»Mladi za napredek Maribora 2016«
33. srečanje

KRMILJENJE AVTOCESTNEGA PREDORA

Raziskovalno področje: ELEKTROTEHNIKA, ELEKTRONIKA
inovacijski predlog

PROSTOR ZA NALEPKO

Avtor: GREGOR STAVBAR, DOMINIK GRIL

Mentor: MILAN IVIČ

Šola: SREDNJA ELEKTRO-RAČUNALNIŠKA ŠOLA

2016, Maribor

1. KAZALO

1.1. KAZALO SLIK:

1.2. KAZALO GRAFOV

2. ZAHVALA

Rada bi se zahvalila najinemu mentorju za podporo, pomoč in različne nasvete pri izdelavi najinega inovacijskega predloga ter vsem ostalim, ki so na kakršenkoli način pomagali pri izdelavi inovacijskega predloga.

3. POVZETEK

V najinem inovacijskem predlogu sva si zadala cilj, da raziščeva kako deluje vodenje nekega avtocestnega predora in da predlagava nekaj najinih izboljšav. Za lažjo obdelavo in prikaz delovanja sva se odločila, da bova naredila maketo predora iz lesa. Potrebne naprave bova vodila oz. nadzirala z razvojno ploščo Arduino. Poiskala sva več idej in se nato odločila, da bova predor nadzirala s senzorji za plin, ogenj in vlažnost. Arduino vse skupaj nadzoruje in ob nepravilnosti javi napako. Glede na napako se potem zapornica zapre, vključi ventilacija itd. Za javljanje napak so izven tunela nameščeni ustrezni znaki in grafični LCD, ki prikazuje hitrost vožnje in druga pomembna opozorila. Uporabila sva tudi programsko okolje LabVIEW, s katerim lahko preko računalnika nadzorujeva in spremljava dogajanje v predoru ter nadzirava vse potrebne meritve.

4. UVOD

V najinem inovacijskem predlogu sva se zamislila, kako delujejo avtomobilski predori, kako razvita je že tehnologija v današnjih predorih itd. Osredotočila sva se tudi na stvari, ki jih predori še nimajo in s katerimi bi lahko pripomogla k izboljšanju.

Dandanes so avtomobilski predori opremljeni z najnovejšimi tehnologijami, ki omogočajo varno vožnjo in preprečujejo morebitne zaplete v predorih. Uporabljajo se mnogi senzorji, s pomočjo katerih imamo vpogled v dogajanje v predoru, tako da lahko v trenutku odreagiramo v primeru kakršnih koli napak. Vsi predori imajo primerno osvetlitev znotraj kot tudi zunaj predora. Prav tako sva v najinem inovacijskem predlogu uporabila vse senzorje in upoštevala vse faktorje, ki so potrebni pri avtomobilskem predoru. Poskrbela sva za osvetlitev z LED diodami itd.

Sama sva dodala inovacijo, ki še je nisva opazila in bi bila zelo koristna pri predorih. Gre za zapornico, ki bi preprečevala vstop v predor voznikom, ki vozijo v napačno smer. Tako bi preprečili morebitne nesreče. Dodala sva še sistem za prilagoditev svetlobe, tako da če voznik dlje časa vozi v predoru in potem pride iz njega, zanj ni prevelike spremembe svetlobe. To sva odpravila s pametno razsvetljavo zunaj in znotraj predora.

5. HIPOTEZE

1) Povišanje varnosti v predoru z zapornicami.

2) Znižanje stroškov razsvetljave z LED sijalkami.

3) Učinkovitejše prezračevanje predora.

4) Uporaba table s spremenljivo vsebino.

6. VSEBINSKI DEL

6.1. Kaj je avtomobilski predor ?

Predor je cevast ali pravokotni prostor pod zemljo, urejen za železniški ali cestni promet. Predori se uporabljajo se za skrajšanje poti. Namesto da je pot speljana okoli določene ovire, se naredi predor skozi oviro. Po navadi je ovira gora. Če želimo uporabiti na nekem odseku predor, potem moramo upoštevati geološke pogoje.

Slika 1 : Primera predorov (Vir:Avtor naloge)

6.2. Zgodovina predorov

Predore so poznali že v davni preteklosti. Najstarejši vodni predor naj bi bil zgrajen v Turčiji imenovan Trelek Kaya in naj bi bil zgrajen 2000 let pred našim štetjem. Bil je del vodnega sistema in je z vodo oskrboval prebivalce v naseljih, služil pa je tudi za namakanje. Tudi Rimljani so gradili predore, znan predor je na cesti Via Flaminia, ki so ga zgradili leta 76-77. Cesta SS 3 Flaminia še danes poteka skozi ta predor.

Gradnjo daljših predorov je omogočila šele iznajdba razstreliva. Moderna doba gradnje predorov se je začela z izgradnjo ceste in 60 m dolgega predora na prelazu St. Gotthard v Švici leta 1707. Prvi železniški predor je bil leta 1829 zgrajen v Liverpoolu (graditelj George Stephenson), inženir Isambard Brunel pa je v letih 1825-1841 zgradil prvi podvodni predor Sapperton Canal Tunnel v Londonu pod reko Temzo.

6.3. Moderni predori

Najdaljši cestni predor na svetu Laerdal se nahaja na Norveškem. Občudovanja vreden tehniški presežek, dolg kar 24,5 kilometra je speljan skozi živo skalo. Med vožnjo skozi predor se nahajamo pod 1000 metri skalovja. Za prezračevanje je poskrbljeno z 6,5 kilometrov dolgim prezračevalnim jaškom. Sveži zrak prihaja skozi obe strani predora onesnaženi pa odhaja skozi prezračevalni jašek.

Najdaljši slovenski cestni predor je predor Karavanke s 7864 metri oziroma 8019 metrov s portali (dolžina slovenskega dela predora je 3450 metrov), najdaljši dvocevni avtocestni predor pa trojanski z 2931 metri.

