Igra pamćenja na arduino platformi

Majić, Iva

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:771617

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

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

Sveučilišni studij računarstva

IGRA PAMĆENJA NA ARDUINO PLATFORMI

Završni rad

Iva Majić

Osijek, 2017.

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. ARDUINO MIKROUPRAVLJAČKA PLATFORMA 2

2.1. Arduino Uno 2

2.2. Arduino IDE i Arduino skica 3

3. IGRA PAMĆENJA 5

3.1. Sklopovlje 5

3.1.1. RTC 6

3.1.2. LCD ekran 6

3.1.3. LED matrični pokaznik 7

3.1.4. Tipkalo 9

3.1.5. Shema sustava 10

3.2. Programska podrška 10

3.2.1. Globalne varijable i biblioteke 11

3.2.2. Funckija setup() 12

3.2.3. Funkacija loop() 13

3.2.4. Korisnički definirane funkcije 15

3.3. Testiranje 18

4. ZAKLJUČAK 22

LITERATURA 23

SAŽETAK 24

ABSTRACT 25

ŽIVOTOPIS 26

PRILOZI 27

1. UVOD

Zadatak završnog rada izrada je igre pamćenja koristeći Arduino mikroupravljačku platformu. Igra se temelji na popularnoj društvenoj igri gdje je zadatak pronaći parove kartica i testirati vlastito pamćenje. Postoje četiri težine igre, a razlikuju se u broju kartica kojima je potrebno pronaći parove. Korisnik pomoću Arduino sklopovlja odabire željenu karticu, a u pozadini se pomoću programskog koda provjeravaju parovi. U prvom poglavlju objasnit će se kako je sklop sastavljen. Zatim će se u drugom poglavlju obraditi struktura cijelog programa na kojem je igra zasnovana. Naposljetku će u trećem poglavlju biti objašnjeno na koji način je testirana ispravnost sklopa te će se u posljednjem poglavlju zaključiti rad.

1.1. Zadatak završnog rada

Potrebno je osmisliti, razviti i testirati igru pamćenja na Arduino platformi.

2. ARDUINO MIKROUPRAVLJAČKA PLATFORMA

Arduino je mikroupravljačka platforma otvorenog koda koja koristi mikroupravljače kako bi različite ulaze (primjerice pritisak gumba ili svjetlost na senzoru) pretvorila u izlaze (ispis teksta na ekranu, uključivanje LE (eng. *light-emitting*) diode i sl.) [1]. Većina Arduino razvojnih sustava koristi Atmel 8-bitne AVR mikroupravljače.

Arduino razvojni sustavi sadrže više digitalnih i analognih nožica (eng. *pin*) za ulaze i izlaze, kao i serijska komunikacijska sučelja, primjerice USB (eng. *Universal Serial Bus*). Nožice se, ovisno o razvojnom sustavu, nalaze u jednom ili dva reda i koriste se kako bi se razvojni sustav povezao s različitim modulima, senzorima i slično. USB se koristi kako bi se razvojni sustav spojio s računalom, tj. preko njega se izvodi programiranje Arduino razvojnog sustava [2].

Postoji više različitih Arduino razvojnih sustava koji se razlikuju po dimenzijama, količini memorije, frekvenciji procesora, broju nožica itd. Za potrebe ovog rada korišten je razvojni sustav Arduino Uno.

Većina programa za Arduino pisana je u varijacijama programskih jezika C i C++, a putem kompajlera pretvara se u strojni jezik. U praksi se za to najčešće upotrebljava službeno programsko okruženje Arduino IDE (eng. *Integrated Development Environment*) [2].

2.1. Arduino Uno

Arduino Uno baziran je na ATmega328 mikroupravljaču te u skladu s time ima 32 KB memorije. Osim toga, ima i 2 KB statičke radne memorije. Sadrži četrnaest nožica za digitalne ulaze/izlaze (od kojih se šest može koristiti kao PWM (eng. *Pulse-width Modulation*) izlazi), šest analognih ulaza, keramički rezonator frekvencije 16 MHz, USB vezu, utičnicu za napajanje, ICSP (eng. *In Circuit Serial Programming*) priključak i tipkalo za ponovno postavljanje [3].

Arduino UNO napaja se pomoću USB veze ili putem vanjskog izvora istosmjerne električne struje. Napon treba biti između 6 V i 20 V, s time da proizvođač preporuča napon između 7 V (minimum potreban da bi sklop radio stabilno) i 12 V (budući da veći naponi mogu izazvati pregrijavanje razvojnog sustava) [3].

Arduino UNO sastoji se od nožica za digitalne signale, nožica za analogne signale, nožica za napajanje te još dvije dodatne nožice.

Nožice za digitalne signale nalaze se pod brojevima od 0 do 13, a to su:

- serijske nožice (0 (RX) i 1 (TX)) koje se koriste za primanje/slanje TTL serijskih podataka
- nožice pod brojevima 2 i 3 koje se mogu konfigurirati kao okidač za prekid (eng. *interrupt*) pri određenom naponu (niski napon, padajući/rastući brid, promjena vrijednosti)
- PWM nožice (3, 5, 6, 9 i 10) koje omogućuju generiranje analognog signala koristeći digitalni ulaz
- SPI (eng. *Serial Peripheral Interface*) nožice (10, 11, 12 i 13) koje omogućuju brzu komunikaciju na manjoj udaljenosti između mikroupravljača i perifernog uređaja [4]
- LED nožica (pod brojem 13) – na razvojnom sustavu postoji ugrađena LED dioda koja je spojena s ovom nožicom.

Nožice za analogne signale imaju oznake od A0 do A5 te također mogu služiti i kao digitalni ulazi/izlazi. Bitno je spomenuti nožice za TWI (eng. *Two Wire Interface*). TWI omogućuje brzu komunikaciju na manjoj udaljenosti između više perifernih uređaja (eng. *Slave*) i jednog ili više upravljača (eng. *Master*). Pri tome je nožica A4 za podatke: SDA (eng. *data*), a nožica A5 za takt: SCL (eng. *clock*).

Na Arduino UNO razvojnom sustavu postoji šest nožica za napajanje, a to su:

- VIN koja služi kao ulazni napon kada razvojni sustav za napajanje koristi vanjski izvor struje – ulazni napon je 9 V, za razliku od 5 V prilikom USB napajanja
- 5V, tj. izlazni napon iznosa 5 V pri čemu napon dolazi iz izvora (USB ili vanjski izvor), a koristi se za napajanje mikroupravljača i ostalih komponenti sklopa
- 3V3, tj. napon iznosa 3.3 V koji se generira regulatorima na razvojnom sustavu
- tri nožice pod nazivom GND (eng. *Ground*) koje se koriste za uzemljenje.

Osim toga sadrži još dvije nožice:

- AREF (eng. *Analog Reference Input*), što je referentni napon za analogne ulaze te
- RESET, odnosno iglica za ponovno postavljanje – ukoliko se ova iglica dovede u stanje niskog napona, mikroupravljač se ponovno postavi na početne postavke [5].

2.2. Arduino IDE i Arduino skica

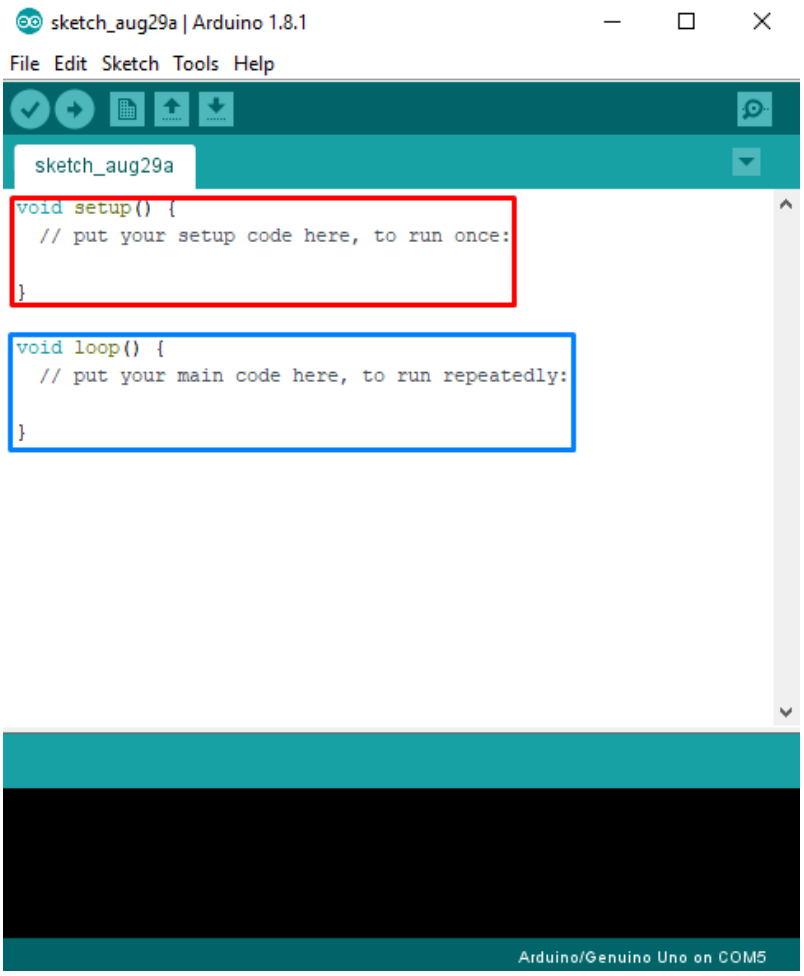
Arduino IDE program je otvorenog koda, napisan u programskom jeziku Java, a koristi se kako bi se kôd napisan u programskom jeziku C ili C++ preveo u strojni jezik i poslao procesoru [6]. Program napisan za Arduino naziva se skica (eng. *sketch*) i sprema se na računalo kao tekstualna

datoteka s nastavkom *.ino*. Svaka skica sastoji se od minimalno dvije funkcije: *setup()* i *loop()*. Na slici 2.1. prikazan je prozor Arduino IDE s novom, praznom skicom.

