МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра систем автоматизации управления

Дата сдачи на проверку:

11 февраля 2025 г.

Проверено:

11 февраля 2025 г.

**Отчет по практической работе №2**

**«ПЬЕЗОЭЛЕМЕНТ, МИКРОСХЕМЫ»**

по дисциплине

«Информационные технологии»

Вариант 10

Выполнил студент гр. УТБ-1301-02-00 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Борисов И. А/

(Подпись)

Проверил ст. преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Шмакова Н. А/

(Подпись)

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2025 г.

Киров 2025

Цель работы: ознакомление с элементами Arduino IDE, изучение основ работы со средой для программирования, а также сборка схем с пьезодинамиком, микросхемами, дисплеем и моторами.

1 Азбука Морзе

Индивидуальное задание Морзе. Используйте мелодии для пьезодинамика из вложения (можно выбрать любые 2). Добавьте световую индикацию (не менее 3-х светодиодов). Добавьте кнопку для переключения мелодий.

Схема сборки Азбуки Морзе представлена на рисунке 1.1

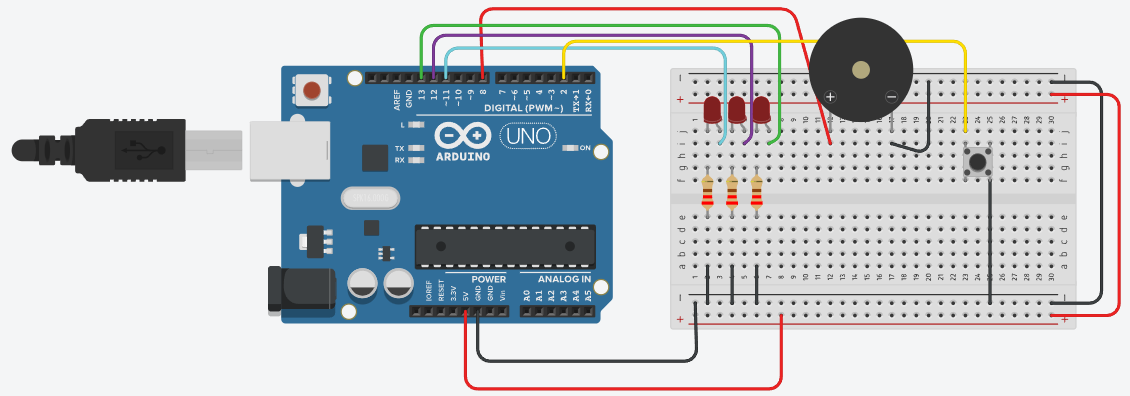


Рисунок 1.1 – Схема сборки на макетной плате

Принципиальная схема представлена на рисунке 1.2

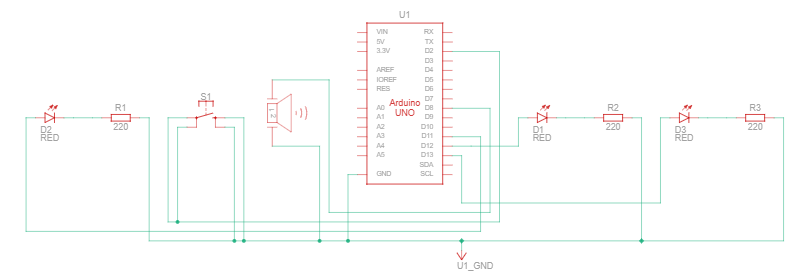


Рисунок 1.2 – Принципиальная схема

Листинг программы

#define NOTE\_B0 31  
 #define NOTE\_C1 33  
 #define NOTE\_CS1 35  
 #define NOTE\_D1 37  
 #define NOTE\_DS1 39  
 #define NOTE\_E1 41  
 #define NOTE\_F1 44  
 #define NOTE\_FS1 46  
 #define NOTE\_G1 49  
 #define NOTE\_GS1 52  
 #define NOTE\_A1 55  
 #define NOTE\_AS1 58  
 #define NOTE\_B1 62  
 #define NOTE\_C2 65  
 #define NOTE\_CS2 69  
 #define NOTE\_D2 73  
 #define NOTE\_DS2 78  
 #define NOTE\_E2 82  
 #define NOTE\_F2 87  
 #define NOTE\_FS2 93  
 #define NOTE\_G2 98  
 #define NOTE\_GS2 104  
 #define NOTE\_A2 110  
 #define NOTE\_AS2 117  
 #define NOTE\_B2 123  
 #define NOTE\_C3 131  
 #define NOTE\_CS3 139  
 #define NOTE\_D3 147  
 #define NOTE\_DS3 156  
 #define NOTE\_E3 165  
 #define NOTE\_F3 175  
 #define NOTE\_FS3 185  
 #define NOTE\_G3 196  
 #define NOTE\_GS3 208  
 #define NOTE\_A3 220  
 #define NOTE\_AS3 233  
