SUSTAV BIOMETRIJSKE REGISTRACIJE PRISUTNOSTI

Kundid, Marin

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij, elektronika

MARIN KUNDID

ZAVRŠNI RAD

SUSTAV BIOMETRIJSKE REGISTRACIJE PRISUTNOSTI

Split, rujan 2021.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij, elektronika

Predmet: Automatski regulacijski sustavi

Z A V R Š N I R A D

Kandidat: Marin Kundid

Naslov rada: Sustav biometrijske registracije prisutnosti

Mentor: Dr. sc. Barbara Džaja

SADRŽAJ

Sažetak

Sustav biometrijske registracije prisutnosti

U završnom radu izrađen je sustav biometrijske registracije prisutnosti temeljen na senzoru otiska prsta koji je upravljiv pomoću Arduina. Koristi se modul senzora otiska prsta za autentifikaciju prave osobe (studenta, zaposlenika), te je otisak prsta pohranjen u sustav. Koriste se 4 tipke za registraciju novog korisnika, brisanje korisnika, te usklađivanje spremljenog otiska prsta. Vrijeme skeniranja postojećeg korisnika se vrši preko RTC modula DS3231. Također evidenciju prisutnosti je moguće izvesti pomoću serijskog monitora arduina.

Ključne riječi: senzor otiska prsta, RTC modul, skeniranje, evidencija.

Biometric presence registration system

In the final work, a biometric presence registration system based on an Arduino-controlled fingerprint sensor was developed. The fingerprint sensor module is used to authenticate the right person (student, employee), and the fingerprint is stored in the system. 4 buttons are used to register a new user, delete a user, and synchronize a saved fingerprint. The scanning time of the existing user is done via the RTC module DS3231. Also the presence record can be performed using arduino serial monitor.

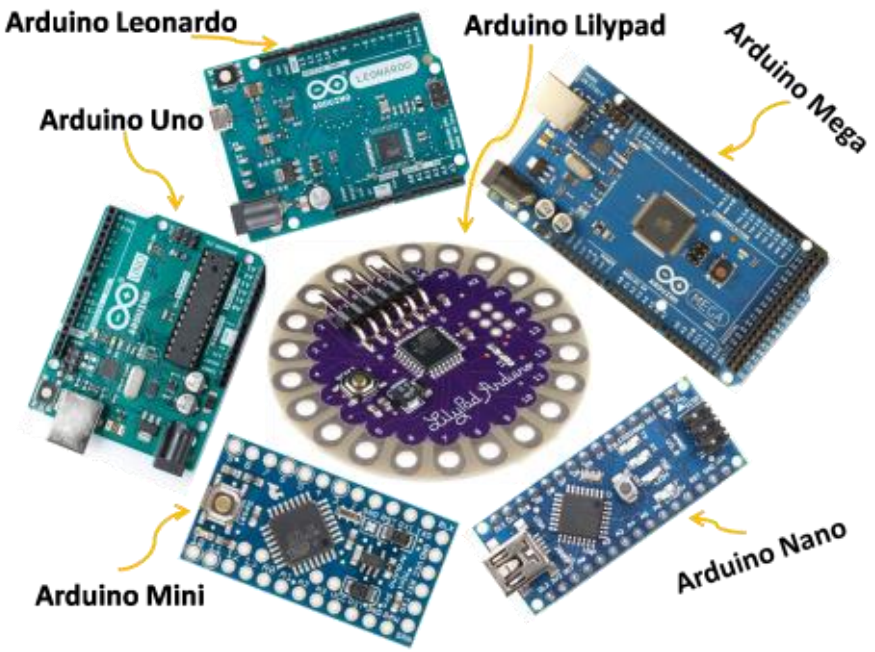
Key words: fingerprint, RTC module, scanning, presence.

1. UVOD

U ovom projektu se dizajnira biometrijski sustav registracije prisutnosti pomoću: Senzora otiska prsta, arduina, lcd zaslona, rtc modula, buzzera, te ostalih popratnih komponenata. Sustav biometrijske prisutnosti se najčešće koristi u školama i uredima kako bi se vodila evidencija prisutnosti. Ovaj projekt ima široku primjenu u školama, na fakultetima, u poslovnim organizacijama, uredima gdje se s vremenom zahtjeva točno obilježavanje prisutnosti. Korištenjem senzora otiska prsta sustav će postati sigurniji za korisnike, te povjerljiviji za osoblje koje vrši kontrolu and studentima. Prednost ovog sustava je ta što osoba mora biti prisutna kako bi se vlastoručno evidentirala.

2. ARDUINO

Arduino je razvojna platforma bazirana na Atmel-ovoj AVR seriji mikrokontrolera, a sastoji se od razvojne pločice na kojoj je smješten 8-bitni mikrokontroler, te pripadajućeg programskog paketa, pomoću kojeg se mikrokontroler programira. Razvojna pločica i pripadajući softver su izdani pod Open Source licencom, što u prijevodu znači da su koncipirani tako da ih svatko za svoje potrebe može mjenjati i unapređivati. Slijedom toga, danas na tržištu postoji više razvojnih platformi baziranih na Arduinu – tzv. "klonovi", a neki od najpoznatijih su Freeduino i Sanguino. Razvojnu pločicu možemo kupiti ili je sami izraditi, po nacrtima koji su javno dostupni na službenoj Arduino web stranici. Osnovna ideja nastala je u Italiji 2005. godine, s namjerom da se studentima i hobistima omogući jeftinija alternativa tada dostupnim razvojnim sustavima. Razvojne ploče su obično dosta skup komad hardvera, i nerijetko uključuju potrebu za komercijalnim softverskim paketima, što je predstavljalo veliku prepreku za mnoge. Relativno visoku cijenu razvojnog hardvera možemo opravdati velikim brojem ugrađenih periferija na razvojne ploče (senzori, LCD, prekidači, LED, itd.), no Arduino je tom problem doskočio na jedan specifičan način, korištenjem tzv. "shield"-ova. Princip je jednostavan – osnovna pločica sadrži samo glavni mikrokontroler i neophodne periferije (USB ili Serial sučelje, napajanje, kristalni oscilator, priključnice), dok se ovisno o namjeni i potrebama na osnovnu pločicu mogu spajati druge kompatibilne pločice (eng. shields), te tako ostvarivati željene funkcionalnosti. Neki od najpopularnijih "shield"-ova su: Ethernet, LCD, Wireless, Motor, ZigBee, RFID i drugi. Postoji više izvedbi Arduino razvojnih pločica (Slika 2.1), ovisno o korištenom mikrokontroleru, koje se razlikuju po broju digitalnih IO portova, analognih ulaza, PWM izlaza, te količini FLASH, RAM i EEPROM memorije. [1]



