na drugo, te potom na treće mjesto.

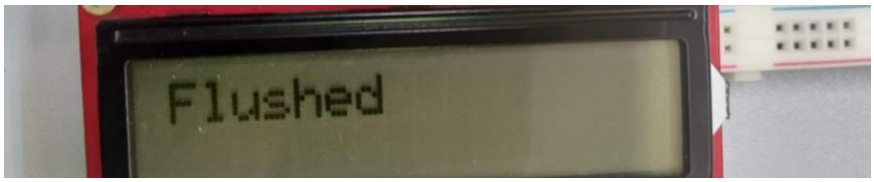
Kartica će biti spremljena u jedno od tri mjesta - u ovom slučaju nikad nećemo doći u situaciju da je memorija puna, s obzirom da smo pri ulasku u akciju već napravili provjeru punoće memorije, kako bi izbjegli bilo kakve poteškoće u daljnjem radu.



Nakon otkrivenog praznog mjesta, kartica se sprema u određeni EEPROM blok, a na ekranu se ispisuje “Card n saved.”, gdje n označava broj mjesta kartice. Vraćamo se na glavni izbornik.

4. Čišćenje memorije (Flush)

Flush je akcija kojom u potpunosti praznimo memoriju od prethodno spremljenih kartica. Ovo je dodano kao opcionalna funkcija kojom osiguravamo da ukoliko više nemamo pristup nekoj od spremljenih kartica, i dalje ju možemo ukloniti iz memorije. Pritiskom na gumb 4 na lokacije prve, druge i treće kartice se zapisuje “empty” string, koji sadrži sve nule. Nakon toga na ekranu se ispisuje “Flushed” te se vraćamo na početni izbornik. Osim spremanja praznih vrijednosti u EEPROM, varijable trenutno pokrenutog programa ponovno se čitaju iz EEPROM-a, kao što se to činilo i odmah nakon paljenja pločice.



3. Software - opis programskog koda

Na početku main.c datoteke, koja je ujedno i glavna datoteka ovog programa, radimo *import* svih knjižnica potrebnih za rad ovog sustava.

```
#define F_CPU 7372800UL
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>
#include <avr/eeprom.h>
#include "lcd.h"
#include "spi.h"
#include "mfrc522.h"
#include <string.h>
```

Nakon toga, inicijaliziramo potrebne varijable, a to su: card1, card2 i card3 - polja znakova u koja ćemo spremati pročitane kartice, varijablu empty koja će biti fiksna od početka do kraja, a u nju će biti spremljene sve nule kako bi to označilo "prazni" ID kartice, card1_empty, card2_empty i card3_empty - integeri koji će služiti kao oznaka jesu li pojedine kartice prazne, te scan, flush, add i delete - također integeri koji će služiti koja je od akcija odabrana. U početku ove 4 varijable postavljamo na nulu, te se one mijenjaju tek kada se neka od radnji odabere.

```
char card1[11];
char card2[11];
char card3[11];
char empty[] = "0000000000";

int card1_empty = 0;
int card2_empty = 0;
int card3_empty = 0;

int scan = 0;
int delete = 0;
int add = 0;
int flush = 0;
```

U main funkciji postavljamo dodatne varijable (str, byte, rez i temp) koje će služiti kao privremene varijable za očitavanje i spremanje broja kartice u memoriju EEPROM-a. Potom

pozivamo funkciju init() koja postavlja potrebne portove u stanja čitanja ili pisanja, inicijalizira brojilo za kontrast LCD-a, te inicijalizira LCD.

```
int main() {
    uint8_t byte;
    uint8_t str[16];
    char rez[11];
    _delay_ms(50);
    char temp[11];
    int flag=0;

    init();
    eeprom_read();

    card1_empty = card_empty_check(1);
    card2_empty = card_empty_check(2);
    card3_empty = card_empty_check(3);

    lcd_puts("RFID Reader");

    spi_init();
    _delay_ms(1000);
    lcd_clrscr();
    mfrc522_init();
```

```
void init() {
    DDRD |= _BV(4);

    DDRB = 0x00;
    PORTB = 0x0f;

    TCCR1A = _BV(COM1B1) | _BV(WGM10);
    TCCR1B = _BV(WGM12) | _BV(CS11);
    OCR1B = 96;
    lcd_init(LCD_DISP_ON);
    lcd_clrscr();
}
```

Nakon pozivanja funkcije init(), pozivamo funkciju eeprom_read() koja će iz EEPROM-a pročitati već zapisane vrijednosti. Ovo je ključno kako bi zadržali spremljene kartice i nakon gubitka napajanja na mikrokontroleru.

```
void eeprom_read() {
    eeprom_read_block((void*)&card1, (const void*)0, 10);
    eeprom_read_block((void*)&card2, (const void*)10, 10);
    eeprom_read_block((void*)&card3, (const void*)20, 10);
}
```

Uz to, varijable card_empty označavamo sa 0 ili 1 pozivom funkcije card_empty_check() (koja kao argument prima 1, 2, ili 3 kao oznaku broja kartice) koja vraća 0 ukoliko je u memoriji predviđenoj za neku određenu karticu bilo što drugo osim "praznog" stringa (sve nule). Također, pozivamo spi_init() i mfrc522_init() koje služe za inicijalizaciju SPI veze između čitača i mikrokontrolera, te inicijalizaciju samog čitača.

```
while(1){
    menu_reset();
    if(bit_is_clear(PINB, 0)) {
        scan = 1;
        add = 0;
        delete = 0;
        lcd_clrscr();
    }

    if(bit_is_clear(PINB, 1)) {
        scan = 0;
        add = 1;
        delete = 0;
        lcd_clrscr();
    }

    if(bit_is_clear(PINB, 2)) {
        scan = 0;
        add = 0;
        delete = 1;
        lcd_clrscr();
    }

    if(bit_is_clear(PINB, 3)) {
        scan = 0;
        add = 0;
        delete = 0;
        flush = 1;
        lcd_clrscr();
    }
```

```
void menu_reset() {
    lcd_clrscr();
    lcd_puts("1-Scan 2-Add");
    lcd_gotoxy(0,1);
    lcd_puts("3-Delete 4-Flush");

    scan = 0;
    add = 0;
    delete = 0;
    flush = 0;
}
```

Nakon toga, počinje beskonačna petlja u kojoj se izvršava glavni dio programa. Na početku svake petlje pozivamo funkciju menu_reset() koja na ekranu ispisuje glavni izbornik i postavlja sve oznake za radnje na nulu.

```
if(scan) {
    if(full_card_empty_check()) {
        lcd_puts("No stored cards.");
        _delay_ms(1000);
    } else {
        lcd_puts("Scan card");
        flag = 1;
        while(flag) {
        byte = mfrc522_request(PICC_REQALL,str);
        if(byte == CARD_FOUND){
            byte = mfrc522_get_card_serial(str);
            if(byte == CARD_FOUND){
                temp[0] = '\0';
                for(byte=0;byte<5;byte++) {
                    itoa(str[byte],rez,16);
                    strcat(temp,rez);
                }
                flag = 0;
                lcd_clrscr();
            }
            else{
                lcd_puts("Error");
            }
        }
    }
        if(!strcmp(card1,temp)) lcd_puts("Card 1 OK.");
        else if(!strcmp(card2,temp)) lcd_puts("Card 2 OK.");
        else if(!strcmp(card3,temp)) lcd_puts("Card 3 OK.");
        else lcd_puts("Unknown card.");
    }
}
```

Prvu radnju koju provjeravamo je li “upaljena” je radnja skeniranja kartice odnosno provjera ako je kartica već spremljena u memoriju. Na početku pozivamo funkciju full_card_empty_check() koja provjerava jesu li sve kartice “prazne” u memoriji, jer onda nemamo potrebu ići dalje u ostatak akcije skeniranja ukoliko jesu. Ako postoji neka kartica u memoriji, prelazimo u while petlju koja čeka da korisnik očita karticu tako da prvo šalje request čitaču, dobiva odgovor, te nakon toga čeka stream podataka. Kada je kartica očitana, pretvaramo dobivenu vrijednost u heksadekadsku preko for petlje tako da pretvaramo byte po byte, a cijela vrijednost zapisuje se u temp varijablu. Potom, radimo usporedbu sa sve tri varijable kartice putem strcmp() funkcije koja provjerava podudarnost 2 stringa. Ukoliko je jedna od kartica pronađena, ispisuje se na ekranu poruka “Card n OK.” (gdje je n broj kartice), a ukoliko niti jedna kartica ne odgovara očitanoj vrijednosti, ispisuje se poruka “Unknown card.”

```
int full_card_empty_check() {
    for(int i = 1; i < 4; i++) {
        if(!card_empty_check(i)) return 0;
    }

    return 1;
}
```

```
  if(delete) {
      if(full_card_empty_check()) {
          lcd_puts("No stored cards.");
          _delay_ms(1000);
      }else {
          lcd_puts("Delete card");

          flag = 1;
          while(flag) {
              byte = mfrc522_request(PICC_REQALL,str);
              if(byte == CARD_FOUND){
                  byte = mfrc522_get_card_serial(str);
                  if(byte == CARD_FOUND){
                      temp[0] = '\0';
                      for(byte=0;byte<5;byte++) {
                          itoa(str[byte],rez,16);
                          strcat(temp,rez);
                      }
                      flag = 0;
                      lcd_clrscr();
                  }
                  else{
                      lcd_puts("Error");
                  }
              }
          }

          if(!strcmp(card1,temp)) {
              eeprom_flush(1);
              strcpy(card1,empty);
              lcd_puts("Card 1 deleted.");
          } else if(!strcmp(card2,temp)) {
              eeprom_flush(2);
              strcpy(card2,empty);
              lcd_puts("Card 2 deleted.");
          }else if(!strcmp(card3,temp)){
              eeprom_flush(3);
              strcpy(card3,empty);
              lcd_puts("Card 3 deleted.");
          } else lcd_puts("Unknown card.");
      }
  }
```

Sljedeća akcija koju provjeravamo je akcija brisanja odnosno delete. Kao i u prethodnoj akciji, provjeravamo jesu li sve kartice "prazne", te ukoliko jesu, preskačemo ostatak programskog dijela ove funkcije. Izvodimo isti postupak za čitanje kartice kao i u prethodnoj akciji, te opet uspoređujemo pročitanu karticu sa svakom koja je već spremljena u memoriji. Ukoliko se jedna od vrijednosti podudara, pozivamo funkciju eeprom_flush() koja kao argument prima broj kartice. Ta funkcija će na predviđeno mjesto u EEPROM-u zapisati sve nule, kako bi označilo prazno mjesto u memoriji. No, ako nema podudarne vrijednosti, na ekran ispisujemo poruku "Unknown card.".

```
void eeprom_flush(int n) {
    if(n == 1 || n == 4) eeprom_write_block((void*)&empty, (void*)0, 10);
    if(n == 2 || n == 4) eeprom_write_block((void*)&empty, (void*)10, 10);
    if(n == 3 || n == 4) eeprom_write_block((void*)&empty, (void*)20, 10);
}
```

```
if(add) {
    lcd_puts("Add card");
    flag = 1;
    while(flag) {
        byte = mfrc522_request(PICC_REQALL,str);
        if(byte == CARD_FOUND){
            byte = mfrc522_get_card_serial(str);
            if(byte == CARD_FOUND){
                temp[0] = '\0';
                for(byte=0;byte<5;byte++) {
                    itoa(str[byte],rez,16);
                    strcat(temp,rez);
                }
                flag = 0;
                lcd_clrscr();
            }
            else{
                lcd_clrscr();
                lcd_puts("Error");
                _delay_ms(1000);
                lcd_clrscr();
            }
        }
    }
```

```
    if(!strcmp(card1,temp)) lcd_puts("Card 1 exists.");
    else if(!strcmp(card2,temp)) lcd_puts("Card 2 exists.");
    else if(!strcmp(card3,temp)) lcd_puts("Card 3 exists.");
    else if(card_empty_check(1)) {
        strcpy(card1,temp);
        write_card(1);
        lcd_puts("Card 1 saved.");
    }
    else if(card_empty_check(2)) {
        strcpy(card2,temp);
        write_card(2);
        lcd_puts("Card 2 saved.");
    }
    else if(card_empty_check(3)){
        strcpy(card3,temp);
        write_card(3);
        lcd_puts("Card 3 saved.");
    }else lcd_puts("No space.");

    _delay_ms(1000);
}
```

Treća funkcija izvodi dodavanje kartice u memoriju. Kao i ostale dvije funkcije, bazira se na istom principu čitanja kartice sa RFID čitača. Kada smo očitali dobivenu vrijednost, potrebno je usporediti dobivenu vrijednost sa vrijednostima u memoriji jer ne želimo imati više unosa u memoriji sa istom karticom. Ako ne postoji ista kartica, provjerava se punoća memorije preko funkcije card_empty_check() koja kao argument prima broj kartice, a vraća 0 ili 1 s obzirom na

to je li ta varijabla “prazna” ili ne. Ako postoji prazno mjesto, zapisujemo ju u EEPROM pomoću funkcije write_card() koja kao argument prima broj kartice. Ukoliko niti jedno mjesto nije slobodno, na ekranu ispisujemo “No space.”

```
void write_card(int n){
    if(n == 1) eeprom_write_block((void*)&card1, (void*)0, 10);
    if(n == 2) eeprom_write_block((void*)&card2, (void*)10, 10);
    if(n == 3) eeprom_write_block((void*)&card3, (void*)20, 10);
}


int card_empty_check(int n) {
    if(n == 1) {
        if(!strcmp(card1,empty)) return 1;
    }
    if(n == 2) {
        if(!strcmp(card2,empty)) return 1;
    }
    if(n == 3) {
        if(!strcmp(card3,empty)) return 1;
    }

    return 0;
}
```

Posljednja radnja koju je moguće odabrati je radnja flush, koja čisti sve unose u EEPROM tako da na njih zapiše “praznu” karticu, odnosno varijablu koja sadrži sve nule.

```
if(flush) {
    eeprom_flush(4);
    strcpy(card1,empty);
    strcpy(card2,empty);
    strcpy(card3,empty);
    lcd_puts("Flushed");
    flush = 0;
    _delay_ms(1000);

}
```

4. Zaključak

Ovaj projekt bio je zaista zahtjevan, s obzirom na to da smo počeli raditi sa UART komunikacijom, a zbog neispravne tehničke opreme bili smo prisiljeni prebaciti se na novi RFID čitač, RC522, koji komunikaciju implementira putem SPI komunikacijskog protokola. Pošto nitko od nas iz grupe nije radio na SPI protokolu, morali smo ispočetka učiti sve o tome.

Ako ignoriramo tehničke probleme, ovaj projekt bio je zanimljiv, te je krajnji rezultat itekako iskoristiv u svakodnevici. Naime, kontrola ulaza pomoću RFID kartica široko je primjenjiva, npr. u bolnicama, u raznim tvrtkama kao kontrola vremena ulaska i izlaska itd. Naš projekt može se i proširiti dodavanjem motora koji otvara i zatvara vrata, ili žarulje koja se pali i gasi kao indikator zauzetosti prostorije čiju dostupnost pokušavamo ograničiti.