Programirani svijetleći 3D križić - kružić

Orlandini, Mladen

Istarsko veleučilište
Università Istriana
di scienze applicate

Mladen Orlandini

PROGRAMIRANI SVIJETLEĆI 3D KRIŽIĆ-KRUŽIĆ

Završni rad

Pula, 2020.

Istarsko veleučilište
Università Istriana
di scienze applicate

Mladen Orlandini

PROGRAMIRANI SVIJETLEĆI 3D KRIŽIĆ-KRUŽIĆ

Završni rad

JMBAG: 0233006392 , redoviti student

Studijski smjer: Preddiplomski stručni studij politehnike

Predmet: Elektronika 2

Mentor: Sanja Grbac Babić, mag. računarstva, viši predavač

Pula, rujan 2020.

IZJAVA O SAMOSTALNOSTI IZRADE ZAVRŠNOG RADA

Ovom izjavom potvrđujem da sam završni rad pod nazivom “Programirani svijetleći 3D križić-kružić” napisao samostalno uz pomoć mentorice Sanja Grbac Babić v.pred., primjenjujući znanje stečeno tijekom studiranja te stručnu literaturu koja je navedena na kraju rada. Završni rad je napisan u duhu hrvatskog jezika.

Student: *Mladen Orlandini*

Potpis: _____

SAŽETAK

Ovim je radom izrađeno sklopovlje za trodimenzionalnu igru križić-kružić u elektronskom obliku, te implementirano programsko rješenje korištenjem Arduino platfome. Povezivanjem hardvera i softvera dobivena je kompleksna i dinamična igra koja se najčešće igra u dvoje.

KLJUČNE RIJEČI

3D križić-kružić, Arduino, Qubic

ABSTRACT ispraviti prijevod

This thesis was used to create the hardware for a three-dimensional game of tic-tac-toe which was made in electronic form, as well as the implementation of a software solution by using the Arduino platform. By connecting hardware and software, a complex and dynamic game has been obtained, which is most often played in pairs.

KEY WORDS

3D tic-tac-toe, Arduino, Qubic

Sadržaj

POPIS OZNAKA I KRATICA

Kratice:

OZNAKA	OPIS	JEDINICA
3D	Trodimenzionalno	-
NPN	Negative Positive Negative	-
LCD	Liquid Crystal Display	-
LED	Light Emitting Diode	-
IDE	Integrated Development Environment	-
ICSP	In Circuit Serial Programming	-
USB	Universal Serial Bus	-
I/O	Input/Output	-
AC	Alternating Current	-
DC	Direct Current	-
UART	Universal Asynchronous Receiver - Transmitter	-
PWM	Pulse - Width Modulation	-
RGB	Red Green Blue	-

Oznake:

P	Snaga	W (wat)
U	Napon	V (vlot)
I	Struja	A (amper)
R	Otpor	Ω (ohm)

POPIS SLIKA

1. UVOD

Trodimenzionalni ili 3D križić-kružić (eng. tic-tac-toe), poznatiji pod imenom Qubic, je apstraktna, strateška i matematička igra za najčešće dva igrača. Koncept same igre je sličan tradicionalnom križić-kružiću koji se igra na papiru, ali se Qubic, za razliku od uobičajenog križić-kružića koji se igra u dvodimenzionalnom prostoru, igra u kubičnom dizajnu koji je obično dimenzija 4x4x4. [20]

Igru Qubic su u više navrata objavili razni izdavači pod različitim imenima te je igra pripisana raznim dizajnerima. Tako su kroz godine mjenjani dizajni, materijali od kojih je igra napravljena te načini igranja (dvodimenzionalno - na papiru, trodimenzionalno te igra na računalu). [2]

U ovom radu prikazati će se način izrade trodimenzionalnog križić-kružića sačinjenog od LED dioda te će se na taj način pokušati spojiti više različitih verzija križić-kružića s kojima smo se susretali kroz povijest.

S obzirom na to da je 3D križić-kružić kompleksniji od svojih prethodnih verzija, što zbog komponenti koje su uključene u ovaj projekt, što zbog programskog koda bez kojega ova verzija ne bi mogla funkcionirati kao takva, porebno je napomenuti još barem jednu razliku između ove i prethodnih verzija igre, a to je puno veći broj mogućih kombinacija koji se povećava s 8 u uobičajenoj 3x3 verziji na 49 mogućih kombinacija u 3x3x3 verziji.

Isto tako potrebno je napomenuti da postoji pravilo kojeg se igrači najčešće pridržavaju prilikom igranja križić-kružića u 3x3x3 verziji. To je pravilo koje zabranjuje zauzimanje središnje točke u kocki kao prvi potez bilo kojeg igrača iz razloga što će onaj igrač koji u prvom potezu odabere središnju točku odmah pobjediti te je iz tog razloga 4x4x4 preferirana verzija igre.

1.1. Definicija problema

Za izradu 3D križić-kružića potrebno je poznavanje osnova elektrotehnike i elektronike, zbog susretanja s nekim osnovnim komponentama, te osnovama računalnog programiranja, zbog toga što je potrebno programirati mikroprocesor koji će omogućiti upravljanje LED kockom preko tipkala. Okvir kocke se konstruira na način da se lemljenjem nožica (priključnica) LED dioda napravi okvir dimenzija 3x3x3. Diode su spojene na odgovarajući način tako da se na krajevima dioda spajaju vodljive žice koje idu u mikrokontroler. Na mikrokontroleru je unesen program, spojen je LCD ekran koji prikazuje koji je igrač na redu te tipkala kojima se upravljaju LED diode i odabire koja će od njih svijetliti u LED kocki. Samo poznavanje dijelova te njihovih svojstava je veoma bitno jer u slučaju lošeg spoja ili odabira sama kocka ne bi funkcionirala. Iz tog razloga bitno je dobro proučiti shemu izrade.

1.2. Cilj i svrha rada

Cilj ovog završnog rada je prikazati kako bi igra križić-kružić izgledala u trodimenzionalnom prostoru umjesto u dvodimenzionalnom te prikazati da čak i najjednostavnija igra može zahtjevati dobro poznavanje električnih komponenti i programiranja mikrokontrolera.

1.3. Hipoteza

Izrada ovog završnog rada zahtijeva dobro poznavanje elektroničkih komponenti i programiranja jer bi bilo nemoguće izraditi 3D križić-kružić bez poznavanja komponenata i programskog jezika. Povezivanjem sklopovlja i programiramjem mikrokontrolera moguće je napraviti igru križić-kružić u trodimenzionalnom I dinamičnom obliku.

1.4. Metode rada

U ovom radu korištene su metode analize i sinteze. Pomoću metode analize je složeni problem raspodijeljen na jednostavnije dijelove koje je bilo lakše realizirati. Pomoću metode sinteze pojedinačna rješenja u konačnici su povezana u jednu cjelinu, odnosno rješenje problema.

1.5. Struktura rada

Završni rad sastoji se od 7 poglavlja. Prvi dio rada nas uvodi u rad. Drugi dio je povijest igre križić – kružić te na koje načine se može igra može igrati. Treći dio obuhvaća komponente koje su nam potrebne za izradu rada te opise pojedinih komponenti. U četvrtom dijelu prikazana je shema spoja komponenti te način na koji su te komponente povezane. U petom dijelu prikazan je postupak izrade u koji su uključene slike uz opise spajanja komponenti. U šestom dijelu opisan je programski kod i njegove funkcije. Sedmi dio obuhvaća zaključak, popis literature, izvora te priloge koji su korišteni za izradu ovog završnog rada. Cjeloviti programski kod sastavni je dio priloga ovog rada.

2. POVIJEST

Društvena igra križić-kružić ili, kako je na engleskom jeziku nazivaju, tic-tac-toe je kroz povijest imala mnogo različitih verzija i dizajna koji su pripisivani raznim izdavačima. Prema knjizi “Tic Tac Toe: And Other Three-In-A-Row Games from Ancient Egypt to the Modern Computer”, autorice C. Zaslavsky, neka vrsta ove igre postoji još od drevnog Egipta. Naime, svoje ime i današnje značenje je dobila puno kasnije. Ime “tic-tac-toe” je izvedeno od zvuka “tick-tack”, imena za stariju verziju Backgammona prvi put opisanu 1558. godine. Prvo spominjanje igre pod nazivom “tick-tack-toe” bilo je 1884. godine, ali se odnosilo na dječju igru potpuno drugačiju od igre koja danas nosi taj naziv. Pojam “tic-tac-toe” se kao naziv današnje igre pojavio tek u 20. stoljeću. [5]

Kroz povijest je nastalo mnogo društvenih igara koje su imale zajedničko slaganje ili nizanje kuglica ili diskova horizontalno, vertikalno i dijagonalno. Neki od najranijih dokaza za 3D igru nizanja kuglica je primjerak patenta Theodore R. Duncana iz 1946. godine. Ideja je bila da se kuglice spuštaju jedna na drugu umjesto da ih se samo stavlja na ploču. Može se reći da su iz tog patenta nastale ostale slične igre poput Score Four (Funtastic, 1968. godine), gdje su se na štapiće na ploči 4x4 nizale kuglice i dvodimenzionalna igra Connect 4 (Hasbro). [12]

Najbolji poznati primjer takve igre bio je Qubic, koji je prvi puta proizvela tvrtka Duplicon 1964. godine. Tvrtka Parker Brothers počela ju je prozvoditi i stavljati na tržište 1964. godine, a ponovno je izdana 1972. godine u modernijem dizajnu. Glavne razlike između dva izdanja su bile to što je, za razliku od prve verzije koja je najdonju ploču imala izrađenu od neprozirne plastike, a gornje tri od prozirne, druga verzija imala sve četiri ploče izrađene od prozirne plastike te zaobljene rubove. [20]

Poput tradicionalnog križić-kružića postoji i varijacija na trodimenzionalnu verziju koja se igra na papiru. Tvrtka 3M Games prodavala ju je u kompletu s ostalim društvenim igrama koje su se igrale na papiru 1970. godine te su kupci igre dobivali blokić s pedeset listova papira na kojima su bile isprintane slike ploča za 3D križić-kružić. [21]

Uz verzije na papiru i one na plastičnim pločama, napisano je i nekoliko računalnih programa te videoigara. [22]

Postoji nekoliko računalnih programa u kojima igrač igra protiv računala, a prvi program je napisao William Daly Jr. kao dio svojeg rada na MIT-u[1]. Najranije konzole su se koristile svijetlima i prekidačima, tekstualnim terminalima i sličnim interakcijama u kojima bi igrač unosio svoje poteze brojevima (pr. koristeći ”2 3 1” za drugi nivo, treći red, prvi stupac) kako bi odabrao željeno polje. U program je bilo uračunato predviđanje dvanaest slijedećih poteza i spremala se povijest svih prethodnih igara sa svim protivnicima kako bi se prema njihovim prošlim odabirima mogla prilagoditi strategija računalnog programa.

Godine 1978. Atari je izdao grafičku verziju igre za Atari 2600 konzolu i Atari 8-bitna računala. Za igru su se koristili standardni kontroleri te se mogla igrati u dvoje, igrač protiv igrača ili igrač protiv programa na jednoj od osam različitih težina. Godine 1990. igra je bila uključena u Microsoft Windows Entertainment Pack pod nazivom TicTactics te je 2010. godine postala dostupna za Xbox 360. [20]

[1] Massachusetts Institute of Technology

3. KOMPONETE ZA IZRADU

S obzirom na konačni izgled ovoga rada, za izradu rada je bilo potrebno nabaviti odgovarajuće komponente:

1. Arduino MEGA 2560 Rev3
2. LCD zaslon I2C 16x2
3. BDX53C NPN TO220 DAR 8A 100V Tip: BDX
4. Pločica Vetronit 200 x 300
5. Taster otvoreni CRNI za šasiju Tip: taster
6. Nosač za LED kocku
7. R 0,5 W 100 R Tip: 0,5 W
8. R 1 W 5,6 KR 1 W
9. LED diode
10. Kutija
11. Vanjsko napajanje

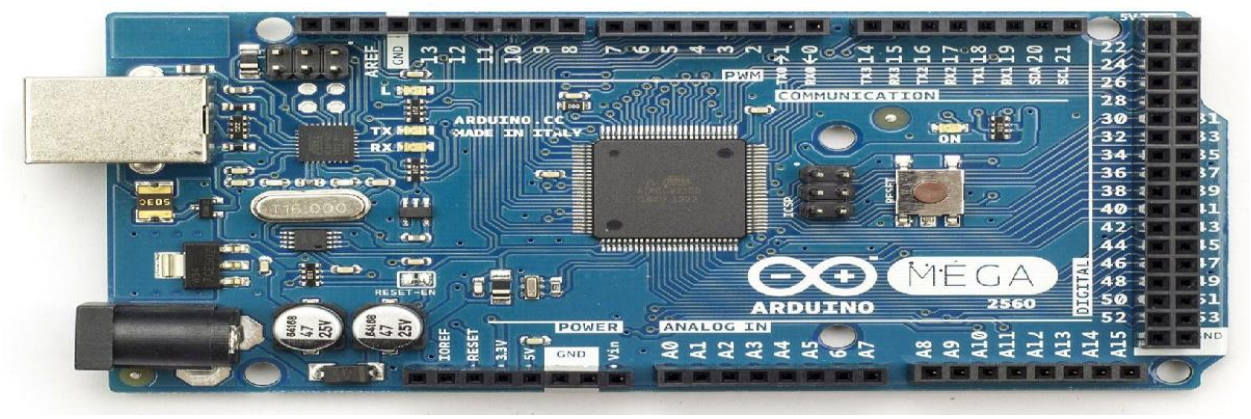
U nastavku ovog poglavlja slijedi opis korištenih komponenti.

3.1. Arduino Mega

Arduino je elektronička prototipna platforma namijenjena kreiranju elektroničkih projekata. Sastoji se od hardware-a, koji je zapravo fizički elektronički programibilni strujni krug (poznat i kao mikrokontroler), te softwarskog dijela koji se naziva IDE (Integrated Development Environment), kojega se pokreće na računalu te je iz njega moguće programiranje i upravljanje samom pločicom. Arduino se može koristiti za očitavanje senzora ili upravljanje uređajima poput motora ili svjetla. To omogućava nadogradnju programa na ploču koja može vršiti interakciju sa uređajem u stvarnom svijetu. S ovakvim neograničenim mogućnostima mogu se napraviti interesantni I korisni uređaji.

Arduino Mega 2560 (slika 1) je mikrokontroler napravljen na temelju ATmega2560. Ima 54 digitalne input/output igle (od kojih se 15 mogu koristiti kao PWM izlazi), 16 analognih inputa, 4 UART (hardverski serijski portovi), kristalni oscilator od 16 MHz, USB priključak, priključak za napajanje, ICSP zaglavlje i gumb za resetiranje. Sadrži sve što je potrebno za podršku mikrokontrolera, jednostavno se poveže s računalom USB kabelom ili napajanje napajanjem AC-to-DC adapterom ili baterijom. Arduino Mega 2560 kompatibilan je s većinom oklopa dizajniranih za Arduino Duemilanove ili Diecimila.

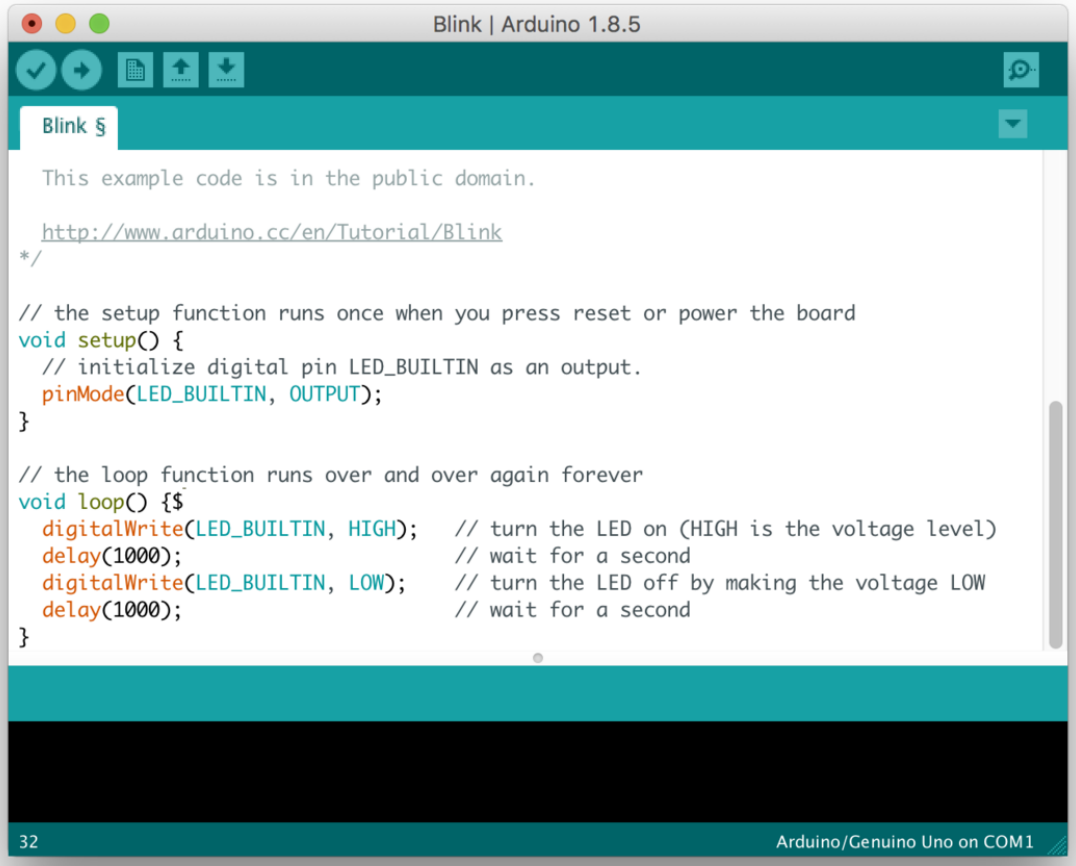
Radi količine komponenata i njihovih zahtjeva, Arduino Mega je idealan za ovaj rad. Ima dovoljno procesorske snage za rad, digitalnih I/O pinova koji su potrebni za spajanje i kontroliranje LED dioda. Da se koristi npr. Arduino Uno trebaloi bi koristiti više multiplexera, koji nam služe u slučaju kada ima više outputa od inputa.



Slika 1: Arduino Mega [3]

Programiranje Arduina se vrši u prilagođenoj verziji C++ programskog jezika.

Za pokretanje programa potrebno je prvo preuzeti na računalo Arduino IDE (slika 2). Arduino IDE se može preuzeti sa Arduino službene stranice (https://www.arduino.cc/en/Main/Software). Nakon što se program instalira na računalo preko kojega se piše kod, USB kabel iz Arduina se priključuje na USB utor na računalu, a računalo prepoznaje Arduino kao virtualni uređaj.



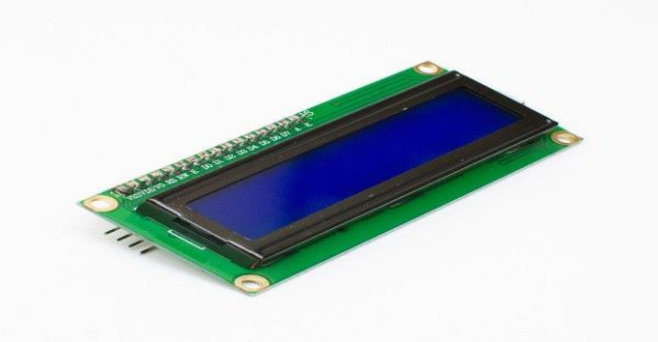
Slika 2: Arduino IDE [14]

Prije samog početka programiranja potrebno je odrediti tip ploče i serijski port.

Tip ploče se određuje tako da se prvo klikne na Tools te onda na Boards, gdje se odabere tip ploče koji se koristi.

3.2. LCD zaslon

LCD zaslon (slika 3) služi za prikazivanje stanja uređaja kada se upali, prikaz pobjednika te za prikaz igrača koji je slijedeći na redu za potez.

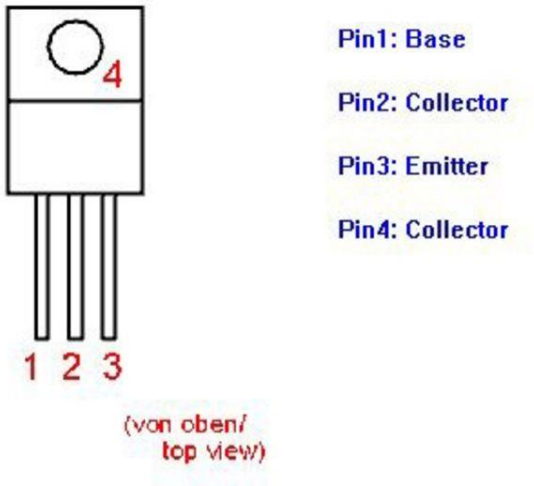


Slika 3: LCD zaslon [15]

3.3. Tranzistor

Tranzistor (slika 4) je poluvodički elektronički element koji se koristi za pojačavanje električnih signala, kao elektronička sklopka, za stabilizaciju napona i mnoge druge primjene. Osnovni je tvorni element mnogih elektroničkih sklopova, integriranih krugova i elektroničkih računala. Tranzistor postoji u dvijema vrstama i dijeli se prema načinu rada.

Postoje bipolarni te unipolarni tranzistor.

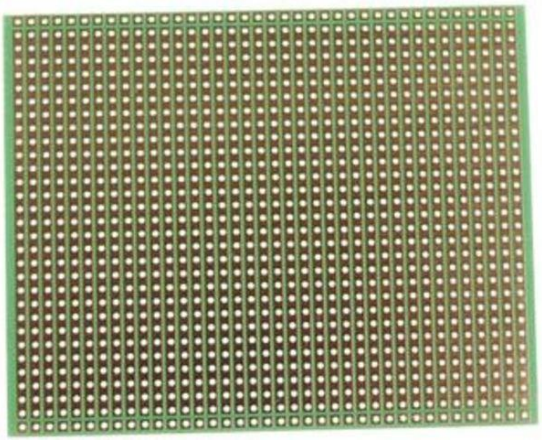


Slika 4: Tranzistor [16]

3.4. Stripboard (pločica)

Stripboard (slika 5) je naziv za vrste električnih prototipa ploča koje karakterizira pravokutni redoslijed rupa udaljen 2,54 mm. Rupe povezuju bakrene obloge koje se protežu sa jedne strane ploče na drugu. Ove ploče su jednostavan način povezivanja

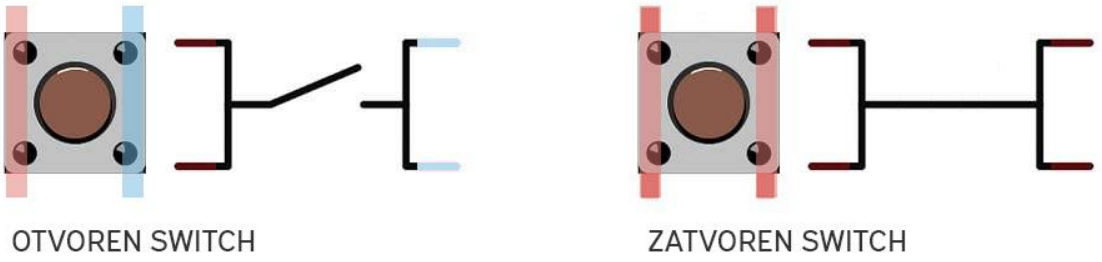
lakših ili srednje kompliciranih elektroničkih krugova. Prekidi na ploči su izrađeni okolo rupa kako bi podijelili trake u više električnih čvorova. Moguće je 10ctivat između rupa kako bi omogućili komponenti koja ima dva pin 10ctivat samo jedan položaj.



