F.O.C. (Fire Observing Contraption)

Nume: Vlad Costin Andrei

Grupa: 331 CD

Introducere

Ce face

Este un sistem de alarma complet, prevazut cu un dispozitiv de detectare a surselor de incendiu si informarea utilizatorului prin alerte vizuale, auditive si prin intermediul platformei web accesibila atat pe laptop cat si pe mobil. Sistemul va informa in mod constant utilizatorul in privinta umiditatii si a temperaturii din incaperea in care este plasat.

Care este scopul lui

Scopul proiectului este testarea cunostintelor de PM invatate in cadrul laboratorului si punerea lor in practica. In cadrul acestui proiect, vom intercepta semnalele trimise de anumiti senzori si vom interactiona cu un ecran, un buzzer si vom trimite un mesaj printr-o platforma web.

Care a fost ideea de la care aţi pornit

Ideea de baza de la care m-am inspirat este un sistem de alarma de incendiu modern, similar sistemelor smart-home, cu care utilizatorul poate interactiona prin intermediul telefonului.

De ce credeţi că este util pentru alţii şi pentru voi

Un astfel de sistem de alarma prezinta o utilitate bonus fata de alte sisteme de alarma invechite, si anume abilitatea de a tine utilizatorul la curent printr-o platforma web accesibila din smartphone.

Descriere generală

Sistemul F.O.C. se bazeaza pe un microcontroler ESP32 cu acces la WiFi si Bluetooth si foloseste o combinatie de senzori (senzor de flacara/lumina, senzor de temperatura si umiditate, senzor de gaz) si elemente de ouptut vizuale si auditive (ecran LCD, buzzer) pentru a detecta potentiale incendii in cadrul unei locuinte.

In mod constant, senzorii vor receptiona diverse valori din incaperea in care este plasata alarma si va trimite rezultatele respective spre placuta ESP32. In acelasi timp, valorile respective sunt transmise atat catre serverul web hostat local, cat si catre placuta LCD pentru a fi accesibile utilizatorului.

In cazul detectiei unei anomalii: lumina foarte puternica, gaz metan, temperatura foarte ridicata, senzorii trimit un mesaj de alarma, actionand simultan buzzer-ul pentru a alerta persoanele din jurul alarmei sau cei ce verifica pagina web.

Schema bloc

Elemente de noutate ale proiectului

Fata de alte proiecte similare, am incercat sa incapsulez toate functionalitatile oferite de un sistem de alarma traditional intr-un singur produs, in loc sa creez un spatiu de test (ex. o casuta/camera in care sa montez senzorii), aducand astfel proiectul cat mai aproape de realitate.

Laboratoare folosite

- **USART** : https://ocw.cs.pub.ro/courses/pm/lab/lab1-2023 (Folosim USART pentru debugging si pentru analizarea real-time a datelor inregistrate de senzori)

- **Intreruperi** : https://ocw.cs.pub.ro/courses/pm/lab/lab2-2023 (Folosim intreruperi pentru butonul de reset/oprire al alarmei)

- **I2C** : https://ocw.cs.pub.ro/courses/pm/lab/lab6-2023-2024 (Folosim I2C in comunicarea cu ecranul LCD)

Hardware Design

Listă de piese

- Placa dezvoltare WiFi + Bluetooth ESP32

- Modul Buzzer Pasiv Piezo

- Ecran LCD 1602 I2C

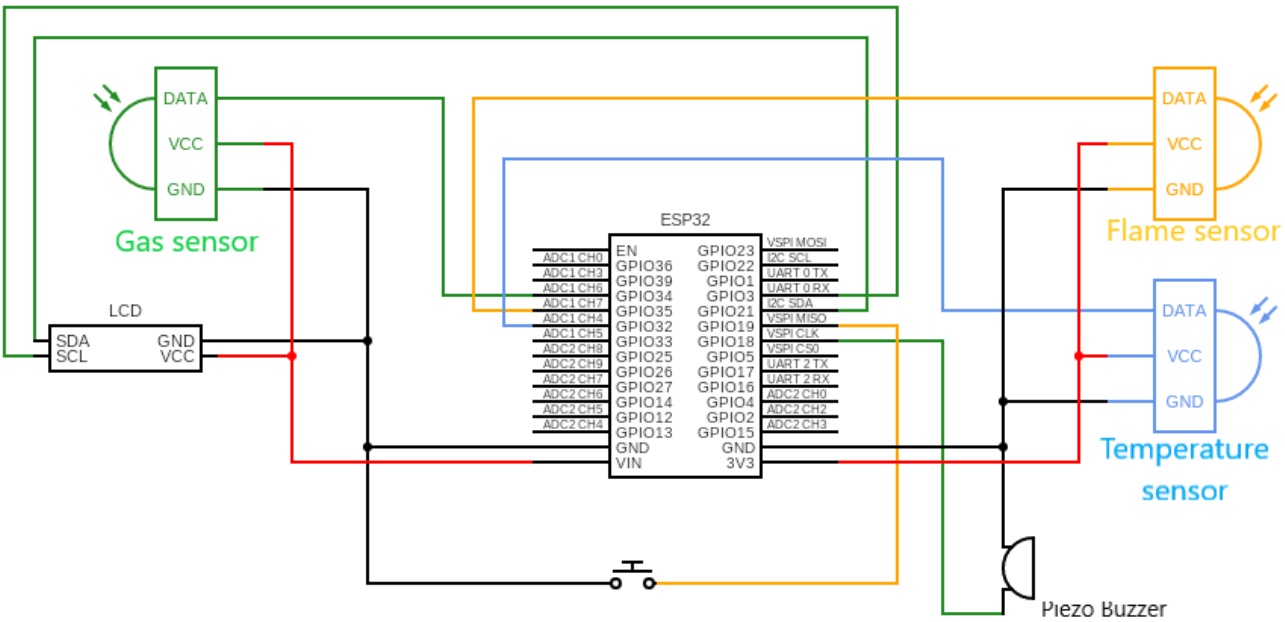
- Senzor de temperatura si umiditate DHT22

- Senzor de foc si lumina

- Senzor detector de gaz MQ2

- Push button

Scheme electrice



Calibrare senzori

In cadrul proiectului am folosit 3 senzori, de lumina, de gaz si de temperatura + umiditate. Pentru a folosi acesti senzori avem nevoie sa intelegem ce valori pot inregistra acestia si cum sa ajustam potentiometrul integrat fiecarui senzor pentru a obtine input valid in orice mediu (pe cat posibil).

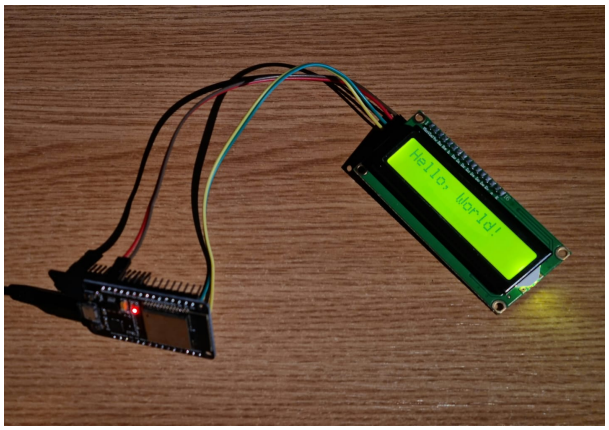
- **Senzorul de gaz MQ2** : Senzorul poate inregistra valori in intervalul [0, 4095] ppm. O valoare peste medie ar putea indica prezenta gazului metan in aer la un nivel suficient de semnificativ pentru a prezenta un pericol de incendiu. Pentru a obtine o valoare mai stabila (tinem cont de mici variatii din aer), vom normaliza valorile in intervalul [0, 100], asteptandu-ne sa depistam valori peste 50ppm pentru a declansa alarma.

- **Senzorul de flacara (lumina)** : Similar cu senzorul de gaz, doar ca de aceasta data cautam valori care in urma normalizarii se afla in intervalul [0, 10]. Calibrarea acestui senzor a fost mai greu de realizat daatorita conditiilor luminoase diferite din fiecare incapere in care am testat senzorul.

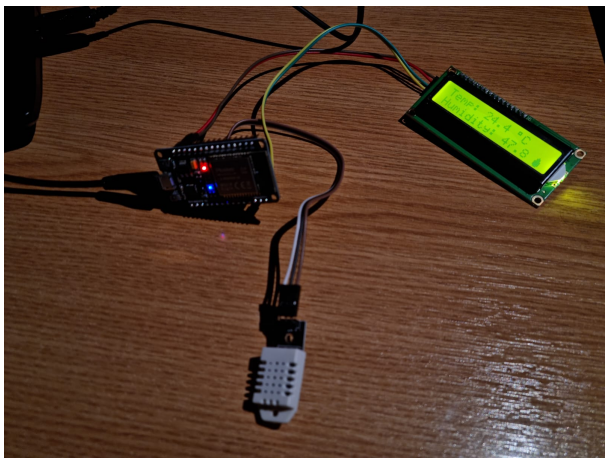
- **Senzorul de temperatura si umiditate** : Acest senzor nu trebuie calibrat, vom lucra direct cu valorile inregistrate de acesta intrucat in urma unor teste (cu un termometru de camera), rezultatele s-au dovedit precise.

Etape de dezvoltare

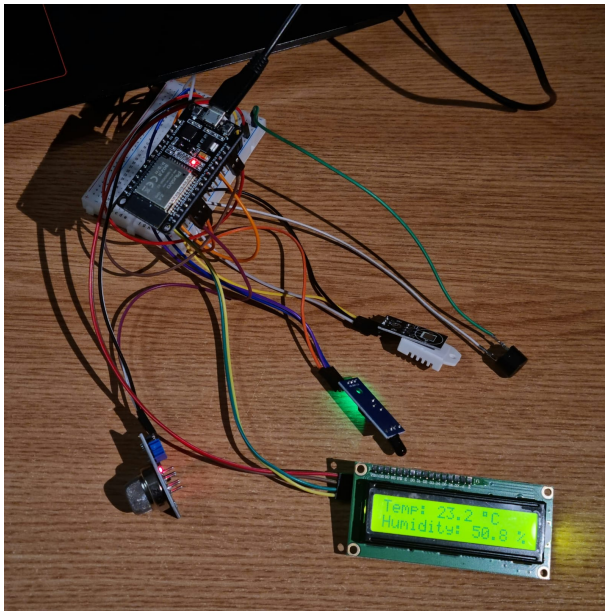
Testare ecran LCD



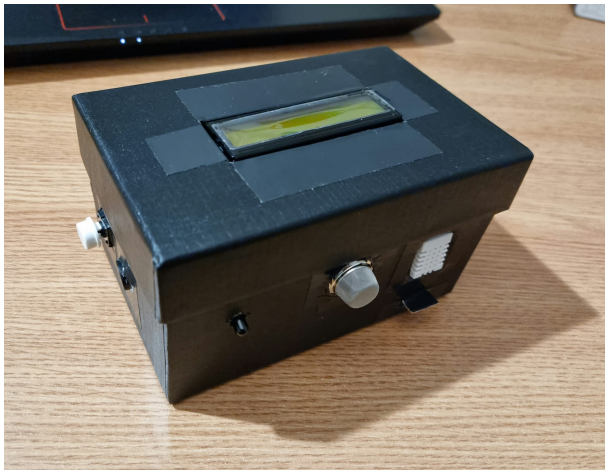
Testare senzor de temperatura si umiditate

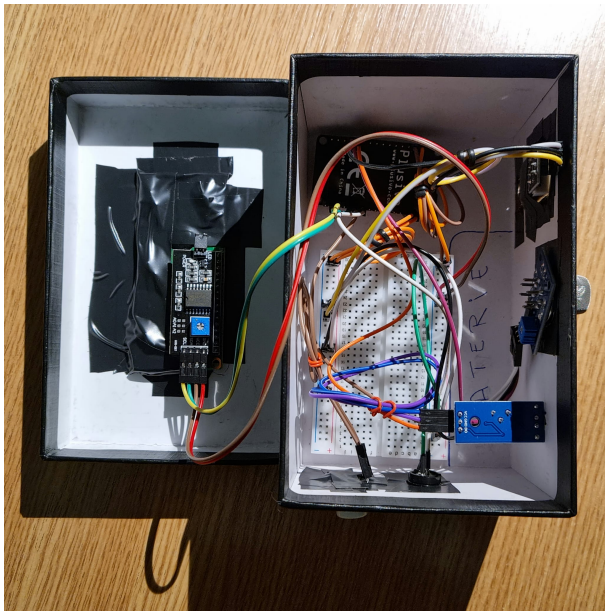


Asamblarea senzorilor prin breadboard



Produs final





Demo produs final

F.O.C. Demo

Software Design

Mediu de dezvoltare

Pentru proiectarea placutei ESP32 am folosit Arduino IDE, ajutandu-ma de diverse funcii si librarii.

Librarii

- **“DHT.h”** : librarie pendru diversi senzori DHT (DHT11, DHT22, etc.) de temperatura si/sau umiditate; folosita pentru a interactiona cu senzorul DHT22 (pentru a citi valorile oferite de el)

- **“LiquidCrystal_I2C.h”** : o librarie pentru ecrane LCD cu I2C, cu diverse functionalitati pentru afisaj (pozitionare cursor, crearea caracterelor custom, etc.)

- **“WiFi.h”** : Permite conexiunea la retea (locala si internet). De asemenea, putem instantia servere, clienti si trimite/primi pachete UDP prin WiFi. Adresa IP poate fi atribuită static sau printr-un DHCP. Biblioteca poate gestiona și DNS

- **“WiFiClient.h”** : Librarie pentru clienti ce se pot conecta la o anumita adresa IP

- **“WebServer.h”** : Librarie pentru clienti compatibila pe diverse placute, ce suporta cereri HTTP de GET si POST, folosita pentru a interactiona cu pagina web

- **“ESPmDNS.h”** : Librarie compatibila cu placuta ESP32 ce perite adresarea paginii web hostata local prin domeniul /esp32.local/

Functii implementate

- **setup()** : functie care initializeaza diverse componente, server, etc.

