»Mladi za napredek Maribora 2019«
36. srečanje

Tank voden z uporabo wi-fija

Raziskovalno področje ELEKTROTEHNIKA, ELEKTRONIKA

Inovacijski predlog

PROSTOR ZA NALEPKO

Avtor: TINE LUBEJ , KRISTJAN ŽIGART, JANEZ KEČEK

Mentor: MARJAN URANJEK
Šola: SREDNJA ELEKTRO-RAČUNALNIŠKA ŠOLA MARIBOR
Število točk: 142
Mesto: 2
Priznanje: srebrno

Maribor, 2019

»Mladi za napredek Maribora 2019«
36. srečanje

Tank voden z uporabo wi-fija

Raziskovalno področje ELEKTROTEHNIKA, ELEKTRONIKA

Inovacijski predlog

PROSTOR ZA NALEPKO

Maribor, 2019

1.KAZALO

Vsebina

2.KAZALO SLIK

3. ZAHVALA

Zahvala gre našemu mentorju, ki nam je splošno omogočil da smo lahko opravili raziskovalno nalogo.

4. HIPOTEZE

- Vozilo bo mogoče upravljati neodvisno od roke.

- Roko bo možno opravljati z vsaj minimalno kontrolo uporabnika nad roko.

- Roka bo sposobna dvigovanja manjših predmetov brez večjih težav.

5. ZUNANJE NAPRAVE

Skozi samo izdelavo smo uporabljali več naprav, ki so pa tukaj tudi naštete in opisane.

5.1 L298N dual H bridge 2A

Je krmilna ploščica za kontrolo hitrosti in smeri delovanja motorjev. Ima pa tudi druge uporabe, kot je za uporabo kontrole svetlobe, kot na primer v LED konstrukcijah. L298N je sposoben obračanja polarnosti toka in obračanja delovanja smeri delovanja elektro motorja

5.2 ESP32

ESP32 je nizko cenovni in nizko energetski mikrokrmilnik. Na trg je prišel 6.septembra 2016. Je naslednjik mikrokrmilnika ESP8266. Ta novi krmilnik ima veliko novih funkcij od prejšnje verzije. V njem sta vgrajena Wi-Fi in Bluetooth modul hkrati. ESP32 uporablja mikroprocesor Tensilica Xtensa LX6, poznamo dvojederne različice in enojederne, vključuje vgrajena antenska stikala, RF balun, ojačevalnik moči, nizko šumni sprejemni ojačevalnik, filtre in module za upravljanje porabe energije,… ESP32 je ustvarilo in razvilo podjetje Espressif Systems, kitajsko podjetje s sedežem v Šanghaju. ESP32 žal ni standardiziran, zaradi tega je več različnih modelov in je treba biti previden katero ploščo imate natanko.

6. KONČNI CILJ

Končni cilj naše naloge je bil ustvariti vozilo z gosenicami in pa nekakšno dodatno funkcijo katerega pa je mogoče kontrolirati preko spletne strani z telefona, tablice, itd. Medtem ko je na originalni prijavnici napisan top smo se skozi izdelavo odločili da top zamenjamo za robotsko roko, saj smo bili mnenja da bi bila robotska roka zelo bolj praktična, zanimiva za upravljalca in pa da bo ta ideja predstavljala večji izziv, kot pa bi navaden top. Med samim delom smo bili mnenja in še vedno smo da je to bila prava odločitev.

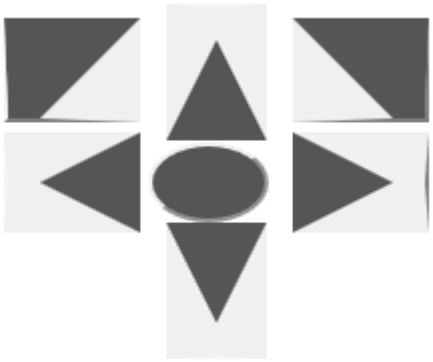
7. PROGRAMIRANJE

Program je duh vsakega robota in zato smo morali tudi sprogramirati našega robota. Večina programiranja je potekala skozi ARDUINO in pa še HTML zato da smo lahko naredili obliko strani skozi katero je mogoče upravljati robota.

7.1 HTML PROGRAMIRANJE

Kot smo že omenili zgoraj je bil HTML uporabljen za izdelavo strani preko katere je bilo potem mogoče upravljanje vozila. Seveda pa je vse moralo iti skozi program ARDUINOTA in smo zato bili omejeni pri izgledu strani. Originalna ideja je bila da bi bila stran narejena z slikami vendar pa to ni bilo mogoče in smo zato morali ti nazaj na osnovne like, ki jih ponuja HTML.

Osnova tega je seveda bila kontrola vozila. Zato se je uporabila oblika z puščicami z kontrolami: naprej, levo, desno, nazaj, stop, ostro levo in ostro desno. Zato da so v pravilni obliki uporabimo preglednico. Izgled kontrole vozila je sledeč:



Slika 1: Slika kontrole vozila na spletni strani

Z vidika strani gre le za gumbe z dodatno prevleko v obliki teles. Z malo dodatnega spreminjanja v ARDUINO programu pa lahko tudi to kar polepšamo, kot pa kako zgleda tukaj.

Kot smo pa rekli gre bistveno le za uporabo linka (anchor) z obliko gumba in prevleko lika. Lik le zgoraj definiramo kot kaj z tem želimo in na za gumb spremenimo oznako class na ime z katerim smo definirali lik.

```
.triangle-left {
    width: 0;
    height: 0;
    border-top: 25px solid transparent;
    border-right: 50px solid #555;
    border-bottom: 25px solid transparent;
}
.triangle-right {
    width: 0;
    height: 0;
    border-top: 25px solid transparent;
    border-left: 50px solid #555;
    border-bottom: 25px solid transparent;
}
.circle {
  height: 50px;
  width: 60px;
  background-color: #555;
  border-radius: 50%;
}
```

Slika 2: Slika kode definiranja lika

```
<td>
<a href="ObratLevo"><button class="triangle-left" style="height:40px" onclick="moveObratLevo()">

</button> </a> </td>
<td><a href="Stop"> <button class="circle"  style="height:40px"   onclick="StopRobot()">
</button></a></td>
<td><a href="ObratDesno"><button class="triangle-right" style="height:40px" onclick="moveObratDesno()">

</button> </a></td>
```

Slika 3:Slika kode za gumbe z zgoraj definiranimi liki

Naslednji del so drsniki, ki jih uporabljamo za hitrost vozila in pa za kontrolo robotske roke. Za kontrolo roke potrebujemo tri z dodatnim gumbom zato da se roka odre ali zapre. Drsnik za hitrost pa mora biti samo eden.

Za sam izgled drsnika gremo pa spet samo definiramo kako hočemo da izgleda in potem ime, ki določimo damo pod class.

```
.slider{
-webkit-appearance: none;
width: 50%;
height: 20px;

background: #d3d3d3;
outline: none;

}
.slider:hover {

  opacity: 1;
}

.slider::-moz-range-thumb {
  width: 20px;
  height: 20px;
  border-radius: 50%;
   background: #0018f7;
  cursor: pointer;
}
```

Slika 4 : Slika kode za obliko drsnikov

```
<p> <!--hitrost-->
HITROST
<br></br>
<input type="range"
min="180"
max="255"
id="motorSlider"
step="15"
class="slider"
onchange="motorSpeed(this.value)"
value=  vrednostNisa  />
</p>

<p> <!--drsnik-->
DRSNIK:
<br></br>
<input type="range"
min="0"
max="180"
class="slider"
id="base"
step="10"
/>
```

Slika 5:Slika kode za drsnike

Za vsak drsnik pa je potrebno dodati nekaj JAVA script-a. Z njim poskrbimo da gredo podatki dam kamor morajo iti.

```
<script>
function motorSpeed(PWM) { <!--hitrost-->
                           var xhr = new XMLHttpRequest();
                           xhr.open("GET", "vrednost=" + PWM + "&" ,true) ;
                           xhr.send(); location.reload(true);}
function basel(vrednost) { <!--drsnik-->
 var xhr = new XMLHttpRequest();
 xhr.open('GET', "valueBase" + vrednost + "&", true);
 xhr.send();
location.reload(true); }

function Ramal(vrednost) { <!--rama-->
 var xhr = new XMLHttpRequest();
 xhr.open('GET', "valueRama" + vrednost + "&", true);
 xhr.send();
location.reload(true); }

function Komolecl(vrednost) {   <!--komolec-->
 var xhr = new XMLHttpRequest();
 xhr.open('GET', "valueKomolec" + vrednost + "&", true);"
 xhr.send();
 location.reload(true);
</script>
```

Slika 6: Slika JAVA script-a

7.2 ARDUINO PROGRAMIRANJE

Program bo vzpostavil zaprto dostopno točko. Na to točko se povežemo lahko z mobilno napravo. V brskalnik vpišemo dan IP naslov in se prikaže spletna stran. Na spletni strani so podani osnovni ukazi za premik vozila. Pod temi ukazi so spletni drsniki za spreminjanje pozicije posameznih servo motorjev in tipka za odpiranje ali zapiranje ročke robotske roke.

```
#include <WiFi.h>// wifi knjižnica

WiFiServer server(80);// port
```

Slika 7: Slika kode ARDUINO #1

Uporabimo WiFi knjižnico, kjer se nahajajo funkcije, ki jih bomo potrebovali za vzpostavitev dostopne točke. Nato je bilo nastavljeno skozi kateri port bo upravljal strežnik, priporočen je port 80 saj po njem poteka HTTP protokol.

```
// ime in geslo dostopne točke
const char* ime = "TankBabyESP32";
const char* geslo = "123456789";
```

Slika 8: Slika kode ARDUINO #2

Nastavljeno je bilo ime dostopne točke, ter pod tem imamo geslo za dostop do dostopne točke.

```
String glava;// brskalna glava
```

Slika 9: Slika kode ARDUINO #3

Uporabljen je bil niz(String) z imenom glava, saj bomo iz naslovne vrstice v brskalniku brali poslane ukaze.

```
String StanjeClaw = "zaprto";// stanje servo ročke
```

Slika 10: Slika kode ARDUINO #4

Niz, ki nam bo izpisal stanje servo motorčka »claw«. To je ročka katera nam, ki se bo odpirala ali zapirala.

```
// servo
const int base = 19;
const int rama = 18;
const int komolec = 5;
const int claw = 17;
```

Slika 11: Slika kode ARDUINO #5

Definirani so elementi, na katerem pinu bodo delovali. Tu so predstavljeni posamezni servo motorčki robotske roke.

```
// Motor 1
const int motor1a = 13;
const int motor1b = 12;
const int NadzorM1 = 26;

// Motor 2
const int motor2a = 14;
const int motor2b = 27;
const int NadzorM2 = 25;
```

Slika 12: Slika kode ARDUINO #6

Tu pa sta predstavljena enosmerna motorčka.

```
// servo vrednosti niza
String vrednostNizaBase = "80";
String vrednostNizaRama = "90";
String vrednostNizaKomolec = "150";
```

Slika 13: Slika kode ARDUINO #7

Ti nizi bodo uporabljeni kot vrednosti na katerem bo stal spletni drsnik na spletni strani.

```
// pretvorjena servo vrednost iz niza
int baseVrednost = 19;
int ramaVrednost = 16;
int komolecVrednost = 33;
int clawVrednost = 17;
```

Slika 14: Slika kode ARDUINO #8

To je pretvorjena vrednost niza, katero bodo brali servo motorčki. Ta vrednost je le začetna vrednost, kar pomeni da jo mi lahko spremenimo preko ustvarjene spletne strani.

```
// PWM lastnosti servo motorčkov
const int servoFrekvenca = 50;
const int baseKanal = 0;
const int ramaKanal = 1;
const int komolecKanal = 2;
const int clawKanal = 3;
const int resolucija = 8;
```

Slika 15: Slika kode ARDUINO #9

Za razliko od Arduino plošče moramo pri ESP32 sami nastaviti frekvenco, kanal in resolucijo PWM signala. Frekvenca 50Hz je primerna za servo motor. Nastavimo na kanale, skozi katerih bodo tekli posamezni signali. Resolucija je nastavljena na 8 bitov, to nam omogoči vrednost od 0 do 255.

```
// PWM lastnosti DC motorjev za vozilo
const int MotorFrekvenca = 30000;
const int pwmKanalM1 = 4;
const int pwmKanalM2 = 5;
```

Slika 16: Slika kode ARDUINO #10

Pri enosmernih motorjih pa je primerna frekvenca 30000Hz.

```
// vrednost PWM signala za DC motorja
String ciklM1 = "0";
String ciklM2 = "0";
```

Slika 17: Slika kode ARDUINO #11

Niza »ciklM1« in »ciklM2« sta uporabljena kot izpisani vrednosti na spletni strani, za posamezno vrednost motorja1 in motorja2.

```
// iskanje vrednosti
String vrednostNizaPWM = "180";//iskanje PWM vrednosti preko spleta
int pos1 = 0;
int pos2 = 0;
```

Slika 18: Slika kode ARDUINO #12

Dve spremenljivki sta bili ustvarjeni, da poiščeta vrednost PWM signala za enosmerna motorja, ki jo bomo pošiljali preko spletnega drsnika. Niz »vrednostNizaPWM« pa je začetna vrednost PWM signala, vendar bo vozilo stalo.

```
void setup() {

  Serial.begin(115200);// baud hitrost
  delay(2000);// čas da odpremo Serial Monitor
  //-----------------------------------------------------------------
```

Slika 19: Slika kode ARDUINO #13

Hitrost serijske komunikacije je nastavljeno na 115200 bitov na sekundo, to je primerna hitrost za ta mikrokrmilnik. Nastavljen je tudi čas, ki nam odpremo serijski monitor v Arduino programskem okolju.

```
// vstavljanje PWM lastnosti v sistemsko funkcijo
ledcSetup(baseKanal, servoFrekvenca, resolucija);
ledcSetup(ramaKanal, servoFrekvenca, resolucija);
ledcSetup(komolecKanal, servoFrekvenca, resolucija);
ledcSetup(clawKanal, servoFrekvenca, resolucija);

ledcSetup(pwmKanalM1, MotorFrekvenca, resolucija);
ledcSetup(pwmKanalM2, MotorFrekvenca, resolucija);
```

Slika 20: Slika kode ARDUINO #14

V sistemsko funkcijo »ledcSetup« so vstavljeni potrebni parametri za nastanek PWM signala, v prvem parametru je kanal, v drugem je frekvenca in tretjem resolucija.

```
// definiranje pinov
ledcAttachPin(base, baseKanal);
ledcAttachPin(rama, ramaKanal);
ledcAttachPin(komolec, komolecKanal);
ledcAttachPin(claw, clawKanal);

ledcAttachPin(NadzorM1, pwmKanalM1);
ledcAttachPin(NadzorM2, pwmKanalM2);
```

Slika 21: Slika kode ARDUINO #15

S sistemsko funkcijo »ledcAttachPin« sta nastavljena dva parametra, prvi parameter vsebuje pin oziroma element kateremu se bo stanje spreminjalo, v drugem pa kanal skozi katerega bo tekel PWM signal.

```
// določanje skozi kateri kanal bo tekla njihova PWM vrednost
ledcWrite(baseKanal, baseVrednost);
ledcWrite(ramaKanal, ramaVrednost);
ledcWrite(komolecKanal, komolecVrednost);
ledcWrite(clawKanal, clawVrednost);
```

Slika 22: Slika kode ARDUINO #16

S sistemska funkcijo »ledcWrite« vstavljamo skozi kateri kanal bo tekel določeni PWM signal. Prej so bile definirane te vrednosti.

```
// določanje izhodov
pinMode(motor1a, OUTPUT);
pinMode(motor1b, OUTPUT);
pinMode(motor2a, OUTPUT);
pinMode(motor2b, OUTPUT);
```

Slika 23: Slika kode ARDUINO #17

Spremenljivke za enosmerna motorja so bile definirane kot izhodne spremenljivke.

```
Serial.print("dostopna točka se vzpostavlja...");
WiFi.softAP(ime, geslo);

//IP naslov spletne strani nam poda
IPAddress IP = WiFi.softAPIP();
Serial.print("Naslov dostopne točke: ");
Serial.println(IP);

server.begin();
```

