ДАТЧИК И ПРИВОД

Содержание

- Датчик
  - Измерение расстояния с помощью ультразвукового датчика
- Привод
  - Управление DC Мотором
  - Управление Сервомотором

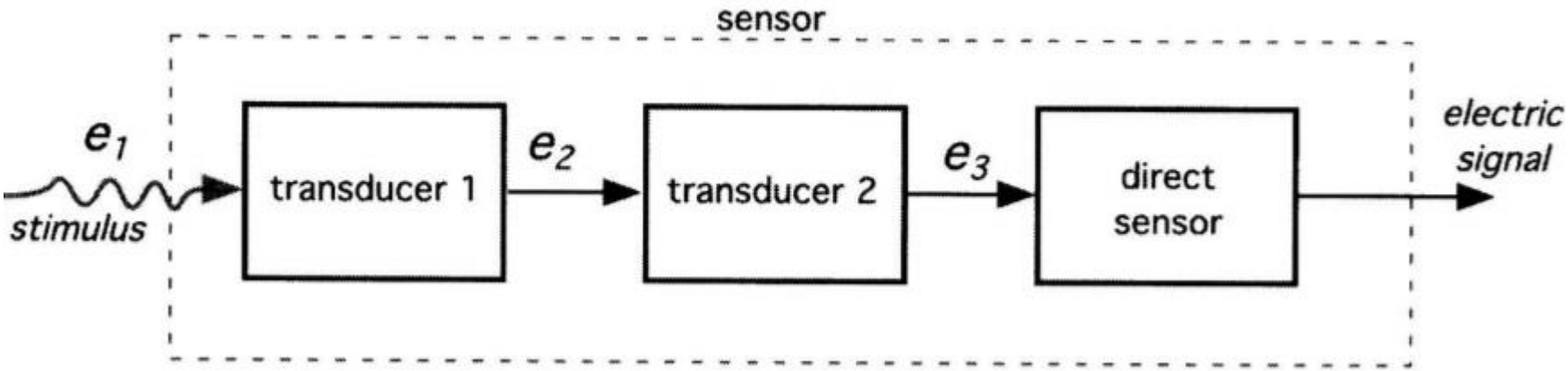
Sensor (Датчик)

- Датчик - это устройство, модуль или подсистема, целью которых является обнаружение событий или изменений в его среде и отправка информации в другую электронику, часто в компьютерный процессор.
- Датчики - это устройства, преобразующие любые физиче ские атрибуты (температура, яркость, сила, ускорение и т. Д.) В понятную форму для людей или машин.

Sensor (Датчик)

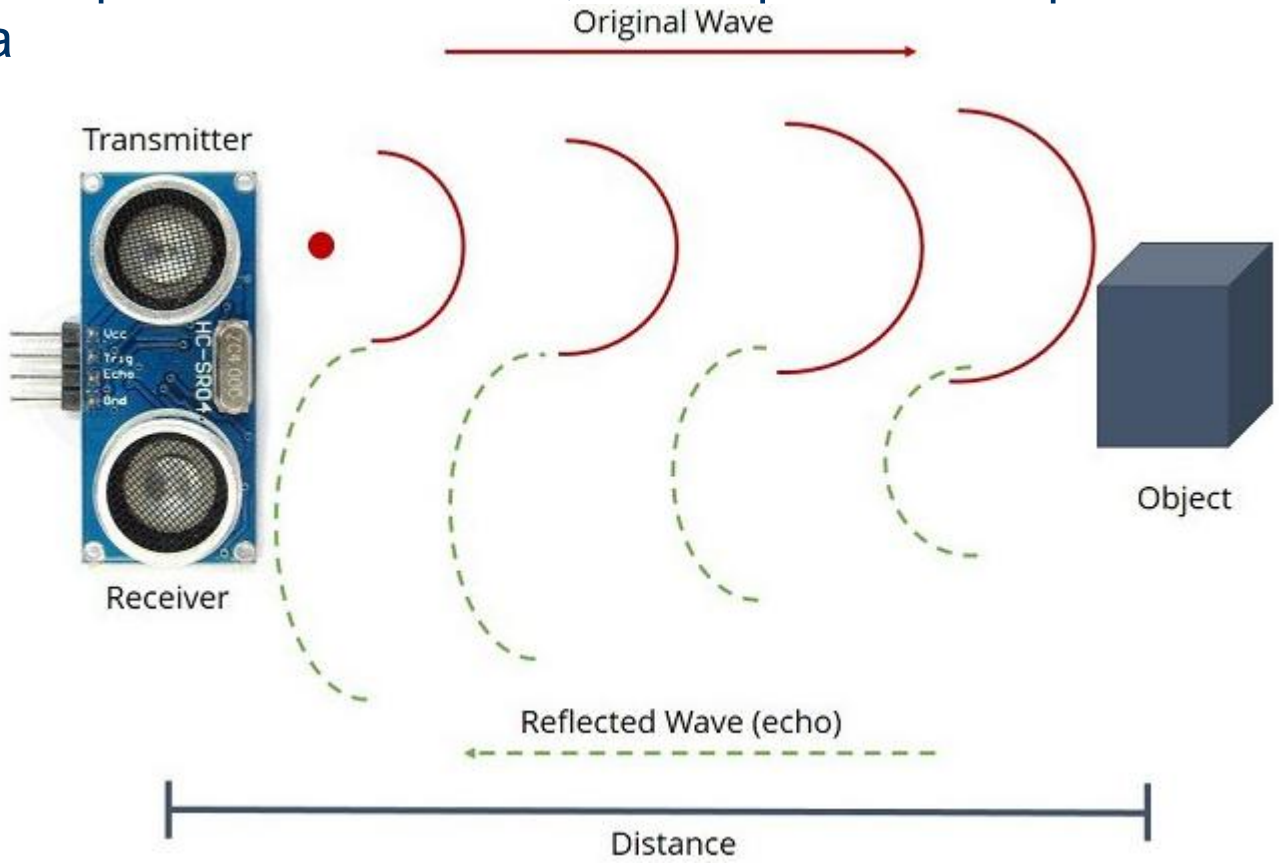
- Датчик приобретает физическую величину и преобразует его в сигнал, подходящий для обработки
- Общие датчики преобразуют измерение физических явлений в электрический сигнал
- Датчик называется преобразователем,

  который преобразует одну форму энергии в другую
  - Когда вход представляет собой физическую величину и выходной электрический ток → Sensor (датчик)



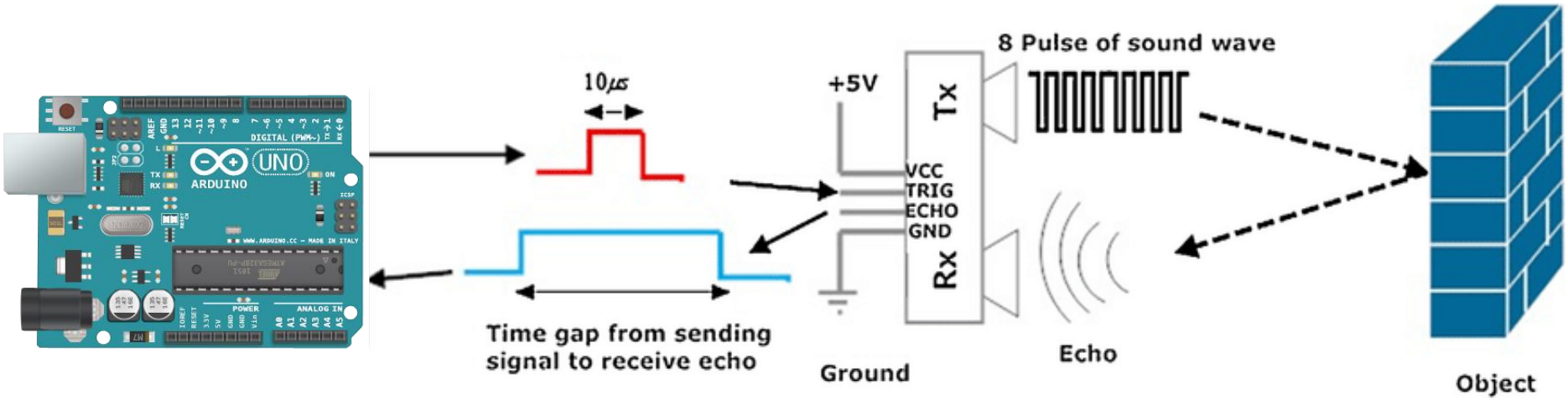
Ультразвуковой датчик(HC-SR06)

