

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет Информатика и вычислительная техника

Кафедра Кибербезопасность информационных систем

**ОТЧЕТ ПО ЛАБОРАТОРНЫМ РАБОТАМ**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Автор |  | | |  | Ковалев Д.П. | | | |
|  | (подпись, дата) | | |  |  | | | |
| Обозначение | ИиВТ.10.05.01.13 | | Группа | | | | ВКБ32 |  |
| Направление подготовки | | 10.05.01 Компьютерная безопасность | | | | | |  |
| Профиль | Математические методы защиты информации | | | | | | |  |
| Проверил |  | | |  | |  | |  |
|  | (подпись, дата) | | |  | |  | |  |

Ростов-на-Дону

2024

Содержание

[Лабораторная работа №1 3](#_Toc159059439)

[Лабораторная работа №2 4](#_Toc159059440)

[Лабораторная работа №3 8](#_Toc159059441)

[Лабораторная работа №4 10](#_Toc159059442)

[Лабораторная работа №5 12](#_Toc159059443)

[Лабораторная работа №6 14](#_Toc159059444)

[Лабораторная работа №7 16](#_Toc159059445)

# 17

# **Лабораторная работа №1**

**Тема:** «Изучения создания сети в Cisco Packet Tracer»

**Цель:** научиться создавать простые сетевые соединения.

**Задание 1**. Создание сети из двух ПК в программе Cisco Parket Tracer.

Нажмем на кнопку “End devices”, где находятся персональные компьютеры. На рисунке 1 также видно знак молнии, чтобы подключить наши 2 компьютера. Во вкладке “Connections” есть медные провода с обжимом.

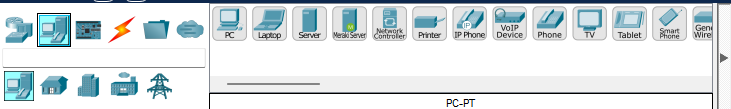


Рисунок 1 – интерфейс программы с Drag & Drop

В результате проделанных операций получилось подключение, которое представлено на рисунке 2.

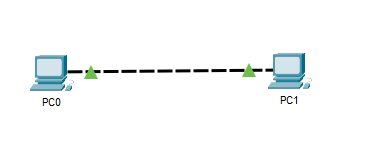


Рисунок 2 – физическое подключение 2 ПК

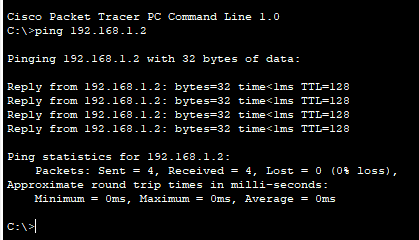
 Далее нам нужно настроить IP адрес и маску для каждого ПК, как было показано в методическом материале. Теперь можно приступить к проверке видимости другого ПК в локальной сети с помощью команды “ping 192.168.1.2”. На рисунке 3 представлен результат.

Рисунок 3 – результат команды ping

**Задание 2.** Организация Режим симуляции работы сети.

Нужно задать в рабочем пространстве программы сеть из 4х ПК и 2х хабов. Результат подключения представлен на рисунке 4.

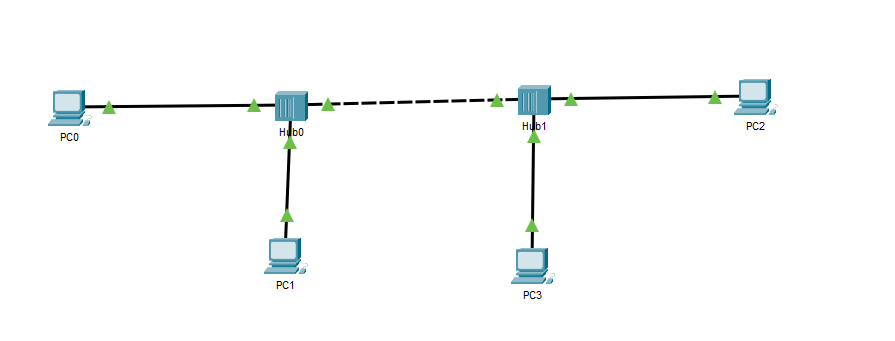


Рисунок 4 – результат подключения устройств

Далее я отключил все протоколы, оставив только ICMP (Internet Control Message Protocol) — сетевой протокол, входящий в стек протоколов TCP/IP. В основном ICMP используется для передачи сообщений об ошибках и других исключительных ситуациях, возникших при передаче данных. Фильтры были отключены, как представлено на рисунке 5.

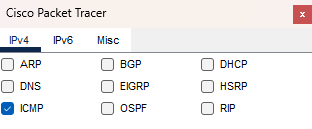


Рисунок 5 – фильтры, которые я использовал

Теперь создадим пакет ICMP, который будет передаваться от 1 ПК к другому, здесь вот создал я пакет, который будет в дальнейшем отправлен, данный конверт представлен на рисунке 6.

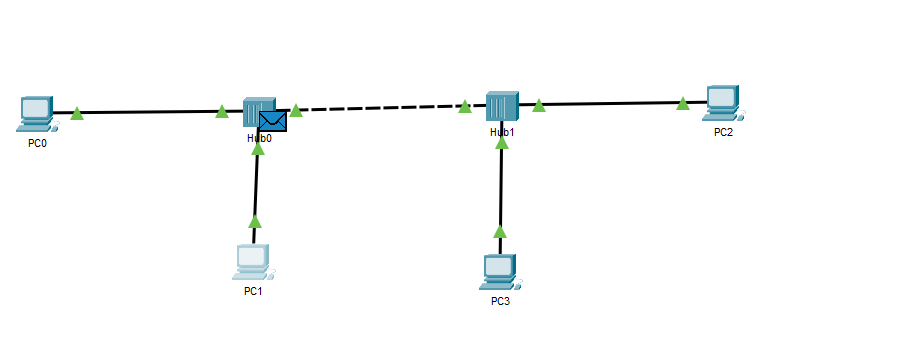


Рисунок 6 – процесс отправки пакетов с 1 ПК на другой

Чтобы отображалось в списке событий, нужно нажать на английскую букву “P” или нажать на иконку “письмо”. Дальше остается только связать два элемента. В результате должно появиться, как на рисунке 7.



Рисунок 7 – список событий после прогона пакета

**Задание 3.** Настройка сетевых параметров ПК в его графическом интерфейсе.

Настроить IP адрес и маску подсети можно, используя несколько вариантов. Первый вариант – использование GUI интерфейса, другой вариант – это использование командной строки. В нашем случае подойдет команда `ipconfig 192.168.0.1 255.255.255.0`, где первый аргумент – это IP, а второй 0 маска. Также с помощью данной команды можно проверять настройки. Ipconfig для 4 ПК представлен на рисунке 8.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 8 – ipconfig для 4 ПК

# **Лабораторная работа №2**

**Тема:** «Мигание светодиодом»

**Цель:**

1. Запрограммировать плату так, чтобы она миганием светодиода передавала сигнал SOS;
2. Включить в цепь три светодиода и обозначьте их цвета так, чтобы они симулировали работу светофора.

На рисунке 2 представлена реализация лабораторной работы. Для уменьшения напряжения и увеличения продолжительности работы светодиода необходимо повысить сопротивление при помощи резистора на 220 Ом. К минусу, соединённому с землёй, мы присоединяем катод, а к плюсу, который идёт к тринадцатому пину, анод.

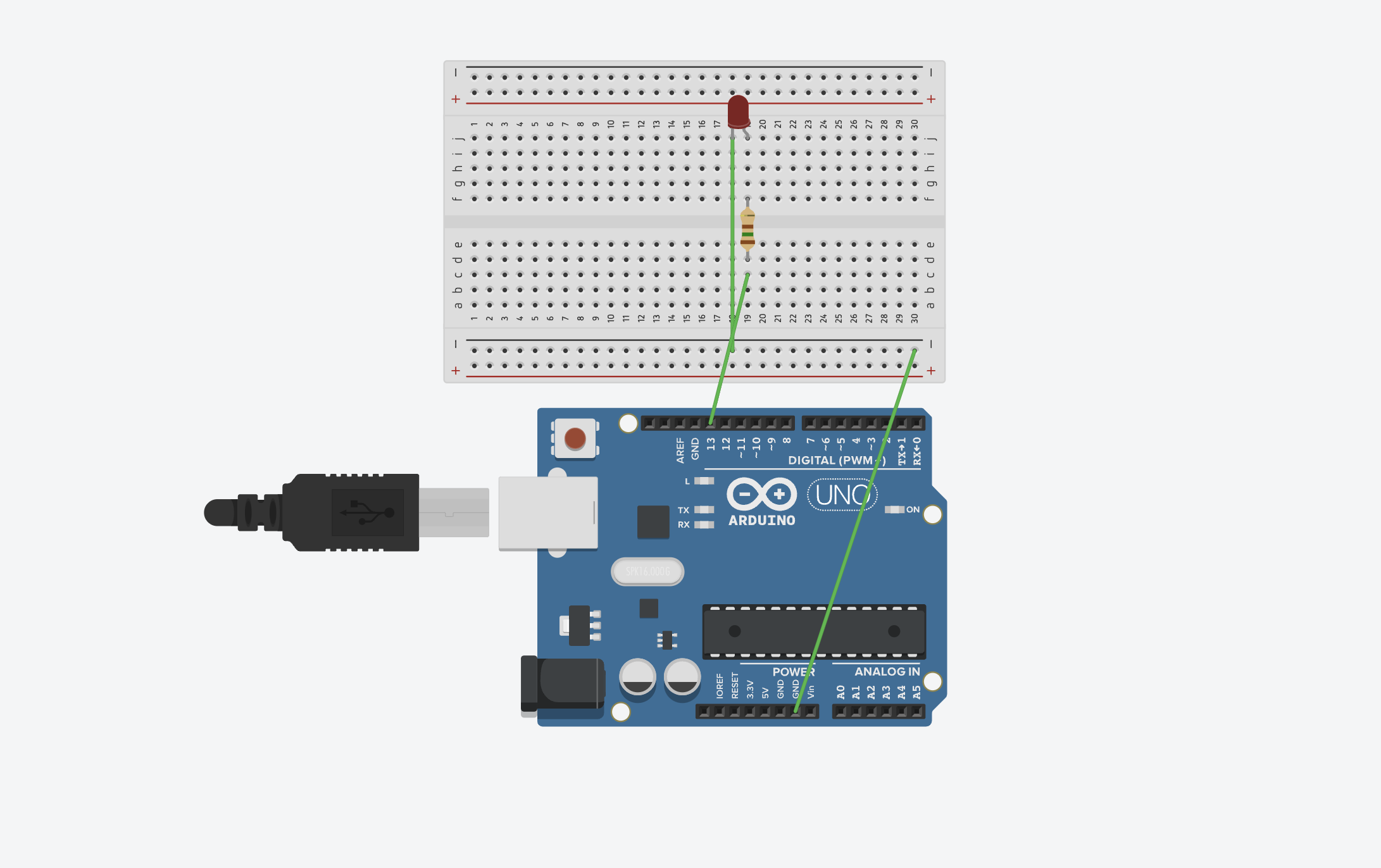


Рисунок 2 – Реализация задания 1

Программный код:

// C++ code

int counter = 0;

int counter2;

int counter3;

int counter4;

void setup()

{

pinMode(LED\_BUILTIN, OUTPUT);

}

void loop()

{

for (counter2 = 0; counter2 < 3; ++counter2) {

digitalWrite(LED\_BUILTIN, HIGH);

delay(200); // Wait for 200 millisecond(s)

digitalWrite(LED\_BUILTIN, LOW);

delay(200); // Wait for 200 millisecond(s)

}

for (counter3 = 0; counter3 < 3; ++counter3) {

digitalWrite(LED\_BUILTIN, HIGH);

delay(400); // Wait for 400 millisecond(s)

digitalWrite(LED\_BUILTIN, LOW);

delay(200); // Wait for 200 millisecond(s)

}

for (counter4 = 0; counter4 < 3; ++counter4) {

digitalWrite(LED\_BUILTIN, HIGH);

delay(200); // Wait for 200 millisecond(s)

digitalWrite(LED\_BUILTIN, LOW);

delay(200); // Wait for 200 millisecond(s)

}

}

На рисунке 3 представлена реализация задания. На элементе breadboard закрепляется три светодиода цветов светофора и к ним подключается три резистора на 150-220 Ом. Каждый светодиод подключается к минусу (GND) и к присвоенному пину, для управления светодиодом через плату.

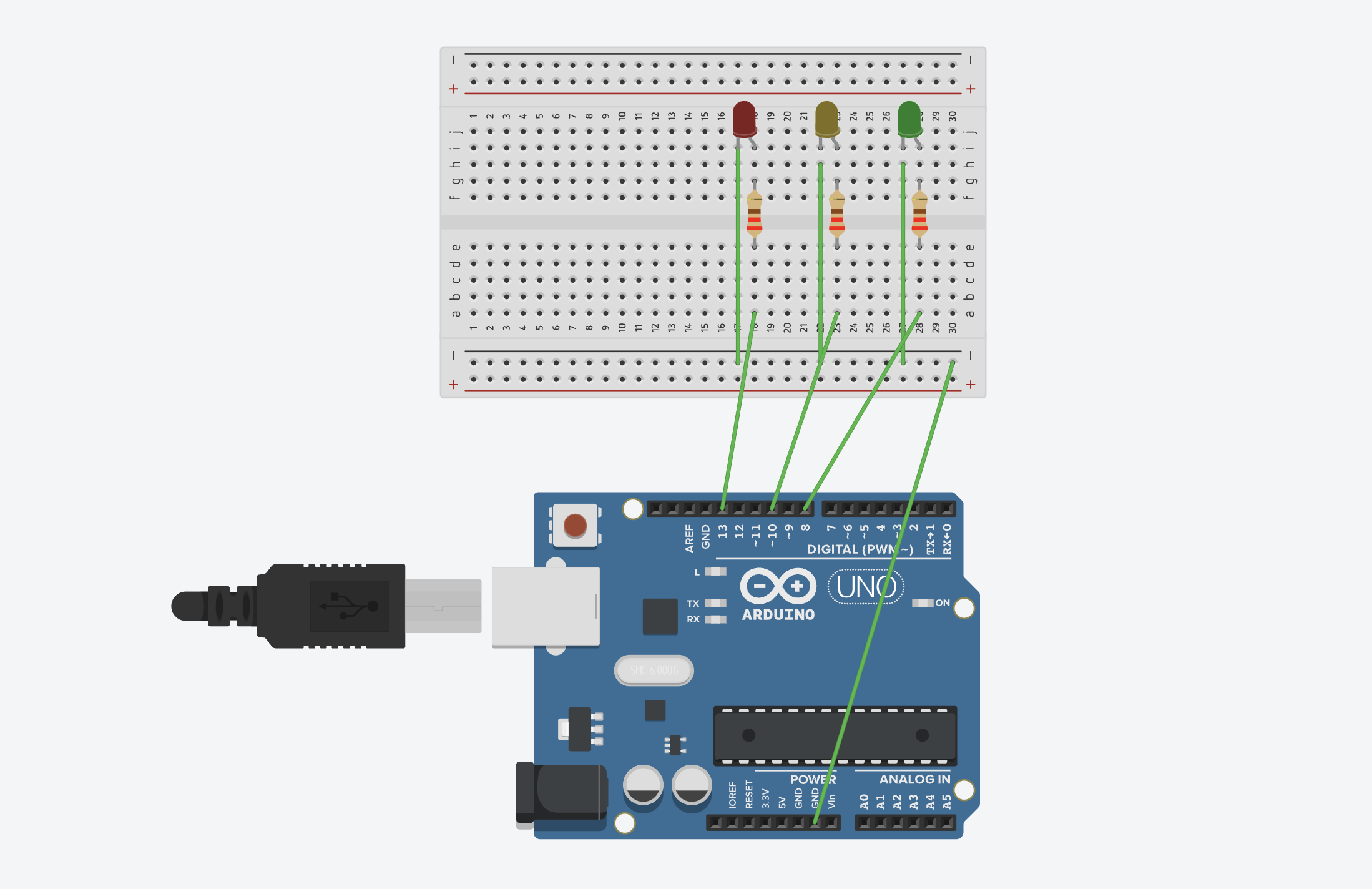


Рисунок 3 – Реализация задания 2

Программный код:

// C++ code

int counter;

int counter2;

int counter3;

int counter4;

void setup()

{

pinMode(13, OUTPUT);

pinMode(10, OUTPUT);

pinMode(8, OUTPUT);

}

void loop()

{

digitalWrite(13, HIGH);

delay(8000); // Wait for 8000 millisecond(s)

for (counter = 0; counter < 3; ++counter) {

digitalWrite(13, HIGH);

delay(3000); // Wait for 3000 millisecond(s)

digitalWrite(13, LOW);

delay(1000); // Wait for 1000 millisecond(s)

}

for (counter2 = 0; counter2 < 3; ++counter2) {

digitalWrite(13, HIGH);

delay(1000); // Wait for 1000 millisecond(s)

digitalWrite(13, LOW);

delay(1000); // Wait for 1000 millisecond(s)

}

digitalWrite(10, HIGH);

delay(4000); // Wait for 4000 millisecond(s)

digitalWrite(10, LOW);

digitalWrite(8, HIGH);

delay(8000); // Wait for 8000 millisecond(s)

for (counter3 = 0; counter3 < 3; ++counter3) {

digitalWrite(8, HIGH);

delay(5000); // Wait for 5000 millisecond(s)

digitalWrite(8, LOW);

delay(1000); // Wait for 1000 millisecond(s)

}

for (counter4 = 0; counter4 < 3; ++counter4) {

digitalWrite(8, HIGH);

delay(1000); // Wait for 1000 millisecond(s)

digitalWrite(8, LOW);

delay(1000); // Wait for 1000 millisecond(s)

}

}

# **Лабораторная работа №3**

**Тема:** «Звуковые сигналы»

**Цель:** Используя два пьезоэлемента, запрограммируйте плату на воспроизведение сигнала SOS в нескольких разных тональностях.

Сигнал SOS из пьезоэлементов. На рисунке 4 представлена реализация с использованием двух резисторов и двух пъезоэлементов, соответственно подключенных к плюсам и к пинам.

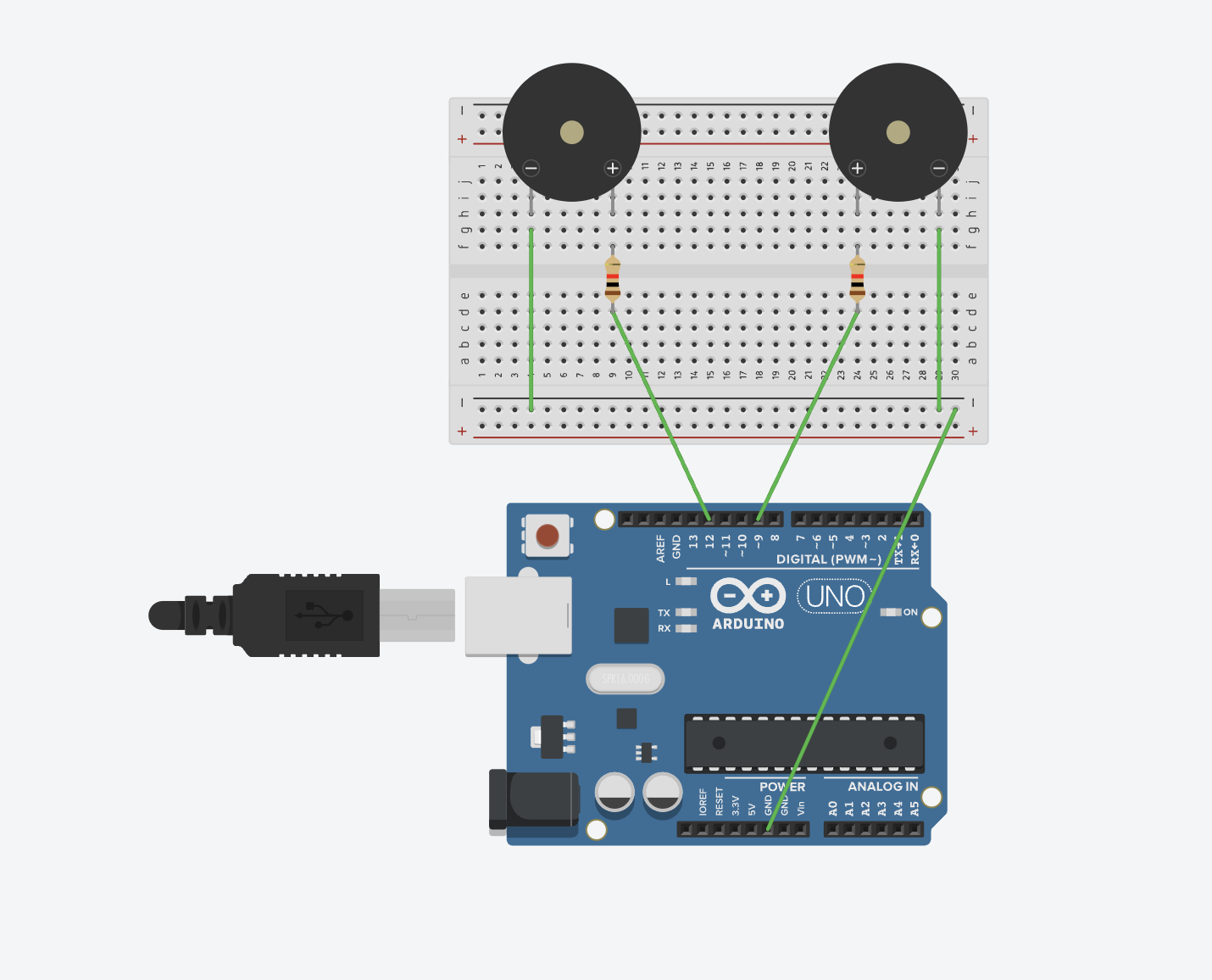
****

Рисунок 4 – Реализация сигнала SOS с помощью пьезоэлементов

Программный код:

int counter;

void setup()

{

pinMode(12, OUTPUT);

pinMode(9, OUTPUT);

}

void loop()

{

for (counter = 0; counter < 3; ++counter) {

tone(12, 1000, 120); // play tone 70 (A#5 = 932 Hz)

delay(500);

}

for (counter = 0; counter < 3; ++counter) {

tone(12, 500, 120); // play tone 70 (A#5 = 932 Hz)

delay(250);

}

delay(250);

for (counter = 0; counter < 3; ++counter) {

tone(12, 1000, 120); // play tone 70 (A#5 = 932 Hz)

delay(500);

}

delay(2000); // Wait for 2000 millisecond(s)

for (counter = 0; counter < 3; ++counter) {

tone(9, 300, 120); // play tone 70 (A#5 = 932 Hz)

delay(500);

}

for (counter = 0; counter < 3; ++counter) {

tone(9, 2000, 120); // play tone 70 (A#5 = 932 Hz)

delay(250);

}

delay(250);

for (counter = 0; counter < 3; ++counter) {

tone(9, 300, 120); // play tone 70 (A#5 = 932 Hz)

delay(500);

}

delay(2000); // Wait for 2000 millisecond(s)

}

# **Лабораторная работа №4**

**Тема:** «Работа с кнопками»

**Цель:** При нажатии кнопки, запрограммировать плату на воспроизведение короткого фрагмента, выбранной Вами музыкальной композиции (количество пьезоэлементов выбирайте самостоятельно).

На рисунке 5 показана реализация музыкального фрагментам при помощи пьезоэлемента, где правая кнопка отвечает за запуск, а левая за остановку. Резисторы подключаются к кнопкам для понимажения напряжения и увеличения сопротивления.

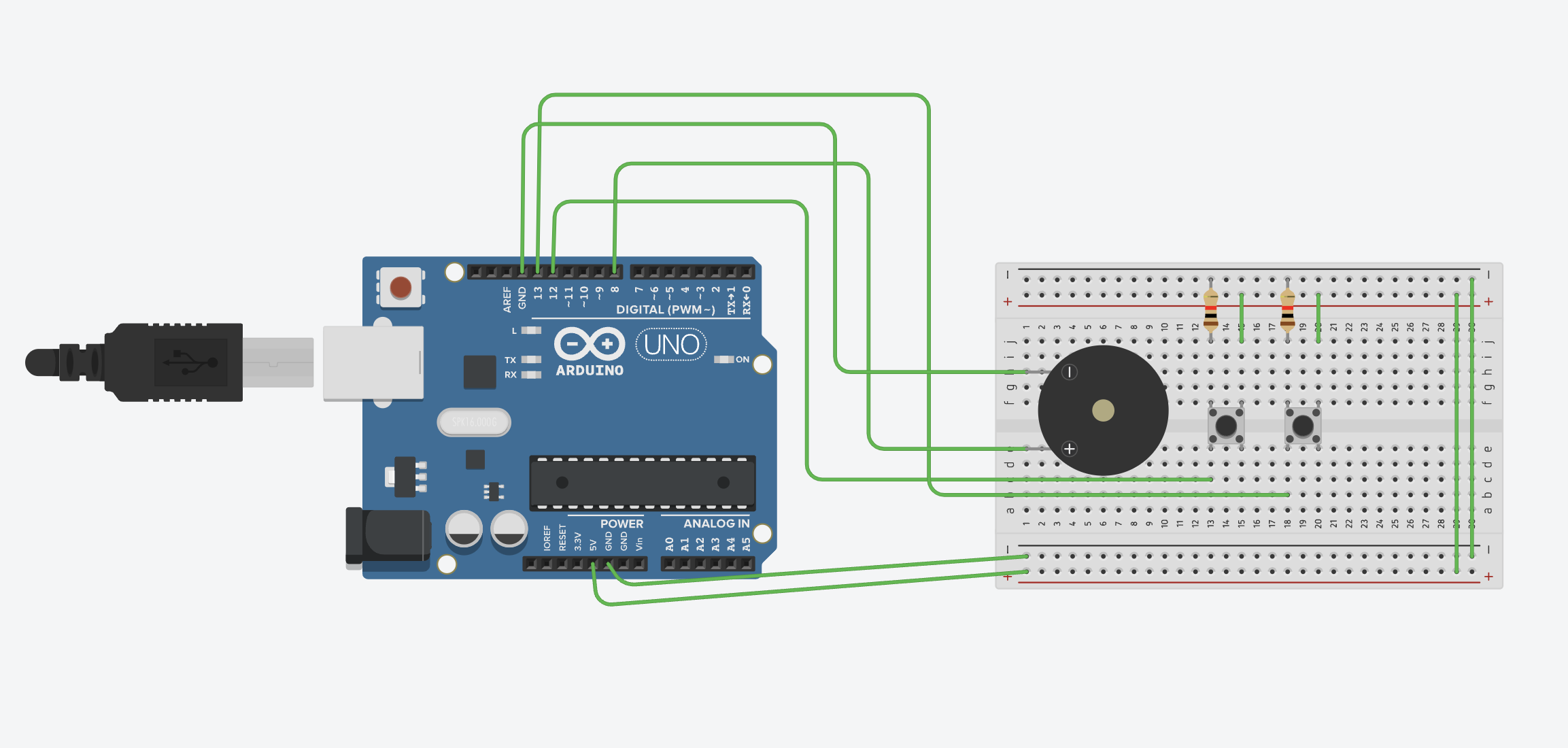
****

Рисунок 5 – Реализация задания с пьезоэлементами и кнопками

Программный код:

// C++ code

//

int playButtonPin = 13;

int stopButtonPin = 12;

int piezoPin = 8;

int notes[] = {262, 262, 294, 262, 349, 330, 262, 262, 294, 262, 392, 349, 262, 262, 523, 440, 349, 330, 294, 466, 466, 440, 349, 392, 349};

int duration[] = {4, 4, 8, 8, 8, 4, 4, 8, 8,ч 8, 4, 4, 8, 8, 4, 4, 8, 8, 8, 4, 4, 8, 8, 8, 4};

void setup() {

pinMode(playButtonPin, INPUT);

pinMode(stopButtonPin, INPUT);

pinMode(piezoPin, OUTPUT);

}

void loop() {

if (digitalRead(playButtonPin) == HIGH) {

for (int i = 0; i < sizeof(notes) / sizeof(notes[0]); i++) {

tone(piezoPin, notes[i], duration[i] \* 100);

delay(duration[i] \* 100);

if (digitalRead(stopButtonPin) == HIGH) {

noTone(piezoPin);

break;

}

}

}

}

# **Лабораторная работа №5**

**Тема:** «Работа с ЖК-дисплеем»

**Цель:** Подсоединить к плате несколько светодиодов, на экран выводить номер и цвет светодиода.

Для работы с ЖК-дисплеем в инструментах Tinkercad имеется встроенная плата. На рисунке 6 к готовой плате подключается 3 резистора к соответствующим трём светодиодам цветов светофора.

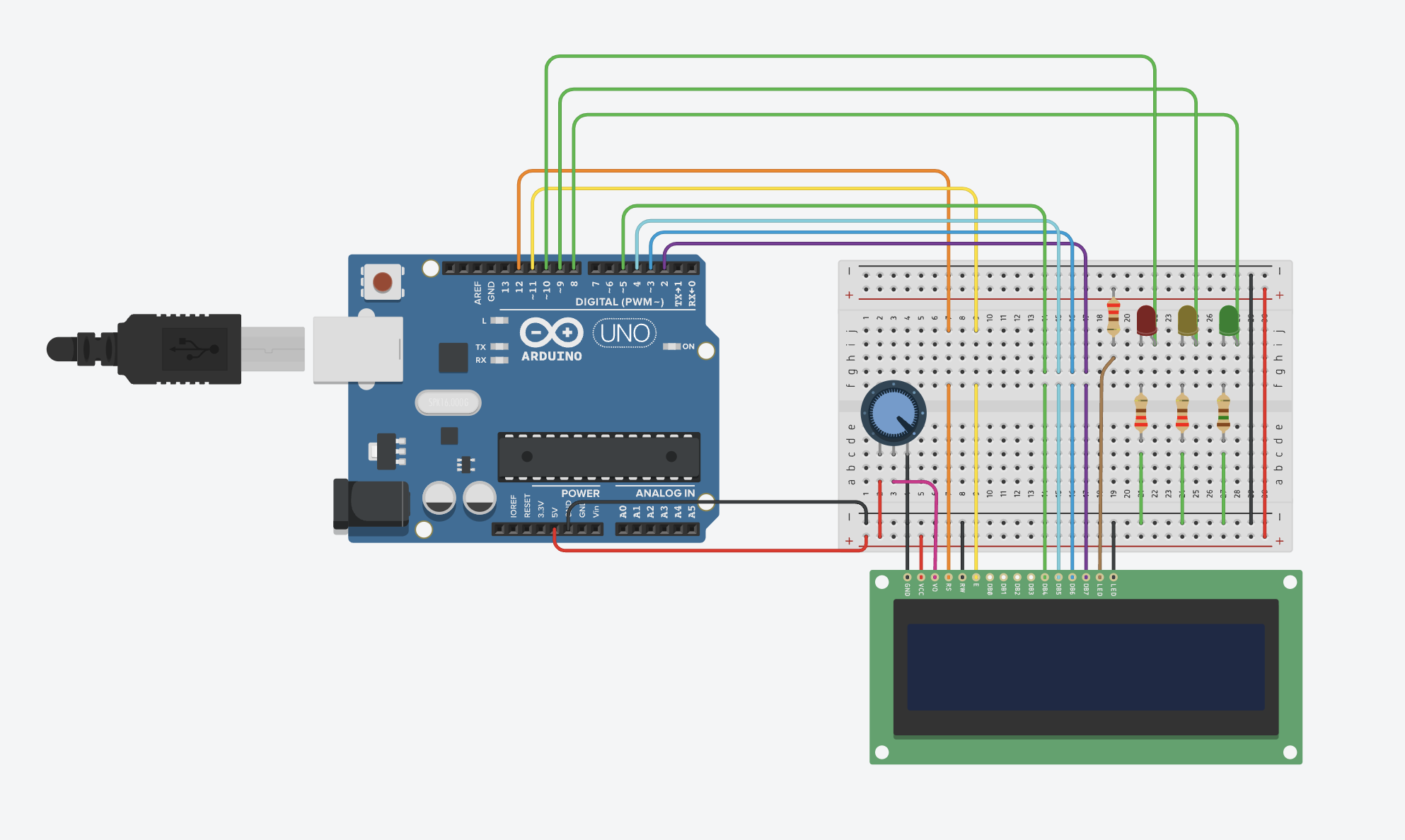


Рисунок 6 – Реализация задания с ЖК-дисплеем

Программный код:

// C++ code

//

#include <LiquidCrystal.h>

int led1 = 8;

int led2 = 9;

int led3 = 10;

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup(){

lcd.begin(0, 15);

pinMode(led1, OUTPUT);

pinMode(led2, OUTPUT);

pinMode(led3,OUTPUT);

}

void loop(){

digitalWrite(led1, HIGH);

digitalRead(led1) == HIGH;

lcd.setCursor(8, 1);

lcd.print("8 Green");

delay(3000);

digitalRead(led1) == HIGH;

lcd.clear();

digitalWrite(led1, LOW);

delay(1000);

digitalWrite(led2, HIGH);

digitalRead(led2) == HIGH;

lcd.setCursor(0, 1);

lcd.print("9 Yellow");

delay(3000);

digitalRead(led2) == HIGH;

lcd.clear();

digitalWrite(led2, LOW);

delay(1000);

digitalWrite(led3, HIGH);

digitalRead(led3) == HIGH;

lcd.setCursor(7, 0);

lcd.print("10 Red");

delay(3000);

digitalRead(led3) == HIGH;

lcd.clear();

digitalWrite(led3, LOW);

delay(1000);

}

# **Лабораторная работа №6**

**Тема:** «Изучение кода»

**Цель:** Написать код, задающий работу железнодорожного светофора.

На рисунке 7 представлена реализация железнодорожного светофора с помощью трёх светодиодов, четырёх резисторов и одного пьезоэлемента.

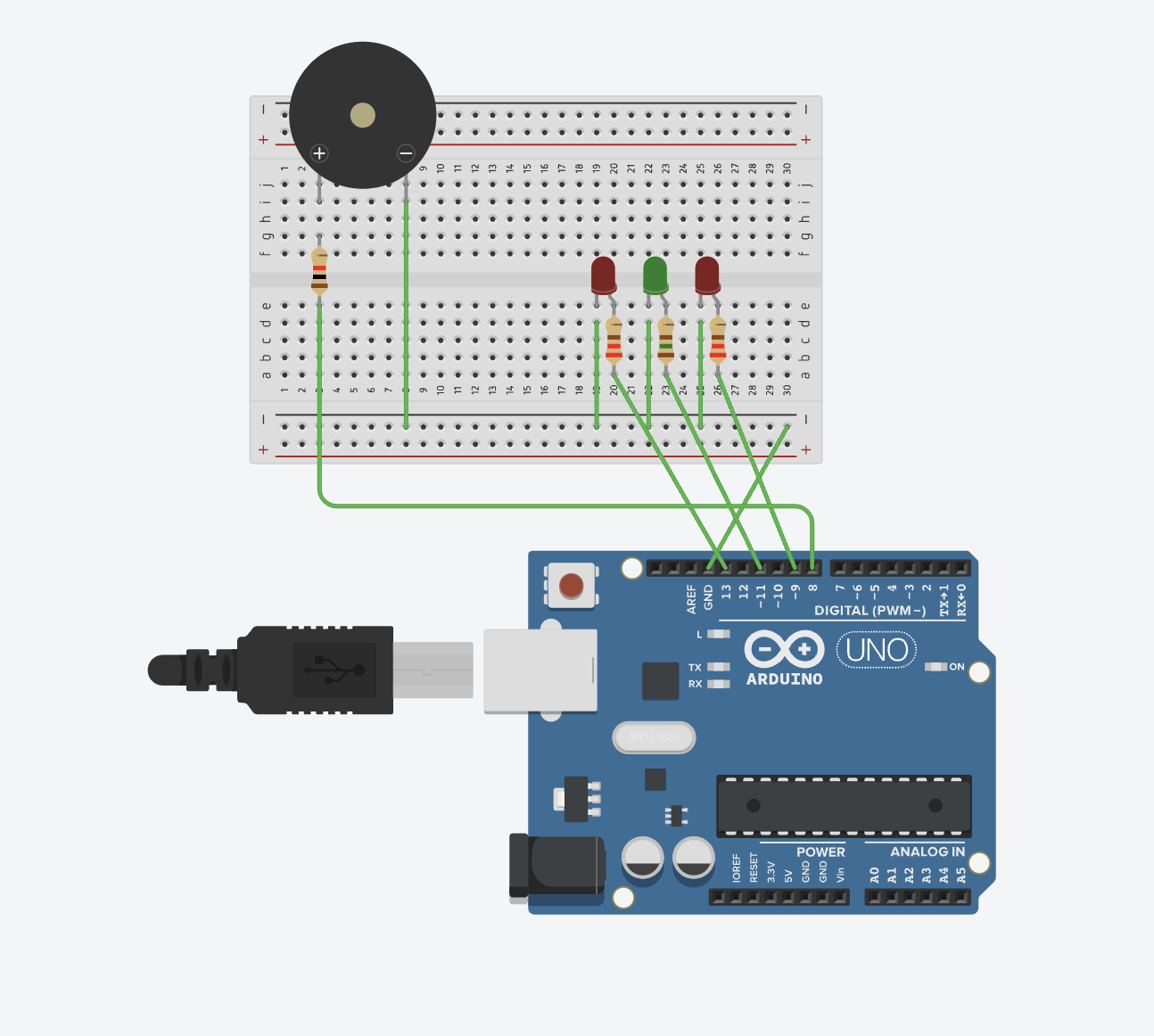


Рисунок 7 – Реализация железодорожного светофора

Программный код:

// C++ code

//

int counter;

void setup()

{

pinMode(8, OUTPUT);

pinMode(13, OUTPUT);

pinMode(9, OUTPUT);

pinMode(11, OUTPUT);

}

void loop()

{

for (counter = 0; counter < 5; ++counter) {

tone(8, 450, 500);

digitalWrite(13, HIGH);

delay(500);

tone(8, 550, 500);

digitalWrite(13, LOW);

delay(500);

tone(8, 500, 500);

digitalWrite(9, HIGH);

delay(500);

tone(8, 600, 500);

digitalWrite(9, LOW);

delay(500);

noTone(8);

}

digitalWrite(11, HIGH);

delay(5000);

digitalWrite(11, LOW);

}

# **Лабораторная работа №7**

**Тема:** «Самостоятельное кодирование платы»

**Цель:** Запрограммировать плату так, чтобы начальная температура была 40 градусов (при этом не горит ни один светодиод), потом при поднятии температуры на каждые 10 градусов зажигался один светодиод.  
(50 град.- один, 60 град.- два, 70 град.- все три светодиода).

Используя термодатчик и три светодиода на рисунке 3 показан способ реализации задания.

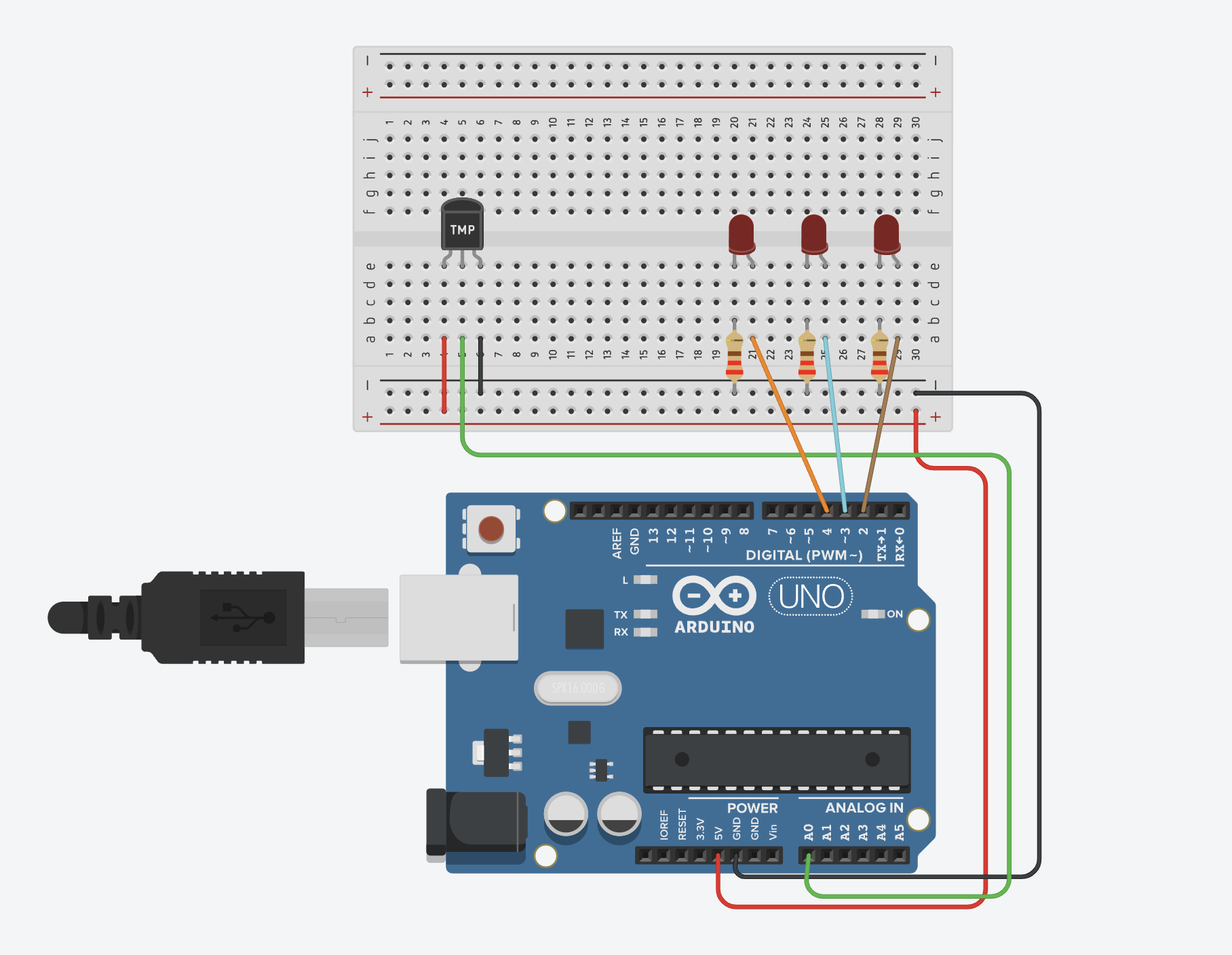


Рисунок 8 – Реализация с помощью термодатчика

Программный код:

// C++ code

//

float tempC;

void setup()

{

pinMode(A0, INPUT);

pinMode(2, OUTPUT);

pinMode(3, OUTPUT);

pinMode(4, OUTPUT);

}

void loop()

{

tempC = map(((analogRead(A0) - 20) \* 3.04), 0, 1023, -40, 125);

if (tempC > 40) {

digitalWrite(2, LOW);

digitalWrite(3, LOW);

digitalWrite(4, LOW);

}

if (tempC > 50) {

digitalWrite(2, HIGH);

digitalWrite(3, LOW);

digitalWrite(4, LOW);

}

if (tempC > 60) {

digitalWrite(2, HIGH);

digitalWrite(3, HIGH);

digitalWrite(4, LOW);

}

if (tempC > 70) {

digitalWrite(2, HIGH);

digitalWrite(3, HIGH);

digitalWrite(4, HIGH);

}

if (tempC >= 70) {

digitalWrite(2, HIGH);

digitalWrite(3, HIGH);

digitalWrite(4, HIGH);

}

delay(1000);

}