Izrada meteorološke stanice primjenom PIC32 ugradbenog sustava sa zaslonom

Dokša, Robert

Sveučilište
Sjever

Završni rad br. 014/MEH/2023

Izrada meteorološke stanice primjenom PIC32 ugradbenog sustava sa zaslonom

Robert Dokša, 0336048902

Varaždin, srpanj 2023. godine

Sveučilište Sjever

Odjel mehatronike

Završni rad br. 014/MEH/2023

Izrada meteorološke stanice primjenom PIC32 ugradbenog sustava sa zaslonom

Student

Robert Dokša, 0336048902

Mentor

Josip Srpak, viši predavač

Varaždin, srpanj 2023. godine

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

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL Odjel za mehatroniku

STUDIJ preddiplomski stručni studij Mehatronika

PRISTUPNIK Robert Dokša | MATIČNI BROJ 0336048902

DATUM 14.06.2023 | KOLEGIJ Elektronički elementi i sklopovi

NASLOV RADA Izrada meteorološke stanice primjenom PIC32 ugradbenog sustava sa zaslonom

NASLOV RADA NA ENGL. JEZIKU Creation of a weather station by using a PIC32 embedded system with a display

MENTOR Josip Srpak | ZVANJE viši predavač

ČLANOVI POVJERENSTVA

1. mr.sc. Ivan Šumiga, viši predavač
2. Josip Srpak, viši predavač
3. Doc.dr.sc. Dunja Srpak
4. Miroslav Horvatić, viši predavač
5.

Zadatak završnog rada

BROJ 014/MEH/2023

OPIS

U ovom završnom radu potrebno je izraditi vlastito rješenje meteorološke stanice primjenom ugradbenog sustava temeljenog na PIC32 mikrokontroleru.
Opisati principe rada mjernih senzora kao i rješenja primijenjena u izradi.
Opisati princip mjerenja brzine vjetra, a zatim i način razvoja vlastitog rješenja senzora pomoću optičke vilice i diska enkodera.
Nakon izrade sustava i programskog koda, potrebno je testirati funkcionalnost senzora, te navesti moguća unaprjeđenja u budućnosti.

ZADATAK URUČEN 30.6.2023. | POTPIS MENTORA Josip Srpak

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER

Predgovor

Zahvaljujem se svojem mentoru dipl.ing. Josipu Srpak koji mi je pomogao prilikom izrade i dokumentacije završnog rada. Konzultacijama i savjetima pomogao mi je u rješavanju problema koji su se pojavili tokom izrade ovog rada. Također bih htio zahvaliti svim ostalim profesorima Sveučilišta Sjever na pomoći i prenesenom znanju kroz studij te velika hvala obitelji, djevojci, prijateljima i kolegama na podršci.

Ideja za ovaj rad proizašla je iz ljubavi prema senzorima, mikroupravljačima, ugradbenim (tzv. embedded) sustavima i općenito iz interesa prema izradi uređaja. Još kao mali htio sam izraditi neki uređaj ili automatizirani sustav, odnosno htio sam biti izumitelj kad odrastem. Smatram da je širok spektar znanja koje mehatronika donosi vrlo dobar alat za izradu novih uređaja i sustava koji se potencijalno mogu probiti na tržištu.

Sažetak

PICadillo-35T interesantan je ugradbeni modul koji ima veliku raznolikost značajki. Može se koristiti za razne projekte, odnosno za izgradnju prijenosnih uređaja. U ovom završnom radu bit će opisan proces izrade meteorološke stanice primjenom PIC32 ugradbenog sustava sa zaslonom. Uz meteorološku stanicu zadatak je bio izraditi anemometar, odnosno senzor brzine vjetra. Opisuju se komponente potrebne za realizaciju i izgradnju stanice isto kao i postupak programiranja PIC32 mikrokontrolera. Spominju se određeni problemi koji su se pojavili prilikom izrade ovoga projekta. Kao i uvijek, postoji mogućnost nadogradnje stanice ugradnjom dodatnih senzora (npr. senzor padalina, senzor točke rosišta, i sl.). Isto tako se stanica može bazirati na nekom drugom mikrokontroleru s još više značajki, kao što je ESP32.

Ključne riječi: meteorološka stanica, PICadillo, PIC32, IoT, mikrokontroler, senzori

Abstract

PICadillo-35T is an interesting embedded module that has a wide variety of features. It can be used for various projects, for example for building portable devices. This paper contains the process description of making a weather station using a PIC32 embedded system with a display. Apart from building the weather station there was a task of making an anemometer, in other words a „wind speed sensor“. There is information about all of the components used for building the station as well as the process of programming the PIC32 microcontroller. Certain challenges, which are described here, appeared during this project. As always, there is room for improvement in terms of upgrading the station with other types of sensors (rainfall sensor, dew point sensor, etc.). The weather station can also be based on some other microcontroller with even more features, such as ESP32.

Keywords: weather station, PICadillo, PIC32, IoT, microcontroller, sensors

Popis korištenih kratica

4G	4th Generation mobile networks četvrta generacija mobilnih mreža
CAD	Computer-Aided Design dizajn potpomognut računalom
CAE	Computer-Aided Engineering inžinjering potpomognut računalom
Flash	Flash memory vrsta EEPROM elektroničke memorije
I^2C	Inter-Integrated Circuit serijsko komunikacijsko sučelje/protokol
IoT	**„**Internet of Things“ „Internet stvari“
LoRa	Long Range radio-komunikacijska tehnika
LoRaWAN	Long Range Wide Area Network radio-komunikacijski protokol/arhitektura
MQTT	Message Queuing Telemetry Transport Technical Committee mrežni protokol koji se često koristi u IoT-u
RTC	Real Time Clock elektronički uređaj koji mjeri prolazak vremena
SPI	Serial Peripheral Interface serijsko komunikacijsko sučelje/protokol
SRAM	Static random-access memory statička RAM memorija, jedna od izvedbi poluvodičke memorije
WiFi	**„**Wireless Fidelity“ bežični mrežni protokol

Sadržaj

1. Uvod

Uz malo mašte, volje i znanja moguće je realizirati velik broj ugradbenih sustava pomoću raznih mikroupravljača koji iz godine u godinu postaju sve brži, bolji i kvalitetniji zahvaljujući razvoju elektronike, automatizacije te IoT-a (Internet of Things).

Cilj ovoga završnoga rada jest prikaz procesa izrade funkcionalnog ugradbenog sustava sa zaslonom, u ovom slučaju meteorološke stanice koja sadrži više mjernih instrumenata.

Slika 1.1. – površinska meteorološka postaja [1]

Na slici 1.1. može se vidjeti profesionalnu površinsku meteorološku postaju. Bijela kuglasta građevina s lijeve strane slike neizbježna je kod svake ozbiljnije meteorološke stanice/postaje. Naime, radi se o meteorološkom (Doppler) radaru koji radi u više frekvencijskih područja. Ovaj tip radara služi za određivanje intenziteta i područja oborina te strukture oblaka u atmosferi. [1]

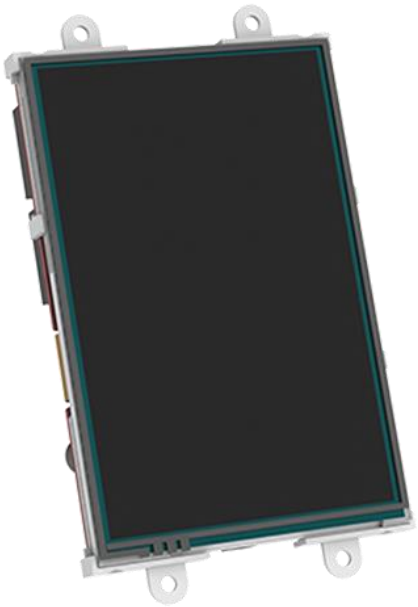
Tipična meteorološka stanica sadrži, osim radara, sljedeće mjerne instrumente:

- termometar,
- barometar,
- anemometar,
- higrometar (vlagomjer),
- pluviometar (kišomjer),
- vjetrokaz i sl.

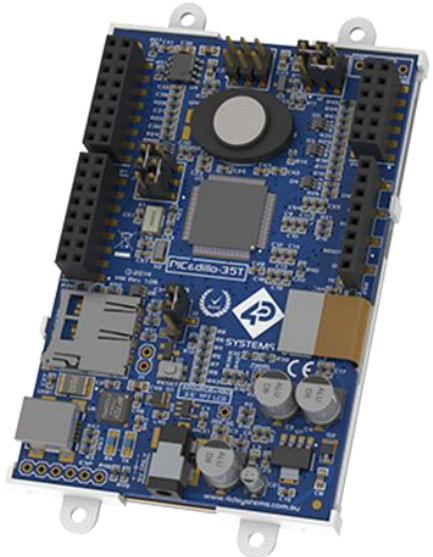
2. PICadillo-35T ugradbeni modul

Za ovaj projekt odabran je **PICadillo-35T** ugradbeni modul iz razloga što sadrži ugrađeni zaslon na dodir (*tzv. touchscreen*). Naziv modula inspiriran je tradicionalnim jelom mnogih latino-američkih država zvanim *picadillo*.

PICadillo-35T je ugradbeni modul sa rezistivnim zaslonom na dodir, baziran je na Microchip-ovom **PIC32MX795F512L** 32-bitnom mikroupravljaču.



Slika 2.1. – PICadillo-35T, prikaz prednje strane



Slika 2.2. – PICadillo-35T, prikaz stražnje strane

2.1. Specifikacije modula

PICadillo-35T osim zaslona na dodir sadrži audio pojačalo i zvučnik, RTC oscilator te microSD utor za karticu. Detaljne specifikacije nalaze se na tablici 2.1.

Tablica 2.1. – specifikacije PICadillo-35T modula [2]

PICadillo-35T	
Dijagonala zaslona	3.5 incha
Rezolucija zaslona	320x480 piksela
Dimenzije modula (mm)	56.9 x 97.6 x 15.7
Dimenzije zaslona (mm)	73.4 x 49
Masa modula	56 g
Luminancija	~216 cd/m^2
Vrsta touch zaslona	rezistivni touch
Dozvoljena radna temperatura	-15 °C do +65 °C
Dozvoljena temperatura skladištenja	-30 °C do +70 °C
Napon napajanja	5V DC
Struja napajanja	~250 mA
Radni takt	80 MHz
Količina različitih boja	RGB 65K
Flash memorija	512 KB
RAM memorija	125 KB SRAM
I2C	4 kanala
SPI	3 kanala
GPIO priključci	47 priključaka

3. Meteorološka stanica

Nekada ljudi nisu mogli predvidjeti prognozu pomoću tehnologije, no zasigurno su mogli proučavati ponašanje biljaka i životinja te kretanje i formiranje oblaka.

Razvojem tehnologije za trenutno meteorološko stanje više nije potrebno promatrati prirodu, već to mogu činiti relativno male meteorološke stanice koje vrše mjerenja zraka i oborina.

Slika 3.1. – krovna meteorološka stanica [3]

Na slici 3.1. prikazana je meteorološka stanica montirana na krovu kuće. Takva stanica je imobilna i fiksna; nije praktična za mjerenje na više lokacija.

U svrhu mobilizacije stanice ona može biti manjih dimenzija te može biti napajana baterijama. Zbog eliminacije ovisnosti sustava o nekoj mreži (npr. **WiFi**, **4G**, **LoRa**, **LoRaWAN** i sl.) svi mjereni podaci mogu biti prikazani na zaslonu stanice.

Za ovaj projekt koristit će se sljedeći mjerni instrumenti: termometar, barometar, higrometar i anemometar.

3.1. Odabrane komponente

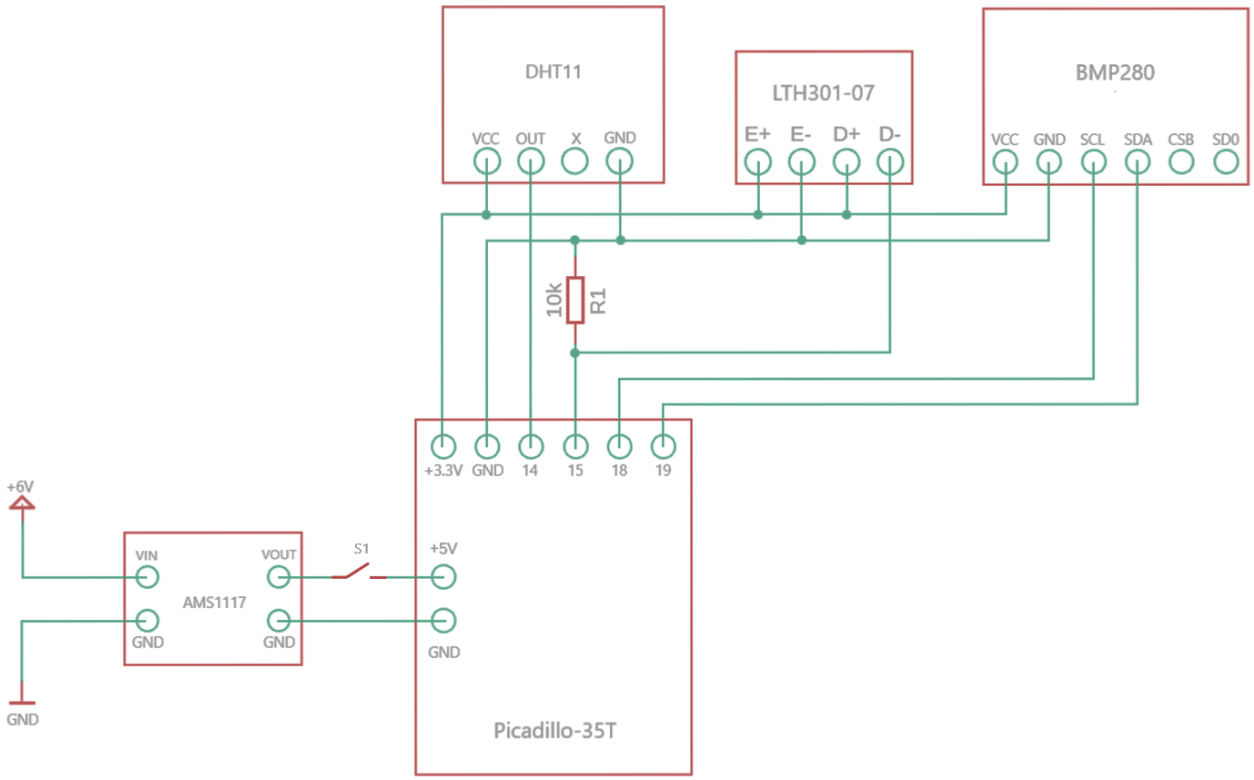
Za izradu meteorološke stanice potreban je već naveden modul **PICadillo-35T** te senzori koji će činiti mjerne instrumente.

Popis mjernih instrumenata i komponenata:

- termometar – BMP280
- barometar – DHT11
- higrometar – BMP280
- anemometar – LTH301-07 IC krajnji prekidač i enkoderski disk

Cijeli sustav bit će napajan pomoću 4x AA baterije uz regulator napona za 5V. Baterije će biti spojene na regulator napona, a izlaz regulatora će biti spojen na priključke modula VIN i GND.

Osim navedenih komponenata potreban je prekidač i otpornik od 10 kΩ. Električna shema prikazana je na slici 3.2., a veći prikaz sheme nalazi se u prilogu rada.



Slika 3.2. – električna shema sustava

3.1.1. BMP280 – senzor temperature i tlaka

Slika 3.3. – BMP280 [4]

BMP280 je senzor barometarskog tlaka koji osim mjerenja tlaka ima sposobnost mjerenja temperature zraka. Ovaj 6-pinski modul podržava SPI i I^2C protokole.

Nije ga potrebno kalibrirati pošto je prethodno tvornički kalibriran.

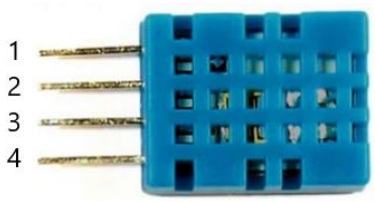
Tablica 3.1. – opis priključaka BMP280 senzora [4]

Broj priključka	**Naziv priključka**	**Opis priključka**
1	VCC	+3.3V
2	GND	0V
3	SCL	Serial clock pin (**I^2C**)
4	SDA	Serial data pin (**I^2C**)
5	CSB	Chip select pin **LOW = SPI**, **HIGH = I^2C**
6	SDO	Serial data output

Tablica 3.2. – tehničke specifikacije BMP280 [5]

BMP280	
Dimenzije kućišta	2 x 2.5 x 0.95 mm
Raspon rada	Tlak: 300 – 1100 hPa Temperatura: 0 – 65°C
Komunikacijski protokoli	I^2C i SPI
Razlučivost podataka	Tlak: 0.18 Pa Temperatura: 0.01 K
Apsolutna točnost **(za p = 950 – 1100 hPa)**	~ ±1 hPa

3.1.2. DHT11 – senzor vlage



Slika 3.4. – DHT11 [6]

DHT11 je senzor temperature i vlage s kalibriranim digitalnim izlaznim signalom. Ima 4 priključka od kojih se jedan ne koristi (priključak 3). Za ovu stanicu neće se koristiti podatak o temperaturi zbog uporabe senzora s boljom razlučivošću (BMP280).

Tablica 3.3. – opis priključaka

Broj priključka	**Naziv priključka**	**Opis priključka**
1	VCC	+3.3V do +5.5V DC
2	Signal	Izlazni priključak
3	X	Ne koristi se
4	GND	0V

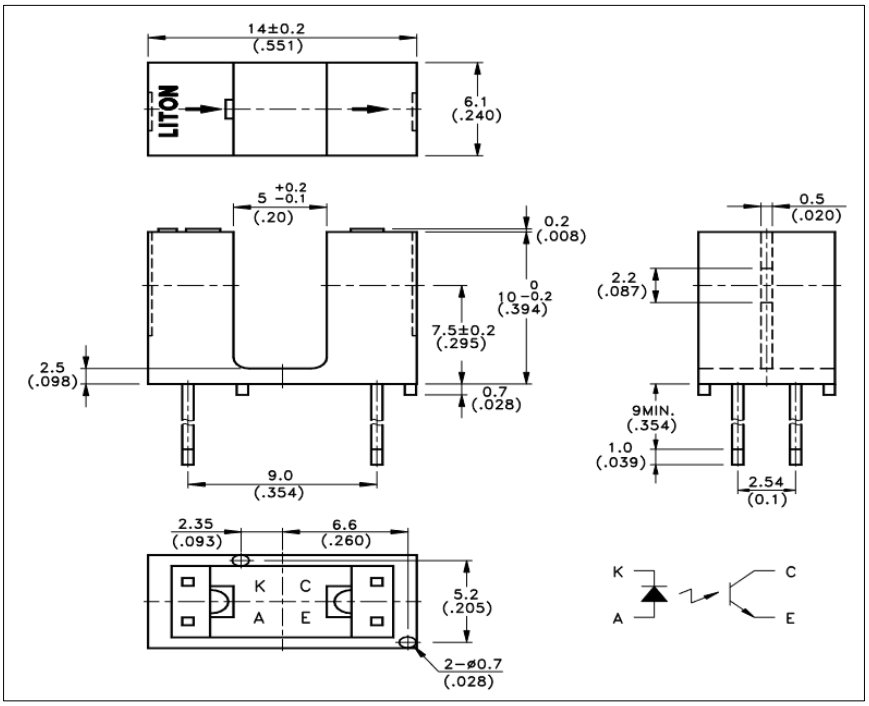
Tablica 3.4. – tehničke specifikacije DHT11 [7]

DHT11	
Dimenzije kućišta	12 x 15.5 x 5.5 mm
Raspon rada	Relativna vlažnost: 20% do 90% Temperatura: 0°C do 50°C
Razlučivost podataka	16 bit
Apsolutna točnost	Relativna vlažnost: ±1% Temperatura: ±1°C

3.1.3. LTH301-07 – optički krajnji prekidač

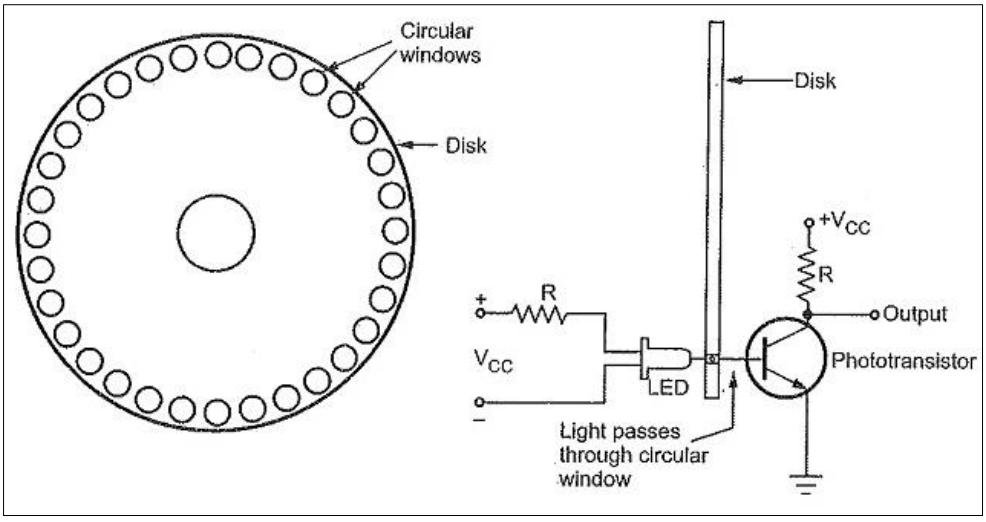
LTH301-07 je optički krajnji prekidač (eng. interrupter). S jedne strane optičke vilice nalazi se infracrvena LED dioda koja emitira infracrvene zrake koje u konačnici lovi fototranzistor.

Na slici 3.5. prikazane su dimenzije optičke vilice isto kao i shema sa priključcima.



Slika 3.5. – LTH301-07 [8]

Princip rada anemometra zasniva se na mjerenju količine impulsa LTH301-07 optičke vilice u jednoj sekundi. Na slici 3.6. nalazi se grafički prikaz principa rada enkodera.

