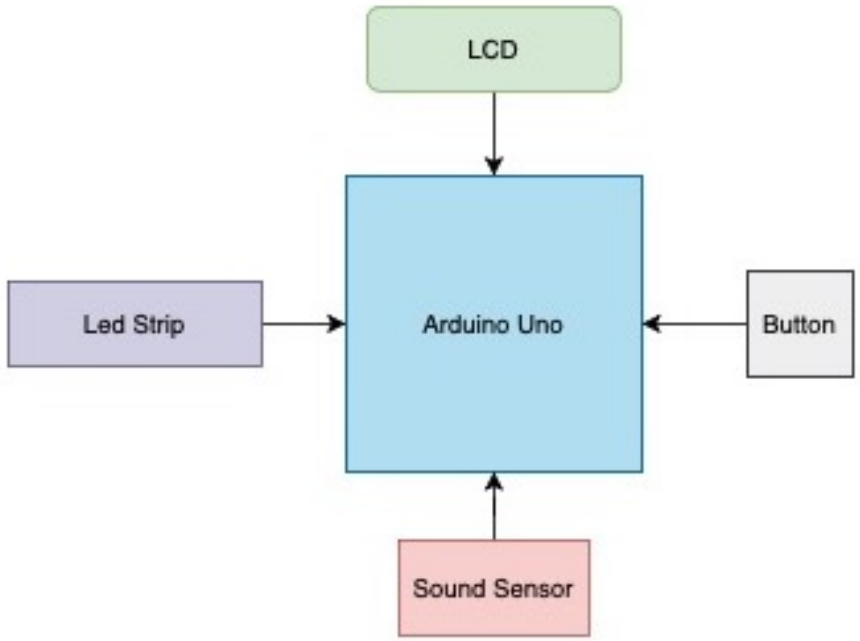
# Sound Reactive VU Meter


- Nume: State Andreea-Rebecca
- Grupa: 331CA


## Introducere

Sound Reactive VU Meter este un dispozitiv care răspunde la sunet prin afișarea nivelului de semnal audio în timp real, prin intermediul unor indicatori vizuali (LED-uri), care se vor mișca în funcție de intensitatea sunetului. Acest dispozitiv este util pentru monitoriza nivelului de volum al muzicii sau al altor surse audio într-un mod vizual atrăgător, datorită LED-urilor colorate care se aprind și se stinge în funcție de nivelul de volum al sunetului, creând astfel o reprezentare vizuală a intensității sunetului.

## Descriere generală




Banda led va avea mai multe tipuri de aprindere al beculețelor:

- de jos în sus
- de sus în jos

- din interior spre exterior
- din exterior spre interior

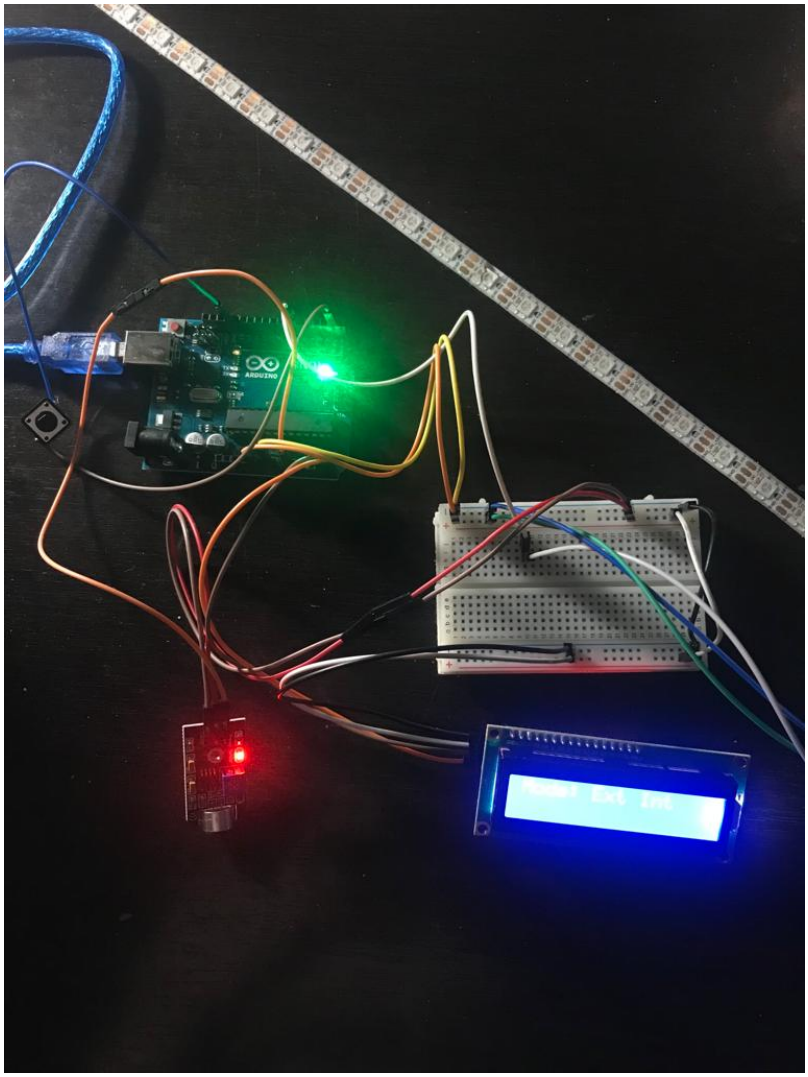
Utilizatorul poate să treacă de la un mod de aprindere la altul prin intermediul unui buton care va cicla prin cele 4 moduri de aprindere. Dacă timp de 10 de secunde nu este detectată nicio apăsare de buton, se va trece automat în următorul mod de aprindere. Pe ecranul LCD va apărea numele modului curent de aprindere al beculețelor și o scară de punctulețe care formează o reprezentare vizuală a creșterii intensității sunetului.