Slika 5: Stripboard (pločica) [17]

3.5. Tester – otvoreni (tipkalo)

Tipkalo ili tester (slika 6) je jedan od najčešćih komponenti korištenih u elektronici. Razlikuju se po otvorenom I zatvorenom 10ctiva (sklopka). Tipkala mogu služiti kao 10ctivator, prekidač ili brojač.

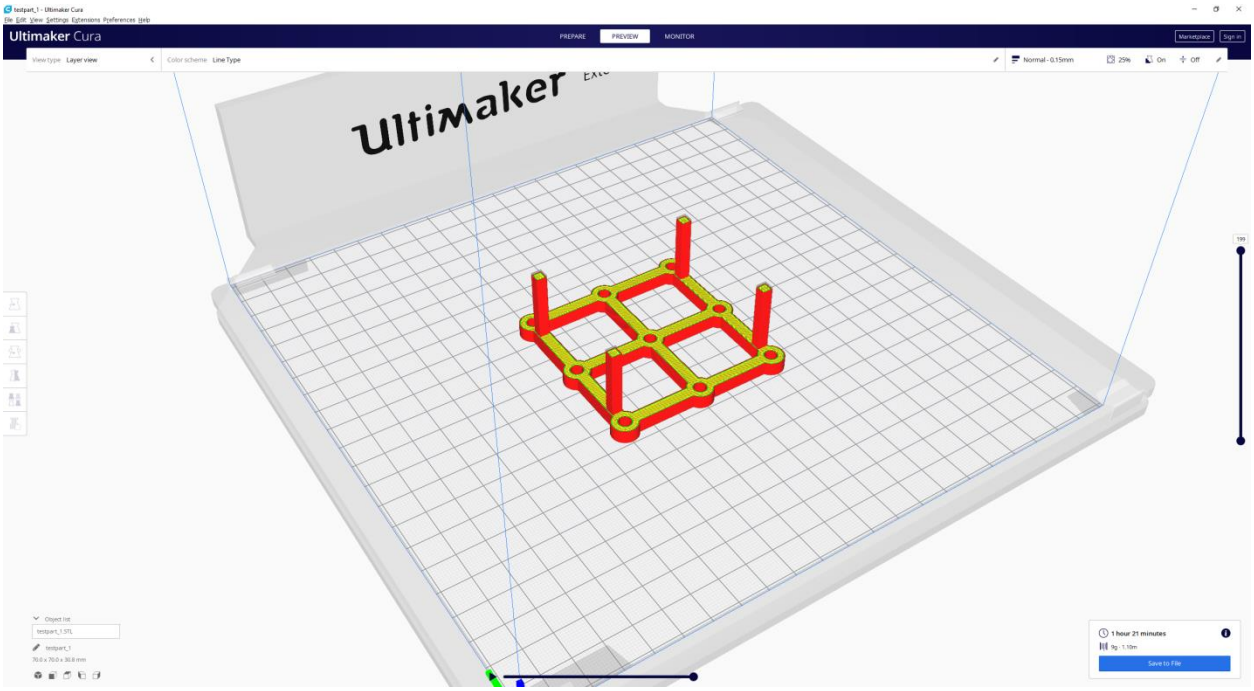


Slika 6: Tester (tipkalo) [18]

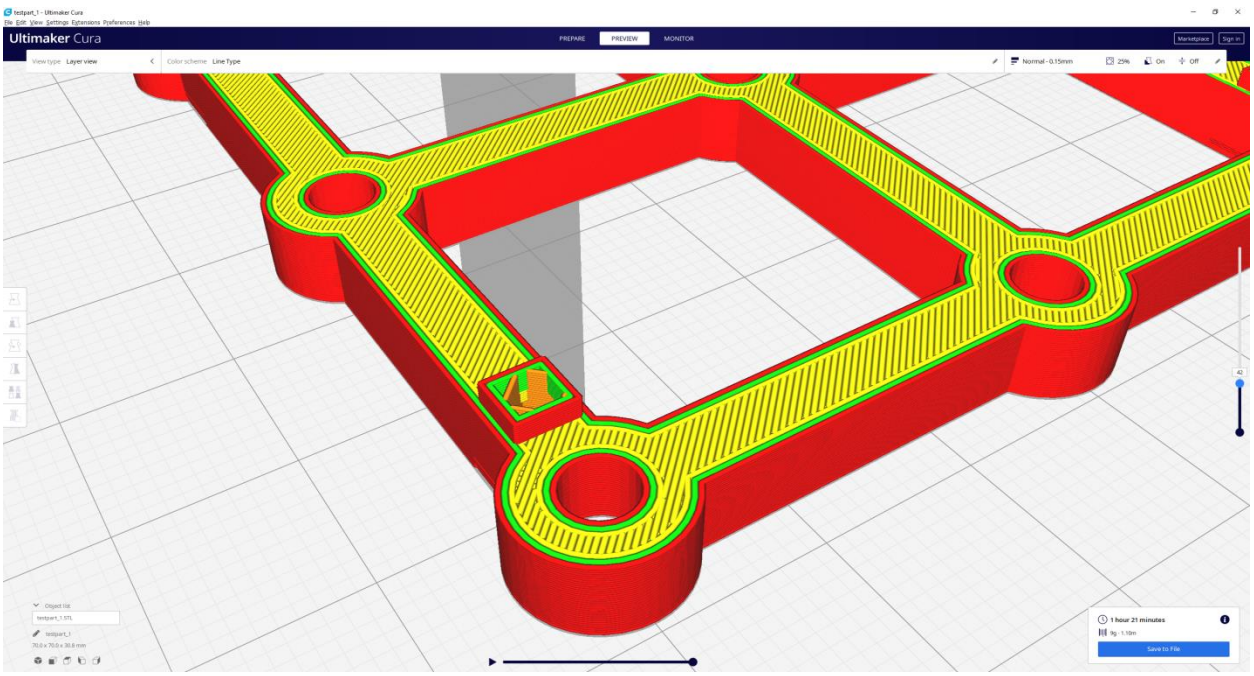
3.6. Nosač za LED kocku

Nosači za LED kocku (slike 7, 8 i 9) su dizajnirani u Solid works-u 2019. Trebalo je oko 2 sata dizajniranja te 6 sati printanja na 3D printeru “Ultimaker” koji se nalazi na Istarskom veleučilištu. Prilikom dizajniranja najbitnije stavke su bile da bude pregledno

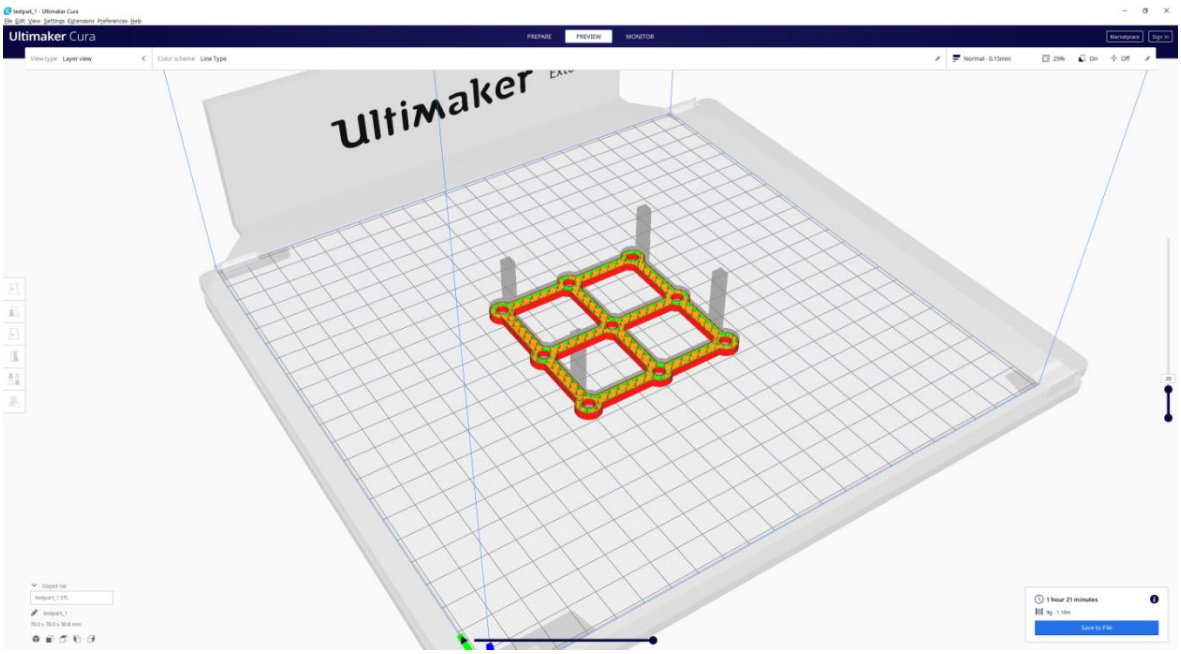
za igranje 3D križić-kružića tijekom svijetljenja LED dioda te da sam nosač izdrži težinu LED dioda i žica koje spajaju diode s Arduinom.



Slika 7: Prikaz dizajna nosača za LED kocku
[Izvor: Izradio autor]



Slika 8: Prikaz dizajna nosača za LED kocku – utori za LED diode
[Izvor: Izradio autor]



Slika 9: Prikaz dizajna nosača za LED kocku
[Izvor: Izradio autor]

