RAČUNARSKA GIMNAZIJA

Lazar Premović

**ČOVEK-MAŠINAU MUZINTERFEJS ZA PRICI IMENU**

Professional Hardware Interfaces - Generic Professional Controller **"PHI - GPC"**

maturski rad iz predmeta:

**Napredne tehnike programiranja**

**Mentor**:

Prof. dr Filip Marić

Beograd, maj 2019.

Naslov maturskog rada: ovek-ma†ina interfejs za primenu u muzici

Rezime: Upravljanje ra£unarom uz pomo¢ mi†a i tastature je sasvim dovojno za ve¢inu obi£nih korisnika ali za veliku ve¢inu profesionalnih primena ra£unara (pogotovo u Audio/Video oblasti) mi† i tastatura nisu optimalan na£in upravljanja ra£unarom. Zato se za ve¢inu takvih primena na ra£uare povezuju posebno dizajnirani kontroleri koji olak†avaju i ubrzavaju posao koji treba obaviti. Me utim kontroleri pored visokih cena imaju jo† nekoliko problema koji ih £ine neprakti£nim: dizajniraju se za ta£no odre enu vrstu posla (npr. te†ko je obra ivati slike na kontroleru dizajniranom za miksovanje zvuka) i £esto se dizajniraju da rade sa samo jenim programom i vrlo je te†ko ili nemogu¢e koristiti ga za upravljane nekim drugim programom.

Zato je cilj ovog rada bio izrada kontrolera koji je primenjljiv u †irokom spektru poslova i mogu¢e ga je povezati sa bilo kojim programom. Da bi se taj cilj ostvario hardver kontrolera mora biti op†t da bi mogao da se primeni na bilo koju vrstu softvera kojim treba da se upravlja, a softver za kon gurisanje mora da omogu¢i veoma detaljnu kon guraciju kontrolera i lako pro†irivanje funkcionalnosti. Tako su nastali:

* GPC "Generic Professional Controller" (Kontroler),
* PHI Control Center[[1]](#footnote-1) (Softver za kon gurisanje i obradu u realnom vremenu),
* GHP "Generic HID[[2]](#footnote-2) Protocol" (Protokol za komunikaciju izme u kontrolera i softvera).

Sadr”aj

[1 Kori†¢ene Tehnologije 1](#_Toc47936)

[1.1 Arduino 1](#_Toc47937)

[1.2 C# desktop aplikacije 2](#_Toc47938)

[1.3 JSON 2](#_Toc47939)

[1.4 MIDI 2](#_Toc47940)

[2 GHP Protokol 4](#_Toc47941)

[3 Kontroler 6](#_Toc47942)

[3.1 Hardver 7](#_Toc47943)

[3.2 Elektronika i intergirsana kola 8](#_Toc47944)

[3.3 Mikrokontrolerski Kod 10](#_Toc47945)

[4 Desktop softver 15](#_Toc47946)

[4.1 Upotreba softvera 15](#_Toc47947)

[4.2 Detaji implementacije 17](#_Toc47948)

[5 Zaklju£ci i dalji rad 22](#_Toc47949)

[Dodatni Materijal 23](#_Toc47950)

[Literatura 31](#_Toc47951)

iii

Glava 1

# Kori†¢ene Tehnologije

## 1.1 Arduino

Arduino je projekat otvorenog koda koji dizajnira i proizvodi mikrokontrolere bazirane na Atmel 8-bitnim AVR miktrokontrolerima £ije se sve neophodne komponente nalaze na jednoj †tampanoj plo£i. Arduino mikrokontroleri se programiraju uz pomo¢ Arduino IDE-a[[3]](#footnote-3) uz pomo¢ programskog jezika C++ i nekoliko dodatnih biblioteka. U kodu je umesto main() funkcije neophodno imati funkcije setup() koja se ivzr†i samo jednom £im se mikrokontroler uklju£i i loop() koja se poziva dokle god je mikrokontroler uklju£en. Arduino IDE tako e omogu¢ava prevo enje koda i njegovo u£itavanje na mikrokontroler uz pomo¢ samo jednog klika i time olak†ava i ubrzava razvoj softvera za Arduino mikrokontrolere. Digitalni pinovi mogu biti u samo dva stanja: logi£ki 1[[4]](#footnote-4) i logi£ki 0[[5]](#footnote-5) u daljem tekstu ¢emo ih zvati HIGH i LOW.

PWM Pulse Width Modulation, je tehnika kojom je mogu¢e aproksimirati bilo koji izlazni napon £ak iako pin mo”e biti samo u upaljenom ili uga†enom stanju. To se posti”e tako †to pin menja stanja vrlo brzo a pode†avanjem odnosa vremena kada je pin upaljen i kada je pin uga†en mogu¢e je ta£no precizirati tra”eni napon.

GLAVA 1. KORI ENE TEHNOLOGIJE

## 1.2 C# desktop aplikacije

C# je programski jezik op†te namene koji je razvio Microsoft za potrebe razvojnog okru”ennja .NET Framework. C# podr”ava objekno orjentisanu, imperativnu, deklarativnu i generi£ku programsku paradigmu. Deo C#-a je i paket Windows Forms koji omogu¢ava brzo i lako kreiranje desktop aplikacija sa gra£kim interfejsom uz pomo¢ Microsoft Visual Studio IDE-a.

## 1.3 JSON

JavaScript Object Notaon JSON je format za £uvanje i prenos podataka u kome su podatci zabele”eni tako da ih je vrlo lako u£itati i prevesti u odgovaraju¢i objekat. Svaki zabele”en podatak je reprezentovan kao par klju£a i vrednosti. Tipovi podataka koje JSON podr”ava su: string,int,bool,null,niz i objekat (koji mo”e sadr”ati bilo koji tip).

## 1.4 MIDI

Musical Instrument Digital Interface MIDI je standard koji opisuje komunikacioni protokol, digitalni interfejs i zi£e konektore koji omogu¢uju povezivanje raznovrsnih digitalnih audio ure aja. MIDI protokol de ni†e MIDI poruke koje su sa£injene od statusnog bajta i do dva bajta sa podatcima koji se prenose serijski brzinom od 31.25 kbit\s. Statusni bajtovi su uvek ≥128 dok su bajtovi sa podatcima <128. MIDI de ni†e nekoliko statusnih bajtova kao †to su NoteOn (nota je pritisnuta), NoteO (nota je otpu†tena) i ControlChange (kontrola se promenila) koji je nama bitan zato †to se uglavnom koristi za komunikaciju kontrolera i audio softvera. MIDI de ni†e da posle ControlChange statusnog bajta slede dva bajta: prvi bajt govori koja kontrola je u pitanju slanjem njenog indenti kacionog broja koji je iz opsega 0-127 i drugi bajt koji govori o novoj poziciji kontrole tako e iz opsega 0-127

### virtualMIDI

Kako MIDI standard de ni†e samo komunikaciju izme u dva hardverska ure aja i komunikaciju izme u hardvera i softvera, da bi smo ostvarili kounikaciju izme u dva softvera (u ovom slu£aju odabranog audio softvera i PHI Control Center-a GLAVA 1. KORI ENE TEHNOLOGIJE

koji slu”i kao posrednik izme u audio softvera i kontrolera) potrebno je da koristimo biblioteku koja nam omogu¢ava da kreiramo virtuelni MIDI ure aj i koristimo njega za slanje MIDI poruka. virutalMIDI Tobias Erichsen-a je biblioteka koja nam to omogu¢ava. Da bismo je koristili potrebno je da instaliramo loopMIDI ili rtpMIDI softver uz koji dolazi drajver za koji se virtualMIDI povezuje. Nakon toga je dovojno uklju£iti .cs fajl biblioteke u projekat i koristiti metode za kreiranje virtuelnog ure aja i slanje poruka preko njega.

Glava 2

# GHP Protokol

Da bi kotroler mogao da komunicira sa softverom za kon guraciju bilo je potrebno osmisliti i de nisati protokol koji ¢e se koristiti. GHP kao svoju osnovu koristi serijsku komunikaciju preko USB-a sa pode†enim baud rate-om[[6]](#footnote-6) od 1152002.Poruke koje se †alju ovim protokolom imaju slede¢i format:

CommandByte, ChannelByte, [0 − 20]DataBytes.

Komandni bajt govori o kojoj komandi je re£ i tako e time speci kuje broj bajtova sa podatcima koji ¢e uslediti, za spisak komandi pogledati tabelu 2.1. Radi lak†eg razumevanja sve vrednosti komandnih bajtova su iz alfanumeri£kog opsega ASCII enkodiranja tako da ih mo”emo lak†e obele”avati po ASCII karakteru koji im odgovara.

Bajt koji ozna£ava kanal govori na koju kontrolu ili memorisku adresu se odnosi komanda.

