MP3 player

Introducere

Cu pasiunea pentru tehnologie și dorința de a crea o experiență muzicală captivantă, am dezvoltat o simulare meticulos elaborată a unui MP3 player utilizând placa Arduino UNO. Acest dispozitiv inteligent integrează o gamă de componente sofisticate pentru a aduce muzica la viață într-un mod inovator și practic.

Cu ajutorul unui ecran LCD I2C, interfața este elegantă și informativă, oferind utilizatorului informații esențiale despre melodia care rulează în prezent, stadiul de redare/pauză și nivelul volumului. Astfel, fiecare melodie devine o experiență vizuală și auditivă de neuitat.

Modulul SD card servește ca un depozit eficient pentru playlist-uri și melodii, oferind astfel o gamă vastă de opțiuni muzicale la îndemână. Cu ajutorul unui potențiometru, utilizatorul poate ajusta volumul melodiilor pentru a obține exact nivelul dorit de imersiune și calitate sonoră.

Controlul melodiilor este intuitiv și ușor de utilizat, datorită celor patru butoane dedicate: “previous” pentru melodia anterioară, “pause” pentru a întrerupe temporar redarea, “start” pentru a relua redarea și “next” pentru a trece la următoarea melodie. Această interfață simplă și ergonomică adaugă un plus de confort și eficiență în experiența de auditie.

Prin această implementare minuțioasă și profesională, am creat nu doar un simplu MP3 player, ci o piesă de artă tehnologică care îmbină perfect utilitatea și inovația într-un dispozitiv compact și elegant. Este o demonstrație a abilităților noastre tehnice și a dedicării noastre pentru a aduce inovația în lumea muzicii digitale.

Descriere generală

- Modul MP3 player DFPlayer Mini: Un modul compact și versatil care permite redarea fișierelor MP3 de pe un card microSD. Este dotat cu funcții de control ușor de integrat și oferă o soluție eficientă pentru redarea muzicii digitale.
- Ecran LCD I2C: Un ecran LCD cu interfață I2C, care permite afișarea informațiilor într-un mod clar și ușor de citit. Interfața I2C reduce cablajul și face integrarea mai simplă.
- 4 butoane: Butonul reprezintă o interfață fizică pentru controlul funcțiilor MP3 player-ului, cum ar fi play/pause, next și previous.
- Potențiometru linear 10K: Folosit pentru reglarea volumului melodiilor, potențiometrul oferă o modalitate analogică și precisă de ajustare a nivelului de sunet.

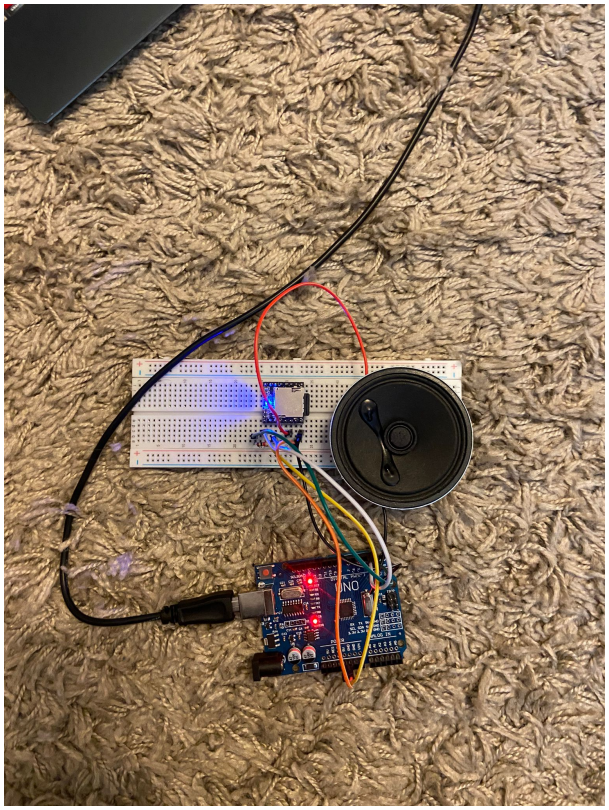
- Modul SD card: Folosit pentru stocarea fișierelor audio, modulul SD card permite accesul rapid și eficient la o colecție extinsă de melodii, organizate în playlist-uri.
- Placa de dezvoltare UNO R3 Arduino: Placa centrală a proiectului, care găzduiește toate componentele și coordonează funcționarea MP3 player-ului.
- Breadboard 830 puncte MB-102: Folosit pentru prototipare și testare, breadboard-ul oferă o platformă flexibilă pentru conectarea și interconectarea componentelor fără a fi nevoie de sudură.
- Fire și rezistoare: Utilizate pentru a conecta și interconecta componentele, firele și rezistoarele sunt elemente esențiale în construirea circuitului.
- Speaker: Folosit pentru redarea sunetului, difuzorul transformă semnalul digital în sunet auditiv, oferind o experiență audio captivantă.

