Predmet:	**MIKROUPRAVLJAČI**
Vježba: 10	**Arduino – 4x4 matrična tipkovnica i relej**
Ishodi vježbe:	Moći povezati 4x4 tipkovnicu na mikroupravljač te upravljati sustavom za dozvolu/zabranu prolaza na temelju pina, koristeći LCD, relej i servo motor.

Priprema za vježbu:

Budući da je mikroupravljač računalo na čipu koji se programira kako bi upravljao priključenim vanjskim elektroničkim komponentama, priprema za vježbu se sastoji od dva dijela:

1. **Opis elektroničkih komponenti koje će se koristiti na LV** – proučiti tekst u uvodnom dijelu vježbe, proanalizirati i u bilježnicu ispisati najvažnije informacije za elektroničke komponente.
2. **Opis naredbi korištenih u LV** – proanalizirati programski kod za sve zadatke, ispisati nove naredbe i funkcije, objasniti njihovu namjenu i argumente. Ako ne možeš pronaći sve informacije u kodu priloženih zadataka, posluži se internetom npr. www.arduino.cc ...

Radu laboratoriju:

- Svaki zadatak treba prije prevođenja (eng. compile) pohraniti u napravljeni folder na Desktopu, tako da, u slučaju pogreške (HW, SW) imaš sačuvan kod.
- Na kraju LV, sve zadatke spremi na USB ili pošalji na svoj mail.
- Nazivi datoteka, zbog preglednosti, neka budu: LV01_ZAD01, LV01_ZAD02, itd.
- Vježbe se rade u paru, preporuka - jedan učenik spaja komponente, drugi piše programski kod, a na slijedećoj vježbi se uloge zamjenjuju.
- U zadacima koji zahtijevaju samostalno rješavanje, oba učenika sudjeluju u spajanju i programiranju.

- Za pojedini zadatak potrebno je u bilježnicu nacrtati električnu shemu s vidljivim oznakama korištenih pinova i vezu istih s oznakama u programskom kodu.
- Dobiveno rješenje treba komentirati, tj. dati zaključak što je novo u tom zadatku i kako je to riješeno, ukratko ispisati važniji dio koda (ne prepisivati cijeli kod) te navesti eventualne probleme i kako su isti riješeni.
- Ako su uz neki zadatak postoje pitanja, potrebno je u bilježnicu odgovoriti na ista.
- Ako u kodu postoji greška (negdje će biti namjerno stavljena) kòd treba korigirati i objasniti!

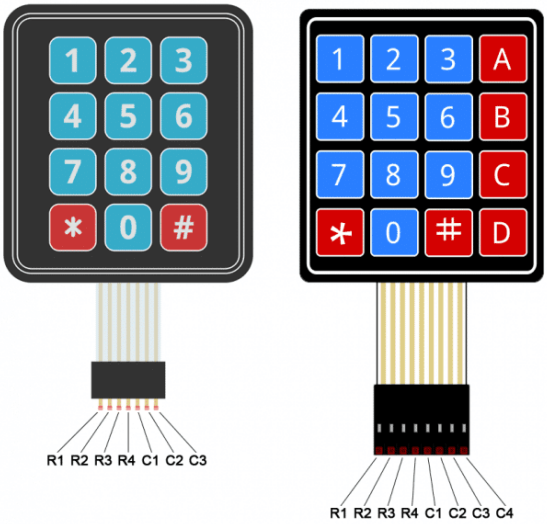
- Budući da se na vježbama koriste stvarne komponente, postoji mogućnost da je neka komponenta neispravna (pregorena LED, oštećen kontakt tipkala, prekinut vodič…). Ukoliko se sklop ponaša drugačije od očekivanog, predvidjeti i tu mogućnost i pokušati zamijeniti komponentu drugom. ***Isto vrijedni za ispitnu vježbu!***
- Prilikom spajanja, za Vcc (+5V) koristi crveni vodič, a za GND (-) crni vodič. Za ostale signale koristiti ostale boje.

- Za zadatke koje nisi stigao odraditi na vježbi, treba kod kuće razmisliti kako bi ih riješio
- Po završetku izvođenja vježbe, na temelju odrađene pripreme te riješenih zadataka, očekuje se da učenik zna odgovoriti na pitanja na kraju ovih materijala.

Pregledavanje priprema i provjeravanje znanja bit će na svakoj LV, uključujući i prethodne vježbe

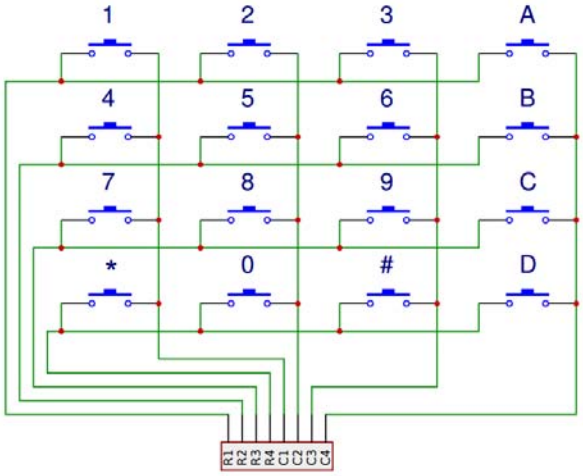
4x4 MATRIČNA TIPKOVNICA

Matrične tipkovnice su uobičajene ulazne jedinice koje mogu imati različiti broj redaka i stupaca. U laboratoriju ćemo koristiti 4x4 matričnu tipkovnicu (slika 1). Slične tipkovnice mogu se naći na bankomatima, različitim alarmnim sustavima i sl. Budući da takva tipkovnica ima 16 tipki, s uobičajenim načinom spajanja tipkala trebali bismo imati 16 pinova. Za veće tipkovnice još više.

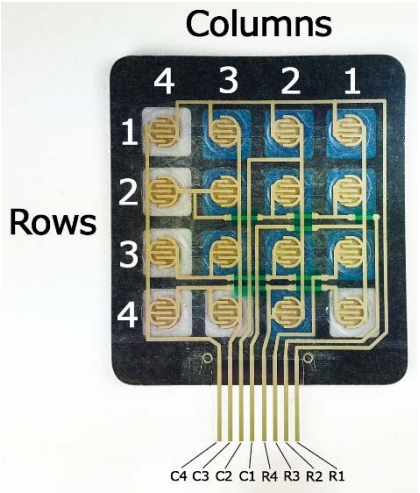


Slika x. 3x4 i 4x4 matrična tipkovnica [1]

Kako bi se smanjio broj pinova potrebnih za povezivanje tipkovnice na mikroupravljač, ovakve tipkovnice se izvode u obliku matrice koja se sastoji od stupaca i redaka. 4x4 matrična tipkovnica ima 8 linija od kojih su 4 linije za redove i 4 za stupce (slika 2).



Slika 2a. Električna shema 4x4 tipkovnice [1]



Slika 2b. Pogled sa stražnje strane 4x4 membranske tipkovnice (vodovi su zrcalni)

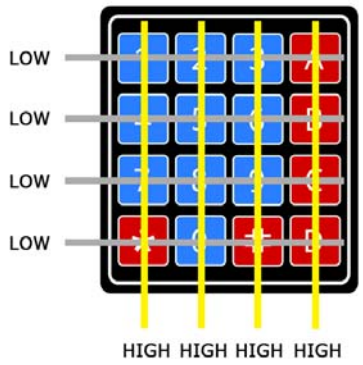
Ispod svake tipke, na sjecištu retka i stupca nalazi se membransko tipkalo koje je s jednim krajem povezan u jedan red, a s drugim krajem povezan s jednim stupcem. Preciznije, svako tipkalo u jednom retku povezano je s ostalim tipkalima u tom redu pomoću vodljivog dijela na pločici ili foliji (vodiči R1, R2, R3 i R4). Na isti način povezana su i sva tipkala u jednom stupcu – jedna strana tipkala je spojena na sve ostala tipkala u tom stupcu (C1, C2, C3, C4). Svaki se red i svaki stupac povezuje na jedan pin mikroupravljača što ukupno zahtijeva 8 pinova za 4x4 tipkovnicu.

Pritiskom na neku tipku, spajamo redak i stupac te tipke, čime omogućujemo tijek struje između pinova kojim je određen redak i stupac, odnosno spuštamo ili dižemo potencijal ulaznih pinova (ovisno koristimo li Pull-Up ili Pull-Down otpornik).

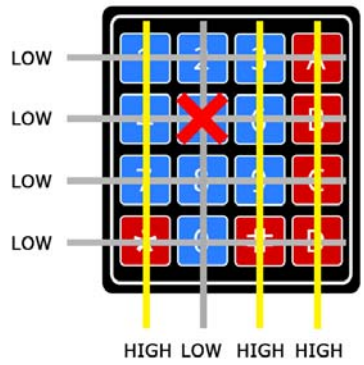
Očitanje koja je tipka pritisnuta dobiva se postupkom skeniranja. Postoji više tehnika na koji se može očitati pritisnuta tipka. U nastavku je opisana jedna tehnika.

Ardiuno određuje koja je tipka pritisnuta tako da detektira redak i stupac u kojem se nalazi pritisnuto tipkalo.

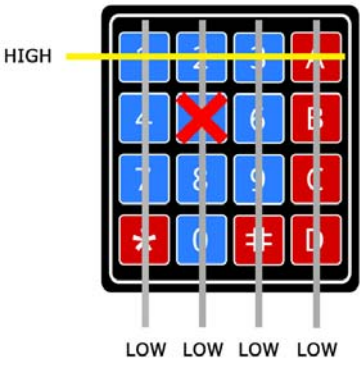
Postupak očitavanja tipke odvija se u 4 koraka [1]:



Prvo, kada niti jedno tipkalo nije pritisnuto, svi pinovi stupaca (COLUMN) postavljeni su kao ulazni pinovi (INPUT) i postavljeni su u logičku 1 radi uključenog Pull-Up otpornika. Svi pinovi redaka (ROW) postavljeni su kao izlazni pinovi (OUTPUT) i postavljeni u logičku 0 (LOW=GND).

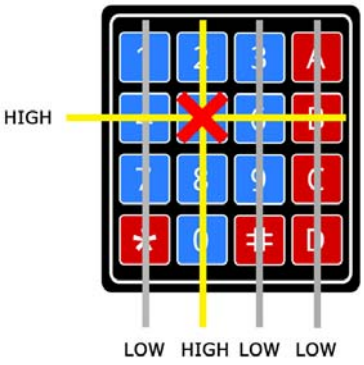


Nakon što se pritisne neka tipka (npr. tipka 5), pin tog stupca se spusti u logičku nulu (stupac 2). Sada Arduino zna u kojem je stupcu pritisnuta tipka.



Pin stupca 2 se postavlja kao ulazni, bez uključenog Pull-Up otpornika.

Određivanje stupca radi se na način da se pin za svaki redak (R1, R2, R3 i R4) pojedinačno postavlja u logičku jedinicu i prati kod kojeg se retka pin tog stupca 2 vrati u logičku 1 (HIGH).



Kad se stupac (u ovom slučaju stupac 2) vrati u logičku 1 zbog logičke 1 na pinu retka 2, Arduino je pronašao koja je tipka pritisnuta.

Različite biblioteke mogu imati različito programski riješeno skeniranje i očitavanje pritisnute tipke. Biblioteka **Keypad.h**, iako koristi isti princip skeniranja, ima programski nešto drugačije riješen način očitavanja.

Programiranje tipkovnice
Nakon povezivanja tipkovnice treba napisati programski kod kako bi tipkovnica dobila željenu funkcionalnost. Moguće je samostalno napisati kod za očitavanje tipki, ili se može koristiti gotova biblioteka, npr. **Keypad.h** [2].

1. Prvo se definira broj stupaca i redaka, koji pinovi predstavljaju koje stupce/retke te koje tipke su u kojim redcima i stupcima.
2. Nakon toga se kreira klasa Keypad koja predstavlja tipkovnicu [9]: *Keypad [tipkovnica] = Keypad(makeKeymap([znakoviTipkovnice]), [pinoviRedaka], [pinoviStupaca], [brojRedova], [brojStupaca]);*
    - znakoviTipkovnice – dvodimenzionalni char niz koji predstavlja znakove koji se nalaze na tipkovnici (npr. char znakovi[2][2] = { {'1', '2'}, {'3', '4'} })
    - pinoviRedaka – jednodimenzionalno polje koje predstavlja pinove na koje su redci tipkovnice spojeni (npr. byte rowPins = { 10, 11,12,13 })
    - pinoviStupaca – jednodimenzionalno polje koje predstavlja pinove na koje su stupci tipkovnice spojeni (npr. byte columnPins = { 6,7,8,9 })
    - brojRedova – broj redaka na tipkovnici
    - brojStupaca – broj stupaca na tipkovnici

Funkcija **[tipkovnica].getKey()** vraća simbol trenutno pritisnutog gumba. Ako nije pritisnuta nijedna tipka, vraća se 0 u ASCII kodu [9].

U nastavku su još neke od funkcija biblioteke Keypad.h [3].
Više na https://playground.arduino.cc/Code/Keypad/

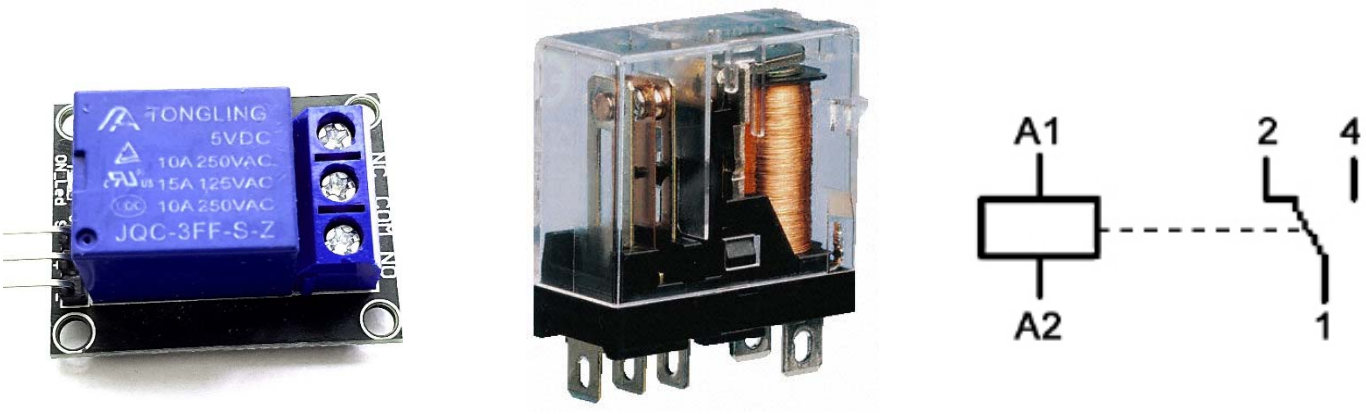
- **begin(makeKeymap(userKeymap))** - inicijalizira internu mapu ključa da bude jednaka korisničkom ključu.
- **waitForKey()** – Ova funkcija čeka dok se ne pritisne tipka (Napomena: Blokira sve ostatak koda dok ne se tipka ne pritisne.)
- **getKey()** – Vraća ime pritisnute tipke, ako postoji. Neblokirajuća funkcija.
- **KeyState getState()** – Vraća trenutačno stanje tipki. 4 stajna su: IDLE, PRESSED, RELEASED i HOLD.
- **keyStateChanged()** – Vraća informaciju kada se ključ promijenio iz jednog u drugi status.
- **setHoldTime(unsigned int time)** – Postavlja vrijeme za koje će korisnik morati držati tipku kako bi ona ušla u HOLD način rada. Upis je u milisekundama.
- **setDebounceTime(unsigned int time)** – Postavite vrijeme koje će trebati da tipkovnica opet počne prihvaćati novi upis. Upis je u milisekundama.
- **addEventListener(keypadEvent)** – pokrenuti događaj ako se koristi tipkovnica

Više informacija o matričnoj tipkovnici na slijedećim linkovima:
https://playground.arduino.cc/Code/Keypad
https://playground.arduino.cc/Main/KeypadTutorial

https://www.youtube.com/watch?v=4zAmzCTN20k
https://www.youtube.com/watch?v=bNOVg9vwFRM

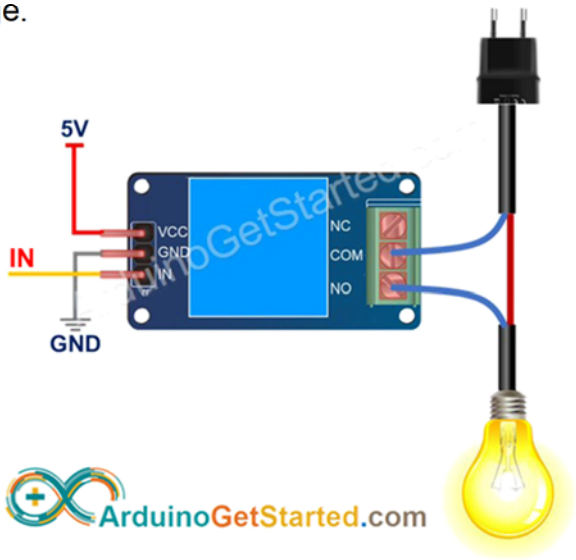
RELEJ

Relej je elektromehanička električna sklopka pomoću koje se s manjim naponom i strujom može upravljati većim strujnim krugovima koji rade na većim naponima i gdje teče veća struja. Na primjer, ako želimo pomoću Arduina uključivati i isključivati rasvjetu spojenu na 230V, to možemo ostvariti pomoću releja (slika 1).



Slika 1. Primjeri elektromehaničkih releja i simbol releja s izmjeničnim kontaktima [6]

Relej se povezuje na dva električna kruga: jedan krug je upravljački krug koji se priključuje na mikroupravljač (manja struja i napon) i drugi krug kojim želimo upravljati na koji se spaja trošilo većeg radnog napona i struje.



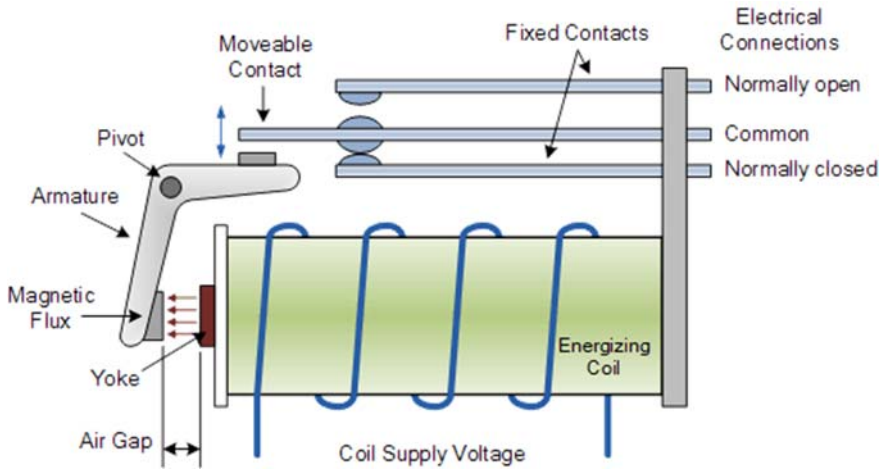
Slika 2. Primjena releja – uključivanje strujnog kruga većeg napona i struje [prilagođeno 7]

Relej je zapravo sklopka, samo što umjesto da rukom uključimo ili isključimo trošilo (npr. svjetlo, motor i sl.) to će napraviti mikroupravljač – primjerice na temelju mjerenja osvjetljenja u prostoriji. Simbol releja s izmjeničnim kontaktima prikazan je na slici 1.

Relej se sastoji od zavojnice koja proizvodi magnetsko polje kad kroz nju teče struja. Magnetsko polje će mehanički privući polugu koja će pomaknuti pokretni kontakt i na taj način zatvoriti strujni krug (slika 3).

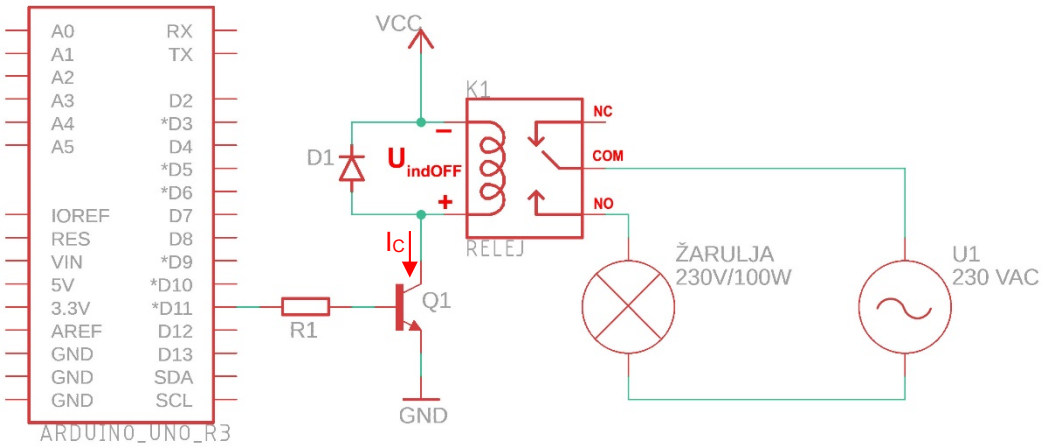
Releji se često izvode tako da osim jednog zajedničkog kontakta (COM), imaju dva kontakta - mirni kontakt i radni kontakt. Mirni kontakt je onaj koji je zatvoren kad je relej isključen, odnosno kad kroz zavojnicu ne teče struja. Radni kontakt je u tom trenutku otvoren. Kad se kroz zavojnicu pusti struja, tada se zajednički kontakt prebaci u drugi položaj i sada je mirni kontakt otvoren, a radni kontakt zatvoren. (slike 1 i 3).

Engleski izraz za mirni kontakt je *Normally Closed* (NC), a za radni kontakt je *Normally open* (NO). Zajednički kontakt ima oznaku *Common* (COM).



Slika 3. Dijelovi releja i kontakti releja [6]

Digitalni sklop ili u našem slučaju mikroupravljač, često ne može dati dovoljnu struju za uključenje releja, pa se u takvom slučaju koristi tranzistorska sklopka. Izlaz digitalnog sklopa spojen je tada preko otpornika na bazu tranzistora, ako se radi o bipolarnom tranzistoru, ili bez otpornika (ili eventualno otpornik malog otpora) na Gate, ako se radi o unipolarnom tranzistoru (slika 4).



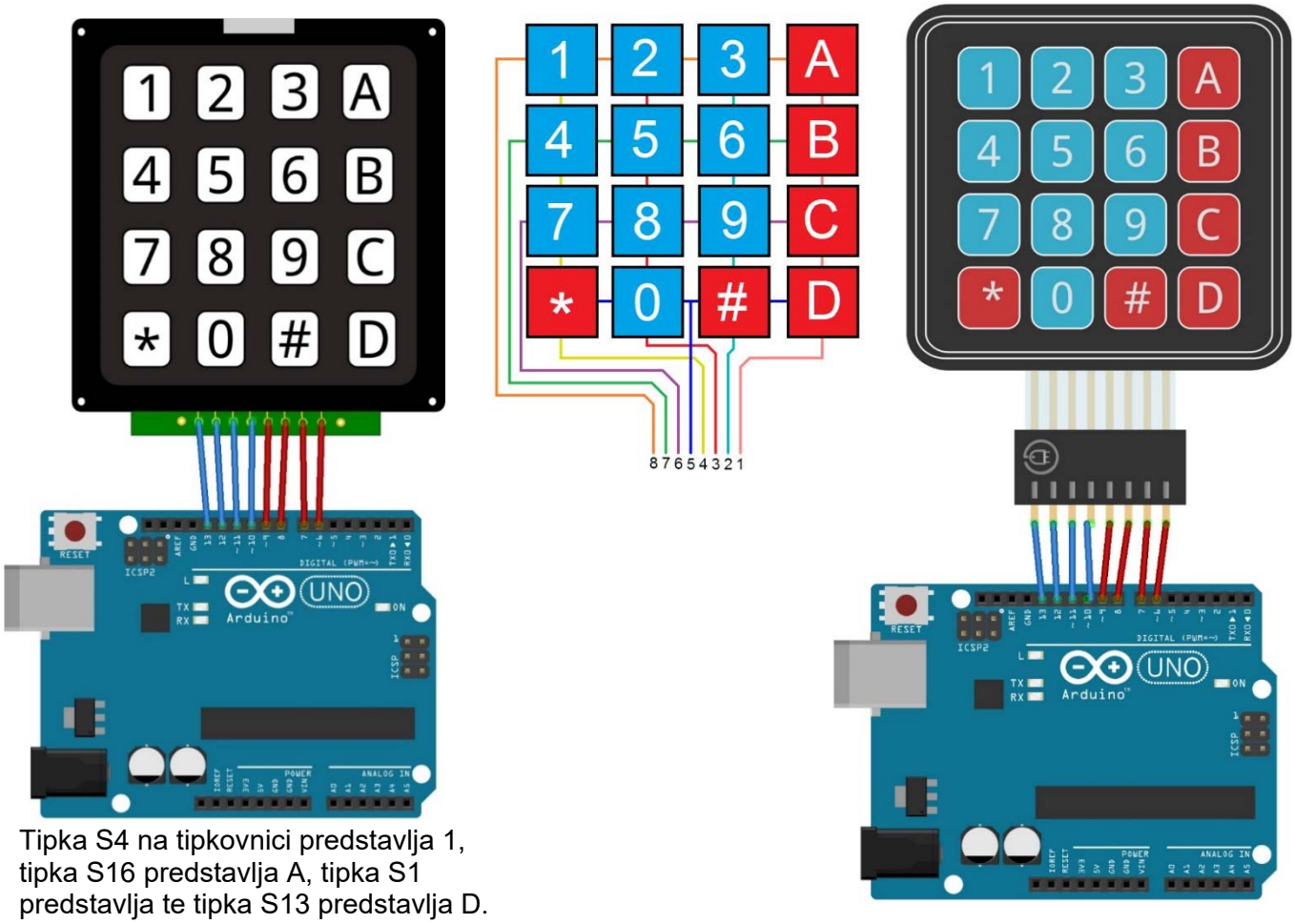
Slika 4. Povezivanje releja na mikroupravljač preko tranzistorske sklopke.

Kad je na izlazu D11 logička 1 (5V), tranzistor vodi struju, kroz zavojnicu teče struja i stvara se magnetsko polje koje zatvara mirni kontakt releja (NO). Kad je na izlazu D11 logička 0 (GND) tranzistor ne vodi struju, nema magnetskog polja, pa je mirni kontakt otvoren. Kolektor tranzistora može biti spojen na 5V ili na veći napon, npr 12 V.

Dioda D1 služi za zaštitu tranzistora od induciranog prenapona prilikom isključivanja tranzistora. Isto kao i kod isključenja motora, zavojnica je induktivno trošilo koje se suprotstavlja prestanku toka struje, pa se prilikom isključenja tranzistora na njoj inducira veliki naponski šiljak U_{indOFF} od nekoliko desetaka V. Polaritet napona je takav da je **+** potencijal na strani tranzistora (*Drain*), a **–** na strani Vcc. Iz tog razloga se ugrađuje zaštitna dioda D1, koja ograničava taj napon na približno 0,7V – koliko je napon na njoj dok vodi struju. U normalnom radu, dioda D1 je reverzno polarizirana i kroz nju ne teče struja (oscilogram u vježbi MU-LV09 - Arduino - DC i Servo motor, slika 5).

Jednokanalna relejna pločica koju koristimo na laboratorijskim vježbama (slika 1) ima osim releja na sebi ugrađene i komponente R1, Q1 i D1.

Zadatak 1. Spoji 4x4 tipkovnicu prema slici te napiši program koji će ispisivati vrijednosti pritisnutih tipki na Serial monitor.



Tipka S4 na tipkovnici predstavlja 1, tipka S16 predstavlja A, tipka S1 predstavlja te tipka S13 predstavlja D.

Plavi vodovi spajaju RETKE (ROW) tipkovnice, dok crveni vodovi spajaju STUPCE (COLUMN) tipkovnice.

Kòd zadatka

```
#include <Keypad.h> // Dodavanje Keypad.h biblioteke

const byte ROWS = 4; // Definiranje broja redaka tipkovnice
const byte COLS = 4; // Definiranje broja stupaca tipkovnice

char keys[ROWS][COLS] = {  // U dvodimenzionalno polje definirati
      { '1','2','3','A' },  // oznaku svake tipke
      { '4','5','6','B' },
      { '7','8','9','C' },
      { '*','0','#','D' },
};

byte rowPins[ROWS] = { 13, 12, 11, 10 };    // Definiramo pinove na koji
                                            // su spojeni redci - provjeri!

byte colPins[COLS] = { 9, 8, 7, 6 };        // Definiramo pinove na koji
                                            // su spojeni stupci - provjeri!
//Kreiranje tipkovnice
Keypad kpd = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);
```

```
void setup()
{
      Serial.begin(9600);
}

void loop()
{
      char key = kpd.getKey();   // Očitanje pritisnute tipke
      if (key) {                 //Ako je registrirana pritisnuta tipka ispiši
njenu vrijednost
            Serial.println(key);
      }
}
```

Zadatak 2. Dodaj LCD zaslon i doradi program prema predlošku u nastavku tako da se upisom ispravne šifre, vrata otključaju na 5 sekundi.

a) Dok uređaj čeka na unos pina, na LCD zaslonu u prvom retku ispisati poruku „Unesite pin:“.
b) Pritiskom na pojedinu tipku u nastavku 1. retka ispisivati zvjezdicu („*„).

Primjer izgleda LCD zaslona pogledaj u zadatku 4. Konačna električna shema dana je u zadatku 7.

Kòd zadatka

```
#include <Keypad.h> // Dodavanje Keypad.h biblioteke

const byte ROWS = 4; // Definiranje broja redaka tipkovnice
const byte COLS = 4; // Definiranje broja stupaca tipkovnice

char keys[ROWS][COLS] = {  // U dvodimenzionalno polje definirati
      { '1','2','3','A' },  // oznaku svake tipke
      { '4','5','6','B' },
      { '7','8','8','C' },
      { '#','0','*','D' },
};

byte rowPins[ROWS] = { 13, 12, 11, 10 };   // Definiramo pinove na koji
                                           // su spojeni redci - provjeri!

byte colPins[COLS] = { 9, 8, 7, 6 };       // Definiramo pinove na koji
                                           // su spojeni stupci - provjeri!

char kombinacija[] = { '1','2','3','A' };  // Željena kombinacija za
                                           // otključavanje

//Polje u koje upisujemo pritisnute tipke
char kombinacija1[sizeof(kombinacija) / sizeof(char)];

int i = 0, j = 0;

//Kreiranje tipkovnice
```

```
Keypad kpd = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup()
{
      Serial.begin(9600); //Inicijalizacija serijskog porta na brzini od 9600
}

void loop()
{
char key = kpd.getKey();

if(key) { //Ako je  pritisnuta tipka registrirana ispiši njenu vrijednost
      Serial.println(key);
      kombinacija1[i] = key;
      if(kombinacija[i] == kombinacija1[i]){  // „Algoritam“ za šifru
            i++;
            j++;
            if(j == sizeof(kombinacija)/sizeof(char)){
            Serial.println("Vrata Otkljucana!");
            i = 0;
            j = 0;
            }
      }
      else{
      i = 0;
      j = 0;
      }
      }
}
```

Zadatak 3. Proširi funkcionalnost prethodnog zadatka:

c) Otključana vrata simuliraj ispisivanjem poruke „Vrata otključana“ u drugom retku LCD-a.
d) U slučaju neispravne šifre, u drugom retku ispisati „Neispravan PIN!“, pričekati 3 sekunde i nakon toga obrisati taj tekst i čekati unos novog pina.

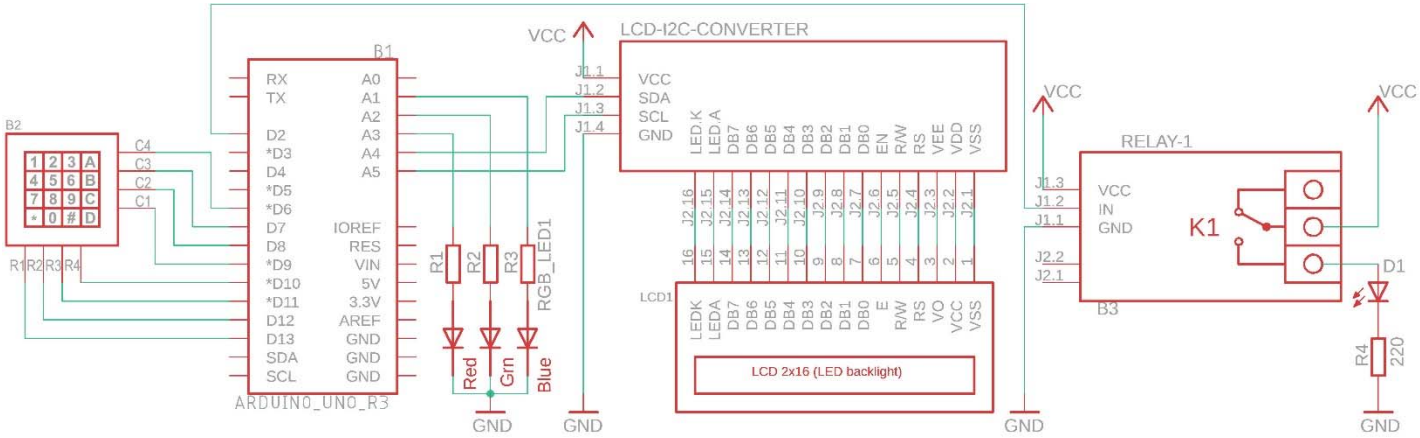
Zadatak 4. Proširi funkcionalnost prethodnog zadatka:

e) U drugom retku u desnom kutu odbrojavati vrijeme u sekundama 5, 4, 3, 2, 1.

Zadatak 5. Modificiraj spoj i program iz prethodnog zadatka dodavanjem RGB LED diode. Pomoću RGB LED diode treba davati informaciju o statusu sustava: plava boja znači „čekam unos“, zelena boja „brava otvorena“, crvena boja „neispravan pin/zabranjen prolaz“.

Zadatak 6. Modificiraj spoj i program iz prethodnog zadatka dodavanjem releja. Relej treba uključiti električnu bravu u trajanju 5 sekundi, tijekom odbrojavanja na LCD zaslonu. U energetski krug releja priključiti LED diodu prema shemi u zadatku 7.

Zadatak 7. Modificiraj spoj i program iz prethodnog zadatka tako da dodaš servo motor koji će simulirati zasun. Kad se vrata otključaju, servo motor se treba u zakrenuti za 90 stupnjeva. Taj položaj treba zadržati 5 sekundi. Nakon toga, servo motor se treba vratiti u početni položaj (0 stupnjeva). Konačna električna shema dana je u nastavku:



Zadatak 8. Doradi program tako da u slučaju netočnog pina, dozvoliš još jedan pokušaj. Dakle ukupno dva pokušaja. Nakon toga ispisati odgovarajuću poruku i uključiti crvenu LED diodu.

Zadatak 9. Doradi sustav tako da omogućiš dva ili više korisnika, od kojih svaki ima svoj pin vezan uz svoje ime. Korisnike organizirati tako da koristiš dvodimenzionalno polje ***korisnik[pin, ime]***. Sustav treba prema pinu prepoznati korisnika i u drugom retku umjesto slobodan prolaz ispisati „Pozdrav ***ime***“.

Zadatak 10. Koristeći 4x4 matričnu tipkovnicu i LCD, konstruiraj kalkulator s funkcijama „+“. „–“ , „*“ i „/“ [4]. Osim brojeva, u nastavku se nalaze funkcije pojedinih tipki:

- “A” - Zbrajanje (+)
- “B” - Oduzimanje (-)
- “C” - Množenje (*)
- “D” - Dijeljenje (/)
- “*” - Brisanje (C)
- “#” - Jednako (=)

Po potrebi, modificiraj programski kod kako bi korektno radio sa I2C LCD-om.

```
/*
    © Techtronic Harsh [4], [5].
*/

#include <Keypad.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(0, 1, 2, 3, 4, 5);
const byte ROWS = 4;
const byte COLS = 4;

char keys[ROWS][COLS] = {
  {'1', '2', '3', '+'},
  {'4', '5', '6', '-'},
  {'7', '8', '9', '*'},
  {'C', '0', '=', '/'}
};
```

```
byte rowPins[ROWS] = { 13,12,11,10 };
byte colPins[COLS] = { 9,8,7,6 };

Keypad myKeypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

boolean presentValue = false;
boolean next = false;
boolean final = false;
String num1, num2;
int answer;
char op;

void setup()
{
    lcd.begin();
    lcd.setCursor(0, 0);
    lcd.print("Techtronic Harsh");
    lcd.setCursor(0, 1);
    lcd.print("   Calculator");
    delay(3000);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("    Like And");
    lcd.setCursor(0, 1);
    lcd.print("  Subscribe Us");
    delay(3000);
    lcd.clear();
}

void loop() {

    char key = myKeypad.getKey();

    if (key != NO_KEY && (key == '1' || key == '2' || key == '3' || key ==
'4' || key == '5' || key == '6' || key == '7' || key == '8' || key == '9' ||
key == '0'))
    {
        if (presentValue != true)
        {
            num1 = num1 + key;
            int numLength = num1.length();
            lcd.setCursor(15 - numLength, 0);    //to adjust one whitespace
                                                 // for operator
            lcd.print(num1);
        }
        else
        {
            num2 = num2 + key;
            int numLength = num2.length();
            lcd.setCursor(15 - numLength, 1);
            lcd.print(num2);
            final = true;
```

```
        }
    }

    else if (presentValue == false && key != NO_KEY && (key == '/' || key ==
'*' || key == '-' || key == '+'))
    {
        if (presentValue == false)
        {
            presentValue = true;
            op = key;
            lcd.setCursor(15, 0);
            lcd.print(op);
        }
    }

    else if (final == true && key != NO_KEY && key == '=') {
        if (op == '+') {
            answer = num1.toInt() + num2.toInt();
        }
        else if (op == '-') {
            answer = num1.toInt() - num2.toInt();
        }
        else if (op == '*') {
            answer = num1.toInt() * num2.toInt();
        }
        else if (op == '/') {
            answer = num1.toInt() / num2.toInt();
        }
        lcd.clear();
        lcd.setCursor(15, 0);
        lcd.autoscroll();
        lcd.print(answer);
        lcd.noAutoscroll();
    }
    else if (key != NO_KEY && key == 'C') {
        lcd.clear();
        presentValue = false;
        final = false;
        num1 = "";
        num2 = "";
        answer = 0;
        op = ' ';
    }
}
```

Pitanja za provjeru znanja:

1. Na koji način se izvodi 4x4 matrična tipkovnica?
2. Kako se povezuje 4x4 matrična tipkovnica na mikroupravljač?
3. Koja je razlika između 3x4 i 4x4 matrične tipkovnice?
4. Objasni princip skeniranja 4x4 matrične tipkovnice kako bi se dobila informacija o pritisnutoj tipki.
5. Što je relej i čemu služi?
6. Nacrtaj i objasni dva kruga releja!
7. Od kojih dijelova se sastoji relej?
8. Nacrtaj i objasni princip rada releja?
9. Objasni mirni i radni kontakt?
10. Objasni uporabu tranzistora za povezivanje releja na mikroupravljač!
11. Čemu služi zaštitna dioda D1?
12. Nacrtaj graf napona na kolektoru tranzistora bez zaštitne diode i sa zaštitnom diodom te objasni pojave.
13. Koje tehničke podatke si pronašao na kućištu releja i što oni znače?

LITERATURA:

1. Circuit Basics, How To Set Up A Keypad On An Arduino, https://www.circuitbasics.com/how-to-set-up-a-keypad-on-an-arduino/, (pregledano 13.03.2021.)
2. Arduino.cc stranica, Keypad.h, https://playground.arduino.cc/Code/Keypad , (pregledano 13.03.2021.)
3. Arduino.cc stranica, The Matrix Keypad how-to, https://playground.arduino.cc/Main/KeypadTutorial/ , (pregledano 13.03.2021.)
4. Techtronic Harsh, Arduino Calculator Using 4X4 Keypad and 16X2 LCD, http://techtronicharsh.com/2020/03/18/arduino-calculator-using-4x4-keypad-and-16x2-lcd-display/, (pregledano 13.03.2021).
5. Instructables circuits, Arduino Calculator Using 4X4 Keypad, https://www.instructables.com/Arduino-Calculator-Using-4X4-Keypad/ , (pregledano 13.03.2021).
6. Automatika, Releji, https://www.automatika.rs/baza-znanja/teorija-upravljanja/releji.html, (pregledano 13.03.2021.)
7. Ardiuno Get Started, Arduino – Relay, https://arduinogetstarted.com/tutorials/arduino-relay, (pregledano 13.03.2021.)