3.7. Otpornici

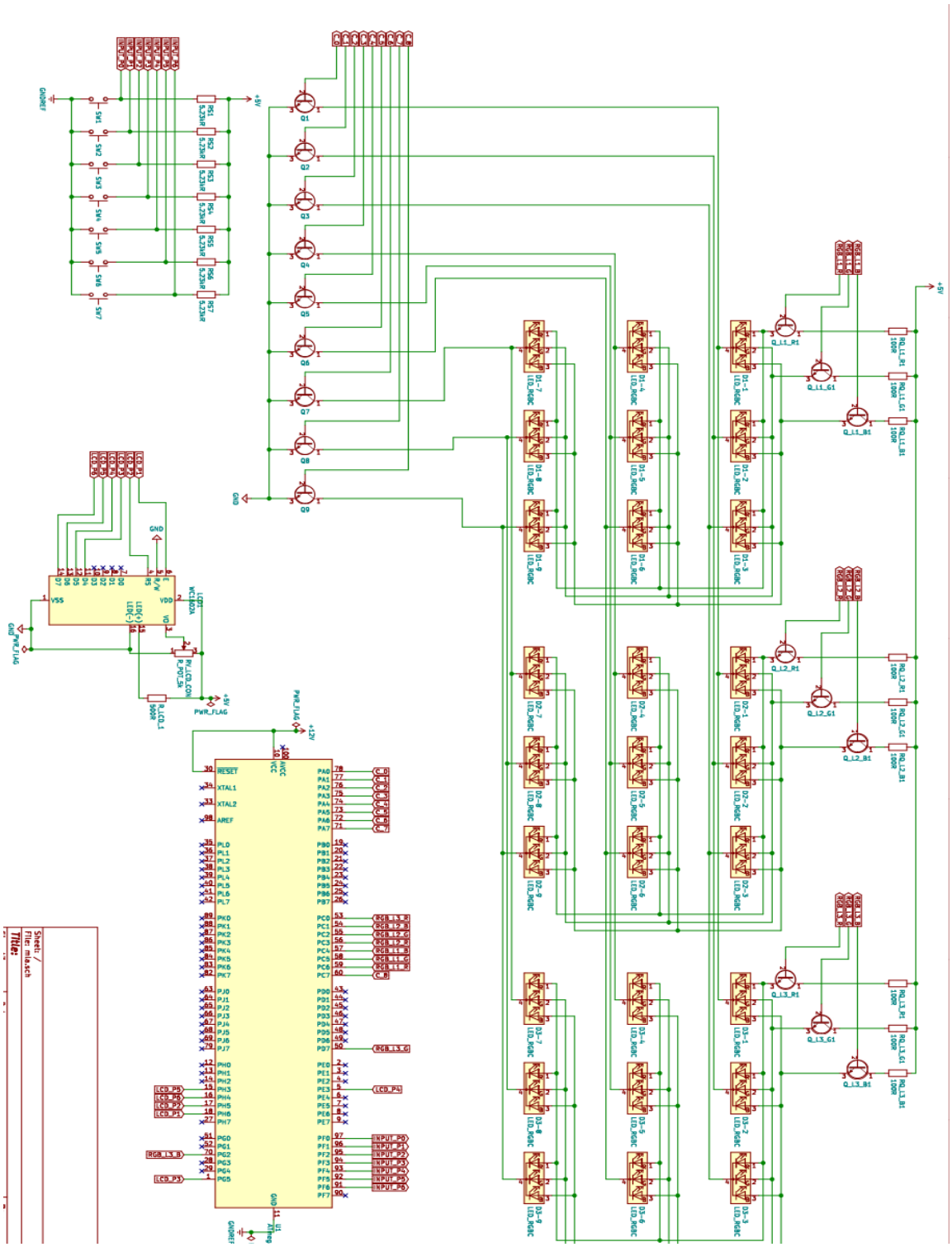
Otpornik je pasivna elektronska komponenta koja se koristi kao osnovni element električnih mreža i elektronskih uređaja. Osnovna karakteristika ove komponente je otpor koji stvara prolasku struje kroz sebe. Dijele se na fiksne otpornike koji uvijek imaju konstantnu vrijednost i na promjenjive otpornike kojima se vrijednost može mijenjati ručno. U ovom radu korišteni su fiksne metalno slojni otpornici (slika 10). Metal-film otpornici su slično izrađeni kao ugljenoslojni te je razlika ta što se na metalnoslojnima na keramičku jezgru nanosi tanak sloj metalnog filma, najčešće legura nikal-kroma (NiCr).

Slika 10: Fiksni metalnoslojni otpornici [19]

4. SHEMA SPOJA

Na shemi sklopa (slika 11) je prikazano kojim redoslijedom i kako se spajaju komponente. Shema je crtana u KiCad [23] programu. Radi lakše preglednosti spoja na neke žice su stavljeni mali kvadratići s brojem koji prikazuje gdje počinje i gdje završava koja žica. Spoj se napaja na 5V 2A te se dijeli na napajanje dioda i Arduina. RGB LED diode su tipa zajedničke katode što znači da ako se gleda pojedina boja kao zasebna dioda, sve “pod” diode bi imale katodu spojenu zajedno u jednu točku. Kocka je spojena tako da su sve katode u jednom stupcu zajedno spojene.

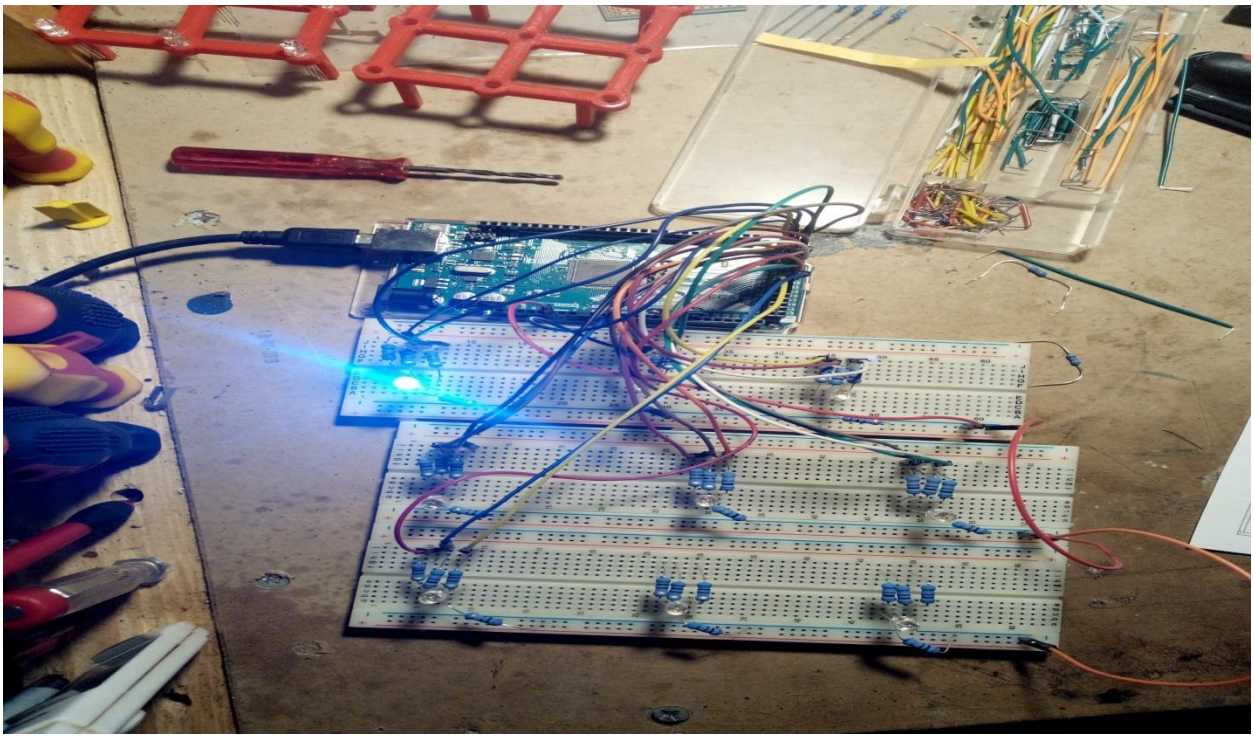
Primjer: Na shemi (slika 11) se vidi da su D1-1 s prvog sloja (ili kata), D2-1 s drugog sloja i D3-1 s trećeg sloja spojeni na zajedničku točku. Tipkala se spajaju na Arduino i služe za slanje signala Arduinu koji će upaliti željene LED diode. Tipkala su spojena na Arduino pomoću Pullup otpornika, što znači da je na svaki ulazni pin zasebno spojeno 5V napajanja preko otpornika (koji je u ovom slučaju 5.3 kΩ) i tipkalo koje kada je aktivno zatvori krug na ground napajanje. Tako su spojena kako bismo spriječili problem “floating” pin-a. Floating pin je kad na ulazni pin nije ništa spojeno ili je spojeno samo tipkalo koje je u stabilnom stanju otvoreno. Zbog toga kada se čita sa pina se ne može sa sigurnošću reći je li HIGH ili LOW jer kad je pin slobodan na očitanje znatno utječu vanjski i unutarnji šumovi. Na shemi se vidi da ima sveukupno 18 NPN tranzistora koji služe kao sklopke. Prva grupa tranzistora u kojoj je sveukupno 9 komada služi za kontroliranje boja svakog sloja, dok se druga grupa od 9 komada koristi za kontroliranje koji stupac katoda se spoji na ground napajanje. LCD display, na kojem će se prikazivati koji je igrač na redu za potez i tko je pobjedio u igri, spaja se direktno na Arduino sa 6 pinova.



Slika 11: Shema spoja
[Izvor: Izradio autor]

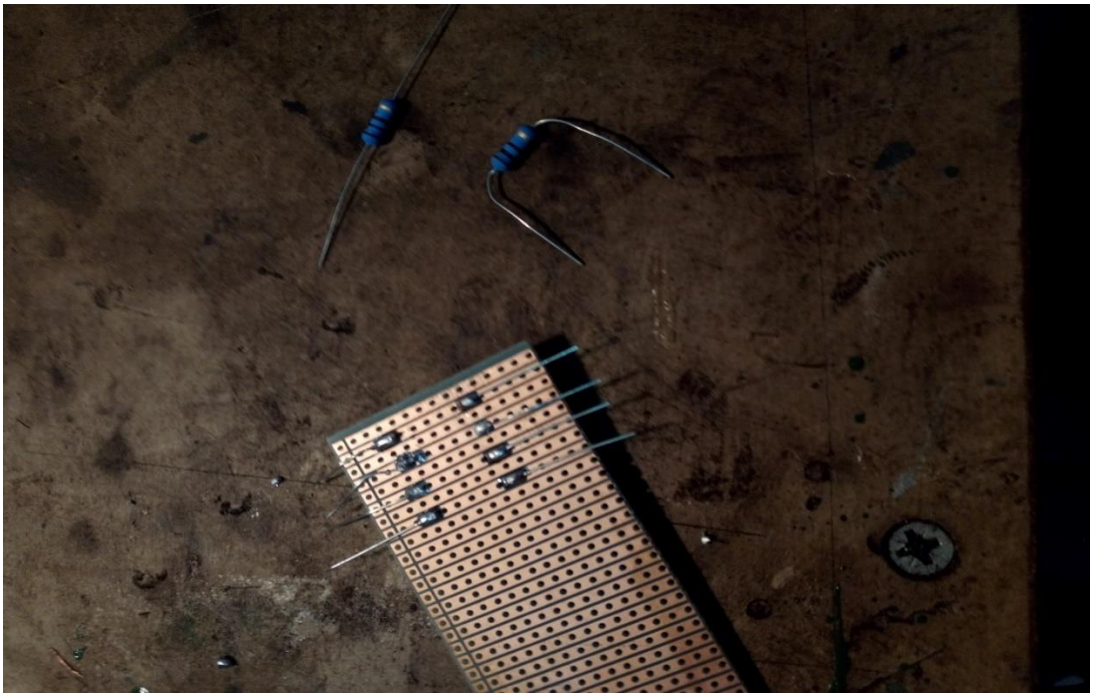
5. POSTUPAK IZRADE

Prije spajanja svih dijelova potrebno je provjeriti ispravnost komponenti koje su korištene u izradi rada. Pomoću programa za testiranje dioda i protoboarda spojen je sloj dioda i isprobana je jedna po jedna te je provjereno svijetle li sve njihove boje. Nakon testiranja LED dioda (slika 12) započeto je lemljenje otpornika, tranzistora, tipkala i žica na tiskane pločice koje se zatim spajaju na LED diode.