 #define NOTE\_B3 247  
 #define NOTE\_C4 262  
 #define NOTE\_CS4 277  
 #define NOTE\_D4 294  
 #define NOTE\_DS4 311  
 #define NOTE\_E4 330  
 #define NOTE\_F4 349  
 #define NOTE\_FS4 370  
 #define NOTE\_G4 392  
 #define NOTE\_GS4 415  
 #define NOTE\_A4 440  
 #define NOTE\_AS4 466  
 #define NOTE\_B4 494  
 #define NOTE\_C5 523  
 #define NOTE\_CS5 554  
 #define NOTE\_D5 587  
 #define NOTE\_DS5 622  
 #define NOTE\_E5 659  
 #define NOTE\_F5 698  
 #define NOTE\_FS5 740  
 #define NOTE\_G5 784  
 #define NOTE\_GS5 831  
 #define NOTE\_A5 880  
 #define NOTE\_AS5 932  
 #define NOTE\_B5 988  
 #define NOTE\_C6 1047  
 #define NOTE\_CS6 1109  
 #define NOTE\_D6 1175  
 #define NOTE\_DS6 1245  
 #define NOTE\_E6 1319  
 #define NOTE\_F6 1397  
 #define NOTE\_FS6 1480  
 #define NOTE\_G6 1568  
 #define NOTE\_GS6 1661  
 #define NOTE\_A6 1760  
 #define NOTE\_AS6 1865  
 #define NOTE\_B6 1976  
 #define NOTE\_C7 2093  
 #define NOTE\_CS7 2217  
 #define NOTE\_D7 2349  
 #define NOTE\_DS7 2489  
 #define NOTE\_E7 2637  
 #define NOTE\_F7 2794  
 #define NOTE\_FS7 2960  
 #define NOTE\_G7 3136  
 #define NOTE\_GS7 3322  
 #define NOTE\_A7 3520  
 #define NOTE\_AS7 3729  
 #define NOTE\_B7 3951  
 #define NOTE\_C8 4186  
 #define NOTE\_CS8 4435  
 #define NOTE\_D8 4699  
 #define NOTE\_DS8 4978  
  
 #define melodyPin 8  
 #define BUTTON\_PIN 2  
  
*//Mario main theme melody*int melody[] = {  
 NOTE\_E7, NOTE\_E7, 0, NOTE\_E7,  
 0, NOTE\_C7, NOTE\_E7, 0,  
 NOTE\_G7, 0, 0, 0,  
 NOTE\_G6, 0, 0, 0,  
  
 NOTE\_C7, 0, 0, NOTE\_G6,  
 0, 0, NOTE\_E6, 0,  
 0, NOTE\_A6, 0, NOTE\_B6,  
 0, NOTE\_AS6, NOTE\_A6, 0,  
  
 NOTE\_G6, NOTE\_E7, NOTE\_G7,  
 NOTE\_A7, 0, NOTE\_F7, NOTE\_G7,  
 0, NOTE\_E7, 0, NOTE\_C7,  
 NOTE\_D7, NOTE\_B6, 0, 0,  
  
 NOTE\_C7, 0, 0, NOTE\_G6,  
 0, 0, NOTE\_E6, 0,  
 0, NOTE\_A6, 0, NOTE\_B6,  
 0, NOTE\_AS6, NOTE\_A6, 0,  
  
 NOTE\_G6, NOTE\_E7, NOTE\_G7,  
 NOTE\_A7, 0, NOTE\_F7, NOTE\_G7,  
 0, NOTE\_E7, 0, NOTE\_C7,  
 NOTE\_D7, NOTE\_B6, 0, 0  
};  
*//Mario main them tempo*int tempo[] = {  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
  
