Embedded Systems Project

Designing and Implementation of Traffic Lights System using Python as Coding Language.

Name: Ms Sushama. S. Shetty

Department: Masters of Information Technology

Organization: SRH University

City, Country: Heidelberg, Germany

Email-id: 11013334@stud.hochschule-heidelberg.de

Abstract

The purpose of this project is to design and implement a traffic light system consisting of two traffic lights that control the traffic on a narrow bridge with only one lane. This system will be built using Raspberry-pi as a development platform and programming will be done using python as coding language. The LED's (Red, Amber and Green) must be controlled using PCA9685 in PWM (Pulse Width Modulation).

Keywords: Raspberry-pi, I2C Bus, PCA9685, Case structure, LED's, Python.

1. INTRODUCTION:

Traffic light is considered to be an optical signalling device which can be used at places related to traffic such as roadways, rail networks, pedestrian streets etc. The main aim of the traffic lights is to ensure safety and hence this paper gives an insight on controlling the traffic lights with PCA9685 in PWM and giving a visual representation. For implementing this project, we require components such as Raspberry-pi, PCA9685 module, Jumper wires, Breadboard, LED's (2*(Red, Amber, Green)), Resistors (approx. 300ohms)

2. Description of Components:

2.1. Raspberry Pi (3 B):

- The **Raspberry-Pi** is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries.The original model became far more popular than anticipated, selling outside its target market for uses such as robotics. It now is widely used even in research projects, such as for weather monitoring because of its low cost and portability. It does not include peripherals (such as keyboards, mouse).
- **Raspberry Pi 3 Model B** was released in February 2016 with a 1.2 GHz 64-bit quad core processor, on-board 802.11n Wi-Fi, Bluetooth and USB boot capabilities.

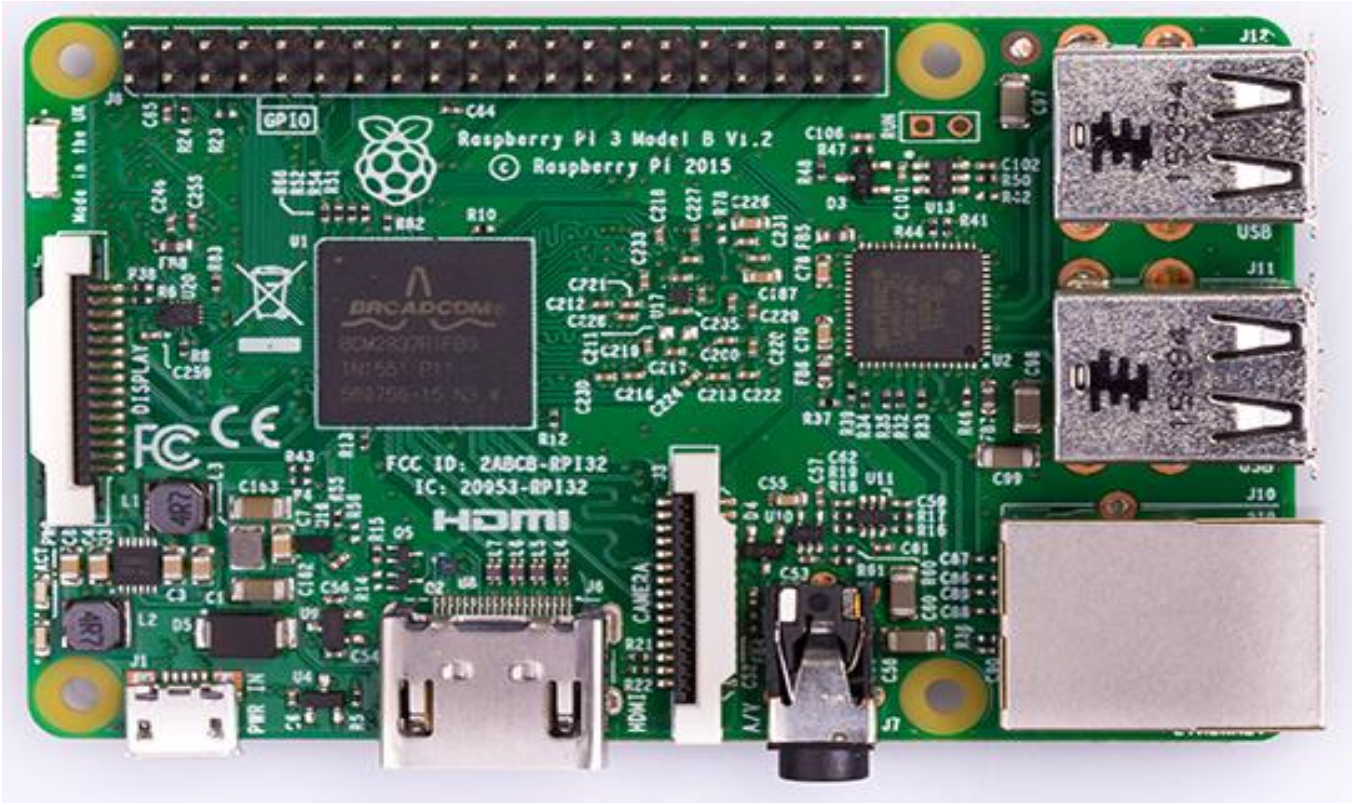

Fig 2.1.1: Hardware Raspberry-pi 3 B

2.2.PCA9685:

- The PCA9685 is an I²C-bus controlled 16-channel LED controller optimized for Red/Green/Blue/Amber (RGBA) colour backlighting applications.
- Each LED output has its own 12-bit resolution (4096 steps) fixed frequency individual PWM controller that operates at a programmable frequency from a typical of 24 Hz to 1526 Hz with a duty cycle that is adjustable from 0 % to 100 % to allow the LED to be set to a specific brightness value.
- All outputs are set to the same PWM frequency.

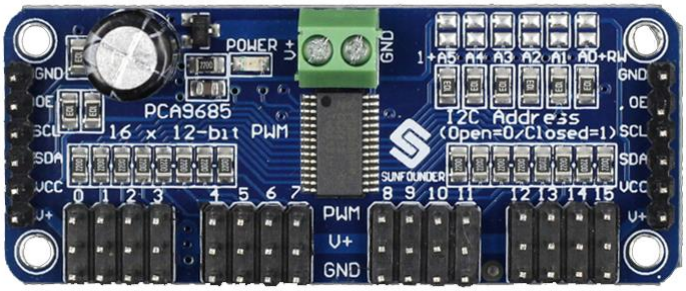

Fig 2.2.1: Hardware PCA9685

2.3. Breadboard, LEDs and Resistors:

- A **breadboard** is a construction base for prototyping of electronics.
- A **light-emitting diode** (**LED**) is a semiconductor light source that emits light when current flows through it.
- LEDs have many advantages over incandescent light sources, including lower energy consumption, longer lifetime, improved physical robustness, smaller size, and faster switching.
- A **resistor** is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses.

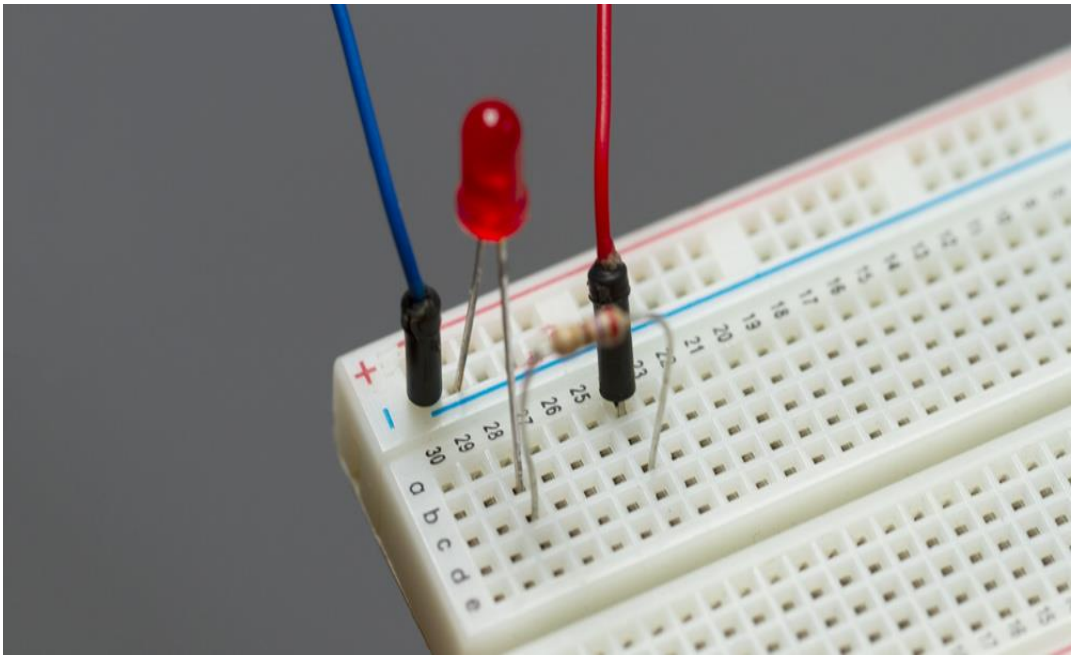

Fig 2.3.1: Hardware Breadboard, LED and Resistor

3. Description of the Project:

Here in the project we need to implement a traffic light system which has two traffic lights that is used for controlling the traffic for a narrow bridge with only one lane. Hence, we will consider some conditions which is called as phases with respect to the programming language so that we can ensure safety and no accidents take place. PCA9685 module is in respect with PWM which is used for controlling the traffic lights. The Raspberry-pi is connected with PCA9685 via I2C bus. The Inter-Integrated circuit is used for internal communication between different circuitry parts, such as between a controller and other peripherals. With the given connections of Raspberry-pi and PCA9685, we can then mount the remaining components on the breadboard that is the leds and the resistors. Batteries would be needed for external voltage supply.

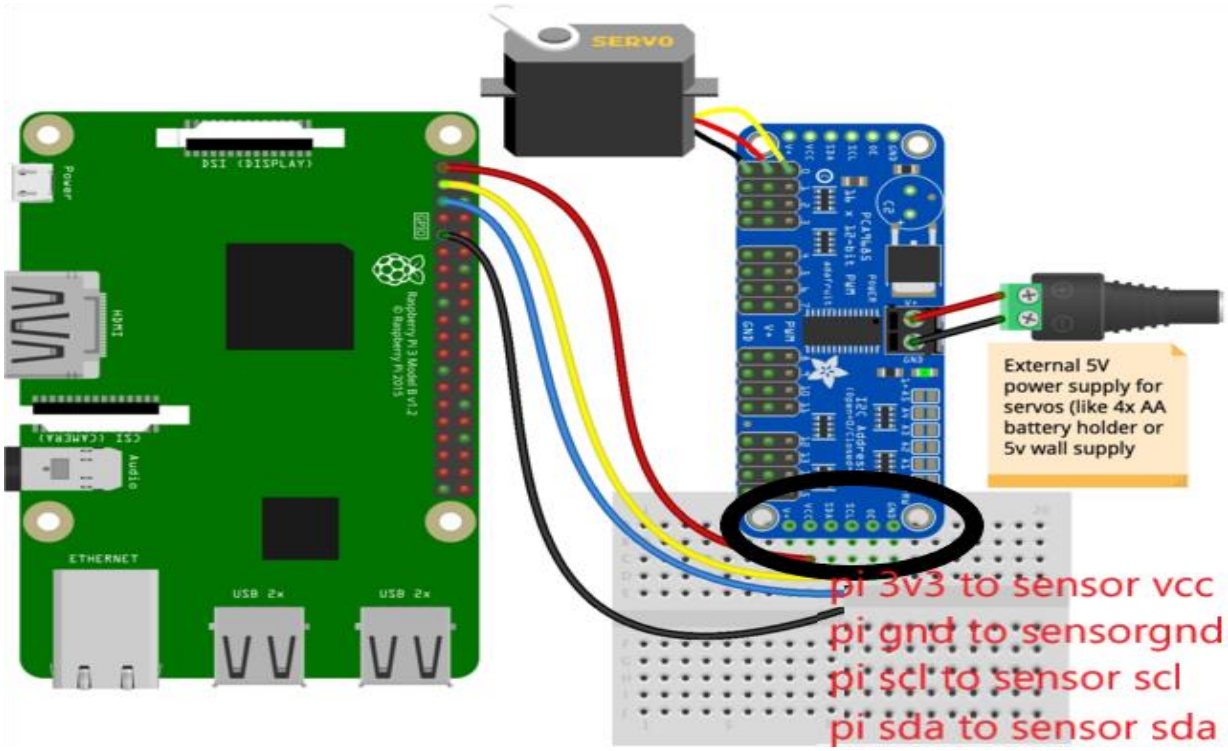

Fig 3.1: Connection of Raspberry-pi to PCA9685 with respect to breadboard

4. Implementation of the Program:

Since it is a two traffic lights system we need 6 Led's, 2 each of red, amber and green colour. We connect them in series with a 300ohm resistor and follow the connection with Raspberry-pi and PCA9685 with external power supply. We have given certain cases according to which the traffic lights should take place with a given period of time. So according to the given phases the lights on the respective traffic light will example. Let us consider that Phase 1 has red light on one traffic system and the other has green which would be on for 3 seconds and then it would switch to Phase 2 where the first traffic light has red and amber lights on while second traffic has only amber light on.

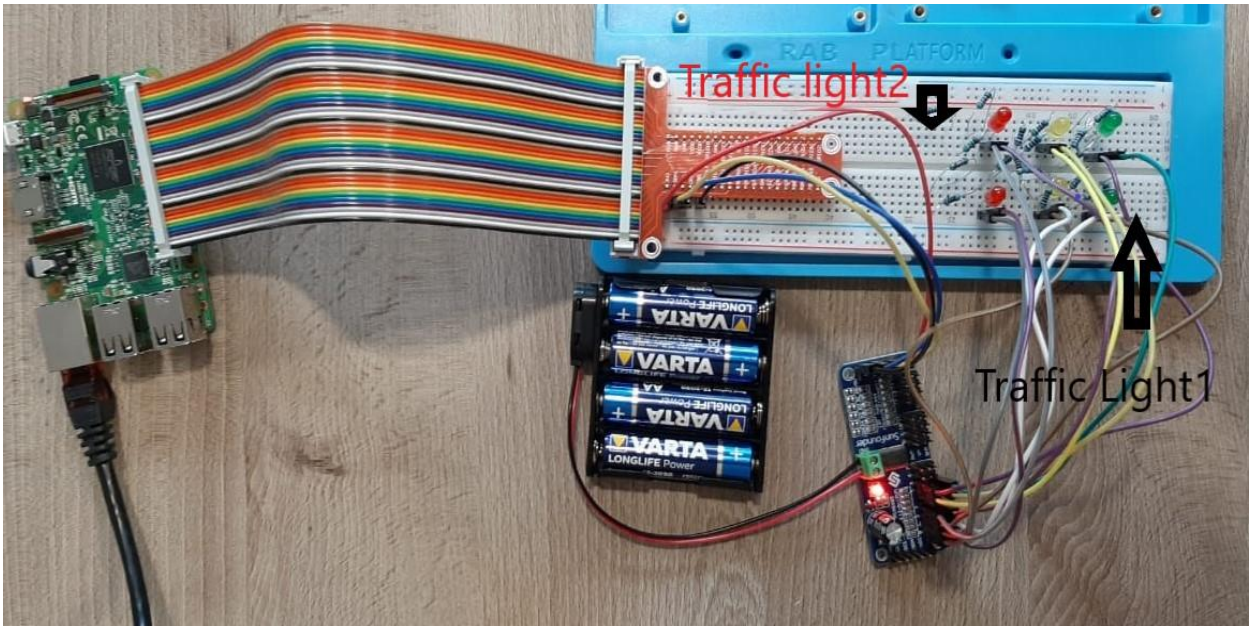

Fig 4.1: Implementation of the Traffic Lights System

5. Source Code used for implementing the Traffic Lights System:

```
#Embedded Systems Project(2020)

#LED controlled with PCA
import time
import busio
import board
import adafruit_pca9685

def SS_phase_1():
    #setting red LED on
    pca9685.channels[SS_trafficLight_0].duty_cycle = SS_onLED
    print("Red LED is ON")
    #setting green LED on
    pca9685.channels[SS_trafficLight_5].duty_cycle = SS_onLED
    print("green LED is on")
    time.sleep(3) # 3 seconds
    #setting red LED off
    pca9685.channels[SS_trafficLight_0].duty_cycle = SS_offLED
    #setting green LED off
    pca9685.channels[SS_trafficLight_5].duty_cycle = SS_offLED
    return

def SS_phase_2():
    #setting red LED and amber LED on
    pca9685.channels[SS_trafficLight_0].duty_cycle = SS_onLED
    pca9685.channels[SS_trafficLight_1].duty_cycle = SS_onLED
    print("red and amber LEDs are on")
    # setting Amber LED on
    pca9685.channels[SS_trafficLight_4].duty_cycle = SS_onLED
    print("Amber LED is on")
    time.sleep(3) # 3 seconds
    pca9685.channels[SS_trafficLight_0].duty_cycle = SS_offLED
    pca9685.channels[SS_trafficLight_1].duty_cycle = SS_offLED
    pca9685.channels[SS_trafficLight_4].duty_cycle = SS_offLED
    return

def SS_phase_3():
```

```
    #setting green LED on
    pca9685.channels[SS_trafficLight_2].duty_cycle = SS_onLED
    print("green LED is on")
    #setting red LED on
    pca9685.channels[SS_trafficLight_3].duty_cycle = SS_onLED
    print("Red LED is ON")
    time.sleep(3) # 3 seconds
    #setting green LED off
    pca9685.channels[SS_trafficLight_2].duty_cycle = SS_offLED
    #setting red LED off
    pca9685.channels[SS_trafficLight_3].duty_cycle = SS_offLED
    return

def SS_phase_4():
    pca9685.channels[SS_trafficLight_1].duty_cycle = SS_onLED
    print("Amber LED is on")
    #setting red LED and amber LED on
    pca9685.channels[SS_trafficLight_3].duty_cycle = SS_onLED
    pca9685.channels[SS_trafficLight_4].duty_cycle = SS_onLED
    time.sleep(3) # 3 seconds
    pca9685.channels[SS_trafficLight_1].duty_cycle = SS_offLED
    pca9685.channels[SS_trafficLight_3].duty_cycle = SS_offLED
    pca9685.channels[SS_trafficLight_4].duty_cycle = SS_offLED
    return

def SS_allLEDsOff():
    #setting all the LEDs off
    for trafficLightNr in range(6):
        pca9685.channels[trafficLightNr].duty_cycle = SS_offLED
    return

#Creating the I2C bus instance
i2c_bus = busio.I2C (board.SCL, board.SDA)



# Creating a PCA instance
pca9685 = adafruit_pca9685.PCA9685(i2c_bus)

#Setting the PWM frequency to 60Hz
```

```
pca9685.frequency  = 60

#per 1 of the 60 Hz there are
#65536 duty cycles-values

SS_onLED         = 65535 # PWM duty cycle 100%
SS_offLED        =     0 # PWM duty cylce   0%
#SS_ledNr        =     0 # index of LED port = 0
SS_trafficLight_0 =     0 # red light
SS_trafficLight_1 =     1 # amber light
SS_trafficLight_2 =     2 # green light
SS_trafficLight_3 =     3 # red light
SS_trafficLight_4 =     4 # amber light
SS_trafficLight_5 =     5 # green light

SS_allLEDsOff()
print("To finish press Ctrl + C")
try:
    while True:
        print()
        SS_phase_1()
        SS_phase_2()
        SS_phase_3()
        SS_phase_4()
except KeyboardInterrupt:
    print("finish")
    SS_allLEDsOff()
print("The End")
```

