X și O

Autor: Anca-Maria BUZATU - 336CB

Introducere

Implementarea jocului X și O, având doi jucători reprezentați de led-uri verzi, respectiv roșii.
Jocul are două moduri:

- 2 jucători: RED vs GREEN (single player)
- 1 jucător: RED vs GREEN Arduino (two players)

Poziția se alege prin intermediul butoanelor, iar scorul fiecărei runde va fi afișat pe un ecran LCD.
Proiectul are ca scop realizarea unui joc folosind cunoștințele acumulate în cadrul laboratoarelor.
Am ales acest proiect, deoarece X și O este un joc clasic și foarte îndrăgit și îl voi folosi pentru a-mi distra verișorii mai mici de sărbători.

Descriere generală

Schema Bloc:

Intrări:

- butonul Up
- butonul Down
- butonul Left
- butonul Right
- butonul Submit

Ieșiri:

- cele 9 led-uri verzi
- cele 9 led-uri roșii
- ecranul LCD

Jucătorul va apăsa butoanele up, down, right, left pentru a ajunge pe poziția dorită. Aceste acțiuni sunt interceptate de către placa Arduino care va schimba starea led-ului corespunzător fiecărei poziții. Când jucătorul decide poziția finală, apasă butonul submit, astfel anunțându-l pe celălalt. Schimbarea jucătorului va fi afișată și pe ecranul LCD.

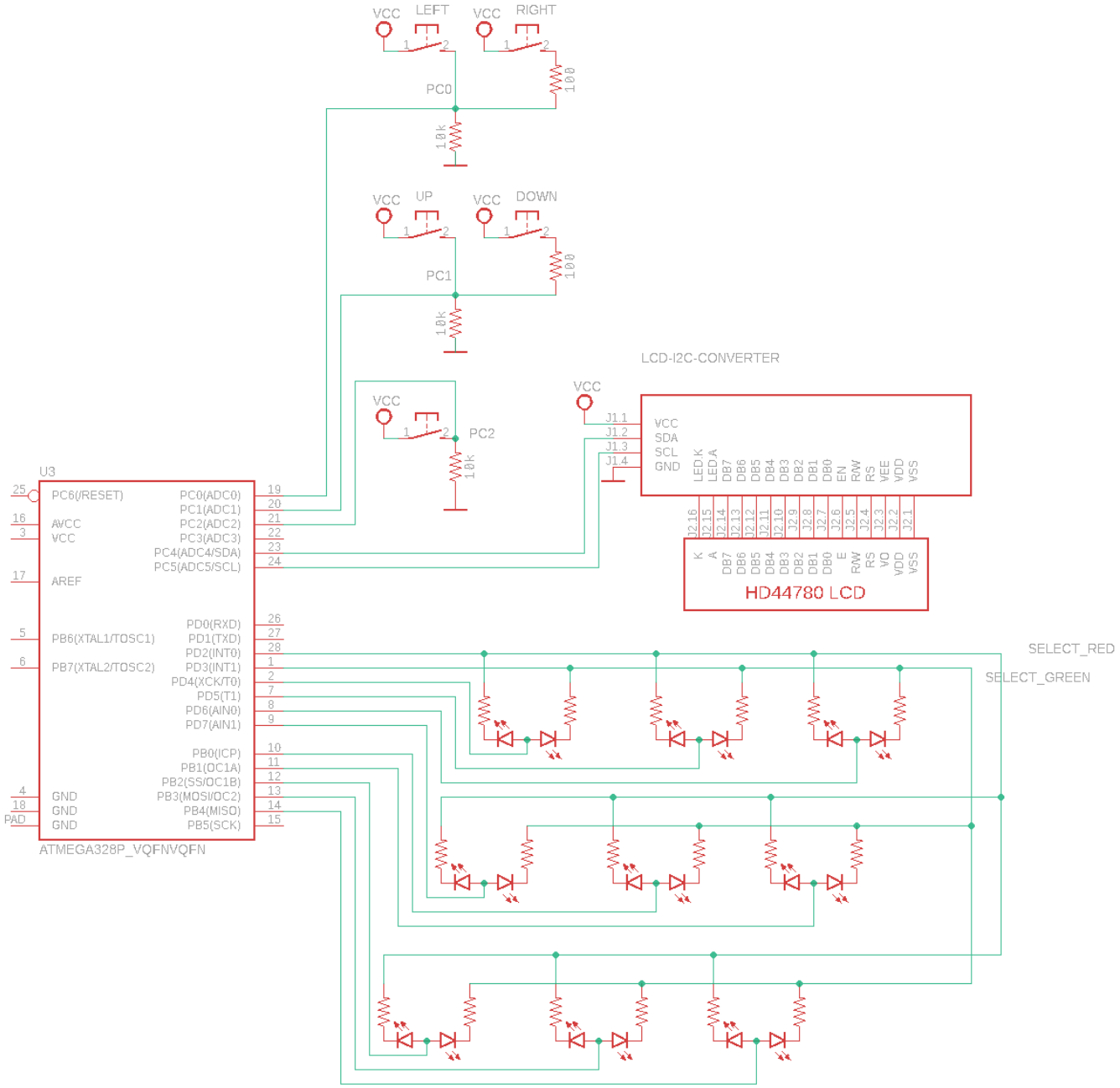
În funcție de rezultatul rundei se actualizează scorul care, de asemenea, va fi afișat pe ecranul LCD.

