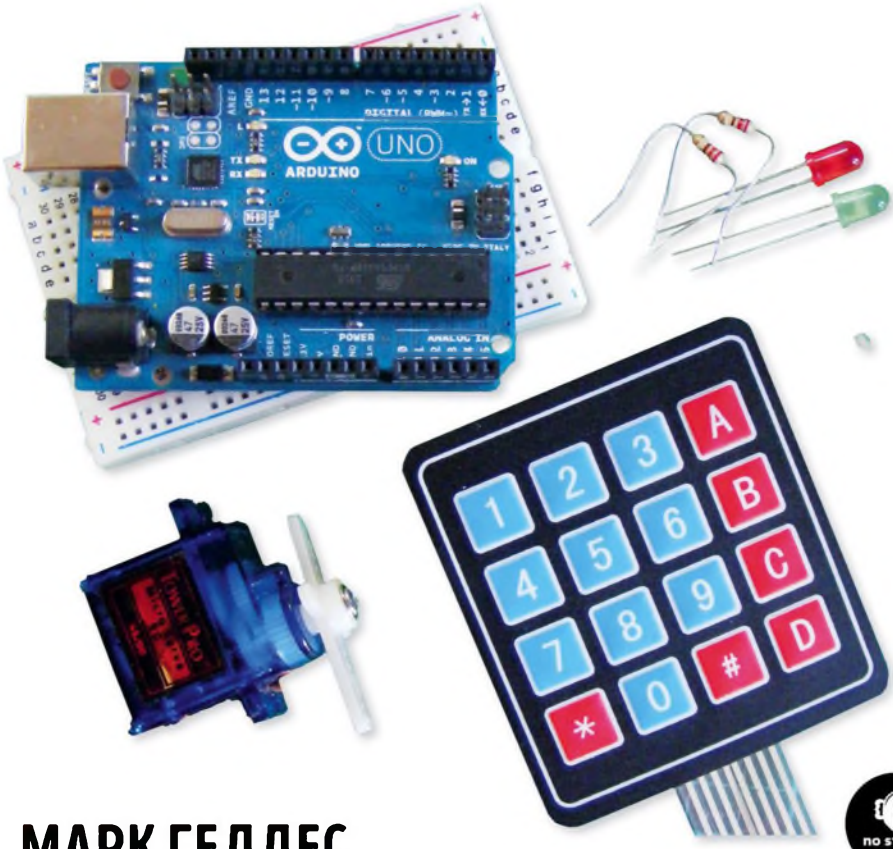
25 КРУТЫХ ПРОЕКТОВ С ARDUINO

САМЫЕ ИНТЕРЕСНЫЕ ПРОЕКТЫ, КОТОРЫЕ МОЖЕТ СОБРАТЬ ДАЖЕ НАЧИНАЮЩИЙ



МАРК ГЕДДЕС

no starch press

ЭЛЕКТРОНИКА
ДЛЯ НАЧИНАЮЩИХ

MARK GEDDES

ARDUINO PROJECT HANDBOOK

25 PRACTICAL PROJECTS TO GET YOU STARTED

МАРК ГЕДДЕС

25 КРУТЫХ ПРОЕКТОВ С ARDUINO

САМЫЕ ИНТЕРЕСНЫЕ ПРОЕКТЫ,
КОТОРЫЕ МОЖЕТ СОБРАТЬ ДАЖЕ НАЧИНАЮЩИЙ

Москва
2019

УДК 004
ББК 32.97
Г28

Mark Geddes

ARDUINO PROJECT HANDBOOK: 25 PRACTICAL PROJECTS TO GET YOU STARTED

Геддес, Марк.

Г28 25 крутых проектов с Arduino / Марк Геддес ; [пер. с англ. М.А. Райтмана]. — Москва : Эксмо, 2019. — 272 с. — (Электроника для начинающих).

ISBN 978-5-04-090263-7

Автор книги, Марк Геддес, — энтузиаст Arduino и преподаватель с десятилетним стажем. В своем самоучителе он собрал 25 уникальных проектов, собирая которые можно освоить азы работы с популярным конструктором. В книгу вошли инструкции по созданию таких проектов, как: детектор призраков, монитор полива цветов, дискотечный стробоскоп, световой диммер, ракетная пусковая установка, детектор привидений, предсказатель судьбы и многие другие.

УДК 004
ББК 32.97

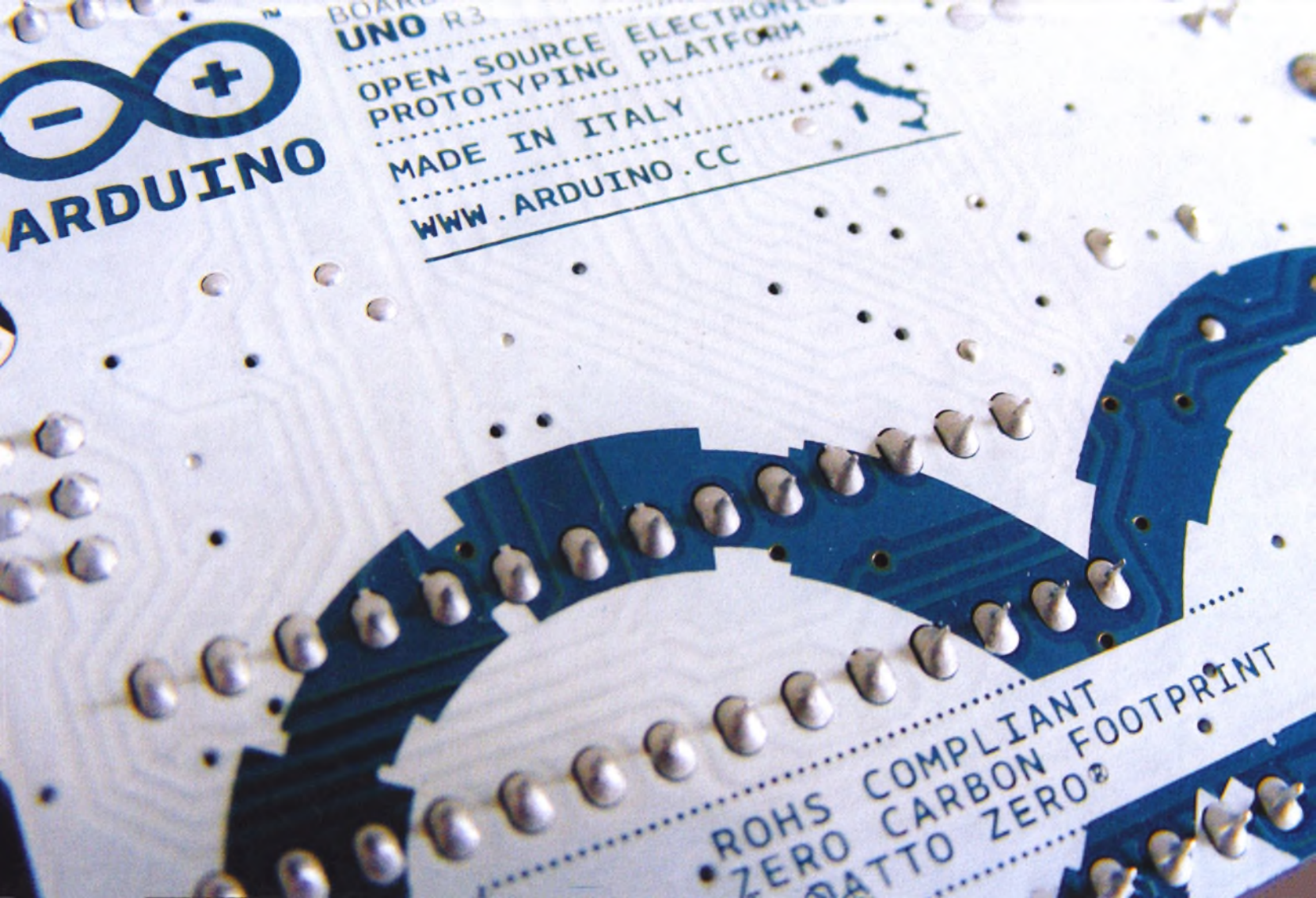
ISBN 978-5-04-090263-7

КЭМЕРОН И ДЖЕММА,

ВЫ — ТВОРЦЫ

И СОЗИДАТЕЛИ БУДУЩЕГО.

ЭТА КНИГА ДЛЯ ВАС!


ARDUINO
UNO R3
OPEN-SOURCE ELECTRONIC
PROTOTYPING PLATFORM
MADE IN ITALY
WWW.ARDUINO.CC
ROHS COMPLIANT
ZERO CARBON FOOTPRINT

СОДЕРЖАНИЕ

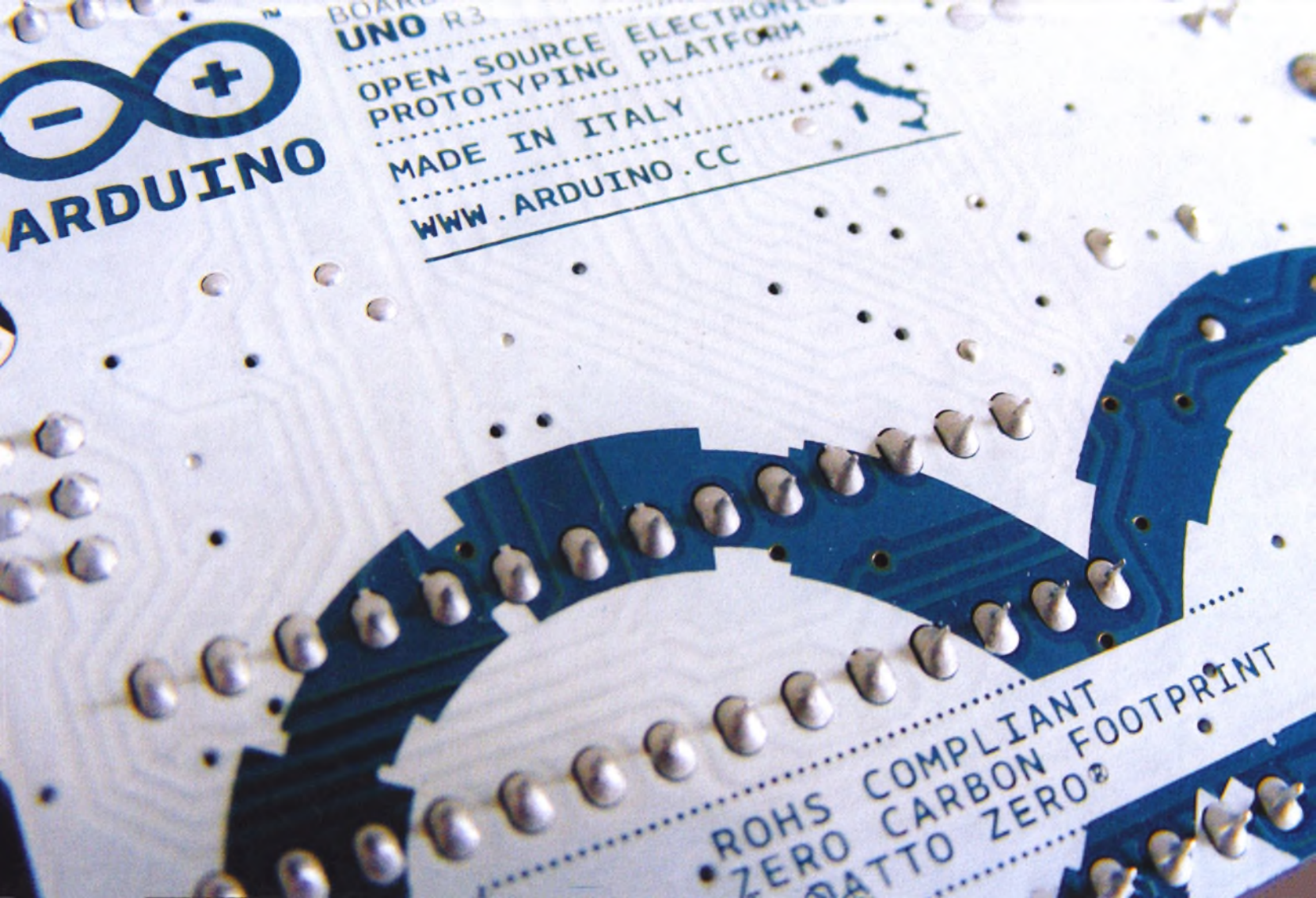
Часть 3. Движение

Часть 4. Отображение

Часть 5. Работа с числами

Часть 6. Безопасность

Часть 7. Продвинутые проекты


ARDUINO
UNO R3
OPEN-SOURCE ELECTRONIC
PROTOTYPING PLATFORM
MADE IN ITALY
WWW.ARDUINO.CC
ROHS COMPLIANT
ZERO CARBON FOOTPRINT

БЛАГОДАРНОСТИ

Большое спасибо фантастической команде No Cramble Press, особенно Элизабет Чедвик и Серене Янг, за их поддержку и руководство в создании этой книги. Спасибо Кристоферу Стэнтону за его технические обзоры и предложения.

Эта книга не существовала бы, если бы не вдохновляющие основатели Arduino: Массимо Банзи, Дэвид Куартиель, Том Иго, Джанлука Мартино и Дэвид Меллис. Спасибо, что познакомили меня и весь мир с чудом, которым является Arduino. Спасибо, Кен Ширрифф, Уорвик Смит, Стивен де Ланнуа, и Абдулла Альхазми, за любезное разрешение воспроизвести ваши проекты.

Я должен поблагодарить мою замечательную жену, Эмили, что она так поддерживает и терпит меня последний год и освободила комнату в нашем доме, чтобы, я как «человек-пещера», мог собрать все эти проекты — и за поддержание порядка ежедневно!

Спасибо моим родителям, Фрэнку и Лорне, за то, что позволили мне в детстве свободно разбирать вещи и не жаловаться, когда у меня повсюду были провода. Если бы не их поддержка, у меня не было бы страсти к электронике и гаджетам, которые у меня все еще есть сегодня. Спасибо также Дэвиду и Линде за их фантастическую поддержку, одобрение и веру.

ВВЕДЕНИЕ

Arduino — это недорогой маленький компьютер, который можно запрограммировать, создав на его основе бесчисленное множество проектов, ограниченных только вашим воображением. Как вы вскоре увидите, с помощью Arduino можно собрать самые разные устройства, такие как детектор призраков, управляемый джойстиком лазер, электронные игровые кости, лазерная сигнализация, датчик движения, система ввода с клавиатуры и многие другие. Все эти проекты легко реализуются и имеют общее место — они все построены на базе Arduino.

В начале 1980-х годов в местном книжном магазине я наткнулся на журнал с великолепной статьей, озаглавленной в духе «Гаджеты и штуковины». Проекты в статье были простыми, вроде использования прожекторов для сборки маяка или построения вращающегося стола на основе старых часов. Идеи из этой статьи вдохновили меня творить всю жизнь.

Мое любопытство привело к тому, что я разобрал несколько домашних электронных приборов для экспериментов и разобрался, как они работают. Обычно мне с трудом удавалось собрать их обратно, но зато оставался неплохой комплект деталек, с которыми я обожал возиться. (Кстати, это прекрасный метод получения различных комплектующих для проектов.)

Я помню, как соединил несколько маленьких лампочек для фонарика, построив прожектор для настольного футбола Subbuteo, и собрал небольшой громкоговоритель, чтобы слушать музыку во время перерыва между таймами. Мне даже удалось достать несколько светодиодов из игрушки «Звездные войны», которые я моментально сжег, так как тогда не подозревал о существовании резисторов. Я баловался с моторчиками, пьезоизлучателями и солнечными батареями, создавая сигнализации для защиты от негодяев и гоночные супертачки (не забыв при этом спалить несколько моторчиков).

Примерно в это же время, в 1983 году, английской компанией Sinclair Research был выпущен микрокомпьютер ZX Spectrum 48k, благодаря которому домашние компьютеры стали доступны каждому. В США распространение получил компьютер Commodore 64.

Будучи заявленным как серьезная и мощная машина, ZX Spectrum использовался чаще всего для создания игр благодаря предустановленной на нем среде программирования BASIC. В результате пользователи занялись домашней разработкой игр для «Спектрума».

Интерес к программированию возник и у меня, но в то время я не мог совместить эти два увлечения. Физическая компьютеризация, когда программное и аппаратное обеспечение стали отражать тренды в реальном мире, началась в районе 80-х годов, но была ограничена сферой высоких компьютерных и робототехнических технологий, недоступных большинству домашних потребителей. Сейчас, более чем 30 лет спустя, с появлением Arduino я обнаружил, что опять вожусь с электроникой, но теперь я могу заняться программированием, чтобы претворить свои проекты в жизнь.

РЕВОЛЮЦИЯ ARDUINO

Arduino представляет собой небольшой компьютер, который можно запрограммировать для связи с различными электронными компонентами и управления ними. Плата Arduino оборудована несколькими контактами, способными работать как в режиме *ввода*, что означает возможность получать через них данные от различных компонентов, таких как коммутаторы, кнопки и датчики, так и *вывода*, допуская передачу данных для управления устройствами, такими как моторы, лампы и пьезоизлучатели. Такой тип программируемых плат известен под названием *микроконтроллер*.

Проект Arduino был запущен в 2005 году в итальянском городе Ивреа с целью создания устройства, подходящего под студенческие интерактивные разработки, которое было бы менее дорогостоящим, чем имеющиеся на то время прототипы. Основатели, Массимо Банзи и Дэвид Куартиллес, назвали проект в честь местного бара «Ардуино»*.

Плата Arduino состоит из двух основных частей: аппаратной (микроконтроллера), являющейся «мозгом» платы, и программной, которая используется для пересылки кода вашей программы «мозгу». Вы можете бесплатно загрузить

* С итальянского слово переводится как «близкий друг».

программное обеспечение Arduino, называемое *интегрированной средой разработки*, или кратко IDE, integrated development environment.

Среда разработки имеет простой интерфейс и может работать на компьютере под управлением Windows, macOS или Linux. Она используется для создания *скетчей* (файлов с кодом программ для Arduino), которые затем передаются на плату Arduino через USB-кабель. Скетч сообщает микрокомпьютеру, какие действия следует выполнять. Я расскажу более подробно об аппаратной и программной частях в нескольких следующих главах.

После того, как плата Arduino запрограммирована, ее можно отключить от компьютера и использовать автономно, подав питание от USB-интерфейса, внешнего источника питания или батареек.

ОБ ЭТОЙ КНИГЕ

Вы спросите, что вдохновило меня на написание этой книги? Ведь Интернет содержит массу руководств, видеороликов и статей об Arduino и собираемых на базе этой платы проектах. Я отвечу просто — в большинстве случаев отсутствует наглядный подход к коду, требуемый для выполнения этих проектов. Подобно авторам публикации «Гаджеты и штуковины», вдохновившей меня много лет назад, в этой книге я задался целью помочь вам собрать простые проекты, которые дадут вам толчок к созданию собственных безделушек, используя опыт и получаемые знания.

В этой книге цепи собираются на *макетных платах*. Это наилучший способ разобраться, как работают цепи, т.к. подключения не припаяны намертво; если вы сделаете ошибку, то можете просто отсоединить перемычку и подключить ее иначе. Каждый проект содержит пошаговую инструкцию по подключению основных компонентов, а также фотографии, которые помогут вам быстрее разобраться со схемой. В большинстве проектов в качестве краткого справочника используются таблицы.

В проектах представлены электрические схемы для наглядного отображения соединений, подобно рис. 1. Такие схемы я создал в бесплатной программе Fritzing (скачать можно по адресу www.fritzing.org), используемой энтузиастами для построения наглядных схематических изображений своих проектов.

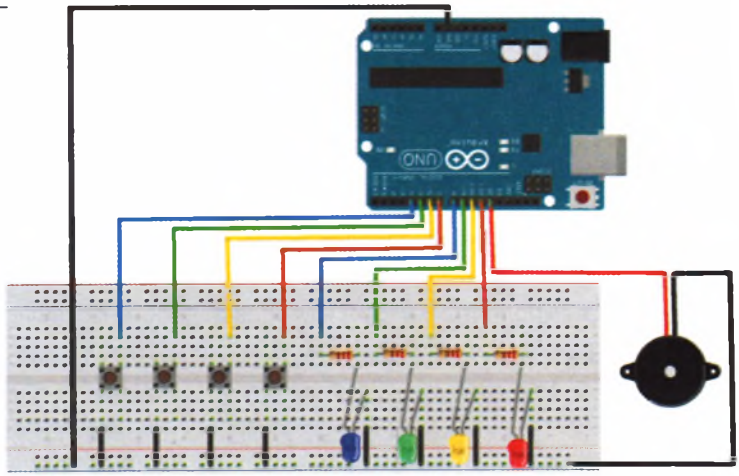


РИСУНОК 1.1
Пример схемы в программе Fritzing

Каждый проект также содержит код, необходимый для программирования платы Arduino, поэтому не стоит волноваться об изучении программы перед тем, как вы начнете сборку. Первые проекты содержат простые объяснения происходящего в коде, чтобы помочь вам понять процесс программирования и разобраться, как вносить собственные изменения, если на то возникнут причины. Если вы не хотите набирать код самостоятельно, то можете загрузить скетчи с сайта https://eksmo.ru/files/arduino_geddes.zip.

Проекты в этой книге начинаются с элементарных и постепенно усложняются по мере прочтения. Не могу сказать, что эту книгу можно назвать углубленным руководством по электронике или программированию, но она отлично подойдет начинающим. Я написал эту книгу, чтобы научить вас собирать собственные устройства. Предоставляя вам технические приемы, я могу сконцентрироваться на творческих элементах разработки. Смысл в том, что изучение работы цепей поможет вашему воображению отыскать пути для применения этих цепей на практике.

В этой книге содержится полезная практическая информация, благодаря которой вы сможете, к примеру, разобраться в способах подключения контактов и использовать эти знания в других проектах. Вы

также можете объединять проекты для сборки более сложных и интересных устройств.

Множество книг по Arduino фокусируются на программировании, и это неплохо, но, на мой взгляд, в моей книге очень удачно нашлось место для электронных проектов в духе «подключи и работай». Выполняя упражнения, вы постепенно научитесь, как их делать.

СТРУКТУРА ЭТОЙ КНИГИ

Сложность проектов в книге постепенно возрастает, благодаря чему ваши знания и навыки работы с компонентами будут расти. Материал в книге я разделил на следующие части.

Часть I: Свет. Вы начнете с управления старыми добрыми светодиодами с помощью кнопок и переменных резисторов, а затем научитесь подключать компоненты и соберете дискотечный стробоскоп, прибор для контроля полива комнатных растений и даже детектор призраков (!).

Часть II: Звук. Из этой части вы узнаете о пьезоизлучателях — очень полезных устройствах, которые могут не только издавать, но и обнаруживать звук. Вы напишете музыку с помощью Arduino, создадите простую и смешную игру для тренировки памяти и соорудите секретную систему опознавания громкости стука.

Часть III: Движение. Во всех проектах из этой части используются сервоприводы — небольшие моторы с рычагом, которые можно использовать в самых разнообразных целях. Вы построите лазер, управляемый с джойстика, и научитесь контролировать сервоприводы дистанционно с помощью кнопок.

Часть IV: Отображение. Жидкокристаллический дисплей пригодится во множестве проектов для отображения сообщений и результатов игры. В этой части вы узнаете, как настроить ЖК-дисплей, собрать метеостанцию для оповещения о погоде, а также создать игры «Предсказатель судьбы» и «Игра на скорость».

Часть V: Работа с числами. В этой части вы воспользуетесь числовыми светодиодными индикаторами для создания

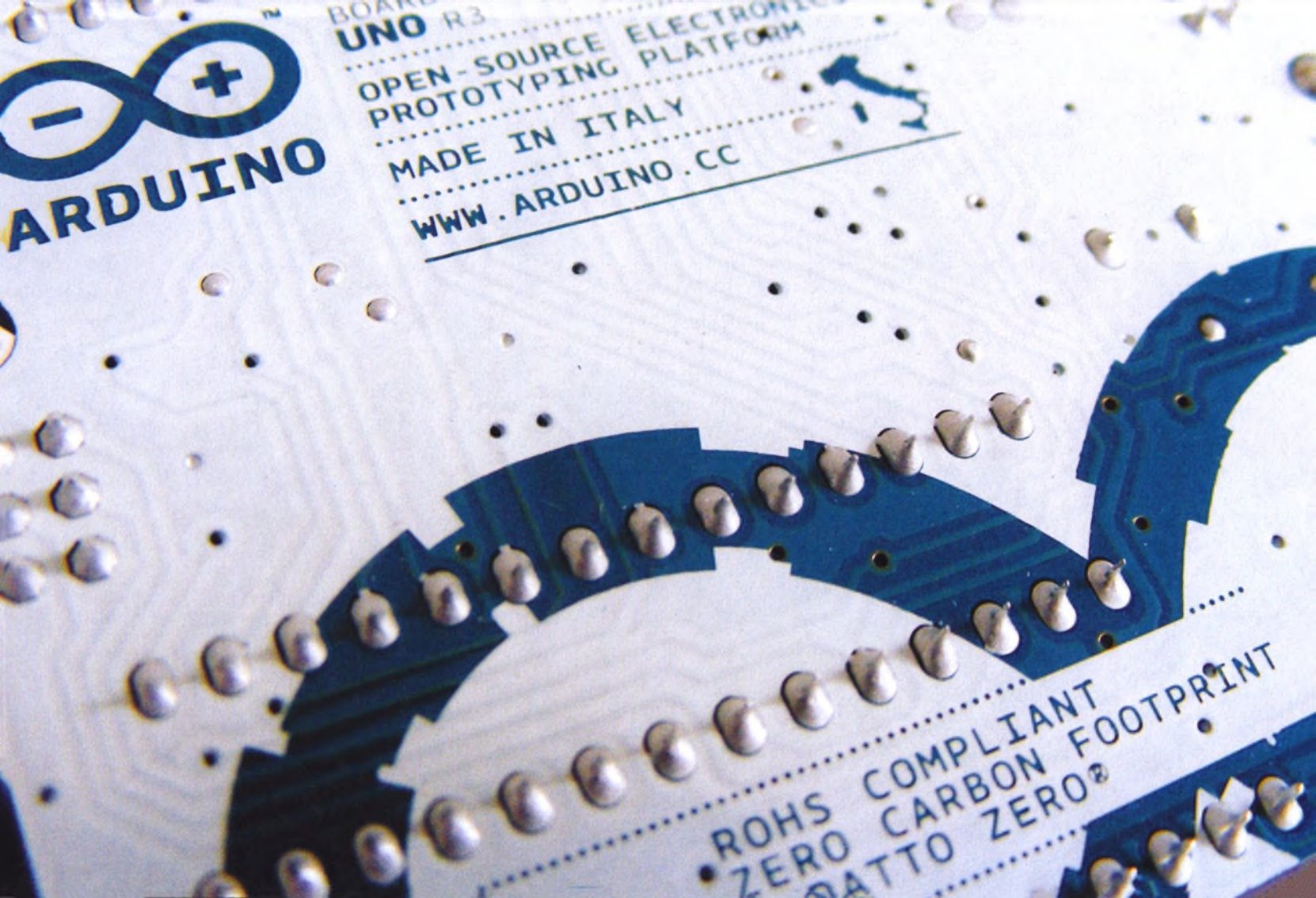
электронного аналога игровых костей и системы обратного отсчета с предохранителем для ракетной пусковой установки.

Часть VI: Безопасность. Описываемые здесь довольно сложные проекты позволят вам защитить помещение с помощью датчиков движения, лазерной сигнализации и автоматической турели, а также вы сможете собрать системы безопасности с числовым кодом и кардридером, чтобы ограничить доступ нежеланным посетителям.

Часть VII: Продвинутые проекты. В этой части вы подключите к Arduino светодиодную матрицу и устроите настоящее световое шоу. А «на десерт» получите порцию дополнительных навыков, собрав собственный компьютер Arduino для использования в своих будущих проектах.

Эти проекты необязательно собирать строго последовательно, поэтому, если какой-нибудь проект вам нравится больше остальных, переходите прямо к нему. Но все же я рекомендую собрать несколько начальных проектов, чтобы получить базовые сведения и навыки, которые пригодятся при создании более сложных конструкций.

Я написал книгу, которую безуспешно искал во времена моего знакомства с Arduino. Надеюсь, эта книга станет для вас успешным начинанием.


ARDUINO
UNO R3
OPEN-SOURCE ELECTRONIC
PROTOTYPING PLATFORM
MADE IN ITALY
WWW.ARDUINO.CC
ROHS COMPLIANT
ZERO CARBON FOOTPRINT

ПРОЕКТ 0: НАЧАЛО РАБОТЫ

ПРЕЖДЕ ЧЕМ НАЧАТЬ СБОРКУ ЦЕПЕЙ НА ОСНОВЕ ARDUINO, СЛЕДУЕТ УЗНАТЬ О НЕКОТОРЫХ ВАЖНЫХ ВЕЩАХ И ВЫПОЛНИТЬ ПАРУ ДЕЙСТВИЙ. ДАВАЙТЕ ВЗГЛЯНЕМ НА АППАРАТНОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, КОТОРОЕ ВАМ ПОНАДОБИТСЯ ДЛЯ РАБОТЫ С ПРОЕКТАМИ ИЗ ЭТОЙ КНИГИ, И НАСТРОИМ КОМПЬЮТЕР ДЛЯ РАБОТЫ С НИМИ. ЗАТЕМ ВЫ ОПРОБУЕТЕ ARDUINO В ПРОСТОМ ПРОЕКТЕ СО СВЕТОДИОДАМИ И ИЗУЧИТЕ НЕСКОЛЬКО ПОЛЕЗНЫХ ПРИЕМОВ, ТАКИХ КАК ПАЙКА И ЗАГРУЗКА ПОЛЕЗНЫХ БИБЛИОТЕК КОДА.

АППАРАТНОЕ ОБЕСПЕЧЕНИЕ

Для начала взглянем на плату Arduino Uno и некоторые дополнительные компоненты, которые вы будете использовать практически в каждом проекте.

Arduino Uno

Доступно множество плат Arduino, но в этой книге используется наиболее популярная из них — Arduino Uno, показанная на рис. 0.1. Arduino Uno — продукт с открытым кодом (т. е. его архитектура может копироваться для других изделий бесплатно), поэтому помимо оригинальной платы стоимостью около 2000 рублей в магазинах вы найдете еще множество совместимых клонов примерно за 700 рублей.

Давайте рассмотрим плату Arduino Uno.

Рисунок 0.1

Плата Arduino Uno

Компьютер Arduino управляет подключенными к нему компонентами, такими как электромоторы или светодиоды, пересылая им *выходные* данные (которые передаются с платы Arduino). Данные, которые Arduino считывает с датчиков, называются *входными* (т. е. поступающими на Arduino). Плата оборудована 14 цифровыми контактами ввода/вывода (контакты 0–13). Каждый из них может быть настроен как вход, так и как выход. Полная спецификация контактов приведена в приложении Б.

Питание

Плата Arduino Uno получает электропитание через интерфейс USB от компьютера при подключении к нему для загрузки программы. Если Arduino не подключена к компьютеру, вы можете обеспечить ее автономную работу, подключив блок

питания постоянного тока на 9 В или батарейный отсек на 9 В со штекером 2,1 мм, в котором центральный контакт — положительный (см. рис. 0.2). Штекер нужно вставить в соответствующий разъем на плате Arduino.

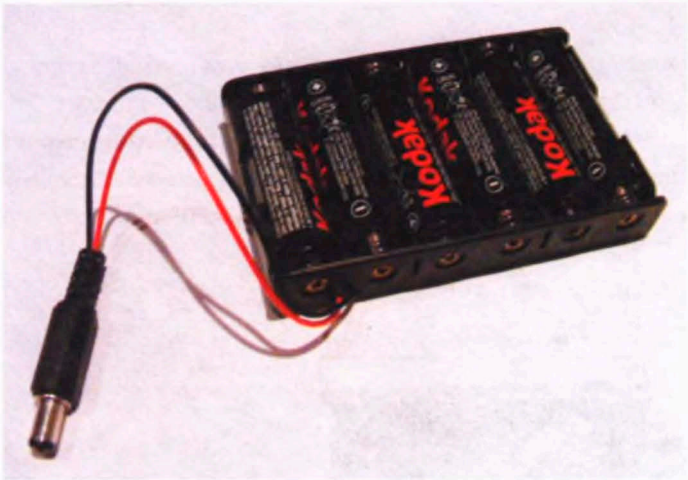


Рисунок 0.2

Батарейный отсек на 9 В, который вы можете подключить к Arduino и таким образом обеспечить питание платы

Макетные платы

Макетная плата используется в качестве строительной основы для прототипирования электронных цепей. Во всех проектах этой книги макетные платы используются вместо пайки.

Современные макетные платы, такие как на рис. 0.3, сделаны из пластика с просверленными отверстиями (называемыми *точками привязки*), в которые вставляются ножки электронных компонентов или перемычки и удерживаются с помощью защелок. Отверстия соединены между собой полосами проводящего материала, которые проходят под платой. Схема соединений контактов на макетной плате показана на рис. 0.3.

Макетные платы бывают разных размеров. Для сборки цепей из этой книги вам понадобятся четыре макетные платы: две полноразмерные, обычно с 830 отверстиями, одна полуразмерная с 420 отверстиями и одна мини-плата с 170 отверстиями. Полноразмерная макетная плата идеальна для проектов, в которых используется жк-дисплеи или большое количество компонентов. Полуразмерная и мини-платы удобнее для небольших проектов. Я рекомендую для проектов в этой книге приобрести макетные платы, показанные на рис. 0.3, с красными и голубыми линиями и канавкой посередине.

В основной области платы расположены 30 столбцов отверстий, замкнутых вертикально, как показано на рис. 0.3. В центре платы находится канавка. При

сборке цепей ее часто будут перекрывать компоненты, ножки которых будут вставляться в отверстия с обеих сторон канавки.

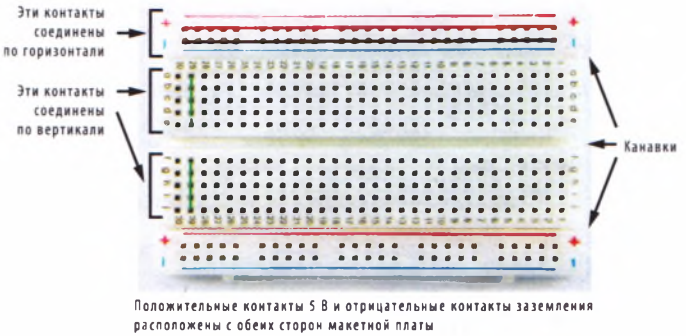


Рисунок 0.3

Схема соединений контактов макетной платы

СОВЕТ

В области электроники принято использовать красные провода для подключения к контактам 5 В (+) и черные — к контактам заземления (–), чтобы случайно не перепутать схему подключения. Остальные провода могут иметь любой другой цвет.

Канавка позволяет подключать компоненты с двумя и более парами ног, избегая их замыкания, которое может привести к неработоспособности цепи или даже повреждению компонентов. В верхней и нижней части макетной платы расположены ряды отверстий, помеченные голубыми и красными линиями, которые используются для подачи питания компонентам, установленным в основной части платы (см. рис. 0.4). Они называются *шинами питания*.

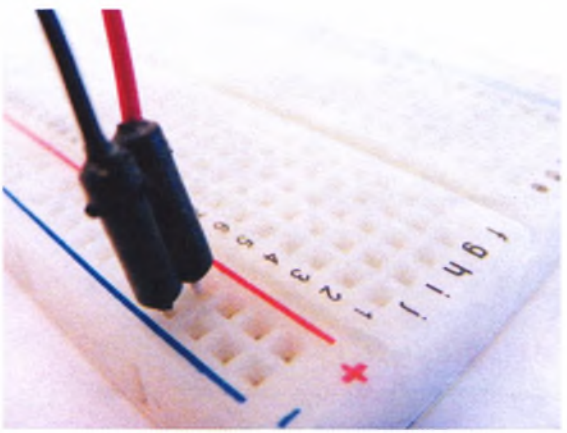


Рисунок 0.4

Положительная и отрицательная шины питания макетной платы

Эти отверстия соединены по горизонтали. Красные линии обозначают положительный контакт, а голубые — отрицательный (или заземление, как вы вскоре увидите).

Провода-перемычки

Для подключения компонентов на макетной плате вы будете использовать *провода-перемычки*. Они представляют собой изолированные одножильные провода с обработанными концами, что упрощает вставку и снятие проводов. Вы можете использовать и обычные провода, но убедитесь, что они одножильные, т.к. многожильный провод будет проблематично воткнуть в отверстие для зажима.

Когда вы вставляете перемычку в отверстие макетной платы, провод удерживается маленьким пружинным зажимом, образуя электрическое соединение в этом ряду, состоящее обычно из пяти отверстий. Затем вы можете установить компонент в соседнее отверстие, чтобы замкнуть цепь, как это показано на рис. 0.5.

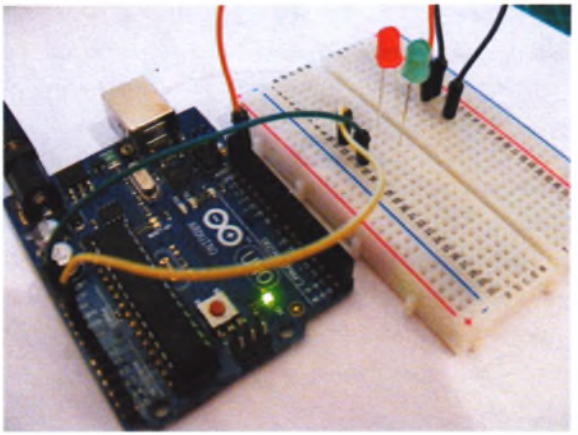


Рисунок 0.5

Пример сборки цепи на макетной плате

ПРИМЕЧАНИЕ

Т. к. версии среды разработки могут меняться довольно часто, я не буду описывать процесс их установки. Все версии среды разработки и подробное описание процесса установки для той или иной операционной системы доступны в Интернете по адресу www.arduino.cc.

ПРОГРАММИРОВАНИЕ ARDUINO

Чтобы компоненты в проекте выполняли необходимые нам действия, нам нужно написать программу, которая предоставит Arduino нужные инструкции. Разработка программ для Arduino ведется в инструменте, *именуемом интегрированной средой разработки (IDE)*. Эту среду разработки можно

скачать бесплатно с сайта *www.arduino.cc/en/Main/Software*, выбрав дистрибутив для операционной системы Windows, macOS или Linux. После установки и запуска среды вы сможете писать компьютерные программы (наборы пошаговых инструкций, известных в мире Arduino под термином скетчи), которые затем передаются в память Arduino через USB-интерфейс. Микрокомпьютер Arduino будет выполнять инструкции из скетча, попутно взаимодействуя с окружающей средой.

Интерфейс среды разработки

Запустив среду разработки Arduino, вы должны увидеть окно, показанное на рис. 0.6.

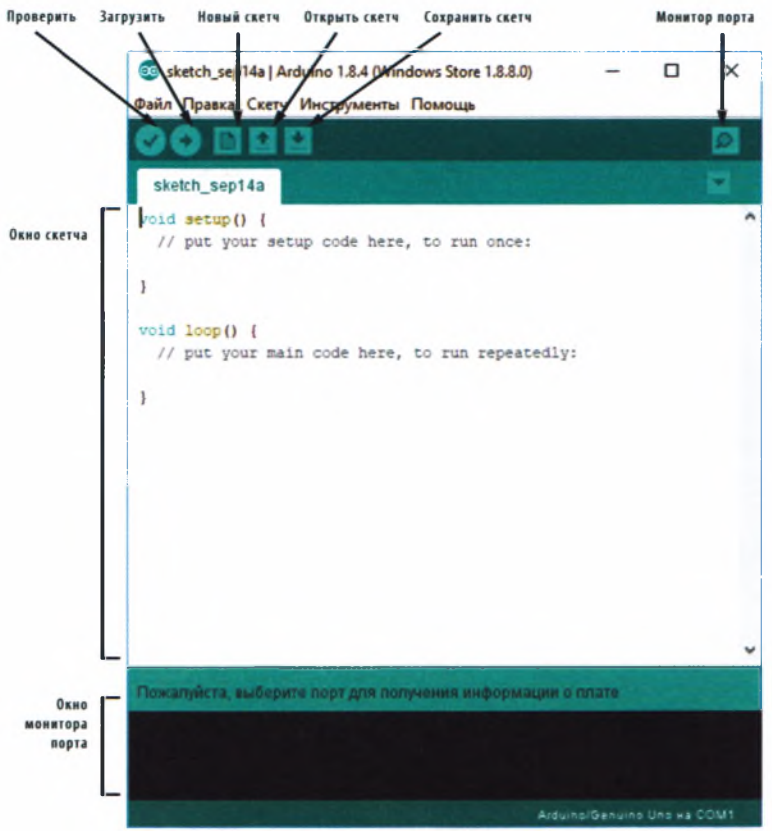


Рисунок 0.6

Среда разработки Arduino

Окно среды разработки Arduino содержит в верхней части панель инструментов с кнопками для наиболее часто используемых действий, в центре область скетча, в которой вы будете писать или просматривать свои программы, и в нижней части монитор порта. На панели монитора порта выводятся уведомления о соединении между вашим компьютером и Arduino, а также возникающие ошибки, если ваш скетч не компилируется должным образом.

Скетчи Arduino

Я предоставлю вам скетчи для всех проектов, и мы разберем их работу. Все скетчи доступны для скачивания по адресу *eksmo.ru/files/arduino_geddes.zip*.

Подобно любой программе, скетчи представляют собой небольшой набор инструкций и весьма критичны к ошибкам. Чтобы убедиться в правильности кода скетча, нажмите кнопку **Проверить** (Verify) в верхней части окна. С ее помощью производится проверка кода на ошибки, а результат выводится на панели монитора порта, например что скетч скомпилирован правильно. Если возникает ошибка, вы всегда можете скачать авторскую версию скетча, а затем скопировать его код и вставить в окне среды разработки.

Библиотеки

В мире Arduino библиотека представляет собой небольшой фрагмент кода, выполняющий определенную функцию. Вместо того чтобы постоянно вводить один и тот же код в каждый скетч, вы можете добавить команду, которая добавит код из библиотеки. Этот прием экономит время и позволяет легко подключаться к таким электронным компонентам, как датчик, индикатор или модуль.

Среда разработки Arduino содержит множество встроенных библиотек, таких как LiquidCrystal, позволяющей легко обмениваться данными с ЖК-дисплеями.

В Интернете вы можете найти гораздо больше библиотек. Для выполнения проектов этой книги вам нужно будет импортировать следующие библиотеки: RFID, Tone, Pitches, Keypad, Password, Ultrasonic, NewPing, IRRemote и DHT. Вы найдете все необходимые библиотеки в архиве по ссылке *eksmo.ru/files/arduino_geddes.zip*.

После того как библиотеки будут загружены и распакованы, вам нужно их установить. Чтобы установить библиотеку в среде разработки Arduino версии 1.0.6 и выше, выполните следующие шаги.

1. Выберите команду меню **Скетч ▸ Подключить библиотеку ▸ Добавить .ZIP библиотеку** (Sketch ▸ Include Library ▸ Add .ZIP Library).
2. Выберите загруженный ZIP-файл и нажмите кнопку **Open** (Открыть). В более старых версиях среды разработки Arduino нужно будет распаковать архив с библиотекой и скопировать каталог с содержимым в папку *sketchbook/ libraries* (Linux), *My Documents\ Arduino\ Libraries* (Windows) или *Documents/Arduino/libraries* (macOS).

Чтобы установить библиотеку вручную, найдите и распакуйте ZIP-файл с библиотекой. Например, если вы устанавливаете библиотеку под названием

keypad из архивного файла *keypad.zip*, вам следует открыть файл *keypad.zip*, который будет распакован в каталог с именем *keypad*, содержащий файлы наподобие *keypad.cpp* и *keypad.h*. Перетащите каталог keypad в папку с библиотеками в соответствии с вашей операционной системой: *sketchbook/ libraries* в Linux, *My Documents\ Arduino\ Libraries* в Windows и *Documents/Arduino/libraries* в macOS. Затем перезапустите среду разработки Arduino.

Библиотеки перечисляются в начале скетча. Эти строки легко определить, т.к. код начинается с команды #include. Имена библиотек заключены в угловые скобки <> и заканчиваются расширением .h, как показано в следующем вызове библиотеки Servo:

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

Сразу установите перечисленные библиотеки, которые вам понадобятся в проектах, чтобы сэкономить в будущем немного времени.

ПЕРВЫЙ ТЕСТ ARDUINO: МИГАЮЩИЙ СВЕТОДИОД

Теперь, когда мы ознакомились с аппаратной и программной частями наших проектов, начнем с классического первого проекта Arduino: *мигающего светодиода*.

Это не только простейший способ убедиться в работоспособности платы Arduino, но также знакомство с простейшим скетчем. Как я упоминал ранее, скетч — это просто набор инструкций, выполняемых на компьютере. В памяти Arduino одновременно может храниться только один скетч, поэтому, как только вы загрузите ваш скетч в память Arduino, он будет выполняться каждый раз при включении Arduino, пока вы не загрузите новый скетч.

В этом проекте мы будем использовать тестовый скетч, поставляемый в составе со средой разработки Arduino. Эта программа включает светодиод на 1 секунду, затем выключает на 1 секунду и т.д. Светодиод излучает видимый свет, когда через него проходит ток с небольшим напряжением. Светодиод будет работать только при постоянном токе, текущем в одном направлении, поэтому более длинная ножка должна быть подключена к положительному контакту. Для работы светодиода также требуется резистор, снижающий силу тока, иначе светодиод может перегореть. На контакте **13** платы Arduino встроен резистор, которым мы и воспользуемся.

Выполните следующие шаги, чтобы провести тестирование:

1. Вставьте длинную ножку (также известную как +5 В, положительный контакт или *анод*) светодиода в контакт **13** на плате Arduino, как показано на рис. 0.7. Подключите короткую ножку (также известную как отрицательный контакт или *катод*) к контакту **GND** (рядом с контактом **13**).

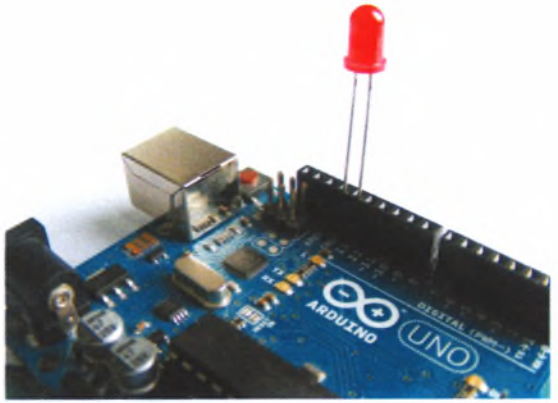


Рисунок 0.7

Сборка проекта мигающего светодиода

2. Подключите Arduino к вашему компьютеру с помощью USB-кабеля.
3. Введите следующий код в окне среды разработки:

```
❶ // Проект мигающего светодиода

❷ int led = 13;
❸ void setup() {
❹   pinMode(led, OUTPUT);
  }
❺ void loop() {
❻   digitalWrite(led, HIGH);
❼   delay(1000);
❽   digitalWrite(led, LOW);
❾   delay(1000);
❿ }
```

4. Нажмите кнопку **Verify** (Проверить) в виде галочки и убедитесь, что программа работает правильно.
5. Теперь нажмите кнопку **Upload** (Загрузка), чтобы передать ваш скетч на Arduino.

Структура кода

Рассмотрим, что происходит в каждой строке скетча.

❶ Это комментарий. Каждая строка программы, которая начинается с символов //, предназначена для чтения только пользователем и игнорируется Arduino, что вы можете использовать для добавления комментариев и описания вашего кода (этот процесс называется *комментированием* кода). Если комментарий занимает больше одной строки, начните его с символов /*

и закончите символами */. Строки текста внутри этих символов будут проигнорированы Arduino.

❷ Присваиваем контакту **13** имя led. Каждое упоминание led в скетче относится к контакту **13**.

❸ Обозначает, что код настройки между фигурными скобками {} после этой инструкции будет выполнен однократно при запуске программы. С открывающей фигурной скобки { начинается код настройки.

❹ Сообщает Arduino, что контакт **13** работает в режиме вывода. Таким образом мы будем подавать питание на светодиод. Закрывающая фигурная скобка } обозначает конец кода настройки.

❺ Создает цикл. Код, который находится между фигурными скобками {}, после инструкции loop() запустится один раз при включении Arduino и будет повторяться, пока на Arduino подается питание.

❻ Инструктирует Arduino перевести led (контакт **13**) в режим HIGH, что значит подать питание на этот контакт. Другими словами, это перевод контакта во включенное состояние. В этом скетче обозначает включение светодиода.

❼ Инструктирует Arduino подождать 1 секунду. Т.к. время в Arduino отсчитывается в миллисекундах, а 1 секунда = 1000 миллисекунд, здесь указано значение 1000.

❽ Инструктирует Arduino перевести led (контакт **13**) в режим LOW, т. е. отключить питание на контакте. Эта инструкция отключает светодиод.

❾ Инструктирует Arduino опять подождать 1 секунду.

❿ Этой закрывающей фигурной скобкой заканчивается цикл. Весь код, следующий после кода настройки (setup), должен быть заключен в фигурные скобки. Часто пользователи допускают ошибку, пропуская открывающую или закрывающую скобку в скетче, тем самым не позволяя ему скомпилироваться правильно. После этой фигурной скобки скетч возвращается назад к началу цикла в строке ❺.

Запустив этот код, вы увидите, как светодиод начал мигать.

Вы протестировали плату Arduino и поняли принцип работы скетча и процесс его загрузки. Теперь обратите внимание на компоненты, которые вам понадобятся при выполнении проектов из этой книги. В приложении А содержится более подробная информация о каждом компоненте и его предназначении.

СПИСОК КОМПОНЕНТОВ ДЛЯ ПРОЕКТОВ

Ниже приведен полный перечень компонентов, которые вам понадобятся при работе над проектами из этой книги. Наиболее важный из них — это, разумеется, сама плата Arduino; во всех проектах используется версия Arduino Uno R3. Как упоминалось ранее, словом Arduino обозначаются только официальные платы, но у некоторых компаний, таких как «Электронные войска», SlicMicro, Sainsmart и Adafruit, можно приобрести совместимые клоны. (Вы найдете список официальных магазинов по адресу *arduino.cc/en/Main/Buy/.)*

В начале каждого проекта приводится список необходимых комплектующих, поэтому, если вы хотите выполнить только определенные проекты, можете перейти на соответствующую страницу проекта, который вам понравился, и приобрести компоненты, необходимые только для него. Хотя вы можете приобрести каждый компонент по отдельности, я рекомендую обратить внимание на стартовые наборы Arduino. Множество из них можно найти в Интернете, а в приложении А приведен список интернет-магазинов.

1	Arduino Uno R3 (или клон)
1	батарейный отсек, дающий напряжение 9 В, со штекером 2,1 мм
2	полноразмерные макетные платы
1	полуразмерная макетная плата
1	мини-плата
50	соединительных проводов (перемычек) «папа-папа»
10	соединительных проводов (перемычек) «мама-папа»
30	резисторов с сопротивлением 220 Ом
10	резисторов с сопротивлением 330 Ом
1	резистор с сопротивлением 470 Ом
1	резистор с сопротивлением 10 кОм
1	резистор с сопротивлением 1 МОм
40	5-мм светодиодов: красный, зеленый, желтый, синий (по 10 каждого цвета)
1	потенциометр с сопротивлением 50 кОм
4	тактовых кнопки с четырьмя контактами (ножками)
1	датчик влажности почвы типа HL-69
1	пьезоизлучатель (зуммер)
1	3,5-мм аудиоштекер
2	сервопривода Tower Pro 9g SG90
1	фоторезистор (также известен как LDR)
1	аналоговый пятиконтактный двухосный джойстик
1	корпус с поворотным устройством
1	четырехконтактный ультразвуковой дальномер HC-SR04
1	мембранная клавиатура размером 4×4 кнопки
1	семисегментный светодиодный индикатор
1	четырехразрядный семисегментный последовательный индикатор
1	датчик температуры и влажности DHT11
1	ЖК-дисплей размером 16×2 (совместимый с Hitachi HD44780)
1	датчик наклона с шариком
1	светодиодная RGB-матрица размером 8×8

1 инфракрасный (ИК) датчик с частотой 38 кГц
1 пассивный инфракрасный датчик движения HC-SR501
1 считыватель Mifare RFID RC-522, карта и брелок
4 выходных сдвиговых регистра 74HC595
1 маломощная лазерная указка
1 ракетная установка для квадрокоптера WLToys RC V959
1 микроконтроллер ATMEL ATmega328P
1 кварцевый генератор с частотой 16 МГц (HC-495)
1 5-вольтовый стабилизатор напряжения L7805cv
2 электролитических конденсатора емкостью 100 мкФ
1 клемма для батареи напряжением 9 В (типа «Крона»)
2 дисковых конденсатора емкостью 22 пФ
1 батарея напряжением 9 В (типа «Крона»)

ОБУСТРОЙСТВО ВАШЕГО РАБОЧЕГО МЕСТА

Чтобы максимально эффективно работать с Arduino, вам следует позаботиться об организации рабочего места. Так вам будет проще придумывать новые проекты, а все инструменты будут на своих местах. По возможности, это должно быть отдельное рабочее место, наподобие показанного на рис. 0.8. Некоторые проекты могут потребовать нескольких часов на сборку, а у вас может не хватить времени закончить работу за один день. В этом случае нет ничего хуже, чем необходимость останавливаться и разбирать все только для того, чтобы собрать обратно в следующий раз.

Рисунок 0.8

Пример обустройства рабочего места

Рабочее место может быть устроено где угодно. Вам точно понадобится стол или ровная поверхность, достаточная для компьютера или ноутбука, на котором вы

будете использовать среду разработки и загружать программы, при этом также имеющая достаточно места для сборки самих проектов.

Вам также может понадобиться место для хранения компонентов под рукой и инструментов, которые могут понадобиться, например паяльника, кусачек, ножа, дрели и т.д. Не очень практично держать все инструменты и материалы на столе, поэтому рекомендуется приобрести ящики или кейсы для хранения. Я использую большой кейс для инструментов, таких как паяльник или кусачки, и маленькие кейсы для компонентов. Пластиковые ящики для рыболовных снастей и прочей мелочевки идеальны для хранения комплектующих (см. рис. 0.9), а раздвижной ящик отлично подойдет для хранения паяльника и других небольших инструментов (рис. 0.10).

Рисунок 0.9

Ящики или кейсы для инструментов удобны для хранения компонентов

Рисунок 0.10

Раздвижной ящик отлично подойдет для хранения паяльника и других небольших инструментов

Небольшие пластиковые ящички, обычно используемые для хранения украшений, также хорошо подходят для размещения маленьких компонентов (рис. 0.11).

Рисунок 0.11

Пластиковые контейнеры с маленькими отсеками идеально подойдут для организации очень маленьких предметов

Рассмотрите возможность покупки коврика для резки с разметкой, его можно использовать как твердую непроводящую поверхность (он не проводит электричество), так что вы избавитесь от риска замкнуть накоротко ваши чувствительные электронные компоненты.

НЕОБХОДИМОЕ ОБОРУДОВАНИЕ И ИНСТРУМЕНТЫ

Ниже представлены наиболее полезные инструменты, которые вы можете приобрести для повышения эффективности и комфорта работы (они не обязательны для работы над проектами из этой книги).

- Штатив для плат (третья рука) — полезно для удерживания предметов

- Антистатический коврик с разметкой

- Игольчатые плоскогубцы

- Кусачки

- 30-ваттный паяльник и припой (см. раздел «Краткое руководство по пайке»)
- Оловоотсос — приспособление для удаления припоя

- Инструмент для снятия изоляции (стриппер) — особенно полезен для изготовления перемычек

- Кабель USB (A — B) (принтерный) для подключения Arduino

- Цифровой мультиметр

- Набор отверток

- Многофункциональный инструмент и насадки

- Клеевой пистолет

КРАТКОЕ РУКОВОДСТВО ПО ПАЙКЕ

Некоторые из компонентов, которые вам понадобятся, могут поставляться без штырьковых соединителей (рис. 0.12) для удобства транспортировки, и вам нужно будет припаять их к компонентам самостоятельно. Штырьковые соединители представляют собой линейки штырьков, которые вы прикрепляете к компоненту, чтобы установить соединения с помощью перемычек или вставить их в макетную плату. Они продаются в виде длинных полос, которые можно легко разломить на части требуемого размера, и обычно вставляются в соответствующие отверстия на компоненте.

Рисунок 0.12

Штырьковые соединители

К примеру, модуль RFID, используемый в проекте 23, поставляется без штырьковых соединителей, поэтому я покажу, как припаять их в этом кратком руководстве по пайке.

Если вы хотите глубже изучить приемы пайки, то можете просмотреть удобное руководство в комиксах по ссылке *mightyohm.com/files/soldercomic/FullSolderComic_EN.pdf*.

Первое, что вам потребуется, это паяльник (рис. 0.13). 30-ваттный универсальный паяльник с тонким жалом прекрасно подойдет. Лучше всего приобрести набор, содержащий паяльник, подставку и припой.

Рисунок 0.13

Набор из паяльника и припоя

Во время пайки вы прогреваете область, которую хотите припаять, с помощью паяльника — например, место, где соединяются контакт и ножка компонента, — а затем прикладываете паяльную проволоку к нагретой области. Проволока быстро расплавляется и создает прочное соединение между двумя элементами, которые вы спаяли. Рассмотрим пошаговый процесс.

1. Включите паяльник и подождите не менее пяти минут, чтобы он прогрелся до рабочей температуры.
2. Отломайте нужное количество штырьковых соединителей для вашего компонента. Для модуля RFID из проекта 23 понадобится линия из восьми соединителей. Вставьте их в модуль, как показано на рис. 0.14.
3. Теперь следует припаять контакты. Начните с крайнего левого контакта. Подержите жало паяльника в соприкосновении с контактом и модулем примерно две секунды. Удерживая паяльник в этом месте, добавьте припой; припой должен расплавиться и создать соединение.

ПРИМЕЧАНИЕ

Припой наносится не на сами ножки, а на места их контакта с поверхностью платы модуля.

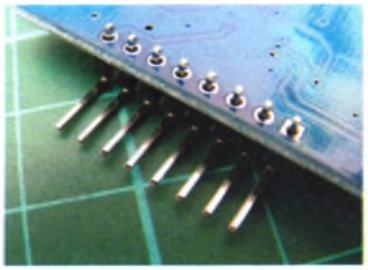


Рисунок 0.14

Вставьте ножки соединителя в отверстия на плате модуля

4. Уберите паяльник и припой — контакт свыше пары секунд может перегреть ваши компоненты и повредить их. Подождите, пока область пайки остынет.
5. Качественная пайка должна выглядеть как сверкающий конус (рис. 0.15).

Попрактиковавшись, вы можете быстро научиться паять.

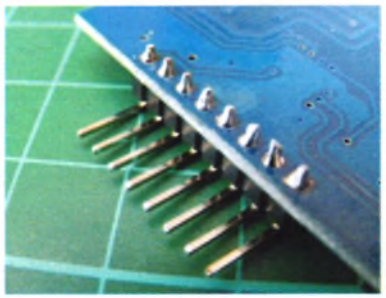


Рисунок 0.15

Качественная пайка должна выглядеть так

Обеспечение безопасности работы

Паяльник нагревается очень и очень сильно! Используйте паяльник с особой осторожностью. Не оставляйте паяльник в присутствии детей без присмотра! Обязательно соблюдайте следующие правила безопасности.

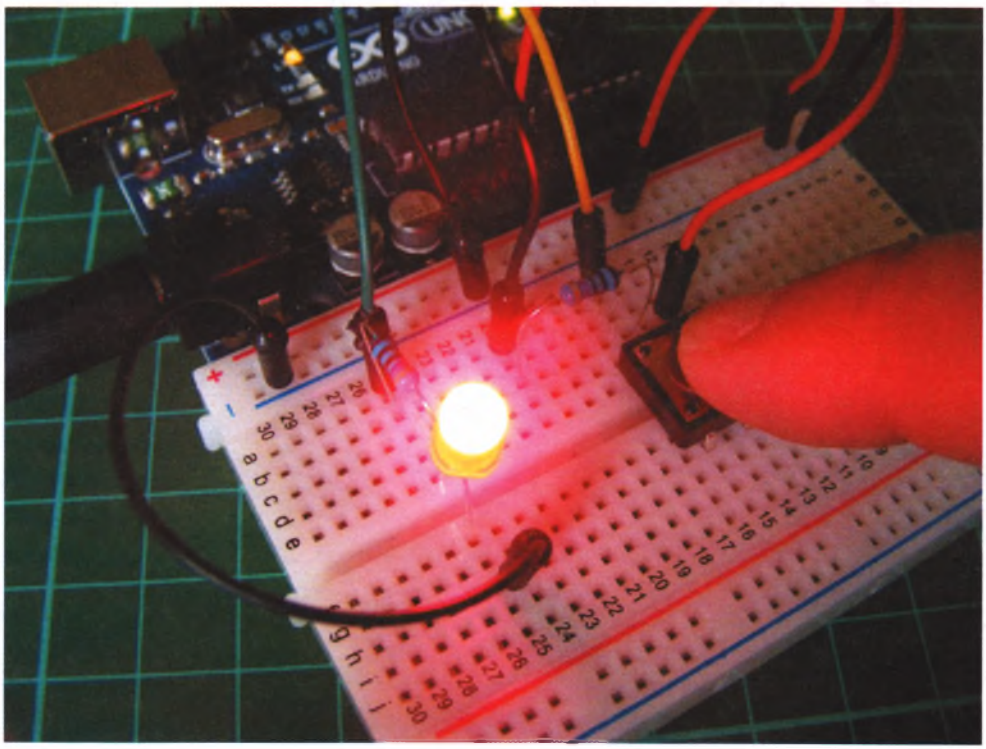
- Обязательно используйте подставку и никогда не кладите горячий паяльник на стол.
- Паяйте в хорошо проветриваемом помещении. Пары, выделяемые из плавильного припоя, могут быть вредными.
- Храните легковоспламеняющиеся материалы вдали от рабочей зоны.
- Держите оборудование в недоступных для детей местах.
- Надевайте защитные очки.
- Подождите, пока паяльник полностью остынет, прежде чем убирать его.

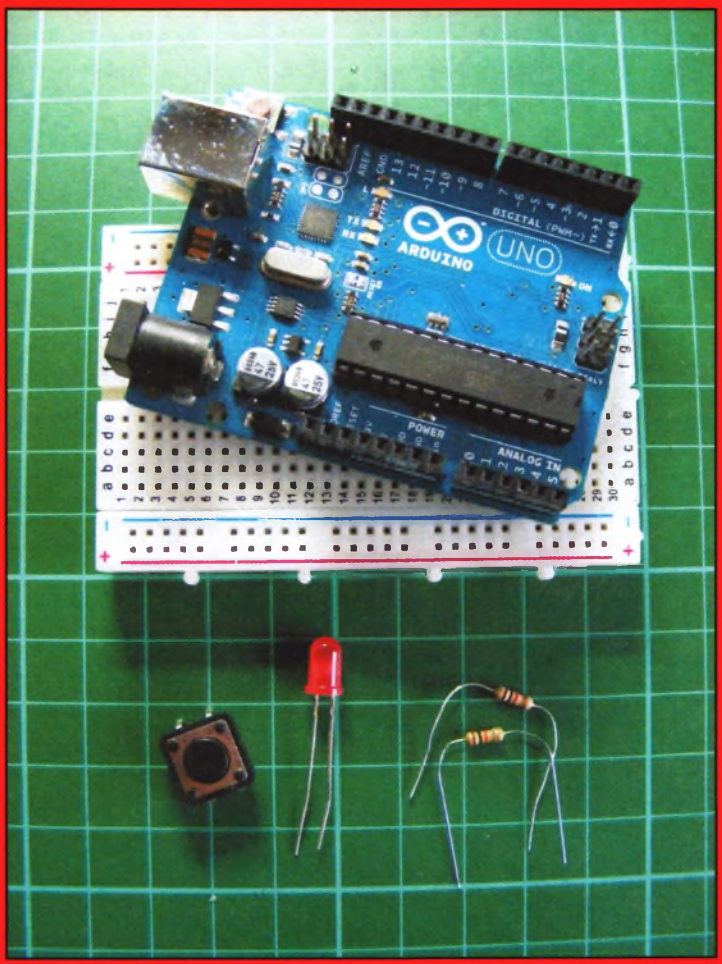
ЧАСТЬ 1

СВЕТ

ПРОЕКТ 1: УПРАВЛЯЕМЫЙ СВЕТОДИОД

В ЭТОМ ПРОЕКТЕ ВЫ ДОБАВИТЕ В ЦЕПЬ СО СВЕТОДИОДОМ ТАКТОВУЮ КНОПКУ, ЧТОБЫ УПРАВЛЯТЬ ЕГО СВЕЧЕНИЕМ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Четырехконтактный кнопочный переключатель
- Светодиод
- Резистор с сопротивлением 10 кОм
- Резистор с сопротивлением 220 Ом

В этом проекте вы познакомитесь с принципами переключателей, которые будете использовать в этой книге и дальше. Практически все электрические устройства содержат переключатели для включения и выключения питания. Существует множество видов переключателей, и сейчас вы будете использовать один из них — кнопочный (рис. 1.1).

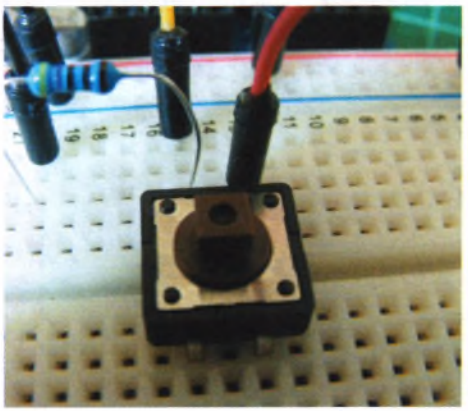


РИСУНОК 1.1

Кнопочный переключатель или кнопка

ПРИНЦИП РАБОТЫ

При нажатии кнопка замыкает цепь, подавая питание. Как только кнопка отпущена, она размыкает цепь, прерывая питание. Такие кнопочные переключатели также известны как *моментального действия* или *нормально открытые* и используются, к примеру, в компьютерных клавиатурах. Кнопки отличаются от *тумблеров*, которые остаются во включенном/выключенном состоянии, пока вы не переключите их в другое положение, как, например, выключатель освещения.

Наша кнопка имеет четыре контакта (ножки), но для подключения обычно используются только два из них. В этом проекте вы будете использовать верхние ножки (две неиспользуемые нижние ножки выполняют ту же функцию). На рис. 1.2 показано, как контакты работают в цепи. Контакты A и C всегда замкнуты, так же как B и D. При нажатии кнопки цепь замыкается.

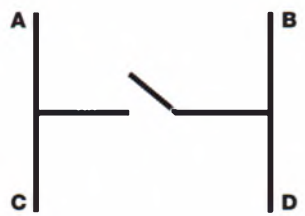


РИСУНОК 1.2

Схема разомкнутой кнопки

СБОРКА

1. Установите кнопку на макетную плату, как показано на рис. 1.3.

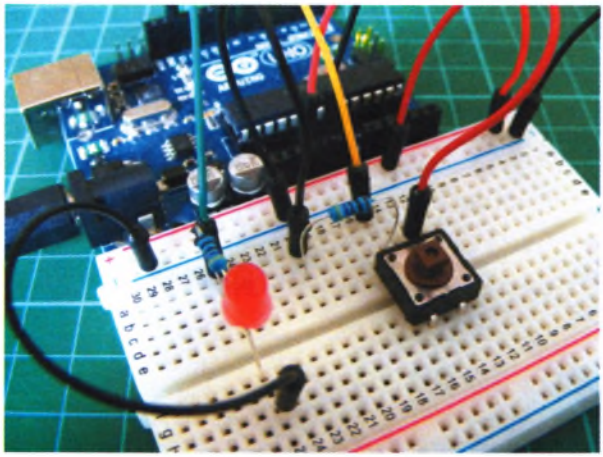


РИСУНОК 1.3

Установка кнопки на макетную плату

2. Соедините ножку A кнопки с одной из ножек резистора с сопротивлением 10 кОм и подключите эту же ножку резистора к контакту 2 платы Arduino. Подключите другую ножку резистора к шине заземления, а саму шину — к контакту **GND** платы Arduino. Подключите ножку B кнопки к шине питания +5 В, а шину, в свою очередь, — к контакту **5V** платы Arduino.

КНОПКА	ARDUINO
Контакт A	Контакт GND и контакт 2 через резистор с сопротивлением 10 кОм
Контакт B	Контакт 5V

3. Установите светодиод на макетную плату, подключив длинную ножку анода к контакту 13 платы Arduino через резистор с сопротивлением 220 Ом, а короткую ножку — к заземлению (GND).

СВЕТОДИОД	ARDUINO
Длинная ножка (анод)	Контакт 13 через резистор с сопротивлением 220 Ом
Короткая ножка (катод)	Контакт GND

4. Убедитесь, что ваша цепь соответствует схеме на рис. 1.4, а затем загрузите код, приведенный в разделе «Скетч» далее в этом проекте.

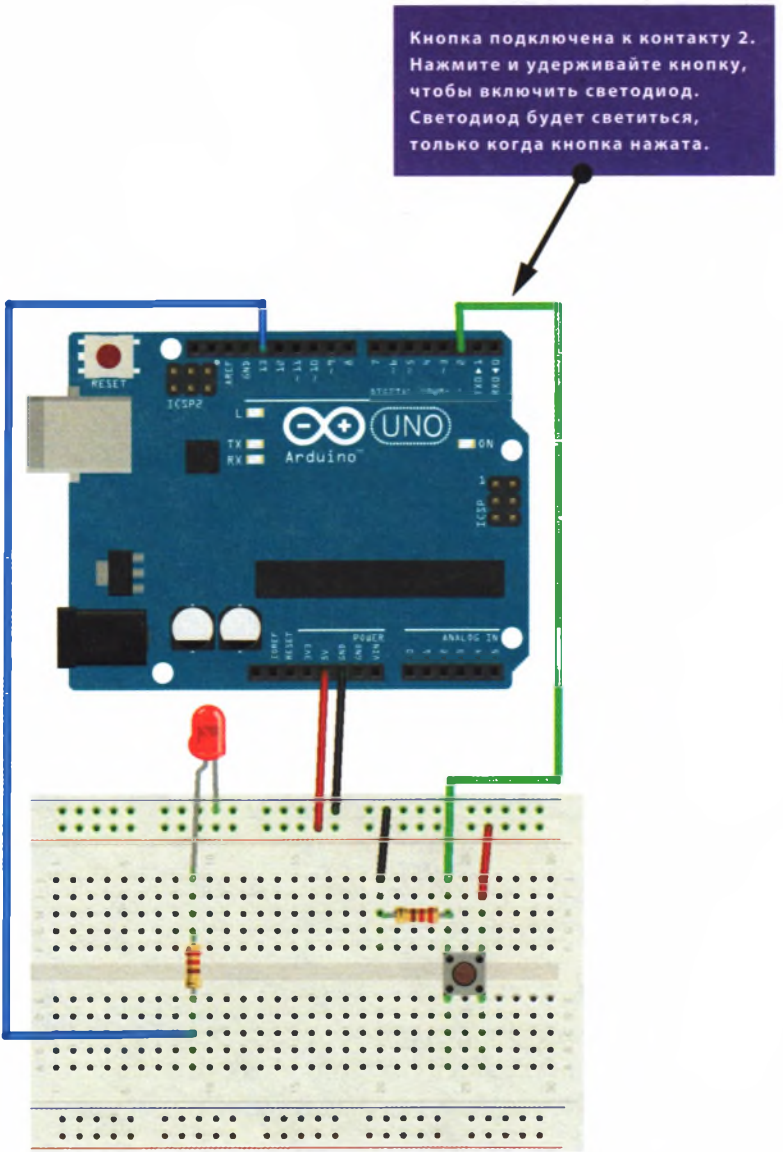


РИСУНОК 1.4

Принципиальная схема светодиода, управляемого с помощью кнопки

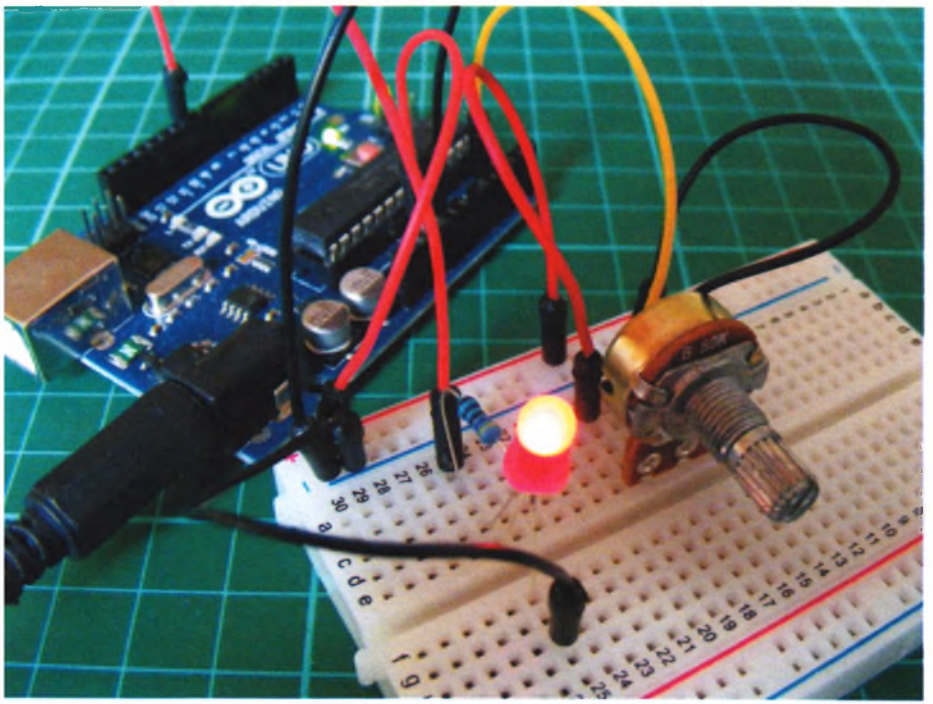
СКЕТЧ

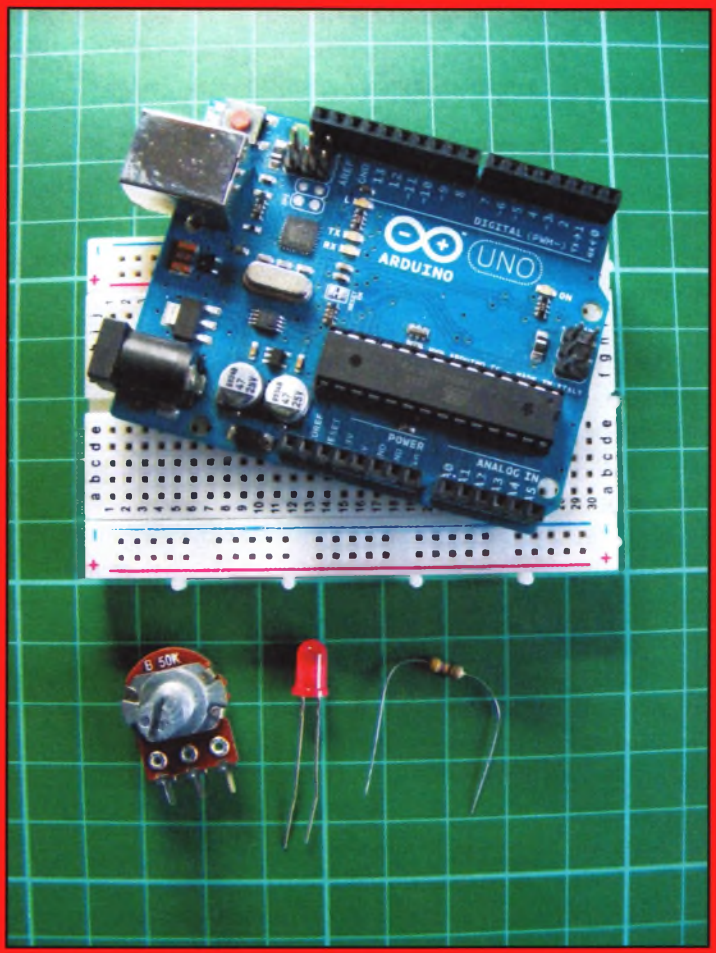
В этом скетче вы определяете контакт кнопки и переводите его в режим ввода — INPUT, а контакт светодиода — в режим вывода — OUTPUT. Код инструктирует Arduino включить светодиод, пока нажата кнопка (замыкающая цепь), и держать светодиод отключенным, если кнопка не нажата. При отпускании кнопки цепь размыкается и светодиод отключается.

```
/*Создан DojoDave <http://www.0j0.org>
   изменен 30 августа 2011 Томом Иго.
   Этот код - общественное достояние.
   http://www.arduino.cc/en/Tutorial/Button
*/

const int buttonPin = 2;       // Контакт, к которому подключена кнопка
const int ledPin = 13;         // Контакт, к которому подключен светодиод
int buttonState = 0;           // Задание значения кнопки

void setup() {
   pinMode(ledPin, OUTPUT);    // Перевод контакта светодиода в режим вывода
   pinMode(buttonPin, INPUT); // Перевод контакта кнопки в режим ввода
}

void loop() {
   buttonState = digitalRead(buttonPin);      // Чтение ввода с контакта 2
   if (buttonState == HIGH) {                 // Если кнопка нажата, уста-
                                              // навливается состояние HIGH
      digitalWrite(ledPin, HIGH);             // Включение светодиода
   }
   else {
      digitalWrite(ledPin, LOW);              // В противном случае
                                              // отключаем светодиод
   }
}
```

ПРОЕКТ 2: ДИММЕР ОСВЕЩЕНИЯ

В ЭТОМ ПРОЕКТЕ ВЫ СОЗДАДИТЕ СВЕТОРЕГУЛЯТОР ТИПА «ДИММЕР», ПРИМЕНИВ ПОТЕНЦИОМЕТР ДЛЯ УПРАВЛЕНИЯ ЯРКОСТЬЮ СВЕТОДИОДА.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Светодиод
- Потенциометр с сопротивлением 50 кОм
- Резистор с сопротивлением 470 Ом

Потенциометр представляет собой переменный резистор с рукояткой, поворачивая которую вы изменяете сопротивление потенциометра. Такие переменные резисторы широко используется в электрических устройствах, например, для управления уровнем громкости в звуковой аппаратуре. В этом проекте используется потенциометр с сопротивлением 50 кОм.

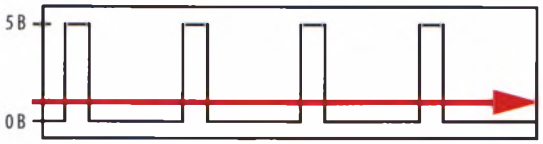
ПРИНЦИП РАБОТЫ

Потенциометр управляет постоянным *аналоговым* сигналом. Люди воспринимают мир в аналоговом режиме; все, что мы видим и слышим — это непрерывная передача информации нашим органам чувств. Этот непрерывный поток и есть аналоговые данные. С другой стороны, цифровая информация представляет аналоговые данные, используя исключительно числа. Чтобы принять непрерывные аналоговые данные, поступающие от потенциометра, компьютер Arduino должен представить сигнал как серию дискретных чисел — в данном случае, напряжения. Центральный контакт потенциометра посылает сигнал на аналоговый вход платы Arduino, любой контакт с метками от A0 до A5, для считывания значения.

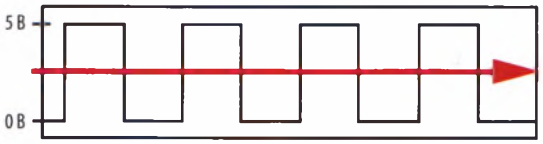
По сути, светодиод просто включается и выключается, но это происходит так быстро, что благодаря компенсации наши глаза видят постоянное свечение светодиода, но с разным уровнем освещенности. Этот эффект известен под термином *инерция зрения*.

Чтобы создать этот эффект, в компьютере Arduino используется технология, называемая *широтно-импульсной модуляцией (ШИМ)*. Arduino создает импульсы, очень быстро включая и выключая питание. Длительность включения и отключения питания (так называемая *ширина импульса*) в цикле определяет *средний уровень выхода*, и, изменяя значение ширины импульса, устройство может симулировать значения напряжения в диапазоне от 5 до 0 В (т.е. от включенного до отключенного состояния). Если питание от Arduino подается в течение половины времени и отключается также на половину времени, средний уровень выхода составит 2,5 В, т.е. по центру между 0 и 5 В. Если сигнал подается 80% всего времени, а отключен в течение 20%, средний уровень выхода составит 4 В, и т.д. Вы можете изменять сигнал, который, в свою очередь, меняет ширину импульса, поворачивая ручку потенциометра влево или вправо, увеличивая или уменьшая сопротивление.

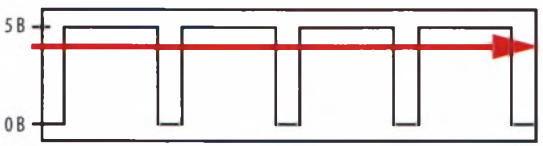
Используя этот метод, вы можете изменить напряжение, посылаемое на светодиод, и сделать его свечение более тусклым или ярким, соответствующим аналоговому сигналу от потенциометра. Только контакты 3, 5, 6, 9, 10 или 11 платы Arduino поддерживают ШИМ. На рис. 2.1 показаны примеры работы ШИМ в виде формы сигнала.



Высокий сигнал (5 В) в течение 20% времени и низкий (0 В) в течение 80% времени, поэтому средний уровень выходного напряжения (красная линия) будет составлять 0,2 × 5 В = 1 В.



Высокий сигнал 50% и низкий сигнал 50%, поэтому средний уровень выходного напряжения будет составлять 0,5 × 5 В = 2,5 В.



Высокий сигнал 80% и низкий сигнал 20%, поэтому средний уровень выходного напряжения будет составлять 0,8 × 5 В = 4 В.

РИСУНОК 2.1

Широтно-импульсная модуляция в виде формы сигнала

СБОРКА

1. Установите потенциометр на макетную плату и подключите центральный контакт потенциометра к контакту A0 платы Arduino. Подключите один из внешних контактов потенциометра к положительной шине 5 В, а другой — к заземлению (GND) на макетной плате (не имеет значения, в каком порядке подключены внешние контакты потенциометра; эти инструкции отражают вариант подключения, продемонстрированный в этой книге), как показано на рис. 2.2.

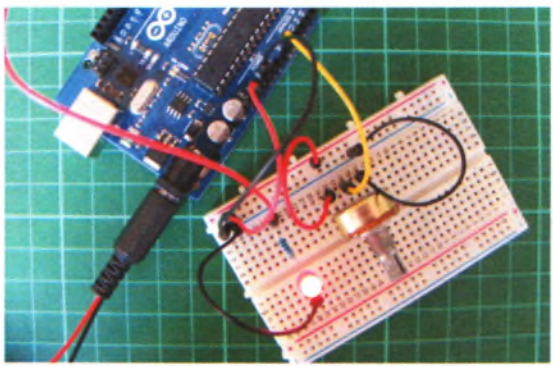


РИСУНОК 2.2

Подключение потенциометра к Arduino

ПОТЕНЦИОМЕТР	ARDUINO
Левый контакт	Контакт 5V
Центральный контакт	Контакт A0
Правый контакт	Контакт GND

2. Установите светодиод на макетную плату. Подключите ножку анода (длинную) к контакту 9 платы Arduino через резистор с сопротивлением 470 Ом, а ножку катода (короткую) — к контакту GND, как показано на рис. 2.3.

СВЕТОДИОД	ARDUINO
Длинная ножка (анод)	Контакт 9
Короткая ножка (катод)	Контакт GND через резистор с сопротивлением 470 Ом

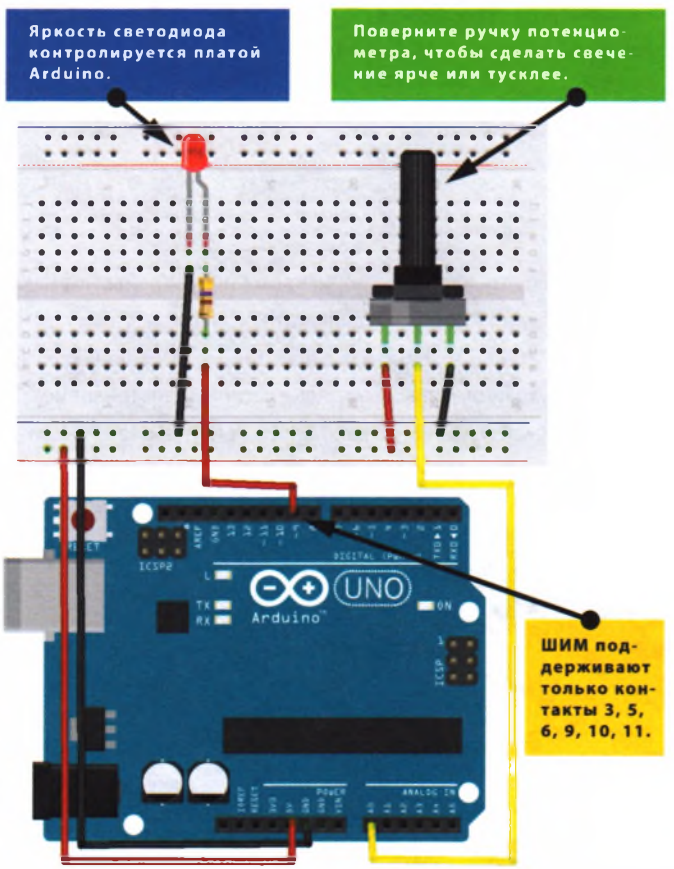


РИСУНОК 2.3

Принципиальная схема диммера освещения

3. Загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.
4. Поворачивайте ручку потенциометра для управления яркостью светодиода.

У этого проекта есть множество потенциальных возможностей для применения: вы можете сгруппировать несколько светодиодов вместе, чтобы создать регулируемый фонарик, ночник, подсветку для выставочного стенда или витрины либо какое-нибудь другое устройство с приглушенным светом.

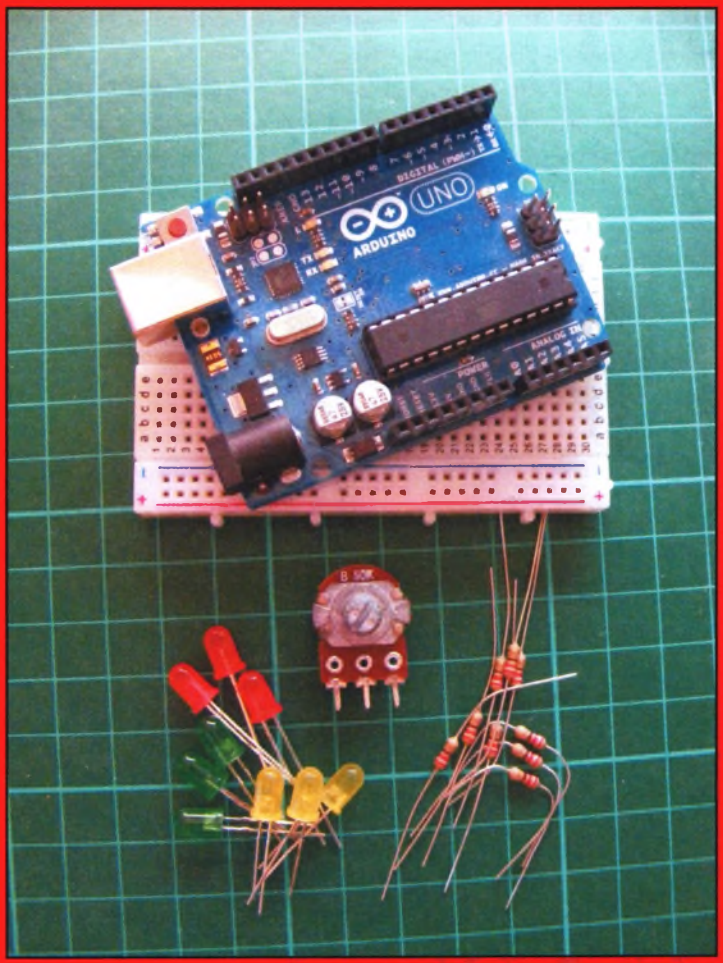
СКЕТЧ

При работе этого скетча контакт A0 настроен на работу с потенциометром, а контакт 9 служит выводом для питания светодиода. Затем запускается цикл, который постоянно считывает значение с потенциометра и использует его для установки напряжения, подаваемого на светодиод. Напряжение колеблется в пределах от 0 до 5 В, согласно значению которого устанавливается соответствующая яркость светодиода.

```
/* http://arduino.cc/en/Reference/AnalogWrite,
    Том Иго, itp.nyu.edu/physcomp/Labs/AnalogIn */

int potPin = A0;      // Контакт аналогового входа, подключенный
                      // к потенциометру
int potValue = 0;     // Значение, считываемое с потенциометра
int led = 9;          // Контакт 9 (подключен к светодиоду)
                      // поддерживает ШИМ

// Выполняется однократно в начале программы
void setup() {
   pinMode(led, OUTPUT);     // Перевод контакта 9 в режим вывода
}

// Непрерывное выполнение
void loop() {
   potValue = analogRead(potPin);   // Чтение значения потенциометра
                                    // с контакта A0
   analogWrite(led, potValue/4);    // Отправка значения потенциометра
                                    // на светодиод для управления
                                    // яркостью с помощью ШИМ
   delay(10);                       // Ожидание 10 мс
}
```

ПРОЕКТ 3: СВЕТОДИОДНАЯ ПАНЕЛЬ

В ЭТОМ ПРОЕКТЕ ВЫ ПРИМЕНИТЕ ЗНАНИЯ ИЗ ПРЕДЫДУЩИХ ПРОЕКТОВ И СОЗДАДИТЕ СВЕТОДИОДНУЮ ПАНЕЛЬ, УПРАВЛЯЕМУЮ ПОТЕНЦИОМЕТРОМ.



ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- 9 светодиодов
- Потенциометр с сопротивлением 50 кОм
- 9 резисторов с сопротивлением 220 Ом

ПРИНЦИП РАБОТЫ

Устройство из этого проекта представляет собой ряд светодиодов в виде линии. Нечто подобное вы могли видеть в виде спектроанализатора на аудиотехнике и в аудиопрограммах. Панель состоит из ряда светодиодов с аналоговым входом, например потенциометром или микрофоном. В этом проекте вы будете подавать аналоговый сигнал от потенциометра для управления светодиодами. Когда вы поворачиваете ручку потенциометра в одну сторону, светодиоды загораются по очереди, как показано на рис. 3.1 (а), пока они не будут включены все, как показано на рис. 3.1 (б). Когда вы поворачиваете его в другую сторону, они поочередно выключаются, как показано на рис. 3.1 (в).

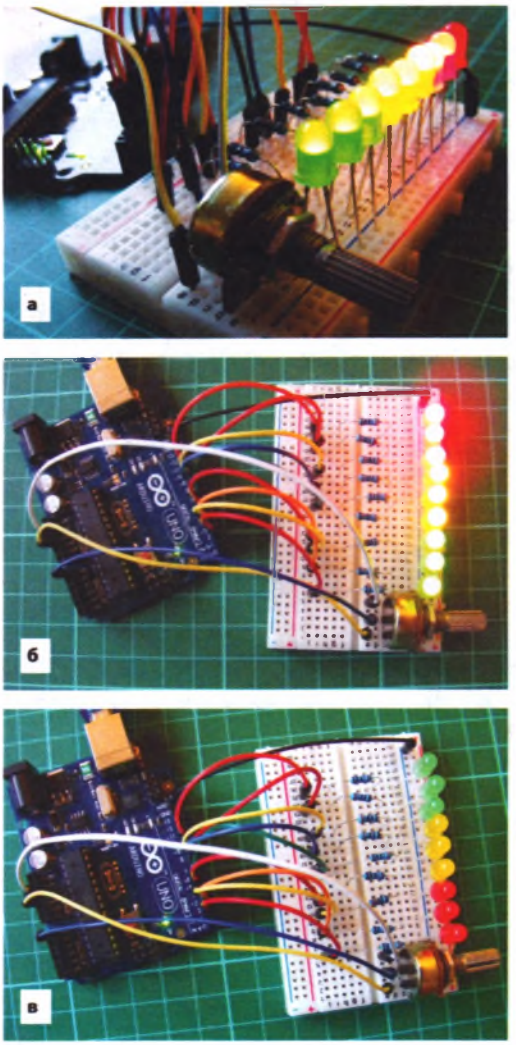


РИСУНОК 3.1

Светодиоды включаются/выключаются в соответствии с поворотом ручки потенциометра

СБОРКА

1. Установите светодиоды на макетную плату, при этом короткие ножки (катоды) должны быть подключены к шине заземления. Подключите шину заземления к контакту GND платы Arduino с помощью перемычки.
2. Установите на макетную плату по одному резистору с сопротивлением 220 Ом на каждый светодиод, одна ножка резистора должна быть соединена с длинной ножкой (анодом) светодиода. Другую ножку каждого резистора подключите к цифровым контактам 2–10 соответственно, как показано на рис. 3.2. Важно, чтобы резисторы при подключении как бы образовали «мостики» через канавку макетной платы, как показано на рисунке.

СВЕТОДИОДЫ	ARDUINO
Длинные ножки (аноды)	Контакты 2–10 через резисторы с сопротивлением 220 Ом
Короткие ножки (катоды)	Контакт GND

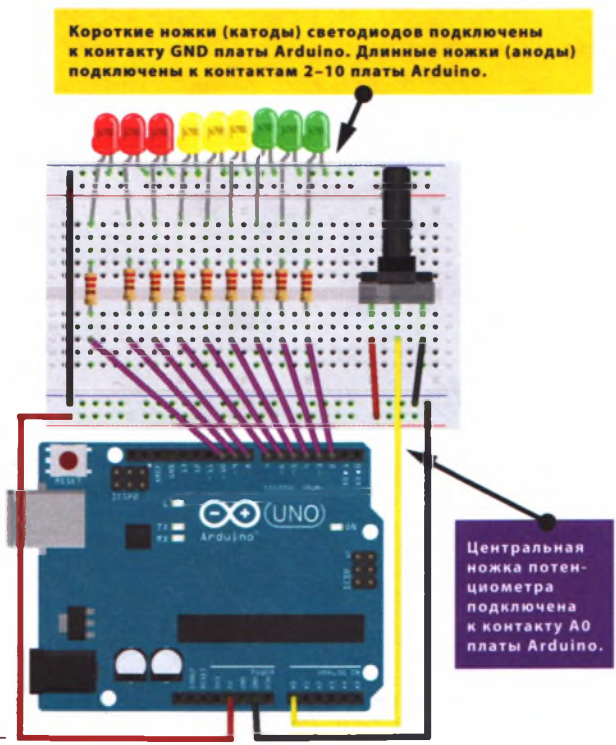


РИСУНОК 3.2

Принципиальная схема светодиодной панели

ПРИМЕЧАНИЕ

Как упоминалось в проекте 2, на самом деле не имеет значения, каким образом подключаются внешние контакты потенциометра. В тексте я привел инструкции в соответствии с показанным изображением.

3. Установите потенциометр на макетную плату и подключите его центральную ножку потенциометра к контакту A0 платы Arduino. Подключите правую внешнюю ножку потенциометра к контакту 5V платы Arduino, а левую потенциометра — к контакту GND платы Arduino.

ПОТЕНЦИОМЕТР	ARDUINO
Левый контакт	Контакт GND
Центральный контакт	Контакт A0
Правый контакт	Контакт 5V

4. Загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

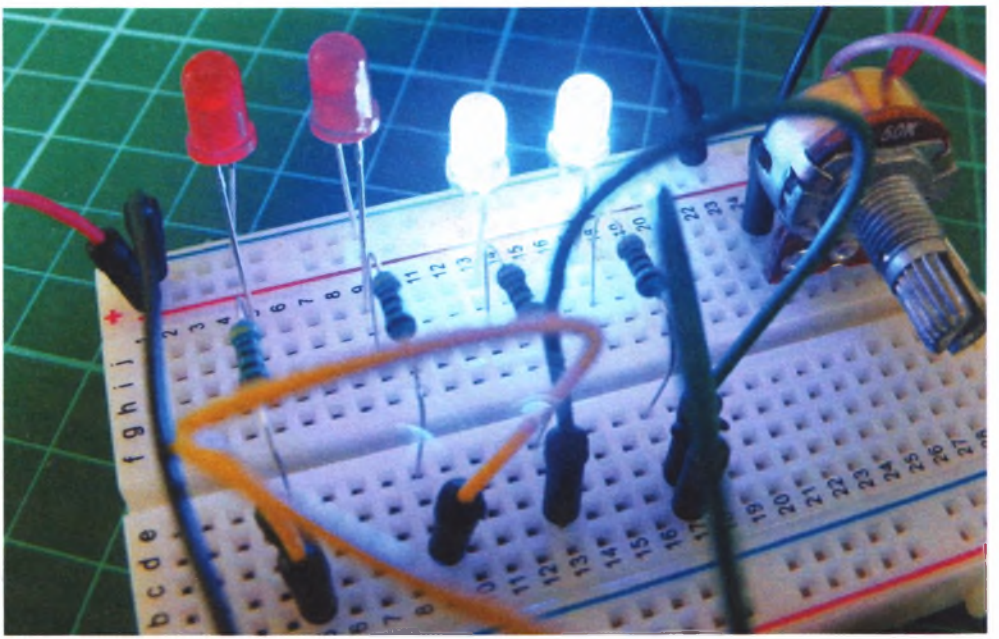
СКЕТЧ

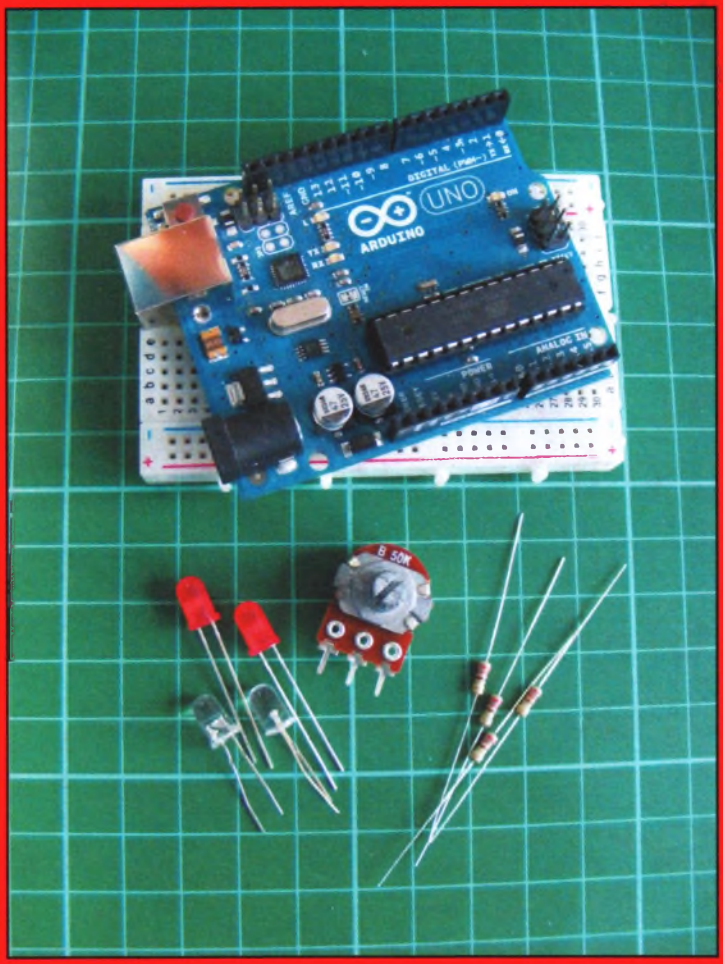
Сначала скетч считывает данные с потенциометра. Код распределяет входное значение в виде выходного диапазона, в данном случае на девять светодиодов. Затем запускается цикл for для подачи питания на выводы. Если номер выхода светодиода на панели меньше, чем значение в диапазоне, светодиод включается; если больше — отключается. Как видите, все просто. Если вы повернете ручку потенциометра вправо, светодиоды будут последовательно загораться. Повернете влево — гаснуть один за другим.

```
/* Создан Томом Иго. Этот код - общественное достояние.
    http://www.arduino.cc/en/Tutorial/BarGraph */

const int analogPin = A0;  // Контакт, к которому подключен потенциометр
const int ledCount = 9;    // Количество светодиодов
int ledPins[] = {2,3,4,5,6,7,8,9,10};       // Контакты, к которым
                                            // подключены светодиоды

void setup() {
   for (int thisLed = 0; thisLed < ledCount; thisLed++) {
      pinMode(ledPins[thisLed], OUTPUT);     // Перевод контактов,
                     // к которым подключены светодиоды, в режим вывода
   }
}

// Начало цикла
void loop() {
   int sensorReading = analogRead(analogPin);       // Аналоговый вход
   int ledLevel = map(sensorReading, 0, 1023, 0, ledCount);
   for (int thisLed = 0; thisLed < ledCount; thisLed++) {
      if (thisLed < ledLevel) {                  // Последовательное вклю-
         digitalWrite(ledPins[thisLed], HIGH);  // чение светодиодов
      }
      else {           // Последовательное выключение светодиодов
         digitalWrite(ledPins[thisLed], LOW);
      }
   }
}
```

ПРОЕКТ 4: ДИСКОТЕЧНЫЙ СТРОБОСКОП

В ЭТОМ ПРОЕКТЕ ВЫ ПРИМЕНИТЕ ЗНАНИЯ, ПОЛУЧЕННЫЕ В ПРОЕКТЕ 3, ЧТОБЫ СОЗДАТЬ СТРОБОСКОП С ВОЗМОЖНОСТЬЮ УПРАВЛЕНИЯ СКОРОСТЬЮ МЕРЦАНИЯ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- 2 голубых светодиода
- 2 красных светодиода
- Потенциометр с сопротивлением 50 кОм
- 4 резистора с сопротивлением 220 Ом

ПРИНЦИП РАБОТЫ

Поворот ручки потенциометра вправо/влево изменяет скорость мигания светодиодов, создавая стробоскопический эффект. Вы можете использовать красные и синие светодиоды для создания эффекта полицейских проблесковых маячков (см. рис. 4.1). Чтобы светодиоды одного и того же цвета светились вместе, подключите их к одному контакту на плате Arduino. А если вы сделаете корпус и поместите светодиоды внутрь, у вас получится собственный мобильный стробоскоп! Вы можете установить до 10 светодиодов; понадобится лишь внести минимальные изменения в скетч, добавив информацию о дополнительных светодиодах и выходах.

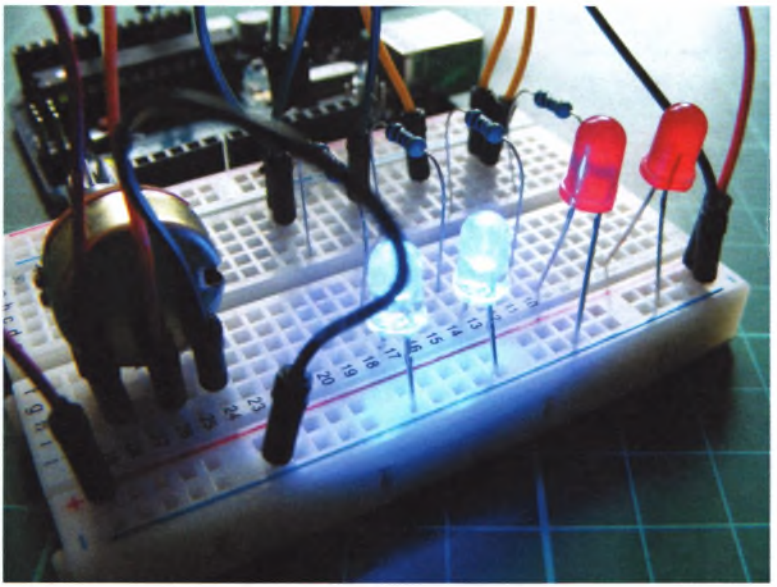


РИСУНОК 4.1

Красные и синие светодиоды имитируют свечение полицейских проблесковых маячков

СБОРКА

1. Установите светодиоды на макетную плату, подключив короткие ножки (катоды) к шине заземления, а затем к контакту GND платы Arduino.
2. Установите резисторы на макетную плату, подключив их к длинным ножкам (анодам) светодиодов. Используя перемычки, подключите два красных светодиода к одному контакту на плате Arduino, а два синих светодиода — к другому. Подключение к плате Arduino осуществляется после резисторов, как показано на рис. 4.2. Так вы сможете управлять светодиодами одного цвета через один контакт.

ПРИМЕЧАНИЕ

Не забудьте подать напряжение на макетную плату.

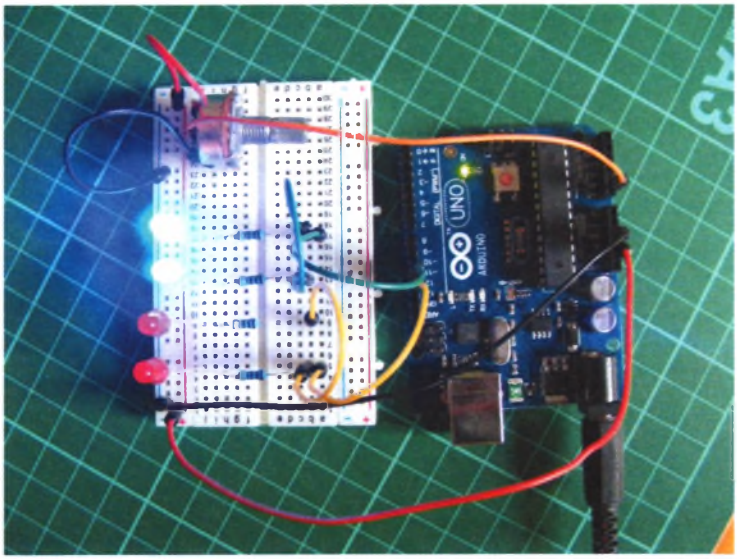


РИСУНОК 4.2

Подключение светодиодов с помощью перемычек

3. Подключите красные светодиоды к контакту 12 на плате Arduino, а синие — к контакту 11 на плате Arduino.

СВЕТОДИОДЫ	ARDUINO
Короткие ножки (катоды)	Контакт GND
Длинные ножки (аноды) (красные светодиоды)	Контакт 12
Длинные ножки (аноды) (синие светодиоды)	Контакт 11

4. Установите потенциометр на макетную плату и подключите центральную ножку к контакту A0 на плате Arduino, левую — к контакту GND, а правую — к контакту 5V.

ПОТЕНЦИОМЕТР	ARDUINO
Левый контакт	Контакт GND
Центральный контакт	Контакт A0
Правый контакт	Контакт 5V

5. Убедитесь, что ваша цепь соответствует схеме на рис. 4.3, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

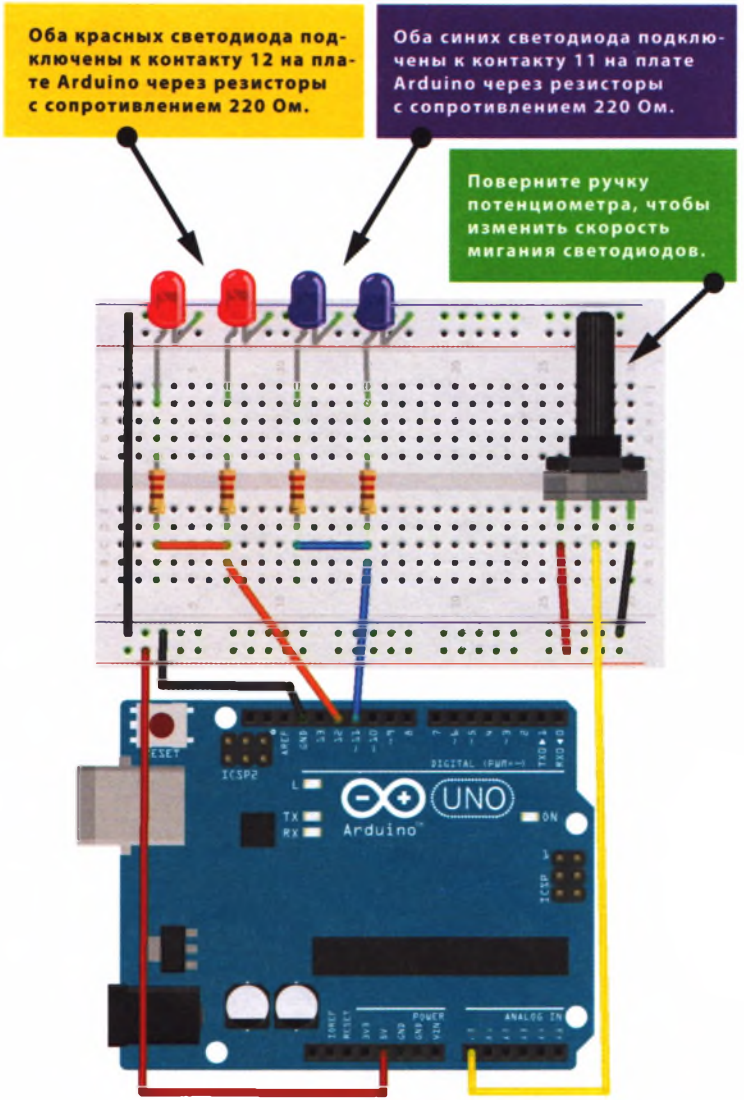


РИСУНОК 4.3

Принципиальная схема дискотечного стробоскопа

СКЕТЧ

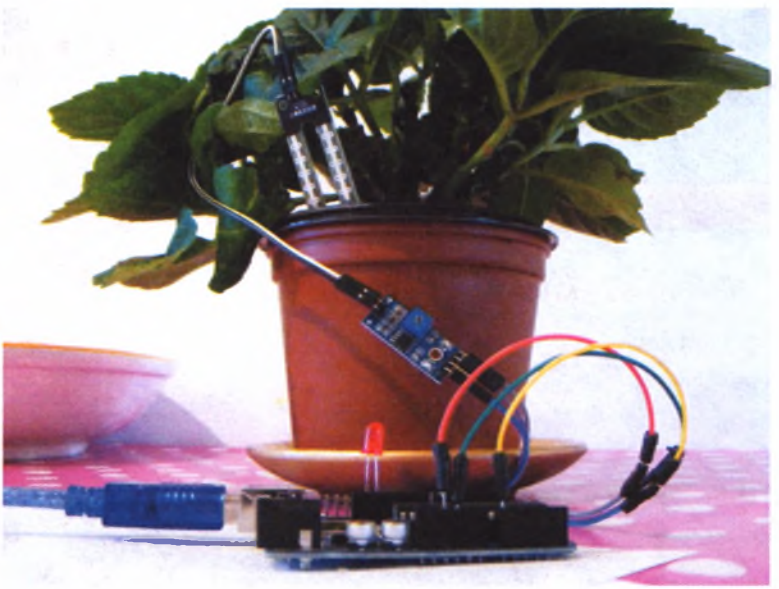
Скетч начинает работу с настройки контакта Arduino, принимающего аналоговый сигнал с потенциометра, на режим ввода, а контактов, к которым подключены светодиоды, на режим вывода. Микрокомпьютер Arduino считывает аналоговый сигнал, поступающий от потенциометра, и использует полученные данные в качестве *значения задержки* — количества времени, которое проходит до смены светодиодами своего состояния (включены или выключены). Это означает, что светодиоды

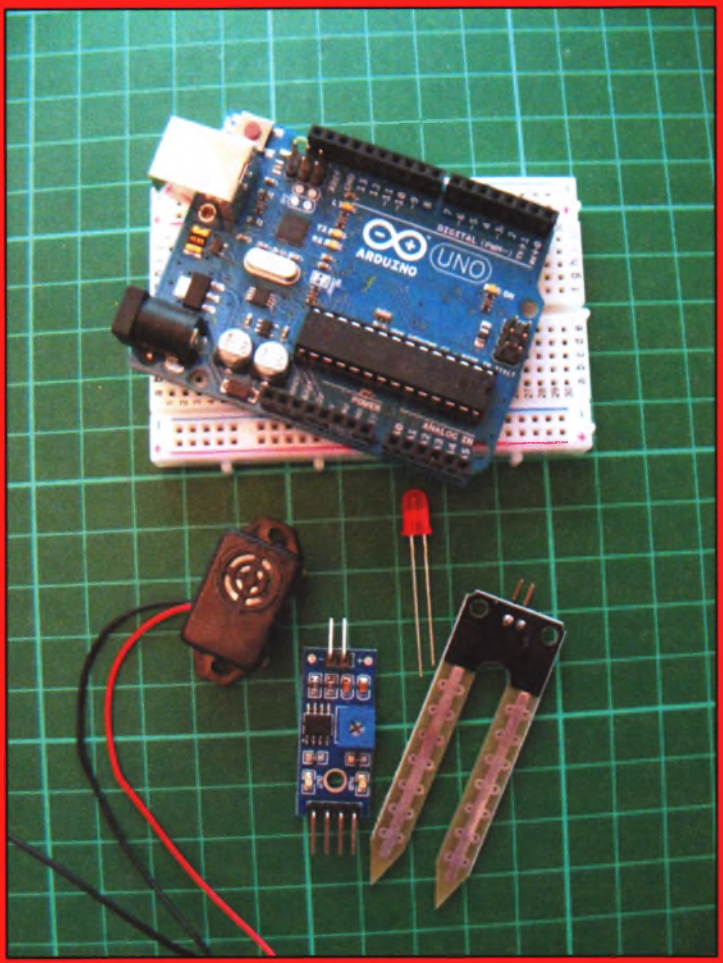
включаются и выключаются с частотой, определяемой потенциометром, поэтому изменение его значения влияет на скорость мигания. Скетч в цикле обрабатывает все светодиоды для имитации стробоскопического эффекта.

```
const int analogInPin = A0;           // Аналоговый контакт, к которому
                                      // подключен потенциометр
int sensorValue = 0;                  // Значение, полученное
                                      // от потенциометра
int timer = 0;                        // Задержка

// Перевод контактов 11 и 12 в режим вывода
void setup() {
   pinMode(12, OUTPUT);
   pinMode(11, OUTPUT);
}

// Начало цикла, включающего/выключающего
// светодиоды с настраиваемой задержкой
void loop() {
   sensorValue = analogRead(analogInPin);     // Считывание значения
                                              // от потенциометра
   timer = map(sensorValue, 0, 1023, 10, 500);// Задержка 10 - 500 мс
   digitalWrite(12, HIGH);                    // Подача напряжения
                                              // на светодиоды
   delay(timer);                              // Задержка в зависимости
                                              // от значения потенциометра
   digitalWrite(12, LOW);                     // Прекращение подачи
                                              // напряжения на светодиоды
   delay(timer);
   digitalWrite(12, HIGH);
   delay(timer);
   digitalWrite(12, LOW);
   digitalWrite(11, HIGH);
   delay(timer);
   digitalWrite(11, LOW);
   delay(timer);
   digitalWrite(11, HIGH);
   delay(timer);
   digitalWrite(11, LOW);
}
```

ПРОЕКТ 5: ПРИБОР ДЛЯ КОНТРОЛЯ ПОЛИВА

В ЭТОМ ПРОЕКТЕ ВЫ ОСВОИТЕ РАБОТУ С АНАЛОГОВЫМ ДАТЧИКОМ НОВОГО ТИПА, ОПРЕДЕЛЯЮЩИМ УРОВЕНЬ ВЛАЖНОСТИ.
ВЫ СКОНСТРУИРУЕТЕ СИСТЕМУ, КОТОРАЯ БУДЕТ СВЕТОМ И ЗВУКОМ ОПОВЕЩАТЬ ВАС, КОГДА РАСТЕНИЕ НУЖДАЕТСЯ В ПОЛИВЕ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Перемычки
- Светодиод
- Датчик влажности почвы HL-69
- Пьезоизлучатель

ПРИНЦИП РАБОТЫ

В этом проекте вы будете использовать датчик влажности почвы HL-69, его (или его аналог) можно приобрести за пару сотен рублей в Интернете или магазинах, перечисленных в приложении А. Штыри датчика определяют уровень влажности в окружающей почве, пропуская через нее ток и измеряя сопротивление. Влажная почва легко проводит электричество, имея более низкое сопротивление, в то время как сухая почва имеет более высокое сопротивление.

Датчик состоит из двух частей, как показано на рис. 5.1: собственно, сам датчик (а) и его контроллер (б). Два контакта на датчике подключаются к двум соответствующим контактам на одной стороне контроллера (как правило, специальный шлейф для подключения входит в комплект). Другая сторона контроллера содержит четыре контакта, три из которых подключаются к плате Arduino.

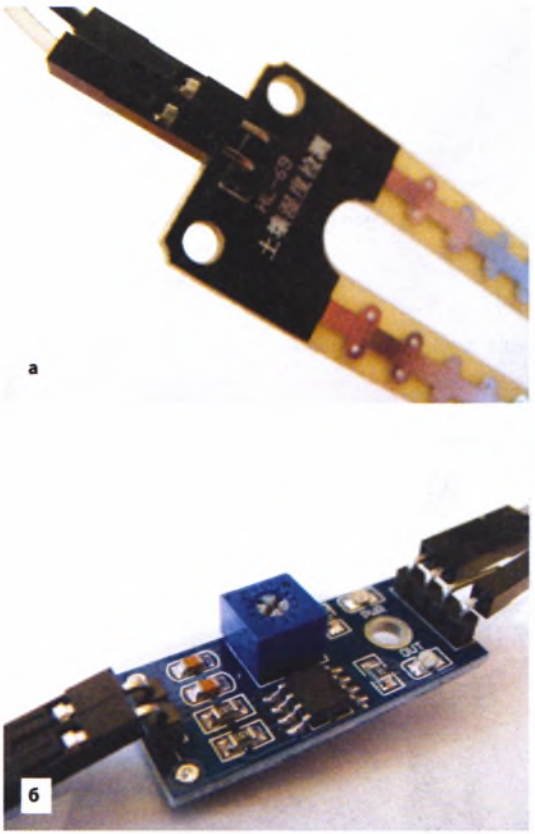


РИСУНОК 5.1

Комплект HL-69 — датчик влажности почвы (а) и контроллер (б)

Перечислю четыре контакта контроллера, слева направо: AO (аналоговый выход), DO (цифровой выход), GND (заземление) и VCC (питание) (см. рис. 5.2). Вы можете считывать значения, получаемые с контроллера, в среде разработки

Arduino, подключив контроллер с датчиком к компьютеру. В этом проекте нет необходимости в макетной плате, поэтому все компоненты подключаются к плате Arduino напрямую.

РИСУНОК 5.2

Обозначения контактов на оборотной стороне контроллера

Малые показатели датчика указывают на присутствие влаги в почве, а высокие — на ее сухость. Если замеры выдали значение, равное или превышающее 900, у вашего растения серьезный недостаток влаги. В этом случае загорится светодиод, и пьезоизлучатель издаст звук. Пьезоизлучатели — это недорогие излучатели звука, которые мы рассмотрим подробнее при работе над проектом № 7.

СБОРКА

1. С помощью входящего в комплект шлейфа подключите два контакта датчика к контактам + и — на плате контроллера, как показано на рис. 5.3.

РИСУНОК 5.3

Подключение датчика к контроллеру

2. Подключите три штыря на плате контроллера напрямую к контактам 5V, GND и A0 на плате Arduino. Схема подключения показана в таблице ниже. Контакт DO не используется.

КОНТРОЛЛЕР ДАТЧИКА	ARDUINO
Штырь VCC	Контакт 5V
Штырь GND	Контакт GND
Штырь A0	Контакт A0
Штырь DO	используется

3. Напрямую к плате Arduino подключите светодиод. Короткая ножка (катод) — к контакту GND, а длинная (анод) — к контакту 13, как показано на рис. 5.4 и в таблице ниже.

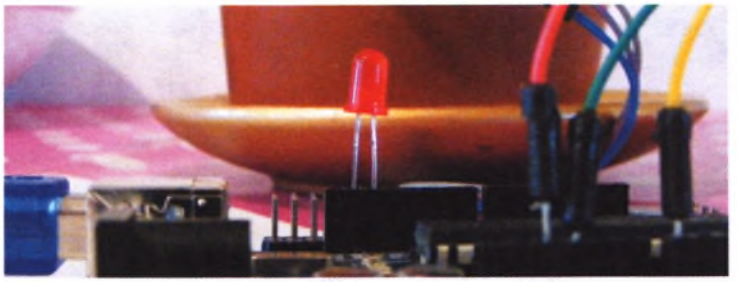


РИСУНОК 5.4

Подключение светодиода к плате Arduino

СВЕТОДИОД	ARDUINO
Длинная ножка (анод)	Контакт 13
Короткая ножка (катод)	Контакт GND

4. Подключите черный провод пьезоизлучателя к контакту GND, а красный провод — к контакту 11 платы Arduino.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Красный провод	Контакт 11
Черный провод	Контакт GND

5. Убедитесь, что ваша цепь соответствует схеме на рис. 5.5, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

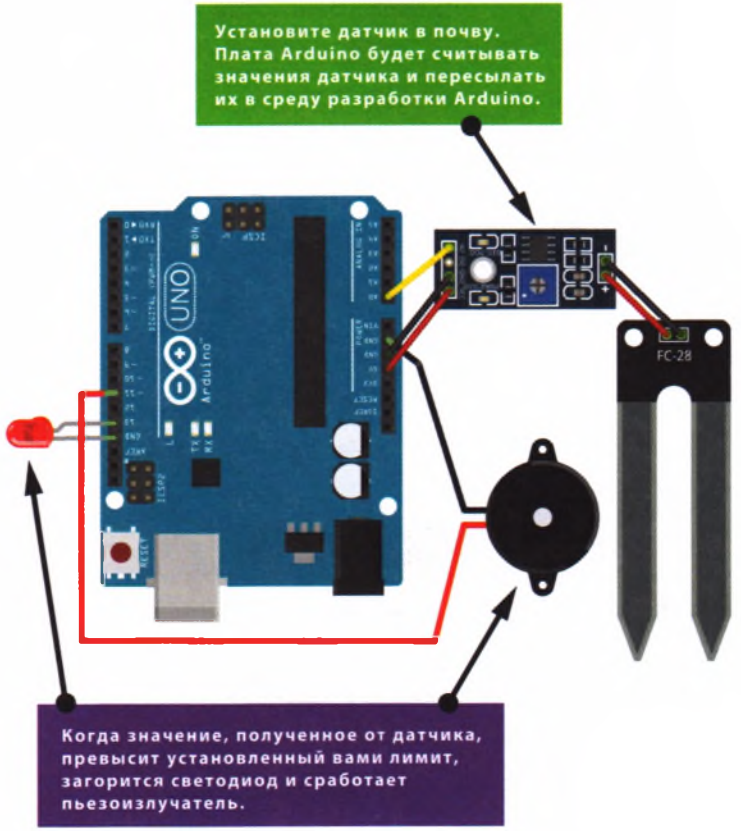


РИСУНОК 5.5

Принципиальная схема прибора для контроля полива

6. Подключите плату Arduino к компьютеру с помощью USB-кабеля. Откройте панель монитора порта в среде разработки Arduino, чтобы увидеть значения с датчика. Так вы сможете откалибровать устройство контроля полива вашего растения. В среде разработки отобразится актуальное показание датчика. В моем случае значение было равно 1000, если датчик не был установлен в почву. Так я вычислил значение минимальной влажности. Для калибровки этого значения вы можете поворачивать ручку потенциометра на контроллере по часовой стрелке для увеличения сопротивления и против часовой стрелки для уменьшения (см. рис. 5.5).

   Когда датчик установлен во влажную почву, значение снизится примерно до 400. По мере высыхания почвы значение датчика возрастает; когда оно достигнет 900, загорится светодиод и раздастся звуковой сигнал.

РИСУНОК 5.6

Поворачивайте ручку потенциометра, чтобы откалибровать устройство для контроля полива растения.

СКЕТЧ

Сначала скетч настраивает контакт A0 платы Arduino на считывание показаний датчика влажности. Затем он настраивает контакт 11 платы Arduino на вывод на пьезоизлучатель, а контакт 13 — на вывод на светодиод. Функция Serial.Println() используется для передачи показаний датчика в среду разработки Arduino, чтобы вы могли увидеть значение влажности на экране.

В своем проекте измените значение в строке

```
if(analogRead(0) > 900){
```

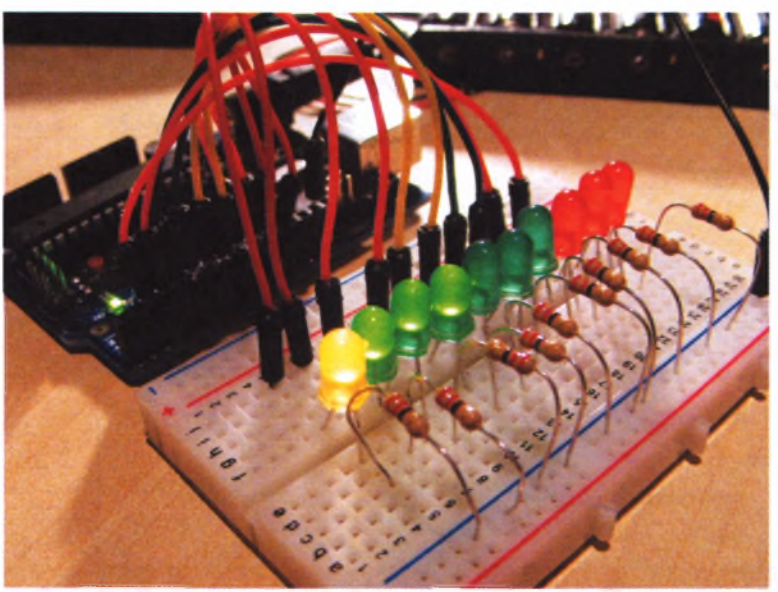
таким образом, чтобы оно соответствовало показанию сухого датчика (у меня получилось значение 900). Когда почва влажная, значение будет ниже 900, поэтому светодиод и пьезоизлучатель останутся выключенными. Когда значение достигнет 900, это будет означать, что почва высохла, о чем оповестят пьезоизлучатель и светодиод. Так вы узнаете, что растение необходимо полить.

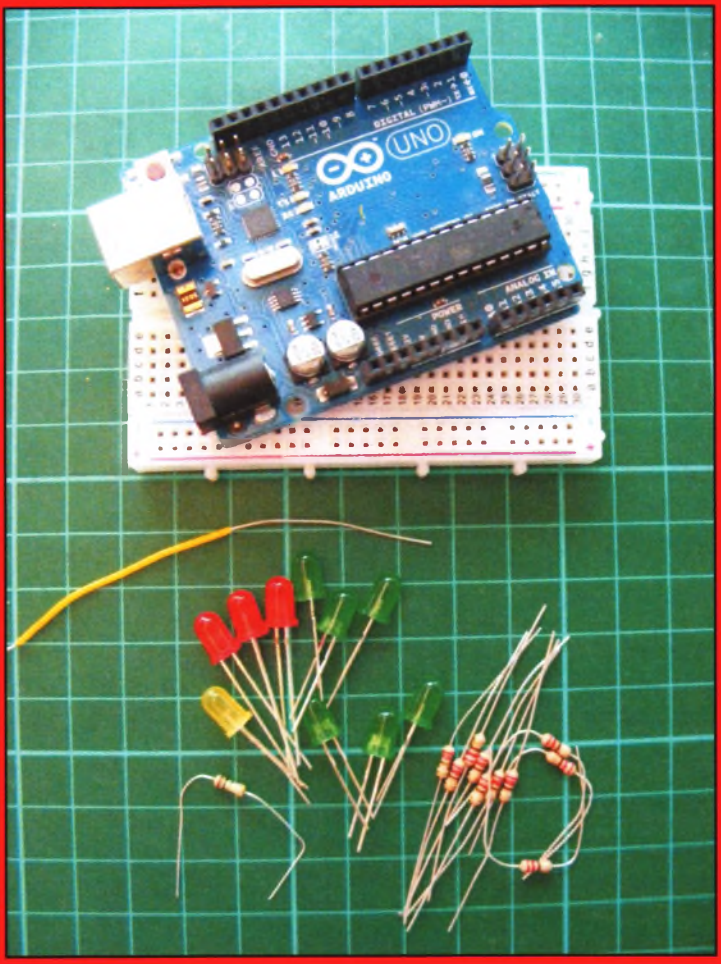
```
const int moistureAO = 0;
int AO = 0;                    // Контакт, к которому подключен штырь A0
                               // контроллера
int tmp = 0;                   // Значение аналогового контакта
int buzzPin = 11;              // Контакт, к которому подключен
                               // пьезоизлучатель
int LED = 13;                  // Контакт, к которому подключен светодиод

void setup () {
   Serial.begin(9600);         // Передача считываемых Arduino данных
                               // в IDE
   Serial.println("Soil moisture sensor");
   pinMode(moistureAO, INPUT);
```

```
    pinMode(buzzPin, OUTPUT);  // Перевод контакта в режим вывода
    pinMode(LED, OUTPUT);      // Перевод контакта в режим вывода
}

void loop () {
    tmp = analogRead( moistureAO );
    if ( tmp != AO ) {
        AO = tmp;
        Serial.print("A = ");   // Вывести значение сопротивления датчика
                               // в IDE
        Serial.println(AO);
    }
    delay (1000);
    if (analogRead(0) > 900) {         // Если показание датчика
                                       // превышает 900,
        digitalWrite(buzzPin, HIGH);   // подается питание
                                       // на пьезоизлучатель
        digitalWrite(LED, HIGH);       // и на светодиод
        delay(1000);                   // Ожидание в 1 секунду
        digitalWrite(buzzPin, LOW);
        digitalWrite(LED, HIGH);
    }
    else {
        digitalWrite(buzzPin, LOW);    // Если показание датчика меньше
                                       // 900, питание на пьезоизлучатель
                                       // и светодиод остаются выключенными
        digitalWrite(LED, LOW);
    }
}
```

ПРОЕКТ 6: ДЕТЕКТОР ПРИЗРАКОВ

ДАВАЙТЕ СОБЕРЕМ ДЕТЕКТОР ПРИЗРАКОВ! НЕ СМОТРИТЕ НА МЕНЯ КАК НА СУМАСШЕДШЕГО. ЭТО ДОВОЛЬНО ПРОСТОЙ ПРОЕКТ, НА СБОРКУ КОТОРОГО УЙДЕТ НЕМНОГО ВРЕМЕНИ, ПОСЛЕ ЧЕГО ВЫ СРАЗУ СМОЖЕТЕ ПРИСТУПИТЬ К ПОИСКУ ПРИВИДЕНИЙ. ВАМ СТРАШНО? МНЕ — НЕТ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- 3 красных светодиода
- 1 желтый светодиод
- 6 зеленых светодиодов
- 10 резисторов с сопротивлением 220 Ом
- 20-сантиметровый одно-проволочный провод
- Резистор с сопротивлением 1 МОм

ПРИНЦИП РАБОТЫ

Соглашусь, что я немного пофантазировал, назвав этот проект детектором призраков. По сути, устройство обнаруживает *электромагнитные поля*, но многие люди считают, что это как раз и есть призрачные субстанции.

В этом проекте вы настроите антенну, обнаруживающую «призраков», и светодиодную панель, оповещающую о высоком уровне электромагнитной активности вокруг. Оголенный провод действует как антенна, определяя электромагнитное поле в радиусе двух метров. В зависимости от силы сигнала светодиоды будут загораться последовательно: чем сильнее сигнал, тем больше светодиодов загорится. Включите Arduino и направьте ваш детектор в пространство комнаты, чтобы зафиксировать присутствие чего-нибудь сверхъестественного. Имейте в виду, что электрические приборы, например телевизоры, будут оказывать влияние на детектор из-за электромагнитного поля, которое они излучают.

СБОРКА

1. Установите светодиоды на макетную плату, поместив их ножки по обе стороны от канавки (см. раздел «Макетные платы» в проекте 0, чтобы узнать больше о схеме подключения компонентов к макетной плате), как показано на рис. 6.1. Я начал с желтого светодиода, а затем установил шесть зеленых и три красных светодиода, чтобы собрать шкалу (перечисляю цвета слева направо). Вы можете использовать светодиоды любого цвета и устанавливать их по своему усмотрению.

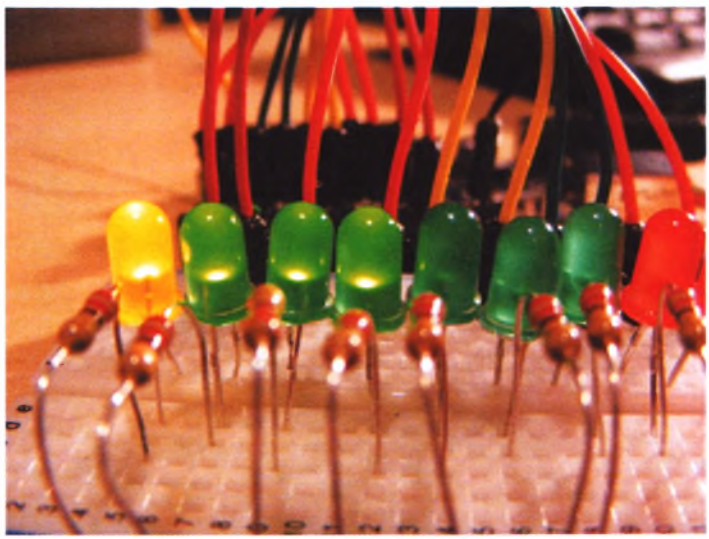
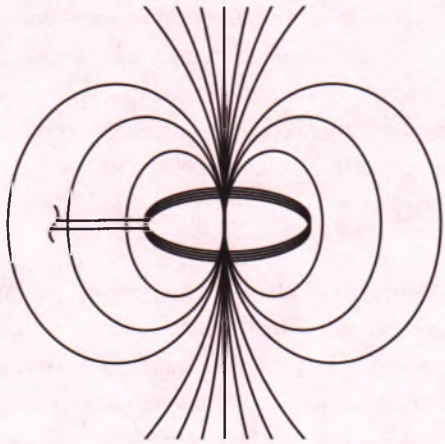


РИСУНОК 6.1

Установка светодиодов

ЭЛЕКТРОМАГНИТНЫЕ ПОЛЯ

Электрические поля создаются различиями в потенциалах: чем выше напряжение, тем сильнее результирующее поле. *Магнитные поля* создаются при течении электрического тока: чем сильнее ток, тем сильнее магнитное поле. *Электромагнитное поле* можно рассматривать как сочетание этих двух полей.



Электромагнитные поля присутствуют повсюду в окружающей среде, но невидимы для человеческого глаза. Электрические поля производятся локальным накоплением электрических зарядов в атмосфере и связаны с грозами. Земля постоянно излучает магнитное поле. Оно используется птицами и рыбой для навигации и заставляет стрелку компаса ориентироваться на север.

2. Подключите к каждому светодиоду резистор с сопротивлением 220 Ом. Одна из ножек резистора подключается к короткой ножке (катоду) светодиода, а вторая ножка резистора устанавливается в гнездо шины заземления макетной платы (см. рис. 6.2). Длинные ножки (аноды) светодиодов подключите к цифровым контактам 2–11 платы Arduino.

СВЕТОДИОДЫ	ARDUINO
Длинные ножки (аноды)	Контакты 2–11
Короткие ножки (катоды)	Контакт GND через резисторы с сопротивлением 220 Ом

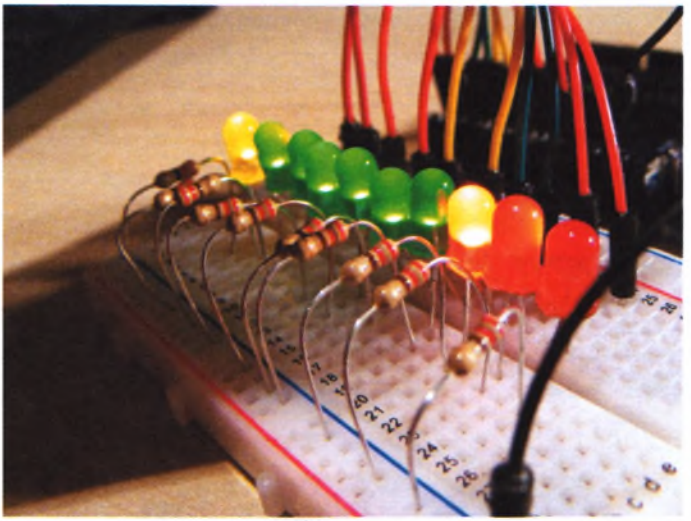


РИСУНОК 6.2

Размещение светодиодов на макетной плате

3. Возьмите одножильный провод длиной примерно 20 см и с помощью стриппера (или ножа) зачистите примерно сантиметр изоляции с одного конца. Этот конец прикрепите к контакту A5 платы Arduino. Зачистите примерно 7 см изоляции с другого конца — этот оголенный конец провода и будет антенной, улавливающей электромагнитное излучение (см. рис. 6.3).

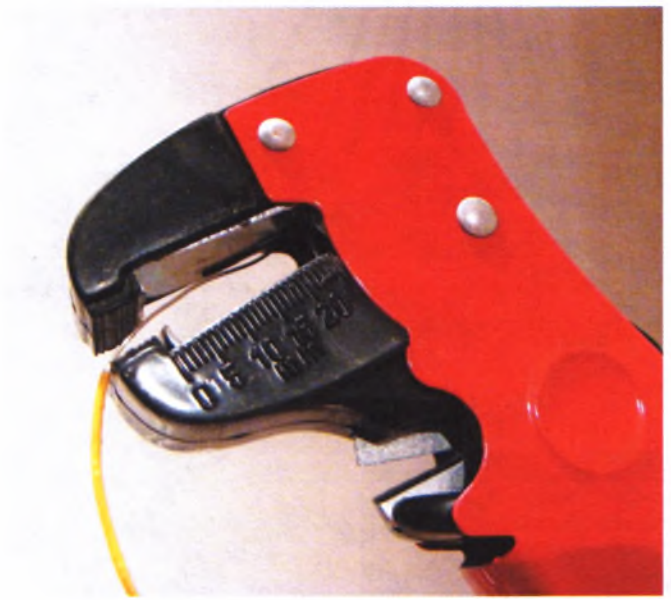


РИСУНОК 6.3

Зачистка провода для создания антенны

4. Подключите одну ножку резистора с сопротивлением 1 МОм к контакту GND платы Arduino, а вторую — к контакту A5 платы Arduino; это повысит чувствительность устройства.
5. Убедитесь, что ваша цепь соответствует схеме на рис. 6.4, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

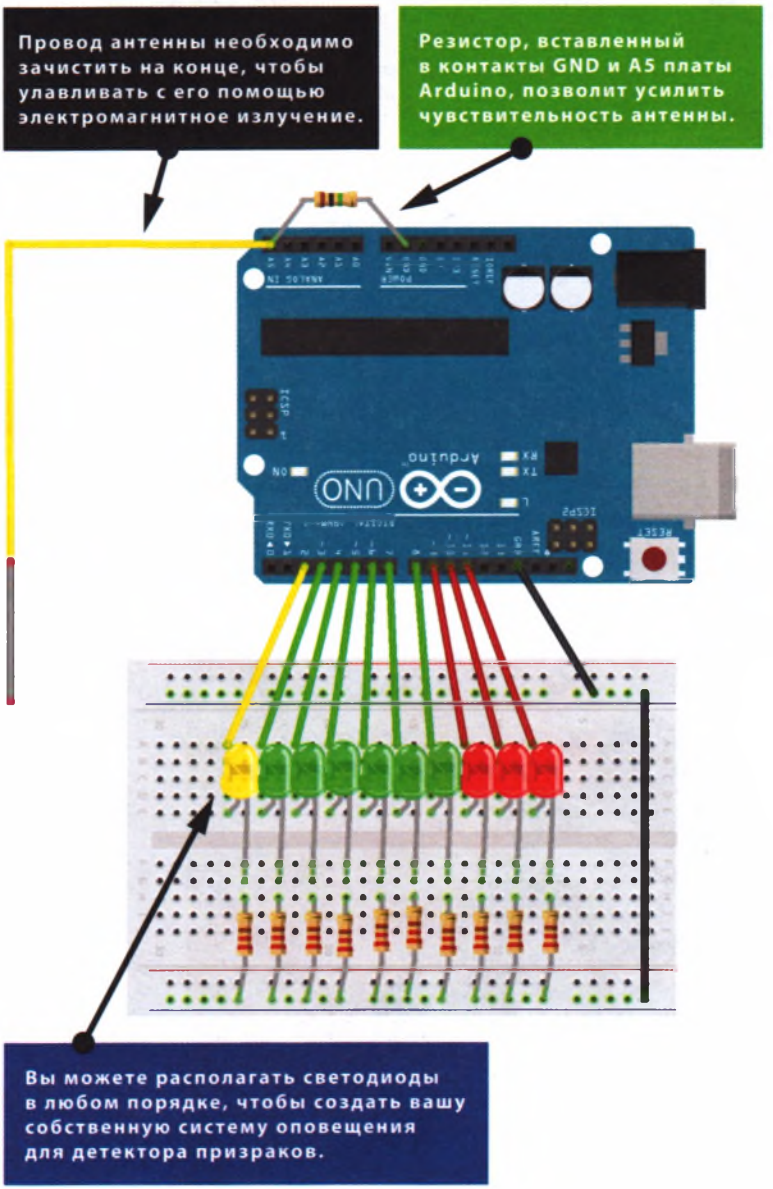


РИСУНОК 6.4

Принципиальная схема детектора призраков

СКЕТЧ

Оголенный провод улавливает электромагнитные импульсы в окружающей среде и отправляет значение в диапазоне от 0 до 1023 на плату Arduino. Программа анализирует показания с аналогового выхода, чтобы вычислить, сколько светодиодов следует последовательно включить или выключить, оповестив пользователя о силе электромагнитного сигнала. В данном случае, 1023 — это самое высокое значение, при котором все светодиоды будут гореть; при значении 550 загорятся пять светодиодов. Скетч работает в цикле, чтобы постоянно анализировать поступающие аналоговые данные, а на светодиоды непрерывно подается напряжение, чтобы моментально отображать результат замеров. Если вы обнаружите, что любые показания электромагнитного излучения приводят к свечению всех светодиодов, поправьте это, уменьшив значение переменной senseLimit. Во время цикла программа выполняет в среднем 25 замеров и использует усредненное значение по результатам показаний для уменьшения сильных колебаний, способных привести к излишне быстрому зажиганию светодиодов.

ПРИМЕЧАНИЕ

После того, как вы собрали детектор призраков, попробуйте добавить компоненты для звукового сопровождения (с изменением скорости воспроизведения или уровня громкости в зависимости от показаний). Соорудите корпус для устройства, чтобы получился персональный карманный детектор, и отправляйтесь на охоту за призраками. Вы также можете поэкспериментировать с проводом разного типа/толщины и/или снять резистор, чтобы настроить уровень чувствительности.

```
// Код Джеймса Ньюболда используется с его согласия
#define NUMREADINGS 25  // Увеличьте значение, чтобы увеличить скорость
                        // захвата данных
int senseLimit = 1023; // Увеличьте значение, чтобы снизить
                        // чувствительность антенны (не более 1023)
int probePin = 5;       // Установка аналогового контакта, к которому
                        // подключена штырь A0 контроллера антенна
int val = 0;            // Считывание из probePin

// Подключение контактов к светодиодной панели с последовательно
// подключенными резисторами
int LED1 = 11;
int LED2 = 10;
int LED3 = 9;
int LED4 = 8;
int LED5 = 7;
int LED6 = 6;
int LED7 = 5;
int LED8 = 4;
int LED9 = 3;
```

```
int LED10 = 2;
int readings[NUMREADINGS];   // Считывание с аналогового входа
int index = 0;               // Индекс текущего считывания
int total = 0;               // Общее количество
int average = 0;             // Окончательное усреднение считанных проб

void setup() {
   pinMode(2, OUTPUT);       // Перевод контактов, к которым
   pinMode(3, OUTPUT);       //подключена светодиодная панель,
   pinMode(4, OUTPUT);       // в режим вывода
   pinMode(5, OUTPUT);
   pinMode(6, OUTPUT);
   pinMode(7, OUTPUT);
   pinMode(8, OUTPUT);
   pinMode(9, OUTPUT);
   pinMode(10, OUTPUT);
   pinMode(11, OUTPUT);
   Serial.pinMode(9600);     // Запуск последовательного подключения
                             // к IDE для отладки и т.п.
   for (int i = 0; i < NUMREADINGS; i++)
      readings[i] = 0;       // Инициализация всех считываний в виде 0
}
void loop() {
   val = analogRead(probePin);      // Чтение показаний с антенны
   if (val >= 1) {                  // Если показание не равно нулю,
                                    // продолжить
      val = constrain(val, 1, senseLimit);        // Если показание
                                                  // превышает текущее
                             // значение senseLimit, присвоить
                             // senseLimit более высокое значение
      val = map(val, 1, senseLimit, 1, 1023);     // Сопоставить лимиты
                             // в диапазоне от 1 до 1023
      total -= readings[index];     // Вычитание последнего значения
                                    // считывания
      readings[index] = val;        // Чтение с датчика
      total += readings[index];     // Добавление показания к итогу
      index = (index + 1);          // Переход к следующему индексу
      if (index >= NUMREADINGS)     // Если это последний индекс
                                    // в массиве,
         index = 0;                 // перейти в начало
      average = total / NUMREADINGS; // Вычисление среднего значения
                                    // считывания
      if (average > 50) {           // Если среднее значение
                                    // считывания выше 50,
         digitalWrite(LED1, HIGH);  // подать напряжение на первый
```

Й

```
                                    // светодиод
}
else {                              // Если нет,
   digitalWrite(LED1, LOW);         // выключить этот светодиод
}
if (average > 150) {                // И т.д.
   digitalWrite(LED2, HIGH);
}
else {
   digitalWrite(LED2, LOW);
}
if (average > 250) {
   digitalWrite(LED3, HIGH);
}
else {
   digitalWrite(LED3, LOW);
}
if (average > 350) {
   digitalWrite(LED4, HIGH);
}
else {
   digitalWrite(LED4, LOW);
}
if (average > 450) {
   digitalWrite(LED5, HIGH);
}
else {
   digitalWrite(LED5, LOW);
}
if (average > 550) {
   digitalWrite(LED6, HIGH);
}
else {
   digitalWrite(LED6, LOW);
}
if (average > 650) {
   digitalWrite(LED7, HIGH);
}
else {
   digitalWrite(LED7, LOW);
}
if (average > 750) {
   digitalWrite(LED8, HIGH);
}
else {
```

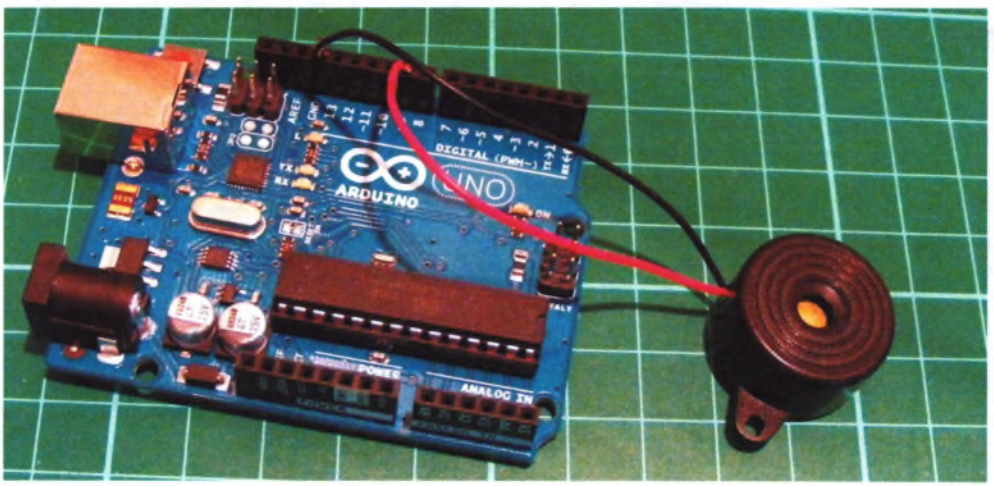
```
        digitalWrite(LED8, LOW);
    }
    if (average > 850) {
        digitalWrite(LED9, HIGH);
    }
    else {
        digitalWrite(LED9, LOW);
    }
    if (average > 950) {
        digitalWrite(LED10, HIGH);
    }
    else {
        digitalWrite(LED10, LOW);
    }
    Serial.println(val); // Применение выходных данных для калибровки
  }
}
```

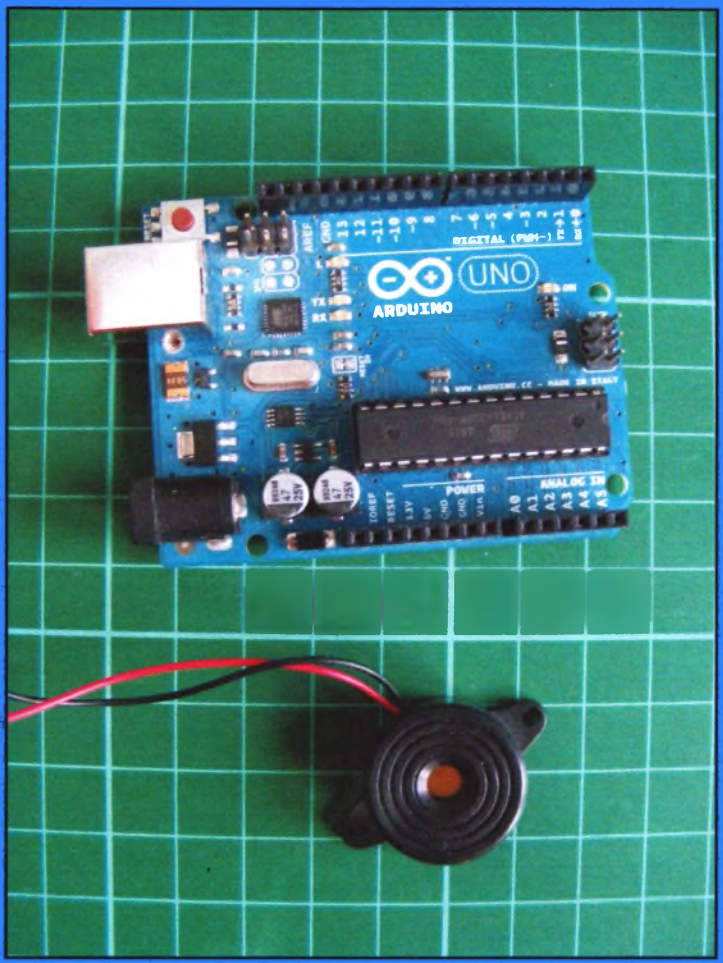
ЧАСТЬ 2

ЗВУК

ПРОЕКТ 7: ПРОИГРЫВАТЕЛЬ ARDUINO

ДО СИХ ПОР ВСЕ НАШИ ПРОЕКТЫ БЫЛИ ВИЗУАЛЬНЫМИ. ДАВАЙТЕ СДЕЛАЕМ МУЗЫКАЛЬНУЮ ПАЗУ. В ЭТОМ ПРОЕКТЕ ВЫ ПРИМЕНИТЕ ПЬЕЗОИЗЛУЧАТЕЛЬ (ЗУММЕР) ДЛЯ ВОСПРОИЗВЕДЕНИЯ ПРОСТЫХ МЕЛОДИЙ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Пьезоизлучатель

ПРИНЦИП РАБОТЫ

Наш проигрыватель Arduino включает пьезоизлучатель для генерации сигналов определенной частоты, напоминающих ноты. Вы также воспользуетесь средой разработки Arduino, чтобы указать порядок исполнения, скорость и длительность звучания нот для воспроизведения определенной мелодии.

Пьезоизлучатели дешевы в производстве и часто используются в детских интерактивных игрушках. Пьезоэлемент без пластикового корпуса выглядит как позолоченный металлический диск с подключенными положительным (обычно красным) и отрицательным (черным) проводами. Он способен издавать только пищащие звуки при подаче на него напряжения. Можно имитировать знакомые ноты, заставив пьезоэлемент вибрировать сотни раз в секунду на определенной частоте, но сначала нам нужно знать частоту тональных звуков, которые нужно воспроизвести. В табл. 7.1 перечислены ноты и соответствующие частоты. *Период* — это продолжительность времени в микросекундах, в течение которого генерируются колебания определенной частоты. Я вдвое уменьшил это число, чтобы получить значения переменной timeHigh, которые используются в коде для имитации ноты.

ТАБЛИЦА 7.1

Ноты и их частоты

НОТА	ЧАСТОТА	ПЕРИОД	TIMEHIGH
C	261 Гц	3,830	1915
D	294 Гц	3,400	1700
E	329 Гц	3,038	1519
F	349 Гц	2,864	1432
G	392 Гц	2,550	1275
A	440 Гц	2,272	1136
B	493 Гц	2,028	1014
C	523 Гц	1,912	956

Код передает на пьезоизлучатель прямоугольную волну соответствующей частоты пьезо, генерируя соответствующий тон (см. проект 2 для получения дополнительной информации о форме сигнала). Значения переменной timeHigh (тоны) рассчитываются по следующей формуле:

```
timeHigh = период / 2 = 1 / (2 * toneFrequency)
```

Цепь этого проекта очень проста и включает только две перемычки, подключенные к плате Arduino.

СБОРКА

1. Подключите черный провод пьезоизлучателя непосредственно к контакту GND платы Arduino, а красный провод — к контакту 9 платы Arduino.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Красный провод	Контакт 9
Черный провод	Контакт GND

2. Убедитесь, что ваша цепь соответствует схеме на рис. 7.1, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

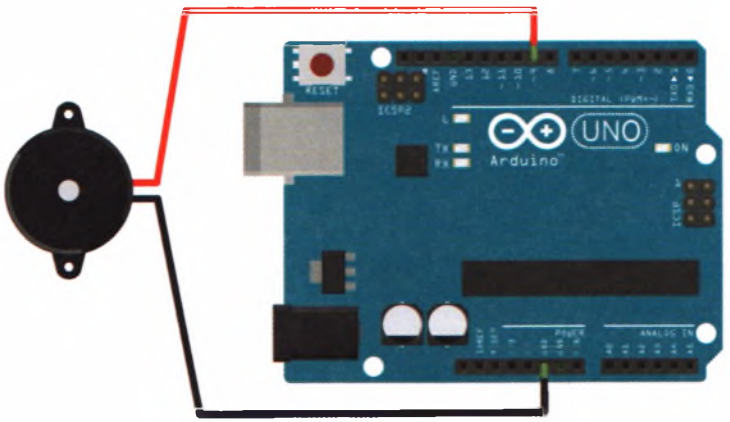


РИСУНОК 7.1

Принципиальная схема проигрывателя Arduino

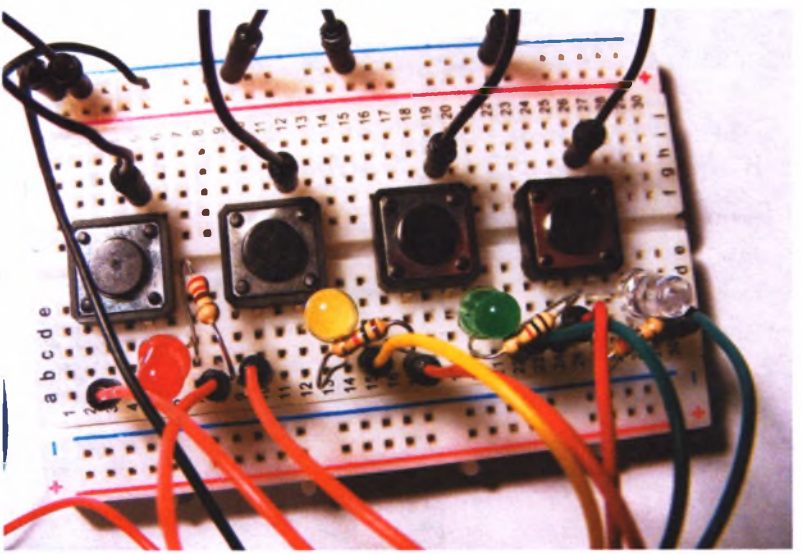
СКЕТЧ

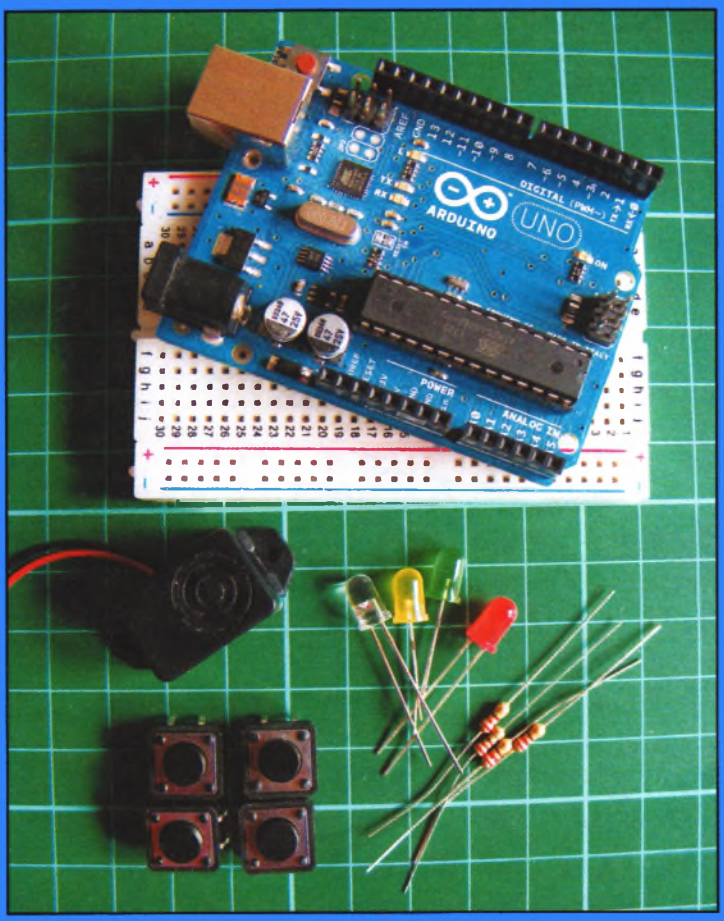
Начнем с простой мелодии. В строке ❶ мы сообщаем среде разработки Arduino, что мелодия будет состоять из 15 нот. Затем мы сохраняем ноты мелодии в символьном массиве в виде текстовой строки в том порядке, в котором они должны проигрываться. После этого сохраняем значение длительности звучания каждой ноты в целочисленном массиве. Если вы хотите изменить мелодию, можете поменять ноты в массиве в строке ❷ и количество тактов, в течение которых звучит каждая нота, в строке ❸. И наконец, в строке ❹ мы определяем темп, в котором будет проигрываться мелодия. Итак, что у нас получилось?

```
// Melody (фрагмент) 2005 Д. Куартиллье для К3
int speakerPin = 9;  // Контакт, к которому подключен пьезоизлучатель
Œ  int length = 15;  // Число нот
   char notes[] = "ccggaagffeeddc ";       // Пробел - пауза
Ž  int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 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);
   }
}

// Присвоение переменной timeHigh значения определенной ноты
   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 };
   for (int i = 0; i < 8; i++) {    // Воспроизведение тона,
                                    // соответствующего ноте
      if (names[i] == note) {
         playTone(tones[i], duration);
      }
   }
}

void setup() {
   pinMode(speakerPin, OUTPUT);            // Перевести speakerPin
                                           // в режим вывода
}

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

ПРОЕКТ 8: ИГРА НА ЗАПОМИНАНИЕ

В ЭТОМ ПРОЕКТЕ МЫ СОЗДАДИМ СОБСТВЕННУЮ ВЕРСИЮ АРКАДНОЙ ЛОГИЧЕСКОЙ ИГРЫ TOUCH ME КОМПАНИИ ATARI, ИСПОЛЬЗУЯ ЧЕТЫРЕ СВЕТОДИОДА, ЧЕТЫРЕ КНОПКИ, ПЬЕЗОИЗЛУЧАТЕЛЬ, НЕСКОЛЬКО РЕЗИСТОРОВ И ПЕРЕМЫЧКИ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Пьезоизлучатель
- 4 четырехконтактные кнопки
- 4 светодиода
- 4 резистора с сопротивлением 220 Ом

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- Tone

ПРИНЦИП РАБОТЫ

В оригинальной игре компании Atari было четыре оборудованных светодиодами цветных панели, которые загорались в определенном порядке, после чего игрок должен был повторить комбинацию (см. рис. 8.1).

РИСУНОК 8.1

Оригинальная игра Touch Me

Наша игра на запоминание воспроизводит короткую начальную мелодию и мигает светодиодами в определенном порядке. После того, как вы повторите комбинацию, нажав кнопки в соответствующем порядке, программа выдаст комбинацию, усложненную на 1 шаг. Так происходит при каждом корректном вводе, чтобы усложнить процесс игры. Если вы ошибетесь при повторении комбинации, игра сбрасывается на начальный уровень.

СБОРКА

1. Установите кнопки на макетную плату так, чтобы они перекрывали собой канавку (словно мостики), а контакты A и B оказались на одной стороне от канавки, а C и D — на другой (см. рис. 8.2). (См. проект 1 для получения подробной информации о том, как работает кнопка.)

РИСУНОК 8.2

Схема четырехконтактного кнопочного переключателя

2. Подключите контакт B каждой кнопки к шине заземления макетной платы, а саму шину — к контакту GND платы Arduino.

3. Подключите контакт D каждой кнопки к цифровым контактам 2–5 платы Arduino (по порядку).
4. Установите светодиоды в макетную плату так, чтобы короткая ножка (катод) каждого из них была соединена с контактом C каждой кнопки. Установите длинную ножку (анод) каждого светодиода в отверстие справа, как показано на рис. 8.3.

КНОПКА	ARDUINO/СВЕТОДИОД
Ножка B	Контакт GND платы Arduino
Ножка C	Короткие ножки (катоды)
Ножка D	Контакты 2–5 платы Arduino

5. Установите на макетную плату резисторы с сопротивлением 220 Ом так, чтобы одна из ножек каждого резистора была соединена с длинной ножкой (анодом) каждого светодиода. Другую ножку каждого резистора с помощью перемычек подключите к Arduino к контактам 8–11, как показано на рис. 8.3.

СВЕТОДИОДЫ	ARDUINO/КНОПКА
Длинные ножки (аноды)	Контакты 8–11 платы Arduino через резисторы с сопротивлением 220 Ом
Короткие ножки (катоды)	Ножка C кнопки

6. Убедитесь, что красный светодиод, подключенный к контакту 11 платы Arduino, соединен с кнопкой, подключенной к контакту 5. Желтый светодиод, подключенный к контакту 10, соединен с кнопкой, подключенной к контакту 4. Зеленый светодиод, подключенный к контакту 9, соединен с кнопкой, подключенной к контакту 3. Синий светодиод, подключенный к контакту 8, соединен с кнопкой, подключенной к контакту 2.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Красный провод	Контакт 12
Черный провод	Контакт GND

7. Подключите черный провод пьезоизлучателя напрямую к контакту GND платы Arduino, а красный — к контакту 12 на плате Arduino.
8. Убедитесь, что ваша цепь соответствует схеме на рис. 8.3, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

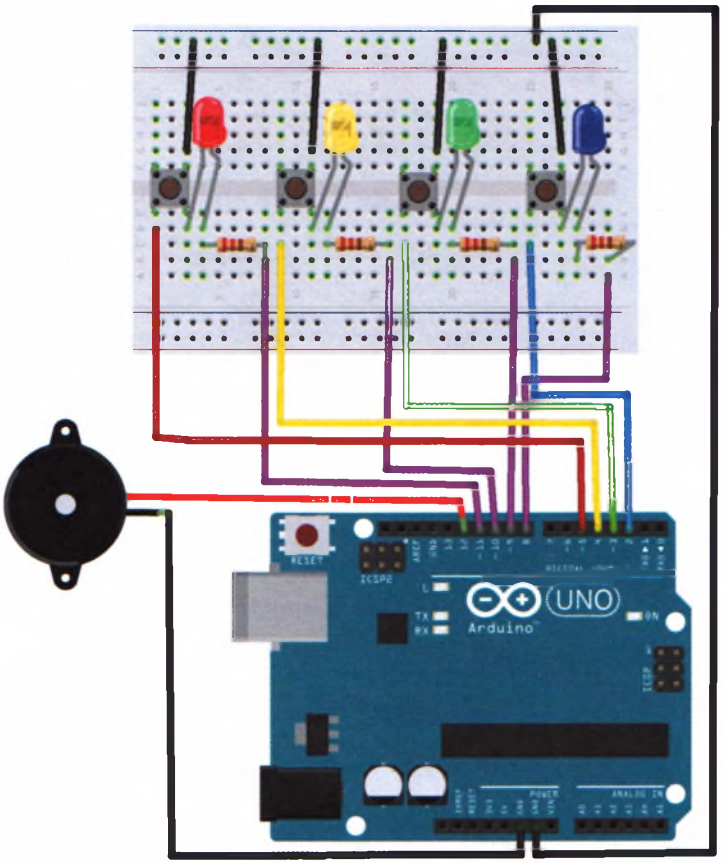


РИСУНОК 8.3

Принципиальная схема игры на запоминание

СКЕТЧ

Скетч генерирует случайную последовательность, в которой будут зажигаться светодиоды. Случайное значение переменной y, генерируемое в цикле последовательности, определяет, на какой светодиод подается питание (например, если переменной y присваивается значение 2, загорается светодиод, подключенный к контакту 2). Во время игры вы должны запомнить последовательность и повторить ее, чтобы перейти на следующий уровень.

На каждом уровне предыдущая последовательность повторяется и добавляется еще одна вспышка случайного светодиода. Вспышка каждого светодиода сопровождается собственным тоном пьезоизлучателя, поэтому мелодия также меняется. Если вы ошиблись при повторе последовательности, программа перезапускается и воспроизводит другую последовательность. Для корректной компиляции скетча нужно подключить библиотеку Tone (доступную по адресу *https://eksmo.ru/files/arduino_geddes.zip*). Подробнее см. раздел «Библиотеки» в проекте 0.

```
// Код Абдуллы Алхазми используется с его согласия, www.Alhazmy13.net
#include <Tone.h>
Tone speakerpin;
int starttune[] = {NOTE_C4, NOTE_F4, NOTE_C4, NOTE_F4, NOTE_C4, NOTE_
                   F4, NOTE_C4, NOTE_F4, NOTE_G4, NOTE_F4, NOTE_
                   E4, NOTE_F4, NOTE_G4};
int duration2[] = {100, 200, 100, 200, 100, 400, 100, 100, 100, 100,
                   200, 100, 500};
int note[] = {NOTE_C4, NOTE_C4, NOTE_G4, NOTE_C5, NOTE_G4, NOTE_C5};
int duration[] = {100, 100, 100, 300, 100, 300};
boolean button[] = {2, 3, 4, 5};    // Контакты, к которым подключены
                                    // кнопки
boolean ledpin[] = {8, 9, 10, 11}; // Контакты, к которым подключены
                                    // светодиоды
int turn = 0;         // Счетчик включений
int buttonstate = 0; // Проверка состояния кнопки
int randomArray[100];// Массив, содержащий до 100 вводов
int inputArray[100];

void setup() {
   Serial.begin(9600);
   speakerpin.begin(12);     // Контакт, к которому подключен
                             // пьезоизлучатель
   for (int x = 0; x < 4; x++) {
      pinMode(ledpin[x], OUTPUT);   // Перевод контактов, к которым
                             // подключены светодиоды, в режим вывода
   }
   for (int x = 0; x < 4; x++) {
      pinMode(button[x], INPUT);    // Перевод контактов, к которым
                                    // подключены кнопки, в режим ввода
      digitalWrite(button[x], HIGH);       // Включение внутреннего
                    // подтягивания; кнопки активны в верхнем
                    // положении (логическая инверсия)
   }
   // Генерация "большей случайности" функции randomArray, чтобы
   // последовательность каждый раз менялась
   randomSeed(analogRead(0));
   for (int thisNote = 0; thisNote < 13; thisNote ++) {
      speakerpin.play(starttune[thisNote]);       // Воспроизведение
                                                  // следующей ноты
      if (thisNote == 0 || thisNote == 2 || thisNote == 4 || thisNote
   == 6) {           // Продолжение ноты
         digitalWrite(ledpin[0], HIGH);
      }
      if (thisNote == 1 || thisNote == 3 || thisNote == 5 || thisNote
   == 7 || thisNote == 9 || thisNote == 11) {
         digitalWrite(ledpin[1], HIGH);
      }
      if (thisNote == 8 || thisNote == 12) {
         digitalWrite(ledpin[2], HIGH);
      }
      if (thisNote == 10) {
         digitalWrite(ledpin[3], HIGH);
      }
      delay(duration2[thisNote]);
```

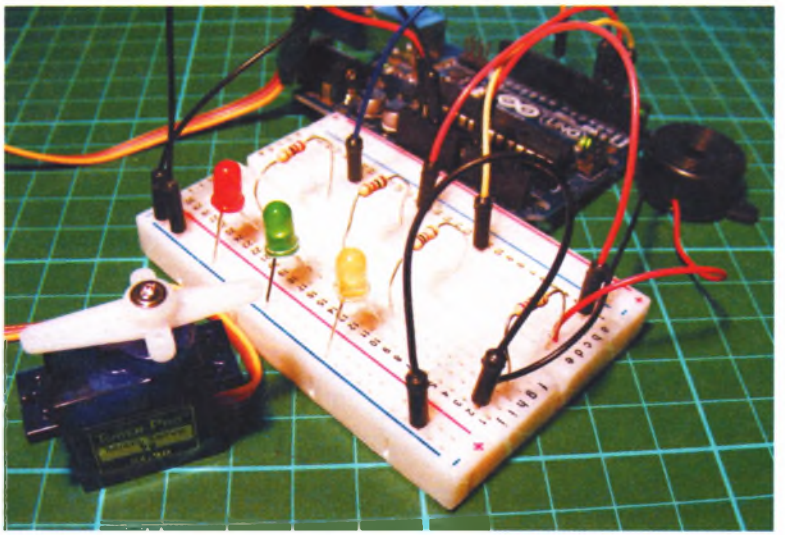
```
      speakerpin.stop();              // Остановка для воспроизведения
                                      // следующей ноты
      digitalWrite(ledpin[0], LOW);
      digitalWrite(ledpin[1], LOW);
      digitalWrite(ledpin[2], LOW);
      digitalWrite(ledpin[3], LOW);
      delay(25);
   }
   delay(1000);
}

void loop() {
   // Генерация массива, назначаемого игроку
   for (int y = 0; y <= 99; y++) {
      digitalWrite(ledpin[0], HIGH);
      digitalWrite(ledpin[1], HIGH);
      digitalWrite(ledpin[2], HIGH);
      digitalWrite(ledpin[3], HIGH);
      // Воспроизведение следующей ноты
      for (int thisNote = 0; thisNote < 6; thisNote ++) {
         speakerpin.play(note[thisNote]);          // Продолжение ноты
         delay(duration[thisNote]); // Остановка для воспроизведения
                                    // следующей ноты
         speakerpin.stop();
         delay(25);
      }
      digitalWrite(ledpin[0], LOW);
      digitalWrite(ledpin[1], LOW);
      digitalWrite(ledpin[2], LOW);
      digitalWrite(ledpin[3], LOW);
      delay(1000);
      // Ограничения переменной turn
      for (int y = turn; y <= turn; y++) {
         Serial.println("");
         Serial.print("Turn: ");
         Serial.print(y);
         Serial.println("");
         randomArray[y] = random(1, 5);     // Присвоение случайного
                                            // номера (1-4)
         // Включение светодиодов в случайном порядке
         for (int x = 0; x <= turn; x++) {
            Serial.print(randomArray[x]);
            for (int y = 0; y < 4; y++) {
               if (randomArray[x] == 1 && ledpin[y] == 8) {
                  digitalWrite(ledpin[y], HIGH);
                  speakerpin.play(NOTE_G3, 100);
                  delay(400);
                  digitalWrite(ledpin[y], LOW);
                  delay(100);
               }
               if (randomArray[x] == 2 && ledpin[y] == 9) {
                  digitalWrite(ledpin[y], HIGH);
                  speakerpin.play(NOTE_A3, 100);
                  delay(400);
                  digitalWrite(ledpin[y], LOW);
                  delay(100);
```

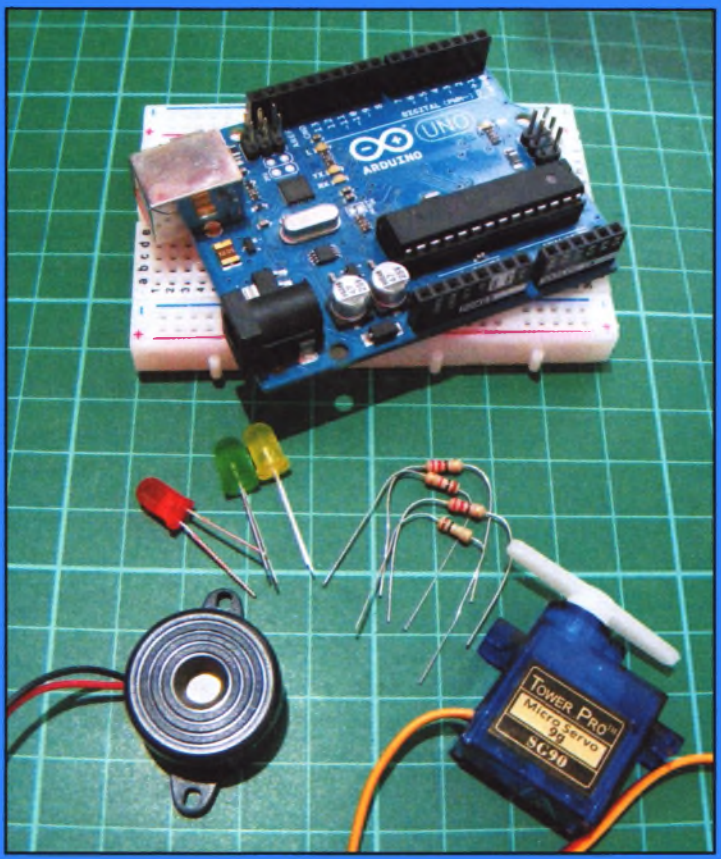
```
            }
            if (randomArray[x] == 3 && ledpin[y] == 10) {
               digitalWrite(ledpin[y], HIGH);
               speakerpin.play(NOTE_B3, 100);
               delay(400);
               digitalWrite(ledpin[y], LOW);
               delay(100);
            }
            if (randomArray[x] == 4 && ledpin[y] == 11) {
               digitalWrite(ledpin[y], HIGH);
               speakerpin.play(NOTE_C4, 100);
               delay(400);
               digitalWrite(ledpin[y], LOW);
               delay(100);
            }
         }
      }
   }
   input();
  }
}

// Проверка ввода на соответствие последовательности
void input() {
   for (int x = 0; x <= turn;) {
      for (int y = 0; y < 4; y++) {
         buttonstate = digitalRead(button[y]);     // Проверка нажатия
                                                   // кнопки
         if (buttonstate == LOW && button[y] == 2) {
            digitalWrite(ledpin[0], HIGH);
            speakerpin.play(NOTE_G3, 100);
            delay(200);
            digitalWrite(ledpin[0], LOW);
            inputArray[x] = 1;
            delay(250);
            Serial.print(" ");
            Serial.print(1);
            // Проверка введенного пользователем значения на соответствие
            // сгенерированному массиву
            if (inputArray[x] != randomArray[x]) {
               fail();// Если нет, вызывается функция fail
            }
            x++;
         }
         if (buttonstate == LOW && button[y] == 3) {
            digitalWrite(ledpin[1], HIGH);
            speakerpin.play(NOTE_A3, 100);
            delay(200);
            digitalWrite(ledpin[1], LOW);
            inputArray[x] = 2;
            delay(250);
            Serial.print(" ");
            Serial.print(2);
            if (inputArray[x] != randomArray[x]) {
               fail();
            }
```

```
            x++;
        }
        if (buttonstate == LOW && button[y] == 4) {
            digitalWrite(ledpin[2], HIGH);
            speakerpin.play(NOTE_B3, 100);
            delay(200);
            digitalWrite(ledpin[2], LOW);
            inputArray[x] = 3;
            delay(250);
            Serial.print(" ");
            Serial.print(3);
            if (inputArray[x] != randomArray[x]) {
               fail();
            }
            x++;
        }
        if (buttonstate == LOW && button[y] == 5) {
            digitalWrite(ledpin[3], HIGH);
            speakerpin.play(NOTE_C4, 100);
            delay(200);
            digitalWrite(ledpin[3], LOW);
            inputArray[x] = 4;
            delay(250);
            Serial.print(" ");
            Serial.print(4);
            if (inputArray[x] != randomArray[x]) {
            fail();
            }
               x++;
        }
      }
   }
   delay(500);
   turn++;     // Увеличение значения переменной turn
}

// Функция, используемая в случае, если игрок ошибся в последовательности
void fail() {
   for (int y = 0; y <= 2; y++) {          // Индикация светодиодами
                                           // для обозначения ошибки
      digitalWrite(ledpin[0], HIGH);
      digitalWrite(ledpin[1], HIGH);
      digitalWrite(ledpin[2], HIGH);
      digitalWrite(ledpin[3], HIGH);
      speakerpin.play(NOTE_G3, 300);
      delay(200);
      digitalWrite(ledpin[0], LOW);
      digitalWrite(ledpin[1], LOW);
      digitalWrite(ledpin[2], LOW);
      digitalWrite(ledpin[3], LOW);
      speakerpin.play(NOTE_C3, 300);
      delay(200);
   }
   delay(500);
   turn = -1;  // Сбросить значение переменной turn для начала новой игры
}
```

ПРОЕКТ 9: ЭЛЕКТРОННЫЙ ПРИВРАТНИК

НА ПРОТЯЖЕНИИ ВЕКОВ СЕКРЕТНЫЕ АГЕНТЫ ПОЛЬЗОВАЛИСЬ ТАЙНЫМИ КОМБИНАЦИЯМИ СТУКА ДЛЯ ПРЕДОТВРАЩЕНИЯ ДОСТУПА В ГРУППУ ПОСТОРОННИХ ЛИЦ. МЫ ПОПРОБУЕМ РЕАЛИЗОВАТЬ ЭТУ СИСТЕМУ НА ПРАКТИКЕ, СОЗДАВ ПЕРСОНАЛЬНОГО ЭЛЕКТРОННОГО ПРИВРАТНИКА.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Сервопривод Tower Pro 9g SG90
- Пьезоизлучатель
- 3 светодиода разного цвета
- Резистор с сопротивлением 1 МОм
- 3 резистора с сопротивлением 220 Ом

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- Servo

ПРИНЦИП РАБОТЫ

В этом проекте вы соберете цепь, которая при правильной комбинации стука перемещает рычаг сервопривода, чтобы открыть сундук или дверь. До сих пор мы использовали пьезоизлучатель только для воспроизведения звука, но мы также можем использовать его в качестве датчика для обнаружения звуков, в данном случае стука. Когда пьезоизлучатель фиксирует стук, он вместо воспроизведения звука выдает напряжение, значение которого зависит от силы удара. Программа будет фиксировать значения этого напряжения и если они попадают в определенный диапазон, Arduino зарегистрирует их как корректные. Если будут определены три удара с правильным напряжением, код будет взломан, а рычаг сервопривода переместится, чтобы разблокировать коробку или дверь.

Ниже показаны две строки кода, которые мы будем использовать позже в скетче. Они определяют диапазон значений напряжения. Если три последовательных значения напряжения будут находиться в диапазоне от 10 до 100, то стук будет принят как корректный.

```
const int quietKnock = 10;
const int loudKnock = 100;
```

Если вы постучите слишком слабо или слишком громко, стук не будет принят. Вам нужно сделать три «правильных» удара, чтобы заставить рычаг сервопривода переместился. После определения правильной последовательности и силы стука, рычаг сервопривода повернется на 90 градусов, чтобы «разблокировать» коробку или дверь. Светодиоды, показанные на рис. 9.1, служат индикаторами состояния блокировки: красный светодиод загорается, если стук неверный и рычаг сервопривода не перемещается (то есть коробка или дверь все еще заблокированы); желтый светодиод мигает, когда регистрируется стук и отображается правильный код; а зеленый светодиод загорается и сервопривод перемещается после трех правильных ударов.

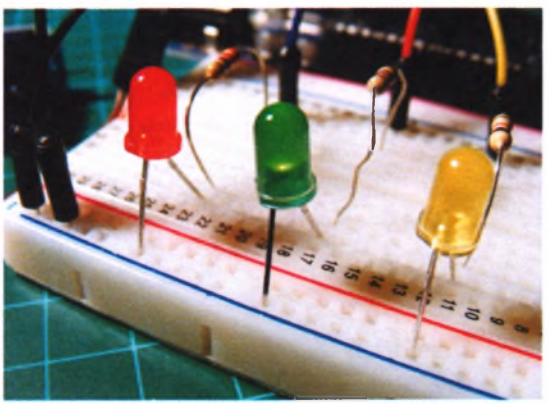


РИСУНОК 9.1

Установка светодиодов

Для достижения наилучшего результата выньте пьезоизлучатель из его корпуса и прикрепите пьезоизлучатель непосредственно к внутренней части коробки или снаружи двери, чтобы датчик был более чувствителен к вибрации от стука.

СБОРКА

1. Установите на макетную плату резистор с сопротивлением 1 МОм и подключите красный провод пьезоизлучателя к одной ножке, а черный — к другой. Подключите черный провод к шине заземления, а красный провод — к контакту A0 платы Arduino.

ПЬЕЗО	ARDUINO
Красный провод	Контакт A0 через резистор с сопротивлением 1 МОм
Черный провод	Контакт GND через резистор с сопротивлением 1 МОм

2. Подключите желтый провод сигнала управления сервопривода напрямую к контакту 9 платы Arduino, коричневый провод — к контакту GND, а красный провод — к контакту 5V.

СЕРВОПРИВОД	ARDUINO
Желтый провод	Контакт 9
Красный провод	Контакт 5V
Коричневый провод	Контакт GND

3. Установите светодиоды на макетную плату. Короткие ножки (катоды) должны быть подключены к контакту GND. Длинные ножки (аноды) следует подключить к контактам Arduino через резисторы с сопротивлением 220 Ом следующим образом: желтый светодиод подключается к контакту 3 платы Arduino, зеленый — к контакту 4, а красный — к контакту 5.

СВЕТОДИОДЫ	ARDUINO
Длинные ножки (аноды)	Контакты 3–5 через резисторы с сопротивлением 220 Ом
Короткие ножки (катоды)	Контакт GND

4. Подключите шину питания макетной платы к контакту 2 платы Arduino. В нашем проекте питание подается непрерывно, но вы можете добавить переключатель в цепь между контактом 2 платы Arduino и шиной питания, чтобы экономить электричество, когда проект не используется.

5. Проверьте подключение шин макетной платы к контактам GND и 5V платы Arduino.
6. Убедитесь, что ваша цепь соответствует схеме на рис. 9.2, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

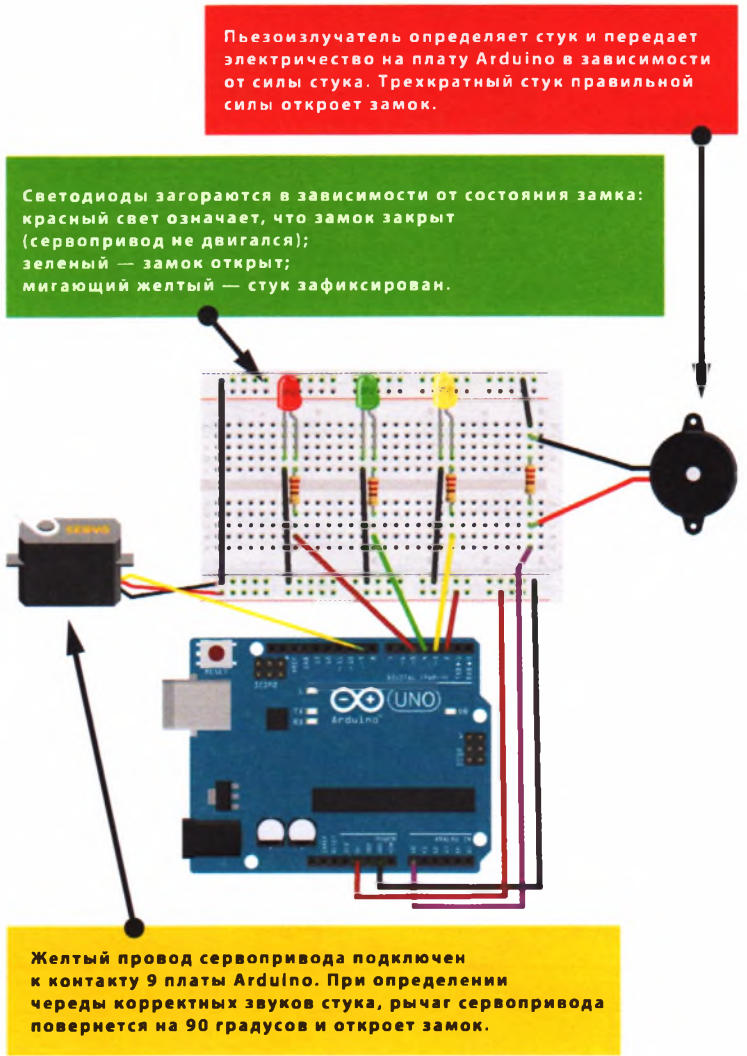


РИСУНОК 9.2

Принципиальная схема замка секретного стука

СКЕТЧ

Вначале вызывается библиотека Servo и настраивается контакт 9 платы Arduino, предназначенный для управления сервоприводом. Светодиоды подключены к контактам 3, 4 и 5 платы Arduino и будут загораться в зависимости от правильности

стука. Пьезоизлучатель в этом проекте действует как датчик, а не средство воспроизведения звука, и подключен к контакту A0 платы Arduino. Когда раздается стук, он фиксируется пьезоизлучателем, и на аналоговый контакт A0 платы Arduino передается электрический ток, напряжение которого зависит от силы стука — чем он сильнее, тем выше напряжение. Стук, генерирующий напряжение со значением менее 10, рассматривается как слишком слабый, а со значением выше 100 — как слишком громкий, так что в любом из этих случаев стук не будет определен как корректный. Красный светодиод загорается, если стук неправильный, а желтый — если правильный. Допустимо любое значение между 10 и 100, которое рассматривается как «правильный» стук и считается. Если получено три «правильных» стука, сервопривод приходит в движение и загорается зеленый светодиод.

Как было упомянуто, уровень напряжения задается в следующих двух строках кода:

```
const int quietKnock = 10;
const int loudKnock = 100;
```

Для повышения уровня безопасности можно уменьшить данный диапазон, чтобы усложнить код для взлома. Ниже представлен полный код скетча:

```
/* Создано 18 сентября 2012 Скоттом Фитцжеральдом
   Благодарность Федерико Ванцати за улучшения
   http://arduino.cc/starterKit
   Этот код - общественное достояние.
*/

#include <Servo.h>
Servo servo9; // Контакт, к которому подключен контакт управления
              // сервопривода

const int piezo = A0;         // Контакт, к которому подключен
                              // пьезоизлучатель
const int switchPin = 2;      // Контакт, к которому подключен
                              // сервопривод
const int yellowLed = 3;      // Контакт, к которому подключен желтый
                              // светодиод
const int greenLed = 4;       // Контакт, к которому подключен зеленый
                              // светодиод
const int redLed = 5;         // Контакт, к которому подключен красный
                              // светодиод

int knockVal;                 // Значение силы стука
int switchVal;

const int quietKnock = 10;    // Установка минимального допустимого
                              // значения
const int loudKnock = 100;    // Установка максимального допустимого
                              // значения
boolean locked = false;       // Логическая переменная
int numberOfKnocks = 0;       // Значение количества стуков

void setup() {
   servo9.attach(9);
   pinMode(yellowLed, OUTPUT);      // Перевод контактов, к которым
                                    // подключены светодиоды, в режим
                                    // вывода
```

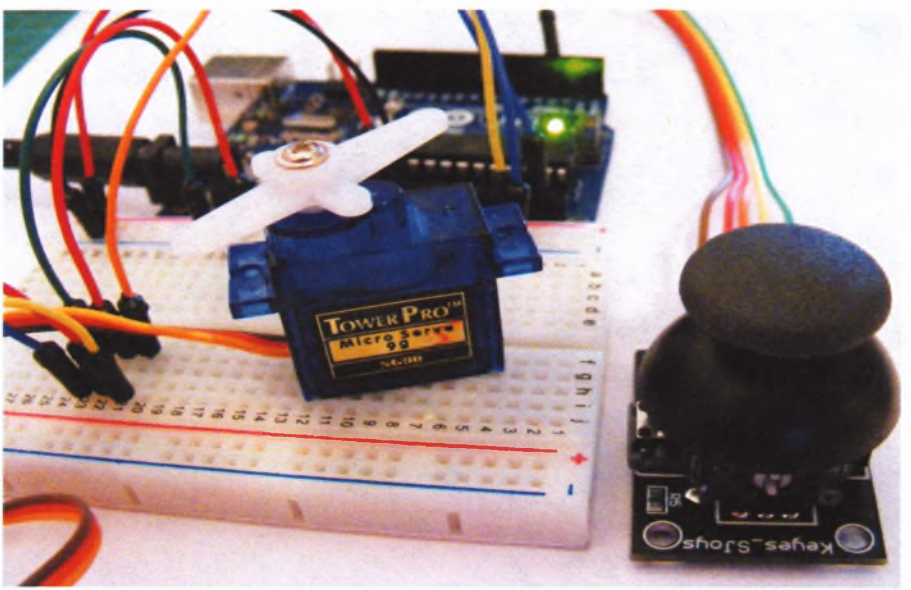
```
   pinMode(greenLed, OUTPUT);
   pinMode(redLed, OUTPUT);
   pinMode(switchPin, INPUT);        // Перевод контакта, к которому
                                     // подключен сервопривод, в режим
                                     // ввода
   Serial.begin(9600);
   digitalWrite(greenLed, HIGH);     // Подается питание на зеленый
                                     // светодиод, если комбинация
                                     // стуков корректна
   servo9.write(0);
   Serial.println("The box is unlocked!");
}

void loop() {
   if (locked == false) {
      switchVal = digitalRead(switchPin);
      if (switchVal == HIGH) {
         locked = true;
         digitalWrite(greenLed, LOW);
         digitalWrite(redLed, HIGH);
         servo9.write(90);
         Serial.println("The box is locked!");
         delay(1000);
      }
   }
   if (locked == true) {
      knockVal = analogRead(piezo); // Значение стука, считываемое
                                     // аналоговым контактом
      if (numberOfKnocks < 3 && knockVal > 0) {
         if (checkForKnock(knockVal) == true) {    // Проверка
                                     // корректности количества ударов
            numberOfKnocks++;
         }
         Serial.print(3 - numberOfKnocks);
         Serial.println(" more knocks to go");
      }
      if (numberOfKnocks >= 3) {     // Подача питания на сервопривод,
                                     // если зафиксированы 3 корректных
                                     // стука

         locked = false;
         servo9.write(0);
         delay(20);
         digitalWrite(greenLed, HIGH);
         digitalWrite(redLed, LOW);
         Serial.println("The box is unlocked!");
      }
   }
}

boolean checkForKnock(int value) {  // Проверка силы стука
   if (value > quietKnock && value < loudKnock) { // Значение силы
                     // стука должно находиться в указанном диапазоне
      digitalWrite(yellowLed, HIGH);
      delay(50);
      digitalWrite(yellowLed, LOW);
      Serial.print("Valid knock of value ");
      Serial.println(value);
      return true;
   }
   else {                        // Если значение false, передать текущие
                                 // данные на монитор порта IDE
      Serial.print("Bad knock value ");
      Serial.println(value);
      return false;
   }
}
```

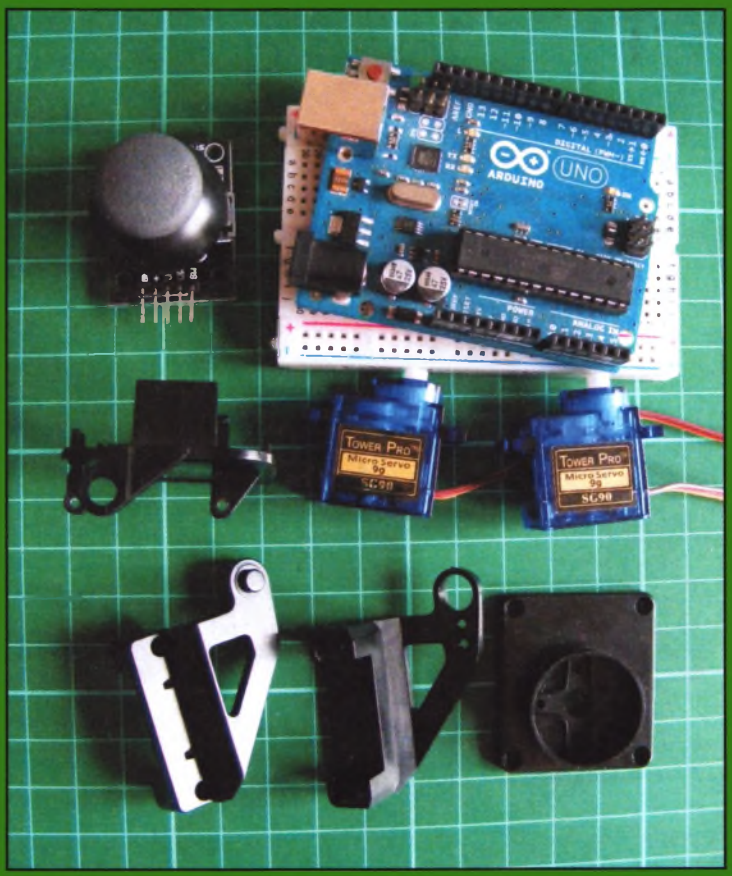
ЧАСТЬ 3

ДВИЖЕНИЕ

ПРОЕКТ 10: ЛАЗЕР, УПРАВЛЯЕМЫЙ ДЖОЙСТИКОМ

В ЭТОМ ПРОЕКТЕ МЫ С ПОМОЩЬЮ ДЖОЙСТИКА БУДЕМ УПРАВЛЯТЬ ЛАЗЕРНОЙ УКАЗКОЙ. В ПРОЕКТЕ ИСПОЛЬЗУЮТСЯ ДВА СЕРВОПРИВОДА, АНАЛОГОВЫЙ ПЯТИКОНТАКТНЫЙ ДВУХОСНЫЙ ДЖОЙСТИК И КОРПУС С ПОВОРОТНЫМ УСТРОЙСТВОМ ДЛЯ НАВЕДЕНИЯ ЛАЗЕРА.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Arduino
- Макетная плата
- Перемычки
- 2 сервопривода Tower Pro 9g SG90
- Аналоговый пятиконтактный двухосный джойстик
- Корпус с поворотным устройством

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- Servo

ПРИНЦИП РАБОТЫ

Сервоприводы — это небольшие моторы, которые могут точно поворачивать свои рычаги в любое положение в диапазоне от 0 до 180 градусов. В этом проекте мы поместим сервоприводы в корпус с поворотным устройством (так называемый pan-and-tilt), который имеет две оси вращения. Такой корпус оправдывает затраты на него, так как значительно упрощает присоединение лазера к сервоприводу. В этом проекте используется лазерная указка, но вы можете легко сменить ее на веб-камеру или другое небольшое устройство. В цепи используются два сервопривода: один для движения влево и вправо, а другой для движения вверх и вниз. Как вы, возможно, помните, от сервопривода идут три провода; все они показаны на рис. 10.1: положительный питания (красный), отрицательный питания или заземления (черный или коричневый) и для передачи сигнала управления (обычно желтый, оранжевый или белый).

РИСУНОК 10.1

Сервопривод имеет три контакта

Прежде чем мы приступим к сборке, вам нужно разобраться в том, как работает джойстик. Джойстик, показанный на рис. 10.2, по сути, представляет собой два потенциометра и кнопку, которые позволяют определить позицию стержня в двух измерениях.

РИСУНОК 10.2

Джойстик содержит два потенциометра и кнопку для измерения движения

Потенциометры представляют собой переменные резисторы — своего рода датчики, выдающие напряжение в зависимости от поворота устройства вокруг своей оси. По мере перемещения джойстика вокруг своего центра его сопротивление — и, следовательно, вывод — изменяется. Выходы потенциометров являются аналоговыми, поэтому могут передавать значение только в диапазоне от 0 до 1023 при считывании аналоговым контактом платы Arduino. В соответствии с полученным значением на Arduino передается импульс, который, в свою очередь, определяет движение сервоприводов. (Более подробная информация о потенциометрах приведена выше в проекте 2.)

Обычно джойстик имеет пять штырьковых контактов: VRx (сигнал по оси x), VRy (сигнал по оси y), SW (кнопка, которую мы не будем использовать в этом проекте), GND (заземление) и 5V (питание).

Когда джойстик перемещается влево или вправо (по оси x), соответствующий сервопривод перемещается в этом направлении; когда джойстик перемещается вверх или вниз (по оси y), другой сервопривод перемещается вверх или вниз.

СБОРКА

1. Подключите оба красных провода сервоприводов к шине 5V, а коричневые провода — к шине заземления макетной платы.
2. Подключите желтый провод сигнала управления одного из сервоприводов напрямую к контакту 9 платы Arduino, а второго — к контакту 10 платы Arduino, как показано на схеме на рис. 10.4.

СЕРВОПРИВОДЫ	ARDUINO
Красные провода	Контакт 5V
Коричневые провода	Контакт GND
Желтый провод первого сервопривода	Контакт 9
Желтый провод второго сервопривода	Контакт 10

3. Подключите штырь GND модуля джойстика к шине заземления макетной платы, штырь 5V — к шине питания макетной платы. Подключите штырь VRx напрямую к контакту A0 платы Arduino, а VRy — к контакту A1. Напомню, что штырь SW модуля джойстика в этом проекте не используется.

ДЖОЙСТИК	ARDUINO
Штырь 5V	Контакт 5V (через макетную плату)
Штырь GND	Контакт GND (через макетную плату)
Штырь VRx	Контакт A0
Штырь VRy	Контакт A1
Штырь SW	Не используется

4. Подключите шины макетной платы к соответствующим контактам платы Arduino — GND и 5V. Затем убедитесь, что ваша цепь соответствует схеме, показанной на рис. 10.3. Загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее.

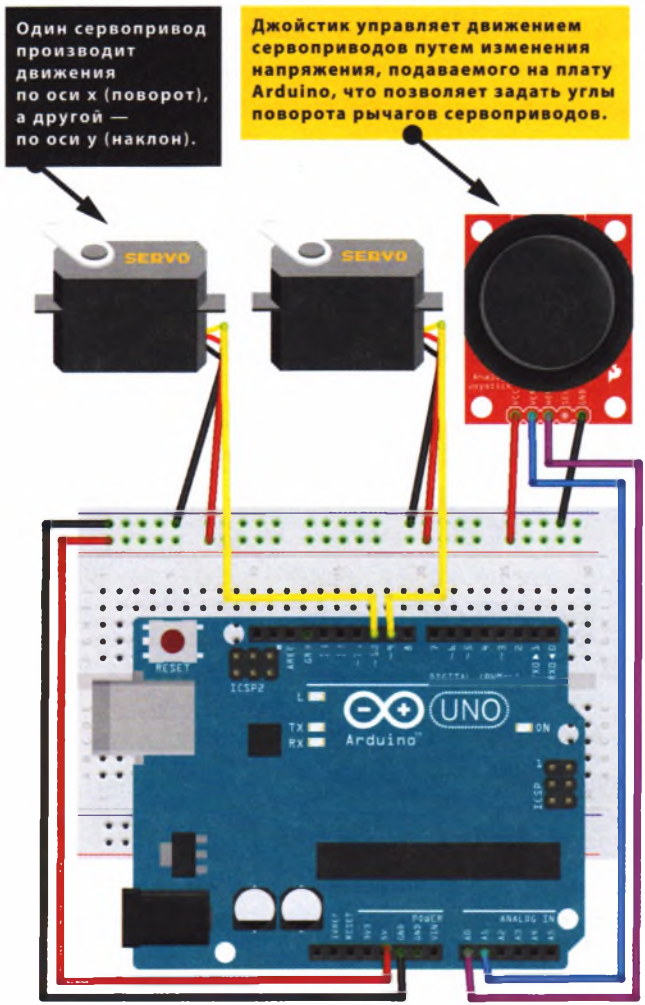


РИСУНОК 10.3

Принципиальная схема лазерной установки, управляемой с помощью джойстика. Джойстик на этой схеме внешне немного отличается от нашего, но их контакты идентичны, поэтому инструкции в проекте будут работать без проблем.

УСТАНОВКА ЛАЗЕРА

В этом проекте я поместил сервоприводы в корпус с поворотным устройством, имеющим две оси. Вы можете найти такой или аналогичный корпус за вменяемую цену на сайте типа eBay, выполнив поиск «сервопривод Arduino pan and

tilt kit». Возможно, вам придется собрать такой корпус самостоятельно, но если следовать инструкции, то это довольно просто.

Прикрепите лазерную указку к верхней части модуля; я рекомендую воспользоваться клеевым пистолетом для постоянного крепления, но вы можете использовать скотч, если планируете разобрать устройство в будущем. Теперь вы можете управлять лазером с помощью джойстика. Сервоприводы зафиксированы в корпусе с поворотным устройством, как показано на рис. 10.5.

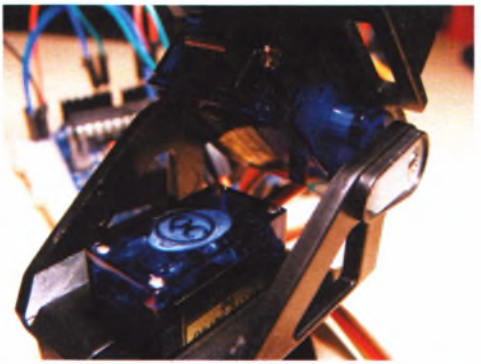


РИСУНОК 10.4

Установка сервоприводов в корпус с поворотным устройством

Перемещение джойстика влево/вправо приведет к подаче питания на сервопривод оси x, а перемещение вверх/вниз — к подаче питания на сервопривод оси y. Общий вид конструкции показан на рис. 10.5.

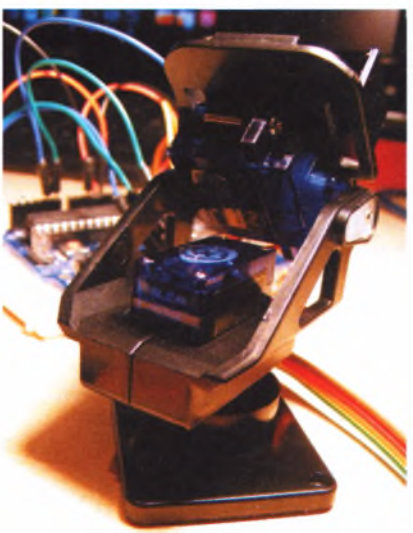


РИСУНОК 10.5

Полная конструкция

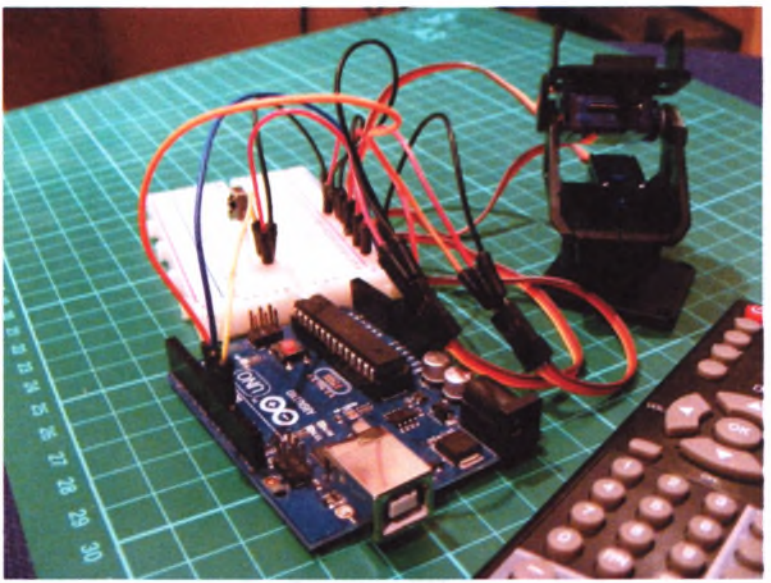
СКЕТЧ

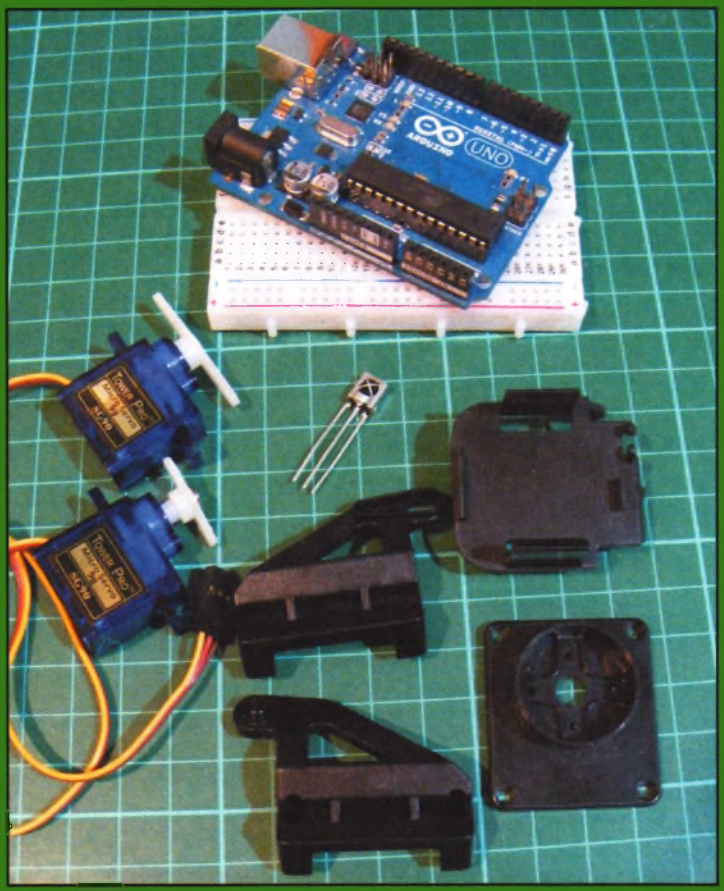
Сначала в скетче вызывается библиотека Servo, а затем определяются два сервопривода для наклона и поворота с именами tilt и pan соответственно. Штырь оси *x* джойстика соединен с контактом A0 платы Arduino, а оси *y* — с контактом A1; они обеспечивают входные данные. Затем эти данные присваиваются в виде значений переменным движения по осям *x* и *y*. Сервопривод tilt подключен к контакту 9 платы Arduino, а сервопривод pan — к контакту 10. Оба контакта работают в режиме вывода. В процессе работы плата Arduino считывает поступающие от джойстика данные и передает полученные напряжения на выводы, заставляя сервоприводы двигаться в соответствии с заданным направлением.

```
// Код используется с согласия разработчиков http://learn.explorelabs.com/
// С указанием авторства-С сохранением условий 4.0 Всемирная лицензия
// (CC BY-SA 4.0)

#include <Servo.h>
Servo tilt, pan;        // Создание объекта servo
int joyX = A0;          // Аналоговый контакт, к которому подключен штырь
                        // VRx джойстика
int joyY = A1;          // Аналоговый контакт, к которому подключен
                        // штырь VRy джойстика
int x, y;               // Переменные для считываемых значений

void setup() {
   tilt.attach(9);      // Подключение сервопривода tilt (наклона),
                        // подключенного к контакту 9, к объекту servo
   pan.attach(10);      // Подключение сервопривода pan (поворота),
                        // подключенного к контакту 10, к объекту servo
}

void loop() {
   x = joyX;   // Чтение значений перемещения по оси x (от 0 до 1023)
   y = joyY;   // Чтение значений перемещения по оси y (от 0 до 1023)
   x = map(analogRead(joyX), 0, 1023, 900, 2100);  // Шкалирование
                        // значений для применения к сервоприводам в виде
                        // значения в диапазоне от 900 до 2100 мкс
   y = map(analogRead(joyY), 0, 1023, 900, 2100);
   tilt.write(x);       // Установка сервопривода в позицию,
                        // соответствующую шкалированному значению
   pan.write(y);
   delay(15);           // Ожидание, пока сервоприводы занимают новое
                        // положение
}
```

ПРОЕКТ 11: ДИСТАНЦИОННОЕ УПРАВЛЕНИЕ СЕРВОПРИВОДАМИ

В ЭТОМ ПРОЕКТЕ МЫ БУДЕМ ИСПОЛЬЗОВАТЬ ARDUINO ДЛЯ ПРИЕМА И ДЕКОДИРОВАНИЯ СИГНАЛОВ ПУЛЬТА ДИСТАНЦИОННОГО УПРАВЛЕНИЯ, А ЗАТЕМ ПРИМЕНЕНИЯ ПЕРЕДАННЫХ КОДОВ ДЛЯ УПРАВЛЕНИЯ СЕРВОПРИВОДАМИ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Инфракрасный (ИК) датчик с частотой 38 кГц
- Пульт дистанционного управления
- 2 сервопривода Tower Pro 9g SG90
- Корпус с поворотным устройством

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- Servo
- IRremote

ПРИНЦИП РАБОТЫ

Сначала мы научимся получать и декодировать сигналы пульта дистанционного управления с помощью инфракрасного датчика. ИК-датчик оборудован тремя контактами: OUT, GND и VCC (слева направо на рис. 11.1). Изучите техническую документацию, прилагаемую к купленному вами ИК-датчику и убедитесь, что его контакты соответствуют приведенной схеме. У некоторых моделей расположение контактов может отличаться, но вы можете правильно соединить контакты для подключения такого датчика.

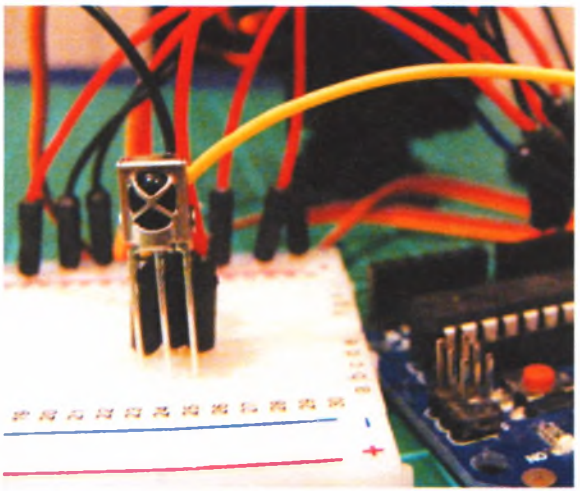


РИСУНОК 11.1

ИК-датчик с тремя контактами слева направо: OUT (выход), GND (заземление) и VCC (питание)

Вам также понадобится пульт дистанционного управления. Можно использовать любой, например телевизионный, но лучше взять какой-нибудь старый, который вы больше не используете. Когда вы нажимаете кнопку на пульте, он передает некое числовое значение, которое принимает ИК-датчик. Значения различны для каждой кнопки. С помощью скетча мы научимся распознавать значение каждой кнопки, а затем присваивать их соответствующим контактам платы Arduino, чтобы управлять выходным устройством — в данном случае двумя сервоприводами.

Указав в скетче значения, полученные при декодировании сигналов, вы можете назначить выполнение определенных инструкций при нажатии той или иной кнопки и использовать пульт дистанционного управления для управления сервоприводами. Если вы уже собрали корпус с поворотным устройством, описанный в проекте 10, то можете воспользоваться им и в этом проекте. В противном случае вернитесь к проекту 10 и прочитайте инструкции по сборке корпуса.

Мы назначим четыре кнопки для управления вращением сервоприводов в корпусе с поворотным устройством, чтобы движение осуществлялось во всех направлениях: влево/вправо для сервопривода оси x и вверх/вниз для сервопривода оси y. Кратковременное нажатие кнопки будет приводить к небольшому повороту

сервопривода, а удерживание кнопки — к непрерывному вращению сервопривода, пока не будет достигнуто максимальное или минимальное значение поворота.

НАСТРОЙКА

1. Загрузите библиотеку IRremote (архив, содержащий библиотеку, доступен по адресу *eksmo.ru/files/arduino_geddes.zip*) и установите ее в среде разработки Arduino, как это продемонстрировано в разделе «Библиотеки» проекта 0.
2. Установите ИК-датчик на макетную плату. Подключите штырь OUT датчика к контакту 11 платы Arduino, штырь GND — к контакту GND Arduino, а штырь VCC — к контакту 5V платы Arduino. Повторюсь, у некоторых ИК-датчиков расположение контактов может отличаться от описываемого здесь. Изучите техническую документацию, прилагаемую к купленному вами ИК-датчику.

ИК-ПЕРЕДАТЧИК	ARDUINO
Штырь OUT	Контакт 11
Штырь GND	Контакт GND
Штырь VCC	Контакт 5V

3. Теперь загрузите и выполните следующий код.

```
/* Авторские права принадлежат Кену Ширриффу, 2009.
   Код использован с его согласия.
   http://arcfn.com
*/

#include <IRremote.h>          // Импорт библиотеки
int receiver = 11;             // Контакт, к которому подключен ИК-датчик

IRrecv irrecv(receiver);
decode_results results;
void setup() {
   Serial.begin(9600);         // Вывод кодов нажатых кнопок в IDE
   irrecv.enableIRIn();        // Включение датчика
}

void loop() {
   if (irrecv.decode(&results)) {   // Если получены входные данные,
                                    // декодировать значение
      Serial.println(results.value, HEX);  // Передача значения кнопки
                                           // в монитор порта
                                           // в шестнадцатеричном виде
      irrecv.resume();              // Получение следующего значения
   }
}
```

Вначале код скетча обращается к библиотеке IRremote, с помощью которой перехватывается сигнал с ИК-датчика и полученные данные передаются на плату Arduino. ИК-датчику назначается контакт 11 платы Arduino, и код скетча формирует канал связи со средой разработки Arduino. Теперь при нажатии той или иной кнопки на пульте ее значение сразу отображается в окне монитора порта. Скетч продолжает работать в зацикленном режиме, ожидая нажатия кнопок и передавая соответствующие значения в среду разработки Arduino.

4. Откройте окно монитора порта в среде разработки Arduino.
5. Направьте пульт дистанционного управления на ИК-датчик и попробуйте нажать несколько кнопок. Их значения отобразятся в окне монитора порта в шестнадцатеричном виде, как показано на рис. 11.2. Для достижения наилучшего результата, нажимайте кнопки быстро, сразу отпуская их. Если удерживать кнопку нажатой, монитор порта выведет значение *Fs* и будет отображать его до тех пор, пока вы не отпустите кнопку.

РИСУНОК 11.2

При нажатии кнопки на пульте дистанционного управления в окне монитора порта среды разработки Arduino отображается шестнадцатеричный код нажатой клавиши

Запишите появляющиеся цифры и названия кнопок, которым они соответствуют. Позже вам понадобятся эти данные.

Теперь мы расшифровали сигналы кнопок пульта и можем использовать их для управления двумя сервоприводами.

СБОРКА

1. Используя схему подключения из таблицы в шаге 2 (выше в этой главе) и макетную плату с уже установленным ИК-датчиком, подключите сервоприводы к плате Arduino. Подключите коричневый провод каждого сервопривода к контакту GND, а красный — к контакту 5V платы Arduino. Затем подключите желтый провод сигнала управления первого сервопривода к контакту 10 платы Arduino, а желтый провод второго сервопривода — к контакту 9.

СЕРВОПРИВОДЫ	ARDUINO
Красные провода	Контакт 5V
Коричневые провода	Контакт GND
Желтый провод (сервопривод 1)	Контакт 10
Желтый провод (сервопривод 2)	Контакт 9

2. Не забудьте подать питание на макетную плату.
3. Убедитесь, что ваша цепь соответствует схеме на рис. 11.3, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

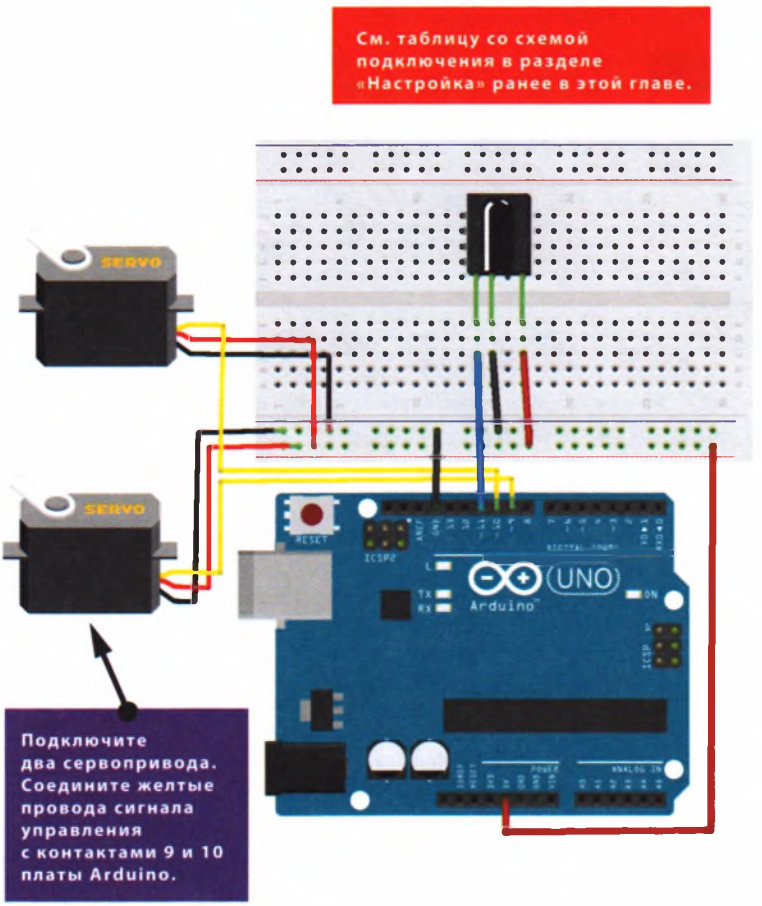


РИСУНОК 11.3

Принципиальная схема устройства для дистанционного управления сервоприводами

СКЕТЧ

Обязательно проверьте, что в коде вы указали значения, полученные на шаге 3 в разделе «Настройки» ранее в этой главе, а не те, которые приведены в книге. При изменении значений на собственные не удаляйте символы `0x`, которые определяют начало шестнадцатеричного кода. Например, нажав первую кнопку в этом примере, я получил ее шестнадцатеричное значение FFA05F, которое прописал в скетче следующим образом:

```
unsigned long Value1 = 0xFFA05F;
```

В этом проекте осуществляется дистанционное управление сервоприводами, но вы можете адаптировать код для удаленного управления любыми компонентами, подключаемыми к контактам с режимом HIGH, например светодиодом или пьезоизлучателем.

Код скетча обращается к библиотеке IRremote для считывания данных от ИК-датчика и к библиотеке Servo для приведения сервоприводов в движение. Значения первых двух кнопок настроены на вращение сервопривода оси *x* с ограничениями поворота на 70° влево и на 160° вправо. Значения третьей и четвертой кнопок настроены на вращение сервопривода оси *y* для управления наклоном вверх/вниз.

Если вы хотите адаптировать код для управления другим компонентом, измените код:

```
servo.write
```

на:

```
digitalWrite(pin, HIGH)
```

Полный код скетча приведен ниже:

```
/* Авторские права на IR Library принадлежат Кену Ширриффу,
   2009.
   Код использован с его согласия.
   http://arcfn.com
*/

#include <Servo.h>                  // Импорт библиотеки Servo
#include <IRremote.h>               // Импорт библиотеки IRremote

unsigned long Value1 = 0xFFA05F;    // Измените это значение
                                    // на собственное
unsigned long Value2 = 0xFF50AF;    // Измените это значение
                                    // на собственное
unsigned long Value3 = 0xFF807F;    // Измените это значение
                                    // на собственное
```

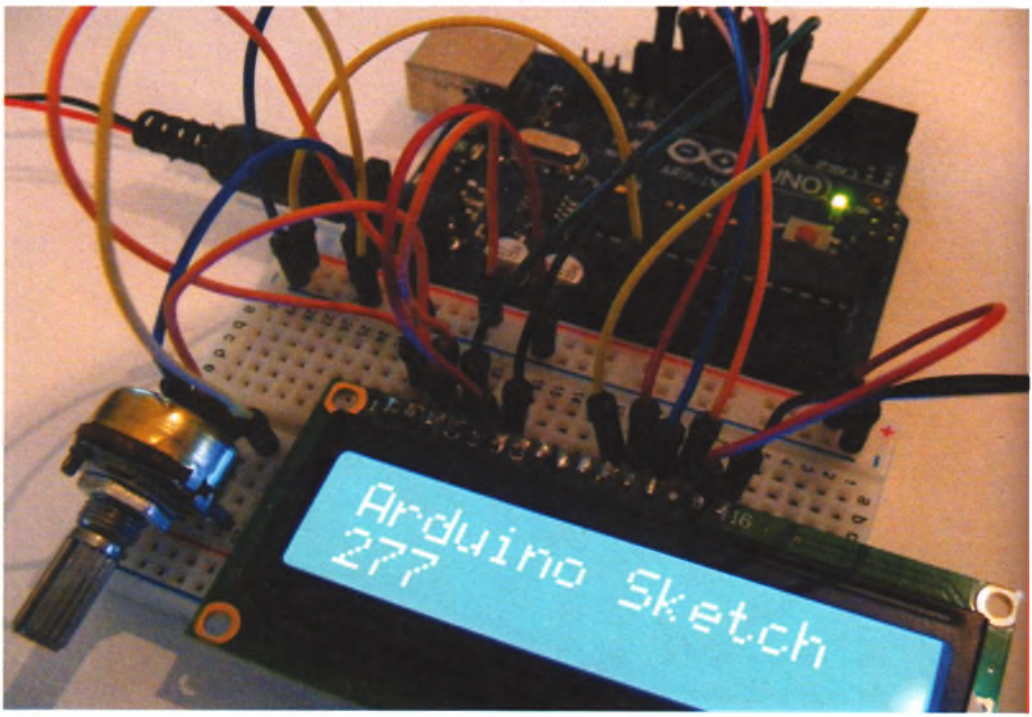
```
unsigned long Value4 = 0xFF609F;    // Измените это значение
                                    // на собственное

int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);
decode_results results;
Servo servo1;
Servo servo2;

void setup() {                      // Настройка алгоритма работы
   Serial.begin(9600);
   irrecv.enableIRIn();             // Включение ИК-датчика
   servo1.attach(10);               // Контакт, к которому подключен
                                    // сервопривод 1
   servo2.attach(9);                // Контакт, к которому подключен
                                    // сервопривод 2
}

void loop() {                       // Зацикленный алгоритм,
                                    // выполняемый непрерывно
   if (irrecv.decode(&results)) {
      Serial.println(results.value, HEX);
      irrecv.resume();              // Получение следующего значения
   }
   if (results.value == Value1) {   // Если полученный код
                                    // соответствует значению Value1,
                                    // управлять соответствующим
                                    // сервоприводом в указанном
                                    // направлении
      servo1.write(160);
   }
   else if (results.value == Value2) {     // Если полученный код
                                           // соответствует значению
                                           // Value2, управлять
                                           // соответствующим
                                           // сервоприводом в указанном
                                           // направлении
      servo1.write(70);
   }
   else if (results.value == Value3) {
      servo2.write(70);
   }
   else if (results.value == Value4) {
      servo2.write(160);
   }
}
```

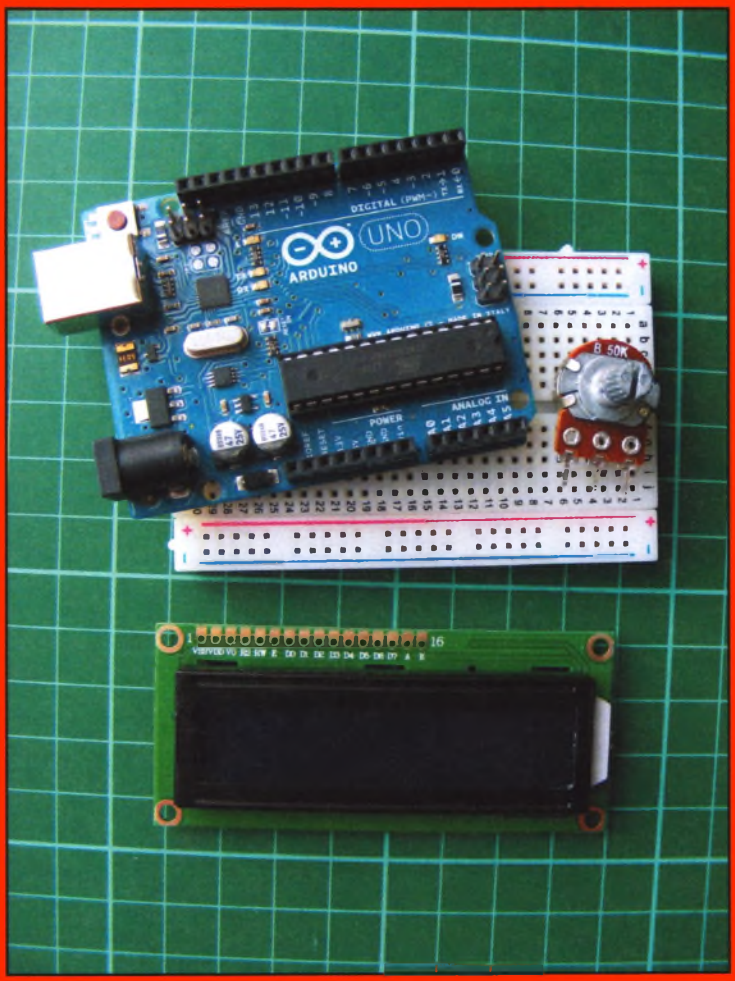
ЧАСТЬ 4

ОТОБРАЖЕНИЕ

ПРОЕКТ 12: ВЫВОД ДАННЫХ НА ЖК-ДИСПЛЕЙ

ЕСЛИ НА ЖК-ДИСПЛЕЙ ВЫВОДЯТСЯ ВАШИ СОБСТВЕННЫЕ СООБЩЕНИЯ, ЭТО НЕ ТОЛЬКО ПРИЯТНО, НО И ОЧЕНЬ ПОЛЕЗНО.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- ЖК-дисплей размером 16×2 (совместимый с Hitachi HD44780)
- Потенциометр 50 кОм

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- LiquidCrystal

ПРИНЦИП РАБОТЫ

Жидкокристаллический дисплей (ЖКД, LCD) сделан из двух подложек поляризационного материала с раствором жидких кристаллов между ними. Ток, проходящий через раствор, создает изображение или, в данном случае, символы. Для этого проекта вам понадобится ЖК-дисплей, совместимый с драйвером Hitachi HD44780, для работы с Arduino. Таких дисплеев существует великое множество, отличаются они 16-контактным интерфейсом.

Мы будем использовать библиотеку LiquidCrystal для передачи символов на ЖК-дисплей. Библиотека LiquidCrystal сопоставляет символы и использует команды print.lcd для копирования сообщения из скетча на дисплей.

Сначала нужно подготовить ЖК-дисплей.

ПОДГОТОВКА ЖК-ДИСПЛЕЯ

ЖК-дисплей, скорее всего, потребует сборку. Ваш дисплей, вероятно, будет содержать 16 отверстий (как показано на рис. 12.1) и отдельно ленту штырьковых соединителей.

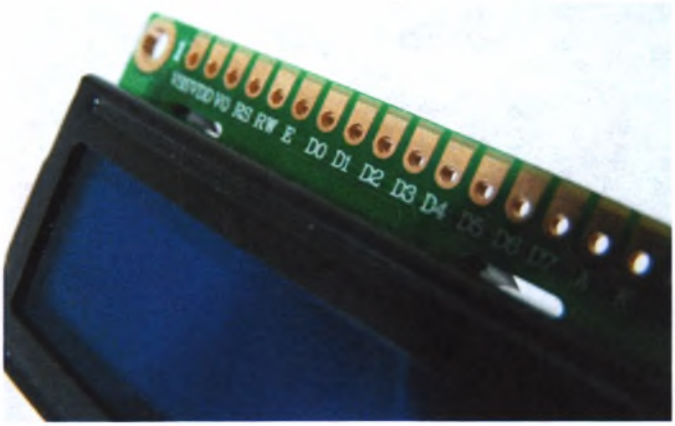


РИСУНОК 12.1

ЖК-дисплей оборудован в верхней части 16 контактами

Возьмите ленту соединителей и отломите ряд из 16 штырьков. Вставьте короткие ножки штырьков в 16 отверстий ЖК-дисплея. Вам нужно припаять их: сначала припаяйте крайние правые и крайние левые соединения, чтобы закрепить ленту на месте, и подождите, пока припой не застынет. Затем припаяйте все остальные места соединений по очереди, удерживая паяльник с припоем на каждом штырьке. Не держите паяльник слишком долго, чтобы не повредить микросхему — пары секунд достаточно. (Если вы никогда не паяли до этого, см. раздел «Краткое руководство по пайке» в проекте 0.)

СБОРКА

1. Установите ваш ЖК-дисплей на макетную плату, вставив штырьки в отверстия платы. Также установите потенциометр на макетную плату и используйте перемычки для подключения к ЖК-дисплею, Arduino и потенциометру, как показано в следующей таблице и на рис. 12.2. Модуль ЖК-дисплея оборудован тремя контактами заземления; их нужно подключить к шине заземления макетной платы.

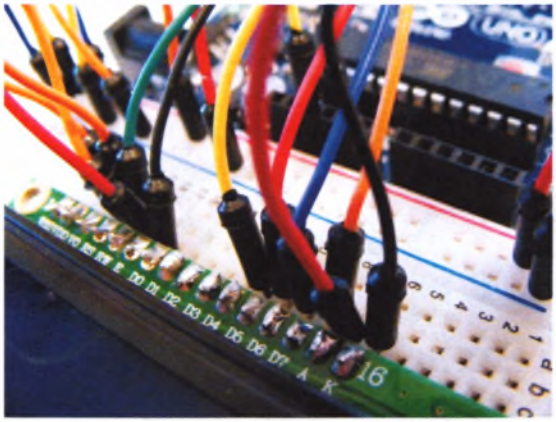


РИСУНОК 12.2

Подключение ЖК-дисплея и Arduino к макетной плате. Контакты 15 и 16 модуля ЖК-дисплея — это питание и заземление для подсветки дисплея

ЖК-ДИСПЛЕЙ	ARDUINO
1 штырь (VSS)	Контакт GND
2 штырь (VDD)	Контакт 5V
3 штырь (VO — контраст)	Центральный контакт потенциометра
4 штырь (RS)	Контакт 7
5 штырь (R/W)	Контакт GND
6 штырь (E — включение)	Контакт 8
7 штырь (D0)	Не используется
8 штырь (D1)	Не используется
9 штырь (D2)	Не используется
10 штырь (D3)	Не используется
11 штырь (D4)	Контакт 9
12 штырь (D5)	Контакт 10
13 штырь (D6)	Контакт 11
14 штырь (D7)	Контакт 12
15 штырь (A — BcL+)	Контакт 5V
16 штырь (K — BcL–)	Контакт GND

2. Центральный контакт потенциометра с сопротивлением 50 кОм подключается к контакту 3 (VO) ЖК-дисплея. Потенциометр управляет контрастностью дисплея. Поворачивайте ручку, пока не увидите на дисплее четкие символы. Теперь подключите один из внешних контактов потенциометра к заземлению, а другой — к питанию 5 В.

3. ЖК-дисплеи с подсветкой (см. рис. 12.3) имеют встроенные резисторы, но если у вас ЖК-дисплей без подсветки, нужно добавить резистор с сопротивлением 220 Ом между контактом 15 ЖК-дисплея и контактом питания 5 В. (На упаковке дисплея обычно указано, имеется ли подсветка или нет.)

РИСУНОК 12.3

ЖК-дисплей с подсветкой

4. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 12.5, и рис. 12.4, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

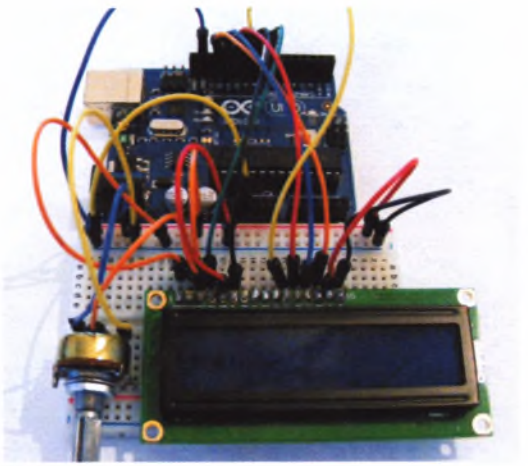


РИСУНОК 12.4

Готовая цепь

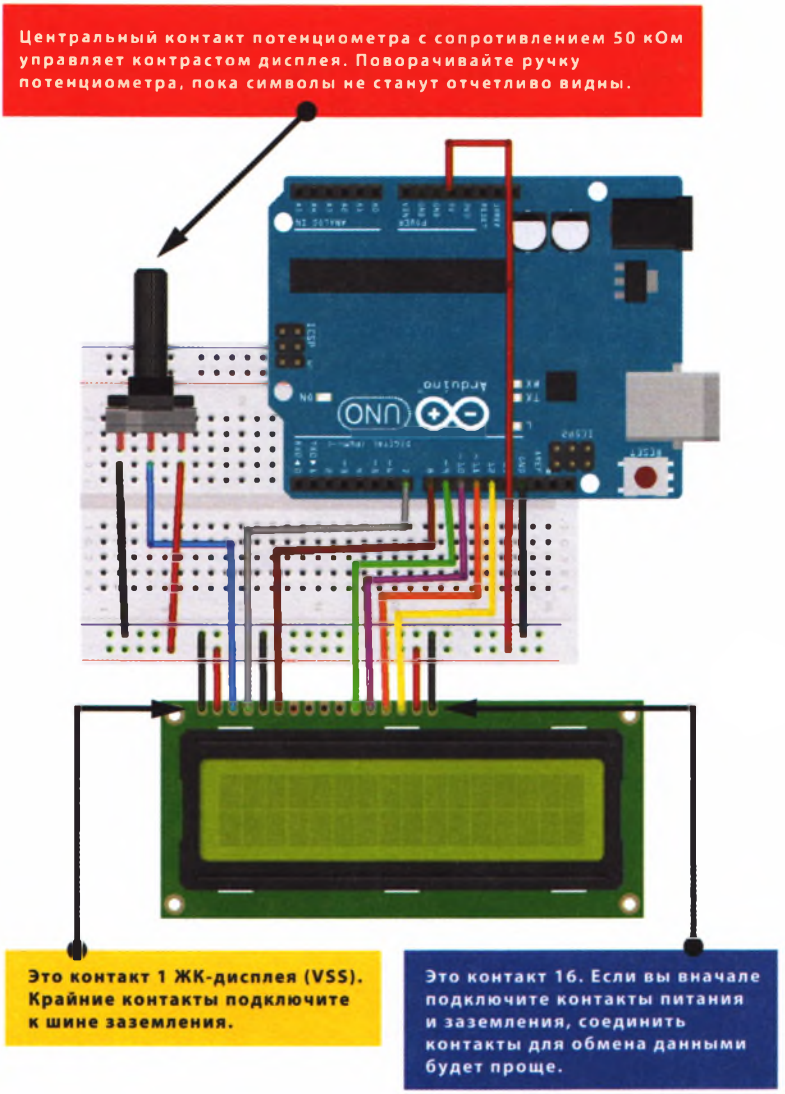


РИСУНОК 12.5

Принципиальная схема цепи устройства вывода данных на ЖК-дисплей

СКЕТЧ

Данный скетч доступен среди примеров, предустановленных со средой разработки Arduino. Загрузите его, выполнив команду меню **Файл ▸ Примеры ▸ LiquidCrystal ▸ Scroll** (File ▸ Examples ▸ LiquidCrystal ▸ Scroll). В скетче используется библиотека LiquidCrystal, поставляемая со средой

разработки Arduino, для передачи данных с платы Arduino на ЖК-дисплей. Вы можете изменить сообщение, заменив значение Arduino Sketch в строке ❷.

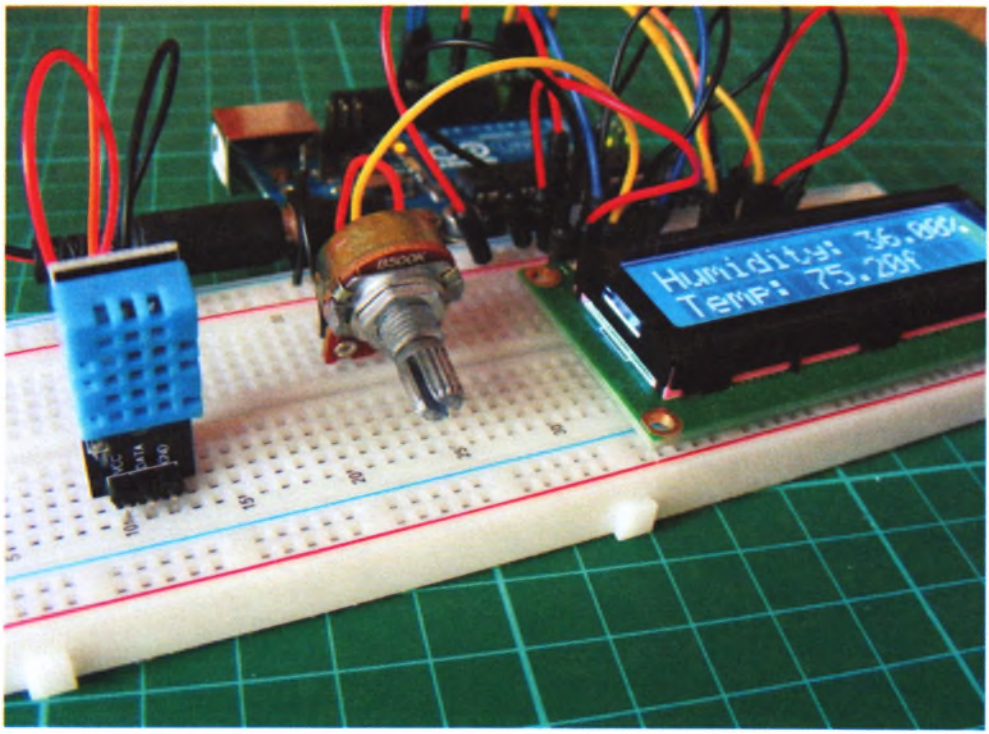
Чтобы использовать нашу цепь с указанным скетчем, мы также сменим в коде контакты ЖК-дисплея (12, 11, 5, 4, 3, 2) в строке ❶ на 7, 8, 9, 10, 11, 12, так как это используемые нами контакты. Я изменил код скетча, и если вы откроете приложенный к книге файл, вы увидите его в среде разработки Arduino с уже внесенными изменениями.

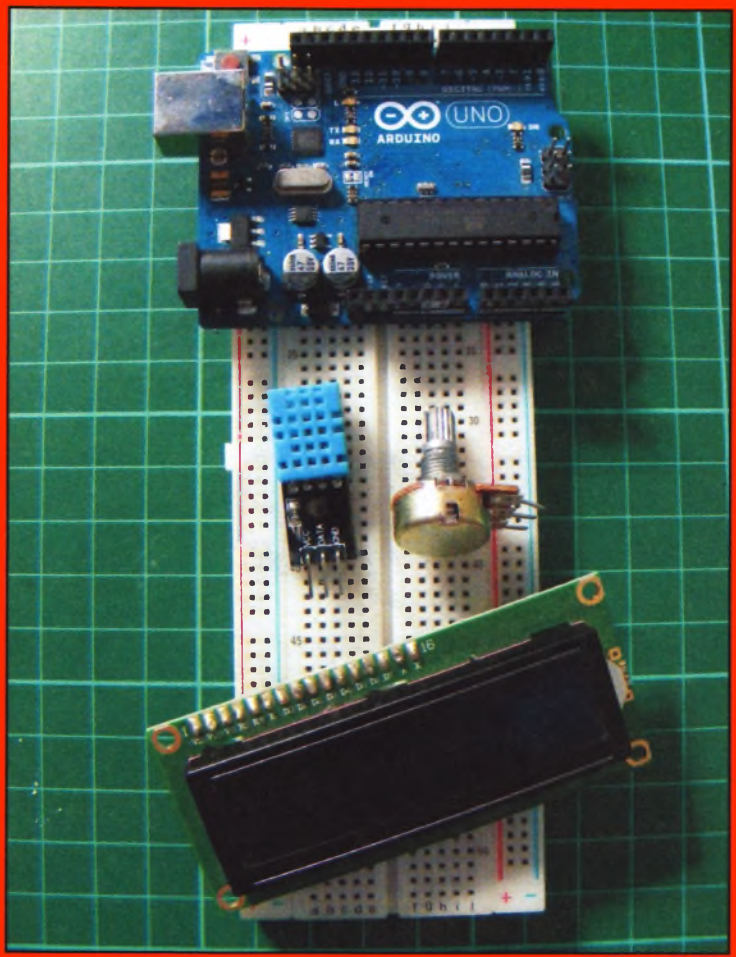
```
/*
Библиотека первоначально добавлена 18 апреля 2008
   Дэвидом А. Меллисом
   Изменена 5 июля 2009 Лимором Фрайдом (www.ladyada.net)
   Пример добавлен 9 июля 2009 Томом Иго
   Изменен 22 ноября 2010 Томом Иго
   Этот код - общественное достояние.
   http://www.Arduino.cc/en/Tutorial/LiquidCrystal

библиотека LiquidCrystal - scrollDisplayLeft() и scrollDisplayRight()

Демонстрирует пример использования ЖК-дисплея 16x2. Библиотека
LiquidCrystal поддерживается всеми ЖК-дисплеями, совместимыми
с драйвером Hitachi HD44780. Таких дисплеев существует великое
множество, отличаются они 16-контактным интерфейсом.

Этот скетч выводит на ЖК-дисплей текст "Arduino Sketch" и применяет
методы scrollDisplayLeft() и scrollDisplayRight() для прокрутки
текста.
*/
// Подключение кода библиотеки
#include <LiquidCrystal.h>

// Инициализировать библиотеку номерами контактов интерфейса
❶ LiquidCrystal lcd(7, 8, 9, 10, 11, 12);

   void setup() {
void setup() {
   // Установка количества столбцов и строк ЖК-дисплея
   lcd.begin(16, 2);
   // Вывод сообщения на ЖК-дисплей
❷    lcd.print("Arduino Sketch");
   delay(1000);
}

void loop() {
// Прокрутка на 13 позиций (длина строки) влево, чтобы скрыть строку
   for (int positionCounter = 0; positionCounter < 13;
   positionCounter++) {
      // Прокрутка на 1 позицию влево
      lcd.scrollDisplayLeft();
      // Задержка
      delay(150);
   }
```

```
  // Прокрутка на 29 позиций (длина строки + длина дисплея) вправо,
  // чтобы скрыть строку
  for (int positionCounter = 0; positionCounter < 29;
  positionCounter++) {
    // Прокрутка на 1 позицию вправо
    lcd.scrollDisplayRight();
    // Задержка
    delay(150);
  }
  // Прокрутка на 16 позиций (длина дисплея + длина строки) влево,
  // чтобы выровнять строку по центру
  for (int positionCounter = 0; positionCounter < 16;
  positionCounter++) {
    // Прокрутка на 1 позицию влево
    lcd.scrollDisplayLeft();
    // Задержка
    delay(150);
  }
  // Задержка в конце полного цикла
  delay(1000);
}
```

ПРОЕКТ 13: МЕТЕОСТАНЦИЯ

В ЭТОМ ПРОЕКТЕ ВЫ ПОСТРОИТЕ МЕТЕОСТАНЦИЮ ДЛЯ ИЗМЕРЕНИЯ ТЕМПЕРАТУРЫ И ВЛАЖНОСТИ, И ВЫВОДА РЕЗУЛЬТАТОВ НА ЖК-ДИСПЛЕЙ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Потенциометр с сопротивлением 50 кОм
- ЖК-дисплей размером 16×2 (совместимый с Hitachi HD44780)
- Датчик температуры и влажности DHT11

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- LiquidCrystal
- DHT

ПРИНЦИП РАБОТЫ

Датчик, используемый в этом проекте, — это относительно дешевая модель DHT11, показанная на рис. 13.1, которая измеряет и влажность, и температуру. В этом модуле используется емкостный датчик влажности и резистивный температурный датчик для считывания параметров окружающей среды. Модуль передает результаты замеров плате Arduino в виде электрических импульсов, а Arduino преобразует их в понятные человеку значения и выводит на дисплей. Для достижения наилучших результатов рекомендуется установить датчик снаружи помещения на открытом воздухе. А ЖК-дисплей расположить в помещении либо упаковать в прозрачный непромокаемый чехол или корпус, чтобы защитить от влияния внешних факторов.

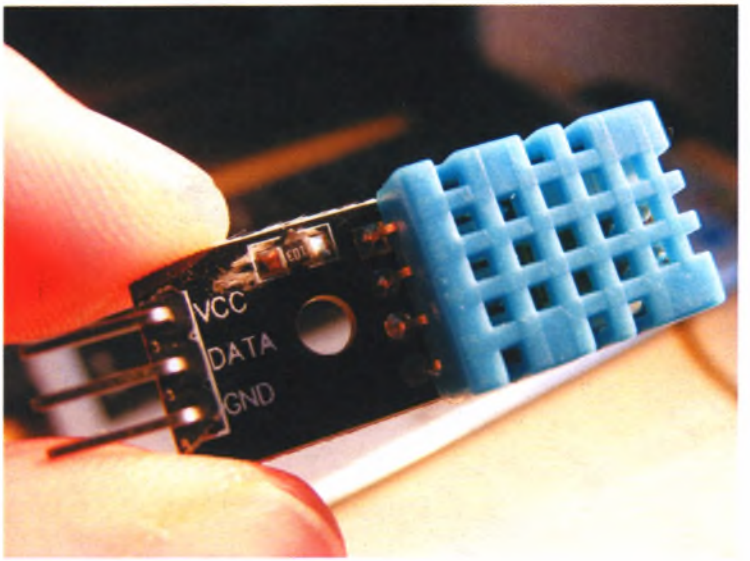


РИСУНОК 13.1

Датчик DHT11 измеряет как температуру, так и влажность

Модель датчика DHT11 оборудована тремя контактами, как показано на рис. 13.1. Непосредственно сам датчик на модуле содержит четыре ножки, среди которых контакт 3 не используется. Посмотрите список магазинов в конце книги, чтобы найти и приобрести датчик DHT11.

СБОРКА

1. Сначала подготовьте ЖК-дисплей в соответствии с инструкциями, приведенными в разделе «Подготовка ЖК-дисплея» в проекте 12. Установите датчик DHT11 на макетную плату. Контакты датчика DHT11 нумеруются от 1 до 3 справа налево, когда лицевая сторона обращена к вам. Подключите ножку 1 к шине питания 5 В, ножку 2 соедините непосредственно с контактом 8 платы Arduino и подключите ножку 3 к шине заземления.

DHT11	ARDUINO
Ножка 1 (VCC)	Контакт 5V
Ножка 2 (Data)	Контакт 8
Ножка 3 (GND)	Контакт GND

2. Установите ЖК-дисплей на макетную плату и подключите его контакты к Arduino, как показано в следующей таблице и на рис. 13.2. Шины заземления и питания 5 В будут содержать несколько подключений.

ЖК-ДИСПЛЕЙ	ARDUINO
1 штырь (VSS)	Контакт GND
2 штырь (VDD)	Контакт 5V
3 штырь (VO — контраст)	Центральный контакт потенциометра
4 штырь (RS)	Контакт 12
5 штырь (RW)	Контакт GND
6 штырь (E — включение)	Контакт 11
7 штырь (D0)	Не используется
8 штырь (D1)	Не используется
9 штырь (D2)	Не используется
10 штырь (D3)	Не используется
11 штырь (D4)	Контакт 5
12 штырь (D5)	Контакт 4
13 штырь (D6)	Контакт 3
14 штырь (D7)	Контакт 2
15 штырь (A — BcL+)	Контакт 5V
16 штырь (K — BcL–)	Контакт GND

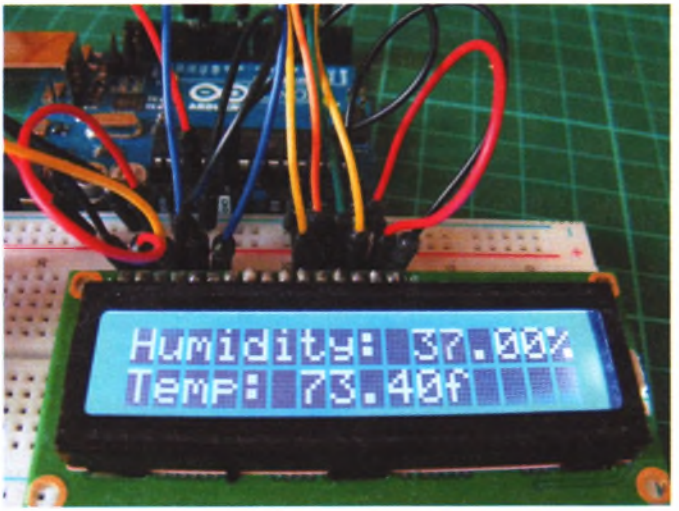


РИСУНОК 13.2

Подключение ЖК-дисплея к макетной плате

3. Установите потенциометр на макетную плату, как показано на рис. 13.3, и подключите его центральный контакт к контакту 3 ЖК-дисплея. Подключите один внешний контакт потенциометра к шине питания 5 В, а другой — к шине заземления.

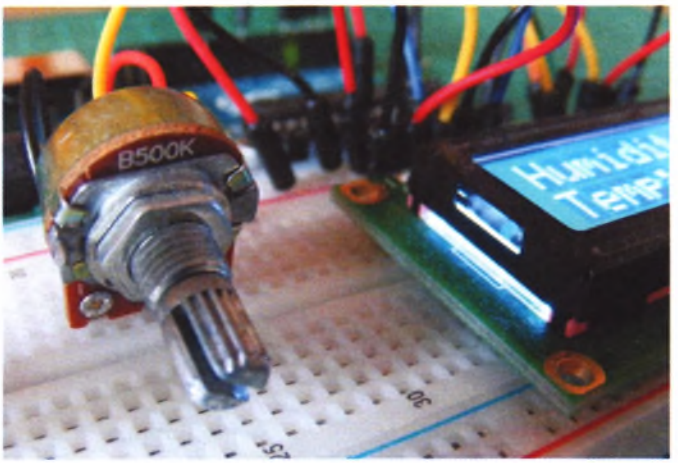


РИСУНОК 13.3

Подключение потенциометра к макетной плате

4. Подключите шины макетной платы к контактам GND и 5V платы Arduino. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 13.4, и загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

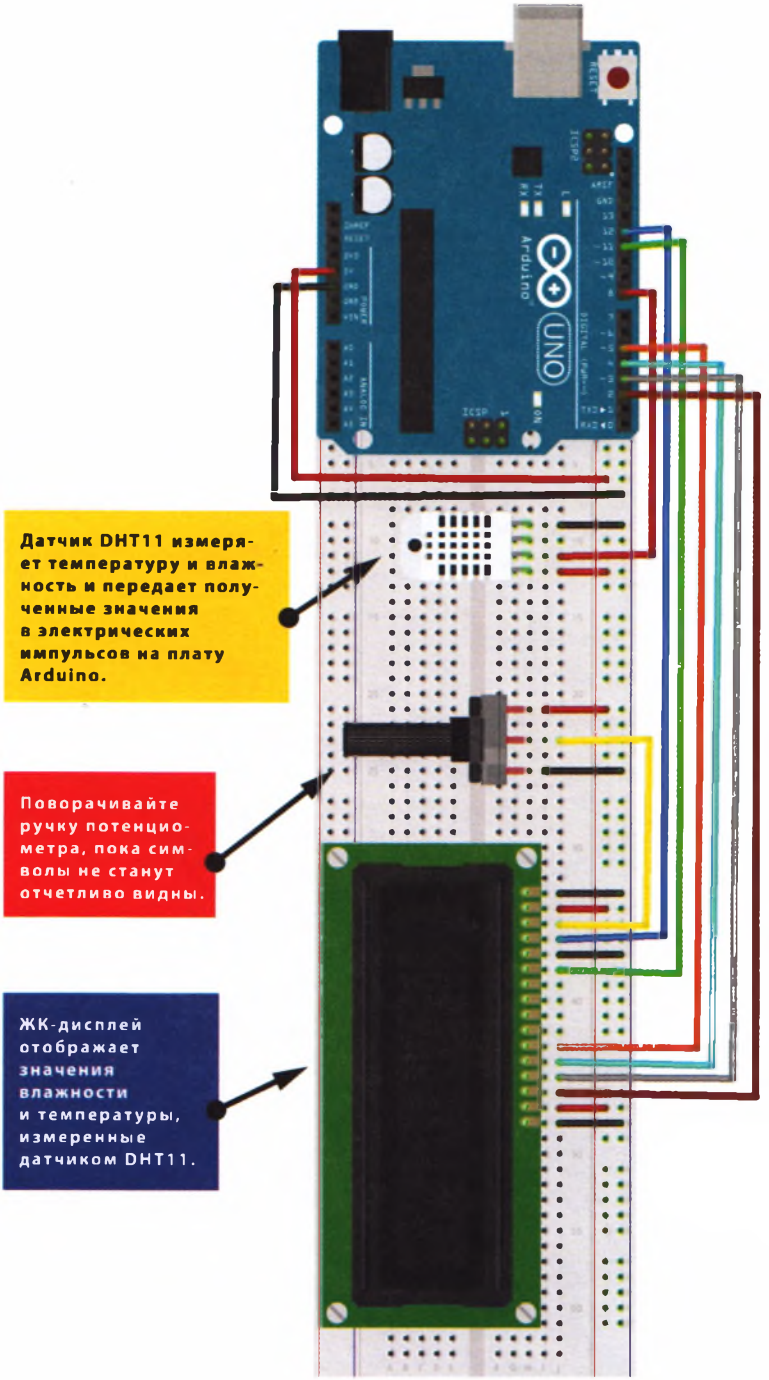


РИСУНОК 13.4

Принципиальная схема цепи метеостанции

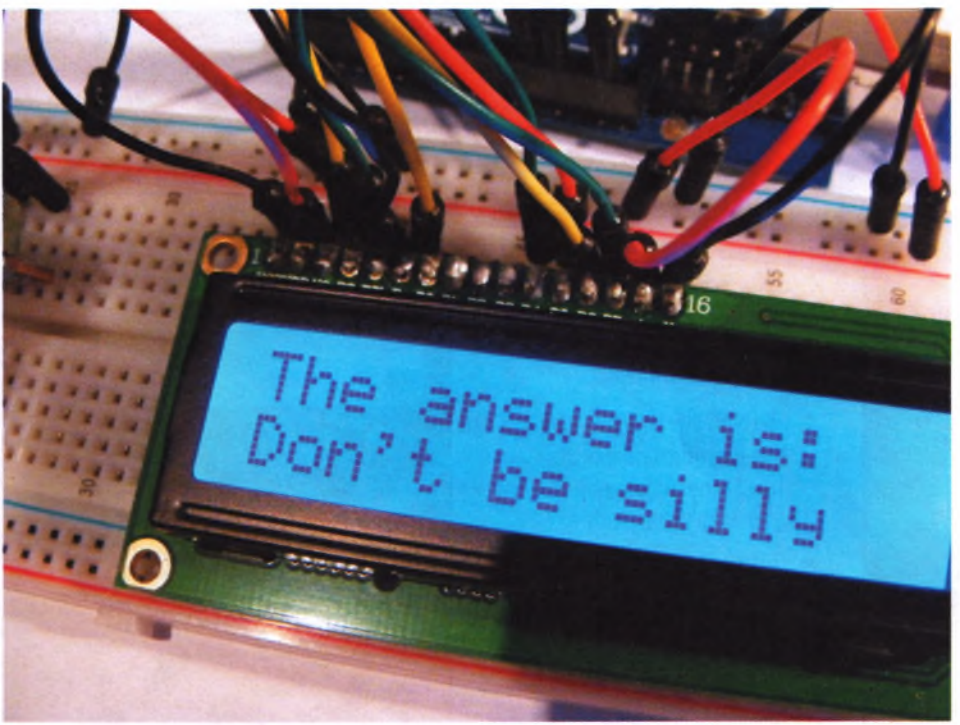
СКЕТЧ

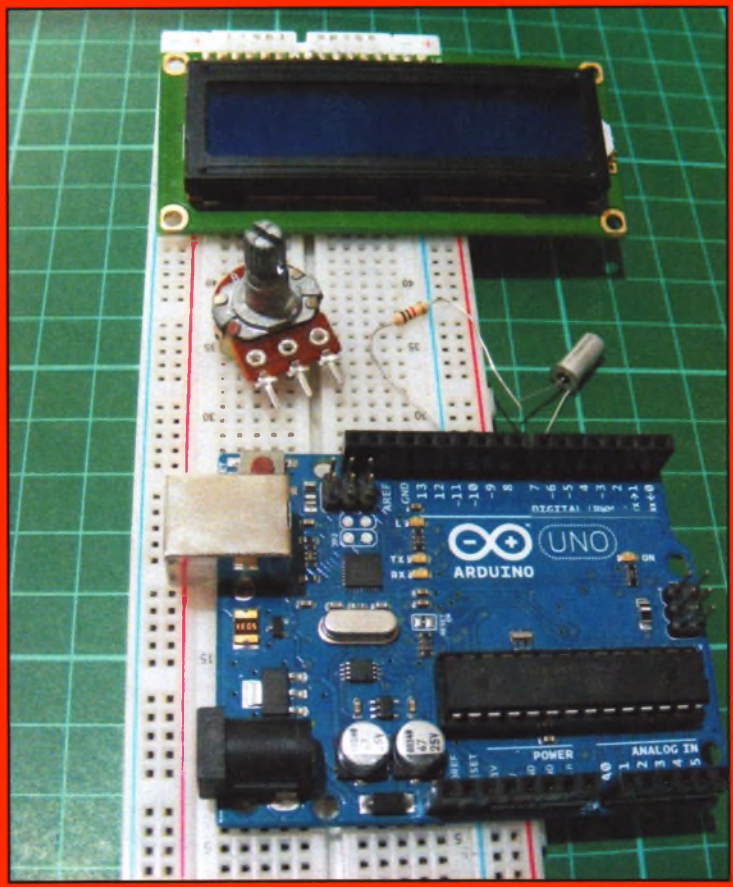
В этом скетче используется библиотека LiquidCrystal, доступная в составе дистрибутива среды разработки Arduino, и библиотека DHT, которую вам нужно скачать в архиве по адресу *eksmo.ru/files/arduino_geddes.zip* (и установить в среде разработки Arduino, как это продемонстрировано в разделе «Библиотеки» проекта 0). Библиотека DHT позволяет управлять работой датчика, а библиотека LiquidCrystal выводит полученные данные на дисплей.

```
/* Скетч для тестирования различных датчиков влажности/температуры
типа DHT. Создан пользователем ladyada, этот код - общественное
достояние. */

#include <LiquidCrystal.h>
#include "DHT.h"      // Вызов библиотеки DHT
#define DHTPIN 8      // Контакт, к которому подключен датчик DHT
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
#define DHTTYPE DHT11         // Определение типа модуля DHT
DHT dht(DHTPIN, DHTTYPE);     // Команда библиотеке DHT.h

void setup() {
   dht.begin();       // Включение датчика
   lcd.begin(16, 2);  // ЖК-дисплей отображает 16 символов в 2 строках
}

void loop() {
   float h = dht.readHumidity();     // Значение влажности
   float t = dht.readTemperature();  // Значение температуры
   t = t * 9 / 5 + 32;               // Преобразование значения
                                     // из Цельсия в Фаренгейты
   if (isnan(t) || isnan(h)) {       // Проверка работоспособности
                                     // датчика DHT
      lcd.setCursor(0, 0);
      lcd.print("Failed to read from DHT");       // Если датчик DHT не
                     // работает, выводится это сообщение
   } else {          // В противном случае на дисплей выводятся
                     // результаты замеров
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Humidity: ");
      lcd.print(h);
      lcd.print("%");
      lcd.setCursor(0, 1);
      lcd.print("Temp: ");
      lcd.print(t);
      lcd.print("f");
   }
}
```

ПРОЕКТ 14: ПРЕДСКАЗАТЕЛЬ СУДЬБЫ

В ЭТОМ ПРОЕКТЕ МЫ СОЗДАДИМ ЭЛЕКТРОННУЮ ВЕРСИЮ КЛАССИЧЕСКОГО АТРИБУТА ЛЮБОЙ УВАЖАЮЩЕЙ СЕБЯ ГАДАЛКИ — ШАРА СУДЬБЫ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- ЖК-дисплей размером 16×2 (совместимый с Hitachi HD44780)
- Датчик наклона
- Потенциометр с сопротивлением 50 кОм
- Резистор с сопротивлением 1 кОм

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- LiquidCrystal

ПРИНЦИП РАБОТЫ

Шар судьбы (также известный как магический шар или шар вопросов и ответов) — игрушка, созданная в 1950-е годы и состоящая из полого шара, в котором икосаэдр (фигура с 20 гранями) плавает в темной жидкости. Когда вы задаете вопрос и встряхиваете шар, одна сторона фигуры всплывает, и вы видите ответ в окошке шара.

Для этого проекта понадобится датчик наклона, показанный на рис. 14.1. Датчик наклона состоит из металлической капсулы с шариком внутри. Шарик перекатывается в капсуле и замыкает или размыкает цепь. Таким образом, датчик выдает простой цифровой сигнал: логический ноль или единицу, в зависимости от того, в какую сторону наклонена капсула. Существует множество датчиков наклона, и все они основаны на одном и том же принципе. В этом проекте вам нужно будет задать вопрос и потрясти датчик. Когда датчик сработает, произойдет подключение к плате Arduino, будет выбран случайный ответ из восьми предустановленных и выведен на ЖК-дисплей.

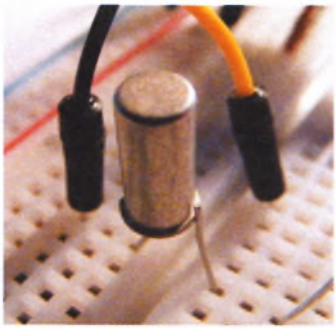


РИСУНОК 14.1

Датчик наклона, установленный на макетную плату

Потенциометр управляет контрастностью ЖК-дисплея.

СБОРКА

1. Подготовьте ЖК-дисплей в соответствии с инструкциями, приведенными в разделе «Подготовка ЖК-дисплея» в проекте 12.
2. Установите ЖК-дисплей на макетную плату, вставив штырьки в соответствующие отверстия платы. Также установите на макетную плату потенциометр и с помощью перемычек подключите ЖК-дисплей и потенциометр к плате Arduino.

ЖК-ДИСПЛЕЙ	ARDUINO
1 штырь (VSS)	Контакт GND
2 штырь (VDD)	Контакт 5V
3 штырь (VO — контраст)	Центральный контакт потенциометра
4 штырь (RS)	Контакт 12

ЖК-ДИСПЛЕЙ	ARDUINO
5 штырь (R/W)	Контакт GND
6 штырь (E — включение)	Контакт 11
7 штырь (D0)	Не используется
8 штырь (D1)	Не используется
9 штырь (D2)	Не используется
10 штырь (D3)	Не используется
11 штырь (D4)	Контакт 5
12 штырь (D5)	Контакт 4
13 штырь (D6)	Контакт 3
14 штырь (D7)	Контакт 2
15 штырь (A — BcL+)	Контакт 5V
16 штырь (K — BcL–)	Контакт GND

3. Не забудьте использовать шины макетной платы, чтобы создать несколько подключений к контакту GND платы Arduino, как показано на рис. 14.2.

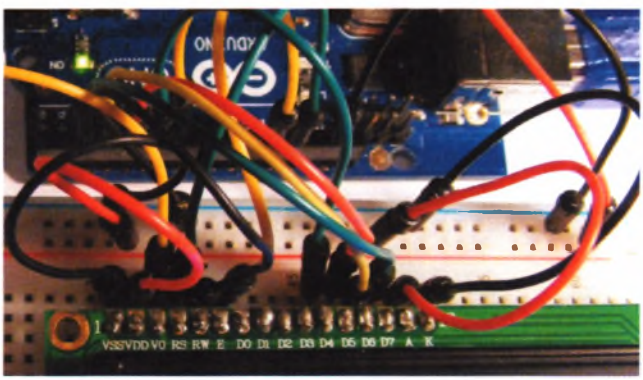


РИСУНОК 14.2

ЖК-дисплей подключен к плате Arduino

4. Центральный контакт потенциометра должен быть подключен к контакту 3 ЖК-дисплея (VO). Теперь подключите один из внешних контактов потенциометра к контакту GND, а другой — к контакту 5V платы Arduino. Потенциометр будет управлять контрастностью ЖК-дисплея.
5. Установите датчик наклона на макетную плату и подключите одну его ножку через резистор с сопротивлением 1 кОм к контакту 6 платы Arduino, а другую ножку — к контакту GND.

ДАТЧИК НАКЛОНА	ARDUINO
Ножка 1	Контакт 6 через резистор с сопротивлением 1 кОм
Ножка 2	Контакт GND

6. Подключите шины макетной платы к контактам 5V и GND платы Arduino для обеспечения питания.
7. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 14.3, и загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

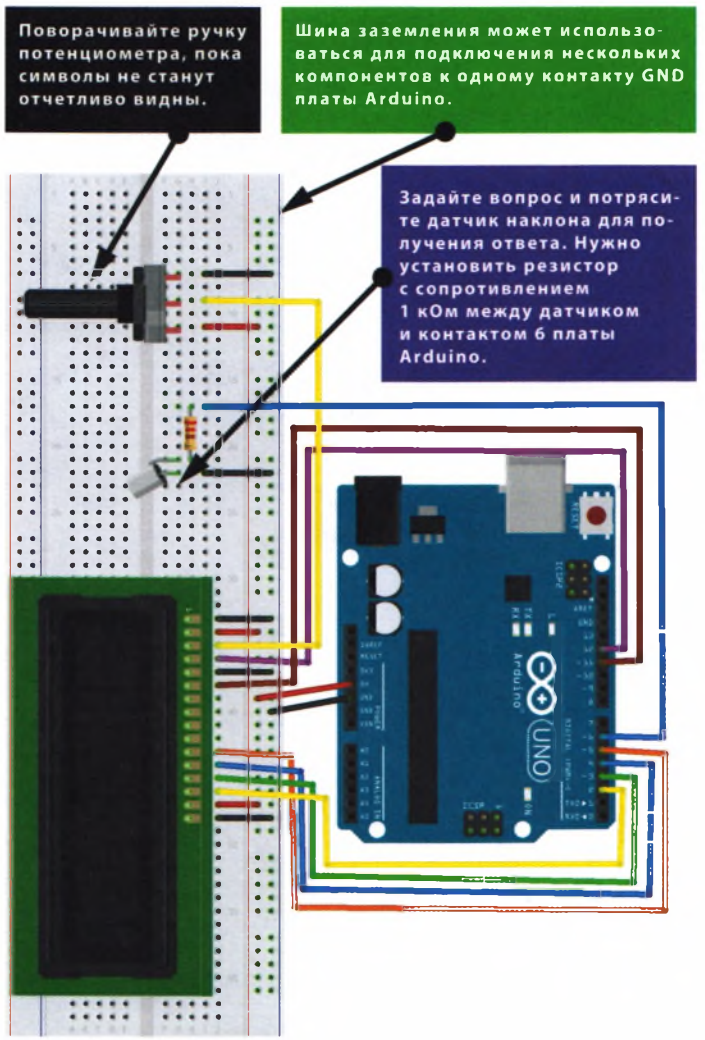


РИСУНОК 14.3

Принципиальная схема цепи предсказателя судьбы

СКЕТЧ

Код в этом проекте довольно прост. Когда вы включаете Arduino, на ЖК-дисплее появляется текст Ask a Question. Встряхивание датчика наклона приводит к выполнению кода скетча, и плата Arduino выбирает случайный ответ из восьми возможных (case в диапазоне от 0 до 7).

Ниже показана строка, в которой это происходит:

```
reply = random(8);
```

Чтобы добавить свои собственные ответы, измените значение 8 на количество возможных ответов, затем добавьте собственные ответы в том же формате, как остальные:

```
case 8:
  lcd.print("You betcha");
break;
```

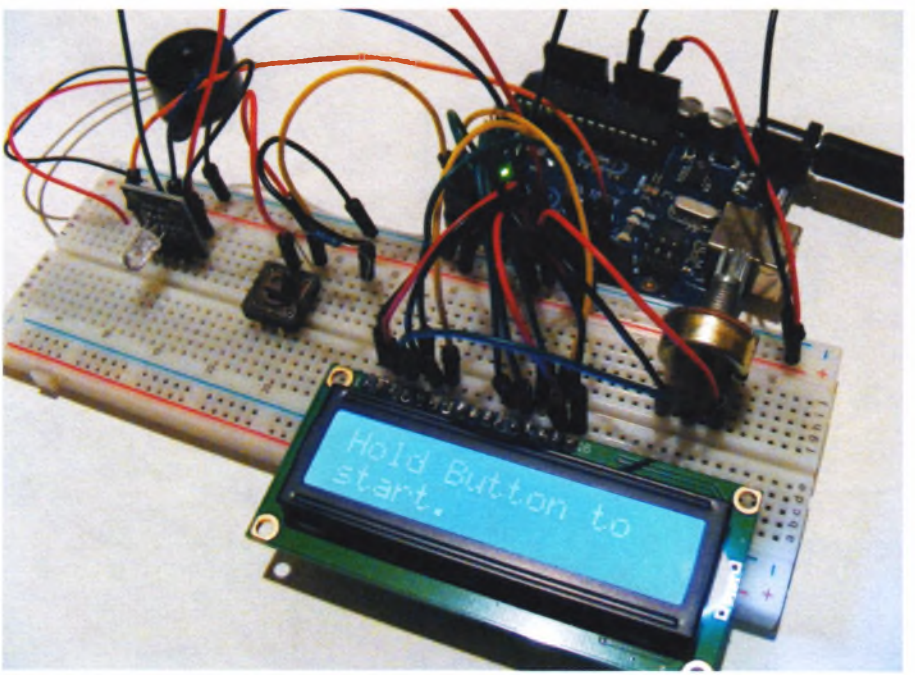
Ниже показан полный код скетча:

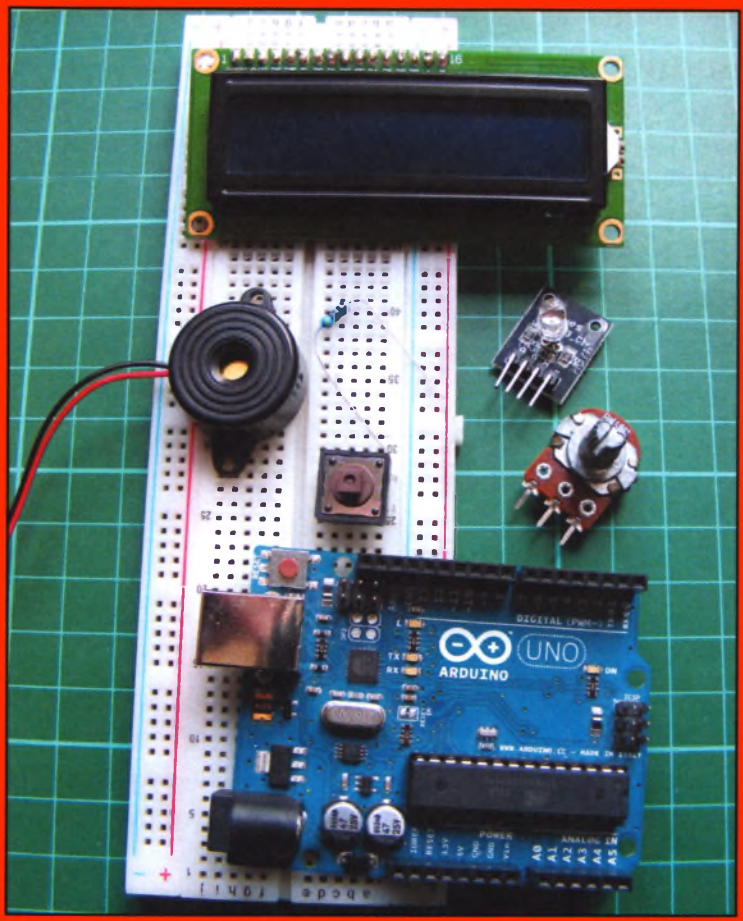
```
/* Создан 18 сентября 2012 Скоттом Фитцджеральдом
   http://arduino.cc/starterKit
   Этот код - общественное достояние
*/

#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);     // Контакты, к которым
                                           // подключен ЖК-дисплей

const int switchPin = 6;     // Контакт, к которому подключен датчик
int switchState = 0;
int prevSwitchState = 0;
int reply;

void setup() {
   lcd.begin(16, 2);
   pinMode(switchPin, INPUT);        // Перевод контакта датчика в режим
                                     // ввода
   lcd.print("FORTUNE TELLER");      // Это выводится в строке 1
   lcd.setCursor(0, 1);
   lcd.print("Ask a Question");      // Это выводится в строке 2
}

void loop() {
   switchState = digitalRead(switchPin);   // Считывание значения
                                           // с контакта датчика
   if (switchState != prevSwitchState) {
      if (switchState == LOW) {      // Если цепь разорвана, дать ответ
         reply = random(8);          // Повторить варианты с 1 по 8 ниже
         lcd.clear();
```

```
        lcd.setCursor(0, 0);
        lcd.print("The answer is: ");      // Это выводится на дисплей
        lcd.setCursor(0, 1);
        switch (reply) {                   // Ответом будет одним из
                                           // следующих вариантов
           case 0:
              lcd.print("Yes");
              break;

           case 1:
              lcd.print("Probably");
              break;

           case 2:
              lcd.print("Definitely");
              break;

           case 3:
              lcd.print("Don't be silly");
              break;

           case 4:
              lcd.print("Of course");
              break;

           case 5:
              lcd.print("Ask again");
              break;

           case 6:
              lcd.print("Doubtful");
              break;

           case 7:
              lcd.print("No");
              break;
        }
     }
   }
   prevSwitchState = switchState;   // Обнуление значения датчика
}
```

ПРОЕКТ 15: ИГРА НА СКОРОСТЬ

В ЭТОМ ПРОЕКТЕ МЫ СОЗДАДИМ ИГРУ НА СКОРОСТЬ РЕАКЦИИ. ЭТО БУДЕТ УСТРОЙСТВО ИЗ ЖК-ДИСПЛЕЯ, RGB-СВЕТОДИОДА И ПЬЕЗОИЗЛУЧАТЕЛЯ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- ЖК-дисплей размером 16×2 (совместимый с Hitachi HD44780)
- RGB-светодиод (с модулем)
- Пьезоизлучатель
- Тактовая четырехконтактная кнопка
- Потенциометр с сопротивлением 50 кОм
- Резистор с сопротивлением 220 Ом

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- LiquidCrystal

ПРИНЦИП РАБОТЫ

Начинаем игру, нажав и удерживая кнопку. RGB-светодиод загорается и светится разными цветами. Ваша цель — отреагировать как можно быстрее и отпустить кнопку, когда он загорится красным цветом. На ЖК-дисплее будет показана скорость вашей реакции в миллисекундах, с момента включения красного цвета до того, как вы отпустили кнопку (см. рис. 15.1).

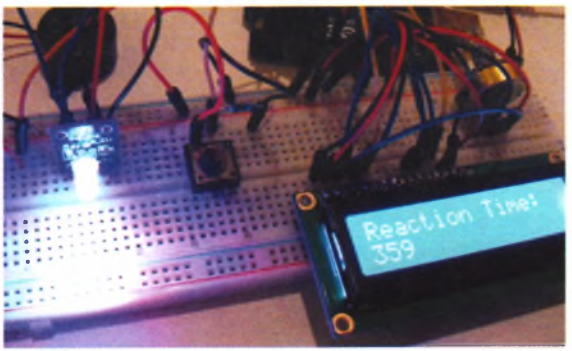


РИСУНОК 15.1

Когда вы отпустите кнопку, время реакции отобразится на ЖК-дисплее

Пьезоизлучатель будет отвлекать вас, издавая звуки в случайном порядке. Если вы отпустите кнопку слишком рано, на ЖК-дисплее появится соответствующее сообщение, и вам придется начать игру заново. RGB-светодиод на самом деле состоит из трех светодиодов в одном корпусе: красного, зеленого и синего цветов (см. рис. 15.2).

РИСУНОК 15.2

RGB-светодиод может светиться красным, зеленым и синим цвето

RGB — аддитивная цветовая модель. Это означает, что, комбинируя свет двух или более цветов, мы можем создавать другие цвета. Красный, зеленый и синий — это основные цвета, используемые для получения других оттенков, как показано на рис. 15.3.

Давайте рассмотрим RGB-светодиод более детально. На рис. 15.4 показан прозрачный светодиод с общим катодом. Обратите внимание, что светодиод имеет четыре ножки вместо привычных двух: по одной для красного, зеленого и синего, а последняя — либо катод, либо анод. В данном случае самым длинным выводом является катод, и он подключается к заземлению.

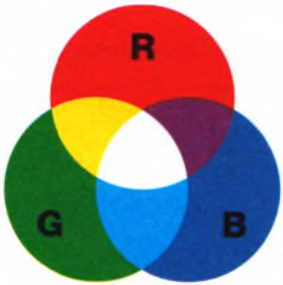


РИСУНОК 15.3
В модели RGB цвета смешиваются аддитивным образом.

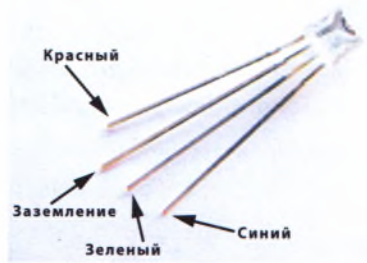


РИСУНОК 15.4
RGB-светодиод имеет четыре ножки вместо обычных двух

RGB-светодиод, используемый в этом проекте, уже установлен на модуль со встроенными резисторами, что позволяет нам сэкономить место на макетной плате.

СБОРКА

1. Подготовьте ЖК-дисплей в соответствии с инструкциями, приведенными в разделе «Подготовка ЖК-дисплея» в проекте 12.
2. Установите ЖК-дисплей, вставив его штырьки в отверстия макетной платы. Также установите на макетную плату потенциометр и с помощью перемычек подключите ЖК-дисплей и потенциометр к плате Arduino.

ЖК-ДИСПЛЕЙ	ARDUINO
1 штырь (VSS)	Контакт GND
2 штырь (VDD)	Контакт 5V
3 штырь (VO — контраст)	Центральный контакт потенциометра
4 штырь (RS)	Контакт 11
5 штырь (R/W)	Контакт GND
6 штырь (E — включение)	Контакт 12
7 штырь (D0)	Не используется
8 штырь (D1)	Не используется
9 штырь (D2)	Не используется
10 штырь (D3)	Не используется
11 штырь (D4)	Контакт 5
12 штырь (D5)	Контакт 4
13 штырь (D6)	Контакт 3
14 штырь (D7)	Контакт 2
15 штырь (A — BcL+)	Контакт 5V
16 штырь (K — BcL–)	Контакт GND

3. Центральный контакт потенциометра должен быть подключен к контакту 3 ЖК-дисплея (VO). Теперь подключите один из внешних контактов потенциометра к контакту GND, а другой — к контакту 5V платы Arduino. Потенциометр управляет контрастностью ЖК-дисплея.

4. Установите кнопку на макетную плату так, чтобы она перекрыла канавку в центре. Метки контактов показаны на рис. 15.5.

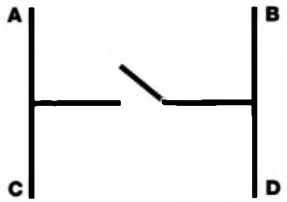


РИСУНОК 15.5

Кнопка перекрывает канавку в центре макетной платы

Подключите ножку A кнопки через резистор с сопротивлением 220 Ом к контакту GND, ножку C — к контакту 9, а ножку D — к контакту 5V платы Arduino (см. проект 1 для получения подробной информации о том, как работают тактовые кнопки).

КНОПКА	ARDUINO
Ножка A	Контакт GND через резистор с сопротивлением 220 Ом
Ножка C	Контакт 9
Ножка D	Контакт 5V

5. Установите модуль RGB-светодиода и подключите ножку красного светодиода к контакту 8 платы Arduino, зеленого — к контакту 6, синего — к контакту 7, а ножку питания (+) — к контакту 5V.

RGB-СВЕТОДИОД	ARDUINO
Ножка R	Контакт 8
Ножка G	Контакт 6
Ножка B	Контакт 7
Ножка +*	Контакт 5V

* Если на вашем модуле используется штырь – (катод) вместо + (анод), его следует подключать к контакту GND платы Arduino *(прим. ред.)*.

6. Подключите красный провод пьезоизлучателя непосредственно к контакту 13 платы Arduino, а черный провод — к заземлению.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Красный провод	Контакт 13
Черный провод	Контакт GND

7. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 15.6, и загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

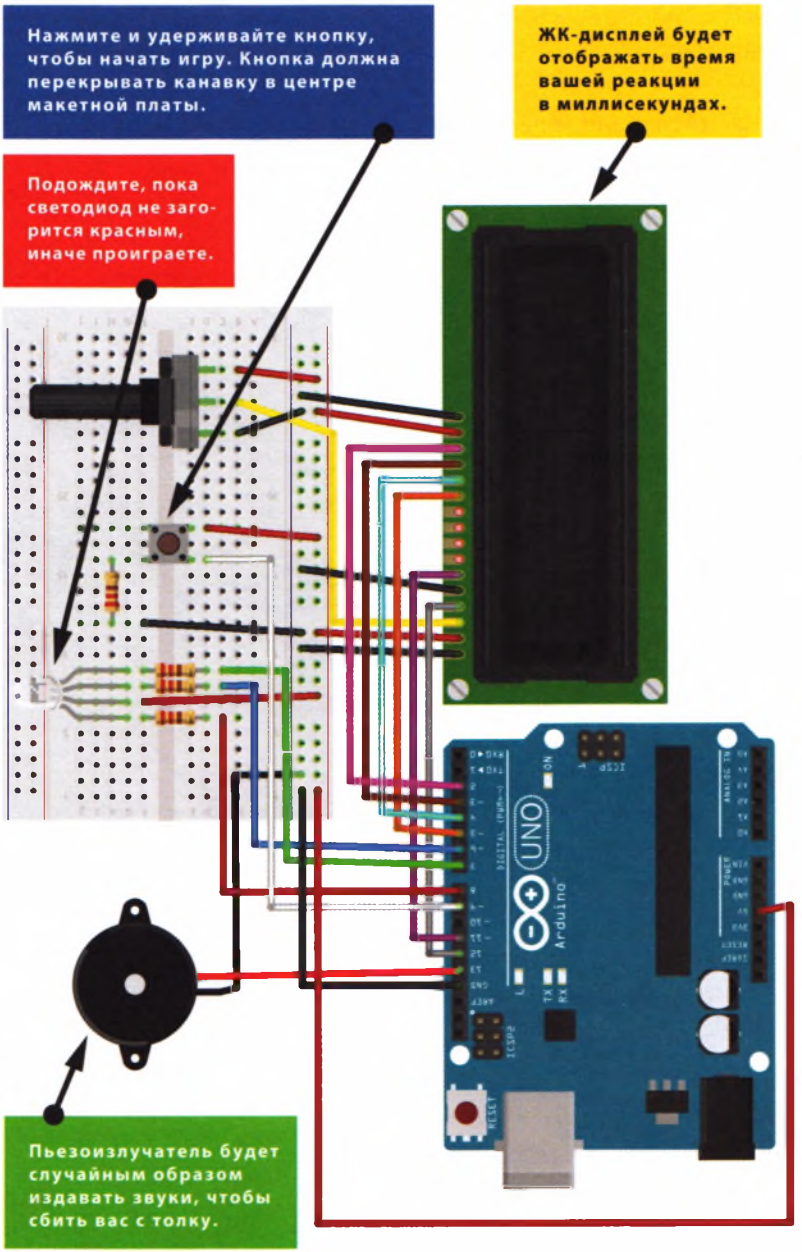


РИСУНОК 15.6

Принципиальная схема цепи игры на скорость реакции. Удобнее сначала подсоединить контакты 5V и GND, а затем уже добавлять перемычки для передачи данных

СКЕТЧ

Когда вы нажимаете и удерживаете кнопку, светодиод мигает случайными цветами и в какой-то момент становится красным. Время, на протяжении которого горит каждый цвет, устанавливается случайным образом, равно как и продолжительность пауз между цветами. Это означает, что вы не можете вычислить последовательность свечения цветов и предсказать, когда светодиод загорится красным.

Вы можете усложнить игру, увеличив продолжительность интервалов в следующей строке кода скетча:

```
    PSE = random(500, 1200);
```

Вот полный скетч:

```
// Создано Стивеном Де Ланнойем и используется с его согласия
// http://www.wingbike.nl
// Применимо для модуля RGB-светодиода с общим анодом
// (и тремя катодами: R, G, B)
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
int LEDR = 8;          // Контакт, к которому подключен красный
                       // светодиод
int LEDB = 7;          // Контакт, к которому подключен синий светодиод
int LEDGr = 6;         // Контакт, к которому подключен зеленый
                       // светодиод
int Button = 9;        // Контакт, к которому подключена кнопка
int COLOR;             // Переменная цвета
int Beep;
int PSE;               // Переменная паузы
int TME;               // Время
int RTME = 0;          // Время реакции

void setup() {
   lcd.begin(16, 2);
   pinMode(LEDR, OUTPUT);    // Перевод контактов светодиода
                             // в режим вывода
   pinMode(LEDB, OUTPUT);
   pinMode(LEDGr, OUTPUT);
   pinMode(Button, INPUT);   // Перевод контактов кнопки в режим ввода
   digitalWrite(LEDR, LOW);  // Включение всех цветов светодиода
   digitalWrite(LEDB, LOW);
   digitalWrite(LEDGr, LOW);
}

void loop() {
   lcd.clear();
   // Очистка дисплея
   lcd.print("Hold Button to");      // Вывод сообщения на ЖК-дисплей
   lcd.setCursor(0, 1);              // Переход на вторую строку
   lcd.print("start.");
```

```
while (digitalRead(Button) == LOW) {     // Тест не начнется, пока не
                                         // будет нажата (и удержана)
                                         // кнопка
   tone(13, 1200, 30);
   delay(1400);
   noTone(13);
}
lcd.clear();
digitalWrite(LEDR, HIGH);          // Прекращение исходного свечения
digitalWrite(LEDB, HIGH);
digitalWrite(LEDGr, HIGH);
randomSeed(analogRead(0));         // Случайный шум с контакта 0
COLOR = random(1, 4);              // Генерация случайного цвета
PSE = random(500, 1200);           // Установка паузы случайной
                                   // длительности между свечениями.
                                   // Повторять этот цикл, пока цвет
                                   // зеленый или синий, а кнопка
                                   // нажата
while (COLOR != 1 && digitalRead(Button) == HIGH) {
   digitalWrite(LEDGr, HIGH);
   digitalWrite(LEDB, HIGH);
   delay(PSE);
   randomSeed(analogRead(0));
   Beep = random(1, 4);            // Случайный сигнал пьезоизлучателя
                                   // (звучит 1 - 3 раз)
   PSE = random(750, 1200);        // Пауза случайной длительности
                                   // между свечениями (усиливает
                                   // эффект неожиданности)
   if (Beep == 1) {
      tone(13, 1600, 350);
      delay(750);
      noTone(13);
   }
   if (COLOR == 2) {
      digitalWrite(LEDGr, LOW);
   }
   if (COLOR == 3) {
      digitalWrite(LEDB, LOW);
   }
   delay(PSE);
   randomSeed(analogRead(0));
   COLOR = random(1, 4);           // Выбор случайного цвета
}
// Выполнить цикл, если цвет красный
if (COLOR == 1 && digitalRead(Button) == HIGH) {
   digitalWrite(LEDGr, HIGH);
   digitalWrite(LEDB, HIGH);
   delay(PSE);
   TME = millis(); // Запись времени с момента запуска программы
   digitalWrite(LEDR, LOW);
   while (digitalRead(Button) == HIGH) { // Продолжение, пока
                                         // кнопка не отпущена,
                                         // с записью времени
                                         // реакции
```

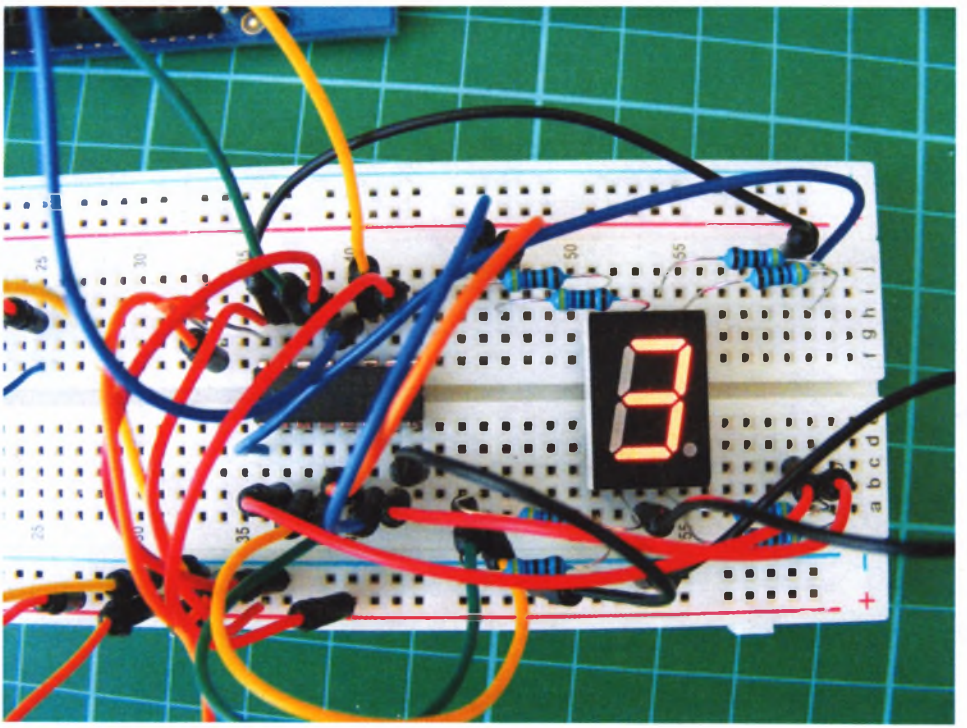
```
            delay(1);
        }
        lcd.display();
        RTME = millis() - TME;          // Время реакции в мс
        lcd.print("Reaction Time:");  // Вывод значения на ЖК-дисплей
        lcd.setCursor(0, 1);
        lcd.print(RTME);
    }
    // Выполнить, если цвет не красный, но кнопка отпущена
    if (COLOR != 1) {
        lcd.print("Released too");
        lcd.setCursor(0, 1);          // Переход на вторую строку
        lcd.print("soon!!!");
        tone(13, 3000, 1500);
        delay(500);
        noTone(13);
    }
    // Тест не перезапустится, пока кнопка не будет однократно нажата
    while (digitalRead(Button) == LOW) {
        delay(10);
    }
    digitalWrite(LEDR, LOW);  // Сброс всех цветов к исходному состоянию
    digitalWrite(LEDB, LOW);
    digitalWrite(LEDGr, LOW);
    lcd.clear();
    lcd.print("Hold Button to");
    lcd.setCursor(0, 1);
    lcd.print("start.");
    int Time = 0;
    delay(1000);
}
```

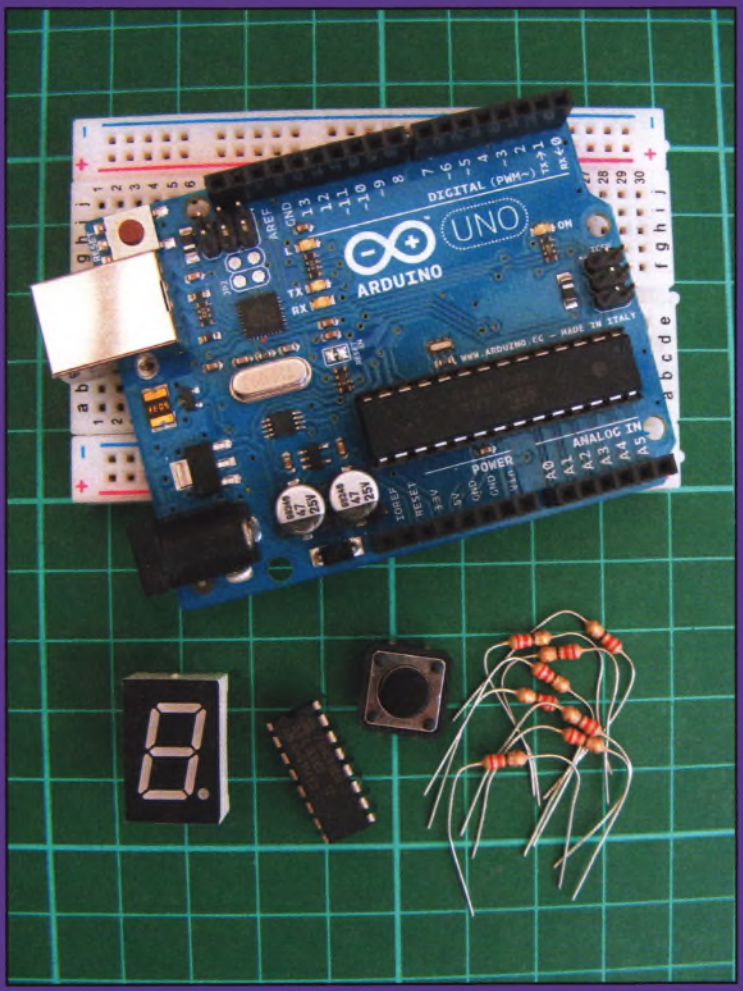
ЧАСТЬ 5

РАБОТА С ЧИСЛАМИ

ПРОЕКТ 16: ЭЛЕКТРОННЫЕ ИГРАЛЬНЫЕ КУБИКИ

НАСТОЛЬНЫЕ ИГРЫ, НЕСОМНЕННО, ДОВОЛЬНО НЕУДОБНЫ ИЗ-ЗА УКАТЫВАНИЯ И ПОТЕРИ КУБИКОВ. ОТЛИЧНОЕ РЕШЕНИЕ — ЭЛЕКТРОННЫЕ ИГРАЛЬНЫЕ КУБИКИ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- 8 резисторов с сопротивлением 220 Ом
- Семисегментный светодиодный индикатор
- Выходной сдвиговый регистр 74HC595
- Тактовая четырехконтактная кнопка

ПРИНЦИП РАБОТЫ

В этом проекте мы создадим электронные игральные кубики с использованием семисегментного светодиодного индикатора. При нажатии кнопки импульс посылается на плату Arduino, а дисплей вспыхивает и отображает случайную цифру в диапазоне от 1 до 6.

В этом проекте используется *выходной сдвиговый регистр* 74HC595 — небольшая интегральная схема с последовательным логическим счетчиком, который позволяет увеличить количество выходов микроконтроллера Arduino путем «сдвига» и хранения данных. Сдвиговый регистр имеет 16 контактов; на одном конце регистра вы увидите точку или полукруг, которая отмечает контакт 1 слева. Контакты пронумерованы против часовой стрелки. На рис. 16.1 показана распиновка, а в таблице 16.1 описывается функция каждого контакта.

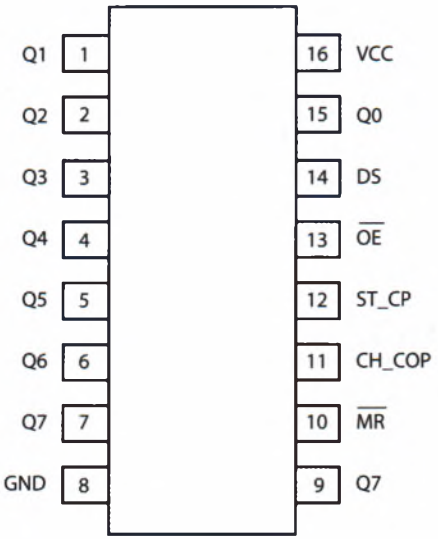


РИСУНОК 16.1

Распиновка сдвигового регистра 74HC595

ТАБЛИЦА 16-1:

Контакты сдвигового регистра 74HC595

КОНТАКТ	ОБОЗНАЧЕНИЕ	ПРЕДНАЗНАЧЕНИЕ
Контакты 1–7, 15	Q0–Q7	Параллельные выходы
Контакт 8	GND	Заземление, VSS
Контакт 9	Q7	Выход для последовательного соединения регистров
Контакт 10	MR	Сброс значений регистра при получении сигнала LOW

Й

КОНТАКТ	ОБОЗНАЧЕНИЕ	ПРЕДНАЗНАЧЕНИЕ
Контакт 11	SH_CP	Вход для тактовых импульсов (контакт CLOCK)
Контакт 12	ST_CP	Синхронизация выходов (контакт LATCH)
Контакт 13	OE	Вход для переключения состояния выходов из высокоомного в рабочее
Контакт 14	DS	Вход для последовательных данных (контакт DATA)
Контакт 16	VCC	Питание

Перемычка от контакта 2 платы Arduino подключается к кнопке, которая при нажатии передает импульс. Чтобы использовать кубики, нажмите кнопку, чтобы сгенерировать и отобразить случайную цифру.

СБОРКА

1. Установите семисегментный светодиодный индикатор на макетную плату, убедившись, что он перекрывает канавку; в противном случае противоположные контакты будут замыкаться. Подключите контакт 3 к шине заземления и резисторы с сопротивлением 220 Ом к остальным контактам, за исключением вывода 8, который не используется. Резисторы необходимы для предотвращения выгорания светодиодов индикатора. См. рис. 16.2 для наглядности.

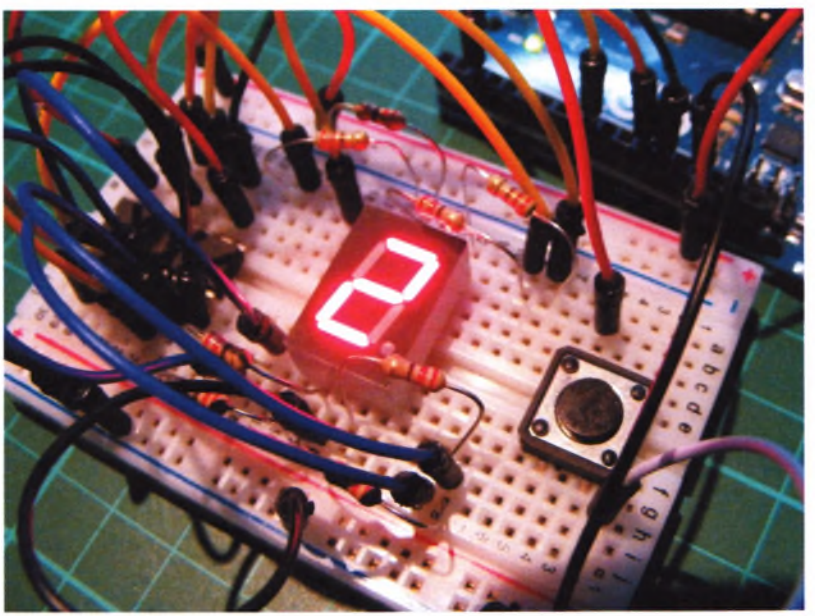


РИСУНОК 16.2

Подключение семисегментного светодиода

2. Установите регистр 74HC595 на макетную плату с маркером с левой стороны. Нижним левым выводом должен быть контакт 1. Регистр должен перекрывать канавку макетной платы, как показано на рис. 16.3.

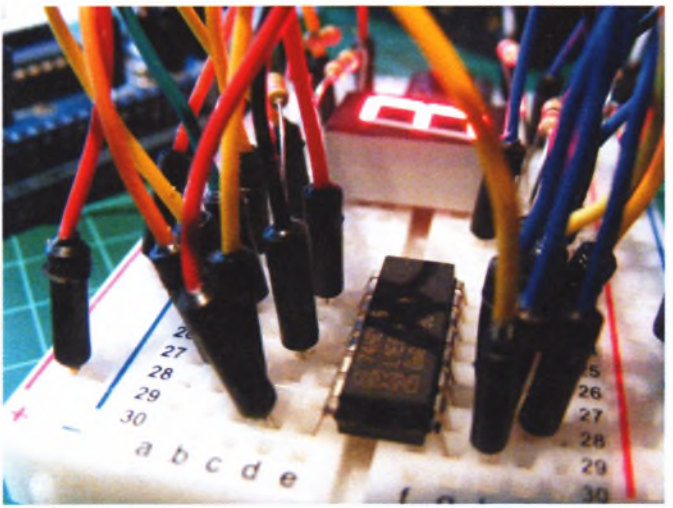


РИСУНОК 16.3

Сдвиговый регистр 74HC595 должен перекрывать канавку макетной платы

3. Осторожно выполните подключения между светодиодным индикатором и сдвиговым регистром согласно таблице ниже.

СЕМИСЕГМЕНТНЫЙ СВЕТОДИОДНЫЙ ИНДИКАТОР	СДВИГОВЫЙ РЕГИСТР	ARDUINO
Контакт 1 (E)*	Контакт 4	
Контакт 2 (D)*	Контакт 3	
Контакт 3		Контакт GND
Контакт 4 (C)*	Контакт 2	
Контакт 5 (DP)*	Контакт 7	
Контакт 6 (B)*	Контакт 1	
Контакт 7 (A)*	Контакт 15	
Контакт 8		Не используется
Контакт 9 (F)*	Контакт 5	
Контакт 10 (G)*	Контакт 6	

* Для этих контактов нужно устанавливать резисторы с сопротивлением 220 Ом между светодиодным индикатором и сдвиговым регистром.

4. Теперь подключите оставшиеся контакты сдвигового регистра (а также кнопки) к плате Arduino, как показано в следующей таблице.

СДВИГОВЫЙ РЕГИСТР	ARDUINO
Контакт 9	Не используется
Контакт 10	Контакт 5V
Контакт 11	Контакт 12
Контакт 12	Контакт 8
Контакт 13	Контакт GND
Контакт 14	Контакт 11
Контакт 16	Контакт 5V
Импульс (от кнопки)	Контакт 2

5. Установите кнопку на макетную плату, перекрыв канавку макетной платы, как показано на рис. 16.4. Подключите ножку с одной стороны кнопки к контакту 2 платы Arduino, а с другой — к контакту GND.

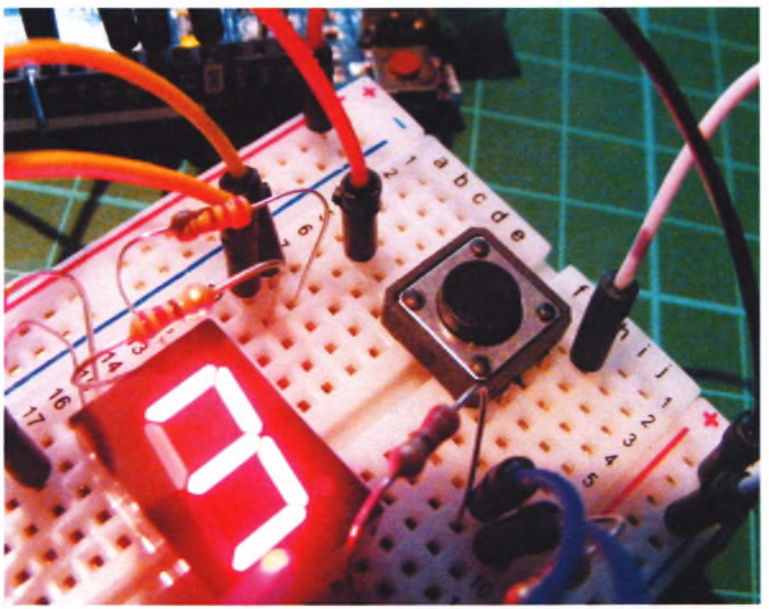


РИСУНОК 16.4

Кнопка также должна находиться по обе стороны от канавки макетной платы

6. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 16.5 и загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

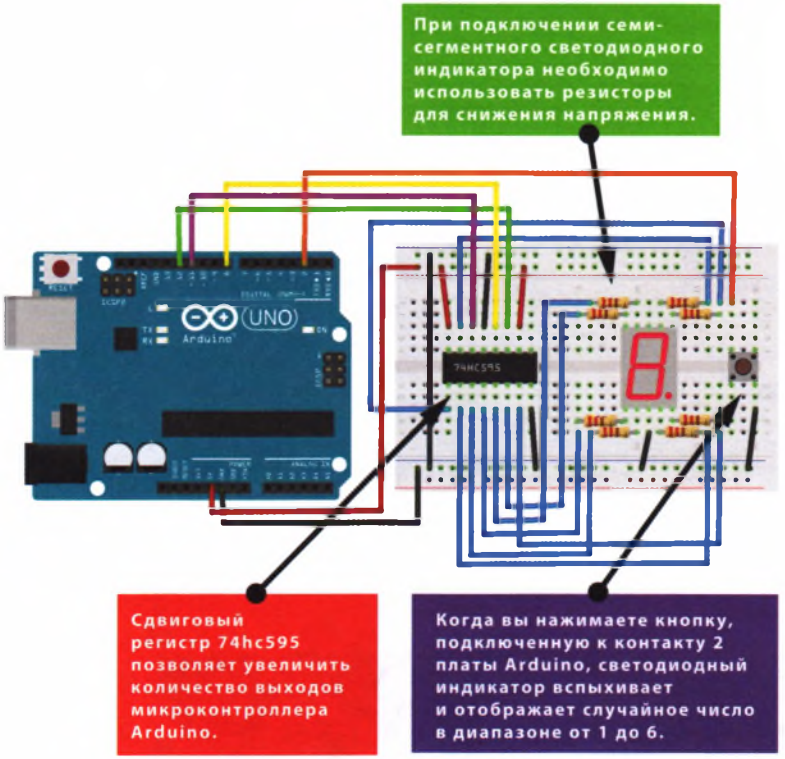


РИСУНОК 16.5

Принципиальная схема цепи электронных костей

СКЕТЧ

В коде скетча сначала определяются контакты для управления сдвиговым регистром 74HC595, обеспечивающим работу семисегментного светодиодного индикатора. Когда на светодиодный индикатор подается питание, на нем загорается точка. При нажатии кнопки светодиодные сегменты быстро мигают, как будто игральные кубики встряхиваются. Через некоторое время выводится случайное число в диапазоне от 1 до 6. Нажмите кнопку еще раз, чтобы сымитировать следующий бросок кубиков.

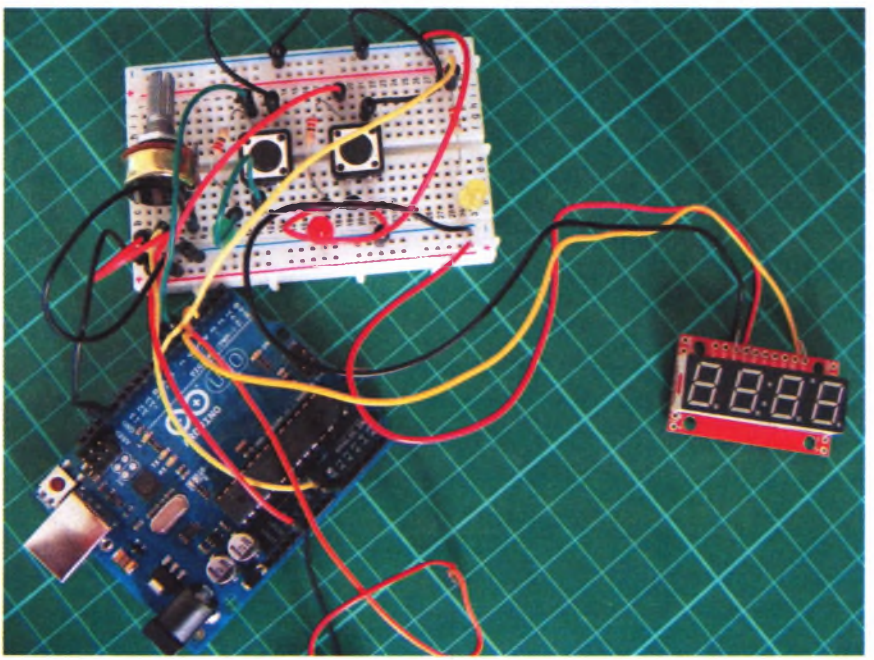
```
// Создано Уорриком А Смитом и используется с его согласия
// http://startingelectronics.com

const int latchPin = 8;      // Контакты, к которым подключен сдвиговый
                             // регистр
const int clockPin = 12;
const int dataPin = 11;
```

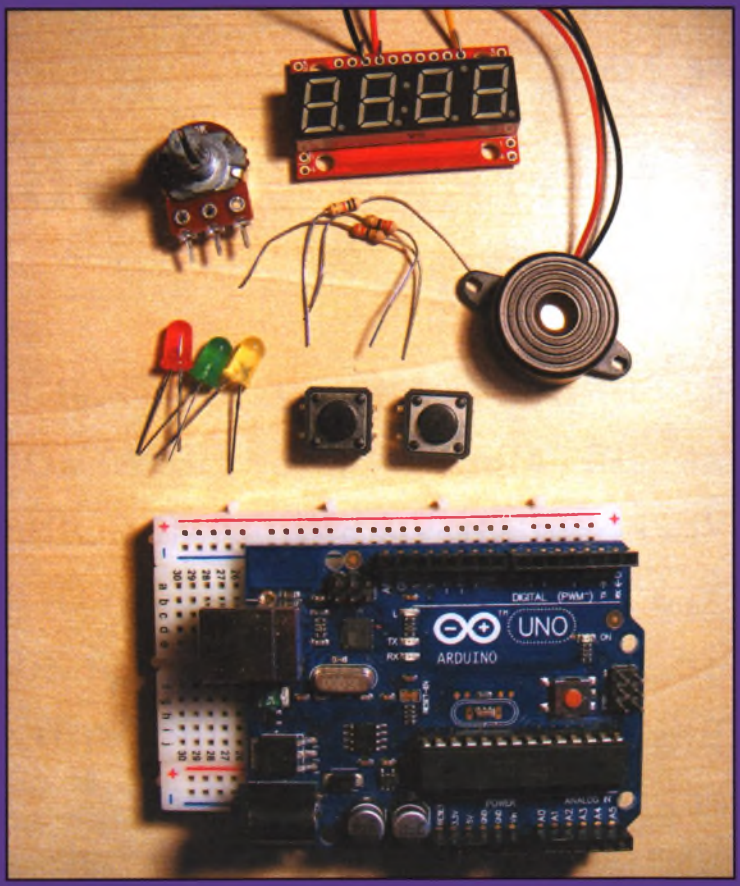
```
const int buttonPin = 2;     // Контакт, к которому подключен
                             // переключатель от 1 до 6 и десятичная
                             // точка семисегментного индикатора
unsigned char lookup_7seg[] = {0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x80};
// Формирование изображения на семисегментном индикаторе
unsigned char shake_dice[] = {0x63, 0x5C};
// Вращение костей на семисегментном индикаторе
unsigned char roll_dice[] = {0x1C, 0x58, 0x54, 0x4C};
// Изменение времени до момента выпадения числа
int rand_seed;
int rand_num = 0;                    // Генерация случайного числа
unsigned char shake_toggle = 0;      // Анимации встряхивания костей
int index = 0;                       // Анимации вращения костей
int shake_speed;                     // Ускорение встряхивания костей

void setup() {
   pinMode(latchPin, OUTPUT);        // Выходные контакты для управления
                                     // сдвиговым регистром
   pinMode(clockPin, OUTPUT);
   pinMode(dataPin, OUTPUT);
   pinMode(buttonPin, INPUT);        // Считывание состояния
                                     // переключателя
   digitalWrite(latchPin, LOW);      // Вывод точки на семисегментном
                                     // индикаторе при запуске
   shiftOut(dataPin, clockPin, MSBFIRST, lookup_7seg[6]);
   digitalWrite(latchPin, HIGH);
   randomSeed(analogRead(0));        // Генерация случайного зерна
}

void loop() {
   if (digitalRead(buttonPin)) {
      shake_speed = 150;     // Сброс скорости встряхивания костей
      delay(30);
      // Генерация случайного значения скорости и вывод анимации
      // встряхивания
      while (digitalRead(buttonPin)) {
         rand_seed++; // Генерация случайного значения
         // Анимация встряхивания костей
         if (shake_toggle) {
            AnimateDice(0, shake_dice);
            shake_toggle = 0;
         }
         else {
            AnimateDice(1, shake_dice);
            shake_toggle = 1;
         }
         delay(80 + shake_speed);   // Увеличение скорости анимации
         if (shake_speed > 0) {
            shake_speed -= 10;
         }
      }
      // Анимация вращения костей
      for (int rolls = 0; rolls < (rand_seed % 10 + 14); rolls++) {
         AnimateDice(index, roll_dice);
```

```
            delay((1 + rolls) * 20);
            index++;
            if (index > 3) {
               index = 0;
            }
         }
         rand_num = random(0, 6);        // Генерация числа, выпавшего
                                         // на кости
         DiceNumber(rand_num);
      }
   }

// Вывод выпавшего числа на семисегментный дисплей
void DiceNumber(unsigned char num) {
   digitalWrite(latchPin, LOW);
   shiftOut(dataPin, clockPin, MSBFIRST, lookup_7seg[num]);
   digitalWrite(latchPin, HIGH);
}

// Вывод одного кадра трясущейся или вращающейся кости
void AnimateDice(int seg, unsigned char *table) {
   digitalWrite(latchPin, LOW);
   shiftOut(dataPin, clockPin, MSBFIRST, table[seg]);
   digitalWrite(latchPin, HIGH);
    }
```

ПРОЕКТ 17: РАКЕТНАЯ ПУСКОВАЯ УСТАНОВКА

В ЭТОМ ПРОЕКТЕ МЫ СОЗДАДИМ ПРОГРАММИРУЕМЫЙ ТАЙМЕР ОБРАТНОГО ОТСЧЕТА, КОТОРЫЙ БУДЕМ ИСПОЛЬЗОВАТЬ ДЛЯ ЗАПУСКА РАКЕТЫ. КОГДА ОТСЧЕТ ДОСТИГНЕТ 0, МЫ ПЕРЕЖЖЕМ ПЛАВКИЙ ПРЕДОХРАНИТЕЛЬ И ЗАПУСТИМ РАКЕТУ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Четырехразрядный семисегментный последовательный индикатор
- Пьезоизлучатель
- 2 тактовые четырехконтактные кнопки
- Потенциометр с сопротивлением 50 кОм
- 3 светодиода (красный, зеленый, желтый)
- 3 резистора с сопротивлением 220 Ом
- Лампочка от елочной гирлянды (опционально)

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- SoftwareSerial

Мы будем использовать четырехразрядный семисегментный последовательный индикатор, который имеет встроенную интегральную схему для управления светодиодами и может быть подключен к Arduino всего тремя проводами. При выборе индикатора убедитесь, что он оборудован входом RX, позволяющим управлять индикатором через один провод.

ПРИНЦИП РАБОТЫ

Вы можете использовать подобный таймер, чтобы создать некое устройство, требующее подачи питания, например сервопривод, светодиодный индикатор или будильник. Вы будете использовать потенциометр для выбора продолжительности обратного отсчета (в диапазоне от 5 до 60 секунд). Светодиодный индикатор при этом отобразит число, чтобы вы могли видеть, до какого значения настраиваете обратный отсчет. Еще будет добавлено две кнопки: *готовности* и *запуска*. После того, как вы выбрали длительность обратного отсчета, нажмите кнопку готовности, чтобы подготовить таймер. Красный светодиод оповещает о готовности к запуску. (Кнопка готовности — это специальный узел безопасности, который предотвращает случайное срабатывание ракетной установки.) После того, как вы водрузили ракету, нажмите кнопку запуска, чтобы начать обратный отсчет. Зеленый светодиод оповещает об успешной подготовке к запуску и начале обратного отсчета.

Когда таймер начинает обратный отсчет, пьезоизлучатель подает звуковой сигнал каждую секунду. Когда остается пять секунд, таймер учащает звуковой сигнал до самого момента запуска. Когда таймер достигает 0, через контакт 7 подается питание на любой доступный выход — в данном случае зажжется желтый светодиод. Вы можете подключить этот таймер к пьезоизлучателю, сервоприводу для открытия двери или даже предохранителю для запуска ракеты. Я покажу вам, как сделать простой воспламенитель позже в этом проекте.

THE BUILD

1. Подключите контакт RX семисегментного последовательного индикатора к контакту 3 платы Arduino, а контакты VCC и GND — к шинам питания и заземления соответственно макетной платы, как показано на рис. 17.1.

СЕМИСЕГМЕНТНЫЙ СВЕТОДИОДНЫЙ ИНДИКАТОР	ARDUINO
Контакт RX	Контакт Pin 3
Контакт VCC	Контакт 5V
Контакт GND	Контакт GND

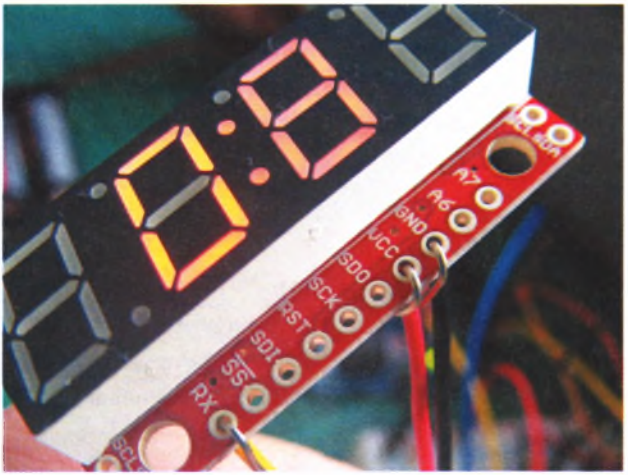


РИСУНОК 17.1

Контакты семисегментного индикатора

2. Установите потенциометр на макетную плату и подключите его левую ножку к шине питания 5 В, центральную — непосредственно к контакту A0 платы Arduino, а правую — к шине заземления, как показано на рис. 17.2.

ПОТЕНЦИОМЕТР	ARDUINO
Левая ножка	Контакт 5V
Центральная ножка	Контакт A0
Правая ножка	Контакт GND

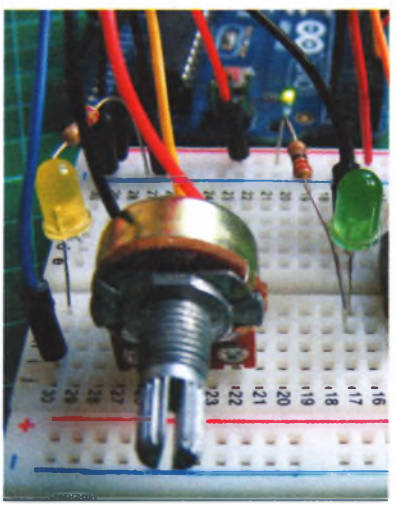


РИСУНОК 17.2

Установка потенциометра на макетную плату

3. Подключите красный провод пьезоизлучателя к контакту 4 платы Arduino, а черный — к контакту GND, как показано на рис. 17.3.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Красный провод	Контакт 4
Черный провод	Контакт GND

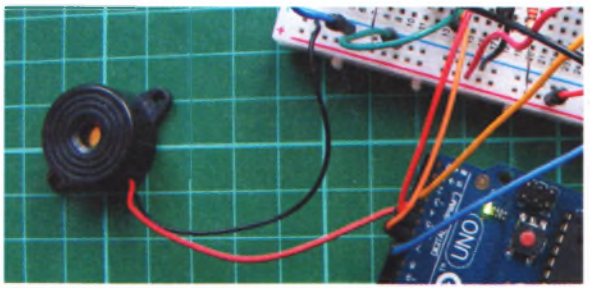


РИСУНОК 17.3

Подключение пьезоизлучателя

4. Установите две кнопки на макетную плату, с контактами A и B с одной стороны от канавки, и контактами D и C с другой, следуя схеме, показанной на рис. 17.4.

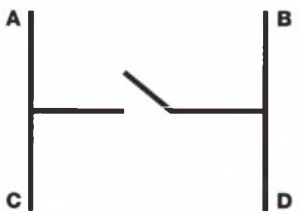


РИСУНОК 17.4

Схема тактовой кнопки

5. Затем подключите кнопки, как показано на рис. 17.5. Чтобы создать кнопку готовности, подключите ножку C первой кнопки к контакту GND, а ножку D — к контакту 5 платы Arduino. Чтобы создать кнопку запуска, подключите ножку C второй кнопки к контакту GND, а ножку D — к контакту 6 платы Arduino.

КНОПКИ	ARDUINO
Кнопка готовности, ножка C	Контакт GND
Кнопка готовности, ножка D	Контакт 5
Кнопка запуска, ножка C	Контакт GND
Кнопка запуска, ножка D	Контакт 6

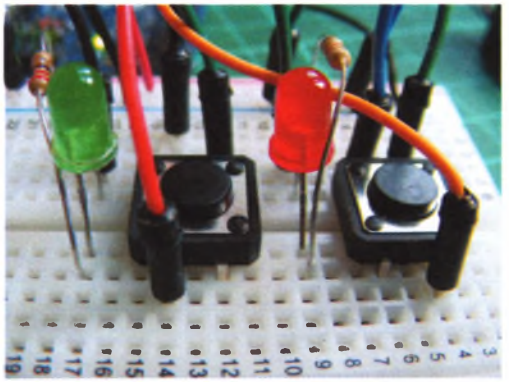


РИСУНОК 17.5

Подключение кнопок и светодиодов

6. Установите красный светодиод на макетную плату так, чтобы короткая ножка (катод) была подключена к контакту B кнопки готовности. Подключите другую ножку (анод) к резистору с сопротивлением 220 Ом. Вторую ножку резистора подключите к шине питания 5 В. Затем установите зеленый светодиод так, чтобы короткая ножка (катод) была подключена к контакту B кнопки запуска. Подключите другую ножку (анод) к резистору с сопротивлением 220 Ом. Вторую ножку резистора подключите к шине питания 5 В.

КРАСНЫЙ И ЗЕЛЕНЫЙ СВЕТОДИОДЫ	ARDUINO
Короткие ножки (катоды)	Контакт GND
Длинные ножки (аноды)	Контакт 5V через резистор с сопротивлением 220 Ом

7. Подключите воспламенитель. В примере мы используем желтый светодиод в качестве индикатора воспламенителя. Установите его на макетную плату так, чтобы короткая ножка (катод) была подключена к контакту GND платы Arduino, а длинная (анод) — через резистор с сопротивлением 220 Ом к контакту 7 платы Arduino. (См. раздел «Создание рабочего предохранителя» далее в этом проекте, чтобы узнать, как сделать свой собственный плавкий предохранитель.)

ЖЕЛТЫЙ СВЕТОДИОД (ВОСПЛАМЕНИТЕЛЬ)	ARDUINO
Короткая ножка (катод)	Контакт GND
Длинная ножка (анод)	Контакт 7 через резистор с сопротивлением 220 Ом

Когда обратный отсчет достигнет 0, контакт 7 переводится в режим HIGH и запускает воспламенитель. Вместо того, чтобы по-настоящему пережечь предохранитель, мы зажигаем желтый светодиод, имитирующий воспламенение.

8. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 17.6, и загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

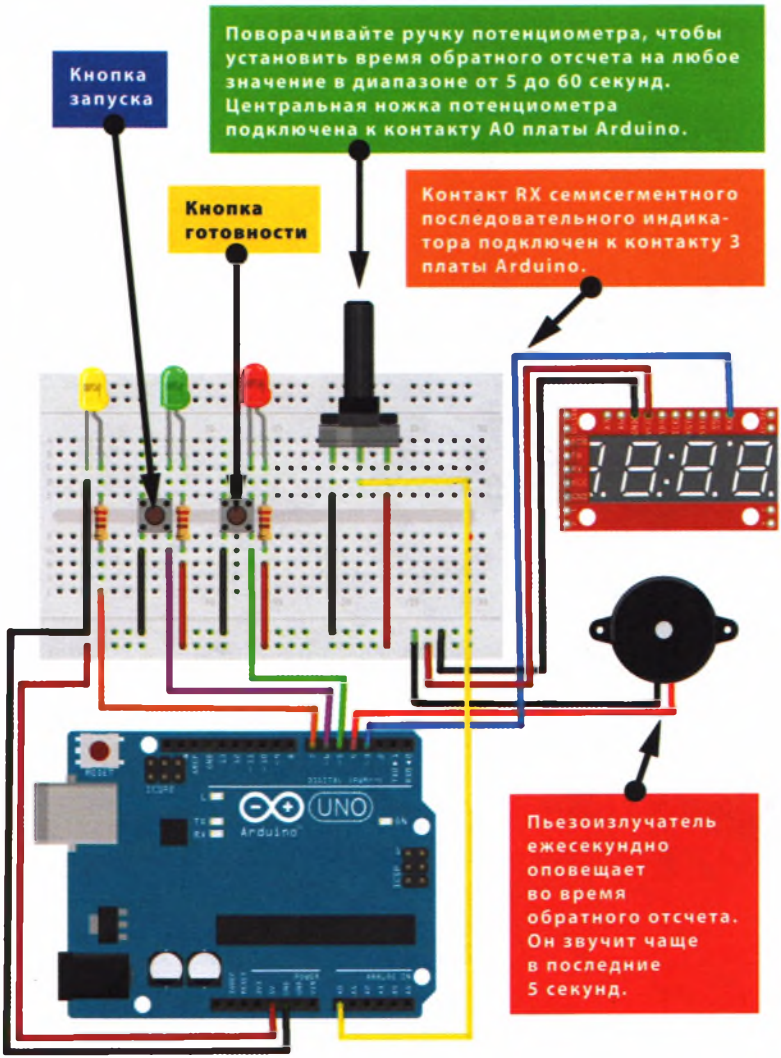


РИСУНОК 17.6

Схема ракетной пусковой установки

ВНИМАНИЕ!

В вашей стране могут быть введены ограничения на запуск любительских ракет или фейерверков, поэтому, пожалуйста, предварительно ознакомьтесь с законодательством. Вы несете ответственность в рамках закона.

СОЗДАНИЕ РАБОЧЕГО ПРЕДОХРАНИТЕЛЯ

Вместо применения светодиода для имитации воспламенения вы можете создать рабочий предохранитель, используя лампочку от старой елочной гирлянды. При создании плавкого предохранителя обязательно соблюдайте меры предосторожности! Эти инструкции предназначены для развлекательных целей и должны выполняться только взрослыми.

1. С помощью мини-дрели или дремеля надпилите верхнюю часть стеклянной колбы лампочки от елочной гирлянды, чтобы обрезать ее (см. рис. 17.7).

РИСУНОК 17.7

Обрезка колбы лампочки с помощью мини-дрели

2. Вырежьте часть наконечника стеклянного корпуса, и верхняя часть должна легко отделиться (рис. 17.8).

РИСУНОК 17.8

Удаление наконечника

3. Теперь отрежьте головку деревянной спички (аккуратно, чтобы не воспламенить ее!) и аккуратно установите головку спички в корпус лампочки, стараясь не повредить нить накаливания (рис. 17.9).

РИСУНОК 17.9

Установка головки спички в колбу лампочки

4. Наконец, соедините провода лампочки с проводами воспламенителя. Когда питание подается на лампочку, нить накаливается и поджигает головку спички (рис. 17.10), создавая достаточную энергию для воспламенения плавкого предохранителя.

РИСУНОК 17.10

Сгоревший предохранитель

СКЕТЧ

Сначала в коде скетча определяется каждый компонент и его подключение к Arduino. Библиотека SoftwareSerial управляет четырехразрядным семисегментным последовательным светодиодным индикатором, а потенциометр изменяет время отсчета в диапазоне от 5 до 60 секунд.

Кнопка готовности работает как цифровой переключатель и функция безопасности, допускающая нажатие кнопки запуска. Если во время обратного отсчета нажать кнопку готовности, обратный отсчет прекращается и значение на индикаторе сбрасывается.

Инструкции `tone` в скетче заставляют пьезоизлучатель звучать во время обратного отсчета. Когда обратный отсчет достигает 0, контакт воспламенителя (в данном случае, подключенный к светодиоду) переводится в режим `HIGH` и включает светодиод.

```
// Ardunaut Arduining.com, используется с согласия разработчика

#define FuseTIME 1500          // Длительность подачи тока через
                               // предохранитель в мс
#include <SoftwareSerial.h>    // Вызов библиотеки SoftwareSerial

#define Fuse    7       // Контакт, к которому подключен предохранитель
                        // (светодиод или воспламенитель)
#define GoButt  6       // Контакт, к которому подключена кнопка запуска
#define ArmButt 5       // Контакт, к которому подключена кнопка готовности
#define BuzzPin 4       // Контакт, к которому подключен пьезоизлучатель
#define TXdata  3       // Контакт, к которому подключен ножка RX
                        // индикатора
#define RXdata  2       // Не используется
#define SetPot  0       // Аналоговый контакт, к которому подключен
                        // потенциометр

SoftwareSerial mySerialPort(RXdata, TXdata);

void setup() {
   pinMode(TXdata, OUTPUT);
   pinMode(RXdata, INPUT);
   pinMode(Fuse, OUTPUT);
   pinMode(ArmButt, INPUT);          // Перевод контакта кнопки
                                     // готовности в режим ввода
   pinMode(GoButt, INPUT);           // Перевод контакта кнопки запуска
                                     // в режим ввода
   digitalWrite(Fuse, LOW);          // Открытие цепи воспламенителя
   digitalWrite(ArmButt, HIGH);      // Включение резистора
   digitalWrite(GoButt, HIGH);       // Включение резистора
   mySerialPort.begin(9600);
   delay(10);                        // Ожидание запуска индикатора
   mySerialPort.print("v");          // Сброс индикатора
   mySerialPort.print("z");          // Яркость
```

```
   mySerialPort.write(0x40);          // 3/4 интенсивности
   mySerialPort.print("w");           // Управление десятичной точкой
   mySerialPort.write(0x10);          // Включение двоеточия на индикаторе
}

int DownCntr;                         // Обратный отсчет (1/10 секунд)
int Go = 0;                           // Остановка

void loop() {
   if (!digitalRead(GoButt) || !digitalRead(ArmButt)) {
      Go = 0;                         // Отмена обратного отсчета
      tone(BuzzPin, 440, 1500);
      delay(1500);
   }

   if (Go == 0) {
      WaitARM();
      WaitGO();
   }
   ShowTimer();
   if (DownCntr > 50) {
      if (DownCntr % 10 == 0)tone(BuzzPin, 1000, 50);      // Один сигнал
                                                           // в секунду
   }
   else if (DownCntr % 2 == 0)tone(BuzzPin, 1000, 50);     // Учащение
                                                          // сигнализирования

   if (DownCntr == 0) {
      tone(BuzzPin, 440, FuseTIME);                    // Звук запуска
      digitalWrite(Fuse, HIGH);       // Закрытие цепи предохранителя
      delay(FuseTIME);
      digitalWrite(Fuse, LOW);        // Открытие цепи предохранителя
      Go = 0;
   }
   while (millis() % 100);            // Ожидание 50 мс
   delay(50);
   DownCntr--;
}

void WaitGO() {
   ShowTimer();
   while (digitalRead(GoButt));
   Go = 1;
   delay(20);
   while (!digitalRead(GoButt));      // Устранение ложных нажатий кнопки
                                      // запуска
}

void ReadTimer() {
   DownCntr = map(analogRead(SetPot), 0, 1023, 5, 60);
   DownCntr *= 10;
}

void ShowTimer() {
```

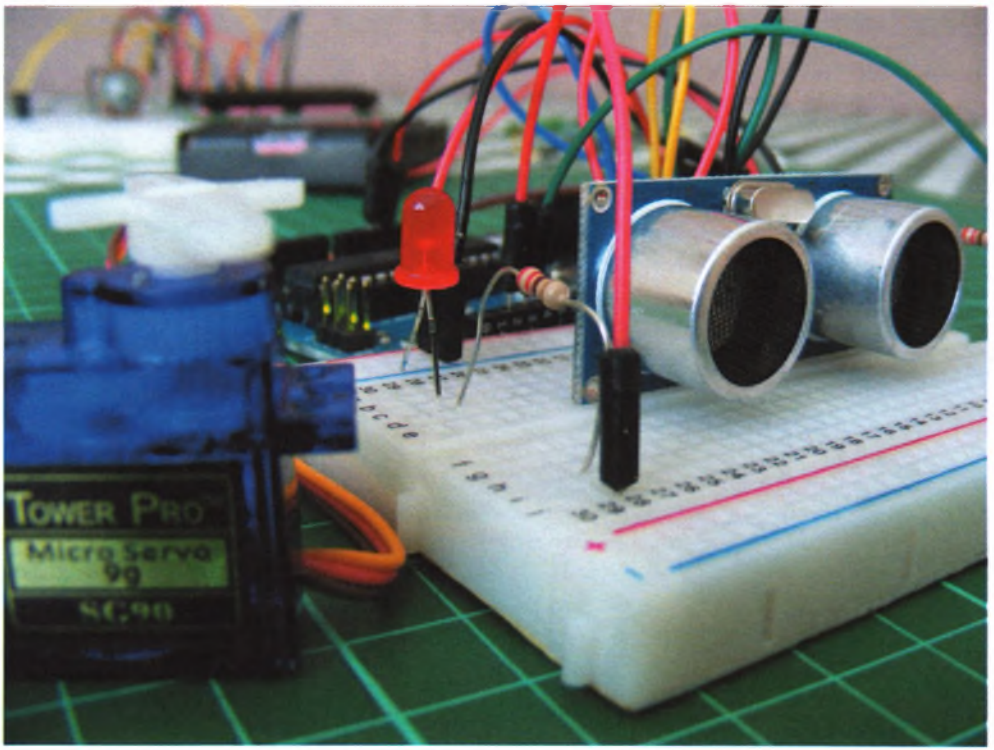
```
    String seconds = String (DownCntr, DEC);
    while (seconds.length() < 3)seconds = "0" + seconds;    // Формати-
                                                            // рование в 3-значное число

    mySerialPort.print(seconds);                            // Вывод на последовательный
                                                            // индикатор
    mySerialPort.print(" ");                                // Отключение последней цифры
}

void WaitARM() {
    while (digitalRead(ArmButt) == 1) {
        ReadTimer();
        delay(50);
        ReadTimer();
        ShowTimer();
        delay(150);
    }

    Go = 0;
    ShowTimer();
    tone(BuzzPin, 2000, 150);
    delay(200);
    tone(BuzzPin, 2000, 150);
    delay(200);
    tone(BuzzPin, 2000, 150);

    delay(20);
    while (!digitalRead(ArmButt));    // Устранение ложных нажатий кнопки
                                      // готовности

}
```

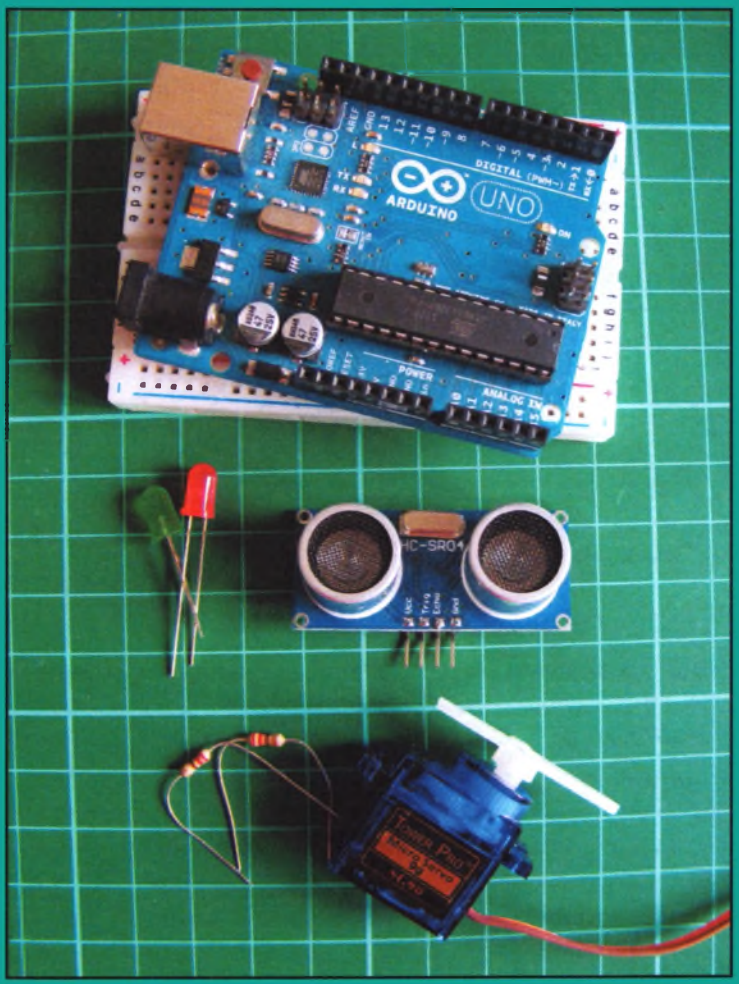
ЧАСТЬ 6

БЕЗОПАСНОСТЬ

ПРОЕКТ 18: ДАТЧИК ВТОРЖЕНИЯ

В ЭТОМ ПРОЕКТЕ МЫ БУДЕМ ИСПОЛЬЗОВАТЬ УЛЬТРАЗВУКОВОЙ ДАЛЬНОМЕР ДЛЯ ОБНАРУЖЕНИЯ ВТОРЖЕНИЙ НА НАШУ ТЕРРИТОРИЮ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Четырехконтактный ультразвуковой дальномер HC-SR04
- Сервопривод
- Красный светодиод
- Зеленый светодиод
- 2 резистора с сопротивлением 220 Ом

Мы подключим дальномер к сервоприводу и нескольким светодиодам, поэтому, когда кто-нибудь окажется на определенном расстоянии от устройства, зеленый светодиод выключится, загорится красный светодиод, и заработает сервопривод (см. рис. 18.1).

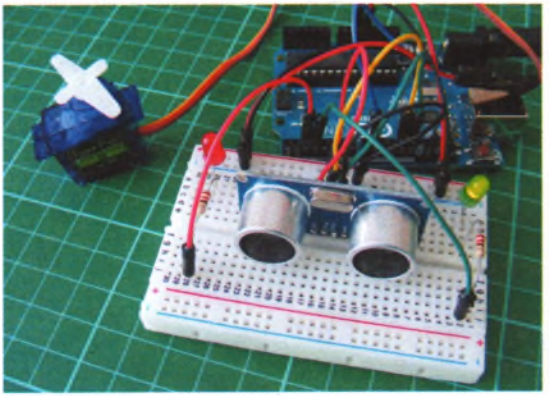


РИСУНОК 18.1

Светодиоды предупреждают о вторжении

ПРИНЦИП РАБОТЫ

Этот проект универсален и может быть использован и адаптирован к различным условиям. Поскольку ультразвуковой дальномер учитывает расстояние до объекта, вы можете использовать его, к примеру, чтобы охранять определенную территорию и запускать сигнал тревоги, когда этот периметр нарушен. Дальномер аналогичен радару: он посылает ультразвуковой сигнал, или *пинг*. Когда он достигает объекта, сигнал отражается обратно, словно эхо, и время от момента отправки сигнала и до его получения обратно используется для вычисления расстояния. Плата Arduino может использовать такие значения для выполнения каких-либо действий.

В этом проекте, когда датчик обнаруживает злоумышленника в пределах определенной зоны доступа, загорается красный светодиод, и сервопривод начинает двигаться. Вы можете адаптировать этот проект для выполнения другого события при обнаружении злоумышленника, например нажатия кнопки системы безопасности или блокировки двери. В игровом проекте вы можете определить небольшое расстояние, чтобы, когда вы машете рукой перед дальномером, сервопривод нажимал кнопку и выдавал приз, например конфету.

СБОРКА

1. Установите ультразвуковой дальномер на макетную плату. Дальномер, который мы используем в этом проекте, имеет четыре ножки, как показано на рис. 18.2. Подключите ножку GND датчика к шине заземления, ножку VCC — к шине питания, а ножки Trig и Echo — к контактам 12 и 13 платы Arduino соответственно.

УЛЬТРАЗВУКОВОЙ ДАЛЬНОМЕР	ARDUINO
Ножка GND	Контакт GND
Ножка VCC	Контакт 5V
Ножка Trig	Контакт 12
Ножка Echo	Контакт 13

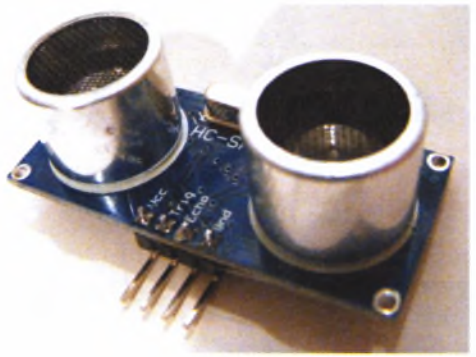


РИСУНОК 18.2

Ультразвуковой дальномер HC-SR04

2. Подключите коричневый провод (земля) сервопривода к шине заземления, красный (питание) — к шине питания 5 В и желтый провод (сигнал управления) — к контакту 9 платы Arduino.

СЕРВОПРИВОД	ARDUINO
Красный провод	Контакт 5V
Коричневый провод	Контакт GND
Желтый провод	Контакт 9

3. Установите красный и зеленый светодиоды на макетную плату, подключив короткие ножки (катоды) к шине заземления. Подключите длинную ножку (анод) каждого светодиода через резистор с сопротивлением 220 Ом к плате Arduino: красный светодиод к контакту 2, а зеленый — к контакту 3.

СВЕТОДИОДЫ	ARDUINO
Короткие ножки (катоды)	Контакт GND
Длинная ножка (анод) красного светодиода	Контакт 2 через резистор с сопротивлением 220 Ом
Длинная ножка (анод) зеленого светодиода	Контакт 3 через резистор с сопротивлением 220 Ом

4. Подключите шины макетной платы к контактам 5V и GND платы Arduino. Собранная цепь показана на рис. 18.3.

Й

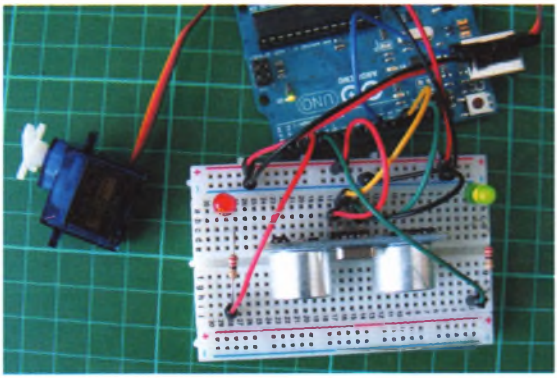


РИСУНОК 18.3

Законченный проект датчика вторжения

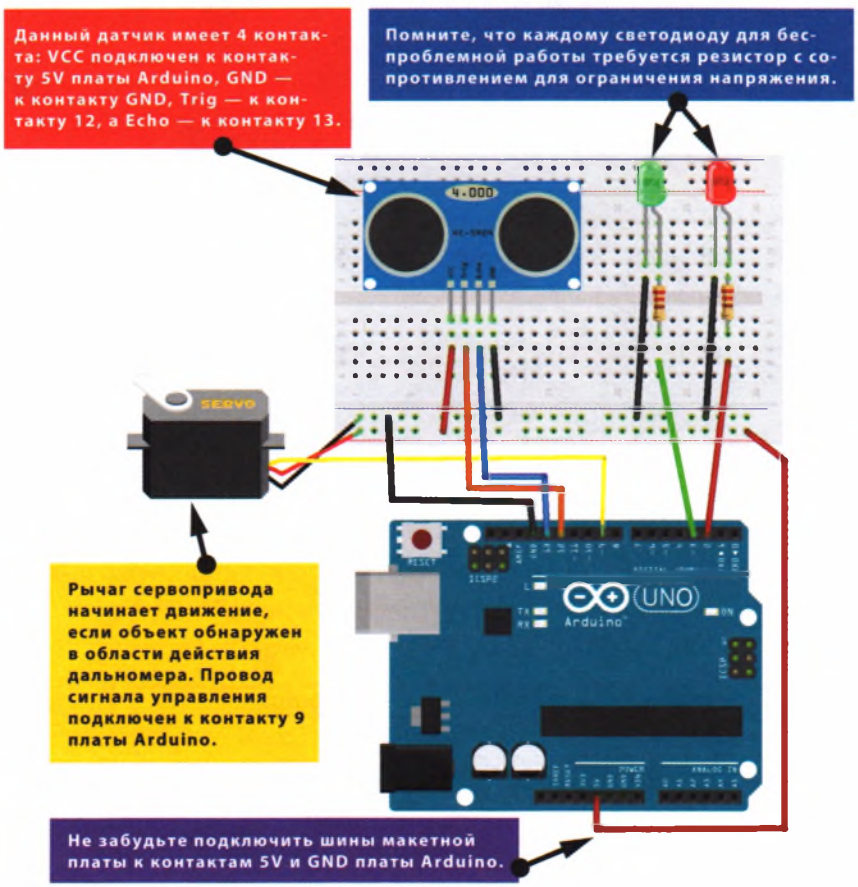


РИСУНОК 18.4

Принципиальная схема цепи датчика вторжения

5. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 18.4, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

СКЕТЧ

Когда объект попадает в область действия дальномера, загорается красный светодиод, а сервопривод поворачивается на 45 градусов. Вы можете изменить область действия дальномера в следующей строке кода скетча:

```
if (distance <= 15)
```

В этом примере, если объект находится на расстоянии не более 15 см от дальномера, будет выполнен последующий блок кода.

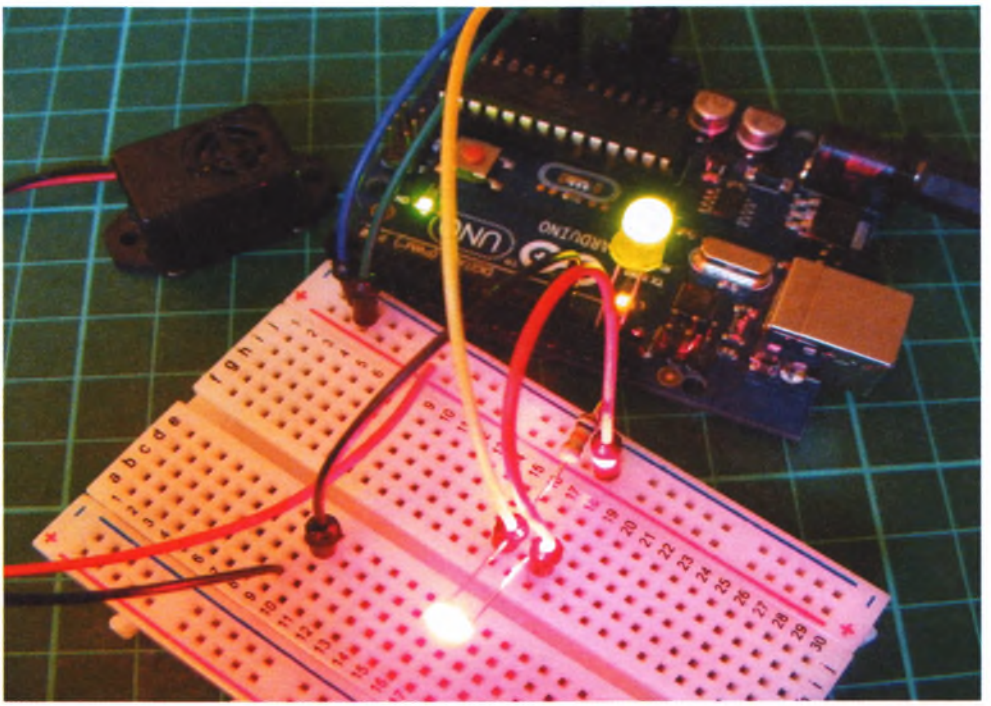
Ножка Trig дальномера подключена к контакту 12 платы Arduino и служит для передачи ультразвукового сигнала (пинга). Когда сигнал достигает объекта, он возвращается к дальномеру и передается на контакт 13 платы Arduino. Разница во времени между двумя сигналами позволяет определить расстояние от дальномера до объекта. Если расстояние выше установленного минимума, зеленый светодиод остается включенным; в противном случае загорается красный светодиод, и сервопривод начинает работать.

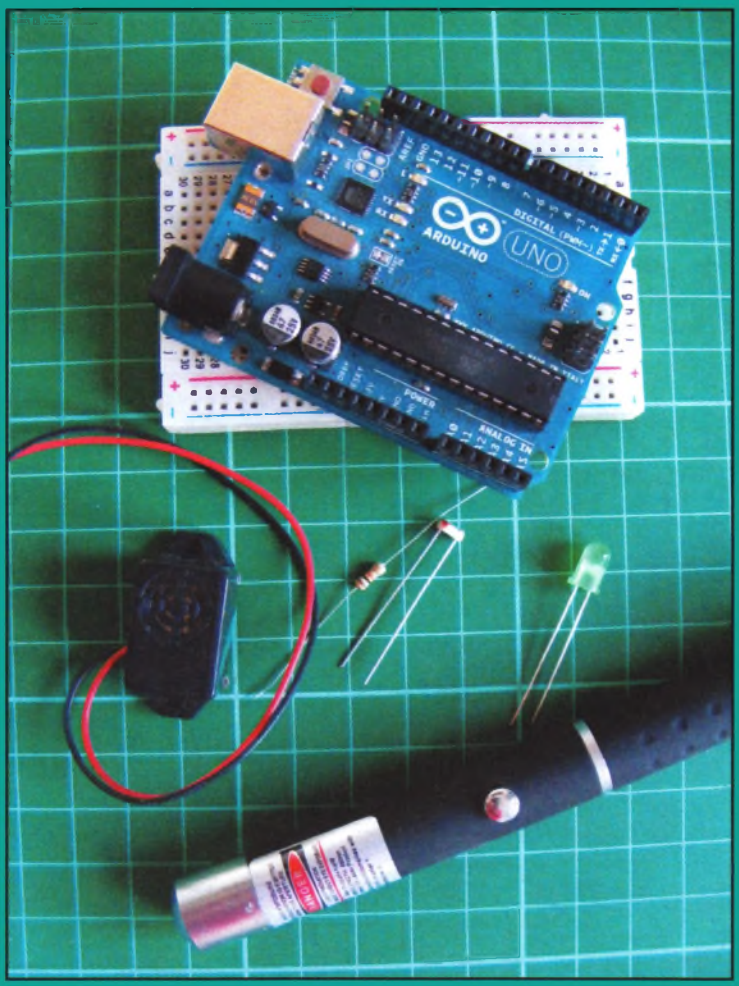
```
/* Библиотека NewPing создана Мимом Экелем, teckel@leethost.com.
   Авторские права 2012, Лицензия: GNU GPL v3
   http://www.gnu.org/licenses/gpl-3.0.html
*/

#include <NewPing.h>          // Вызов библиотеки NewPing
#include <Servo.h>            // Вызов библиотеки Servo
#define trigPin 12            // Контакт Trig подключен к контакту 12
                              // Arduino
#define echoPin 13            // Контакт Echo подключен к контакту 13
                              // Arduino
#define MAX_DISTANCE 500
NewPing sonar(trigPin, echoPin, MAX_DISTANCE);      // Установка
                                                    // библиотеки
int greenLed = 3, redLed = 2;         // Назначение зеленого светодиода
                                      // контакту 3, а красного
                                      // -контакту 2

int pos = 20;
Servo myservo;

void setup() {
   Serial.begin (115200);
   pinMode(trigPin, OUTPUT);
   pinMode(echoPin, INPUT);
   pinMode(greenLed, OUTPUT);
```

```
   pinMode(redLed, OUTPUT);
   myservo.attach(9);          // Сервопривод подключен к контакту 9
}

void loop() {
   int duration, distance, pos = 0, i;
   digitalWrite(trigPin, LOW);
   delayMicroseconds(2);
   digitalWrite(trigPin, HIGH);              // Контакт Trig посылает пинг
   delayMicroseconds(10);
   digitalWrite(trigPin, LOW);
   duration = pulseIn(echoPin, HIGH);        // На контакт Echo получаем
                                             // ответ
   distance = (duration / 2) / 29.1;
   Serial.print(distance);
   Serial.println(" cm");
   // Если дальномер обнаруживает объект в пределах 15 см
   if (distance <= 15) {
      digitalWrite(greenLed, LOW);    // Выключение зеленого светодиода
      digitalWrite(redLed, HIGH);     // Включение красного светодиода
      myservo.write(180);             // Поворот сервопривода
                                      // на 180 градусов
      delay(450);
      digitalWrite(redLed, LOW);      // Выключение красного светодиода
      myservo.write(90);
      delay(450);
      digitalWrite(redLed, HIGH);
      myservo.write(0);
      delay(450);
      digitalWrite(redLed, LOW);
      myservo.write(90);
   }
   // Иначе
   else {
      digitalWrite(redLed, LOW);      // Выключение красного светодиода
      digitalWrite(greenLed, HIGH);   // Включение зеленого светодиода
      myservo.write(90);
   }
   delay(450);
    }
```

ПРОЕКТ 19: ЛАЗЕРНАЯ СИГНАЛИЗАЦИЯ

В ЭТОМ ПРОЕКТЕ ВЫ СОЗДАДИТЕ ПРОСТУЮ ЛАЗЕРНУЮ СИГНАЛИЗАЦИЮ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Фоторезистор
- Пьезоизлучатель
- Резистор с сопротивлением 10 кОм
- Лазерная указка

Вероятно, вы видели фильм, в котором ценный предмет защищен сеткой лазерных лучей. Лучи выглядят круто и кажутся очень высокотехнологичными, но принципы, лежащие в основе их работы, на самом деле очень просты.

ПРИНЦИП РАБОТЫ

Когда лазерная указка светит на фоторезистор, загорается зеленый светодиод, обозначая работоспособность цепи. Если загородить луч лазера, светодиод выключается, и пьезоизлучатель издает звук.

Как вы узнали из проектов 13 и 18, фоторезисторы обладают переменным сопротивлением, зависящим от силы падающего на них света. Когда фоторезистор не фиксирует свечение лазера, сопротивление снижается, а Arduino передает на контакты пьезоизлучателя напряжение.

Лазерные лучи, видимые при дневном свете или даже в темноте, очень мощные и могут быть чрезвычайно опасными. В этом проекте мы используем довольно безопасную маломощную лазерную ручку (см. рис. 19.1).

РИСУНОК 19.1

Лазерные ручки также могут представлять опасность. Никогда не направляйте их в глаз!

СБОРКА

1. Установите фоторезистор на макетную плату. Подключите одну ножку к шине питания 5 В с помощью перемычки. Подключите другую ножку через резистор с сопротивлением 10 кОм к контакту A0 платы Arduino и шине заземления макетной платы.

ФОТОРЕЗИСТОР	ARDUINO
Ножка 1	Контакт 5V
Ножка 2	Контакты A0 и GND через резистор с сопротивлением 10 кОм

2. Подключите красный провод пьезоизлучателя непосредственно к контакту 11 платы Arduino и черный (GND) к шине заземления макетной платы.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Черный провод	Контакт GND
Красный провод	Контакт 11

3. Установите длинную ножку зеленого светодиода в контакт 13 платы Arduino, а короткую — в контакт GND.

СВЕТОДИОД	ARDUINO
Короткая ножка (катод)	Контакт GND
Длинная ножка (анод)	Контакт 13

4. Подключите шины макетной платы к контактам 5V и GND платы Arduino.
5. Перед загрузкой кода вам необходимо вычислить значение фоторезистора при окружающем освещении. Запустите следующую небольшую программу и настройте фоторезистор в соответствии с инструкциями.

```
void setup() {
  pinMode(4, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  digitalWrite(4, HIGH);
  Serial.println(analogRead(0));
}
```

6. Откройте окно монитора порта в среде разработки Arduino. В нем отобразится значение, которое считывается из фоторезистора, — на рис. 19.2 оно равно 964 (при естественном освещении). Запишите число, полученное в вашем случае, — оно будет разным при различной освещенности.

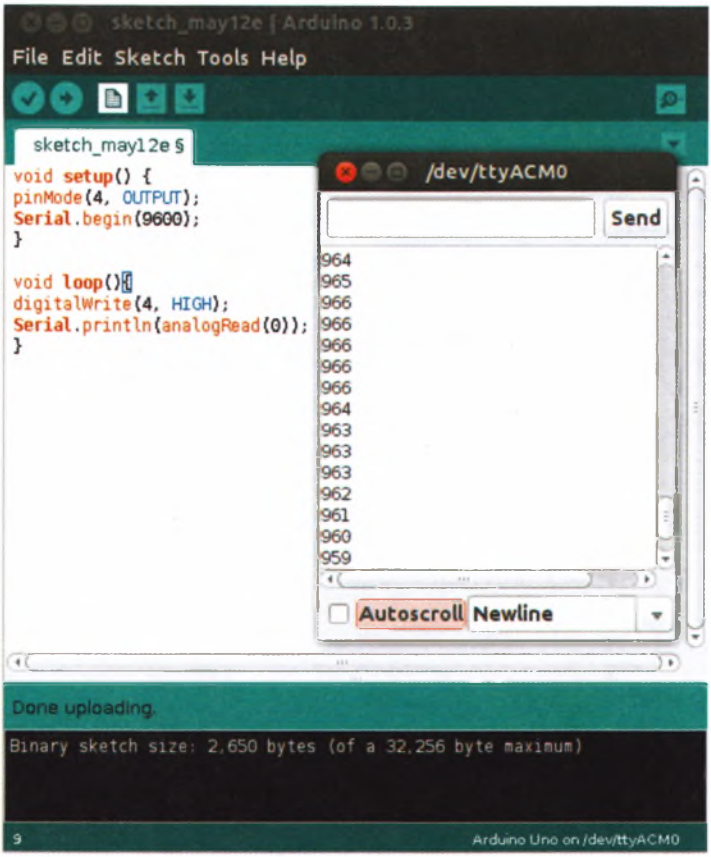


РИСУНОК 19.2

Считывание значений с фоторезистора

Теперь посветите лазером на фоторезистор и также запишите полученное значение; у меня вышло 620. Полученный результат может показаться нелогичным, так как вы ожидаете, что большее количество света даст большее значение, но на самом деле это число соответствует сопротивлению фоторезистора: больше света = меньше сопротивление. Ваши значения будут отличаться от приведенных в книге, поэтому обязательно запишите оба показания.

7. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 19.3, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

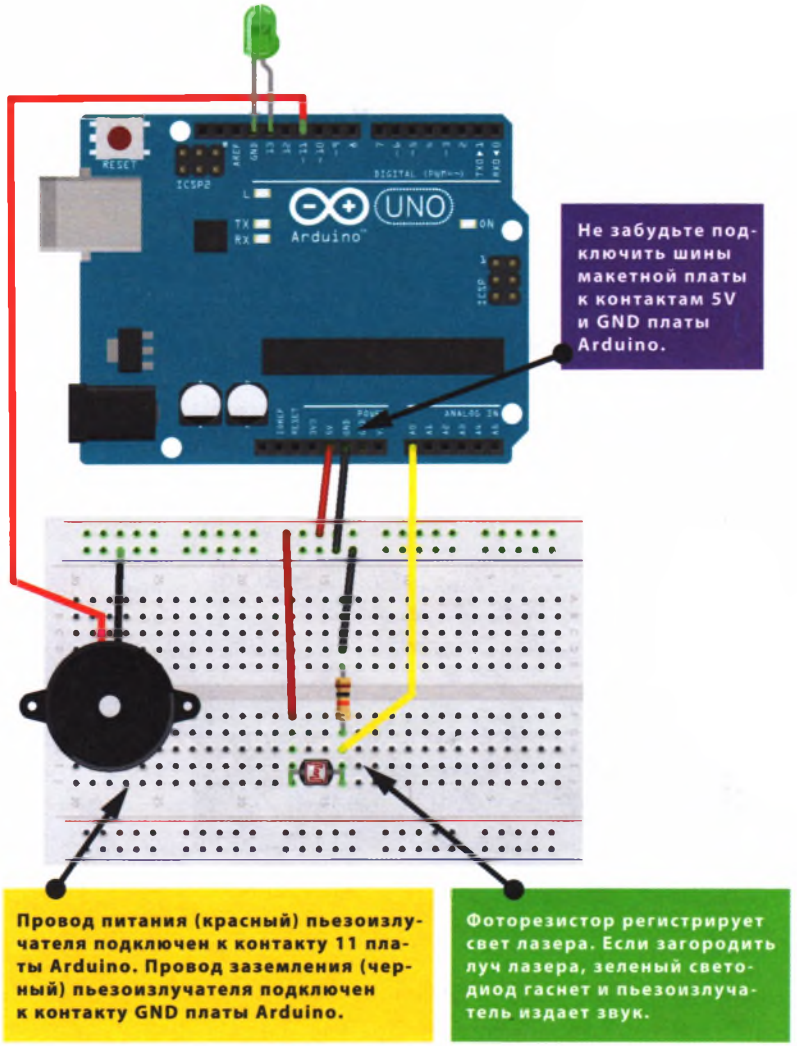


РИСУНОК 19.3

Принципиальная схема цепи лазерной сигнализации

СКЕТЧ

Сначала в коде скетча контакты 11 и 13 платы Arduino переводятся в режим вывода: первый для пьезоизлучателя, а второй — для светодиода. Фоторезистор подключен к контакту A0 платы Arduino. Если аналоговое значение, полученное на контакт A0, превышает 850 (т.е. яркость света снизилась — лазерный луч был перекрыт), контакт пьезоизлучателя переводится в режим HIGH (пьезоизлучатель издает звук), а светодиод отключается.

Не забудьте изменить значение сопротивления в зависимости от вашего результата в следующей строке:

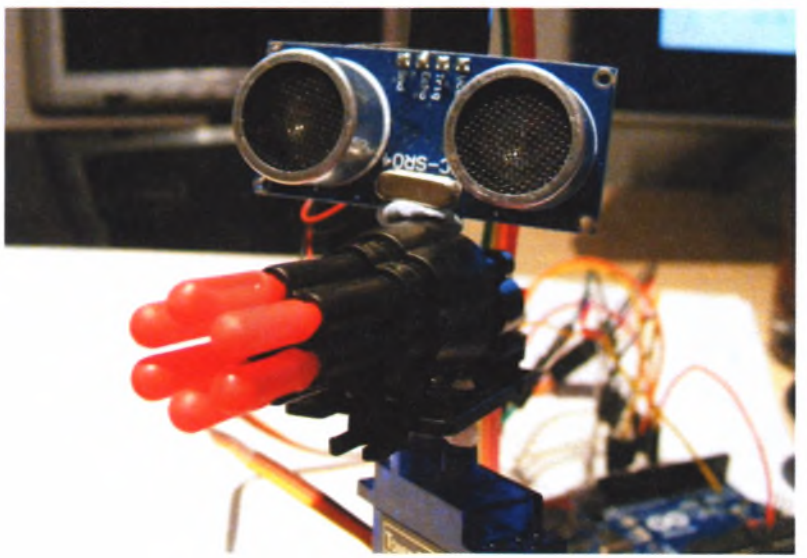
```
if (analogRead(0) > 850) {
```

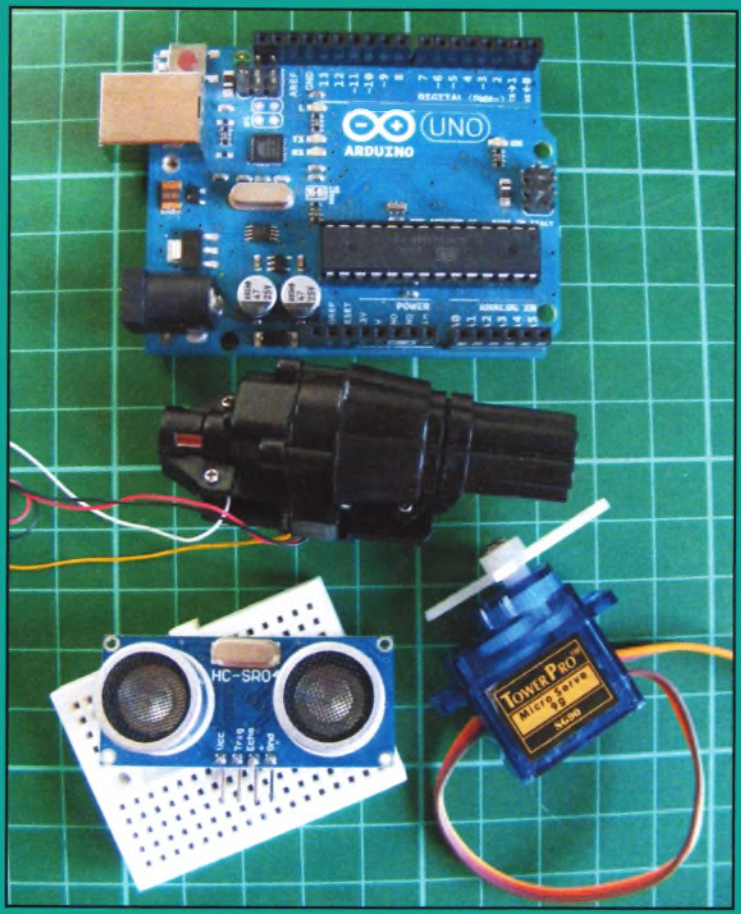
Как отмечалось ранее, когда на фоторезистор светит лазер, значение сопротивления составляет около 620, поэтому в скетче я настроил звуковое оповещение, когда оно превышает 850. Так я буду знать, что лазерный луч не достигает фоторезистора, если значение превысило 850.

```
int buzzPin = 11;     // Контакт, к которому подключен пьезоизлучатель
int LED = 13;         // Контакт, к которому подключен светодиод

void setup() {
   pinMode(buzzPin, OUTPUT);          // Перевод контакта в режим вывода
   pinMode(LED, OUTPUT);              // Перевод контакта в режим вывода
}

void loop() {
   if (analogRead(0) > 850) {         // Измените значение в зависимости
                                      // от вашего результата работы
                                      // фоторезистора
      digitalWrite(buzzPin, HIGH);    // Если значение больше 850,
                                      // на пьезоизлучатель подается
                                      // питание
      digitalWrite(LED, LOW);         // Если значение больше 850,
                                      // светодиод отключается
      delay(1000);                    // Задержка в 1 секунду
      digitalWrite(buzzPin, LOW);
      digitalWrite(LED, LOW);
   } else {
      digitalWrite(buzzPin, LOW);     // Если значение равно или меньше
                                      // 850 (свет попадает на
                                      // фоторезистор), пьезоизлучатель
                                      // не включается
      digitalWrite(LED, HIGH);        // Если значение равно или меньше
                                      // 850 (свет попадает на
                                      // фоторезистор), светодиод включен
   }
}
```

ПРОЕКТ 20: АВТОМАТИЧЕСКАЯ ТУРЕЛЬ

АВТОМАТИЧЕСКАЯ ТУРЕЛЬ — НЕУПРАВЛЯЕМОЕ ОРУЖИЕ, СПОСОБНОЕ АВТОНОМНО ОПРЕДЕЛЯТЬ ВРАЖЕСКИЕ ЦЕЛИ С ПОМОЩЬЮ УЛЬТРАЗВУКОВОГО ДАЛЬНОМЕРА И ВЕСТИ ПО ЦЕЛЯМ ОГОНЬ. В ЭТОМ ПРОЕКТЕ МЫ СОЗДАДИМ МИНИАТЮРНУЮ ВЕРСИЮ ТАКОГО «ОХРАННИКА».





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная мини-плата
- Перемычки
- Перемычки папа-папа
- Четырехконтактный ультразвуковой дальномер HC-SR04
- Ракетная установка WLT-V959-19
- Сервопривод Tower Pro 9g SG90

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- Servo
- NewPing

ПРИНЦИП РАБОТЫ

Мы подключим игрушечную ракетную установку и ультразвуковой дальномер к сервоприводу (см. рис. 20.1), чтобы сервопривод поворачивал оружие и дальномер назад и вперед на 180 градусов, увеличивая охраняемую территорию. Когда обнаружен неприятель, Arduino подает питание на установку и выпускает ракеты. Подробнее об ультразвуковом дальномере вы можете узнать в проекте 18.

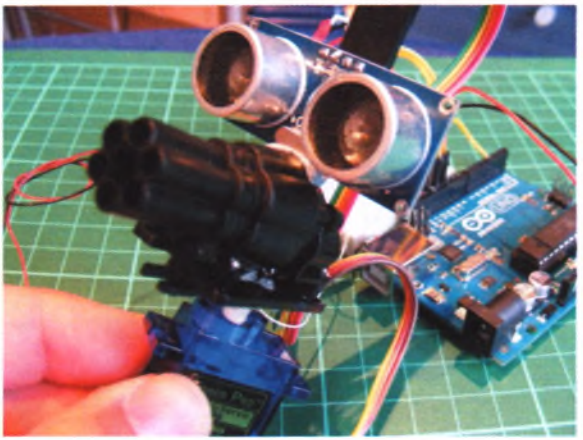


РИСУНОК 20.1

Монтаж ракетной установки и ультразвукового дальномера на сервопривод увеличивает область обзора, на которой фиксируется вторжение

Ключевой компонент этого проекта — игрушечная ракетная установка WLT-V959-19, предназначенная для квадрокоптеров (рис. 20.2).

РИСУНОК 20.2

Ракетная установка WLT-V959-19

Эта установка продается по низкой цене (около 300–400 рублей) и широко доступна в Интернете. Внутри этой ракетной установки находится мини-сервопривод, который вращается для запуска ракет. Провода, управляющие этим

сервоприводом — белый (заземление) и желтый (питание 5 В). Вы также увидите черный и красный провода, предназначенные для одиночной стрельбы, но мы будем использовать только желтый и белый для непрерывного ведения огня в духе пулемета Гатлинга.

СБОРКА

1. Вначале подготовим игрушечную ракетную установку. Осторожно вытащите четыре провода из маленького пластикового гнезда; у вас не должно возникнуть трудностей. Вы можете воспользоваться перемычкой папа-папа, чтобы вытащить их.
2. Провода установки тонкие и легко гнутся, поэтому зачистите концы желтого и белого проводов и припаяйте их к твердым сердечникам (или перемычкам), которые можно вставить в гнезда макетной платы и Arduino, как показано на рис. 20.3. Обрежьте черный и красный провода или закрепите их на корпусе.

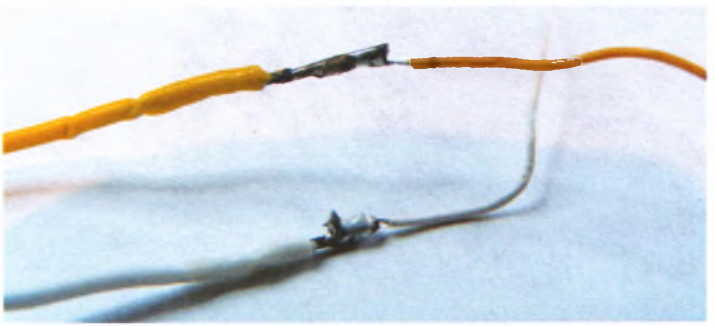


РИСУНОК 20.3

Зачистка и пайка проводов ракетной установки

3. Приклейте рычаг сервопривода к основанию ракетной установки, как показано на рис. 20.4.

РИСУНОК 20.4

Приклеивание рычага сервопривода

4. Прикрепите ультразвуковой дальномер к верхней части ракетной установки, как показано на рис. 20.5. Вы можете использовать клеевой пистолет для надежного крепления или скотч, если планируете в дальнейшем разобрать устройство.

РИСУНОК 20.5

Подключение ультразвукового дальномера

5. Используйте перемычки для подключения ультразвукового дальномера к Arduino: подключите ножку Trig дальномера к контакту 13, а ножку Echo — к контакту 12 платы Arduino. Мы будем использовать макетную мини-плату, чтобы можно было выполнить несколько подключений к контактам 5V и GND платы Arduino.

УЛЬТРАЗВУКОВОЙ ДАЛЬНОМЕР	ARDUINO
Ножка VCC	Контакт 5V
Ножка Trig	Контакт 13
Ножка Echo	Контакт 12
Ножка GND	Контакт GND

6. Подключите коричневый провод сервопривода к контакту GND, а красный провод — к 5V платы Arduino через мини-плату, а желтый (белый) провод — напрямую к контакту 9 платы Arduino.

СЕРВОПРИВОД	ARDUINO
Красный провод	Контакт GND
Коричневый провод	Контакт 5V
Желтый провод	Контакт 9

7. Подключите белый провод ракетной установки к шине заземления мини-платы, а желтый провод — непосредственно к контакту 3 платы Arduino.

РАКЕТНАЯ УСТАНОВКА	ARDUINO
Белый провод	Контакт GND
Желтый провод	Контакт 3

8. Ваша турель должна выглядеть так, как показано на рис. 20.6. Установите ракеты в пусковую установку.

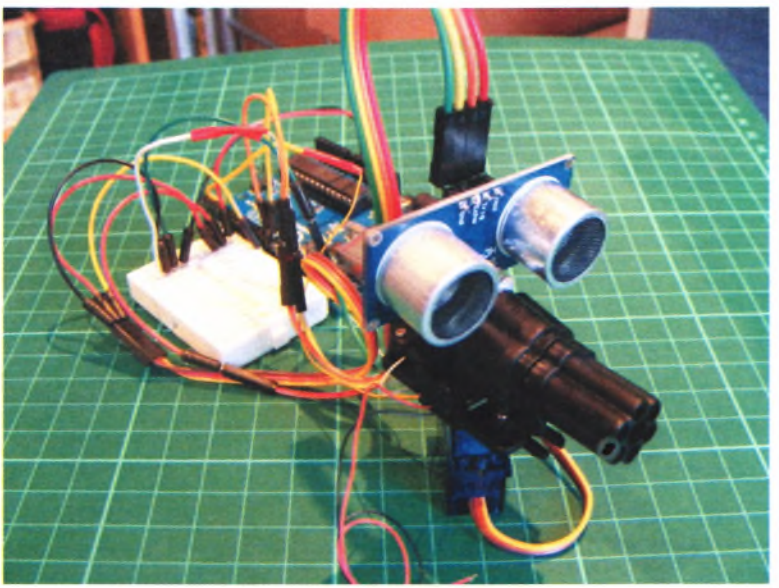


РИСУНОК 20.6

Турель готова к атаке!

9. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 20.7, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

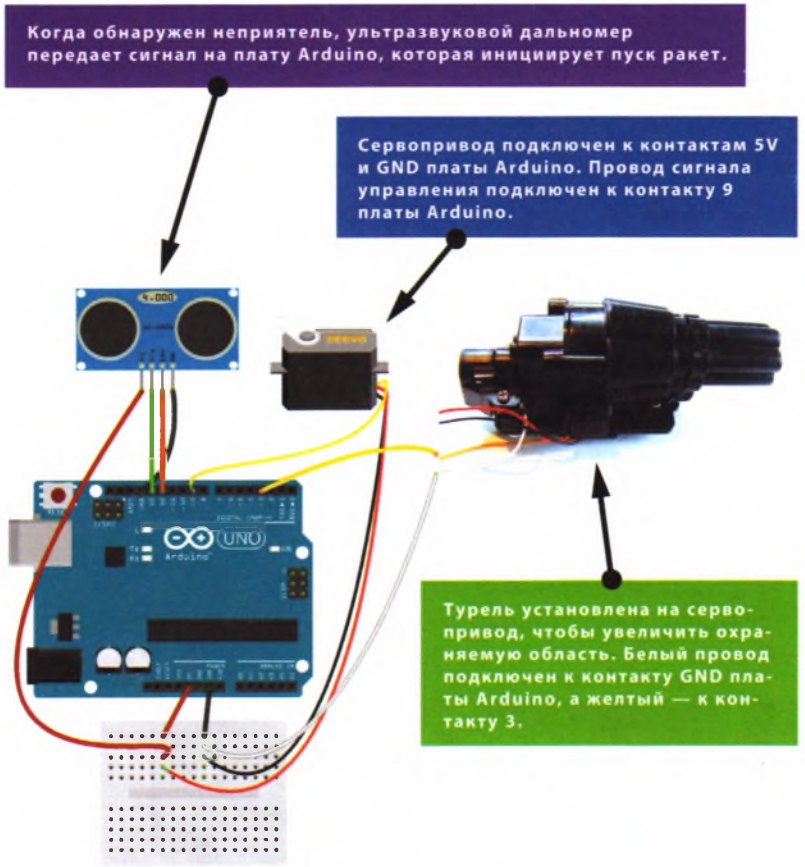


РИСУНОК 20.7

Принципиальная схема цепи автоматической турели

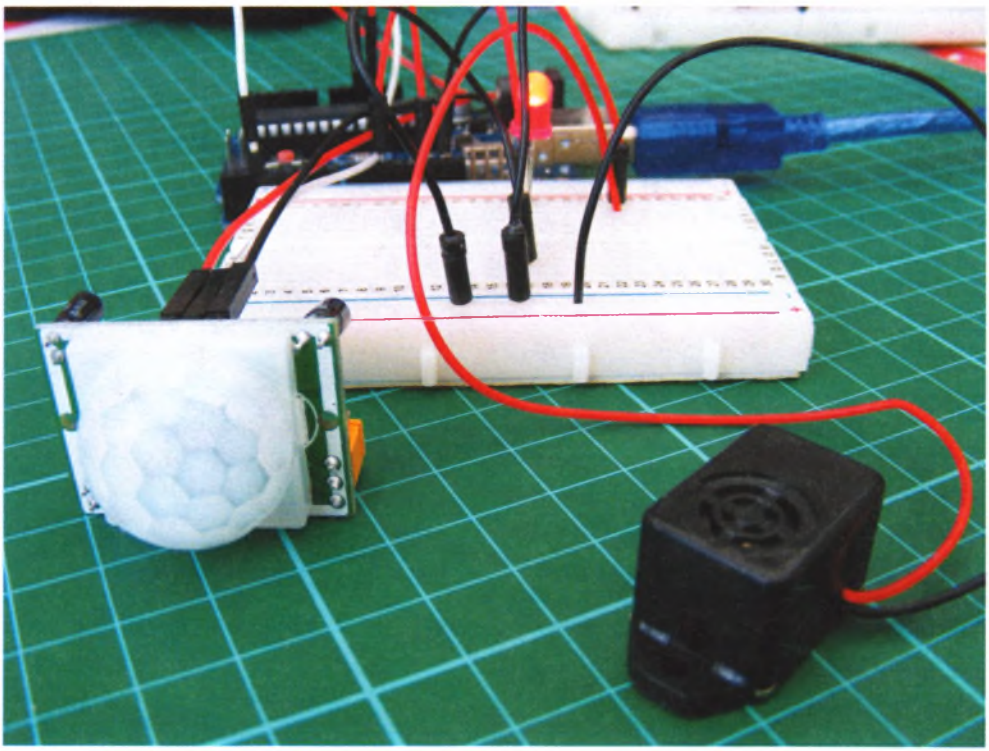
СКЕТЧ

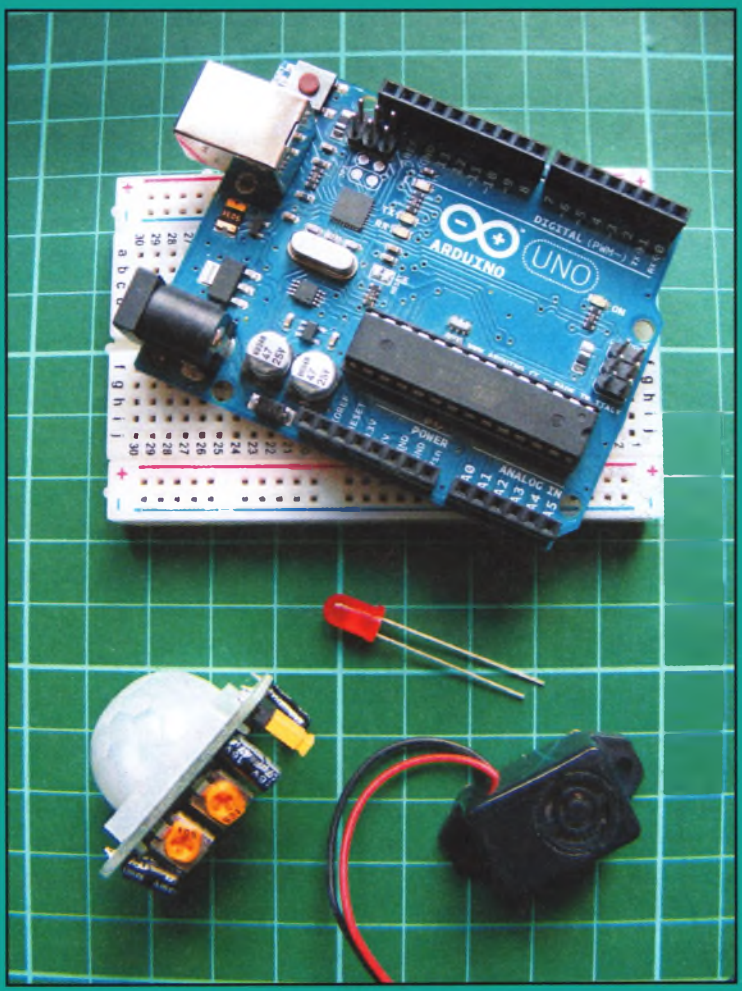
Код скетча сначала вызывает библиотеки NewPing и Servo для доступа к функциям, необходимым для управления сервоприводом и ультразвуковым дальномером, соответственно. (Убедитесь, что библиотека NewPing загружена по ссылке *eksmo.ru/files/arduino_geddes.zip* и сохранена в папке среды разработки Arduino.) Сервопривод сначала движется назад, а затем вперед, перемещая ультразвуковой дальномер на 180 градусов. Дальномер посылает ультразвуковой сигнал (пинг), и когда он достигает объекта, сигнал возвращается обратно через определенный промежуток времени. Плата Arduino преобразует полученное значение в расстояние между датчиком и объектом. Если расстояние до объекта меньше или равно 15 см, сервопривод останавливается, а на ракетную установку подается питание для выпуска снаряда по объекту. Вы можете изменить указанное предельное расстояние (в сантиметрах) в строке ❶.

```
#include <NewPing.h> // Вызов библиотеки NewPing
#include <Servo.h>   // Вызов библиотеки Servo
#define trigPin 12   // Контакт, к которому подключена ножка Trig
                     // дальномера
#define echoPin 13   // Контакт, к которому подключена ножка Echo
                     // дальномера
#define MAX_DISTANCE 500

NewPing sonar(trigPin, echoPin, MAX_DISTANCE);

int blaster = 3;     // Контакт, к которому подключена ракетная
                     // установка

int angle = 0;       // Установка позиции сервопривода в градусах

Servo servo;

void setup() {
   Serial.begin (115200);
   pinMode(trigPin, OUTPUT);
   pinMode(echoPin, INPUT);
   pinMode(blaster, OUTPUT);
   servo.attach(9);  // Контакт, к которому подключен сервопривод
}

void loop() {
   int duration, distance, pos = 0, i;
   digitalWrite(trigPin, LOW);
   delayMicroseconds(2);
   digitalWrite(trigPin, HIGH);    // Контакт trigPin посылает пинг
   delayMicroseconds(10);
   digitalWrite(trigPin, LOW);
   duration = pulseIn(echoPin, HIGH);      // Контакт echoPin получает
                                           // пинг
   distance = (duration / 2) / 29.1;
   Serial.print(distance);
   Serial.println(" cm");
   if (distance <= 15) {       // Если расстояние меньше 15 см
      digitalWrite(blaster, HIGH);  // Открывается огонь
      servo.write(90);
   }
   else {
      digitalWrite(blaster, LOW);   // Иначе установка не активируется
      for (angle = 0; angle < 180; angle++) {     // Поворот
                                                  // сервопривода
         servo.write(angle);
         delay(15);
      }
      for (angle = 180; angle > 0; angle--) {
         servo.write(angle);
      }
      delay(450);
   }
}
```

ПРОЕКТ 21: ДАТЧИК ДВИЖЕНИЯ

В ЭТОМ ПРОЕКТЕ МЫ ПОСТРОИМ СИСТЕМУ ОПОВЕЩЕНИЯ О ДВИЖЕНИИ, ИСПОЛЬЗУЯ ПАССИВНЫЙ ИНФРАКРАСНЫЙ ДАТЧИК ДВИЖЕНИЯ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Пассивный инфракрасный датчик движения HC-SR501
- Светодиод
- Пьезоизлучатель

Вы можете использовать данное устройство для выполнения различных функций, таких как световое оповещение, перемещение чего-либо или даже вывод сообщения «Добро пожаловать домой», когда вы приближаетесь к входной двери.

ПРИНЦИП РАБОТЫ

Этот проект основан на датчике HC-SR501, который широко доступен в Интернете за пару сотен рублей. Мы настроим датчик таким образом, чтобы при появлении перед ним человека загорался светодиод и звучал пьезоизлучатель (см. рис. 21.1). Вы также можете адаптировать его под другие цели.

РИСУНОК 21.1

Для данного проекта подойдет любой пьезоизлучатель. Главное, соблюдайте полярность: красный провод подключается к источнику питания с напряжением 5 В, а черный — к заземлению

Другие подобные датчики движения также подойдут для нашего проекта, но важно проверить расположение контактов вашей модели датчика по файлу спецификации, поскольку они могут отличаться. У всех датчиков должны быть контакты питания, заземления и выходы для передачи данных. В этой модели контакты с первого взгляда не определить, но если вы снимете внешнюю линзу (она держится на заклепках и может быть легко отсоединена), можно увидеть метки контактов, как показано на рис. 21.2.

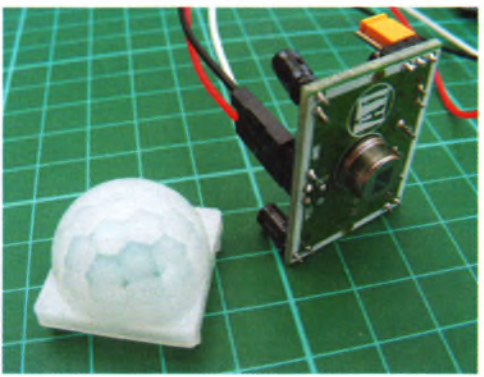


РИСУНОК 21.2

Датчик движения со снятой линзой

Два оранжевых потенциометра на датчике позволяют его настраивать. При положении датчика, показанном на рис. 21.3, левый потенциометр управляет длительностью работы выхода в режиме HIGH при обнаружении объекта и позволяет установить это время в диапазоне от 5 до 200 с. Если к выходу подключен светодиод, потенциометр определяет длительность его свечения от 5 до 200 секунд в зависимости от настройки. Правый потенциометр ограничивает диапазон обнаружения объектов от 0 до 7 метров.

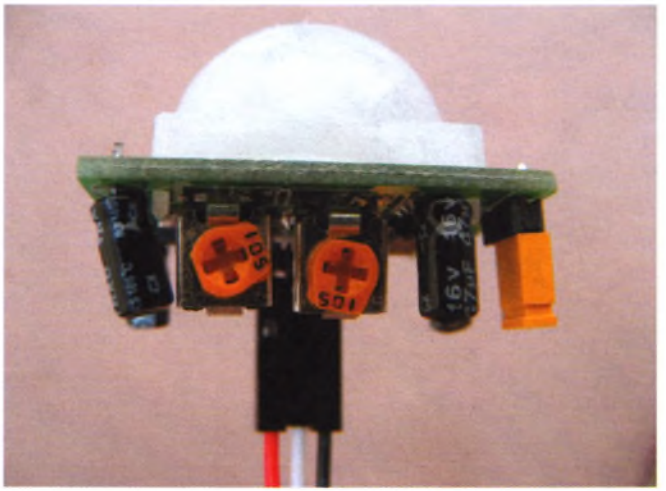


РИСУНОК 21.3

Потенциометры датчика движения. Левый определяет, как долго выход будет находиться в состоянии HIGH (5–200 секунд); правый управляет чувствительностью датчика (0–7 метров)

Датчик фиксирует инфракрасное излучение, испускаемое теплыми объектами. Кристаллический материал внутри датчика реагирует на инфракрасное излучение. По достижении им установленного уровня он запускает выходной сигнал датчика. Плата Arduino определяет этот сигнал как напряжение, поэтому мы можем использовать его как простой переключатель, чтобы включить что-то, в данном случае светодиод.

Мы настроим устройство так, чтобы при срабатывании раздавался звуковой сигнал. Также допустимы другие действия. Например, вы можете напугать своих друзей, подключив сервопривод и настроив его так, чтобы, когда они пройдут мимо, сработала рогатка.

СБОРКА

1. Подключите контакты датчика движения 5V и GND к шинам питания и заземления макетной платы, соответственно, а затем соедините эти шины с соответствующими контактами платы Arduino. Подключите контакт OUT датчика к контакту 2 платы Arduino (см. рис. 21.4).

ДАТЧИК ДВИЖЕНИЯ	ARDUINO
Контакт 5V	Контакт 5V
Контакт GND	Контакт GND
Контакт OUT	Контакт 2

РИСУНОК 21.4

Датчик движения с подключенной перемычкой

2. Установите светодиод на макетную плату и подключите длинную ножку (анод) к контакту 13 платы Arduino, а короткую — к контакту GND. В этом проекте для работы светодиода вам не понадобится резистор.

СВЕТОДИОД	ARDUINO
Короткая ножка (катод)	Контакт 13
Длинная ножка (анод)	Контакт GND

3. Подключите пьезоизлучатель: красный провод к контакту 10 платы Arduino, а черный — к контакту GND.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Красный провод	Контакт 10
Черный провод	Контакт GND

4. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 21.5, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

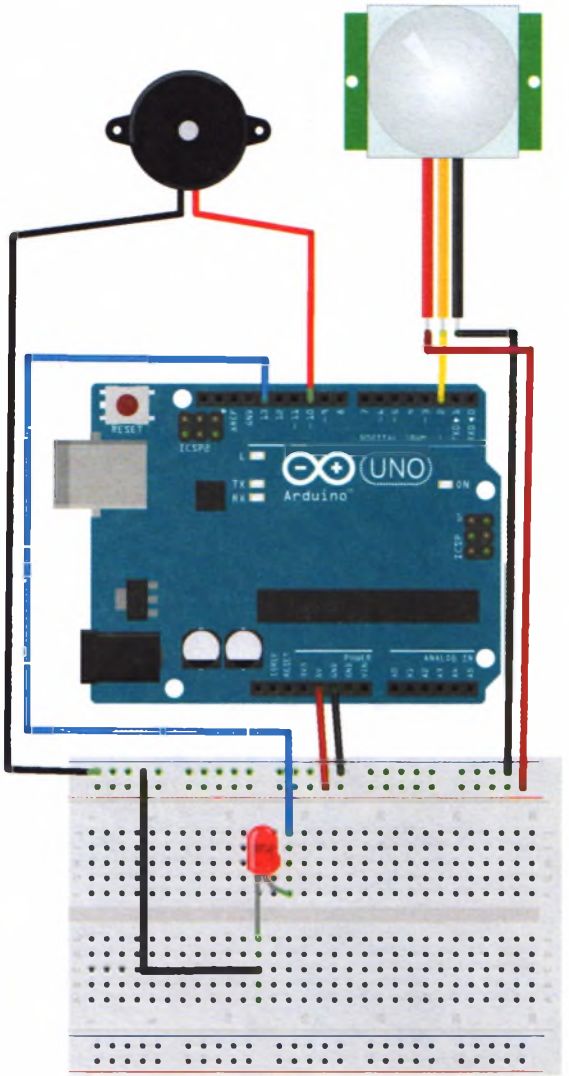


РИСУНОК 21.5

Принципиальная схема цепи датчика движения

СКЕТЧ

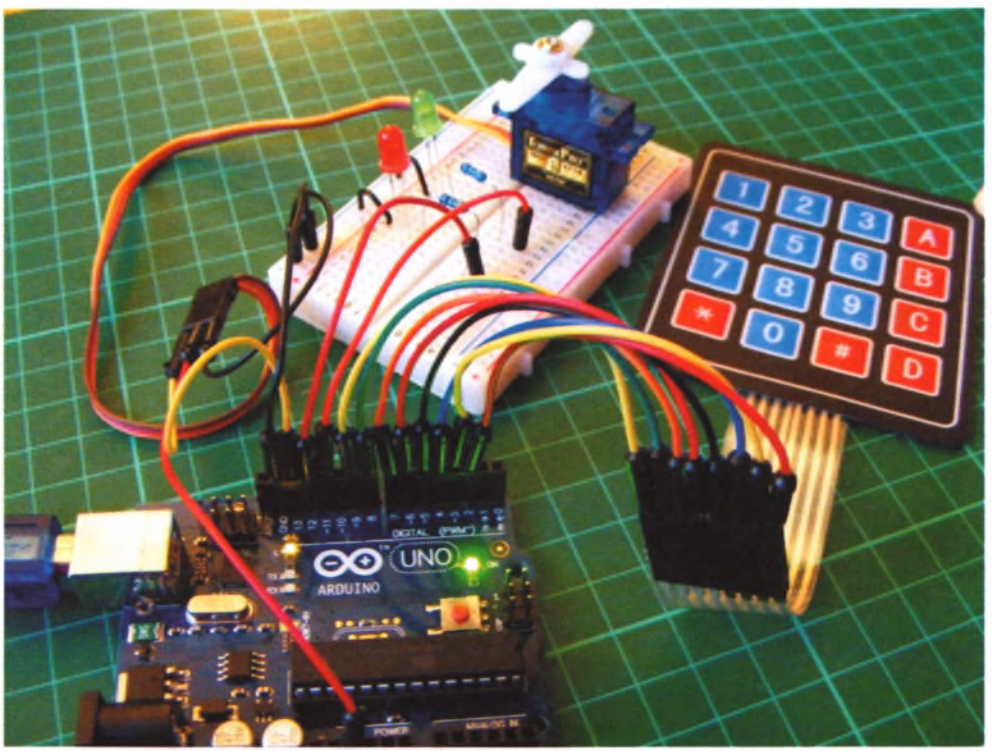
В скетче контакт 13 платы Arduino устанавливается как выход для светодиода, контакт 2 — как вход для датчика движения, а контакт 10 — как выход для пьезоизлучателя. Когда срабатывает датчик движения, на Arduino посылается сигнал HIGH, который, в свою очередь, инициирует включение светодиода и пьезоизлучателя.

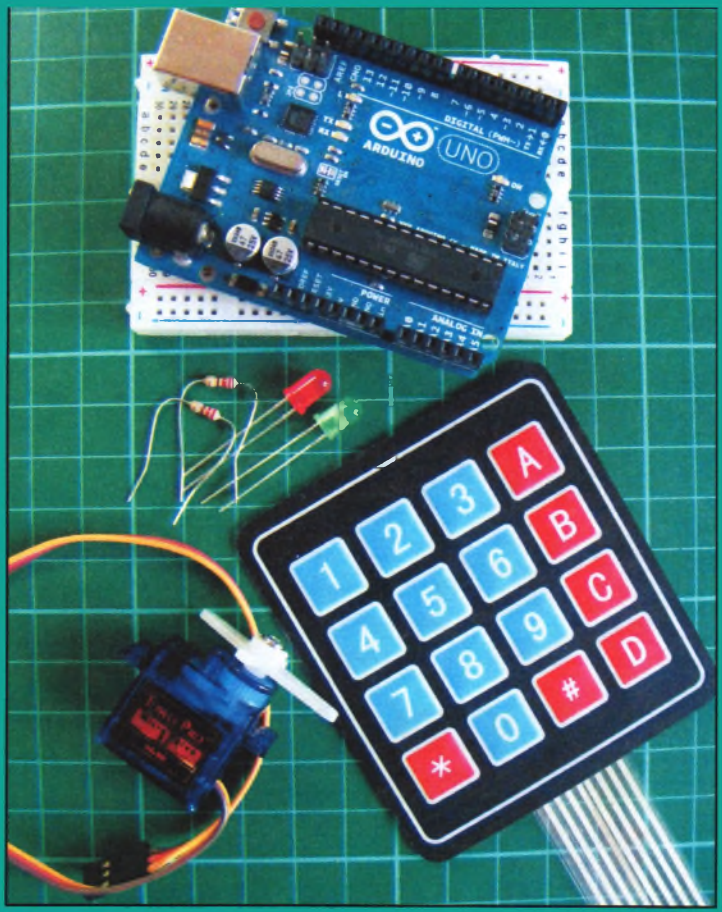
```
int ledPin = 13;      // Контакт, к которому подключен светодиод
int inputPin = 2;     // Контакт, к которому подключен датчик движения
```

```
int pirState = LOW;             // Начало с состояния датчика движения LOW
                                // (нет движения)
int val = 0;                    // Переменная для считывания состояния
                                // контакта
int pinSpeaker = 10;            // Контакт, к которому подключен
                                // пьезоизлучатель

void setup() {
   pinMode(ledPin, OUTPUT);  // Перевод контакта светодиода в режим
   выхода
   pinMode(inputPin, INPUT); // Перевод контакта датчика в режим входа
   pinMode(pinSpeaker, OUTPUT);
   Serial.begin(9600);
}

void loop() {
   val = digitalRead(inputPin);     // Считывание входного значения
                                    // датчика
   if (val == HIGH) {               // Проверка, является ли значение
                                    // входа HIGH
      digitalWrite(ledPin, HIGH);   // Если нет, включается светодиод
      playTone(300, 160);
      delay(150);
      if (pirState == LOW) {
         // Передача на монитор порта, если обнаружено движение
         Serial.println("Motion detected!");

         pirState = HIGH;
      }
   } else {
         digitalWrite(ledPin, LOW); // Если входное значение не HIGH,
                                    // выключение светодиода
         playTone(0, 0);
         delay(300);
         if (pirState == HIGH) {
         Serial.println("Motion ended!");
         pirState = LOW;
      }
   }
}

void playTone(long duration, int freq) {   // Продолжительность в мс,
                                           // частота в Гц
   duration *= 1000;
   int period = (1.0 / freq) * 1000000;
   long elapsed_time = 0;
   while (elapsed_time < duration) {
      digitalWrite(pinSpeaker, HIGH);
      delayMicroseconds(period / 2);
      digitalWrite(pinSpeaker, LOW);
      delayMicroseconds(period / 2);
      elapsed_time += (period);
   }
}
```

ПРОЕКТ 22: СИСТЕМА ВВОДА С КЛАВИАТУРЫ

САМОЕ ВРЕМЯ ДОБАВИТЬ К ARDUINO КЛАВИАТУРУ ДЛЯ СБОРКИ СИСТЕМЫ ВВОДА ПАРОЛЯ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- Сервопривод Tower Pro 9g SG90
- Зеленый светодиод
- Красный светодиод
- Мембранная клавиатура размером 4×4 кнопки
- 2 резистора с сопротивлением 220 Ом

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- Keypad
- Servo
- Password

В этом проекте используется 4 × 4 мембранная клавиатура со шлейфом из восьми проводов снизу. К устройству подключен сервопривод, который открывает замок.

ПРИНЦИП РАБОТЫ

Клавиатура представляет собой группу кнопок, которые возвращают число или символ в зависимости от того, какая кнопка нажата. Для клавиатуры, обращенной лицевой стороной, провода нумеруются от 1 до 8 слева направо. Первые четыре провода соответствуют строкам кнопок, а последние четыре — столбцам.

Вам необходимо загрузить библиотеку для клавиатуры в архиве по ссылке *eksmo.ru/files/arduino_geddes.zip* и сохранить ее в папке среды разработки Arduino.

Мы подключим эту клавиатуру к сервоприводу и двум светодиодам, чтобы создать систему защиты, подобную электронному привратнику из проекта 9. Чтобы получить доступ, введите пароль и нажмите кнопку «звездочка» (*) для подтверждения. Если введенный код соответствует указанному в скетче паролю, замигает зеленый светодиод и сервопривод повернется на 90 градусов. Если код неправильный, загорится красный светодиод. Используйте кнопку с символом хэша (#) для повторного ввода пароля.
Вы можете сменить используемый сервопривод на более мощный, способный открыть тяжелую защелку на двери или запереть ящик стола изнутри. Клавиатура и светодиоды устанавливаются снаружи.

ПРОВЕРКА КЛАВИАТУРЫ

Вначале мы протестируем клавиатуру с помощью следующего кода.

