STEPPER MOTOR WORKING

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. You learned previously that like poles of a magnet repel and unlike poles attract. Figure 1 shows a typical cross-sectional view of the rotor and stator of a stepper motor. From this diagram you can see that the stator (stationary winding) has eight poles, and the rotor has six poles (three complete magnets). 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. The number of degrees the rotor will turn when a pulse of electricity is delivered to the motor can be calculated by dividing the number of degrees in one revolution of the shaft (360°) by the number of poles (north and south) in the rotor. In this stepper motor 360° is divided by 24 to get 15°.

When no power is applied to the motor, the residual magnetism in the rotor magnets will cause the rotor to detent or align one set of its magnetic poles with the magnetic poles of one of the stator magnets. This means that the rotor will have 24 possible detent positions. When the rotor is in a detent position, it will have enough magnetic force to keep the shaft from moving to the next position. This is what makes the rotor feel like it is clicking from one position to the next as you rotate the rotor by hand with no power applied.

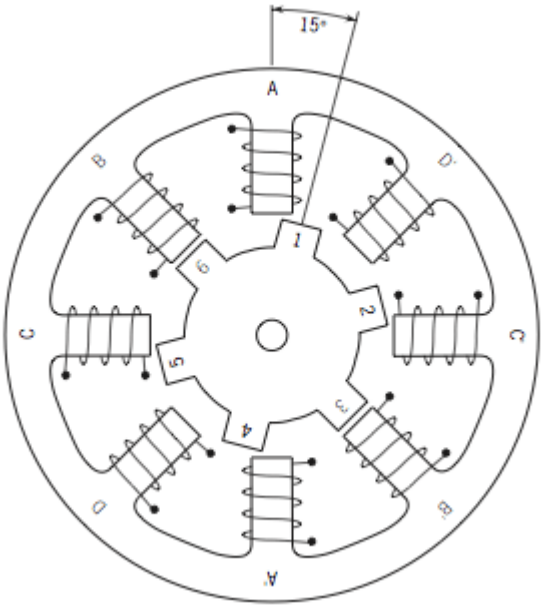

Fig 1. Diagram that shows the position of the six-pole rotor and eight-pole stator of a typical stepper motor.

Stepper Motor Switching Sequence: The stepper motor can be operated in three different stepping modes, namely, full-step, half-step, and microstep.

Full-Step : The stepper motor uses a four-step switching sequence, which is called a full-step switching sequence which is already described above.

Half-Step : Another switching sequence for the stepper motor is called an eight-step or half-step sequence. The switching diagram for the half-step sequence is shown in Fig. 2. The main feature of this switching sequence is that you can double the resolution of the stepper motor by causing the rotor to move half the distance it does when the full-step switching sequence is used. This means that a 200-step motor, which has a resolution of 1.8°, will have a resolution of 400 steps and 0.9°. The half-step switching sequence requires a special stepper motor controller, but it can be used with a standard hybrid motor. The way the controller gets the motor to reach the half-step is to energize both phases at the same time with equal current.

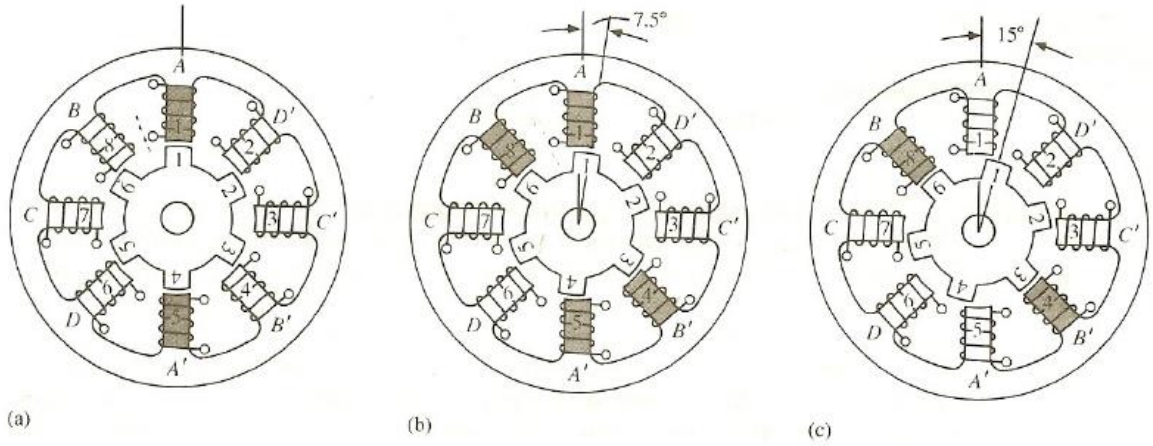

Fig 2: Switching sequence for half – step mode

Types of Stepper Motors : A stepper, or stepping motor converts electronic pulses into proportionate mechanical movement. Each revolution of the stepper motor's shaft is made up of a series of discrete individual steps. A step is defined as the angular rotation produced by the output shaft each time the motor receives a step pulse. These types of motors are very popular in digital control circuits, such as robotics, because they are ideally suited for receiving digital pulses for step control. Each step causes the shaft to rotate a certain number of degrees. A step angle represents the rotation of the output shaft caused by each step, measured in degrees. The most popular types of stepper motors are permanent-magnet (PM) and variable reluctance (VR).

Permanent-magnet (PM) Stepper Motors : The permanent-magnet stepper motor operates on the reaction between a permanent-magnet rotor and an electromagnetic field. Figure 6 shows a basic two-pole PM stepper motor. The rotor shown in Figure 3(a) has a permanent magnet mounted at each end. The stator is illustrated in Figure 3(b). Both the stator and rotor are shown as having teeth. The teeth on the rotor surface and the stator pole faces are offset so that there will be only a limited number of rotor teeth aligning themselves with an energized stator pole. The number of teeth on the rotor and stator determine the step angle that will occur each time the polarity of the winding is reversed. Greater the number of teeth, smaller the step angle.

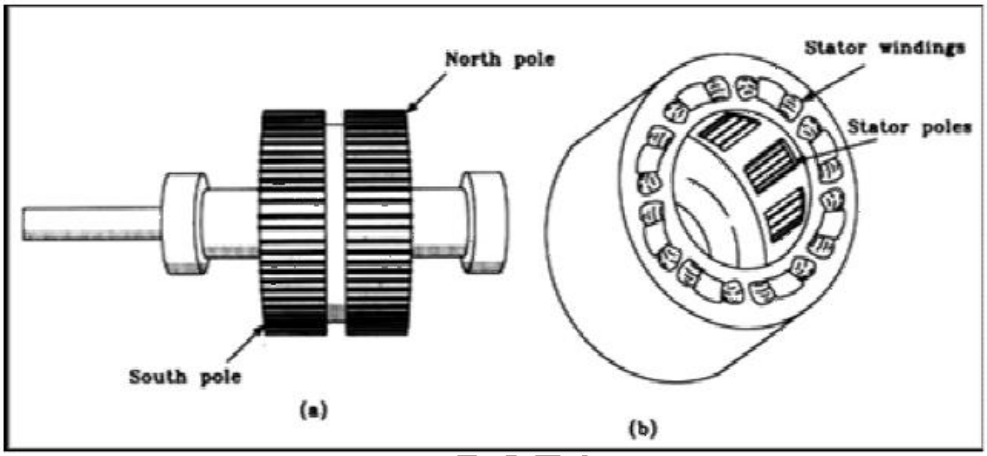

Fig 3: Components of a PM Stepper Motor (a)Rotor and (b) Stator

Variable-reluctance (VR) Stepper Motors: The variable-reluctance (VR) stepper motor differs from the PM stepper in that it has no permanent-magnet rotor and no residual torque to hold the rotor at one position when turned off. When the stator coils are energized, the rotor teeth will align with the energized stator poles. This type of motor operates on the principle of minimizing the reluctance along the path of the applied magnetic field. By alternating the windings that are energized in the stator, the stator field changes, and the rotor is moved to a new position. The stator of a variable-reluctance stepper motor has a magnetic core constructed with a stac of steel laminations. The rotor is made of unmagnetized soft steel with teeth and slots.

Stepper Motor Applications: Stepper motors are used in a wide variety of applications in industry, including computer peripherals, business machines, motion control, and robotics, which are included in process control and machine tool applications. A complete list of applications is shown below.

Computer Peripherals

Application	Use
Floppy Disc	position magnetic pickup
Printer	carriage drive
Printer	rotate character wheel
Printer	paper feed
Printer	ribbon wind/rewind
Printer	position matrix print head
Tape Reader	index tape
Plotter	X-Y-Z positioning
Plotter	paper feed

Business Machines

Application	Use
Card Reader	position cards
Copy Machine	paper feed
Banking Systems	credit card positioning
Banking Systems	paper feed
Typewriters (automatic)	head positioning
Typewriters (automatic)	paper feed
Copy Machine	lens positioning
Card Sorter	route card flow

Process Control

Application	Use
Carburetor Adjusting	air-fuel mixture adjust
Valve Control	fluid gas metering
Conveyor	main drive
In-Process Gaging	parts positioning
Assembly Lines	parts positioning
Silicon Processing	I. C. wafer slicing
I. C. Bonding	chip positioning
Laser Trimming	X-Y positioning
Liquid Gasket Dispensing	valve cover positioning
Mail Handling Systems	feeding and positioning

Machine Tool

Application	Use
Milling Machines	X-Y-Z table positioning
Drilling Machines	X-Y table positioning
Grinding Machines	downfeed grinding wheel
Grinding Machines	automatic wheel dressing
Electron Beam Welder	X-Y-Z positioning
Laser Cutting	X-Y-Z positioning
Lathes	X-Y positioning
Sewing	X-Y table positioning

Interfacing Stepper motor with BBB

For interfacing Stepper motor with BBB, 4 GPIO pins of BBB are used. General interfacing and working steps are:

1) Import GPIO part of Adafruit BBIO Library using following instruction in python program

```
import Adafruit_BBIO.GPIO as GPIO
```

2) Import standard python libraries

```
import sys
import os
import time
```

3) GPIO Pin for Stepper Motor Interfacing Board are:

```
S1   =    "P9_11"
S2   =    "P8_7"
S3   =    "P9_12"
S4   =    "P8_8"
```

4) Following python can be written which turns the stepper PIN ON. It will setup a PIN as GPIO, sets the direction as OUT and then makes the PIN high.

```
def stepperOn(pin):
        GPIO.setup(pin, GPIO.OUT)
        GPIO.output(pin, GPIO.HIGH)
        return
```

5) Another python function turns the stepper PIN OFF. It will setup a PIN as GPIO, sets the direction as OUT and then makes the PIN low

```
def stepperOff(pin):
        GPIO.setup(pin, GPIO.OUT)
        GPIO.output(pin, GPIO.LOW)
```

6) Now in main function, an infinite loop can be use to rotate stepper motor in clockwise direction. Changing the output sequence will move motor in anticlockwise direction.

```
stepperOn(S1)
stepperOn(S2)
stepperOn(S3)
stepperOn(S4)

for i in range (0,12):
```

```
        GPIO.output(S1, GPIO.HIGH)
        GPIO.output(S2, GPIO.LOW)
        GPIO.output(S3, GPIO.LOW)
        GPIO.output(S4, GPIO.LOW)
        time.sleep(0.5)
        GPIO.output(S1, GPIO.LOW)
        GPIO.output(S2, GPIO.HIGH)
        GPIO.output(S3, GPIO.LOW)
        GPIO.output(S4, GPIO.LOW)
        time.sleep(0.5)
        GPIO.output(S1, GPIO.LOW)
        GPIO.output(S2, GPIO.LOW)
        GPIO.output(S3, GPIO.HIGH)
        GPIO.output(S4, GPIO.LOW)
        time.sleep(0.5)
        GPIO.output(S1, GPIO.LOW)
        GPIO.output(S2, GPIO.LOW)
        GPIO.output(S3, GPIO.LOW)
        GPIO.output(S4, GPIO.HIGH)
        time.sleep(0.5)
    stepperOff(S1)
    stepperOff(S2)
    stepperOff(S3)
    stepperOff(S4)
    exit()
```

7) In above program, alternative stators are activated by enabling GPIO pins which make rotor teeth move by step angle.