Hardware Design

Listă de piese:

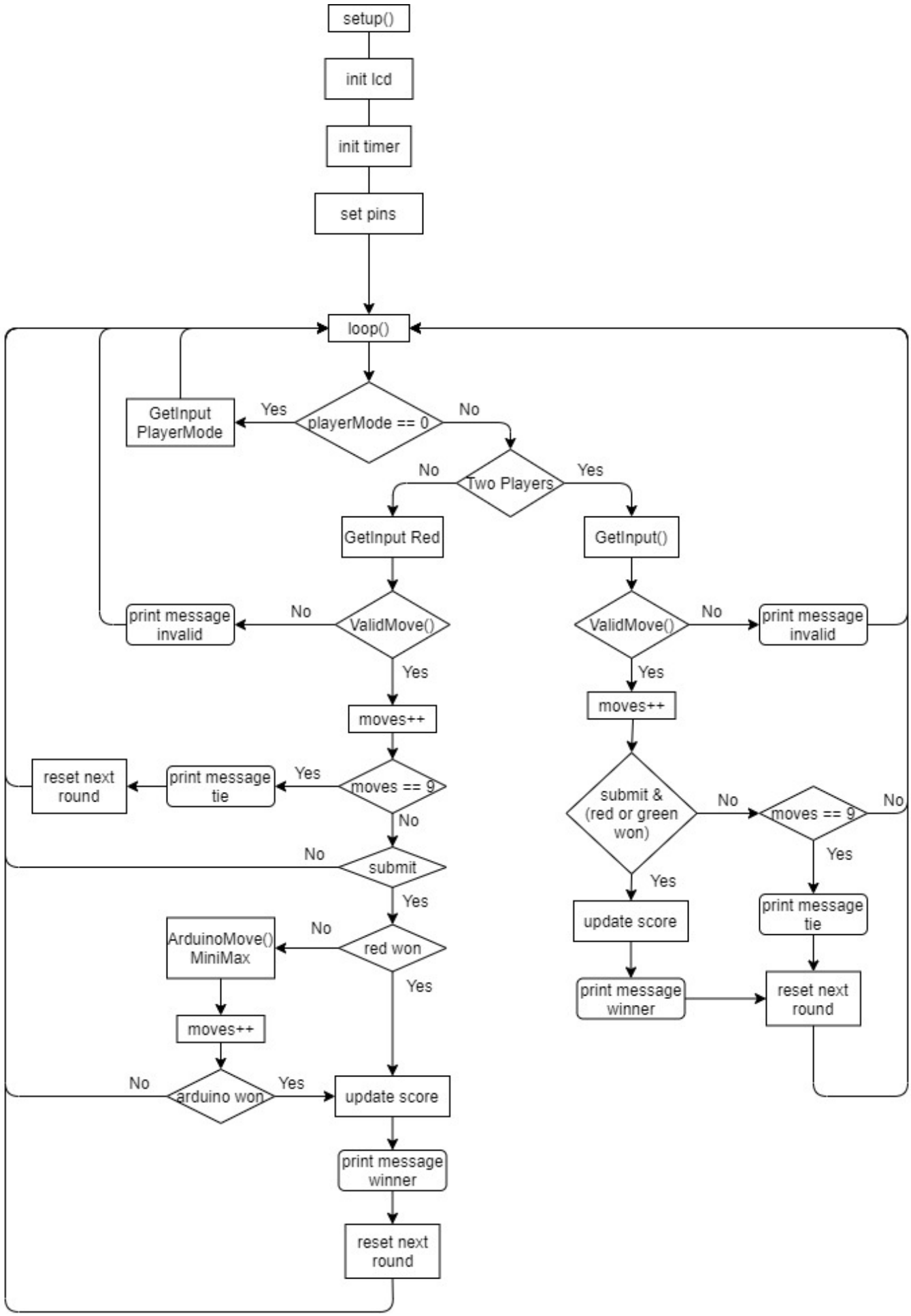
- Arduino UNO
- Fire
- Rezistențe
- BreadBoard
- Leduri verzi 9, leduri rosii 9
- Butoane 5
- Ecran LCD