Setup() se poziva na početku skice, odnosno nakon što se sustav pokrene. Uglavnom se koristi za inicijalizaciju varijabli, definiranje ulaza i izlaza i sl. Na slici 2.1. *setup()* funkcija označena je crvenom bojom.

Loop() se poziva nakon funkcije *Setup()* te se ponavlja sve dok se sustav ne ugasi ili ponovno pokrene. Na slici 2.1. *loop()* funkcija označena je plavom bojom.

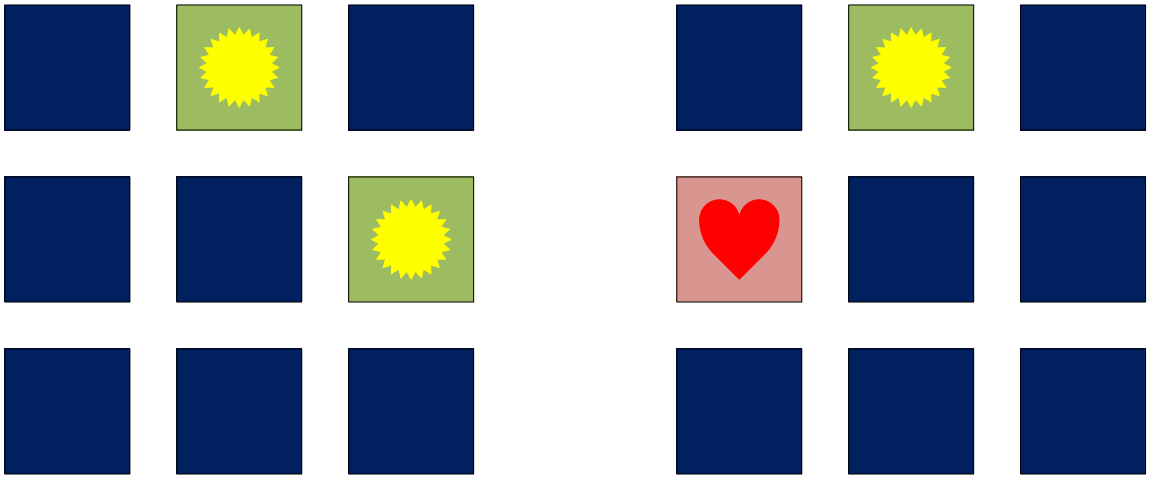
Osim ove dvije funkcije, u skici se mogu nalaziti i druge, korisnički definirane funkcije koje se koriste za bolju preglednost koda te njegovo pojednostavljivanje. Ukoliko određeni kôd treba ponoviti više puta, preglednije je iz glavnog programa pozvati funkciju u kojoj se taj kôd nalazi, nego ponavljati isti kôd nekoliko puta u glavnom programu.



Sl. 2.1. *Sučelje Arduino IDE.*

3. IGRA PAMĆENJA

Igra je osmišljena na način da se pri uključivanju Arduino UNO razvojnog sustava pojavi izbornik za odabir težine. Postoje četiri različite težine, a razlika je u broju parova kartica koje treba pronaći. Moguće je birati između dva, četiri, šest ili osam parova. Nakon odabira težine, postavlja se igra: na ekranu se ispisuje trenutni rezultat (0) te se igraču omogućuje da tipkalima bira koju karticu želi otvoriti. Nakon odabira prve kartice, igrač može odabrati drugu karticu. Kad su odabrane obje kartice, program provjerava jesu li kartice jednake, odnosno je li igrač pronašao par. Ukoliko jest, kartice ostaju otvorene kako ih se više ne bi biralo, rezultat se povećava i igrač može odabrati sljedeći par kartica. Općeniti primjer pogođenog para u igri pamćenja prikazan je na lijevoj strani slike 3.1. Ukoliko kartice nisu jednake, neko vrijeme ostaju otvorene kako bi igrač stigao zapamtiti o kojim se karticama radi, nakon čega se zatvaraju bez povećavanja rezultata i igraču se omogućuje ponovno biranje para. Općeniti primjer pogrešnog izbora para kartica u igri pamćenja prikazan je na desnoj strani slike 3.1. Kad je igrač pronašao sve parove, igra ga obavještava o završetku te se nakon nekoliko sekundi ponovno pokreće na početnom izborniku.



Sl. 3.1. *Općeniti prikaz igre pamćenja, lijevo pogođeni par, desno pogrešan par.*

3.1. Sklopovlje

Kako bi igra bila u potpunosti funkcionalna, Arduino UNO razvojni sustav spaja se s dodatnim modulima. Koriste se RTC (eng. *Real Time Clock*) modul, LCD (eng. *Liquid Crystal Display*) ekran, četiri 8x8 LED matrična pokaznika te pet tipkala. U sljedećim potpoglavljima objašnjeni su pojedini moduli korišteni u izradi igre.

3.1.1. RTC

RTC modul je uređaj za precizno praćenje vremena. Funkcionira pomoću kristalnih oscilatora kako bi mjerio datum (godina, mjesec, dan te dan u tjednu) i vrijeme (sate, minute, sekunde).

Budući da se RTC u ovom radu koristi isključivo kao generator nasumičnih brojeva za položaj kartica pri postavljanju igre (što će detaljnije biti objašnjeno pri analizi samog koda), nije nužno da uređaj mjeri točno trenutno vrijeme, samo je bitno da postoji promjena vremena koja bi omogućila da se svaki put generira drugačiji nasumični broj. Međutim, kada se jednom postavi na točno vrijeme, ovaj uređaj mjeri datum i vrijeme sve dok postoji izvor napajanja (preko Arduino pločice ili preko vanjske baterije).

RTC modul se s Arduino pločicom povezuje pomoću I^2C komunikacije. Način spajanja prikazan je u tablici 3.1.

Tab. 3.1. *Povezivanje RTC modula s Arduinom.*

RTC	Arduino
SCL	A5
SDA	A4
VCC	5V
GND	GND

3.1.2. LCD ekran

LCD ekran bazira se na tehnologiji tekućih kristala. Ekran se sastoji od više točaka (eng. *pixel*) od kojih se svaka sastoji od sloja molekula poredanih između elektroda (koje te molekule okreću u određenom smjeru) te dva polarizatora. Zbog elektroda, molekule su orijentirane spiralno, što utječe na polarizaciju svjetlosti te ekran „ne svijetli“. Međutim, ukoliko se dovede određeni napon (ovaj ekran radi na naponu od 5 V), molekule se ispravljaju, svjetlost je okomito polarizirana na filter te se apsorbira što omogućuje da točka „svijetli“ [7].

Za potrebe igre korišten je LCD ekran s HD44780 upravljačem na koji je moguće ispisati tekst u dva reda po šesnaest znakova, odnosno ukupno 32 znaka.

Kako bi se za spajanje ekrana izravno s Arduino sustavom trebao iskoristiti velik broj nožica, upotrijebljen je I^2C adapter koji uvelike smanjuje broj potrebnih nožica. Način spajanja LCD ekrana na Arduino sustav pomoću I^2C modula prikazan je u tablici 3.2.

Tab. 3.2. *Spajanje LCD ekrana s Arduinom pomoću I^2C adaptera.*

I^2C	Arduino
SCL	A5
SDA	A4
VCC	5V
GND	GND

Može se primijetiti da je način spajanja isti kao i za RTC modul jer se koristi isti način komunikacije. Iz tog razloga ni RTC ni LCD nisu na Arduino pločicu spojeni izravno, nego putem eksperimentalne pločice.

Arduino IDE dolazi s bibliotekom za LCD, no ista ne funkcionira kad je ekran spojen pomoću I^2C adaptera. Zato je preuzeta prilagođena biblioteka te se ona koristila u kôdu [8].

