

ŠOLSKI CENTER VELENJE
ELEKTRO IN RAČUNALNIŠKA ŠOLA
Trg mladosti 3, 3320 Velenje
MLADI RAZISKOVALCI ZA RAZVOJ ŠALEŠKE DOLINE

RAZISKOVALNA NALOGA

OBRNJEN RADIATORSKI TERMOSTAT

Tematsko področje: Interdisciplinarno (elektronika, strojništvo)

Avtorji:

Martin Sinigajda

Željko Štukelja

Žiga Slemenšek

Mentor:

Tomaž Juvan

Velenje, 2021

Raziskovalna naloga je bila opravljena na Šolskem centru Velenje, Elektro in računalniški šoli.

Mentor: Tomaž Juvan

Datum predavitve: april 2021

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

ŠD Elektro in računalniška šola Velenje, šolsko leto 2020/2021

KG Interdisciplinarna

AV Martin Sinigajda, Željko Štukelj, Žiga Slemenšek

SA Tomaž Juvan

KZ 3320 Velenje, SLO, Trg mladosti 3

ZA Elektro in računalniška šola

LI 2021

IN Obrnjen radiatorski termostat

TD Raziskovalna naloga

OP VI, 29 str., 26 sl., 5 pril., 3 vir.

LJ SL

AL sl/en

AB V današnjem času se vedno bolj pogosteje v naših domovih pojavlja "smart" tehnologija kot so lučke, zaves, elektronske naprave... Ampak bolj redkeje najdemo pametne radiatorje oz. ventile za radiatorje. Večina smart ventilov za radiatorje je zelo opazna in deluje na principu binarnega delovanja. V naši nalogi smo si zadali cilj da te težave odpravimo in kontroliramo našo temperaturo iz udobja našega kavča. To bi pa dosegli z aplikacijo, termostatom in senzorjem, kateri bi bil vgrajen v ventil. Najprej bi v aplikacijo vnesli zaželeno temperaturo, katero bi termostat zaznaval, ventil se bi pa glede na to osredotočal in se počasi odpiral ali zapiral odvisno od temperature sobe ali hiše. Temperaturo bi lahko tudi kontrolirali če nas ni doma preko telefona in z tem varčevali z ogrevanjem. Namen te naloge je najti nov in lažji način s katerim bi lahko naredili našo vsakdanjo življenje bolj udobno ter bolj varčno.

KEY WORDS DOCUMENTATION

ND Elektro in računalniška šola Velenje, šolsko leto 2020/2021

CX Interdisciplinary

AA Martin Sinigajda, Željko Štukelj, Žiga Slemenšek

AU Tomaž Juvan

PP 3320 Velenje, SLO, Trg mladosti 3

PB Šolski center Velenje

PY 2021

TI Obrnjen radiatorski termostat

DT Research work

NO VI, 29 p., 26 fig., 5 pril., 3 vir.

LA SL

AL sl/en

AB In today's age we find ourselves using more and more of so called "smart" technology, like for example lights, curtain, electrical devices... But we don't often hear about smart radiator valves. Most smart valves are very flashy and work on the principle of binary operation. We set ourselves a goal that we get rid of these mistakes and control our temperature from the comfort of our own couch. We would achieve that by using an app, a thermostat and a sensor, which would be built-in. First of we would set our preferable temperature in the app, the app would then send this information to the thermostat. Then the valve would open or close depending on the results. The purpose of this task is to create a new and easier way of living, which could make our everyday life more comfortable and more economical.

Vsebina

1.	Uvod	1
2.	Termostatski ventil	2
3.	Izdelava delovnih komponent	3
3.1.	Zobnik	5
3.2.	Material	6
4.	Mehansko delovanje polža	7
5.	Programiranje	8
6.	Izdelava aplikacije	15
7.	Zgodovina krmiljenja preko aplikacije in spleta	16
8.	Program in izdelava vezja	17
8.1.	ESP-32S	17
8.2.	Programiranje in upravljanje motorja	17
8.3.	Temperaturni senzor	19
8.4.	Programiranje	20
9.	Analiza hipotez	21
9.1.	Venil bo neopazen	21
9.2.	Toploto bomo lahko kontrolirali preko telefona	21
9.3.	Ali ga lahko naredimo doma	21
10.	Razprava	22
11.	Zaključek	22
12.	Povzetek	23
13.	Zahvala	23
14.	Priloga	24
15.	Viri	29

Kazalo slik

Slika 1: Danfoss termostatski ventil	2
Slika 2: Delovanje na principu polža	3
Slika 3: Delovanje na principu dveh gredi	3
Slika 4: Delovanje na principu dveh navojev	4
Slika 5: Delovanje na principu dveh gredi in navoja	4
Slika 6: Izboljšano delovanje na principu dveh gredi in navoja	5
Slika 7: Elementi z zobniki	5
Slika 8: 3D-printer:	6
Slika 9: Izgled delovanja z polžastem gonilu	7
Slika 10: Skica izdelave	15
Slika 11: 2SESP32	16
Slika 12: Esp-32s	17
Slika 13: NPN in PNP tranzistorja	17
Slika 14: H-bridge za motor	18
Slika 15: L293D	18
Slika 16: DC motor driver	19
Slika 17: Ntk Senzor	19
Slika 18: Dht11 senzor	20
Slika 19: ARDUINO IDE	20
Slika 20: Vezava Arduinota	24
Slika 21: Nadzornovanje temperature	24
Slika 22: Pogled 1 ventila z polžastim gonilu	25
Slika 23: Pogled 2 ventila z polžastim gonilu	25
Slika 24: Program 1	26
Slika 25: Program 2	27
Slika 26: Program 3	28

1. Uvod

V današnjem času se vedno bolj pogosteje v naših domovih pojavlja "smart" tehnologija kot so lučke, zaves, elektronske naprave... Ampak bolj redkeje najdemo pametne radiatorje oz. ventile za radiatorje. Večina smart ventilov za radiatorje je zelo opazna in deluje na principu binarnega delovanja. V naši nalogi smo si zadali cilj da te težave odpravimo in kontroliramo našo temperaturo iz udobja našega kavča. To bi pa dosegli z aplikacijo, termostatom in senzorjem, kateri bi bil vgrajen v ventil. Najprej bi v aplikacijo vnesli zaželeno temperaturo, katero bi termostat zaznaval, ventil se bi pa glede na to osredotočal in se počasi odpiral ali zapiral odvisno od temperature sobe ali hiše. Temperaturo bi lahko tudi kontrolirali če nas ni doma preko telefona in z tem varčevali z ogrevanjem. Namen te naloge je najti nov in lažji način s katerim bi lahko naredili našo vsakdanjo življenje bolj udobno ter bolj varčno.

