Практическое занятие №
Тема: «Использование светодиодов с ШИМ-контроллером и кнопками. Подключение RGB-светодиода»

Цель работы: приобрести практические навыки по подключению и программированию светодиодов, RGB-светодиодов с помощью ШИМ-контроллера, кнопок и потенциометра.

Последовательность выполнения работы:

– Собрать схемы на макетной плате, иначе при отсутствии набора Arduino в web-приложениях (https://wokwi.com/projects/new/arduino-uno или https://www.tinkercad.com/) для приведенных примеров.

– Запрограммировать микроконтроллер согласно заданию в примере.

Содержание отчета:

– название практического занятия, его цель;

– фото или скриншоты собранной схемы;

– написанный программный код вставить текстом, Courier New, 12 кегль, одинарный отступ без абзацев;

– вывод о проделанной работе;

– файл Fritzing с принципиальной и монтажной схемой.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

ПОДКЛЮЧЕНИЕ СВЕТОДИОДОВ

Светодиоды — это полупроводниковые элементы, которые служат для индикации и освещения. Они имеют полярность (+ и —) и чувствуют направление движения постоянного тока. Если подключить светодиод неправильно, то постоянный ток не пройдет и прибор не засветится. Кроме того, светодиод может выйти из строя при неправильном подключении. Анод (длинная ножка светодиода) подключается к плюсу.

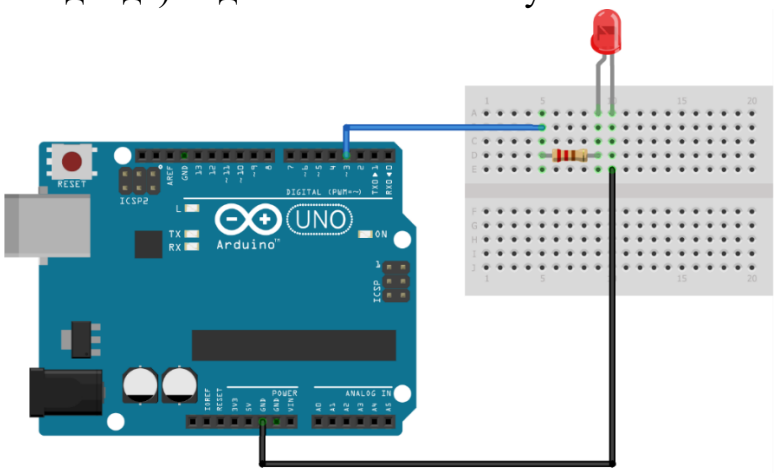

Рисунок 1 – Светодиод

1.1 Управление светодиодом с помощью ШИМ Ардуино

На некоторых МК есть ЦАП - цифро-аналоговый преобразователь (DAC, Digital-to-Analog Converter), он преобразует численное значение из программы в соответствующее напряжение в Вольтах и выводит его на пин. Простейший ЦАП можно собрать из десятка резисторов - R2R ЦАП, рассмотрим его в отдельном уроке.

Для большинства задач ЦАП на самом деле и не нужен - его заменяет ШИМ сигнал. В Arduino UNO ШИМ можно реализовать на цифровых выводах, обозначенных «PWM» или «~». На плате Arduino Uno это 3, 5, 6, 10, 11. Скважность импульса задается в пределах от 0 до 255.

Если светодиод половину времени будет включён, а половину выключен, то визуально будет казаться, что он светится в половину своей яркости. Это называется широтно-импульсная модуляция (ШИМ или PWM по-английски). С помощью ШИМ можно управлять «аналоговым» компонентом с помощью цифрового кода.

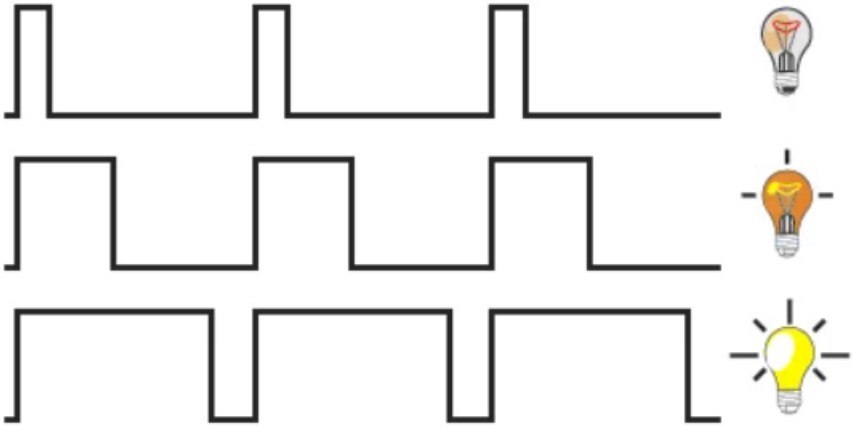

Рисунок 2 – Демонстрация ШИМ

Для реализации ШИМ применяется команда analogWrite имеющий следующий синтаксис:

```
analogWrite(вывод, значение);
```

вывод - номер вывода ШИМ,

значение - значение ШИМ (0-255), применительно к светодиодам это будет их яркость свечения.

0 - всегда выключен, ***255*** - всегда включен.

ЗАДАНИЯ

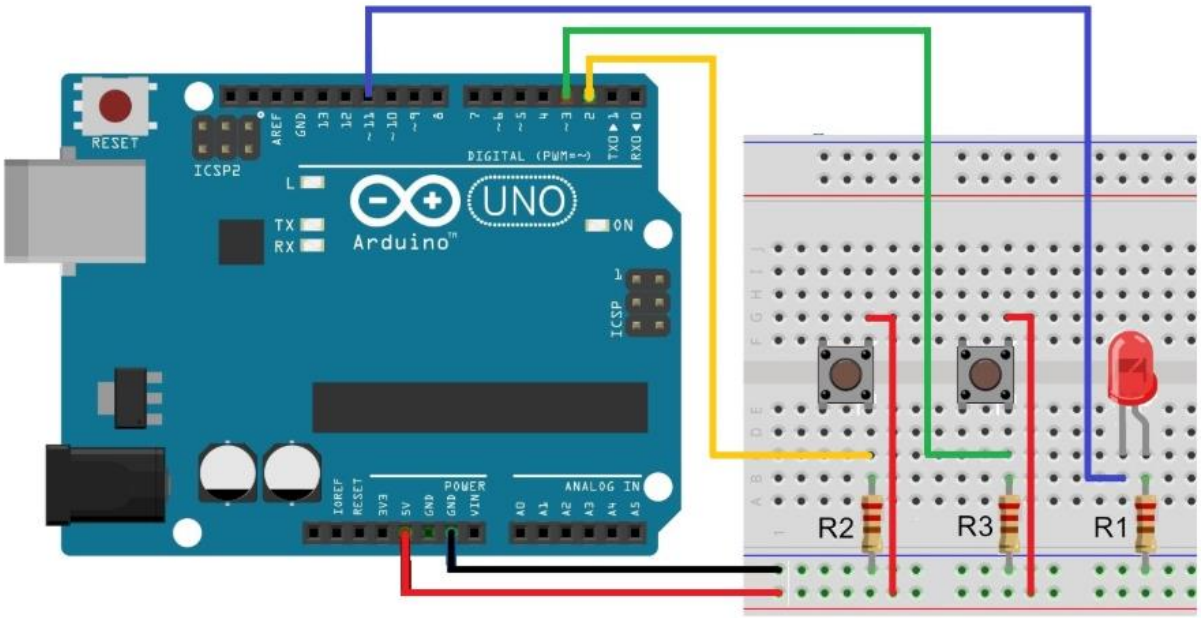

Рисунок 3 – Схема подключения к программе 1

R1, R2 = 10 кОМ, R3 =220-560 Ом

Программа 1:

```
int led = 11;      // Номер вывода к которому подключен светодиод
int brightness = 0;// Переменная в которой хранится уровень яркости (От 0 до 254)
int fadeValue = 5;// шаг изменения яркости

int buttonPlus = 2; // Номер вывода к которому подключена кнопка "+"
int buttonMinus = 3;// Номер вывода к которому подключена кнопка "-"

void setup()  {
  pinMode(led, OUTPUT); // Порт 11 (led) будет работать как Выход.
  pinMode(buttonPlus, INPUT);
  pinMode(buttonMinus, INPUT)
}

void loop()  { //Цикл будет выполняться бесконечное количество раз
  if (digitalRead(buttonPlus) == HIGH) {//если вход 2 имеет состояние 1(кнопка "+" нажата)
```

```
    brightness += fadeValue; // Увеличиваем значение
переменной яркости на fadeValue единиц
  }
  if (digitalRead(buttonMinus) == HIGH) {//если вход 3 имеет
состояние 1 (кнопка "-" нажата)
    brightness -= fadeValue;//Уменьшаем значение переменной
яркости на fadeValue единиц
  }
  brightness = constrain(brightness, 0, 254); //Эта функция
контролирует, чтобы
  //переменная brightness не стала больше 254 и меньше 0,
  //если значение выходит за границу то функция 0 или
254
  analogWrite(led, brightness);    // Устанавливаем
состояние яркости для светодиода
  delay(50); // Пауза 50 миллисекунд.
}
```

Программа 2:

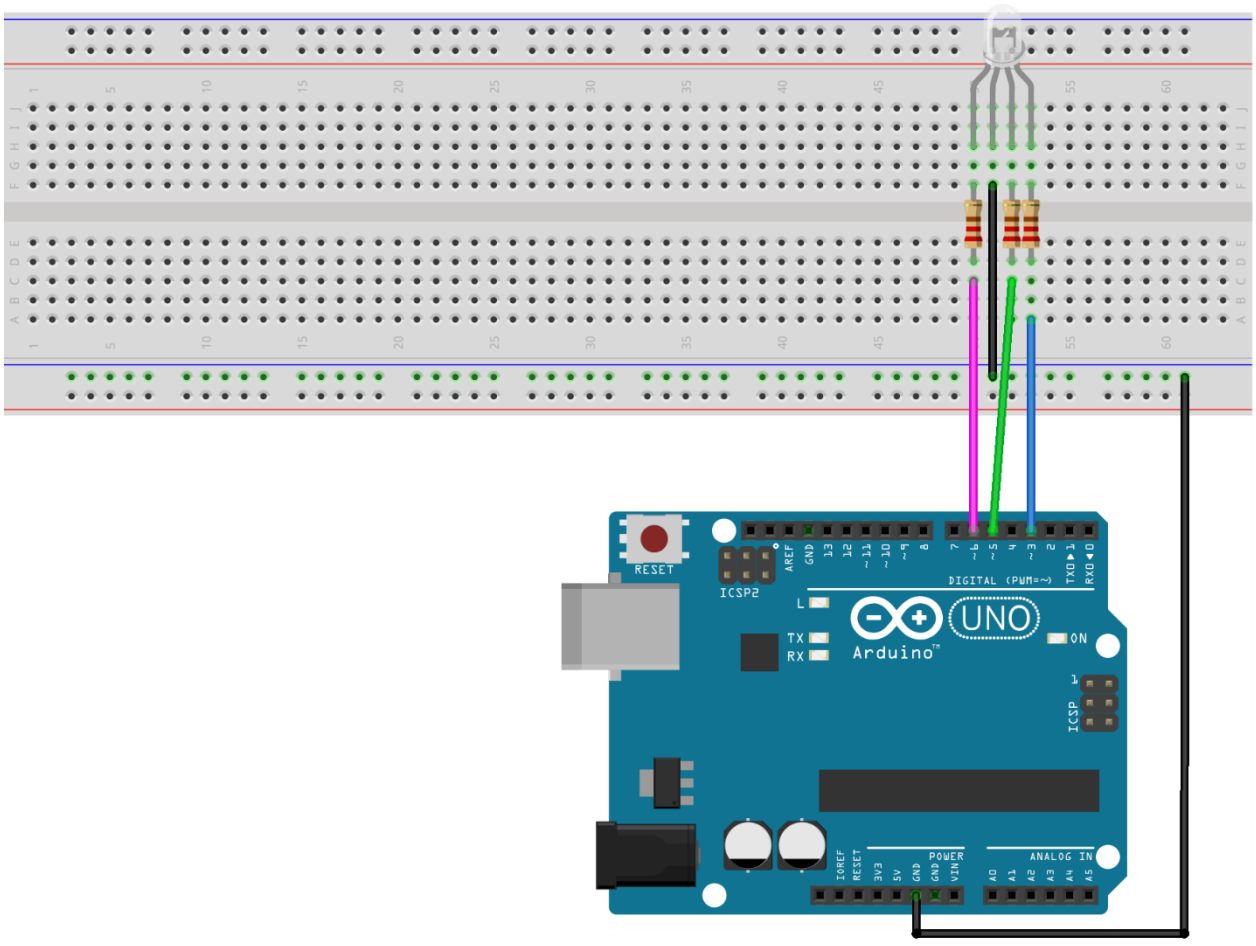

Рисунок 4 – Схема подключения к программе 2

```
#define LED 3

void setup() {
}

void loop() {
    for (int i = 0; i < 255; i += 10) {
        analogWrite(LED, i);
        delay(20);
    }
    for (int i = 255; i > 0; i -= 10) {
        analogWrite(LED, i);
        delay(20);
    }
}
```

Программа 3:

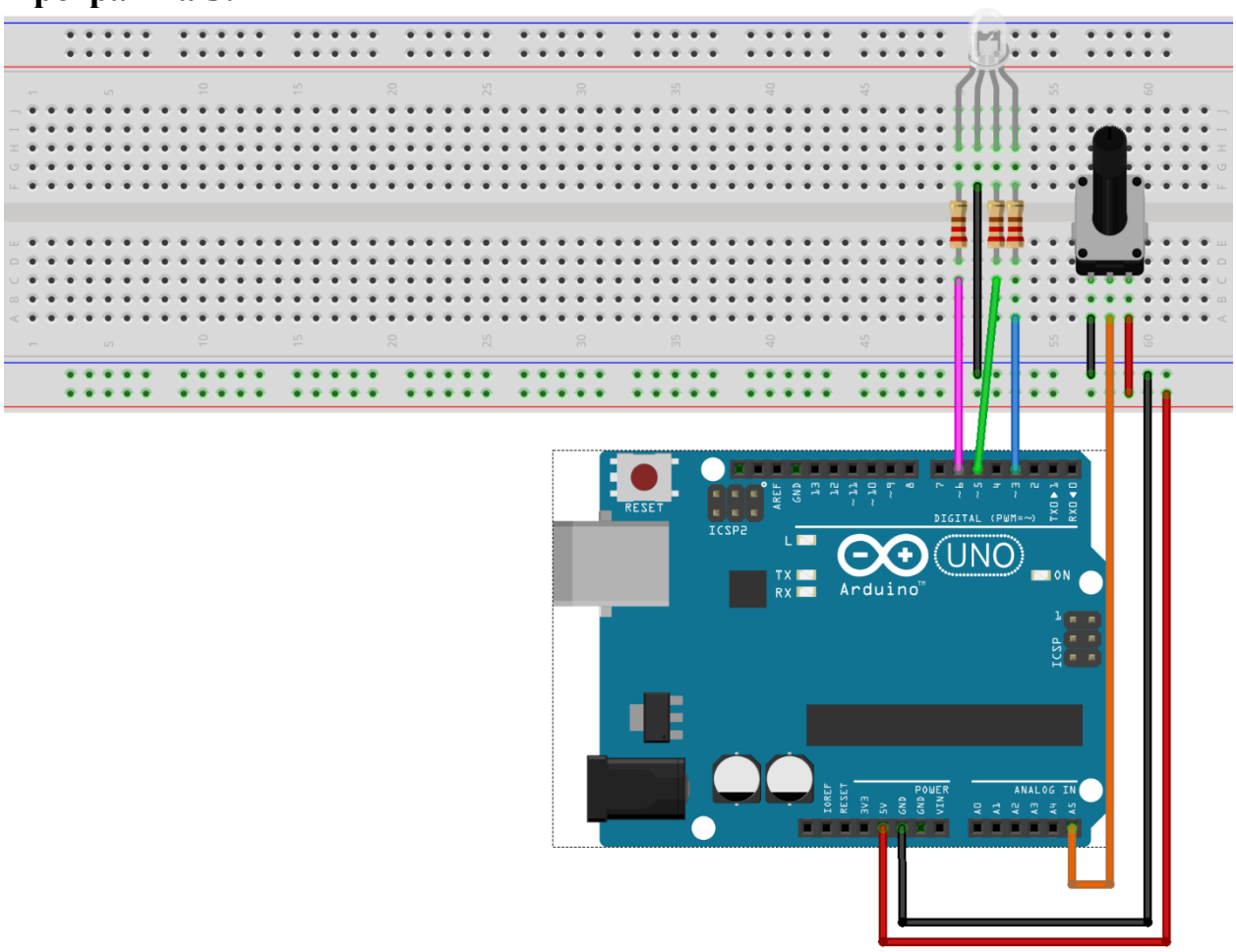

Рисунок 5 – Схема подключения к программе 3 и 4

```
#define LED 3
#define POT_PIN 0
```

```
void setup() {
}

void loop() {
    analogWrite(LED, analogRead(POT_PIN) / 4);
    delay(100);
}
```

Программа 4:

```
#define LED_R 3
#define LED_G 5
#define LED_B 6
#define POT_PIN 0

void setup() {
}

void loop() {
    int v = analogRead(POT_PIN) / 4;
    analogWrite(LED_R, v);
    analogWrite(LED_G, 255 - v);
    analogWrite(LED_B, 255 - v);
    delay(100);
}
```

Программа 5:

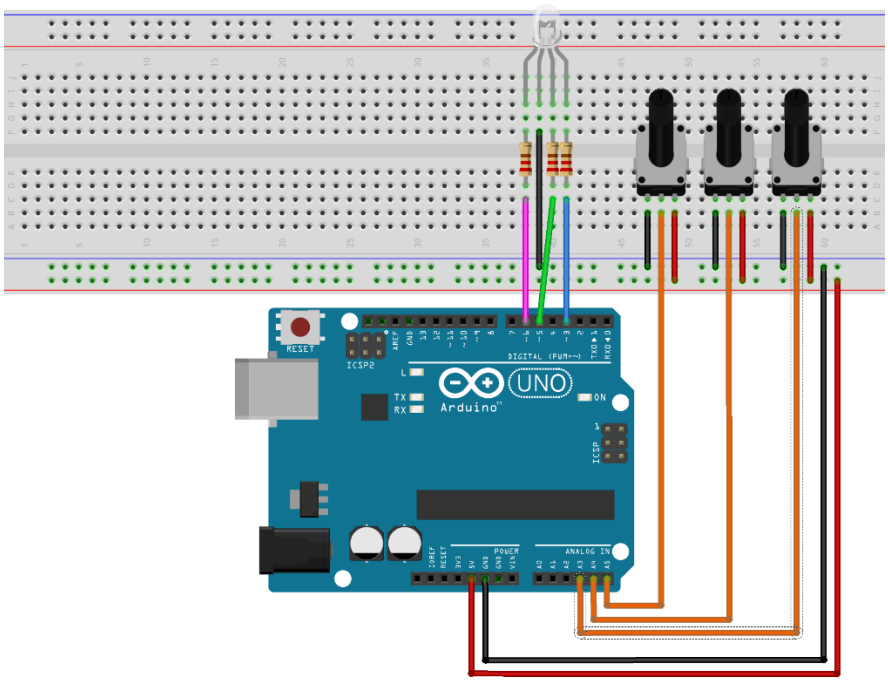

Рисунок 5 – Схема подключения к программе 5

```
#define LED_B 3
#define LED_G 5
#define LED_R 6

#define POT_1 3
#define POT_2 4
#define POT_3 5

void setup() {
}

void loop() {
    analogWrite(LED_B, analogRead(POT_1) / 4);
    delay(100);
    analogWrite(LED_G, analogRead(POT_2) / 4);
    delay(100);
    analogWrite(LED_R, analogRead(POT_3) / 4);
    delay(100);
}
```