Faculty of Electronics,
Telecommunications and
Information Technology

Architecture of the Microprocessors

Miniproject Title:

Sound-Activated LED Strip using Arduino

Student name: Cantor Lavinia-Denisa

Contents

1. Introduction

This project, **Sound-Activated LED Strip using Arduino**, focuses on creating a reactive lighting system where an addressable LED strip changes its brightness based on the sound intensity in the environment. Using an **Arduino Uno R3**, a **sound sensor**, and the **FastLED library**, the system can dynamically adjust the light output in real-time to produce visually engaging effects.

Goals

- To read and process sound intensity using a sensor.
- To use the processed signal to control the brightness of an addressable LED strip.
- To create a real-time, visually appealing light display.

This **Sound-Activated LED Strip with Arduino** project can be used for various purposes, such as dynamic decorative lighting for parties and events, music visualization for DJs or studios, and interactive art installations.

2. Theoretical Background

To understand the functioning of this system, it’s important to know the following key components and concepts:

Arduino Uno R3

The Arduino Uno R3 is a popular microcontroller board based on the ATmega328P chip. It features:

- 14 digital I/O pins and 6 analog input pins.
- A clock speed of 16 MHz.
- Integrated USB interface for programming and power supply.

In this project, the Arduino reads the sound signal, processes it, and adjusts the LED brightness.

Sound Sensor

The sound sensor (commonly KY-038 or similar) detects sound waves in the environment and outputs an **analog signal** proportional to the sound intensity (changes depending on how loud the sound is) . Key features:

- **OUT Pin (Analog Output):** Produces a signal between 0V and 5V, depending on the sound level and sends it to the Arduino.
- **Sensitivity Adjustment:** A potentiometer on the sensor lets you adjust how sensitive the sensor is to sound (we can adjust it with a small screw).

Addressable LED Strip

The project uses an addressable LED strip (WS2812B), where each LED can be controlled individually using a single data pin. Key characteristics:

- **Data Pin:** Serial communication line for controlling the LEDs.
- **RGB Control:** Each LED has red, green, and blue color channels (0–255 intensity for each).
- **Power Consumption:** Requires an external power supply for longer strips (LEDs can draw a lot of current).

FastLED Library

The FastLED library simplifies the process of controlling addressable LEDs. It:

- Handles low-level communication protocols for LED strips.
- Provides functions for setting colors, brightness, and animations.
- Without this library, controlling each LED manually would be much harder.

Signal Processing

- **Analog-to-Digital Conversion (ADC):** The Arduino converts the analog signal (0V to 5V) from the sound sensor into a number between 0 and 1023.
- **Normalization:** The analog signal is mapped into a usable range (0–200 to match the brightness of the LEDs).

- **Thresholding:** Values below a minimum threshold (MINVAL) are ignored to filter out noise (like background noise).

3. Implementation

3.1 Hardware Design

Components

1. **Arduino Uno R3:** Microcontroller board for processing sensor data and controlling LEDs.
2. **Sound Sensor (KY-038):** Captures sound and outputs analog values.
3. **LED Strip (WS2812B):** Displays brightness effects based on sound intensity.
4. **External Power Supply:** Provides sufficient current for the LED strip.

Connections

1. **Sound Sensor:**
   - OUT pin → Arduino A0.
   - GND pin → Arduino GND.
   - VCC pin → Arduino 5V.
2. **LED Strip:**
   - Data pin → Arduino D3.
   - GND → Arduino GND.
   - VCC → External 5V supply.

3.2 Software (Detailed Code Explanation)

The code was written in the Arduino IDE and uses the FastLED library for controlling the LED strip.

```
#include <FastLED.h>              // Include FastLED library for controlling the strip
#define NUMLEDS 41                  // Number of LEDs in the strip
#define DATA_PIN 3                // Arduino pin connected to LED strip data line
#define SENSITIVITY 600             // Maximum sensor value to consider
#define MAX_BRIGHTNESS 200          // Maximum brightness for LEDs
#define ENVELOPE_PIN A0             // Analog pin for sound sensor
#define SATURATION 200              // Fixed color saturation
#define MINVAL 60                 // Minimum sensor value to filter out noise
#define HUE_INIT 100              // Initial hue value for the LED color (0-255 scale,
                                  //100 = green)

CRGB leds[NUMLEDS];                 // Array to store color data for each LED
uint8_t brightness [NUMLEDS];   // Array to store brightness values for each LED
uint8_t hue [NUMLEDS];            // Array to store hue values for each
int analogVal;                  // Variable to store the current sound sensor reading

void setup() {
  pinMode(ENVELOPE_PIN, INPUT);    // Set the sound sensor pin as an input
  FastLED.addLeds<WS2812B, DATA_PIN>(leds, NUMLEDS); // Initialize the
                                                  // LED strip with WS2812B protocol
}
```

```
void loop() {
  analogVal = analogRead(ENVELOPE_PIN);  // Read the analog value from the
                                         //    sound sensor

  if (analogVal > SENSITIVITY)           // Check if the sensor value exceeds the
                                           defined sensitivity
    analogVal = SENSITIVITY;             // If true, cap it at the maximum allowed
                                         // value (600) to prevent overflow

  if (analogVal < MINVAL)                // Check if the sensor value is below the noise
                                         // threshold
    analogVal = 0;                       // If true, set the value to 0 to ignore
                                         //background noise

  BrightnessReactive();          // Update the brightness of the LEDs based on the
                                  // sound level

  FastLED.show();                   // Send the updated color data to the LED strip
}

void BrightnessReactive() {
  uint8_t val = map(analogVal, 0, SENSITIVITY, 0, MAX_BRIGHTNESS);
                                 // Map the sound sensor value to a brightness range
for (int i = 0; i <= NUMLEDS; i++) {            // Loop through all LEDs
   leds[i] = CHSV(hue, SATURATION, val); // Set the LED color using HSV
                                          (Hue, Saturation, Value)
                                        // Hue: determines color, Saturation:
                                         //color intensity, Value: brightness
  }
}
```

4. Conclusions

This project successfully demonstrates the integration of sound sensors with addressable LEDs to create a dynamic, sound-reactive light system. It highlights:

- The real-time processing capabilities of the Arduino Uno.
- The flexibility and power of the FastLED library.
- The practical use of sensors for interactive lighting applications.

5. Bibliography

1. https://randomnerdtutorials.com/
2. https://chatgpt.com/
3. https://www.circuito.io/blog/arduino-uno-pinout/
4. https://circuitdigest.com/microcontroller-projects/arduino-sound-sensor-tutorial
5. https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/
6. https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf
7. https://docs.arduino.cc/hardware/uno-rev3/
8. https://github.com/FastLED/FastLED