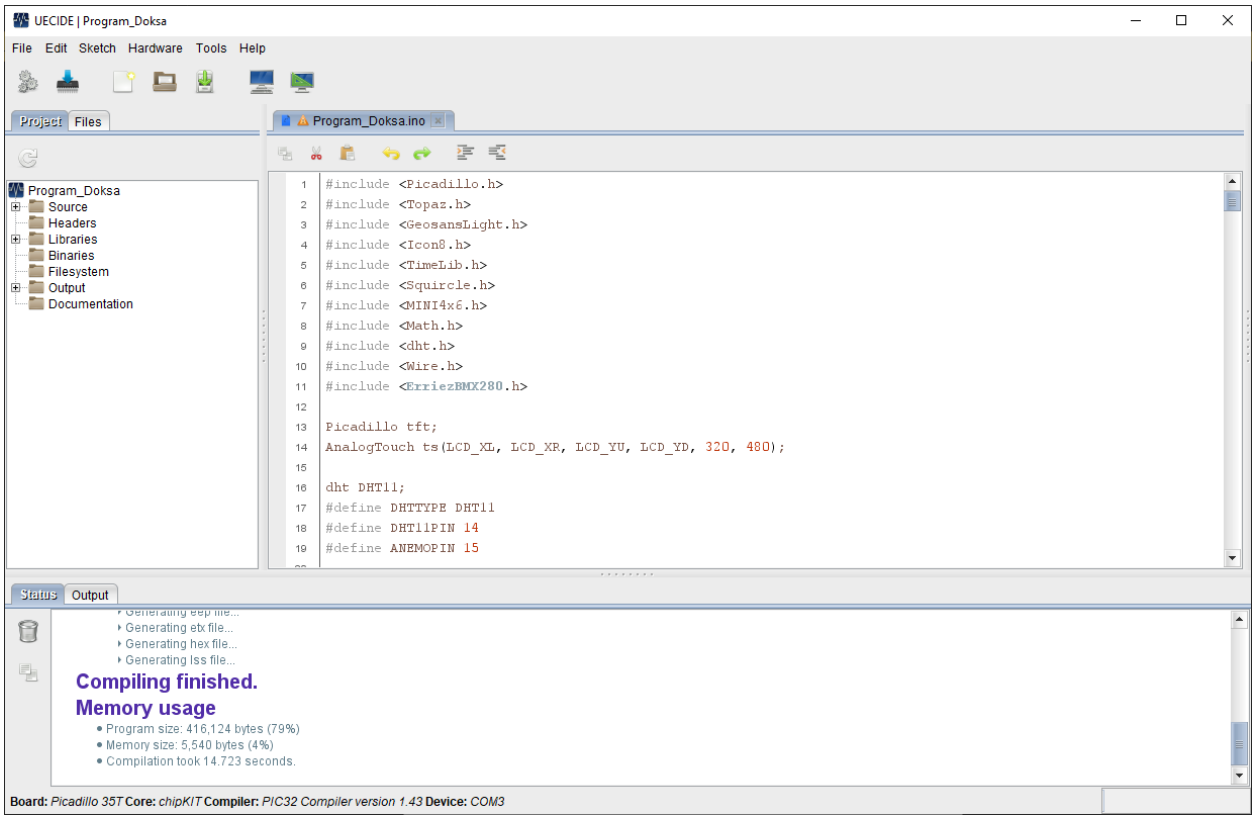


Slika 3.6. – princip rada enkodera [9]

4. Programiranje PICadillo-35T modula

Za programiranje navedenog modula potrebno je koristiti UECIDE razvojno okruženje. Popis korištenih biblioteka za ovaj projekt:

- Picadillo,
- Topaz,
- GeosansLight,
- Icon8,
- Squircle,
- MINI4x6,
- Math,
- dht,
- Wire,
- ErriezBMX280.

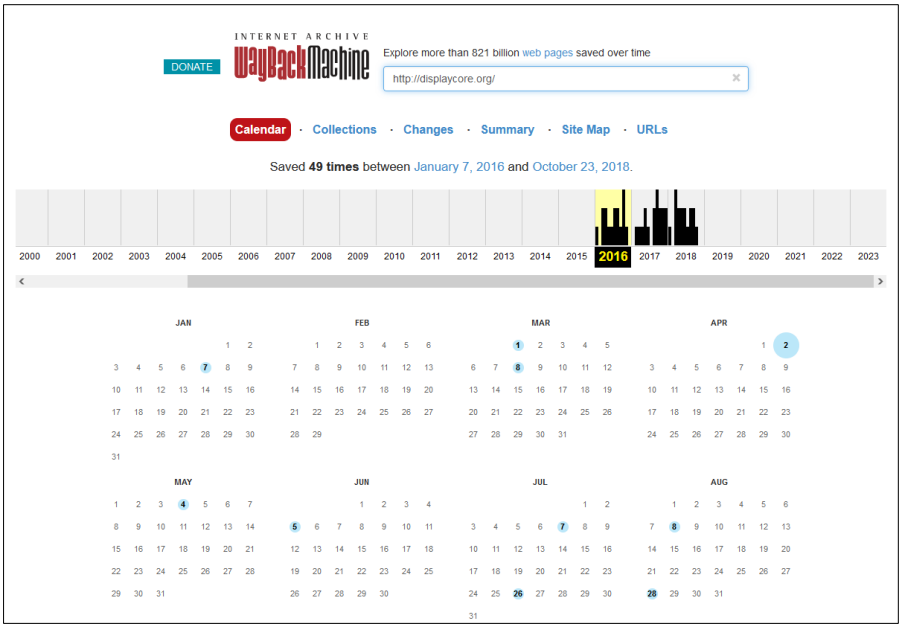


Slika 4.1. – izgled UECIDE razvojnog okruženja

4.1. Pripreme za programiranje

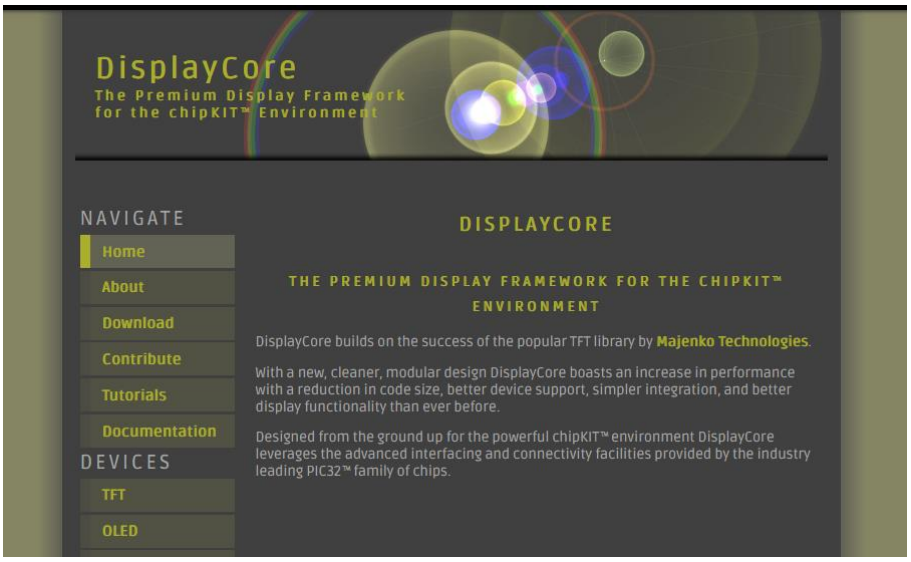
Prvi pokušaj testiranja mikroupravljača nije bio uspješan zbog nedostupnosti biblioteke zvane „DisplayCore“. Navedena biblioteka omogućuje korištenje mnogih zaslona vezanih uz PIC32 mikroupravljače.

Ovaj problem riješen je korištenjem stranice WayBack Machine (slika 4.2.) koja omogućuje pristup prijašnjim verzijama željene stranice.



Slika 4.2. – stranica WayBack Machine [10]

Pritiskom na bilo koji od označenih datuma otvara se displaycore.org stranica (slika 4.3.).



Slika 4.3. – web stranica display.org [11]

Ovdje se može pronaći dokumentacija te ostali materijali vezani za razne vrste zaslona. Između ostalog ovdje se nalaze datoteke DisplayCore biblioteke.

4.2. Izračun brzine vjetra

Enkoderski disk unutar anemometra ima 24 rupe, što znači da za 1 puni okretaj senzor treba poslati 24 impulsa. Ako u roku od sekunde senzor pošalje 24 impulsa, može se zaključiti da je to 1 okretaj u sekundi. Ako pak senzor u roku od sekunde pošalje 42 impulsa, to su 2 okretaja u sekundi. Isto vrijedi i za npr. 12 impulsa, to je 0.5 okretaja u sekundi. Formula za dobivanje okretaja u sekundi glasi:

$$\mathbf{RPS} = \frac{\mathbf{impulsi\ u\ sekundi}}{\mathbf{24}}$$

gdje je:

RPS = broj okretaja u sekundi.

Broj okretaja u sekundi može se pomnožiti sa 60 dobivanje broja okretaja u minuti:

$$\mathbf{RPM} = \mathbf{RPS} * \mathbf{60}$$

gdje je:

RPM = broj okretaja u minuti.

Kod jednog okretaja lopatica prekrije put jednak opsegu kruga. Planirana udaljenost centra kružnice lopatice od centra gornjeg dijela anemometra jest **~10 cm**. Formula za opseg kružnice jest **2rπ**. Ako se u tu formulu uvrsti prethodno definiran polumjer, opseg kružnice iznosi **20π cm**, odnosno **62.83 cm**.

Pomoću opsega kružnice može se izračunati brzina gibanja lopatice. Za primjer se može uzeti npr. 60 okretaja u minuti, to znači da sredina lopatice prijeđe linearni put od 60 x 62.83 cm u minuti. Dakle, to se može zapisati ovako:

$$\frac{60 * 62.83\ cm}{1\ min} * \frac{60\ min}{1\ h} * \frac{1\ km}{100000 cm} = 60 * 0.0377 = 2.262\ km/h$$

U principu se može bilo koji broj okretaja u minuti pomnožiti sa 0.0377 za dobivanje brzine lopatice u kilometrima na sat.

Nakon što se izračuna brzina lopatice, jednostavno je izračunati brzinu vjetra. Anemometar se giba 1/6 brzine vjetra, stoga se može pomnožiti brzina lopatice sa 6 za dobivanje brzine vjetra u km/h te se taj iznos podijeli sa 3.6 za brzinu u metrima u sekundi [12]:

$$brzina\ vjetra = \frac{brzina\ lopatica * 6}{3.6}$$

U ovom slučaju:

$$brzina\ vjetra = \frac{2.262\ km/h * 6}{3.6} = 3.77\ m/s$$

4.3. Programski kod

U nastavku se nalazi opis programskog koda koji omogućuje rad meteorološke stanice. Cijeli kod bez komentara nalazi se u prilogu ovog rada.

