Realizacija sustava za automatsko zalijevanje biljke

Tretnjak, Karlo

Undergraduate thesis / Završni rad

2015

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:445329

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

University North Digital Repository

Sveučilište
Sjever

Završni rad br. 359/EL/2015

Realizacija sustava za automatsko zalijevanje biljke

Karlo Tretnjak, 5737/601

Varaždin, rujan 2015. godine

Sveučilište
Sjever

Odjel za elektrotehniku

Završni rad br. 359/EL/2015

Realizacija sustava za automatsko zalijevanje biljke

Student

Karlo Tretnjak, 5737/601

Mentor

Miroslav Horvatić, dipl. ing

Varaždin, rujan 2015. godine

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

UNIVERSITY NORTH

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL	Odjel za elektrotehniku		
PRISTUPNIK	Karlo Tretnjak	MATIČNI BROJ	5737/601
DATUM	09.09.2015		
KOLEGIJ	Automatsko upravljanje		
NASLOV RADA	Realizacija sustava za automatsko zalijevanje biljke		
MENTOR	Miroslav Horvatić dipl. ing.	ZVANJE	predavač

ČLANOVI POVJERENSTVA

1. mr.sc. Ivan Šumiga, dipl. ing.
2. Dunja Srpak, dipl. ing.
3. Miroslav Horvatić, dipl.ing.

Zadatak završnog rada

BROJ 359/EL/2015

OPIS

Osmisliti i realizirati sustav za automatsko zalijevanje biljke. Sustav treba imati mogućnost ručnog i automatskog načina rada. U ručnom načinu rada zalijevanjem upravlja operater. U automatskom načinu rada zalijevanje se vrši kada vlažnost tla biljke padne ispod određene razine. Sustav treba omogućiti automatsku regulaciju temperature tekućine kojom se vrši zalijevanje. Upravljanje sustavom za automatsko zalijevanje realizirati korištenjem Arduino DUE razvojne platforme.

U radu je potrebno:
- osmisliti sustav za automatsko zalijevanje biljke,
- opisati pojedine komponente sustava i princip rada sustava,
- realizirati i objasniti sklopovski dio sustava korištenjem Arduino DUE razvojne platforme,
- realizirati i objasniti programski dio sustava upravljanja,
- isprobati i dokumentirati funkcioniranje sustava upravljanja.

REPUBLIKA HRVATSKA 4 SVEUČILIŠTE SJEVER KOPRIVNICA

ZADATAK URUČEN 15.09.2015

POTPIS MENTORA

Predgovor

Zahvaljujem mentoru Miroslavu Horvatiću, dipl. ing, na velikoj i nesebičnoj pomoći te praćenju procesa nastajanja ovog završnog rada. Hvala što me svojim savjetima i entuzijazmom prema poslu kojim se bavi usmjeravao kako da uspješno izvršim sve zadatke završnog rada.

Zahvaljujem se profesorima Sveučilišta Sjever u Varaždinu, koji su se trudili prenijeti što više znanja na predavanjima i laboratorijskim vježbama, koje mi je uvelike pomoglo kod izrade ovog završnog rada.

Također veliko hvala mojoj obitelji koja mi je omogućila obrazovanje na Sveučilištu Sjever u Varaždinu te me nesebično podržavala u toku obrazovanja. Zahvaljujem i svima koji su mi na bilo koji način pomogli oko obrazovanja i izrade ovog završnog rada.

Sažetak

Osnovna ideja završnog rada je osmišljanje i realizacija sustava za automatsko zalijevanje biljke korištenjem Arduino razvojne platforme. Odabirom Arduino DUE razvojne platforme, koja svojom cijenom, specifikacijama i mogućnostima kasnije nadogradnje sustava sasvim zadovoljava zadatke ovog završnog rada, dobiva se glavni dio sustava za automatsko zalijevanje biljke. Razmatranjem ideje završnog rada i praktičnom realizacijom iste, napravljena je maketa za prikaz upravljanja zalijevanjem biljke koja je smanjeni prikaz sustava koji se može primijeniti u svakodnevnom životu kod navodnjavanja vrta i ostalih poljoprivrednih parcela.

Maketa se sastoji od upravljačke jedinice (Arduino DUE), spremnika, pumpe, senzora i solenoidnih ventila. Ulaskom u izbornik sustava za zalijevanje, pomoću tipki i prikazu na LCD zaslonu na upravljačkoj jedinici, postavljaju se vrijednosti temperature u spremniku, vlažnosti tla oko biljke te način rada sustava (automatski ili ručno). Na upravljačkoj jedinici, osim tipki i LCD zaslona, izvedena je priključnica za pumpu za dovođenje tekućine u spremnik upravljana preko releja, LED diode za signalizaciju statusa sustava za upravljanje (normalni rad ili greška), sklopka za isključenje postrojenja te priključnice za senzore i solenoidne ventile. Odvođenje tekućine iz spremnika prema biljci realizirano je pomoću solenoidnog ventila koji je upravljan pomoću releja.

Rad makete je dokumentiran kako bi se mogla napraviti analiza teoretskih pretpostavki upravljanja sustavom za zalijevanje biljke.

KLJUČNE RIJEČI: Arduino DUE, mikrokontroler, solenoidni ventil, senzor temperature, senzor razine u spremniku, senzor vlažnosti, pumpa

Abstract

The basic idea is designing and implementing system for automatic watering plants using Arduino DUE development platform. Choosing Arduino DUE development platform, which price, specifications and capabilities later upgrade the system completely fulfill the tasks of this work, gets the main part of the system for automatic watering plants. After examining the ideas of work and the practical realization of the same, made a model for display control watering a plant which is reduced display system that can be applied in everyday life when watering the garden.

The scale model consists of a control unit (Arduino DUE), tanks, pumps, sensors and solenoid valves. Entering the menu system for watering, using the keys and the display on the LCD screen on the control unit, set the value of the tank temperature, soil moisture around the plant and the operation of the system (automatically or manually). On the control unit, except the buttons and LCD screen, performed a connector for the pump for supplying the liquid in the tank controlled by relays, LEDs for indicating the status management system (normal operation or error), switch for ON/OFF a model and connectors for the sensors and the solenoid valves . Drainage of liquid from the tank to the plant is realized by a solenoid valve which is controlled by the relay.

Working of the model is documented in order to make the analysis of theoretical assumptions management system for watering plants.

KEY WORDS**:** Arduino DUE, microcontroller, soleonoid valve, temperature sensor, tank level sensor, moisture sensor, pump

Popis korištenih kratica

PWM	Pulse Width Modulation Modulacija širine impulsa
DC	Istosmjerna struja
AC	Izmjenična struja
AO	Analog output Analogni izlaz
DO	Digital output Digitalni izlaz
IN	Input Ulazni signal
VCC	Positive supply voltage Pozitivni polaritet napona
GND	Ground „Masa“
PU	Polyurethan Poliuretan
NC	Normally closed Zatvoreno u normalnom stanju
A/D	Analog/Digital Conversation Analogno/Digitalna pretvorba

Sadržaj

1. Uvod 6
2. SUSTAV ZA AUTOMATSKO ZALIJEVANJE BILJKE 8
2.1. Postojeća rješenja automatskog sustava za zalijevanje biljaka 9
3. REALIZACIJA SUSTAVA ZA AUTOMATSKO ZALIJEVANJE BILJKE 10
3.1. Arduino DUE razvojna platforma 10
3.2. Magnetsko osjetilo razine tekućine u spremniku 11
3.3. Osjetilo temperature DS18B20 12
3.4. Osjetilo za mjerenje vlažnosti 13
3.5. Pumpa i regulacija protoka 14
3.6. Solenoidni ventil 16
4. MAKETA SUSTAVA ZA AUTOMATSKO ZALIJEVANJE BILJKE 17
4.1. Upravljačka jedinica sustava 17
4.2. Realizacija programskog koda automatskog upravljanja za Arduino DUE 20
4.3. Prikaz rada sustava upravljanja automatskog zalijevanja biljke 35
5. Zaključak 40
6. Literatura 41
7. Prilozi 43

1. Uvod

Završni rad „Realizacija sustava za automatsko zalijevanje biljke“ je izrađen na temelju svakodnevne potrebe zalijevanja vrta u doba ljetnih visokih temperatura te ideje da se ta radnja automatizira zbog uštede vremena i povećanja učinkovitosti samog zalijevanja. Zbog mogućnosti primjene sustava za automatsko zalijevanje biljke, kako u vlastitom vrtu tako i kod većih poljoprivrednih parcela i rasadnika, ideja je da se izradi univerzalni sustav kojeg će se moći lako konfigurirati te po potrebi prilagoditi i nadograditi hardverom za specifično mjesto upotrebe (vrt, rasadnik, poljoprivredno zemljište ...).

Odabirom Arduino DUE razvojne platforme, uz mnoštvo mogućnosti same platforme i mogućnosti eventualne kasnije nadogradnje sustava (Ethernet, Wi-Fi...), dobiva se glavni dio (srce) samog upravljanja. Za ulazne elemente upravljanja koriste se magnetska osjetila razine tekućine u spremniku, osjetilo temperature DS18B20 te otporničko osjetilo vlažnosti tla. Za izvršne elemente upravljanja koriste se releji koji upravljaju pumpom za dovođenje tekućine u spremnik te solenoidnim ventilom za odvođenje tekućine iz spremnika. Prikaz stanja sustava i podešavanje parametara upravljanja realizirano je pomoću LCD 16*2 zaslona i pripadajućih tipki. Programiranje samog upravljanja, odnosno Arduino DUE razvojne platforme, vrši se pomoću Arduino IDE programskog okruženja u programskom jeziku C.

Nakon nabave svih potrebnih dijelova hardvera, sastavila se maketa sustava upravljanja koja prikazuje realni sustav upravljanja u umanjenom obliku. Na temelju sastavljene makete sustava, u programskom okruženju Arduino IDE izrađen je programski kod upravljanja za Arduino DUE razvojnu platformu. Testiranje same makete se prvobitno obavlja na eksperimentalnoj pločici zbog mogućnosti eventualne lake i brze modifikacije same makete. Nakon faze testiranja same makete i odgovarajućeg programskog koda te uz modifikaciju istog, izrađena je finalna verzija programskog koda koji je implementiran u Arduino DUE razvojnu platformu. Po završetku izrade finalne verzije programskog koda i provjere istog, izrađena je tiskana pločica u Eagle programskom okruženju za spajanje senzora i ostalih dijelova s Arduino DUE razvojnom platformom. Elektronički dio završnog rada je smješten u plastično kućište s IP55 zaštitom. Za potrebe testiranja makete sustava korištena je 12V DC pumpa za dovođenje tekućine u spremnik. Upravljanje protokom pumpe je pomoću PWM regulatora koji je izrađen za tu namjenu. Odvođenje tekućine iz spremnika realizirano je pomoću solenoidnog ventila i 12V DC pumpe, koja se uključuje i isključuje zajedno sa solenoidnim ventilom. Rezultati testiranja funkcionalnosti makete su dokumentirani te detaljno obrađeni u nastavku završnog rada.

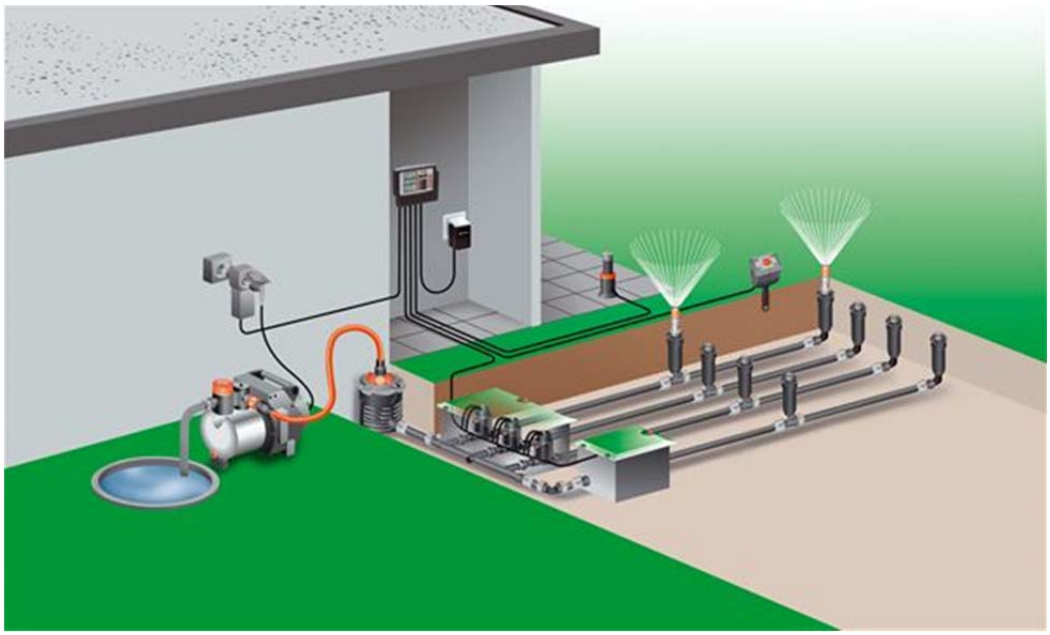
Glavna zamisao kod razrade ideje sustava za automatsko zalijevanje biljke je laka prilagodba mjestu upotrebe samog sustava te mogućnost nadogradnje sustava prema potrebama. Maketa

sustava za automatsko zalijevanje biljke može se koristiti i na laboratorijskim vježbama pojedinih kolegija kao konkretan primjer realizacije nekog sustava automatskog upravljanja. Otvorenost i dostupnost programskog koda te samog sklopovlja, omogućuje studentima modifikaciju te testiranje istog na konkretnom primjeru realiziranog sustava.

Izradom ovog završnog rada omogućava se iskustvo realizacije sustava automatskog upravljanja koji je koristan i može se koristiti u svakodnevnom životu kao pomoć kod proizvodnje namirnica kako za svoje potrebe tako i za potrebe ostalog stanovništva. Na temelju znanja i iskustva stečenih kroz studijsko obrazovanje, uvelike je olakšan put prema izvršenju postavljenog zadatka završnog rada. Problemi koji su se pojavili tokom izrade rada omogućavaju istraživački način pronalaska rješenja problema te samim time i nadogradnju stečenih znanja i iskustva koje će biti korisno u daljnjem poslu i obrazovanju.

2. SUSTAV ZA AUTOMATSKO ZALIJEVANJE BILJKE

Sustav za automatsko zalijevanje biljke (slika 1) je automatizirani sustav koji se sastoji od centralne upravljačke jedinice koja upravlja samim procesom zalijevanja, ulaznih elemenata (osjetila) upravljačke jedinice koji daju podatke mjerenim veličinama (temperatura, vlažnost ...) te izvršnih elemenata koji obavljaju izlazne zadatke upravljačke jedinice. Ulazne elemente čine različita osjetila za mjerenje temperature, osjetila za mjerenje vlažnosti tla te osjetila razine tekućine u spremniku ukoliko se koristi spremnik tekućine kod zalijevanja. Izvršne elemente čine releji koji upravljaju pumpama za dovod tekućine za zalijevanje i solenoidnim ventilima za regulaciju tekućine u cijevima i spremniku.



Slika 1 : Sustav za automatsko zalijevanje [12]

Zbog povećanja produktivnosti samog zalijevanja biljaka potrebno je koristiti različita osjetila koja nam daju informaciju o stanju mjerene veličine. Korištenjem osjetila vlažnosti tla dobiva se informacija o stanju tla, odnosno je li potrebno zalijevanje biljke, što omogućuje uštedu vode i energije. Zbog činjenice da biljke ne bi trebalo zalijevati hladnom vodom zbog šoka koji biljku pogodi u tom trenutku i time umanjuje produktivnost samog zalijevanja, zbog pritom možebitnog oštećenja biljke, potrebno je biljku zalijevati vodom sobne temperature. Kod realizacije sustava za automatsko zalijevanje biljke potrebno je koristiti spremnik u kojem će se voda zagrijati na sobnu temperaturu pomoću utjecaja sunčeve topline ili pomoću grijača.

2.1. Postojeća rješenja automatskog sustava za zalijevanje biljaka

Različiti svjetski proizvođači nude gotova rješenja za sustave automatskog zalijevanja biljaka (slika 2), koja su većinom standardizirana i ograničena mogućnostima. Svako eventualno proširenje sustava se posebno naplaćuje ili je u većini slučajeva gotovo neizvedivo, što daje još jedan razlog za izradu vlastitog rješenja sustava koje će biti fleksibilno i proširivo uz minimalno ulaganje. Cijena komercijalnih gotovih rješenja sustava je ponekad i nekoliko puta višestruka u odnosu na izradu vlastitog rješenja.

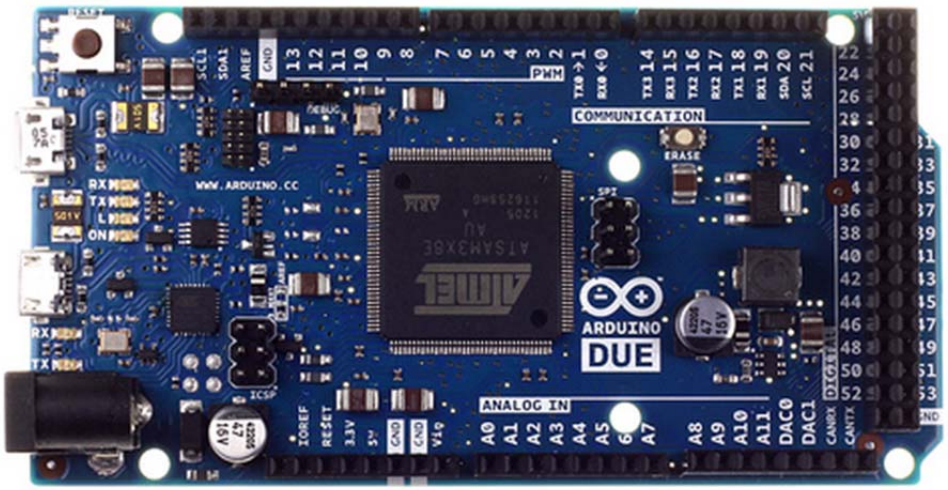
Slika 2: Komercijalno rješenje sustava za automatsko zalijevanje biljaka [13]

3. REALIZACIJA SUSTAVA ZA AUTOMATSKO ZALIJEVANJE BILJKE

Zadatak završnog rada je projektiranje i realizacija sustava za automatsko zalijevanje biljke korištenjem mikrokontrolerske razvojne platforme. Sustav za zalijevanje treba imati mogućnost ručnog i automatskog načina rada. U slučaju ručnog načina rada operater upravlja zalijevanjem. Automatski način rada omogućuje zalijevanje biljke kada je temperatura vode u spremniku unutar zadanih granica i kada je vlažnost tla biljke ispod zadane razine. Korišteni hardver te programsko okruženje korišteno za izradu ovog rada opisano je u nastavku.

3.1. Arduino DUE razvojna platforma

Upravljanje sustavom za automatsko zalijevanje biljke je realizirano pomoću Arduino DUE razvojne platforme. Arduino DUE (slika 3) je mikrokontrolersko razvojno okruženje bazirano na Atmel SAM3X8E ARM Cortex – M3 32-bitnom mikroprocesoru brzine 84 MHz, radne memorije 96kB te unutarnje memorije 512 kB. Sastoji se od 54 digitalnih ulaza/izlaza, od kojih 12 se može koristiti kao PWM izlazi, 12 analognih ulaza i 4 UART (serijskih) porta [1].



Slika 3: Arduino DUE razvojna platforma [14]

Ulazni priključni napon je istosmjerni napon u granicama 6 – 16V (preporučljivo 7-12V). Mikrokontroler radi na naponskoj razini do max 3.3V pa je potrebno pripaziti kod spajanja ulaza i izlaza da naponska razina ne prelazi 3.3V jer se u protivnom mikrokontroler može nepovratno oštetiti. Programiranje samog mikrokontrolera se vrši pomoću Arduino IDE programskog okruženja preko microUSB porta koji se nalazi na platformi. Shema Arduino DUE razvojne platforme nalazi se u prilogu 1.

3.2. Magnetsko osjetilo razine tekućine u spremniku

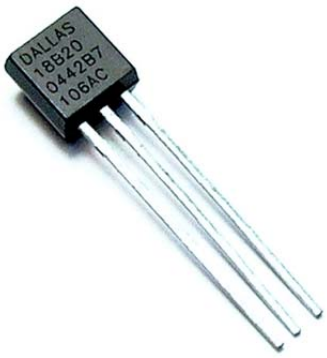
Za određivanje razine u spremniku koriste se magnetska osjetila razine (slika 4), koja se sastoje od kliznog prstena unutar kojeg se nalazi magnet i plastičnog kućišta unutar kojeg se nalaze otvoreni kontakti (NO) prekidača [2].

Slika 4: Magnetski senzor razine u spremniku [15]

Osjetilo radi na principu zatvaranja kontakata uslijed djelovanja magneta koji se nalazi u prstenu. Kada se razina tekućine u spremniku podigne do razine gdje je montirano osjetilo, dolazi do podizanja prstena i magneta koji zatvara kontakte prekidača koji se nalazi u plastičnom kućištu osjetila. Zatvaranje kontakta na osjetilu se registrira spajanjem osjetila na digitalni ulaz Arduino DUE razvojne platforme. Na spremniku se nalaze dva osjetila razine, jedno na samom dnu spremnika koje omogućava detekciju praznog spremnika te jedno na samom vrhu spremnika koje omogućava detekciju napunjenosti spremnika. Osim prije navedenih osjetila, razinu u spremniku moguće je odrediti vaganjem spremnika, pomoću ultrazvučnih osjetila, pomoću kapacitivnih pretvornika, pomoću otporničkih traka ... Ostale navedene metode određivanja razine u spremniku iziskuju u većini slučajeva dodatni dio elektroničkog sklopovlja što automatski povećava cijenu pretvornika. Upotrebom magnetskih osjetila razine u spremniku omogućena nam je fleksibilnost kod ugradnje (nismo ograničeni na veličinu spremnika).

3.3. Osjetilo temperature DS18B20

Za određivanje temperature tekućine u spremniku koristi se digitalno osjetilo temperature Maxim DS18B20 (slika 5) u TO-92 kućištu [3]. Digitalno osjetilo je spojeno pomoću Maxim 1-wire bus protokola na analogni ulaz Arduino DUE razvojne platforme. Maxim 1 – wire bus protokol omogućuje korištenje jedne žice za slanje i primanje podataka (DQ pin osjetila) sa i na osjetilo te spajanje do 127 osjetila na istu žicu. Osjetilo ima jedinstveni 64-bitni ROM kod, poput serijskog broja osjetila, koji nam omogućuje definiciju i pronalazak osjetila kod korištenja 1-wire bus protokola.



Slika 5: DS18B20 senzor [16]

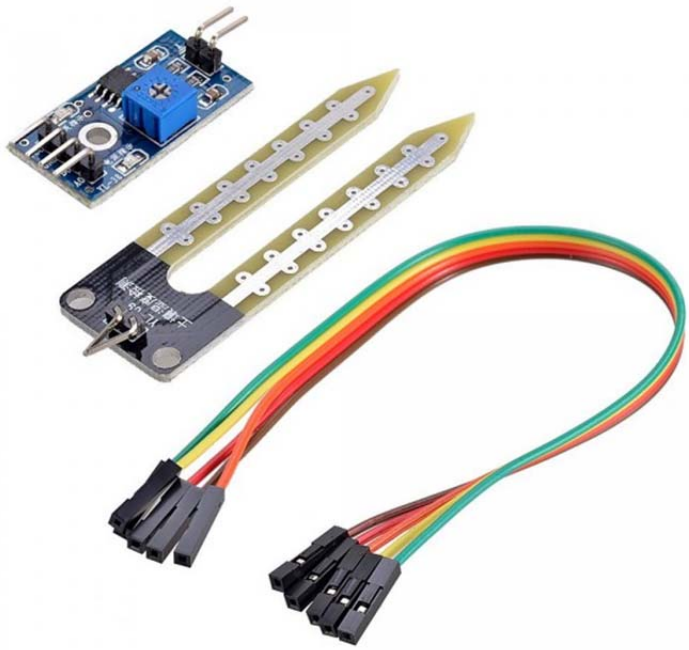
Kod korištenja digitalnog osjetila preko 1-wire bus protokola potrebno je preuzeti odgovarajuću knjižnicu (library) vezanu uz navedeni protokol te ju ubaciti na početku programa (`#include <OneWire.h>`) u Arduino IDE programskom okruženju. Dohvat mjerenja temperature sa osjetila vrši se naredbom [`44h`] prema osjetilu (`ds.write(0x44, 1)`). Nakon dohvata rezultata mjerenja sa osjetila pokreće se proces konvertiranja podataka koji je relativno spor i može potrajati do 750 ms pa je potrebno osigurati pauzu od minimalno 750ms (preporučljivo 1000ms) prije izvršavanja slijedeće naredbe u programskom kodu. Rezultati mjerenja su spremljeni u radnu memoriju (RAM) osjetila pa ih je potrebno dohvatiti (pročitati) naredbom [`Beh`] prema osjetilu (`ds.write(0xBe)`). Dohvatom podataka iz radne memorije dobivamo 9 bajta podataka, od kojih su nam potrebni Bajt0 i Bajt1 za izračun mjerene temperature. Izračun mjerene temperature se vrši prema formuli:

$$Temp = \big((data[1] \ll 8) + data[0]\big) * 0.0625 \qquad (3.1)$$

Učitani bitovi Bajta1 (data[1]) se zbrajaju s bitovima Bajta0 (data[0]) te se taj zbroj multiplicira s koeficijentom 0.0625, zbog korištenja 12-bitne rezolucije osjetila, te se dobiva mjerena temperatura u °C. Koeficijent 0.0625 se koristi zbog korištenja 4 najniža bita (bit0, bit1, bit2, bit3) bajta0 za decimalni zapis izmjerene temperature kod korištenja 12-bitne rezolucije (1/(2^4)=1/16=0.0625). Tablica specifikacija senzora DS18B20 dostupna je na poveznici [4].

3.4. Osjetilo za mjerenje vlažnosti

Za određivanje vlažnosti tla biljke koristi se otporničko osjetilo vlažnosti (slika 6) spojen na svoj popratni elektronički sklop [5]. Elektronična shema elektroničkog sklopa osjetila vlažnosti se nalazi u prilogu 2. Na stezaljke IN priključuje se samo osjetilo koje se nalazi u tlu u blizini biljke koja se zalijeva. Istosmjerni napon razine 3.3V – 5.0V priključuje se na stezaljke VCC (pozitivni polaritet) i GND (negativni polaritet, odnosno „masa“).



Slika 6: Senzor za mjerenje vlažnosti [17]

Rezultat mjerenja sa osjetila dobiva se na analognom izlazu (AO) elektroničkog sklopa koji se spaja na analogni ulaz Arduino DUE razvojne platforme. Na elektroničkom sklopu osjetila se nalazi digitalni izlaz (DO) koji postaje aktivan kada je vrijednost mjerenja sa osjetila (vrijednost otpora) jednaka ili veća od postavljene vrijednosti pomoću promjenjivog otpornika otpora 10kΩ koji se nalazi na samom sklopu. Usporedbu vrijednosti otpora sa osjetila i zadanog otpora vrši dvostruki komparator LM393. Digitalni izlaz se može spojiti na Arduino DUE razvojnu platformu za signalizaciju alarma ili direktno na upravljanje relejom za uključenje alarma (sirena), pumpe, solenoidnog ventila ...

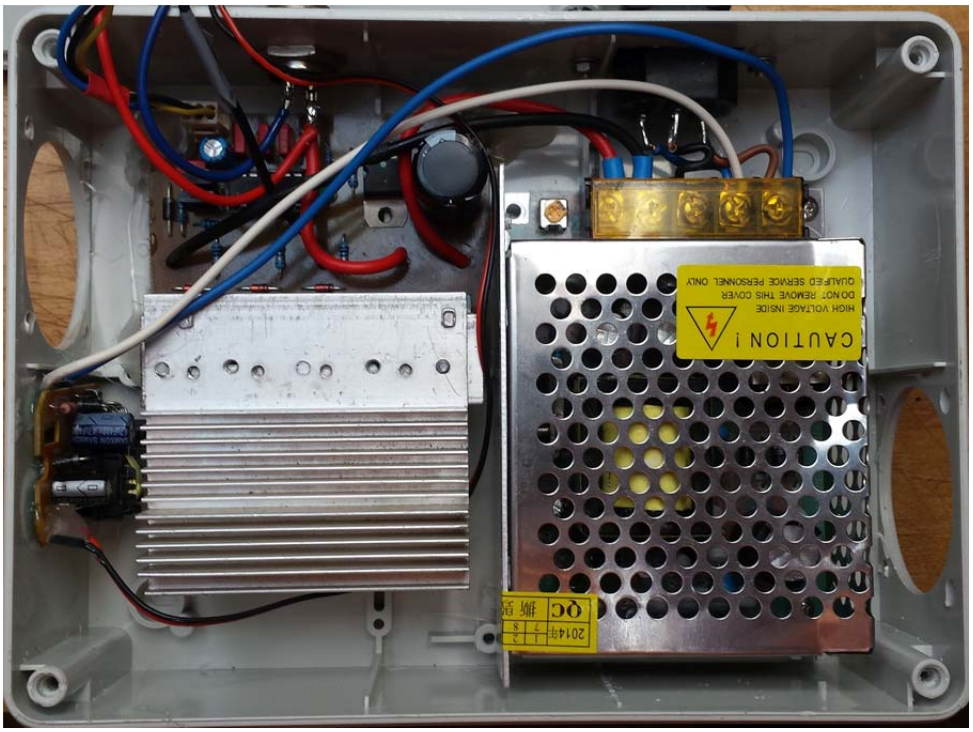
3.5. Pumpa i regulacija protoka

Punjenje spremnika se obavlja pomoću male JT-800 12V DC pumpe (slika 7) protoka do 1000 l/h [20]. Istosmjerni napon napajanja pumpe je u granicama od 6V DC do 12V DC koji omogućuje regulaciju protoka od 450 l/h do 1000 l/h. Visina dobave tekućine iznosi od 1 m do maksimalnih 6 m. Ovisnost protoka o priključenom naponu prikazana je na grafičkom prikazu u prilogu 3.

Slika 7: JT-800 pumpa [18]

Spojevi pumpe sa spremnikom tekućine za zalijevanje i spremnikom vode za dobavu su izvedeni pomoću brzih spojnica 1/4" i PU crijeva vanjskog promjera 8 mm. Regulacija protoka pumpe se vrši pomoću PWM regulatora (slika 8 i 9) koji omogućuje regulaciju istosmjernog napona od 0 do maksimalne vrijednosti ulaznog priključenog napona [6]. Električna shema PWM regulatora, shema vodova i elemenata na tiskanoj pločici prikazani su u prilogu 4, 5 i 6. Upravljanje uključenjem PWM regulatora obavlja relej čiji su upravljački kontakti spojeni preko opto-couplera i NPN tranzistora na digitalni izlaz Arduino DUE razvojne platforme. Radni kontakti releja spojeni su na ulazni fazni vod (L) napona 230V AC i šuko priključnicu na kućištu upravljačke jedinice.

Slika 8: PWM regulator pumpe u plastičnom IP55 kućištu



Slika 9: Dijelovi PWM regulatora pumpe

Izvor napajanja PWM regulatora je ispravljač (AC-DC) snage 60W, izlaznog napona 12V DC i maksimalne izlazne jakosti struje od 5A [21]. Širina impulsa PWM regulatora je upravljana pomoću čipa TL494. Izlazna snaga PWM regulatora je određena izlaznim N kanalnima power MOSFET tranzistorima (IRF3205 110A 55V TO-220 [7]) koji obavezno moraju biti montirani na rashladno tijelo (aluminijski hladnjak) zbog disipacije topline.

3.6. Solenoidni ventil

Kontrola odvođenja tekućine iz spremnika obavlja se pomoću 12V DC solenoidnog ventila (slika 10), maksimalnog radnog tlaka do 8 bara [8]. Kontakti za napajanje solenoidnog ventila su spojeni na priključnicu koja se nalazi na upravljačkoj jedinici. Upravljanjem solenoidnog ventila upravlja relej čiji su upravljački kontakti spojeni preko opto-couplera i NPN tranzistora na digitalni izlaz Arduino DUE razvojne platforme. Radni kontakti releja spojeni su na pozitivni pol izvora istosmjernog napona od 12V i priključnicu na kućištu upravljačke jedinice. Masa (GND) izvora istosmjernog napona od 12V je direktno spojena s priključnicom za solenoidni ventil na kućištu upravljačke jedinice.

Slika 10: Solenoidni ventil 12V DC [19]

Solenoidni ventil je u inicijalnom stanju zatvoren (NC) što znači da je za njegovo uključenje (aktivaciju) potrebno dovesti istosmjerni napon od 12V na stezaljke solenoidnog ventila. Kada napon na stezaljkama ventila padne na nulu, solenoidni ventil se zatvara i ostaje u takvom stanju sve dok se ponovno ne dovede istosmjerni napon od 12V. Spoj spremnika i solenoidnog ventila je izveden pomoću brzih spojnica 1/4" i PU cijevi promjera 8mm uz korištenje redukcije 1/2" na 1/4" zbog priključaka solenoidnog ventila.