- Ультразвуковые датчики генерируют высокочастотные звуковые волны и оценки эхо-сигнала, который отражается обратно к датчику. Микросхема времени измеряет интервал времени между передачей сигнала и приема эхо-сигнала, чтобы рассчитать расстояние до объекта

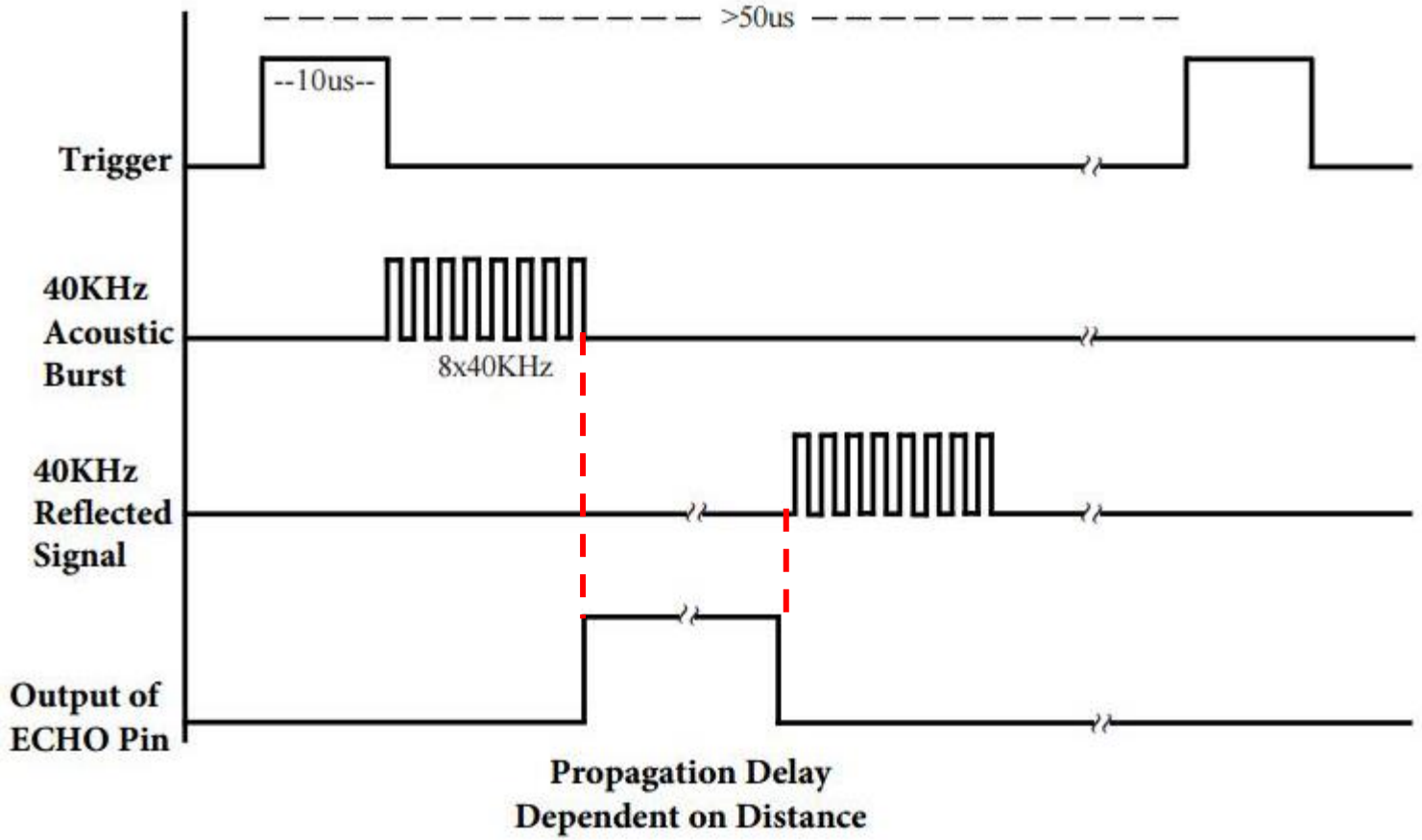


Принцип действия

- Основной принцип работы:
  1. Отправляет запускающий импульс в течение по крайней мере 10us к ультразвуковому модулю
  2. Модуль HC-SR04 автоматически посылает восемь до 40 кГц и обнаружить, есть ли импульсный сигнал обратно
  3. Если сигнал возвращается, ЭХО выход датчика будет в высоком состоянии (5В) в течение длительной времени, необходимой для передачи и приема ультразвукового взрыва.



Временная диаграмма



- После ультразвукового взрыва передается ECHO, pin переходит на высокий уровень и остается высоким, пока волна не возвращается.
- Для того, чтобы получить расстояние, нужно измерить ширину ECHO pin

Измерение ширины Pin

- pulseIn()
  - Считывает пульс (высокий или низкий) с pin
  - Синтаксис
    - pulseIn(pin, value)
      - pin: количество pin, на котором вы хотите прочитать пульс (int)
      - value: тип импульса следующим образом: HIGH или LOW. (int)
  - Возврат (unsigned long)
    - длина импульса (в микросекундах)
    - 0, если импульс не начался до истечения времени ожидания

Если значение HIGH, pulseIn() ждет pin чтобы перейти от LOW до HIGH начинается отсчет времени затем ждет pin , чтобы перейти на низкий уровень и останавливает отсчет времени.

Возвращает длину импульса в микросекундах или

дает и возвращает 0, если не полный импульс не был получен в течение тайм-аута

Расчет расстояний

- найти расстояние до объекта

: Расстояние = «скорость звука (344 млн / сек) x Время в» / 2

Расстояние в сантиметрах = Время / 2 * 0,0344 (или времени / 58)

Время = ширина эхо-импульса, в нас (микро секунд)

Скорость звука

: Скорость звука в воздухе с изменением температуры и влажности

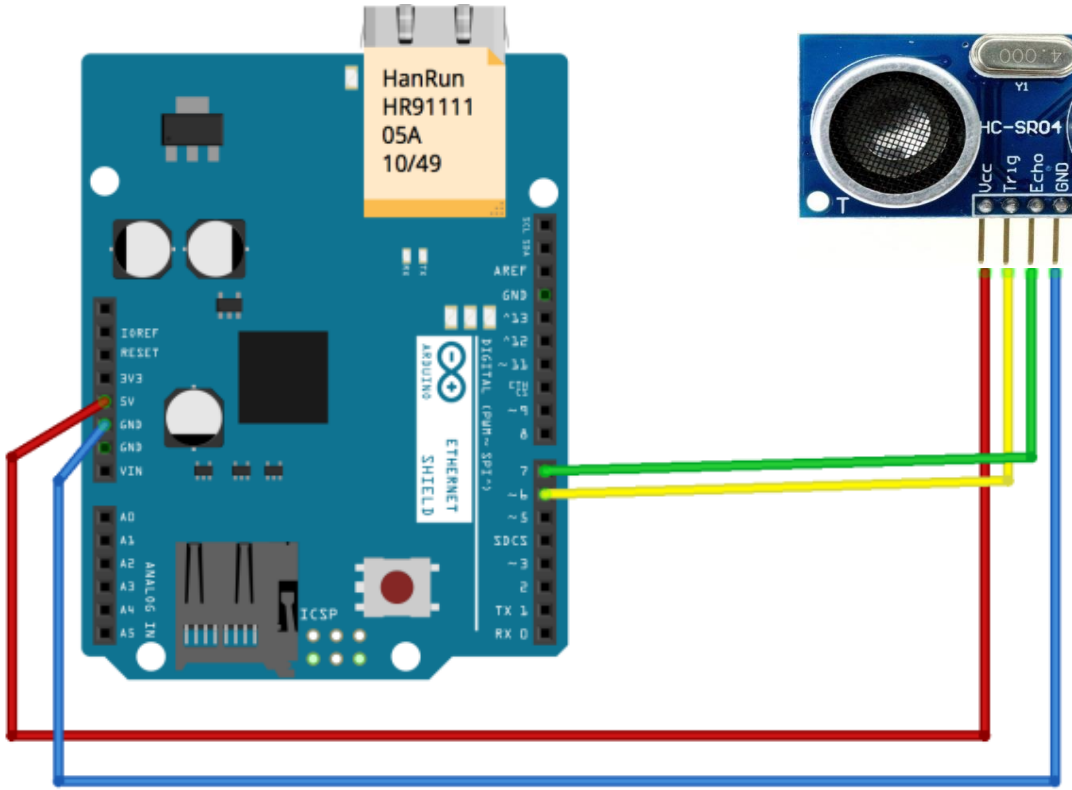
c(m/s) = 331.4 + (0.606 x T) + (0.0124 x H)

331.4: Скорость звука (в м / с) при влажности 0 ℃ и 0%

T: температура в ℃

H: % huminity

Соединение

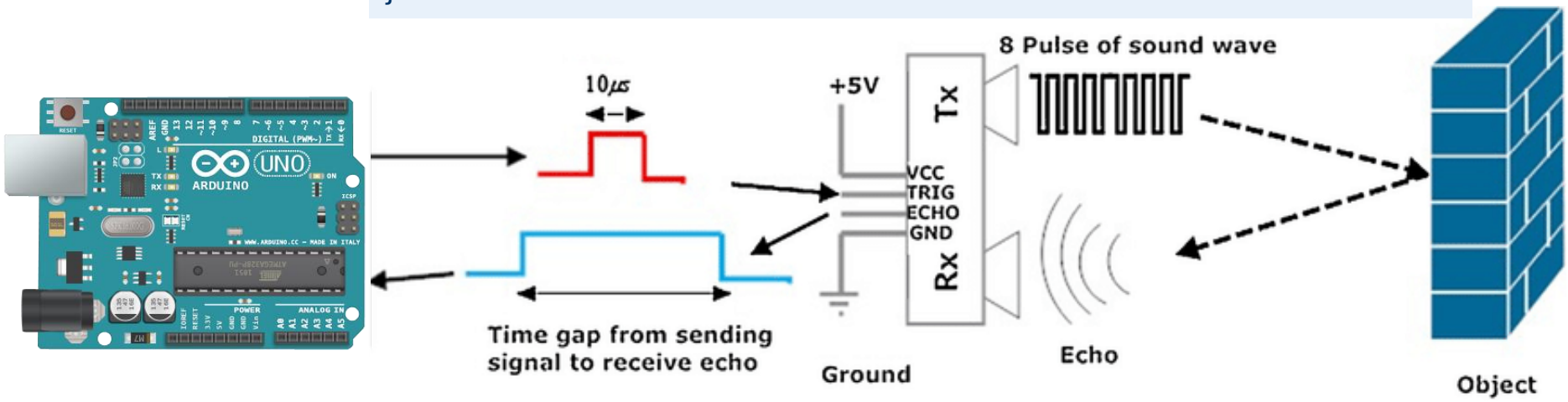


❑ Pin соединение

Arduino	Ультразвуковой датчик
5V	VCC
Pin 3 (Output)	Trig
Pin 2 (Input)	Echo
GND	GND

Пример кода


```
void setup() {
   pinMode(trigPin, OUTPUT);          // sets the trigPin as output
   pinMode(echoPin, INPUT);           // sets the echoPin as input
}
void loop() {
    float duration, distance;

    digitalWrite(trigPin, HIGH);      // the trigPin is at 5 volts
    delayMicroseconds(10);            // 10us delay
    digitalWrite(trigPin, LOW);       // the trigPin is at ground

   duration = pulseIn(echoPin, HIGH); // read the width of Echo pin
   distance = (duration) / 58;        // calculate the distance in cm
}
```



Пример кода

```
#define trigPin   A4
#define echoPin A5

void setup() {
   Serial.begin(9600);                // откр.т порт, устанавл.т скорость 9600 бит/сек
   pinMode(trigPin, OUTPUT);     // устанавливает в качестве выходного сигнала trigPin
   pinMode(echoPin, INPUT);      // устанавливает в качестве входных данных echoPin
}
void loop() {
    float duration, distance;

    digitalWrite(trigPin, HIGH);              // trigPin находится в 5 вольт
    delayMicroseconds(10);                    // задержка 10 мкс
    digitalWrite(trigPin, LOW);               // trigPin находится на земле

   duration = pulseIn(echoPin, HIGH);    // читать ширину Echo pin
   distance = (duration) / 58;                // вычислить расстояние в см

   Serial.println(distance);                  // расстояние дисплея
   Serial.println(“ cm”);                     // с последующим блоком (см)
   delay(500);                                // задержка 500мс
}
```

#1 задача: Измерить расстояние

❑ Task(Задание)

: Измерьте расстояние до объекта и показать соответствующие сообщения и управление LED, как показано ниже таблицы.

Диапазон расстояний	Сообщения	LED (Pin 13)
distance ≤ 2	Out of range	OFF
2 < distance ≤ 10	Obstacle	ON
10 < distance < 200	No obstacle	Blinking
distance ≥ 200	Out of range	OFF

❑ Use Tip(Подсказка)

- Печать сообщения через последовательный порт с Serial.print() и Serial.println()
- Написать HIGH или LOW значение в13 pin с pinMode() и digitalWrite()

#1-a задача: функция измерения расстояния

- Task(Задание)

  : Напишите функцию для измерения расстояния , используя код с 1го задания

  Результат такой же, как на 1-й задаче.

  - имя функции: Calc_dist
  - Параметр функции: ни один
  - Возврат функции: distance (Типыцелочисл)

- Use Tip(Подсказка)

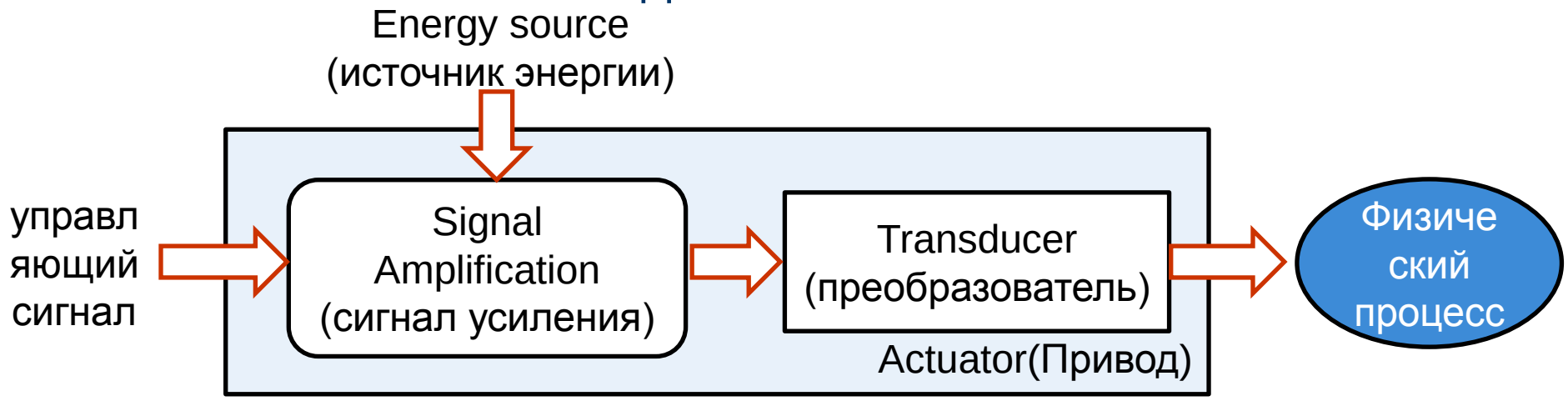
```
int Calc_dist()
{
   ….
   return (int) distance;
}
```

Содержание

- Датчик
  - Измерение расстояния с помощью ультразвукового датчика
- Привод
  - Управление DC Мотором
  - Управление Сервомотором

Привод

- **Привод** представляет собой компонент машины, которая отвечает за перемещение и управление механизмом или системой
- Приводу требуется управляющий сигнал и источник энергии (в основном электрический сигнал, воздух, жидкости)
- Когда он получает управляющий сигнал, исполнительный механизм реагирует путем преобразования энергии сигнала в механическое движение



Типы приводов

- Механические приводы

- Пневматические приводы
  - Использует сжатый воздух в качестве движущей силы

- Гидравлический привод
  - Использует гидравлическую жидкость для усиления командного сигнала контроллера
- Электрические приводы
  - Электродвигатели
    - AC мотор
    - DC мотор
    - Stepper мотор (шаговые)
  - Solenoids (электромагниты)

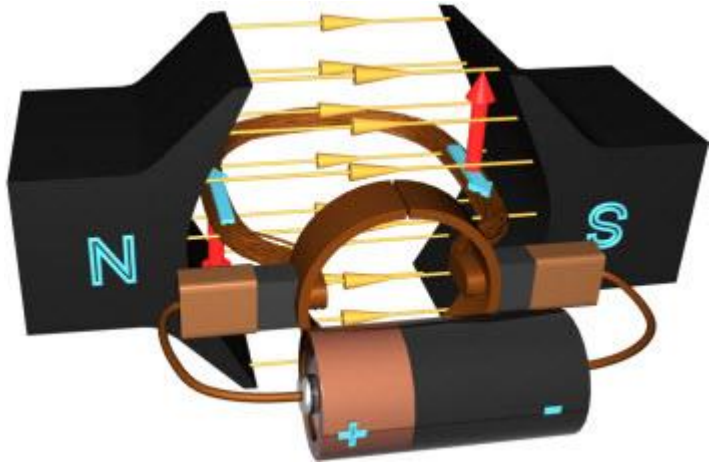
Электрические приводы

- Привод, который использует электричество для создани я механического движения
  - Управляемый двигателем, который преобразует электрическую энергию в механический крутящий момент
  - Более точные и точные
  - Может быть запрограммирован для сложных путей движения

- Использование
  - Приложения общего назначения
  - Широко используется в промышленных процессах

DC Motors (двигатель постоянного тока)

- Электромеханические устройства, которые используют в заимодействие магнитных полей и проводников для пре образования электрической энергии в вращающуюся ме ханическую энергию
- Состоит из двух частей: неподвижного тела двигателя, н азываемого **статором**, и внутренней части, которая вра щается, создавая движение, называемое **ротором** или «**арматурой**»,



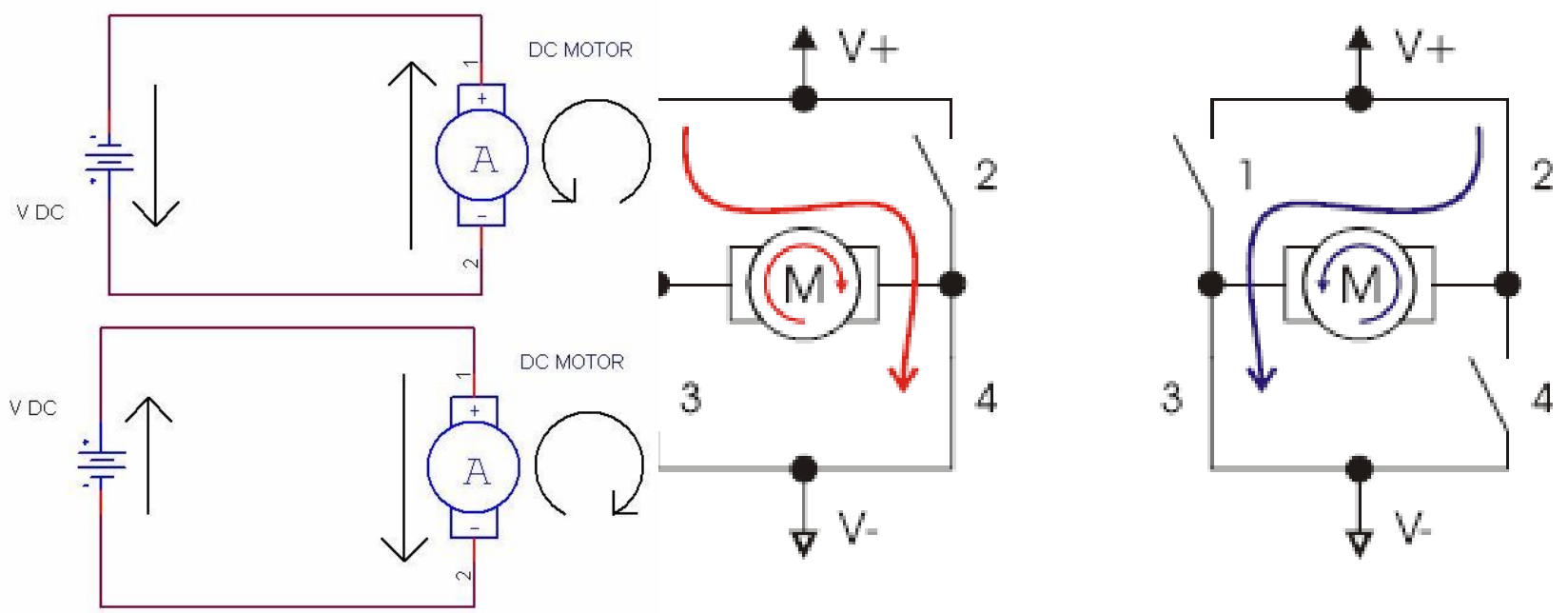
BRUSHLESS
- Longer Life
- More Power
- Quieter