7. SESTAVNI DELI

7.1. Razvojna plošča Arduino

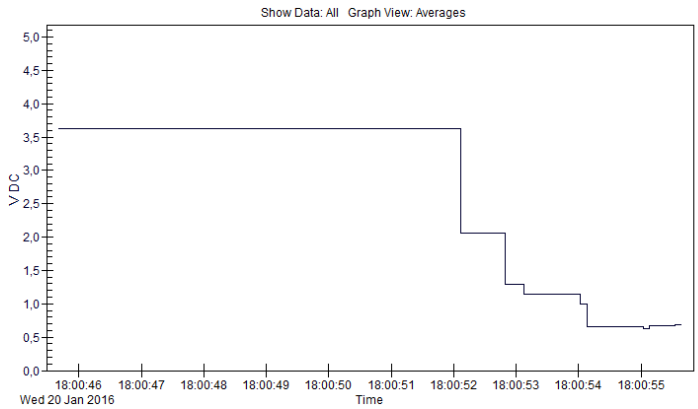
Za naš inovacijski projekt se nama je zdela najbolj primerna razvojna plošča Arduino Mega 2560, saj ima dovolj vhodov in izhodov. Razvojna plošča temelji na mikrokontrolerju Atmel 2560, ki ima 54 digitalnih vhodov in izhodov ter 16 analognih vhodov. Razvojna plošča ima tudi svoje napajanje 5 V, kar zadostuje vsem našim senzorjem in ni potrebnega dodatnega napetostnega napajanja. Ploščo enostavno preko USB vhoda povežemo z računalnikom. Programiramo jo v programskem okolju Arduino, ki temelji na programskem jeziku C++. Programiranje ni tako zahtevno, kot če bi programirali mikrokontrolerje družine PIC.

Slika 2: Arduino Mega 2560 (Vir: Avtor naloge)

7.2. Senzor za vlago

Senzor za vlago oz. senzor za dež sva uporabila za zaznavanje mokrega cestišča. Če senzor zazna vlažnost, potem se dovoljena hitrost skozi predor zmanjša. Senzor je vključen kot analogni senzor, na izhodu spreminja svojo napetost od 0 V do 5 V. Ta razpon vrednosti napetosti program (A/D pretvornik) pretvori na bitno vrednost od 0 do 1023, saj je A/D pretvornik 10-biten. S pomočjo vmesnika programskega okolja LabVIEW lahko na podlagi podatkov operater predora sproti nastavlja nivo, nad katerim se vklopi ustrezno opozorilo.

Slika 3: Senzor vlage (Vir: Avtor naloge)

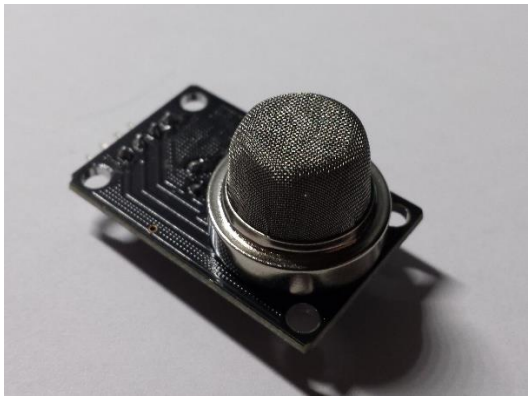


Graf 1: Odziv analognega izhoda na vlago (Vir: Avtor naloge)

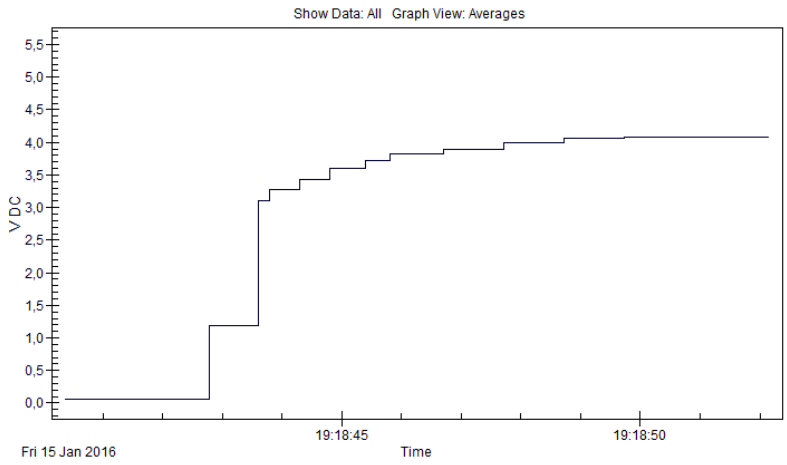
7.3. Senzor za plin

Senzor za plin sva uporabila zato, da lahko preverjava količino plinov v predoru. Sistem je razdeljen na 3 dele oz. stopnje, ki glede na različno koncentracijo plina v predoru, različno vključuje prezračevalne ventilatorje. Ob morebitnem vklopu opozorila za ogenj se izklopijo vsi ventilatorji. Tudi ta senzor je analogen, torej na izhodu v odvisnosti od koncentracije plinov spreminja napetost od 0 V do 5 V. PWM modulacija poskrbi za različne stopnje ventilacije. V najinem primeru so uporabljene tri stopnje, za vsako stopnjo posebej lahko operater sproti nastavlja kdaj se bo izvršila. Uporabila sva senzor MQ-2, kateri zaznava več vrst plina: metan, butan, propan, alkohol, vodik in dim. Ta senzor sva uporabila predvsem zaradi enostavnejše simulacije. Senzor ima tudi grelno tuljavico, kar pripomore k hitrejšem zaznavanju različnih plinov.

.



Slika 4: Senzor plina MQ-2 (Vir: Avtor naloge)

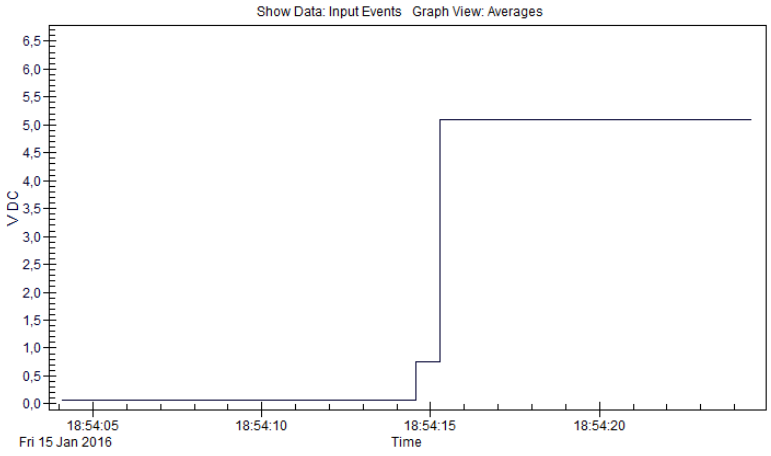


Graf 2: Odziv analognega izhoda senzorja na plin (Vir: Avtor naloge)

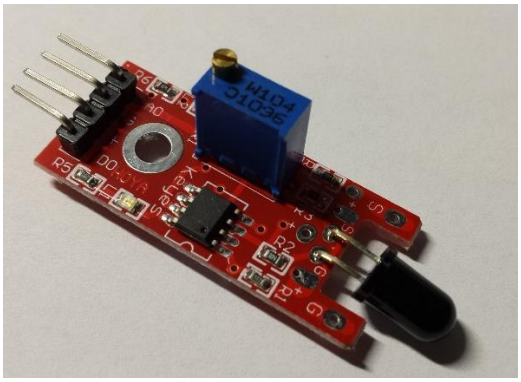
Kot je razvidno iz Grafa 2, senzor na začetku zelo burno reagira na pojav plina, nato pa vrednost napetosti na izhodu senzorja počasi narašča. Ta lastnost nama ja povzročala tudi manjše preglavice z nastavljanjem posameznih stopenj ventilacije v programu, vendar sva s praktičnim preizkušanjem našla idealne vrednosti.