Na početku programa potrebno je deklarirati i „uključiti“ (eng. include) pojedine biblioteke. To se vrši korištenjem naredbe `#include`.

```
#include <Picadillo.h>
#include <Topaz.h>
#include <GeosansLight.h>
#include <Icon8.h>
#include <TimeLib.h>
#include <Squircle.h>
#include <MINI4x6.h>
#include <Math.h>
#include <dht.h>
#include <Wire.h>
#include <ErriezBMX280.h>
```

Nakon deklariranja biblioteka potrebno je inicijalizirati PICadillo modul i zaslon.

```
Picadillo tft;
AnalogTouch ts(LCD_XL, LCD_XR, LCD_YU, LCD_YD, 320, 480);
```

Kada se odradi inicijalizacija modula i zaslona, deklarira se DHT11 senzor te se definira tip DHT senzora, pošto biblioteka podržava i korištenje DHT22 senzora. Uz to se definira izlazni pin DHT11 senzora te izlazni pin optičke vilice anemometra pretprocesorskom naredbom `#define`.

```
dht DHT11;
#define DHTTYPE DHT11
#define DHT11PIN 14
#define ANEMOPIN 15
```

Deklariraju se i inicijaliziraju varijable za provjeru količine znamenaka u svrhu čišćenja zaslona (više o tome kasnije u kodu).

```
int16_t lastTemp = 0;
int16_t lastPressure = 0;
int16_t lastHumidity = 0;
```

Deklaracija i inicijalizacija početnih zastavica.

```
bool initVal = true;
bool flag = false;
bool init_flag = true;
```

Inicijalizacija broja rotacija i parametara za mjerenje proteklog vremena, odnosno vremena od 1 sekunde nakon kojeg će se ekran osvježiti.

```
volatile uint32_t rot; // broj rotacija
unsigned long measureTime = 0;
unsigned long startMillis;
unsigned long currentMillis;
const unsigned long period = 1000; // refresh rate (svakih 1000ms)
```

Korištenjem **ErriezBMX280** biblioteke inicijalizira se BMP280 senzor. Ovisno o komunikacijskom protokolu koji se koristi potrebno je definirati adresu **0x76** za **I²C** protokol, odnosno **0x77** za **SPI** protokol.

```
ErriezBMX280 bmx280 = ErriezBMX280(0x76);
```

Funkcija koja uzima neki broj i vraća količinu znamenaka.

```
int numDigits(int x)  {
    x = abs(x);
    return (x < 10 ? 1 :
         (x < 100 ? 2 :
         (x < 1000 ? 3 : 4
         )));
}
```

Funkcija kojom se ekran postepeno uključuje.

```
void fadeUp() {
  for (int i = 0; i < 255; i++) {
    tft.setBacklight(i);
    delay(1);
  }
}
```

Funkcija kojom se ekran postepeno isključuje.

```
void fadeDown() {
  for (int i = 0; i < 255; i++) {
    tft.setBacklight(254 - i);
    delay(1);
  }
}
```

Funkcija koja uzimajući koordinate točke briše liniju desno od te točke, odnosno briše bilo kakva slova, brojeve i simbole.

```
void clearLine(uint16_t x, uint16_t y) {
  tft.setCursor(x, y);
  tft.print("                                          ");
  Serial.println("LINE CLEARED");
}
```

Funkcija koja crta dugačku horizontalnu liniju na sredini zaslona.

```
void drawLine() {
  tft.drawHorizontalLine(20, 180, 440, Color::DarkOrange);
}
```

Funkcija koja u gornjem lijevom kutu zaslona prikazuje ime i prezime autora. Valja napomenuti da u GeosansLight fontu nema dijakritičkih znakova pa je bilo potrebno na vrhu slova „s“ prikazati jako malo slovo „v“ kako bi se dobilo slovo „š“.

```
void displayName() {
  tft.setFont(Fonts::GeosansLight16);
  tft.setCursor(14, 14);
  tft.print("Autor:");
  tft.setCursor(14, 34);
  tft.print("Robert Doksa");

  // za kvacicu na slovu s
  tft.setCursor(86, 36);
  tft.setFont(Fonts::MINI4x6);
  tft.print("v");
  tft.setFont(Fonts::GeosansLight16);
}
```

Slijedi funkcija kojom se prikazuju sve vrijednosti na zaslonu. Kao ulazne argumente uzima temperaturu, tlak zraka, relativnu vlažnost i brzinu vjetra.

```
void displayValues(float temperature, float pressure, int humidity, float
windSpeed) {
```

Za lakši ispis podataka koristi se buffer imenom „snum“. U njega se stavlja sadržaj koji će se ispisati. Uz „snum“ se definira zastavica „clearFlag“ te redni broj simbola stupnja u Squircle fontu.

```
  char snum[50];
  bool clearFlag = false;
  const char deg = 167;    //definiranje simbola stupnja u ASCII kodu
```

Slijedi ispis temperature. Prvo se postavi font GeosansLight20 kako bi se ispisao tekst „Temperatura“. Postavljeni su razni uvjeti za čišćenje linije kako bi se izbjeglo preklapanje simbola. Na kraju ove sekcije ispiše se simbol za stupnjeve Celzijuseve, no u fontu GeosansLight u korištenoj biblioteci nema simbola za stupnjeve, stoga se koristi drugi font samo za taj simbol.

```
//TEMPERATURA PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setTextColor(Color::DarkOrange, Color::Gray5);
  tft.setCursor(28, 210);
  tft.print("Temperatura");

  tft.setFont(Fonts::GeosansLight34);

  /* u slucaju razlike broja znamenki trenutne temperature i prosle
  temperature, ili u slucaju aktivne zastavice clearFlag, ocisti se
  linija kako se nebi preklapali simboli */

  if(numDigits(temperature) != lastTemp || clearFlag) {
    clearLine(28, 250);
    clearFlag = false;
  }
  tft.setCursor(28, 250);
  if(temperature < 0 || temperature > 50) {
     clearFlag = true;
     tft.print("       ");
  }
  if(temperature>=0 && temperature < 10){
     sprintf(snum, "%.1f ", temperature);
     tft.print(snum);
  }
  else if(temperature>=10 && temperature <100) {
     sprintf(snum, "%.1f ", temperature);
     tft.print(snum);
  }
  else if(temperature>=100 && temperature <1000) {
     sprintf(snum, "%.1f  ", temperature);
     tft.print(snum);
  }
  else {
     sprintf(snum, "%.1f", temperature);
     tft.print(snum);
  }
  lastTemp = numDigits(temperature);
  //promjena fonta u svrhu ispisa simbola stupnja
  tft.setFont(Fonts::Squircle);
  tft.print(deg);
  tft.setFont(Fonts::GeosansLight34);
  tft.print("C ");
```

Slijedi ispis tlaka zraka. Prvo se ispiše riječ „Tlak“ na definirano područje, a zatim se ispituje je li zadnji iznos tlaka imao više znamenaka od trenutnog tlaka. Ukoliko je uvjet ispunjen, linija se mora čistiti.

```
//TLAK PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setCursor(220, 210);
  tft.print("Tlak");
  tft.setCursor(175, 250);
  tft.setFont(Fonts::GeosansLight34);

  if(numDigits(pressure) != lastPressure) {
    clearLine(175, 250);
  }
  tft.setCursor(175, 250);
  if(pressure < 0 || pressure < 700) {
    clearLine(175, 250);
  }
  else {
    sprintf(snum, "%.2f hPa", pressure);
    tft.print(snum);
  }
  lastPressure = numDigits(pressure);
```

Nakon ispisa tlaka slijedi ispis vlažnosti zraka. Valja uočiti da se i tu pojavljuje dijakritički znak, odnosno slovo „ž“. Potrebno je ponovno s drugim fontom ispisati slovo „v“ iznad slova „z“.

```
//VLAGA PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setCursor(380, 210);
  tft.print("Vlaznost");

  tft.setCursor(408, 212);
  tft.setFont(Fonts::MINI4x6);
  tft.print("v");

  tft.setCursor(380, 250);
  tft.setFont(Fonts::GeosansLight34);

  if(numDigits(humidity) != lastHumidity || humidity <= 0) {
    clearLine(380, 250);
  }
  tft.setCursor(380, 250);
  sprintf(snum, "%d%%", humidity);
  tft.print(snum);
  lastHumidity = numDigits(humidity);
```

Posljednji ispis jest ispis brzine vjetra.

```
//VJETAR PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setCursor(230, 16);
  tft.print("Brzina vjetra");
  tft.drawHorizontalLine(200, 49, 155, Color::DarkOrange);
  tft.setFont(Fonts::GeosansLight38);

  tft.setCursor(230, 80);
  sprintf(snum, "%.1f m/s", windSpeed);
  tft.print(snum);
}
```

Funkcija fontTest( ) može se koristiti za provjeru svih simbola koje nudi font biblioteka.

```
void fontTest() {
  char a = 0;
  for(int i = 80; i <190; i++){
    Serial.println(i);
    a = i;
    tft.fillScreen(Color::Black);
    tft.setFont(Fonts::Squircle);
    tft.setTextColor(Color::DarkOrange);
    tft.setCursor(220, 210);
    tft.print(a);
    delay(500);
  }
}
```

Funkcija setup( ) je prva funkcija koja se izvrši prilikom pokretanja mikrokontrolera. Unutar te funkcije inicijalizira se zaslon/touchscreen, pokrene se serijska komunikacija brzinom 9600 bauda, te se pokrene I^2C komunikacija.

```
void setup() {
  ts.initializeDevice();
  Serial.begin(9600);
  Wire.begin();
  Wire.setClock(100000);
```

Nakon inicijalizacije i pokretanja komunikacije, vrijeme je za početak komunikacije sa BMP280 senzorom. Pokušava se pokrenuti komunikacija tako dugo dok ne započne, odnosno u beskonačnost, a ako BMP280 nije prepoznat, u serial monitor se ispiše „Error: BMP280 nije detektiran!“. Nakon uspostave komunikacije sa senzorom, konfiguriraju se parametri uzorkovanja.

```
  while (!bmx280.begin()) {
    Serial.println(F("Error: BMP280 nije detektiran!"));
    delay(3000);
  }

  bmx280.setSampling(BMX280_MODE_NORMAL,
                     BMX280_SAMPLING_X16,
                     BMX280_SAMPLING_X16,
                     BMX280_SAMPLING_X16,
                     BMX280_FILTER_OFF,
                     BMX280_STANDBY_MS_500);
```

Definiranje parametara zaslona i touchscreena (manje važno u ovome projektu). Konfiguriranje zaslona, postavljanje pozadinskog svjetla u 0. Postavljanje rotacije.

```
  // postavke za touchscreen
  ts.scaleX(4.5);
  ts.scaleY(3.4);
  ts.offsetY(5);

  tft.initializeDevice();
  tft.setBacklight(0);
  tft.setTextWrap(false);
  tft.setRotation(3);
  tft.fillScreen(Color::Gray5);
  tft.setFont(Fonts::GeosansLight40);
  tft.setTextColor(Color::DarkOrange, Color::Gray5);
```

Priključci na koje se spajaju izlaz DHT11 i izlaz anemometra definiraju se kao ulazi. Poziva se funkcija displayName( ) koja ispisuje ime i prezime autora u gornji desni kut. Nakon toga se definira kao ulaz. Početnom vremenu se definira trenutna vrijednost funkcijom millis( ).

```
  pinMode(DHT11PIN, INPUT);
  displayName();
  drawLine();

  pinMode(ANEMOPIN, INPUT);
  startMillis = millis();
}
```

