ASSIGNMENT NUMBER: C2

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

Pre-requisite:
Basic knowledge of GPIO of Raspberry pi/BBB
Basic knowledge of Python programming.
Working and connections of sensors/actuators.

Learning Objectives:
Understanding the controlling of devices through Raspberry Pi /Beagle board.

Learning Outcomes:
The students will be able to
To simulate traffic signal through LEDs.
To control this simulated traffic signal through Raspberry Pi /Beagle board.
Can perform actuation.

H/W AND S/W Requirements :
Raspberry Pi/Beagle board Development Boards
PC / Monitor/Keyboard
Raspbian (OS), Debian LINUX and Python

Theory:
Beagle Bone black is an open hardware, community-supported embedded computer for developers. It works with 1GHz with the SitaraTM ARM® Cortex-A8 processor. The REV C comes with Debian Linux pre installed. It has GPIO(69 max), which can be programmed
using Python.

LEDs are the light emitting Diodes. LEDs have two wires. One wire is the anode (positive)
and another is the cathode (negative). Push the LED leads into the breadboard, with the longer (positive) lead towards the top of the breadboard. It does not matter which way around
the resistor goes. The top two connections on the BBB expansion header we are using (P8)
are both GND. The other lead is connected to pin 10, which is the right-hand connector on
the fifth row down.

Steps to do :

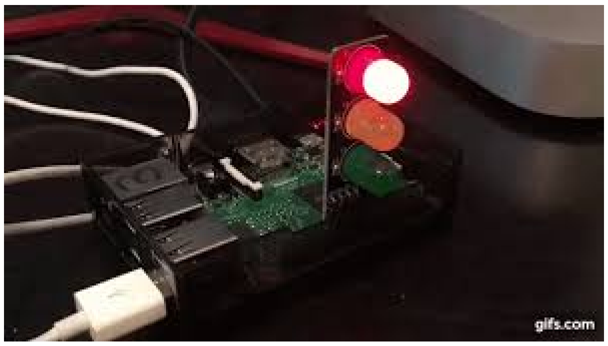
1. Connect BBB board to Machine using USB cable.

. Install Python library to perform I/O programming on GPIOs
3. Connect Red LED to P8 pin no 10
4. Connect Yellow LED to P8 pin no 12
5. Connect Green LED to P8 pin no 14
6. Reset all 3 LEDs using GPIO.OUT command
7. Turn on Red LED using GPIO.HIGH command
8. Put some delay
9. Turn off Red LED using GPIO.LOW command
10. Repeat steps 7, 8, 9 for Yellow LED
11. Repeat steps 7, 8, 9 for Green LED
12. Go to step

Code:

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

red_led_a = 22
yellow_led_a = 24
green_led_a = 26
#print "Hello-A"
red_led_b = 32
yellow_led_b = 31
green_led_b = 16
#print "Hello-B"
red_led_c = 15
yellow_led_c = 13
green_led_c = 11
#print "Hello-C"
red_led_d = 40
yellow_led_d = 38
green_led_d = 37
#print "Hello-D"
#--------- high low logic
HIGH=1
LOW=0
#------------------ Rpi Config
RUNNING= True
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

#------------------ pin config
GPIO.setup(red_led_a,GPIO.OUT)
GPIO.setup(yellow_led_a,GPIO.OUT)
GPIO.setup(green_led_a,GPIO.OUT)
GPIO.setup(red_led_b,GPIO.OUT)
GPIO.setup(yellow_led_b,GPIO.OUT)
GPIO.setup(green_led_b,GPIO.OUT)

GPIO.setup(red_led_c,GPIO.OUT)
GPIO.setup(yellow_led_c,GPIO.OUT)
GPIO.setup(green_led_c,GPIO.OUT)

GPIO.setup(red_led_d,GPIO.OUT)
GPIO.setup(yellow_led_d,GPIO.OUT)
GPIO.setup(green_led_d,GPIO.OUT)

#---------------------------- all def
'''
    Def for only one Lane Green On other low
'''
def Greenon(red, yellow, green):
    GPIO.output(green, HIGH)
    GPIO.output(red, LOW)
    GPIO.output(yellow, LOW)
    red_on(red)
    all_green_low(green)
    all_yellow_low()


'''
 Def for all red Signal on Except current Green On
'''
def red_on(r):
    GPIO.output(red_led_a, HIGH)
    GPIO.output(red_led_b, HIGH)
    GPIO.output(red_led_c, HIGH)
    GPIO.output(red_led_d, HIGH)
    GPIO.output(r, LOW)

def all_green_low(g):
    GPIO.output(green_led_a, LOW)
    GPIO.output(green_led_b, LOW)
    GPIO.output(green_led_c, LOW)
    GPIO.output(green_led_d, LOW)
    GPIO.output(g, HIGH)

def all_yellow_low():
    GPIO.output(yellow_led_a, LOW)
    GPIO.output(yellow_led_b, LOW)
    GPIO.output(yellow_led_c, LOW)
    GPIO.output(yellow_led_d, LOW)

def yellow_high(y):
    GPIO.output(yellow_led_a, LOW)
    GPIO.output(yellow_led_b, LOW)
    GPIO.output(yellow_led_c, LOW)
    GPIO.output(yellow_led_d, LOW)
    GPIO.output(y, HIGH)


# Main loop
try:
    while RUNNING:
        # Green for 13 seconds LA other LB LC LD RED
        #---------- LA
        Greenon(red_led_a,yellow_led_a,green_led_a)
        time.sleep(3); #green time
        yellow_high(yellow_led_a)
        time.sleep(2);
        #---------- LB
        Greenon(red_led_b,yellow_led_b,green_led_b)
        time.sleep(3);
        yellow_high(yellow_led_b)
        time.sleep(2);
        #---------- LC
        Greenon(red_led_c,yellow_led_c,green_led_c)
        time.sleep(3);
        yellow_high(yellow_led_c)
        time.sleep(2);
        #---------- LD
        Greenon(red_led_d,yellow_led_d,green_led_d)
        time.sleep(3);
        yellow_high(yellow_led_d)
        time.sleep(2);
# If CTRL+C is pressed the main loop is broken
except KeyboardInterrupt:
    RUNNING = False
    print "\Quitting"

# Actions under 'finally' will always be called
finally:
    # Stop and finish cleanly so the pins
    # are available to be used again
    GPIO.cleanup()
```

Output:



Conclusion:

Successfully Controlling the operation of a hardware simulated traffic signal using Raspberry Pi /Beagle board circuit