 9, 9, 9,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
  
 9, 9, 9,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
};  
*//Underworld melody*int underworld\_melody[] = {  
 NOTE\_C4, NOTE\_C5, NOTE\_A3, NOTE\_A4,  
 NOTE\_AS3, NOTE\_AS4, 0,  
 0,  
 NOTE\_C4, NOTE\_C5, NOTE\_A3, NOTE\_A4,  
 NOTE\_AS3, NOTE\_AS4, 0,  
 0,  
 NOTE\_F3, NOTE\_F4, NOTE\_D3, NOTE\_D4,  
 NOTE\_DS3, NOTE\_DS4, 0,  
 0,  
 NOTE\_F3, NOTE\_F4, NOTE\_D3, NOTE\_D4,  
 NOTE\_DS3, NOTE\_DS4, 0,  
 0, NOTE\_DS4, NOTE\_CS4, NOTE\_D4,  
 NOTE\_CS4, NOTE\_DS4,  
 NOTE\_DS4, NOTE\_GS3,  
 NOTE\_G3, NOTE\_CS4,  
 NOTE\_C4, NOTE\_FS4, NOTE\_F4, NOTE\_E3, NOTE\_AS4, NOTE\_A4,  
 NOTE\_GS4, NOTE\_DS4, NOTE\_B3,  
 NOTE\_AS3, NOTE\_A3, NOTE\_GS3,  
 0, 0, 0  
};  
*//Underwolrd tempo*int underworld\_tempo[] = {  
 12, 12, 12, 12,  
 12, 12, 6,  
 3,  
 12, 12, 12, 12,  
 12, 12, 6,  
 3,  
 12, 12, 12, 12,  
 12, 12, 6,  
 3,  
 12, 12, 12, 12,  
 12, 12, 6,  
 6, 18, 18, 18,  
 6, 6,  
 6, 6,  
 6, 6,  
 18, 18, 18, 18, 18, 18,  
 10, 10, 10,  
 10, 10, 10,  
 3, 3, 3  
};  
  
int status = 0;  
void setup(void)  
{  
 pinMode(8, OUTPUT); *//buzzer* for (int pin = 11; pin <= 13; ++pin) pinMode(pin, OUTPUT); *//led* pinMode(2, INPUT\_PULLUP); *// btn* Serial.begin(9600);  
 attachInterrupt(digitalPinToInterrupt(BUTTON\_PIN), buttonInterrupt, CHANGE);  
  
}  
int buttonPressed = false;  
void buttonInterrupt()  
{  
 buttonPressed = true; *// Устанавливаем флаг нажатия кнопки*}  
void loop()  
{  
 if (buttonPressed && status == 0)  
 {  
 buttonPressed = false;  
 status = 1;  
 delay(300);  
 sing(1);  
 }  
 if (buttonPressed && status == 1)  
 {  
 buttonPressed = false;  
 status = 0;  
 delay(300);  
 sing(2);  
 }  
}  
  
int song = 0;  
void sing(int s)  
{  
 song = s;  
 if (song == 2)  
 {  
 Serial.println(" 'Underworld Theme'");  
 int size = sizeof(underworld\_melody) / sizeof(int);  
 for (int thisNote = 0; thisNote < size; thisNote++)  
 {  
 if (buttonPressed) break;  
 int noteDuration = 1000 / underworld\_tempo[thisNote];  
 buzz(melodyPin, underworld\_melody[thisNote], noteDuration, 1);  
 int pauseBetweenNotes = noteDuration \* 1.30;  
 delay(pauseBetweenNotes);  
 buzz(melodyPin, 0, noteDuration, 1);  
 }  
 }  
 else  
 {  
 Serial.println("'Mario Theme'");  
 int size = sizeof(melody) / sizeof(int);  
 for (int thisNote = 0; thisNote < size; thisNote++)  
 {  
 if (buttonPressed) break;  
 int noteDuration = 1000 / tempo[thisNote];  
 buzz(melodyPin, melody[thisNote], noteDuration, 0);  
 int pauseBetweenNotes = noteDuration \* 1.30;  
 delay(pauseBetweenNotes);  
 buzz(melodyPin, 0, noteDuration, 0);  
 }  
 }  
 delay(300);  
}  
  
