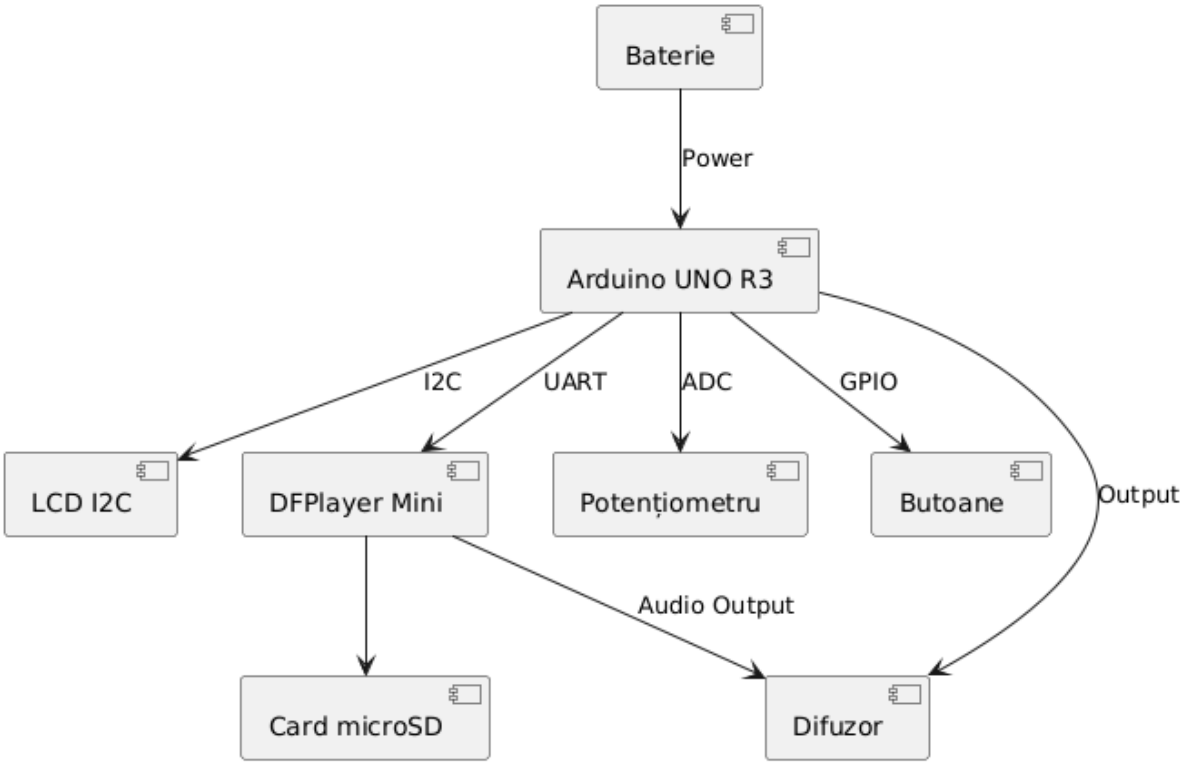
Player mp3

Ciobanu Alexandra 333CD

Introducere

Proiectul presupune crearea unui **MP3 Player** portabil care îi permite utilizatorului să încarce muzică pe un card SD, iar titlul melodiei va fi afișat pe un ecran LCD. Utilizatorul poate schimba piesele prin 4 butoane (next/back, pause/play), iar volumul poate fi controlat cu un potențiometru.

Descriere generală



Music playerul construit cu Arduino UNO permite redarea fisierelor audio de pe un card microSD folosind DFPlayer mini. Intreg sistemul este alimentat la o baterie 9v, iar interactiunea utilizatorului cu dispozitivul se face prin 4 butoane(next, prev, pause, play) conectate la pinii digitali ai placii, un potentiometru de 10k pentru reglarea volumului citit prin ADC si un LCD display cu interafata I2C, pe care se afiseaza titlul piesei. La pornirea sistemului, Arduino configureaza I2C si UART, stabileste conexiunile cu LCD, modulul DFPlayer si citeste valoarea analogica de la potentiometru care este convertita intr-o unitate de volum. Simultan, Arduino selecteaza o melodie de pe cardul microSD si afiseaza titlul pe display.

Hardware Design

Lista piese:

- Arduino UNO
- LCD Display I2C
- Potentiometru 10k
- Breadboard
- MicroSD Card 2GB
- DFPlayer mini
- Speaker
- Rezistenta 1k
- 4 butoane fara hold

Inițial, am vrut ca funcția de play/pause să fie controlată de un comutator cu funcție de hold, însă am renunțat la idee deoarece butonul era prea mare pentru a fi montat pe breadboard și nu permitea o integrare ușoară în carcasă. În locul lui, am folosit două butoane simple fără funcție de hold.

