Sat s riječima pomoću niza upravljivih dioda

Veselin, Ivan

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

Sveučilišni studij

SAT S RIJEČIMA POMOĆU NIZA UPRAVLJIVIH DIODA

Završni rad

Ivan Veselin

Osijek, 2016.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

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

Osijek, 17.09.2016.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Ivan Veselin
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3597, 02.08.2013.
OIB studenta:	01814808697
Mentor:	Doc.dr.sc. Ivan Aleksi
Sumentor:	Dr.sc. Ivan Vidović
Naslov završnog rada:	Sat s riječima pomoću niza upravljivih dioda
Znanstvena grana rada:	**Arhitektura računalnih sustava (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 Postignuti rezultati u odnosu na složenost zadatka: 3 Jasnoća pismenog izražavanja: 3 Razina samostalnosti: 3
Datum prijedloga ocjene mentora:	17.09.2016.
Datum potvrde ocjene Odbora:	28.09.2016.

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, 28.09.2016.

Ime i prezime studenta:	Ivan Veselin
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3597, 02.08.2013.
Ephorus podudaranje [%]:	3

Ovom izjavom izjavljujem da je rad pod nazivom**: Sat s riječima pomoću niza upravljivih dioda**

izrađen pod vodstvom mentora Doc.dr.sc. Ivan Aleksi

i sumentora Dr.sc. Ivan Vidović

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:

SADRŽAJ

1. UVOD

Cilj ovog završnog rada je pojednostaviti dizajn sata s riječima izradom vlastite makete pomoću WS2812b adresibilnih LED-ica (eng. Light emmiting diode) i Arduino Nano mikroupravljača te dodati neke druge funkcije kao što su različite boje sata, animacije na ekranu i sl. Maketa se sastoji od prednjeg stakla sa nalijepljenim slovima ispod kojega će stajati podloga koja se sastoji od paus papira, drvenog nosača izbušenog na mjestu svake diode/slova, 14 traka LED-ica i stražnjeg poklopca za bolje vizualno rješenje.

1.1 Zadatak završnog rada

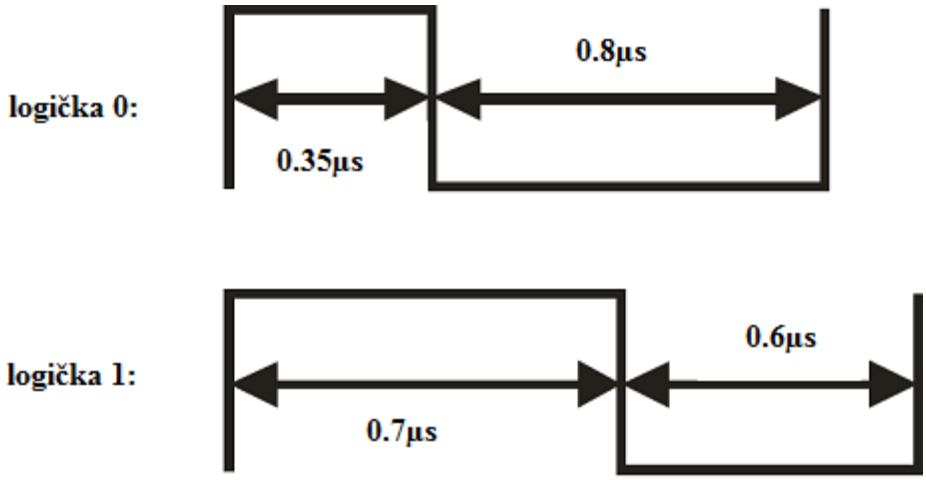
U ovom završnom radu potrebno je napraviti maketu za prikaz sata na kojoj su slovima napisane rečenice. Pomoću paljenja gašenja određenih slova omogućeno je prikazivanje sati i minuta. Pomoću upravljivih RGB dioda potrebno je unaprijediti i pojednostaviti već postojeće rješenje, tzv. QLOCK2.

2. PRIMJENJENE TEHNOLOGIJE I ALATI

2.1 WS2812b diode

WS2812b LE diode su posebne zbog digitalnog adresiranja pojedine diode na traci. Trake sa diodama dolaze u 3 izvedbe: 30, 60 i 144 LED-ica po metru. Sve trake imaju 3 kontakta na početku i kraju, +5V, GND (eng. Ground) za napajanje i Di (eng. Data input) odnosno Do (eng. Data output) za ulaz/izlaz podatkovne linije sa mikroupravljača. Pojedine diode su serijski spojene na traci i dijele isto napajanje i podatkovnu liniju. Posebnost ovih i sličnih dioda je što iza same LE diode imaju integrirani upravljački krug koji je zadužen za prijem podataka, obradu i slanje podataka idućoj diodi u seriji.

Slanje podataka je serijsko, ali za razliku od UART ili sličnih protokola i logička jedinica (1) i logička nula (0) zahtijevaju pravokutni signal gdje razliku između jedinice i nule čini različito vrijeme trajanja impulsa (Sl. 2.1).



Sl. 2.1 Način slanja podataka

Potrošnja trake proporcionalna je sa brojem LED-ica u nizu iz čega se može zaključiti da će potrošnja direktno ovisiti o odabranoj izvedbi trake i duljini segmenta koji se koristi. Osnovne vrijednosti za segmente duljine jedan metar dane su u tablici 2.1 i izračunate su iz podatka da jedna dioda treba 60mA jakosti struje kako bi svjetlila maksimalnim intenzitetom.

Cijena traka je također povezana sa odabranom izvedbom tj. brojem dioda u metru trake i prikazana je u tablici 2.1

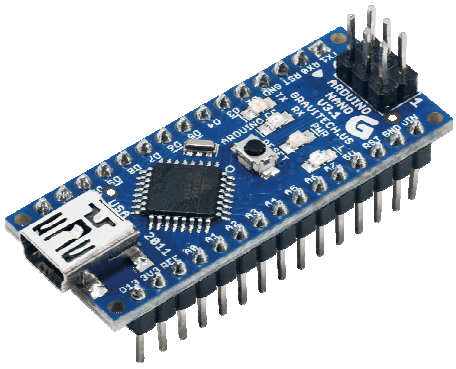
Tab. 2.1 Prikaz cijene i potrošnje po izvedbi LED trake

Izvedba trake[broj dioda/m]	Potrošnja po metru[W/m]	Cijena po metru[kn/m]
30 LED/m	9	34,78
60 LED/m	18	44,54
144 LED/m	43,2	99,56

Zbog svog dizajna, trake sa ovim diodama su osjetljive na variranje napona i prenapone koji se javljaju pri uključenju na neke izvore napajanja. Prema [1] taj problem se lako rješava stavljanjem kondenzatora na izvor napajanja kako bi se filtrirao prenapon. U ovoj maketi se koristi elektrolitski kondenzator kapaciteta 470µF, nazivnog napona 16V što je manji kapacitet nego što je predložen, ali također koristimo kvalitetno napajanje koje ima ugrađene sklopove za smanjenje prenapona. Iskustvo korisnika ovih dioda je pokazalo da se prva dioda u nizu zna oštetiti zbog naglih skokova vrijednosti na pinovima mikroupravljača pa je preporuka da se na ulazu podatkovne linije stavi mali otpornik, u našem slučaju 330Ω, kako do toga ne bi dolazilo. Kako ove LED-ice zahtijevaju vremenski precizne promjene na podatkovnoj liniji, u programu će se koristiti unaprijed definirana biblioteka tvrtke Adafruit za kontroliranje dioda, koje oni nazivaju Neopixel elementima.

2.2 Arduino Nano

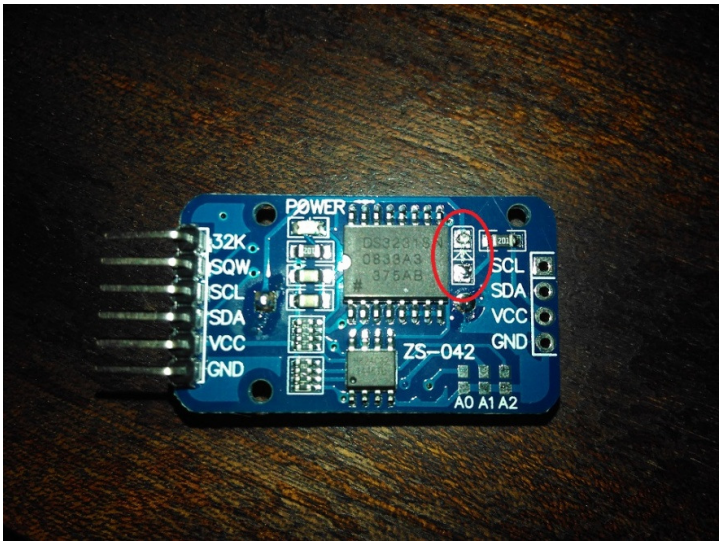
Odabir druge ključne komponente, mikroupravljača, morao je odgovarati LED-icama, tj. odabrani mikroupravljač morao je biti u mogućnosti kontrolirati niz od min. 224 LED-ice. Nano verzija (Sl. 2.2) je odabrana zbog svojih malih dimenzija, male cijene i mogućnosti lakog umetanja u prototipnu pločicu. Prema [2], Nano se temelji na ATmega 328 čipu koji ima 2kB RAM memorije i 32kB flash memorije. Zbog ograničenosti RAM-a program mora biti napisan učinkovito i sa što manje nepotrebnih varijabli i sl.



Sl. 2.2 Arduino Nano

2.3 DS3231 RTC modul

Budući da Arduino Nano razvojna maketa nema integrirani sklop za praćenje stvarnog vremena, potrebno je dodati RTC modul kompatibilan sa Arduinom. Zbog niske cijene, a zadovoljavajuće preciznosti odabran je DS3231 modul koji se sa Arduinom povezuje preko I2C protokola (eng. Inter-integrated Circuit). Prema [3], I2C protokol služi za povezivanje više podređenih digitalnih sklopova na jedan ili više nadređenih sklopova. Prema [2], pinovi SDA i SCL se na Arduino Nano mikroupravljaču nalaze na A4 (SDA) i A5 (SCL). DS3231 modul ima i 32K i SQW pinove, ali nama nisu potrebni jer dobivamo punu funkcionalnost korištenjem I2C protokola. Problem kod RTC modula je taj što ima integrirani krug za dopunjavanje baterije što znači da bi se trebala umetnuti skuplja i punjiva LIR2032 baterija. No prema [4], postoji mogućnost prekidanja kruga za punjenje (Sl. 2.3) i korištenja jeftinije i dugotrajnije CR2032 baterije.

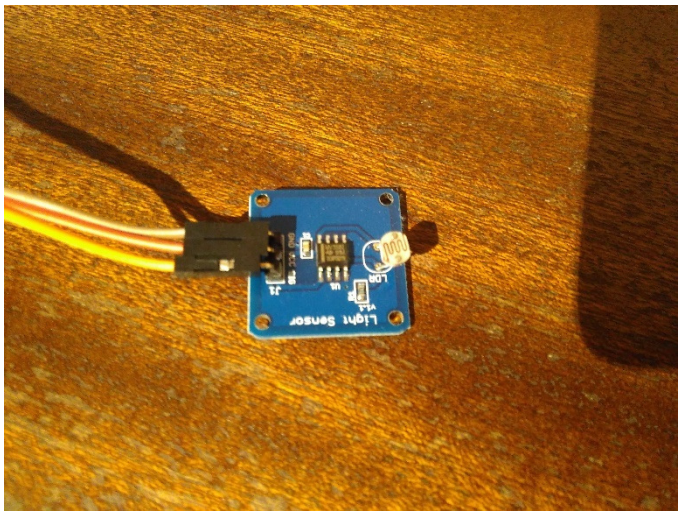


Sl. 2.3 RTC modul prilagođen za CR2032 bateriju

Još jedna prednost DS3231 modula je i mogućnost očitanja temperature što se može iskoristiti kao dodatna funkcionalnost makete.