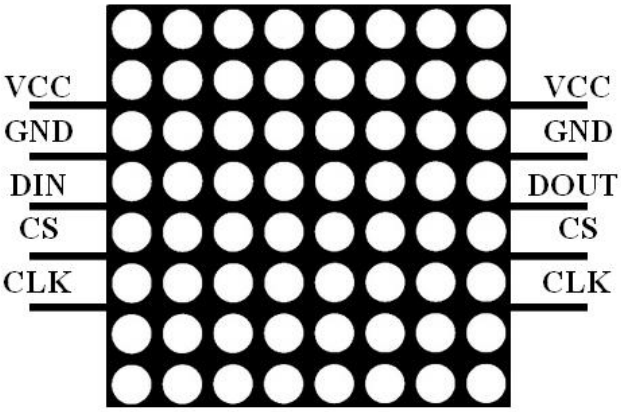
LCD ekran korišten je za ispis početnog izbornika (za odabir težine), ispis rezultata tijekom igre te ispis obavijesti o završetku igre.

3.1.3. LED matrični pokaznik

Kao glavni izlaz sklopa korištena su četiri LED matrična pokaznika. Svaki matrični pokaznik sastoji se od 8x8, odnosno 64 međusobno povezane LE diode. Matrični pokaznik upravljan je MAX7219 sklopom koji omogućuje serijsko spajanje više LED matričnih pokaznika. Način spajanja prvog matričnog pokaznika s Arduino razvojnim sustavom prikazan je u tablici 3.3. Promatrajući pokaznik, prikazan na slici 3.2., može se vidjeti da su nožice s lijeve strane matrice identične onima s desne strane, s iznimkom nožice DIN, kojoj je na desnoj strani nožica DOUT. Kako bi se spojilo više pokaznika u seriju, potrebno je samo spojiti nožice na desnoj strani jednog pokaznika s njihovim paritetima na lijevoj strani sljedećeg pokaznika. Na taj način su četiri potrebna pokaznika povezana u seriju.

Tab. 3.3*. Način spajanja LED matričnog pokaznika s Arduinom.*

LED pokaznik	Arduino
VCC	5V
GND	GND
DIN	D8
CS	D10
CLK	D11

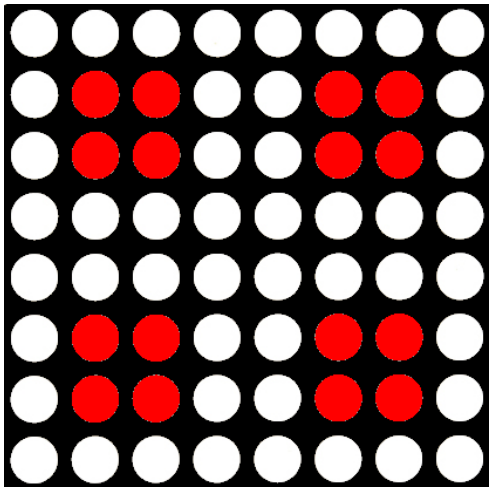


Sl. 3.2. *Nožice LED matričnog pokaznika.*

U sklopu su LED matrični pokaznici korišteni za prikaz kartica. Svaka matrica je isprogramirana tako da kartice prikazuje po shemi na slici 3.3.

Dakle, svaka kartica sastoji se od četiri LE diode, što znači da postoji 2^4 mogućih kombinacija (LE dioda svijetli ili ne svijetli). Ako se izuzmu kombinacije gdje svijetli samo po jedna LE dioda, gdje sve četiri diode svijetle te gdje nijedna dioda ne svijetli, preostaje deset mogućih kombinacija. Te kombinacije, prikazane na slici 3.4., upotrijebljene su kao „slike“ koje se nalaze na karticama. Na slici crvena točka predstavlja diodu koja svijetli, dok crna točka prikazuje diodu koja ne svijetli.

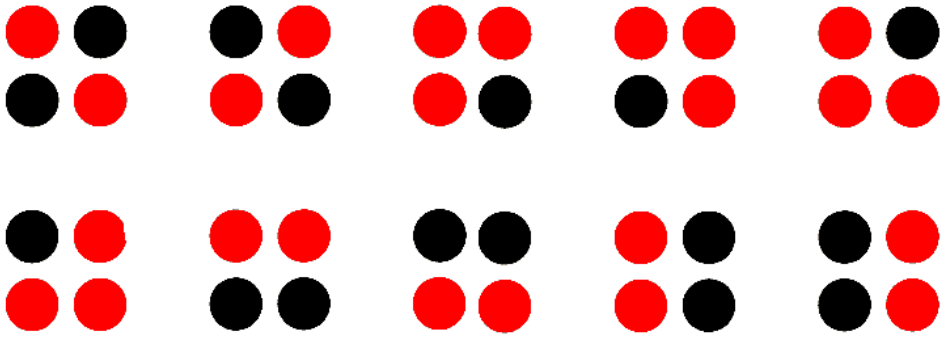
Matrični pokznik može prikazati četiri kartice pa broj pokaznika (od četiri dostupna) koji će se koristiti u igri ovisi o odabranoj težini igre. Ovisnost broja upotrebljenih matričnih pokaznika o težini igre, odnosno broju parova koje je potrebno pronaći prikazana je u tablici 3.4.



Sl. 3.3. *Shema položaja kartica na LED matričnom pokazniku.*

Tab. 3.4. *Ovisnost broja korištenih matričnih pokaznika o težini igre.*

Težina	Broj parova kartica	Broj pokaznika
4	2	1
8	4	2
12	6	3
16	8	4



Sl. 3.4. *Prikaz „slika“ na otvorenim karticama.*

3.1.4. Tipkalo

Tipkalo (eng. *push button*) jednostavan je element koji spaja ili prekida strujni krug, tj. u ovom radu na digitalni ulaz dovodi logičku '0' ili '1'. U radu je korišteno ukupno pet tipkala. Četiri tipkala pomiču pokazivač na LED matričnom pokazniku ili LCD ekranu, a jedno se koristi za odabir težine ili kartice.

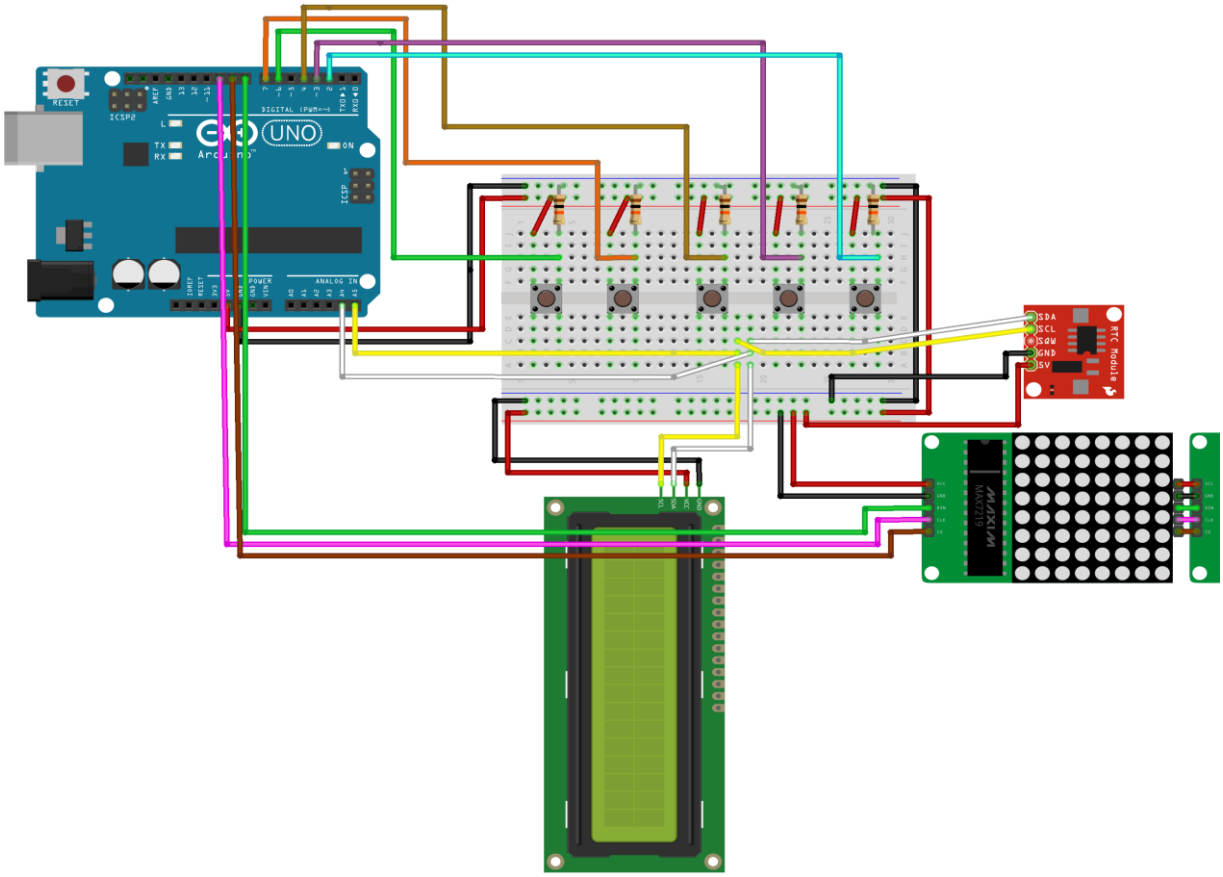
Tipkalo je potrebno spojiti na eksperimentalnu pločicu te dovesti napajanje i spojiti sa željenom digitalnom nožicom na Arduinu (tablica 3.5). Međutim, nužno je i svakom tipkalu dodati i *pull-down* otpornik vrijednosti 10 kΩ. Kad tipkalo nije pritisnuto, nije niti u stanju logičke nule niti u stanju logičke jedinice, ono je na tzv. „plivajućem naponu“. Arduino tada neće ispravno interpretirati pritisak tipkala. *Pull-down* otpornik osigurava da je na ulazu napon od 0 V (logička nula) ako tipkalo nije pritisnuto. Pritiskom tipkala vrijednost ulaza prelazi u logičku jedinicu.

Tab. 3.5. *Spajanje tipkala na eksperimentalnoj pločici s Arduinom.*

Tipkalo	Lijevo	Desno	Sredina	Gore	Dolje
Nožica	6	7	4	3	2

3.1.5. Shema sustava

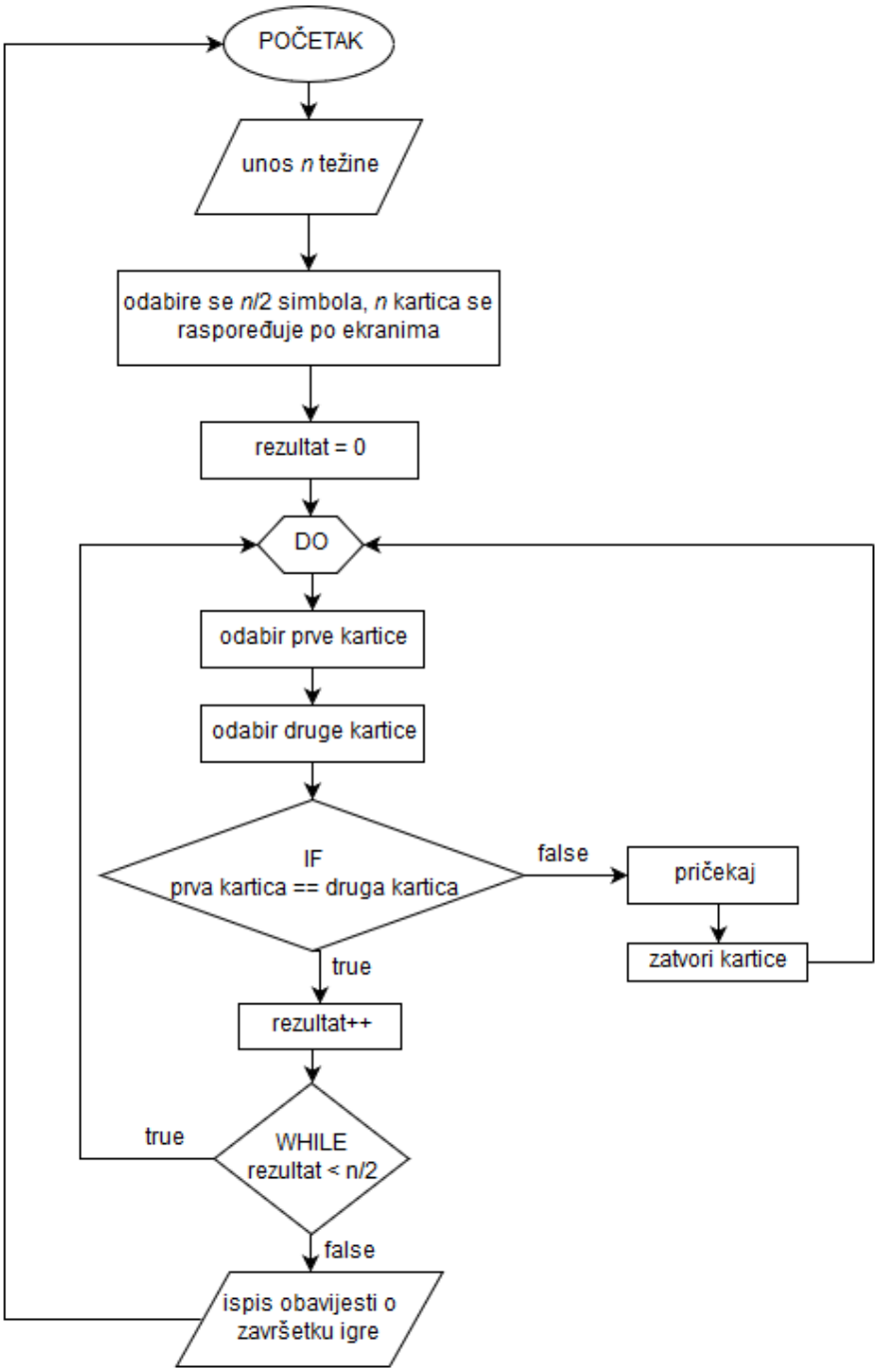
Kada se svi dijelovi spoje s Arduino UNO razvojnim sustavom, dobije se sklop čija je shema prikazana na slici 3.5. Radi bolje preglednosti, prikazani su samo prvi te početak drugog LED matričnog pokaznika, no svi pokaznici su spojeni na jednak način.



Sl. 3.5. *Shema gotovog sklopa.*

3.2. Programska podrška

Dijagram toka koji opisuje igru prikazan je na slici 3.6. Kôd je zbog svoje složenosti podijeljen na nekoliko funkcija, a i sama *loop()* funkcija može se podijeliti na više dijelova. Međutim, kôd se ugrubo može podijeliti na četiri glavna dijela: postavljanje globalnih varijabli i uključivanje biblioteka, funkciju *setup()*, funkciju *loop()* te korisnički definirane funkcije. U sljedećih nekoliko poglavlja bit će detaljno objašnjeno kako je igra izvedena programski.



Sl. 3.6. *Dijagram toka igre pamćenja.*

3.2.1. Globalne varijable i biblioteke

Na samom početku kôda nužno je uključiti potrebne biblioteke: *Wire.h* za I^2C komunikaciju, *LiquidCrystal_I2C.h* za LCD ekran, *LedControl.h* za LED matrice te *DS1307RTC.h, Time.h* i *TimeLib.h* za RTC modul. Zatim slijedi deklaracija globalnih varijabli – prvo se definiraju nožice za sve korištene module i elemente kako bi Arduino znao gdje tražiti pojedini ulaz ili gdje proslijediti određeni izlaz. Nakon toga slijede korisnički definirane varijable koje omogućuju

funkcioniranje samog programa. Varijable su definirane globalno kako bi se, za razliku od lokalnih varijabli, mogle koristiti u svim funkcijama.

3.2.2. Funckija *setup()*

Nakon osnovnih definicija varijabli i biblioteka slijedi funkcija *setup()*. U ovoj funkciji definiraju se ulazi i izlazi te postavljaju moduli. Pomoću funkcije *pinMode()* tipkala se definiraju kao ulazi. Pokreće se LCD ekran te se na njemu pomoću funkcija *setCursor()* i *print()* ispisuje tekst glavnog izbornika. Zatim se funkcijama *cursor()* i *blink()* pokreće pokazivač na ekranu kako bi korisnik znao odabrati željenu težinu. Kôd za postavljanje LCD ekrana prikazan je na slici 3.7. Također se, funkcijama *shutdown(), setIntensity()* i *clearDisplay()* postavljaju i LED matrični pokaznici, i to na način prikazan na slici 3.8.

U *setup()* funkciji također se generira i ključ (eng. *seed)* za slučajne brojeve. Budući da računalo vrijednosti definira deterministčkim algoritmima, brojevi nikad nisu potpuno slučajni, nego samo pseudoslučajni. *Seed* je pri tome broj pomoću kojega se inicijalizira generator pseudoslučajnih brojeva. Korištenje istog *seeda* pri generiranju nekog niza pseudoslučajnih brojeva, dakle, garantira da će taj niz uvijek biti jednak. S obzirom na to da se generator slučajnih brojeva u ovom programu koristi za definiranje položaja pojedine kartice, jasno je da *seed* svaki put mora biti drugačiji – kako bi se dobilo nešto poput miješanja kartica u stvarnom životu. Iz tog razloga se za argument funkcije *randomSeed()* koristi vrijeme očitano iz RTC modula. Kako bi se dobio *seed* dovoljno slučajan za potrebe ove igre, iz RTC modula se učitavaju sekunde i minute prema formuli (3-1), tj. moguće je 3600 različitih kombinacija položaja kartica.

$$seed = minute * sekunde \tag{3-1}$$

```
lcd.setCursor(0, 0);
lcd.print("Broj kartica:");
lcd.setCursor(0, 1);
lcd.print("4  8  12 16");
lcd.setCursor(0, 1);
lcd.cursor();
lcd.blink();
delay(2000);
```

Sl. 3.7. *Postavljanje LCD ekrana.*

```
led.shutdown(0, false);
led.setIntensity(0, 1);
led.clearDisplay(0);
```

Sl. 3.8. *Postavljanje prvog LED matričnog pokaznika.*

3.2.3. Funkacija *loop()*

S pokretanjem sustava korisniku je prikazan tekst *„Broj kartica: 4 8 12 16“* na LCD ekranu te se od korisnika traži odabir željenog broja kartica, odnosno težine. Kôd za odabir težine nalazi se unutar funkcije *loop()*, što je prikazano na slici 3.9.

