AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Faculty of Engineering

Lab Report

Experiment # 02

Experiment Title: Familiarization with an STM32 microcontroller board, studying an LED blink test, and implementing a 2-push button LED activation circuit using an STM32 microcontroller board.

Date of Perform:	10 March 2025	**Date of Submission:**	17 March 2025
Course Title:	Microprocessor and Embedded Systems Lab		
Course Code:	EE4103	**Section:**	P
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 # 01

Sl No	Name	ID	PROGRAM	SIGNATURE
1.	Md.Saikat Hossain	23-51242-1	BSc in CSE	
2.	Md.Mosharof Hossain Khan	23-51259-1	BSc in CSE	
3.	Rimal Banik	23-51260-1	BSc in CSE	
4.	Md.Rahidul Islam	23-51269-1	BSc in CSE	
5.	Rahat Ahmed	21-44911-2	BSc in CSE	

Faculty use only

FACULTY COMMENTS	**Marks Obtained**	
	Total Marks	

Contents

Objectives:

The objectives of this experiment are to

1. Study the STM32 Microcontroller Board.
2. Learn basic programming commands of the STM32 Microcontroller Board.
3. Apply the coding techniques of the STM32 Microcontroller Board.
4. Implement the LED blink test using the STM32 Board.
5. Develop a circuit that includes two pushbuttons and two LEDs to control two LED separately by ensuring proper connections for efficient functionality.
6. Develop a simulation circuit model using Proteus incorporating an LED blink test and controlling two LEDs with two pushbuttons separately

Apparatus:

1.STM32 Cube IDE 1.18.0
2.STM32 Microcontroller board
3.One LED lights (Red)
4.Two resistors (220 Ω & 10k Ω)
5.One push button switches
6.Jumper wires

Circuit Diagram:

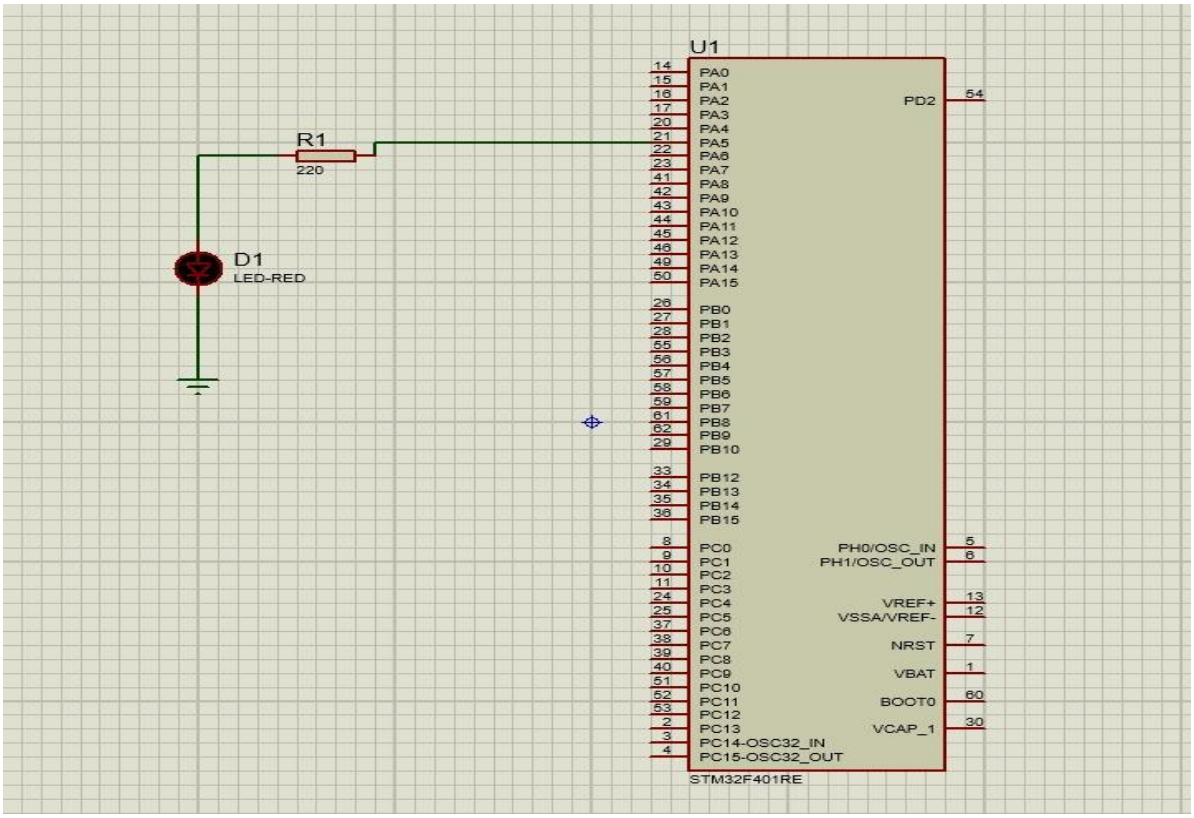

Figure 01: Circuit diagram of Blink Test using an STM32 Nucleo-F401RE Board.

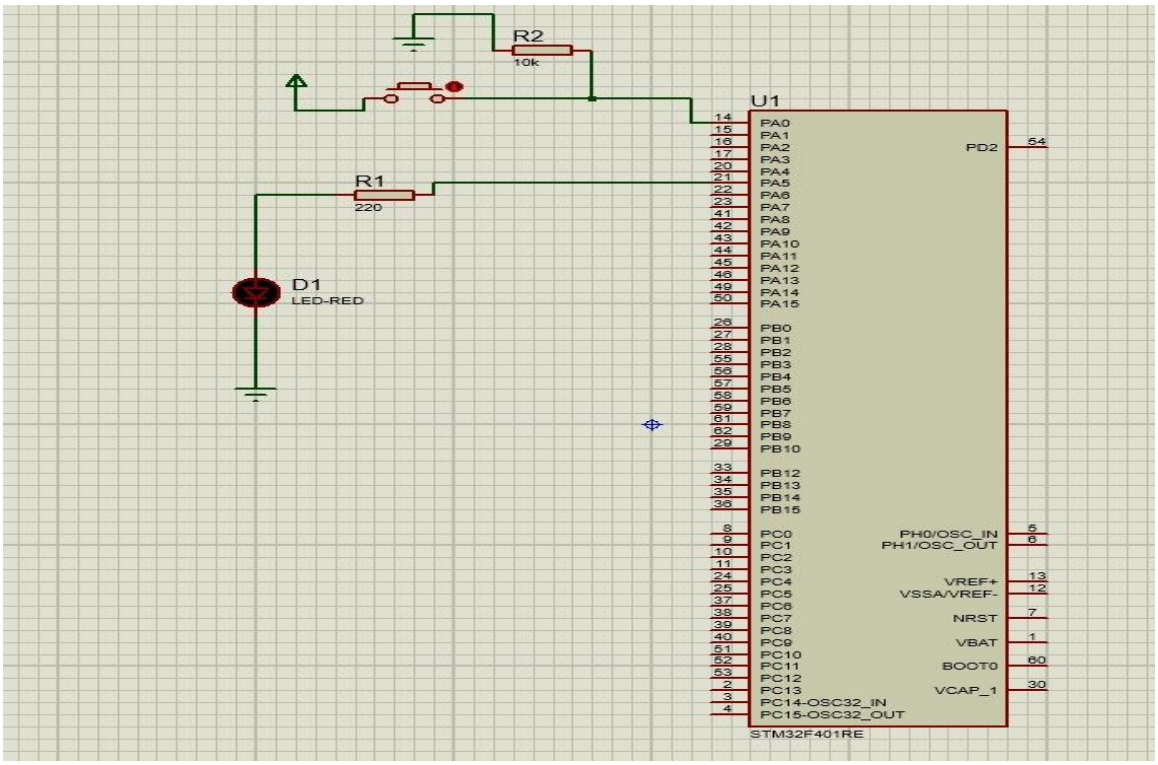

Figure 02: Circuit diagram for push button LED activation circuit using an STM32 Board

Code Explanation:

1.Program for LED blinking test:

```
while (1)
{
  /* USER CODE END WHILE */
    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
    HAL_Delay(1000);
  /* USER CODE BEGIN 3 */
}
```

Explanation: The while (1) loop in the code creates an infinite loop, ensuring that the LED blinking operation continues indefinitely until the microcontroller is reset or powered off. Inside the loop, the function HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5); is used to toggle the state of GPIOA Pin 5, which means if the LED connected to this pin is currently ON, it will turn OFF, and if it is OFF, it will turn ON. This allows the LED to blink continuously. To control the blinking speed, the HAL_Delay(1000); function introduces a delay of 1000 milliseconds (or 1 second) before the next toggle occurs. As a result, the LED remains in its current state for 1 second before changing, creating a blinking effect with a 1-second interval between each toggle.

2. Program for Controlling LEDs with Push Buttons:

```
while (1)
{
  /* USER CODE END WHILE */
    if (HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) == 0) {
    // Button 1 is pressed, turn on the Green LED and turn off the Red LED
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET); // Green LED ON
    }
    else {
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET); // Green LED ON
    }
  /* USER CODE BEGIN 3 */
}
```

Explanation: The while (1) loop in this program ensures that the microcontroller continuously checks the status of the two push buttons and responds accordingly. Inside the loop, the function HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) checks if Button 1 is pressed by reading the state of GPIOA Pin 0. If the button is pressed (logic low), the function HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET); turns ON the Green LED by setting GPIOA Pin 5 to high. If the button is not pressed, HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET); turns the LED OFF. This logic ensures that each button independently controls its respective LED, allowing separate activation and deactivation based on user input.

Hardware implementation and Explanation

1.LED blink test:

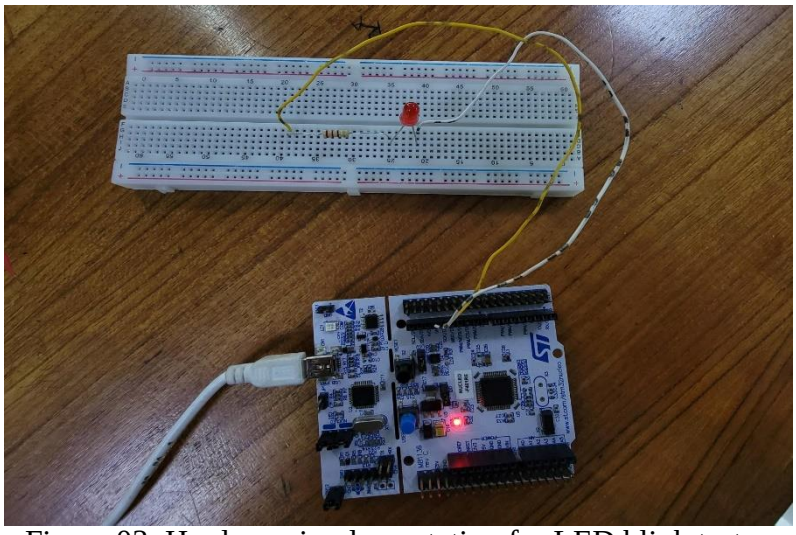

Figure 03: Hardware implementation for LED blink test

Explanation: In the following implementation, a jumper wire was connected at the D13 pin of the STM32 Microcontroller board. The wire was connected to the breadboard. The anode of an LED light was connected with 220 Ω resistor. The cathode of the LED light was connected with GND pin of the STM32 Microcontroller board. The STM32 Microcontroller Board was connected to a PC to compile and import necessary codes.

2.LED control with a push button:

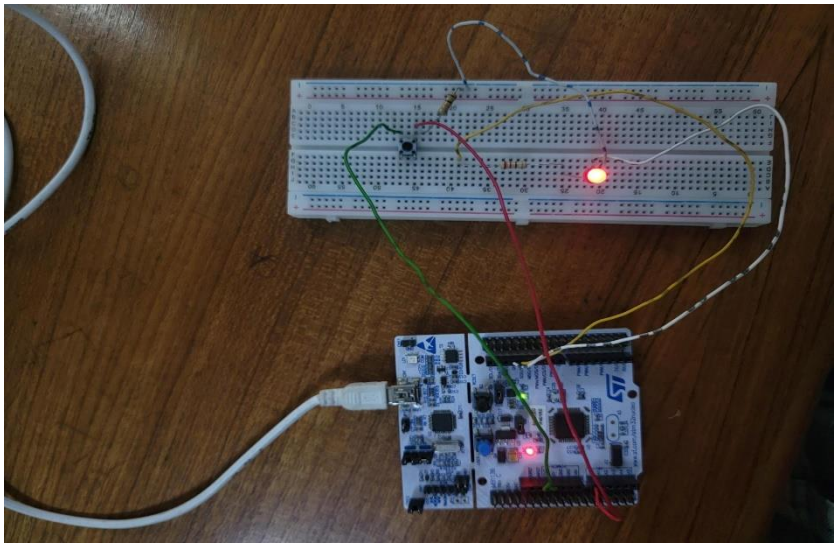

Figure 04:Hardware implementation for LED blink test using a push button

Explanation: In the following implementation, a jumper wire was connected at the D13 pin of the STM32 Microcontroller board. The wire was connected to the breadboard. The anode of an LED light was connected with a 220 Ω resistor. The cathode of the LED light was connected with the GND pin of the STM32 Microcontroller board. Additionally, a switch was connected to the breadboard, with one terminal of the switch connected to the PA5 pin of the STM32 Microcontroller board and the other terminal connected to a 10k Ω resistor. The other end of the resistor was connected to the GND to act as a pull-down resistor. The STM32 Microcontroller Board was connected to a PC to compile and import codes.

Experimental Output Results:

1. LED blink test:

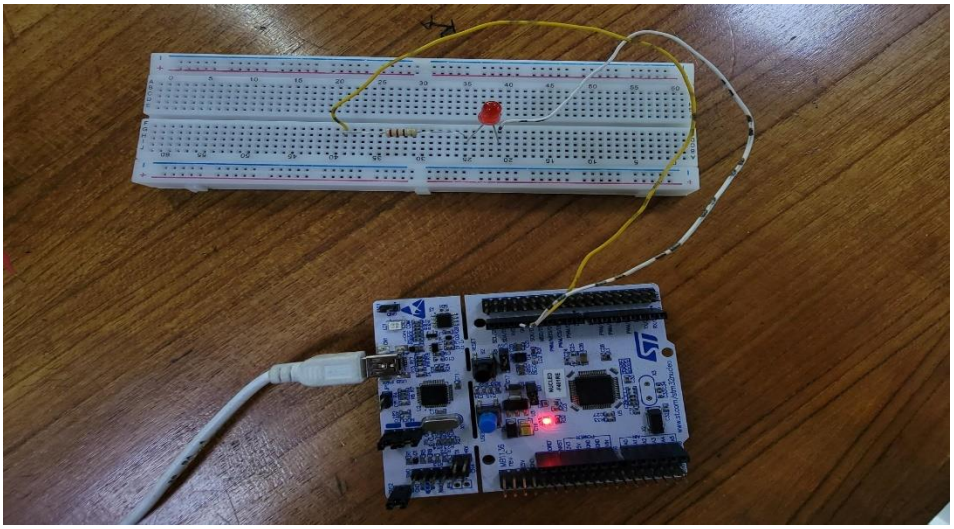

Figure 05: LED blink test (When LED is off)

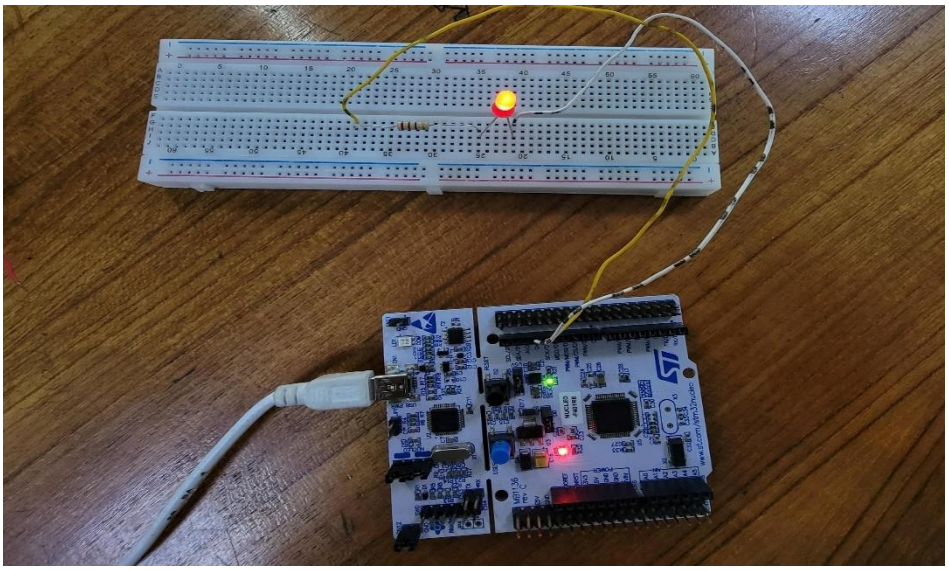

Figure 06: LED blink test (When LED is on)

Explanation: When the code is implemented and the loop() function starts, the code executes a digitalWrite() operation on the microcontroller, providing a high voltage at pin D13 as output. As a result, the LED connected to pin D13 via a 220 Ω resistor on the breadboard turns ON (Figure 06). A delay occurs afterward for a specific time period, which is defined in milliseconds. According to the implemented code, a 1000 millisecond delay occurs after the LED is turned ON.Next, another digitalWrite() operation occurs, and pin D13 is set to LOW as output. Therefore, the LED turns OFF (Figure 05). Afterward, another 1000 milliseconds delay occurs once the LED is set to low. The delay operation is achieved by using the delay() function in the loop() function.This process of turning the LED ON and OFF continues repeatedly until the microcontroller is either turned off or disconnected from the computer.