void buzz(int targetPin, long frequency, long length, bool melody)  
{  
 if (melody) digitalWrite(11, HIGH);  
 else digitalWrite(12, HIGH);  
 digitalWrite(13, LOW);  
 long delayValue = 1000000 / frequency / 2; *// calculate the delay value between transitions* long numCycles = frequency \* length / 1000; *// calculate the number of cycles for proper timing* for (long i = 0; i < numCycles; i++)  
 { *// for the calculated length of time...* digitalWrite(targetPin, HIGH); *// write the buzzer pin high to push out the diaphram* delayMicroseconds(delayValue); *// wait for the calculated delay value* digitalWrite(targetPin, LOW); *// write the buzzer pin low to pull back the diaphram* delayMicroseconds(delayValue); *// wait again or the calculated delay value* }  
 if (melody) digitalWrite(11, LOW);  
 else digitalWrite(12, LOW);  
 digitalWrite(13, HIGH);  
}

Ссылка на проект

[Азбука морзе](https://www.tinkercad.com/things/gTQkgK3JfBU-21-azbuka-morzemetodichka/editel?sharecode=qq9H91jmxAgu18d1by94-Sxxk8EoZHxMy1U79HWnBoc)

2 Терменвокс

Индивидуальное задание представлено в таблице 2.

Таблица 1 – Индивидуальное задание

|  |  |  |
| --- | --- | --- |
| При падении освещенности звук... | Звуковой сигнал | Частота ноты |
| Уменьшается | Непрерывно | Малая октава |

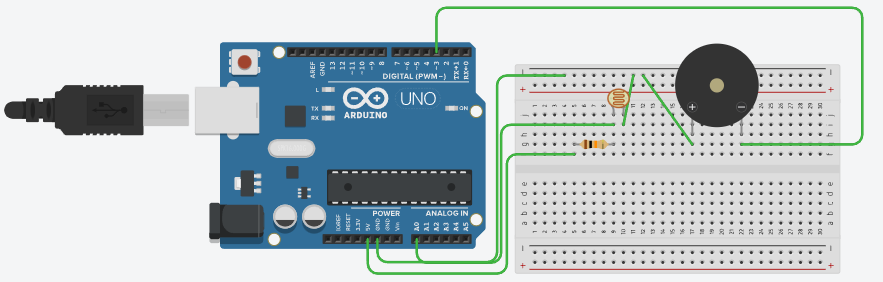
 Схема сборки терменвокса на макетной плате представлена на рисунке 2.1

Рисунок 2.1 – Схема сборки на макетной плате

Принципиальная схема представлена на рисунке 2.2

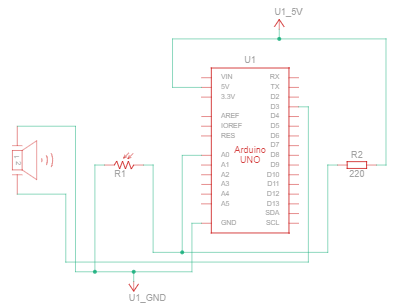


Рисунок 2.2 – Принципиальная схема

Листинг программы

#define BUZZER\_PIN 3  
 #define LDR\_PIN A0  
int octave[] = {131,139,148,156,165,175,185,196,207,220,233,247};  
 int val, frequency;  
void setup()  
{  
 pinMode(BUZZER\_PIN, OUTPUT);  
 Serial.begin(9600);  
}  
void loop()  
{  
 val = analogRead(LDR\_PIN);  
 frequency = map(val, 969, 49, 0, 11);  
 tone(BUZZER\_PIN, octave[frequency]);  
 Serial.println(val);  
}

Ссылка на проект

[Терменвокс](https://www.tinkercad.com/things/1F90VnxHV0H-prakt-22-laba/editel?returnTo=https%3A%2F%2Fwww.tinkercad.com%2Fdashboard%2Fdesigns%2Fcircuits%3Fpage%3D2)

3 Мерзкое пианино

Индивидуальное задание представлено в таблице 3.

