Tehnički fakultet Rijeka

Računalom podržana mjerenja

Seminarski rad

Izradili: Deni Klen, Ani Perušić, Mihael Petranović, Mateo Srića

Mentor: prof. dr. sc. Saša Vlahinić

Rijeka, veljača 2021

Sadržaj

Uvod

Tema seminarskog rada je izrada grafičkog sučelja u Python-u koje će prikazivati podatke o temperaturi i vlažnosti pomoću senzora spojenog na Arduino mikročip, kao i izrada data loggera, uređaja koji prikuplja podatke sa senzora i sprema ih za kasnije korištenje.

Razvojna okruženja i sklopovlje

Arduino platforma

Mikročipska ploča korištena u izradi projekta je Arduino Micro. Ona je jedna od Arduinovih mikročipovskih ploča koje su napravljene za početnike; jednostavna je za upotrebu te pisanje koda, a služi kao dobra polazna točka za upoznavanje i učenje elektronike. Zasnonavana je na Atmegi32U4 i razvijena zajedno s Adafruitom. Ima 20 digitalnih ulazno/izlaznih pinova (od kojih se 7 mogu koristiti kao PWM izlazi, a 12 kao analogni ulazi), kristalni oscilator od 16 MHz, mikro USB vezu, ICSP zaglavlje i gumb za resetiranje. Sadrži sve potrebno za podršku mikrokontrolera i jednostavno se spaja na računalo uz pomoć mikro USB kabela. Ugrađena USB komunikacija uklanja potrebu za sekundarnim procesorom. To omogućuje mikročipskoj ploči da se na povezanom računalu prikaže kao miš i tipkovnica, uz virtualni (CDC) serijski / COM priključak.

Programsku podršku za mikročip pisali smo u Arduino IDE-u. To je softver otvorenog koda koji olakšava pisanje programske podrške i prijenos na bilo koju Arduino ploču. Aktivni razvoj se odvija na GitHub-u.

Python

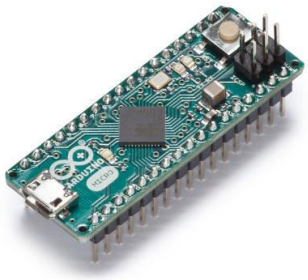
Za izradu grafičkog sučelja koristili smo programski jezik Python. Python je programski jezik visoke razine i opće namjene koji omogućava brzi rad i učinkovitu integraciju sustava. Njegove jezične konstrukcije i objektno orijentirani pristup imaju kao cilj pomoći programerima da napišu jasan i logičan kod bez obzira o veličini projekta. Python se dinamički upisuje i prikuplja smeće, stoga programer na mora misliti o alokaciji i brisanju memorije, podržava više paradigmi programiranja, uključujući strukturirano (posebno proceduralno), objektno

orijentirano i funkcionalno programiranje. Python se često opisuje kao jezik "s uključenim baterijama" zbog svoje sveobuhvatne standardne biblioteke koja uvelike olakšava programiranje.

Senzor

Senzor korišten za prikupljanje podataka o temperaturi i vlažnosti je DHT11. U njemu se nalazi senzor temperature – mali termistor zalemljen na pločicu te senzor vlage – mala tiskana pločica koja je nadolemljena na osnovnu. Što je više vlage u zraku, više vlage dolazi i na same vodove te je otpor među njima manji (ili počinje postojati kada se pojavljuje voda — to je razlog zašto senzor očitava vrijednosti od 20%, tek onda može očitati nekakav otpor). Osim navedene dvije komponente još nalazimo jedan integrirani krug koji analzira ulaze od prethodno navedenih senzora i komunicira s Arduinom.

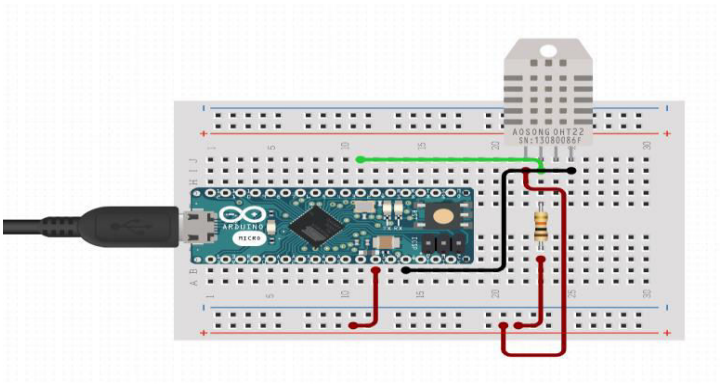
Slika 1. VMA311 DHT11 senzor



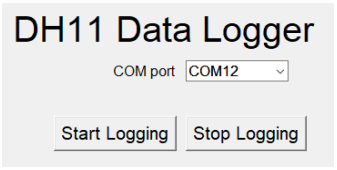
Slika 2. Arduino Micro

Opis rada

Prije otvaranja programa, potrebno je spojiti Arduino pločicu na računalo i senzor na Arduino pločicu. Senzor se na Arduino spaja na 5V, uzemljenje i analogni ulaz A0. Nakon pravilnog spajanja možemo otvoriti program. Potrebno je odabrati port na koji je Arduino spojen nakon čega se može pokrenuti praćenje podataka gumbom *Start Logging*. U svakom trenutku se može isto praćenje zaustaviti gumbom *Stop Logging*.

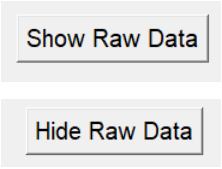


Slika 3. DHT22/11 senzor spojen na Arduino Micro



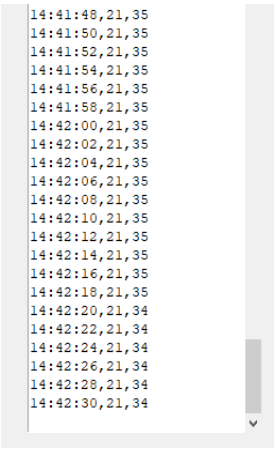
Slika 4. Prikaz zaglavlja prozora

U slučaju da želimo vidjeti podatke kako ih Arduino šalje, moguće je prikazati i sakriti tekstualni objekt sa ispisom najnovijih podataka. Neovisno o tome, uvijek su prisutni zadnje primljeni podaci.



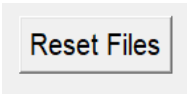
Slika 5. Gumb za pregled podataka

Slika 6. Zadnje dokumentirani podaci

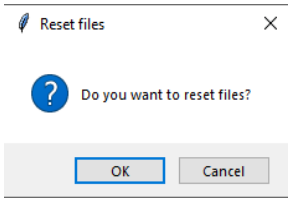


Slika 7. Tekstualni objekt sa podacima

Brisanje predhodno dokumentiranih podataka vrši se pritiskom na gumb *Reset Files* nakon čega se traži potvrda korisnika.

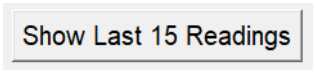


Slika 8. Gumb za brisanje podataka

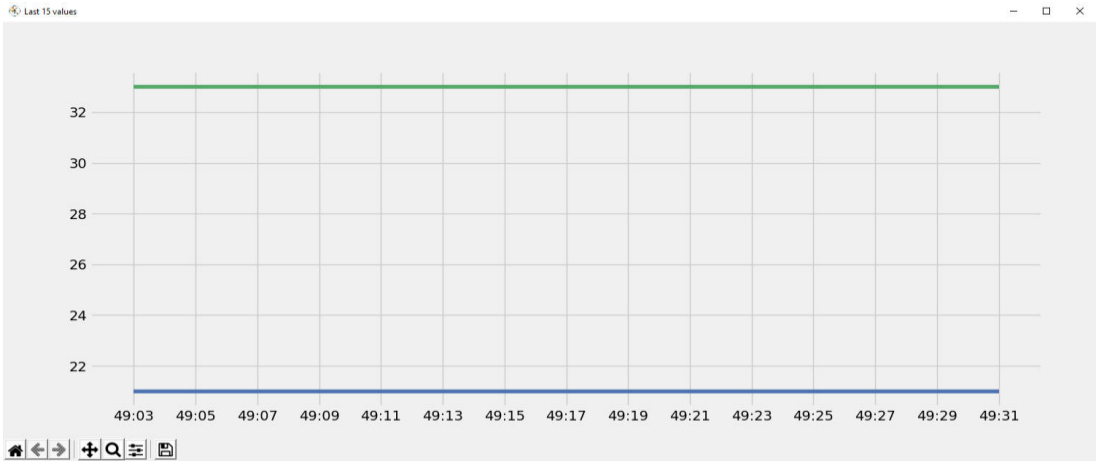


Slika 9. Potvrda brisanja

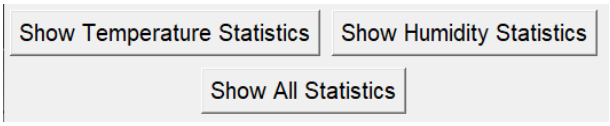
Za očitanje zadnjih petnaest vrijednosti temperature i vlage koristi se gumb *Show Last 15 readings*, dok se za sveukupnu statistiku o tim podacima koriste gumbi *Show Temperature Statistics, Show Humidity Statistics* i *Show All Statistics.*



