AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH
Faculty of Engineering

Lab Report

Experiment # 10

Experiment Title: Familiarization with the Raspberry Pi

Date of Perform:	14th May 2025	**Date of Submission:**	21st May 2025
Course Title:	Microprocessor and Embedded Systems Lab		
Course Code:	EEE4103	**Section:**	**R**
Semester:	Spring 2024-25	**Degree Program:**	BSc in CSE
Course Teacher:	**Prof. Dr. Engr. Muhibul Haque Bhuyan**		

Declaration and Statement of Authorship:

1. I/we hold a copy of this Assignment/Case Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case Study is my/our original work; no part has been copied from any other student's work or any other source except where due acknowledgment is made.
3. No part of this Assignment/Case Study has been written for me/us by any other person except where such collaboration has been authorized.
   by the concerned teacher and is acknowledged in the assignment.
4. I/we have not previously submitted or am submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared, and archived to detect plagiarism.
6. I/we permit a copy of my/our marked work to be retained by the Faculty Member for review by any internal/external examiners.
7. I/we understand that Plagiarism is the presentation of the work, idea, or creation of another person as though it is your own. It is a form of cheating and is a very serious academic offense that may lead to expulsion from the University. Plagiarized material can be drawn from, and presented in, written, graphic, and visual forms, including electronic data, and oral presentations. Plagiarism occurs when the origin of the source is not appropriately cited.
8. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or copy my/our work.

** Student(s) must complete all details except the faculty use part.*
** Please submit all assignments to your course teacher or the office of the concerned teacher.

Group # 05

Sl No	Name	ID	PROGRAM	SIGNATURE
1	Shahriar Hossain	22-48990-3	BSc in CSE	
2	Al Mubtasim	22-49002-3	BSc in CSE	
3	Adiba Tanzila	22-49012-3	BSc in CSE	
4	MD.Rakib Hasan	22-49029-3	BSc in CSE	
5	Md.Imdadul Hasan(Ayon)	22-49959-3	BSc in CSE	

Faculty use only

FACULTY COMMENTS	**Marks Obtained**	
	Total Marks	

Table of Contents

Marking Rubrics (to be filled by Faculty):

Level Category	Excellent [5]	Proficient [4]	Good [3]	Acceptable [2]	Unacceptable [1]	No Response [0]
Title and Objectives	Able to clarify the understanding of the lab, no issues are missing and formatting is good.	Able to clarify the understanding of the lab experiment, no issues are missing but its formatting is not good.	Able to clarify the understanding of the lab experiment, but a few issues are wrong, and its formatting is bad.	Able to clarify the understanding of the lab experiment, but it lacks a few important issues of the experiment without maintaining the format.	Unable to clarify the understanding of the lab experiment.	No Response/ copied from others/ identical submissions with gross errors/image file printed
Codes and Methods	Able to explain the experimental codes and simulation methods using Proteus very well.	Able to explain the experimental codes and simulation methods using Proteus but is not formatted well.	Able to explain the experimental codes but simulation method using Proteus is not explained well.	Presents the experimental codes but didn't explain simulation methods using Proteus clearly.	Presents the experimental codes but didn't explain simulation methods using Proteus.	
Results	Key results and images are there. Figures/Tables have all identifications and refer to them properly in the texts.	Key results and images are there. Figures/Tables have all identifications, such as the axis labels, numbers, and captions with a few minor errors; the texts refer them.	Key results and images are there. Figures/Tables lack a few identifications, such as the axis labels, numbers, and captions; the texts refer them.	Misses several key results and images. Figures/Tables lack identification, such as the axis labels, numbers, and captions; the texts don't refer them.	Major results, such as experimental and simulation results' images are not included. Figures and tables are poorly constructed or not presented.	
Discussion and Conclusion	Proper interpretation of results and summarizes the results to draw a conclusion, discusses its applications in real-life situations to connect with the report's conclusion.	Proper interpretation of results and summarizes the results to draw a conclusion but didn't discuss its applications in real-life situations to connect with the conclusion of the report.	Interpretation of results is presented. However, there is a disconnect between the results and discussion.	Misses the interpretation of key results. There is little connection between the results and discussion.	Very poor interpretation of the results. No connection between results and discussions.	
Question and Answer	Able to produce all questions' answers correctly maintaining the lab report format.	Able to produce all questions' answers but didn't maintain the lab report format.	Able to produce all questions' answers but wrong answers to a few questions.	Able to produce all questions' answers but wrong/missing answers to multiple questions.	Unable to produce all questions' answers and completely wrong answers.	
Comments						**Total Marks (25)**

Objectives:

The objectives of this experiment are to-

- Familiarize the students with the Raspberry Pi.
- Make an LED blink using the Raspberry Pi and its time.sleep() function.
- Control the LEDs' ON/OFF using the input push switch.
- Implement a traffic light control system

Equipment List:

[1] Activated Raspberry pi
[2] LED
[3] Push switch
[4] Resistor (220 □)
[5] Breadboard
[6] Jumper wires

Circuit Diagram:

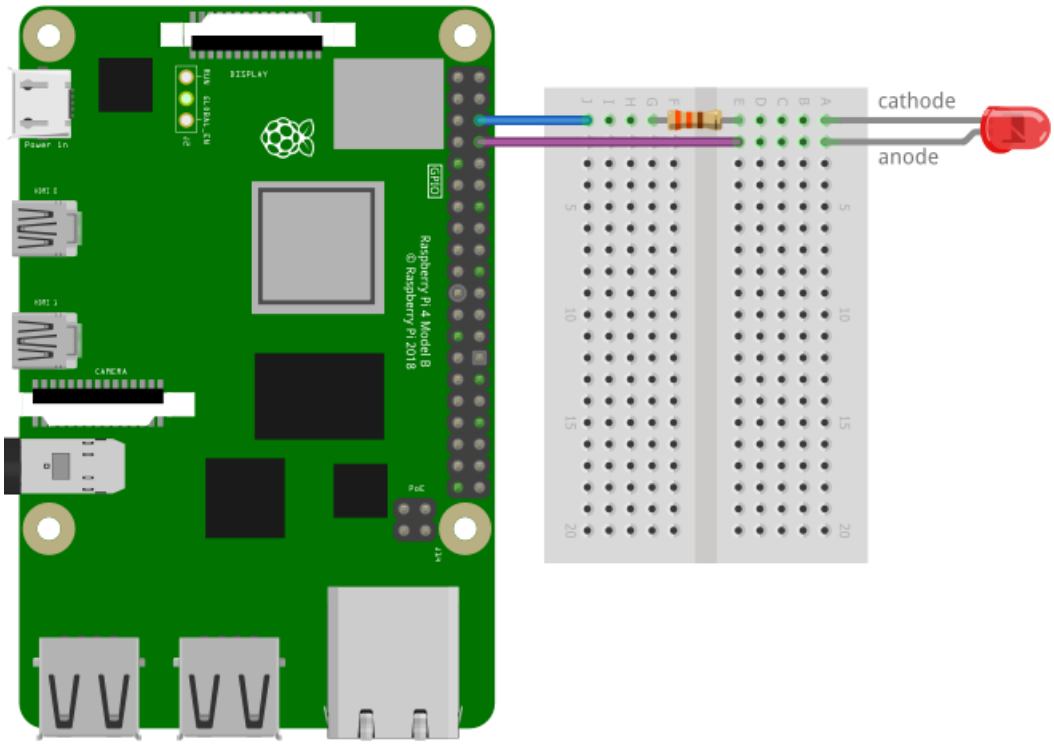

Fig : Setting up the circuit for the LED blinking program.

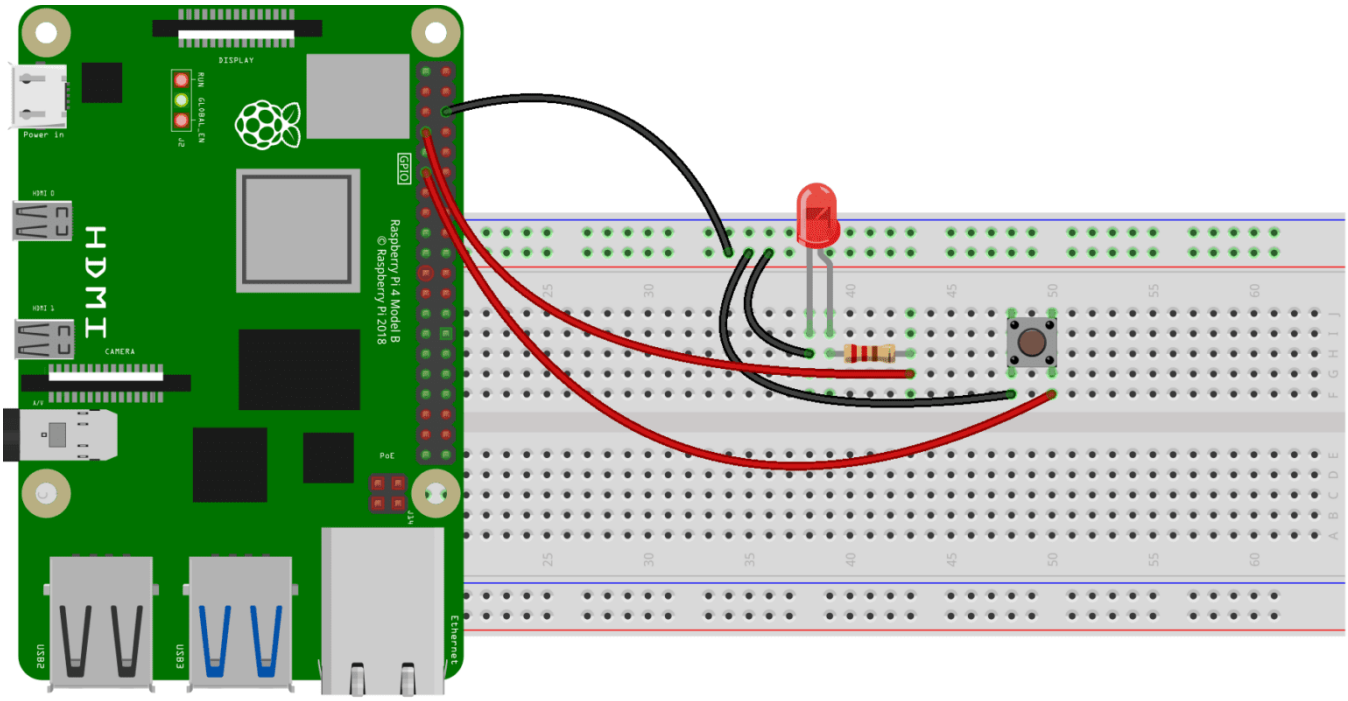

Figure : Setting up the circuit for the LED controlling experiment using a button switch.

