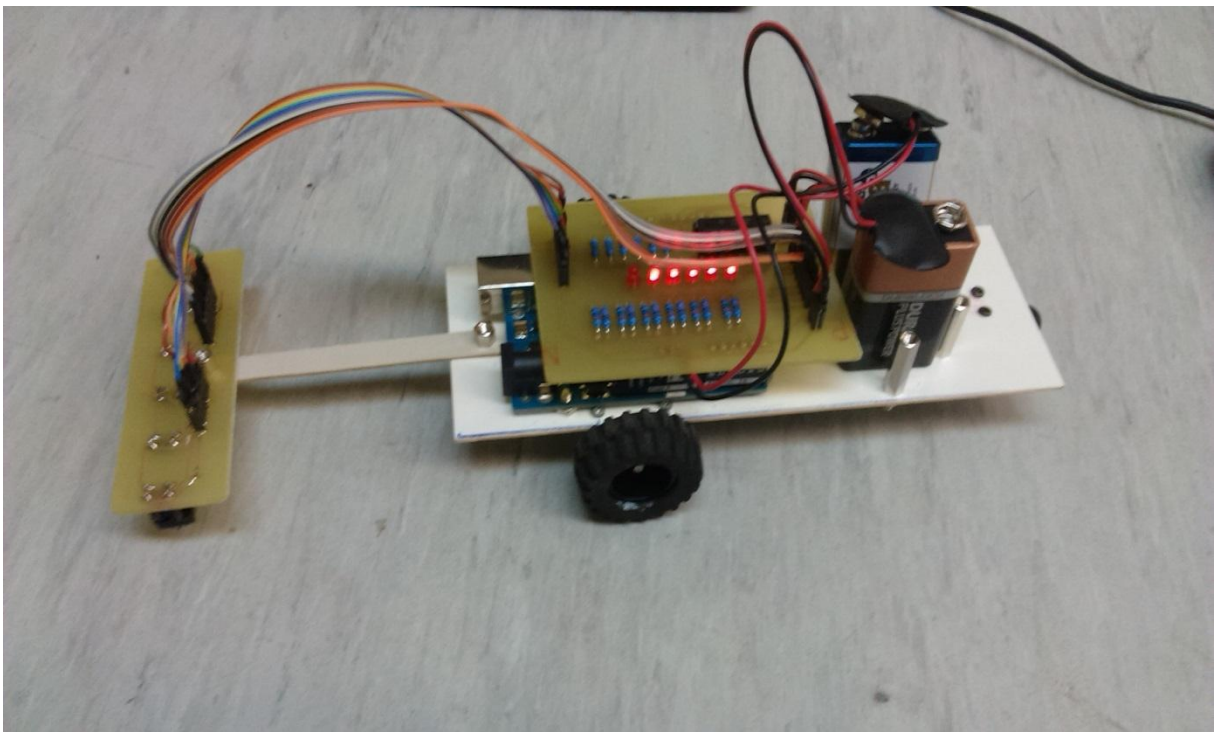
»Mladi za napredek Maribora 2015«

32. srečanje

ROBOSLED

Raziskovalno področje: Elektrotehnika, elektronika



Avtor: TILEN KURNIK, TADEJ GOLOB

Mentor: MILAN IVIČ

Šola: SREDNJA ELEKTRO-RAČUNALNIŠKA ŠOLA MARIBOR

Maribor, januar 2015

Kazalo

Kazalo slik

1 POVZETEK

Raziskovalna naloga predstavlja delo na področju robosleda.

Robosled sledi črni progi na beli podlagi.

Sestavljen je lahko na več različnih načinov. Letos smo se lotili novega načina. Za krmiljenje robosleda smo uporabili razvojno ploščo Arduino Uno, za senzorje pa infra rdeče senzorje TCRT 5000. Robosled vsebuje šest senzorjev, na podlagi katerih se zaznava proga oziroma odmik robosleda od proge. Če robosled zaide izven proge, bo na podlagi zadnjega shranjenega podatka stanja senzorjev ugotovil na katero stran mora zaviti, da spet pripelje do proge. Senzorji omogočajo boljše zaznavanje črne podlage saj so neodvisni od svetlobe v prostoru.

Robosled poganjata dva enosmerna motorčka z zobniškim prenosom znamke Polulo.

Robosled je lahko namenjen kot učni pripomoček ali kot osnova za tekmovanje, saj lahko vsak uporabnik spreminja program po zahtevah, ki jih narekujejo posamezne konfiguracije proge.

2 ZAHVALA

Za nasvete, pomoč, potrpljenje in za veliko pridobljenega znanja, se zahvaljujeva mentorju. Pridobljeno znanje bova lahko izkoristila pri najinem nadaljnjem delu.

3 VSEBINSKI DEL

3.1 Postopek raziskovalne naloge

Najprej smo se odločili o vrsti in številu senzorjev, ki smo jih uporabili za zaznavanje črne proge. Na eksperimentalni ploščici smo sestavili vezje enega senzorja in preizkusili delovanje. Ker je deloval po naših željah, smo se odločili, da jih uporabimo. Premišljevali smo o številu senzorjev. Ker so lahko oblike prog različne, tudi take z ostrimo ovinki, smo se odločili, da jih uporabimo šest. Pojavilo se nam je vprašanje, na kakšni medsebojni razdalji jih naj montiramo. Izdelali smo tri različne variante in se po preizkusu odločili za tisto, ki je delovala najbolje.

Program smo pisali v okolju Arduino 1.0.5-r2, saj smo za krmiljenje robosleda uporabili ploščo Arduino Uno.

Če smo hoteli videti odzive vrtenja posameznega motorčka v odvisnosti od vrednosti senzorjev, smo morali poiskati ustrezen gonilnik za motorčka.

Nato smo po sklopih sestavili vezje na eksperimentalno ploščico ter opazovali odzive vrtenja motorčkov, ko smo pod senzorjem spreminjali podlago, belo oziroma črno.

Sledilo je konstruiranje robosleda. Tiskanino s potrebnimi elementi smo konstruirali tako, da se je lahko vstavila kar na priključke plošče Arduino Uno. Ko je bila konstrukcija robosleda dokončana, smo nanjo pritrdili in priključili vse potrebne komponente ter preizkusili delovanje na izdelani progi.

3.2 Tehnična in tehnološka dokumentacija

3.2.1 Arduino Uno

Arduino Uno mikrokrmilniška tiskanina vsebuje mikrokrmilnik ATmega328. Ima 6 analognih vhodov ter 14 digitalnih vhodno/izhodnih pinov od katerih jih lahko 6 uporabimo za PWM izhode. Deluje s 16 MHz taktom in ima USB konektor, preko katerega lahko programiramo, napajamo in komuniciramo z osebnim računalnikom.

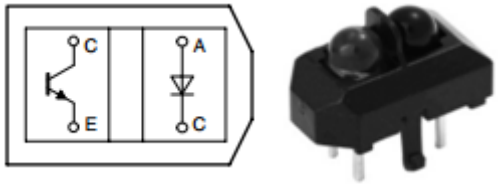
Ostali podatki so:

Mikrokontroler	ATmega328
Delovna napetost	5 V
Vhodna napetost napajanja plošče	7 – 12 V
Digitalni vhodno/izhodni pini	14 (6 jih lahko uporabimo za PWM)
Analogni vhodni pini	6
Tokovna obremenitev na vhodno/izhodni pin	40 mA
Tokovna obremenitev za 3,3 V na pin	50 mA
Flash programski pomnilnik	32 KB
SRAM pomnilnik	2 KB
EEPROM pomnilnik	1 KB
Oscilator	16 MHz

Arduino Uno ima 10 bitne analogne vhode (A/D), na katere lahko priključimo signale v območju 0 – 5 V. To pomeni, da se vhodno območje napetosti 0 – 5 V diskretizira v 2^{10} = 1024 vrednosti. Za analogne izhoda pa lahko uporabimo PWM (pulzno širinska modulacija) izhode. Določene procese lahko krmilimo neposredno z PWM signalom, če je frekvenca PWM signala veliko večja od dinamike procesa (elektromotorja).

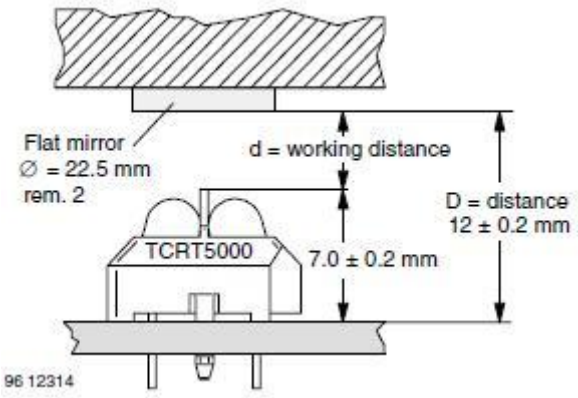
3.2.2 IR senzorji (TCRT 5000)

Za zaznavanje črne podlage uporablja robosled infrardeče senzorje TCRT 5000.



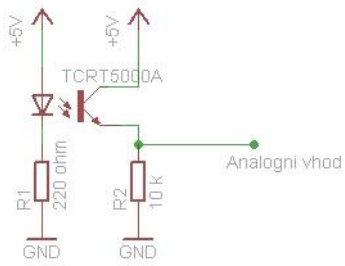
Slika 1: Senzor TCRT 5000 (vir: Vishay)

TCRT5000 je odsevni senzorji ki vključuje infrardeči oddajnik in foto tranzistor v ohišju, ki blokira svetlobo iz okolice. Valovna dolžina IR znaša 950 nm, napetostno napajanje 5 V, obremenimo pa ga lahko s 60 mA. Razdalja med senzorjem in podlago, od katere se odbija IR žarki je lahko do 5 mm, najboljša odzivnost pa je pri razdalji 2,5 mm.

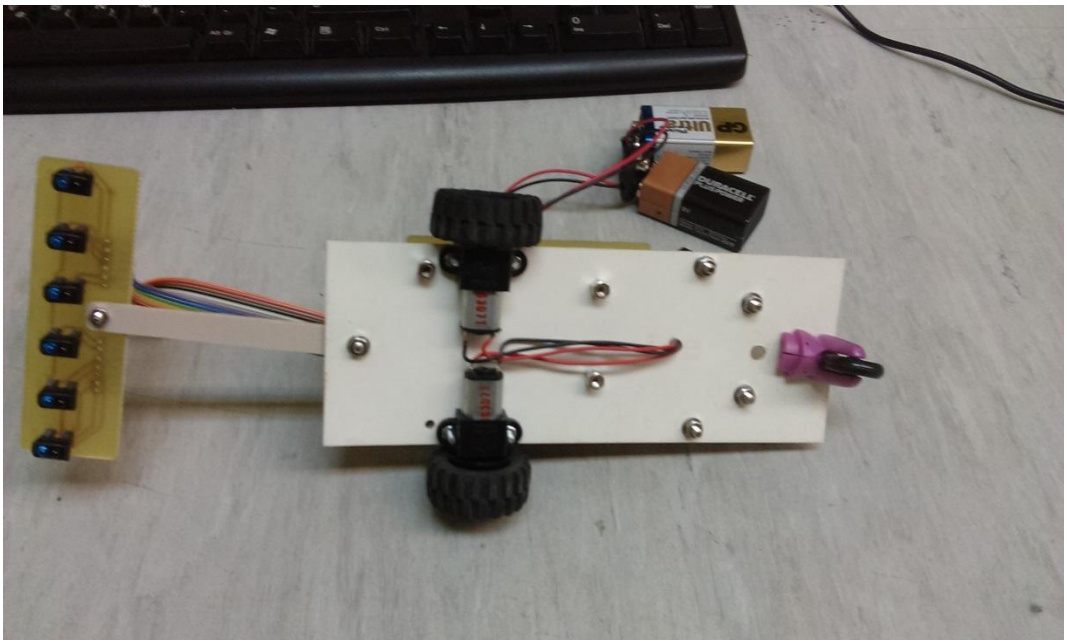


Slika 2: Razdalja med senzorjem in oviro (vir: Vishay)

Infrardeče senzorje smo izbrali zato, ker niso odvisni od dneve svetlobe. Delovanje senzorja smo preizkusili v vezavi z uporoma (slika 3) tako, da smo spreminjali podlago in opazovali odziv na izhodu.



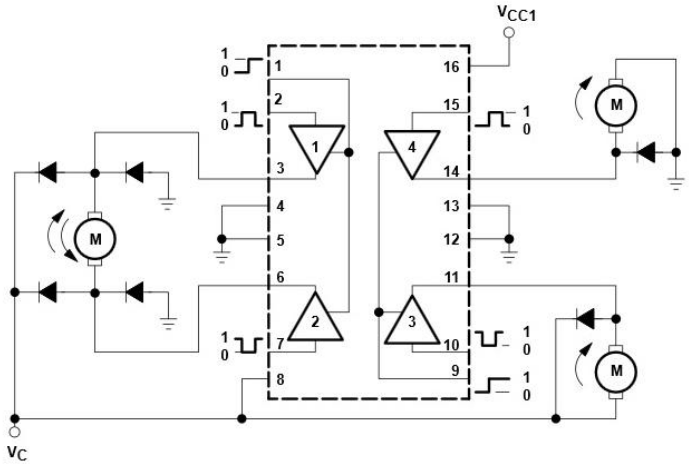
Slika 3: Priključitev senzorja TCRT 5000.



Slika 4: Montirani senzorji na robosledu.

3.2.3 Gonilnik L293

Integrirano vezje L293 je namenjeno za krmiljenje dveh enosmernih motorčkov, da se lahko vrtita v obe smeri. Nanj lahko priključimo motorček, ki za svoje delovanje ne potrebuje večji tok kot 1 A in za napetosti od 4,5 V do 36 V. Vsi vhodi so združljivi s TTL signali.



Slika 5: Integrirano vezje L293 (vir: Texas Instruments)

Za prvi motorček smo uporabili pine 1, 2 in 7:

Pin 1 (EN)	Pin 2 (1A)	Pin 7 (2A)	Funkcija
1	0	1	Vrtenje v smeri urinega kazalca
1	1	0	Vrtenje proti smeri urinega kazalca
1	0	0	Stop
1	1	1	Stop
0	neuporabno	neuporabno	Stop

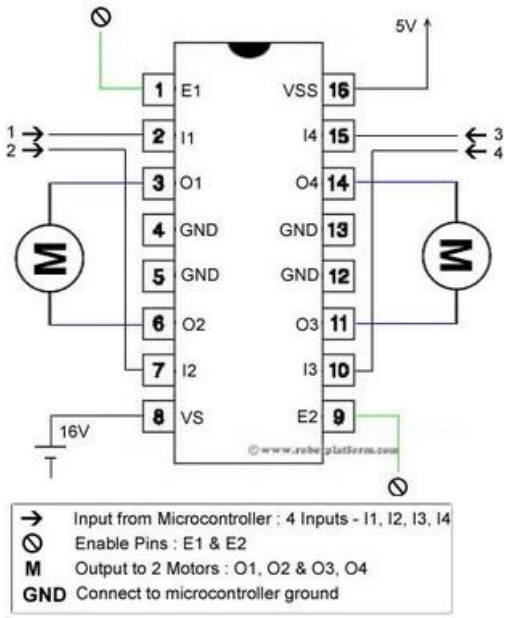
Za drugi motorček smo uporabili pine 9, 10 in 15:

Pin 9 (EN)	Pin 10 (3A)	Pin 15 (4A)	Funkcija
1	0	1	Vrtenje v smeri urinega kazalca
1	1	0	Vrtenje proti smeri urinega kazalca
1	0	0	Stop
1	1	1	Stop
0	neuporabno	neuporabno	Stop

Pina 1 in 9 sta enable pina. Biti morata povezana na +5 V, če hočemo da se bosta motorčka vrtela.

- Pini 4, 5, 12 in 13 so povezani na maso, isto kot je povezana masa (GND) od mikrokontrolerja ali Arduino plošče.
- Pini 2, 7, 10 in 15 so vhodni pini. So kontrolni pini. Pina 2 in 7 nadzorujeta prvi motorček, pina 10 in 15 pa nadzorujeta drugi motorček.
- Pini 3, 6, 11 in 14 so izhodni pini. Pina 3 in 6 sta za prvi motorček, pina 11 in 14 pa za drugi motorček.
- Pin 16 je pin za napetostno napajanje čipa L293. Priklopljen mora biti na napetost +5 V.