2.4 LDR senzor osvjetljenja

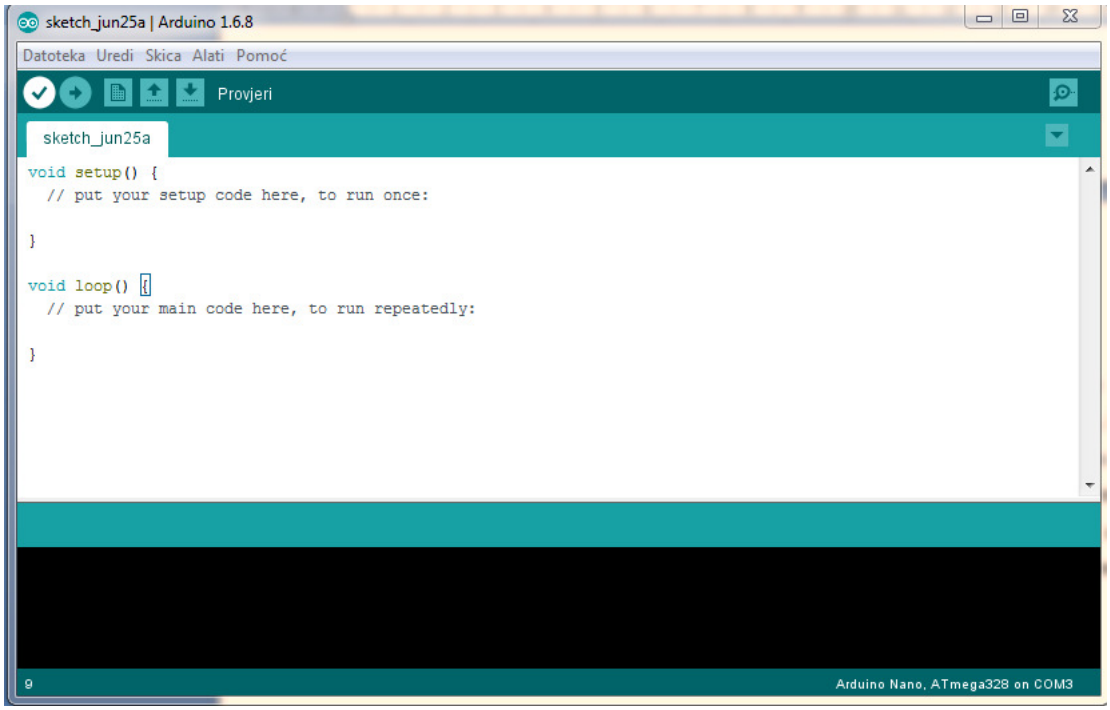
Analogni modul (Sl. 2.4) temeljen na LDR-u (eng. Light Dependent Resistor) koji sadrži potrebne komponente i na Arduino se povezuje na jedan pin za analogno čitanje (A0). U maketi će služiti za prilagođavanje svjetline LED-ica tj. ekrana.



Sl. 2.4 LDR modul osvjetljenja

2.5 Arduino IDE

Program se piše u besplatnom Arduino IDE (Sl.2.5) okruženju napravljenom za Arduino platformu. Sastoji se od uređivača koda, dijela za prijenos programa tj. programatora, i drugih korisnih mogućnosti. Zajedno sa Arduino mikroupravljačima čini cijelinu dizajniranu za lako korištenje, modifikacije i učenje. Velika prednost je i zajednica iza Arduino platforme koja kontinuirano radi na razvoju, otkrivanju i otklanjanju grešaka te pisanju tutoriala za razne dijelove okruženja. Sintaksa koda se temelji na C++ jeziku, tako da nije potrebno učenje novog programskog jezika.



Sl. 2.5 Arduino IDE

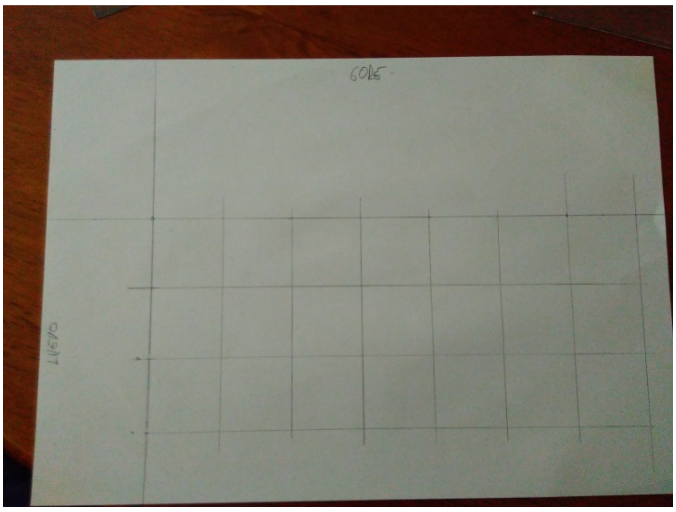
3. REALIZACIJA SUSTAVA

Maketa se sastoji od hardverskog i softverskog rješenja. Kako za Arduino okruženje već postoje napisane biblioteke za kontrolu LED-ica te činjenica da je Arduino platforma otvorenog koda što smanjuje cijenu, Arduino je bio logičan izbor. Realizacija hardverskog rješenja sastoji se od prototipne pločice kojom se povezuju moduli i zalemljenih LED traka prethodno izrezanih na mjeru. Softver je pisan za Arduino uz primjenu široko dostupnih biblioteka za kontrolu LED-ica i čitanje vrijednosti sata.

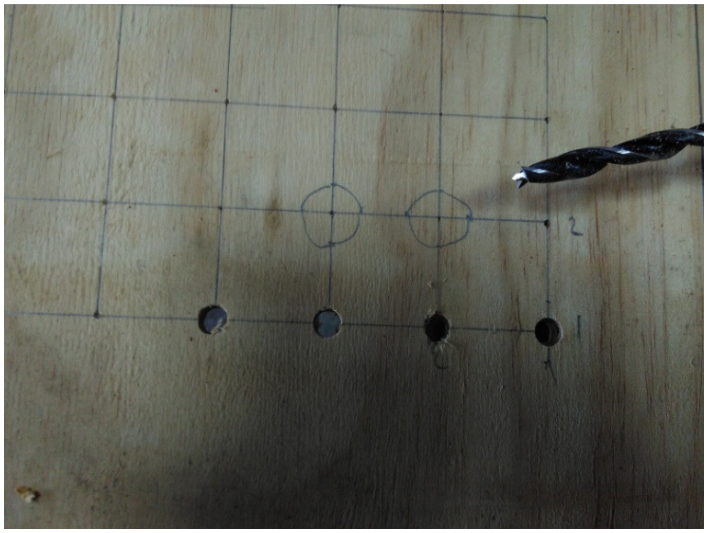
Elektroničke komponente su povezane na prototipnoj pločici. Smještene su iza nosača LED-ica u napravljenom stražnjem poklopcu. Elektroničke komponente i LED-ice napajaju se napajanjem nazivnog napona 5V i nazivne jakosti struje 10A.