6. Result of all the phases implemented:

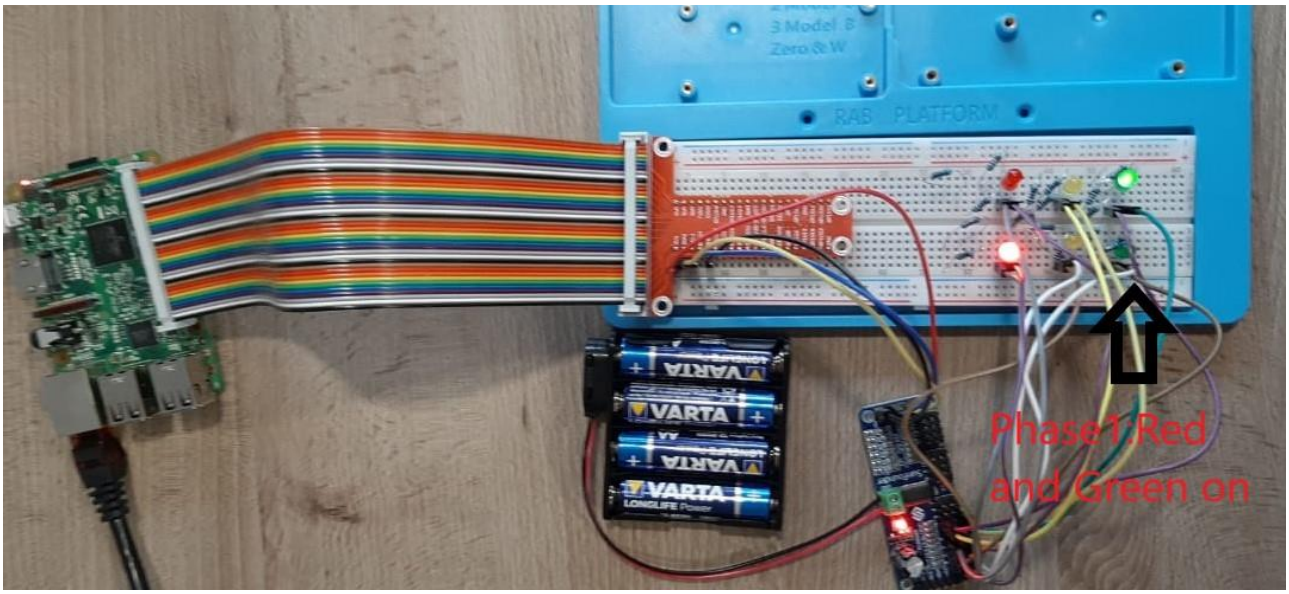

Fig 6.1: Phase 1 Implementation

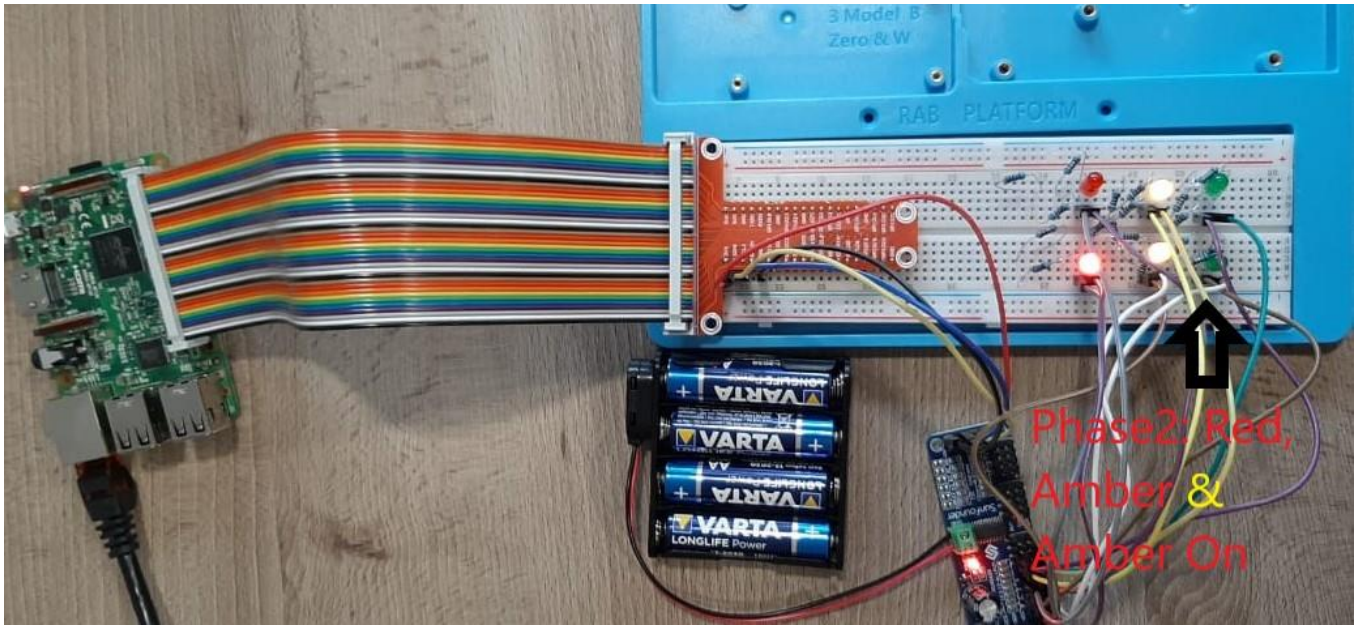

Fig 6.2: Phase 2 Implementation

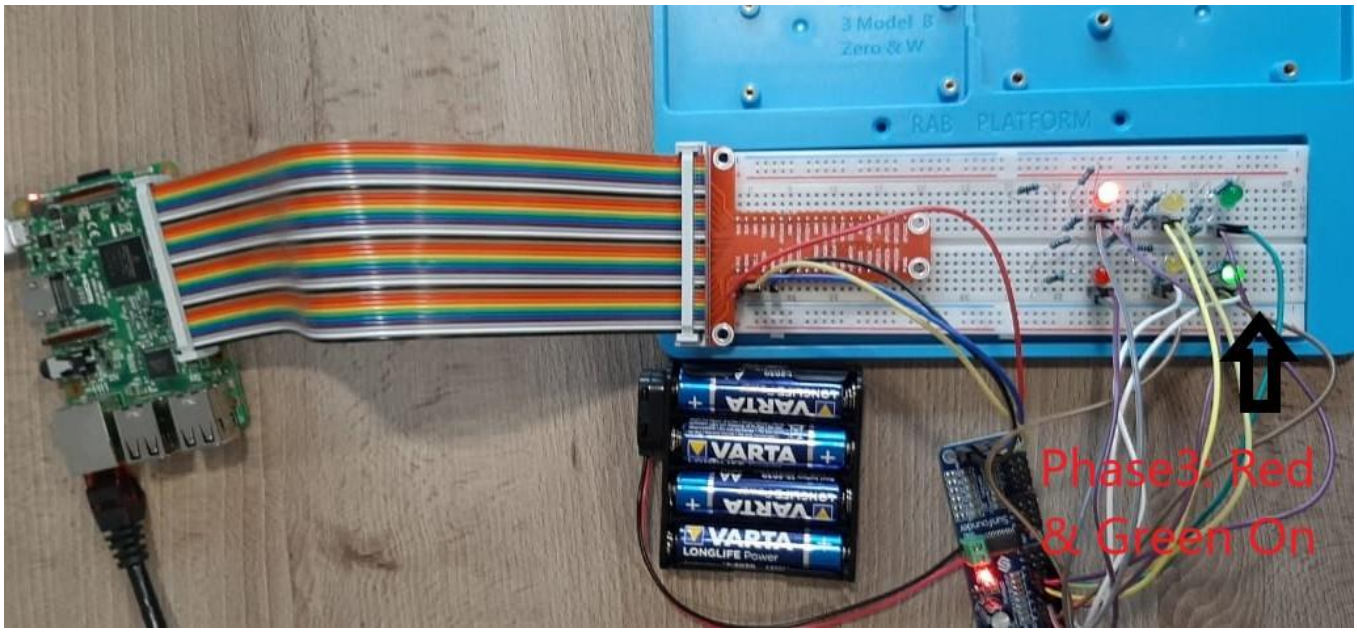

Fig 6.3: Phase 3 Implementation

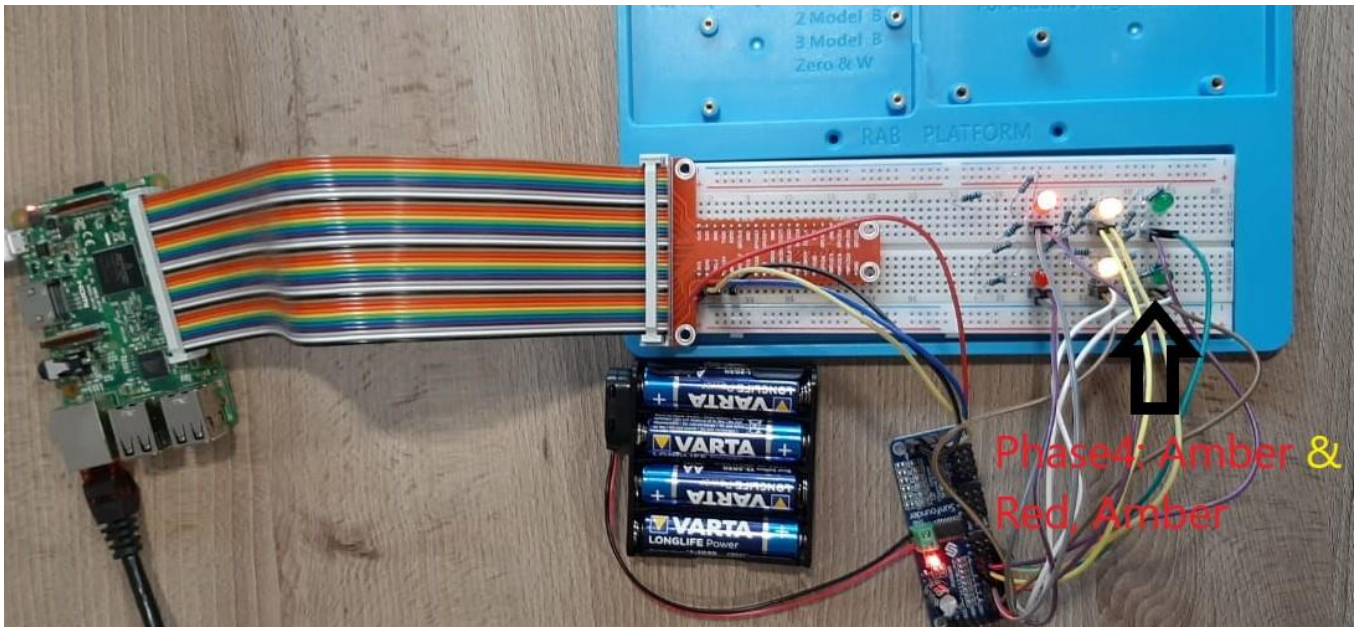

Fig 6.4: Phase 4 Implementation.

7. Conclusion:

   From the above results we can conclude that we have successfully implemented the two traffic light systems for a single lane which can be of practical purpose if proposed and implemented on a large scale.

8. References:

- https://www.nxp.com/products/power-management/lighting-driver-and-controller-ics/ic-led-controllers/16-channel-12-bit-pwm-fm-plus-ic-bus-led-controller:PCA9685

- Images from Google.
- Wikipedia