PROJEKAT IZ MIKRORAČUNARSKE ELEKTRONIKE

NAZIV PROJEKTA:

Senzor požara i koncentracije alkohola u industriji

MENTOR PROJEKTA:

Vladimir Rajs

PROJEKAT IZRADILI:

Veličković Aleksa ET 28/2019

DATUM ODBRANE PROJEKTA:

17,06,2021

Sadržaj

[PROJEKAT IZ MIKRORAČUNARSKE ELEKTRONIKE 1](#__RefHeading__296_1220056955)

[1. Uvod 3](#__RefHeading__298_1220056955)

[2. Analiza problema 3](#__RefHeading__300_1220056955)

3. Povezivanje mikrokontrolera sa PC računarom 3

4. Detaljan opis svih podsistema uređaja 3

4.1. Slika uređaja u krajnjem stadijumu izrade 8

5. Koncept rešenja 8

6. Rezultati testiranja 9

7. Zaključak 9

8. Literatura 9

9. Algoritam rada programa 10

[10. Dodatak 11](#__RefHeading__316_1220056955)

# Uvod

Senzor požara i koncentracije alkohola u industriji

Ovaj senzor se koristi u industriji alkohola npr. vinarija, pivara, kao i ostalih fabrika za proizvodnju proizvoda koji sadrže alkohol.

Program za očitavanje vrednosti temperature i koncentracije alkohola sa RTC senzora temperature i MQ-3 senzora gasa se upisuje u memoriju Waspmote v1.1 razvojnog sistema i izvršava na istom.

# Analiza problema

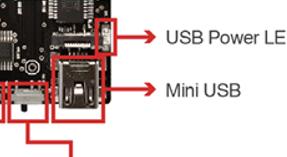
Glavni cilj jeste konstantno očitavanje vrednosti temperature i koncentracije alkohola i dati nam doznanje o situaciji u prostoriji fabrike proizvodnje ili skladištu.

Ukoliko senzor detektuje povišenu vrednost u odnosu na uobičajenu, potrebno je upaliti alarm u vidu ispisa poruke upozorenja, blinkati crvenom LED diodom i upisati datum i vreme u EEPROM memoriju.

Za normalna očitavanja će se regularno ispisivati vrednost temperature i koncentracije alkohola i blinkati zelenom LED diodom.

# Povezivanje mikrokontrolera sa PC računarom

Povezivanje vršimo putem mini USB porta na razvojnom sistemu

Pre povezivanja potrebno je instalirati potrebne drajvere za komunikaciju putem COM porta.

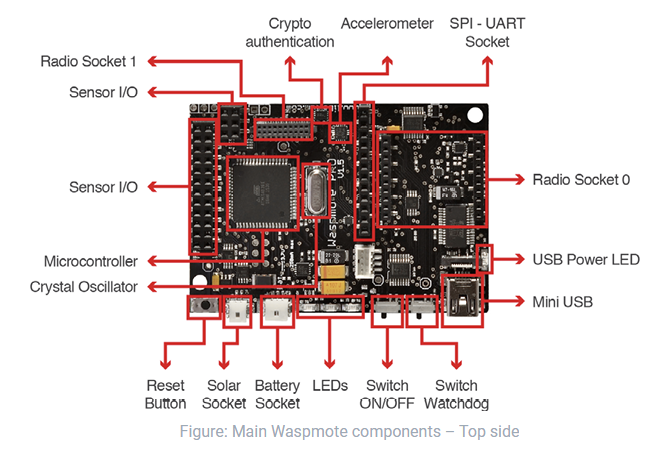
# Detaljan opis svih komponenata uređaja

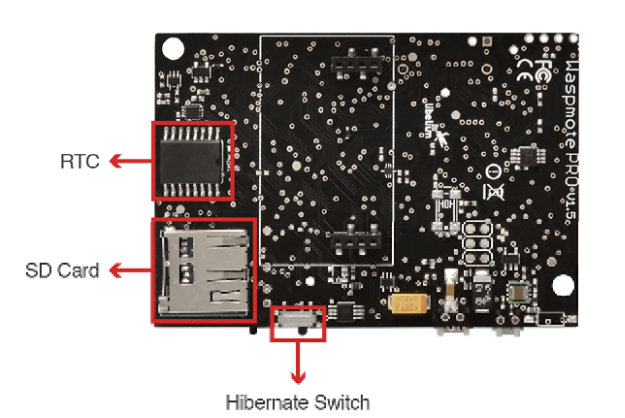
Za ovaj problem koristimo Waspmote razvojni sistem.

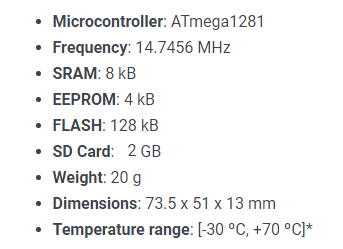
Waspmote je „open-source“ bežična senzorska platforma osmišljena radi implementacije senzorskih čvorova niske potrošnje u IoT platformi. Platforma je kompletno autonomna i napaja se iz baterijskog izvora...

Poseduje mogućnost više tipova komunikacije, kao što su GPRS, GSM, WiFi, BlueTooth...

Specifikacije hardvera:







**Tehnička specifikacija (obavezno proučiti pre povezivanja waspmote-a):**

Minimalni radni napon povezanog baterijskog izvora: 3V3

Maksimalni radni napon povezanog baterijskog izvora: 4V2

USB napon punjenja baterije: 5V

Maksimalna struja na USB konektoru: 480 mA

Apsolutne maksimalne vrednosti:

Napon n a bilo kom I/O pinu: [-0.5, +3V8]

Maksimalna struja za bilo koji I/O pin: 40 mA

USB napajanje: 7V

Napon napunjene baterije: 4V2

**Kopmonente koje se koriste u ovom programu su RTC senzor i MQ-3 senzor gasa.**

**RTC(Real Time Clock)** je elektronski sklop najčešće u vidu integrisanog kola koji meri vreme. RTC je prisutan u gotovo svim elektronskim uređajima koji imaju potrebu da prikazuju tačno vreme. U ovom programu RTC ima ulogu da ispiše datum i vreme očitanih vrednosti. RTC senzor se takodje koristi za očitavanje temperature.

Karakteristike RTC-a:

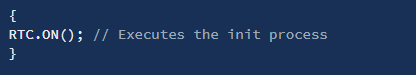
- realizovan kao zaseban sklop koji vodi računa isključivo o funkciji merenja vremena, njegov klok se ne opterećuje raznim prekidima i izvršavanjem programabilnog koda kao što to čini standardni mikrokontroler.

- ima zaseban baterijski izvor koji mu omogućuje da i nakon prekida napajanja ostatka sklopa i dalje nastavi sa radom

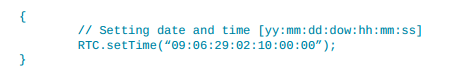
- izuzetno niska potrošnja, koja obezbređuje da uz novčić bateriju uređaj bude napajan i u periodu od nekoliko godina

- poseduje precizni eksterni oscilator 32,768 kHz koji se koristi u quartz satovima.

Pre preuzimanja podataka sa integrisanog kola RTC, mora se inicijalizovati I2C magistala, koja se koristi za komunikaciju.



Podešavanje datuma i vremena:

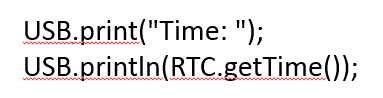


yy – godina, mm- mesec, dd- dan,

dow-dan u nedelji (npr. Sunday 01, Monday se pise kao 02, Tuesday 03….)

hh- sat, mm-minut, ss- sekund

Komanda za štampanje vremena sa RTC-a:



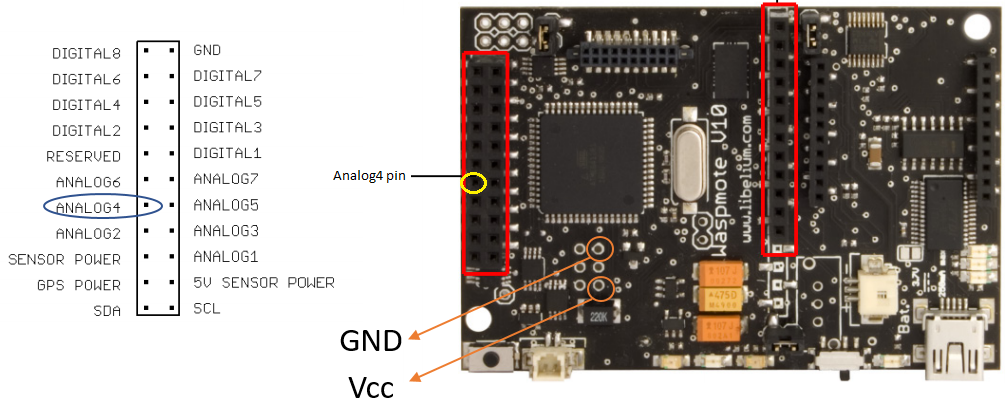
Rezultat štampe:



**MQ-3 senzor gasa**

MQ-3 senzor gasa je Metal Oxide Semiconductor *(MOS)* tip senzora koji se u ovom slučaju koristi za očitavanje koncentracije alkohola u vazduhu i ispisivanje iste na UART. Princip rada je baziran na promeni otpornosti. Termin koncenracija gasase odnosi na opisivanje zapreminske količine gasa u vazduhu. Objašnjenje za ppm: koncenracija jednog gasa (alkohol) u odnosu na drugi (vazduh). Na primer, 500 ppm znači da od milion molekula gasa, 500 molekula je alkohol, a 999500 molekula tog drugog gasa (vazduh). Takodje obezbeđuje analogni naponski signal proporcionalan koncentraciji alkohola koji se konvertuje i prosledjuje na UART.

**Povezivanje MQ-3 senzora:**



1 = Vcc

2 = GND

4 = ANALOG4

Korišćenje MQ-3 senzora gasa:

Vrednost sa senzora očitavamo na sledeći način:

void setup() {

USB.begin(); //setujemo UART

USB.println("MQ3 zagrevanje!");

delay(20000); } //sačekajmo da se senzor dovoljno zagreje

void loop() {

sensorValue = analogRead(ANALOG4); //čitamo analogni unos sa pina 0

USB.print(“Vrednost ocitana sa senzora: ");

USB.println(sensorValue); }

**Upis i čitanje EEPROM-a:**

Primer upisivanja sadržaja u određenu lokaciju:

{

Utils.WriteEEPROM(1024, ‘B’); // upisivanje ascii karaktera ‘B’ u adresu 1024

}

Primer iščitavanja sadržaja iz određene lokacije:

{

uint8\_t podatak = Utils.readEEPROM(1024); // iščitava podatak sa adrese 1024

}

Nakon svakog upisa kritične vrednosti očitane sa senzora, inkrementujemo adresu kako ne bismo overvrajtovali staru vrednost. Takođe je bitno da se ne koriste adrese niže od 512 jer može overvrajtovati konfiguarioni deo memorije mikrokontrolera!

**Rad sa LED diodama:**

Paljenje i gašenje zelene diode izvršavamo sledećim kodom:

Utils.setLED(LED1, LED\_ON);

Utils.setLED(LED1, LED\_OFF);

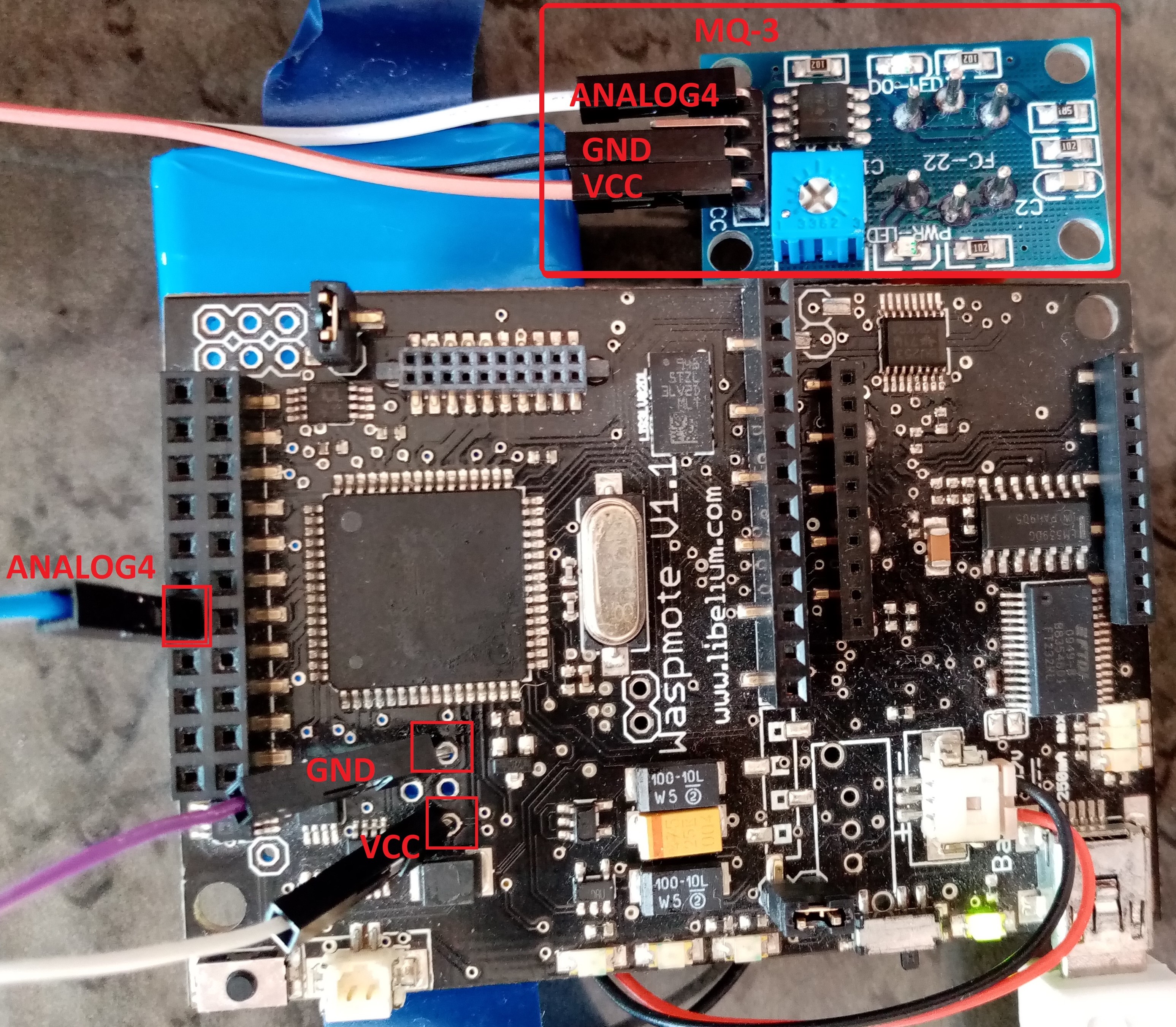
Paljenje i gašenje crvene diode izvršavamo sledećim kodom:

Utils.setLED(LED0, LED\_ON);

Utils.setLED(LED0, LED\_OFF);

Ukoliko želimo simulirati blinkanje, to činimo ubacivanjem for petlje i delay-ova u kod.

## Slika uređaja u krajnjem stadijumu izrade



# Koncept rešenja

Da bi se pristupilo zadatku, prva stvar koju treba uraditi je povezati eksterne senzore na mikrokontroler. U našem slučaju RTC je već integrisan, dok MQ-3 senzor gasa povezujemo kao što je u opisu komponenata prikazano. Nakon toga počinjemo sa ispisom koda u razvojnom okruženju Wasp IDE koji će se izvršavati na našem mikrokontroleru. Kod pišemo u programu za upravljanje Waspmote mikrokontrolerom, kompajlujemo i spuštamo na isti.

Kod se sastoji iz setup() i void () zaglavlja. U setup-u inicijalizujemo komunikaciju sa PC računarom pomoću USB porta koje nam omogućuje spuštanje koda na mikrokontroler. Kada se USB koristi za komunikaciju preko UART0, čip FT232RL radi konverziju na USB standard. Pored inicijalizacije komunikacije preko USB-a, palimo i sve senzore koje koristimo, u ovom slučaju palimo i RTC.

Nakon inicijalizacije svih komponenti, potrebno je definisati i promenljive koje ćemo koristiti za izradu koda. Naime želimo da na svake 3 sekunde bude ispisan rezultat sa stanjem temperature i koncentracije alkohola i dok su vrednosti uobičajene, blinkaće zelena LED dioda. U suprotnom, upalićemo alarm u vidu ispisa na UART, datum i vreme detektovanja ćemo upisati u EEPROM i ispisati, takođe ćemo upaliti brzo blinkanje crvenog LED-a.

Kod programa se može naći na kraju dokumentacije.

# Rezultati testiranja

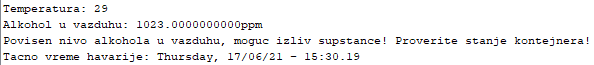
Početak programa:



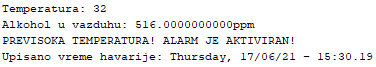
Primer standardnog ispisa:



Primer detektovane povišene koncentracije alkohola na MQ-3 senzoru:



Primer povišene temperature na RTC senzoru:



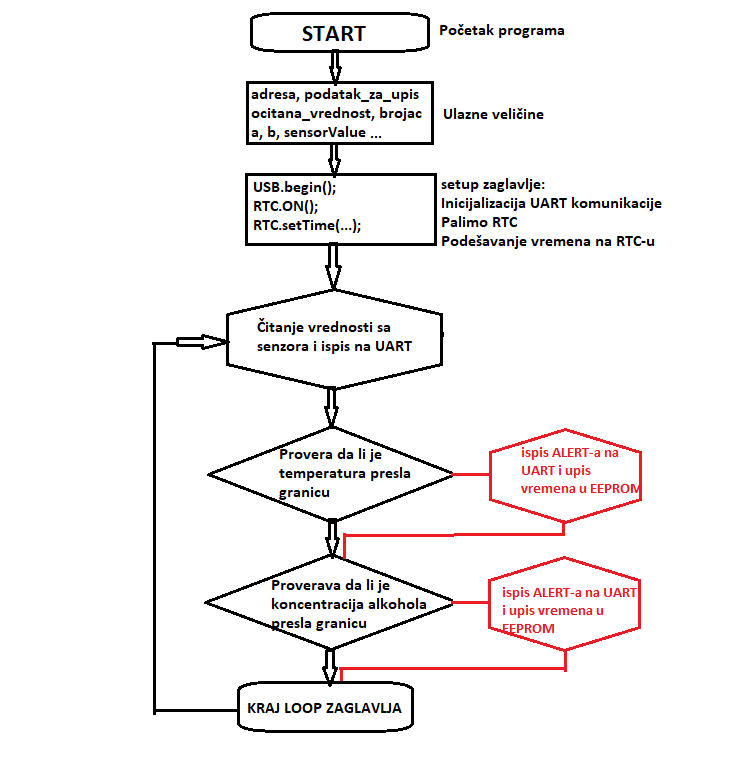
# Zaključak

Konstantno merenje ključnih vrednosti u preduzećima navedenih u uvodu nam omogućuje veću kontrolu situacije. Samim tim možemo i predvideti havariju ukoliko vidimo postepeno povećavanje merenih vrednosti i sprečiti potencijalnu. Mikrokontroler sa navedenim senzorima i upisanim programom bi se instalirao u prostorijjama tipa sale za proizvodnju, skladišta i ostalih gde se radi sa zapaljivom supstancom.

# Literatura

Literatura korišćena za izradu dokumentacije je sadržana u prezentacijama laboratorijskih vežbi.

# Algoritam rada programa



Dodatak

**Kod programa:**

int adresa=1024;

int brojac=0;

int a=3;

int b=50;

char \*writer;

char rider;

char vremeHavarije[30];

float sensorValue;

void setup(){

USB.begin();

USB.println("USB port startovan...");

USB.println("Inicijalizujem RTC..");

RTC.ON();

RTC.setTime("21:05:18:03:12:30:00");

USB.println("Zagrevam MQ-3 senzor.");

delay(20000);

}

void loop(){ // Ispis ocitanih vrednosti temperature i alkohola

USB.print("Temperatura: ");

USB.println(RTC.getTemperature(),DEC);

sensorValue = analogRead(ANALOG4);

USB.print("Alkohol u vazduhu: ");

USB.print(sensorValue);

USB.println("ppm");

if (RTC.getTemperature() > 30) {

USB.println("PREVISOKA TEMPERATURA! ALARM JE AKTIVIRAN! ");

writer=RTC.getTime();

for(int i=0; i<30; i++) {

Utils.writeEEPROM(adresa,writer[i]);

adresa++;

}

adresa=1024;

for(int i=0; i<30; i++){

rider=Utils.readEEPROM(adresa);

vremeHavarije[i]=rider;

adresa++;

}

USB.print("Upisano vreme havarije: ");

USB.println(vremeHavarije);

for(brojac=0; brojac<b; brojac++) {

Utils.setLED(LED0, LED\_ON);

delay(100);

Utils.setLED(LED0, LED\_OFF);

delay(100);

}

}

if (sensorValue > 600) {

USB.println("Povisen nivo alkohola u vazduhu, moguc izliv supstance! Proverite stanje kontejnera! ");

writer=RTC.getTime();

for(int i=0; i<30; i++){

Utils.writeEEPROM(adresa, writer[i]);

adresa++;

}

adresa=1024;

for(int i=0; i<30; i++) {

rider=Utils.readEEPROM(adresa);

vremeHavarije[i]=rider;

adresa++;

}

USB.print("Tacno vreme havarije: ");

USB.println(vremeHavarije);

for(brojac=0; brojac<b; brojac++) {

Utils.setLED(LED0, LED\_ON);

delay(100);

Utils.setLED(LED0, LED\_OFF);

delay(100);

}

}

for(brojac =0; brojac<a; brojac++) {

Utils.setLED(LED1, LED\_ON);

delay(500);

Utils.setLED(LED1, LED\_OFF);

delay(500);

}

if(adresa==4096) {

adresa=1024;

USB.println("Memorija puna, resetujem!");

delay(10000);

}

}