Slika 2.1- Prikaz različitih izvedbi Arduino razvojnih pločica [2]

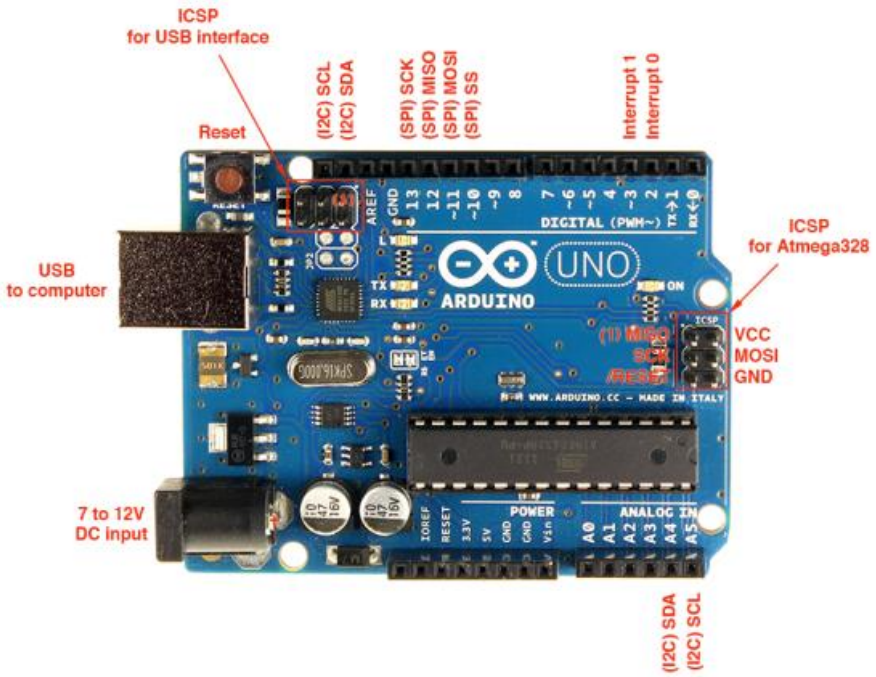
Tablica 2.1 – Karakteristike nekih izvedbi Arduino razvojnih pločica

Arduino Pločica	Ugrađeni Mikrokontroler	Frekvencija (MHz)	Flash Memorija (kB)	EEPROM (kB)	SRAM (kB)	Digitalni (PWM)	Analogni
ADK	ATmega2560	16 MHz	256	4	8	54 (14)	16
Bluetooth	ATmega328	16 MHz	32	1	2	14 (4)	6
Diecimila	ATmega168	16 MHz	16	0.5	1	14 (6)	6
Duemilanove	ATmega328P	16 MHz	32	1	0.5	14 (6)	6
Ethernet	ATmega328	16 MHz	32	1	2	14 (4)	6
Fio	ATmega328P	8 MHz	32	1	2	14 (6)	8
Leonardo	ATmega32u4	16 MHz	32	1	2	14 (6)	12
LilyPad	ATmega328V	8 MHz	16	0.5	1	14 (6)	6
Mega	ATmega1280	16 MHz	128	4	8	54 (14)	16
Mega2560	ATmega2560	16 MHz	256	4	8	54 (14)	16
Nano	ATmega328	16 MHz	32	1	0.5	14 (6)	8
Uno	ATmega328P	16 MHz	32	1	2	14 (6)	6

PWM (eng. Pulse Width Modulation) – Modulacija širine impulsa je način pretvorbe signala u kojoj se vrijeme trajanja impulsa mijenja, a frekvencija ostaje konstantna.

2.1 ARDUINO UNO

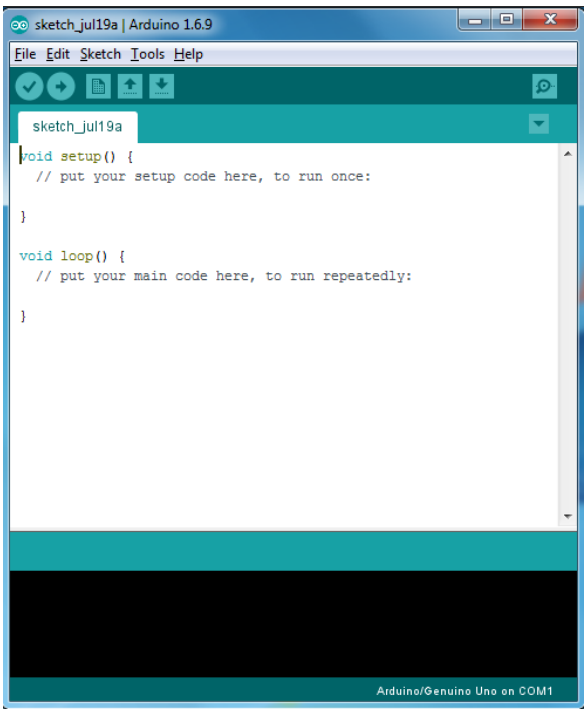
Arduino Uno razvojna pločica bazirana je na Atmelovom ATmega 328P mikrokontroleru AVR serije, na radnom taktu od 16 MHz, a opremljena je sa 14 digitalnih IO portova, od kojih 6 imaju mogućnost PWM izlaznog načina rada (Slika 2.2). Također na raspolaganju nam je 6 analognih ulaza, koji su multipleksirano spojeni na interni ADC (Analog to digital) pretvarač sa 10-bitnom razlučivošću. Spajanje na računalo omogućeno je putem USB porta, što ga čini iznimno lakim za korištenje i na najnovijim računalima, pri čemu je komunikacija sa platformom izvedena kao virtualni COM port. Prilagodba između USB i RS232 porta izvedena je softverski unutar dodatnog Atmega 8U2 mikrokontrolera, dok su neke starije verzije pločica koristile namjenski FTDI čip. Programiranje je omogućeno direktno putem USB porta, iz Arduino IDE sučelja, ili putem ICSP (eng. In Circuit Serial Programming). [1]



Slika 2.2 – Arduino Uno pločica [3]

