Electronic Keyboard

Autor: Enculesei Vlad-Razvan

Introducere

Acest proiect imita comportamentul unui pian electric cu 8 clape reprezentand 8 note din gama DO Major. In momentul in care unul din cele 8 butoane este apasat, se va declansa buzzerul care genereaza sunetul corespunzator butonului apasat, pe display-ul cu 7 segmente se va afisa cifra corespunzatoare de la 1 la 8, iar pe LCD va fi afisata nota muzicala atat in notatia DO_RE_MI, cat si in notatia A_B_C.

Descriere generală

Aceasta este schema bloc cu toate modulele din proiect:

Hardware Design

Aceasta este lista de piese din proiect:

- Arduino UNO
- Breadboard
- LCD cu I2C
- 7-Segment Display cu catod comun
- Buzzer pasiv
- 8 Butoane
- 10 Rezistente de 1 kΩ
- Fire

Mai jos este schema electrica a proiectului realizata in Tinkercad, cu mentiunea ca LCD-ul cu I2C nu putea fi simulat in Tinkercad deoarece nu exista componenta respectiva. In implementarea fizica LCD-ul cu I2C este conectat la GND, VCC, iar iesirile SDA si SCL sunt conectate la A4, respectiv A5.

Software Design

Mediul de dezvoltare a fost Tinkercad, iar mai apoi Arduino IDE, mai ales pentru ca nu exista LCD cu I2C in Tinkercad, asa cum s-a mai mentionat si mai sus.

Pentru LCD-ul cu I2C a fost instalata biblioteca LiquidCrystal I2C by Marco Schwartz version 1.1.2 si a fost initializata o variabila LiquidCrystal_I2C cu adresa 0X27 LCD-ului si 16 caractere pe cate 2 linii. Functia setup seteaza starea initiala a butoanelor (LOW), seteaza corespunzator pinii si seteaza si LCD-ul.

Functia resetDisplay reseteaza display-ul pe 7 segmente si LCD-ul in asa fel incat sa nu se afiseze nimic. Functiile p*, unde * apartine multimii de note muzicale, seteaza LCD-ul si display-ul cu 7 segmente pentru a se afisa o nota muzicala in doua moduri si respectiv o cifra de la 1 la 8. Functia debounce, dupa cum ii spune si numele, face debouncing pentru butoanele pianului.

Finctiile check*, unde * apartine multimii de note muzicale, verifica daca s-a apasat butonul corespunzator notei muzicale respective, activeaza buzzer-ul si apoi apeleaza p* pentru afisaj. In functia loop, se apeleaza functiile check* si mai apoi resetDisplay.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define piezoPin A2
#define DO 2
#define RE 3
#define MI 4
#define FA 5
#define SOL 6
#define LA 7
#define SI 8
#define DO2 13
#define A 12
#define B 9
#define C A3
#define D A0
#define E A1
#define F1 10
#define G 11

int buttonState;
LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x27, 16, 2);

void setup() {
  Serial.begin(9600);
  lcd.init();
  lcd.clear();
  lcd.backlight();
  buttonState = LOW;
  pinMode(DO, INPUT);
  pinMode(RE, INPUT);
```

```
  pinMode(MI, INPUT);
  pinMode(FA, INPUT);
  pinMode(SOL, INPUT);
  pinMode(LA, INPUT);
  pinMode(SI, INPUT);
  pinMode(DO2, INPUT);
  pinMode(A, OUTPUT);
  pinMode(B, OUTPUT);
  pinMode(C, OUTPUT);
  pinMode(D, OUTPUT);
  pinMode(E, OUTPUT);
  pinMode(F1, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(piezoPin, OUTPUT);
}

void resetDisplay(void) {
  lcd.clear();
  lcd.noCursor();
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F1, LOW);
  digitalWrite(G, LOW);
}

void pDO(void) {
  lcd.setCursor(7, 0);
  lcd.print("DO");
  lcd.setCursor(7, 1);
  lcd.print("C");
  digitalWrite(A, LOW);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F1, LOW);
  digitalWrite(G, LOW);
}

void pRE(void) {
  lcd.setCursor(7, 0);
  lcd.print("RE");
  lcd.setCursor(7, 1);
  lcd.print("D");
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, LOW);
  digitalWrite(D, HIGH);
```

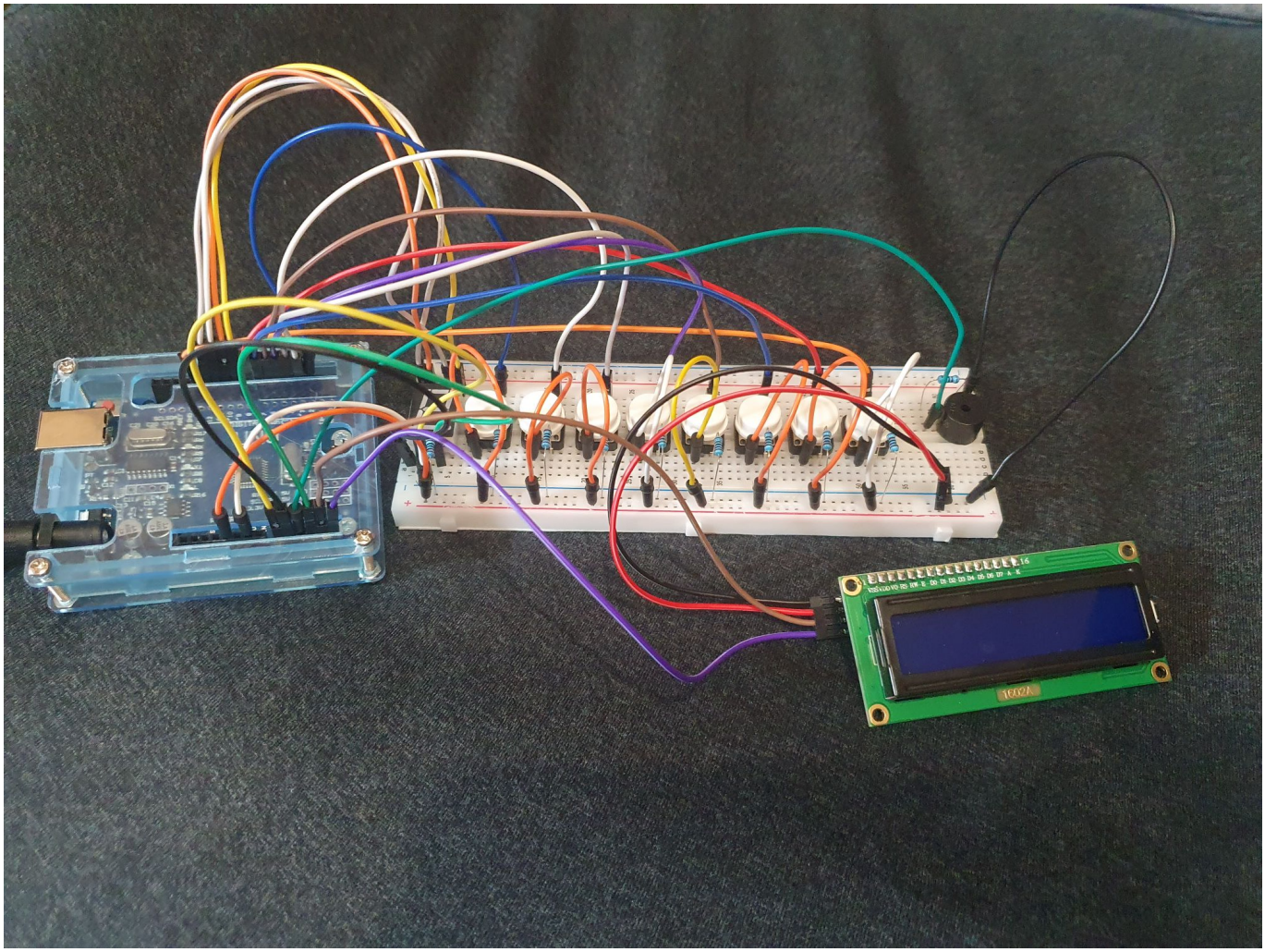
```
  digitalWrite(E, HIGH);
  digitalWrite(F1, LOW);
  digitalWrite(G, HIGH);
}

void pMI(void) {
  lcd.setCursor(7, 0);
  lcd.print("MI");
  lcd.setCursor(7, 1);
  lcd.print("E");
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, LOW);
  digitalWrite(F1, LOW);
  digitalWrite(G, HIGH);
}

void pFA(void) {
  lcd.setCursor(7, 0);
  lcd.print("FA");
  lcd.setCursor(7, 1);
  lcd.print("F");
  digitalWrite(A, LOW);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F1, HIGH);
  digitalWrite(G, HIGH);
}

void pSOL(void) {
  lcd.setCursor(7, 0);
  lcd.print("SOL");
  lcd.setCursor(7, 1);
  lcd.print("G");
  digitalWrite(A, HIGH);
  digitalWrite(B, LOW);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, LOW);
  digitalWrite(F1, HIGH);
  digitalWrite(G, HIGH);
}

void pLA(void) {
  lcd.setCursor(7, 0);
  lcd.print("LA");
  lcd.setCursor(7, 1);
```

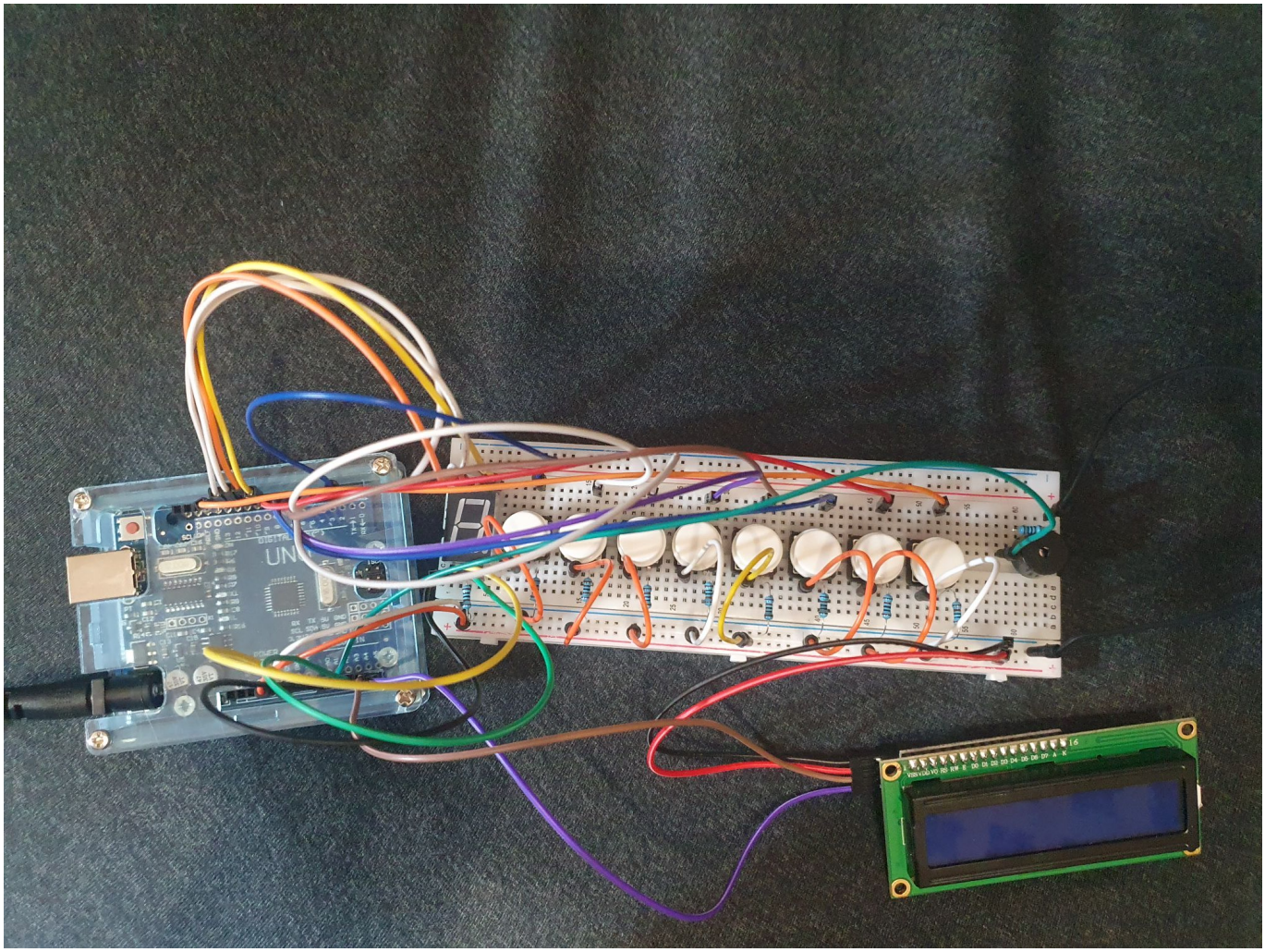
```
  lcd.print("A");
  digitalWrite(A, HIGH);
  digitalWrite(B, LOW);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, HIGH);
  digitalWrite(F1, HIGH);
  digitalWrite(G, HIGH);
}

