

# **Инструкция к набору для моделирования Arduino Uno R3 9V Maximum KIT с RFID модулем**



## **СОДЕРЖАНИЕ**

## **Состав набора**

Состав комплекта:

Перемычка папа-папа: 16 шт.

LCD-дисплей 1602A: 1 шт.

Светодиодная матрица 8x8 1588bs: 1 шт.

Джойстик двухосевой: 1 шт.

Модуль RFID Считыватель/программатор карт с картой и брелоком: 1 шт.

Шаговый двигатель: 1 шт.

Arduino UNO R3: 1 шт.

Четырехразрядный семисегментный индикатор: 1 шт.

ИК пульт: 1 шт.

USB кабель: 1 шт.

Макетная плата 830 точек: 1 шт.

Датчик уровня воды: 1 шт.

Модуль RGB светодиода: 1шт.

Датчик температуры и влажности DHT-11: 1 шт.

Кнопочная матрица 4x4: 1 шт.

Одноразрядный семисегментный индикатор: 1 шт.

Сервопривод SG90 9г: 1шт.

Кабель питания от кроны 9V: 1 шт.

Перемычки папа-мама: 10 шт.

Преобразователь интерфейса LCD в I2C для дисплеев: 1 шт.

Кнопка тактовая: 4 шт.

Потенциометр 10 кОм: 1 шт.

Светодиод 5мм красный: 5 шт.

Светодиод 5мм зеленый: 5 шт

Светодиод 5мм желтый: 5 шт.

Фоторезистор 5516: 3 шт.

Микросхема sn74hc595n сдвиговой регистр: 1 шт.

Зуммер: 2шт.

Датчик наклона и вибрации SW-520D: 1 шт.

ИК-приемник cho 1838: 1 шт.

Микрофонный модуль датчика звука: 1 шт.

Датчик температуры LM35: 1 шт.

ИК передатчик: 1 шт.

Резистор 1 кОм: 10 шт.

Резистор 10 кОм: 10 шт.

Резистор 220 Ом: 10 шт.

Плата управления шаговыми двигателями: 1 шт.

Модуль DS1302 часы в реальном времени с баратрейкой: 1 шт.

1 канальный релейный модуль плата щит для PIC AVR DSP ARM: 1 шт.

Инструкции и проекты для Arduino UNO – 1 шт.

## 1. Как установить интегрированную среду разработки Arduino

Интегрированная среда разработки Arduino (IDE) — это программная часть платформы Arduino. В этом проекте вы узнаете, как настроить компьютер для использования Arduino и как приступить к последующим проектам.

Программное обеспечение Arduino, которое вы будете использовать можно запрограммировать под свой Arduino для Windows, Mac и Linux. Процесс установки отличается для всех трех платформ, и, к сожалению, для установки программного обеспечения требуется определенный объем ручной работы.

Перейдите по ссылке ниже и найдите следующую страницу:  
[Software | Arduino](#)

## Downloads



### Arduino IDE 2.1.0

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

Nightly builds with the latest bugfixes are available through the section below.

[SOURCE CODE](#)

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

#### DOWNLOAD OPTIONS

**Windows** Win 10 and newer, 64 bits  
**Windows** MSI installer  
**Windows** ZIP file

**Linux** AppImage 64 bits (X86-64)  
**Linux** ZIP file 64 bits (X86-64)

**macOS** Intel, 10.14: "Mojave" or newer, 64 bits  
**macOS** Apple Silicon, 11: "Big Sur" or newer, 64 bits

[Release Notes](#)

В зависимости от вашей операционной системы выберите нужную версию.  
Скачайте Arduino IDE, установите её на свой компьютер, откройте приложение.

## 2. Как установить драйвер на Arduino

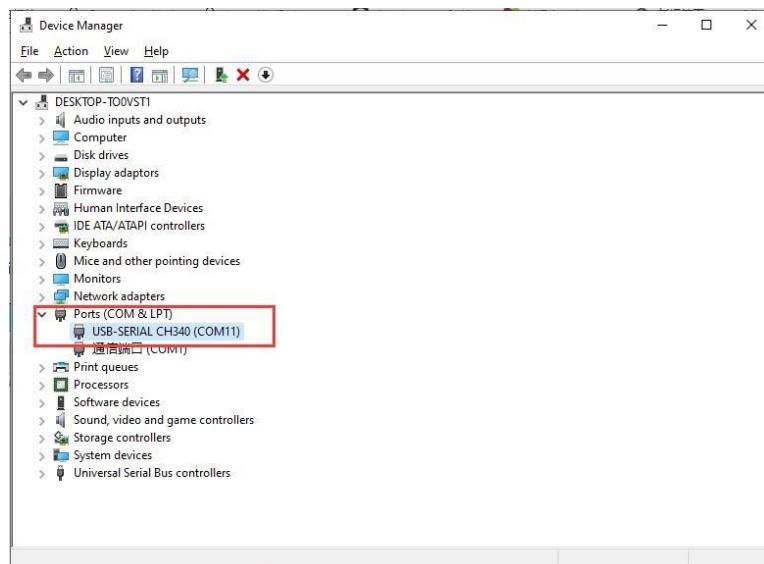
Подключите Arduino к пк, также скачайте программу для драйвера CH341SER.EXE, она имеется в яндекс диске. Её можно установить только на Windows. Скачав её откройте и нажмите кнопку "INSTALL".

Драйвера на вашу ардуино установлены успешно.

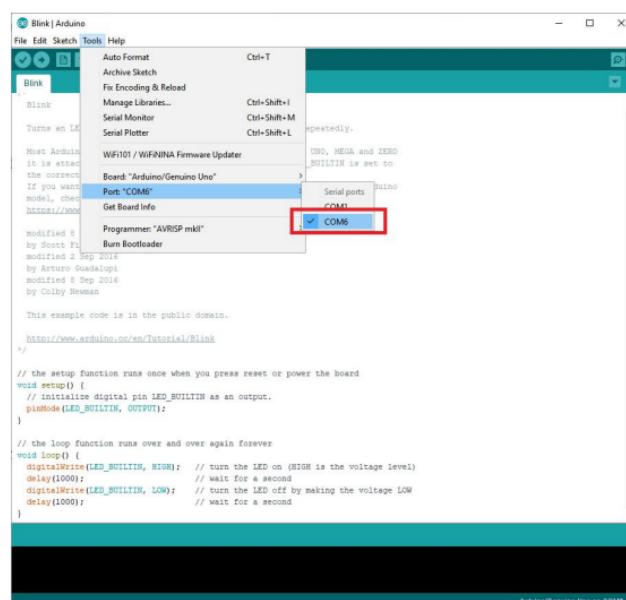
## 3. Как работать с Arduino IDE.

Открываем нашу среду разработки. Первым делом нам надо выбрать com Port, а также нашу модель Arduino.

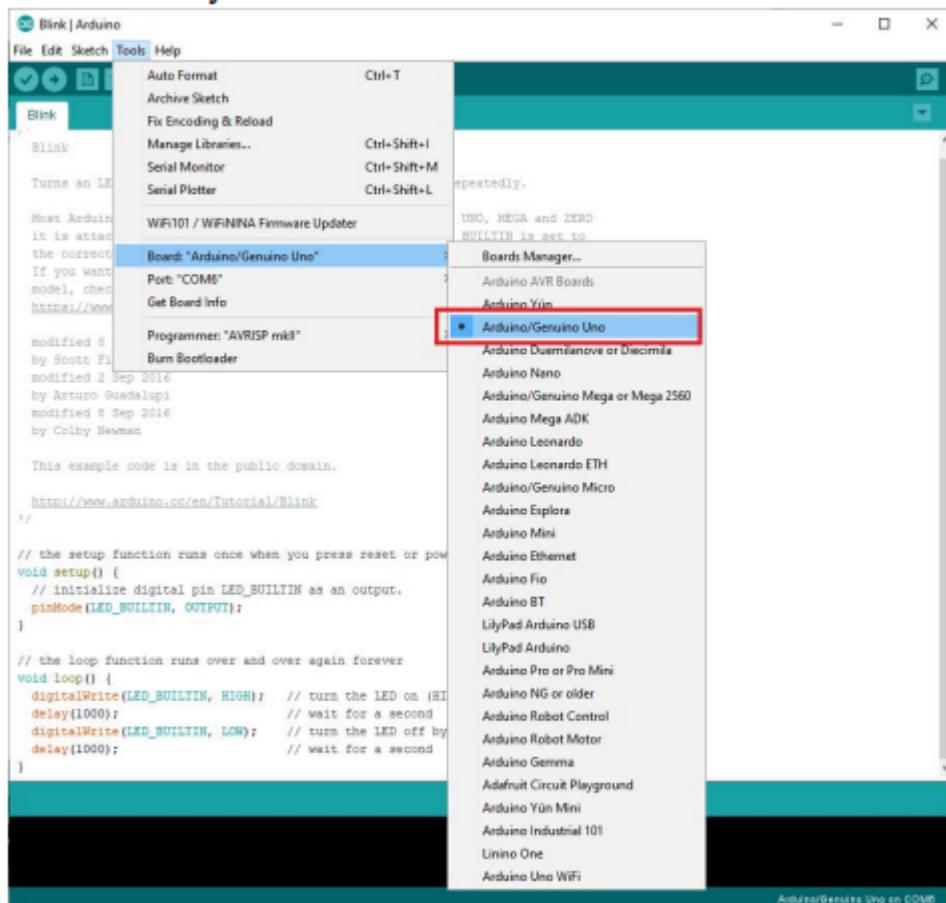
Обычно это Com3. Его можно увидеть в диспетчер устройств.



Поиск нашего com порта



Выбираем тот com порт, который мы  
увидели до этого



Выбор нашего устройства, Arduino Uno

## Добавление библиотек в проект

Когда вы уже освоитесь с программным обеспечением Arduino и используете встроенные функции, вы можете расширить возможности своего Arduino с помощью дополнительных библиотек.

### Что такое библиотеки?

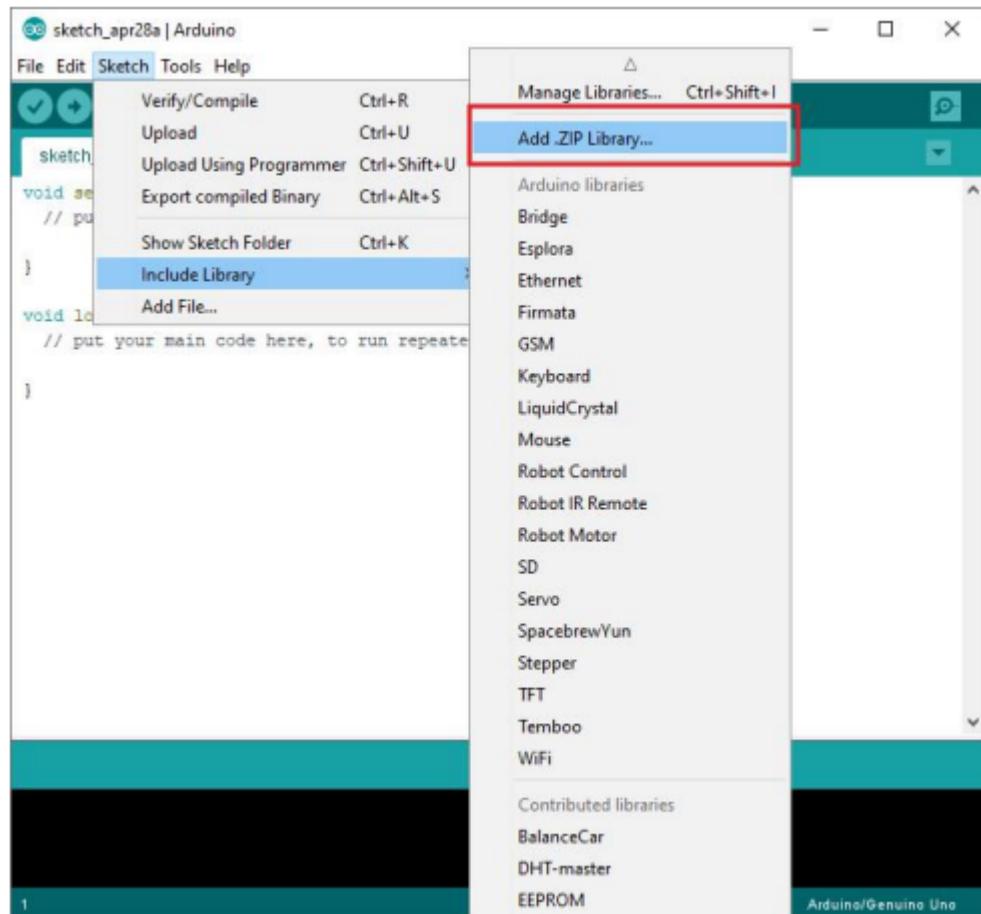
Библиотеки представляют собой коллекцию кода, упрощающий подключение к датчику, дисплею, модулю и т. д. Например, встроенная библиотека LiquidCrystal упрощает взаимодействие с символьными ЖК-дисплеями. В Интернете доступны для скачивания сотни дополнительных библиотек. Встроенные библиотеки и некоторые из дополнительных библиотек перечислены в справочнике(в папке Библиотеки на яндекс диске)

Чтобы использовать дополнительные библиотеки, их необходимо установить.

Скачайте библиотеки в формате .zip

Вы можете установить сторонние библиотеки в IDE. Не распаковывайте загруженную библиотеку, оставьте как есть в .zip формате.

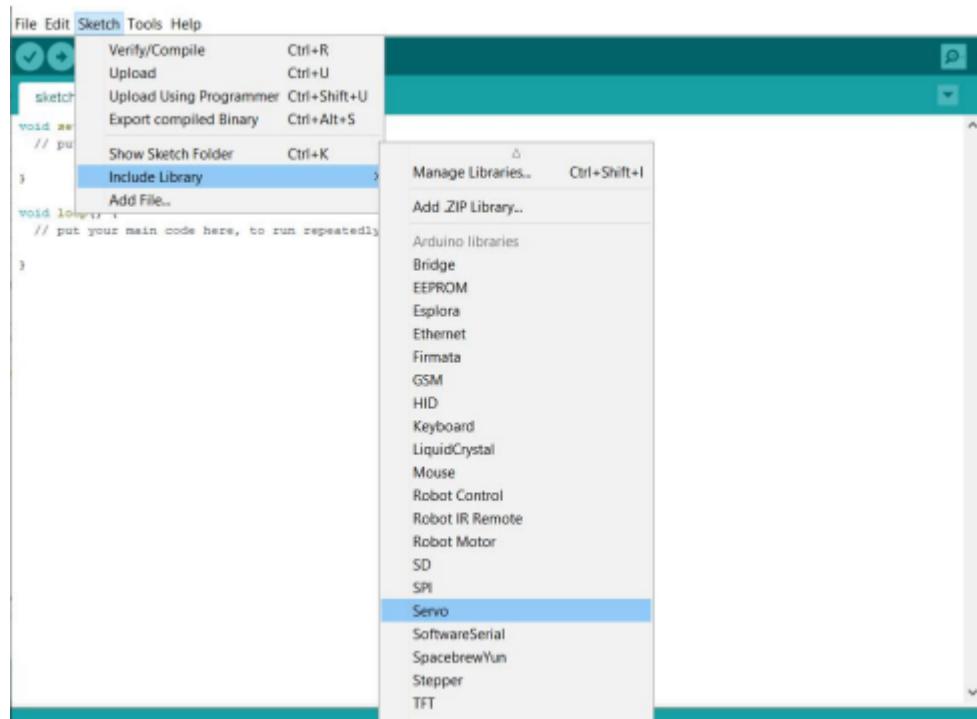
В Arduino IDE перейдите к «Sketch» > «Include Library». В верхней части раскрывающегося списка выберите параметр «Add .ZIP Library». Появится индикация состояния загрузки - спустя минуту ваша библиотека успешно установлена.



Установка библиотеки

Вы также можете из IDE открыть/установить библиотеки.

Снова откройте «Sketch» > «Include Library». Теперь вы должны увидеть библиотеку в нижней части раскрывающегося меню. Он готов к использованию в вашем эскизе.



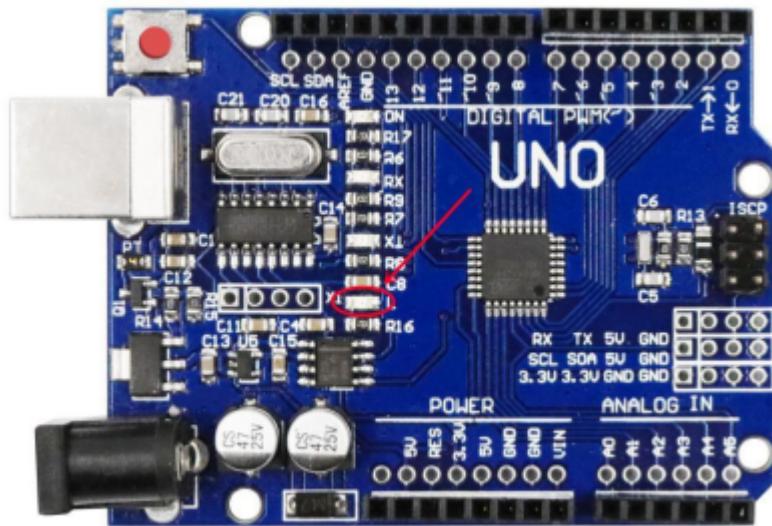
Обзор встроенных библиотек

#### 4. Тест на мигание встроенного светодиода

Вы узнаете, как запрограммировать плату контроллера UNO на мигание встроенного светодиода Arduino и как загрузить программу пошагово.

Плата UNO имеет ряды разъемов по обе стороны, которые используются для подключения к нескольким электронным устройствам и подключаемым "экранам", что расширяет ее возможности.

Он также оснащен одним светодиодом, которым вы можете управлять из своих эскизов. Этот светодиод встроен в плату UNO и часто обозначается как светодиод "L", поскольку именно так он обозначен на плате.



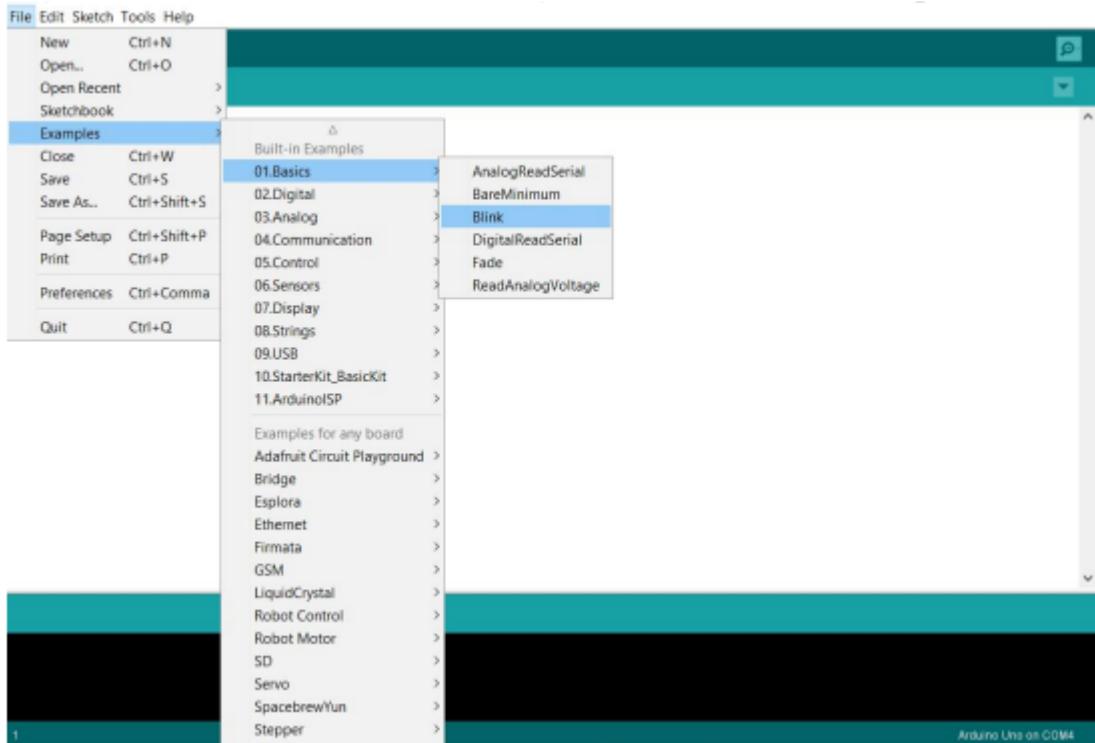
Плата Arduino Uno

Возможно, вы обнаружите, что светодиод "L" на вашей плате UNO уже мигает, когда вы подключаете ее к USB-разъему. Это связано с тем, что платы обычно поставляются с предустановленным эскизом "Blink".

В этом проекте мы перепрограммируем плату UNO с помощью нашего собственного эскиза мигания, а затем изменим частоту, с которой она мигает.

Arduino IDE включает в себя большую коллекцию примеров эскизов, которые вы можете загрузить и использовать. Это включает в себя примерный набор для того, чтобы заставить светодиод "L" LED мигать.

Загрузите эскиз "Blink", который вы найдете в системе меню IDE в разделе File > Examples > 01.Basics



```
File Edit Sketch Tools Help
Blink
by Scott Fitzgerald
modified 2 Sep 2016
by Arturo Guadalupi
modified 8 Sep 2016
by Colby Newman

This example code is in the public domain.

https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

Arduino Uno on COM4

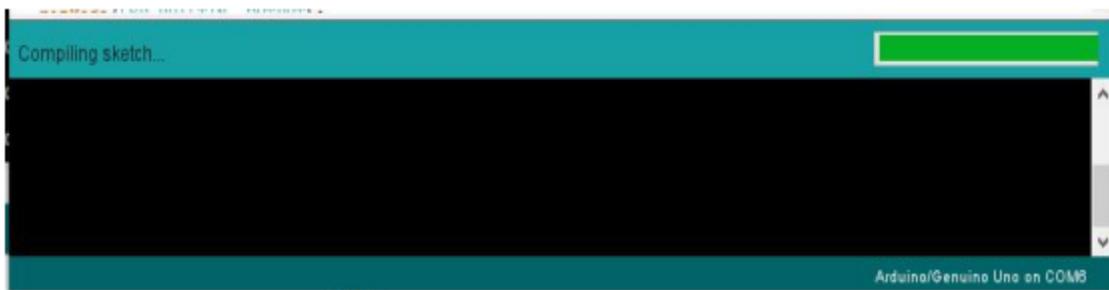
Arduino IDE покажет вам текущие настройки платы в нижней части окна



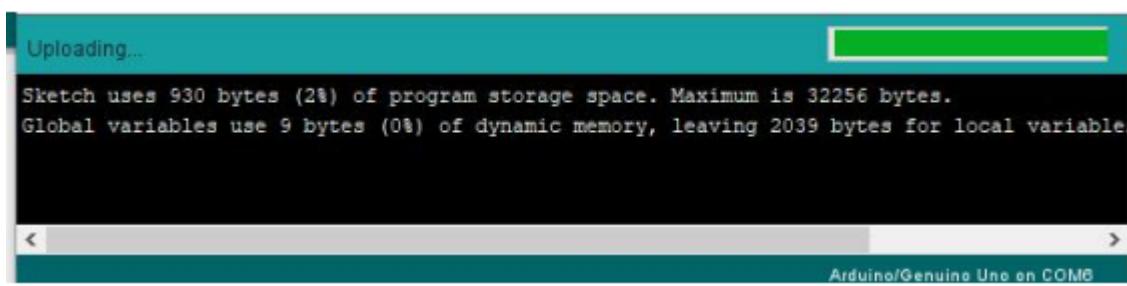
Нажмите на кнопку "Upload". Вторая кнопка слева на панели инструментов.



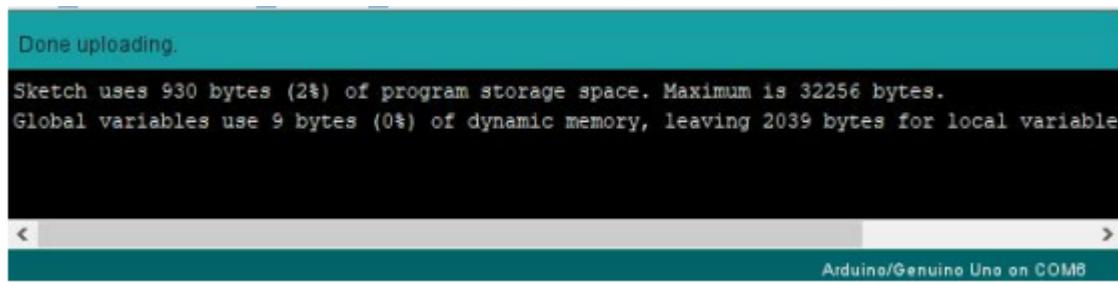
Если вы посмотрите на область состояния IDE, вы увидите индикатор выполнения и серию сообщений. Сначала в нем будет написано "Compiling Sketch...". Это преобразует эскиз в формат, подходящий для загрузки на доску.



Далее статус изменится на "Uploading". На этом этапе светодиоды на Arduino должны начать мигать по мере передачи эскиза.

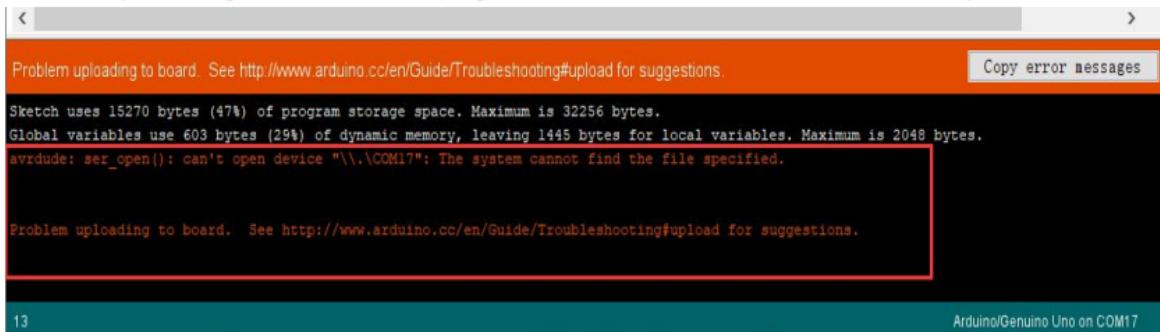


Наконец, статус изменится на "Done uploading".



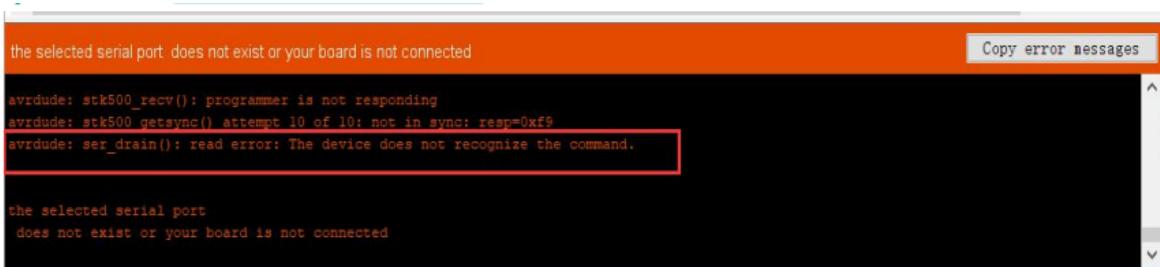
После загрузки кода вы увидите, что светодиод "L" на материнской плате Arduino UNO загорится на 1 секунду, а затем погаснет на 1 секунду, и процесс продолжит цикл. Вы можете менять это значение в самом коде, тем самым экспериментируя с ним.

Если вы пропустили некоторые шаги, Arduino IDE может сообщить о некоторых ошибках. Вы можете обратиться к следующим решениям.



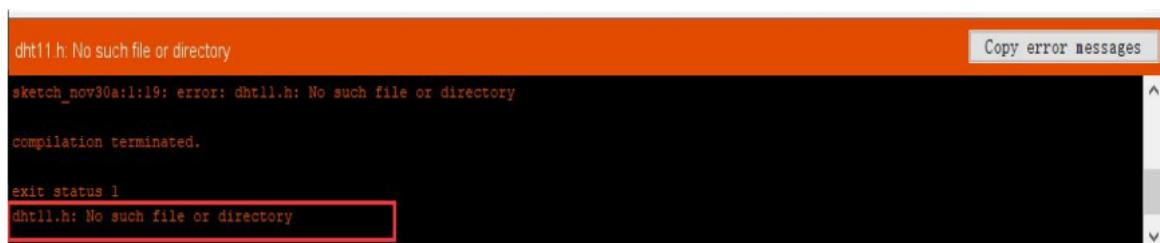
Problem uploading to board. See <http://www.arduino.cc/en/Guide/Troubleshooting#upload> for suggestions.  
Sketch uses 15270 bytes (47%) of program storage space. Maximum is 32256 bytes.  
Global variables use 603 bytes (29%) of dynamic memory, leaving 1445 bytes for local variables. Maximum is 2048 bytes.  
avrdude: ser\_open(): can't open device "\.\COM17": The system cannot find the file specified.  
  
Problem uploading to board. See <http://www.arduino.cc/en/Guide/Troubleshooting#upload> for suggestions.  
  
13 Arduino/Genuino Uno on COM17

Это означает, что ваша плата вообще не подключена, или драйверы не были установлены (при необходимости), или выбран неправильный серийный порт.



the selected serial port does not exist or your board is not connected  
Copy error messages  
avrdude: stk500\_recv(): programmer is not responding  
avrdude: stk500\_getsync() attempt 10 of 10: not in sync: resp=0xf9  
avrdude: ser\_drain(): read error: The device does not recognize the command.  
  
the selected serial port  
does not exist or your board is not connected

Это означает, что последовательный порт связи для загрузки кода занят. Если вы подключаете модуль Bluetooth при загрузке кода, вам необходимо отсоединить модуль Bluetooth от платы расширения Arduino UNO, а затем повторно установить модуль Bluetooth после загрузки. Другая возможность заключается в том, что USB-интерфейс Arduino UNO недостаточно питан, и вам необходимо включить внешний выключатель питания платы расширения.



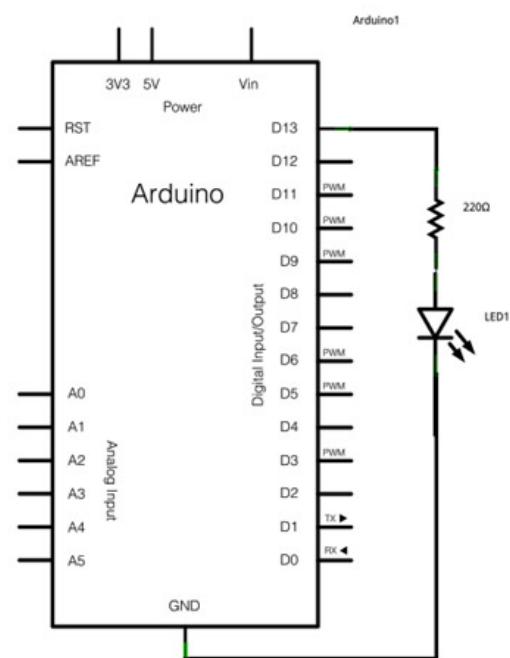
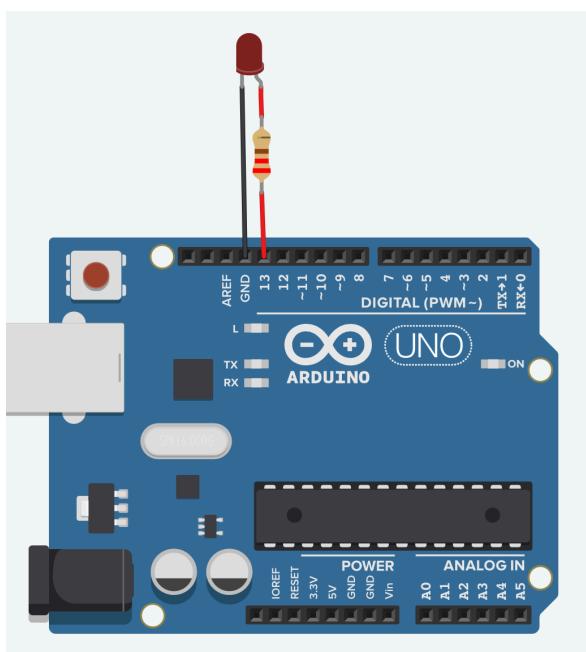
dht11.h: No such file or directory  
Copy error messages  
sketch\_nov30a:1:19: error: dht11.h: No such file or directory  
  
compilation terminated.  
  
exit status 1  
dht11.h: No such file or directory

Если вам нужно использовать файлы библиотеки в некоторых программах проекта, вы можете столкнуться с этой ошибкой. Это означает, что в коде используются некоторые библиотечные файлы, но вы не добавили соответствующие библиотечные файлы в Arduino перед загрузкой кода.

## 5. Тест на мигание внешнего светодиода

Чтобы построить электрическую цепь соедините один вывод резистора на 220Ом с контактом 13 на Arduino. Второй вывод резистора соедините с длинным выводом светодиода (анодом). Короткий вывод светодиода (катод) соедините с контактом GND Arduino. После этого соедините ваш Arduino с ПК, запустите среду Arduino и введите код, написанный ниже.

Большинство плат Arduino имеют встроенный светодиод присоединенный к выводу 13. Если вы запустите этот пример без внешнего светодиода то сможете наблюдать мигание встроенного светодиода. Код вы можете взять из общей папки "КОД" - **5 Тест на мигание внешнего светодиода**



```
// К контакту 13 подключен светодиод через 220 Ом, присвоим ему имя "led"
```

```
int led = 13;
```

```
// Процедура установки запускается один раз при нажатии кнопки сброса:
```

```
void setup(){
```

```
// инициализируем цифровой вывод как выход:
```

```
pinMode(led, OUTPUT);
```

```
}
```

```
// основной цикл:
```

```
void loop() {
```

```
  digitalWrite(led, HIGH); // включить светодиод (HIGH означает высокий уровень напряжения )
```

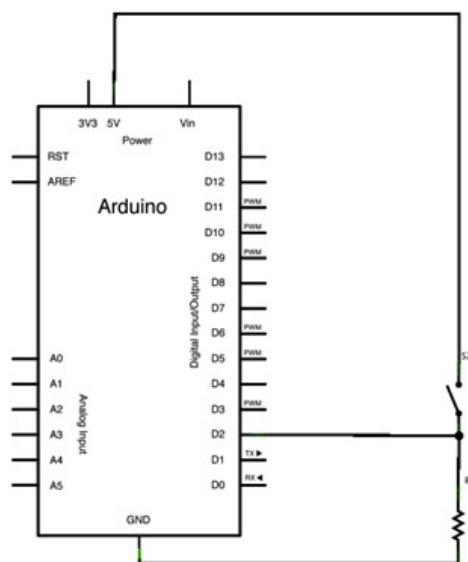
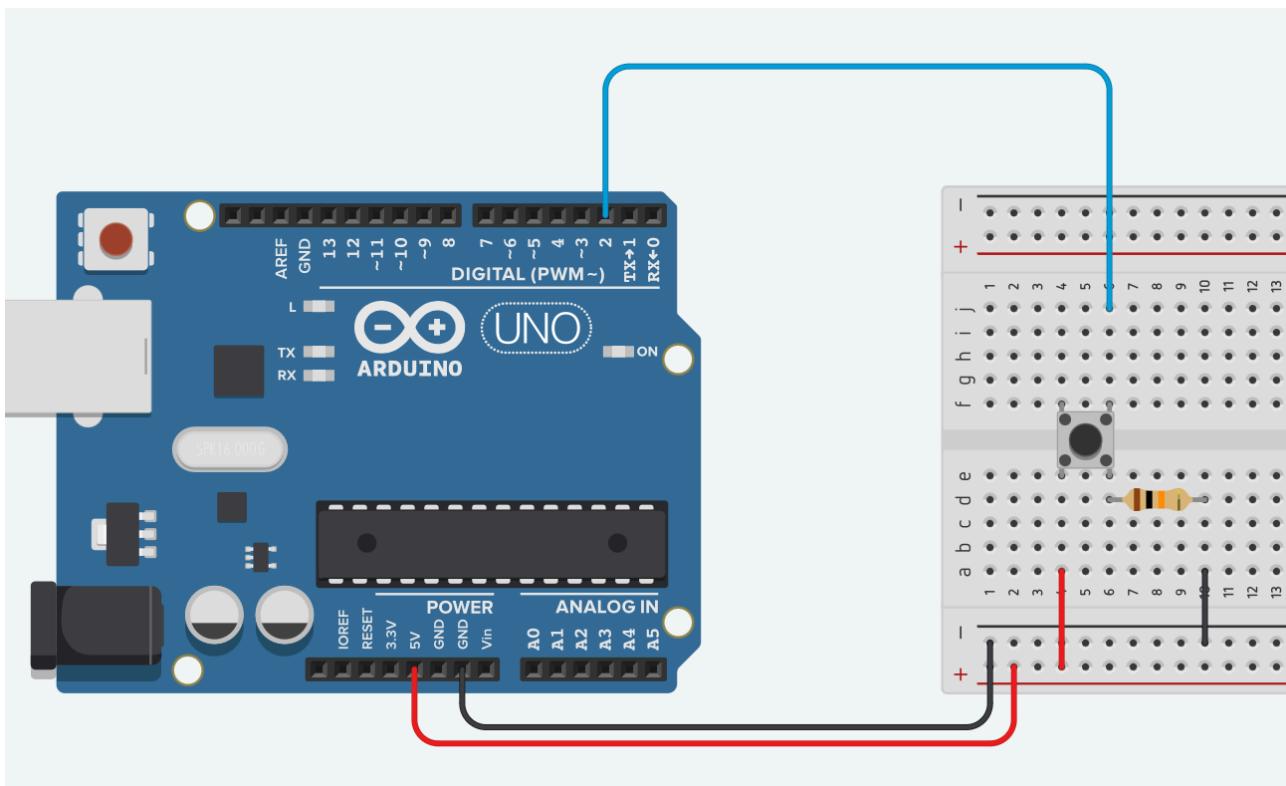
```
  delay(1000); // ждем секунду (1 000 мс)
```

