ELEKTROTEHNIČKA I EKONOMSKA ŠKOLA NOVA GRADIŠKA

ETEŠ HOTAS CONTROLLER

v. 2.0

Nova Gradiška, Svibanj 2021.

Sadržaj

1. Uvod

ETEŠ Hotas Controller je multifunkcionalan USB uređaj koji se koristi prilikom igranja raznih video igara. Stvoren je za igranje igara u kojima je moguće upravljati letjelicama, ali nije ograničen samo na te igre. Kontroler je vrlo jednostavan za osposobiti, koristiti pa čak i nadograditi ako imate sposobnosti koje su potrebne za to.

Kontroler pokreće Arduino MKR1000 mikrokontroler tj. on obavlja svu komunikaciju sa računalom i čita vrijednosti na komponentama.

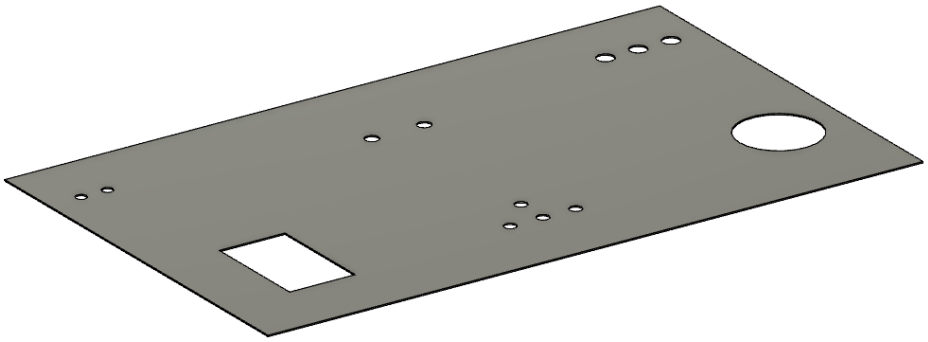
Unutar ove dokumentacija stoje sve najbitnije sheme i crteži ukoliko želite izraditi ovakav kontroler.

Slika 1.0 – „Tasteri na kontroleru“

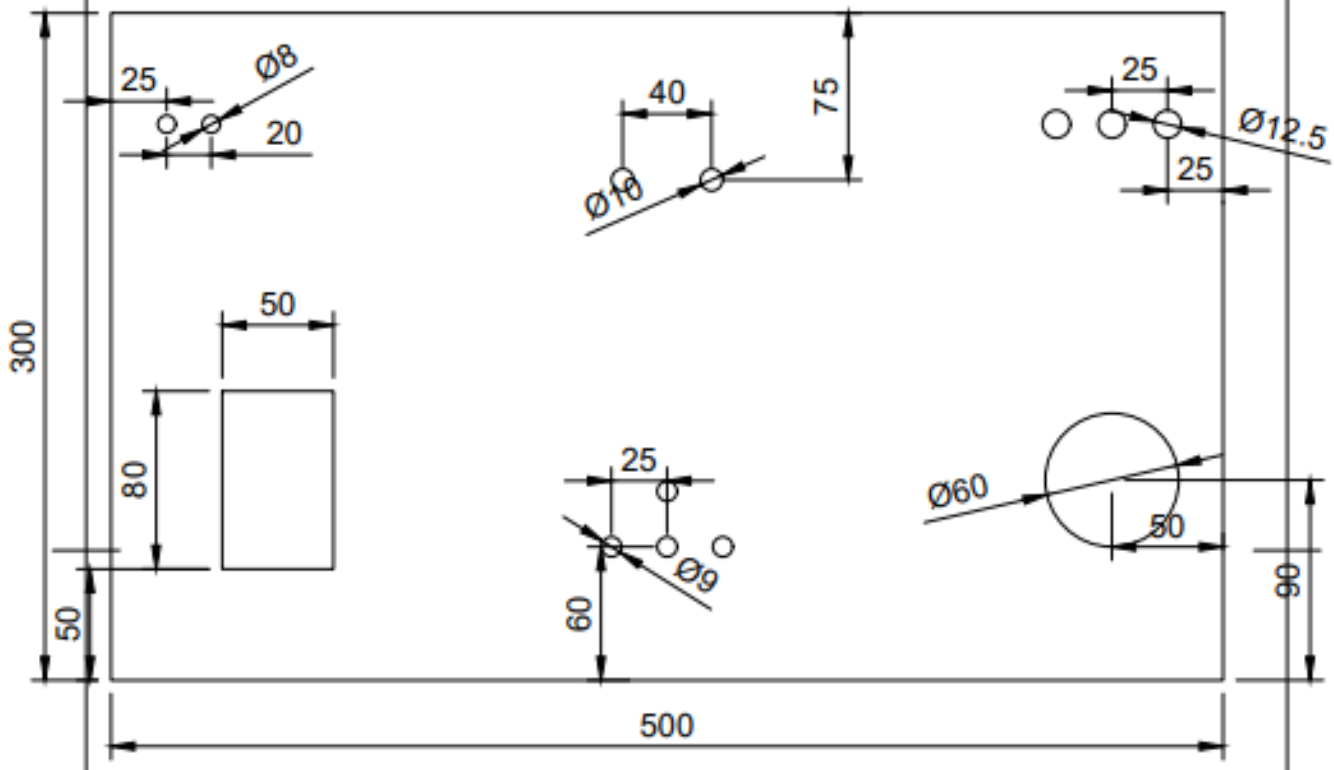
2. Mehanički dijelovi

2.1. Kućište

Kućište kontrolera izrađeno je od lima (1mm). Uzeli smo tablu lima 1m x 2m x 1mm. Prvo su izreza tri dijela od kojih je stvoreno kućište. Dva dijela potrebno je smotati i zatim ih je potrebno povariti ili napraviti mjesta na kojima će se dijelovi spojiti šarafom ili na neki drugi način. Na „gornjoj“ površini potrebno je izbušiti rupe za prekidače, tastere, tipke, joystick i throttle kao na sljedećoj slici:



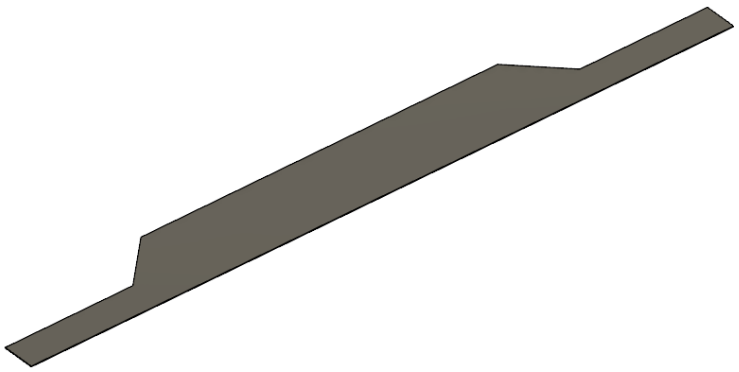
Slika 2.1 – „Gornja ploča lima s rupama“



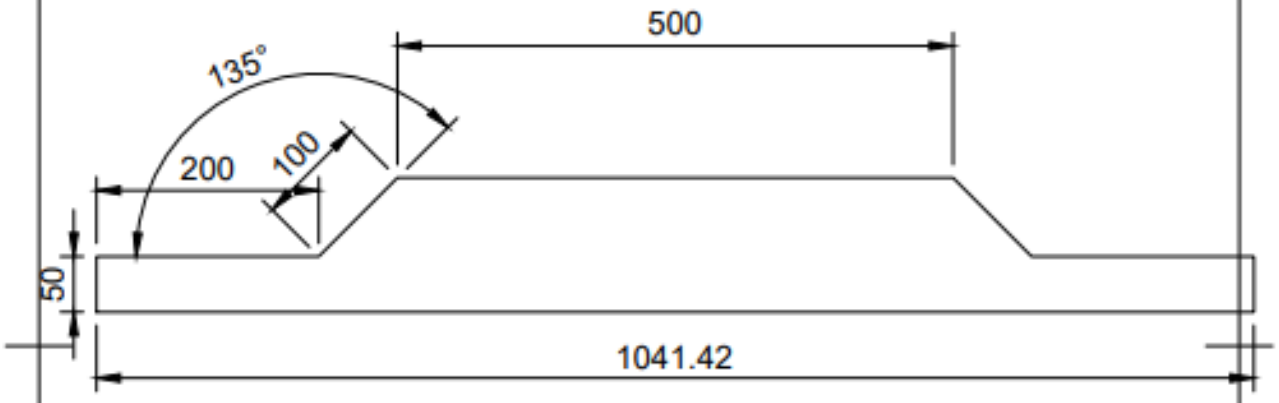
Dept.	Technical reference	Created by Ivan Došlić 30.5.2021.	Approved by
		Document type	Document status
		Title gornjaTabla	DWG No.
			Rev. / Date of issue / Sheet 1/1

Gore nacrtani dio potrebno je smotati na 10cm od gornjeg ruba pod 45° i od donjeg ruba na 5cm pod 90°.

