Servo Motors Control & Arduino

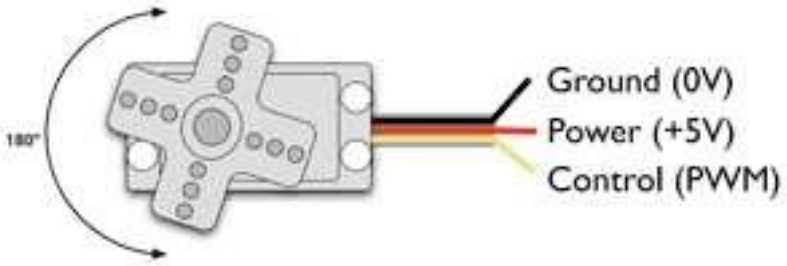
Unlike dc motors, with servo motors you can position the motor shaft at a specific position (angle) using control signal. The motor shaft will hold at this position as long as the control signal not changed. This is very useful for controlling robot arms, unmanned airplanes control surface or any object that you want it to move at certain angle and stay at its new position.

Servo motors may be classified according to size or torque that it can withstand into mini, standard and giant servos. Usually mini and standard size servo motors can be powered by Arduino directly with no need to external power supply or driver.

Usually servo motors comes with arms (metals or plastic) that is connected to the object required to move (see figure below to the right).

How Does the Servo Motor Works?

Your servo have 3 wires:
Black wire: GND (ground) **RED wire:+5v** **Colored wire: control signal**



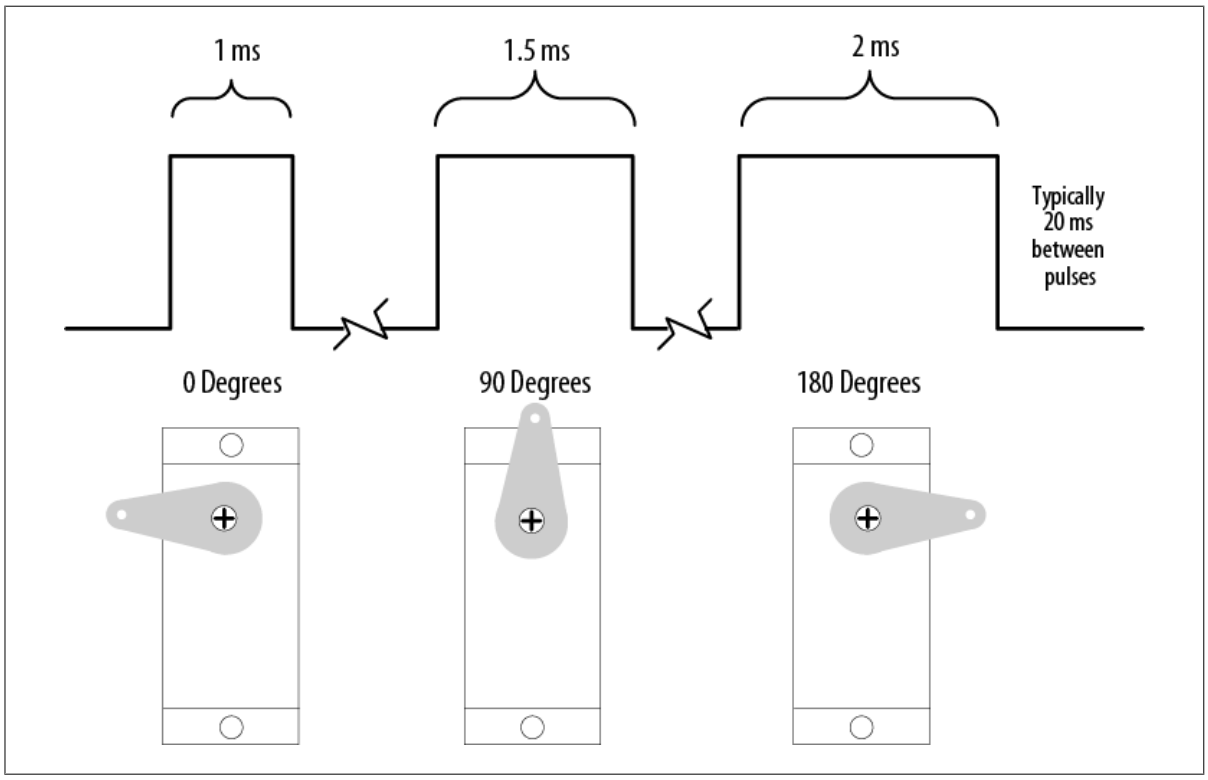
The third pin accept the control signal which is a pulse-width modulation (PWM) signal. It can be easily produced by all micro- controllers and Arduino board. This accepts the signal from your controller that tells it what angle to turn to. The control signal is fairly simple compared to that of a stepper motor. It is just a pulse of varying lengths. The length of the pulse corresponds to the angle the motor turns to.