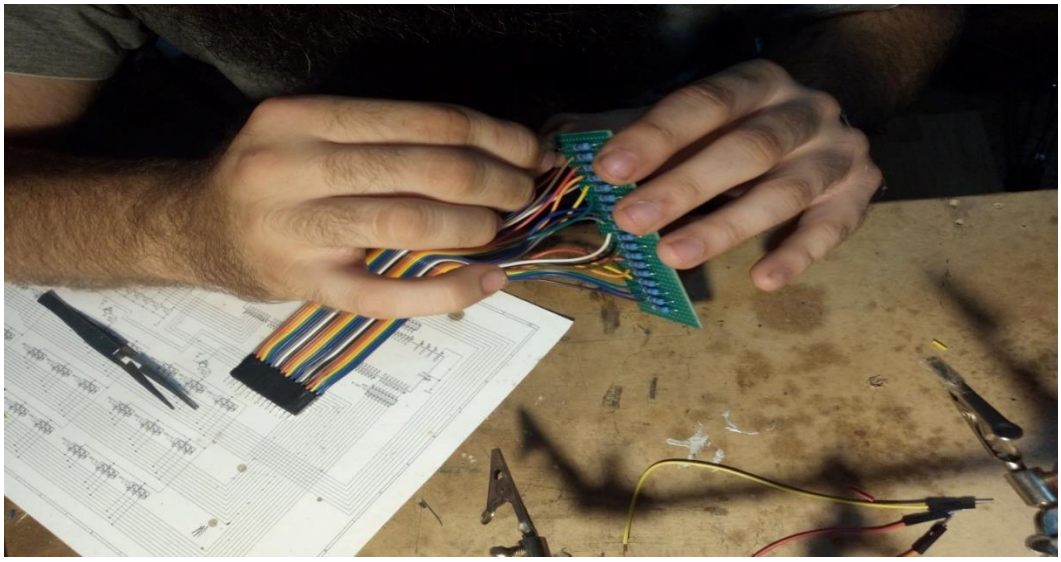


Slika 12: Testiranje LED dioda
[Izvor: Izradio autor]

Otpornici služe za ograničavanje tj. smanjivanje napona koji ide prema LED diodama jer bi bez limitacije LED diode pregorile. Otpornici nisu potrebni ukoliko je izvor napajanja točno onoliko koliko je potrebno LED diodi za rad. Nakon što su otpornici zalemljeni na pločicu (slike 13 i 14), potrebno je ostrugati bakar koji se nalazi između lemova. To je napravljeno kako bi protok struje išao kroz otpornike umjesto linijom manjeg otpora te time preskočio otpornike koji su spojeni na ploču.

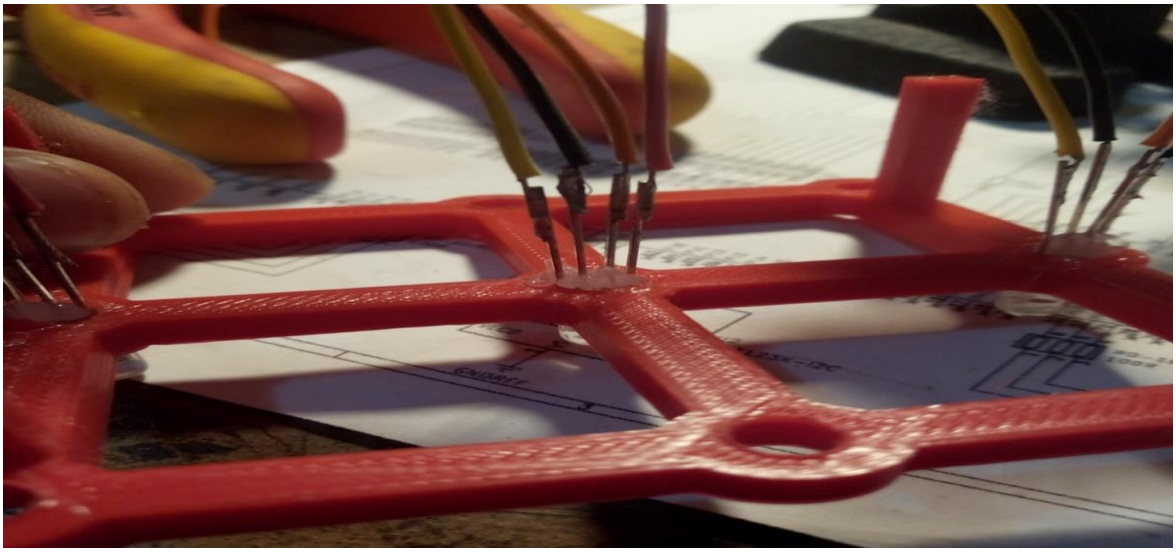


Slika 13: Zalemljeni otpornici
[Izvor: Izradio autor]

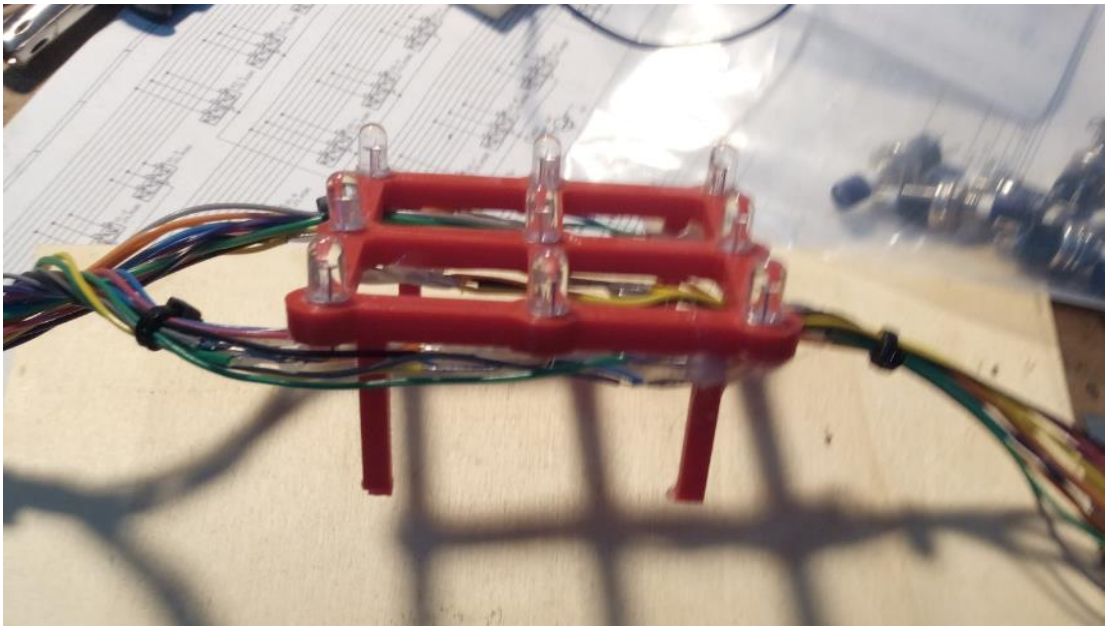


Slika 14: Zalemljeni otpornici i žice
[Izvor: Izradio autor]

Diode su stavljene u rupe koje su na nosačima (slika 15). Na njih su zalemljene žice koje će ići u kućište te je stavljeno toplo ljepilo kako bi ostale fiksirane u rupama.

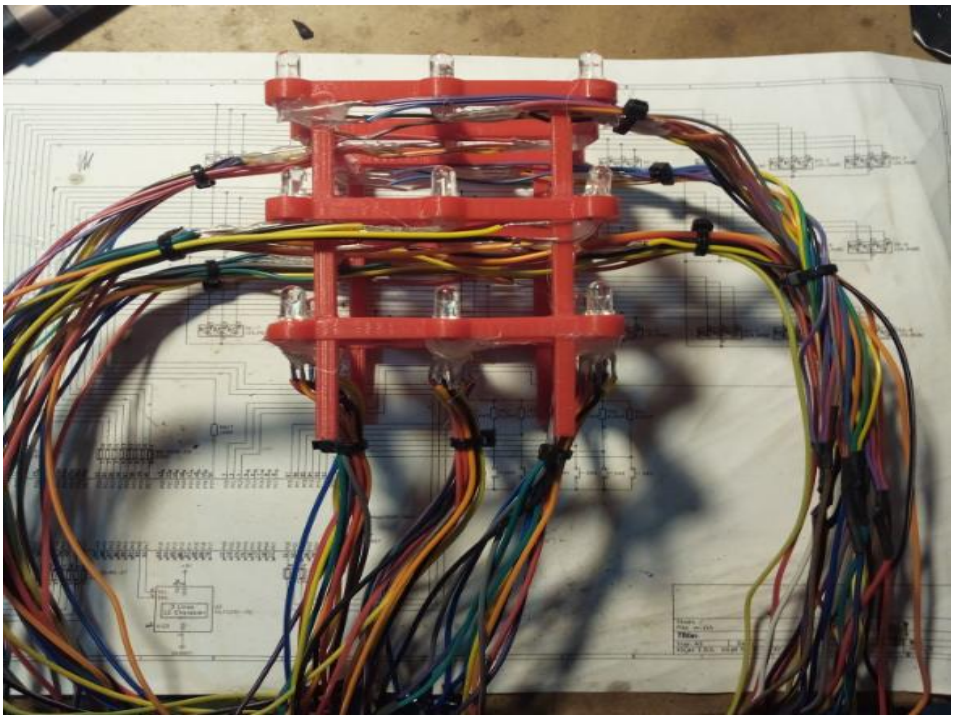


Slika 15: LED diode u utorima
[Izvor: Izradio autor]



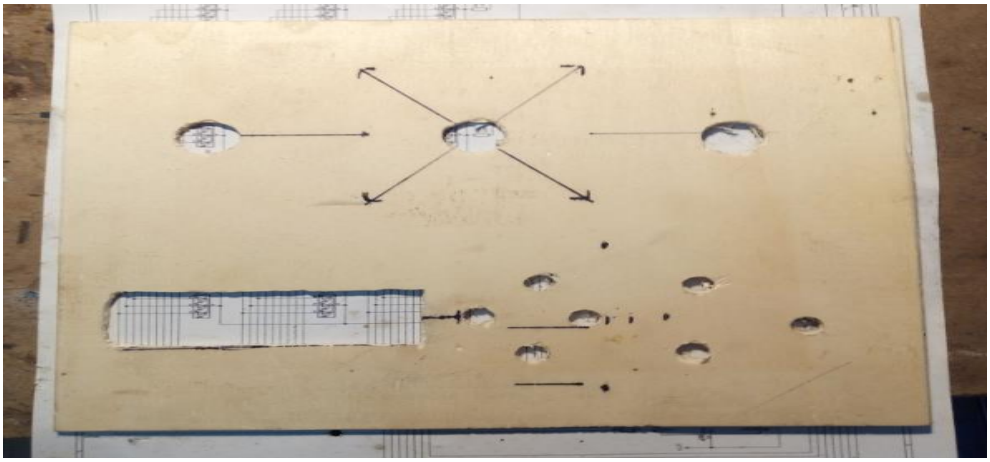
Slika 16: Prikaz izgleda zalemljenih LED dioda i žica
[Izvor: Izradio autor]

Žice za 2. i 3. kat su provučene sa strane (slike 16 i 17) kako bi cijela konstrukcija koja drži diode bila fiksirana na jedno mjesto te na taj način ostala ravna. RGB diode imaju tri boje crvenu, zelenu i plavu boju. Crvena predstavlja križić, plava kružić, a zelena služi kako bi prikazala diodu koju želimo odabrati.

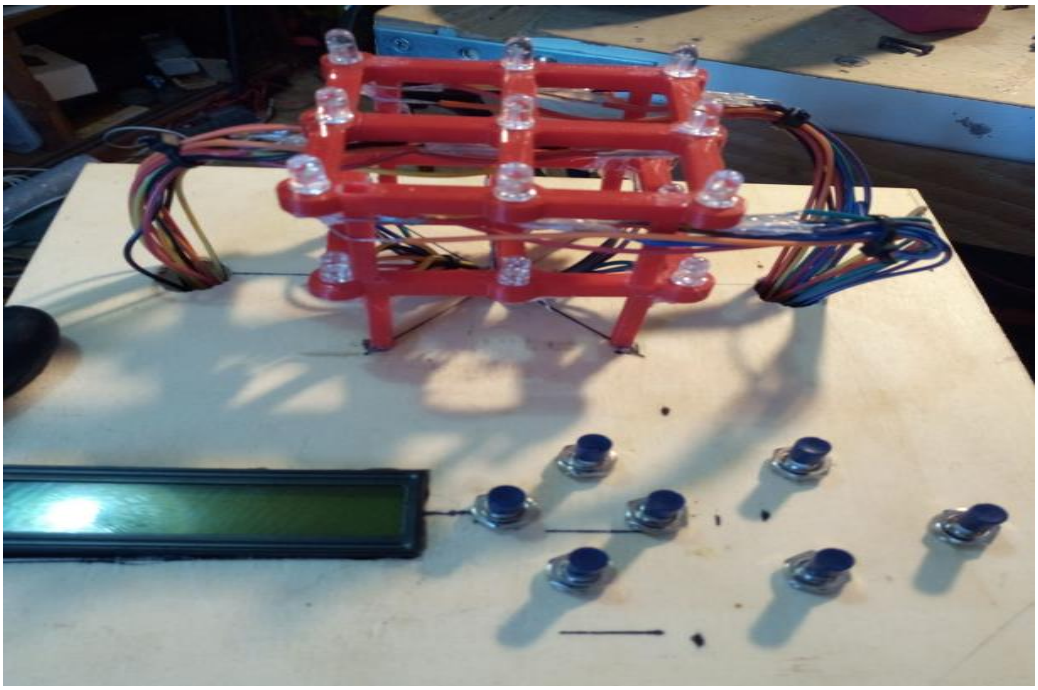


Slika 17: Prikaz kompletiranog nosača sa svim žicama i LED diodama
[Izvor: Izradio autor]

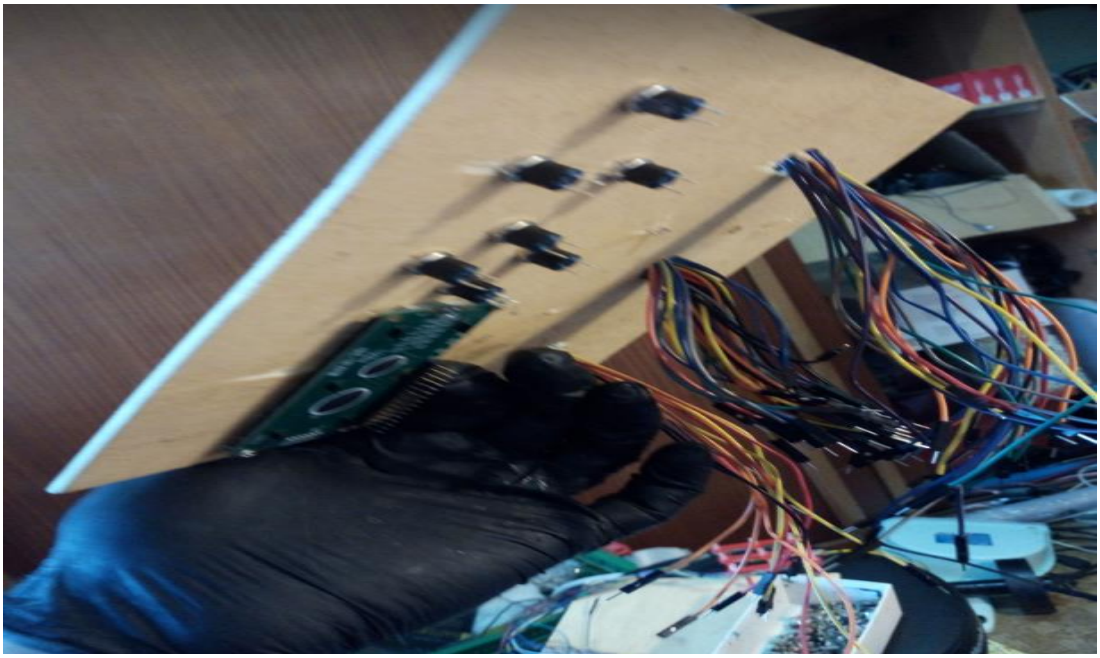
Na poklopcu kućišta (slika 18) su izbušene rupe koje služe za postavljanje LCD ekrana, tipkala te za provlačenje žica koje su spojene na LED diode. Nakon bušenja su dodane i pričvršćene ostale komponente (slike 19 i 20).



Slika 18: Poklopac kućišta
[Izvor: Izradio autor]



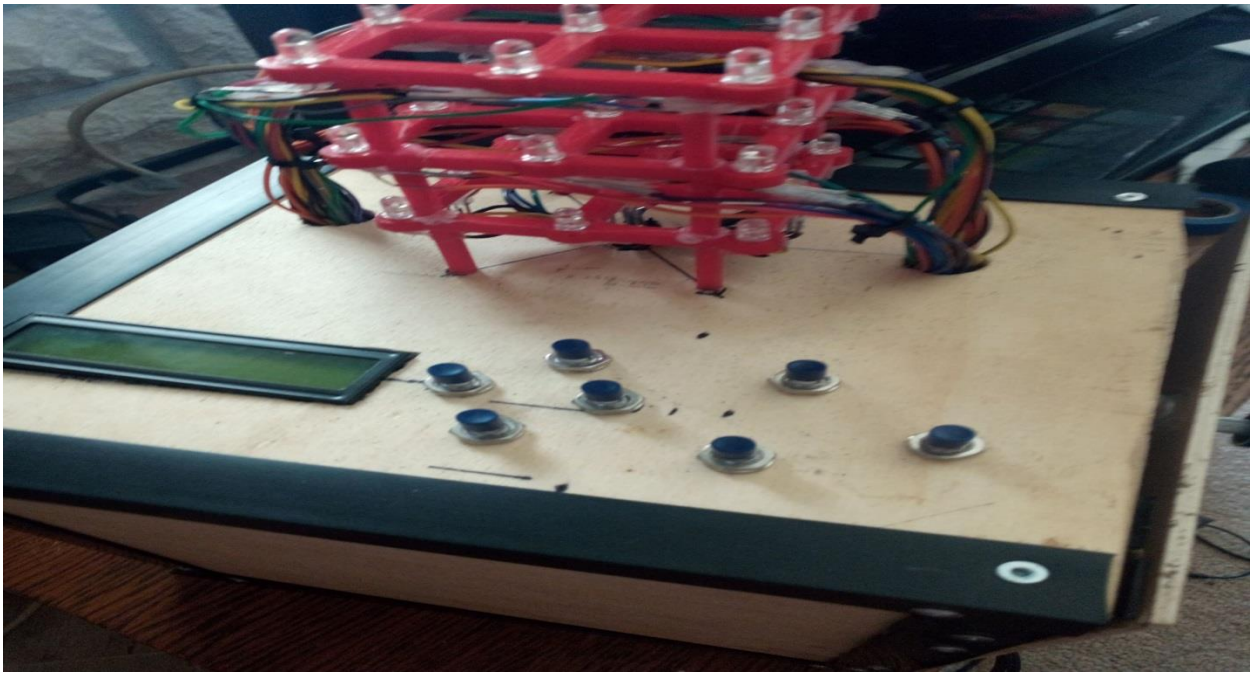
Slika 19: Prikaz ostalih komponenti dodanih na poklopac kućišta
[Izvor: Izradio autor]



Slika 20: Prikaz ostalih komponenti dodanih na poklopac kućišta – donja strana
[Izvor: Izradio autor]

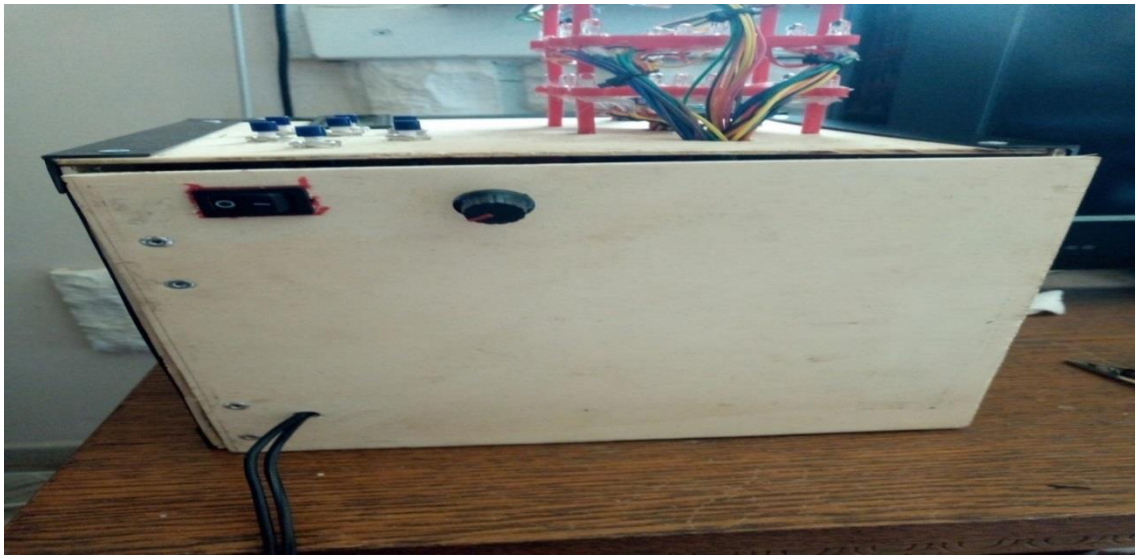
Kutija (slike 21, 22 i 23) je napravljena sa šperpločama koje su spojene plastičnim “L” profilima i pričvršćene zakovicama. Vrata su napravljena s desne strane

kućišta te se mogu otvoriti kako bi se vidjele unutarnje komponente projekta. Na toj strani se nalaze i ON/OFF sklopke, potenciometar te rupa koja je izbušena kako bi se mogle provući žice vanjskog napajanja.



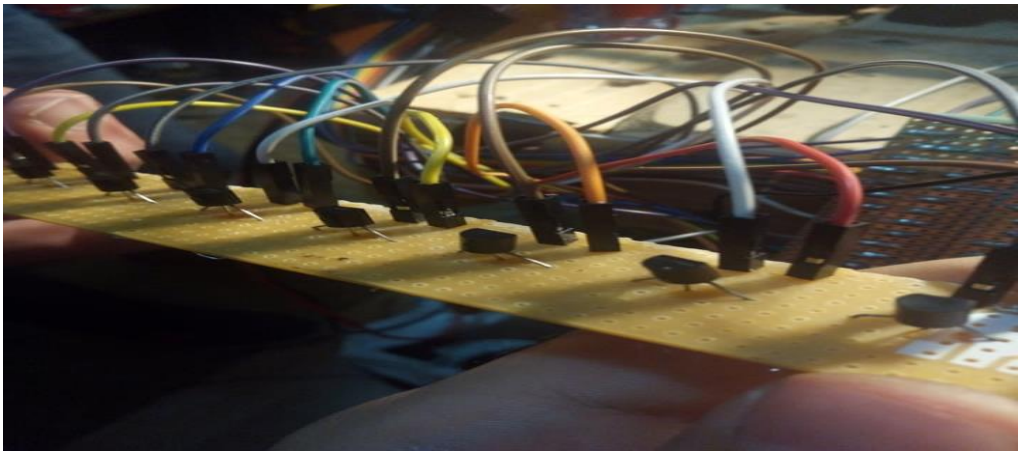
Slika 21: Izgled kutije
[Izvor: Izradio autor]

Slika 22: Izgled kutije
[Izvor: Izradio autor]



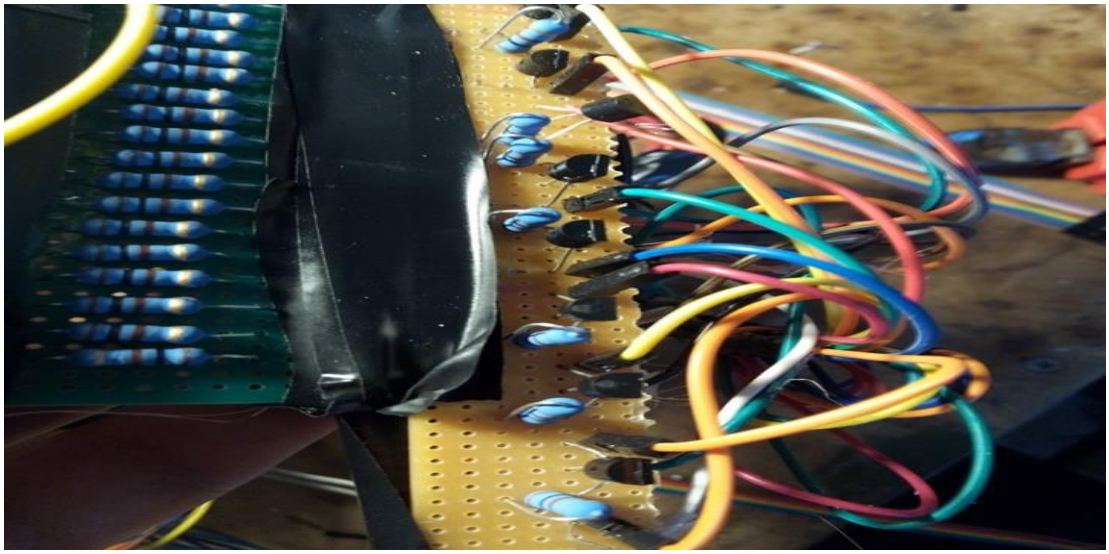
Slika 23: Izgled kutije
[Izvor: Izradio autor]

Tranzistori (slike 24 i 25) su korišteni kao sklopke. Jedan sloj tranzistora od 9 komada služi za kontroliranje pozicije križić-kružića i za eliminaciju reverse power output (povratna snaga). Bez tih tranzistora bi se moglo kontrolirati križić-kružić direktno preko Arduina, ali bi se na diodama koje se u tom trenutku ne koriste, ali su u standby modu i čekaju input da se upali, moglo dogoditi da veoma lagano svijetle što bi naposlijetku moglo ometati tijek igre.



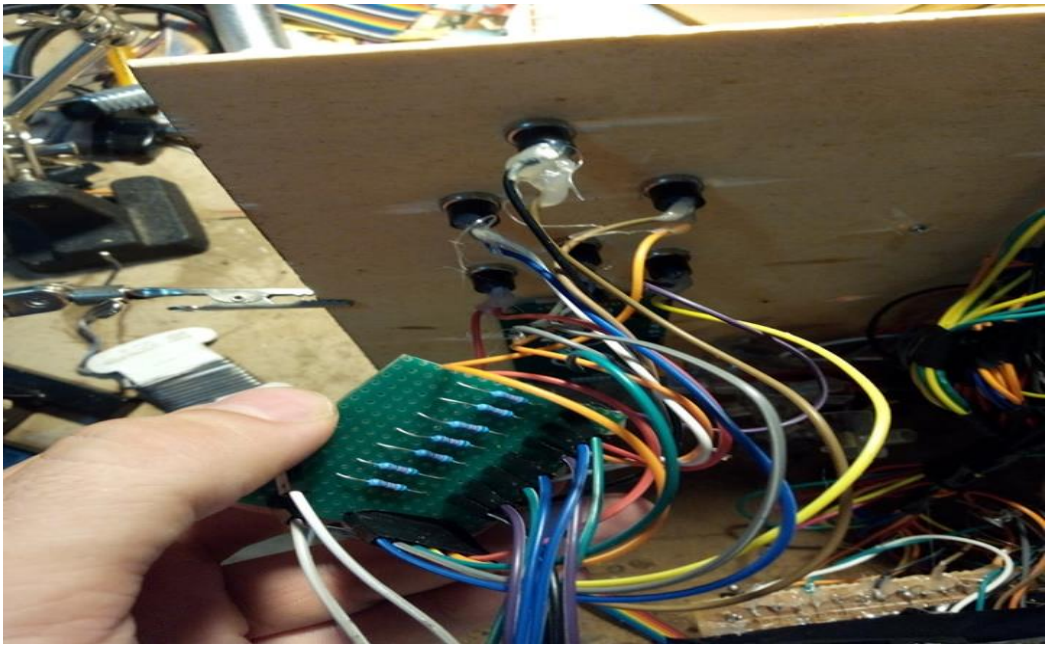
Slika 24: Tranzistori
[Izvor: Izradio autor]

Drugi sloj tranzistora služi za vanjsko napajanje dioda. Arduino nema dovoljno jaku struju da bi diode svijetlile kako treba. Vanjskim napajanjem je omogućeno da diode normalno svijetle kako bi preglednost same igre bila povećana.



Slika 25: Tranzistori
[Izvor: Izradio autor]

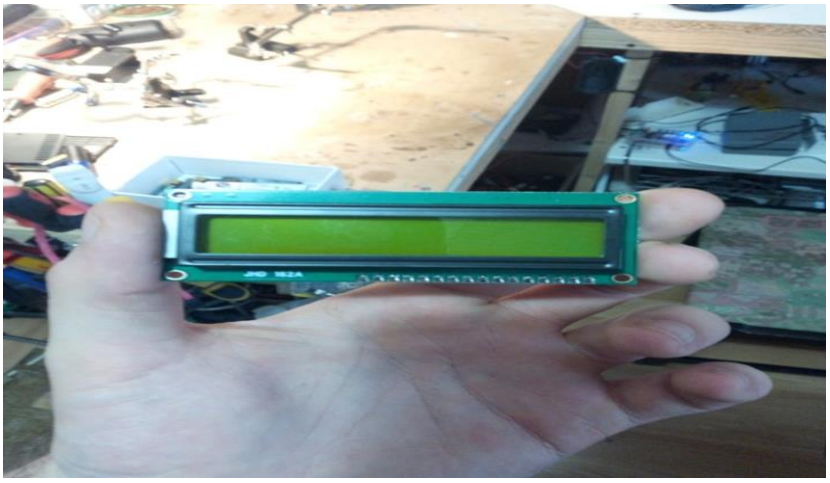
Tipkala (slika 26) služe za kontrolu ulaza tijekom odabira pozicije LED diode. Četiri tipkala služe za odabir pozicije na katu, dva za odabir kata te jedno tipkalo koje služi za potvrdu pozicije igrača. Zelena LED dioda uvijek je upaljena iz razloga što ona služi za orijentaciju po 3D kocki. Nakon što igrač odluči koju poziciju želi odigrati stisne tipkalo za potvrdu kako bi dioda koja je odabrana promijenila boju u crvenu ili plavu, ovisno o tome koji je igrač u tom trenutku bio na potezu.



Slika 26: Tipkala i otpornici
[Izvor: Izradio autor]

LCD je pričvršćen u utoru na poklopcu kućišta te se spaja direktno na Arduino uz pomoć žica koje imaju muško/ženski spoj jer LCD dolazi sa svojim pinovima za prikaz slike (slike 27 i 28).

Slika 27: LCD zaslon – stražnja strana
[Izvor: Izradio autor]



Slika 28: LCD zaslon – prednja strana
[Izvor: Izradio autor]

6. PROGRAMSKI KOD

Tri glavne funkcije programa su:

- "process_input",

- "update_cube" i

- "check_win".

Prije nego što provjeri ulaze koji se moraju “debounce”-ati radi fizičkih problema tipkala, process_input provjeri je li timer za debouncanje gotov. Naime, kod pritiska tipkala, prije nego što se kontakti stabiliziraju oni se par puta odbiju jedan od drugoga te to stvara problem jer je mikrokontroler dovoljno brz da registrira te skokove. Iz tog razloga će svaki put prilikom registriranja ulaza čekati prije registriranja slijedećeg kako bi se to spriječilo. Nakon toga samo prolaze svi ulazni pinovi i ako se registrira ulaz poziva se funkcija za pomicanje cursora ili za odabir pozicije.

Update_cube je funkcija koja u svakoj iteraciji petlje prikaže stanje kocke. U funkciji se prolazi svaki sloj kocke posebno i u svakom sloju ide se boja po boju. Za svaku boju prolazi se trenutno stanje na tom sloju i upale se sve diode koje trebaju i tako za svaku boju/sloj.

Check_win funkcija provjerava postoji li pobjednička kombinacija na kocki. Polje n-točka pozicija reprezentiraju pobjedničku kombinaciju. Prolaze se sve moguće pobjedničke kombinacije i ako se pronađu prikazuje se pobjednik te se kocka nakon nekog vremena resetira.

U prilogu ovog rada nalazi se cjeloviti proramski kod obrađene igre.

7. ZAKLJUČAK

Cilj ovog završnog rada je potvrditi postavljenu hipotezu te je zaključeno da radi kompleksnosti samog projekta izrada ovog rada ne bi bila moguća bez poznavanja korištenih elektroničkih komponenti kao ni bez poznavanja I razumijevanja potrebnog programskog jezika.

Usprkos tome što se rad možda na prvi pogled čini jednostavnim, mora se priznati da je u toku same izrade dolazilo do određenim poteškoćama zbog kompleksnosti izrade pojedinih dijelova te zbog veće količine manjih komponenti tijekom čijeg se spajanja mora pripaziti kako se same komponente važne za izradu ovog rada ne bi oštetile.

Iako je kroz povijest izrađeno mnogo različitih verzija ove igre, izradom ove verzije prikazan je drugačiji način izrade i igranja križić-kružića, odnosno postaje jedinstveno iskustvo. Spajanjem različitih verzija te na taj način kombiniranjem programskog koda i elektroničkih komponenti dolazi se do potpuno nove verziju ove igre za koju neki govore kako vuče svoje korijene još iz starog Egipta.

LITERATURA

Knjige

[1] Mike McGrath: Programiranje u jeziku C ++: prijevod četvrtog izdanja, Zagreb, Dobar plan, 2015.

Internet izvori

[2] https://boardgamegeek.com/boardgame/13714/qubic (14.06.2020.)

[3] https://store.arduino.cc/arduino-mega-2560-rev3 (20.05.2020.)

[4] https://magisterrex.files.wordpress.com/2014/07/qubic.pdf (14.06.2020.)

[5] http://anna.fi.muni.cz/~x139877/prezentace/2011_03_24_prednaska_ mafye_teorie_her/ piskvorky.pdf (14.06.2020.)

[6] https://www.instructables.com/id/Intro-to-Arduino/ (20.05.2020.)

[7] https://content.arduino.cc/assets/Pinout-Mega2560rev3_latest.pdf?fbclid=IwAR3H2kgLT_boC5zkZG8OxXdgOeLzQOtjyl1Z_CcZ1CrOA2d4uyTZ9qZCknM (20.05.2020.)

[8] http://home.physics.leidenuniv.nl/~eliel/teaching/fmt/kawamoto-history_of_lcds-procieee-2002.pdf (20.05.2020.)

[9] https://iuuk.mff.cuni.cz/~koblich/thesis/bachelor.pdf (14.06.2020.)

[10] https://en.wikipedia.org/wiki/Parker_Brothers (14.06.2020.)

[11] https://sites.google.com/site/boardandpieces/list-of-games/qubic (14.06.2020.)

[12] https://www.tradgames.org.uk/games/Four-in-a-row.htm (14.06.2020.)

[13] https://www.chipoteka.hr/artikl/114076/arduino-mega-2560-rev3-microcontroller-board-atmega2560-a000067-8090229035 (20.05.2020.).

[14] https://upload.wikimedia.org/wikipedia/commons/thumb/a/a1/Arduino_IDE_-_Blink.png/1200px-Arduino_IDE_-_Blink.png. (20.05.2020.)

[15] https://www.chipoteka.hr/artikl/145708/lcd-zaslon-i2c-16x2-plavi-za-arduino-9150024984 (30.04.2020.)

[16] https://www.chipoteka.hr/artikl/1166/bdx53c-npn-to220-dar-8a-100v-0470005303 (30.04.2020.)

[17] https://www.chipoteka.hr/artikl/4242/plocica-vetronit-200-x-300-2390005001 *(30.04.2020.)*

[18] https://e-radionica.com/wp/hrvatski/wp-content/uploads/sites/3/2016/07/Figure-3-Pushbutton-Breadboard-and-Schematic-Symbols.jpg *(30.04.2020.)*

[19] https://www.conrad.hr/metaloslojni-otpornik-100-omega%3b-aksijalno-ozicen-1-w-tru-components-tc-mor01sj0101a10203-1-kom. *(04.05.2020.)*

[20] https://en.wikipedia.org/wiki/3D_tic-tac-toe (14.06.2020.)

[21] http://gambiter.com/tabletop/Qubic.html (14.06.2020.)

[22] http://www.wylliedraughts.com/Qubic.htm (14.06.2020.)

[23] https://kicad-pcb.org/ (27.08.2020)

PRILOG

Programski kod

```
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

#define OLD_UPDATE
//#define UPDATE_DELAY

enum BoardState {
	B_NONE = 0,
	B_P1 = 1,
	B_P2 = 2
};

enum GameState {
	S_P1,
	S_P2,
	S_P1W,
	S_P2W
};
const String game_state_strings[][2] = {
	{"Crveni(X) je na","redu"},
	{"Plavi(O) je na","redu"},
	{"Crveni(X) je","pobijedio"},
	{"Plavi(O) je","pobijedio"}
};

enum Color : int {
	C_CLEAR = 0,
	C_RED = 1,
	C_GREEN = 2,
	C_BLUE = 3
};
```

```
enum Move {
	NORTH,
	SOUTH,
	EAST,
	WEST,
	UP,
	DOWN
};

struct Pos {
	int x;
	int y;
	int z;
};


static const int column_lu[] = {28, 26, 25, 23, 24, 22, 30, 27, 29};
static const int rgb_lu[3][3] = {{37, 39, 38}, {34, 36, 35}, {31, 33, 32}} ;

static const int input_lu[] = {54, 56, 57, 55, 58, 59, 60};
static const long int debounce_limit = 100;

static unsigned long int debouncer = 0;
static unsigned long display_timer = 0;

GameState state;
BoardState board[3][3][3];
Pos cursor_pos;
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
inline int column_lu_to_offset(Pos p)
{
	return p.x + p.y*3;
}
```

```
inline BoardState get_board_state(Pos p)
{
	return board[p.x][p.y][p.z];
}

void set_board_state(Pos p, BoardState s)
{
	board[p.x][p.y][p.z] = s;
}



void move_cursor(Move m)
{
#define check_and_update(c, l, u) if (cursor_pos.c == l) { \
	return;		\
} else {		\
	cursor_pos.c u;		\
	break;		\
}
switch (m) {
	case NORTH:
			check_and_update(y, 0, --);
	case SOUTH:
			check_and_update(y, 2, ++);
	case EAST:
			check_and_update(x, 2, ++);
	case WEST:
			check_and_update(x, 0, --);
	case UP:
			check_and_update(z, 2, ++);
	case DOWN:
			check_and_update(z, 0, --);
}
#undef check_and_update
```

```
debouncer = millis();
}

void display_win() {
  lcd.clear();
  lcd.setCursor(0, 0);
  if (state == S_P1W) {
    lcd.print("Crveni(X) je");
  } else {
    lcd.print("Plavi(O) je");
  }
  lcd.setCursor(0, 1);
  lcd.print("pobijedio");

  for (int i = 0; i < 9; i++)
   digitalWrite(column_lu[i], HIGH);
  int c = state == S_P1W? 0: 2;
  for (int i = 0; i < 3; i++)
            digitalWrite(rgb_lu[i][c], HIGH);

  delay(7500);

  setup();
}

void check_win()
{
  Pos win_states [][3] =  // sve moguce pobijednicke kombinacije
  {
     //{{0,0,0},{0,0,0},{0,0,0}},
     {{0,1,0},{1,1,0},{2,1,0}},
     {{0,2,0},{1,2,0},{2,2,0}},
     //{{0,0,1},{0,0,1},{0,0,1}},
     {{0,1,1},{1,1,1},{2,1,1}},
     {{0,2,1},{1,2,1},{2,2,1}},
```

```
//{{0,0,2},{0,0,2},{0,0,2}},
{{0,1,2},{1,1,2},{2,1,2}},
{{0,2,2},{1,2,2},{2,2,2}},
{{0,0,0},{0,1,0},{0,2,0}},
{{1,0,0},{1,1,0},{1,2,0}},
{{2,0,0},{2,1,0},{2,2,0}},
{{0,0,1},{0,1,1},{0,2,1}},
{{1,0,1},{1,1,1},{1,2,1}},
{{2,0,1},{2,1,1},{2,2,1}},
{{0,0,2},{0,1,2},{0,2,2}},
{{1,0,2},{1,1,2},{1,2,2}},
{{2,0,2},{2,1,2},{2,2,2}},
{{0,2,0},{1,1,0},{2,0,0}},
{{0,2,1},{1,1,1},{2,0,1}},
{{0,2,2},{1,1,2},{2,0,2}},
{{0,0,0},{1,1,0},{2,2,0}},
{{0,0,1},{1,1,1},{2,2,1}},
{{0,0,2},{1,1,2},{2,2,2}},
{{0,0,0},{1,1,1},{2,2,2}},
{{0,2,0},{1,1,1},{2,0,2}},
{{2,2,0},{1,1,1},{0,0,2}},
{{2,0,0},{1,1,1},{2,2,0}},
{{0,2,0},{1,2,1},{2,2,2}},
{{0,1,0},{1,1,1},{2,1,2}},
{{0,0,0},{1,0,1},{2,0,2}},
{{0,0,2},{1,0,1},{2,0,0}},
{{0,1,2},{1,1,1},{2,1,0}},
{{0,2,2},{1,2,1},{2,2,0}},
{{0,0,0},{0,0,1},{0,0,2}},
{{0,1,0},{0,1,1},{0,1,2}},
{{0,2,0},{0,2,1},{0,2,2}},
{{1,0,0},{1,0,1},{1,0,2}},
{{1,1,0},{1,1,1},{1,1,2}},
{{1,2,0},{1,2,1},{1,2,2}},
{{2,0,0},{2,0,1},{2,0,2}},
```

```
    {{2,1,0},{2,1,1},{2,1,2}},
    {{2,2,0},{2,2,1},{2,2,2}},
    {{0,0,0},{0,1,1},{0,2,2}},
    {{1,0,0},{1,1,1},{1,2,2}},
    {{2,0,0},{2,1,1},{2,2,2}},
    {{0,0,2},{0,1,1},{0,2,0}},
    {{1,0,2},{1,1,1},{1,2,0}},
    {{2,0,2},{2,1,1},{2,2,0}},
  };


  for (int i = 0; i < 48; i++) {

    BoardState s1 = get_board_state(win_states[i][0]);
    if (s1 == B_NONE)
      continue;
    BoardState s2 = get_board_state(win_states[i][1]);
    if (s2 == B_NONE)
      continue;
    BoardState s3 = get_board_state(win_states[i][2]);
    if (s3 == B_NONE)
      continue;

    if (s1 == s2 && s2 == s3) {
      if (s1 == B_P1)
        state = S_P1W;
      else
        state = S_P2W;

      display_win();
    }
  }
}
void select()
{
```

```
        BoardState s = get_board_state(cursor_pos);
        if (s == B_NONE) {
                if (state == S_P1) {
                        set_board_state(cursor_pos, B_P1);
                        state = S_P2;
                } else {
                        set_board_state(cursor_pos, B_P2);
                        state = S_P1;
                }
        } else {
                lcd.clear();
                lcd.print("Polje je zauzeto");
                delay(555);
        }
        debouncer = millis();
        check_win();
}

void process_input()
{

        if (millis() - debouncer < debounce_limit) {
                return;
        }
        for (int i = 0; i < 7; i++) {
                int status = digitalRead(input_lu[i]);
                if (status == LOW) {
                        switch (i) {
                                case 0:
                                        select();
                                        break;
                                case 1:
                                        move_cursor(UP);
                                        break;
                                case 2:
```

```
  move_cursor(DOWN);
                                                        break;
                                                case 3:
  move_cursor(NORTH);
                                                        break;
                                                case 4:
  move_cursor(SOUTH);
                                                        break;
                                                case 5:
                                                        move_cursor(EAST);
                                                        break;
                                                case 6:
                                                        move_cursor(WEST);
                                                        break;
                                        }

                                }
        }
}

Color get_color_from_state(BoardState s)
{
        switch (s) {
                        case B_NONE:
                                return C_CLEAR;
                        case B_P1:
                                return C_RED;
                        case B_P2:
                                return C_BLUE;
        }
}

#define turn_on(pin) digitalWrite(pin, HIGH)
```

```
#define turn_off(pin) digitalWrite(pin, LOW)

void reset_colors()
{
	for (int j = 0; j < 3; j++)
		for (int i = 0; i < 3; i++)
			turn_off(rgb_lu[j][i]);
}

void reset_colors_of_layer(int l)
{
	for (int i = 0; i < 3; i++)
		turn_off(rgb_lu[l][i]);
}

void reset_columns()
{
	for (int i = 0; i < 9; i++)
		turn_off(column_lu[i]);
}

void update_cube()
{
	for (int l = 0; l < 3; l++) {
		reset_colors();
		for (int c = 1; c < 4; c++) {
			reset_columns();
			int current_color = c-1;
			reset_colors();
			turn_on(rgb_lu[l][current_color]);
			for (int y = 0; y < 3; y++) {
				for (int x = 0; x < 3; x++) {
					Pos p = {x, y, l};
					Color bc;
```

```
                                                                    if (p.x ==
cursor_pos.x && p.y == cursor_pos.y && p.z == cursor_pos.z) {
                                                                        bc =
C_GREEN;
                                                                    } else {

  BoardState state = get_board_state(p);
                                                                        if
(state == B_NONE)

  continue;
                                                                        bc =
get_color_from_state(state);
                                                                    }


                                                                    if ((int)bc != c)

  continue;


                                                                    int col = x + y*3;

  turn_on(column_lu[col]);

  turn_off(column_lu[(col + 8) % 9]);

  delayMicroseconds(500);
                                                            }
                                                    }
                                            }
                    }
}
#undef turn_on
#undef turn_off



void setup()
{
```

```
	Serial.begin(115200);
	lcd.begin(16,2);

	for (int i = 0; i < 10; i++) {
			pinMode(column_lu[i], OUTPUT);
			digitalWrite(column_lu[i], LOW);
	}
	for (int i = 0; i < 3; i++) {
			for (int j = 0; j < 3; j++) {
					pinMode(rgb_lu[i][j], OUTPUT);
					digitalWrite(rgb_lu[i][j], LOW);
			}
	}

	for (int i = 0; i < 8; i++) {
			pinMode(input_lu[i], INPUT);
	}

	for (int i = 0; i < 3; i++) {
			for (int j = 0; j < 3; j++) {
					for (int k = 0; k < 3; k++) {
							board[i][j][k] = B_NONE;
					}
			}
	}
	debouncer = 0;
	state = S_P1;
	cursor_pos.x = 1;
	cursor_pos.y = 1;
	cursor_pos.z = 1;


}

void loop()
```

```
{
        process_input();
        update_cube();
        lcd.setCursor(0, 0);
        lcd.print(game_state_strings[state][0]);
        lcd.setCursor(0, 1);
        lcd.print(game_state_strings[state][1]);
        // print the number of seconds since reset:

}
```