Initializarea senzorului DHT {

```
dht.begin()
```

}

Initializarea ecranului LCD {

```
lcd.init()
lcd.backlight()
```

}

Initializarea serverului prin DNS {

```
WiFi.begin(ssid, password)
MDNS.begin("esp32")
server.begin()
```

}

Initializarea butonului si asocierea intreruperii {

```
pinMode(RESET_BUTTON_PIN, INPUT_PULLUP);
touchAttachInterrupt(RESET_BUTTON_PIN, handle_reset, FALLING);
```

}

- **IRAM_ATTR handle_reset()** : handlerul de intrerupere generata de apasarea butonului. Cand butonul este apasat, dorim setarea valiabilei **fire** pe false, echivalent cu resetarea alarmei. Functia implementeaza si un mecanism de debouncing cu un delay de 1000ms.

- **update_lcd()** : functie care actualizeaza informatiile listate pe ecranul LCD. In functie de valoarea variabilei globale booleene **fire**, ecranul LCD va afisa 2 tipuri de informatii - un mesaj de alerta sau informatii despre temperatura si umiditatea din incapere

- **update_alarm()** : functie care, in situatia in care valoarea variabilei globale booleene **fire** este **true** (a fost detectat o sursa de incendiu), actioneaza buzzerul pentru a emite un semnal de alarma

- **handleRoot()** : functie care actualizeaza informatiile listate pe pagina web. In functie de valoarea variabilei globale booleene **fire**, vom trimite doua serverului doua tipuri diferite de secvente de cod HTML - un mesaj de alerta sau informatii despre temperatura si umiditatea din incapere

- **check_fire()** : functie care actualizeaza valoarea variabilei globale **fire**. In functie de inputul receptionat de senzori (de gaz si flacara), verificam daca valorile inregistrate depasesc anumiti parametrii prestabiliti, caz in care setam variabila **fire** pe **true**

- **check_reset()** : functie care actualizeaza valoarea variabilei globale **fire**. In cazul in care detectam o schimbare a semnalului digital de la **HIGH** la **LOW** (butonul a fost apasat), resetam alarma prin setarea variabilei **fire** pe **false**.

- **loop()** : functie care se ocupa de functionarea alarmei de incendiu - actualizeaza constant informatiile trimise catre ecranul LCD si catre server. In cazul in care valorile citite de senzorii de gaz si de flacara depasesc anumite limite (echivalent cu detectia unei surse de incendiu), se schimba valoarea variabilei **fire**

Reprezentare logica prin diagrama UML

Optimizari

Pentru realizarea proiectului, am folosit un ESP32 (doar un ESP32, nu si o placuta Arduino), deoarece prezenta toate functiile necesare, si anume conectivitate la internet. Astfel, evitam impartirea codului in doua fisiere separate si salvam timp prin comunicare directa intre senzori si ESP32 (fata de situatia in care trebuia sa transmitem anumite date intre ESP si Arduino).

Concluzii

Proiectul realizat demonstreaza cu succes utilizarea unui modul ESP32 si a unor senzori (de temperatura, lumina si gaz) pentru dezvoltarea unei alarme de incendiu eficiente. Implementarea si functionalitatea proiectului evidentiaza versatilitatea si puterea de procesare a modulului ESP32 in aplicatii ce tin de securitatea locuintei.

Download

foc.zip

Jurnal

Inceperea proiectului (5 mai)

Realizarea documentarii despre componentele necesare realizarii proiectului si intocmirea paginii wiki cu informatii generale / sumare.

Inceperea etapei hardware (10 mai)

Am primit piesele necesare realizarii proiectului. Am inceput prin testarea fiecarui, cat si a ecranului LCD si a buzzer-ului.

Inceperea etapei software (11 mai)

Odata de am verificat functionalitatea tuturor componentelor, am realizat serverul web, urmand ca mai apoi sa conectez toate piesele prin intermediului unui breadboard.

Modificari / retusari ale proiectului (1) (17 mai)

Am mai adaugat un push button pentru a adauga o functionalitate suplimentara alarmei, si anume cea de reset. Cumva, nu mai merge ecranul LCD. Mai am de lucru.

Modificari / retusari ale proiectului (2) (18 mai)

Conectasem LCD-ul la 3.3V in loc de 5V prin VIN. Proiectul este aproape finalizat, ramanand sa gasesc o carcasa pentru alarma si sa setez un sunet corespunzator pentru alarma (momentan folosesc o serie de note muzicale alese la intamplare pentru a testa functionalitatea buzzer-ului).

Bibliografie/Resurse

- ESP32 Datasheet

Export to PDF