»Mladi za napredek Maribora 2017«

34. srečanje

Avtomatizacija rastlinjaka z vmesnikom Arduino in LabVIEW

Raziskovalno področje: Elektrotehnika, elektronika

Raziskovalna naloga

Avtor: ROK KAMENEČKI, TIMOTEJ MERKLIN
Mentor: MILAN IVIČ
Šola: SREDNJA ELEKTRO-RAČUNALNIŠKA ŠOLA MARIBOR

Maribor, januar 2017

1. KAZALO

2. KAZALO SLIK

3. POVZETEK

Za izdelavo raziskovalne naloge smo se odločili, ko smo razmišljali o tem, kako že skoraj vse deluje s pomočjo elektronike. Odločili smo se, da bi lahko s pomočjo elektronike naredili rastlinjak, kateri bi deloval skoraj brez človeške pomoči. Rastlinjak je zasnovan tako, da ima vgrajen senzor za merjenje vlage zemlje. Kadar zemlja ni dovolj vlažna, se aktivira ta senzor in posledično vklopi črpalko, katera dovede vodo do določene mere in takrat jo senzor tudi izklopi. Za zalivanje bomo uporabili deževnico, saj je zaradi njene sestave najbolj primerna in potrebujemo samo rezervoar za hrambo, hkrati pa je to tudi ceneje od vode iz vodovodnega omrežja, prav tako pa je okolju prijaznejše. Rastlinjak ima vgrajen tudi senzor temperature. Za regulacijo temperature smo uporabili senzor LM35, kateri bo pošiljal podatke v naš program in bo posledično vklopil grelec, ki bo segreval rastlinjak. Ko pa bo temperatura previsoka, se bo vklopil ventilator, kateri bo temperaturo znižal na primerno temperaturo za rastline, da bodo lahko optimalno uspevale.

4. ZAHVALA

Zahvalili se bi našemu mentorju za pomoč, nasvete in čas katerega nam je posvetil, še posebej za gradivo in potrebne materiale, prav tako pa tudi našim sošolcem, kateri so nam pomagali in nas spodbujali.

5. UVOD

To raziskovalno nalogo smo si izbrali, saj se vsak dan srečujemo z veliko opravki in lahko zato pozabimo na rastline, katere moramo zalivati, ter za katere moramo regulirati določeno temperaturo, da lahko uspevajo. Za to bomo naredili rastlinjak z regulacijo temperature in tudi avtomatizirali zalivanje rastlin, da nam ne bo treba tega delati ročno, ampak bo to nalogo opravljal program, ki smo ga izdelali. V programu bomo lahko sproti spremljali vklop in izklop vodne črpalke, grelca ter ventilatorja. Temperaturo in vlažnost zemlje bomo lahko tudi poljubno nastavljali.

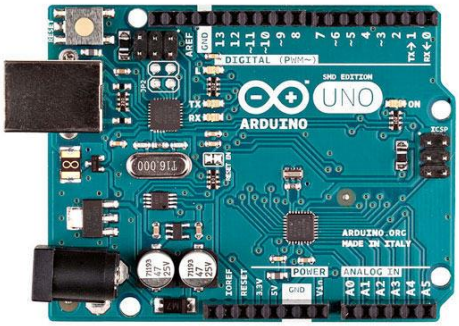
6. HIPOTEZE

- Izdelava načrta
- Izdelava programa za regulacijo
- Priprava vezalnega načrta

- Izdelava rastlinjaka

7. UPORABLJENA OPREMA

7.1. Razvojna ploščica Arduino UNO



Slika 1: Arduino UNO(vir: Avtor naloge).

Arduino UNO je razvojna ploščica, katera temelji na procesorju ATmega328P. Ima 14 digitalnih vhodov in izhodov, od tega jih je lahko 6 uporabljenih kot PWM izhodov (pulzno širinska modulacija), 6 analognih vhodov, takt pa narekuje 16 MHz kristalni oscilator. Delovna napetost te ploščice je 5 V, katero dobiva preko USB povezave z računalnikom. Zaradi tega lahko posamezne priključene elemente napajamo kar preko ploščice, vendar moramo biti pazljivi na preobremenjenost ploščice, katera znaša pri vhodnih in izhodnih priključkih 20 mA, pri 3,3 V priključkih pa 50 mA. Ploščica vsebuje 32 KB flash pomnilnika, na katerega se naloži program. Vsebuje pa tudi 2 KB RAM pomnilnika (bralno-pisalni pomnilnik) in 1 KB EEPROM pomnilnika (električno zbrisljiv in programirljiv bralni pomnilnik)

7.2. LabVIEW

Slika 2: Logotip LABVIEW (vir: https://upload.wikimedia.org).

Programsko okolje LabVIEW (Laboratory Virtual Instrumentation Engineering Workbench) proizvajalca National Instruments (NI) je v celoti grafično zasnovano programsko okolje, katero 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 blokovno orodje, kar pomeni, da se algoritem v tem programskem jeziku kodira blokovno oziroma z uporabo funkcij v obliki blokov, kateri so med seboj povezani s povezavami. Okolje LabVIEW vsebuje tudi Application Builder, ki omogoča kreiranje izvršne (exe) verzije, ki se lahko izvaja na računalniku, kjer razvojnega sistema LabVIEW ni nameščenega.

Ob zagonu programa LabVIEW se pojavi okno, kjer lahko izbiramo med kreiranjem novega projekta ali odpiranjem obstoječih, shranjenih projektov. Novi virtualni instrument (VI) kreiramo z izbiro: File > New VI. Ob tem se pojavita dve okni:

- Čelna plošča (Front Panel)
- Blok diagram (Block Diagram)

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 načina: Preko izbire menija View > Controls Palette ali pa kjerkoli v praznem prostoru Čelne plošče, če kliknemo desno tipko miške.

Paleta kontrol in indikatorjev je sestavljena iz več pod palet. Objekt 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 (Terminal). Priključki služijo za prenos podatkov med Čelno ploščo in Blok diagramom. Ime priključka v Blok diagramu je enako, kot je ime objekta na Čelni plošči. Iz oblike priključka lahko razberemo, ali pripada kontroli ali indikatorju.

Vhodni objekti, ki jim pravimo kontrole (Controls), 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 (Indicators), pa služijo za prikaz podatkov. Med izvajanjem virtualnega instrumenta se izračunani podatki posredujejo indikatorjem, ki jih nato prikažejo v tekstovni ali grafični obliki. 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 kliknemo 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.

Če želimo povezati razvojno ploščico Arduino s programskim okoljem LabVIEW , moramo naložiti datoteko LIFA (LabVIEW Interface for Arduino). Za komunikacijo LabVIEW z Arduino preko serijskega (USB) porta moramo imeti instalirane gonilnike NI-VISA (Virtual Instrument Software Architecture). Do datoteke LIFA pridemo tako, da odpremo okolje Arduino in v menijski vrstici izberemo Datoteka > Odpri... > Lokalni disk (C:) > Programske datoteke (x86) > National Instruments > LabVIEW (2014) > vi.lib > LabVIEW Interface for Arduino > Firmware > LIFA_Base. Preden naložimo program LIFA_Base moramo izbrati pravilna vrata, v našem primeru COM 3. Program LIFA_Base zapišemo v Arduino razvojno ploščo. Ko ga zapišemo, lahko okolje Arduino zapremo, saj je program za kominiciranje z LabVIEW zapisan v Arduino ploščico.

7.3. Temperaturni senzor LM35

Slika 3: Temperaturni senzor LM35 (vir: http://www.learningaboutelectronics.com).

Temperaturni senzor LM35 je integrirano vezje, katerega uporabljamo za merjenje temperature. Ta senzor deluje v temperaturnem območju od -55 °C do +150 °C. Napetost na izhodu se spreminja za 10 mV/°C in je linearno odvisna od temperature. Senzor ima tri priključke in sicer je prvi priključek za napajanje senzorja (+5 V), sredinski oz. drugi priključek za analogni izhod, katerega priključimo na analogni vhod razvojne ploščice Arduino Uno, tretji priključek pa je za maso.

7.4. Senzor vlage proizvajalca FUNDUINO

Senzor FUNDUINO uporabljamo za merjenje vlage. Ima tri kontakte in sicer pozitiven, negativen, kakor tudi analogni izhod. Kadar je senzor v suhem stanju, analogen izhod ne oddaja signala, z večanjem vlage pa se sorazmerno vrednost signala povečuje.

Slika 4: Senzor vlage (vir: http://funduino.de).

7.5. Črpalka BWV 04

Črpalka je uporabljena za črpanje vode iz rezervoarja v cevi, ki so prezentirane kot namakalni sistem. Črpalka deluje pri napetosti 12 V, njena poraba enosmernega toka znaša 0,9 A do 1,7 A.

Slika 5: črpalka MWV 04 (vir: https://www.conrad.de).

