

Дерзай!

Краткий справочник по подключению датчиков, модулей и иных электронных устройств к плате Arduino

(часть 2 руководства к большому набору
«Умный дом на базе Arduino»)



РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

14+

www.bhv.ru/books/kits



Содержание

1. Датчики	3
1.1. Датчики температуры и влажности DHT11/DHT22	3
1.2. ИК-датчик движения HC-SR501	7
1.3. УЗ-датчик расстояния HC-SR04	10
1.4. Датчик пламени YG1006	13
1.5. Датчик влажности почвы YL-38	16
1.6. Датчик уровня жидкости	20
1.7. Датчик газа MQ-135	23
1.8. Датчик звука	26
1.9. Потенциометр	29
1.10. Модуль фоторезистора KY-018	30
2. Модули	32
2.1. Активный пьезоэлектрический зуммер 5 В	32
2.2. RFID-модуль RC522	37
2.3. Модуль реле	41
2.4. 8-разрядный расширитель портов PCF8574	47
2.5. Модуль BLE Bluetooth HM-10/11	50
3. Светодиоды и дисплеи	59
3.1. Светодиод RGB	59
3.2. ЖК-дисплей 1602 с модулем I ² C	60
3.3. Светодиодная матрица 8×8 с драйвером MAX7219	64
4. Двигатели	68
4.1. Серводвигатель TowerPro SG90 9G	68

1. Датчики

1.1. Датчики температуры и влажности DHT11/DHT22

Датчики DHT11/DHT22 (рис. S1.1) предназначены для измерения температуры и влажности воздуха.

Внешний вид, назначение контактов

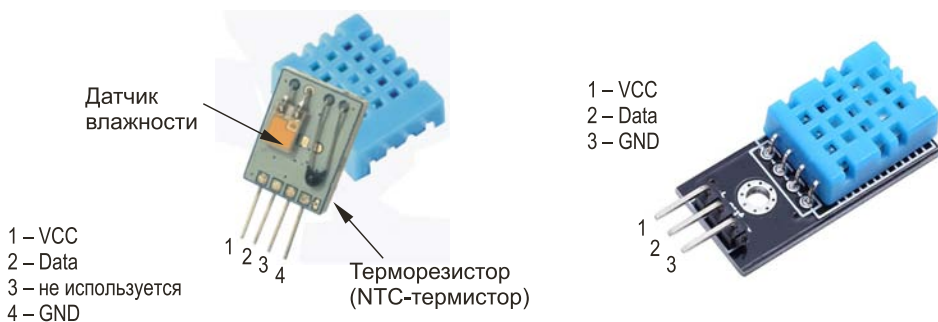


Рис. S1.1. Датчик DHT11

Основные характеристики

Наименование	Значение	
	DHT11	DHT22
Диапазон измерения температуры/точность измерения	0 ÷ 50°C /±2 °C	-40 ÷ 125°C /±0,5 °C
Диапазон измерения относительной влажности	20 ÷ 80% /±5 °C	0 ÷ 100% /±2 ÷ 5 °C
Частота опроса датчиков, Гц	1 (один раз в секунду)	0,5 (1 раз в две секунды)
Размеры, мм	15,5×12,0×5,5	15,1×25,0×7,7
Напряжение питания, В	3 ÷ 5	3 ÷ 5
Максимальный ток, мА	2,5	2,5

Схема подключения

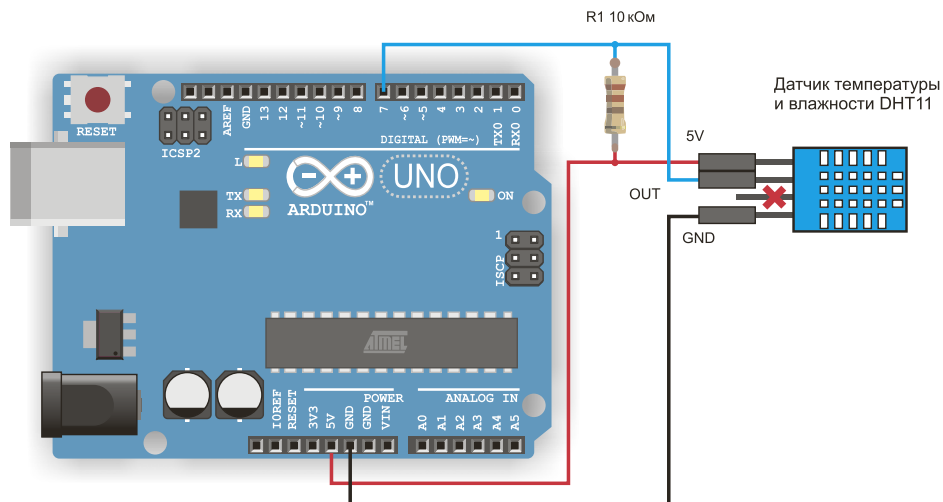


Рис. S1.2. Подключение датчика DHT11

Внимание!

Если расстояние от датчика до Arduino небольшое, рекомендуемый номинал резистора R1 10 кОм (рис. S1.2), а для расстояния больше 20 метров рекомендуется резистор номиналом 5,1 кОм.

Если же датчик имеет только три контакта (5V, OUT и GND), то резистор R1 вообще не нужен.

Программный код

1. Загрузите библиотеку `DHT-sensor-library` для работы Arduino с датчиками DHT11 и DHT22. Для этого откройте Менеджер библиотек, выполнив команду **Инструменты | Управлять библиотеками**. Справа сверху в строке поиска введите `DHT`. В открывшемся списке выберите **DHT sensor library by Adafruit** версии 1.2.3 (в версиях 1.3.0 и новее возможна ошибка при компиляции). Нажмите кнопку **Установка**.
2. Загрузите скетч из листинга S1.1.

Электронный архив скетчей и библиотек

Электронный архив скетчей и библиотек расположен по адресу <ftp://ftp.bhv.ru/9785977566087.zip>. Скетчи в этом архиве вы найдете в папке *Скетчи*, а библиотеки — в папке *Библиотеки*.

Листинг S1.1. Измерение температуры и влажности с помощью датчика DHT11

```
#include "DHT.h"
#define DHTPIN 7 //пин для получения сигнала от датчика DHT11

// Раскомментируйте тип датчика, который вы используете
#define DHTTYPE DHT11 // DHT 11
// #define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321
// #define DHTTYPE DHT21 // DHT 21 (AM2301)
DHT dht(DHTPIN, DHTTYPE);
void setup() {
    Serial.begin(9600);
    dht.begin();
}
void loop() {
    // Задержка перед измерениями
    delay(2000);
    float h = dht.readHumidity();
    //чтение температуры в градусах Цельсия °C (по умолчанию)
    float t = dht.readTemperature();
    //чтение температуры в градусах Фаренгейта °F (isFahrenheit = true)
    float f = dht.readTemperature(true);
    // Проверка чтения данных
    if (isnan(h) || isnan(t) || isnan(f)) {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }
    // Вычисление теплового индекса в °F (по умолчанию)
    float hif = dht.computeHeatIndex(f, h);
    // Вычисление теплового индекса в °C (isFahreheit = false)
    float hic = dht.computeHeatIndex(t, h, false);
    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.print(" %\t");
    Serial.print("Temperature: ");
```

```
Serial.print(t);  
Serial.print(" *C ");  
Serial.print(f);  
Serial.print(" *F\t");  
Serial.print("Heat index: ");  
Serial.print(hic);  
Serial.print(" *C ");  
Serial.print(hif);  
Serial.println(" *F");  
}
```

Результат

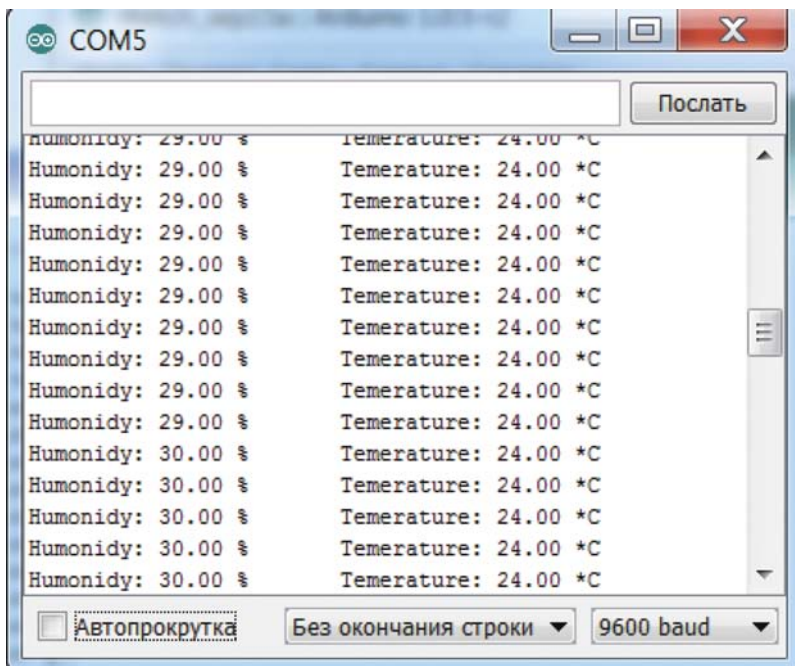


Рис. S1.3. Результаты измерений датчика DHT11

1.2. ИК-датчик движения HC-SR501

Пассивный инфракрасный датчик движений HC-SR501 (PIR, Passive Infrared) фиксирует движения объектов. Матрица из 15 небольших линз фокусирует ИК-излучение из разных участков окружающего пространства на пироэлектрический детектор, основу которого составляет пластина из танталата лития, вырабатывающая небольшое напряжение в ответ на поступающее тепловое излучение. При перемещении объекта из одной зоны в другую генерируется выходной сигнал (рис. S2.1, S2.2).

Основные характеристики

Наименование	Значение
Постоянное напряжение, В	4,5 ÷ 20
Ток потребления в режиме ожидания	менее 50 мкА
Наибольший потребляемый ток во время работы, мА	65
Напряжение логических уровней, В	3,3
Расстояние обнаружения	3 ÷ 7 м, по умолчанию 7 м
Максимальный угол обнаружения 110° на расстоянии 7 м	120°
Время поддержания высокого уровня выхода при присутствии	20 ÷ 300 с
Время игнорирования событий после фиксации	0,2 с
Температура окружающего воздуха при работе	-15 ÷ 70°C
Размеры, мм	32×24×28

Внешний вид, назначение контактов

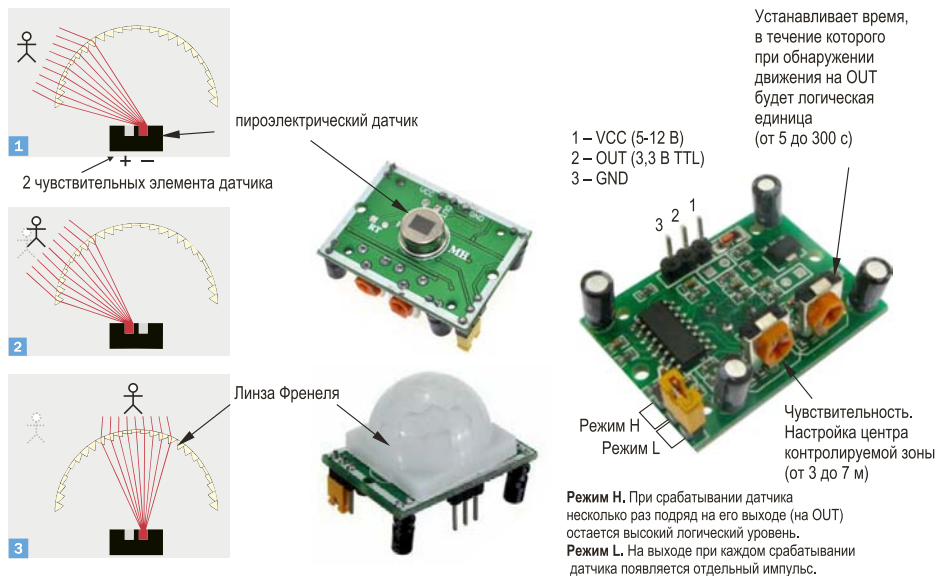


Рис. S2.1. Датчик HC-SR501

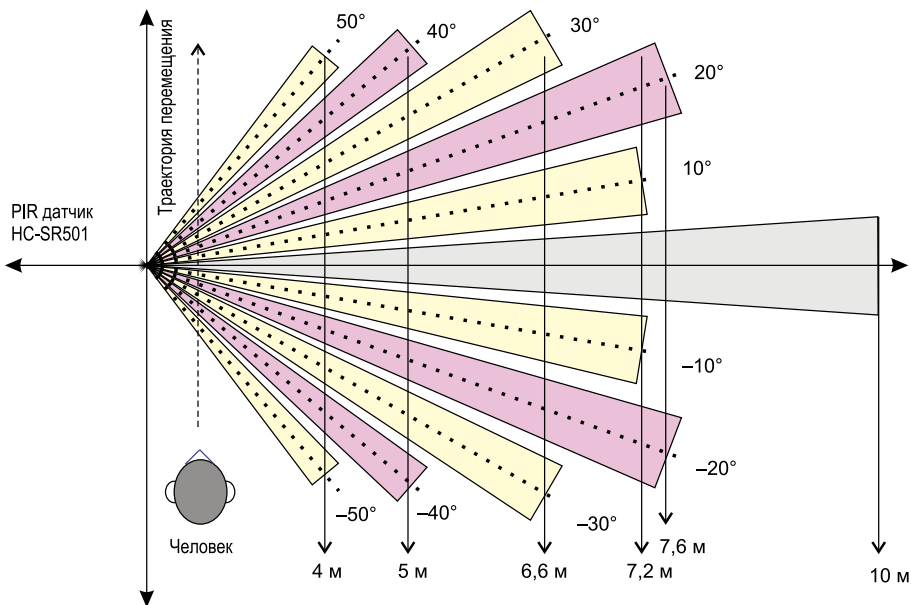


Рис. S2.2. Поле зрения датчика PIR с линзой Френеля

Схема подключения

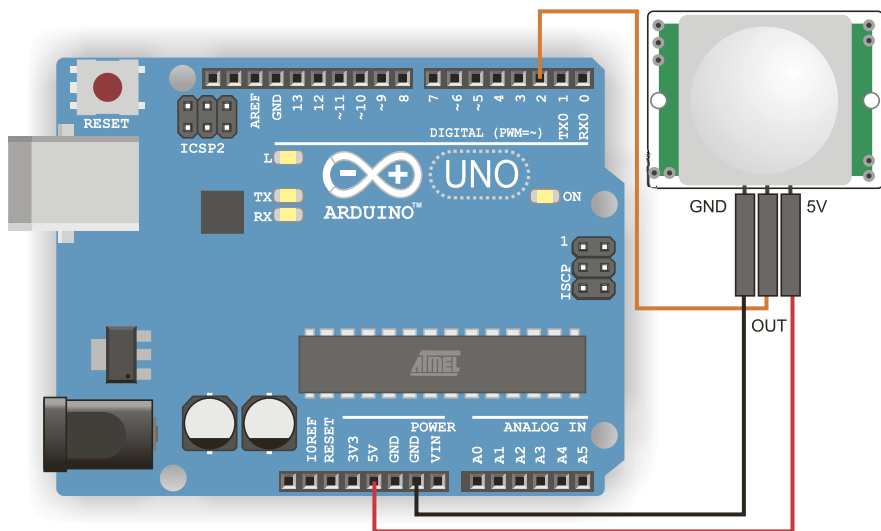


Рис. S2.3. Схема подключения датчика движения

Программный код

Листинг S2.1. Обнаружение перемещения датчиком движения HC-SR501

```
#define IKPin 2      //Номер пина ИК-датчика движения HC-SR501
#define ledPin 13   //Номер пина встроенного светодиода
void setup(){
    Serial.begin(9600);
    pinMode(IKPin, INPUT); // Объявляем пин,
    // к которому подключен датчик движения, входом
    pinMode(ledPin,OUTPUT); // Объявляем пин,
    // к которому подключен светодиод, выходом
}
void loop(){
    int pirVal = digitalRead(IKPin); //считываем
    // значения с датчика движения
    // Если обнаружили движение, то транслируем
    // сигнал тревоги в Монитор порта
    // и включаем светодиод

    if(pirVal == HIGH) {
        digitalWrite(ledPin, HIGH);
        Serial.print("ALARM");
        delay(2000);
    }
    else {
        Serial.print("Scanning");
        digitalWrite(ledPin,LOW);
    }
}
```

Результат

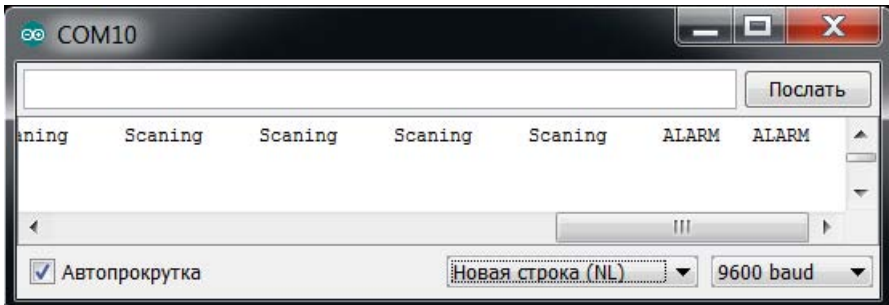


Рис. S2.4. Фиксация движения на Мониторе порта

1.3. УЗ-датчик расстояния HC-SR04

УЗ-датчик HC-SR04 определяет расстояние до объекта, измеряя время прихода отраженной волны (рис. S3.1). Такой же принцип ультразвуковой эхолокации используют летучие мыши.

Основные характеристики

Наименование	Значение
Напряжение питания, В	5
Ток покоя, мА	<2
Эффективный угол	менее 15°
Диапазон измерения расстояния, см	2 ÷ 400
Разрешение, см	0,3
Рабочий ток, мА	15
Рабочая частота, кГц	40
Размеры, мм	45×20×15

Внешний вид, назначение контактов

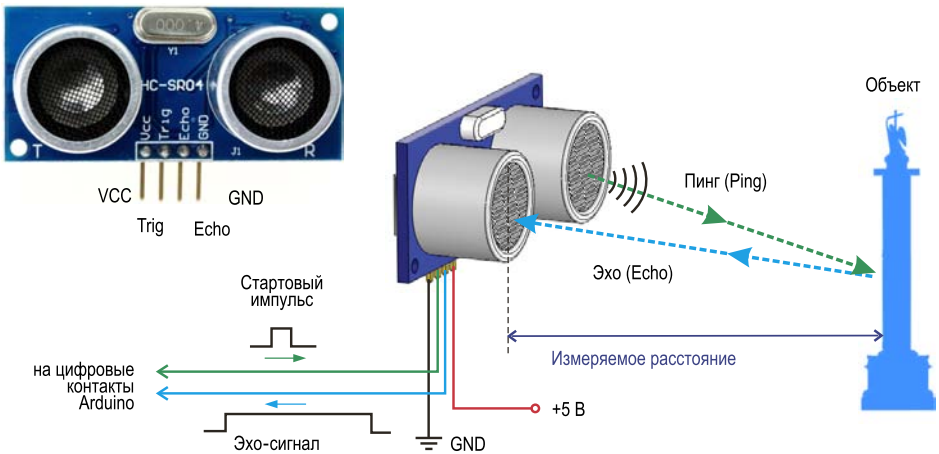


Рис. S3.1. Принцип действия УЗ-датчика

Схема подключения

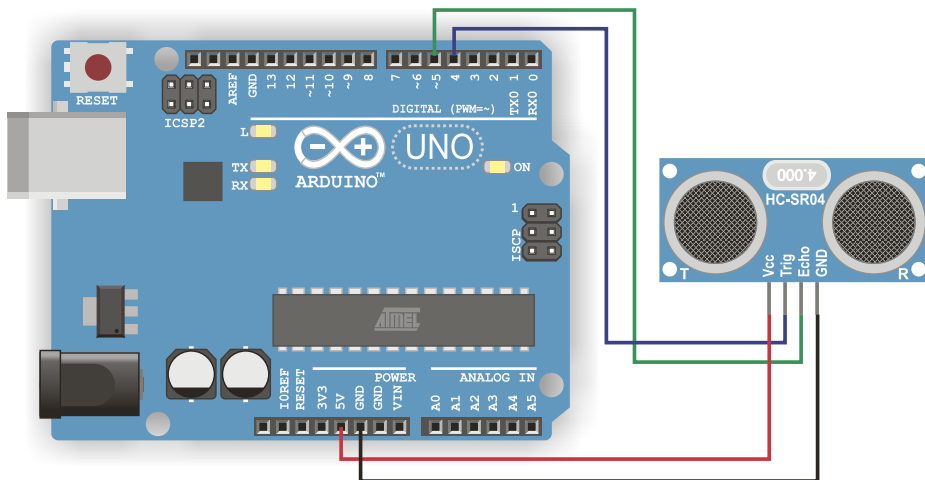


Рис. S3.2. Схема подключения УЗ-датчика

Программный код

Листинг S3.1. Определение дальности с помощью УЗ-датчика HC-SR04

```
#define TrigPin 4 //Номер пина Trig Датчика расстояния(HC-SR04)
#define EchoPin 5 //Номер пина Echo Датчика расстояния(HC-SR04)
long duration, cm, inches; //переменные
void setup() {
  //Инициализация последовательного порта
  Serial.begin (9600);
  //Установка режима работы для пинов
  pinMode(TrigPin, OUTPUT);
  pinMode(EchoPin, INPUT);
}
void loop(){
  // Датчик срабатывает при высоком импульсе 10 или более микросекунд.
  // Для большей точности установим значение LOW на пине Trig:
  digitalWrite(TrigPin, LOW);
  delayMicroseconds(5);
```

```

// Теперь установим высокий уровень на пине Trig
digitalWrite(TrigPin, HIGH);
delayMicroseconds(10);
digitalWrite(TrigPin, LOW);
//Считываем сигнал с датчика: высокий импульс,
//длительность которого равна времени (в микросекундах)
//от отправки пинга до получения его эхо-сигнала от объекта.
duration = pulseIn(EchoPin, HIGH);
cm = (duration/2)/29.1; //преобразуем время в расстояние в см
inches = (duration/2)/74; //преобразуем время в расстояние в дюймах
Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(250);
}

```

Результат

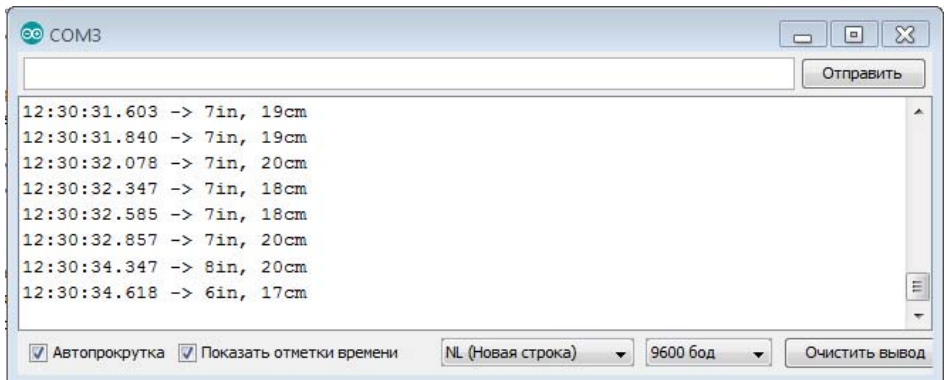


Рис. S3.3. Результаты измерений

1.4. Датчик пламени YG1006

Датчик пламени позволяет фиксировать наличие инфракрасного излучения (открытого пламени) в диапазоне волн $760 \div 1100$ нм в прямой видимости на расстоянии до 1 м (рис. S4.1).

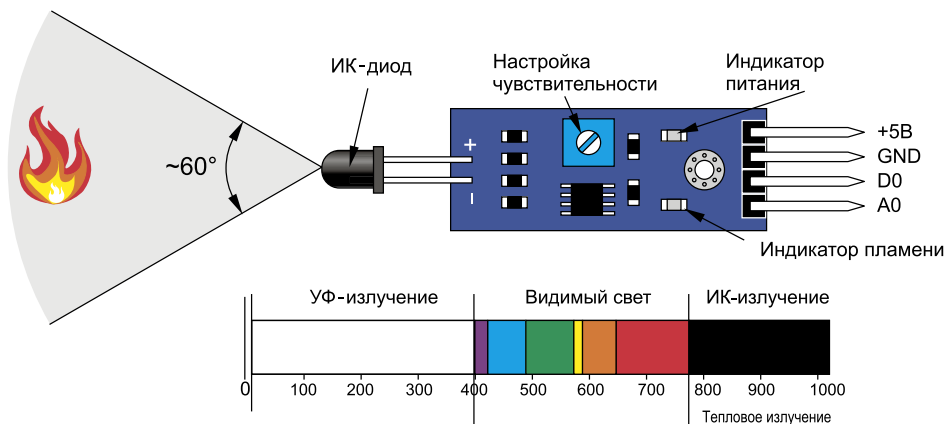


Рис. S4.1. Принцип действия ИК-датчика пламени YG1006

Основные характеристики

Наименование	Значение
Дальность обнаружения пламени, см	20 ÷ 100
Угол обнаружения пламени, град	60
Длина волны, нм	760 ÷ 1100
Пиковая длина волны, нм	940
Напряжение питания, В	3 ÷ 5,5
Потребляемый ток не более, мА	15
Размеры (длина × ширина), мм	36×16

Схема подключения

При подключении датчика только к цифровому выходу (**D0**) фиксируется лишь факт наличия пламени. А при подключении к аналоговому выходу (**A0**) можно оценить и яркость пламени.

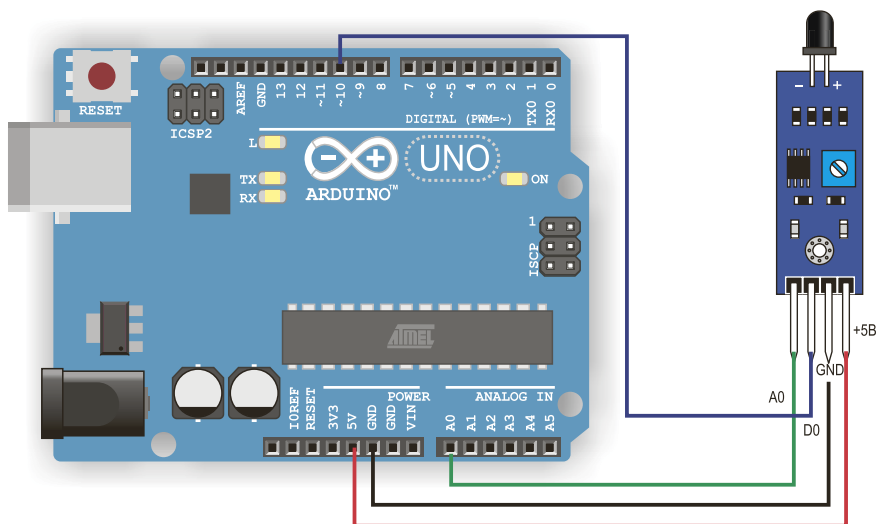


Рис. S4.2. Подключение датчика пламени

Программный код

Листинг S4.1. Подключение датчика пламени

```
#define ledPin 13 //номер пина встроенного светодиода
#define flameDigitalPin 10 //номер пина цифрового входа датчика
#define flameAnalogPin A0 //номер пина аналогового входа датчика
int valueDigital ; //переменная для цифрового значения
float valueAnalog; //переменная для аналогового значения

void setup ()
{
  pinMode (ledPin, OUTPUT) ;
  pinMode (flameDigitalPin, INPUT) ;
  pinMode (flameAnalogPin, INPUT) ;
  Serial.begin(9600);
```

```

}

void loop ()
{
  valueAnalog = analogRead(flameAnalogPin);
  //вывод аналогового значения в Монитор порта
  Serial.println(valueAnalog);

  //чтение цифрового значения
  valueDigital = digitalRead (flameDigitalPin) ;
  if (valueDigital == HIGH) //когда на цифровом входе высокий уровень,
                           //светодиод горит
  {
    digitalWrite (ledPin, HIGH);
    Serial.println("FLAME!");
  }
  else
  {
    digitalWrite (ledPin, LOW);
    Serial.println("no flame");
  }
  delay(1000);
}

```

.....

Результат

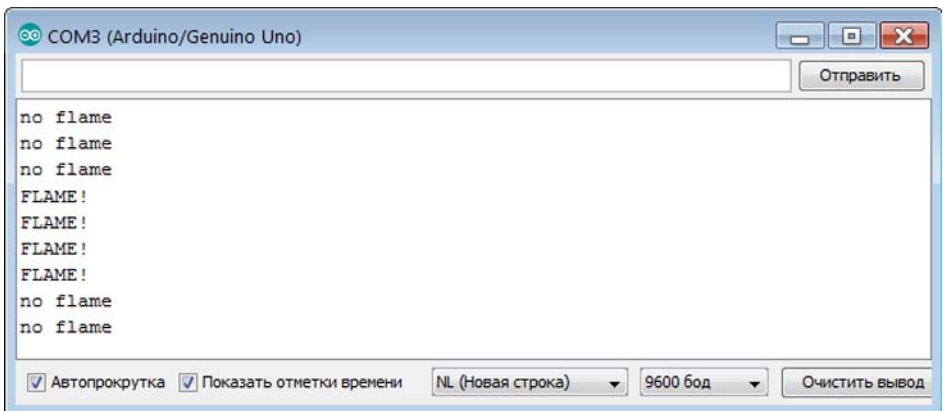


Рис. S4.3. Результаты фиксации пламени

1.5. Датчик влажности почвы YL-38

Модуль датчика состоит из двух частей: контактного щупа и датчика YL-38 (рис. S5.1), в комплекте также идут провода для подключения. Между двумя электродами щупа создается небольшое напряжение. Если почва сухая, то сопротивление большое, а ток небольшой. Если земля увлажняется (например, поливом), то сопротивление уменьшается, а ток немного увеличивается (рис. S5.2).

Внешний вид, назначение контактов

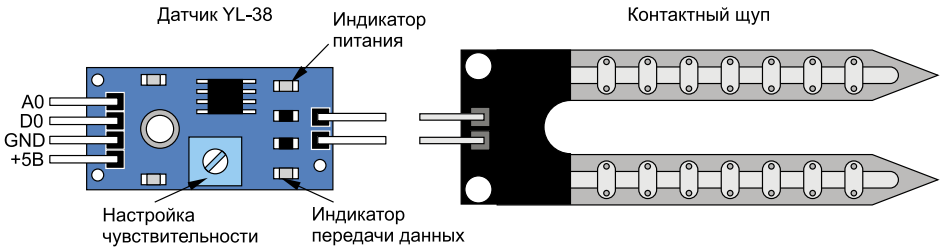


Рис. S5.1. Датчик влажности почвы YL-38



Рис. S5.2. Принцип работы датчика влажности почвы

В качестве датчика влажности почвы можно использовать два оцинкованных гвоздя и проволоку, как показано на рис. S5.3. Когда почва увлажнится, сопротивление между гвоздями уменьшится.

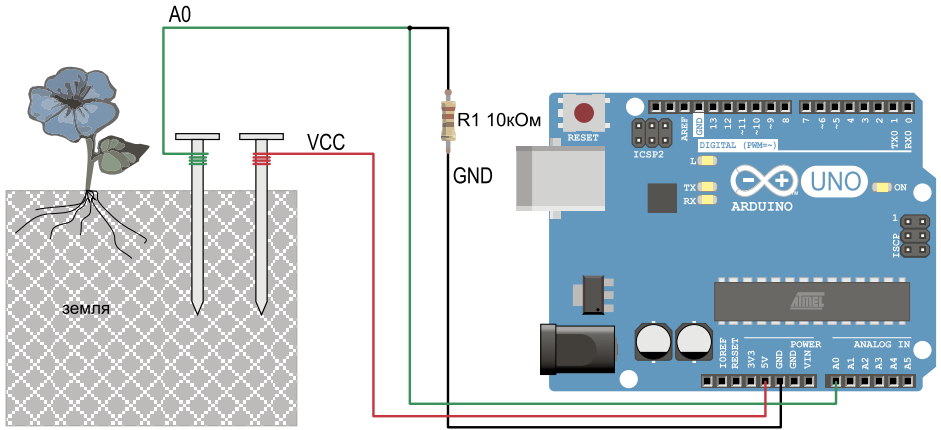


Рис. S5.3. Использование гвоздей в качестве датчика влажности почвы

Основные характеристики

Наименование	Значение
Рабочее напряжение, В	3 ÷ 5
Ток потребления, мА	15
Напряжение цифрового выхода, В	3 ÷ 5
Напряжение аналогового выхода, В	0 ÷ 5

Схема подключения

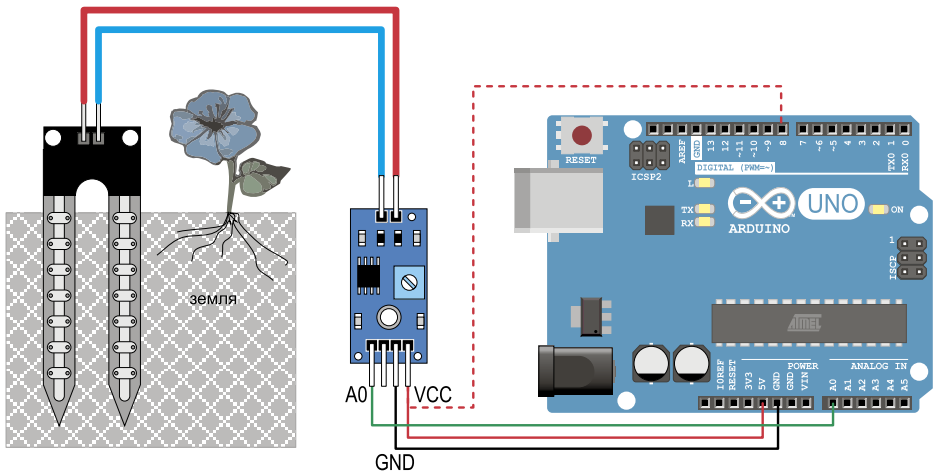


Рис. S5.4. Подключение датчика измерения влажности почвы

Программный код

Листинг S5.1. Подключение датчика влажности почвы

```
#define sensorPin A0 //номер пина влажности почвы
int sensorValue = 0; //переменная значения влажности
void setup() {
    Serial.begin(9600);
}
void loop() {
    sensorValue = analogRead(sensorPin);
    Serial.println(sensorValue);
    delay(100);
}
```

В процессе эксплуатации контактный щуп окисляется, и это происходит достаточно быстро. Чтобы окисление проходило медленнее, можно подключить питание датчика на цифровой вход Arduino и подавать напряжение только на время измерения (см. <https://mxjournal.ru/blog/1319>). В нашем примере контакт питания от модуля влажности (VCC) следует соединить на плате Arduino Uno не с пином **5V**, а с цифровым пином — например, **D8** (на рис. S5.4 показано пунктиром). В этом случае код может быть переписан следующим образом (листинг S5.2) — мы вводим переменную `timing`, в которой будет храниться количество миллисекунд. По умолчанию значение переменной равно 0. В основной части программы проверяем условие: если количество миллисекунд с запуска микроконтроллера минус число, записанное в переменную `timing`, больше, чем записано в переменную `interval` (в минутах), то выполняется функция `get_sensor()`.

