Úvod:

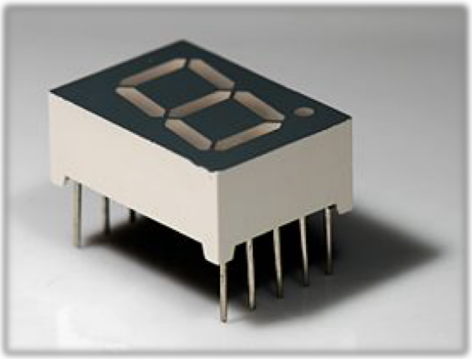
Ako je všeobecne známe v dnešnom svete je veľmi populárne hranie počítačových hier či už on-line alebo off-line medzi mládežou/ mladými ľuďmi. Veľkým lákadlom sú taktiež simulačné programy, hry automobilov s využitím herných ovládačov (volant, pedále, riadiaca páka, ručná brzda), ktorých cena neustále rastie.

Práve preto sme sa rozhodli navrhnúť a skonštruovať lacnejšiu verziu na báze Arduina, ktorá plnohodnotne nahradí drahé herné ovládače. Nami navrhnutý program pre komunikáciu Arduina s počítacom je kompatibilný s množstvom dnešných simulačných hier. Z tohto dôvodu je pre odbornejšieho používateľa výhodnejšie používať pravé Arduino aj kvôli širokým možnostiam programovania vstupov a výstupov. Oproti zakúpenému produktu je nevýhodou nutnosť vlastného návrhu, konštrukcie pre pedále, volant, riadiacu páku, ručnú brzdu.

Pre tento projekt sme si na internete našli vhodnú knižnicu pre zostrojenie. Samotný zdrojový kód sme si museli napísať sami. Pre detegovanie Arduina ako ovládač, sme ho najprv museli dať do DFU (Device Firmware Update). Následne sme použili program, vďaka ktorému náš počítač nedetegoval Arduino ako Arduino, ale ako herný ovládač. V tomto móde nemôžeme ďalej upravovať program v Arduine. Musíme preto urobiť spätné preprogramovanie.

Program sa dá využiť aj na používanie pre Controller emulátory. Tieto Controller emulátory sme použili v prípade, ak naša samotná hra nevedela detegovať Arduino.

Pre riadiacu páku sme použili tlačidlá, ktoré sme dali na digitálny pin. Pre ručnú brzdu a pedále sme použili potenciometre, ktoré sme umiestnili/ zapojili na analógový pin. Pre množstvo digitálnych pinov sme museli niektoré analógové piny naprogramovať na digitálne. K riadiacej páke sme pridali aj 7 segmentový display. Ten je riadený cez Arduino Nano. Táto 7 segmentovka nám slúži na zobrazovanie rýchlosti.



V programe Eagle sme si zhotovili schému zapojenia. Taktiež sme zhotovili jednu hlavnú schému len pre zapojenie a potom schémy pre zhotovenie plošných spojov.
Kľúčové slová: Arduino, DFU, Controller Emulátory, piny, display, Eagle

1. Arduino

Arduino je open-source platforma založená na mikrocontroléri ATMega od firmy Atmel s grafickom vývojovom prostredí, ktoré vychádza z prostredia Wiring (podobný projekt ako Arduino, teda doska s mikrocontrolérom a IDE) a Processing (prostredie pre výučbu programovania).

Arduino môže byť použité k vytváraniu samostatných interaktívnych zapojení alebo môže byť pripojené k softvéru na počítači (napr. Macromedia Flash, Processing, Max/MSP, Pure Data, SuperCollider). Momentálne možno kúpiť verzie, ktoré sú už kompletné; schéma a návrh plošného spoja je dostupný pre tých, ktorí si chcú postaviť Arduino sami.

1.1 Software

Vývojové prostredie Arduina (IDE) je viacplatformová aplikácia naprogramovaná v Jave. Je navrhnuté tak, aby umožnilo programovať aj ľuďom, ktorí nemajú veľké skúsenosti s programovaním. Obsahuje editor kódu s bežnými vlastnosťami ako farebné označovanie syntaxe, automatické zarovnávanie a párovanie zátvoriek. Je schopné program skompilovať a nahrať do Arduina jedným kliknutím tlačidla.

Programy pre Arduino sa píšu v jazyku C alebo C++. IDE obsahuje knižnicu funkcií, ktoré uľahčujú písanie najzákladnejších operácii s hardvérom. Užívateľ musí definovať iba dve funkcie, aby sa získal spustiteľný program:

- setup () : funkcia, ktorá sa spúšťa iba raz na začiatku programu a používa sa na nastavenie parametrov

- loop () : funkcia, ktorá je periodicky spúšťaná, pokiaľ je mikrocontrolér pripojený ku zdroju elektrickej energie.

1.1.1 IDE

Na vývoj je možné použiť niekoľko verzií IDE. Je vhodné používať najaktuálnejšiu verziu, pretože v starších IDE nie sú vykonávané žiadne úpravy.

IDE umožňuje programovať aj ďalšie mikrocontroléry. Napríklad mikrocontroléry ESP8266 so zabudovanou podporou Wi-Fi. Je to vďaka otvorenej architektúre umožňujúcej pridávať ľubovoľné balíky programov (nazývané toolchain), ktoré vedia skompilovať program pre konkrétny cieľový mikrocontrolér.