4. MAKETA SUSTAVA ZA AUTOMATSKO ZALIJEVANJE BILJKE

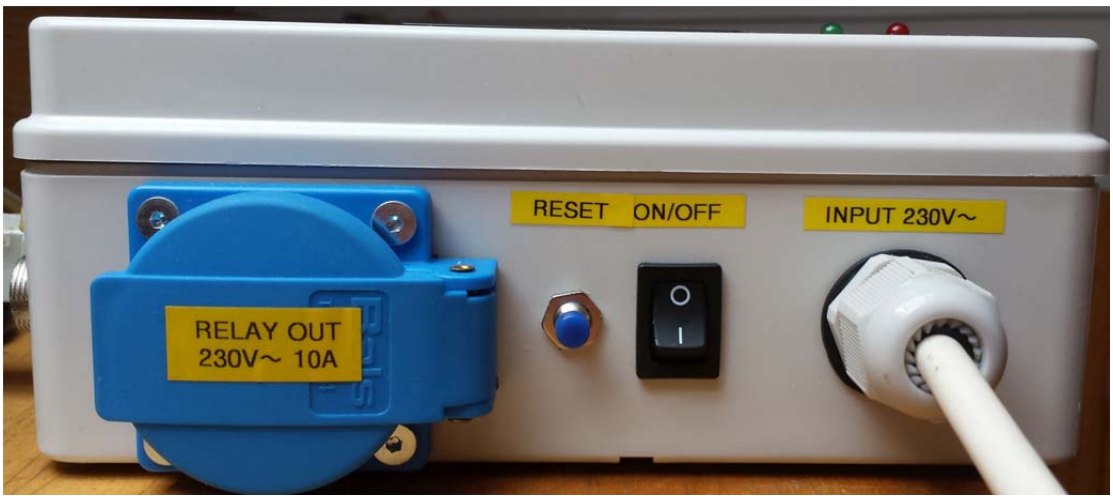
Maketa sustava za automatsko zalijevanje biljke (slika 11) sastoji se od upravljačke jedinice sustava s Arduino DUE razvojnom platformom, spremnika tekućine za zalijevanje s montiranim magnetskim osjetilima razine tekućine u spremniku i digitalnim osjetilom temperature DS18B20, solenoidnog ventila i pumpe za odvođenje tekućine iz spremnika, pumpe za dovođenje tekućine s pripadajućim PWM regulatorom te posude s biljkom i osjetilom za mjerenje vlažnosti tla.

Slika 11: Maketa sustava za automatsko zalijevanje biljke

4.1. Upravljačka jedinica sustava

Upravljačka jedinica sustava za automatsko zalijevanje biljke (slika 12, 13 i 14) glavni je dio samog sustava upravljanja te se svi ulazni i izlazni elementi spajaju na tu jedinicu. Sastoji se od glavnog dijela, Arduino DUE razvojne platforme, istosmjernog izvora napajanja napona 12V za elektroniku same jedinice, DC-DC elektroničkih sklopova koji smanjuju naponsku razinu od 12V na naponsku razinu od 9V i 5V, tipkala za podešavanje parametara sustava upravljanja, LCD 16*2 zaslona za prikaz stanja sustava, priključnica za osjetila i izvršne elemente sustava te sklopke za prekid napajanja same jedinice.

Slika 12: Upravljačka jedinica sustava upravljanja – prednja strana



Slika 13: Upravljačka jedinica sustava upravljanja – stražnja strana

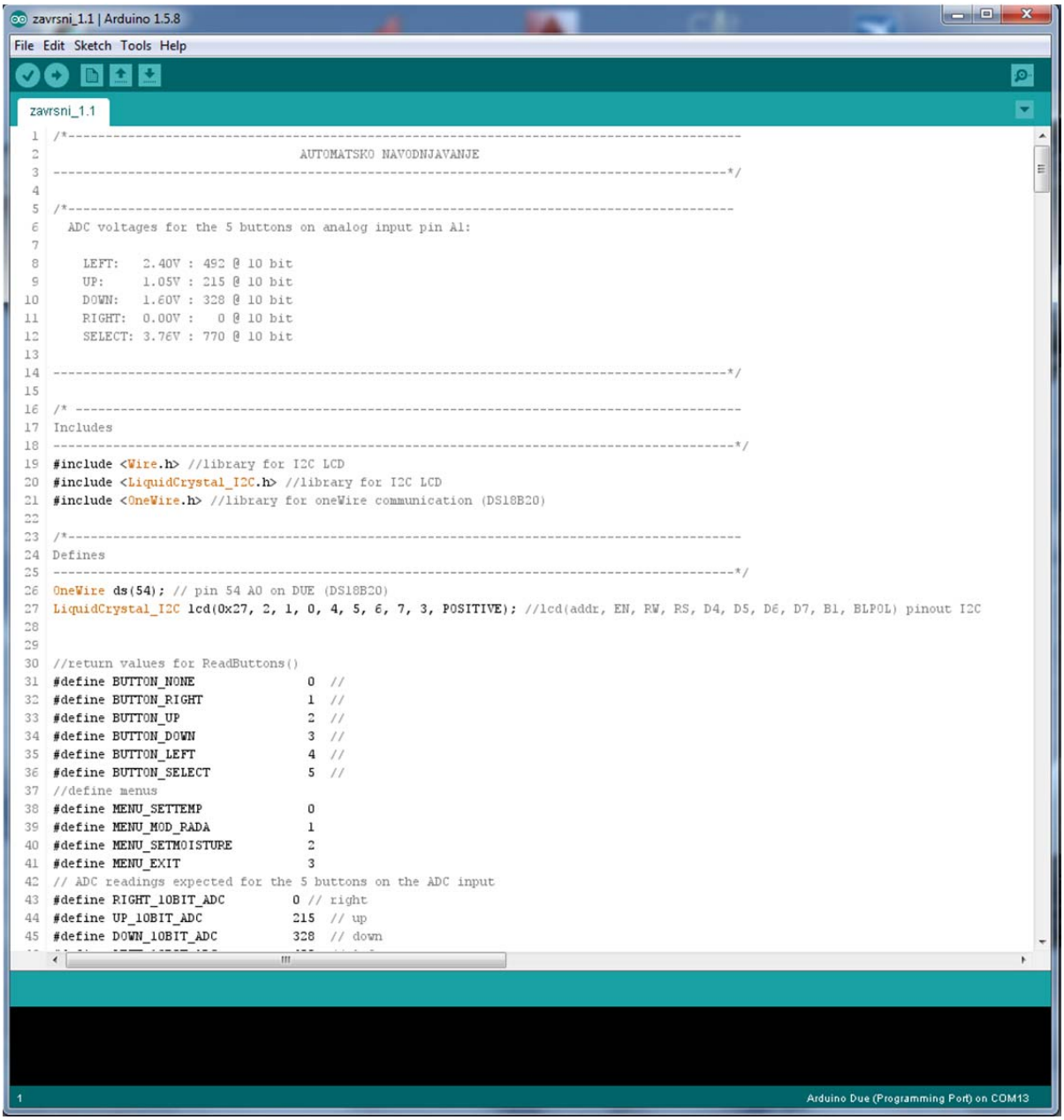
Slika 14: Upravljačka jedinica sustava upravljanja – bočna strana

Kompletna upravljačka jedinica je smještena unutar plastičnog kućišta s IP55 zaštitom, otpornog na sunčevo zračenje. Hlađenje upravljačke jedinice izvedeno je pomoću dva ventilatora promjera 50 mm od kojih jedan upuhuje zrak unutar kućišta dok drugi ispuhuje zrak izvan kućišta. Napajanje ventilatora, zujalice i Arduino DUE razvojne platforme iznosi 9V DC te je izvedeno pomoću DC – DC regulatora, snižavanjem naponske razine od 12V na naponsku razinu od 9V pomoću stabilizatora napona LM7809. Napajanje upravljačkih pinova releja, LCD 16*2 zaslona, tipkovnice za podešavanje, osjetila temperature i osjetila vlažnosti iznosi 5V DC te je izvedeno pomoću DC – DC regulatora, snižavanjem naponske razine od 12V na naponsku razinu od 5V pomoću stabilizatora napona LM7805. Sheme regulatora te raspored vodova na tiskanim pločicama izrađene u programskom okruženje Eagle su prikazane u prilogu 7 i 8.

Prikaz trenutnog stanja sustava ostvaren je pomoću LCD 16*2 zaslona s plavim pozadinskim osvjetljenjem. Pomoću tipkovnice na kućištu upravljačke jedinice i LCD zaslona podešavaju se parametri sustava upravljanja preko izbornika dokumentiranog u nastavku. Tipkovnica za podešavanje parametara sastoji se od 5 tipki (taster prekidača) koje su međusobno povezane otpornicima različitih iznosa. Napajanje tipkovnice iznosi 5V DC, a izlaz je spojen na analogni ulaz Arduino DUE razvojne platforme. Ovisno o pritisku pojedine tipke na tipkovnici dobiva se određena naponska razina koja se A/D pretvorbom u Arduino DUE pretvara u 10 – bitni digitalni podatak (0-5V analogno = 0-1023 digitalno kod 10-bit). Raspored elemenata i vodova tipkovnice prikazan je u prilogu 9.

4.2. Realizacija programskog koda automatskog upravljanja za Arduino DUE

Nakon odabira potrebnih komponenti sustava upravljanja i njihovog međusobnog povezivanja preko eksperimentalne pločice, potrebno je u programskom okruženju Arduino IDE (slika 15) izraditi programski kod samog upravljanja procesom automatskog zalijevanja biljke. Prilikom upoznavanja s načinom funkcioniranja Arduino DUE razvojne platforme potrebno je proučiti neku od navedene literature i shvatiti i isprobati osnovne programske kodove koje programsko okruženje Arduino IDE nudi da bi se realizacija i programiranje samog sustava uvelike olakšalo [9, 10, 11]. Programiranje se izvodi u programskom jeziku C.



Slika 15: Arduino IDE

Na početku programskog koda potrebno je ubaciti sve knjižnice (library) koje će se koristiti prilikom programiranja koje su prije toga preuzete i instalirane u Arduino IDE programsko okruženje. Knjižnice su definicije i funkcije koje služe za funkcioniranje posebnih dijelova sklopovlja koje koristimo (LCD zaslon) i protokola (One Wire). Korištene knjižnice ubacuju se na način prikazan u nastavku:

```
#include <Wire.h> //library for I2C LCD
#include <LiquidCrystal_I2C.h> //library for I2C LCD
#include <OneWire.h> //library for oneWire communication (DS18B20)
```

Oznaka // nam omogućuje komentiranje pojedine linije programskog koda te se tekst nakon // ne uzima u obzir kod prevođenja (kompajliranja) i izvršenja programskog koda. Nakon ubacivanja svih potrebnih knjižnica potrebno je definirati vrijednosti korištenih pinova samog mikrokontrolera te konstante. Poželjno je koristiti simbolička imena kod definicije kao u nastavku, što olakšava kasnije programiranje i snalaženje u programskom kodu. Definicije korištene u programskom kodu ovog rada prikazane su u nastavku:

```
OneWire ds(54); // pin 54 A0 on DUE (DS18B20) – definicija korištenog
pina kod OneWire protokola

LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
//lcd(addr, EN, RW, RS, D4, D5, D6, D7, B1, BLPOL) pinout I2C –
deklaracija lcd zaslona


#define BUTTON_NONE               0  // return from readButtons()
#define BUTTON_RIGHT              1  // return from readButtons()
#define BUTTON_UP                 2  // return from readButtons()
#define BUTTON_DOWN               3  // return from readButtons()
#define BUTTON_LEFT               4  // return from readButtons()
#define BUTTON_SELECT             5  // return from readButtons()

//definicija menija

#define MENU_SETTEMP              0
#define MENU_MOD_RADA             1
#define MENU_SETMOISTURE          2
#define MENU_EXIT                 3

// definicija očekivanih vrijednosti kod A/D pretvorbe pritiskom
tipke na tipkovnici

#define RIGHT_10BIT_ADC         0 // right
#define UP_10BIT_ADC            215  // up
#define DOWN_10BIT_ADC          328  // down
#define LEFT_10BIT_ADC          492  // left
#define SELECT_10BIT_ADC        770 // select
#define BUTTONHYSTERESIS         50  // histereza

#define SOIL_SENSOR       A3 // A3 analogni ulaz za senzor vlažnosti
#define BUTTON_ADC_PIN    A2 // A2 analogni ulaz za tipkovnicu
#define SENZOR_H 9  //magnetski senzor razine u spremniku GORE – pin9
```

```
#define SENZOR_L 8 //magnetski senzor razine u spremniku DOLJE - pin8
#define SOLENOID 10  //relay1 on pin 10 - solenoidni ventil
#define PUMP 11  //relay2 on pin 11 - pumpa
#define LED1 12 // LED1 on pin 12
#define LED2 13 // LED2 on pin 13
#define BUZZER 7 // buzzer on pin 7
#define RELAY_OFF  1 // relej isključen
#define RELAY_ON   0 // relej uključen
```

Nakon definicija pojedinih pinova, potrebno je definirati korištene varijable u programskom kodu:

```
byte buttonJustPressed  = false;
byte buttonJustReleased = false;
byte buttonWas          = BUTTON_NONE;
byte temp =0;
//used by ReadButtons() for detection of button events
boolean menu = true;
boolean set_temp = true;
boolean set_soil = true;
boolean mod_rada = true;
boolean auto_mod=true;
boolean manual_mod=false;
boolean error_flag=false;
boolean puni=true;
boolean prazni=true;
int menu_number = 0;
int mod=0;
float set_temperature = 20.0;
float set_moisture = 30.0;
char* MENU[] = {"Set TEMPERATURE", "Set MOD RADA", "Set MOISTURE",
"EXIT "};
```

Kada su definirane sve varijable programskog koda, potrebno je napraviti inicijalno postavljanje parametara sustava unutar petlje `void setup ()` koja se izvodi samo prilikom pokretanja programskog koda. Petlja `void setup ()` programskog koda završnog rada prikazana je u nastavku:

```
void setup()
{


    // deklaracija pinova

  pinMode( BUTTON_ADC_PIN, INPUT ); // definicija A2 kao ulaza
  digitalWrite( BUTTON_ADC_PIN, LOW );//isključenje internog pullup
otpornika
  pinMode(SOIL_SENSOR, INPUT); //A3 ulaz SOIL SENSOR

  pinMode (SENZOR_H, INPUT_PULLUP);// SENZOR GORE – ulaz s uključenim
internim pullup otpornikom
  pinMode (SENZOR_L, INPUT_PULLUP);// SENZOR DOLJE - ulaz
  pinMode (LED1, OUTPUT);  // LED1 - izlaz
  pinMode (LED2, OUTPUT);  // LED2 - izlaz
  pinMode (SOLENOID, OUTPUT);  // SOLENOIDNI VENTIL - izlaz
  pinMode (PUMP, OUTPUT);  // PUMPA - izlaz
```

```
  digitalWrite(SOLENOID, RELAY_OFF); // SOLENOIDNI VENTIL OFF kod
incijalizacije
  digitalWrite(PUMP, RELAY_OFF);  // PUMPA OFF kod inicijalizacije
  digitalWrite(LED2, LOW);
  digitalWrite(LED1, LOW);

// incijalno stanje spremnika i postavljanje zastavica o stanju
senzora razine

   if ((digitalRead(SENZOR_H)== LOW) && (digitalRead(SENZOR_L) ==
HIGH))//spremnik pun, potrebno pražnjenje
 {
   prazni=true;
   puni=false;
   }
 if  ((digitalRead(SENZOR_L)== LOW)  &&  (digitalRead(SENZOR_H)  ==
HIGH))//spremnik prazan, potrebno punjenje
 {
   puni=true;
   prazni=false;
   }
  if (((digitalRead(SENZOR_H)== HIGH) && (digitalRead(SENZOR_L) ==
HIGH))) //spremnik ok, moguće pražnjenje
 {
   prazni=true;
   puni=false;
   }

  lcd.begin(16,2);  //definiranje tipa LCD zaslona (16*2)
  lcd.backlight(); //uključenje pozadinskog osvjetljenja LCD zaslona

  // početni ispis na LCD - samo kod uključenja

  lcd.setCursor(2,0); // cursor position (CHAR, LINE)
  lcd.print("Sveuciliste"); //ispis na LCD
  lcd.setCursor(5,1); // postavljanje početka ispisa
  lcd.print("Sjever");
  delay(2000); // pauza 2s
  lcd.clear(); //brisanje LCD zaslona
  lcd.setCursor(5,0);
  lcd.print("odjel");
  lcd.setCursor(1,1);
  lcd.print("ELEKTROTEHNIKE");
  delay(2000); // pauza 2s

  lcd.clear(); //erase display
  lcd.setCursor(1,0);
  lcd.print("Karlo Tretnjak");
  lcd.setCursor(3,1);
  lcd.print("0036449080");
  delay(2000); // pauza 2s

  lcd.clear(); //erase display
  lcd.setCursor(3,0);
  lcd.print("AUTOMATSKO");
  lcd.setCursor(2,1);
```

```
lcd.print("UPRAVLJANJE");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(2,0);
lcd.print("Zavrsni rad");
lcd.setCursor(2,1);
lcd.print("2014 / 2015");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(3,0);
lcd.print("AUTOMATSKO");
lcd.setCursor(1,1);
lcd.print("NAVODNJAVANJE");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(5,0);
lcd.print("Mentor:");
lcd.setCursor(3,1);
lcd.print("M.HORVATIC");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(2,0);
lcd.print("Software ver:");
lcd.setCursor(6,1);
lcd.print("1.1");
delay(2000); // pauza 2s

lcd.clear();
delay(1000);
lcd.setCursor(2,0);
lcd.print("INITIALIZING");
lcd.setCursor(3,1);
lcd.print("PLEASE WAIT");
delay(1000);

lcd.clear();
delay(1000);
lcd.setCursor(2,0);
lcd.print("INITIALIZING");
lcd.setCursor(3,1);
lcd.print("PLEASE WAIT");
delay(1000);

lcd.clear();
delay(1000);
lcd.setCursor(2,0);
lcd.print("INITIALIZING");
lcd.setCursor(3,1);
lcd.print("PLEASE WAIT");
delay(1000);
lcd.clear();
```

```
  digitalWrite(LED1, HIGH); // uključi LED status READY
  digitalWrite(BUZZER, LOW); // isključi zujalicu
   lcd.setCursor( 0, 0 );
   lcd.print( "T:       H:"); // ispis oznake temperature i vlažnosti
nakon završetka inicijalizacije

}
```

Unutar petlje `void loop ()` , koja se ciklički ponavlja, potrebno je napisati programski kod koji će se izvršavati kada završi proces inicijalizacije. Nakon petlje `vodi loop ()` moguće je definirati pomoćne funkcije koje se koriste unutar glavne petlje, poput `ReadButtons()`. Unutar glavne petlje izrađen je dio programskog koda koji predstavlja izbornik samog sustava upravljanja te njegovu reakciju na pritisak pojedine tipke na tipkovnici. Programski kod izbornika za sustav upravljanja automatskim zalijevanjem biljke prikazan je u nastavku:

```
void loop()
{
   byte button; // definiranje korištene varijable

   button = ReadButtons();  // pokretanje pomoćne funkcije za čitanje
pritisnute tipke na tipkovnici
   switch( button ) // provjera rezultata pomoćne funkcije
   {
     case BUTTON_SELECT: // slučaj pritisnute tipke SELECT
      {
        menu = true;   // menu mod aktivan
        menu_number=0; // postavljanje prikaza 1. Meni-a
        lcd.clear();
        delay(200);
        while(menu ==true // dok je menu mod aktivan        {
          button = ReadButtons();   // pokretanje pomoćne funkcije za
čitanje pritisnute tipke na tipkovnici
          switch( button) //provjera rezultata pomoćne funkcije
          {
            case BUTTON_LEFT: //slučaj kada je pritisnuta tipka LEFT
             {
               lcd.clear();
               menu_number = menu_number - 1; // prebacivanje na
prethodni menu
               if(menu_number <0 ) menu_number = 3; // ako smo došli
do kraja, ponovno krećeno od 1. meni-a
               delay(200);
               break;
             }
            case BUTTON_UP: //slučaj kada je pritisnuta tipka UP
            {
              lcd.clear();
              menu_number = menu_number + 1; // prebacivanje na
slijedeći meni
              if(menu_number > 3) menu_number = 0;     // ako smo
došli do kraja, ponovno krećemo od 1. meni-a
              delay(200);
              break;
```

```
              }
             case BUTTON_SELECT: // slučaj pritisnute tipke SELECT
             {
               lcd.clear();
               delay(200);
               switch(menu_number) // provjera odabranog broja meni-a
               {

     //SET TEMP menu

                 case MENU_SETTEMP: // odabran slučaj SET TEMPERATURE
                 {
                   set_temp = true;       // zastavica set_temp
postavljena
                   lcd.clear();
                   while(set_temp == true) // dok je postavljena
zastavica set_temp vrti petlju
                   {
                     lcd.setCursor(0,0);
                     lcd.print("Set TEMPERATURE");
                     lcd.setCursor(0,1);
                     lcd.print(set_temperature); //ispis postavljene
temperature na LCD
                     button = ReadButtons(); //pokretanje pomoćne
funkcije za čitanje pritisnute tipke na tipkovnici
                     switch(button) //provjera rezultata funkcije
                     {
                       case BUTTON_UP: // slučaj kada je pritisnuta
tipka UP
                       {
                         set_temperature = set_temperature + 0.5; //
dodaj zadanoj temperaturi 0.5 °C
                         lcd.setCursor(0,1);
                         lcd.print(set_temperature); // ispis zadane
temperature na LCD
                         delay(200);
                         break; //prekid
                       }
                       case BUTTON_DOWN: // slučaj kada je pritisnuta
tipka DOWN
                       {
                         set_temperature = set_temperature - 0.5; //
oduzmi zadanoj temperaturi 0.5°C
                         lcd.setCursor(0,1);
                         lcd.print(set_temperature); // ispis zadane
temperature na LCD zaslon
                         delay(200);
                         break; //prekid
                       }
                       case BUTTON_SELECT: // slučaj kada je
pritisnuta tipka SELECT
                       {
                         lcd.clear();
                         set_temp = false; // postavljanje
temperature završeno, obriši zastavicu set_temp
                         delay(200);
                         break; //prekid
                       }
```

```
                break; //prekid
              }
            }
          break;
          }

//menu SET MOD RADA

          case MENU_MOD_RADA:
          {
            mod_rada = true;

            while(mod_rada == true)
            {
              lcd.setCursor(0,0);
              lcd.print("Set MOD RADA");
              lcd.setCursor(0,1);
              if(mod==0)
              {

               lcd.print("AUTO  ");
               auto_mod=true;
               manual_mod=false;
              }
              else
              {

                lcd.print("MANUAL");
                auto_mod=false;
                manual_mod=true;
                }
              button = ReadButtons();
              switch(button)
              {
                case BUTTON_DOWN:
                {
                  mod = mod - 1;

                  if(mod < 0) mod = 1;
                  lcd.setCursor(0,1);
                     if(mod==0)
                    {

                      lcd.print("AUTO  ");
                      auto_mod=true;
                      manual_mod=false;
                      }
                    else
                   {

                      lcd.print("MANUAL");
                      auto_mod=false;
                      manual_mod=true;
                    }
                  delay(200);
                  break;
                }
                case BUTTON_UP:
```

```
                {
                  mod = mod + 1;

                  if(mod > 1) mod = 0;
                  lcd.setCursor(0,1);
                     if(mod==0)
                     {

                       lcd.print("AUTO  ");
                       auto_mod=true;
                       manual_mod=false;
                     }
                       else
                     {

                       lcd.print("MANUAL");
                       auto_mod=false;
                       manual_mod=true;
                     }
                  delay(200);
                  break;
                }
                case BUTTON_SELECT:
                {
                  lcd.clear();
                  mod_rada = false;
                  delay(200);
                  break;
                }
                break;
              }
            }
          break;
          }

//menu SET MOISTRUE - vlažnost tla

        case MENU_SETMOISTURE:
          {
            set_soil = true;
            lcd.clear();
            while(set_soil == true)
            {
              lcd.setCursor(0,0);
              lcd.print("Set MOISTURE");
              lcd.setCursor(0,1);
              lcd.print(set_moisture);
              button = ReadButtons();
              switch(button)
              {
                case BUTTON_UP:
                {
                  set_moisture = set_moisture + 0.5;
                  lcd.setCursor(0,1);
                  lcd.print(set_moisture);
                  delay(200);
                  break;
                }
```

```
                    case BUTTON_LEFT:
                    {
                      set_moisture = set_moisture - 0.5;
                      lcd.setCursor(0,1);
                      lcd.print(set_moisture);
                      delay(200);
                      break;
                    }
                    case BUTTON_SELECT:
                    {
                      lcd.clear();
                      set_soil = false;
                      delay(200);
                      break;
                    }
                    break;
                  }
                }
              break;
              }

      //  menu EXIT - izlaz iz menu-a

              case MENU_EXIT:
              {

                menu=false;
                lcd.clear();
                delay(200);
                break;
              }
              break;
            }

          }
        break;
        }
       lcd.setCursor(0,0);
       lcd.print("      MENU    ");
       lcd.setCursor(0,1);
       lcd.print(MENU[menu_number]);
      }

   }

   lcd.clear();
   temp = millis()/1000;

}
```

Programski kod se nastavlja dijelom za mjerenje temperature pomoću digitalnog osjetila DS18B20 i korištenje 1 – wire bus protokola te mjerenje vlažnosti tla pomoću osjetila za mjerenje vlažnosti. Spomenuti dio koda s pripadajućim komentarima se nalazi u nastavku:

```
/*--------------------------------------------------------------------
DS18B20
--------------------------------------------------------------------*/

  byte i;
  byte present = 0;
  byte type_s;
  byte data[12];
  byte addr[8];
  float celsius;

 if ( !ds.search(addr)) { //pretraga senzora na 1-wire bus protokolu
    ds.reset_search();
    return;
  }

    switch (addr[0]) {
     case 0x28: // adresa za slučaj senzora DS18B20
      type_s = 0;
      break;
    default:
    return;
}

  ds.reset();
  ds.select(addr);
  ds.write(0x44,1);            // zahtjev za čitanje temperature sa
senzora i spremanje u RAM memoriju senzora

  delay(1000);         // potrebna pauza zbog konvertiranja (750ms
potrebno, 1000ms zbog rezerve)

  present = ds.reset();
  ds.select(addr);
  ds.write(0xBE);             // zahtjev za dohvat podataka iz RAM
memorije senzora

    for ( i = 0; i < 9; i++) {           // Byte0 - Byte8
    data[i] = ds.read();
      }

  // konvertiranje učitanih podataka u temperaturu

  unsigned int raw = (data[1] << 8) | data[0];
  if (type_s) {
    raw = raw << 3; // 9 bit resolution default
    if (data[7] == 0x10) {
      // count remain gives full 12 bit resolution
      raw = (raw & 0xFFF0) + 12 - data[6];
    }
  } else {
    byte cfg = (data[4] & 0x60);
    if (cfg == 0x00) raw = raw << 3;  // 9 bit resolution, 93.75 ms
    else if (cfg == 0x20) raw = raw << 2; // 10 bit res, 187.5 ms
    else if (cfg == 0x40) raw = raw << 1; // 11 bit res, 375 ms
    // default is 12 bit resolution, 750 ms conversion time
  }
```

```
  celsius = (float)raw / 16.0; // ocitanu vrijednost podijeliti sa 16
zbog rezolucije (2^4=16), može se pomnožiti i s 0.0625

  int  soil_s  =  analogRead(SOIL_SENSOR);  //  čitanje  vrijednosti
vlaznosti sa senzora (1023-0 = 10 bit)
  float soil =( 1023-soil_s)   / 10.24;// podijeliti s 10.24 da se
dobije postotak

//ispis učitanih vrijednosti sa senzora na LCD zaslon

   lcd.setCursor( 0, 0 );
   lcd.print( "T:       H:" );
   lcd.setCursor(2,0);
   lcd.print(celsius);
   lcd.setCursor(6,0);
   lcd.print("\337C"); //znak °C
   lcd.setCursor(11,0);
   lcd.print(soil);
   lcd.setCursor(15,0);
   lcd.print("%");
```

U slučaju da se dogodi prekid komunikacije između senzora temperatura i analognog ulaza Arduino DUE razvojne platforme (slučaj neispravnosti senzora ili prekinutih vodiča) potrebno je oglasiti alarm. Alarm označava prekid svih radnji na relejima (gašenje pumpe i solenoidnih ventila) te indikaciju paljenjem crvene LED diode ERROR, paljenjem zujalice pomoću NPN tranzistora BD139 te prikazom poruke na LCD zaslonu. Dio programskog koda za slučaj alarma naveden je u nastavku:

```
//TEMP error

  if(data[6]==255) //data=0 FF FF FF FF FF FF FF FF FF senzor odpojen
4095.94°C
  {
    error_flag=true; // zastavica error postavljena
    lcd.clear();
    digitalWrite(LED1, LOW); // gasi zelenu led diodu ready
    digitalWrite(LED2, HIGH); // pali crvenu led diodu error
    digitalWrite(BUZZER, HIGH); // pali zujalicu
    digitalWrite(SOLENOID, RELAY_OFF); // gasi solenoidni ventil
    digitalWrite(PUMP, RELAY_OFF); //gasi pumpu
    lcd.setCursor(4,0);
    lcd.print("ERROR !!"); //ispis poruke o greški na LCD zaslon
    lcd.setCursor(0,1);
    lcd.print("                ");
    lcd.setCursor(2,1);
    lcd.print("TEMP  SENSOR");
    }
    else  //kada se greška otkloni
    {
    digitalWrite(LED1, HIGH); //pali zelenu led diodu ready
    digitalWrite(LED2, LOW); // gasi crvenu led diodu error
    digitalWrite(BUZZER, LOW); //gasi zujalicu
    error_flag=false; //brisanje error zastavice
    }
```

Završni dio programskog koda bavi se načinima rada samog upravljanja, ručni način gdje upravljanjem zalijevanja biljke upravlja operater te automatski način rada gdje se zalijevanje biljke obavlja ovisno o stanju osjetila vlažnosti tla i temperature tekućine u spremniku. Upravljanje pumpom i solenoidnim ventilom ovisno o načinu rada opisano je u programskom kodu u nastavku:

```
/*-------------------------------------------------------------------
    MAIN PROGRAM
 -------------------------------------------------------------------*/

//MANUAL mod rada

 if(manual_mod == true && error_flag==false) //provjera ako nema
greške i ako se postrojenje nalazi u ručnom načinu rada

 {
   if (digitalRead(SENZOR_H) != LOW) // ako spremnik nije pun
   {
   auto_mod=false; //brisanje zastavica automatskog načina rada
   digitalWrite(SOLENOID, RELAY_ON); //pali solenoidni ventil
   digitalWrite(PUMP, RELAY_ON); //pali pumpu za dovod u spremnik
   lcd.setCursor(0,0);
   lcd.print("                "); //ispis na LCD zaslon
   lcd.setCursor(3,0);
   lcd.print("MANUAL MOD");
   lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(0,1);
   lcd.print("PUMP & SVALVE ON");
   }
  else if (digitalRead(SENZOR_H) == LOW) //ako se spremnik napunio,
slučaj kada je punjenje brže nego pražnjenje
  {
   digitalWrite(PUMP, RELAY_OFF); // gasi pumpu za dovod u spremnik
   lcd.setCursor(0,0);
   lcd.print("                ");
   lcd.setCursor(2,0);
   lcd.print("SPREMNIK PUN"); //ispis na LCD zaslon
   lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(2,1);
   lcd.print("SOLENOID  ON");
   delay(4000); //pauza 4s zbog praznjenja spremnika
   }
 else // ako je prekinut ručni mod rada
 {
   digitalWrite(SOLENOID, RELAY_OFF); //gasi solenoidni ventil
   digitalWrite(PUMP, RELAY_OFF);//gasi pumpu
  }
  }
```

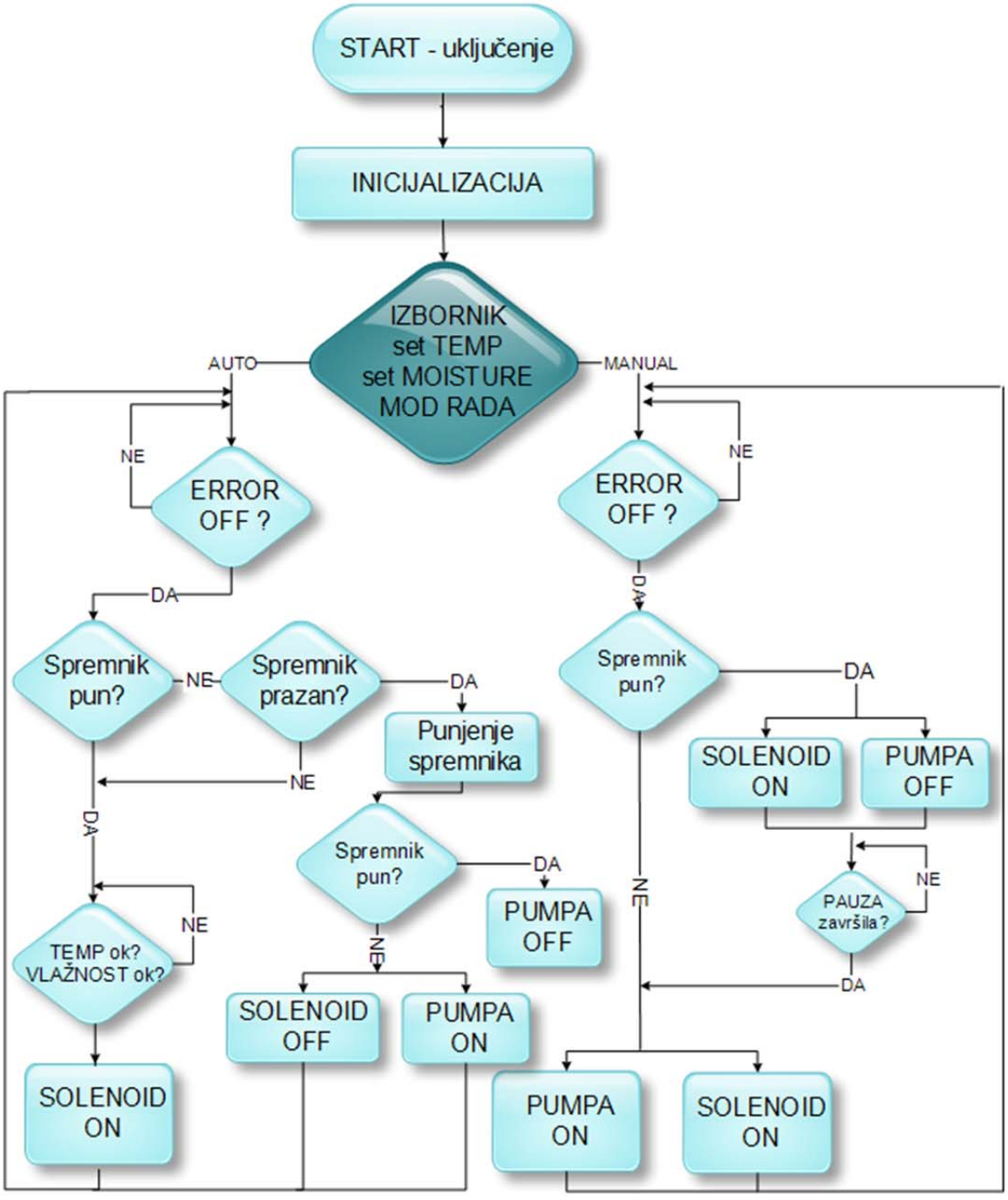
```
//AUTO mod rada

if (auto_mod==true && error_flag==false) //provjera ako se
postrojenje nalazi u automatskom načinu rada i ako nema greške
{

   lcd.setCursor(0,1);
   lcd.print("                "); //ispis stanja na LCD zaslon
   lcd.setCursor(4,1);
   lcd.print("AUTO MOD");
    manual_mod=false; //zastavica ručnog načina rada obrisana
  if (puni == true ) //ako je slučaj za punjenje spremnika
   {
   digitalWrite(PUMP, RELAY_ON); //pali pumpu
   digitalWrite(SOLENOID, RELAY_OFF);//gasi solenoidni ventil

     if ((digitalRead(SENZOR_H)== LOW) && (digitalRead(SENZOR_L) ==
HIGH))//spremnik pun
 {
   puni=false; //brisanje zastavice za punjenje
   prazni=true; // postavljanje zastavice za pražnjenje
   lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(1,1);
   lcd.print("SPREMNIK PUN"); //ispis na LCD
   }
   }
   if (prazni == true  ) // ako je slučaj za pražnjenje spremnika
   {
   {
     digitalWrite(PUMP, RELAY_OFF); //gasi pumpu
    }
     if (celsius >= set_temperature && soil <= set_moisture) // ako
je zadovoljen uvjet temperature u spremniku i vlažnosti tla
     {
     digitalWrite(SOLENOID, RELAY_ON); //pali solenoidni ventil i
prazni spremnik
    }
      if ((digitalRead(SENZOR_L)== LOW) && (digitalRead(SENZOR_H) ==
HIGH))//spremnik prazan
 {
   puni=true; //postavljanje zastavice za punjenje
   prazni=false; //brisanje zastavice za pražnjenje
    lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(0,1);
   lcd.print("SPREMNIK PRAZAN"); // ispis stanja na LCD
   }
     }

}

}
```