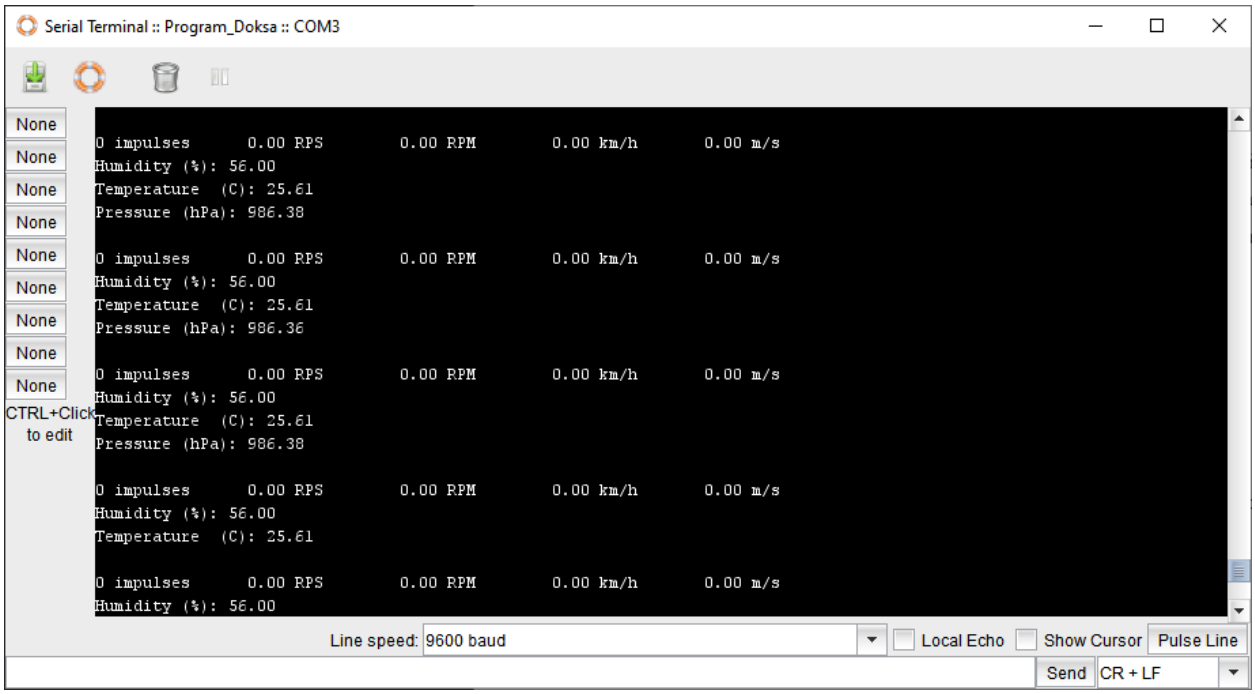
loop( ) funkcija koja očitava senzore i poziva funkciju za prikaz podataka na zaslonu.

```
void loop() {
  currentMillis = millis();

  if (currentMillis - startMillis >= period)
  {
    float rpm = ((float)rot/24) * 60;

    Serial.print(rot);
    Serial.print(" impulses\t");

    Serial.print((float)rot / 24);
    Serial.print(" RPS\t");

    Serial.print(rpm);
    Serial.print(" RPM\t");

    float wind_kmph = rpm * 0.03 * 6;
    float wind_ms = wind_kmph / 3.6;
    Serial.print(wind_kmph);
    Serial.print(" km/h\t");

    Serial.print(wind_ms);
    Serial.println(" m/s\t");

    int chk = DHT11.read11(DHT11PIN);

    Serial.print("Humidity (%): ");
    Serial.println((float)DHT11.humidity, 2);

    float temperature = bmx280.readTemperature();
    Serial.print("Temperature  (C): ");
    Serial.println(temperature, 2);

    float pressure = bmx280.readPressure() / 100.0F;
    Serial.print("Pressure (hPa): ");
    Serial.println(pressure);

    displayValues(temperature, pressure, DHT11.humidity, wind_ms);
    if(initVal) {
      fadeUp();
      initVal = false;
    }
    Serial.println();

    rot = 0;
    startMillis = currentMillis;
  }
```

Na kraju loop( ) funkcije broje se impulsi anemometra. Ukoliko je signal anemometra binarna jedinica, i ako su obje zastavice u stanju „false“, varijabla „rot“ povećava se za 1. Zastavice se poništavaju. Ukoliko je izlaz anemometra binarna nula, zastavice se postave na „false“ kako bi se omogućila inkrementacija u idućoj iteraciji. Zadnji uvjet je u slučaju prve iteracije. U tom slučaju se varijabla „rot“ postavlja u 0. Taj uvjet je postavljen zbog slučaja detekcije impulsa optičke vilice na početku rada bez da se anemometar okreće. Dakle, ne smije se procesuirati taj impuls.

```
  if(digitalRead(ANEMOPIN) == HIGH && flag == false && init_flag == false) {
    rot++;
    flag = true;
    init_flag = false;
    //Serial.println(rot);
  }
  if(digitalRead(ANEMOPIN) == LOW) {
    flag = false;
    init_flag = false;
  }
  if(init_flag == true) {
    rot = 0;
    flag = true;
  }
}
```

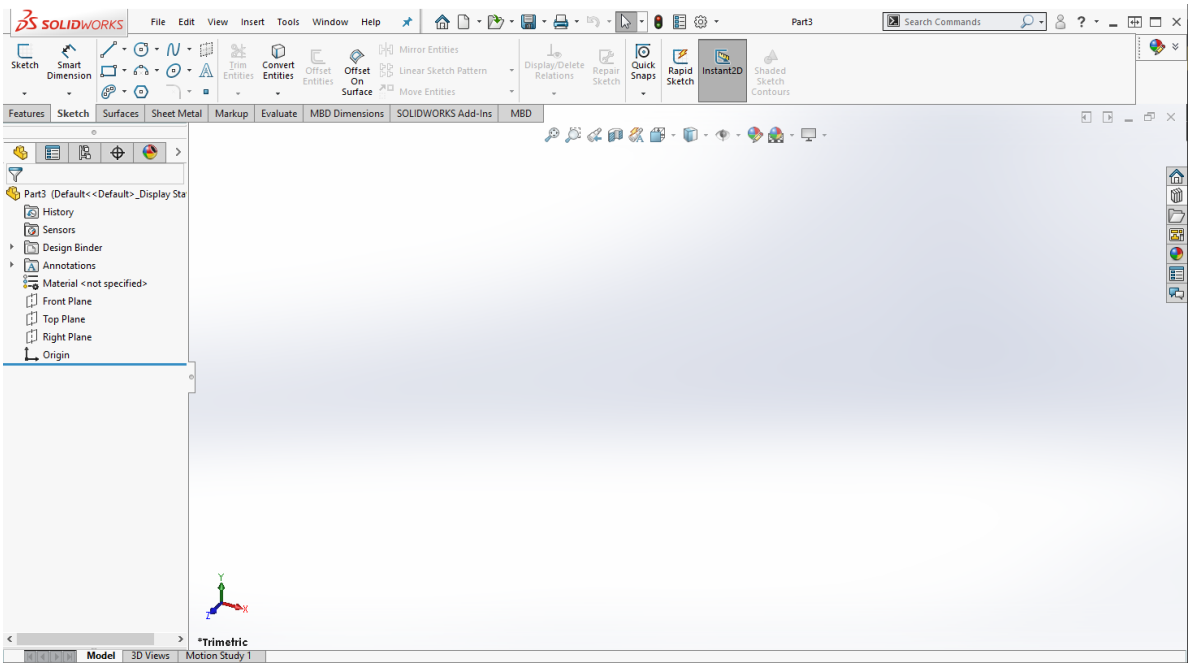
Serijskom komunikacijom podaci se mogu slati na računalo putem USB kabla. Prikaz tzv. serial monitora nalazi se na slici 4.4.



Slika 4.4. – serial monitor

5. Projektiranje dijelova

Za modeliranje svih dijelova korišten je SolidWorks software. SolidWorks je program za CAD dizajn i CAE inženjering. Više od 6 milijuna korisnika koristi ovaj programski paket u svrhu edukacije ili profesionalne upotrebe. [12]

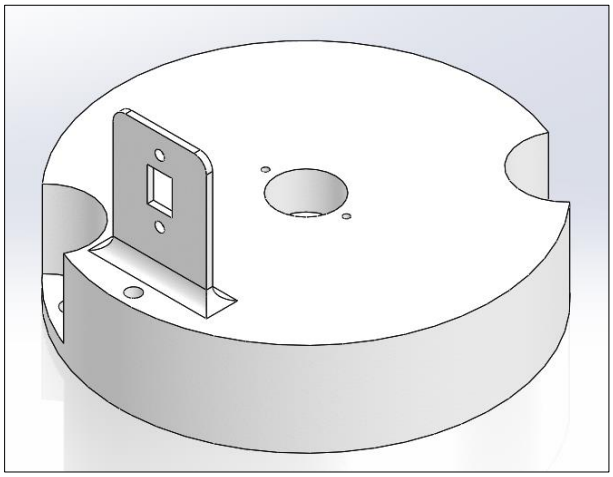


Slika 5.1. – SolidWorks software

Bez skiciranja i više pokušaja teško je napraviti kvalitetne dijelove nekog uređaja ili stroja. Iz ovog razloga slijede prikazi dvije verzije 3D modela projektiranih dijelova. Prva verzija pojedinih dijelova sadrži suvišan materijal, dok je u drugoj verziji taj materijal uklonjen.

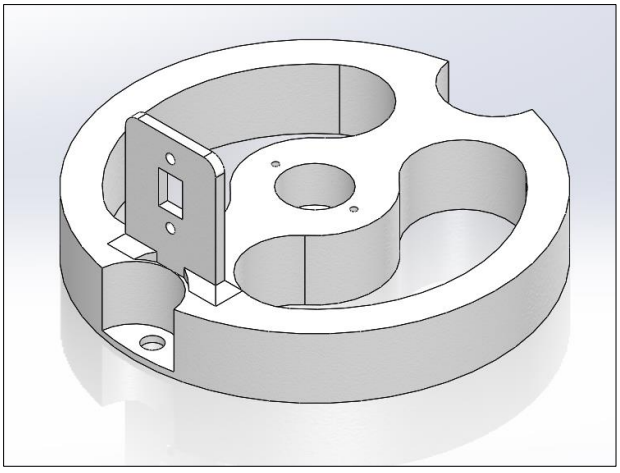
5.1. Baza anemometra

Na slici 5.2. nalazi se prva verzija baze anemometra. Ovaj dizajn baze sadrži previše suvišnog materijala koji se može ukloniti kako bi se smanjio trošak i vrijeme ispisa.



Slika 5.2. – baza anemometra, verzija 1

Na slici 5.3. nalazi se druga i finalna verzija baze anemometra. Uklonjen je velik dio suvišnog materijala dok je strukturalna čvrstoća zadržana.

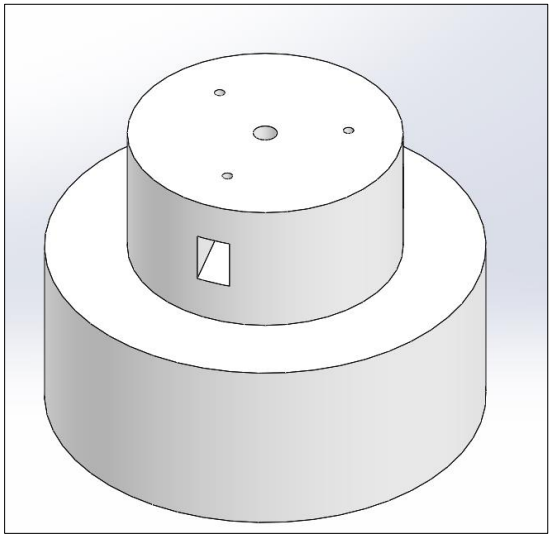


Slika 5.3. – baza anemometra, verzija 2

Na sredini se nalazi provrt za kuglične ležajeve kroz koje je provučena M6 navojna šipka. Okomiti dio baze ima funkciju podloge za optičku vilicu. Na bazi se nalaze i dva polukruga sa rupama za vijke koji se pričvršćuju na aluminijski profil.

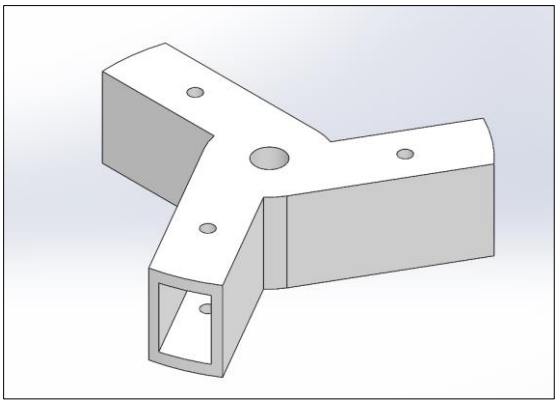
5.2. Gornji dio anemometra

Na slici 5.4. nalazi se prva verzija gornjeg dijela anemometra. Ovaj dizajn također sadrži previše suvišnog materijala koji je uklonjen zbog smanjena troška i vremena ispisa.



Slika 5.4. – gornji dio anemometra, verzija 1

Na slici 5.5. nalazi se druga i finalna verzija gornjeg dijela anemometra. Ovdje je razlika drastična; uklonjena je većina materijala dok je funkcionalnost tog dijela identična.

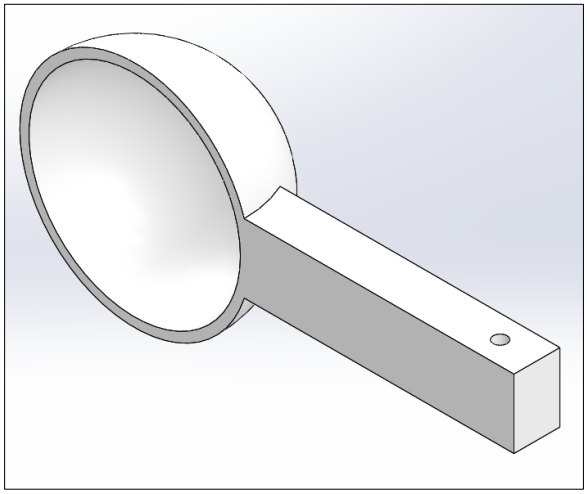


Slika 5.5. – gornji dio anemometra, verzija 2

Gore prikazan dio sadrži 3 utora za 3 lopatice anemometra. Otvori su međusobno razmaknuti za 120°. Na sredini se nalazi provrt za navojnu šipku koja se s gornje i donje strane pričvršćuje maticama. Ostali provrti služe za pričvršćenje lopatica sa tim dijelom.

5.3. Lopatice anemometra

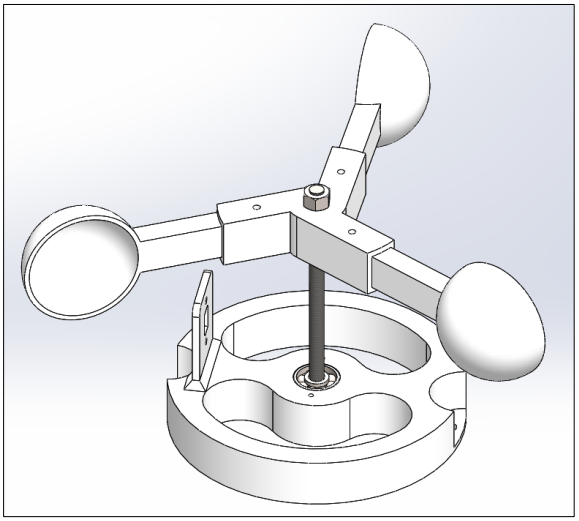
Za ovaj dizajn anemometra koriste se 3 lopatice (slika 5.6.). Ispuna lopatica nije 100% već 40% da budu čim lakše, što će u konačnici omogućiti gibanje uz minimalnu tromost.



Slika 5.6. – lopatica anemometra

5.4. Anemometar – sklop

Konačni sklop svih isprintanih dijelova anemometra prikazan je na slici 5.7.

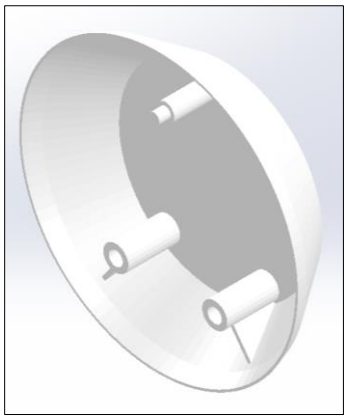


Slika 5.7. – sklop anemometra

5.5. Oklop senzora

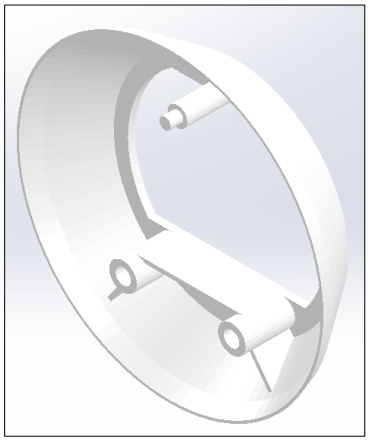
Senzori temperature, vlage i tlaka moraju biti nekako zaštićeni. Navedeni senzori mogu se staviti u tzv. Stevenson screen. [14].

Na slici 5.8. prikazan je gornji dio oklopa senzora koji štiti od padalina.



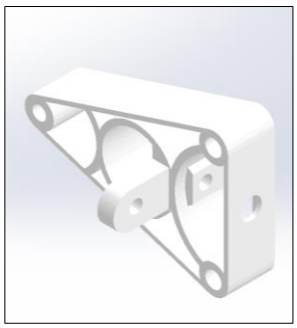
Slika 5.8. – gornji dio oklopa senzora

Na slici 5.9. prikazan je srednji dio oklopa senzora. Ovaj dio može se ponavljati više puta da se dobije čim viši oklop. U ovom slučaju oklop sadrži dva srednja dijela.



Slika 5.9. – srednji dio oklopa senzora

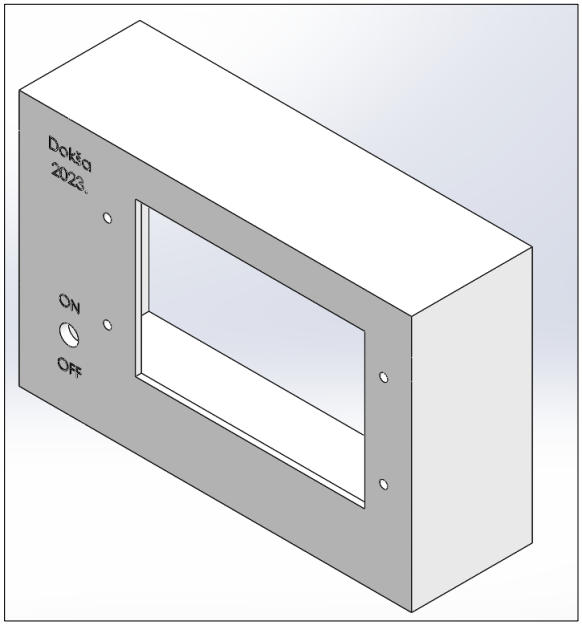
Na slici 5.10. prikazan je donji dio oklopa senzora koji povezuje oklop sa aluminijskim profilom.



Slika 5.10. – donji dio oklopa senzora

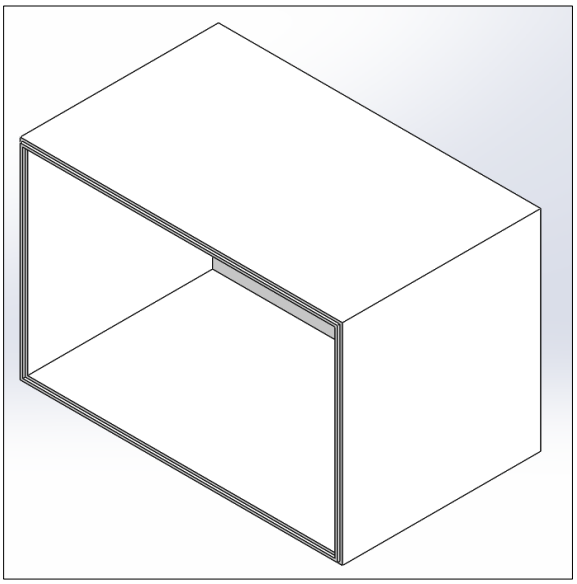
5.6. Kućište zaslona

Zbog čim jednostavnijeg sklapanja koristit će se dva dijela kućišta: prednji dio i stražnji dio. Prednji dio kućišta je nešto kraći od stražnjeg dijela, a ideja je da se može mijenjati ovisno o zaslonu koji se koristi.



Slika 5.11. – prednji dio kućišta

Stražnji dio kućišta je duži od prednjeg dijela kako bi se kućište za baterije moglo montirati unutar kućišta zaslona.



Slika 5.12. – stražnji dio kućišta

Sučeoni spoj polovica kućišta omogućuje spoj dijelova bez dodatnih spojnih elemenata. Uz ovakav spoj kućište izgleda kao da je od jednog komada, što doprinosi estetici cijelog sustava.

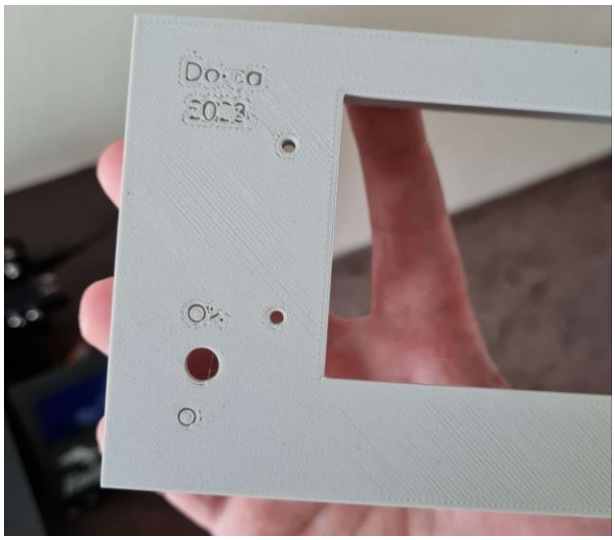
6. 3D printanje dijelova

Nakon odrađenog modeliranja i konstruiranja svih dijelova dolazi na red 3D printanje. Za 3D printanje korišteni su printeri Creality Ender 3 Neo i Zortrax M200.

Slika 6.1. – 3D printanje

Dijelovi oklopa senzora i kućište stanice isprintani su PLA plastikom, dok su dijelovi anemometra isprintani ABS plastikom.

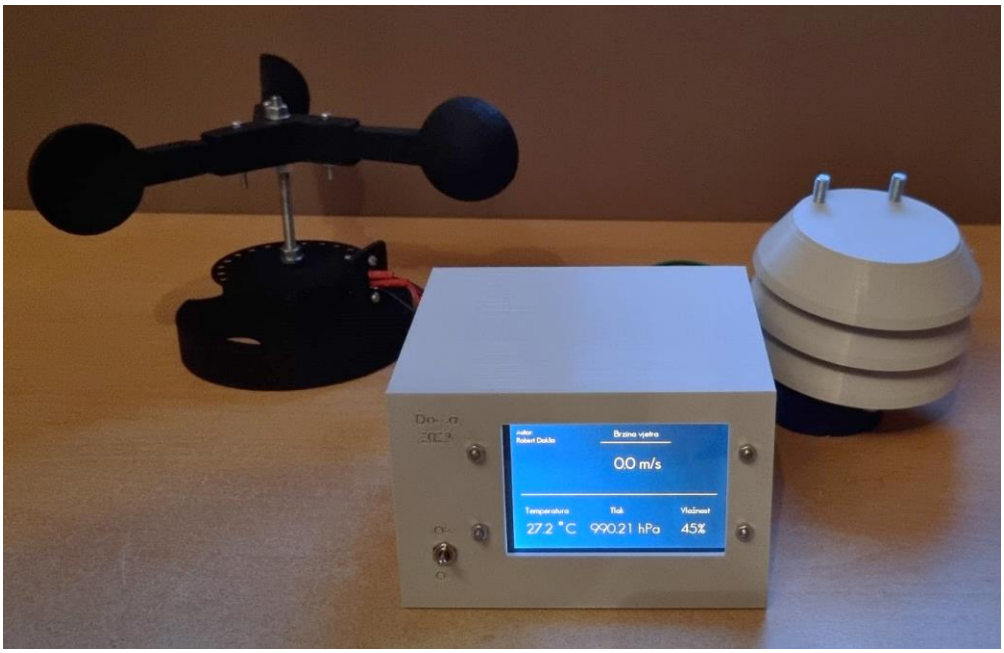
Prednji dio kućišta ispao je dobro osim udubljenih slova „Dokša 2023.“ i „ON OFF“ koja su se popunila filamentom (slika 6.2.).



