**Подключение сервоприводов к Arduino**

Как уже говорилось, сервопривод это точный исполнитель который получая на вход значение управляющего параметра стремится создать и поддерживать значение на выходе исполнительного элемента.

В данной статье рассмотрим что же из себя представляют управляющие импульсы, а также то, как лучше подключать сервоприводы к Arduino.



О том как входные импульсы преобразуются в сигналы управления мотором мы уже рассказали в этой [статье](http://zelectro.com.ua/servoFirst). В данной же статье речь пойдет непосредственно о управляющих импульсах, будут даны примеры как их сгенерировать на Arduino.

Управляющий сигнал представляет из себя импульсы с нужной нам шириной, который посылается с определенной частотой. Для рассматриваемых нами сервоприводов частота посылания импульса почти всегда будет около 50 Гц (это примерно 1 раз в 20мс), а ширина импульса будет лежать в пределе от 544мкс до 2400мкс.

Как видно из картинке, импульс шириной в 544мкс выставит выводной вал в положение 0°, 1520мск соответствует углу в 90°, а 2400мкс соответствуют 180°.



Изменяя ширину импульсов в данных приделах мы сможем точно задавать угол поворота выводного вала, но об этом чуть позже. На данном этапе статьи хочется рассказать о том как подключить сервопривод к Arduino.

Для подключения к контроллеру от сервопривода тянется 3 провода обжатых стандартным 3 пиновым разъемом с шагом 2.54мм . Цвета проводов могут варьироваться. Коричневый или черный - земля (GND), красный - плюс источника питания (VTG), оранжевый или белый - управляющий сигнал (SIG).

**Подключение Серво :**

GND на любой из GND пинов­­­ ардуино.

VTG на + 5 вольт на ардуино.

SIG на PWM(ШИМ) вывод ардуино.

У старых Ардуин, укомплектованных мегой 8, имеется всего три ШИМ вывода (digital 9,10,11), у Ардуин укомплектованных мегой 168 или 328 их 6 (digital 3,5,6,9,10,11). Семейство Arduino MEGA имеет на своем борту целых 14 ШИМ выводов.

Мини сервы, потребляющие слабый ток, можно подключать напрямую к пинам Arduino.



Подключение мощных сервоприводов может вызвать большую просадку напряжения, контроллеру может не хватить питания, мега8 очень привередлива и из-за этого контроллер отключится. Так же на плате Arduino установлен маломощный стабилизатор не рассчитанный на потребление большого тока и чрезмерное потребление может перегреть его и повредить плату.  Во избежание этого при использовании мощных серв, либо больше одной слабой, рекомендую подавать питание на сервопривод отдельно. Можно приобрести блок питания на 5 или 6 вольт, в зависимости от напряжения питания ваше сервы, либо поставить стабилизатор. В данной статье разберем самый простой на мой взгляд стабилизатор из серии 7800, требующий минимум деталей внешней обвязки.



Стабилизатор имеет 3 ноги. первая- вход(подаем напряжение от 6 до 12 вольт) вторая- общий минус. третья выход 5 или 6 вольт в зависимости от выбранной модели.

7805 отечественный аналог КР142ЕН5А - выходное напряжение 5 вольт.

7806 отечественный аналог КР142ЕН5Б - выходное напряжение 5 вольт.

Как видно из рисунка необходима установка конденсаторов, можно и без них, но выходное напряжение будет не стабильным. Рекомендуемые номиналы конденсаторов: на входе 0.33 мкФ, на выходе 0.1 мкФ. Я всегда ставлю два по 100мкФ. Чем больше - тем лучше. И последнее и очень важное о чем стоит упомянуть так это то, что **необходимо соединить земли источников питания**.

Основы разобрали, переходим к примерам скетчей для Arduino.

Вариант 1

В данном скетче зададим 3 угла поворота выходного вала сервопривода используя управление изменением непосредственно значения ширины импульса. Данный метод самый точный, однако для каждого угла ширину импульсов придется подбирать индивидуально.

**Пример программного кода**

// добавляем библиотеку для работы с сервоприводами

#include <Servo.h>

// для дальнейшей работы назовем 9 пин как servoPin

#define servoPin 9

// 544 это стандартная длина импульса при котором сервопривод должен принять положение 0°

#define servoMinImp 544

// 2400 это эталонная длина импульса при котором сервопривод должен принять положение 180°

#define servoMaxImp 2400

Servo myServo;

void **setup**()

{

myServo.attach(servoPin, servoMinImp, servoMaxImp);

// устанавливаем пин как вывод управления сервоприводом,

// а также для работы сервопривода непосредственно в диапазоне углов от 0 до 180° задаем мин и макс значения импульсов.

// импульсы с большей или меньшей длиной восприниматься не будут.

// для сервоприводов даже одной партии значения длин импульсов могут отличаться, может быть даже и 584-2440.

// поэкспериментируйте и найдите идеальные длины импульсов конкретно для вашего сервопривода.

}

void **loop**()

{

  // устанавливаем качалку сервопривода в положение 0°(т.к.импульс равен 544мкс)

  myServo.writeMicroseconds(servoMinImp);

  delay(2000);

  // в данной функции можно задавать длины импульсов непосредственно числами.

  // 90°(т.к.импульс равен 1520мкс)

  myServo.writeMicroseconds(1520);

  delay(2000);

  // 180°(т.к.импульс равен 2400мкс)

  myServo.writeMicroseconds(servoMaxImp);

  delay(2000);

}

Вариант 2

В данном скетче зададим теже 3 угла поворота выходного вала сервопривода используя команду myservo.write. В данной команде мы уже не задаем ширину импульсов, а просто пишем нужный нам угол. Данный вариант намного удобнее, однако настройка не такая точная как при первом.

**Пример программного кода**

#include <Servo.h>

Servo myservo;

void **setup**()

{

  // устанавливаем пин как вывод управления сервой

  myservo.attach(9);

}

void **loop**()

{

  // устанавливаем угол 0°

  myservo.write(0);

  delay(2000);

  // устанавливаем угол 90°

  myservo.write(90);

  delay(2000);

  // устанавливаем угол 180°

  myservo.write(180);

  delay(2000);

}

Также вам могут понадобиться следующие команды:

**myservo.read();** Считывает текущий угол поворота сервопривода, возвращает значение типа int — угол от 0 до 180 градусов.

**myservo.attached();**
Проверяем, привязан ли сервопривод. Возвращает логическое значение bool.

**myservo.detach();**
Отключает сервопривод от пина.