The Binding of Ionuț

Autor: **Ionuț-Gabriel Oțelea**
Grupa: **335CA**

Introducere

Proiectul de față reprezintă o implementarea minimală a jocului *The Binding Of Isaac* păstrând ideea principala a acestuia și modul de joc, dar aducând și elemente unice de design și gameplay. Scopul extrinsec al proiectului este de a obține cât mai multe cunoștințe referitoare la utilizarea elementelor hardware prin programarea și interconectarea acestora pentru a obține produse minimale utile în diverse situații. Astfel, scopul intrinsec al proiectului *The Binding of Ionuț* este acela de a permite relaxarea utilizatorilor oferind o experiență unică de joc video.

Ca viitor inginer în calculatoare, petrecutul timpului liber jucând un joc video nu mai reprezintă doar simpla utilizare a jocului în sine, ci un proces continuu profund, cât și complex sintetizat de întrebarea *“Oare cum au implementat asta în spate?!”*. Jucând jocul original *The Binding of Isaac*, îmi doresc deci să îmi răspund la această întrebare în limitele cunoștințelor mele acumulate până la acest moment. Utilitatea proiectului în ceea ce mă privește este reprezentată de multiplele concepte de inginerie cu care trebuie să interacționez și pe care trebuie să le înțeleg. Pentru oricine altcineva proiectul reprezintă o ocazie unică de relaxare și petrecere a timpului liber într-un mod inedit.