2.2 ARDUINO IDE

Arduino razvojni tim osim same razvojne pločice kontinuirano unaprijeđuje i razvija multiplatformski programski paket, koji je podržan na Microsoft Windows, Linux i Mac OSX operativnim sustavima, pod nazivom Arduino IDE (eng. Integrated Development Environment)(Slika 2.3). Osnovna namjena ovog programskog paketa je sam unos programa u C programskom jeziku, njegovo jednostavno kompajliranje, te učitavanje (eng. upload) u sam mikrokontroler. Podržava bojanje sintakse i pruža povratne informacije (eng. debug info) u slučaju da programski kod sadrži greške. Također, u sam program ugrađena je serijska konzola putem koje možemo komunicirati sa mikrokontrolerom na pločici putem RS232 protokola. Na raspolaganju je veći broj primjera (eng. sketches) za sve ugrađene programske biblioteke, a po potrebi mogu se implementirati i vlastite biblioteke ili mjenjati postojeće. Za potrebe kompajliranja i samog programiranja mikrokontrolera na razvojnoj pločici, Arduino IDE se oslanja na AVR-GCC, kompajler za C/C++ programske jezike, koji je zapravo GCC kompajler prilagođen za upotrebu na Atmelovim mikrokontrolerima AVR serije. [1]



Slika 2.3 – Sučelje Arduino IDE

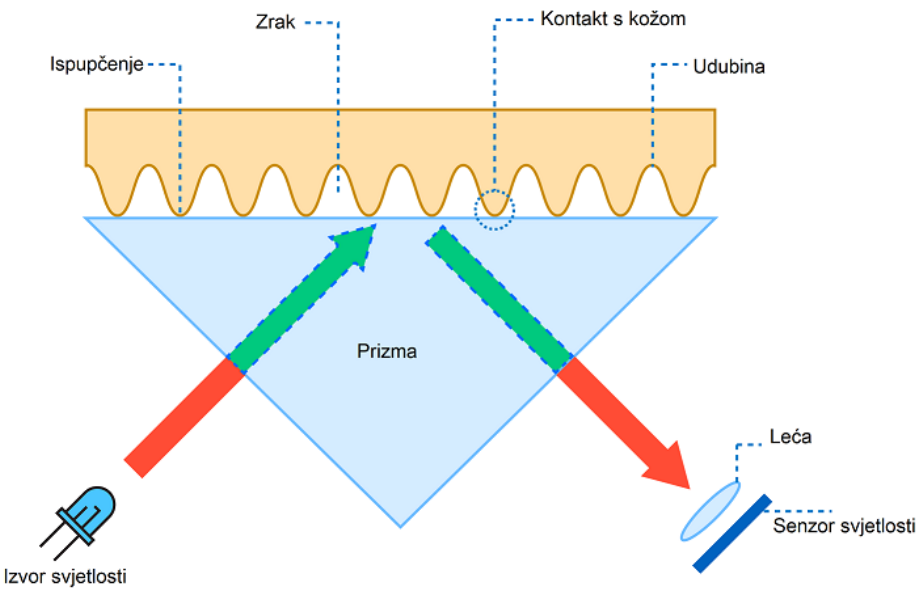
3. SENZOR OTISKA PRSTA

U ovom poglavlju spominje se senzor otiska prsta koji se koristi u mnogim primjenama gdje je potrebno da se određeni korisnik prijavi. Kao što je već poznato većina pametnih telefona podržava otključavanje s otiskom prsta, mnoge prostorije su zaštićene na ovaj način od neovlaštenog ulaza. Ovaj senzor nalazi primjenu u raznim sustavima sigurnosti jer koristi baš otisak prsta koji je jedinstven za svakoga (Slika 3.1). Kako bi zaštitili nešto bitno od sada ćete to moći napraviti s vrlo visokom razinom sigurnosti uz ovaj senzor. [4]

Slika 3.1 – Senzor otiska prsta [4]

Princip rada senzora otiska prsta

Da bi se daljnji koncept razumio treba biti upoznat s načinom rada senzora. Ovaj senzor je optički što znači da pomoću svijetla prepoznaje otisak prsta, a osim optičkog ima još i kapacitivni sensor (koristi se na mobilnim uređajima) i ultrazvučni. Optički senzor otiska prsta je starija tehnologija pa je stoga i jednostavnija od ostalih, ali baš zbog jednostavnosti rada je pristupačna za razne projekte. Kao što je poznato na prstu se nalaze ispupčenja i udubine koje su jedinstvene za svaku osobu i to je otisak prsta. Da bi detektirao udubine i ispupčenja senzor otiska prsta koristi led diode kao osvjetljenje i fotodetektor kako bi primio odbijene zrake svjetlosti (Slika 3.2). Naravno senzor nema samo jednu diodu i fotodetektor nego se unutar njega nalazi puno fotodetektora smještenih u matricu, kao i prizma i leća kako bi točnije mogao primiti odbijenu svjetlost. Na slici je vidljiv princip rada senzora koji je prethodno objašnjen. [4]



Slika 3.2 – Princip rada senzora otiska prsta [4]

4. REAL TIME CLOCK MODUL

Real-Time Clock je elektronički uređaj koji vrlo precizno prati vrijeme. U ovom slučaju riječ je o novijim modelom DS3231SN prikazan na Slici 4.1. RTC moduli uglavnom koriste kristal oscilator frekvencije 32.768 kHz, koja se koristi i u quartz satovima. U usporedbi s DS1307, model DS3231 ima ugrađen TCXO (eng. Temperature Compensated Crystal Oscillator), što donosi stabilnost koju s običnim oscilatorom ne bi mogli dobiti, prilikom promjene temperature. Zajedno čine cjelinu koja omogućuje dugotrajno i precizno praćenje vremena. [5]

Slika 4.1 – RTC Modul

Karakteristike

Napon:	3.3V - 5.5V
Struja:	170uA (stand-by 5.5V)
Komunikacija:	400kHz I2C (default address 0x68)
Temperatura:	-40°C do +85°C
Dimenzije:	38mm x 22mm x 14mm

Princip rada RTC modula

RTC prati sekunde, minute, sate, dane u tjednu, datum: dan u mjesecu, mjesec i godinu. Kada se jednom postavi vrijeme može se pratiti kalendar sve do 2100. godine. To znači da se datum na kraju mjeseca automatski prebacuje, isto vrijedi i za prijestupne godine. Sat radi u 24 i 12 satnom formatu, ugrađena su i dva programibilna alarma. Modul i kontroler povezani su preko dvojsmerne I2C komunikacije. Sve potrebno za ovu komunikaciju nalazi se na breakoutu. Ukoliko postoji potreba za promjenom I2C adrese, to se radi povezivanjem A0, A1 i A2 pinova na breakoutu modula. Baterija se koristi s modulom iz razloga sačuvanja postavljenog vremena. Kada sustav unutar DS3231 primjeti da je došlo do prekida stalnog izvora napajanja, prebacuje se na bateriju. Baterija koja odgovara breakoutu je CR2032, kapaciteta 240 mAh. Ista stvar se automatski obavlja i u suprotnom smjeru. [5]

5. LCD DISPLAY

LCD (eng. Liquid Crystal Display)(Slika 5.1) ekran je vjerojatno jedan od najkorištenijih modula jer je uistinu koristan. Omogućuje ispisivanje bilokakvih vrsta infomacija na lako čitljivom LCD ekranu. LCD je s plavim pozadinskim osvjetljenjem i bijelim slovima. Oba LCDa, 16x2 te 20x4, imaju identičan raspored pinova. [6]

Slika 5.1 – LCD Display

Karakteristike

Veličina: 16 znakova u 2 reda (16x2) ili 20 znakova u 4 reda(20x4)

Napon: 5V

Veličina ekrana: 64.5 x 16 mm(za 16x2)

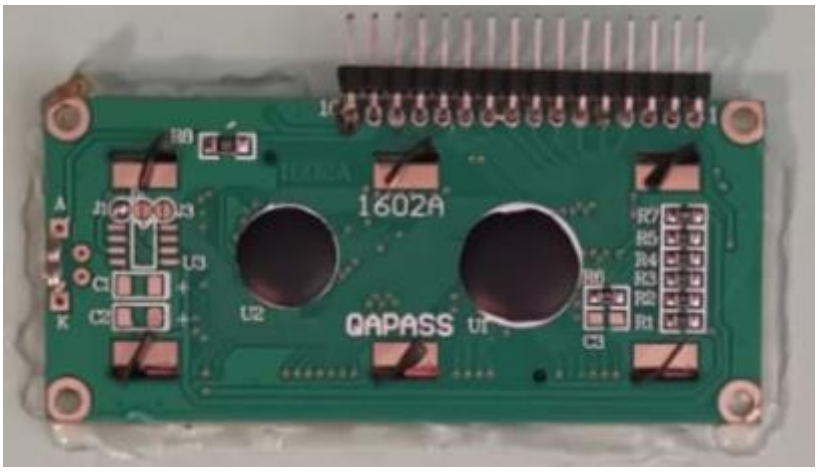
Veličina modula: 80 x 36 x 12 mm(za 16x2)

Boja znakova: bijela

Pozadinsko osvjetljenje: plava

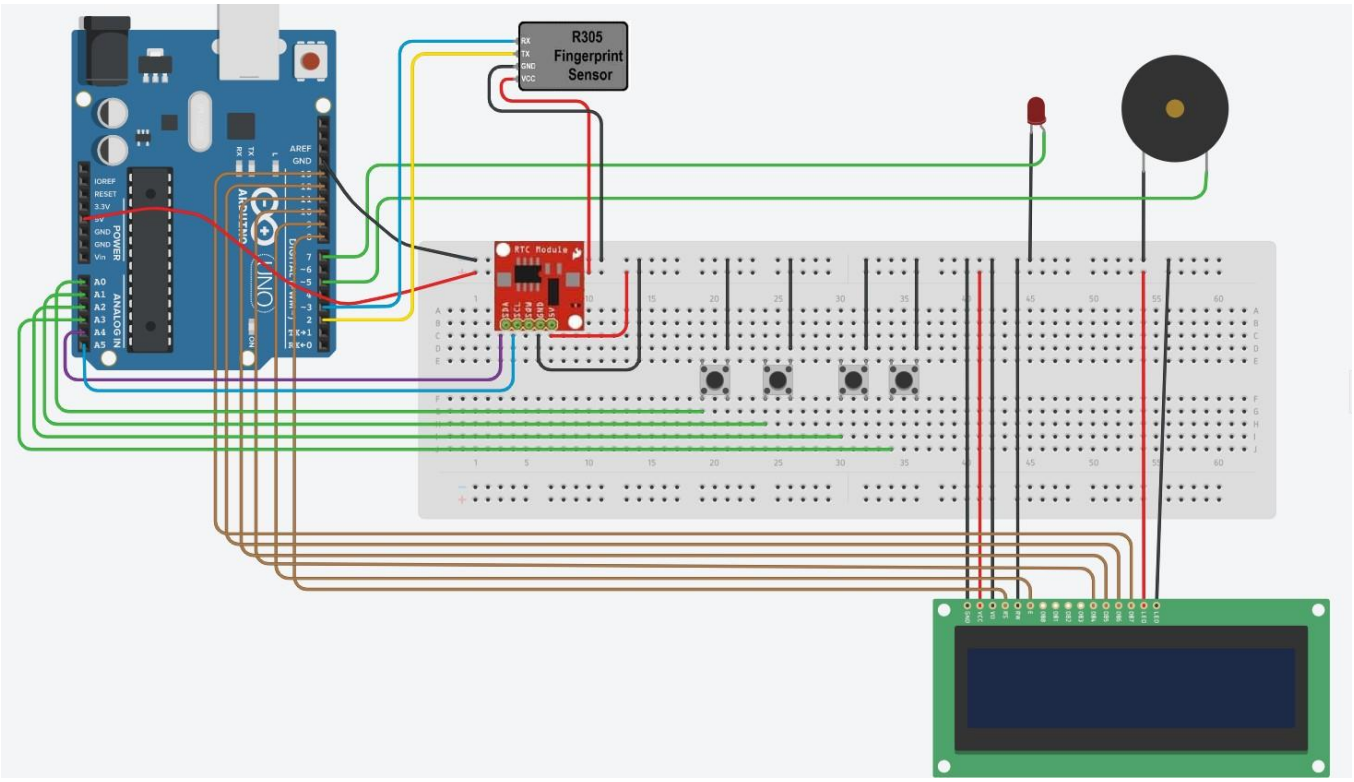
Princip rada LCD Displaya

Svaka točka (eng. pixel) LCD-a obično se sastoji od sloja molekula poredanih između dvije prozirne elektrode i dva filtera-polarizatora čije su osi polarizacije okomite jedna na drugu. Kada između polarizirajućih filtera ne bi bilo tekućeg kristala, svjetlost propuštenu kroz prvi filter, drugi (okomiti filter) bi apsorbirao. Prije pojave električnog polja, orijentacija molekula tekućeg kristala određena je orijentacijom površina elektroda. U zakrenutom nematičkom uređaju (eng. Twisted nematic field effect), zbog okomite orijentacije dvaju elektroda, molekule se orijentiraju u spiralnu strukturu. To smanjuje rotaciju polarizacije odbijene (reflektirane) svjetlosti i uređaj izgleda sivo. Ako se stvori dovoljno visok napon, molekule tekućeg kristala u središtu sloja gotovo se posve ispravljaju i polarizacija ne zakreće odbijenu svjetlost . Svjetlost će dakle većinom biti polarizirana okomito na drugi filter i stoga apsorbirana pa će točka izgledati crno. Upravljanjem naponom koji prolazi kroz sloj tekućeg kristala svake od točaka, utječe se na količinu propuštenog svjetla - dakle na svjetlinu točke (tonovi sive). Na Slici 5.2 prikazana je poleđina LCD-a na kojoj se vide SMD komponente.[7]



Slika 5.2 – LCD Display pozadina

6. SHEMA SPAJANJA SKLOPA



Slika 6.1 – Shema spajanja sklopa

Na Slici 6.1 prikazana je shema sklopa koja je izrađena u tinkercadu.

- 5V pin s Arduina se spaja na donji vodoravni red ploče.

- Slično tome, GND pin s Arduina se spaja na gornji vodoravni red ploče.

- Digitalni pinovi 8,9,10,11,12,13 idu na RS, E, DB4, DB5, DB6, DB7 LCD-a.

- Po jedna nožica svakog tipkala se spaja na analogne ulaze A0, A1, A2, A3 dok se ostale nožice spajaju na GND.

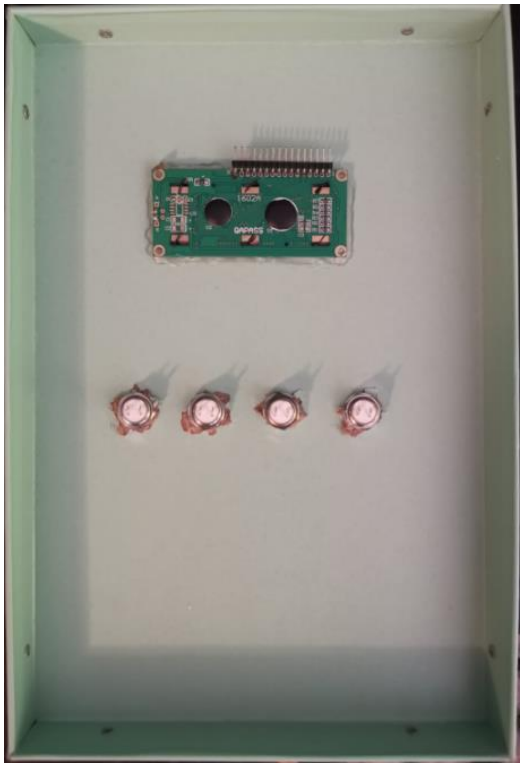
- RX i TX izlazi senzora se spajaju na digitalne pinove 2 i 3, te preostale 2 žice idu na 5V i GND.

- SDA i SCL izlazi RTC modula se spajaju na analogne ulaze A4 i A5, te preostale 2 žice idu na 5V i GND.

7. SLIKE RADA

Slika 7.1 – Postavljanje Arduina u kućište

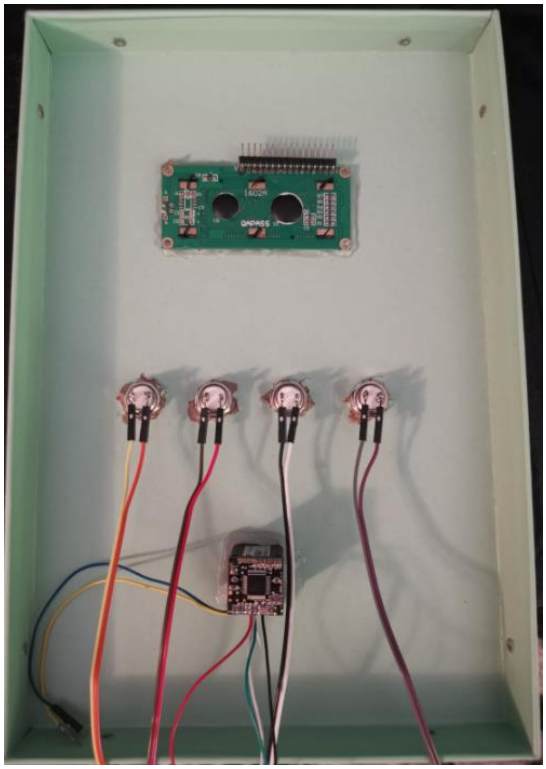
Na Slici 7.1 vidljivo je postavljanje Arduina i breadboarda. Komponente su zalijepljene i osigurane pomoću dvostrane samoljepljive trake i vrućeg ljepila. Arduino je dodatno osiguran pomoću plastičnog dijela breadboarda kako nebi došlo do pomicanja komponenata prilikom iskopčavanja arduina.



Slika 7.2 – Tipkala i LCD prednja strana

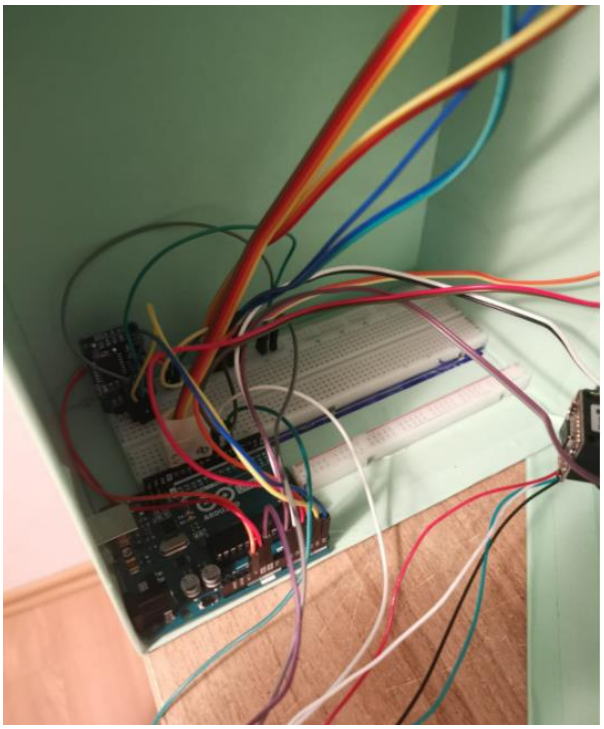
Slika 7.3 – Tipkala I LCD stražnja strana

Slike 7.2 i 7.3 prikazivaju postavljanje i osiguravanje na mjesto 4 tipkala i LCD-a. LCD je osiguran pomoću vrućeg ljepila, dok su tipkala pričvršćena s pripadajućim vijcima i maticama. U isto vrijeme uzela se u obzir preciznost i simetričnost položaja pripadajućih komponenata.



Slika 7.4 – Nalemljivanje vodiča na tipkala

Postavljanje senzora otiska prsta na položaj koji je najprirodniji korisniku prilikom registracije. Slika 7.4 prikaziva postupak mekog lemljenja je izvršen pomoću lemilice Hakko FX888D i lem žice na bazi kositra.