Hardware Design

LISTA DE PIESE:

- Modul MP3 player DFPlayer Mini
- LCD screen l2C
- 4 butoane
- Potentioemetru linear 10K
- SD card
- Placa de dezvoltare UNO R3 Arduino
- Breadboard 830 puncte MB-102
- Fire
- 1k resistor
- Speaker
- baterie 9V

In prima faza am construit ceva de testare, din cauza ca am avut si o problema cu LCD-ul si imi doream mult sa fac ceva cat mai rapid si functional asa ca am facut un simplu AudioPlayer, pana am ajuns la forma finala:



Software Design

Prezentarea stadiului actual al implementării software: Codul actual implementează un sistem de control pentru un modul DFPlayer Mini prin intermediul unui Arduino. Acesta include:

- Inițializarea și configurarea unui display LCD I2C pentru afișarea informațiilor despre piesa curentă și volum.
- Configurarea și utilizarea unui SoftwareSerial pentru comunicarea cu DFPlayer Mini.
- Configurarea întreruperilor externe și a întreruperilor pe schimbarea pinilor pentru a detecta apăsările butoanelor de control (Previous, Pause, Start, Next).
- Citirea și afișarea volumului pe baza unui potențiometru conectat la pinul analogic A0.

Motivarea alegerii bibliotecilor folosite in cadrul proiectului:

- SoftwareSerial: Aceasta biblioteca este utilizata pentru a crea o a doua conexiune seriala software pe pini digitali (RX si TX) pentru a comunica cu DFPlayer Mini. Este esentiala deoarece majoritatea placilor Arduino au doar un port serial hardware, iar acest port este folosit de obicei pentru comunicarea cu computerul.
- DFRobotDFPlayerMini: Aceasta biblioteca ofera functii specifice pentru controlul DFPlayer Mini, facilitand comenzile pentru redarea, pauzarea si schimbarea pieselor, precum si ajustarea volumului. Este special conceputa pentru acest modul audio, simplificand integrarea si utilizarea acestuia.
- LiquidCrystal_I2C: Biblioteca pentru controlul unui display LCD cu interfata I2C. Este folosita pentru a afisa informatiile necesare utilizatorului intr-un mod usor de citit, reducand numarul de pini necesari pentru conectarea unui display LCD standard.

Explicatii legate de scheletul proiectului si interactiunea dintre functionalitati:

Scheletul proiectului: 1.Setup:

- Configurarea comunicarii seriale si initializarea componentelor hardware (LCD, DFPlayer Mini, pini de butoane).
- Setarea intreruperilor externe pentru butoanele Previous si Pause.
- Setarea intreruperilor pe schimbarea pinilor pentru butoanele Start si Next.

2.Loop:

- Verificarea starii variabilelor de intrerupere pentru a determina daca un buton a fost apasat si executarea actiunilor corespunzatoare (schimbarea piesei, pauzare, redare).
- Citirea valorii potentiometrului pentru controlul volumului si actualizarea acestei valori pe display-ul LCD.
- Afisarea numarului piesei curente si a volumului pe display-ul LCD.

Interactiunea dintre functionalitati:

- Butoane de control: Apasarile butoanelor sunt detectate prin intermediul intreruperilor, care seteaza variabilele de stare corespunzatoare. Aceste variabile sunt verificate in bucla principala si declanseaza actiunile corespunzatoare pe modulul DFPlayer Mini.
- Controlul volumului: Potentiometrul conectat la pinul A0 este citit periodic pentru a ajusta volumul modulului DFPlayer Mini. Valoarea volumului este afisata pe display-ul LCD.
- Afisarea pe LCD: Display-ul LCD este utilizat pentru a oferi feedback vizual utilizatorului, afisand numarul piesei curente si volumul.

