

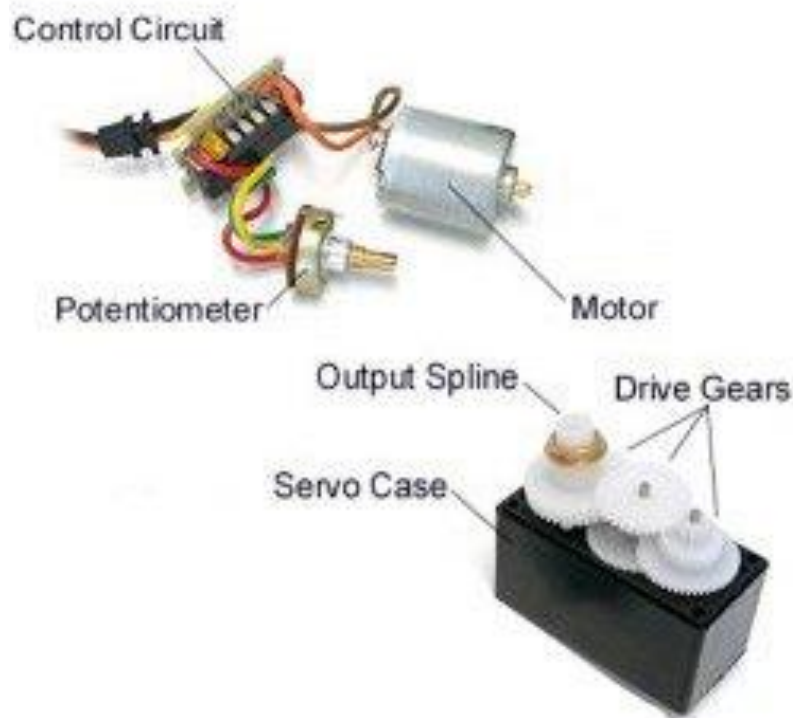
SERVO MOTOR

A servo motor is an electrical device which can push or rotate an object with great precision. If you want to rotate an object at some specific angles or distance, then you use servo motor. It is just made up of simple motor which runs through servo mechanism. If motor is used is DC powered then it is called DC servo motor, and if it is AC powered motor then it is called AC servo motor. We can get a very high torque servo motor in a small and light weight packages. Due to these features they are being used in many applications like toy car, RC helicopters and planes, Robotics, Machine etc.

What's inside the servo?

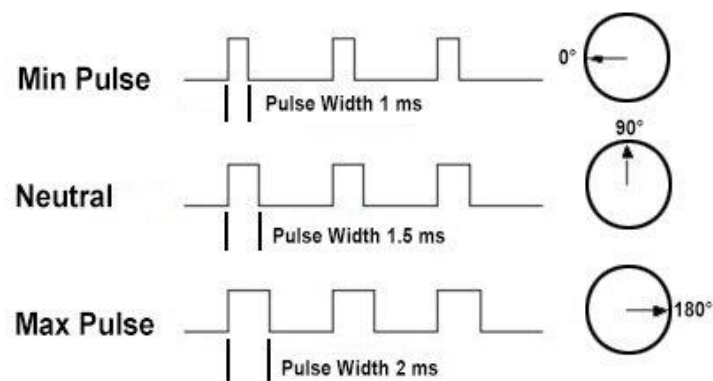
To fully understand how the servo works, you need to take a look under the hood. Inside there is a pretty simple set-up: a small DC motor, potentiometer, and a control circuit. The motor is attached by gears to the control wheel. As the motor rotates, the potentiometer's resistance changes, so the control circuit can precisely regulate how much movement there is and in which direction.

When the shaft of the motor is at the desired position, power supplied to the motor is stopped. If not, the motor is turned in the appropriate direction. The desired position is sent via electrical pulses through the signal wire. The motor's speed is proportional to the difference between its actual position and desired position. So if the motor is near the desired position, it will turn slowly, otherwise it will turn fast. This is called proportional control. This means the motor will only run as hard as necessary to accomplish the task at hand, a very efficient little guy.



How is the servo controlled?

Servos are controlled by sending an electrical pulse of variable width, or pulse width modulation (PWM), through the control wire. There is a minimum pulse, a maximum pulse, and a repetition rate. A servo motor can usually only turn 90° in either direction for a total of 180° movement. The motor's neutral position is defined as the position where the servo has the same amount of potential rotation in the both the clockwise or counter-clockwise direction. The PWM sent to the motor determines position of the shaft, and based on the duration of the pulse sent via the control wire; the rotor will turn to the desired position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90° position. Shorter than 1.5ms moves it in the counter clockwise direction toward the 0° position, and any longer than 1.5ms will turn the servo in a clockwise direction toward the 180° position.



When these servos are commanded to move, they will move to the position and hold that position. If an external force pushes against the servo while the servo is holding a position, the servo will resist from moving out of that position. The maximum amount of force the servo can exert is called the torque rating of the servo. Servo motors are rated in kg/cm (kilogram per centimetre) most hobby servo motors are rated at 3kg/cm or 6kg/cm or 12kg/cm. This kg/cm tells you how much weight your servo motor can lift at a particular distance. For example: A 6kg/cm Servo motor should be able to lift 6kg if the load is suspended 1cm away from the motors shaft, the greater the distance the lesser the weight carrying capacity. Servos will not hold their position forever though; the position pulse must be repeated to instruct the servo to stay in position.

Types of Servo Motors

There are two types of servo motors - AC and DC. AC servo can handle higher current surges and tend to be used in industrial machinery. DC servos are not designed for high current surges and are usually better suited for smaller applications. Generally speaking, DC motors are less expensive than their AC counterparts. These are also servo motors that have been built specifically for continuous rotation, making it an easy way to get your robot moving. They feature two ball bearings on the output shaft for reduced friction and easy access to the rest-point adjustment potentiometer.

Controlling Servo Motor Using Arduino

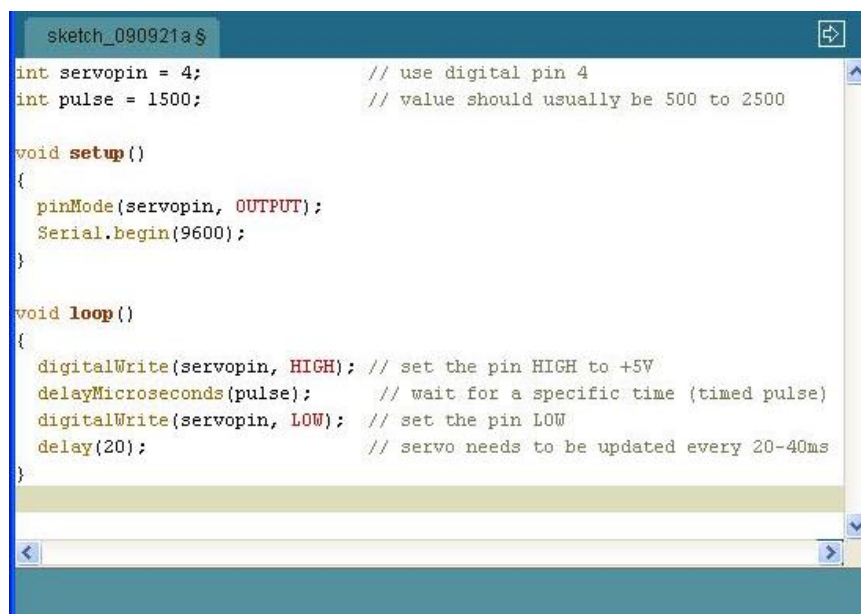
Controlling a servo motor directly from the Arduino is quite easy. However, a servo motor may require significantly more current than the Arduino can provide. The following example uses a standard sized servo (without any load) powered directly from the Arduino via USB. When powering the servo directly from the Arduino board:

Connect the black wire from the servo to the GND pin on the Arduino

Connect the red wire from the servo to the +5V pin on the Arduino

Connect the yellow or white wire from the servo to a digital pin on the Arduino

Alternatively, you can plug the servo's wire into three adjacent pins, and set the pin connected to the red lead to "HIGH" and the pin connected to the black lead to "LOW". If you want to use a more powerful servo, or if you want to connect it to a separate power supply, you would connect the battery / power supply's red (5V) and black (GND) wires to the servo's red and black wires, and connect the signal wire to the Arduino. Note that you also need to connect the batter's GND line to the Arduino's GND pins ("common ground").

A screenshot of the Arduino IDE interface. The title bar of the window reads "sketch_090921a\$". The code editor contains the following C++ code:

```
int servopin = 4;           // use digital pin 4
int pulse = 1500;          // value should usually be 500 to 2500

void setup()
{
  pinMode(servopin, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  digitalWrite(servopin, HIGH); // set the pin HIGH to +5V
  delayMicroseconds(pulse);     // wait for a specific time (timed pulse)
  digitalWrite(servopin, LOW);  // set the pin LOW
  delay(20);                    // servo needs to be updated every 20-40ms
}
```

Another option for controlling servos is to use the Arduino “servo library” (previously separate from the basic Arduino software, it is now included with V1.0). The servo library manages much of the overhead and includes new, custom commands. If you want to control multiple servo motors, you should use a servo motor controller and a separate power supply between 4.8V to 6V.

