

Занятие 1

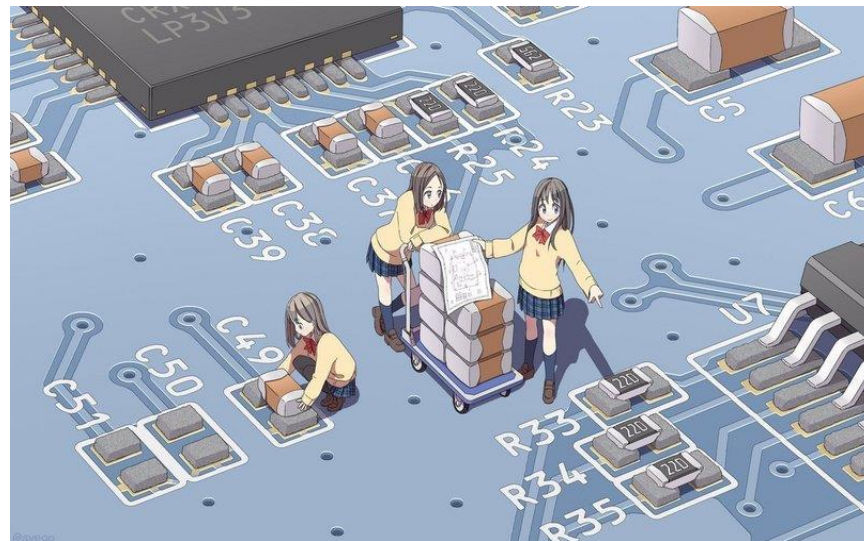
План занятия:

1. Введение

2. Написание программы управления двигателями.

3. Написание программы для связи Arduino с ПК по serial порту.

Что вы представляете, когда слышите слово радиотехника?



Кем вы станете... если...



Ну а если без шуток..

Сейчас смотрят 2 человека

Программист микроконтроллеров

от 250 000 ₽ на руки Опыт 3-6 лет


Даичи  Онлайн

Москва

Отклик без резюме

Инженер-программист микроконтроллеров

180 000 – 250 000 ₽ на руки Опыт более 6 лет


ООО Тетрон 

Москва

Отклик без резюме

Инженер-программист микроконтроллеров


120 000 – 170 000 ₽ на руки Опыт 1-3 года

ООО Прикладная Робототехника 

Москва

FPGA-разработчик

от 300 000 ₽ на руки Опыт 1-3 года

ООО Актив Матрикс 

Москва

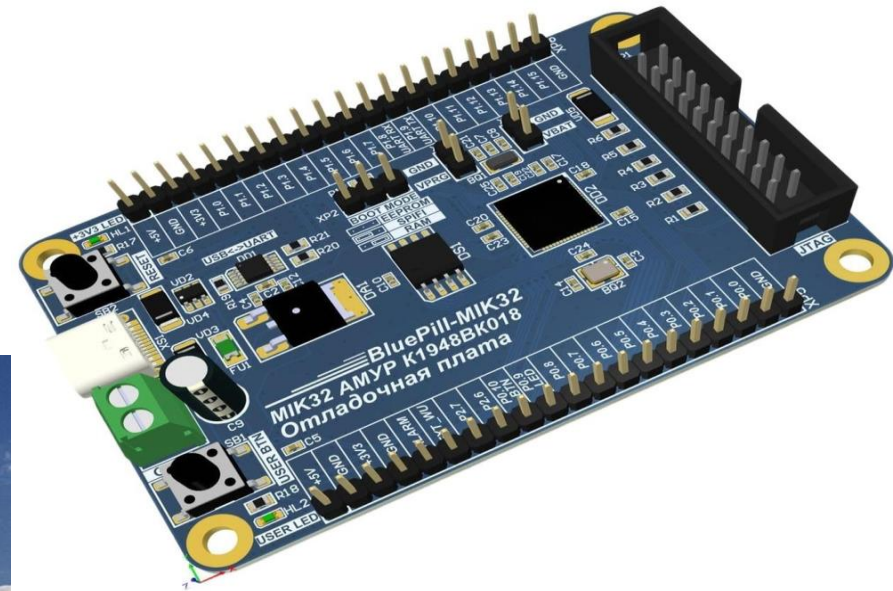
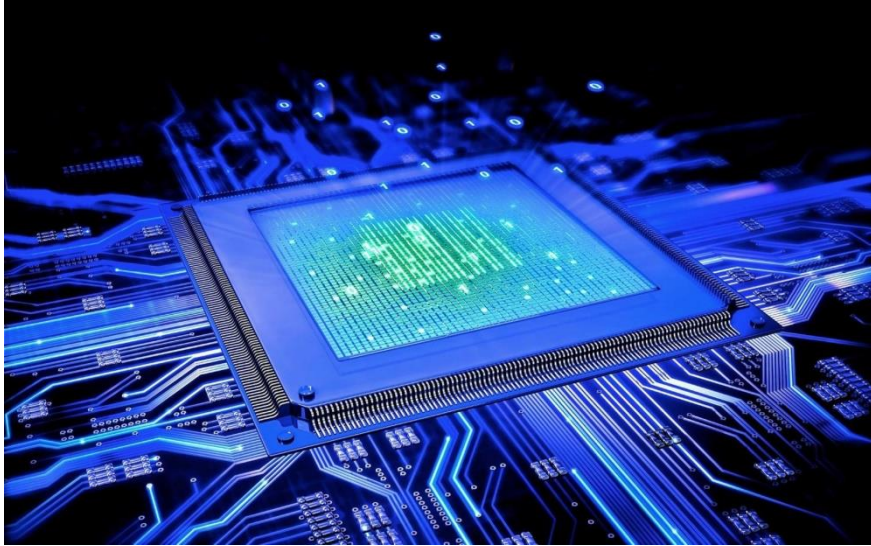
Инженер-разработчик ПЛИС/FPGA

от 200 000 ₽ на руки Опыт 3-6 лет

Единый центр карьеры Госкорпорации Роскосмос 

Москва

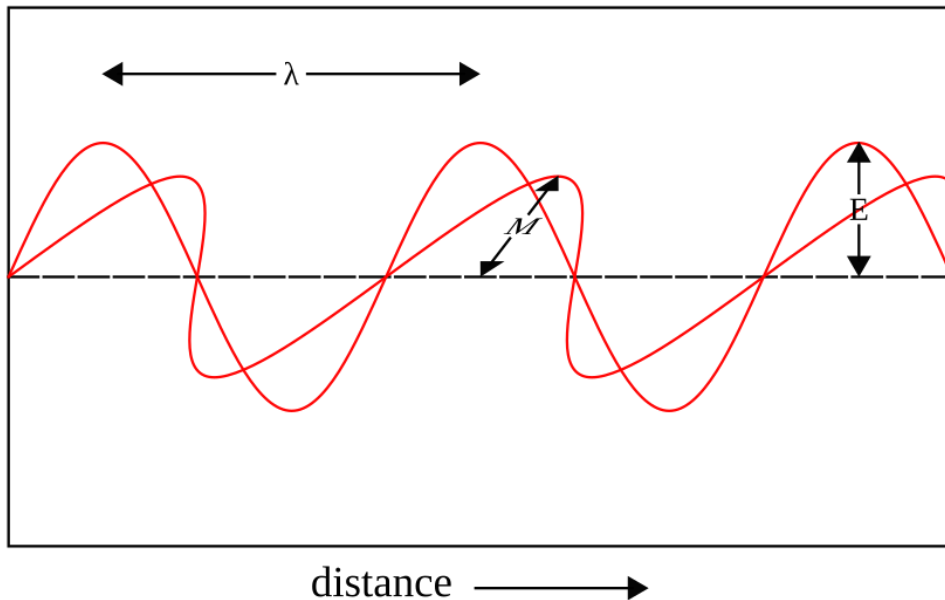
Но для этого нужно очень много и упорно
учиться..



Что такое радиоэлектроника?

Радиоэлектроника — область науки и техники, охватывающая теорию, методы создания и использования устройств для передачи, приёма и преобразования информации с помощью электромагнитной энергии

Light wave



λ = wave length

E = amplitude of
electric field

M = amplitude of
magnetic field

Зачем нужна радиоэлектроника?

Методы и средства радиоэлектроники находят широкое применение в

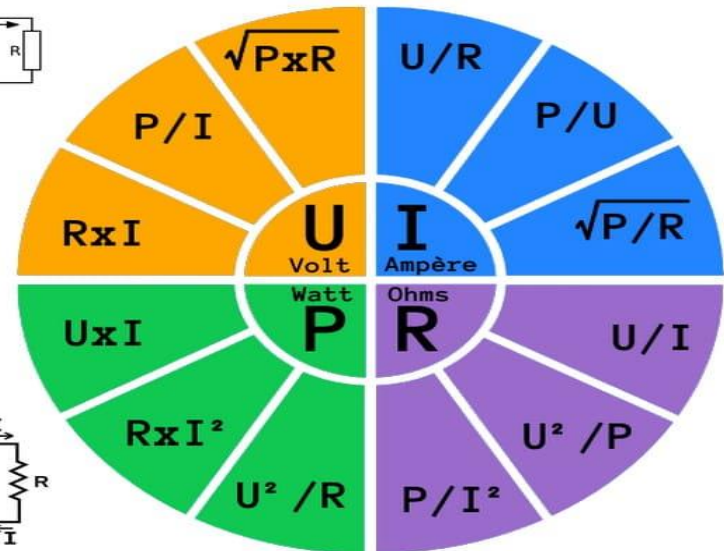
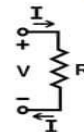
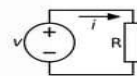
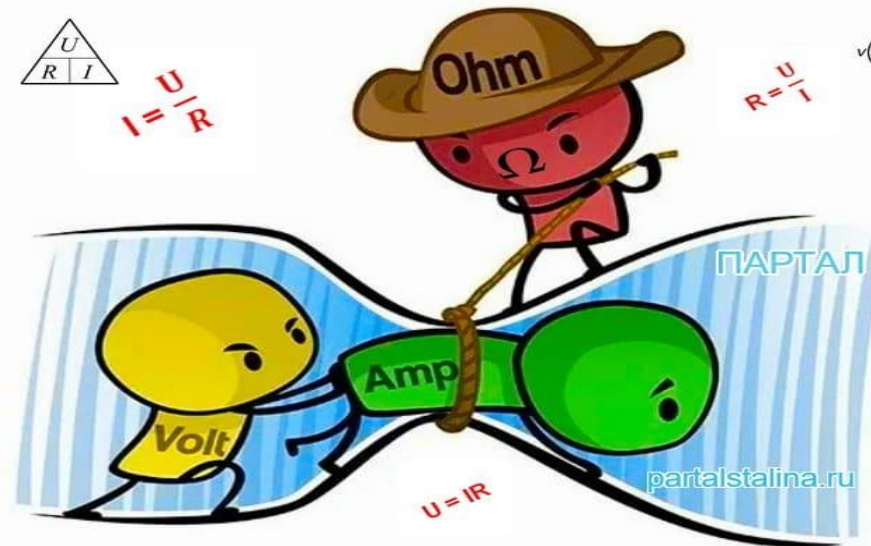
- радиосвязи,
- медицина,
- системах дистанционного управления,
- радионавигации,
- автоматике,
- радиолокации,
- бытовой, военной, космической, вычислительной техниках

Электричество

У **полюсов** есть так называемый **потенциал**, который можно сравнить с высотой уровня воды, например у нас есть два сосуда с водой, расположенных на разной высоте

Так вот, **напряжение** - это разность потенциалов, то есть насколько потенциально быстро вода может течь по трубе

Электрический ток - это сам процесс движения электронов (частиц воды), и физически он определяется как количество электронов за единицу времени, то есть в нашей аналогии это объем воды, протекающий через трубу за единицу времени.



Электрический сигнал

Электрический сигнал бывает двух типов: аналоговый и цифровой.

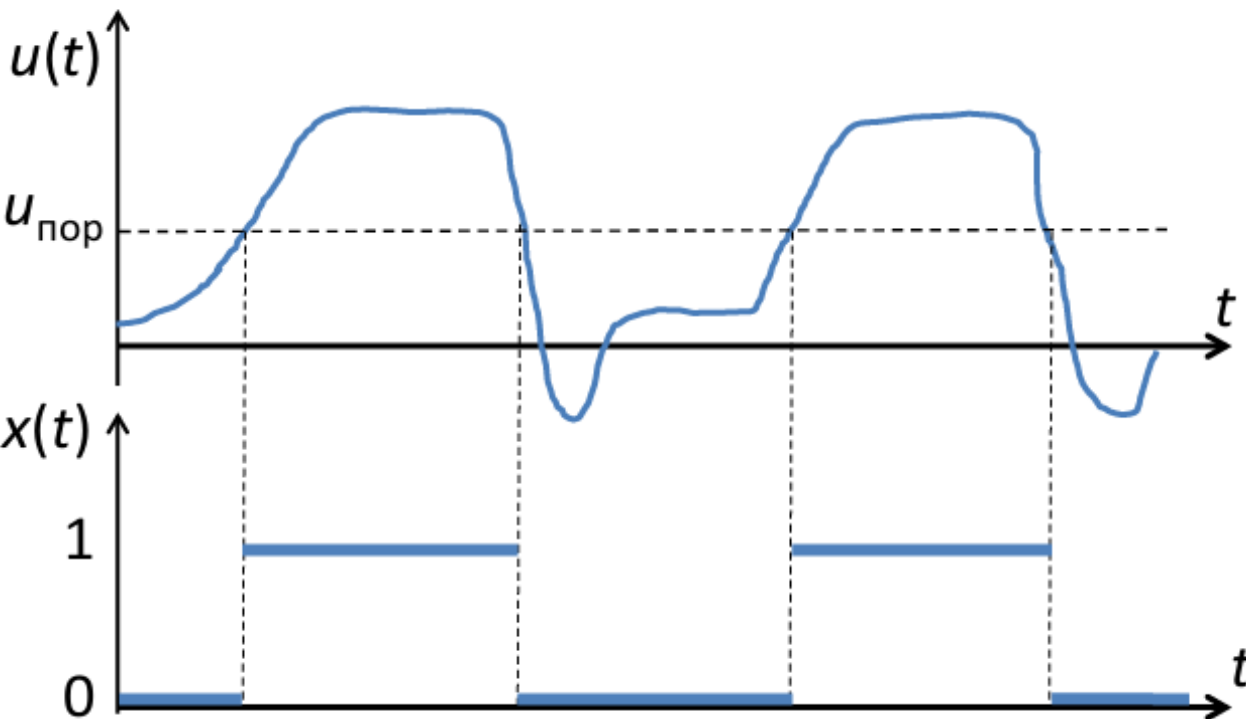
Аналоговый сигнал - напряжение такого сигнала может меняться с очень маленьким шагом в Вольтах. В большинстве случаев такой сигнал *"непрерывен"*, то есть не может измениться резко: в любой момент времени мы можем его измерить и перевести в нужную величину (например, напряжение в температуру).

Цифровой (дискретный, логический) сигнал - напряжение такого сигнала имеет всего два значения, низкое и высокое: 0V и 3.3V/5V. С точки зрения данных это 0 (ноль) и 1 (единица).

Логический уровень практически всегда совпадает с напряжением питания компонента, то есть если микросхема питается от напряжения 5V, то и логический уровень у неё будет 5V.



Интерпретация импульсного сигнала в качестве цифрового сигнала



Цифровой сигнал – это абстракция, придуманная человеком и используемая им для удобства хранения, обработки и передачи данных.

$$x(t) = \begin{cases} 1, & u(t) \geq u_{\text{пор}} \\ 0, & u(t) < u_{\text{пор}} \end{cases}$$

Как устроен компьютер?

В функциональном устройстве компьютера можно выделить следующие основные блоки:

- **Арифметико-логического устройства (АЛУ)**, способного выполнять операции над данными;
- **Устройства управления (УУ)**, осуществляющего выборку команд для АЛУ, считывание операндов и размещение результатов операций.



Шина – это кабель, состоящий из множества проводников. Количество проводников, входящих в состав шины, является **максимальной разрядностью шины**.

Системная шина, в свою очередь, представляет собой совокупность:

- шины данных, служащей для переноса информации;
- шины адреса, которая определяет, куда переносить информацию;
- шины управления, которая определяет правила для передачи информации;
- шины питания, подводящей электропитание ко всем узлам вычислительной машины.

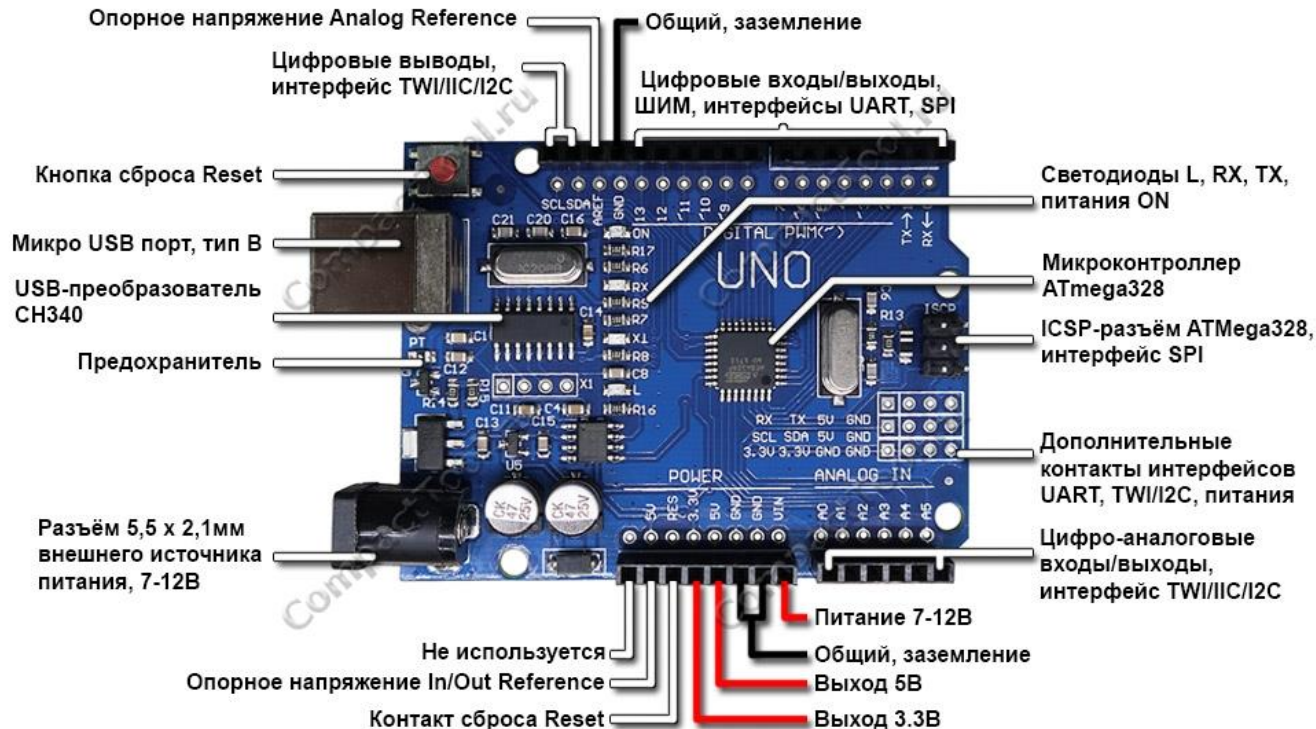
Что такое Arduino?

Arduino UNO – отладочная плата.

Отладочная плата представляет собой как ни странно печатную плату, на которой стоит микроконтроллер (далее МК) – та самая штука, которую мы будем программировать.

В **Arduino UNO** используется микроконтроллер архитектуры **AVR** (Atmega328p).

Arduino IDE– интегрированная среда разработки.

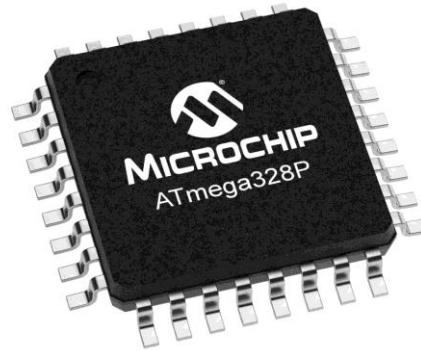


Микроконтроллер (Маленький компьютер)

Микроконтроллер - программируемая микросхема. Самый простой аналог – компьютер. Микроконтроллер работает сам по себе, на нём может быть запущена простенькая операционная система, может даже быть выход в Интернет, а мы можем подключать к нему устройства ввода, датчики, дисплеи и прочие железки.

МК умеет делать всего три вещи:

1. Измерять напряжение на выводе
2. Выдавать напряжение с вывода
3. Программироваться



Аппаратные блоки МК:

1. **Ядро** - Обрабатывает информацию, управляет данными и выполняет вычисления.
2. **Flash память** - постоянное запоминающее устройство (ПЗУ). Хранит исполняемый код программы.
3. **SRAM память** - оперативное запоминающее устройство (ОЗУ). Хранит данные, изменяющиеся в процессе работы программы (промежуточные результаты вычислений, значения переменных, принятые от внешних устройств данные и т.д.). Очищается после сброса питания.

Другие блоки

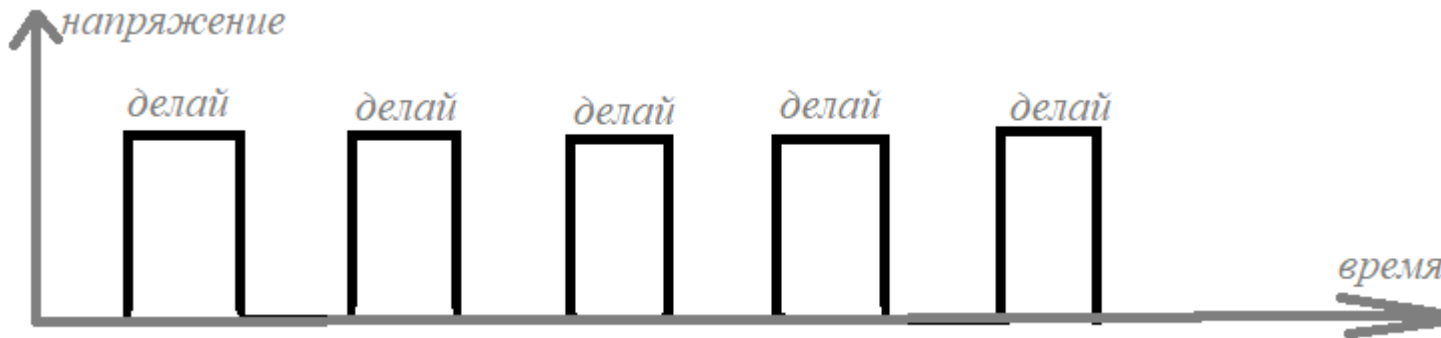
1. **GPIO** (General Purpose Input-Output) - **вход-выход общего назначения**.
Измеряет поданный на пин цифровой сигнал, либо выдаёт его с пина.
2. **АЦП** (ADC, аналогово-цифровой преобразователь) - измеряет поданное на пин аналоговое напряжение, преобразовывает в цифровой сигнал, и передаёт в программу.
3. **ЦАП** (DAC, цифро-аналоговый преобразователь) – Преобразует цифровой сигнал в аналоговый.
4. **Таймер** (счётчик) - считает такты работы процессора.
5. **Watchdog** - данный блок позволяет перезагрузить МК, если он завис, а также выйти из спящего режима.
6. **Интерфейсы связи** - некоторые интерфейсы реализованы отдельно и работают самостоятельно, обмениваясь с ядром готовыми данными.

Как вы наверное поняли, **микроконтроллер - это микросхема с кучей ножек. У каждой ножки есть своя функция**, в частности у блоков GPIO и интерфейсов связи есть свои личные ноги.

Тактовая частота микроконтроллера

Тактовый сигнал в микроконтроллере — это сигнал, который используется для синхронизации операций внутри микроконтроллера.

По приходу логической единицы от тактового сигнала, происходит приказ процессору выполнить одну операцию.



Чем чаще приходят указания процессору «делать», тем выше скорость вычислений

Регистр – ячейка памяти

Регистры – цифровые устройства, осуществляющие хранение и преобразование многоразрядных двоичных чисел.

Регистры являются оперативными запоминающими устройствами (ОЗУ). Регистры – важный узел всех цифровых радиотехнических систем. Управляющие и запоминающие схемы, счётчики, генераторы кодов, кодопреобразователи, арифметические устройства выполняются на регистрах.

Рабочие
регистры
общего
назначения

| 7 | 0 | Адрес |
|-----|---|-------|
| R0 | | 0x00 |
| R1 | | 0x01 |
| R2 | | 0x02 |
| ... | | |
| R13 | | 0x0D |
| R14 | | 0x0E |
| R15 | | 0x0F |
| R16 | | 0x10 |
| R17 | | 0x11 |
| ... | | |
| R26 | | 0x1A |
| R27 | | 0x1B |
| R28 | | 0x1C |
| R29 | | 0x1D |
| R30 | | 0x1E |
| R31 | | 0x1F |

Счётчик команд (PC – counter)

Счётчик команд инкрементирует указатель на адрес памяти программ с каждым новым приходом «делай» от тактового сигнала, начиная с нулевого адреса.

Программист загружает код в память программ микроконтроллера во время прошивки.

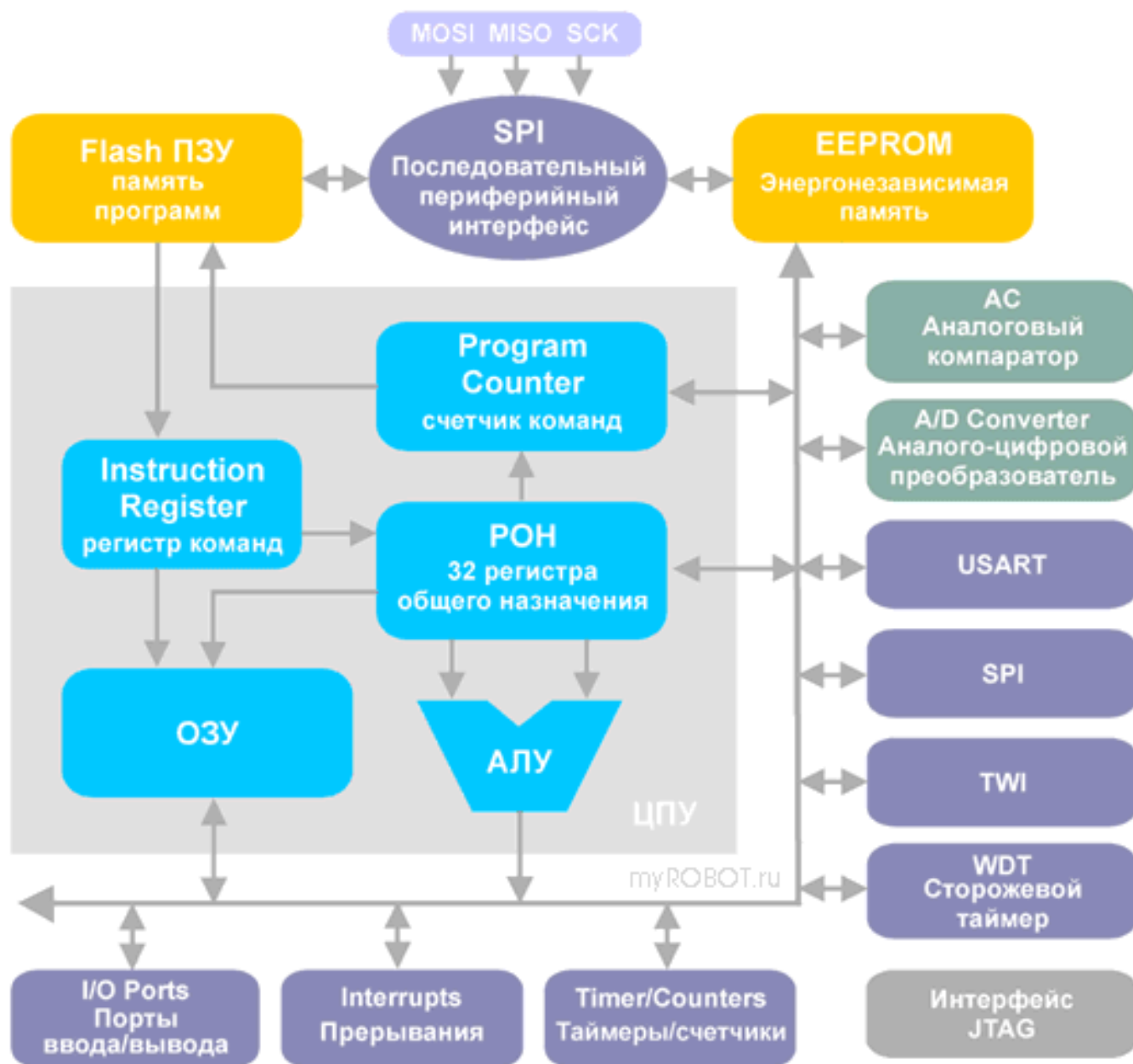
Счётчик команд указывает на адрес текущей инструкции для процессора , которую необходимо выполнить.

Регистр команд

Регистр команд - часть блока управления центрального процессора, содержащая инструкцию, которая выполняется в настоящий момент.

Он предназначен для хранения кода команды на период времени, который необходим для ее выполнения.

В регистре команд содержится информация для **устройства управления(УУ)** об адресах операндов и код операции, для того, чтобы АЛУ «поняло» что ей с этими операндами нужно сделать и где их найти в памяти.



Память программ :

(Flash ROM или Flash ПЗУ) - хранит последовательность команд, управляющих функционированием микроконтроллера.

Память данных:

Энергонезависимая память данных (EEPROM)-

Удобна для хранения промежуточных данных, различных констант, коэффициентов.

Оперативная память (ОЗУ) - предназначена для хранения пользовательских данных.

Последовательный порт UART в Arduino

UART в переводе это **универсальный асинхронный приемопередатчик**. Данные UART передаются последовательным кодом в следующем формате. данные передаются младшим битом вперед



Каждый бит передается за равные промежутки времени. Время передачи одного бита определяется скоростью передачи.
Скорость передачи указывается в бодах (бит в секунду).

Общение с компьютером

Порт UART платы Arduino UNO используется для загрузки в контроллер программы из компьютера при помощи «переходника» **USB – UART**, а также для обмена данными.



| Название сигнала | Направление | Назначение |
|------------------|-------------|--------------------------------------|
| VCCIO | выход | Питание +3,3 или +5 В, ток до 150 мА |
| GND | | Общий |
| TXD ← | выход | Передача данных |
| RXD → | вход | Прием данных |

Какой стиль предпочитаете вы?



Змеиный_стиль

Удобный, когда имя переменной короткое.
Но длинный как чёртова змея, когда слов много.

```
push_something_to_first_queue
```



СтильПаскаля

Паскаль вообще был стильным. Этот способ выглядит очень приятно, но мало где используется.

```
GetItem, SetItem, Convert
```



верблюжийСтиль

Очень распространён. С одним словом выглядит вообще отлично, но когда слов много, непонятно, почему первое обидели.

```
push, reverse, beginBuildingApp
```



шашлычный-стиль

Выглядит аппетитно и легко читается. Но любой здравомыслящий язык программирования пошлёт вас нахрен с таким стилем.

```
start-cooking-kebab, are-you-insane
```



СТИЛЬ_КРИЧАЩЕЙ_ЗМЕИ

Показывает ваше превосходство в коде. Пусть все знают, кто тут самый крутой программист! Но при чтении вытекают глаза.

```
LOOK_AT_THIS, MY_CODE_IS_AMAZING
```

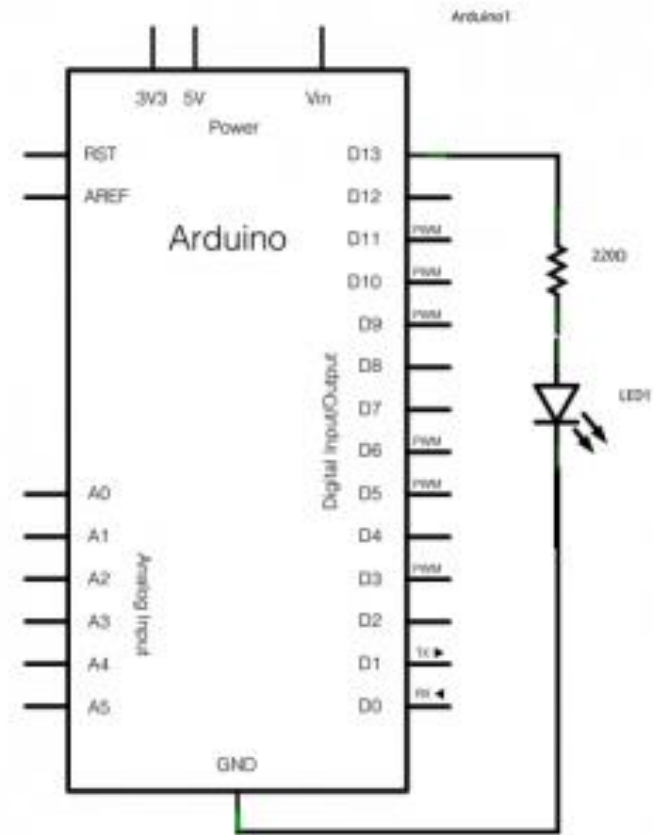
безстиля

Выглядит профессионально. Но на самом деле нет, просто остановитесь уже.

```
abstractfactoryofdestruction, ispenisbig
```

