Guitar Tuner

- Pascu Ioana-Călina
- 332CD
- Îndrumător: Răzvan Vîrtan

Introducere

Proiectul constă în realizarea unui tuner de chitară pentru diferite tipuri de acordare.

În acest moment, cel mai accesibil și simplu mod de a acorda o chitară este printr-o aplicație pe telefon, care are opțiunea unui acordaj E standard. Însă, dacă utilizatorul dorește un alt tip de acordaj (Drop D, Open D, Drop C, etc.), opțiunea nu este, bineînțeles, gratuită.

Scopul proiectului este de a(-mi) oferi o unealtă de acordat chitara, pentru a evita frustrările și acordatul după ureche.

Descriere generală

Mod de funcționare

1. Se selectează prin intermediul unui buton tipul de acordare
2. Programul citește de pe SD frecvențele necesare pentru acordaj
3. Se iau coardele pe rând de la cea mai mică la cea mare frecvență
4. Utilizatorul ciupește coarda respectivă lângă microfon
5. Programul compară frecvențele
6. Se afișează pe ecranul LCD rezultatul comparației (coarda mai trebuie strânsă/slăbită sau frecvența este corectă)
7. Dacă coardele au fost acordate, se afișează un mesaj pe ecran și se trece la ecranul de start

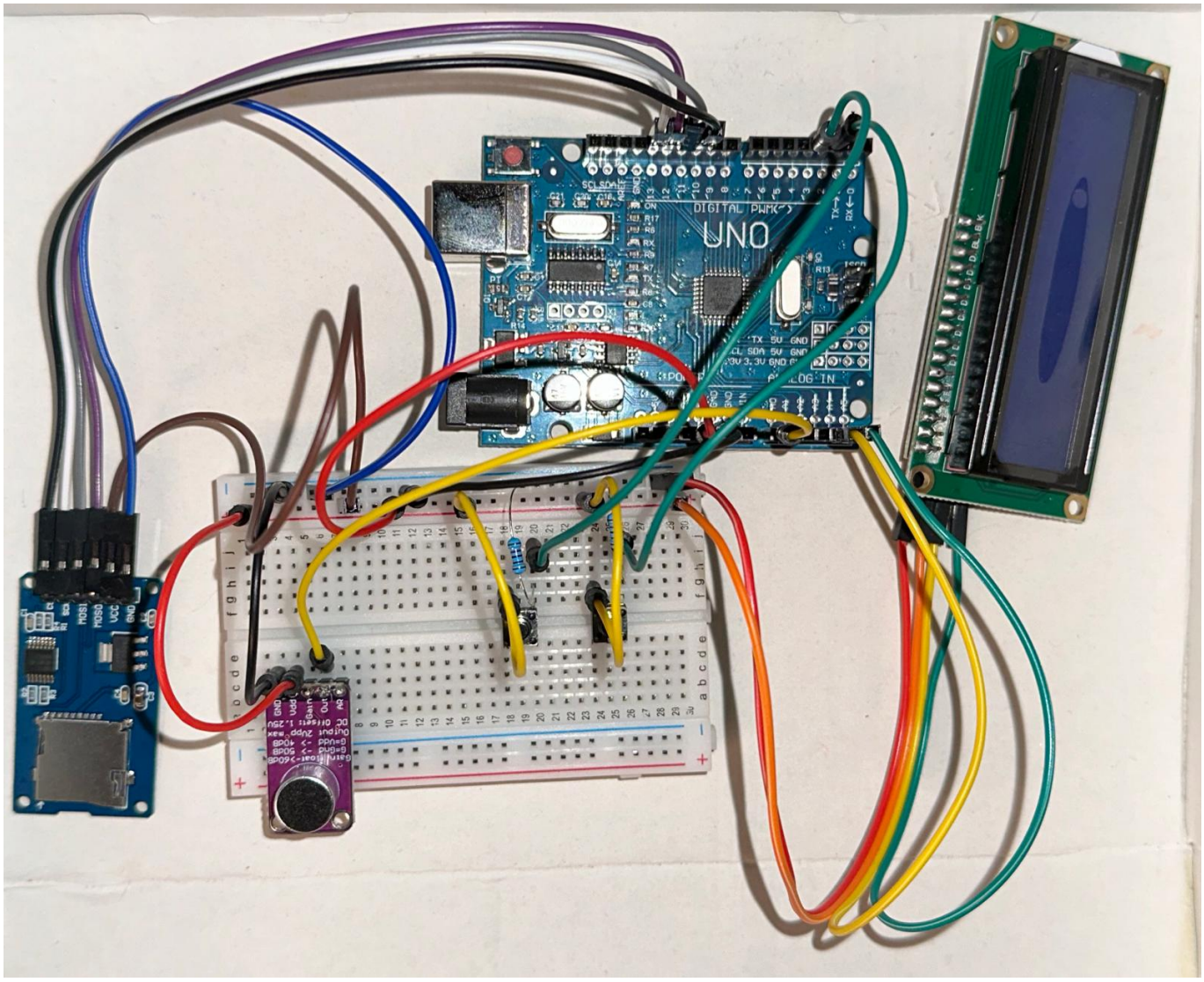
Schemă bloc a modulelor

Hardware Design

Listă de piese utilizate:

- Arduino UNO3 ATMega328P
- Breadboard 400 poli
- Cabluri de conexiune
- 2 rezistențe 10k ohmi
- 2 butoane
- LCD 1602 cu I2C
- amplificator microfon MAX9814
- modul microSD

Diagramă de conectare



Software Design

Link pentru repo-ul de Git: https://github.com/ultracln/PM_GUITAR_TUNER

Mediu de dezvoltare

Arduino IDE

Biblioteci folosite

- arduinoFFT.h → pentru a calcula Transformata Fourier Rapidă (FFT)
- Wire.h, LiquidCrystal_I2C.h → pentru a comunica și controla un display LCD

- SD.h, SPI.h → pentru a comunica cu un card SD

Procesul de Transformare a Semnalului Analogic în Frecvență folosind FFT

- Colectarea Datelor de la Microfon

Semnalul analogic de la microfon este citit în mod repetat pentru a colecta un număr fix de mostre (SAMPLES). Aceste date sunt stocate în vectorul vReal, iar vectorul vImag este inițializat cu zero, deoarece datele inițiale sunt semnale reale fără componentă imaginară.

```
for (int i = 0; i < SAMPLES; i++) {
    microSeconds = micros();
    vReal[i] = analogRead(micPin);
    vImag[i] = 0;
    while ((micros() - microSeconds) < samplingPeriod) {
    // așteaptă până când trece timpul necesar pentru următoarea mostră
    }
    microSeconds += samplingPeriod;
}
```

- Aplicarea Ferestrei Hamming

Ferestrele sunt utilizate pentru a reduce efectele marginilor semnalului, care pot produce erori în rezultatele FFT. Fereastra Hamming este aplicată folosind o funcție din librăria FFT.

- Calcularea FFT și determinarea frecvenței dominante

Se calculează FFT pe datele colectate. După ce FFT este calculată, identificăm frecvența cea mai puternică (dominantă) în semnal. Aceasta este frecvența principală a semnalului, care corespunde notei muzicale detectate.

Utilizarea Rezultatelor FFT pentru Acordare

În modul de acordare, frecvența calculată este comparată cu frecvența corectă pentru coarda curentă. Mesajele afișate pe LCD informează utilizatorul dacă frecvența este prea joasă, prea înaltă sau corectă. Se trece automat la următoarea coardă odată ce coarda curentă este acordata corect. Când utilizatorul termină de acordat toate coardele, se afișează un mesaj pe LCD și se trece la meniul de start.

Încărcarea Frecvențelor din Fișiere SD

Funcția loadTuningFromFile() încarcă frecvențele de acordare dintr-un fișier text de pe cardul SD. Fiecare fișier este de forma ”<coardă>,<frecvență>”. De exemplu, pentru modul standard:

```
E,82.41
A,110.00
D,146.83
G,196.00
B,246.94
e,329.63
```

Rezultate Obţinute

În arhiva de mai jos se poate vedea funcționalitatea proiectului pe un exemplu: trec prin cele 6 tipuri de tuning, îl aleg pe cel Standard; în exemplul acesta toate coardele au frecvența corectă, în afară de cea mai înaltă (e), care este dezacordată și are frecvența joasă. Așa cum era de așteptat, pe LCD se afișează mesajul “In tune” când o coardă este acordată, și “Too low”/“Too high” când încerc să acordez “e”.

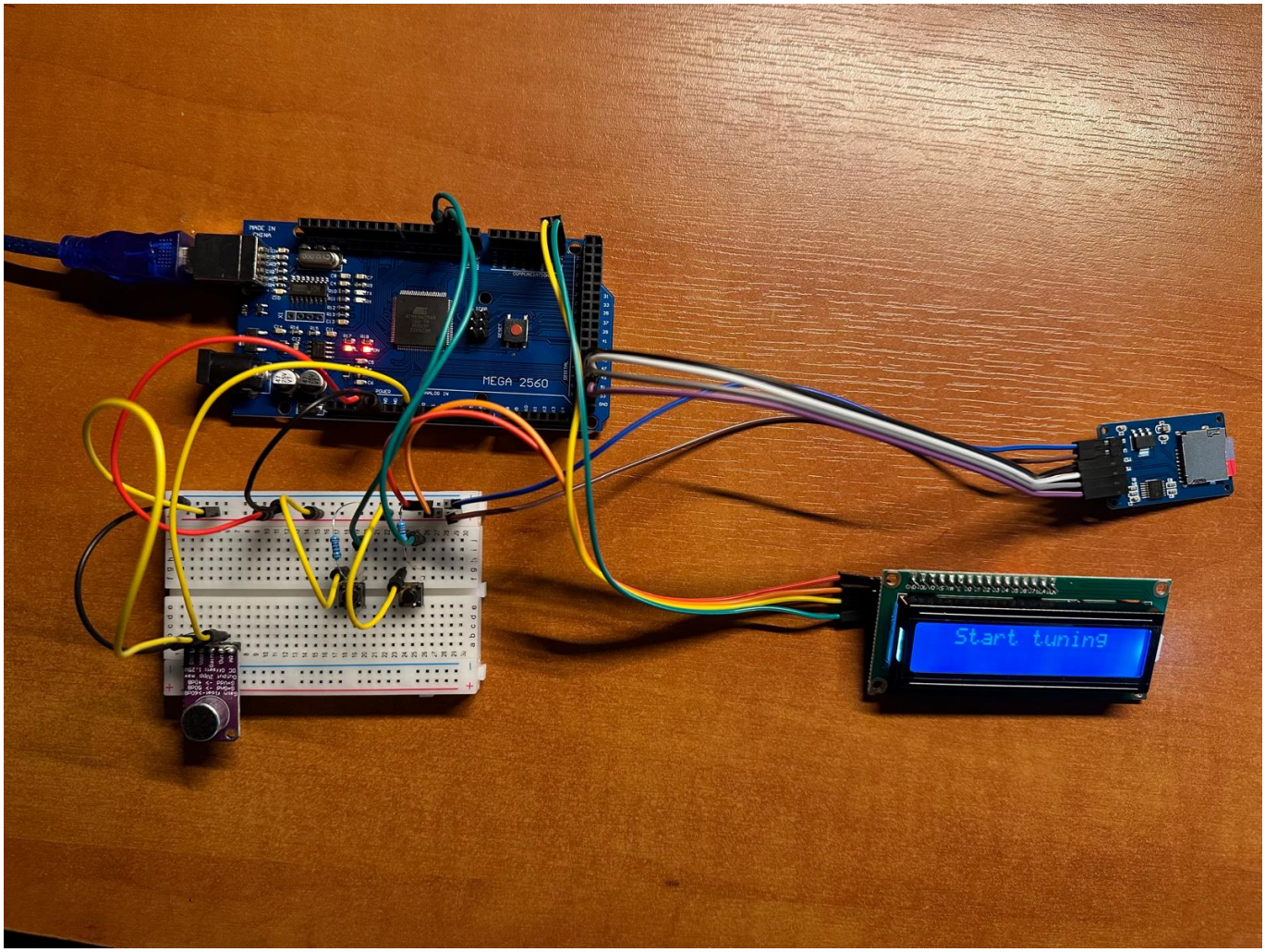
https://youtu.be/00D2SWCyFtE

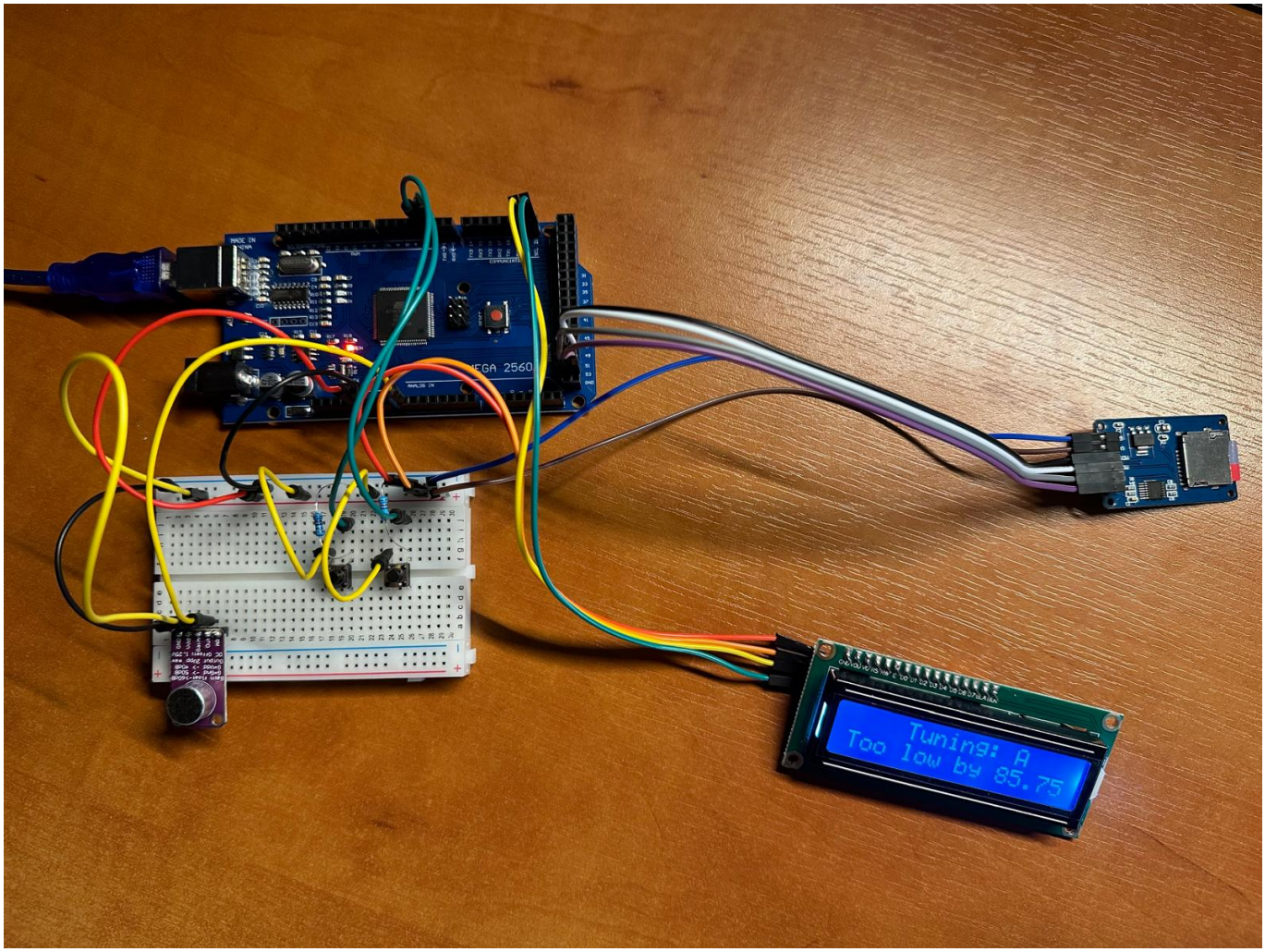
Codul funcționează cu o marjă de eroare a frecvenței calculate de +- 10Hz, întrucât calculul Transformatei Fourier are uneori rezultate neașteptate din cauza zgomotului.

A trebuit să experimentez cu mai multe valori pentru SAMPLES și SAMPLING_FREQUENCY pentru a găsi o formulă intermediară de calcul a frecvenței finale (adică jumătatea peak-ului), luând în calcul zgomotul. Cu cât valoarea SAMPLES este mai mare, cu atât se obține o medie mai precisă a semnalului în timp, ceea ce ajută la reducerea zgomotului.

Inițial, valorile pentru SAMPLES au fost limitate (multipli de 2, cel mai mare fiind 128) pentru Arduino Uno și se obțineau rezultate foarte depărtate de cele dorite. Cei doi vectori folosiți în calculul FFT ocupau mare parte din capacitatea SRAM a microprocesorului și nu mai rămânea destul spațiu pentru restul variabilelor și a funcțiilor. În punctul acesta, nu era corect nici calculul FFT și nici componentele nu funcționau corespunzător .

Deci, am optat pentru un Arduino Mega 2560; cablajul diferă puțin față de cel inițial:


MEGA 2560
Start tuning



Concluzii

Proiectul a fost destul de dificil din cauza calculului frecvenței folosind transformata Fourier Rapidă, care nici acum nu produce rezultate apropiate de frecvența dorită. Am încercat mai multe metode pentru a analiza semnalul analogic: Zero-Crossing, Autocorrelation, Fast Hartley Transform (asemănătoare cu FFT-ul, dar mai rapidă în teorie), dar nu am avut rezultate mulțumitoare cu niciuna. Deci, o mare parte din timpul alocat proiectului a fost ocupată de găsirea și înțelegerea algoritmilor. Desigur, pot exista erori și din cauza modulului de microfon amplificator, dar și aici am încercat să experimentez cu opțiunile de Gain și Attack/Release, neajungând la un rezultat mai frumos.

În orice caz, proiectul a fost interesant de realizat atât pe partea hardware (fun), cât și pe partea software (not so fun). Sper să pot găsi o soluție pentru îmbunătățirea calculului frecvenței pe viitor, pentru a putea folosi tuner-ul fără probleme.

Bibliografie/Resurse

- Arduino x MAX9814: https://youtu.be/2waBFdEBZDg?si=32I7n15I-bm2EmID
- Adresa memoriei LCD: https://youtu.be/CvqHkXeXN3M?si=unwAR5wdvqATehaS
- Utilizare FFT.h pentru detectarea frecvenței în Arduino: https://youtu.be/wbeV0J30LGQ?si=CxhIiYDKvoaHAf0U

Export to PDF