

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Gašper Levačič

**Uporabniški vmesnik dirkalnika
tekmovanja Formula Student**

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM
PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: prof. dr. Branko Šter

Ljubljana, 2023

To delo je ponujeno pod licenco *Creative Commons Priznanje avtorstva-Deljenje pod enakimi pogoji 2.5 Slovenija* (ali novejšo različico). To pomeni, da se tako besedilo, slike, grafi in druge sestavnine dela kot tudi rezultati diplomskega dela lahko prosto distribuirajo, reproducirajo, uporabljajo, priobčujejo javnosti in predelujejo, pod pogojem, da se jasno in vidno navede avtorja in naslov tega dela in da se v primeru spremembe, preoblikovanja ali uporabe tega dela v svojem delu, lahko distribuira predelava le pod licenco, ki je enaka tej. Podrobnosti licence so dostopne na spletni strani creativecommons.si ali na Inštitutu za intelektualno lastnino, Strelška 1, 1000 Ljubljana.



Izvorna koda diplomskega dela, njeni rezultati in v ta namen razvita programska oprema je ponujena pod licenco GNU General Public License, različica 3 (ali novejša). To pomeni, da se lahko prosto distribuira in/ali predeluje pod njenimi pogoji. Podrobnosti licence so dostopne na spletni strani <http://www.gnu.org/licenses/>.

Besedilo je oblikovano z urejevalnikom besedil L^AT_EX.

Kandidat: Gašper Levačič

Naslov: Uporabniški vmesnik dirkalnika tekmovanja Formula Student

Vrsta naloge: Diplomska naloga na visokošolskem programu prve stopnje
Računalništvo in informatika

Mentor: prof. dr. Branko Šter

Opis:

Implementirajte uporabniški vmesnik za dirkalnik tekmovanja Formula Student. Predstavite tekmovanje Formula Student, opišite uporabljeno strojno opremo, uporabljena programska orodja in postopek razvoja vgrajene programske opreme.

Title: User interface of a Formula Student race car

Description:

Implement a user interface for a Formula Student racing car. Introduce the Formula Student competition, describe the hardware used, the applied software tools and the embedded software development process.

Zahvaljujem se vsem, ki so z mano v obdobju izobraževanja delili svoje znanje. S tem niso le razvijali mojo intelektualno sposobnost, temveč me tudi oblikovali kot osebo. Zahvalil bi se družini, ki me popolnoma podpira na vseh področjih. Hvala sočlanom ljubljanske ekipe Formule Student za dober timski duh, ki ustvari delovno vzdušje, pa tudi prijateljstva. Hvala prof. dr. Branku Šteru za pomoč pri pisanju diplomske naloge.

Kazalo

Povzetek

Abstract

1	Uvod	1
2	Pregled področja	3
2.1	Formula 1	3
2.2	Produkcijski avtomobili	5
2.3	Vrednotenje	5
3	Tekmovanje Formula Student	7
3.1	Točkovanje	7
3.2	Pravilnik tekmovanja Formula Student	9
3.3	Ekipa FS Univerze v ljubljani	10
3.4	Dirkalnik sezone 2021-22	11
4	Cilji diplomske naloge	13
4.1	Dosedanja rešitev	13
4.2	Cilji za novo rešitev	14
5	Strojna oprema	15
5.1	Plošči tiskanega vezja	16
5.2	Zaslon LCD	19
5.3	Gonilniška plošča Adafruit RA8875	20

5.4	Vhodne naprave	22
6	Vgrajena programska oprema	25
6.1	Uporabljena razvojna orodja	25
6.2	Razvoj vgrajene programske opreme	28
7	Zaključek	39
	Članki v revijah	41
	Celotna literatura	43

Seznam uporabljenih kratic

kratica	angleško	slovensko
FS	Formula Student	/
UV	user interface	uporabniški vmesnik
DRS	drag reduction system	sistem, ki zmanjšuje zračni upor
EV	electric vehicles	električna vozila
CV	internal combustion engine vehicles	vozila z notranjim zgorevanjem
DC	direct current	enosmerni električni tok
TS	tractive system	sistem za napajanje motorjev
TSAL	tractive system active light	svetlobni indikator za sistem za napajanje motorjev
AMS	accumulator management system	sistem za nadzor baterije
IMD	insulation monitoring device	sistem za nadzor baterije
VCU	vehicle control unit	krmilna enota vozila
RTOS	real-time operating system	operacijski sistem v realnem času
SoC	state of charge	stanje napolnjenosti akumulatorja

Povzetek

Naslov: Uporabniški vmesnik dirkalnika tekmovanja Formula Student

Avtor: Gašper Levačič

Povzetek: Uporabniški vmesnik je sistem, ki združuje vhodne in izhodne naprave ter ostalo strojno in programsko opremo, da uporabniku omogoči kar se da enostavno in intuitivno komunikacijo z napravo. V tem diplomskem delu sem opisal razvoj celotnega uporabniškega vmesnika za dirkalnik tekmovanja Formula Student. Formula Student je letno mednarodno študentsko inženirsko tekmovanje, na katerem študentje dirkajo z dirkalniki v stilu formule, ki so jih razvili in izdelali tekom šolskega leta. V armaturno ploščo in volan dirkalnika smo vgradili plošči tiskanega vezja z mikrokrmlnikoma STM32, zaslon LCD, gonilniško ploščo za LCD Adafruit RA8875 in druge vhodno/izhodne naprave. Za gradnjo vgrajene programske opreme sem uporabil metodo inkrementalnega razvoja, integrirano razvojno okolje STM32CubeIDE in razhroščevalnik SEGGER J-Link.

Ključne besede: Formula Student, dirkalnik, uporabniški vmesnik, grafični uporabniški vmesnik.

Abstract

Title: User interface of a Formula Student race car

Author: Gašper Levačič

Abstract: A user interface is a system consisting of input/output devices, other hardware devices and software that provides the user an easy and intuitive interaction with a device. In this thesis, I described the development of the entire user interface for the race car of the Formula Student competition class. Formula Student is an annual international engineering design competition for university students, in which teams design, build, and race their own formula-style race cars. For the user interface system, we installed two custom printed circuit boards with STM32 microcontrollers, LCD screen, Adafruit RA8875 driver board and other I/O devices into the car's dashboard and steering wheel. For the embedded software, I used the incremental development method, the STM32CubeIDE integrated development environment and the SEGGER J-Link debugger.

Keywords: Formula Student, race car, user interface, graphical user interface.

Poglavlje 1

Uvod

Vmesnik je mesto, kjer se srečujejo in delujejo neodvisni sistemi (npr. človek - stroj), ki med seboj lahko komunicirajo. Uporabniški vmesnik (kratica UV, angl. *user interface*, kratica *UI*) je okolje, ki združuje vhodne in izhodne naprave ter programsko opremo, da uporabniku omogoči kar se da enostavno in intuitivno komunikacijo z napravo. V tem diplomskem delu bom opisal razvoj grafičnega uporabniškega vmesnika za dirlkalnik razvojne sezone 2021/2022, ekipe Formule Student Univerze v Ljubljani, ki se imenuje Superior engineering.

Uporabniški vmesnik močno vpliva na način delovanja produkta, zato se moramo zavedati njegove pomembnosti in ga v razvojnem procesu posebej obravnavati. Snovanje uporabniškega vmesnika pa je zahteven proces, saj smo se razvijalci sistema primorani postaviti v kožo končnega uporabnika. Razviti produkt, ki bo uporabniku omogočal doseganje določenih ciljev na intuitiven način, ni mačji kašelj. Težava ponavadi nastane, ker inženirji v večini primerov ne razmišljamo na enak način kot končni uporabnik in to je tisto, kar zadevo naredi kompleksno. To sicer ni nič slabega, temveč ravno nasprotno, v zahtevnih problemih človek najde izziv in ko ga reši, se počuti zadovoljno, bolj samozavestno, hkrati pa je z iskanjem rešitve pridobi novo znanje in izkušnje.

Področje uporabniških vmesnikov zato mnogokrat uhaja iz inženirskih

voda in se križa tudi z drugimi vedami, kot sta na primer psihologija in oblikovanje. Na začetku načrtovanja UV je potrebno kar se da najbolje razumeti uporabnikov problem, nato pa uporabnika vključevati skozi celoten razvoj in z njim preverjati posamezne iteracije rešitve. Poleg tega je treba poskrbeti, da se držimo principov dobrega snovanja, ki narekujejo lastnosti dobrega vmesnika, kot so na primer estetska všečnost, jasnost, konsistentnost ipd.

Moja naloga za razvojno sezono 2021/2022 pri ljubljanski ekipi Formule Student je bila razviti grafični vmesnik, ki bo našim voznikom pri visokih hitrostih prikazoval najpomembnejše podatke na intuitiven način, pri tem pa jim ukradel minimalno količino pozornosti, ki bi jo sicer namenili obnašanju dirkalnika, progi - dirkanju. Kot sem že omenil, snovanje UV terja veliko truda že za končnega uporabnika v ”normalnih” okoliščinah.

Ne samo voznikom, rad bi olajšal delo tudi študentom, ki delajo na dirkalniku. To bi storil s prikazovanjem nekaterih diagnostičnih podatkov na zaslonu. Imam tudi idejo o sprostivenem programu, ki bi ga lahko vozniki uporabljali, ko na vročem soncu čakajo na pregled.

2. poglavje vključuje področje vmesnikov v profesionalnem motošportu.
3. poglavje vsebuje kratko predstavitev tekmovanja Formula Student. 4. poglavje našteje cilje diplomske naloge. Definiral sem funkcionalnosti, ki jih sem razvil. 5. poglavje predstavi strojno opremo - uporabljeno tehnologijo, izdelavo tiskanega vezja, ključne komponente, ki so na vezju. 6. poglavje opisuje postopek razvoja programske opreme in uporabljenih orodja. Sledil bo samo še zaključek.

Poglavlje 2

Pregled področja

2.1 Formula 1

Avtomobilistično dirkanje je šport, ki ima na svetu ogromno privržencev. Pojavil se je takoj po iznajdbi prvih uspešnih bencinskih avtomobilov in se danes uveljavlja kot eden izmed najbolj priljubljenih športov na svetu. Omenjena športna panoga se deli na veliko kategorij in serij, med katerimi je verjetno vsem najbolj poznana Formula 1 (kratica F1).

Formula 1 je najvišji in najbolj priljubljen razred motošporta. Dirke Formule 1, znane tudi kot Velike nagrade, so veliki televizijski dogodki z milijoni gledalcev v več kot dvestotih državah. Ogled dirk v živo simfonije izpušnih cevi dirkalnikov in vonj po izgorelem motornem olju ter bencinu povzroča gledalcem kurjo polt.

Na dirkalnik razreda Formule 1 se bom v nadaljevanju osredotočil kot zgled dobrega dirkalnika, saj F1 predstavlja kraljico motošporta. Ekipe tega razreda imajo tudi najbogatejše proračune. Le iz tega podatka sicer še ne moremo zagotovo trditi, da so dirkalniki F1 najboljši, iz predstav in doseženih časov le teh pa lahko.

V enosedelzih dirkalnikih F1 so kolesa odkrita, dirkalniki pa so opremljeni z aerodinamičnimi prednjimi in zadnjimi krilci, ki povzročajo podtlak in s tem povečujejo stik dirkalnika s podlogo. Dirkači F1 dirkajo pri velikih

hitrostih, ki pogosto presežejo tudi 300 km/h. Da lahko dirkalnik preseže in vzdrži takšno hitrost, nam pove, da gre za dovršen stroj in pravo inženirsko mojstrovino. Vsako leto se inženirji ekip trudijo razviti hitrejši dirkalnik in uvesti nove tehnologije, ki bi jim dale prednost pred konkurenco.

2.1.1 Volan dirkalnika



Slika 2.1: volan dirkalnika F1, ekipe Mercedes-AMG Petronas, sezona 2021

Na Sliki 2.1 je volan ekipe F1, ki je tisto sezono osvojila konstruktorški naslov prvaka. Volan na dirkalniku predstavlja vmesnik človek-stroj, ki vozniku omogoča pregled nad stanjem dirkalnika in krmiljenje le-tega [5].

Na sredini ima dve izhodni napravi, in sicer so poleg zaslona LCD še LED diode, ki vozniku naznanjajo, kdaj prestaviti v višjo prestavo. Vozniki jih ponavadi imajo v različnih barvah, ki predstavljajo določene razpone vrtljajev motorja.

Položaj omenjenih izhodnih naprav je lahko na volanu ali na armaturni plošči, odvisno od voznikovih preferenc. Obe konfiguraciji imata svoje prednosti in slabosti. Položaj na volanu nudi slabšo vidljivost med zavijanjem, a poveča voznikov prostor za noge, medtem ko ima položaj na armaturi ravno

obratne lastnosti, saj je vgrajen v šasijo.

Programsko je implementiranih več zaslakov, vsak vozniku prikaže določene podatke, to so podatki o telemetriji, trenutni prestavi, čas kroga, razlika med avtom spredaj ter zadaj, stanje napoljenosti baterije, vpogled v zmogljivost avta in morebitne napake. Kateri zaslon je dejansko prikazan na zaslonu LCD, voznik določa s sredinskim rotacijskim gumbom.

Poleg izhodov ima voznik F1 na voljo še več kot dva ducata vhodov - gumbov, vrtljivih gumbov in prestavnih ročic. Z njimi lahko med drugim nastavlja moč zaviranja motorja, DRS (sistem za zmanjšanje zračnega upora, angl. *drag reduction system*) razmerje navora v diferencialu v različnih fazah ovinka, aktivira sklopko ter črpalko za tekočino za pitje in še mnogo več.

2.2 Produkcijski avtomobili

Nekaj tehnoloških napredkov omenjenega razreda F1 se je uveljavilo tudi v produkcijskih avtomobilih, kot na primer hibridni pogon, karbonska vlakna, aktivno vzmetenje in, najbolj relevantno za to diplomsko delo, gumbi na volanu. V avtomobilski industriji se je zelo razširilo načrtovanje uporabniških vmesnikov [1]. Uporabniška izkušnja je postala dodana vrednost in avtomobilski proizvajalci razvoju le te posvečajo ogromno časa, kar se vidi v novejših avtomobilih. Njena funkcionalnost je predvsem uporabniku omogočiti enostavno in intuitivno nastavljanje ogromno funkcij za varnost in udobje. Tak uporabniški vmesnik seveda ni relevanten za dirkalnik. UV v dirkalniku mora biti kar se da minimalen, da čim manj odvrača pozornost vozniku.

2.3 Vrednotenje

Razvoj izdelka, ki ga opisujem v tej nalogi, ni namenjen dirkalniku F1, temveč dirkalniku tekmovanja Formula Student. Že iz virov, ki jih imajo na voljo ekipe, teh dveh razredov nikakor ne moremo primerjati. Razlikujejo se tudi pravila, ki jih bom predstavil v naslednjem poglavju.

Prednost, ko govorimo o željah implementacije vmesnika, sem dal vozni-kom naše ekipe. Z njimi smo skupaj skicirali osnutek ekrana, prikazanega med vožnjo. Če bi za volanom sedel svetovni prvak F1, Max Verstappen, bi bile želje, kaj prikazati na zaslonu, verjetno drugačne. Veliko tehnologij F1 na našem dirkalniku še ni implementiranih, nekaj pa jih tudi nikoli ne bo zaradi omenjenih razlik razredov. Skratka, dirkalnik F1 je vrhunski zgled, a v primerjavi s tekmovanjem FS gre za dva popolnoma različna svetova, zato je smiselno, da je tudi vmesnik prilagojen zahtevam vsakemu razredu.

Poglavlje 3

Tekmovanje Formula Student

Formula Student [14] je inženirsko tekmovanje, pri katerem sodelujejo univerze z vsega sveta. Študenti vsako sezono skozi študijsko leto razvijejo in izdelajo enosedežni dirkalnik, formulo manjše velikosti, nato pa poleti tekmujejo z ostalimi ekipami na tekmovanjih. Tekmovanje ni osvojeno le z najhitrejšim avtomobilom, temveč z najboljšo konstrukcijo in zmogljivostjo dirkalnika ter finančnega in prodajnega načrtovanja. Tovrstno tekmovanje hkrati spodbuja ekipni duh ter reklamira motošport in inženirstvo med mladimi.

Študenti sicer lahko prejemamo nasvete in kritike profesionalnih inženirjev ali učiteljev s fakultete, a moramo celotno zasnovno avtomobila narediti samostojno. Sami smo odgovorni za zbiranje sredstev.

3.1 Točkovanje

Člani ekipe tekmovanja FS se soočajo z izzivom, da naredijo dodaten korak v svojim izobraževanju, tako da vanj vključijo še izdelavo in proizvodnjo ter se srečajo z ekonomskimi vidiki avtomobilske industrije. Ekipe predpostavljajo, da so proizvajalci, ki razvijajo prototip, ki ga je treba oceniti za proizvodnjo. Dirkalnik mora pokazati zelo dobre vozne lastnosti, kot so pospeševanje, zaviranje in vodljivost. Ponujen mora biti po razumni ceni in biti



Slika 3.1: Zagovor načrtovanja dirkalnika pred sodniki, tekmovanje FS Austria, sezona 2021-22

zanesljiv. Poleg tega se avtomobil ocenjuje še s stališča drugih dejavnikov, kot sta estetika in udobje.

Ocenjevanje na tekmovanju je sestavljeno iz statičnih in dinamičnih preizkušenj. Točkovanje na statičnih preizkušnjah je zaupano žiriji strokovnjakov iz motošportne in avtomobilske industrije ter njihovim dobaviteljem. Žirija oceni avto in poročilo o stroških vsake ekipe na podlagi konstrukcije, načrtovanja stroškov in prodajne predstavitev. Slika 3.1 prikazuje zagovor načrtovanja naše ekipe pred sodniki na tekmovanju v Avstriji. Preostali del tekmovanja poteka na stezi, kjer študentje na različnih dinamičnih preizkušnjah pokažejo, kako se dirkalniki odrežejo v njihovem naravnem habitatiju.

Točkovanje se lahko razlikuje od tekmovanja do tekmovanja, v večini FS tekmovanj pa so točke razdeljene med dogodki po formatu, ki ga prikazuje Tabela 3.1.

Tabela 3.1: Točkovanje tekmovanja Formula Student

Statične preizkušnje	
Dogodek o načrtovanju	150
Dogodek o analizi stroškov in proizvodnje	100
Predstavitev poslovnega načrta	75
Dinamične preizkušnje	
Preizkušnja pospeška	50
Preizkušnja Skidpad	50
Preizkušnja avtokros	100
Preizkušnja vzdržljivosti	250

3.2 Pravilnik tekmovanja Formula Student

Pravilnik tekmovanja FS (v orig. *Formula Student Rules*) je dokument, v katerem so definirana vsa pravila in omejitve, ki jih študenti moramo upoštevati pri snovanju in konstrukciji dirkalnika. Skladnost s pravili dirkalnika se na dirkah preveri na tehničnih inšpekcijs (angl. *scrutineering*). Ta so tudi opisana v pravilniku. Če ekipa ni opravila vseh pregledov, nima vstopa na dinamične preizkušnje.

3.2.1 Predpisi

Formula Student ima relativno malo predpisov. Ekipa mora biti sestavljena le iz rednih študentov, vključno z vozniki, kar omejuje razpoložljive delovne ure, nabor spretnosti in izkušenj ter prinaša različne druge izzive, s katerimi se profesionalne dirkaške ekipe ne srečujejo. Ta omejitev tudi pomeni, da so lahko ostali predpisi manj omejujoči, kot bi sicer bili v svetu profesionalnega dirkanja.

Predpisi se med drugim delijo na dva sklopa, in sicer CV in EV. Prva so relevantna le za vozila s pogonom na notranje zgorevanje (angl. *Combustion Vehicles, CV*), slednja, relevantna za našo ekipo, pa le za električna vozila (angl. *Electric Vehicles, EV*). Hibridni pogon ni dovoljen. Zadnja leta se je

pojavila tudi nova kategorija, in sicer samovozeči dirkalniki (angl. *driverless vehicles, DV*).

CV morajo imeti batni, 4-taktni motor z največjo prostornino 710 cm³. Obvezna je uporaba omejevalca pretoka zraka z največjim dovoljenim premerom 20 mm. Moč pogonskega sklopa v EV ne sme presegati 80 kW. Največja dovoljena napetost, ki se lahko pojavi na avtu je 600V DC za motorje in inverteje, za ostale signale pa 630V DC.

Omejitev mase dirkalnika ni.

Na dirkalniku morajo obvezno biti implementirane še določene varnostne tehnologije, kot so element za prevzemanje energije pri trku na nosu vozila, dva jeklena obroča za varnost pri prevračanju, 2 hidravlična zavorna kroga, 6- ali 7-točkovni varnostni pas. Tudi na armaturni plošči so potrebni določeni varnostni sistemi, ki sem jih moral implementirati. Več v nadaljevanju.

3.3 Ekipa FS Univerze v ljubljani



Slika 3.2: Skupinska slika ekipe FS, Superior engineering, sezona 2021-22

V Sloveniji delujeta dve ekipi študentske formule - mariborska in ljubljanska. Superior engineering je ekipa Univerze v Ljubljani, katere član sem tudi

sam. Z letošnjo sezono je izdelala že svoj četrti električni dirkalnik. Ekipo skupaj z dirkalnikom prikazuje skupinska Slika 3.2. Štajerci, po drugi strani, zaenkrat še vedno prisegajo na pogon z notranjim zgorevanjem. Po njihovih besedah naj bi se to kmalu spremenilo. Na splošno je prehajanje na elektromotorje trenutno trend v tekmovanju, saj je tako tudi v celotni avtomobilski industriji.

Sicer ima tudi naša ekipa korenine v pogonu z notranjim zgorevanjem. Prva dva dirkalnika sta bila namreč bencinska. Prvi je bil opremljen s motorjem Aprilia RXV 450, slednji pa s Triumph Daytona 675. Nato smo na pobudo sponzorjev prešli na električni pogon. Čudovitega zvoka in pokanja izpušne cevi Triumphovega 3-valjnika ni več, ostaja pa strast do dirkanja in inženirstva.

3.4 Dirkalnik sezone 2021-22

Letos smo izdelali naš četrti dirkalnik z električnim pogonom. Prvič po našem prvem električnem dirkalniku smo naredili nov kalup za šasijo. Vzmetenje in celotna aerodinamika sta bila konkretno prenovljena.

Zadnji dve polosi vrtita dva elektromotorja uspešnega slovenskega podjetja Emrax. Vsak je omejen 40 kW, da skupaj ne presežeta najvišje dovoljene moči 80 kW. Napetost visokonapetostnega sistema je 588 V. Tehta 220 kg, pri masi imamo še prostor za izboljšavo.

Sicer gre za vrhunski dirkalnik, ki vsekakor ima potencial konkurirati nemško govorečim ekipam. Slika 3.3 prikazuje dirkalnik na progi na dirkališču Redbull Ring.



Slika 3.3: Dirkalnik ekipе Superior engineering na poligonu, Redbull ring, sezona 2021-22

Poglavlje 4

Cilji diplomske naloge

4.1 Dosedanja rešitev

Omenjeni FS ekipi Superior engineering sem se v sezoni 2021/22 pridružil z namenom, da bi prenovil in nadgradil uporabniški vmesnik dirkalnika. Ta je bil v dosedanjih sezонаh precej osnoven, prikaz je bil sestavljen iz 7- ali 16-segmentnih zaslonov ter LED diod. Slika 4.1 prikazuje uporabniški vmesnik na armaturni plošči lanskega dirkalnika.



Slika 4.1: Armaturna plošča dirkalnika Tabaluga v sezoni 2020/21

Ta rešitev je precej omejujoča, saj lahko prikazujemo le številke (ozioroma znake) in še to le tri. V tem primeru je bilo torej med delovanjem dirkalnika mogoče prikazati le eno vrednost naenkrat, in sicer stanje napoljenosti (angl. *state of charge, SoC*)

4.2 Cilji za novo rešitev

Glavna ideja za prenovo je bila uporaba zaslona LCD. Ta bi poleg znakov omogočal prikaz poljubnih grafik, na primer vrstice napredka. Zaslon LCD bi moral izboljšati tudi vidnost prikaza vmesnika. Uporabiti je bilo potrebno zaslon z visoko svetlostjo, prav tako pa je bila potrebno korektno oblikovati grafiko, da je bil dosežen dober kontrast. Dober kontrast je ena izmed višjih prioritet pri tem projektu. Voznik mora jasno razbrati informacije na zaslonu v vseh vremenskih pogojih. Poleg tega mora biti še enostaven, kar pomeni, da prikazuje le kritične informacije.

Programsko sem implementiral več zaslonov. Med njimi se lahko uporabnik "premika" z uporabo vhodov na volanu. Poleg zaslona za vožnjo sem implementiral še zaslona za diagnostiko elektronskih sistemov na avtu. S tem sem razširil funkcionalnost UV in ga naredil uporabnega še za študente, ki delajo na električnem sistemu avtomobila.

Paziti moram tudi, da pokrijem zahteve iz pravilnika FS. Na armaturni plošči morajo biti locirane 3 LED diode (vidne tudi na Slikah 4.1 in 5.1):

1. Zelena svetleča dioda TS OFF (*Tractive System OFF*) sveti takrat, ko naprava TSAL (*Tractive System Active Light*) na avtu sveti zeleno; pomeni, da avto ni pod visoko napetostjo in je varen za delo.
2. Rdeča svetleča dioda AMS (*Accumulator Management System*) sveti takrat, ko naprava AMS zahteva izklop avta.
3. Rdeča svetleča dioda IMD (*Insulation Monitoring Device*) sveti takrat, ko naprava IMD zahteva izklop avta.

Poglavlje 5

Strojna oprema

Strojna oprema se nanaša na fizične komponente računalniškega sistema ali elektronske naprave. V tem poglavju bom opisal tehnologijo, ki sem jo uporabil za strojno opremo - plošči tiskanega vezja, zaslon LCD, gonilniško ploščo za LCD in vhodne naprave (gumbe, LED diode, absolutne rotacijske enkoderje). Slika 5.1 prikazuje armaturno ploščo in volan dirkalnika z vgrajeno strojno opremo.



Slika 5.1: Armaturna plošča in volan dirkalnika z vgrajeno strojno opremo

5.1 Plošči tiskanega vezja

Plošča tiskanega vezja [6] (angl. *Printed Circuit Board, PCB*) je ravna plošča, narejena iz izolacijskega materiala (npr. steklena vlakna), na katero so nato položene elektronske komponente. Komponente med seboj povezujejo na površino plošče vgravirane poti iz prevodnega materiala, običajno bakra.

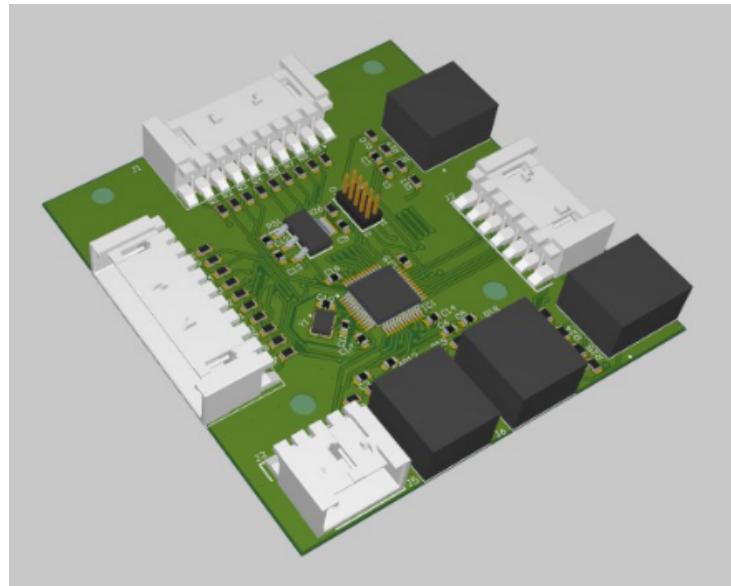
PCB-ji se uporabljajo v številnih elektronskih napravah, od preprostih igrač do kompleksnih računalniških sistemov. Služijo kot osnova za povezovanje različnih elektronskih komponent, kot so upori, kondenzatorji in integrirana vezja, da tvorijo delajoč elektronski sistem.

Izdelava tiskanih vezij je zelo zapletena, zahteva velike naložbe v opremo in več kot 50 procesnih korakov. Izdelavo vezij, ki jih razvijemo v ekipi Superior Engineering, prepustimo podjetju Intectiv iz Kranja. Intectiv je napredno slovensko podjetje, ki izdeluje prototipne plošče tiskanega vezja za domač in tuj trg. Z ekipo smo si tudi ogledali njihovo proizvodnjo in sofisticirano opremo, ki jo uporablja pri procesih proizvodnje.

Dve različni tiskani vezji sta bili po meri izdelani za namene uporabniškega vmesnika dirkalnika. Tiskani vezji je zasnoval kolega iz Fakultete za elektrotehniko, moja naloga pa je bila razviti programsko opremo. Eno izmed tiskanih vezij je na dirkalnik vgrajeno za armaturno ploščo, drugo pa v volan. Njuna naloga je krmiljenje vhodno-izhodnih naprav na teh dveh delih dirkalnika.

5.1.1 Plošča tiskanega vezja Volan

Plošča tiskanega vezja Volan je vgrajena v volan dirkalnika. Osrednja komponenta vezja je mikrokrmlnik STM32F303CCT6, ki bere stanje vhodnih naprav na volanu, te pa posreduje Armaturni plošči prek protokola SPI. Poleg mikrokrmlnika so na plošči še 10-pinski SWD priključek, regulator napetosti in zunanjji oscilator frekvence 24 MHz. Beli priključki so namenjeni povezavi z vhodnimi napravami, desni srednji pa povezavi z Armaturno ploščo - komunikaciji SPI in napajanju 5 V. Povezava z Armaturno ploščo poteka še

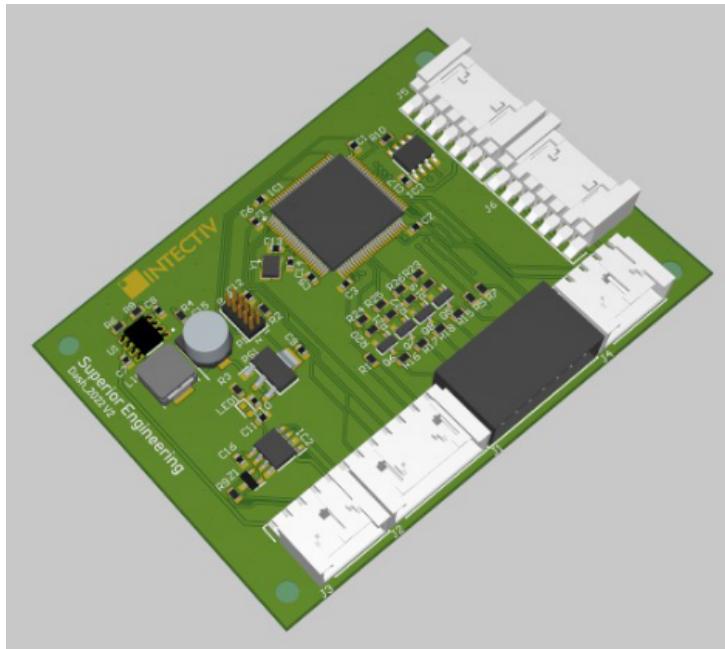


Slika 5.2: 3D upodobitev plošče tiskanega vezja Volan

prek priključka za hitro sprostitev, ki omogoča snetje volana. 3D upodobitev plošče tiskanega vezja je prikazana na Sliki 5.2.

5.1.2 Plošča tiskanega vezja Armaturna plošča

Plošča tiskanega vezja Armaturna plošča je vgrajena v armaturno ploščo dirkalnika. Armaturna plošča ima zmogljivejši mikrokrmilnik STM32H743VIT6 z najvišjo frekvenco delovanja 400 MHz. Poleg komponent, enakim tistim na Volanu, ima še sprejemnik CAN. Vodilo CAN Armaturni plošči omogoča komunikacijo z ostalimi napravami na dirkalniku. Na priključkih so še izhodi za dva gumba in štiri LED, krmilni signali protokola SPI za volan ter gonilniško ploščo RA8875. 3D upodobitev plošče tiskanega vezja je prikazana na Sliki 5.3.



Slika 5.3: 3D upodobitev plošče tiskanega vezja Armaturna plošča

5.1.3 Mikrokrmlniki STM32

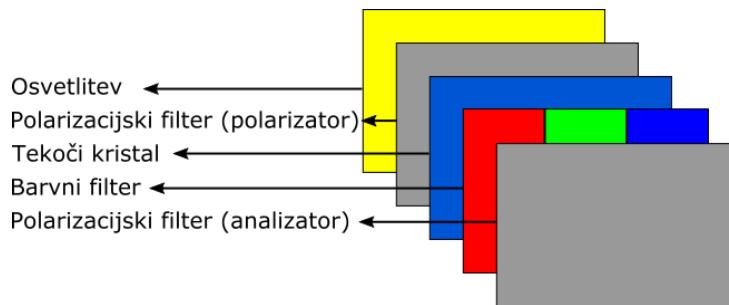
Mikrokrmlnik je kompaktno integrirano vezje, namenjeno opravljanju določene operacije ali izvajaju aplikacije v vgrajenem sistemu. Tipičen mikrokrmlnik vključuje centralno procesno enoto (CPE), pomnilnik in vhodno/izhodne naprave, vse na enem čipu.

STM32 je družina 32-bitnih mikrokrmlnikov, ki temeljijo na procesorskem jedru ARM Cortex-M. Omenjene mikrokrmlnikee izdeluje podjetje STMicroelectronics in se pogosto uporabljajo v različnih aplikacijah, vključno z industrijsko avtomatizacijo, potrošniško elektroniko in avtomobilskimi sistemi. Mikrokrmlniki STM32 so na voljo v različnih konfiguracijah, od naprav z nizko porabo energije s preprostimi perifernimi napravami do visoko zmogljivih naprav z naprednimi funkcijami, kot so povezovanje v omrežje, povezljivost USB in digitalna obdelava signalov. STMicroelectronics nudi tudi različna razvojna orodja, med drugim integrirano razvojno okolje (IDE) STM32CubeIDE, razvojno okolje za programiranje, razhroščevanje napak in

testiranje aplikacij.

5.2 Zaslon LCD

Zaslon LCD [18] (angl. *Liquid Crystal Display*) ali tekočekristalni zaslon je zaslon, ki za prikazovanje slike uporablja tehnologijo tekočih kristalov. LCD je sestavljen iz več plasti, vključno z dvema polarizacijskema filtrom in tekočim kristalom med njima. Izhodni polarizacijski filter je prekrižan glede na vhodnega in prepušča le svetlogo s polarizacijo v pravokotni smeri glede na vpadno polarizacijo. Z električno napetostjo uravnnavamo zasuk tekočega kristala in posledično količino svetlobe, ki prehaja skozi sistem plasti. Kristali sami od sebe ne oddajajo svetlobe, zato je treba poskrbeti še za osvetlitev.



Slika 5.4: Sestavni deli barvnega LCD zaslona

Zasloni LCD so zelo priljubljeni in se pogosto uporabljajo v različnih elektronskih napravah. Praktično povsem so že nadomestili starejše zaslonske tehnologije, kot je CRT (katodne cevi). Prednosti tehnologije LCD so manjša poraba energije, višja ločljivost, širši koti gledanja in tanjši profil.

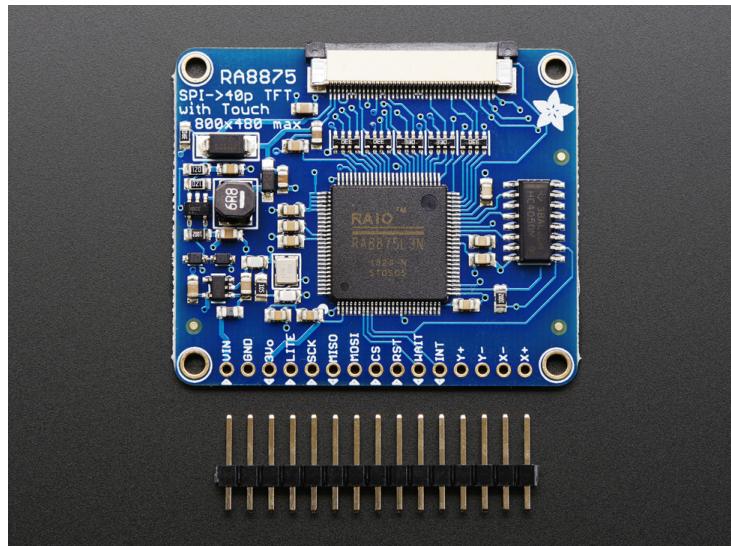
5.2.1 Izbira zaslona LCD

Pri izbiri zaslona je bila glavna zahteva visoka svetlost. Poleg tega je bilo treba upoštevati velikost armaturne plošče dirlkalnika, da se ji bo zaslon prilegal in ne bo prevelik. Odločil sem se za Winstar WF50FSWAGDNN0 [8].

Gre za 5-palčni zaslon LCD z resolucijo 800x480 in ležečo postavitvijo slike. Model zaslona je različica z visoko svetlostjo 1000 cd/m^2 . Ima 24-bitni RGB vmesnik, priključek pa je 40-pinski FPC. Zaslon bom krmilil s pomočjo gonilniške ploščice.