```
digitalWrite(led, LOW); // выключить светодиод (LOW означает низкий уровень напряжения )  
delay(1000); //ждем секунду (1 000 мс)  
}
```

## **6. Чтение цифрового входа и вывод результатов на монитор**

Нам понадобится:

- плата Arduino
  - кнопка
  - резистор 10 кОм
  - макетная плата
  - провода для соединений



Соедините три провода с Arduino. Первые два (красный и черный) присоедините к двум длинным вертикальным рядам на краю макетной платы для получения доступа к 5 вольтам питания и земли. Третий вывод(синий цвет) идет от цифрового вывода 2 к одному из выводов кнопки. Этот же вывод кнопки подсоединен через стягивающий резистор (10кОм) к земле. Второй вывод кнопки подсоединен к 5 вольтам питания.

При нажатии кнопки соединяются две точки цепи. Когда кнопка не нажата - нет никакой связи между двумя ее выводами, так что вывод подключен к земле через стягивающий резистор и находится в состоянии LOW. При нажатии на кнопку устанавливается соединение между двумя выводами, подключается 5 вольт и вывод 2 переходит в состояние HIGH, или 1.

Если не использовать резистор и оставить вывод 2 неподключенным, то напряжение на нем будет изменяться хаотично. Это приведет к случайному формированию на выводе 2 напряжения высокого или низкого уровня. Поэтому необходим стягивающий резистор.

Код приведен ниже, а также в папке **КОД: 6. Чтение цифрового входа и вывод результатов на монитор**

Первым делом нужно в функции настройки включить последовательную передачу данных между ПК и Arduino со скоростью 9600 бит в секунду следующей строкой:  
**Serial.begin(9600);**

Далее инициализируем вывод 2. Он должен читать состояние кнопки, тобто быть настроенным как вход:

**pinMode(2,INPUT);**

Теперь, когда установки завершены, переходим к основному циклу вашего кода.

Когда кнопка нажата, 5 вольт будут свободно проходить через цепь, а когда она не нажата вход будет соединен с землей через резистор 10 кОм. Это цифровой вход, а значит он может быть только во включенном состоянии (читается вашим Arduino как 1 или HIGH) или в выключенном состоянии (читается вашим Arduino как 0 или LOW).

Промежуточных состояний не существует.

Первое что нужно сделать в основном цикле это создать переменную для хранения информации поступающей от кнопки. Поскольку сигнал, приходящий от кнопки может иметь только значения «1» или «0», вы можете использовать целочисленный тип данных int. Назовем эту переменную sensorValue и присвоим ей значение, равное значению считанному с входа 2. Вы можете сделать это с помощью следующей строки:

**int sensorValue = digitalRead(2);**

Как только Arduino считает значение на входе мы передадим его в ПК и выведем на

экран как десятичное значение. Вы можете сделать это используя команду `Serial.println()`, как показано в следующей строке:

**`Serial.println(sensorValue);`**

Теперь, когда вы откроете ваш монитор последовательной передачи (Serial Monitor в среде Arduino), вы сможете увидеть последовательность нулей, когда кнопка разомкнута и последовательность единиц, когда кнопка замкнута.

### **Полный текст программы:**

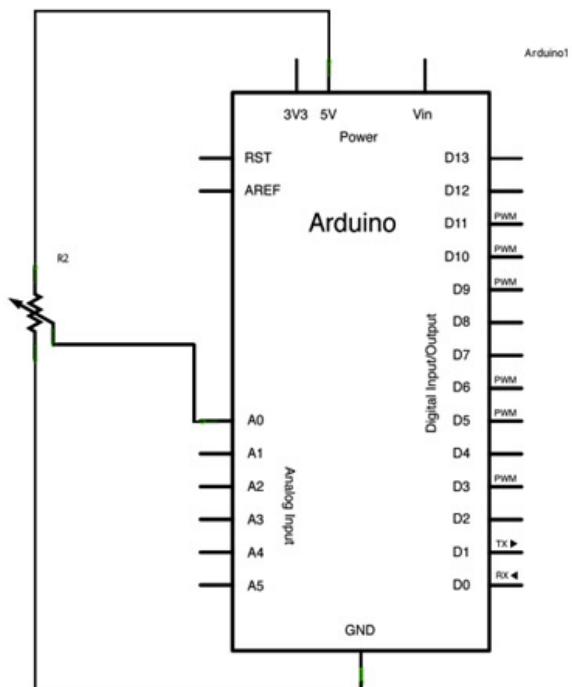
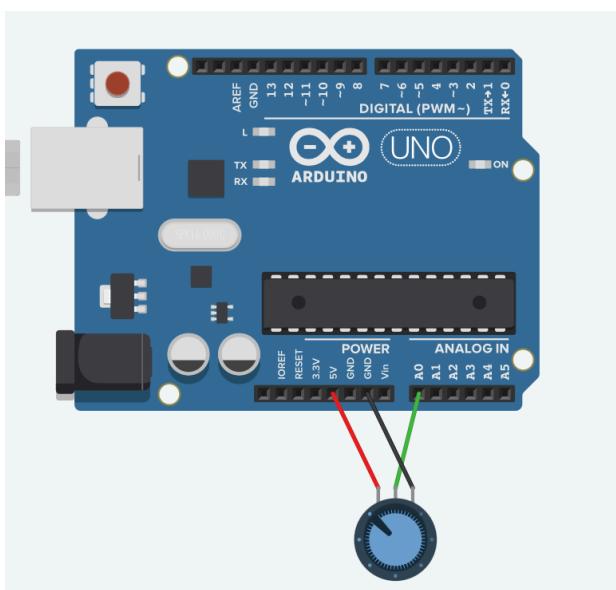
```
// цифровой вход 2 присоединен к кнопке. Назовем его:  
int pushButton = 2;  
  
// проведем необходимые установки:  
void setup() {  
    // инициализируем последовательную передачу данных со скоростью 9600 бит в секунду:  
    Serial.begin(9600);  
    // назначим вывод 2 входом:  
    pinMode(pushButton, INPUT);  
}  
  
// основной цикл:  
void loop() {  
    // читаем значение на входе:  
    int buttonState = digitalRead(pushButton);  
    // вывод и значение на монитор:  
    Serial.println(buttonState);  
    delay(1);      // задержка для стабильного считывания  
}
```

## 7. Чтение аналогового входа и вывод результатов на монитор

Нам понадобится:

- плата Arduino
- потенциометр 10 кОм

Присоедините три провода от потенциометра к вашему Arduino. Один из крайних выводов потенциометра соедините с землей (GND) Arduino. Второй крайний вывод потенциометра соедините с 5 вольтовым выходом Arduino. Третьим соедините аналоговый вход 0 и средний вывод потенциометра.



При вращении вала потенциометра, можно изменять сопротивление по обе стороны от центрального вывода. Это приводит к изменению напряжения на центральном выводе. Когда напряжение между центральным выводом и выводом подключенным к 5 вольтам близится к нулю (а сопротивление с другой стороны близится к 10 кОм), напряжение на центральном выводе приближается к 5 вольтам. Если повернуть ручку в другую сторону, то сопротивления поменяются местами и напряжение на центральном контакте приблизится к 0 вольтам. Это напряжение является аналоговым сигналом который мы будем считывать аналоговым входом.

Arduino имеет встроенный аналого-цифровой преобразователь, который считывает значения напряжения и преобразовывает их в числа от 0 до 1023.

Если ручка потенциометра повернута до упора в одну сторону, то на вход подается 0 вольт и входное значение равно 0. Если ручка потенциометра повернута до упора в другую сторону, то на вывод подается 5 вольт и входное

значение равно 1023. В промежуточных положениях ручки `analogRead()` возвращает число между 0 и 1023, которое пропорционально напряжению на среднем выводе.

Код приведен ниже, а также в папке **КОД: 7. Чтение аналогового входа и вывод результатов на монитор**

В данной программе, единственное что нужно сделать в функции установки - начать последовательную передачу между Arduino и ПК со скоростью 9600 бит данных в секунду командой:

**Serial.begin(9600);**

Далее, в основном цикле программы, вы должны создать переменную для хранения значения напряжения (которая может изменяться от 0 до 1023, лучше всего подойдет тип `int`) которое приходит с потенциометра.

**int sensorValue = analogRead(A0);**

Наконец, вам нужно наблюдать эту информацию на мониторе в виде десятичных (DEC) значений. Вы можете сделать это с помощью команды `Serial.println()` с помощью следующей строки кода:

**Serial.println(sensorValue, DEC)**

Теперь, открыв Serial Monitor в вашей среде Arduino, вы можете увидеть поток цифр от нуля до 1023, соответствующих положению ручки потенциометра. Если вы повернете ручку эти показания изменятся практически мгновенно.

**Полный текст программы:**

```
// установки:  
void setup() {  
    // инициализируем последовательную передачу данных со скоростью 9600 бит в  
    // секунду:  
    Serial.begin(9600);  
}  
// основной цикл:  
void loop() {  
    // читаем значение на аналоговом входе 0:  
}
```

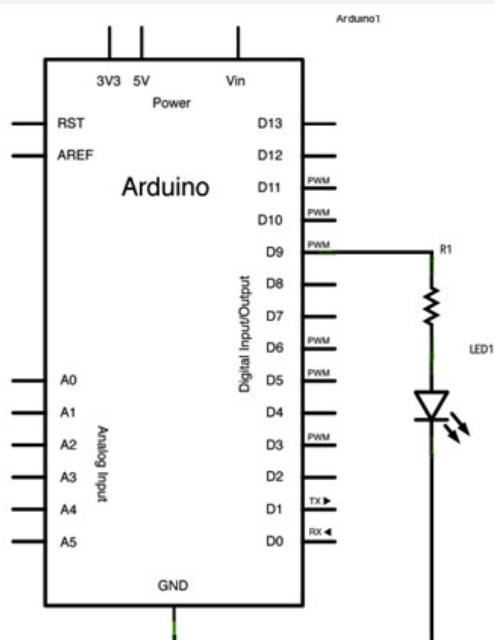
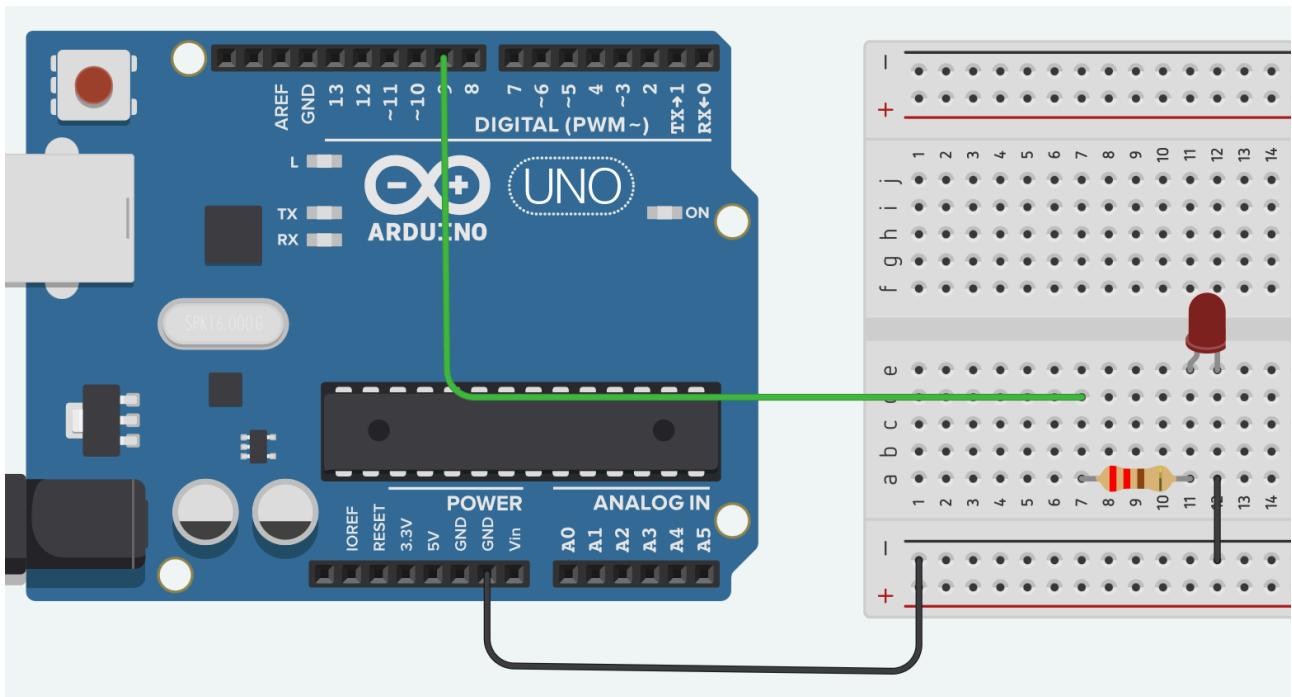
```
int sensorValue = analogRead(A0);  
// выводим на монитор считанное значение:  
Serial.println(sensorValue);  
delay(1); // задержка в промежутке между считываниями для стабильности  
}
```

## 8. Регулирование яркости светодиода

Нам понадобится:

- плата Arduino
- светодиод
- резистор 220 Ом
- макетная плата
- провода для соединений

Соедините анод (длинный вывод) вашего светодиода с цифровым выводом 9 Arduino через резистор 220 Ом. Соедините катод (короткий вывод) с землей (GND) Arduino.



Код приведен ниже, а также в папке **КОД: 8. Регулирование яркости светодиода**

В функции `setup()` нужно назначить вывод 9 выходом.

Функция `analogWrite()` которую вы будете использовать в основном цикле имеет два аргумента: первый говорит функции какой вывод использовать, во второй записывают значение для ШИМ.

Для того чтобы плавно увеличить яркость светодиода, а потом плавно уменьшить вам нужно сначала увеличивать значение ШИМ от 0 (светодиод выключен) до 255 (максимальная яркость), а потом наоборот. В нашей программе переменная отвечающая за значение ШИМ будет называться `brightness`. В каждом цикле эта переменная будет изменяться на значение `fadeAmount`.

Как только `brightness` достигнет значения 255 или 0 `fadeAmount` изменит свой знак. Таким способом мы сможем поменять увеличения яркости на понижение и наоборот.

`analogWrite()` изменяет значение ШИМ очень быстро, потому нужна задержка для контроля скорости изменения яркости. Вы можете сами изменять значения задержки и смотреть как это скажется на работе.

Полный текст программы:

```
int led = 9; //выводсветодиода 9
int brightness = 0; // переменная отвечающая за яркость
int fadeAmount = 5; // переменная, которая задает скорость изменения
                    // яркости за цикл

// настройки:
void setup() {
    // назначим вывод 9 выходом:
    pinMode(led, OUTPUT);
}

// основной цикл:
void loop() {
    // устанавливаем яркость светодиода на выводе 9:
```

```
analogWrite(led, brightness);
```

```
// изменим значение яркости для следующего прохода цикла:
```

```
brightness = brightness + fadeAmount;
```

```
// поменяем направление изменения яркости:
```

```
if (brightness == 0 || brightness == 255) {
```

```
    fadeAmount = -fadeAmount ;
```

```
}
```

```
// ждем 30 миллисекунд для наблюдения эффекта диммирования:
```

```
delay(30);
```

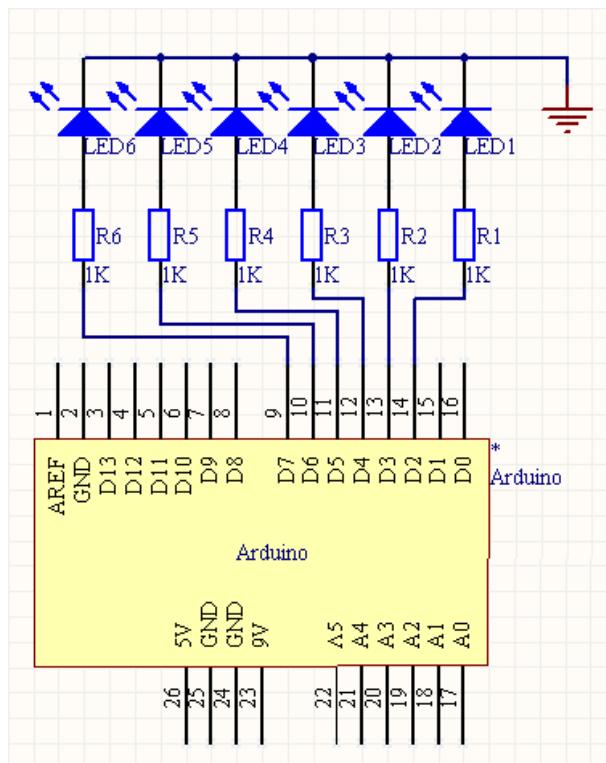
```
}
```

## 9. Опыт с бегущими огнями на 6-ти светодиодах

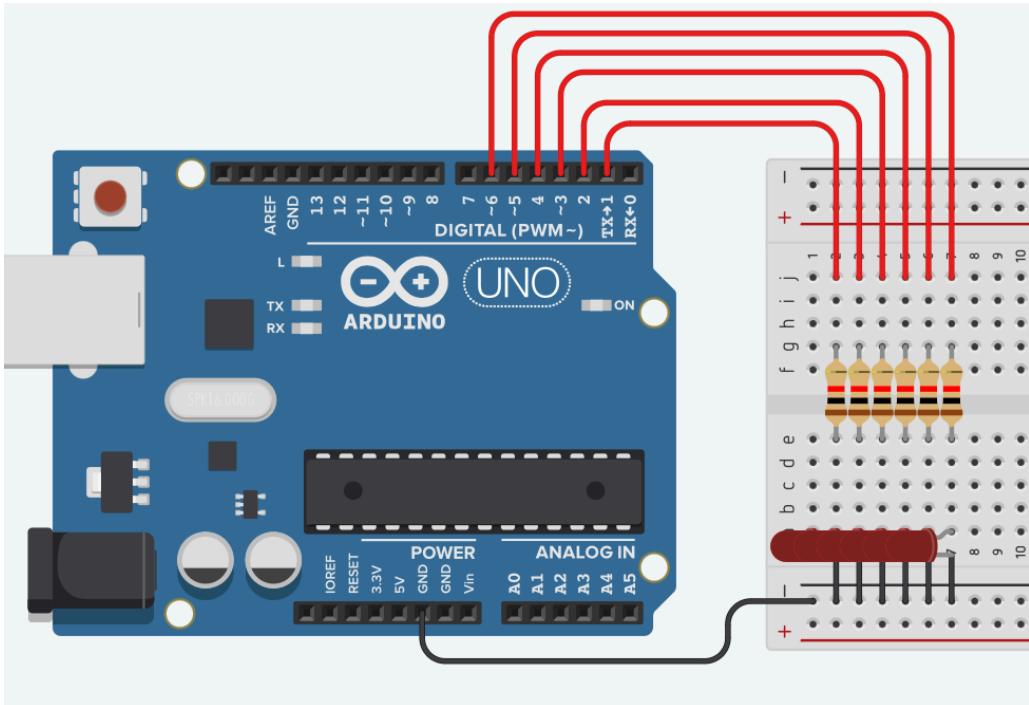
Нам понадобится:

- плата Arduino
- светодиод \*6 штук
- резистор 1 кОм \* 6 штук
- макетная плата
- провода для соединений

Цифровые выводы I/O Arduino располагаются с одной стороны микроконтроллера, сверху и снизу находится по 6 контактов (пинов), это пины с 2-ого по 7-ой и пины с 8-ого по 13-ый. В 13-ый пин мы подключаем резистор 1 кОм, а все остальные напрямую подключаем к контроллеру ATmega. Чтобы проверить функциональность всех выводов I/O Arduino, мы используем бегущие огни на шести светодиодах. Принципиальная схема показана ниже:



Чтобы увеличить сопротивление светодиодов и исключить вероятность выхода светодиодов из строя, в каждую ветвь мы подключаем резисторы 1 кОм.



Код приведен ниже, а также в папке **КОД: 9. Опыт с бегущими огнями на 6-ти светодиодах**

```
//определяем все 6 управляемых контактов на ардуино
int Led1 = 1;
int Led2 = 2;
int Led3 = 3;
int Led4 = 4;
int Led5 = 5;
int Led6 = 6;

void lamp(void) {
    unsigned char j;
    for(j=1; j<=6; j++) //через цикл проходимся по всем 6 светодиодам
    {
        digitalWrite(j,HIGH); //каждому светодиоду присваиваем "HIGH", тем самым он
загорается
        delay(200); //делаем задержку в 200 Мс, что потухание было плавное
    }
    for(j=6; j>=1; j--) //снова делаем цикл, чтобы пройтись по светодиодам и их выключить
    {
        digitalWrite(j,LOW); //каждому светодиоду в обратном порядке присваиваем "LOW", тем
самым его выключая
        delay(200); //снова делаем задержку в 200 Мс, что потухание было плавное
    }
}

void setup() {
    unsigned char i;
    for(i=1;i<=6;i++)// пины 1-6 работают как выходные
```

```
pinMode(i, OUTPUT); // i-й пин работает как выходной  
}  
void loop()  
{  
    lamp();  
}
```

Вы можете поменять цикл, задержку, сделать другую логику, тем самым придумав свой уникальную комбинацию работы.

## 10. Работа с зуммером

Нам понадобится:

- плата Arduino
- зуммер со встроенным генератором
- зуммер без генератора
- макетная плата
- провода для соединений

Следующий опыт посвящен работе с зуммером без источника питания.

Рабочее напряжение зуммера 5 В, что совпадает с выходным напряжением цифрового разъема на Arduino, поэтому зуммер можно подключать напрямую без дополнительного резистора.

Коротко ознакомимся с зуммером.

Из-за небольших размеров (диаметр всего 6 мм), маленького веса, низкой цены и надежной конструкции миниатюрные зуммеры широко используются в электрооборудовании, электронике, микросхемах и любых других устройствах для звукового оповещения. Зуммеры бывают со встроенными генераторами и без них.

Зуммер со встроенным генератором:



Зуммер без генератора:



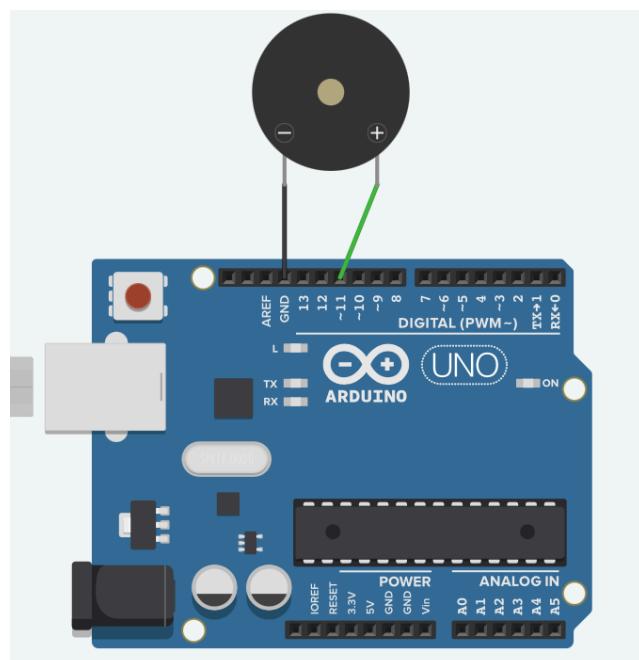
Внешне эти два типа почти идентичны, но есть разница в подключении зуммеров к пинам микроконтроллера.

Зуммер с зеленой микросхемой – без генератора.

Зуммер с черной крышкой и без микросхемы – со встроенным генератором. Визуально сложно отличить устройства, самый надежный способ проверки, кроме, разумеется, прочтения инструкции, измерить сопротивление зуммера с помощью универсального мультиметра. У зуммера без генератора сопротивление 8 Ом или 16 Ом, у зуммеров со встроенным генератором сопротивление в сотни раз выше.

Зуммер со встроенным генератором при прямом подключении к номинальному напряжению (все новые устройства имеют наклейки с пояснениями) будут непрерывно издавать звук. Зуммеры без генератора, как и электромагнитные динамики, нужно подключать в отдельную аудио-цепь

Подключите зуммер, как на картинке ниже:



Код приведен ниже, а также в папке **КОД: 10. Работа с зуммером**

```
int buzzer=11; //переменной buzzer присваиваем номер выходного пина (11)
```

```
void setup()
{
    pinMode(buzzer,OUTPUT);//11 пин будет работать как выходной
}

void loop()
{
    unsigned char i,j;//задаем переменные
    while(1) //цикл while
    {
        for(i=1;i<80;i++)//выходная частота звука
        {
            digitalWrite(buzzer,HIGH);//зуммер издает звук, за счет HIGH
            delay(1);//задержка 1 ms
            digitalWrite(buzzer,LOW);// зуммер не издает звук, за счет LOW
            delay(1);// задержка 1 ms
        }
        for(i=0;i<100;i++)//другая выходная частота звука
        {
            digitalWrite(buzzer,HIGH);// зуммер издает звук, за счет HIGH
            delay(2);// задержка 2 ms
            digitalWrite(buzzer,LOW);// зуммер не издает звук, за счет LOW
            delay(2);// задержка 2ms
        }
    }
}
```

## 11. Опыт с цифровым индикатором

Нам понадобится:

- плата Arduino
- цифровой индикатор модели SM41056 (семисегментный LED индикатор)
- резистор 1 КОм
- макетная плата
- провода для соединений

Цифровой индикатор – это полупроводниковое показывающее устройство на светодиодах. Цифровые индикаторы делятся на 7-сегментные и 8-сегментные и отличаются количеством светодиодов (в 7-сегментном на один светодиод и на один символ меньше, чем в 8-сегментном).

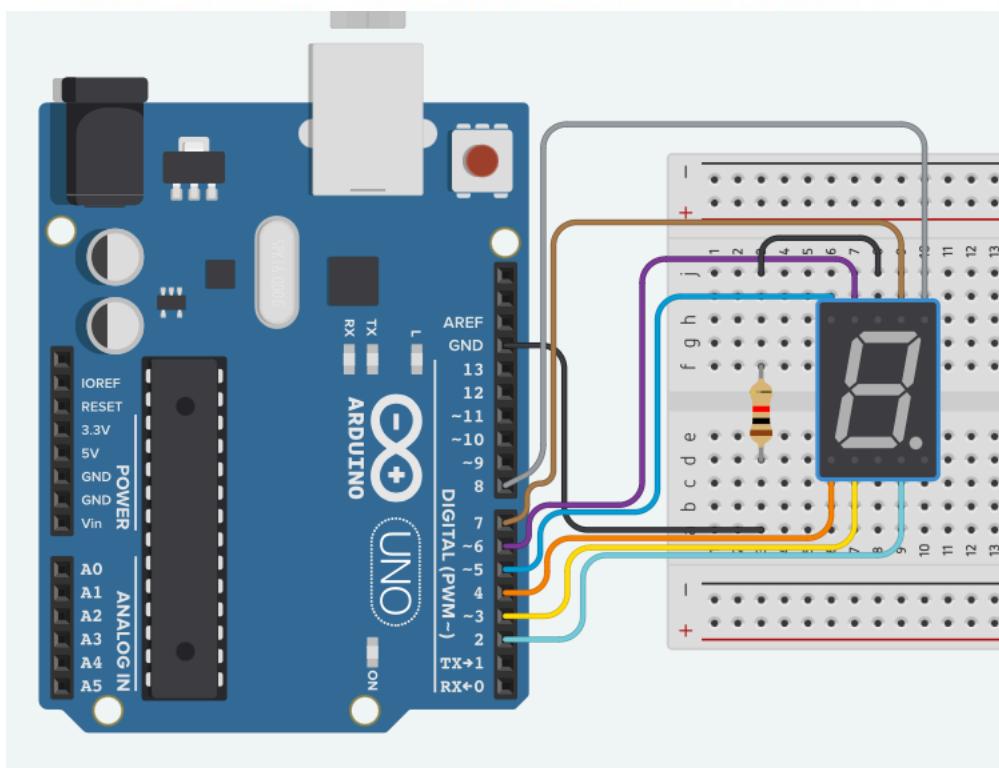
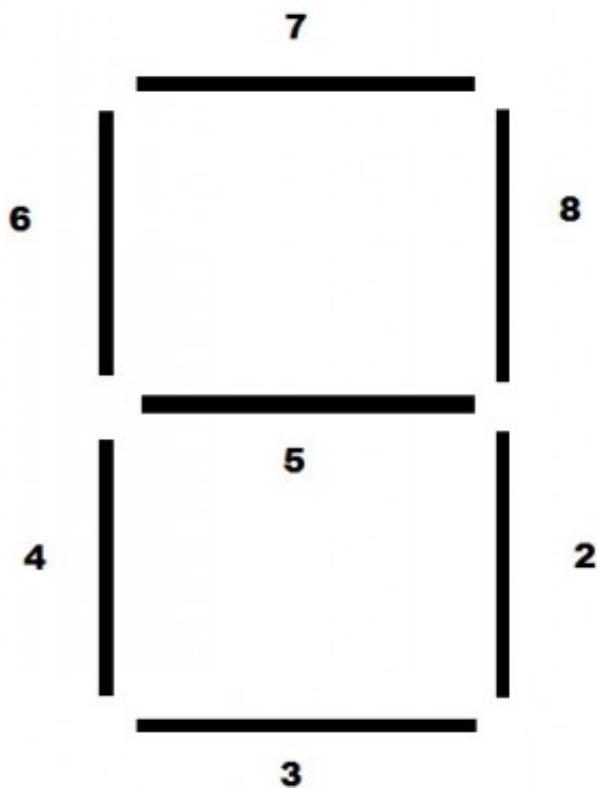
Цифровые индикаторы имеют схемы включения с общим катодом и общим анодом. У индикаторов с общим анодом общие выводы (аноды) подключаются к СОМ-порту. При использовании индикатора с общим катодом общий контакт «PWR» подключается к разъему «PWR» источника питания. Когда на общий катод какого-либо сегмента цифрового индикатора подается мощность низкого уровня, сегмент начинает гореть, а когда подается мощность высокого уровня, соответствующий сегмент не горит. У индикаторов с общим катодом все устроено наоборот, общие катоды образуют один катод, общие аноды подключаются отдельно.

Для начала познакомимся с цифровым индикатором для данного опыта.

Ниже указана схема подключения цифрового индикатора модели SM41056 с высотой символа 0,5”.

Как указано на обратной стороне индикатора, по четырем сторонам устройства имеются 2 питающих входа и два промаркированных пина 5, 10. Отдельно обозначены контактные штырьки 1, 6, 5, 10. Цифровой индикатор, как и светодиоды, нуждается в дополнительном сопротивлении при подключении.

Сначала мы заставим загораться и гаснуть сегменты по очереди. Первый сегмент у нас подключен к выводу 2, второй к 3, и так далее. Если изменим состояние выхода `digitalWrite(led2, HIGH);` на высокое (HIGH) то у нас загорится сегмент с номером 2.



Скорость переключения вы можете за счет переменной  $i$ , сейчас таймаут составляет 200 Мс.

Код приведен ниже, а также в папке **КОД: 11-1. Опыт с цифровым индикатором**

```
const int led2 = 2;
const int led3 = 3;
const int led4 = 4;
const int led5 = 5;
const int led6 = 6;
const int led7 = 7;
const int led8 = 8;

void setup() {
    pinMode(led2, OUTPUT);
    pinMode(led3, OUTPUT);
    pinMode(led4, OUTPUT);
    pinMode(led5, OUTPUT);
    pinMode(led6, OUTPUT);
    pinMode(led7, OUTPUT);
    pinMode(led8, OUTPUT);
}

void loop() {
    int i = 200;
    digitalWrite(led2, HIGH);
    delay(i);
    digitalWrite(led2, LOW);
    delay(i);
    digitalWrite(led3, HIGH);
    delay(i);
    digitalWrite(led3, LOW);
    delay(i);
    digitalWrite(led4, HIGH);
    delay(i);
    digitalWrite(led4, LOW);
    delay(i);
    digitalWrite(led5, HIGH);
    delay(i);
    digitalWrite(led5, LOW);
    delay(i);
    digitalWrite(led6, HIGH);
    delay(i);
    digitalWrite(led6, LOW);
    delay(i);
    digitalWrite(led7, HIGH);
    delay(i);
    digitalWrite(led7, LOW);
    delay(i);
    digitalWrite(led8, HIGH);
    delay(i);
```

```
    digitalWrite(led8, LOW);
    delay(i);
}
```

Сейчас сделаем опыт уже с последовательным загоранием чисел, нам понадобится такая же схема, как в предыдущем. Скопируйте код из файла:

**КОД: 11-2. Опыт с цифровым индикатором**

Вы можете менять задержку через переменную *i*, а также изменить логику программы под себя.

## 12. Опыт с кнопкой

Нам понадобится:

- плата Arduino
- макетная плата
- провода для соединений
- кнопка
- резистор 220 Ом
- резистор 10 КОм

Кнопка – один из самых распространенных элементов, используется для замыкания и размыкания электрических цепей, для управления электродвигателями или для включения/выключения устройств. Внешний вид кнопок может быть совершенно разным. В данном опыте используется микрокнопка 6 мм, показанная ниже.

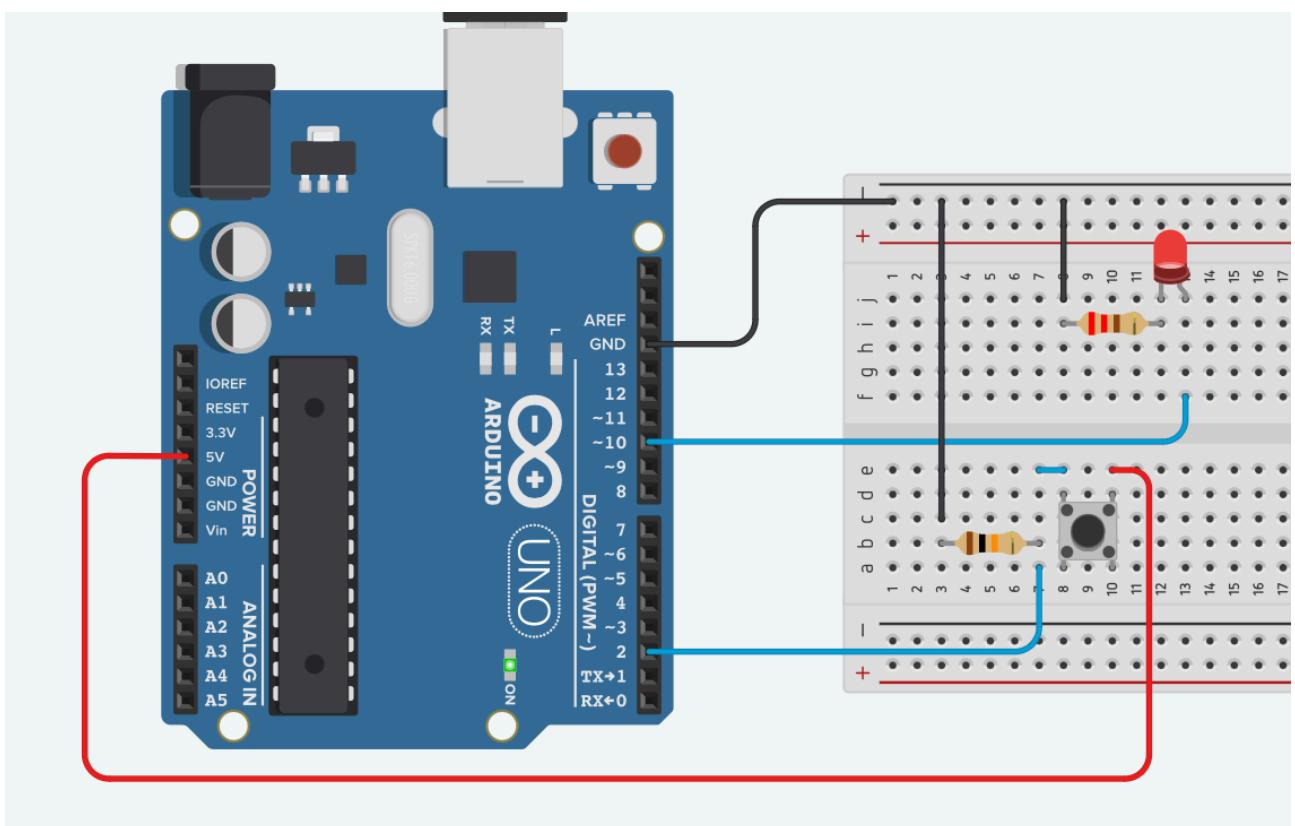


Когда кнопка не вжата, ножки 1-2 и 3-4 соединены попарно. Когда кнопки вжата, все ножки 1,2,3,4 соединяются между собой.

В данном эксперименте мы будем использовать контакт D2 Arduino в качестве входа. Это позволяет подключить к нему кнопку для взаимодействия с проектом в режиме реального времени. При использовании Arduino в качестве входов используют pull-up- и pulldown-резисторы, чтобы вход Arduino не находился в «подвешенном» состоянии, а имел заранее известное состояние (0 или 1). Резисторы pull-up подтягивают вход к питанию +5 В, pull-down-резисторы подтягивают вход к GND. Кроме этого, pull-up- и pull-down-резисторы

гарантируют, что кнопка не создаст короткого замыкания между +5 В и землей при нажатии. В нашем эксперименте для подключения кнопки мы будем использовать pulldown-резистор.

Когда кнопка отключена, вход D2 будет подтянут к «земле» через резистор номиналом 10 кОм, который будет ограничивать поток тока, и на входном контакте будет установлено значение напряжения LOW. При нажатии на кнопку входной контакт напрямую связан с 5 В. Большая часть тока будет протекать по пути наименьшего сопротивления через замкнутую кнопку, и на выходе генерируется уровень HIGH. При нажатии на кнопку включаем светодиод, при отпускании – гасим.



Код приведен ниже, а также в папке **КОД: 12. Опыт с кнопкой**

```
const int LED=10; // вывод для подключения светодиода 10 (D10)
void setup()
{
// Конфигурируем вывод подключения светодиода как выход (OUTPUT)
pinMode(LED, OUTPUT);
}
void loop()
{
// включаем светодиод, подавая на вывод 1 (HIGH)
digitalWrite(LED,HIGH);
// пауза 1 сек (1000 мс)
delay(1000);
```