Sljedeći dio su stranice kontrolera koje se motaju na dva mjesta pod 90° i samo se zatim spoje s gornjom pločom.



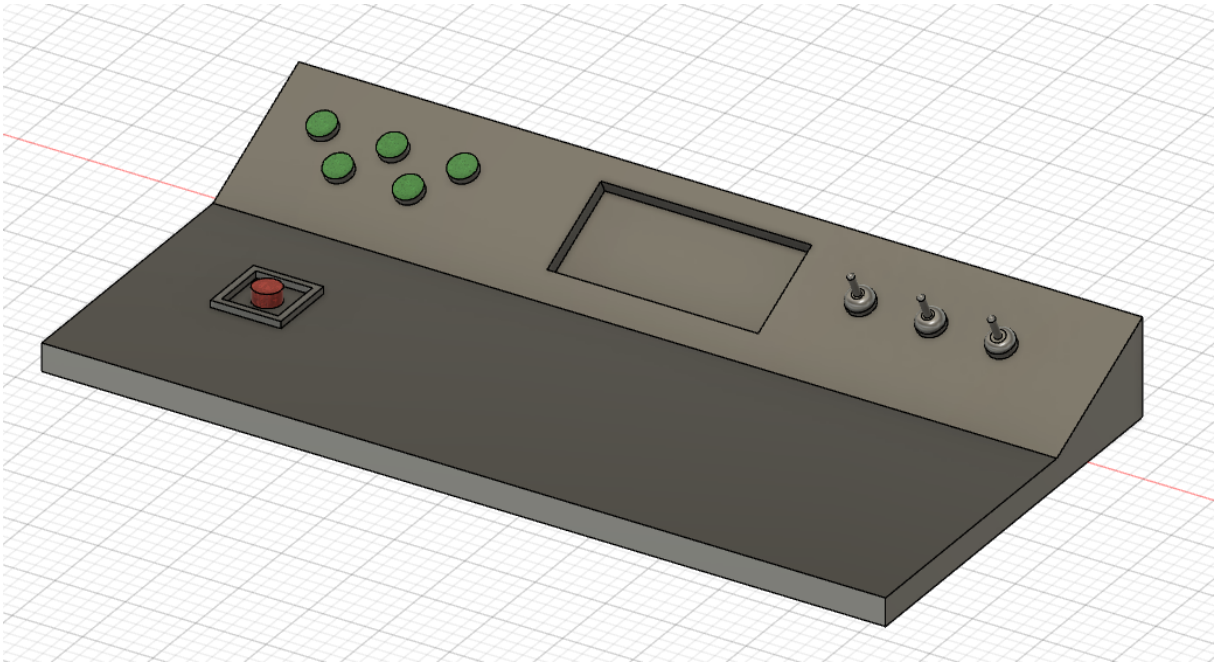
Slika 2.2 – „Ploča za stranice kontrolera“



Dept.	Technical reference	Created by Ivan Došlić 30.5.2021.	Approved by
		Document type	Document status
		Title Untitled	DWG No.
			Rev. / Date of issue / Sheet 1/1

Potrebno je smotati lim pod pravim kutom na 27.7cm od lijeva i od desna.

Posljednji dio kutije je samo ploča 50cm x 27.7cm koja ide odozdo kako bi se kutija mogla zatvoriti. Kada se to sve odradi i zavari trebali bismo dobiti ovo:



Slika 2.3 – „Prvi prototip kontrolera“ (Finalna verzija je drukčija)

2.2. „Mehanizam“ za ispravljanje joysticka

Zbog veličine joysticka nije moguće da ga joystick modul sam izravnava pa smo morali smisliti kako riješiti taj problem. Problem je riješen tako što smo s nutarnje strane kućišta zavarili 4 držača ta gumice koje onda idu oko joysticka i vraćaju ga u početnu poziciju.

