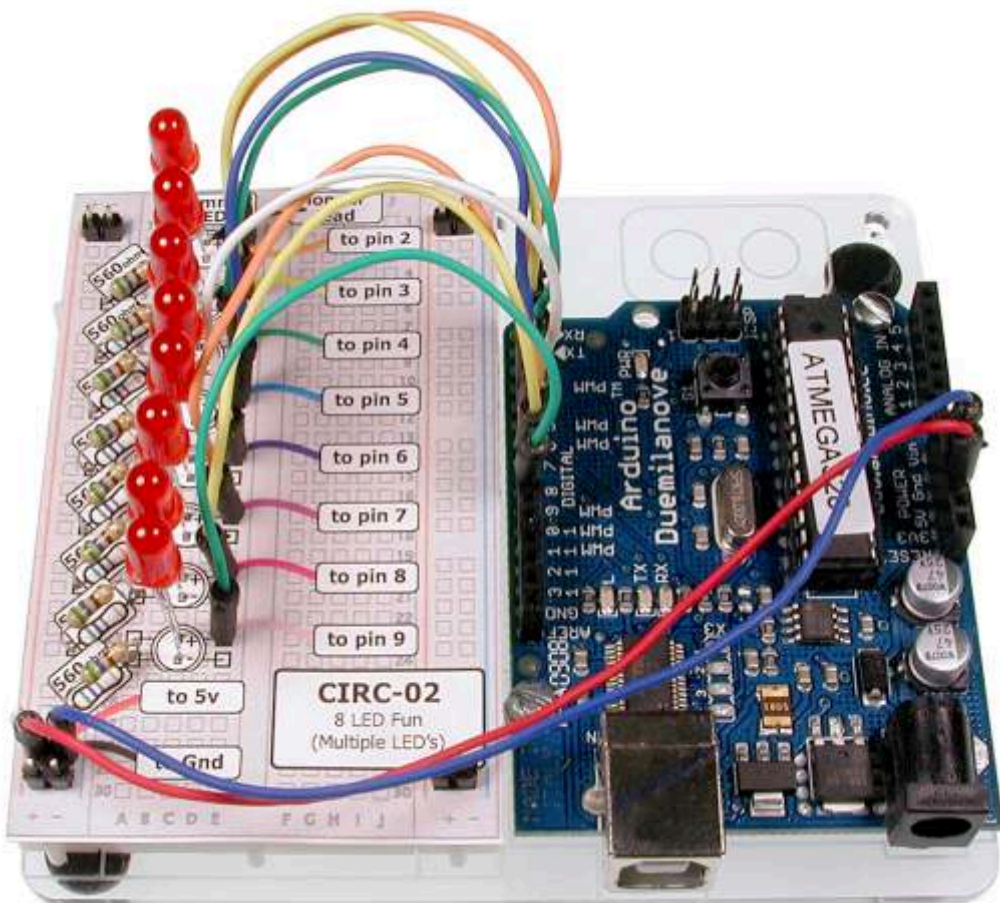


# Руководство по освоению Arduino



## ARDX

Руководство с открытым исходным кодом для Arduino



# Введение

## Об этом наборе

Основная цель этого набора – интересно и с пользой провести время. А помимо этого - освоить разнообразные электронные компоненты путем сборки небольших простых и интересных устройств. Вы получаете работающее устройство и инструмент, позволяющий понять принцип действия. Если у Вас что-то не получается, есть вопросы, либо Вам необходима дополнительная помощь – пишите нам на [help@oomlout.com](mailto:help@oomlout.com).



## Об открытом аппаратном обеспечении

Все проекты .:oomlout:. являются открытыми. Что это значит? Это значит, что все данные, необходимые для производства этого набора, доступны для бесплатного скачивания. И даже больше – вы тоже можете воспроизводить или менять любые материалы по своему усмотрению и затем распространять их. В чем смысл? Все просто, эти материалы выпущены под лицензией Creative Commons. Это значит, что, используя .:oomlout:., вы обязуетесь упоминать его в своих разработках и распространять их таким же образом. Почему? Мы выросли играя и учась с открытым программным обеспечением и это был хороший опыт, мы думаем этот подход оправдывает себя и с реальными компонентами проектов (“железом”).

Более подробно о лицензии Creative Commons вы можете узнать по адресу  
<http://ardx.org/CCLI>

## Об .: OOMLOUT :.

Мы отважная маленькая компания разработчиков, специализирующаяся на производстве «восхитительно прекрасной открытой продукции»

Наши свежие проекты можно увидеть по адресу:

<http://www.oomlout.com>

## О SOLARBOTICS

Мы начали производить робототехнические наборы BEAM больше 15 лет назад (правда-правда!), сейчас мы также поставляем классные электронные наборы

<http://www.solarbotics.com/>

## О проблемах

Мы стараемся поставлять товары высочайшего качества. Если Вы обнаружите неясную инструкцию, недостачу детали или просто захотите задать вопрос, мы сделаем все возможное, чтобы помочь Вам.

[help@oomlout.com](mailto:help@oomlout.com) / [help@solarbotics.com](mailto:help@solarbotics.com)

(Мы предпочитаем знать о проблемах – это помогает нам улучшать последующие версии.)

**Спасибо за то, что выбрали .:oomlout:.  
(и Solarbotics)**

## .: СОДЕРЖАНИЕ .:

**Введение/подготовка к проекту**

|        |                                       |    |
|--------|---------------------------------------|----|
| {ASEM} | Сборка устройства                     | 02 |
| {INST} | Установка программного обеспечения    | 03 |
| {PROG} | Небольшая справка по программированию | 04 |
| {ELEC} | Небольшая справка по электронике      | 06 |

**Устройства**

|          |   |    |
|----------|---|----|
| {CIRC01} | Начнем работу (мигающий светодиод)            | 08 |
| {CIRC02} | Схема с 8 светодиодами                        | 10 |
| {CIRC03} | Крутись мотор, крутись (транзистор и мотор)   | 12 |
| {CIRC04} | Одиночный сервопривод (сервоприводы)          | 14 |
| {CIRC05} | Еще 8 светодиодов (сдвиговый регистр 74НС595) | 16 |
| {CIRC06} | Музыка (пьезоэлемент)                         | 18 |
| {CIRC07} | Нажатие на кнопку (кнопки)                    | 20 |
| {CIRC08} | Кручение (потенциометры)                      | 22 |
| {CIRC09} | Свет (фоторезисторы)                          | 24 |
| {CIRC10} | Температура (температурный датчик TMP36)      | 26 |
| {CIRC11} | Большие нагрузки (реле)                       | 28 |
| {CIRC12} | Многоцветное свечение (светодиоды RGB)        | 30 |

# 01 АЕМ

собираем  
компоненты

## :: Сборка устройства ::



Основание  
для Arduino  
x1



Макетная  
плата  
x1



Arduino  
x1



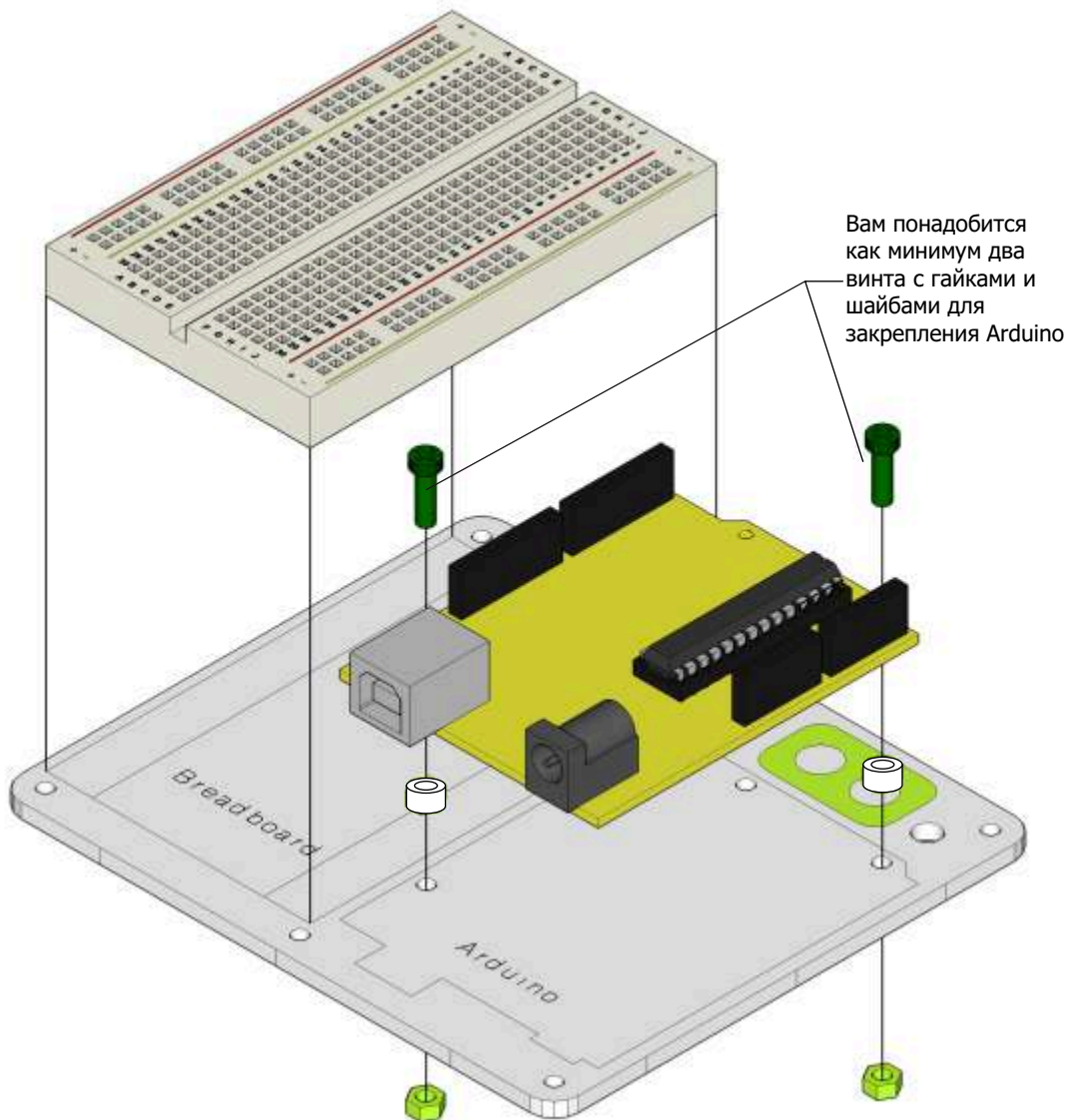
4-40 x 3/8" винт  
x3



4-40 гайка  
x3



#4 x 1/16"  
шайба  
x3



Вам понадобится  
как минимум два  
винта с гайками и  
шайбами для  
закрепления Arduino

# .: Установка программного обеспечения (интерфейса пользователя IDE) :.

**02 INST**  
ИНСТАЛЛЯЦИЯ  
(программное и аппаратное обеспечение)

Интерфейс пользователя предназначен для программирования Arduino. Все это кажется поначалу немного сложным, но когда вы все установите и немного поиграете с программой – все ее секреты раскроются вам сами собой.

**Шаг 1: Загрузка программного обеспечения**  
Откройте страницу

<http://arduino.cc/en/Main/Software>

и загрузите версию для вашей операционной системы

## Windows XP

**Шаг 2: Распаковка архива**

Программа

arduino-00x x-win.zip (x x - номер версии)

Лучше всего в папку

c:\Program Files\

**Шаг 3: Создание ярлыка**

Откройте

c:\program files\arduino-00x x\ (x x - номер версии)

Клик правой кнопкой мыши

Arduino.exe (send to>Desktop (создать ярлык))

**Шаг 4: Подключение**

Подключите Ваш Arduino при помощи USB-кабеля к свободному порту USB.

На экране должно появиться диалоговое окно

**Шаг 5: Добавление нового оборудования**

Пропустите опцию "поиск драйверов в интернете" (просто щелкните "next")

Далее

щелкните "Install from a list or specific location (Advanced)"

Установочные директории  
Duemilanove Board

c:\program files\arduino-00x x\drivers\FTDI USB Drivers\

Uno Board

c:\program files\arduino-00x x\drivers\

## Vista, Seven

**Шаг 5: Добавление нового оборудования**

Запустите Device Manager

Start > Run > devmgmt.msc

Выберите Arduino

Other Devices > Arduino Uno (Uno)

Обновите драйвер

щелкните "Update Driver"

Выберите драйвер

щелкните "Browse My Computer for Driver Software"

c:\program files\arduino-00x x\drivers\

## Mac OSX

**Шаг 2: Откройте.dmg**

Откройте (загрузите)

arduino-00x x-mac.dmg (x x - номер версии)

**Шаг 3: Скопируйте приложение**

Найдите

"Arduino" (в секции браузера "devices")

Переместите

Приложение "Arduino" в папку "Applications" (приложения)

**Шаг 4: Установите драйверы**

.:Только для плат Duemilanove.:

Найдите

"Arduino" device

Двойной щелчок мышью, установите:

FTDI Drivers for Intel Macs (x x x x).pkg

(FTDI Drivers for PPC Macs (x x x x).pkg

Перезагрузитесь

**Шаг 5: Включите Arduino**

Подключите Ваш Arduino при помощи USB-кабеля к свободному порту USB

.: Проблемы при установке? :.  
.: Нужны подробности? Используйте Linux? :.  
.: <http://ardx.org/LINU> :.

## Основы программирования Arduino

Arduino программируется на языке C. Данный раздел ориентирован на тех, кто имеет небольшой опыт программирования, и нуждается только в пояснении особенностей языка C и интерфейса Arduino. Если все это кажется Вам немного сложным, не беспокойтесь, начинайте работать с примерами устройств и понимание придет в процессе. Для более подробного изучения основ используйте сайт [arduino.cc](http://arduino.cc).

### СТРУКТУРА

Каждая программа Arduino (часто называемая «скетч») имеет две обязательные функции (также называемые подпрограммами).

```
void setup() { }
```

Все команды, заключенные между фигурными скобками, выполняются только один раз, при первом запуске программы.

```
void loop() { }
```

Эта подпрограмма выполняется циклически вплоть до отключения питания, после завершения подпрограммы `setup()`.

### Синтаксис

Требования к форматированию в языке C вызывают некоторые затруднения у начинающих (с другой стороны, благодаря своей структуре, язык C обладает большими возможностями). Если Вы запомните следующие правила, этого будет вполне достаточно.

```
// (однострочный комментарий)
```

Часто используется для размещения в тексте программы комментариев. Можно пояснять, что значит каждая строка программы. Все что размещается после двойной черты и до конца строки будет игнорироваться компилятором.

```
/* */ (многострочный комментарий)
```

Вы можете использовать эту структуру, если Вам надо создать подробный комментарий на нескольких строках. Все находящееся между этими символами будет игнорироваться компилятором.

```
{ } (фигурные скобки)
```

Используются для определения начала и конца блока команд (используются в функциях и циклах).

```
;
```

Каждая команда должна заканчиваться этим символом (потерянная точка с запятой — наиболее распространенная ошибка, приводящая к невозможности компиляции).

### Переменные

Любая программа всего лишь определенным образом манипулирует числами. Переменные помогают жонглировать цифрами.

**int** (целочисленная)

Основная рабочая лошадка, хранится в памяти с использованием двух байт (16 бит). Может содержать целое число в диапазоне -32 768 ... 32 767.

**long** (длинная)

Используется в том случае, когда не хватает емкости int. Занимает в памяти 4 байта (32 бита) и имеет диапазон -2 147 483 648 ... 2 147 483 647.

**boolean** (двоичная)

Простой тип переменной типа True/False. Занимает только один бит в памяти.

**float** (с плавающей запятой)

Используется для вычислений с плавающей запятой. Занимает в памяти 4 байта (32 бита) и имеет диапазон -3.4028235E+38.

**char** (символ)

Хранит один символ, используя кодировку ASCII (например «А» =65). Использует один байт памяти (8 бит). Arduino оперирует со строками как с массивами символов.

.:Для более подробной справки по программированию посетите:  
<http://ardx.org/PROG>

## Математические операторы

Операторы используются для преобразования чисел.

= (присвоение) делает что-то равным чему-то(например  $x=10*2$  записывает в переменную x число 20).  
% остаток от деления). Например  $12\%10$  дает результат 2.  
+ (сложение)  
- (вычитание)  
\* (умножение)  
/ (деление)

## Операторы сравнения

Операторы, используемые для логического сравнения.

== (равно) (Например  $12==10$  не верно (FALSE),  $5==5$  верно(TRUE).)  
!= (не равно) (Например  $12!=10$  верно (TRUE),  $5!=5$  не верно (FALSE).)  
< (меньше) (Например  $12<10$  не верно (FALSE),  $12<12$  не верно (FALSE),  $12<14$  верно (TRUE).)  
> (больше) (Например  $12>10$  верно (TRUE),  $12>12$  не верно (FALSE),  $12>14$  не верно (FALSE).)

## Управляющие структуры

Для определения порядка выполнения команд (блоков команд) служат управляющие структуры. Здесь приведены только основные структуры. Более подробно можете ознакомиться на сайте Arduino.

```
if (условие 1) {}  
else if (условие 2) {}  
else {}
```

Если условие 1 верно (TRUE) выполняются команды в первых фигурных скобках. Если условие 1 не верно (FALSE) то проверяется условие 2. Если условие 2 верно, то выполняются команды во вторых фигурных скобках, в противном случае выполняются команды в третьих фигурных скобках.

```
for (int i=0;  
i<число повторов;  
i++) {}
```

Эта структура используется для определения цикла. Цикл повторяется заданное число раз. Переменная i может увеличиваться или уменьшаться.

## Цифровые сигналы

```
digitalWrite(pin, value);
```

Если порт установлен в режим OUTPUT, в него можно записать HIGH (логическую единицу, +5В) или LOW (логический ноль, GND).

```
pinMode(pin, mode);
```

Используется, чтобы определить режим работы соответствующего порта. Вы можете использовать адреса портов 0...19 (номера с 14 по 19 используются для описания аналоговых портов 0...5). Режим может быть или INPUT (вход) или OUTPUT (выход).

```
digitalRead(pin);
```

Если порт установлен в режим INPUT эта команда возвращает значение сигнала на входе HIGH или LOW.

## Аналоговые сигналы

Arduino - цифровое устройство, но может работать и с аналоговыми сигналами при помощи следующих двух команд:

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

Некоторые порты Arduino (3,5,6,9,10,11) поддерживают режим ШИМ (широтно-импульсной модуляции). В этом режиме в порт посылаются логические единицы и нули с очень большой скоростью. Таким образом среднее напряжение зависит от баланса между количеством единиц и нулей и может изменяться в пределах от 0 (0В) до 255 (+5В).

```
analogRead(pin);
```

Если аналоговый порт настроен в режим INPUT, то можно измерить напряжение на нем. Может принимать значения от 0 (0В) до 1024 (+5В).

## Основы электроники

Вам не потребуется опыт работы с электроникой для работы с этим набором. Ниже приведены некоторые сведения об электронных компонентах, которые позволят Вам легче их идентифицировать и возможно, понять принцип действия. Если Вам что-либо не понятно или компонент работает не так как должен — обращайтесь в нашу службу поддержки по адресу [help@oomlout.com](mailto:help@oomlout.com)

## Описание компонентов

**LED**

(светодиод)

**Что делает:**

Излучает свет если пропустить через него небольшой ток. Ток может проходить через светодиод только в одном направлении.

**Вид:**

Похож на небольшую лампочку.

**Число выводов:** 2

(более длинный вывод (анод) подключается к положительному потенциалу)

**Важно:**

Работает только при правильном включении. Требуется резистор для ограничения силы тока.

**Дополнительная информация:**

<http://ardx.org/LED>

**Диод****Что делает:**

Электронный эквивалент однонаправленного клапана. Ток через диод может течь только в одну сторону.

**Вид:**

Обычно цилиндрической формы с выводами на противоположных сторонах (на одной из сторон нанесена линия, определяющая полярность).

**Число выводов:** 2**Важно:**

Пропускает ток только в одном направлении. Ток будет течь, если сторона с полоской подключена к низкому потенциалу, а другая сторона к более высокому.

**Дополнительная информация:**

<http://ardx.org/DIOD>

**Резистор****Что делает:**

Ограничивает силу тока, протекающего в цепи.

**Вид:** Обычно цилиндрической формы с выводами на противоположных сторонах. Значение номинала указывается при помощи цветных полосок (информацию о цветовой кодировке смотри далее).

**Число выводов:** 2**Важно:**

Легко перепутать номинал. Тщательно проверьте значение.

**Дополнительная информация:**

<http://ardx.org/RESI>

**Транзистор****Что делает:**

Используется для коммутации или усиления сигналов.

**Вид:**

Выпускается в разнообразных корпусах. Название обычно наносится на корпус (в данном наборе используются P2N2222AG)

**Число выводов:**

3 (База, Коллектор, Эмиттер)

**Важно:**

Не путать выводы. Для ограничения тока часто используются резисторы.

**Дополнительная информация:**

<http://ardx.org/TRAN>

**Сервопривод****Что делает:**

Преобразует электрические импульсы в угол поворота оси.

**Вид:**

Пластиковая коробочка с тремя проводками и металлической осью с кронштейном.

**Число выводов:**

3

**Важно:**

Убедитесь в правильном подключении (разъем без ключа).

**Дополнительная информация:**

<http://ardx.org/SERV>

**Коллекторный двигатель****Что делает:**

Вращается, когда через него протекает электрический ток.

**Вид:**

Это просто, он выглядит как мотор. Обычно цилиндрической формы с осью посередине.

**Число выводов:**

2

**Важно:**

Используйте транзистор или реле соответствующей мощности для подключения двигателя.

**Дополнительная информация:**

<http://ardx.org/MOTO>



## Описание компонентов (продолжение)

### Пьезоэлемент


**Что делает:**

Импульс тока преобразуется в щелчок. Последовательность импульсов преобразуется в музыкальный тон.

**Вид:** В этом наборе он выглядит как черный бочонок. Иногда может выглядеть как золотой диск.

**Число выводов:** 2

**Важно:**

Трудно подключить неправильно.

**Дополнительная информация:**

<http://ardx.org/PIEZ>

### ИС (интегральная микросхема)


**Что делает:**

Содержит в себе электронику любой сложности.

**Вид:** Название компонента обычно нанесено на корпус (часто для того чтобы прочесть требуется увеличительное стекло или хорошее освещение).

**Число выводов:**

от двух до нескольких сотен. В этом наборе TMP36 имеет 3 вывода и 74HC595 имеет 16 выводов.

**Важно:**

Не перепутать ориентацию микросхемы.

**Дополнительная информация:**

<http://ardx.org/ICIC>

### Кнопка


**Что делает:**

Замыкает контакты при нажатии.

**Вид:**

Маленький квадратик с выводами внизу и кнопкой наверху.

**Число выводов:**

4

**Важно:**

Практически квадратная, можно вставить с поворотом на 90 градусов.

**Дополнительная информация:**

<http://ardx.org/BUTT>

### Потенциометр


**Что делает:**

Резистор с номиналом, величина которого зависит от угла поворота оси.

**Вид:**

Выпускается в разнообразных корпусах.

**Число выводов:**

3

**Важно:**

Может иметь линейную или логарифмическую шкалу.

**Дополнительная информация:**

<http://ardx.org/POTE>

### Фоторезистор


**Что делает:**

Резистор с номиналом, величина которого зависит от интенсивности падающего на него света.

**Вид:**

Обычно выглядит как небольшой диск с прозрачным покрытием и зигзагообразным проводником под ним.

**Число выводов:**

2

**Важно:** Чтобы получить полезный сигнал, необходимо использовать фоторезистор как часть делителя напряжения.

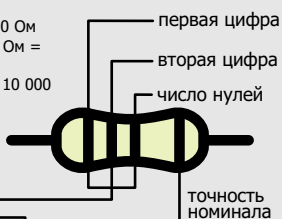
**Дополнительная информация:**

<http://ardx.org/PHOT>

## Цветовая кодировка резисторов

**Примеры:**

зеленый-голубой-коричневый = 560 Ом  
красный-красный-красный = 2 200 Ом = 2.2 кОм  
Коричневый-черный-оранжевый = 10 000 Ом = 10 кОм



|                |                |                   |
|----------------|----------------|-------------------|
| 0 - черный     | 5 - зеленый    | 20% - нет полоски |
| 1 - коричневый | 6 - синий      | 10% - серебряный  |
| 2 - красный    | 7 - фиолетовый | 5% - золотой      |
| 3 - оранжевый  | 8 - серый      |                   |
| 4 - желтый     | 9 - белый      |                   |

## Обрезка выводов

Некоторые компоненты в этом наборе поставляются с очень длинными выводами. Для более удобного использования можно сделать следующие изменения:

**Светодиоды:**

Укоротите длинный вывод до 10 мм, короткий — до 7.


**Резисторы:**

Согните выводы вниз под углом 90° и укоротите до 6 мм.

**Остальные компоненты:**

У остальных компонентов можете укоротить выводы по своему усмотрению.

# CIRC-01

..Начнем работу..  
..(мигающий светодиод)..  


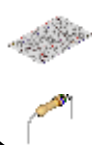
## Описание задания:

Светодиоды очень широко используются в различных устройствах, поэтому мы и добавили их в этот набор. Начнем с чего-то очень простого, например будем включать и выключать светодиод в бесконечном цикле. Отберите детали по списку, прикрепите карточку задания к макетной плате, установите все детали. После сборки устройства, необходимо загрузить программу. Для этого подключите Arduino к свободному порту USB. Затем установите порт в программе **Tools>Serial Port>**(порт, назначенный для Arduino). Загрузка программы осуществляется из меню **File>Upload to I/O Board** (ctrl+U). Наконец, наслаждайтесь способностью контролировать светодиод!

Если Вы столкнулись с проблемами при загрузке — обратитесь к руководству: <http://ardx.org/TRBL>

## УСТРОЙСТВО:

### Компоненты:



Карточка задания  
**CIRC-01**  
x1  
Резистор 560 Ом  
зеленый-синий-коричневый  
x1



2-контактный  
разъем  
x4

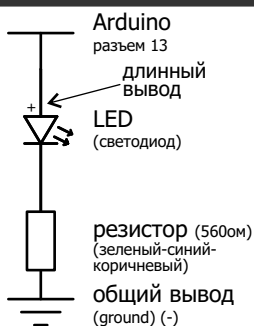


Светодиод  
10мм  
x1



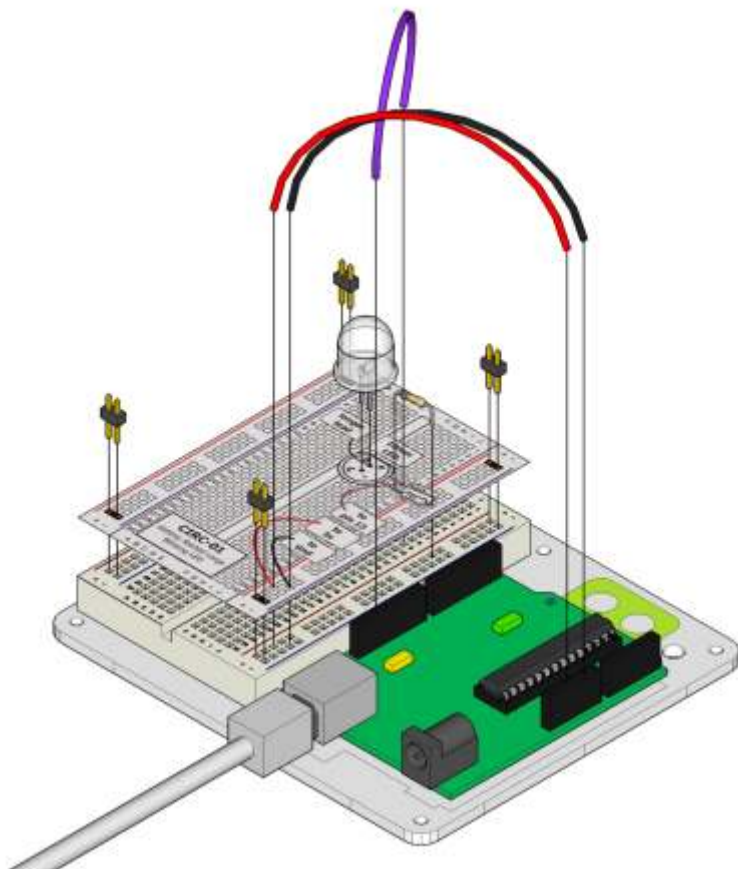
Провод

### Схема



### в Интернете:

..скачать..  
карточку задания  
<http://ardx.org/BBLS01>  
..посмотреть..  
видео сборки  
<http://ardx.org/VIDE01>



## File > Examples > 1.Basic > Blink

(это пример с сайта arduino.cc, также там можно найти много отличных идей)

```
/* мигание
 * включает светодиод на 1 секунду, выключает на 1 секунду,
 * функция вызывается по кругу
 * Created 1 June 2005 By David Cuartielles
 * http://arduino.cc/en/Tutorial/Blink
 * based on an original by H. Barragan for the wiring i/o board
 */

int ledPin = 13;    // светодиод подключен к выходу 13

// функция начальных установок setup() вызывается 1 раз в начале скетча
void setup() {      // устанавливаем 13 контакт в режим вывода:
  pinMode(ledPin, OUTPUT);
}

// функция loop() вызывается по кругу
// пока Arduino подключен к питанию
void loop() {
  digitalWrite(ledPin, HIGH); // включение светодиода
  delay(1000);                // задержка 1 сек
  digitalWrite(ledPin, LOW);  // выключение светодиода
  delay(1000);                // задержка 1 сек
}
```

## Не работает? (3 проблемы и их решения)

### Светодиод не зажигается?

Светодиод работает только при соблюдении правильной полярности. Попробуйте вытащить его и вставить наоборот. Не беспокойтесь — неправильное включение не должно его повредить. Убедитесь, что подключили его к порту номер 13.

### Программа не загружается?

Это случается иногда. Чаще всего причиной является неправильно указанный последовательный порт. Изменить порт можно в меню **tools>serial port>**

### Все еще не работает?

Неисправное оборудование это неприятно, пошлите нам e-mail с описанием проблемы и мы свяжемся с Вами как можно скорее.

[help@oomlout.com](mailto:help@oomlout.com)

## Усовершенствуем устройство

### Поменяем управляющий порт:

Светодиод подключен к порту 13, однако мы можем использовать любой из портов Arduino. Переключите провод на любой порт по Вашему выбору (от 0 до 13, или аналоговые порты — от 14 до 19). Затем замените команду:  
`int ledpin = 13;` на `int ledpin = ваш номер;`  
Загрузите программу в Arduino (ctrl-u).

### Изменим частоту мигания:

Для изменения времени включения и выключения — редактируйте аргументы функции `delay(x)`. Задержка определяется как  $x = \text{число секунд} * 1000$ . Например: для задания более низкой частоты мигания программа должна выглядеть следующим образом:

```
digitalwrite(ledpin,HIGH);
delay(2000);
digitalwrite(ledpin,LOW);
delay(2000);
```

### Управление яркостью свечения:

Кроме простого включения и выключения можно контролировать яркость свечения светодиода (более подробно это будет объясняться в последующих заданиях). Подключите светодиод к порту номер 9:  
`int ledpin = 13;` замените на `int ledpin = 9;`  
Замените код в фигурных скобках процедуры `loop()` следующим:  
`analogwrite(ledPin, new number);`  
`newnumber` — любое число из диапазона 0...255. 0 будет соответствовать отключенному светодиоду, 255 — максимальная яркость.

### Плавное изменение яркости:

Откройте пример:

#### File > Examples > 3.Analog > Fading

Загрузите эту программу в Arduino и посмотрите получившийся эффект.

## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC01>



### Описание задания:

Мы заставили светодиод мигать, время поднимать ставки! Давайте подключим сразу восемь светодиодов. У нас также будет возможность создать несколько различных световых эффектов. Это устройство — хорошая основа для дальнейших экспериментов и освоения Arduino. Одновременно с управлением светодиодами рассмотрим методы оптимизации программы.

`for()` `loops` - используется, если Вам необходимо выполнить часть программы несколько раз.

`arrays []` - используется для упрощения работы с переменными (фактически это группа переменных).

### УСТРОЙСТВО:

#### Компоненты:



Карточка задания CIRC-02 x1



2-контактный разъем x4



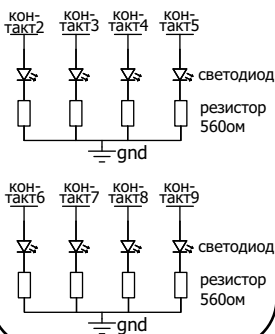
Светодиод 5мм, зеленый x8



Провод

Резистор 560 Ом  
Зеленый-синий-коричневый x8

#### Схема



#### в Интернете:

.:скачать:.

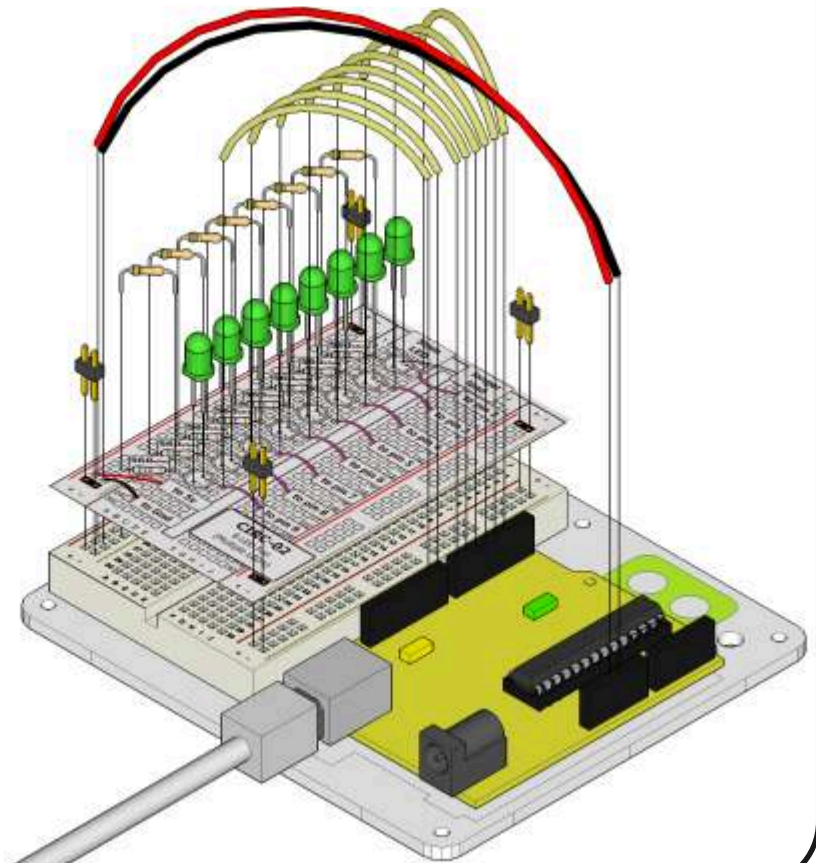
карточку задания

<http://ardx.org/BBLS02>

.:посмотреть:.

видео сборки

<http://ardx.org/VIDE02>



**его можно загрузить с <http://ardx.org/CODE02>**

(скопируйте текст и вставьте его в новое окно Arduino Sketch)

```
//переменные светодиодов
int ledPins[] = {2,3,4,5,6,7,8,9};
//множество, задающее какой
//светодиод присоединен к какому
//порту, например светодиод №0 - к
//порту2

void setup()
{
  for(int i = 0; i < 8; i++){
    //цикл повторяющийся 8 раз
    pinMode(ledPins[i],OUTPUT);
    //устанавливает порты, к которым подключены
    //светодиоды, в режим вывода
  }
}

void loop()          // бесконечный цикл
{
  oneAfterAnotherNoLoop();
  //включает каждый светодиод по одному и
  //затем так же выключает
  //oneAfterAnotherLoop();
  //делает то же самое что oneAfterAnotherNoLoop
  //но команда короче
  //oneOnAtATime();
  //inAndout();
}

/*
 * oneAfterAnotherNoLoop() - зажигает один
```

```
светодиод, задержка на
delayTime, включение
следующего светодиода,
выключение

void oneAfterAnotherNoLoop(){
  int delayTime = 100;
  //время задержки между включениями
  //светодиодов в миллисекундах
  digitalWrite(ledPins[0], HIGH);
  //включение светодиода №0
  //на выходе 2

  delay(delayTime); //задержка на delayTime
                    //(в миллисекундах)

  ...
  digitalWrite(ledPins[7], HIGH);
  //включение светодиода №7 на выходе 9

  delay(delayTime); //задержка на delayTime (в миллисекундах)
  //выключение каждого светодиода

  digitalWrite(ledPins[7], LOW); //выключение
  светодиода №7
  delay(delayTime); //задержка на delayTime
                    //(в миллисекундах)

  ...

  -----полная программа в электронной версии-----
```

**Не работает?** (3 проблемы и их решения)**Некоторые светодиоды не загораются**

Легко ошибиться и вставить светодиод наоборот. Проверьте полярность всех неработающих светодиодов.

**Светодиоды загораются в неправильном порядке**

При подключении восьми проводов легко можно перепутать пару. Убедитесь, что первый светодиод подключен к порту 2 и каждый последующий к следующему порту по порядку.

**Начните сначала**

Часто проще все разобрать и собрать заново, чем искать неисправность.

**Усовершенствуем устройство****Использование циклов:**

В функции loop() записаны четыре команды. Последние три начинаются с символа «//». Это значит, что компилятор игнорирует эти строки (рассматривает их как комментарий). Для использования циклов измените текст программы следующим образом:

```
//oneAfterAnotherNoLoop();
oneAfterAnotherLoop();
//oneOnAtATime();
//inAndout();
```

Загрузите программу в Arduino и обратите внимание, что программа выполняется так же. Обратите внимание на эти две функции, они выполняют одно и то же, но с использованием различных принципов (вторая функция

использует структуру for).

**Дополнительные эффекты:**

Устали от этого эффекта? Попробуйте два других образца. Убирайте поочередно значки комментария («//») с двух последних строк, загружайте программу в Arduino и любуйтесь новыми эффектами.

**Создайте свой собственный эффект:**

Попробуйте изменить что-нибудь в программе. Основная идея во включении светодиодов командой digitalWrite(pinNumber, HIGH); и выключении при помощи команды digitalWrite(pinNumber, LOW); Не бойтесь экспериментировать, вне зависимости от того, что Вы измените, ничего не должно сломаться!

**Есть еще вопросы?**

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC02>



### Описание задания:

Порты Arduino идеально подходят для управления чем-то маленьким с небольшим потреблением (например светодиод). Таким образом, для управления большими нагрузками (например моторами) требуется использовать дополнительный элемент — транзистор. Транзистор очень полезный элемент, он позволяет управлять большими нагрузками и при этом потребляет совсем небольшой ток. Транзистор имеет три вывода. Для транзисторов типа NPN Вы должны подключать нагрузку к коллектору, а эмиттер к «земле» (нулевому потенциалу). Если от базы к эмиттеру будет протекать небольшой ток (если подключить к базе сигнал HIGH от Arduino), то транзистор «откроется», ток потечет через транзистор и мотор начнет вращаться. Существует великое множество транзисторов, позволяющих решать самые разнообразные задачи. В наборе используются транзисторы общего применения P2N2222AG. Максимальное напряжение, с которым может работать этот транзистор, составляет 40В, максимальный ток — 600 мА. Эти параметры позволяют использовать его с нашим мотором. Более подробно можете параметры транзистора приведены по адресу: <http://ardx.org/2222>.

В качестве защиты от обратного тока используется диод 1N4001. Более подробно про это рассказано по адресу: <http://ardx.org/4001>

### УСТРОЙСТВО:

#### Компоненты:



**Карточка задания CIRC-03**  
x1



**2-контактный разъем**  
x4



**Транзистор P2N2222AG (ТО92)**  
x1



**Провод**  
x1



**Коллекторный двигатель**  
x1

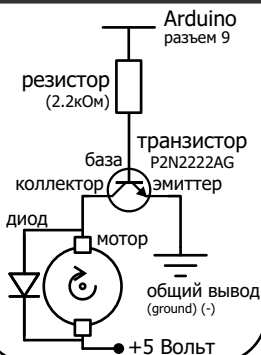


**Диод (1N4001)**  
x1



**Резистор 2.2k Ом Красный-красный-красный**  
x1

#### Схема



#### в Интернете:

.:скачать.:

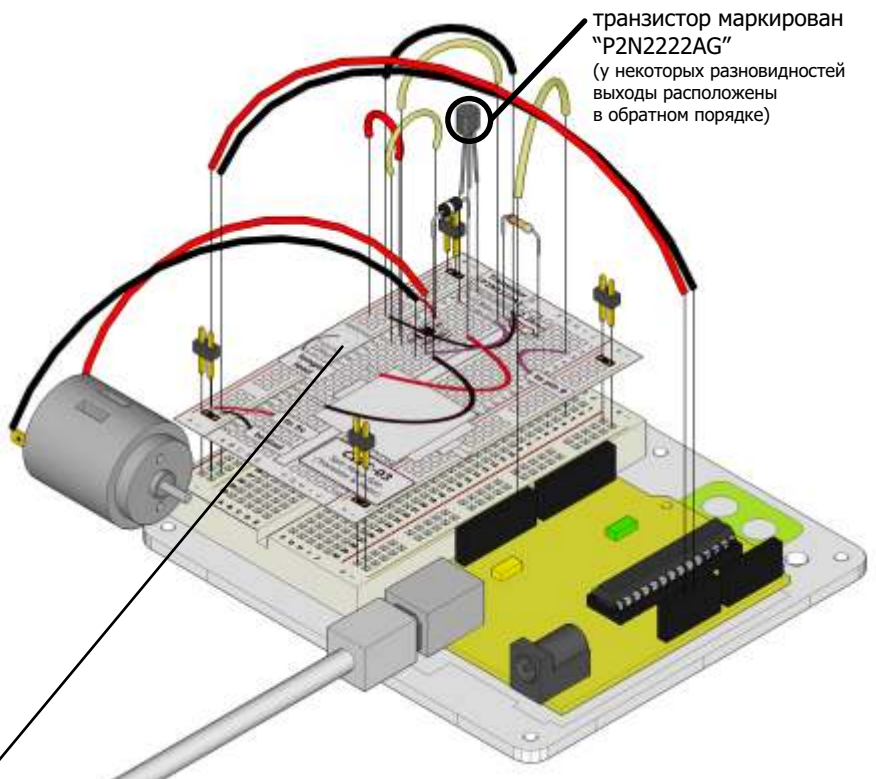
карточку задания

<http://ardx.org/BBL503>

.:посмотреть.:

видео сборки

<http://ardx.org/VIDE03>



.:Примечание: если при запуске мотора Arduino перезагружается или зависает, тогда необходимо установить дополнительный конденсатор.:

**его можно загрузить с <http://ardx.org/CODE03>**

(скопируйте текст и вставьте его в новое окно Arduino Sketch)

```

int motorPin = 9; //мотор подсоединен к разъему 9

void setup() //запускается единожды
{
  pinMode(motorPin, OUTPUT);
}

void loop() // выполняется бесконечно
{
  motorOnThenOff();
  //motorOnThenOffwithSpeed();
  //motorAcceleration();
}

/* motorOnThenOff() - включает мотор затем
выключает его (обратите внимание что код идентичен
тому что мы использовали для мигающего светодиода)
*/
void motorOnThenOff(){
  int onTime = 2500;
  //время во включенном состоянии
  int offTime = 1000;
  //время в выключенном состоянии
  digitalWrite(motorPin, HIGH);
  // включение мотора
  delay(onTime);
  // задержка на onTime миллисекунд
  digitalWrite(motorPin, LOW);
  // выключение мотора
  delay(offTime); // задержка на offTime миллисекунд
}

void motorOnThenOffwithSpeed(){
  int onSpeed = 200;
  // число между 0 (остановка) и 255 (максимальная скорость)
  int onTime = 2500;
  int offSpeed = 50;
  // число между 0 (остановка) и 255 (максимальная скорость)
  int offTime = 1000;
  analogwrite(motorPin, onSpeed);
  // включение мотора
  delay(onTime); // задержка на onTime миллисекунд
  analogwrite(motorPin, offSpeed);
  // выключение мотора
  delay(offTime); // задержка на offTime миллисекунд
}

void motorAcceleration(){
  int delayTime = 50; //задержка между скоростями
  for(int i = 0; i < 256; i++){
    //изменяет скорость от 0 до 255
    analogwrite(motorPin, i); //sets the new speed
    delay(delayTime); // задержка на delayTime миллисекунд
  }
  for(int i = 255; i >= 0; i--){
    //изменяет скорость от 255 до 0
    analogwrite(motorPin, i); //sets the new speed
    delay(delayTime); //задержка на delayTime миллисекунд
  }
}

```

**Не работает?** (3 проблемы и их решения)**Мотор не вращается?**

Если Вы используете свой собственный транзистор, убедитесь что расположение выводов совпадает с P2N222A. Многие транзисторы имеют другое расположение выводов.

**Все ещё не работает?**

Если Вы используете свой собственный мотор, убедитесь, что он может работать от напряжения 5В и не потребляет слишком много.

**Снова не работает?**

Иногда Arduino самопроизвольно отключается от компьютера. Попробуйте отключить и затем подключить обратно кабель USB Arduino.

**Усовершенствуем устройство****Контроль скорости:**

Ранее мы рассматривали возможность управления яркостью светодиодов при помощи Arduino. Этот же принцип можно использовать для управления скоростью вращения мотора. Arduino формирует сигналы, называемые ШИМ (широотно-импульсная модуляция). На выход с высокой частотой поступает последовательность логических нулей и единиц. Баланс между количеством единиц и нулей определяет результирующее напряжение. Например, для формирования напряжения 2,5 В необходимо чтобы за единицу времени количество нулей равнялось количеству единиц. В функции loop() измените текст программы следующим образом:

```

// motorOnThenOff();
  motorOnThenOffwithSpeed();
// motorAcceleration();

```

Загрузите программу в Arduino. Вы можете изменять скорость путем редактирования переменных onSpeed и offSpeed.

**Ускорение и замедление:**

Зачем останавливаться на двух фиксированных скоростях? С использованием этой технологии можно ускорять и замедлять мотор. Для проверки этого измените текст программы в функции loop() следующим образом:

```

// motorOnThenOff();
  // motorOnThenOffwithSpeed();
  motorAcceleration();

```

Загрузите программу и Вы увидите, как мотор плавно разгоняется до максимальной скорости, а затем плавно тормозит. Если Вы хотите поменять скорость разгона, то измените переменную delayTime (большее значение соответствует более медленному разгону).

**Есть еще вопросы?**

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC03>

# CIRC-04

## .:Одиночный сервопривод:.



### Описание задания:

Крутить мотор достаточно интересно, но когда мы делаем проект, в котором необходим контроль движения, хочется чего-то большего. Рассмотрим пример управления сервоприводом. Сервоприводы массово производятся, легко доступны, стоимость составляет от пары долларов до сотен. Внутри сервопривода встроен небольшой редуктор (чтобы увеличить мощность) и электроника (для упрощения управления). Стандартный сервопривод позиционируется от 0 до 180 градусов. Позиция задается длительностью управляющего импульса, от 1.25 мс (0 градусов) до 1.75 мс (180 градусов, 1.5 мс для 90 градусов). Временные параметры могут отличаться у различных производителей. Если посылать импульсы каждые 25...50 мс, то сервопривод может плавно вращаться. Одним из преимуществ Arduino является готовая к использованию библиотека подпрограмм, позволяющая легко управлять двумя сервоприводами (подключенными к портам 9 и 10).

### УСТРОЙСТВО:

#### Компоненты:



Карточка задания CIRC-04 x1



2-контактный разъем x4



3-контактный разъем x1

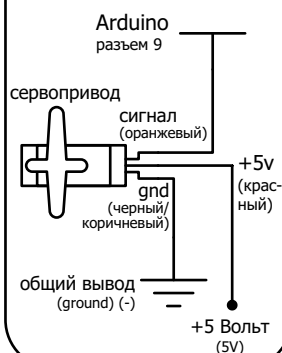


Провод



Сервопривод x1

#### Схема



#### в Интернете:

..скачать..

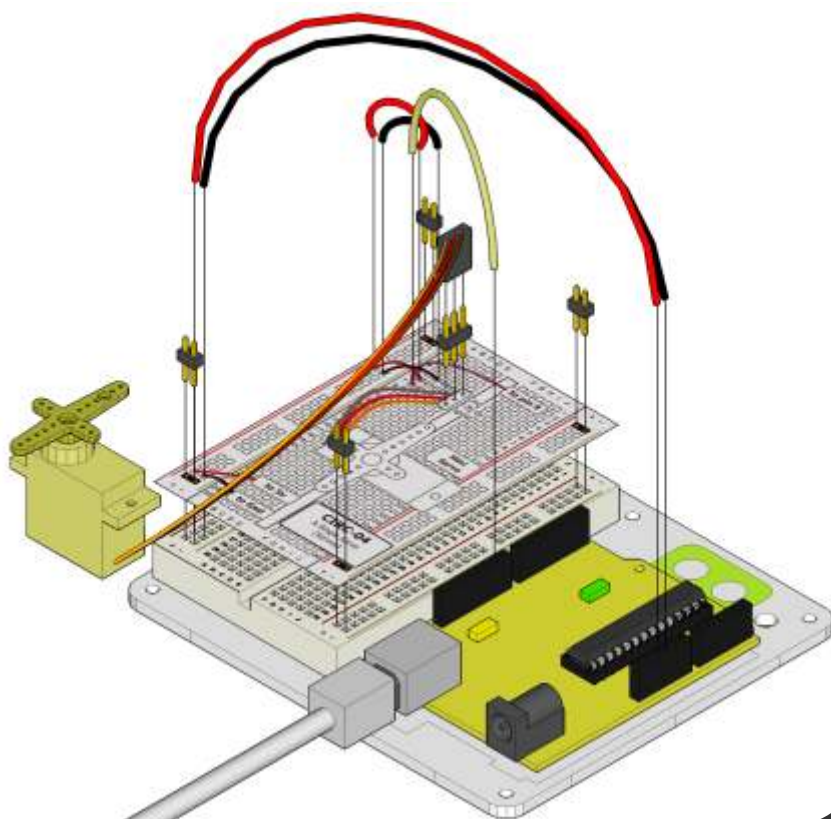
карточку задания

<http://ardx.org/BBLS04>

..посмотреть..

видео сборки

<http://ardx.org/VIDE04>





**Код** не надо набирать вручную, он находится по адресу:

**CIRC-04**

### File > Examples > Servo > Sweep

(это пример с сайта [arduino.cc](http://arduino.cc), также там можно найти много отличных идей)

// Поворот сервопривода (by BARRAGAN <<http://barraganstudio.com>>)

```
#include <Servo.h>
Servo myservo; // создание объекта "серво" для контроля сервопривода
int pos = 0; // переменная для хранения позиции серво

void setup() {
  myservo.attach(9); // назначает вывод 9 как управляющий для объекта серво
}

void loop() {
  for(pos = 0; pos < 180; pos += 1) // перемещается от 0° до 180° с шагом в 1°
    myservo.write(pos); // команда сервоприводу
    //перейти в позицию с переменной 'pos'
    delay(15); // задержка 15мс for the servo to reach the
    position
  }
  for(pos = 180; pos >= 1; pos -= 1) // перемещается от 180° до 0°
  {
    myservo.write(pos); //команда сервоприводу
    //перейти в позицию с переменной 'pos'
    delay(15); // задержка 15мс для достижения заданной позиции
  }
}
```

## Не работает? (3 проблемы и их решения)

### Сервопривод не вращается?

Даже с цветовой маркировкой проводов их по-прежнему легко можно перепутать. Возможно причина в этом.

### Все еще не работает?

Не забывайте про питание! Красный и коричневый проводники должны быть подключены к +5В и к «земле» соответственно.

### Если сервопривод двигается рывками

Если сервопривод дергается и светодиод питания на Arduino мигает, возможно мощность источника питания недостаточна. Попробуйте использовать питание от внешнего источника, а не от USB.

## Усовершенствуем устройство

### Управление потенциометром:

Мы еще не экспериментировали со входами, но если хотите попробовать, то загрузите пример **File > Servo >**

**Knob**. В этом примере для управления сервоприводом используется потенциометр (CIRC08). С пояснениями можно познакомиться по адресу: <http://ardx.org/KNOB>

### Прямое управление:

Управлять сервоприводом очень легко при помощи библиотеки подпрограмм. Но иногда необходимо самостоятельно создать программу для управления. Можно непосредственно управлять сервоприводом, если посылать импульс заданной длительности на любой порт Arduino. Образец программы:

```
int servoPin = 9;
void setup() {
  pinMode(servoPin, OUTPUT);
}
void loop() {
```

```
int pulseTime = 2100;
//величина задержки в микросекундах
//900 для 0 градусов, 1500 для 90 градусов,
//2100 для 180 градусов
digitalWrite(servoPin, HIGH);
delayMicroseconds(pulseTime);
digitalWrite(servoPin, LOW);
delay(25);
}
```

### Интересные идеи:

Сервопривод может использоваться во многих интересных проектах. Приведем несколько примеров:

Рождественский счетчик  
<http://ardx.org/XMAS>

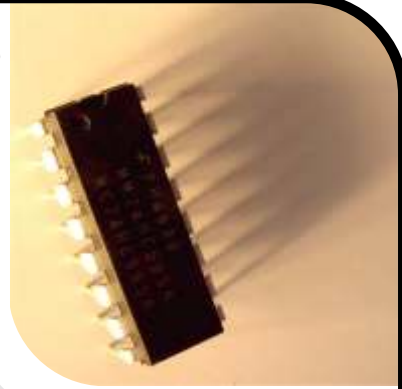
Манипулятор робота (открытый проект с использованием сервоприводов и Arduino)  
<http://ardx.org/RARM>

Шагающий робот с сервоприводами  
<http://ardx.org/SEWA>

## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC04>



### Описание задания:

Пришло время познакомиться с микросхемами. Внешний вид микросхемы может быть очень обманчивым. Например, микросхема контроллера на плате Arduino и микросхема сдвигового регистра выглядят очень похоже, хотя по сути очень сильно отличаются. Цена микроконтроллера ATmega составляет несколько долларов, в то время как цена сдвигового регистра 74HC595 — несколько центов. Освоение сдвигового регистра - хорошее начало, и если Вы разберетесь как им управлять и как работать с документацией (<http://ardx.org/74HC595>), то Вам уже будет не страшен мир микросхем. Сдвиговый регистр (также называется преобразователем последовательного интерфейса в параллельный) дает Вам 8 дополнительных цифровых выходов (для управления светодиодами или чем-то подобным) используя только три порта Arduino. Также эти регистры можно соединять последовательно, что позволяет организовать практически неограниченное число выходов, при использовании тех же трех портов Arduino. Для использования сдвигового регистра необходимо записать в него значение через последовательный интерфейс и затем выдать его на выход через параллельный интерфейс. Последовательный интерфейс представляет собой два входа: вход данных и вход тактов. Для передачи байта данных через такой интерфейс необходимо поочередно устанавливать на входе сдвигового регистра уровни, соответствующие значениям битов байта и посылать импульс на тактовый вход. После того как весь байт проходит в сдвиговый регистр, подается команда выдать этот байт через параллельный интерфейс. Более подробная документация содержится по адресу: <http://ardx.org/SHIF>.

### УСТРОЙСТВО:

#### Компоненты:



Карточка задания CIRC-05 x1



2-контактный разъем x4



Микросхема сдвигового регистра 74HC595 x1



Провод

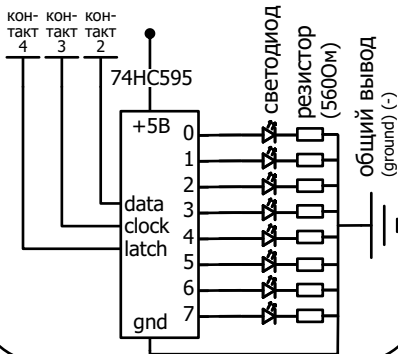


Красный светодиод x8



Резистор 560 Ом Зеленый-синий-коричневый x8

#### Схема



#### в Интернете:

.:скачать:.

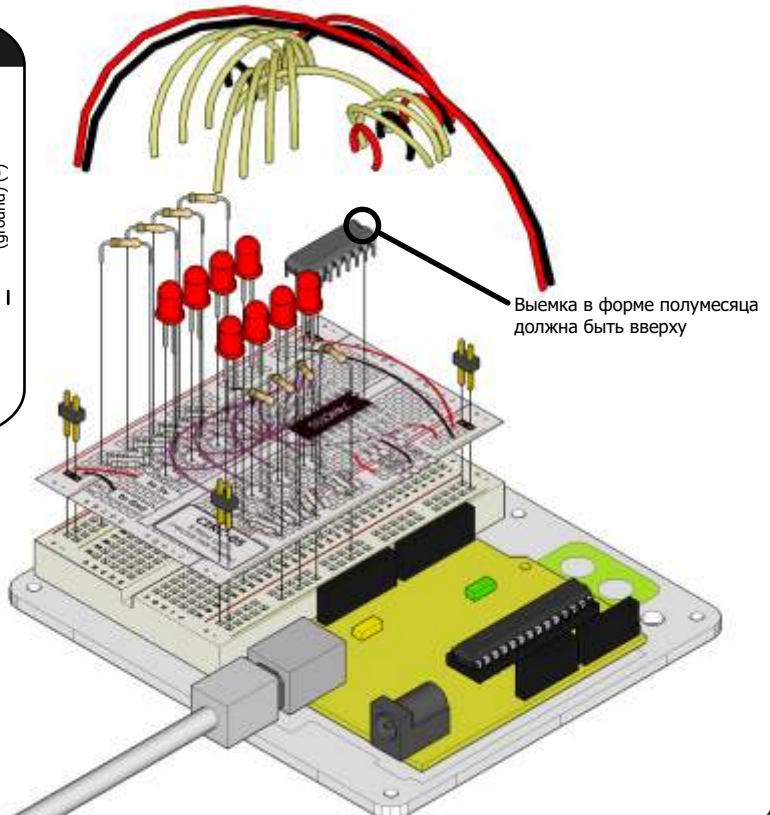
карточку задания

<http://ardx.org/BBLS05>

.:посмотреть:.

видео сборки

<http://ardx.org/VIDE05>



# Код

не надо набирать вручную,

# CIRC-05

его можно загрузить с <http://ardx.org/CODE05>

(скопируйте текст и вставьте его в новое окно Arduino Sketch)

```
Определение разъемов
//74HC595 использует протокол SPI
//у которого 3 разъема:
int data = 2;
int clock = 3;
int latch = 4;

void setup()
//функция выполняется один раз
{
  pinMode(data, OUTPUT);
  pinMode(clock, OUTPUT);
  pinMode(latch, OUTPUT); }

void loop()
// функция выполняется бесконечно
{
  int delayTime = 100;
  //задержка между обновлениями светодиодов
  for(int i = 0; i < 256; i++){
    updateLEDS(i);
    delay(delayTime); }
}

/*
 * updateLEDS() - посылает состояния
 светодиодов в виде значений переменной на
```

```
последовательность 74HC595
*/
void updateLEDS(int value){

  digitalWrite(latch, LOW);
  //сигнал защелки установлен на LOW

  shiftOut(data, clock, MSBFIRST, value);
  //выдает 8 бит в сдвиговый регистр

  digitalWrite(latch, HIGH);
  //открывает защелку - выводит информацию

  -----полная программа в электронной версии-----
```

## Не работает? (3 проблемы и их решения)

### Индикатор питания Arduino не горит

Возможно вы вставили микросхему сдвигового регистра наоборот. Если Вы быстро отключите питание и переустановите микросхему — ничего не должно испортиться.

### Работает неправильно?

Скорее всего, перепутаны какие-то провода. Внимательно проверьте подключение.

### Не получилось?

Напишите нам письмо с описанием проблемы. Постараемся помочь так быстро, насколько возможно. [help@oomlout.com](mailto:help@oomlout.com)

## Усовершенствуем устройство

### Усложним задачу:

Arduino может делать сложные вещи относительно легко. Выдача данных через последовательный интерфейс - одна из таких задач. Тем не менее эту задачу возможно решить и более сложным образом (на Ваш выбор). Попробуйте изменить текст программы следующим образом: updateLEDS(i); замените на updateLEDSlong(i); Загрузите программу в контроллер и Вы убедитесь, что ничего не изменилось. Посмотрите на текст функции updateLEDSlong(i); и Вы увидите, что передача данных идет по одному биту (подробнее на <http://ardx.org/SPI>).

### Независимое управление светодиодами:

Попробуем управлять светодиодами как в задании CIRC02. Состояние всех восьми светодиодов хранится при помощи одного байта (8 бит). Arduino может легко

оперировать с отдельными битами в байте. Подробное описание двоичных операций находится по адресу <http://ardx.org/BITW>.

```
Отредактируйте текст программы следующим образом:
int delayTime = 100;
//время задержки в мс между циклами
for(int i = 0; i < 8; i++){
  changeLED(i,ON);
  delay(delayTime);
}for(int i = 0; i < 8; i++){
  changeLED(i,OFF);
  delay(delayTime);
}
```

Загрузите программу в Arduino. Светодиоды должны загораться один за другим и так же гаснуть.

### Дополнительные эффекты:

Вы можете скопировать подпрограмму эффектов из программы CIRC02 и заменить команду digitalWrite(led,state) на changeLED(led,state).

## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC05>

## .:Музыка (пьезоэлемент):.



### Описание задания:

До сих пор мы управляли светом, движением и электронами. Давайте попробуем управлять звуком. Звук, как известно, аналоговое явление. Возможно ли создавать звуки при помощи цифрового Arduino?

Возможно, благодаря относительно высокой скорости работы микроконтроллера. Пьезоэлемент щелкает каждый раз, когда на него подается электрический импульс. Если посылать эти импульсы с определенной частотой (например 440 раз в секунду для воспроизведения ноты Ля), то щелчки превратятся в музыкальный тон. Давайте поэкспериментируем и заставим Arduino сыграть какую-нибудь мелодию.

### УСТРОЙСТВО:

#### Компоненты:



Карточка задания  
CIRC-06  
x1



2-контактный  
разъем  
x4

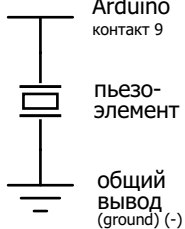


Пьезоэлемент  
x1



Провод

#### Схема



#### в Интернете:

.:скачать:.

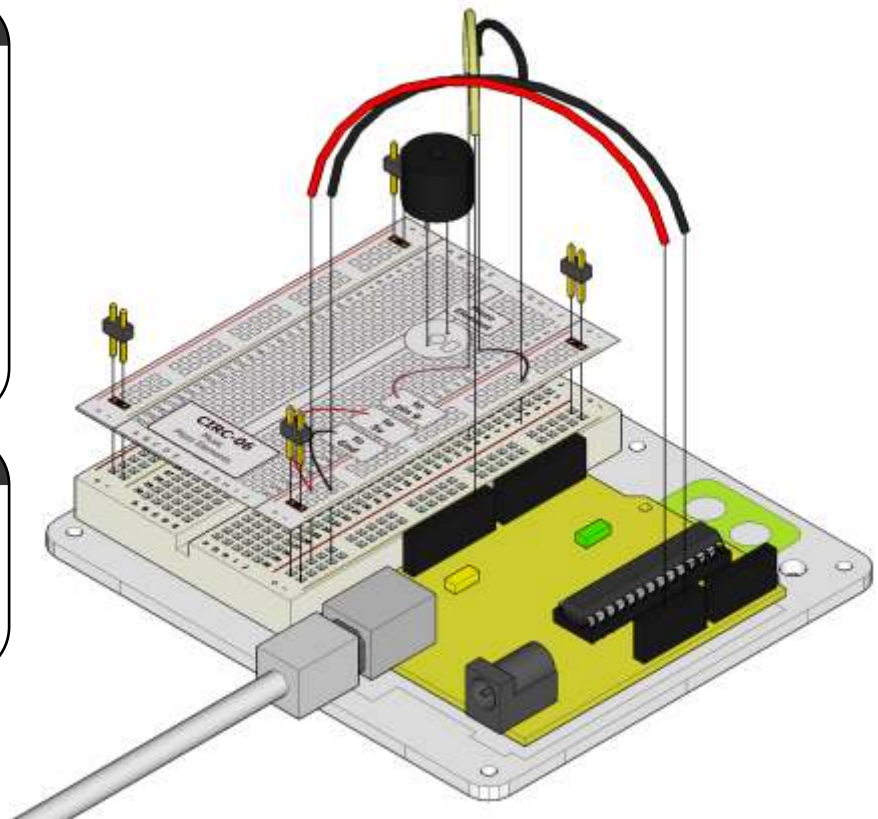
карточку задания

<http://ardx.org/BBLS06>

.:посмотреть:.

видео сборки

<http://ardx.org/VIDE06>



**его можно загрузить с <http://ardx.org/CODE06>**

(скопируйте текст и вставьте его в новое окно Arduino Sketch)

```

/* Мелодия
 * (cleft) 2005 D. Cuartielles for C3
 *
 * В этом примере мы проигрываем мелодию с помощью
 * пьезоэлемента. Он посылает ШИМ-сигнал соответствующей
 * частоты, в результате генерируется музыкальный тон.
 * Вычисление требуемой задержки осуществляется с помощью
 * следующей формулы:
 *
 *      timeHigh = period / 2 = 1 / (2 * toneFrequency)
 *
 * задержки задаются следующей таблицей:
 *
 * нота      частота (period)      timeHigh
 * c          261 Hz                3830      1915
 * d          294 Hz                3400      1700
 * e          329 Hz                3038      1519
 * f          349 Hz                2864      1432
 * g          392 Hz                2550      1275
 * a          440 Hz                2272      1136
 * b          493 Hz                2028      1014
 * c          523 Hz                1912      956
 *
 * http://www.arduino.cc/en/Tutorial/Melody
 */

int speakerPin = 9;
int length = 15; // число нот
char notes[] = "ccggaagffeeddc "; // пробел представляет паузу
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 4 };
int tempo = 300;

void playTone(int tone, int duration) {
  for (long i = 0; i < duration * 1000L; i += tone * 2) {
    digitalWrite(speakerPin, HIGH);
    delayMicroseconds(tone);
    digitalWrite(speakerPin,
LOW);
    delayMicroseconds(tone);
  }
}

void playNote(char note, int duration) {
  char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'c' };
  int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };
  // проигрывать тональность соответствующую названию ноты
  for (int i = 0; i < 8; i++) {
    if (names[i] == note) {
      playTone(tones[i], duration);
    }
  }
}

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

void loop() {
  for (int i = 0; i < length; i++) {
    if (notes[i] == ' ') {
      delay(beats[i] * tempo); // пауза
    } else {
      playNote(notes[i], beats[i] * tempo);
    }
    // пауза между нотами
    delay(tempo / 2);
  }
}

```

**Не работает?** (3 проблемы и их решения)**Нет звука**

Благодаря форме и размерам пьезоэлемента легко промахнуться мимо правильных контактов на макетной плате. Проверьте расположение пьезоэлемента.

**Не возможно думать, пока играет мелодия**

Просто отключите пьезоэлемент от макетной платы, подумайте, затем подключите его обратно.

**Устали от этой мелодии?**

Программа готова, поэтому Вам надо лишь заменить ноты на другие.

**Усовершенствуем устройство****Изменение скорости:**

Для изменения скорости воспроизведения мелодии необходимо редактировать строку

```
int tempo = 300; --->
int tempo = (новое значение)
```

Для замедления темпа необходимо увеличивать это число, для ускорения темпа — уменьшать.

**Подстройка нот:**

Если Вам кажется, что какие-либо ноты звучат фальшиво, вы можете их скорректировать. Для этого необходимо редактировать соответствующую переменную в массиве tones[].

```
char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'c' };
int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };
```

**Написание собственной мелодии:**

Программа играет "Twinkle Twinkle Little Star", но мелодию легко изменить. Каждая мелодия определяется одной переменной (число нот - int length) и двумя массивами. Один массив, notes[] задает последовательность нот, другой, beats[] — длительность их звучания. Например:

Twinkle Twinkle Little Star

```
int length = 15;
char notes[] = {"ccggaagffeeddc "};
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 4 };
```

Happy Birthday (first line)

```
int length = 13;
char notes[] = {"ccdcfecdcg "};
int beats[] = {1,1,1,1,1,2,1,1,1,1,1,2,4};
```

**Есть еще вопросы?**

Подробности, где купить детали к проекту, где задать вопросы:

**<http://ardx.org/CIRC06>**



### Описание задания:

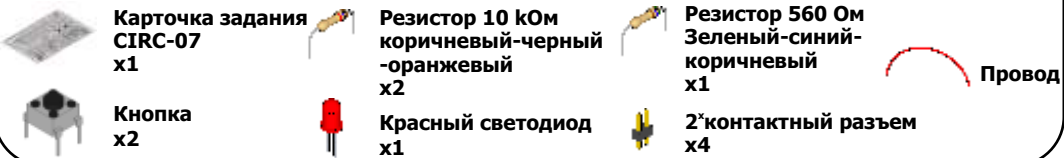
До этого момента мы рассматривали различные варианты управления. Настало время научить Arduino слышать, смотреть и чувствовать. Начнем с простой кнопки.

Подключить кнопку очень просто. Необходим только один компонент, подтягивающий резистор. Он необходим потому, что Arduino получает информацию не так как мы. Он не чувствует нажата кнопка или нет, вместо этого он контролирует значение напряжения на своем входе. Нажатие на кнопку приводит к обнулению напряжения (LOW) на входе, когда кнопка отжата, напряжение становится высоким (HIGH). Резистор необходим, чтобы на входе Arduino уверенно формировалась логическая единица когда кнопка отжата.

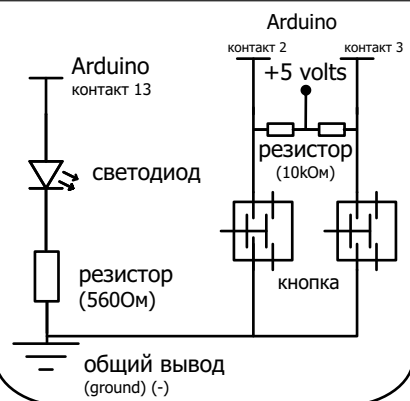
**Обратите внимание:** в первом примере используется только одна кнопка.

### УСТРОЙСТВО:

#### Компоненты:

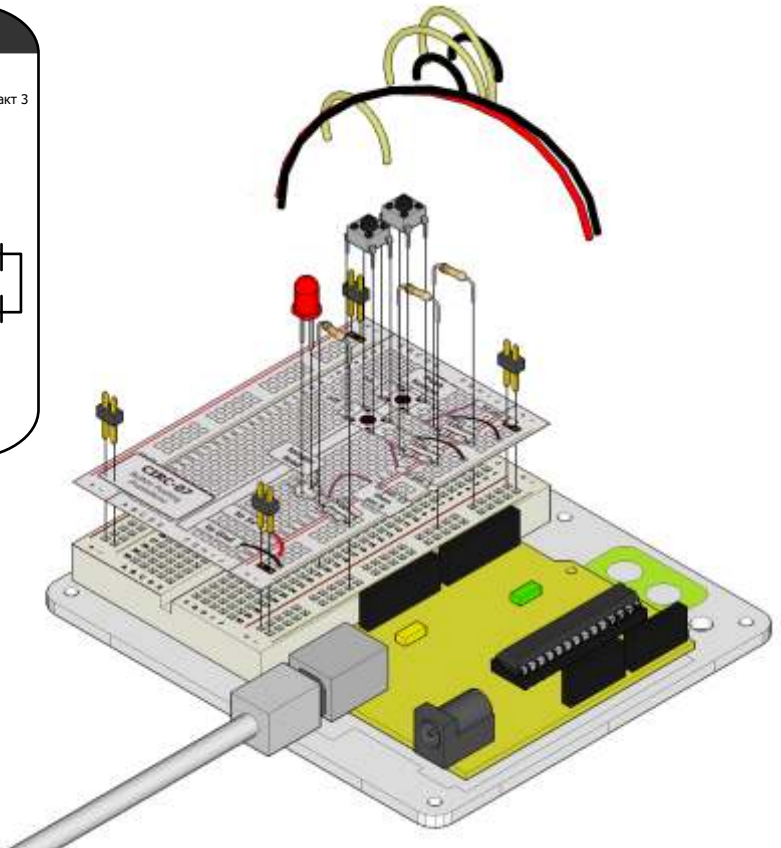


#### Схема



#### в Интернете:

.:скачать.:  
карточку задания  
<http://ardx.org/BBL507>  
.:посмотреть.:  
видео сборки  
<http://ardx.org/VIDE07>



# Код

не надо набирать вручную, он находится по адресу:

# CIRC-07

## File > Examples > 2.Digital > Button

(это пример с сайта arduino.cc, также там можно найти много отличных идей)

```
/*
 * Кнопка
 * by DojoDave <http://www.0j0.org>
 *
 * Включаем и выключаем светодиод нажатием кнопки.
 * http://www.arduino.cc/en/Tutorial/Button
 */
int ledPin = 13;           // задаем номер выхода светодиода
int inputPin = 2;         // задаем номер входа, подключенный к кнопке
int val = 0;              // переменная для хранения состояния кнопки

void setup() {
  pinMode(ledPin, OUTPUT);
  // инициализируем контакт, подключенный к светодиоду, как выход
  pinMode(inputPin, INPUT);
  // инициализируем контакт, подключенный к кнопке, как вход
}

void loop(){
  val = digitalRead(inputPin); // считываем значения с входа кнопки
  if (val == HIGH) {           // проверяем нажата ли кнопка
    digitalWrite(ledPin, LOW); // выключение светодиода
  } else {
    digitalWrite(ledPin, HIGH); // включение светодиода
  }
}
```

## Не работает? (3 проблемы и их решения)

### Светодиод не включается

Кнопка квадратная, поэтому ее легко вставить повернув на 90 градусов. Проверьте правильность подключения кнопки.

### Светодиод не загорается плавно (тухнет)

Не забудьте переключить светодиод с порта 13 на порт 9 для этого задания.

### Разочарованы?

Не беспокойтесь, все устройства в этом наборе прощены до предела, чтобы было проще понять основы. Когда Вы разберетесь в основах — перед вами откроются неограниченные возможности.

## Усовершенствуем устройство

### Кнопка включения и кнопка выключения:

Первый пример может немного разочаровать (например, «мне не нужен Arduino, чтобы сделать это»), давайте его немного усложним. Одна кнопка будет включать светодиод, а другая выключать. Измените программу следующим образом:

```
int ledPin = 13;
// выберите разъем для светодиода
int inputPin1 = 3; // кнопка 1
int inputPin2 = 2; // кнопка 2

void setup() {
  pinMode(ledPin, OUTPUT);
  // светодиод задается как выход
  pinMode(inputPin1, INPUT); // кнопка 1 - вход
  pinMode(inputPin2, INPUT); // кнопка 2 - вход
}

void loop(){
  if (digitalRead(inputPin1) == LOW) {
    digitalWrite(ledPin, LOW); // выключение светодиода
  } else if (digitalRead(inputPin2) == LOW) {
    digitalWrite(ledPin, HIGH); // включение светодиода
  }
}
Загрузите программу и проверьте работоспособность.
```

### Плавное включение/выключение светодиода:

Давайте используем кнопку для управления аналоговыми сигналами. Для этого необходимо переключить светодиод с порта 13 на порт 9 и изменить программу:

```
int ledPin = 13; ----> int ledPin = 9;
Теперь измените функцию loop() следующим образом:
int value = 0;
void loop(){
  if (digitalRead(inputPin1) == LOW) { value--; }
  else if (digitalRead(inputPin2) == LOW) { value++; }
  value = constrain(value, 0, 255);
  analogWrite(ledPin, value);
  delay(10);
}
```

### Изменение скорости включения/выключения светодиода:

Если Вы хотите, чтобы светодиод загорался или гас быстрее, необходимо изменить только одну команду в программе:

```
delay(10); ----> delay(новое значение);
```

Для ускорения эффекта необходимо уменьшать это число. Для замедления включения/выключения это число необходимо увеличить.

## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC07>

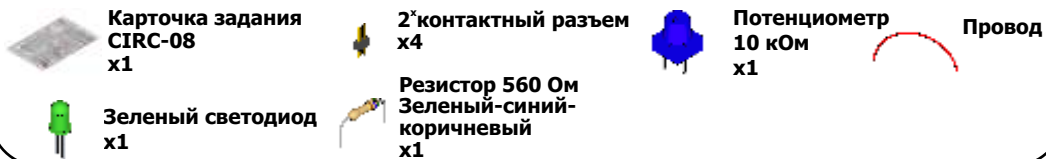


### Описание задания:

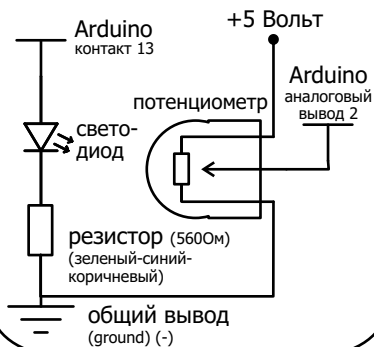
Кроме цифровых портов Arduino оборудован шестью портами, которые могут быть аналоговыми входами. При помощи такого входа можно преобразовать напряжение в диапазоне 0...5В в цифровой код с диапазоном 0...1023 (соответствует разрешению 10 бит). Потенциометр (переменный резистор) это очень полезное устройство, которое можно использовать совместно с таким входом. Когда потенциометр подключен крайними выводами к «земле» и к +5В, на среднем выводе будет присутствовать некое напряжение из диапазона 0...5В, зависящее от положения движка потенциометра (угла поворота). В среднем положении значение на среднем выводе будет соответствовать 2.5В. Можно использовать измеренное значение напряжения на среднем выводе потенциометра в качестве переменной в программе.

### УСТРОЙСТВО:

#### Компоненты:

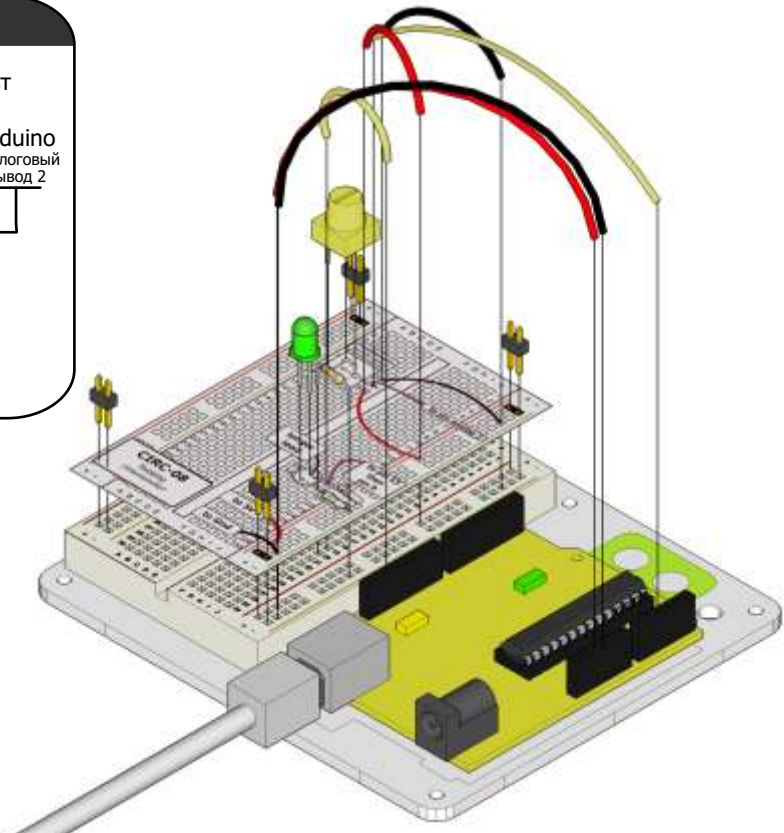


#### Схема



#### в Интернете:

**::скачать::**  
карточку задания  
<http://ardx.org/BBL08>  
**::посмотреть::**  
видео сборки  
<http://ardx.org/VIDE08>





## File > Examples > 3.Analog > AnalogInput

(это пример с сайта [arduino.cc](http://arduino.cc), также там можно найти много отличных идей)

```
/*
 * Демонстрирует аналоговый вход, считывая значение аналогового сенсора со входа 0 и включая и
 * выключая светодиод, подключенный к цифровому выводу 13. Время пребывания светодиода во
 * включенном или выключенном состоянии зависит от значения, полученного при помощи analogRead().
 * Created by David Cuartielles
 * Modified 16 Jun 2009
 * By Tom Igoe
 * http://arduino.cc/en/Tutorial/AnalogInput
 */

int sensorPin = 0; // задаем номер входа для потенциометра
int ledPin = 13; // задаем номер выхода светодиода
int sensorValue = 0; // переменная для хранения значения, поступающего с сенсора

void setup() {
  pinMode(ledPin, OUTPUT); //инициализируем контакт, подключенный к светодиоду, как выход
}

void loop() {
  sensorValue = analogRead(sensorPin); // считываем значения с сенсора
  digitalWrite(ledPin, HIGH); // включает светодиод
  delay(sensorValue); // задержка на <sensorValue> в миллисекундах
  digitalWrite(ledPin, LOW); // выключает светодиод
  delay(sensorValue); // задержка на <sensorValue> в миллисекундах
}
```

## Не работает? (3 проблемы и их решения)

### Потенциометр периодически перестает работать

Это возможно из-за плохого контакта выводов потенциометра. Попробуйте прижать потенциометр плотнее к макетной плате.

### Не работает

Убедитесь, что не подключили средний вывод потенциометра к цифровому входу 2 вместо аналогового входа 2 (аналоговые входы находятся в одном ряду с разъемом питания).

### Все еще не работает

Попробуйте изменить полярность подключения потенциометра. Иногда это помогает.

## Усовершенствуем устройство

### Пороговый переключатель:

Иногда необходимо переключать что-то при достижении измеренным параметром заданной величины (например отключать подачу воды в стиральную машину, при достижении заданного уровня). Чтобы сделать это с потенциометром необходимо изменить программу следующим образом:

```
void loop() {
  int threshold = 512;
  if(analogRead(sensorPin) > threshold){
    digitalWrite(ledPin, HIGH);}
  else{ digitalWrite(ledPin, LOW);}
}
```

Светодиод будет включаться/выключаться при достижении среднего значения. Вы можете изменить чувствительность, путем редактирования параметра threshold.

### Контроль яркости:

Можно управлять яркостью светодиода непосредственно при помощи потенциометра. Для этого необходимо переключить светодиод с порта 13 на порт 9 и изменить

текст программы:

```
int ledPin = 13; ----> int ledPin = 9;
Измените функцию loop на:
void loop() {
  int value = analogRead(potPin) / 4;
  analogWrite(ledPin, value);
}
```

Загрузите программу в микроконтроллер и убедитесь, что яркость светодиода зависит от положения движка потенциометра. Мы разделили измеренное значение на 4, потому что яркость может изменяться только в пределах 0...255 (8 бит), тогда как функция `analogRead()` - в пределах 0...1024 (10 бит).

### Контроль сервопривода:

Это интересный пример, объединяющий два устройства в одном. Подключите сервопривод как в задании CIRC-04, откройте пример **File>Examples>Servo>Knob**, измените строку:

```
int potpin = 0; ----> int potpin = 2;
Загрузите программу в микроконтроллер и убедитесь, что положение потенциометра определяет положение сервопривода.
```

## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC08>










### Описание задания:

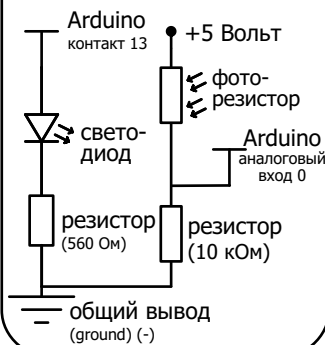
В предыдущем задании мы рассмотрели потенциометр, который может использоваться для ввода информации человеком. Существуют разнообразные сенсоры, позволяющие получить информацию об окружающей обстановке. Например фоторезистор может использоваться в качестве датчика освещенности. Принцип работы с Arduino не изменяется. Arduino не может непосредственно измерять сопротивление (можно измерять напряжение), поэтому фоторезистор включается как часть делителя напряжения (<http://ardx.org/VODI>). Можно рассчитать точное значение напряжения на аналоговом входе, но для нашей задачи это не нужно. Будем измерять относительный уровень освещенности. Малые значения будут соответствовать яркому свету, большие значения — темноте.

### УСТРОЙСТВО:

#### Компоненты:

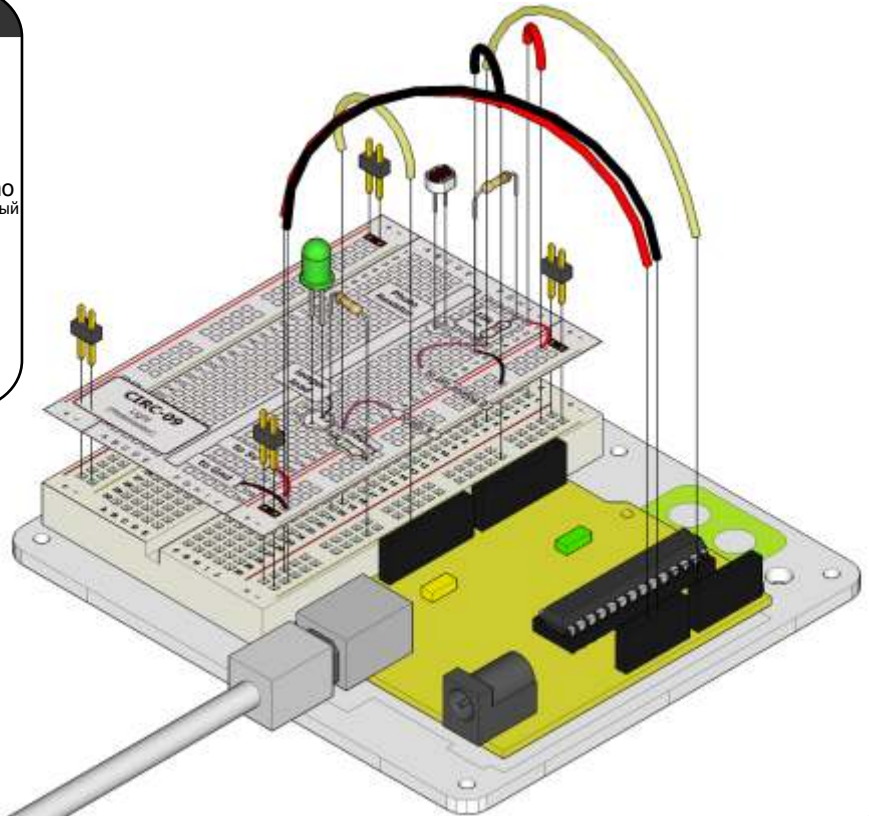
- |  |   |  |  |
|--|---|--|--|
|  Карточка задания CIRC-09 x1                     |  2-контактный разъем x4                       |  Фоторезистор x1       |  Провод |
|  Резистор 10 кОм коричневый-черный-оранжевый x2 |  Резистор 560 Ом Зеленый-синий-коричневый x1 |  Зеленый светодиод x1 |  |

#### Схема



#### в Интернете:

..скачать..  
карточку задания  
<http://ardx.org/BBLS09>  
..посмотреть..  
видео сборки  
<http://ardx.org/VIDE09>



его можно загрузить с <http://ardx.org/CODE09>

(скопируйте текст и вставьте его в новое окно Arduino Sketch)

```

/*
 * Простая программа, изменяющая интенсивность
 * свечения светодиода на основе количества
 * света, падающего на фоторезистор.
 */

//PhotoResistor Pin
int lightPin = 0; //аналоговый вход к которому
// присоединен фоторезистор.
// фоторезистор не откалиброван по каким-либо
// единицам измерения, а просто выдает напряжение
// пропорциональное уровню освещенности

value (relative light)
//LED Pin
int ledPin = 9;
//контакт к которому подключен светодиод.
//мы контролируем яркость, поэтому используем
аналоговый выход ШИМ (Широтно-Импульсная
модуляция)

void setup()
{
    pinMode(ledPin, OUTPUT);
    //инициализируем //контакт, подключенный к
    //светодиоду, как выход
}
/*
 * функция loop() начнется после окончания функции
 * setup и будет повторяться
 */
void loop()
{
    int lightLevel = analogRead(lightPin);
    //считываем уровень освещенности

    lightLevel = map(lightLevel, 0, 900, 0, 255);
    //выставляем значение переменной в пределах 0-900
    lightLevel = constrain(lightLevel, 0, 255);
    //ограничиваем значение переменной промежутком 0-255
    analogWrite(ledPin, lightLevel);
    //выводим значение переменной
}

```

## Не работает? (3 проблемы и их решения)

### Светодиод остается темным

Очень часто светодиод подключается с несоблюдением полярности. Попробуйте повернуть его на 180 градусов.

### Устройство не реагирует на изменение освещенности

Проверьте правильно ли подключен фоторезистор.

### Все еще не работает?

Возможно в комнате, где Вы проводите эксперимент, слишком темно или наоборот слишком светло. Попробуйте включить/выключить свет — возможно это поможет.

Если у вас есть под рукой фонарь — попробуйте осветить им датчик.

## Усовершенствуем устройство

### Инвертирование выхода:

Возможно Вы хотите, чтобы светодиод работал в инверсном режиме. Для этого отредактируйте программу следующим образом:  
`analogWrite(ledPin, lightLevel); ---->`  
`analogWrite(ledPin, 255 - lightLevel);`

### Ночник:

Вместо того, чтобы изменять яркость светодиода в зависимости от освещенности, можно просто включать и выключать его. Замените текст подпрограммы loop() следующим образом:

```

void loop(){
    int threshold = 300;
    if(analogRead(lightPin) > threshold){
        digitalWrite(ledPin, HIGH);
    }else{
        digitalWrite(ledPin, LOW);
    }
}

```

### Сервопривод, управляемый светом:

Давайте используем этот сенсор для управления сервоприводом. Подключите сервопривод к порту 9 (как в CIRC-04). Загрузите в контроллер программу Knob File > Examples > Servo > Knob (пример из CIRC-08). Посмотрите как будет работать сервопривод.

Вы наверняка обратили внимание, что сервопривод работает не во всем диапазоне. Это происходит потому, что благодаря делителю напряжения мы работаем в меньшем диапазоне, чем 0...5В. Это можно скорректировать при помощи команды `val = map(val, 0.1023, 0, 179)`; Подробное объяснение этой функции можно найти по адресу <http://arduino.cc/en/Reference/Map>.

## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC09>

# CIRC-10

## .:Температура (температурный датчик TMP36):.



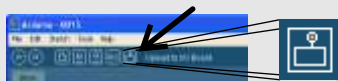
### Описание задания:

Что еще можно измерить при помощи Arduino? Температуру!

Для измерения температуры будем использовать достаточно сложную микросхему, спрятанную в корпусе аналогичном корпусу транзистора P2N222AG. Эта микросхема имеет три вывода: «земля», сигнал и питание +5В. Она выдает 10мВ/градус

Цельсия (чтобы измерять отрицательные температуры предусмотрен сдвиг напряжения на 500 мВ, т. е. 25°C = 750 мВ, 0°C = 500 мВ). Чтобы преобразовать значение напряжения в температуру будем использовать математические способности Arduino. Для отображения температуры используем отладочное окно интерфейса Arduino. Будем посылать значения температуры через последовательный порт для отображения на экране компьютера.

Это устройство использует монитор последовательного порта. Чтобы запустить его необходимо загрузить программу в Arduino, затем нажать кнопку похожую на прямоугольник с антенной:



Документация по датчику TMP36: <http://ardx.org/TMP36>

### УСТРОЙСТВО:

#### Компоненты:



Карточка задания  
CIRC-10  
x1



2-контактный  
разъем  
x4

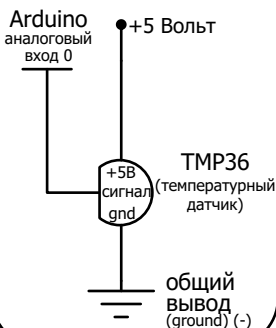


Температурный  
датчик TMP36  
x1



Провод

#### Схема



#### в Интернете:

..скачать..

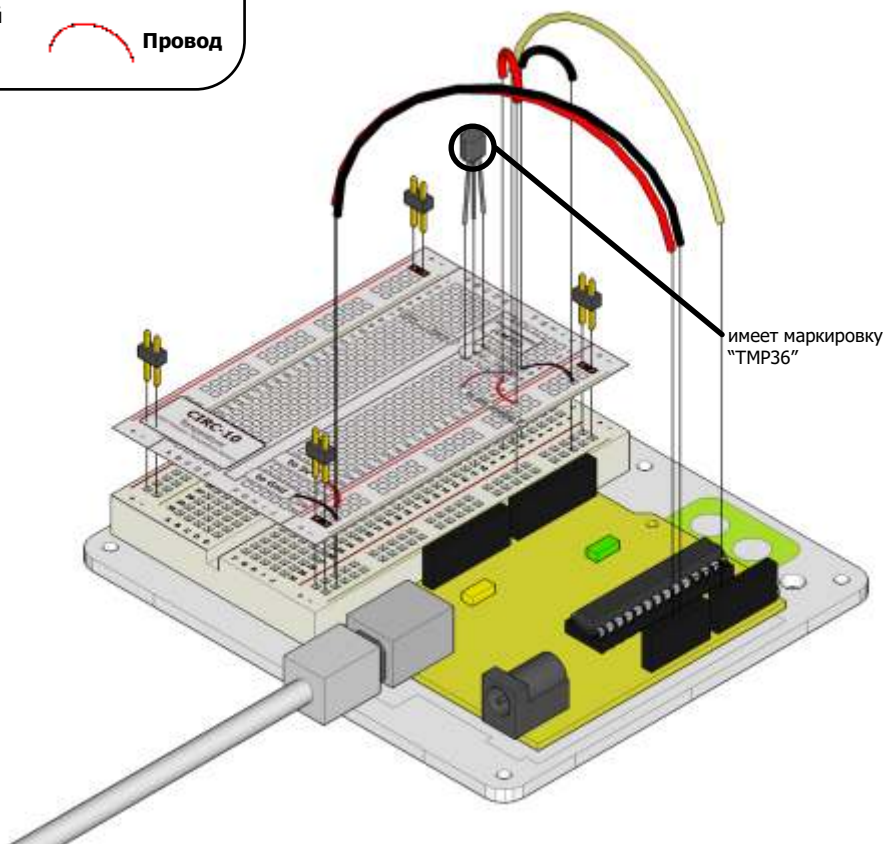
карточку задания

<http://ardx.org/BBL510>

..посмотреть..

видео сборки

<http://ardx.org/VIDE10>



## Код не надо набирать вручную,

его можно загрузить с <http://ardx.org/CODE10>

(скопируйте текст и вставьте его в новое окно Arduino Sketch)

```
/*
 * | Экспериментальный набор Arduino |
 * | пример программы для задания CIRC-10 |
 * | .: Температурный датчик :. |
 *
 * Простая программа для вывода текущей температуры
 * на монитор последовательного порта
 */

//TMP36 Pin variables
int temperaturePin = 0;
// задаем аналоговый вывод, к которому присоединен
// выход датчика TMP36. Разрешение - 10мВ/градус цельсия
// (смещение на 500мВ для отображения негативных
// температур)

void setup()
{
  Serial.begin(9600);
  //открываем последовательное соединение. Для
  //отображения результата необходимо запустить монитор
  //последовательного порта (последняя кнопка под главным
  //меню, выглядит как прямоугольник с антенной)
}

void loop() // бесконечный цикл
{
  float temperature = getVoltage(temperaturePin);
  /*получает значение напряжения с температурного
  датчика*/

  temperature = (temperature - .5)
  * 100;
  //переводит полученное значение из 10мВ/градус в градусы
  // (напряжение - 500мВ)х100

  Serial.println(temperature); //выводим результат
  delay(1000); //задержка 1 сек
}

/*
 * getVoltage() - возвращает значение напряжения на
 * аналоговом входе
 */
float getVoltage(int pin){
  return (analogRead(pin) * .004882814);
  //преобразует значение в цифровом диапазоне 0...1024 в
  //диапазон 0...5 мВ (каждое показание ~ 5 милливольт)
}
```

## CIRC-10

### Не работает? (3 проблемы и их решения)

#### Ничего не происходит

Эта программа не использует индикацию. Для того чтобы увидеть результат, Вам необходимо запустить монитор последовательного порта.

#### Отображаются странные данные

Скорее всего это происходит из-за того, что Ваш монитор последовательного порта принимает данные не с той скоростью. Переключите скорость работы монитора на «9600 baud».

#### Значение температуры не изменяется

Прикоснитесь к сенсору пальцем, чтобы нагреть его. Или используйте кусочек льда, чтобы охладить сенсор.

### Усовершенствуем устройство

#### Отображение напряжения:

Это очень просто сделать, поскольку сенсор уже выдает напряжение. Достаточно удалить строку  
delete the line `temperature = (temperature - .5) * 100;`

#### Отображение температуры в Фаренгейтах:

Это тоже несложно, поскольку затрагивает только вычислительную часть. Чтобы перейти от градусов Цельсия к градусам Фаренгейта необходимо использовать следующую формулу:

$$(F = C * 1.8) + 32$$

добавьте строку

```
temperature =
(((temperature - .5) * 100)*1.8) + 32;
перед строкой Serial.println(temperature);
```

#### Более информативный вывод данных:

Давайте добавим сообщение к данным, посылаемым через последовательный порт. Вернитесь к исходной программе, затем замените:  
`Serial.println(temperature);`

```
----->
Serial.print(temperature);
Serial.println(" degrees centigrade");
```

Изменения в первой строке означают, что следующий вывод данных будет осуществляться в ту же строку на дисплее. Следующая строка выводит вспомогательную информацию и осуществляет переход на новую строку.

#### Изменение скорости передачи:

Если Вы собираетесь посылать большой объем информации через последовательный порт, то имеет смысл делать это так быстро, насколько возможно. В нашем примере мы передаем данные на скорости 9600 бод, однако возможно использовать намного большие скорости. Измените строку:  
`Serial.begin(9600);` -----> `Serial.begin(115200);`

Таким образом, скорость передачи увеличивается в 12 раз. Не забудьте изменить скорость работы монитора последовательного порта.

### Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC10>












### Описание задания:

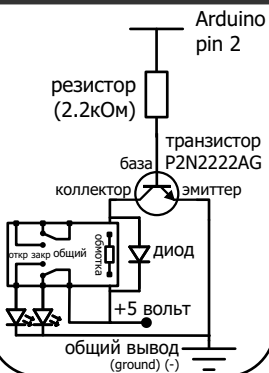
Последние задания всегда в какой-то степени являются испытанием. Мы используем наши знания о транзисторах (CIRC03) чтобы управлять реле. Реле - это управляемый электронно-механический переключатель. Внутри пластиковой коробочки содержится электромагнит, при подаче питания на который происходит переключение механического контакта (обычно с заметным щелчком). Различные реле очень сильно отличаются по размерам и параметрам (от совсем крошечных, до реле размером с холодильник). Обычно, чем больше размер реле — тем больший ток оно способно коммутировать. При помощи реле и Arduino Вы сможете управлять чем-то значительным.

### УСТРОЙСТВО:

#### Компоненты:

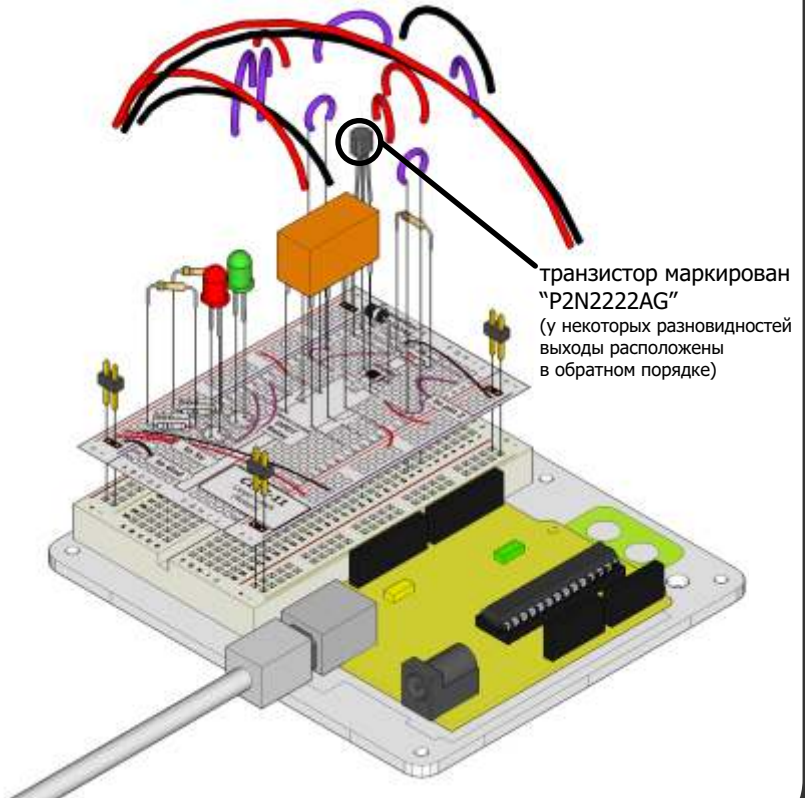
- |   |   |   |   |
|---|---|---|---|
|  Карточка задания CIRC-11 x1                  |  Диод (1N4001) x1                             |  Транзистор P2N2222AG (TO92) x1 |  Реле (DPDT) x1        |
|  Резистор 2.2 кОм красный-красный-красный x1 |  Резистор 560 Ом Зеленый-синий-коричневый x2 |  Зеленый светодиод x1          |  Красный светодиод x1 |
|  2-контактный разъем x4                      |   |   |   |

#### Схема



#### в Интернете:

..скачать..  
карточку задания  
<http://ardx.org/BBL11>  
..посмотреть..  
видео сборки  
<http://ardx.org/VIDE11>



**Код** не надо набирать вручную, он находится по адресу:

**CIRC-11**

## File > Sketchbook > 1.Basic > Blink

(это пример с сайта arduino.cc, также там можно найти много отличных идей)

```
/* Мигание
```

```
*/  
* Простой пример: программа включает светодиод на 1 секунду, потом выключает на 1  
секунду, и тд.. Мы используем вывод 13, потому что в зависимости от того, какой у вас  
Ардуино, у него есть либо встроенный светодиод, либо резистор и тогда вам нужен  
только светодиод.
```

```
*/  
* http://www.arduino.cc/en/Tutorial/Blink  
*/
```

```
int ledPin = 2; // ***** CHANGE TO PIN 2 *****  
  
void setup() // выполняется один раз, потом запускается скетч  
{  
  pinMode(ledPin, OUTPUT); // инициализируем цифровой контакт как выход  
}  
  
void loop() // функция выполняется бесконечно  
{  
  digitalWrite(ledPin, HIGH); // включает светодиод  
  delay(1000); // задержка 1 сек  
  digitalWrite(ledPin, LOW); // выключает светодиод  
  delay(1000); // задержка 1 сек  
}
```

## Не работает? (3 проблемы и их решения)

### Ничего не происходит

Программа использует порт 13, а наше реле подключено к порту 2. Не забудьте соответственно соответственно отредактировать программу.

### Нет щелчков (звуков переключения реле)

Возможно не работает транзистор. Тщательно проверьте правильность установки.

### Не совсем работает

Включенное в набор реле рассчитано на пайку, а не на установку в макетную плату. Возможно поможет если Вы прижмете его к макетной плате с небольшим усилием.

## Усовершенствуем устройство

### Визуализация импульсов обратного тока

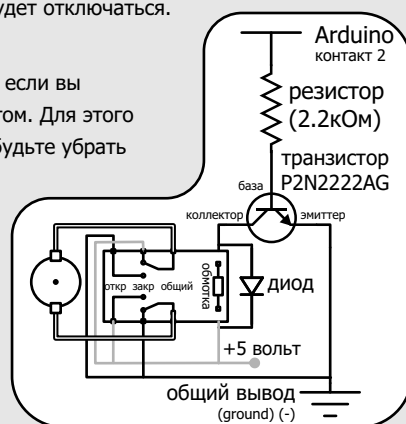
Замените диод светодиодом. Вы увидите мигание каждый раз, когда реле будет отключаться.

### Управление мотором

В задании CIRC03 мы управляли мотором при помощи транзистора. Однако, если вы собираетесь управлять более мощным мотором, реле будет лучшим вариантом. Для этого просто снимите красный светодиод и вместо него подключите мотор (не забудьте убрать резистор 560 Ом).

### Изменение направления вращения мотора

Немного усложним устройство напоследок. Для изменения направления вращения мотора постоянного тока необходимо изменить направление протекания тока. Если это делать вручную — достаточно поменять местами провода. Для электронной коммутации необходимо использовать что-то вроде H-моста. Это можно сделать при помощи DPDT реле. Соберите следующую схему:



## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://ardx.org/CIRC11>

## ..:Многоцветное свечение (светодиоды RGB):.



### Описание задания:

Вы можете мигать светодиодами и использовать ШИМ для управления моторами. Давайте применим эти знания и создадим светодиод, который может светиться любым цветом и с любой интенсивностью (на основе RGB светодиода).

RGB светодиод представляет собой три светодиода — R красный, G зеленый и B синий в одном корпусе. Используя различные комбинации этих трех цветов можно синтезировать практически любой цвет.

Для контроля каждого канала RGB светодиода используется обычный светодиод, таким образом смешение цветов происходит наглядно.

### УСТРОЙСТВО:

#### Компоненты:



Карточка задания CIRC-RGB x1



2-контактный разъем x4



Светодиод RGB 5мм x1



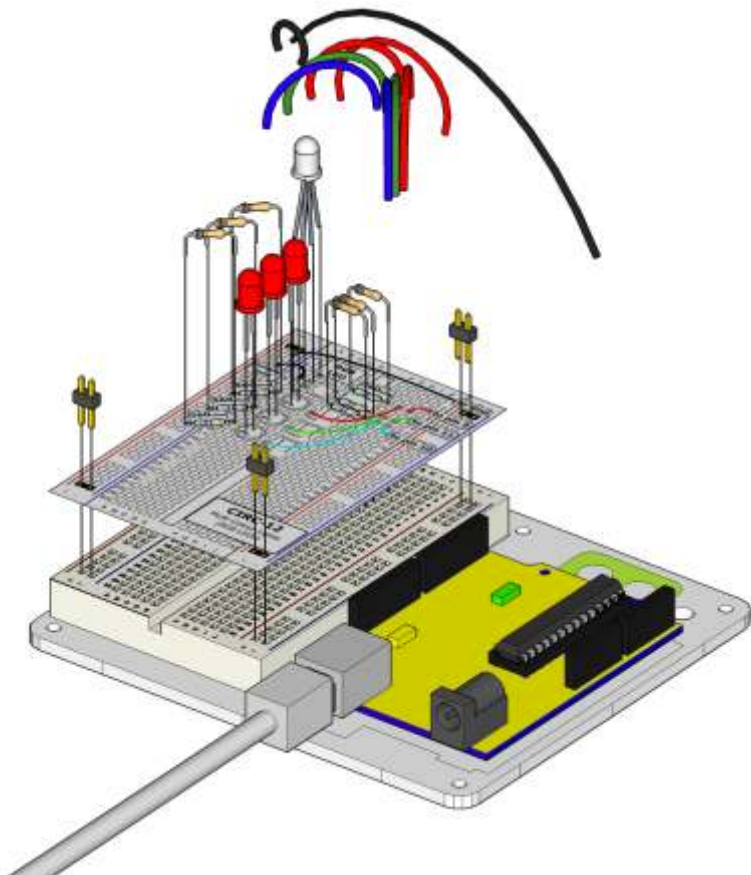
Провод x1



Резистор 560 Ом Зеленый-синий-коричневый x6



Красный светодиод x1



#### в Интернете:

..:скачать..  
карточку задания  
<http://ardx.org/BBS12R>



его можно загрузить с <http://ardx.org/CODE12R>

(скопируйте текст и вставьте его в новое окно Arduino Sketch)

```
//RGB LED pins
int ledDigitalOne[] = {9, 10, 11};
//3 цифровых вывода светодиода
//9 = redPin, 10 = greenPin, 11 = bluePin

const boolean ON = LOW;
//задаем включенное состояние как LOW
// (потому что мы используем RGB светодиод с
// общим анодом) общий вывод подключен к +5В
const boolean OFF = HIGH;
//задаем выключенное как HIGH

//Заданные цвета
const boolean RED[] = {ON, OFF, OFF};
const boolean GREEN[] = {OFF, ON, OFF};
const boolean BLUE[] = {OFF, OFF, ON};
const boolean YELLOW[] = {ON, ON, OFF};
const boolean CYAN[] = {OFF, ON, ON};
const boolean MAGENTA[] = {ON, OFF, ON};
const boolean WHITE[] = {ON, ON, ON};
const boolean BLACK[] = {OFF, OFF, OFF};

//массив хранящий заданные цвета
const boolean* COLORS[] =
  {RED, GREEN, BLUE, YELLOW, CYAN, MAGENTA,
  WHITE, BLACK};

void setup(){

  for(int i = 0; i < 3; i++){
    pinMode(ledDigitalOne[i], OUTPUT);
  }
  //инициализирует выходы 3х светодиодов как выходы
}

void loop(){
  setColor(ledDigitalOne, CYAN);
  //задает цвет светодиода

  //randomColor()
}

void randomColor(){
  int rand = random(0, sizeof(COLORS) / 2);
  //возвращает случайное число из диапазона цветов
  setColor(ledDigitalOne, COLORS[rand]);
  //задает цвет светодиода как случайный
  delay(1000);
}

void setColor(int* led, boolean* color){
  for(int i = 0; i < 3; i++){
    digitalWrite(led[i], color[i]);
  }
}
}
```

## Не работает? (3 проблемы и их решения)

### Светодиод не загорается, или показывает неправильные цвета

Проверьте внимательно подключение RGB светодиода. Не забудьте про подключение общего вывода к «земле».

### Все слишком красное

Возможно красный светодиод в RGB светодиоде светится ярче остальных. Это можно скорректировать путем увеличения резистора в канале R.

### Слишком много светодиодов?

Одноцветные светодиоды используются для индикации сигналов в каналах RGB. После того как Вы разберетесь как это работает — можете их удалить.

## Усовершенствуем устройство

### Больше цветов

Наверное вы не очень удивлены голубым цветом светодиода. Чтобы изменить его цвет на другой — отредактируйте программу: setColor(ledDigitalOne, CYAN); ----> setColor(ledDigitalOne, \*\*NEW COLOR\*\*);

### Отображение случайных цветов

Конечно мы можем больше, чем просто показывать постоянные цвета. Отредактируйте программу следующим образом:

```
void loop(){
  //setColor(ledDigitalOne, CYAN);
  randomColor()
}
```

### Аналоговый контроль цвета

Переключаться между фиксированными цветами достаточно интересно. Но если использовать аналоговый контроль, можно получить практически неограниченное количество цветов. Загрузите пример по адресу:

<http://ardx.org/MABE12R>

## Есть еще вопросы?

Подробности, где купить детали к проекту, где задать вопросы:

<http://www.solarbotics.com>

A large rectangular area with rounded corners, filled with horizontal lines for writing notes. The lines are evenly spaced and cover most of the page's width and height, leaving margins at the top and bottom.

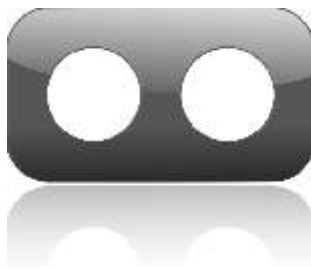
**.:Место для записей:.**

**ЗАМЕТКИ**

A large rectangular area with rounded corners, filled with horizontal lines for writing notes. The lines are evenly spaced and cover most of the page's width and height, leaving margins at the top and bottom.



[www.oomlout.com](http://www.oomlout.com)



Эти материалы выпущены под лицензией Creative Commons Attribution-Share Alike 3.0 Unported License. Чтобы ознакомиться с лицензией, посетите: <http://creativecommons.org/licenses/by-sa/3.0/> или напишите в Creative Commons по адресу: 171 Second Street, Suite 300, San Francisco, California 94105, USA.

