Consolă de jocuri clasice

Introducere

Proiectul constă în realizarea unei console de jocuri clasice printre care Tetris, X și O și 2048 folosind o plăcuță Arduino. Jocurile vor fi afișate pe un LCD, iar input-ul jucătorului se va face printr-un joystick și 2 butoane. De asemenea, se vor folosi led-uri și un buzzer pentru pentru a îmbunătăți experiența utilizatorului.

Scopul proiectului este de a implementa o consolă de jocuri clasice care simulează experiența de joc oferită de o consolă din anii 2000.

Consider că acest proiect este util pentru a experimenta funcționalitățile oferite de un Arduino, modalitatea de a conecta un modul LCD și diverse alte componente (joystick, butoane, led-uri, buzzer) la plăcuța Arduino pentru a implementa jocuri clasice apărute încă din anii 80'.

Descriere generală

Plăcuța Arduino va prelua date de intrare furnizate de joystick (va citi o valoare corespunzătoare deplasării pe coordonata Ox și o valoare corespunzătoare deplasării pe coordonata Oy) și le va interpreta în mod corespunzător, va actualiza datele din joc (poziția pieselor pe tablă / actualizarea scorului jucătorului) și va trimite către LCD imaginea de afișat. Pe parcursul și la finalul jocului se va îmbunătăți experiența de joc cu ajutorul buzzer-ului și a led-urilor disponibile.

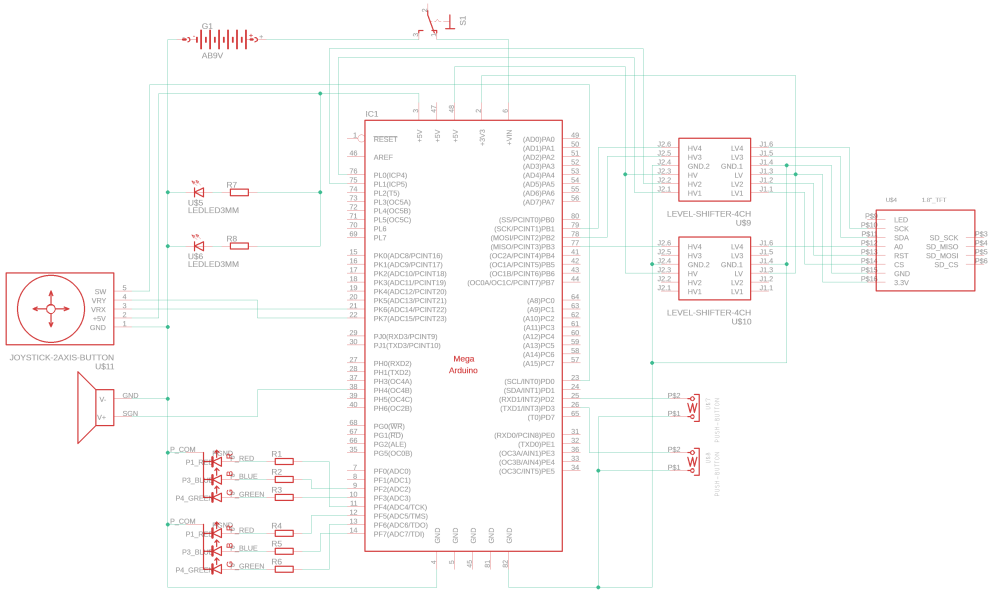
Schema bloc este următoarea:

Hardware Design

Componentele folosite în proiect sunt următoarele:

- 1 x Arduino MEGA 2560 R3
- 1 x Modul LCD SPI de 2.8'
- 2 x Translator de nivel bidirectional
- 1 x Joystick biaxial
- 1 x Buzzer
- 2 x LED RGB
- 1 x LED rosu
- 1 x LED albastru
- 8 x Rezistor 150 ohm
- 2 x Butoane
- 1 x Switch
- 1 x Baterie 9V
- 1 x Suport baterie 9V
- 1 x Cablaj de test
- 1 x Materiale pentru lipit (fludor / decapant / fire de legătură)

Schema electrică este următoarea:



Software Design

Mediul de dezvoltare folosit pentru implementarea proiectului este Arduino IDE. Începutul proiectului din punct de vedere hardware și software a constat în:

- Testarea fiecărei componente pentru a valida funcționalitatea / deprindere cu API-ul disponibil; dificultăți am întâmpinat la conectarea LCD-ului la Arduino (conexiuni slabe din cauza breadboard-ului, lipsa experienței în utilizarea bibliotecii Ucglib)
- Testarea joystick-ului

- Testarea butoanelor și a led-urilor rgb
- Testarea buzzer-ului
- Montarea tuturor pieselor pe un cablaj de test; lipirea firelor pentru conexiuni stabile; construcția cutiei și asamblarea tuturor componentelor
- Programarea propriu-zisă a jocurilor

Din punct de vedere software, implementarea proiectului a presupus:

- Implementarea meniului principal: afișare de text, implementare de butoane controlabile cu joystick-ul
- Implementarea fiecărui joc în parte
- Implementarea stării de pauză în care se poate ajunge la apăsarea unui buton în timpul jocului
- Implementarea revenirii la meniul principal la apăsarea unui buton în timpul jocului
- Implementarea efectelor vizuale / sonore folosind cele 2 led-uri rgb și buzzer-ul pentru fiecare joc în parte

Pentru fiecare joc în parte, există o anumită structură care memorează statusul jocului și permite implementarea logicii jocului:

- Tetris

Într-o buclă infinită, se generează următoarea poziție pentru piesa în mișcare pe baza inputului de la joystick. În momentul în care jucătorul apasă pe joystick, se execută o rutină de tratare a întreruperii de pe pinul corespunzator în care se marchează faptul că piesa curentă trebuie rotită. Se verifică dacă prin rotire / deplasare stânga - dreapta / coborâre cu o poziție nu apare vreo coliziune. Dacă noua poziție este validă, se redesenează piesa prin suprascriere cu negru și redesenare în noua poziție. Altfel piesa rămâne pe loc și la următoarea iterație se va genera o nouă piesă. La fiecare iterație se verifică existența rândurilor complete care vor fi eliminate din tabla de joc, aducând o bonificație la scor.

- X și O

Într-o buclă infinită, se așteaptă input-ul jucătorului. În momentul în care acesta se deplasează cu joystick-ul stânga - dreapta, sus - jos, se redesenează pătratul care permite selecția următoarei mutări. În momentul în care se apasă pe joystick, prin rutina de tratare a întreruperii, se marchează un flag în memorie. La următoarea iterație, dacă acest flag este setat, se verifică dacă poziția selectată este validă, caz în care se desenează un X în celula corespunzătoare. În continuare se generează o poziție pentru jucătorul O după o logică trivială: se verifică dacă se poate câștiga dintr-o mutare cu un O; altfel se verifică dacă se poate câștiga cu o mutare cu un X (pentru apărare); altfel se alege aleator o poziție liberă. Se verifică în permanență dacă jocul s-a încheiat și care jucător a câștigat.

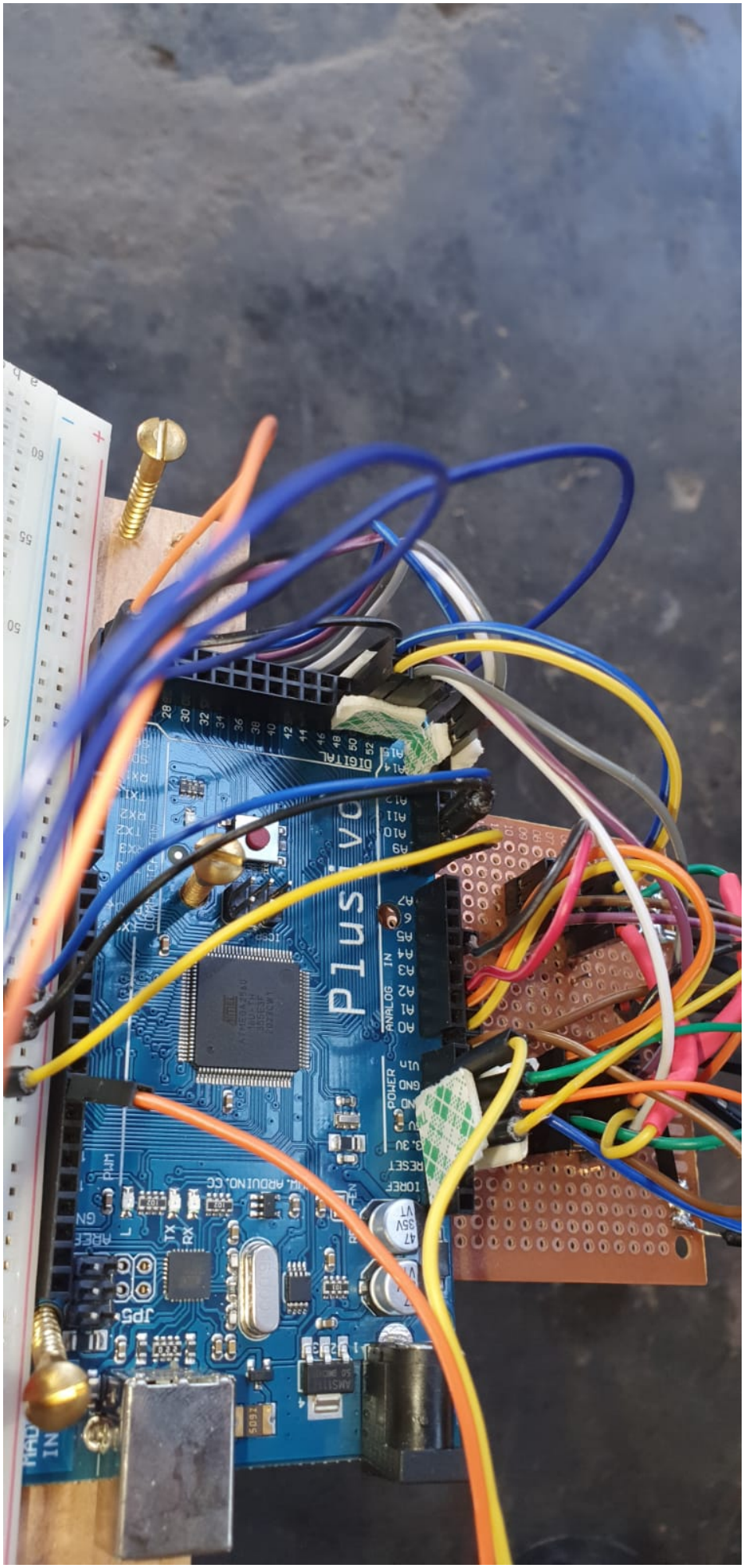
- 2048

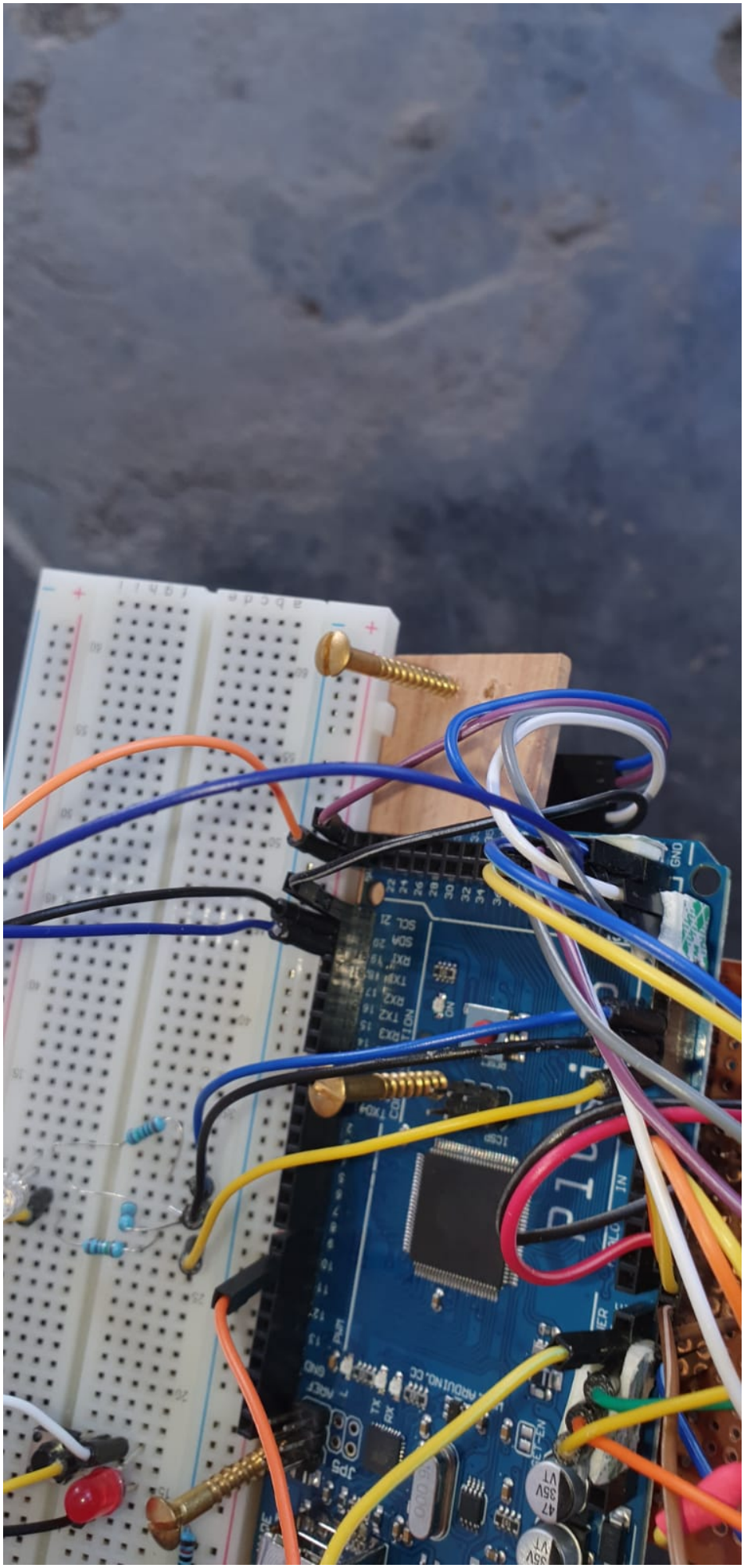
Într-o buclă infinită, se așteaptă după deplasarea joystick-ului. Se verifică dacă este o mutare validă. Dacă aceasta este validă, se updatează tabla de joc și se redesenează în sensul în care a fost deplasat joystick-ul (exemplu: jucătorul a deplasat joystick-ul spre dreapta, actualizarea tablei de joc se va face din partea stângă spre partea dreaptă a tablei).

Rezultate Obținute

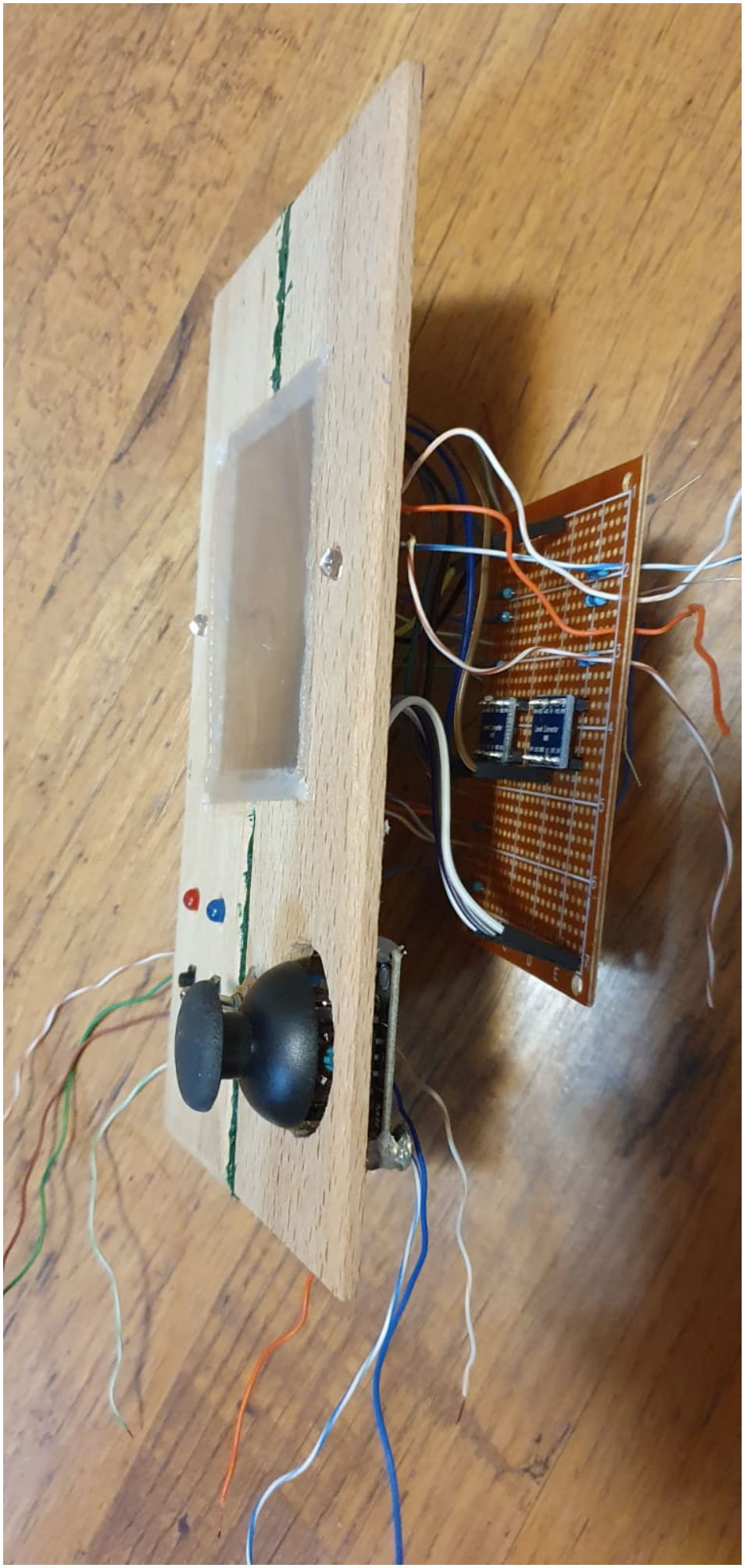
Pentru monitorizarea progresului am realizat câteva imagini din timpul implementării:

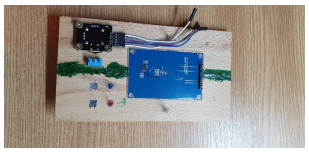
- Testarea componentelor pe breadboard (o parte din legături sunt deja lipite pe un cablaj de test pentru a obține conexiuni ferme):

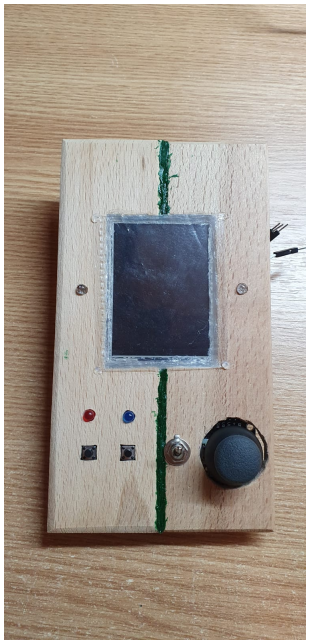

DIGITAL
ANALOG IN
POWER
RESET
IOREF
TX
RX
JP5


GND
SCL 21
SDA 20
RX1 19
TX1 18
RX2 17
TX2 16
RX3 15
TX3 14
TXD4
IN
JP6
47
35V

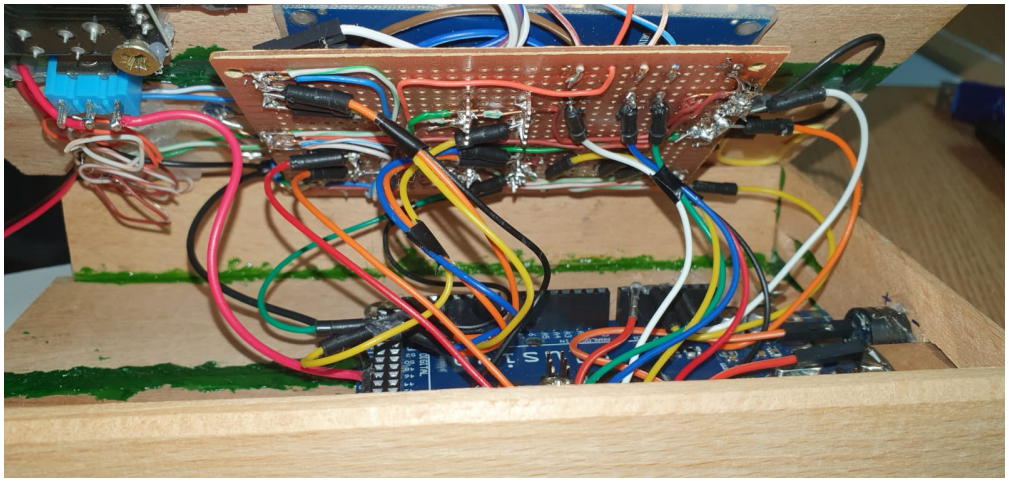
- Montarea componentelor în cutie:

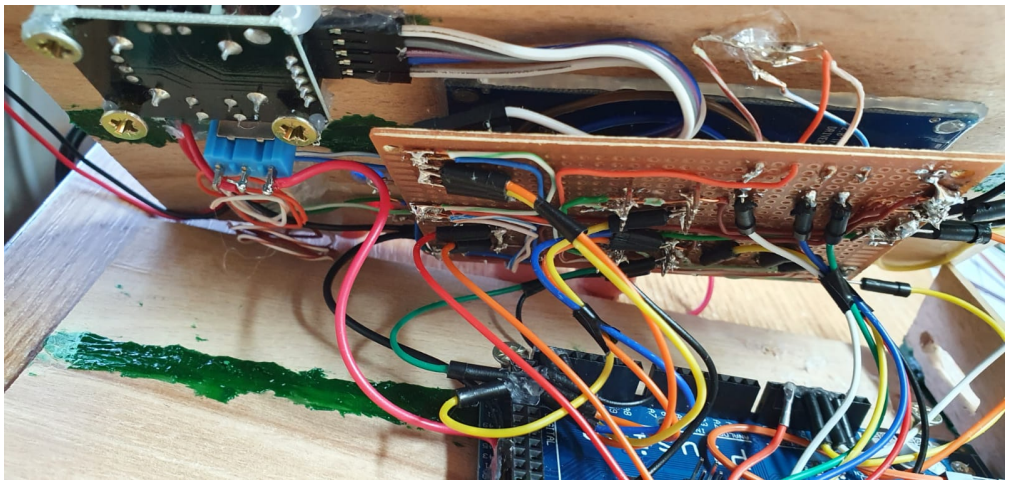


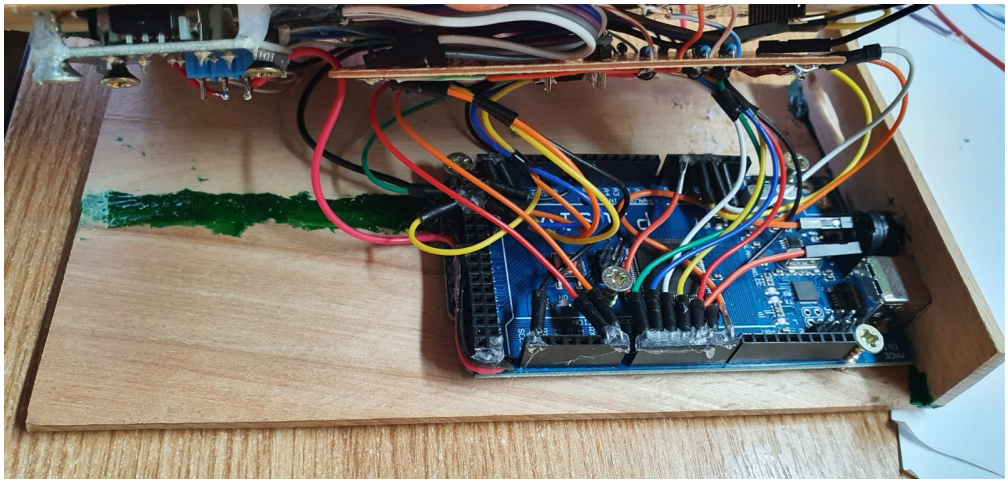




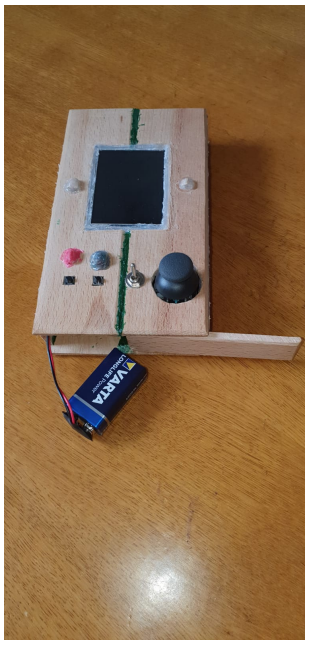
- Realizarea tuturor conexiunilor pe cablajul de test (lipirea tuturor firelor):

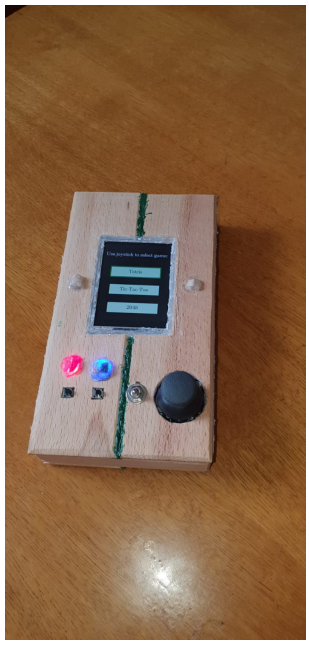






- Forma finală a proiectului:





Concluzii

Proiectul a fost o oportunitate de a experimenta implementarea unor jocuri clasice, simple, atât din perspectivă software de nivel înalt (logica jocului), cât și dintr-o perspectivă software de nivel jos (desenarea pieselor de Tetris din figuri geometrice simple / updatarea imaginii de pe ecran încât întârzierea produsă de această acțiune să nu reducă semnificativ calitatea experienței de joc - ștergerea și redesenarea întregului ecran la fiecare nou cadru nu a reprezentat o soluție întrucât tranziția nu avea loc la o frecvență suficient de mare încât să devină insesizabilă pentru ochiul uman, caz în care am adoptat o tehnică de a actualiza doar părțile din imagine care s-au modificat de la ultimul cadru). De asemenea, necesitatea de a lipi fire pe un cablaj de test și a monta toate componentele într-o cutie de dimensiuni cât mai mici (reducerea spațiului ocupat pe cât posibil printr-o organizare atentă a firelor / aranjarea componentelor pentru a realiza un dispozitiv “ergonomic”) au reprezentat aspecte plăcute ale implementării proiectului.

Download

Arhiva cuprinde:

- Componenta software implementată
- Poze din timpul implementării proiectului
- Schema bloc
- Schema electrică
- Datasheet Arduino Mega
- Pinout Arduino Mega

resurse.zip

Jurnal

1. Stabilirea temei proiectului ✔
2. Stabilirea componentelor necesare / Studierea compatibilității pieselor ✔
3. Testarea individuală a componentelor ✔
4. Montarea tuturor componentelor pe breadboard ✔
5. Montarea tuturor componentelor pe cablajul de test ✔
6. Implementarea componentei software a proiectului ✔
7. Testare ✔

Bibliografie/Resurse

Resurse software:

- Codul este disponibil la: https://github.com/andreitraistaru/Game-console
- Biblioteca Ucglib (pentru LCD): https://github.com/olikraus/ucglib/wiki/reference
- https://www.w3schools.com/colors/colors_picker.asp

Resurse hardware:

- Demonstrație cu proiectul în forma finală: https://drive.google.com/file/d/1S-WoGj8A0A0UtUowCiCEAjtv1QPca7kY/view
- Scheme pentru conectarea pinilor componentelor: https://www.optimusdigital.ro/ro/
- https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html#

Export to pdf