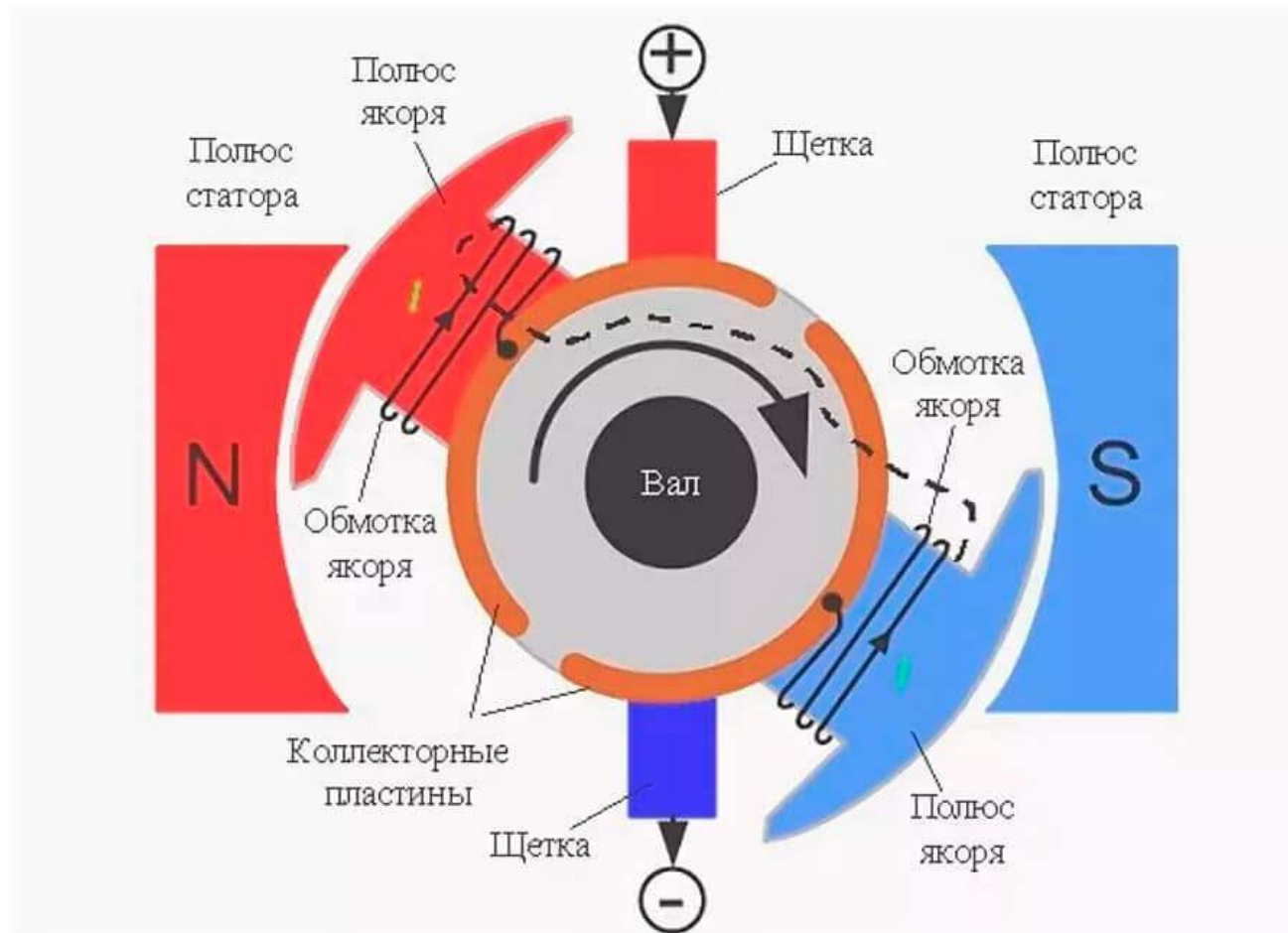

Помигаем светодиодом

```
/*  
  Зажигаем светодиод на одну секунду, затем выключаем его на  
  одну секунду в цикле.  
*/  
  
void setup() {  
  // Инициализируем цифровой вход/выход в режиме выхода.  
  // Выход 13 на большинстве плат Arduino подключен к светодиоду на плате.  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(13, HIGH);  // зажигаем светодиод  
  delay(1000);             // ждем секунду  
  digitalWrite(13, LOW);   // выключаем светодиод  
  delay(1000);             // ждем секунду  
}
```



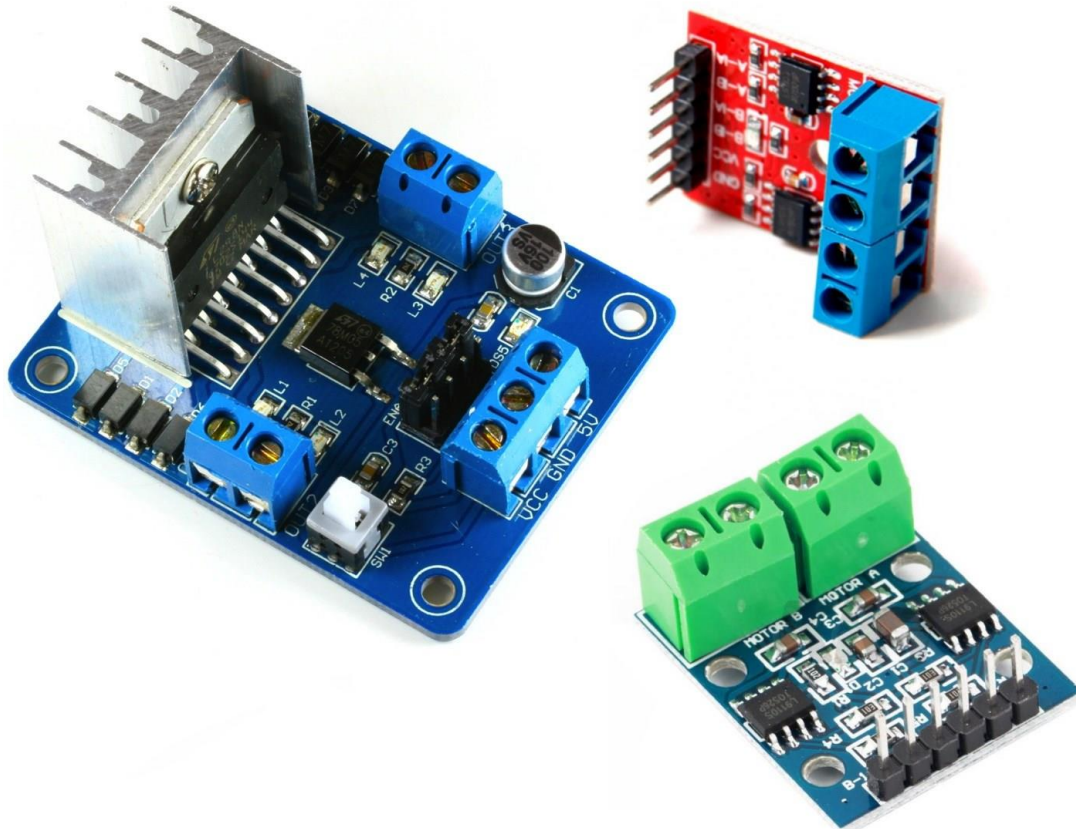
Как работает коллекторный мотор?