Pentru redarea fișierelor MP3, am optat pentru modulul DFPlayer Mini în locul unui cititor SD clasic, în principal pentru că DFPlayer-ul era disponibil în stoc și, fiind plat, se potrivea mai bine în ansamblul proiectului, mai ales în ceea ce privește ascunderea părții electronice într-o carcasă. Alimentarea s-a făcut astfel: VCC la 5V pe Arduino Uno, GND la GND, TX și RX la pinii 10 și 11, iar pe linia RX am adăugat o rezistență de 1k. La început am crezut că modulul este defect, deoarece nu reușea să citească fișierele MP3, dar am descoperit că problema era, de fapt, cardul SD, care fusese formatat greșit.

Ulterior, am înlocuit difuzorul din imaginea inițială cu unul mai puternic, care oferea un sunet mai clar. Acesta a fost conectat la ieșirile SPK1 și SPK2 ale DFPlayer-ului.

LCD-ul I2C a fost conectat astfel: VCC la 5V, GND la GND, SDA la pinul A4 și SCL la pinul A5. Pentru controlul volumului am folosit un potențiometru, ales datorită simplității și ușurinței în utilizare. Pinul + al potențiometrului a fost conectat la VCC, pinul - la GND, iar cursorul (pinul central) la pinul A0 de pe Arduino.

Software Design

Proiectul fost realizat in Arduino IDE.

Acest program este destinat unui sistem audio controlat de Arduino UNO, care permite redarea de fișiere audio, afișarea informațiilor pe un ecran LCD și controlul interacțiunii printr-un potențiometru și câteva butoane fizice.

Redarea melodiilor se face cu ajutorul modulului DFPlayer Mini, un player MP3 autonom ce poate reda fișiere direct de pe un card microSD. Modulul este controlat printr-o conexiune serială realizată

software, folosind pinii digitali 10 și 11. Programul inițializează acest modul, îl configurează și începe automat redarea unei melodii. De asemenea, detectează când o melodie s-a terminat și trece automat la următoarea.

```
SoftwareSerial mySoftwareSerial(10, 11); // tx = 10, rx = 11
DFRobotDFPlayerMini myDFPlayer;
```

Inițializarea în setup()

```
mySoftwareSerial.begin(9600);
if (!myDFPlayer.begin(mySoftwareSerial)) {
  lcd_setCursor(0, 1);
  lcd_print("DFPlayer ERR");
  while (true);
}
myDFPlayer.setTimeOut(500);
myDFPlayer.volume(20);
myDFPlayer.EQ(DFPLAYER_EQ_NORMAL);

currentTrack = 1;
myDFPlayer.play(currentTrack);
displaySong(currentTrack);
```

Detectarea sfârșitului unei melodii în loop()

```
if (myDFPlayer.available()) {
  uint8_t type = myDFPlayer.readType();
  uint16_t value = myDFPlayer.read();
  if (type == DFPlayerPlayFinished) {
    currentTrack++;
    if (currentTrack > 5) currentTrack = 1;
    myDFPlayer.play(currentTrack);
    displaySong(currentTrack);
  }
}
```

Volumul este controlat printr-un potențiometru conectat la o intrare analogică (ADC0). Codul configurează manual convertorul analog-digital al microcontrollerului pentru a citi valoarea de tensiune de la potențiometru, care este apoi transformată într-un procent între 0 și 100. Această valoare este folosită atât pentru a ajusta volumul în modulul DFPlayer, cât și pentru a fi afișată pe ecranul LCD.

Configurarea ADC-ului

```
void adc_init() {
  ADMUX = (1 << REFS0); // Referință AVcc, canal ADC0
  ADCSRA = (1 << ADEN) | (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); //
Activare + prescaler 128
}
```

Citirea valorii analogice

```
uint16_t adc_read() {
  ADMUX = (ADMUX & 0xF0) | 0; // Selectează ADC0
  ADCSRA |= (1 << ADSC);      // Start conversie
  while (ADCSRA & (1 << ADSC));
  return ADC;
}
```

Utilizarea valorii pentru volum

```
uint16_t potValue = adc_read();
int volumePercent = map(potValue, 0, 1023, 0, 100);

if (volumePercent != lastVolume) {
  lastVolume = volumePercent;
  lcd_setCursor(0, 1);
  lcd_print("Volum:     ");
  lcd_setCursor(7, 1);
  char buf[5];
  itoa(volumePercent, buf, 10);
  lcd_print(buf);
  lcd_print("%");

  int dfVol = map(volumePercent, 0, 100, 0, 30);
  myDFPlayer.volume(dfVol);
}
```

Pentru afișaj se folosește un LCD 16×2 conectat prin I2C. Comunicarea I2C este realizată direct, la nivel de registru, prin perifericul TWI al microcontrollerului. Sunt implementate funcții proprii pentru inițializare, trimiterea comenzilor și datelor către LCD, precum și funcții pentru poziționarea cursorului și scrierea textului.

Inițializare și comunicație I2C

```
void TWI_init();
void TWI_start();
void TWI_stop();
void TWI_write(uint8_t data);
```

Funcții pentru trimiterea comenzilor către LCD

```
void lcd_pulse_enable(uint8_t data);
void lcd_send_nibble(uint8_t nibble, uint8_t mode);
void lcd_send_byte(uint8_t value, uint8_t mode);
void lcd_command(uint8_t cmd);
void lcd_data(uint8_t data);
void lcd_init();
```

Funcții pentru scriere text și control cursor

```
void lcd_setCursor(uint8_t col, uint8_t row);
void lcd_clear();
void lcd_print(const char* str);
```

Pe ecranul LCD sunt afișate informații despre melodia curentă (nume prestabilite pentru fiecare piesă de la 1 la 5) și nivelul actual al volumului în procente. La fiecare schimbare de volum sau melodie, ecranul este actualizat corespunzător.

Afișarea informațiilor despre melodie și volum

```
void displaySong(int song) {
  lcd_clear();
  lcd_setCursor(0, 0);
  switch (song) {
    case 1: lcd_print("The Smiths Bigmouth Strikes Again"); break;
    case 2: lcd_print("Enjambre Impacto"); break;
    case 3: lcd_print("Surf Curse Disco"); break;
    case 4: lcd_print("Steve Lacy Some"); break;
    case 5: lcd_print("Abandoned Pools Armed To The Teeth"); break;
  }
  lcd_setCursor(0, 1);
  lcd_print("Volum: ");
  char buf[5];
  itoa(lastVolume >= 0 ? lastVolume : 0, buf, 10);
  lcd_print(buf);
  lcd_print("%");
}
```

Interacțiunea utilizatorului este facilitată de patru butoane conectate la pinii digitali 4 până la 7. Acestea sunt configurate ca intrări cu rezistențe de pull-up interne. Butoanele permit pornirea redării, oprirea ei, trecerea la melodia următoare sau revenirea la cea anterioară.

Configurarea butoanelor ca intrări cu pull-up

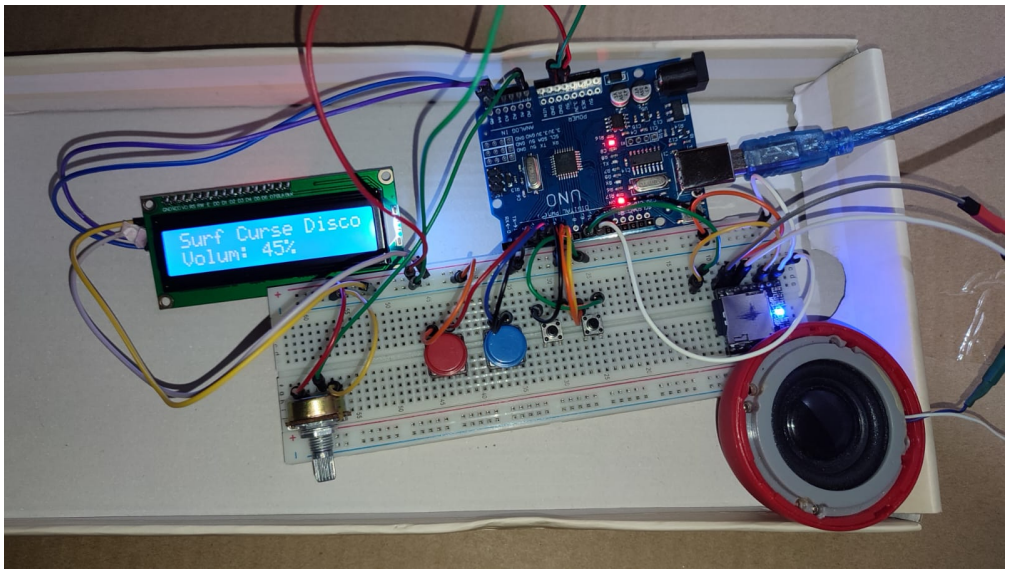
```
DDRD &= ~((1 << DDD4) | (1 << DDD5) | (1 << DDD6) | (1 << DDD7));
PORTD |= (1 << PORTD4) | (1 << PORTD5) | (1 << PORTD6) | (1 << PORTD7);
```

Citirea butoanelor și acțiunile corespunzătoare

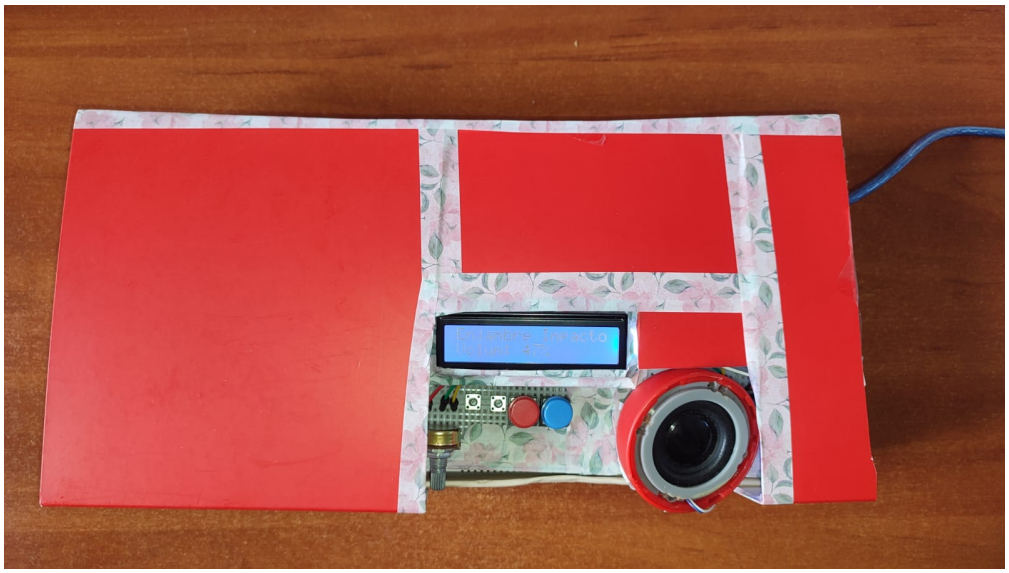
```
if ((PIND & (1 << PIND4)) == 0) { // Buton PLAY
  myDFPlayer.play(currentTrack);
  displaySong(currentTrack);
  delay(300);
}
```

Cod sursă: https://github.com/ciobanualexandra330/Proiect-PM/tree/main

Rezultate Obţinute



https://www.youtube.com/watch?v=fqUf77s7fck



https://youtu.be/TIxMRiT05GA

Concluzii

Proiectul integrează într-un mod eficient mai multe tehnologii pentru a crea un sistem complet de redare audio controlat de utilizator. Folosind modulul DFPlayer Mini pentru redarea fișierelor MP3, un ecran LCD 16×2 pentru afișarea informațiilor, un potențiometru pentru ajustarea volumului și patru butoane pentru controlul interactiv, proiectul oferă o experiență intuitivă și autonomă de utilizare. Comunicarea între componente este realizată atât serial, cât și I2C, iar controlul la nivel de registru al perifericelor TWI și ADC permite realizarea comunicației I2C cu afișajul LCD și citirea valorii analogice de la potențiometru pentru reglarea volumului.

Acest proiect mi-a oferit ocazia să lucrez direct cu un Arduino și să învăț mai multe despre partea

practică a interacțiunii cu hardware-ul. Deși nu am întâmpinat probleme legate de software, am avut dificultăți neașteptate cu redarea fișierelor MP3 — melodiile nu erau recunoscute de modulul DFPlayer, deși cardul SD părea în regulă. S-a dovedit că formatarea fișierelor era cauza, o problemă aparent banală, dar care mi-a consumat timp până am identificat-o.

Download

aciobanu_mp3player.zip

Jurnal

- 28/04/2025 - alegerea proiectului
- 05/06/2025 - achizitionarea pieselor
- 14/06/2025 - realizarea hardware-ului
- 24/06/2025 - implementarea software-ului

Bibliografie/Resurse

https://arduinoyard.com/dfplayer-mini-with-arduino/

https://projecthub.arduino.cc/arduino_uno_guy/i2c-liquid-crystal-displays-5eb615

https://docs.arduino.cc/learn/electronics/potentiometer-basics/