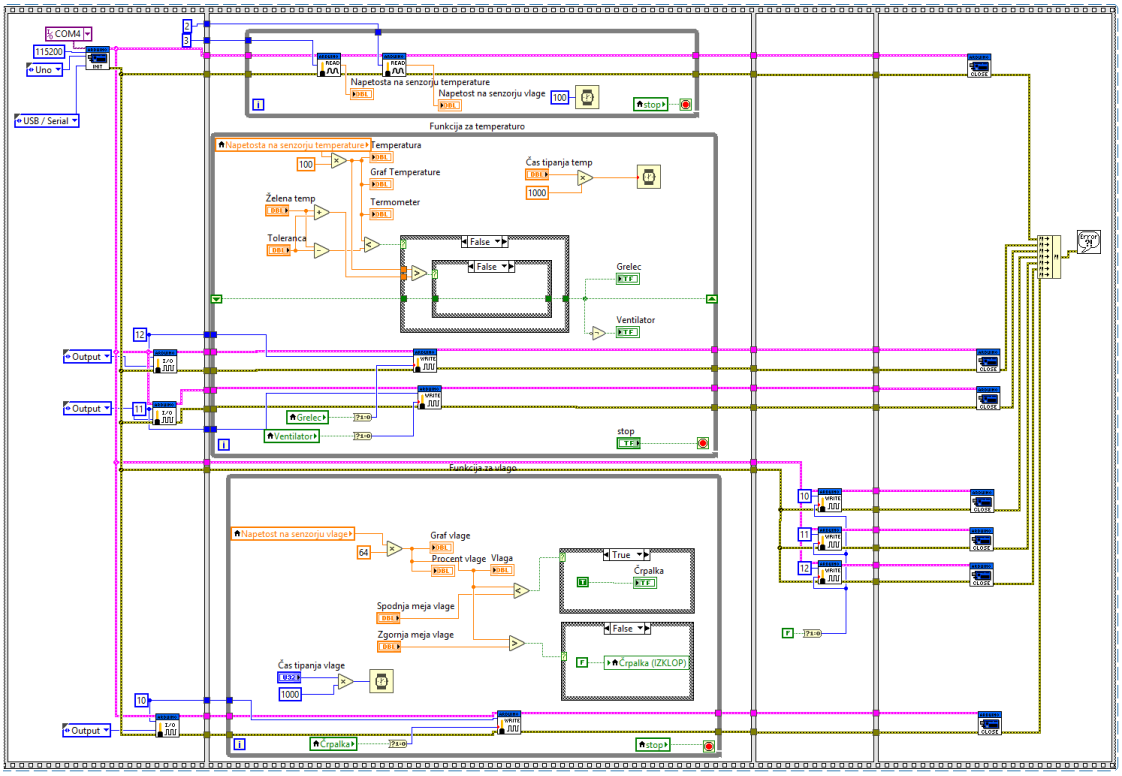
7.6. Ventilator Akasa AK-5010MS 50 mm

Ventilator je v modelu prikazan kot sistem za hlajenje. Ima 12 V napajanje in tri priključke, dimenzije pa so 50×50×10 mm.

Slika 6: Ventilator Akasa (vir: https://www.mimovrste.com).

8. PROGRAM

Izdelali smo program, ki je zmožen brati dva analogna vhoda in sicer A2 (senzor vlage) in A3 (senzor temperature). Pridobljene podatke nato pretvori v želeno količino, kasneje pa glede na vrednost posameznega senzorja krmili digitalne izhode D12 (grelec), D11 (ventilator) in D10 (črpalka). Kadar zaustavimo program s pritiskom na gumb STOP, se tudi vsi izhodi postavijo na vrednost logične 0, da preprečimo neželeno delovanje.



Slika 7: Celoten program v LabVIEW (vir: Avtor naloge).

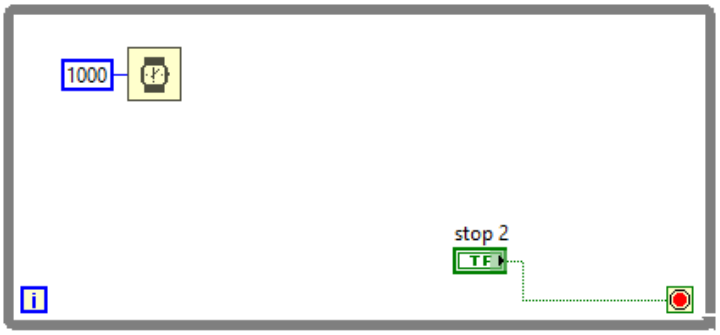
8.1. Opis uporabljenih struktur

While zanka:

Zanka While vsebuje dva priključka, pogojni priključek (desno spodaj) in števec ponovitev (levo spodaj). Na pogojni priključek moramo povezati logično vrednost. S tem podatkom krmilimo izvajanje zanke. Števec ponovitev 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. Koda v zanki While se torej izvede zagotovo vsaj 1- krat.

Na pogojni priključek zanke While smo povezali kontrolni podatek logičnega tipa, tipko stop, s pomočjo katere lahko ustavimo izvajanje zanke While.



Slika 8: While zanka (vir: Avtor naloge).

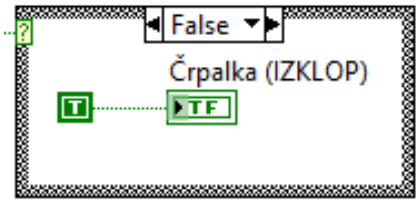
Case struktura:

Programska struktura Case spada v skupino odločitvenih (pogojnih) stavkov. Ti stavki ponujajo možnost izbiranja različnih poti izvajanja. Struktura Case se uporablja, ko je potrebno izvesti določen del kode samo če je izpolnjen pogoj.

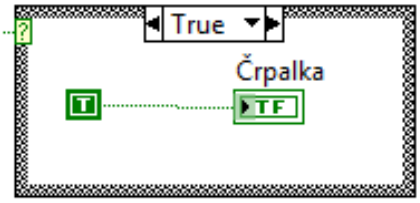
Privzeto se ob namestitvi strukture kreirata dva okvirja (True in False), ki sta nameščena drug nad drugim. Struktura Case vsebuje izbirni priključek in selektor strukture.

Okvir oziroma koda, ki se bo izvedla, je odvisena od podatka, pripeljanega na izbirni priključek in podatka, vnesenega v selektor strukture.

Če 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 koda, ki se nahaja znotraj okvirja True, v nasprotnem primeru pa koda znotraj okvirja False.



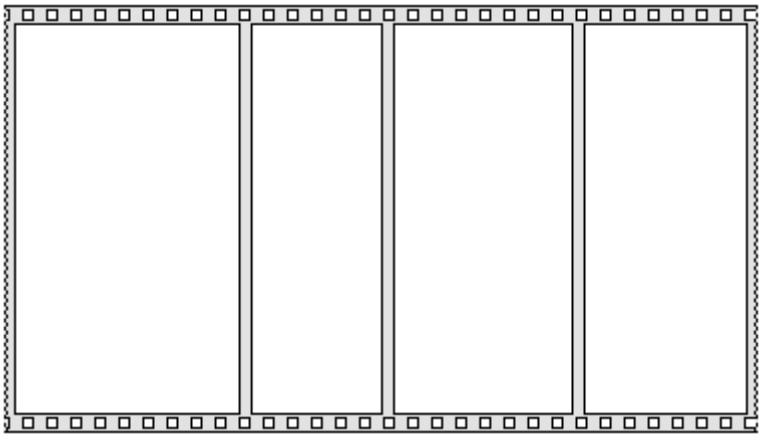
Slika 9: Case struktura v False (vir: Avtor naloge).



Slika 10: Case struktura v True (vir: Avtor naloge).

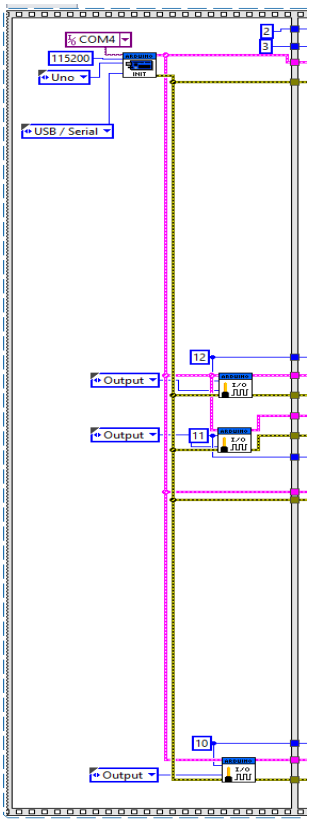
Flat sequence:

Če želimo, da se program izvaja po določenem zaporedju, uporabimo flat sequence. Izvajanje si sledi iz levega okvirja proti desnemu. Okvirjev lahko uporabimo toliko, kolikor jih potrebujemo. Ko se proces v podanem koraku zaključi, prestopi na naslednjega. S tem dosežemo, da se le želen del programa izvaja, ne pa celoten, kot na primer v našem programu, kjer se začne z inicializacijo ploščice Arduino UNO. Tu se definira tip ploščice, prav tako pa se postavijo želeni izhodi, na katere kasneje pošiljamo signale. Sledi osrčje programa, v katerem se izvajajo meritve, branje vhodnih podatkov in regulacija izhodov glede na želeno vrednost. Ta korak se bo zaključil po pritisku na gumb STOP. S tem se zaključi ta korak in preide na naslednjo okvir, v katerem postavimo vse izhode na vrednost logične 0, da bi preprečili neželeno delovanje naprav po prekinitvi oziroma izklopu programa. Nato se program zaključi, prav tako pa izpišejo morebitne napake v delovanju.



Slika 11: Izgled flat sekvence (vir: Avtor naloge).

8.2. Inicializacija

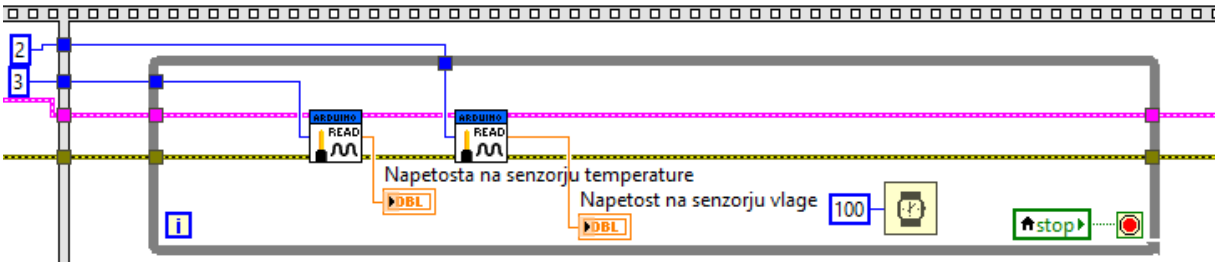


Program se začne z inicializacijo, v kateri izberemo vrata, na katera bo priključen Arduino UNO. Določimo še konstanto 'Baud' (hitrost komunikacije), izberemo tip Arduino plošče, v našem primeru je to Arduino UNO, nato pa določimo še tip povezave med Arduino UNO in računalnikom. Nadaljuje se z postavljanjem izhodov. Izbrali smo si digitalne izhode 10, 11 in 12 na ploščici Arduino.

Slika 12: Prvi korak programa (vir: avtor naloge).

8.3. Branje senzorjev

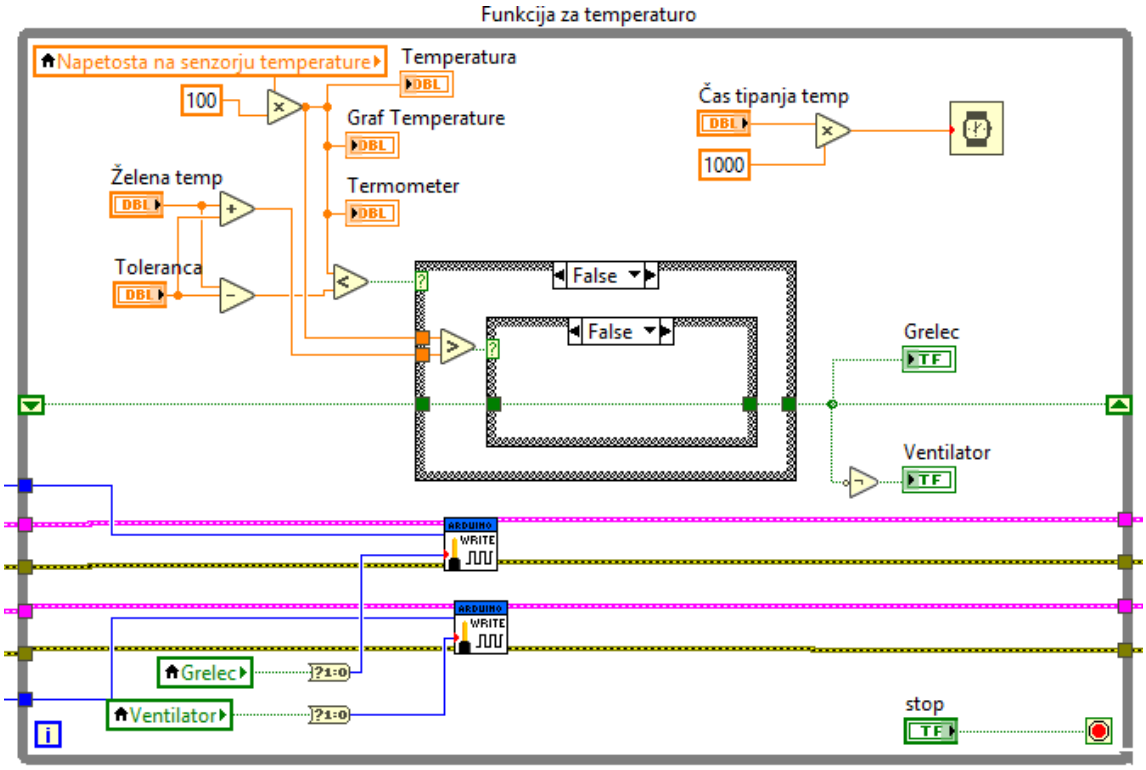
Ta del programske kode predstavlja samo branje dveh senzorjev. Tukaj sta izbrana dva vhoda, to sta analogna vhoda Arduino UNO A2 in A3. V zanki je nastavljen čas tipanja (ponavljanja zanke), ki znaša 100 milisekund. Dodatno sta priključena dva indikatorja, ki prikazujeta vrednosti obeh senzorjev. Uporabljena sta kot vir podatkov, s katerimi manipuliramo v ostalih zankah.



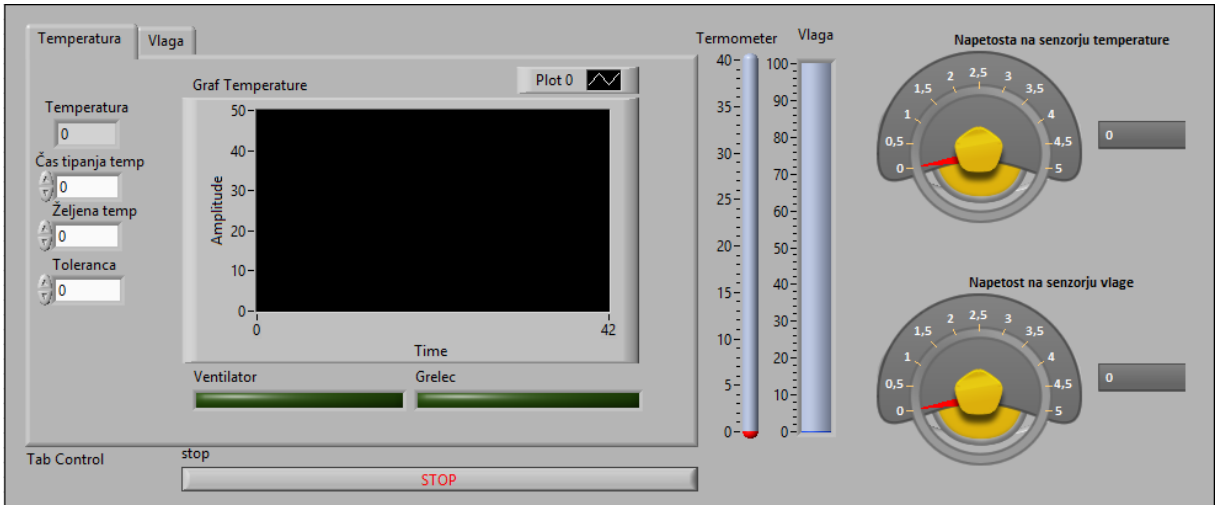
Slika 13: Nastavitev senzorjev (vir: Avtor naloge).

8.4. Koda za regulacijo temperature

Koda se začne z indikatorjem napetosti na izhodnem priključku senzorja temperature LM35, katere vrednost množimo s 100, saj senzor temperature podaja signal v mV. Senzorju se namreč poveča izhodna napetost za 10 mV na stopinjo Celzija. Nato sledi prikaz temperature na grafu, indikatorju s točno vrednostjo in indikatorju ki prikazuje termometer. V programski kodi sta uporabljeni dve kontroli z imeni Želena temperatura in Toleranca, s katerima nastavljamo željeno temperaturo in pa dovoljeno odstopanje. Če je željena temperatura nastavljena na vrednost 25 °C in toleranca na vrednost 3, se bo ventilator vklopil pri 28 °C, grelec pa pri 22 °C. V tej kodi sta uporabljeni dve programski strukturi Case, ki na podlagi podatkov, pripeljanih na izbirna priključka vklapljata/izklapljata grelec oziroma ventilator. Nastavljamo lahko čas tipanja temperature, če je ta 1 s, se bo vrednost temperaturnega senzorja posodobila vsako sekundo, rezultati pa prikazali na ustreznih indikatorjih. Vsebuje še tudi dve LED diodi, ki na vmesniku na čelni plošči prikazujeta delovanja ventilatorja oziroma grelca.



Slika 14: Regulacija temperature (vir: Avtor naloge).

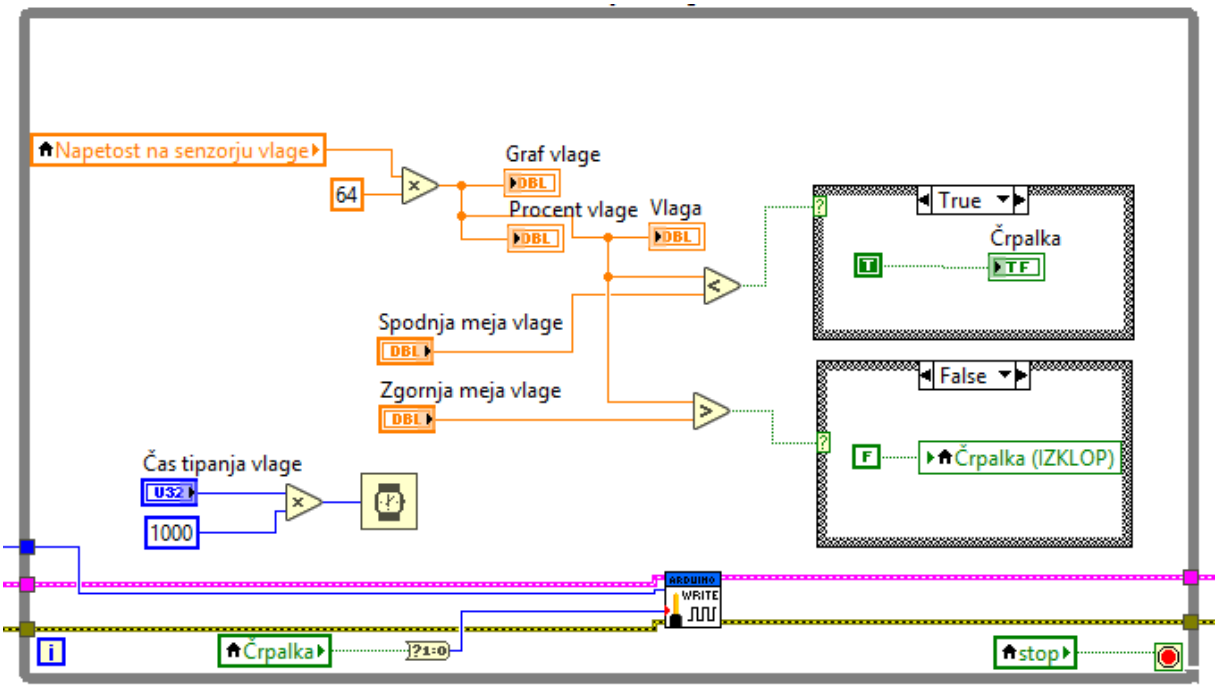


Slika 15: Krmilni prikaz za regulacijo temperature (vir: Avtor naloge).

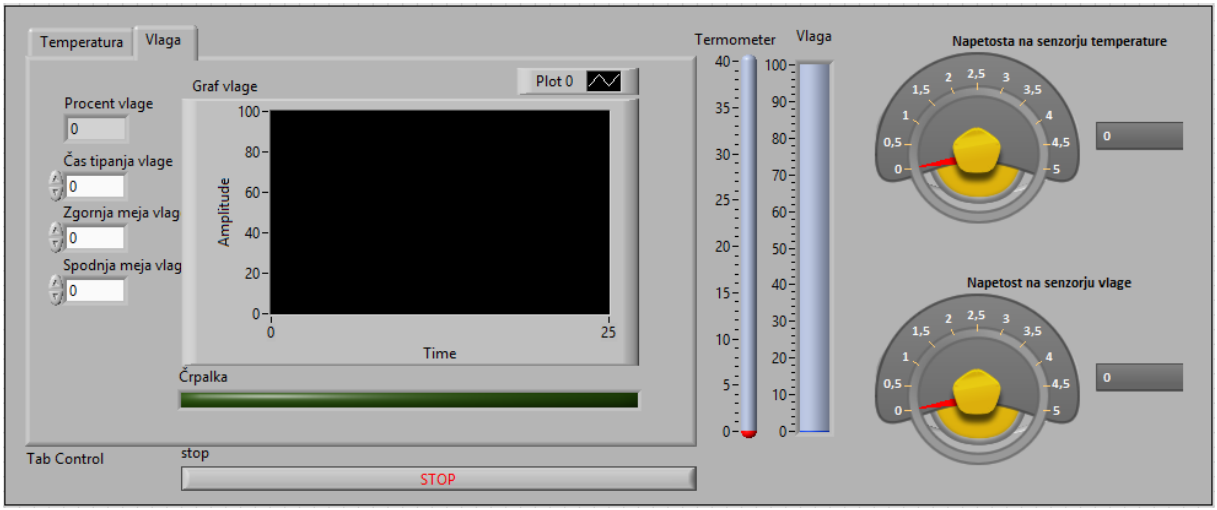
8.5. Koda za regulacijo vlage

Koda se začne z indikatorjem napetosti na senzorju vlage, katerega vrednost množimo s 64, da dosežemo v vrednost vlage 100 %, če je senzor vlage potopljen v vodo. Ustrezni indikatorji prikazujejo relativno vlago in izhodno napetost senzorja vlage. Z ustreznimi kontrolami lahko nastavimo spodnjo mejo vlage, ko se bo črpalka vklopila in zgornjo mejo,

ko se bo črpalka izklopila. Če je spodnja meja vlage 30 % in zgornja meja 50 %, se bo črpalka vklopila, ko bo trenutna vlažnost zemlje padla pod 30 %. Voda se bo dolivala, dokler vlažnost zemlje ne doseže 50 %, pri tej vrednosti se črpalka izklopi. Ko vrednost vlage ponovno pade pod spodnjo mejo, se črpalka ponovno vklopi. Prav tako lahko nastavimo čas tipanja, enako kot pri regulaciji temperature. LED dioda na vmesniku se vklopi, ko črpalka deluje.

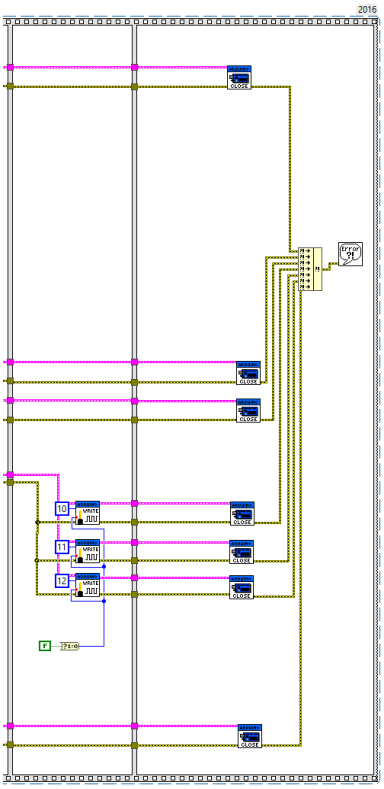


Slika 16: Regulacija vlage v zemlji (vir: Avtor naloge).



Slika 17: Krmilni prikaz za regulacijo vlage (vir: Avtor naloge).

8.6. Končne nastavitve



Ta stopnja se izvrši po pritisku na gumb Stop, ki prekine izvajanje zank While. Ventilator, grelec in pa črpalka se morajo izklopiti, če so pred zaustavitvijo delovali. To je narejeno zato, da na primer grelec ne bi pregreval rastlinjaka, če smo virtualni instrument, program izklopili. Ko se izvajanje zanke While ustavi, s Close.vi prekinemo delovanje vseh digitalnih vhodno/izhodnih kanalov, digitalne izhode postavimo na logično stanje 0 in onemogočimo digitalne izhode.

Če se zgodi kakšna napaka, nas bo program o tem obvestil in izpisal problem oziroma samo napako.

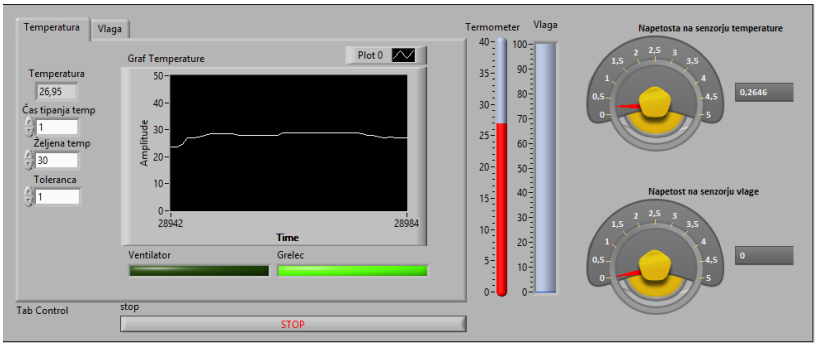
Slika 18: Zadnji korak programa (vir: Avtor naloge).

9. METODOLOGIJA DELA

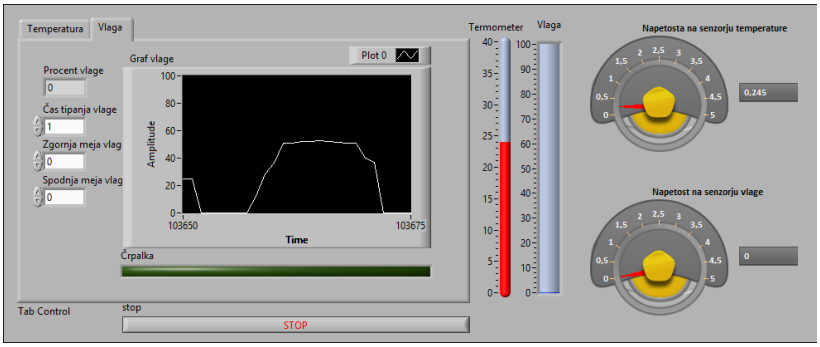
Delo se je začelo z izdelavo samega programa v okolju LabVIEW. Hkrati smo imeli priključeno ploščico Arduino UNO. Na analogne vhode smo priključili ustrezne senzorje, na digitalne izhode pa LED diode, ki so simulirale delovanje grelce, ventilatorja in črpalke. Po načelu od lažjega k težjemu smo gradili programsko kodo v okolju LabVIEW. Po večkratnem popravljanju kode in ugotavljanju vzrokov napak, ki so se pojavljale, smo prišli do končne verzije programa, ki je deloval po naših željah. Sledila je izdelava načrta za rastlinjak in dobava potrebnega materiala. Naloga se je končala s sestavo komponent in ponovnim testiranjem.

10. REZULTATI

Pri meritvah je prikazano samo delovanje senzorjev in svetlobni prikaz delovanja črpalke, grelca in ventilatorja.



Slika 19: Prikaz temperature (vir: Avtor naloge).



Slika 20: Prikaz vlage (vir: Avtor naloge).

11. DRUŽBENA ODGOVORNOST

Projekt je v veliko pomoč vsakemu, ki si želi doma pridelovati ali gojiti različne rastline, saj v večini primerov posameznikom primanjkuje časa za ukvarjanje z rastlinami. Naloga zaradi svoje avtomatizacije in enostavne uporabe predstavlja preprosto rešitev krmilja, saj neprestano uravnava temperaturo in vlago, ki jo lahko poljubno določimo, odvisno od potrebe posameznika, njegovega okolja in gojenih rastlin. Je tudi okolju prijazna, saj zbira deževnico za zalivanje rastlin, tako da ne le prihranimo le na ceni za vodo, temveč tudi pripomoremo k manjšemu onesnaževanju okolja. V primeru da primanjkuje deževne voda, se začne dolivati voda iz mestnega vodovoda.

12. ZAKLJUČEK

Kot naloga se je izkazala avtomatizacija rastlinjaka kar zahteven izziv, za katerega je bilo potrebno mnogo testiranja in mnogo preizkušanja različnih variant programske kode. Kljub vsem nastalim težavam je iz enostavne zamisli nastal projekt, ki ga je možno realizirati v večjih merilih, kot je primer domače uporabe za enostavno gojenje najrazličnejših rastlin.

13. VIRI

Literatura:

Arduino [ONLINE]. 2017,
<https://en.wikipedia.org/wiki/Arduino>

Conrad [ONLINE]. 2017,
<https://www.conrad.de/de/niedervolt-tauchpumpe-barwig-0444-600-lh-6-m-539090.html>

Funduino [ONLINE]. 2017,
<http://funduino.de/nr-17-tropfensensor>

Learning about Electronics [ONLINE]. 2017,
<http://www.learningaboutelectronics.com/Articles/LM35-temperature-sensor-circuit.php>

Mimovrste [ONLINE]. 2017,
<https://www.mimovrste.com/pc-modding-in-hlajenje/akasa-ventilator-za-ohisje-akasa-ak-5010ms-50-mm>

Šola elektronike SERŠ [ONLINE]. 2016, Milan Ivič
<https://sites.google.com/site/solaelektronikesers/home>

Wikipedia [ONLINE]. 2017,
<https://en.wikipedia.org/wiki/LabVIEW>

Viri slik:

Slika 2, Labview logo

<https://upload.wikimedia.org/wikipedia/en/3/38/Labview-logo.png>

Slika 3, senzor temperature LM35

<http://www.learningaboutelectronics.com/Articles/LM35-temperature-sensor-circuit.php>

Slika 4, senzor vlage Funduino

<http://funduino.de/nr-17-tropfensensor>

Slika 5, vodna črpalka BWV 04

<https://www.conrad.de/de/niedervolt-tauchpumpe-barwig-0444-600-lh-6-m-539090.html>

Slika 6, Akasa ventilator

<https://www.mimovrste.com/pc-modding-in-hlajenje/akasa-ventilator-za-ohisje-akasa-ak-5010ms-50-mm>