Assignment No: C1

Title: Controlling the operation of stepper motor using Raspberry Pi /Beagle board circuit.

PROBLEM STATEMENT /DEFINITION: Write an application using Raspberry-Pi /Beagle board to control the operation of stepper motor.

OBJECTIVE: Understanding the connectivity of Raspberry Pi /Beagle board circuit with stepper motor. To understand the actuation.

Pre-requisite: Basic knowledge of GPIO of Raspberry pi/BBB Basic knowledge of Python programming. Working and connections of sensors.

Learning Objectives: 1.nderstanding the connectivity of Raspberry Pi /Beagle board circuit with stepper motor. 2. To perform actuation using Raspberry Pi /Beagle. Learning Outcomes: The students will be able to 3.To interface stepper motor to Raspberry pi/Beagle board. 4. To control operation of stepper motor through Raspberry pi/Beagle boarThe5. Can perform actuation

H/W AND S/W Requirements : Raspberry Pi/Beagle board Development Boards PC / Monitor/Keyboard Raspbian (OS), Debian LINUX and Python

Theory: A stepper motor is an electromechanical device which converts electrical pulses into discrete mechanical movements. The shaft or spindle of a stepper motor rotates in discrete step increments when electrical command pulses are applied to it in the proper sequence. The motors rotation has several direct relationships to these applied input pulses. The sequence of the applied pulses is directly related to the direction of motor shafts rotation. The speed of the motor shafts rotation is directly related to the frequency of the in put pulses and the length of rotation is directly related to the number of input pulses applied.One of the most significant advantages of a stepper motor is its ability to be accurately controlled in an open loop system. Open loop control means no feedback information about position is needed. This type of control eliminates the need for expensive sensing and feedback devices such as optical encoders. Your position is known simply by keeping track of the input step pulses.

Interfacing with Raspberry Pi The motor connects to the controller board with a pre-supplied connector. The controller board has 4+2 pins that need to be connected to the Pi header (P1).

1.5V (P1-02)

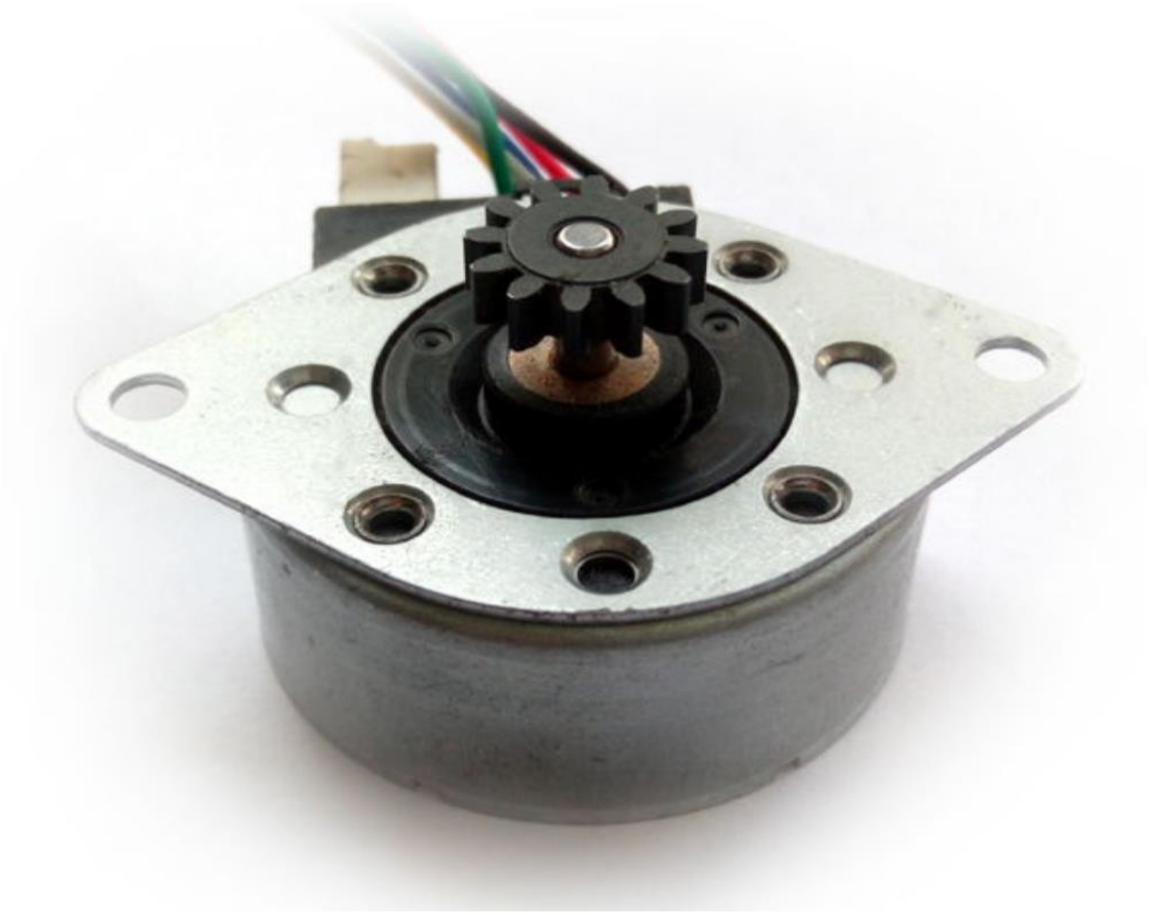
2 GND (P1-06)

3 Inp1 (P1-11)

4.Inp2 (P1-15)

5.Inp3 (P1-16)

6. Inp4 (P1-18)



Stepper Motor

The P1-XX references above represent the Pi header pins used. These are defined in the Python example below in the StepPins list so if you use different pins be sure to update the Python list as well. You can use other GPIO pins if required just remember to update your Python script. To rotate the stepper motor you provide a sequence of "high" and "low" levels to each of the 4 inputs in sequence. By setting the correct sequence of high and low levels the motor spindle will rotate. The direction can be reversed by reversing the sequence.

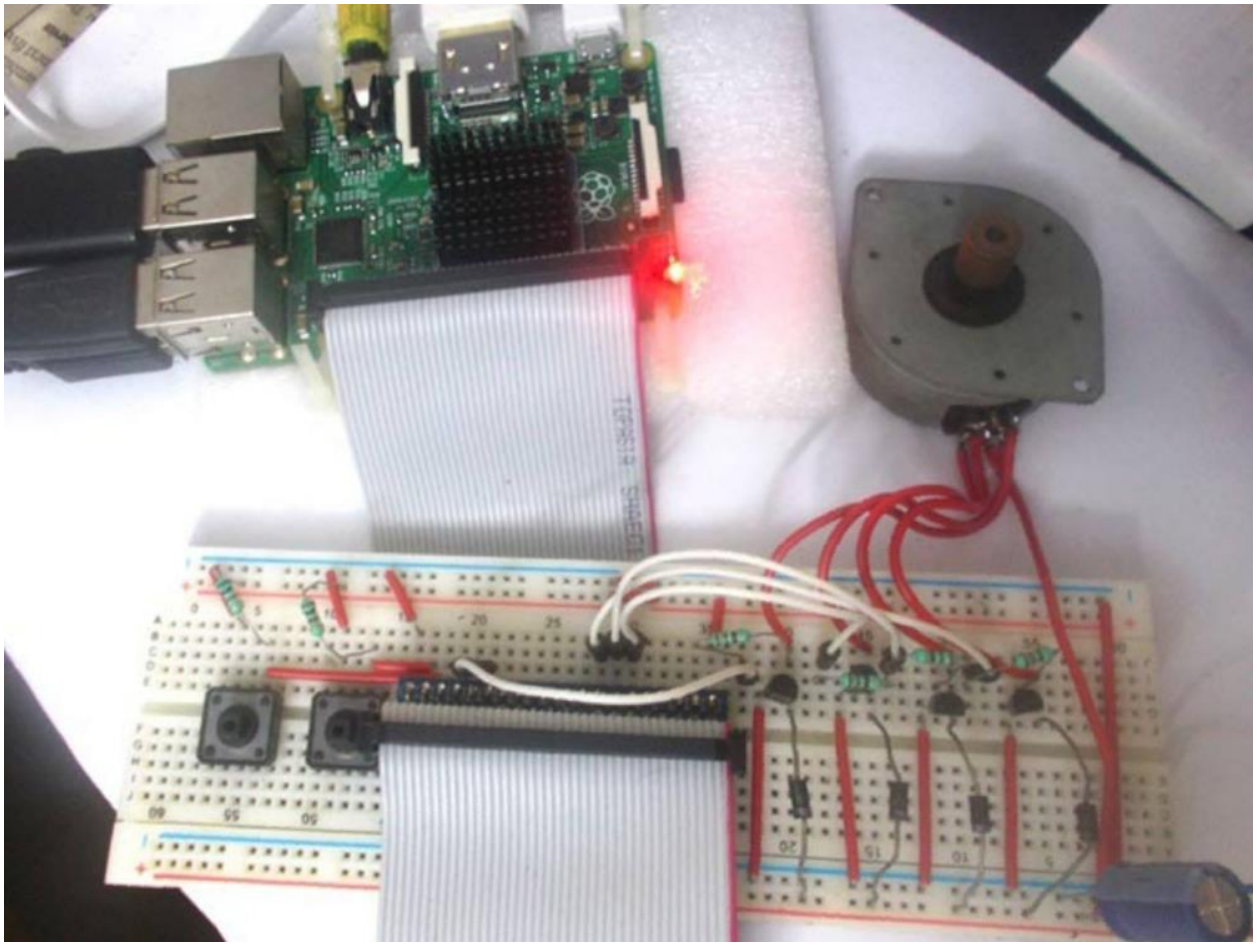
Source code:

```
#!/usr/bin/python

# Import required libraries

import sys

import time

import RPi.GPIO as GPIO

Use BCM GPIO references

# instead of physical pin numbers

GPIO.setmode(GPIO.BCM)

#Define GPIO signals to use

# Physical pins 11,15,16,18

# GPIO17,GPIO22,GPIO23,GPIO24

StepPins = [17,22,23,24]

# Set all pins as output for pin in StepPins:

print "Setup pins"

GPIO.setup(pin,GPIO.OUT)

GPIO.output(pin, False)

# Define advanced sequence

# as shown in manufacturers datasheet

Seq = [[1,0,0,1], [1,0,0,0], [1,1,0,0], [0,1,0,0], [0,1,1,0], [0,0,1,0], [0,0,1,1],
[0,0,0,1]]
```

```
StepCount  =  len(Seq)
StepDir =  1  # Set to 1 or  2 for  clockwise
# Set to  -1 or  -2 for  anti-clockwise
# Read wait  time from command line
 if  len(sys.argv)>1: WaitTime =  int(sys.argv[1])/float(1000)
else: WaitTime =  10/float(1000)
#Initialise variables StepCounter =  0
 # Start main loop while  True:
print  StepCounter,
print  Seq[StepCounter]
for  pin in  range(0,4):
 xpin=StepPins[pin]
# Get  GPIO
if  Seq[StepCounter][pin]!=0:
 print  "  Enable  GPIO %i"  %(xpin)
 GPIO.output(xpin, True)
 else: GPIO.output(xpin, False)
StepCounter +=  StepDir
#  If we  reach the end of the  sequence
# start again
 if  (StepCounter>=StepCount):
 StepCounter =  0
if  (StepCounter<0):
StepCounter =  StepCount+StepDir
# Wait  before  moving
```

time.sleep(WaitTime)

Output:



Stepper motor control with raspberry pi

Conclusion:

In this way we are successfully controlled the operation of stepper motor using raspberry pi.