Andrei TUDOSE (66871) - Flappy Something

Autorul poate fi contactat la adresa: **Login pentru adresa**

Introducere

Proiectul consta intr-un joc asemanator cu Flappy Bird pe LCD, acompaniat de sunetele de pe Buzzer.

Descriere generală

Utilizatorul va putea folosi 2 butoane:

- unul prin care va face „pasarea” sa sara (+ reporni jocul)
- altul prin care va trage in inamici

Depasirea unui obstacol/sfarsitul jocului/lansarea unui proiectil vor fi semnalate printr-un sunet scos de buzzer.

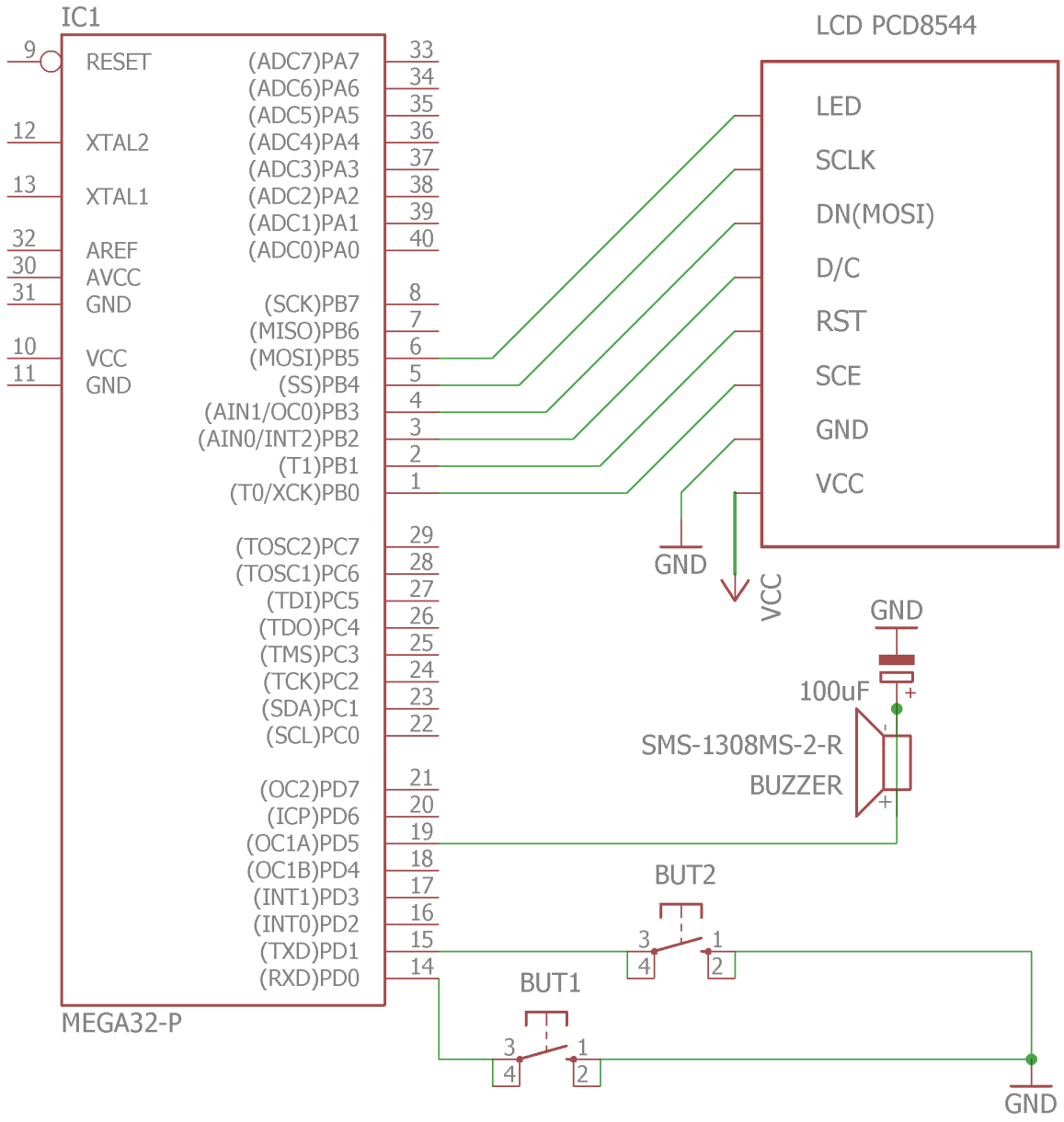
Intre anumite obstacole vor aparea si inamici. Acestia pot fi doborati prin lansarea unui proiectil.

Hardware Design

Lista piese:

- ATMega324
- Componente de baza
- 2 Butoane
- LCD cu Controller PCD8544
- Rezistenta 1k pentru LED-ul LCD-ului
- 12 Fire mama-mama + pini tata-tata
- Buzzer (+condensator 10uF)

Schema electrica:



Software Design

Mediu de Dezvoltare:

- API-ul WinAVR si editorul Programmer's Notepad winavr.sourceforge.net
- Bootloader-ul bootloader

Biblioteci 3rd-party:

- biblioteca LCD-ului 37

Functionalitati:

1. Avion(player) care coboara treptat si urca la apasarea unui buton
2. Obstacole care vin permanent catre player
3. Scorul creste la depasirea unui obstacol (semnalata printr-un sunet)

4. Lansare de bullet-uri
5. Inamici aflati pe unele din obstacole
6. Coliziuni
7. Senzatia de miscare

Implementare

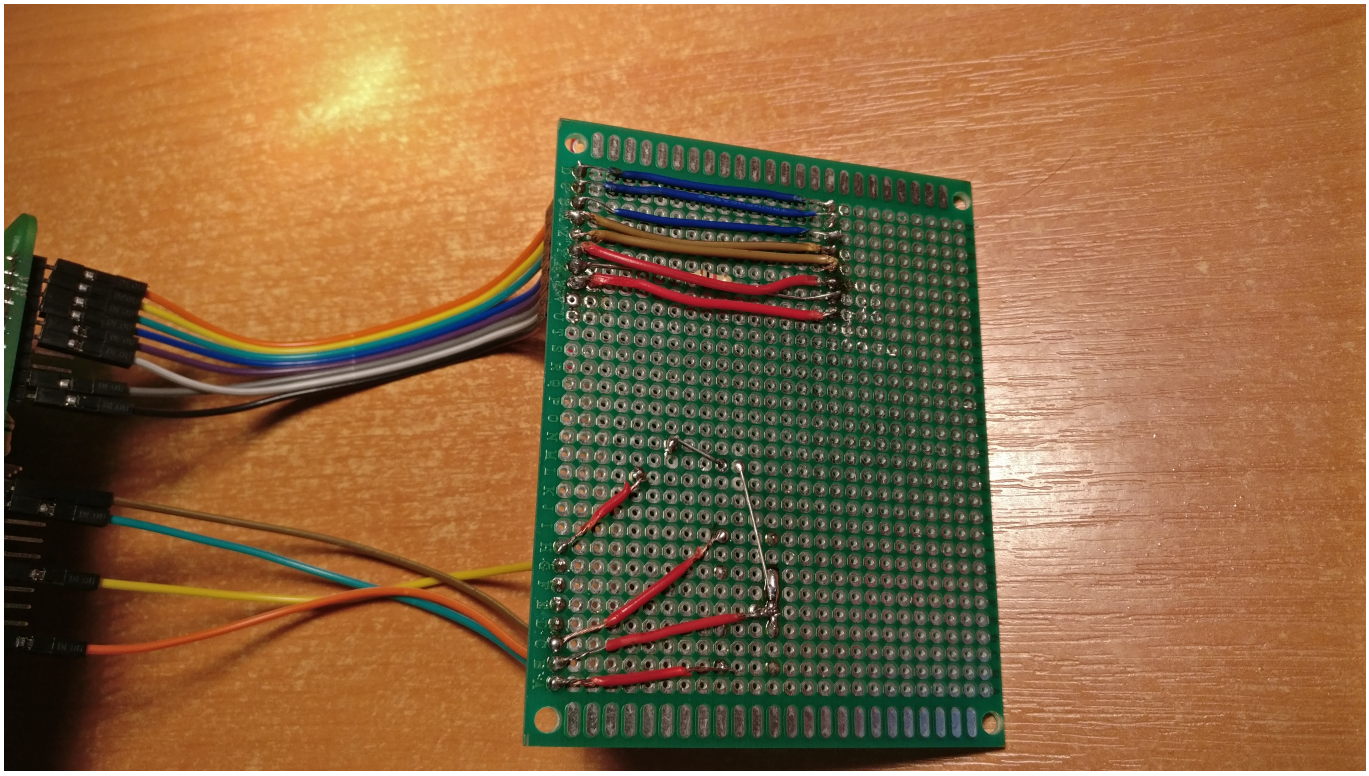
Am setat porturile corespunzatoare in lcd.h. Restul implementarii o fac in main.c Inainte de inceperea jocului exista 3 ecrane in care afisez numele jocului, numele meu si instructiunile de joc. Trecerea de aceste ecrane se face apasand butonul din stanga (legat la PD0). Aceste ecrane sunt afisate in cilcuri while(1), oprite la apasarea butonului. O data ce player-ul a murit, se afiseaza scorul si se reia jocul prin apasarea aceluiasi buton.

1. Am creat avionul folosind o matrice 11×11 de 1 si 0, unde forma avionului e data de valorile de 1. Setez pixelii corespunzatori cu ajutorul functiei set_pixel(x, y, culoare) - preluata de la alte proiecte din anii anteriori. Avionul coboara cu cate 2 pixeli la fiecare frame. O data cu apasarea butonului de pe portul PD0 (butonul din stanga), avionul urca treptat cu pana la 10 pixeli. Se creeaza astfel iluzia de “zbor”.
2. Pentru obstacole, am creat un array cu 3 elemente de tip obstacol. Acest vector se actualizeaza cu un obstacol nou de fiecare data cand unul din ele iese prin partea stanga a ercanului. La un moment dat pot fi maxim 3 obstacole active (pe ecran). Un nou obstacol se adauga folosind functia add_new_obstacle(index in vector, type), unde type este un numar random intre 0 si 4 (exista 5 tipuri de obstacole: cu gap-ul mai sus, mai jos etc). Alegerea tipului se face in acesta functie cu ajutorul unui switch (tipul consta doar in modificarea ordonatei obstacolului).
3. Scorul creste o data ce un obstacol a fost depasit. Aceasta depasire e semnala cu un sunet scos de buzzer. Acest lucru e realizat prin apelarea functiei buzzer_morse(frecventa) din laboratorul 0. Un obstacol isi mareste viteza de deplasare o data cu atingerea unui scor de 10. Un obstacol are o latime de 10 pixeli.
4. La apasarea butonului din dreapta (cel legat la PD1) se lanseaza unui bullet. Un obiect de tip bullet, care exista deja in joc pe pozitia avionului (dar nu este vizibil pt ca nu e activ), devine activ si incepe sa se deplaseze catre dreapta (creste abscisa). La lansarea bullet-ului, buzzer-ul scoate un sunet de frecventa joasa.
5. Unele obstacole pot avea si monstri (hasMonster = 1). Acesti monstri sunt ficsi si desenati prin acelasi mod in care e desenat si avionul (cu alta matrice de 1 si 0). Daca un bullet loveste monstrul, acesta dispare. Daca player-ul loveste monstrul, jocul se incheie. Incheierea e semnalata si ea prin sunete scoase de buzzer.
6. Am incercat sa fac coliziunile cat mai precise. Nu am folosit metoda sferei (sau altele de genul) ci pur si simplu am hardcodat valori pe care le-am adaugat/scazut din punctul din care incepe desenarea obiectului pt care se trateaza coliziunea.
7. Senzatia de miscare pe care i-o dau obiectelor este facuta prin desenarea obiectelor cu negru in prima faza, efectuarea unui delay de cateva milisecunde, apoi redesenarea cu culoare alba (il fac sa dispara) si in cele din urma cresterea/scaderea coordonatelor.