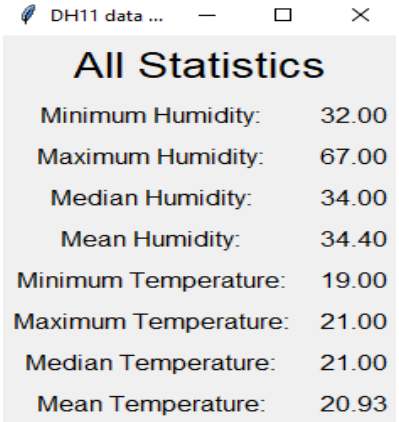
Slika 10. Gumb za najnovija očitanja



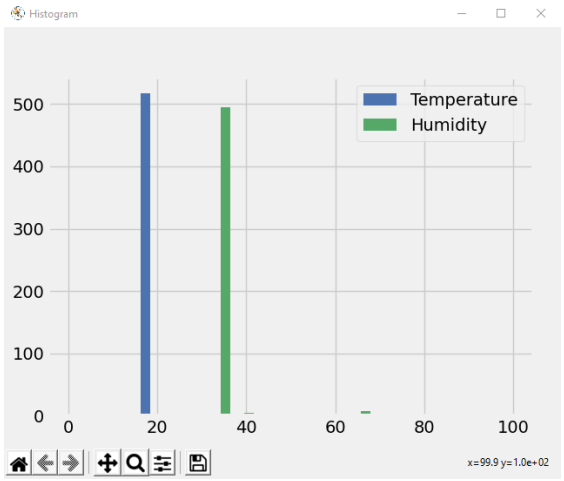
Slika 11. Graf najnovijih očitanja



Slika 12. Gumbi za prikaz statističkih podataka



Slika 13. Brojevni podaci



Slika 14. Histogram vrijednosti

Programska podrška

Praćenje podataka

Arduino pločica služi kao posrednik između senzora i Python koda. Programska podrška za Arduino uključuje blago izmjenjenu DHT biblioteku koja je obavljala čitanje vrijednosti, prebacivanje i spremanje u varijable te provjeru kontrolne sume. U glavnom kodu svake dvije sekunde izvršava se čitanje sa serijskog porta. Vrijednosti kraće od dvije sekunde mogle bi remetiti očitanja, no u ovom slučaju češća očitanja nisu potrebna. Nakon očitanja vrijednosti, temperatura i vlažnost se ispisuju na serijski izlaz odvojeni delimiterom ",". Programska podrška Arduina je jednostavna jer se sav posao odrađuje u programskoj podršci u Pythonu.

```
#include "dht.h"
#define dhtPin A0

dht DHT;
int freq = 2000;

void setup(){
  Serial.begin(9600);
}

void loop(){
    delay(freq);
    DHT.read11(dhtPin);
    Serial.print((int)DHT.temperature);
    Serial.print(",");
    Serial.println((int)DHT.humidity);
}
```

Slika 15. Arduinova programska podrška

Grafičko sučelje

Python-ova programska podrška se temelji na bibliotekama *tkinter* (izrada prozora i objekata), *serial* (čitanje sa serijskog porta), *threading* (lakše upravljanje procesima i memorijom), *time* i *Matplotlib* (izračun i prikaz raznih grafova) te *NumPy* i *pandas* (manipulacija podataka). Koriste se tri različite datoteke u koje se spremaju i iz kojih se čitaju podaci. Sve tri nalaze se u *data* folderu. Prva sadrži "vrijeme,temperaturu,vlažnost" gdje je svaki novi red spremljen u takvom formatu. Druga i treća datoteka sadrže odvojeno vrijednosti temperature i vlažnosti u .cvs datotekama.

Inicijalizacija serijskog porta kao i dretve koja obrađuje serijski ulaz, dakle main funkciju, odrađuje se pritiskom na *Start Logging* gumb. Za aktivaciju ulaza označujemo jedan od

ponuđenih iz kombiniranog okvira. Istom logikom, pritiskom Stop Logging gumba zaustavlja se dretva, odnosno čitanje podataka.

```
# Serial --------------------
def serialPorts():
    return [p.device for p in serial.tools.list_ports.comports()]
    pass

def serialInit():
    global serialPort
    serialPort = serial.Serial()
    serialPort.baudrate = 9600
    serialPort.port = COMPortCombobox.get()
    serialPort.open()
    main()

def startSerialThread():
    global serialThreadBoolean
    if not serialThreadBoolean and not (COMPortCombobox.get() == 'None' or COMPortCombobox.get() == ''):
        serialThreadBoolean = True
        serialThread = Thread(target=serialInit, daemon=True)
        serialThread.start()

def stopSerialThread():
    global serialThreadBoolean
    serialThreadBoolean = False
```

Slika 16. Funkcije za čitanje sa serijskog porta, kao i za dretve

Main funkcija sadrži *serialPortToLine()* funkciju koja čita podatke u bitovnom zapisu i prebacuje ih u podatke tipa string. Ukoliko string nije prazan u njega dodaje trenutno vrijeme. Tom varijablom ažurira vrijednosti na grafičkom sučelju kao i vrijednosti u datotekama. U datoteke koje sadrže pojedinačne vrijednosti temperature i vlažnosti podaci se dodavaju na kraj dok se u datoteku sa vremenima linije dodaju na početak radi lakšeg iščitavanja pri izradi grafova.

```
def main():
    global serialThreadBoolean
    global line
    if serialThreadBoolean:
        serialPortToLine()
        if len(line) > 1:
            addTimeToLine()
            updateValues()
            linePrepender(line)

        loopMain()
```

Slika 17. main() funkcija

```
def updateValues():
    tempHumTime = line.split(",")
    timeValueLabel.config(text=tempHumTime[0])

    temperatureValueLabel.config(text=tempHumTime[1] + " °C")
    addToTempFile(tempHumTime[1])

    tempHumTime[2] = tempHumTime[2][:len(tempHumTime[2])-1]
    humidityValueLabel.config(text=tempHumTime[2] + " %")
    addToHumFile(tempHumTime[2])

    rawText.insert("end", line)
    rawText.see("end")
```

Slika 18. Ažuriranje vrijednosti

```
def addToTempFile(string):
    file = open(tempFileName, "a")
    file.write(string + ",")
    file.close()
```

```
def addToHumFile(string):
    file = open(humFileName, "a")
    file.write(string + ",")
    file.close()
```

```
def linePrepender(line):
    global textFileLines
    textFileLines += 1
    with open(timeTempHumFile, 'r+') as file:
        content = file.read()
        file.seek(0, 0)
        file.write(line.rstrip('\r\n') + '\n' + content)
        file.close()
```

```
timeTempHumFile = "data/timeTempHum.txt"
tempFileName = "data/temp.cvs"
humFileName = "data/hum.cvs"
```

Slike 19.-22. Dodavanje vrijednosti u datoteku

U slučaju da želimo izbrisati sve zapise u datotekama, poziva se funkcija *resetFiles()* koja otvara sve datoteke sa pristupom *write*, koji briše sve i dodaje željeni ispis na kraj, ali u ovom slučaju nam ispis nije potreban pa se ista datoteka odmah zatvara.

```
def resetFiles():
    if messagebox.askokcancel("Reset files", "Do you want to reset files?"):
        file = open(timeTempHumFile, "w")
        file.close()
        file = open(tempFileName, "w")
        file.close()
        file = open(humFileName, "w")
        file.close()
```

Slika 23. Brisanje podataka iz datoteka

Prikaz statističkih podataka je uvelike olakšan bibliotekom *NumPy* gdje se pritiskom na gumb *Show Temperature Statistics* poziva funkcija *tempStatistics()* koja poziva *tempStatsWindow()* i *tempHistogram()* funkcije. *tempStatsWindow()* uzima podatke iz datoteke koja sadržava temperature i sprema ih u niz vrijednosti funkcijom *getTempFile()*. Nakon toga stvara novi prozor sa svim elementima i kao vrijednosti stavlja minimum, maksimum, medijan i srednju vrijednost. *tempHistogram()* koristi funkcije od *matplotlib* biblioteke. Dohvaćaju se vrijednosti iz *temp.cvs* datoteke, postavljaju se ograničenja u *bins* varijabli i stvara se histogram pozivom *plt.hist()*. Isti proces se odvija za vlažnost te kombinaciju vlažnosti i temperature.

```
temps = getTempFile()
```

```
def getTempFile():
    file = open(tempFileName, 'r')
    content = file.read()
    file.close()
    content = content[:len(content)-1]
    content = content.split(',')
    return np.array(content).astype(np.float64)
```

Slika 24. i 25. Dohvaćanje vrijednosti iz datoteke

```
minValLabel = ttk.Label(tStatsWin, text="{:.2f}".format(temps.min()))
minValLabel.grid(row=1, column=1)
maxValLabel = ttk.Label(tStatsWin, text="{:.2f}".format(temps.max()))
maxValLabel.grid(row=2, column=1)
medValLabel = ttk.Label(tStatsWin, text="{:.2f}".format(np.median(temps)))
medValLabel.grid(row=3, column=1)
meanValLabel = ttk.Label(tStatsWin, text="{:.2f}".format(temps.mean()))
meanValLabel.grid(row=4, column=1)
```

Slika 26. Postavljanje vrijednosti u prozoru

```
def tempHistogram():
    style.use('seaborn-deep')
    temps = getTempFile()
    bins = np.linspace(0, 50, 20)

    plt.figure("Temperature Histogram")
    plt.hist(temps, bins)
    plt.show()
```

Slika 27. Stvaranje histograma

U slučaju da želimo vidjeti samo zadnjih 15 očitanja u grafu poziva se funkcija *graphSecondsInit()* koja inicijalizira prozor u kojemu se graf nalazi. Potrebno je stvaranje subplota jer imamo dvije različite vrijednosti – temperaturu i vlažnost. Zatim se poziva funkcija *showSecondsGraph()* koja svake sekunde ažurira graf vrijednostima iz varijable *figureSeconds* pozivom na funkciju *animateSeconds()*. *animateSeconds()* čita prvih petnaest ili manje vrijednosti iz datoteke i sprema podatke u odvojene nizove iz kojih zatim iznova stvara graf svake sekunde.

```
def graphSecondsInit():
    global figureSeconds
    global axisSeconds
    style.use('fivethirtyeight')
    figureSeconds = plt.figure("Last 15 values", figsize=(15, 6))
    axisSeconds = figureSeconds.add_subplot()
    axisSeconds.set_xlabel('Time')
    axisSeconds.set_ylabel('Values')
    showSecondsGraph()
```

Slika 28. Inicijalizacija grafa

```
def showSecondsGraph():
    animationSeconds = animation.FuncAnimation(figureSeconds, animateSeconds, interval=1000)
    plt.show()
```

Slika 29. Prikaz grafa

```
def animateSeconds(i):
    linesS = []
    if textFileLines >= numberOfGraphPoints:
        head = []
        with open(timeTempHumFile) as myfile:
            head = [next(myfile) for x in range(numberOfGraphPoints)]
        head.reverse()
        linesS = listToString(head).split('\n')
    else:
        head = []
        for line in reversed(list(open(timeTempHumFile))):
            head += line.rstrip() + "\n"
        linesS = listToString(head).split('\n')

    timesS = []
    temperaturesS = []
    humiditiesS = []
    for oneLine in linesS:
        if len(oneLine) > 1:
            tTimeS, tTempS, tHumS = oneLine.split(',')
            tTimeS = tTimeS[3:len(tTimeS)]
            timesS.append(tTimeS)
            temperaturesS.append(int(tTempS))
            humiditiesS.append(int(tHumS))
        axisSeconds.clear()
        axisSeconds.plot(timesS, temperaturesS, label='Temperature')
        axisSeconds.plot(timesS, humiditiesS, label='Humidity')
```

Slika 30. Upisivanje podataka u graf

Sažetak

U radu se upoznajemo načinom na koji smo realizirali mjerenje vrijednosti temperature i vlažnosti, koristeći se Arduinom i DHT11 senzorom, te grafičkim sučeljem napisanim u Python jeziku za vizualno prikazivanje istih. Dotičemo se razvojnog okruženja, sklopovlja i programske podrške korištene pri izradi projekta te detaljnije objašnjavamo neke dijelove koda i funkcije potrebne za ispravan rad projekta.

Literatura

https://store.arduino.cc/arduino-micro

https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

https://www.arduino.cc/en/software

https://github.com/arduino/Arduino

https://docs.python.org/3/library/tkinter.html

https://matplotlib.org/stable/index.html

https://numpy.org

https://docs.python.org/3/library/time.html