Experimental Output Results:

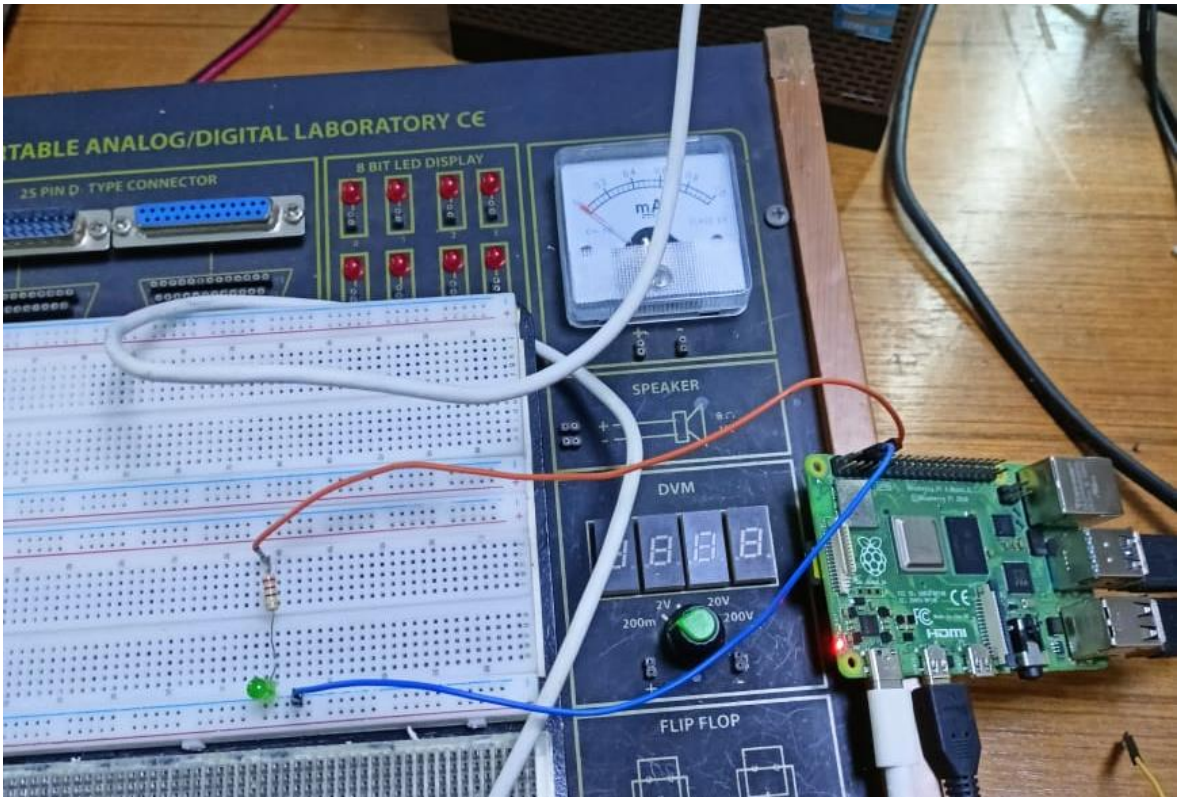

Figure: Hardware implementation of LED blinking without switch(Off state)

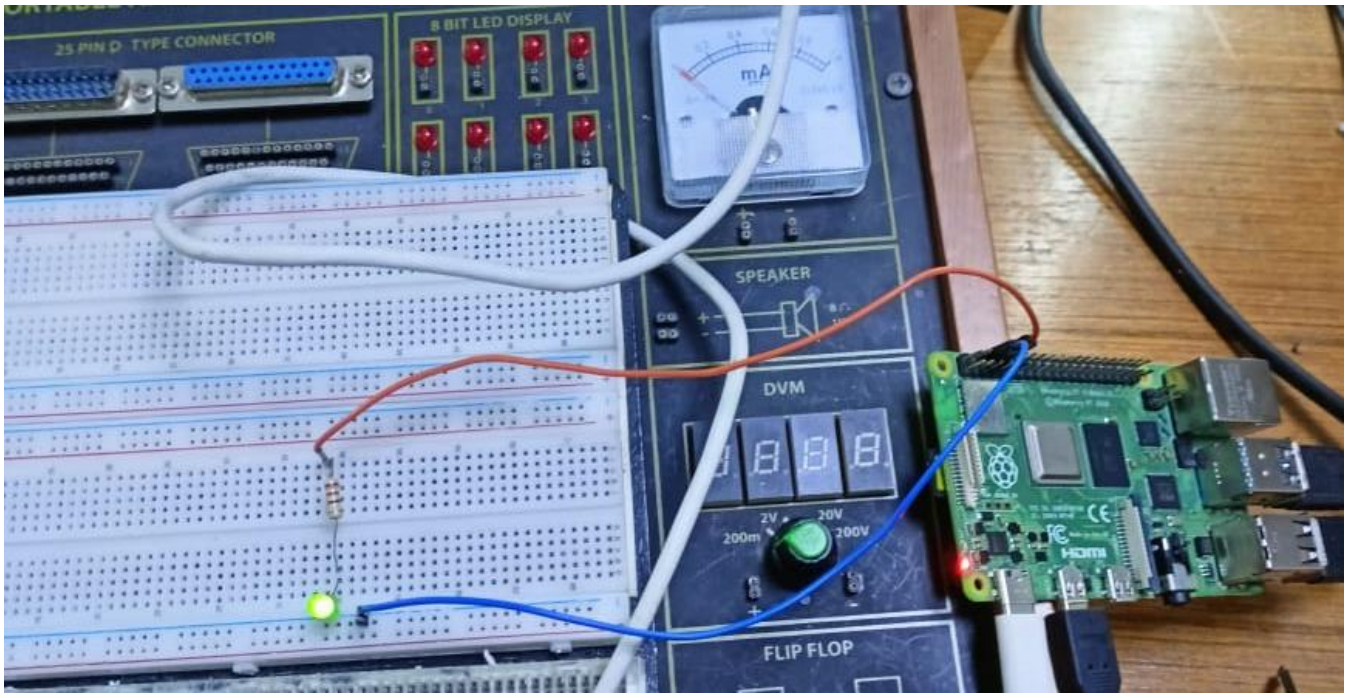

Figure: Hardware implementation of LED blinking without switch(On state)

