Gameboy

Introducere

Ideea proiectului este proiectarea unei console de tip Gameboy. Aceasta va pune la dispozitia jucatorului 3 jocuri clasice: Flappy Bird, Pong, Breakout. Acesta se face util prin faptul ca poate reprezenta o sursa de entertainment.

Descriere generală

Schema Bloc

Descriere elemente:

- Breadboard: este platforma pe care voi construi circuitul, oferind conexiuni intre componentele folosite.
- Arduino UNO ATmega328P: este creierul sistemului, gestionând logica si interacțiunea cu componentele.
- Display Matrix 8×8 LED: este ecranul principal al Gameboy-ului.
- Butoane: sunt interfața principala de control, permitand user-ului să interacționeze cu jocul.
- Jump Wires: sunt cabluri flexibile folosite pentru a face conexiuni intre componente pe breadboard.
- Speaker: furnizeaza sunete și efecte audio pentru a îmbunatati experiența de joc.
- User: este persoana care va interacționa cu consola.

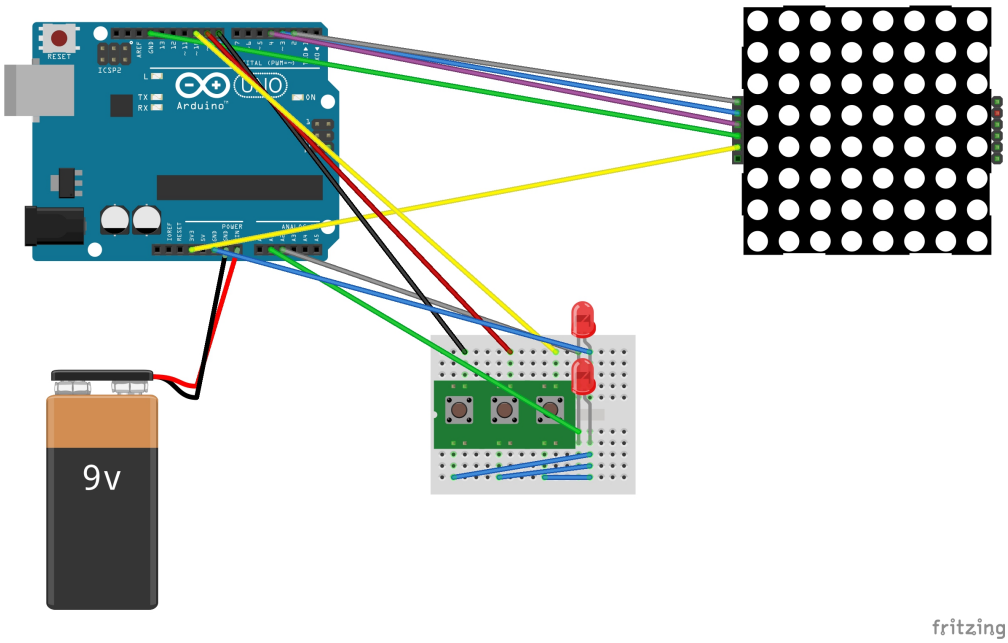
Hardware Design

Listă de piese

Componenta	Cantitate	Imagine	Descriere
Arduino Uno ATmega328p	1		Reprezinta creierul din spatele joculetelor

Display Matrix LED 8×8 7219	1		Componenta pe care se vor afisa jocurile
Breadboard	1		Ofera conectivitatea intre componente
Jump Wires	13		Fac conexiunea intre componentele de pe breadboard si microcontroller, precum si conexiunea cu display-ul
Butoane	3		Interfata prin care user-ul interactioneaza cu dispozitivul
Led	2		Mod de reprezentare a unor mecanici din joc
Baterie 9V	1		Componenta cu care ofer viata gameboy-ului

Scheme electrice



Pini folositi pentru Arduino UNO:

- 2, 3, 4 - pentru cele 3 butoane
- 5, 6 - pentru LED-uri
- 8, 9, 10 - pentru a conecta pinii CIN, CS, CLK de pe matrix led
- GND - pentru a conecta ground-ul de pe matrix led si cel de pe breadboard

- 5V - pentru a conecta la VCC de pe matrix led

Software Design

Descrierea codului aplicaţiei (firmware):

- **mediu de dezvoltare**:
  - Arduino IDE
  - Fritzing
- **librării şi surse 3rd-party**:
  - LedControl: folosita pentru implementarea functiilor ce pun la dispozitie controlul led-urilor de pe matricea led 8×8
- **algoritmi şi structuri implementate**:
  - Debouncing
  - Counter de timpi

Surse

- **Menu**

Locul unde implementez meniul prinicipal alaturi de instantierea pin-urilor folositi, precum si a unor varabile de control care permit fluidizarea jocului.

Functii:

1. setup() - aceasta initializeaza pin-urile pentru butoane cat si pentru led-uri (pentru care am folosit input de pullup); pregateste random seed-ul pentru generarea random a variabilelor din program, si apeleaza setup-urile jocurilor.
2. loop() - functia principala, care verifica tipul jocului urmand sa apeleze functiile necesare jocului selectat. De asemenea acesta, goleseste display-ul in momenetul in care jucatorul pierde la unul din jocuri, afisand un emoji corespunzator, urmand ca dupa un delay de cateva secunde sa revina in meniul principal
3. changeGame() - se ocupa se selectarea jocului
4. drawEnd(), drawBird(), drawSimon(), drawBreakout() - functii de afisare

- **Bird**

Sursa ce contine implementarea mecanicilor referitoare la primul joc, Flappy Bird. Aici se folosesc algoritmi de deboucing pentru butoane astfel incat sa preiau doar semnalele importante.

Functii:

1. SetupGameF() - functia care initializeaza structura Bird, varabilele jocului, precum si pozitia initiala a jucatorului
2. moveBird() - se ocupa de controlul personajului, astfel ca prin algoritmi de deboucing se efectueaza corect citirea semnalelor provenite
3. din apasarea butonului de jump

4. switchWall(), doWall() - se ocupa de generarea random a obstacolelor
5. adjustGameDifficultyF() - mareste sau scade dificultatea jocului in functie de un anumit timp
6. checkCollisionF() - verifica coliziunea cu obstacolele, aceasta decide finalul jocului

- **Breakout**

Implementeaza jocul Breakout, folosind algoritmi pentru dinamica mingii, bouncing random din pereti si obstacole, selectarea la intamplarea a unuia din cele 3 nivele.

Functii:

1. setupGameB() - initializeaza matricea corespunzatoarea fiecarui nivel selectat random, pozitia initiala a jucatorului si a mingii, precum si alte variabile ce tin de directia mingii
2. drawLevel() - deseneaza nivelul selectat anterior
3. movePlayer() - functia ce se ocupa de controlul jucatorului, precum si a schimbarii directiei mingii in momentul in care acesta loveste
4. mingea cu una din cele 3 puncte
5. drawPlayerB() - deseneaza pozitia jucatorului pe tot parcusul jocului
6. moveBallB() - manevreaza mingea in functie de directie. Verifica modul in care aceasta face bounce in urma a 3 tipuri de coliziune:
7. player, wall si block.
8. checkBlockBallCollision() - verifica daca mingea are coliziune cu unul din obstacole, moment in care mingea poate ricosa stanga sau dreapta.
9. checkWallBallCollision() - verifica daca mingea are coliziune cu peretii astfel: daca este vorba de unul dintre peretii stanga, dreapta, atunci mingea va ricosa in directia opusa, in schimb daca loveste peretele de sus va ricosa random.
10. checkPlayerBallCollision() - verifica punctul in care mingea are coliziune cu jucatorul, moment ce decide directia in care mingea se deplaseaza, astfel ca daca mingea atinge punctul din stanga al jucatorului, va ricosa in stanga; daca loveste punctul din mijloc ricoseaza pe directie dreapta; iar daca loveste punctul din dreapta va ricosa in dreapta

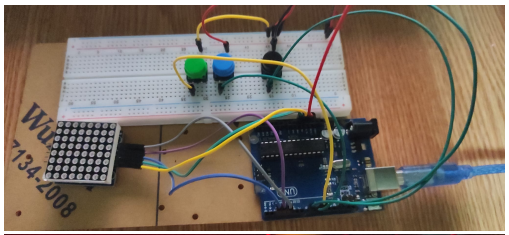
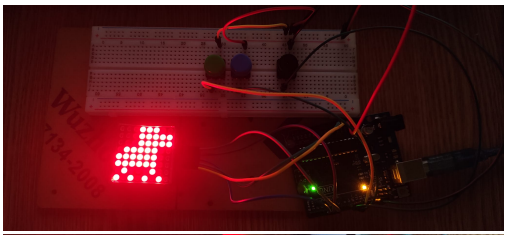
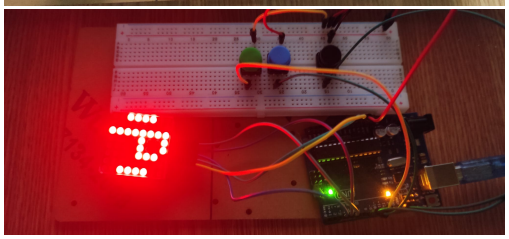
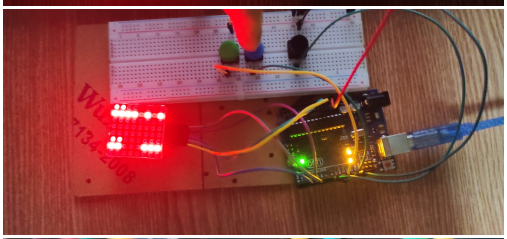
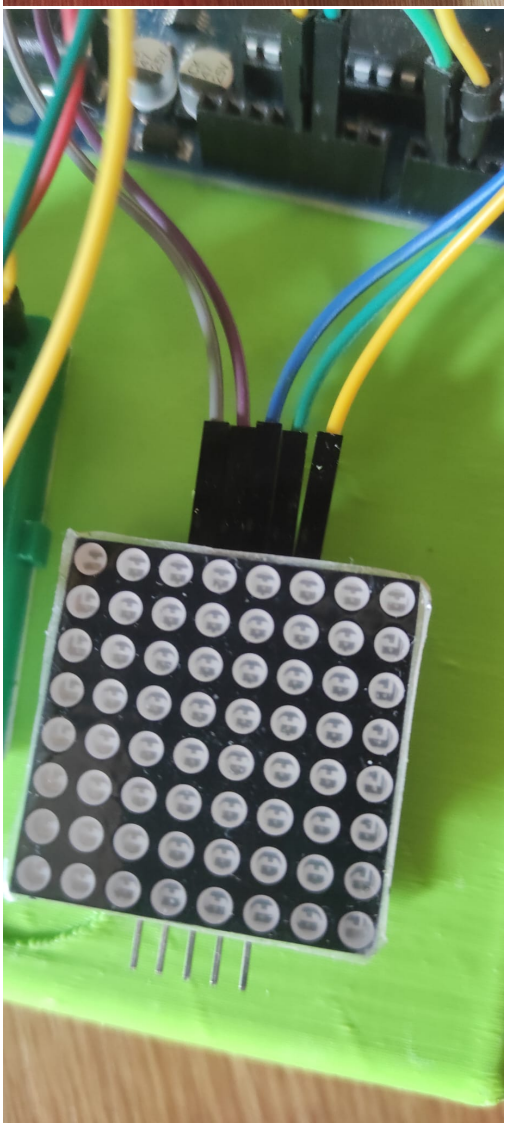
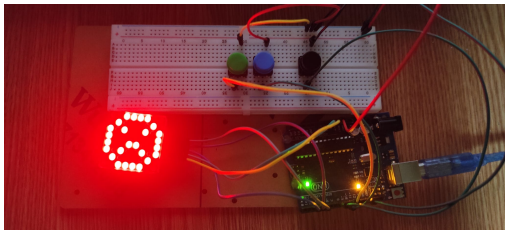
- **Simon**

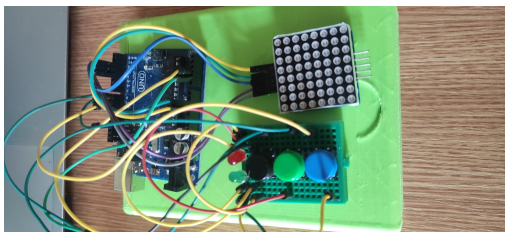
Aceasta sursa contine implementarea jocului Simon Says, prin declararea a doua array-uri, unul care retine alegerile random ale jocului, iar celalalt alegerile jucatorului.

Functii:

1. setupGameS() - initializeaza varabilele caracteristice acestui joc, precum si a array-urilor ce retin alegerile jocului generate random
2. startCounting() - afiseaza numaratoarea ce reprezinta start-ul jocului
3. drawChoices() - afiseaza alegerile jocului prin sageti in 4 directii
4. drawPlayerChoices() - afiseaza alegerile jucatorului
5. playGameS() - functia principala in care sunt apelate celalte functii necesare jocului, locul unde se verifica daca jucatorul poate continua sau s-a terminat jocul. Aici se verifica ce buton este apasat, si daca prin apasarea butonului de pe pin-ul 8, se schimba directia unui din cele 2 butoane care initial sunt dreapta, sus.
6. makePlayerChoices() - verifica daca jucatorul a ales corect si poate continua, altfel jocul se incheie.
7. checkGameState() - se ocupa de aprinderea led-urilor, cel rosu pentru semnalizarea faptului ca este timpul jocului de alegere, si cel verde care indica faptul ca functionalitatea unui buton a fost schimbata.
8. drawL(), drawR(), drawU(), drawD() - functiile care se ocupa de afisarea alegerilor

Rezultate Obţinute



Download

Download Link:

- proiect.zip

Jurnal

5 Mai: Creare pagina OCW

10 Mai: Incepe montare componente

11 Mai: Start creare joc Flappy Bird

13 Mai: Pas avansat in crearea primului joc Flappy Bird:

- Generare obstacole random
- Cresterea dificultatii de obstacole
- Controlul jucatorului printr-un buton

15 Mai: Finalizarea primului joc:

- End Game
- Coliziune

22 Mai: Finalizare proiect

Demo-uri:

- https://www.youtube.com/shorts/vpvCSkZ4g1I
- https://www.youtube.com/shorts/wpiPJdVrDOE
- https://www.youtube.com/shorts/3Kfx7MjyFC4
- https://www.youtube.com/shorts/dk1sGpKEsL0

Bibliografie/Resurse

Tool-uri

- Draw.IO pentru schema bloc
- Tool pentru desenare pe matricea led https://xantorohara.github.io/led-matrix-editor/

Datasheets

- https://www.analog.com/media/en/technical-documentation/data-sheets/MAX7219-MAX7221.pdf
- https://content.arduino.cc/assets/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

Export to PDF