BRUSHED
- Shorter Life
- Less Power
- Noisier

Управление DC Мотором (направление)

- Управление двигателем постоянного тока

: Когда мы применяем постоянное напряжение с правильным током к двигате лю, он вращается в определенном направлении, но когда мы меняем напряж ение между двумя клеммами, двигатель вращается в другом направлении

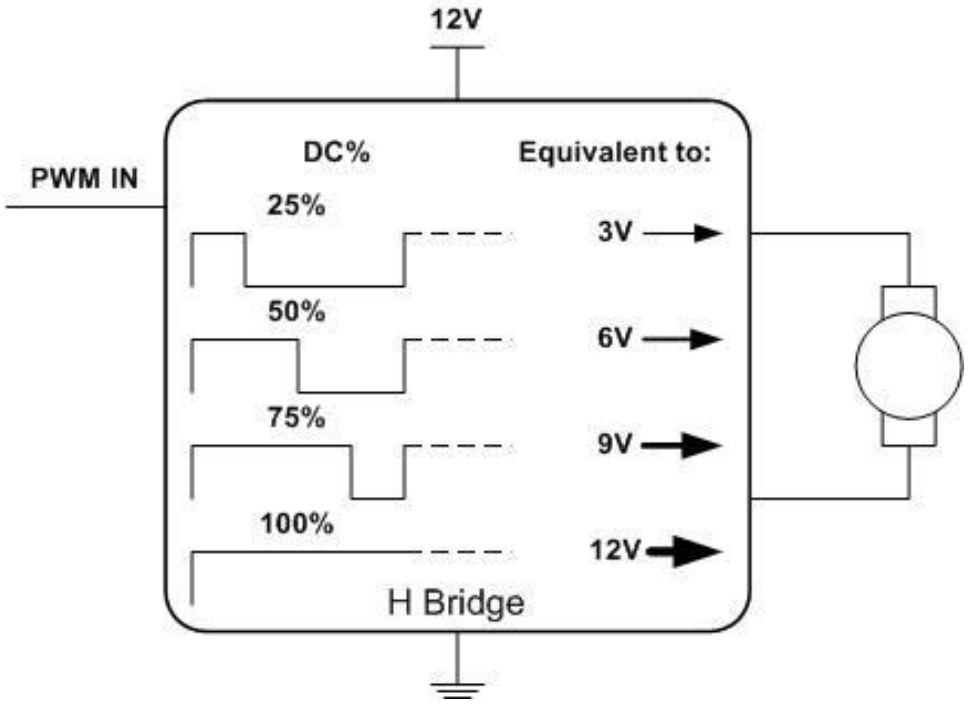


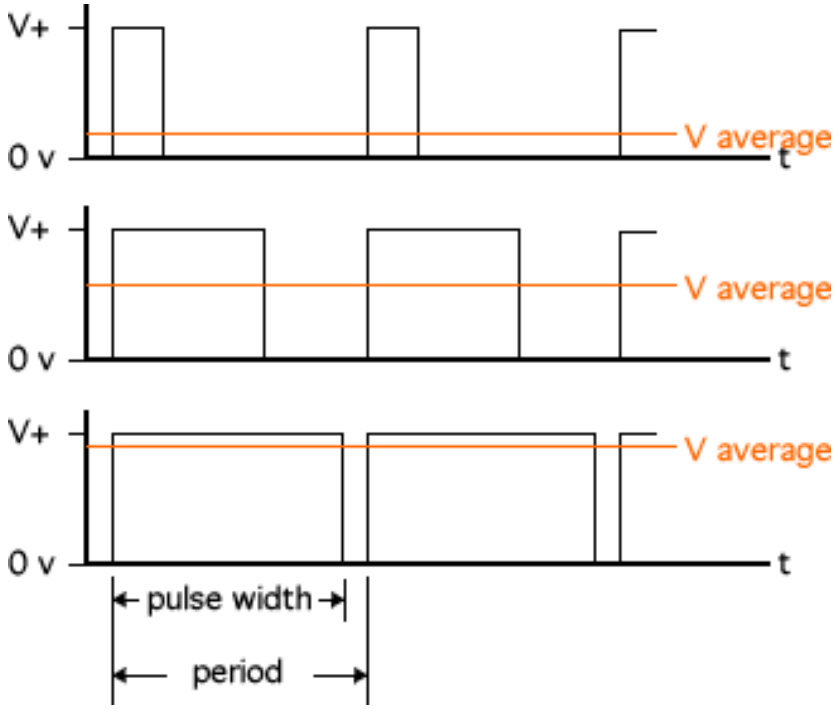
Н - специальная схема, которая позволяет вращать двигатель в обоих на правлениях

Управление DC Мотором (скорость)

- Регулятор скорости двигателя постоянного тока

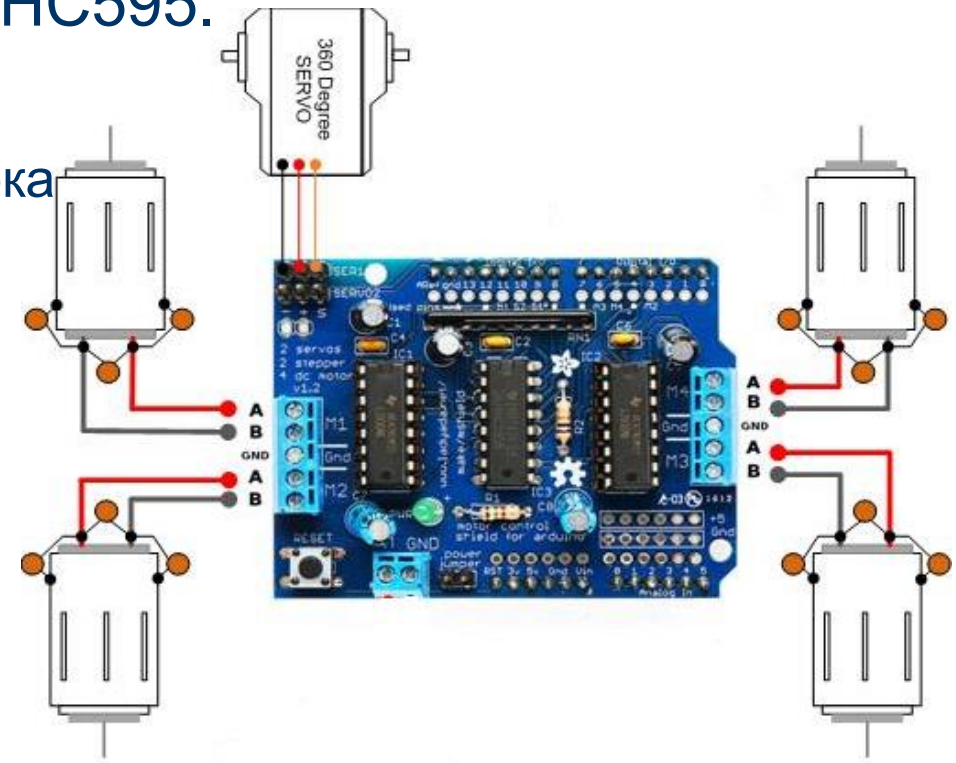
  : Рабочий цикл будет прямо пропорционален результирующему напряжению, приложенному к нагрузке. И на двигателе постоянного тока наложенное напряжение прямо пропорционально скорости двигателя.



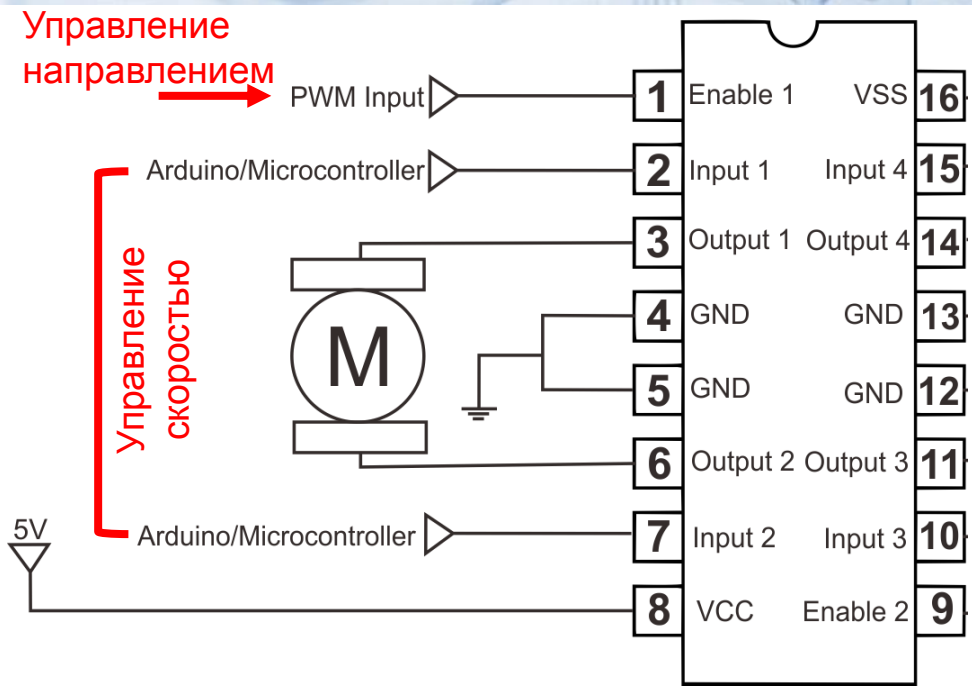


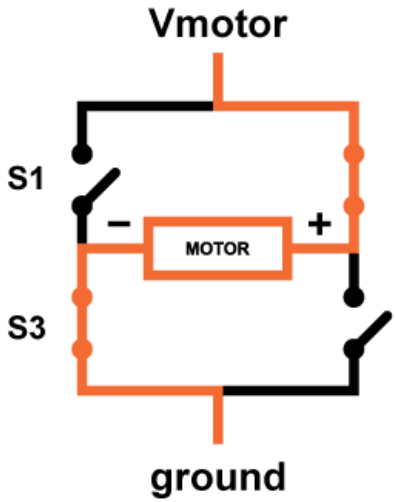
Моторный щит Adafruit

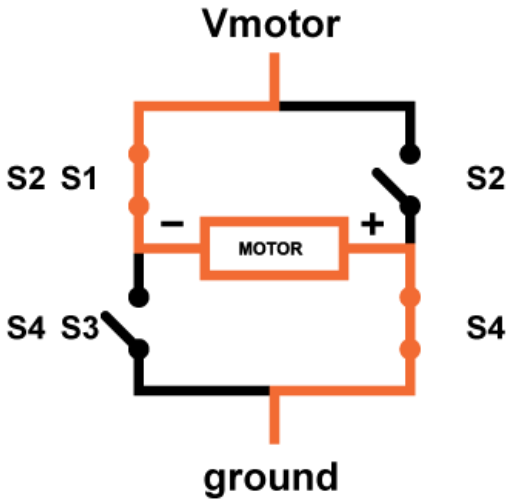
- Adafruit Motor Shield - отличный и быстрый способ управ ления двигателями постоянного тока, сервомоторами ил и даже шаговыми двигателями.
- Экран содержит два двигателя L293D и один сдвиговый регистр 74HC595.
- Управление
  - 4 двигателей постоянного тока
  - двух шаговых двигателей
  - 2 сервоприводом



Управление DC Мотором с L293D







Enable1	Input1	Input2	Управление Мотором
HIGH	LOW	HIGH	Повернуть в направлении по часовой стрелке
HIGH	HIGH	LOW	Повернуть в направлении против часовой стрелки
HIGH	LOW	LOW	Стоп
HIGH	HIGH	HIGH	Стоп
LOW	X	X	Стоп