- Pin 8 je pin, kamor priključimo pozitivno napetost iz drugega napajalnika oz. baterije za napetost DC motorčkov (od 4,5 V do 36 V).



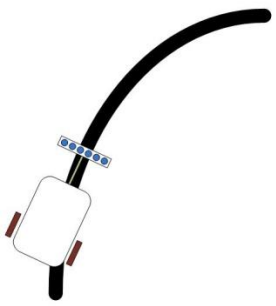
Slika 6: Priključitev na LM293 Integrirano vezje L293 (vir: Texas Instruments)

Za zaščito ima L293 vgrajene hitre diode, ki ščitijo IC pred napetostnimi konicami pri vklopu in izklopu motorčka (predvsem pri izklopu). Vsebuje notranje senzorje temperature. Če se IC zagreje čez mejo 70°C, senzorji ustavijo delovanje motorčkov.

3.2.4 Program in delovanje

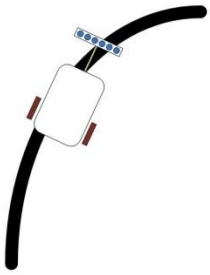
Princip delovanja robosleda je sedeč:

Če sta nad črno progo le sredinska senzorja (senzorja 3 in 4), se obe kolesi, ki ju poganjata enosmerna motorčka vrtita enakomerno in z največjo hitrostjo, ki jo nastavimo programsko.



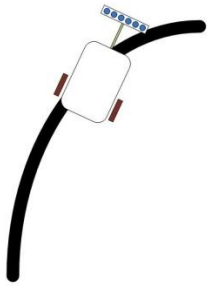
Slika 7: Sredinska senzorja (3 in 4) sta nad progo.

Če sta nad progo senzorja 4 in 5, kot prikazuje slika 6, se mora desni motorček vrteti počasneje od levega, da robosled zopet ujame progo.



Slika 8: Nad črno progo sta senzorja 4 in 5.

V kolikor je robosled zašel iz linije proge, kot je prikazano na sliki 7, se mora desni motorček vrteti še počasneje, morda za določeni čas ustaviti, da robosled zopet ujame progo. Program si je namreč zapomnil, da je bil pred tem, ko so vsi senzorji nad belo podlago, le senzor 6 nad črno podlago. Zato mora biti proga na desni strani robosleda.



Slika 9: Robosled je zašel iz linije proge.

Pri izdelavi programske kode smo si pomagali s kodo, ki jo je izdelal g. Stan [2]. V programu smo najprej določili pine za priklop motorčkov preko gonilnika (driverja) L293:

```
//Definiranje pinov za krmiljenje obeh DC motorčkov:
const int Motor_Desni_Pin1 = 2;      //Signal za desni motor, priključen na pin 2 na L293.
const int Motor_Desni_Pin2 = 3;      //Signal za desni motor, priključen na pin 7 na L293.
const int Motor_Desni_Enable = 5;  //Enable pin za desni motor, priključen na pin 1 na L293 (PWM omogočen)

const int Motor_Levi_Pin1 = 4;       //Signal za levi motor, priključen na pin 10 na L293.
const int Motor_Levi_Pin2 = 7;       //Signal za levi motor, priključen na pin 15 na L293.
const int Motor_Levi_Enable = 6;    //Enable pin za levi motor, priključen na pin 9 na L293 (PWM omogočen)
```

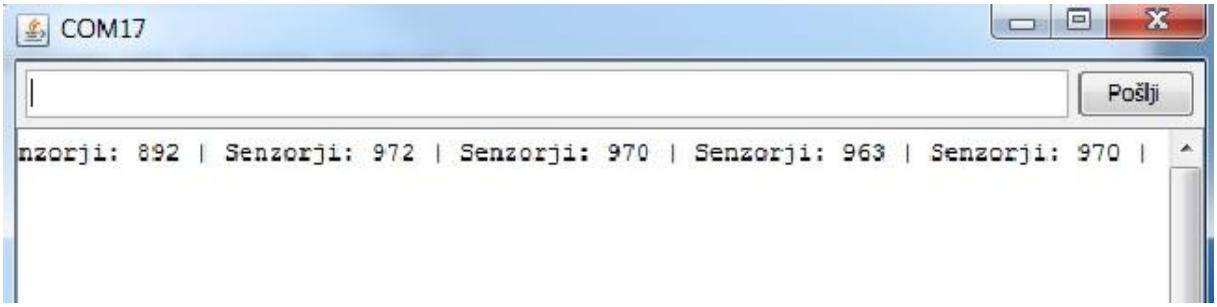
Senzorje bomo priključili na analogne vhode A0 – A5. Vrednosti na analognih vhodih bomo shranjevali, določili ustrezno mejo vrednosti med belo in črno podlago ter jih pretvorili v digitalne vrednosti, za lažjo manipulacijo v nadaljnjem programu.

Določili smo še izhodne pine za LED diod, ki nam simulirajo stanje senzorjev. Če LED dioda sveti pomeni, da je pripadajoči senzor nad belo podlago:

```
//Definiranje pinov za IR senzorje:
const int IR_Pini[6] = {A0, A1, A2, A3, A4, A5};  //IR senzorji so priključeni na analogne vhode.

//Definiranje pinov za LED diode:
```

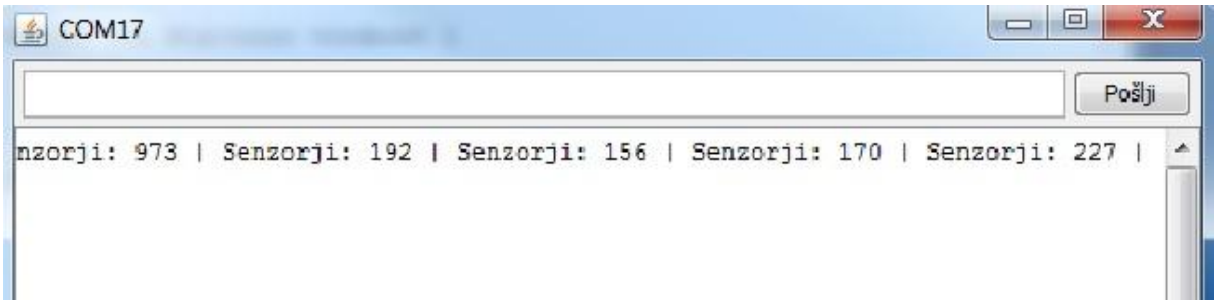
```
const int LED_Pini[6] = {8, 9, 10, 11, 12, 13};

//Polje za shranjevanje vrednosti (analogRead), ki so analognih vhodih => od IR senzorjev:
int IR_Senzorji_Analog[6] = {0,0,0,0,0,0};

/*Polje za shranjevanje digitalnih vrednosti (0 oz. 1) od IR senzorjev. Analogne
  prebrane vrednosti spremenimo v digitalne.
*/
int IR_Senzorji_Digital[6] = {0,0,0,0,0,0};
```

V program smo vključili serijsko komunikacijo med Arduino Uno in računalnikom. Na serijskem monitorju smo lahko prebrali vrednosti, dobljene na analognih vhodih, če so bili senzorji nad belo podlago. Te so se gibale okoli 970.



Slika 10: Vrednosti na analognih vhodih, senzorji nad belo podlago.

Če so bili senzorni nad črno podlago, so se te vrednosti gibale okoli 200.



Slika 11: Vrednosti na analognih vhodih, senzorji nad črno podlago.

Na podlagi dobljenih rezultatov smo določili mejo preklopa na vrednost 750.

```
int Prag_preklopa = 750;
for (int i = 0; i < 6; i++)
  {
    IR_Senzorji_Analog[i] = analogRead(IR_Pini[i]);
    if (IR_Senzorji_Analog[i] >= Prag_preklopa)        //Ali so senzorji nad črno progo?
    {
      IR_Senzorji_Digital[i] = 1;      //Ne, senzorji so nad belo podlago, dodelimo jim digitalno vrednost 1
    }
```

```
    else
    {
      IR_Senzorji_Digital[i] = 0;        //Da, senzorji so nad črno progo, dodelimo jim digitalno vrednost 0
    }
```

Senzorjem nad črno podlago smo dodelili digitalno vrednost 0, nad belo podlago pa digitalno vrednost 1. Nato smo določili odmik, ki se giblje od vrednosti -180 do 180. Od -180 do 0 za levi motorček in od 0 do 180 za desni motorček. Ta podatek nam služi za popravo robosleda na linijo proge. Popravo določimo izkustveno oziroma je odvisna od več faktorjev. Najprej poglejmo kako to zgleda v programu:

```
Odmik_Prejsnji = Odmik;
switch (IR_Senzorji)
  {
    case B111111:
    if (Odmik_Prejsnji < 0)
    {
      Odmik = -180;
    }
    else if (Odmik_Prejsnji > 0)
    {
      Odmik = 180;
    }
    break;

    //Po dva senzorja sta nad črno progo
    case B110011:      //A2 in A3 sta nad črno => L in D motor hitro
      Odmik = 30;
      digitalWrite(LED_Pini[0], HIGH);
      digitalWrite(LED_Pini[1], HIGH);
      digitalWrite(LED_Pini[2], LOW);
      digitalWrite(LED_Pini[3], LOW);
      digitalWrite(LED_Pini[4], HIGH);
      digitalWrite(LED_Pini[5], HIGH);
      break;

    case B100111:     //A1 in A2 sta nad črno => D motor počasneje
      Odmik = 90;
      digitalWrite(LED_Pini[0], HIGH);
      digitalWrite(LED_Pini[1], LOW);
      digitalWrite(LED_Pini[2], LOW);
      digitalWrite(LED_Pini[3], HIGH);
      digitalWrite(LED_Pini[4], HIGH);
      digitalWrite(LED_Pini[5], HIGH);
      break;
```

Strukturo switch case smo napisali za več primerov stanja senzorjev na progi. Na podlagi vrednosti odmika smo določili popravo, kot na primer:

```
void Posodobitev_Poprava()
 {
  if (Odmik >= 0 && Odmik < 30)
  {
    Poprava = 0;
```

```
  }
  else if (Odmik >= 30 && Odmik < 60)
  {
    Poprava = 15;
  }
```

Poprava pa nam služi za popravo hitrosti ustreznega motorčka:

```
if (Poprava >= 0)
  {
    Motor_Desni_Hitrost = Max_Hitrost - Poprava;
    Motor_Levi_Hitrost = Max_Hitrost;
  }
  else if (Poprava < 0)
  {
    Motor_Desni_Hitrost = Max_Hitrost;
    Motor_Levi_Hitrost = Max_Hitrost + Poprava;
  }
```

Maksimalno hitrost motorčkov smo določili na začetku programa, mi smo določili vrednost 195 (maksimalna je lahko 255 => 8-bitni register). Na podlagi položaja robosleda (senzorjev) in s tem podanega odmika, se od vrednosti maksimalne hitrosti motorčka odšteje vrednost poprave, zato se ustrezni motorček vrti počasneje, dokler se stanja senzorjev ne spremenijo.