Slika 7.5 – Povezivanje komponenata

Slika 7.6 – Povezivanje ostatalih komponenti

Završno povezivanje svih pripadajućih komponenti poput led diode i piezo buzzera se može vidjeti na Slikama 7.5 i 7.6. Također bilo je potrebno produžiti vodiče senzora otiska prsta, te samim time rješiti problem loše konekcije između vodova i breadboarda. Time su se izbjegla moguća pomicanja i iskopčavanja koja bi uzrokovala nepravilan rad sustava.

Slika 7.7 – Završni izgled rada

Za kraj Slika 7.7 prikaziva završnu izvedbu projekta na kojoj se vidi korisničko sučelje, te pripadajuće komponente potrebne za interakciju korisnika i sustava. Svako tipkalo tipkalo ima svoj opis funkcije kako bi sustav bio jednostavan za korištenje.

8. ZAKLJUČAK

Ostaje izazov za odabir odgovarajućih vrsta hardvera za projektiranje biometrijskog sustava pohađanja. Stoga se daje pregled zajedno s informacijama u kojima su sažeta svojstva i karakteristike svake vrste hardverskih komponenti. Postoje različite vrste ploča mikrokontrolera, biometrijskih senzora, komunikacijskih kanala, skladišta baza podataka i drugih komponenti koje istraživači mogu odabrati na temelju vlastitih potreba i zahtjeva. Konačno, potrebne su i druge komponente koje nadopunjuju cijeli biometrijski sustav pohađanja. U velikoj učionici istraživači se mogu odlučiti za snažniji mikrokontroler, beskontaktni senzor, veću pohranu baze podataka i komunikacijski kanal s velikom brzinom prijenosa podataka. Ukratko, mjerilo za odabir hardverskih uređaja ili komponenti uvijek se sužava na tri važna kriterija, a to su cijena, potrošnja energije i brzina.