7.4. Senzor za ogenj

Senzor za ogenj sva uporabila za zaznavanje ognja v predoru. Ob nastanku ognja v predoru se sproži najvišji nivo programa in se promet v predoru ustavi. Na grafičnem LCD-ju se prikaže rdeči križ in na manjšem LCD-ju se izpiše da je predor zaprt. Ta senzor je digitalen, za svoje delovanje uporablja IR led diodo, ki zaznava samo določen svetlobni spekter v katerem se nahaja svetloba ognja, zato ga ostala svetloba ne moti. Ko se na izhodu pojavi napetost, to krmilnik zazna in takoj pošlje ukaz za izklop ventilacije brez operaterjevega dovoljenja, saj bi ventilacija razmere samo še poslabšala. Tudi predor se zapre. Spust zapornic pa mora odobriti operater le takrat, ko je prepričan da je to varno.



Graf 3: Odziv digitalnega izhoda senzorja na ogenj (Vir: Avtor naloge)



Slika 5: Senzor ognja (Vir: Avtor naloge)

7.5. Grafični LCD

Grafični LCD sva uporabila kot tablo s spremenljivo vsebino. Na njej se prikazujejo različni prometni znaki kot so zaprtje predora, znaki za nevarnost pa tudi različne omejitve. Pozimi, ko je cestišče spolzko, se zniža dovoljena hitrost skozi predor, prikazuje pa se tudi znak za nevarnost. Grafični LCD uporablja gonilnik ILI9325C, ki je kompatibilen z Arduino razvojnimi ploščami, v najinem primeru z Arduino Mega 2560, ki ima dovolj pomnilnika za hrambo slik. LCD ima diagonalo 7,1 cm in ločljivost 240 x 320 slikovnih točk, kar je bilo ravno dovolj. Program sva napisala v Arduino programskem okolju, da sva pa lahko na displeju prikazovala slike, sva jih morala najprej pretvoriti iz jpg formata v C-kodo, ki sva jo nato vnesla v program. Pri tem sva bila omejena pri količini in velikosti slik, saj imajo Arduino razvojne plošče malo pomnilnika (Arduino mega 2560 ima 256 bitov).

Slika 6: Grafičen LCD (Vir: Avtor naloge)

8. IZDELAVA MAKETE IN PROGRAMSKA KODA

8.1. Maketa

Maketo sva izdelala iz lesa, ker se ga da najlažje oblikovati. Sestavljena je iz glavne plošče na katero sva postavila predor. Velikost predora sva izbrala glede na velikost komponent. Na osnovno ploščo sva narisala cestišče. S svinčnikom sva zarisala, kje se bodo nahajali senzorji, nato pa sva zvrtala luknje in namestila vse potrebne senzorje. Na vrhu sva naredila škatlo, v katero sva namestila vse komponente kot tudi Arduino razvojno ploščo, katera nadzoruje predor. Škatlo sva naredila montažno, tako da je mogoče odstraniti pokrov in dostopati do komponent in spreminjati nastavitve. Celotno maketo sva postavila na kratke noge, da je dvignjena od tal in da sva lahko spodaj speljala kable.

Slika 7: Maketa iz lesa (Vir: Avtor naloge)

8.2. Senzorji

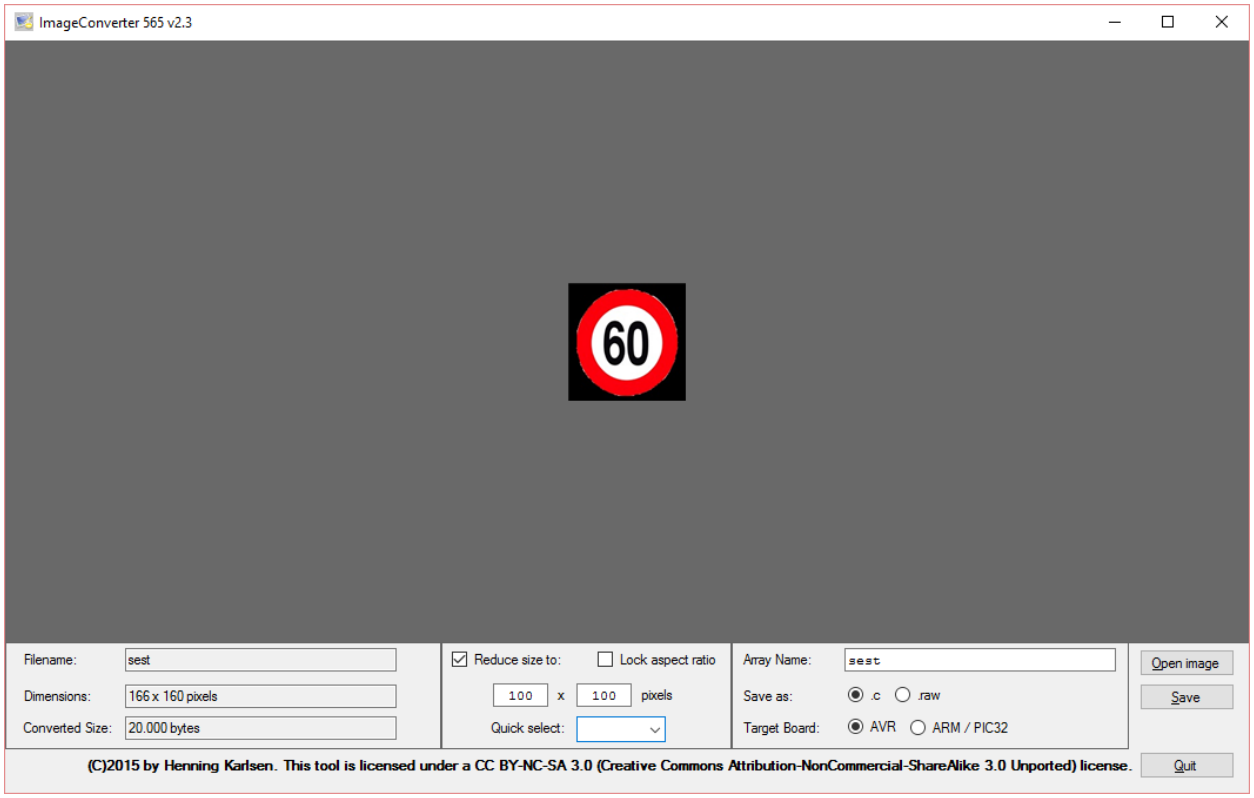
Uporabila sva senzorje za vlago, ogenj, plin in svetlobo. Vsi senzorji delujejo preko razvojne plošče Arduino. Vsi senzorji opravljajo določeno nalogo.