3.1 Kućište i nosač hardvera

Nosač LED-ica i stražnji poklopac su izrađeni od materijala medijapana zbog svoje težine i strukturne stabilnosti, a dovoljne mekoće za ručnu obradu. Vanjske dimenzije nosača, a samim time i stakla sa naljepnicama, su 600x600mm. Prvi korak pri izradi nosača je bilo ocrtati dimenzije na komadu materijala (Sl. 3.1). Kako je razmak između pojedinih dioda na traci 33.334mm bilo je potrebno precizno označiti, a zatim i izbušiti 224 rupe kroz medijapan ploču kako bi svjetlost dioda kroz njih prolazila. Kako bi se olakšao postupak, određen je položaj prve rupe (gore lijevo), 50mm od lijevog ruba i 80mm od gornjeg ruba. Ostatak rupa je označen pomoću unaprijed izrađene šablone na komadu A4 papira (Sl. 3.2). Nakon toga su rupe izbušene svrdlom za drvo promjera 8mm sa vrškom kako bi se precizno pratile oznake. Na onoj strani nosača gdje će ići staklo sa naljepnicom potrebno je potrebno je rupe proširiti na dimenziju promjera 22mm. U tu svrhu se koristi svrdlo za metal toga promjera. S tim je izrada nosača gotova. Stražnji poklopac je također izrađen od medijapan letvica i tanke medijapan ploče kako bi se smanjila debljina i masa gotove makete.



Sl. 3.1 Šablona za označavanje rupa



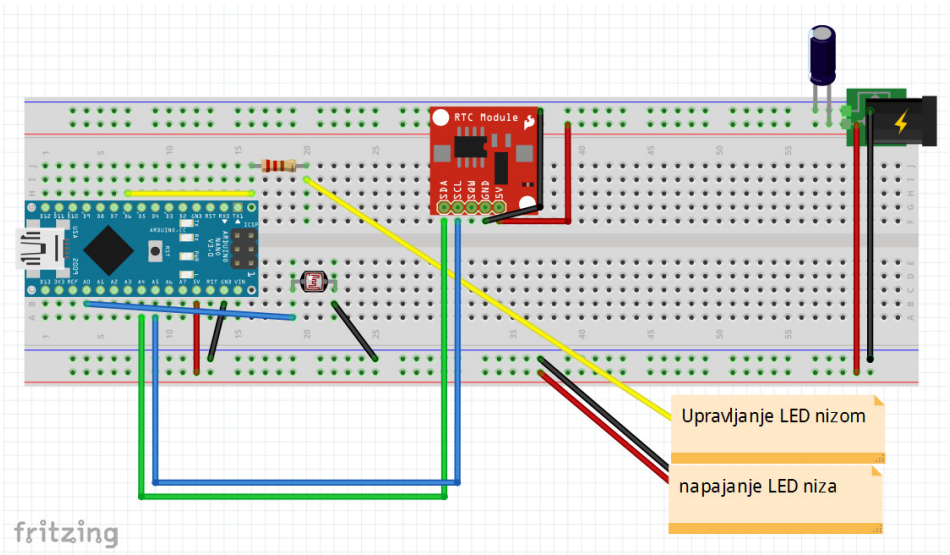
Sl. 3.2 Svrdlo sa vrhom

Izrada stakla sa naljepnicom je povjerena tvrtki koja se bavi izradom reklama zbog nedostatka znanja i alata za taj posao. Dizajn naljepnice sa određenim položajem slova i centriranjem ovisno o položaju diode i slova odrađen je u besplatnom programskom alatu Inkscape. Slova su pozicionirana (Sl. 3.3) tako da se njihov centar poklopi sa diodom kako bi se ravnomjerno osvijetlila.

Sl. 3.3 Inkscape i korištenje vodilica za pozicioniranje slova

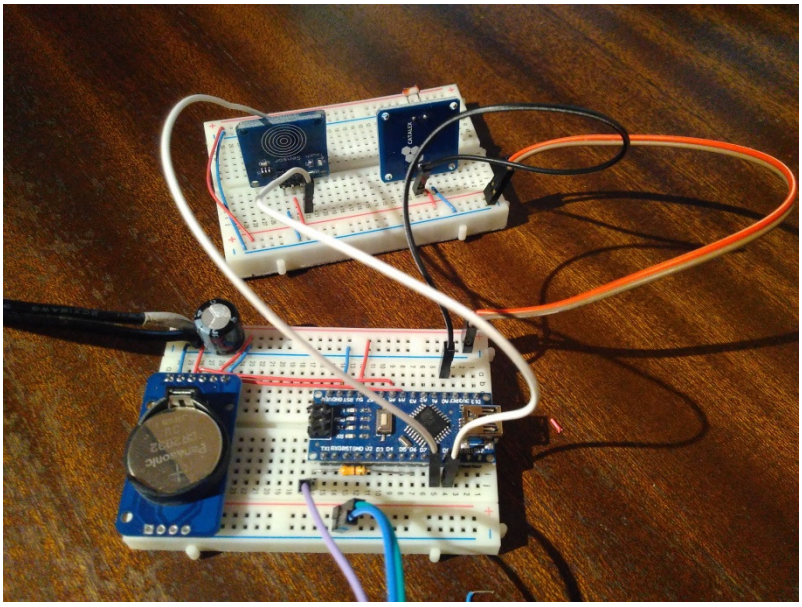
3.2 Montažna shema

Prilikom ugradnje hardvera u kućište moralo se odrediti gdje bi bilo najbolje provesti kabel napajanja kako bi najmanje vizualno kvario dojam sata. Odlučeno je, zbog mjesta na kojemu će maketa stajati, da će to biti donji lijevi kut makete, no nije nužno potreban taj položaj. Montažna shema (Sl.3.4) napravljena je u programu Fritzing i samo je simbolična te se može razlikovati od stvarnih modula te načina spajanja. Na shemi su vidljivi vodovi koji vode do LED-ica.



