Old School Game Console

Nume: Mihai-Catalin Stan

Grupa: 335CA

Introducere

Proiectul va consta intr-o consola de jocuri inspirata de produsul de mai jos, dar si de consola ArduBoy. Pe aceasta va rula un joc retro clasic in stil Chicken Invaders, cu muzica, logs, mecanica de tilting si un modul de ceas pentru surprize regulate.

Dupa cum spune si numele, scopul proiectului este de a crea un produs fizic, portabil si cu un design de moda veche, suficient de complex incat sa acopere concepte din 3 laboratoare si 3 componente mai complexe/senzori.

Ideea de la care am pornit este o dezbatere nostalgica recenta pe care am avut-o, la care s-a adus vorba de consola din poza de mai sus. Atunci am realizat ca mi-ar placea sa simt atat nostalgia vechii console cat si satisfactia de a o crea cu propriile maini. Cred ca altora le-ar fi utila deoarece pe langa nostalgia adusa de aceasta exista elemente noi care ar putea captiva temporar utilizatorul. (mecanica de tilting si reward-urile zilnice)

Descriere generală

Hardware Design

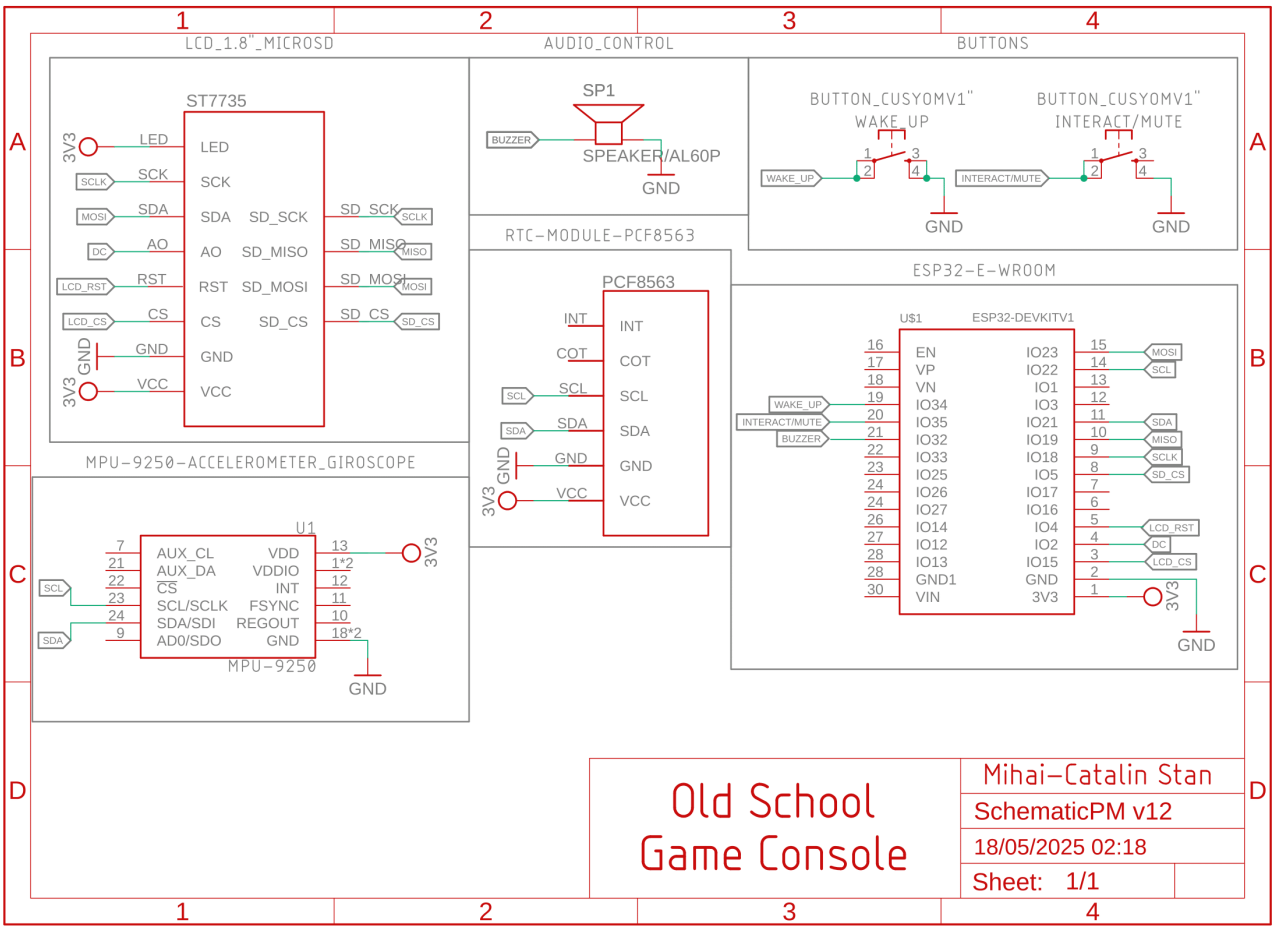
Intrucat ideea de la care am plecat este destul de simpla, am decis sa adaug si alte module pentru a oferi functionalitati in plus. Astfel, proiectul in varianta finala va avea urmatoarele componente:

- Placa de dezvoltare ESP32
- Fire
- Rezistene
- Diode

- LED RGB
- Butoane
- Buzzer
- Display LCD
- Modul RTC
- Card SD
- Accelerometru si giroscop

Proiectul foloseste concepte din laboratoarele:

- GPIO (LED/Buzzer)
- Intreruperi (Prin butoane)
- Timere
- SPI (Card SD)
- I2C (RTC, Giroscop & Accelerometru)



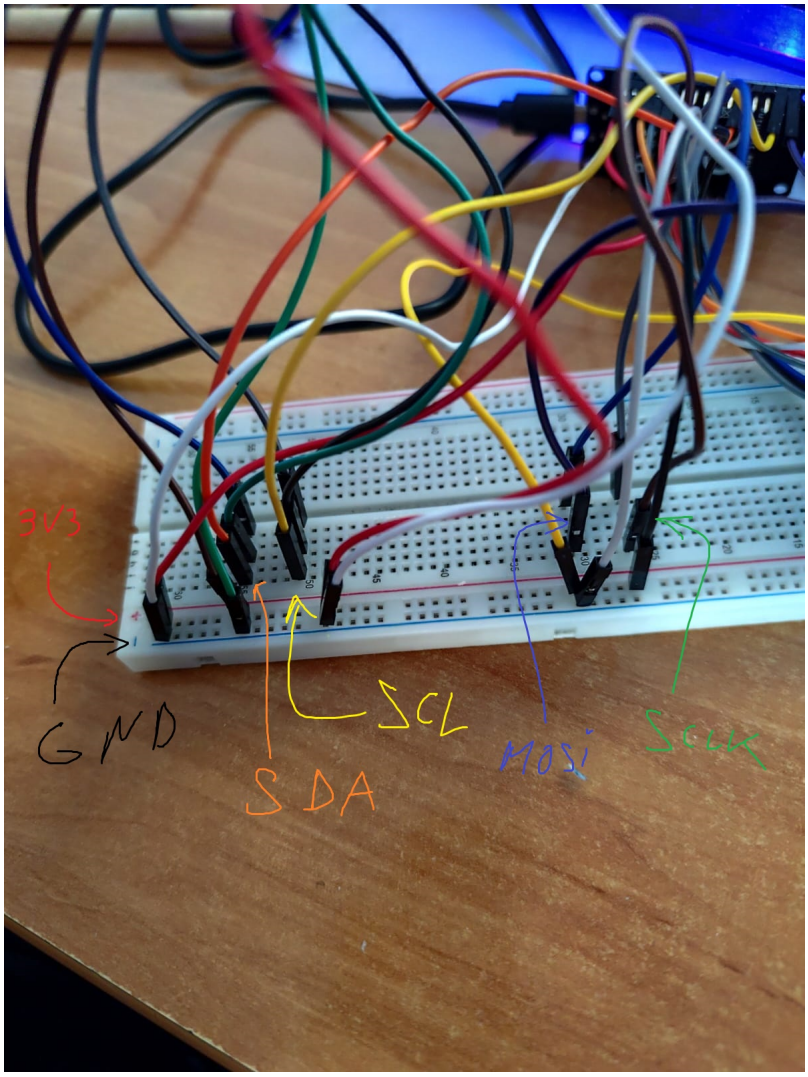
Nume Componenta	Site	Datasheet
ESP32-DEVKIT-V1	OptimusDigital	Datasheet ESP32
ST7735 cu SD card	ArduShop Display	Datasheet ST7735
RTC PCF8563	ArduShop RTC	Datasheet RTC
Accelerometru cu Giroscop MPU6500	Ardushop MPU6500	Datasheet MPU6500

Nume	Protocol	Port
ST7735	SPI	CS - GPIO15, MOSI - GPIO23, SCK - GPIO18, RESET - GPIO4, DC - GPIO2
CardSD	SPI	CS - GPIO5, MOSI - GPIO23, MISO - GPIO19, SCK - GPIO18
PCF8563	I2C	SDA - GPIO21, SCL - GPIO22

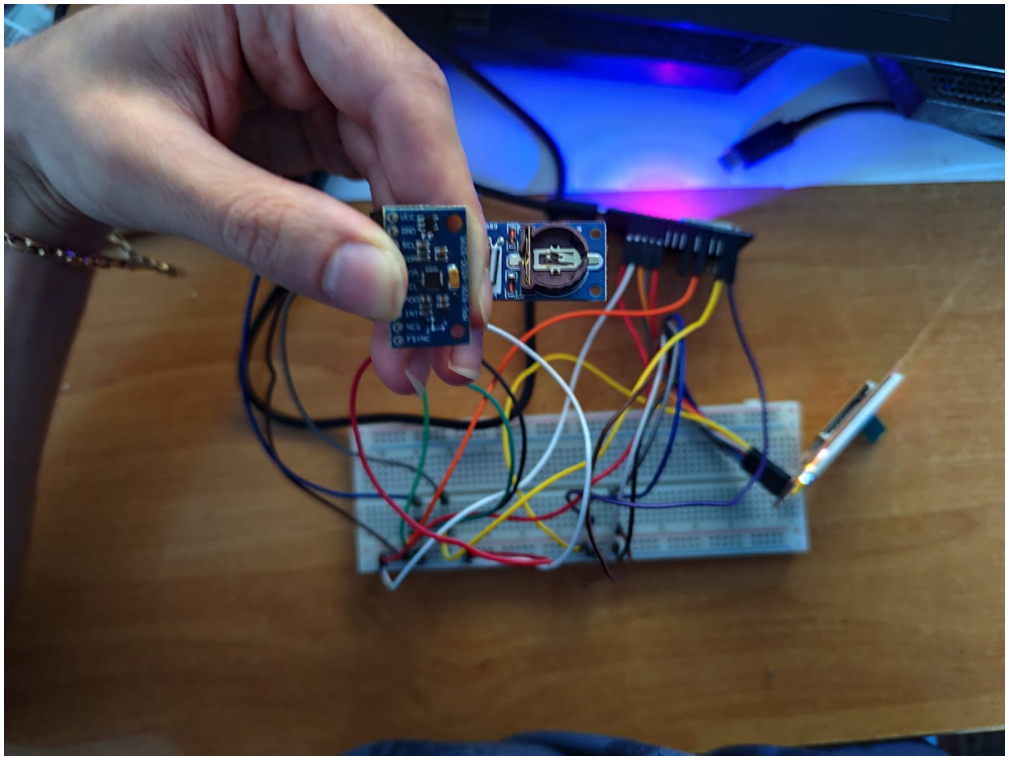
MPU6500	I2C	SDA - GPIO21, SCL - GPIO22

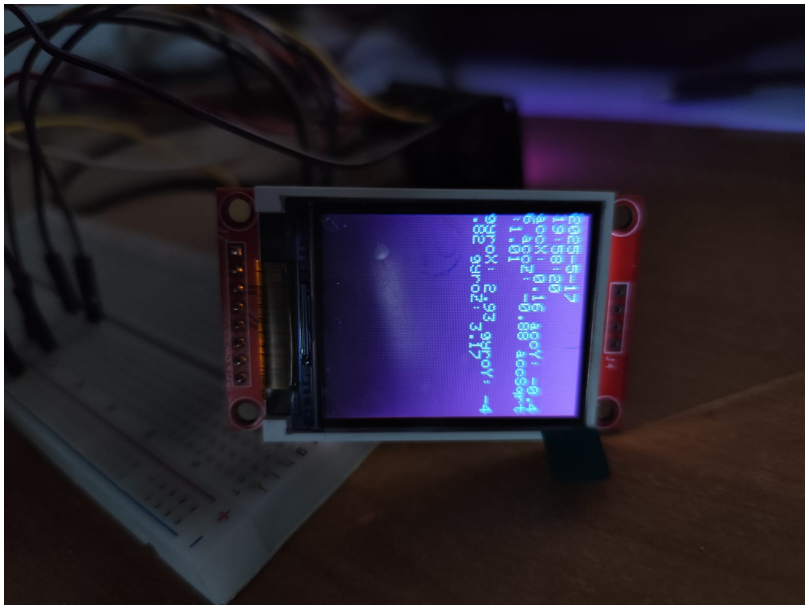
Observatie: Display-ul desi este pe SPI, nu foloseste MISO deoarece scopul sau nu este sa transmita datele catre controller. De asemenea, deoarece semnalul de RST trebuie tinut pe HIGH, acesta poate fi conectat intr-un update ulterior la 3V3.

Protocoalele mentionate mai sus pot fi vazute explicit pe breadboard in faza de testare a componentelor

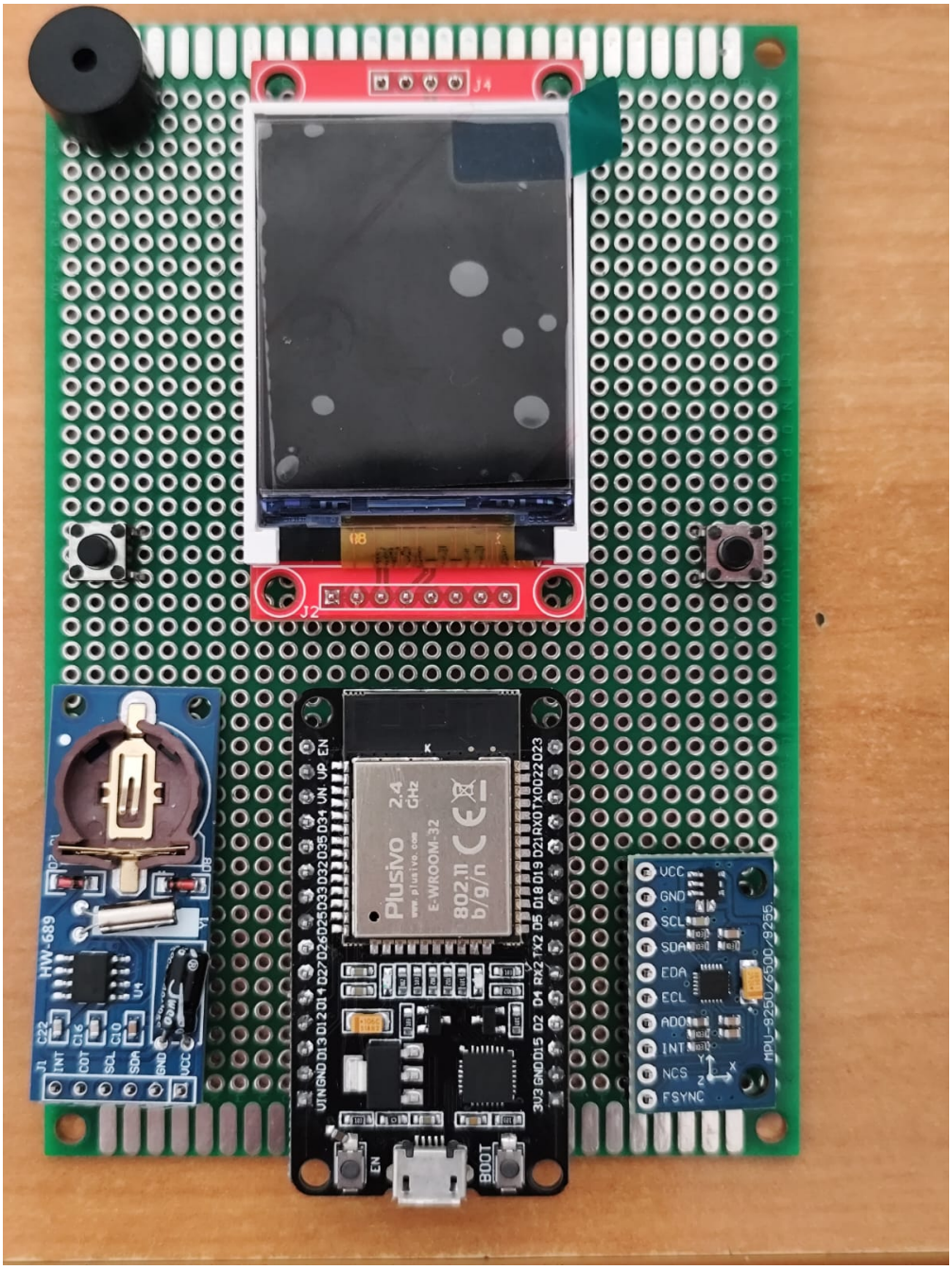


Urmatoarele poze prezinta conectivitatea modulului de ceas si a giroscopului, dar si functionalitatea acestora impreuna cu cea a display-ului. Am afisat pe display data si timpul primit de pe modulul de ceas dupa ce l-am setat la data si ora de pe sistem, si valorile primite de accelerometru si giroscop, inca necalibrate.





Nu in ultimul rand, pe pcb-ul de prototipare aranjamentul pieselor - dupa lipituri - va arata astfel:



Software Design

Stadiu milestone software:

- Un joc inspirat classic, retro, inspirat dupa Chicken Invaders, a fost implementat
- Interactiunea utilizator-giroscop-joc a fost implementata, prin interfata I2C
- Interactiunea utilizator-buton-joc a fost implementata prin pini de GPIO cu intreruperi
- Interactiunea joc-modul de ceas a fost implementata prin I2C si intreruperi al timerului extern
- Interactiunea joc-display este facuta prin interfata SPI

Biblioteci folosite:

- Display - AdafruitGFX.h, Adafruit_ST7735.h, SPI.h
- Giroscop - MPU9250_asukiaaa.h (biblioteca custom), Wire.h (biblioteca standard pentru I2C)

- Modul de ceas - I2C_RTC.h

Am ales sa folosesc bibliotecile de mai sus intrucat sunt foarte folosite in utilizarea componentelor pe care le-am cumparat, cu multe forum-uri deschise pentru eventualele probleme si exemple de utilizare a acestora.

Elementul de noutate al proiectului este dat de interactiunea intre joc si utilizator prin intermediul giroscopului, iar calibrarea acestuia a fost facuta prin citirea valorii analogice a giroscopului si trecerea vitezei unghiulare intr-un unghi de inclinatie in jurul axelor Oz (Roll) si Ox (Pitch). Cu toate acestea la momentul curent nu folosesc si unghiul de Pitch.

La momentul curent nu exista optimizari, dar in varianta finala doresc sa folosesc al doilea buton pentru a pune consola in deep sleep, respectiv a o trezi din deep sleep, si sa o conectez la baterii.

Utilizarea functionalitatilor din laborator a fost in special intalnit la transmiterea de date prin I2C (Wire.begin/beginTransmission/endTransmission/requestFrom), prin SPI (SPI.begin/beginTransaction/endTransaction/setClockDivider), si la utilizarea intreruperilor (setarea unei intreruperi pe un anumit GPIO cu attachInterrupt(pin, functia_isr, frontul/valorea pe care se activeaza), si scrierea de isr-uri prin atasarea antetului IRAM_ATTR). De asemenea, am folosit minimal si cunostinte din laborator si datasheet-ul RTC-ului pentru setarea unui Timer.

In cod am folosit structuri de date pentru obstacole si gloante, pentru a le contoriza starea de existenta si coordonatele. Ca functii avem doua pe care le-am trecut in memoria interna a uC-ului pentru intreruperi: fire_bullet() - pentru a trage un glont si rewards() - pentru a primit un reward in game la scurgerea timer-ului extern. Alte functii implementate de mine sunt: start_screen(), game_over_screen(), reset(), spawn_asteroids(). Functii folosite, dar neimplementate de mine sunt cele din biblioteca display-ului, pentru desenarea primitivelor geometrice, folosite pentru gloante (pixel), asteroizi (cercuri) si nava (triunghi). Logica de mutarea a navei, asteroizilor, gloantelor si de coliziune este facuta in functia loop(), pentru a nu incarca stiva cu apeluri inutile din moment ce complexitatea ciclomatica si cognitiva a codului este redusa.

DEMO:

Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Concluzii

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună 😉.

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

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

Export to PDF