A screenshot of the Arduino IDE interface. The title bar reads "AnalogInOutSerial | Arduino 1.8.4". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checking, running, saving, and other functions. The main text area shows a C++ sketch for controlling a servo motor. The sketch includes the Servo library, initializes a servo object, and contains a loop that moves the servo from 0 to 180 degrees and back. The status bar at the bottom indicates "28" on the left and "Arduino/Genuino Uno on COM5" on the right.

```
AnalogInOutSerial | Arduino 1.8.4
File Edit Sketch Tools Help

AnalogInOutSerial$
#include<Servo.h>           //Servo library

Servo servo_test;          //initialize a servo object for the connected servo

int angle = 0;

void setup()
{
  servo_test.attach(9);     // attach the signal pin of servo to pin9 of arduino
}

void loop()
{
  for(angle = 0; angle < 180; angle += 1)    // command to move from 0 degrees to 180 degrees
  {
    servo_test.write(angle);                 //command to rotate the servo to the specified angle
    delay(15);
  }

  delay(1000);

  for(angle = 180; angle>=1; angle-=5)      // command to move from 180 degrees to 0 degrees
  {
    servo_test.write(angle);                 //command to rotate the servo to the specified angle
    delay(5);
  }

  delay(1000);
}

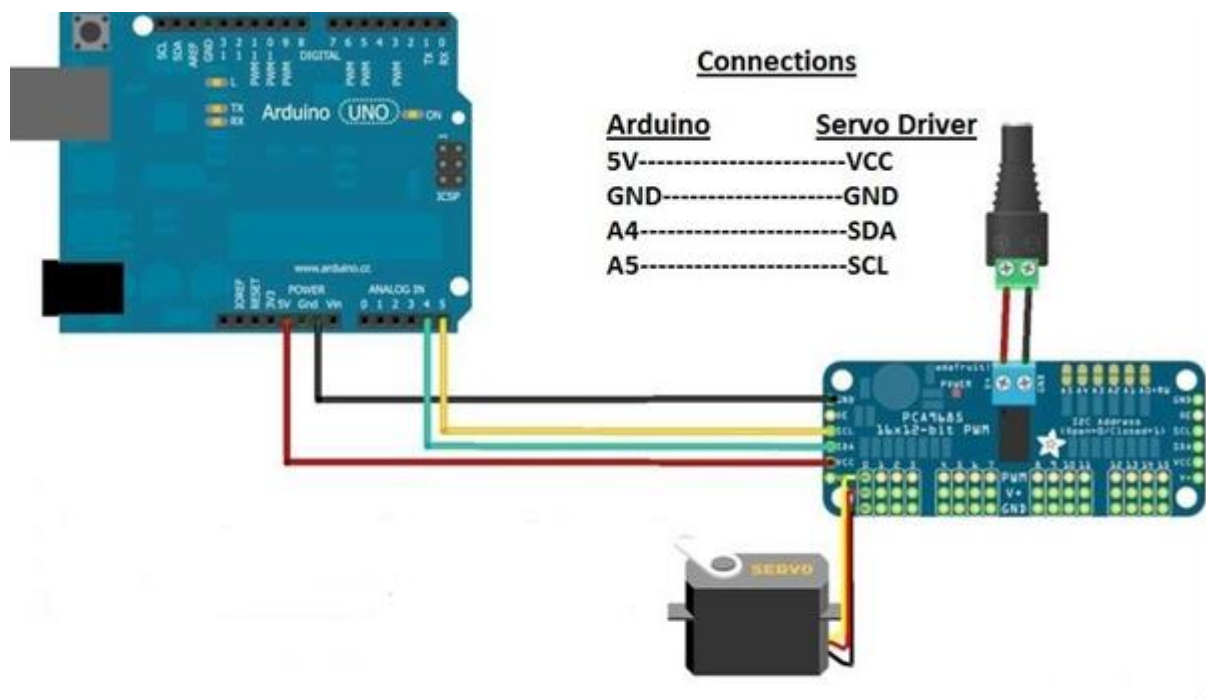
28 Arduino/Genuino Uno on COM5
```

PCA9685 Servo Motor Drive

We can use PCA9685 to operate tens of servo motor simultaneously.

The PCA9685 is a 16 Channel each output has its own 12-bit resolution (4096 steps) fixed frequency individual PWM controller that operates at a programmable frequency from a typical of 24 Hz to 1526 Hz with a duty cycle that is adjustable from 0 % to 100 % to allow the servo to be set to a specific angel. All outputs are set to the same PWM frequency. The Driver can very easily connected to your arduino, Raspberry Pie and easily programmed to control single or multiple servo motors and make your own RC plane, car, ship, quadrapod, hexapod or anything you want.

Connection for servo motor controlled by Arduino and PCA9685



There are two pins on the driver board for power; VCC and V+. It is important that, for the working of the IC inside the motor driver, 5V power supply is required. This 5 V can be provided from arduino 5 V pin as I did. Arduinos 5 V should be connected

to VCC only and not V+. V+ is used to power the motor. It will take more current than VCC.

After setting up the regulator, Follow the above schematics and connect the arduino board and Driver board

The SDA and SCL will be different for different arduino board. Just do a quick google search for the correct pins. I have mentioned some below.

Uno, Ethernet -----A4 (SDA), A5 (SCL)

Mega256020 ----- (SDA), 21 (SCL)

Leonardo ----- 2 (SDA), 3 (SCL)

Due ----- 20 (SDA), 21 (SCL), SDA1, SCL1

Arduino Code:

```
#include <Wire.h>

#include <Adafruit_PWMServoDriver.h>

Adafruit_PWMServoDriver pwm =
Adafruit_PWMServoDriver();

#define MIN_PULSE_WIDTH          650
#define MAX_PULSE_WIDTH          2350
#define DEFAULT_PULSE_WIDTH      1500
#define FREQUENCY                 50

uint8_t servonum = 0;

void setup()
{
    Serial.begin(9600);
```

```
Serial.println("16 channel Servo test!");

pwm.begin();

pwm.setPWMFreq(FREQUENCY);
}

int pulseWidth(int angle)
{
    int pulse_wide, analog_value;

    pulse_wide = map(angle, 0, 180,
MIN_PULSE_WIDTH, MAX_PULSE_WIDTH);

    analog_value = int(float(pulse_wide) /
1000000 * FREQUENCY * 4096);

    Serial.println(analog_value);

    return analog_value;
}

void loop() {
    pwm.setPWM(0, 0, pulseWidth(0));

    delay(1000);

    pwm.setPWM(0, 0, pulseWidth(120));

    delay(500);

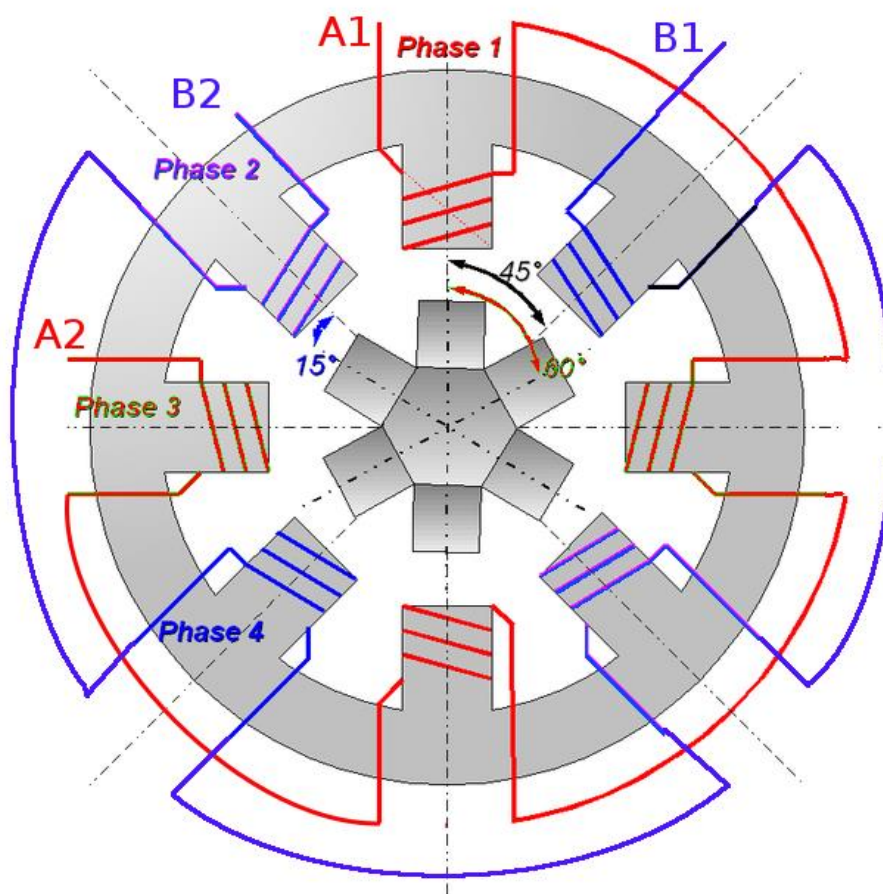
    pwm.setPWM(0, 0, pulseWidth(90));

    delay(1000);
}
```


STEPPER MOTOR

A stepper motor is an electromechanical device it converts electrical power into mechanical power. Also it is a brushless, synchronous electric motor that can divide a full rotation into an expansive number of steps. The motor's position can be controlled accurately without any feedback mechanism, as long as the motor is carefully sized to the application. Stepper motors are similar to switched reluctance motors.

The stepper motor uses the theory of operation for magnets to make the motor shaft turn a precise distance when a pulse of electricity is provided. The stator has eight poles, and the rotor has six poles. The rotor will require 24 pulses of electricity to move the 24 steps to make one complete revolution. Another way to say this is that the rotor will move precisely 15° for each pulse of electricity that the motor receives.



Types of Stepper Motor:

There are three main types of stepper motors, they are:

1. Permanent magnet stepper
2. Hybrid synchronous stepper
3. Variable reluctance stepper

Permanent Magnet Stepper Motor: Permanent magnet motors use a permanent magnet (PM) in the rotor and operate on the attraction or repulsion between the rotor PM and the stator electromagnets.

Variable Reluctance Stepper Motor: Variable reluctance (VR) motors have a plain iron rotor and operate based on the principle that minimum reluctance occurs with minimum gap, hence the rotor points are attracted toward the stator magnet poles.

Hybrid Synchronous Stepper Motor: Hybrid stepper motors are named because they use a combination of permanent magnet (PM) and variable reluctance (VR) techniques to achieve maximum power in a small package size.

Advantages of Stepper Motor:

1. The rotation angle of the motor is proportional to the input pulse.
2. The motor has full torque at standstill.
3. Precise positioning and repeatability of movement since good stepper motors have an accuracy of 3 – 5% of a step and this error is noncumulative from one step to the next.
4. Excellent response to starting, stopping and reversing.
5. Very reliable since there are no contact brushes in the motor. Therefore the life of the motor is simply dependant on the life of the bearing.
6. The motors response to digital input pulses provides open-loop control, making the motor simpler and less costly to control.
7. It is possible to achieve very low speed synchronous rotation with a load that is directly coupled to the shaft.

8. A wide range of rotational speeds can be realized as the speed is proportional to the frequency of the input pulses.

Operation of Stepper Motor:

Stepper motors operate differently from DC brush motors, which rotate when voltage is applied to their terminals. Stepper motors, on the other hand, effectively have multiple toothed electromagnets arranged around a central gear-shaped piece of iron. The electromagnets are energized by an external control circuit, for example a microcontroller.

To make the motor shaft turn, first one electromagnet is given power, which makes the gear's teeth magnetically attracted to the electromagnet's teeth. The point when the gear's teeth are thus aligned to the first electromagnet, they are slightly offset from the next electromagnet. So when the next electromagnet is turned ON and the first is turned OFF, the gear rotates slightly to align with the next one and from there the process is repeated. Each of those slight rotations is called a step, with an integer number of steps making a full rotation. In that way, the motor can be turned by a precise. Stepper motor doesn't rotate continuously, they rotate in steps. There are 4 coils with 90° angle between each other fixed on the stator. The stepper motor connections are determined by the way the coils are interconnected. In stepper motor, the coils are not connected together. The motor has 90° rotation step with the coils being energized in a cyclic order, determining the shaft rotation direction. The working of this motor is shown by operating the switch. The coils are activated in series in 1 sec intervals. The shaft rotates 90° each time the next coil is activated. Its low speed torque will vary directly with current.

Stepper Motor Control Using Arduino

Components Required

- Arduino UNO
- L293D Motor Driver IC
- Bipolar Stepper Motor (Here 28-BYJ48)
- Power Supply (suitable for your stepper motor)
- Breadboard (Prototyping Board)
- Connecting Wires

How to Design the Control Circuit?

We have used a bipolar stepper motor. Hence, we used the Motor Driver IC L293D, which is an H – bridge type driver. Since it is a bipolar stepper motor, there are only 4 wires we need to connect.

So, connect the two wires from one coil to outputs 1 and 2 of L293D and the other two wires from second coil to outputs 3 and 4.

The 4 inputs of the L293D Motor Driver IC are given from Arduino UNO. So connect them to any of the 4 digital I/O pins (here, we connected them to pins 2, 3, 4 and 5 of Arduino UNO).

Understand the power requirements of your stepper motor and provide necessary power supply. Wrong power supply would permanently damage the motor.

The control of steps is done with the help of computer using serial monitor. So, make sure that the RX and TX pins of the Arduino are not used as digital I/O. Alternatively,

we can control the steps or rotation of the motor with the help of analog input via a potentiometer.

Calculating the Steps per Revolution for Stepper Motor:

It is important to know how to calculate the steps per Revolution for your stepper motor because only then you can program it effectively.

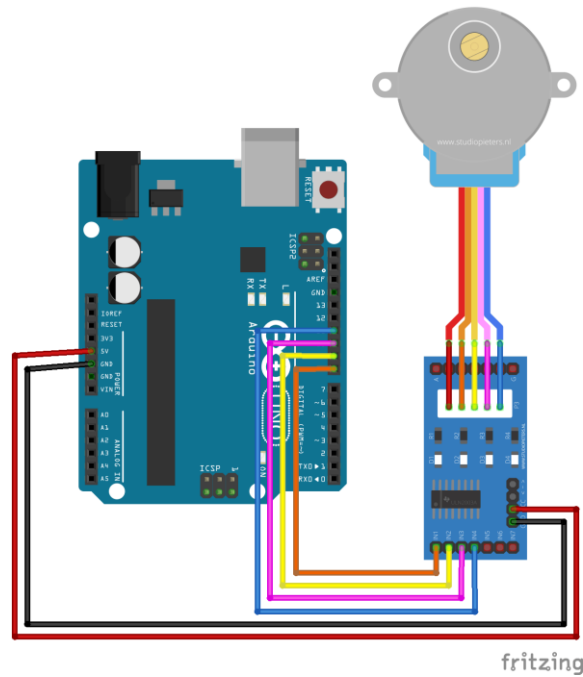
In Arduino we will be operating the motor in 4-step sequence so the stride angle will be 11.25° since it is 5.625° (given in datasheet) for 8 step sequence it will be 11.25° ($5.625 \times 2 = 11.25$).

Steps per revolution = $360/\text{step angle}$

Here, $360/11.25 = 32$ steps per revolution.

Rated voltage :	5VDC
Number of Phase	4
Speed Variation Ratio	1/64
Stride Angle	$5.625^\circ/64$
Frequency	100Hz
DC resistance	$50\Omega \pm 7\% (25^\circ\text{C})$
Idle In-traction Frequency	> 600Hz
Idle Out-traction Frequency	> 1000Hz
In-traction Torque	> 34.3mN.m(120Hz)
Self-positioning Torque	> 34.3mN.m
Friction torque	600-1200 gf.cm
Pull in torque	300 gf.cm
Insulated resistance	> 10M Ω (500V)
Insulated electricity power	600VAC/1mA/1s
Insulation grade	A
Rise in Temperature	< 40K(120Hz)
Noise	< 35dB(120Hz, No load, 10cm)
Model	28BYJ-48 – 5V

Datasheet for the servo motor



Circuit Diagram

Arduino Code:

```
// Arduino stepper motor control code
#include <Stepper.h> // Include the header file
// change this to the number of steps on your motor
#define STEPS 32
// create an instance of the stepper class using the steps
// and pins
Stepper stepper(STEPS, 8, 10, 9, 11);
int val = 0;
void setup() {
  Serial.begin(9600);
  stepper.setSpeed(200);
}
void loop() {
```

```
if (Serial.available()>0)
{
    val = Serial.parseInt();
    stepper.step(val);
    Serial.println(val); //for debugging
}

}
```

REFEERENCE:

- <https://www.electronicshub.org/stepper-motor-control-using-arduino/>
- <https://circuitdigest.com/microcontroller.../arduino-stepper-motor-control-tutorial>
- <https://www.electricaltechnology.org/.../stepper-motor-construction-types-and-modes-of...>
- <https://www.elprocus.com/stepper-motor-types-advantages-applications/>