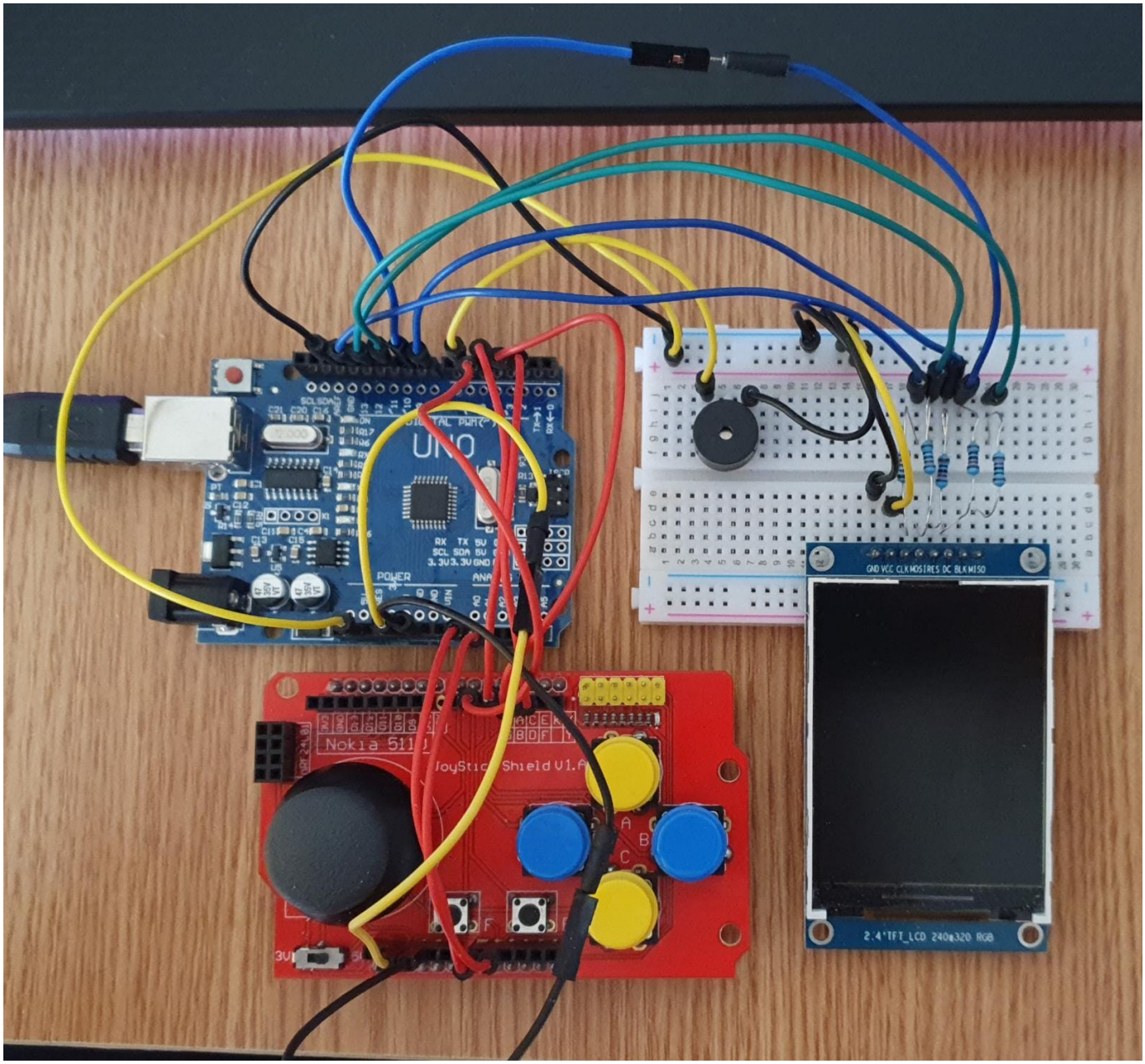
Descriere generală

Laptopul personal va asigura prin cablul USB tensiunea necesară funcționării plăcuței Arduino Uno (ATMega328p). Aceasta va alimenta la rândul său ecranul LCD (la 5V), cât și controllerul ce conține joystickul și butoanele de acțiuni (la 5V). Plăcuța Arduino va comunica cu ecranul prin SPI, iar controllerul conținând joystickul și butoanele va transmite acțiunile utilizatorului prin intermediul a 4 pini digitali și a 2 pini analogici (s-ar fi putut folosi și protocolul I2C dacă ar fi existat mai multe dispozitive de controlat). Buzzerul va fi acționat prin intermediul unui pin digital de pe plăcuța Arduino.

Utilizatorul va folosi joystickul pentru a se mișca pe hartă pe una dintre cele 4 direcții principale(sus, jos, dreapta, stânga), cât și pe diagonale. Cele 4 butoane aflate pe același controller vor permite jucătorului să lanseze proiectile înspre inamicii din apropiere. În diverse situații (atacul inamicilor, înfrângerea unui inamic, etc.) buzzerul va emite sunete specifice. Jucătorul trebuie să încerce să colecteze power-upul, să rămână în viață (să nu piardă toate viețile pe care le are la dispoziție) și să omoare fiecare inamic de pe hartă pentru a câștiga jocul. O dată ce a colectat power-up-ul de pe hartă, jucătorului îi va crește puterea de atac împotriva inamicilor săi.

Hardware Design

- Arduino UNO (ATMega328p)
- Ecran LCD mare (Modul LCD SPI de 2.8'' cu Touchscreen - Controller ILI9341 și XPT2046 (240×320 px))
- 5 * rezistori 10k (+ un rezistor opțional pentru buzzer)
- Fire de legătură mamă-tată și tată-tată
- Buzzer pasiv
- Controller conținând joystick și butoane (Joystick Shield V1.A)
- Breadboard



După cum se poate observa, joystickul și butoanele sunt de fapt plasate pe același controller (**Joystick Shield V1.A**). Butoanele corespund câte unui pin digital de pe plăcuța **Arduino UNO**, iar joystickul în sine folosește 2 pini analogici (câte unul pentru fiecare axă). Ecranul **LCD ILI9341** necesită utilizarea unor **rezistori** de 10k pe fiecare pin (există posibilitatea de a arde ecranul în lipsa acestor rezistori). **Buzzerul** ar trebui în teorie înseriat cu un rezistor pentru a obține un sunet mai fin. Totuși, am preferat să elimin rezistorul pentru un sunet cât mai puternic.

Software Design

Pentru a putea realiza un proiect ușor de dezvoltat și de extins m-am folosit intens de concepte de **OOP**. Astfel, am realizat o clasă pentru jucător (*Ionut*), una pentru proiectile (*Ball*), cât și una pentru monștri (*Monster*). Toate aceste clase au *set-ere* și *get-ere*, cât și metode de mișcare (*Ionut* și *Ball*), afișare și curățare (de pe ecran). Jucătorul unic este declarat global alături de un array de pointeri către viitorii monștri. Tot global sunt declarate și cate un array de pointeri către proiectilele

jucătorului și un altul către proiectilele monștrilor. Aceste array-uri au asociate câte un array de stare pentru a putea ști daca un pointer către un proiectil mai este valid sau nu (de exemplu, după ce proiectilul iese din scenă sau a lovit un inamic).

În cadrul funcției **setup()** se pornește mai întâi interfața serială pentru comunicarea cu ecranul LCD. Tot aici se creează monștri și sunt afișați pe ecran alături de viețile jucătorului și de un power-up. Pozițiile monștrilor și ale power-upului sunt pseudo-aleatoare (va exista mereu câte un monstru undeva pe fiecare latură a ecranului înafară de latura dreaptă unde vor fi câte 2). Se stabilesc modurile de utilizare ale diferiților pini (INPUT, OUTPUT) și se afișează scorul inițial.

În funcția principală **loop()** fiecare proiectil activ este curățat, mișcat și reafișat. Apoi, se verifică coliziunea acestuia fie cu monștri dacă este vorba de proiectilele jucătorului, fie cu jucătorul dacă este vorba de proiectilele monștrilor. Această verificare se realizează în funcția **check_collision()** care tratează coliziunea separat în funcție de originea proiectilului. În sine tratarea coliziunii este de tip **circle-collision**, aceasta pretându-se extrem de bine formei rotunde a proiectilelor. Funcția **check_collision()** se asigură după caz și de scăderea vieților jucătorului sau ale monștrilor loviți și de acționarea buzzerului în cazul detectării coliziunii. Atunci când un monstru rămâne fără viață este curățat de pe ecran iar scorul jucătorului este crescut și reafișat.

Monștri lansează aleator proiectile în direcția opusă laturii pe care se află. Pentru ca jucătorul să tragă se citește valoarea de pe **pinii digitali** corespunzători butoanelor (A, B, C, D). Dacă se observă că butonul este apăsat și a trecut suficient timp de la ultima lansare a unui proiectil, atunci se creează un nou obiect de tip *Ball* lansat în direcția de tragere. În ceea ce privește mișcarea jucătorului, sunt citite permanent valori discrete de pe **pinii analogici**. Dacă valorile diferă față de ultimele valori înregistrate înseamnă că joystick-ul a fost mutat și că jucătorul trebuie să se miște pe ecran în concordanță. Valorile noi ale pozitiei sunt updatate în cadrul obiectului de tip *Ionut*.

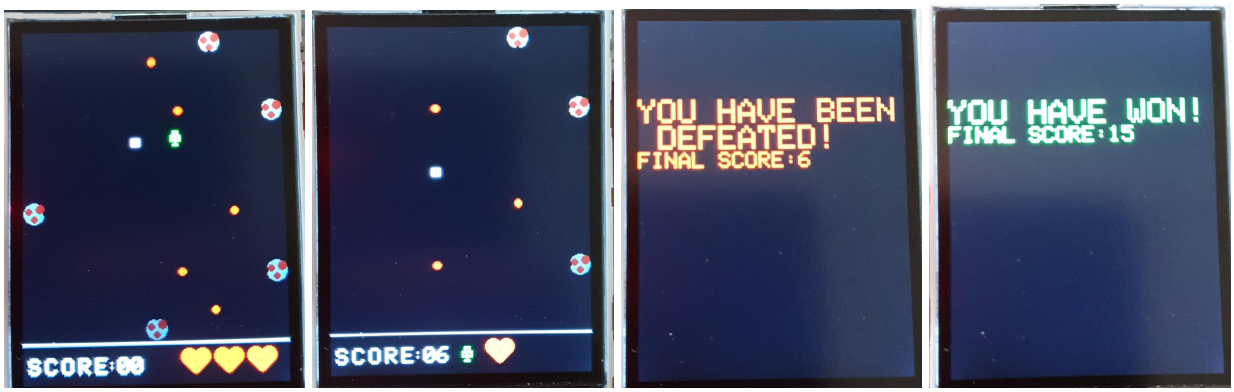
Dacă jucătorul s-a mișcat acesta este mai întâi șters de pe ecran apoi reafișat la poziția nouă. Tot în funcția **loop()** se verifică și coliziunea cu power-upul de pe hartă (nefiind o coliziune asemanătoare celor cu proiectile). În final există 2 funcții pentru tipurile de final de joc. În primul caz, dacă nu mai există monștri pe hartă, se apelează **final_win()** care va afișa victoria jucătorului și scorul final al acestuia. Altfel, dacă jucătorul a rămas fără vieți, se va apela **final_defeat()** care îl va anunța pe jucător că a pierdut jocul.

Realizare și încărcarea codului pe plăcuța Arduino s-a făcut cu ajutorul aplicației *Arduino*. Bibliotecile folosite au fost *SPI.h* (pentru comunicarea prin intermediul protocolului SPI cu ecranul LCD), *Adafruit_GFX.h* și *Adafruit_ILI9341.h* pentru desenarea diverselor forme și caractere pe ecranul ILI9341 într-un mod facil și eficient.

Rezultate obținute

Prin toate cele prezentate am reușit implementarea unui joc minimal, plăcut și ușor de jucat. Mai jos se poate găsi un link către un scurt demo al jocului conținând atât o victorie, cât și o înfrângere.

Demo The Binding Of Ionuț



Concluzii

Pe parcursul dezvoltării proiectului m-am lovit de limitările fizice ale componentelor utilizate. Astfel, plăcuța Arduino Uno nu poate transmite un flux de date mare, suficient de repede, către ecranul LCD (motiv pentru care jucătorul pare uneori a clipi pe ecran și fapt din cauza căruia am decis ca monștri să fie imobili). Totodată, verificarea constantă a coliziunilor (la fiecare frame), este extrem de costisitoare din punct de vedere al puterii de calcul (se realizează înmulțiri și radicali). Din acest motiv am limitat numărul de proiectile active la 5 pentru jucător și la 10 pentru monștri (Pe parcurs ce monștri mor și nu mai lansează proiectile chiar se observă o creștere a performanței jocului). Având în vedere toate acestea, a trebuit să reduc cu mult complexitatea jocului pe care o prevedeam la începutul proiectului în încercarea de a asigura o funcționalitate bună și o experiență de joc cursivă și plăcută.

Dincolo de cele amintite anterior, realizarea proiectului a reprezentat pentru mine o experiență extraordinară din care simt că am avut extrem de multe de învățat. Schimbarea unei linii de cod și modificarea imediată pe micul ecran din fața ta a întregului joc, conectarea greșită a unor pini și rugăciunea de a nu fi ars vreo componentă și pura satisfacție de a realiza prin propriile puteri un produs palpabil sunt experiențe cu totul unice pe care mă bucur nespus că mi le-a oferit acest proiect.

Download

Mai jos se găsește codul sursă realizat pentru implementarea jocului.

Cod sursă

Jurnal

- 25.04.2021: Realizarea paginii proiectului și a descrierii succinte a acestuia.
- 02.05.2021: Conectarea inițială a componentelor și verificarea funcționalităților de bază.
- 08.05.2021: Implementarea minimală a unui personaj și a mișcării acestuia.
- 16.05.2021: Adăgarea monștrilor și a scorului, cât și a proiectilelor.

- 22.05.2021: Realizarea coliziunilor și cuplarea tuturor componentelor software.
- 30.05.2021: Finisarea documentației și adăugarea unui demo al jocului.

Bibliografie/Resurse

the_binding_of_ionut.pdf
Lab2 PM
Lab4 PM
Lab5 PM
Bibliotecă ecran LCD
Cod exemplu ecran LCD
Cod exemplu Joystick Shield
Pinout Joystick Shield