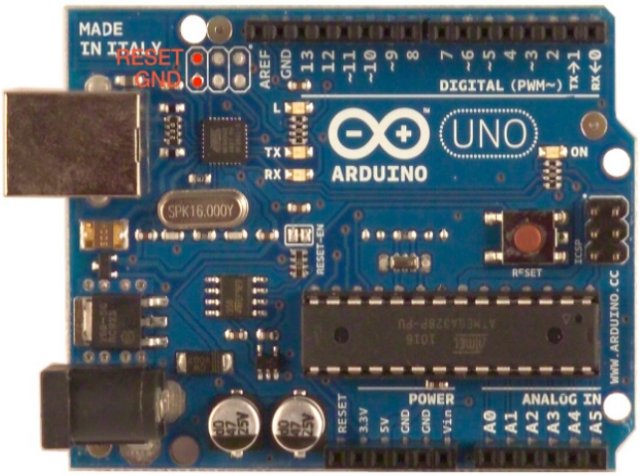
1.1.2 Knižnice

Programovanie Arduina uľahčuje existencia obrovského množstva knižníc. Knižnica je ucelený zdrojový kód, ktorý je obvykle umiestnený na serveri GitHub. Okrem zdrojového kódu v C++ je doplnená metaznačkami, ktoré umožňujú IDE indexovať tieto zdrojové kódy a poskytovať ich na pohodlnú inštaláciu prostredníctvom Manažéra knižníc. Knižnica obvykle poskytuje ucelenú sadu funkcií alebo tried určených na ovládanie konkrétneho hardvéru.

1.1.3 DFU

Čip ATmega16U2 na doske Arduino funguje ako most medzi USB portom počítača a sériovým portom hlavného procesora. Predchádzajúce verzie zariadení Uno a Mega2560 mali Atmega8U2. Spúšťa softvér nazývaný firmware (pomenovaný preto, že sa nedal zmeniť, akonáhle bol naprogramovaný v čipe), ktorý možno aktualizovať pomocou špeciálneho USB protokolu nazvaného DFU (Device Firmware Update).

Musíme si stiahnuť software. Tento software je od Atmelu nazývaný Atmel Flip. Tento program si následne musíme nainštalovať. Potom musíme resetovať naše Arduino tak, že spojíme piny RESET a GND. Tieto piny sa nachádzajú za USB.



Potom použijeme Flip program, aby sme doň nahrali hex súbor (arduino-usbserial/Arduino-usbserial-uno.hex alebo arduino-usbserial/Arduino-usbserial-mega.hex) podľa toho, ktorý typ mikrocontroléra máme. Po nahratí nám už len stačí odpojiť a znovu pripojiťArduino.

2. Problematika

Najväčším problémom bolo nájsť ten správny projekt, ktorý by sme vedeli využiť v praxi, budúcnosti alebo pre osobné účely. Nakoniec sme sa rozhodli pre tento projekt, keďže sme milovníkmi automobilových počítačových hier.

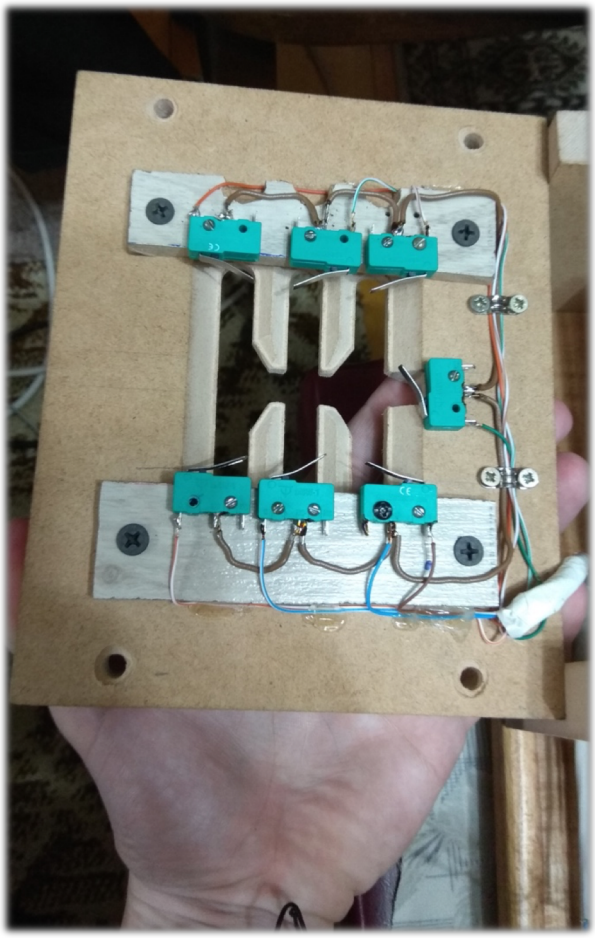
V prvom rade kým sme vôbec začali navrhovať konštrukciu, tak sme si museli najprv zhotoviť program, ktorý by bol funkčný. Preto sme si našli vhodnú knižnicu na GitHube. Táto knižnica bola veľmi dobre popísaná, čo nám uľahčilo vytvorenie zdrojového kódu. Zdrojový kód sa nám podaril v pomerne krátkom čase napísať, čo nás veľmi potešilo. Tento program si môžeme pozrieť v prílohe A. Avšak hneď prišiel na rad ďalší problém. Niektoré hry nám nedokázali identifikovať naše Arduino. Preto sme hľadali nejakú možnosť, ako by to bolo možné dokázať. Preto sme vyskúšali takzvaný Controller Emulátor (PSX, X360). My sme si pre našu prácu vybrali X360. V ňom nastal ďalší problém a to, že nám dovoľoval využívať len 4 osi. Práve kvôli tomuto problému sme sa rozhodli, že samotný volant nebudeme vyrábať. Použili sme originálny volant Logitech Driving Force GT, ktorý je lacný a plne programovateľný. Takto sme vedeli dosiahnuť aj to, že v hre sme využívali 2 ovládače naraz.