Codul final:

#include <SoftwareSerial.h> #include <DFRobotDFPlayerMini.h> #include <LiquidCrystal_I2C.h>

#define TX 2 #define RX 3 SoftwareSerial mySoftwareSerial(RX, TX);

DFRobotDFPlayerMini myMp3;

LiquidCrystal_I2C myLcd(0x27, 16, 2);

const int previousButtonPin = 2; const int pauseButtonPin = 3; const int startButtonPin = 4; const int nextButtonPin = 5;

volatile bool previousButtonPressed = false; volatile bool pauseButtonPressed = false; volatile bool startButtonPressed = false; volatile bool nextButtonPressed = false;

void setup() {

```
Serial.begin(9600);

mySoftwareSerial.begin(9600);
```

```
myLcd.init();
myLcd.backlight();
```

```
pinMode(previousButtonPin, INPUT_PULLUP);
```

```
pinMode(pauseButtonPin, INPUT_PULLUP);
pinMode(startButtonPin, INPUT_PULLUP);
pinMode(nextButtonPin, INPUT_PULLUP);
```

```
EICRA |= (1 << ISC01) | (1 << ISC00);
EICRA |= (1 << ISC11) | (1 << ISC10);
EIMSK |= (1 << INT0);
EIMSK |= (1 << INT1);
```

```
PCICR |= (1 << PCIE2);
PCMSK2 |= (1 << PCINT20);
PCMSK2 |= (1 << PCINT21);
```

```
myLcd.setCursor(0, 0);
myLcd.print("TRACK");
myLcd.setCursor(10, 0);
myLcd.print("PLAY ");
myLcd.setCursor(0, 1);
myLcd.print("VOLUME");


if (myMp3.begin(mySoftwareSerial)) {

  myMp3.play();
  myMp3.enableLoopAll();
} else {
  Serial.println("Connecting to DFPlayer Mini failed!");
}
sei();
```

}

void loop() {

```
if (previousButtonPressed) {
  previousButtonPressed = false;
  myMp3.previous();
  TrackNumberClear();
}
if (pauseButtonPressed) {
  pauseButtonPressed = false;
  myMp3.pause();
  myLcd.setCursor(10, 0);
  myLcd.print("PAUSE");
}
if (startButtonPressed) {
  startButtonPressed = false;
  myMp3.start();
  myLcd.setCursor(10, 0);
  myLcd.print("PLAY ");
}
```

```
if (nextButtonPressed) {
  nextButtonPressed = false;
  myMp3.next();
  TrackNumberClear();
}
```

myMp3.volume(map(analogRead(A0), 0, 1023, 0, 30));

```
myLcd.setCursor(6, 0);
myLcd.print(myMp3.readCurrentFileNumber());

myLcd.setCursor(7, 1);
if (myMp3.readVolume() < 10)
  myLcd.print(String("0") + myMp3.readVolume());
else
  myLcd.print(myMp3.readVolume());
```

}

void TrackNumberClear() {

```
delay(500);
myLcd.setCursor(6, 0);
myLcd.print("   ");
```

}

ISR(INT0_vect) {

```
previousButtonPressed = true;
```

}

ISR(INT1_vect) {

```
pauseButtonPressed = true;
```

}

ISR(PCINT2_vect) {

```
if (digitalRead(startButtonPin) == LOW) {
  startButtonPressed = true;
}
if (digitalRead(nextButtonPin) == LOW) {
  nextButtonPressed = true;
}
```

}

Jurnal

De-a lungul proiectului am intampinat niste momente mai ghinioniste in legatura cu partea de hardware, cum ar fi am cumparat un LCD gresit si a trebuit sa recumpar unul nou, pentru ca aveam nevoie de I2C, am avut probleme cu placa arduino, nu facea bine contact cu firele tata-tata si a trebuit sa fac rost de una noua, apoi mi s-a ars DFPlayerul din cauza ca nu am pus rezistenta intr-un mod corect, insa am reusit sa invat din aceste greseli si sa fiu mai atenta pe viitor. La partea de Soft n-as putea spune ca am intampinat prea multe probleme pentru ca m-au ajutat destul de mult cunostintele de la laborator. In concluzie mi-a placut tare mult sa implementez un astfel de proiect de la 0, faptul ca am reusit sa fac toate astea cu mana mea si sa nu fie strict un cod scris, sa vad si ceva materializat care sa functioneze a fost foarte enjoyable pentru mine.

Export to PDF