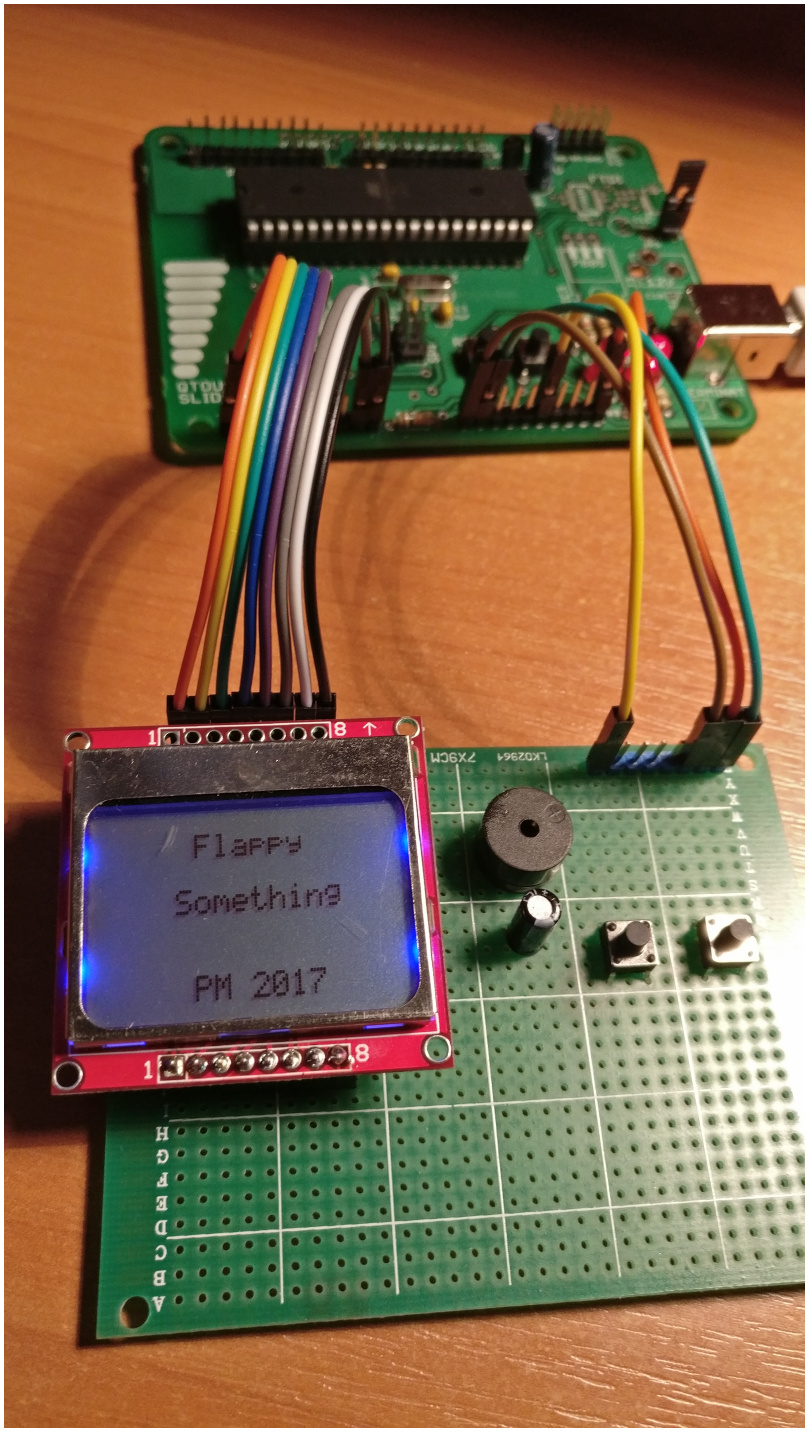
In caz ca am uitat sa acopar ceva, se pot observa comentariile din cod.

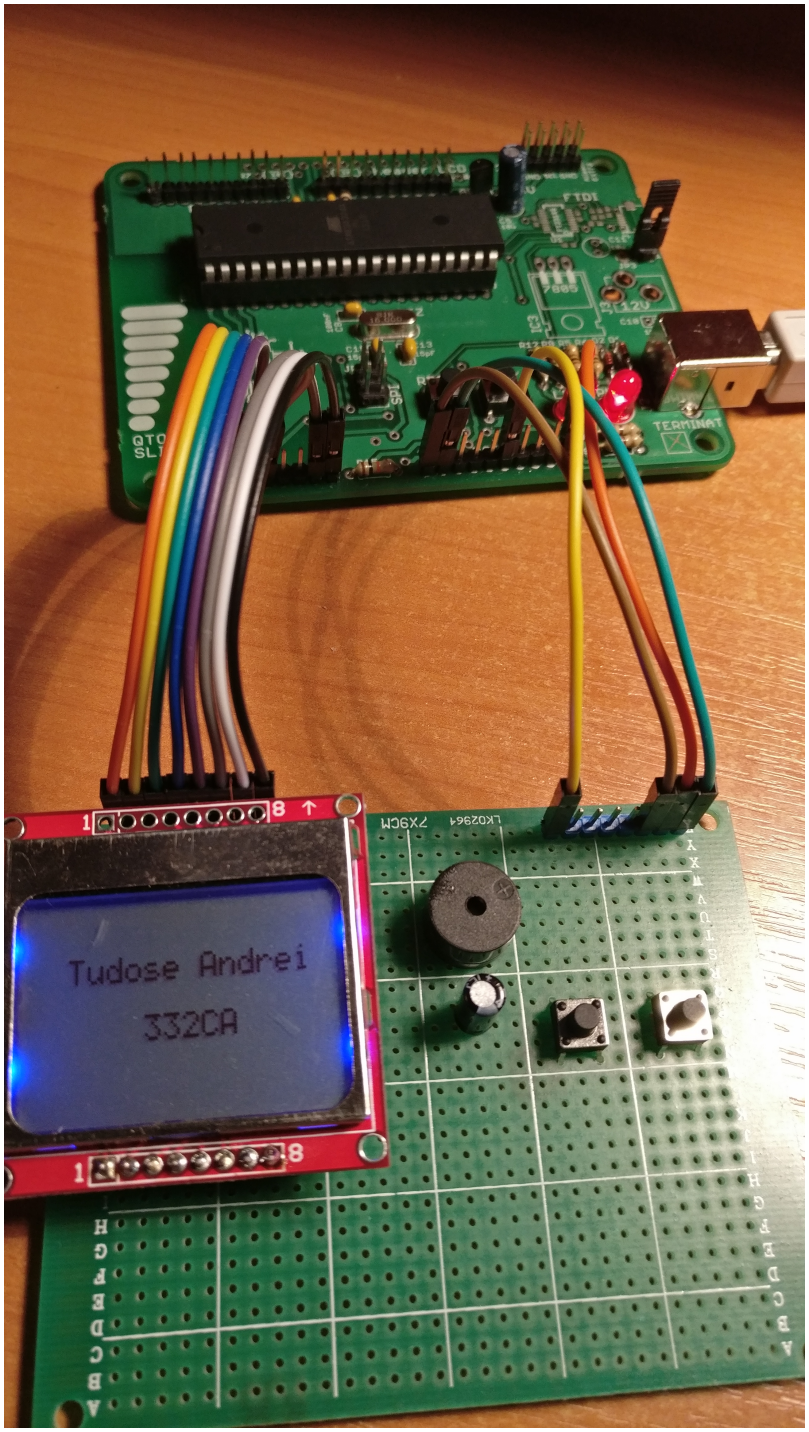
Rezultate Obţinute

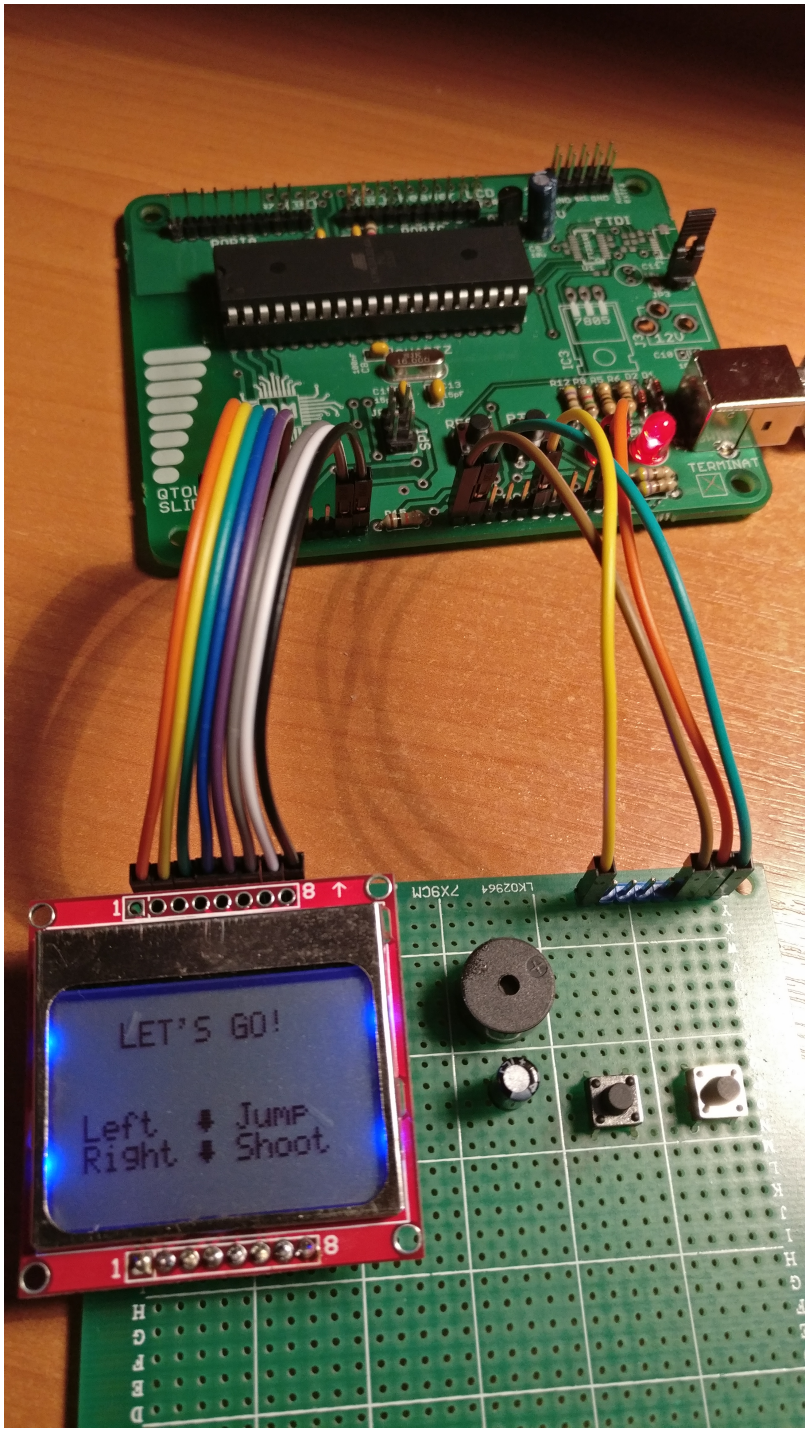
Spatele placutei de test unde am lipit legaturile:

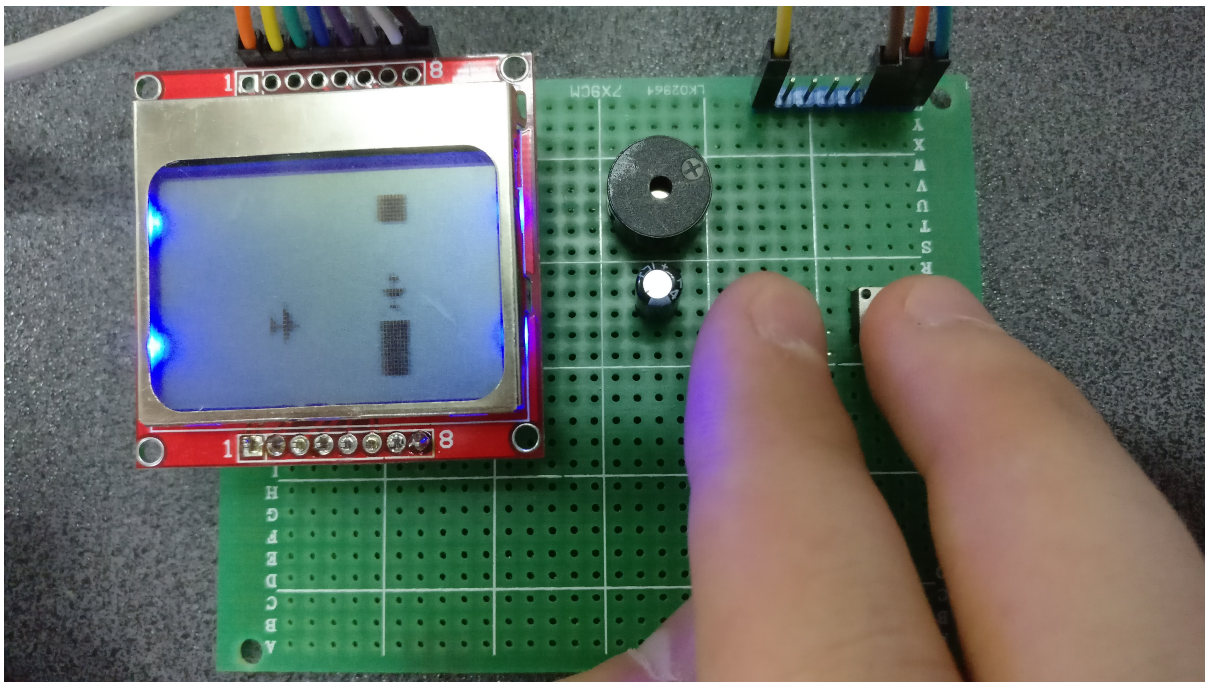


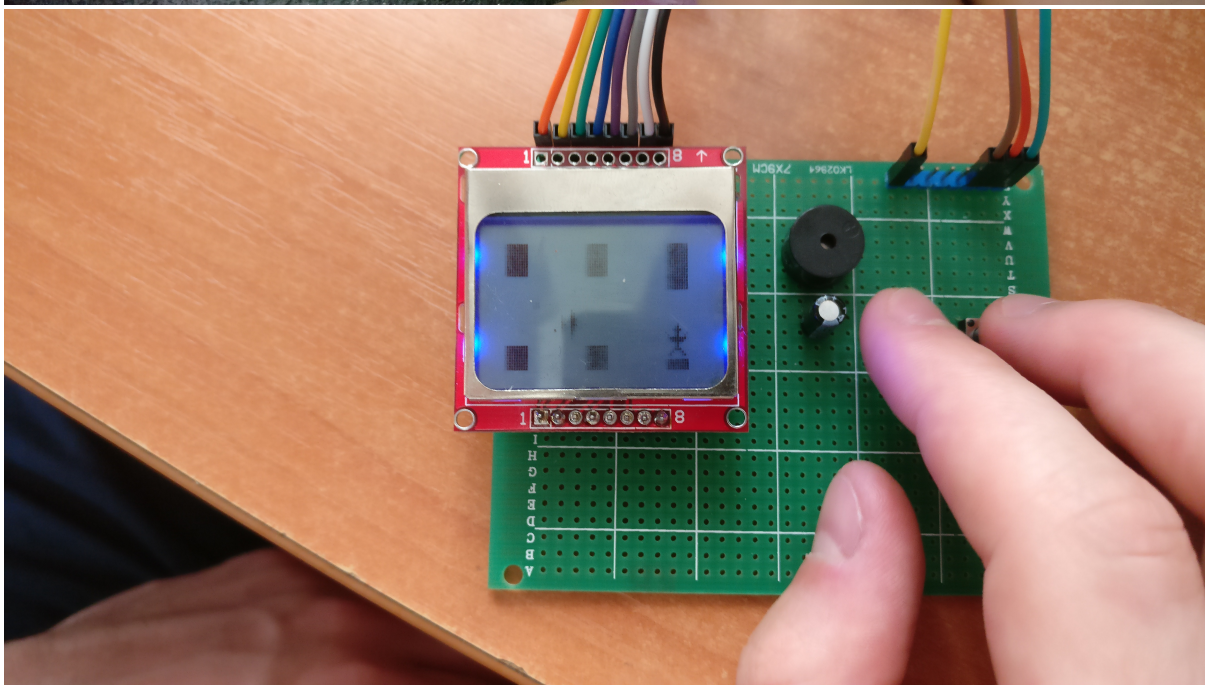
Rezultatul final:

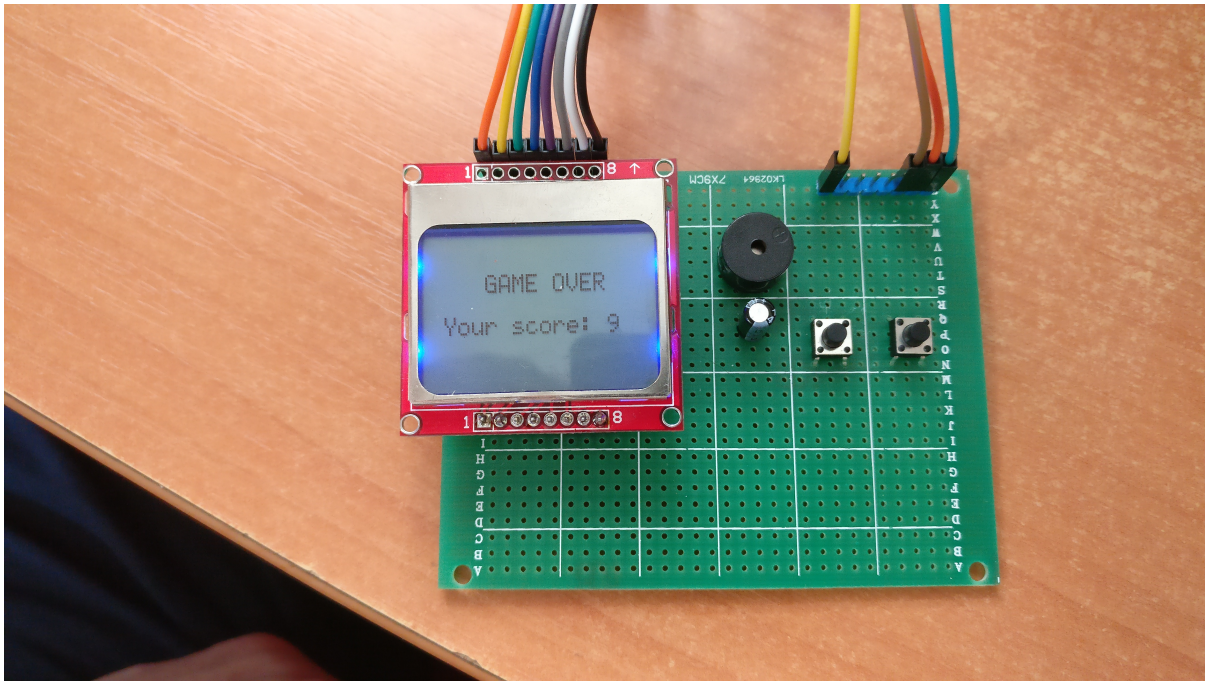

Flappy
Something
PM 2017


Tudose Andrei
332CA


LET'S GO!
Left Jump
Right Shoot
FTDI
TERMINAT







Mini exemplu de functionare: FWPtuLs98og

Concluzii

- Dupa cateva zile de munca (adunate in total) pot sa spun ca am reusit sa realizez un joc functional, fara bug-uri sau alte probleme.
- A fost mult mai usor decat ma asteptam la inceput.
- Jocul este unul simplist, dar cred ca am atins scopul acestui proiect.
- Am invatat multe lucruri in timpul realizarii acestui proiect, cu care nu as fi ramas doar dupa terminarea laboratoarelor.

Download

Codul sursa: 332ca_tudoseandrei_codsursapm.zip

Jurnal

- 23 Aprilie - Am incarcat pe wiki tema proiectului, lista de piese și schema bloc
- 26 Aprilie - Terminare de lipit placa de baza
- 6 Mai - Schema electrica a circuitului
- 10 Mai - Terminat de lipit pe placa de test LCD, butoane & Buzzer
- 17 Mai - Incepere implementare Software
- 23 Mai - Finalizare Proiect

Bibliografie/Resurse

Resurse Software

- biblioteca pentru LCD: 37
- laboratorul 0 PM: lab0

Resurse Hardware

- datasheet LCD: Nokia5110.pdf
- datasheet ATMega324: http://cs.curs.pub.ro/wiki/pm/_media/doc8272.pdf

Documentația în format PDF