Experiment NO: 08.

Name: Kajal Sunil Pagare.

Rollno: 26 Div: B

class: TE.

Aim : Write an application Using Raspbeerry Pi / Beagle board to. Control the operation of h/w Simulated traffic signal.

Theory

Attaching the traffic Lights:

The low Voltages labs traffic lights Connects to the pi Using four pins. One of these needs to be ground, the other three being actual GPIO pins. Used to control each of the individual LED's Before powering up the pi, attach the traffic lights so that the pins connects to the GPIO pins highlighted inred.

Programming the traffic lights:

First, you need to install a couple of extra s/w packages needed to allow you to download my samples codes, and to give python access to the GPIO pins On pi. Enter the following command line sudo apt-get install python -dev python -rpi.gpio git.

How It works.

The code for this is very simple. It work. starts by importing the RPi.GPIO library, plus time which gives us a wait function ,signals that allows us to trap the signal sent when the usertries to quit the program and sys so we can send an. a appropriate exist signal back to the os before terminating.

```
import RPi.GPIO as GPIO.
import time
import signal
import sys.
```

Next we put the GPIO library into "BCN" or Broadcom" mode (so we can refer to pins by the same numbers as are labelled with in GPIO pin diagrams) ,and sets pins9 (red LED), 10 (amber LED) and all"(green LED) to be used as outputs.

```
# setup :
  GPIO. Setmode (GPIO. BCM)
  GPIO. Setup (9, GPIO. OUT)
  GPIO. Setup (10, GPIO. OUT)
  GPIO. Setup (11, GPIO. OUT)
```

The main part of the program will run. in an infinite loop until the user exists it by stopping python with ctrl c. It's a good idea to add a handler functions that will run whenever this happens, so that we can turn off all the lights prior to existing (thus ensuring they'll also be in the state we expect them to start in the next time the program run).

```
# turn off all lights, when user
  ends demo
def allLights off (Signal, frames):
    GPIO. output (9, false)
    GPIO. output (10, false)
    GPIO, output (11, false)
    GPIO, Cleanup()
     sys. exit(0)
  Signal. Signal (Signal. SIGNIT, all
                   LightsOff).
```

The main body of the code then consists of an infinite while loop that turns on the the redlight (pin 9) waits, turns, on the amber light (pin 10). waits, then cycles through the rest of the traffic. light pattern by turning the approximate LED's on and off.

When control-C is pressed an interrupt signal . SIGINT is sent. This is handled by the all lights off function that Switches all the lights off, tidlies up the GPIO library states and exists cleanly back to the operating system.

Conclusion → Thus, we have implemented the application for traffic signals using Raspberry pi.