UniverzitetSingidunum u Nišu

Tehnički fakultet

Elektrotehnika i računarstvo

Izveštaj o radu

Internet stvari

Pripremili:

Lazar Kostić, Strahinja Jovanović, Borivoje Marković, Natalija Nikolić

Zadatak:

Potrebno je realizovati IoT sistem za akviziciju signala sa senzora i njihovo skladištenje uz pomoć Arduino mikrokontrolera i Raspberry Pi računara. Sistem se sastoji od senzora temperature LM35, ultrazvučnog senzora rastojanja i fototpornika koji su povezani na Arduino mikrokontroler. Pored ovih senzora na Arduino kontroler je povezan i LCD displej. Arduino je povezan na Raspberry Pi računar putem asinhrone serijske veze. Na Rapsberry Pi su pored Arduino kontrolera povezane i dve diode (jedna crvena i jedna zelena dioda).

Arduino kontroler treba da šalje podatke o izmerenoj temperature, razdaljini i osvetljenosti Raspberry Pi računaru na svakih 0.5 sekundi. Rapsberry Pi računar treba da skladišti primljene podatke i da izračunava srednje vrednosti u vremenskim intervalima koje korisnik konfiguriše putem e-mail poruke. Raspberry Pi računar treba da šalje e-mail poruke kojima korisnika obaveštava o srednjim vrednostima izmerenih veličina u vremenskim intervalima od interesa. Posebna e-mail poruka treba da se šalje za svaku od tri merene veličine. Poruke za odgovarajuće veličine treba da imaju Subject: Temperatura, Razdaljina i Osvetljenost. U telu poruke treba da u zavisnosti od merene veličine stoji informacija o srednjoj vrednosti izmerene vrednosti u proteklom periodu. Merena temperature, razdaljina i osvetljenost treba da u porukama budu iskazane u stepenima Celzijusa, centimetrima i luxevima.

Prilikom svake nove promene konfiguracije, zelena LED diode na Raspberry Pi računaru treba da zasvetli u trajanju od 40 sekundi i da se potom ugasi. Takođe na LCD ekranu koji je prikačen na Arduino mikrokontroler treba u prvoj (gornjoj) liniji da bude ispisano “Poslednja konf.”, a u drugoj (donjoj) vreme poslednje konfiguracije u formatu hh:mm:ss, gde hh oynačava časove, mm minute a ss sekunde.

Potrebni materijal:

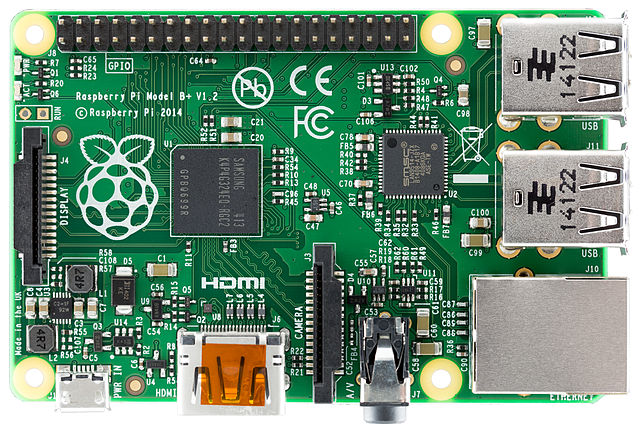
* Arduino Uno RV3
* Raspberry Pi 3
* Test ploča
* Senzor temperature (LM 35)
* Ultrazvučni senzor daljine
* Fotootpornik (photocell)
* 1 x Otpornik od 10kΩ , 2 x 220Ω
* Displej 2x16
* 2 led diode (1 x crvena i 1 x zelena)
* Žice za povezivanje

Arduino UNO R3

Hardverski Arduino UNO je kontroler baziran na ATMEGA328 mikrokontroleru koji radina 16 MHz. Na ploči se nalaze 14 digitalnih ulaza/izlazai 6 analognih ulaza. Spajanje na računar vrši se putem USB kabla. Napajanje je moguće zasebnim napajanjem od 7-12V ili putem računara (preko USB kabla). Arduino UNO se od svojih prethodnih ploča razlikuje u tome što ne koriste FTDI USB serijski čip drajvera, već umesto toga on ima Atmega16U2 (Atmega8U2 verzije R2) programiran kao USB serijski konverter.