Таблица 2 – Индивидуальное задание

|  |  |  |
| --- | --- | --- |
| Подключение кнопок | Входной сигнал | Частоты нот |
| Подтягивающий резистор | switch/case | Первая октава |

Схема сборки пульсара на макетной плате представлена на рисунке 3.1

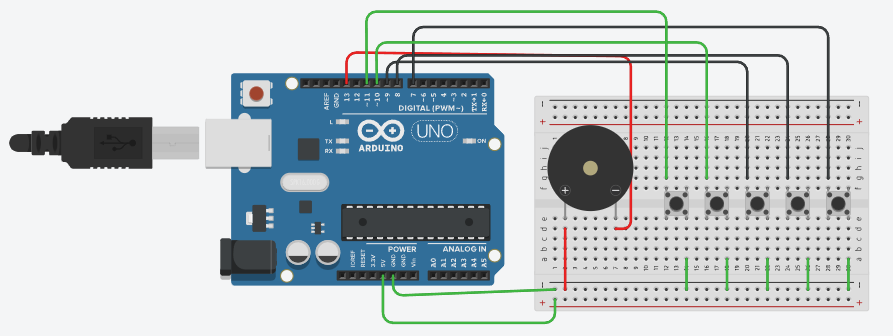


Рисунок 3.1 – Схема сборки на макетной плате

Принципиальная схема представлена на рисунке 3.2

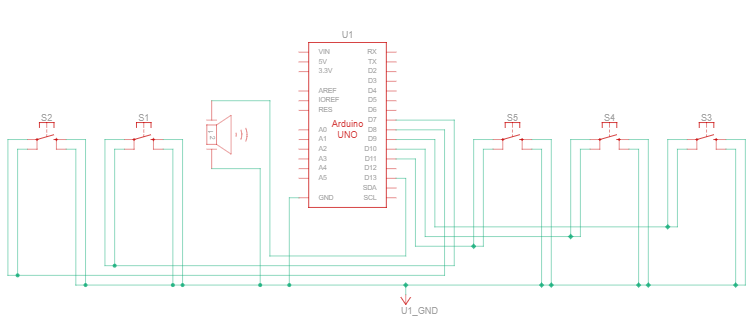


Рисунок 3.2 – Принципиальная схема

Листинг программы

int buttonPin[] = {7, 8, 9, 10, 11}; *// пины кнопок*int piezoPin = 13;  
int k = 0;  
void setup() {  
 for (int i = 0; i < 5; i++) {  
 pinMode(buttonPin[i], INPUT\_PULLUP);  
 }  
 pinMode(piezoPin, OUTPUT);  
}  
void loop() {  
 if (digitalRead(buttonPin[k]) == LOW) {  
 switch (k) {  
 case 0:  
 tone(piezoPin, 262);  
 break;  
 case 1:  
 tone(piezoPin, 294);  
 break;  
 case 2:  
 tone(piezoPin, 330);  
 break;  
 case 3:  
 tone(piezoPin, 349);  
 break;  
 case 4:  
 tone(piezoPin, 392);  
 break;  
 default:  
 noTone(piezoPin);  
 }  
 delay(100);  
 noTone(piezoPin);  
 }  
 k = (k + 1) % 5;  
}

Ссылка на проект

[Мерзкое пианино](https://www.tinkercad.com/things/aneNAEFkdlt-prakt-23-laba/editel?returnTo=https%3A%2F%2Fwww.tinkercad.com%2Fdashboard%2Fdesigns%2Fcircuits%3Fpage%3D1)

4 Перетягивание каната

Индивидуальное задание представлено в таблице 3

|  |  |  |
| --- | --- | --- |
| Подключение кнопок | Выигрыш | Нажатий для перехода на  следующий светодиод |
| Без резистора | Сигнализация на пьезодинамике | 4 |