5.3 Gonilniška plošča Adafruit RA8875

Zaslon LCD bom krmilil s pomočjo plošče RA8875 podjetja Adafruit. Adafruit [3] je podjetje, ki načrtuje in prodaja elektronske komponente. Ponujajo široko paleto izdelkov, vključno z mikrokrmilniki, senzorji, zasloni, napajalniki in orodjem. Osredotočeni so na odprtokodnost njihovih izdelkov, tako programske kot tudi strojne opreme. Podjetje ponuja tudi vadnice, projektne ideje in forum, ki inženirjem in začetnikom pomagajo pri začetku njihovih projektov.



Slika 5.5: Gonilniška plošča Adafruit RA8875. Na zgornjem robu je 40-pinski priključek, na spodnjem pa priključki za vmesnik SPI in ostale krmilne signale. V sredini je mikrokrmilnik RAiO RA8875, desno od mikrokrmilnika je eksterni pomnilnik RAM. Ostale komponente služijo napajanju.

Adafruit RA8875 [4], prikazana na Sliki 5.5, je gonilniška plošča zasnovana za povezovanje z različnimi zasloni LCD. Vsebuje relativno zmogljiv grafični krmilnik RA8875, ki lahko krmili zaslone z ločljivostjo do 800 x 480 slikovnih točk. Vsebuje 768KB spomina, ki služi kot medpomnilnik slike zaslona. Vmesnik za krmiljenje plošče je SPI. V krmilniku RA8875 je implementirano strojno pospešeno risanje grafičnih oblik, kot so črte, pravokotniki, trikotniki, elipse in pravokotniki. Vgrajen ima tudi evropski nabor pisav in strojno pospešeno do 4-kratno povečavo znakov.

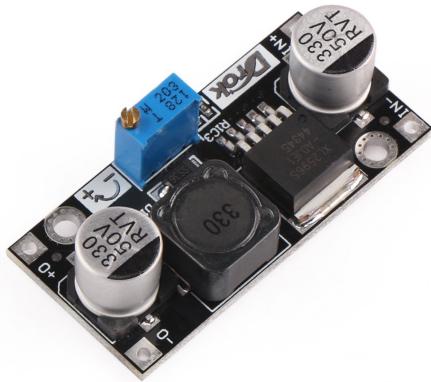
Na splošno je gonilniška plošča Adafruit RA8875 zadovoljiva rešitev za prikaz grafičnega uporabniškega vmesnika za prvo leto uporabe zaslona LCD, saj je implementacija dokaj enostavna. Pri tekmovanju Formula Student imamo za razvoj čas le do poletja, nato sledijo dirke. Dirkalnik v celoti temelji na principu iterativnega razvoja, vsako leto izdelamo nov dirkalnik, ki je (vsaj v teoriji) boljši od prejšnjega. Trenutna rešitev predstavlja preboj na področju uporabniškega vmesnika dirkalnika zaradi uporabe zaslona LCD. V prihodnjih letih nato pričakujem in že načrtujem izboljšave in dodatne funkcionalnosti.

5.3.1 Napajanje zaslona LCD

Kot sem že omenil, pri zaslolu LCD potrebujemo osvetlitev. Gonilniška plošča Adafruit RA8875 ima za osvetlitev zaslona vgrajen ojačevalnik s konstantnim tokom, ki lahko zagotovi toka 25 mA ali 50 mA pri napetosti do 24 V. To se ne sklada z zaslonom, ki sem ga izbral. Za napajanje osvetlitve zaslona Winstar WF50FSWAGDNN0 potrebujemo 40 mA in 30 V [9].

Rešitev je bila preprosta. Komponente za napajanje zaslona sem odstranil s plošče, nanjo pa prispejakal modul za regulacijo napetosti (Slika 5.6). Pri tej operaciji sem bil hvaležen za odprtakodnost plošče, saj sem imel na voljo vse shematike vezja plošče. Modul za regulacijo napetosti ima potenciometer, s katerim nastavimo želeno izhodno napetost. S pomočjo multimetra in laboratorijskega napajjalnika sem nastavil potenciometer tako, da je bila izhodna napetost 36 V. Sicer to ni najboljša rešitev, a deluje dovolj dobro. V

prihodnje bi bilo regulator za napajanje osvetlitve zaslona bolje implementirati kar na naši plošči tiskanega vezja.



Slika 5.6: Modul za regulacijo enosmerne napetosti. Z vgrajenim potenciometrom (modra komponenta) nastavimo želeno izhodno napetost.

5.4 Vhodne naprave

Vhodna naprava [16] je zunanja naprava, ki se uporablja za posredovanje podatkov ali krmilnih signalov računalniku ali drugi elektronski napravi. Vhodne naprave se uporabljajo za vnos vhoda v sistem, ki se nato lahko obdelava in uporabi za nadzor delovanja naprave ali izvajanje drugih nalog.

Uporabil sem dva tipa vhodnih naprav, in sicer tipke in rotacijske enkoderje.

5.4.1 Pritisna stikala

Kot tipke sem uporabil okrogla industrijska pritisna stikala. Stikala so vodooodporna - lahko so potopljena v globini 1 m za 60 minut [12]. To je

pomembno, saj morajo električni dirkalniki pred dinamičnimi dogodki prestatи t.i. dežni test (angl. *rain test*). Na dežnem testu dirkalnik škropijo 120 sekund. Test je opravljen, če med testom in 120 sekund po testu električni sistem formule še vedno deluje normalno.

Dva pritisna stikala sta vgrajena v armaturno ploščo. Namenjena sta vklopu visokonapetostnega sistema dirkalnika. Šest tipk je vgrajenih v volan, trenutno ima le ena od teh pomemben pomen. Namenjena je sprožanju radijske komunikacije voznika z ekipo. Komunikacija med voznikom in ekipo je ključnega pomena pri dirkanju, saj obema stranema omogoča, da sprejemata informirane odločitve, ki lahko vplivajo na izid dirke.

5.4.2 Diode LED

LED (angl. *Light Emitting Diode*) je elektronska komponenta, ki oddaja svetljobo, ko skozi njo teče električni tok. LED diode se uporabljajo v številnih aplikacijah, vključno z razsvetljavo, zasloni, indikatorji in senzorji.

Kot sem že omenil, je v skladu s pravilnikom tekmovanja na armaturni plošči potrebno implementirati 3 LED diode. Za vse sem uporabil isti tip diode, in sicer z napajalno napetostjo 12 V in svetilnostjo 100 mcd. Tako kot pritisna stikala so diode vodooodporne.

5.4.3 Rotacijski enkoderji

Rotacijski enkoder [17], prikazan na Sliki 5.7, je vhodna naprava, ki se uporablja za zaznavanje rotacijskega gibanja in položaja. Sestavljen je iz gredi, ki jo je mogoče vrteti v obe smeri, in mehanizma kodirnika, ki zaznava smer in količino vrtenja. Obstajata dve glavni vrsti rotacijskih enkoderjev: absolutni in inkrementalni. Absolutni rotacijski enkoderji podajo edinstveno izhodno kodo za vsak položaj osi, medtem ko inkrementalni rotacijski enkoderji podajo izhodne signale, ki kažejo smer in količino vrtenja od prejšnjega položaja. Na kratko povedano, z absolutnimi rotacijskimi enkoderji lahko določimo točen položaj osi, pri inkrementalnih pa le smer vrtenja.

Dva absolutna enkoderja sta vgrajena na volan. Levi je namenjen menjavi prikazov na zaslonu, desni pa menjavi prednastavitev vektoriranja navora. Vektoriranje navora je sistem v vozilih za nadzor količine moči, ki se prenaša na posamezno kolo. S tem lahko izboljšamo vodljivost, stabilnost in splošno zmogljivost vozila. Vektoriranje navora sicer še nismo izpilili do te mere, da bi ga uporabljali na dirkah. Uporabljeni enkoderji se lahko nahajata v desetih različnih stanjih.



Slika 5.7: Absolutni rotacijski enkoder

Poglavlje 6

Vgrajena programska oprema

6.1 Uporabljena razvojna orodja

6.1.1 Integrirano razvojno okolje STM32CubeIDE

STM32CubeIDE [11] je integrirano razvojno okolje (angl. *Integrated Development Environment, IDE*), zasnovano za razvoj aplikacij za mikrokontrolerje STM32. Razvilo ga je podjetje STMicroelectronics in temelji na okolju Eclipse.

STM32CubeIDE ponuja celoten nabor orodij za razvoj vgrajene programske opreme. Vključuje urejevalnik kode z označevanjem sintakse in zmogljiv razhroščevalnik s podporo za odpravljanje napak v realnem času, sledenje spremenljivkam in profiliranje kode.

V okolje STM32CubeIDE je integrirano okolje STM32CubeMX, grafično orodje za konfiguracijo mikrokontrolerjev STM32. STM32CubeMX omogoča generacijo inicializacijske kode glede na vnešene parametre, na primer inicializacijo nožic mikrokontrolerja, konfiguracijo notranjih ur, operacijskega sistema v realnem času.