<https://www.youtube.com/shorts/vf32-TITMmQ>



Драйвер двигателя

Максимальный ток на выводах Arduino слаб (около 50 мА) для такой мощной нагрузки как электромотор (десятки и сотни миллиампер). Поэтому напрямую к выводам Arduino подключать электродвигатель нельзя: есть риск сжечь вывод, к которому подключён двигатель. Для безопасного подключения электродвигателей разных типов к Arduino необходим самодельный или промышленно изготовленный т.н. **драйвер двигателей**.



Также драйвер двигателей позволяет регулировать скорость вращения с помощью **ШИМ** и направление вращения

Драйвер двигателя L298N



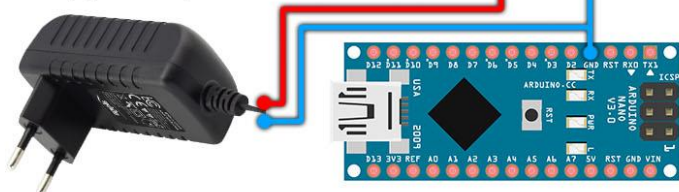
Драйвер L298n

- Напряжение: 4-50V
- Ток (макс): 1A (2A)

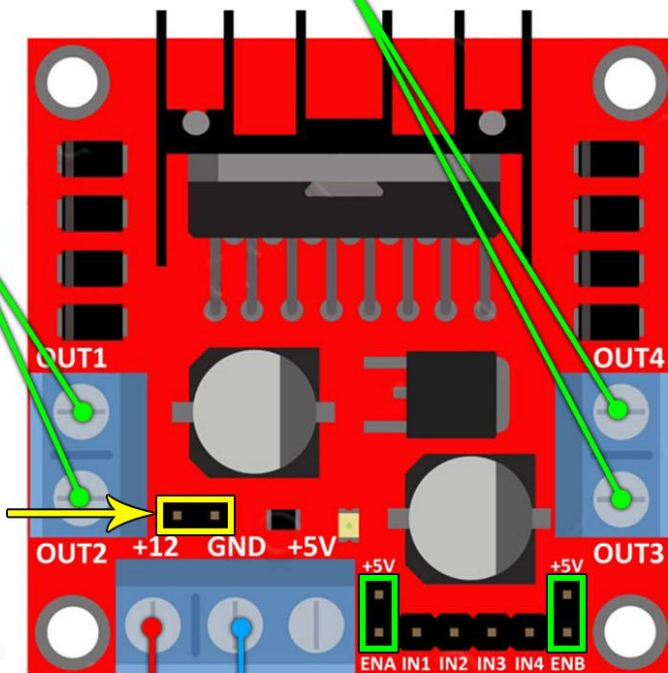
Мотор 1

Если перемычка установлена, драйвер можно питать от 6.. 12V. Пин +5V в этом случае является выходом 5V, можно использовать в своих целях (питать Ардуино).

Если перемычка не установлена, драйвер можно питать от 6.. 35V. Пин +5V в этом случае является входом, на который нужно подать 5V для питания драйвера!



Мотор 2



Пины IN отвечают за подачу сигнала, который задаётся пином EN. В то же время можно поставить перемычку и управлять скоростью при помощи ШИМ в один из IN.

Если перемычка EN-5V установлена

| | Мотор 1 | | Мотор 2 | |
|----------|----------|------|----------|------|
| | IN1 | IN2 | IN3 | IN4 |
| Вперёд | HIGH/PWM | LOW | HIGH/PWM | LOW |
| Назад | LOW/PWM | HIGH | LOW/PWM | HIGH |
| Холостой | LOW | LOW | LOW | LOW |
| Тормоз | HIGH | HIGH | HIGH | HIGH |

Если перемычка EN-5V не установлена

| | Мотор 1 | | | Мотор 2 | | |
|----------|---------|------|-----|---------|------|-----|
| | IN1 | IN2 | ENA | IN3 | IN4 | ENB |
| Вперёд | HIGH | LOW | PWM | HIGH | LOW | PWM |
| Назад | LOW | HIGH | PWM | LOW | HIGH | PWM |
| Холостой | LOW | LOW | 0 | LOW | LOW | 0 |
| Тормоз | HIGH | HIGH | PWM | HIGH | HIGH | PWM |

https://3d-diy.ru/blog/drayer-dvigatelya-l298n/?srsId=AfmBOoq6UjVegDdONko9AtfeLNyyo_1gCYcs29vq68CdPPdddUOqzU_m

Управление двигателем

<https://alexgyver.ru/lessons/dc-motors/>

https://3d-diy.ru/blog/drayver-dvigatelya-l298n/?srsltid=AfmBOoq6UjVegDdONko9AtfeLNyyo1gCYcs29vq68CdPPdddUOqzU_m

Пример программы

```
// задаем имена для портов
#define IN1 4
#define IN2 5
#define IN3 6
#define IN4 7

void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}

void loop() {
  // вращаем моторчики в одну сторону
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);

  delay(2000); // ждем 2 секунды

  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);

  delay(1000); // выключаем на 1 секунду
```

```
// вращаем моторчики в обратную сторону
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
digitalWrite(IN3, LOW);
digitalWrite(IN4, HIGH);

delay(2000); // ждем 2 секунды

digitalWrite(IN1, LOW);
digitalWrite(IN2, LOW);
digitalWrite(IN3, LOW);
digitalWrite(IN4, LOW);

delay(1000); // выключаем на 1 секунду
}
```


Отправка и парсинг Serial

<https://alexgyver.ru/lessons/parsing/>

<https://mypractic.ru/urok-12-posledovatelnyj-port-uart-v-arduino-biblioteka-serial-otladka-programm-na-arduino.html>