Листинг S5.2. Измерение влажности почвы с интервалом

```
#define sensorPin A0 //номер аналогового пина влажности почвы
#define powerPin 8 //номер пина питания датчика влажности почвы

int sensorValue = 0; //переменная значения влажности
unsigned long timing = 0; //переменная для хранения точки отсчета
int interval=60; //интервал измерений в минутах

void setup() {
    Serial.begin(9600);
    pinMode (powerPin, OUTPUT); // питание датчика влажности
```

```

digitalWrite (powerPin, LOW);
digitalWrite (13, LOW);
}
void loop() {
  if (abs(millis() - timing)/1000 > interval*60) { //один раз в час
    get_sensor(); // снимаем показания датчика влажности почвы
    timing = millis();
  }
}
void get_sensor() {
  digitalWrite (powerPin, HIGH); //включаем датчик влажности
  digitalWrite (13, HIGH); //включаем датчик влажности
  delay (3000);
  sensorValue = analogRead(A0); //получение значения влажности
                                //с аналогового вывода датчика
  Serial.println(sensorValue); //контролируем влажность на
                                //Мониторе порта

  delay(100);
  digitalWrite (powerPin, LOW); //выключаем датчик влажности
}

```

1.6. Датчик уровня жидкости

Датчик уровня жидкости предназначен для определения уровня жидкости в различных емкостях. На датчике расположены резисторы, транзистор и чередующиеся оголенные проводящие контакты (рис. S6.1). Чем глубже датчик погружен в воду (бóльшая часть длины контактов находится в воде), тем меньше сопротивление между проводящими контактами.

Внешний вид, назначение контактов

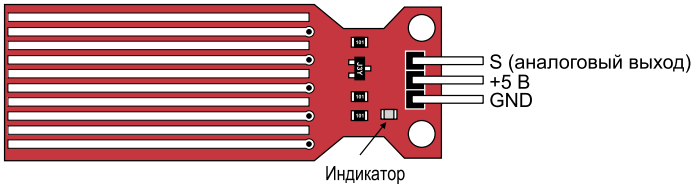


Рис. S6.1. Датчик уровня воды (глубины)

Основные характеристики

Наименование	Значение
Зона обнаружения, мм	16 ÷ 40
Напряжение питания, В	3,3 ÷ 5
Ток потребления, мА	20
Размеры, мм	62×20×8
Рабочая температура, °C	10 ÷ 30

Схема подключения

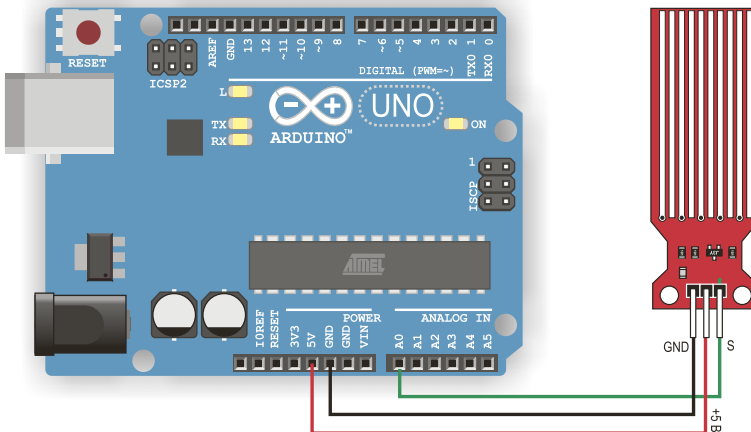


Рис. S6.2. Схема подключения датчика уровня жидкости

Программный код

Листинг S6.1. Измерения уровня жидкости

```
#define aPin A0 // пин для подключения аналогового выхода датчика
int avalue=0; //переменная
int levels[3]={600,500,400}; // значение уровней
void setup(){
  pinMode(aPin, INPUT); // настройка аналогового пина на вход
  Serial.begin(9600); // инициализация последовательного порта
}
void loop(){
  // получение значения с аналогового вывода датчика
  avalue=analogRead(aPin);
  // вывод значения в Монитор последовательного порта Arduino
  Serial.print("avalue=");
  Serial.print(avalue);

  if (avalue>=levels[0]) Serial.println("->MAX");
  if ((avalue>levels[2]&&(avalue<levels[0])) Serial.println("->NORM");
  if (avalue<=levels[2]) Serial.println("->MIN");

  // пауза перед следующим получением значения 1000 мс
  delay(1000);
}
```

Результаты измерений

Значение аналоговых сигналов на аналоговом входе Arduino вы можете определить экспериментальным путем. Они могут оказаться такими, как показано на рис. S6.3.

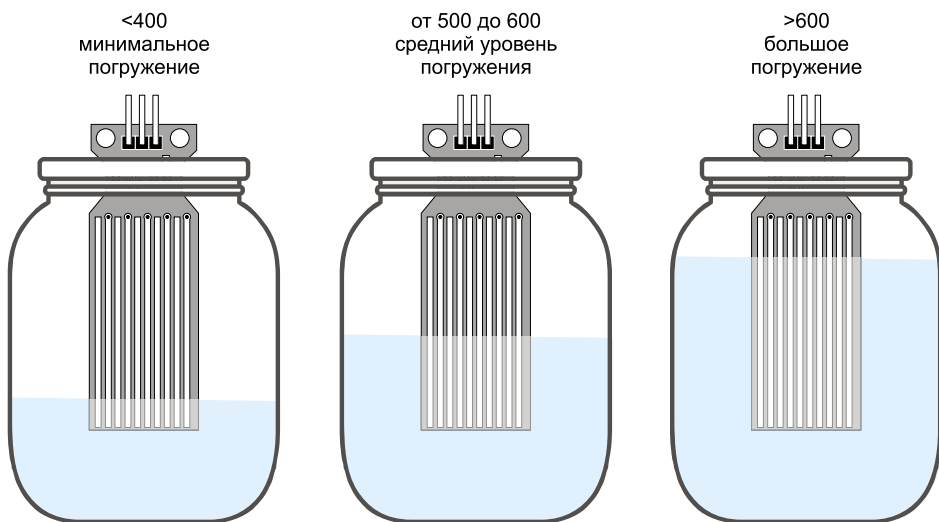


Рис. S6.3. Экспериментальные значения аналоговых сигналов для разных уровней погружения

1.7. Датчик газа MQ-135

Датчик газа MQ-135 предназначен для измерения наличия в окружающем воздухе вредных примесей газа. В качестве чувствительного элемента в датчике служит пластина диоксида олова (SnO_2), который имеет низкую проводимость в чистом воздухе. Когда датчик оказывается в среде с парами токсичных газов, его проводимость возрастает. Датчик MQ-135 очень чувствителен к аммиаку, сульфидам, парам бензола и алкоголя, CO_2 идеально подходит для мониторинга дыма и других вредных примесей в воздухе. На рис. S7.1 показано изменение сопротивления датчика в зависимости от концентрации различных газов в окружающем воздухе в миллионных долях (от общего объема газа).

Внешний вид, назначение контактов

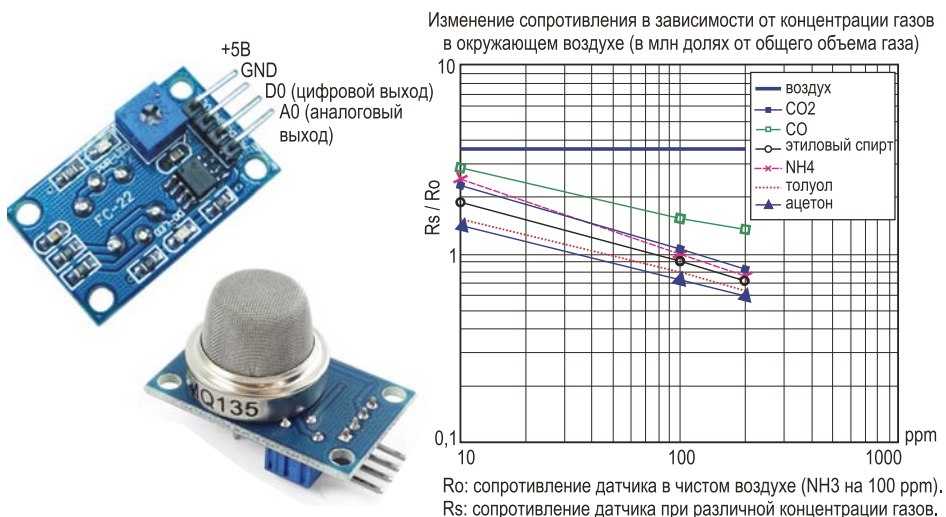


Рис. S7.1. Изменение сопротивления датчика в зависимости от концентрации различных газов в окружающем воздухе

Основные характеристики

Наименование	Значение
Напряжение питания, В	5
Потребляемый ток, мА	160
Рабочая температура, °С	10 ÷ 45
Относительная влажность, %	менее 95
Концентрация кислорода в воздухе (стандартная), %	21
Стандартная температура измерения, °С	20
Влажность, %	65
Диапазон измерений	аммиак: 10 ppm ÷ 300 ppm бензин: 10 ppm ÷ 1000 ppm этиловый спирт: 10 ppm ÷ 300 ppm

Схема подключения

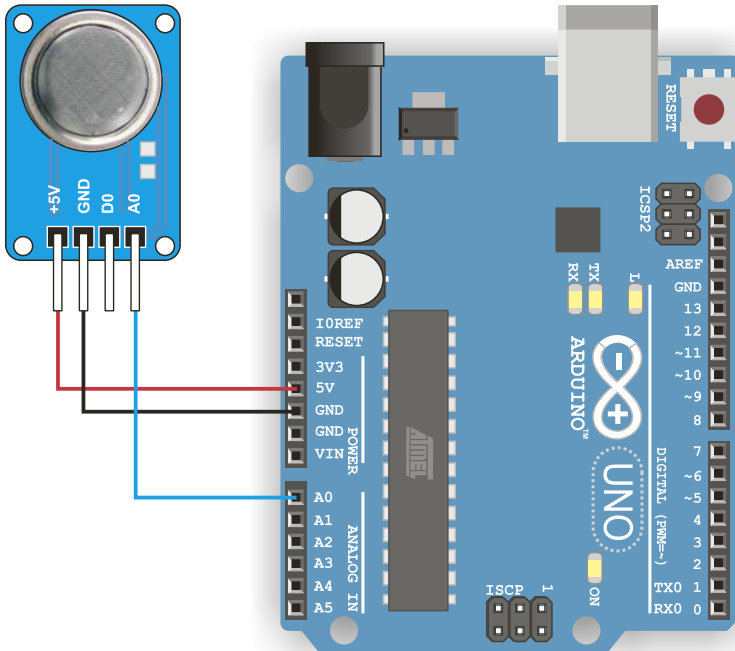


Рис. S7.2. Подключение датчика газа

Программный код

Листинг S7.1. Измерение концентрации газов с помощью датчика MQ-135

```
#define MQPin A0           //пин, к которому подключен датчик газа
#define ledPin 13         //пин встроенного светодиода
int sensorValue = 0;     //переменная для хранения значений

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  Serial.println("MQ135 Test" ); //Посылаем текст в Монитор порта
}
```



```
void loop() {
  // считываем значения с датчика
  sensorValue = analogRead(MQPin);
  if (sensorValue >= 400)
    // и если превышен заданный порог,
  {
    digitalWrite(ledPin, HIGH); // то включаем светодиод,
  }
  else // а если нет...
  {
    digitalWrite(ledPin, LOW); // то выключаем
  }
  Serial.print("MQ135 value= ");
  // Для отслеживания данных с датчиков
  // транслируем их в Монитор порта
  Serial.println(sensorValue);
  delay(1000);
}
```

Результат

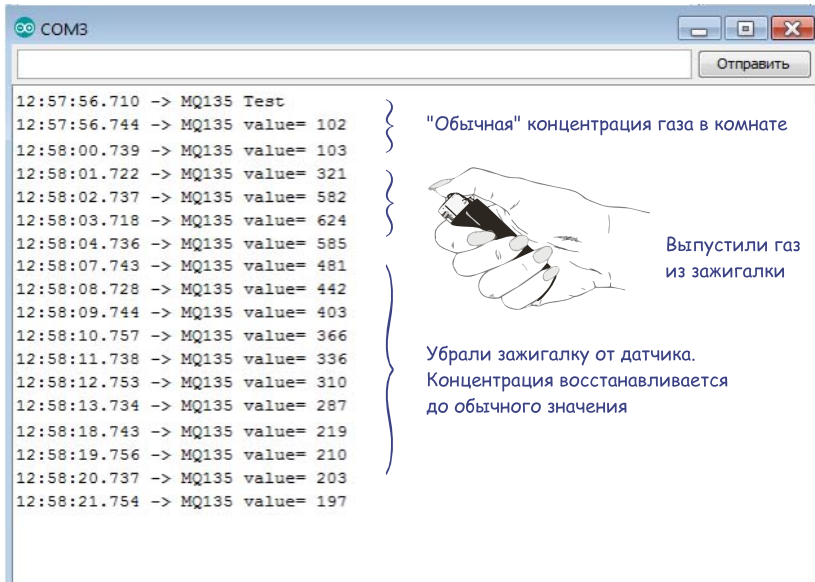


Рис. S7.3. Результаты измерений

1.8. Датчик звука

Датчик звука, как следует из названия, предназначен для обнаружения звука (фиксирует появление громкого звука). На рис. S8.1 показаны наиболее широко применяемые в проектах Arduino датчики звука.

Внешний вид, назначение контактов

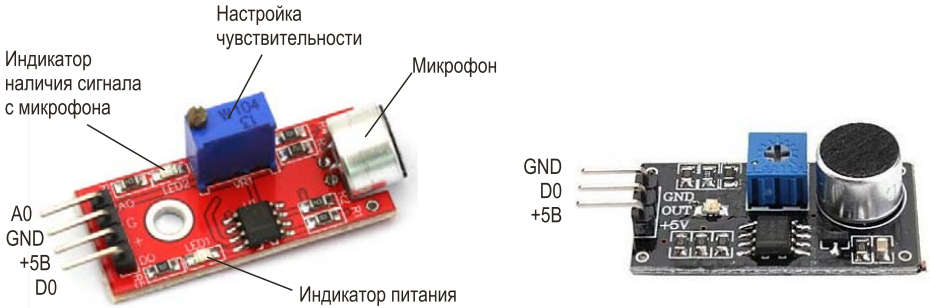


Рис. S8.1. Применяемые в проектах Arduino датчики звука

Схема подключения

Работая с датчиком звука, можно использовать как цифровой, так и аналоговый его выходы (рис. S8.2). Аналоговый выход выдает значение сигнала микрофона, а цифровой выход передает 1, если сигнал превысит пороговое значение, и 0 — в противном случае. Пороговое значение можно настроить с помощью потенциометра, расположенного на плате датчика. Можно подключить одновременно и два выхода (например, для настройки порогового значения).

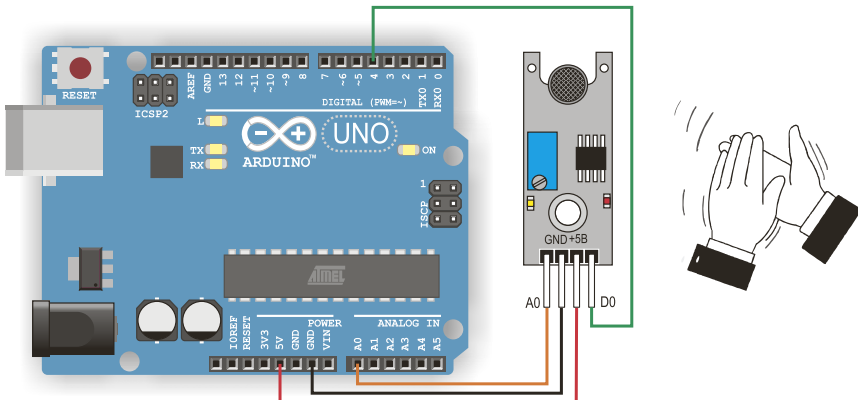


Рис. S8.2. Схема подключения датчика звука

Программный код

Листинг S8.1. Измерение громкости с помощью датчика звука

```
#define soundAnalogPin A0 // пин, к которому аналоговый выход
#define soundDigitalPin 4 // пин, к которому цифровой выход
int analogVal=0; // Объявляем переменные для хранения значений
int digitalVal=0; //с датчика и задаем ее начальное значение 0

void setup()
{
  Serial.begin(9600); // Открываем Монитор порта
  pinMode(soundAnalogPin, INPUT); //Настройка аналогового пина на вход
  pinMode(soundDigitalPin, INPUT); // Настройка цифрового пина на вход
}

void loop()
{
  //присваиваем переменной аналоговое значение
  analogVal =analogRead(soundAnalogPin);
  //присваиваем переменной цифровое значение
  digitalVal=digitalRead(soundDigitalPin);

  //Выводим полученные с датчика значения
  Serial.print("Sound value A0: "); //
  Serial.print(analogVal,DEC);
  Serial.print(" D0: ");
  Serial.println(digitalVal,DEC);

  delay(100); //задаем паузу
}
```

Результат

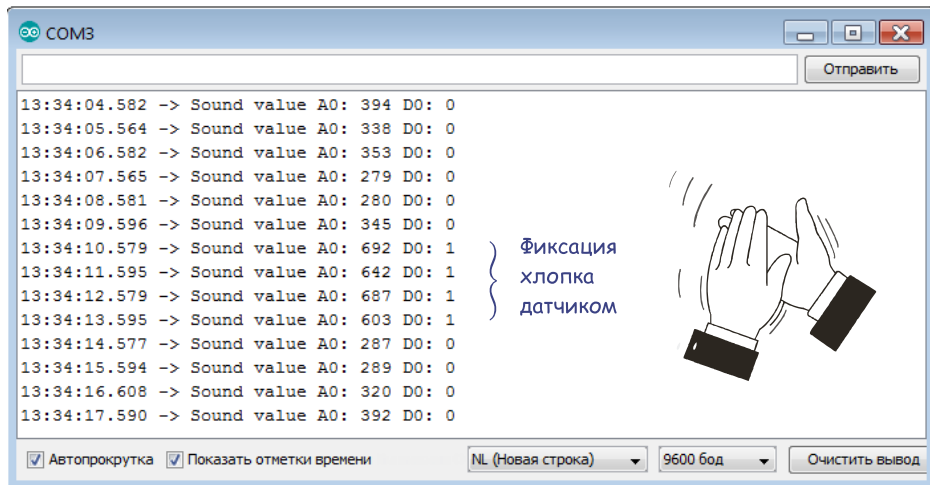


Рис. S8.3. Фиксация хлопка в ладоши на Мониторе последовательного порта

1.9. Потенциометр

Потенциометром называется регулируемый делитель напряжения, который — в отличие от реостата — служит для регулировки напряжения при почти неизменном токе. Снимаемое с подвижного отводного контакта потенциометра напряжение (рис. S9.1) в зависимости от текущего положения подвижного контакта может изменяться от нуля до максимального значения, равного приложенному к потенциометру напряжению.

Внешний вид, назначение контактов

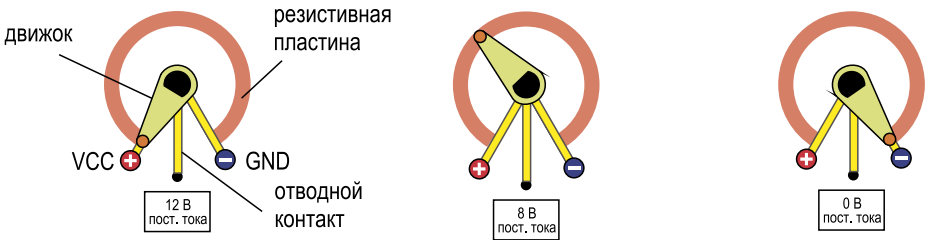



Рис. S9.1. Напряжение на потенциометре

	Подробную информацию о чтении данных с потенциометра можно найти в прилагаемой к набору книге Дж. Блума «Изучаем Arduino: инструменты и методы технического волшебства» (см. раздел 3.5 «Чтение данных с потенциометра»).	стр. 67
---	---	---------

1.10. Модуль фоторезистора KY-018

Фоторезистор обычно представляет собой компонент в виде диска с двумя выводами. Когда освещенность поверхности этого диска увеличивается, сопротивление между выводами уменьшается. Некоторые фоторезисторы в темноте обладают сопротивлением до 10 МОм, а при ярком освещении у некоторых из них сопротивление может снижаться до 500 Ом. Вы всегда сможете определить эти параметры для вашего компонента опытным путем.



Подробную информацию о подключении фоторезистора к плате Arduino вы можете найти в прилагаемой к набору книге Дж. Блума «Изучаем Arduino: инструменты и методы технического волшебства» (см. раздел 3.8 «Использование переменных резисторов для создания собственных аналоговых датчиков».

стр. 75

Внешний вид, назначение контактов

Электроды из проводящего состава, осажденного на сульфиде кадмия



Модуль фоторезистора KY-018

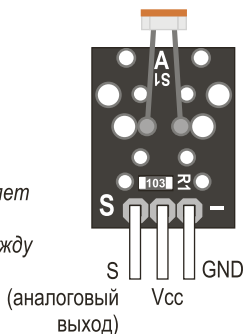


Рис. S10.1. Устройство фоторезистора (слева) и модуль KY-018 (справа)

Схема подключения

Для удобного подключения фоторезисторов существуют специальные платы (с отверстиями для крепления), на которых интегрирован фоторезистор и понижающий резистор (рис. S10.2, справа). Кроме того, там может быть размещен триммер настройки. Модуль может иметь кроме аналогового выхода дополнительный цифровой, который будет фиксировать факт изменения освещенности выше «настроенного порога».

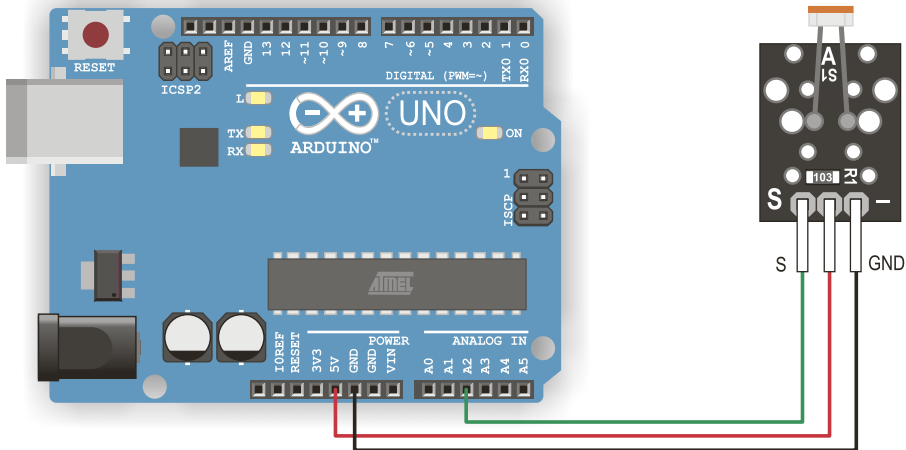


Рис. S10.2. Подключение платы с фоторезистором

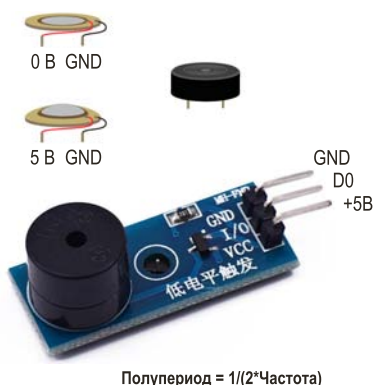
2. Модули

2.1. Активный пьезоэлектрический зуммер 5 В

С помощью модуля зуммера можно генерировать звуковые сигналы и даже простые мелодии. Принцип действия зуммера основан на пьезоэлектрическом эффекте, согласно которому при подаче электричества на зуммер он начинает деформироваться. При этом производятся удары о металлическую пластинку, которая и издает звук определенной частоты.

Существуют активные и пассивные зуммеры. Главное различие их состоит в том, что активный можно только включить и выключить, подав напряжение на его контакты. Для пассивного зуммера (рис. М1.1) кроме питания требуется источник, который задаст параметры звукового сигнала. В качестве такого источника может выступать плата Arduino. Для генерации мелодий следует подключать цифровой пин зуммера к пину Arduino, который поддерживает ШИМ (~D5, ~D6, ~D9, ~D10, ~D11).

Внешний вид, назначение контактов



Полупериод = $1/(2 \cdot \text{Частота})$

Ноты	Частота, Гц		Полупериод, с	
C1 (До)	261	277	1916	1805
D1 (Ре)	294	311	1701	1608
E1 (Ми)	329		1520	
F1 (Фа)	349		1433	
G1 (Соль)	392	370	1276	1351
A1 (Ля)	440	415	1136	1205
B1 (Си)	493	466	1014	1073

C2 (До)	523		956	
D2 (Ре)	587	554	852	903
E2 (Ми)	659	622	759	804
F2 (Фа)	698		716	
G2 (Соль)	784	740	638	676
A2 (Ля)	880	831	568	602
B2 (Си)	988	932	506	536

Рис. М1.1. Пассивный зуммер

Основные характеристики

Наименование	Значение
Напряжение, В	5
Максимальный ток, мА	32
Минимальный уровень звука на расстоянии 10 см, дБ	85
Диапазон частот, Гц	2300 ± 300

Схема подключения

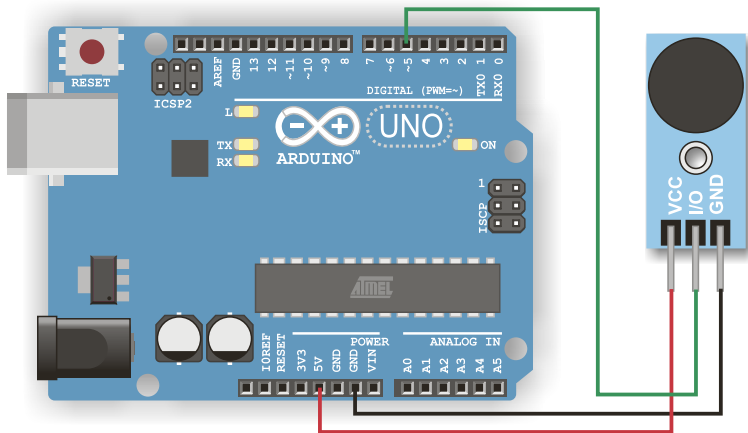


Рис. М1.2. Подключение зуммера

Программный код

Листинг М1.1. Подключение зуммера

```
#define SoundPin 5 // пин пьезоизлучателя
int DelaySound = 1000; //Пауза. 1000 миллисекунд=1 секунда
void setup(){
}
void loop(){
    tone(SoundPin, 1915); //Воспроизводим сигнал с частотой тона
                          //1/(2*1915)=261 Гц (нота До)
    delay(DelaySound); //Длительность воспроизведения сигнала 1 с
    tone(SoundPin, 1700);
    delay(DelaySound);
    tone(SoundPin, 1519);
    delay(DelaySound);
    tone(SoundPin, 1432);
    delay(DelaySound);
    tone(SoundPin, 1275);
    delay(DelaySound);
}
```

```
tone(SoundPin, 1136);  
delay(DelaySound);  
tone(SoundPin, 1014);  
delay(DelaySound);  
noTone(7); // Выключаем звук  
}
```

Пояснение

Функция `tone()` генерирует на порту входа/выхода сигнал — прямоугольную «волну» заданной частоты и с 50%-ным рабочим циклом. Длительность может быть задана параметром `delay`, в противном случае сигнал генерируется, пока не будет вызвана функция `noTone()`.

Вы можете использовать активный пьезоэлектрический зуммер (пьезоэлемент) для воспроизведения различных мелодий. При этом на него следует подавать сигнал определенной частоты и длительности.

Примечание

Подробную информацию о соответствии частоты сигнала воспроизводимым музыкальным нотам, а также примеры программ, можно найти на сайте разработчика Arduino: <http://www.arduino.cc/en/Tutorial/Melody>.

Добавив в схему подключения пьезоэлемента несколько кнопок, можно сделать небольшой электронно-клавишный инструмент, используя несколько пьезоэлементов — собрать целый оркестр, а с помощью фоторезистора — создать «световой терменвокс».

Световой терменвокс

Соберем простейший имитатор музыкального инструмента терменвокса (рис. М1.3). Только играя на нашем терменвоксе, исполнитель будет подносить руки не к антенне (как при игре на классическом терменвоксе), а к фоторезистору.

Примечание

Сопrotивление фоторезисторов уменьшается под воздействием света и увеличивается в темноте. Фоторезисторы просты в использовании, но слишком медленно реагируют на изменение уровня освещенности и имеют весьма низкую точность. Как правило, сопротивление фоторезисторов может варьироваться от 50 Ом при дневном освещении до более чем 10 МОм в темноте.

Схема подключения

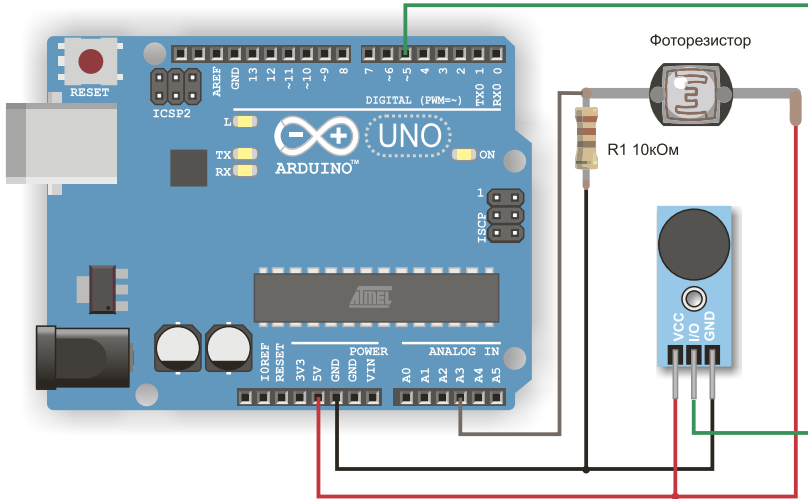


Рис. М1.3. Световой терменвокс

Программный код

Листинг М1.2. Световой терменвокс

```
#define buzzerPin 5 //номер пина зуммера
#define photoPin A3 //номер пина фоторезистора
int aVal; //Значение на фоторезисторе
int sMax=1023; //максимальное значение фоторезистора
int buzzerFreq; //частота звука
const long BUZZ_FREQ_MAX = 2500; //задаем макс. частоту излучения

void setup(){
  // Объявляем пин, к которому подключен зуммер как выход
  pinMode(buzzerPin,OUTPUT);
  Serial.begin(9600);
}

void loop(){
  aVal=analogRead(photoPin); //считываем значение с фоторезистора
```

```

Serial.print("aVal:");
Serial.println(aVal);
// задаем частоту излучения // пьезоизлучателя
buzzerFreq = (aVal * BUZZ_FREQ_MAX)/sMax;
buzz(buzzerPin, buzzerFreq, 10);
}

//подпрограмма генерации звука buzz
void buzz(int targetPin, long frequency, long length) {
    long delayValue = 1000000/frequency/2;
    long numCycles = frequency * length/ 1000;
    for (long i=0; i < numCycles; i++){
        digitalWrite(targetPin,HIGH);
        delayMicroseconds(delayValue);
        digitalWrite(targetPin,LOW);
        delayMicroseconds(delayValue);
    }
}

```

Всё — простейший терменвокс готов! Теперь попробуйте, изменяя освещенность фоторезистора, создать звуковую композицию.

2.2. RFID-модуль RC522

Радиочастотная идентификация RFID (от англ. Radio Frequency IDentification) представляет собой бесконтактную технологию, которая широко используется в системах контроля доступа, аутентификации персонала, в логистике, в розничной торговле и др. Для идентификации людей наиболее популярным является формат бесконтактной пластиковой карточки, по размеру совпадающей с банковской. Такую карточку для запроса доступа, как правило, нужно осознанно подносить к считывающему устройству на расстояние порядка 10 см.

Внешний вид, назначение контактов

Система RFID состоит из двух основных компонентов: RFID-метки (тега), находящейся у объекта, который мы хотим идентифицировать, и RFID-считывателя (рис. M2.1).

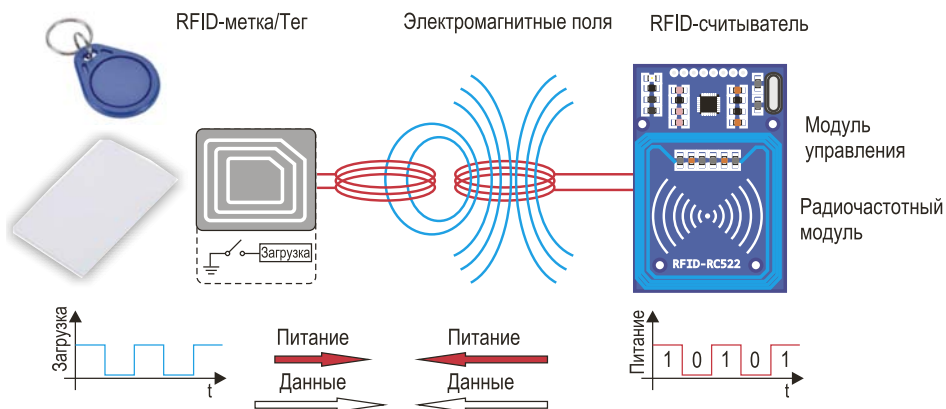


Рис. M2.1. RFID-модуль RC522

RFID-считыватель состоит из радиочастотного модуля, блока управления и антенной катушки, которая генерирует высокочастотное электромагнитное поле. RFID-метка является пассивным компонентом, в состав которого входит антенна и электронный микрочип. Когда микрочип приближается к электромагнитному полю считывателя на расстояние около 2 см, в катушке его антенны благодаря индукции генерируется напряжение, которое включает микрочип. После этого RFID-метка может обмениваться информацией с RFID-считывателем.

В проектах мы воспользуемся RFID-считывателем RC522 на базе популярной микросхемы MFRC522 производства фирмы NXP Semiconductors. В комплект поставки считывателя RC522 входят две RFID-метки типа Mifare Classic с памятью объемом 1 Кбайт: карточка и брелок. Каждая метка имеет уникальный идентификационный ключ (UID).

Основные характеристики

Наименование	Значение
Напряжение, В	3,3
Потребляемый ток в активном состоянии, мА	13 ÷ 26
Потребляемый ток в состоянии ожидания мА	10 ÷ 13
Рабочая частота, МГц	13,56
Поддерживаемые типы карт	MIFARE S50, MIFARE S70, MIFARE UltraLight, MIFARE Pro, MIFARE DESfire
Размеры, мм	40×60

Схема подключения

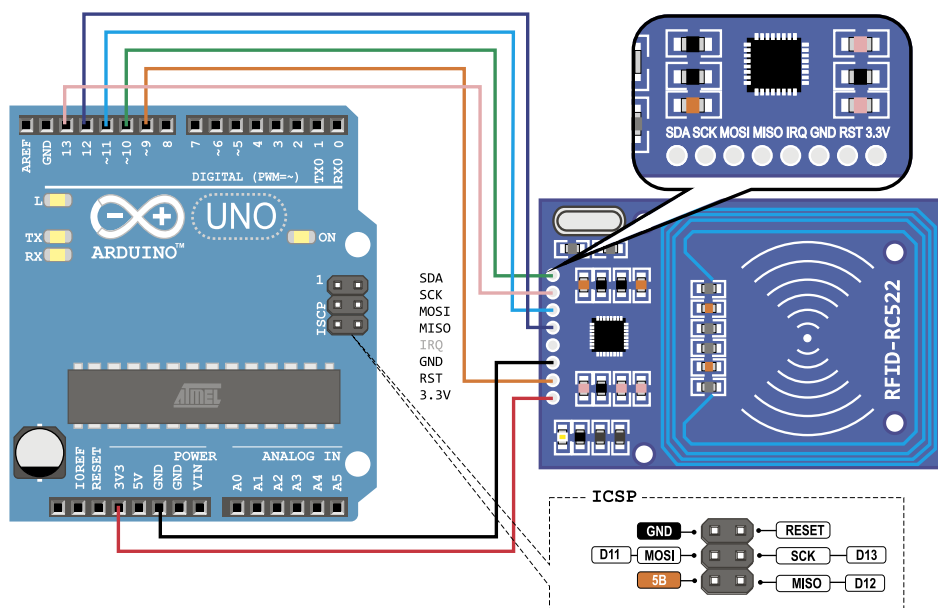


Рис. М2.2. Схема подключения RFID-модуля RC522 к Arduino Uno

Примечание

Для подключения модуля RFID вы также можете использовать контактную группу ICSP на плате Arduino Uno, подключив туда контакты **MISO** (ICSP-1), **SCK** (ICSP-3) и **MOSI** (ICSP-4).

Получаем UID метки RFID

1. Загрузите специализированную библиотеку MFRC522 (<https://github.com/miguelbalboa/rfid/archive/master.zip>).
2. Подключите библиотеку в среду разработки Arduino IDE: **Скетч | Подключить библиотеку | Добавить .ZIP библиотеку...**
3. Загрузите скетч **DumpInfo** из папки примеров библиотеки MFRC522: **Файл | Примеры | MFRC522 | DumpInfo**.
4. Запустите Монитор порта.

Программа считывает информацию с поднесенной RFID-метки и выводит ее в терминал. Для RFID-меток из комплекта RC522 она будет иметь примерно следующий вид:

Card UID: **7E 56 37 D5**

Card SAK: 08

PICC type: MIFARE 1KB

- первая строка — уникальный идентификатор RFID-метки (UID) для этого типа меток (4 байта);
- вторая строка — поле SAK (Select Acknowledge), содержащее информацию о типе метки и производителе. Значение 08 означает, что это метка типа Mifare Classic 1K производства NXP;
- третья строка — расшифровка типа метки, извлеченная из поля SAK: Mifare 1KB.

Внимание!

Запишите уникальный идентификатор (UID) вашей RFID-метки! Он понадобится вам в следующем примере.

Программный код

Загрузите скетч из листинга M2.1. При поднесении метки к считывателю на мониторе порта будет появляться надпись **Access granted**, а на плате загораться встроенный светодиод.

Листинг M2.1. Подключение модуля RFID

```
#include <SPI.h>
#include <MFRC522.h>

// Настраиваем пины SS и RST для нашей платы - Arduino Uno
#define SS_PIN 10
#define RST_PIN 9
#define LED13 13 //Встроенный светодиод. Зажигается когда доступ разрешен
```

```

MFRC522 mfrc522(SS_PIN, RST_PIN); // Создаем объект MFRC522
void setup() {
    pinMode(LED13, OUTPUT);
    Serial.begin(9600);

    SPI.begin(); // Инициализируем интерфейс SPI
    mfrc522.PCD_Init(); // Инициализируем MFRC522
    Serial.println("Approximate your card to the reader...");
    Serial.println();
}
void loop(){
    if (!mfrc522.PICC_IsNewCardPresent()) // Ожидаем RFID-метку
        return;
    // При обнаружении RFID-метки пытаемся считать ее данные
    if (!mfrc522.PICC_ReadCardSerial())
        return;
    // Извлекаем уникальный идентификатор RFID-метки (UID)
    String content = "";
    // Преобразуем поле UID в строку типа "XX XX XX XX"
    for (byte i = 0; i < mfrc522.uid.size; i++)
    {
        content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
        content.concat(String(mfrc522.uid.uidByte[i], HEX));
    }
    content.toUpperCase();
    // Выводим UID в терминал
    Serial.print("Tag UID:");
    Serial.println(content);

    // Сравниваем UID с заданным
    if (content.substring(1) == "7E 56 37 D5") // Укажите здесь UID
                                                //RFID-метки, которой разрешен доступ
    {
        // Если UID совпал с заданным – доступ разрешен
        Serial.println("Access granted");
        Serial.println();
    }
    // Включаем встроенный на плате светодиод на 3 секунды
    digitalWrite(LED13, HIGH);
    delay(3000);
    digitalWrite(LED13, LOW);
}
else // Если UID не совпал с заданным – доступ запрещен
{
    Serial.println("Access denied");
    Serial.println();
}
}

```


2.3. Модуль реле

Реле позволяет электрическому сигналу или импульсу включать (или выключать) электрический ток. Для управления реле используется низкое напряжение или слабый ток, чтобы с его помощью управлять высоким напряжением и/или сильным током.

В электромеханическом реле ток протекает через катушку, которая действует как электромагнит, замыкающий (или размыкающий) контакты силовой части реле. Кроме электромеханических существуют и твердотельные реле, использующие твердотельную электронику (без катушки и механических движущихся частей).

Внешний вид, назначение контактов

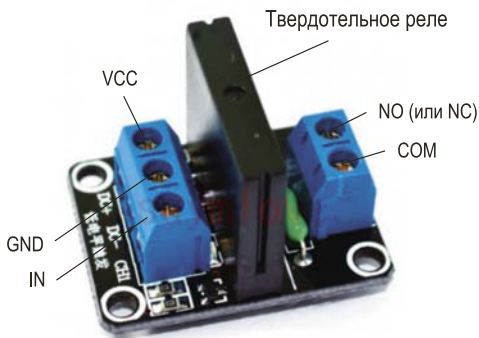
Для удобства управления и подключения к Arduino реле устанавливаются на платы, где, кроме самого реле, расположены контакты для подключения нагрузки и другие элементы (рис. М3.1). На одной плате могут размещаться несколько реле.

Разъёмы для подключения реле к внешней нагрузке:

NO — нормально разомкнутый (Normally Open)

NC — нормально замкнутый (Normally Closed)

COM — общий (Common)



Контакты для подключения модуля реле к Arduino:

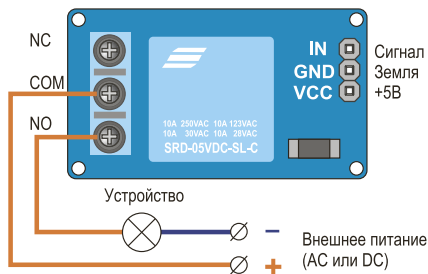
VCC — питание +5 В

GND — земля

IN1 — управление реле № 1

IN2 — управление реле № 2

Режим NC (Normal Close)



Режим NO (Normal Open)

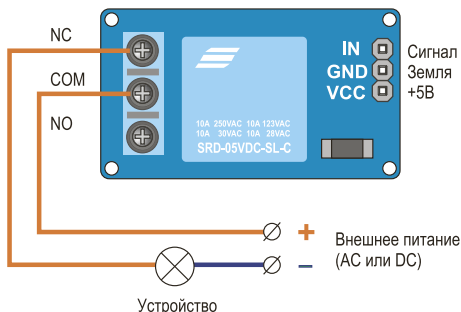


Рис. М3.1. Назначение контактов одноканальных модулей реле

Основные характеристики

Наименование	Значение
Рабочее напряжение, В	5
Потребляемый ток при переключении контактов, мА	5
Потребляемый ток в состоянии ожидания, мА	10 ÷ 13
Рабочая частота, МГц	13,56
Размеры (L×W×H), мм	50×26×18,5
Максимальная нагрузка	AC 250 В/10 А DC 30 В/10 А

Предупреждение

Модуль реле, входящий в набор, рассчитан на коммутирование небольших нагрузок. Вы можете подключать к нему бытовые приборы с рабочим напряжением 220 В, но нагрузка не должна превышать 3-х ампер (мощность до 660 Вт). Имейте в виду, что мощность утюга составляет 1000÷1500 Вт, и поэтому такое реле не годится для создания устройства для включения/выключения утюга. Существуют специальные модули реле, рассчитанные на большие мощности.

Схема подключения

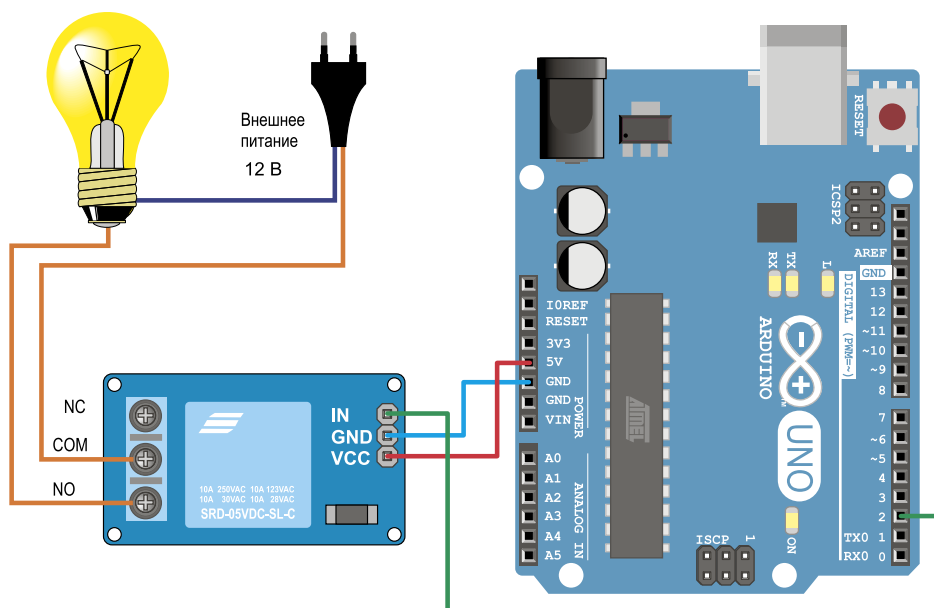


Рис. М3.2. Схема подключения реле

Программный код

Тестовая программа для включения и выключения лампочки через каждые 2 секунды приведена в листинге М3.1.

Листинг М3.1. Включение лампочки с помощью реле

```
#define relayPin 2 //номер пина для управления реле
void setup() {
  pinMode(relayPin, OUTPUT); //настройка пина реле на выход
}
void loop() {
  digitalWrite(relayPin, HIGH); // замыкаем реле
  delay(3000); // ждем 3 секунды
  digitalWrite(relayPin, LOW); // размыкаем реле
  delay(3000); // ждем 3 секунды
}
```

«Умный горшок»

С помощью модуля реле можно создать систему автоматического полива растений в цветочном горшке. Для этого, кроме модуля реле, мы задействуем микронасос, помещенный в банку с водой, и датчик влажности почвы (рис. М3.3). Как только влажность почвы опускается ниже заданных значений, Arduino включает микронасос и выключает его, когда почва становится снова достаточно влажной.

Систему можно усложнить, добавив датчик глубины, чтобы сигнализировать об окончании воды в банке и недопустить сгорания микронасоса.

Схема подключения

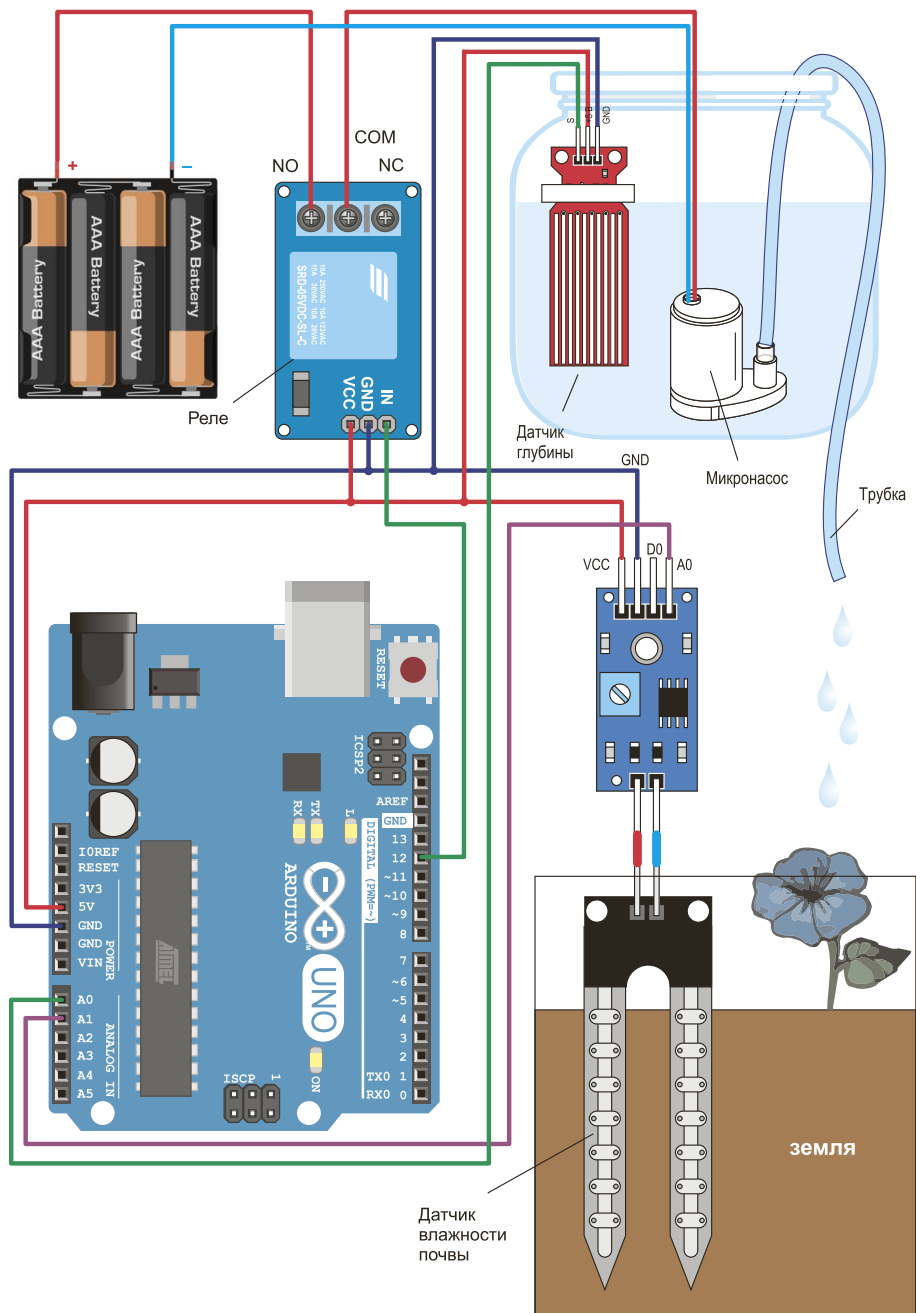


Рис. М3.3. Система автоматического полива цветочного горшка

Программный код

Листинг М3.2. Система автоматического полива цветочного горшка

```
//определения
// пин аналогового выхода датчика уровня воды
    #define pinWaterLevel A0
//пин аналогового выхода датчика влажности почвы
    #define pinSoilMoisture A1
//пин реле для управления насосом
    #define pinRelayPump 12

//константы
    const int delayPumpBefore=2;    //время полива (в секундах)
    const int delayPumpAfter=30;    //время после полива, чтобы
//земля пропиталась (в секундах)
    const int minMoisture=600;      //минимальный порог влажности почвы

// переменные
    int aLevel = 0; // значение датчика уровня воды
    int aMoisture = 0; // состояние датчика влажности почвы
    int levels[3]={600,500,400}; //массив значений уровней воды

//установки
void setup() {
    //объявляем пин реле для включения насоса как выход:
    pinMode(pinRelayPump, OUTPUT);
    //объявляем пины датчиков глубины и влажности почвы как входы:
    pinMode(pinWaterLevel, INPUT);
    pinMode(pinSoilMoisture, INPUT);
    Serial.begin(9600);
}

void loop() {
    // считываем значение датчика уровня воды
    aLevel=analogRead(pinWaterLevel);
```

```

// считываем состояния датчика влажности почвы
aMoisture = analogRead (pinSoilMoisture);
Serial.println(aMoisture); //выводим для тестирования
delay(100);

// если почва сухая, и вода в банке есть, то включаем полив
if ((aMoisture >minMoisture)&&(aLevel>levels[2])) {
    digitalWrite(pinRelayPump, HIGH); //включаем насос
    delay(delayPumpBefore*1000);      //задержка на полив
    digitalWrite(pinRelayPump, LOW);  //выключаем насос
    delay(delayPumpAfter*1000); //задержка на слив воды из
//шланга после выключения насоса
}
else {
    digitalWrite(pinRelayPump, LOW);
}
}
.....

```

Для адаптации программы к конкретному цветочному горшку надо произвести небольшую «тонкую настройку» системы:

- во-первых, следует правильно установить время полива (`delayPumpBefore`), которое определяется паузой между включением и выключением насоса. Чем больше горшок, тем больше должна быть пауза и, как следствие, время полива;
- во-вторых, надо установить правильное время после полива (`delayPumpAfter`), чтобы земля успела пропитаться, и система не включила повторный полив. Трубку подачи воды при этом удобно разместить рядом с датчиком влажности почвы, чтобы земля в районе датчика сразу пропитывалась.

2.4. 8-разрядный расширитель портов PCF8574

Если вы исчерпали для подключения внешних устройств (датчиков, модулей, светодиодов и др.) все выводы Arduino (для Arduino Uno это 6 аналоговых и 14 цифровых), то модуль расширителя портов (рис. М4.1) поможет вам добавить дополнительно 8 контактов (P0 ... P7). Эти контакты можно использовать как в качестве входов (например, для чтения сигнала с датчика), так и для выходов (например, для включения светодиодов). Модули расширителя портов работают на шине I²C, и, если подключить в проекте последовательно 8 таких устройств, вы соответственно получите дополнительно $8 \times 8 = 64$ контакта для ввода и вывода.

Внешний вид, назначение контактов

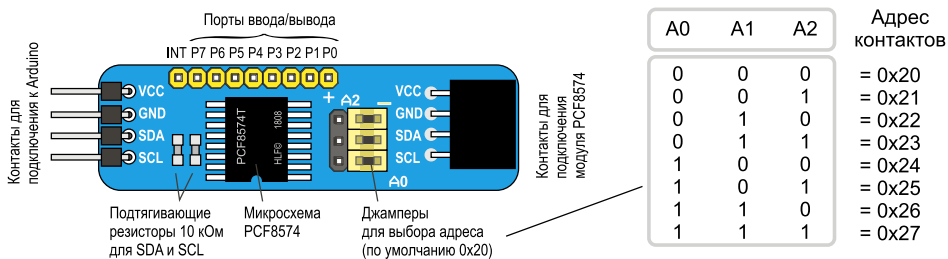


Рис. М4.1. 8-разрядный расширитель портов PCF8574

Основные характеристики

Наименование	Значение
Рабочий режим питания, В	от 2,5 до 6 В
Низкий ток покоя, мА	максимум 10 мА
Адресация	на 3 вывода аппаратных адресов для использования до 8 устройств (до 16 устройств при использовании PCF8574A)

Схема подключения

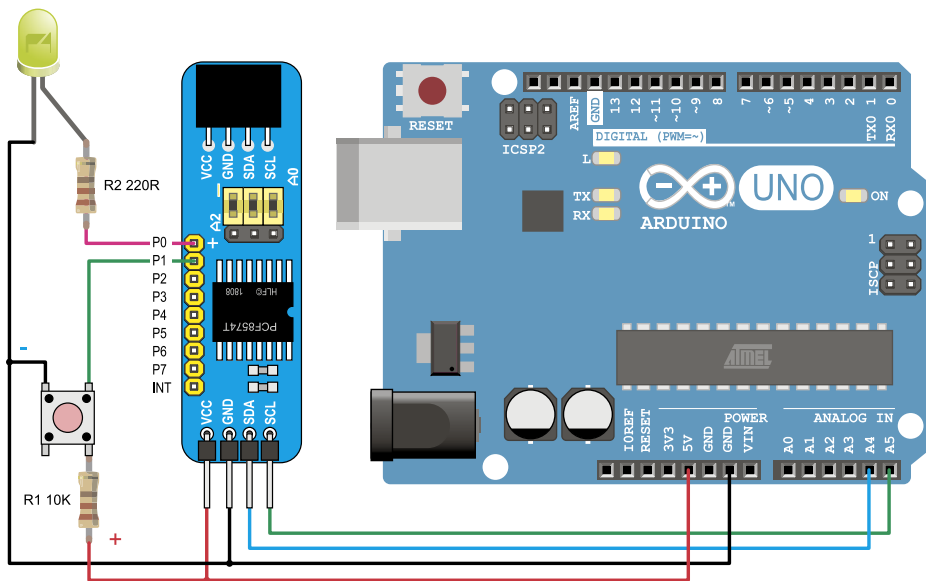


Рис. М4.2. Схема подключения кнопки и светодиода

Программный код

Скачайте по ссылке https://github.com/skywodd/pcf8574_arduino_library библиотеку PCF8574/PCF8575 Arduino library для работы Arduino с расширителем PCF8574.

Скачав библиотеку, выберите ее и установите с помощью команды **Скетч | Подключить библиотеку | Добавить ZIP библиотеку...**

1. Загрузите скетч из листинга М4.1.

Листинг М4.1. Подключение расширителя портов PCF8574

```
#include <Wire.h>
#include "PCF8574.h" //Библиотека для расширителя портов.

PCF8574 expander; //создадим объект класса PCF8574 с расширителем

#define pinLed 0 //Номер пина светодиода на расширителе (P0)
#define pinButton 1 //Номер пина кнопки на расширителе (P1)
```



```

void setup(){
    expander.begin(0x20);    / Инициализация расширителя по адресу 0x20

    expander.pinMode(pinLed,OUTPUT); //Настройка пина светодиода
                                   //(на расширителе) на выход
    expander.pinMode(pinButton,INPUT_PULLUP); // Настройка пина кнопки
                                   //(на расширителе) на вход
}

void loop(){
    if (expander.digitalRead(pinButton)==HIGH)
        expander.digitalWrite(pinLed, HIGH);
    else
        expander.digitalWrite(pinLed,LOW);
}

```

Примечание

Если вы используете для управления проектом со смартфона платформу RemoteXY, прокомментируйте в файле PCF8574.h библиотеки PCF8574 строку:

```
#define PCF8574_INTERRUPT_SUPPORT
```

Это строка номер 36. Тем самым вы отключите у расширителя поддержку прерываний, но зато предотвратите конфликт приложения RemoteXY и расширителя PCF8574, которые используют одинаковый вектор прерываний.

2.5. Модуль BLE Bluetooth HM-10/11

Модули Bluetooth оснащены последовательным портом, способным взаимодействовать с платой Arduino через контакты RX (прием) и TX (передача), и приемопередатчиком, использующим протокол беспроводной связи Bluetooth для взаимодействия «по воздуху» с другими устройствами, который понимают этот протокол (например, со смартфонами). Такой приемопередатчик функционирует как модем, преобразовывая сигналы протокола Bluetooth в обычный асинхронный последовательный протокол и обратно.

Существуют различные протоколы Bluetooth. Модули HM-10 и HM-11 поддерживают протокол Bluetooth 4.0, что дает возможность их подключения к смартфонам и на ОС Android, и на iOS (iPhone) — в отличие от популярных модулей HC-05 и HC-06, которые могут взаимодействовать только с ОС Android. HM-10/11 нельзя соединить с модулями HC-06 и HC-05, т. к. последние поддерживают протокол Bluetooth 2/2.1.

Наиболее существенным достоинством модулей Bluetooth BLE (Bluetooth Low Energy, Bluetooth LE) является их сверхмалое пиковое энергопотребление, среднее энергопотребление и энергопотребление в режиме простоя, что весьма важно для устройств с автономным питанием.

Примечание

Модуль HM-11 по своим эксплуатационным характеристикам аналогичен HM-10, но занимает меньшую площадь и имеет меньшее количество контактов.

Внешний вид, назначение контактов

Модуль BLE Bluetooth 4.0 HM-10

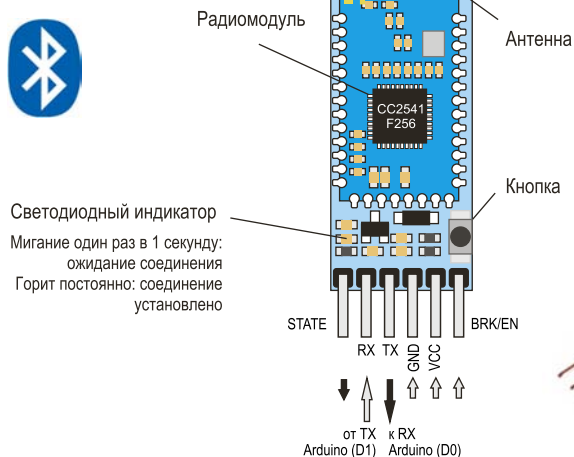


Рис. М5.1. Внешний вид и назначение контактов Bluetooth HM-10

Технические характеристики

Наименование	Значение
Чип Bluetooth	CC2540 или CC2541
Протокол связи	Bluetooth 4.0 BLE
Радиус действия	до 100 метров
Объем flash-памяти (для хранения прошивки и настроек), Мбит	8
Частота радиосигнала, ГГц	2,40 ÷ 2,48
Ток	<ul style="list-style-type: none">• в режиме установки связи — до 50 мА• после установки связи 9 мА• в режиме сна — 50 ÷ 200 мкА
Напряжение питания, В	+ 2,5 В до + 3,3 В
Пароль по умолчанию	000000
Имя Bluetooth устройства по умолчанию	HMSoft, BT05-A

До тех пор, пока модуль HM-10 не сопряжен ни с одним устройством, он находится в режиме команд и об этом сообщает мигающим светодиодом. При сопряжении с устройством светодиод начинает гореть постоянно, а модуль переходит в режим передачи данных.

Схема подключения

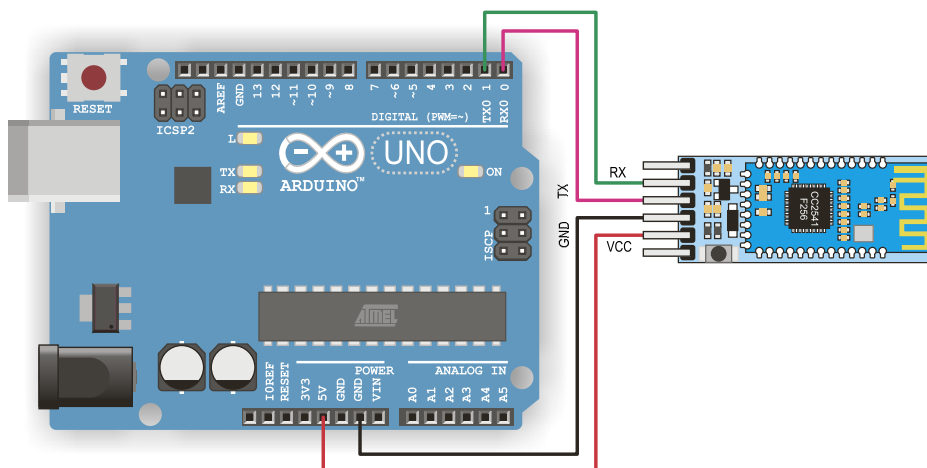


Рис. М5.2. Подключение Bluetooth HM-10

Тестируем подключение Bluetooth

Установим терминальное приложение для смартфона. Их достаточно много, и функциональные возможности их примерно одинаковы. Предлагаем для смартфонов с ОС Android популярное бесплатное приложение Serial Bluetooth Terminal, ориентированное на работу как с «классическими» модулями Bluetooth, так и с модулями, которые поддерживают протокол Bluetooth 4.0 BLE (рис. М5.3, а). Для iPhone и iPad можно использовать приложение HM10 Bluetooth Serial Lite (рис. М5.3, б).

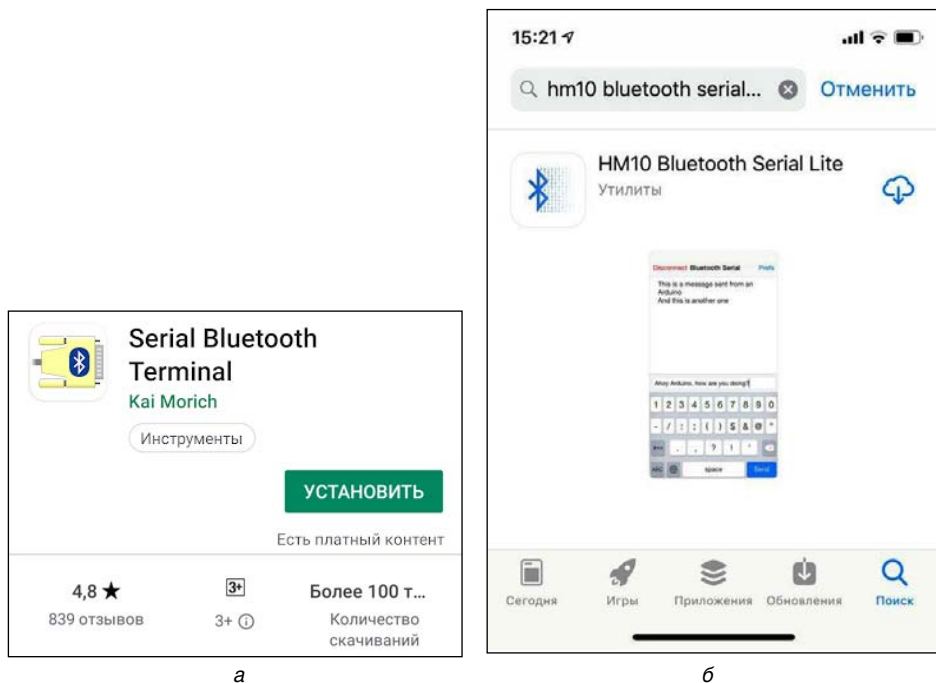
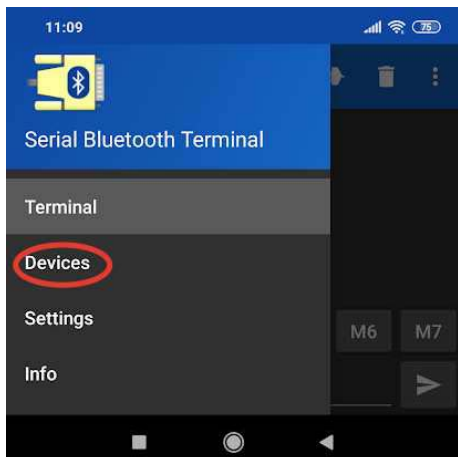
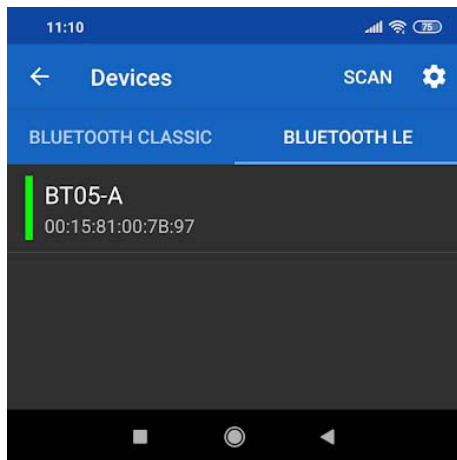


Рис. М5.3. а — окно установки Serial Bluetooth Terminal на Google Play; б — установка приложения HM10 Bluetooth Serial Lite для iOS

1. Установив приложение на смартфон, откройте его. При этом Bluetooth на смартфоне должен быть включен.
2. Подключите Bluetooth, как показано на рис. М5.2. Индикатор, расположенный на модуле Bluetooth, начнет мигать в ожидании соединения.
3. Выберите в меню команду **Devices** (рис. М5.4, а), а затем нажмите кнопку **SCAN** в правом верхнем углу окна (рис. М5.4, б).



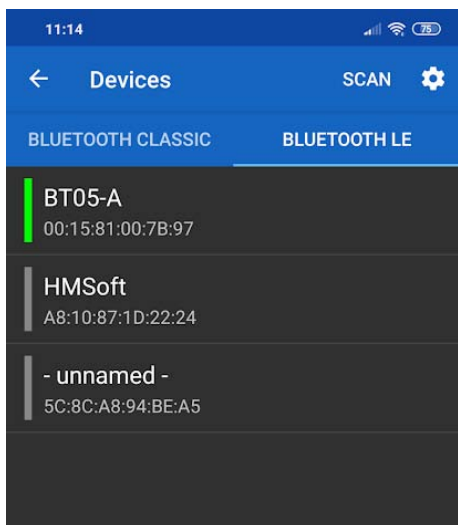
a



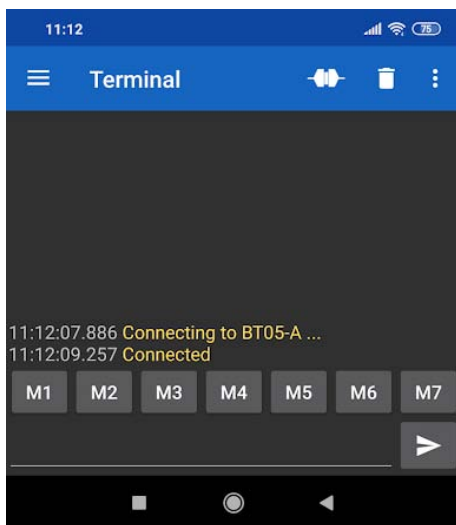
б

Рис. М5.4. Поиск устройств Bluetooth BLE в радиусе действия вашего телефона

4. После недолгого поиска на экране появится список обнаруженных модулей Bluetooth (рис. М5.5, *a*). Ваш модуль может быть обозначен в списке на вкладке **BLUETOOTH LE** как HM-10 или BT05-A (в зависимости от производителя).
5. Щелкните двойным щелчком на имени вашего модуля Bluetooth, и произойдет соединение модуля с платой Arduino через последовательный порт. При этом светодиодный индикатор на модуле перестанет мигать, а на экране смартфона появится надпись **Connected** (рис. М5.5, *б*).



a



б

Рис. М5.5. Подключение к Bluetooth BLE

Совет

В зависимости от производителя конкретного устройства Android и версии операционной системы (для Android она должна быть выше 4.3) вы можете увидеть или не увидеть HM-10 среди устройств, обнаруженных Bluetooth (Bluetooth devices). Если ваш Android не находит HM-10 в своих настройках Bluetooth, попробуйте использовать приложение BLE Scanner app.

