

**Lab Manual for Embedded System Design**

**Lab No. 13**

**Raspberry PI Hardware Applications**

*Objectives*

*Design and Interface Raspberry*

*-*

*PI*

*Hardware Applications.*

*Design Traffic Lights and Pulse Width Modulation (PWM)*

*circuits*

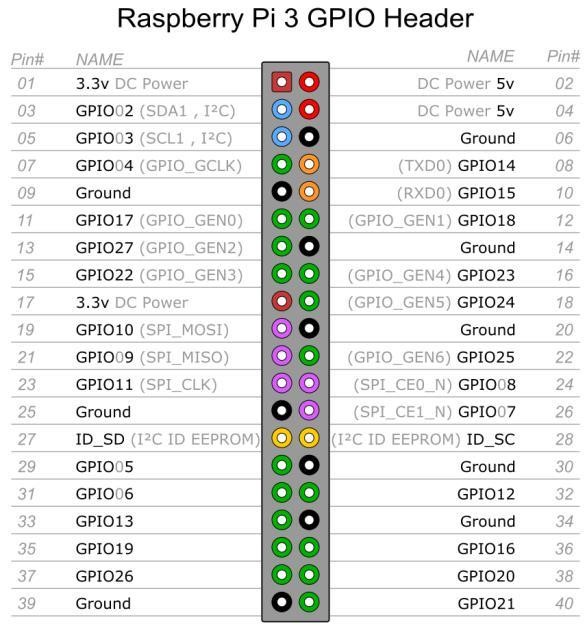
*.*

# LAB # 13

**Raspberry PI Hardware Applications**

## Introduction

***Raspberry – Pi GPIO Controller:***

* The R-Pi GPIO connector actually has a number of different types of connection on them. There are:
* True GPIO (General Purpose Input Output) pins that you can use to turn LEDs on and off etc.
* I2C interface pins that allow you to connect hardware modules with just two control pins
* SPI interface with SPI devices, a similar concept to I2C but a different standard
* Serial Rx and Tx pins for communication with serial peripherals
* In addition, some of the pins can be used for PWM (pulse Width Modulation) for power control and another type of pulse generation for controlling servo motors called PPM

(Pulse Position Modulation). Fig. 9.1: GPIO Inputs

**PWM:**

Pulse width modulation (PWM) is a powerful technique for controlling analog circuits with a microprocessor's digital outputs.

## Time Boxing

|  |  |  |
| --- | --- | --- |
| **Activity Name** | **Activity Time** | **Total Time** |
| Login Systems + Setting up Raspberry PI Environment | 3 mints + 5 mints | 8 mints |
| Walk through Theory & Tasks | 60 mints | 60 mints |
| Implement Tasks | 80 mints | 80 mints |
| Evaluation Time | 30 mints | 30 mints |
|  | Total Duration | 178 mints |

## Objectives

1. Design and Interface Raspberry-PI Hardware Applications.
2. Design Traffic Lights and Pulse Width Modulation (PWM) circuits.

## Lab Tasks/Practical Work

1. In this task we will write a simple Python script in LXTerminal to glow LED using GPIO 7 of Raspberry-Pi.

*import RPi.GPIO as GPIO GPIO.setmode (GPIO.BOARD)*

*GPIO.setup (7, GPIO.OUT)*

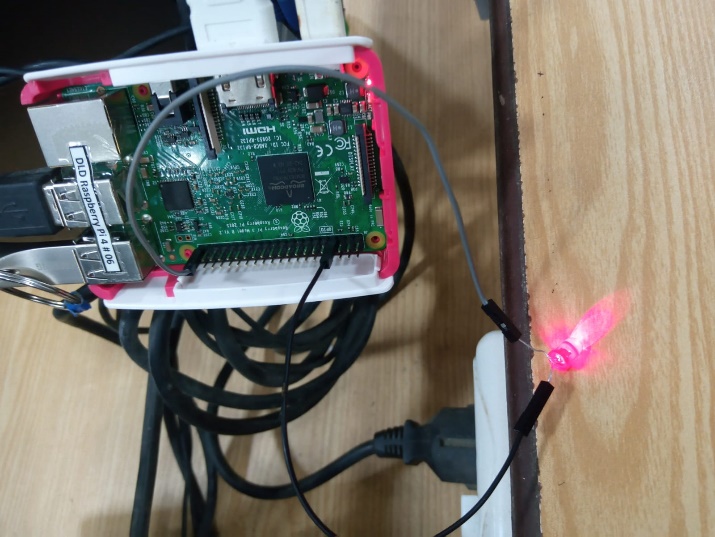
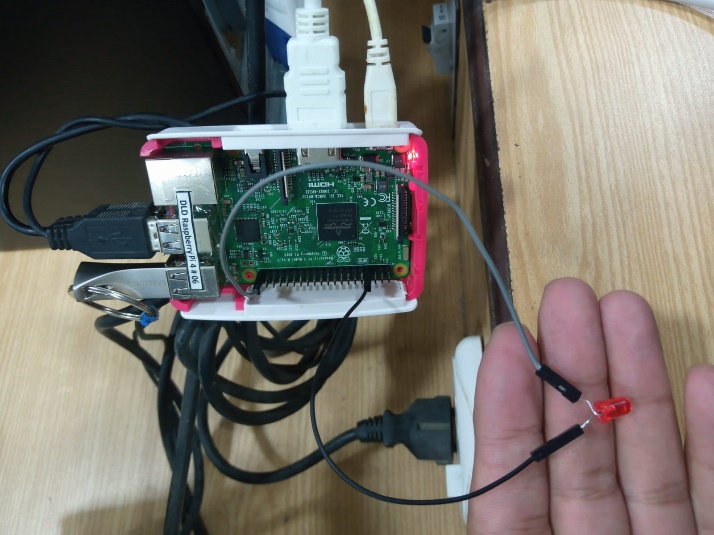
*GPIO.output (7, True)*