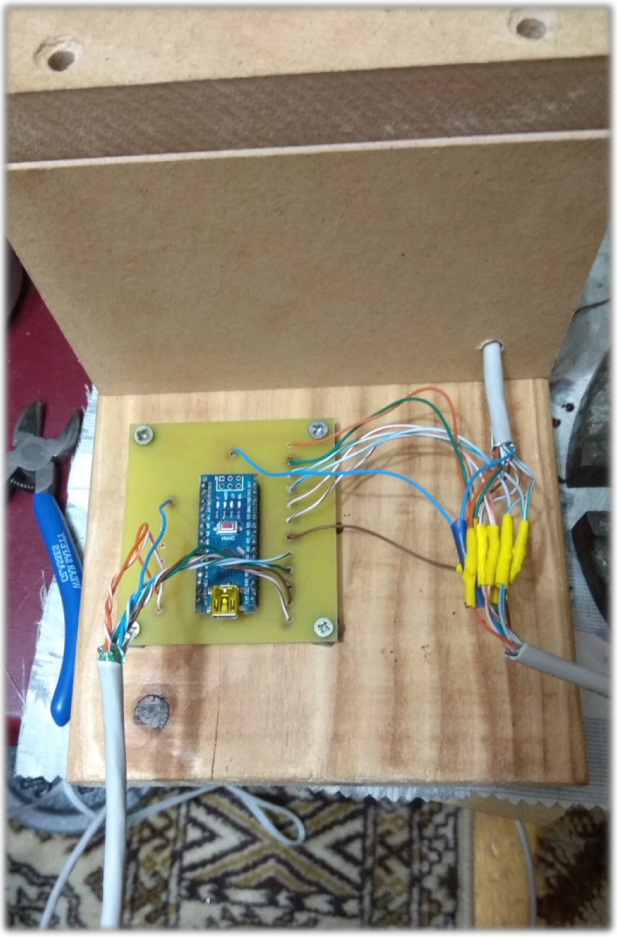
Keď sme mali programovaciu časť zhotovenú, tak sme sa mohli pustiť do navrhnutia a zhotovenia celej konštrukcie ktorej nákres je v prílohe B. Na návrh sme nepoužívali žiaden 3D program z dôvodu, že ani jeden z nás v takomto programe nevie pracovať. Konštrukciu sme si preto navrhli na papier. Taktiež sme sa inšpirovali aj originálnymi výrobkami, ktoré si vieme zakúpiť. Tento návrh sa nachádza v prílohe C. Rozmery sme si neurčili vopred vzhľadom na to, že sme nevedeli, aký materiál sa nám podarí zohnať. S hľadaním materiálov sme nemali problém. Na zistenie pozície sme použili potenciometre. Tie avšak museli mať hodnotu aspoň 10kΩ kvôli dobrej citlivosti a presnosti.

Najprv sme začali s konštrukciou ručnej brzdy, pre ktorú sme si zvolili kovový materiál. Táto ručná brzda je tvorená podľa hydraulickej ručnej brzdy. Prvý problém, ktorý sa nám vyskytol bolo to, že sme si museli zaobstarať dostatočne veľký U profil. Ten sme si nechali vyrobiť. Tento U profil musel byť dostatočne široký, aby sa nám doňho vošiel posuvný potenciometer. Použili sme pružinový systém, ktorý sa nám podaril urobiť tak, že máme takmer celý rozsah potenciometra.

Po dokončení riadiacej páky sme sa pustili do realizácie pedálov, kde sme použili drevo aj železo. Pre pedále sme museli použiť dokopy už 6 U profilov namiesto jedného ako to bolo pri ručnej brzde. Taktiež sme mali väčší problém s uložením potenciometrov, nakoľko tu sme použili otočné potenciometre.

Nakoniec sme si nechali konštrukciu riadiacej páky, pri ktorej sme použili hlavne drevo. Samotnú tyčku sme umiestnili do pružiny, aby sme dosiahli vycentrovanie. Museli sme si vymerať aj veľkosť medzier pre jednotlivé rýchlosti z dôvodu veľkosti tlačidiel. Ďalšie problémy sa nevyskytli.





3. Ciele

Ako sme už na začiatku spomínali, dnešné herné ovládače sú finančne náročne a ich cena sa môže vyšplhať na stovky eur. Naším hlavným cieľom je dokázať, že takýto herný ovládač sa dá zostrojiť za oveľa menší finančný obnos.

Pre tento cieľ sme si preto museli nájsť čo najlacnejší spôsob komunikácie s počítačom a naším zariadením. Práve preto sme si vybrali Arduino, ktoré nie je veľmi finančne náročné a má množstvo iných výhod. Medzi tie patrí napríklad to, že je to Open-Source platforma, ktorá sa jednoducho programuje. Medzi ďalšie patrí možnosť naprogramovania jednotlivých pinov podľa potreby.

Pri programovaní sme museli dbať aj na to, že niektoré hry nemusia naše Arduino detegovať. Práve preto sme museli použiť Controller Emulátor (PSX, X360) a hry, ktoré cez tento emulátor dokázal detegovať náš ovládač. V tomto emulátore sme si následne nakonfigurovali naše zariadenie a potom sme mohli šťastne hrať všetky hry.

Materiál na zostrojenie konštrukcie sme použili železo a drevo. Museli sme prihliadať na opotrebovanie jednotlivých častí, prípadnú deformáciu a na ich jednoduchosť spracovania či obrábania. Na uchytenie sme najčastejšie použili zverákový systém, ktorý je prispôsobený na prácu s rôznou hrúbkou materiálu.