8.3. Ventilatorji

Za prezračevanje sva uporabila ventilatorje. Najprej sva želela narediti cevaste ventilatorje potem pa sva zaradi mnenja, da odprti ventilatorji bolje poskrbijo za prezračevanje uporabila kar te.

8.4. Programska koda za grafičen LCD

Za programiranje grafičnega LCD- ja sva uporabila programski jezik Arduino, ki je podoben jeziku C++. Za najin displej sva morala uporabiti tudi posebno knjižico, saj mora podpirati gonilni čip na najinem displeju. Ker sva želela prikazovati prometne znake, sva morala najti slike le teh in jih pretvoriti v C-kodo, ki jo razume mikrokrmilnik. Za to nalogo sva uporabila poseben progam, ki pretvarja slike iz nam znanih formatov JPG, PNG, BMP … v C kodo, ki jo razumejo krmilniki. Tukaj sva morala precej paziti na velikost slik, saj sva bila omejena z velikostjo pomnilnika najine Arduino ploščice.



Slika 8 Program za pretvarjanje slik ImageConverter 565 v2.3 (Vir: Avtor naloge)

V programu za grafičen LCD je potrebno najprej vključiti knjižnico, ki omogoča uporabo displeja z gonilnikom ILI9325C. Določiti je potrebno tudi katere pine Arduino ploščice bomo uporabili. Ta displej omogoča tudi 16 in 8 bitni prenos podatkov, za najin primer sva uporabila 8 bitni prenos, zato je potrebno pine za 16 bitni prenos povezati z maso, pine za 8 bitni prenos pa povezati na določene pine Arduino ploščice, ki jih določa knjižnica.

```
#include <UTFT.h>  //Knjižica za najin tft display.
#include <avr/pgmspace.h>  //Ukaz ki omogoča hrambo slik v programskem pomnilniku Arduino ploščice.

UTFT myGLCD(ILI9325C,38,39,40,41);  //Določitev gonilnika displaya in pinov na arduino ploščici.
```

Slika 9: Vključitev knjižnice UTFT in določitev gonilnika (Vir: Avtor naloge)

Kot je že prej opisano, sva poiskal slike prometnih znakov in jih nato pretvorila v C kodo, to kodo je kasneje potrebno vnesti tudi v program. Ukazi so razvidni iz slike 10. Iz slike je tudi razvidna možnost uporabe nove pisave. Midva sva želela posebno obliko pisave.

```
extern unsigned int prenos[0x1000];  //Ukaz za vključitev slik v program
extern unsigned int sest[0x1000];
extern unsigned int sto[0x1000];

extern uint8_t DotMatrix_M[];  //Ukaz za izbiro pisave
```

Slika 10: Vnos slik v program in določitev nove pisave. (Vir: Avtor naloge)

V zanki z nastavitvami moramo določiti, kako bodo teksti in slike prikazane, izbiramo lahko med portretno in horizontalno usmeritvijo. Za najin primer je uporabljena portretna nastavitev.

```
void setup()  //nastavitvena zanka
{
  myGLCD.InitLCD(PORTRAIT);  //Usmerjenost slike :
  Serial.begin(9600);
}
```

Slika 11: Usmerjenost slike. (Vir: Avtor naloge)

Z ukazom **myGLCD.**drawBitmap prikažemo sliko shranjeno v programskem pomnilniku krmilnika na displeju. Tukaj določimo še X in Y pozicije za prikaz na dipleju, zapišemo pa tudi velikost slike, ki smo jo že določili pri pretvarjanju v C kodo. Po zapisanem imenu slike lahko po želji tudi vnesemo povečavo slike. S tem sicer dobimo večjo sliko vendar izgubimo na resoluciji.

```
  myGLCD.fillScr(3, 3, 3);  //Barva odzadja displaya.
  myGLCD.drawBitmap (15, 50, 100, 100, sest,2);  //Vstavimo sliko
```

Slika 12: Uporaba slike .(Vir: Avtor naloge)

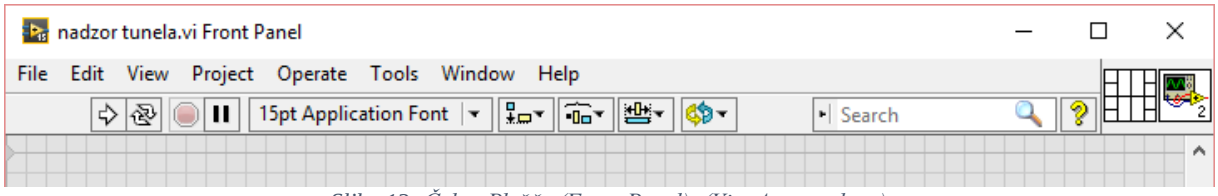
8.5. Programsko okolje LabVIEW

Programsko okolje LabVIEW (Laboratory Virtual Instrument Engineering Workbench) proizvajalca National Instruments (NI) sva uporabila za nadzor najinega predora. To

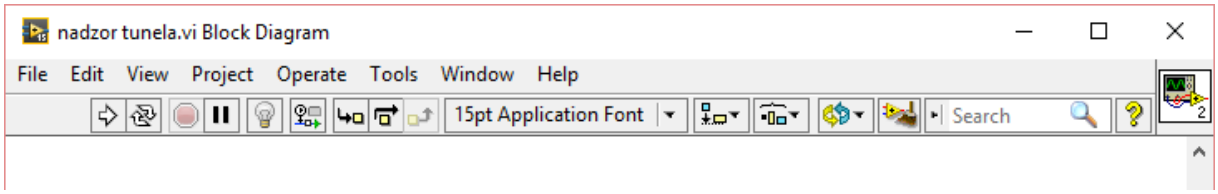
programsko okolje ja zasnovano tako, da omogoča razvoj testnih in merilnih aplikacij, zato se nama je zdel prava izbira. Vsak program je sestavljen iz dveh delov in sicer čelne plošče in blok diagrama. Ker je LabVIEW blokovno orodje to pomeni, da se za vsak program kodira blokovno shemo, sestavljeno iz posameznih blokov, ki so med seboj povezani. Poznamo več različnih povezav. LabVIEW s funkcijo Application Builder omogoča kreiranje programa (exe), ki pa se lahko izvaja na vsakem računalniku, tudi če nima nameščenega LabVIEW razvojnega sistema. Ta funkcija nama je dala le še dodatno potrditev, da je izbira prava, saj bi program tako lažje uporabljali drugi uporabniki, ki nimajo LabVIEW razvojnega sistema, saj je ta kar drag.