Slika 24: Slika kode ARDUINO #18

V serijskem monitorju se izpiše »dostopna točka se vzpostavlja«. V funkcijo »Wifi.softAP(ime, geslo)« so vstavljeni potrebi parametri za vzpostavitev zaprte dostopne točke. V prvem parametru je ime WiFi omrežja, ter geslo za dostop v omrežje. V vrstici »IPAdress IP = Wifi.softAPIP« ustvari IP naslov strežnika. Nato se v serijskem monitorju izpiše »Naslov dostopne točke: « in IP naslov strežnika.

```
void loop() {

  WiFiClient Odjemalec = server.available();   // iskanje odjemalcev

  if (Odjemalec) {                             // če se novi odjemalec priklopi
    Serial.println("Novi odjemalec.");          // napiši, da je novi odjemalec v Serial Monitor
    String TrenutnaVrstica = "";                // ustarjen niz za shranjevanje prihajajočih podatkov
    while (Odjemalec.connected()) {            // zanka dokler je odjemalec povezan
      if (Odjemalec.available()) {             // če so podatki za brat od odjemalca
        char c = Odjemalec.read();             // jih preberi
        Serial.write(c);                     // in izpiši v Serial Monitor
        glava += c;
        if (c == '\n') {                     // če je podatek v novi vrstici

          if (TrenutnaVrstica.length() == 0) {
            // HTTP brskalna vrstica (glava) se naj vedno začne z odzivno kodo (e.g. HTTP/1.1 200 OK)
            // odjemalec naj pričakuje tip datoteke, HTML
            Odjemalec.println("HTTP/1.1 200 OK");
            Odjemalec.println("Content-type:text/html");
            Odjemalec.println("Connection: close");
            Odjemalec.println();
```

Slika 25: Slika kode ARDUINO #19

Spremenljivka »odjemalec« bo iskala odjemalce, ki so se povezali na strežnik. Nato, če je odjemalec najden, se bo v serijskem monitorju izpisalo »Novi odjemalec«. Zahteve, ki jih bo odjemalec pošiljal strežniku, jih bo strežnik prebral oziroma spremenljivka »c«. Ob vsaki novi zahtevi pošlje strežnik novi odziv.

```
// prikaz HTML datoteke
Odjemalec.println("<!DOCTYPE html><html>");
Odjemalec.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");
Odjemalec.println("<link rel=\"icon\" href=\"data:,\">");
Odjemalec.println("<style>"
                  "html {"//nasploh slog
                  "font-family: Helvetica;"//slog pisave
                  "margin: 0px auto;"//ustvari prostor okoli elementa
                  "text-align: center;}");//pozicija besedila
```

Slika 26:Slika kode ARDUINO #20

Strežnik pošlje odjemalcu, da mu pošilja HTML datoteko, ter določene podatke glede na prikaz datoteke na zaslonu in prenosu podatkov.

```
Odjemalec.println(".button {"
                  "background-color: DodgerBlue;"//barva ozadja na tipki
                  "border: none;"// obrobitev
                  "color: white;"//barva besedila
                  "padding: 12px 28px;" //oblazinitev elementa
                  "font-size: 23px;"//velikost besedila
                  "margin: 1px;"
                  "cursor: pointer;}");//funkcija miške, ikona miške
Odjemalec.println(".button2 {"
                  "background-color: gray;}");

Odjemalec.println(".triangle-up {"
                  "width: 0;"
                  "height: 0;"
                  "border-top: 25px solid transparent;"
                  "border-right: 50px solid DodgerBlue;"
                  "border-bottom: 25px solid transparent;"
                  "transform: rotate(90deg);}");

Odjemalec.println(".triangle-down {"
                  "width: 0;"
                  "height: 0;"
                  "border-top: 25px solid transparent;"
                  "border-right: 50px solid DodgerBlue;"
                  "border-bottom: 25px solid transparent;"
                  "transform: rotate(270deg);}");
```

Slika 27: Slika kode ARDUINO #21

```
Odjemalec.println(".triangle-left {"
                  "width: 0;"
                  "height: 0;"
                  "border-top: 25px solid transparent;"
                  "border-right: 50px solid DodgerBlue;"
                  "border-bottom: 25px solid transparent;}");

Odjemalec.println(".triangle-right {"
                  "width: 0;"
                  "height: 0;"
                  "border-top: 25px solid transparent;"
                  "border-left: 50px solid DodgerBlue;"
                  "border-bottom: 25px solid transparent;}");

Odjemalec.println(".circle {"
                  "height: 50px;"
                  "width: 60px;"
                  "background-color: DodgerBlue;"//#555
                  "border-radius: 50%;}");

Odjemalec.println(".slider{"
                  "-webkit-appearance: none;"
                  "width: 70%;"
                  "height: 5px;"
                  "background: DodgerBlue;"
                  "outline: none;}");

Odjemalec.println(" </style> ");
```

Slika 28: Slika kode ARDUINO #22

S CSS-om je oblikovan slog pisave, prostor okoli elementa in pozicija besedila. Nato je oblikovana posebej tipka »button«, ki je namenjen za odpiranje ali zapiranje ročke robotske roke. Oblikovano ima barvo ozadja, obrobitev, barva črke znotraj tipke, oblazinitev, velikost črke, prostor okrog elementa, funkcija miške. »button 2« je oblikovana tipka za izklop led

diode. Pri njej je oblikovano samo ozadje. Nato so oblikovani trikotniki, ki služijo kot tipke za osnovne smeri vozila. Oblikovan je tudi slog kroga oziroma elipse, ki bo služila kot za smeri nazaj levo, nazaj desno, obračanje v desno in obračanje v levo.

```
if (glava.indexOf("GET /Naprej") >= 0) {
  Serial.println("Naprej");

  ciklM1 = cikl;
  ciklM2 = cikl;
  ledcWrite(pwmKanalM1, ciklM1.toInt());
  ledcWrite(pwmKanalM2, ciklM2.toInt());
  digitalWrite(motor1a, LOW);
  digitalWrite(motor1b, HIGH);
  digitalWrite(motor2a, LOW);
  digitalWrite(motor2b, HIGH);
}
else if (glava.indexOf("GET /Levo") >= 0) {
  Serial.println("Levo");

  ciklM1 = "255";// motor 1 bo imel najvišjo vrednost
  ledcWrite(pwmKanalM1, ciklM1.toInt());
  digitalWrite(motor1a, LOW);
  digitalWrite(motor1b, HIGH);

  ciklM2 = "190";// motor 2 bo imel vrednost 190
  ledcWrite(pwmKanalM2, ciklM2.toInt());
  digitalWrite(motor2a, LOW);
  digitalWrite(motor2b, HIGH);
}
```

Slika 29: Slika kode ARDUINO #23

V naslovni vrstici bo http funkcija »GET« iskalo niz »Naprej« ali kateri drugi ukaz. Ko je niz najden, se bo vrednost posameznega motorja spremenilo glede na dani ukaz. S sistemsko funkcijo »ledcWrite« smo določili kanal skozi katerega bo tekel posamezen PWM signal. Vrednost PWM signala pa je določena s spremenljivko »ciklM1« in »ciklM2«. S funkcijo »digitalWrite« določamo elementu stanje kot vklop ali izklop.

```
Odjemalec.println("<p><a href = \"/ObratLevo\"><button class=\"circle\" onclick=\"moveObratLevo()\">ObratL</button></a>"
                  // Naprej
                  "<a href=\"/Naprej\"><button class=\"triangle-up\" onclick=\"moveNaprej()\"></button></a>"
                  // obrat desno
                  "<a href=\"/ObratDesno\"><button class=\"circle\" onclick=\"moveObratDesno()\">ObratD</button></a></p>");

// Levo
Odjemalec.println("<p><a href=\"/Levo\"><button class=\"triangle-left\" onclick=\"moveLevo()\"></button></a>"
                  // Stop
                  "<a href=\"/Stop\"><button class=\"circle\" onclick=\"StopRobot()\">Stop</button></a>"
                  // Desno
                  "<a href = \"/Desno\"><button class=\"triangle-right\" onclick=\"moveDesno()\"></button></a></p></div>");

Odjemalec.println("<p><a href = \"/Levo_nazaj\"><button class=\"circle\" onclick=\"moveNazajLevo()\">NazajL</button></a>"
                  "<a href=\"/Levo\"><button class=\"triangle-down\" onclick=\"movenazaj()\"></button></a>"
                  "<a href = \"/Desno_nazaj\"><button class=\"circle\" onclick=\"moveNazajDesno()\">NazajD</button></a></p>");
```