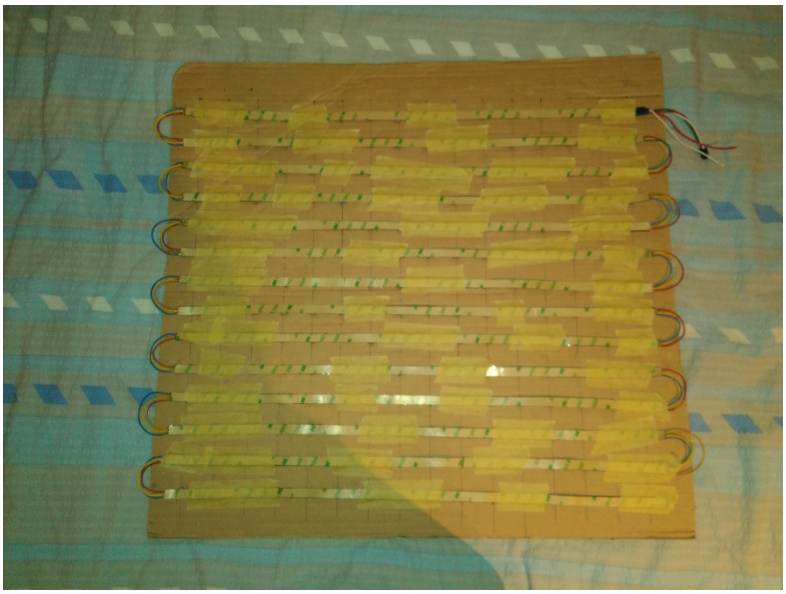
Sl. 3.4 Montažna shema na prototipnoj pločici

Kako je na slici 3.4 vidljivo, montažna shema nije kompliciranа te ne sadrži puno elemenata što olakšava samoizradu (Sl. 3.5) ovakvog sata.



Sl. 3.5 Samoizrađena maketa po uzoru montažne sheme

Shema lemljenja samih LED traka je jednostavna i sastoji se od 14 međusobno zalemljenih traka koje tvore *cik-cak* uzorak (Sl. 3.6).



Sl. 3.6 LED trake lemljene u cik-cak uzorak

3.3 Program

Program (prilog A) pisan za ovaj projekt sastoji se od nekoliko glavnih cjelina. Kao i svaki Arduino kod, osnova su dvije obvezne funkcije setup() i loop(). Osim njih tu se nalaze i konstantna dvodimenzionalna polja koja sadrže redne brojeve dioda za svaku pojedinu riječ, unaprijed definirane boje, te razne funkcije koje se pozivaju prilikom izvođenja.

Biblioteka koja se koristi za kontrolu LED-ica naziva se Neopixel i njeno korištenje se deklarira korištenjem naredbe *#include <Adafruit_NeoPixel.h>*. Za korištenje ove biblioteke, prvo se mora kreirati objekt klase Adafruit_neopixel na način: *Adafruit_NeoPixel traka = Adafruit_NeoPixel(suma_led, ledpin, NEO_GRB + NEO_KHZ800);* gdje parametar suma_led predstavlja ukupan broj LED-ica u nizu, ledpin predstavlja pin koji se koristi za slanje podataka iz Arduina, NEO_GRB + NEO_KHZ800 predstavljaju podvrstu dioda koje se koriste. Da bi inicijalizacija bila kompletna, potrebno je u setup() dijelu programa pokrenuti traku pozivom funkcije *traka.begin();*

RTC modul također ima propisanu deklaraciju koja se koristi u originalnom obliku jer nam nisu potrebne nikakve izmijenjene funkcionalnosti. Potrebno je stvoriti RTC objekt i pozivom njegovih funkcija se iščitavaju vrijeme, datum i temperatura.

U loop() funkciji se obavlja svo potrebno računanje, očitavanje i pozivanje drugih funkcija po potrebi. Osnovno načelo rada je obrisati ekran od prethodnih znakova, pročitati trenutno vrijeme sa RTC modula, odrediti željenu boju ovisno o tome radi li se o poslijepodnevnim ili prijepodnevnim satima (12 satni način prikaza), pozvati funkcije za ispis vrijednosti sati, minuta i pripadnih riječi proslijedivši im trenutno vrijeme i odabranu boju.

Brisanje cijelog ekrana je moguće jednostavnom funkcijom *clean_strip()* jer se promjene na ekranu ne događaju do poziva funkcije *traka.show()* te tako ne dolazi do treperenja ni problema sa višestrukim prikazom.

Funkcija *prikazi_sat()* kao argumente dobiva unaprijed odabranu boju i vrijednost sa RTC modula. Nakon što pročita vrijednost sata, jednostavnim if grananjem se 24 satni režim prebacuje na 12 satni režim prikaza i zatim se poziva funkcija ispis2d() i kao argumenti joj se šalju polje iz koje čita brojeve LED-ica, prvu koordinatu 2D polja i trenutna boja.

Funkcija *prikazi_minute()* funkcionira slično kao i funkcija za prikazivanje sata, samo što minute dijele na prvu i drugu znamenku i onda se sukladno tome prva i druga znamenka zasebno prikazuju. Ovaj način je izabran kako bi se jednostavnije odredila kombinacija znamenki tj. riječi koja se mora ispisati.

Funkcija *prikazi_rijec()* služi za prikaz prikladnih riječi na satu ovisno o trenutnim vrijednostima sata i minuta (npr. jedan SAT, dva SATA ili jedna MINUTA, dvije MINUTE). Temelji se na jednostavnim if grananjima sa uvjetima koji određene vrijednosti sati i minuta prikazuju sa pripadajućim riječima.

Funkcija *ispis2d()* koja kao argumente prima 2d polje, prvu koordinatu polja i trenutnu boju prikaza služi za ispis pojedinih riječi na ekran. Ovu funkciju pozivaju sve druge funkcije kada žele ispisati riječ na ekran. Temelji sa jednoj for petlji koja služi za čitanje vrijednosti polja

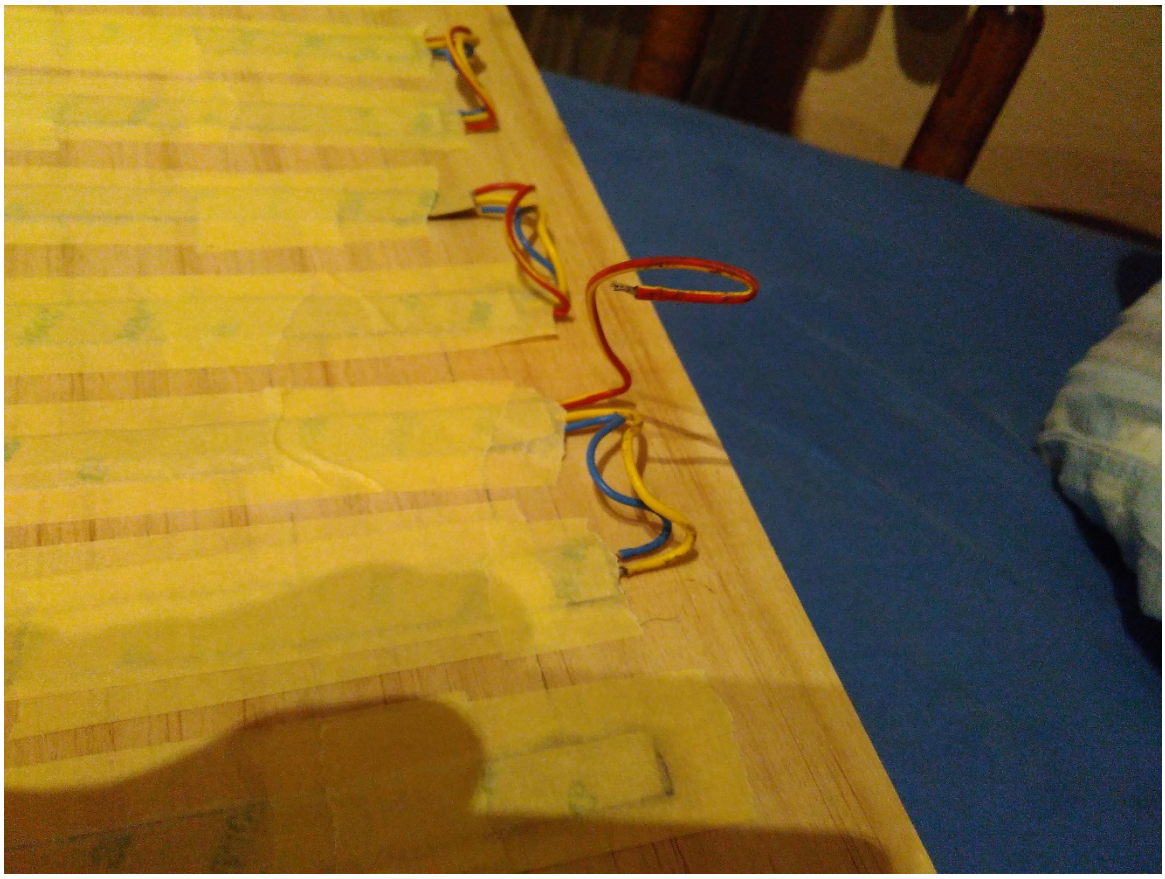
i if grananju koje petlju prekida ako dođe do kraja retka. Za prekid se koristi zastavica u obliku broja 255 jer se on nigdje drugdje ne koristi. Ukoliko se uvjet if grananja ne ispuni, prelazi se na ispis trenutnog piksela na ekran i for petlja prelazi na slijedeći element dok se ne ispuni uvjet prekida petlje.

3.4 Povezivanje dijelova i provjera ispravnosti

Zadnji korak izrade je samo sklapanje makete i testiranje rada makete kao cjeline. Postupak kreće umetanjem paus papira (sl. 3.7) i spajanjem nosača sa staklom uz pomoć ljepila za staklo. Nakon toga se sa druge strane nosača pomoću ljepljive trake postavljaju LED-ice (sl. 3.8) i vrši kontrola spojeva jer prilikom pomicanja žice znaju otpasti (sl. 3.9) sa LED trake.

Sl. 3.7 Paus papir i staklo spremno za lijepljenje

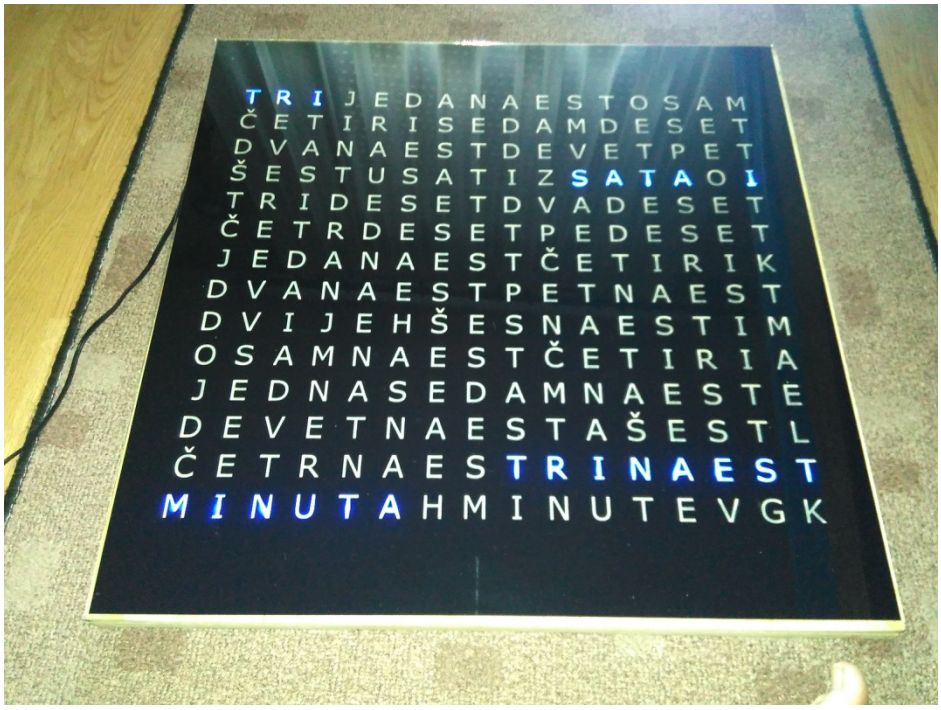
Sl. 3.8 LED-ice pričvršćene ljepljivom trakom



Sl. 3.9 Otpala žica zbog pomicanja

Nakon kontrole i eventualnog popravka, u zadnji poklopac naljepljena je prototipna pločica sa elektroničkim komponentama. Na kraju, pomoću šest vijaka povezan je zadnji poklopac sa nosačem (sl. 3.10) i maketa je upaljena kako bi se ispitala ispravnost gotove makete (sl. 3.11).