4. MATERIÁL A METODIKA

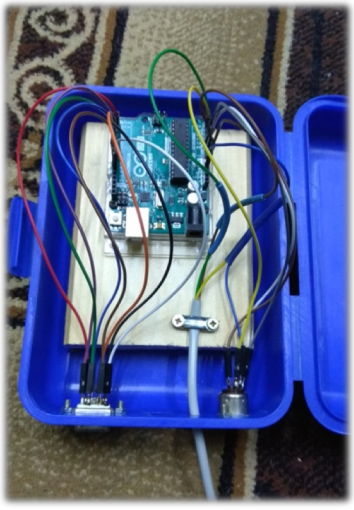
Pri návrhu konštrukcie pre pedále, ručnú brzdu, riadiacu páku sme vychádzali zo sériovo vyrábaných častí modelov. Z dôvodu jednoduchosti konštrukcie a ľahšej manipulácie sme použili len dva druhy materiálov a to železo a drevo. Najviac sme však využívali železný formát materiálu kvôli pevnosti. Využívali sme L a U profily. Konštrukciu sme si navrhli na papier z dôvodu nenáročnosti. Pre výrobu konštrukcie sme potrebovali sústruh, brúsku, skrutky rôznych veľkosti, pružinový systém do jednotlivých častí.

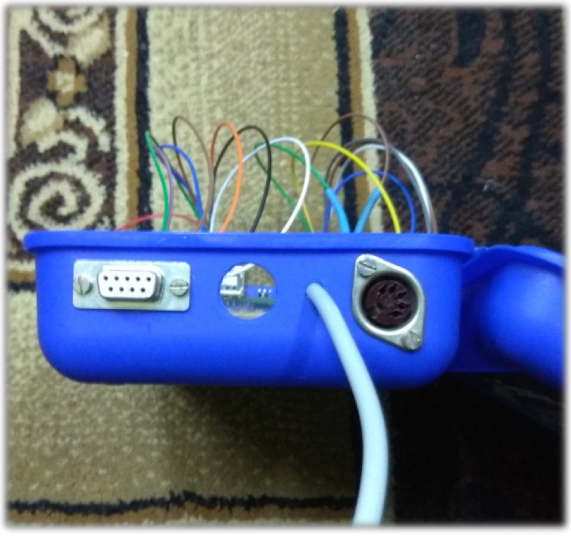
V programe Eagle sme si navrhli schému zapojenia. Taktiež sme si v tomto programe navrhli aj plošné spoje, ktoré nám výrazne uľahčili prácu pre prepojenie vodičov z konektorov do Arduina. schéma celkového zapojenia je v prílohe C, v prílohe C.1 je návrh plošného spoja pre 7 segmentový display.

Na vodiče z jednotlivých častí sme pripevnili vhodné konektory, čo nám dodá jednoduchú manipuláciu pri premiestňovaní alebo pri usporiadaní jednotlivých vodičov. Zároveň nám to dodáva jednoduchosť pri prípadnej oprave.

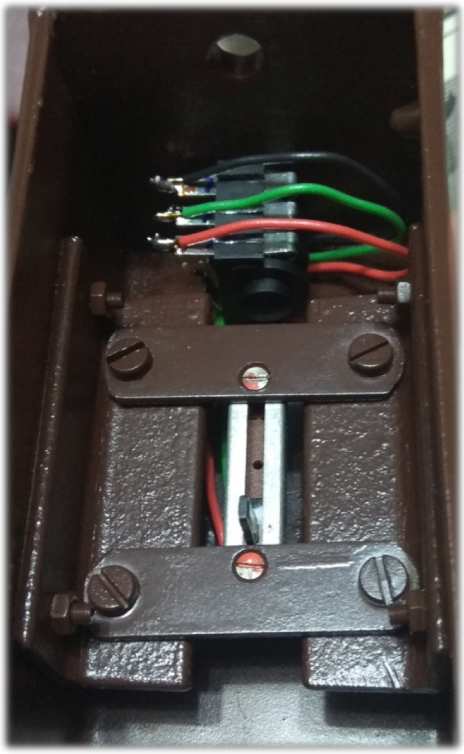
Z dôvodu surového dreva sme museli tieto tvary najprv vyhladiť a potom prispôsobiť na potrebnú veľkosť. Železo, ktoré sme použili malo častokrát narušenú vrstvu, ktorú sme museli najprv obrúsiť a potom nastriekať farbou, a taktiež sme si museli prispôsobiť veľkosť jednotlivých častí.

Pre Arduino sme si vytvorili jednoduchú krabičku, z ktorej sú vyvedené napájacie konektory pre jednotlivé časti.



