VEGOVA | ELEKTROTEHNIŠKO-RAČUNALNIŠKA STROKOVNA ŠOLA IN GIMNAZIJA LJUBLJANA

Podatkovno modeliranje in SQL: TempLogger

Tilen Anzeljc, R3B

Loški Potok, 6. 5. 2020

Vsebina

1. SISTEM ZA ZAJEM PODATKOV

1.1 Opis

Za zajem podatkov sem uporabil Arduino Nano in senzor BME 280. Senzor je bilo potrebno priključiti na 4 pine Arduinota : VCC, GND in dva analogna vhoda (A4 in A5).

1.2 Program

Program je sestavljen iz 3 delov:

- **inicializacija** (knjižnice, definiramo pine, tlak, ustvarimo objekt senzorja in časovnik, funkcija Setup())

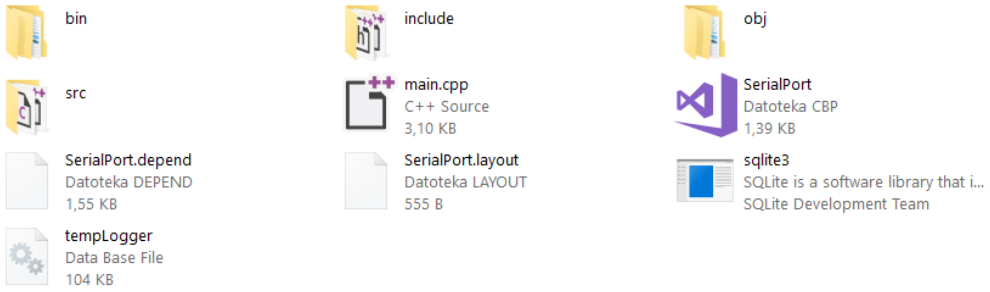
-**glavni program**(Loop() v katerem na vsake 3 minute kličemo funkcijo printValues())

-**funkcija printValues** (prebere podatke iz senzorja in jih izpiše na serijski vmesnik)

```
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <Wire.h>
#include <SPI.h>

#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10

#define SEALEVELPRESSURE_HPA (1013.25)  //definiramo zračni tlak

Adafruit_BME280 bme; // objekt BME senzorja
unsigned long long timer = millis();  //časovnik

void setup() {
  Serial.begin(9600);
  unsigned status = bme.begin(0x76);
}

//loop zanka, kjer se izvaja program
void loop() {
  if(millis()-timer>=180000){
    printValues();
    timer = millis();
  }
}

//izpis vseh vrednosti
void printValues() {
   //pretvorimo vrednosti senzorjev v string, da ga lahko izpišemo
   Serial.println(bme.readTemperature());
   Serial.println(bme.readPressure()/100.0F);
   Serial.println(bme.readHumidity());
}
```

2. SISTEM ZA PRENOS PODATKOV

Podatke je potrebno iz serijskega vmesnika prenesti na računalnik, te pa v sqlite3. To sem realiziral s pomočjo C++ v okolju CodeBlocks. Za delo s Sqllite sem potreboval sledeče datoteke: sqlite3.h, sqlite3ext.h, shell.c in sqlite3.c. Za branje iz serisjkega vmesnika pa 2 datoteki: SerialPort.hpp in SerialPort.cpp. Najprej sem moral v nastavitvah nastaviti linker, da sem lahko vse datoteke vključil v projekt.



2.1 Program

Pred main- om moramo najprej vključiti knjižnice in deklarirati potrebne globalne spremenljivke.

```
#include <iostream>
#include "SerialPort.hpp"
#include <stdio.h>
#include <string>
#include <sstream>
#include <sqlite3.h>

using namespace std;

char* portName = "COM3";      //ime porta iz katerega beremo

//#define MAX_DATA_LENGTH 255

char incomingData[6];       //podatki, ki jih preberemo iz serijskega vmesnika
char incomingData1[7];
char incomingData2[6];


SerialPort *arduino;     //naredimo objekt SerialPort razreda

void ReceiveData(sqlite3* DB, int& exit);    //funckija, ki prebere podatke
void autoConnect(sqlite3* DB, int& exit);    //funkcija, ki vzpostavi povezavo
```

Sedaj sledi main funkcija, ki se od navadne razlikuje po argumentih, ki jih dobi. To je potrebno, saj beremo iz seriskega vmesnika.

```
int main(int argc, char** argv)
{
    sqlite3* DB;     //ustvari se sqlite objekt
    int exit = 0;
    exit = sqlite3_open("tempLogger.db", &DB);  //odpremo podatkovno bazo
    if (exit) { //preverimo, če se je podatkovna baza pravilno naložila
        cerr << "Error open DB " << sqlite3_errmsg(DB) << endl;
        return (-1);
    }
    else
        cout << "Opened Database Successfully!" << endl;

    arduino = new SerialPort(portName);
    autoConnect(DB, exit);   //povežemo program s serijskim vmesnikom
}
```

Funkcija, ki prebere podatke iz serijskega vmesnika.

```
void ReceiveData(sqlite3* DB, int& exit)
{
    float readResult = arduino->readSerialPort(incomingData, 6);     //preberejo se rezultati
    readResult = arduino->readSerialPort(incomingData1, 7);
    readResult = arduino->readSerialPort(incomingData2, 6);

    if(readResult){ //ce je bil rezultat prebran
        cout<<incomingData;
        cout<<incomingData1;         //sledi izpis podatkov
        cout<<incomingData2<<endl;

        //pretvorimo prebrane podatke v c++ string
        string t(incomingData), tl(incomingData1), v(incomingData2);
        //stavek, ki ga pošljemo v sqlite
        string ddl =  "INSERT INTO PODATKI(DATUM, TEMPERATURA, TLAK, VLAGA) VALUES(CURRENT_TIMEST
        char* messaggeError;    //hrani error

        exit = sqlite3_exec(DB, ddl.c_str(), NULL, 0, &messaggeError);  //pošljemo ukaz v sqlite
        if (exit != SQLITE_OK) {     //preverimo, če se je ukaz izvedel pravilno
            cerr << "Error Adding elements" << endl;
            sqlite3_free(messaggeError);
        }
        else
            cout << "Elements added successfuly" << endl;
    }

    Sleep(10);  //nekaj delay-a
}
```

Stavek, ki se pošlje v sqlite3.

```
string ddl =  "INSERT INTO PODATKI(DATUM, TEMPERATURA, TLAK, VLAGA) VALUES(CURRENT_TIMESTAMP, "+t+", "+tl+ ", "+v+");";
```

Funkcija, ki se poveže s portom Arduinota

```
void autoConnect(sqlite3* DB, int& exit)
{
    //better than recusion
    //avoid stack overflows
    while(1) {
        // ui - searching
        std::cout << "Iskanje v teku...";
        // wait connection
        while (!arduino->isConnected()) {
            Sleep(100);
            std::cout << ".";
            arduino = new SerialPort(portName);
        }

        //Checking if arduino is connected or not
        if (arduino->isConnected()) {
            std::cout  << std::endl << "Povezava vzpostavljena na portu " << portName << std::endl;
        }

        #ifdef SEND
            //tu bi lahko pisali na serijski port
        #else
            while(arduino->isConnected()) ReceiveData(DB, exit);
        #endif
    }
}
```

4. ANALIZA PODATKOV

4.1 Struktura tabele podatki

```
sqlite> .header on
sqlite> .mode column
sqlite> pragma table_info('podatki');
cid         name          type        notnull     dflt_value  pk
----------  ------------  ----------  ----------  ----------  ----------
0           ST_BELEZENJA  INT         0                       1
1           DATUM         DATE        1                       0
2           TEMPERATURA   FLOAT       1                       0
3           TLAK          FLOAT       1                       0
4           VLAGA         FLOAT       1                       0
```

4.2 Skupno število zapisov

```
sqlite> select count(*) from podatki;
1505
```

4.3 Največja in najmanjša velikost podatka v celotnem obdobju

```
sqlite> select max(temperatura), datum from podatki;
26.07             2020-05-03 11:33:55
```

```
sqlite> select min(temperatura), datum from podatki;
19.71             2020-05-02 16:31:52
```

```
sqlite> select min(vlaga), datum from podatki;
25.2        2020-05-03 09:04:16
```

```
sqlite> select max(vlaga), datum from podatki;
46.9        2020-05-03 17:27:58
```

```
sqlite> select max(tlak), datum from podatki;
926.37      2020-05-03 19:48:53
```

```
sqlite> select min(tlak), datum from podatki;
914.33      2020-05-02 00:02:28
```

4.4 Povprečna temperatura

```
sqlite> select avg(temperatura) from podatki;
23.4269900332226
```

4.5 Razlika med največjo in najmanjšo temperaturo

```
sqlite> select max(temperatura) - min(temperatura) from podatki;
6.36
```

4.6 Število zapisov po dnevih

Ker sem imel težave s funkcijo strftime() sem moral za vsak podatek napisati drugačno poizvedbo. Prvotna poizvedba je bila enaka tej na zgornji sliki.

```
sqlite> select strftime('%d', 'datum'), count(*) from podatki group by strftime('%d', 'datum');
|1505
```

```
sqlite> select count(*) from podatki where datum like '%2020-04-30%';
186
sqlite> select count(*) from podatki where datum like '%2020-05-01%';
465
sqlite> select count(*) from podatki where datum like '%2020-05-02%';
479
sqlite> select count(*) from podatki where datum like '%2020-05-03%';
375
```

4.7 Največja in najmanjša temperatura po dnevih

```
sqlite> select max(temperatura) from podatki where datum like '%2020-04-30%';
24.29
sqlite> select max(temperatura) from podatki where datum like '%2020-05-01%';
25.51
sqlite> select max(temperatura) from podatki where datum like '%2020-05-02%';
24.98
sqlite> select max(temperatura) from podatki where datum like '%2020-05-03%';
26.07
```

```
sqlite> select min(temperatura) from podatki where datum like '%2020-04-30%';
23.28
sqlite> select min(temperatura) from podatki where datum like '%2020-05-01%';
22.08
sqlite> select min(temperatura) from podatki where datum like '%2020-05-02%';
19.71
sqlite> select min(temperatura) from podatki where datum like '%2020-05-03%';
19.87
```

4.8 Velikost vlage (30% in 70%)

```
sqlite> select count(*) from podatki where vlaga <30;
107
```

```
sqlite> select count(*) from podatki where vlaga >70;
0
```

4.9 Trend temperature

Ne moremo natančno določiti ali je temperatura padala ali rastla, saj je bil senzor v hiši in je bila temperatura precej konstantna, razen, ko je bilo odprto okno.

5. ZAKLJUČEK

Preden sem začel s projektom, si nisem predstavljal, kako bom prenesel podatke z Arduinota v podatkovno bazo. Imel sem pomisleke glede programiranja, ali bi bilo možno narediti samo en program za Arduinota in branje iz serijskega vmesnika, kateri programski jezik naj uporabim,… Zelo sem zadovoljen, saj se je projekt obnesel zelo dobro, vse je delovalo tako kot bi moralo. Naučil sem se, kako deluje C++ v povezavi s podatkovno bazo in kako lahko podatki iz serijskega vmesnika Arduinota prenesemo v program in jih tam uporabimo.