Slika 30: Slika kode ARDUINO #24

Tu so postavljene tipke z njihovimi slogi. Vsaka tipka ob pritisku pošlje neko zahtevo preko HTTP protokola. Ukaz oziroma zahteva se bo izpisala v naslovni vrstici.

```
// Izpis vrednosti posameznega motorja
Odjemalec.println("<p>PWM M1:" + ciklM1 + "</p>"// izpiši PWM vrednost za motor 1
                  "<p>PWM M2:" + ciklM2 + "</p>");// izpiši PWM vrednost za motor 2
```

Slika 31: Slika kode ARDUINO #25

Na spletni strani se bo izpisala vrednost posameznega motorja.

```
// Spletni drsnik za določanje PWM signala
Odjemalec.println("<input type=\"range\""
                  "min=\"180\""// minimalna vrednost
                  "max=\"255\""// maximalna vrednost
                  "step=\"15\""// za koliko zvišati/znižati vrednost
                  "id=\"motorSlider\""// ime
                  "onchange=\"motorSpeed(this.value)\""//ob spremembi vrednosti znova pošlje podatek
                  "class=\"slider\""
                  "value=\"" + vrednostNizaPWM + "\"/>");
// Funkcija za pošiljanje PWM signala
Odjemalec.println("<script> function motorSpeed(PWM) {"
                  "var xhr = new XMLHttpRequest();"
                  "xhr.open('GET', \"/?vrednost=\" + PWM + \"&\", true);"// tako pošilja vrednost
                  "xhr.send();"// pošlji na lokacijo vrednost
                  "location.reload(true);}"// osveži spletno stran ob spremembi vrednosti
                  " </script>");
```

Slika 32: Slika kode ARDUINO #26

Tu je napisan spletni drsnik, kateremu je določena najmanjša in največja vrednost, za koliko se bo vrednost povečala ali zmanjševala in prepoznavno ime. Z oklepajema »<script>« in »</script>« je določeno, da bo vsebina vmes zapisana v JavaScript. Znotraj je funkcija, ki pošilja vrednost drsnika na strežnik v XML. Ustvarjena je spremenljivka »xhr«, ki je nova XML zahteva. Tej spremenljivki je nakazano katero vrednost naj pošlje na strežnik. Ob spremembi vrednosti drsnika se spletna stran osveži in pozicija drsnika ostane na mestu spremembe.

```
..
if (glava.indexOf("GET /odprto") >= 0) {// poišči v brskalni vrstic

  clawVrednost = map(vrednostNizaBase.toInt(), 0, 180, 0, 33);// pretvori območje
  StanjeClaw = "odprto"; // izpiši stanje
  ledcWrite(clawKanal, 7);// pretvorjena vrednost
}

else if (glava.indexOf("GET /zaprto") >= 0) {// poišči v brskalni vrstic

  StanjeClaw = "zaprto";// izpiši stanje
  clawVrednost = map(vrednostNizaBase.toInt(), 0, 180, 0, 33);// pretvori območje
  ledcWrite(clawKanal, 17);// pretvorjena vrednost
}
```

Slika 33: Slika kode ARDUINO #27

V naslovni vrstici bo http funkcija »GET« iskalo niz »odprto« ali »zaprto«.
Ko je niz najden, se spremenljivki »clawVrednost« pretvori vrednostno območje. Spremenljivka »StanjeClaw« spremeni stanje. S funkcijo »ledcWrite« pa pošlje pretvorjeno vrednost PWM signala.

```
Odjemalec.println("<p>Vrednost base:" + vrednostNizaBase + "</p>");// izpiši vrednost drsniko
Odjemalec.println("<input type=\"range\""//določitev drsnika
                  "min=\"0\""//minimalna vrednost drsnika
                  "max=\"180\""//maximalna vrednost drsnika
                  "id=\"base\""//prepoznavno ime elementa
                  "class=\"slider\""
                  "onchange=\"base1(this.value)\""//ob spremembi vrednosti znova pošlje podatek
                  "value=\"" + vrednostNizaBase + "\"/>");//izpis vrednosti

// v JavaScript ustvarimo funkcijo za izpis vrednost drsnika
Odjemalec.println("<script>"
                  "function base1(vrednost) { "//funkcija za pošiljanje vrednosti PWM
                  "var xhr = new XMLHttpRequest();"//nova zahteva z XML za HTTP
                  "xhr.open('GET', \"/?valueBase=\" + vrednost + \"&\", true);"//odpre vrednost
                  "xhr.send();"//pošlje na lokacijo vrednost
                  "location.reload(true); }"//stran se osveži, vrednosti in nastavitve se ohranijo
                  "</script>");
```

Slika 34: Slika kode ARDUINO #28

Definirani so spletni drsniki za posamezne servo motorje. Vsak ima svojo vrednostno območje, ime in funkcijo napisana v XML da pošlje vrednost drsnika.

```
if (glava.indexOf("GET /?vrednost=") >= 0) {// poišči vrednost za PWM
  pos1 = glava.indexOf('=');// poišči pozicijo "="
  pos2 = glava.indexOf('&');// poišči pozicijo "&"
  vrednostNizaPWM = glava.substring(pos1 + 1, pos2);// odvzemi vsebino vmes

  if (vrednostNizaPWM == "180") {// če vrednost enaka 180

    //se ne premikaj
    ledcWrite(pwmKanalM1, 0);
    ledcWrite(pwmKanalM2, 0);
    digitalWrite(motor1a, LOW);
    digitalWrite(motor1b, LOW);
    digitalWrite(motor2a, LOW);
    digitalWrite(motor2b, LOW);
  }
  else {
    // drugače pretvori vrednost v celo število
    cikl = vrednostNizaPWM.toInt();
    Serial.println(vrednostNizaPWM);
  }
}
```

Slika 35: Slika kode ARDUINO #29

Ko je funkcija napisana v XML poslala vrednost, jo funkcija »GET« poišče. Ko je poslana vrednost bila najdena, spremenljivka »pos1« poišče pozicijo znaka »=« in spremenljivka »pos2« poišče pozicijo znaka »&«. Ko sta bili poziciji najdeni, bo funkcija »substring« vrne vsebino med tema pozicijama oziroma vrednost. Vrnjena vsebina pa je enaka spremenljivki »vrednostNizaPWM«. Če je ta vrednost enaka 180 se vozilo nebo premaknilo, drugače naj niz pretvori v celo število spremenljivke »cikl« in jo izpiše v serijskem monitorju.

```
if (glava.indexOf("GET /?valueBase=") >= 0) { // poišči vrednost za base servo
  pos1 = glava.indexOf('=');// poišči pozicijo "="
  pos2 = glava.indexOf('&');// poišči pozicijo "&"
  vrednostNizaBase = glava.substring(pos1 + 1, pos2);// odvzemi vsebino vmes

  baseVrednost = map(vrednostNizaBase.toInt(), 0, 180, 0, 33); // pretvori območje
  ledcWrite(baseKanal, baseVrednost);// uporabi pretvorjeno vrednost
  //izpis obeh vrednosti
  Serial.println(vrednostNizaBase);
  Serial.println(baseVrednost);
}
//-------------------------------------------------------------------------------------
if (glava.indexOf("GET /?valueRama=") >= 0) { // poišči vrednost za rama servo
  pos1 = glava.indexOf('=');// poišči pozicijo "="
  pos2 = glava.indexOf('&');// poišči pozicijo "&"
  vrednostNizaRama = glava.substring(pos1 + 1, pos2);// odvzemi vsebino vmes

  ramaVrednost = map(vrednostNizaRama.toInt(), 0, 180, 0, 33);// pretvori območje
  ledcWrite(ramaKanal, ramaVrednost);// uporabi pretvorjeno vrednost
  //izpis obeh vrednosti
  Serial.println(vrednostNizaRama);
  Serial.println(ramaVrednost);
}
//-----------------------------------------------------------------------
if (glava.indexOf("GET /?valueKomolec=") >= 0) {// poišči vrednost za komolec servo
  pos1 = glava.indexOf('=');// poišči pozicijo "="
  pos2 = glava.indexOf('&');// poišči pozicijo "&"
  vrednostNizaKomolec = glava.substring(pos1 + 1, pos2);// odvzemi vsebino vmes

  komolecVrednost = map(vrednostNizaKomolec.toInt(), 0, 180, 0, 33);// pretvori območje
  ledcWrite(komolecKanal, komolecVrednost);// uporabi pretvorjeno vrednost
  //izpis obeh vrednosti
  Serial.println(vrednostNizaKomolec);
  Serial.println(komolecVrednost);
}
```

Slika 36: Slika kode ARDUINO #30

Ko je funkcija napisana v XML poslala vrednost, jo funkcija »GET« poišče. Ko je poslana vrednost bila najdena, spremenljivka »pos1« poišče pozicijo znaka »=« in spremenljivka »pos2« poišče pozicijo znaka »&«. Ko sta bili poziciji najdeni, bo funkcija »substring« vrne vsebino med tema pozicijama oziroma vrednost. Spremenljivkam za PWM signal posameznega motorja se spremeni vrednostno območje. Nato pretvorjene PWM vrednosti potujejo preko njihovih določenih kanalih.

```
            Odjemalec.println();// prazna vrstica za berljivost
            break;// izhod iz zanke
          } else { // če je novi ukaz, izbriši starega
            TrenutnaVrstica = "";
          }
        } else if (c != '\r') {  // če je vse drugo kot povratek znaka (carriage return-enter tipka)
          TrenutnaVrstica += c;      //jo dodaj v konec 'TrenutnaVrstica'
        }
      }
    }
    // počisti brskalno vrstico
    glava = "";
    // prekini povezavo
    Odjemalec.stop();
    Serial.println("Povezava je prekinjena.");
    Serial.println("");
  }
}
```

Slika 37: Slika kode ARDUINO #31

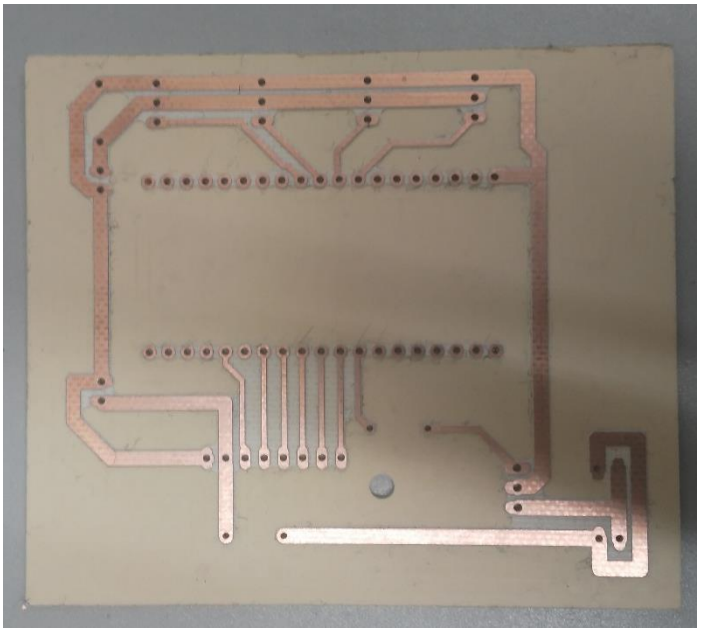
Ob vsakem ukazu, se povezava zapre.

8. IZPELJAVA

Ko je prišlo do časa izpeljave je bilo potrebno med seboj povezati vse elemente, ki smo jih zato potrebovali. Zaradi praktičnosti je bilo za ta namen izdelano tiskano vezje, ki je bilo ustvarjeno z uporabo rezkalnika v programu Eagle.

8.1 TISKANO VEZJE

Kot je bilo že omenjeno je bilo samo vezje narejeno znotraj programa Eagle, kar je pa bilo zaradi hitrejše in lažje izdelave posameznega vezja v rezkalniku v nasprotju z jedkanjem. Vezje je bilo ustvarjeno tako da vsebuje pine za mikrokrmilnik ESP32s, 1000 mikrofarad-ov kondenzator za namene glajenja, servo-motorčke, tranzistor IRLZ44N, dva upora, osempinski priključki za motorčka in 2 priključka za napajanje.

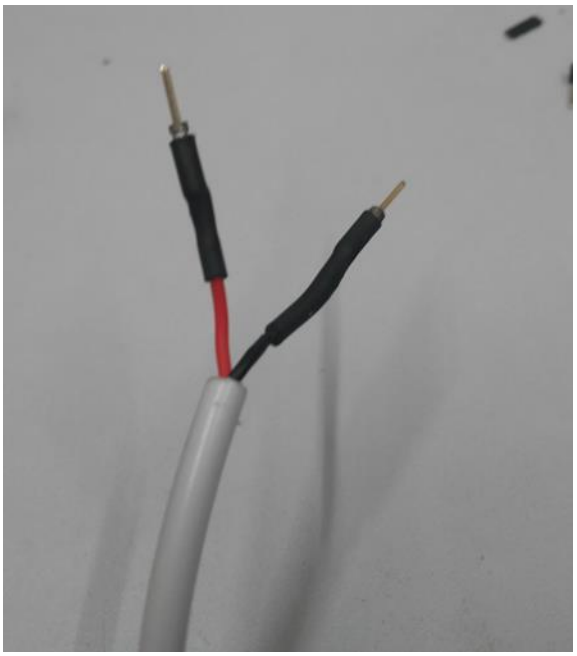


Slika 38: Slika integriranega vezja

8.2 SESTAVLJANJE

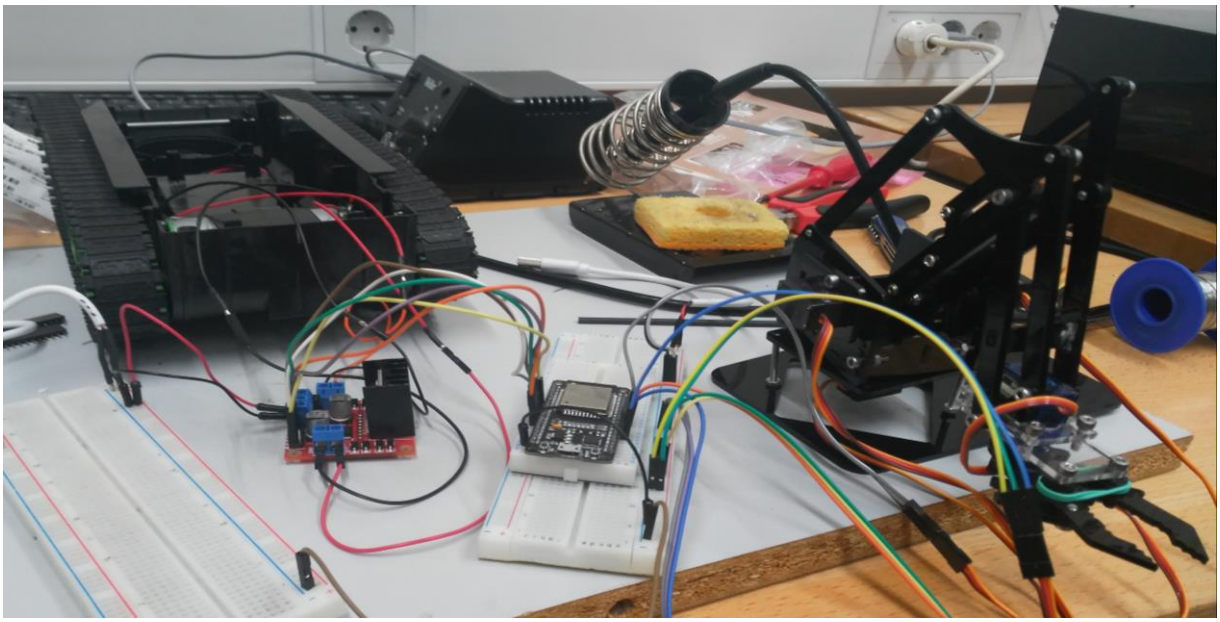
Za prototipe še seveda nismo uporabljali integriranega vezja in je za ta namen uporabljen breadboard.