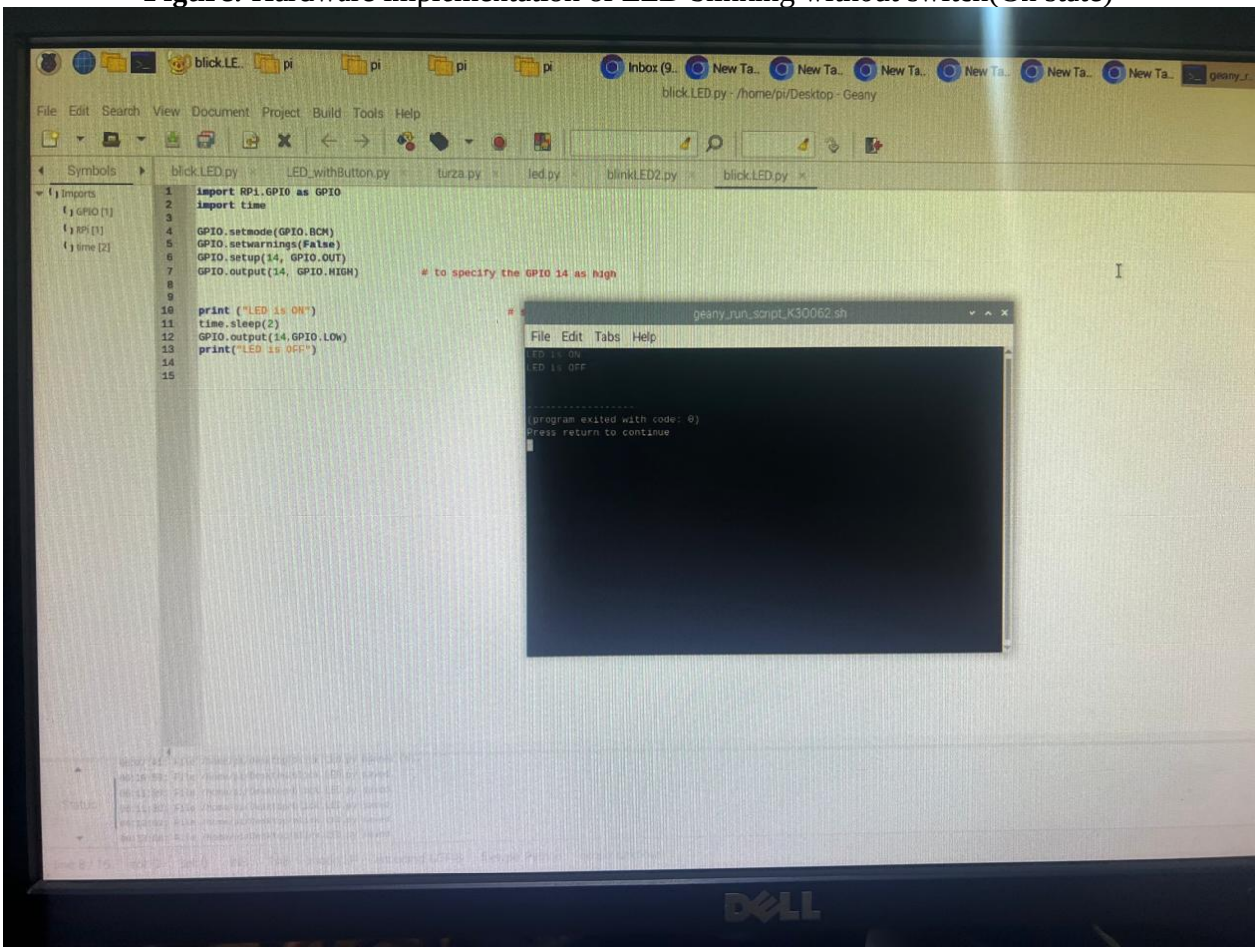

Figure: Raspberry Pi output of LED Blinking without switch

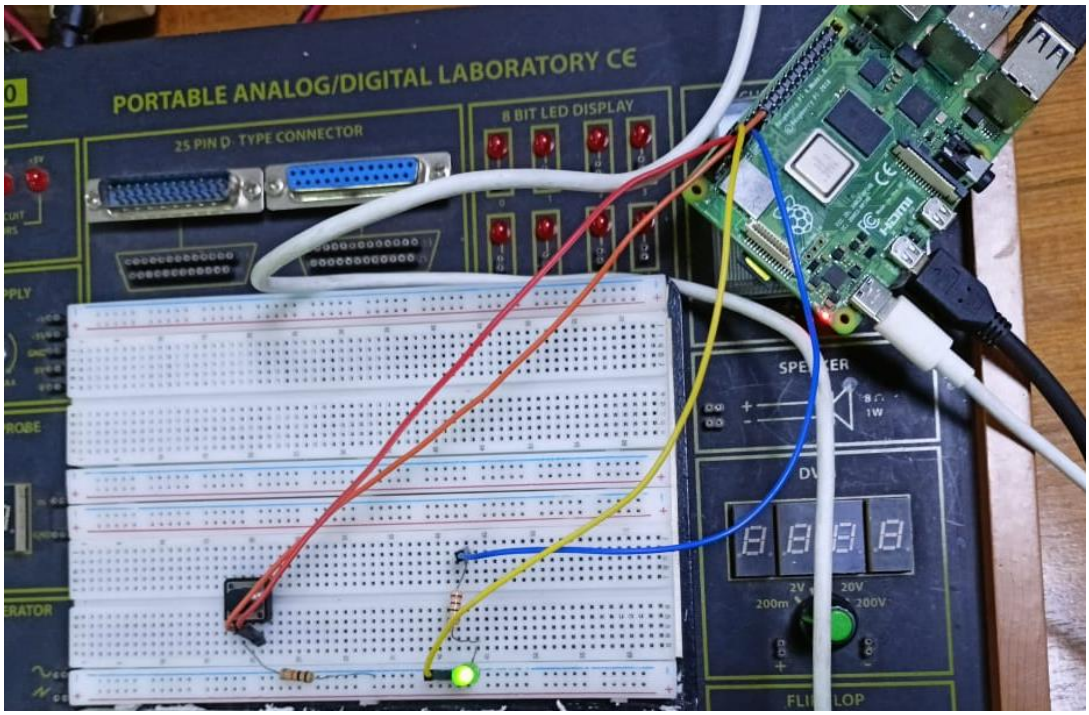

Figure: Hardware implementation of LED without switch press(On state)

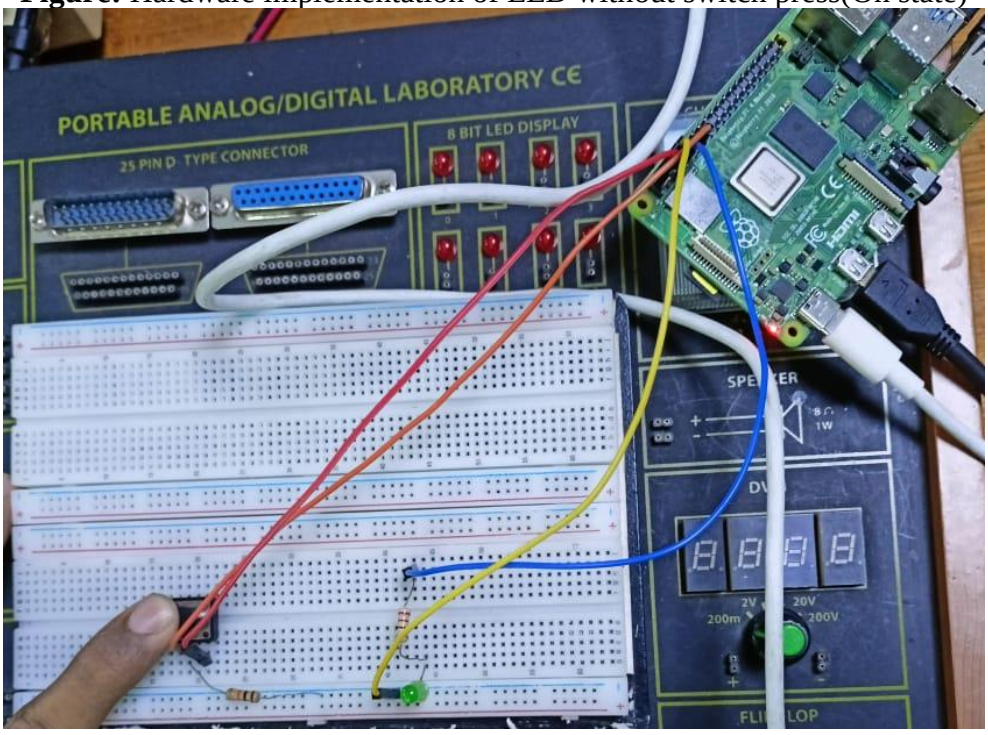

Figure: Hardware implementation of LED with switch press(Off state)

