

Лабораторная работа №1.

RGB-светодиод


Цель:


Научиться управлять RGB-светодиодом для создания цветовых сигналов робота-паука. Освоить смешивание цветов и ШИМ-управление для индикации состояний робота.

Теоретическая часть



Роль RGB-индикации в роботе-пауке

Визуальное отображение режимов работы
(движение, ожидание, ошибка)








Эмоциональная обратная связь
(цвета настроения)



Что такое RGB-светодиод?

RGB-светодиод – это светодиод, который объединяет в себе три кристалла:

Изменяя яркость каждого цвета, можно получить **любые оттенки**

-  **R** — красный (Red)
-  **G** — зелёный (Green)
-  **B** — синий (Blue)

Типы RGB-светодиодов

| Тип | Подключение общего вывода | Особенности |
|-------------|---------------------------|--|
| Общий катод | Катод → GND | Значения analogWrite() соответствуют яркости |
| Общий анод | Анод → +5V | Значения инвертированы (0 = максимум) |

Схема подключения (общий катод)

Общий вывод ➔ GND

Красный (R) ➔ пин 10

Зелёный (G) ➔ пин 11

Синий (B) ➔ пин 12

Компоненты:

RGB-светодиод

3 резистора типа SMT
(впаяны в плату)

Arduino Nano

Зачем резисторы?
Защита светодиодов от перегорания

ШИМ-управление в Arduino

Работает только на пинах с ~

(D3, D5, D6, D9-D11)

Частота ШИМ: ~490 Гц

(достаточно для плавных переходов)

ШИМ (широотно-импульсная модуляция)
— метод управления яркостью светодиодов

Функция:

```
analogWrite(pin, value)
```

pin — ШИМ-пин

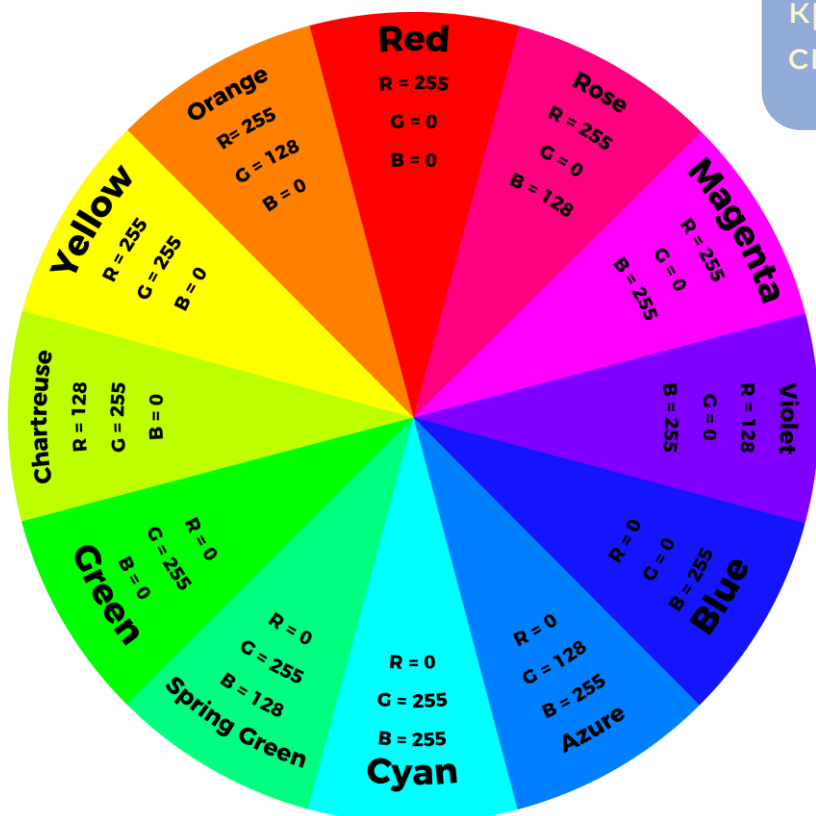
value — от 0 (выключено)
до 255 (максимум яркости)

Примеры:

```
analogWrite(10, 255); // максимум красного  
analogWrite(11, 0);  // зелёный выключен  
analogWrite(12, 0);  // синий выключен
```

Цвета и значения

Цвет формируется комбинацией красного (R), зеленого (G) и синего (B) в диапазоне 0–255:



Чтобы получить красивые цвета — не обязательно использовать только 0 и 255.

Можно варьировать уровни для плавных переходов:

Разбираем функцию `rgb()`

Функция `rgb()` позволяет установить на светодиоде конкретный цвет, указав яркость каждого из трех каналов (R, G, B).

```
void rgb(int r, int g, int b) {  
  analogWrite(blueLED, b); // Подаем на синий светодиод значение b  
  analogWrite(greenLED, g); // Подаем на зеленый светодиод значение g  
  analogWrite(redLED, r); // Подаем на красный светодиод значение r  
}
```

Как это работает?

`analogWrite()` подает на пин сигнал с широтно-импульсной модуляцией (ШИМ)

Значение 255 – цвет горит с максимальной яркостью

Значение 0 – цвет выключен

Примеры вызова функции:

```
rgb(255, 0, 0);    // Красный (только R)
rgb(0, 255, 0);    // Зеленый (только G)
rgb(0, 0, 255);    // Синий (только B)
rgb(255, 255, 0);  // Желтый (R + G)
rgb(0, 255, 255);  // Голубой (G + B)
```

Разбираем функцию `rgb()`

Эта функция создает плавный переход между цветами (эффект радуги) путем последовательного изменения значений R, G и B.

```
void rainbow() {
  // 1. Начинаем с красного цвета (R=255, G=0, B=0)
  rgb(255, 0, 0);
  delay(100); // Ждем 100 мс

  // 2. Переход от красного к желтому (увеличиваем зеленый)
  for (int i = 0; i <= 255; i += 5) {
    rgb(255, i, 0); // Красный остается 255, зеленый растет
    delay(100);     // Задержка для плавности
  }
}
```

Продолжите написание кода по той же структуре

Как это работает?

1 Циклы `for` плавно меняют значения R, G и B.

Цикл `for` — это основа для создания плавных анимаций и переборов значений.

```
for (инициализация; условие; изменение) {
  // Тело цикла
}
```



Инициализация — задаем начальное значение счетчика (например, `int i = 0`).

Условие — цикл выполняется, пока условие истинно (например, `i <= 255`).

Изменение — как меняется счетчик после каждого шага (например, `i += 5`).

2 Циклы `for` плавно меняют значения R, G и B.

Цикл `for` — это основа для создания плавных анимаций и переборов значений.

3 `delay(100)` – задержка между изменениями (в миллисекундах).

Как использовать эти функции в своем коде?

Подключение светодиода

1

Подключить RGB-светодиод к Arduino

2 Объявить пины в коде

```
int redLED = 10;
int greenLED = 11;
int blueLED = 12;
```

GND

пин 10

пин 11

пин 12

Полный пример программы

```
int redLED = 10;
int greenLED = 11;
int blueLED = 12;

void setup() {
  pinMode(redLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
  pinMode(blueLED, OUTPUT);
}
```

```
void loop() {  
    rainbow(); // Запускаем эффект радуги  
}
```

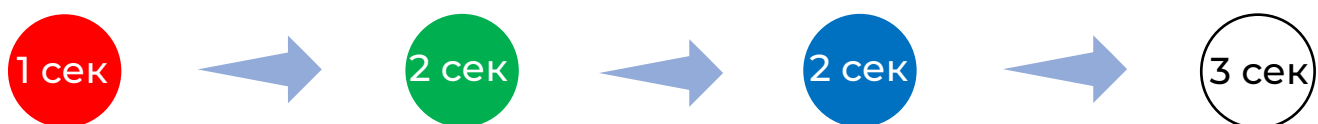
Возможные проблемы и их решение

| Проблема | Решение |
|----------------------------|--|
| Светодиод не загорается | Проверьте подключение GND и пинов. |
| Цвета отображаются неверно | Убедитесь, что R, G, B правильно подключены. |
| Анимация не плавная | Уменьшите шаг в <code>for</code> (например, <code>i += 5</code> → <code>i += 2</code>). |

Практическая часть

Задание 1. Статичные цвета

Напишите программу, которая циклически включает:



Реализуйте эффект моргания:

Светодиод периодически включается и выключается с заданным интервалом

Задание 2. Ручная настройка через Serial

Принимайте из монитора порта значения R, G, B (через пробел: 255 100 0) и вводите их в `rgb()`

Пример кода для считывания данных:

```
if (Serial.available() > 0) {  
  int r = Serial.parseInt();  
}
```

Задание 3. Цветовая смесь

Создайте плавный переход:



Подсказка: Используйте
цикл `for`

Задание 4. Ускорение/замедление радуги

Измените шаг в циклах `for` (например, `i += 5` → `i += 10`) и
задержку `delay()`

! Наблюдайте, как меняется
плавность перехода



Как сделать анимацию
максимально плавной?

Задание 5. Своя палитра переходов

Напишите функцию `customRainbow()`, где цвета переходят в другом
порядке (например: красный → фиолетовый → зеленый → желтый →
синий)

Используйте несколько
последовательных циклов `for`