```
#include <Keypad.h>

const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
   {'1','2','3','A'},
   {'4','5','6','B'},
   {'7','8','9','C'},
   {'*','0','#','D'}
};
byte rowPins[ROWS] = {2,3,4,5};
byte colPins[COLS] = {6,7,8,9};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
   Serial.begin(9600);
}

void loop() {
   char key = keypad.getKey();
   if (key != NO_KEY){
      Serial.println(key);
   }
}
```

Загрузите этот код на плату и откройте окно монитора порта в среде разработки Arduino (рис. 22.1).

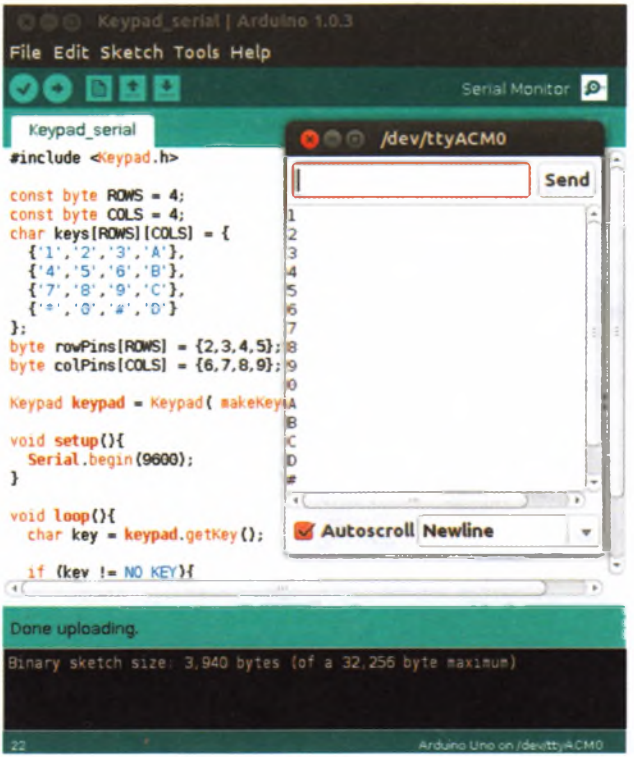


РИСУНОК 22.1

Тестирование клавиатуры

Держа клавиатуру лицевой стороной вверх, последовательно подключите провода слева направо к цифровым контактам 9–2 платы Arduino. После загрузки кода нажмите несколько кнопок на клавиатуре. При нажатии каждой кнопки соответствующий символ должен появляться в отдельной строке консоли среды разработки Arduino Arduino.

СБОРКА

1. Подключите контакты клавиатуры непосредственно к контактам Arduino, как указано в таблице ниже. Контакты клавиатуры пронумерованы, как показано на рис. 22.2.

КЛАВИАТУРА	ARDUINO
Контакт 1	Контакт 9
Контакт 2	Контакт 8

КЛАВИАТУРА	ARDUINO
Контакт 3	Контакт 7
Контакт 4	Контакт 6
Контакт 5	Контакт 5
Контакт 6	Контакт 4
Контакт 7	Контакт 3
Контакт 8	Контакт 2

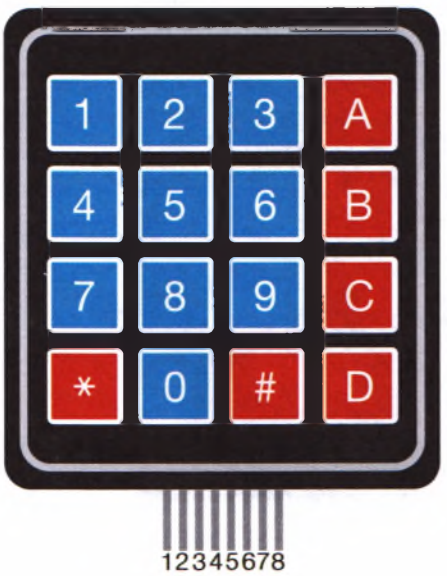


РИСУНОК 22.2

Контакты 1–8 клавиатуры

2. Установите зеленый и красный светодиоды на макетную плату. Короткие ножки (катоды) подключите к шине заземления, а длинные ножки (аноды) через резисторы с сопротивлением 220 Ом к контакту 11 (зеленый светодиод) и контакту 12 (красный светодиод) платы Arduino.

СВЕТОДИОДЫ	ARDUINO
Длинная ножка (анод) зеленого светодиода	Контакт 11 через резистор с сопротивлением 220 Ом
Длинная ножка (анод) красного светодиода	Контакт 12 через резистор с сопротивлением 220 Ом
Короткие ножки (катоды)	Контакт GND

3. Теперь подключите сервопривод (см. рис. 22.3). Подключите коричневый провод к шине заземления, красный — к шине питания 5 В, а желтый (белый) провод непосредственно к контакту 13 платы Arduino.

СЕРВОПРИВОД	ARDUINO
Красный провод	Контакт 5V
Коричневый провод	Контакт GND
Желтый провод	Контакт 13

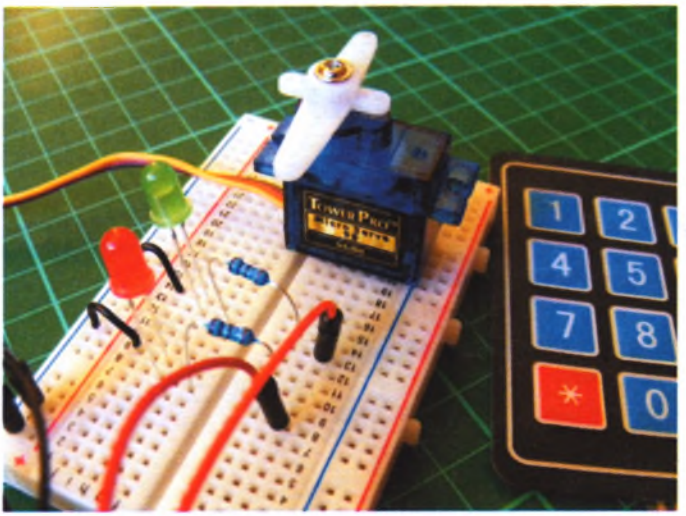


РИСУНОК 22.3

Подключение сервопривода

4. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 22.4, и загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

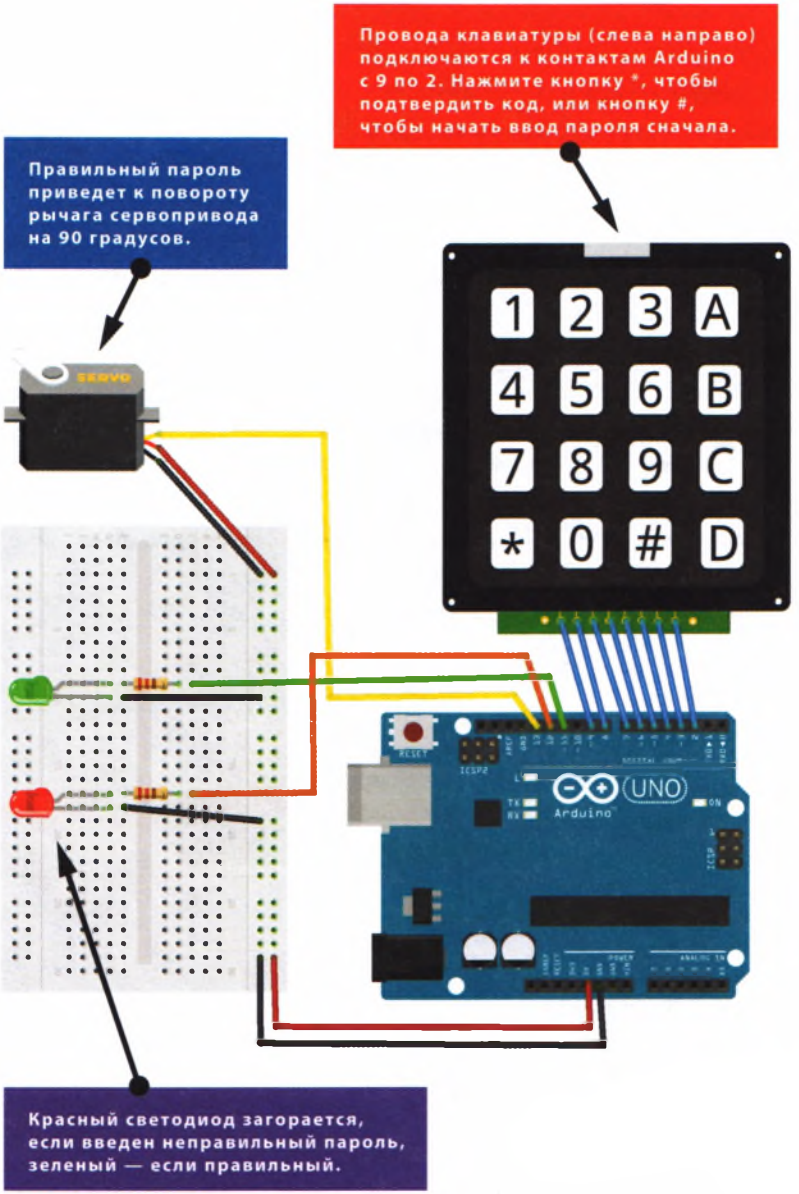


РИСУНОК 22.4

Принципиальная схема цепи системы ввода с клавиатуры

СКЕТЧ

Сначала в коде скетча вызываются библиотеки Keypad, Servo и Password. Библиотека Servo входит в дистрибутив среды разработки Arduino, а библиотеки Keypad и Password необходимо загрузить из архива по ссылке *eksmo.ru/files/arduino_geddes.zip* и сохранить в папке среды разработки Arduino. Затем мы настраиваем восемь контактов, принимающих ввод с клавиатуры, и определяем контакты 11 и 12 платы Arduino для управления светодиодами, а контакт 13 — для управления сервоприводом. Плата Arduino ждет ввода пароля с клавиатуры и нажатия кнопки * для подтверждения ввода. Как только вы нажали кнопку *, скетч проверяет соответствие пароля записанному в коде. Если пароль не соответствует записи, контакт красного светодиода переводится в режим HIGH, и светодиод начинает светиться; если пароль соответствует записи, контакт зеленого светодиода переводится в режим HIGH, и светодиод начинает светиться, а сервопривод повернется. Нажатие кнопки # сбрасывает выполнение кода, и скетч будет готов к следующему вводу пароля.

Чтобы задать другой пароль, измените число в кавычках в строке, показанной ниже.

```
Password password = Password("2468");
```

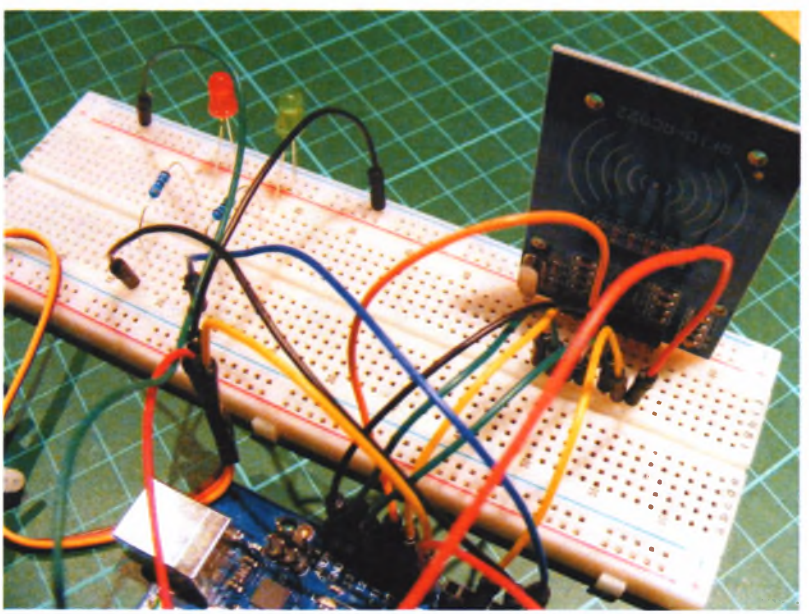
Пароль по умолчанию — 2468.

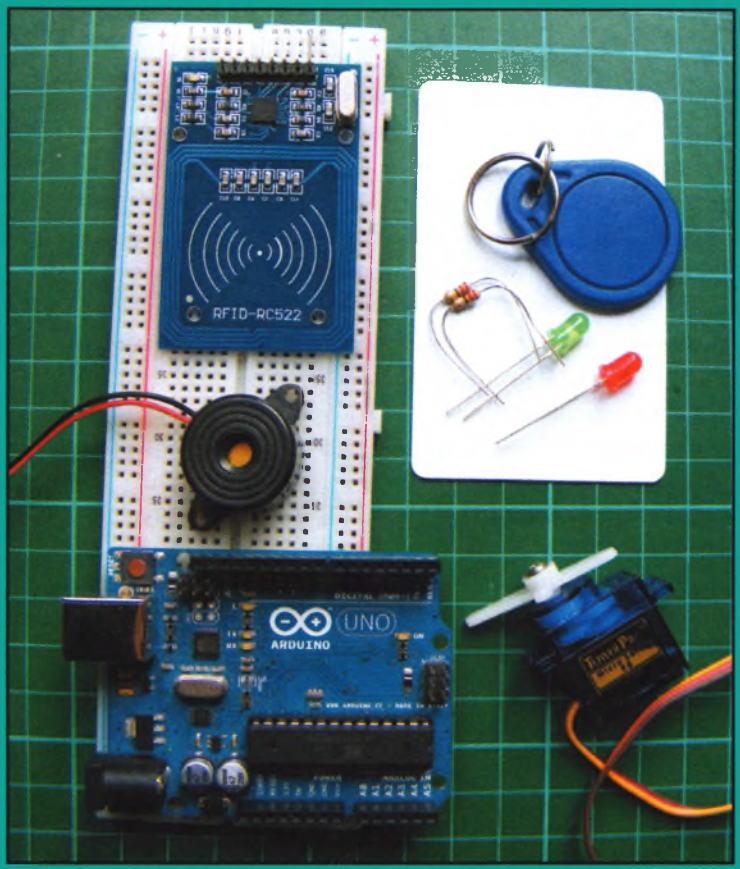
```
/* Библиотека Keypad для Arduino
   Авторы: Марк Стэнли, Александр Бревик
   http://playground.arduino.cc/Main/KeypadTutorial
*/

#include <Password.h>
#include <Keypad.h>
#include <Servo.h>

Servo myservo;
Password password = Password("2468");      // Записанный пароль

const byte ROWS = 4;                  // Установка 4 строк
const byte COLS = 4;                  // Установка 4 столбцов

char keys[ROWS][COLS] = {             // Определение схемы кнопок
   {'1','2','3','A'},
   {'4','5','6','B'},
   {'7','8','9','C'},
   {'*','0','#','D'}
};
byte rowPins[ROWS] = { 9,8,7,6 };     // Контакты, к которым подключена
                                      // клавиатура
                                          // ROW0, ROW1, ROW2 и ROW3
byte colPins[COLS] = { 5,4,3,2, };      // Контакты, к которым подключена
                                        // клавиатура
                                         // COL0, COL1 и COL2
// Создание клавиатуры
```

```
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);
void setup() {
   Serial.begin(9600);
   delay(200);
   pinMode(11, OUTPUT);       // Перевод контакта зеленого светодиода
                              // в режим выхода
   pinMode(12, OUTPUT);       // Перевод контакта красного светодиода
                              // в режим выхода
   myservo.attach(13);        // Контакт, к которому подключен
                              // сервопривод
   keypad.addEventListener(keypadEvent);   // Добавление слушателя
                              // событий для обнаружения нажатия кнопок
}

void loop() {
   keypad.getKey();
   myservo.write(0);
}

void keypadEvent(KeypadEvent eKey) {
   switch (keypad.getState()) {
      case PRESSED:
      Serial.print("Pressed: ");
      Serial.println(eKey);
      switch (eKey) {
         case '*': checkPassword(); break;
         case '#': password.reset(); break;
         default: password.append(eKey);
      }
   }
}

void checkPassword() {
   if (password.evaluate() ){
      Serial.println("Success");     // Если пароль правильный...
      myservo.write(90);             // Поворот сервопривода
                                     // на 90 градусов
      digitalWrite(11, HIGH);        // Включение зеленого светодиода
      delay(500);                    // Задержка 5 секунд
      digitalWrite(11, LOW);         // Выключение зеленого светодиода
   } else {
      Serial.println("Wrong");       // Если пароль неправильный...
      myservo.write(0);
      digitalWrite(12, HIGH);        // Включение красного светодиода
      delay(500);                    // Задержка 5 секунд
      digitalWrite(12, LOW);         // Выключение красного светодиода
   }
}
```

ПРОЕКТ 23: БЕСКОНТАКТНЫЙ ЭЛЕКТРОННЫЙ ПРОПУСК

В ЭТОМ ПРОЕКТЕ МЫ ИСПОЛЬЗУЕМ СЧИТЫВАТЕЛЬ РАДИОЧАСТОТНЫХ МЕТОК (RFID-МОДУЛЬ) ДЛЯ ПОСТРОЕНИЯ БЕСПРОВОДНОЙ СИСТЕМЫ ВВОДА ID-КАРТЫ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- Макетная плата
- Перемычки
- RFID-модуль Mifare RC-522, карта и брелок
- Сервопривод Tower Pro 9g SG90
- Пьезоизлучатель
- Красный светодиод
- Зеленый светодиод
- 2 резистора с сопротивлением 220 Ом

ТРЕБУЕМЫЕ БИБЛИОТЕКИ

- RFID
- SPI
- Wire
- Servo
- Pitches

ПРИНЦИП РАБОТЫ

RFID-модуль основан на беспроводной технологии для идентификации карты, метки или брелока без непосредственного контакта с ними. Модуль реагирует, если RFID-метка находится рядом с ним. Для нашего проекта нам нужно, чтобы модуль прочитал уникальный номер нашей RFID-карты. Также мы добавим сервопривод, который будет двигаться в зависимости от того, распознает RFID-модуль карту или нет. Можно использовать эту систему идентификации для блокировки двери или ящика, как в случае с электронным привратником из главы 9.

Возможно, вы видели наклейки, подобные показанным на рис. 23.1, на товарах в магазине. Эти наклейки представляют собой RFID-метки, позволяющие сотрудникам магазина защищать товары от воровства. Если вы пройдете с товаром через специальные RFID-турникеты на выходе без оплаты, прозвучит сигнал тревоги. RFID-модули и карты также часто используются для доступа определенного круга лиц к секретным лабораториям и военным базам.

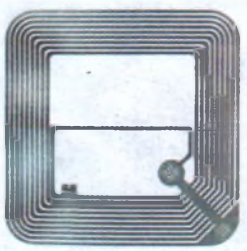


РИСУНОК 23.1
Наклейка с RFID-меткой

Существует два типа RFID-систем: пассивная и активная. Каждая RFID-система использует радиоволны определенной частоты для обмена сигналами между RFID-модулем и метками или картами. Этот сигнал содержит уникальный код метки или карты, и если RFID-модуль распознает этот код, он реагирует соответствующим образом, например позволяет без оповещения проносить товары через детекторы в магазине или разблокирует двери.

В пассивной системе, когда оба элемента находятся близко друг к другу, радиосигнал RFID-модуля обеспечивает питанием метку или карту, достаточным для обмена данными с ними. Активные системы оснащены и RFID-модулем с питанием, и метками с питанием, и позволяют считывать метки с гораздо большего расстояния. Активные системы очень дороги и используются для более сложных конструкций, поэтому мы будем использовать пассивную RFID-систему, состоящую из RFID-модуля Mifare RC-522 и пустой карты, и брелока, как показано на рис. 23.2. RFID-модуль работает на частоте 13,56 МГц и может идентифицировать карту или брелок, которые получат питание от самого RFID-модуля, только если

они находятся на расстоянии не более 6 см. Важно помнить об этом, настраивая свою RFID-систему.

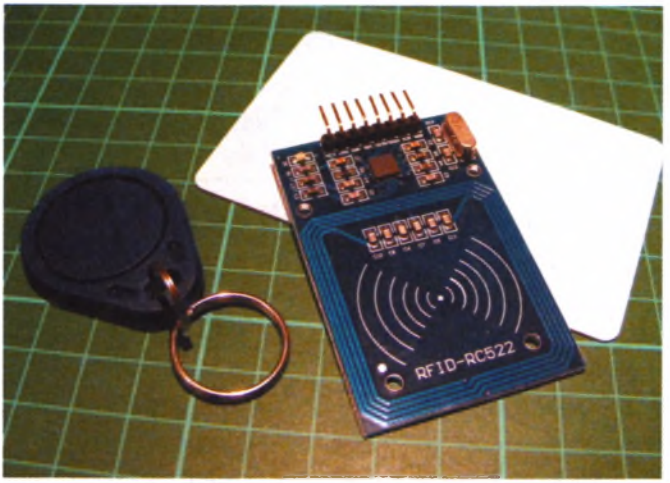


РИСУНОК 23.2

RFID-модуль с картой и брелоком для ключей

Мы сконструируем сервопривод, управляемый посредством RFID. Когда вы подносите карту к RFID-модулю, он считывает карту. Если модуль распознает карту и на ней записаны права доступа, загорается зеленый светодиод, звучит сигнал, и сервопривод поворачивается на 180 градусов. Если модуль не распознает карту, загорается красный светодиод, звучит другой сигнал, и сервопривод не двигается.

В табл. 23.1 описаны различные функции RFID-модуля.

ТАБЛИЦА 23.1

Функции контактов RFID-модуля

КОНТАКТЫ RFID	РАСШИФРОВКА	ПРИМЕЧАНИЕ
3.3V	3.3 В	Для питания модуля допустимо только такое напряжение.
RST	Сброс	Сброс модуля до исходного состояния.
GND	Заземление	Подключается к контакту GND платы Arduino.
IRQ	Запрос на прерывание	Не используется в этом проекте.
MISO	Вход ведущего, выход ведомого	Служит для передачи данных от ведомого устройства ведущему.
MOSI	Выход ведущего, вход ведомого	Служит для передачи данных от ведущего устройства ведомому.

КОНТАКТЫ RFID	РАСШИФРОВКА	ПРИМЕЧАНИЕ
SCK	Последовательный тактовый сигнал	Служит для передачи тактового сигнала для ведомых устройств.
SDA/SS	Последовательные данные/Выбор ведомого	Модуль может иметь контакт SDA или SS, хотя они одинаковы. С помощью него Arduino и модуль обмениваются данными.
Pin 16	VCC	Питание.

СБОРКА

1. Возможно, для установки модуля вам потребуется сначала припаять к нему штырьковые соединители. Выломайте полоску из восьми штырьков. Припаяйте по одному к каждому отверстию. Обязательно удерживайте паяльник в месте пайки не более нескольких секунд, чтобы не повредить микросхему. Если вы никогда не паяли до этого, см. раздел «Краткое руководство по пайке» в проекте 0.
2. Установите RFID-модуль на макетную плату, как показано на рис. 23.3, а затем подключите штырьки RFID-модуля к контактам Arduino согласно следующей таблице. Не забудьте, что RFID-модуль следует подключать к источнику питания напряжением 3,3 В (но не 5 В !), иначе модуль выйдет из строя.

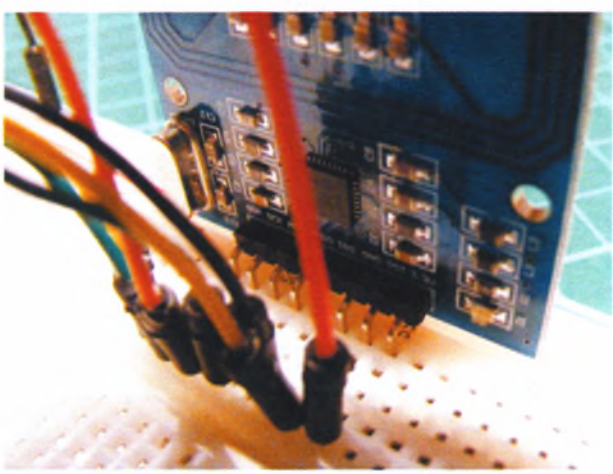


РИСУНОК 23.3

Установка модуля RFID на макетную плату

RFID-МОДУЛЬ	ARDUINO
Контакт 3.3V	Контакт 3.3V
Контакт RST	Контакт 5
Контакт GND	Контакт GND
Контакт IRQ	Не используется
Контакт MISO	Контакт 12
Контакт MOSI	Контакт 11
Контакт SCK	Контакт 13
Контакт SDA	Контакт 10

3. Теперь нам нужно проверить работу RFID-модуля. Загрузите библиотеку RFID из архива по ссылке *eksmo.ru/files/arduino_geddes.zip* и установите ее в среде разработки Arduino, как это показано в разделе «Библиотеки» проекта 0. Загрузите следующий тестовый скетч для RFID-модуля. Плата Arduino должна быть подключена USB-кабелем к компьютеру.

```
// Библиотека RFID создана Мигелем Бальбоа (circuitito.com)
#include <SPI.h>
#include <RFID.h>
#define SS_PIN 10
#define RST_PIN 9
RFID rfid(SS_PIN, RST_PIN);

// Определение переменных
int serNum0;
int serNum1;
int serNum2;
int serNum3;
int serNum4;

void setup() {
   Serial.begin(9600);
   SPI.begin();
   rfid.init();
}

void loop() { // В этом цикле выполняется поиск карт для считывания
   if (rfid.isCard()) {
      if (rfid.readCardSerial()) {
         if (rfid.serNum[0] != serNum0
            && rfid.serNum[1] != serNum1
            && rfid.serNum[2] != serNum2
            && rfid.serNum[3] != serNum3
            && rfid.serNum[4] != serNum4
```

```
            ) {
            // Если карта найдена, выполняется следующий код
            Serial.println(" ");
            Serial.println("Card found");
            serNum0 = rfid.serNum[0];
            serNum1 = rfid.serNum[1];
            serNum2 = rfid.serNum[2];
            serNum3 = rfid.serNum[3];
            serNum4 = rfid.serNum[4];

            // Выводится идентификатор карты в мониторе порта IDE
            Serial.println("Cardnumber:");
            Serial.print("Dec: ");
            Serial.print(rfid.serNum[0], DEC);
            Serial.print(", ");
            Serial.print(rfid.serNum[1], DEC);
            Serial.print(", ");
            Serial.print(rfid.serNum[2], DEC);
            Serial.print(", ");
            Serial.print(rfid.serNum[3], DEC);
            Serial.print(", ");
            Serial.print(rfid.serNum[4], DEC);
            Serial.println(" ");
            Serial.print("Hex: ");
            Serial.print(rfid.serNum[0], HEX);
            Serial.print(", ");
            Serial.print(rfid.serNum[1], HEX);
            Serial.print(", ");
            Serial.print(rfid.serNum[2], HEX);
            Serial.print(", ");
            Serial.print(rfid.serNum[3], HEX);
            Serial.print(", ");
            Serial.print(rfid.serNum[4], HEX);
            Serial.println(" ");
          } else {
            // Если идентификатор совпадает, выводится точка
            // в мониторе порта
            Serial.print(".");
          }
        }
    }
    rfid.halt();
}
```

4. Откройте окно монитора порта в среде разработки Arduino.
5. Проведите картой или брелком перед RFID-модулем. В мониторе порта должен отобразиться уникальный номер, как показано на рис. 23.4. Запишите этот номер, он понадобится в дальнейшем. В данном случае номер моей карты был таким: 4D 55 AD D3 66.

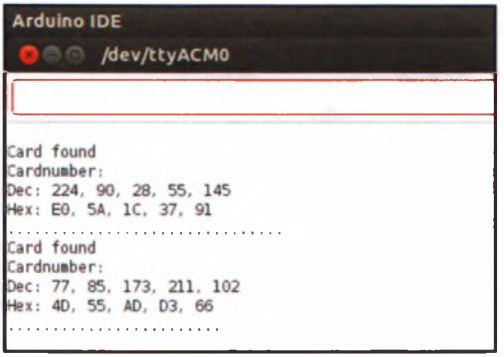


РИСУНОК 23.4

Номер RFID-метки отображается в шестнадцатеричном виде

6. Установите два светодиода на макетную плату, подключив короткие ножки (катоды) к шине заземления, а длинные (аноды) — через резистор с сопротивлением 220 Ом к контакту 3 (красный светодиод) и контакту 2 (зеленый светодиод) платы Arduino.

СВЕТОДИОДЫ	ARDUINO
Длинная ножка (анод) зеленого светодиода	Контакт 2 через резистор с сопротивлением 220 Ом
Длинная ножка (анод) красного светодиода	Контакт 3 через резистор с сопротивлением 220 Ом
Короткие ножки (катоды)	Контакт GND

7. Подключите сервопривод к Arduino: красный провод к шине питания, коричневый (или черный) — к шине заземления, а желтый — к контакту 9 платы Arduino.

СЕРВОПРИВОД	ARDUINO
Красный провод	Контакт 5V
Черный провод	Контакт GND
Желтый провод	Контакт 9

8. Подключите пьезоизлучатель к Arduino: красный провод к контакту 8, а черный — к контакту GND платы Arduino. Теперь цепь должна выглядеть примерно так, как показано на рис. 23.5.

ПЬЕЗОИЗЛУЧАТЕЛЬ	ARDUINO
Красный провод	Контакт 8
Черный провод	Контакт GND

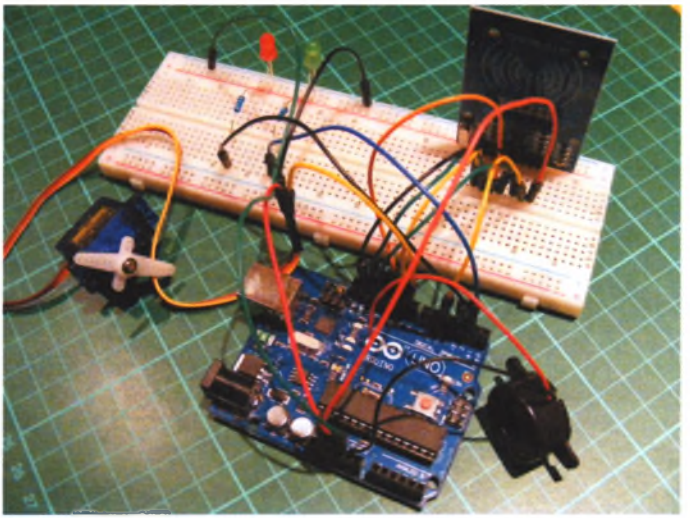


РИСУНОК 23.5

Законченный проект RFID-системы

9. Откройте код проекта в среде разработки Arduino и измените следующую строку, чтобы она соответствовала шестнадцатеричному номеру, определенному с помощью RFID-модуля для вашей карты или брелока на шаге 5. Оставьте символы 0x без изменений и подставьте далее свое значение.

```
byte card[5] = {0x4D,0x55,0xAD,0xD3,0x66};
```

10. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 23.6, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

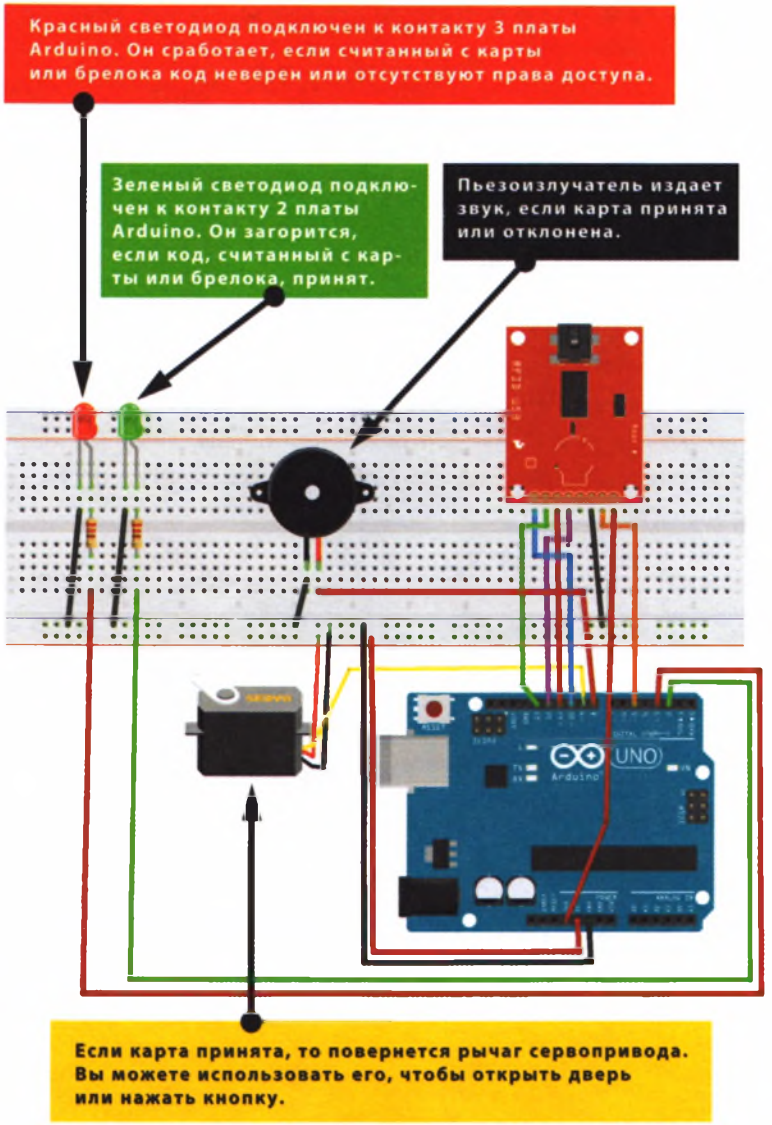


РИСУНОК 23.6

Принципиальная схема цепи беспроводной системы ввода ID-карты

СКЕТЧ

Скетч начинается с вызова библиотек SPI, RFID, Servo, Pitches и Wire для обеспечения обмена данными между Arduino, RFID-модулем и сервоприводом. Установлены две мелодии, одна проигрывается при успешном считывании карты, другая — при неудачном. Зеленый светодиод подключен к контакту 2

платы Arduino, красный — к контакту 3, пьезоизлучатель — к контакту 8, а сервопривод — к контакту 9.

В показанной ниже строке нужно добавить шестнадцатеричное значение вашей карты:

```
byte card[5] = {0x4D,0x55,0xAD,0xD3,0x66};
```

Проведите вашей картой перед RFID-модулем. Если шестнадцатеричный код на карте совпадет со значением в коде скетча, загорится зеленый светодиод, зазвучит сигнал успеха, и сервопривод начнет двигаться. RFID-модуль отвергнет любые другие карты, если их номера не добавлены в код в строке ❶. Если карта отвергнута, загорится красный светодиод и зазвучит другой сигнал, а сервопривод останется в исходном положении.

```
#include <SPI.h>
#include <RFID.h>
#include <Servo.h>
#include "pitches.h"
#include <Wire.h>

RFID rfid(10, 5);          // Определение RFID-метки

// Замените строку ниже на код вашей карты в шестнадцатеричном виде
❶ byte card[5] = {0x4D,0x55,0xAD,0xD3,0x66};
// Перечислите здесь все коды дополнительных карт доступа

byte serNum[5];
byte data[5];

// Определение мелодий для успешной авторизации и отказа
int access_melody[] = {NOTE_G4, 0, NOTE_A4, 0, NOTE_B4, 0, NOTE_A4,
0, NOTE_B4, 0, NOTE_C5, 0};
int access_noteDurations[] = {8, 8, 8, 8, 8, 4, 8, 8, 8, 8, 8, 4};
int fail_melody[] = {NOTE_G2, 0, NOTE_F2, 0, NOTE_D2, 0};
int fail_noteDurations[] = {8, 8, 8, 8, 8, 4};

int LED_access = 2;        // Контакт, к которому подключен зеленый
                           // светодиод
int LED_intruder = 3;      // Контакт, к которому подключен красный
                           // светодиод
int speaker_pin = 8;       // Контакт, к которому подключен
                           // пьезоизлучатель
int servoPin = 9;          // Контакт, к которому подключен
                           // сервопривод

Servo doorLock;            // Определение сервопривода

void setup() {
  doorLock.attach(servoPin);       // Назначение контакта сервопривода
  Serial.begin(9600);              // Начало последовательной связи
```

```
  SPI.begin();                          // Начало последовательной связи
                                        // между RFID-модулем и компьютером
  rfid.init();                          // Инициализация RFID-модуля
  Serial.println("Arduino card reader");
  delay(1000);
  pinMode(LED_access, OUTPUT);
  pinMode(LED_intruder, OUTPUT);
  pinMode(speaker_pin, OUTPUT);
  pinMode(servoPin, OUTPUT);
}

void loop() {          // Создание переменной для каждого пользователя
  boolean card_card = true;          // Определение вашей карты
  if (rfid.isCard()) {
    if (rfid.readCardSerial()) {
      delay(1000);
      data[0] = rfid.serNum[0];
      data[1] = rfid.serNum[1];
      data[2] = rfid.serNum[2];
      data[3] = rfid.serNum[3];
      data[4] = rfid.serNum[4];
    }
    Serial.print("Card found - code:");
    for (int i = 0; i < 5; i++) {
      // Если это не ваша карта, она считается ложной
      if (data[i] != card[i]) card_card = false;
    }
    Serial.println();
    if (card_card) {        // Обнаружение карты с правами доступа
      Serial.println("Hello!");          // Вывод на монитор порта
      for (int i = 0; i < 12; i++) {     // Воспроизведение
      приветственной мелодии
        int access_noteDuration = 1000 / access_noteDurations[i];
        tone(speaker_pin, access_melody[i], access_noteDuration);
        int access_pauseBetweenNotes = access_noteDuration * 1.30;
        delay(access_pauseBetweenNotes);
        noTone(speaker_pin);
      }
    }
    else {   // Если карта не распознана
      // Вывод сообщения на монитор порта
      Serial.println("Card not recognized! Contact administrator!");
      digitalWrite(LED_intruder, HIGH); // Включение красного
      светодиода
      for (int i = 0; i < 6; i++) {      // Воспроизведение мелодии
                                         // отказа
        int fail_noteDuration = 1000 / fail_noteDurations[i];
        tone(speaker_pin, fail_melody[i], fail_noteDuration);
        int fail_pauseBetweenNotes = fail_noteDuration * 1.30;
        delay(fail_pauseBetweenNotes);
        noTone(speaker_pin);
      }
      delay(1000);
      digitalWrite(LED_intruder, LOW);   // Выключение красного
                                         // светодиода
    }
    if (card_card) {        // Добавление других пользователей
                            // с правом доступа
      Serial.println("Access granted.......Welcome!");
      digitalWrite(LED_access, HIGH);    // Включение зеленого
```

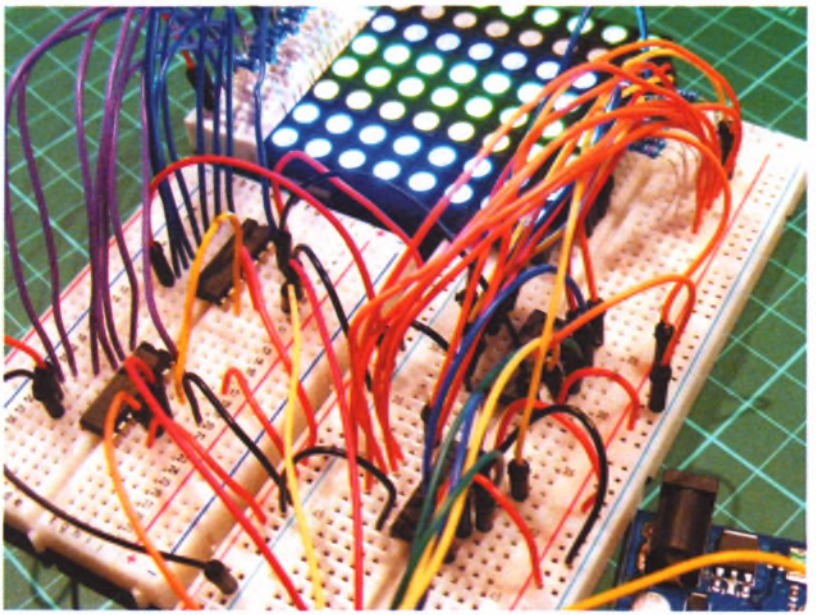
```
                                                // светодиода
        doorLock.write(180);                    // Поворот сервопривода
                                                // на 180 градусов
        delay(5000);                            // Задержка 5 секунд
        doorLock.write(0);                      // Поворот сервопривода
                                                // обратно до 0 градусов
        digitalWrite(LED_access, LOW);          // Выключение зеленого
                                                // светодиода
    }
    Serial.println();
    delay(500);
    rfid.halt();
  }
}
```

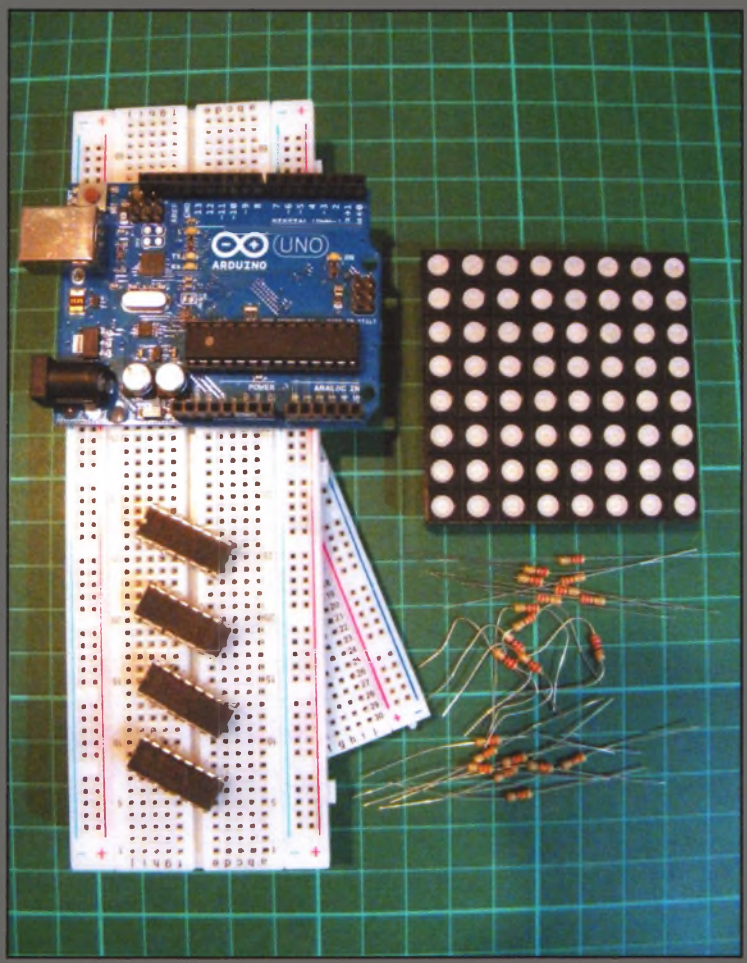
ЧАСТЬ 7

ПРОДВИНУТЫЕ ПРОЕКТЫ

ПРОЕКТ 24: РАЗНОЦВЕТНОЕ СВЕТОВОЕ ШОУ

В ЭТОМ ПРОЕКТЕ МЫ УСТРОИМ РАЗНОЦВЕТНОЕ СВЕТОВОЕ ШОУ С ПОМОЩЬЮ СВЕТОДИОДНОЙ RGB-МАТРИЦЫ РАЗМЕРОМ 8×8. МЫ ТАКЖЕ ИСПОЛЬЗУЕМ СДВИГОВЫЕ РЕГИСТРЫ, ЧТОБЫ РАСШИРИТЬ ВОЗМОЖНОСТИ ARDUINO И УПРАВЛЯТЬ МАТРИЦЕЙ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Плата Arduino
- 4 сдвиговых регистра
- 2 макетные платы
- Перемычки
- Светодиодная RGB-матрица размером 8×8
- 4 сдвиговых регистра 74HC595
- 16 резисторов с сопротивлением 220 Ом
- 8 резисторов с сопротивлением 330 Ом

ПРИНЦИП РАБОТЫ

Светодиодная RGB-матрица (рис. 24.1) представляет собой сетку из 64 красных, зеленых и синих светодиодов. Вы можете создавать разные оттенки, по отдельности управляя каждым светодиодом и смешивая цвета.

РИСУНОК 24.1

Светодиодная RGB-матрица

Светодиодная матрица оборудована в общей сложности 32 контактами (рис. 24.2). 8 контактов представляют собой общий анод каждого светодиода, и еще 24 контакта управляют свечением красного, зеленого и синего светодиода (по 8 контактов на цвет). В матрице, которую использовал я, контакты 17–20 и 29–32 — это аноды, контакты 9–16 управляют свечением красных светодиодов, 21–28 — зеленых и 1–8 — синих, но у вашей матрицы могут быть другие контакты. Контакт 1 — это нижняя левая ножка на рис. 24.2, остальные номера контактов на этом рисунке отсчитываются по порядку по часовой стрелке.

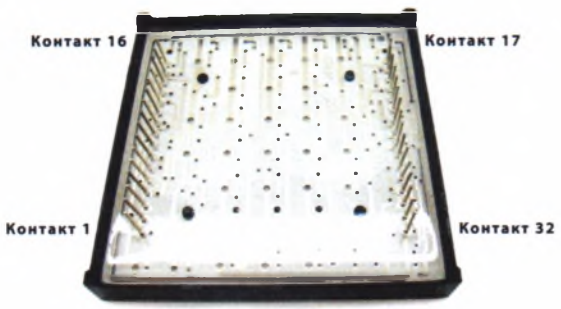


РИСУНОК 24.2

Контакты светодиодной RGB-матрицы

К любой матрице прилагается файл спецификации, в котором указано, какие контакты управляют красными, зелеными и синими светодиодами. Если номера контактов в вашем файле спецификации отличаются от номеров, указанных в табл. 24.1, учитывайте эти поправки при подключении сдвиговых регистров и Arduino. Каждый контакт управления светодиодом требует установки резистора,

чтобы предотвратить перегрузку и выгорание, причем требуются разные резисторы: с сопротивлением 220 Ом для синих и зеленых светодиодов и 330 Ом — для красных.

ТАБЛИЦА 24.1

Конфигурация контактов светодиодной RGB-матрицы

ФУНКЦИЯ МАТРИЦЫ	НОМЕР КОНТАКТОВ МАТРИЦЫ
Общий анод (+)	17, 18, 19, 20, 29, 30, 31, 32
Красные светодиоды	9, 10, 11, 12, 13, 14, 15, 16
Зеленые светодиоды	21, 22, 23, 24, 25, 26, 27, 28
Синие светодиоды	1, 2, 3, 4, 5, 6, 7, 8

Расположение нужных контактов может напугать, но не волнуйтесь. Просто выполняйте этот проект шаг за шагом.

Требуется столь много подключений, что контактов платы Arduino не хватит. Поэтому мы расширим ее возможности с помощью *сдвиговых регистров*. (Подобно тому, как мы делали это в проекте 16.) В этом проекте используется сдвиговый регистр 74HC595 для одновременного управления восемью выходами, при этом занимая только три контакта платы Arduino. Мы подключим несколько регистров вместе, чтобы управлять еще большим количеством контактов одновременно, используя один контакт для общего анода и по одному контакту для светодиодов каждого цвета.

Расположение контактов сдвигового регистра показано на рис. 24.3, а их предназначения описаны в табл. 24.2. При работе над проектом мы будем ссылаться на номер контакта сдвигового регистра и его предназначение, чтобы вам легче было определить, какой контакт вы подключаете.

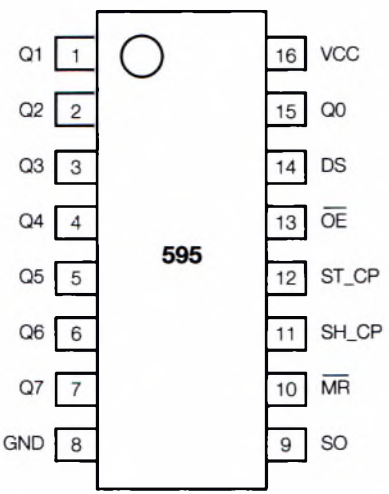


РИСУНОК 24.3

Расположение контактов сдвигового регистра

ТАБЛИЦА 24.2

Контакты сдвигового регистра 74HC595

КОНТАКТ	ОБОЗНАЧЕНИЕ	ПРЕДНАЗНАЧЕНИЕ
Контакты 1–7, 15	Q0–Q7	Параллельные выходы
Контакты 8	GND	Заземление, VSS
Контакты 9	SO	Выход для последовательного соединения регистров
Контакты 10	MR	Сброс значений регистра при получении сигнала LOW
Контакты 11	SH_CP	Вход для тактовых импульсов (контакт CLOCK)
Контакты 12	ST_CP	Синхронизация выходов (контакт LATCH)
Контакты 13	OE	Вход для переключения состояния выходов из высокоомного в рабочее
Контакты 14	DS	Вход для последовательных данных (контакт DATA)
Контакты 16	VCC	Питание

СБОРКА

1. Установите светодиодную RGB-матрицу 8×8 на две полноразмерные макетные платы, расположенные параллельно.
2. Подключите резисторы с сопротивлением 330 Ом к контактам красных светодиодов, а с сопротивлением 220 Ом — к контактам зеленых и синих светодиодов.
3. Установите один сдвиговый регистр на одну из макетных плат рядом с контактами общего анода светодиодной матрицы. Установите микросхему так, чтобы она перекрывала канавку макетной платы, как показано на рис. 24.4. Подключите контакты общего анода светодиодной матрицы к первому сдвиговому регистру в соответствии с таблицей ниже. Эти контакты не нуждаются в резисторах.

КОНТАКТЫ ОБЩЕГО АНОДА		КОНТАКТЫ СДВИГОВОГО РЕГИСТРА 1	
СВЕТОДИОДНАЯ МАТРИЦА	СДВИГОВЫЙ РЕГИСТР	СДВИГОВЫЙ РЕГИСТР	ARDUINO
32	15: Q0	8: GND	Контакт GND
31	1: Q1	9: SO	Сдвиг 3 DS
30	2: Q2	10: MR	Контакт 5V

КОНТАКТЫ ОБЩЕГО АНОДА		КОНТАКТЫ СДВИГОВОГО РЕГИСТРА 1	
СВЕТОДИОДНАЯ МАТРИЦА	СДВИГОВЫЙ РЕГИСТР	СДВИГОВЫЙ РЕГИСТР	ARDUINO
29	3: Q3	11: SH-CP	Контакт 13
20	4: Q4	12: ST-CP	Контакт 10
19	5: Q5	13: OE	Контакт GND
18	6: Q6	14: DS	Сдвиг 2 SO
17	7: Q7	16: VCC	Контакт 5V

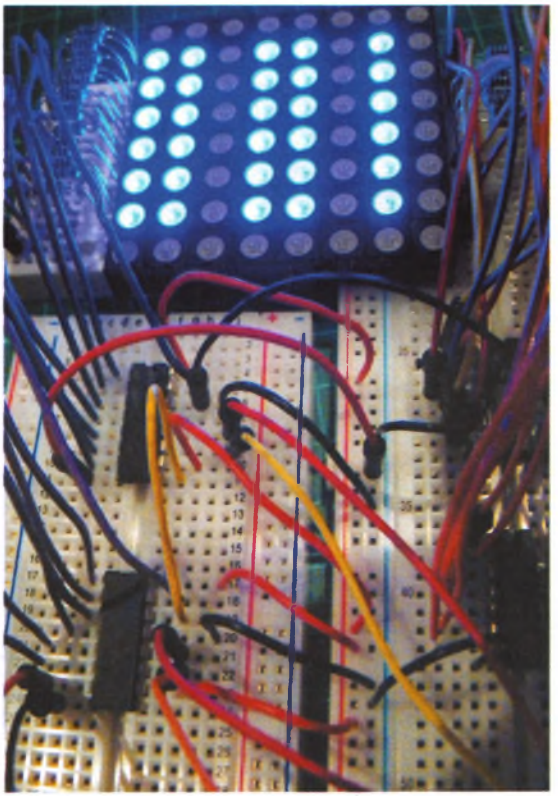


РИСУНОК 24.4

Сдвиговые регистры должны перекрывать канавку на макетной плате

4. Теперь установите оставшиеся три сдвиговых регистра на макетную плату. Второй сдвиговый регистр управляет зелеными светодиодами, третий — синими, а четвертый — красными. Подключите контакты каждого сдвигового регистра в соответствии с таблицами ниже. Все контакты цветных светодиодов должны подключаться через резисторы.

КОНТАКТЫ ЗЕЛЕНОГО СВЕТОДИОДА		КОНТАКТЫ СДВИГОВОГО РЕГИСТРА 2	
СВЕТОДИОДНАЯ МАТРИЦА	СДВИГОВЫЙ РЕГИСТР	СДВИГОВЫЙ РЕГИСТР	ARDUINO
28	15: Q0	8: GND	Контакт GND
27	1: Q1	9: SO	Сдвиг 1 DS
26	2: Q2	10: MR	Контакт 5V
25	3: Q3	11: SH-CP	Контакт 13
24	4: Q4	12: ST-CP	Контакт 10
23	5: Q5	13: OE	Контакт GND
22	6: Q6	14: DS	Контакт 11
21	7: Q7	16: VCC	Контакт 5V

КОНТАКТЫ СИНЕГО СВЕТОДИОДА		КОНТАКТЫ СДВИГОВОГО РЕГИСТРА 3	
СВЕТОДИОДНАЯ МАТРИЦА	СДВИГОВЫЙ РЕГИСТР	СДВИГОВЫЙ РЕГИСТР	ARDUINO
1	15: Q0	8: GND	Контакт GND
2	1: Q1	9: SO	Сдвиг 4 DS
3	2: Q2	10: MR	Контакт 5V
4	3: Q3	11: SH-CP	Контакт 13
5	4: Q4	12: ST-CP	Контакт 10
6	5: Q5	13: OE	Контакт GND
7	6: Q6	14: DS	Сдвиг 1 SO
8	7: Q7	16: VCC	Контакт 5V

КОНТАКТЫ КРАСНОГО СВЕТОДИОДА		КОНТАКТЫ СДВИГОВОГО РЕГИСТРА 4	
СВЕТОДИОДНАЯ МАТРИЦА	СДВИГОВЫЙ РЕГИСТР	СДВИГОВЫЙ РЕГИСТР	ARDUINO
9	15: Q0	8: GND	Контакт GND
10	1: Q1	9: SO	Сдвиг 3 DS
11	2: Q2	10: MR	Контакт 5V
12	3: Q3	11: SH-CP	Контакт 13
13	4: Q4	12: ST-CP	Контакт 10
14	5: Q5	13: OE	Контакт GND
15	6: Q6	14: DS	Сдвиг 2 SO
16	7: Q7	16: VCC	Контакт 5V

5. Микроконтроллер Arduino управляет светодиодами через три контакта ШИМ, по одному для тактовых импульсов, данных и защелки. Каждый контакт подключается к Arduino следующим образом:

СДВИГОВЫЙ РЕГИСТР	ARDUINO	ПРЕДНАЗНАЧЕНИЕ
Контакт 9 (регистр 2)	Контакт 11	Данные
Контакт 12 (все регистры)	Контакт 10	Защелка
Контакт 11 (все регистры)	Контакт 13	Такты

6. Убедитесь, что ваша цепь соответствует схеме, показанной на рис. 24.5, а затем загрузите в память Arduino код скетча, приведенный в разделе «Скетч» далее в этом проекте.

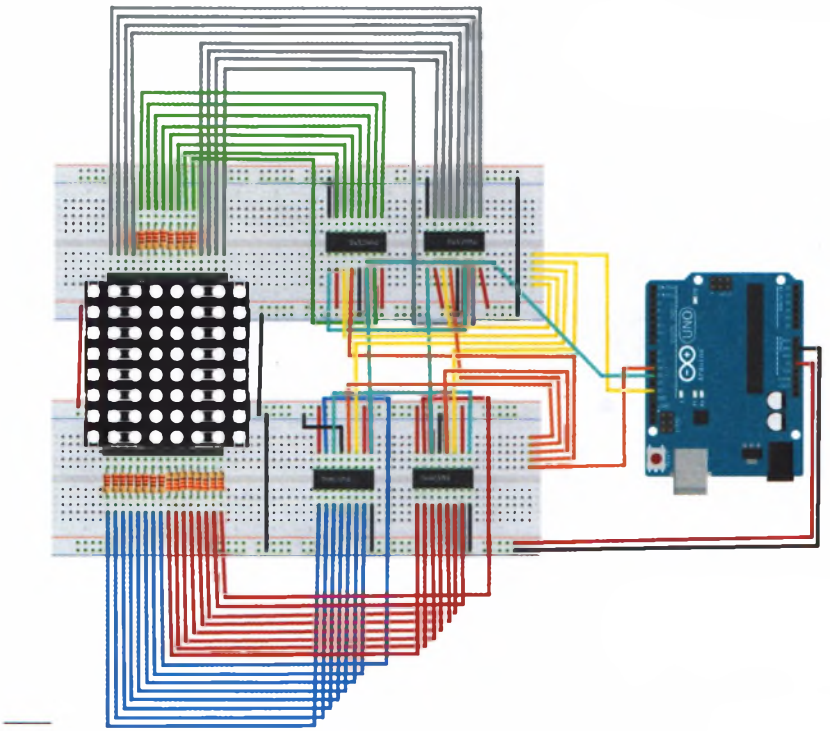


РИСУНОК 24.5

Принципиальная схема цепи устройства светового шоу

СКЕТЧ

Сначала в коде скетча определяются три контакта платы Arduino, управляющие сдвиговыми регистрами. Контакт защелки определен как контакт 10 платы Arduino, контакт тактовых импульсов — как 13, а контакт данных — как 11. Мы определяем

переменные от 0 до 255 для управления яркостью свечения каждого светодиода. Затем скетч по очереди включает каждый светодиод и смешивает три цвета для создания оттенка. Например, при включенном зеленом, выключенном синем и включенном красном светодиодах отображается желтый цвет. Затем скетч зацикленно зажигает светодиоды в произвольном порядке.

