# Udristioiu Victor: Smart Music Reactive LED Lamp

## Introducere

Proiectul consta intr-o lampa inteligenta care reactioneaza la muzica, controlabila prin Bluetooth. Aceasta lampa este capabila sa interpreteze semnalele audio si sa sincronizeze efectele de lumina in timp real cu ritmul si intensitatea muzicii. Cu ajutorul unei aplicatii mobile, utilizatorii se pot conecta prin Bluetooth la lampa si sa controleze culorile, modelele si alte efecte luminoase, creand astfel o experienta vizuala unica si personalizata.

## Descriere generală

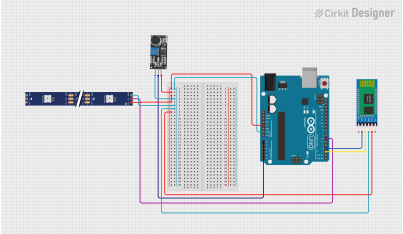
Specificatii generale:

- Modulul Bluetooth ii va permite utilizatorului sa comande banda cu LEDuri prin intermediul uC-ului.
- Exista si un mod ambiental, ce va folosi datele primite de la senzorul de sunet pentru a schimba intensitatea si culorile in fuctie de ritmul si volumul muzicii.

## Hardware Design

Componente:

1. Arduino UNO
2. Senzor Sunet 5V
3. Modul Bluetooth HC-05
4. Banda LED Adresabila WS2812
5. Breadboard




## Software Design

```
#include "FastLED.h"
#include <SoftwareSerial.h>

uint8_t r = 0;
uint8_t g = 0;
uint8_t b = 0;
uint8_t a = 0;

SoftwareSerial MyBlue(2, 3); // RX | TX

#define NUM_LEDS  60
#define PIN       6
#define INPUT_SIZE 32
#define ANALOG_READ A0
const byte MAX_STRING_LEN = 32;
char incoming_value = 0;

int state = 0;
int prev_state = 0;

CRGB leds[NUM_LEDS];

char inputString[MAX_STRING_LEN];
byte strLen   = 0;

void setup() {
  state = 0;
  Serial.begin(9600);
  MyBlue.begin(9600);
  FastLED.addLeds<WS2812B, PIN, GRB>(leds,
NUM_LEDS).setCorrection(TypicalLEDStrip);
  FastLED.setBrightness(100);
  randomSeed(analogRead(0));
}

boolean processSerial() {
```

```
  while (MyBlue.available()) {
    char inChar = (char)MyBlue.read();

    if ((inChar == '\n') || (inChar == '\r'))
      return true;

    if (strLen < (MAX_STRING_LEN - 1)) {
      inputString[strLen++] = inChar;
      inputString[strLen]   = '\0';
    }
  }

  return false;
}

void compute_next_state()
{
  if (processSerial()) {
      if(strstr(inputString, "on"))
      {
        state = 1;
      }
      else if (strstr(inputString, "off"))
      {
        state = 2;
      }
      else if (strstr(inputString, "music"))
      {
        state = 4;
      }
      else if (strstr(inputString, "&"))
      {
        char* token = strtok(inputString, "&");
        int count = 0;
        while(token)
        {
          if (count == 0)
            r = (uint8_t)atoi(token);

          if (count == 1)
            g = (uint8_t)atoi(token);

          if (count == 2)
            b = (uint8_t)atoi(token);
          count++;
          token = strtok(NULL, "&");
        }
        state = 3;
      }

      inputString[0] = '\0';
```

```
      strLen         = 0;
    }
}

void loop() {
  if (state == 0)
  {
    compute_next_state();
  }
  if (state == 1)
  {
    FastLED.setBrightness(100);
    for(int i = 0; i < NUM_LEDS; i++) {
        leds[i].setRGB(255, 255, 255);
    }
    FastLED.show();
    state = 0;
  }
  if (state == 2)
  {
    FastLED.setBrightness(100);
    for(int i = 0; i < NUM_LEDS; i++) {
        leds[i].setRGB(0, 0, 0);
    }
    FastLED.show();
    state = 0;
  }
  if (state == 3)
  {
    FastLED.setBrightness(100);
     for(int i = 0; i < NUM_LEDS; i++) {
        leds[i].setRGB(r, g, b);
    }
    FastLED.show();
    state = 0;
  }
  if (state == 4)
  {
     for(int i = 0; i < NUM_LEDS; i++) {
        leds[i].setRGB(0, 0, 0);
    }
    FastLED.show();

    long sum = 0;
    for(int i=0; i<300; i++)
    {
      sum += analogRead(ANALOG_READ);
    }

    sum = sum/300;
    FastLED.setBrightness(random(100));
```

```
    int no_leds = map(sum, 40, 900, 0, 59);

    if(no_leds < NUM_LEDS)
    {
      for(int i = 0; i < no_leds; i++) {
        leds[i].setRGB(random(255), random(255), random(255));
      }
      FastLED.show();
    }
    delay(10);
    if (!MyBlue.available())
      state = 4;
    else
      state = 0;
  }
}
```

## Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

## Concluzii

## Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună 😉.

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

## Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

# Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF