Практическое занятие №
Тема: «Работа с памятью МК. Перезапись данных»

Цель работы: приобрести практические навыки по программированию блоков кода динамической перезаписи ячеек памяти микроконтроллера.

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.

– Выполнить задание для самостоятельной работы.

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.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

С помощью монитора порта Arduino IDE можно не только просматривать, но и записывать данные в конкретные переменные программы во время её выполнения. Это один из основных способов отладки и интерактивного управления устройством.

1. КАК РАБОТАЕТ ЗАПИСЬ ДАННЫХ

Когда данные отправляются в монитора порта, они по USB попадают в буфер последовательного порта Arduino. В программе нужно написать код, который будет:

– проверять наличие данных в буфере с помощью ***Serial.available()***.

– считывать эти данные с помощью ***Serial.read()*** (для одного символа), ***Serial.parseInt()*** (для целых чисел) или других подобных функций .

– присваивать считанное значение нужной переменной.

ПРИМЕР: УПРАВЛЕНИЕ ЯРКОСТЬЮ СВЕТОДИОДА

В этом примере программа ждет число от 0 до 255 из монитора порта и записывает его в переменную brightness, которая тут же меняет яркость светодиода:

```
const int ledPin = 9;  // Пин со светодиодом
int brightness = 0;     // Переменная для хранения яркости

void setup() {
  Serial.begin(9600);    // Инициализация порта
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // Проверяем, есть ли данные в порту
  if (Serial.available() > 0) {
    // Считываем целое число и сохраняем в переменную
    brightness = Serial.parseInt();
    // Ограничиваем значение
    brightness = constrain(brightness, 0, 255);
    // Применяем значение
    analogWrite(ledPin, brightness);

    // Отправляем подтверждение обратно
    Serial.print("Яркость установлена на: ");
    Serial.println(brightness);
  }
}
```

Откройте монитор порта (Ctrl+Shift+M), убедитесь, что выставлена скорость 9600 baud, введите в верхней строке, например, 128 и нажмите "Отправить". Светодиод изменит яркость, а переменная brightness в программе станет равна 128.

ПОДКЛЮЧИТЕ СВЕТОДИОД К ПЛАТФОРМЕ ARDUINO И НАПИШИТЕ ПРОГРАММУ ВЫШЕ, ВВЕДИТЕ НЕСКОЛЬКО РАЗНЫХ ЗНАЧЕНИЙ, РЕЗУЛЬТАТ СФОТОГРАФИРУЙТЕ И ЗАСКРИНШОТЬТЕ (МОНИТОР ПОРТА) ДЛЯ ОТЧЁТА.

2. ФУНКЦИИ ДЛЯ ЧТЕНИЯ

Serial.read() — считывает один символ (возвращает код символа). Удобно для простых команд, например, '1' или '0' .

Serial.parseFloat() — для считывания чисел с плавающей точкой.

Serial.readString() — для считывания целых строк текста.

Важно, чтобы тип отправляемых данных совпадал с типом переменной, в которую вы их сохраняете. Отправка текста «привет» в переменную типа int вызовет ошибку .

Скорость в скетче (Serial.begin(9600)) и в настройках монитора порта (правый нижний угол) должна совпадать, иначе вместо данных вы увидите иероглифы, «кракозябры» .

В мониторе порта есть опция выбора окончания строки (Новая строка, Возврат каретки и т.д.). Иногда это полезно, но если отправлять просто число, эти служебные символы могут быть проигнорированы или восприняты как часть данных (в зависимости от функции чтения). Например, Serial.parseInt() их проигнорирует, а Serial.readString() может добавить их в конец строки.

Таким образом, отправка данных в переменную — это стандартный и очень полезный прием для создания интерактивных проектов на Arduino.

3. ЗАПИСЬ НЕСКОЛЬКИХ ПЕРЕМЕННЫХ

Для управления несколькими переменными через монитор порта нужно придумать способ, как программа будет различать, какую переменную необходимо изменить. Наиболее распространены способ с командами с префиксами или форматированный ввод.

СПОСОБ 1: БУКВЕННЫЕ КОМАНДЫ

Отправляется буква и число, например: s150 (установить скорость 150) или t180 (установить таймер 180).

```
int speedValue = 0;
int timerValue = 0;

void setup() {
  Serial.begin(9600);
  Serial.println("Введи команду: s<число> или t<число>");
}
```

```
void loop() {
  if (Serial.available() > 0) {
    // Читаем первый символ (букву команды)
    char command = Serial.read();

    // Ждем немного, чтобы успело прийти число
    delay(10);

    // Читаем число, которое идет после буквы
    int number = Serial.parseInt();

    // Выбираем, что делать с числом, в зависимости от
команды
    switch(command) {
      case 's':
      case 'S': // Чтобы принимать и маленькие, и большие
буквы
        speedValue = number;
        Serial.print("Скорость установлена: ");
        Serial.println(speedValue);
        break;

      case 't':
      case 'T':
        timerValue = number;
        Serial.print("Таймер установлен: ");
        Serial.println(timerValue);
        break;

      default:
        Serial.println("Неизвестная команда. Используй s или
t");
    }

    // Очищаем буфер от возможного мусора (перевод строки и
т.д.)
    while (Serial.available() > 0) {
      Serial.read();
    }
  }
}
```

ВЫПОЛНИТЕ КОД ВЫШЕ НА ПЛАТФОРМЕ ARDUINO, РЕЗУЛЬТАТЫ ВСТАВЬТЕ В ОТЧЁТ

СПОСОБ 2: РАЗДЕЛИТЕЛИ (CSV СТИЛЬ)

Отправляются два числа через запятую (или пробел), и они по порядку записываются в переменные.

Например, отправить: 100, 500

```
int var1 = 0;
int var2 = 0;

void setup() {
  Serial.begin(9600);
  Serial.println("Введите два числа через запятую (пример: 150,300):");
}

void loop() {
  if (Serial.available() > 0) {
    // Читаем первое число (до запятой или пробела)
    int first = Serial.parseInt();
    // Читаем второе число (игнорируем запятую/пробел между ними)
    int second = Serial.parseInt();

    // Проверяем, что действительно прочитали что-то
    // Функции parseInt возвращает 0, если данных нет, но 0 может быть допустимым значением.
    // Поэтому можно проверять, были ли данные.

    if (Serial.read() == ',') { // если после второго числа что-то есть
      // просто присваиваем, если числа не -1 (но parseInt не возвращает -1)
    }

    // Присваиваем значения, если отправили только одно число, второе станет 0)
    var1 = first;
    var2 = second;

    Serial.print("var1 = ");
    Serial.print(var1);
    Serial.print(", var2 = ");
    Serial.println(var2);
```

```
    // Чистим буфер
    while (Serial.available() > 0) {
      Serial.read();
    }
  }
}
```

В мониторе порта убедитесь, что в выпадающем списке «Новая строка» (Newline) или «Без окончания» — в данном случае это не критично, так как parseInt игнорирует нецифровые символы.

ВЫПОЛНИТЕ КОД ВЫШЕ НА ПЛАТФОРМЕ ARDUINO, РЕЗУЛЬТАТЫ ВСТАВЬТЕ В ОТЧЁТ

СПОСОБ 3: ИМЕНОВАННЫЕ ПЕРЕМЕННЫЕ

Если переменных много, можно сделать парсинг как в запросах: speed=150&timer=300.

Этот способ сложнее, но можно использовать strtok.

```
int speed = 0;
int timer = 0;
String inputString = "";

void setup() {
  Serial.begin(9600);
  Serial.println("Введи speed=число timer=число");
}

void loop() {
  while (Serial.available()) {
    char c = Serial.read();
    if (c == '\n') {
      // Конец строки - начинаем обработку
      if (inputString.startsWith("speed=")) {
        speed = inputString.substring(6).toInt();
        Serial.print("Speed set to ");
        Serial.println(speed);
      }
      else if (inputString.startsWith("timer=")) {
        timer = inputString.substring(6).toInt();
        Serial.print("Timer set to ");
```

```
        Serial.println(timer);
      }
      else {
        // Если строка содержит и то, и другое, нужен более
хитрый парсинг
        // Например, ищем индексы "speed=" и "timer="
        int speedIndex = inputString.indexOf("speed=");
        int timerIndex = inputString.indexOf("timer=");

        if (speedIndex >= 0) {
          int val = inputString.substring(speedIndex +
6).toInt();
          speed = val;
        }
        if (timerIndex >= 0) {
          int val = inputString.substring(timerIndex +
6).toInt();
          timer = val;
        }

        Serial.print("speed = ");
        Serial.print(speed);
        Serial.print(", timer = ");
        Serial.println(timer);
      }
      inputString = ""; // Очищаем для следующей команды
    }
    else {
      inputString += c; // Добавляем символ к строке
    }
  }
}
```

ПРИМЕР РАБОТЫ:

Открыть монитор порта.

