Общие сведения об Arduino

Ардуино (Arduino) – это название комплекса аппаратно-программных средств для создания простых электронных систем автоматики и робототехники. Система имеет полностью открытую архитектуру и ориентирована на непрофессиональных пользователей. Она может служить для разработки автономных интерактивных устройств и работает под управлением ПО, установленного на соединенном с ним компьютером. Arduino можно собрать самостоятельно вручную, а можно приобрести в готовом виде. Интегрированную среду разработки с открытым кодом можно загрузить бесплатно с веб-сайта www.arduino.cc

Аппаратные средства Arduino Nano

Arduino представляет собой небольшую плату, на которую устанавливается микроконтроллер с прошитым в него загрузчиком. С помощью загрузчика записывается программа в микроконтроллер из персонального компьютера без применения аппаратных программаторов.

14 цифровых разъемов ввода-вывода (контакты 0-13)

Это могут быть как входы, так и выходы, что задается программным модулем, который создаетя в интегрированной среде разработки Arduino IDE. Цифровые контакты помечены буквой D (digital).

8 аналоговых входов (контакты A0-A7)

На эти входы от разных датчиков подаются, которые затем преобразуются в цифровые значения в диапазоне от 0 до 1023. Обозначены на плате буквой A (analog).

6 аналоговых выходов (контакт 3, 5, 6,9,10,11)

В действительности это шесть цифровых выводов, которые можно перепрограммировать в условно аналоговые (диапазон принимаемых значений от 0 до 255, что соотвктствует значениям напряжения от 0 до 5 Вольт)

Среда разработки

Программная часть Ардуино представлена интегрированной программной среды (IDE), позволяющей писать, компилировать программы, а также загружать их в аппаратуру. Для программировании Ардуино используется язык C/C++ с некоторыми особенностями.

Электронный конструктор «Робот Печенег Батана» разработан на основе одной из популярных модификаций платы — Arduino Nano.

Arduino Nano.

Arduino Nano – платформа, построенная на микроконтроллере Atmega 328 (Arduino Nano 3.0) или Atmega 168 (Arduini Nano 2.x). Ее особенностями являются:

- небольшие размеры, что позволяет создавать компактные устройства;
- отсутствие разъема постоянного тока;

- работа через MiniUSB;

Некоторые технические характеристики:

- 14 цифровых пинов, 6 из которых могут использоваться в качестве выходов ШИМ;
- 8 входов аналогового сигнала;
- флеш-память 16 Кб или 32 Кб в зависимости от микроконтроллера;
- ОЗУ 1 Кб или 2 Кб, в зависимости от микроконтроллера;

Контакты аналогового и цифрового ввода и вывода работают в диапазоне от 0 до 5В, а протекающий ток не должен превышать 40 мА.

Также на плате установлены 4 светодиода, являющихся индикаторами сигнала. Они обозначены как TX и RX (индикаторы передачи данных), PWR (индикатор питания палаты) и L (светодиод общего назначения).

Работа в Arduino IDE

Установка необходимого ПО

Программная часть платформы Arduino представлена интегрированной средой разработки Arduino IDE. Установленная на компьютере, она позволяет составлять программные модули для платы Arduino.

Среду разработки можно свободно скачать с сайта www.arduino.cc. После распаковки файла и установки программы, следует обеспечить возможность общения платы и компьютера через USB-порт. Для этого необходимо установить драйвер, в случае с Arduino Nano, входящих в наборы конструктора компании «ЛАРТ» это будет CH340.

После запуска Arduini IDE следует в меню *Инструменты/Плата* выбрать плату, с которой вы собираетесь работать (в нашем случае — Arduino Nano), а также в *Инструменты/Порт* выбрать порт, который соответствует плате.

Структура языка.

Как правило, тело программы состоит из двух основных блоков:

- void setup() - фрагмент программы, содержащий код инициализации — блок команд, устанавливающий плату в состояние, необходимое для запуска основного цикла программы.
- void loop() - основной блок программы. Состоит из набора команд, которые повторяются до тех пор, пока не будет выключено питание платы.

Константы

В языке Arduino имеется ряд предопределенных ключевых слов, каждому из которых соответствует свое значение. Наиболее часто встречаются:

- INPUT/OUTPUT – константы, служащие для назначения пина на вход или выход;
- константы HIGH/LOW используются для обозначения подачи питания на пин или его отсутствия;
- true/false -логические константы, соответствуют логическим 1 и 0;

Переменные

Переменные являются именованными областями памяти платы Arduino, где хранятся данные, которые можно использовать и обрабатывать в пользовательской программе. Переменные подразделяются на локальные (работают только в пределах функции или цикла, где были объявлены) и глобальные (работают во всех функциях программы).