- Slika 13: Čelna plošča
- Slika 14: Blok diagram

Čelna plošča ima sivo ozadje z mrežo in je namenjena kreiranju uporabniškega vmesnika, blok diagram pa kreiranju algoritma virtualnega instrumenta.

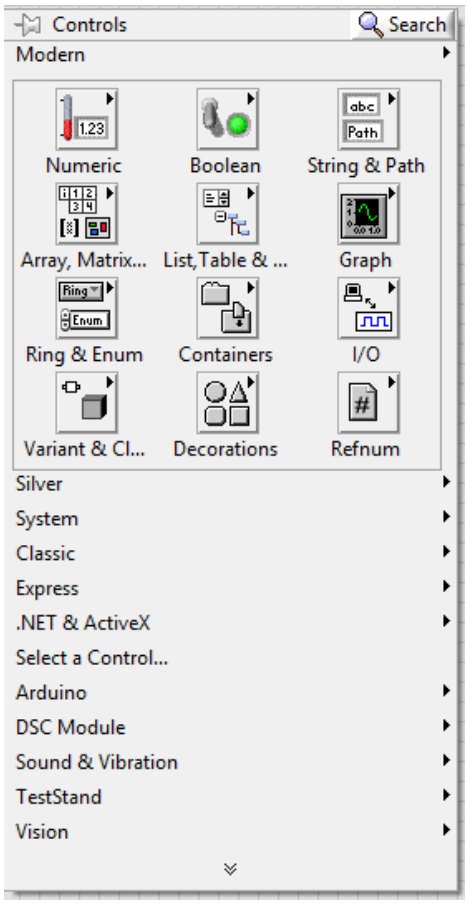


Slika 13: Čelna Plošča (Front Panel). (Vir: Avtor naloge)



Slika 14: Blok diagram (Block Diagram). (Vir: Avtor naloge)

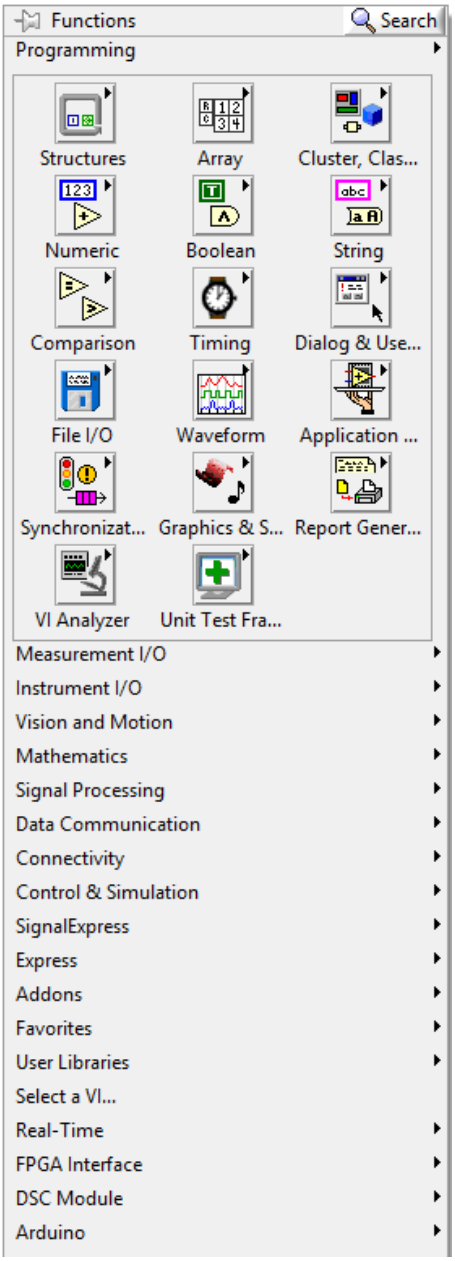
Čelana plošča predstavlja uporabniški vmesnik virtualnega instrumenta, zgradimo ga iz objektov, ki se nahajo na paleti kontrol in indikatorjev. Do te paleta najhitreja dostopamo z desnim klikom miške ali pa View > Controls Pallet. Paleta je sestavljena iz več podpalet, na katerih lahko izbiramo razne kontrole in indikatorje. Te objekte enostavno z miško povlečemo na prazno polje čelne plošče. Kontrole (Controls) ali vhodni objekti predstavljajo vhodne podatke virtualnega instrumenta. Med izvajenjem programa jih lahko spreminjajmo in s tem vplivamo na indikatorje (Indicators) ali na izhodne objekte, ki služijo za tekstovni ali grafični prikaz podatkov.



Slika 15: Paleta kontrol in indikatorje (Vir: Avtor naloge)

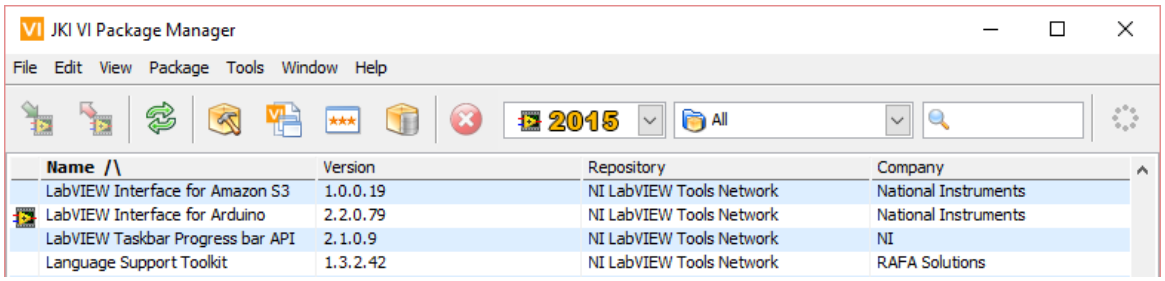
Blok diagram je namenjen kodiranju algoritma virtualnega instrumenta. Zgradimo ga iz objektov, ki se nahajajo na paleti funkcij do katere dostopamo z desnim klikom miške ali pa View > Functions Pallet. Funkcijo namestimo v blok digram tako, da jo izberemo in povlečemo v blok diagram, kjer posamezne funkcije povežemo. V paleti lahko izbiramo različne funkcije ko naprimer:

- Strukture (Structurs), različne programske zanke.
- Številčne funkcije (Numeric), različni matematični ukazi.
- Bollean, različne logične funkcije.
- Arduino, omogoča uporabo Arduino plošč z okoljem LabVIEW



Slika 16: Paleta funkcij (Functions Pallet). (Vir: Avtor naloge)

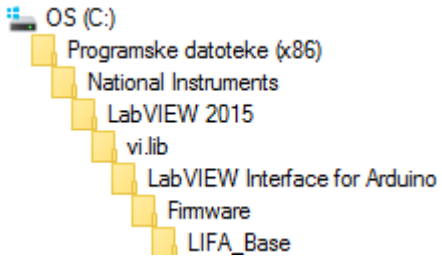
Ker v najinem primeru uporabljava Arduino ploščico, sva morala v LabVIEW vnesti še LIFA base (LabVIEW Interface For Arduino), ki jo najdemo v VI Package Manager-ju. Tukaj sva pridobila funkcije, s katerimi lahko preko serijskega porta krmilimo razvojne plošče Arduino Uno ali Arduino Mega. Nekaj teh funkcij bova kasneje podrobneje opisala.



Slika 17: VI Package Maneger. (Vir: Avtor naloge)

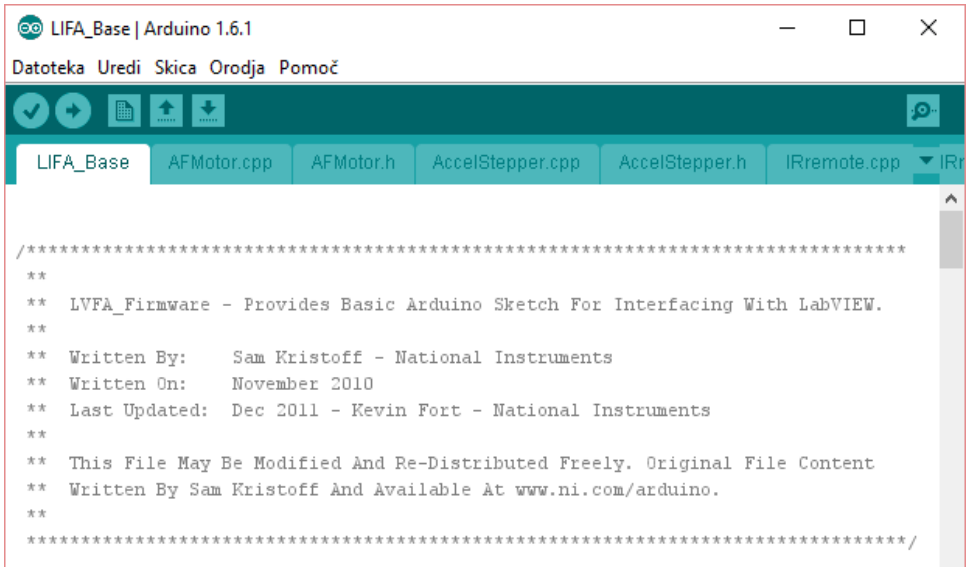
Ko smo namestili LIFA base, se nam naloži tudi gonilnik, ki ga moramo pred uporabo zapisati na našo Arduino ploščico. Najdemo ga po naslednji poti:

C: > Program Files (x86) > National Instruments > LabVIEW 2015 > vi.lib > LabVIEW Interface for Arduino > Firmware > LIFA_Base.



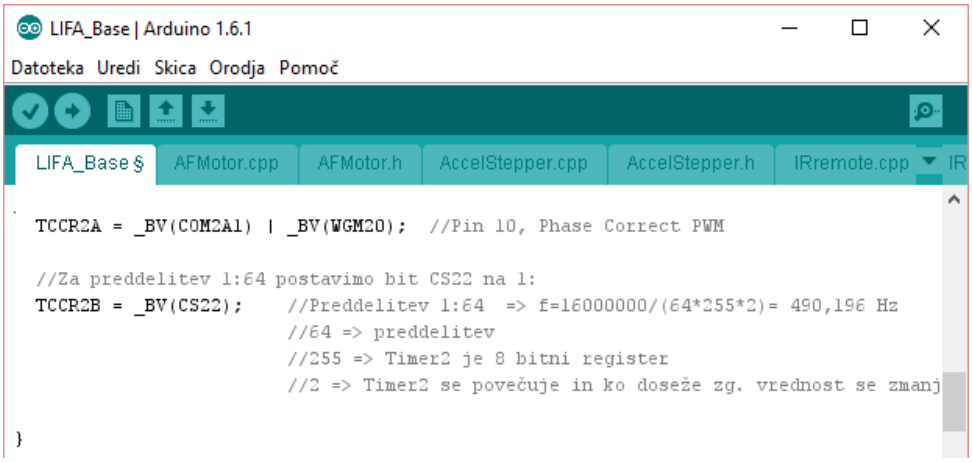
Slika 18: Pot do gonilnika LIFA base (Vir: Avtor naloge)

Tukaj najdemo program napisan v Arduino programskem okolju, ki ga naložimo na Arduino ploščico. Ta program lahko tudi prilagajamo kar sva tudi storila.



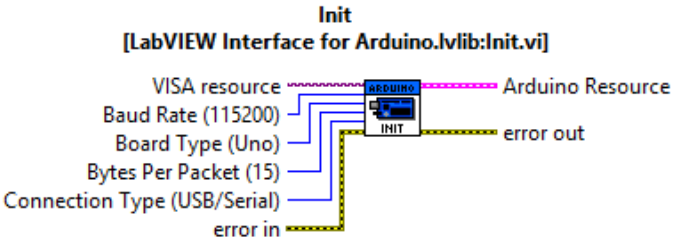
Slika 19: Program LIFA Base, ki ga naložimo na Arduino ploščico. (Vir: Avtor naloge)

Kot sva že prej omenila, sva program nekoliko spremenila in sicer sva dodala v funkcijo voidSetup() del programa, ki spreminja lastnosti časovnika Timer 2, da dosežemo na pinu 10 PWM frekvenco 490.169 Hz, Po najinih praktičnih preizkusih je ventilator najlepše tekel prav pri tej frekvenci.



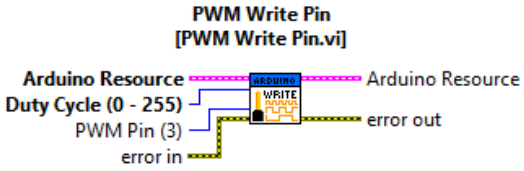
Slika 20: Dodaten del programa za PWM. (Vir: Avtor naloge)

V vsakem virtualnem instrumentu, ki ga želimo uporabljati z Arduino razvojnimi ploščicami moramo najprej vnesti funkcijo Int, s pomočjo katere določimo tip Arduino plošče, serijski vmesnik, vrsto povezane in hitrost prenosa.