Ker smo za glavni vir energije uporabljali banke za napajanje smo morali modificirati USB kabla tako, da je bila sneta izolacija z vodnikov, da je bilo za tem možno na vodnike spajkati ženski priključki , ki so imeli na drugi strani moški priključki z novo izolacijo. Te smo zatem lahko priključili v breadboard in zatem v integrirano vezje.

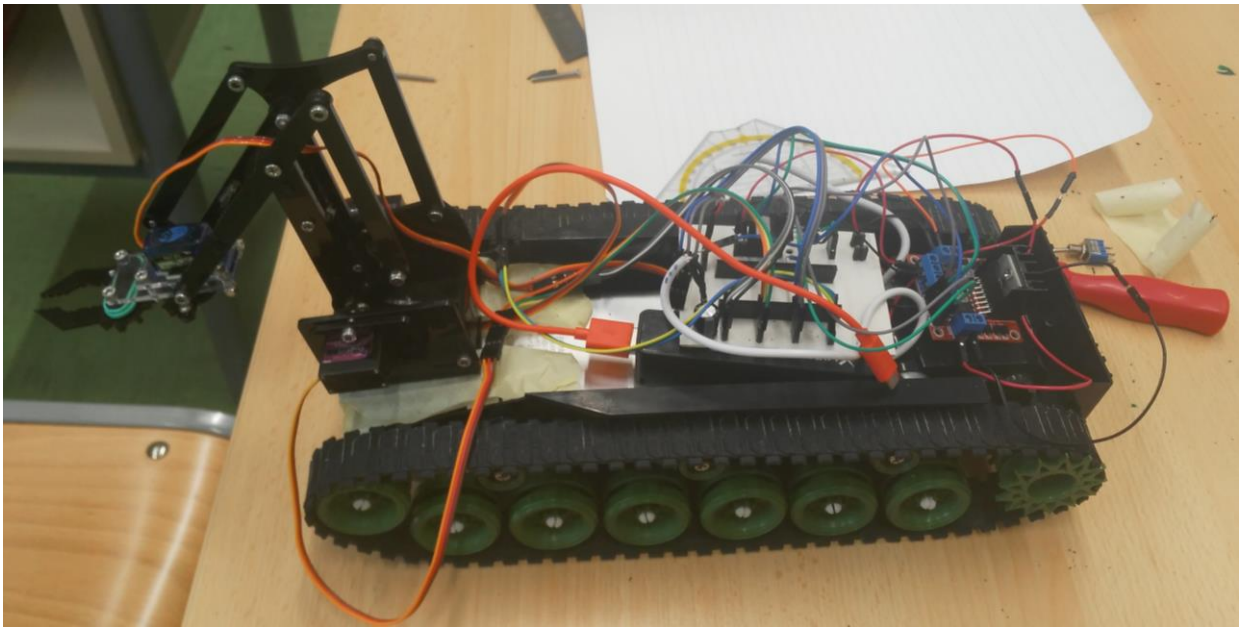


Slika 39:Slika priključkov

V ohišje sta bili nato pritrjeni banki za napajanje. Skozi ohišje je bila nato zvrtana luknja skozi katero sta potekala kabla za napajanje vezja, motorjev, itd. Za tem so bili v ohišje dani vsi elementi. Tu je bil tudi del, kjer so bili izvedeni vsi testi, saj je bilo razstavljanje relativno enostavno, kar je olajševalo reševanje problemov. Ko je bilo določeno da je vozilo v delujočem stanju smo breadboard zamenjali za ploščico.



Slika 40:Slika vseh povezav



Slika 41: Slika vozila v stanju testiranja

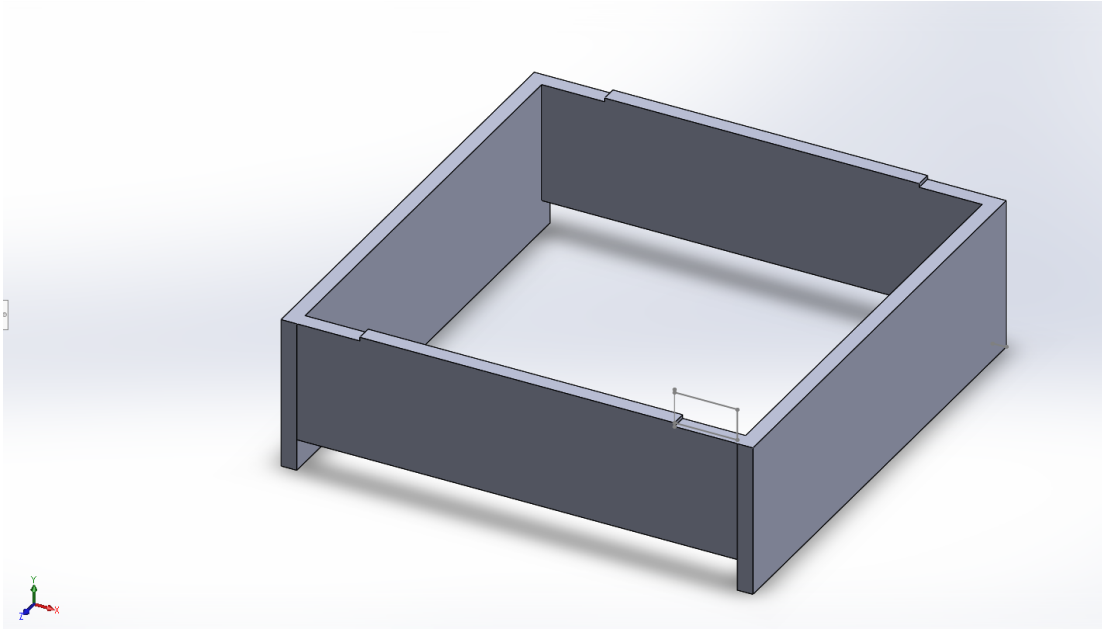
Temu je sledilo še odločitev za izgled. Ker top no bil več del vozila je bilo spremenjeno tudi zunanje ohišje, saj kupola pri roki ni smiselna zato je bilo smiselni le uporabiti ploščo da zakrije banki za napajanje in drugi del ohišja, ki bo zakril preostali del vezja. Za pritrditev smo uporabili kos dvostranskega lepilnega traka na ploščama nad obema gosenicama. Postavljena je bila plošča iz pleksi stekla, ki meri približno 30x16cm. Na to plošči so 3 pravokotne odprtine, ki merijo približno 1x5cm, njihov namen je prehod kablov pod plošče do tiskane plošče. V pleksi ploščo so bile zvrtane 4 luknje s premerom 2.5mm, na te luknje je bila pritrjena robotska roka z vijaki. Na pleksi steklo je pritrjena tiskana ploščica z dvostranskim lepilnim trakom. V zadnjem predelu tiskane ploščice pa je pritrjen H-bridge tudi z dvostranskim lepilnim trakom. Nato so bili priključeni USB kabli za napajanje. Žičke so bile prevezane z vezicami.

9. OHIŠJE

Za izdelavo ohišja je bilo uporabljeno programsko okolje SolidWorks (CAD program), ki je uporabljeno za konstrukcijo 3D modelov. Uporabljen je bil predvsem zaradi preprostosti uporabe. Z programom je bilo tako ustvarjeno ohišje, ki je bilo kasneje ustvarjeno z 3D printerjem.

9.1 IZDELAVA MODELOV ZA 3D PRINTANJE

V programu je bilo ustvarjeno naslednje ohišje. To je opis končnega izgleda ohišja.