Sl. 3.10 Sklapanje poklopca sa nosačem



Sl. 3.11 Maketa u radu

4. ZAKLJUČAK

U ovom radu izrađena je maketa sata s riječima pomoću WS2812b LED-ica. Sustav se sastoji od 3 glavne cjeline: samog hardvera, nosača i stakla sa slovima te od programskog rješenja koje spaja prve dvije cjeline. Prednosti rješenja s WS2812b LED-icama je olakšana izrada sa većim mogućnostima kao što su animacije, višebojni prikaz i, zahvaljujući velikoj zajednici iza Arduino projekta koji kontrolira ove diode, stalne novosti u mogućnostima i idejama za izradu i izmjenu projekta. Mane sustava su veća cijena LED-ica, te ograničenost na unaprijed zadane dimenzije LED traka. Moguća poboljšanja rješenja su u izradi dodatne tiskane pločice koja bi povezala hardver u čvrstu cjelinu, dodavanje novih funkcionalnosti programa kao što je prikaz temperature ili datuma.

LITERATURA

[1] Tutorial za WS2812b LED-ice i neopixel biblioteku, 27.06.2016
https://learn.adafruit.com/adafruit-neopixel-uberguide/overview

[2]Arduino Nano mikroupravljač, 27.06.2016
https://www.arduino.cc/en/Main/ArduinoBoardNano

[3]Opis i objašnjenje I2C protokola, 27.06.2016.
https://learn.sparkfun.com/tutorials/i2c

[4]Potrebne modifikacije za RTC modul, 27.06.2016.
http://woodsgood.ca/projects/2014/10/21/the-right-rtc-battery/

SAŽETAK

Naslov: Sat s riječima pomoću niza upravljivih dioda

U ovom radu izrađen je pojednostavljeni sat s riječima pomoću niza LED-ica. Sat se temelji na trakama WS2812b digitalno upravljivih LED-ica i Arduino Nano mikroupravljaču. Kroz rad su objašnjene tehnologije i alati koji se koriste. Nakon toga je objašnjen postupak same izrade sata sa svim pripadajućim dijelovima. Na kraju je objašnjen tok programa koji prikazuje odgovarajuće riječi.

Ključne riječi: sat s riječima, ws2812b, Arduino Nano, Arduino IDE,

ABSTRACT

Title: Word clok with a a string of controllable LED's

In this thesis a simplified word clock with a string of controllable LED's was made. Clock is based around strips of WS2812b digitaly controllable LED's and Arduino Nano microcontroller. Through the work technology and the tools used are explained. Next the process of making a clock is explained. At the end, the program logic controlling the clock is explained.

Keywords: word clock, ws2812b, Arduino Nano, Arduino IDE

ŽIVOTOPIS

Ivan Veselin rođen je u Požegi 15.1.1995. godine. Od rođenja živi u mjestu Radovanci nadomak Požege. Pohađao je osnovnu školu Ivan Goran Kovačić u Velikoj gdje se prvi puta susreće sa informatikom i računalima. Zainteresiranost za područje računalnih znanosti iskazuje kada upisuje Tehničku školu u Požegi, smjer Tehničar za računalstvo. Osim odličnog uspjeha na nastavi, treba spomenuti i odlazak na državno natjecanje iz osnova elektrotehnike u Zagreb. 2013. godine u redovnom roku završava školovanje u srednjoj školi te upisuje preddiplomski studij na tadašnjem Elektrotehničkom fakultetu u Osijeku, smjer Računarstvo.