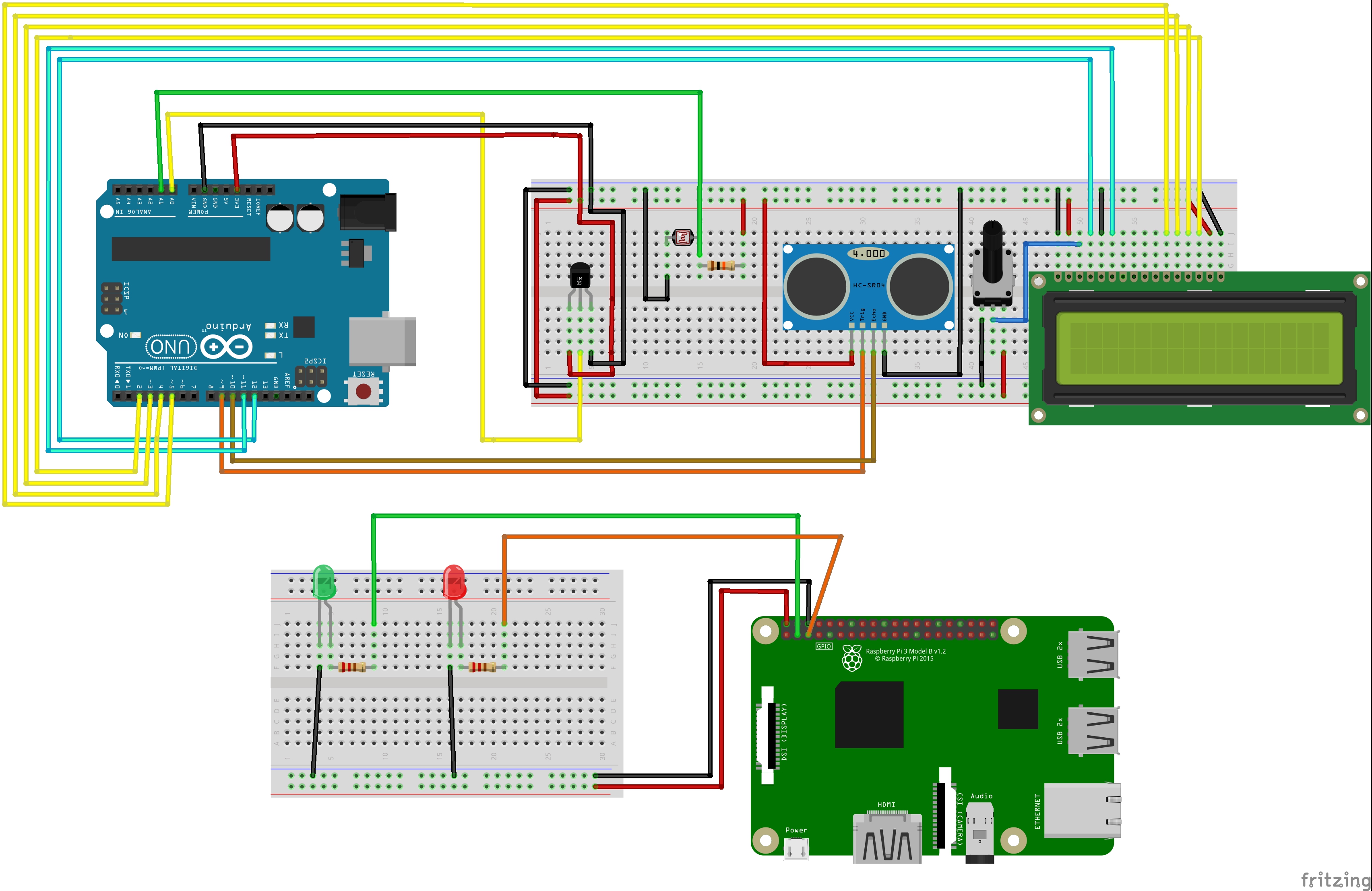
**Rapsberry Pi**

Raspberry Pi je [single-board](https://sr.wikipedia.org/w/index.php?title=Single-board_computer&action=edit&redlink=1) kompjuter veličine [kreditne kartice](https://sr.wikipedia.org/w/index.php?title=Credit_card&action=edit&redlink=1) razvijen u [Ujedinjenom Kraljevstvu(Engleska)](https://sr.wikipedia.org/wiki/%D0%A3%D1%98%D0%B5%D0%B4%D0%B8%D1%9A%D0%B5%D0%BD%D0%BE_%D0%9A%D1%80%D0%B0%D1%99%D0%B5%D0%B2%D1%81%D1%82%D0%B2%D0%BE) od strane [Fondacije Raspberry Pi](https://sr.wikipedia.org/w/index.php?title=Raspberry_Pi_Foundation&action=edit&redlink=1) sa namerom da promoviše i nauči osnove [kompjuterske nauke](https://sr.wikipedia.org/w/index.php?title=Computer_science&action=edit&redlink=1) u školama i [razvojnim zemljama](https://sr.wikipedia.org/w/index.php?title=Developing_countries&action=edit&redlink=1). Orginalni Raspberry Pi i Raspberry Pi 2 modeli su proizvedeni u više različitih konfiguracija preko licenciranih sporazuma za proizvodnju sa [Newark element14](https://sr.wikipedia.org/w/index.php?title=Newark_element14&action=edit&redlink=1) ([Premier Farnell](https://sr.wikipedia.org/w/index.php?title=Premier_Farnell&action=edit&redlink=1)),[RS Components](https://sr.wikipedia.org/w/index.php?title=RS_Components&action=edit&redlink=1) i Egoman. Hardver je isti kod svih proizvođača. Nekoliko generacija Raspberry Pi modela je već izašlo na tržište. Prva generacija ( Pi 1) izbačena je u februaru 2012. kao osnovni model A i specifičniji model B. Modeli A+ i B+ izašli su godinu dana kasnije. Raspberry Pi 2 model B izbačen je u februaru 2015.,a Raspberry Pi 3 model B u februaru 2016. Vrednost ovih ploča je između 20 i 35 US$. Pi Zero sa manjom štampanom pločom i ograničenim IO (GPIO) mogućnostima izbačena je u novembru 2015. u vrednosti od 5 US$. Svi modeli imaju Broadcom sistem čip koji podrazumeva ARM kompatibilni CPU i ugrađeni čip GPU (a VideoCore IV). CPU dostiže brzinu od 700 MHz do 1.2 GHz za Pi 3, sa memorijom ploče u opsegu od 256 MB do 1 GB RAM. SD kartice se koriste za čuvanje operativnog sistema i programske memorije u ili SDHC ili MicroSDHC veličinama. Većina ploča ima između jednog i četiri slota, HDMI i video izlaz, kao i 3.5 mm audio džek. Low-level izlaz snabdeven je brojem GPIO pinova koji podržavaju zajedničke protokole kao sto je I2C. Neki modeli imaju RJ45 Ethernet port a Pi 3 ima na ploči WiFi 802.11n i Bluetooth. Pronalazači Raspberry-ja daju Debian i Arch Linux ARM (Linux distribucije) za besplatno preuzimanje i promovišu Python kao glavni programski jezik, sa podrškom za BBC BASIC i podrškom (za RISC OS), C (programski jezik), C++, Java, Perl, Ruby, Squeak i jos mnogo drugih.U Februaru 2016, Raspberry Pi osnivači su izjavili da su prodali preko 8 miliona uredjaja, i time postigli da budu najbolji prodavci personalnih računara u UK, pobedivši Amstrad PCW.



Izgled Raspberry Pi 3 uredjaja

Šema:



Kod:

ARDUINO KOD

#include <LiquidCrystal.h>

const int rs = 12;

const int en = 11;

const int d4 = 5;

const int d5 = 4;

const int d6 = 3;

const int d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

int trig = 9;

int echo = 10;

void setup() {

analogReference(INTERNAL);

Serial.begin(9600);

pinMode(trig, OUTPUT);

pinMode(echo, INPUT);

lcd.begin(16, 2);

lcd.clear();

}

void loop() {

razdaljina();

temperatura();

svetlost();

delay(500);

lcd.setCursor(1, 0);

lcd.print("Poslednja konf.");

if (Serial.available()){

delay(100);

while (Serial.available() > 0){

char vreme = Serial.read();

lcd.setCursor(0,1);

lcd.write(vreme);

}

}

}

float temperatura(){

float temp;

float reading;

int tempPin = A1;

reading = analogRead(tempPin);

temp = reading/9.31;

Serial.println(temp);

}

int svetlost(){

int svetlostPin = A0;

int svetlostCitanje;

svetlostCitanje = analogRead(svetlostPin);

Serial.println(svetlostCitanje);

}

int razdaljina(){

long trajanje;

int razdaljinaCM;

digitalWrite(trig, LOW);

delayMicroseconds(2);

digitalWrite(trig, HIGH);

delayMicroseconds(10);

digitalWrite(trig, LOW);

trajanje = pulseIn(echo, HIGH);

razdaljinaCM = trajanje\*0.034/2;

Serial.println(razdaljinaCM);

}

Raspberry Pi 3 kod:

##Biblioeteke

import RPi.GPIO as GPIO

import serial

import http.client

import urllib

import string

import imaplib

import smtplib

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

import time

import email

from datetime import datetime

import requests

from email.parser import Parser

#Serijska komunikacija

ser = serial.Serial('/dev/ttyACM0',9600)

#Promenljive za prijem podataka

temperatura = "'Temperatura"

razdaljina = "'Razdaljina"

osvetljenost = "'Osvetljenost"

#definisanje pinova

LEDCrvena = 3

LEDZelena = 4

sleep = 60

#rad sa diodama

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(LEDCrvena, GPIO.OUT)

GPIO.setup(LEDCrvena, GPIO.LOW) #pri pokretanju programa diode ugasiti

GPIO.setup(LEDZelena, GPIO.OUT)

GPIO.setup(LEDZelena, GPIO.LOW) #pri pokretanju programa diode ugasiti

#rad sa email adresom

mail = imaplib.IMAP4\_SSL('imap.gmail.com')

mail.login("internetstvarinis@gmail.com", "singi123")

#inicijalno postavljanje promenljivih na 0

podaci = 0

#vremena za slanje izvestaja

vremeTemperature = 0 #t2

vremeRastojanja = 0 #d2

vremeOsvetljenosti = 0 #l2

#vrednosti podataka ako se mail salje posebno za svaku vrednost

temperaturaPosebno = 0 #srt

rastojanjePosebno = 0 #srd

osvetljenostPosebno= 0 #srf

# vrednosti podataka ukoliko se podaci salju na zahtev konfiguracije

temperaturaKonfig = 0 #srx

rastojanjeKonfig = 0 #sry

osvetljenostKonfig = 0 #srz

# pristizanje podataka u niz

prviPodatak = 0 #x

drugiPodatak= 0 #y

treciPodatak= 0 #z

#brojac

brojac = 0

#timeri

timerTemp = 0

timerRast = 0

timerOsvet = 0

#niz S

s = []

def konfiguracija():

mail.list()

mail.select('inbox')

status, response = mail.search(None,'(SUBJECT "Konfiguracija" UNSEEN)')

#Broj poruka u mailu

brojPoruka = (response[0].split())

i=len(response[0].split())

#niz sa primljenim podacima

podaci=[]

#ako u nizu ima nesto, upaliti crvenu diodu

if i>0:

GPIO.output(LEDCrvena, GPIO.HIGH)

#kada se upali dioda, proveriti koje je vreme("koliko je sati")

vreme = datetime.time.now().strftime("%HH:%MM:%SS")

#ispisivanje vremena konfiguracije u formatu "%HH:%MM:%SS"

#slanje ispisanog vremena na lcd

ser.write(vreme.encode)

#zatim ugasiti diodu

GPIO.output(LEDCrvena, GPIO.LOW)

# za svaku poruku u inboxu

for m\_id in brojPoruka:

#prepoznaj svaku "liniju poruke"

retcode, response = mail.fetch(m\_id, '(UID BODY[TEXT])')

podaci.append(response[0][1].decode("utf-8"))

str1 = ''.join(map(str,podaci))

#regex za provere sadrzaja poruke

rx = re.compile(r'''^(?P<key>\w+):\s\*(?P<value>.+)$''', re.MULTILINE)

#cmds za svaku vrednost datog kljuca u poruci pronalazi str1 koji je

#mapiranje vrednosti podataka koji se nalaze u poruci

cmds = {m.group('key'): m.group('value') for m in rx.finditer(str1)}

#izvlacenje vrednosti iz maila

O1 = cmds['Osvetljenost'] # naslov

vremeOsvetljenosti = int(re.search(r"\d+",O1).group()) # prva rec

O3 = str(re.search(r"\w+",O1).group()) # druga rec

T1 =cmds['Temperatura'] #naslov

vremeTemperature= int(re.search(r"\d+",T1).group()) #prva rec

T3 = str(re.search(r"\w+",T1).group()) #druga rec

U1 = cmds['Razdaljina'] #naslov

vremeRastojanja = int(re.search(r"\d+",U1).group()) #prva rec

U3 = str(re.search(r"\w+",U1).group()) #druga rec

def zahtev():

mail.list()

mail.select('inbox')

result, podaci = mail.uid('search', None, '(SUBJECT "Posalji" UNSEEN)')

#uzima samo jedan podatak sa indexom 1

i = len(podaci[0].split())

for x in range(i):

latest\_email\_uid = podaci[0].split()[x]

#uzimanje maila

result, email\_data = mail.uid('fetch', latest\_email\_uid, '(RFC822)')

raw\_email = email\_data[0][1]

raw\_email\_string = raw\_email.decode('utf-8')

email\_message = email.message\_from\_string(raw\_email\_string)

#prolazak kroz email

for part in email\_message.walk():

if part.get\_content\_type() == "text/plain":

body = part.get\_payload(decode=True)

#body email

bodystr=str(body)

split = bodystr.split('\\r\\n')

prvi=split[0]

drugi = prvi.split('b')

treci = drugi[1]

if treci==temperatura:

izvestajTemperaturaK()

print("Poslat Izvestaj")

if treci==razdaljina:

izvestajRastojanjeK()

if treci==osvetljenost:

izvestajOsvetljenostK()

else:

continue

def izvestajTemperaturaPosebno():

msg = MIMEMultipart()

msg['Subject'] = 'Temperatura '

sadrzajPoruke = "Prosecna temperatura u poslednjih "+str(vremeTemperature) +"minuta" + "je " + str(temperaturaPosebno) + "stepeni celzijusa"

textPart = MIMEText(sadrzajPoruke, 'plain')

msg.attach(textPart)

s = smtplib.SMTP('smtp.gmail.com', 587)

#slanjePoruke = MIMEText(sadrzajPoruke,'sadrzaj')

s.ehlo()

s.starttls()

s.ehlo()

s.login("internetstvarinis@gmail.com", "singi123")

s.sendmail("internetstvarinis@gmail.com", "internetstvarinis@gmail.com", msg.as\_string())

s.quit()

def izvestajTemperaturaK():

msg = MIMEMultipart()

msg['Subject'] = 'Temperatura '

sadrzajPoruke = "Prosecna temperatura u poslednjih "+str(vremeTemperature) +"minuta" + "je " + str(temperaturaKonfig) + "stepeni celzijusa"

textPart = MIMEText(sadrzajPoruke, 'plain')

msg.attach(textPart)

s = smtplib.SMTP('smtp.gmail.com', 587)

#slanjePoruke = MIMEText(sadrzajPoruke,'sadrzaj')

s.ehlo()

s.starttls()

s.ehlo()

s.login("internetstvarinis@gmail.com", "singi123")

s.sendmail("internetstvarinis@gmail.com", "internetstvarinis@gmail.com", msg.as\_string())

s.quit()

def izvestajRastojanje():

msg = MIMEMultipart()

msg['Subject'] = 'Razdaljina '

sadrzajPoruke = "Prosecna razdaljina u poslednjih "+str(vremeRastojanja) +" minuta" + "je "+str(rastojanjePosebno) + "cm"

textPart = MIMEText(sadrzajPoruke, 'plain')

msg.attach(textPart)

s = smtplib.SMTP('smtp.gmail.com', 587)

#slanjePoruke = MIMEText(sadrzajPoruke,'sadrzaj')

s.ehlo()

s.starttls()

s.ehlo()

s.login("internetstvarinis@gmail.com", "singi123")

s.sendmail("internetstvarinis@gmail.com", "internetstvarinis@gmail.com", msg.as\_string())

s.quit()

def izvestajRastojanjeK():

msg = MIMEMultipart()

msg['Subject'] = 'Razdaljina '

sadrzajPoruke = "Prosecna razdaljina u poslednjih "+str(vremeRastojanja) +" minuta" + "je "+str(rastojanjeKonfig) + "cm"

textPart = MIMEText(sadrzajPoruke, 'plain')

msg.attach(textPart)

s = smtplib.SMTP('smtp.gmail.com', 587)

#slanjePoruke = MIMEText(sadrzajPoruke,'sadrzaj')

s.ehlo()

s.starttls()

s.ehlo()

s.login("internetstvarinis@gmail.com", "singi123")

s.sendmail("internetstvarinis@gmail.com", "internetstvarinis@gmail.com", msg.as\_string())

s.quit()

def izvestajOsvetljenost():

msg = MIMEMultipart()

msg['Subject'] = 'Osvetljenost '

sadrzajPoruke = "Prosecna osvetljenost u poslednjih "+str(vremeOsvetljenosti) +" minuta" + "je "+str(osvetljenostPosebno) + "lux"

textPart = MIMEText(sadrzajPoruke, 'plain')

msg.attach(textPart)

s = smtplib.SMTP('smtp.gmail.com', 587)

#slanjePoruke = MIMEText(sadrzajPoruke,'sadrzaj')

s.ehlo()

s.starttls()

s.ehlo()

s.login("internetstvarinis@gmail.com", "singi123")

s.sendmail("internetstvarinis@gmail.com", "internetstvarinis@gmail.com", msg.as\_string())

s.quit()

def izvestajOsvetljenostK():

msg = MIMEMultipart()

msg['Subject'] = 'Osvetljenost '

sadrzajPoruke = "Prosecna razdaljina u poslednjih "+str(vremeOsvetljenosti) +" minuta" + "je "+str(osvetljenostKonfig) + "lux"

textPart = MIMEText(sadrzajPoruke, 'plain')

msg.attach(textPart)

s = smtplib.SMTP('smtp.gmail.com', 587)

#slanjePoruke = MIMEText(sadrzajPoruke,'sadrzaj')

s.ehlo()

s.starttls()

s.ehlo()

s.login("internetstvarinis@gmail.com", "singi123")

s.sendmail("internetstvarinis@gmail.com", "internetstvarinis@gmail.com", msg.as\_string())

s.quit()

def notifikacija():

#kada rastojanje padne ispod 5 cm

#pali se zelena dioda

GPIO.output(LEDZelena, GPIO.HIGH)

#salje se notifikacija

requests.post("https://maker.ifttt.com/trigger/Udaljenost/with/key/ibaAV5QZYrwkD8a06SgteVF6h\_8hELBIylhPzy1rnnO")

#provera da li je rastojanje ispod 5

print ("notifikacija")

#Potrbno je staviti minut, zbog potreba testiranja postavljeno 10

#time.sleep(10)

time.sleep(10)

#ugasiti diodu

GPIO.output(LEDZelena, GPIO.LOW)

def citanjePodataka(ser):

return int(ser.readline())

while True:

mail.list()

mail.select('inbox')

status, response = mail.search(None,'(SUBJECT "Konfiguracija" UNSEEN)')

status1, response1 = mail.search(None,'(SUBJECT "Posalji" UNSEEN)')

#broj mailova

msg\_num = (response[0].split())

msg\_num2 = (response1[0].split())

msg\_num2

podaci1=[]

podaci2=[]

i=len(response[0].split())

#ako ima podataka

if i>0:

GPIO.output(LEDCrvena, GPIO.HIGH)

time.sleep(10)

GPIO.output(LEDZelena, GPIO.LOW)

i=0

for m\_id in brojPoruka:

#prepoznaj svaku "liniju poruke"

retcode, response = mail.fetch(m\_id, '(UID BODY[TEXT])')

podaci1.append(response[0][1].decode("utf-8"))

str1 = ''.join(map(str,podaci1))

#regex za provere sadrzaja poruke

rx = re.compile(r'''^(?P<key>\w+):\s\*(?P<value>.+)$''', re.MULTILINE)

#cmds za svaku vrednost datog kljuca u poruci pronalazi str1 koji je

#mapiranje vrednosti podataka koji se nalaze u poruci

cmds = {m.group('key'): m.group('value') for m in rx.finditer(str1)}

#izvlacenje vrednosti iz maila

O1 = cmds['Osvetljenost'] # naslov

vremeOsvetljenosti = int(re.search(r"\d+",O1).group()) # prva rec

O3 = str(re.search(r"\w+",O1).group()) # druga rec

T1 =cmds['Temperatura'] #naslov

vremeTemperature= int(re.search(r"\d+",T1).group()) #prva rec

T3 = str(re.search(r"\w+",T1).group()) #druga rec

U1 = cmds['Razdaljina'] #naslov

vremeRastojanja = int(re.search(r"\d+",U1).group()) #prva rec

U3 = str(re.search(r"\w+",U1).group()) #druga rec

for m\_id in msg\_num:

mail.store(m\_id, '+FLAGS', '\seen')

vreme = datetime.now()

v = ("{:02d}:{:02d}:{:02d}".format(vreme.hour,vreme.minute,vreme.second))

ser.write(v.encode())

mail.expunge()

podaci = ser.readline()

if podaci:

s.append(podaci)

#stizu 3 podatka

#kada se u niz s smeste 3 podatka

#onda se podaci dodaju ponovo na vec postojece kako bi se izracunala srednja vrednost

if len(s)==3:

prviPodatak = int(s[0])

drugiPodatak = float(s[1])

treciPodatak = int(s[2])

#stampanje samo radi provere, nije neophodno

print(prviPodatak)

print(drugiPodatak)

print(treciPodatak)

if x<5:

notifikacija()

#racunanje srednje vrednosti

ppSV += prviPodatak

dpSV += drugiPodatak

tpSV += treciPodatak

#ponistavanje

prviPodatak = 0

drugiPodatak = 0

treciPodatak = 0

brojac+=1

timerTemp +=1

timerRast +=1

timerOsvet+=1

#racunanje srednje vrednosti

temperaturaKonfig = ppSV/brojac

rastojanjeKonfig = dpSV/brojac

osvetljenostKonfig = tpSV/brojac

zahtev()

#vremeTemperature je zahtev/period procitan iz maila

if (int((timerTemp)) == int((vremeTemperature\*60))):

temperaturaPosebno = temperaturaKonfig

izvestajTemperatura()

print("Temperatura")

timerTemp = 0

if (int((timerRast)) == int((vremeRastojanja\*60))):

rastojanjePosebno = rastojanjeKonfig

izvestajRastojanje()

print("Rastojanje")

timerRast = 0

if (int((timerOsvet)) == int((vremeOsvetljenosti\*60))):

osvetljenostPosebno = osvetljenostKonfig

izvestajOsvetljenost()

print("Osvetljenost")

timerOsvet = 0

s = []