Bajtovi sa podatcima sadr”e podatke karakteristi£ne za odre enu komandu.

GLAVA 2. GHP PROTOKOL

|  |  |  |  |
| --- | --- | --- | --- |
| Komandni bajt | Komanda | Kanal | Opis bajtova sa podatcima |
| ’B’ 66 | Taster je pritisnut | Indeks tastera | NEMA |
| ’R’ 82 | Taster je otpu†ten | Indeks tastera | NEMA |
| ’A’ 65 | Analogna kontrola je pomerena | Indeks analogne kontrole | 2 bajta. Nova vrednost analogne  kontrole, Little  Endian enkodiranje |
| ’I’ 73 | Rotacioni enkoder je inkrementovan | Indeks enkodera | NEMA |
| ’D’ 68 | Rotacioni enkoder je dekrementovan | Indeks enkodera | NEMA |
| ’L’ 76 | Podesi LED | Indeks diode | 3 bajta. R, G, B |
| ’l’ 108 | Podesi LCD | Broj linije ekrana | 20 bajtova. Tekst |
| ’P’ 80 | Podesi trajni podatak | Memorijska adresa | 1 bajt. Vrednost |
| ’p’ 112 | Zatra”i vrednost trajnog podatka | Memorijska adresa | NEMA |
| ’r’ 114 | Odgovor na zatra”enu vrednost | Memorijska adresa | 1 bajt. Vrednost |
| ’S’ 83 | Sistemska komanda, videti tabelu 2.2 | Vrsta sistemske komande | Zavisi od vrste sistemske komande |

Tabela 2.1: Obja†njenje komandnih bitova

|  |  |  |
| --- | --- | --- |
| Vrsta sistemske komande | Komanda | Opis bajtova sa podatcima |
| ’W’ 87 | Zatra”i indenti kacioni broj ure aja | NEMA |
| ’w’ 119 | Odgovor na zatra”en indenti kacioni broj ure aja | 1 bajt. Identi kacioni broj ure aja |
| ’R’ 82 | Resetuj ure aj | NEMA |

Tabela 2.2: Obja†njenje sistemskih komandi

Glava 3

# Kontroler



Slika 3.1: GPC kontroler

Najprostije re£eno posao kontrolera je da akcije korisnika prenese softveru za obradu u realnom vremenu. Me utim taj proces zahteva nekoliko koraka:

1. u£itavanje stanja kontrola,
2. detekciju promena stanja kontrola,
3. slanje informacija o promeni stanja kontrole koriste¢i GHP protokol,
4. a”uriranje stanja displeja i dioda na osnovu informacija primljenih GHP protokolom.

Zato GPC ima nekoliko integalnih kola koja poma”u sa u£itavanjem stanja kontrola i Arduino Mega mikrokontoler koji vr†i sve ostale operacije i implementira GHP protokol.

Karakteristike Arduino Mega mikrokontrolera

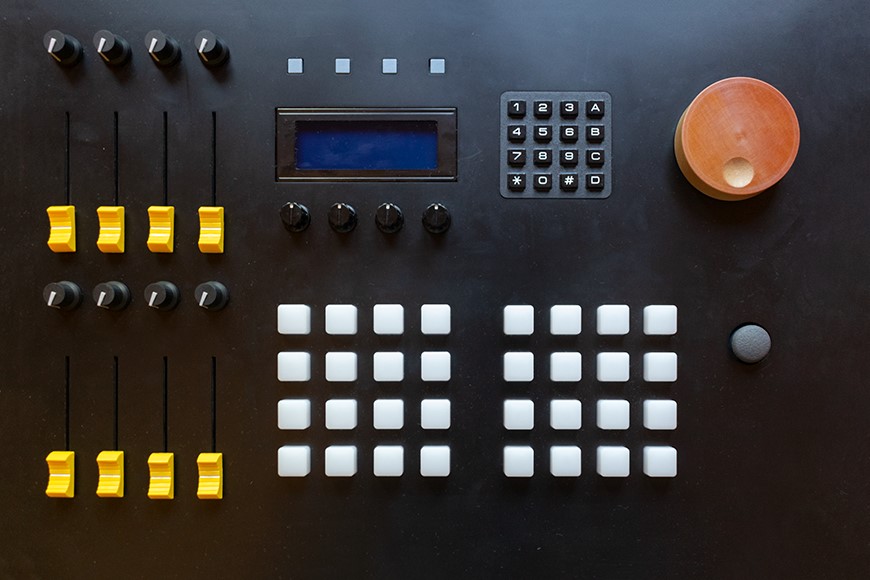
* ATmega2560 mikrokontroler
* 54 Digitalna ulazno/izlazna pina (od kojih 15 imaju podr†ku za PWM[[7]](#footnote-7))
* 16 Analognih ulaznih pinova
* 256 KB Programske memorije
* 8 KB RAM memorije
* 4 KB EEPROM[[8]](#footnote-8) stalne memorije

## 3.1 Hardver

### Kontrolna povr†ina

Kotnrolnu povr†inu £ine nekoliko disjunktnih grupa, videti sliku 3.2.

* Analogna sekcija (krajnje levo) sadr”i 8 kliznih i 8 rotacionih potenciometara koji u zavisnosti od svoje pozicije mogu imati vrednosti u opsegu 0-1023
* Sekcija sa tasterima (dve matrice sa belim tasterima i numeri£ka tastatura) sadr”i 48 tastera podeljena u tri 4x4 matrice (donja leva matrica tako e sadr”i RGB LED diode ispod svakog tastera) i †alje informaciju kada se neki od tastera pritisne ili otpusti.



Slika 3.2: Kontrolna povr†ina

* Displej i prate¢e kontole, ispod displeja se nalaze 4 beskona£na rotaciona enkodera koji †alju informaciju kada su inkremenovani[[9]](#footnote-9) ili dekremenovani[[10]](#footnote-10) a iznad se nalaze 4 tastera i svi zajedno su namenjeni da rade u kontekstu onoga †to je prikazano na displeju.
* To£ak za brzu navigaciju i d”ojstik su zasebne grupe i realizuju se uz pomo¢ rotacionog enkodera (to£ak) i kao spoj dve analogne kontrole za x i y ose i tastera koji reaguje kada se d”ojstik pritisne.

## 3.2 Elektronika i intergirsana kola

Da bi se sve gorenavedene kontrole povezale potrebno je:

16 + 48 + 3 ∗ 16[[11]](#footnote-11) + 4 + 2 ∗ 5[[12]](#footnote-12) + 6[[13]](#footnote-13) + 3 = 135

pinova, mnogo vi†e nego †to na† mikrokontroler ima, zato moramo da uz pomo¢ nekoliko tehnika smanjimo taj broj na ne†to manje od ≈70

Matrice tastera Matrice tastera nam omogu¢avaju da *N* tastera pove”emo kori-

√ ste¢i samo 2 *N* pinova tako †to tasere rasporedimo u matricu. itanje vr†imo tako †to redom po jednu kolonu postavljamo HIGH i onda £itamo vrednosti kolona. Ono

†to mo”emo da uradimo da dodano pobolj†amo ovo re†enje je da sve ulazne[[14]](#footnote-14) pinove

√ √

svih matrica spojimo i efektivno dobijemo jednu veliku *N*× 3 ∗ *N* matricu koja

√

koristi samo 4 ∗ *N* pinova za 3 ∗*N* tastera. Sli£nu stvar mo”emo da uradimo i za LED diode i dobjemo tri matrice (po jednu za svaku boju) koje opet mo”emo

√ √

da spojimo i dobijemo jo† jednu *N*× 3 ∗ *N* matricu. Sada smo broj potrebnih pinova smanjili na:

16 + 16[[15]](#footnote-15) + 16[[16]](#footnote-16) + 4 + 2 ∗ 5 + 6 + 3 = 71

†to je jo† uvek malo vi†e nego †to mo”emo da priu†timo.

Multipleksiranje Multipleksiranje nam omogu¢ava da uz pomo¢ specijalnog integrisanog kola smanjimo broj potrebnih pinova sa *N* na (log2*N*)+1. To posti”emo tako †to svaki ulaz adresiramo po binarnoj repre”entaciji njegovog indeska a broj bitova u binarnoj reprezenraciji nekog broja je log2*N* i jo† jedan pin za £itanje izabranog ulaza i tako dobijamo (log2*N*) + 1. U ovom projektu su kori†£ena dva tipa multipleksera: Multiplekser sa 8 ulaza (koristi samo 4 pina) i multiplekser sa 4 ulaza (koristi samo 3 pina). Slede¢a optimizacija koju mo”emo da i”vr†imo jeste da pinove za selekciju svih multipleksera istog tipa pove”emo zajedno i time dobijamo *N*∗*M*[[17]](#footnote-17)ulaza sa samo (log2*N*) + *M* pinova. U kontroleru se nalaze 4 multipleksera sa 8 ulaza i 3 multipleksera sa 4 ulaza. Dva multipleksera sa 8 ulaza se koriste za ulaze iz analogne sekcije dok druga dva pokrivaju izlaze iz matrice tastera i tastere iznad displeja. Svaki od multipleksera sa 4 kanala je zadu”en za ulaze po jedne matrice LED dioda (po jedna matrica za svaku boju).Trenutni broj potrebnih pinova je:

3[[18]](#footnote-18) + 4[[19]](#footnote-19) + 2[[20]](#footnote-20) + 3[[21]](#footnote-21) + 4[[22]](#footnote-22) + 4[[23]](#footnote-23) + 2 ∗ 5 + 6 + 3 = 39

†to je dovojno mali broj da se pove”e na na† mikrokontroler.

Za †emu elektronike pogledajte sliku 1.

A detaljnije †eme i tehni£ki crte”i se nalaze na prilo”enom CD-u.

## 3.3 Mikrokontrolerski Kod

Kod koji se izvr†ava na mikrokontroleru se sastoji od glavnog fajla koji sadr”i funkcije setup() i loop() i jo† nekoliko biblioteka koje sam napisao[[24]](#footnote-24), a to su:

* de ne biblioteka koja sadr”i sve konstante (npr. brojeve pinova) i globalne promenljive koje su potrebne,
* lib biblioteka koja sadr”i neke funkcije koje mogu biti korisne i u drugim projektima (upravljanje matricama tastera i multiplekserima i opreacije nad nizovima pinova),
* GHP biblioteka koja implementira GHP protokol za komunikaciju sa ra£unarom,
* Peristence biblioteka koja olak†ava trajno sklad†tenje i £itanje podataka.

### itanje tastera

Ukoliko su tasteri lo†eg kvaliteta ili se ne pritisnu u potpunosti mo”e do¢i do odskakivanja kontakata unutar tastera †to se mo”e registovati kao vi†e pritisaka umesto samo jednog, zato je svim tasterima kontrolera implementrian Debouncing[[25]](#footnote-25)algoritam.

byte val=digitalRead(MUXDR[0]); if(val==HIGH && ButtonDebounce[ind]<DEBOUNCE)

{

ButtonDebounce[ind]++; if(ButtonDebounce[ind]==DEBOUNCE && ButtonBool[ind]==false)

{

SendMessage(BUTTON\_PRESS,ButtonAlias[ind]);

ButtonBool[ind]=true;

}

}

else if(val==LOW && ButtonDebounce[ind]>0)

{

ButtonDebounce[ind]--; if(ButtonDebounce[ind]==0 && ButtonBool[ind]==true)

{

SendMessage(BUTTON\_RELEASE,ButtonAlias[ind]);

ButtonBool[ind]=false;

}

}

Algoritam radi tako †to se za svako dugme pamti broj koji se inkrementuje ako je dugme pritisnuto a dekrementuje ako nije i proverava se da li je ta vrednost jednaka nuli (†to zna£i da je dugme ve¢ neko vreme sigurno otpu†teno) ili je ta vrednost jednaka DEBOUNCE konstanti (†to zna£i da je dugme ve¢ neko vreme pritisnuto) i ako se novo stanje dugmeta razlikuje od poslednjeg stanja koje smo poslali ra£unaru †aljemo mu novo stanje. DEBOUNCE konstanta je podesiva sistemom trajnih podataka i govori programu koliko dugo je potrebno da dugme bude pritisnuto ili otpu†teno pre nego †to se kao takvo i registruje.

### itanje analognih kontrola

Analogne kontrole su tako e podlo”ne uno†enju †uma ali za njih je potrebno koristiti druga£ije algoritme, nekada je £ak potebno da neke kontrole imau druga£ije algorime za otklanjanje †uma od ostalih. GPC na ve¢ini kontrola implementira kombinaciju dva algoritma.

short val = analogRead(MUXAR[0]); val = constrain(map(val,0,1005,0,1023),0,1023); short avg = (val+AnalogAvg[i])/2; AnalogAvg[i]=val; if(abs(avg-AnalogPre[i])>TRESHOLD)

{

SendMessage(ANALOG\_CHANGE,AnalogAlias[i],avg,true); AnalogPre[i]=avg;

}

Prvo se pre bilo kakvog ltiranja se vr†i mapiranje vrednosti da bi se osiguralo da je mogu¢e dosti¢i maksimalnu vrednost koja je 1023, potom se kao prvi korak u otklanjanju †uma koristi prose£na vrednost poslednja dva £itanja a informacija se †alje ra£unaru samo ako se razlikuje za vi†e od konstante TRESHOLD od poslednje poslate vrednosti i time se dodatno smanjuje †um i protok nepotrebnih informacija. Specijalan slu£aj je analogna pedala na opcionom dodatku za ovaj kontroler, zato †to pedala ima izuzetno mali hod (rezolucije samo 50 jedinica) gore navedeni algoritam nije dobro re†nje zbok gubitka rezolucije koji se de†ava u njemu. Stoga je na pedali implementiran Eksponencijalni rekurzivni lter koji se implementira preko formule:

*Ri* = *V* ∗*E* + *Ri*−1 ∗ (1 −*E*)

gde je *R* izlaz ltera, *V* pro£itana vrednost kontrole i *E* konstanta ltera u opsegu

0-1.

### Rotacioni enkoderi

Rotacioni enkoderi rade na principu dva signala koji su 90◦ van faze tako da je mogu¢e detektovati pomeraj enkodera tra”eci rastu¢u[[26]](#footnote-26) ili opadaju¢u[[27]](#footnote-27) ivicu na jednom signalu a onda odrediti smer rotacije na osnovu drugog signala. Po†to je zbok preciznosti neophodno detektovati svaku rastu¢u ivicu ru£no skeniranje u svakoj iteraciji loop() funkcije nije dobro re†enje jer mo”e presko£iti neku ivicu ukoliko se enkoder vrti brzo i procesor je optere¢en. Zato se za detekciju rastu¢ih ivica koriste sistemski prekidi £iju upotrebu je Arduino pojednostavio. Za po£etak potrebno je omogu¢iti sistemske prekide na nekom pinu u i proslediti mu funkciju koja se izvr†ava kada se prekid desi, a to se posi”e sledeom linijom koda:

attachInterrupt(digitalPinToInterrupt(ENC[0][0]), ENC0ISR, RISING); gde je ENC0ISR funkcija koju ¢e prekid pozvati.

void ENC0ISR()

{ if(digitalRead(ENC[0][0])==HIGH)

{ if(digitalRead(ENC[0][1])==LOW)

{

ENCPre[0]++;

} else

{

ENCPre[0]--;

}

}

}

Sistemski prekidi su veoma osetljivi na odskakivanje kontakata[[28]](#footnote-28) a enkoderi umeju da budu veoma polo”ni odskakivanju i zato je neophodno da imamo dosta ltriranja kada radimo sa njima. Prvi korak je da u samoj funkciji proverimo da li pin zaista HIGH i ako jeste onda a”uriramo relativnu po”iciju enkodera u odnosu na pro†lu iteraciju.

void Enc()

{ for(byte i=0;i<5;i++)

{

ENCPre[i]=constrain(ENCPre[i],-((short)ENCMS),(short)ENCMS); while(ENCPre[i]!=0)

{ if(ENCPre[i]>0)

{ if(ENClast[i]==true)

{

SendMessage(ENCODER\_INCREMENT,i);

}

ENCPre[i]--;

ENClast[i]=true;

} else{ if(ENClast[i]==false)

{

SendMessage(ENCODER\_DECREMENT,i);

}

ENCPre[i]++;

ENClast[i]=false;

}

}

}

}

Funkcija Enc() se izvr†ava svake iteracije loop() funkcije i †alje najnovije podatke u vezi sa enkoderima ra£unaru. Funkcija prolazi kroz svaki enkoder i prvo ograni£i broj pomeraja na konstantu ENCMS koja ozna£ava maksimali broj koraka enkodera koji ¢e se registrovati za vreme jedne iteracije. Nakon toga procesiramo promenu relativne pozicije i ukoliko je ista kao i prethodna obave†tavamo kompijuter o njoj, ovo je jo† jedan vid ltriranja koje spre£ava da se u nizu operacija inkrementovanja jedna pogre†no protuma£i kao dekrementovanje i obratno.

Glava 4

# Desktop softver

Desktop softver je drugi bitan deo GHP ekositema i slu”i da omogu¢i kon guraciju kontrolera i da u realnom vremenu prevodi komande sa kontrolera u ne†to †to ciljani softver razume

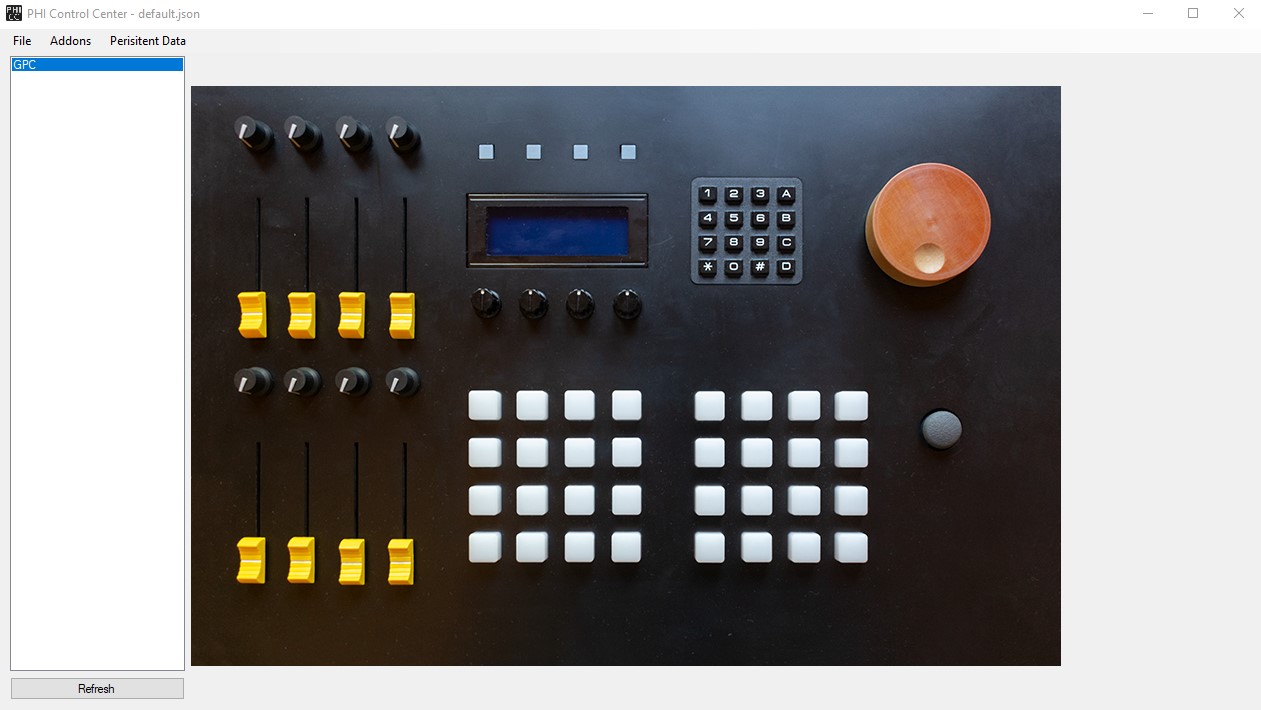
## 4.1 Upotreba softvera

Kada se softver pokrene pojavi¢e se ikonica u donjem desnom uglu trake se zadatcima, klikom na ikonicu pojavi¢e se meni koji omogu¢ava lak pristup glavnim segmentima softvera.

### Segment za kon gurisanje

Na po£etnom ekranu segmenta za kon gursanje se nalaze:

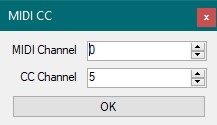
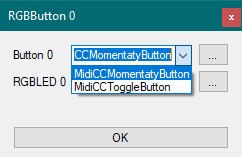
* Lista svih trenutno povezanih ure aja,
* Dugme koje osve”ava listu svih povezanih ure aja,
* Vizuelna reprezentacija trenutno izabranog ure aja na kojoj se i vr†i sama kon guracija ure aja,
* File meni koji omogu¢ava u£itavanje £uvanje i biranje aktivnog pro la,
* Meni za aktiviranje i deaktiviranje opcionih dodataka,
* Meni za pregled i izmenu trajnih podataka na kontroleru.



Slika 4.1: Po£etni ekran segmenta za kon gurisanje

Kon gurisanje odre ene kontrole se posti”e klikom na nju na vizuelnoj reprezentaciji kontrolera, tada ¢e se pojaviti dijalog u kome je mogu¢e izabrati funkiju koju ”elimo da ta kontrola obavlja a klikom na tatster ... mogu¢e je podesiti dodatne parametre funkcije[[29]](#footnote-29).

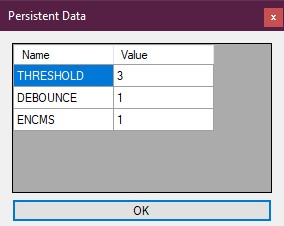
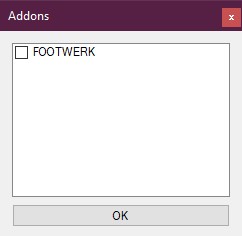
Dijalog za upravljanje dopunskim kontrolama sadr”i listu svih dopunskih kontro-



Slika 4.3: Pode†avanje dodatnih paraSlika 4.2: Izbor funkcije za kontolu metara

lera koji se mogu povezati na izabrani kontroler i svaki od njih se mo”e pojedina£no aktivirati ili deaktivirati klikom na kravrati¢ pored njegovog imena i £uvanjem promena pritiskom na taster OK.

Pode†avanje trajnih podataka se vr†i u svom dijalogu gde se za trenutno selektovani kontroler izlistavaju svi trajni podatci sa njihovim trenutnim vrednostima. Vrednosti je mogu¢e promeniti direknim unosom u polje u kome se nalazi vrednost tog podatka, pritiskom na taster OK ¢e se sve promene vrednosti a”urirati na kontroleru.



Slika 4.5: Pode†avanje trajnih poda-

Slika 4.4: Upravljanje opcionim do- taka datcima

### Segment za obradu u realnom vremenu

Segment za obradu u realnom vremenu slu”i da informacije o stanju kontrola koje primi od kotrolera prevede u komande koje je korisnik de nisao u trenutno aktivnom pro lu[[30]](#footnote-30). Na korisniku je samo da startuje nit koja vr†i obradu u realnom vremenu klikom na dugme Start RTP. Bitno je napomenuti da po†to i segment za kon guraciju i segment za obradu u realnom vremenu komuniciraju sa povezanim ure ajima nije mogu¢e da oba budu aktivna istovremeno pa je potrebno isklju£iti segment za obradu u realnom vremenu pri promeni kon guracije ure aja.

## 4.2 Detaji implementacije

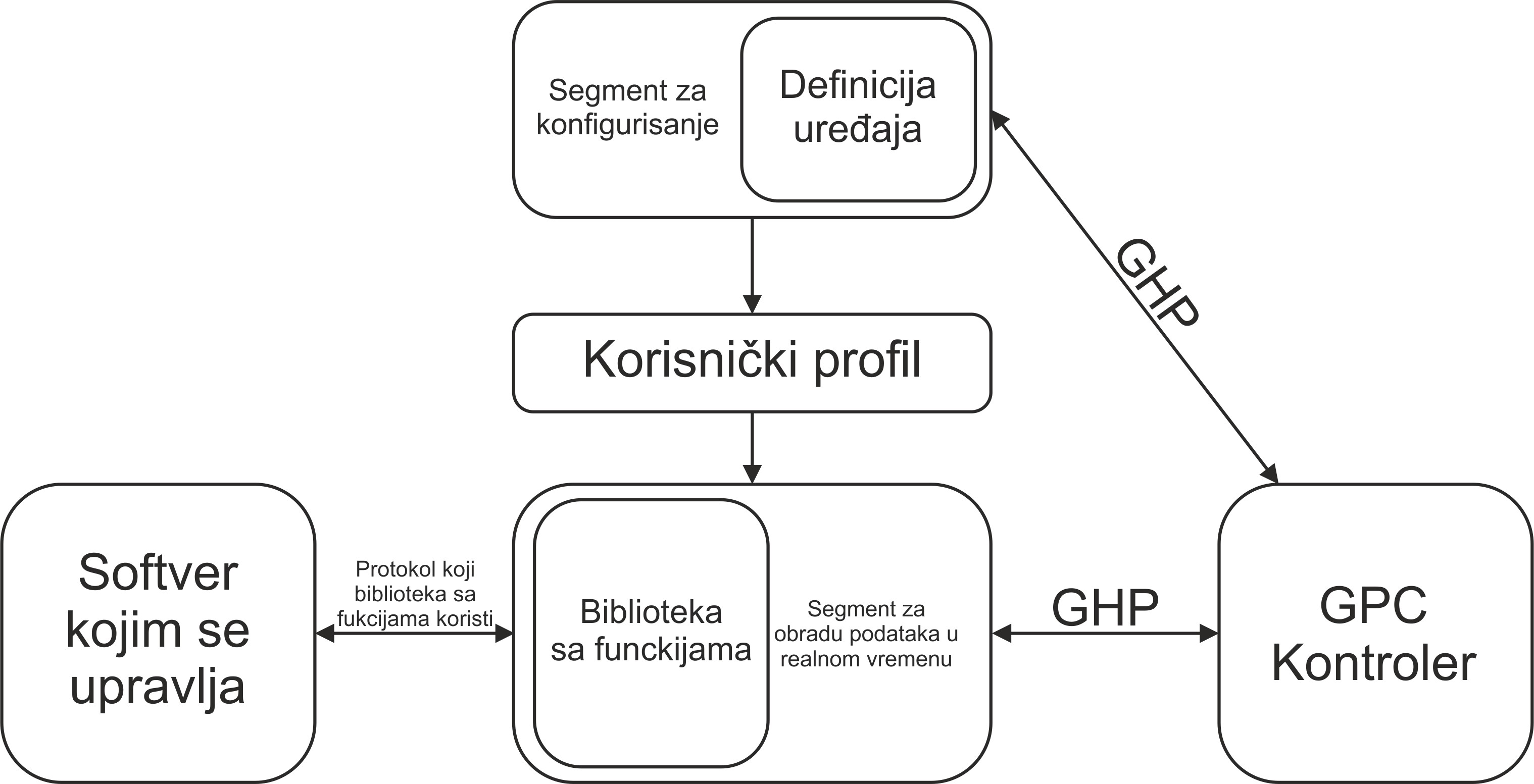
### Osnovni protok informacija i dodatne komponente

Osnovni diagram protoka informacija izgleda ovako. Ali radi lak†eg odr”avanja softvera i pro†irivanja njegovih mogu¢nosti bilo je potrebno podeliti ga na manje



Slika 4.6: Osnovni diagram protoka informacija

zasebne delove, Tako dobijamo ne†to detaljniji diagram protoka informacija, vidi sliku 4.7 Novi elementi koje smo uveli su:



Slika 4.7: Detaljniji diagram protoka informacija

• De nicija ure aja je .json fajl koji uz sliku samog kontrolera opisuje kontroler koji se mo”e povezati. Za svaki ure aj koji je mogu¢e povezati potrebno je da u folderu Devices postoji njegova de nicija ure aja i slika samog kontrolera koja se koristi za njegovu vizelnu reprezentaciju. U samoj de niciji ure aja se pamte: ime ure aja, njegov indenti kacioni broj, imena i adrese svih trajnih podataka na ure aju, imena i adrese svih opcionih dodataka i podatci o svakoj kontroli na ure aju. Za svaku kontrolu se bele”i njen tip, index, pozicija na vizuelnoj reprezentaciji (kako bi bilo mogu¢e kliknuti na nju) i jo† neki parametri vezani za nju[[31]](#footnote-31), sve kontrole su tako e organizovane

u logi£ke celine. Primer jedne de nicije ure aja mo”ete videtu u dodatnim materijalima 1.

* Korisni£ki pro l je .json fajl koji opisuje mapiranje svake kontrole na neku odre enu nkciju neke biblioteke sa funkcijama i njene dodatne parametre. Kontrole se grupi†u po svom tipu pa se utunutar toga adresiraju po svom indeksu i za svaku kontrolu je zabele”eno kojoj grupi pripada. Jedno mapiranje se sastoji iz imena biblioteke kojoj funkcija pripada, imena same funkcje i niza dodatnih parametara.
* Biblioteka sa funkcijama je .dll fajl i mo”emo ih imati koliko god ”elimo u Plugins folderu, sve one ¢e biti u£itane pri pokretanju softvera i bi¢e ih mogu¢e koristiti u kon guracijama. Svaka biblioteka mora da implementira interfejs IPlugin koji izgleda ovako:

public interface IPlugin

{

Dictionary<string, FunctionBox> GetFunctions(); bool Selftest(); string GetPluginName(); void Init(); void Close();

}

public class FunctionBox

{

public delegate void Function(byte chan, int val,List<object> par); public Function function; public CType functionType; public IDialogForm dialogForm; public FunctionBox(Function f, CType fType, IDialogForm df)

{

function = f; functionType = fType; dialogForm = df;

}

public List<object> aditionalArgs(List <object> start)

{ return dialogForm.go(start);

}

}

public enum CType

{

Analog = 0,

Button = 1,

RGBButton = 2, Encoder = 3,

LCD = 4,

JoyStick = 5,

RGBLED = 6

}

public interface IDialogForm

{

List<object> go(List<object> start);

}

### Detalji implementacije GHP protokola

Implementacija GHP protokola se nalazi u klasi GHP £ija jedna instanca predstavlja komunikaciju sa jednim GHP ure ajem[[32]](#footnote-32) na jednom serijskom portu, tako da ¢e Nit za obradu u realnom vremenu imati po jednu instancu ove klase za svaki povezani kontroler. GHP klasa tako e ima i stati£ke metode koji se koriste za detekciju i inicijalno povezivanje sa ure ajima kao i stati£ke metode koje segment za kon guraciju koristi jer segment za kon guraciju za razliku od segmenta za obradu u realnom vremenu ne odr”ava stalnu konekciju sa ure ajima. Sve stati£ke funkcije ili koriste ili se u nekoj meri baziraju na DeviceConnectedToPort funkciji koja za zadati serijski port vra¢a indenti kacioni broj ure aja koji je na njega povezan ili 0 ukoliko ni jedan ure aj nije povezan na taj port. Njena implementacija izgleda ovako:

public static byte DeviceConnectedToPort(string com)

{

SerialPort tmp = new SerialPort(); tmp.BaudRate = 115200; tmp.PortName = com; tmp.ReadTimeout = 500; tmp.Open(); tmp.DiscardInBuffer(); tmp.Write(new byte[] { SYSTEM, SYSTEM\_DISCOVER }, 0, 2); try

{

int Bc = 0; while (tmp.ReadByte() != RESPONCE\_DISOVER && Bc<75) { Bc++; } if (Bc == 75)

{ tmp.Close(); return 0;

}

byte res = (byte)tmp.ReadByte(); tmp.Close(); return res;

} catch

{ tmp.Close(); return 0;

}

}

Funkcija prvo otvori port i po†alje poruku za indenti kaciju ure aja i nakon toga £ita bajtove sa porta dok ne pro£ita sve bajtove (i time istekne timeout operacije £itanja) ili pro£ita 75 bajtova i nijedan od njih nije odgovor na zahtev za indenti kaciju (u oba slu£aja funkcja vra¢a nulu †to zna£i da ure aj nije prona en) ili ukoliko u tih 75 bajtova prona e odgovor vra¢a indenti kaciju ure aja. Ostatak GHP klase nije preterano ra”li£i od GHP implementacije na samom kontroleru.

### Detalji implementacije niti za obradu podataka u realnom vremenu

Nit za obradu podataka sama po sebi nije preterano komplikovana. Pri starovanju u£ita¢e se aktivni pro l (default.json), u£ita¢e se sve biblioteke sa funkicjama i proveri¢e se za svaki serijski port da li na njemu postoji ure aj i ukoliko postoji u listu ¢e se dodati nova instanca GHP klase vezana za taj port. Dok se niti izvr†ava u svakoj iteraciji se za svaku instancu GHP klase poziva funkcija koja obra uje sve poruke koje su stigle u me uvremenu.

Glava 5

# Zaklju£ci i dalji rad

Koriste¢i softverska i hardverska re†nja prikazana u ovom radu mogu¢e je vrlo lako koristiti jedan kontroler za †iroki spekar primena i upravljanje najrazli£itijim aplikacijama. Tako e su prikaza i re†enja za dizajniranje softvera i hardvera kompatibilnog sa GHP ekosistemom. Kontroleri koji koriste GPH protokol ne moraju da budu dizajnirani sa odre enim softverom u vidu. A softveri mogu da implemetiraju svoju biblioteku funkcija i da odmah omogu¢e korisnicima da korite veliki broj razli£itih kontrolera da upravljaju njime.

Softver je struktuiran tako da naknadno pro†irivanje funkcionalnosti bude vrlo lako zahvaljuju¢i bibliotekama sa funkcijama i de nicijama ure aja, a”uriranje samog softvera ne bi trebalo da bude neophodno osim malih promena koje bi omogu¢ile neku dodatnu funkcionalnost bibliotekama sa funkcijama. Potenijalna unapre enja sofvera bi mogla biti:

* Pod†ka za enkodere, LED diode i Lcd displej u biblioteci MIDI funkcija,
* Biblioteka koja omogu¢ava da kontrole emuliraju mi† i tastaturu,
* Biblioteka koja omogu¢ava integraciju sa AutoHotkey skriptama,
* Podr†ka za modi kacione tastere i stranice kontrola (ovo bi zahtevalo a”uriranje celog softvera).

# Dodatni Materijal

Primer koda 1: Primer de nicije ure aja

{

"Name": "GPC",

"DeviceId": 53, "IsAddon": false, "Banks":

[

{

"Name": "AnalogBank", "Controls":

[

{"Type": 0, "Id": 0,"X": 45,"Y": 351,"W": 35,"H": 150},

{"Type": 0, "Id": 1,"X": 95,"Y": 351,"W": 35,"H": 150},

{"Type": 0, "Id": 2,"X": 145,"Y": 351,"W": 35,"H": 150},

{"Type": 0, "Id": 3,"X": 195,"Y": 351,"W": 35,"H": 150},

{"Type": 0, "Id": 4,"X": 36,"Y": 276,"W": 50,"H": 45},

{"Type": 0, "Id": 5,"X": 89,"Y": 276,"W": 50,"H": 45},

{"Type": 0, "Id": 6,"X": 142,"Y": 276,"W": 50,"H": 45},

{"Type": 0, "Id": 7,"X": 195,"Y": 276,"W": 50,"H": 45}, {"Type": 0, "Id": 8,"X": 45,"Y": 107,"W": 35,"H": 150},

{"Type": 0, "Id": 9,"X": 95,"Y": 107,"W": 35,"H": 150},

{"Type": 0, "Id": 10,"X": 145,"Y": 107,"W": 35,"H": 150},

{"Type": 0, "Id": 11,"X": 195,"Y": 107,"W": 35,"H": 150},

{"Type": 0, "Id": 12,"X": 36,"Y": 25,"W": 50,"H": 45},

{"Type": 0, "Id": 13,"X": 89,"Y": 25,"W": 50,"H": 45},

{"Type": 0, "Id": 14,"X": 142,"Y": 25,"W": 50,"H": 45},

{"Type": 0, "Id": 15,"X": 195,"Y": 25,"W": 50,"H": 45}

]

},

{

"Name": "ButtonBank", "Controls":

[

{"Type": 2, "Id": 0,"IdL": 0,"X": 278,"Y": 304,"W": 32,"H": 32},

{"Type": 2, "Id": 1,"IdL": 1,"X": 325,"Y": 304,"W": 32,"H": 32},

{"Type": 2, "Id": 2,"IdL": 2,"X": 372,"Y": 304,"W": 32,"H": 32},

{"Type": 2, "Id": 3,"IdL": 3,"X": 419,"Y": 304,"W": 32,"H": 32},

{"Type": 2, "Id": 4,"IdL": 4,"X": 278,"Y": 351,"W": 32,"H": 32}, {"Type": 2, "Id": 5,"IdL": 5,"X": 325,"Y": 351,"W": 32,"H": 32},

{"Type": 2, "Id": 6,"IdL": 6,"X": 375,"Y": 351,"W": 32,"H": 32},

{"Type": 2, "Id": 7,"IdL": 7,"X": 419,"Y": 351,"W": 32,"H": 32},

{"Type": 2, "Id": 8,"IdL": 8,"X": 278,"Y": 398,"W": 32,"H": 32},

{"Type": 2, "Id": 9,"IdL": 9,"X": 325,"Y": 398,"W": 32,"H": 32},

{"Type": 2, "Id": 10,"IdL": 10,"X": 372,"Y": 398,"W": 32,"H": 32},

{"Type": 2, "Id": 11,"IdL": 11,"X": 419,"Y": 398,"W": 32,"H": 32},

{"Type": 2, "Id": 12,"IdL": 12,"X": 278,"Y": 445,"W": 32,"H": 32},

{"Type": 2, "Id": 13,"IdL": 13,"X": 325,"Y": 445,"W": 32,"H": 32},

{"Type": 2, "Id": 14,"IdL": 14,"X": 372,"Y": 445,"W": 32,"H": 32},

{"Type": 2, "Id": 15,"IdL": 15,"X": 419,"Y": 445,"W": 32,"H": 32},

{"Type": 1, "Id": 16,"X": 502,"Y": 304,"W": 32,"H": 32},

{"Type": 1, "Id": 17,"X": 549,"Y": 304,"W": 32,"H": 32},

{"Type": 1, "Id": 18,"X": 596,"Y": 304,"W": 32,"H": 32},

{"Type": 1, "Id": 19,"X": 643,"Y": 304,"W": 32,"H": 32},

{"Type": 1, "Id": 20,"X": 502,"Y": 351,"W": 32,"H": 32},

{"Type": 1, "Id": 21,"X": 549,"Y": 351,"W": 32,"H": 32},

{"Type": 1, "Id": 22,"X": 596,"Y": 351,"W": 32,"H": 32},

{"Type": 1, "Id": 23,"X": 643,"Y": 351,"W": 32,"H": 32},

{"Type": 1, "Id": 24,"X": 502,"Y": 398,"W": 32,"H": 32},

{"Type": 1, "Id": 25,"X": 549,"Y": 398,"W": 32,"H": 32},

{"Type": 1, "Id": 26,"X": 596,"Y": 398,"W": 32,"H": 32},

{"Type": 1, "Id": 27,"X": 643,"Y": 398,"W": 32,"H": 32},

{"Type": 1, "Id": 28,"X": 502,"Y": 445,"W": 32,"H": 32},

{"Type": 1, "Id": 29,"X": 549,"Y": 445,"W": 32,"H": 32},

{"Type": 1, "Id": 30,"X": 596,"Y": 445,"W": 32,"H": 32},

{"Type": 1, "Id": 31,"X": 643,"Y": 445,"W": 32,"H": 32},

{"Type": 1, "Id": 32,"X": 507,"Y": 99,"W": 20,"H": 20},

{"Type": 1, "Id": 33,"X": 533,"Y": 99,"W": 20,"H": 20},

{"Type": 1, "Id": 34,"X": 559,"Y": 99,"W": 20,"H": 20},

{"Type": 1, "Id": 35,"X": 585,"Y": 99,"W": 20,"H": 20},

{"Type": 1, "Id": 36,"X": 507,"Y": 124,"W": 20,"H": 20},

{"Type": 1, "Id": 37,"X": 533,"Y": 124,"W": 20,"H": 20},

{"Type": 1, "Id": 38,"X": 559,"Y": 124,"W": 20,"H": 20},

{"Type": 1, "Id": 39,"X": 585,"Y": 124,"W": 20,"H": 20},

{"Type": 1, "Id": 40,"X": 507,"Y": 149,"W": 20,"H": 20},

{"Type": 1, "Id": 41,"X": 533,"Y": 149,"W": 20,"H": 20},

{"Type": 1, "Id": 42,"X": 559,"Y": 149,"W": 20,"H": 20},

{"Type": 1, "Id": 43,"X": 585,"Y": 149,"W": 20,"H": 20},

{"Type": 1, "Id": 44,"X": 507,"Y": 174,"W": 20,"H": 20},

{"Type": 1, "Id": 45,"X": 533,"Y": 174,"W": 20,"H": 20},

{"Type": 1, "Id": 46,"X": 559,"Y": 174,"W": 20,"H": 20},

{"Type": 1, "Id": 47,"X": 585,"Y": 174,"W": 20,"H": 20}

]

},

{

"Name": "ScreenBank", "Controls":

[

{"Type": 4, "Id": 0,"Lin":4,"Char":20,"X": 296,"Y": 120,"W": 140,"H":

50},

{"Type": 1, "Id": 48,"X": 286,"Y": 58,"W": 16,"H": 16},

{"Type": 1, "Id": 49,"X": 334,"Y": 58,"W": 16,"H": 16},

{"Type": 1, "Id": 50,"X": 382,"Y": 58,"W": 16,"H": 16},

{"Type": 1, "Id": 51,"X": 430,"Y": 58,"W": 16,"H": 16},

{"Type": 3, "Id": 0,"X": 279,"Y": 200,"W": 32,"H": 35},

{"Type": 3, "Id": 1,"X": 327,"Y": 200,"W": 32,"H": 35},

{"Type": 3, "Id": 2,"X": 375,"Y": 200,"W": 32,"H": 35},

{"Type": 3, "Id": 3,"X": 423,"Y": 200,"W": 32,"H": 35}

]

},

{

"Name": "JogBank", "Controls":

[

{"Type": 3, "Id": 4,"X": 671,"Y": 75,"W": 130,"H": 130}

]

},

{

"Name": "JoyBank", "Controls":

[

{"Type": 5, "Id": 1,"IdA1": 16,"IdA2": 17,"IdS": 52,"X": 729,"Y": 321," W": 45,"H": 45}

]

} ],

"PersistentVariables":

[

{"Name": "THRESHOLD", "Id": 1},

{"Name": "DEBOUNCE", "Id": 2},

{"Name": "ENCMS", "Id": 4}

],

"Addons":

[

{"Name": "FOOTWERK", "AddonAddress":0 }

]

}

Primer koda 2: Primer korisni£kog pro la

{

"ConnectedDevices":

{

"53": {

"COMPort": null, "DeviceId": 53, "Controlls":

{

"Analog": {

"0": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {"

PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [1, 0]}}},

"1": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [2, 0]}}},

"2": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [3, 0]}}},

"3": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [4, 0]}}},

"4": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [9, 0]}}},

"5": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [10, 0]}}},

"6": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [11, 0]}}},

"7": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [12, 0]}}},

"8": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [5, 0]}}},

"9": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [6, 0]}}},

"10": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [7, 0]}}},

"11": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [8, 0]}}},

"12": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [13, 0]}}},

"13": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [14, 0]}}},

"14": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [15, 0]}}},

"15": {"BankName": "AnalogBank", "Mappings": {"AnalogBank1": {" PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [16, 0]}}},

"16": {"BankName": "JoyBank"," Mappings": {}},

"17": {"BankName": "JoyBank", "Mappings": {}},

"18": {"BankName": "AnalogPedal", "Mappings": {"AnalogPedal1": {"

PluginName": "MidiPlugin", "FunctionName": "MidiCCAnalog", " args": [100, 0]}}}

},

"Button": {

"0": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCMomentaryButton", "args": [32, 0]}}},

"1": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCMomentaryButton", "args": [33, 0]}}},,

"2": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCMomentaryButton", "args": [34, 0]}}},,

"3": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCMomentaryButton", "args": [35, 0]}}},,

"4": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCToggleButton", "args": [36, 0]}}},,

"5": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCToggleButton", "args": [37, 0]}}},,

"6": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCToggleButton", "args": [38, 0]}}},,

"7": {"BankName": "ButtonBank", "Mappings": {"ButtonBank1": { "

PluginName": "MidiPlugin", "FunctionName": "

MidiCCToggleButton", "args": [39, 0]}}},,

"8": {"BankName": "ButtonBank","Mappings": {}},

"9": {"BankName": "ButtonBank","Mappings": {}},

"10": {"BankName": "ButtonBank","Mappings": {}},

"11": {"BankName": "ButtonBank","Mappings": {}},

"12": {"BankName": "ButtonBank","Mappings": {}},

"13": {"BankName": "ButtonBank","Mappings": {}},

"14": {"BankName": "ButtonBank","Mappings": {}},

"15": {"BankName": "ButtonBank","Mappings": {}},

"16": {"BankName": "ButtonBank","Mappings": {}},

"17": {"BankName": "ButtonBank","Mappings": {}},

"18": {"BankName": "ButtonBank","Mappings": {}},

"19": {"BankName": "ButtonBank","Mappings": {}},

"20": {"BankName": "ButtonBank","Mappings": {}},

"21": {"BankName": "ButtonBank","Mappings": {}},

"22": {"BankName": "ButtonBank","Mappings": {}},

"23": {"BankName": "ButtonBank","Mappings": {}},

"24": {"BankName": "ButtonBank","Mappings": {}},

"25": {"BankName": "ButtonBank","Mappings": {}},

"26": {"BankName": "ButtonBank","Mappings": {}},

"27": {"BankName": "ButtonBank","Mappings": {}},

"28": {"BankName": "ButtonBank","Mappings": {}},

"29": {"BankName": "ButtonBank","Mappings": {}},

"30": {"BankName": "ButtonBank","Mappings": {}},

"31": {"BankName": "ButtonBank","Mappings": {}},

"32": {"BankName": "ButtonBank","Mappings": {}},

"33": {"BankName": "ButtonBank","Mappings": {}},

"34": {"BankName": "ButtonBank","Mappings": {}},

"35": {"BankName": "ButtonBank","Mappings": {}},

"36": {"BankName": "ButtonBank","Mappings": {}},

"37": {"BankName": "ButtonBank","Mappings": {}},

"38": {"BankName": "ButtonBank","Mappings": {}},

"39": {"BankName": "ButtonBank","Mappings": {}},

"40": {"BankName": "ButtonBank","Mappings": {}},

"41": {"BankName": "ButtonBank","Mappings": {}},

"42": {"BankName": "ButtonBank","Mappings": {}},

"43": {"BankName": "ButtonBank","Mappings": {}},

"44": {"BankName": "ButtonBank","Mappings": {}},

"45": {"BankName": "ButtonBank","Mappings": {}},

"46": {"BankName": "ButtonBank","Mappings": {}},

"47": {"BankName": "ButtonBank","Mappings": {}},

"48": {"BankName": "ScreenBank", "Mappings": {}},

"49": {"BankName": "ScreenBank", "Mappings": {}},

"50": {"BankName": "ScreenBank", "Mappings": {}},

"51": {"BankName": "ScreenBank", "Mappings": {}},

"52": {"BankName": "JoyBank", "Mappings": {}},

"53": {"BankName": "DigitalPedals", "Mappings": {"DigitalPedals1"

: {"PluginName": "MidiPlugin", "FunctionName": " MidiCCToggleButton", "args": [101, 0]}}},

"54": {"BankName": "DigitalPedals", "Mappings": {"DigitalPedals1"

: {"PluginName": "MidiPlugin", "FunctionName": " MidiCCMomentaryButton", "args": [102, 0]}}}

},

"RGBLED": {

"0": {"BankName": "ButtonBank", "Mappings": {}},

"1": {"BankName": "ButtonBank", "Mappings": {}},

"2": {"BankName": "ButtonBank", "Mappings": {}},

"3": {"BankName": "ButtonBank", "Mappings": {}},

"4": {"BankName": "ButtonBank", "Mappings": {}},

"5": {"BankName": "ButtonBank", "Mappings": {}},

"6": {"BankName": "ButtonBank", "Mappings": {}},

"7": {"BankName": "ButtonBank", "Mappings": {}},

"8": {"BankName": "ButtonBank", "Mappings": {}},

"9": {"BankName": "ButtonBank", "Mappings": {}},

"10": {"BankName": "ButtonBank", "Mappings": {}},

"11": {"BankName": "ButtonBank", "Mappings": {}},

"12": {"BankName": "ButtonBank", "Mappings": {}},

"13": {"BankName": "ButtonBank", "Mappings": {}},

"14": {"BankName": "ButtonBank", "Mappings": {}},

"15": {"BankName": "ButtonBank", "Mappings": {}}

},

"LCD": {

"0": {"BankName": "ScreenBank", "Mappings": {}}

},

"Encoder": {

"0": {"BankName": "ScreenBank", "Mappings": {}},

"1": {"BankName": "ScreenBank", "Mappings": {}},

"2": {"BankName": "ScreenBank", "Mappings": {}},

"3": {"BankName": "ScreenBank", "Mappings": {}},

"4": {"BankName": "JogBank", "Mappings": {}}

}

},

"ActivePage": {

"AnalogBank": "AnalogBank1",

"ButtonBank": "ButtonBank1",

"ScreenBank": "ScreenBank1",

"JogBank": "JogBank1",

"JoyBank": "JoyBank1",

"AnalogPedal": "AnalogPedal1",

"DigitalPedals": "DigitalPedals1"

