الحمـــادي للألكترونيـــات
ALHAMMADI FOR ELECTRONICS

AA019MG90MG90S180

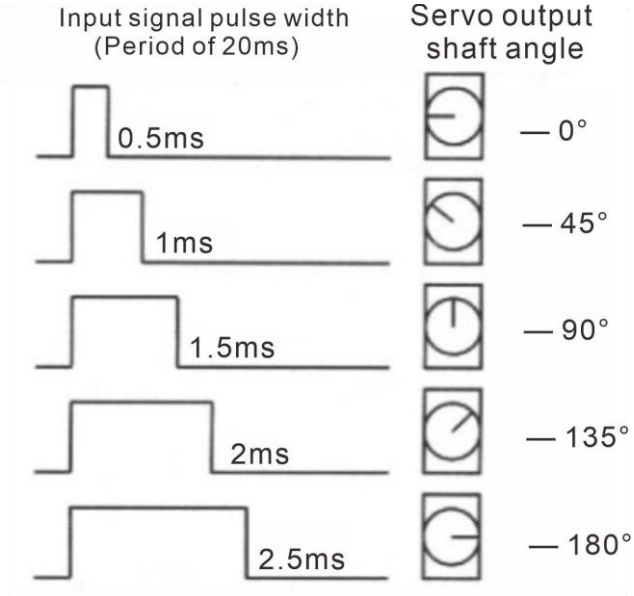
Product Description

This is a 180 degree servo which is a position controlled rotary actuator. It consists mainly of a housing, circuit board, coreless motor, gears and position sensor. The servo comes with a variety of white motor mounts that can be attached to the servo shaft. You can choose to mount any bracket you want for the circuit. It will act as a visual aid, making it easier for you to see the servo rotate. When in use, you can set the servo rotation angle with a setup code up to 180 degrees.

Working Principle

Servos come in a variety of sizes. But they all have three connection wires, brown, red and orange. Brown is the ground wire, red is the positive power wire, and orange is the signal wire.

The control line is used to transmit an angle control signal. This angle is determined by the duration of the control signal pulse, which is called pulse code modulation (PCM). Control of a servo generally requires a time base pulse of about 20 ms. The high level portion of this pulse is generally in the range of 0.5 ms 2.5 ms with a total interval of 2 ms. The width of the pulse will determine how far the motor will turn. For example, for a 1.5 ms pulse, the motor will turn to a 90° position (often referred to as the neutral position, which for a 180° servo would be the 90° position). If the pulse width is less than 1.5 milliseconds, then the motor is axially oriented towards 0 degrees. If the pulse width is greater than 1.5 milliseconds, the axial direction is toward 180°. The corresponding control relationship for this servo is shown below.



Product Parameters

Operating voltage: DC 5V

No-load speed: 0.12sec/60°

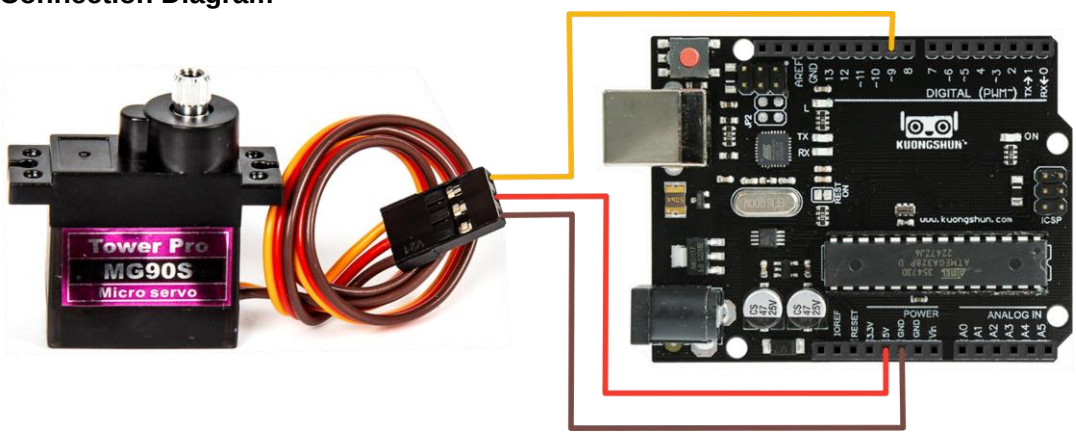
No-load current: 90mA

Blocking torque: 1.8kg.cm

Working temperature: 10~60℃

Storage temperature: 10~60℃

Connection Diagram



Sample Code

There are two ways to control the servo using the Arduino.

One is to use the Arduino's digital port to generate square waves with different duty cycles to simulate a PWM signal and use that signal to control the servo angle.

The other method is to call the commonly used Servo.h library file when setting up the Arduino program to control the servo rotation.

The Arduino has limited drive capability. Therefore, if you need to control multiple motors, you need an external power supply.

Code 1 is as follows:

```
/*
https://www.kuongshun-ks.com/
*/
int servopin=9;// select digital pin 9 for servomotor signal line int
myangle;// initialize angle variable int pulsewidth;// initialize width
variable void servopulse(int servopin,int myangle)// define a servo pulse
function
{
pulsewidth=round(map(myangle,0,180,500,2500));
//Serial.println(pulsewidth); digitalWrite(servopin,HIGH);// set the
level of servo pin as “high” delayMicroseconds(pulsewidth);// delay
microsecond of pulse width digitalWrite(servopin,LOW);// set the
level of servo pin as “low” delayMicroseconds(20000-pulsewidth);
}
void setup()
{
pinMode(servopin,OUTPUT);// set servo pin as “output”
Serial.begin(9600);// connect to serial port, set baud rate at “9600”
Serial.println("servo=o_seral_simple ready" ) ;
}
void loop()
{
for(int i=0;i<=200;i++) // giving the servo time to rotate to commanded position
{
servopulse(servopin,0);// use the pulse function
}
for(int i=0;i<=200;i++) // giving the servo time to rotate to commanded position
{
servopulse(servopin,180);// use the pulse function
}
}
```

Code 2 is as follows:

```
/*
```

```
https://www.kuongshun-ks.com/
*/
#include <Servo.h>
Servo myservo;// define servo variable name
void setup()
{
myservo.attach(9);// select servo pin(9 or 10)
}
void loop()
{
myservo.write(0);// set rotate angle of the motor
delay(1000); myservo.write(180);// set rotate
angle of the motor delay(1000);
}
```

Test Result

After successfully uploading the program and powering up, the servo will rotate back and forth between 0° and 180°.