Kôd se nalazi unutar funkcije *loop()* kako bi se cijelo vrijeme provjeravalo je li neki gumb pritisnut (provjera se izvodi funkcijom *digitalRead()*). Budući da su brojevi na LCD ekranu razmaknuti za tri mjesta, kad se pritisne gumb za lijevo ili desno, varijabla *pomak* (prvotno postavljena u vrijednost 0) se smanjuje ili povećava za 3 te se u skladu s time pomiče kursor na ekranu. Ako je kursor na lijevom ili desnom rubu ekrana, neće se pomaknuti. Funkcija *delay()* koristi se kako bi se na 500 ms odgodila ponovna provjera je li gumb pritisnut. Bez ovog koraka ne bi se dobilo željeno ponašanje gumba jer bi Arduino jedan ljudski pritisak gumba registrirao kao više pritisaka (zbog brzine kojom se funkcija *loop()* ponavlja).

Kad Arduino registrira pritisak tipke za odabir težine (srednja tipka), prelazi na sljedeći korak – postavljanje kartica, što je zasebna, korisnički definirana funkcija.

Unutar funkcije *loop()* također se nalazi i kôd koji omogućuje „treperenje“ kartica. Kako bi igrač znao odabrati željenu karticu, potrebno je imati pokazivač koji će mu dati do znanja gdje se pomicanjem po LED pokazniku trenutno nalazi. Po uzoru na pokazivač na LCD ekranu osmišljeno je da se četiri LE diode koje predstavljaju odabranu, trenutno zatvorenu karticu istovremeno „trepere“.

```
if (tezinaflag == true) {
  if (digitalRead(desno) == true && pomak != 9) {
    pomak = pomak + 3;
    lcd.setCursor(pomak, 1);
    delay(500);
  }
  if (digitalRead(lijevo) == true && pomak != 0) {
    pomak = pomak - 3;
    lcd.setCursor(pomak, 1);
    delay(500);
  }
  if (digitalRead(sredina) == true && (pomak == 0 || pomak == 3 || pomak == 6 || pomak == 9)) {
    PostaviIgru();
    tezinaflag = false;
  }
}
```

Sl. 3.9. *Odabir težine.*

Najveći dio *loop()* funkcije čini kôd za pomicanje pokazivača po LED pokazniku. Za potrebe označavanja trenutnih „koordinata“ pokaznika koriste se tri globalne varijable: *ledpomakdis, ledpomakx i ledpomaky*.

Varijabla *ledpomakx* može imati vrijednost 1 (ako je pokazivač na lijevoj strani matričnog pokaznika) ili 5 (ako je pokazivač na desnoj strani matričnog pokaznika). *Ledpomaky* može imati vrijednost 1 (ako je pokazivač na gornjoj strani pokaznika) ili 5 (ako je pokazivač na donjoj strani pokaznika). Varijabla *ledpomakdis* može poprimiti vrijednost od 0 do 3, ovisno o rednom broju pokaznika na kojoj se pokazivač trenutno nalazi.

Pri postavljanju igre *ledpomakdis* postavljena je u vrijednost 0, a *ledpomakx* i *ledpomaky* u 1, što znači da je na početku igre pokazivač na prvoj kartici prvog LED matričnog pokaznika. Također se dodjeljuje indeks trenutnim koordinatama. Svaka kartica, odnosno njen položaj na matričnom pokazniku ima svoj redni broj – gornja lijeva kartica prvog matričnog pokaznika ima indeks 0, ona ispod nje indeks 1 i tako sve do donje desne kartice na četvrtom pokazniku koja ima indeks 15. Primjeri dodjeljivanja indeksa kada se pokazivač trenutno nalazi na prvom pokazniku prikazan je na slici 3.10. Analogno se dobije kod za preostala tri pokaznika.

```
if (ledpomakdis == 0) {
  if (ledpomakx == 1) {
    if (ledpomaky == 1) {
      koordindex = 0;
    }
    if (ledpomaky == 5) {
      koordindex = 1;
    }
  }

  if (ledpomakx == 5) {
    if (ledpomaky == 1) {
      koordindex = 2;
    }
    if (ledpomaky == 5) {
      koordindex = 3;
    }
  }
}
```

Sl. 3.10. *Dodjeljivanje indeksa trenutnim koordinatama.*

Pomicanje pokazivača po matričnim pokaznicima izvodi se tipkalima. Pomicati se može ulijevo, udesno, prema gore i prema dolje. Iako za pritisak svakog tipkala, odnosno za pomicanje u svaku

stranu postoji zaseban kôd, oni su vrlo slični i iz jednog dijela kôda mogu se analogno izvesti preostali dijelovi. Potrebno je voditi računa o prelasku s jednog pokaznika na drugi, o rubnim koordinatama (npr. pokazivač ne smije ići prema gore ako je trenutno odabrana kartica gornja) te o preskakanju već otvorenih (pogođenih) kartica. Programsko rješenje za navedene probleme prikazan je na slikama 3.11. i 3.12.

Pritiskom srednjeg tipkala, ovisno o tome bira li igrač prvu ili drugu karticu, poziva se jedna od korisnički definiranih funkcija: *OdabirKartice1()*, *OdabirKartice2()* ili *ProvjeraPara()*. Funkcija *OdabirKartice1()* pozvat će se ukoliko je igrač upravo odabrao prvu karticu u trenutnom paru. Ako je odabrao drugu karticu u trenutnom paru, tj. ako je prva već odabrana, poziva se funkcija *OdabirKartice2()* nakon čega se odmah poziva i funkcija *ProvjeraPara()*.

```
if (ledpomakx == 1) {
  for (int i = 0; i < brojac + 1; i++) {
    if (ledpomakdis == koordinate[i][0] && ledpomaky == koordinate[i][2]) {
      karticaflag = true;
      break;
    }
  }
  if (karticaflag == false ) {
    ledpomakx = 5;
  }
```

Sl. 3.11. *Provjera slobodnog mjesta pri pomicanju udesno.*

```
   else if (ledpomakdis < brojdisplaya) {
     int temp = koordindex + 2;

     for (int i = 0; i < brojac+1; i++) {
       if (temp == koordinateindex[i] || temp == odabranakartical) {
         temp=temp+2;
       }
     }
     if (temp<(brojdisplaya+1)*4) {
       Pokaznik(temp);
     }
   }
```

Sl. 3.12. *Potraga za prvom slobodnom karticom.*

3.2.4. Korisnički definirane funkcije

U kôdu postoji ukupno pet korisnički definiranih funkcija: *PostaviIgru()*, *Sort()*, *OdabirKartice1()*, *OdabirKartice2()* te *ProvjeraPara()*.

Funkcija *PostaviIgru()* koristi se kako bi se kartice razmjestile po „ploči“, tj. po matričnim pokaznicima. Na početku su kartice zatvorene pa sve četiri LE diode pojedine kartice svijetle.

Također se na LCD ekranu nakratko pomoću *print()* i *setCursor()* funkcija ispisuje odabrana težina, a zatim se istim funkcijama ispisuje rezultat koji se ažurira tijekom igre.

Zatim je potrebno generirati kartice. Svaka slika bit će predstavljena jednim brojem od 0 do 9 pa se koristi generator pseudoslučajnih brojeva. Bitno je da brojevi budu različiti kako se kartice ne bi ponavljale. Kako bi se to postiglo, deklariraju se dva polja veličine deset cijelih brojeva te se u prvo polje redom spremaju brojevi od 0 do 9.

Nakon toga se pomoću generatora pseudoslučajnih brojeva generira lista kartica. U varijablu *j* spremi se nasumični broj od 0 do 9 te se zapravo radi miješanje prve liste kako bi se dobio niz nasumično poredanih brojeva. Kako se *for* petlja ponavlja samo onoliko puta koliko ima kartica u odabranoj težini, tako će se odabrati samo potreban broj kartica za tu težinu. Zatim se *lista1* kopira u globalnu varijablu *kartice[]* dva puta kako bi svaka generirana kartica imala svoj par. Cijeli taj postupak prikazan je u kodu na slici 3.13. Zatim se niz *kartice[]* izmiješa algoritmom napisanim po uzoru na *BubbleSort* algoritam za sortiranje niza, prema [9], kao što je prikazano na slici 3.14.