6.1.2 Razhroščevalnik SEGGER J-Link

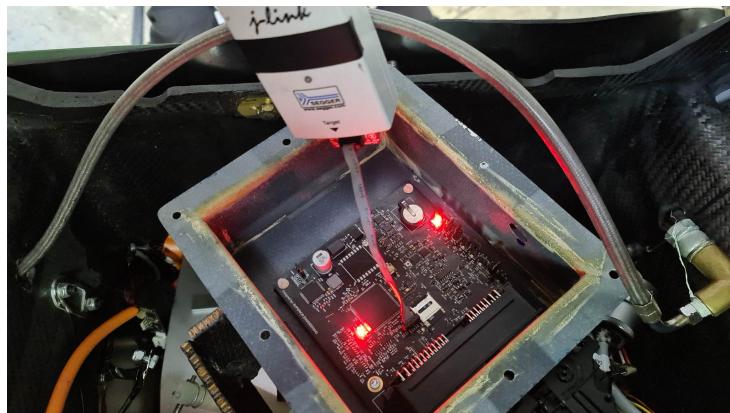
SEGGER J-Link [13] je razhroščevalnik za vgrajene sisteme. Razvilo ga je nemško podjetje SEGGER Microcontroller GmbH, specializirano za programska in strojna orodja za vgrajene sisteme. Razvijalcem nudi hiter in zanesljiv način za odpravljanje napak in programiranje strojne opreme. Podpira široko paletto mikrokrmilnikov različnih proizvajalcev, vključno z napravami arhitekture ARM in Renesas. Podpira tudi funkcije za razhroščevanje, kot je sledenje v realnem času, ki razvijalcem omogoča spremljanje izvajanja kode v realnem času.

Orodje za razhroščevanje, kot je SEGGER J-Link, je potrebno pri razvoju vgrajenih sistemov iz več razlogov:

- Razhroščevanje: vgrajeni sistemi pogosto delujejo na platformah strojne opreme z omejenimi zmožnostmi razhroščevanja, zaradi česar je težko diagnosticirati in odpraviti nepravilnosti v programski opremi. Razhroščevalnik, kot je J-Link, ponuja način za zaustavitev izvajanja kode na kateri koli točki, pregled stanja sistema in korakanje skozi kodo, kar omogoča lažje opravljanje težav.
- Programiranje: orodje, kot je J-Link, se uporablja tudi za programiranje in posodabljanje vgrajene programske opreme v vgrajenih sistemih.
- Testiranje: J-Link omogoča testiranje in validacijo vgrajene programske opreme.

6.1.3 Programski jezik C

Za razvoj vgrajene programske opreme sem uporabil programski jezik C. C je strukturiran, proceduralni visokonivojski programski jezik, ki se uporablja za sistemsko programiranje, razvoj vgrajenih sistemov in programiranje aplikacij. V začetku sedemdesetih let prejšnjega stoletja ga je razvil Dennis Ritchie, ki je bil tudi eden ključnih razvijalcev operacijskega sistema UNIX.



Slika 6.1: Razhroščevalnik SEGGER J-Link, povezan na napravo VCU (*Vehicle Control Unit*)

C je prevajan jezik, kar pomeni, da mora izvorno kodo pred izvajanjem prevajalnik prevesti v strojno kodo. Alternativa so skriptni programski jeziki, kot sta Python in Javascript, ki se v strojno kodo tolmačijo med samim izvajanjem. C in prevajani jeziki so hitrejši in učinkovitejši od skriptnih.

Programski jezik C nudi neposreden dostop do strojne opreme, kot so pomnilnik in periferne naprave. Ta neposredni dostop razvijalcem omogoča pisanje učinkovite in optimizirane kode, ki lahko neposredno komunicira s strojno opremo. Če jeziku prištejemo še dobro prenosljivost in učinkovitost, ga lahko upoštevamo kot resnega kandidata za sistemsko programiranje, kot je programiranje operacijskih sistemov, gonilnikov naprav in ostalih nizkonivojskih aplikacij, na primer vgrajene programske opreme.

6.1.4 Gonilniški sloj HAL

STM32 HAL [10] (*Hardware Abstraction Layer*) je nabor gonilnikov in knjižnic, ki jih nudi STMicroelectronics za svoje mikrokontrolnice STM32. HAL zagotavlja visokonivojski vmesnik za dostop do strojne opreme, kar razvijalcem omogoča pisanje kode aplikacije, ne da bi skrbeli za podrobnosti strojne opreme na nizki ravni. V praksi to pomeni, da nisem potreboval pisati po-

sameznih bitov v registre, temveč le poklical primerno HAL funkcijo, ki je poskrbela za konfiguracijo strojne opreme.

Gonilniki HAL predstavljajo API (*Application Programming Interface*) za dostop do različnih perifernih naprav strojne opreme mikrokontrolerja STM32, vključno z GPIO, SPI, UART, ADC, časovniki in številnimi drugimi. HAL zagotavlja standardiziran programski vmesnik, ki ga je mogoče uporabiti v različnih mikrokontrolerjih STM32. HAL abstrahira podrobnosti strojne opreme, kar omogoča prenos kode na različne mikrokontrolerje STM32 ali spremenjanje konfiguracije strojne opreme brez spremenjanja izvirne kode aplikacije.

STM32 HAL je dobro dokumentiran in podprt s strani STMicroelectronics, z obsežno dokumentacijo in primeri, ki so na voljo na njihovi spletni strani.

6.2 Razvoj vgrajene programske opreme

6.2.1 Inkrementalni razvoj programske opreme

Pri razvoju vgrajene programske opreme sem uporabljal metodo inkrementalnega razvoja. Inkrementalni razvoj [15] je pristop, pri katerem se programska oprema gradi v majhnih, postopnih korakih ali stopnjah. Razvojni proces je razdeljen na vrsto majhnih, obvladljivih delov, pri čemer vsak del predstavlja del končnega programskega sistema.

Inkrementalni razvojni proces običajno vključuje gradnjo osnovne različice programskega sistema in nato dodajanje dodatnih funkcionalnosti v naslednjih iteracijah. Vsaka ponovitev se testira in ovrednoti, preden se preide na naslednji korak, s čimer se poskuša zagotoviti, da programska oprema na vsaki stopnji deluje, kot je predvideno.

V praksi to pomeni, da sem na začetku generiral inicializacijsko kodo na obeh mikrokontrolnikih, nato vzpostavil SPI povezavo med njima in postopoma dodajal še ostale funkcionalnosti. Pri grafičnem uporabniškem vmesniku sem naprej naredil gonilnik s funkcijami za izris likov in besedila, nato pa z njimi

sestavljal vse večje gradnike, dokler nisem sestavil celotnega zaslona. Med posameznimi koraki sem novosti v programu kar se da dobro testiral.

6.2.2 Generiranje inicializacijske kode

Po ustvarjanju svežega projekta za izbrana mikrokrmlnika, sem uporabil okolje za generiranje inicializacijske kode STM32CubeMX. Izbral sem periferijo, ki jo bom uporabljal (SPI, CAN, GPIO) in z grafičnim vmesnikom nastavil potrebne parametre. V zavihku za konfiguracijo ure sem nastavil frekvenco zunanje ure na tisto od zunanjega oscilatorja, to je 24 MHz. Nato sem z grafičnim vmesnikom nastavil vse potrebne delilnike in množilnike ure tako, da je bila sistemski ura najvišja možna za določen mikrokrmlnik (72 MHz za STM32F303 in 400 MHz za STM32H743). Na koncu sem pritisnil gumb za generiranje kode. Po generaciji sem v izvorni kodi dobil že pripravljeno vso potrebno inicializacijo. Program sem lahko zgradil in ga testno naložil na mikrokrmlnik.

6.2.3 Volan

Vgrajena programska oprema mikrokrmlnika STM32F3 v volanu je precej enostavna in poteka znotraj neskončne zanke While True. Znotraj zanke preberem vse vhode na volanu (2 rotacijska enkoderja in 6 gumbov) ter prebrane vrednosti pošljem Armaturni plošči prek protokola SPI. Stanja gumbov zase-dejo vsako po en bit, stanja enkoderjev pa vsako po 4 bite, skupaj bom torej potreboval prostor dolžine 2 Byte. Vrednosti shramim v tabelo in pošljem, kot je razvidno iz Izseka 6.1.

Izsek 6.1: Izvorna koda za sprejem podatkov s povezave SPI

```
uint8_t spiData[] = {  
    readBttns,  
    readRot  
};  
  
/* Preberi dolzino 2 Byte in podatke zapisi v strukturo spiData */
```

```
HAL_SPI_Transmit(&hspi2, spiData, 2, 0xff);
```

6.2.4 Armaturna plošča

Vgrajena programska oprema mikrokrmlnika STM32H7 na Armaturni plošči je kompleksnejša. Opravlja več nalog hkrati, zato sem uporabil operacijski sistem v realnem času in ustvaril več opravil, ki se izvajajo vzporedno.

Operacijski sistem v realnem času FreeRTOS

FreeRTOS [7] je priljubljen in odprtakodni operacijski sistem v realnem času (angl. *Real Time Operating System, RTOS*). FreeRTOS je tudi zelo prenosljiv, s podporo za široko paleto mikrokontrolerjev in procesorjev ter več razvojnih okolij in orodij. Prav tako je dobro dokumentiran, z obsežno dokumentacijo in primeri, ki so na voljo razvijalcem za pomoč pri začetku uporabe operacijskega sistema.

Ključne funkcije FreeRTOS:

1. Večopravilnost: sočasno izvajanje opravil s prioriteto. Opravila lahko upravljamo s funkcijami, kot so ustvarjanje, brisanje, začasna prekinitve in nadaljevanje.
2. Upravljanje pomnilnika: dinamično dodeljevanje pomnilnika, kar omogoča opravilom, da si po potrebi dodelijo pomnilnik in ga sprostijo, ko ga ne potrebujejo več.
3. Mehanizmi za sinhronizacijo in komunikacijo: semaforji, ključavnice, vrste.
4. Obravnavanje prekinitve.

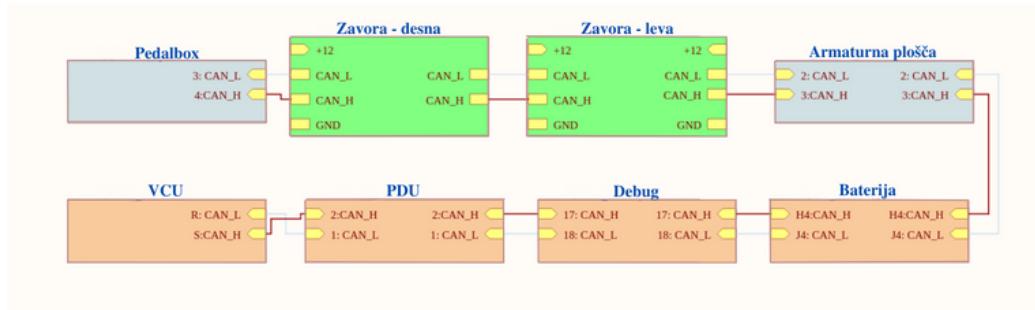
Opravila v RTOS

V operacijskem sistemu v realnem času so opravila neodvisne izvršljive enote, ki se izvajajo sočasno z drugimi opravili v sistemu. Opravilo je programska

funkcija, ki izvaja določeno nalog ali nabor nalog znotraj aplikacije. Sam sem v programu definiral naslednja opravila:

- **Opravilo za pisanje na vodilo CAN**

Vodilo CAN (*Controller Area Network*) je robustno interno komunikacijsko omrežje, ki povezuje komponente znotraj vozila (npr. avtomobila, avtobusa, vlaka, industrijskega ali kmetijskega vozila, ladje ali letala). V elektroniki je vodilo preprosto naprava, ki povezuje več elektronskih naprav skupaj. CAN je protokol, ki temelji na sporočilih. Če prenaša več kot ena naprava hkrati, lahko naprava z najvišjo prednostjo nadaljuje, medtem ko se druge umaknejo. Sporočila sprejemajo vse naprave, vključno z oddajno napravo.



Slika 6.2: Naprave na vodilu CAN

Na našem dirkalniku je na vodilu več naprav, vse pa vodilo CAN primarno uporabljajo za komunikacijo z enoto VCU. Slika 6.2 prikazuje naprave v dirkalniku priključene na vodilo CAN. Na Armaturni plošči sem moral implementirati pošiljanje stanja gumbov na armaturni plošči, ki se uporablja v proceduri za vklop visokonapetostnega sistema avtomobila. V Izseku 6.2 je izvorna koda funkcije, ki implementira to funkcionalnost.

Izsek 6.2: Koda za pošiljanje paketa po vodilu CAN

```
HAL_StatusTypeDef CAN_Write_Transmit()
```

```
{
    can_dashboard_t CanData;
    CanData.s.buttons = DashInput_Buttons_Read();
    //CanData.s.angle = getAngle(); /* Rotacijski enkoder za merjenje
        kota volana na tem dirkalniku ni bil implementiran */

    /* Shrani vrednosti v globalno spremenljivko za izpis na LCD */
    CAN_Write_Dashboard.buttons = CanData.s.buttons;
    CAN_Write_Dashboard.angle = CanData.s.angle;

    /* Poslji po CAN */
    return HAL_FDCAN_AddMessageToTxFifoQ(&hfdcan2, &TxHeader, CanData
        .u8);
}
```

- Opravilo za branje vodila CAN

V tem opravilu berem vse pakete na vodilu CAN poslane s strani bilo-katere naprave. Prebrane vrednosti shramim v globalne spremenljivke, da jih nato lahko izpišem na LCD v Opravilu za izris slike na LCD. Izsek 6.3 prikazuje izvorno kodo, ki se izvrši ob sprejemu paketa z vodila CAN. Glede na identifikator sprejetega paketa ločim v katere globalne spremenljivke bom shranil vsebino paketa.

Izsek 6.3: Izvorna koda za branje paketa z vodila CAN

```
// izlusci sporocilo in modificiraj globalno spremenljivke
uint16_t SenderID = RxHeader.Identifier;
switch (SenderID)
{
    case CAN_ID_SOBJMS: /* Stanje BMS */
    {
        can_SOBJMS_t SOBJMS_data;
        memcpy(SOBJMS_data.u8, RxData, sizeof(SOBJMS_t));

        /* Shrani vrednosti paketa */
        CAN_Read_SOBJMS.BMS_AIR_state = SOBJMS_data.s.BMS_AIR_state;
    }
}
```

```

        break;
}

case CAN_ID_LV: /* Status nizkonapetostnega sistema */
{
    can_vcu_data_t VCU_data_data;
    memcpy(VCU_data_data.u8, RxData, sizeof(vcu_data_t));

    /* Shrani vrednosti paketa */
    CAN_Read_VCU_Data.lv_current = VCU_data_data.s.lv_current;
    CAN_Read_VCU_Data.lv_voltage = VCU_data_data.s.lv_voltage;
    CAN_Read_VCU_Data.error_leds = VCU_data_data.s.error_leds;
    CAN_Read_VCU_Data.sdc_status = VCU_data_data.s.sdc_status;

    // Izlusci in nastavi stanje IMD ERROR na izhodno nozico
    HAL_GPIO_WritePin(dash___imd___OUT_GPIO_Port,
                       dash___imd___OUT_Pin, (CAN_Read_VCU_Data.error_leds & (1 << 0)) );
    // Izlusci in nastavi stanje AMS ERROR na izhodno nozico
    HAL_GPIO_WritePin(dash___ams___OUT_GPIO_Port,
                       dash___ams___OUT_Pin, (CAN_Read_VCU_Data.error_leds & (1 << 1)) );
    // Izlusci in nastavi stanje TS OFF na izhodno nozico
    HAL_GPIO_WritePin(dash___ts___OUT_GPIO_Port, dash___ts___OUT_Pin,
                      (CAN_Read_VCU_Data.error_leds & (1 << 2)));
    // Izlusci in nastavi stanje BSPD ERROR na izhodno nozico
    HAL_GPIO_WritePin(dash___bspd___OUT_GPIO_Port,
                      dash___bspd___OUT_Pin, (CAN_Read_VCU_Data.error_leds & (1 << 3)));

    break;
}
/* case CAN_ID_xy: */
/* ... */
/* In tako naprej za druge identifikatorje paketov */

```

- Opravilo za branje povezave SPI z Volanom

Podobno kot v slednjem opravilu v tem opravilu berem povezavo SPI z

Volanom in prebrane vrednosti tipk ter rotacijskih enkoderjev shranim v globalni spremenljivki, do katerih lahko dostopa Opravilo za izris slike.

- **Opravilo za izris slike na LCD**

V tem opravilu krmilim gonilniško ploščo Adafruit RA8875 in s tem rišem grafike na LCD. Implemeniral sem več zaslonov, med katerimi se uporabnik premika s rotacijskim enkoderjem na volanu:

1. Zaslon za razhroščevanje

Na zaslonu Vsi paketi prebrani z vodila CAN so prikazani. Ostali člani ekipe, ki delajo na strojni in programski opremi dirkalnika lahko uporabijo ta zaslon za namene razhroščevanja. Pred implementacijo tega zaslona je bilo za branje sporočil na vodilu CAN potrebno na vodilo priklopiti računalnik in uporabiti temu namejeno programsko opremo.

Omenjen zaslon je prikazan na Sliki 6.3.



Slika 6.3: Zaslon za razhroščevanje. Nekatere naprave v času zajema posnetka še niso bile priklopljene na vodilo.

2. Zaslon za dirkanje

Na Sliki 6.4 je zaslon prikazan med vožnjo. Ta zaslon prikazuje najpomembnejše informacije za voznika. Srednji stolpec prikazuje stanje napolnjenosti baterije SoC, stranska pa temperaturi baterije in pogonskega sklopa (upošteva se najvišja izmed temperatur inverterjev in motorjev). V Izseku 6.4 je izvorna koda funkcije za izris stolpca na zaslon.