При первом обращении к переменной указывают ее тип. Наиболее часто используют:

- boolean – логический тип, переменные могут принимать два значения: *true* или *false*;

  boolean a = true; // присваивает переменной a значение true;

  boolean b = 0; // присваивает переменной b значение false;

  boolean c = 1; // присваивает переменной c значение true;

  boolean d = 5; // присваивает переменной d значение true;

- char – тип переменной, содержащей один символ, хранится в виде числа от -128 до 127. Символьный тип может быть использован для хранения чисел;

  char a = ‘A’;

  char b = 65; // Эквивалентные записи

- byte – численный тип, может принимать значения от 0 до 255;

  byte x = 35; // переменные типов int, unsigned int, long, unsigned long, float;

- int – целочисленный тип, используется для записи чисел от -32765 до 32767*;*
- unsigned int – целочисленный тип, используется для записи чисел от 0 до 65535
- long – целочисленный тип, используется для записи чисел от -2 147 483 648 до 2 147 483 645
- unsigned long – целочисленный тип, используется для записи чисел от 0 до 4 294 967 295

- float число с плавающей запятой. Позволяет хранить числа, записанные в виде десятичной дроби

- array – массив, то есть список переменных, к которым можно получить организованный доступ посредством индекса. Массив может состоять из переменных требуемого типа, в зависимости от поставленной задачи.

  int array a[5] ={2, 4, 5, 2, 7}; // Объявлен массив переменных типа int

- String – массив символов;

  char sting_1[4] = {'l', 'a', 'r', 't'};

  char sting_2[4] = "lart'; // Эквивалентные записи

Управляющие конструкции

- if...else – оператор условия. После if в круглых скобках задается логическое выражение. Если выражение верно, выполняется блок программы, следующий за этим условием (он должен быть заключен в фигурные скобки), или, если выражение ложно, выполняется блок после else;

```
if (x >=255)
        {x = 0;
         Serial.println('counter is full');
        }
 else {x = x + 10;
      delay(100);}
```

- switch case – условный оператор, который может иметь более двух различных алгоритмов действия, в зависимости от значения некой заданной переменной

```
switch(var) {
    case 1:
        // выполняется, когда var равно 1
        break;
    case 2:
        // выполняется, когда var равно 2
        break;
    default:
        // выполняется, если не выбрана ни одна альтернатива (default не обязателен)
```

- for – оператор цикла. Используется, когда заранее известно, сколько раз необходимо выполнить зацикленный алгоритм;

```
for (int i = 0; i <=255; i++){
        analogWrite(x, I);
        delay(10);
        }
```

- while – оператор цикла с условием. Блок программы выполняется снова и снова, пока некоторое заданное выражение не примет значение TRUE. Проверка осуществляется на входе цикла;

```
while (x < 255){
        x++;
        delay(10);
    }
```

- do while — оператор цикла с условием. Аналогичен while, за исключением того, что оценка условия происходит после выполнения программного кода. Эта конструкция используется, когда нужно, чтобы код внутри блока выполнился хотя бы раз до проверки условием;

```
do {
        x++;
       delay(10);}
while (x < 255)
```

Оператор присваивания

В языке Ардуино знак «=» играет роль оператора присваивания. Используется он следующим образом: переменной, указанной левее оператора присваивания присваивается значение выражения, указанного правее. Например:

```
b = 21;  // переменной b присвоен значение 21
b = b + 30;  // значение переменной b увеличится на 30
b = min (x, y);  // переменной b присвоено минимальное значение из x и y
```

Арифметика, формулы

Помимо стандартных арифметических операций — сложение (+), вычитание(-), умножение(*), деление (/) - существуют некоторые дополнительные операторы. Например, Modulo (%), возвращает остаток целочисленного деления. Также удобно использовать инкремент (++) и декремент(--).

Операторы сравнения

- == - равно;

- != - не равно;
- <, > - меньше, больше;
- >=, <= - больше или равно, меньше или равно;

Логические операторы

- && - логическое И;
- || - логическое ИЛИ;
- ! - логическое отрицание.

Функции для входов и выходов

- pinMode(pin, INPUT/ OUTPUT) - функция, задающая конфигурацию для цифрового пина в качестве входа (INPUT) или выхода (OUTPUT);
- digitaiWrite (pin, LOW/HIGH) – функция управления напряжением на цифровом пине. Для ее использования необходимо предварительно назначить пин на выход при помощи функции pinMode;
- digitalRead(pin) – функция считывания состояния входного контакта. Принимает значение HIGH при наличии напряжении на цифровом входе и LOW при его отсутствии;
- analogRead(pin) – функция, считывающая напряжение на аналоговом входе. Возвращает значение от 0 до 1023, которое соответствует напряжению от 0 до 5 В;
- analogWrite(pin, value) — функция, изменяющая частоту широтно-импульсной модуляции на одном из пинов с ШИМ. Аргумент value может принимать значение от 0 до 255, что соответствует напряжению от 0 до 5 В.

Функции времени

- delay (mSec) – функция, останавливающая исполнение кода программы на указанное число миллисекунд;
- millis() - функция, возвращающая время, прошедшее с момента запуска программы в миллисекундах.

Математические функции

- min(x, y), (max(x, y)) – функции поиска минимума (максимума) из двух чисел x и y;
- abs(x) – функция, возвращающая модуль числа x;
- constrain(x, a, b) – возвращает число x, если оно принадлежит отрезку [a,b], число a, если x меньше, чем a, и число b, если x больше, чем b;

- map(x, a1, a2, b1, b2) — функция, отображающая значение x из диапазона[a1,a2] в диапазон [b1,b2]. Ту же операцию можно выполнить алгебраически.

- pow(a, x) — возводит основание a в степень x;

- sqrt(x) – извлекает квадратный корень из x;

- sin(x), cos(x), tan(x) – определяет синус/косинус/тангенс от x, выраженного в радианах;

Последовательный интерфейс

USB-порт на плате Arduino используется не только для того, для загрузки программного кода с компьютера, но и для передачи данных в компьютер по протоколу последовательного интерфейса. Именно для этого применяется объект *Serial*. Некоторые функции последовательного интерфейса:

- Serial.begin(speed) – функция, задающая работу последовательного интерфейса. Обычно используют значение скорости 9600 (бит/сек);

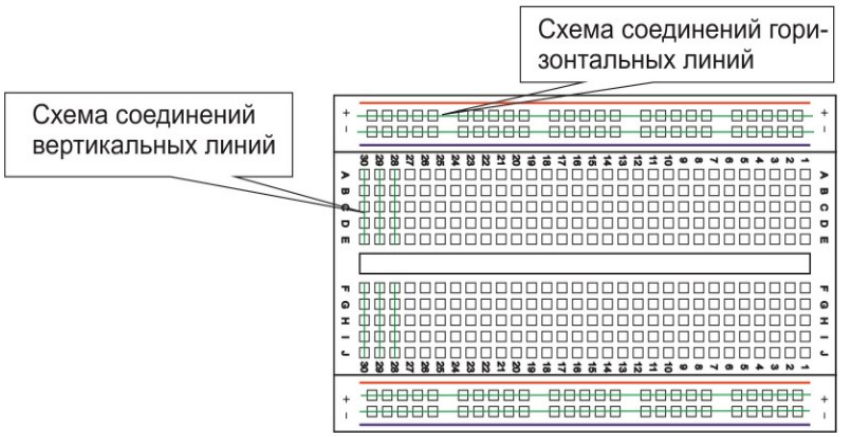
- Serial.print(данные), Serial.print(данные, кодировка) — функция, отправляющая данные в последовательный порт. Возможно изменение кодировки;

- Serial.println(данные), Serial.println(данные, кодировка) — тоже функция, отправляющая данные в последовательный порт, но добавляются символы перехода на новую строку и возврата каретки.

Работа с макетной платой. Часть 1.

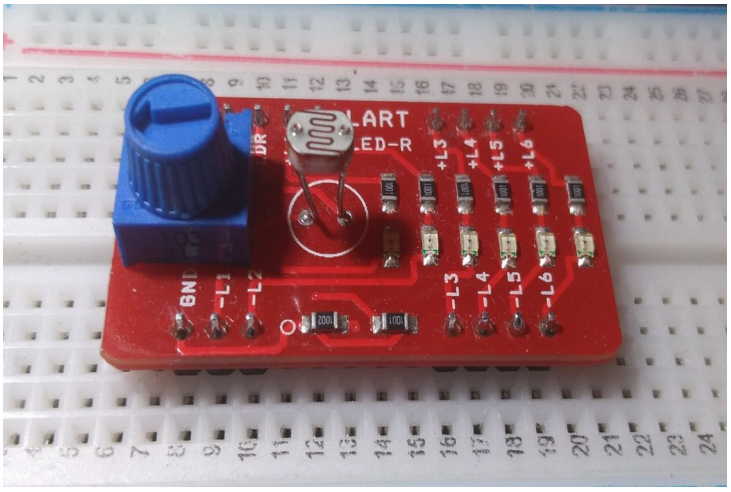
Макетная плата — универсальная печатная плата для сборки и моделирования прототипов электронных устройств.

