Assignment 8

Name : Shardul Vikram Dharmadhikari

Class : TE Comp

Roll No: 8016

Title:

Writing an application using Raspberry-Pi board to control the operation of hardware simulated traffic signal.

Aim/Objectives:

- To simulate the 4 lane Traffic signal working using Raspberry Pi board model

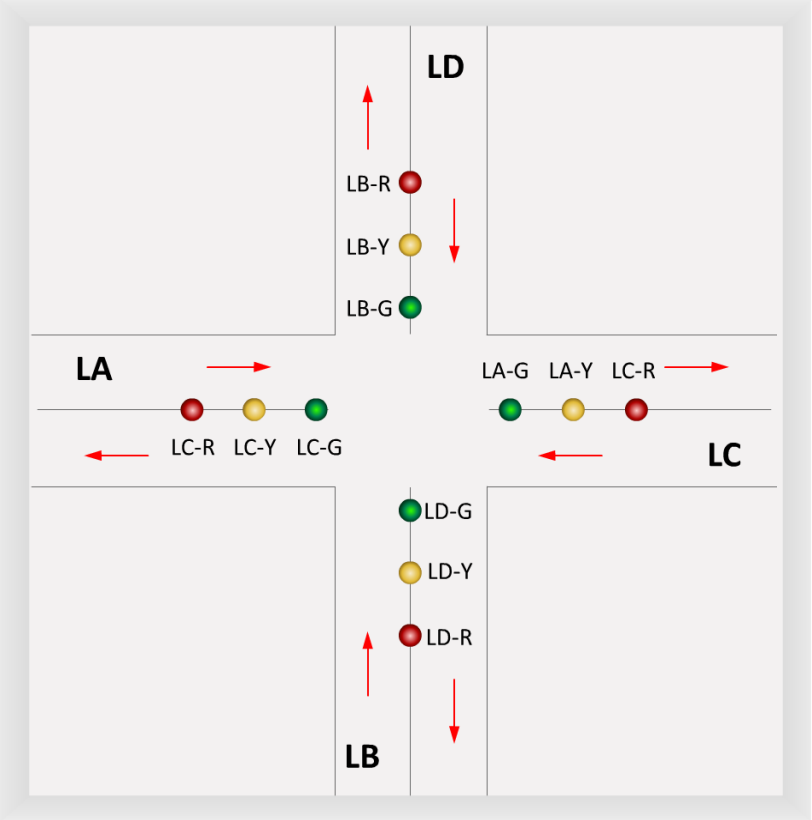
Software:

- Raspbian OS (IDLE)

Hardware Modules:

- Raspberry Pi Board module
- Red Leds (qty. 4)
- Yellow Leds (qty. 4) Green Leds (qty. 4)

Theory:



- A simple traffic light system for a 4 way intersection is implemented using Raspberry pi where the traffic is controlled in a pre-defined timing system.
- There are 4 lanes LA, LB, LC and LD going towards the signal.
- At the cross road there are 4 sets of Traffic lights opposite to each lane.
- These sets are LA(LA-G, LA-Y, LA-R), LB(LB-G, LB-Y, LB-R), LC(LC-G, LC-Y, LC-R), LB(LD-G, LD-Y, LD-