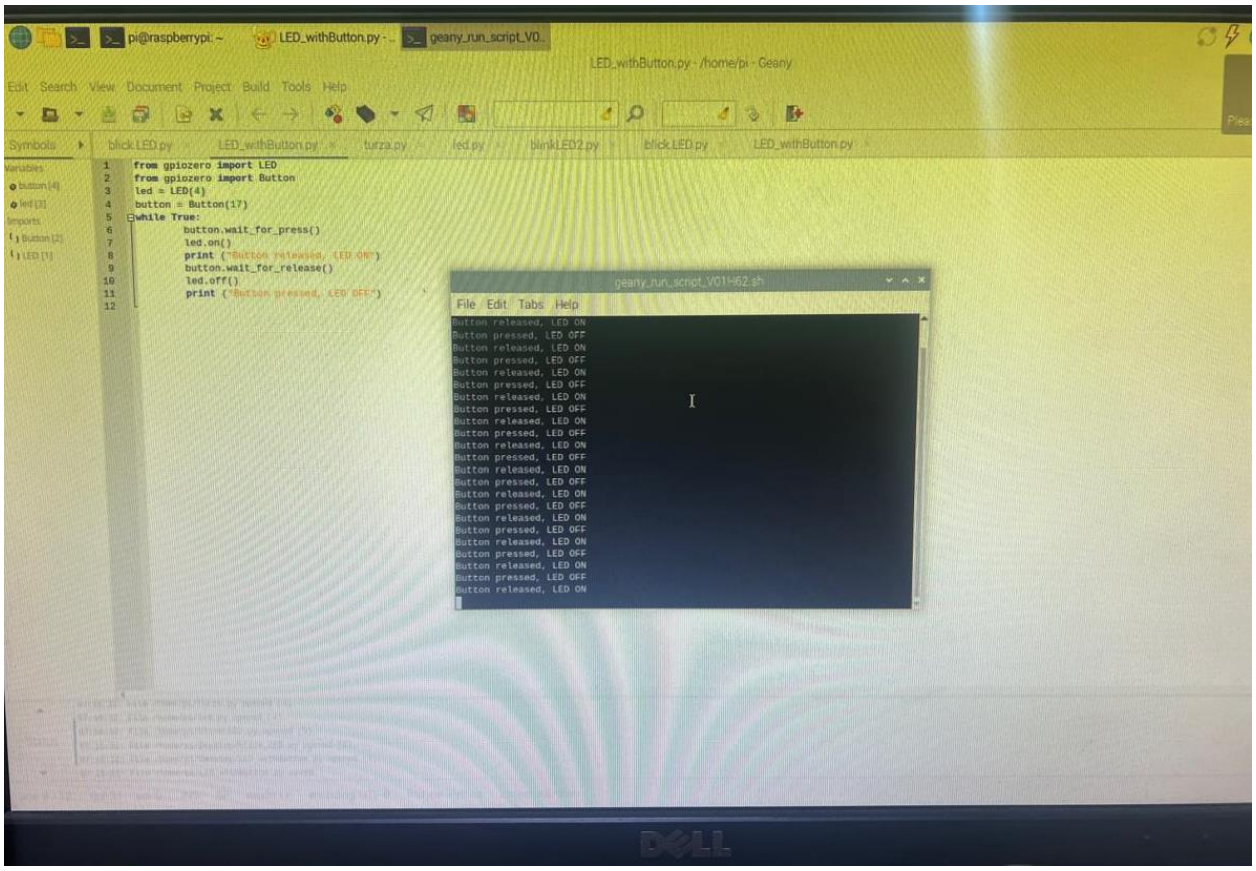

Figure: Raspberry Pi output of LED Blinking with switch

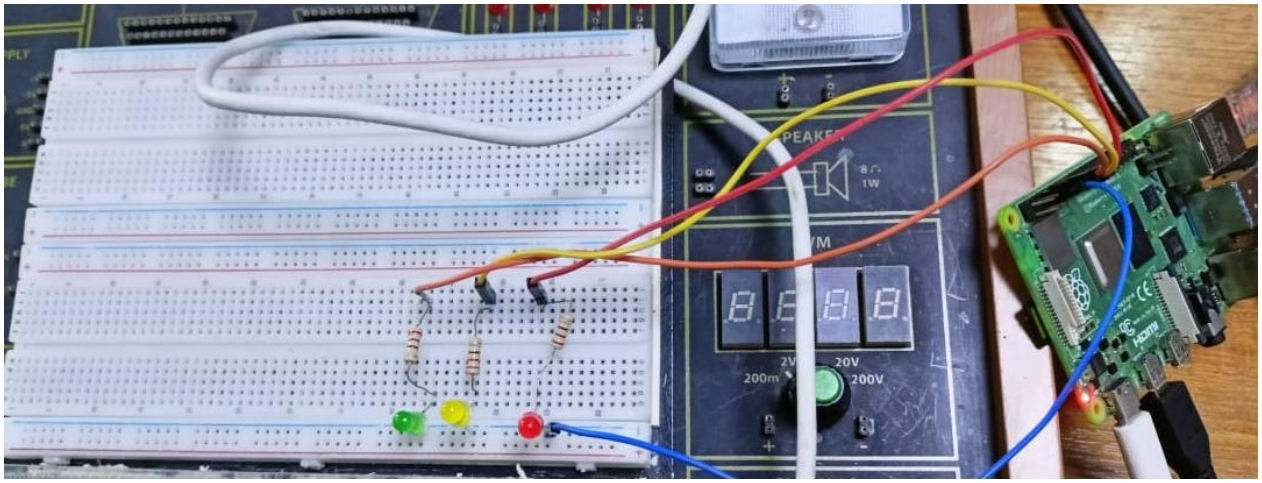

Figure: Simple Traffic Control System using Raspberry Pi(All Off)

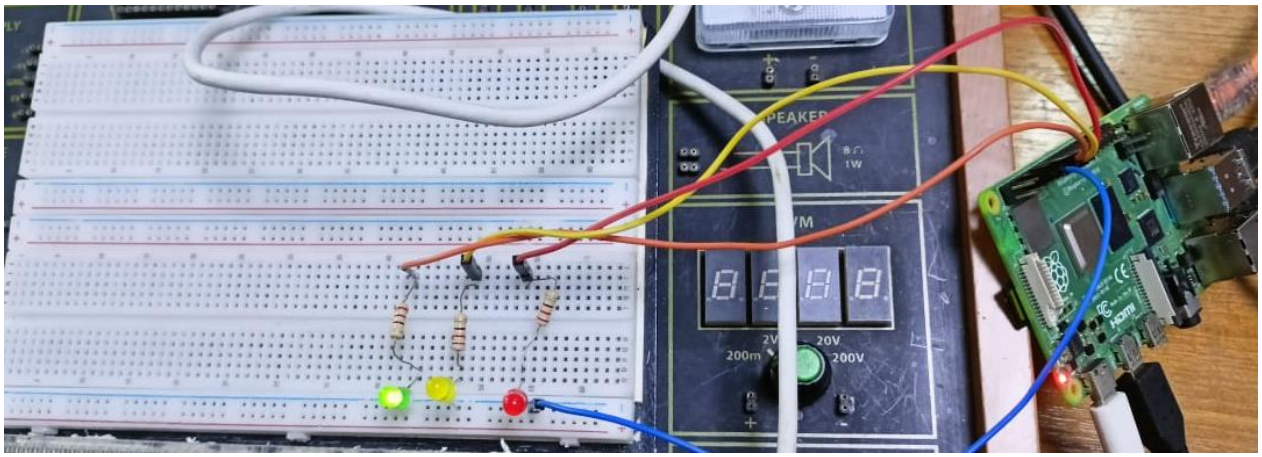

Figure: Simple Traffic Control System using Raspberry Pi(Green On)

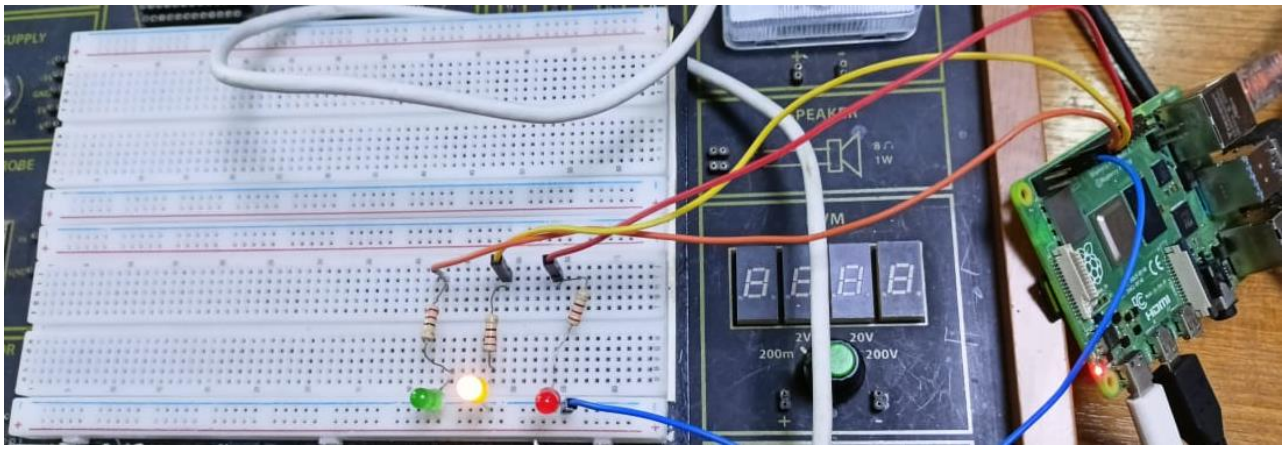

Figure: Simple Traffic Control System using Raspberry Pi(Yellow On)

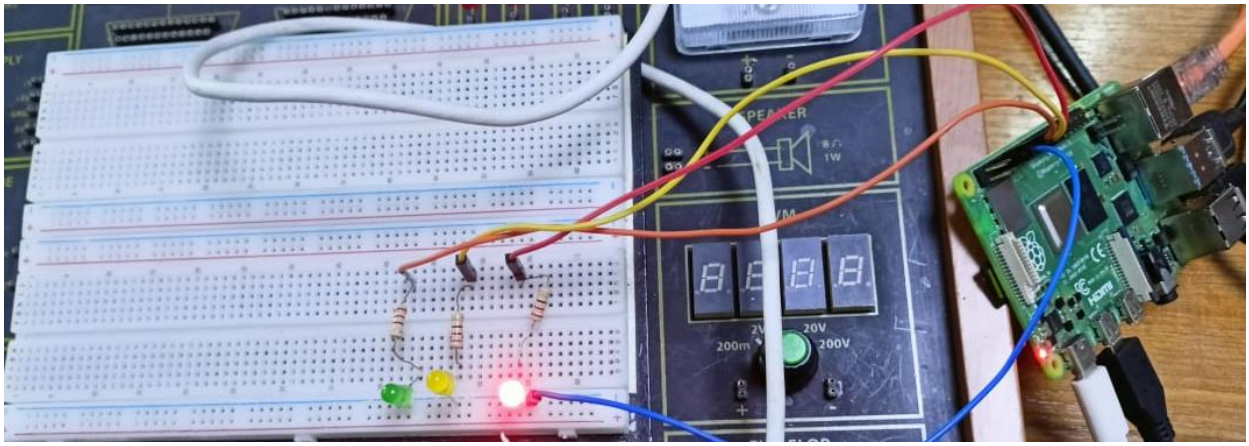

Figure: Simple Traffic Control System using Raspberry Pi(Red On)

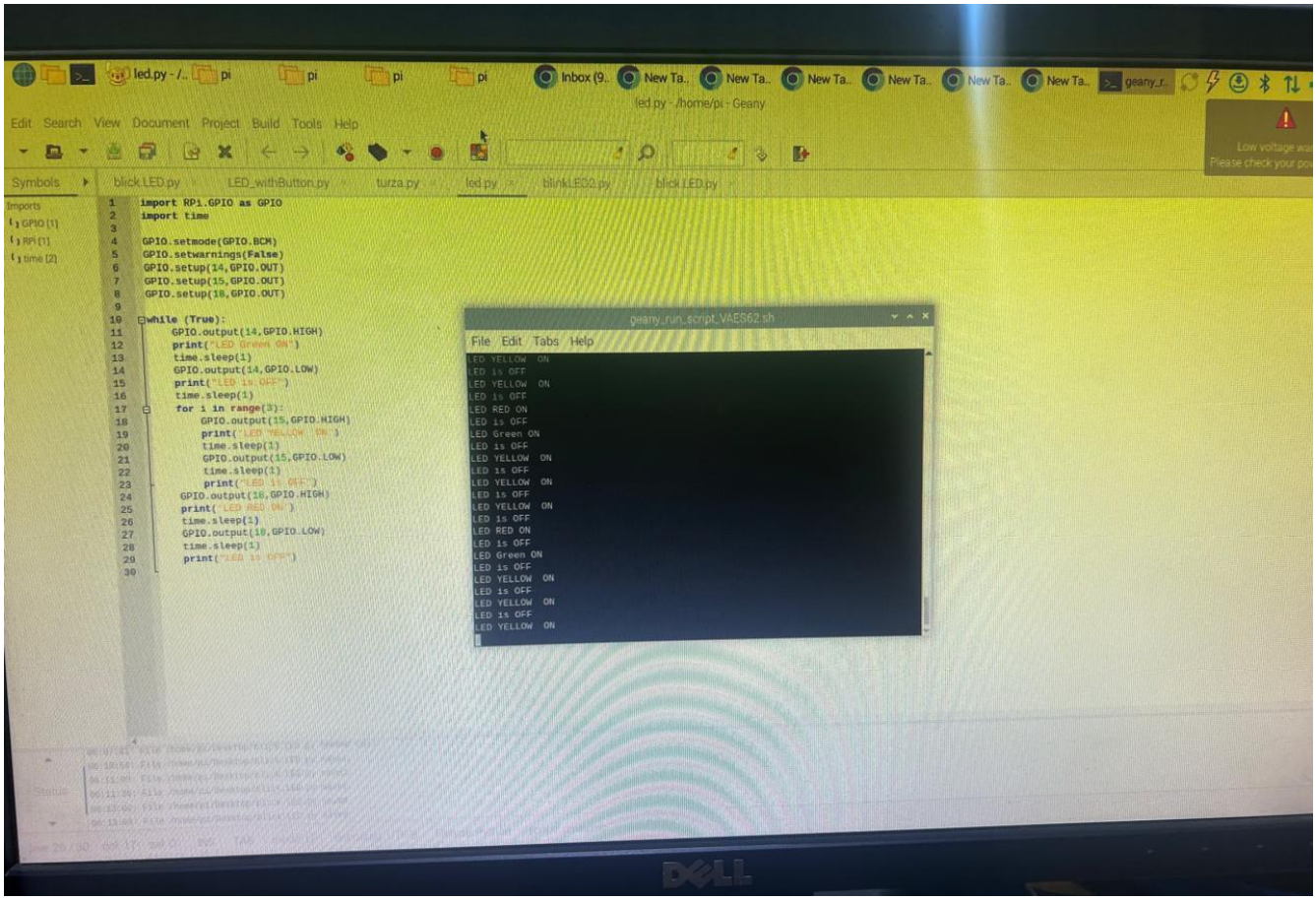

Figure: Raspberry Pi output of Simple Traffic Control System

Simulation Output Results:

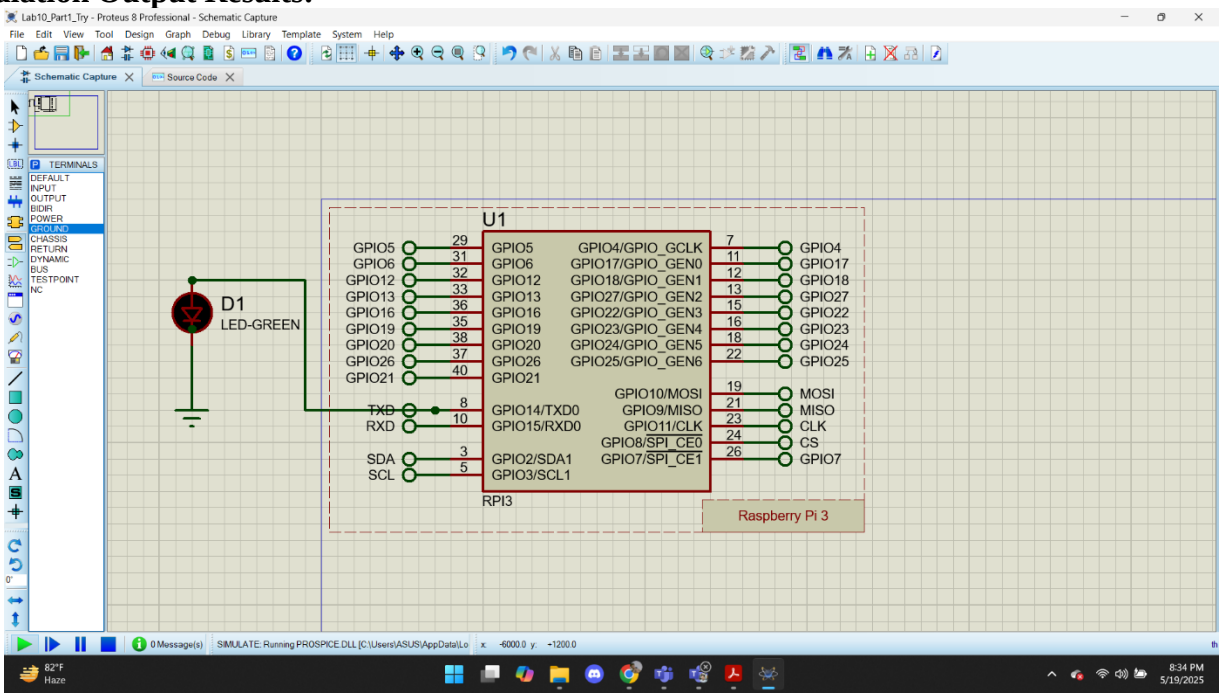

Figure: Proteus Simulation of Raspberry Pi LED Blink without switch(Off State)

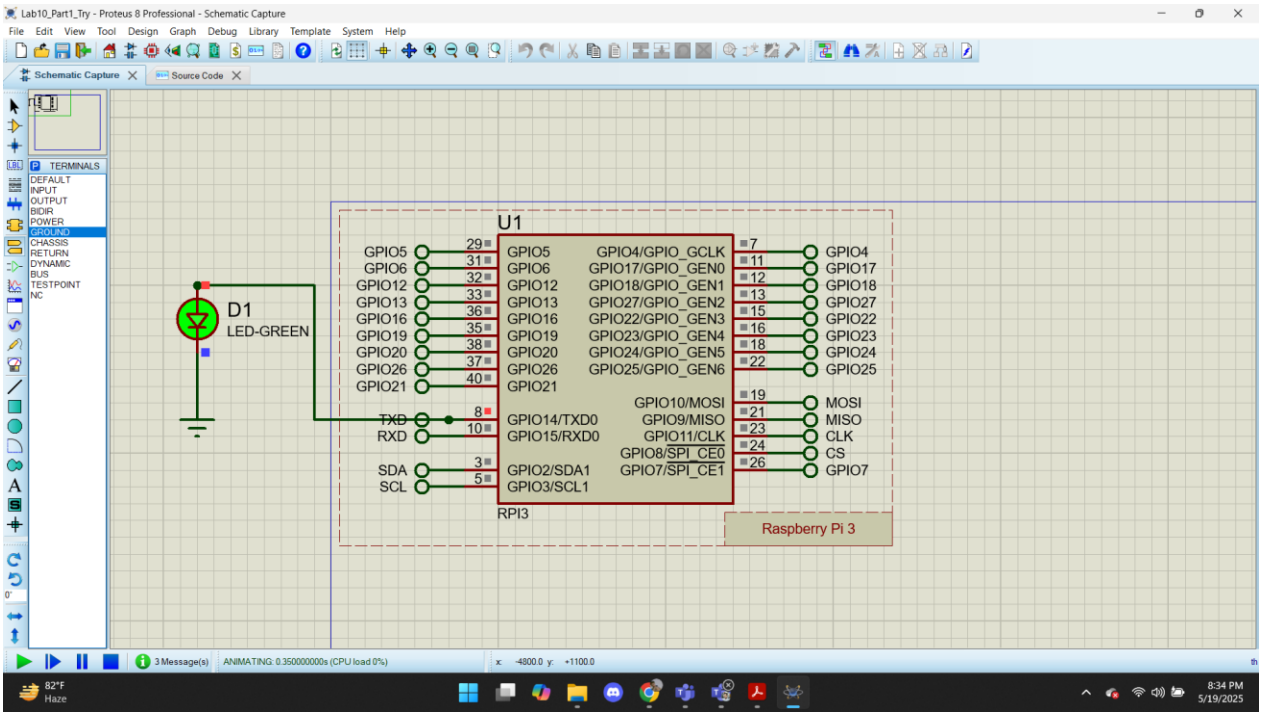

Figure: Proteus Simulation of Raspberry Pi LED Blink without switch(On State)

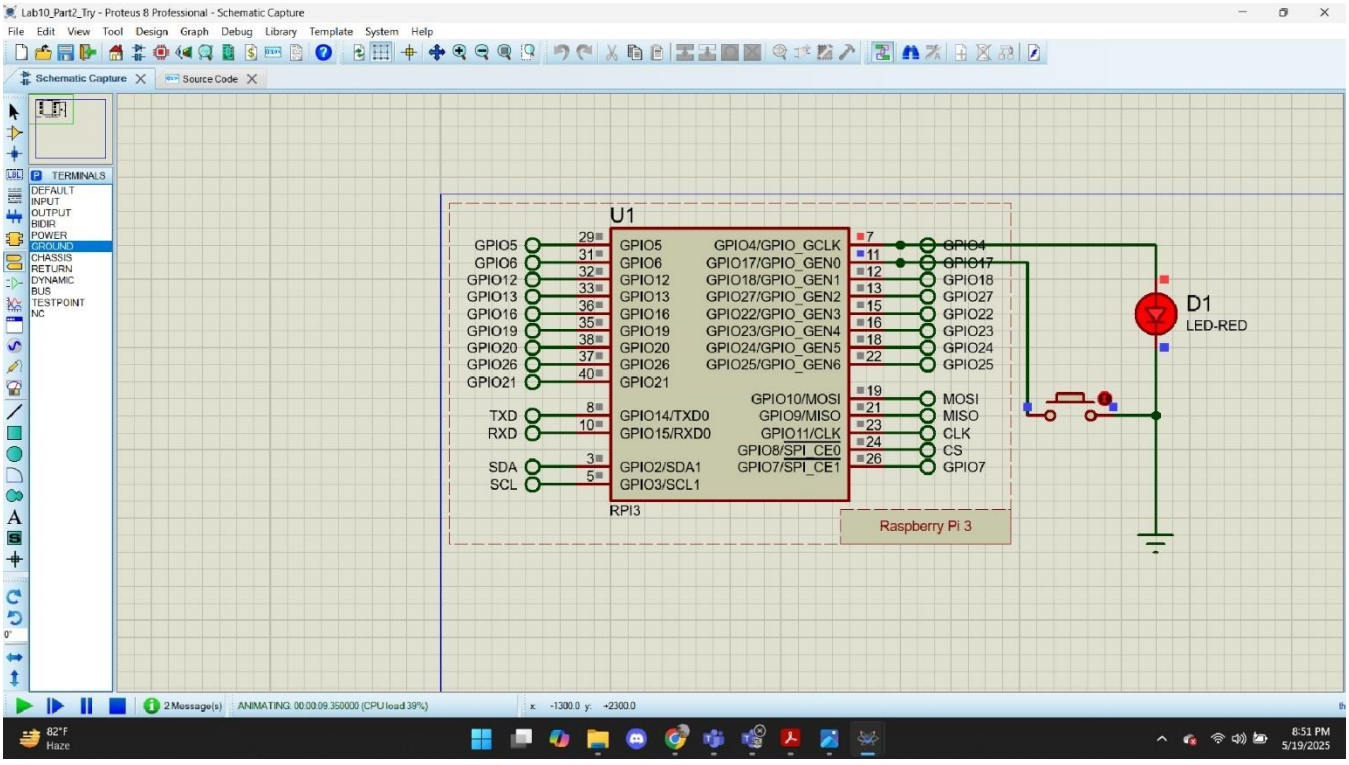

Figure: Proteus Simulation of Raspberry Pi LED Blink with switch(On State)

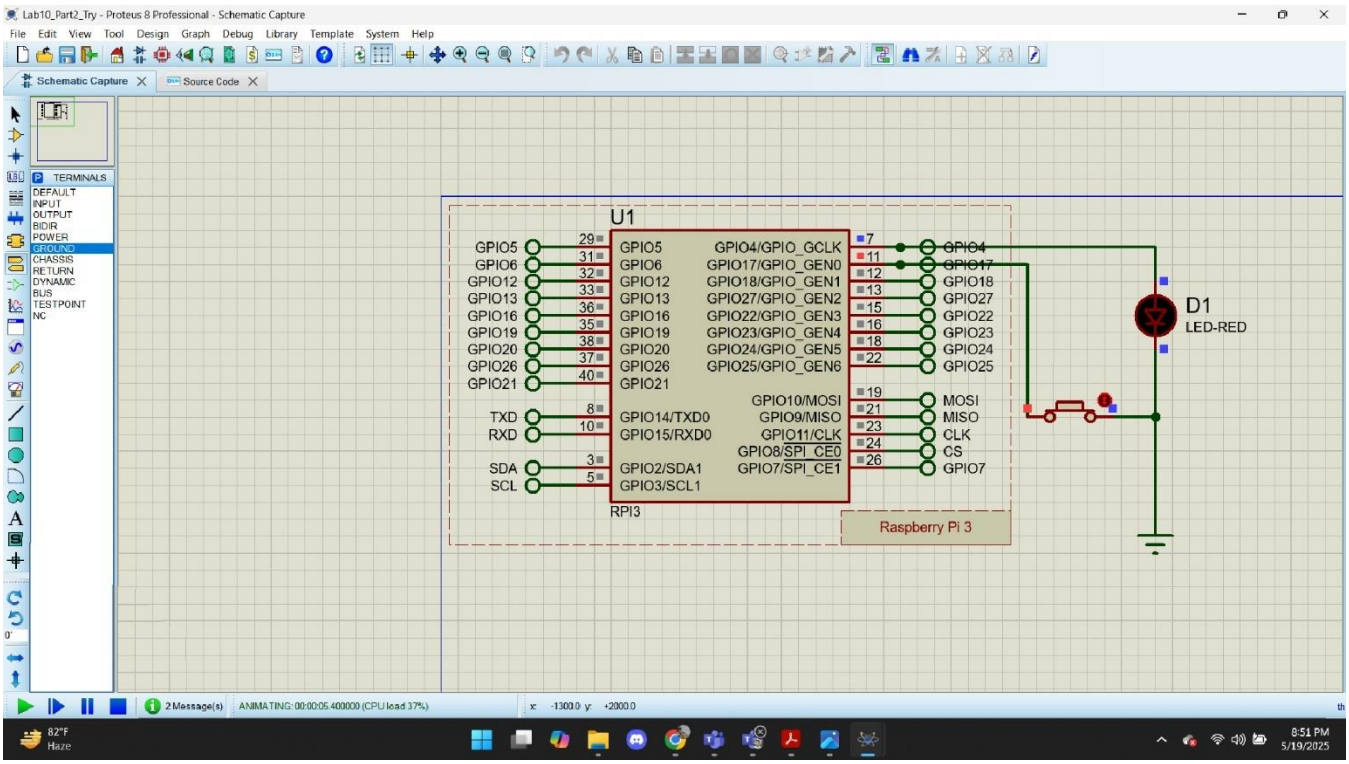

Figure: Proteus Simulation of Raspberry Pi LED Blink with switch press(Off State)

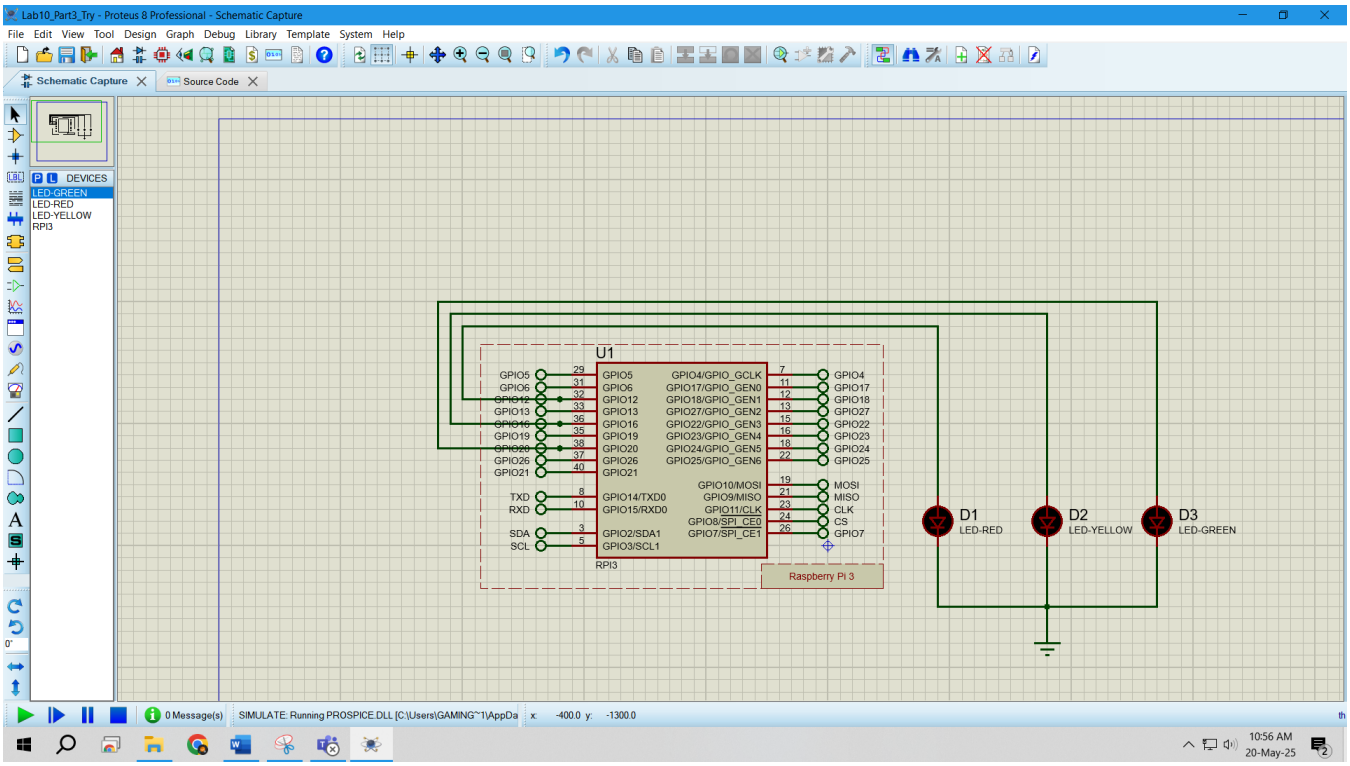

Figure: Proteus Simulation of Raspberry Pi LED Traffic LED System(Off State)

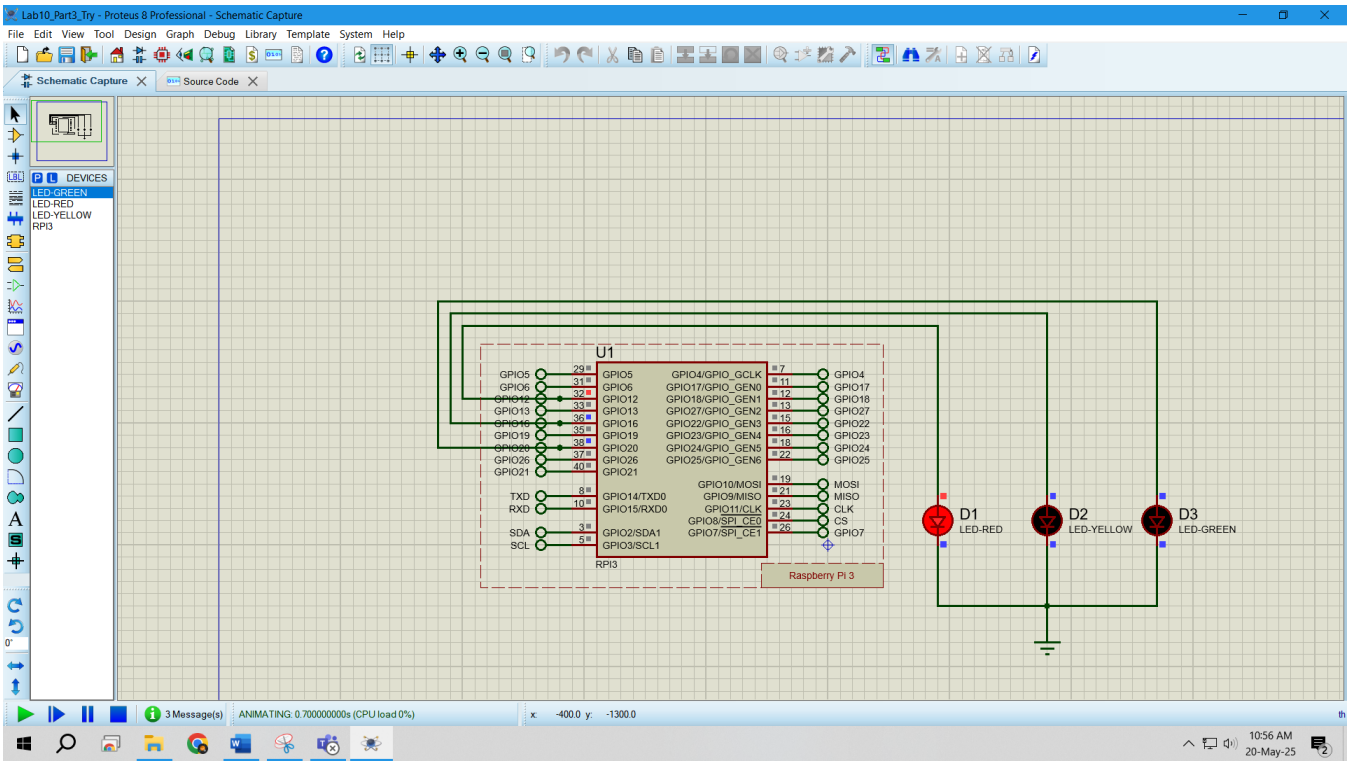

Figure: Proteus Simulation of Raspberry Pi LED Traffic LED System(Red On)

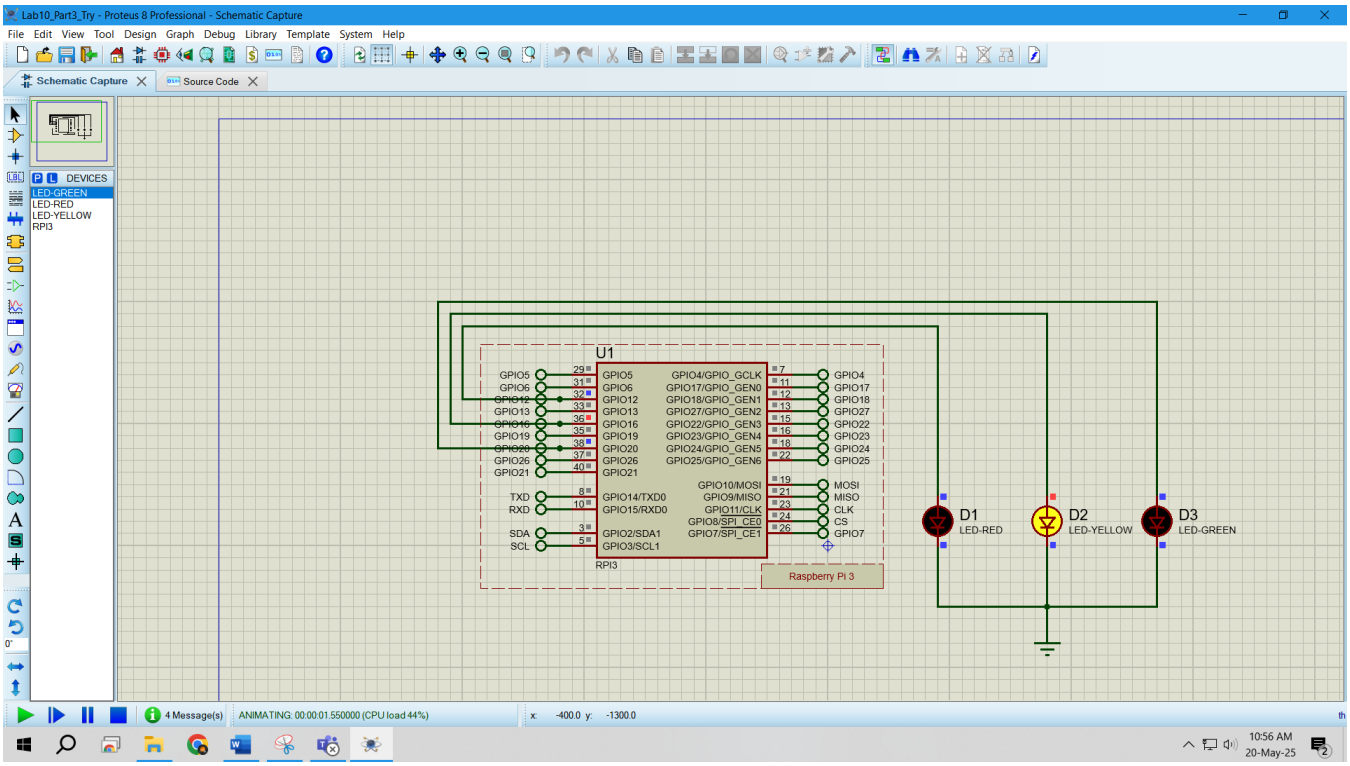

Figure: Proteus Simulation of Raspberry Pi LED Traffic LED System(Yellow On)

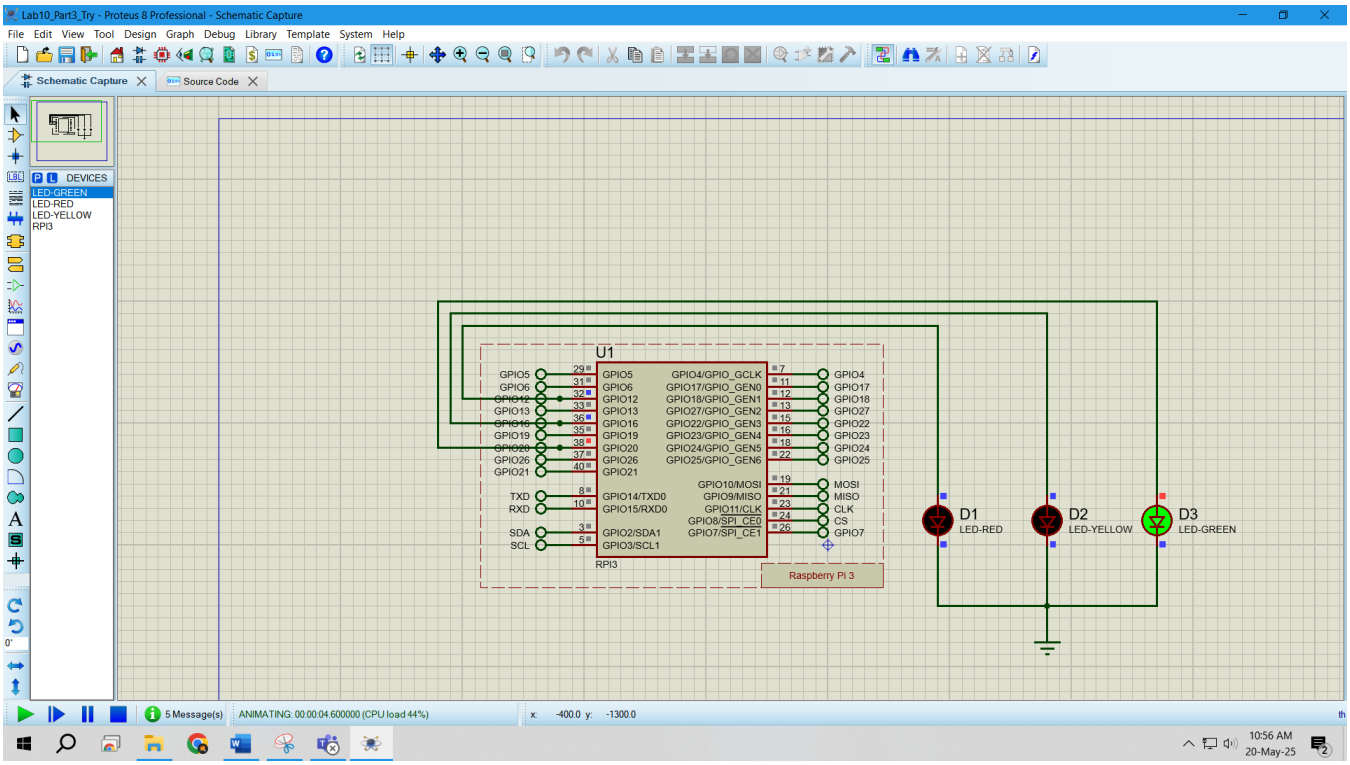

Figure: Proteus Simulation of Raspberry Pi LED Traffic LED System(Green On)

Explaination:

- Proteus simulation software was used for circuit simulation.
- To create a Raspberry Pi project, first open a new project and select "Create Firmware Project"
- Then, select Raspberry under the Family section and proceed to create the project.
- Once created, the Raspberry Pi project will function normally within Proteus.
- A "Source Code" option will appear at the top inside Proteus, where the code can be written directly to run the project.
- Then the code was used and the simulations were run.

Answers to the Questions in the Lab Manual:

1. LED Blink without Switch

```
$ nano blinkLED.py
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(14, GPIO.OUT)
GPIO.output(14, GPIO.HIGH)
print "LED is ON"
time.sleep(2)
```

```
GPIO.output(14, GPIO.LOW)
print "LED is OFF"
```

Explaination:

$ nano blinkLED.py

o Opens the nano text editor and creates a Python file named blinkLED.py.

• import RPi.GPIO as GPIO

o Imports the RPi.GPIO library and assigns it the alias GPIO to interact with the Raspberry Pi's GPIO pins.

• import time

o Imports the time library which provides time-related functions like sleep().

• GPIO.setmode(GPIO.BCM)

o Sets the pin numbering mode to **BCM**, referring to the GPIO pin numbers rather than the physical pin locations.

• GPIO.setwarnings(False)

o Disables warning messages that might occur if a pin is already in use or configured multiple times.

• GPIO.setup(14, GPIO.OUT)

o Configures GPIO pin 14 as an output pin, which will be used to control the LED.

• GPIO.output(14, GPIO.HIGH)

o Sets GPIO pin 14 to **HIGH** (3.3V), which turns the LED **ON**.

• print "LED is ON"

o Prints a message to the terminal indicating that the LED is currently turned ON.

• time.sleep(2)

o Pauses the program for **2 seconds** to keep the LED ON for that duration.

• GPIO.output(14, GPIO.LOW)

o Sets GPIO pin 14 to **LOW** (0V), which turns the LED **OFF**.

• print "LED is OFF"

o Prints a message to the terminal indicating that the LED is currently turned OFF.

2. LED Blink with switch

```
from gpiozero import LED
from gpiozero import Button
led = LED(4)
button = Button(17)
while True:
```

```
button.wait_for_press()
led.on()
button.wait_for_release()
led.off()
```

from gpiozero import LED
o Imports the LED class from the gpiozero library, allowing control of an LED connected to a GPIO pin.

• from gpiozero import Button
o Imports the Button class from the gpiozero library to detect button press/release events.

• led = LED(4)
o Initializes an LED object on **GPIO pin 4** and assigns it to the variable led for output control.

• button = Button(17)
o Initializes a Button object on **GPIO pin 17** and assigns it to the variable button to detect input events.

• while True:
o Starts an infinite loop that keeps the program running continuously to monitor button interactions.

• button.wait_for_press()
o Pauses the loop until the button is **pressed**, using the wait_for_press() method.

• led.on()
o Turns **ON** the LED connected to GPIO 4 by calling the on() method of the led object.

• button.wait_for_release()
o Waits until the button is **released**, using the wait_for_release() method.

• led.off()
o Turns **OFF** the LED by calling the off() method of the led object.

3. Traffic Control System

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(14, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
GPIO.setup(18, GPIO.OUT)
while (True):
   GPIO.output(14, GPIO.HIGH)
   print("LED Green ON")
   time.sleep(1)
   GPIO.output(14, GPIO.LOW)
   print("LED Green OFF")
   time.sleep(1)
   for i in range(3):
      GPIO.output(15, GPIO.HIGH)
      print("LED Yellow ON")
      time.sleep(1)
```

```
        GPIO.output(15, GPIO.LOW)
        print("LED Yellow OFF")
        time.sleep(1)
    GPIO.output(18, GPIO.HIGH)
    print("LED Red ON")
    time.sleep(1)
    GPIO.output(18, GPIO.LOW)
    time.sleep(1)
    print("LED Red OFF")
```

Explaination:

• import RPi.GPIO as GPIO
o Imports the RPi.GPIO library and gives it the alias GPIO to access and control the Raspberry Pi GPIO pins.
• import time
o Imports the time module to enable use of the sleep() function for delays.
• GPIO.setmode(GPIO.BCM)
o Sets the GPIO pin numbering mode to **BCM**, which uses the Broadcom SOC channel numbers.
• GPIO.setwarnings(False)
o Disables warning messages that may occur if GPIO pins are reused or configured multiple times.
• GPIO.setup(14, GPIO.OUT)
o Configures **GPIO pin 14** as an output pin to control the **green LED**.
• GPIO.setup(15, GPIO.OUT)
o Configures **GPIO pin 15** as an output pin to control the **yellow LED**.
• GPIO.setup(18, GPIO.OUT)
o Configures **GPIO pin 18** as an output pin to control the **red LED**.
• while (True):
o Starts an infinite loop to repeatedly execute the LED control sequence.
• GPIO.output(14, GPIO.HIGH)
o Turns **ON** the green LED by setting GPIO 14 to **HIGH**.
• print("LED Green ON")
o Displays the message in the terminal to indicate that the green LED is on.
• time.sleep(1)
o Waits for **1 second** while the green LED remains on.
• GPIO.output(14, GPIO.LOW)
o Turns **OFF** the green LED by setting GPIO 14 to **LOW**.
• print("LED Green OFF")
o Displays the message in the terminal to indicate that the green LED is off.
• time.sleep(1)
o Waits for **1 second** before moving to the next LED sequence.
• for i in range(3):
o Repeats the following block **3 times** to blink the yellow LED.
• GPIO.output(15, GPIO.HIGH)
o Turns **ON** the yellow LED by setting GPIO 15 to **HIGH**.
• print("LED Yellow ON")
o Prints a message indicating the yellow LED is on.
• time.sleep(1)
o Keeps the yellow LED on for **1 second**.
• GPIO.output(15, GPIO.LOW)
o Turns **OFF** the yellow LED by setting GPIO 15 to **LOW**.
• print("LED Yellow OFF")
o Prints a message indicating the yellow LED is off.

• time.sleep(1)
o Waits for **1 second** before the next blink.
• GPIO.output(18, GPIO.HIGH)
o Turns **ON** the red LED by setting GPIO 18 to **HIGH**.
• print("LED Red ON")
o Prints a message indicating the red LED is on.
• time.sleep(1)
o Keeps the red LED on for **1 second**.
• GPIO.output(18, GPIO.LOW)
o Turns **OFF** the red LED by setting GPIO 18 to **LOW**.
• time.sleep(1)
o Waits for **1 second** before restarting the loop.
• print("LED Red OFF")
o Prints a message indicating the red LED is off.

Discussions: In this lab experiment, we learned how to use the Raspberry Pi by simulating three simple tasks: blinking a light with and without switch and controlling traffic lights. For the blinking light, a while loop was used to make the LED blink continuously,along with a switch which turned off the led while pressed and kept led on while not pressed. And in the traffic light system, a while loop ensured the system repeated, and a for loop blinked the yellow light three times. Both the hardware and software simulations were executed without any issues.

Reference(s):

[1] Arduino IDE, https://www.arduino.cc/en/Main/Software accessed on 2nd July 2023.
[2] https://www.tinkercad.com/things/b6oU31mFyQa-brilliant-snaget/editel?tenant=circuits, accessed on 2nd July 2023.