void pSI(void) {
  lcd.setCursor(7, 0);
  lcd.print("SI");
  lcd.setCursor(7, 1);
  lcd.print("B");
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F1, LOW);
  digitalWrite(G, LOW);
}

void pDO2(void) {
  lcd.setCursor(7, 0);
  lcd.print("DO");
  lcd.setCursor(7, 1);
  lcd.print("C");
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, HIGH);
  digitalWrite(F1, HIGH);
  digitalWrite(G, HIGH);
}

int debounce(int buttonPin) {
  int stateNow = digitalRead(buttonPin);
  if(buttonState != stateNow) {
    delay(10);
    stateNow = digitalRead(buttonPin);
  }
  return stateNow;
}

void checkDO(void) {
  while(debounce(DO) == HIGH) {
    tone(piezoPin, 261.63, 200);
    pDO();
```

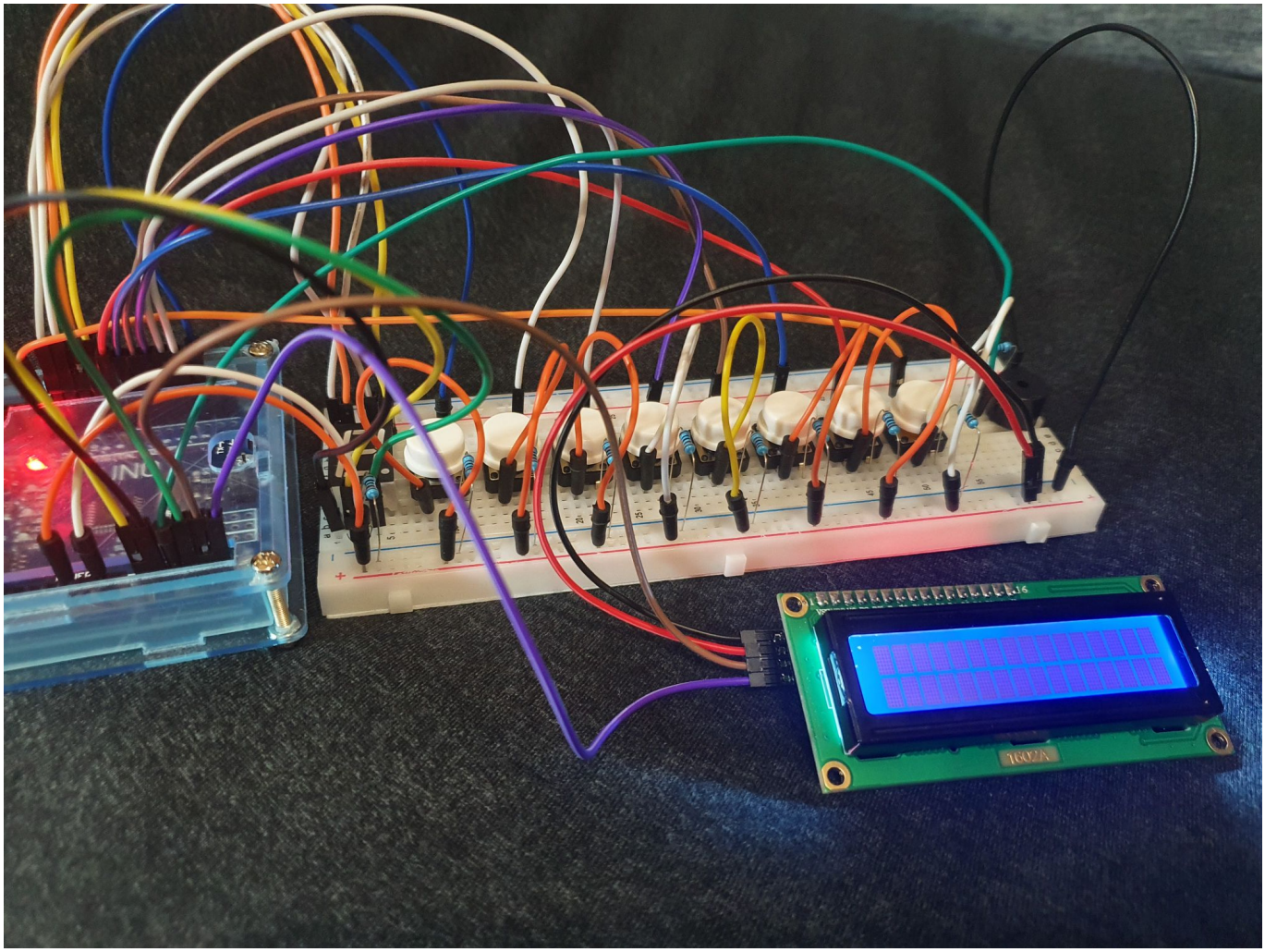
```
    delay(200);
  }
}