Izsek 6.4: Funkcija za izris stolpca na zaslon LCD

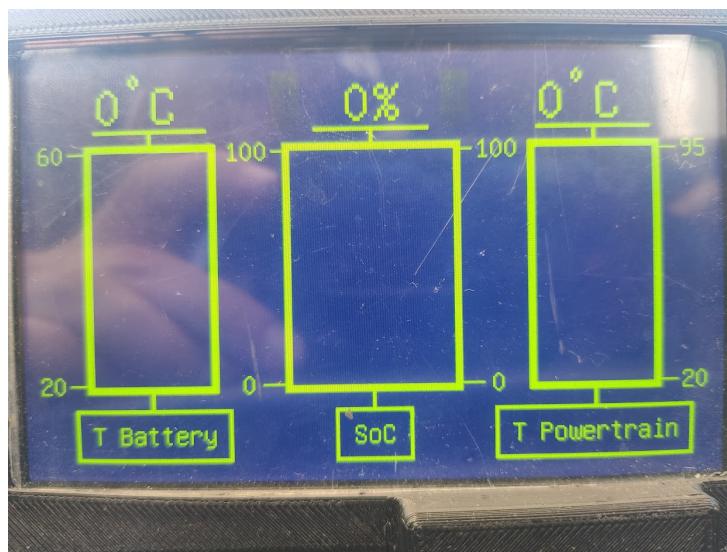
```
void LCD_Screens_Drive_DrawBar(uint16_t x, uint16_t y, uint16_t
w, uint16_t h, uint16_t wBorder,
                                uint8_t filledPrc, uint16_t color, uint16_t
                                bgColor, uint8_t direction) {
    /* Izris roba stolpca */
    Adafruit_RA8875_FillRect(x, y, w, h, color);
```

```

/* Izracun parametrov notranjega dela stolpca */
uint16_t x1 = x + wBorder; /* Pozicija x */
uint16_t w1 = w - 2 * wBorder; /* Sirina */
uint16_t h1 = ((h - 2 * wBorder) * ((filledPrc-100) * -1)) /
    100; /* Visina */
uint16_t y1 = y + wBorder; /* Pozicija y */

if (h1 != 0) {
    /* Izris notranjega dela stolpca */
    Adafruit_RA8875_FillRect(x1, y1, w1, h1, bgColor);
}
}
}

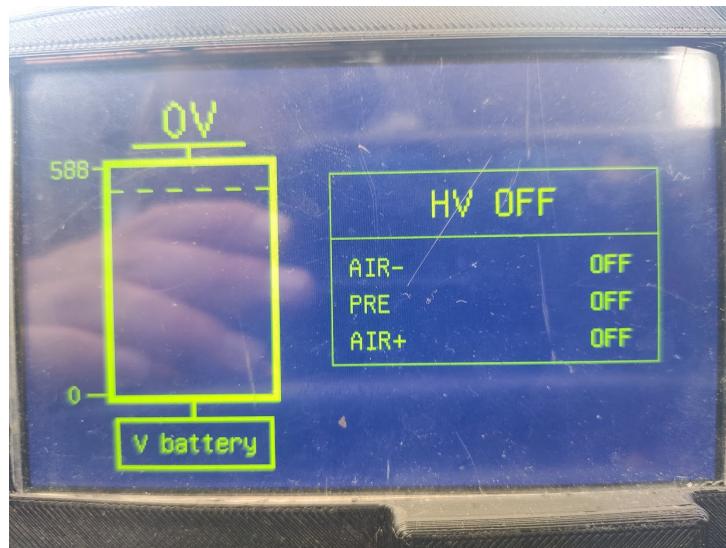
```



Slika 6.4: Zaslon za dirkanje

3. Zaslon za predpolnjenje baterije

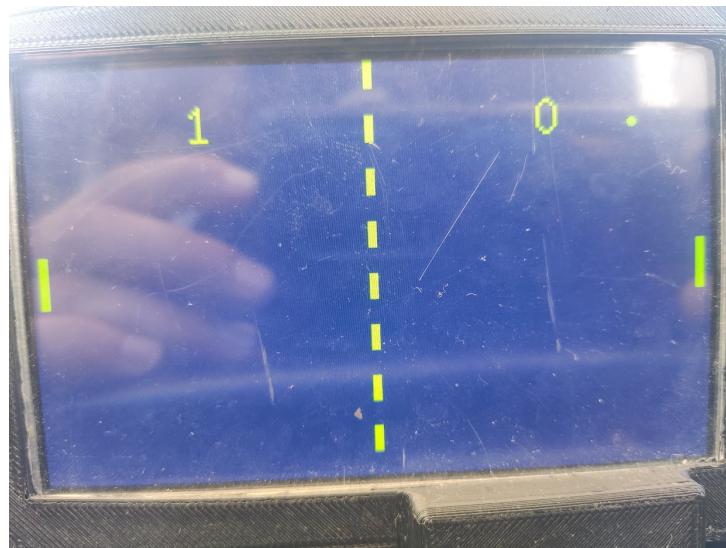
Na Sliki 6.5 je zaslon, ki ga študentje elektrotehnike uporabljajo za opazovanje predpolnjenja baterije. Prikazana so tudi stanja kontaktorjev.



Slika 6.5: Zaslon za predpolnjenje baterije

4. Igra Pong

Zaslon na Sliki 6.6 je namenjen igranju igre, ki je klon priljubljene igre Pong [2], ki jo je razvilo podjetje Atari leta 1972.



Slika 6.6: Zaslon za zabavo

Poglavlje 7

Zaključek

Z izdelkom, ki sem ga opisal v tej nalogi, sem prenovil sistem uporabniškega vmesnika na dirkalniku ljubljanske ekipe tekmovanja Formula Student. Mislim, da sem zadevo uspešno izpeljal od same ideje do delajočega izdelka, ki ustreza prvotnim zahtevam. Ta projekt je bil moj prvi večji projekt in prvič sem sodeloval s tako veliko ekipo.

Vsekakor rezultati ne bi bili tako uspešni brez dobre ekipe. Še posebej sem hvaležen študentom višjih letnikov Fakultete za elektrotehniko, ki so že imeli izkušnje s tekmovanja. Posredovali so mi veliko znanja z njihovega področja, ki je še kako potrebno pri razvoju vgrajenih sistemov, jaz pa sem jim pomagal pri pisanju kode ali razhroščevanju. Ustvarila se je nekakšna simbioza, kjer sta se srečali dve sorodni inženirski panogi, in srečen sem, da sem bil del tega.

Z mojim izdelkom so zadovoljni naši vozniki, oziroma glavni končni uporabniki, kar je najpomembnejše. Zadovoljni so tudi študentje, ki razvijajo električni sistem dirkalnika, saj sem jim omogočil enostaven vpogled v vsebino na vodilu CAN, kar jim omogoča enostavno razhroščevanje. Sodniki na tekmovanjih niso imeli pritožb nad uporabniškim vmesnikom, niti posebnih komentarjev, kar si razlagam tako, da je sistem dovolj dober, a nič posebnega. Mislim, da sem za zdaj postavil dober temelj, na katerem se bo delalo še prihodnja leta.

Na splošno sem hvaležen za izkušnje in priložnosti, ki mi jih je prineslo sodelovanje pri ekipi Superior Engineering. Veselim se nove sezone tekmovanja Formule Student in drugih novih izzivov v življenju.

Članki v revijah

- [6] Joseph LaDou. "Printed circuit board industry". V: *International Journal of Hygiene and Environmental Health* 209.3 (2006), str. 211–219. ISSN: 1438-4639. DOI: <https://doi.org/10.1016/j.ijheh.2006.02.001>. URL: <https://www.sciencedirect.com/science/article/pii/S1438463906000204>.

Celotna literatura

- [1] Manan Gupta. *Automotive UX Superior UX approaches by leading car manufacturers in 2022*. URL: <https://uxplanet.org/automotive-ux-superior-ux-approaches-by-leading-car-manufacturers-in-2022-d371082f61bc> (pridobljeno 20. 12. 2022).
- [2] Atari Interactive Inc. *Pong Game*. URL: <https://uxplanet.org/automotive-ux-superior-ux-approaches-by-leading-car-manufacturers-in-2022-d371082f61bc/> (pridobljeno 4. 3. 2023).
- [3] Adafruit Industries. *About Adafruit*. URL: <https://www.adafruit.com/about> (pridobljeno 2. 3. 2023).
- [4] Adafruit Industries. *RA8875 Driver Board for 40-pin TFT Touch Displays - 800x480 Max*. URL: <https://www.adafruit.com/product/1590> (pridobljeno 28. 2. 2023).
- [5] Jonathon Klein. *F1 Steering Wheels: Every Button, Paddle, and Knob Explained*. URL: <https://www.thedrive.com/accelerator/40976/how-does-an-f1-steering-wheel-work> (pridobljeno 20. 12. 2022).
- [6] Joseph LaDou. “Printed circuit board industry”. V: *International Journal of Hygiene and Environmental Health* 209.3 (2006), str. 211–219. ISSN: 1438-4639. DOI: <https://doi.org/10.1016/j.ijheh.2006.02.001>. URL: <https://www.sciencedirect.com/science/article/pii/S1438463906000204>.
- [7] Real Time Engineers Ltd. *The FreeRTOS™ Kernel*. URL: <https://www.freertos.org/RTOS.html> (pridobljeno 4. 3. 2023).

- [8] Winstar Display Co. Ltd. *Sunlight Readable 5 inch IPS TFT LCD Panel 800x480*. URL: <https://www.winstar.com.tw/products/tft-lcd/ips-tft/ips-5.html> (pridobljeno 3. 3. 2023).
- [9] Winstar Display Co. Ltd. *TFT Display Specification*. Rev. C. Jul. 2021.
- [10] STMicroelectronics. *Description of STM32H7 HAL and low-layer drivers*. UM2217. Rev. 6. Dec. 2022.
- [11] STMicroelectronics. *Integrated development environment for STM32 products*. DB3871. Rev. 6. Nov. 2021.
- [12] IP68 Sub-Miniature Pushbutton Switches. *TFT Display Specification*. 0-2329328-1. Mar. 2018.
- [13] Segger Microcontroller Systems. *J-Link BASE — The JTAG/SWD debug probe with USB interface*. URL: <https://www.segger.com/products/debug-probes/j-link/models/j-link-base/> (pridobljeno 4. 3. 2023).
- [14] Wikipedia contributors. *Formula SAE — Wikipedia, The Free Encyclopedia*. 2022. URL: https://en.wikipedia.org/wiki/Formula_SAE (pridobljeno 22. 12. 2022).
- [15] Wikipedia contributors. *Incremental build model — Wikipedia, The Free Encyclopedia*. 2023. URL: https://en.wikipedia.org/wiki/Incremental_build_model.
- [16] Wikipedia contributors. *Input device — Wikipedia, The Free Encyclopedia*. 2023. URL: https://en.wikipedia.org/wiki/Input_device.
- [17] Wikipedia contributors. *Rotary encoder — Wikipedia, The Free Encyclopedia*. 2023. URL: https://en.wikipedia.org/wiki/Rotary_encoder.
- [18] Wikipedia contributors. *Zaslon LCD — Wikipedia, The Free Encyclopedia*. 2023. URL: https://sl.wikipedia.org/wiki/Zaslon_LCD.