Interface DC motor using motor driver IC and Generate PWM for velocity control of the motor

e-Yantra Team

July 7, 2015

Contents

1 Objective

In this tutorial we shall learn to control the velocity of DC motors (using PWM) interfaced with Raspberry Pi using a motor driver IC.

2 Prerequisites

- Python programming skills
- An R-Pi (with I2C; I will be using version 2)
- An idea about working of a dc motor
- Interfacing an MCP23017 IC with an R-Pi should be known

3 Hardware Requirement

1. Raspberry Pi (I will be using Version 2 Model B)
2. MCP23017
3. Power adapter
4. DC motor
5. L293D
6. Capacitors (0.1 uF)
7. External power supply (for 12V)
8. Connecting wires
9. Bread board

4 Software Requirement

1. PyScripter (version 2.7 or above)
2. Mobaxterm (for windows user)

5 Theory and Description

A DC motor is any of a class of electrical machines that converts direct current electrical power into mechanical power. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current flow in part of the motor. Most types produce rotary motion; a linear motor directly produces force and motion in a straight line.[1]

Motor Driver

Normal DC gear-head motors require current greater than 250mA. Most of the ICs like 555 timer,74 series ICs cannot supply this amount of current.Instead if we directly connect motors to the output of any of the above IC's, they might get damaged. There is a need of a circuitry that can act as a bridge between the above mentioned ICs and the motors. This is where a motor driver plays a crucial role. It regulates the current flowing through the circuit hence preventing any damage to the device. Different motor driver circuits can be classified as ones:

- Using Transistor
- Using L293D/L298
- Using relays

For controlling motor in both directions H bridge circuit is used. Its working is very simple and is described below.

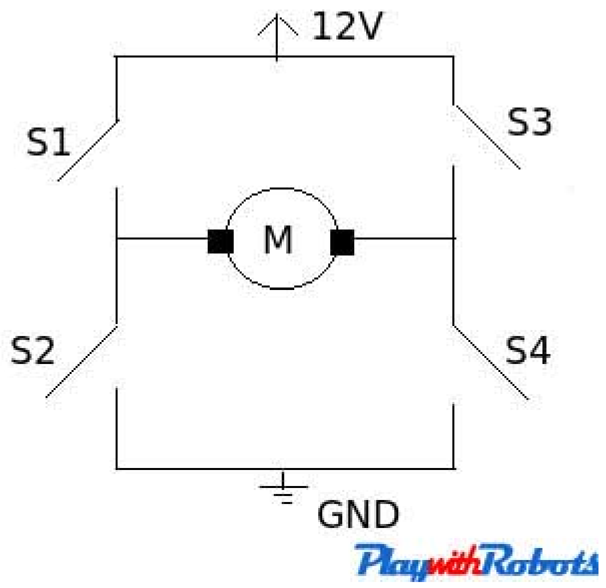


Figure 1: [2]

Closed Switches	Open Switches	Voltage across motor	Motion
Nil	S1,S2,S3,S4	0	No motion
S1,S4	S2,S3	12V	Clockwise
S2,S3	S1,S4	-12V	Anti-clockwise
S1,S3	S2,S4	0V	Brake

Ref: [2]

L293D Motor Driver

L293D is dual H-bridge motor driver ICs. Using these we can control the rotation of two motors in both clockwise and anti-clockwise direction.

Pin configuration

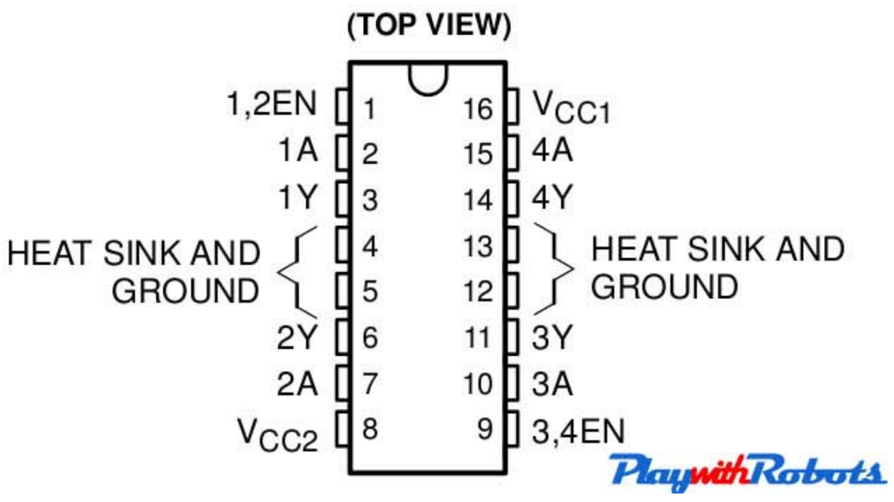


Figure 2: [2]

The description of each pin is as follows:

- Enable pins: These are pin no. 1 and pin no. 9. Pin no. 1 is used to enable Half-H driver 1 and 2.(H bridge on Left side). Pin no. 9 is used to enable H-bridge driver 3and 4.(H bridge on right side). The concept is simple, if you want to use a particular H bridge you have to give a high logic to corresponding enable pins along with the power supply to the IC. This pin can also be used to control speed of the motor using PWM technique.
- VCC1 (Pin 16): Power supply pin. Connect it to 5V supply.
- VCC2 (Pin 8): Power supply for motor. Apply +ve voltage to it as per motor rating. If you want to drive your motor at 12V, apply 12V on this pin. It is also possible to drive motor directly on a battery, other than the one used for supplying power to the circuit, Just connect +ve terminal of that battery to VCC2 pin and make GND of both the batteries common. (MAX voltage at this pin is 36V as per its datasheet).
- GND (Pins 4,5,12,13): Connect them to common GND of circuit.
- Inputs (Pins 2,7,10,15): These are input pins through which control signals are given by microcontrollers or other circuits/ICs. For

example, if on pin 2 (Input of 1st half H driver) we give Logic 1 (5V), we will get a voltage equal to VCC2 on corresponding output pin of 1st half H driver i.e pin no. 3. Similarly for Logic 0 (0V) on Pin 2, 0V on Pin 3 appears.

- Outputs (Pin 3,6,11,14): Outputs pins. According to input signal output signal comes. [2]

Circuit Diagram

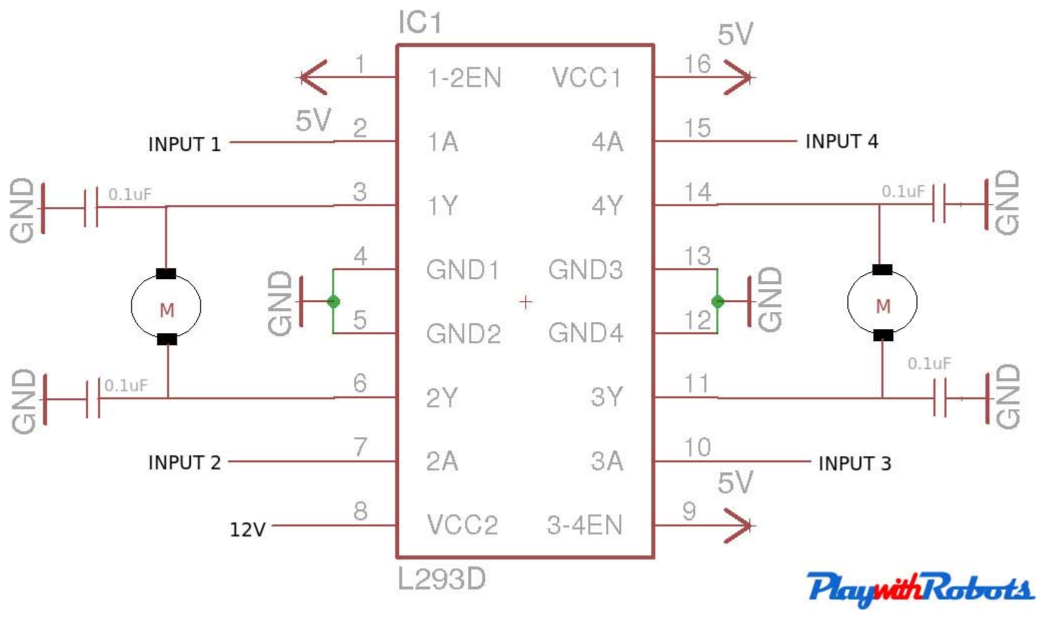


Figure 3: [2]

1A	2A	1Y	2Y	Motor 1
Logic 0	Logic 0	0	0	Stop
Logic 1	Logic 0	12V	0	Clockwise
Logic 0	Logic 1	0	12V	Anti-clockwise
Logic 1	Logic 1	12V	12V	Brake

Ref: [2]

PWM

There are many different ways to control the speed of motors but one very simple and easy way is to use Pulse Width Modulation. The use of pulse

width modulation to control a small motor has an advantage that the power loss in the switching transistor is small because the transistor is either fully ON or fully OFF. As a result the switching transistor has a much reduced power dissipation giving it a linear type of control which results in better speed stability. Also the amplitude of the motor voltage

remains constant so the motor is always at full strength. The result is that the motor can be rotated much more slowly without it stalling.[3]

PWM can be performed in a number of ways on the Raspberry Pi:

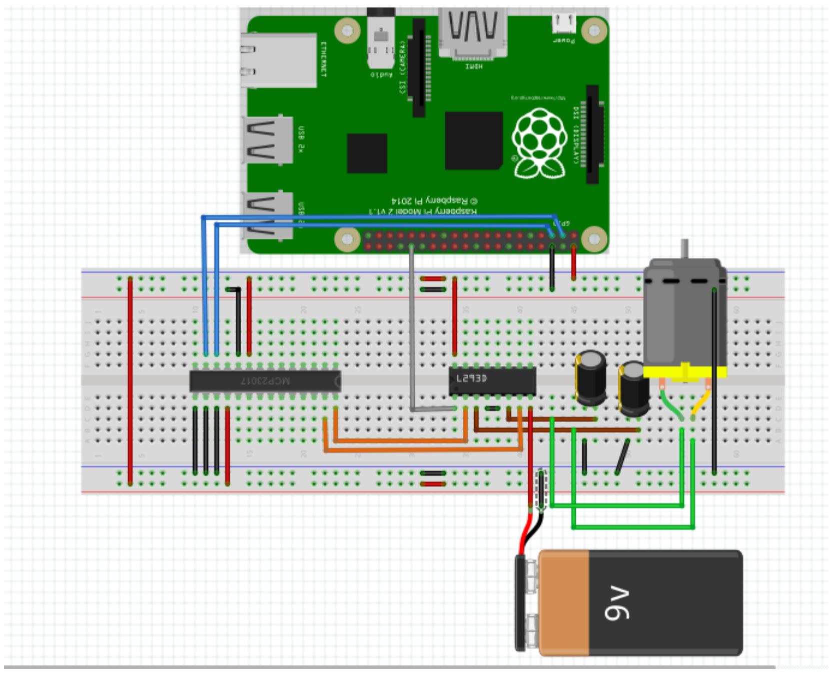
1. Inbuilt hardware; The Pi can perform PWM in hardware, but this can only be done on one pin (GPIO18 in R-Pi 1) and when enabled it interferes with the audio jack.

2. Software(using delay function); PWM can be performed in software. This is a very easy option. It isnt as precise as hardware PWM however in most instances software PWM works well on all GPIO pins .

6 Experiment

PWM generation for velocity control DC motor interfaced with MCP23017 IC

In this experiment we will be controlling a DC motor using software pwm generation technique.



As shown in the figure:

- Pin 9 (VDD) is connected to 5V (Red)
- Pin 10 (VSS) is connected to Ground (Black)
- Pin 12 (SCL) is connected to Pin 5 on the Pi GPIO (Blue)
- Pin 13 (SDA) is connected to Pin 3 on the Pi GPIO (Blue)
- Pin 18 (Reset) should be set high for normal operation so we connect this to 3.3V (Red)
- Pins 15, 16 & 17 (A0-A2) determine the number assigned to this device. We are only using one device so we will give it a binary zero by setting all three of these pins to 0 (ground) (Black)
- 1-2EN pin of L293D is connected to GPIO 12 of R-Pi (Grey)
- Input 1 and Input 2 of L293D is connected to GPB0 and GPB1 of MCP23017 (Orange)