На макетной плате имеется большое количество отверстий, связанных между собой, например, металлическими полосками. Контакты в двух верхних и двух нижних отверстиях соединены друг с другом — это шины питания. Отверстия в средней части связаны между собой вертикально, как показано на рисунке.



Модуль LED-R

Первым модулем, с которым мы будем работать, является модуль LED-R. Он включает в себя 6 светодиодов (L1 – L6), включенных через резисторы, фоторезистор (LDR) и потенциометр (POT). Контакты питания и заземления промаркированы соответственно GND и +5V.



Разберем, из чего состоит модуль:

- Светодиод — полупроводниковый элемент, преобразующий электрический ток в световое излучение. Имеет два контакта — анод (длинная ножка) и катод (короткая), подключающиеся соответственно к плюсу и минусу;

- Резистор — элемент, обладающий сопротивлением. Снимает излишек напряжения со светодиода.
- Фоторезистор — резистор, сопротивление которого меняется в зависимости от освещенности.
- Потенциометр — регулируемый делитель электрического напряжения.

Для подключения каждого светодиода необходимо соединить через макетную плату контакт +L* и один из цифровых контактов Ардуино, а контакт -L* к общему минусу. Чтобы управлять несколькими светодиодами через один цифровой вывод, необходимо подключить их параллельно. Для этого подключим контакты светодиодов к общей шине питания на макетной плате, а ее, в свою очередь к цифровому выводу Ардуино.

Широтно-импульсная модуляция (ШИМ, PWM)

Микроконтроллеры обычно не могут выдавать произвольное напряжение. Они могут выдать либо напряжение питания (например, 5 В), либо землю (т.е. 0 В)

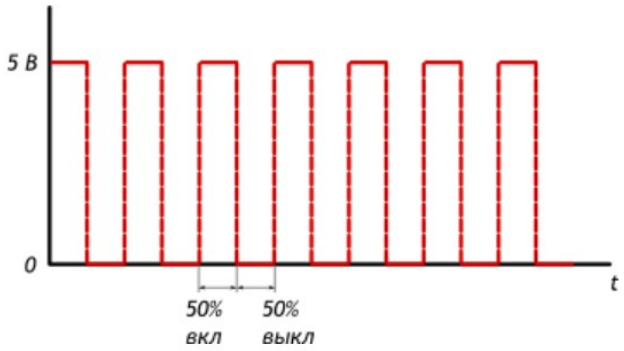
Но уровнем напряжения управляется многое: например, яркость светодиода или скорость вращения мотора. Для симуляции неполного напряжения используется ШИМ (Широтно-Импульсная Модуляция, англ. Pulse Width Modulation или просто PWM)

Выход микроконтроллера переключается между землёй и Vcc тысячи раз в секунду. Или, как ещё говорят, имеет частоту в тысячи герц. Глаз не замечает мерцания более 50 Гц, поэтому нам кажется, что светодиод не мерцает, а горит в полсилы.

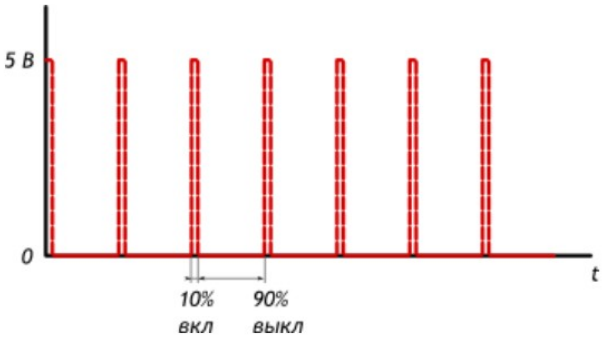
Аналогично, разогнанный мотор не может остановить вал за миллисекунды, поэтому ШИМ-сигнал заставит вращаться его в неполную силу.

Отношение полного периода к времени включения называют *скважностью* (англ. duty cycle). Рассмотрим несколько сценариев при напряжении питания *Vcc* равным 5 вольтам.

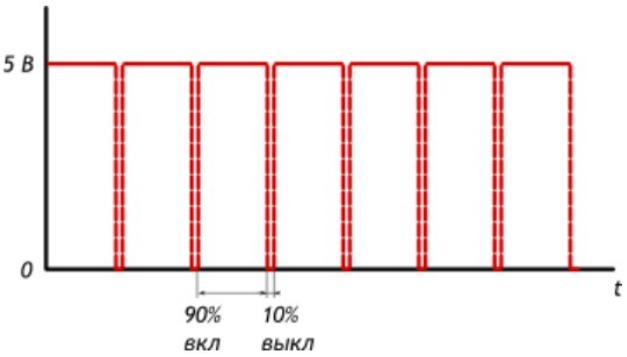
50% — эквивалент 2,5 В



10% — эквивалент 0,5 В



90% — эквивалент 4,5 В



Использование ШИМ.

Для использования возможностей работы с широтно-импульсной модуляцией подходят только цифровые контакты под номерами 3, 5, 6, 9, 10, 11.

Для аналогового вывода используется функция analogWrite(pin, value), где значение value может принимать значения от 0 до 255, что будет соответствовать напряжению от 0 до 5 вольт.

Считывание показаний датчиков.

Чтобы сделать устройство более интерактивным, интегрированным в среду, оно должно принимать сигналы извне и реагировать на них. Для этого в Arduino предусмотрена возможность чтения цифрового и аналогового сигналов.

Для считывания аналоговых показаний датчик должен быть подключен к аналоговому входу (они промаркированы A0 - A6. В таком случае используется функция analogRead(pin). Показания датчика могут принимать значение от 0 до 1024.

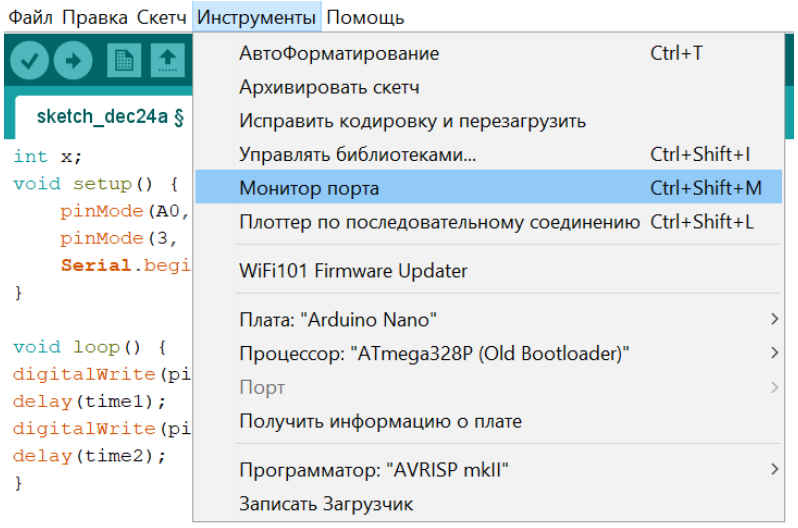
Для цифровых показаний (например, положение кнопки) используют цифровые контакты и функцию digitalRead(pin).

Работа с объектом Serial.

Значения, считанные датчиками, для удобной работы и подбора констант для сравнения выводят в монитор порта. Чтобы начать обмен данными необходимо указать в программе на начало работы с объектом Serial в разделе setup при помощи функции Serial.begin(speed), указав скорость обмена данными в качестве аргумента. Внаших программах на достаточно скорости 9600 бит/с.

Функции Serial.print() и Serial.println() позволяют выводить значения датчиков в монитор порта.

После загрузки программы Ардуино, осталось только открыть монитор порта, чтобы увидеть значения, которые выдают датчики. Для этого для этого во вкладке *Инструменты* выберите *Монитор порта.*



Фоторезистор. Подключение и считывание показаний.

Фоторезистор — резистор, меняющий сопротивления в зависимости от освещения.

Для снятия показаний с фоторезистора нужно подключить один контакт к питанию, а другой к аналоговому входу Arduino.

Подключение модуля: контакт 5+ модуля подключается к питанию на макетной плате, а контакт LDR должен быть соединен с одним из аналоговых входов Arduino.