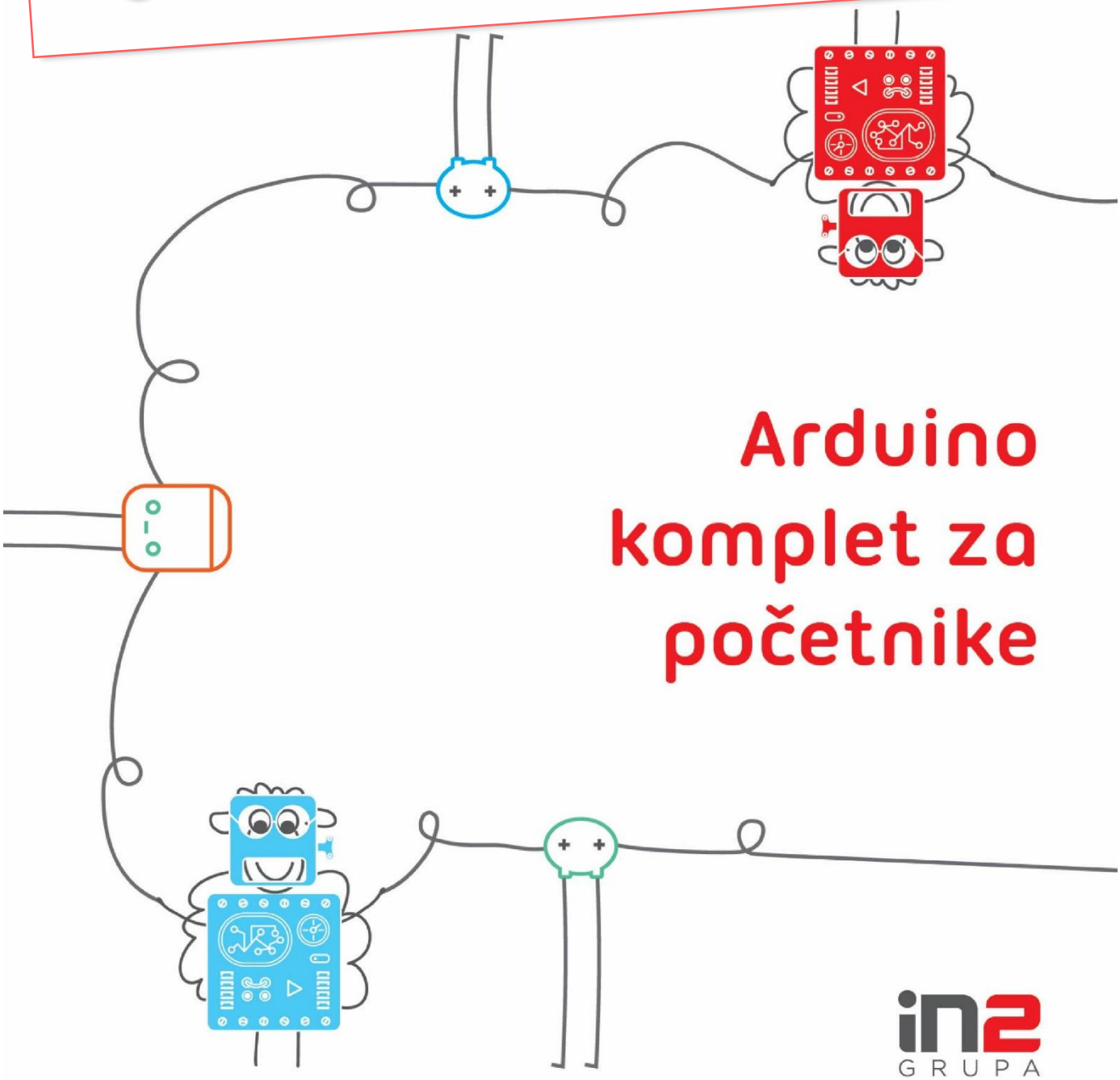
LABUS

LABORATORIJ ZA UČENIKE
SREDNJIH I OSNOVNIH ŠKOLA

U suradnji s tvrtkom IN2!



Arduino komplet za početnike

in2 GRUPA

Arduino komplet za početnike

Sadržaj

Uvod

Arduino komplet za početnike je potpuni paket koji sadrži sve komponente i senzore koji su potrebni za savladavanje osnova Arduino razvojne pločice. Cilj ovog priručnika jest da vam pokaže neke od najčešćih metoda pri izradi vlastitog projekta.

Sadržaj kompleta jest

No.	Oprema	Senzori	
1	Infiniduino Uno R3	Šifra modula	Opis modula
1	Pločica za proširivanje U/I sučelja	KY-001	Temperaturni senzor (1)
1	Breadboard	KY-002	Vibracijski senzor
1	Mali breadboard	KY-003	Hall-efekt senzor
10	R LE dioda	KY-004	Tipkalo
10	G LE dioda	KY-005	Infracrveni odašiljač
10	Y LE dioda	KY-006	Pasivni piezo (buzzer)
1	RGB LE dioda	KY-008	Laserski odašiljač
5	Tipkalo	KY-009	SMD RGB LE dioda
20	330 ohm, 1k ohm, 10k ohm otpornici	KY-010	Opto-coupler
1	1k ohm potenciometar	KY-011	RG LE dioda
1	10k ohm potenciometar	KY-012	Aktivni piezo (buzzer)
1	40 Pin letva (muška) - zglob	KY-013	Temperaturni senzor (2)
1	40 Pin letva (muška) - ravno	KY-015	Senzor temperature / vlage
1	5516 LDR – foto-otpornik	KY-016	RGB LE dioda
1	Kuglični prekidač	KY-017	Optički nagibni prekidač
1	Aktivni piezo	KY-018	Foto-otpornik
1	Pasivni piezo	KY-019	Relej (+5V)
1	74HC595 - Posmični registar	KY-020	Nagibni prekidač
1	MAX7219 - LED driver	KY-021	Maleni magnetski prekidač (Reed)
1	LM35 - Temperaturni senzor	KY-022	IR prijemnik
1	SS8050 - NPN tranzistor	KY-023	Komandna palica (Joystick)
1	IR odašiljač (daljinski upravljač)	KY-024	Linearni Hall-efekt senzor
1	IR prijemnik	KY-025	Magnetski prekidač (Reed)
1	IR foto-tranzistor	KY-026	Senzor plamena
1	IR daljinski upravljač	KY-027	Nagibni prekidač sa LE diodom
1	Koračni motor	KY-028	Temperaturni senzor (3)
1	Servo motor	KY-029	RG LE dioda
1	16x2 LCD ekran	KY-031	Senzor kucanja
1	Jednoznamenkasti 8-seg. displej	KY-032	Infracrveni senzor za koliziju

1	Četveroznamenkasti 8-seg. Displej	KY-033	Analogni Hall-efekt senzor
1	8x8 LED Matrični displej	KY-034	LE dioda
1	Upravljač koračnog motora	KY-035	Analogni Hall-efekt senzor
1	Mikrofon	KY-036	Senzor dodira
1	Upravljačka palica	KY-037	Mikrofon (veća osjetljivost)
1	Sat u realnom vremenu (RTC)	KY-038	Mikrofon
1	Relej	KY-039	Detektor otkucaja srca
1	HC-SR04 – Ultrasonični senzor udaljenosti		
1	HC-SR501 - Pasivni IR senzor		
1	RC-522 – RFID čitač		
1	S50 - RFID Oznaka (kartica)		
1	S50 - RFID Oznaka (privjesak)		
65	M-M Kratkospojnik		
10	Ž-Ž kratkospojnik		
10	M-Ž kratkospojnik		
1	9V/1A – Napajanje		

Prije nego krenemo

Za svaku lekciju koristit ćemo Infiniduino Uno R3 razvojnu pločicu, ali za uspješno izvršavanje tih lekcija funkcionirati će i originalne Arduino razvojne pločice kao i druge kompatibilne third-party pločice (SainSmart, Teensy itd.).

Lekcije su zamišljene tako da vas upoznaju na najlakši način principa rada komponenata u Arduino kompletu dok se njihova moguća i proširena upotreba upoznaje preko dodatnih zadataka čiji je uz ovaj cilj da vas nauči kako razmišljati pri izradi programske potpore za projekte.

Lekcija 1 – Arduino općenito

***Ovu lekciju nije potrebno proći ako imate iskustva sa Arduino razvojnom pločicom**

Arduino je trenutno jedna od najpopularnijih otvorenih razvojnih platformi na svijetu. Arduino je dobar izbor za sve one koji nisu profesionalni elektronički dizajneri a žele sami kreirati kompleksne i zapanjujuće projekte.

Za više informacija o Arduino platformi posjetite službenu Arduino stranicu(Poveznice se nalaze u dodatku).

U ovoj lekciji ćemo proći kroz osnovni proces pisanja arduino programa te programiranja same razvojne pločice. Arduino platforma koristi verziju C jezika zvanu Objective C koja se sintaksno ne razlikuje od C programskog jezika. Više o programiranju u Arduino platformi možete vidjeti u službenoj dokumentaciji.

Za izvedbu ove lekcije (i svih slijedećih) potrebni su vam:

- Infiniduino R3 razvojna pločica (Arduino) i pripadajući USB kabel
- Računalo sa serijskim pristupom (USB) i instaliranim Arduino razvojnim okruženjem (kako skinuti i instalirati Arduino možete vidjeti na službenoj stranici za programsku podršku)
    - **Pažnja:** pri prvom spajanju na računalo arduino će morati instalirati FTDI komunikacijske drivere što zahtijeva odobrenje administratora računala.

*Pri navođenju pribora za sve daljnje vježbe neće biti navedene komponente iz ove lekcije

Izvršavanje vježbe:

1. Spajanje sa računalom – spojite Arduino razvojnu pločicu putem USB kabla u slobodno USB sučelje računala.
2. Arduino razvojno okruženje (IDE) – otvorite arduino razvojno okruženje te provjerite da li je razvojna pločica pravilno spojena na računalo (**Tools → Port → COM**#). Ovisno o USB priključku na računalu, možete imati drugačije sučelje od svojih kolega. Ako imate više sučelja za odabir najčešće je pravi onaj zadnji (skroz donji).

   Nakon odabira pravog porta potrebno je odabrati pravilnu pločicu (**Tools → Boards → Arduino/Genuino Uno**).

3. Sada možete napisati i prenijeti svoj prvi program na Arduino razvojnu pločicu.

```
void setup() {
  /* Ovdje stavite kod koji ce se jednom pokrenuti
     9600 - Brzina komunikacije sa racunalom (Baud rate)
     Serial.println() → Ispis teksta u port
  */
  Serial.begin(9600);
  Serial.println("Dobrodosli u Arduino");
  Serial.println("Ovo je vasa prva aplikacija!");
}

void loop() {
  /* Ovdje stavite glavni kod koji ce se konstantno ponavljati */
}
```

- Osnovna postava svakog arduino programa
    - **Svaki** arduino program sastoji se od dvije glavne funkcije:
      `void setup()`
      `void loop()`
    - Setup funkcija se izvršava samo jednom pri pokretanju mikro upravljača (ili resetiranju)
    - Loop funkcija se izvršava u nedogled (do isključenja ili resetiranja mikro upravljača)

4. Prenošenje na razvojnu pločicu

   Postupak:
    - Provjeravanje (verify) i sastavljanje (compile) koda (**Sketch → Verify and Compile** ili **Ctrl + R**)
    - Prijenos na razvojnu pločicu (Upload) (**Sketch → Upload** ili **Ctrl + U**)
5. Serijski monitor
    - Ako su vam svi koraci dosada bili uspješni, kada otvorite serijski monitor (**Tools → Serial Monitor** ili **Ctrl + Shift + M**) ispisati će se dotični tekst unutar serijskog monitora.
        - **Napomena:** Ako vam pri uključenju serijskog monitora ne prikazuju poruke provjerite da li vam je na serijskom monitoru dobra brzina komunikacije (baud rate) te resetirajte pločicu.

*Slike za sve postupke se nalaze u **dodatku** na kraju priručnika.

Lekcija 2 – Treptanje LE diode

U ovoj lekciji ćemo pomoću Arduino razvojne pločice regulirati treptanje LE diode. Ovaj projekt spada pod „Hello World“ u Arduino zajednici.

Pribor:

No	Naziv
1	Eksperimentalna pločica
1	LE dioda (Proizvoljna boja)
1*	330ohm otpornik
	Kratko spojnici (žice)

Potrebno znanje:

1. Upravljanje **ulaznim/izlaznim sučeljem** razvojne pločice. (IO port)
    - Svako ulazno/izlazno sučelje razvojne pločice može se postaviti u 2 stanja **ULAZ** i **IZLAZ**. Stanja postavljamo ključnim riječima `INPUT` i `OUTPUT` pomoću funkcije pinMode. Parametri funkcije pinMode su pin kojemu želimo odrediti stanje te to stanje.
    - Postoje 3 vrste ulazno/izlaznih sučelja na Arduino razvojnoj pločici; Digitalni, Digitalni PWM i Analogni, mi ćemo se zasada fokusirati na samo Digitalno ulazno/izlazno sučelje.
        - Digitalno ulazno/izlazno sučelje prepoznaje dva stanja; VISOKO i NISKO naponsko stanje koja se određuju ključnim riječima `HIGH` i `LOW`. Naponska stanja postavljamo pomoću naredbe digitalWrite gdje su parametri funkcije sučelje na koje želimo postaviti vrijednost i to stanje. Visoko naponsko stanje na Infiniduino UNO R3 odgovara +3.3V dok nisko naponsko stanje odgovara 0V.
            - Naponska stanja mogu se očitati i kao cjelobrojne vrijednosti gdje visoko naponsko stanje odgovara 1 a nisko 0.
            - Svako digitalno ulazno/izlazno sučelje može propustiti maksimalno 20mA struje.
    - Kako bi mogli upravljati brzinu treptanja diode, trebati će nam način da upravljamo vremenskim razmacima što postižemo pomoću delay naredbe čiji je parametar vrijeme za koje želimo zaustaviti rad mikro upravljača (u milisekundama)

2. **LE dioda** (Light Emmiting)
    - Sastoji se od dvije nožice, anode i katode. Kako bi dioda ispuštala svjetlosnu energiju, na anodi se mora nalaziti veći napon nego na katodi (u ovom slučaju će na anodi biti 3.3v a na katodi 0v). Svaka LE

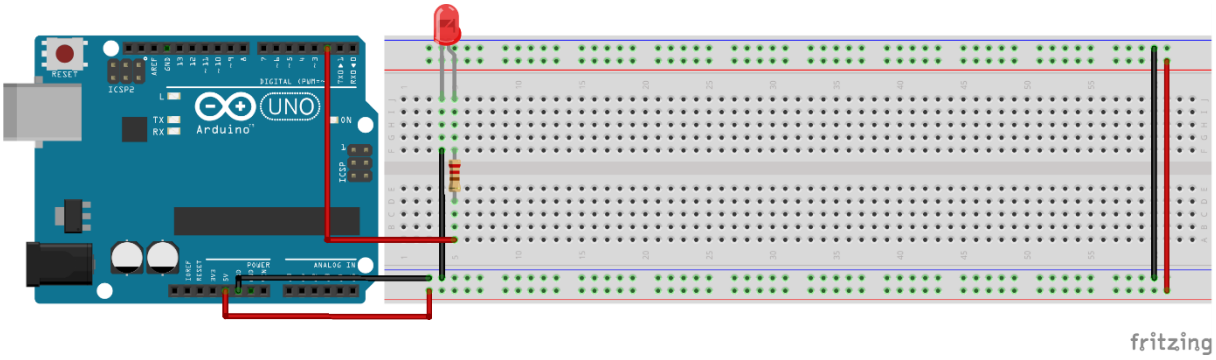
dioda ima svoj operativni napon, tj. **napon praga** (**Umin** i **Umax**) koji mora biti osiguran za pravilan rad LE diode.

Tablica 1.

Umin	Umax	Boja	
1.9V	2.1V	Crvena	
3.0	3.4	Plava	Zelena
2.9	4.2	Ljubičasta	

- Kako bi osigurali pravilan rad LE diode potrebno je osigurati pravilan napon na diodi kako kroz nju ne bi potekla prevelika struja. To postižemo dodavanjem dodatnog otpora koji računamo pomoću Ohmovog zakona. (Za ove primjene biti će dovoljno koristiti otpornik od 330Ohma)

Shema:



Kod:

```
/* LE dioda je spojena na digitalno suÄŤelje 2 razvojne ploÄŤice */
const int LED = 2;

void setup() {
 // inicijalizacija digitalnog pina na tip izlaz
 pinMode(LED, OUTPUT);
}
void loop() {
 digitalWrite(LED, HIGH); /* upali LED */
 delay(1000); /* sekunda pauze (1000ms --> 1s) */
 digitalWrite(LED, LOW); /* ugasi LED */
 delay(1000); /* sekunda pauze */
}
```

Lekcija 3 – Pomičuće svijetlo

U ovoj lekciji ćemo pomoću razvojne pločice napraviti efekt pomicanja svijetla kroz 8 LE dioda.

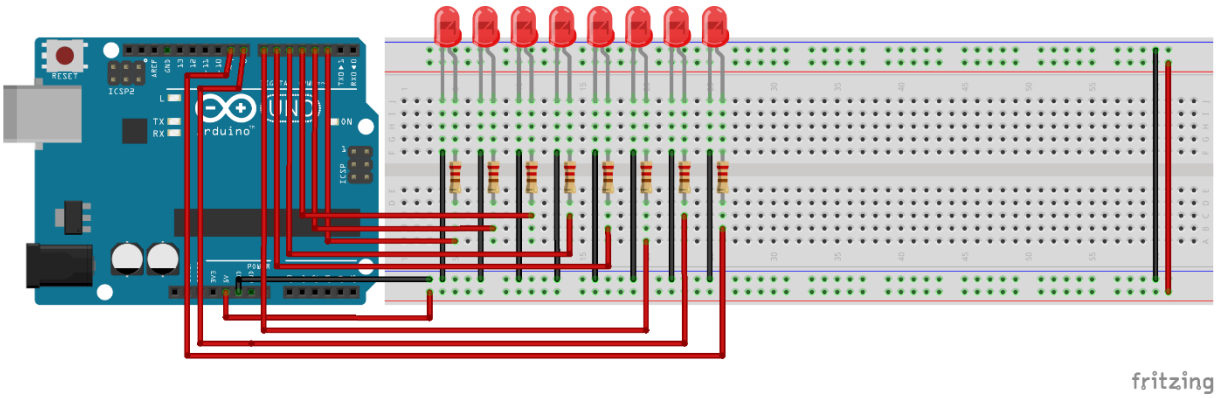
Pribor:

No	Naziv
1	Eksperimentalna pločica
8	LE dioda (Proizvoljna boja - ista)
8	330ohm otpornik
	Kratko spojnici (žice)

Potrebno znanje:

1.) Upravljanje ulaznim/izlaznim sučeljem razvojne pločice
- U ovom primjeru koristimo više od jednog digitalnog sučelja na razvojnoj pločici. To znači da ćemo morati definirati više od jednog izlaznog sučelja u programskoj podršci.
  - Postoje dva načina na koja se to može riješiti:
    1. Deklaracijom svih izlaznih sučelja pojedinačno
    2. Deklaracijom svih izlaznih sučelja preko `for` petlje
       - Mi ćemo se fokusirati na ovaj način inicijalizacije kako je jednostavniji i brži za implementaciju u ovom slučaju u kojemu se sva sučelja koriste za istu namjenu

Shema:



Kod:

```
const int prva_LED = 2;
const int br_LED = 8;

void setup() {
  // inicijalizacija 8 digitalnih pinova na izlaz
  for(int i = prva_LED; i < prva_LED + br_LED; i++)
  {
   pinMode(i, OUTPUT);
  }
}

void loop() {
  for(i = prva_LED; i < 10; i++)
  {
    digitalWrite(i, HIGH); // upaliti LED diodu
    delay(100); // čekanje 100 milisekundi
    digitalWrite(i, LOW); // ugasiti LED diodu
  }
}
```

Lekcija 4 – Tipkalom upravljana LE dioda

U ovoj lekciji pomoću tipkala upravljamo LE diodom.