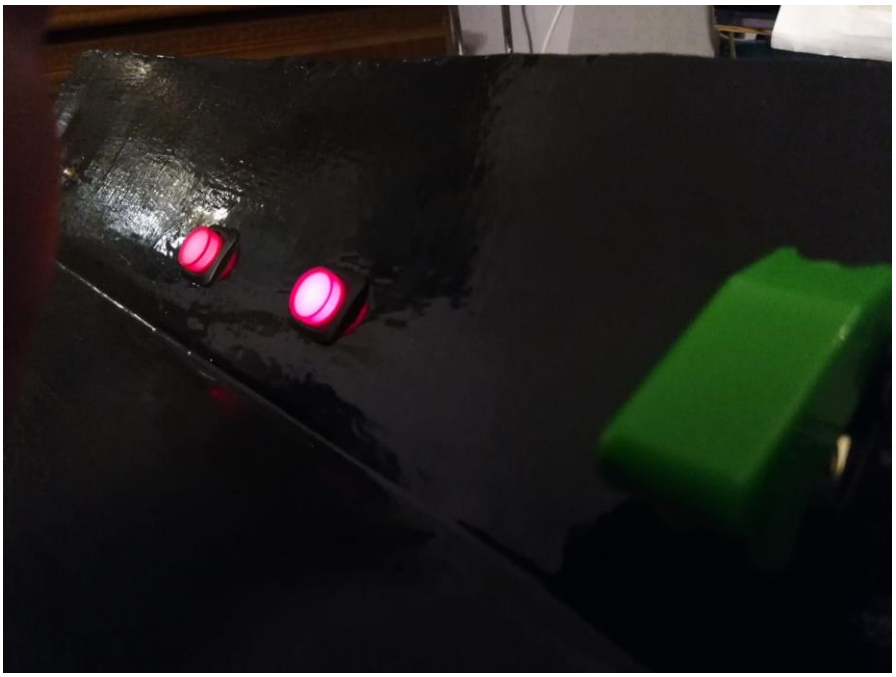
Slike 2.4 – „Gumice ispravljaju joystick“

2.3. Joystick

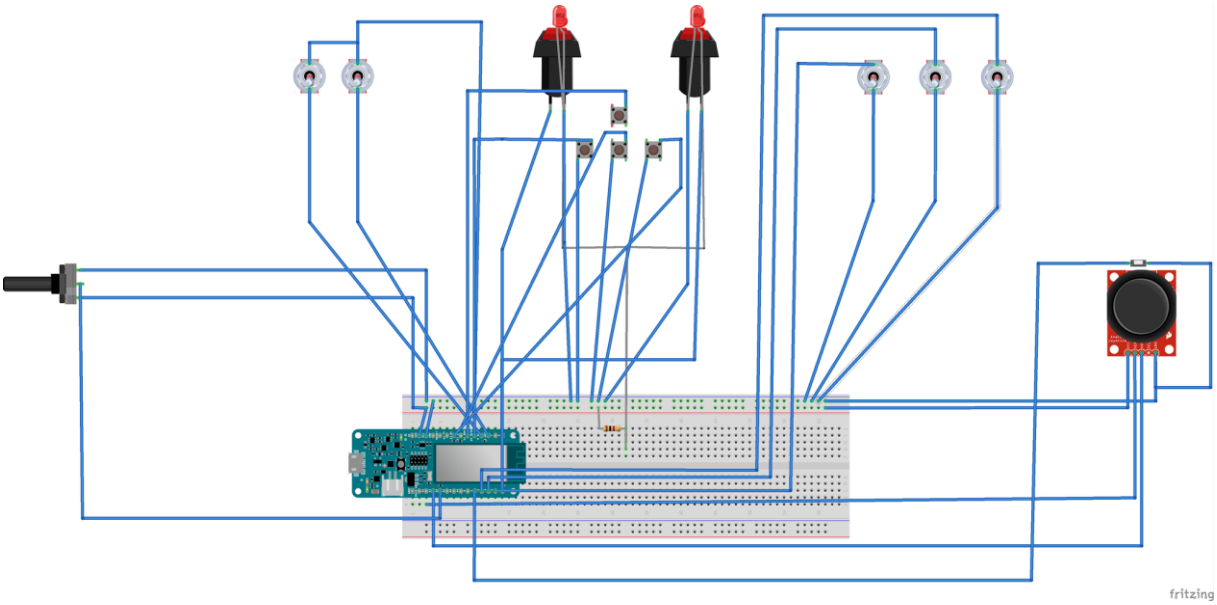
Joystickov 3D model isti je kao i prošlogodišnji samo mu je ovaj put dodan prekidač s prednje strane koji se najčešće koristi za pucanje iz nekog oružja.

3. Elektronički dijelovi

Elektroničke komponente koje koristimo u našem kontroleru mogu biti pronađene vrlo lako na chipoteka.hr. Od elektroničkih komponenti imamo 5 prekidača (3 velika i 2 mala), 2 tastera, 4 obična tipkala, GRAVITY Joystick modul i potenciometar (za throttle).



Slika 3.1 – „Komponente na kontroleru“



Slika 3.2 – „El. Shema kontrolera“

4. Program

hotas2

```
//Potrebne biblioteke
#include <Joystick.h>

//Definiranje varijabli i instanci
Joystick_ Joystick;
int XAxis_ = 0;
int YAxis_ = 0;
int Throttle = 0;

void setup() {
  //Postavljanje pinove u input mode
pinMode(8, INPUT_PULLUP); //Trigger na palici za upravljanje
pinMode(7, INPUT_PULLUP); //Sklopka 1
pinMode(6, INPUT_PULLUP); //Središnja tipka 1
pinMode(5, INPUT_PULLUP); //Središnja tipka 2
pinMode(4, INPUT_PULLUP); //Sigurnosna sklopka 1
pinMode(3, INPUT_PULLUP); //Sigurnosna sklopka 2
pinMode(2, INPUT_PULLUP); //Sigurnosna sklopka 3
pinMode(A3, INPUT);
pinMode(A1, INPUT);
pinMode(A2, INPUT);

//Započinjanje instance
Joystick.begin();
delay(500);
}

void loop() {
  //Čitanje analognih pinova za X axis,Y axis i throttle, i postavljanje na virtualni joystick.
  XAxis_ = analogRead(A2);
  XAxis_ = map(XAxis_, 720, 260, 0, 1022);
  Joystick.setXAxis(XAxis_);

  YAxis_ = analogRead(A1);
  //Postavljanje XAxisa kontrolera
  YAxis_ = map(YAxis_, 480, 720, 1022, 0);
  Joystick.setYAxis(YAxis_);

  Throttle = analogRead(A3);
```

Slika 4.1 – „Prvi dio koda“

hotas2

```
Throttle = analogRead(A3);
//Mapiranje(pretvaranje) hod senzora u vrijednosti od 0 do 1022(puni hod).
Throttle = map(Throttle, 1023, 820, 1022, 0);
//Postavljanje vrijednosti
Joystick.setThrottle(Throttle);



//Čitanje statusa tipki.

//Čitanje ulaza 8 za input
if (digitalRead(8) == LOW){

Joystick.pressButton(0);
//Pritiskanje virtualnog gumba
  } else {
    //Puštanje virtualnog gumba
    Joystick.releaseButton(0);
  }


//Čitanje ulaza 7 za input
if (digitalRead(7) == LOW){

Joystick.pressButton(1);
//Pritiskanje virtualnog gumba
  } else {
    //Puštanje virtualnog gumba
    Joystick.releaseButton(1);
  }


//Čitanje ulaza 6 za input
if (digitalRead(6) == LOW){

Joystick.pressButton(2);
//Pritiskanje virtualnog gumba
  } else {
    //Puštanje virtualnog gumba
```

Slika 4.2 – „Drugi dio koda“

hotas2

```
    //Puštanje virtualnog gumba
    Joystick.releaseButton(2);
  }


//Čitanje ulaza 5 za input
if (digitalRead(5) == LOW){

Joystick.pressButton(3);
//Pritiskanje virtualnog gumba
  } else {
    //Puštanje virtualnog gumba
    Joystick.releaseButton(3);
  }


//Čitanje ulaza 4 za input
if (digitalRead(4) == LOW){

Joystick.pressButton(4);
//Pritiskanje virtualnog gumba
  } else {
    //Puštanje virtualnog gumba
    Joystick.releaseButton(4);
  }

//Čitanje ulaza 3 za input
if (digitalRead(3) == LOW){

Joystick.pressButton(5);
//Pritiskanje virtualnog gumba
  } else {
    //Puštanje virtualnog gumba
    Joystick.releaseButton(5);
  }

  //Čitanje ulaza 2 za input
if (digitalRead(2) == LOW){
```

Slika 4.3 – „Treći dio koda“

Kod koji hotas koristi vrlo je jednostavan. Koristimo se funkcijama za čitanje stanja na određenoj komponenti i onda koristimo Joystick.h library i njegove funkcije kako bismo mogli koristiti kontroler unutar video igara. Također koristili smo funkciju map() na throttleu i joysticku zbog nedostatka hoda na njima i kako bismo si podesili „osjetljivost“ obje komponente kako želimo.

Zbog problema koji postoje u nekim od librarya (specifično pluggableusb.h) morali smo manualno proći kroz izvorni kod nekih od librarya i modificirati ih kako bi program proradio.