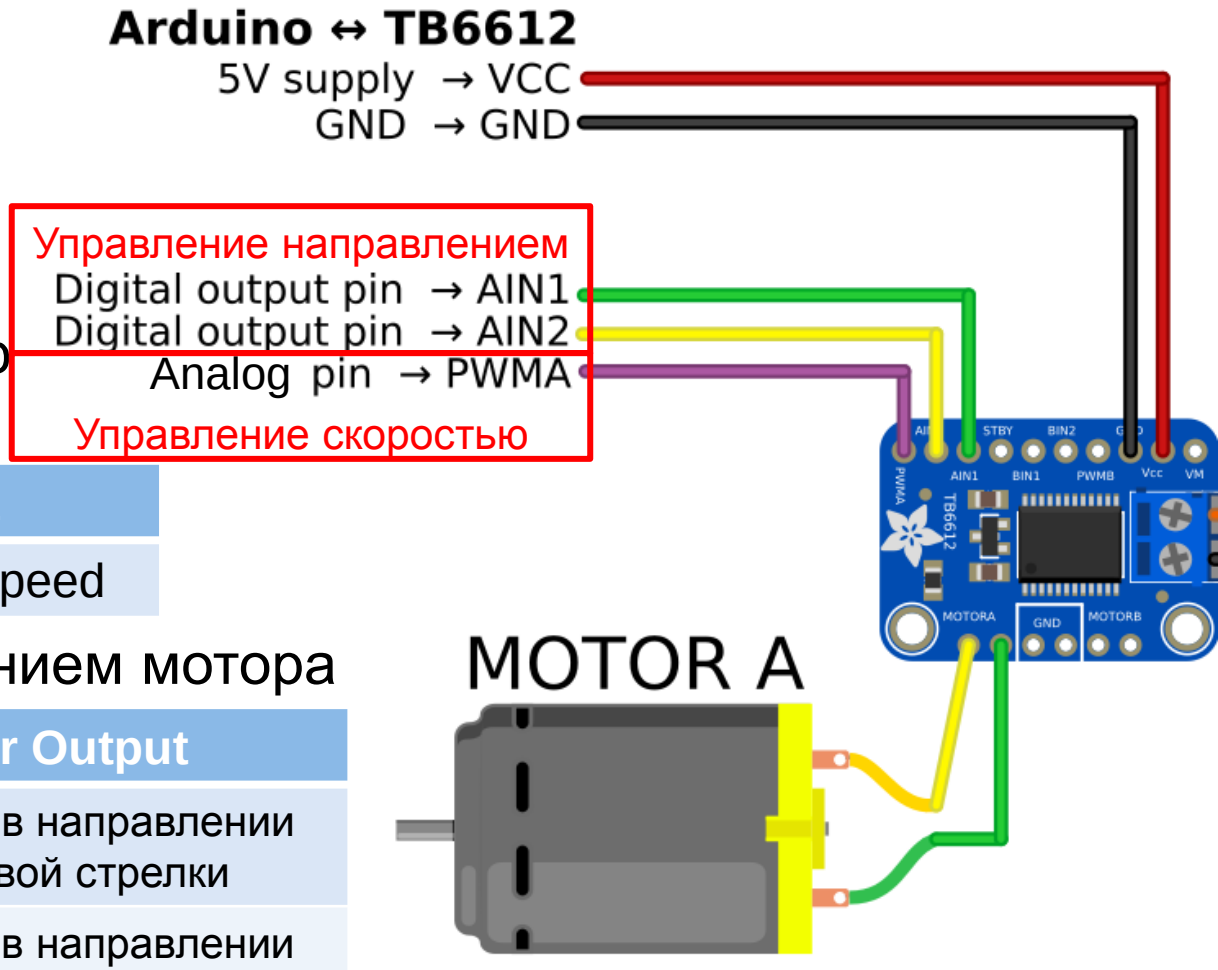
Управление DC Мотором с помощью AFMotor

- 4 шага в вашем эскизе:
  1. Включите библиотеку в свой код: **#include <AFMotor.h>**
  2. Создайте объект AF_DCMotor классом **AF_DCMotor(*motor#*)**
  3. Установите скорость двигателя с помощью **setSpeed(*speed*)** где ***скорость*** колеблется от 0 (остановл) до 255 (полная скорость)
  4. Для запуска двигателя вызовите **run(*direction*)** , где ***direction***
     - **FORWARD**
     - **BACKWARD**
     - **RELEASE**

Примерный код для L293D

```
#include <AFMotor.h>                    // include motor driver library

AF_DCMotor  motor1(1);                  // создать объект motor #1
AF_DCMotor  motor2(2);                  // создать объект motor #2
AF_DCMotor  motor3(3);                  // создать объект motor #3
AF_DCMotor  motor4(4);                  // создать объект motor #4

void loop() {
   motor1.setSpeed(200);                // установить скорость 200/255
   motor2.setSpeed(200);                // установить скорость 200/255
   motor3.setSpeed(200);                // установить скорость 200/255
   motor4.setSpeed(200);                // установить скорость 200/255

   motor1.run(FORWARD);                  // повернуть его вперед
   motor2.run(FORWARD);                  // повернуть его вперед
   motor3.run(FORWARD);                  // повернуть его вперед
   motor4.run(FORWARD);                  // повернуть его вперед
   delay(1000);
}
```

Управление двигателем с TB6612



- Управление скоростью двигателя

PWM	Motor Output
0~255	0 is off, 255 is full speed

- Управление направлением мотора

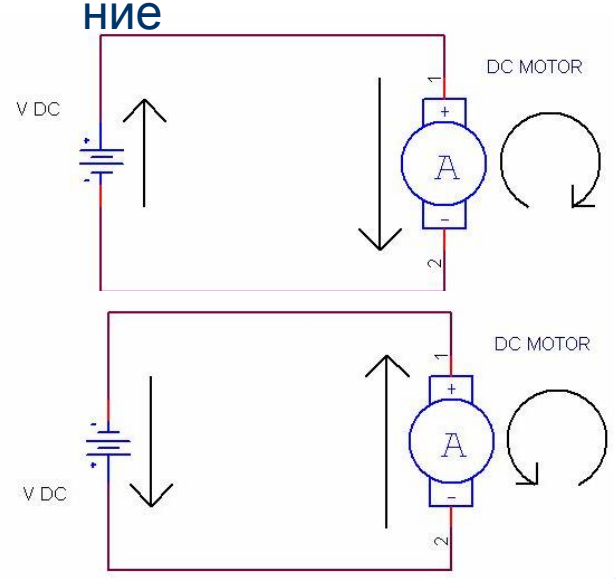
Input1	Input2	Motor Output
LOW	HIGH	Повернуть в направлении по часовой стрелки
HIGH	LOW	Повернуть в направлении против часовой стрелке
All HIGH or LOW		Стоп

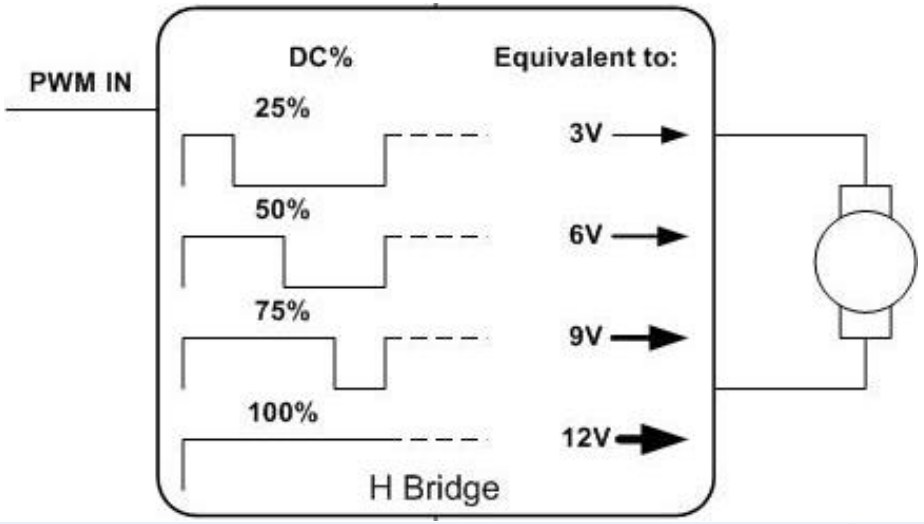
Примерный код для TB6612

```
#define PWMA   6                  //Left Motor Speed pin (ENA)
#define AIN1   A1                 //Motor-L backward (IN1)
#define AIN2   A0                 //Motor-L forward (IN2).

#define PWMB   5                  //Right Motor Speed pin (ENB)
#define BIN1   A2                 //Motor-R forward (IN3)
#define BIN2   A3                 //Motor-R backward (IN4)

#define SPEED 100                 // motor Speed

analogWrite(PWMA,SPEED)           // Set the speed on MotorA
digitalWrite(AIN1, LOW)           // Move MotorA forward
digitalWrite(AIN2, HIGH)          // Move MotorA forward

analogWrite(PWMB,SPEED)           // Set the speed on MotorB
digitalWrite(BIN1, LOW)           // Move MotorB forward
digitalWrite(BIN2, HIGH)          // Move MotorB forward
```

#2 задача: управление мотором

- Task(Задание)
  1. Напишите код для перемещения робота или машину назад
  2. Напишите код, который медленно вращает робота или машину влево или вправ
- Use Tip(Подсказка)
  - Правильно подайте напряжение постоянного тока на два цифровых контакта
  - Установите скорость двигателя, применяя соответствующее напряже ние





