Statie meteo - Popescu David 332 AA

Introducere

Pentru acest proiect am ales să proiectez o stație meteo inteligentă. Prin acest proiect, se urmărește în primul rând afișarea condițiilor meteo din zona respectivă. Ulterior, aceste date vor fi comparate cu date reale preluate din API-uri care stocheaza date de la statii profesionale. Astfel, putem stabili acuratețea stației meteo locale.

Autor: Popescu David 332 AA

Descriere generală

Schema bloc:

Statia meteo are rolul de a achizitiona date prin intermediul senzorilor(barometric si de lumina) si sa afiseze pe un ecran LCD datele obtinute. De asemenea, proiectul foloseste protocolul de transmisie WI-FI pentru a schimba informatii pe internet cu scopul de a verifica acuratetea datelor preluate de statia locala. In esenta, se compara datele obtinute de statia proiectata de mine cu datele obtinute de statii meteo profesionale.

Pentru implementarea proiectului, am aplicat notiuni din următoarele laboratoare:

- Laboratorul 1: UART
- Laboratorul 5: SPI
- Laboratorul 6: I2C

Hardware Design

Pentru implementarea partii Hardware am avut nevoie de următoarele componente:

- Arduino UNO
- senzor barometric BME280
- senzor de lumină VEML7700
- LCD
- fire
- rezistente
- breadboard

- modul wifi esp8266

Software Design

Pentru implementarea proiectului am preluat de la senzorii de lumina si senzorul barometric datele necesare care trebuie afisate pe statia meteo. Pentru aceasta am avut nevoie de urmatoarele biblioteci:

- #include <Adafruit_Sensor.h>
- #include <Adafruit_BME280.h>

Datele obtinute care sunt relevante au fost afisate ulterior pe un ecran LCD. Pentru afisajul LCD am folosit libraria LiquidCrystal_I2C. Pentru comunicarea cu modulul wifi am folosit biblioteca SoftwareSerial.

Am realizat cu ajutorul modului ESP8266 o conexiune wifi cu scopul de a obtine datele reale dintr-un API public pentru a compara cu datele obtinute de mine.De asemenea, am actualizat in timp real datele pe un cloud numit ThingSpeak pentru a umari evolutia datelor obtinute. Pentru obtinerea datelor, am facut un GET request la API https://api.openweathermap.org/data/2.5/weather?q=Bucharest,RO&APPID=ca16c14f0f02fd6a377a18aa9d0533df, dupa locatia orasului Bucuresti(exista posibilitatea si de a alege ceva mai specific). Intr-un final, se va afisa pe ecran acuratetea medie a datelor obtinute.

Implementarea codului software a fost facuta in Arduino IDE.

Logica de implementare a codului În primul rând, am declarat variabile globale pe care le voi folosi cand voi implementa comunicarea cu modulul wifi. Am tratat si o intrerupere deoarece, va fi utila pentru a schimba starile ulterior.

În funcția setup(), se inițializează comunicarea serială cu o viteză de 9600 bps și se configurează modulul WiFi ESP8266 pentru a comunicare seriala cu viteza de 115200bps. Astfel, se obtine o conexiune la internet, folosind datele declarate global. De asemenea, se inițializează afișajul LCD și se setează pragurile și opțiunile pentru senzorul de lumină VEML7700.

În funcția loop(), se citesc valorile de la senzorii BME280 (senzorul barometric) și VEML7700 (senzorul de lumină). Apoi, se construiește un URL pentru a trimite aceste date la platforma ThingSpeak prin modulul wifi ESP8266 printr-un post request. Pentru a face acest request cu ajutorul modulului, am folosit o serie de comenzi pe seriala care initiaza cererea. Astfel, am deschis o conexiune TCP catre ThingSpeak.com pe portul 80.Dupa aceea, am trimis datele pe care doresc sa le obtin in cloud si am verificat ca am primit acknowledge, moment in care am oprit conexiunea. Tot în această funcție, am afișat și datele obținute pe serială pentru a mă asigură că acestea sunt corecte. În cele din urmă, am apelat o serie de funcții, precum:

- printVEML7700Data() și printBME280Data() - sunt utilizate pentru afișarea datelor citite de la senzorii VEML7700 și BME280 , pe portul serial.
- Funcția printLCDData() este utilizată pentru a afișa datele senzorilor pe afișajul LCD.
- switchState() - o funcție care contorizează starea în care ne aflăm și afișează pe ecran date în funcție de starea în care mă aflu.

Functia switchState() este declansata de apasarea butonului, moment in care se declanseaza, o