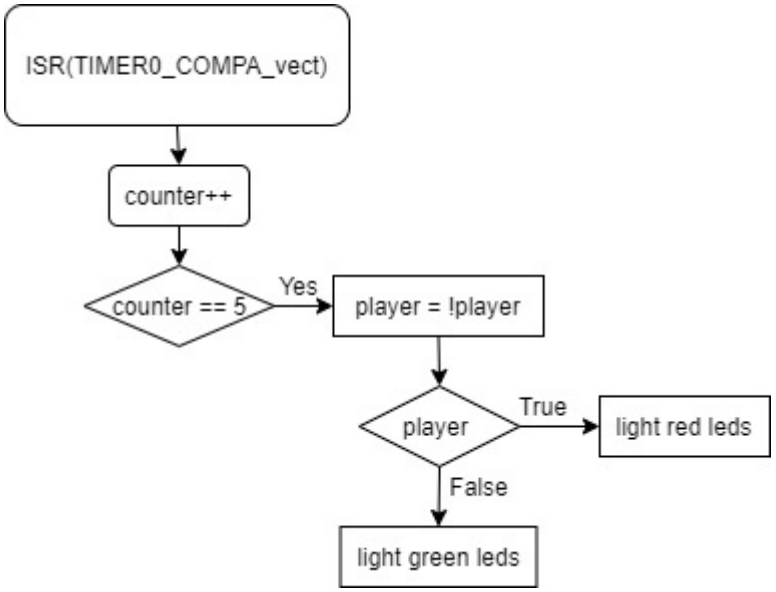
Schema Electrică:



Software Design

Diagramă de activitate


setup()
init lcd
init timer
set pins
loop()
GetInput PlayerMode
Yes
playerMode == 0
No
No
Two Players
Yes
GetInput Red
GetInput()
print message invalid
No
ValidMove()
ValidMove()
No
print message invalid
Yes
Yes
moves++
moves++
reset next round
print message tie
Yes
moves == 9
submit & (red or green won)
No
moves == 9
No
No
submit
Yes
Yes
Yes
update score
print message tie
ArduinoMove() MiniMax
No
red won
print message winner
reset next round
Yes
moves++
No
arduino won
Yes
update score
print message winner
reset next round



Implementare

Pinii folosiți

- playerSelectRed: 3 - PD3
- playerSelectGreen: 2 - PD2
- led11: 4 - PD4
- led12: 5 - PD5
- led13: 6 - PD6
- led21: 7 - PD7
- led22: 8 - PB0
- led23: 9 - PB1
- led31: 10 - PB2
- led32: 11 - PB3
- led33: 12 - PB4

Pozițiile jucătorilor sunt păstrate în matrice 3×3: ledStateRed, ledStateGreen.
Pin-urile reprezentative pentru led-uri sunt configurate ca ieșiri în setup(). Led-urile sunt aprinse într-o rutină de întreruperi. Am folosit Timer0 în modul normal cu top 0xFF. Mai întai sting toate ledurile și configurez pinii pe valoarea LOW. Apoi cu ajutorul unei variabile palyer:

- dacă este true - aprind ledurile roșii: pinul playerSelectGreen va fi configurat ca input, playerSelectRed va fi configurat ca output, iar pentru led-urile care au în matricea ledStateRed valori diferite de 0, pinii vor fi setați pe HIGH.
- dacă este false - aprind ledurile verzi: pinul playerSelectGreen va fi configurat ca output, playerSelectRed va fi configurat ca input, iar pentru led-urile care au în matricea ledStateGreen valori diferite de 0, pinii vor fi setați pe HIGH.

Pentru a nu schimba player la ficare întrerupere am adăugat o variabila care este incrementată în cadrul rutinei. Când variabila ajunge la 5, se schimbă playerul și se aprind ledurile acestuia.

Inputul este recepționat în funcția GetInput() care citește valorile pe pinii analogici A0, A1, A2. Pinii sunt împărțiți astfel:

- A0 linia 1: stânga, dreapta
- A1 linia 2: sus, jos
- A2 submit

Pentru a diferenția butoanele stânga-dreapta, respectiv sus-jos am mai adaugat câte o rezistență de 100 ohm. Astfel pentru butonul din stânga este citită valoarea 1023, iar pentru cel din dreapta valoarea 1013.

Invalid moves

- jucătorul curent dă submit pe o poziție ocupată de celălalt jucător sau o poziție aleasă într-un pas anterior
- această mutare duce la depășirea matricei 3×3

Când se apelează funcția ValidMove, se verifică dacă mutarea este validă și se completează în matricea fiecarui jucator:

- 1, dacă nu este deja aprins led-ul
- 2, dacă led-ul deja a fost aprins, jucătorului nu îi este permis să aleagă această poziție, dar poate trece peste ea pentru a ajunge într-o celula disponibilă.

Pentru modul cu un singur jucător există și matricea ledState folosită în cadrul algoritmului minimax:

- 0, poziție liberă
- 1, poziție ocupată de red
- 2, poziție ocupată de arduino.

ArduinoMove() realizeză mutarea jucătorului vrede când se alege modul single player. Folosește algoritmul minimax, care evaluează toate mutările posibile și alege soluția care are cele mai multe șanse de câștig.
Dupa apăsarea butonului submit, se verifică dacă există un caștigător, caz în care sunt stinse led-urile pierzătorului, se actualizează scorul și se resetează jocul pentru runda următoare. Dacă nu există câștigător și numărul de mutări este egal cu 9, este egalitate și se resetează runda. Altfel așteaptă următoarea mutare.

Când scorul însumat al celor doi jucători este egal cu o valoarea maximă, impară, predefinită, se resetează iarăși jocul și se poate alege înca o dată modul jocului.

Mediu de dezvolare

- Arduino IDE

Biblioteci folosite

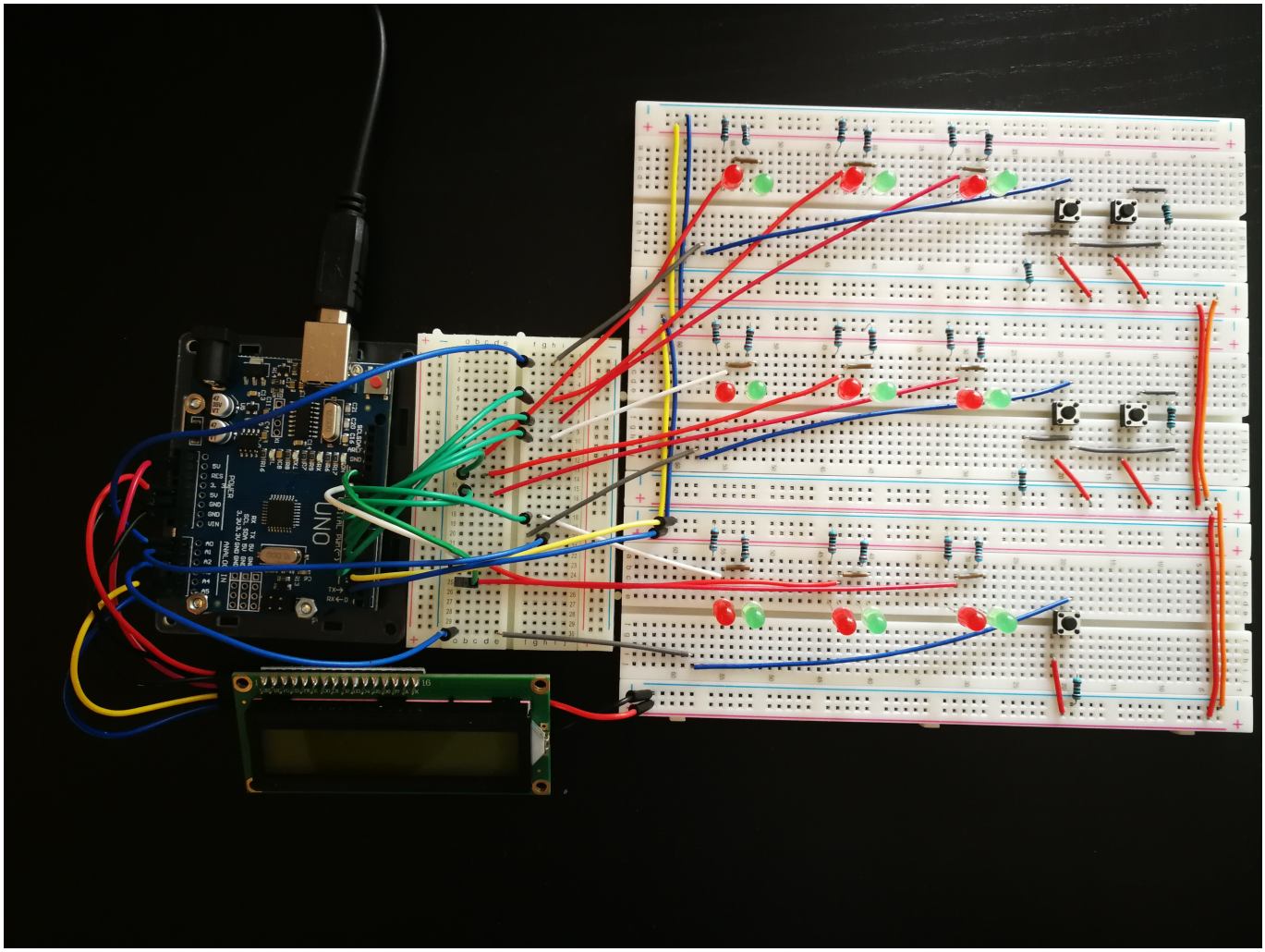
- LiquidCrystal_I2C - pentru ecranul LCD cu modul I2C

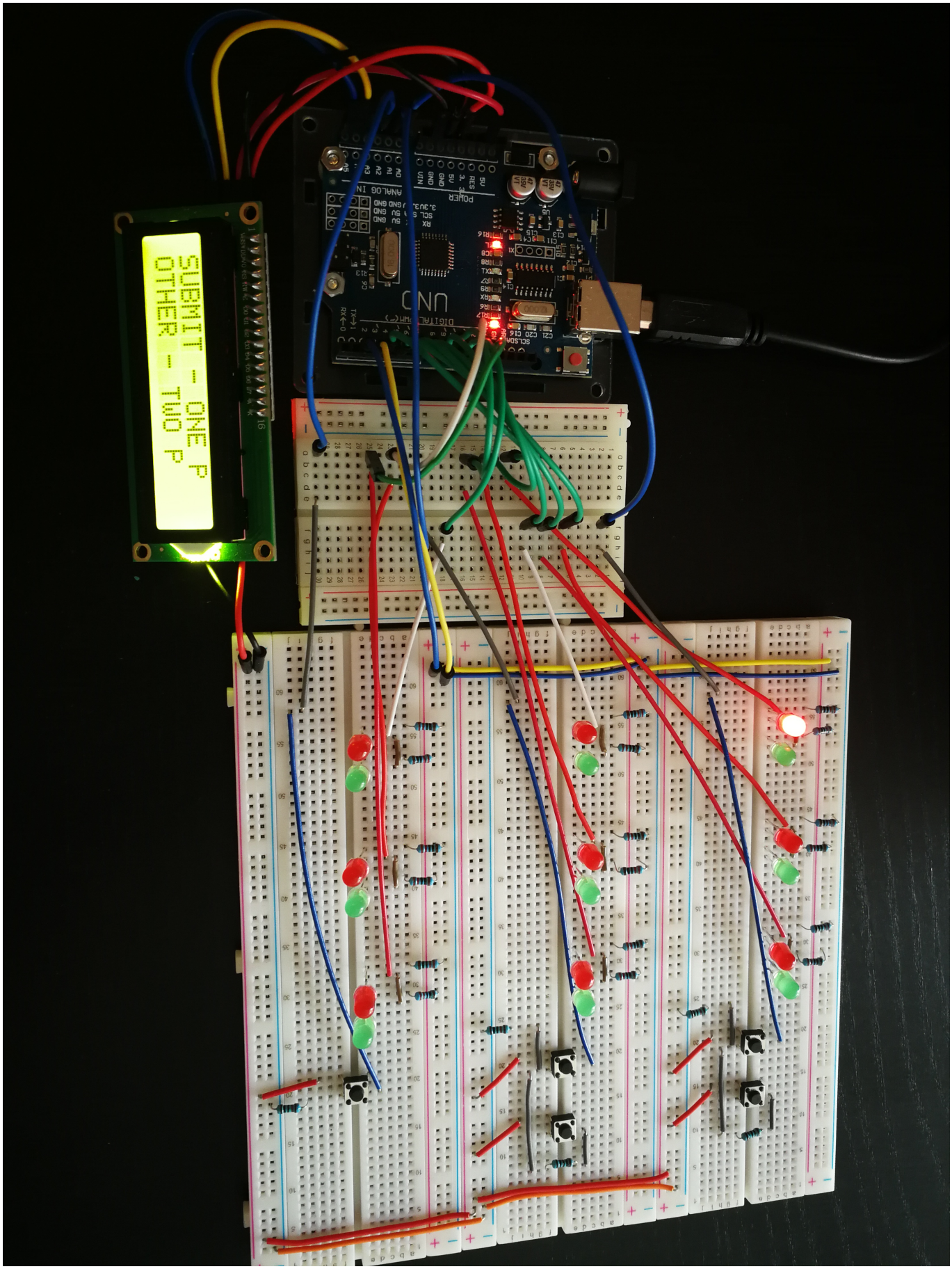
Algoritmi folosiți

- Algoritmul Minimax - pentru ca arduino să facă mutarea optimă astfel încât să nu piardă

Rezultate Obținute

După ce se alege modul single player sau two players, jocul continuă cu câte o rundă nouă până când există un număr de câștiguri egal cu MAX_ROUNDS definit cu valoarea 3. Se va afișa pe ecran câștigătorul final.
Demo




SUBMIT - ONE P
OTHER - TWO P

Concluzii

Proiectul a fost destul de simplu de realizat, jocul are implementate toate funcționalitățile menționate în descriere.
Partea hardware am încercat să o realizez cât mai ordonat, chiar dacă am folosit breadboards și un număr mare de pini. O problemă ar fi faptul că există momente când valoarea citită cu analogRead() nu este cea așteptă, iar prima mutare arduino necesita mai mult timp de calcul.
În final, să lucrez cu un ecran lcd a fost foarte simplu datorită bibliotecii ajutătoare, la fel si partea de debugging, datorită multitudinii de led-uri.

Download

Cod sursă și README

Bibliografie/Resurse

- Datasheet LCD și I2C
- Algoritm Minimax pentru X și O
- Tutorial I2C LCD

Export to PDF