Slika 6.2. – prednji dio kućišta, isprintan

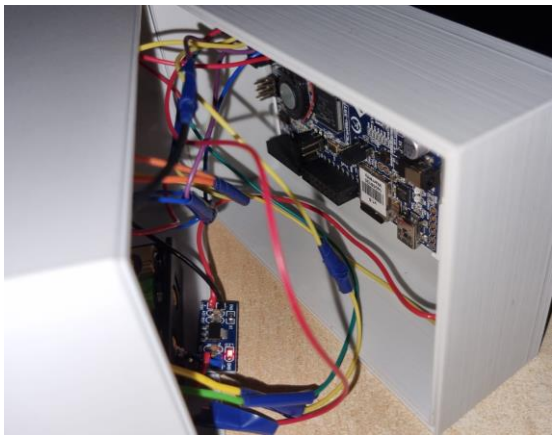
7. Sklapanje stanice

Nakon 3D printanja svih dijelova, lemljenja svih žica i komponenata, učvršćenja svih vijaka i navojnih šipki nastao je finalni sustav koji je prikazan na slici 7.1.

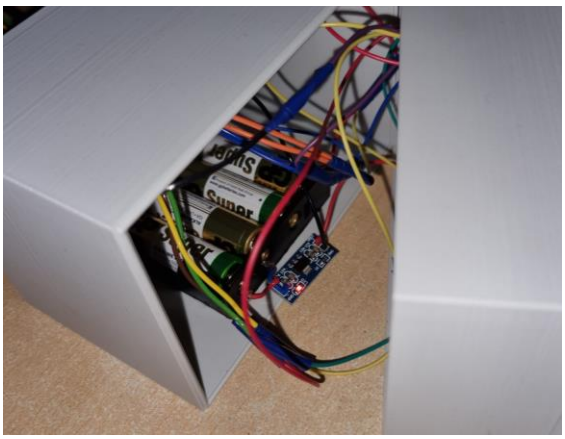


Slika 7.1. – prikaz izrađene meteorološke stanice

Na slikama 7.2. i 7.3. prikazana je unutrašnjost sklopljene stanice. Na donjem dijelu stražnjeg poklopca nalaze se četiri AA baterije u svom kućištu, dok se na prednjoj strani nalaze PICadillo modul i KIP preklopnik.



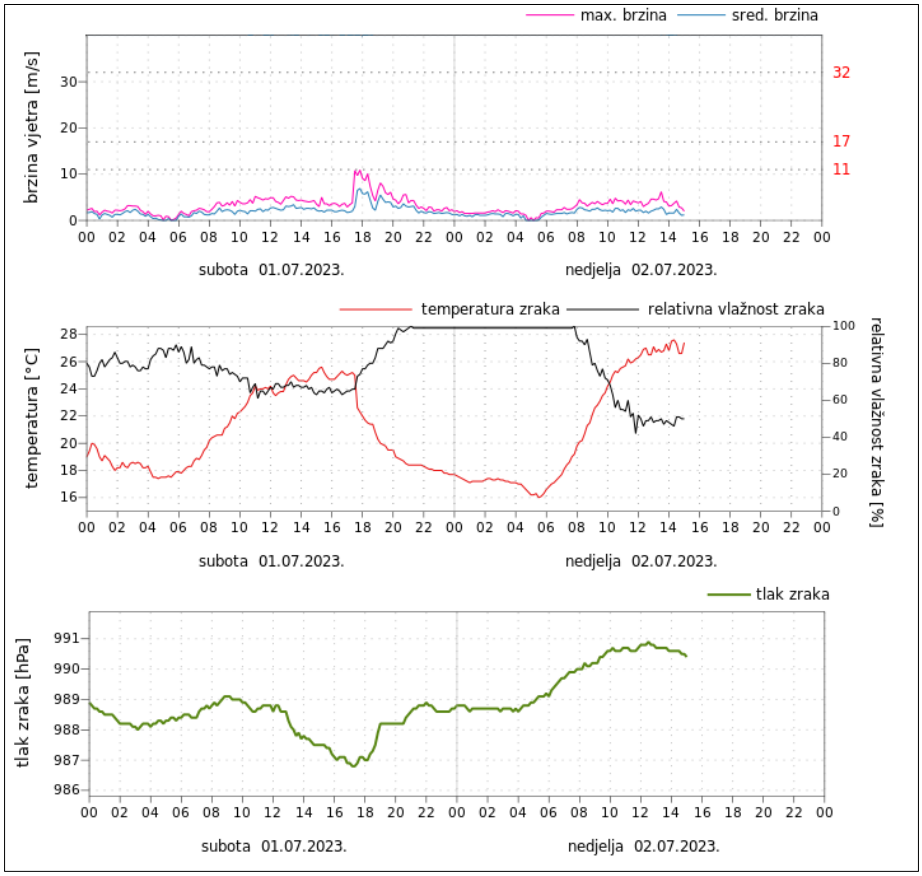
Slika 7.2. – unutrašnjost prednje strane kućišta stanice



Slika 7.3. – unutrašnjost stražnje strane kućišta stanice

8. Analiza rezultata

Najpouzdaniji meteorološki podaci u Hrvatskoj nalaze se na stranici Državnog Hidrometeorološkog Zavoda (DHMZ). Moguće je pristupiti podacima tih automatskih postaja na stranici DHMZ-a (slika 8.1.).



Slika 8.1. – DHMZ podaci [15]

Na slici 8.2. vide se podaci sa izrađene meteorološke stanice. Tlak uvelike ovisi o visini na kojoj se nalazi stanica; čim je visina veća, tlak zraka je manji. Anemometar je pokrenut rukom zbog nedostatka vjetra u trenutku testiranja. Podaci se poklapaju uz minimalna odstupanja.

Slika 8.2. – podaci meteorološke stanice

9. Zaključak

Za izradu ovog rada nije potrebno puno dijelova. Međutim, moguća je pojava mnogih problema: toplinske dilatacije 3D isprintanih komada, prevelika masa lopatica anemometra, loši ležajevi anemometra, oksidirani kontakti, itd.

Samo programiranje PICadillo modula nije prezahtjevno, no može biti problematično. Kod korištenja malo starijih mikrokontrolera moguće je (ali ne nužno) naići na poteškoće sa driverima i općenito probleme sa softverom.

Što se tiče odabira mikrokontrolera, za buduću poboljšanu verziju stanice može se koristiti neki noviji mikrokontroler poput ESP32 koji ima Bluetooth i WiFi podršku. Uporabom novijeg mikrokontrolera sustavu se može dodati funkcionalnost komunikacije s mobilnim uređajem ili računalom putem tzv. MQTT (skraćeno od Message Queuing Telemetry Transport Technical Committee) protokola korištenjem MQTT brokera. Također postoji varijanta komunikacije preko LoRa modula, no za to je potreban drugi mikrokontroler koji bi služio kao prijamnik.

Izrađena stanica mogla bi se bolje izolirati i zaštititi od vode i sitnih čestica. Također bi kućište moglo biti robusnije uz mogućnost montiranja na krov, balkon i sl.

Osim izrađenih mjernih instrumenata mogao bi se izraditi senzor smjera vjetra uporabom magnetometra koji bi davao podatak o kutu zakreta mjernog tijela u odnosu na sjever.

10. Literatura

[1] https://en.wikipedia.org/wiki/Weather_station, dostupno 2.7.2023.

[2] https://ambientweather.com/media/blog/sturdy-secure-choose-right-mounting-accessories-match-your-location.jpg, dostupno 2.7.2023.

[3] https://4dsystems.com.au/products/picadillo-35t/, dostupno 2.7.2023.

[4] https://www.apogeeweb.net/circuitry/bmp280-barometric-pressure-sensor.html, dostupno 2.7.2023.

[5] https://www.bosch-sensortec.com/media/boschsensortec/downloads/product_flyer/bst-bmp280-fl000.pdf, dostupno 2.7.2023.

[6] https://components101.com/sites/default/files/component_datasheet/DHT11-Temperature-Sensor.pdf, dostupno 2.7.2023.

[7] https://components101.com/sensors/dht11-temperature-sensor, dostupno 2.7.2023.

[8] https://eu.mouser.com/datasheet/2/239/H301-07-1141226.pdf, dostupno 2.7.2023.

[9] https://www.eeeguide.com/optical-encoders/, dostupno 2.7.2023.

[10] https://archive.org/web/, dostupno 2.7.2023.

[11] https://web.archive.org/web/20180313013713/http://displaycore.org:80/, dostupno 2.7.2023.

[12] https://sciencefirst.com/wp-content/uploads/2017/05/24-0506-05-005-652-1010-anemometer.pdf, dostupno 2.7.2023.

[13] https://www.javelin-tech.com/blog/2022/12/mechanical-design-certifications-from-solidworks/, dostupno 2.7.2023.

[14] https://en.wikipedia.org/wiki/Stevenson_screen, dostupno 2.7.2023.

[15] https://meteo.hr/podaci.php?section=podaci_vrijeme&param=amp&Code=varazdin, dostupno 2.7.2023.

Popis slika

Popis tablica

Prilozi

Programski kod:

```
#include <Picadillo.h>
#include <Topaz.h>
#include <GeosansLight.h>
#include <Icon8.h>
#include <TimeLib.h>
#include <Squircle.h>
#include <MINI4x6.h>
#include <Math.h>
#include <dht.h>
#include <Wire.h>
#include <ErriezBMX280.h>

// inicijalizacija PICadillo modula i touch zaslona
Picadillo tft;
AnalogTouch ts(LCD_XL, LCD_XR, LCD_YU, LCD_YD, 320, 480);

dht DHT11;
#define DHTTYPE DHT11
#define DHT11PIN 14
#define ANEMOPIN 15

int16_t lastTemp = 0;
int16_t lastPressure = 0;
int16_t lastHumidity = 0;

bool initVal = true;
bool flag = false;
bool init_flag = true;

volatile uint32_t rot; // broj rotacija
unsigned long measureTime = 0;
unsigned long startMillis;
unsigned long currentMillis;
const unsigned long period = 1000; // refresh rate (svakih 1000ms)

ErriezBMX280 bmx280 = ErriezBMX280(0x76);

int numDigits(int x)  {
    x = abs(x);
    return (x < 10 ? 1 :
         (x < 100 ? 2 :
         (x < 1000 ? 3 : 4
         )));
}

void fadeUp() {
  for (int i = 0; i < 255; i++) {
    tft.setBacklight(i);
    delay(1);
  }
}

void fadeDown() {
```

```
  for (int i = 0; i < 255; i++) {
    tft.setBacklight(254 - i);
    delay(1);
  }
}

void clearLine(uint16_t x, uint16_t y) {
  tft.setCursor(x, y);
  tft.print("                                                ");
  Serial.println("LINE CLEARED");
}

void drawLine() {
  tft.drawHorizontalLine(20, 180, 440, Color::DarkOrange);
}

void displayName() {
  tft.setFont(Fonts::GeosansLight16);
  tft.setCursor(14, 14);
  tft.print("Autor:");
  tft.setCursor(14, 34);
  tft.print("Robert Doksa");

  // za kvacicu na slovu s
  tft.setCursor(86, 36);
  tft.setFont(Fonts::MINI4x6);
  tft.print("v");
  tft.setFont(Fonts::GeosansLight16);
}

void displayValues(float temperature, float pressure, int humidity,
float windSpeed) {
  char snum[50];
  bool clearFlag = false;
  const char deg = 167;

//TEMPERATURA PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setTextColor(Color::DarkOrange, Color::Gray5);
  tft.setCursor(28, 210);
  tft.print("Temperatura");

  tft.setFont(Fonts::GeosansLight34);

  /* u slucaju razlike broja znamenki trenutne temperature i prosle
  temperature, ili u slucaju aktivne zastavice clearFlag, ocisti se
  linija kako se nebi preklapali simboli */

  if(numDigits(temperature) != lastTemp || clearFlag) {
    clearLine(28, 250);
    clearFlag = false;
  }
  tft.setCursor(28, 250);
  if(temperature < 0 || temperature > 50) {
     clearFlag = true;
     tft.print("       ");
  }
```

```
  if(temperature>=0 && temperature < 10){
     sprintf(snum, "%.1f ", temperature);
     tft.print(snum);
  }
  else if(temperature>=10 && temperature <100) {
     sprintf(snum, "%.1f ", temperature);
     tft.print(snum);
  }
  else if(temperature>=100 && temperature <1000) {
     sprintf(snum, "%.1f  ", temperature);
     tft.print(snum);
  }
  else {
     sprintf(snum, "%.1f", temperature);
     tft.print(snum);
  }
  lastTemp = numDigits(temperature);
  //promjena fonta u svrhu ispisa simbola stupnja
  tft.setFont(Fonts::Squircle);
  tft.print(deg);
  tft.setFont(Fonts::GeosansLight34);
  tft.print("C ");

//TLAK PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setCursor(220, 210);
  tft.print("Tlak");
  tft.setCursor(175, 250);
  tft.setFont(Fonts::GeosansLight34);

  if(numDigits(pressure) != lastPressure) {
    clearLine(175, 250);
  }
  tft.setCursor(175, 250);
  if(pressure < 0 || pressure < 700) {
    clearLine(175, 250);
  }
  else {
    sprintf(snum, "%.2f hPa", pressure);
    tft.print(snum);
  }
  lastPressure = numDigits(pressure);

//VLAGA PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setCursor(380, 210);
  tft.print("Vlaznost");

  tft.setCursor(408, 212);
  tft.setFont(Fonts::MINI4x6);
  tft.print("v");

  tft.setCursor(380, 250);
  tft.setFont(Fonts::GeosansLight34);

  if(numDigits(humidity) != lastHumidity || humidity <= 0) {
    clearLine(380, 250);
```

```
  }
  tft.setCursor(380, 250);
  sprintf(snum, "%d%%", humidity);
  tft.print(snum);
  lastHumidity = numDigits(humidity);

//VJETAR PRINT
  tft.setFont(Fonts::GeosansLight20);
  tft.setCursor(230, 16);
  tft.print("Brzina vjetra");
  tft.drawHorizontalLine(200, 49, 155, Color::DarkOrange);
  tft.setFont(Fonts::GeosansLight38);

  tft.setCursor(230, 80);
  sprintf(snum, "%.1f m/s", windSpeed);
  tft.print(snum);
}
void fontTest() {
  char a = 0;
  for(int i = 80; i <190; i++){
    Serial.println(i);
    a = i;
    tft.fillScreen(Color::Black);
    tft.setFont(Fonts::Squircle);
    tft.setTextColor(Color::DarkOrange);
    tft.setCursor(220, 210);
    tft.print(a);
    delay(500);
  }
}

void setup() {
  ts.initializeDevice();
  Serial.begin(9600);
  Wire.begin();
  Wire.setClock(100000);

  while (!bmx280.begin()) {
    Serial.println(F("Error: BMP280 nije detektiran!"));
    delay(3000);
  }

  bmx280.setSampling(BMX280_MODE_NORMAL,
                     BMX280_SAMPLING_X16,
                     BMX280_SAMPLING_X16,
                     BMX280_SAMPLING_X16,
                     BMX280_FILTER_OFF,
                     BMX280_STANDBY_MS_500);

  // postavke za touchscreen
  ts.scaleX(4.5);
  ts.scaleY(3.4);
  ts.offsetY(5);

  tft.initializeDevice();
  tft.setBacklight(0);
  tft.setTextWrap(false);
```

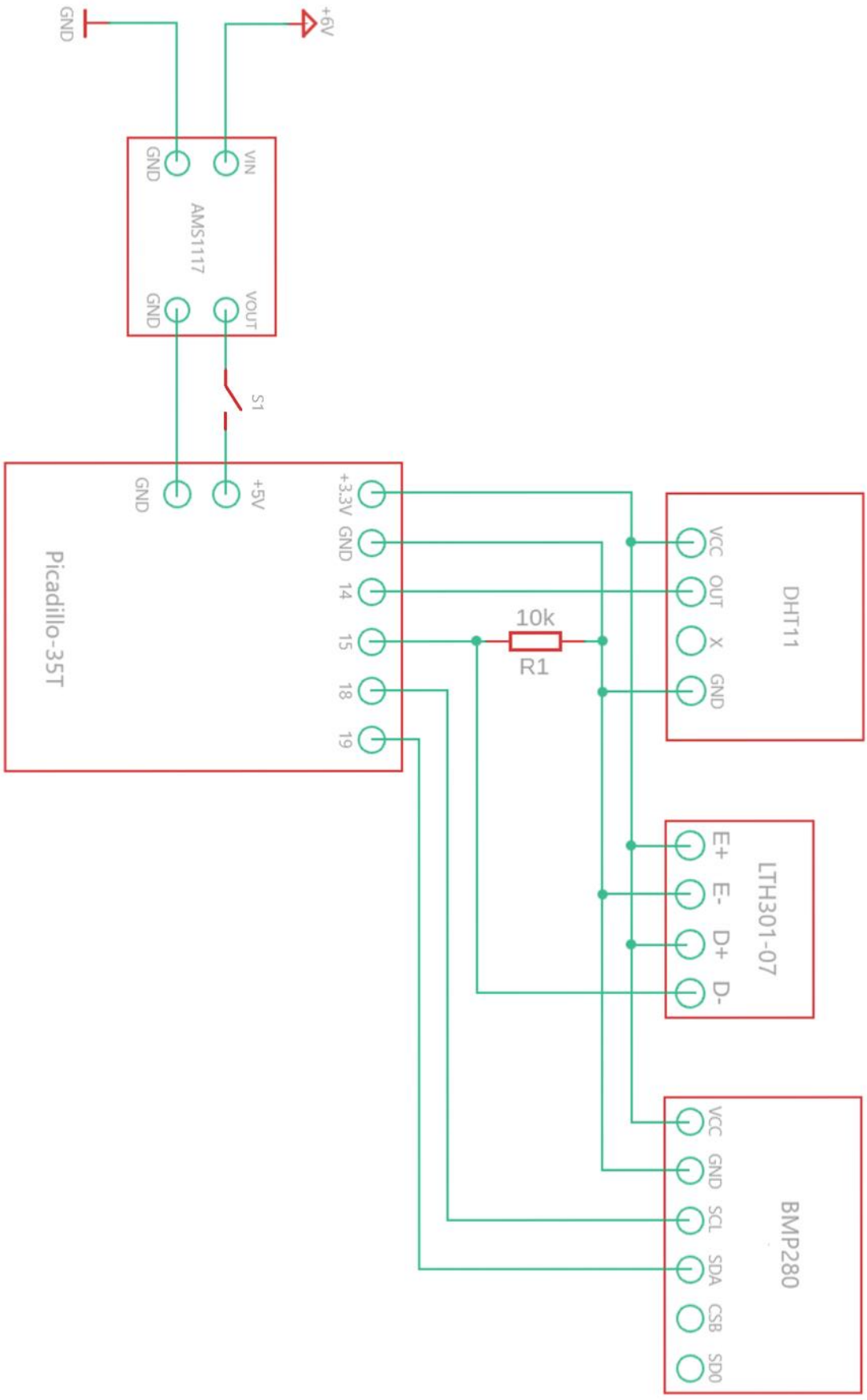
```
  tft.setRotation(3);
  tft.fillScreen(Color::Gray5);
  tft.setFont(Fonts::GeosansLight40);
  tft.setTextColor(Color::DarkOrange, Color::Gray5);

  pinMode(DHT11PIN, INPUT);
  displayName();
  drawLine();

  pinMode(ANEMOPIN, INPUT);
  startMillis = millis();
}

void loop() {
  currentMillis = millis();

  if (currentMillis - startMillis >= period)
  {
    float rpm = ((float)rot/24) * 60;

    Serial.print(rot);
    Serial.print(" impulses\t");

    Serial.print((float)rot / 24);
    Serial.print(" RPS\t");

    Serial.print(rpm);
    Serial.print(" RPM\t");

    float wind_kmph = rpm * 0.03 * 6;
    float wind_ms = wind_kmph / 3.6;
    Serial.print(wind_kmph);
    Serial.print(" km/h\t");

    Serial.print(wind_ms);
    Serial.println(" m/s\t");

    int chk = DHT11.read11(DHT11PIN);

    Serial.print("Humidity (%): ");
    Serial.println((float)DHT11.humidity, 2);

    float temperature = bmx280.readTemperature();
    Serial.print("Temperature  (C): ");
    Serial.println(temperature, 2);

    float pressure = bmx280.readPressure() / 100.0F;
    Serial.print("Pressure (hPa): ");
    Serial.println(pressure);

    displayValues(temperature, pressure, DHT11.humidity, wind_ms);
    if(initVal) {
      fadeUp();
      initVal = false;
    }
    Serial.println();
```

```
    rot = 0;
    startMillis = currentMillis;
  }

  if(digitalRead(ANEMOPIN) == HIGH && flag == false && init_flag ==
false) {
    rot++;
    flag = true;
    init_flag = false;
    //Serial.println(rot);
  }
  if(digitalRead(ANEMOPIN) == LOW) {
    flag = false;
    init_flag = false;
  }
  if(init_flag == true) {
    rot = 0;
    flag = true;
  }
}
```

Spisak komponenata:

- PICadillo-35T
- BMP280
- DHT11
- LTH301-07
- AMS1117
- 10 kΩ 0.25W otpornik
- 4x AA baterije
- KIP prekidač ON-OFF, mali

Električna shema:



Sveučilište
Sjever

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, Robert Dokša *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog *(obrisati nepotrebno)* rada pod naslovom Izrada meteorološke stanice prinjenom PIC 32 upravljanog sučelja za zaslon *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)

Robert Dokša

(vlastoručni potpis)

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, Robert Dokša *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog *(obrisati nepotrebno)* rada pod naslovom Izrada meteorološke stanice prinjenom PIC32 upravljanog sučelja za zaslon *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)

Robert Dokša

(vlastoručni potpis)