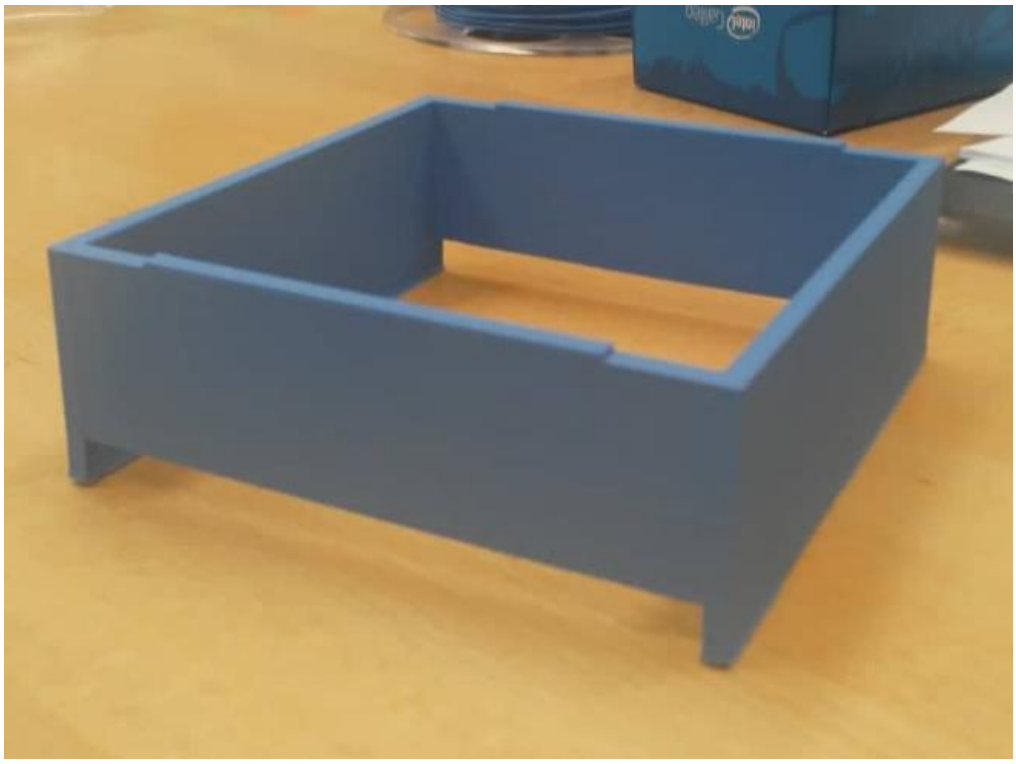


Slika 42:Slika glavnega dela ohišja v programu

Debeline stranic znašajo 5mm. Višina desne in leve plošče modela na katerega sta bila pritrjena nosilca za pokrov v višino merita 48mm, medtem ko v dolžino stranice merijo 152,5mm. Sprednja in zadnja plošča sta za 1cm dvignjeni z namenom, da smo lahko robotsko roko preprosto povezali z vodniki.

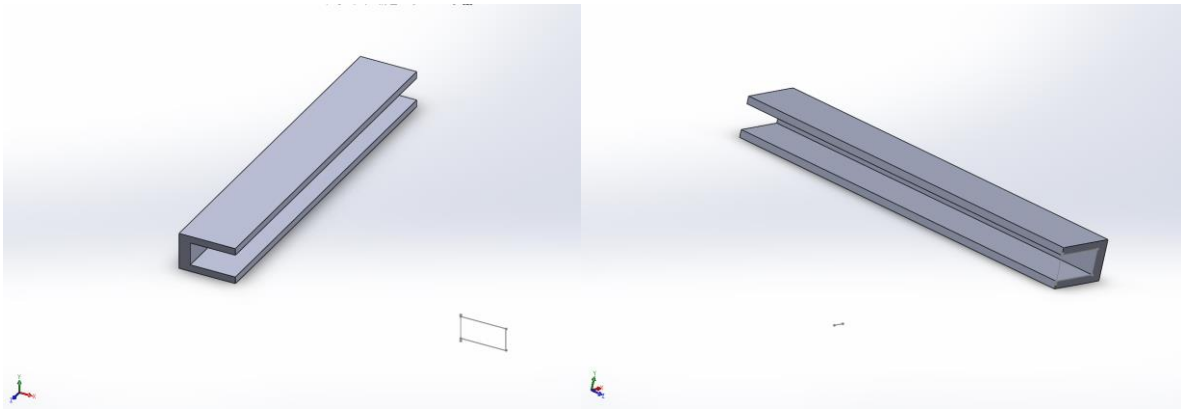
Slika 43:Slika glavnega dela ohišja v programu s strani

Sprednja in zadnja plošča sta za 1cm dvignjeni z namenom, da smo lahko robotsko roko preprosto povezali z vodniki.



Slika 44:Glavni del ohišja

Temu sledita zgornja nosilca.

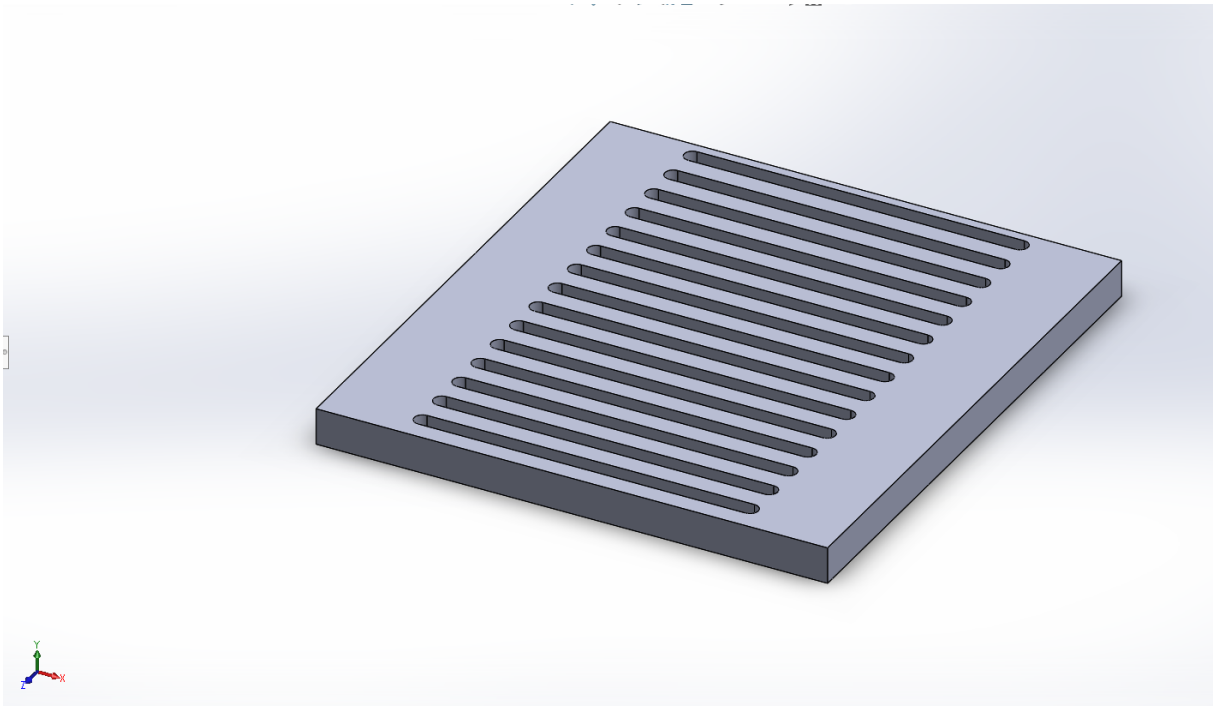


Slika 45: Slika zgornjih nosilcev v programu

Višina modelov nosilcev pokrova znaša 16mm, debelina 3mm dolžina pa 152,5mm.