HIPOTEZE:

1. Ventil bo neopazen
2. Toploto bomo lahko kontrolirali preko telefona
3. Ali ga lahko naredimo doma

2. Termostatski ventil

Leta 1944 je Danski izumitelj Mads Clausen izdelal prvi termostatski ventil. Najprej ga je začela proizvajati podjetje Danfoss, katerim je termostatski ventil takoj prinesel veliko uspeha. Veliko podjetji je začelo tudi proizvajati svoje termostatske ventile in vsak je imel svojo posebno lastnost. Te naprave so poceni, natančne in zanesljive. V termostatski ventil je vgrajeno posebno tipalo, ki se na temperaturne spremembe v prostoru odziva tako, da odpre oziroma zapre dotok vroče vode v radiator. Tipalo je polnjeno s plinom, tekočino ali voskom, ki se jim ob spremembi temperature spremeni prostornina. Prek mehanizma, ki sestavlja tipalo, se sprememba prenese na ventil, tako da se ob povišanju temperature v prostoru zapre, ko temperatura pade, pa odpre. Višino temperature nastavimo s termostatsko glavo, ki primerja temperaturo, zaznano s tipalom, in nastavljeno vrednost, glede na to pa se potem odpira ali zapira ventil. Na termostatski glavi so oznake, temperature, ki jim ustrezajo, pa so navedene v navodilih proizvajalca. Obrnjen ventil bi uporabili raje uporabljali zaradi lepšega videza in nebi prišlo do butanja ob njega. Obrnjen ventil bi pa bil enostavno neviden do posameznika.



Slika 1: Danfoss termostatski ventil

3. Izdelava delovnih komponent

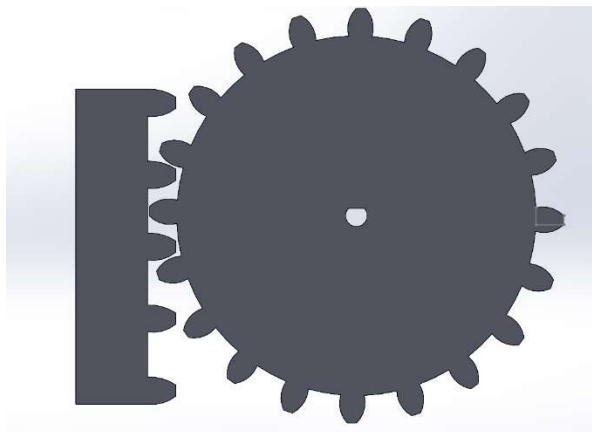
Ves izgled in mehanično delovanje je bilo izdelano v SolidWorks, saj je bil program z katerim se bil najbolj seznanjen. Solidworks je program za računalniško podprto konstruiranje in inženirske analize.

Pri izdelavi izgleda in delovanja smo začeli z delovanjem na principu polža. Na principu, ki bi izgledalo kot tehnica, bi motor poganjal polža, kateri bi uravnaval palico glede na željeno temperaturo in odpiral/zapiral ventil. Problem pri temu je prišlo pri samem izgledu in delovanju, saj je izgledalo površno in delovanje je bilo nezanesljivo, ker nebi polž proizvedel dovolj visoko silo da bi potisnil ventil in ga zadržal.



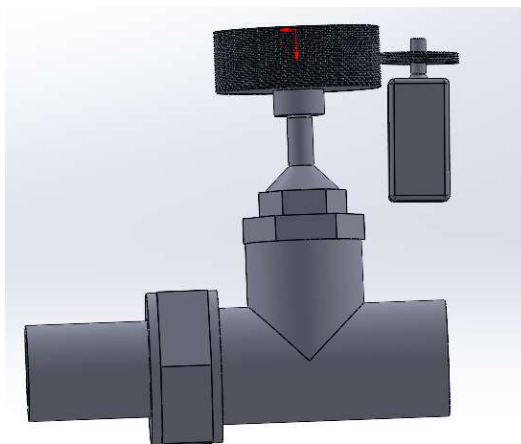
Slika 2: Delovanje na principu polža

Drugi prototip je bil zelo drugačen in manjši vse kar bi zahteval je 2 dela, 1. del je bil krog z zobniki, katerega bi uravnaval motor in potiskal 2. del, da bi odpiral in zapiral ventil. Problem je nastal podobno kot prit prvem sila, ki bi držala 2. del na mestu in da nebi slučajno odpadel, ker ga nebi mogli pritrditi.



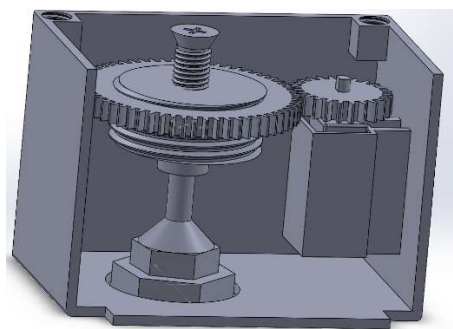
Slika 3: Delovanje na principu dveh gredi

Tretji prototip je bil narejen iz dveh elementov (oba z navojju). Delovala bi tako da bi 1 del ki bi bil na motorju poganjal drugi del ki bi potem potiskal ventil in ga s tem odpiral/zapiral. Napaka se je pokazala pri pritrditvi.



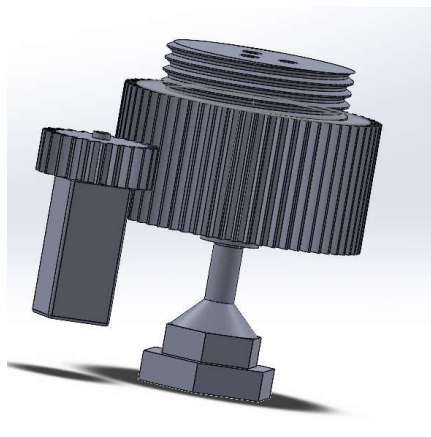
Slika 4: Delovanje na principu dveh navojev

Naslednji prototip je bil narejen iz več elementov ta je uporabljal dva zobnika in navoj. 1. zobnik bi bil na motorju in poganjal 2. zobnik, ta bi pa potiskal navoj ki bi pritiskal na ventil in ga zapiral/odpiral. Problem je prišel pri delovanju, saj se bi drugi zobnik vrtel po navoju namesto da bi on premikal navoj.



Slika 5: Delovanje na principu dveh gredi in navoja

Zadnji prototip je nastal iz prejšnjega saj je podoben z 2 zobnikoma in navojem. Motor potiska mali zobnik ki potem premika velik zobnik ta je navit na navoju, kateri je pritrjen na škatlo. Velik zobnik se s tem premika in pritiska na ventil in ga s tem odpira/zapira.



Slika 6: Izboljšano delovanje na principu dveh gredi in navoja

Začasno pa še eksperimentiramo, saj je bilo veliko ovir z COVID-19, zato še najbolj primerni model še iščemo in preizkušamo.

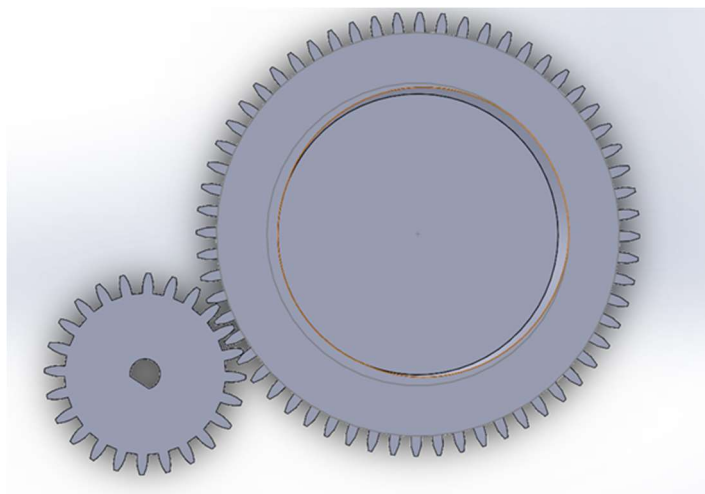
3.1. Zobnik

Glede zobnikov pri končni izdelavi (v primeru koliko zobnikov potrebujemo) sem uporabil enačbo za prestavno razmerje:

$$i = \frac{d_2}{d_1} = \frac{z_2}{z_1}$$

d- Premer elementa

z- Število zob zobnika



Slika 7: Elementi z zobniki

3.2. Material

Z vizijo je prišlo do vprašanja z katerim materialom bi to naredili najprej smo imeli idejo da naredimo z 3D printerjem, ampak plastika nebi bila tako zanesljiva. To smo pa zamenjala z idejo da bi naredili nekatere elemente na CNC stroju. Zato smo morali narediti iz kakšnega tršega material. Odločili smo se za kovino, ki bi bila bolj vzdržljiva in zanesljiva, ampak zaradi trenutnih okoliščin jih ni bilo možno narediti. Zato smo se začasno odločili narediti preko 3D printerja. Ti elementi, kateri niso preveč zanesljivi so cenejši in lažje za naredit. Takšna izdelava ima veliko prednosti in slabosti. Prednosti so da je lažja in cenejša izdelava, izdelava pa ima tudi prednost potrditve naše hipoteza za izdelavo doma saj vse kar bi potrebovali je 3D printer. Slabosti takšne izdelave pa je da so elementi krhki in nezanesljivi pri tehnični izvedbi zaradi plastike.



Slika 8: 3D-printer:

4. Mehansko delovanje polža

Delovanje polža deluje na principu, ko ga nekaj poganja da se vrti in s tem se regulira od 0% do 100%. Odvisno od katerim kotom je polž. To delovanje smo uporabili tako da odvisno od kota polža bo palica (katera je pritrjena na sredino) odpirala ali zapirala ventil odvisno od kota samega polža. Sama palica, katero regulira polž, je pa izdelana iz kovine saj s tem deluje palica z dovolj močno silo na ventil da ga odpre ali zapre in ostane na mestu.



Slika 9: Izgled delovanja z polžastem gonilu

5. Programiranje

//tukaj uvozimo knjižnice ki jih potrebujemo za wifiserver, za povezavo na wifi in za temperaturni senzor

#include <WiFi.h>

#include <WiFiClient.h>

#include <WiFiServer.h>

#include <WebServer.h>

#include "DHT.h"

#define motorPin1 26 // tukaj definiramo digitalni izhod na arduinotu

#define motorPin2 18 // tukaj definiramo digitalni izhod na arduinotu

#define tempSensor 23 // tukaj definiramo digitalni izhod na arduinotu

#define DHTYPE DHT11

#define ssid "wifi ime" //wifi ime

#define password "wifi geslo" //wifi geslo

WebServer server(80); // tukaj naredimo spremenljivko "server" in nastavimo server port

String header; // tukaj nastavimo spremenljivko "header"

int targetTemp = 20; // tukaj nastavimo spremenljivko "targetTemp" s katero sledimo nastavljeni temperaturi

float trenutnaTemp = 20; // tukaj nastavimo spremenljivko "targetTemp" s katero sledimo trenutno temperaturo v prostoru

bool zaprtRadiator = true; // tukaj nastavimo spremenljivko "targetTemp" s katero sledimo ali radiatorski ventil zaprt ali odprt

//tukaj lahko nastavimo statični ip, ampak tega tukaj ne uporabljamo

IPAddress local_IP(192, 168, 1, 212);

// Set your Gateway IP address

```
IPAddress gateway(192, 168, 1, 1);
```

```
IPAddress subnet(255, 255, 0, 0);
```

```
DHT dht(tempSensor,DHTYPE);// tukaj nastavimo spremenljivko "dht", s katero lahko preberemo  
temperaturo
```

```
void setup() {
```

```
    // put your setup code here, to run once:
```

```
    pinMode(motorPin1, OUTPUT); // tukaj nastavimo ali je pin izhod ali vhod
```

```
    pinMode(motorPin2, OUTPUT); // tukaj nastavimo ali je pin izhod ali vhod
```

```
    pinMode(tempSensor, INPUT); // tukaj nastavimo ali je pin izhod ali vhod
```

```
    Serial.begin(9600);
```

```
    WiFi.begin(ssid,password); // tukaj esp-ju vpišemo ime in geslo wifi-ja
```

```
    //tukaj se z esp-jom povežemo na wifi
```

```
    while(WiFi.status() != WL_CONNECTED){
```

```
        delay(500);
```

```
        Serial.println("Connecting to WiFi..");
```

```
    }
```

```
    //tukaj v seriski vmesnik izpišemo ip esp-ja na wifi-ju
```

```
    Serial.print("Connected to ");
```

```
    Serial.print(ssid);
```

```
    Serial.print("\n with this ip: ");
```

```
    Serial.print(WiFi.localIP());
```

```
server.on("/",[](){ // tukaj poslušamo na zahtevo na tem zavihku
```

```
String output = ""; // tukaj nastavimo spremenljivko "output"
```

```
output += "<!DOCTYPE html>";
```

```
output += "<html>";
```

```
output += "  <head>";
```

```
output += "    <style>";
```

```
output += "  ";
```

```
output += "    html, body {";
```

```
output += "      height: 100%;";
```

```
output += "    }";
```

```
output += "";
```

```
output += "    body {";
```

```
output += "      ";
```

```
output += "      display:flex;";
```

```
output += "      align-items:center;";
```

```
output += "      justify-content: center;";
```

```
output += "      flex-direction: column;";
```

```
output += "      background-color: black;";
```

```
output += "    }";
```

```
output += "";
```

```
output += "    button {";
```

```
output += "      display:flex;";
```

```
output += "      align-items:center;";
```

```
output += "      justify-content: center;";
```

```
output += "      border-radius: 20px;";
```

```
output += "      border: none;";
```

```
output += "      height: 20px;";
```

```
output += "      margin-top :10px;";
```



```
output += "        width: 300px;";

output += "        height: 50px;";

output += "        font-size: xx-large;";

output += "        color: white;";

output += "";

output += "    }";

output += "    .gumbp {";

output += "        background-color: #9D0D0D;";

output += "    }";

output += "    .gumbm {";

output += "        background-color: #1C7E9F;";

output += "    }";

output += "    .maind {";

output += "        display: flex;";

output += "        align-items: center;";

output += "        justify-content: center;";

output += "        flex-direction: column;";

output += "        height: 70%;";

output += "        width: 70%;";

output += "        background-color: rgb(31, 31, 31);";

output += "        border-radius: 20px;";

output += "        color: white;";

output += "        font-family: Arial, Helvetica, sans-serif;";

output += "        font-size: xx-large;";

output += "    }";

output += "    </style>";

output += "    </head>";

output += "<body>";

output += "";
```

```
output += "  <div class='maind'>";

output += "    <p>Trenutna temperatura: "+(String) trenutnaTemp+"&deg;C</p>";

output += "    <p>Nastavljena temperatura: "+(String) targetTemp+"&deg;C</p>";

output += "    <button class='gumpb' onclick='up()'></button>";

output += "  ";

output += "    <button class='gumbm' onclick='down()'></button>";

output += "";

output += "  <script>";

output += "    async function up(){";

output += "      await fetch('/temp/up');";

output += "      location.reload();";

output += "    }";

output += "    async function down(){";

output += "      await fetch('/temp/down');";

output += "      location.reload();";

output += "    }";

output += "  </script>";

output += "  </div>";

output += "</body>";

output += "</html>";
```

```
server.send(200, "text/html", output); // tukaj pošlemo zahtevek po HTML-ju

});
```

```
server.on("/temp/up",[]){ // tukaj poslušamo na zahtevo na tem zavihku

  targetTemp += 1; // tukaj spremenljivko "targetTemp" spremenimo za 1

  server.send(200, "text/plain", "OK");// tukaj se odzovemo na zahtevek
```

```
});
```

```
server.on("/temp/down",[]){ // tukaj poslušamo na zahtevo na tem zavihku
```

```
    targetTemp -= 1; // tukaj spremenljivko "targetTemp" spremenimo za -1
```

```
    server.send(200, "text/plain", "OK");// tukaj se odzovemo na zahtevek
```

```
});
```

```
server.begin(); // tukaj zaženemo strežnik
```

```
dht.begin(); // tukaj zaženemo senzor
```

```
}
```

```
void loop() {
```

```
    // put your main code here, to run repeatedly:
```

```
    server.handleClient(); // tukaj poslušamo na zahteve na strežnik
```

```
    nastaviRadiator(); // v tej funkciji vrtimo motor glede na temperaturo
```

```
    trenutnaTemp = Temp(); // iz funkcije dobimo trenutno temperaturo in jo zapišemo v spremenljivko
```

```
}
```

```
void nastaviRadiator(){ // tukaj definiramo funkcijo "nastaviRadiator"
```

```
    if((trenutnaTemp + 0.5) <= targetTemp && zaprtRadiator){ // tukaj preverimo ali je trenutna temperatura  
manjša + 0.5 od nastavljene temperature in če je radiator zaprt
```

```
        obrniMotor(false); // tukaj obrnemo motor v smeri urinega kazalca
```

```
        delay(1000); // 1s = 1000, tukaj počakamo 1s
```

```
        ustaviMotor(); // tukaj ustavimo motor
```

```
        zaprtRadiator = false; // tukaj spremenljivko "zaprtRadiator" na false
```

```
    }else if(trenutnaTemp >= targetTemp && !zaprtRadiator){ // tukaj preverimo če je trenutna temperatura  
dosegla nastavljeno temperaturo in če je radiator odprt
```

```
        obrniMotor(true); // tukaj obrnemo motor v nasprotni smeri urinega kazalca
```

```
        delay(1000); // 1s = 1000, tukaj počakamo 1s
```

```
        ustaviMotor(); // tukaj ustavimo motor
```

```
    zaprtRadiator = true;// tukaj spremenljivko "zaprtRadiator" na true
}
}

void obrniMotor(bool nasprotiUrinegaKazalca){// definiramo funkcijo "obrnMotor"

//s tem preverimo kam se mora motor obrniti in ga obrnemo v tisto smer

if(nasprotiUrinegaKazalca){

    digitalWrite(motorPin2,LOW); // s to funkcijo nastavimo izklopimo napetost na pinu

    digitalWrite(motorPin1,HIGH); // s to funkcijo nastavimo vklopimo napetost na največje na pinu

}else{

    digitalWrite(motorPin1,LOW); // s to funkcijo nastavimo izklopimo napetost na pinu

    digitalWrite(motorPin2,HIGH); // s to funkcijo nastavimo vklopimo napetost na največje na pinu

}

}

void ustaviMotor(){

    digitalWrite(motorPin1,LOW); // s to funkcijo nastavimo izklopimo napetost na pinu

    digitalWrite(motorPin2,LOW); // s to funkcijo nastavimo izklopimo napetost na pinu

}

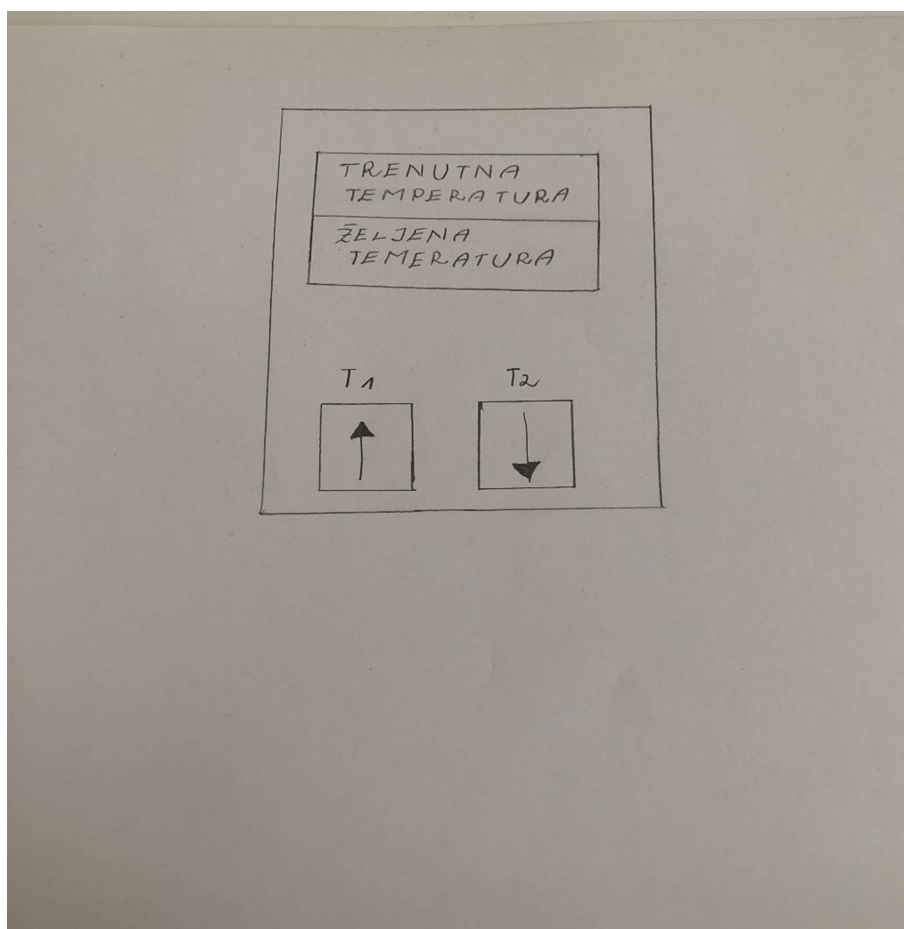
float Temp(){// tukaj definiramo funkcijo "Temp"

    float temp = dht.readTemperature(false); // tukaj prebermo temperaturo iz senzorja v °C in
    nastavimo spremenljivko "temp"

    return temp; // tukaj vrnemo spremenljivko "temp"
```