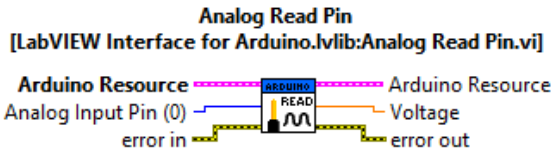
Slika 21: Funkcija int. (Vir: Avtor naloge)

Uporabila sva tudi PWM funkcijo, ki sva ji spremenila frekvenco. S to funkcijo spreminjava hitrost vrtenja ventilatorja. S tem sva dosegla nižjo porabo električne energije, saj v dosedanjih predorih ventilacija stalno deluje s polno močjo ne glede na onesnaženost zraka. V funkciji moramo določiti na katerem pinu bomo spreminjali PWM signal. V najinem primeru je to pin 10, določimo tudi vrednost PWM signala (0-255).



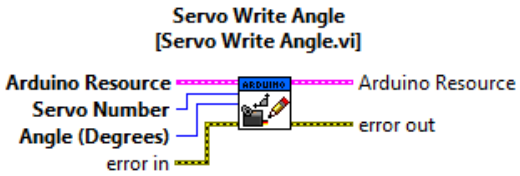
Slika 22: PWM funkcija. (Vir: Avtor naloge)

Ker sva uporabila različne analogne senzorje, sva v najin virtualni instrument vključila blok za branje analognih vhodov. Eden takšnih senzorjev je senzor plina, ki določa s kakšno hitrostjo se bo vrtel ventilator za prezračevanje. Analog Write funkciji moramo določiti kateri vhod bomo brali, na drugi strani funkcije pa dobimo vrednost, ki jo prikažemo tekstovno ali grafično, v najinem primeru je ta vrednost od 0 do 5



Slika 23: Analog Write funkcija (Vir: Avtor naloge)

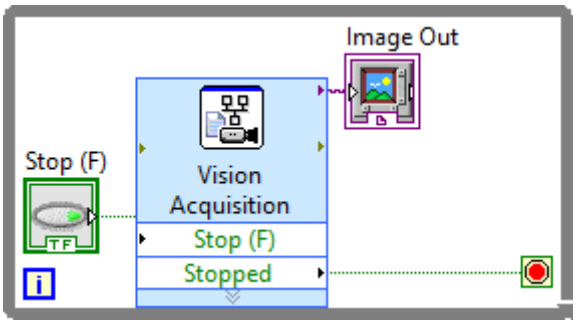
Za spust in dvig zapornic sva uporabila servo motorčke, zato sva morala uporabiti blok za kontrolo servo motorčkov. Bloku moramo podati dva podatka in sicer kateremu servo motorčku želimo spremeni kot (ker po navadi uporabimo več servo motorčkov, moramo vsakemu določiti številko), in kot za koliko naj se premakne os servo motorčka.



Slika 24: Servo Write funkcija. (Vir: Avtor naloge)

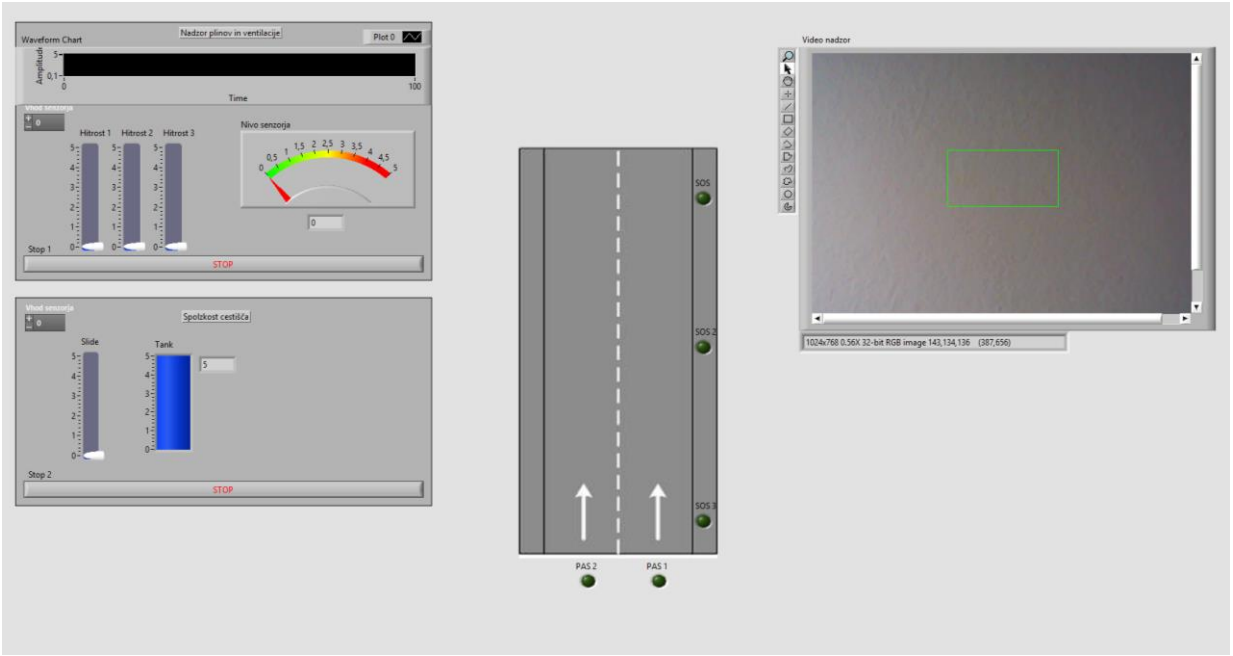
Za video nadzor tunela sva uporabila spletno kamero, saj LabVIEW to omogoča. Za ta namen je potrebno uporabiti funkcijo Vision Acquisition Expres, ki jo najdemo v paleti funkcij pod Vision and Motion. Ko funkcijo povlečemo v polje blok diagrama, se nam samodejno odpre čarovnik za nastavitev kamer. Tukaj moramo izbrati želeno spletno kamero ter neprekinjeno zajemanje slike (posnetek v živo). Ko smo vse storili, se nam prikaže zanka while, namenjena

za prikazovanje posnetka v živo (Slika 25). Hkrati se odpre tudi okno na čelni plošči, kjer lahko spremljamo posnetek.