Схема сборки пульсара на макетной плате представлена на рисунке 4.1

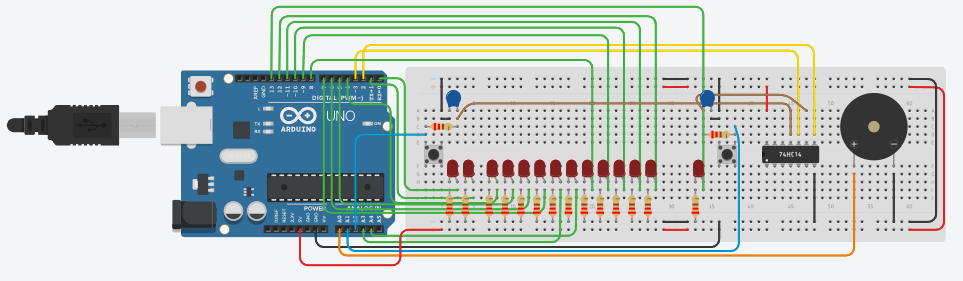


Рисунок 4.1 – Схема сборки на макетной плате

Принципиальная схема представлена на рисунке 4.2, 4.3

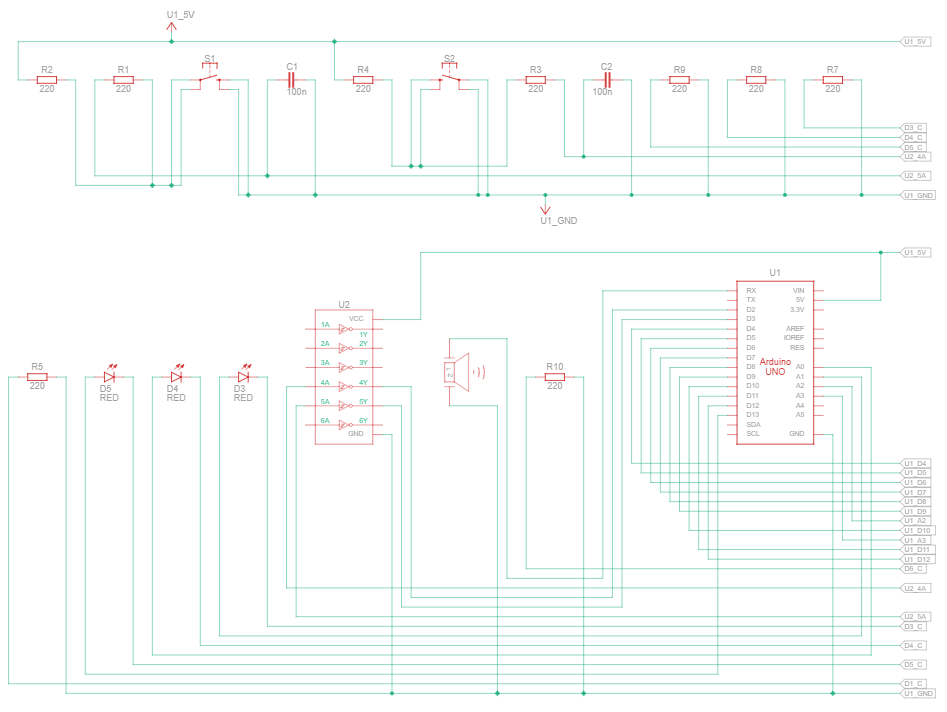


Рисунок 4.2 – Принципиальная схема (1)

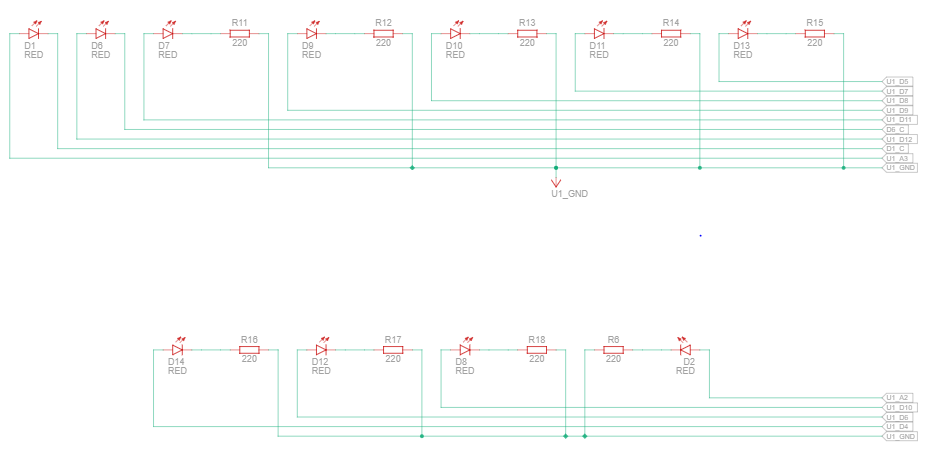


Рисунок 4.3 – Принципиальная схема (2)

Листинг программы

*// Определение пина для подключения пищалки*#define BUZZER\_PIN A0  
*// Максимальное значение очков*#define MAX\_SCORE 32  
*// Первый пин для светодиодов* #define FIRST\_LED\_PIN 0  
*// Последний пин для светодиодов* #define LAST\_LED\_PIN 13  
*// Общее количество светодиодов* #define LED\_COUNT 8  
*// Порог для отображения очков на светодиодах* #define SCORE\_THRESHOLD 4  
int flag = 1;  
*// Переменные для хранения очков команд*volatile int leftScore = 0;  
volatile int rightScore = 0;  
  
