Assignment No: 7

Aim Write an application using Raspberry-Pi Beagle board to control the operation of stepper

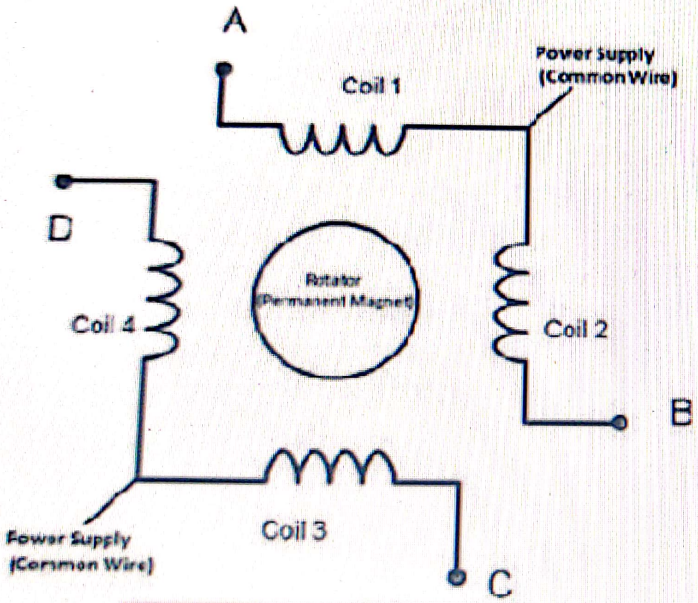
Theory

① Stepper Motor

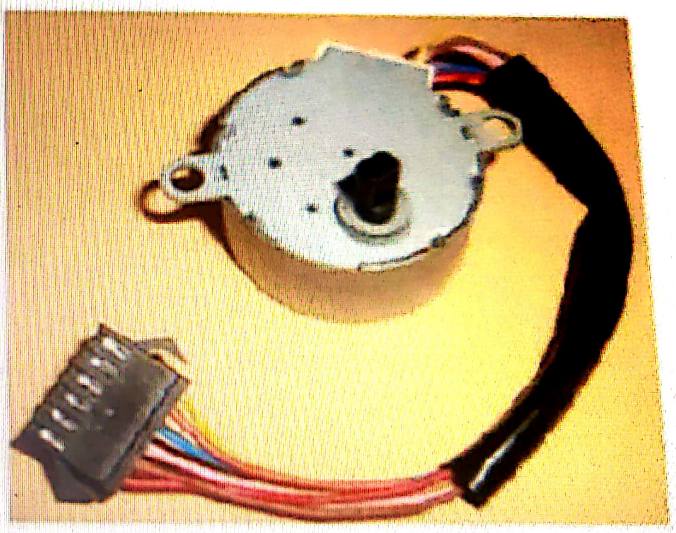
- In stepper Motor as the name itself says the rotation of shaft in step form. There are different types of stepper Motor in here we will be using the most popular one that is Unipolar Steppes Motor
- Unlike DC motor we can rotate steppes motor to any particular angle by using proper instructi

= To rotate this 4 stage stepper Motor we will deliver power pulses by using Stepper Motor Driver Circuit

- The driver circuit takes logic triggres from PI. If we control the loger triggers we control the power pulse and hence the speed of steper motor
- There are 40 GPIO o/p pins in Raspberry Pi 2 But out of 40, only 26 GPIO pins can be prog
- Some of these pins perform some special functions
- With special GPIO put aside we have only 17 GPI remaining
- Each of these 17 GPIO pin can deliver a max



