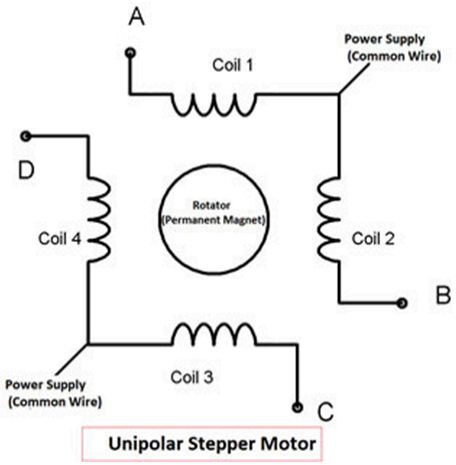
6.Stepper Motor Control: PWM to manage stepper motor speed using Uno/Raspberry Pi.

Remark: Signature:

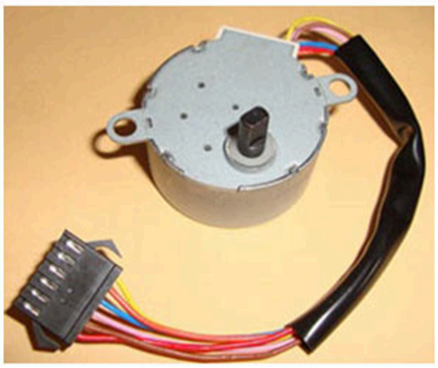
Raspberry Pi is an ARM architecture processor based board designed for electronic engineers and hobbyists. The PI is one of most trusted project development platforms out there now. With higher processor speed and 1 GB RAM, the PI can be used for many high profile projects like Image processing and Internet of Things.

In this tutorial, we will **Control the Speed of a Stepper Motor using Raspberry Pi**. In Stepper Motor, as the name itself says, the rotation of the shaft is in Step form. There are different types of Stepper Motor; in here we will be using the most popular one that is **Unipolar Stepper Motor**. Unlike DC motors, we can rotate a stepper motor to any particular angle by giving it proper instructions.

To rotate this Four Stage Stepper Motor, we will deliver power pulses by using Stepper Motor Driver Circuit. The driver circuit takes logic triggers from PI. If we control the logic

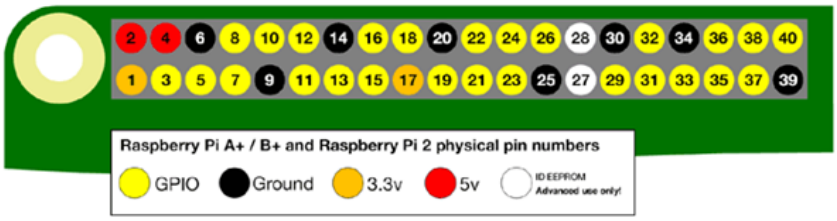


Unipolar Stepper Motor



triggers, we control the power pulses and hence the speed of the stepper motor.

There are **40 GPIO output pins in Raspberry Pi 2**. But out of 40, only 26 GPIO pins (GPIO2 to GPIO27) can be programmed. Some of these pins perform some special functions. With special GPIO put aside, we have only 17 GPIO remaining. Each of these 17 GPIO pin can deliver a maximum of **15mA** current. And the sum of currents from all GPIO Pins cannot exceed **50mA.**



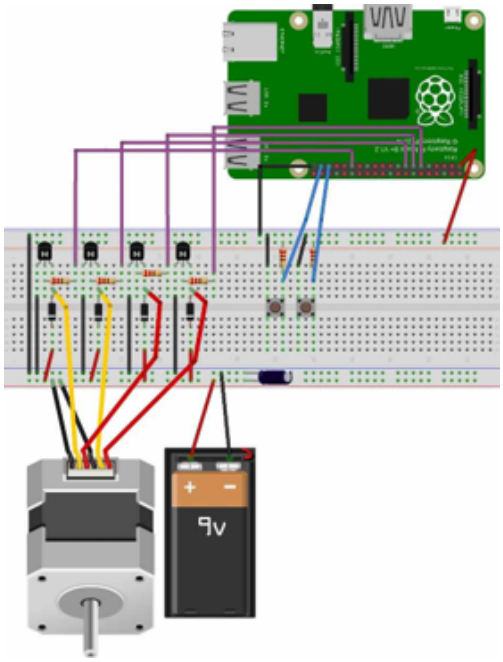
There are **+5V (Pin 2 & 4) and +3.3V (Pin 1 & 17) power output pins** on the board for connecting other modules and sensors. These power rails cannot be used to drive the Stepper Motor, because we need more power to rotate it. So we have to deliver the power to Stepper Motor from another power source. My stepper motor has a voltage rating of 9V so I am using a 9v battery as my second power source. Search your stepper motor model number to know voltage rating. Depending on the rating choose the secondary source appropriately.

Components Required:

Here we are using **Raspberry Pi 2 Model B with Raspbian Jessie OS**. All the basic Hardware and Software requirements are previously discussed, you can look it up in the Raspberry Pi Introduction, other than that we need:

- Connecting pins
- 220Ω or 1KΩresistor (3)
- Stepper Motor
- Buttons (2)
- 2N2222 Transistor (4)
- 1N4007 Diode (4)
- Capacitor- 1000uF
- Bread Board

Circuit Explanation:



Working Explanation:

Once everything is connected as per the circuit diagram, we can turn ON the PI to write the program in PYTHON.

We will talk about a few commands which we are going to use in the PYTHON **program**.

```
import RPi.GPIO as IO              # we are calling for header file which helps us use GPIO's of PI
import time                        # we are calling for time to provide delays in program
IO.setwarnings(False)              # do not show any warnings
x=1                                # integer for storing the delay multiple
IO.setmode (IO.BCM)
IO.setup(5,IO.OUT)                 # initialize GPIO5 as an output.
IO.setup(17,IO.OUT)
IO.setup(27,IO.OUT)
IO.setup(22,IO.OUT)
IO.setup(19,IO.IN)                 # initialize GPIO19 as an input.
IO.setup(26,IO.IN)
while 1:                           # execute loop forever
   IO.output(5,1)                  # Step1 go high
   IO.output(22,0)
   for y in range(x):              # sleep for  x*100msec
      time.sleep(0.01)
   IO.output(17,1)                 # step2 go high
   IO.output(5,0)
   for y in range(x):
      time.sleep(0.01)             # sleep for  x*100msec
   IO.output(27,1)                 #step 3 go high
   IO.output(17,0)
```

```
for y in range(x):
    time.sleep(0.01)        # sleep for  x*100msec
IO.output(22,1)             #step 4 go high
IO.output(27,0)
for y in range(x):
    time.sleep(0.01)                    # sleep for  x*100msec
        if(IO.input(26) == False):          #if button1 is pressed
                if(x<100):
                x=x+1                           #increment x by one if x<100
                time.sleep(0.5)                  #sleep for 500ms
        if(IO.input(19) == False):          #if button2 is pressed
      if(x>1):
      x=x-1                             #decrement x by one if x>1
      time.sleep(0.5)                    #sleep for 500ms
```