Написать speed=150 и нажать «Отправить» — меняется переменная speed на значение 150.

Написать timer=300 — меняется переменная timer на значение 300.

Скетч при этом продолжает работать со старыми значениями, пока не придет новая команда.

ВЫПОЛНИТЕ КОД ВЫШЕ НА ПЛАТФОРМЕ ARDUINO, РЕЗУЛЬТАТЫ ВСТАВЬТЕ В ОТЧЁТ

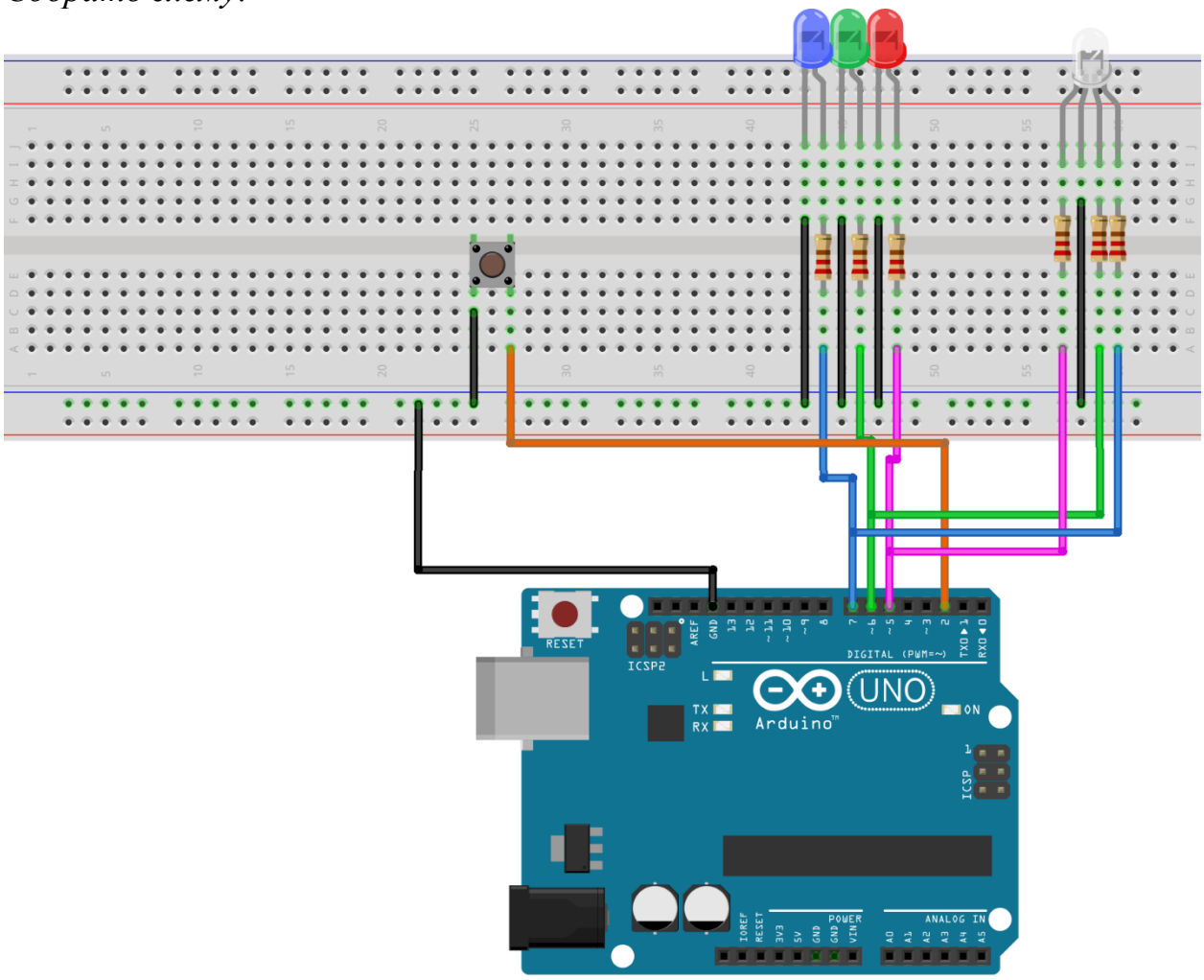
4. ЗАДАНИЕ

1) УПРАВЛЕНИЕ АНИМАЦИЕЙ СВЕТОДИОДОВ БУКВЕННЫМИ КОМАНДАМИ

Таблица компонентов:

№	Название компонента	Количество, шт.
1.	Светодиод 5мм	3
2.	RGB-светодиод 5мм	1
3.	Резисторы 220 Ом	3
4.	Тактовая кнопка 5/10 мм	1
5.	Соединительные провода	8

Собрать схему:



Алгоритм работы:

Три светодиода создают эффект бегущего огня. Кнопка запускает или останавливает анимацию. Через монитор порта можно изменять параметры, но только когда анимация остановлена. Команды формируются буквой и числом: s50 (скорость), r128 (яркость красного), g200 (яркость зеленого), b75

(яркость синего). Программа парсит первый символ как команду, а остальное как число.

Написать программный код:

```
// Пины светодиодов
const int redPin = 9;
const int greenPin = 10;
const int bluePin = 11;
const int buttonPin = 2;

// Переменные для хранения яркости (0-255)
int redBrightness = 128;
int greenBrightness = 128;
int blueBrightness = 128;
int animationSpeed = 100; // Задержка в миллисекундах

// Состояния
bool animationRunning = true;
bool lastButtonState = HIGH;
int step = 0; // Текущий шаг анимации

void setup() {
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP); // Подтяжка к питанию,
кнопка замыкает на GND

  Serial.begin(9600);
  Serial.println("Система запущена. Команды: s[скорость],
r[яркость], g[яркость], b[яркость]");
}

void loop() {
  checkButton();      // Проверка нажатия кнопки
  checkSerial();      // Проверка команд из монитора порта

  if (animationRunning) {
    runAnimation();   // Выполнение шага анимации
  }

  delay(animationSpeed); // Пауза между шагами
}
```

```
// Функция проверки кнопки (переключение старт/стоп)
void checkButton() {
  bool buttonState = digitalRead(buttonPin);

  // Если кнопка была нажата (переход из HIGH в LOW)
  if (lastButtonState == HIGH && buttonState == LOW) {
    animationRunning = !animationRunning; // Инверсия
состояния

    if (animationRunning) {
      Serial.println("Анимация запущена");
    } else {
      Serial.println("Анимация остановлена. Можно менять
параметры");
    }

    delay(50); // Защита от дребезга
  }

  lastButtonState = buttonState;
}

// Функция проверки последовательного порта
void checkSerial() {
  if (Serial.available() > 0) {
    // Если анимация работает, изменение параметров
запрещено
    if (animationRunning) {
      char clearBuffer;
      Serial.println("Остановите анимацию для изменения
параметров");
      // Очистка буфера
      while (Serial.available() > 0) {
        clearBuffer = Serial.read();
      }
      return;
    }

    // Чтение первого символа (команда)
    char command = Serial.read();
    delay(10); // Ожидание поступления числа

    // Чтение числового значения
    int value = Serial.parseInt();
```

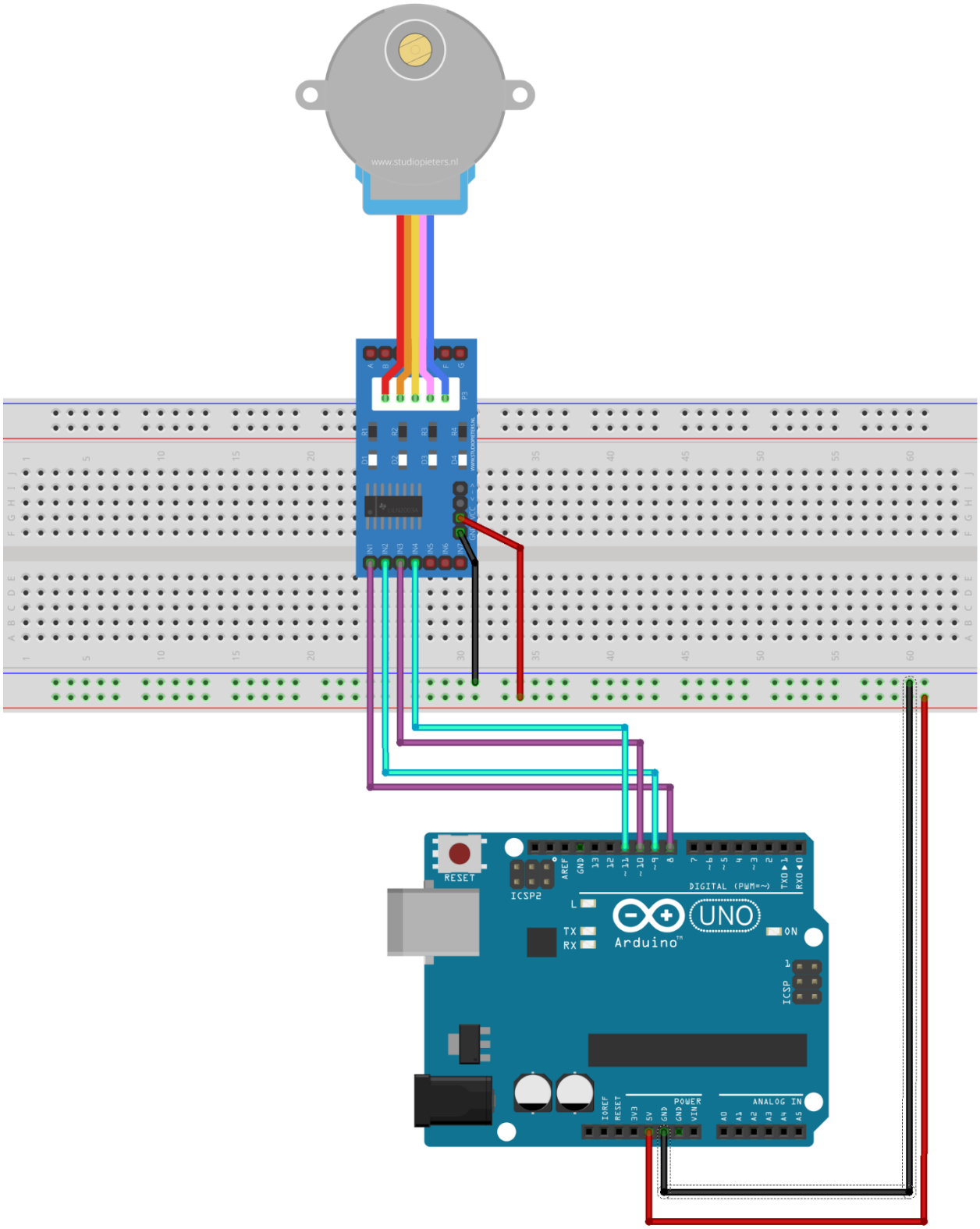
```
    // Ограничение значения в пределах 0-255
    value = constrain(value, 0, 255);

    // Обработка команд
    switch (command) {
      case 's':
      case 'S':
        animationSpeed = value;
        Serial.print("Скорость анимации установлена: ");
        Serial.println(animationSpeed);
        break;

      case 'r':
      case 'R':
        redBrightness = value;
        analogWrite(redPin, redBrightness);
        Serial.print("Яркость красного установлена: ");
        Serial.println(redBrightness);
        break;

      case 'g':
      case 'G':
        greenBrightness = value;
        analogWrite(greenPin, greenBrightness);
        Serial.print("Яркость зеленого установлена: ");
        Serial.println(greenBrightness);
        break;

      case 'b':
      case 'B':
        blueBrightness = value;
        analogWrite(bluePin, blueBrightness);
        Serial.print("Яркость синего установлена: ");
        Serial.println(blueBrightness);
        break;

      default:
        Serial.println("Неизвестная команда. Используйте s,
r, g, b");
    }

    // Очистка оставшихся символов в буфере
    while (Serial.available() > 0) {
```

```
      Serial.read();
    }
  }
}

// Функция анимации "бегущий огонь"
void runAnimation() {
  // Выключение всех светодиодов
  analogWrite(redPin, 0);
  analogWrite(greenPin, 0);
  analogWrite(bluePin, 0);

  // Включение текущего светодиода с его индивидуальной
яркостью
  switch (step) {
    case 0:
      analogWrite(redPin, redBrightness);
      break;
    case 1:
      analogWrite(greenPin, greenBrightness);
      break;
    case 2:
      analogWrite(bluePin, blueBrightness);
      break;
  }

  // Переход к следующему шагу
  step++;
  if (step > 2) {
    step = 0;
  }
}
```

2) УПРАВЛЕНИЕ ШАГОВЫМ ДВИГАТЕЛЕМ С ПОМОЩЬЮ CSV СТИЛЯ

№	Название компонента	Количество, шт.
1.	Шаговый двигатель 28BYJ-48	*1*
2.	Драйвер ULN2003	*1*
3.	Соединительные провода	*6-8*

