War Strategy Game

Autor: Marina-Crenguța Simion, grupa 331CA

Introducere

Ce face proiectul?

Proiectul este o versiune interactivă a jocului de cărți „Război”, realizată sub forma unui sistem electronic. Jocul poate fi jucat fie împotriva unui adversar controlat automat, fie împotriva unui al doilea jucător. Interacțiunea se face prin butoane, afișaje și efecte luminoase, astfel încât experiența să fie mai dinamică decât în varianta clasică.

Scop

Scopul proiectului este să transform un joc simplu și cunoscut într-o experiență mai atractivă și mai modernă. Pe lângă regulile de bază ale jocului, sistemul adaugă feedback vizual, posibilitatea de a alege între mai multe moduri de joc și un element de strategie, deoarece jucătorul trebuie să decidă când merită să păstreze o carte și când este mai bine să riște pentru una mai bună.

Idee

Ideea a pornit de la dorința de a realiza un proiect ușor de înțeles pentru oricine, dar care să nu fie doar o simplă simulare software. Am vrut ca utilizatorul să interacționeze fizic cu jocul, iar fiecare rundă să includă un mic factor de risc și anticipare, asemănător jocurilor de noroc, dar într-un context simplu și controlat.

Utilitate

Consider că proiectul este util deoarece oferă o variantă modernă și interactivă a unui joc clasic, ușor de folosit și de înțeles. În plus, poate fi jucat atât singur, împotriva unui adversar automat, cât și în doi jucători. Pentru mine, proiectul este o ocazie de a duce o idee simplă până la un produs funcțional, cu reguli clare, interacțiune fizică și feedback vizual.

Descriere generală

Arhitectura proiectului este construită în jurul microcontrollerului **ATmega2560**, care coordonează logica jocului, interacțiunea cu utilizatorul și feedback-ul vizual. Sistemul este împărțit în mai multe module: input de la jucători, afișare pe display-uri, control al LED-urilor și transmitere de informații pentru debugging.

Microcontrollerul ATmega2560: Este componenta centrală a sistemului. Acesta gestionează logica jocului de tip „Război”, generarea pseudo-aleatoare a cărților, schimbarea rundelor, verificarea câștigătorului și comportamentul adversarului automat. Tot el primește comenzile de la butoane, citește valoarea potențiometrului și controlează afișajele și LED-urile.

Display-urile OLED I2C: Sistemul folosește cinci display-uri OLED pentru afișarea informațiilor din joc. Fiecare jucător are câte două display-uri: unul pentru cartea extrasă și unul pentru scorul curent. Al cincilea display este folosit pentru starea generală a jocului, afișând mesaje precum începutul rundei, rândul jucătorului, rezultatul rundei sau finalul jocului.

Multiplexorul I2C: Deoarece display-urile OLED folosesc aceeași adresă I2C, acestea sunt conectate printr-un multiplexor I2C. Microcontrollerul selectează pe rând canalul dorit, astfel încât poate comunica separat cu fiecare display.

Butoanele fizice: Butoanele sunt folosite atât pentru configurarea inițială a jocului, cât și pentru acțiunile din timpul rundelor. La început, acestea permit confirmarea alegerilor făcute, precum modul de joc sau setările inițiale, și pornirea efectivă a jocului. În timpul jocului, butoanele sunt folosite pentru acțiuni precum extragerea unei cărți sau oprirea la cartea curentă.

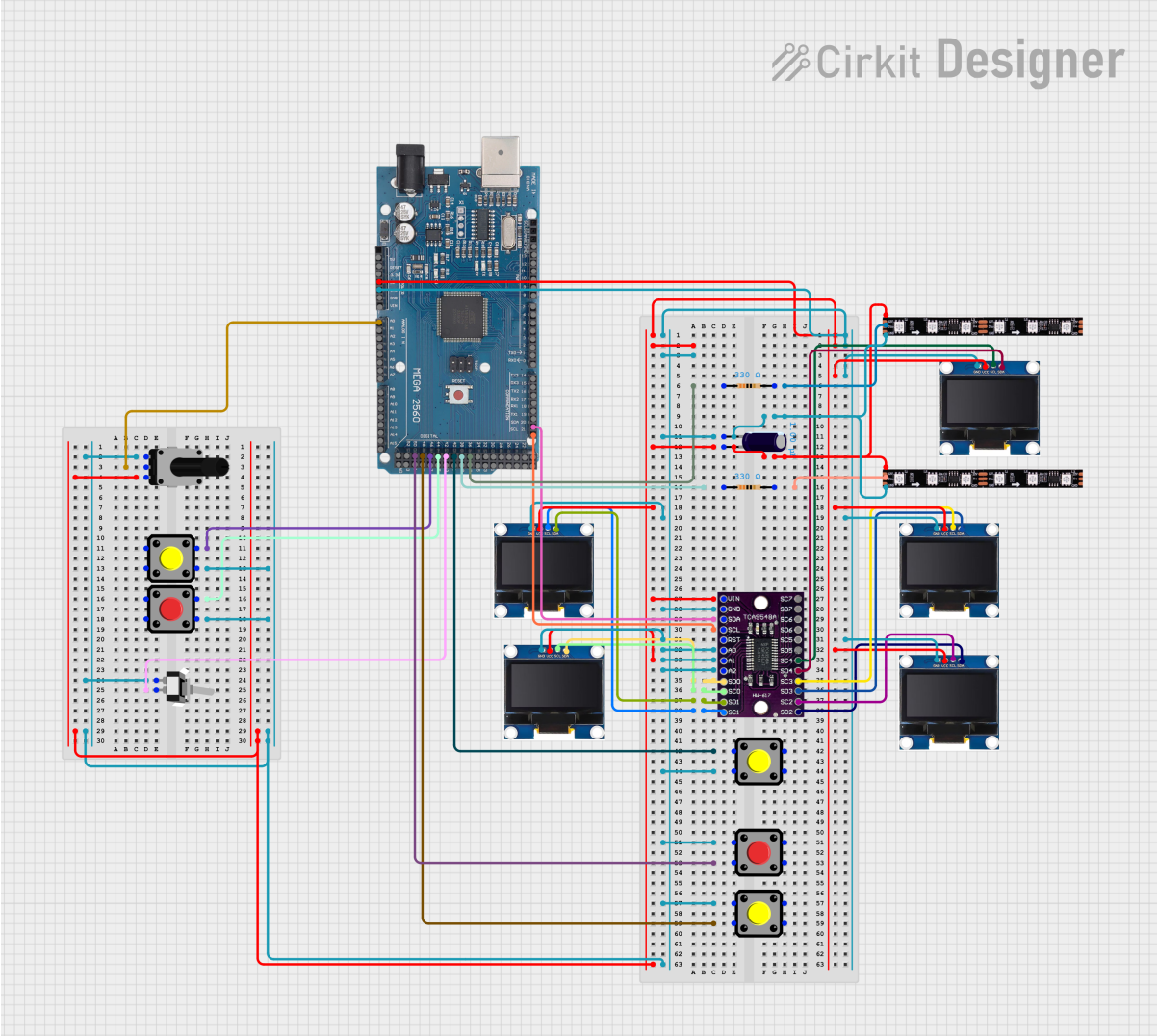
Switch-ul pentru modul de joc: Switch-ul permite alegerea modului de joc: Player vs AI sau Player vs Player. În funcție de poziția acestuia, microcontrollerul schimbă modul în care este controlat adversarul.

Potențiometrul: Potențiometrul este folosit în modul Player vs AI pentru reglarea agresivității adversarului automat: ușor, mediu sau dificil. În funcție de valoarea citită, AI-ul poate lua decizii mai prudente sau mai riscante, de exemplu dacă păstrează o carte sau încearcă să obțină una mai bună.

Banda LED RGB: LED-urile oferă feedback vizual pentru starea jocului. Acestea pot indica câștigarea sau pierderea unei runde și pot crea animații simple pentru a face jocul mai atractiv.

Hardware Design

Schema electrică



Lista componentelor

Componentă	Cod/Model	Cantitate	Rol
Placă Arduino Mega	ATmega2560	1	Controlează logica jocului și toate modulele
Display OLED I2C	SSD1306, 0.96”	5	Afișează cărțile, scorurile și starea jocului
Multiplexor I2C	TCA9548A	1	Permite conectarea celor 5 display-uri OLED
Bandă LED RGB	WS2812B	2	Oferă feedback vizual pentru jucători și rezultat
Potențiometru	10kΩ	1	Reglează agresivitatea AI-ului
Switch	ON-ON	1	Selectează modul Player vs AI sau Player vs Player
Buton	Buton cu revenire	5	Confirmă alegerile, pornesc jocul și controlează acțiunile
Condensator electrolitic	100 µF, ≥ 6.3 V	2	Stabilizează alimentarea benzii LED
Placă de prototipare	PCB perforat 4 x 6	2	Suport pentru montarea componentelor
Rezistență	330Ω	2	Protejează linia de date către WS2812B

Maparea pinilor și conectivitatea

Componentă	Tip pin/conexiune	Pin MCU/canal
Multiplexor TCA9548A	I2C	SDA 20, SCL 21
Display OLED 1	I2C	Canal TCA 0
Display OLED 2	I2C	Canal TCA 1
Display OLED 3	I2C	Canal TCA 2
Display OLED 4	I2C	Canal TCA 3
Display OLED 5	I2C	Canal TCA 4
Buton 1	Intrare digitală	D30
Buton 2	Intrare digitală	D31
Buton 3	Intrare digitală	D32
Buton 4	Intrare digitală	D33
Buton 5	Intrare digitală	D34
Potențiometru	Intrare analogică	A0
Comutator	Intrare digitală	D45
LED-uri RGB WS2812B	Ieșire digitală	D50, D52

Software Design

Laboratoare utilizate

- Laborator 0 - citirea butoanelor, debounce și logica jocului pe stări
- Laborator 1 - testare și debugging în timpul dezvoltării
- Laborator 4 - citirea potențiometrului și alegerea dificultății AI
- Laborator 5 - controlul display-urilor OLED prin multiplexorul I2C

Mediu de dezvoltare

Mediul de dezvoltare folosit este PlatformIO, cu platforma atmelavr, placa configurată ca megaatmega2560 și framework-ul arduino. Configurația proiectului este definită în fișierul platformio.ini.

Frecvența de lucru a plăcii este setată la:

```
board_build.f_cpu = 16000000L
```

Încărcarea programului pe placă se face folosind protocolul:

```
upload_protocol = wiring
```

Pentru monitorizare serială, proiectul este configurat cu viteza de **9600 baud**.

Librării și surse 3rd-party

Proiectul folosește biblioteci standard pentru AVR și C:

- avr/io.h pentru acces direct la registrele microcontrolerului;
- util/delay.h pentru temporizări;
- avr/interrupt.h pentru întreruperi;
- avr/pgmspace.h pentru stocarea imaginilor în memoria program;
- stdint.h, stdio.h, stdlib.h pentru tipuri de date, afișare formatată și generare pseudo-aleatoare.

În platformio.ini este declarată și dependența:

- adafruit/Adafruit NeoPixel

Aceasta este folosită pentru partea de control a **benzilor LED WS2812B**. În proiect, benzile LED sunt conectate și sunt folosite ca feedback vizual la finalul jocului, când partida se termină.

Display-urile OLED sunt controlate prin module proprii, implementate în fișierele display.c și display.h. Comunicarea cu acestea se face prin interfața TWI/I2C, implementată în twi.c și twi.h. Deoarece sunt folosite mai multe display-uri OLED cu aceeași adresă I2C, proiectul folosește un **multiplexor TCA9548A**, iar canalul activ este selectat prin funcția tca_select().

Algoritmi și structuri implementate

Logica aplicației este implementată sub forma unei **mașini de stări finite**. Jocul trece prin mai multe stări: ecranul de start, afișarea modului de joc, alegerea dificultății pentru AI, începutul rundei, tura jucătorului 1, tura jucătorului 2 sau a AI-ului, afișarea rezultatului rundei, finalul jocului și restartarea partidei.

Structura principală folosită este **Game**, care reține informațiile necesare desfășurării jocului: starea curentă, modul de joc, dificultatea AI-ului, runda curentă, scorurile, numărul de acțiuni HIT rămase, cărțile curente și vectorul de cărți deja folosite.

Pentru extragerea cărților se folosește un **algoritm de alegere aleatoare fără repetare**. Fiecare carte extrasă este marcată într-un vector used_cards, astfel încât aceeași carte să nu fie extrasă de mai multe ori în timpul aceleiași partide.

Jocul permite **două moduri de funcționare**: Player vs Player și Player vs AI. În modul cu AI, dificultatea poate fi aleasă prin citirea unei valori analogice. Sunt implementate trei niveluri de dificultate: Easy, Medium și Hard. La nivelul Easy, AI-ul ia decizii simple pe baza propriei cărți. La nivelul Medium, ia în considerare și cartea jucătorului. La nivelul Hard, analizează cărțile rămase și estimează probabilitatea de a obține o carte mai bună.

Surse și funcții implementate

Fișierul **main.c** reprezintă punctul de intrare în aplicație. Acesta inițializează comunicația TWI/I2C, intrările, ADC-ul și interfața grafică, apoi pornește logica jocului prin apelul funcției game_run().

Fișierul **game.c** conține logica principală a jocului. Aici sunt definite stările jocului, structura Game, funcțiile pentru inițializarea datelor, extragerea cărților, gestionarea tururilor, calcularea câștigătorului unei runde, actualizarea scorului și comportamentul AI-ului.

Fișierul **game_ui.c** gestionează interfața cu utilizatorul pe cele cinci display-uri OLED. Sunt implementate funcții pentru afișarea ecranului de start, a modului de joc, a dificultății, a cărților, a scorului, a mesajelor pentru jucători și a rezultatului final.

Fișierul **display.c** implementează funcțiile de bază pentru display-ul OLED: inițializare, ștergere ecran, setarea cursorului, afișarea textului și desenarea bitmap-urilor 128×64. Pentru afișarea textului este folosit un font 5×7 definit în memoria program.

Fișierul **cards.c** conține imaginile tuturor celor 52 de cărți de joc. Acestea sunt memorate ca bitmap-uri în PROGMEM, pentru a economisi memoria RAM a microcontrolerului.

Fișierul **input.c** gestionează butoanele și citirile analogice. Butoanele sunt configurate ca intrări cu rezistențe interne de pull-up, iar funcția button_pressed() implementează o metodă simplă de debounce. Există funcții separate pentru confirmare, HIT și STAND pentru fiecare jucător.

Fișierul **adc.c** configurează convertorul analog-digital al microcontrolerului. Funcția adc_init() setează referința ADC și prescaler-ul, iar funcția myAnalogRead() citește valoarea analogică de pe canalul selectat.

Fișierul **twi.c** implementează comunicația TWI/I2C. Sunt definite funcții pentru inițializarea magistralei, trimiterea condițiilor START și STOP, scrierea și citirea datelor, scanarea dispozitivelor I2C și selectarea canalului activ al multiplexorului TCA9548A.

Fișierul **uptime.c** implementează un contor de timp în milisecunde folosind Timer2 în modul CTC. Acesta poate fi folosit pentru temporizări și măsurarea timpului de funcționare al aplicației.

Fișierul **usart.c** conține funcții auxiliare pentru comunicarea serială prin USART0, utile pentru testare și debugging.

Fișierul **test.c** conține cod auxiliar pentru testarea benzilor LED WS2812B și a switch-ului. În varianta finală a proiectului, aceste elemente sunt conectate la sistem și sunt folosite pentru feedback vizual, în special la terminarea jocului.

În concluzie, firmware-ul este organizat pe module independente, ceea ce face codul mai ușor de urmărit, testat și extins. Separarea logicii jocului de interfața grafică și de partea hardware permite modificarea sau adăugarea unor funcționalități fără rescrierea întregii aplicații. `

Rezultate Obţinute

În urma implementării, proiectul a ajuns într-o stare funcțională, în care jocul poate fi rulat complet pe microcontroler, cu interacțiune prin butoane și afișare pe display-urile OLED.

Au fost obținute următoarele rezultate:

- jocul pornește corect și afișează ecranul de început;
- rundele se desfășoară succesiv, cu actualizarea scorului după fiecare rundă;
- fiecare jucător poate alege acțiunile disponibile folosind butoanele dedicate;
- cărțile sunt generate aleator și sunt afișate grafic pe display-uri;
- sistemul evită reutilizarea aceleiași cărți în cadrul unei partide;
- cele cinci display-uri OLED funcționează simultan prin intermediul multiplexorului I2C;
- mesajele afișate sunt diferite în funcție de etapa jocului și de jucătorul activ;
- modul Player vs Player este funcțional;
- modul Player vs AI este implementat, cu niveluri diferite de dificultate;
- switch-ul este conectat pentru selectarea modului de joc;
- benzile LED sunt conectate și se aprind la finalul jocului;
- sistemul rulează stabil alimentat la 5V, fără blocări în timpul unei partide normale.

Un rezultat important este faptul că proiectul nu rămâne doar o simulare software, ci funcționează ca un joc fizic complet, cu input real, afișare separată pentru mai multe zone ale jocului și feedback vizual la final. Astfel, obiectivul principal al proiectului a fost atins: realizarea unui sistem embedded interactiv, bazat pe o logică de joc implementată direct pe microcontroler.

Concluzii

Proiectul a reprezentat o experiență practică utilă, deoarece a combinat partea de programare embedded cu integrarea efectivă a mai multor componente hardware. Pe lângă implementarea logicii jocului, a fost necesară conectarea și testarea display-urilor OLED, a butoanelor, a switch-ului, a multiplexorului I2C și a benzilor LED.

O parte interesantă a proiectului a fost și realizarea structurii fizice. A fost destul de amuzant, dar și migălos, să adaptez cutii pentru carcasă, să tai spațiile necesare pentru componente și să lipesc firele astfel încât totul să fie cât mai stabil și ușor de folosit.

Din punct de vedere tehnic, proiectul a evidențiat câteva aspecte importante:

Organizarea codului

Codul a fost împărțit în module separate pentru logica jocului, interfața grafică, citirea intrărilor, comunicația I2C/TWI, ADC și resursele grafice. Această structură a făcut aplicația mai ușor de urmărit și de modificat.

Gestionarea memoriei

Imaginile cărților au fost stocate în memoria program folosind PROGMEM, pentru a reduce consumul de RAM. Acest lucru este important în proiectele embedded, unde memoria disponibilă este limitată.

Comunicarea cu display-urile OLED

Pentru folosirea mai multor display-uri OLED cu aceeași adresă I2C, a fost necesară integrarea multiplexorului TCA9548A. Astfel, fiecare display poate fi selectat separat și poate afișa informații diferite.

Logica jocului

Fluxul aplicației este organizat ca o mașină de stări finite, ceea ce permite separarea clară a etapelor jocului: start, rundă, turele jucătorilor, rezultat și final.

Ca îmbunătățiri viitoare, proiectul ar putea fi montat într-o carcasă mai solidă, eventual realizată prin imprimare 3D sau pe o placă PCB dedicată. De asemenea, se pot adăuga animații pe display-uri, efecte LED mai complexe și o integrare mai avansată a modului Player vs AI.

Download

Github Repo

https://github.com/marinaa13/War_Strategy_Game

Demo proiect

https://youtube.com/shorts/xQ2b3eQk9FA?is=A0yvwEU_AsSVsj_M

Jurnal

15.04 – Au ajuns toate componentele comandate.

24.04 – Am început testarea display-urilor OLED și a comunicației I2C, precum și afișarea cărților de joc pe display-uri.

8.05 – Am completat secțiunea de Hardware a documentației.

9.05 – Am conectat, folosind breadboard-uri, butoanele, display-urile și potențiometrul.

15.05 – Am completat secțiunea de Software a documentației.

16.05 – Am decupat și asamblat cutia pentru structura finală a proiectului.

18.05 – Am lipit componentele finale ale proiectului: benzile LED și comutatorul.

21.05 – Am terminat de decorat și pregătit proiectul, aducându-l în forma lui finală.

Bibliografie/Resurse

Resurse Hardware

Microchip ATmega2560 Datasheet

TCA9548A Datasheet – I2C Multiplexer

WS2812B Datasheet

SSD1306 OLED Controller Datasheet

Resurse Software

Image to byte array converter

Adafruit NeoPixel Library Documentation

AVR-LibC Documentation

Export to PDF