Practical no: 8

Title: Write an application using Raspberry-Pi /Beagle board to control the operation of a hardware simulated traffic signal.

Name: Aditi Dinesh Mulay

Class: T.E. Computer

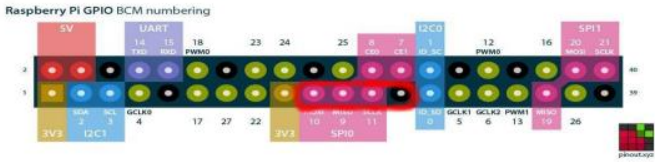
Subject: ES&IOT

Div: A

Roll no: 02

PRN No. 71918146B

Before powering up the Pi, attach the traffic lights so that the pins connect to the GPIO pins highlighted in red:



ESIOT Practical 8

Aditi Dinesh Mulay
T.E. Comp Div: A
Roll no: 02

Aim

Write an application using Raspberry-Pi / Beagle board to control the operation of a hardware simulated traffic signal.

Theory

Attaching the Traffic Lights.

The low voltage Labs Traffic Lights connect to Pi using 4 pins. One of these needs to be ground, the other three being actual GPIO pins used to control each of individual LEDs.

Before providing up the Pi, attach the traffic lights so that the pins connect to the GPIO pins highlighted in red.

Programming the Traffic Lights

First you need to download my sample code, & to give Python access to the GPIO pins on Pi; Enter following at command line:

sudo apt-get install python-dev python-rpi.gpio git

How it works.

The code for this is very simple. It starts by importing the Rpi.GPIO library, plus time which gives

us a timed wait function, signal that allows us to trap the signal sent when the user tries to quit the program and system so we can send an appropriate exit signal back to O.S. before terminating.

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

Next we put GPIO library into "BCM" or "Broadcom" mode and sets pins 9, 10, 11. to be used as o/p:

```
# setup

GPIO.setmode(GPIO.BCM)
GPIO.setup(9, GPIO.OUT)
GPIO.setup(10, GPIO.OUT)
GPIO.setup(11, GPIO.OUT)
```

The main part of prg. will run in an infinite loop until the user exists it by stopping Python with Cntrl C It's a good idea to add a handler function that will run whenever this happens, so that we can turn off all the lights prior to existing:

```
# Turn off all lights when user ends demo
def allLightsOff(signal, frame):
        GPIO.output(9, False)
        GPIO.output(10, False)
        GPIO.output(11, False)
        GPIO.cleanup()
        Sys.exit(0)
signal.signal(signal.SIGHT, allLightsOFF)
```

The main body of the code then consists of an infinite while loop that turns on the red light, waits, turns on the amber light, waits, then cycles through the rest of traffic light pattern by turning the appropriate LED's on & off.

When cntrl-C pressed an interrupt signal SIGHT is sent. This is handled by all the lights off function that switches all the Lights off, tidies up the GPIO library state & exists cleanly back to the O.S.

Conclusion -

Thus, we have implemented the application for traffic signals using Raspberry Pi.