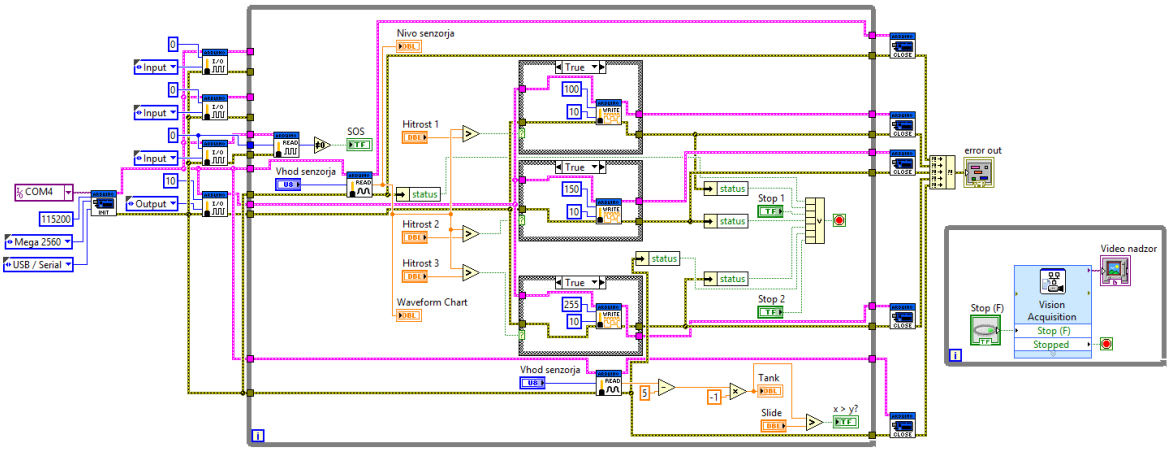


Slika 25: While zanka funkcije Vision Acquisition Expres. (Vir: Avtor naloge)

Kot sva že prej omenjala, lahko operater predora upravlja predor z LabVIEW virtualnim instrumentom. Vključila sva več različnih kontrol in indikatorjev, s katerimi lahko operater nastavlja nivoje za vklop ventilatorjev in kdaj se bo vklopilo katero od opozoril na cesti. Le operater lahko spusti zapornice, ko se preko video nadzora prepriča, da je to varno. Na posnetku lahko vidi ali je kdo pritisnil tipko SOS in temu primerno ukrepa. Različni indikatorji ga opozarjajo na onesnaženost zraka, spolzkost cestišča ter kateri vozni pas je trenutno zaprt.



Slika 26: Virtualni instrument-čelna plošča. (Vir: Avtor naloge)



Slika 27: Virtualni instrument- blok diagram (Vir: Avtor naloge)

9. RAZPRAVA

Naredila sva maketo predora, s pomočjo katere sva preverila vse hipoteze, ki sva si jih postavila. Poskusila sva uporabiti zapornico, s katero bi zaščitili oz. zmanjšali možnost prometne nesreče ob vožnji v nasprotno smer. Rešitev se je izkazala za zelo koristno vendar bi se morala stvar nekoliko prilagoditi. Naši predori niso tako napredni, da bi bila ta rešitev primerna, je pa ideja koristna in bi se čez nekaj let lahko uporabila.

Z uvedbo LED razsvetljave bi lahko zmanjšali porabo električne energije, ta hipoteza je potrjena vendar so LED svetilke močnejše in bleščijo, zato jih ne moremo uporabljati. Napako bi bilo možno odpravit z najinim sistemom razsvetljave, kjer bi svetlost prilagodili in bi bleščanje odstranili.

Uporabila sva 3 stopenjsko prezračevanje, ki se je izkazalo za zelo koristno, saj se glede na težavo, ki se pojavi v predoru, prilagaja ventilacija. Uporabila sva enostavnejši primer tega. kako naj bi to izgledalo. Vendar bi se lahko pri pravih predorih s tehnologijo, ki jo premoremo, v današnjem času dal zgraditi sistem prezračevanja, ki bi odpravil veliko pomanjkljivosti v predorih.

Uporabila sva tablo s spremenljivo vsebino, ki se je izkazala kot zelo koristna. Tablo je možno opaziti na daleč in lahko nadomesti posamezne prometne znake ter voznike obvešča o napakah v predoru in prepreči hujše nesreče. Vozniki so bolj pozorni na table kot na znake. Prav tako se vsebina table stalno spreminja glede na razmere v predoru, kar se nama zdi zelo koristno.

10. ZAKLJUČEK

Z nalogo sva zadovoljna. Rezultati so takšni kot sva jih vnaprej pričakovala. Pri nalogi ni bilo večjih zapletov. Z nalogo sva ugotavljala napredek in tehnologijo v tunelih. Dopolnila sva svoje znanje iz programiranja in elektronike, ki ga bova uporabila v nadaljnjem študiju. Prav tako sva dopolnila znanje o predorih in dobila nekaj zelo zanimivih podatkov o svetovnih predorih. Zelo bi bila vesela, če bi nekega dne uporabili najino tehnologijo v pravih tunelih in bi Slovenija bila ena izmed prvih držav s takšno tehnologijo. Glede na to da se število prometnih nesreč povečuje je pomembno, da se na varnost v prometu veliko vlaga.

11. DRUŽBENA ODGOVORNOST

Delala sva družbeno odgovorno. Meniva da bo najin inovacijski predlog koristil ljudem in izboljšal pogoje v predorih. Uporabljala sva materiale, ki so okolju prijazni in ne škodujejo ljudem. Z uporabo LED razsvetljave bi lahko privarčevali in s tem pomagali okolju in družbi, prav tako bi s tem zmanjšali razlike svetlobe pri prehodu v predor ali odhodu iz predora. S tablo bi zmanjšali možnost nesreč, saj je tabla bolj vidljiva od znakov. V današnjem času je vse več avtomobilov in število prometnih nesreč se stalno povečuje, zato je potrebno vložiti veliko truda za izboljševanje razmer in tehnologij v cestnem prometu in predorih.

12. VIRI:

https://www.arduino.cc/en/Products/Counterfeit(18.01.2016)
http://www.pisrs.si/Pis.web/pregledPredpisa?id=PRAV6453 (18.01.2016)
http://wol.jw.org/sl/wol/d/r64/lp-sv/102002487 (20.01.2016)
https://sl.wikipedia.org/wiki/Predor#Slovenski_predori (20.01.2016)
https://sites.google.com/site/solaelektronikesers/ (30.09.2015)
http://tronixstuff.com/2013/04/26/tutorial-arduino-and-ili9325-colour-tft-lcd-modules/ (6.01.2016)
http://www.rinkydinkelectronics.com/library.php?id=51 (29.01.2016)