Computer de bord

Introducere

Autor: Rareș Constantin - 332CA

Proiectul reprezintă un computer simplu de bord al unei mașini, care monitorizează diverși factori externi/interni, cum ar fi nivelul de umiditate pe suprafața parbrizului, temperatura și presiunea la nivelul roților, senzori de proximitate, iar apoi controlează diverse funcții ale mașinii, precum intensitatea luminoasă a farurilor, viteza ștergătoarelor și avertizor sonor pentru parcare, dar și să afișeze într-un mod user friendly aceste date pe un ecran LCD. M-am gândit că acest proiect ar putea exemplifica foarte bine utilizarea unor senzori și a folosi aceste date pentru a modifica starea sistemului implementat. Acest proiect ar fi util în industria auto prin eliminarea cablurilor pentru comunicare, folosind o soluție wireless detaliată mai jos.

Descriere generală

Am gândit acest proiect ca o interacțiune dintre o placă de achiziție (ESP32) și placă de acționare (ESP32). Astfel, pot implementa propriul protocol de comunicație prin ESP NOW care poate suporta o eventuala securitate și elimină în același timp necesitatea cablajului intern care îngreunează mașina.

De asemenea, prin folosirea a două plăci diferite se compartimentează funcționalitățile, iar implementările pot fi menținute și updatate intern fără modificarea protocolului de comunicație, astfel putând folosi plăci cu implementări diferite interne și atașând mai multe plăci în interiorul sistemului dacă este nevoie.

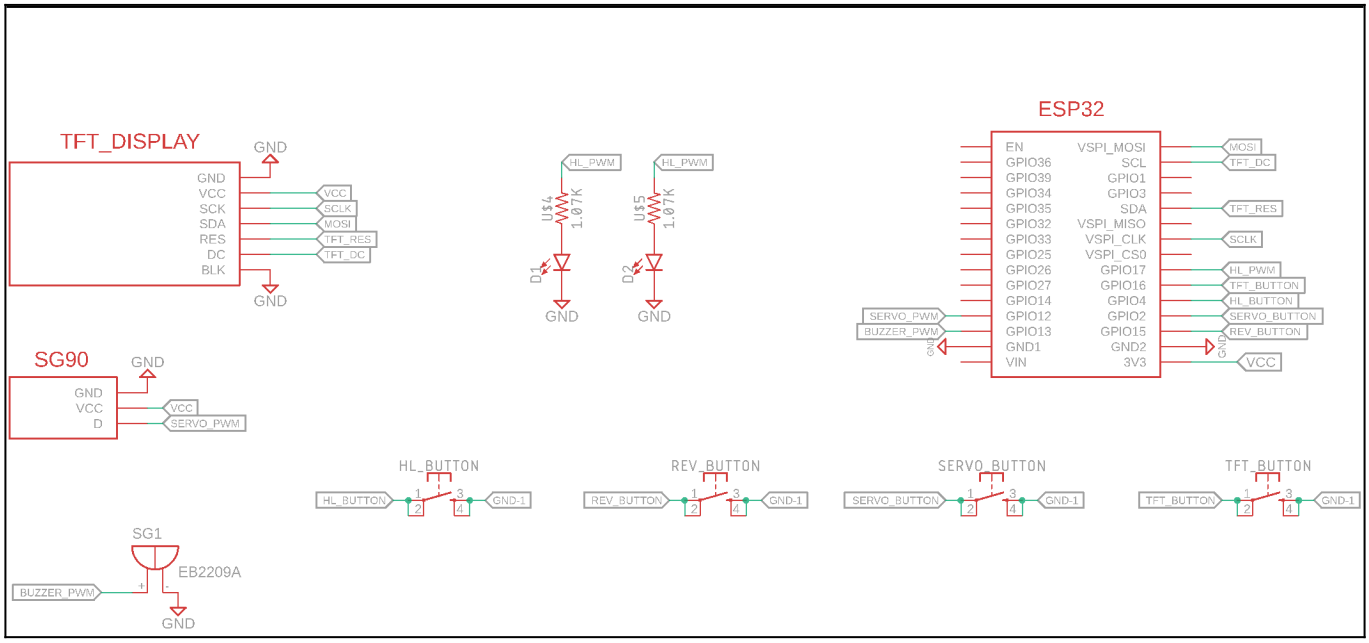
Utilizatorul poate controla sistemul prin intermediul butoanelor (mock-up pentru un sistem de butoane fizice pe bordul mașinii), poate interoga datele date de senzori și starea curentă a sistemului (intensitate faruri, viteză ștergătoare etc.) prin ecranul LCD.

Legendă: **BMP280** - senzor temperatură/presiune, **HC-SR04** - senzor de proximitate ultrasonic, **YL-83** - senzor umiditate la suprafață/ploaie

Hardware Design

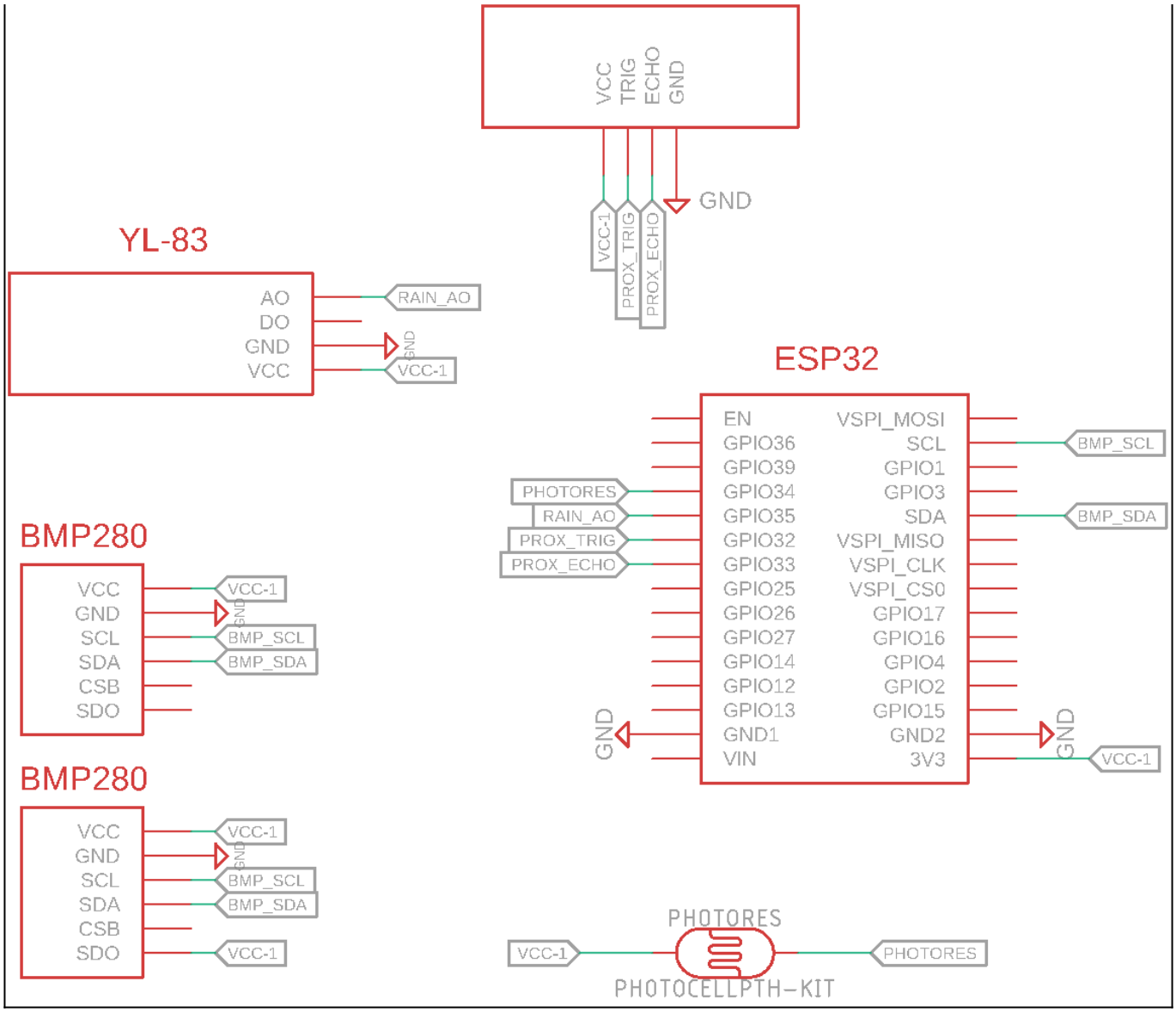
ESP32 - Action board schematic

Mai jos se regăsește schema electrică pentru placa care se va ocupa de acționarea diverselor module în funcție de datele venite de la placa de achiziție.



ESP32 - Acquisition board schematic

Mai jos se regăsește schema electrică pentru placa de achiziție care are mai mulți senzori conectați și va aștepta cereri de la placa de control pentru a trimite datele sau va trimite date de urgență în cazul unei avarii.



Software Design

Am folosit PlatformIO (VSC extension), fiind o alternativa foarte buna pentru Arduino IDE. Peste framework-ul standard de ESP32-DevKit am adaugat mai multe librarii pe care le-am folosit pentru a comunica cu perifericele. Lista se poate gasi pe repo-ul GitHub al proiectului lasat mai jos sau la urmatorul link, care reprezinta fisierul de configuratie al proiectului, in sectiune `lib_deps`. De asemenea, am adaugat o librarie proprie, si anume protocolul de comunicatie comun celor doua placi, acesta va fi explicat mai jos.

Protocolul de comuncatie se bazeaza pe un SERDES simplu de pachete, care la nivelul inferior contine tipul de transmisie, care poate reprezenta atat un request al placii de actiune, cat si un response de la placa de achizitie sau chiar o transmisie de urgenta de la aceasta in cazul unor valori extreme de la senzori, in principal cei de temperatura/presiune care s-ar ocupa de pneuri. Apoi se adauga lungimea mesajului si datele in sine. Apoi, la nivelul urmator al protocolului, se serializeaza in cadrul de date informatiile venite de la senzori intr-o structura care suporta mai multi senzori de temperatura, fiind modular, deci pot fi adaugati senzori de temperatura. De asemenea aceste structuri pot fi modificate si implementate functii de serializare pentru a suporta mai multe tipuri de senzori.

Pentru implementarea procesului de comunicare am folosit un timer care va genera o intrerupere atunci cand ajunge sa numere 800 milisecunde, cand va seta un flag pentru a face request catre placa de achizitie. Toata comunicatia este implementata peste ESP Now, protocol proprietar Espressif, care suporta comunicatie rapida si implementare de callbacks pentru transmisie si receptie de date, putand adauga asa numiti peers, adrese MAC de placi cu care poate comunica, astfel se poate face pairing la producator pentru cele doua placi.

In partea de control al perifericelor pe placa de actiune, toate sunt controlate prin intreruperi date de butoanele conectate la GPIO pins, lasand placa sa isi desfasoare afisarea datelor pe display. Legat de display, implementarea nu face refresh la tot ecranul atat timp cat ecranul nu a fost schimbat de catre utilizator, salvand astfel performanta si lasand ce este rendered la un anumit timp pe ecran si facand update doar la datele afisate efectiv.

Perifericele de actiune (servo, buzzer si faruri) sunt controlate in functie de valorile date de senzori.

Am folosit un display TFT ST7789, folosind protocolul SPI, pentru a putea suporta schimbarile rapide de date care vin de la senzori si pentru a putea arata user-ului aproape in timp real controlul sistemului. Pentru senzorii de temperatura am folosit I2C, fiind usor sa adaugi senzori pe alt canal si suportand viteze decente de transmisie. De asemenea, am folosit si lucruri mai simple, cum sunt GPIO pins si ADC pentru senzori. Calibrarea senzorilor de temperatura, fiind integrata in chip-ul on board.

Optimizari am realizat atat in zona protocolului de comunicatie, trimitand doar cantitatea necesara de data, cat si in zona display-ului, care va face refresh doar la partea de ecran care se schimba si doar atunci cand si se schimba afisajul refresh la tot display-ul.

Github repo

Rezultate Obţinute

Aici gasiti demo-ul proiectului, nu este in forma finala pentru PM Fair, arata doar functionalitatile:

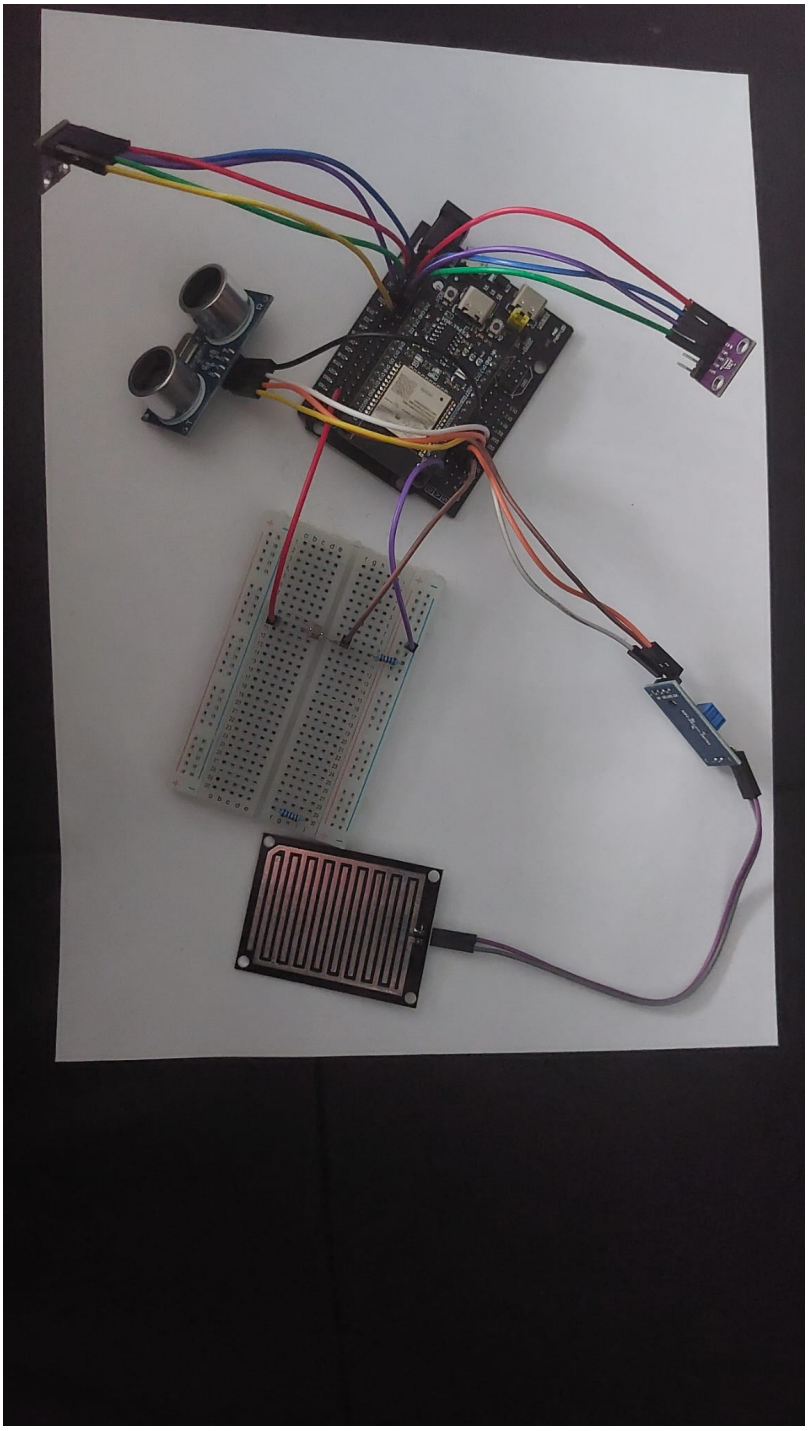
Demo proiect

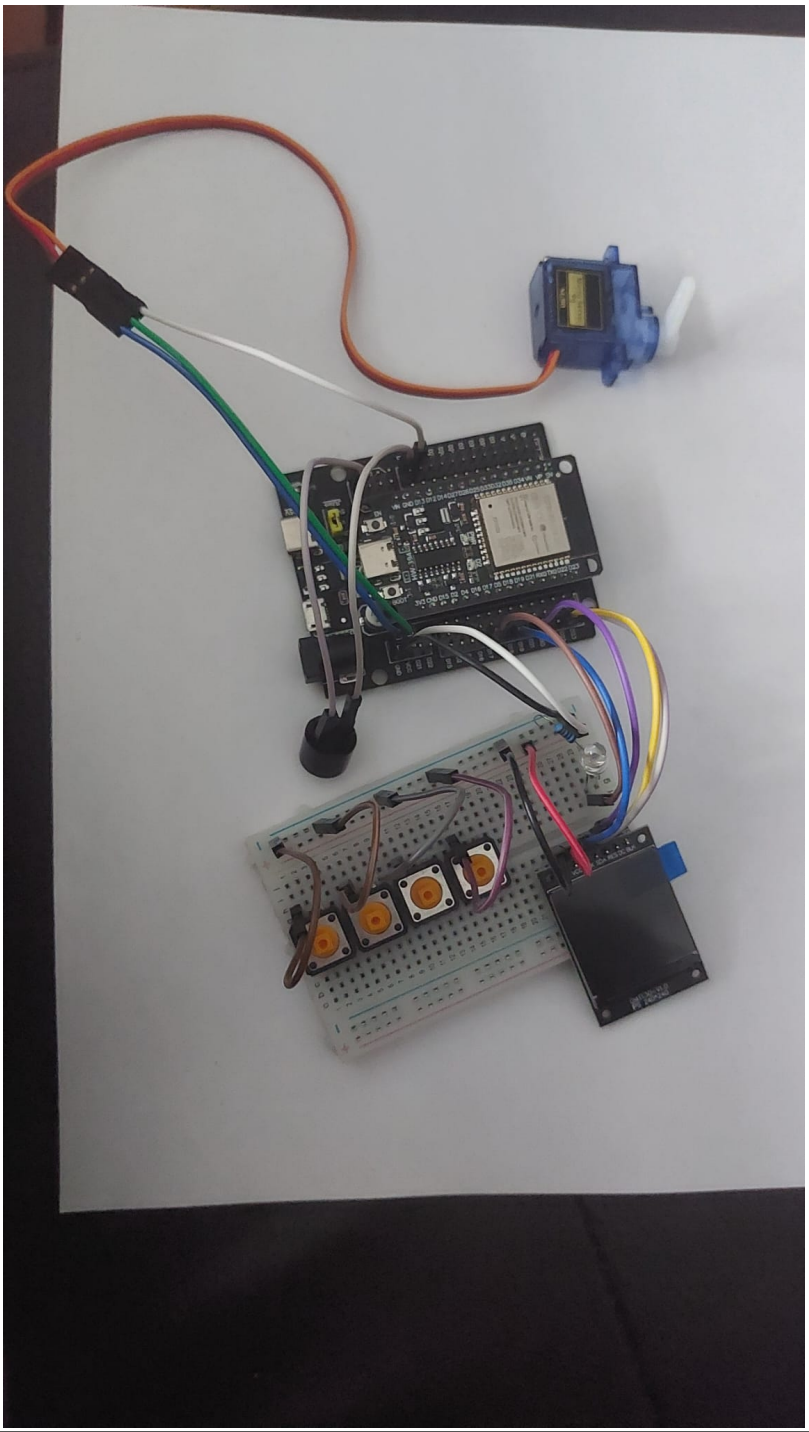
Concluzii

Jurnal

- 15.05.2024 - Am terminat cablajul pentru placa de achiziție, în principiu este final, cablajul pentru placa de acționare nu este încă complet, butoanele nefiind conectate la placă (aștept un set de cabluri mai scurte pentru a conecta mai ușor butoanele la GND pe breadboard). Am adăugat și repo-ul de Github în secțiunea Software, unde se poate găsi și un test simplu pentru placa de achiziție

```
light: 25
bmp1: 100843.718750 23.059999
bmp2: 101079.562500 25.010000
light: 23
bmp1: 100843.078125 23.070000
bmp2: 101076.101562 25.020000
light: 25
bmp1: 100842.968750 23.070000
bmp2: 101081.984375 25.030001
light: 25
bmp1: 100846.398438 23.070000
bmp2: 101078.156250 25.040001
light: 23
```





- 21.05.2024 - Am adaugat in cod tot protocolul de comunicatie modular, care poate fi folosit pentru diverse tipuri de senzori si numar variabil al acestora, deocamdata este o implementare specifica pentru senzorii pe care ii am atasati, dar poate fi usor schimbata sau adaugat cod pentru a suporta alte lucuri, am adaugat si codul pentru display (deocamdata este implementat doar modul de display al temperaturii, care este acum un placeholder, o sa-l inlocuiesc), codul pentru actionarea buzzerului in functie de distanta data de placa de achizitie. Peste tot am incercat sa folosesc intreruperi si flag-uri, codul fiind foarte robust, nu am avut probleme pana acum de sincronizare, fiind mai importanta rapiditatea primirii datelor actuale, nu neaparat procesarea tuturor datelor. Pe GitHub se pot regasi commit-urile noi, unde am lasat descrieri pentru a fi usor de track-uit progresul

Temperature display demo
Buzzer control demo

Bibliografie/Resurse

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

Export to PDF

From:
http://ocw.cs.pub.ro/courses/ - **CS Open CourseWare**

Permanent link:
http://ocw.cs.pub.ro/courses/pm/prj2024/mdinica/rares.constantin02

Last update: **2024/05/25 09:01**