void checkRE(void) {
  while(debounce(RE) == HIGH) {
    tone(piezoPin, 293.665, 200);
    pRE();
    delay(200);
  }
}

void checkMI(void) {
  while(debounce(MI) == HIGH) {
    tone(piezoPin, 329.628, 200);
    pMI();
    delay(200);
  }
}

void checkFA(void) {
  while(debounce(FA) == HIGH) {
    tone(piezoPin, 349.228, 200);
    pFA();
    delay(200);
  }
}

void checkSOL(void) {
  while(debounce(SOL) == HIGH) {
    tone(piezoPin, 391.995, 200);
    pSOL();
    delay(200);
  }
}

void checkLA(void) {
  while(debounce(LA) == HIGH) {
    tone(piezoPin, 440, 200);
    pLA();
    delay(200);
  }
}

void checkSI(void) {
  while(debounce(SI) == HIGH) {
    tone(piezoPin, 493.883, 200);
    pSI();
    delay(200);
  }
}
```

```
void checkDO2(void) {
  while(debounce(DO2) == HIGH) {
    tone(piezoPin, 523.251, 200);
    pDO2();
    delay(200);
  }
}

void loop() {
  checkDO();
  checkRE();
  checkMI();
  checkFA();
  checkSOL();
  checkLA();
  checkSI();
  checkDO2();
  resetDisplay();
}
```

Rezultate Obţinute







Concluzii

Proiectul acesta poate fi folositor ca jucarie pentru copiii foarte mici care doresc sa se joace cu un pian si sa si invete, in acelasi timp, notele muzicale. Mie mi-a placut proiectul pentru ca am invatat cum sa folosesc un LCD cu I2C si un buzzer pasiv. In plus, am invatat sa fac debouncing pentru butoanele pianului.

Download

Arhiva proiectului (cod + README)

Bibliografie/Resurse

- Display pe 7 segmente (catod comun vs anod comun)
- Debounce butoane
- LCD cu I2C

- Note muzicale (de la C4 la C5)
- Buzzer pasiv

Export to PDF