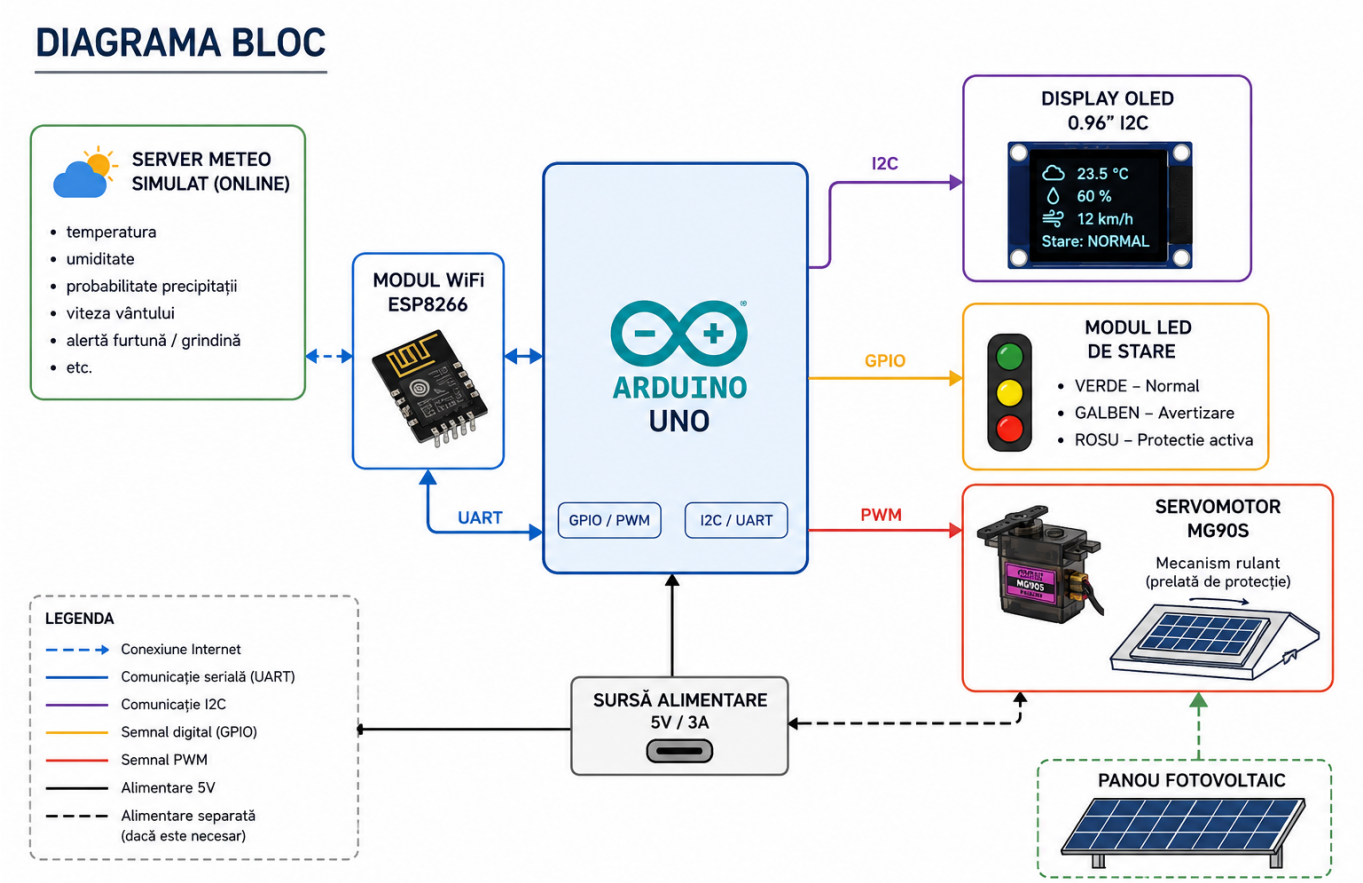
WKH_2: Sistem automatizat protecție panouri fotovoltaice

Delureanu Ana-Maria 333CB

1. Introducere

- Odată cu extinderea utilizării energiei regenerabile, panourile fotovoltaice au devenit o componentă esențială atât în sectorul rezidențial, cât și în cel industrial. Deși eficiente din punct de vedere energetic, acestea rămân vulnerabile la fenomene meteorologice severe, precum grindina, furtunile sau precipitațiile abundente, care pot produce deteriorări semnificative și costuri ridicate de întreținere sau înlocuire.
- Majoritatea soluțiilor existente utilizează senzori locali și reacționează doar în momentul apariției fenomenului meteorologic. Proiectul propus urmărește realizarea unui sistem automatizat de protecție pentru panouri fotovoltaice, bazat pe monitorizarea online a datelor meteorologice și anticiparea condițiilor meteo nefavorabile.
- Sistemul utilizează o placă Arduino UNO și un modul WiFi ESP8266 pentru preluarea periodică a datelor meteo. În funcție de informațiile analizate, un servomotor acționează automat mecanismul de protecție al panoului fotovoltaic. Pentru afișarea informațiilor și semnalizarea stării sistemului sunt utilizate un display OLED și LED-uri de stare.
- Scopul proiectului este realizarea unei soluții automate și preventive pentru protecția panourilor fotovoltaice, utilizând concepte specifice sistemelor embedded, comunicației IoT și controlului automatizat.

Descriere generală



Sistemul este alcătuit dintr-o placă Arduino UNO, un modul WiFi ESP8266, un servomotor MG90S, un display OLED și module LED pentru semnalizarea stării sistemului. Scopul proiectului este protejarea unui panou fotovoltaic prin acționarea automată a unei prelate rulante atunci când sunt detectate condiții meteorologice nefavorabile.

Arduino UNO reprezintă unitatea principală de control. Acesta primește date meteorologice prin intermediul modulului ESP8266 și decide extinderea sau retragerea mecanismului de protecție.

Pentru simularea condițiilor reale, sistemul utilizează un server meteo online simulat, inspirat de platformele utilizate de instituții precum ANM. Acesta furnizează informații precum probabilitatea de precipitații, viteza vântului sau alerte de furtună și grindină. Astfel, sistemul poate lua decizii preventive înainte de apariția efectivă a fenomenelor periculoase.

Mecanismul de protecție este acționat de servomotorul MG90S, care extinde sau retrage prelata de protecție deasupra panoului fotovoltaic. Starea sistemului este afișată pe display-ul OLED și semnalizată vizual prin LED-uri de stare.

Alimentarea sistemului se realizează printr-o sursă de 5V/3A cu conector Type-C, utilizată pentru alimentarea plăcii Arduino și a componentelor conectate.

2. Hardware Design

Nr.	Componenta	Cantitate	Link
1	Arduino UNO R3 (ATmega328P)	1	https://sigmanortec.ro/Placa-dezvoltare-UNO-R3-Arduino-Compatibil-ATmega328p-CH340G-cu-bara-pini-p170362384

2	Modul WiFi ESP8266 ESP-01	1	https://sigmanortec.ro/Modul-adaptor-ESP-01-ESP8266-p192538782
3	Adaptor ESP-01 pentru ESP8266	1	https://sigmanortec.ro/Modul-adaptor-ESP-01-ESP8266-p192538782
4	Servomotor MG90S (180°)	1	https://sigmanortec.ro/Servomotor-MG90S-angrenaje-metal-p209610310
5	Modul buzzer activ 5V	1	https://sigmanortec.ro/Modul-buzzer-activ-p136261325
6	Modul LED semafor 3.3–5V	4	https://sigmanortec.ro/modul-led-semafor-56mm-33-5v
7	Breadboard MB-102 (830 puncte)	1	https://sigmanortec.ro/Breadboard-830-puncte-MB-102-p125923983
8	Fire Dupont Mama-Mama	1 set	https://sigmanortec.ro/40-fire-Dupont-10cm-Mama-Mama-p129872525
9	Fire Dupont Tata-Mama	1 set	https://sigmanortec.ro/40-fire-Dupont-10cm-Tata-Mama-p210855157
10	Display OLED 0.96” I2C	1	https://sigmanortec.ro/Display-OLED-096-I2C-IIC-Albastru
11	Cablu USB Type-C	1	—
12	Sursă alimentare 5V / 3A	1	—
13	Mini panou solar / machetă panou solar	1	—
14	Capac rigid / mini prelată pentru protecție	1	—
15	Suport mecanic pentru panou și servomotor	1	—
16	Bandă dublu adezivă / șuruburi / coliere	-	—