V programu lahko določimo, da je vrednost poprave večja od vrednosti maksimalne hitrosti motorčka. Takrat bi bila vrednost za hitrost motorčka manjša od 0. V tem primeru se spremenita logična stanja na pinih, kamor sta povezana priključka gonilnika L293, ki določata v katero smer se vrti motorček. Ko pa je vrednost poprave enaka vrednosti maksimalne hitrosti motorčka, je njuna razlika enaka 0 in motorček se takoj ustavi:

```
void Voznja()
 {
  if (Motor_Desni_Hitrost > 255)
  {
    Motor_Desni_Hitrost = 255;
  }
  else if (Motor_Desni_Hitrost < -255)
  {
    Motor_Desni_Hitrost = -255;
  }
  if (Motor_Levi_Hitrost > 255)
  {
    Motor_Levi_Hitrost = 255;
  }
  else if (Motor_Levi_Hitrost < -255)
  {
    Motor_Levi_Hitrost = -255;
  }

  if (Motor_Desni_Hitrost > 0)      //Vrtenje desnega motorja naprej s PWM signalom.
  {
    analogWrite(Motor_Desni_Enable, Motor_Desni_Hitrost);
    digitalWrite(Motor_Desni_Pin1, HIGH);
    digitalWrite(Motor_Desni_Pin2, LOW);
```

```
}

else if (Motor_Desni_Hitrost < 0)      //Vrtenje desnega motorja nazaj s PWM signalom.
{
  analogWrite(Motor_Desni_Enable, abs(Motor_Desni_Hitrost));
  digitalWrite(Motor_Desni_Pin1, LOW);
  digitalWrite(Motor_Desni_Pin2, HIGH);
}

else if (Motor_Desni_Hitrost == 0)     //Desni motor se takoj ustavi.
{
  analogWrite(Motor_Desni_Enable, HIGH);
  digitalWrite(Motor_Desni_Pin1, LOW);
  digitalWrite(Motor_Desni_Pin2, LOW);
}


if (Motor_Levi_Hitrost > 0)      //Vrtenje levega motorja naprej s PWM signalom.
{
  analogWrite(Motor_Levi_Enable, Motor_Levi_Hitrost);
  digitalWrite(Motor_Levi_Pin1, HIGH);
  digitalWrite(Motor_Levi_Pin2, LOW);
}

else if (Motor_Levi_Hitrost < 0)      //Vrtenje levega motorja nazaj s PWM signalom.
{
  analogWrite(Motor_Levi_Enable, abs(Motor_Levi_Hitrost));
  digitalWrite(Motor_Levi_Pin1, LOW);
  digitalWrite(Motor_Levi_Pin2, HIGH);
}

else if (Motor_Levi_Hitrost == 0)      //Levi motor se takoj ustavi.
{
  analogWrite(Motor_Levi_Enable, HIGH);
  digitalWrite(Motor_Levi_Pin1, LOW);
  digitalWrite(Motor_Levi_Pin2, LOW);
}
```

Priključka enable, pin 1 na L293 za levi motorček in pin 9 na L293 za desni motorček, torej dobivata PWM signal, katerega duty cycle je odvisen od položaja senzorjev oziroma podlage nad katero so senzorji (neposredno od končne hitrosti motorčka, ki se dobi iz razlike med maksimalno hitrostjo minus poprava).

Frekvenco PWM signala smo iz privzete spremenili na 62 kHz. Arduino Uno, oziroma njegovo srce ATmega 328 vsebuje tri timer-je. Timer0 je 8-bitni register in določa (če mi tako želimo) frekvenco PWM signala na pinih 5 in 6 Arduino ploščice (zato smo mi priključili enable pina L293 na ta dva pina). Timer0 smo določili zato, ker omogoča hitri, Fast PWM. Prvi trije biti timerjevega registra določajo preddelitev, ki je privzeta 1:64. Mi smo preddelitev spremenili na 1:1 kar pomeni, da se timerjeva vrednost povečuje z urinim ciklom.