Unipolar Stepper Motor



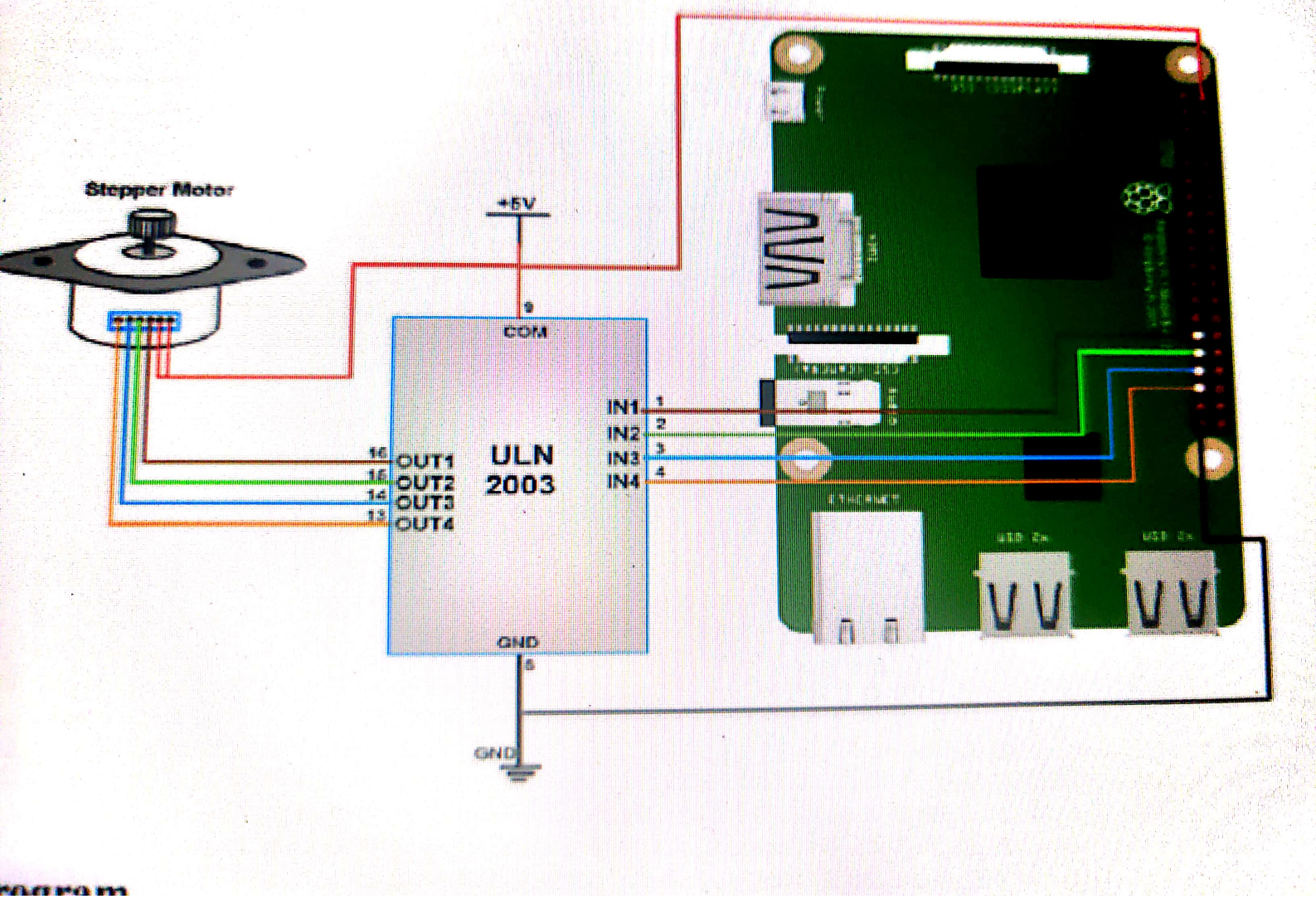
Unipolar Stepper motor Fig 1

of 15 mA current And the sum of currents from all GPIO Pins cannot exceed ~~onto~~ 50mA
- There are +5V and +3.3V power o/p pin 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 vt rating of 9v so I am using a 9v battery as my second power source
- Search your stepper motor model no to know vt rating Depending on the rating choose the secondary source appropriately

=

Conclusion

Thus we have implemented application of stepper motors using Python with Raspberry Pi



Python Program

Stepper Motor interfacing with Raspberry Pi

```
import RPi.GPIO as GPIO
from time import sleep
import sys

#assign GPIO pins for motor
motor_channel = (29,31,33,35)
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
#for defining more than 1 GPIO channel as input/output use
GPIO.setup(motor_channel, GPIO.OUT)

motor_direction = input('select motor direction a=anticlockwise, c=clockwise: ')
while True:
try:
if(motor_direction == 'c'):
print('motor running clockwise\n')
GPIO.output(motor_channel, (GPIO.HIGH,GPIO.LOW,GPIO.LOW,GPIO.HIGH))
sleep(0.02)
GPIO.output(motor_channel, (GPIO.HIGH,GPIO.HIGH,GPIO.LOW,GPIO.LOW))
sleep(0.02)
GPIO.output(motor_channel, (GPIO.LOW,GPIO.HIGH,GPIO.HIGH,GPIO.LOW))
sleep(0.02)
GPIO.output(motor_channel, (GPIO.LOW,GPIO.LOW,GPIO.HIGH,GPIO.HIGH))
sleep(0.02)

elif(motor_direction == 'a'):
print('motor running anti-clockwise\n')
```

```
GPIO.output(motor_channel, (GPIO.HIGH,GPIO.LOW,GPIO.LOW,GPIO.HIGH))
sleep(0.02)
GPIO.output(motor_channel, (GPIO.LOW,GPIO.LOW,GPIO.HIGH,GPIO.HIGH))
sleep(0.02)
GPIO.output(motor_channel, (GPIO.LOW,GPIO.HIGH,GPIO.HIGH,GPIO.LOW))
sleep(0.02)
GPIO.output(motor_channel, (GPIO.HIGH,GPIO.HIGH,GPIO.LOW,GPIO.LOW))
sleep(0.02)


#press ctrl+c for keyboard interrupt
except KeyboardInterrupt:
#query for setting motor direction or exit
motor_direction = input('select motor direction a=anticlockwise, c=clockwise or q=exit: ')
#check for exit
if(motor_direction == 'q'):
print('motor stopped')
sys.exit(0)
```