KOD

```
#include<EEPROM.h>    //naredba za pohranu podataka
#include<LiquidCrystal.h>
LiquidCrystal lcd(8,9,10,11,12,13);
#include <SoftwareSerial.h>
#include <Adafruit_Fingerprint.h>
SoftwareSerial fingerPrint(2, 3); //za tx/rx komunikaciju između arduina
& r305 fingerprint senzora
#include <Wire.h>
#include "RTClib.h"      //biblioteka za DS3231 RTC Module
RTC_DS3231 rtc;
uint8_t id;
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&fingerPrint);
#define register_back 14
#define delete_ok 15
#define forward 16
#define reverse 17
#define match 5
#define indFinger 7
#define buzzer 5
#define records 10 // 10 za 10 korisnika
int user1,user2,user3,user4,user5,user6,user7,user8,user9,user10;
DateTime now;
void setup()
{
delay(1000);
lcd.begin(16,2);
Serial.begin(9600);
pinMode(register_back, INPUT_PULLUP);
pinMode(forward, INPUT_PULLUP);
pinMode(reverse, INPUT_PULLUP);
```

```
pinMode(delete_ok, INPUT_PULLUP);
pinMode(match, INPUT_PULLUP);
pinMode(buzzer, OUTPUT);
pinMode(indFinger, OUTPUT);
//rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
//rtc.adjust(DateTime(2021, 6, 23, 23, 5, 0));
digitalWrite(buzzer, LOW);
if(digitalRead(register_back) == 0)
{
digitalWrite(buzzer, HIGH);
delay(500);
digitalWrite(buzzer, LOW);
lcd.clear();
lcd.print("Preuzimanje !");
lcd.setCursor(0,1);
lcd.print("Podataka..");
Serial.println("Preuzimanje");
Serial.println("Podataka");
Serial.println();
Serial.print("S.No. ");
for(int i=0;i<records;i++)
{
digitalWrite(buzzer, HIGH);
delay(500);
digitalWrite(buzzer, LOW);
Serial.print(" Korisnicki ID");
Serial.print(i+1);
Serial.print(" ");
}
Serial.println();
int eepIndex=0;
for(int i=0;i<30;i++)
{
```

```
if(i+1<10)
Serial.print('0');
Serial.print(i+1);
Serial.print(" ");
eepIndex=(i*7);
download(eepIndex);
eepIndex=(i*7)+210;
download(eepIndex);
eepIndex=(i*7)+420;
download(eepIndex);
eepIndex=(i*7)+630;
download(eepIndex);
eepIndex=(i*7)+840;
download(eepIndex);
eepIndex=(i*7)+1050;
download(eepIndex);
eepIndex=(i*7)+1260;
download(eepIndex);
eepIndex=(i*7)+1470;
download(eepIndex);
eepIndex=(i*7)+1680;
download(eepIndex);
Serial.println();
}
}
if(digitalRead(delete_ok) == 0)
{
lcd.clear();
lcd.print("Molim pricekajte");
lcd.setCursor(0,1);
lcd.print("Ponovno pokretanje...");
for(int i=1000;i<1005;i++)
EEPROM.write(i,0);
```

```
for(int i=0;i<841;i++)
EEPROM.write(i, 0xff);
lcd.clear();
lcd.print("Ponovno pokretanje");
lcd.setCursor(0,1);
lcd.print("sistema");
delay(1000);
}

lcd.clear();
lcd.print("  Evidencija");
lcd.setCursor(0,1);
lcd.print("  Prisutnosti");
delay(4000);
lcd.clear();

digitalWrite(buzzer, HIGH);
delay(500);
digitalWrite(buzzer, LOW);
for(int i=1000;i<1000+records;i++)
{
if(EEPROM.read(i) == 0xff)
EEPROM.write(i,0);
}
finger.begin(57600);
Serial.begin(9600);
lcd.clear();
lcd.print("Trazenje Modula ..");
lcd.setCursor(0,1);
delay(2000);
if (finger.verifyPassword())
{
Serial.println("Senzor otiska pronaden!");
```

```
lcd.clear();
lcd.print(" Modul pronaden");
delay(3000);
}
else
{
Serial.println("Senzor otiska nije pronaden");
lcd.clear();
lcd.print("Modul nije pronaden");
lcd.setCursor(0,1);
lcd.print("Provjeri vezu");
while (1);
}
if (! rtc.begin())
{
Serial.println("RTC nije pronaden");
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
rtc.adjust(DateTime(2021, 8, 31, 20, 5, 0));
}
if (rtc.lostPower())
{
Serial.println("RTC nije pokrenut!");
}
lcd.setCursor(0,0);
lcd.print("   Pokretanje   ");
lcd.setCursor(0,1);
lcd.print("    sistema   ");
delay(3000);

user1=EEPROM.read(1000);
user2=EEPROM.read(1001);
user3=EEPROM.read(1002);
user4=EEPROM.read(1003);
```

```
user5=EEPROM.read(1004);
lcd.clear();
digitalWrite(indFinger, HIGH);

}
void loop()
{
now = rtc.now();
lcd.setCursor(0,0);
lcd.print("Vrijeme: ");
lcd.print(now.hour(), DEC);
lcd.print(':');
lcd.print(now.minute(), DEC);
lcd.print(':');
lcd.print(now.second(), DEC);
lcd.print(" ");
lcd.setCursor(0,1);
lcd.print("Datum: ");
lcd.print(now.day(), DEC);
lcd.print('/');
lcd.print(now.month(), DEC);
lcd.print('/');
lcd.print(now.year(), DEC);
lcd.print(" ");
delay(500);

if(digitalRead(forward) == 0 & digitalRead(reverse) == 0)
{
digitalWrite(buzzer, HIGH);
delay(500);
digitalWrite(buzzer, LOW);
lcd.clear();
lcd.print("Preuzimanje");
```

```
lcd.setCursor(0,1);
lcd.print("Podataka");

Serial.println("Preuzimanje");
Serial.println("Podataka...");
Serial.println();

Serial.print("S.No. ");
for(int i=0;i<records;i++)
{
digitalWrite(buzzer, HIGH);
delay(500);
digitalWrite(buzzer, LOW);
Serial.print(" Korisnik ");
Serial.print(i+1);
Serial.print(" ---------------------- ");
}
Serial.println();
int eepIndex=0;
for(int i=0;i<30;i++)
{
if(i+1<10)
Serial.print('0');
Serial.print(i+1);
Serial.print(" ");
eepIndex=(i*7);
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+210;
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+420;
download(eepIndex);
```

```
Serial.print(" --------- ");
eepIndex=(i*7)+630;
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+840;
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+1050;
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+1260;
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+1470;
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+1680;
download(eepIndex);
Serial.print(" --------- ");
eepIndex=(i*7)+1890;
download(eepIndex);
Serial.println(" --------- ");
}
}
int result=getFingerprintIDez();
//Serial.println(digitalRead(forward));
if(result>0)
{
digitalWrite(indFinger, LOW);
digitalWrite(buzzer, HIGH);
delay(100);
digitalWrite(buzzer, LOW);
lcd.clear();
```

```
lcd.print("ID:");
lcd.print(result);
lcd.setCursor(0,1);
lcd.print("Molim pricekajte...");
delay(3000);
attendance(result);
lcd.clear();
lcd.print("Prisutnost ");
lcd.setCursor(0,1);
lcd.print("Registrirana");
delay(3000);
digitalWrite(indFinger, HIGH);
return;
}
checkKeys();
delay(300);
}
void attendance(int id)
{
int user=0,eepLoc=0;
if(id == 1)
{
eepLoc=0;
user=user1++;
}
else if(id == 2)
{
eepLoc=210;
user=user2++;
}
else if(id == 3)
{
eepLoc=420;
```

```
user=user3++;
}
else if(id == 4)
{
eepLoc=630;
user=user4++;
}
else if(id == 5)
{
eepLoc=0;
user=user5++;
}
else if(id == 6)
{
eepLoc=840;
user=user5++;
}
else if(id == 7)
{
eepLoc=1050;
user=user7++;
}
else if(id == 8)
{
eepLoc=1260;
user=user8++;
}
else if(id == 9)
{
eepLoc=1470;
user=user9++;
}
else if(id == 10)
```

```
{
eepLoc=1680;
user=user10++;
}
else
return;

int eepIndex=(user*7)+eepLoc;
EEPROM.write(eepIndex++, now.hour());
EEPROM.write(eepIndex++, now.minute());
EEPROM.write(eepIndex++, now.second());
EEPROM.write(eepIndex++, now.day());
EEPROM.write(eepIndex++, now.month());
EEPROM.write(eepIndex++, now.year()>>8 );
EEPROM.write(eepIndex++, now.year());

EEPROM.write(1000,user1);
EEPROM.write(1001,user2);
EEPROM.write(1002,user3);
EEPROM.write(1003,user4);
EEPROM.write(1004,user5);
EEPROM.write(1005,user6);
EEPROM.write(1006,user7);
EEPROM.write(1007,user8);
EEPROM.write(1008,user9);
EEPROM.write(1009,user10);
}

void checkKeys()
{
if(digitalRead(register_back) == 0)
{
lcd.clear();
```

```
lcd.print("Molim pricekajte");
delay(3000);
while(digitalRead(register_back) == 0);
Enroll();
}

else if(digitalRead(delete_ok) == 0)
{
lcd.clear();
lcd.print("Molim pricekajte");
delay(3000);
delet();
}
}

void Enroll()
{
int count=1;
lcd.clear();
lcd.print("Odaberi ID prsta:");

while(1)
{
lcd.setCursor(0,1);
lcd.print(   count);
if(digitalRead(forward) == 0)
{
count++;
if(count>records)
count=1;
delay(500);
}
```

```
else if(digitalRead(reverse) == 0)
{
count--;
if(count<1)
count=records;
delay(500);
}
else if(digitalRead(delete_ok) == 0)
{
id=count;
getFingerprintEnroll();
for(int i=0;i<records;i++)
{
if(EEPROM.read(i) != 0xff)
{
EEPROM.write(i, id);
break;
}
}
return;
}

else if(digitalRead(register_back) == 0)
{
return;
}
}
}

void delet()
{
int count=1;
lcd.clear();
```

```
lcd.print("Odaberi ID otiska:");


while(1)
{
lcd.setCursor(0,1);
lcd.print(count);
if(digitalRead(forward) == 0)
{
count++;
if(count>records)
count=1;
delay(500);
}

else if(digitalRead(reverse) == 0)
{
count--;
if(count<1)
count=records;
delay(500);
}
else if(digitalRead(delete_ok) == 0)
{
id=count;
deleteFingerprint(id);
for(int i=0;i<records;i++)
{
if(EEPROM.read(i) == id)
{
EEPROM.write(i, 0xff);
break;
}
}
```

```
return;
}

else if(digitalRead(register_back) == 0)
{
return;
}
}
}

uint8_t getFingerprintEnroll()
{
int p = -1;
lcd.clear();
lcd.print("ID otiska:");
lcd.print(id);
lcd.setCursor(0,1);
lcd.print("Prislonite prst");
delay(3000);
while (p != FINGERPRINT_OK)
{
p = finger.getImage();
switch (p)
{
case FINGERPRINT_OK:
Serial.println("Otisak ocitan");
lcd.clear();
lcd.print("Otisak ocitan");
break;
case FINGERPRINT_NOFINGER:
Serial.println("Nema otiska");
lcd.clear();
lcd.print("Prisloni prst");
```

```
break;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("GRESKA ");
lcd.clear();
lcd.print("GRESKA");
break;
case FINGERPRINT_IMAGEFAIL:
Serial.println("GRESKA");
lcd.clear();
lcd.print("GRESKA");
break;
default:
Serial.println("GRESKA");
lcd.clear();
lcd.print("GRESKA");
break;
}
}

p = finger.image2Tz(1);
switch (p) {
case FINGERPRINT_OK:
Serial.println("Otisak prepoznat");
lcd.clear();
lcd.print("Otisak prepoznat");
break;
case FINGERPRINT_IMAGEMESS:
Serial.println("Pokusaj ponovo");
lcd.clear();
lcd.print("Pokusaj ponovo");
return p;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("GRESKA");
```

```
lcd.clear();
lcd.print("GRESKA");
return p;
case FINGERPRINT_FEATUREFAIL:
Serial.println("Otisak nije pronaden");
lcd.clear();
lcd.print("Otisak nije pronaden");
return p;
case FINGERPRINT_INVALIDIMAGE:
Serial.println("Otisak nije pronaden");
lcd.clear();
lcd.print("Otisak nije pronaden");
return p;
default:
Serial.println("GRESKA");
lcd.clear();
lcd.print("GRESKA");
return p;
}

Serial.println("Maknite prst");
lcd.clear();
lcd.print("Maknite prst");
delay(3000);
p = 0;
while (p != FINGERPRINT_NOFINGER) {
p = finger.getImage();
}
Serial.print("ID "); Serial.println(id);
p = -1;
Serial.println("Prisloni isti prst");
lcd.clear();
//lcd.print("Prisloni prst");
```

```
//lcd.setCursor(0,1);
lcd.print("Ponovno prisloni prst");
while (p != FINGERPRINT_OK) {
p = finger.getImage();
switch (p) {
case FINGERPRINT_OK:
Serial.println("Otisak ocitan");
break;
case FINGERPRINT_NOFINGER:
Serial.print(".");
break;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("GRESKA");
break;
case FINGERPRINT_IMAGEFAIL:
Serial.println("GRESKA");
break;
default:
Serial.println("GRESKA");
return;
}
}
p = finger.image2Tz(2);
switch (p) {
case FINGERPRINT_OK:
Serial.println("Otisak preuzet");
break;
case FINGERPRINT_IMAGEMESS:
Serial.println("Ponovi");
return p;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("GRESKA");
return p;
```

```
case FINGERPRINT_FEATUREFAIL:
Serial.println("GRESKA");
return p;
case FINGERPRINT_INVALIDIMAGE:
Serial.println("GRESKA");
return p;
default:
Serial.println("GRESKA");
return p;
}
Serial.print("Creating model for #"); Serial.println(id);

p = finger.createModel();
if (p == FINGERPRINT_OK) {
Serial.println("Otisak se podudara!");
} else if (p == FINGERPRINT_PACKETRECIEVEERR) {
Serial.println("GRESKA");
return p;
} else if (p == FINGERPRINT_ENROLLMISMATCH) {
Serial.println("Otisak se ne podudara!");
return p;
} else {
Serial.println("GRESKA");
return p;
}
Serial.print("ID "); Serial.println(id);
p = finger.storeModel(id);
if (p == FINGERPRINT_OK) {
Serial.println("Otisak pohranjen!");
lcd.clear();
lcd.print("Otisak pohranjen!");
delay(3000);
} else if (p == FINGERPRINT_PACKETRECIEVEERR) {
```

```
Serial.println("GRESKA");
return p;
} else if (p == FINGERPRINT_BADLOCATION) {
Serial.println("Odaberi drugi ID");
return p;
} else if (p == FINGERPRINT_FLASHERR) {
Serial.println("GRESKA");
return p;
}
else {
Serial.println("GRESKA");
return p;
}
}
int getFingerprintIDez()
{
uint8_t p = finger.getImage();

if (p != FINGERPRINT_OK)
return -1;

p = finger.image2Tz();
if (p != FINGERPRINT_OK)
return -1;

p = finger.fingerFastSearch();
if (p != FINGERPRINT_OK)
{
lcd.clear();
lcd.print("Otisak nije ");
lcd.setCursor(0,1);
lcd.print("pronaden");
delay(3000);
```

```
return -1;
}
// found a match!
Serial.print("Pronaden ID #");
Serial.print(finger.fingerID);
return finger.fingerID;
}
uint8_t deleteFingerprint(uint8_t id)
{
uint8_t p = -1;
lcd.clear();
lcd.print("Molim pricekajte");
p = finger.deleteModel(id);
if (p == FINGERPRINT_OK)
{
Serial.println("Obrisan!");
lcd.clear();
lcd.print("Otisak uspjesno");
lcd.setCursor(0,1);
lcd.print("obrisan");
delay(3000);
}
else
{
Serial.print("Nesto nije u redu");
lcd.clear();
lcd.print("Nesto nije u redu");
lcd.setCursor(0,1);
lcd.print("Pokusaj ponovno");
delay(3000);
return p;
}
}
```

```
void download(int eepIndex)
{
if(EEPROM.read(eepIndex) != 0xff)
{
Serial.print("T->");
if(EEPROM.read(eepIndex)<10)
Serial.print('0');
Serial.print(EEPROM.read(eepIndex++));
Serial.print(':');
if(EEPROM.read(eepIndex)<10)
Serial.print('0');
Serial.print(EEPROM.read(eepIndex++));
Serial.print(':');
if(EEPROM.read(eepIndex)<10)
Serial.print('0');
Serial.print(EEPROM.read(eepIndex++));
Serial.print(" D->");
if(EEPROM.read(eepIndex)<10)
Serial.print('0');
Serial.print(EEPROM.read(eepIndex++));
Serial.print('/');
if(EEPROM.read(eepIndex)<10)
Serial.print('0');
Serial.print(EEPROM.read(eepIndex++));
Serial.print('/');
Serial.print(EEPROM.read(eepIndex++)<<8 | EEPROM.read(eepIndex++));
}
else
{
Serial.print("--------------------------");
}
Serial.print(" ");
}
```

LITERATURA

[1] https://www.arduino.cc/

[2] https://pijaeducation.com/arduino/introduction-arduino/specification-of-arduino-uno/

[3] https://www.electroschematics.com/arduino-pinout/

[4] https://e-radionica.com/hr/blog/2020/05/03/kkm-senzor-otiska-prsta/

[5] https://e-radionica.com/hr/blog/2016/02/17/kkm-rtc-real-time-clock/

[6] https://e-radionica.com/hr/blog/2015/08/19/kkm-lcd-16x2/

[7] https://hr.wikipedia.org/wiki/LCD

POPIS SLIKA

POPIS TABLICA