```
// выключаем светодиод, подавая на вывод 0 (LOW)
digitalWrite(LED,LOW);
// пауза 1 сек (1000 мс)
delay(1000);
}
```

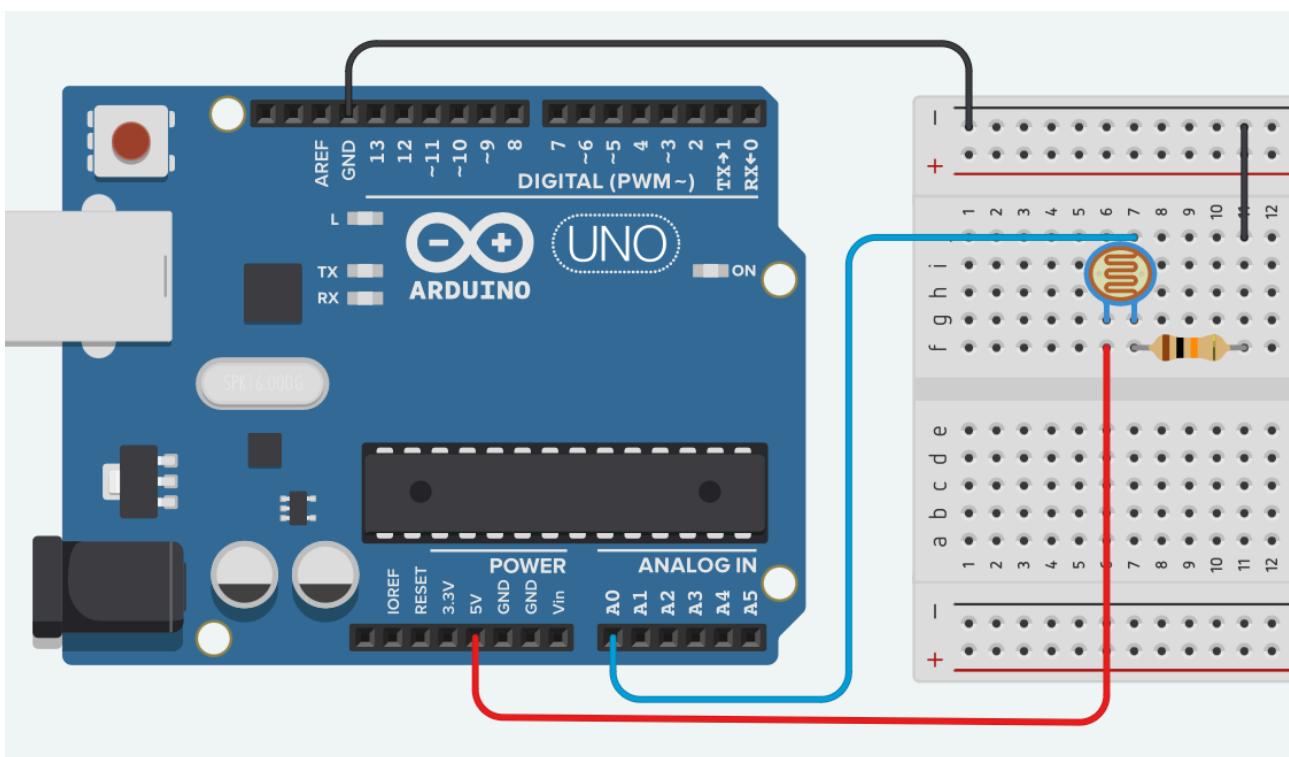
## 13. Опыт с фоторезистором

Нам понадобится:

- плата Arduino
- макетная плата
- провода для соединений
- фоторезистор
- резистор 10 КОм

Распространённое использование фоторезистора – измерение освещённости. В темноте его сопротивление растёт. Когда на фоторезистор попадается свет, сопротивление падает пропорционально освещённости.

Для измерения освещённости необходимо собрать данную схему, в котором верхнее плечо будет представлено фоторезистором, нижнее – обычным резистором достаточного большого номинала. Будем использовать резистор 10 кОм. Среднее плечо делителя подключаем к аналоговому входу A0 Arduino. Соберите схему, как ниже на рисунке.



Код приведен ниже, а также в папке **КОД: 13-1. Опыт с фоторезистором**

```
int light; // переменная для хранения данных с фоторезистора
void setup()
{
Serial.begin(9600);
}
void loop()
{
```

```

light = analogRead(0);
Serial.println(light);
delay(100);
}

```

Теперь вы можете смотреть в последовательном порте данные, которые выдает фоторезистор. Регулировать значения вы можете с помощью своей руки, накрывая фоторезистор, вы уменьшаете свет, который на него попадает и тем самым сопротивление растет. Убирая руку, оно снова падает.

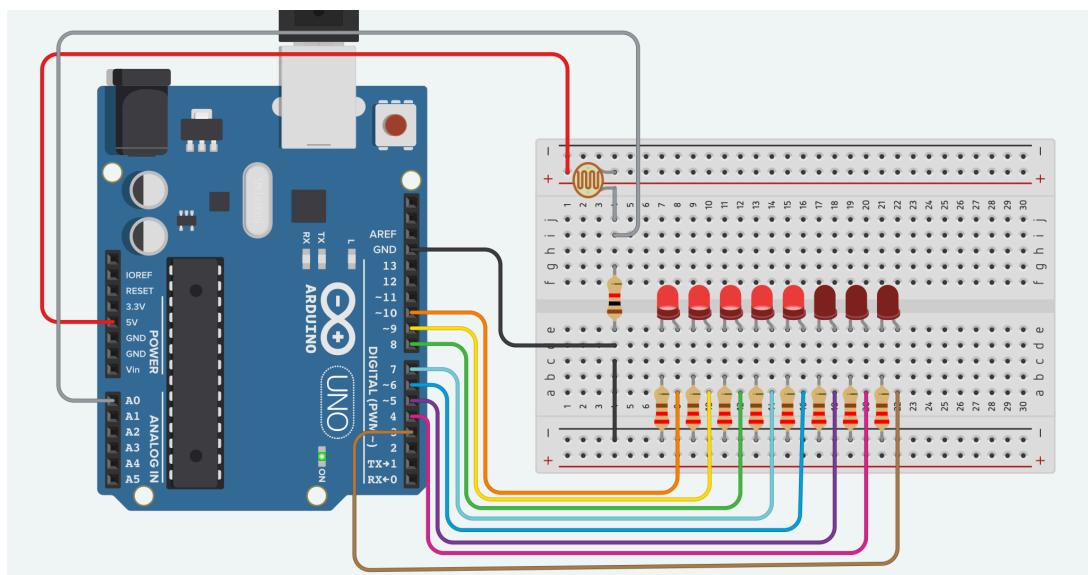
Давайте сделаем немного интереснее проект, добавив в него пару лампочек. Этот опыт будет чем-то похож на опыт из 9 опыта.

Сейчас создадим индикатор освещенности с помощью светодиодного ряда из 8 светодиодов. Количество горящих светодиодов пропорционально текущей освещенности. На самих светодиодах используем ограничительные резисторы номиналом 220 Ом. Регулируя рукой освещенность фоторезистора мы можем определить по количеству светодиодов текущий уровень освещенности.

Нам понадобится:

- плата Arduino
- макетная плата
- провода для соединений
- фоторезистор
- резистор 10 КОм \* 1шт
- резистор 220 Ом \* 8шт
- светодиод\*8 штук

Создайте схему, как на фото ниже:



Код приведён ниже, а также в папке **КОД: 13-2. Опыт с фоторезистором**

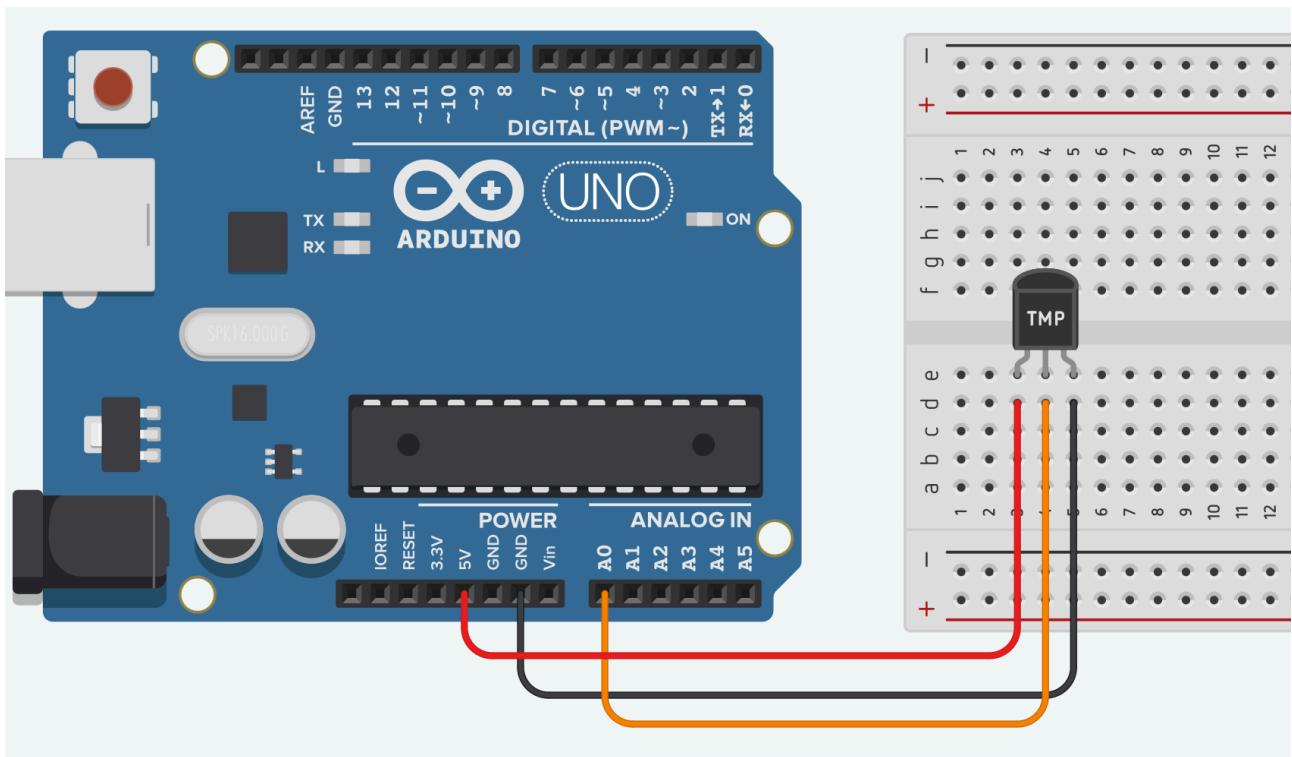
```
// Контакт подключения светодиодов
const int leds[]={3,4,5,6,7,8,9,10};
const int LIGHT=A0; // Контакт A0 для входа фоторезистора
const int MIN_LIGHT=200; // Нижний порог освещенности
const int MAX_LIGHT=900; // верхний порог освещенности
// Переменная для хранения данных фоторезистора
int val = 0;
void setup()
{
// Сконфигурировать контакты светодиодов как выход
for(int i=0;i<8;i++) // делаем через цикл for
pinMode(leds[i],OUTPUT);
}
void loop()
{
val = analogRead(LIGHT); // Считываем показаний фоторезистора
// Применение функции map()
val = map(val, MIN_LIGHT, MAX_LIGHT, 8, 0);
// ограничиваем, чтобы не превысило границ
val = constrain(val, 0, 8);
// зажечь кол-во светодиодов, пропорциональное освещенности,
// остальные потушить
for(int i=1;i<9;i++)
{
if(i>=val) // зажечь светодиоды
digitalWrite(leds[i-1],HIGH);
else // потушить светодиоды
digitalWrite(leds[i-1],LOW);
}
delay(1000); // пауза перед следующим измерением
}
```

## 14. Опыт с датчиком температуры LM35

Нам понадобится:

- плата Arduino
- макетная плата
- провода для соединений
- Датчик температуры LM35

Датчик аналоговый, а значит на его выходе мы имеем не 0 или 1, а напряжение в диапазоне от 0 до 5 вольт. Следовательно, мы должны вспомнить раздел про аналого-цифровое преобразование (АЦП) сигналов в Arduino. Держа в уме, что у Ардуино Уно есть шесть аналоговых входов (A0-A5), подключаем наш датчик по следующей схеме:



Вид на датчик снизу приведен ниже:



**PIN 1, +V<sub>S</sub>; PIN 2, V<sub>OUT</sub>; PIN 3, GND**

Подключив датчик температуры к Ардуино, тестируем нашу программу. Первое что мы сделаем, это выведем необработанный сигнал с аналогового входа в последовательный порт, для того чтобы просто понять, как меняется значение на входе A0. Нам понадобится простая программа:

Код приведён ниже, а также в папке **КОД: 14. Опыт с датчиком температуры LM35**

```
int raw = 0;
float temp = 0;

void setup() {
    Serial.begin(9600);
    pinMode(A0, INPUT );
}

void loop() {
    raw = analogRead(A0);
    temp = (temp = (((raw*5.0)/1024.0)-0.5)*100; //эта формула
нужна для правильного преобразования аналогового сигнала в градусы
Цельсия
    Serial.println(temp);
    delay(1000);
}
```

Запустив код, мы увидим, что датчик измеряет окружающую температуру. Измерять он может её с задержками - это нормально. Вы можете зажать датчик пальцами и выводимое числа в последовательном порте будут расти. Чтобы ускорить время обновления температуры - просто изменит последнюю строчку с Delay (1000)

## **15. Опыт с управлением сервоприводом**

Нам понадобится:

- плата Arduino
- макетная плата
- провода для соединений
- сервопривод sg90

Сервопривод : двигатели постоянного тока (ДПТ), редуктор с валом, датчики и цепь управления образуют автоматическую систему управления. С помощью сигнала управления задается угол вращения объекта. У сервоприводов максимальный угол вращения (например, 180 градусов) зависит от ДПТ. Если ротор ДПТ делает несколько оборотов, то вал сервопривода проворачивается на несколько градусов, он не может сделать полный оборот (цифровые сервоприводы с режимами сервопривода и двигателя такой проблемы не имеют). В стандартных ДПТ не предусмотрено обратной связи по углу вращения ротора, а в сервоприводах она есть. Назначение сервопривода так же отличается: стандартный ДПТ используется для создания механического вращающего момента, а сервоприводы используются для управления углом поворота объекта (например, в робототехнике).

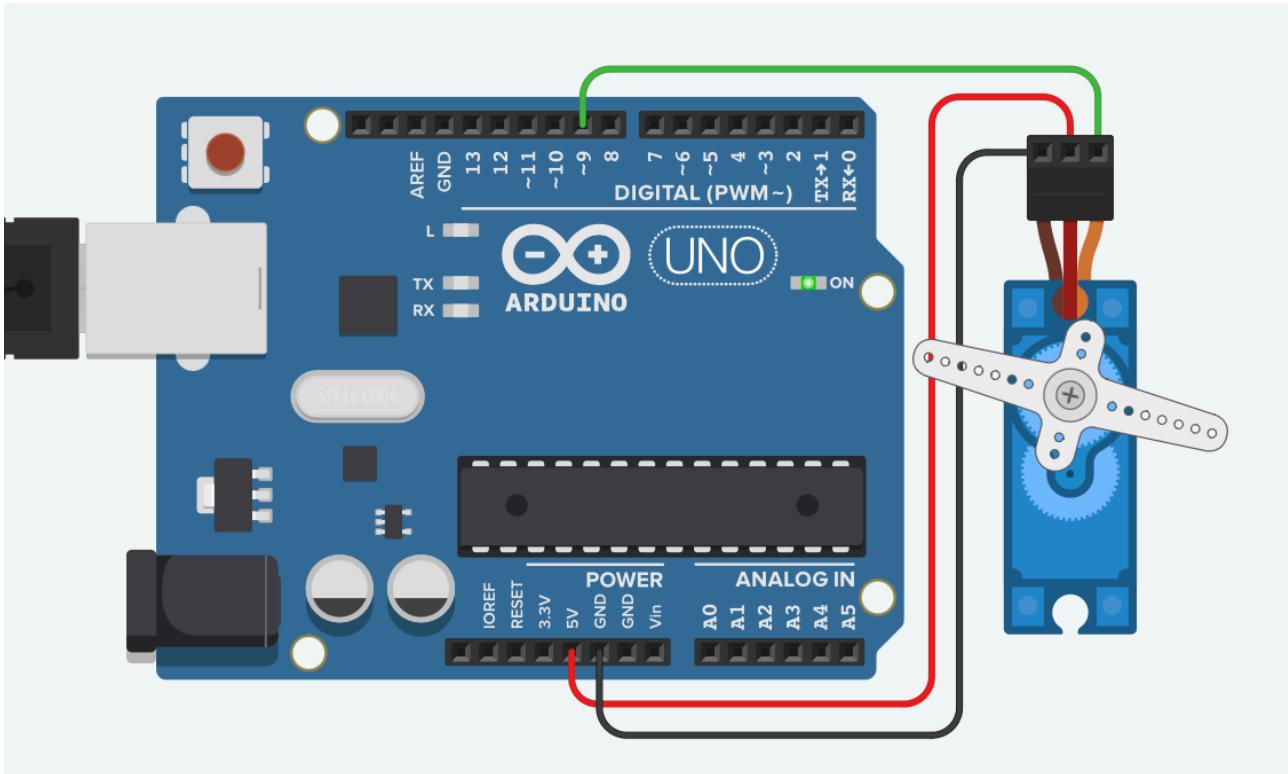
### **Принцип работы:**

Управляющий сигнал, с помощью устройств приема и передачи, поступает на микросхему с модулятором сигнала, и создается напряжение смещения постоянного тока. Внутри сервопривода есть стандартная электрическая цепь с периодом 20 мс и шириной стандартного сигнала 1,5 мс, на которую поступает напряжение смещения постоянного тока и сравнивается с напряжением потенциометра, в результате чего на выходе микросхемы появляется разность напряжений. В конце разность напряжений с положительного и отрицательного выводов поступает на управляющую двигателем микросхему и в зависимости от закона управления задаются прямое и обратное вращения двигателя.

В нашем примере сервопривод будет делать полный оборот в одну сторону до упора. Изменять шаг сервопривода через переменную servoPosition, либо меняя задержку в самом конце "delay".

Вы можете изменить код на свой выбор, добавив различные фишki.

Подключаем сервопривод по следующей схеме:



Код приведён ниже, а также в папке **КОД: 15. Опыт с управлением сервоприводом**

```
int servoPin=9;
int servoPosition=4000;

void setup() {
pinMode(servoPin, OUTPUT);
}

void loop() {
digitalWrite(servoPin, HIGH);
delayMicroseconds(servoPosition);
digitalWrite(servoPin, LOW);
delay(20);
}
```

## 16. Опыт с управлением Инфракрасный приемник сигналов

Нам понадобится:

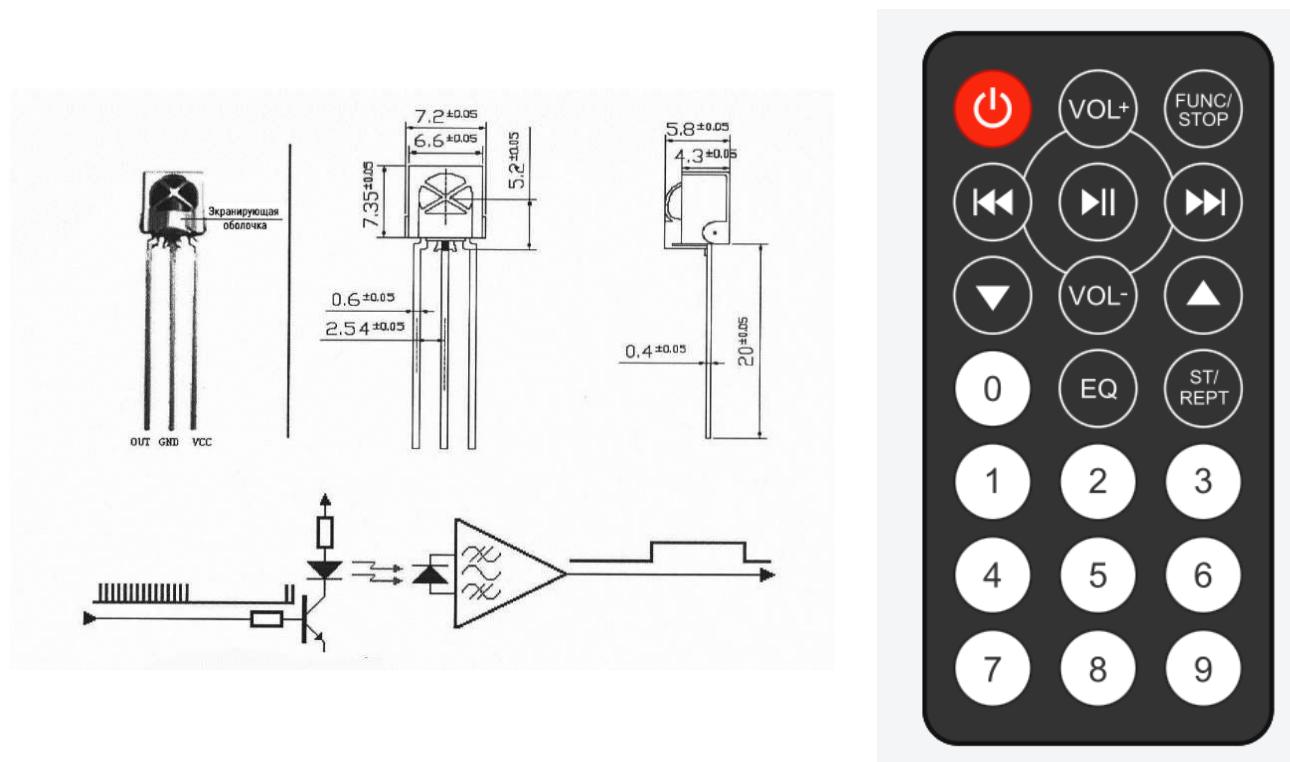
- плата Arduino
- макетная плата
- провода для соединений
- ик приемник
- пульт

### Принцип работы инфракрасного приемника:

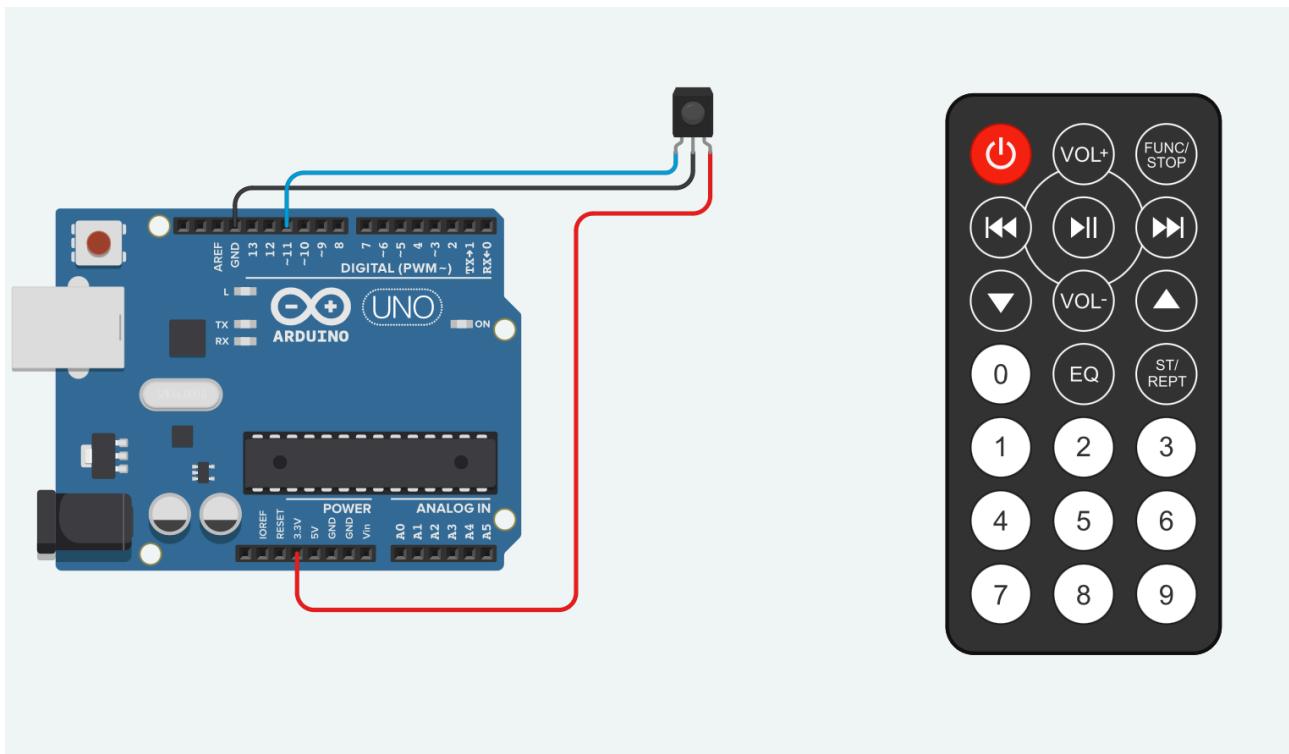
Состоит из систем излучения и приема ИК-сигнала.

**ИК-приемник:** цепь приемника образуется из приемника ИК-сигнала и усилителя, интегрированных в общий модуль. Способен выполнять все процессы от приема ИК-сигнала до передачи сигнала на уровень TTL. Подходит для дистанционного приема данных. Модуль ИК-приемника имеет три контакта: линию ИК-сигнала, VCC- и GND-линии. Приемник удобно подключать к Arduino и другим микроконтроллерам.

**ИК-излучатель(пульт):** дистанционный излучатель с сигналом несущей частоты 38 кГц, производит кодировку сигнала посредством кодирующей микросхемы. Принцип кодирования сигнала представлен в инструкции по эксплуатации ИК-приемника.



Подключаем пульт и ик приемник по следующей схеме: (ик примник лучше подключать к 3.3v, чтобы он "не сгорел" так быстро)



Направляем пульт в ик приемник и на последовательно мониторе мы видим комбинации чисел/букв, которые ик приемник получает, когда мы нажимаем на кнопки на пульт.

Код приведён ниже, а также в папке **КОД: 16. Опыт с управлением Инфракрасный приемник сигналов.**

```
#include "IRremote.h"
```

```
IRrecv irrecv(11); // указываем вывод, к которому подключен приемник
```

decode results results;

```
void setup() {
```

```
Serial.begin(9600); // выставляем скорость COM порта  
irrecv.enableIRIn(); // запускаем прием
```

}

```
void loop() {
```

```
if ( irrecv.decode( &results ) ) { // если данные пришли
```

```
Serial.println( results.value, HEX ); // печатаем данные
```

```
irrecv.resume(); // принимаем следующую команду
```

}

}