6. Izdelava aplikacije

Najprej smo se lotili izdelave vezja, saj smo lahko samo tako preverili ali gremo v pravo smer z izdelavo serverja. Ko smo izdelali vezje smo najprej naredil server za vklop in izklop led diode. Nato smo še priključili motor na vezje in še njega upravljali tako, da se je vrtel. Vendar naša naloga je bila, da naredimo server, ki meri temperaturo, na njemu pa se mora prikazati željena vrednost temperature, trenutno vrednost temperature in dve tipki s katerima lahko višamo in nižamo temperaturo. Ko smo višali temperaturo smo morali narediti v programu tako, da se je motor vrtel v desno ali levo smer odločili smo se, ko bo temperatura naraščala oz. ko bomo z tipko1 višali temperaturo se bo motor vrtel v desno, ko bo temperatura začela padati oz. bomo jo z tipko2 znižali se bo motor vrtel v levo smer.



Slika 10: Skica izdelave

7. Zgodovina krmiljenja preko aplikacije in spleta

Včasih smo krmilili samo s stikali in nismo imeli pametnih računalnikov dan danes pa že obstajajo sistemi za pametni nadzor doma, ki so sestavljeni iz raznih senzorjev računalnikov in naprav, ki so povezane preko nekega omrežja v centralnih sistemih in s temi sistemi lahko upravljamo z več napravami hkrati. Dan danes lahko te naprave upravljamo lokalno in tudi oddaljeno preko medmrežja. Ti sistemi za pametno krmiljenje so zelo funkcionalni ampak so cenovno nedostopni za veliko ljudi. Zato smo se mi odločili razviti enostavni pametni sistem, ki bo dostopnejši in bo imel cenovno dostopnejše elemente. Wi-fi server smo uporabili, ker smo ga lahko sami naredili preko arduina ESP32.