- Out 1 and Out 2 are connected to a DC motor

Code

```
import smbus  # module to access i2c based interfaces
import time
import RPi.GPIO as GPIO  # module to control Pi GPIO channels

GPIO.setmode(GPIO.BOARD) # to use Raspberry Pi board pin numbers
GPIO.setup(12,GPIO.OUT) # set up GPIO output channel i.e. on pin 11

bus = smbus.SMBus(1) # Rev 2 Pi uses 1

DEVICE = 0x20 # Device address (A0-A2)
IODIRB = 0x01 # Pin direction register
OLATB  = 0x15 # Register for outputs

en = GPIO.PWM(12, 50) #dc motor enable pin,channel=12
                      #frequency=50Hz (software pwm specifications)
en.start(0) # start pwm with duty cycle 0

# all bits of IODIRB register are set to 0 meaning GPA pins are outputs
bus.write_byte_data(DEVICE,IODIRB,0x00)

# Set output all 7 output bits of GPB to 0
bus.write_byte_data(DEVICE,OLATB,0)

# Function name :forward
# Input : None
# Output : Motor moves in clockwise direction i.e. forward
# Example call: forward()
def forward():
    bus.write_byte_data(DEVICE,OLATB,0b00000001
        # one input to the motor is set to logic high and the
    # other input is set to logic low resulting in clockwise motion
    time.sleep(1)

# Function name :backward
# Input : None
# Output : Motor moves in anticlockwise direction i.e. backward
# Example call: backward()
def backward():
    bus.write_byte_data(DEVICE,OLATB,0b00000010)
        # one input to the motor is set to logic low and the
        # other input is set to logic high resulting in
        # anticlockwise motion
```

```
      time.sleep(1)


try:
    while 1:
             forward()
         for dc in range(0,100,10):
                    en.ChangeDutyCycle(dc) # duty cycle keeps
                                # changing by 10% starting from 0
                    time.sleep(0.5)

         en.ChangeDutyCycle(0)
             backward()
         for dc in range(0,100,10):
                    en.ChangeDutyCycle(dc) # duty cycle keeps changing
                                # starting from 0
                    time.sleep(0.5)



except KeyboardInterrupt:
       pass
       GPIO.cleanup() # all GPIO pins are cleared
```

7 Appendix

7.1 Raspberry Pi 2 Pin-out Diagram

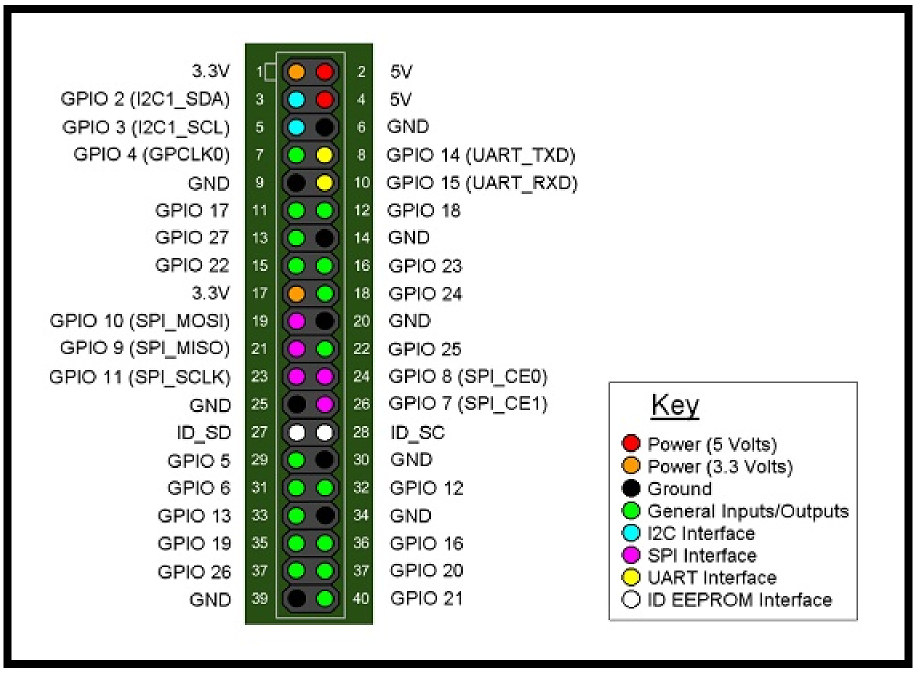


Figure 4: [4]

7.2 MCP23017 datasheet

`http://ww1.microchip.com/downloads/en/DeviceDoc/21952b.pdf`

8 References

1. `https://en.wikipedia.org/wiki/DC_motor`
2. `http://playwithrobots.com/dc-motor-driver-circuits/`
3. `http://www.electronics-tutorials.ws/blog/pulse-width-modulation.html`
4. `http://data.designspark.info/uploads/images/53bc258dc6c0425cb44870b50ab30621`