Sudoku 4x4

Student: Bogdan-Andrei Buga

Grupa: 332CB

E-mail: bogdandrei04@gmail.com

Introducere

M-am gandit sa implementez un joc clasic de Sudoku, dar limitat la o tabla de 4×4 casute, pentru a putea implementa mai usor mai multe nivele cu grad de dificultate variabil.

Descriere generala

Jucatorul are, in fata sa, 7 butoane disponibile:

- 4 butoane de **navigare** printre casutele tablei de joc (stanga, jos, sus, dreapta);
- 1 buton de **toggle** prin care schimba valoarea casutei pe care se afla cursorul;
- 1 buton de **clear**, prin care sterge o valoare scrisa de-a lungul jocului;
- 1 buton de **check**, prin care verifica daca tabla noastra este completata corect.

Gameplay

Jucatorul va primi o tabla 4×4 completata cu numere de la 1 la 4 si cu spatii libere. Numarul de spatii libere variaza in functie de dificultatea aleasa de jucator la inceputul jocului, dupa ce se termina intro-ul acestuia. Pentru a termina cu succes nivelul dat, jucatorul trebuie sa completeze spatiile cu numere de la 1 la 4 astfel incat:

A) fiecare numar sa apara o singura data pe linia sa;

B) fiecare numar sa apara o singura data pe coloana sa;

C) fiecare numar sa apara o singura data in cadranul sau.

Pe ecranul de joc se vor afisa atat tabla de joc, cat si pozitia cursorului mutabil pe tabla curenta de joc, precum si caracterul din dreptul acestei pozitii.

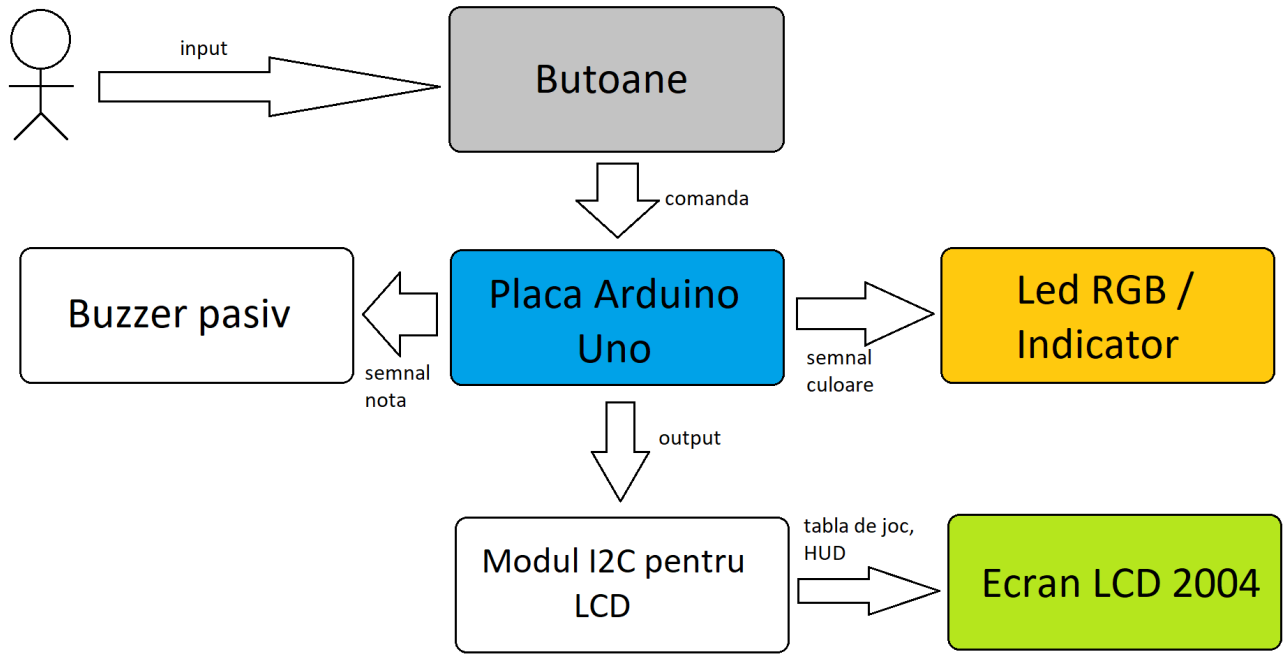
Jucatorul poate sa schimbe valoarea unui spatiu prin apasarea unui buton desemnat pentru a cicla printre numerele de la 1 la 4 sau sa stearga valoarea scrisa daca acesta considera ca nu este corecta. El nu poate schimba un numar deja existent pe tabla, fiind atentionat corespunzator intr-o asemenea situatie.

Verificarea jocului

Jucatorul poate, de asemenea, sa verifice corectitudinea tablei curente. Daca exista vreo greseala in asezarea curenta a numerelor, jucatorul va pierde una din cele 3 vieti cu care a inceput jocul actual. Daca jucatorul isi pierde toate vietile, jocul se incheie intr-un esec.

In cazul in care nu exista greseli, jucatorului i se afiseaza pe ecran, langa tabla de joc, si cate spatii mai are de completat. Atunci cand acesta a completat toate spatiile corect si isi verifica rezultatele, jocul se termina cu un succes.

La terminarea jocului, jucatorul va avea parte de o melodie si de un mesaj corespunzator starii finale a jocului curent (succes sau esec).



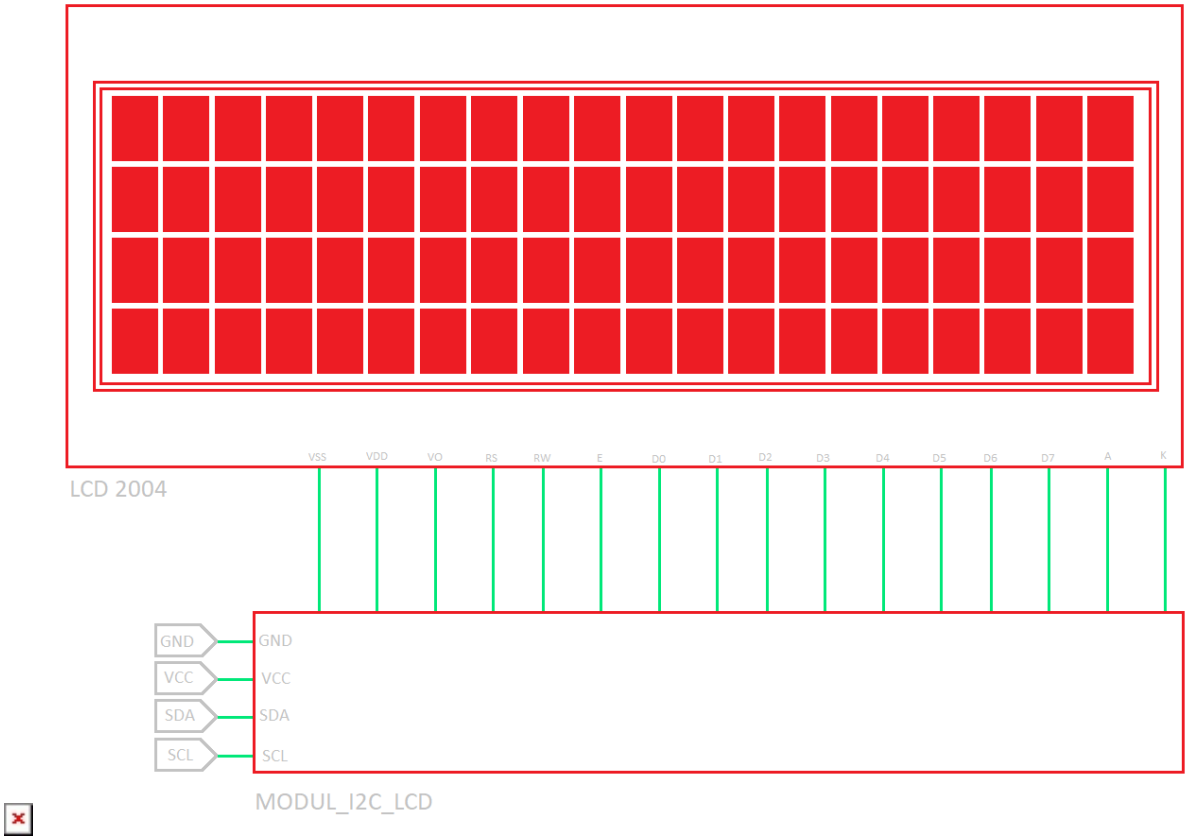
Hardware Design

Componente necesare

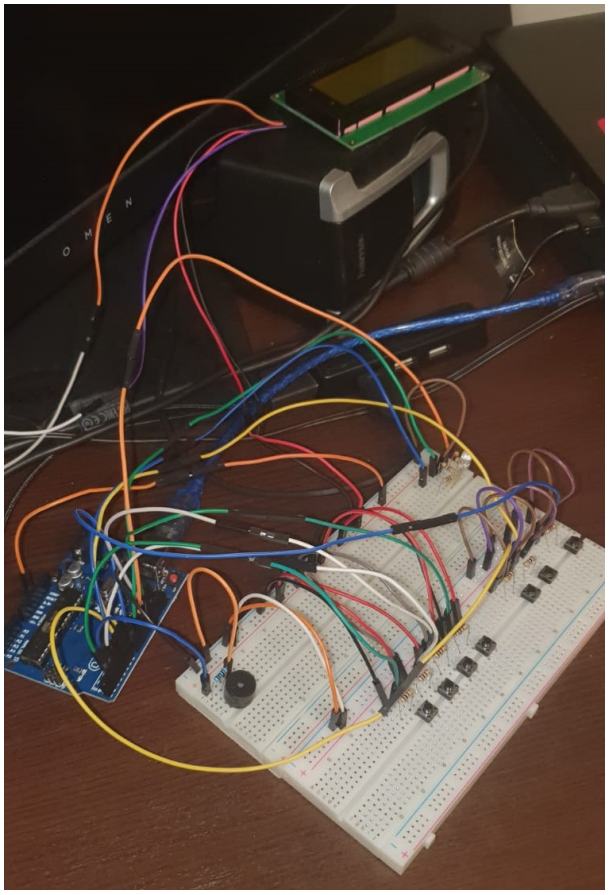
- 1 placa de dezvoltare Arduino Uno
- 1 cablu USB A-B

- 7 butoane
- 1 ecran LCD 2004
- 1 adaptor I2C pentru LCD
- 1 buzzer pasiv
- 1 LED RGB
- 10 rezistori de 1k ohmi
- 1 rezistor de 100 ohmi

Schema electrica



Layout fizic



Butoanele au fost asezate in urmatoarea ordine, de la stanga la dreapta: LEFT, DOWN, UP, RIGHT, CELAR, TOGGLE si CHECK.

Cea mai dificila parte a fost aranjarea tuturor pinilor necesari astfel incat PWM-ul din pinii pentru indicatorul LED RGB sa nu tulbure semnalul PWM-Tone trimis catre buzzer.

Software Design

Medii de dezvoltare folosite

- **Arduino 1.8.13**, pentru scrierea si testarea programului
- **Notepad++**, pentru comentarii si aranajarea codului

Variabile importante folosite

- checkDifficulty
  - un bool care este “true” daca s-a trecut de introducere si nu s-a ales niciun grad de dificultate;
  - acesta devine “false” pe tot restul jocului odata cu alegerea dificultatii jocului.
- variabile de memorare a pozitiei cursorului: X si Y;

- prompt
  - un char[20] folosit pentru afisarea mesajelor de HUD pe ecranul LCD.

Functiile utilizate si scopul acestora

Functii de iesire

Aceste functii sunt folosite pentru a transmite date catre ecranul LCD, buzzer si led-ul RGB.

1. playIntro()
   - Trimite note muzicale catre buzzer pentru a canta melodia de introducere aleasa (“We Are Number One” de Stefan Karl Stefansson).
2. playVictoryJingle()
   - Trimite note muzicale catre buzzer pentru a canta o melodie compusa manual care felicita jucatorul pentru victoria obtinuta.
   - Transmite culoarea verde catre indicator.
3. playGameOverJingle()
   - Trimite note muzicale catre buzzer pentru a canta o melodie compusa manual care ii spune, subtil, jucatorului ca si-a pierdut toate vietile si nu isi mai poate continua jocul.
   - Transmite culoarea rosie catre indicator.
4. beep(int note)
   - Trimite catre buzzer o nota muzicala data ca parametru pentru o durata scurta de timp.
5. printGameScreen()
   - Afiseaza, pe ecranul LCD, toate detaliile jocului curent: tabla de joc, pozitia cursorului, caracterul din dreptul acestuia si vietile ramase.
6. printWarning()
   - Trimite catre buzzer o nota muzicala specifica unei mutari ilegale.
   - Jucatorul este avertizat pe ecranul LCD ca nu poate modifica pozitia curenta, care nu era spatiu pe tabla originala.
   - Transmite culoarea mov / magenta catre indicator.
7. printGoodCheck(char no_spaces)
   - Trimite catre buzzer o nota muzicala specifica unui joc bine jucat pana la momentul verificarii inclusiv.
   - Scrie pe ecranul LCD atat un mesaj incurajator, cat si numarul de spatii ramase (stocat in parametrul “no_spaces”).
   - Transmite indicatorului o culoare ce indica progresul jocului: albastru, daca sunt completate mai putin de jumatate din spatii, sau turcoaz, in caz contrar.
8. printLevelComplete()
   - Transmite catre buzzer melodia de victorie definita in *playVictoryJingle()*.
   - Termina jocul cu mesaje adecvate scrise pe ecranul LCD.
   - Transmite culoarea verde catre indicator.
9. printBadCheck(char lives)
   - Trimite catre buzzer o nota muzicala specifica unei greseli.
   - Jucatorul va fi instiintat pe ecranul LCD cate vieti mai are (numar stocat in parametrul “lives”).
   - Transmite indicatorului o culoare rosie de intensitate invers proportionala cu numarul de vieti ramase.

Functii de gameplay

Aceste functii implementeaza partea de back-end a jocului, ocupandu-se de verificarile necesare pentru ca jocul sa se desfasoare cum trebuie.

1. checkBoardElement(char board[16], char original[16], char y, char x)
   - verifica daca elementul tablei “board” de pe pozitia data (Linia “y”, Coloana “x”) este unic pe linia sa, unic pe coloana sa si unic pe cadranul sau (cadran = bloc 2×2 rezultat prin impartirea tablei de joc in 4 patrate egale), numai in cazul in care pozitia curenta reprezinta un spatiu pe tabla initiala (tabla “original”).
   - Returneaza “true” daca verificarea mentionata este valida sau “false” in caz contrar.
2. checkBoard(char board[16], char original[16])
   - Verifica asezarea corecta a tuturor numerelor pe tabla curenta (“board”) prin apelurile functiei *checkBoardElement(board, original, i, j)*, unde i si j iau valori in intervalul [0,4).
   - Returneaza “false” daca s-a gasit cel putin un element numeric asezat incorect pe tabla curenta sau “true” in caz contrar.
3. setup()
   - Initializeaza ecranul LCD, caracterele speciale, pinii de pe placa Arduino si porneste introducerea jocului.
   - La inceputul fiecarui joc, jucatorul va avea 3 vieti.
4. loop()
   - Pentru inceput, trebuie selectata dificultatea dorita a jocului. In functie de alegerea jucatorului, se va genera o tabla de joc aleator dintr-o baza de date de cate 35 de table de joc pentru fiecare dificultate si, astfel, jocul va incepe.
   - Pentru fiecare buton apasat, se vor lua urmatoarele decizii:
     - **LEFT** : deplaseaza cursorul la stanga pe tabla de joc;
     - **DOWN** : deplaseaza cursorul in jos pe tabla de joc;
     - **UP** : deplaseaza cursorul in sus pe tabla de joc;
     - **RIGHT** : deplaseaza cursorul la drepata pe tabla de joc;
     - **CLEAR** : sterge valoarea numerica scrisa din dreptul unui spatiu de pe tabla originala;
     - **TOGGLE** : scrie o valoare numerica in dreptul unui spatiu sau modifica aceasta valoare daca a fost deja scrisa;
     - **CHECK** : verifica corectitudinea jocului actual; daca exista minim o greseala, jucatorul va pierde o viata, iar daca isi pierde toate cele 3 vieti, nu va mai putea continua jocul curent; daca nu exista nicio greseala, i se arata cate spatii libere mai are de completat sau, daca toate spatiile au fost completate corect, inseamna ca a terminat runda actuala si jocul se incheie cu un succes al jucatorului.
   - Daca jocul nu s-a terminat, se va afisa tabla curenta de joc, impreuna cu celelalte elemente de HUD (pozitia cursorului, caracterul curent, vietile ramase).

Rezultate Obtinute

Introducerea jocului

Proba 1

- Joc usor (pe dificultatea EASY)
- Navigarea printre casutele tablei de joc
- Scrierea, modificarea si stergerea unei casute
- Verificari corecte
- Completarea nivelului dat cu succes

Proba 2

- Joc greu (pe dificultatea HARD)
- Incercarea de a face CLEAR sau TOGGLE pe un numar neschimbabil
- Greseli la verificare si pierdrea vietilor
- Terminarea nivelului dat cu esec

Concluzii

A fost un proiect interesant, la care mi-a placut sa lucez si pe care mi-as fi dorit sa-l fac si mai amplu. Datorita limitarilor de memorie, nu am reusit sa incadrez decat 35 de configuratii intiale ale tablei de joc pentru fiecare dificultate, tinta initiala fiind de 60 de configuratii initiale per dificultate. De asemenea, mi-ar fi placut daca as fi putut folosi 2 buzzere simultan, dar placa de dezvoltare Uno nu permite decat unui singur buzzer sa functioneze la un moment dat.

Download

Codul sursa - Sudoku 4x4 pe LCD
Documentatia proiectului (PDF)

Jurnal

- 27/04/2021 : Alegerea temei
- 18/05/2021 : Adunarea tuturor componentelor necesare
- 27/05/2021 : Finalizarea design-ului hardware
- 31/05/2021 : Finalizarea design-ului software
- 04/06/2021 : Finalizarea documentatiei

Bibliografie / Resurse

https://ocw.cs.pub.ro/courses/pm/prj2010/mcarjaliu/sudoku4x4 (proiectul de la care am plecat)

https://arduinogetstarted.com/tutorials/arduino-lcd-i2c, pentru functiile LCD folosite in codul sursa.

https://www.arduino.cc/reference/en/language/functions/advanced-io/tone/, pentru transmiterea notelor si a melodiilor pe buzzer.

https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/, pentru alegerea pinilor specifici indicatorului LED RGB.

https://www.youtube.com/watch?v=iEXPkv7lJgc (sursa de inspiratie pentru alegerea melodiei de introducere)

https://ocw.cs.pub.ro/courses/_media/pm/prj2010/mcarjaliu/60_sudokus_4x4_easy.pdf (configuratiile EASY de joc)

https://ocw.cs.pub.ro/courses/_media/pm/prj2010/mcarjaliu/60_sudokus_4x4_difficult.pdf (configuratiile HARD de joc)