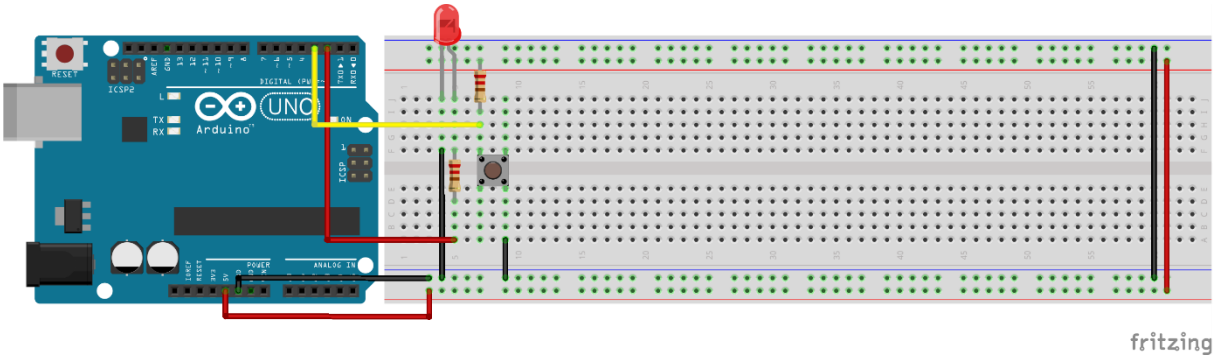
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	Tipkalo
1	LE dioda (Proizvoljna boja - ista)
2	330ohm otpornik
	Kratko spojnici (žice)

Potrebno znanje:

1. Podsjetnik: upravljanje ulazno/izlaznim sučeljima razvojne pločice:
   - Tipkalo moramo spojiti preko otpora za povlačenje na visoku naponsku razinu na sučelje koje je u stanju `INPUT` postavljeno pomoću pinMode funkcije.
     - *Nije potrebno spajati vanjski otpor na tipkalo ako pri inicijalizaciji sučelja umjesto parametra `INPUT` koristimo `INPUT_PULLUP`

Shema:



Kod:

```
const int LED = 2;
const int BTN = 3;

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(BTN, INPUT);
}

void loop() {
  /* Spremanje stanja tipkala u varijablu */
  int BTN_stanje = digitalRead(BTN);
  /* Ispitivanje stanja tipkala
     tipkalo je aktivirano ako je u stanju LOW*/
  if(BTN_stanje == LOW){
    digitalWrite(LED, HIGH);
  }else{
    digitalWrite(LED, LOW);
  }
  delay(100);
}
```

Lekcija 5 – PWM upravljanje LE diode

U ovoj lekciji upravljamo LE diodom pomoću pulsno-širinske modulacije (PWM).

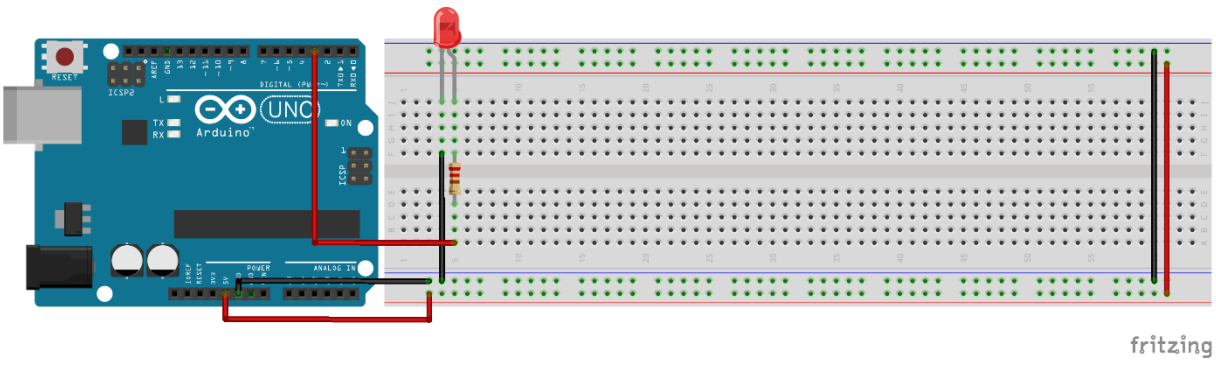
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	LE dioda (Proizvoljna boja - ista)
1	330ohm otpornik
	Kratko spojnici (žice)

Potrebno znanje:

1. Upravljanje ulazno/izlaznim sučeljima razvojne pločice
    - Ako pogledate bliže na svoju razvojnu pločicu primijetiti ćete da se pokraj nekih sučelja nalazi znak ' # ' (na originalnim pločicama ' ~ ') . Takva sučelja imaju mogućnost pulsno-širinske modulacije. Ako želimo imati bilo kakav oblik upravljanja pomoću pulsno-širinske modulacije moramo sve vanjske uređaje spajati na takva sučelja
    - **Pulsno-širinska modulacija** je oblik upravljanja signala preko efektivne vrijednosti. Naime efektivna vrijednost je signala je ona kojom će izvor djelovati na trošilo (u našem slučaju LE diodu). Pojednostavljeno**, efektivna vrijednost** je omjer djela periode koji se nalazi u visokoj naponskoj razini i onoga koji se nalazi u niskoj naponskoj razini (**Vrijedi isključivo za kvadratni signal!**). To nam omogućava da jednostavnim vremenskim upravljanjem određujemo koliki se **efektivni napon** nalazi na LE diodi, automatski upravljajući njenim osvjetljenjem.
        - Pulsno-širinskom upravljanju pomoću digitalnih ulazno/izlaznih sučelja pristupamo naredbom analogWrite kojoj su parametri efektivna vrijednost izlaznog signala (0 – 255 → **kvant uzorkovanja** je 2^-8 tj. 1/256 gdje je snaga izlaznog signala 0% za 0 te 100% za 255)

Shema:



Kod:

```
const int LED = 3;

void setup() {
  pinMode(LED, OUTPUT);
}

void loop() {
  /* Jakost osvjetljenja LE diode se povecava od 0 do 255*/
  for(int i = 0; i < 255; i++){
    analogWrite(LED, i);
    delay(15);
  }
  /* Jakost osvjetljenja LE diode se smjanjuje od 255 do 0*/
  for(int i = 255; i >= 0; i++){
    analogWrite(LED, i);
    delay(15);
  }
}
```

Lekcija 6 – RGB LE dioda

U ovoj lekciji prikazujemo cijeli spektar boja RGB LE diodom pomoću pulsno-širinske modulacije.

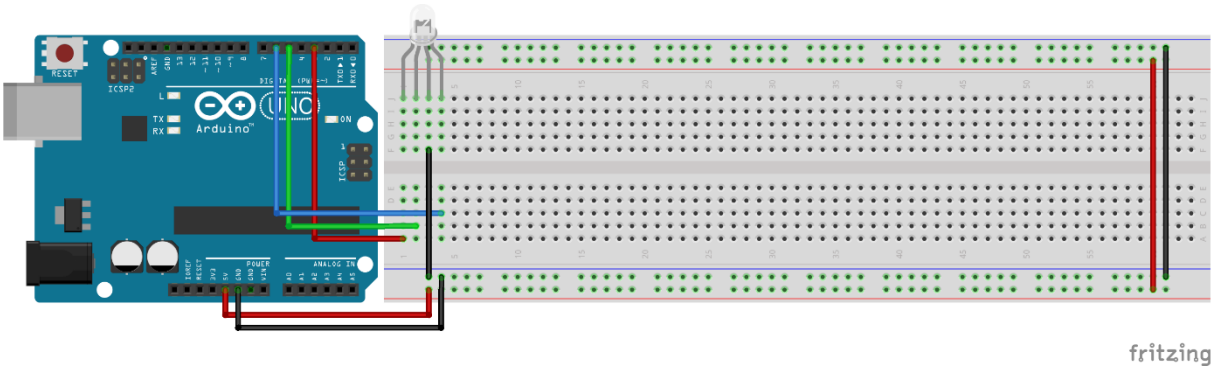
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	RGB LE dioda
3	330ohm otpornik
	Kratko spojnici (žice)

Potrebno znanje:

1. RGB LE diodu upravljamo pomoću teorije kolori metrije, naime, cijeli spektar boja se nalazi u bijeloj boji, odnosno nju postižemo miješanjem 3 glavne boje (**[R]** crvene **[G]** zelene **[B]** plave). Dok postižemo bijelu boju, važno je da se boje miješaju u maksimalnim iznosima i jednakim omjerima, što je na Arduino razvojnoj pločici 255 (sjetimo se da se u pulsno-širinskoj modulaciji odabiru vrijednosti između 0 i 255).
    - Druge boje dobivamo jednostavnom promjenom unosa određenih boja u konačnu, npr. Kako bi dobili plavu boju moramo postići unos plave boje da bude 100% a zelene i crvene boje 0% tj. Ako predstavimo svaku boju u obliku vektora (R, G, B) onda bi plava boja bila (0, 0, 255). za ljubičastu boju (255, 0, 255) itd.
    - Upravljanje jačinom osvjetljena LE diode u pojedinoj boji vršimo skaliranjem svih vrijednosti vektora za onu količinu koju želimo smanjiti osvjetljenje, npr. Ako želimo da LE dioda svijetli bijelom bojom ali na 50% jačine vektor bi izgledao ovako: (127, 127, 127). Analogno vrijedi i za druge boje i jačine.
        - Programski se ovaj vektor može izvesti kao polje.

Shema:



Kod:

```
const int R = 3;
const int G = 5;
const int B = 6;
void setup() {
  pinMode(R, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(B, OUTPUT);
}

void loop() {
  unsigned int RGB[3] = {0, 0, 0};

  /* s - odabir boje koja se smanjuje
     p - odabir boje koja se povecava */
  for (unsigned int s = 0; s < 3; s += 1) {
    /* Pomocu ternarnog operatora postizemo ciklicko mijenjanje boja*/
    int p = s == 2 ? 0 : s + 1;
    /* Smanjujemo i povecavamo odabrane boje kako i prosli kroz spektar
       boja */
    for(unsigned int i = 0; i < 255; i += 1) {
      RGB[s] -= 1;
      RGB[p] += 1;
      /* Pozivamo deklariranu funkciju */
      prikaziBoju(RGB);
      delay(10);
    }
  }
}
/* Funkcija koja nam omogucava da upravljamo sa diodom pomocu vektora
boja*/
void prikaziBoju(unsigned int* rgb){
  analogWrite(R, rgb[0]);
  analogWrite(G, rgb[1]);
  analogWrite(B, rgb[2]);
}
```

Lekcija 7 – Pravljenje zvuka pasivnom zujalicom

U ovoj lekciji naučiti ćemo upravljati pasivnom zujalicom.

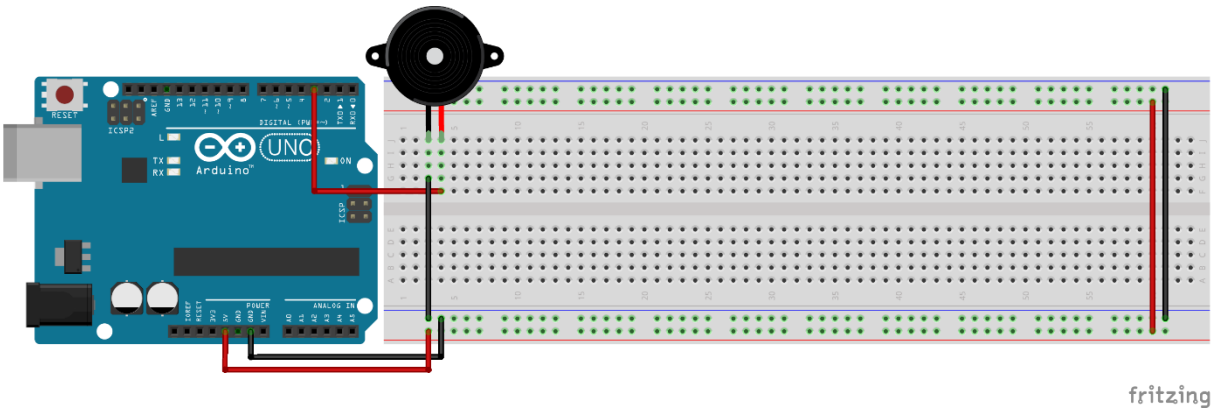
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	LE dioda (Proizvoljna boja - ista)
1	Aktivna zujalica (piezo)
1	Pasivna zujalica (piezo)
1	330ohm otpornik
	Kratko spojnici (žice)

Potrebno znanje:

1. Piezo element
    - Piezo je Elektro-mehanička komponenta koja pretvara električnu energiju (signal) u mehaničku energiju (oscilacija → zvuk)
        - Pasivni piezo element ima mogućnost regulacije osciliranja što mu omogućava da se koristi kao glazbeni uređaj.
    - Mi ćemo za upravljanje piezo elementa koristiti ugrađenu funkciju tone koja nam omogućava da jednostavno odredimo notu i njeno trajanje.
        - Parametri funkcije tone su sučelje na koje spajamo piezo element, nota te njeno trajanje.

Shema:



Kod:

```
#define C 2093.00
#define D 2349.32
#define E 2637.02
#define F 2793.83
#define G 3135.96
#define A 3520.01

const int SPKR = 3;

double melodija[14] = { C, C, G, G, A, A, G, F, F, E, E, D, D, C};
int    trajanja[14] = { 8, 8, 8, 8, 8, 8, 4, 8, 8, 8, 8, 8, 8, 4};

void setup() {
 pinMode(SPKR, OUTPUT);
 digitalWrite(SPKR, LOW);
}

void loop() {
  /* Petlja koja prolazi kroz melodiju odabiruci notu i trajanje*/
  for(int n = 0; n < 15; n++){
    /* Trajanje note u odnosu na sekundu*/
    int t = 1000 / trajanja[n];
    tone(SPKR, melodija[n], t);
    /* Pauza izmedu svake note 30% trajanja te note*/
    delay(t * 1.3);
    /* Stisavanje zvucnika za slucaj da se sviraju dvije note zaredom*/
    noTone(SPKR);
  }
}
```

Lekcija 8 – Čitanje analognih vrijednosti

Dosada smo naučili očitavati stanja prekidača, tj. Digitalne vrijednosti. U ovoj lekciji naučiti ćemo čitati analogne vrijednosti pomoću potenciometra te ispisivati te vrijednosti na serijski monitor.

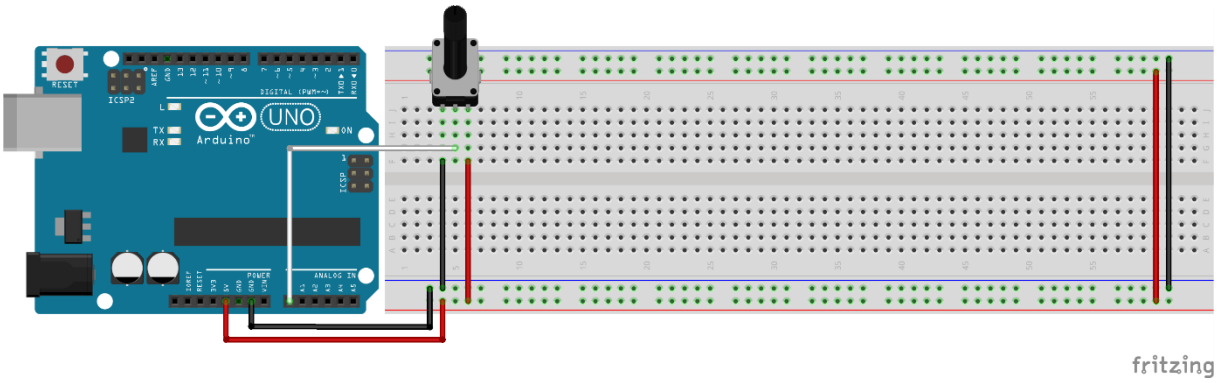
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	10kOhm Potenciometar
	Kratko spojnici (žice)

Potrebno znanje:

1. Upravljanje ulazno/izlaznim sučeljima Arduino razvojne pločice
    - Dosada smo koristili samo digitalna sučelja razvojne pločice i kao ulaze i kao izlaze. Vidjeli smo da neka digitalna sučelja imaju mogućnost PWM modulacije kada u stanju izlaza, no ako želimo očitavati analogne vrijednosti, obrat ne vrijedi, moramo koristiti analogna sučelja.
        - Analognim sučeljima pristupamo preko vrijednosti A0 – A5 (inicijalizacija u prvom primjeru)
        - Postavljanje stanja sučelja se i dalje izvršava preko funkcije pinMode
        - Ako je sučelje u izlaznom stanju, izlaz možemo upravljati i preko digitalWrite i analogWrite
        - Ako se sučelje nalazi u ulaznom stanju, očitanje ulaznih vrijednosti se može izvršiti preko digitalRead i preko analogRead funkcije.
            - Pri korištenju analogRead funkcije vrijednosti koje mikro upravljač očitava se kreću u rasponu od 0 – 1023, što znači da je kvant uzorkovanja od 2^-10.

Shema:



Kod:

```
const int POT = A0;
void setup(){
  Serial.begin(9600);
}

void loop(){
  /* Spremanje vrijednosti potenciometra u varijablu*/
  int POT_v = analogRead(POT);
  /* Ispisivanje varijable u Serijski monitor*/
  Serial.println(POT_v);
}
```

Lekcija 9 – LE dioda upravljana količinom osvjetljenja

Dosada smo upravljali uređaje digitalnim senzorima, tj. Prekidačima, u ovoj lekciji ćemo upravljati LE diodu analognim senzorom osvjetljenja

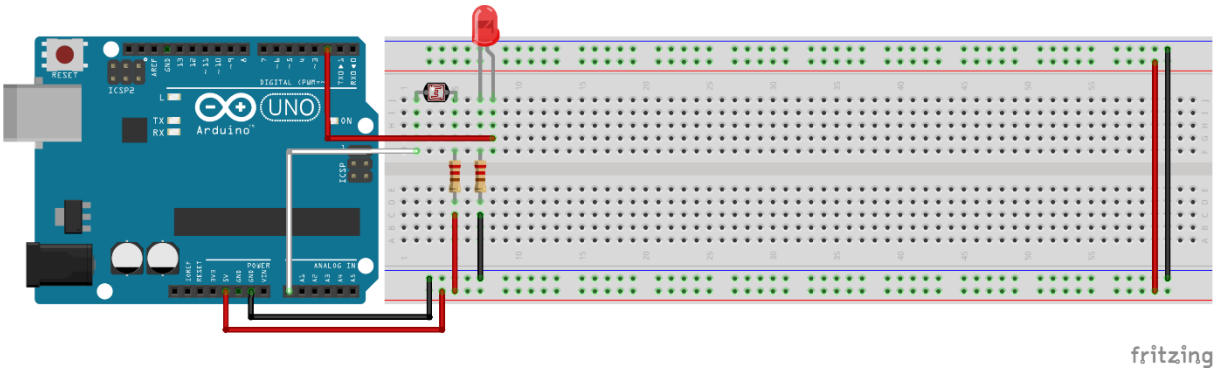
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	LDR otpornik
1	LE dioda (Proizvoljna boja - ista)
2	330ohm otpornik
	Kratko spojnici (žice)

Potrebno znanje:

1. Foto-otpornik (Light-dependent Resistor - LDR)
    - Foto-otpornik ili otpor ovisan o količini svijetla je komponenta kojoj otpor ovisi isključivo o količini svijetla. LDR može imati i do nekoliko Mohma otpora u potpunom mraku dok u dobrom osvjetljenju čak i nekoliko stotina oma.
        - Kako je ovo komponenta kojoj se mijenja otpor, njenu „vrijednost“ ćemo očitavati pomoću analogWrite funkcije.
            - U seriju sa foto-otpornikom spajamo otpor kako se otpor u toj grani ne bi mogao spustiti dovoljno da dođe do kratkog spoja.