Time se dobije konačan raspored kartica – u polju kartice sada postoji točno onoliko jedinstvenih brojeva koliko ima parova pri odabranoj težini igre, svaki od tih brojeva ponavlja se točno dvaput i potpuno su nasumično raspoređeni.

```
for (int i = 0; i < tezina / 2; i++) {
  int j = random(10);
  int temp = listal[i];
  listal[i] = listal [j];
  listal [j] = temp;
}

for (int i = 0; i < tezina / 2; i++) {
  kartice[i] = listal[i];
  kartice[i + tezina / 2] = listal[i];
}
```

Sl. 3.13. *Generiranje kartica slučajnim brojevima.*

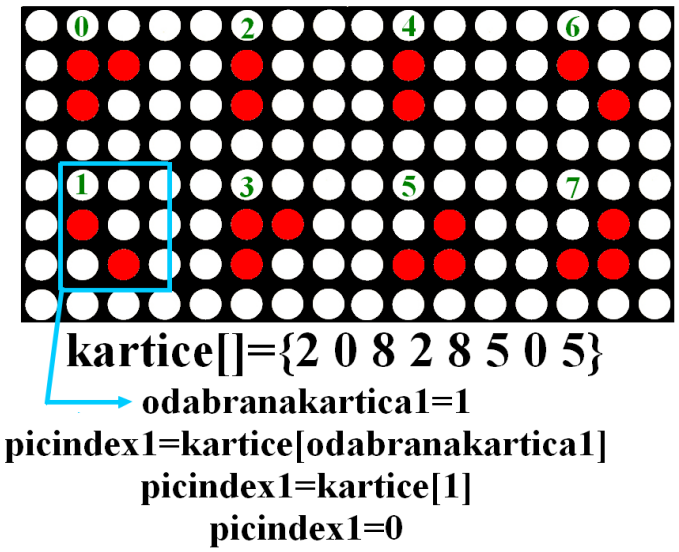
```
for (int i = 0; i < tezina; i++) {
  int j = random(i, tezina - 1);
  int temp = kartice[i];
  kartice[i] = kartice[j];
  kartice[j] = temp;
}
```

Sl. 3.14. *Miješanje kartica.*

Funkcija *Sort()* sortira po veličini polje s indeksima kartica kako bi se pri sljedećem pomicanju pokaznika mogla pronaći ispravna slobodna koordinata. Algoritam prema kojem se izvodi sortiranje zove se *Selection Sort*, prema [9]. Kad je niz poredan po veličini, vrijednost pokazivača će rasti ili padati dok ne poprimi vrijednost prve slobodne koordinate – u praksi, pokazivač će „znati“ preskočiti već otvorene kartice.

Funkcije *OdabirKartice1()* i *OdabirKartice2()* su jednake, samo što svaka ima zasebne varijable. U funkciji *OdabirKartice1()* prvo se postavlja vrijednost varijabli *odabranakartica1* na vrijednost indeksa koordinate odabrane kartice. Nakon što je postavljena varijabla *odabranakartica1*, postavlja se varijabla *picindex1* koja predstavlja indeks slike koja se nalazi na pojedinoj kartici. U varijablu *picindex1* sprema se vrijednost *kartice[odabranakartica1]*. Pri postavljanju igre, u niz *kartice[]* postavljeni su izmiješani brojevi s mogućom vrijednošću od 0 do 9, svaki dva puta. Sada se u varijablu *picindex1* sprema broj spremljen u tom nizu, na indeksu odabrane kartice. U skladu s time LE diode poprimaju unaprijed određene vrijednosti i umjesto zatvorene kartice pojavljuje se slika.

Primjerice, na slici 3.15. nalazi se primjer odabira kartice u igri s četiri para. Kartice su nasumično odabrane te čine niz *kartice[]={2 0 8 2 8 5 0 5}*. Igrač odabire karticu u donjem lijevom kutu. Ta kartica se nalazi pod indeksom 1 pa se varijabla *odabranakartica1* postavlja u vrijednost 1. Zatim se vrijednost varijable *picindex1* postavlja u vrijednost *kartice[odabranakartica1]*, tj. *kartice[1]*. Ako se pogleda niz *kartice[]*, vidi se da se pod brojem 1 nalazi vrijednost 0, što znači da se u *picindex1* sprema vrijednost 0. Pod indeksom 0 nalazi se slika označena plavom bojom na slici 3.15. pa se upravo ona pojavljuje na mjestu odabrane kartice.



Sl. 3.15. *Primjer odabira kartice.*

Nakon što je katrica otvorena, u polje s koordinatama spremaju se trenutne koordinate (*ledpomakdis, ledpomakx, ledpomaky*) kako bi se pri pomicanju pokazivača ta kartica mogla preskočiti.

Pošto su odabrane obje kartice u trenutnom pokušaju, poziva se funkcija *ProvjeraPara()* koja provjerava jesu li odabrane kartice jednake, tj. je li igrač pronašao par. Funkcija provjerava je li vrijednost varijable *picindex1* (u koju je spremljen indeks simbola prve odabrane kartice) jednaka vrijednosti varijable *picindex2* (u koju je spremljen indeks simbola druge odabrane kartice).

Ako se pri provjeri para ustanovi da *picindex1* i *picindex2* nemaju jednaku vrijednost, tj. da igrač nije pronašao par, na 1700 ms se odgodi nastavak funkcije. To se radi kako bi igrač imao vremena zapamtiti o kojim karticama se radilo. Nakon toga te dvije kartice se „zatvaraju“ –uključuju se sve četiri LE diode koje predstavljaju pojedinu karticu.

Ukoliko *picindex1* i *picindex2* imaju jednaku vrijednost, znači da su i prikazane slike jednake – igrač je pronašao par.

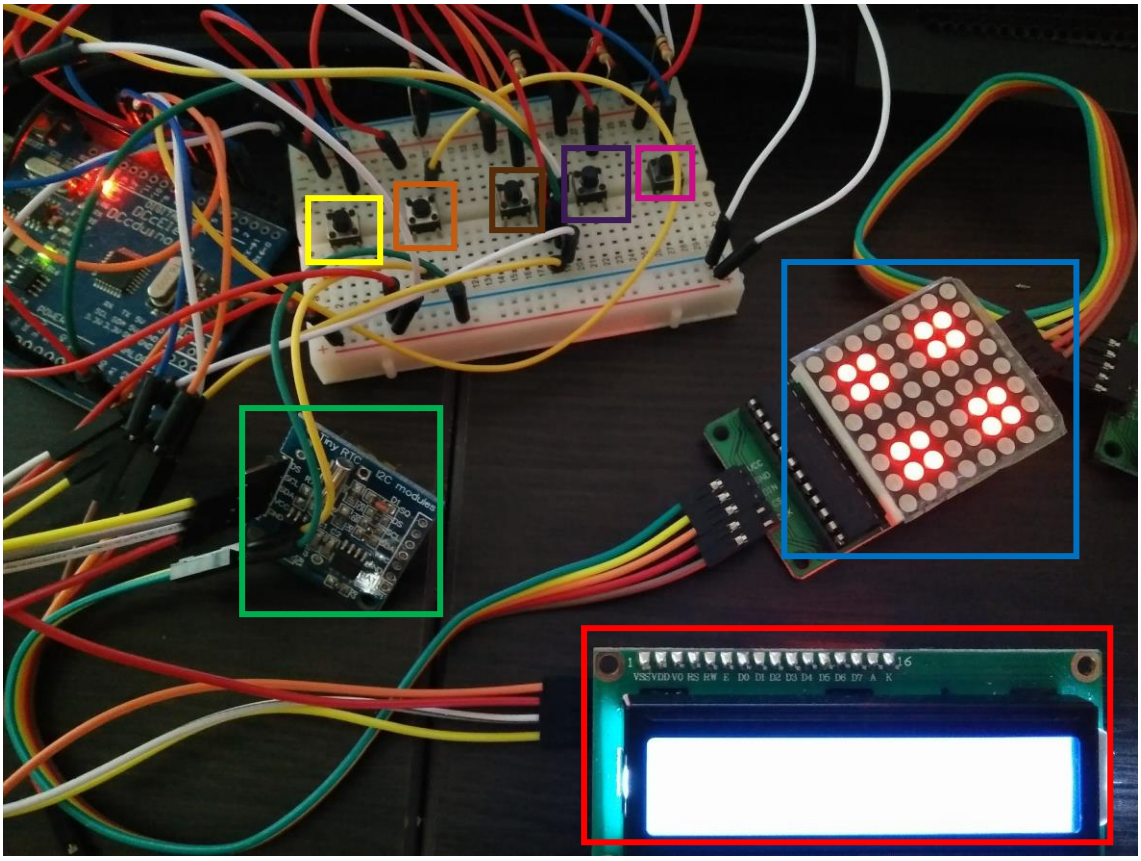
Ukoliko je pronađeni par ujedno i posljednji par na „ploči“, na LCD ekranu se ispisuje obavijest o završetku igre te se nakon tri sekunde ponovno izvodi funkcija *setup()* i sve varijable se vraćaju u početno stanje.

3.3. Testiranje

Igra je testirana od strane nekoliko osoba tijekom izrade te nakon same izrade s ciljem uočavanja nepravilnosti i grešaka u funkcioniranju. Tijekom ranih testiranja uočeno je nekoliko nepravilnosti pri pomicanju pokaznika – pokaznik bi se najčešće premjestio na mjesto gdje je već otvorena kartica i time je „zatvorio“. Pogreške su zabilježene i ispravljene u narednim verzijama kôda.

Nakon završetka izrade igra je dana na testiranje nekolicini korisnika. Testirani su različiti aspekti funkcionalnosti igre, s naglaskom na generiranje pseudoslučajnih brojeva, pomicanje pokaznika po karticama te sparivanje kartica. Također, igra je testirana na svakoj težini kako bi se osiguralo da je funkcionalna u svim slučajevima.

Konačni izgled sklopa nalazi se na slici 3.16. Na slici je crvenom bojom označen LCD ekran, plavom bojom LED matrični pokaznik, a zelenom bojom RTC modul. Tipkalo za pomicanje ulijevo označeno je žutom bojom, tipkalo za pomicanje udesno narančastom bojom, tipkalo za pomicanje prema gore ljubičastom bojom, a tipkalo za pomicanje prema gore ružičastom bojom. Smeđom bojom označeno je tipkalo za odabir težine/kartice.

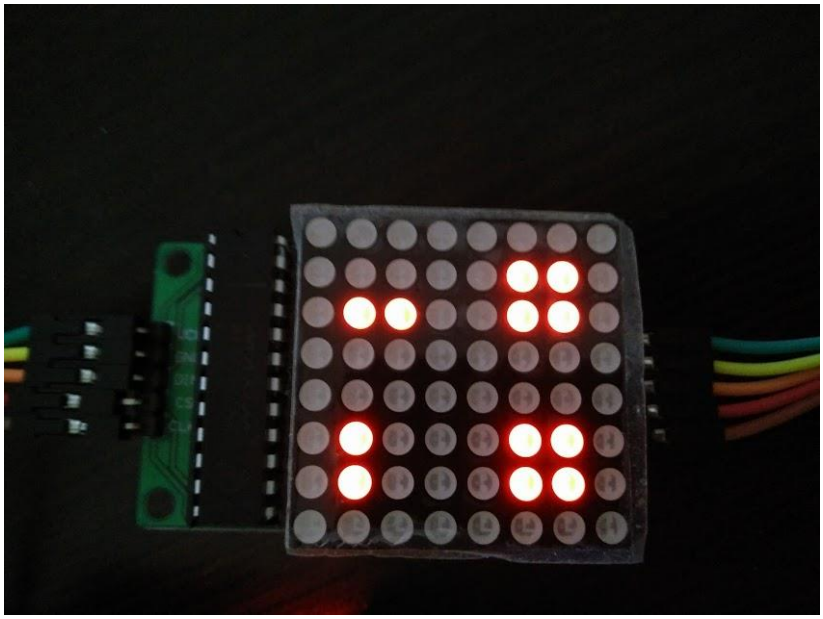


Sl. 3.16. *Izgled sustava.*

Igra je prvo pokrenuta po deset puta zaredom na svakoj od mogućih težina kako bi se utvrdilo radi li generator pseudoslučajnih brojeva kako je predviđeno. Budući da su kartice svaki puta bile drugačije raspoređene, zaključeno je da se pseudoslučajni brojevi uspješno generiraju pa su kartice dovoljno nasumične svaki put.

Kada su igrači tipkalima pomicali pokazivač po karticama, nisu uočene nepravilnosti. Pokazivač se ne pomiče izvan kartica (ni u koju stranu), tj. ako se nalazi na granici, neće se pomaknuti izvan nje. Također, pokazivač uspješno „preskače“ otvorene kartice, bila to otvorena kartica u trenutnom paru ili neka od kartica već pronađenih parova. Pokazivač uspješno preskače jednu karticu, kao i više njih. Zaključeno je da su uspješno predviđene sve kombinacije pomicanja pokaznika koje bi korisnik mogao pokušati pa pokazivač uvijek reagira kako je predviđeno.

Kada igrač pritisne tipkalo za odabir kartice, ako je u pitanju prva kartica u paru, omogućuje se odabir druge kartice. Nemoguće je ponovno odabrati istu karticu na istom mjestu te na taj način „prevariti“ igru jer se pokazivač pomiče na neko drugo, slobodno mjesto. Kad igrač odabere drugu karticu, ako te dvije kartice nisu iste, kartice se, nakon definirane stanke, zatvaraju, a rezultat ispisan na LCD ekranu se ne mijenja. Na slici 3.17. prikazan je matrični pokaznik s otvorene dvije kartice koje nisu par.

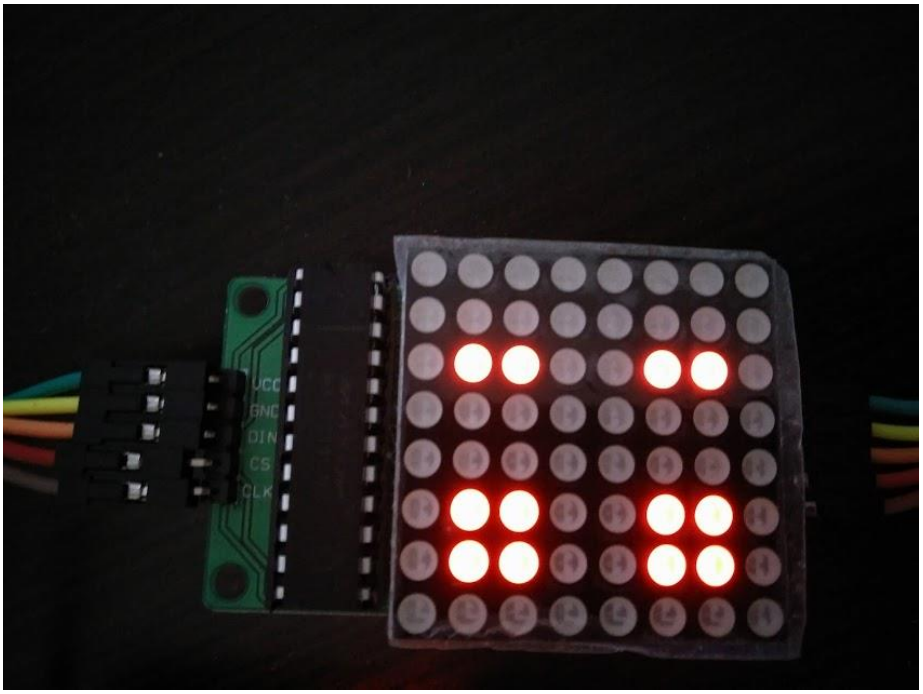


Sl. 3.17. *Otvorene kartice koje nisu par.*

Ako su dvije kartice u paru jednake, rezultat na LCD ekranu se poveća za jedan bod te kartice ostaju otvorene. Na slici 3.18. prikazan je LCD ekran s ispisom rezultata, a matrični pokaznik s dvije otvorene kartice koje jesu par prikazan je na slici 3.19. Ponovno se pokazivač pomiče na prvo slobodno mjesto i nemoguće je opet odabrati istu karticu. Kako se pri pomicanju preskaču već pogođene kartice, igrač ni u jednom stadiju biranja kartica ne može odabrati te kartice i na taj način „prevariti“ igru. Prema svemu opisanome, zaključeno je da i odabir kartica funkcionira kako je predviđeno.

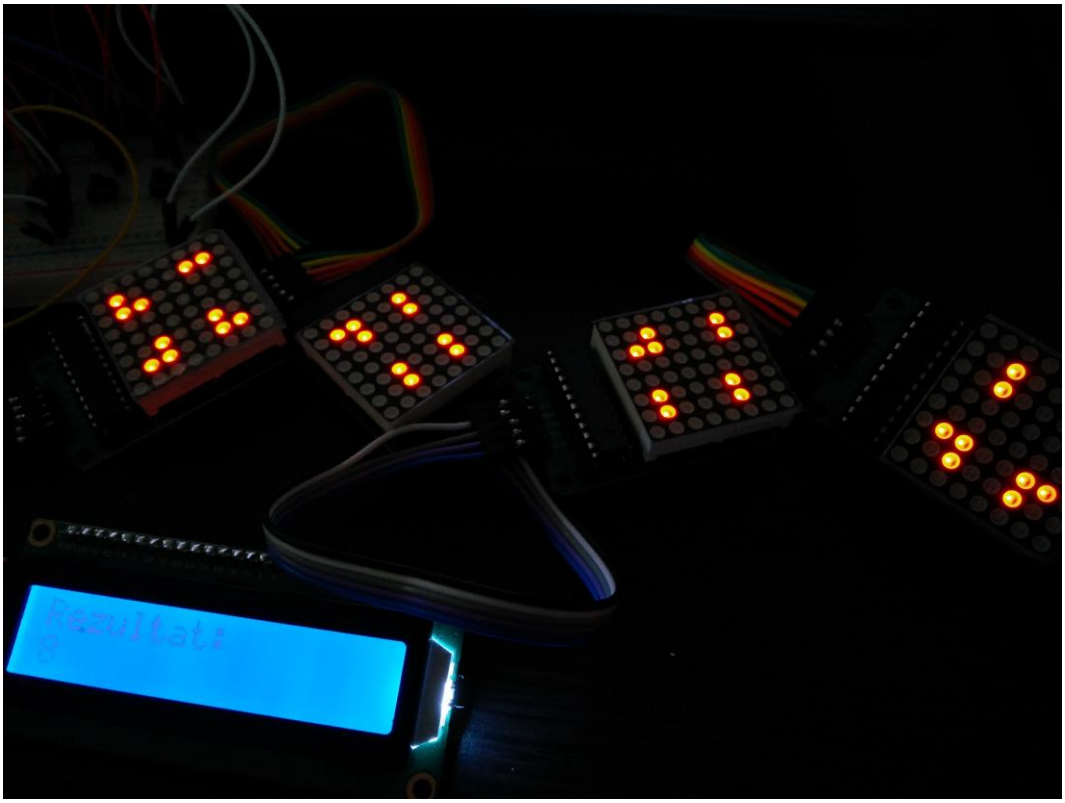


Sl. 3.18. *LCD ekran.*



Sl. 3.19. *Otvorene katrice koje jesu par.*

Dakle, nakon višestrukog testiranja igre, nisu uočene nikakve nepravilnosti pa je zaključeno kako je igra potpuno funkcionalna te radi kako je i namijenjeno. Primjer završene igre, gdje su otvorene sve kartice na najtežoj razini te je prikazan konačni rezultat, nalazi se na slici 3.20.



Sl. 3.20. *Primjer završene igre na najtežoj razini.*

4. ZAKLJUČAK

Kako bi izrada igre pamćenja bila moguća, bilo je potrebno shvatiti princip rada Arduino UNO mikroupravljačke platforme te na koji način se koristi, kao i upoznati se sa sučeljem Arduino IDE koje je zaduženo za programiranje Arduino platforme. Također je bilo bitno upoznati se s različitim modulima nužnima za funkcionalnost igre, a to su: RTC modul, LED matrični pokaznici, LCD ekran i tipkala. Iste je bilo potrebno ispravno spojiti s Arduino UNO mikroupravljačkom platformom.

Programski kod igre se, uz uključivanje biblioteka i definiranje varijabli, sastoji od tri osnovna dijela: funkcije *setup(),* funkcije *loop()* te korisnički definiranih funkcija. U funkciji *setup()* definiraju se ulazi i izlazi te postavljaju moduli. Također se generira i *seed* za slučajne brojeve. Funkcija *loop()* nadzire pomicanje tipkala po LCD ekranu i LED matričnom pokazniku. Unutar te funkcije se pozivaju korisnički definirane funkcije. Korisnički definirane funkcije koriste se za postavljanje kartica na matrične pokaznike, za odabir kartica, za provjeru jesu li odabrane kartice par te za sortiranje polja s indeksima kartica.

Testiranjem igre zaključeno je da nema pogrešaka u funkcionalnosti, odnosno da razvijeni sustav radi kako je zamišljeno. Cilj rada je postignut: napravljena je potpuno funkcionalna igra pamćenja na Arduino platformi. Igra se može unaprijediti dodatnim mogućnostima, kao što je, primjerice, dodavanje dva različita načina igre: prvi u kojem se mjeri vrijeme potrebno da se završi igra, te drugi u kojem se mjeri broj pokušaja potreban da se završi igra. Rezultati bi se mogli pratiti i shodno tome spremati „top-lista“ rezultata. Također se na vrlo jednostavan način može implementirati način igre za dva igrača. U tom slučaju pratio bi se rezultat i jednog i drugog igrača, a igrači bi se izmjenjivali u pokušaju pronalaska parova, kao i u originalnoj društvenoj igri pronalaska parova. Međutim, osnovni cilj bio je napraviti jednostavnu igru pamćenja za jednog igrača, tako da ove ideje ostaju kao ciljevi za daljnje unaprijeđenje igre.

LITERATURA

[1] Arduino 2017, Introduction, https://www.arduino.cc/en/Guide/Introduction (pristupljeno 5.7.2017.)

[2] Wikipedia, Arduino, https://en.wikipedia.org/wiki/Arduino (pristupljeno 5.7.2017.)

[3] Arduino 2017, Arduino UNO, http://www.arduino.org/products/boards/arduino-uno (pristupljeno 8.7.2017.)

[4] Arduino UNO, SPI library, https://www.arduino.cc/en/Reference/SPI (pristupljeno 8.7.2017.)

[5] Arduino 2017, Arduino UNO Rev3, https://store.arduino.cc/arduino-uno-rev3 (pristupljeno 5.9.2017.)

[6] Wikipedia, Arduino, https://hr.wikipedia.org/wiki/Arduino (pristupljeno 8.7.2017.)

[7] Wikipedia, LCD, https://hr.wikipedia.org/wiki/LCD (pristupljeno 29.8.2017.)

[8] e-radionica 2017, KKM: LCD 16x2, https://e-radionica.com/hr/blog/2015/08/19/kkm-lcd-16x2/ (pristupljeno 1.6.2017.)

[9] D. E. Knuth, The Art of Computer Programming, Volume 3: Sorting and Searching, Addison-Wesley, 1998.

SAŽETAK

Tema završnog rada izrada je igre pamćenja sparivanjem kartica na Arduino mikroupravljačkoj platformi. Igra je izrađena na Arduino UNO mikroupravljačkoj platformi, a isprogramirana pomoću Arduino IDE okruženja. Osim same Arduino platforme, koristio se RTC modul, LCD ekran, četiri LED matrična pokaznika (dimenzije 8x8) te pet tipkala. Programski kod sastoji se od dva glavna dijela. Prvi dio je postavljanje igre gdje se definira sve potrebno za igru (uključuju se biblioteke potrebne za funkcioniranje modula, definiraju ulazi i izlazi kao i varijable), a igrač odabire težinu pri čemu se na LED matričnim pokaznicima generiraju kartice generatorom pseudoslučajnih brojeva. Drugi dio je tijek igre gdje igrač tipkalima odabire po dvije kartice s LED pokaznika, a sustav provjerava jesu li te dvije kartice par. Igra traje dok igrač ne pronađe sve parove, nakon čega se ponovno pokreće. Igra je testirana te je ustanovljeno da nema nikakvih pogrešaka u radu, čime je utvrđena njena potpuna funkcionalnost – kako u smislu sklopovlja tako i programski.

Ključne riječi: Arduino, igra pamćenja, LCD ekran, LED matrični pokaznik, tipkalo

ABSTRACT

Arduino Based Memory Game

The topic of this bachelor thesis is the making of a match-pairs memory game using the Arduino microcontroller platform. The game was made on the Arduino UNO microcontroller platform and compiled using Arduino IDE. In addition, an RTC module, an LCD screen, four LED matrices (with the dimension of 8 by 8) and five push buttons were used. The code is comprised of two main parts. The first part is the game setup where everything needed for the game is defined (the inclusion of the libraries needed for the modules to function, the definition of the inputs and outputs as well as the variables), and the player can choose desired game difficulty wherein the cards are generated on the LED matrices using a pseudorandom number generator. The second part is the game itself where the player chooses two cards on the LED matrices using the push-buttons and the system checks if the two cards are a pair. The game is over when the player finds all the card pairs, whereupon it restarts. The game was play-tested and no bugs were found, which leads to conclusion that the game is fully functional – in terms of hardware as well as software.

Keywords: Arduino, memory game, LCD screen, LED matrix, push button

ŽIVOTOPIS

Iva Majić rođena je 6. lipnja 1996. godine u Đakovu, Hrvatska. Pohađa Osnovnu školu Vladimira Nazora u Đakovu do 2010., kad upisuje opći smjer Gimnazije Antuna Gustava Matoša Đakovo. Sva četiri razreda srednje škole završava s odličnim uspjehom te 2014. godine upisuje računarstvo na tadašnjem Elektrotehničkom Fakultetu Osijek, danas Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku.

Izvrsno se služi engleskim jezikom i Microsoft Office alatima. Ima srednju razinu znanja s programskim jezicima C i C++ te osnovnu razinu znanja s programskim jezikom C#. Također ima osnovnu razinu znanja s opisnim jezicima HTML i CSS.

Zahvaljujući vrlo dobrom uspjehu tijekom dosadašnjeg studiranja, dobitnica je stipendije za deficitarna zanimanja Nacionalne zaklade za potporu učeničkom i studentskom standardu za akademske godine 2014./15., 2015./16. te 2016./17. Radno iskustvo stekla je sezonskim radom na analizi kvalitete žitarica za tvrtku Inspecto d.o.o. u Đakovu te kao agentica u teleprodajnim centrima za Studio Moderna d.o.o. u Osijeku i Hrvatski Telekom d.d. također u Osijeku.

Iva Majić

PRILOZI

Rad u .doc i .pdf formatu

Programski kod u .txt i .ino formatu

Datasheet za ATmega328

Datasheet za HD44780 LCD ekran

Datasheet za PCF8574 I2C-sabirnicu (adapter za LCD ekran)

Datasheet za MAX7219 upravljač

Shema sklopa