Базовое сервоуправление

В этом уроке мы узнаем, как управлять стандартным серводвигателем, чтобы двигаться вперед и назад на 180 градусов, используя цикл for(). Это делается с помощью библиотеки Servo, которая является предустановленной библиотекой в Arduino IDE (как в автономной, так и в онлайн-версии).

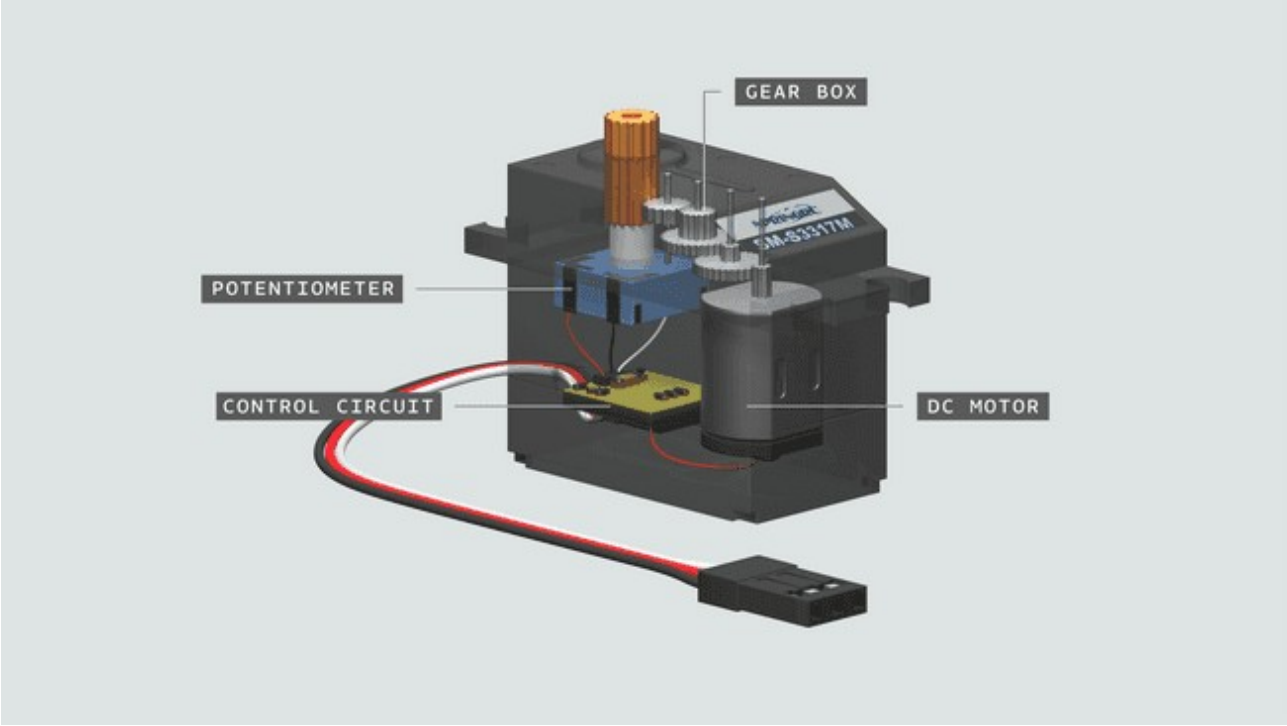
Стандартные серводвигатели

Стандартные серводвигатели — это приводы, позволяющие точно контролировать положение (угол). Типичной характеристикой является угол поворота двигателя от 0 до 180 градусов. Другими словами, он может сделать половину оборота.

Стандартный серводвигатель, как и другие двигатели, по сути представляет собой двигатель постоянного тока, но с некоторыми дополнительными функциями:

Цепь управления для управления двигателем, например, установка угла.
Шестерни, которые преобразуют скорость в крутящий момент, что позволяет ему выполнять «тяжелый подъем» на более низкой скорости, в отличие от обычного двигателя постоянного тока, который просто вращается очень быстро!
Потенциометр, который отслеживает его угол. Это позволяет сервоприводу «знать, где он находится».

Взгляните на изображение ниже, чтобы увидеть, как сервопривод выглядит внутри:



Различные провода

Почти все сервоприводы поставляются с набором из 3 проводов. Это PWR, GND и Signal. Для очень простой схемы все, что нужно, это соединить каждый из этих двух контактов на Arduino:

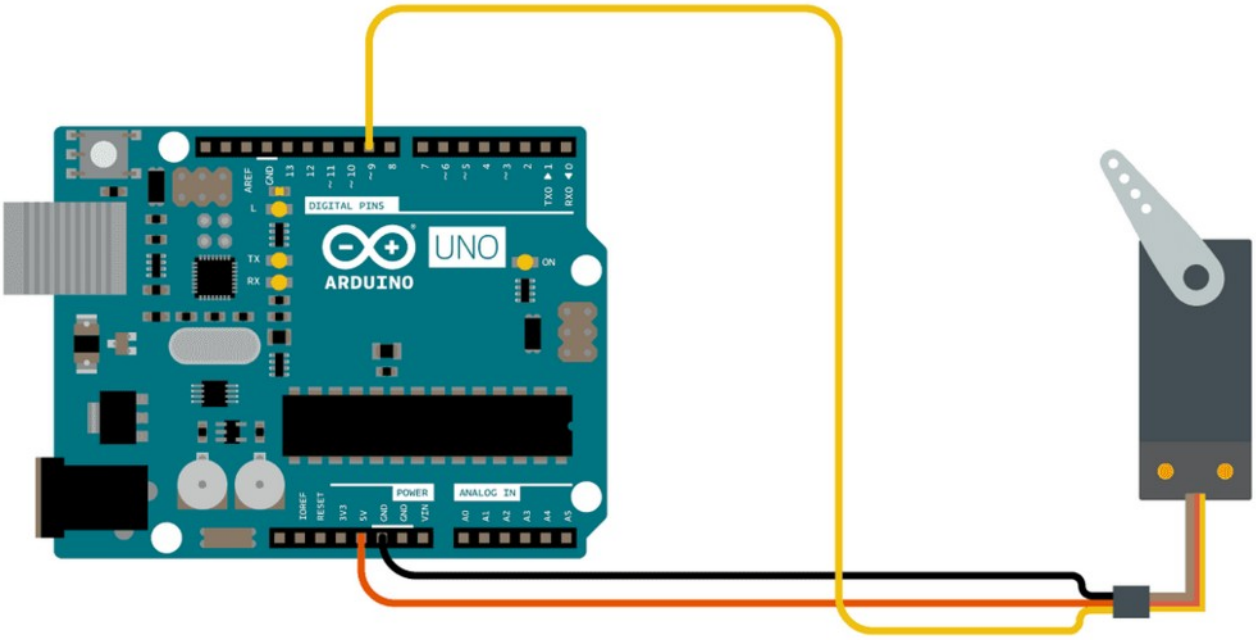
PWR (КРАСНЫЙ) — подключается к 5В на Arduino.
GND (ЧЕРНЫЙ) — подключается к GND на Arduino.
Signal (БЕЛЫЙ) — подключается к цифровому выводу на Arduino (обычно 9).

Примечание. Сочетание цветов варьируется от сервопривода к сервоприводу, но обычно остается красный и черный.

Примечание. В зависимости от того, какой Arduino вы используете, сигнальный контакт может различаться.

Схема

Просто подключите стандартный серводвигатель к Arduino, следуя приведенной ниже схеме:



Программирование платы

Чтобы запрограммировать плату, вам необходимо установить автономный редактор или использовать онлайн-редактор. Нет необходимости устанавливать какие-либо внешние библиотеки.

Прежде чем мы начнем, давайте взглянем на некоторые основные функции программы:

#include <Servo.h> - включает библиотеку Servo.

```
Servo myservo          - создать сервообъект.
myservo.attach(9)      - прикрепите сервопривод к pin.
myservo.write(pos)      - записать значение в сервопривод (0-180)
```

Код можно найти, перейдя в «Файл»> «Примеры»> «Сервопривод»> «Развертка», или его можно скопировать непосредственно снизу. Загрузите программу.

```
#include <Servo.h>

Servo myservo;  // создать сервообъект для управления сервоприводом

// двенадцать сервообъектов могут быть созданы на большинстве плат

int pos = 0;    // переменная для хранения положения сервопривода

void setup() {

  myservo.attach(9);  // прикрепляет сервопривод на контакте 9 к объекту сервопривода

}

void loop() {

  for (pos = 0; pos <= 180; pos += 1) { // идет от 0 градусов до 180 градусов

    // с шагом 1 градус

    myservo.write(pos);    // сказать сервоприводу перейти в положение в переменной
'pos'

    delay(15);               // ждет 15 мс, пока сервопривод не достигнет положения

  }

  for (pos = 180; pos >= 0; pos -= 1) { // идет от 180 градусов до 0 градусов

    myservo.write(pos);      // сказать сервоприводу перейти в положение в переменной
'pos'

    delay(15);               // ждет 15 мс, пока сервопривод не достигнет положения

  }

}
```

Тестирование

После того, как мы успешно загрузили код на плату, стандартный сервопривод теперь должен начать движение от 0 до 180, а затем начать движение от 180 до 0. Это связано с двумя циклами for в программе, которые постепенно увеличивают pos переменная, которая записывается в сервопривод.