Взаимодействие смартфона и Arduino через Bluetooth BLE

Теперь, когда соединение Arduino и модуля Bluetooth установлено, осуществим обмен информацией между устройствами. Для этого:

1. Временно отсоедините провода, подключенные к контактам **D0 (RX)** и **D1(TX)** платы Arduino, и загрузите код, приведенный в листинге M5.1.

Листинг M5.1. Обмен информацией между Arduino и модулем Bluetooth

```
void setup() {
  Serial.begin(9600);
  Serial.println("poexali!");
}
void loop() {
  if (Serial.available()) {
    char c = Serial.read(); // читаем из software-порта
    Serial.print(c);        // пишем в hardware-порт
  }
}
```

Если при загрузке скетча на Arduino не отключать эти провода, то компилятор выдаст ошибку, показанную на рис. M5.6. Конфликт связан с тем, что пины **D0** и **D1** также используются для связи с компьютером посредством USB.

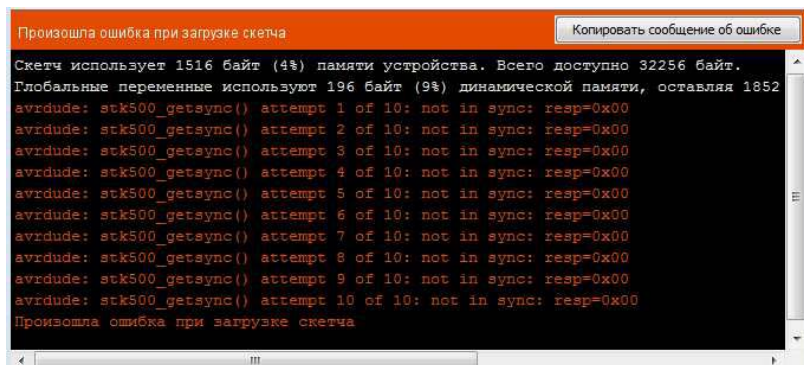


Рис. M5.6. Ошибки загрузки скетча в Arduino

2. Снова подключите провода к контактам **D0 (RX)** и **D1(TX)** и выполните сопряжение смартфона с модулем Bluetooth, как было показано на рис. M5.5.
3. Выполнив сопряжение, можно поздороваться с хостом на другом конце Bluetooth-канала (рис. M5.7). Как можно видеть, приветствие прошло успешно!

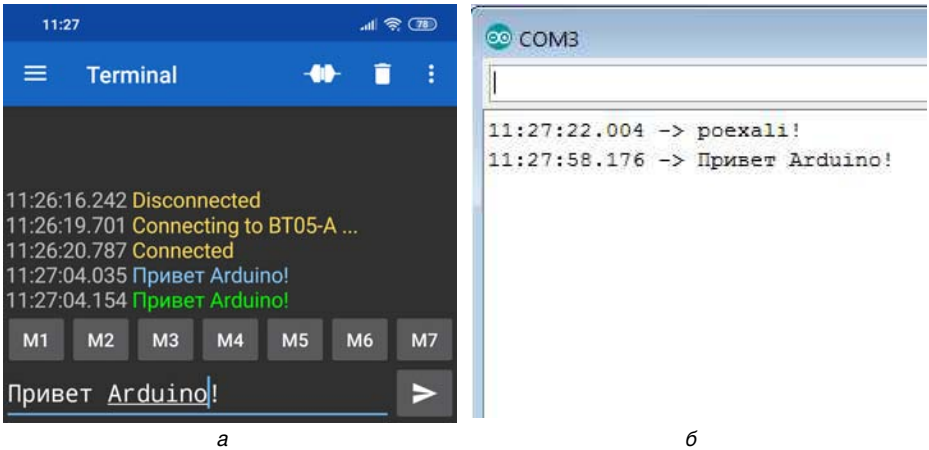


Рис. M5.7. Привет Arduino! — вводим на смартфоне (а) и получаем в Мониторе порта (б)

При отладке скетча вам придется многократно отключать провода от контактов **D0** и **D1** во время загрузки скетча в Arduino. Кроме того, одного порта вам может быть недостаточно. В этом случае целесообразно использовать библиотеку `SoftwareSerial`, которая предустановлена в Arduino IDE. Библиотека `SoftwareSerial` позволяет реализовать последовательный интерфейс не только на **D0** и **D1**, но и на любых других цифровых выводах Arduino с помощью программных средств, дублирующих функциональность UART (отсюда и название «`SoftwareSerial`»). Кроме того, она дает возможность программно создавать несколько последовательных портов, работающих на скорости до 115 200 бод.

1. Подключите контакты **TX** и **RX** модуля Bluetooth к выводам **D2** и **D3 (RX→D3, TX→D2)** платы Arduino (а не к **D0** и **D1**, как вы делали ранее!).
2. Загрузите код, приведенный в листинге M5.2.

Листинг M5.2. Сопряжение платы Arduino и модуля Bluetooth

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2,3); //указываем пины RX и TX соответственно

void setup() {
    pinMode(2, INPUT);
    pinMode(3, OUTPUT);
}
```

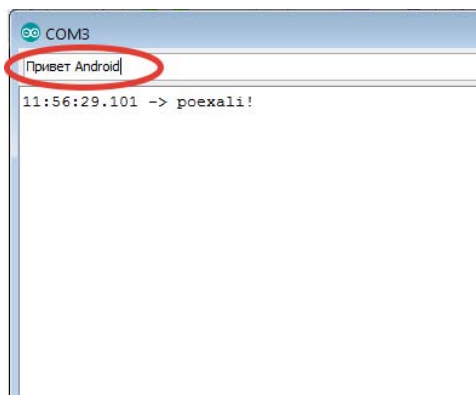
```

Serial.begin(9600);
mySerial.begin(9600);
Serial.println("поехали!");
}

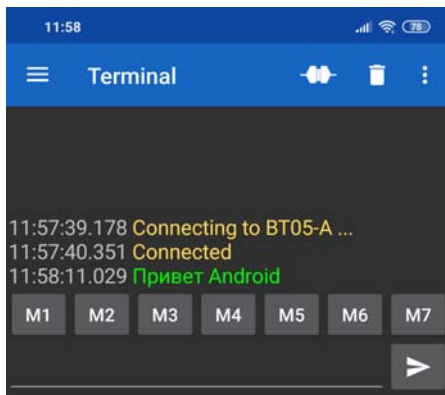
void loop() {
  if (mySerial.available()) {
    char c = mySerial.read(); // читаем из software-порта
    Serial.print(c);         // пишем в hardware-порт
  }
  if (Serial.available()) {
    char c = Serial.read(); // читаем из hardware-порта
    mySerial.write(c);      // пишем в software-порт
  }
}
}

```

3. Введите в Мониторе порта любой текст (рис. М5.8, а) и нажмите кнопку **Отправить** — текст появится в окне терминала смартфона (рис. М5.8, б).



а



б

Рис. М5.8. Привет Android! вводим в Мониторе порта (а) и получаем на смартфоне (б)

Управление светодиодом RGB по Bluetooth

Теперь осуществим на практике управление электронным прибором со смартфона по каналу Bluetooth.

1. Соберите схему, представленную на рис. М5.9.

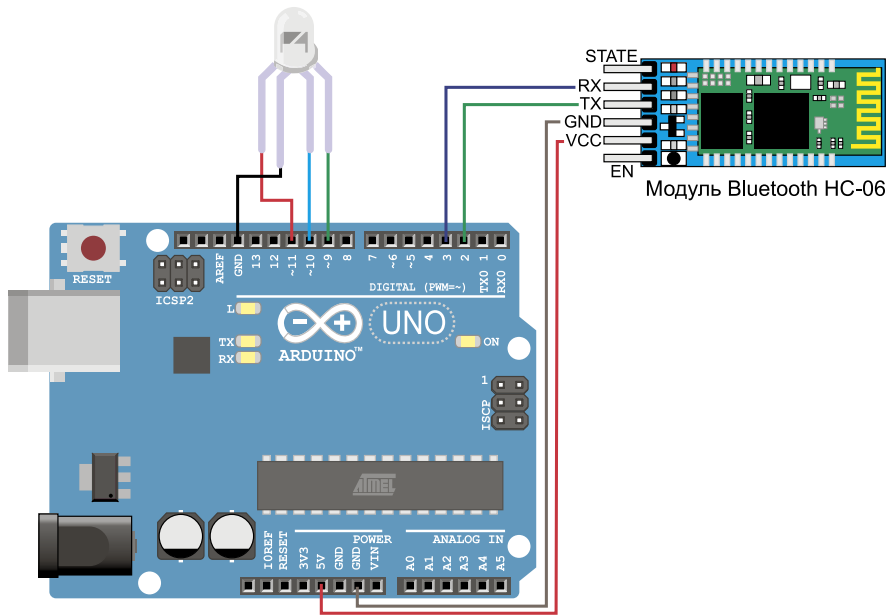


Рис. М5.9. Управление RGB-светодиодом через Bluetooth

2. Загрузите скетч, приведенный в листинге М5.3.

Листинг М5.3. Управление светодиодом RGB через Bluetooth

```
#include <SoftwareSerial.h>
// Константы для хранения номеров контактов вывода:
const int greenPin = 9; // "зеленый" контакт LED RGB
const int bluePin = 10; // "синий" контакт LED RGB
const int redPin = 11; // "красный" контакт LED RGB
const int pinRX = 2; // RX - вход для приема данных -> к TX bluetooth
const int pinTX = 3; // TX - выход для передачи данных -> к RX bluetooth
int currentPin = 0; // текущий контакт для задания яркости
```

```
SoftwareSerial mySerial(pinRX, pinTX); // указываем пины rx и tx
```

```
void setup() {
    pinMode(pinRX, INPUT);
    pinMode(pinTX, OUTPUT);
```

```

pinMode(redPin, OUTPUT);
pinMode(greenPin, OUTPUT);
pinMode(bluePin, OUTPUT);

Serial.begin(9600);
mySerial.begin(9600);
Serial.println("poexali!");
}

void loop() {
  if (mySerial.available()) {
    char inByte = mySerial.read(); // читаем из software-порта
    Serial.print(inByte); // пишем в hardware-порт
    rgb(inByte); //включаем светодиод "цветом" inByte
  }

  if (Serial.available()) {
    char c = Serial.read(); // читаем из hardware-порта
    mySerial.write(c); // пишем в software-порт
    rgb(c); //включаем светодиод "цветом" c
  }
}

//функция для включения светодиода RGB
void rgb(byte cByte){
  digitalWrite(currentPin,LOW);
  // игнорируем любые другие значения:
  if (cByte == 'r') {
    currentPin = redPin;
  }
  if (cByte == 'g') {
    currentPin = greenPin;
  }
  if (cByte == 'b') {
    currentPin = bluePin;
  }
  digitalWrite(currentPin,HIGH);
}


```

3. Теперь, вводя символы `r`, `g` и `b` в Мониторе порта или на смартфоне, подключенном через модуль Bluetooth с помощью терминальной программы, мы можем изменять цвет светодиода RGB.

3. Светодиоды и дисплей

3.1. Светодиод RGB

Конструктивно трехцветный светодиод представляет собой три цветных светодиода: красный (R), зеленый (G) и синий (B), смонтированных в общем корпусе. Различные оттенки цвета получаются путем смешения 3-х базовых цветов (модель RGB). Поскольку светодиоды расположены очень близко друг к другу, наши глаза видят результат сочетания цветов, а не три цвета по отдельности. Для регулировки интенсивности каждого светодиода можно использовать сигнал ШИМ.

	Подробную информацию о подключении трехцветного светодиода к плате Arduino вы можете найти в прилагаемой к набору книге Дж. Блума «Изучаем Arduino: инструменты и методы технического волшебства» (см. раздел 2.9 «Создание управляемого ночника на RGB-светодиоде».	стр. 58
--	--	---------

Внешний вид, назначение контактов

Имейте в виду, что существуют два вида светодиодов RGB: с *общим анодом* (рис. L1.1, *справа*) и с *общим катодом* (рис. L1.1, *слева*), которые имеют разную схему подключения общего вывода: к минусу (-) или к плюсу (+).

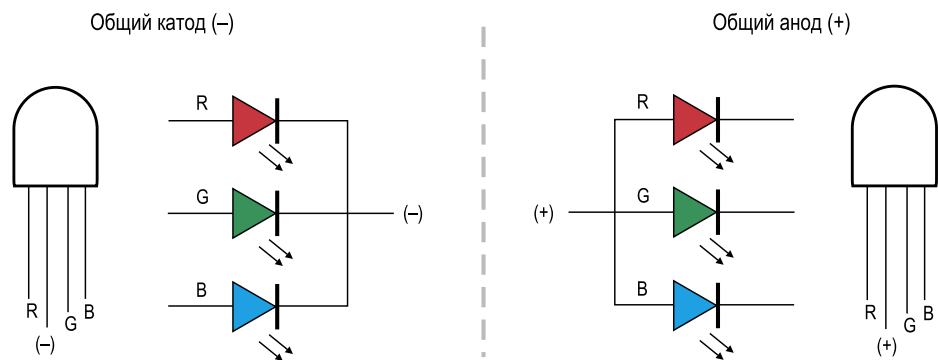


Рис. L1.1. RGB-светодиод с общим катодом (слева) и с общим анодом (справа)

3.2. ЖК-дисплей 1602 с модулем I²C

В 10-й главе книги Дж. Блума «Изучаем Arduino: инструменты и методы технического волшебства» подробно описано применение LCD-дисплея в проектах Arduino (стр. 202).

На Arduino Uno при разработке больших проектов очень часто для подключения различных устройств не хватает выводов. Например, для подключения одного LCD-дисплея в проекте из книги Дж. Блума (стр. 205, рис. 10.2) задействовано 6 цифровых выводов (D2 ÷ D7) и два вывода питания платы Arduino.

Чтобы сократить количество выводов для подключения LCD-дисплеев, можно использовать специальный интерфейсный модуль IIC/I²C (рис. L2.1), с помощью которого обмен данными между LCD-дисплеем (ЖКД) и Arduino осуществляется по последовательному протоколу I²C (Inter-Integrated Circuit), который подробно описан в 8-й главе книги Дж. Блума.

В этом случае будут задействованы только 4 вывода Arduino:

- **GND** (Земля),
- **VCC** (Питание +5V),
- **A4** (SDA, *Serial Data* — последовательные данные),
- **A5** (SCL, *Serial Clock* — сигнал последовательного тактирования).

Внешний вид, назначение контактов

Удобно приобрести модуль LCD-дисплея, на котором модуль IIC/I²C уже установлен (см. рис. L2.1). На модуле LCD-дисплея расположен потенциометр для управления контрастом дисплея.

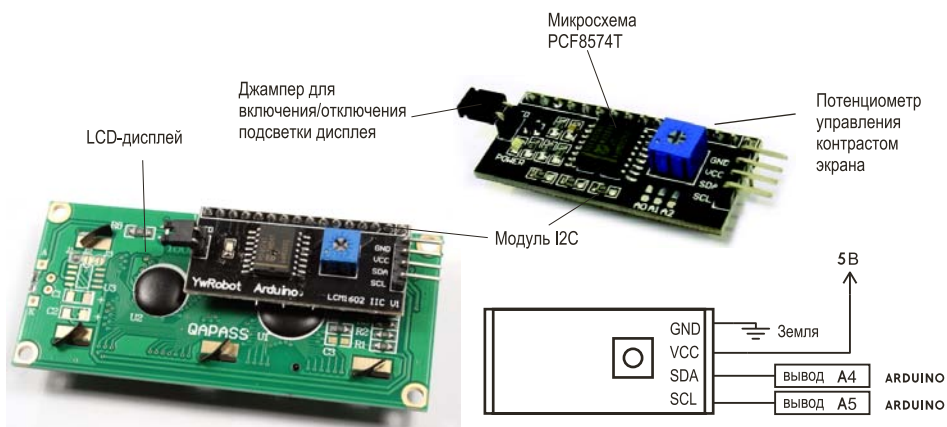


Рис. L2.1. Плата ЖКД 1602 с модулем I²C

Схема подключения

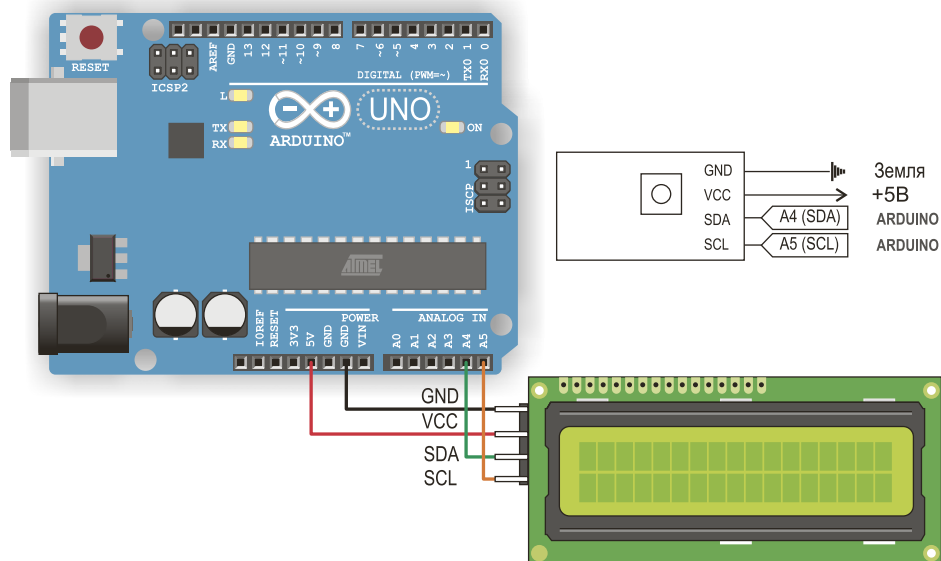


Рис. L2.2. Схема подключения LCD-дисплея с модулем I²C

Основные характеристики

Наименование	Значение
Напряжение, В	От 4,5 до 5,5
Ток, мА	1,5
Ток подсветки дисплея, мА	120
Напряжение подсветки дисплея, В	От 4,1 до 4,3

Программный код

1. Загрузите библиотеку `LiquidCrystal I2C by Frank de Brabander` для работы Arduino с LCD-дисплеем по протоколу I²C. Для этого откройте **Менеджер библиотек**, выполнив команду **Инструменты | Управлять библиотеками**. Справа сверху в строке поиска введите `LiquidCrystal I2C`. В открывшемся списке выберите **LiquidCrystal I2C by Frank de Brabander**. Нажмите кнопку **Установка**.
2. Загрузите скетч из листинга L2.1. Адрес шины I²C по умолчанию: `0x27`.

Примечание

Библиотеку также можно скачать в виде ZIP-файла по ссылке <https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library> и подключить в среду разработки Arduino IDE с помощью команды **Скетч | Подключить библиотеку | Добавить .ZIP библиотеку...**

Листинг L2.1. Тест LCD-экрана

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

//задаем адрес экрана 0x27, 16 символов, 2 строки
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  lcd.init(); // Инициализируем экран, включаем подсветку
  lcd.backlight();
  //Устанавливаем положение курсора для первой строки.
  lcd.home();
  //выводим строку 1
  lcd.print("String 1 Test");
  //выводим строку 2
  lcd.setCursor(0, 1);
  lcd.print("String 2 Test");
}
void loop() {
}
```

Библиотека `LiquidCrystal_I2C` не поддерживает кириллические символы. Для их отображения по адресу https://github.com/ssilver2007/LCD_1602_RUS скачайте и установите библиотеку `LCD_1602_RUS` (автор Сергей Сироткин). При этом основная библиотека `LiquidCrystal_I2C` должна быть уже установлена. Имейте в виду, что максимально возможно отображение восьми уникальных по начертанию русских символов (например: Ж, Д, И, Ю и т. п.).

Загрузите скетч из листинга L2.2 — теперь символы кириллицы должны отображаться корректно.

Листинг L2.2. Тест кириллических шрифтов на LCD-экране

```
#include <LCD_1602_RUS.h>
LCD_1602_RUS lcd(0x27, 16, 2);
void setup()
{
    lcd.init();
    lcd.backlight();
    lcd.setCursor(0, 0);
    lcd.print("Первая строка");
    lcd.setCursor(0, 1);
    lcd.print("Вторая строка");
}
void loop()
{
}
```

3.3. Светодиодная матрица 8×8 с драйвером MAX7219

Светодиодная матрица представляет собой набор светодиодов, сгруппированных в квадратную или прямоугольную матрицу. Каждый светодиод имеет отрицательный вывод (катод), который подключен к «земле», и положительный (анод), подключенный к источнику питания. Аноды и катоды соединены так, что образуют столбцы и строки. Таким образом, подавая питание от Arduino на определенные строки и столбцы, мы можем «зажигать» нужный светодиод.

Внешний вид, назначение контактов

Для управления матрицей 8×8 требуются 16 контактов (рис. L3.1, *слева*). Однако задействовать 16 контактов Arduino Uno для вывода символов на матрицу очень расточительно. Лучшим решением является использование специального расширителя выводов на базе микросхемы MAX7219. В этом случае для управления матрицей достаточно пяти контактов. Такие модули выпускаются в различном исполнении (рис. L3.1, *справа*).

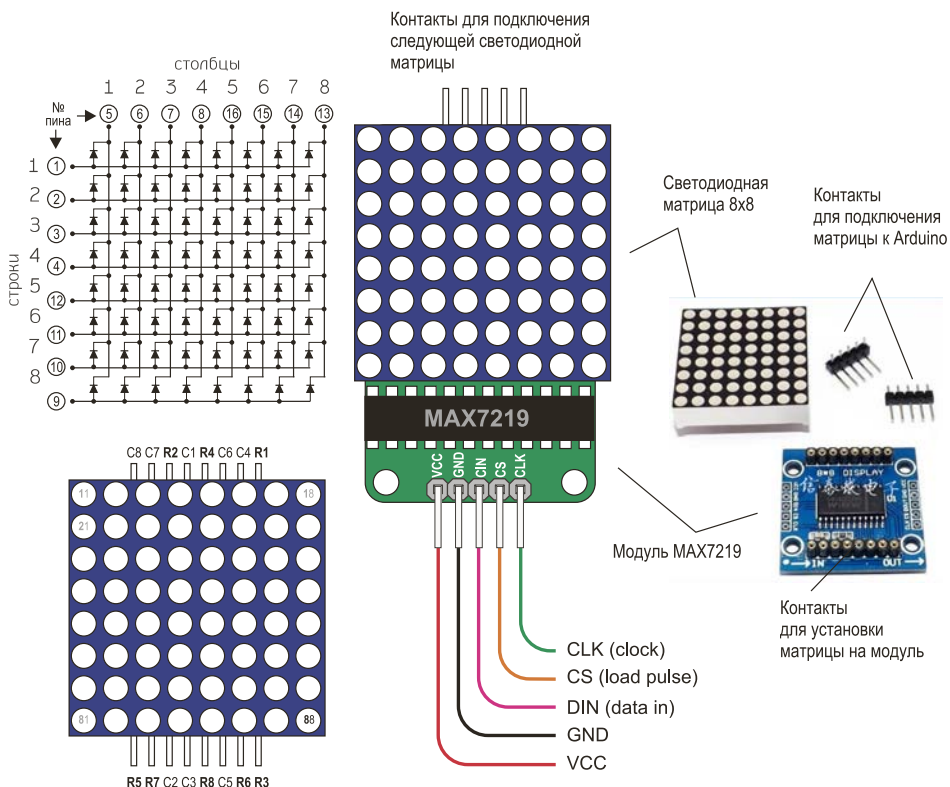


Рис. L3.1. Светодиодная матрица 8×8 с модулем MAX7219

Схема подключения

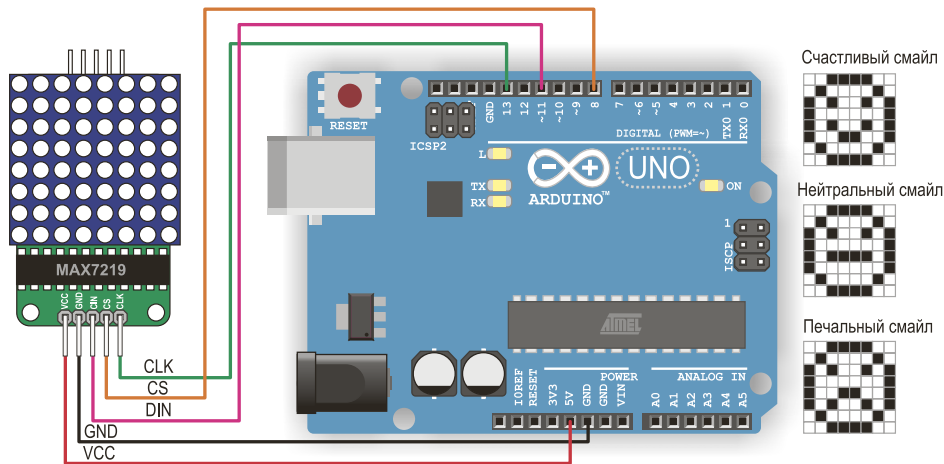


Рис. L3.2. Схема подключения светодиодной матрицы 8×8 с модулем MAX7219

Программный код

1. Загрузите библиотеку `LedControl` для работы Arduino с модулем MAX7219. Для этого откройте **Менеджер библиотек**, выполнив команду **Инструменты | Управлять библиотеками**. Справа сверху в строке поиска введите `LedControl`. В открывшемся списке выберите **LedControl by Eberhard Fahle**. Нажмите кнопку **Установка**.
2. Загрузите скетч из листинга L3.1 — на матрице появится смайлик, изменяющий «мимику» через каждую секунду.

Листинг L3.1. Вывод динамического смайла на светодиодную матрицу

```
#include "LedControl.h"
#include "binary.h"

/*
DIN подключен к пину 11
CLK подключен к пину 13
CS подключен к пину 8
*/
```

```

LedControl matr=LedControl(11,13,8,1);

// СЧАСТЛИВЫЙ СМАЙЛ
byte hf[8]= {B00111100,B01000010,B10011001,B10100101,B10000001,B101001
01,B01000010,B00111100};
// Нейтральный смайл
byte nf[8]= {B00111100,B01000010,B10000001,B10111101,B10000001,B101001
01,B01000010,B00111100};
// Печальный смайл
byte sf[8]= {B00111100,B01000010,B10100101,B10011001,B10000001,B101001
01,B01000010,B00111100};

void setup() {
    matr.shutdown(0,false); //Включаем светодиодную матрицу
    matr.setIntensity(0,8); // Установка яркости на среднее значение
    matr.clearDisplay(0); // Очистка матрицы
    Serial.begin(9600);
}

void loop(){
//Вывод счастливого смайла
    matr.setRow(0,0,hf[0]);
    matr.setRow(0,1,hf[1]);
    matr.setRow(0,2,hf[2]);
    matr.setRow(0,3,hf[3]);
    matr.setRow(0,4,hf[4]);
    matr.setRow(0,5,hf[5]);
    matr.setRow(0,6,hf[6]);
    matr.setRow(0,7,hf[7]);
    delay(1000); //задержка 1 с
//Вывод нейтрального смайла
    matr.setRow(0,0,nf[0]);
    matr.setRow(0,1,nf[1]);
    matr.setRow(0,2,nf[2]);
    matr.setRow(0,3,nf[3]);
    matr.setRow(0,4,nf[4]);
    matr.setRow(0,5,nf[5]);

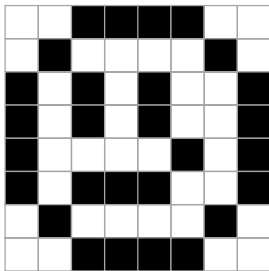
```

```

matr.setRow(0,6,nf[6]);
matr.setRow(0,7,nf[7]);
delay(1000); //задержка 1 с
//Вывод печального смайла
matr.setRow(0,0,sf[0]);
matr.setRow(0,1,sf[1]);
matr.setRow(0,2,sf[2]);
matr.setRow(0,3,sf[3]);
matr.setRow(0,4,sf[4]);
matr.setRow(0,5,sf[5]);
matr.setRow(0,6,sf[6]);
matr.setRow(0,7,sf[7]);
delay(1000); //задержка 1 с
}

```

Вы можете самостоятельно создавать собственные изображения для вывода на матрицу 8×8. Изображение можно смоделировать на обычном листке в клетку, в табличном редакторе MS Excel или онлайн. Например, на ресурсе <http://arduino.on.kg/matrix-font> можно создать и получить программный код собственного изображения или выбрать готовый символ (рис. L3.3).



byte		customChar[8]						=	{
B	0	0	1	1	1	1	0	0	,
B	0	1	0	0	0	0	1	0	,
B	1	0	1	0	1	0	0	1	,
B	1	0	1	0	1	0	0	1	,
B	1	0	0	0	0	1	0	1	,
B	1	0	1	1	1	0	0	1	,
B	0	1	0	0	0	0	1	0	,
B	0	0	1	1	1	1	0	0	} ;

Рис. L3.3. Моделирование изображения на матрице 8×8

4. Двигатели

4.1. Серводвигатель TowerPro SG90 9G

Серводвигатель TowerPro SG90 — это такой вид привода, который может повернуть свой вал на определенный угол. Схема работы сервопривода основана на использовании обратной связи. Конструктивно сервопривод состоит из двигателя, датчика позиционирования (позиционера) и управляющей системы (рис. D1.1).



Подробную информацию об управлении серводвигателем с помощью платы Arduino вы можете найти в прилагаемой к набору книге Дж. Блума «Изучаем Arduino: инструменты и методы технического волшебства» (см. раздел 4.11 «Управление серводвигателем»).

стр. 98

Внутреннее устройство, назначение контактов

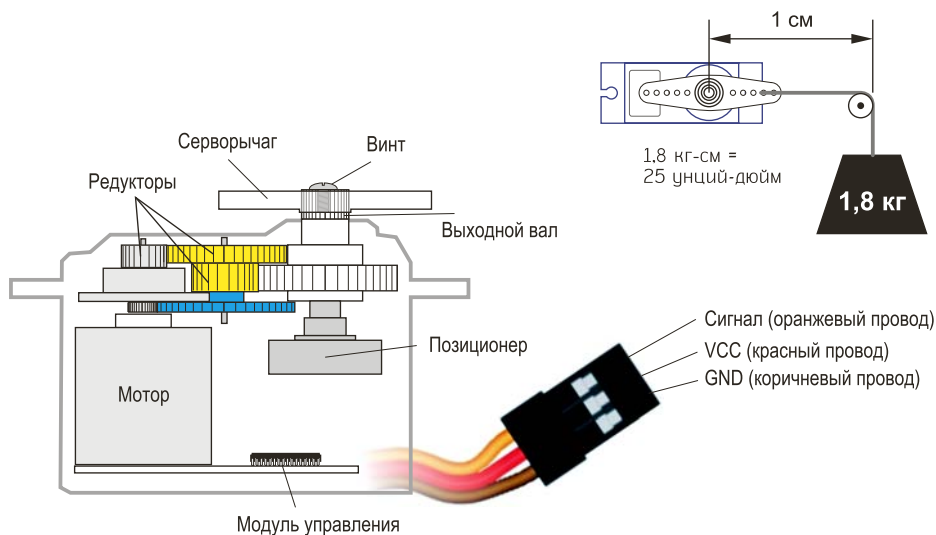


Рис. D1.1. Внутреннее устройство серводвигателя

Основные характеристики

Наименование	Значение
Ток в режиме ожидания, мА	5
Рабочий ток без нагрузки, мА	100
Ток с максимальный с нагрузкой, мА	700
Рабочее напряжение, В	От 4,8 до 6
Крутящий момент, кг/см	1,8
Скорость без нагрузки	0,12 секунд / 60 градусов (4,8V)
Предельный угол поворота	90/180/360
Длина кабеля, см	20
Размеры (Г×Ш×В), мм	32,6×12,5×27,3
Вес сервопривода без серворычагов и винтов, г	9

Схема подключения

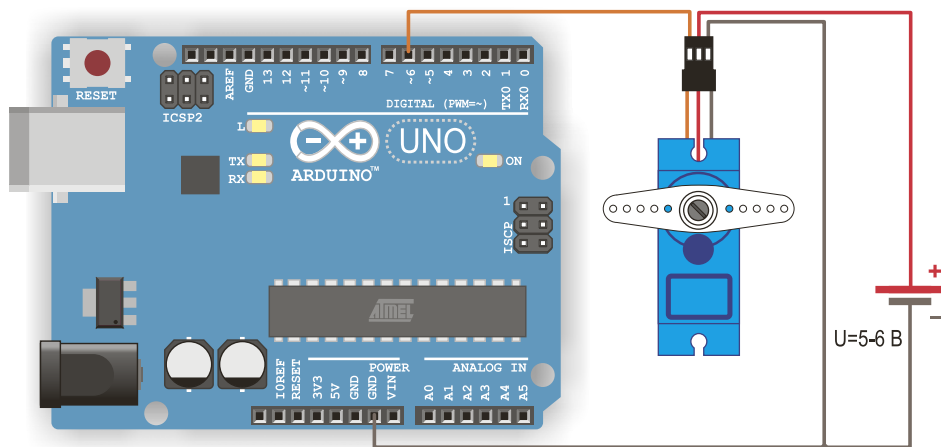


Рис. D1.2. Схема подключения серводвигателя к Arduino Uno с внешним источником питания ($U = 5 \div 6$ В)

Программный код

Листинг D1.1. Управление сервоприводом с помощью Arduino Uno

```
#include <Servo.h>
Servo myservo;

void setup() {
    myservo.attach(6); // Назначим пин управления сервоприводом
}

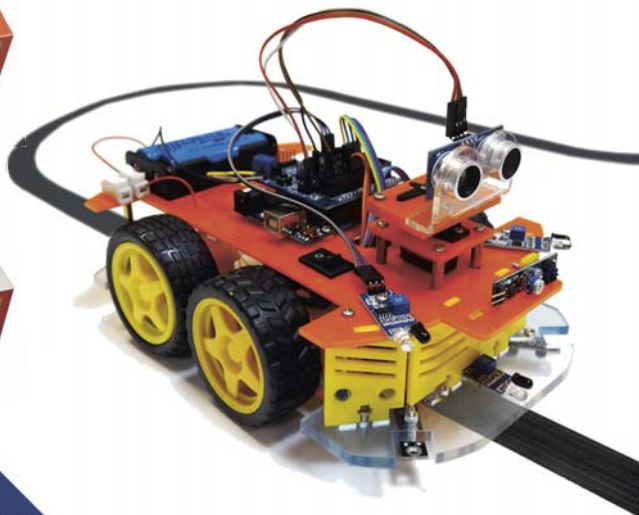
void loop() {
    myservo.write(60); // Установим положение сервопривода на 60°
    delay(3000);       // Ждем 3с
    myservo.write(120); // Установим положение сервопривода на 120°
    delay(3000);       // Ждем 3с
}
```



КНИГА + ЭЛЕКТРОННЫЕ КОМПОНЕНТЫ
В ОДНОЙ КОРОБКЕ!



дерзай!



НАБОРЫ

ПО ЭЛЕКТРОНИКЕ
И РОБОТОТЕХНИКЕ

www.bhv.ru/books/kits