1. In this task we will write a simple Python script in LXTerminal to blink single LED using a GPIO 7 of Raspberry-Pi.

*import RPi.GPIO as GPIO import time*

*GPIO.setmode (GPIO.BOARD) GPIO.setup (7, GPIO.OUT) While True:*

*GPIO.output (7, True) time.sleep (0.5) GPIO.output (7, False) time.sleep (0.5)*

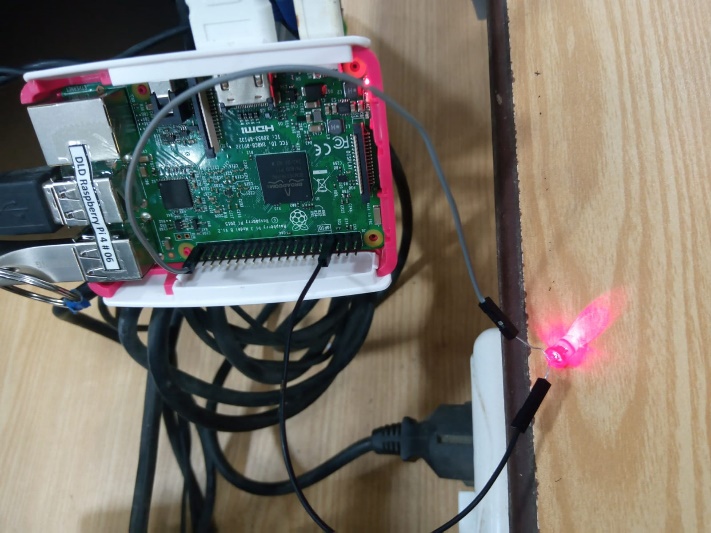
1. In this task we will write a simple Python script to control LED using GPIOs as PWM application.

*import RPi.GPIO as GPIO GPIO.setmode(GPIO.BOARD) GPIO.setup(7, GPIO.OUT) pwm\_led = GPIO.PWM(7, 500)*

*pwm\_led.start(100) while True:*

*duty\_s = input("Enter Brightness (0 to 100):")*

*duty = int(duty\_s) pwm\_led.ChangeDutyCycle(duty)*



1. In this task we will write a simple Python script for Traffic control signals using various GPIOs.



*import RPi.GPIO as GPIO*

*import time GPIO.cleanup()*

*GPIO.setmode(GPIO.BOARD)*

*GPIO.setup(3,GPIO.OUT)*

*GPIO.setup(5,GPIO.OUT)*

*GPIO.setup(7,GPIO.OUT) GPIO.setwarnings(False) while True:*

*GPIO.output(7,GPIO.HIGH)* *time.sleep(5)*

*GPIO.output(5,GPIO.HIGH) time.sleep(2)*

*GPIO.output(7,GPIO.LOW)*

*GPIO.output(5,GPIO.LOW) GPIO.output(3,GPIO.HIGH) time.sleep(10)*

*GPIO.output(3,GPIO.LOW)*

*GPIO.output(5,GPIO.HIGH)*

4. Write a simple Python script to drive dc motors in both directions.

Solution:

import RPi.GPIO as GPIO

import time

# Set the GPIO mode to BCM numbering

GPIO.setmode(GPIO.BCM)

# Set the pin numbers for the motor control pins

motor\_pin\_1 = 18

motor\_pin\_2 = 17

# Set the motor control pins as output pins

GPIO.setup(motor\_pin\_1, GPIO.OUT)

GPIO.setup(motor\_pin\_2, GPIO.OUT)

# Function to drive the motor in a given direction

def drive\_motor(direction):

if direction == "forward":

# Set the motor pins to drive the motor forward

GPIO.output(motor\_pin\_1, GPIO.HIGH)

GPIO.output(motor\_pin\_2, GPIO.LOW)

elif direction == "reverse":

# Set the motor pins to drive the motor in reverse

GPIO.output(motor\_pin\_1, GPIO.LOW)

GPIO.output(motor\_pin\_2, GPIO.HIGH)

# Drive the motor forward for 2 seconds

drive\_motor("forward")

time.sleep(2)

# Drive the motor in reverse for 2 seconds

drive\_motor("reverse")

time.sleep(2)

# Turn off the motor

GPIO.output(motor\_pin\_1, GPIO.LOW)

GPIO.output(motor\_pin\_2, GPIO.LOW)

GPIO.cleanup()

Output:

