»MLADI ZA NAPREDEK MARIBORA«
34 SREČANJE

Uporaba senzorjev s krmilnim modulom NI myRIO

Raziskovalno področje: Elektrotehnika, elektronika

Raziskovalna naloga

Avtor: ANDREJ ŠRUMPF
Mentor: MILAN IVIČ
Šola: SREDNJA ELEKTRO-RAČUNALNIŠKA ŠOLA MARIBOR

Maribor, januar 2017

Vsebina

Kazalo slik

1. POVZETEK

Raziskovalna naloga predstavlja uporabo različnih senzorjev ter krmiljenje določenih naprav na podlagi rezultatov, ki jih podajajo uporabljeni senzorji. V nalogi je uporabljen senzor gibanja, senzor alkohola in drugih plinov, senzor temperature, senzor razdalje, senzor gibanja in senzor svetlobe. Za izvršilne elemente pa so uporabljeni enosmerni motorček, ventilatorja, LED diode, grelec in drugi elementi, ki zagotavljajo delovanje po postavljenih zahtevah.

Za krmiljenje izhodnih elementov ter kontrolo vhodnih elementov je uporabljen krmilni modul myRIO proizvajalca National Instruments. Programska koda je izdelana v okolju LabVIEW, omogoča pa nam izdelavo uporabniškega vmesnika, ki nam slikovito z uporabo primernih kontrol in indikatorjev prikazuje delovanje naloge.

Za namen prikaza delovanja naloge je izdelan praktični model, s pomočjo katerega lahko spremljamo, nadziramo in krmilimo delovanje uporabljenih elementov.

2. UVOD

V današnjem času se avtomatsko upravlja oziroma krmili vse več naprav, od razsvetljave v domačem okolju, mestnih ulicah in na določenih cestnih odsekih do avtomatizacije proizvodnih procesov, ko prisotnost človeka ni potrebna, saj njegovo nalogo prevzamejo stroji, pa do vse pogostejše uporabe inteligentnih sistemov.

V tej raziskovalni nalogi je izdelan preprost, pa vendar uporaben sistem, s katerim bi lahko avtomatizirali delovanje naprav v prostorih stanovanjske hiše, v obrtnih delavnicah in drugih objektih. Za nadzorno enoto smo se odločili uporabiti računalnik z nameščenim programskim okoljem LabVIEW, z njim izdelati nadzorno ploščo – programski vmesnik, na katerem lahko spremljamo, nastavljamo in upravljamo delovanje modela.

3. VSEBINSKI DEL

3.1 Zahteve delovanja

Uporabljeni so naslednji senzorji:

- Senzor gibanja (PIR - **p**asivni **IR**[1] - senzor HC-SR501).
- Senzor svetlobe, fotoupor LDR.
- Senzor alkohola in drugih plinov (MQ3).

[1] IR infra rdeči

- Senzor temperature LM35.
- Senzor razdalje GP2Y0A21.

Senzor gibanja:

Na uporabniškem vmesniku naj bo prikazana indikacija stanja senzorja gibanja. V kolikor senzor gibanja zazna gibanje, se mora na modelu vklopiti rdeča LED[2] dioda, na vmesniku pa se morata vklopiti simulacijski LED diodi. Prva simulacijska LED vklopi, napis iz *Ni zaznanega gibanja* se spremeni v *Zaznano gibanje*. Druga, večja simulacijska LED dioda na vmesniku se vklopi (obarva rdeče) in prikaže napis *Pozor gibanje!*

Senzor plinov:

V kolikor senzor zazna večjo koncentracijo plinov, vklopi ventilator. Mejna koncentracija plinov, ko vklopi ventilacija, se določa v programski kodi. Ventilator na modelu krmilimo s PWM[3] signalom, večja kot je koncentracija plinov, hitreje se ventilator vrti. Na uporabniškem vmesniku se naj prikazuje koncentracija plinov (napetost na izhodnem priključku senzorja v območju od 0 V do 5 V), indikacija vklopljenega oziroma izklopljenega ventilatorja, duty cycle[4] PWM signala v območju od 0 % do 100 %, frekvenco PWM signala pa je prek uporabniškega vmesnika mogoče spreminjati v območju od 50 Hz do 100 Hz.

Senzor svetlobe, fotoupor:

Razsvetljava na modelu vklopi oziroma izklopi v odvisnosti od svetlobe, ki jo zaznava fotoupor LDR[5]. Na uporabniške vmesniku naj bo prikazana napetost na fotouporu v območju od 0 V do 5 V ter simulacijska LED dioda, ki prikazuje stanje razsvetljave. Mejo, ko naj vklopi oziroma izklopi razsvetljava je možno spreminjati na uporabniškem vmesniku. Na modelu se mora vklopiti oz. izklopiti razsvetljava, odvisno od svetlobe, ki jo zaznava fotoupor.

Senzor temperature:

Na uporabniškem vmesniku se naj prikazuje izmerjena temperatura na termometru v °C (območje od 10 °C do 40 °C). Želeno temperaturo in temperaturno toleranco je mogoče nastaviti na vmesniku. Če temperatura pade pod spodnjo nastavljeno vrednost, vklopi grelec in izklopi, ko je dosežena zgornja

[2] LED (ang. Light emmiting diode, svetleča dioda)
[3] PWM (ang. pulse width modulation), pulzno širinska modulacija.
[4] Duty cycle predstavlja razmerje med širino impulza in trajanjem periode, izraženo je v %.
[5] LDR (ang. light dependent resistor), svetlobno odvisen upor.

nastavljena vrednost. Takrat vklopi ventilator ki izklopi, ko temperatura pade pod spodnjo nastavljeno vrednost. Delovanje grelca in ventilatorja naj simulirata LED diodi na uporabniškem vmesniku. Na modelu naj senzor temperature greje upor (tok skozi upor), hladi pa ventilator.

Senzor razdalje:

Na uporabniškem vmesniku se naj prikazuje izmerjena razdalja predmeta, oddaljenega od senzorja v razdalji od 10 cm do 80 cm. Izmerjena razdalja se naj prikazuje tudi na grafu. V odvisnosti od razdalje predmeta od senzorja se mora spreminjati hitrost vrtenja enosmernega motorčka (sprememba PWM signala), kateremu lahko spreminjamo tudi smer vrtenja. Če senzor razdalje izmeri manjšo razdaljo predmeta od senzorja, ki smo jo določili, mora simulacijska LED dioda na uporabniškem vmesniku na to opozoriti z utripanjem. Smer vrtenja enosmernega motorčka spreminjamo z gumboma na uporabniškem vmesniku. Simulacijske LED diode naj prikazujejo smer vrtenja in delovanje enosmernega motorčka.

Vklop/Izklop virtualnega instrumenta:

Vklop in izklop virtualnega instrumenta naj bo izvedeno s stikalom. Pred zagonom virtualnega instrumenta moramo stikalo vklopiti (ON). Virtualni instrument deluje, dokler ga s stikalom ne izklopimo (OFF). Po izklopu virtualnega instrumenta morajo vse simulacijske LED diode na vmesniku (Čelna plošča) izklopiti. Izklopiti morajo tudi vse naprave na modelu, ventilatorja, enosmerni motorček, grelec, razsvetljava in LED dioda, ki prikazuje gibanje v prostoru.

Privzete vrednosti:

Vrednosti, ki so privzeto nastavljene ko vklopimo virtualni instrument oziroma se nastavijo, ko virtualni instrument izklopimo:

- Želena temperatura v prostoru: 25 °C.
- Temperaturna toleranca: 1 °C.
- Nastavitev vklopa razsvetljave v odvisnosti od svetlobe: 2,5 V.
- Frekvenca PWM signala za ventilator in enosmerni motorček: 50 Hz.
- Duty cycle PWM signala za ventilator in enosmerni motorček: 0 %.
- Indikator gibanja: Prva LED => izklopljena, napis *Ni zaznanega gibanja*, druga LED => izklopljena.
- Minimalna razdalja predmeta od senzorja razdalje: 10 cm.
- Vse ostale simulacijske LED diode na vmesniku: Izklopljene.

Vhodi:

- Senzor plinov bomo priključili na konektor B, priključek 3 (B/AI0).
- Senzor svetlobe bomo priključili na konektor B, priključek 5 (B/AI1).
- Senzor gibanja bomo priključili na konektor A, priključek 3 (A/AI0).
- Senzor razdalje bomo priključili na konektor B, priključek 7 (B/AI2).
- Senzor temperature bomo priključili na konektor B, priključek 9 (B/AI3).

Izhodi:

- Ventilator, ki vklopi ob povečani koncentraciji plinov, konektor C, priključek 14 (C/DIO3/PWM0).
- Enosmerni motorček, konektor C, priključek 18 (C/DIO7/PWM1).
- Razsvetljava, ki vklopi v odvisnosti od svetlobe, konektor C, priključek 11 (C/DIO0).
- LED dioda, ki vklopi, če senzor gibanja zazna gibanje, konektor C, priključek 12 (C/DIO1).
- Grelec za gretje prostora, konektor C, priključek 13 (C/DIO2).
- Ventilator za hlajenje prostora, konektor C, priključek 15 (C/DIO4).
- Gonilnik za enosmerni motorček, vrtenje v desno, konektor B, priključek 31 (B/DIO10).
- Gonilnik za enosmerni motorček, vrtenje v levo, konektor B, priključek 29 (B/DIO9).

3.2 Sestavni deli

3.2.1 Senzor gibanja

Uporabili smo senzor gibanja HC-SR501, pasivni infra rdeči senzor. Osnovne lastnosti:

- Napetost napajanja: 5 V do 20 V.
- Tokovna poraba: 65 mA.
- Tokovna poraba v času mirovanja: manjša od 50 µA.
- Področje zaznave gibanja: 120°, 7 m.
- Temperaturno območje: -15 °C do +70 °C.
- Digitalni izhod: ni gibanja => 0V, zaznano gibanje => 3,3 V.

Slika 1: Senzor gibanja HC-SR501 (vir: http://www.datasheet-pdf.download/hc-sr501-pir-motion-sensor/).

3.2.2 Senzor plinov MQ3

Uporabili smo senzor Openjumper OJ-CG310. Osnovne lastnosti:

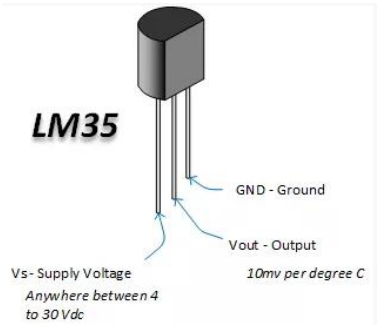
- Delovna napetost: 5 V.
- Izhod: analogni izhod.
- Osnova: MQ3.
- Zaznava: alkohol, etanol, ogljikov monoksid, …

Slika 2: Senzor plinov (vir: miniinthebox.com).

3.2.3 Senzor temperature LM35

Uporabili smo analogni temperaturni senzor LM55. Osnovne lastnosti:

- Temperaturno območje: -55 °C do 150 °C.
- Sprememba napetosti na izhodu: 10 mV/°C.
- Napetost na izhodu pri temperaturi 25 °C: 250 mV.



Slika 3: Senzor temperature LM35 (vir: http://henrysbench.capnfatz.com/).

3.2.4 Senzor razdalje Sharp GP2Y0A21

Uporabili smo analogni senzor razdalje GP2Y0A21. Osnovne lastnosti:

- Merilna razdalja: 10 cm do 80 cm.
- Analogni izhod: 0,25 V do 3,1 V.
- Napetost napajanja: 4,5 V do 5,5 V.
- Tokovna poraba: 30 mA.
- Odzivni čas: 38 +/- 10 ms.

3.2.5 Enosmerni motorček

Ko na enosmerni motorček priključimo enosmerno napetost, se zaradi menjavanja smeri magnetnega polja vrti. Zgrajen je iz mirujočega, statorskega železnega jedra, na katerem se nahaja vzbujalno navitje za ustvarjanje magnetnega polja. Med magnetnimi poli statorja se nahaja rotor z navitjem, povezanim preko komutatorja in ščetk na zunanji vir enosmerne napetosti. Če menjamo polariteto priključene enosmerne napetosti, se motorček vrti v drugo smer.

Za spreminjanje polaritete priključene napetosti in s tem spreminjanje smeri vrtenja DC motorčka uporabimo H-mostiček v integriranem vezju L293D.

Slika 4: Enosmerni motorček (vir: avtor naloge).

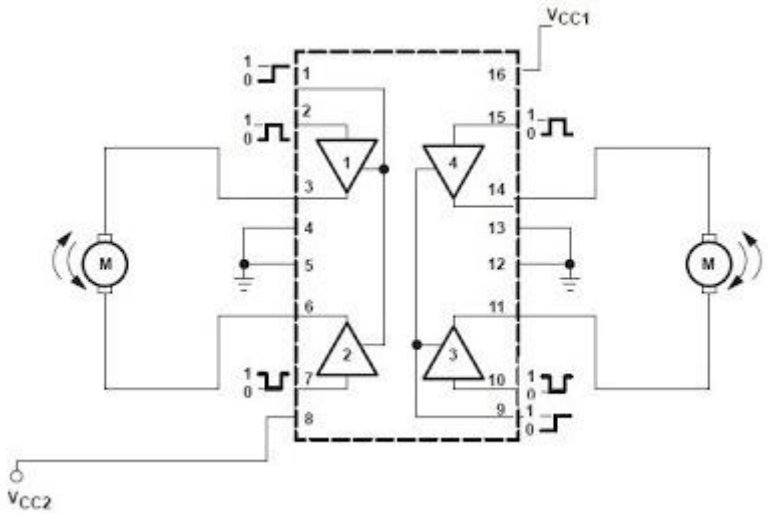
3.2.6 Brezkrtačni enosmerni motorček

Brezkrtačni električni motorček je električni motorček, ki nima komunikatorjev ali drsnih obročev ter kolektorja. Namesto tega ima krmilno elektroniko, ki uravnava delovanje motorčka. Prednost je daljša življenjska doba, manjša obraba, večji izkoristek, visoka hitrost vrtenja, odsotnost iskrenja, visoka moč in zanesljivost. Slabost teh motorčkov pa je višja cena zaradi elektronike in večja kompleksnost. Uporabili smo dva različno velika računalniška ventilatorja s to vrsto motorja.

Slika 5: Računalniški ventilator (vir: avtor naloge).

3.2.7 Gonilnik L293D

Integrirano vezje L293D je namenjeno za krmiljenje dveh enosmernih motorčkov, da se lahko vrtita v obe smeri. Mi bomo krmilili en motorček. Nanj lahko priključimo motorček, ki za svoje delovanje ne potrebuje toka, večjega od 600 mA in za napetosti od 4,5 V do 36 V. Vsi vhodi so združljivi s TTL signali. Za zaščito ima L293D vgrajene hitre diode. Te ščitijo integrirano vezje pred napetostnimi konicami, ki nastanejo pri vklopu in izklopu motorčka (predvsem pri izklopu). Če se L293D segreje čez mejo 70°C, vgrajeni senzorji ustavijo delovanje motorčka.



Slika 6: Priključitev enosmernih motorčkov na L293D (vir: Texas Instruments).

Motorček bomo priključili na priključka 3 in 6, krmilili pa ga bomo s priključki 1, 2 in 7. Na priključek 8 (Vcc_2) priključimo napetost, ki jo potrebuje enosmerni motorček za delovanje, na priključek 16 (Vcc_1) pa napetost za gonilnik L293D, ki znaša 5 V.

Tabela 1: Vloga priključkov gonilnika L293D.

Priključek 1 (EN)	Priključek 2	Priključek 7	Funkcija
1	0	1	Vrtenje motorčka v smeri urinega kazalca.
1	1	0	Vrtenje proti smeri urinega kazalca.
1	0	0	Stop.
1	1	1	Stop.
0	neuporabno	neuporabno	Stop.

Priključka 1 in 9 sta enable priključka. Povezana morata biti na +5 V (logična 1), če želimo da se bosta motorčka vrtela. Če želimo spreminjati hitrost vrtenja enosmernega motorčka, na ta priključek

pripeljemo PWM signal. Ker imamo mi priključen le en motorček, priključen na levo stran, poglejmo način krmiljenja motorčka (enako velja za drugi motorček):

- Priključka 4 in 5 morata biti povezana na skupno maso (GND).
- Priključka 2 in 7 sta vhodna kontrolna priključka gonilnika L292D. Nadzorujeta delovanje motorčka.
- Priključka 3 in 6 sta izhodna priključka gonilnika L293D, kamor priključimo DC motorček.
- Priključek 16 je namenjen napetostnemu napajanju gonilnika L293D. Priključen mora biti na napetost +5 V.
- Priključek 8 je priključek, kamor priključimo pozitivno napetost iz drugega napajalnika oziroma baterije za napetost DC motorčka (od 4,5 V do 36 V, odvisno od uporabljenega motorčka).

3.3 Krmilni modul myRIO – 1900

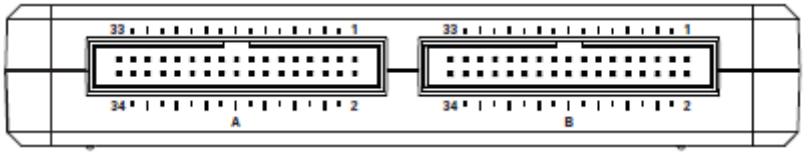
Modul myRIO - 1900 proizvajalca National Instruments poganja dvojederni procesor ARM Cortex A9, vsebuje čip Xilinx FPGA (Field Programmable Gate Array). Na treh konektorjih A, B in C je na razpolago 10 analognih vhodov, 6 analognih izhodov in 40 digitalnih vhodov/izhodov. Pozorni moramo biti, da ne preobremenimo posameznih izhodov.

Slika 7: Krmilni modul myRIO - 1900 (vir: National Instruments).

Modul myRIO najprej priključimo na napetostno napajanje, nato pa ga prek USB vrat povežemo z računalnikom. Odpre se okno *myRIO USB Monitor*, kjer je podatek s serijsko številko modula in IP naslovom. Z modulom se lahko povežemo tudi prek Wi-Fi[6] povezave.

[6] Wi-fi (ang. Wireless Fidelity, lokalno brezžično omrežje)

Konektor B vsebuje 16 vhodno/izhodnih digitalnih priključkov od DIO0 do DIO15. Posamezne priključke lahko uporabimo kot vhodne ali pa kot izhodne priključke. Priključki od DIO0 do DIO13 so prek pull-up[7] upora vrednosti 40 kΩ povezani na napetost 3,3 V, priključka DIO14 in DIO15 pa prek upora vrednosti 2,1 kΩ povezana na napetost 3,3 V. Enako velja za **konektor A**. Posamezni digitalni izhod lahko obremenimo s tokom 4 mA.



Slika 8: Konektorja A in B modula myRIO (vir: National Instruments).

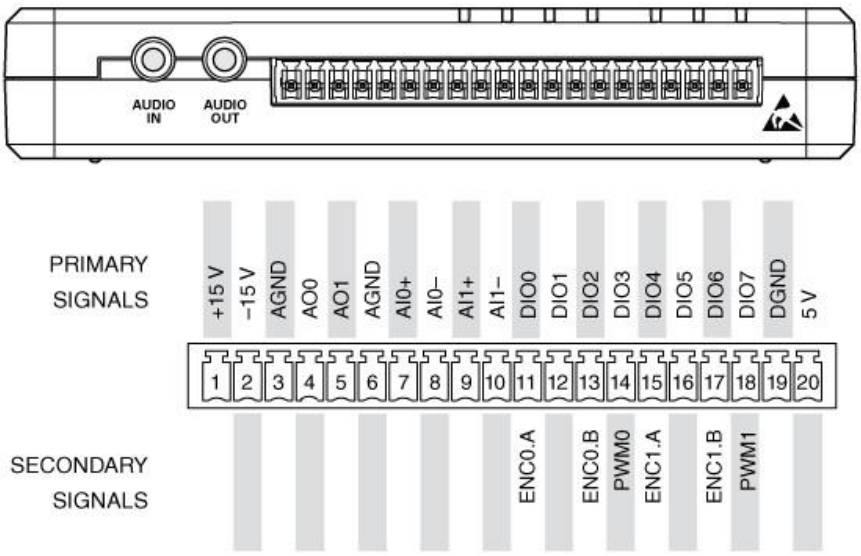
+3.3 V	DIO10 / PWM2	DIO9 / PWM1	DIO8 / PWM0	DIO7 / SPI.MOSI	DIO6 / SPI.MISO	DIO5 / SPI.CLK	DIO4	DIO3	DIO2	DIO1	DIO0	AI3	AI2	AI1	AI0	+5V
33	31	29	27	25	23	21	19	17	15	13	11	9	7	5	3	1
34	32	30	28	26	24	22	20	18	16	14	12	10	8	6	4	2
DIO15 / I2C.SDA	DIO14 / I2C.SCL	DGND	DGND	DIO13	DGND	DIO12 / ENC.B	DGND	DIO11 / ENC.A	DGND	UART.TX	DGND	UART.RX	DGND	AGND	AO1	AO0

Slika 9: Priključki konektorja A oz. B (vir: National Instruments).