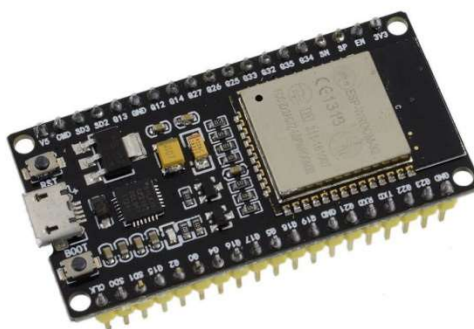


Slika 11: 2SESP32

8. Program in izdelava vezja

8.1. ESP-32S

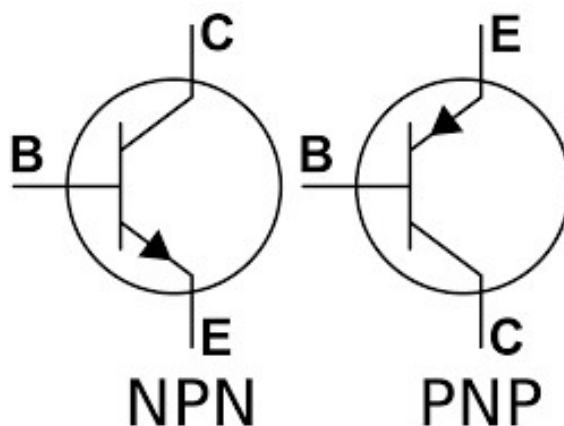
V naši raziskovalni nalogi smo uporabili esp-32s, ker je eden manjših arduinotov in ima možnosti brezžične povezave (preko wifija in bluetootha). Je 32-bitni mikrokrmilnik in ima odlične možnosti programiranja. Ta arduino je bil zelo primeren za našo nalogo zaradi wifi-serverja, ki smo ga naredili in s tem nismo imeli težav



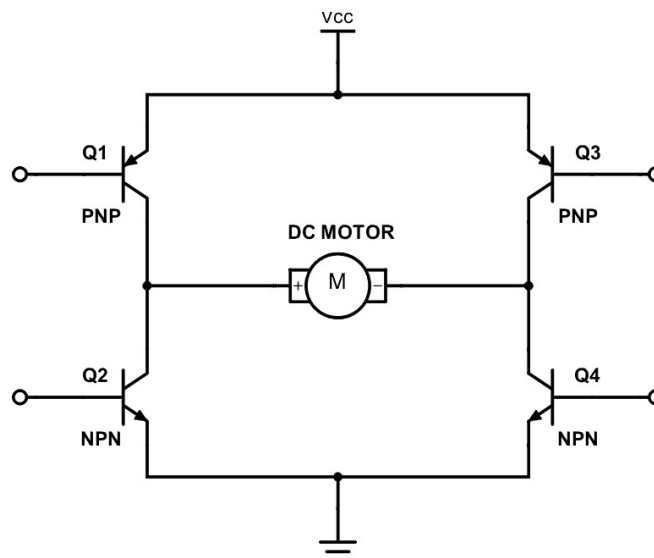
Slika 12: Esp-32s

8.2. Programiranje in upravljanje motorja.

Težave smo imeli pri programiranju in upravljanju motorja in pri merjenju temperature. Pri programiranju motorja smo imeli 3 možnosti upravljanja. Prva možnost je bila z tranzistorji PNP in NPN, ampak se zato nismo odločili, ker smo rabili bolj enostavnejšo različico pri vezavi, ker nismo imeli neomejeno prostora za vodnike.

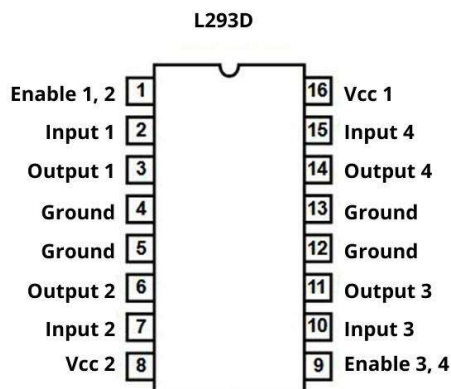


Slika 13: NPN in PNP tranzistorja

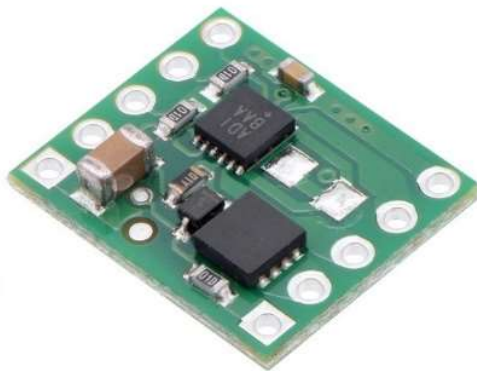


Slika 14: H-bridge za motor

Ostale sta nam še 2 enostavnejši možnosti in to sta bili DC motor driver in pa čip L293D in odločili smo se za L293D, ker je manjši od motor driverja in preprost za programirati po nekaj poskusih.



Slika 15: L293D



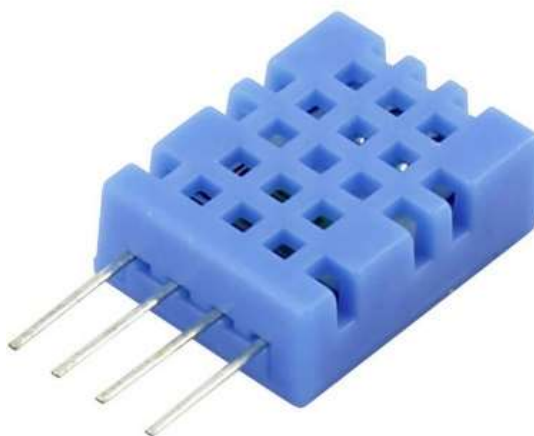
Slika 16: DC motor driver

8.3. Temperaturni senzor

Za temperaturni senzor smo od samega začetka uporabljali temperaturni senzor NTK, ki nam je delal precej težav zaradi samega računanja in enačb, ki nam bi naj dale na koncu vrednost temperature v stopinj celzijah. Te enačbe so nam delale precej preglavic in smo veliko časa porabili za reševanje teh problemov in raziskovanja, kako sestaviti enačbo, da nam da pravilno vrednost trenutne temperature. Ker pa nas je začel preganjat čas smo namesto NTK senzorja uporabili senzor DHT11, ki isto meri temperaturo, ampak vrednost da že v stopinj celzijah in nismo rabili sestavljati enačb.



Slika 17: Ntk Senzor



Slika 18: Dht11 senzor

8.4. Programiranje

Ta senzor nam je bil bolj všeč, ker je naprednejši in pametnejši od NTK-ja in ima že tudi knjižnice na arduinotu in je s tem še bolj preprosto programiranje. Pri programiranju programa smo si veliko pomagali z internetom in pregledali mnogo spletnih strani, da nam je bilo lažje in smo se naučili nekaj novega.



Slika 19: ARDUINO IDE

9. Analiza hipotez

9.1. Venil bo neopazen

Pri prvi hipotezi smo se najbolj zavzeli, saj je to eden izmed največjih delov naloge. Ventil smo naredili tako da bo celotni del ventila izgledal kot del radiatorja. To smo pa naredili v solidworks, tako da smo na nad in pod ventilu naredili majhno štalico v kateri bi bili vgrajeni vsi elementi in nebi bili tako vidni in moteči. Če upoštevamo vse to kar smo našli je naša hipoteza bila potrjena.

9.2. Toploto bomo lahko kontrolirali preko telefona

Druga hipoteza je zahtevala narediti spletno stran povezano z ventilom. To smo naredili tako da smo napisali arduino program katerega smo lahko kontrolirali preko spletne strani. Potem je pa prišlo da nekaj težav z IP naslovom in samim programom, ampak smo z času vse te napake opravili. Zato smo drugo hipotezo tudi potrdili.

9.3. Ali ga lahko naredimo doma

Z tretjo hipotezo je pa prišlo do nekako vmesnega odgovora, saj čeprav bi zlahka naredili ventil če imamo 3D-printer. Ampak večina domov nima takšnih naprav. Za program in vezavo je pa bi lahko vse zlahka naredili doma. Zadnjo hipotezo pa lahko delno potrdimo.

10. Razprava

Veliko ovir, ki so nam bili na poti smo poskušali popraviti, ampak nekatere probleme zaradi omejenega časa in trenutne situacije z COVID-19 nismo uspeli rešiti. Največji problem se je pokazal pri mehanskem delovanju, saj je bilo veliko prototipov, ampak nismo vedeli kateri izmed njih bi bil najboljši. Naš prvi uspeh je bil ko smo programirali motor tako da se je odzival na določeno temperaturo. Je pa tudi zelo veliko stvari, ki bi jih lahko nadgradili. Prva izboljšava je boljši material pri mehanizmu. Pri našem izdelku smo najprej si želeli narediti komponente iz kovine, ampak se zaradi omejenega časa tega nismo mogli doseči. Pri samem delovanju, če bi bil narejen iz kovine bi bil bolj trden in bolj močen pri delovanju. Druga izboljšava bi bila pri programu, saj želimo izdelati program, katerega bi lahko imeli kot aplikacija na telefonu. Sam izgled današnjega programa pa ni nekaj jasen. Poleg tega nima dobrega izgleda.

11. Zaključek

Raziskovalno nalogo, ki smo jo naredili povezuje veliko različnih območji. Pri izdelavi smo si pa vsak razdelili svoje območje dela v strojništvu, elektrotehniki in računalništvu. Strojništvo je zavzemalo vso 3D modulacijo (SolidWorks), izdelavo elementov v 3D printerju in sestavo elementov. Delo za elektrotehnično delovanje se je ukvarjalo z samo vezavo elementov, delovanje arduinota, motorja in senzorja ter povezavo med vsemi elementi. Računalniški je pa deloval na programu in delovanju spletne strani. Rezultat vseh naših izdelkov je bil da smo zgradili dober končni izdelek. Kljub končani raziskavi za nas to ni konec, saj imamo še veliko v planu za izboljšati.

12. Povzetek

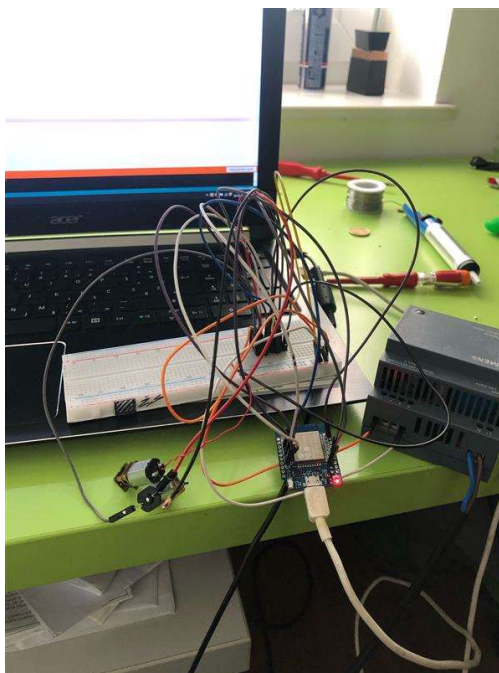
Naša raziskava obrnjenega termostatskega ventile bi bil zelo uporaben v našem vsakdanjem življenju. S tem bi olajšali našo vsakdanjo življenje in nas popeljalo en korak bližje popolnoma "smart house" sistemu. Pametni sistemi se počasi vedno bolj pojavljajo v naših domovih. Za današnje ventile potrebujemo vsakega posebej nastavit če hočemo željeno temperaturo. Poleg tega se termostatski ventili večkrat pokvarijo in jih je težko nastavit. Pri termostatskih ventilih večkrat pride tudi do okvare, katere ne vidimo ali ugotovimo. V našem izdelku pa ni problema z tem saj lahko temperaturo vseh sob spreminjamo na telefonu brez posamičnega nastavljanja. Poleg tega je neviden in se ne moremo v njega udarit. Naš izdelek bi bilo mogoče uporabiti v vsakem domu saj je lahek za izdelavo in za montirat. Poleg tega ima enostavno uporabo, katero lahko reguliramo kjerkoli smo.

13. Zahvala

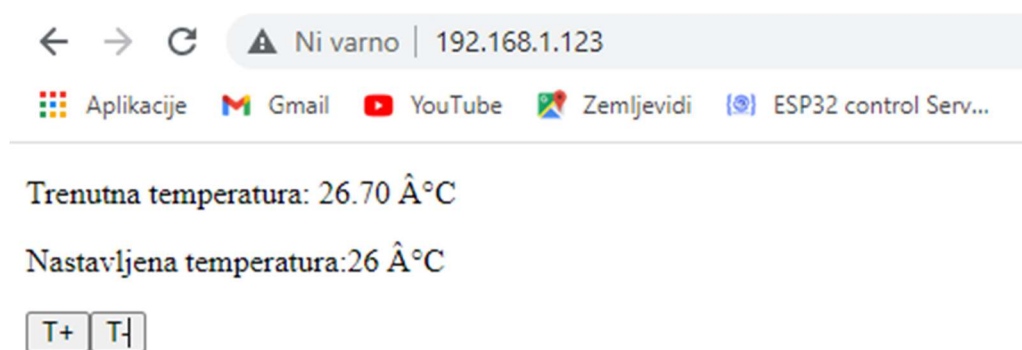
Za pomoč pri izdelavi raziskovalne naloge se zahvaljujemo:

- Mentorju Tomažu Juvanu, ki nam je pomagal pri izvedbi naših idej in podporo.
- Priateljem, ki so nam pomagali z idejami in popravki.
- Šolski Center Velenje- Elektro in računalniški šoli, kjer so nam nudili izvrstne pogoje za izdelavo raziskovalne naloge

14. Priloga



Slika 20: Vezava Arduinota



Slika 21: Nadzornovanje temperature



Slika 22: Pogled 1 ventila z polžastim gonilu



Slika 23: Pogled 2 ventila z polžastim gonilu

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <WiFiServer.h>
#include <WebServer.h>
#include "DHT.h"

#define motorPin1 26
#define motorPin2 18
#define tempSensor 23
#define DHTYPE DHT11

#define ssid "dd-wrt" //wifi ime
#define password "burekburek" //wifi geslo

WebServer server(80);

String header;

int targetTemp = 20;
float trenutnaTemp = 20;
bool zaprtRadiator = true;

IPAddress local_IP(192, 168, 1, 212);
// Set your Gateway IP address
IPAddress gateway(192, 168, 1, 1);

IPAddress subnet(255, 255, 0, 0);
DHT dht(tempSensor,DHTYPE);

void setup() {
    // put your setup code here, to run once:
    pinMode(motorPin1, OUTPUT);
    pinMode(motorPin2, OUTPUT);

    Serial.begin(9600);

    WiFi.begin(ssid,password,1);

    while(WiFi.status() != WL_CONNECTED){
        delay(500);
```

Slika 24: Program 1


```
#include <WiFi.h>
#include <WiFiClient.h>
#include <WiFiServer.h>
#include <WebServer.h>
#include "DHT.h"

#define motorPin1 26
#define motorPin2 18
#define tempSensor 23
#define DHTYPE DHT11

#define ssid "dd-wrt" //wifi ime
#define password "burekburek" //wifi geslo

WebServer server(80);

String header;

int targetTemp = 20;
float trenutnaTemp = 20;
bool zaprtRadiator = true;

IPAddress local_IP(192, 168, 1, 212);
// Set your Gateway IP address
IPAddress gateway(192, 168, 1, 1);

IPAddress subnet(255, 255, 0, 0);
DHT dht(tempSensor, DHTYPE);

void setup() {
  // put your setup code here, to run once:
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);

  Serial.begin(9600);

  WiFi.begin(ssid,password,1);

  while(WiFi.status() != WL_CONNECTED){
    delay(500);
    Serial.println("Connecting to WiFi..");
  }

  Serial.print("Connected to ");
  Serial.print(ssid);
  Serial.print("\n with this ip: ");
  Serial.print(WiFi.localIP());

  server.on("/",[]() {
    String output = "";
    output += "<!DOCTYPE html>";
    output += "<html>";
    output += "<body>";
    output += "<p>Trenutna temperatura: "+String(trenutnaTemp)+" °C</p>";
    output += "<p>Nastavljena temperatura: "+String(targetTemp)+" °C</p>";
    output += "<button onclick='up()'>T+</button>";
    output += "<button onclick='down()'>T-</button>";
    output += "<script>async function up(){await fetch('/temp/up')\nlocation.reload()}</script>";
    output += "<script>async function down(){await fetch('/temp/down')\nlocation.reload()}</script>";
    output += "</body>";
    output += "</html>";

    server.send(200, "text/html", output);
  });
```

Slika 25: Program 2

```
server.on("/temp/up",[]){
    targetTemp += 1;
    server.send(200, "text/plain", "OK");
};

server.on("/temp/down",[]){
    targetTemp -= 1;
    server.send(200, "text/plain", "OK");
};

server.begin();
dht.begin();
}

void loop() {
    // put your main code here, to run repeatedly:

    server.handleClient();
    nastaviRadiator();
    trenutnaTemp = Temp();

}

void nastaviRadiator(){
    if((trenutnaTemp + 0.5) <= targetTemp && zaprtRadiator){
        obrniMotor(false);
        delay(1000); // 1s = 1000
        ustaviMotor();
        zaprtRadiator = false;
    }else if(trenutnaTemp >= targetTemp && !zaprtRadiator){
        obrniMotor(true);
        delay(1000); // 1s = 1000
        ustaviMotor();
        zaprtRadiator = true;
    }
}

void obrniMotor(bool nasprotiUrinegaKazalca){
    if(nasprotiUrinegaKazalca){
        digitalWrite(motorPin2,LOW);
        digitalWrite(motorPin1,HIGH);
    }else{
        digitalWrite(motorPin1,LOW);
        digitalWrite(motorPin2,HIGH);
    }
}

void ustaviMotor(){
    digitalWrite(motorPin1,LOW);
    digitalWrite(motorPin2,LOW);
}

float Temp(){

    float temp = dht.readTemperature(false);
    Serial.println(temp);
    return temp;

}
```

Slika 26: Program 3

15. Viri

<http://plumbingandhvac.ca/thermostatic-radiator-valves/>

<https://gradbenistvo.finance.si/136258/Termostatski-ventili-Ne-mecimo-polen-skozi-okno>

<https://www.onlyradiators.co.uk/blog/buying-guides/thermostatic-valve-guide>