```
/* Пример 18.1 — эксперименты со светодиодной RGB-матрицей
   Лицензия CC by-sa 3.0
   http://tronixstuff.wordpress.com/tutorials
*/

int latchpin = 10;    // Контакт, к которому подключены
                      // контакты 12 всех сдвиговых регистров
int clockpin = 13;    // Контакт, к которому подключены
                      // контакты 11 на всех сдвиговых регистров
int datapin = 11;     // Контакт, к которому подключен
                      // контакт 14 второго сдвигового регистра
int zz = 500;         // Переменная задержки
int va[] = {
   1, 2, 4, 8, 16, 32, 64, 128, 255
};
int va2[] = {
   1, 3, 7, 15, 31, 63, 127, 255
};

void setup() {
   pinMode(latchpin, OUTPUT);
   pinMode(clockpin, OUTPUT);
   pinMode(datapin, OUTPUT);
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 0);
   shiftOut(datapin, clockpin, MSBFIRST, 0);
   shiftOut(datapin, clockpin, MSBFIRST, 0);
   shiftOut(datapin, clockpin, MSBFIRST, 0);
   digitalWrite(latchpin, HIGH);
   randomSeed(analogRead(0));
}

void allRed() {              // Включение всех красных светодиодов
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 255);     // Включение катодов
                                                   // на полную мощность
   shiftOut(datapin, clockpin, MSBFIRST, 0);       // Включение
                                                   // зеленого на 0
   shiftOut(datapin, clockpin, MSBFIRST, 0);       // Включение
                                                   // синего на 0
   shiftOut(datapin, clockpin, MSBFIRST, 255);     // Включение красного
                                                   // на полную мощность
   digitalWrite(latchpin, HIGH);
}
```

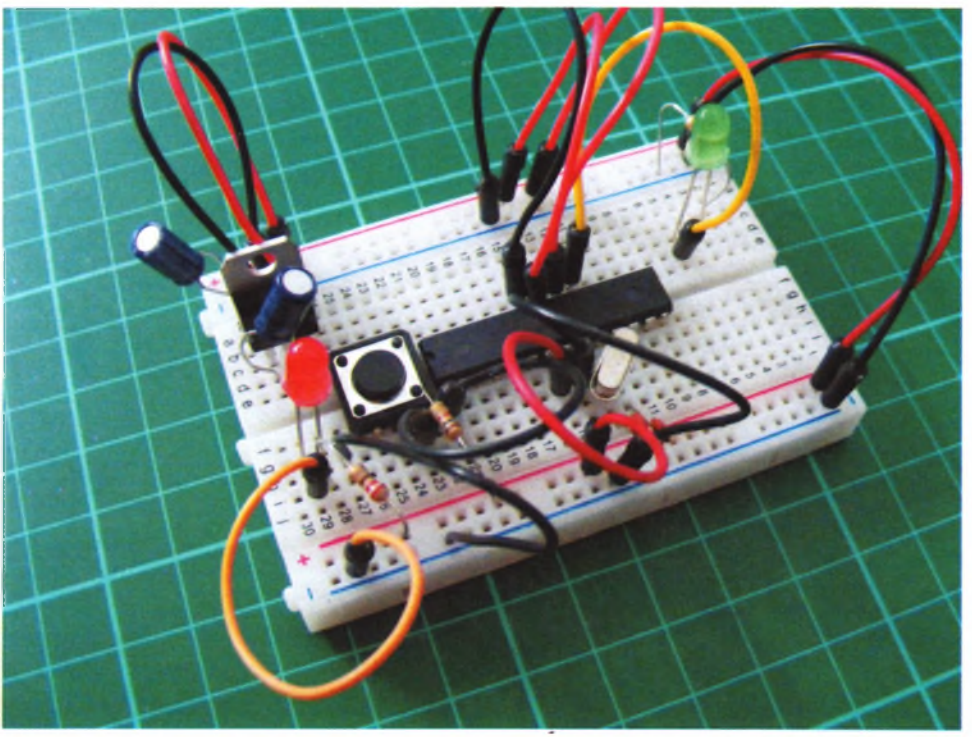
```
void allBlue() {                    // Включение всех синих светодиодов
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Синий
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Красный
   digitalWrite(latchpin, HIGH);
}

void allGreen() {            // Включение всех зеленых светодиодов
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Синий
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Красный
   digitalWrite(latchpin, HIGH);
}

void allOn() {               // Включение всех светодиодов
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Синий
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Красный
   digitalWrite(latchpin, HIGH);
}

void allYellow() {           // Включение зеленых и красных светодиодов
                             // (желтый)
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Синий
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Красный
   digitalWrite(latchpin, HIGH);
}

void allAqua() {      // Включение зеленых и синих светодиодов
                      // (голубой)
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Синий
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Красный
   digitalWrite(latchpin, HIGH);
}

void allPurple() {    // Включение синих и красных светодиодов
                      // (лиловый)
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Синий
   shiftOut(datapin, clockpin, MSBFIRST, 255);    // Красный
```

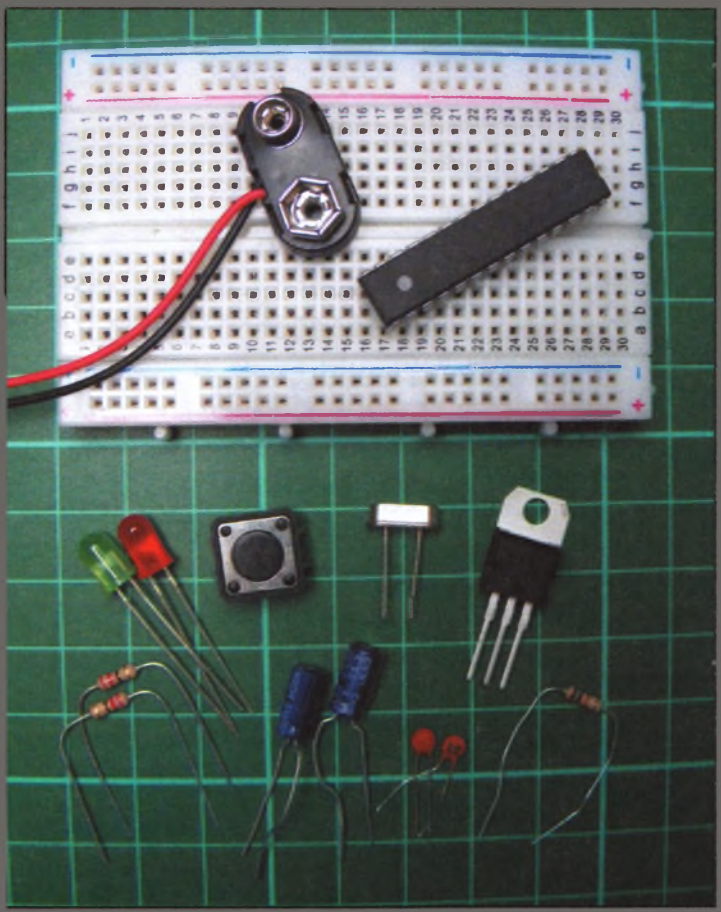
```
   digitalWrite(latchpin, HIGH);
}

void clearMatrix() {                    // Включение всех светодиодов
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Синий
   shiftOut(datapin, clockpin, MSBFIRST, 0);      // Красный
   digitalWrite(latchpin, HIGH);
}

void lostinspace() {          // Мигание светодиодами случайным образом
   for (int z = 0; z < 100; z++) {
      digitalWrite(latchpin, LOW);
      shiftOut(datapin, clockpin, MSBFIRST, va[random(8)]); // Катоды
      shiftOut(datapin, clockpin, MSBFIRST, va[random(8)]); // Зеленый
      shiftOut(datapin, clockpin, MSBFIRST, va[random(8)]); // Синий
      shiftOut(datapin, clockpin, MSBFIRST, va[random(8)]); // Красный
      digitalWrite(latchpin, HIGH);
      delay(100);
   }
}

void displayLEDs(int rr, int gg, int bb, int cc, int dd) {
// Добавление десятичных значений в функции shiftOut
// и не выключать матрицу в течение dd миллисекунд
   digitalWrite(latchpin, LOW);
   shiftOut(datapin, clockpin, MSBFIRST, cc);     // Катоды
   shiftOut(datapin, clockpin, MSBFIRST, gg);     // Зеленый
   shiftOut(datapin, clockpin, MSBFIRST, bb);     // Синий
   shiftOut(datapin, clockpin, MSBFIRST, rr);     // Красный
   digitalWrite(latchpin, HIGH);
   delay(dd);
}

void loop() {          // Включение всей матрицы сплошным цветом
   allOn();
   delay(zz);
   delay(zz);

   allRed();
   delay(zz);
   delay(zz);

   allGreen();
   delay(zz);
   delay(zz);

   allBlue();
   delay(zz);
   delay(zz);

   allPurple();
```

```
    delay(zz);
    delay(zz);

    allYellow();
    delay(zz);
    delay(zz);

    allAqua();
    delay(1000);
    // Включение некоторых отдельных светодиодов, используя случайные
    // значения
    lostinspace();     // Прокрутка некоторых горизонтальных
                       // и вертикальных линий
    for (int z = 0; z < 5; z++) {
       for (int q = 1; q < 129; q *= 2) {
          displayLEDs(255, 0, 0, q, 200);
       }
    }
    clearMatrix();
    delay(1000);

    for (int z = 0; z < 5; z++) {
       for (int q = 1; q < 129; q *= 2) {
          displayLEDs(0, 255, 0, q, 200);
          displayLEDs(q, 0, 0, 255, 200);
       }
    }
    clearMatrix();
    delay(1000);

    for (int z = 0; z < 5; z++) {
       for (int q = 1; q < 9; q++) {
          displayLEDs(0, 0, 255, va2[q], 200);
       }
    }
    clearMatrix();
    delay(1000);
}
```

ПРОЕКТ 25: СОБСТВЕННАЯ ПЛАТА ARDUINO!

В ЭТОМ ПРОЕКТЕ ВЫ СОБЕРЕТЕ АНАЛОГ ARDUINO, ИСПОЛЬЗУЯ МИНИМУМ КОМПОНЕНТОВ.





ТРЕБУЕМЫЕ КОМПОНЕНТЫ

- Микроконтроллер ATMEL ATmega328P
- Макетная плата
- Зеленый светодиод
- Красный светодиод
- 3 резистора с сопротивлением 220 Ом
- Кварцевый генератор с частотой 16 МГц (HC-495)
- 5-вольтовый стабилизатор напряжения L7805cv
- 2 электролитических конденсатора емкостью 100 мкФ
- Клемма для батареи напряжением 9 В (типа «Крона»)
- 2 дисковых конденсатора емкостью 22 пФ
- Тактовая четырехконтактная кнопка
- Батарея напряжением 9 В (типа «Крона»)

Эта самодельная недорогая плата может заменить оригинальную Arduino из магазина, потому что является ее полной аппаратной копией. Вы можете использовать собственную плату в постоянных проектах взамен более дорогой платы Arduino.

ПРИНЦИП РАБОТЫ

Плата из нашего проекта работает точно так же, как и Arduino. В ее основе используется микроконтроллер ATMEL ATmega328P (рис. 25.1), к которому мы подключим дополнительные компоненты. Микроконтроллер ATmega — это «мозг» Arduino, выполняющий инструкции из загруженного скетча.

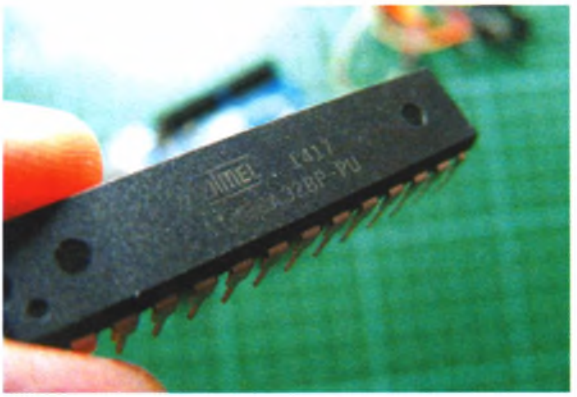


РИСУНОК 25.1

Микроконтроллер ATMEL ATmega328P

5-вольтовый стабилизатор напряжения L7805cv контролирует напряжение и снижает напряжение тока от 9-вольтовой батареи до 5 В, до уровня, на котором работает микросхема ATmega, тем самым защищая микроконтроллер и дополнительные компоненты. Кварцевый генератор с частотой 16 МГц (рис. 25.2) позволяет Arduino вычислять время, а конденсаторы действуют как фильтр для выравнивания напряжения.

РИСУНОК 25.2

Кварцевый генератор с частотой 16 МГц (HC-495)

В табл. 25.1 перечислены контакты микроконтроллера ATmega328P и их соответствие контактам Arduino. Например, контакт 13 платы Arduino, который мы использовали для тестирования Arduino в начале книги, соответствует контакту 19 на микроконтроллере ATMEL ATmega328P.

Распознать верхнюю часть микросхемы можно по небольшой полукруглой выемке (рис. 25.3). Контакт 1 находится ниже этой выемки, а затем контакты нумеруются от 1 до 28 *против* часовой стрелки.

ТАБЛИЦА 25.1

Контакты микроконтроллера ATmega и их соответствие контактам Arduino

КОНТАКТ ATMEGA	ФУНКЦИЯ ARDUINO	КОНТАКТ ATMEGA	ФУНКЦИЯ ARDUINO
1	Сброс	15	Контакт 9
2	Контакт 0	16	Контакт 10
3	Контакт 1	17	Контакт 11
4	Контакт 2	18	Контакт 12
5	Контакт 3	19	Контакт 13
6	Контакт 4	20	BCC
7	VCC	21	AREF
8	GND	22	GND
9	Кварц	23	A0
10	Кварц	24	A1
11	Контакт 5	25	A2
12	Контакт 6	26	A3
13	Контакт 7	27	A4
14	Контакт 8	28	A5

РИСУНОК 25.3

Наверху микроконтроллера есть полукруглая выемка

ПОДГОТОВКА МИКРОКОНТРОЛЛЕРА

Обязательно покупайте микроконтроллер ATmega с предустановленным загрузчиком Arduino, так как он также содержит предустановленный скетч мигающего светодиода, который вам понадобится для этого проекта.

У нашей поделки нет USB-разъема для подключения микросхемы непосредственно к компьютеру, поэтому, если вы хотите использовать самодельную плату Arduino с другим скетчем (или если в вашем микроконтроллере не был установлен загрузчик), вам необходимо взять полноценную плату Arduino и загрузить скетч на свой микроконтроллер ATmega следующим образом:

1. Осторожно снимите микроконтроллер Arduino ATmega с оригинальной платы Arduino (рис. 25.4) и замените его своим микроконтроллером ATmega.

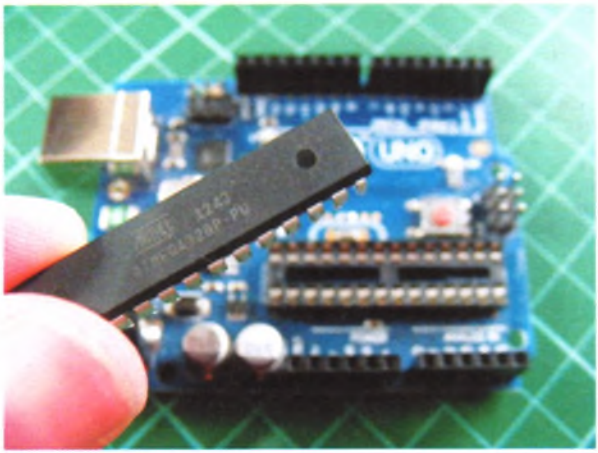


РИСУНОК 25.4

Снятие микроконтроллера ATmega с платы Arduino

2. Подключите Arduino к компьютеру с помощью USB-кабеля.
3. Откройте среду разработки Arduino на компьютере.
4. Загрузите скетч на микроконтроллер.
5. После загрузки скетча отключите Arduino от компьютера, аккуратно снимите ваш микроконтроллер с платы и замените его на оригинальную микросхему Arduino ATmega.

В новый микроконтроллер ATmega обязательно нужно загрузить нужный вам скетч. Как правило, самодельные платы Arduino используются в постоянных проектах, поэтому часто не требуется загружать новые скетчи; вы загружаете один скетч во время работы над проектом и используете его.

Теперь вы готовы к сборке собственной платы.

СБОРКА ARDUINO

Во всех проектах я приводил принципиальную схему в самом конце, но в этом случае полезно сначала взглянуть на нее, чтобы ссылаться на расположение и определять используемые компоненты (рис. 25.5).

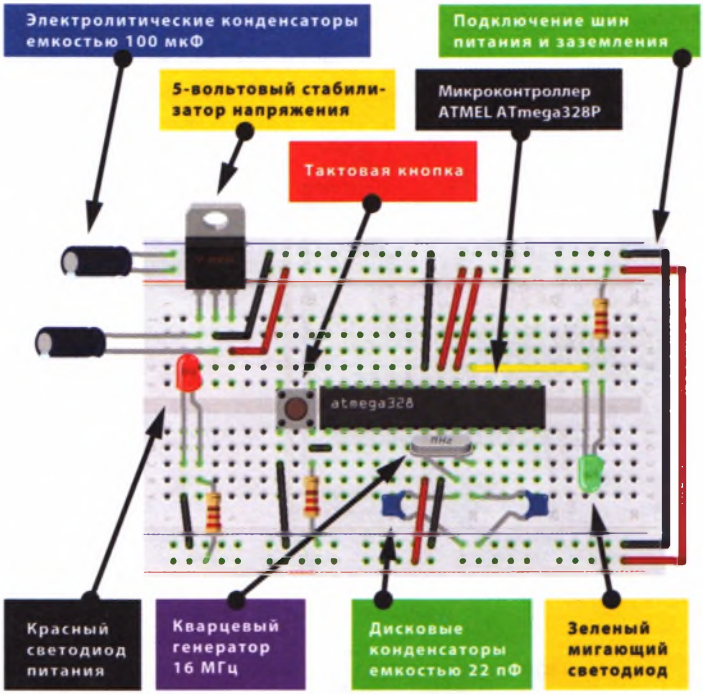


РИСУНОК 25.5

Полная схема

6. Установите микросхему ATmega на макетную плату, перекрыв канавку. Для размещения компонентов вам понадобится небольшое пространство с двух сторон, поэтому установите ее примерно так, как показано на рис. 25.6. Помните, что контакт 1 микроконтроллера ATmega328P находится рядом с небольшим полукруглым углублением на чипе. От него контакты отсчитываются по порядку, против часовой стрелки. Учитывайте это, чтобы правильно разместить свой микроконтроллер. Полукруглое углубление должно быть слева, как показано на схеме.

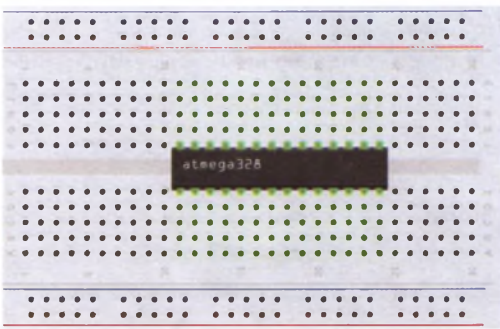


РИСУНОК 25.6

Расположение микросхемы ATmega таким образом, чтобы она перекрывала канавку макетной платы

7. Подключите контакты 7, 20 и 21 микроконтроллера ATmega к ближайшей шине питания макетной платы, а контакты 8 и 23 — к шине заземления. Используйте перемычки для соединения шин питания и заземления по обеим сторонам платы, как показано на рис. 25.7.

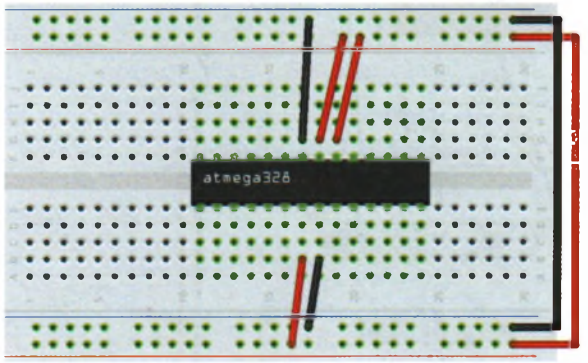


РИСУНОК 25.7

Подключение шин питания и заземления

8. Подключите одну ножку кварцевого генератора к контакту 9 микросхемы ATmega, а другую — к контакту 10. Подключите ножки одного дискового конденсатора емкостью 22 пФ к контактам 9 и GND, а ножки второго — к контактам 10 и GND, как показано на рис. 25.8.

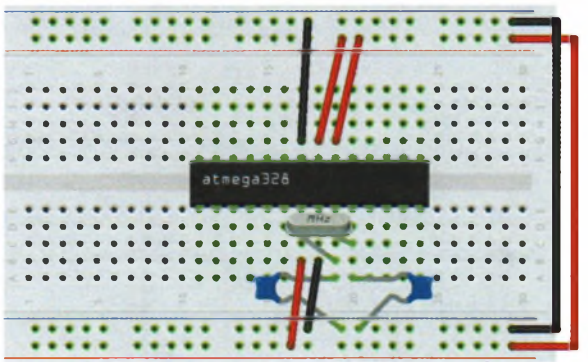


РИСУНОК 25.8

Установка кварцевого генератора и дисковых конденсаторов

9. Установите кнопку на макетную плату слева от микросхемы ATmega так, чтобы перекрыть канавку. Используя перемычки, подключите нижний правый контакт кнопки к контакту 1 микроконтроллера ATmega, а нижний левый контакт — к контакту GND, как показано на рис. 25.9. Подключите резистор с сопротивлением 220 Ом к нижнему правому контакту кнопки, а другой контакт этого резистора — к шине заземления. Эта кнопка будет выполнять функцию сброса.

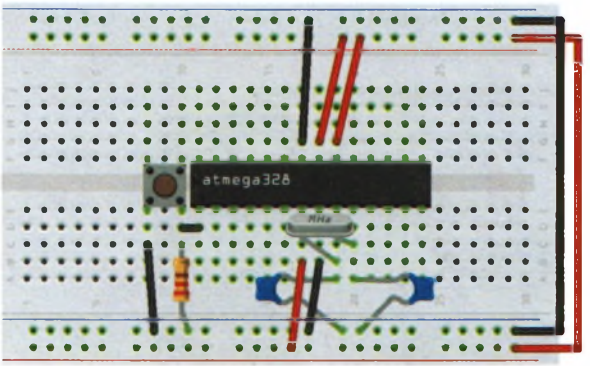


РИСУНОК 25.9

Установка кнопки сброса

10. Установите 5-вольтовый стабилизатор напряжения L7805cv в верхний левый угол макетной платы, чтобы его номер был обращен к вам, как показано на рис. 25.10 — его контакты 1–3 пронумерованы слева направо. Установите один электролитический конденсатор емкостью 100 мкФ в верхнюю часть макетной платы, одной ножкой в шину питания, а второй — в шину заземления. Подключите второй конденсатор емкостью 100 мкФ к контактам 1 и 2 стабилизатора напряжения. Затем подключите контакт 2 стабилизатора напряжения к шине заземления, а контакт 3 — к шине питания.

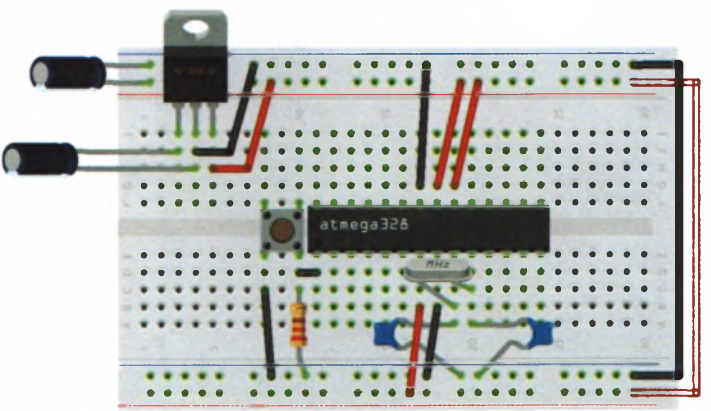


РИСУНОК 25.10

Подключение электролитических конденсаторов емкостью 100 мкФ и 5-вольтового стабилизатора напряжения L7805cv

11. Установите красный светодиод на макетную плату, подключив длинную ножку (анод) через резистор с сопротивлением 220 Ом к шине питания, короткую ножку (катод) — к шине заземления. Затем установите зеленый светодиод, подключив короткую ножку (катод) к контакту 21 микроконтроллера ATmega, а длинную (анод) через резистор с сопротивлением 220 Ом к шине питания, как показано на

рис. 25.11. Подключите контакт + батареи 9 В к контакту 1 стабилизатора напряжения, а контакт — соедините с контактом 2 стабилизатора напряжения.

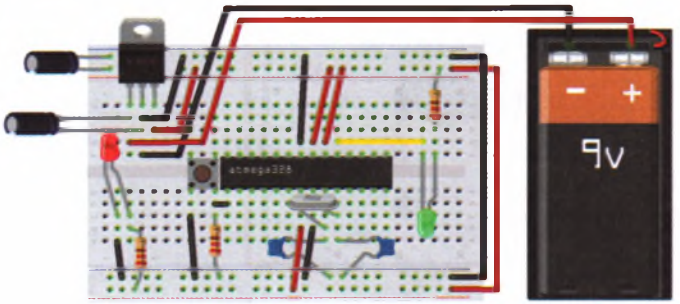


РИСУНОК 25.11

Установка светодиодов и подключение батареи 9 В

Ваша плата готова и должна выглядеть так, как показано на рис. 25.12. Красный светодиод загорается, когда на шины макетной платы подается питание, оповещая, что Arduino включен и работает, а зеленый светодиод загорается, реагируя на скетч «Мигающий светодиод», загруженный в микроконтроллер ATmega.

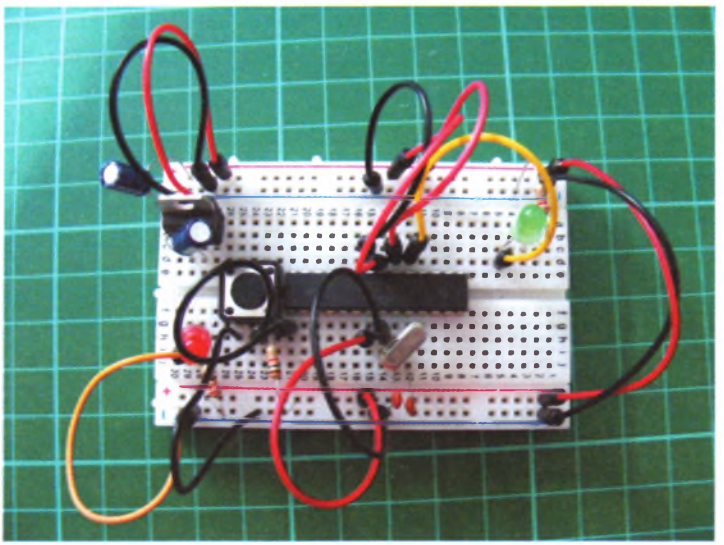


РИСУНОК 25.12

Завершенная цепь

Используя табл. 25.1 для информации, вы можете применять эту цепь так же, как и Arduino Uno, подключая компоненты к контактам микросхемы ATmega. Если вы хотите собрать какой-либо проект из этой книги на постоянной основе, предварительно соберите свою плату Arduino, чтобы сэкономить деньги! Не забудьте при сборке загрузить скетч в микросхему ATmega с помощью оригинальной платы Arduino.

ПРИЛОЖЕНИЕ А: КОМПОНЕНТЫ

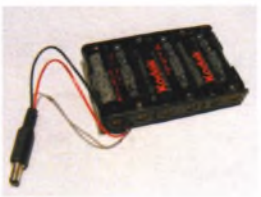
ЭТО ПРИЛОЖЕНИЕ СОДЕРЖИТ ДОПОЛНИТЕЛЬНУЮ ИНФОРМАЦИЮ О КОМПОНЕНТАХ, ИСПОЛЬЗУЕМЫХ В ПРОЕКТАХ ЭТОЙ КНИГИ. ОПИСАНИЕ КАЖДОГО КОМПОНЕНТА СОПРОВОЖДАЕТСЯ ФОТОГРАФИЕЙ И НЕКОТОРЫМИ СВЕДЕНИЯМИ. ТАКЖЕ Я ПРИВЕЛ СПИСОК МАГАЗИНОВ, В КОТОРЫХ МОЖНО ПРИОБРЕСТИ УКАЗАННЫЕ ДЕТАЛИ. А В САМОМ КОНЦЕ ВЫ РАЗБЕРЕТЕСЬ С МАРКИРОВКОЙ РЕЗИСТОРОВ.

РУКОВОДСТВО ПО КОМПОНЕНТАМ

Ниже приведено руководство по компонентам, которые используются в проектах, а также полезная дополнительная информация. Компоненты перечислены в том порядке, в котором они используются в книге. Большинство компонентов можно купить в интернет-магазинах, таких, как «Чип и Дип» или eBay, а некоторые у специализированных розничных продавцов, список которых приведен в конце приложения.

Батарейный отсек, рассчитанный на напряжение 9 В

Батарейный отсек, рассчитанный на напряжение 9 В, подключается к плате Arduino для подачи электропитания. Вы вставляете батарейки и подключаете штекер к гнезду на плате Arduino, как описано в разделе «Питание» в проекте 0. Обратите внимание, что плату Arduino также можно запитать через USB-кабель.



- Необходимое количество: 1
- Количество контактов: 1
- Используется в проектах: Все (опционально)

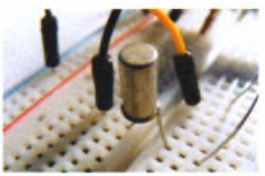
Датчик влажности почвы HL-69

Этот датчик измеряет влажность почвы. Он оборудован двумя штырями и двумя контактами (сверху). В проектах этой книги используется датчик влажности почвы HL-69. Он поставляется с модулем драйвера, который вы подключаете к своему Arduino, вместо того, чтобы напрямую подключаться к датчику.

- Необходимое количество: 1
- Количество контактов: 2
- Используется в проектах: 5

Датчик наклона

Датчик наклона представляет собой капсулу с металлическим шариком внутри, который замыкает цепь, когда капсула находится в вертикальном положении. Наклоните ее, и цепь разорвется.



- Необходимое количество: 1
- Количество контактов: 2
- Используется в проектах: 14

Датчик температуры и влажности DHT11

Датчик DHT11 измеряет влажность и температуру. Это небольшая синяя или белая пластиковая коробочка с четырьмя контактами, которая иногда сразу монтируется на плате модуля с тремя контактами. В этой книге используется датчик DHT11 с модулем и тремя контактами: 5V, DATA и GND.

- Необходимое количество: 1
- Количество контактов: 4 (но мы используем только 3)
- Используется в проектах: 13

Джойстик

Джойстик имеет аналоговый вход, изменение которого джойстик выдает на цифровом выходе. Джойстик состоит из двух потенциометров, подающих сигнал по двум осям: влево/вправо и вверх/вниз. Джойстики используются в самых разных проектах, таких, как игры или управление сервоприводами.

- Необходимое количество: 1
- Количество контактов: 5
- Используется в проектах: 10

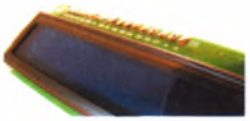
Дисковый конденсатор

Дисковый конденсатор также может накапливать заряд, но это другой тип конденсатора (в данном случае, емкостью 22 пФ). Он выглядит как маленький диск с двумя ножками. Номинал обычно указан на лицевой стороне. Существуют различные типы дисковых конденсаторов; на рисунке показан керамический вариант.

- Необходимое количество: 2
- Количество контактов: 2
- Используется в проектах: 25

ЖК-дисплей

Жидкокристаллический дисплей служит для вывода символов. Дисплеи бывают разных размеров. В книге используется модель HD44780 (16 символов × 2 строки) с 16 контактами. ЖК-дисплей состоит из двух листов поляризующего материала с жидкокристаллическим раствором между ними; ток, проходящий через кристалл, создает изображение.



- Необходимое количество: 1
- Количество контактов: 16
- Используется в проектах: 12, 13, 14, 15

Инфракрасный датчик

Инфракрасный (ИК) датчик фиксирует инфракрасные сигналы, поступающие, например, с пульта дистанционного управления. ИК-датчик представляет собой светодиод в небольшом корпусе с тремя ножками: OUT (данные), GND и 5V. При подключении ИК-датчика необходимо соблюдать полярность. Проверьте файл спецификации к вашему датчику на тот случай, если контакты отличаются.

- Необходимое количество: 1
- Количество контактов: 3
- Используется в проектах: 11

Кварцевый генератор 16 МГц

Кварцевый генератор с частотой 16 МГц позволяет Arduino вычислять время. Это небольшой металлический компонент с двумя ножками, причем к каждой из них нужно подключать конденсатор, чтобы выровнять скачки напряжения. Частота генератора обозначена на лицевой стороне.

- Необходимое количество: 1
- Количество контактов: 2
- Используется в проектах: 25

Клавиатура

Клавиатура размером 4×4, используемая в книге, представляет собой, по сути, ряд переключателей. Она содержит 16 кнопок, объединенных в ряды. Также существует 12-кнопочная модель. Из восьми контактов четыре управляют строками и еще четыре — столбцами. Плата Arduino повторяет номер каждой нажатой кнопки.

- Необходимое количество: 1
- Количество контактов: 8
- Используется в проектах: 22

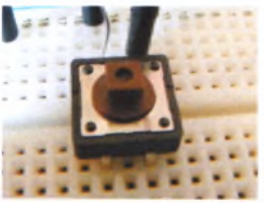
Клеммы для батареи

Клеммы для батареи 9 В (типа «Крона») представлены в виде небольшого черного разъема, который имеет два контакта и провода: черный для заземления (–) и красный для питания (+).

- Необходимое количество: 1
- Количество контактов: 2
- Используется в проектах: 25

Кнопка

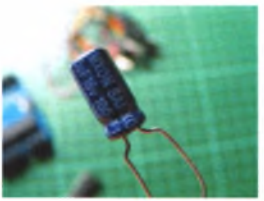
Кнопка представляет собой простой переключатель, который замыкает цепь при нажатии и размыкает при отпускании. Ее также называют мгновенным переключателем. Кнопки бывают различных размеров и большинство из них имеют четыре контакта.



- Необходимое количество: 4
- Количество контактов: 4
- Используется в проектах: 2, 8, 15, 16, 17, 25

Конденсатор

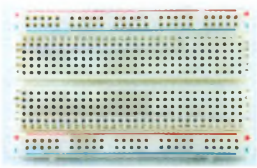
Конденсаторы могут накапливать небольшое количество заряда. Они часто применяются для стабилизации выходного напряжения и тока. Они выглядят как маленькие цилиндры с двумя ножками, а их номинал обычно указывается на боковой стороне. Конденсаторы зависимы от полярности и должны устанавливаться правильно. Длинная ножка подключается к +, а короткая к –; обычно ножки помечаются символами на цилиндре. Существуют различные типы конденсаторов; на рисунке изображен алюминиевый электролитический конденсатор емкостью 100 мкФ.



- Необходимое количество: 2
- Количество контактов: 2
- Используется в проектах: 25

Макетная плата

Макетная плата используется для подключения компонентов и создания цепей, описанных в проектах. См. раздел «Макетные платы» в проекте 0 для получения дополнительной информации.



- Необходимое количество: 2 полноразмерные платы, 1 полуразмерная плата, 1 мини-плата
- Количество контактов: 940 на полноразмерной плате, 420 на полуразмерной плате, 170 на мини-плате
- Используется в проектах: Все, за исключением проекта 7

Микроконтроллер ATmega328P

Микроконтроллер ATMEL ATmega328P — это мозг Arduino; именно он выполняет инструкции из загруженного скетча. Это маленький черный микроконтроллер с 32 ножками. На одном его конце вы увидите точку или полукруг — контакт 1 находится слева от этой метки.

- Необходимое количество: 1
- Количество контактов: 32
- Используется в проектах: 25

Пассивный инфракрасный датчик движения

Датчик движения (пассивный инфракрасный) обнаруживает движение в пределах определенной области. В книге используется наиболее распространенная модель HC SR501. Датчик оборудован линзой на передней панели и тремя контактами: 5V, OUTPUT и GND. Оранжевые компоненты на плате датчика — это потенциометры, которые изменяют чувствительность датчика и время активного состояния.

- Необходимое количество: 1
- Количество контактов: 3
- Используется в проектах: 21

Плата Arduino Uno R3

Arduino Uno R3 — основной компонент и «мозг» для всех наших проектов.

- Необходимое количество: 1
- Количество контактов: 14
- Используется в проектах: Все

Потенциометр

Потенциометр — это переменный резистор, сопротивлением которого вы можете управлять, изменяя проходящее через него напряжение. Он оборудован ручкой, которую вы можете поворачивать, изменяя сопротивление. У потенциометра три контакта снизу. Центральный контакт служит для передачи сигнала управления и влияет на каждую сторону (неважно, как подключены два других контакта). Потенциометр обычно используется для управления выходом, таким как уровень громкости радио.

- Необходимое количество: 1 Потенциометр с сопротивлением 50 кОм
- Количество контактов: 3
- Используется в проектах: 2, 3, 4, 12, 13, 14, 15, 17

Пьезоизлучатель

Пьезоизлучатель — это примитивный динамик. Электрические импульсы заставляют его очень быстро щелкать, создавая звуковой тон. Пьезоизлучатель часто выглядит как маленький черный ящик с двумя проводами. Если его извлечь из корпуса, вы увидите маленький позолоченный диск. Это дешевый компонент, часто используемый в недорогих игрушках для генерации звука (например, в сиренах игрушечных машинок). Он также может использоваться в качестве датчика звука, как это показано в проекте 9.

- Необходимое количество: 1
- Количество контактов: 2
- Используется в проектах: 5, 7, 8, 9, 15, 17, 18, 19, 21, 23

Ракетная установка WLT-V959-19

Ракетная установка WLT-V959-19, предназначенная для квадрокоптеров, представляет собой мини-версию пулемета Гатлинга, способную быстро выпустить шесть пластиковых ракет. Установка оборудована четырьмя проводами, но для непрерывной стрельбы нам понадобятся только желтый и белый.

- Необходимое количество: 1
- Количество контактов: 4 (используется только 2)
- Используется в проектах: 20

Резистор

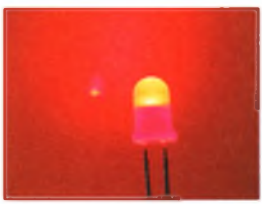
Резисторы ограничивают протекающий в цепи ток и предотвращают перегорание компонентов. Резисторы выглядят как цилиндры с цветными полосками и ножками с обоих

концов. Значение сопровления резистора определяется цветовым кодом — для получения дополнительной информации см. раздел «Расшифровка значений резисторов» в конце приложения. Проверяйте значения внимательно, иначе очень легко ошибиться. Резисторы могут содержать две, четыре или пять цветных полос, поэтому имейте в виду, что, например, четырехполосный резистор с сопротивлением 220 Ом может несколько отличаться от пятиполосного резистора с тем же номиналом.

- Необходимое количество: 30 с сопротивлением 220 Ом, 10 с сопротивлением 330 Ом, 1 с сопротивлением 10 кОм, 1 с сопротивлением 1 МОм
- Количество контактов: 2
- Используется в проектах: 1 — 4, 6, 8, 9, 16, 17, 18, 19, 22, 23, 24, 25

Светодиод

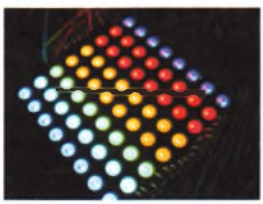
Светодиод излучает свет, когда на него подается небольшой ток. Светодиод похож на маленькую лампочку с двумя ножками. Длинная нога — это анод, предназначенный для подключения к источнику питания +. Светодиодам обычно требуется резистор, иначе они могут перегореть. Светодиоды зависят от полярности, то есть ток через них течет только в одном направлении.



- Необходимое количество: 40 (по 10 красных, синих, желтых и зеленых).
- Количество контактов: 2.
- Используется в проектах: 1 — 6, 8, 9, 17, 18, 19, 21, 22, 23, 25.

Светодиодная RGB-матрица

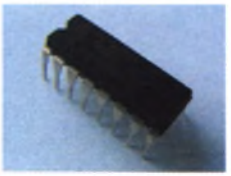
Светодиодная RGB-матрица размером 8×8 представляет собой панель из 64 светодиодов, которые могут светиться красным, зеленым или синим цветом, создавая различные оттенки. Матрица оборудована 32 контактами: 8 контактов для общего анода всех светодиодов и по 8 контактов для красных, зеленых и синих светодиодов. При подаче питания на катоды каждого светодиода необходимо использовать резисторы.



- Необходимое количество: 1
- Количество контактов: 32
- Используется в проектах: 24

Сдвиговый регистр

Сдвиговый регистр представляет собой небольшую интегральную схему. Это последовательный логический счетчик, который позволяет увеличить количество контактов Arduino путем «сдвига» и хранения данных. Регистр выглядит как маленький черный микроконтроллер с 16 ножками. На одном его конце вы увидите точку или полукруг — контакт 1 находится слева от этой метки.



- Необходимое количество: 1
- Количество контактов: 16
- Используется в проектах: 16, 24

Семисегментный светодиодный индикатор

Такой индикатор отображает цифру или символ, выстраивая ее из семи светодиодных сегментов. Светодиодные индикаторы часто используются для отображения номеров счетчиков, часов или таймеров. Доступны индикаторы разных размеров, от одноразрядных (с одним отображаемым символом) до восьмиразрядных; а четырехразрядные индикаторы обычно используются для создания цифровых часов.

- Необходимое количество: 1
- Количество контактов: 10 — 12
- Используется в проектах: 16, 17

Сервопривод

Сервопривод — это двигатель с рычагом, который можно поворачивать на определенное количество градусов, передавая на сервопривод закодированный сигнал. Он выглядит как небольшая коробка с тремя проводами и выходным валом, к которому может крепиться рычаг. Красный провод отвечает за питание и подключается к контакту 5V, а черный (или коричневый) отвечает за заземление и подключается к контакту GND. Оранжевый (или белый) провод передает сигнал управления и подключается к аналоговым контактам платы Arduino. Сервоприводы Tower Pro 9g, используемые в этой книге, поворачиваются на 180 градусов, но существуют модели, способные поворачиваться на 360 градусов.

- Необходимое количество: 2
- Количество контактов: 3
- Используется в проектах: 9, 10, 11, 18, 20, 22, 23

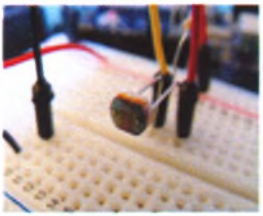
Ультразвуковой дальномер

Ультразвуковой дальномер посылает сигнал (часто называемый пингом), который отражается от объекта и возвращается к датчику. Время от момента отправки сигнала и до получения обратно используется для вычисления расстояния. В проектах этой книги используется датчик модели HC-SR04. Это модульная плата с двумя круглыми датчиками и четырьмя контактами.

- Необходимое количество: 1
- Количество контактов: 4
- Используется в проектах: 18, 20

Фоторезистор

Фоторезистор, также называемый светозависимым резистором, изменяет свое сопротивление в зависимости от количества падающего на него света и используется для определения уровня освещенности. Существуют разные модели, но обычно это небольшой овальный компонент с волнистыми линиями и двумя ножками. Вам необходимо его откалибровать, чтобы определить уровень освещенности в своем помещении, прежде чем использовать в своей программе.



- Необходимое количество: 1
- Количество контактов: 2
- Используется в проектах: 19

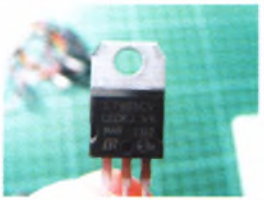
Четырехразрядный семисегментный последовательный индикатор

Это четырехразрядная версия семисегментного светодиодного индикатора, описанного выше, с дополнительной встроенной цепью, поэтому им можно управлять с помощью небольшого количества контактов. На рисунке показан индикатор компании SparkFun, отображающий разные цвета. Он оборудован 10 контактами, но может быть подключен к Arduino с помощью лишь трех из них (VCC, GND и RX).

- Необходимое количество: 1
- Количество контактов: 10 (используется только 3)
- Используется в проектах: 17

5-вольтовый стабилизатор напряжения

5-вольтовый стабилизатор напряжения L7805cv принимает напряжение от 7 до 11 В и стабилизирует его до 5 В.



- Необходимое количество: 1
- Количество контактов: 3
- Используется в проектах: 25

RGB-светодиод

RGB-светодиод содержит три светодиода в одном: красный, зеленый и синий. Смешивая эти цвета в разных соотношениях, вы можете получить любой оттенок. Это прозрачный светодиод с четырьмя ножками, иногда устанавливаемый на модуль со встроенными резисторами, как показано на рисунке. Резисторы необходимы для ограничения напряжения, иначе светодиод перегорит. Самая длинная нога у светодиода без модуля — это либо общий катод, либо анод.

- Необходимое количество: 1
- Количество контактов: 4
- Используется в проектах: 15

RFID-модуль

RFID-модуль предназначен для радиочастотной идентификации. Он считывает RFID-карты и брелки для разрешения (либо запрета) доступа в зависимости от уровня карты. Это небольшая плата с восемью контактами и встроенной антенной. RFID-модуль, используемый в книге, — Mifare RC-522 — обычно поставляется с картой и брелоком.

- Необходимое количество: 1
- Количество контактов: 8
- Используется в проектах: 23

ИНТЕРНЕТ-МАГАЗИНЫ

Большинство электронных компонентов можно найти на крупных сайтах, таких, как «Чип и Дип» или eBay, но если у вас есть проблемы с нахождением той или иной детали, перечисленные ниже розничные продавцы могут вам помочь.

США

Adafruit *www.adafruit.com*
DigiKey *www.digikey.com*
Jameco Electronics *www.jameco.com*
Little Bird Electronics *www.littlebirdelectronics.com*
MCM *www.mcmelectronics.com*
Newark element14 *www.newark.com*
RadioShack *www.radioshack.com*
RS Components *www.rs-components.com*
Seeed Studio *www.seeedstudio.com/depot*
SparkFun *www.sparkfun.com*

Европа

Electronic Sweet Pea's *www.sweetpeas.se*
Element 14 *www.element14.com*
Farnell *www.farnell.com*
Jameco Electronics *www.jameco.com*

Великобритания

4tronix *www.4tronix.co.uk/store*
Cool Components *www.coolcomponents.co.uk*
CPC *cpc.farnell.com*
Hobby Components *www.hobbycomponents.com*
Mallinson Electrical *www.mallinson-electrical.com/shop*
Maplin *www.maplin.co.uk*
Oomlout *oomlout.co.uk*
The Pi Hut *thepihut.com*
Proto-pic *proto-pic.co.uk*
Rapid Electronics *www.rapidonline.com*
RS *uk.rs-online.com/web*
Spiratronics *spiratronics.com*

Россия

Farnell element14 *ru.farnell.com*
Mouser Electronics *ru.mouser.com*
Radiotech *www.rct.ru*
Амперка *amperka.ru*
Вольтмастер *voltmaster.ru*
Импульс *www.impulsi.ru/catalog*
Контест *www.kontest.ru*
Платан *www.platan.ru*
Радио-комплект *www.radio-komplekt.ru*
Чип и Дип *www.chipdip.ru*
Чип-НН *chip-nn.ru*
Электронные компоненты *elecomp.ru*

РАСШИФРОВКА ЗНАЧЕНИЙ РЕЗИСТОРОВ

Почти в каждом проекте этой книги мы использовали резисторы — электрические компоненты, которые ограничивают величину передаваемого через цепь тока (сопротивление измеряется в омах). Они используются для защиты компонентов, таких как светодиоды, от перегрузки и перегорания. Сопротивление резистора

определяется цветными полосками на теле. Резисторы могут иметь четыре, пять или шесть цветных полос.

Важно правильно определять значения резисторов, чтобы использовать корректные детали в своих проектах. Попробуем определить значение резистора с четырьмя полосами, показанного на рис. А.1.

РИСУНОК А.1

Резистор с четырьмя полосками

Глядя на резистор с серебряной или золотой полосой справа, обратите внимание на порядок цветов слева направо. Если резистор не имеет серебряной или золотой полосы, убедитесь, что сторона с тремя цветными полосами находится слева.

Используйте табл. А.1, чтобы определить сопротивление резистора.

ТАБЛИЦА А.1

Расчет значений резистора

ЦВЕТ	ПЕРВАЯ ПОЛОСА	ВТОРАЯ ПОЛОСА	ТРЕТЬЯ ПОЛОСА	МНОЖИТЕЛЬ	ПОГРЕШНОСТЬ
Черный	0	0	0	1Ω	
Коричневый	1	1	1	10Ω	±1%
Красный	2	2	2	100Ω	±2%
Оранжевый	3	3	3	1KΩ	
Желтый	4	4	4	10KΩ	
Зеленый	5	5	5	100KΩ	±0.5%
Синий	6	6	6	1MΩ	±0.25%
Фиолетовый	7	7	7	10MΩ	±0.10%
Серый	8	8	8		±0.05%
Белый	9	9	9		
Золотой				0.1Ω	±5%
Серебряный				0.01Ω	±10%

ПРИМЕЧАНИЕ

Полоса, обозначающая погрешность, обычно окрашена в серебряный или золотой цвет, хотя может быть и любой другой, которому соответствует значение диапазона допуска, указанное в столбце «Погрешность». Если у вашего резистора полоса погрешности окрашена в цвет, отличный от серебряного и золотого, вы увидите небольшой промежуток между полосками значений и полоской допуска, чтобы вы могли отличить их друг от друга.

Значения, соответствующие первой и второй полосам, позволяют вычислить численное значение, третья полоса сообщает, сколько нулей добавить к этому значению, а четвертая означает погрешность, т.е. на сколько настоящее значение может отличаться от обозначенного. Для резистора, показанного на рис. А.1:

- Первая полоса коричневая (1) = 1.
- Вторая полоса черная (0) = 0.
- Третья полоса красная (2) = 00 (2 — это количество нулей).
- Четвертая полоса золотая, т.е. погрешность (точность) будет равна ± 5 %.

Таким образом, это резистор с сопротивлением в 1000 Ом, или 1 кОм, с точностью 5%, что значит, что фактическое значение может быть на 5% больше или меньше, чем 1 кОм. Можно выполнить аналогичные расчеты для резисторов с пятью или шестью полосками.

Если вы все еще не уверены в значении резистора, можете выполнить поиск в Интернете, указав цвета в правильном порядке, прочитав их слева направо, с погрешностью справа.

ПРИЛОЖЕНИЕ Б: СПРАВКА ПО КОНТАКТАМ ARDUINO

ЭТО ПРИЛОЖЕНИЕ ПРЕДСТАВЛЯЕТ СОБОЙ КРАТКИЙ СПРАВОЧНИК ПО КОНТАКТАМ ARDUINO UNO, ИХ ТЕХНИЧЕСКИМ НАЗВАНИЯМ И ФУНКЦИЯМ. КОНТАКТЫ ОПИСЫВАЮТСЯ БОЛЕЕ ПОДРОБНО В ПРОЕКТАХ, ГДЕ ИСПОЛЬЗУЮТСЯ, ТАК ЧТО ПРИВЕДЕННАЯ ЗДЕСЬ ИНФОРМАЦИЯ, ВЕРОЯТНО, БУДЕТ ИМЕТЬ БОЛЬШИЙ СМЫСЛ ПОСЛЕ ТОГО, КАК ВЫ ПОСТРОИТЕ НЕСКОЛЬКО ПРОЕКТОВ.

КОНТАКТ	ФУНКЦИЯ И МЕТКА	ПРИМЕЧАНИЕ
0	RX — используется для получения последовательных данных TTL	
1	TX — используется для отправки последовательных данных TTL	
2	Внешнее прерывание	
3	Внешнее прерывание	Широтно-импульсная модуляция
4	XCK/TO — внешние часы Ввод/Вывод (Таймер/Счетчик 0)	
5	T1 (Таймер / Счетчик 1)	Широтно-импульсная модуляция
6	AIN0 — Аналоговый компаратор, положительный вход	Широтно-импульсная модуляция
7	AIN1 — Аналоговый компаратор, отрицательный вход	
8	ICP1 — Захват ввода	
9	OC1A — Регистр таймера	Широтно-импульсная модуляция
10	SS (Slave Select) — предназначен для активизации ведущим устройством (Master) того или иного периферийного устройства.	Широтно-импульсная модуляция
11	MOSI (Master Out Slave In) — линия для передачи данных от ведущего устройства (Master) к ведомым (Slave).	Широтно-импульсная модуляция
12	MISO (Master In Slave Out) — линия для передачи данных от ведомого устройства (Slave) к ведущему (Master).	
13	SCK (Serial Clock) — тактовые импульсы, генерируемые ведущим устройством (Master) для синхронизации процесса передачи данных.	

КОНТАКТ	ФУНКЦИЯ И МЕТКА	ПРИМЕЧАНИЕ
AREF	Опорное напряжение для аналоговых входов.	
A0	Аналоговый вход может принимать 1024 различных значения.	
A1	Аналоговый вход может принимать 1024 различных значения.	
A2	Аналоговый вход может принимать 1024 различных значения.	
A3	Аналоговый вход может принимать 1024 различных значения.	
A4	Аналоговый вход может принимать 1024 различных значения.	Посредством выводов осуществляется связь I2C (TWI), для создания которой используется библиотека Wire.
A5	Аналоговый вход может принимать 1024 различных значения.	Посредством выводов осуществляется связь I2C (TWI), для создания которой используется библиотека Wire.
RESET	Может использоваться для сброса микроконтроллера.	
3.3V	Напряжение на выводе 3.3 В генерируемое встроенным регулятором на плате.	
5V	Регулируемый источник напряжения 5 В.	
GND	Выводы заземления.	
Vin	Вход используется для подачи питания от внешнего источника или с помощью разъема питания.	

Последовательные: 0 (RX) и 1 (TX). Эти контакты используются для приема (RX) и передачи (TX) последовательных данных в транзисторно-транзисторной логике (TTL). Мы используем контакт TX в ракетной установке в проекте 17.

Внешние прерывания: 2 и 3. Эти контакты могут быть сконфигурированы так, чтобы инициировать прерывание по низкому значению, восходящему или спадающему фронту (сигнал, идущий от низкого до высокого или высокого к низкому соответственно) или изменению значения. Прерывание — это сигнал, который сообщает Arduino об остановке и выполнении другой функции, когда контакты обнаруживают внешнее событие, например нажатие кнопки.

ШИМ: 3, 5, 6, 9, 10 и 11. Эти контакты могут использоваться с широтно-импульсной модуляцией с помощью функции `analogWrite()`. В проекте 2 приведена дополнительная информация.

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Эти контакты поддерживают связь SPI с использованием библиотеки SPI и используются несколько раз в этой книге. Мы используем связь SPI для электронной игры в кости в проекте 16, чтобы Arduino мог отправлять и получать данные из сдвигового регистра, используемого для управления семисегментным светодиодом.

Светодиод: 13. На плате есть встроенный светодиод, подключенный к цифровому выходу 13. Когда контакт находится в режиме `HIGH`, светодиод горит; в режиме `LOW` — выключен. Встроенный светодиод используется для индикации запуска загрузчика микроконтроллер ATmega328p, как правило, при загрузке Arduino.

AREF. Это опорное напряжение для аналоговых входов; контакт используется с функцией `analogReference()`. Допустимо вводить напряжение в диапазоне от 0 до 5 В, поэтому, если ваш датчик использует напряжение ниже 5 В, вы можете использовать этот контакт для увеличения разрешения для более точного считывания.

Аналоговые входы: A0–A5. Плата Uno имеет шесть аналоговых входов, каждый из которых может принимать 1024 различных значения.

TWI: A4 и A5. Эти контакты поддерживают связь TWI (двухпроводного интерфейса) с использованием библиотеки Wire. Используется для управления и связи с устройством I2C, таким как последовательный ЖК-дисплей, через два провода.

RESET. Установите для этого контакта режим `LOW`, чтобы сбросить микроконтроллер. Обычно этот контакт используется для добавления кнопки сброса.

Не беспокойтесь, если эта информация не пригодилась вам прямо сейчас. Она вам понадобится при сборке будущих проектов на основе Arduino, а также вы можете обращаться к ней, работая с проектами из этой книги.

Предметный указатель

Книга «25 крутых проектов с Arduino» — шикарная подборка электронных проектов, которые можно собрать на основе недорогой платы Arduino. С помощью нескольких компонентов, платы Arduino и компьютера вы научитесь создавать и программировать все, что хотите, начиная световыми шоу и мониторами полива растений и заканчивая аркадными играми и ультразвуковыми системами безопасности.

Сначала вы познакомитесь с платой Arduino и получите ценные советы по инструментам и компонентам. Затем вы сможете работать с проектами из книги по порядку или выбирать понравившиеся. Каждый проект содержит простые инструкции, цветные фотографии, наглядные схемы и весь необходимый код.

Книга «25 крутых проектов с Arduino» — быстрый и интересный способ начать работу с этим замечательным микроконтроллером, которой идеально подходит новичкам, гикам, детям, родителям и преподавателям.

Научно-популярное издание

ЭЛЕКТРОНИКА ДЛЯ НАЧИНАЮЩИХ

Марк Геддес

25 КРУТЫХ ПРОЕКТОВ С ARDUINO

Главный редактор *Р. Фасхутдинов*
Ответственный редактор *Е. Истомина*
Младший редактор *Е. Минина*
Художественный редактор *А. Гусев*

ООО «Издательство «Эксмо»
123308, Москва, ул. Зорге, д. 1. Тел.: 8 (495) 411-68-86.
Home page: www.eksmo.ru E-mail: info@eksmo.ru
Өндіруші: «ЭКСМО» АҚБ Баспасы, 123308, Мәскеу, Ресей, Зорге көшесі, 1 үй.
Тел.: 8 (495) 411-68-86.
Home page: www.eksmo.ru E-mail: info@eksmo.ru.
Тауар белгісі: «Эксмо»
Интернет-магазин : www.book24.ru
Интернет-дүкен : www.book24.kz
Импортёр в Республику Казахстан ТОО «РДЦ-Алматы».
Қазақстан Республикасындағы импорттаушы «РДЦ-Алматы» ЖШС.
Дистрибьютор и представитель по приему претензий на продукцию,
в Республике Казахстан: ТОО «РДЦ-Алматы»
Қазақстан Республикасында дистрибьютор және енім бойынша арыз-талаптарды
қабылдаушының өкілі «РДЦ-Алматы» ЖШС,
Алматы қ., Домбровский көш., 3«а», литер Б, офис 1.
Тел.: 8 (727) 251-59-90/91/92; E-mail: RDC-Almaty@eksmo.kz
Өнімнің жарамдылық мерзімі шектелмеген.
Сертификация туралы ақпарат сайтта: www.eksmo.ru/certification

Сведения о подтверждении соответствия издания согласно законодательству РФ
о техническом регулировании можно получить на сайте Издательства «Эксмо»
www.eksmo.ru/certification
Өндірген мемлекет: Ресей. Сертификация қарастырылмаған

Подписано в печать 15.11.2018.
Формат 70x100$^{1}/_{16}$. Печать офсетная. Усл. печ. л. 22,04.
Тираж 2000 экз. Заказ № 1188.

Отпечатано в ОАО «Можайский полиграфический комбинат».
143200, г. Можайск, ул. Мира, 93
www.oaompk.ru, www.оаомпк.рф тел.: (495) 745-84-28, (49638) 20-685

ISBN 978-5-04-090263-7

Оптовая торговля книгами «Эксмо»:
ООО «ТД «Эксмо». 123308, г. Москва, ул.Зорге, д.1, многоканальный тел.: 411-50-74.
E-mail: **reception@eksmo-sale.ru**

По вопросам приобретения книг «Эксмо» зарубежными оптовыми покупателями *обращаться в отдел зарубежных продаж ТД «Эксмо»*
E-mail: **international@eksmo-sale.ru**

International Sales: International wholesale customers should contact Foreign Sales Department of Trading House «Eksmo» for their orders.
international@eksmo-sale.ru

По вопросам заказа книг корпоративным клиентам, в том числе в специальном оформлении, *обращаться по тел.: +7 (495) 411-68-59, доб. 2261.*
E-mail: **ivanova.ey@eksmo.ru**

Оптовая торговля бумажно-беловыми и канцелярскими товарами для школы и офиса «Канц-Эксмо»:
Компания «Канц-Эксмо»: 142702, Московская обл., Ленинский р-н, г. Видное-2, Белокаменное ш., д. 1, а/я 5. Тел.:/факс +7 (495) 745-28-87 (многоканальный).
e-mail: **kanc@eksmo-sale.ru**, сайт: www.**kanc-eksmo.ru**

В Санкт-Петербурге: в магазине «Парк Культуры и Чтения БУКВОЕД», Невский пр-т, д. 46.
Тел.: +7(812)601-0-601, **www.bookvoed.ru**

Полный ассортимент книг издательства «Эксмо» для оптовых покупателей:
Москва. ООО «Торговый Дом «Эксмо». Адрес: 123308, г. Москва, ул.Зорге, д.1.
Телефон: +7 (495) 411-50-74. **E-mail:** reception@eksmo-sale.ru
Нижний Новгород. Филиал «Торгового Дома «Эксмо» в Нижнем Новгороде. Адрес: 603094, г. Нижний Новгород, ул. Карпинского, д. 29, бизнес-парк «Грин Плаза»
Телефон: +7 (831) 216-15-91 (92, 93, 94). **E-mail:** reception@eksmonn.ru
Санкт-Петербург. ООО «СЗКО». Адрес: 192029, г. Санкт-Петербург, пр. Обуховской Обороны, д. 84, лит. «Е». Телефон: +7 (812) 365-46-03 / 04. **E-mail:** server@szko.ru
Екатеринбург. Филиал ООО «Издательство Эксмо» в г. Екатеринбурге. Адрес: 620024, г. Екатеринбург, ул. Новинская, д. 2щ. Телефон: +7 (343) 272-72-01 (02/03/04/05/06/08).
E-mail: petrova.ea@ekat.eksmo.ru
Самара. Филиал ООО «Издательство «Эксмо» в г. Самаре.
Адрес: 443052, г. Самара, пр-т Кирова, д. 75/1, лит. «Е».
Телефон: +7(846)207-55-50. **E-mail:** RDC-samara@mail.ru
Ростов-на-Дону. Филиал ООО «Издательство «Эксмо» в г. Ростове-на-Дону. Адрес: 344023, г. Ростов-на-Дону, ул. Страны Советов, д. 44 А. Телефон: +7(863) 303-62-10. **E-mail:** info@rnd.eksmo.ru
Центр оптово-розничных продаж Cash&Carry в г. Ростове-на-Дону. Адрес: 344023, г. Ростов-на-Дону, ул. Страны Советов, д. 44 В. Телефон: (863) 303-62-10
Режим работы: с 9-00 до 19-00. **E-mail:** rostov.mag@rnd.eksmo.ru
Новосибирск. Филиал ООО «Издательство «Эксмо» в г. Новосибирске. Адрес: 630015, г. Новосибирск, Комбинатский пер., д. 3. Телефон: +7(383) 289-91-42. **E-mail:** eksmo-nsk@yandex.ru
Хабаровск. Обособленное подразделение в г. Хабаровске. Адрес: 680000, г. Хабаровск, пер. Дзержинского, д. 24, литера Б, офис 1. Телефон: +7(4212) 910-120. **E-mail:** eksmo-khv@mail.ru
Тюмень. Филиал ООО «Издательство «Эксмо» в г. Тюмени.
Центр оптово-розничных продаж Cash&Carry в г. Тюмени.
Адрес: 625022, г. Тюмень, ул. Алебашевская, д. 9А (ТЦ Перестройка+).
Телефон: +7 (3452) 21-53-96/ 97/ 98. **E-mail:** eksmo-tumen@mail.ru
Краснодар. ООО «Издательство «Эксмо» Обособленное подразделение в г. Краснодаре
Центр оптово-розничных продаж Cash&Carry в г. Краснодаре
Адрес: 350018, г. Краснодар, ул. Сормовская, д. 7, лит. «Г». Телефон: (861) 234-43-01(02).
Республика Беларусь. ООО «ЭКСМО АСТ Си энд Си» Центр оптово-розничных продаж Cash&Carry в г.Минске. Адрес: 220014, Республика Беларусь, г. Минск, пр-т Жукова, д. 44, пом. 1-17, ТЦ «Outleto». Телефон: +375 17 251-40-23; +375 44 581-81-92.
Режим работы: с 10-00 до 22-00. **E-mail:** exmoast@yandex.by
Казахстан. РДЦ Алматы. Адрес: 050039, г. Алматы, ул. Домбровского, д. 3 «А»
Телефон: +7 (727) 251-59-90 (91,92). **E-mail:** RDC-Almaty@eksmo.kz
Интернет-магазин: www.book24.kz
Украина. ООО «Форс Украина». Адрес: 04073 г. Киев, ул. Вербовая, д. 17а
Телефон: +38 (044) 290-99-44. **E-mail:** sales@forsukraine.com

Полный ассортимент продукции Издательства «Эксмо» можно приобрести в книжных магазинах «Читай-город» и заказать в интернет-магазине www.chitai-gorod.ru.
Телефон единой справочной службы: 8 (800) 444 8 444. Звонок по России бесплатный.

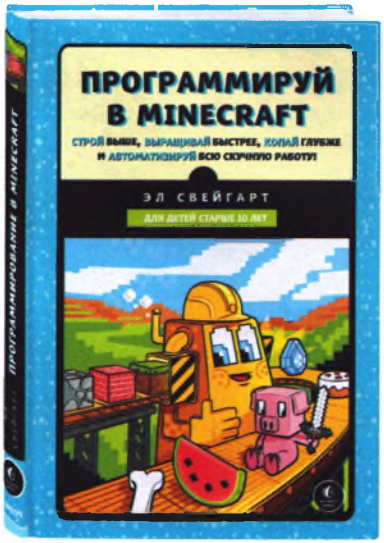
Интернет-магазин ООО «Издательство «Эксмо»
www.book24.ru
Розничная продажа книг с доставкой по всему миру
Тел.: +7 (495) 745-89-14. E-mail: **imarket@eksmo-sale.ru**

BOOK24.RU интернет-магазин книги

EKSMO.RU новинки издательства

НАУЧИТЕ ДЕТЕЙ ПРОГРАММИРОВАТЬ УЖЕ СЕЙЧАС С ПОМОЩЬЮ НАШЕЙ НОВОЙ СЕРИИ КНИГ!

ПРОГРАММИРОВАНИЕ ДЛЯ ДЕТЕЙ



ПОЛЕЗНЫЕ, ИНТЕРЕСНЫЕ И ЗАБАВНЫЕ УПРАЖНЕНИЯ НЕ ДАДУТ ЗАСКУЧАТЬ!

2018-401

ВСЕ ПРОЕКТЫ ВЫПОЛНЕНЫ С ИСПОЛЬЗОВАНИЕМ ARDUINO UNO

ВЫ КУПИЛИ ARDUINO – ЧТО ДАЛЬШЕ?

«25 КРУТЫХ ПРОЕКТОВ С ARDUINO» – это лучшая книга для начинающих, ведь все проекты в ней выполнены на основе самой бюджетной из плат Arduino. Имея под рукой всего две составляющие, Arduino и компьютер, вы научитесь создавать и программировать самые разные вещи – от светомузыки и игр до ультразвуковой охранной системы. Работать с этой книгой очень просто: выбирайте любые проекты, которые вам нравятся, и приступайте к их созданию! Каждый проект снабжен простой инструкцией, цветными фотографиями, электрическими схемами и кодом.

Эта книга – самый веселый способ начать работать с микроконтроллерами и подходит как для начинающих, так и для опытных пользователей Arduino, а также для занятий родителей с детьми или обучения.

25 ПОШАГОВЫХ ПРОЕКТОВ

- СВЕТОДИОДНАЯ ПАНЕЛЬ
- ДИСКОТЕЧНЫЙ СТРОБОСКОП
- МОНИТОР ПОЛИВА
- ДЕТЕКТОР ПРИЗРАКОВ
- ПРОИГРЫВАТЕЛЬ
- ЭЛЕКТРОННЫЙ ПРИВРАТНИК
- ЛАЗЕР, УПРАВЛЯЕМЫЙ ДЖОЙСТИКОМ
- МЕТЕОСТАНЦИЯ
- ПРЕДСКАЗАТЕЛЬ СУДЬБЫ
- ЭЛЕКТРОННЫЕ КОСТИ
- РАКЕТНАЯ ПУСКОВАЯ УСТАНОВКА
- ЛАЗЕРНАЯ СИГНАЛИЗАЦИЯ
- ДАТЧИК ДВИЖЕНИЯ
- СВЕТОВОЕ ШОУ

И ДРУГИЕ.

«Эта книга отлично подойдет тем, кто знаком с основами программирования Arduino и хочет проверить и закрепить знания на практике. В ней хорошо объясняются основы электроники и технические моменты, касающиеся пайки, подготовки рабочего места, и многие другие. Еще одним плюсом книги является наличие иллюстраций и таблиц. Сами проекты интересные и хорошо подходят для обучения – большую часть из них можно включать в практические части обучающих занятий по Arduino».

МИХАИЛ КИСЕЛЕВ,
руководитель отдела разработки в проектах
«Лига Роботов» и «Эра Инженеров»

ISBN 978-5-04-090263-7