Собрать схему:



Написать программный код:

```
// Подключение библиотеки для шагового двигателя
#include <Stepper.h>

// Количество шагов на один оборот для 28BYJ-48 (в режиме полного шага)
```

```
const int stepsPerRevolution = 2048;

// Инициализация двигателя с указанием пинов
Stepper myStepper(stepsPerRevolution, 8, 10, 9, 11);

// Переменные для хранения параметров
int targetFloor = 0;      // Этаж назначения (в шагах)
int motorSpeed = 10;      // Скорость (задержка между
шагами)
int moveDelay = 1000;     // Задержка перед отправкой и
после прибытия (мс)

// Состояние двигателя
bool isMoving = false;
int currentPosition = 0;  // Текущая позиция в шагах от нуля

void setup() {
  Serial.begin(9600);
  myStepper.setSpeed(10); // Начальная скорость

  Serial.println("Система лифта запущена");
  Serial.println("Формат ввода: этаж(в шагах), скорость(1-
20), задержка(мс)");
  Serial.println("Пример: 1024,5,2000");
}

void loop() {
  checkSerial();

  // Если есть команда на движение и двигатель не занят
  if (targetFloor != 0 && !isMoving) {
    startMoving();
  }

  // Небольшая задержка для стабильности
  delay(10);
}

// Функция проверки последовательного порта
void checkSerial() {
  if (Serial.available() > 0) {
    // Чтение трех чисел, разделенных запятыми или пробелами
    int floor = Serial.parseInt();
    int speed = Serial.parseInt();
```

```
    int delayTime = Serial.parseInt();

    // Проверка, что данные были прочитаны
    // parseInt возвращает 0, если данных нет, поэтому
требуется проверка контекста
    // Если прочитано только первое число, остальные будут 0

    // Очистка буфера
    while (Serial.available() > 0) {
      Serial.read();
    }

    // Применение значений, если они не нулевые или
специально обработаны
    if (floor != 0 || speed != 0 || delayTime != 0) {

      // Обновление значений, если они были переданы
      if (floor != 0) {
        targetFloor = floor;
      }

      if (speed != 0) {
        // Ограничение скорости в разумных пределах
        motorSpeed = constrain(speed, 1, 20);
        myStepper.setSpeed(motorSpeed);
      }

      if (delayTime != 0) {
        moveDelay = delayTime;
      }

      // Вывод подтверждения
      Serial.print("Параметры загружены: этаж=");
      Serial.print(targetFloor);
      Serial.print(" шагов, скорость=");
      Serial.print(motorSpeed);
      Serial.print(", задержка=");
      Serial.print(moveDelay);
      Serial.println(" мс");

      // Сигнал о готовности к движению
      Serial.println("Лифт готов к отправке");
    } else {
```

