Курс «Основы системной инженерии»

Arduino Uno

Федулеев А.А.

Федулеева Е.А.

Москва

2017

Оглавление

[Знакомство с микроконтроллером Arduino Uno 3](#_Toc489019477)

[Подключение микроконтроллера Arduino Uno к компьютеру 3](#_Toc489019478)

[Знакомство со средой программирования Arduino IDE 5](#_Toc489019479)

[Создание программы мигания встроенным светодиодом 6](#_Toc489019480)

[Подключение кнопки к Arduino Uno 7](#_Toc489019481)

[Создание программы управления работой светодиода при помощи кнопки 9](#_Toc489019482)

[Подключение внешнего светодиода к Arduino Uno 9](#_Toc489019483)

[Управление работой внешнего светодиода при помощи кнопки 10](#_Toc489019484)

[Различия между цифровыми и аналоговыми датчиками 12](#_Toc489019485)

[Аналоговые датчики 12](#_Toc489019486)

[Цифровые датчики 13](#_Toc489019487)

[Изучение принципа работы фоторезистора 14](#_Toc489019488)

[Подключение фоторезистора к Arduino Uno 15](#_Toc489019489)

[Управление работой светодиода в зависимости от уровня освещения 15](#_Toc489019490)

[Изучение принципа работы ультразвукового дальномера 17](#_Toc489019491)

[Подключение ультразвукового дальномера к Arduino UNO 18](#_Toc489019492)

[Управление работой светодиода в зависимости от расстояния до ближайшего предмета 18](#_Toc489019493)

[Способы электропитания Arduino UNO 20](#_Toc489019494)

[Датчик температуры и влажности воздуха DHT11 22](#_Toc489019495)

[Знакомство с «Монитором порта» 23](#_Toc489019496)

[Подключение DHT 11 к Arduino UNO 25](#_Toc489019497)

[Знакомство с текстовым экраном LCD 16x2 28](#_Toc489019498)

[Подключение библиотеки LiquidCrystal для работы с текстовым экраном 30](#_Toc489019499)

[Подключение текстового экрана к Arduino UNO 31](#_Toc489019500)

[Включение подсветки 31](#_Toc489019501)

[Включение питания знакосинтезатора 32](#_Toc489019502)

[Подключение шины данных 33](#_Toc489019503)

[Вывод текста на текстовый экран 34](#_Toc489019504)

[Вывод специальных символов на текстовый экран 34](#_Toc489019505)

[Изучение принципа действия коммутационного 4-х канального реле 36](#_Toc489019506)

[Изучение принципа работы светодиодной ленты 37](#_Toc489019507)

[Подключения реле и светодиодной ленты к Arduino UNO 38](#_Toc489019508)

[Управление работой светодиодной ленты при помощи реле 40](#_Toc489019509)

[Основные понятия технологий построения сетей 41](#_Toc489019510)

[Знакомство с Ethernet shield 42](#_Toc489019511)

[Подключение Ethernet shield к Arduino UNO 43](#_Toc489019512)

[Настройка сетевого интерфейса 44](#_Toc489019513)

[Получение и отправка данных в Интернет 45](#_Toc489019514)

## Знакомство с микроконтроллером Arduino Uno

Краткие сведения: Arduino (ардуино) — популярная аппаратная вычислительная платформа, основными компонентами которой являются плата ввода-вывода и среда разработки. Arduino может использоваться как для создания автономных интерактивных объектов, роботов, так и подключаться к программному обеспечению, выполняемому на компьютере. Платы имеют аналоговые и цифровые порты, к которым можно подключить большое количество датчиков и исполнительных устройств: кнопка, помпа, мотор, светодиод, экран и т.д.

Изучение Arduino часто входит в учебные программы по робототехнике и изучается в кружках робототехники. Но плата популярна и в создании «настоящих» роботов. Именно Arduino может стать «мозгом» вашего робота.  
Для начала, мы проверим работоспособность Arduino на примере мигания светодиода.

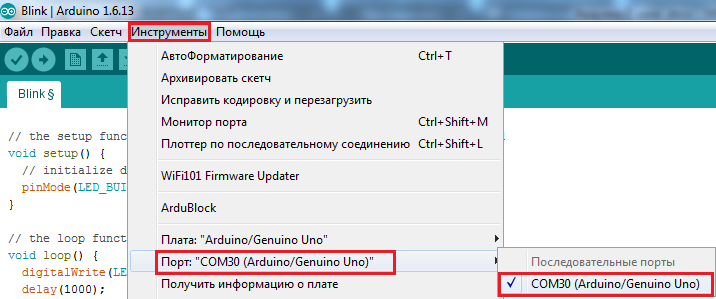
## Подключение микроконтроллера Arduino Uno к компьютеру

Для того чтобы подключить Arduino Uno к компьютеру необходимо использовать кабель USB 2.0 Type-B.

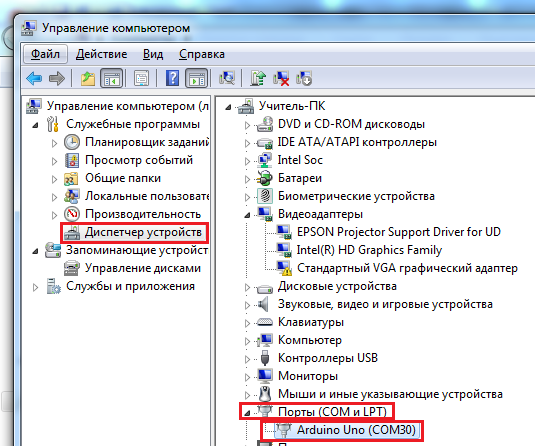


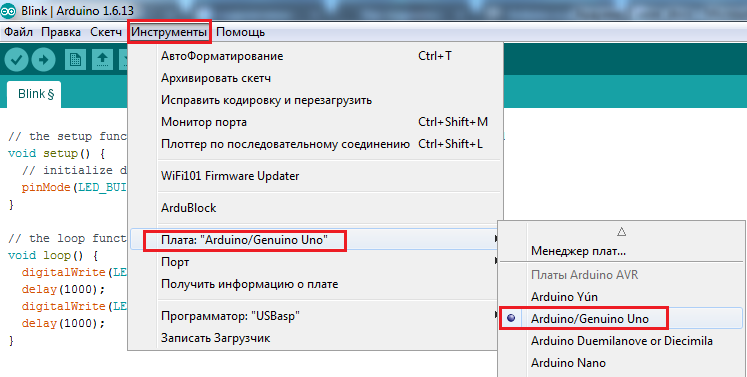
После того, как вы подключили Arduino к компьютеру и открыли Arduino IDE, вам необходимо установить в программе нужный COM порт, и тип используемой платы.

Для этого в главном меню вы нажимаете на вкладку Инструменты, выбираете порт и устанавливаете тот COM, к которому подключен ваш микроконтроллер.



Если пункт меню «Порт» не активен, проверьте, правильно ли подключен кабель. Если вы не знаете, к какому COM порту подключена ваша плата, посмотрите в диспетчере устройств.



Далее необходимо выбрать нужную плату: Инструменты/Плата/Arduino/Genuino Uno

## Знакомство со средой программирования Arduino IDE

****

Главное меню обеспечивает доступ ко всем функциям среды Arduino IDE.

Под строкой главного меню находятся кнопки наиболее часто используемых команд.

Редактор кода автоматически выделяет цветом различные части программы (переменные, процедуры, функции и т.д.).

Окно вывода сообщений: вывод информационных сообщений во время компиляции, загрузки программы и т.д.

## Создание программы мигания встроенным светодиодом

Контроллер Arduino UNO уже содержит резистор и LED-светодиод, подключенный к 13 выводу, поэтому никаких других внешних радиоэлементов нам не понадобится.

Для того чтобы ваш светодиод начал мигать, вам необходимо открыть соответствующую программу из примеров, которые уже есть в самом языке программирования Arduino.

Для этого в верхнем меню вы выбираете Файл/Примеры/Basics/Blink, либо же вы можете скопировать программу ниже:

/\*

\* Lesson1\_1

\* Мигание LED

\* ------------

\*

\* Включает и выключает светодиод (LED) подсоединенный

\* к выходу 13, с интервалом в 2 секунды

\*

\*/

int ledPin = 13; // LED подсоединен к выводу 13

void setup()

{

pinMode(ledPin, OUTPUT); // устанавливаем вывод 13 как выход

}

void loop()

{

digitalWrite(ledPin, HIGH); // включаем LED

delay(1000); // пауза 1 секунда (1000 мс)

digitalWrite(ledPin, LOW); // выключаем LED

delay(1000); // пауза 1 секунда (1000 мс)

}

|  |  |
| --- | --- |
|  |  |

После подачи напряжения на Arduino автоматически запускается процедура *setup*, которая устанавливает вывод 13 как выход. Затем автоматически в цикле запускается процедура *loop*, которая включает и выключает светодиод с паузой в 1 секунду.

После того, как программа откомпилируется и загрузится в микроконтроллер – ваш светодиод должен мигать. Если этого не происходит, убедитесь, что вы нигде не допустили ошибок.

*Для загрузки программы в микроконтроллер Arduino нажмите кнопку «Загрузка» на панели инструментов *

Если программа не загружается, попробуйте отключить Arduino и подключить заново.

**Задание:** Попробуйте изменить промежуток времени между включением и выключением светодиода.

**Решение:** Функция *delay* отвечает за паузу (промежуток между включением и выключением светодиода), попробуйте изменить значения в этой функции.

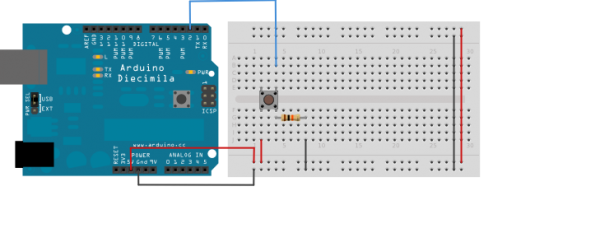
## Подключение кнопки к Arduino Uno

Теперь, когда все разобрались со светодиодом, можно немного усложнить задачу. Теперь светодиод будет включаться и выключаться не сам по себе, а при помощи «кнопки».

**Необходимые компоненты**

* контроллер Arduino
* тактовая кнопка
* 10кОм резистор
* breadboard
* соединительные провода

**Подключение**

[](http://arduino.ru/sites/default/files/tutorials/Example_button.png)

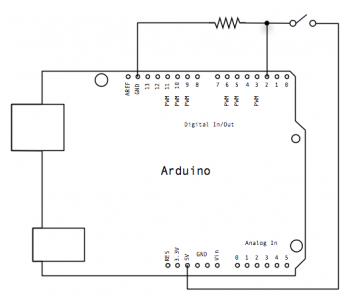
*Примечание: мигает светодиод с подписью L.*

Подключаем выход питания (5V) и землю (Gnd), красным и черным проводом к макетной плате. Обычно на макетных платах для питания и земли используют крайние ряды контактов, как показано на рисунке. Третьим синим проводом мы соединяем цифровой пин 2 контроллера Arduino к контакту тактовой кнопки. К этому же контакту, либо к контакту, постоянно соединенному с ней в 4х штырковом исполнении, подключаем подтягивающий резистор 10 кОм, который в свою очередь соединяем с землей. Другой выход кнопки соединяем с питанием 5 В.

Когда тактовая кнопка не нажата, выход 2 подключен только к земле через подтягивающий резистор и на этом входе будет считываться [LOW](http://arduino.ru/Reference/Constants). А когда кнопка нажата, появляется контакт между входом 2 и питанием 5В, и считываться будет [HIGH](http://arduino.ru/Reference/Constants).

*Замечание: Чаще всего тактовые кнопки имеют по два контакта с каждой стороны так, как это показано на рисунке подключение. При этом по форме кнопка почти квадратная. ВАЖНО не перепутать при подключении какие контакты соединены, а какие замыкаются при нажатие. Лучше всего прозвонить кнопку, если вы не уверены.*

Можно также подключить кнопку наоборот — через подтягивающий резистор к питанию и через кнопку к земле. Тогда с входа будет считываться HIGH, а при нажатие кнопки, LOW.

**Схема**[](http://arduino.ru/sites/default/files/tutorials/Example_button_scheme.png)

Если вход оставить неподключенным, то на входе будет считываться HIGH или LOW случайным образом. Именно поэтому мы используем подтягивающий резистор, чтобы задать определенное значение при ненажатой кнопке.

## Создание программы управления работой светодиода при помощи кнопки

**Программный код lesson1\_2**

/\*

\* lesson 1\_2

Управление работой светодиода с помощью кнопки

\*/

// задаем константы

const int buttonPin = 2; // номер входа, подключенный к кнопке

const int ledPin = 13; // номер выхода светодиода

// переменные

int buttonState = 0; // переменная для хранения состояния кнопки

void setup() {

// инициализируем пин, подключенный к светодиоду, как выход

pinMode(ledPin, OUTPUT);

// инициализируем пин, подключенный к кнопке, как вход

pinMode(buttonPin, INPUT);

}

void loop(){

// считываем значения с входа кнопки

buttonState = digitalRead(buttonPin);

// проверяем нажата ли кнопка

// если нажата, то buttonState будет HIGH:

if (buttonState == HIGH) {

// включаем светодиод

digitalWrite(ledPin, HIGH);

}

else {

// выключаем светодиод

digitalWrite(ledPin, LOW);

}

}

После загрузки программы, когда вы нажимаете кнопку, то светодиод включен, а когда кнопка в ненажатом состоянии, то светодиод выключен.

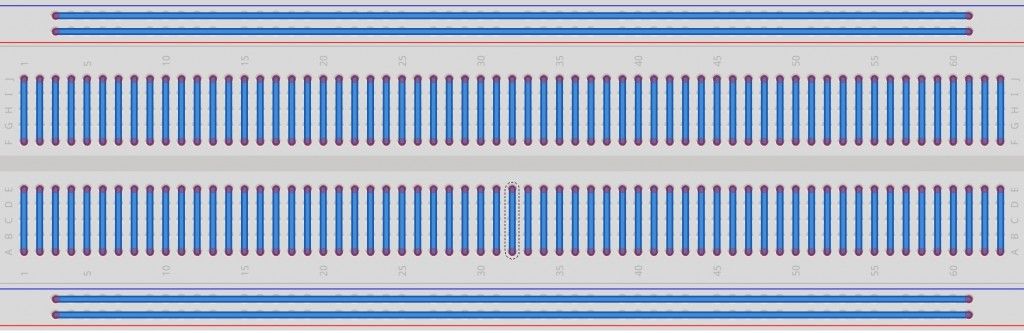
## Подключение внешнего светодиода к Arduino Uno

Если у вас есть необходимые компоненты, то вы можете подключить резистор через макетную плату (breadboard).

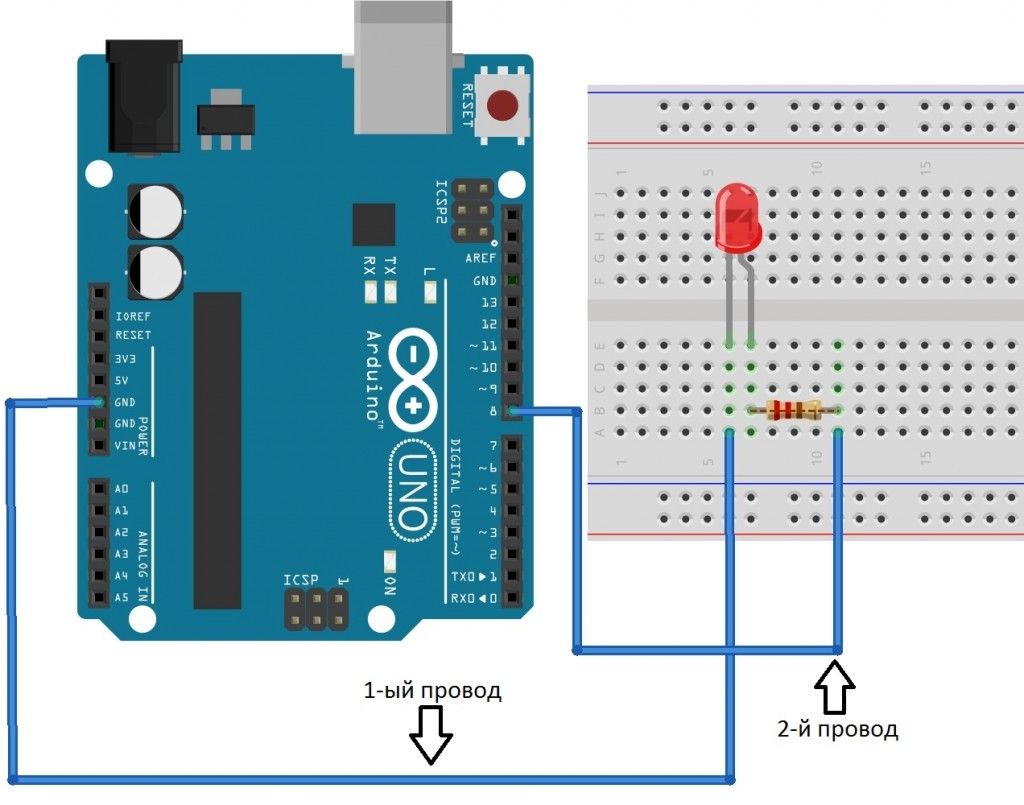
Для подключения мигающего светодиода на Arduino и управления им вам понадобится:

* плата Arduino;
* breadboard;
* 2 провода «папа-папа»;
* светодиод;
* резистор.

Breadboard представляет из себя сетку из гнезд, которые обычно соединяются так:



для удобства приводим схему подключения светодиода на Arduino:



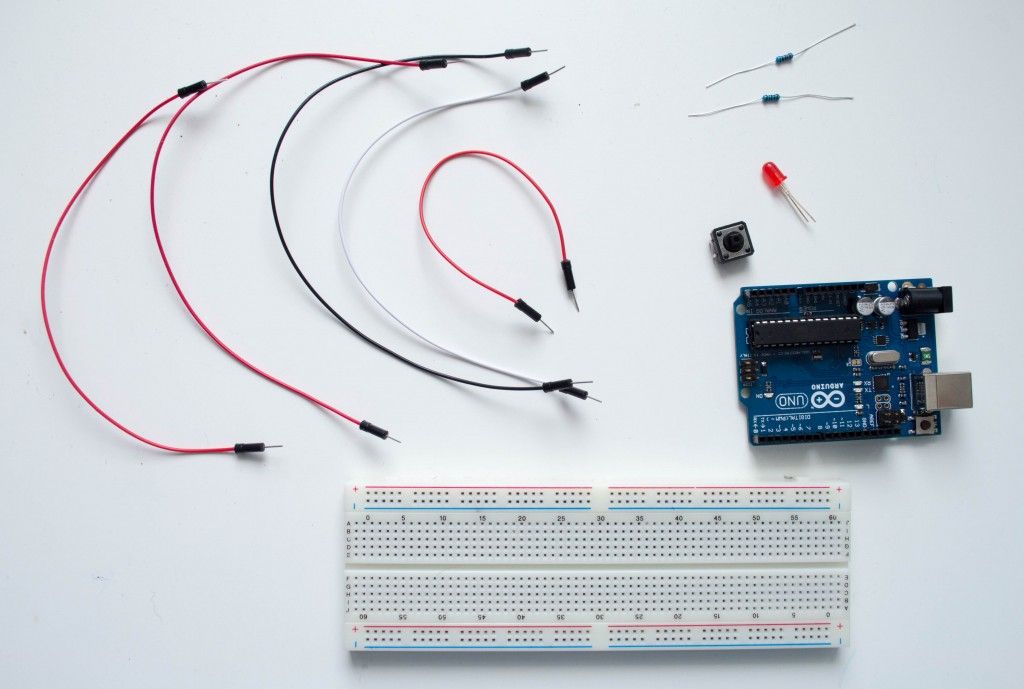
*Примечание: Длинная ножка светодиода – плюс, короткая – минус.*

Для работы этой модели подойдет программа Lesson1\_1.

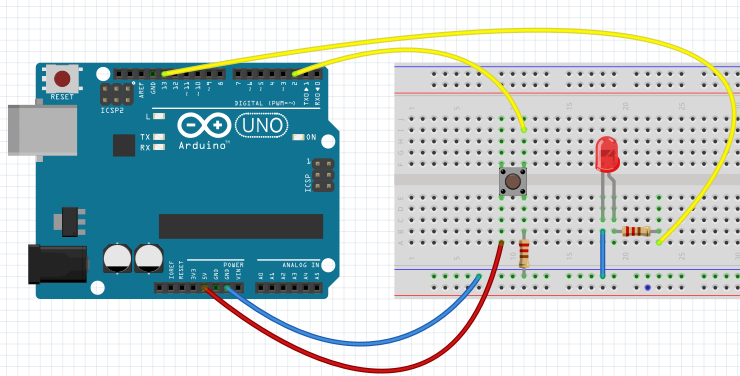
После того, как программа откомпилируется и загрузится в микроконтроллер – ваш светодиод должен мигать. Если этого не происходит, убедитесь, что вы правильно собрали схему и не допустили ошибок в коде.

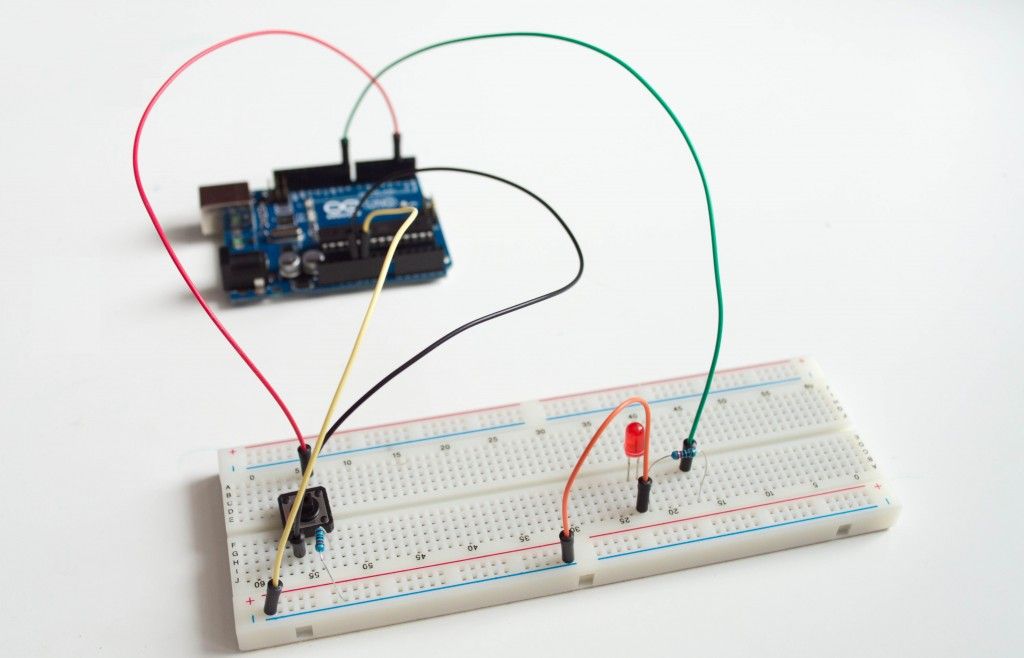
## Управление работой внешнего светодиода при помощи кнопки

Можно подключить внешний светодиод при помощи кнопки так же, как мы это делали со светодиодом, встроенным в микроконтроллер.

[](http://edurobots.ru/wp-content/uploads/2014/03/%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D1%8B1.jpg)Для сборки модели нам потребуется:

* плата Arduino;
* Breadboard;
* 5 проводов и/или перемычек «папа-папа»;
* светодиод;
* кнопка;
* резисторы на 10 кОм и 220 Ом.

Схема подключения модели Arduino с кнопкой и светодиодом:

Так выглядит собранная модель Arduino кнопки со светодиодом:

Для работы этой модели подойдет программа Lesson1\_2.

## Различия между цифровыми и аналоговыми датчиками

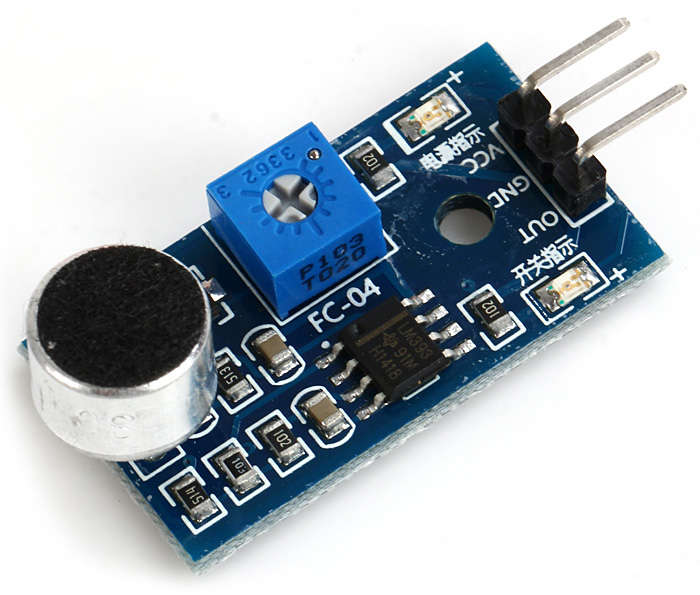
Arduino может принять сигналы от различных датчиков, сенсоров, кнопок. Например от датчиков температуры, освещенности, загазованности, скорости вращения двигателя, клавиатуры и т. д. По типу сигнала датчики разделяются на аналоговые и цифровые.

### Аналоговые датчики

При использовании аналогового сигнала, показания датчика передаются в виде переменного напряжения на сигнальном проводе. Сигнальное напряжение может принимать значение от 0 В до напряжения питания. Хотя обычно «рабочий диапазон» напряжений более узкий.

На Arduino Uno имеется 6 аналоговых входов, с помощью которых можно считывать переменное напряжение и, исходя из его значения, получать значения с датчика. Эти входы объединены на плате в группу «Analog In» и пронумерованы от A0 до A5.

Примеры аналоговых датчиков:

**Фоторезистор: Датчик звука:**

### Цифровые датчики

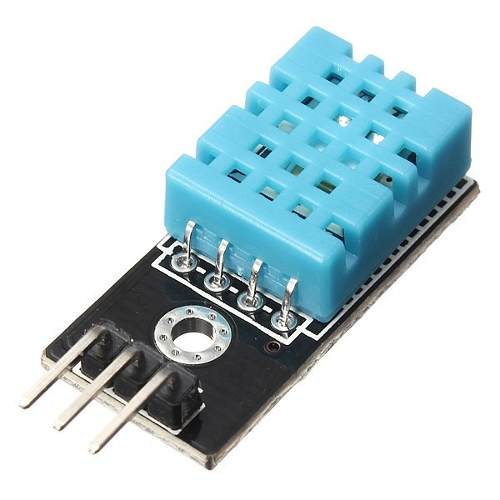
При использовании цифрового сигнала, сенсор в любой момент времени выдаёт на сигнальный провод либо 0 В, либо напряжение своего питания — 5 В. Промежуточных значений нет. Для того, чтобы абстрагироваться от конкретных значений напряжения, которые не важны при обработке цифровых сигналов, существуют понятия логического нуля (LOW) и логической единицы (HIGH). 0 В — это логический ноль, напряжение питания — это логическая единица.

На Arduino UNO имеется 14 цифровых входов, любой из которых может быть использован для подключения такого датчика.

Примеры цифровых датчиков:

**Датчик температуры и Ультразвуковой дальномер:**

**влажности воздуха:**

**Преимущества и недостатки каждого типа датчиков:**

|  |  |  |
| --- | --- | --- |
|  | Преимущества | Недостатки |
| Аналоговые датчики | * Простота использования * Не требуется расшифровка протокола | * Неустойчивость к внешним шумам * Ограничение по длине провода |
| Цифровые датчики | * Простота использования если датчик возвращает 0 или 1 * Нет ограничения по длине провода | * Требуется расшифровка протокола для датчиков с широкой градацией |

## Изучение принципа работы фоторезистора

Фоторезистор — электронный компонент, меняющий сопротивление в зависимости от количества света падающего на него. В полной темноте он имеет максимальное сопротивление в сотни кОм, а по мере роста освещённости сопротивление уменьшается до десятков кОм.

Фоторезисторы не воспринимают весь диапазон световых волн. В большинстве исполнений они чувствительны к световым волнам в диапазоне между 700 нм (красный) и 500 нм (зеленый).

Керамическая подложка

Металлические электроды

Полупроводник (сернистый таллий, сернистый свинец, сернистый кадмий)



Значения сопротивления фоторезистора в зависимости от уровня освещенности:

|  |  |  |
| --- | --- | --- |
| Освещение окружения | Освещение окружения (Лк\*) | Сопротивление фоторезистора (Ом\*\*) |
| Безлунная ночь | 0.1 Лк | 600 кОм |
| Лунная ночь | 1 Лк | 70 кОм |
| Темная комната | 10 Лк | 10 кОм |
| Облачный день/ярко освещенная комната | 100 Лк | 1,5 кОм |
| Солнечный день | 1000 Лк | 300 Ом |

\*Лк (Люкс) - единица измерения освещённости в Международной системе единиц (СИ). Люкс равен освещённости поверхности площадью 1 м² при световом потоке падающего на неё излучения, равном 1 лм.

\*\*Ом - единица измерения электрического сопротивления в Международной системе единиц (СИ). Ом равен электрическому сопротивлению проводника, между концами которого возникает напряжение 1 вольт при силе постоянного тока 1 ампер.

## Подключение фоторезистора к Arduino Uno

Фоторезисторы используются в робототехнике как датчики освещенности. Встроенный в робота фоторезистор позволяет определять степень освещенности, белые или черные участки на поверхности и в соответствие с этим двигаться по линии или совершать другие действия.

В нашей модели светодиод горит, только если яркость света над фоторезистором меньше определенной. Эту яркость можно регулировать программно.

Обычно датчики освещенности имеют три вывода:

**G (может также маркироваться как GND):**минус питающего напряжения

**V (может также маркироваться как VTG, +5V):**плюс питающего напряжения

**S (может также маркироваться как AO(analog output), SIG, OUT):**выходной сигнал, подключается к любому аналоговому входу

Напряжение питания у различных датчиков может быть 3.3V либо 5V (смотреть в спецификации к датчику).

**Обратите внимание, что неправильное подключение датчика к плате Arduino, может привести к выходу его из строя.**

Arduino датчики оборудованы трех пиновым разъемом стандарта 2.54мм, что позволит подключить их плате с помощью соединительных проводов.

## Управление работой светодиода в зависимости от уровня освещения

Необходимые компоненты:

* плата Arduino;
* 6 проводов “папа-папа”;
* фоторезистор;
* светодиод;
* резистор на 220 Ом и на 10 кОм.

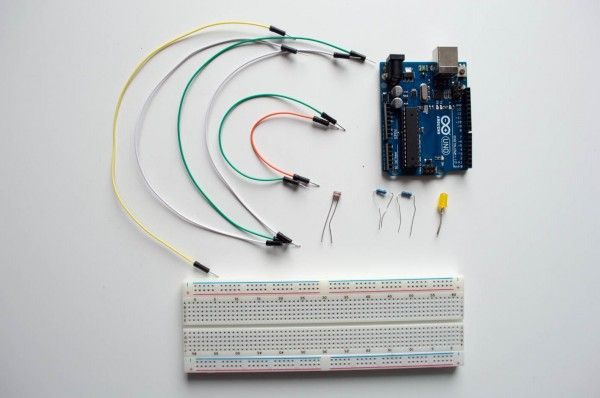
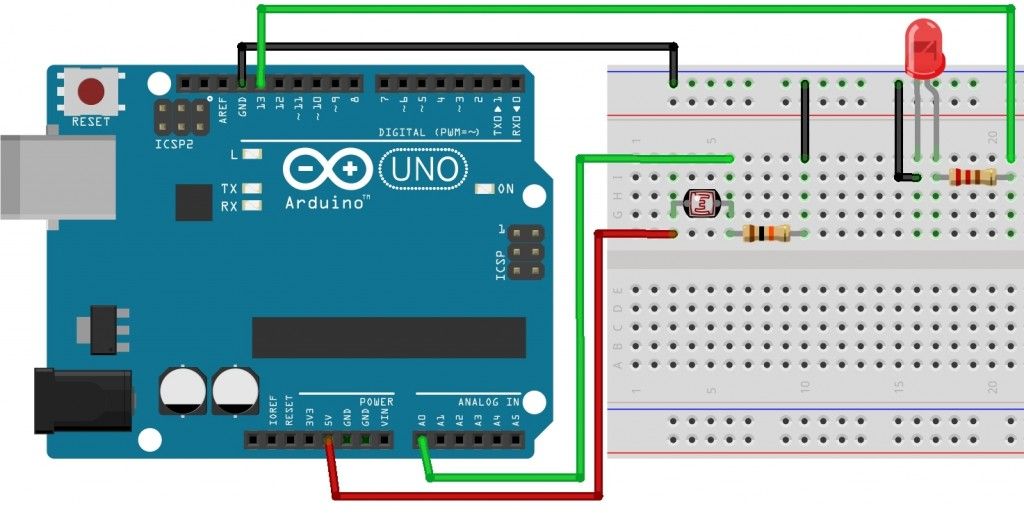
[](http://edurobots.ru/wp-content/uploads/2014/05/%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D1%8B.jpg)

Схема подключения модели Arduino с фоторезистором:

[](http://edurobots.ru/wp-content/uploads/2014/05/7%D1%81%D1%85%D0%B5%D0%BC%D0%B0.jpg)

Для работы этой модели подойдет программа Lesson2\_1:

/\*

\* Lesson2\_1

\* Включает и выключает светодиод (LED) подсоединенный

\* к выходу 13, в зависимости от уровня освещенности над фоторезистором

\*

\*/

int led = 13; //LED подсоединен к выводу 13

int ldr = A0; //фоторезистор подключен к входу A0

void setup() //процедура setup

{

pinMode(led, OUTPUT); //указываем, что светодиод - выход

}

void loop() //процедура loop

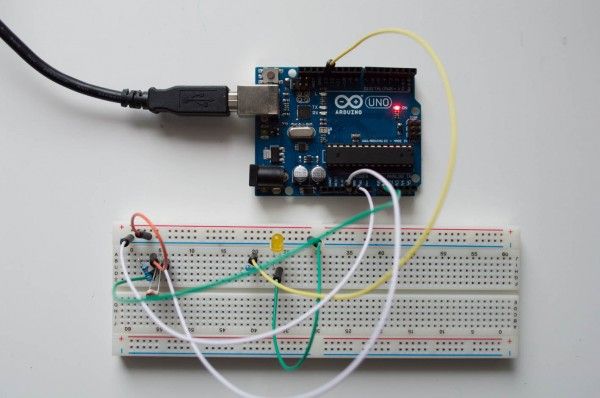
{

if (analogRead(ldr) > 500) digitalWrite(led, HIGH); //если показатель освещенности больше 500, включаем светодиод

else digitalWrite(led, LOW); //иначе выключаем

}

Так выглядит собранная модель Arduino с фоторезистором:

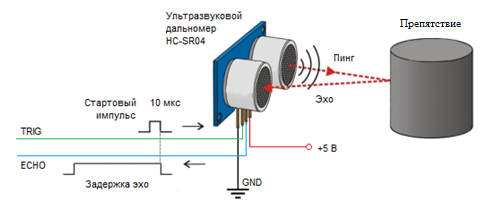
[](http://edurobots.ru/wp-content/uploads/2014/05/%D0%B3%D0%BE%D1%82%D0%BE%D0%B2%D0%BE.jpg)

Если светодиод не реагирует на изменение освещенности, то попробуйте поменять число *500* в программе, если он все время горит  — уменьшите, если не горит — увеличьте.

**Задание**: Обсудите, для каких целей можно применять фоторезистор при создании систем Интернета вещей.

## Изучение принципа работы ультразвукового дальномера

**Ультразвуковой дальномер** HC-SR04 предназначен для измерения расстояния от устройства до объекта. Сенсор дальномера работает по принципу сонара, а точнее — посылает ультразвуковой пучок и по задержке отражённого от объекта сигнала определяет расстояние до цели.

Ультразвуковой дальномер определяет расстояние до объектов точно так же, как это делают дельфины или летучие мыши. Он генерирует звуковые импульсы на частоте 40 кГц и слушает эхо. По времени распространения звуковой волны туда и обратно можно однозначно определить расстояние до объекта.

**Для определения расстояния ультразвуковой дальномер выполняет следующие действия:**

**Шаг 1:**На вход **Trig** подаётся импульс длительностью 10 микросекунд. Для дальномера это команда начать измерение расстояния перед ним.

**Шаг 2:**Устройство генерирует 8 ультразвуковых импульсов**с частотой 40 кГц**через выходной сенсор **T**.

**Шаг 3:**Звуковая волна отражается от препятствия и попадает на принимающий сенсор **R**.

**Шаг 4:**На выходе **Echo** формируется импульс, длительность которого прямо пропорциональна измеренному расстоянию.

**Шаг 5:**На стороне управляющего контроллера переводим длительность импульса **Echo** в расстояние по формуле: **ширина импульса (мкс) / 58 = дистанция (см)**.

## Подключение ультразвукового дальномера к Arduino UNO

Arduino датчики оборудованы трех пиновым разъемом стандарта 2.54мм, что позволит подключить их плате с помощью соединительных проводов.

**G: подключается к выводу GND**

**V: подключается к выводу +5V или 3.3V в зависимости от датчика**

**Trig: подключается к любому цифровому входу**

**Echo: подключается к любому цифровому входу**

## Управление работой светодиода в зависимости от расстояния до ближайшего предмета

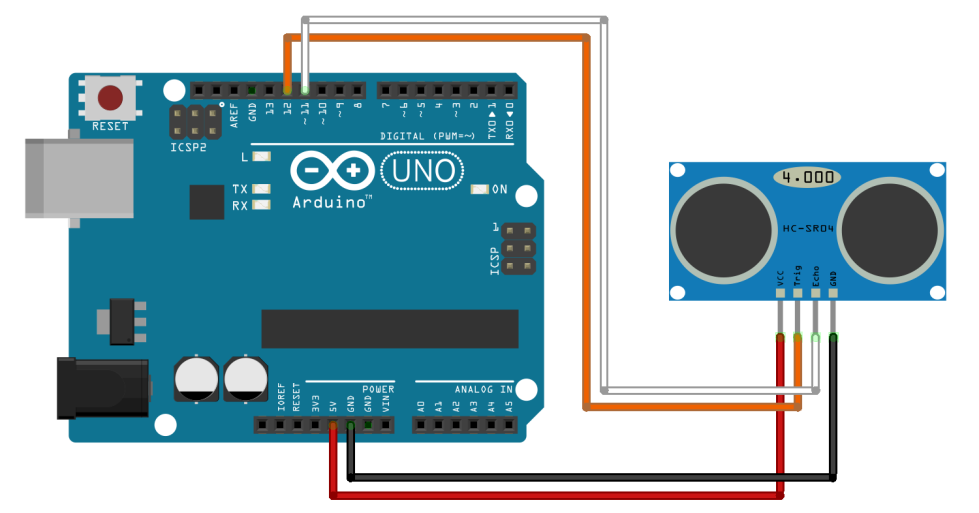
В этом уроке мы проверим работу ультразвукового дальномера при помощи светодиода. Если расстояние до ближайшего объекта будет меньше 50 см, то светодиод будет гореть, а если больше 50 см, то светодиод погаснет.

**Вам понадобится**

* Arduino;
* ультразвуковой датчик HC-SR04;
* соединительные провода;
* светодиод;
* резистор на 10 кОм.

**Схема подключения дальномера:**

Питание ультразвукового дальномера осуществляется напряжением +5 В. Два других вывода подключаются к любым цифровым портам Arduino (мы подключим к 12 и 11).

[](https://soltau.ru/images/HC-SR04/2.png)

Чтобы подключить светодиод см. подключение фоторезистора к Arduino

Текст программы Lesson2\_2:

/\*

\* Lesson2\_2

\* Включает и выключает светодиод (LED) подсоединенный

\* к выходу 13, в зависимости от расстояния до ближайшего к сенсору расстояния объекта

\*

\*/

int Trig = 12; //Выход TRIG сенсора расстояния подсоединен к выводу 12

int Echo = 11; //Выход ECHO сенсора расстояния подсоединен к выводу 11

int ledPin = 13; //LED подсоединен к выводу 13

unsigned int time\_us=0; //переменная для хранения времени отклика

unsigned int distance\_sm=0; //переменная для хранения расстояния до объекта

void setup()

{

pinMode(Trig, OUTPUT); //указываем, что вывод Trig - выход

pinMode(Echo, INPUT); //указываем, что Echo - вход

pinMode(ledPin, OUTPUT); //указываем, что светодиод - выход

}

void loop()

{

digitalWrite(Trig, HIGH); // Подаем сигнал на выход TRIG

delayMicroseconds(10); // Пауза 10 микросекунд

digitalWrite(Trig, LOW); // Убираем сигнал на выходе TRIG

time\_us=pulseIn(Echo, HIGH); // Замеряем длину импульса

distance\_sm=time\_us/58; // Пересчитываем в сантиметры

if (distance\_sm < 50) // Если расстояние менее 50 сантиметром

{

digitalWrite(ledPin, HIGH); // включаем светодиод

}

else

{

digitalWrite(ledPin, LOW); // иначе выключаем светодиод

}

delay(100); //пауза перед следующим измерением

}

## Способы электропитания Arduino UNO

**Способ 1 - питание Arduino через разъём DC (5,5мм/2,1мм)**

**Напряжение**

У Arduino UNO есть линейный регулятор напряжения и разъём подключения блока питания 5,5мм/2,1мм (внешний / внутренний диаметр). К нему можно подключать источник питания постоянного тока с напряжением от 7 до 12 В.

Золотая середина - 9...12 Вольт - позволит регулятору напряжения рассеивать выделяемое на радиаторе тепло без перегрузок и подавать на микроконтроллер требуемое напряжение.

Если у вас есть блок питания с напряжением меньше 7 В, то плата, возможно, будет работать, но на портах ввода-вывода платы будет уже не 5 В, а меньше и ваша схема будет работать непредсказуемо.

С другой стороны, напряжение выше рекомендуемого заставит регулятор напряжения на плате работать в режиме перегрузки, перегреваться и выделять слишком много тепла.

**Ток**

Потребляемый платой ток зависит от того, какие части микроконтроллера работают в данный момент и какая нагрузка подключена к плате. Чем больший ток может обеспечить ваш блок питания, тем лучше.

Минимальное требование - 250 мА. Это позволит запустить только плату Arduino без подключённых к ней датчиков или исполнительных устройств. *Лучше всего использовать источник питания на ток от 0,5 А до 2 А.*

Вот некоторые ограничения по току для разных устройств:

* порты USB имеют автоматический предохранитель на 500 мА, который должен отключить порт при его перегрузке.  Любое устройство, потребляющее ток большее 500 мА, подключённое к порту, будет работать непредсказуемо или не работать совсем;
* максимальный ток для порта вводы-выводы микроконтроллера ATmega составляем 40 мА (рекомендуется не более 35 мА) при общем токе потребления микроконтроллером не более 200 мА;
* максимальный ток, который может выдержать порт **5V** платы Arduino, составляет 0,8 А, однако рекомендуется допускать не более 500 мА;
* порт **Vin** платы Arduino мог-бы выдержать такой же ток, как и источник питания, но его ток ограничен защитным диодом этой цепи, а он рассчитан та ток 1А;
* некоторая периферия (такие как двигатели, реле, нагревательные элементы и т.п.) могут потреблять очень большой ток, так что вам следует внимательно читать описание компоненты и обеспечивать требуемый ток. Фактически, устройство, потребляющее ток более 0,5 А, должно быть запитано от своего источника.

**Способ 2 - питание Arduino через USB (5 В, 500 мА)**

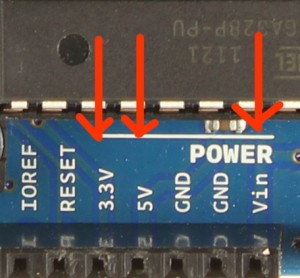
Плата Arduino может получать электропитание от порта USB, который должен быть подключен к стабилизированному источнику напряжения 5В или компьютеру.

При подключении к питанию через порт USB входящее напряжение подаётся прямо на линию питания микроконтроллера 5В и при превышении его можно повредить.

На плате установлен автоматический предохранитель, который отключит порт USB от платы при потреблении более 500 мА до тех пор, пока нагрузка не будет отключена. При этом работа вашей платы будет выглядеть как постоянная перезагрузка.

Если к плате Arduino одновременно подключены и порт USB и отдельный источник питания, то плата переключится на внешний источник питания, если напряжение на нём больше 6-7 В.

**Способ 3 - через разъёмы ввода-вывода**

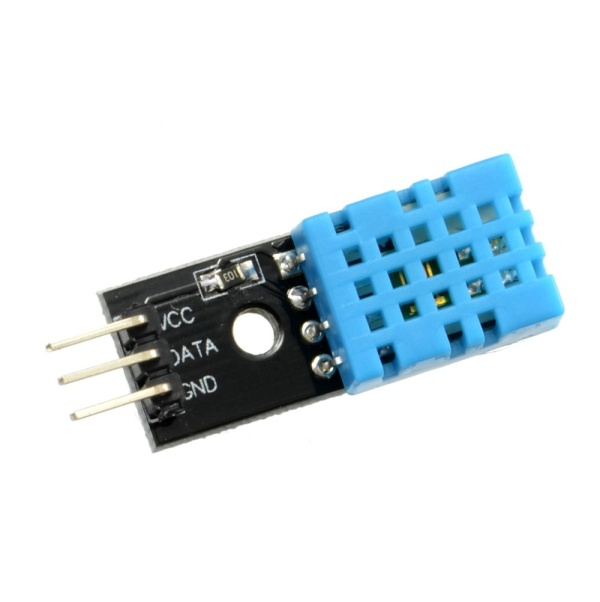


**Vin** - порт может быть использовать как вход напряжения для источника питания, например от батареек, аккумулятора или блока питания. Напряжение должно быть в пределах 9 - 12 В и будет снижено регулятором на плате до 5 В;

**5V** - порт выдаёт стабилизированное регулятором напряжение 5 В. Вы можете подавать через этот порт напряжение питания прямо на микроконтроллер, однако оно должно быть стабилизировано!

**3V3** - выход 3,3 В от отдельного стабилизатора на плате. Максимальный ток - 150 мА;

## Датчик температуры и влажности воздуха DHT11

[**DHT11**](http://makerplus.ru/shop/cifrovoj-datchik-temperatury-i-vlazhnosti-dht11)  — это цифровой датчик влажности и температуры, состоящий из термистора и емкостного датчика влажности.  Также датчик содержит в себе АЦП для преобразования аналоговых значений влажности и температуры. Датчик DHT11 не обладают высоким быстродействием и точностью, но зато прост, недорог и отлично подходят для обучения и контроля влажности в помещении. Температура и влажность отдаются датчиком по одному сигнальному проводу (DATA). DHT11 общается с [Arduino](http://amperka.ru/collection/Platy/product/Arduino-Uno) UNO по собственному протоколу. Коммуникация двунаправлена и в общих чертах выглядит так:

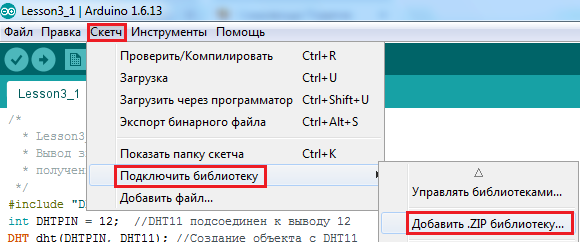
1. Микроконтроллер говорит о том, что хочет считать показания. Для этого он устанавливает сигнальную линию в 0 на некоторое время, а затем устанавливает её в 1.
2. Сенсор подтверждает готовность отдать данные. Для этого он аналогично сначала устанавливает сигнальную линию в 0, затем в 1.
3. После этого сенсор передаёт последовательность 0 и 1, последовательно формирующих 5 байт (40 бит). В первых двух байтах передаётся температура, в третьем-четвёртом — влажность, в пятом — контрольная сумма, чтобы микроконтроллер смог убедиться в отсутствии ошибок считывания.

Благодаря тому, что сенсор делает измерения только по запросу, достигается энергоэффективность: пока общения нет, датчик потребляет очень небольшой ток.

Характеристики датчика DHT11:

* определение влажности в диапазоне 20-80%;
* определение температуры от 0°C до +50°C;
* частота опроса 1 раз в секунду;

Для работы с сенсорами DHT вам необходимо установить соответствующую библиотеку, которую можно установить из файла DHT\_sensor\_library.zip.



Основные команды для работы с библиотекой DHT:

//Инициализация датчика

DHT dht(DHTPIN, DHT11); //Создание объекта с DHT11, подключенного к DHTPIN

dht.begin(); //Инициализация датчика DHT

//Получение данных

dht.readHumidity(); //Получение данных о влажности

dht.readTemperature(); //Получение данных о температуре

## Знакомство с «Монитором порта»

Arduino UNO имеет встроенный контроллер для последовательной передачи данных, который может использоваться как для связи между Arduino устройствами, так и для связи с компьютером. На компьютере соответствующее соединение представлено обычным COM-портом, который появляется в системе после установки необходимого драйвера.   
  
Связь происходит по цифровым портам **0 и 1**, и поэтому Вы не сможете использовать их для цифрового ввода/вывода если используете функции последовательной передачи данных.   
  
Вызов:

Serial.begin(скорость\_передачи);

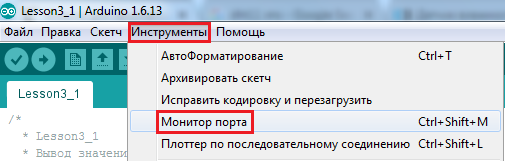
Описание:   
Устанавливает скорость передачи информации COM порта битах в секунду для последовательной передачи данных. Для того чтобы поддерживать связь с компьютером, используйте одну из этих нормированных скоростей: 300, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600, или 115200. Также Вы можете определить другие скорости при связи с другим микроконтроллером по портам 0 и 1.   
Параметры: 

скорость\_передачи: скорость потока данных в битах в секунду. 

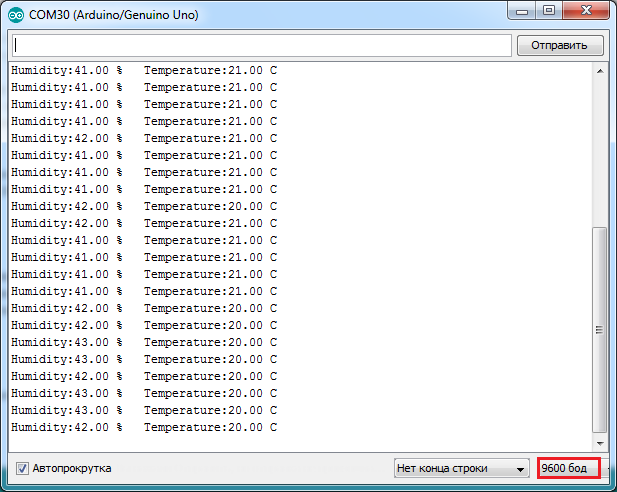
Пример:

Serial.begin(9600); //устанавливаем скорость 9600 бит/сек

Для того чтобы открыть Монитор порта необходимо в главном меню выбрать Инструменты/Монитор порта (либо использовать горячие клавиши Ctrl+Shift+M)



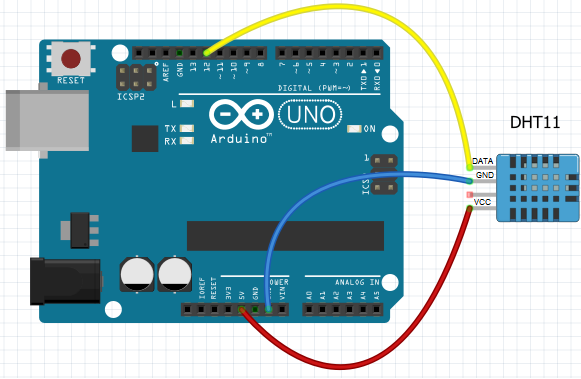
Если данные в окне Монитора порта не появляются, проверьте, какая установлена скорость передачи данных.

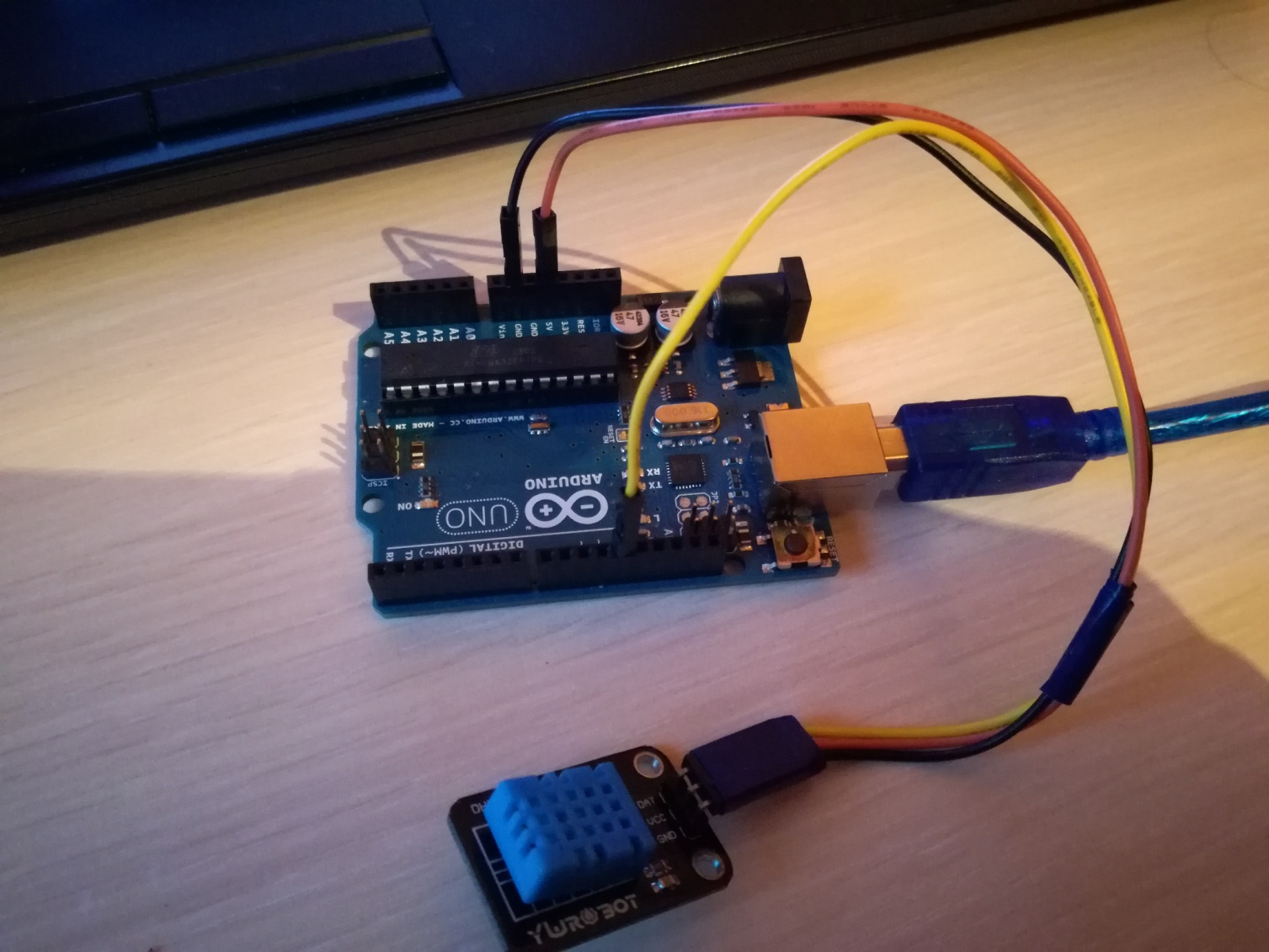


## Подключение DHT 11 к Arduino UNO

Необходимые компоненты:

* плата Arduino UNO;
* датчик DHT11;
* Breadboard;



****

**Программный код**

Воспользуемся библиотекой DHT.h, созданной специально для датчиков DHT.

/\*

\* Lesson3\_1

\* Вывод значений температуры и влажности воздуха,

\* полученных с датчика DHT11, в Монитор порта

\*/

#include "DHT.h" //Подключаем библиотеку для работы с DHT11

int DHTPIN = 12; //DHT11 подсоединен к выводу 12

DHT dht(DHTPIN, DHT11); //Создание объекта с DHT11

void setup()

{

Serial.begin(9600); //Устанавливаем скорость 9600 бит/сек

dht.begin(); //Инициализация датчика DHT11

}

void loop()

{

delay(2000); // Задержка 2 секунды между измерениями

float h = dht.readHumidity(); //Считываем влажность

float t = dht.readTemperature(); // Считываем температуру

Serial.print("Humidity:");

Serial.print (h);

Serial.print (" %");

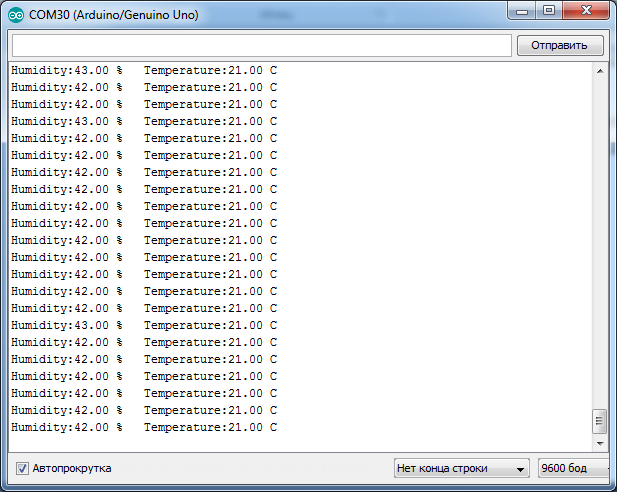
Serial.print(" Temperature:");

Serial.print(t); //Вывод данных в Монитор порта

Serial.println(" C");

}

В окне Монитора порта вы должны увидеть температуру и влажность воздуха. Изменения можно увидеть, например, выдыхая на датчик (как для затуманивания окна). Дыхание увеличивает влажность.



## Знакомство с текстовым экраном LCD 16x2

LCD дисплей 16х2 предназначен для вывода текста и специальных символов. Экран оборудован подсветкой. По изображению похож на дисплеи старых мобильных телефонов вроде Nokia 3310 или Siemens C35.

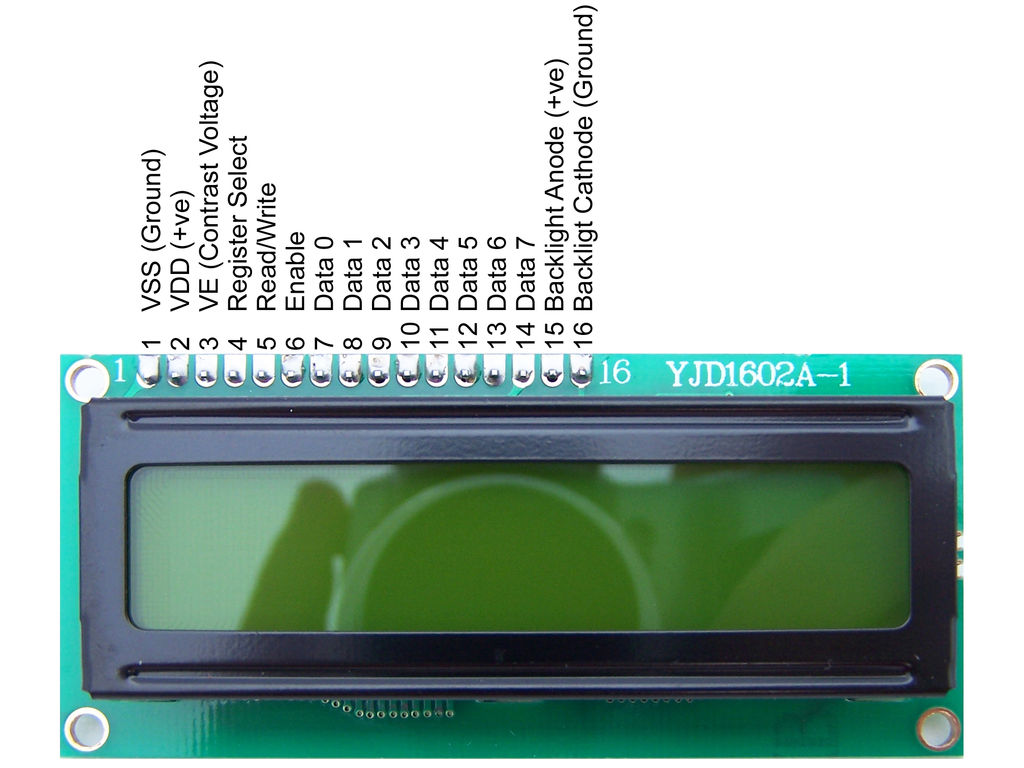
Достоинства:

* малая потребляемая мощность;
* низкое напряжение питания;
* долговечность.

Рассмотрим LCD-дисплеи на базе контроллера HD44780.  
Этот монохромный дисплей имеет опциональную подсветку и может отображать 2 строки по 16 символов. Разрешение символов — 5x8 точек.   
Чем хороши такие дисплеи? Контроллер HD44780 — стандарт де-факто среди небольших монохромных LCD-дисплеев.

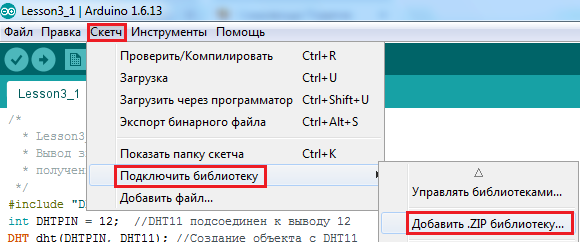
Назначения выводов LCD дисплея 16х2:

|  |  |  |  |
| --- | --- | --- | --- |
| Номер вывода | Сигнал | I - вход  O - выход | Назначение сигнала |
| 1 | Vss | - | Земля (общий провод) |
| 2 | Vdd | - | Питание + 5 В |
| 3 | Vo | - | Управление контрастностью дисплея. Вход для подключения среднего вывода делителя напряжения + 5 В. Можно использовать подстроечный резистор сопротивлением 10-20 кОм. |
| 4 | RS | I | Выбор регистра: 0 – регистр команд; 1 – регистр данных. Низкий уровень сигнала означает, что на шине данных сформирована команда, высокий уровень – на шине данные. |
| 5 | R/W | I | Направление передачи данных:  0 – запись;  1 – чтение.  Во многих приложениях функция чтения не используется, поэтому сигнал часто подключается к земле. |
| 6 | E | I | Строб операции шины (по отрицательному фронту). |
| 7 | DB0 | I/O | Младшие биты восьми битного режима. При четырех битном интерфейсе не используются. |
| 8 | DB1 | I/O |
| 9 | DB2 | I/O |
| 10 | DB3 | I/O |
| 11 | DB4 | I/O | Старшие биты восьми битного режима или биты данных  четырех битного интерфейса. |
| 12 | DB5 | I/O |
| 13 | DB6 | I/O |
| 14 | DB7 | I/O |
| 15 | A | - | Анод питания подсветки (+). |
| 16 | K | - | Катод питания подсветки (-).  Ток должен быть ограничен. |



## Подключение библиотеки LiquidCrystal для работы с текстовым экраном

Для того, чтобы подключить библиотеку LiquidCrystal вам необходимо в главном меню выбрать Скетч/Подключить библиотеку/Добавить .ZIP библиотеку/LiquidCrystal.zip



Основные команды для работы с библиотекой LiquidCrystal:

//Работа с курсором

lcd.setCursor(0, 0); // Устанавливаем курсор (номер ячейки, строка)

lcd.home(); // Установка курсора в ноль (0, 0)

lcd.cursor(); // Включить видимость курсора (подчеркивание)

lcd.noCursor(); // Убрать видимость курсора (подчеркивание)

lcd.blink(); // Включить мигание курсора (курсор 5х8)

lcd.noBlink(); // Выключить мигание курсора (курсор 5х8)

//Вывод информации

lcd.print("Arduino"); // Вывод информации

lcd.clear(); // Очистка дисплея, (удаление всех данных)   
 установка курсора в ноль

lcd.rightToLeft(); // Запись производится справа налево

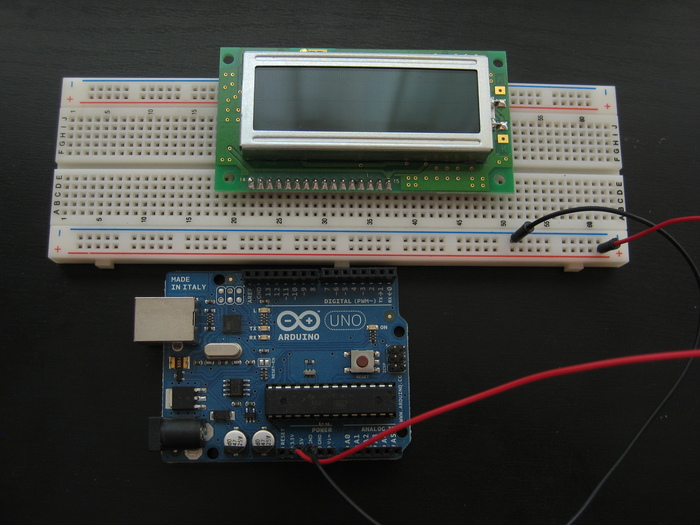
lcd.leftToRight(); // Запись производится слева направо

lcd.scrollDisplayRight(); // Смещение всего изображенного на дисплее на один   
 символ вправо

lcd.scrollDisplayLeft(); // Смещение всего изображенного на дисплее на один   
 символ влево

## Подключение текстового экрана к Arduino UNO

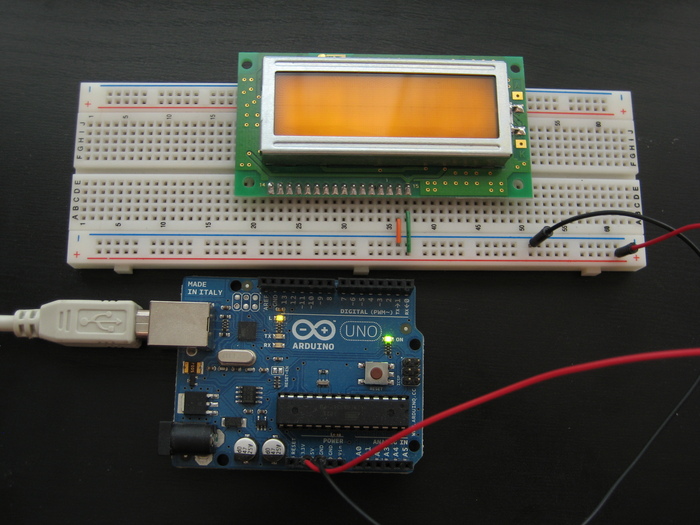
Закрепите экран на breadboard'е и подведите к рельсам питания breaboard'а питание +5 В и землю с Arduino.



Питание и земля понадобятся не один раз, поэтому удобнее пробросить их именно на рельсы.

### Включение подсветки

Фоновая подсветка дисплея — это отдельный контур, не связанный с остальным. Включить её можно подав +5 В на 15-й контакт дисплея и подключив 16-й контакт к земле. Соединив эти два контакта с соответствующими рельсами, можно включить Arduino и увидеть, что дисплей засветился.



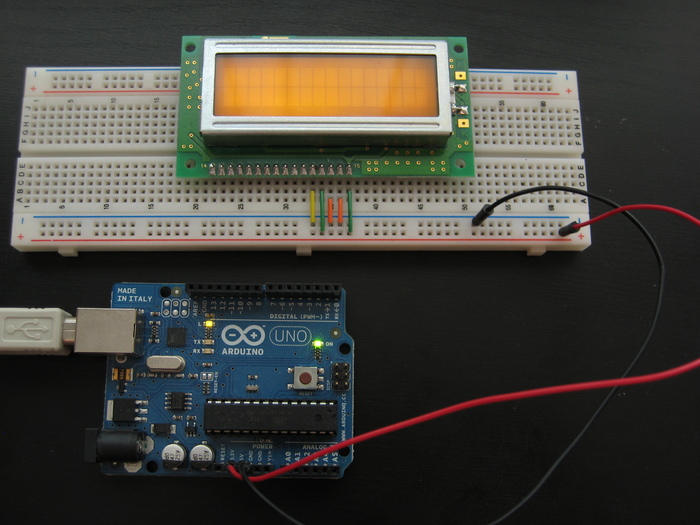
### Включение питания знакосинтезатора

Далее необходимо подключить цепь, отвечающую за отображение символов. Для этого предназначены контакты 1, 2 и 3 на дисплее. Перед подключением отключите Arduino от питания.

* Первый — это земля. Соедините его с рельсой земли.
* Второй — питание. Соедините его с рельсой +5 В.
* Третий — контрастность.

Для получения максимально контрастного изображения соедините его с рельсой земли. Вы можете подать на этот контакт произвольное напряжение от 0 до 5 В, чем оно выше, тем тусклее будет изображение, но вместе с этим снизится энергопотребление. Для возможности плавной регулировки контрастности можете подать на этот контакт выходной сигнал потенциометра.

После подключения, если включить Arduino, вы можете увидеть прямоугольные знакоместа. В зависимости от комбинации цветов текста и подсветки они могут быть как яркими и хорошо заметными, так и едва заметными. Это нормально: в любом случае, текст будет смотреться отлично.



### Подключение шины данных

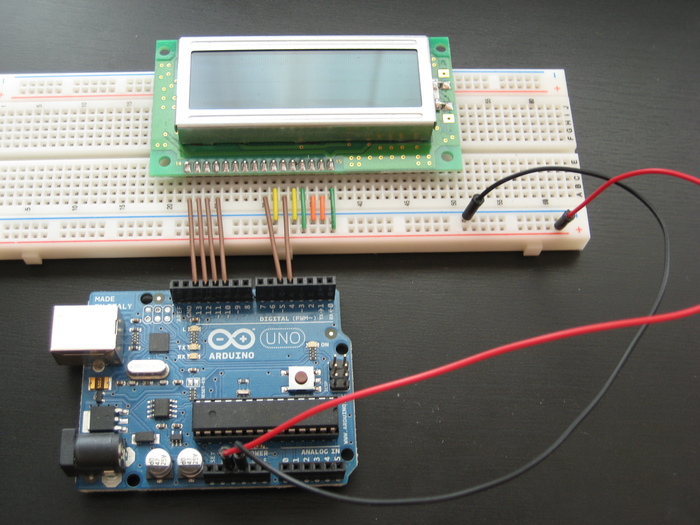
Для подключения дисплея достаточно использовать 6 линий, 6 контактов на Arduino.

Соединим 5-й контакт дисплея, который отвечает за выбор чтение/запись с рельсой земли. Это означает «всегда писать».

Затем, соединяем Arduino и экран нашими 6-ю линиями коммуникации. Какие именно контакты будут выбраны на Arduino не имеет значения: мы зададим их в программе, но для примера была выбрана такая конфигурация:

* 4-й контакт дисплея — 4-й контакт Arduino. Это линия адресного сигнала. Известная как A0 или RS. В зависимости от того, 0 она или 1, дисплей понимает, имеем ли мы на линии данных команду вроде «передвинуть курсор» или код символа для отображения;
* 6-й контакт дисплея — 5-й контакт Arduino. Это линия разрешения доступа к данным. Известная, как E или Enable. Когда эта линия становится единицей, дисплей исполняет команду или выводит символ с линии данных;
* 11-й, 12-й, 13-й, 14-й контакт дисплея — 10-й, 11-й, 12-й, 13-й контакт Arduino соответственно. Это линии данных. Известные как DB4, DB5, DB6, DB7.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LCD экран 16х2 | 1 | 2 | 3 | 4 | 6 | 11 | 12 | 13 | 14 | 15 | 16 |
| Arduino UNO | GND | +5V | GND | 4 | 5 | 10 | 11 | 12 | 13 | +5V | GND |



Экран подключен и готов к приёму данных. Осталось написать программу для Arduino.

## Вывод текста на текстовый экран

Для вывода текста с Arduino воспользуйтесь библиотекой LiquidCrystal. Для вывода приветствия, воспользуйтесь кодом ниже:

**Программный код**

/\*

\* Lesson4\_1

\* Вывод текста на текстовый экран LCD 16x2

\*/

#include <LiquidCrystal.h> // Подключаем стандартную библиотеку LiquidCrystal

LiquidCrystal lcd(4, 5, 10, 11, 12, 13); // Инициализируем объект-экран, передаём использованные

// для подключения контакты на Arduino в порядке:

// RS, E, DB4, DB5, DB6, DB7

void setup()

{

lcd.begin(16, 2);// устанавливаем размер (количество столбцов и строк) экрана

lcd.print("Hello world!");// печатаем первую строку

lcd.setCursor(0, 1); // устанавливаем курсор в колонку 0, строку 1. То есть на

// самом деле это вторая строка, т.к. нумерация начинается с нуля

lcd.print("I love Arduino");// печатаем вторую строку

}

void loop()

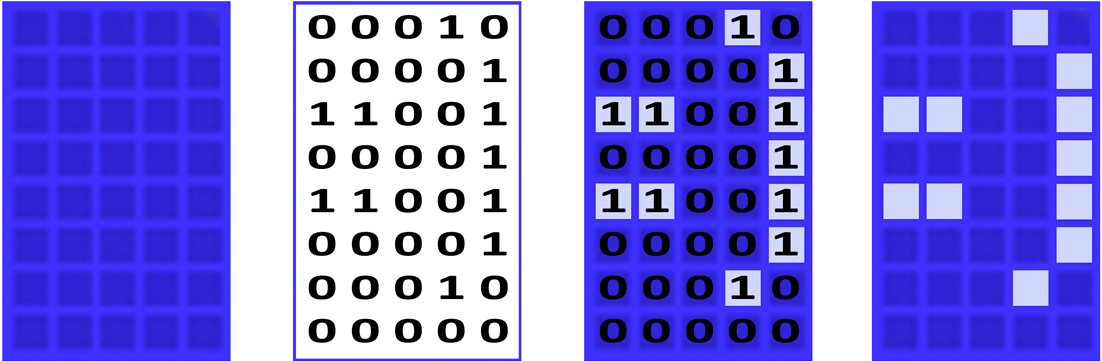
{

}

## Вывод специальных символов на текстовый экран

С выводом текста разобрались, буквы английского алфавита зашиты в память контроллера внутри дисплея и с ними проблем нет. А вот что делать, если нужного символа в памяти контроллера нет?

Не беда, требуемый символ можно сделать вручную (всего до 7ми символов). Ячейка, в рассматриваемых нами дисплеях, имеет разрешение 5х8 точек. Все, к чему сводится задача создания символа, это написать битовую маску и расставить в ней единички в местах где должны гореть точки и нолики где нет. В ниже приведенном примере нарисуем смайлик.



**Программный код**

/\* Lesson4\_2

\* Создание собственных символов на LCD экране

\*/

#include <LiquidCrystal.h> // добавляем необходимую библиотеку

byte smile[8] = // битовая маска символа улыбки

{

B00010,

B00001,

B11001,

B00001,

B11001,

B00001,

B00010,

};

LiquidCrystal lcd(4, 5, 10, 11, 12, 13); // RS, E, DB4, DB5, DB6, DB7

void setup(){

lcd.begin(16, 2); // задаем размерность экрана

lcd.createChar(1, smile); // создаем символ под номером 1

lcd.setCursor(0, 0); // устанавливаем курсор в начало 1 строки

lcd.print("\1"); // выводим смайлик (символ под номером 1) - "\1"

}

void loop(){

}

## Изучение принципа действия коммутационного 4-х канального реле

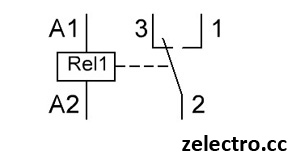
По своей сути реле — это просто механический рубильник, которым можно управлять при помощи микроконтроллера, такого как [Arduino](http://amperka.ru/collection/arduino). При этом электрическая связь между управляющей электроникой и коммутируемой нагрузкой отсутствует. А это значит, что никакие помехи не испортят ваши планы. С помощью реле вы можете включать и выключать любые исполнительные устройства (помпа, светодиодная лента и т.д.). Без реле не обойтись при обустройстве «умного дома».



Коммутационное 4-х канального реле — объединяет 4 независимых реле, расположенных на одной плате. О текущем состоянии каждого из них можно судить по индикаторным светодиодам, расположенным на плате. К каждому реле подведён клеммник на 3 провода, что позволяет использовать реле как в режиме «нормально разомкнутое», так и в режиме «нормально замкнутое». Это удобно в ряде случаев. Например, если нужно что-то включать при обесточивании микроконтроллера.

Реле управляется напряжением 5 В и способно коммутировать до 10А 30 В постоянного тока и 10A 250 В переменного тока.

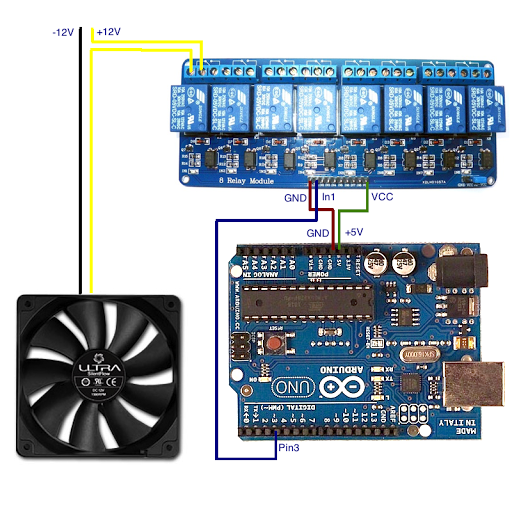
Реле имеет две раздельных цепи: цепь управления, представленная контактами А1, А2  и управляемая цепь, контакты 1, 2, 3. Цепи никак не связаны между собой.

Между контактами А1 и А2 установлен металлический сердечник, при протекании тока по которому к нему притягивается подвижный якорь(2). Контакты же 1 и 3 неподвижны. Стоит отметить что якорь подпружинен и пока мы не пропустим ток через сердечник, якорь будет удерживается прижатым к контакту 3. При подаче тока, как уже говорилось, сердечник превращается в электромагнит и притягивается к контакту 1. При обесточивании пружина снова возвращает якорь к контакту 3.

Управление реле производится через цифровые выходы Arduino с подачей сигнала HIGH или LOW для включения и выключения исполнительных устройств.

**Внимание!***Если вы сомневаетесь как подключить к реле электроприбор, работающий от общей сети 220 В и у вас есть сомнения, вопросы на тему того как это делается, остановитесь: вы можете устроить пожар или убить себя.*

Пример подключения исполнительного устройства (вентилятора) к плате Arduino UNO при помощи реле:



## Изучение принципа работы светодиодной ленты

Светодиодная RGB лента представляет собой гибкую ленту, с нанесенными на ней проводниками и RGB-светодиодами (полноцветными). В последнее время светодиодные ленты получили широкое распространение в архитектуре, авто и мото тюнинге, костюмах, декорациях и т.п. Также бывают водонепроницаемые ленты, которые можно использовать, к примеру, в бассейнах.

Светодиодные ленты бывают двух типов: аналоговые и цифровые. В нашем уроке мы будем использовать аналоговую светодиодную ленту.

Схема светодиодной RGB ленты

Лента поставляется в рулонах и состоит из секций длиной по 10 см.

В каждой секции размещается 3 RGB светодиода, типоразмера 5050. Т.е. в каждой секции получается, что содержится 9 светодиодов: 3 красных, 3 зеленых и 3 синих. Границы секций отмечены и содержат медные площадки. Поэтому, при необходимости, ленту можно обрезать и спокойно припаиваться. Схема светодиодной ленты:



## Подключения реле и светодиодной ленты к Arduino UNO

Для начала, убедимся в работоспособности реле на примере управления светодиодной лентой.

Необходимые компоненты:

1. Arduino UNO.
2. Коммутационное 4-х канальное реле.
3. Светодиодная лента.
4. Блок питания на 12 В.
5. Соединительные провода.

Подключение реле к Arduino:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Arduino UNO | GND | + 5 В | 2 | 3 | 4 |
| Реле | GND | VCC | IN1 | IN2 | IN3 |

Подключение реле к светодиодной ленте:

1. Вывод R ленты подключаем к крайнему контакту первого реле.
2. Вывод G ленты подключаем к крайнему контакту второго реле.
3. Вывод B ленты подключаем к крайнему контакту третьего реле.
4. Вывод GND ленты подключаем к GND блока питания на 12 В.
5. Центральные контакты реле1, реле2, реле3 подключаем к +12 В блока питания.

## C:\Users\Учитель\Downloads\IMG_20170726_175344.jpg

## Управление работой светодиодной ленты при помощи реле

**Программный код**

/\*

\* Lesson5\_1

\* Управление работой светодиодной ленты

\* при помощи реле

\*/

int relay1 = 2; //реле 1 подсоеденено к выводу 2

int relay2 = 3; //реле 2 подсоеденено к выводу 3

int relay3 = 4; //реле 3 подсоеденено к выводу 4

void setup()

{

pinMode(relay1, OUTPUT); //указываем,что пин реле1 используется на вывод

pinMode(relay2, OUTPUT); //указываем,что пин реле2 используется на вывод

pinMode(relay3, OUTPUT); //указываем,что пин реле3 используется на вывод

}

void loop()

{

digitalWrite(relay1, HIGH); //включаем реле1

delay(1000); //ждем 1 секунду

digitalWrite(relay1, LOW); //выключаем реле1

digitalWrite(relay2, HIGH); //включаем реле2

delay(1000); //ждем 1 секунду

digitalWrite(relay2, LOW); //выключаем реле2

digitalWrite(relay3, HIGH); //включаем реле3

delay(1000); //ждем 1 секунду

digitalWrite(relay1, HIGH); //выключаем реле1

digitalWrite(relay2, HIGH); //выключаем реле2

digitalWrite(relay3, HIGH); //выключаем реле3

delay(1000); //ждем 1 секунду

digitalWrite(relay1, LOW); //выключаем реле1

digitalWrite(relay2, LOW); //выключаем реле2

digitalWrite(relay3, LOW); //выключаем реле3

delay(1000); //ждем 1 секунду

}

## Основные понятия технологий построения сетей

**Сервер -**это компьютер, который обслуживает другие компьютеры в сети. Существуют разнообразные виды серверов, отличающиеся друг от друга услугами, которые они предоставляют; серверы баз данных, файловые серверы, принт-серверы, почтовые серверы, веб-серверы и т. д.

**Протоколы TCP/IP** (Transmission Control Protocol/Internet Protocol - Протокол управления передачей данных/Интернет протокол) являются основными межсетевыми протоколами и управляют передачей данных между сетями разной конфигурации и технологии. Именно это семейство протоколов используется для передачи информации в сети Интернет, а также в некоторых локальных сетях. Семейство протоколов TPC/IP включает все промежуточные протоколы между уровнем приложений и физическим уровнем. Общее их количество составляет несколько десятков.

При построении локальной сети на основе протокола TCP/IP каждый компьютер получает уникальный IP-адрес, который может назначаться либо DHCP-сервером, либо вручную.

**IP-адрес**  - уникальный [сетевой адрес](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%B0%D0%B4%D1%80%D0%B5%D1%81) [узла](https://ru.wikipedia.org/wiki/%D0%A3%D0%B7%D0%B5%D0%BB_%D1%81%D0%B5%D1%82%D0%B8) в [компьютерной сети](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), построенной на основе стека протоколов [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP).

**MAC-адрес** (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Media Access Control* — управление доступом к среде, также Hardware Address) — уникальный идентификатор, присваиваемый каждой единице активного оборудования или некоторым их интерфейсам в [компьютерных сетях](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) [Ethernet](https://ru.wikipedia.org/wiki/Ethernet" \o "Ethernet).

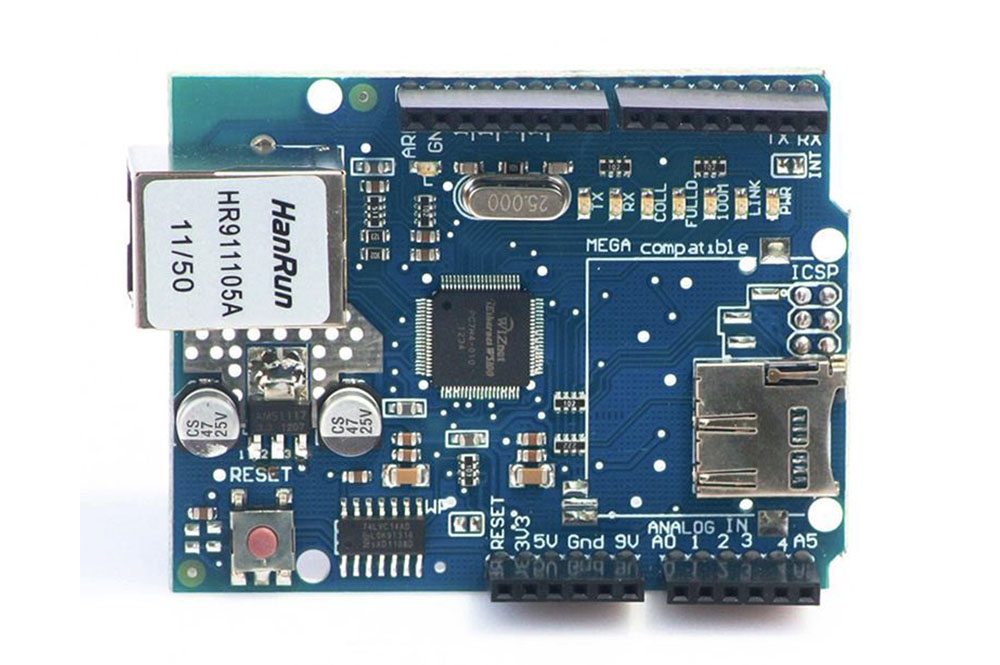
**DHCP** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Dynamic Host Configuration Protocol — протокол динамической настройки узла) — [сетевой протокол](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB), позволяющий компьютерам автоматически получать [IP-адрес](https://ru.wikipedia.org/wiki/IP-%D0%B0%D0%B4%D1%80%D0%B5%D1%81) и другие параметры, необходимые для работы в сети [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP). Данный протокол работает по модели «[клиент-сервер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80)». Для автоматической конфигурации компьютер-клиент на этапе конфигурации сетевого устройства обращается к так называемому [серверу](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5)) DHCP и получает от него нужные параметры.

**Сетевой шлюз** — это точка сети, которая служит выходом в другую сеть. В сети Интернет узлом или конечной точкой может быть или сетевой шлюз, или хост. Интернет-пользователи и компьютеры, которые доставляют веб-страницы пользователям — это хосты, а узлы между различными сетями — это сетевые шлюзы.

**DNS** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Domain Name System* — система доменных имён) — компьютерная [распределённая система](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D1%91%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) для получения информации о [доменах](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D0%B8%D0%BC%D1%8F). Чаще всего используется для получения IP-адреса по имени [хоста](https://ru.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82) (компьютера или устройства).

**Ethernet** - семейство технологий пакетной передачи данных для компьютерных сетей. Название «Ethernet» (буквально «эфирная сеть» или «среда сети») отражает первоначальный принцип работы этой технологии: всё, передаваемое одним узлом, одновременно принимается всеми остальными (то есть имеется некое сходство с радиовещанием). В настоящее время практически всегда подключение происходит через коммутаторы (switch), так что кадры, отправляемые одним узлом, доходят лишь до адресата (исключение составляют передачи на широковещательный адрес) — это повышает скорость работы и безопасность сети.

## Знакомство с Ethernet shield

Arduino Ethernet Shield W5100 одна из интереснейших плат расширения позволяющая управлять платой Arduino через локальную сеть или интернет. Это может пригодиться, если вам нужно получать какую-то информацию для работы из сети, разместить свой собственный веб сайт или управлять ардуино по сети. На данной плате расширения также имеется слот для microSD карты памяти. Для управления платой используется базовая библиотека Arduino IDE - Ethernet library, а для работы microSD картой - SD library. Для работы, помимо Arduino и, собственно, шилда вам также понадобится сетевой кабель (Витая пара).

Для соединения с локальной сетью используйте стандартный разъём 8P8C (RJ45) на борту. Один конец [патч-корда витой пары](http://amperka.ru/product/patch-cord) подключите к роутеру или хабу, второй — к Ethernet Shield.

Характеристики:

* Ethernet-чип: Wiznet W5500;
* рабочее напряжение: 5 В;
* размер буфера: 32 КБ;
* скорость соединения: 10/100 Мбит;
* занимаемые пины: SPI (MISO, MOSI, SCK), 10;
* габариты: 69×53 мм (RJ45 выступает на несколько мм).

## Подключение Ethernet shield к Arduino UNO



Для установки Ethernet shield вам необходимо поместить его сверху на плату Arduino UNO так, чтобы контакты Ethernet shield входили в соответствующие гнезда, установленные на Arduino UNO. Сетевой кабель (витую пару) необходимо подключить к Ethernet shield и к коммутатору. При подаче напряжения на плату Arduino на Ethernet shield должны засветиться индикаторы приема и передачи данных.

В плату встроены следующие индикаторы:

* *PWR* загорается, когда подведено питание;
* *LINK* загорается, если сеть доступна. Мигает при получении и передачи данных;
* *FULLD* горит, если установлено дуплексное соединение. То есть возможен одновременный приём и передача;
* *100M* горит, если соединение осуществлено на уровне 100 Мбит/с (не 10 Мбит/с);
* *RX* мигает при получении данных;
* *TX* мигает при передаче данных;
* *COLL* мигает при возникновении коллизий в сети. То есть когда в режиме полудуплекса два устройства пытаются одновременно передать пакет.

*Примечание: если индикатор LINK не горит, проверьте сетевой кабель (витую пару). Так же убедитесь, что маршрутизатор, к которому подключен Ethernet shield имеет доступ в Интернет.*

## Настройка сетевого интерфейса

Рассмотрим настройку сетевого интерфейса на примере программы ниже.

**Программный код**

\* Lesson6\_1

\*Настройка сетевого интерфейса

\*Ethernet shield

\*/

#include <SPI.h>

#include <Ethernet.h>

// Укажите MAC адресс вашего контроллера ниже

// В СЕТИ НЕ ДОЛЖНО БЫТЬ НЕСКОЛЬКО УСТРОЙСТВ С ОДИНАКОВЫМ MAC АДРЕСОМ

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

// Укажите статический IP адрес, если в сети не используется DHCP сервис

IPAddress ip(192, 168, 1,211);

// Инициализация Ethernet контроллера

EthernetClient client;

void setup() {

Serial.begin(9600); // Устанавливаем скорость 9600 бит/сек

// устанавливаем Ethernet соединение:

Serial.println("Connecting...");

if (Ethernet.begin(mac) == 0) { //если контроллеру не удалось получить

//настройки по DHCP, выводим сообщение

Serial.println("Failed to configure Ethernet using DHCP");

Ethernet.begin(mac, ip); //пытаемся настроить IP адрес вручную

}

delay(1000);

Serial.print("IP address: ");

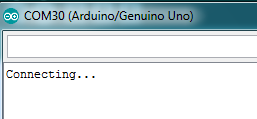
Serial.println (Ethernet.localIP()); //выводим текущий IP адрес устройства

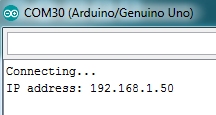
}

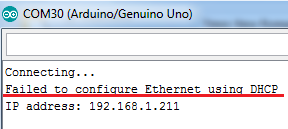
void loop() {

}

После загрузки и запуска программы, откройте Монитор порта и проверьте выведенное сообщение.

Сначала должна появиться строка Connecting…

Затем, если в сети используется сервис DHCP, то в течение нескольких секунд должно появиться сообщение об IP адресе, присвоенном устройству.

В противном случае, в течение приблизительно одной минуты контроллер будет пытаться найти сервис DHCP, затем выведется сообщение об его отсутствии и будет использоваться IP адрес, указанный в программном коде.

Примечание: для правильной работы, убедитесь, что MAC адрес является уникальным в пределах локальной сети, а IP адрес верен.

## Получение и отправка данных в Интернет

Рассмотрим пример получения и отправки данных на примере обращения к серверу прогноза погоды (samples.openweathermap.org).

**Программный код**

/\*

\* Lesson6\_2

\*Пример получения и отправки данных

\*на сервер прогноза погоды

\*/

#include <SPI.h>

#include <Ethernet.h>

// Укажите MAC адресс вашего контроллера ниже

// В СЕТИ НЕ ДОЛЖНО БЫТЬ НЕСКОЛЬКО УСТРОЙСТВ С ОДИНАКОВЫМ MAC АДРЕСОМ

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

// Укажите статический IP адрес, если в сети не используется DHCP сервис

char server[] = "samples.openweathermap.org"; // адрес сервера прогноза погоды

IPAddress ip(192, 168, 1,211);

// Инициализация Ethernet контроллера

EthernetClient client;

void setup() {

Serial.begin(9600); // Устанавливаем скорость 9600 бит/сек

// устанавливаем Ethernet соединение:

Serial.println("Connecting...");

if (Ethernet.begin(mac) == 0) { //если контроллеру не удалось получить

//настройки по DHCP, выводим сообщение

Serial.println("Failed to configure Ethernet using DHCP");

Ethernet.begin(mac, ip); //пытаемся настроить IP адрес вручную

}

delay(1000);

Serial.print("IP address: ");

Serial.println (Ethernet.localIP()); //выводим текущий IP адрес устройства

// если смогли подключиться к серверу, то отправляем запрос:

if (client.connect(server, 80)) {

Serial.println("connected");

// формируем HTTP запрос:

const String html\_cmd1 = "GET /data/2.5/forecast?id=524901&appid=b1b15e88fa797225412429c1c50c122a1 HTTP/1.1";

const String html\_cmd2 = "Host: samples.openweathermap.org";

const String html\_cmd3 = "Connection: close";

// выполняем HTTP запрос:

client.println(html\_cmd1);

client.println(html\_cmd2);

client.println(html\_cmd3);

client.println();

} else {

// если не смогли подключиться к серверу:

Serial.println("connection failed");

}

}

void loop() {

// выводим на экран все, что получили от сервера

if (client.available()) {

char c = client.read();

Serial.print(c);

}

// если произошло отключение от сервера, останавливаем клиент:

if (!client.connected()) {

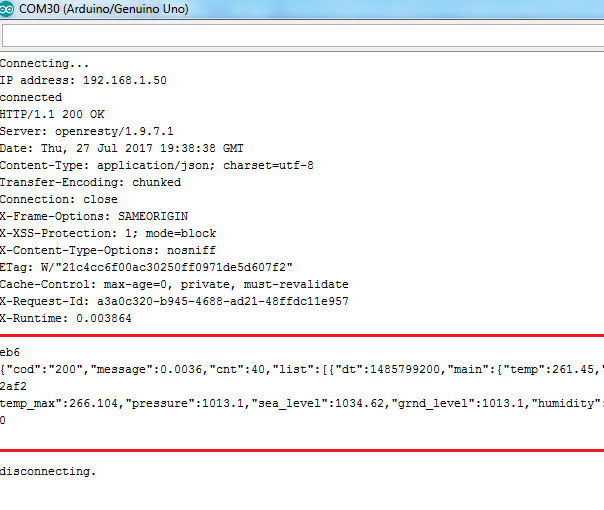
Serial.println();

Serial.println("disconnecting.");

client.stop();

}

}

После загрузки программы, откройте Монитор порта. В нем должны выводиться данные о прогнозе погоды, полученные с сервера samples.openweathermap.org .