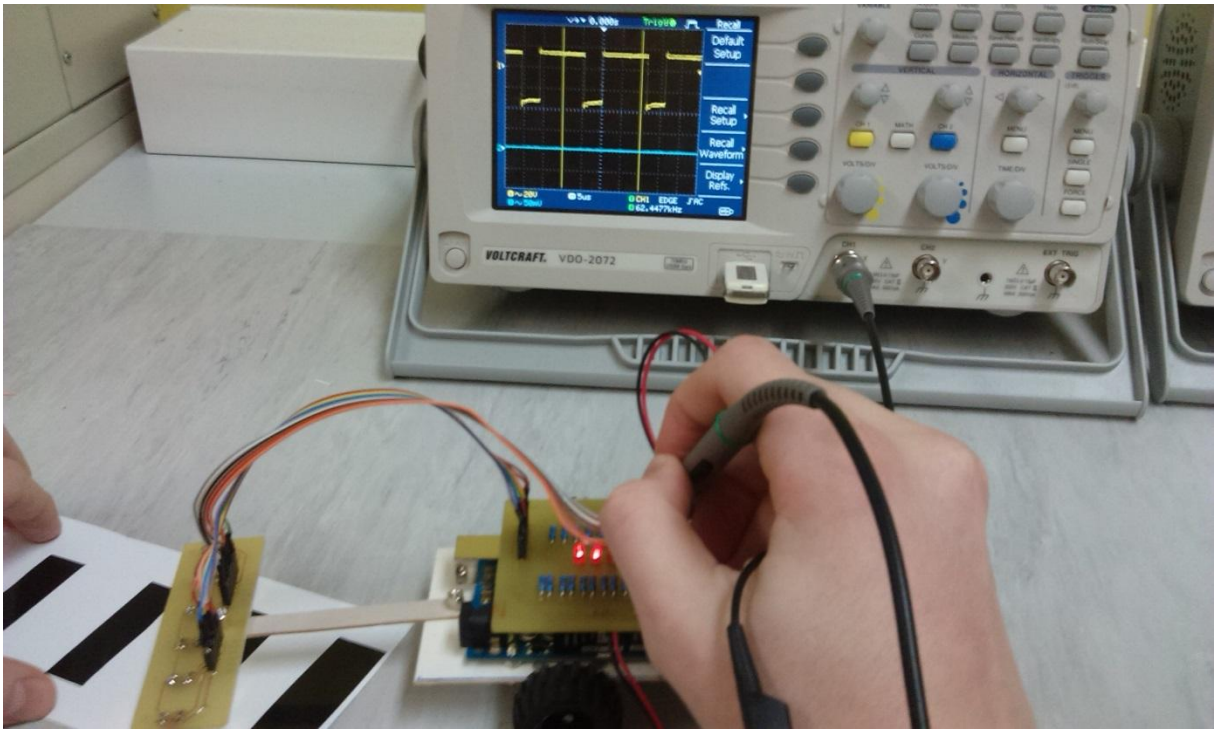
Frekvenca PWM = 16 000 000 / (preddelitev * 256)

f_{PWM} = 16 000 000 / 256 = 62500 Hz = 62,5 kHz

V programu smo to storili na naslednji način:

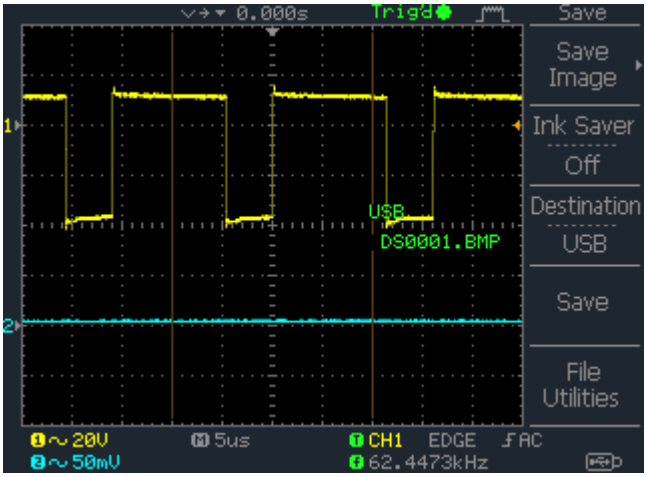
```
TCCR0A = _BV(COM0A1) | _BV(COM0B1) | _BV(WGM01) | _BV(WGM00);
TCCR0B = _BV(CS00);
```

Frekvenco pwm signala smo tudi izmerili z digitalnim osciloskopom:

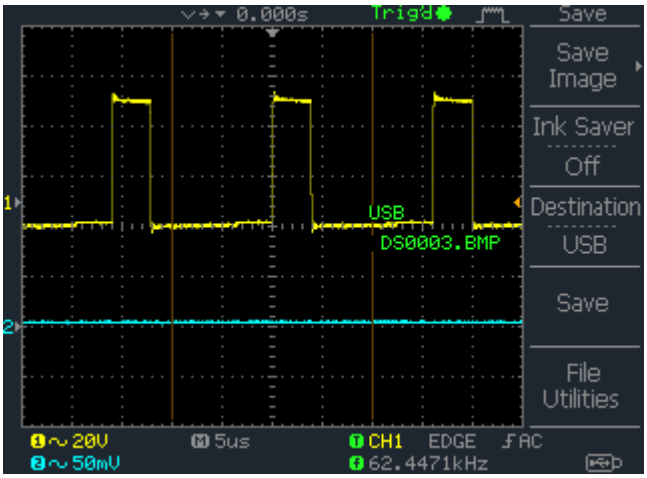


Slika 12: Merjenje PWM signala z osciloskopom.

Pri različni podlagi pod senzorjem, se je tudi duty cycle PWM signala spreminjal (hitrost vrtenja motorčka) kar prikazujejo slika 13 in slika 14.



Slika 13: Izmerjen PWM signal, primer 1.



Slika 14: Izmerjen PWM signal, primer 2.

Vrednosti odmikov in poprav kakor tudi različne pozicije senzorjev, ki se lahko pojavijo so odvisne od več faktorjev. Prvi je že hitrost motorčkov, ki smo ju izbrali. Mi smo izbrali Polulo-ve motorčke s prevelikim številom obratov oziroma takšnim prenosom, da sta bila motorčka prehitra. Zato smo zmanjšali maksimalni hitrost iz 255 na 195.

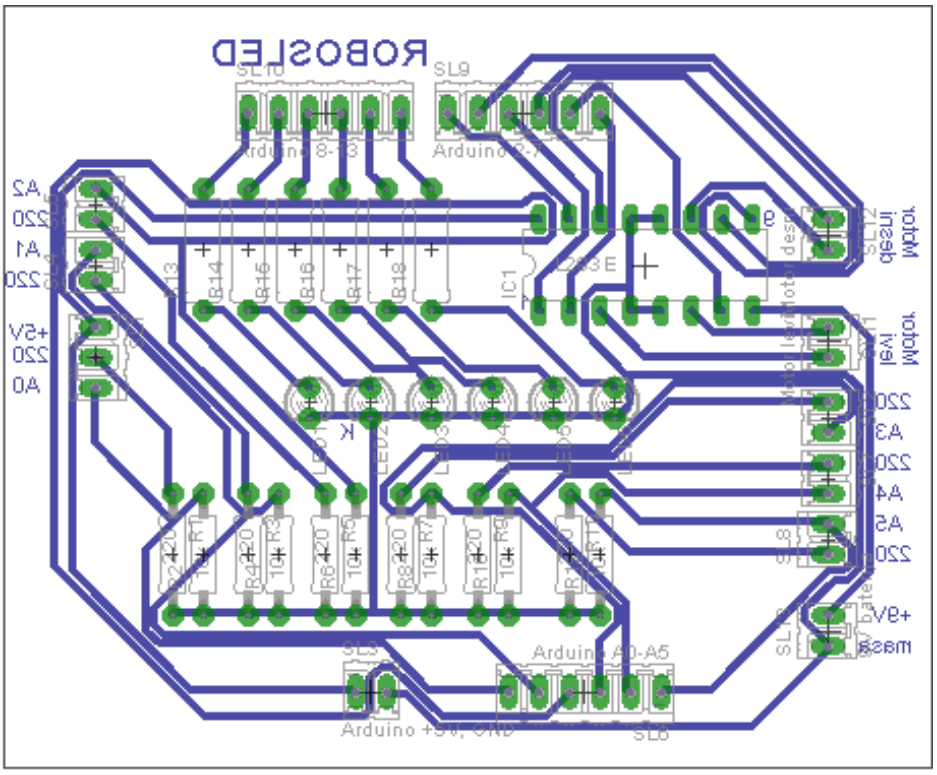
Pomembno vlogo igra oddaljenost senzorjev od osi motorčkov. Zato smo tudi to razdaljo spreminjali. Medsebojna oddaljenost senzorjev je odvisna predvsem od širine črne proge. Večje število senzorjev nam pomaga pri večji odzivnosti robosleda vendar se pisanje programa zakomplicira.

Zgodi se, da robosled »nastavimo« za eno progo, ki jo tudi uspešno prevozi, na drugi progi z drugačno konfiguracijo črne proge pa je neuspešen. Zato ga moramo ponovno »nastaviti«.

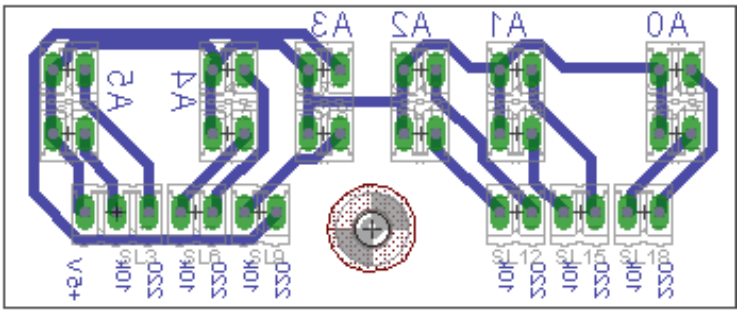
Težava je nastajala tudi s tem, ko smo morali za vsako spremembo programa, Arduino povezati na USB vhod računalnika, da smo lahko spremenjen program zapisali.

3.2.6 Tiskanina za robosled

Tiskanino za gonilnik L293, LED diode, povezave do senzorjev in ostale potrebne elemente smo izdelali v okolju Eagle.



Slika 15: Osnovna plošča, vstavljena v priključke plošče Arduino Uno.



Slika 16: Plošča s senzorji.

4 SKLEP

Z uspešno zaključeno izdelavo najine raziskovalne naloge sva zelo zadovoljena in ponosna. Če pogledava na začetek raziskovanja, ko sva imela samo zamisel, sva videla pred sabo veliko dela in problemov. Že na začetku sva veliko premišljevala, kako naj se naloge sploh lotiva. Veliko časa sva porabila z raziskovanjem delovanja in tudi s konstrukcijo robosleda, saj je tudi od konstrukcije odvisna uspešnost premagovanja ovinkov na progi.

S pomočjo mentorja in radovednosti sva svoj robosled izpopolnjevala. Sedaj vidiva, da končno izdelan robosled ni takšen, kot sva si ga na začetku zamislila.

Izdelava robosleda je bila zahtevna, pa na trenutke tudi zabavna, saj so nama take stvari zelo zanimive. Seveda so se pri izdelavi pojavile težave, ki sva jih korak za korakom premagovala.

5 Viri

- Vishay: http://www.vishay.com/docs/83760/tcrt5000.pdf (12. 12. 2014)
- Texas Instruments: http://www.ti.com/lit/ds/symlink/l293d.pdf (17. 12. 2014)
- http://sl.wikipedia.org/wiki/Arduino (23. 10. 2014)
- [2] http://42bots.com/competitions/arduino-line-following-code-video/ (07. 10. 2014)