Ventilator IoT

Nume: Nicolae Paul
Grupa: 333CB

Introducere

Prezentare

Proiectul consta intr-un ventilator responsabil sa mentine o atmosfera placuta si racoroasa. Ventilatorul o sa actioneze cat timp va repera miscare in proximitatea sa, ramanand activ in plus si pentru cateva secunde dupa ce nu o sa mai detecteze miscare in apropiere. In plus, ventilatorul poate fi accesat prin Wi-Fi de catre orice device conectat in aceeasi retea locala cu acesta.

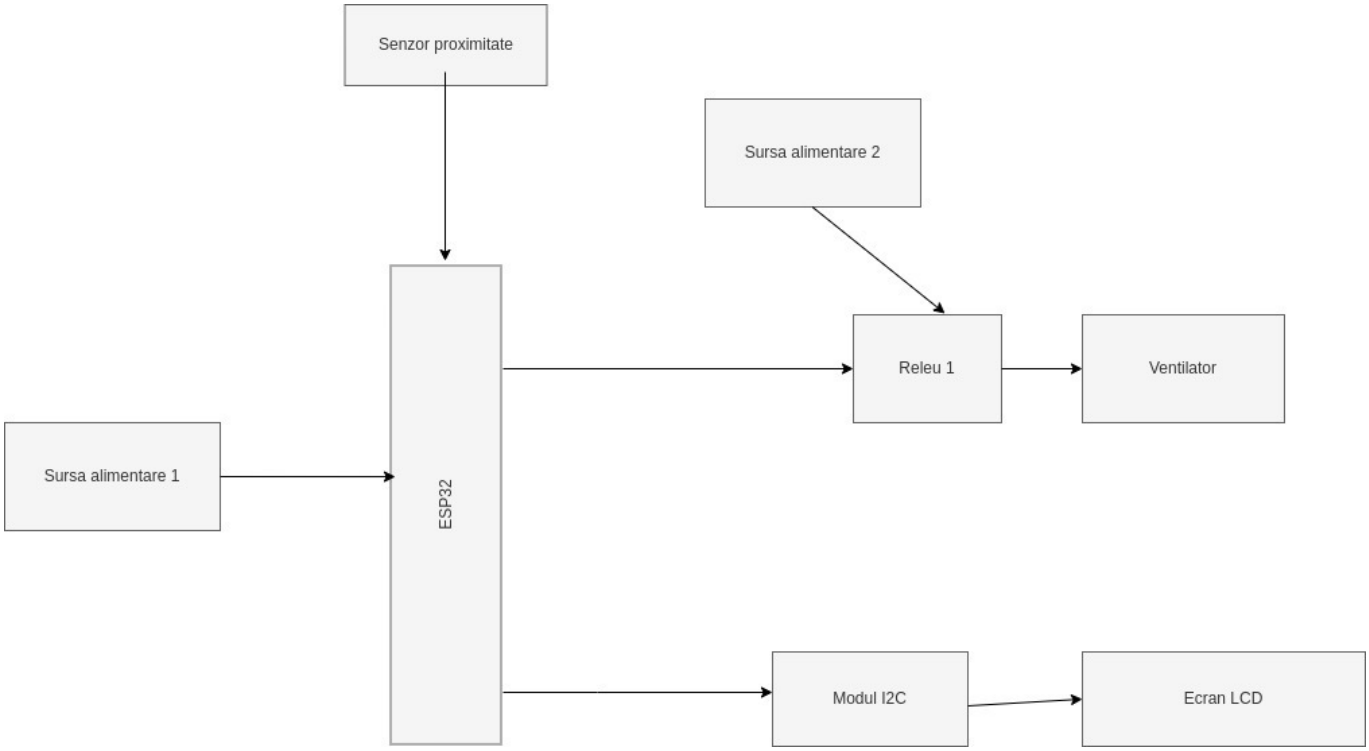
Ideea

Odata cu venirea sesiunii de vara, in camin se face mult prea cald, asa ca m-am gandit sa imbin utilul cu placutul si sa fac un ventilator sa-mi fac sesiunea mai suportabila 😉

Descriere generală

Placuta ESP32 este alimentata la PC. Senzorul de proximitate este conectat la placuta si va citi mereu datele din mediul inconjurator. Bateria de 12V va alimenta ventilatorul, insa alimentarea este conditionata de un Releu care este controlat de catre placuta. Ecranul LCD, care are integrat un modul I2C, este si el conectat la placuta, primind starea de functionare a ventilatorului.

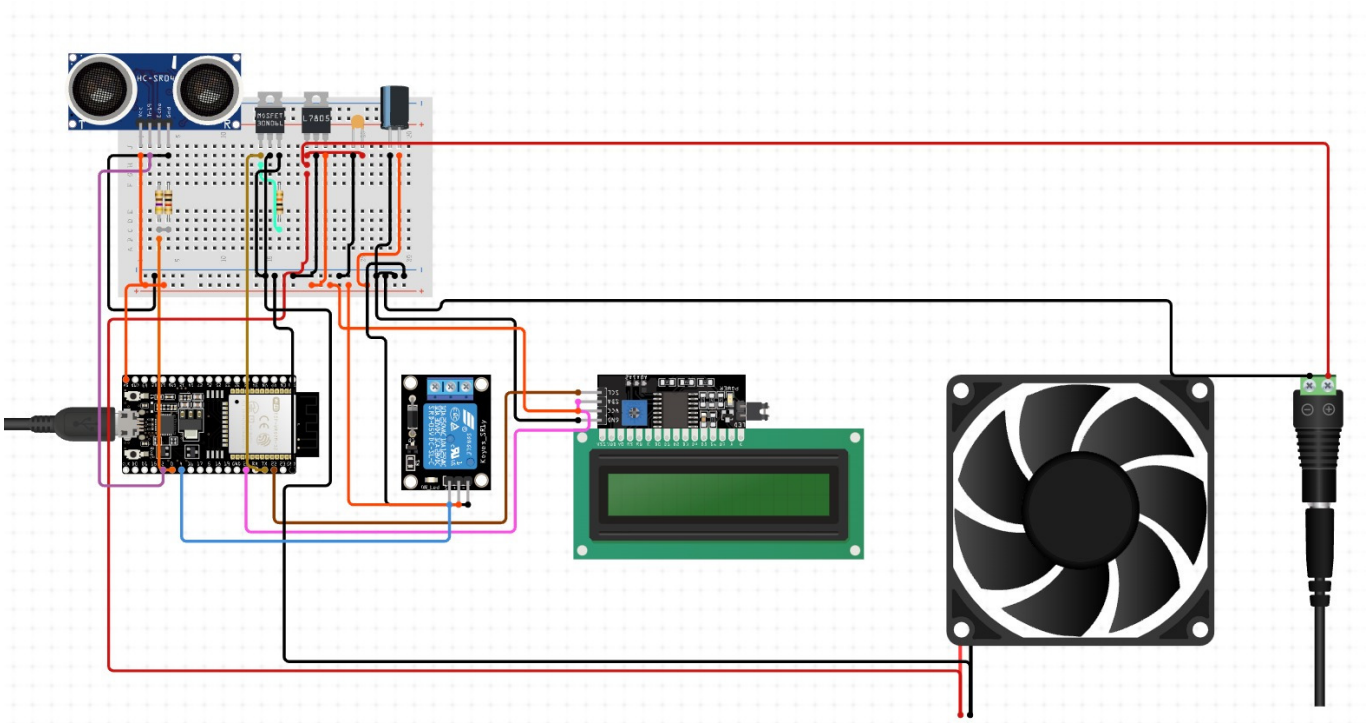
Schema Bloc



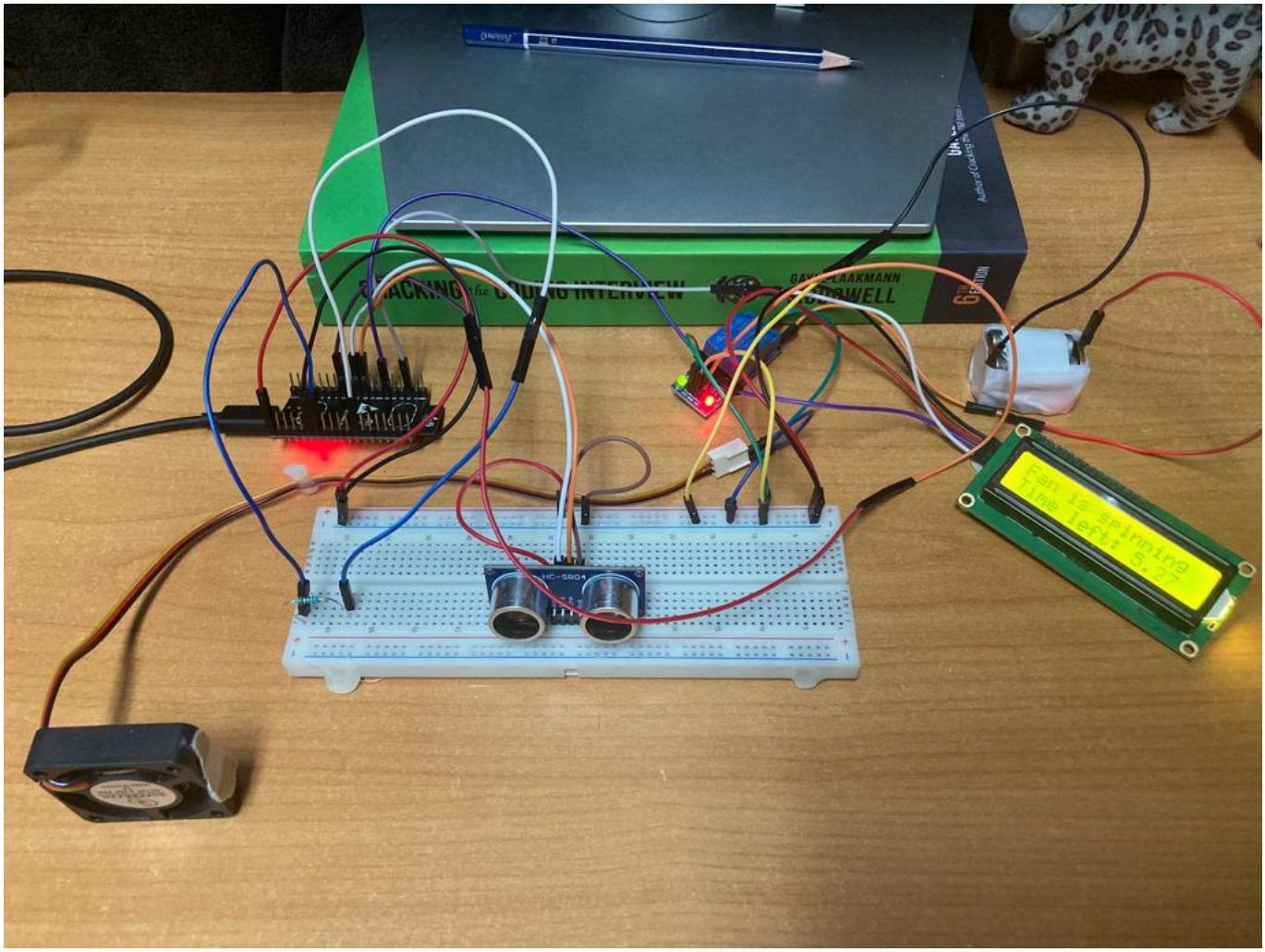
Hardware Design

- ESP32-WROOM-32D
- Senzor de proximitate ultrasonic HC-SR04
- Ventilator 12V CY410/A
- Modul Releu canal 5V
- 4 baterii de 12V
- Ecran LCD 1602 integrat cu Modul I2C
- Fire mama-mama, mama-tata, tata-tata

Imagine prototip Hardware



Implementare finala Hardware



Software Design

Mediu de dezvoltare

La nivel de implementare a codului, am folosit Arduino IDE. Schema electrica a fost realizata in Circuit.io.

Biblioteci folosite

- LiquidCrystal_I2C.h, care ofera un API pentru comunicarea prin protocolul I2C cu ecranul LCD
- NewPing.h, care ofera un API pentru comunicarea cu senzorii ultrasonici, pentru a simplifica utilizarea la nivel de cod a senztorului de proximitate
- WifiClient.h, pentru a initia si intretine conexiuni cu retele Wi-Fi
- BlynkSimpleEsp32.h pentru a integra ESP32-ul cu platforma Blynk IoT pentru a imi putea crea aplicatia cu care sa controlez diversi pin ai placutei mele

Descriere functionala

- Codul urmeaza, in toata logica scrisa in loop(), workflowul intregului proiect. Ideea principala a proiectului este de a porni ventilatorul prin doua moduri diferite: prin apropierea de senzorul de proximitate sau din aplicatia Blynk IoT, prin intermediul conexiunii la aceeasi retea locala.
- Fireste, componenta cea mai utila a proiectului fiind fix posibilitatea de a controla ventilatorul de la distanta, codul prioretizeaza setarile din aplicatie ale ventilatorului fata de diferenta de stare de miscare din jurul senzorului ultrasonic. Pentru a face posibila aceasta prioretizare, a fost nevoie de un pin de input care sa citeasca semnalul primit de la pinul pe care il poate seta switch-ul din aplicatie, pentru a determina astfel daca starea ventilatorului este strict controlata via WiFi, fara necesitatea prezentei miscarii langa senzorul de proximitate.
- Legat de comportamentul ventilatorului la diverse detectari de miscare in apropierea senzorului ultrasonic, workflow-ul este urmatorul: cat timp senzorul detecteaza obiecte in proximitatea sa, releul o sa permita alimentarea ventilatorului de catre baterie. Din momentul in care senzorul nu mai detecteaza miscare, ventilatorul se va mai invarti o scurta perioada de timp egala cu un delay setat in prealabil in cod print-un macro. Calcularea constanta a delay-ului si respectarea nedepasirii acestuia o fac prin intermediul metodei millis(), calculand constant diferenta de timp dintre momentul curent si momentul in care senzorul nu a mai detectat miscare. Daca timpul de delay al ventilatorului nu s-a scurt si senzorul a inceput sa detecteze iar miscare, la viitoarea recalculare contorul o sa fie resetat la timpul maxim de asteptare (de delay).
- Orice mesaj afisat la LCD este facut prin apelul metodelor puse la dispozitie de API-ul oferit de biblioteca LiquidCrystal_I2C.
- Pentru ca senzorul de proximitate sa detecteze distanta dintre el si un posibil obiect, folosect metoda ping_cm() pusa la dispozitie de biblioteca NewPing.h
- Conexiunea la serverul Blynk IoT este facuta prin intermediul unui token generat de aplicatie, fiind

nevoie sa se introduca in plus credentialele retelei locale pentru a face pozibila comunicarea intre server si placuta ESP32 prin intermediul modulului WiFi integrat in placuta.

- Logica ventilatorului la detectarea obiectelor de catre senzorul de proximitate

```
78    distanceToObj = motionSensor.ping_cm();
79
80    if (!isSpinning && distanceToObj) {
81      countingDelay = false;
82      isSpinning = true;
83      digitalWrite(RELAY_PIN, HIGH);
84    }
85
86    if (!distanceToObj && !countingDelay && isSpinning) {
87      countingDelay = true;
88      start = millis();
89    }
90
91    if (!distanceToObj && countingDelay && isSpinning) {
92      currTime = millis();
93      timeSpend = currTime - start;
94    }
95
96    if (distanceToObj && isSpinning) {
97      start = 0;
98      currTime = 0;
99      timeSpend = 0;
100
101     countingDelay = false;
102   }
103
104   if (timeSpend > FAN_DELAY) {
105     start = 0;
106     currTime = 0;
107     timeSpend = 0;
108
109     isSpinning = false;
110     countingDelay = false;
111
112     digitalWrite(RELAY_PIN, LOW);
113   }
114
```

Rezultate Obţinute

La final am obtinut un ventilator care ar putea fi cu usurinta adaugat in gama device-urile unui Smart

Home, pastrand in acelasi timp si caracteristicile si logica unui ventilator obisnuit si receptiv la miscarile oamenilor.

Concluzii

- Am realizat cat de importante si detaliate sunt componentele Hardware intr-un proiect, fiecare avand particularitatile ei aparte, caracteristici care te pot ingreuna daca nu le stapanesti si nu stii sa le controlezi, sau care te pot salva, cum a fost pentru mine placuta ESP32 care are integrat si modulul WiFi, facand conexiunea cu un server mult mai usora.
- Am ajuns la aceasta concluzie, deoarece eu initial foloseam in proiect si un atomizor, dar pentru ca nu am stiut cum sa pun in schema electrica un transformator, am fost nevoit sa renunt la atomizor.
- In afara de acest impediment, a fost foarte fun sa scriu codul pentru intregul proiect si sa vad cum ma pot ajuta de putina electronica pe care o stiu pentru a face posibila interactiunea unei componente fizice atat cu mediul inconjurator, cat si cu niste servere prin intermediul unor requesturi.
- Foarte placuta experinta si simt ca m-a dezvoltat ca inginer :)

Download

Codul se gaseste in fisierul .ino in arhiva de mai jos.

ventilator_iot.zip

Bibliografie/Resurse

Resurse Hardware

ESP32 Pinout Reference: https://randomnerdtutorials.com/esp32-pinout-reference-gpios/
ESP32 with HC-SR04: https://randomnerdtutorials.com/esp32-hc-sr04-ultrasonic-arduino/

Resurse Software

Biblioteca LCD:
https://create.arduino.cc/projecthub/arduino_uno_guy/i2c-liquid-crystal-displays-5b806c
Arduino Time Functions:
https://linuxhint.com/time-functions-arduino/#:~:text=In%20Arduino%2C%20time%20functions%20are,by%20using%20the%20time%20functions

Export to PDF