#3 задача: управление мотором по Диапазону

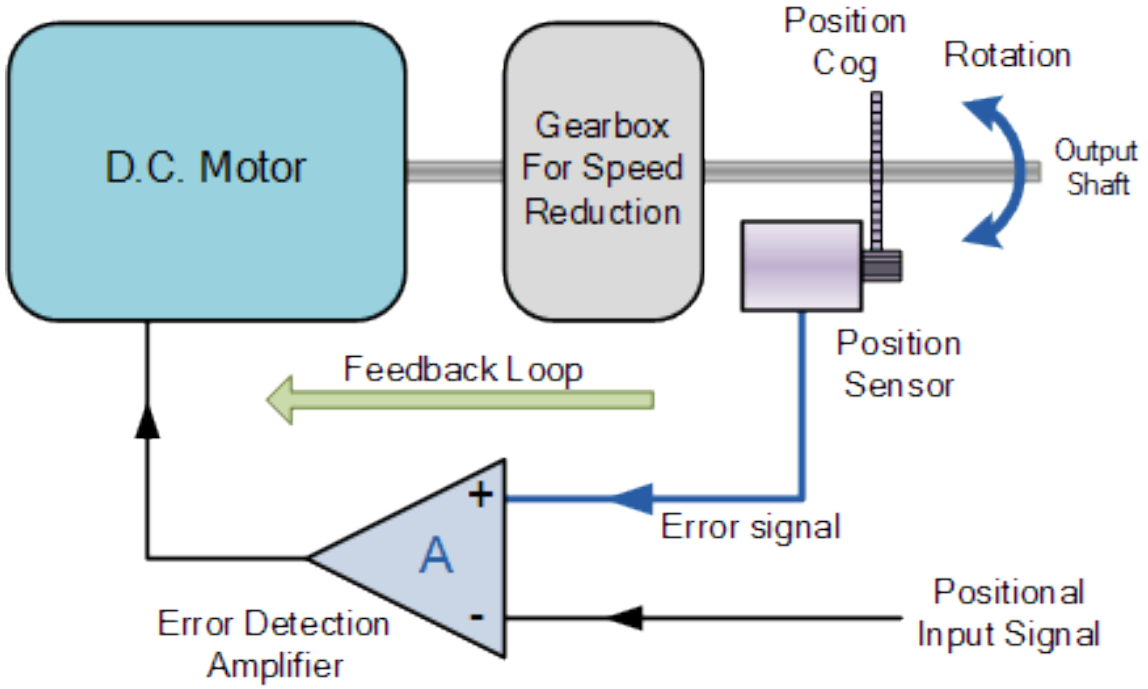
- Task(Задание)

  : В соответствии с диапазоном, управляйте мотором и покажите соответствующие сообщения в таблице ниже.

Расстояние	Мотор	Сообщения
distance ≤ 5	Стоп	Stop
5 < distance ≤ 20	Медленно вперед	Obstacle
distance ≥ 20	Быстро вперед	No obstacle

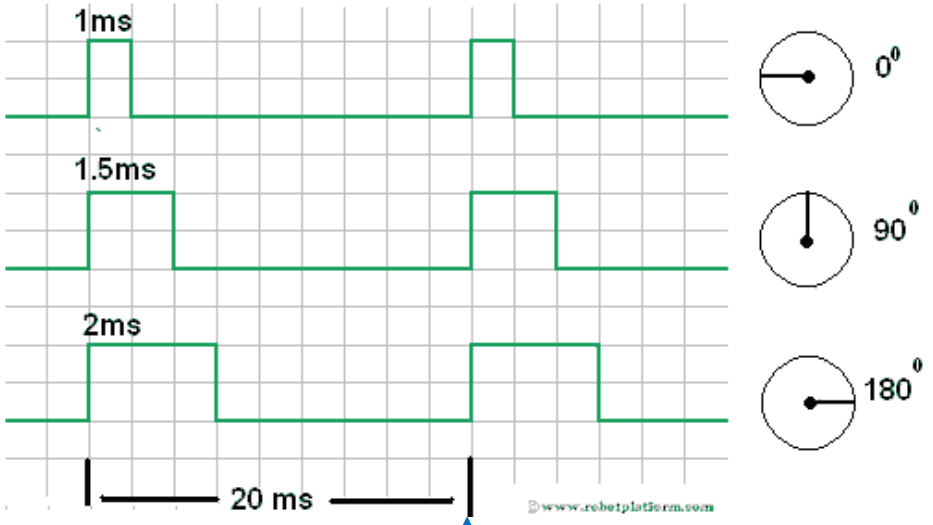
DC Сервомотор

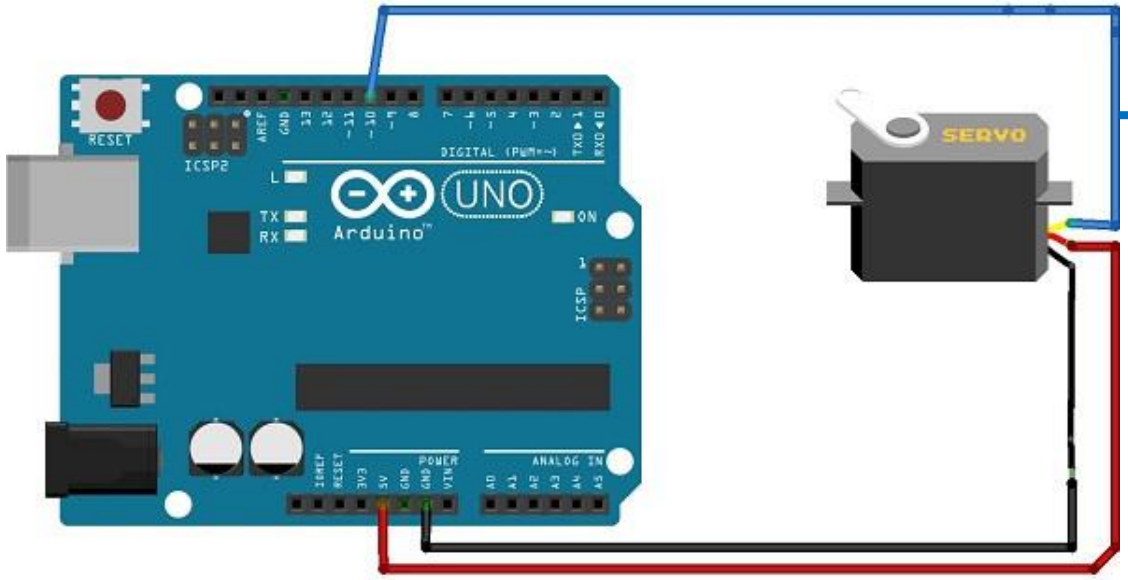
- Сервомотор состоит из нескольких устройств в одной упаковке , двигателя постоянного тока, коробки передач, устройства обратной связи и исправления ошибок.
- Легко управляется с использованием всего трех проводов, управления мощностью, заземлением и сигналом. Угол повор ота контролируется длительностью применяемого импульса для PIN-кода сигнала.



Управление сервомотором с Arduino

- Данные положения к управлению должны быть отправлены в виде сигнала PWM через сигнальный вывод серводвигателя





Signal pin

- Серво проверяет импульс каждые 20 мс
- Пульс PWM на пульте управления
  - ширина 1 мс может вращать сервопривод до 0 °
  - ширина 1 мс может вращать сервопривод до 0 °
  - ширина 2 мс может вращаться до 180°

Программное обеспечение для сервомотора

- 4 шага в вашем эскизе:
  1. Включите библиотеку в свой код: **#include <Servo.h>**
  2. Создайте объект с классом **Servo**
  3. Назначьте pin на Arduino сервомотору, используя **attach(*pin No.*)**
  4. Запишите положение серводвигателя, используя **write(*angle*)**

     где ***angle*** *(*угол) составляет от **0 до 180 градусов**

Примерный код

```
#include <Servo.h>

Servo myservo;                          // create a servo

void setup() {
    myservo.attach(10);                 // assign 10 pin on the board  to servo motor
}

void loop() {
    myservo.write(90);                  // set the position of servo motor to center
    delay(20);                          // delay time for moving of servo motor
}
```

#4 задача: управление сервомотор

- Task(Задание)
  1. Установите положение сервомотора от 0 до 180 градусов
     - Увеличивайте на 1 градус каждую задачу
     - Применять время задержки 20 мс для перемещения серводвигателя после отправки команды для увеличения на 1 градус
  2. После перемещения позиции до 180 градусов переместите пол ожение в центр