D&D Dice

Introducere

Autor: Pătrașcu Ioana-Alexandra

Grupa: 334CB

În cadrul proiectului mi-am propus să simulez digital o variație de zaruri care se folosesc în cadrul jocului Dungeons & Dragons. Se poate selecta tipul de zar dorit și se generează aleator un număr de pe zar, după ce se termină de „amestecat”.

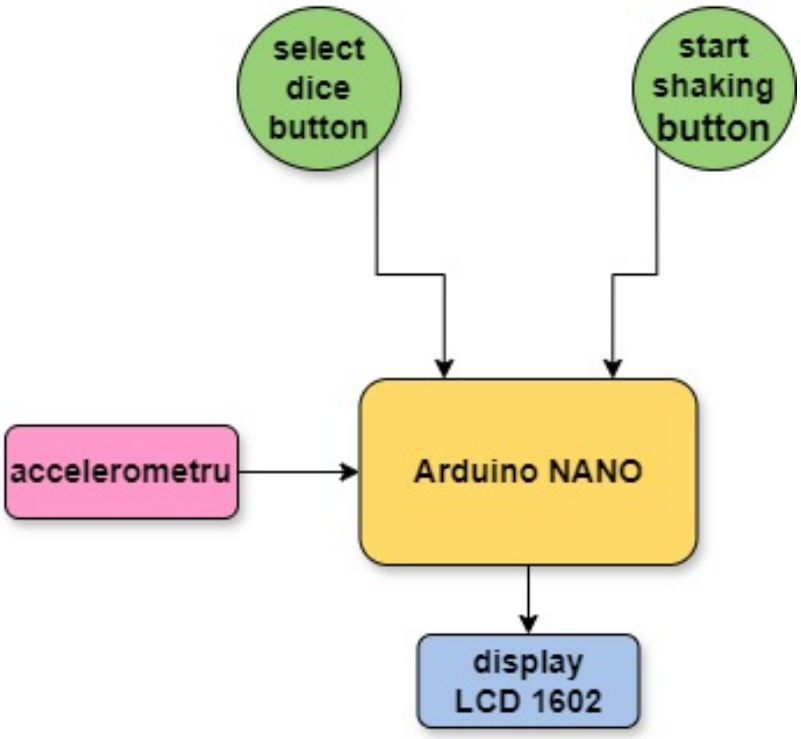
Această „amestecare” este simulată cu ajutorul unui accelerometru, care înregistrează momentul în care plăcuța nu mai este mișcată. Numărul generat random se afisează pe un ecran.

Acest tip de zar generalizat poate fi utilizat în diverse jocuri care folosesc zaruri. A fost gândit special pentru D&D, dar prin selectarea unui zar D6 poate fi folosit și în jocuri de table, de exemplu.

Descriere generală

Utilizatorul poate să își aleagă tipul de zar dorit dintr-un meniu care apare pe display la început. Pe primul rând al display-ului va apărea comanda ***select dice***, iar pe al doilea rând va fi tipul de zar. Opțiunile existente sunt cele clasice, folosite în D&D: D4, D6, D8, D10, D12, D20. Va fi implicit selectat un D6, iar prin apăsarea butonului se va schimba opțiunea.

După alegere, utilizatorul apasă butonul ***start shaking button*** și începe să miște plăcuța pentru amestecare. Se va detecta când amestecarea se termină (cu ajutorul accelerometrului) și va apărea pe display numărul generat.

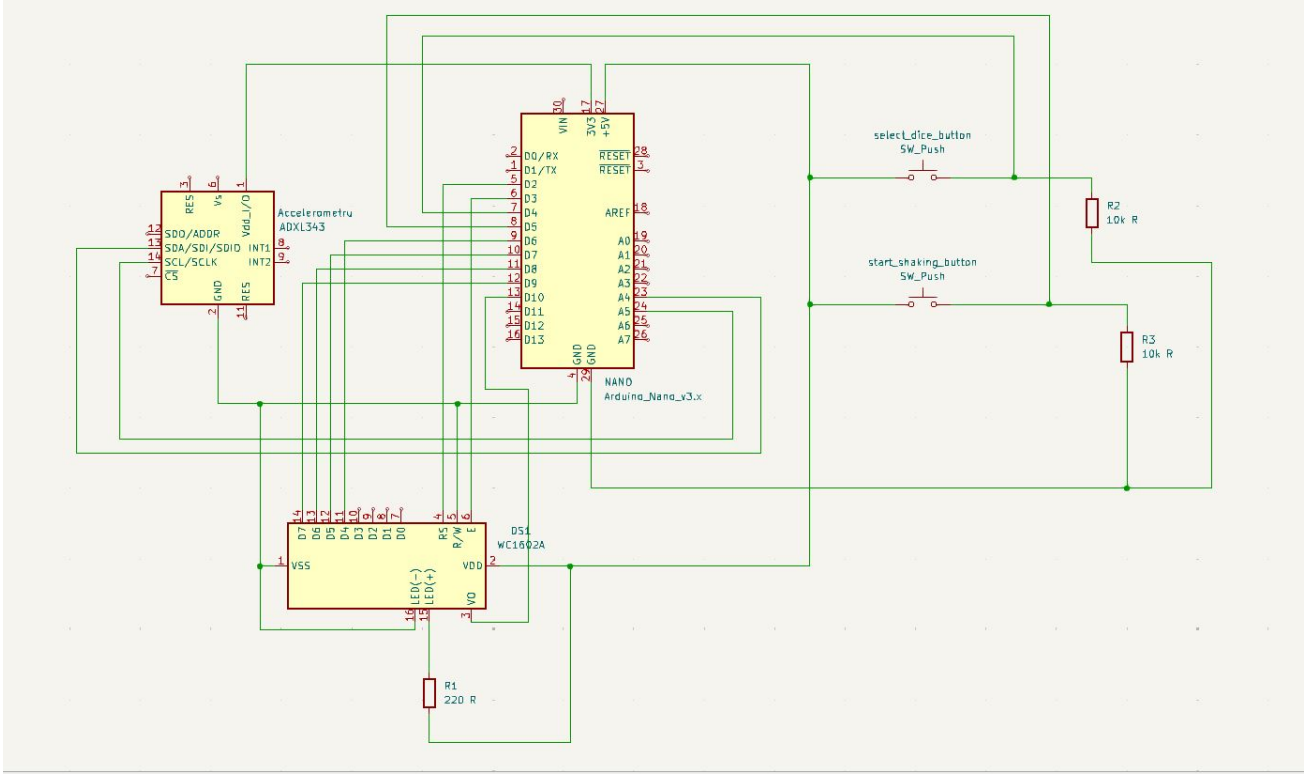


Hardware Design

Listă de piese:

- 1 x Arduino Nano
- 1 x Modul Accelerometru cu 3 axe ADXL345
- 2 x push button
- 1 x display LCD 1602
- 1 x rezistență 220Ω
- 2 * rezistență 10kΩ
- 1 x breadboard
- fire tată-tată și fire mamă-mamă

Schema electrică a proiectului:



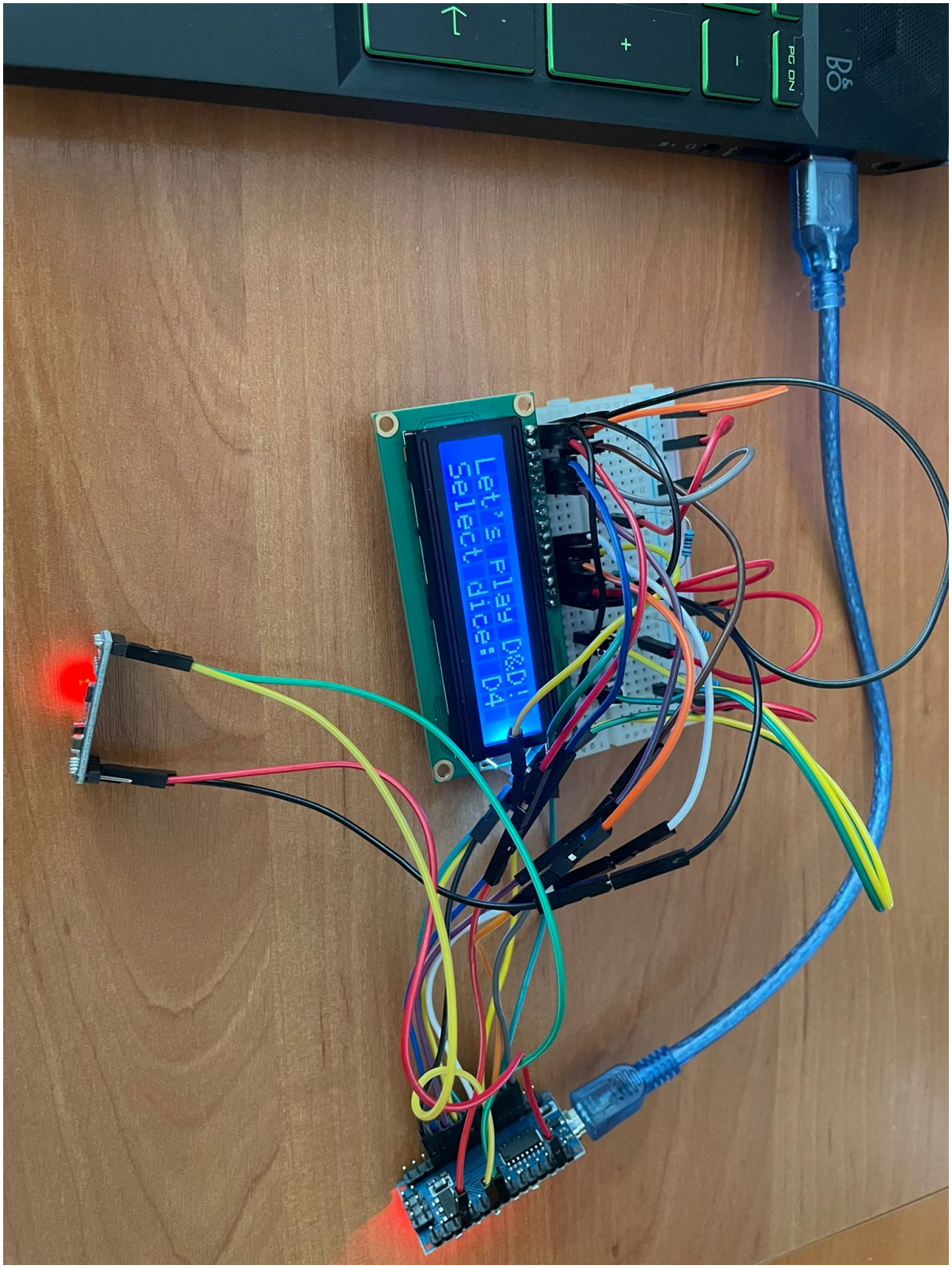
Conectare pini în schema electrică și pe cablaj

În realizarea schemei electrice am conectat mai întâi pinii dintre Andruino Nano și accelerometru, apoi dintre Andruino Nano și display (unde am avut nevoie și de o rezistență de 220Ω). Pentru acestea, am urmărit documentația componentelor sau tutoriale de conectare. Acestea sunt disponibile în secțiunea *Bibliografie/Resurse*. Butoanele au fost legate la final de Arduino Nano, folosind doi dintre pinii lăsați liberi și cele două rezistențe necesare de 10kΩ.

În realizarea cablajului am păstrat aproximativ aceeași ordine, dar am testat mai întâi display-ul prin conectarea la Arduino Nano, apoi am conectat accelerometrul și am scris un cod de testare pentru a verifica funcționalitatea. La final am conectat și butoanele, împreună cu rezistențele necesare.

În acest proces am ales pinii la fel ca în schema electrică.

Cablaj complet, în starea de start:



Software Design

Mediu de dezvoltare: Visual Studio Code + PlatformIO

Librării folosite:

- **Arduino** pentru Arduino Nano
- **LiquidCrystal** pentru a putea cotrola display-ul, folosind o plăcuță Arduino
- **Adafruit_Sensor** și **Adafruit_ADXL345_U** pentru accelerometru

```
#include <Arduino.h>
#include <LiquidCrystal.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
```

Configurări componente:

Pentru a folosi display-ul și accelerometrul, trebuie să le configurăm mai întâi (folosind I2C pentru amândouă). Acest lucru se face mai simplu folosind LiquidCrystal pentru display (**lcd**) și Adafruit_ADXL345_Unified() pentru accelerometru (**accelerometer**). Setăm și backlight-ul display-ului.

```
#define LCD_Backlight 10

// configurare LCD intern cu I2C, transferul de date este direct
const int rs = 2, en = 3, d4 = 6, d5 = 7, d6 = 8, d7 = 9;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

// configurare accelerometru
Adafruit_ADXL345_Unified accelerometer = Adafruit_ADXL345_Unified();
int ADXL345 = 0x53; // adresa I2C a senzorului ADXL345
```

Inițializăm alte variabile folosite:

- coordonatele pe care le înregistrează accelerometrul: X_out, Y_out, Z_out
- pinii folosiți pentru butoane, corespunzători pinilor de pe Arduino la care sunt legate

```
float X_out, Y_out, Z_out;
int selectPin = 4; // pinul butonului de selectare a zarului
int startShakePin = 5; // pinul butonului start_shaking
```

Setup:

```
void setup() {
  Serial.begin(9600);
  Serial.println("Hello!");
  delay(200);

  // mesaj de eroare dacă nu avem conexiune la acelerometru
  if (!accelerometer.begin(ADXL345)) {
    Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
    while (1);
  }
  accelerometer.setRange(ADXL345_RANGE_2_G);

  // setam numarul de linii si coloane ale LCD-ului
  pinMode(LCD_Backlight, OUTPUT);
  lcd.begin(16, 2);
```

```
  // setam contrast pentru textul de pe display
  analogWrite(LCD_Backlight, 128);

  // afisam un mesaj de inceput
  lcd.print("Let's play D&D!");

  // configuram pinii butoanelor
  pinMode(selectPin, INPUT);
  pinMode(startShakePin, INPUT);
  delay(2000);
}
```

Definire și inițializare variabile folosite:

```
float prev_x = 0, prev_y = 0, prev_z = 0;
int dices[6] = {4, 6, 8, 10, 12, 20};
int diceIndex = 0, dice = dices[diceIndex];
bool diceSelected = false;
bool startedShaking = false, shaking = false;
```

Implementare funcționalități:

```
void loop() {
  randomSeed(millis());

  // informatiile date de accelerometru sunt salvate in event
  sensors_event_t event;
  accelerometer.getEvent(&event);

  // setam cursorul de pe display la linia 0, coloana 1
  lcd.setCursor(0, 1);

  // determinam daca accelerometrul mai este agitat
  prev_x = X_out;
  prev_y = Y_out;
  prev_z = Z_out;
  X_out = event.acceleration.x; // valoarea inregistrata de accelerometru pe
axa X
  Y_out = event.acceleration.y; // valoarea inregistrata de accelerometru pe
axa Y
  Z_out = event.acceleration.z; // valoarea inregistrata de accelerometru pe
axa Z
  shaking = (abs(X_out - prev_x) >= 1 || abs(Y_out - prev_y) >= 1 || abs(
Z_out - prev_z) >= 1);

  // functionalitate propriu-zisa
  if(!diceSelected) {
    lcd.print("Select dice: D");
    lcd.print(dice);
    if (digitalRead(selectPin) == HIGH) {
```

```
      lcd.clear();
      lcd.setCursor(0, 1);
      lcd.print("Select dice: D");
      diceIndex++;
      if (diceIndex > 5) {
        diceIndex = 0;
      }
      dice = dices[diceIndex];
      lcd.print(dice);
      delay(500);
    }
    if (digitalRead(startShakePin) == HIGH) {
      diceSelected = true;
      lcd.clear();
      delay(500);
    }
  } else {
    lcd.setCursor(0, 1);
    if (!startedShaking) {
      lcd.print("Shake it!");
      startedShaking = shaking;
    } else {
      if (shaking) {
        lcd.print("Shaking...");
      } else {
        lcd.clear();
        lcd.print("Rolling...");
        delay(random(1000, 3000));
        lcd.clear();
        lcd.print("Dice shows: ");
        lcd.print(random(1, dice + 1));
        diceSelected = false;
        startedShaking = false;
        delay(2000);
      }
    }
    delay(100);
  }
}
```

Rezultate Obţinute

Am obținut o varietate de zaruri folosite în cadrul jocului de imaginație Dungeons & Dragons, în format digital. Sunt disponibile: D4, D6, D8, D10, D12, D20.

Proiectul simulează experiența de aruncat cu zarul și este util dacă nu ai toate aceste zaruri în format fizic, necesare în cadrul jocului Dungeons & Dragons. Însă, poate fi folosit pentru orice tip de joc care folosește zaruri. În general, D6, zarul cu 6 fețe, este cel mai folosit, iar acesta este simulat prin

selectarea tipului de zar dorit la D6.

Video Demo on YouTube

Concluzii

Până acum am mai lucrat la un astfel de proiect doar în cadrul laboratoarelor de la DEEA, ED sau PM, sub supraveghere și folosind componente oferite de facultate. A fost o experiență interesantă, care mi-a arătat câte erori și probleme pot să apară într-un proiect de acest tip și cât de ușor ar putea fi rezolvate dacă ai avea mai multă experiență în domeniu.

Download

Toate fișierele vizibile deja în această documentație se regăsesc în arhiva de mai jos.

d_d_dice.zip

Jurnal

3 mai

- ideea proiectului
- documentație inițială

10-12 mai

- analiza componentelor necesare pentru realizarea proiectului
- schema electrică
- achiziționarea componentelor

15-16 mai

- soldare pini și testare display LCD
- soldare pini și testare Arduino Nano

20-23 mai

- finalizare hardware design
- implementare software și rezolvare probleme
- testare proiect

24 mai

- documentație hardware
- documentație software
- demo

25-26 mai

- finalizare documentație: rezultate, concluzii și alte informații suplimentare

Probleme întâmpinate:

- Piesele comandate nu au venit soldate (Arduino Nano, accelerometrul și display-ul LCD). Am achiziționat suplimentar un kit de soldare și le-am soldat singură acasă.
- Accelerometrul a venit de 2 ori dezmembrat. A treia bucată a fost în regulă.

Bibliografie/Resurse

Resurse Software:

- Arduino Display Integration Video Tutorial
- Arduino Accelerometer using the ADXL345

Resurse Hardware:

- How To Track Orientation with Arduino and ADXL345 Accelerometer
- Arduino Display Integration Video Tutorial
- How to Connect and Calibrate the ADXL345 with Arduino Video Tutorial

Export to PDF