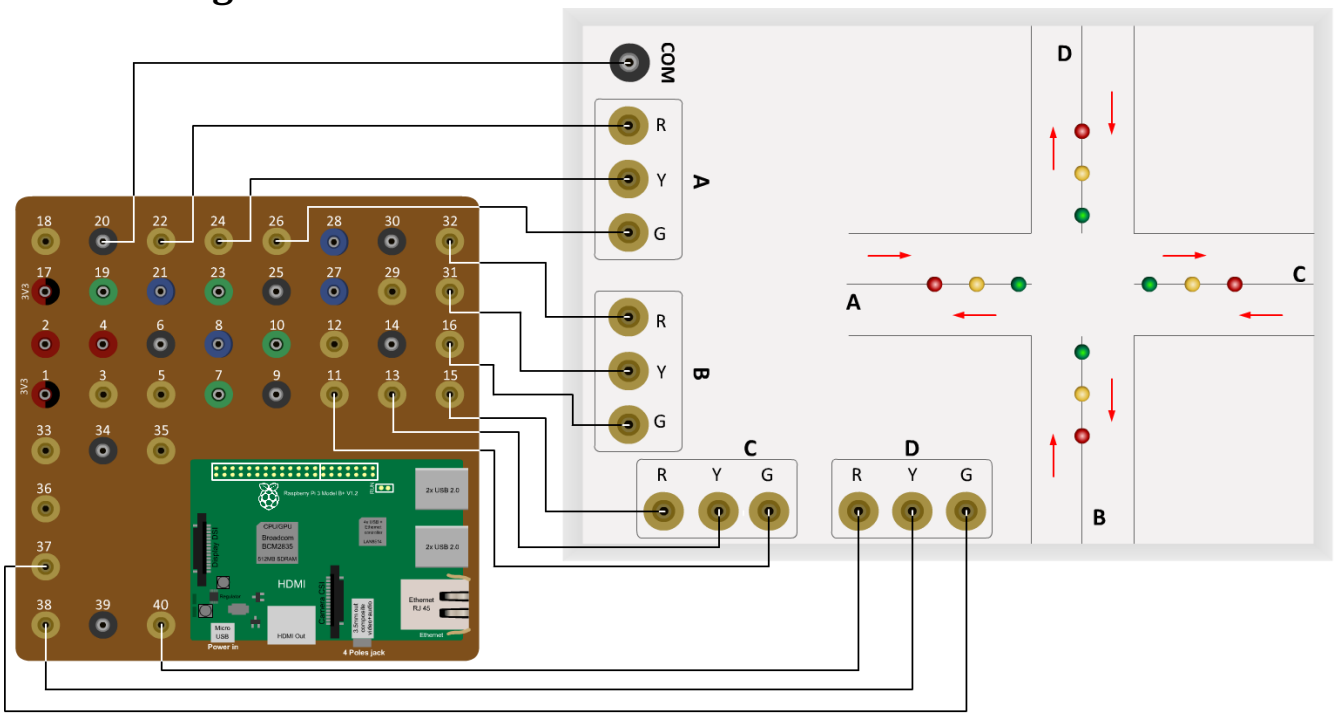
R),

- Traffic from any lane moves when its corresponding Green light is ON.
- "ON time" of any Red light is dependent on the "ON time" of Yellow light and Green light of other 3 signal lights. • "ON time" of Yellow light is same for all lanes.
- User can specify and change the "ON time" of the Green light and Red light of each signal separately.
- The Traffic light pattern keeps on repeating till the next change made by the user.

Safety precautions:

- Raspberry-Pi provides 3.3V and 5V VCC pins Raspberry-Pi operates on 3.3V.
- Various sensors and actuators operate on different voltages.
- Read datasheet of a given sensor or an actuator and then use appropriate VCC pin to connect a sensor or an actuator.
- Ensure that signal voltage coming to the Raspberry-Pi from any sensor or actuator does not exceed 3.3V.
- If signal/data coming to Raspberry-Pi is greater than 3.3V then use voltage level shifter module to decrease the incoming voltage.
- The Raspberry-Pi is a costly device, hence you should show the circuit connections to your instructor before starting your experiment.

Interface diagram:



Steps for assembling circuit:

- Connect the R, Y, G pins of “Lane A” to 22, 24, 26 pins of Raspberry Pi module respectively. Connect the R, Y, G pins of “Lane B” to 32, 31, 16 pins of Raspberry Pi module respectively. Connect the R, Y, G pins of “Lane C” to 11, 13, 15 pins of Raspberry Pi module respectively.
- Connect the R, Y, G pins of “Lane D” to 40, 38, 37 pins of Raspberry Pi module respectively.
- Connect the “COM” pin of the Traffic Signal module to the GND pin of Raspberry Pi module.

Procedure:

- Write the program as per the algorithm given.
- Save the program
- Run code using Run module.

Algorithm:

- Import RPi.GPIO library
- Import Time library
- Declare all the LED pins which is connected to the GPIO pins of Raspberry pi board
- Set mode i.e. GPIO.BOARD
- Take delay time from user for each lane
- Set all the LED pins as Output
- Define 4 functions to control the traffic light as
    a. trafficState1
    b. trafficState2
    c. trafficState3
    d. trafficState4
- In main loop,
    a. Firstly for the LA, the signal becomes Green.
    b. Hence, for all other Lanes (LB,LC,LD), the corresponding Red signal is on.
    c. After a time delay, as a warning indicator, the Yellow light in LA signal is turned on indicating that the red light is about to light up.
    d. After a time delay for the Lane 3, the signal becomes Green. So at the same time the signal for Lane 1 becomes Red.
    e. Second time for the LB, the signal becomes Green.
    f. Hence, for all other Lanes (LA,LC,LD), the corresponding Red signal is on.
    g. After a time delay, as a warning indicator, the Yellow light in LB signal is turned on indicating that the red light is about to light up.
    h. Third time for the LC, the signal becomes Green.
    i. Hence, for all other Lanes (LA,LB,LD), the corresponding Red signal is on.
    j. After a time delay, as a warning indicator, the Yellow light in LC signal is turned on indicating that the red light is about to light up.
    k. Fourth time for the LD, the signal becomes Green.
    l. Hence, for all other Lanes (LA,LB,LC), the corresponding Red signal is on.
    m. After a time delay, as a warning indicator, the Yellow light in LD signal is turned on indicating that the red light is about to light up.
    n. The process mentioned above will be repeated all over again.

Observation:

- Observe the output on LEDs and Seven Segment Display.

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()
```