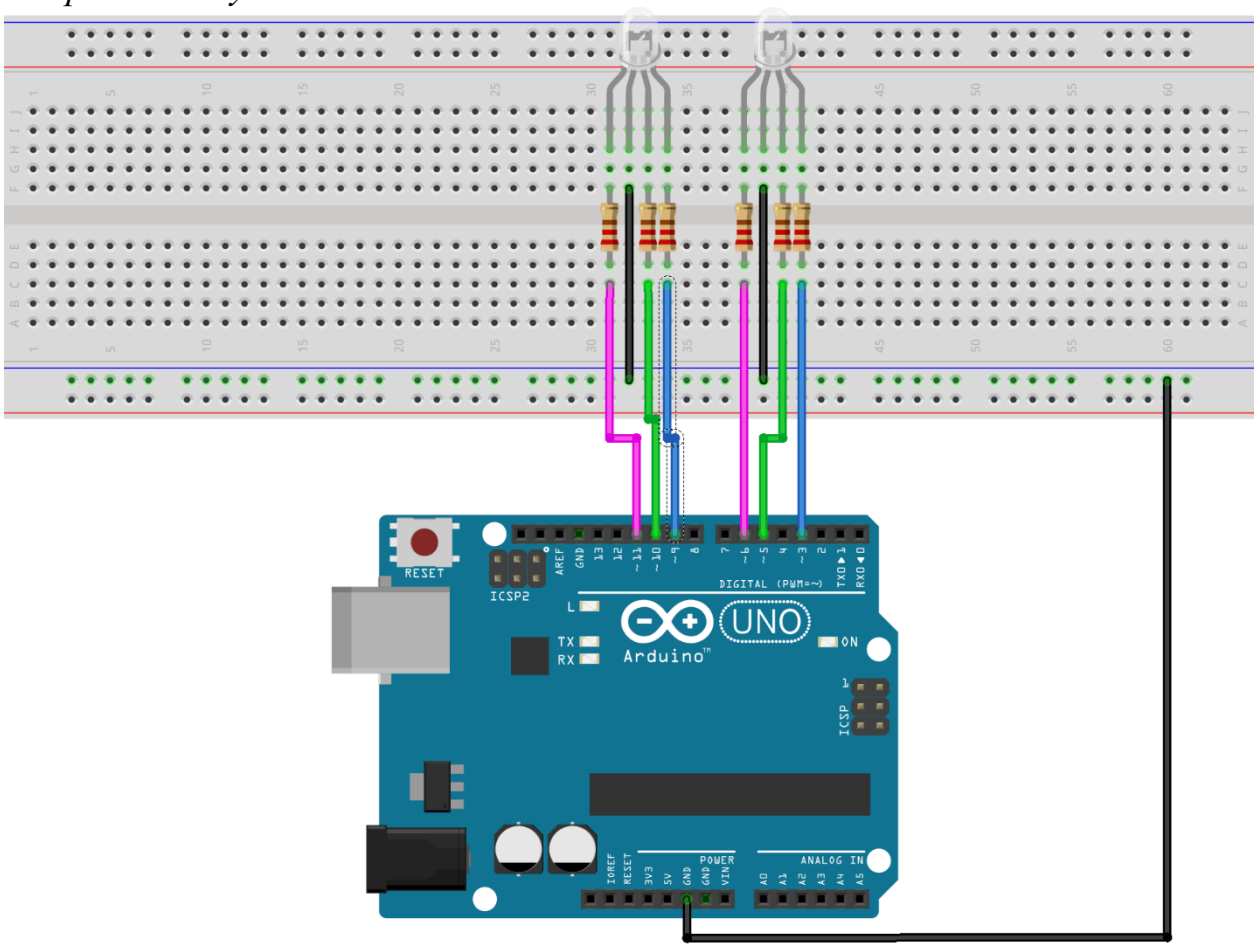
```
      Serial.println("Ошибка ввода. Используйте формат:
число,число,число");
    }
  }
}

// Функция запуска движения лифта
void startMoving() {
  isMoving = true;

  Serial.print("Ожидание перед отправкой: ");
  Serial.print(moveDelay);
  Serial.println(" мс");
  delay(moveDelay);

  Serial.println("Лифт отправляется...");

  // Расчет необходимого количества шагов
  int stepsToMove = targetFloor - currentPosition;

  // Выполнение движения
  myStepper.step(stepsToMove);

  // Обновление текущей позиции
  currentPosition = targetFloor;

  Serial.print("Лифт прибыл на этаж. Текущая позиция: ");
  Serial.println(currentPosition);

  Serial.print("Ожидание после прибытия: ");
  Serial.print(moveDelay);
  Serial.println(" мс");
  delay(moveDelay);

  // Сброс флага движения, но целевой этаж сохраняется
  isMoving = false;

  Serial.println("Лифт готов к следующей команде");
}
```

3) УПРАВЛЕНИЕ RGB-СВЕТОДИОДАМИ С ПОМОЩЬЮ ИМЕНОВАННЫХ ПЕРЕМЕННЫХ

Таблица компонентов:

№	Название компонента	Количество, шт.
1.	RGB-светодиод 5мм	2
2.	Соединительные провода	9

Собрать схему:



Написать программный код:

```
// Пины для первого RGB светодиода
const int redOnePin = 3;
const int greenOnePin = 5;
const int blueOnePin = 6;

// Пины для второго RGB светодиода
const int redTwoPin = 9;
const int greenTwoPin = 10;
const int blueTwoPin = 11;

// Переменные для хранения значений яркости (0-255)
```

```
int redOne = 0;
int greenOne = 0;
int blueOne = 0;
int redTwo = 0;
int greenTwo = 0;
int blueTwo = 0;

// Буфер для приема строки
String inputString = "";

void setup() {
  // Настройка пинов как выходы
  pinMode(redOnePin, OUTPUT);
  pinMode(greenOnePin, OUTPUT);
  pinMode(blueOnePin, OUTPUT);
  pinMode(redTwoPin, OUTPUT);
  pinMode(greenTwoPin, OUTPUT);
  pinMode(blueTwoPin, OUTPUT);

  Serial.begin(9600);
  Serial.println("Управление RGB светодиодами");
  Serial.println("Доступные переменные:");
  Serial.println("redone, greenone, blueone (для первого
светодиода)");
  Serial.println("redtwo, greentwo, bluetwo (для второго)");
  Serial.println("Формат команды: переменная=значение");
  Serial.println("Пример: redone=128, redtwo=255");
}

void loop() {
  // Чтение данных из последовательного порта
  while (Serial.available()) {
    char receivedChar = Serial.read();

    // Если получен символ конца строки
    if (receivedChar == '\n') {
      // Обработка полученной строки
      parseCommand(inputString);
      // Очистка буфера
      inputString = "";
    } else {
      // Добавление символа в буфер
      inputString += receivedChar;
    }
```

```
  }

  // Обновление состояния светодиодов
  updateLEDs();
}

// Функция парсинга команды вида "переменная=значение"
void parseCommand(String command) {
  // Удаление пробелов в начале и конце
  command.trim();

  // Поиск символа '='
  int equalsIndex = command.indexOf('=');

  if (equalsIndex == -1) {
    Serial.println("Ошибка: отсутствует символ '='");
    return;
  }

  // Извлечение имени переменной и значения
  String varName = command.substring(0, equalsIndex);
  String valueStr = command.substring(equalsIndex + 1);

  // Преобразование значения в число
  int value = valueStr.toInt();

  // Ограничение значения
  value = constrain(value, 0, 255);

  // Приведение имени переменной к нижнему регистру для
унификации
  varName.toLowerCase();

  // Поиск соответствия и обновление переменной
  bool variableFound = true;

  if (varName == "redone") {
    redOne = value;
    Serial.print("redone установлен в ");
    Serial.println(value);
  }
  else if (varName == "greenone") {
    greenOne = value;
    Serial.print("greenone установлен в ");
```

```
    Serial.println(value);
  }
  else if (varName == "blueone") {
    blueOne = value;
    Serial.print("blueone установлен в ");
    Serial.println(value);
  }
  else if (varName == "redtwo") {
    redTwo = value;
    Serial.print("redtwo установлен в ");
    Serial.println(value);
  }
  else if (varName == "greentwo") {
    greenTwo = value;
    Serial.print("greentwo установлен в ");
    Serial.println(value);
  }
  else if (varName == "bluetwo") {
    blueTwo = value;
    Serial.print("bluetwo установлен в ");
    Serial.println(value);
  }
  else {
    variableFound = false;
    Serial.print("Неизвестная переменная: ");
    Serial.println(varName);
    Serial.println("Доступные переменные: redone, greenone,
blueone, redtwo, greentwo, bluetwo");
  }

  // Обновление светодиодов сразу после изменения
  if (variableFound) {
    updateLEDs();
  }
}

// Функция обновления яркости всех светодиодов
void updateLEDs() {
  // Первый RGB светодиод
  analogWrite(redOnePin, redOne);
  analogWrite(greenOnePin, greenOne);
  analogWrite(blueOnePin, blueOne);

  // Второй RGB светодиод
```

```
  analogWrite(redTwoPin, redTwo);
  analogWrite(greenTwoPin, greenTwo);
  analogWrite(blueTwoPin, blueTwo);
}
```

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Можно ли с помощью монитора порта Arduino IDE не только читать данные из программы, но и записывать их в переменные?
2. По какому физическому интерфейсу данные из монитора порта попадают в Arduino?
3. Куда именно попадают данные, отправленные из монитора порта, прежде чем программа сможет их прочитать?
4. Для чего используется интерактивное изменение переменных через монитор порта? (Назовите хотя бы одну причину).
5. Какая функция проверяет, есть ли в буфере последовательного порта данные для чтения?
6. Какую функцию нужно использовать, чтобы считать целое число (int) из строки, отправленной в монитор порта?
7. С помощью какой функции можно считать число с плавающей точкой (float)?
8. Для чего используется функция Serial.readString()?
9. Какая команда в коде программы (setup) является обязательной для инициализации работы с монитором порта?
10. В примере с буквенными командами, для чего использовалась конструкция case 's': case 'S':?
11. В чем заключается потенциальная проблема способа с разделителями (CSV), если программа ожидает два числа, а пользователь отправит только одно?
12. Для чего в примере с управлением яркостью используется функция constrain(brightness, 0, 255);?