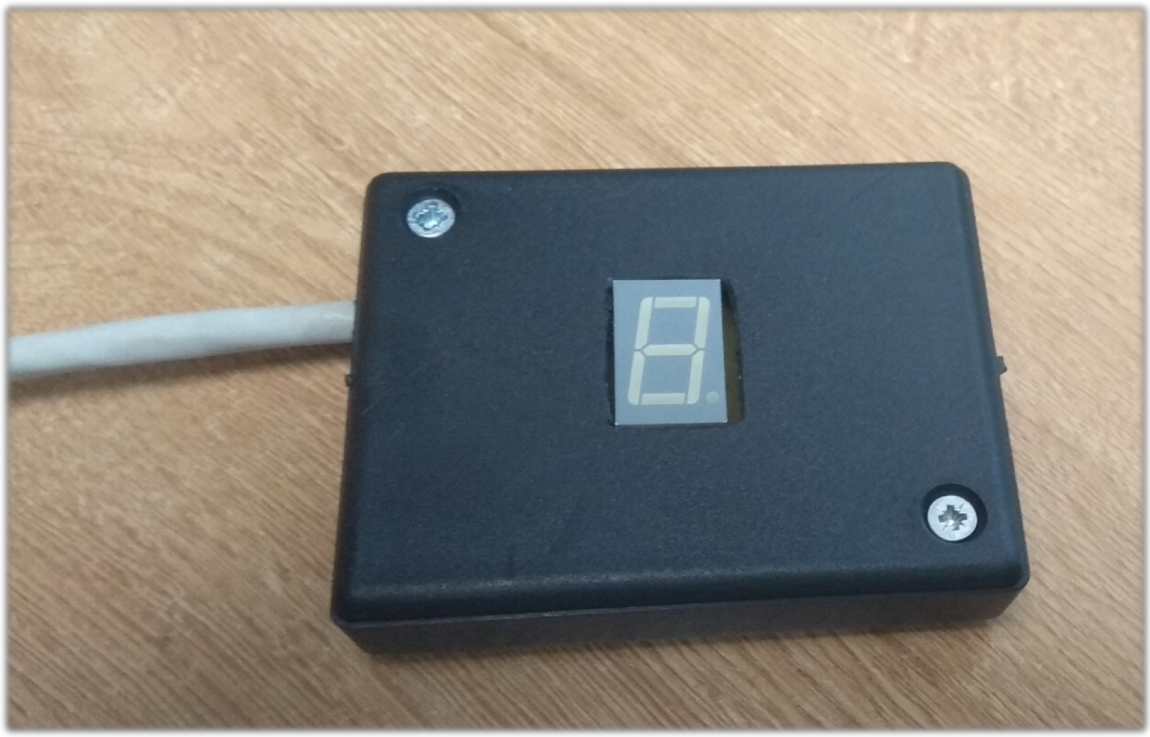


Pre ručnú brzdu, ako sme už spomínali, sme použili železo a základ kostry nám tvoril U profil. Vnútri bol uložený posuvný potenciometer tak, aby sa nám vodivé časti nedotýkali kostry. Ten sme uskočili tak, že potenciometer sme pripevnili o obdĺžnikové pliešky, ktoré boli pripevnené ako pliešky L profilu. Na posúvanie potenciometra sme použili malý pliešok L profilu, ktorý bol pripevnený o pružinový systém riadiacej páky. Pružinový systém sa nám stará o návrat páky do spätnej polohy resp. do začiatočnej polohy. Na kostru sme pripevnili zverák, ktorý nám zabezpečí pohodlne pripevnenie na rôznu veľkosť materiálu a taktiež bezproblémovú manipuláciu. Vodiče sme vyvádzali zo spodnej strany konštrukcie pre jednoduchosť.



Pre konštrukciu pedálov sme použili kombináciu železa a dreva. Samotná kostra je tvorená z dreva do L profilu a je vystužená železom, čo nám dodá pevnosť celej kostry. Avšak pre jednotlivé pedále sme použili železo. Táto kostra sa nachádza v prílohe D. Konštrukcia pedálu je tvorená z dvoch železných U profilov, pružinového systému a otáčavého potenciometra. Jeden z U profilov je napevno pripevnený ku kostre. Druhý U profil je pripevnený o prvý tak, aby sa s ním dalo hýbať a zároveň o pružinový systém. Samotný potenciometer je pripevnený o U profil, ktorý je napevno pripevnený, ale otáčavá časť je pripevnená o pohyblivú časť. Pre pedále sme sa snažili použiť rôznu tvrdosť pružín, aby sme dostali čo najreálnejší pôžitok.

Pre riadiacu páku sme taktiež použili kombináciu dreva a železa. Kostra sa skladá z dreva, ale vnútro konštrukcie je tvorené zo železa. Na ukotvenie riadiacej páky v pozícii zvolenej používateľom sme použili magnetický systém – kovové jadro v riadiacej páke bude uchytené v danej pozícií tak, aby sa nehýbalo a držalo požadovanú polohu. Z vnútornej časti horného krytu sú umiestnené spínače, ktoré nám slúžia ako snímače. Naša riadiaca páka sa skladá z dvoch častí. Prvá je samotná riadiaca páka a druhá je 7 segmentový displej. Ten je riadený cez ďalšie Arduino Nano, ktorý je uložený pod riadiacou pákou. Experimentálne zapojenie 7 segmentového displeja s Arduino nano sa nachádza v prílohe E. Tento displej nám ukazuje aktuálne zaradenú rýchlosť. Na riadiacu páku je pripevnený zverákový systém. Tu sme však použili už 2 zveráky kvôli stabilite a veľkosti samotnej konštrukcie.



5. Záver a zhrnutie

Naše ciele, ktoré sme si na začiatku stanovili sa nám podarilo dosiahnuť. Náš ovládač nás vyšiel niekoľkonásobne lacnejšie, než kúpa originálneho zariadenia. Týmto sme dokázali, že v dnešnom svete ak niečo chcete, tak sa to dá dokázať. Takto sme si jednoducho mohli vytvoriť náš samostatný herný ovládač, pri ktorom sme si vedeli prispôsobiť všetko, čo sme chceli. Počas vytvárania tejto práce sme sa naučili veľa nových poznatkov z programovania, ale aj väčšej zručnosti pri obrábaní materiálov. Naučili sme sa taktiež efektívnejšie rozhodovať v situáciách pri výbere materiálu a návrhu konštrukcie. Takýto herný ovládač nám môže dať viacero skúseností pre ovládanie vozidla, najmä však pri dnešnom veľmi obľúbenom štýle jazdenia „driftom“. Pri správnom výbere hry tak môžeme dosiahnuť pôžitok vyrovnávajúci sa realite.

Cena ktorú sme dali za náš výrobok sa pohybuje okolo 95€ zaokrúhlene. Do úvahy sme nebrali volant ktorý už máme. Ak by sme ho brali do úvahy aj náš volant tak ten stojí okolo 50€. Najdrahšia položka bola Arduino Uno, ktore stálo 26€. Samozrejme Arduino Uno sa dá kupiť aj za 10€ no pre tento projekt sme chceli využit originál z Talianska. Pre porovnanie sme použili stranku Alza.sk. Najlacnejšia riadiaca páka Logitech Driving Force Shifter stoji 49€. Najlacnejšie pedále (so spojkou, plynom aj brzdou) sú Thrustmaster T3PA Pedals a stoja 96€. Takéto pedále by nás vyšli ako skoro naša celá zostava. Ručná brzda sériovo ani nepredáva. Samozrejme, že tento setup by nebol vhodný z dôvod kombinovania Logitech a Thrustmaster. Logitech sám o sebe nepredáva samostatne pedále, čiže by sme si museli zaobstarať Logitech G29 Driving Force. Tento volant obsahuje aj pedále stojí 235€ po zľave, normálne okolo 400€. Thrustmaster pedále ktoré sme užspomínali by sa dali kombinovať s riadiacou pákou Thrustmaster TH8A Add-on shifter ktorého cena sa pohybuje okolo 140€. Volant taktiež okolo 50€.

We completed the goals that we set. Our controller cost less than the original ones. We have proven that in today´s world if you want to achieve something you can do it. This way, we could easily create our own game controller, where we could adapt everything we wanted. When we were creating this project we got a lot of new experiences from programming and also we learned to work with materials. The controller like that can give us a lot of experiences with riding a car. Mostly in drift style which is popular these days. If we choose right game we will get almost realistic feeling from ride.

6. Zoznam použitej literatúry

www.wikipedia.sk

www.github.com

www.arduino.cc

7. Prílohy

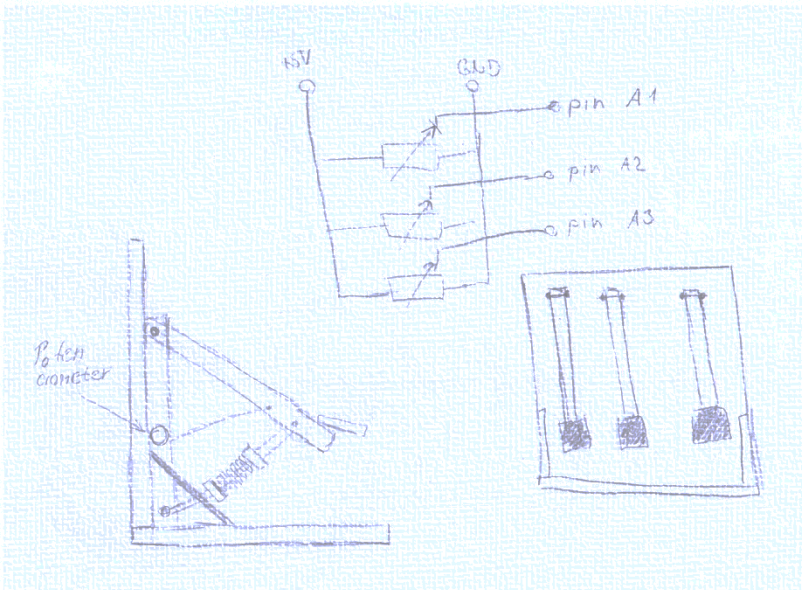
Príloha A

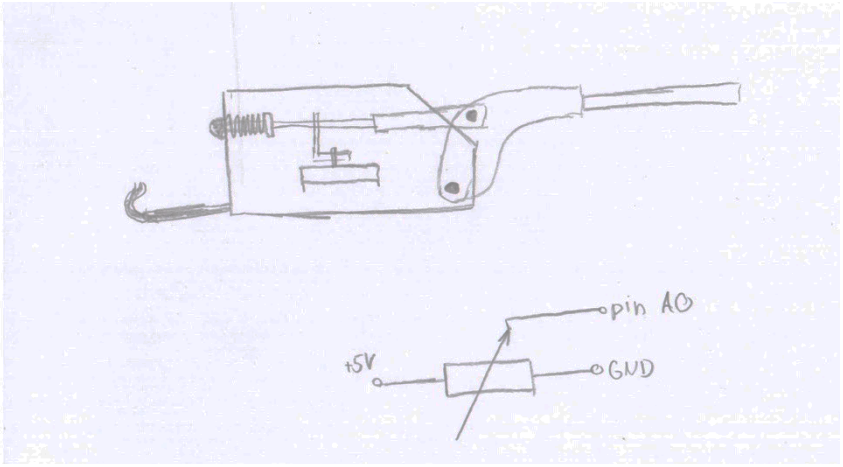
```
#ifndef UNOJOY_H
#define UNOJOY_H
    #include <stdint.h>
    #include <util/atomic.h>
    #include <Arduino.h>


  typedef struct dataForController_t
  {
    uint8_t triangleOn : 1;
    uint8_t circleOn : 1;
    uint8_t squareOn : 1;
    uint8_t crossOn : 1;
    uint8_t l1On : 1;
    uint8_t l2On : 1;
    uint8_t l3On : 1;
    uint8_t r1On : 1;
    uint8_t r2On : 1;
    uint8_t r3On : 1;
    uint8_t selectOn : 1;
    uint8_t startOn : 1;
    uint8_t homeOn : 1;
    uint8_t dpadLeftOn : 1;
    uint8_t dpadUpOn : 1;
    uint8_t dpadRightOn : 1;

    uint8_t dpadDownOn : 1;
        uint8_t padding : 7;
```

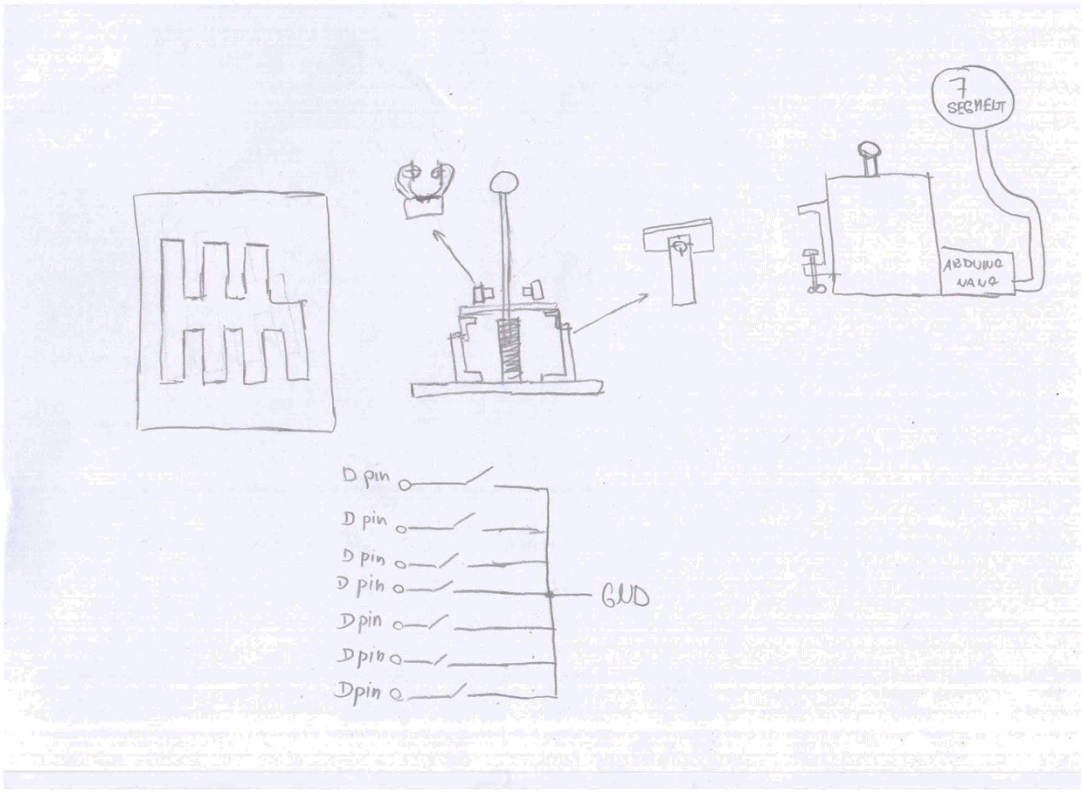
```
#include "UnoJoy.h"

void setup(){
  setupPins();
  setupUnoJoy();
}

void loop(){
  dataForController_t controllerData = getControllerData();
  setControllerData(controllerData);
}

void setupPins(void){
  for (int i = 2; i <= 12; i++){
    pinMode(i, INPUT);
    digitalWrite(i, HIGH);
  }
  pinMode(A4, INPUT);
  digitalWrite(A4, HIGH);
  pinMode(A5, INPUT);
  digitalWrite(A5, HIGH);
}

dataForController_t getControllerData(void){
  dataForController_t controllerData = getBlankDataForController();

  controllerData.triangleOn = !digitalRead(2);
  controllerData.circleOn = !digitalRead(3);
  controllerData.squareOn = !digitalRead(4);
  controllerData.crossOn = !digitalRead(5);
  controllerData.l1On = !digitalRead(6);
  controllerData.l2On = !digitalRead(7);
  controllerData.l3On = !digitalRead(8);
  controllerData.r1On = !digitalRead(9);
  controllerData.r2On = !digitalRead(10);
  controllerData.r3On = !digitalRead(11);
  controllerData.selectOn = !digitalRead(12);
  controllerData.startOn = !digitalRead(A4);
  controllerData.homeOn = !digitalRead(A5);
  controllerData.leftStickX = map(analogRead(A0, 90, 370, 0, 1023)) >> 2;
  controllerData.leftStickY = map(analogRead(A1), 95, 350, 0, 1023) >> 2;
  controllerData.rightStickX = map(analogRead(A2) ,20,280, 0, 1023) >> 2;
  controllerData.rightStickY = analogRead(A3) >> 2;
  return controllerData;
}
```

```
  uint8_t leftStickX : 8;
  uint8_t leftStickY : 8;
  uint8_t rightStickX : 8;
  uint8_t rightStickY : 8;

} dataForController_t;


  void setupUnoJoy(void);
  void setupUnoJoy(int);
  void setControllerData(dataForController_t);
  dataForController_t getBlankDataForController(void);
  dataForController_t controllerDataBuffer;


void setControllerData(dataForController_t controllerData){

  ATOMIC_BLOCK(ATOMIC_FORCEON){
    controllerDataBuffer = controllerData;
  }
}
volatile int serialCheckInterval = 1;

int serialCheckCounter = 0;

void setupUnoJoy(void){

 void setupUnoJoy(void){

   controllerDataBuffer = getBlankDataForController();

   Serial.begin(38400);

   OCR0A = 128;
   TIMSK0 |= (1 << OCIE0A);
 }


 void setupUnoJoy(int interval){
   serialCheckInterval = interval;
   setupUnoJoy();
 }

 ISR(TIMER0_COMPA_vect){
   serialCheckCounter++;
   if (serialCheckCounter >= serialCheckInterval){
     serialCheckCounter = 0;

     while (Serial.available() > 0) {
       pinMode(13, OUTPUT);

       byte inByte = Serial.read();

       Serial.write(((uint8_t*)&controllerDataBuffer)[inByte]);

     dataForController_t getBlankDataForController(void){

       dataForController_t controllerData;

       controllerData.triangleOn = 0;
       controllerData.circleOn = 0;
       controllerData.squareOn = 0;
       controllerData.crossOn = 0;
       controllerData.l1On = 0;
       controllerData.l2On = 0;
       controllerData.l3On = 0;
       controllerData.r1On = 0;
       controllerData.r2On = 0;
       controllerData.r3On = 0;
       controllerData.dpadLeftOn = 0;
       controllerData.dpadUpOn = 0;
       controllerData.dpadRightOn = 0;
       controllerData.dpadDownOn = 0;
       controllerData.selectOn = 0;
       controllerData.startOn = 0;
       controllerData.homeOn = 0;

       controllerData.leftStickX = 128;
       controllerData.leftStickY = 128;
       controllerData.rightStickX = 128;
       controllerData.rightStickY = 128;

       return controllerData;
     }
```