_____

Ivan Veselin

PRILOG A: Program za Arduino Nano razvojni sustav

```
#include <Wire.h>
#include <RtcDS3231.h>
#include <Adafruit_NeoPixel.h>
#include <Adafruit_GFX.h>
#include <Adafruit_NeoMatrix.h>


#define ledpin 6
#define suma_led 224

//postavljanje rtc modula
RtcDS3231 Rtc;
//postavljanje trake
Adafruit_NeoPixel traka = Adafruit_NeoPixel(suma_led, ledpin, NEO_GRB + NEO_KHZ800);

//postavljanje ekrana za animacije
Adafruit_NeoMatrix matrix = Adafruit_NeoMatrix(16, 14, ledpin,
                            NEO_MATRIX_TOP     + NEO_MATRIX_LEFT +
                            NEO_MATRIX_ROWS + NEO_MATRIX_ZIGZAG,
                            NEO_GRB            + NEO_KHZ800);


//boje
uint32_t bijela = traka.Color(127, 127, 127);
uint32_t crna = traka.Color(0, 0, 0);
uint32_t crvena = traka.Color(127, 0, 0);
uint32_t zelena = traka.Color(0, 127, 0);
uint32_t plava = traka.Color(0, 0, 127);

//kodovi riječi:      255 zastavica za kraj riječi
//sati
byte sat2d[][11] = {
  {3, 4, 5, 6, 7, 255}, //1,2,3...
  {32, 33, 34, 255},
  {0, 1, 2, 255},
  {26, 27, 28, 29, 30, 31, 255},
  {45, 46, 47, 255},
  {60, 61, 62, 63, 255},
  {21, 22, 23, 24, 25, 255},
  {12, 13, 14, 15, 255},
  {40, 41, 42, 43, 44, 255},
  {16, 17, 18, 19, 20, 255},
  {3, 4, 5, 6, 7, 8, 9, 10, 11, 255},
  {32, 33, 34, 35, 36, 37, 38, 39, 255}
};

//minute
byte minute2d[][11] = {
  {160, 161, 162, 163, 164, 255},                 //1
  {128, 129, 130, 131, 132, 255},                 //2
  {200, 201, 202, 255},                           //3
  {105, 106, 107, 108, 109, 110, 255},            //4
  {117, 118, 119, 255},                           //5
  {177, 178, 179, 180, 255},                      //6
  {165, 166, 167, 168, 169, 255},                 //7
  {156, 157, 158, 159, 255},                      //8
  {187, 188, 189, 190, 191, 255},                 //9
```

```
  {67, 68, 69, 70, 71, 255}                    //10


};
byte jedanaestice[][11] = {
  {96, 97, 98, 99, 100, 101, 102, 103, 104, 255},          //11
  {120, 121, 122, 123, 124, 125, 126, 127, 255},           //12
  {200, 201, 202, 203, 204, 205, 206, 207, 255},           //13
  {192, 193, 194, 195, 196, 197, 198, 199, 200, 255},      //14
  {112, 113, 114, 115, 116, 117, 118, 119, 255},           //15
  {134, 135, 136, 137, 138, 139, 140, 141, 255},           //16
  {165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 255}, //17
  {151, 152, 153, 154, 155, 156, 157, 158, 159, 255},      //18
  {182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 255}  //19
};
byte desetice2d[][11] = {
  {72, 73, 74, 75, 76, 77, 78, 79, 255},                   //20
  {64, 65, 66, 67, 68, 69, 70, 71, 255},                   //30
  {87, 88, 89, 90, 91, 92, 93, 94, 95, 255},               //40
  {80, 81, 82, 83, 84, 85, 86, 255}                        //50
};
//riječi
#define minuta 3
#define minute 4
#define sati 0
#define sata 1
#define sat 2

byte rijec[][11] = {
  {55, 56, 57, 58, 255},              //sati    1
  {50, 51, 52, 53, 255},              //sata    2
  {56, 57, 58, 255},                  //sat     3
  {218, 219, 220, 221, 222, 223, 255}, //minuta  4
  {211, 212, 213, 214, 215, 216, 255}  //minute  5
};

void setup ()
{
  traka.begin();
  matrix.begin();
  traka.show();
  traka.setBrightness(50);
  rainbowCycle(5);

  //--------RTC SETUP ------------
  Rtc.Begin();
  RtcDateTime compiled = RtcDateTime(__DATE__, __TIME__);

  if (!Rtc.IsDateTimeValid())
  {

    Rtc.SetDateTime(compiled);
  }

  if (!Rtc.GetIsRunning())
  {

    Rtc.SetIsRunning(true);
  }
```

```
  RtcDateTime now = Rtc.GetDateTime();
  if (now < compiled)
  {

    Rtc.SetDateTime(compiled);
  }
  else if (now > compiled)
  {

  }
  else if (now == compiled)
  {

  }

  //gašenje nepotrebnih stavki na RTC modulu
  Rtc.Enable32kHzPin(false);
  Rtc.SetSquareWavePin(DS3231SquareWavePin_ModeNone);
}

void loop () {
  clean_strip();
  uint32_t tr_boja;
  RtcDateTime now = Rtc.GetDateTime();
  int sensorValue = analogRead(A0);
  if (now.Hour() <= 12) {
    tr_boja = crvena;
  } else {
    tr_boja = plava;
  }
  float Rsensor;
  Rsensor = (float)(1023 - sensorValue) * 10 / sensorValue;
  if (Rsensor < 20) {
    traka.setBrightness(255);
  } else if (Rsensor < 200) {
    traka.setBrightness(127);
  } else traka.setBrightness(64);
  prikazi_sat(now, tr_boja);
  prikazi_rijec(now, tr_boja);
  prikazi_minute(now, tr_boja);
  traka.show();
  delay(1000); // 1 sec

}


//funkcija za ispis riječi, 255 zastavica za kraj riječi
void ispis2d(byte broj[][11], byte a,  uint32_t boja) {

  for (byte i = 0; i < 12; i++) {
    if (broj[a][i] == 255) {
      break;
    } else {
      traka.setPixelColor(broj[a][i], boja);
    }
  }
}
```

```
//dekoder minute=>riječi
void prikazi_minute(const RtcDateTime& dt, uint32_t tr_boja) {
  byte tr_min = (byte)dt.Minute();
  byte prva_z = tr_min / 10;
  byte druga_z = tr_min % 10;

  if (tr_min <= 10) {
    ispis2d(minute2d, tr_min - 1, tr_boja);
  } else if (tr_min < 20) {
    ispis2d(jedanaestice, druga_z - 1, tr_boja);
  } else if (druga_z == 0) {
    ispis2d(desetice2d, prva_z - 2, tr_boja);
  } else {
    ispis2d(desetice2d, prva_z - 2, tr_boja);
    ispis2d(minute2d, druga_z - 1, tr_boja);
  }
}


//dekoder za odgovarajuće riječi
void prikazi_rijec(const RtcDateTime& dt, uint32_t tr_boja) {
  byte satbr;
  if (dt.Hour() <= 12) {
    satbr = (byte)dt.Hour();
  } else {
    satbr = (byte)dt.Hour() - 12;
  }
  if (satbr == 1) {
    // ispisati sat
    ispis2d(rijec, sat, tr_boja);
  } else if (satbr == 2 || satbr == 3 || satbr == 4) {
    //ispisati sata
    ispis2d(rijec, sata, tr_boja);
  } else {
    //ispisati sati
    ispis2d(rijec, sati, tr_boja);
  }
  byte tr_min = (byte)dt.Minute();
  byte prva_z = tr_min / 10;
  byte druga_z = tr_min % 10;
  if (tr_min != 0) {
    //slovo I
    traka.setPixelColor(48, tr_boja);
    if (prva_z != 1 && (druga_z == 2 || druga_z == 3  || druga_z == 4)) {
      ispis2d(rijec, minute, tr_boja);
    } else ispis2d(rijec, minuta, tr_boja);
  }
}


//dekoder sati=>riječi
void prikazi_sat(const RtcDateTime& dt, uint32_t tr_boja) {
  byte satbr;
  if (dt.Hour() <= 12) {
    satbr = (byte)dt.Hour();
  } else {
    satbr = (byte)dt.Hour() - 12; //pomak na 12 satni režim sata
  }
  ispis2d (sat2d, satbr - 1, tr_boja); //satbr-1 jer polje ide od 0
```

```
}


//funkcija za brisanje trake da ne dolazi do pokazivanja prošlih riječi
void clean_strip() {
  for (byte i = 0; i < 224; i++) traka.setPixelColor(i, crna);
}


//funkcije za animaciju na početku rada
uint32_t Wheel(byte WheelPos) {
  WheelPos = 255 - WheelPos;
  if (WheelPos < 85) {
    return traka.Color(255 - WheelPos * 3, 0, WheelPos * 3);
  }
  if (WheelPos < 170) {
    WheelPos -= 85;
    return traka.Color(0, WheelPos * 3, 255 - WheelPos * 3);
  }
  WheelPos -= 170;
  return traka.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
}

void rainbowCycle(uint8_t wait) {
  uint16_t i, j;

  for (j = 0; j < 256 * 5; j++) { // 5 cycles of all colors on wheel
    for (i = 0; i < traka.numPixels(); i++) {
      traka.setPixelColor(i, Wheel(((i * 256 / traka.numPixels()) + j) & 255));
    }
    traka.show();
    delay(wait);
  }
}
```