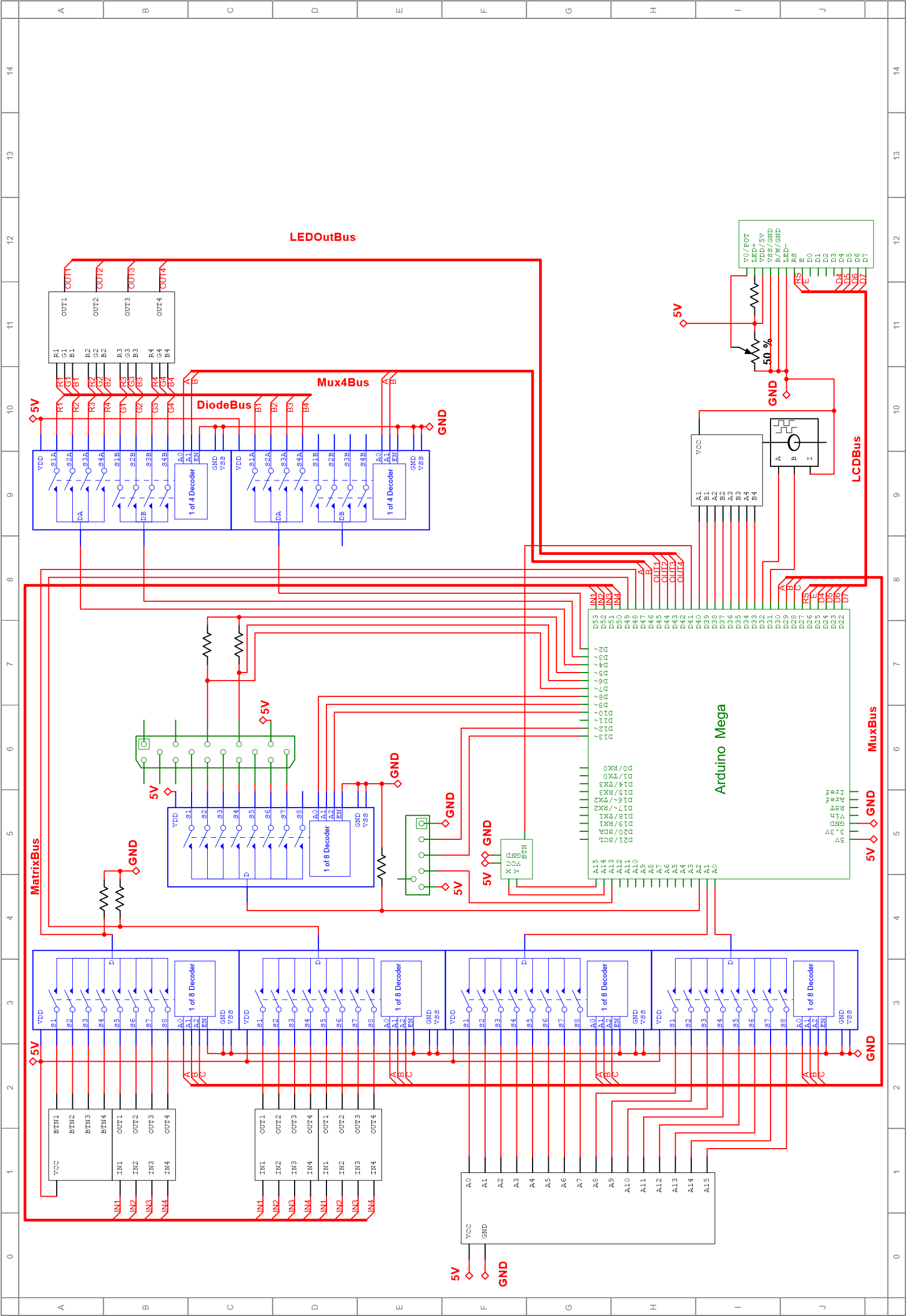
}

}

},

"PluginStates": { }

}



Slika 1: ema elektronike

# Literatura

1. Arduino AG. Arduino. 2019. url: [https://www.arduino.cc/.](https://www.arduino.cc/)
2. Arduino AG. Reading Rotary Encoders. 2019. url: [https://playground.](https://playground.arduino.cc/Main/RotaryEncoders/)

[arduino.cc/Main/RotaryEncoders/.](https://playground.arduino.cc/Main/RotaryEncoders/)

1. MIDI Manufacturers Association. MIDI. 2019. url: [https://www.midi.org/.](https://www.midi.org/)
2. Dave Dribin. Keyboard Matrix Help. 2000. url: [https://www.dribin.org/ dave/keyboard/one\_html/.](https://www.dribin.org/dave/keyboard/one_html/)
3. SparkFun Electronics. Button Pad Hookup Guide. 2019. url: [https://learn.](https://learn.sparkfun.com/tutorials/button-pad-hookup-guide)

[sparkfun.com/tutorials/button-pad-hookup-guide.](https://learn.sparkfun.com/tutorials/button-pad-hookup-guide)

1. Tobias Erichsen. virtualMIDI SDK. 2019. url: [http://www.tobias-erichsen](http://www.tobias-erichsen.de/software/virtualmidi/virtualmidi-sdk.html).

[de/software/virtualmidi/virtualmidi-sdk.html.](http://www.tobias-erichsen.de/software/virtualmidi/virtualmidi-sdk.html)

1. Number Eight Innovation. THREE METHODS TO FILTER NOISY ARDUINO MEASUREMENTS. 2019. url: [https://www.megunolink.com/ articles/coding/3-methods-filter-noisy-arduino-measurements/.](https://www.megunolink.com/articles/coding/3-methods-filter-noisy-arduino-measurements/)
2. EETech Media LLC. Switch Bounce and How to Deal with It. 2019. url:

[https://www.allaboutcircuits.com/technical- articles/switchbounce-how-to-deal-with-it/.](https://www.allaboutcircuits.com/technical-articles/switch-bounce-how-to-deal-with-it/)

1. Blackmagic Design Pty. Ltd. Blackmagic Design. 2019. url: [https://www.](https://www.blackmagicdesign.com/)

[blackmagicdesign.com/.](https://www.blackmagicdesign.com/)

1. COMPULITE SYSTEMS LTD. Compulite. 2019. url: [http://www.compulite](http://www.compulite.com/). [com/.](http://www.compulite.com/)
2. Microsoft. Microsoft Developer Network. 2019. url: [https://msdn.microsoft](https://msdn.microsoft.com/).

[com](https://msdn.microsoft.com/).

1. Microsoft. Using threads and threading. 2019. url: [https://docs.microsoft. com/en-us/dotnet/standard/threading/using-threads-and-threading.](https://docs.microsoft.com/en-us/dotnet/standard/threading/using-threads-and-threading)

LITERATURA

1. JSON org. Introducing JSON. 2019. url: [https://www.json.org/.](https://www.json.org/)
2. Wikipedia. Keyboard matrix circuit. 2019. url: [https://en.wikipedia.org/ wiki/Keyboard\_matrix\_circuit.](https://en.wikipedia.org/wiki/Keyboard_matrix_circuit)
3. Wikipedia. Multiplexer. 2019. url: [https://en.wikipedia.org/wiki/ Multiplexer.](https://en.wikipedia.org/wiki/Multiplexer)

1. Professional Hardware Interfaces Control Center [↑](#footnote-ref-1)
2. Human Interface Device [↑](#footnote-ref-2)
3. Integrated Development Environment - Integrisano razvojno okru”enje [↑](#footnote-ref-3)
4. Odgovara naponu od 5V [↑](#footnote-ref-4)
5. Odgovara naponu od 0V [↑](#footnote-ref-5)
6. Brzina prenosa podataka, u digitalnoj komunikaciji ekvivalentno sa brojem bitova u sekundi 2115.2 kbit\s [↑](#footnote-ref-6)
7. Pogledati 1.1 [↑](#footnote-ref-7)
8. Electrically Erasable Programmable Read-Only Memory [↑](#footnote-ref-8)
9. Okrenuti za jedan stepen u smeru kazaljke na satu [↑](#footnote-ref-9)
10. Okrenuti za jedan stepen u smeru suprotnom od kazaljke na satu [↑](#footnote-ref-10)
11. Po tri za svaku LED diodu [↑](#footnote-ref-11)
12. Po dva za svaki enkoder [↑](#footnote-ref-12)
13. Za displej [↑](#footnote-ref-13)
14. pinove na kolonama√ √ [↑](#footnote-ref-14)
15. √*N*×3∗√*N* matrica tastera [↑](#footnote-ref-15)
16. *N*×3∗ *N* matrica LED dioda [↑](#footnote-ref-16)
17. N broj ulaza multipleksera, M broj multipleksera [↑](#footnote-ref-17)
18. Selektori za multipleksere sa 8 kanala [↑](#footnote-ref-18)
19. Ulazi za multipleksere sa 8 kanala [↑](#footnote-ref-19)
20. Selektori za multipleksere sa 4 kanala [↑](#footnote-ref-20)
21. Izlazi za multipleksere sa 4 kanala [↑](#footnote-ref-21)
22. Izlazi za matricu tastera [↑](#footnote-ref-22)
23. Izlazi za matricu LED dioda [↑](#footnote-ref-23)
24. Sve biblioteke su sa£injene od istoimenih .cpp i .h fajlova [↑](#footnote-ref-24)
25. Algoritam za uklanjanje odskakivanja [↑](#footnote-ref-25)
26. Prelaz pina sa LOW na HIGH stanje [↑](#footnote-ref-26)
27. Prelaz pina sa HIGH na LOW stanje [↑](#footnote-ref-27)
28. Videti 3.3 [↑](#footnote-ref-28)
29. U ve¢ini slu£ajeva je neophodno podesiti dodatne parametre da bi kontrola funckionisala kako ”elimo [↑](#footnote-ref-29)
30. Trenutno aktivni pro l je uvek fajl default.json u Profile direktorijumu [↑](#footnote-ref-30)
31. Npr. za displej je zapisan broj kolona i redova na njemu [↑](#footnote-ref-31)
32. Ure aj koji implementrira GHP protokol kao †to je GPC kontroler [↑](#footnote-ref-32)