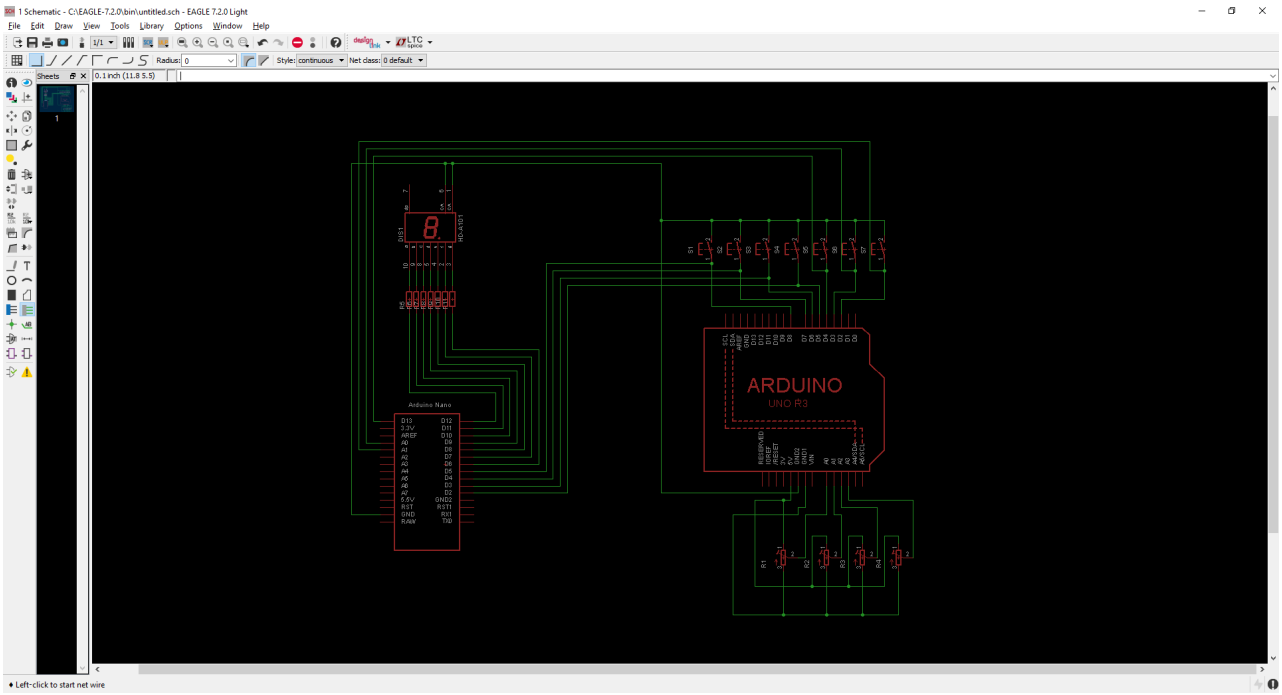
Príloha B –


+5V
GND
pin A1
pin A2
pin A3
Potentiometer

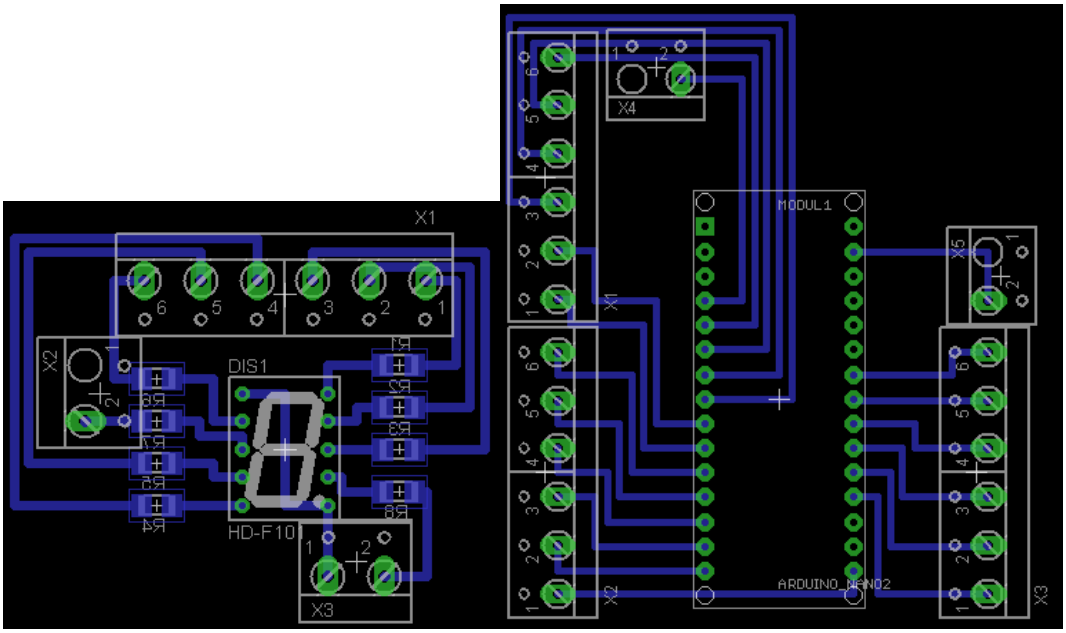

pin A0
+5V
GND


7 SEGMENT
ARDUINO UNO
D pin
D pin
D pin
D pin
D pin
D pin
D pin
GND

Príloha C



Príloha C.1



Príloha D →

Príloha E →