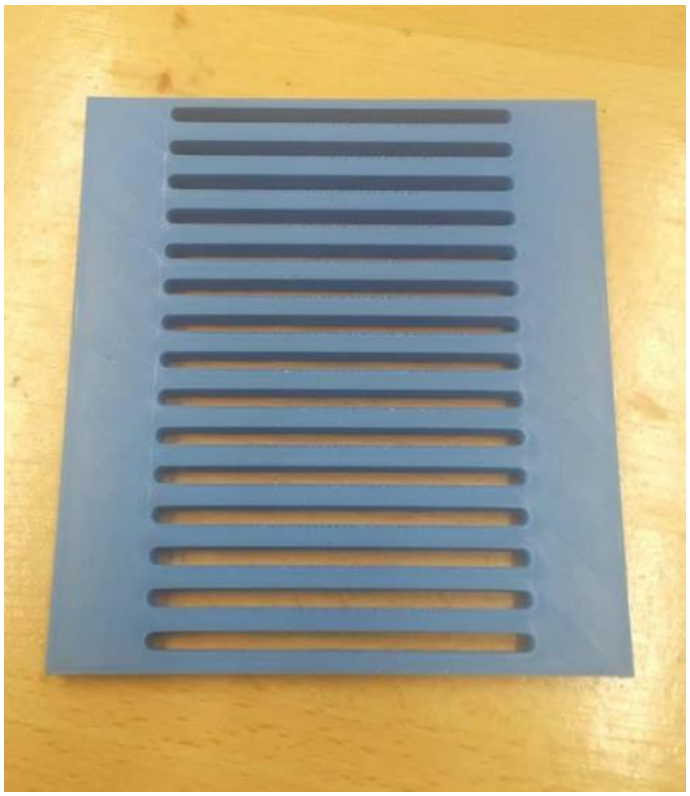
Slika 46: Slika zgornjih nosilcev

Temu sledi še pokrov z rebrmi po srednjem delu.



Slika 47: Slika rebrastega pokrova v programu

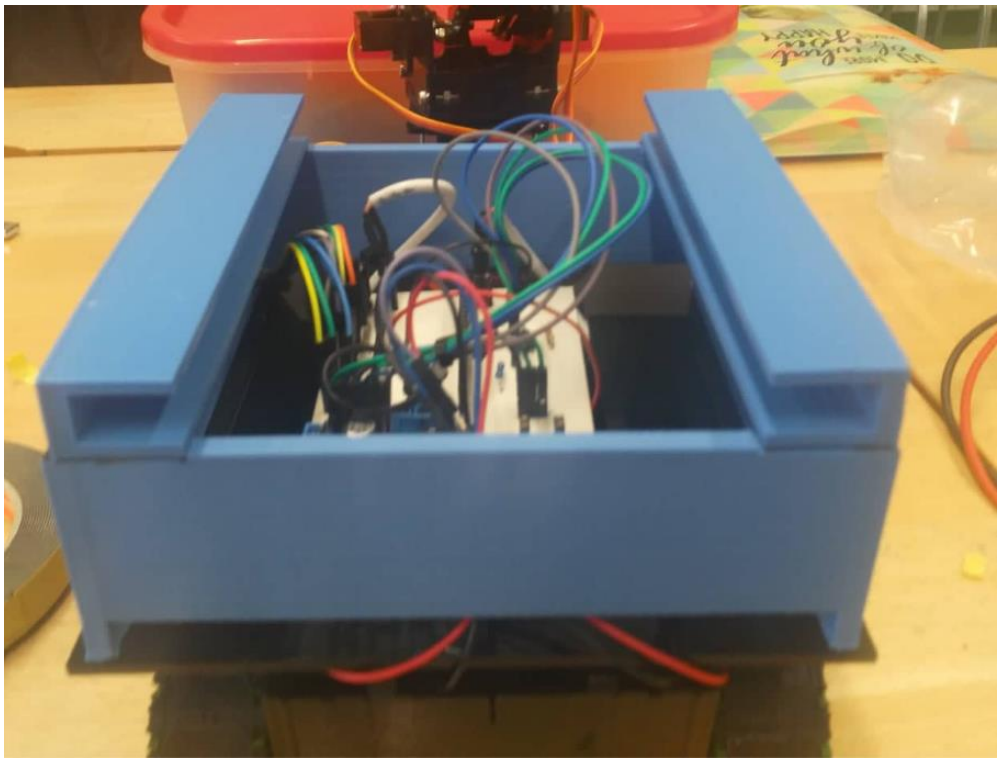
Debelina rebrastega pokrova je 10mm, da se lahko ujema z nosilci, dolžina pa znaša 152,5mm da ustreza nosilcem in glavnemu ohišju.



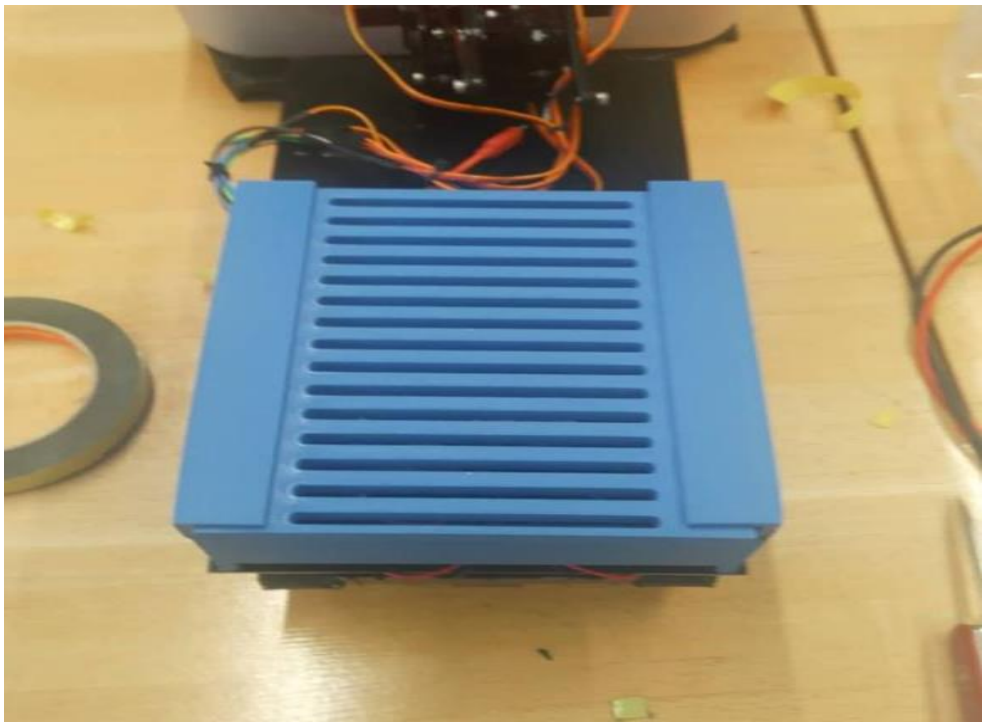
Slika 48: Slika rebrastega pokorva

9.2 SESTAVLJANJE OHIŠJA IN KONČNI IZDELEK

Kot je bilo prikazano na zgornji sliki je bil spodnji del vozila pokrit z plastično ploščo na katero sta potem pritrjeni vezji. Sedaj imamo spodnji del vozila in zgornjo ohišje. Glavni del ohišja je bil pritrjen z dvostranskim lepilnim trakom na ploščo nato sta bila pritrjena oba zgornja nosilca, ki delujeta kot nekakšen predal.



Slika 49: Slika pritrjena ohišja brez pokrova



Slika 50: Slika pritrjenega ohišja z pokrovom

10. UGOTOVITVE

Že v dobi »prototipa« smo seveda že testirali ali je naše vozilo sposobno naših minimalnih zahtev. Samo vozilo je bilo sposobno osnovnega premagovanja ovir in plezanja. Kar se pa tiče naših hipotez pa so bile ugotovitve naslednje:

Vozilo bo mogoče upravljati neodvisno od roke:

Vozilo je mogoče opravljati neodvisno od roke in obratno.

Roko bo možno opravljati z vsaj minimalno kontrolo uporabnika nad roko:

Roko lahko uporabnik upravlja v vsaj minimalne pričakovanju. Drsniki omogočajo da je osnovno upravljanje roke preprosto vendar pa je počasno in ni najbolj natančno.

Roka bo sposobna dvigovanja manjših predmetov brez večjih težav:

Roka je sposobna dvigovanja manjših predmetov brez večjih problemov.

12. VIRI

- **http://www.robotpark.com/L298N-Dual-H-Bridge-Motor-Driver** (28.12.2018)
- https://tronixlabs.com.au/news/tutorial-l298n-dual-motor-controller-module-2a-and-arduino/ (28.12.2018)
- https://en.wikipedia.org/wiki/ESP32 (28.12.2018)
- https://www.espressif.com/en/products/hardware/esp32/overview (28.12.2018)
-Rui Santos, Sara Santos: LEARN ESP32 with Arduino IDE