Nekateri priključki konektorja B (enako velja za konektor A) imajo še dodatne funkcije kot so SPI vodilo, I2C, pulzno širinsko modulacijo (PWM) in druge funkcije.

Konektor C vsebuje 8 digitalnih vhodno/izhodnih priključkov od DIO0 do DIO7. Posamezne priključke lahko uporabimo kot vhodne ali pa kot izhodne priključke. Vsak priključek je prek pull-down[8] upora vrednosti 40 kΩ povezan na maso. Za maso imajo skupni priključek DGND (priključek 19). Logični nivo 1 da na izhodnem priključku napetost 3,3 V. Obremenimo ga lahko s tokom 4 mA.

[7] Pull-up Upor povezan med signalno linijo in pozitivno napetostjo.
[8] Pull-down Upor povezan med signalno linijo in maso.



Slika 10: Konektor C in opis priključkov (vir: National Instruments).

Analogni vhodi imajo 12-bitno ločljivost. Nanje lahko priključimo napetost od 0 V do 5 V. Na diferencialna analogna vhoda AI0 in AI1 konektorja C lahko priključimo napetost o območju od -10 V do +10 V.

Analogni izhodi vsebujejo 12-bitne digitalno analogne pretvornike. Maksimalna napetost na analognem izhodu znaša 5 V. Posamezni analogni izhod lahko obremenimo s tokom 3 mA. Maksimalna napetost na analognih izhodih AO0 in AO1 konektorja C znaša od -10 V do +10 V.

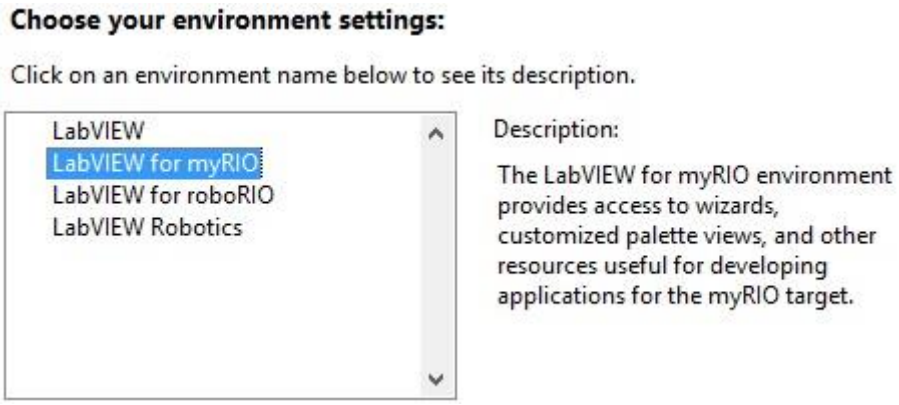
Za napetostno napajanje modula myRIO potrebujemo napajalnik moči 14 W z napetostjo od 6 V do 16 V.

3.4 Programsko okolje LabVIEW

Programsko okolje LabVIEW (Laboratory Virtual Instrument Engineering Workbench) proizvajalca National Instruments je zasnovano za znanstvenike in inženirje, da lahko opravljajo meritve in analize zajetih podatkov. Je v celoti grafično zasnovano ter omogoča razvoj merilnih in testnih aplikacij. Programi napisani z orodjem LabVIEW se imenujejo virtualni instrumenti (VI). Vsak virtualni instrument je sestavljen iz čelne plošče in blok diagrama.

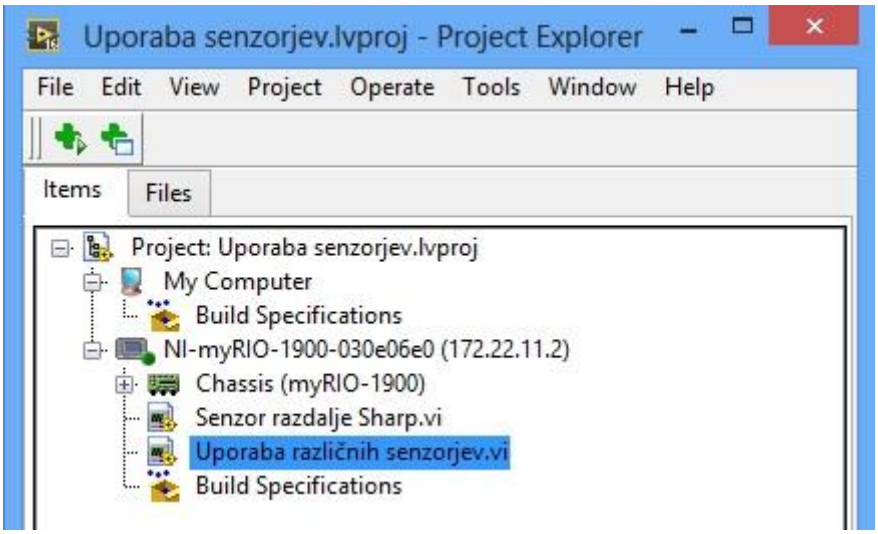
LabVIEW je blokvno orodje, kar pomeni, da se algoritem v tem programskem jeziku kodira blokovno oziroma z uporabo funkcij v obliki blokov, ki so med seboj povezani s povezavami. Vsebuje tudi *Application Builder*, ki omogoča kreiranje izvršne (exe) verzije, ki se lahko izvaja na računalniku, kjer programsko okolje LabVIEW ni nameščeno.

Ob zagonu programa LabVIEW se pojavi okno, kjer lahko izbiramo med različnimi nastavitvami. Ker bomo uporabili krmilni modul myRIO, izberemo LabVIEW for myRIO.



Slika 11: Različne nastavitve okolja LabVIEW (vir: avtor naloge).

Po izbiri nastavitve se pojavi okno, kjer kreiramo novi projekt. Novi projekt imenujemo in mu dodelimo prek UBS vrat priključen krmilni modul myRIO. Projektu kreiramo virtualni instrument (VI).



Slika 12: Kreirani projekt v okolju LabVIEW (vir: avtor naloge).

Ob kreiranju virtualnega instrumenta se pojavita dve okni, blok diagram in čelna plošča. Čelna plošča ima privzeto sivo ozadje in je namenjena kreiranju uporabniškega vmesnika, blok diagram pa kreiranju algoritma virtualnega instrumenta.

Čelna plošča (Front Panel) predstavlja uporabniški vmesnik virtualnega instrumenta (VI). Zgradimo ga iz objektov, ki se nahajajo na paleti kontrol in indikatorjev. Do te palete lahko dostopamo na dva ničina: Preko izbire menija *View > Controls Palette* ali pa kjerkoli v praznem prostoru čelne plošče pritisnemo na desno tipko miške.

Paleta kontrol in indikatorjev je sestavljena iz več podpalet. Objekt (kontrolo ali indikator) namestimo na čelno ploščo tako, da ga izberemo in ga povlečemo na čelno ploščo. Za vsak objekt, ki ga vstavimo na čelno ploščo, se v blok diagramu avtomatsko kreira priključek. Priključki služijo za prenos podatkov med čelno ploščo in blok diagramom. Ime priključka v blok diagramu je enako, kot je ime pripadajočega objekta na čelni plošči. Iz oblike priključka lahko razberemo, ali pripada kontroli ali indikatorju.

Vhodni objekti, pravimo jim kontrole, predstavljajo vhodne podatke virtualnega instrumenta. Te objekte lahko med izvajanjem virtualnega instrumenta spreminjamo in s tem vplivamo na njegovo delovanje. Izhodni objekti, ki jim pravimo indikatorji, pa služijo za prikaz podatkov.

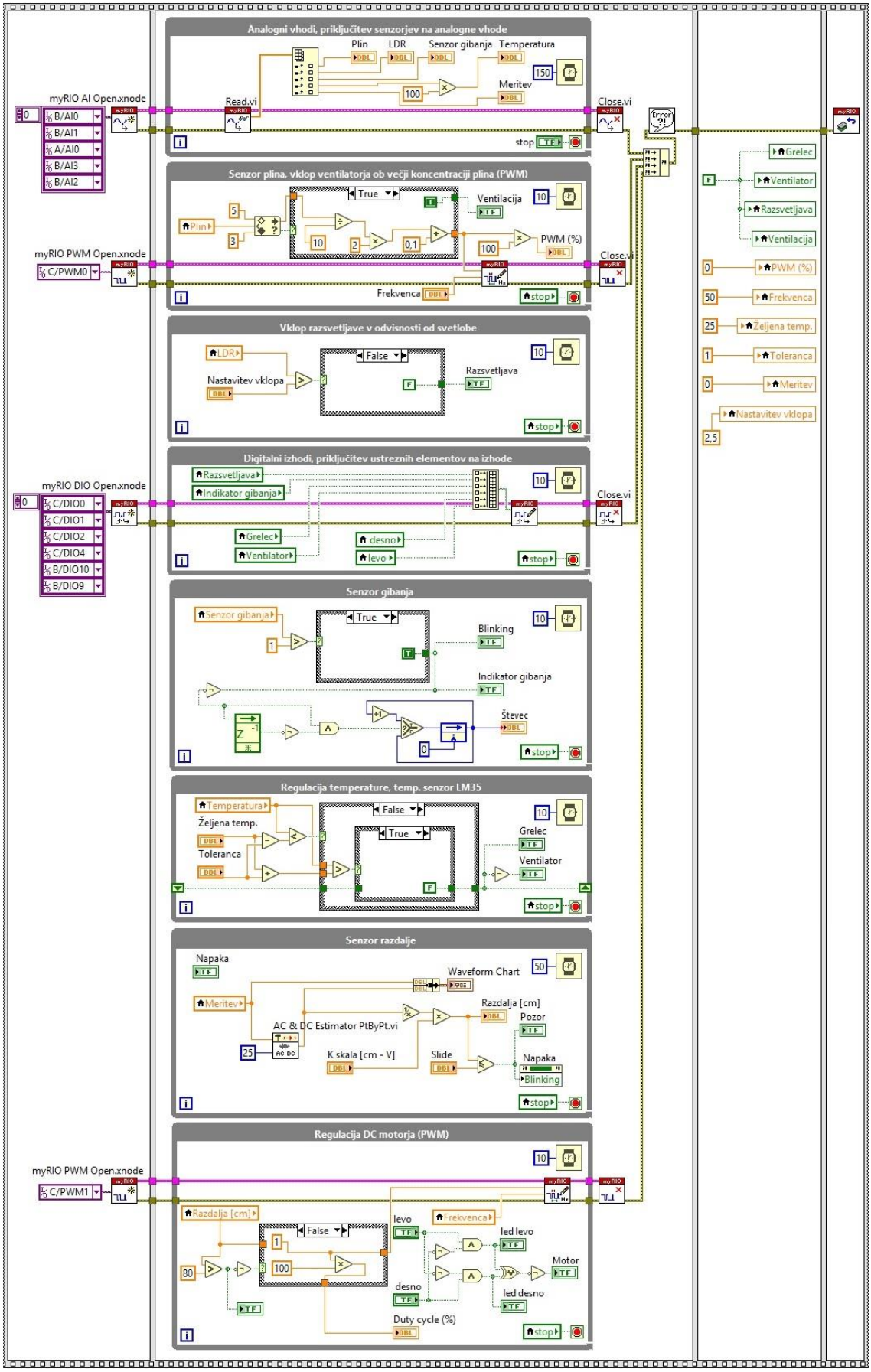
Blok diagram (Block Diagram) je namenjen kodiranju algoritma virtualnega instrumenta (VI). Algoritem zgradimo iz objektov, ki se nahajajo na paleti funkcij, te objekte pa med seboj povežemo s povezavami. Do palete funkcij dostopamo na dva načina: Preko izbire menija *View > Function Palette* ali pa kjerkoli v praznem prostoru pritisnemo na desno tipko miške.

Funkcijo namestimo v blok diagram tako, da jo na paleti funkcij izberemo in povlečemo v blok diagram. V blok diagramu virtualnega instrumenta se lahko nahajajo naslednji objekti:

- Priključki (Terminals), to so priključni elementi objektov čelne plošče.
- Podprogrami (SubVIs), to so dejansko virtualni instrumenti, ki prav tako vsebujejo čelno ploščo in blok diagram.
- Funkcije (Functions).
- Konstante (Constants), to so objekti v blok diagramu, katerim se vrednost med izvajanjem virtualnega instrumenta ne spreminja.
- Programske strukture (Structures), to so objekti, kot so zanka While, zanka For, pogojni stavki (Case) zaporedje (Flat Sequence) in drugi objekti.
- Povezave (Wires), ki povezujejo posamezne objekte Blok diagrama in služijo prenosu podatkov.
- Objekti in funkcije iz podpalete myRIO.

3.5 Programska koda (algoritem) naloge Uporaba senzorjev

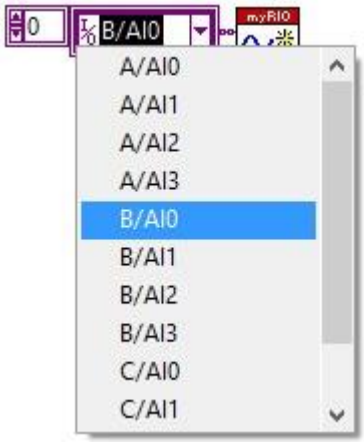
Programska koda oziroma algoritem virtualnega instrumenta je razdeljena v štiri sekvence Flat. Sekvenca oziroma zaporedje je programska struktura, ki je namenjena določitvi vrstnega reda izvajanja kode. Uporabili smo jo zato, ker je pomemben vrstni red izvajanja posameznih segmentov kode.



Slika 13: Programska koda (algoritem), blok diagram (vir: avtor naloge).

3.5.1 Prvi okvir sekvence Flat

V prvem okvirju sekvence določimo vhodne analogne kanale (priključke), izhodne digitalne kanale (priključke) in PWM izhodna kanala (priključka). Za določitev analognih vhodnih priključkov izberemo funkcijo *Open.vi* (*Functionc > Programming > myRIO > Low Level > Analog Input 1 Sample > Open*). Z miško se približamo vhodnemu terminalu (*Channel Names*), pritisnemo na desno tipko miške ter kreiramo konstanto. V bistvu smo kreirali eno dimenzionalno polje, ki predstavlja zaporedje podatkov - vhodnih kanalov. Desni okvirček kreirane konstante razširimo ter na spustnem seznamu eno dimenzionalnega polja določimo kanale, analogne vhode, ki jih bomo uporabili v nalogi. Na razpolago imamo vse analogne vhode modula myRIO na vseh treh konektorjih.



Slika 14: Izbira analognih vhodov modula myRIO (vir: avtor naloge).

Na enak način izberemo PWM izhodna priključka ter digitalne izhodne priključke, ki jih bomo uporabili v nalogi. Pri izbiri PWM izhoda na podpaleti *Low Level* izberemo *PWM > Open.vi*, pri izbiri digitalnih izhodov pa na podpaleti *Low Level* izberemo *Digital Input/Output 1 Sample > Open.vi*.

3.5.2 Drugi okvir sekvence Flat:

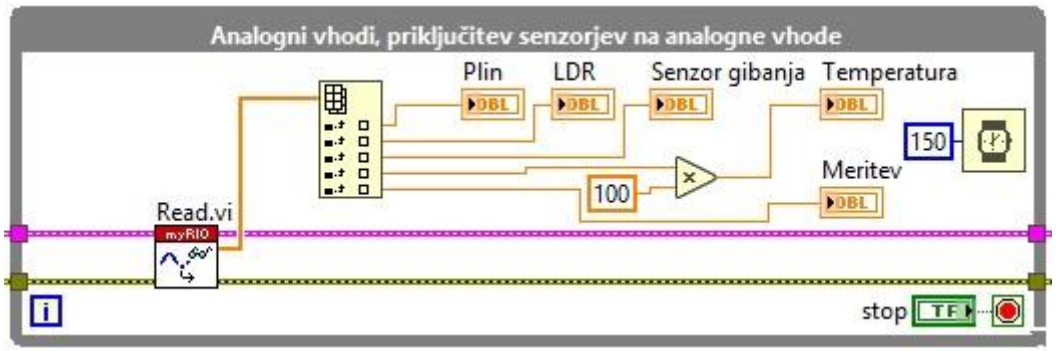
V drugem okvirju sekvence imamo nameščene zanke While. Zaradi preglednosti delovanja posameznih sklopov smo jih namestili več.

Zanka While vsebuje dva priključka, pogojni priključek (desno spodaj) in števec iteracij (levo spodaj). Na pogojni priključek moramo povezati logično vrednost (*True* ali *False*). S tem podatkom krmilimo izvajanje zanke. Pogojni priključek lahko v njegovem priročnem meniju nastavimo na vrednost *Stop if True* ali na *Continue if True*. Števec iteracij podaja informacijo o številu ponovitev zanke in teče od 0 naprej, kar pomeni, da je vrednost tega števca v prvi izvedbi zanke enaka 0.

LabVIEW izvaja zanko While tako, da najprej izvede algoritem zanke, nato pa preveri, če je izpolnjen pogoj za ponovno izvedbo zanke. Algoritem zanke While se torej izvede zagotovo vsaj 1- krat. Vsaka zanka While deluje neodvisno od izvajanja drugih zank While.

Prva zanka While:

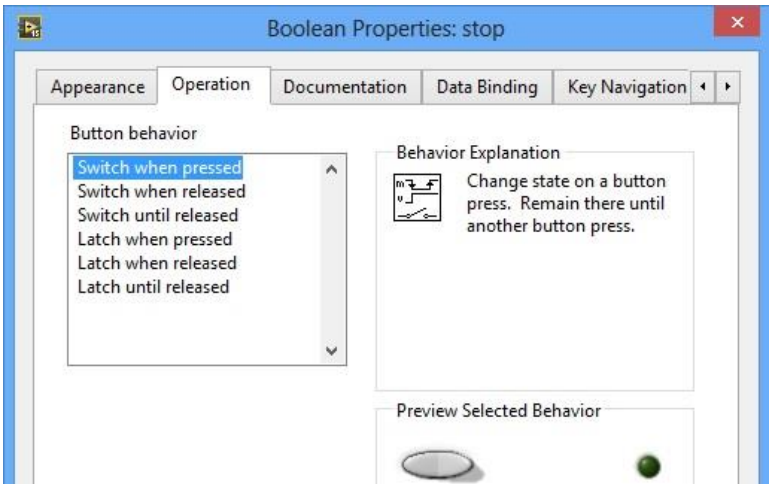
V zanko smo namestili funkcijo *Read.vi* in ji na izhodnem terminalu *Values* kreirali funkcijo *Index Array*. Ker imamo 5 vhodnih kanalov, funkcijo *Index Array* ustrezno razširimo. Vrstni red elementov funkcije *Index Array* ustreza vrstnemu redu izbranih analognih kanalov. Posameznim elementom smo kreirali indikatorje in jih ustrezno preimenovali. Indikatorje lahko kreiramo v blok diagramu ali na čelni plošči. Če jih kreiramo v blok diagramu, jih lahko spremenimo na čelni plošči (v njihovem priročnem meniju izberemo *Replace* ter v ustrezni paleti izberemo ustrezni indikator, npr. termometer). Ob vklopu virtualnega instrumenta bodo indikatorji prikazovali vrednosti na izbranih analognih vhodih. Ker uporabljamo analogni temperaturni senzor LM35, vrednost na analognem vhodu AI3 konektorja B pomnožimo s 100 zato, da bo indikator prikazoval vrednosti v °C. Analogne vhode na konektorjih A in B lahko uporabimo za merjenje signalov napetosti od 0 V do 5 V.



Slika 15: Prva zanka While (vir: avtor naloge).

Da sprostimo računalniški procesor, v zanko While namestimo časovno funkcijo *Wait (ms)* in ji na terminal *milliseconds to wait* pripeljemo vrednost 150. To pomeni, da bo LabVIEW najprej izvedel algoritm zanke *While*, nato pa bo procesor za 150 ms na voljo drugim opravilom, torej za 150 ms sprosti procesor.

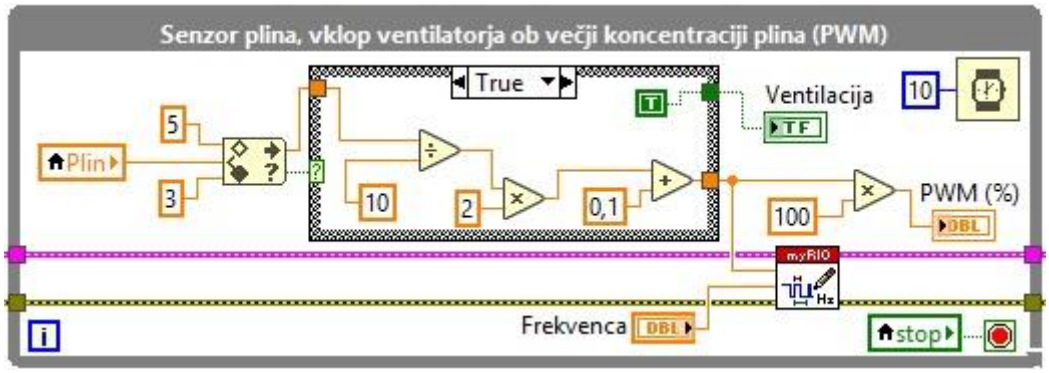
Ker bomo z enim stikalom ustavili delovanje vseh zank While, moramo v priročnem meniju gumba (stikala) stop nastaviti njegovo delovanje. Na jezičku *Operation* pod *Button behavior* izberemo *Switch when pressed* (slika 16).



Slika 16: Operacije delovanja gumba stop (vir: avtor naloge).

Druga zanka While:

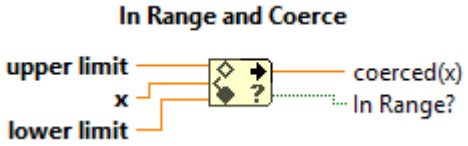
Zanka je namenjena krmiljenju ventilacije v odvisnosti od koncentracije plinov, ki jih zaznava senzor plinov.



Slika 17: Druga zanka While (vir: avtor naloge).

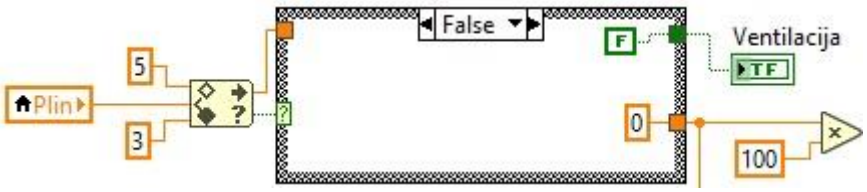
Indikatorju *Plin* smo v prvi zanki While kreirali lokalno spremenljivko (*priročni meni indikatorja Plin > Create > Local Variable*) in jo namestili v drugo zanko While. Delovanje lokalne spremenljivke smo v njenem priročnem meniju spremenili v *Change to Read*. Ventilator bomo krmilili s PWM signalom le takrat, če bo koncentracija plinov narastla na takšno vrednost, da bo na analognem izhodu senzorja plinov napetost večja od 3 V. Zato smo uporabili funkcijo *In Range and Coerce* (*Functions > Programming > Comparison > In Range and Coerce*). Vhodni terminal *upper limit* smo nastavili na 5, *lower limit* pa na 3. V tem območju bo funkcija *In Range and Coerce* na izhod *coerced(x)* posredovala vse vrednosti med 3 in 5, ki jih dobi iz izhodnega priključka senzorja plinov. Te vrednosti dobi iz

lokalne spremenljivke *Plin* na vhodni terminal *x*. Če je na vhodnem terminalu *x* vrednost med 3 in 5, se izvede okvir *True* programske strukture Case, sicer pa okvir *False*.



Slika 18: Funkcija In Range and Coerce (vir: avtor naloge).

V okviru *True* vrednosti med 3 in 5 pretvorimo v primerno vrednost duty cycle PWM signala, da se bo hitrost vrtenja ventilatorja primerno spreminjala. Paziti moramo tudi na to, da na vhod *Duty Cycle* funkcije *PWM* ne pripeljemo večje vrednosti od 1. Na ta vhod lahko pripeljemo le vrednosti od 0 do 1. Ker na vmesniku (čelna plošča) prikazujemo duty cycle PWM signala v %, od 0 % do 100 %, moramo vrednost pred indikatorjem pomnožiti s 100. Na pogojni priključek zanke While smo povezali lokalno spremenljivko gumba stop iz prve zanke While.



Slika 19: Okvir False programske strukture Case (vir: avtor naloge).

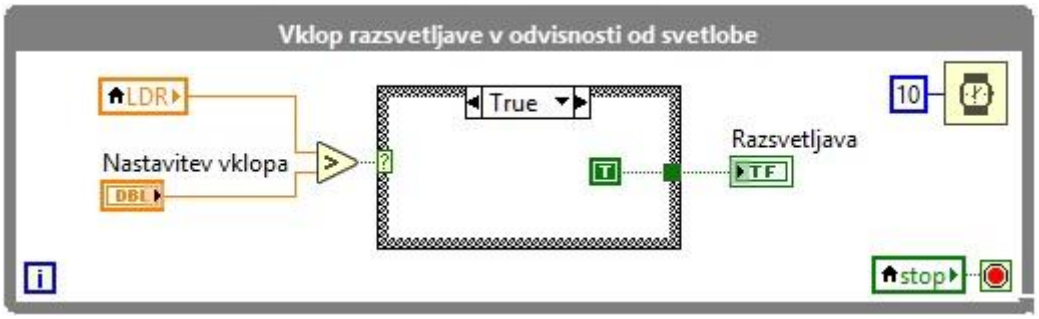
Programska struktura Case spada v skupino odločitvenih (pogojnih) stavkov. Ti stavki ponujajo možnost izbiranja različnih poti izvajanja. Struktura Case uporabljamo takrat, ko je potrebno izvesti določen del kode samo če je izpolnjen pogoj. Strukturo Case kreiramo z orodjem Case Structure, ki se nahaja na paleti Functions > Programming > Structures > Case Structure.

Privzeto se ob namestitvi strukture kreirata dva okvirja (True in False), ki sta nameščena drug nad drugim. Struktura Case vsebuje izbirni priključek (Selector Terminal) in selektor strukture (Case Selector). Okvir oziroma algoritem, ki se bo izvedel, je odvisen od podatka, pripeljanega na izbirni priključek in podatka, vnesenega v selektor strukture Case. Ko je na izbirni priključek pripeljan logični podatek kot v našem primeru, vsebuje struktura Case dva okvirja, okvir True in okvir False. Če je vhodni podatek enak True, se izvede algoritem, ki se nahaja znotraj okvirja True, v nasprotnem primeru pa algoritem znotraj okvirja False.

Tretja zanka While:

Tretja zanka While je namenjena krmiljenju razsvetljave v odvisnosti od svetlobe.

Če je vrednost napetosti na fotouporu (na analognem vhodu AI1 konektorja B) večja od izbrane napetosti (nastavljamo jo s kontrolo *Nastavitev vklopa razsvetljave (V)*), se izvede okvir *True* programske strukture *Case*, sicer pa okvir *False*. V okvirju *True* z Booleanovo konstanto *True* vklopimo razsvetljavo, v okvirju *False* pa jo z Booleanovo konstanto *False* izklopimo.



Slika 20: Tretja zanka While (vir: avtor naloge).

Fotouporu kot senzorju svetlobe se spreminja upornost v odvisnosti od svetlobe. Zaporedno z njim smo povezali upor vrednosti 10 kΩ in vezje priključili na napetost 5 V. Zaradi uporabe tega delilnika napetosti, se na fotouporu spreminja napetost v odvisnosti od svetlobe. To spremembo čuti analogni vhod AI1 na konektorju B krmilnega modula myRIO.

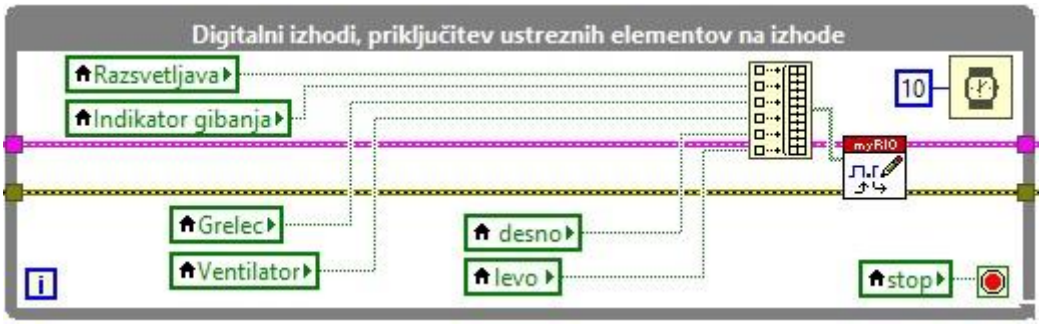
Na pogojni priključek zanke While smo povezali lokalno spremenljivko gumba stop iz prve zanke While.

Četrta zanka While:

Četrta zanka While je namenjana pošiljanju ustreznih podatkov na izbrane digitalne izhode modula myRIO.

V njej je nameščena funkcija *Write.vi*. Na njen vhodni terminal *Values* pripeljemo izhod funkcije *Build Array* (*Functions > Programming > Array > Build Array*). S funkcijo *Build Array* smo programsko kreirali polje podatkov. Na vhodne elemente funkcije *Build Array* pripeljemo vrednosti za krmiljenje posameznih digitalnih izhodov. Vrstni red elementov ustreza vrstnemu redi izbranih digitalnih izhodov. Če je npr. senzor svetlobe zaznal temo, se vklopi LED indikator *Razsvetljava*. Temu indikatorju smo kreirali lokalno spremenljivko, ki je povezana na prvi element funkcije *Build Array*. Zato bo myRIO poslal logično 1 (napetost 3,3 V) na digitalni izhod DIO0 konektorja C, saj je v eno

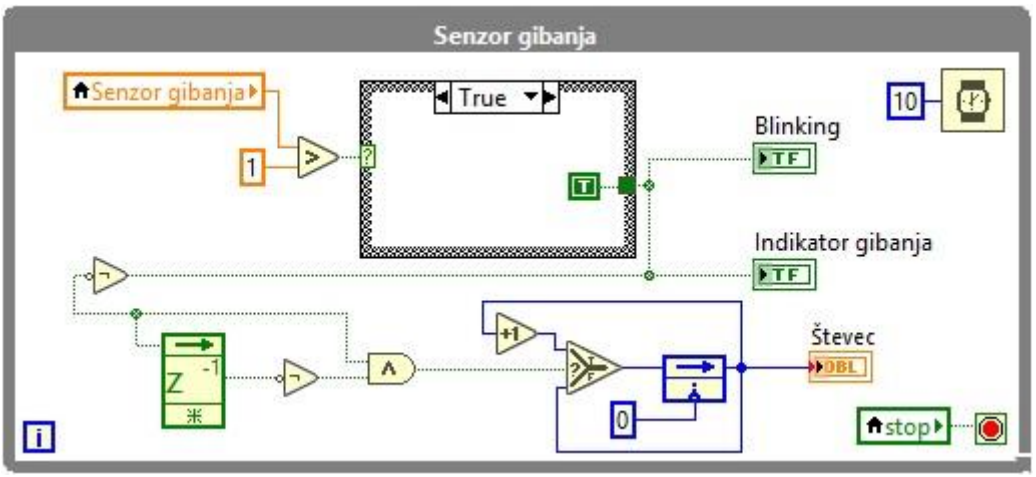
dimenzionalnem polju kreiranih digitalnih izhodov na prvem mestu. Lokalne spremenljivke ostalih indikatorjev (Indikator gibanja, Grelec, Ventilator, vrtenje DC motorčka v levo smer in vrtenje DC motorčka v desno smer) vklapljajo oziroma izklapljajo ustrezne naprave, ki so priključene na digitalne izhode modula myRIO.



Slika 21: Četrta zanka While (vir: avtor naloge).

Peta zanka While:

V peti zanki While krmilimo digitalni izhod DIO1 konektorja C v odvisnosti od tega, ali je senzor gibanja zaznal gibanje ali ne.



Slika 22: Peta zanka While (vir: avtor naloge).

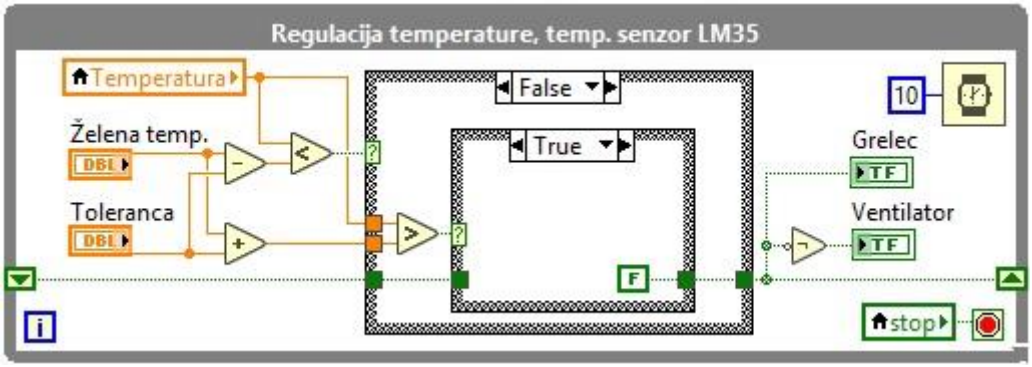
Dodali smo dve simulacijski LED diodi, pravokotno in okroglo, ki prikazujeta stanje na vmesniku. V njunem priročnem meniju smo izbrali barve in napise, ki se prikažejo ob vklopu oziroma izklopu, ko senzor gibanja zazna gibanje ali pa ga ne zazna. V programsko kodo smo dodali indikator Števec, ki

šteje število zaznanih gibanj. Koda za štetje poleg logičnih vrat vsebuje funkciji *Feedback Node*, eno za podatkovni tip Boolean, drugo pa za celoštevilčni prenos podatkov. Funkcija *Feedback Node* (*Functions* > *Structures* > *Feedback Node*) shrani podatek iz predhodne iteracije zanke While in ga vrne v naslednji iteraciji zanka While.

V kolikor senzor gibanja zazna gibanje, se izvede okvir *True* programske strukture *Case*, sicer pa okvir *False*. V okvirju *True* z Booleanovo konstanto *True* vklopimo indikator gibanja, v okvirju *False* pa ga z Booleanovo konstanto *False* izklopimo.

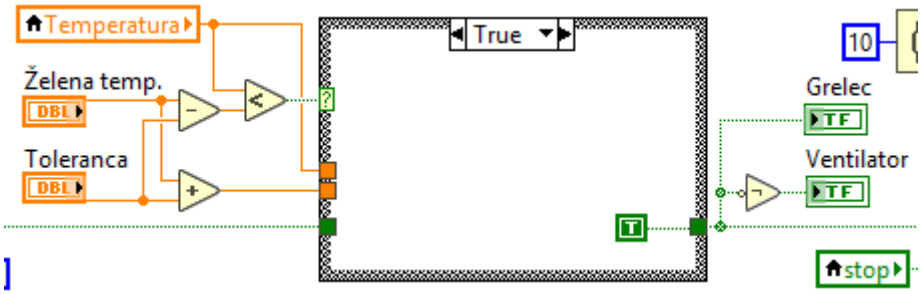
Šesta zanka While:

V šesti zanki While krmilimo temperaturo v prostoru.



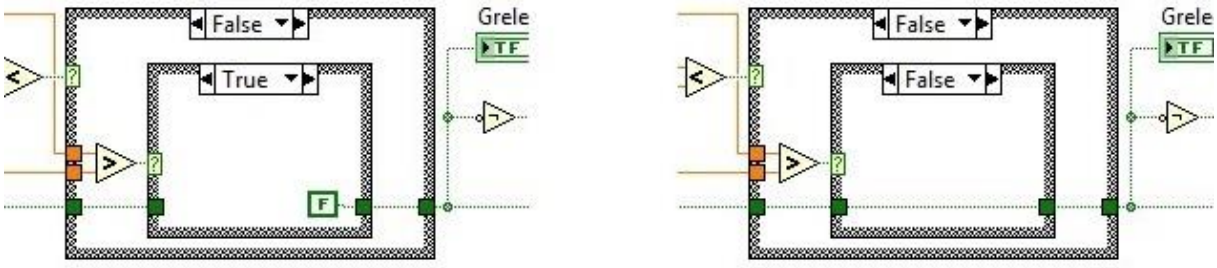
Slika 23: Šesta zanka While (vir: avtor naloge).

Če je izmerjena temperatura nižja od želene (nastavljene) zmanjšana še za toleranco, se izvede okvir *True* zunanje programske strukture *Case*. V tem okvirju z Booleanovo konstanto *True* vklopimo grelec in izven programske strukture *Case* prek negatorja izklopimo ventilator.



Slika 24: Okvir True zunanje programske strukture Case (vir: avtor naloge).

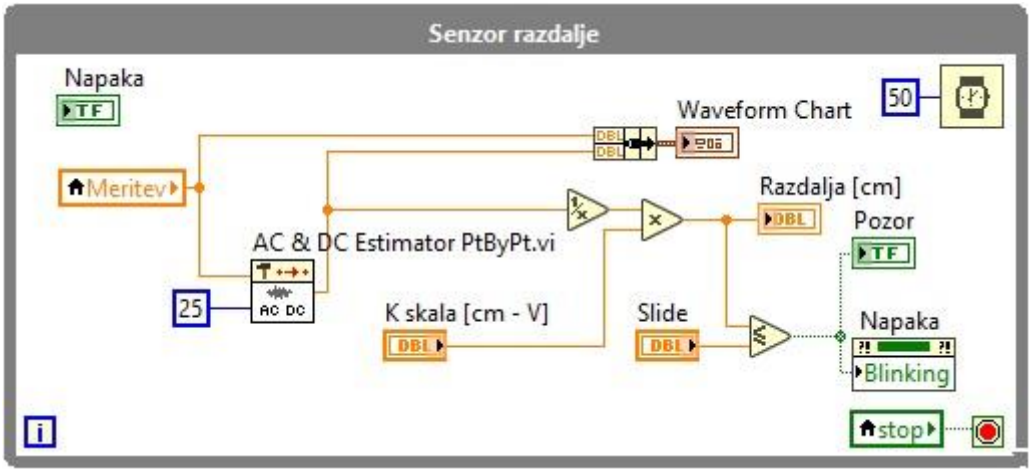
Če pa izmerjena temperatura ni manjša od želene (nastavljene) zmanjšana še za toleranco, se izvede okvir *False* zunanje strukture *Case* v katerem preverjamo, ali je izmerjena temperatura večja od želene (nastavljene) povečana še za toleranco. Če je večja, se izvede okvir *True* notranje programske strukture *Case*, v katerem z Booleanovo konstanto *False* izklopimo Grelec in zunaj obeh programskih struktur *Case* prek negatorja vklopimo ventilator. Če izmerjena temperatura ni manjša od izmerjene minus toleranca in ni večja od želene (nastavljene) povečana še za toleranco, se izvede okvir *False* notranje programske strukture *Case*.



Slika 25: Okvir False zunanje in oba okvirja notranje programske strukture Case (vir: avtor naloge).

Sedma zanka While:

V zanki While merimo in prikazujemo razdaljo od senzorja razdalje do predmeta na indikatorju razdalje in na grafu. Če je razdalja manjša od nastavljene vrednosti, se vklopi opozorilna LED dioda na vmesniku, hkrati pa začne utripati indikator z imenom Napaka.

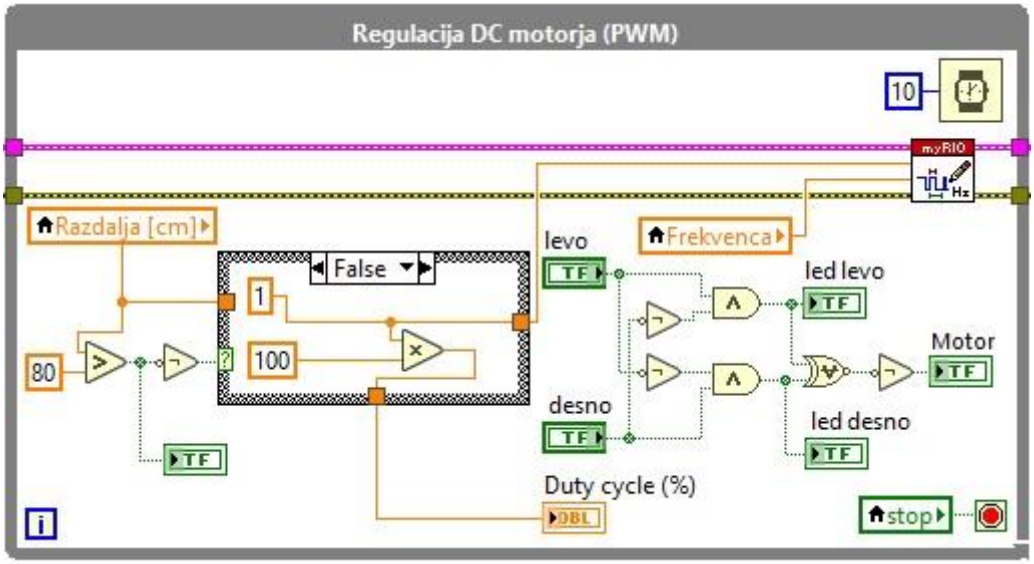


Slika 26: Sedma zanka While (vir: avtor naloge).

S funkcijo *AC & DC Estimator PtByPt.vi* (*Functions > Signal Processing > Signal Operation > AC & DC Estimator .vi*) izračunamo oceno vrednosti vhodnega signala iz niza 25-tih pridobljenih podatkov senzorja razdalje. Na grafu je prikazana amplituda trenutnih vrednosti napetosti senzorja in ocenjena vrednost. Z upoštevanjem koeficienta za uporabljen senzor razdalje dobimo na indikatorju z imenom Razdalja [cm] prikazano razdaljo predmeta od senzorja v cm.

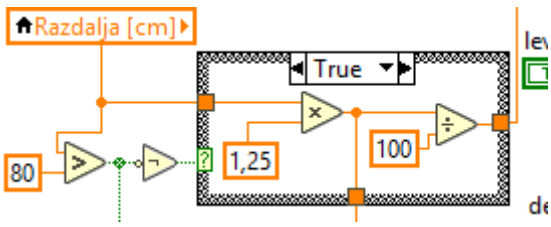
Osma zanka While:

V tej zanki krmilimo DC motorček, smer vrtenja ter hitrost vrtenja.



Slika 27: Osma zanka While (vir: avtor naloge).

Za nadzor hitrosti vrtenja uporabljamo podatek o izmerjeni razdalji predmeta od senzorja razdalje. Na podlagi tega podatka določamo duty cycle PWM signala, ki določa hitrost vrtenja motorčka. Manjša kot je razdalja, počasneje se bo vrtel DC motorček. Če je razdalja manjša od 80 cm, se izvede okvir True programske strukture Case, v nasprotnem primeru pa okvir False.



Slika 28: Okvir True programske strukture Case (vir: avtor naloge).

Ta zanka vsebuje tudi lokalne spremenljivke stikal (levo, desno), s katerimi na vmesniku določamo smer vrtenja DC motorčka. Z logičnim vezjem dosežemo, da nista obe stikali vključeni istočasno. Simulacijski LED diodi na vmesniku prikazujeta smer vrtenja DC motorčka indikator z imenom Motor pa ali DC motorček deluje ali ne deluje.

3.5.3 Tretji okvir sekvence Flat:

Ko s stikalom izklopimo izvajanje zank *While*, s funkcijami *Close.vi* prekinemo komunikacijo z vsemi analognimi vhodi, prekinemo komunikacijo s PWM izhodoma, postavimo frekvenco ter duty cycle na vrednost 0 in prekinemo komunikacijo z vsemi digitalnimi izhodi ter jih postavimo na logično stanje 0. Nato se izvede tretji okvir sekvence Flat, v katerem nastavimo privzete vrednosti:

- Želena temperatura v prostoru: 25 °C.
- Temperaturna toleranca: 1 °C.
- Nastavitev vklopa razsvetljave v odvisnosti od svetlobe: 2,5 V.
- Frekvenca PWM signala za ventilator in enosmerni motorček: 50 Hz.
- Duty cycle PWM signala za ventilator in enosmerni motorček: 0 %.
- Indikator gibanja: Prva LED => izklopljena, napis *Ni zaznanega gibanja*, druga LED => izklopljena.
- Minimalna razdalja predmeta od senzorja razdalje: 10 cm.
- Vse ostale simulacijske LED diode na vmesniku: Izklopljene.

3.5.4 Četrti okvir sekvence Flat:

V četrtem okvirju sekvence Flat resetiramo krmilni modul myRIO.

3.5.5 Vmesnik naloge Uporaba senzorjev

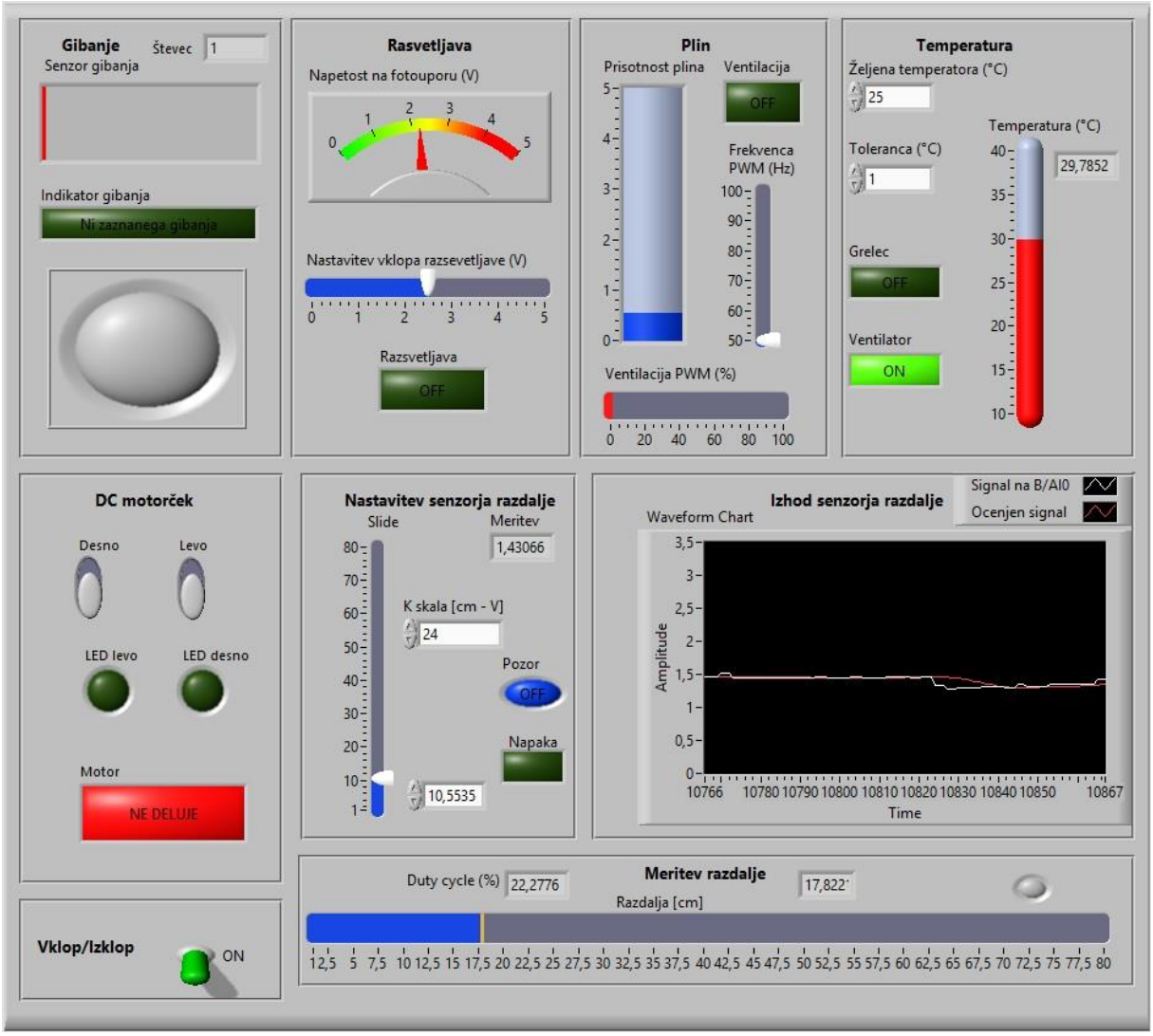
Prikaz delovanja različnih senzorjev, stanja senzorjev ter nastavitve prikazuje vmesnik na čelni plošči. Zgoraj levo je del za prikazovanje zaznavanja gibanja. Ko je gibanje zaznano, se vklopi indikator, ki nas na to opozori. Števec beleži in šteje število zaznanih gibanj, dokler se program izvaja.

Naslednjo okno nam prikazuje stanje razsvetljave, napetost na fotouporu, pod njim pa se nahaja drsnik, s katerim nastavimo vrednost, pri kateri se naj razsvetljava vklopi.

V tretjem oknu spremljamo koncentracijo plina v prostoru, zraven je drsnik, s katerim nastavljamo frekvenco delovanja ventilatorja. Pod njima se prikazuje duty cycle krmilnega PWM signala, od katerega je odvisna hitrost vrtenja ventilatorja. Ta se spreminja s koncentracijo plina. Večja kot je ta, bolj hitro bo ventilator odsesaval ta plin.

V zadnjem zgornjem delu vmesnika na čelni plošči pa nadzorujemo gretje in hlajenje prostora. Uporabnik določi želeno temperaturo in toleranco le te in program bo vklopil hlajenje (v našem

primeru ventilator za hlajenje) oziroma gretje. Temperaturni senzor bo meril temperaturo in po potrebi vklapljal gretje oziroma hlajenje.



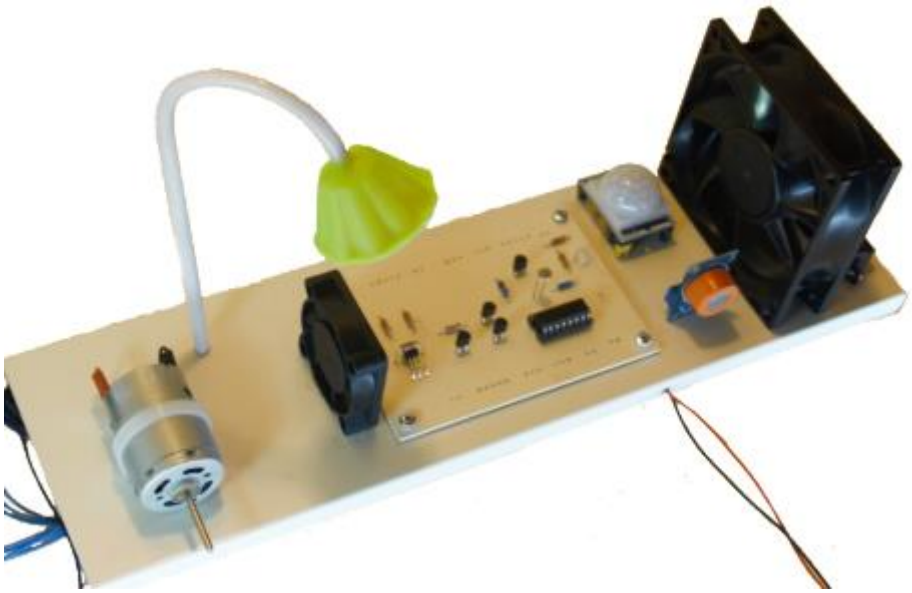
Slika 29: Vmesnik, čelna plošča (vir: avtor naloge).

V spodnjem desnem kotu se nahaja stikalo za ustavitev delovanja programa (virtualnega instrumenta). Nad njim se nahaja okvir za nadzor DC motorčka. Nameščeni sta dve stikali, s katerima določimo smer vrtenja.

Zadnji trije okvirji so namenjena meritvi razdalje in krmiljenju hitrosti vrtenja DC motorčka iz prejšnjega okvirja. V sredinskem okvirju določimo minimalno vrednost razdalje. Če je predmet bližje senzorju, nas na to opozorita indikatorja. Prikazana je tudi meritev razdalje v cm, pod njo pa umeritev skale. Desno nam graf prikazuje ocenjene in izmerjene vrednosti senzorja razdalje v odvisnosti od oddaljenosti predmeta od senzorja. V spodnjem desnem okvirju nam vmesnik prikazuje trenutno

razdaljo oddaljenosti predmeta od senzorja razdalje, indikator Duty cycle pa povprečno moč delovanja DC motorčka.

3.6 Izdelava modela



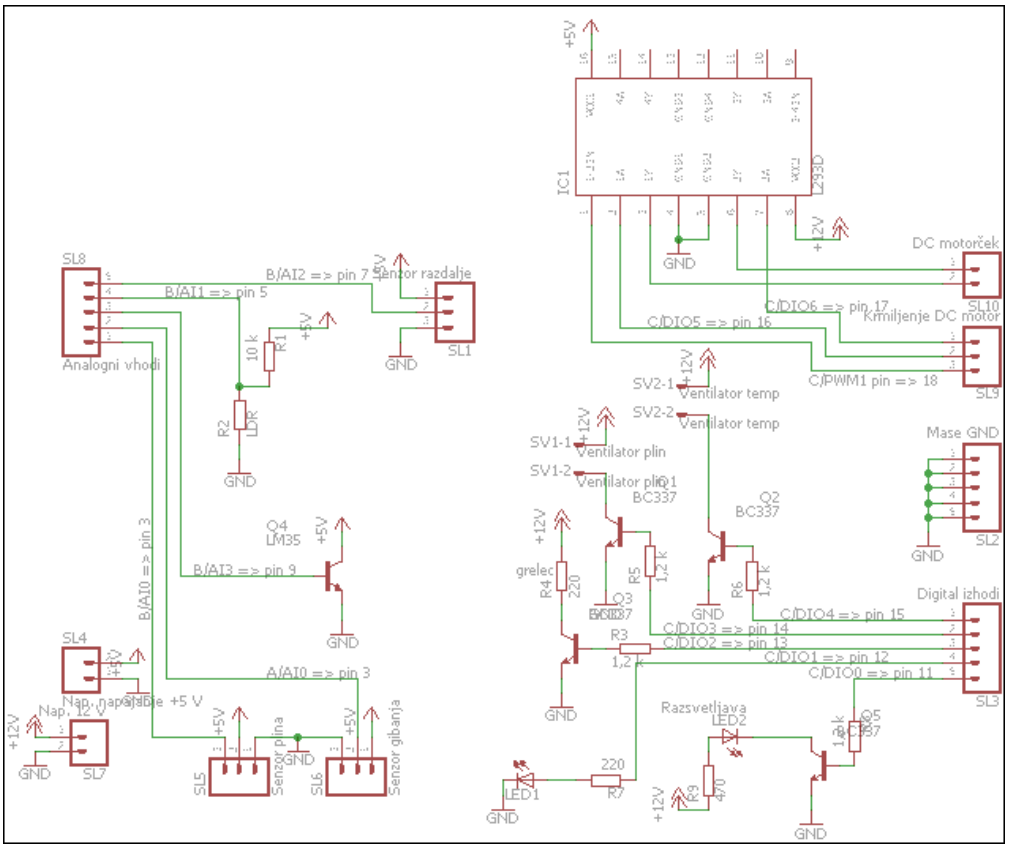
Slika 30: Model za nalogo Uporaba senzorjev (vir: avtor naloge).

Za povezavo posameznih elementov vezja smo izdelali tiskano vezje v okolju Eagle. V oknu Shematic smo izdelali elektronski načrt. Upoštevali smo podatke o krmilnem modulu myRIO glede dopustnih obremenitev digitalnih izhodov. Grelec, ventilator za prezračevanje, ventilator za hlajenje in razsvetljavo smo priključili prek bipolarnega NPN tranzistorja BC337, saj lahko digitalne izhode na konektorju C krmilnega modula myRIO obremenimo s tokom največ 4 mA.

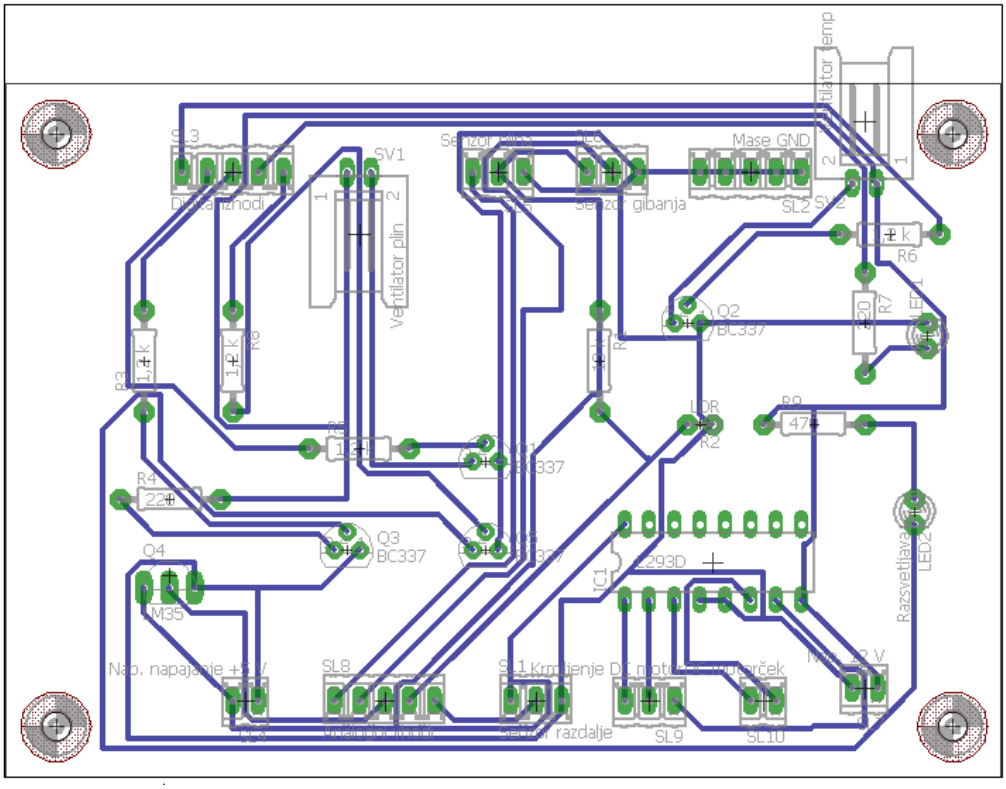
Gonilnik L293D smo uporabili za krmiljenje DC motorčka. Na priključek enable smo priključili PWM signal iz konektorja C (priključek 18, PWM1), krmilna priključka za določitev smeri vrtenja pa smo priključili na digitalna izhoda modula myRIO.

Za razsvetljavo, DC motorček in oba ventilatorja smo uporabili zunanji vir napetosti 12 V, mase pa smo vse povezali skupaj.

Na podlagi izdelane elektronske shemo smo konstruirali tiskanino. Določili smo velikost ploščice, debeline povezav in pozicije posameznih elementov. Z orodjem Autorouter smo dobili načrt tiskanine. Ploščico tiskanega vezja smo izdelali z rezkarjem. Na izdelano ploščico smo zaspajkali vse potrebne elemente in vezice za povezavo s konektorji krmilnega modula myRIO.



Slika 31: Elektronski načrt vezja (vir: avtor naloge).



Slika 32: Tiskano vezje (vir: avtor naloge).

Na kos plastike s spodnjim robom smo z lepilom in vijaki pritrdil senzorje in oba ventilatorja. Na sredino plastične podlage smo namestili tiskano vezje in nanj priključili ustrezne elemente. Z vezicami smo povezali krmilni modul myRIO z modelom in preizkusili delovanje.

4. ZAKLJUČEK

V tej raziskovalni nalogi sem se naučil veliko novega, spoznal sem programsko okolje LabVIEW, katerega prej nisem poznal. Naučil sem se uporabe krmilnega modula NI myRIO, ter spremljanja podatkov v realnem času. Izdelani model prikazuje delovanje takega sistema. Narediti je bilo potrebno dokaj zahtevno programsko kodo in vmesnik na čelni plošči, na katerem se prikazujejo ustrezni uporabni podatki.

Najprej sem za nalogo uporabil CompactDAQ s štirimi moduli, mudulom z analognimi vhodi, modulom z analognimi izhodi, modulom z digitalnimi vhodi in modulom z digitalnimi izhodi. Ko pa je šola nabavila module myRIO, sem se odločil zanj, saj ga lahko krmilim prek WiFi povezave.

Želim si, da bo ta naloga pripomogla k napredku, k uporabi takšnih sistemov v praksi in pripomogla k boljšemu in kvalitetnejšemu življenju.

5. DRUŽBENA ODGOVORNOST

S to nalogo smo predstavili delovanje nekaterih senzorjev, ki bi lahko bili uporabljeni v vsakodnevnih situacijah in nam omogočile lažje, varnejše, bolj zdravo ter manj skrbno življenje.

6. VIRI

https://www.sparkfun.com/datasheets/Components/GP2Y0A21YK.pdf (10. Dec. 2016)

https://sites.google.com/site/solaelektronikesers/home (28. Nov. 2016)

http://www.ni.com/myrio/ (12. Dec. 2016)

http://www.datasheet-pdf.download/hc-sr501-pir-motion-sensor/ (10. Dec. 2016)

http://www.miniinthebox.com/mq-3-alcohol-ethanol-sensor-module-for-arduino_p4209566.html?prm=2.5.1.1 (12. Dec. 2016)

http://henrysbench.capnfatz.com/henrys-bench/arduino-temperature-measurements/ (12. Dec. 2016)

https://www.google.si/?gws_rd=ssl#q=l293+texas+instruments (12. Dec. 2016)