Stadiul actual al implementării hardware

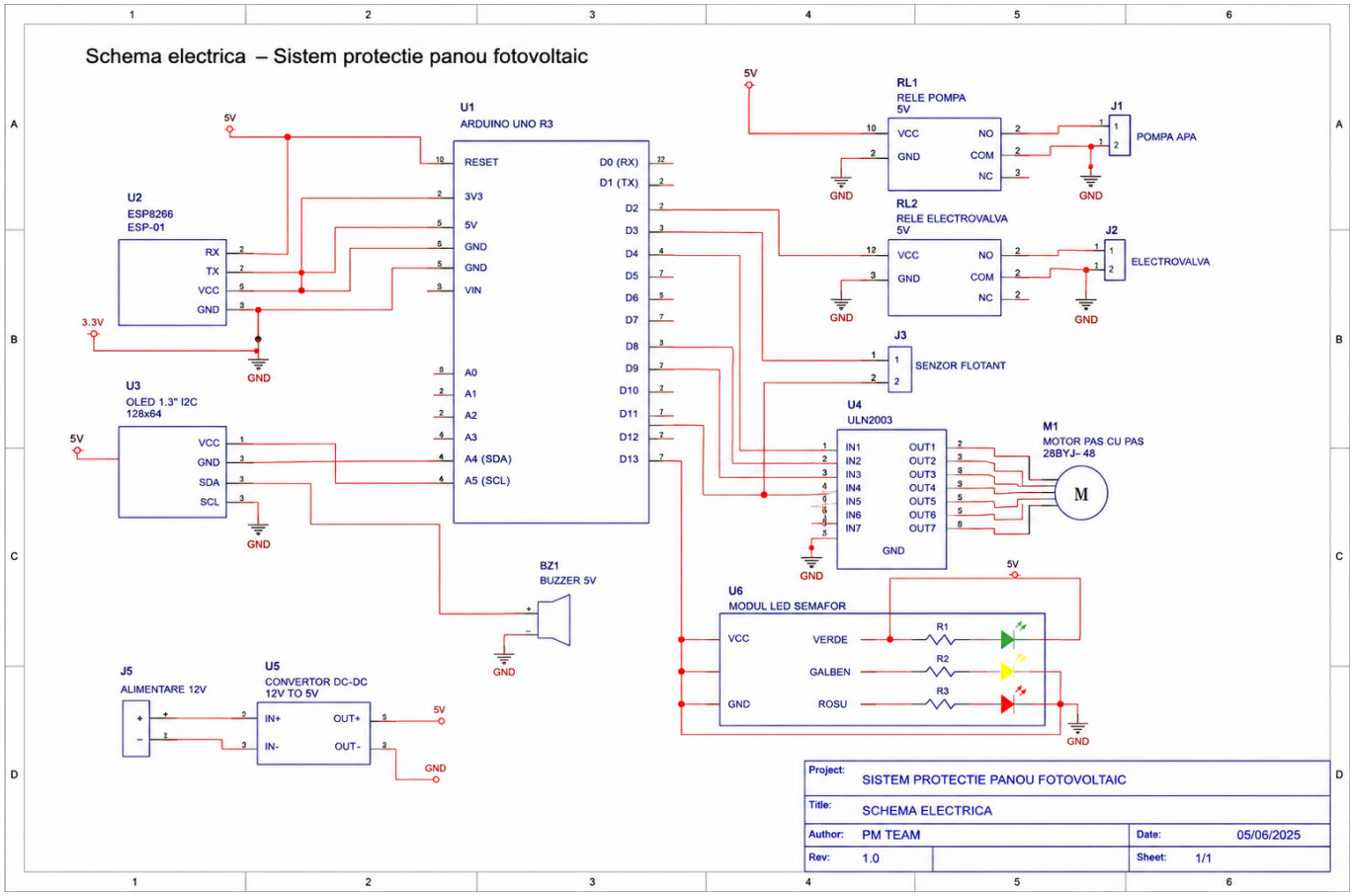
În această etapă au fost stabilite componentele principale ale sistemului și modul în care acestea vor interacționa. Proiectul este format dintr-o unitate de control Arduino UNO, un modul WiFi ESP8266 pentru preluarea datelor meteo online, un display OLED pentru afișarea stării sistemului, un modul LED pentru semnalizare vizuală și un mecanism motorizat pentru acționarea prelatei rulante.

Partea mecanică a proiectului este în curs de realizare și include un suport fizic pentru panoul fotovoltaic, două ghidaje laterale din profil U și un ax pe care se va rula materialul de protecție. Mecanismul are rolul de a simula extinderea și retragerea automată a unei prelate în funcție de condițiile meteorologice primite de la serverul meteo simulat.

Conexiuni și pini utilizați

Componentă	Pin componentă	Pin Arduino UNO	Explicație
ESP8266 ESP-01	TX	D2	Linie de recepție software serial pentru datele trimise de modulul WiFi.

ESP8266 ESP-01	RX	D3	Linie de transmitere software serial către modulul WiFi.
ESP8266 ESP-01	VCC	3.3V	Modulul ESP8266 funcționează la 3.3V.
ESP8266 ESP-01	GND	GND	Masă comună cu Arduino.
OLED I2C	SDA	A4	Linie de date pentru comunicația I2C.
OLED I2C	SCL	A5	Linie de ceas pentru comunicația I2C.
OLED I2C	VCC	5V / 3.3V	Alimentarea display-ului, în funcție de specificațiile modulului.
OLED I2C	GND	GND	Masă comună.
LED verde	IN/G	D8	Indică funcționarea normală.
LED galben	IN/Y	D9	Indică stare de avertizare.
LED roșu	IN/R	D10	Indică protecție activată.
Motor rulare prelată	IN1	D4	Control motor / driver.
Motor rulare prelată	IN2	D5	Control motor / driver.
Motor rulare prelată	IN3	D6	Control motor / driver.
Motor rulare prelată	IN4	D7	Control motor / driver.
Driver motor	VCC	5V	Alimentare motor/driver.
Driver motor	GND	GND	Masă comună cu Arduino.



3. Software Design

3.1 STADIUL ACTUAL AL IMPLEMENTARII SOFTWARE

În cadrul proiectului a fost realizată implementarea completă a unui sistem embedded inteligent destinat protecției automate a panourilor fotovoltaice (și a culturilor agricole) împotriva fenomenelor meteorologice severe, precum grindina.

Sistemul este format din două componente software principale:

- firmware-ul embedded care rulează pe placa Arduino UNO / ATmega328P;
- serverul software care simulează un serviciu meteorologic de tip ANM și transmite alerte către sistem.

Firmware-ul embedded are rolul de:

- a realiza conexiunea la rețeaua WiFi prin intermediul modulului ESP8266;
- a interoga periodic serverul pentru obținerea stării meteorologice;
- a interpreta comenzile primite;
- a controla elementele hardware ale sistemului:
  - servomotorul;
  - display-ul OLED;
  - LED-urile de stare;
  - buzzer-ul de avertizare.

Serverul software are rolul de:

- a simula un serviciu de tip nowcasting;
- a furniza starea meteorologică curentă;
- a permite modificarea manuală a stării sistemului pentru demonstrarea funcționării proiectului.

Implementarea software este complet funcțională și are rolul unui proof-of-concept, scopul principal fiind demonstrarea fezabilității unei soluții embedded capabile să reacționeze automat la alerte meteorologice primite prin intermediul unei infrastructuri client-server.

Deși sistemul nu reprezintă, în forma actuală, o soluție industrială completă, dezvoltarea proiectului a urmărit valorificarea cât mai eficientă a cunoștințelor dobândite în cadrul laboratoarelor și cursurilor de microprocesoare, precum și utilizarea resurselor hardware disponibile, pentru a evidenția potențialul unei astfel de soluții în aplicații reale.

3.2 MEDIUL DE DEZVOLTARE UTILIZAT

Dezvoltarea proiectului a presupus utilizarea unui mediu software atât pentru componenta embedded, cât și pentru componenta server-side. Pentru implementarea firmware-ului destinat microcontrolerului ATmega328P au fost utilizate instrumente specifice dezvoltării pe platforme AVR, iar pentru simularea serviciului meteorologic a fost utilizat un server HTTP implementat în Python.

Componentă	Rol / Utilizare
Visual Studio Code	mediu principal de dezvoltare și organizare a proiectului
PlatformIO	compilarea, configurarea și încărcarea firmware-ului pe placa Arduino UNO

avr-gcc	compilator utilizat pentru generarea codului executabil destinat microcontrolerului
Python 3	dezvoltarea componentei software de tip server
Flask	implementarea serverului HTTP și a interfeței web de simulare ANM
Serial Monitor	debugging și monitorizarea comunicației seriale

Firmware-ul embedded a fost dezvoltat în limbajul C, utilizând biblioteci AVR standard și acces direct la registrele microcontrolerului pentru configurarea și controlul perifericelor hardware.

Componenta server-side a fost implementată în limbajul Python, utilizând framework-ul Flask pentru realizarea unei infrastructuri client-server capabile să transmită periodic stările meteorologice către sistemul embedded.

3.3 LIBRARII SI SURSE THIRD-PARTY UTILIZATE

În cadrul proiectului au fost utilizate atât biblioteci standard specifice platformei AVR, necesare controlului direct al perifericelor hardware, cât și biblioteci externe utilizate pentru implementarea componentei server-side și a comunicației dintre sistem și infrastructura software.

Biblioteci utilizate în firmware

Bibliotecă	Rol / Utilizare
avr/io.h	acces direct la registrele microcontrolerului și configurarea perifericelor hardware
avr/interrupt.h	gestionarea întreruperilor hardware
stdio.h	debugging și afișare mesaje prin comunicația USART
string.h	prelucrarea și parsarea răspunsurilor HTTP primite de la server
util/delay.h	generarea întârzierilor scurte necesare anumitor secvențe hardware

Utilizarea bibliotecilor AVR standard a permis dezvoltarea unei aplicații embedded eficiente, cu acces low-level la registrele microcontrolerului și control direct asupra perifericelor utilizate în cadrul proiectului.

Biblioteci utilizate pentru componenta server-side

Bibliotecă	Rol / Utilizare
Flask	implementarea serverului HTTP și a infrastructurii client-server
jsonify	generarea și transmiterea răspunsurilor HTTP
render_template	implementarea interfeței web pentru simularea stărilor meteorologice

Framework-ul Flask a fost ales datorită simplității și flexibilității sale, permițând dezvoltarea rapidă a unei interfețe web utilizate pentru simularea unui serviciu meteorologic de tip ANM și transmiterea alertelor către sistemul embedded.

3.4 ALGORITMI SI STRUCTURI IMPLEMENTATE

Mașină de stări finite (FSM)

Principala structură software utilizată în cadrul proiectului este o mașină de stări finite (Finite State Machine - FSM), responsabilă de gestionarea întregii logici de funcționare a sistemului.

Utilizarea unui FSM permite separarea clară a comportamentelor aplicației și gestionarea predictibilă a tranzițiilor dintre diferitele stări ale sistemului.

Stările implementate sunt:

Stare	Rol
ST_INIT	inițializarea sistemului și configurarea perifericelor
ST_CONNECTING_WIFI	conectarea la rețeaua WiFi prin intermediul ESP8266
ST_NORMAL	funcționarea normală a sistemului
ST_ALERTA_GRINDINA	activarea mecanismului de protecție împotriva grindinei
ST_ALERTA_SOARE	retragerea prelatei în condiții normale / însorite
ST_ERROR	tratarea erorilor de comunicație sau funcționare

Implementarea FSM-ului oferă mai multe avantaje:

- control predictibil al sistemului;
- tranziții clare între stări;
- recuperare controlată în cazul erorilor;
- modularitate și extindere facilă a aplicației.

Scheduler cooperativ bazat pe task-uri

Firmware-ul utilizează un scheduler cooperativ non-blocant bazat pe execuția periodică a unor task-uri independente.

Fiecare funcționalitate importantă a sistemului este executată periodic:

- actualizarea FSM-ului;
- controlul servomotorului;
- controlul LED-urilor;
- polling-ul serverului;
- actualizarea display-ului OLED;
- debugging și monitorizare serială.

Execuția periodică este realizată utilizând timestamp-uri generate prin intermediul modulului `uptime`, fără utilizarea delay-urilor blocante.

Această abordare permite funcționarea simultană și fluidă a tuturor componentelor sistemului și îmbunătățește timpul de răspuns al aplicației.

Polling HTTP periodic

Comunicarea dintre sistemul embedded și serverul software este realizată prin interogări HTTP GET periodice efectuate de modulul ESP8266.

Serverul răspunde cu una dintre următoarele stări:

- STARE_NORMALA
- ALERTA_GRINDINA
- ALERTA_SOARE

Răspunsurile primite sunt analizate folosind funcția `strstr()`, metodă aleasă pentru:

- reducerea consumului de memorie;
- simplificarea implementării;
- creșterea vitezei de procesare pe microcontrolerul AVR.

Surse și funcții implementate

Proiectul a fost modularizat în mai multe fișiere sursă independente, fiecare responsabil pentru o anumită funcționalitate a sistemului.

Fișier	Rol
main.c	inițializarea sistemului și gestionarea scheduler-ului
app_fsm.c	implementarea logicii FSM
esp8266.c	comunicație HTTP și control ESP8266
esp_serial.cpp	comunicație serială cu modulul ESP8266
motor.c	controlul servomotorului
led.c	controlul LED-urilor de stare
buzzer.c	controlul avertizării sonore
oled.c	controlul display-ului OLED
twi.c	implementarea comunicației I2C/TWI
usart.c	debugging și comunicație serială
uptime.c	gestionarea timpului și a task-urilor periodice
server.py	implementarea serverului ANM simulat

3.5 ELEMENTUL DE NOUTATE

Elementul principal de noutate al proiectului constă în integrarea unui sistem embedded cu o infrastructură software de tip client-server pentru realizarea unei soluții automate de protecție

împotriva fenomenelor meteorologice severe. Sistemul combină comunicații WiFi, automatizări și control electromechanic, permițând detectarea alertelor meteorologice și reacția autonomă a mecanismului de protecție. Deși proiectul reprezintă un proof-of-concept realizat în context academic, arhitectura propusă evidențiază potențialul unei astfel de soluții pentru dezvoltări viitoare și integrarea cu servicii meteorologice reale.

3.6 JUSTIFICAREA UTILIZARII FUNCTIONALITATILOR DIN LABORATOR

Timere și PWM

Servomotorul utilizat pentru acționarea mecanismului de protecție este controlat folosind tehnici PWM studiate în cadrul laboratorului 3. Utilizarea PWM-ului permite controlul precis al sensului și duratei rotației servomotorului, fiind totodată o soluție eficientă din punct de vedere al resurselor hardware utilizate.

Timer-ele microcontrolerului sunt utilizate și pentru:

- generarea mecanismului de uptime;
- implementarea scheduler-ului periodic;
- controlul avertizării sonore prin buzzer.

I2C / TWI

Display-ul OLED utilizează comunicație I2C/TWI, conform conceptelor studiate în laboratorul 6. În cadrul proiectului au fost utilizate:

- magistrala SDA/SCL;
- adresarea dispozitivelor I2C;
- transmisia serială sincronă.

Utilizarea protocolului I2C a permis conectarea eficientă a display-ului utilizând un număr redus de pini ai microcontrolerului.

USART

Interfața USART este utilizată pentru:

- debugging și monitorizare serială;
- afișarea stărilor FSM;
- analiza comunicației cu modulul ESP8266.

Prin intermediul comunicației seriale au fost validate:

- comenzile AT trimise către ESP8266;
- răspunsurile HTTP primite de la server;
- tranzițiile dintre stările FSM.

Programare non-blocantă

Conceptele de programare non-blocantă și scheduling periodic utilizate în proiect au la bază utilizarea timerelor și a mecanismelor de temporizare studiate în cadrul laboratoarelor.

Această abordare permite:

- evitarea delay-urilor blocante;
- funcționarea simultană a mai multor module;
- îmbunătățirea timpului de răspuns al sistemului;
- prevenirea pierderii evenimentelor importante.

3.7 SCHELETUL PROIECTULUI SI INTERACTIUNEA COMPONENTELOR

Arhitectura software a proiectului este modulară, fiecare componentă având un rol bine definit în funcționarea sistemului. Această organizare a permis dezvoltarea, testarea și depanarea separată a modulelor hardware și software.

Fluxul principal de funcționare este următorul:

- inițializarea perifericelor;
- conectarea la rețeaua WiFi;
- interogarea periodică a serverului;
- parsarea stării meteorologice primite;
- actualizarea mașinii de stări finite;
- controlul elementelor hardware.

Firmware-ul rulează într-o buclă infinită, în care sunt executate periodic task-uri independente. Această abordare permite funcționarea simultană a modulelor sistemului fără utilizarea unor delay-uri blocante.

Interacțiunea dintre componente este următoarea:

- modulul ESP8266 primește starea curentă de la server;
- FSM-ul interpretează starea și decide comportamentul sistemului;
- servomotorul acționează mecanismul de protecție;
- display-ul OLED afișează statusul curent;
- LED-urile indică vizual starea sistemului;
- buzzer-ul avertizează acustic în cazul alertei de grindină.

Validarea funcționării

Funcționarea sistemului a fost validată incremental, prin testarea separată a fiecărei componente și apoi prin integrarea acestora în aplicația finală.

Au fost realizate următoarele teste:

- testarea individuală a servomotorului;
- testarea LED-urilor de stare;
- testarea display-ului OLED pe magistrala I2C;
- testarea buzzer-ului;
- verificarea comunicației WiFi prin ESP8266;
- verificarea polling-ului HTTP către server;
- testarea tranzițiilor FSM;
- validarea reacției sistemului pentru stările NORMAL, GRINDINĂ și SOARE.

Serverul Flask a permis simularea rapidă a diferitelor scenarii meteorologice, ceea ce a facilitat testarea comportamentului sistemului fără dependența de un serviciu meteorologic real.

3.8 CALIBRAREA ELEMENTELOR DE SENZORISTICA

În forma actuală, sistemul utilizează o „senzoristică software”, bazată pe informațiile meteorologice primite de la serverul care simulează un serviciu de tip ANM. Astfel, procesul de calibrare a constat în principal în ajustarea parametrilor software și a reacțiilor hardware ale sistemului.

Au fost calibrate:

- perioada de polling către server;
- timpii de reacție ai sistemului;
- durata mișcărilor servomotorului;
- durata avertizărilor sonore generate de buzzer;
- mesajele și timpul de actualizare al display-ului OLED.

Perioada de polling a fost stabilită la 5 secunde, valoare considerată optimă pentru:

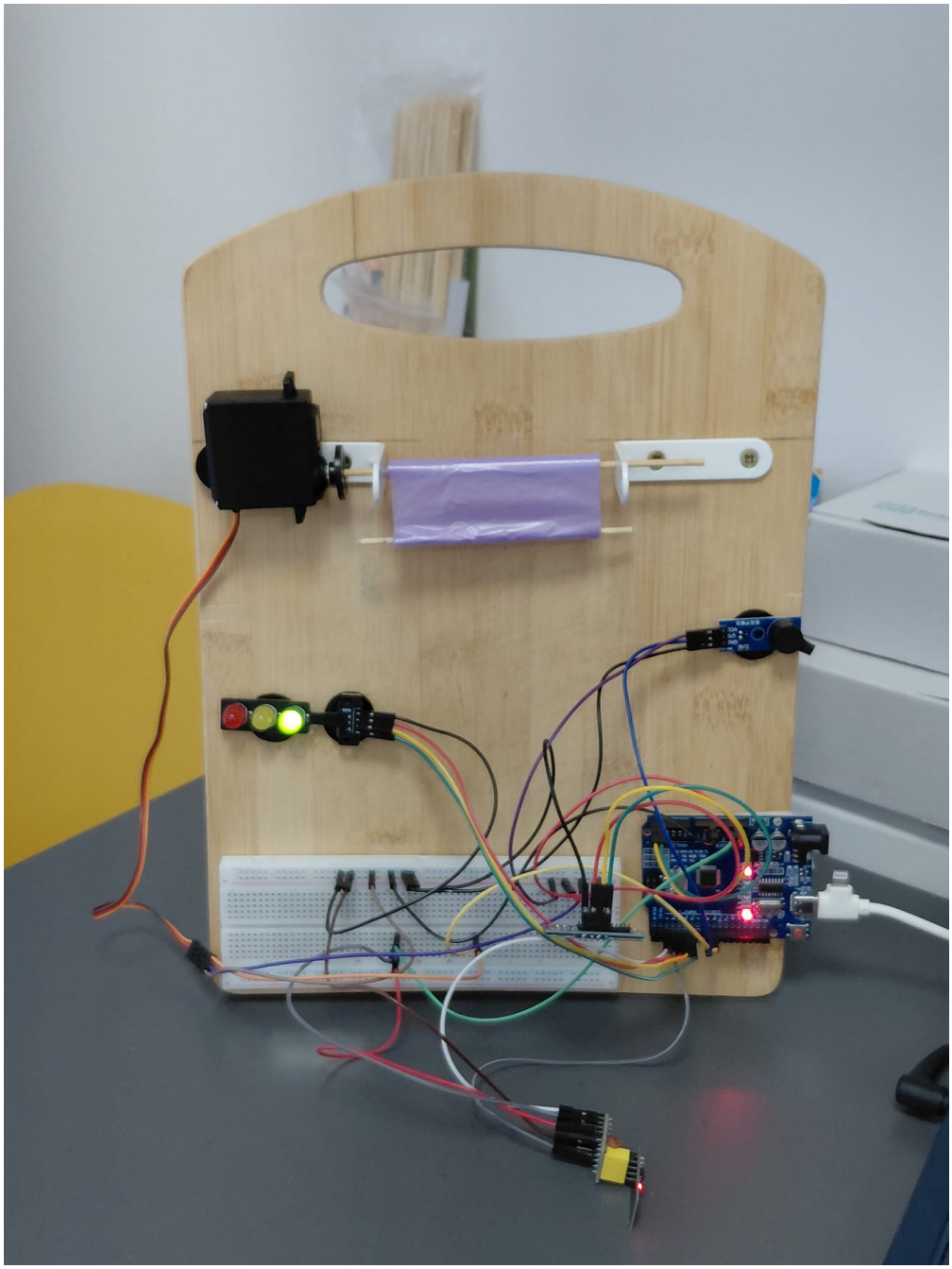
- reducerea traficului de rețea;
- menținerea stabilității comunicației;
- obținerea unui timp de reacție suficient de rapid pentru demonstrarea funcționării sistemului.

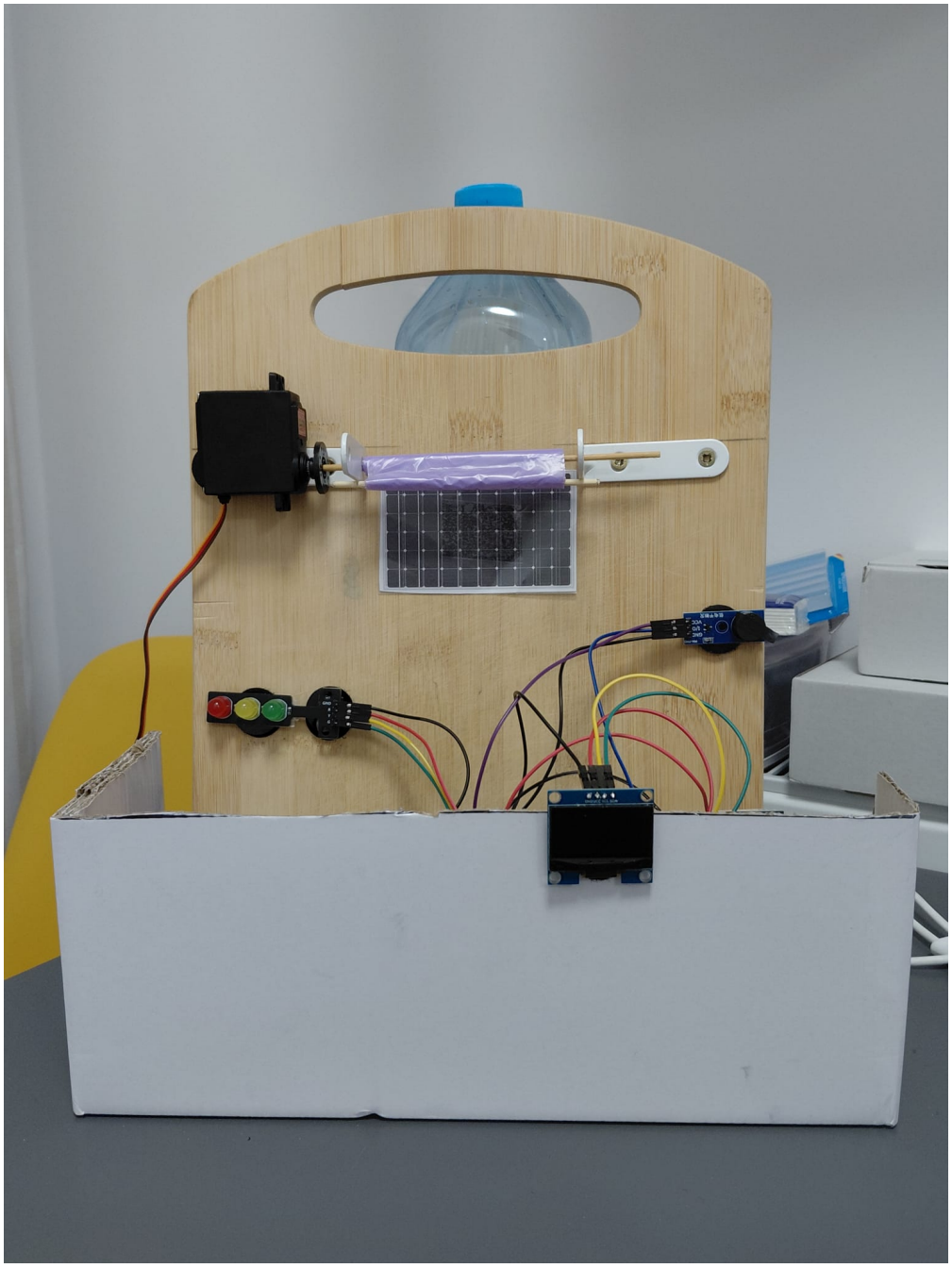
De asemenea, au fost ajustați timpii de acționare ai servomotorului astfel încât mecanismul de protecție să execute corect operațiile de deschidere și închidere, iar semnalizarea sonoră a fost limitată la intervale scurte pentru a evita funcționarea continuă și disconfortul acustic.

Rezultate Obţinute

Demo video: https://www.youtube.com/shorts/0mEXFdJo1yw

Repo GitHub: https://github.com/anadelureanu/Proiect_PM_Protectie_PanouriFotovoltaicee





Concluzii

Realizarea acestui proiect a reprezentat o experiență foarte interesantă și valoroasă, atât din punct de vedere tehnic, cât și personal. Pe parcursul dezvoltării au existat numeroase provocări și momente de incertitudine legate de funcționarea corectă a sistemului, însă satisfacția finală a fost cu atât mai mare în momentul în care toate componentele au funcționat împreună și proiectul a devenit un produs fizic complet funcțional.

Consider că unul dintre cele mai importante aspecte ale acestui proiect a fost posibilitatea de a aplica în practică noțiunile studiate în cadrul laboratoarelor și cursurilor de microprocesoare, observând direct modul în care conceptele teoretice se transformă într-un sistem real.

Îmi doresc ca pe viitor să dezvolt această idee în continuare, prin integrarea unor senzori reali și a

unor servicii meteorologice oficiale, deoarece consider că o astfel de soluție are potențial pentru aplicații practice reale.

PS: multumiri speciale lui Gabiii, Stefi si Nelutu pentru prinderea cadrului cu bormasina si Ioanei pentru cele mai precise lipituri cu fludor

Bibliografie/Resurse

Resurse hardware

- Arduino UNO R3 (ATmega328P);
- ESP8266 ESP-01;
- Display OLED 1.3” I2C;
- Servomotor MG996R 360°;
- LED-uri și buzzer activ.

Resurse software

- Visual Studio Code;
- PlatformIO;
- avr-gcc;
- Python 3;
- Flask Framework;
- Documentația oficială ATmega328P;
- Documentația ESP8266 AT Commands.

Resurse financiare

- mama & tata