

**ПМГ „Академик Боян Петканчин“ –
Хасково**

Национална Програма „ИТ-КАРИЕРА“

Документация

**Модул 8: Въведение в операционни системи и
вградени системи**

Тема: Игра - Alien Invasion

**Изготвил:
Георги Добриков**

2020г.

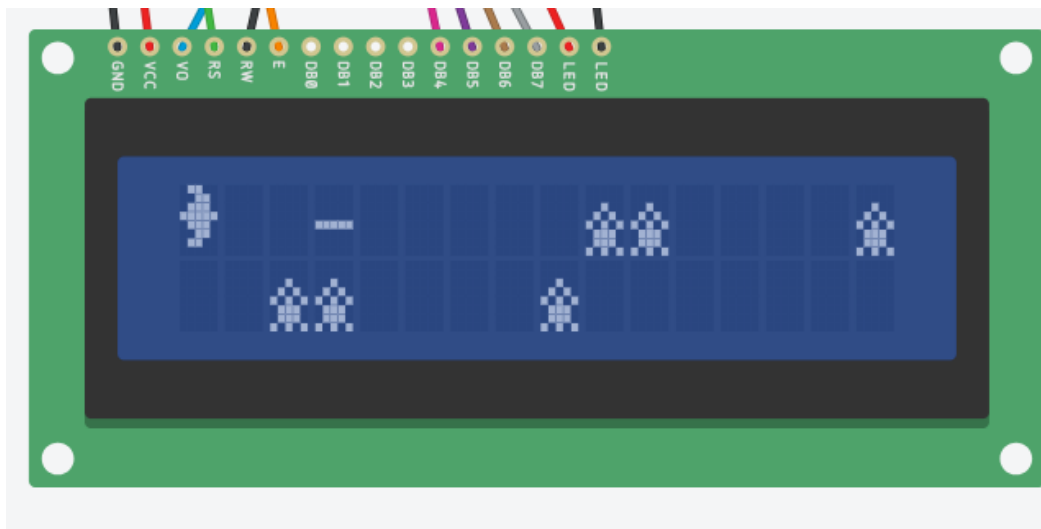
СЪДЪРЖАНИЕ:

- I. Въведение
- II. Списък от компоненти
- III. Електрическа схема
- IV. Блок схема
- V. Source code
- VI. Заключение

Въведение

- **Същност**

Проектът представлява игра, в която управляваш космически кораб в битка с извънземни. Космическият кораб може да се движи нагоре и надолу докато извънземните се придвижват към него. Корабът има 3 живота, като когато се удари в извънземно губи един живот. Играчът може да се отбранява с лазери като всяко уцелено извънземно носи една точка, но броя на лазерите е ограничен до 3. На края на играта се изписва крайният резултат и се дава възможност да се започне наново.



- **Функционалности**

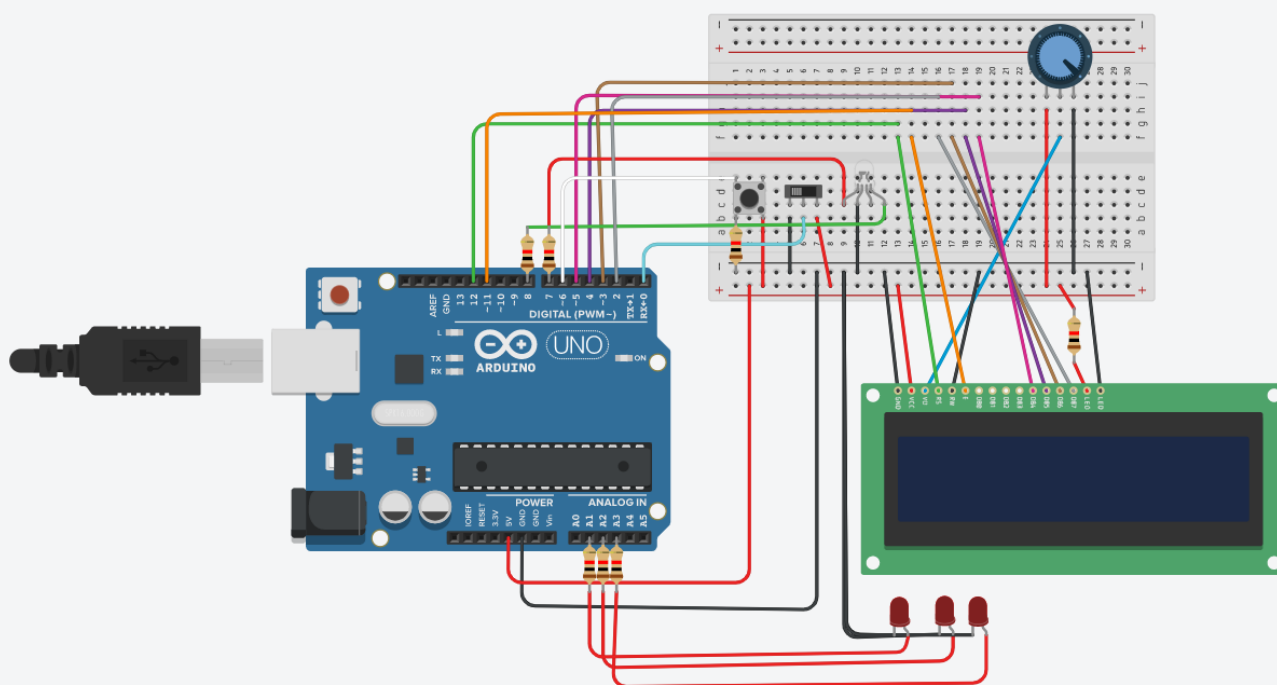
- LED лампи, които показват останалите жизни
- LED лампа показваща броя на изстреляните лазери
- Възможност за контрол над яркостта на екрана

Списък от съставни части

- 1.Arduino Uno**
- 2.Потенциометър X1**
- 3.LED лампи X3**
- 4.Резистори ($\sim 1k\Omega$) X7**
- 5.LCD дисплей 16x2**
- 6.LED RGB X1**
- 7.Бутон X1**
- 8.Ключ X1**

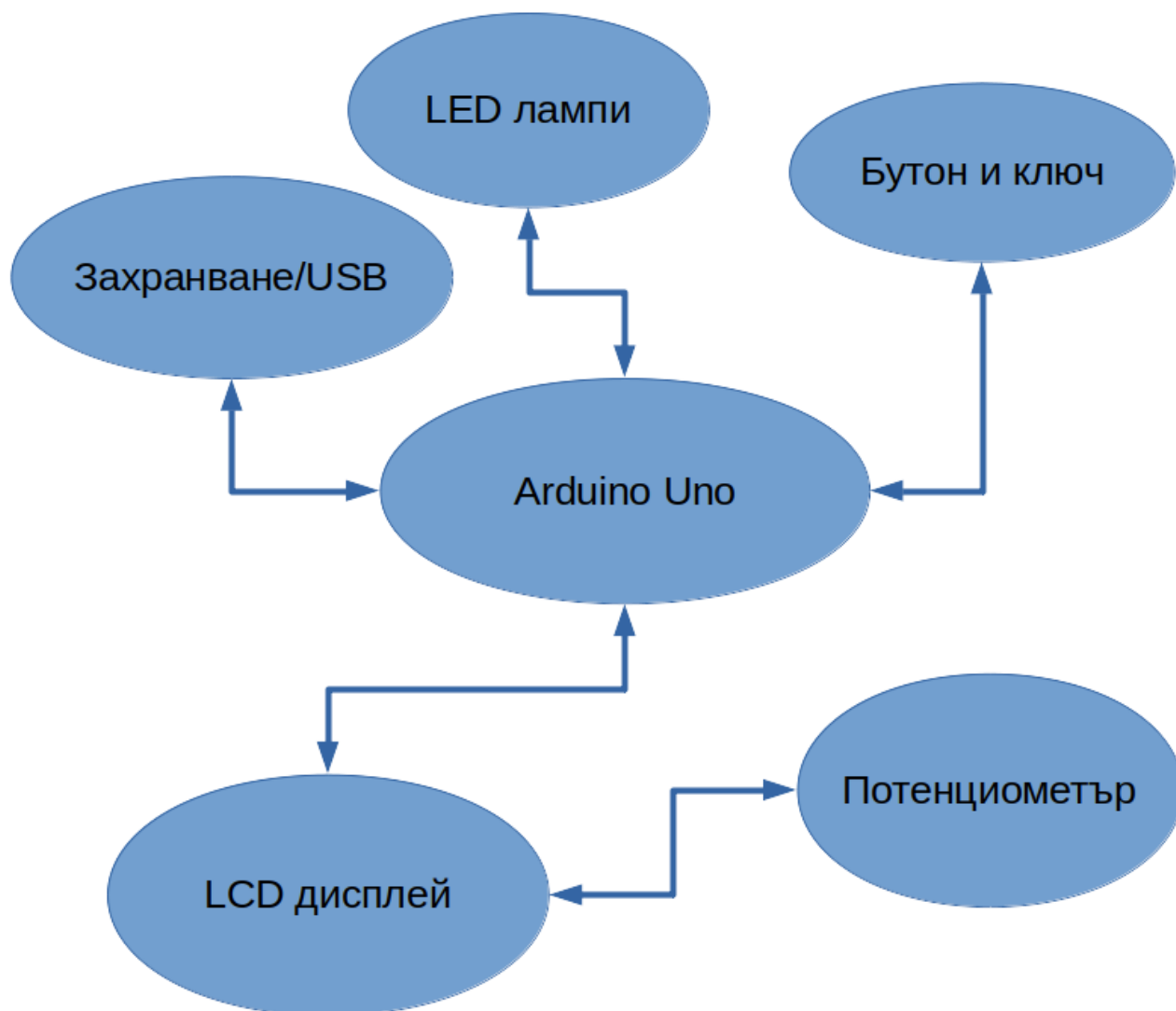
Електрическа схема

Цифрови входове 12, 11, 5, 4, 3, 2 на Arduino са аналогично свързани с изходи RS, E, DB4-7 на LCD дисплей. Потенциометърът е свързан с двата LED входа на LCD дисплей за контрол на яркостта. Аналогови входове от A1 до A3 на Arduino са свързана с LED лампи. Цифров вход 6 се свързва към бутон; цифров вход 0 е свързан към ключа. Цифрови изходи 8 и 7 са свързани към LED RGB лампата.



Блок схема

Цялото захранване в проекта се предоставя от Arduino платката. Тя приема информация от ключа и бутона, която предава на LCD дисплея и LED лампите.



Source code

```
#include <LiquidCrystal.h> // uses the liquid crystal library
//lcd wire setup
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
//character models
byte alien[8] = {
    0b00000,
    0b00000,
    0b00100,
    0b01010,
    0b10101,
    0b01110,
    0b01110,
    0b10101
};

byte spaceship1[8] = {
    0b00000,
    0b01100,
    0b00110,
    0b01110,
    0b11111,
    0b01110,
    0b00110,
    0b01100
};

byte spaceship2[8] = {
    0b01100,
    0b00110,
    0b01110,
    0b11111,
    0b01110,
    0b00110,
    0b01100,
    0b00000
};

byte laser[8] = {
    0b00000,
    0b00000,
    0b00000,
    0b00000,
    0b11111,
    0b00000,
    0b00000,
    0b00000
};

int laserCount = 0; //number of lasers on screen
int lasers[6]; //laser array
int direction = 0; //space ship orientation
int aliens[30]; //alien array
int alienSpawnrate = 2000; //the lower the number the faster the spawning
```

```

int health = 3;//hitpoints
int score = 0;//number of aliens killed
int gamestate = 0;//game on/game off
void setup()
{
    //character model setup
    lcd.createChar(0, spaceship1);
    lcd.createChar(1, spaceship2);
    lcd.createChar(2, alien);
    lcd.createChar(3, laser);
    lcd.begin(16, 2);// lcd grid setup
    pinMode(0, INPUT);//switch
    pinMode(6, INPUT);//button
    pinMode(A1, OUTPUT);//LED1
    pinMode(A2, OUTPUT);//LED2
    pinMode(A3, OUTPUT);//LED3
    pinMode(8, OUTPUT);//LED R
    pinMode(7, OUTPUT);//LED G
}

void loop() {
    if(!gamestate){//game not started
        lcd.print("Start game!");
        delay(500);
        lcd.clear();
        delay(500);
        if(digitalRead(6)){//resets variables
            gamestate = 1;
            health = 3;
            score = 0;
            laserCount = 0;
            alienSpawnrate = 2000;
            analogWrite(A1, 255);
            analogWrite(A2, 255);
            analogWrite(A3, 255);
            analogWrite(8, 200);
            analogWrite(7, 0);
            for(int i = 0; i < 30; i++)
                aliens[i] = 0;
            for(int i = 0; i < 6; i++)
                lasers[i] = 0;
        }
    }
    else{//started game
        if( direction != digitalRead(0)){//reads switch
            lcd.setCursor(0, direction);
            lcd.print(" ");
            direction = digitalRead(0);
        }

        SpaceshipBlink();

        if((millis()/alienSpawnrate)%2 == 0){//adjusts spawn rate depending on
time passed and score
            SpawnAliens(random(2));
            if((millis()/1000)%2 == 0)
                alienSpawnrate = alienSpawnrate - score*5;

```



```

        if(alienSpawnrate <= 0)
            alienSpawnrate = 2000;
    }
    LaserLogic();

    AlienMove();
}

}

void SpaceshipBlink() { //Space ship animation
    lcd.setCursor(0,direction);
    lcd.write((byte)0);
    delay(250);
    lcd.setCursor(0,direction);
    lcd.write((byte)1);
    delay(250);
}

void AlienMove() { //moves enemies forward

    for(int i = 0; i < 15; i++){
        if(alien[i] != 0) { //collision detector with lasers
            if(laserCount >= 1){
                for(int j = 0; j < 3; j++){
                    if(lasers[j] != 0 && (lasers[j] == alien[i] || lasers[j]+1 ==
alien[i]) && lasers[j+3] == alien[i+15]){
                        destroyLaser(j);
                        destroyAlien(i);
                        score++;
                        break;
                    }
                }
            }
            if(alien[i] != 0){
                for(int j = 0; j < 15; j++) { //fixes invisible enemy bug
                    if(alien[j] == alien[i] && j != i && alien[i+15] ==
alien[j+15]){
                        break;
                    }
                    else if(j == i){
                        lcd.setCursor(alien[i],alien[i+15]);
                        lcd.print(" ");
                        break;
                    }
                }
                alien[i] = alien[i]-1;
                lcd.setCursor(alien[i],alien[i+15]);
                lcd.write((byte)2);
                if(alien[i] == 0) { //collision detector with boundaries and space
ship
                    if(alien[i+15] == direction)
                        health--;
                    destroyAlien(i);
                }
            }
        }
    }
}

```

```

//changes LEDs depending on health
if(health == 2){
    analogWrite(A3,0);
}
else if(health == 1){
    analogWrite(A2,0);
}
else if(health == 0){
    analogWrite(A1,0);
    GameOver();
}
}

void SpawnAliens(int pos){//spawns enemies

lcd.setCursor(15,pos);
lcd.write((byte)2);
for(int i = 0;i < 15;i++){
    if(aliens[i] == 0){
        aliens[i] = 15;
        aliens[i+15] = pos;
        break;
    }
}
delay(200);
}

void LaserLogic(){
    //moves laser forward
    if(laserCount >= 1){
        for(int i = 0;i < 3;i++){
            if(lasers[i] != 0){
                for(int j =0; j < 3;j++){//fixes invisible laser bug
                    if(lasers[j] == lasers[i] && j!= i){
                        break;
                    }
                    else if(j == i){
                        lcd.setCursor(lasers[i],lasers[i+3]);
                        lcd.print(" ");
                        break;
                    }
                }
                lasers[i] = lasers[i]+1;
                lcd.setCursor(lasers[i],lasers[i+3]);
                lcd.write((byte)3);
                if(lasers[i] >= 16)//collision detector with boundaries
                    destroyLaser(i);
            }
        }
    }

    //creates laser
    if(digitalRead(6) == HIGH && laserCount < 3){
        lcd.setCursor(1,direction);
        lcd.write((byte)3);
        laserCount++;
        for(int i = 0;i < 3;i++){

```

```

        if(lasers[i] == 0){
            lasers[i] = 1;
            lasers[i+3] = direction;
            break;
        }
    }
}

//laser RGB controls
if(laserCount == 1 || laserCount == 0){
    analogWrite(8,200);
    analogWrite(7,0);
}
else if(laserCount == 2){
    analogWrite(7,255);
    analogWrite(8,160);
}
else if(laserCount == 3){
    analogWrite(7,255);
    analogWrite(8,0);
}
}

void destroyLaser(int id){//deletes laser and wipes it off screen
    laserCount--;
    lcd.setCursor(lasers[id],lasers[id+3]);
    lcd.print(" ");

    lasers[id] = 0;
}

void destroyAlien(int id){//deletes alien and wipes it off screen
    lcd.setCursor.aliens[id],aliens[id+15]);
    lcd.print(" ");
    aliens[id] = 0;
}

void GameOver(){//displays game over screen
    lcd.clear();
    lcd.print("Game over!");
    lcd.setCursor(0,1);
    lcd.print("Score:");
    lcd.print(score);
    gamestate = 0;
    delay(1500);
    lcd.clear();
}

```

Заключение

Проектът за играта се е получил добре и е добро решение за всеки със твърде много време. Има и още някои други функции, които ако бъдат добавени ще го направят много по-забавен:

- Добавяне на музика
- Създаване на алгоритъм за повишаване трудността на играта според прогреса

GitHUB: https://github.com/Bewtan/ItKariera_Module8
TinkerCad: <https://www.tinkercad.com/things/9gTRTEa8wLB-alien-invasion-game>