Jocul "2048"

Proiect PM - 2023

Student: Adrian-Valeriu Croitoru
Grupa: 332CA
Asistent: Victor Stoica

Introducere

Proiectul reprezinta o implementare a celebrului joc de smartphone - “2048”. Scopul jocului este, in primul rand, de a atinge milestone-ul de **2048** (pe un tile), dar dupa aceea, un high score cat mai mare poate reprezenta un scop la fel de placut.

A fost ales acest proiect datorita faptului ca acest joc este, dupa multi ani, la fel de jucat si cunoscut, fiind una dintre aplicatiile de smartphone pe care o accesez zilnic.

Utilitatea proiectului este, in primul rand, data de factorul de invatare. Cu siguranta vor fi multe dificultati software & hardware pe parcurs.

Totodata, este demn de amintit faptul ca ecranul telefoanelor mobile poate deveni daunator pentru ochi, motiv pentru care acest proiect vine in ajutorul utilizatorului, livrand aceeasi functionalitate a jocului 2048, dar pe un ecran mult mai prietenos cu ochiul.

Descriere generală

Utilizatorul se va putea juca folosind **joystick-ul**. Exista 4 directii posibile - N, S, E, V, orice mutare intre acestea a joystick-ului fiind aproximata la cea mai apropiata directie.
Va exista si un **buton de restart**, iar fiecare mutare a jucatorului va fi marcata printr-un **sunet specific** produs de **buzzer**.
Interactiunea efectiva cu UI-ul va fi realizata prin **afisajul grafic LCD**.

Schema bloc

Hardware Design

Lista componentelor **hardware**:

- 1 x Groundstudio Jade Uno+
- 1 x Display LCD 128*64 (5V iluminat, ST7920 controller)
- 1 x Modul Buzzer
- 1 x Modul joystick cu push-button
- 1 x Placa de prototipare cablaj PCB 9×15 cm
- fire Dupont mama-tata & tata-tata

Schema electrica

Software Design

1. Mediu de dezvoltare: Arduino IDE
2. Biblioteci utilizate: **u8glib** (pentru interfatarea modulului LCD ST7920)

Implementarea software este, ca structura, foarte asemanatoare cu cea in care se realizeaza jocurile video, in OpenGL. Astfel, de fiecare data, se “re-randeaza” cadrul curent, tinand cont de parametri precum:

1. gameState (in ce stadiu se afla jocul: neinceput, in desfasurare, VICTORY sau GAME OVER)
2. pozitia fiecarui tile pe ecran

Se foloseste **Timer1** de pe placuta pentru a contoriza timpul total trecut de la inceperea jocului curent.

Butonul este setat sa produca o **intrerupere pe falling edge**, aceasta fiind concretizata in oprirea imediata a jocului curent si intoarcerea in meniul principal.

Se foloseste **ADC (Analog digital conversion)** pentru a traduce inputul dat de joystick intr-o miscare propriu-zisa UP, DOWN, LEFT sau RIGHT.

Se folosesc multiple variabile de **debounce** pentru a evita situatia in care aceeasi comanda este efectuata de N ori. (e.g. atunci cand tinem joystick-ul blocat in pozitia RIGHT)

Game Flow

1. `getCurrentCommand()` → se obtin valorile celor doua axe OX si OY de la joystick si, in functie de niste threshold-uri, se intoarce comanda curenta (UP, DOWN, LEFT, RIGHT, NOTHING)

2. `executeCommand()` → se executa comanda obtinuta anterior, doar daca este diferita de cea care a fost executata ultima data (deoarece logica se afla intr-o bucla infinita, daca am tine, spre exemplu, joystick-ul in pozitia LEFT, s-ar executa in continuu mutarea LEFT, ceea ce nu este de dorit). Tot la acest pas, se trigger-uieste un semnal catre **buzzer** (doar cand comanda este valida si posibila), cu o durata de 15ms.

3. `evaluateGame()` → se evalueaza starea jocului. In functie de marimea grid-ului, se verifica daca pe tabla exista tile-ul castigator. In caz pozitiv, este vorba de un **VICTORY**. Altfel, se verifica sa mai existe cel putin o mutare posibila. In caz negativ, este **GAME OVER**. Daca niciuna dintre conditii nu s-a implinit, starea jocului ramane **IN_PROGRESS** si se continua flow-ul curent.