Nakon završetka glavne petlje `voip loop ()` definira se pomoćna funkcija za čitanje pritisnute tipke na tipkovnici `ReadButtons()` prikazana u nastavku:

```
/*--------------------------------------------------------------------
ReadButtons()
--------------------------------------------------------------------*/

byte ReadButtons()
{
   float buttonVoltage;
   byte button = BUTTON_NONE;   // nije pritisnuta nijedna tipka


   buttonVoltage   =   analogRead(   BUTTON_ADC_PIN   );   //   čitanje
vrijednosti s analognog ulaza

      if( buttonVoltage < ( RIGHT_10BIT_ADC + BUTTONHYSTERESIS ) )
   {
      button = BUTTON_RIGHT; // ako je pročitana vrijednost manja od
50 radi se od tipki RIGHT
   }
   else if(   buttonVoltage >= ( UP_10BIT_ADC - BUTTONHYSTERESIS )
          && buttonVoltage <= ( UP_10BIT_ADC + BUTTONHYSTERESIS ) )
   {
      button = BUTTON_UP; // ako je pročitana vrijednost između 165 i
265 radi se od tipki UP
   }
   else if(   buttonVoltage >= ( DOWN_10BIT_ADC - BUTTONHYSTERESIS )
          && buttonVoltage <= ( DOWN_10BIT_ADC + BUTTONHYSTERESIS ) )
   {
      button = BUTTON_DOWN;
   }
   else if(   buttonVoltage >= ( LEFT_10BIT_ADC - BUTTONHYSTERESIS )
          &&buttonVoltage <= ( LEFT_10BIT_ADC + BUTTONHYSTERESIS ) )
   {
      button = BUTTON_LEFT;
   }
   else if(   buttonVoltage >=  SELECT_10BIT_ADC - BUTTONHYSTERESIS )
        && buttonVoltage <= ( SELECT_10BIT_ADC + BUTTONHYSTERESIS ) )

   {
      button = BUTTON_SELECT;
   }
      if( ( buttonWas == BUTTON_NONE ) && ( button != BUTTON_NONE ) )
   {
      buttonJustPressed  = true;
      buttonJustReleased = false;
   }
   if( ( buttonWas != BUTTON_NONE ) && ( button == BUTTON_NONE ) )
   {
      buttonJustPressed  = false;
      buttonJustReleased = true;
   }
    buttonWas = button;

   return( button );
}
```

4.3. Prikaz rada sustava upravljanja automatskog zalijevanja biljke

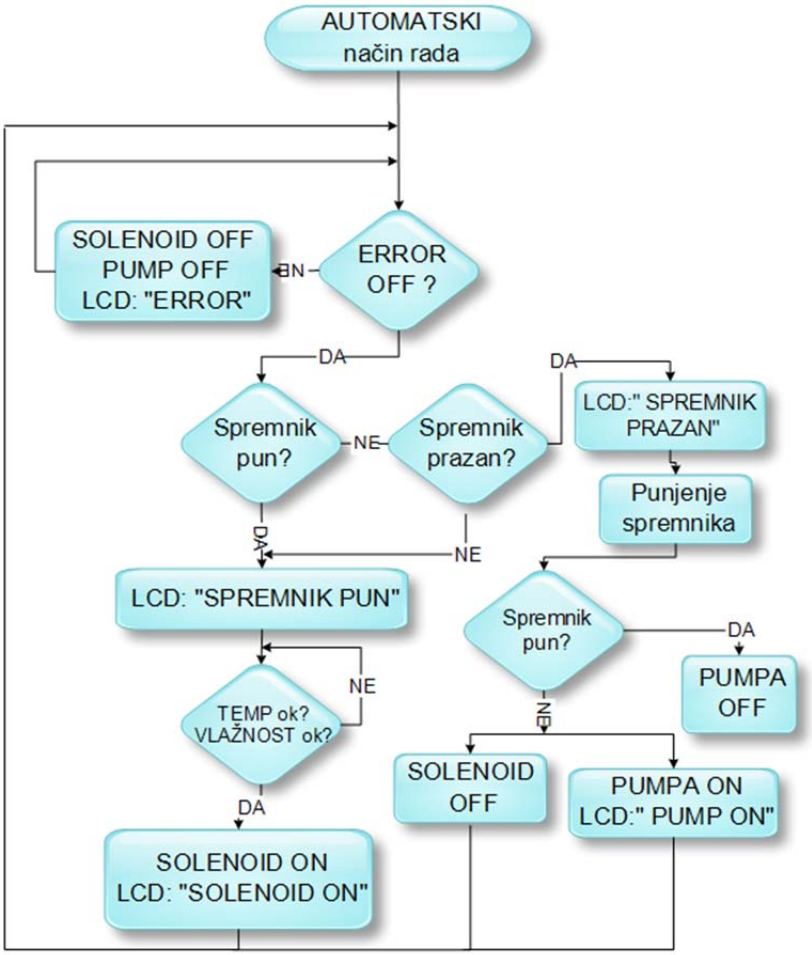
Prikaz rada sustava upravljanja automatskog zalijevanja biljke prikazan je na dijagramu toka u nastavku (slika 16).



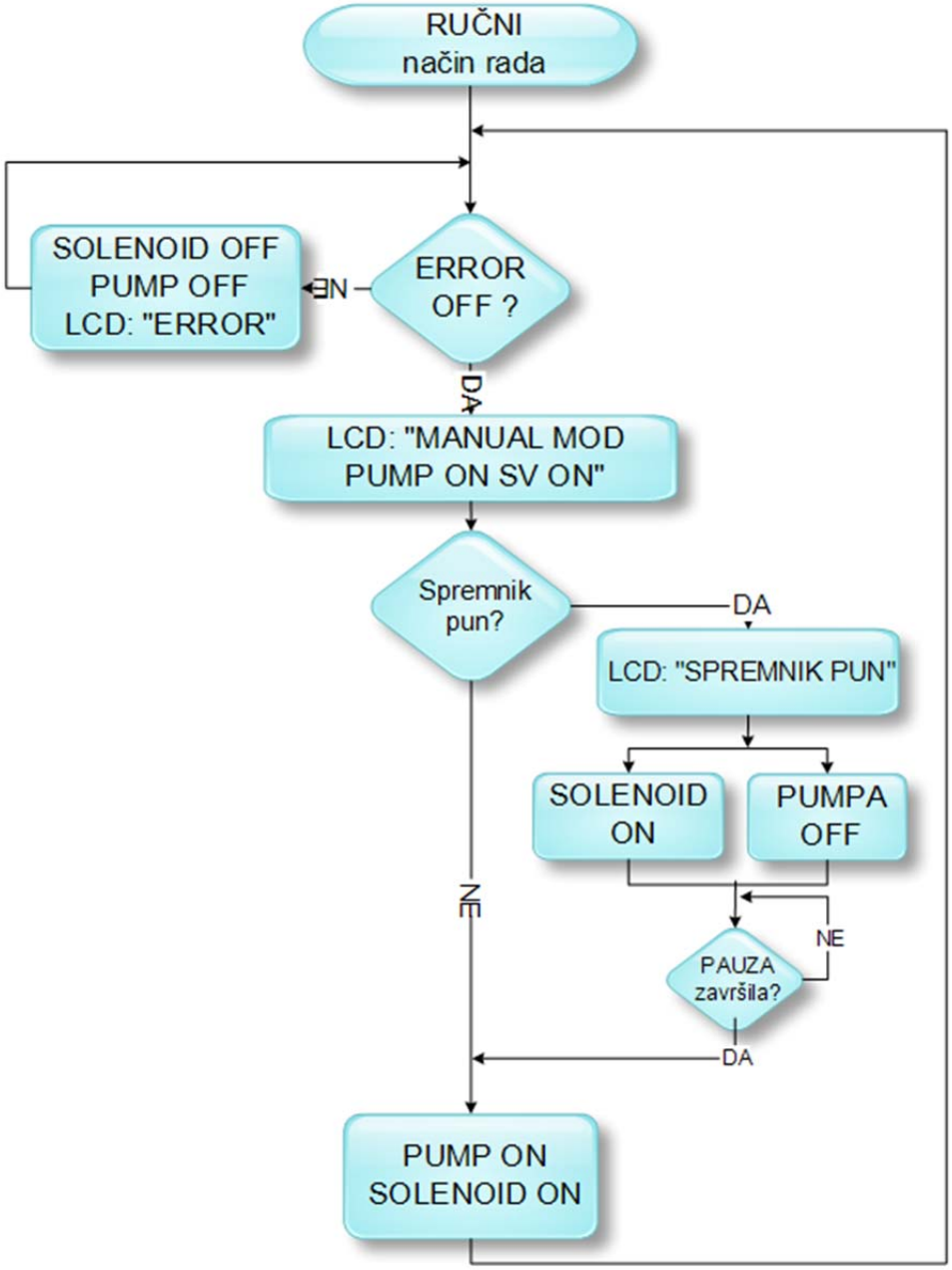
Slika 16: Dijagram toka rada sustava upravljanja

Nakon uključenja postrojenja pomoću sklopke na upravljačkoj jedinici pokreće se inicijalizacija samog sustava upravljanja. Kada je inicijalizacija sustava uspješno završila, sustav upravljanja se pokreće u automatskom načinu rada. Pomoću izbornika podešavaju se parametri

sustava upravljanja i način rada samog sustava (ručno ili automatski). U automatskom načinu rada (slika 17) najprije se provjerava stanje sustava upravljanja, odnosno je li sustav u normalnom stanju ili u stanju greške. Ukoliko je sustav u normalnom stanju, provjerava se stanje spremnika (pun ili prazan). Ako je spremnik prazan, obavlja se punjenje spremnika tekućinom sve dok se spremnik napuni. Kada je spremnik pun, provjerava se stanje temperature tekućine u spremniku te stanje vlažnosti tla oko biljke koja se zalijeva. Ukoliko rezultati mjerenja temperature u spremniku i vlažnosti tla zadovoljavaju postavljene vrijednosti, obavlja se zalijevanje biljke tekućinom iz spremnika. U ručnom načinu rada (slika 18) najprije se provjerava stanje sustava upravljanja te ukoliko je sustav u normalnom stanju, provjerava se stanje napunjenosti spremnika. Ako spremnik nije pun, uključuju se pumpa za dovođenje tekućine u spremnik te solenoidni ventil i pumpa za odvođenje tekućine iz spremnika. Ukoliko se spremnik napuni, dovođenje tekućine u spremnik se pauzira (gašenje pumpe za dovođenje tekućine u spremnik) unaprijed određeno vrijeme te se nakon isteka tog vremena ponovno pokreće. Sustav upravljanja ostaje u zadanom načinu rada sve dok operater pomoću izbornika ne odabere drugi način rada sustava.

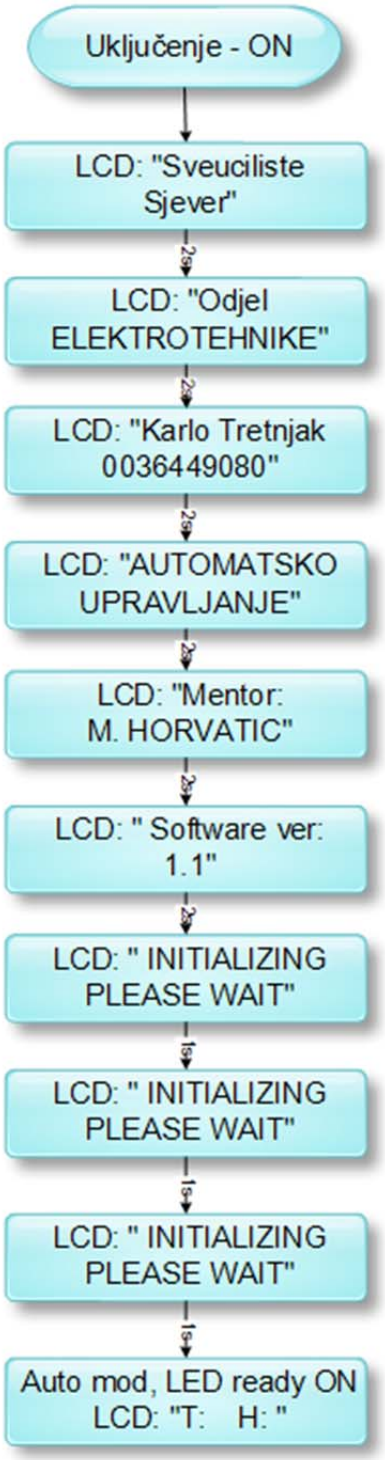


Slika 17: Dijagram toka automatskog načina rada sustava

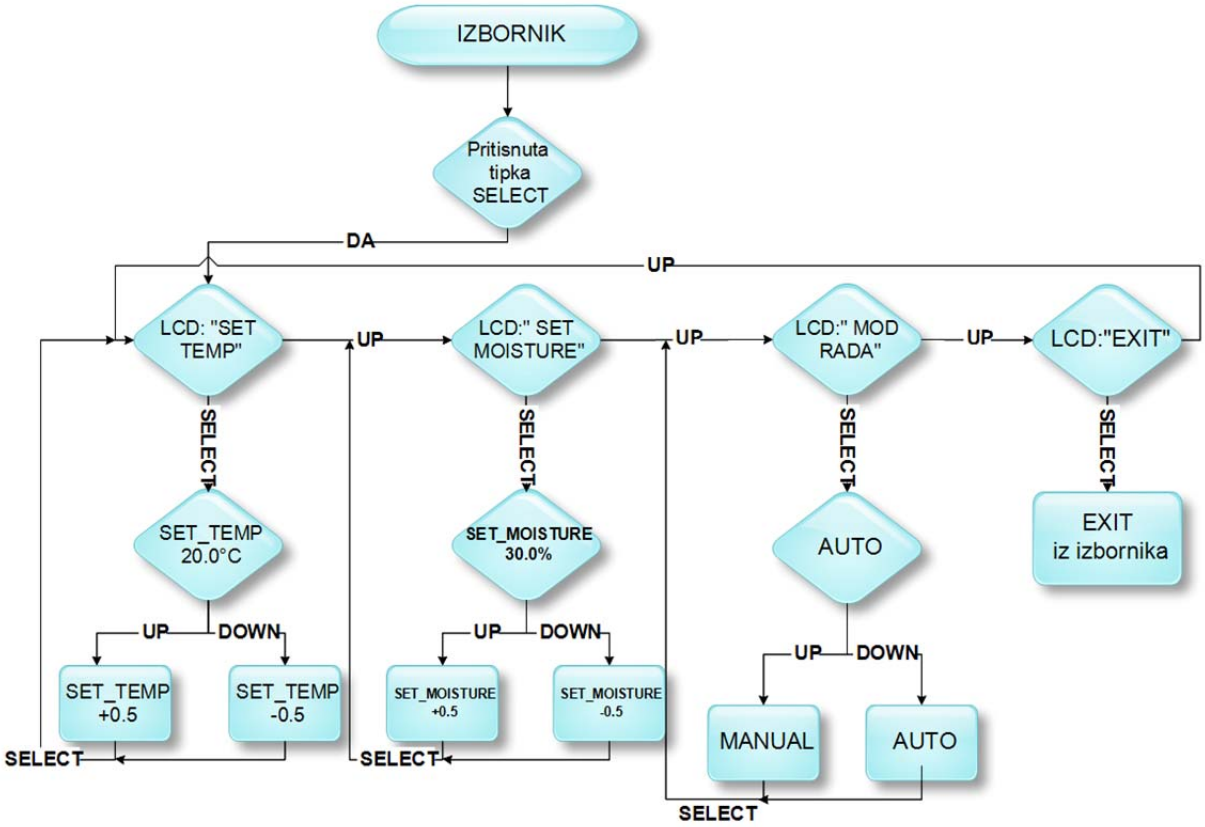


Slika 18: Dijagram toka ručnog načina rada sustava

Nakon uključenja sustava upravljanja obavlja se inicijalizacija sustava upravljanja te se pomoću prikaza na LCD zaslonu prikazuju osnovne informacije o sustavu upravljanja (slika 19) i provjerava se inicijalno stanje magnetskih osjetila razine u spremniku. Pritiskom tipke SELECT omogućen je ulazak u izbornik sustava upravljanja. Izbornik sustava upravljanja (slika 20) sastoji se od podizbornika za podešavanje temperature u spremniku, vlažnosti tla i načina rada sustava te podizbornika za izlazak iz izbornika.