For detailed and simplified explanation of Pulse Width Modulation (PWM), Please click here.

The pulse width sent to servo ranges as follows:

Minimum: 1 millisecond ---> Corresponds to 0 rotation angle.
Maximum: 2 millisecond ---> Corresponds to 180 rotation angle.

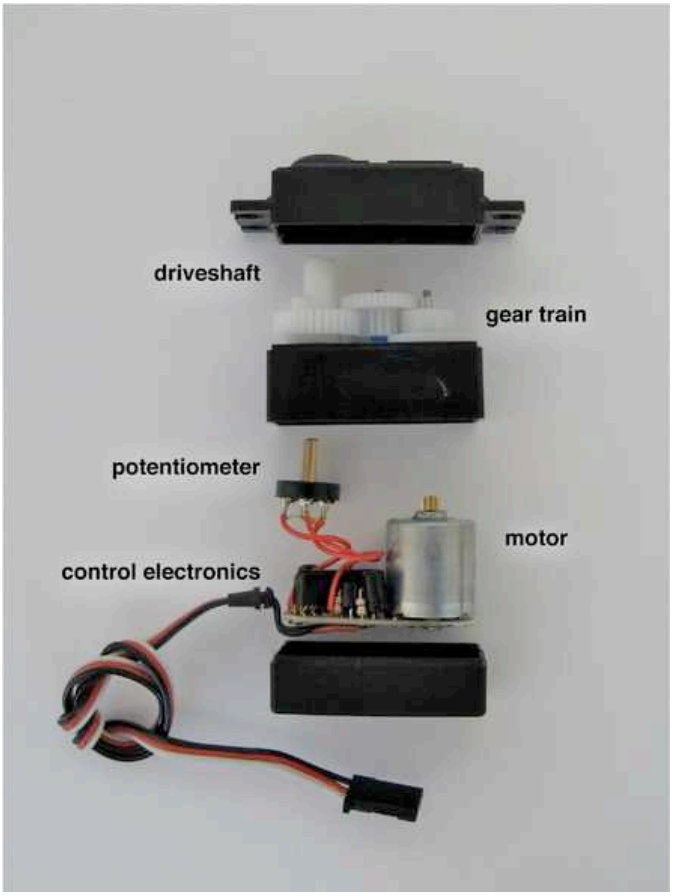
Any length of pulse in between will rotate the servo shaft to its corresponding angle. For example, 1.5 ms pulse corresponds to rotation angle of 90 degree. This is will explained in figure below.

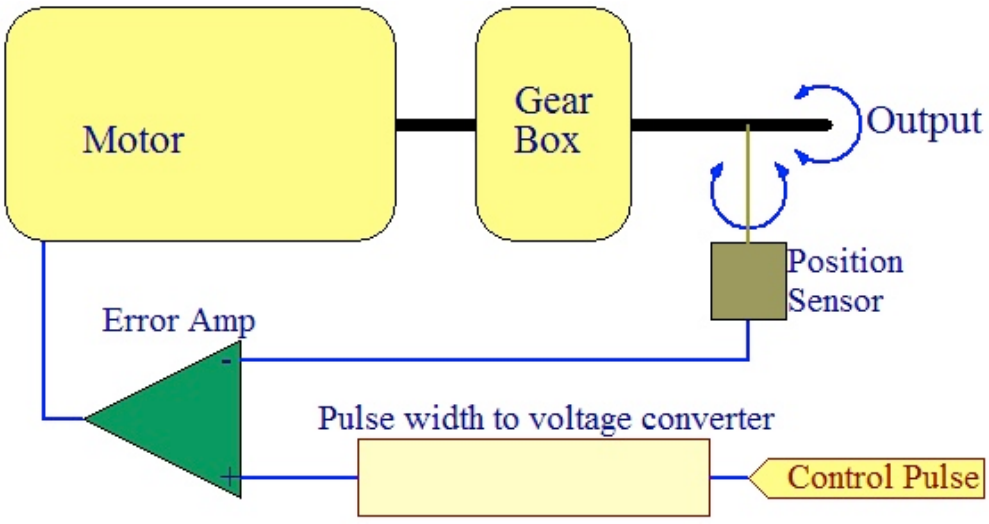


Inside the Servo Motor

Did ever wonder how the servo motors looks from inside?. Have a look at the corresponding picture. A servo motor was taken apart to show the internal parts. You can see a regular dc motor connected to a gear box and a potentiometer that give the feed back for angle position.

This is represented by the diagram below.

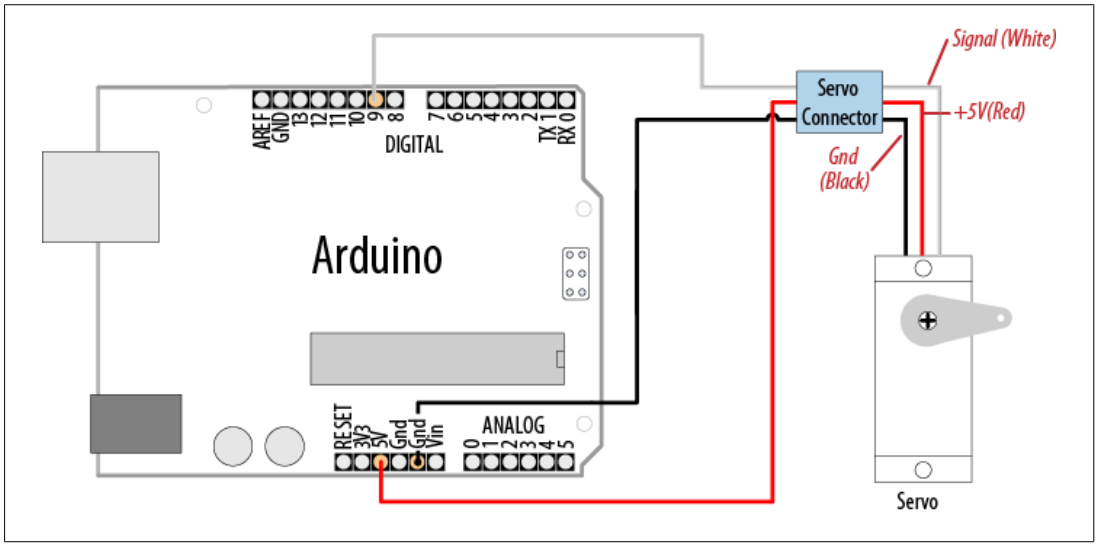


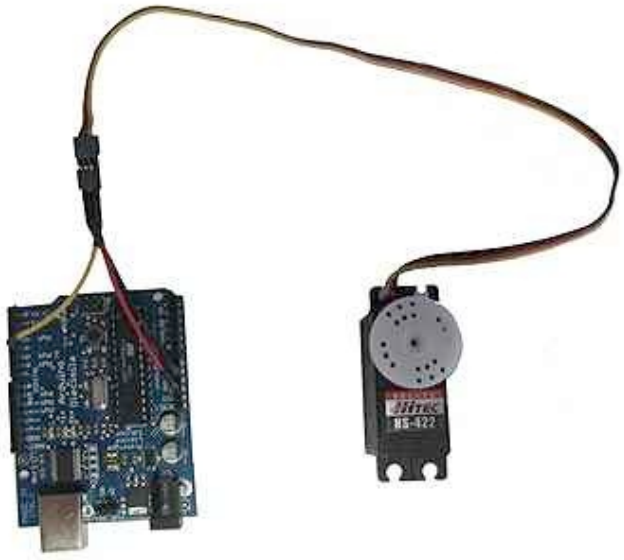


Motor Control Using Arduino

Standard servo motor control using Arduino is extremely easy. This is because the Arduino software comes with a sample servo sketch and servo library that will get you up and running quickly

1. Connect the black wire from the servo to the Gnd pin on the Arduino
2. Connect the red wire from the servo to the +5V pin on the Arduino
3. Connect the third wire (usually orange or yellow) from the servo to a digital pin on the Arduino





Important **Notes:**

1- It is not a good idea to connect a motor of any kind directly to the Arduino because it usually requires more power than the board can provide.

2- In our example, the servo is being used to demonstrate code and is not encountering any resistance. Note that you should use a standard or small size if you are uncertain, check the servo's no load current rating (it should usually be under 150mA).

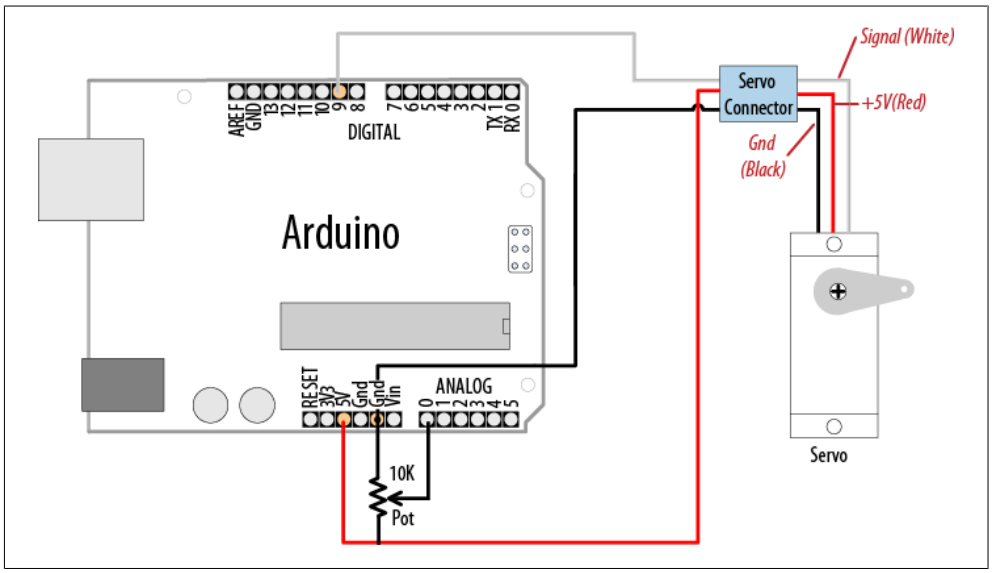
3-You may need an external source of 5 or 6 volts when connecting multiple servos. Four AA cells work well if you want to use battery power. Remember that you must connect the ground of the external power source to Arduino ground.

Here is the example Sweep sketch distributed with Arduino;:

```
#include <Servo.h>
Servo myservo; // create servo object to control a servo
int angle = 0;   // variable to store the servo position
void setup() {
264
|       Chapter 8: Physical Output
}
myservo.attach(9); // attaches the servo on pin 10 to the servo object
void loop() {
}
for(angle = 0; angle < 180; angle += 1) // goes from 0 degrees to 180 degrees
{
myservo.write(angle); delay(20);
// in steps of 1 degree // tell servo to go to position in variable 'angle' // waits 20ms
between servo commands
} for(angle = 180; angle >= 1; angle -= 1) // goes from 180 degrees to 0 degrees {
}
myservo.write(pos); delay(20);
// tell servo to go to position in variable 'pos' // waits 20ms between servo
commands
```

Controlling Servos with a Potentiometer or Sensor

This can be used for example if you want to control the pan and tilt of a camera or sensor connected to the servos. It is almost the same code like the above example with the addition of code to read the voltage on a potentiometer. This value is scaled so that the position of the pot (from 0 to 1023) is mapped to a value between 0 and 180 degrees. The only difference in the wiring is the addition of the potentiometer; please see figure below for hardware connection



Arduino Sketch:

```
#include <Servo.h> Servo myservo; // create servo object to control a servo
int potpin = 0; // analog pin used to connect the potentiometer
int val;     // variable to read the value from the analog pin
void setup()
{
   myservo.attach(9); // attaches the servo on pin 9 to the servo object
}
void loop()
{
   val = analogRead(potpin);              // reads the value of the potentiometer
   val = map(val, 0, 1023, 0, 179);              // scale it to use it with the
servo
    myservo.write(val);                  // sets position to the scaled value
    delay(15);                          // waits for the servo to get there
}
```