Shema: PORPAVITI!! Pin LE diode



Kod:

```
const int LDR = A0;
const int LED = 3;

void setup(){
  pinMode(LDR, INPUT);
  pinMode(LED, OUTPUT);
}

void loop(){
  /* Spremanje vrijednosti potenciometra u varijablu*/
  int LDR_v = analogRead(LDR);
  /* Pretvaranje vrijednosti ocitanja LDR otpornika na
     vrijednost kompatibilnu za prikaz sa LE diodom */
  int LED_v = map(LDR_v, 0, 1023, 0, 255);

  analogWrite(LED, LED_v);
  delay(25);
}
```

Lekcija 10 – Infracrvena komunikacija

U ovoj lekciji naučiti ćemo osnove infracrvene komunikacije pomoću **IRremote** biblioteke.

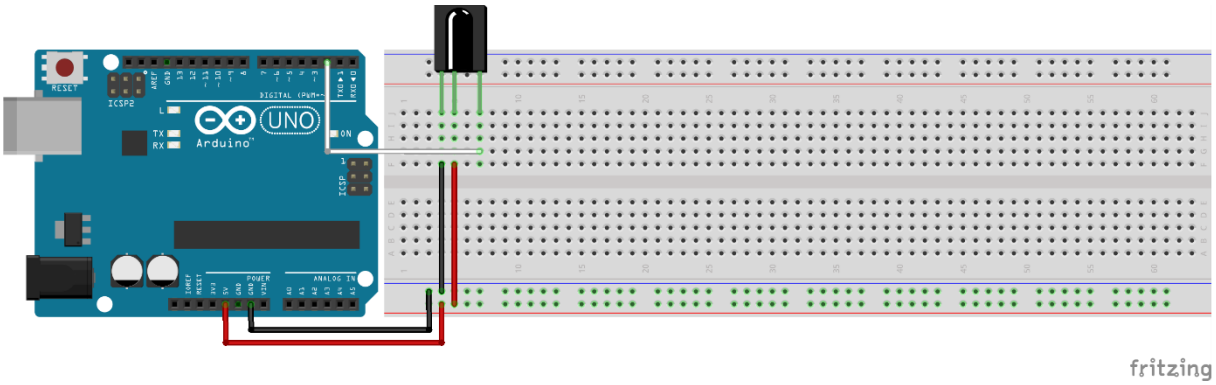
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	IR prijemnik (senzor)
1	IR odašiljač (daljinski upravljač)
	Kratko spojnici

Potrebno znanje:

1. **Infracrvena komunikacija**
   - Infracrvena komunikacija se bazira na vremenskom proračunu signala dekodiranih binarnim kodom (serijski prijenos podataka).
     - Svakom signalu se programski može odrediti funkcija koju će vršiti.
   - Pri izvedbi infracrvene komunikacije u nekom sustavu možemo iskoristiti već napravljene infracrvene predajnike (u ovom primjeru ćemo se fokusirati na ovu metodu) i napraviti svoje pomoću tipkala (kasniji primjer).
     - Pri izvedbi programske podrške za infracrvenu komunikaciju sa postojećim predajnikom, moramo saznati kodove određenih tipkala na predajniku, bilo preko dokumentacije ili očitanja kodova.
2. **Upotreba biblioteka za izvedbu programske potpore**
   - Arduino platforma posjeduje jednu od najvećih zajednica koja je razvila (i dalje razvija) mnoštvo biblioteka kako bi mogli jednostavno implementirati „kompliciranе“ zadatke, poput naše infracrvene komunikacije.
     - Mi ćemo koristiti Iremote biblioteku (poveznica u dodatku)
   - Kako koristiti biblioteke možete naći u dodatku na kraju priručnika.

Shema:



Kod:

```
#include <IRremote.h>

int prij = 11;
IRrecv prijemnik(prij);
decode_results rez;

void setup()
{
  Serial.begin(9600);
  prijemnik.enableIRIn();
}

void loop()
{
  if (prijemnik.decode(&rez))
    {
     Serial.println(rez.value, HEX);
     prijemnik.resume();
    }
}
```

Lekcija 11 – 8-segmentni LED zaslon

U ovoj lekciji naučiti ćemo koristiti 8-segmentni LED zaslon za prikazivanje znamenki. Cilj ove vježbe je da na zaslonu možemo prikazati ciklus automata sa konačnim brojem stanja.

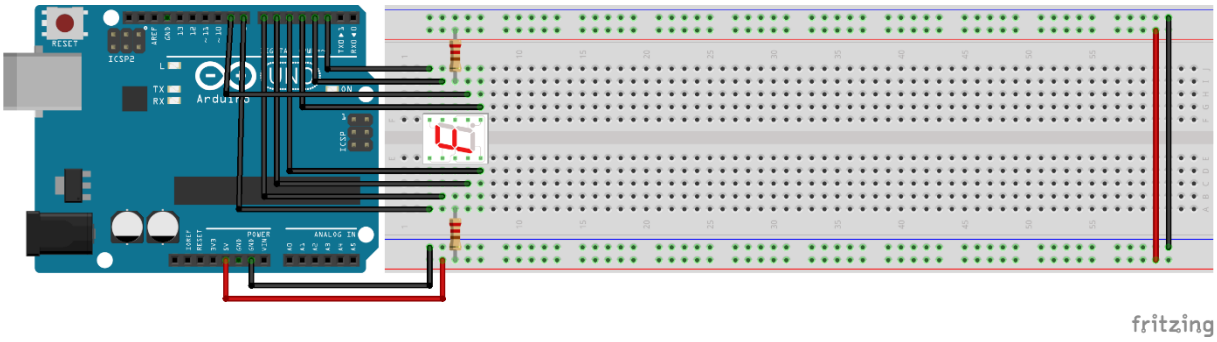
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	8 – segmentni LED zaslon
2	330Ohm otpornik
	Kratko spojnici

Potrebno znanje:

1. **8 - segmentni LED zaslon**
    - Koristi 8 LE dioda spojenih na zajedničku anodu / katodu (određuje način kojim upravljamo zaslonom).
        - U ovom slučaju imam zajedničku katodu što znači da upravljamo zasebnim LE diodama pomoću niske naponske razine.

Shema:



Kod:

```
byte znamenke[10] = {0xFC, 0x60, 0xDA, 0xF2, 0x66,
          0xB6, 0xBE, 0xE0, 0xFE, 0xF6};

void setup(){
  for(int i = 2; i < 10; i++) pinMode(i, OUTPUT);
}

void loop(){
  for(int z = 0; z < 10; z++){
    for(int i = 1,j = 2; i <= 128; i*=2, j++){
      int PIN = (((znamenke[z] & (i)) == 0 ? -1 : 1) * j);
      digitalWrite((PIN < 0) ? -PIN : PIN, (PIN < 0) ? HIGH : LOW);
      i = (i == 0) ? 1 : i;
    }
    delay(1000);
  }
}
```

Lekcija 12 – Četverostruki 8-segmentni LED zaslon

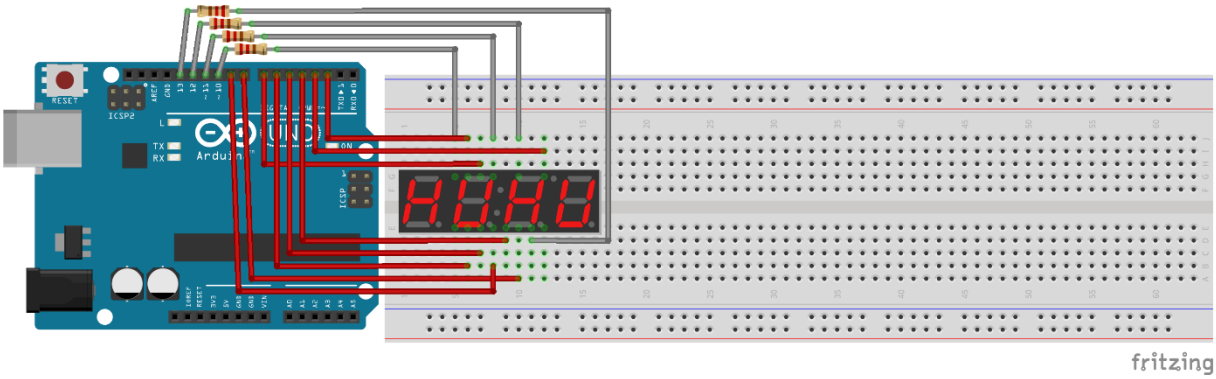
Pribor:

No	Naziv
1	Eksperimentalna pločica
4	330 Ohm otpornik
1	Četverostruki 8-segmentni LED zaslon
	Kratko spojnici

Potrebno znanje:

1. Upravljanje četverostrukim 8-segmentnim LED zaslonom (**Multipleksiranje**)
    - Upravljanje zasebnih 8-segmentnih zaslona se vrši na jednak način kao i jednostruki, dok **odabir pojedinog zaslona** biramo pomoću sučelja koja su spojena na zajedničke anode/katode.
        - Radi relativno „sporog“ mikro-upravljača na Arduino pločici može doći do prekidanja fluidnosti prikaza znamenki na četverostrukom displayu.

Shema:



Kod:

```
byte znamenke[10] = {0xFC, 0x60, 0xDA, 0xF2, 0x66, 0xB6, 0xBE, 0xE0, 0xFE,
0xF6};
int br = 0;
int az = 10;
void setup(){
  Serial.begin(9600);
  for(int i = 2; i < 14; i++) pinMode(i, OUTPUT);
}

void loop(){
  int brzn = br;
  for(int b = 13; b >= 10; b--){
    digitalWrite(b, HIGH);
    for(int i = 1,j = 2; i <= 128; i*=2, j++){
      int PIN = (((znamenke[brzn % 10] & (i)) == 0 ? -1 : 1) * j);
      digitalWrite((PIN < 0) ? -PIN : PIN, (PIN < 0) ? HIGH : LOW);
      i = (i == 0) ? 1 : i;
    }
    brzn /= 10;
    delayMicroseconds(5000);
    digitalWrite(b, LOW);
  }
  br+=50;
  if(br >= 9999) br = 0;
  delay(10);
}
```

Lekcija 15 – 74HC595 Posmični registar

U nekim primjerima nam se može dogoditi da nećemo imati dovoljno sučelja za upravljanje uređajima. U ovoj lekciji naučiti ćemo upravljati sa „beskonačno“ mnogo izlaznih sučelja pomoću samo 3 izlazna sučelja sa razvojne pločice.

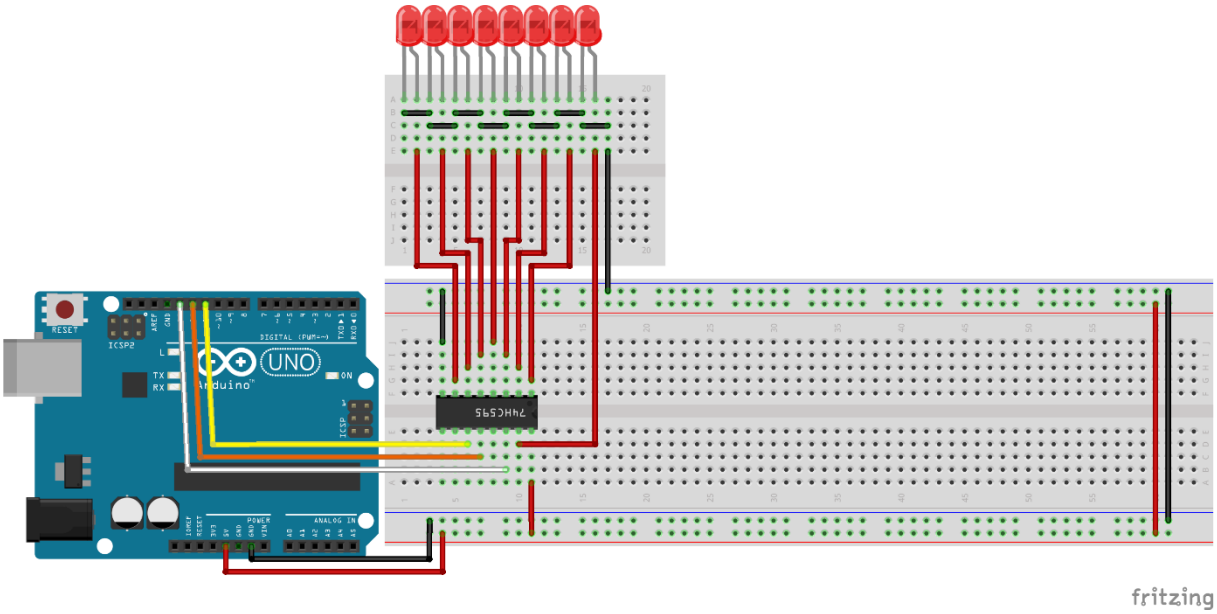
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	74HC595 Posmični registar
8	LE dioda
	Kratko spojnici

Potrebno znanje:

1. **Posmični registar**
   - Skupina bistabila spojenih tako da nam omogućavaju posmak bitova koje dovodimo na ulaz. Postoje paralelni i serijski posmični registri. 74HC595 je serijski posmični registar što znači da ima samo jedan ulaz, onaj na koji dovodimo bitove za posmak

Shema:



Kod:

```
int latchPin = 12;
int clockPin = 11;
int dataPin = 13;

void setup() {
  pinMode(latchPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
}

void loop() {
  for (int j = 0; j < 256; j++) {
    digitalWrite(latchPin, LOW);

    shiftOut(dataPin, clockPin, LSBFIRST, j);
    digitalWrite(latchPin, HIGH);
    delay(1000);
  }
}
```

Lekcija 13 – Dot 8x8 Matrični LED zaslon

U ovoj lekciji naučiti ćemo upravljati 8x8 Matričnim LED zaslonom pomoću kombinacije 74HC595 posmičnog registra i samih sučelja razvojne pločice.

Pribor:

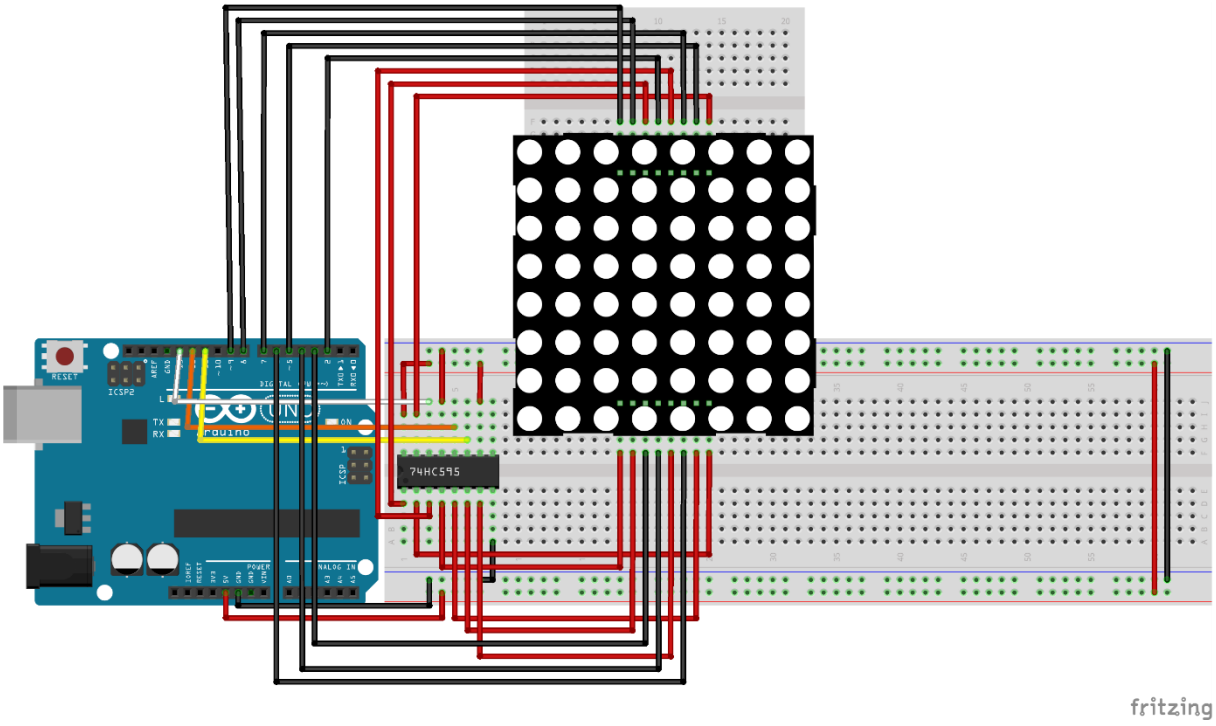
No	Naziv
1	Eksperimentalna pločica
1	Dot 8x8 matrični LED zaslon
1	74HC595 Posmični registar
	Kratko spojnici

Potrebno znanje:

1. **Matrični zaslon**
   - Kao što ime nalaže, zaslon se sastoji od matričnog rasporeda LE dioda koje aktiviramo dovođenjem napona na redak u kojem se nalazi željena LE dioda te uzemljenjem dotičnog stupca.

2. **74HC595 Posmični registar**
   - **Retke** matričnog zaslon upravljati ćemo pomoću 74HC595 posmičnog registra dok ćemo **stupce** zaslona upravljati direktno pomoću sučelja razvojne pločice.

Shema:



Kod:

```
const byte slika[] = { 0x00, 0x00, 0x24, 0x00,
                       0x42, 0x3C, 0x00, 0x00 };
int as = 2;
const int cAnode1 = 2;

const int latchPin = 12;
const int clockPin = 11;
const int dataPin = 13;

void setup() {
  pinMode(latchPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(dataPin, OUTPUT);

  for(int i = 2; i < 10; i++) pinMode(i, OUTPUT);
}

void loop() {
  digitalWrite(latchPin, LOW);
  shiftOut(dataPin, clockPin, LSBFIRST, slika[as - 2]);
  digitalWrite(latchPin, HIGH);

  for(int l = 2; l < 10; l++){
    digitalWrite(l, l == as ? LOW : HIGH);
  }
  as = (as >= 9) ? 2 : ++as;

  delay(1);
}
```

Lekcija 14 – 16x2 LC zaslon

Nakon što smo naučili mjeriti razne uvjete pomoću senzora bilo bi ih korisno prikazati negdje drugdje osim na serijskom monitoru. U ovoj lekciji naučiti ćemo kako koristiti 16x2 LC zaslon za prikaz podataka.

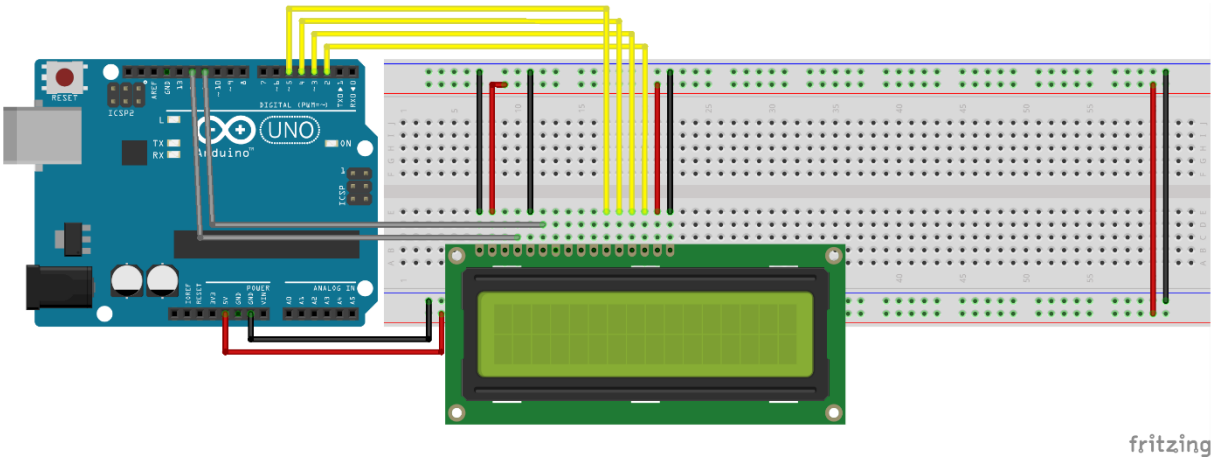
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	16x2 LC zaslon
	Kratko spojnici

Potrebno znanje:

1. LC zaslon
    - Princip rada LC zaslona nije u sklopu ovih lekcija, ono što budemo obradili u ovoj lekciji jest upravljanje LC zaslonom pomoću LiquidCrystal biblioteke

Shema:



Kod:

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup() {
  lcd.begin(16, 2);
  lcd.print("hello, world!");
}

void loop() {
  lcd.setCursor(0, 1);
  lcd.print(millis() / 1000);
}
```

Lekcija 15 – Upravljanje koračnim motorom

Koračni motori su postali jedni od najkorištenijih vrsta motora pojavom modernih CNC strojeva (3D printer, laserski rezač itd.). U ovoj lekciji naučiti ćemo upravljati koračnim motorom.

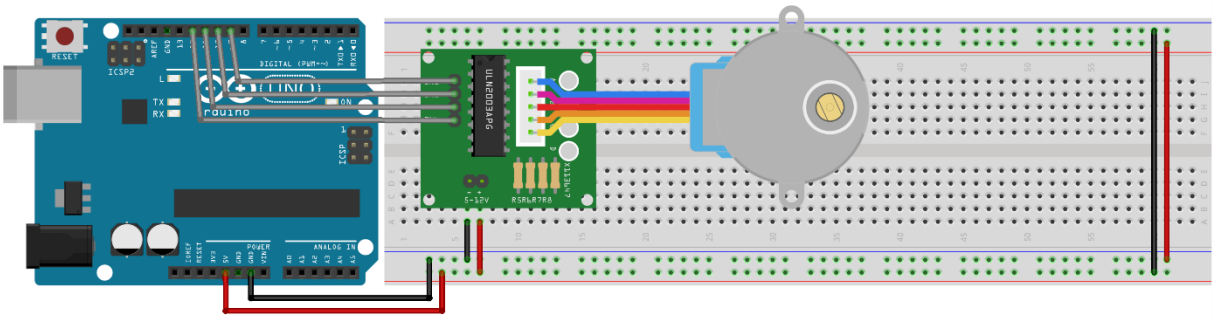
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	Koračni motor
1	Upravljač za koračni motor (driver)
	Kratko spojnici

Potrebno znanje:

1. Koračni motor
    - Koračni motor je vrsta motora koja nam omogućava precizno i konzistentno okretanje/upravljanje. Svaki koračni motor ima svoj **kut preciznosti** tj. najmanji kut za koji se može okrenuti. Zbog načina izvedbe koračnog motora, moramo koristiti **upravljač za koračni motor** koji u biti prevodi naše signale iz mikro upravljača u signale za upravljanje koračnim motorom te mu osigurava dovoljnu struju za upravljanje.

Shema:



Kod:

```
#include <Stepper.h>

const int okretPoRev = 200;
Stepper kor(okretPoRev, 8, 9, 10, 11);

void setup() {
  kor.setSpeed(60);
  Serial.begin(9600);
}

void loop() {
  Serial.println("Smjer kazaljke na sat");
  kor.step(okretPoRev);
  delay(500);

  Serial.println("Smjer nasuprot kazalje na sat");
  kor.step(-okretPoRev);
  delay(500);
}
```

Lekcija 16 – Upravljanje servo motorom

U ovoj lekciji naučiti ćemo upravljati servo motorima.

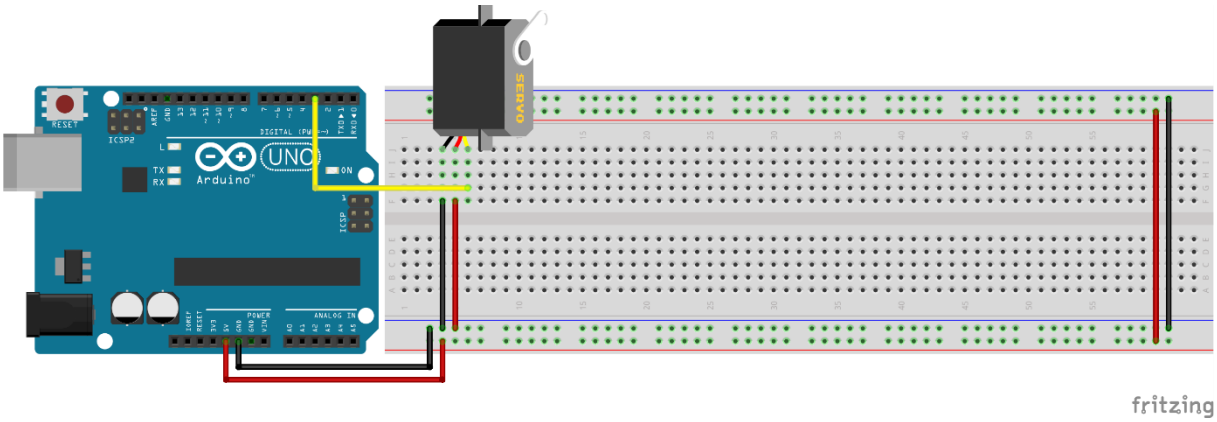
Pribor:

No	Naziv
1	Eksperimentalna pločica
1	Servo motor
	Kratko spojnici

Potrebno znanje:

1. **Servo motor**
    - Servo motor je vrsta motora koja nam omogućava jednostavno upravljanje određenom pozicijom motora. Naime, servo motor uvijek **zna** u kojem se **kutu nalazi** te se upravlja tako da mu odredimo krajnji kut u kojemu mi želimo da bude. Motorom ćemo upravljati pomoću Servo biblioteke.

Shema:



Kod:

```
#include <Servo.h>
Servo myservo;
const int servo = 3;
int poz = 0;
void setup() {
  /*Inicijalizacija servo motora na digitalnom sucelju 3*/
  myservo.attach(servo);
}

void loop() {
  for (poz = 0; poz <= 180; poz += 1) {
    // in steps of 1 degree
    myservo.write(poz);
    delay(15);
  }
  for (poz = 180; poz >= 0; poz -= 1) {
    myservo.write(pos);
    delay(15);
  }
}
```

Dodatak

Dodatni zadaci

Zadatak *

Potrebno je realizirati sustav koji će otključati vrata pomoću servo motora nakon što korisnik pritisne tipkalo.

Zadatak *

Potrebno je realizirati sustav pomoću kojega možemo upravljati sa 8 uređaja pomoću 74HC595 posmičnim registrom pomoću infracrvene komunikacije.

Zadatak **

Potrebno je napraviti Lekciju 2 na način da se osigura siguran rad tipkala. (Više o ovoj temi možete pročitati u dodatku - Debouncing)

Zadatak **

Potrebno je realizirati sustav u kojemu ćemo pri pritisku tipkala A povećati brojač za 50 a pri pritisku tipkala B podijeliti taj broj sa 5. Ako brojač ne bude djeljiv sa 5 resetirati ga u 0. Rezultat brojača prikazivati na četverostrukom 8-segmentnom zaslonu.

Zadatak **

Potrebno je realizirati sustav koji će upravljati servo motorom pomoću potenciometra. (Srednja pozicija potenciometra je 90 stupnjeva na servo motoru). Vrijednost potenciometra i kut pod kojim se nalazi servo motor potrebno je ispisati na LC zaslon.

Zadatak **

Potrebno je realizirati sustav koji će korisniku zvučnim signalom javiti kada se temperatura podigne iznad postavljene vrijednosti. Tu vrijednost morate odrediti ovisno o trenutnoj temperaturi u sobi te temperaturi koju senzor očita pri zatopljenju rukom (obično se ta vrijednost nalazi između 15 i 25 stupnjeva celzijusa)

Zadatak ***

Potrebno je ostvariti automat sa konačnim brojem stanja koji prolazi kroz ciklus: S0 – S1 – S3 – S5 – S4 – S2 – S6 – S7. Za Svako stanje djeljivo sa 2 treba aktivirati LE diodu na analognom sučelju 0, za svako stanje djeljivo sa 3 treba aktivirati LE diodu

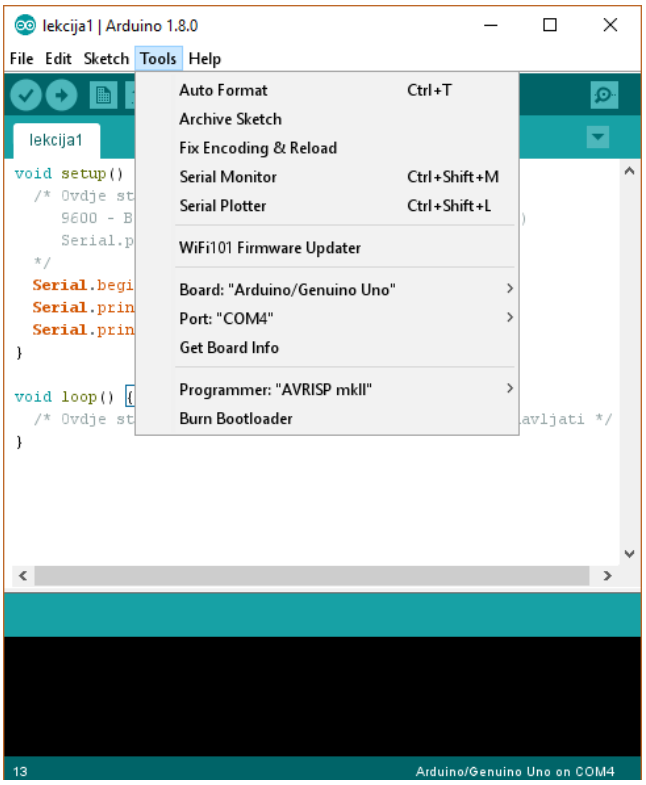
na analognom sučelju 1 te za sva stanja treba ispisati u serijski monitor i LC zaslon u prvom retku „Trenutno stanje: S#“ te u drugome slijedeće stanje „Slijedeće stanje: S#“.-

Zadatak ***

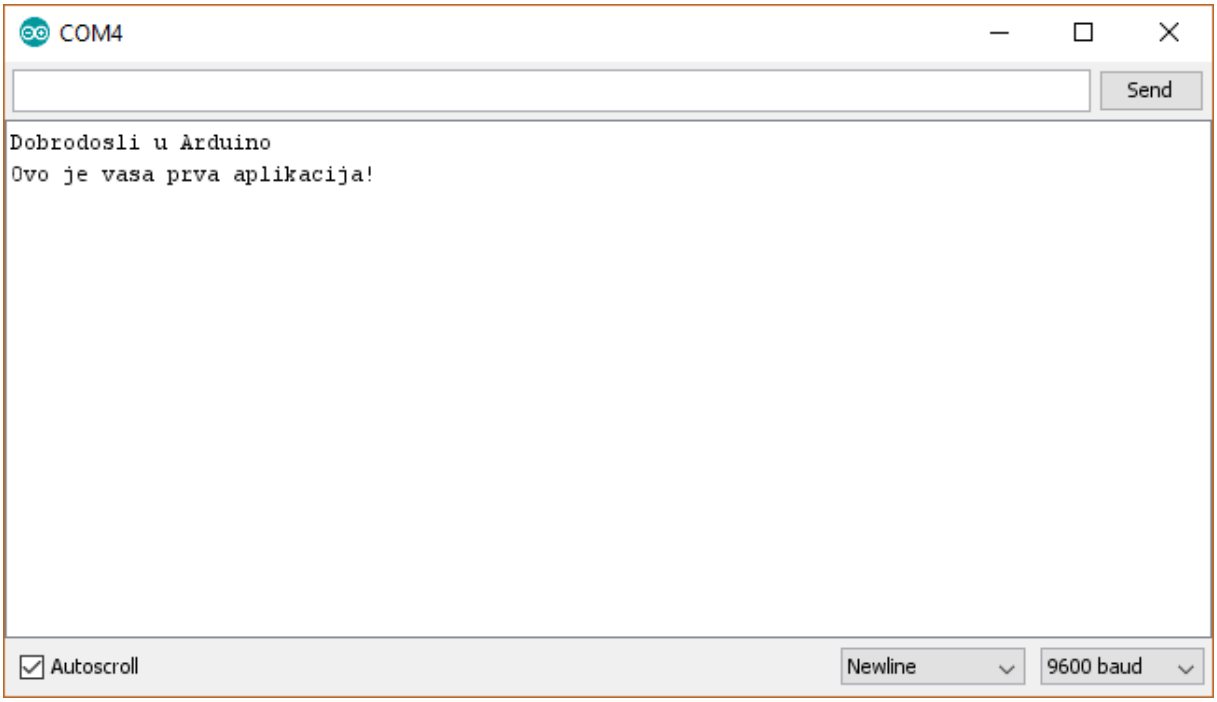
Potrebno je realizirati sustav koji će pri očitanju proizvoljne RFID oznake otvoriti vrata i spustiti rolete (Servo i koračni motor).

Popratno

Postavke sučelja



Izlaz serijskog monitora u Lekciji 1



Piezo

Piezo kao elektro-mehanička komponenta ima mogućnost reprodukcije pojedinih nota pomoću mijenjanja frekvencije osciliranja. Naime frekvencija osciliranja ovisi o dovedenom naponu na piezo element što nam omogućava da ga upravljamo promjenom efektivne vrijednosti napona na elementu odnosno pomoću analogWrite funkcije.

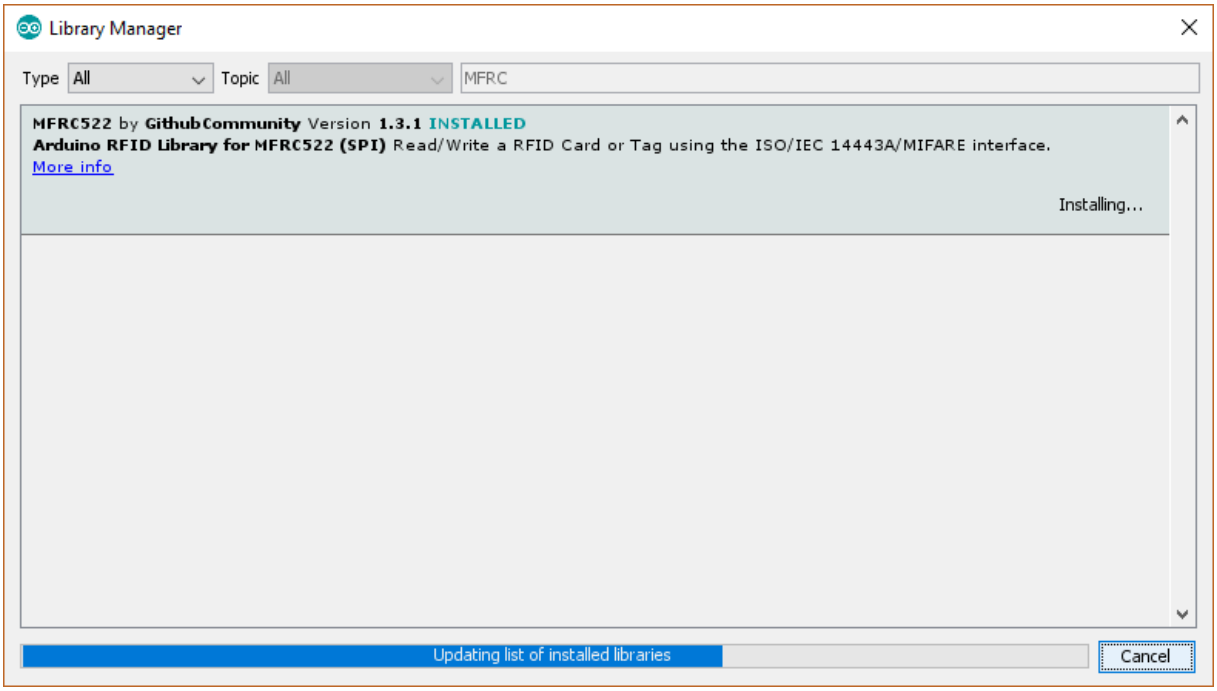
Kako ne bi sami trebali određivati frekvenciju kojom ćemo upravljati piezo elementom, koristit ćemo tone funkciju.

Biblioteke

Kako bi koristili biblioteke u našim primjerima moramo provjeriti da li je biblioteka uključena u razvojno okruženje.
Biblioteka za infracrvenu komunikaciju se nalazi među ostalim zadanim bibliotekama, samo je trebamo uključiti u naš projekt. (**Sketch → Include Library → Robot IR Remote**).

Ako se biblioteka već ne nalazi u razvojnom okruženju onda ju možemo dodati na dva načina:

1.) **Pomoću upravitelja biblioteka** (**Sketch → Include Library → Manage Libraries…**)
   - Kada smo u upravitelju biblioteka, potrebno je pronaći biblioteku i instalirati ju.



2.) **Ručno**
   - Kako bi ručno dodali biblioteke potrebno ih je prvo skinuti u **.zip** formatu te ih uključiti kao takve (**Sketch → Include Library → Add .ZIP Library…**).

Debouncing

- Debouncing je sustav / metoda koja nam omogučava da se zaštitimo od nepravilnog rada tipkala koje može nastati radi prirode tipkala. Naime, kako je tipkalo elektro-mehanička komponenta, postoje intrinzične smetnje zbog mehaničkog djelovanja. U ovom slučaju te smetnje su oscilacije koje nastaju pri pritisku / otpuštanju tipkala.
    - Način rješavanja ovog problema je najjednostavniji za napraviti pomoću programske podrške tako da uvedemo nekakvo vrijeme (obično je do 20ms više nego dovoljno) potrebno da se očita stvarna vrijednost tipkala

Vanjske poveznice

Službena Arduino stranica – http://www.arduino.cc
Službena Arduino dokumentacija – https://www.arduino.cc/en/Reference/HomePage
Službena Arduino programska podrška – https://www.arduino.cc/en/Main/Software
Iremote biblioteka - https://github.com/z3t0/Arduino-IRremote