# Hardware Design

### Listă de piese

- Arduino Uno R3 Atmega328p
- Led Strip WS2812B
- Senzor sunet LM393
- Buton
- LCD I2C 16×2

### Schema electrică



## Software Design

Codul proiectului a fost dezvoltat în Arduino IDE și am folosit bibliotecile FastLED pentru banda led și LiquidCrystal_I2C pentru afișarea pe ecranul LCD.

Funcții utilizate în cod:

- void setup() - inițializez banda led si ecranul LCD și configurez întreuperile pentru buton și pentru TIMER1
- void myISR() - atunci când butonul este apăsat trec la următorul mod de afișare al led-urilor, resetez timer-ul și scriu pe ecranul LCD numele modului curent de afișare
- void displayLcdMode() - afișez pe LCD textul cu numele modului curent de afișsare al led-urilor
- void displayHearts(int noHearts) - afișez pe LCD inimioarele care reprezintă o scară a intensității sunetului
- struct my_leds colorWheel(int s) - în această funcție setez câte led-uri trebuie aprinse, culoarea acestora și numărul de inimioare afișate pe LCD (folosesc această funcție doar pentru modul 0 si modul 1)
- displayLedsAndHearts2(), displayLedsAndHearts3() - pentru modul 2 sau 3 trece prin vectorul de

led-uri și setează culoarea led-urilor și apelează funcția de afișsare a inimioarelor de pe ecranul LCD
- upDown(int s), downUp(int s), intExt(int s), extInt(int s) - în funcție de valoarea citită de la senzorul de sunet stabilește intervalul de led-uri care trebuie aprinse si apelează funcția de displayLedsAndHearts() corespunzătoare modului de afișare
- void loop() - citesc inputul de la senzorul de sunet, verific dacă este nevoie să modific textul de pe ecranul LCD (dacă a fost apăsat butonul sau dacă au trecut 10 secunde de când nu a mai fost apăsat butonul) și verific care este modul curent de afișare al led-urilor și apelez funcția corespunzătoare

În structura my_leds păstrez câte led-uri trebuie aprinse, culoarea lor și numărul de inimi.

Banda led are 4 moduri de afișare care se schimbă fie la o apăsare de buton, fie dupa 10 secunde în care nu a fost detectată nicio apăsare de buton.

În implementare am folosit întreruperi pentru butonul de switch cu ajutorul căruia iterez prin cele 4 moduri de afișare ale led-urilor. Prima apăsare de buton face tranziția de la modul 0 la modul 1, a doua apăsare de buton face tranziția de la modul 1 la modul 2, a treia apăsare face tranziția de la modul 2 la modul 3, iar a patra apăsare face tranziția de la modul 3 la modul 0, astfel reluându-se ciclul.

În cazul în care timp de 10 secunde nu a fost detectată nicio apăsare de buton am folosit TIMER1 pentru a face tranziția către următorul mod de afișare.

De asemenea, în cazul în care este detectată o apăsare de buton timerul este resetat pentru a evita situația în care unul dintre modurile de afișare ar fi durat mai puțin de 10 secunde.

Inițial am vrut să folosesc funcția de map de la arduino pentru a decide câte led-uri să se afișeze, dar efectul vizual rezultat nu era unul așa de frumos, întrucât numărul de led-uri aprinse se schimba brusc. Așa că am decis să verific eu de de mână pentru fiecare mod de aprindere câte led-uri să se aprindă în funcție de valoara primită de la senzorul de sunet. După multe testări cu diverse inputuri și după ajustarea senzitivității senzorului, am găsit combinațiile de input și număr de led-uri aprinse astfel încât să obțin un efect vizual gradual. Faptul ca eu decid pentru fiecare mod în parte ce beculețe să se aprindă m-a ajutat să obțin efectul vizual dorit, dar a rezultat în destul de mult cod de scris.

# Rezultate Obţinute

# Concluzii

Proiectul este funcțional și am reușit să realizez tot ce mi-am propus să fac pentru acesta. Singurul lucru pe care aș vrea să îl mai îmbunătățesc pe viitor este afișarea pe ecranul LCD. Întrucât se primesc foarte multe imputuri de la senzorul de sunet, în momentul în care afișez inimioarele care reprezintă intensitatea sunetului textul care apare pe ecran își pierde din claritate.

# Download

Arhiva ce conține codul sursă al proiectului: soundreactivemeter.zip

# Bibliografie/Resurse

FastLED: https://github.com/FastLED/FastLED/wiki/Basic-usage

LiquidCrystal_I2C: https://reference.arduino.cc/reference/en/libraries/liquidcrystal-i2c/

Întrerupere buton: https://riptutorial.com/arduino/example/9856/interrupt-on-button-press

Datasheet ATmega328P: https://ocw.cs.pub.ro/courses/_media/pm/atmel-7810-automotive-microcontrollers-atmega328p_datasheet.pdf

Export to PDF

From:
http://ocw.cs.pub.ro/courses/ - **CS Open CourseWare**

Permanent link:
**http://ocw.cs.pub.ro/courses/pm/prj2023/apredescu/vumeter**

Last update: **2023/05/30 08:44**