Slika 19: Dijagram toka ispisa na LCD zaslonu nakon uključenja



Slika 20: Dijagram toka izbornika

5. Zaključak

Svakodnevna potreba za zalijevanjem biljaka kako u vlastitom vrtu tako i na drugim poljoprivrednim površinama dovela je do razvoja sustava za automatsko zalijevanje biljaka. Ideja da se radnja zalijevanja biljaka automatizira i samim time poveća iskoristivost samog zalijevanja iskorištena je u ovom radu za projektiranje i realizaciju sustava za automatsko zalijevanje biljaka pomoću neke mikrokontrolerske platforme.

Odabirom cijenom prihvatljive Arduino DUE razvojne platforme dobiva se relativno snažna platforma za realizaciju samog sustava upravljanja. Svojim mogućnostima prelazi potrebe samog sustava, no to omogućuje bezbolno kasnije nadograđivanje i prilagođavanje sustava potrebama korisnika. Dostupnost knjižnica (library-a) za proširenja sustava, jednostavnost programskog jezika C te jaka podrška u vidu foruma i razmjene iskustava s ostalim korisnicima, omogućuje jednostavno programiranje same razvojne platforme i lako rješavanje novonastalih problema. Modularnost same razvojne platforme omogućava proširenje i nadogradnju sustava upravljanja kada dođe do potrebe za time (Ethernet, WiFi shield ...).

Izradom makete sustava upravljanja za automatsko zalijevanje biljke omogućen je prikaz rada sustava upravljanja. Uključenjem makete sustava provodi se inicijalizacija sustava upravljanja i pritom se učitavaju stanja magnetskih osjetila u spremniku te se sustav upravljanja pokreće u automatskom načinu rada. Pomoću izbornika , tipki i prikaza na LCD zaslonu omogućeno je podešavanje parametara samog upravljanja. U automatskom načinu rada sustava zalijevanje biljke obavlja se ovisno o stanju vlažnosti tla te temperature u spremniku. Kod ručnog načina rada sustava zalijevanje biljke se obavlja bez obzira na stanje vlažnosti tla i temperature u spremniku, odnosno operater upravlja zalijevanjem.

Projektiranjem i samom realizacijom sustava upravljanja za automatsko zalijevanje biljaka stečena su nova znanja i iskustva iz područja programiranja te automatskog upravljanja sustavom. Izrađena maketa sustava upravljanja pokazuje sustav za zalijevanje koji se može upotrijebiti u svakodnevnom životu kod zalijevanja vlastitog vrta i sličnih površina. Nadogradnjom sustava s Ethernet ili WiFi shieldom (proširenjem) te izradom web i mobilnog korisničkog sučelja omogućava se upravljanje samim sustavom upravljanja na daljinu, odnosno preko Interneta.

6. Literatura

[1] https://www.arduino.cc/en/Main/ArduinoBoardDue (Dostupno 12.9.2015.)

[2] http://www.chicagosensor.com/HowFloatSwitchesWork.html (Dostupno 12.9.2015.)

[3] https://tushev.org/articles/arduino/10 (Dostupno 12.9.2015.)

[4] http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf (Dostupno 12.9.2015.)

[5] http://smart-prototyping.com/Soil-Hygrometer-Detection-Module-Soil-Moisture-Sensor-For-Arduino.html (Dostupno 13.9.2015.)

[6] http://electronica.mk/all_articles/Electric_Car_Projects/50A_PWM_DC_Motor_Speed_Controll er/50A_DC_Controller.html (Dostupno 13.9.2015.)

[7] http://www.irf.com/product-info/datasheets/data/irf3205.pdf (Dostupno 13.9.2015.)

[8] http://www.adafruit.com/products/997 (Dostupno 13.9.2015.)

[9] M. Banzi: Getting started with Arduino, Second Edition, O'Reilly, Sebastopol, 2011

[10] C. Amariei: Arduino Development Cookbook, Packt Publishing, Birmingham, 2015

[11] J. Šribar, B. Motik: Demistificirani C++, Element, 4. izdanje, Zagreb, 2014

[12] http://www.24zakupy.com/userdata/gfx/835e0f724e667fb1935a6fdd1818da5e.jpg (Dostupno 13.9.2015.)

[13] http://ecx.images-amazon.com/images/I/41b-kKYOajL._SY355_.jpg (Dostupno 13.9.2015.)

[14] https://www.arduino.cc/en/uploads/Main/ArduinoDue_Front_450px.jpg (Dostupno 13.9.2015.)

[15] http://img2.dib-bid.com/gallery3/3/f3/55306/055306-1-05.jpg (Dostupno 13.9.2015.)

[16] https://tushev.org/images/electronics/arduino/ds18x20/DS18B20.jpg (Dostupno 13.9.2015.)

[17] http://www.arduiner.com/2685-thickbox_default/soil-hygrometer-detection-module-soil-moisture-sensor-arduino-compatibile.jpg (Dostupno 13.9.2015.)

[18] http://i.ebayimg.com/00/s/NjAwWDYwMA==/z/xFMAAOSwVupTrR~v/$_57.JPG (Dostupno 13.9.2015.)

[19] http://ibay365goodsimg.s3.amazonaws.com/1408/20120912/201209121207350ed44.jpg (Dostupno 13.9.2015.)

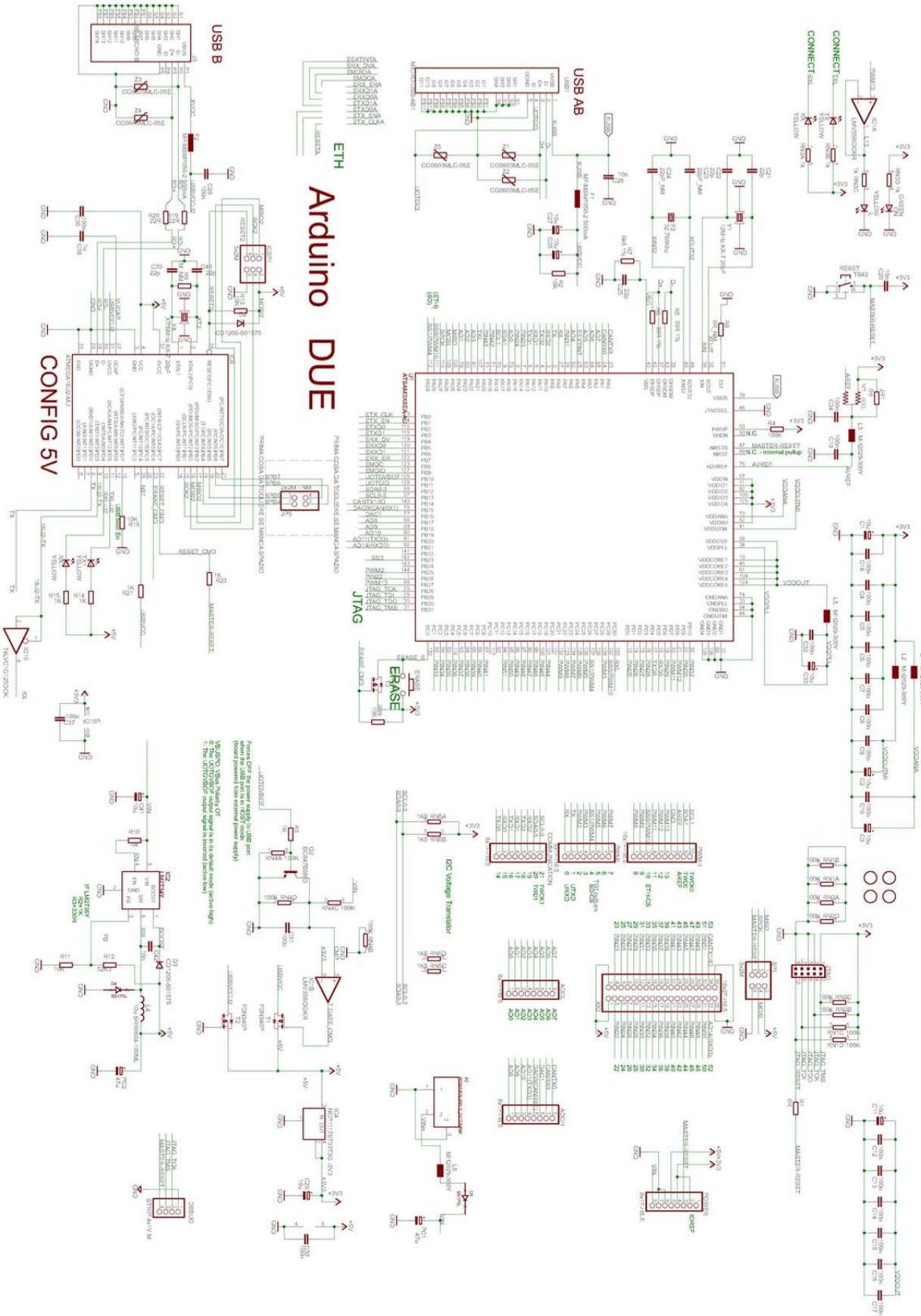
[20] http://shyskytech.en.ec21.com/12V_24V_Micro_Brushless_DC-8464416_8467706.html (Dostupno 18.9.2015.)

[21] http://lighthouseleds.com/12v-5a-led-power-supply-ac-to-dc-60-watts.html (Dostupno 18.9.2015.)

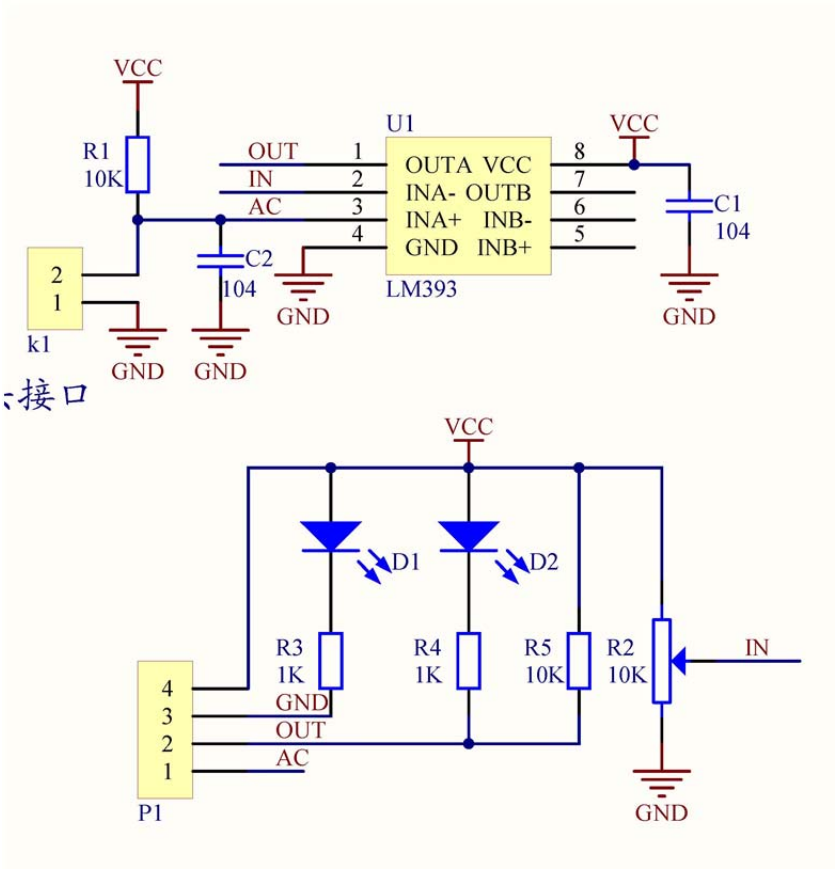
[22] https://www.arduino.cc/en/uploads/Main/arduino-Due-schematic.pdf (Dostupno 18.9.2015.)
[23] http://www.etang.co.uk/datasheet/solid%20humidity%20sensor/solid%20sensor%20schematics.jpg (Dostupno 18.9.2015.)
[24] http://hz00.i.aliimg.com/img/pb/554/738/559/559738554_544.jpg (Dostupno 18.9.2015.)
[25] http://electronica.mk/all_articles/Electric_Car_Projects/50A_PWM_DC_Motor_Speed_Controller/50A%20DC%20Motor%20Controller%20Circuit.png (Dostupno 18.9.2015.)

7. Prilozi

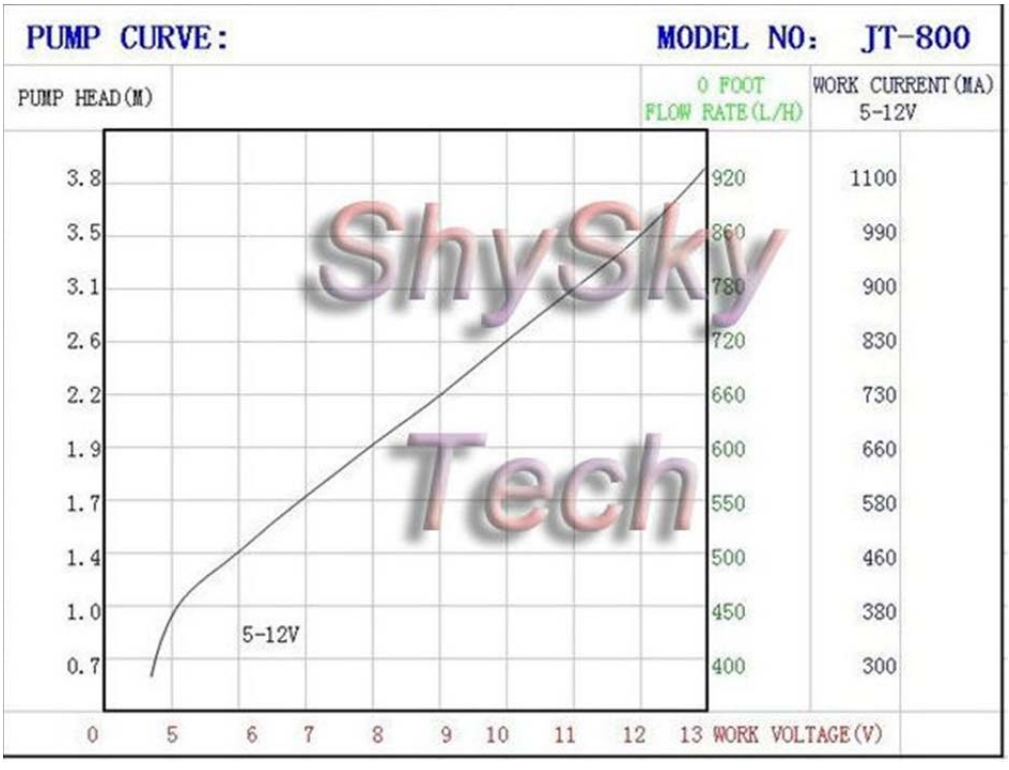
PRILOG 1: SHEMA ARDUINO DUE RAZVOJNE PLATFORME [22]



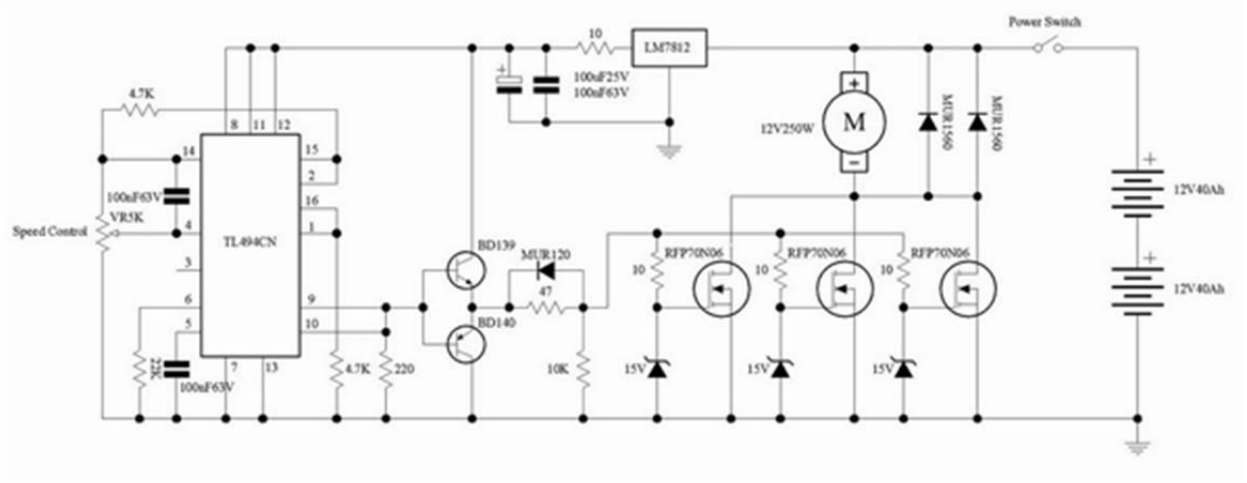
PRILOG 2: SHEMA ELEKTRONIKE SENZORA VLAŽNOSTI [23]



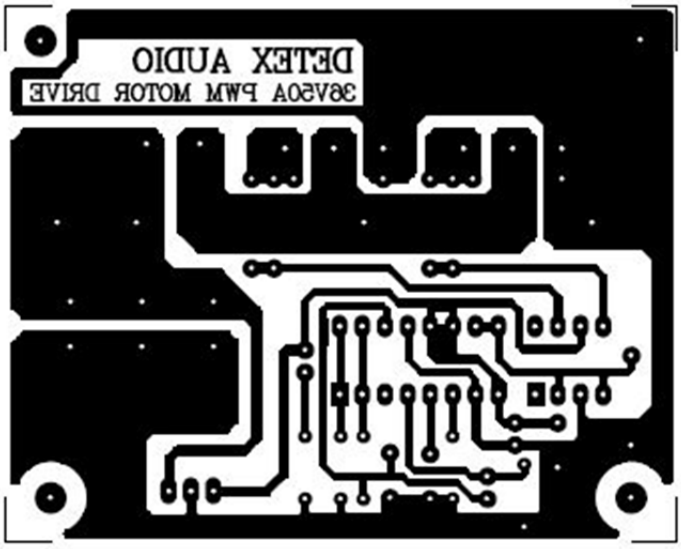
PRILOG 3: OVISNOST PROTOKA JT-800 PUMPE O NAPONU [24]



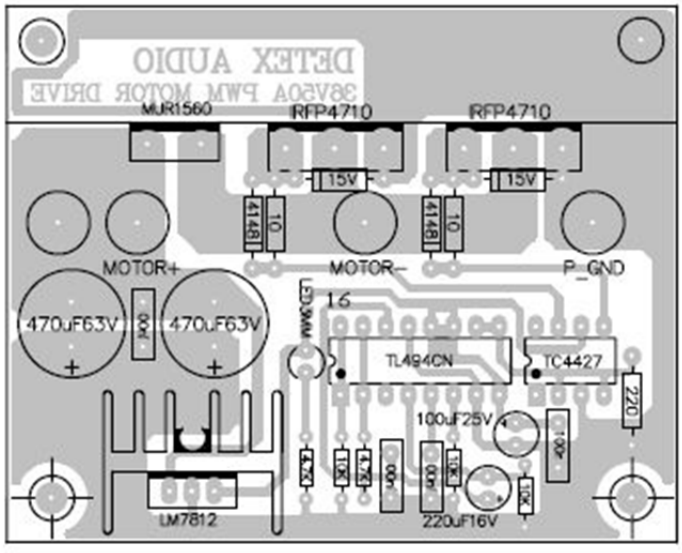
PRILOG 4: ELEKTRIČNA SHEMA PWM REGULATORA [25]



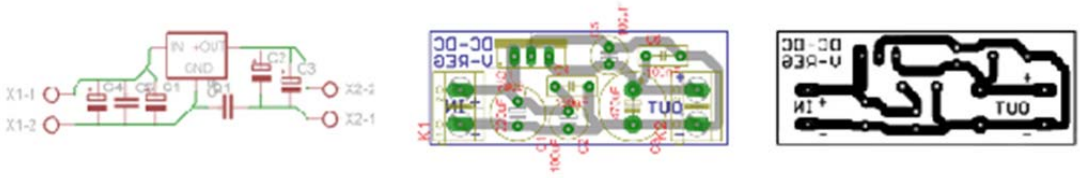
PRILOG 5: RASPOREDVODOVA PWM REGULATORA NA TISKANOJ PLOČICI



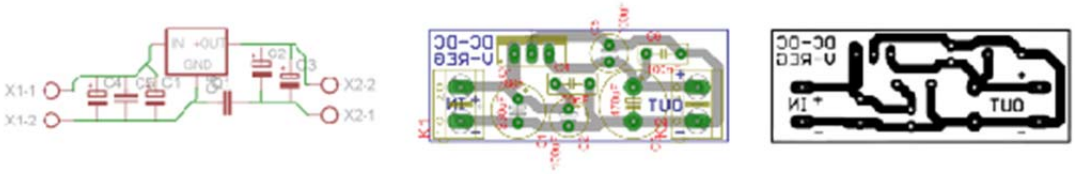
PRILOG 6: RASPOREDELEMENATA PWM REGULATORA NA TISKANOJ PLOČICI



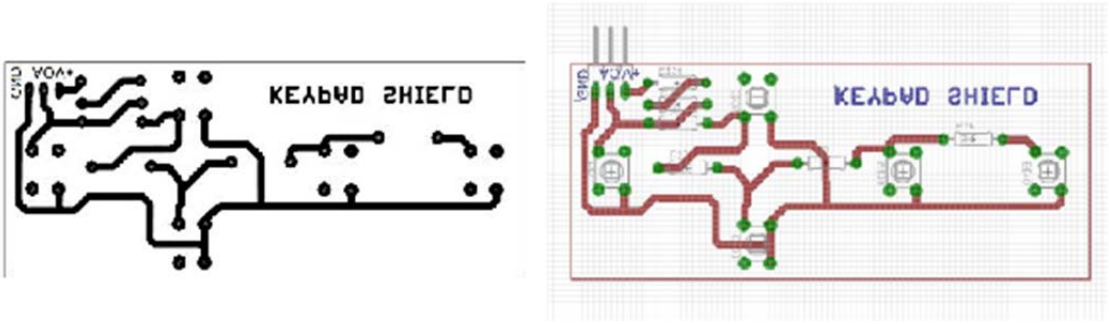
PRILOG 7: SHEMA, RASPORED NA TISKANOJ PLOČICI DC-DC REGULATORA



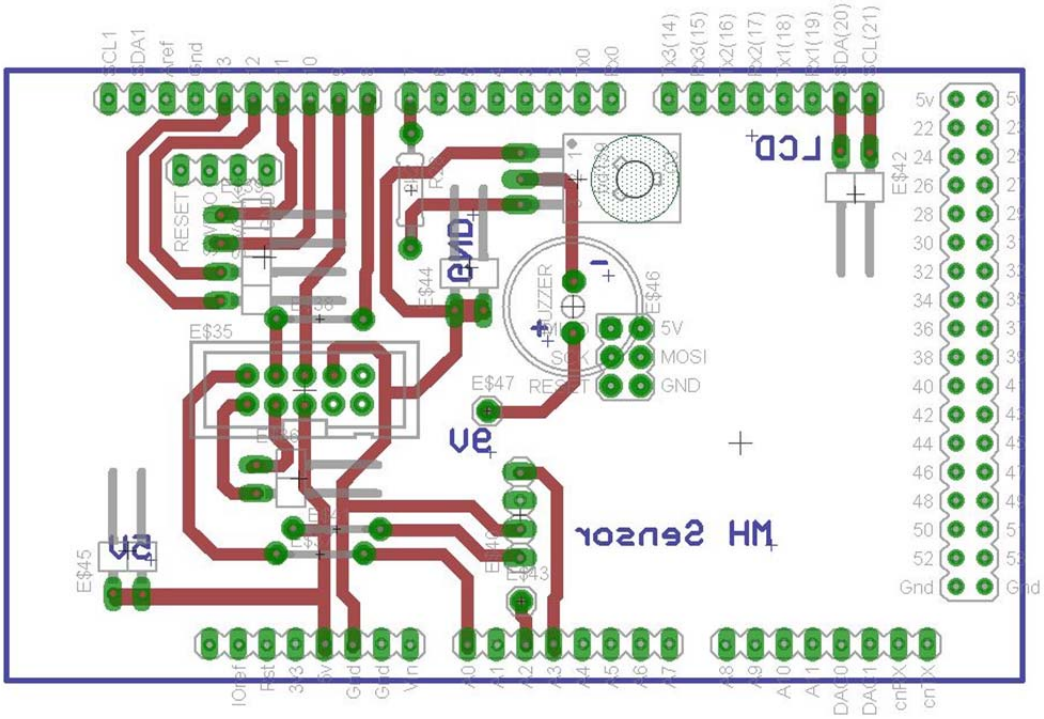
PRILOG 8: SHEMA, RASPORED NA TISKANOJ PLOČICI DC-DC REGULATORA



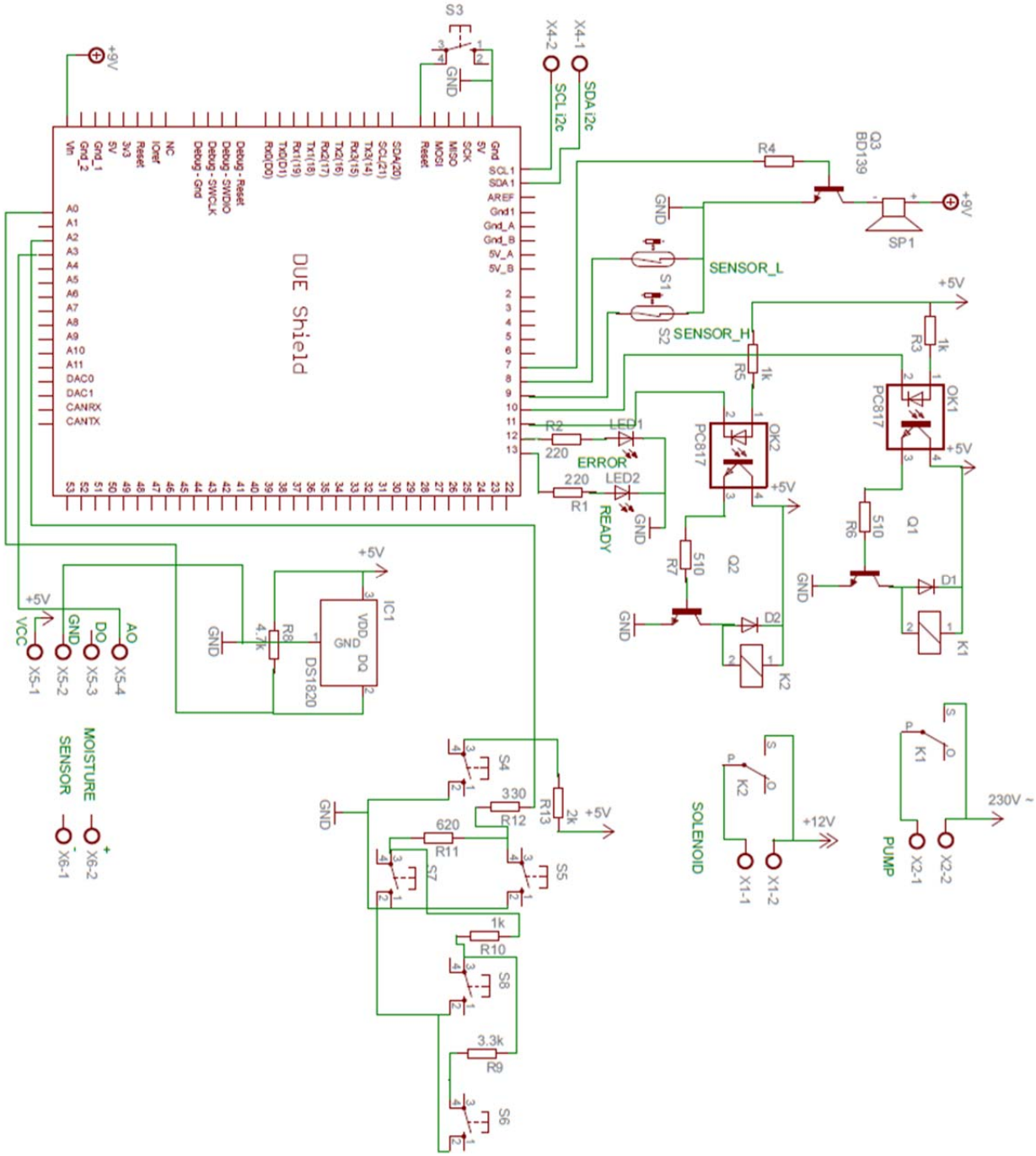
PRILOG 9: RASPORED ELEMENATA I VODOVA TIPKOVNICE



PRILOG 10: RASPORED ELEMENATA NA PLOČICI (SHIELD) ZA ARDUINO DUE



PRILOG 11: SHEMA SUSTAVA UPRAVLJANJA



PRILOG 12: PROGRAMSKI KOD AUTOMATSKOG UPRAVLJANJA ZALIJEVANJA BILJKE IMPLEMENTIRAN U ARDUINO DUE – finalna verzija

```
/*-----------------------------------------------------------------------
AUTOMATSKO NAVODNJAVANJE
-----------------------------------------------------------------------*/

/*-----------------------------------------------------------------------
  ADC voltages for the 5 buttons on analog input pin A1:

    LEFT:   2.40V : 492 @ 10 bit
    UP:     1.05V : 215 @ 10 bit
    DOWN:   1.60V : 328 @ 10 bit
    RIGHT:  0.00V :   0 @ 10 bit
    SELECT: 3.76V : 770 @ 10 bit

-----------------------------------------------------------------------*/

/* -----------------------------------------------------------------------
Includes
-----------------------------------------------------------------------*/
#include <Wire.h> //library for I2C LCD
#include <LiquidCrystal_I2C.h> //library for I2C LCD
#include <OneWire.h> //library for oneWire communication (DS18B20)

/*-----------------------------------------------------------------------
Defines
-----------------------------------------------------------------------*/
OneWire ds(54); // pin 54 A0 on DUE (DS18B20)
LiquidCrystal_I2C  lcd(0x27,  2,  1,  0,  4,  5,  6,  7,  3,  POSITIVE);
//lcd(addr, EN, RW, RS, D4, D5, D6, D7, B1, BLPOL) pinout I2C


//return values for ReadButtons()
#define BUTTON_NONE               0  //
#define BUTTON_RIGHT              1  //
#define BUTTON_UP                 2  //
#define BUTTON_DOWN               3  //
#define BUTTON_LEFT               4  //
#define BUTTON_SELECT             5  //

//define menus
#define MENU_SETTEMP              0
#define MENU_MOD_RADA             1
#define MENU_SETMOISTURE          2
#define MENU_EXIT                 3

// ADC readings expected for the 5 buttons on the ADC input
#define RIGHT_10BIT_ADC         0 // right
#define UP_10BIT_ADC            215  // up
#define DOWN_10BIT_ADC          328  // down
#define LEFT_10BIT_ADC          492  // left
#define SELECT_10BIT_ADC        770 // select
#define BUTTONHYSTERESIS          50  // hysteresis for valid button
sensing window
```

```
#define SOIL_SENSOR              A3
#define BUTTON_ADC_PIN           A2// A2 is the button ADC input
#define SENZOR_H 9
#define SENZOR_L 8
#define SOLENOID 10  //relay1 on pin 10 - solenoidni ventil
#define PUMP 11  //relay2 on pin 11 - pumpa
#define LED1 12 // LED1 on pin 12
#define LED2 13 // LED2 on pin 13
#define BUZZER 7 // buzzer on pin 7
#define RELAY_OFF  1
#define RELAY_ON   0

/*-----------------------------------------------------------------------
variables
-----------------------------------------------------------------------*/
byte buttonJustPressed  = false;          //this will be true after a
ReadButtons() call if triggered
byte buttonJustReleased = false;          //this will be true after a
ReadButtons() call if triggered
byte buttonWas          = BUTTON_NONE;
byte temp =0;
//used by ReadButtons() for detection of button events
boolean menu = true;
boolean set_temp = true;
boolean set_soil = true;
boolean mod_rada = true;
boolean auto_mod=true;
boolean manual_mod=false;
boolean error_flag=false;
boolean puni=true;
boolean prazni=true;
int menu_number = 0;
int mod=0;
float set_temperature = 20.0;
float set_moisture = 30.0;
char* MENU[] = {"Set TEMPERATURE", "Set MOD RADA", "Set MOISTURE",
"EXIT "};



/*-----------------------------------------------------------------------
SETUP
-----------------------------------------------------------------------*/

void setup()
{


    // deklaracija pinova

  pinMode( BUTTON_ADC_PIN, INPUT ); //A2 ulaz
  digitalWrite(  BUTTON_ADC_PIN,  LOW  );//disable  internal  pullup
resistor on A2
  pinMode(SOIL_SENSOR, INPUT); //A3 ulaz SOIL SENSOR

  pinMode (SENZOR_H, INPUT_PULLUP);// SENZOR GORE - ulaz
  pinMode (SENZOR_L, INPUT_PULLUP);// SENZOR DOLJE - ulaz
  pinMode (LED1, OUTPUT);  // LED1 - izlaz
```

```
  pinMode (LED2, OUTPUT);  // LED2 - izlaz
  pinMode (SOLENOID, OUTPUT);  // SOLENOIDNI VENTIL - izlaz
  pinMode (PUMP, OUTPUT);  // PUMPA - izlaz


  digitalWrite(SOLENOID, RELAY_OFF); // SOLENOIDNI VENTIL OFF kod
incijalizacije
  digitalWrite(PUMP, RELAY_OFF);  // PUMPA OFF kod inicijalizacije
  digitalWrite(LED2, LOW);
  digitalWrite(LED1, LOW);

   if ((digitalRead(SENZOR_H)== LOW) && (digitalRead(SENZOR_L) ==
HIGH))//spremnik pun
 {
   prazni=true;
   puni=false;
   }
 if  ((digitalRead(SENZOR_L)== LOW)  &&  (digitalRead(SENZOR_H)  ==
HIGH))//spremnik prazan
 {
   puni=true;
   prazni=false;
   }
  if (((digitalRead(SENZOR_H)== HIGH) && (digitalRead(SENZOR_L) ==
HIGH))) //spremnik ok
 {
   prazni=true;
   puni=false;
   }

  Serial.begin(9600); //start serial port 9600bps
  lcd.begin(16,2);  //type of LCD display (16*2)
  lcd.backlight(); //ON LCD backlight

  // početni ispis na LCD - samo kod uključenja

  lcd.setCursor(2,0); // cursor position (CHAR, LINE)
  lcd.print("Sveuciliste");
  lcd.setCursor(5,1);
  lcd.print("Sjever");
  delay(2000); // pauza 2s
  lcd.clear(); //erase display
  lcd.setCursor(5,0);
  lcd.print("odjel");
  lcd.setCursor(1,1);
  lcd.print("ELEKTROTEHNIKE");
  delay(2000); // pauza 2s

  lcd.clear(); //erase display
  lcd.setCursor(1,0);
  lcd.print("Karlo Tretnjak");
  lcd.setCursor(3,1);
  lcd.print("0036449080");
  delay(2000); // pauza 2s

  lcd.clear(); //erase display
  lcd.setCursor(3,0);
  lcd.print("AUTOMATSKO");
```

```
lcd.setCursor(2,1);
lcd.print("UPRAVLJANJE");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(2,0);
lcd.print("Zavrsni rad");
lcd.setCursor(2,1);
lcd.print("2014 / 2015");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(3,0);
lcd.print("AUTOMATSKO");
lcd.setCursor(1,1);
lcd.print("NAVODNJAVANJE");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(5,0);
lcd.print("Mentor:");
lcd.setCursor(3,1);
lcd.print("M.HORVATIC");
delay(2000); // pauza 2s

lcd.clear(); //erase display
lcd.setCursor(2,0);
lcd.print("Software ver:");
lcd.setCursor(6,1);
lcd.print("1.1");
delay(2000); // pauza 2s

lcd.clear();
delay(1000);
lcd.setCursor(2,0);
lcd.print("INITIALIZING");
lcd.setCursor(3,1);
lcd.print("PLEASE WAIT");
delay(1000);

lcd.clear();
delay(1000);
lcd.setCursor(2,0);
lcd.print("INITIALIZING");
lcd.setCursor(3,1);
lcd.print("PLEASE WAIT");
delay(1000);

lcd.clear();
delay(1000);
lcd.setCursor(2,0);
lcd.print("INITIALIZING");
lcd.setCursor(3,1);
lcd.print("PLEASE WAIT");
delay(1000);
lcd.clear();

digitalWrite(LED1, HIGH);
```

```
  digitalWrite(BUZZER, LOW);
   lcd.setCursor( 0, 0 );
   lcd.print( "T:       H:");
}

/*--------------------------------------------------------------------
MAIN LOOP
--------------------------------------------------------------------*/

void loop()
{
   byte button;


   //Check to see if we need to enter Menu.  Only button that will do
this is the "Select" Button

button = ReadButtons();
switch( button )
   {
//*********************************************************************
                              TOP LEVEL MENU
*********************************************************************
      case BUTTON_SELECT:
      {
          menu = true;                                    //Make sure menu
                                                  while loop flag is set true
           menu_number=0;                                  //Default first
                                                         menu item to 0 case
        lcd.clear();
        delay(200);
         while(menu ==true)                                //Stay in menu as
                                                    long as menu flag is true
        {
          button = ReadButtons();
          switch( button)
          {
            case BUTTON_LEFT:
             {
               lcd.clear();
               menu_number = menu_number - 1;
               if(menu_number <0 ) menu_number = 3;
               delay(200);
               break;
              }
             case BUTTON_UP:
             {
               lcd.clear();
               menu_number = menu_number + 1;
               if(menu_number > 3) menu_number = 0;
               delay(200);
               break;
              }
             case BUTTON_SELECT:
             {
               lcd.clear();
               delay(200);
               switch(menu_number)
```

```
                {
//*********************************************************************
                                    SET TEMP
**********************************************************************
                  case MENU_SETTEMP:
                  {
                    set_temp = true;
                    lcd.clear();
                    while(set_temp == true)
                    {
                      lcd.setCursor(0,0);
                      lcd.print("Set TEMPERATURE");
                      lcd.setCursor(0,1);
                      lcd.print(set_temperature);
                      button = ReadButtons();
                      switch(button)
                      {
                        case BUTTON_UP:
                        {
                          set_temperature = set_temperature + 0.5;
                          lcd.setCursor(0,1);
                          lcd.print(set_temperature);
                          delay(200);
                          break;
                        }
                        case BUTTON_LEFT:
                        {
                          set_temperature = set_temperature - 0.5;
                          lcd.setCursor(0,1);
                          lcd.print(set_temperature);
                          delay(200);
                          break;
                        }
                        case BUTTON_SELECT:
                        {
                          lcd.clear();
                          set_temp = false;
                          delay(200);
                          break;
                        }
                        break;
                      }
                    }
                  break;
                  }
//*********************************************************************
                              SET TEMP SWING
**********************************************************************
                  case MENU_MOD_RADA:
                  {
                    mod_rada = true;

                    while(mod_rada == true)
                    {
                      lcd.setCursor(0,0);
                      lcd.print("Set MOD RADA");
                      lcd.setCursor(0,1);
                      if(mod==0)
```

```
{

 lcd.print("AUTO  ");
 auto_mod=true;
 manual_mod=false;
}
else
{

  lcd.print("MANUAL");
  auto_mod=false;
  manual_mod=true;
  }
button = ReadButtons();
switch(button)
{
  case BUTTON_LEFT:
  {
    mod = mod - 1;

    if(mod < 0) mod = 1;
    lcd.setCursor(0,1);
       if(mod==0)
      {

        lcd.print("AUTO  ");
        auto_mod=true;
        manual_mod=false;
        }
      else
     {

       lcd.print("MANUAL");
       auto_mod=false;
       manual_mod=true;
     }
    delay(200);
    break;
  }
  case BUTTON_UP:
  {
    mod = mod + 1;

    if(mod > 1) mod = 0;
    lcd.setCursor(0,1);
       if(mod==0)
       {

         lcd.print("AUTO  ");
         auto_mod=true;
         manual_mod=false;
       }
         else
       {

         lcd.print("MANUAL");
         auto_mod=false;
         manual_mod=true;
```

```
                    }
                    delay(200);
                    break;
                  }
                  case BUTTON_SELECT:
                  {
                    lcd.clear();
                    mod_rada = false;
                    delay(200);
                    break;
                  }
                  break;
                }
              }
            break;
            }

//*********************************************************************
                          SET MOISTURE
*********************************************************************
            case MENU_SETMOISTURE:
            {
              set_soil = true;
              lcd.clear();
              while(set_soil == true)
              {
                lcd.setCursor(0,0);
                lcd.print("Set MOISTURE");
                lcd.setCursor(0,1);
                lcd.print(set_moisture);
                button = ReadButtons();
                switch(button)
                {
                  case BUTTON_UP:
                  {
                    set_moisture = set_moisture + 0.5;
                    lcd.setCursor(0,1);
                    lcd.print(set_moisture);
                    delay(200);
                    break;
                  }
                  case BUTTON_LEFT:
                  {
                    set_moisture = set_moisture - 0.5;
                    lcd.setCursor(0,1);
                    lcd.print(set_moisture);
                    delay(200);
                    break;
                  }
                  case BUTTON_SELECT:
                  {
                    lcd.clear();
                    set_soil = false;
                    delay(200);
                    break;
                  }
                  break;
                }
```

```
                  }
                break;
                }
//*****************************************************************
                              EXIT MENU
 ****************************************************************
                case MENU_EXIT:
                {

                  menu=false;
                  lcd.clear();
                  delay(200);
                  break;
                }
                break;
              }

            }
          break;
          }
         lcd.setCursor(0,0);
         lcd.print("      MENU    ");
         lcd.setCursor(0,1);
         lcd.print(MENU[menu_number]);
        }

     }

     lcd.clear();
     temp = millis()/1000;

  }




/*-----------------------------------------------------------------
DS18B20
-----------------------------------------------------------------*/

  byte i;
  byte present = 0;
  byte type_s;
  byte data[12];
  byte addr[8];
  float celsius;

 if ( !ds.search(addr)) {
    ds.reset_search();
    return;
  }


  // the first ROM byte indicates which chip
 switch (addr[0]) {
     case 0x28:
      type_s = 0;
```

```
      break;
    default:
    return;
}

  ds.reset();
  ds.select(addr);
  ds.write(0x44,1);         // start conversion, with parasite power
on at the end

         delay(1000);     // potrebna pauza zbog konvertiranja (750ms
                                    potrebno, 1000ms zbog rezerve)

  present = ds.reset();
  ds.select(addr);
  ds.write(0xBE);         // Read Scratchpad

     for ( i = 0; i < 9; i++) {           // Byte0-Byte8 = potrebno 9
                                                              bajta
    data[i] = ds.read();
      }

  // convert the data to actual temperature

  unsigned int raw = (data[1] << 8) | data[0];
  if (type_s) {
    raw = raw << 3; // 9 bit resolution default
    if (data[7] == 0x10) {
      // count remain gives full 12 bit resolution
      raw = (raw & 0xFFF0) + 12 - data[6];
    }
  } else {
    byte cfg = (data[4] & 0x60);
    if (cfg == 0x00) raw = raw << 3;  // 9 bit resolution, 93.75 ms
    else if (cfg == 0x20) raw = raw << 2; // 10 bit res, 187.5 ms
    else if (cfg == 0x40) raw = raw << 1; // 11 bit res, 375 ms
    // default is 12 bit resolution, 750 ms conversion time
  }
    celsius = (float)raw / 16.0; //  ocitanu vrijednost podijeliti sa
                                            16, float zbog decimala

  int soil_s = analogRead(SOIL_SENSOR);        // čitanje vrijednosti
                                          vlaznosti sa senzora 1023-0
  float soil =( 1023-soil_s)  / 10.24;    // podijeliti s 10.24 da se
                                                    dobije postotak

   lcd.setCursor( 0, 0 );
   lcd.print( "T:       H:" );
   lcd.setCursor(2,0);
   lcd.print(celsius);
   lcd.setCursor(6,0);
   lcd.print("\337C");
   lcd.setCursor(11,0);
   lcd.print(soil);
   lcd.setCursor(15,0);
   lcd.print("%");
```

```
  //TEMP error

  if(data[6]==255) //data=0 FF FF FF FF FF FF FF FF FF senzor odpojen
                                                              4095.94°C
  {
    error_flag=true;
    lcd.clear();
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, HIGH);
    digitalWrite(BUZZER, HIGH);
    digitalWrite(SOLENOID, RELAY_OFF);
    digitalWrite(PUMP, RELAY_OFF);
    lcd.setCursor(4,0);
    lcd.print("ERROR !!");
    lcd.setCursor(0,1);
    lcd.print("                ");
    lcd.setCursor(2,1);
    lcd.print("TEMP  SENSOR");
    }
    else
    {
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, LOW);
    digitalWrite(BUZZER, LOW);
    error_flag=false;
    }


  /*--------------------------------------------------------------------
    MAIN PROGRAM
 --------------------------------------------------------------------*/

//MANUAL mod rada

 if(manual_mod == true && error_flag==false)

 {
   if (digitalRead(SENZOR_H) != LOW)
   {
   auto_mod=false;
   digitalWrite(SOLENOID, RELAY_ON);
   digitalWrite(PUMP, RELAY_ON);
   lcd.setCursor(0,0);
   lcd.print("                ");
   lcd.setCursor(3,0);
   lcd.print("MANUAL MOD");
   lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(0,1);
   lcd.print("PUMP & SVALVE ON");
   }
  else if (digitalRead(SENZOR_H) == LOW)
  {
   digitalWrite(PUMP, RELAY_OFF);
   lcd.setCursor(0,0);
   lcd.print("                ");
   lcd.setCursor(2,0);
   lcd.print("SPREMNIK PUN");
```

```
   lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(2,1);
   lcd.print("SOLENOID  ON");
   delay(4000); //pauza 4s zbog praznjenja spremnika
   }
 else
 {
   digitalWrite(SOLENOID, RELAY_OFF);
   digitalWrite(PUMP, RELAY_OFF);
  }
  }
 if (auto_mod==true && error_flag==false)
{

   lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(4,1);
   lcd.print("AUTO MOD");
    manual_mod=false;
  if (puni == true )
   {
   digitalWrite(PUMP, RELAY_ON);
   digitalWrite(SOLENOID, RELAY_OFF);

     if ((digitalRead(SENZOR_H)== LOW) && (digitalRead(SENZOR_L) ==
HIGH))  //spremnik pun
 {
   puni=false;
   prazni=true;
   lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(1,1);
   lcd.print("SPREMNIK PUN");
   }
   }
   if (prazni == true  )
   {
   {
     digitalWrite(PUMP, RELAY_OFF);
    }
     if (celsius >= set_temperature && soil <= set_moisture)
     {
     digitalWrite(SOLENOID, RELAY_ON);
    }
      if ((digitalRead(SENZOR_L)== LOW) && (digitalRead(SENZOR_H) ==
HIGH)) //spremnik prazan
 {
   puni=true;
   prazni=false;
    lcd.setCursor(0,1);
   lcd.print("                ");
   lcd.setCursor(0,1);
   lcd.print("SPREMNIK PRAZAN");
   }
     }

}
```

```
int   buttonVoltage1 = analogRead( BUTTON_ADC_PIN );
Serial.println(buttonVoltage1);
}

/*--------------------------------------------------------------------
ReadButtons()
--------------------------------------------------------------------*/

byte ReadButtons()
{
   float buttonVoltage;
   byte button = BUTTON_NONE;    // return no button pressed if the
below checks don't write to btn

   //read the button ADC pin voltage
   buttonVoltage = analogRead( BUTTON_ADC_PIN );

   //sense if the voltage falls within valid voltage windows
   if( buttonVoltage < ( RIGHT_10BIT_ADC + BUTTONHYSTERESIS ) )
   {
      button = BUTTON_RIGHT;
   }
   else if(   buttonVoltage >= ( UP_10BIT_ADC - BUTTONHYSTERESIS )
           && buttonVoltage <= ( UP_10BIT_ADC + BUTTONHYSTERESIS ) )
   {
      button = BUTTON_UP;
   }
   else if(   buttonVoltage >= ( DOWN_10BIT_ADC - BUTTONHYSTERESIS )
           && buttonVoltage <= ( DOWN_10BIT_ADC + BUTTONHYSTERESIS )
)
   {
      button = BUTTON_DOWN;
   }
   else if(   buttonVoltage >= ( LEFT_10BIT_ADC - BUTTONHYSTERESIS )
           && buttonVoltage <= ( LEFT_10BIT_ADC + BUTTONHYSTERESIS )
)
   {
      button = BUTTON_LEFT;
   }
   else if(   buttonVoltage >= ( SELECT_10BIT_ADC - BUTTONHYSTERESIS
)
           && buttonVoltage <= ( SELECT_10BIT_ADC + BUTTONHYSTERESIS
) )
   {
      button = BUTTON_SELECT;
   }
   //handle button flags for just pressed and just released events
   if( ( buttonWas == BUTTON_NONE ) && ( button != BUTTON_NONE ) )
   {
      //the button was just pressed, set buttonJustPressed, this can
optionally be used to trigger a once-off action for a button press
event
      //it's the duty of the receiver to clear these flags if it
wants to detect a new button change event
      buttonJustPressed  = true;
      buttonJustReleased = false;
   }
   if( ( buttonWas != BUTTON_NONE ) && ( button == BUTTON_NONE ) )
```

```
   {
      buttonJustPressed  = false;
      buttonJustReleased = true;
   }

   //save the latest button value, for change event detection next
time round
   buttonWas = button;

   return( button );
}
```

Sveučilište Sjever

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, **Karlo Tretnjak** pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor završnog rada pod naslovom **Realizacija sustava za automatsko zalijevanje biljke** te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Karlo Tretnjak

_____

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, **Karlo Tretnjak** neopozivo izjavljujem da sam suglasan s javnom objavom završnog rada pod naslovom **Realizacija sustava za automatsko zalijevanje biljke** čiji sam autor.

Karlo Tretnjak

_____