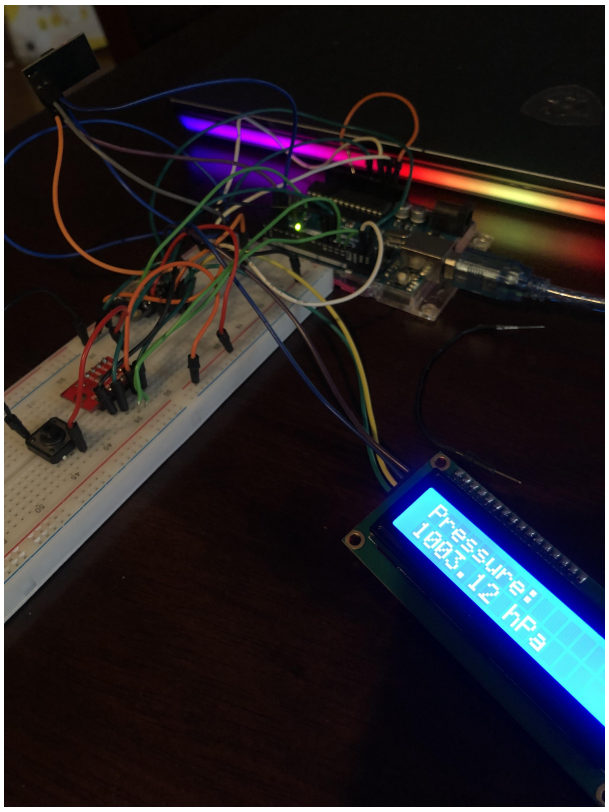
intrerupere hardware. Scopul acestei functii, este de a oferi utilizatorului ocazia, de a vedea cat mai multe informatii pe un ecran mic, obtinand alte informatii de fiecare data cand apasa butonul.

Fiecare stare are un rol specific: - Starea 0 afiseaza temperatura de la senzor

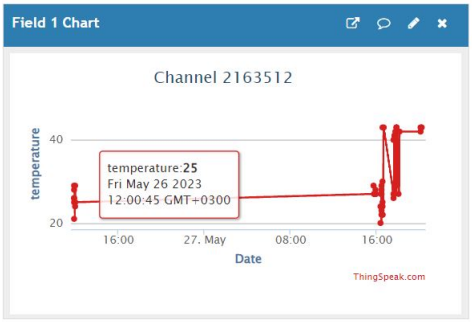
1. Starea 1 afiseaza umiditatea de la senzor
2. Starea 2 afiseaza gradul de lumina de la senzor
3. Starea 3 afiseaza presiunea atmosferica
4. Stareea 4 afiseaza toate datele mentionate, intr-un format compact
5. Starea 5 afiseaza datele preluate de la API-ul de vreme
6. Starea 6 afiseaza acuratetea datelor obtinute prin intermediul senzorilor, in comparatie cu datele obtinute din API

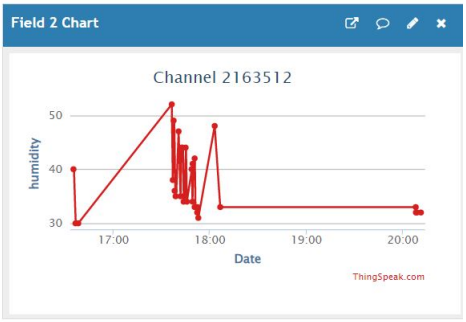
Rezultate Obţinute

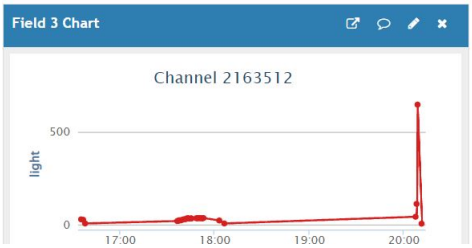
In final, am obtinut un proiect care functioneaza conform asteptarilor.



De asemenea, am reusit sa obtin o evidenta grafica in cloud.







Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Concluzii

În primul rând, doresc să menționez că a fost un proiect interesant, care s-a finalizat cu succes după rezolvarea unei serii lungi de probleme.

În primă instanță am încercat sa proiectez o stație meteo LoRaWAN pentru a primi/transmite date către internet prin intermediul unui gateway de pe thethingsnetwork. Nu am reusit sa duc aceasta provocare la capat din cauza lipsei de experienta din punct de vedere hardware. Nu am achizitionat niste componente care să satisfacă cerințele așteptate. Am reușit să inițiez o conexiune, dar din păcate raza de transmitere era mult prea mică pentru a avea posibilitatea de a prezenta acest proiect. Din aceste considerente, am decis să achiziționez un modul wifi pentru a oferi funționalitatea dorită proiectului. De asemenea, după experiența acumulată în urma realizării acestui proiect, consider că partea hardware ar putea fi îmbunătățită și am dobândit competența de a distinge mai bine pe viitor componente mai mult sau mai puțin performante.

Partea cea mai interesantă a reprezentat-o faptul că am reușit, să creez o legătură funcțională între partea software și cea hardware.

Download

finalpm.rar

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună .

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se

încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

01.04.2023 - Alegerea proiectului

12.04.2023 - Achizitionarea pieselor necesare

03.05.2023 - Conceperea montajului hardware si implementarea unor functionalitati de baza

10.05.2023 - Achizitionarea altor piese

22.05.2023 - Finalizarea etapei software

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

https://ocw.cs.pub.ro/courses/pm/lab/lab1-2022

https://ocw.cs.pub.ro/courses/pm/lab/lab2-2023

https://ocw.cs.pub.ro/courses/pm/lab/lab5-2022

https://ocw.cs.pub.ro/courses/pm/lab/lab6-2022

https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

https://www.electronicshub.org/esp8266-at-commands/
https://learn.adafruit.com/adafruit-bme280-humidity-barometric-pressure-temperature-sensor-breakout/arduino-test

https://learn.adafruit.com/adafruit-veml7700/arduino

https://lastminuteengineers.com/i2c-lcd-arduino-tutorial/

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF