Documentație de proiect

Arduino Battleships

Proiect realizat de: Anton Ioana-Carina

Grupa 30235

# Enunțul problemei

Simularea jocului ,,Battleships’’ utilizând un keypad și un scut LCD. Generarea unei matrici 4x3 sau 3x3 umplute cu 0-uri. Plasarea aleatorie în matrice a cifrei 1 de un număr predefinit de ori. Utilizatorul trebuie sa ghicească unde sunt cifrele de 1 prin apăsarea butoanelor de pe keypad. Să se definească un număr de încercări și un counter (pentru cronometrare). Adăugarea unui buton pentru resetarea jocului.

# Soluția

Pentru generarea corăbiilor am utilizat o matrice de numere întregi de 3x4, iar pentru generarea aleatoare a lor am utilizat funcția random(minValue, maxValue), asigurând-mă că locațiile alese sunt distincte.

*int* numberMap*[3][4] = {0};*

*void* randomizeShips*(int noShips) {*

*int randRow = 0;*

*int randCol = 0;*

*for (int i = 0; i < noShips; i++) {*

*randRow = random(0, 3);*

*randCol = random(0, 4);*

*while (numberMap[randRow][randCol] == 1) {*

*randRow = random(0, 3);*

*randCol = random(0, 4);*

*}*

*numberMap[randRow][randCol] = 1;*

*}*

*}*

Pentru afisarea pe ecranul lcd a hărții jocului într-un mod grafic am utilizat o matrice de caractere: caracterul ‘X’ pentru locurile neatacate, iar caracterul ‘O’ pentru zonele descoperite unde se află o barcă. Functia drawMap verifică care buton a fost apăsat și verifică prin intermediul funcției checkIfShip dacă pe poziția respectivă butonului se află o corabie sau nu. În cazul în care se află atunci se incrementează variabila care ține cont de câte corăbii au fost găsite, iar în caz negativ se va decrementa variabila de încercări.

*char* battleMap*[70][70] = {{'-', '-', '-', '-', '-', '-', '-', '-', '-', '\n'}, //harta 3x4 de joc*

*{'|', 'X', '|', 'X', '|', 'X', '|', 'X', '|', '\n'},*

*{'-', '-', '-', '-', '-', '-', '-', '-', '-', '\n'},*

*{'|', 'X', '|', 'X', '|', 'X', '|', 'X', '|', '\n'},*

*{'-', '-', '-', '-', '-', '-', '-', '-', '-', '\n'},*

*{'|', 'X', '|', 'X', '|', 'X', '|', 'X', '|', '\n'},*

*{'-', '-', '-', '-', '-', '-', '-', '-', '-', '\n'},*

*};*

*void* drawMap*(int key) {*

*mylcd.Set\_Text\_Mode(0);*

*mylcd.Fill\_Screen(0x0000);*

*mylcd.Set\_Text\_colour(BLUE);*

*mylcd.Set\_Text\_Back\_colour(BLACK);*

*mylcd.Set\_Text\_Size(2);*

*if (key > 0)*

*switch (key) {*

*case 0: break;*

*case 1: checkIfShip(0, 0); break;*

*case 2: checkIfShip(0, 1); break;*

*case 3: checkIfShip(0, 2); break;*

*case 4: checkIfShip(0, 3); break;*

*case 5: checkIfShip(1, 0); break;*

*case 6: checkIfShip(1, 1); break;*

*case 7: checkIfShip(1, 2); break;*

*case 8: checkIfShip(1, 3); break;*

*case 9: checkIfShip(2, 0); break;*

*case 10: checkIfShip(2, 1); break;*

*case 11: checkIfShip(2, 2); break;*

*case 12: checkIfShip(2, 3); break;*

*case 13: //butonul de reset*

*Serial.print("Resetting!");*

*Serial.println(" ");*

*resetGame();*

*break;*

*case 14: break;*

*case 15: break;*

*case 16: break;*

*}*

*drawMapOnLCD();*

*}*

*void* checkIfShip*(int i, int j) {*

*if (numberMap[i][j] == 1) {*

*battleMap[2 \* i + 1][2 \* j + 1] = 'O';*

*shipsFound++;*

*}*

*else {*

*lives--;*

*}*

*}*

Pentru utilizarea matricii de butoane am utilizat materialul de curs, în special cursul numărul 3 unde se explica utilizarea unei tastaturi Arduino. Din acest curs am preluat funcția getKey() pentru a putea detecta care buton este apasat la un moment dat.

*int* getKey*() {*

*int key = 0;*

*for (int column = 0; column < numCols; column++) {*

*digitalWrite(colPins[column], LOW);*

*for (int row = 0; row < numRows; row++) {*

*if (digitalRead(rowPins[row]) == LOW) {*

*delay(debounceTime);*

*while (digitalRead(rowPins[row]) == LOW);*

*key = keymap[row][column];*

*}*

*}*

*digitalWrite(colPins[column], HIGH);*

*}*

*return key;*

*}*

Pentru utilizarea ecranului lcd utilizez librăriile specifice acestuia:

*<LCDWIKI\_GUI.h> //Core graphics library*

*<LCDWIKI\_SPI.h> //Hardware-specific library*

*LCDWIKI\_SPI* mylcd*(ST7735S, A5, A3, -1, A2, A4, A1, A3); //software spi,model,cs,cd,miso,mosi,reset,clk,led*

*//initializarea ecranului lcd*

*mylcd.Init\_LCD();*

*mylcd.Fill\_Screen(BLACK);*

*mylcd.Print\_String(buffer, 0, 130);//metoda principala de afisat pe ecran*

*void* drawMapOnLCD*() { //functia de afisat harta de joc*

*int rowDist = 18;*

*for (int i = 0; i < 7; i++) {*

*mylcd.Print\_String(battleMap[i], 0, rowDist);*

*rowDist += 15;*

*}*

*}*

Conditia de câștig a jocului este shipsFound==noShips(număul de corăbii găsite sa coincide cu numărul de bărci introduce pe harta de joc).

*void* checkIfWin*() {*

*if (shipsFound == noShips){*

*Serial.println("Victory!");*

*resetGame();*

*}*

*}*

Funcția de setup începe prin pornirea monitorului serial, urmănd cu inițializarea butoanelor de pe keypad. Se inițializează ecranul lcd utilizând metodele speciale din librăria componentei. În final se apelează funcția de resetare a jocului pentru a inițializa restul variabilelor și a pregăti un nou joc.

*void* setup*() {*

*Serial.begin(9600);*

*//initializarea piniilor butoanelor*

*for (int row = 0; row < numRows; row++) {*

*pinMode(rowPins[row], INPUT);*

*digitalWrite(rowPins[row], HIGH);*

*}*

*for (int column = 0; column < numCols; column++) {*

*pinMode(colPins[column], OUTPUT);*

*digitalWrite(colPins[column], HIGH);*

*}*

*//initializarea ecranului lcd*

*mylcd.Init\_LCD();*

*mylcd.Fill\_Screen(BLACK);*

*Timer1.initialize(1000000);//microseconds*

*//desenarea hartii de joc*

*//drawMap(0);*

*resetGame();*

*}*

Funcția loop preia cheia butonuli apăsat, o scrie în Serial Monitor pentru verificare, dacă o mai rămas încercări atunci se apelează funcția de executare a acțiunii în funcție de butonul apăsat, dacă nu o mai rămas încercări atunci se va atenționa jucătorul că a pierdut și se va încărca în mod automat un nou joc. În final se afișează numărul de încercări rămase și timpul trecut în sesiune, după care se verifică dacă jocul este în stadiul de câștigat sau nu.

*void* loop*() {*

*int key = getKey(); //preluarea butonului apasat*

*if (key != 0) {*

*Serial.print("Got key ");*

*Serial.println(key);*

*if (lives > 0) //daca au mai ramas incercari*

*drawMap(key);*

*else {*

*Serial.println("You lost!");*

*drawMap(13);*

*}*

*}*

*char buffer[200];*

*//counter=millis()/1000;*

*Timer1.attachInterrupt(coundAdd);*

*snprintf(buffer, 100, "Lives:%d %d", lives, counter);*

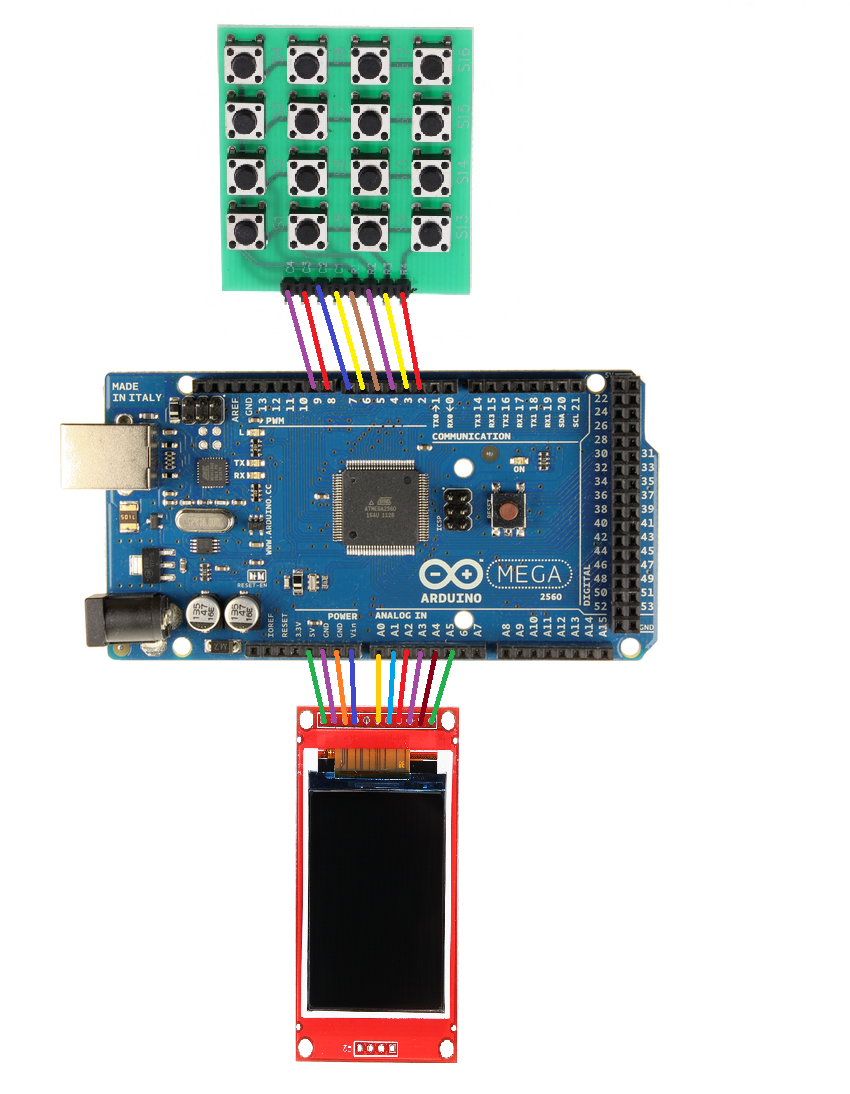
*//countAdd();*

*mylcd.Print\_String(buffer, 0, 130);*

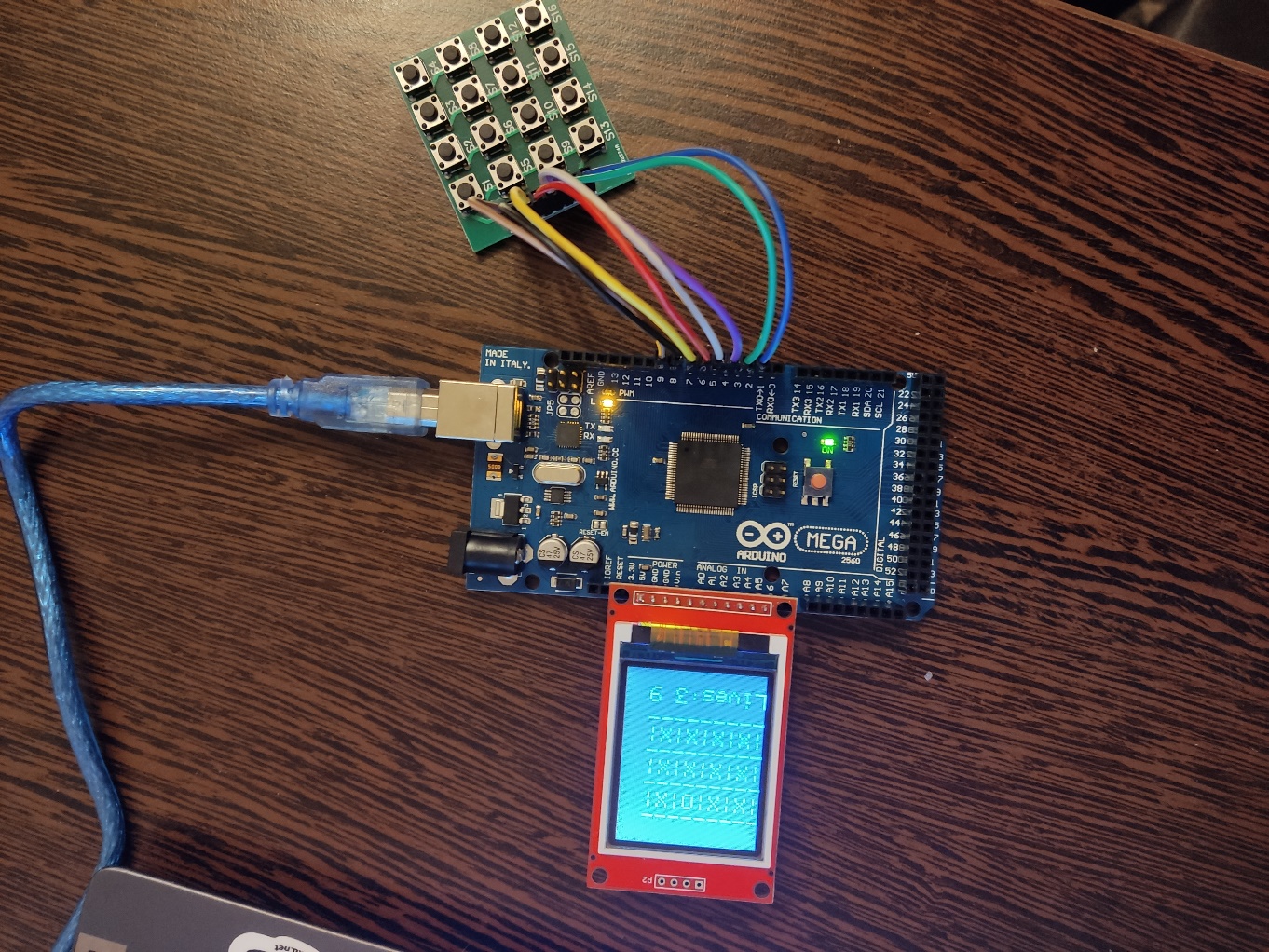
*checkIfWin();*

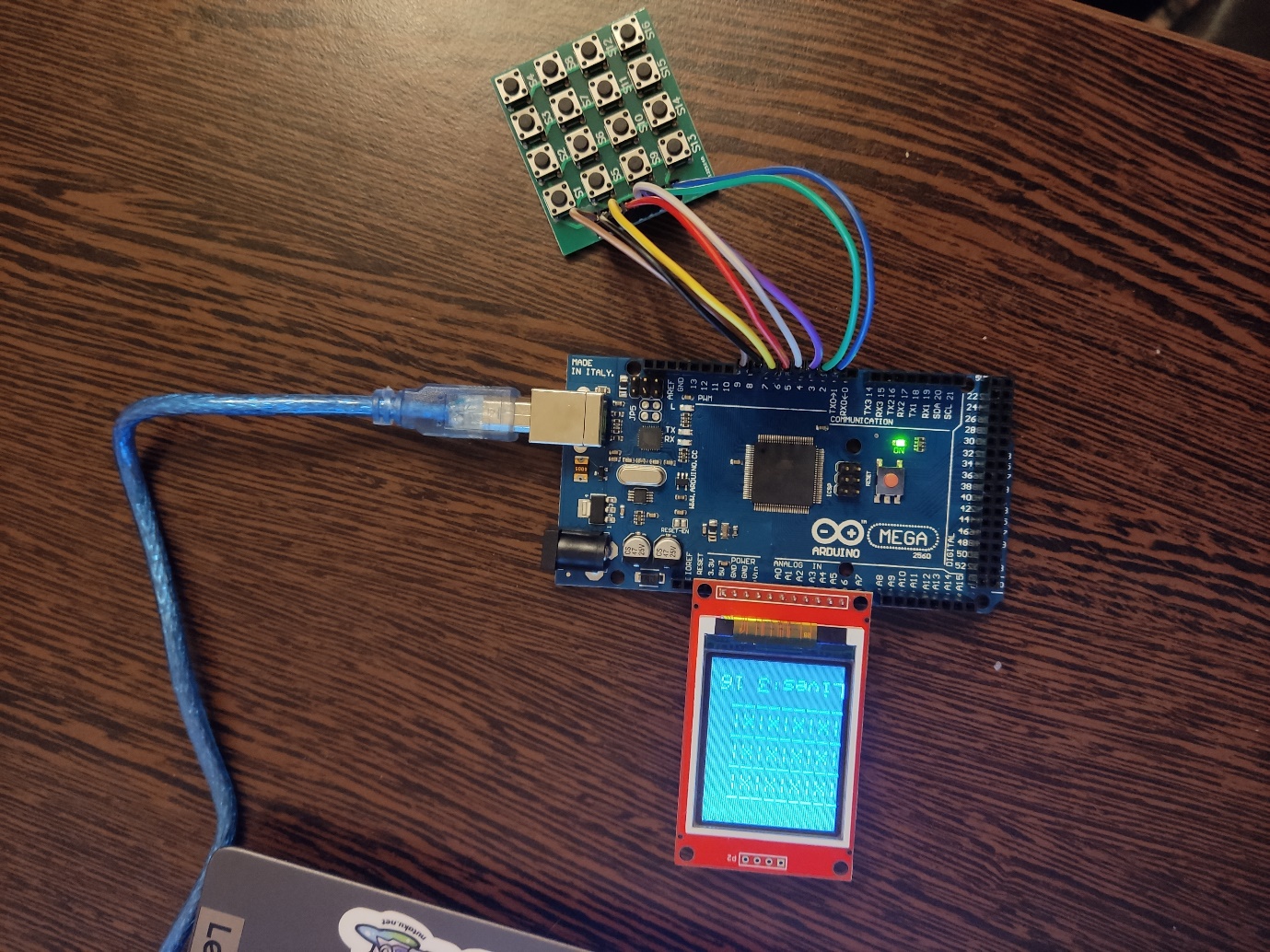
*}*

# Diagrama de circuit



# Poza proiectului





# Bibliografie

* <https://ardushop.ro/ro/home/865-modul-lcd-spi-128x160.html?search_query=Modul+LCD+SPI+128x160%09+&results=404>
* <https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/>
* <https://www.cplusplus.com/reference/cstdio/snprintf/>
* Cursul 3 de PMP
* <https://www.arduino.cc/reference/en/language/functions/time/millis/>
* https://techexplorations.com/guides/arduino/peripherals/arduino-timer1/