*// Пины для светодиодов команд*int leftLedPins[] = {0, A3, 7, 6, 5, 4, 1, 0};  
int rightLedPins[] = {0, A4, 8, 9, 10, 11, 12, 13};  
  
*// Настройка пинов при запуске*void setup() {  
 *// Настройка пинов для светодиодов* for (int i = FIRST\_LED\_PIN; i <= LAST\_LED\_PIN; i++) {  
 if (i == 2 || i == 3) continue; *// Пропуск пинов 2 и 3* pinMode(i, OUTPUT); *// Установка пинов для светодиодов как вывода* }  
 pinMode(A3, OUTPUT);  
 pinMode(A4, OUTPUT);  
 pinMode(BUZZER\_PIN, OUTPUT); *// Установка пина для пищалки как вывод* pinMode(A1, INPUT\_PULLUP); *// Установка пина A1 как входа с подтяжкой* pinMode(A2, INPUT\_PULLUP); *// Установка пина A2 как входа с подтяжкой* attachInterrupt(digitalPinToInterrupt(2), pushRight, FALLING); *// Назначение прерывания для кнопки команды справа* attachInterrupt(digitalPinToInterrupt(3), pushLeft, FALLING); *// Назначение прерывания для кнопки команды слева  
  
 // Включение срединных светодиодов* digitalWrite(A3, HIGH);  
 digitalWrite(A4, HIGH);  
 *//Serial.begin(9600);*}  
  
*// Прерывание при нажатии кнопки команды справа*void pushRight() {  
 if (flag == 1 && rightScore == 3) {  
 flag = 0;  
 rightScore = 8;  
 leftScore = -8;  
 } else {  
 rightScore++;  
 leftScore--;  
 }  
}  
*// Прерывание при нажатии кнопки команды слева*void pushLeft() {  
 if (flag == 1 && leftScore == 3) {  
 flag = 0;  
 leftScore = 8;  
 rightScore = -8;  
 } else {  
 leftScore++;  
 rightScore--;  
 }  
}  
  
*// Бесконечный цикл выполнения*void loop() {  
 tone(BUZZER\_PIN, 2000, 100); *// Проигрывание звука на пищалке  
 // Цикл, пока значение абсолютного очка команды слева и справа не превысит максимальное значение* while ((abs(leftScore) < MAX\_SCORE) && (abs(rightScore) < MAX\_SCORE)) {  
 *// Обновление состояния светодиодов для левой команды* if (abs(leftScore) % SCORE\_THRESHOLD == 0) {  
 int bound = leftScore / SCORE\_THRESHOLD; *// Вычисление, сколько светодиодов должны быть включены* for (int i = 2; i < LED\_COUNT; i++) { *// Начинаем с индекса 2, чтобы пропустить A3* digitalWrite(leftLedPins[i], i <= bound); *// Включение светодиодов в зависимости от bound* }  
 }  
 *// Обновление состояния светодиодов для правой команды* if (abs(rightScore) % SCORE\_THRESHOLD == 0) {  
 int bound = rightScore / SCORE\_THRESHOLD; *// Вычисление, сколько светодиодов должны быть включены* for (int i = 2; i < LED\_COUNT; i++) { *// Начинаем с индекса 2, чтобы пропустить A4* digitalWrite(rightLedPins[i], i <= bound);*// Включение светодиодов в зависимости от bound* }  
 }  
 *// Прерывание цикла, если значение очка команды слева или справа достигло максимального значения* if (leftScore >= MAX\_SCORE) {  
 break;  
 }else if (rightScore >= MAX\_SCORE){  
 break;  
 }  
 }  
 *// Проигрывание звука на пищалке* tone(BUZZER\_PIN, 2000, 200);  
 *// Зацикливание программы для прекращения работы* while (true) {}  
}

Ссылка на проект

[Перетягивание каната](https://www.tinkercad.com/things/2p4WYoTct6L-peretyagivanie-25/editel?returnTo=https%3A%2F%2Fwww.tinkercad.com%2Fdashboard%2Fdesigns%2Fcircuits%3Fpage%3D1)

Вывод

В ходе лабораторной работы познакомились с такими элементами как (конденсатор, пьезодинамик) и такими командами как (attachInterrupt(), tone()).