Ventilator inteligent

Autor: Costache Andreea-Alexandra

Introducere

Deși confortul termic poate avea valori diferite în funcție de preferințele fiecăruia, studiile arată că organismul funcționează cel mai bine în anumite condiții. Astfel, nivelul de umiditate și cel al temperaturii din interior sunt doi factori decisivi care contribuie la obținerea unei stări de spirit pozitive, la o odihnă satisfăcătoare și la un corp sănătos. În interior, temperatura nu trebuie să fie constantă pe tot timpul anului, ci trebuie schimbată odată cu sezonul astfel încât organismul să nu sufere un șoc termic atunci când se trecere de la interior la exterior și invers. Proiectul constă în realizarea unui ventilator inteligent care să gestioneze acești parametri și să ajusteze temperatura din cameră în funcție de preferințele utilizatorului.

Descriere generală

Ventilatorul va începe să funcționeze dacă temperatura din cameră depășește un anumit prag setat de către utilizator. Dacă se depășește acest prag, ventilatorul își mărește sau scade turația în funcție de temperatura din cameră. De asemenea, pe ecranul LCD se vor afișa ora, temperatura și umiditatea. În cazul în care sistemul se supraîncălzește și ia foc, flacăra va fi detectată folosind senzorul pentru flăcări iar buzzerul va emite un sunet.

Schema bloc

Hardware Design

Listă piese

- Plăcuță Arduino UNO R3 ATMEGA328P
- Breadboard
- LCD
- Potențiometru 10K

- DC Motor
- Elice
- Senzor temperatură și umiditate DHT11
- Flame sensor KY-026
- Modul Bluetooth HC-05
- Modul Joystick cu două axe KY-023
- Rezistențe 1KΩ
- Tranzistor NPN 2N2222
- Condensator 68nF
- Diodă
- Buzzer
- Fire de legătură

Scheme electrice

Pentru realizarea schemelor am utilizat Fritzing.

Components

Schematic

Software Design

Mediu de dezvoltare

Pentru dezvoltare am folosit ArduinoIDE.

Biblioteci importate

Pentru senzori și ecran LCD am utilizat următoarele biblioteci:

- #include <LiquidCrystal.h> → pentru ecranul LCD
- #include “DHT.h” → pentru senzorul de temperatură și umiditate DHT11

- #include <SoftwareSerial.h> → pentru modulul bluetooth

Surse şi funcţii implementate

În funcția **setup()** se realizează configurările inițiale:

- afișare mesaj de început;
- configurare pini(input, output);
- inițializare variabile auxiliare;
- introducere parolă + verificare corectitudine valori introduse;
- introducere temperatură de referință și oră + verificare corectitudine valori introduse.

Parolă

Pentru a introduce parola putem controla joystick-ul pe direcțiile sus-jos-stânga-dreapta folosind funcțiile auxiliare:

- moveRight();
- moveLeft();
- decrease(position);
- increase(position).

Verificarea corectitudinii datelor se realizează cu funcția checkPassword().

Temperatura de referință

Pentru setarea temperaturii de referință putem controla joystick-ul pe direcțiile sus-jos-stânga-dreapta folosind funcțiile auxiliare:

- partea întreagă a temperaturii:

- moveRight_temp_intreg();
- moveLeft_temp_intreg();
- decrease_temp_intreg(position);
- increase_temp_intreg(position).

- partea partea zecimală a temperaturii:

- moveRight_temp_zecimal();
- moveLeft_temp_zecimal();
- decrease_temp_zecimal(position);
- increase_temp_zecimal(position).

Verificarea corectitudinii se realizează cu ajutorul funcțiilor: check_temp_intreg() și check_temp_zecimal().

Ora curentă

Pentru setarea orei curente, atât pentru oră cât și pentru minute, joystick-ul poate fi controlat pe direcțiile sus-jos.

Pentru setarea orei se folosesc funcțiile:

- pentru primul digit:

- decrease_hour_first(position);
- increase_hour_first(position).

- pentru cel de-al doilea digit:

- decrease_hour_second(position);
- increase_hour_second(position).

Verificarea corectitudinii datelor se realizează folosind check_hour_first() și check_hour_second().

Pentru minute utilizez funcțiile:

- pentru primul digit:

- decrease_minute_first(position);
- increase_minute_first(position).

- pentru cel de-al doilea digit:

- decrease_minute_second(position);
- increase_minute_second(position).

Verificarea corectitudinii datelor se realizează folosind: check_min_first() și check_min_second().

Tot în funcția setup() se calculează temperatura de referință ca fiind suma dintre partea întreagă și partea zecimală. Asemănător se calculează și ora.

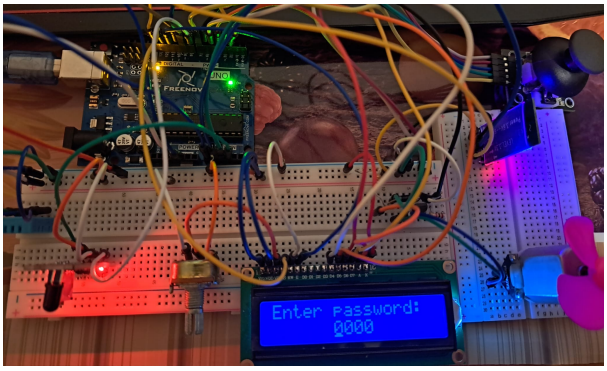
În funcția **loop()** are loc citirea temperaturii, a umidității, se incrementează timpul și se fac verificările pentru ventilator(dacă trebuie să pornească sau nu), pentru detecția flăcărilor și se transmit datele prin Bluetooth.

Rezultate Obţinute

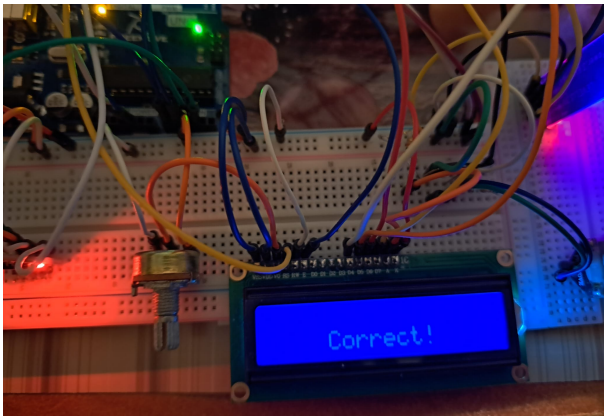
Funcționalitate

Pentru că nu dorim ca altcineva în afară de noi să configureze ventilatorul, acesta are o parolă iar pentru a putea porni ventilatorul trebuie mai întâi să introducem parola și să realizăm câteva

configurări.

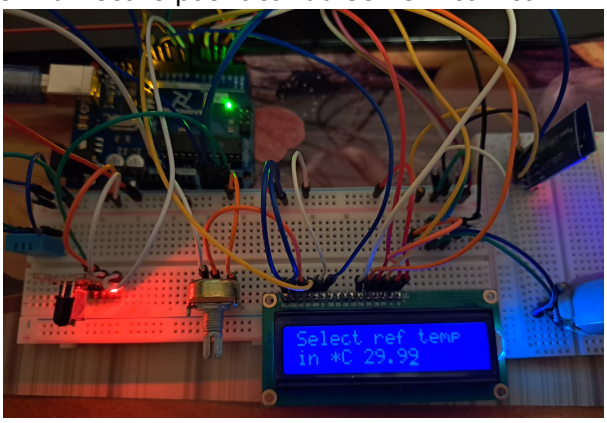


Cu ajutorul joystick-ului vom introduce parola(1234 conform setării făcute în cod) iar dacă a fost introdusă corect atunci vom trece la pasul în care setăm o temperatură de referință, altfel ne vom întoarce la introducerea parolei.



După ce am introdus parola cu succes, se setează temperatura de referință în °C, adică temperatura de la care vrem să pornească ventilatorul, în cazul în care temperatura măsurată de senzor e mai mare decât aceasta, în caz contrar, ventilatorul va fi oprit.

Pentru o precizie cât mai bună, temperatura de referință va fi un număr zecimal, astfel că mai întâi vom introduce partea întreagă și apoi partea zecimală, pentru fiecare pas făcându-se verificări ca datele să fie valide.

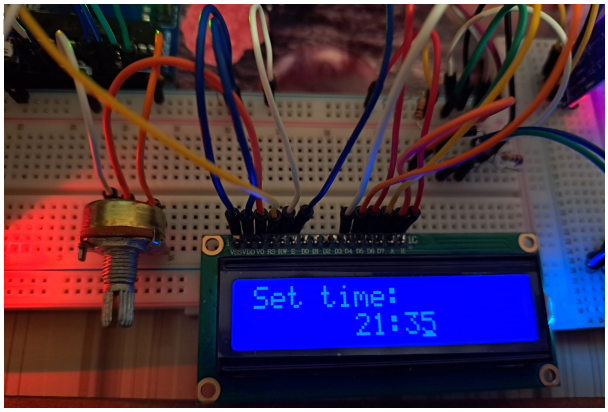


Dacă datele nu sunt valide, ne întoarcem la pasul în care introducem parola.

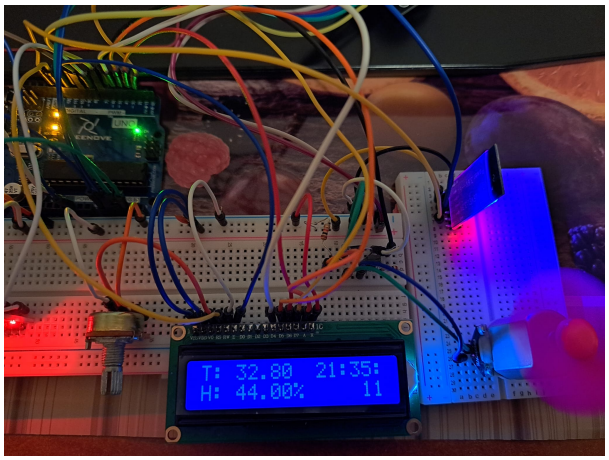
După ce setăm temperatura de referință va fi nevoie să setăm și timpul(ora & minute), deoarece nu am avut la dispozitie un modul RTC.

Atât pentru oră cât și pentru minute, se introduce fiecare digit separat pentru a face verificarea ca datele să fie valide. Secundele vor fi afișate din cod.

Dacă datele sunt incorecte ne întoarcem la pasul în care



Dacă data și minutele sunt valide atunci pe ecran se vor afișa temperatura măsurată în °C, umiditatea(în procente) și ora.



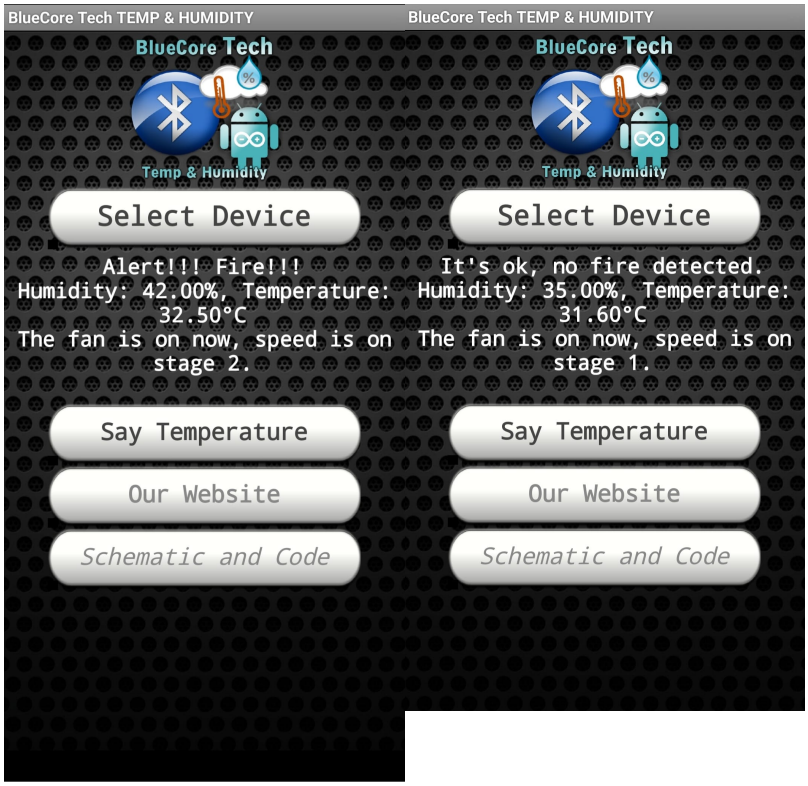
Există 3 trepte de funcționare pentru ventilator, în funcție de diferența dintre cele două temperaturi:

1. diferența este în intervalul [0, 2) ⇒ analogWrite(MOTOR, 100);
2. diferența este în intervalul [2, 5) ⇒ analogWrite(MOTOR, 150);
3. diferența este în intervalul [5, infinit) ⇒ analogWrite(MOTOR, 200).

Cu ajutorul modulului Bluetooth HC-05 și al aplicației *Arduino Temp and Humidity* (din Magazin Play), vom primi următoarele informații pe telefonul mobil conectat prin bluetooth la modul: temperatura, umiditatea, dacă ventilatorul este on sau off și dacă senzorul a detectat sau nu flăcări.

De asemenea, în cazul în care sistemul ia foc, flacăra va fi detectată cu ajutorul senzorului pentru flăcări KY-026 iar buzzer-ul va scoate un sunet.

Funcționarea completă poate fi vizualizată în acest demo: Demo Ventilator Inteligent



Concluzii

Am reușit să implementez funcționalitățile dorite.

Am întâmpinat câteva probleme la pornirea motorului deoarece pe ecran apăreau caractere random.

După câteva încercări, am reușit să rezolv problema punând un condensator683 (68nF) între VCC și GND pentru a atenua spike-urile.

Download

Arhivă proiect

Jurnal

- **21.04.2022** - Alegere temă proiect
- **28.04.2022** - Creare pagină wiki
- **05.05.2022** - Achiziționare piese
- **12.05.2022** - Testare module și senzori
- **12-23.05.2022** - Implementare, realizare schematic + proiect fizic
- **27.05.2022** - Completare pagină wiki

Bibliografie/Resurse

Ecran LCD

Joystick KY-023

Joystick KY-023

Senzor temperatură și umiditate DHT11

Senzor pentru flăcări KY-026

Modul Bluetooth

Caractere random

Caractere random

Export to PDF