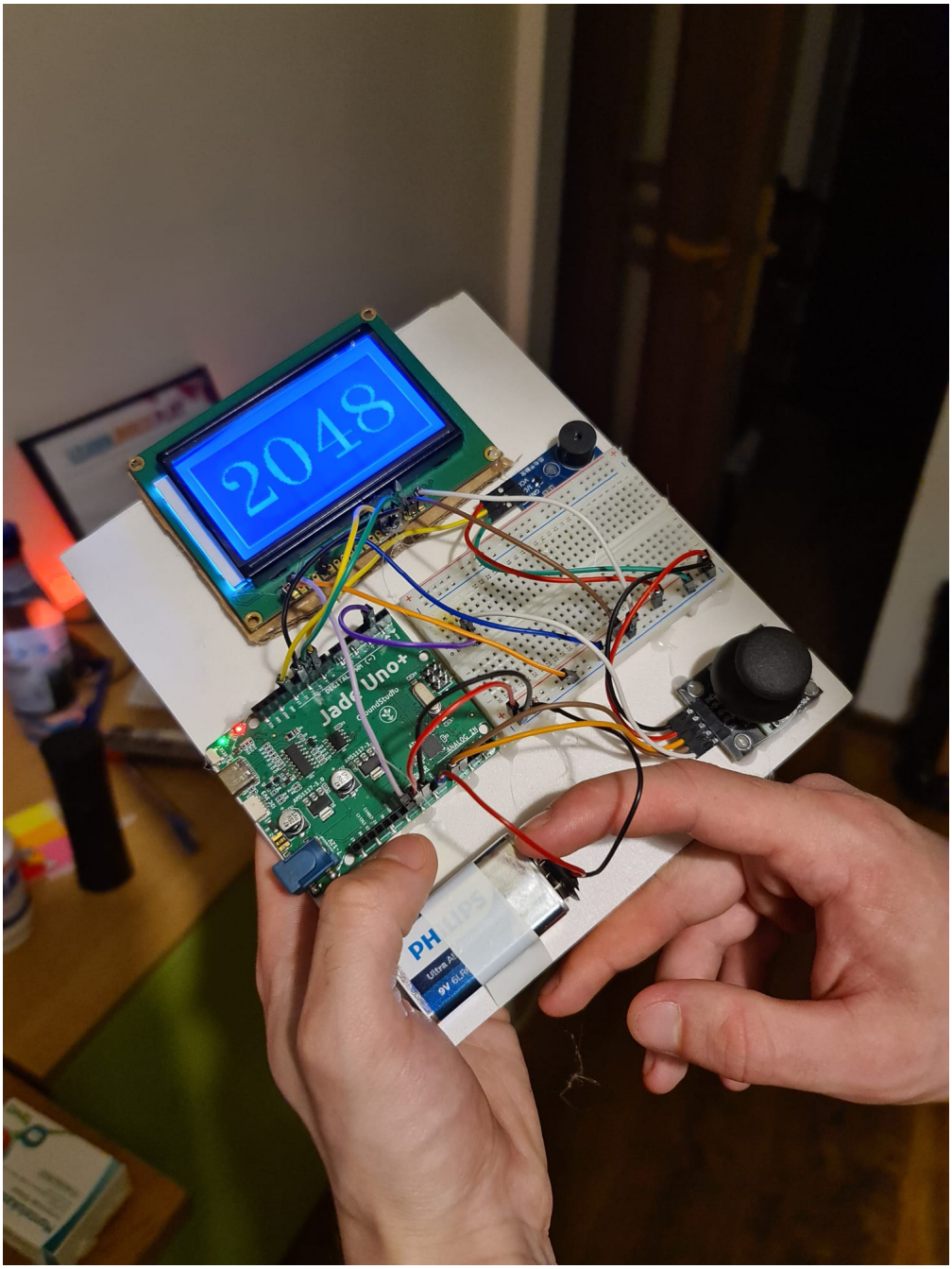
4. `render()` → se randeaza cadrul curent. Se tine cont de **gameState**, de marimea grid-ului si se randeaza cadrul corespunzator.

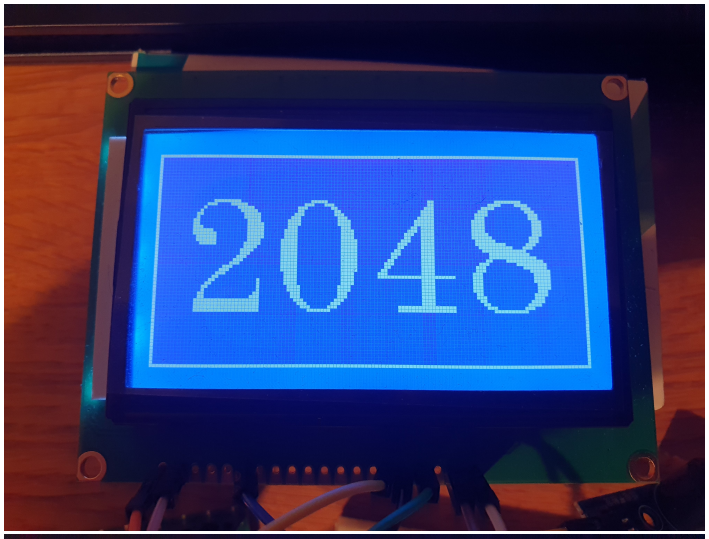
5. `move(moveType);`
`createTile();` → se efectueaza urmatoarea mutare si se creeaza si adauga un nou tile pe tabela (desigur, in mod **random**).

Flow-ul prezentat mai sus **se repeta la infinit**, pana jocul se termina, sau pana cand utilizatorul foloseste intreruperea generata de apasarea butonului, caz in care se face intoarcerea la meniul principal.

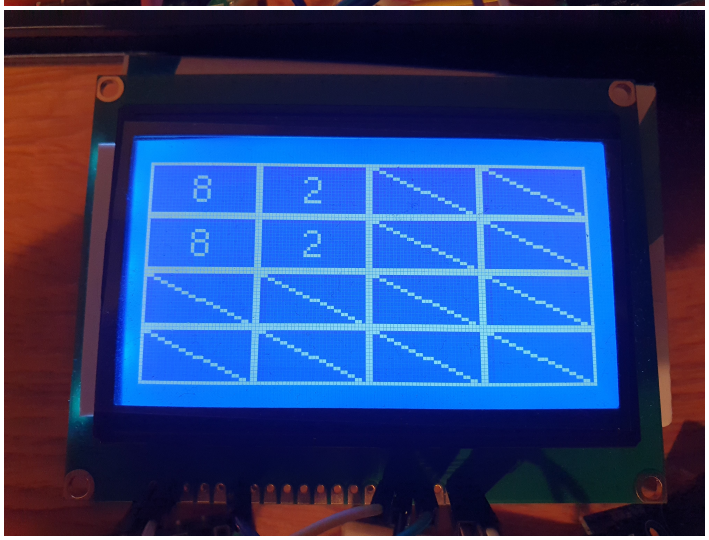
RESET BUTTON → opreste imediat instanta curenta a jocului si face revenirea la meniul principal. Se opreste si reseteaza timer-ul folosit pentru 'elapsedTime'. In meniul principal, acest buton are functia de a selecta dimensiunea dorita a grid-ului si a porni efectiv jocul.

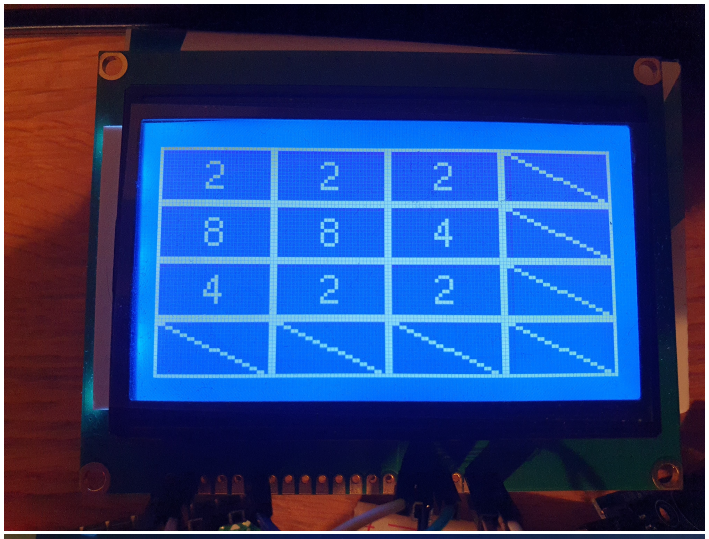
Rezultate Obţinute

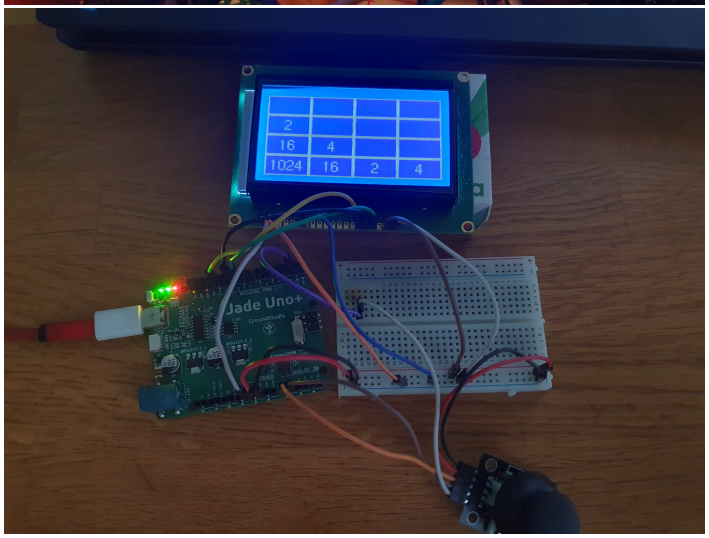


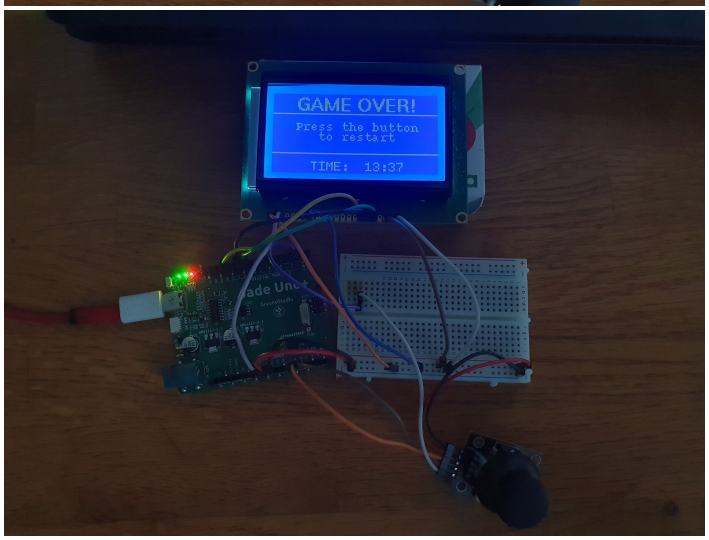

2048


Pick your game!
4x4
3x3
2x2


8
2
8
2


2 2 2
8 8 4
4 2 2


2
16 4
1024 16 2 4
Jade Uno+


GAME OVER!
Press the button
to restart
TIME: 13:37

Concluzii

Proiectul este unul reusit, functionalitatea promisa fiind integral oferita.

Pe plan personal, am apreciat ca a fost un prilej bun de a lua un prim contact cu acest domeniu. Mi-a placut sa descopar si exploatez capabilitatile chip-ului **Atmega328PB** si sa invat mai multe despre **timerele built-in** de pe placuta (folosite pentru calcularea timpului scurs de la inceperea jocului), despre **conversia analog-digital** (joystick-ul), dar si despre modul in care se configureaza si functioneaza **intreruperile** (butonul de reset game).

Mi-a facut placere sa lucrez cu modulul **LCD ST7920 128×64**, lucru in urma caruia am generat alte cateva idei personale pe care sa le dezvolt cu ajutorul acestuia.

Download

2048_sources.zip

Bibliografie/Resurse

Active Buzzer Tutorial
ST7920 128x64 LCD Tutorial
Arduino Analog Joystick Tutorial
Original 2048
2048 Game Logic - 1
2048 Game Logic - 2

Export to PDF