2. Push button LED activation:

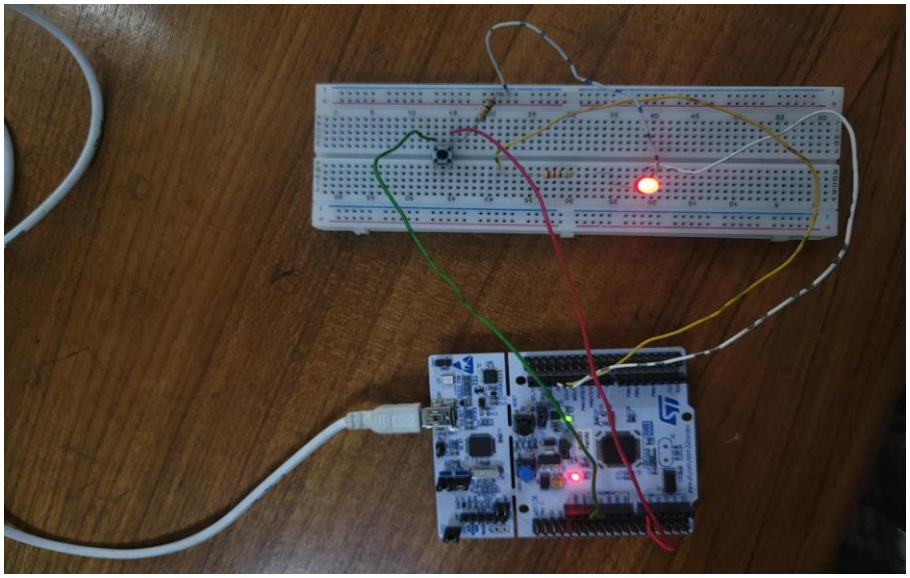

Figure 07: push button LED activation (Button not pressed LED on)

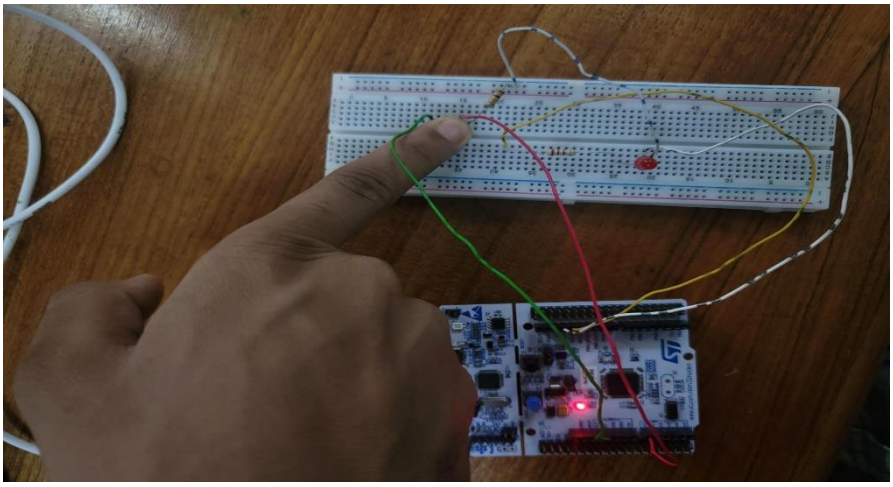

Figure 08: push button LED activation (Button pressed LED off)

Explanation: When the code runs, the loop() function continuously checks the state of the push button connected to PA5 with a 10kΩ pull-down resistor. If the button is pressed , the code turns the LED ON at D13 using digitalWrite(D13, HIGH) and keeps it on for 1000 milliseconds using delay(1000). Otherwise, if the button is not pressed, the LED is turned OFF using digitalWrite(D13, LOW) with a 1000 milliseconds delay. This process repeats until the microcontroller is powered off or disconnected.

Simulation Output Results:

1.LED blink test:

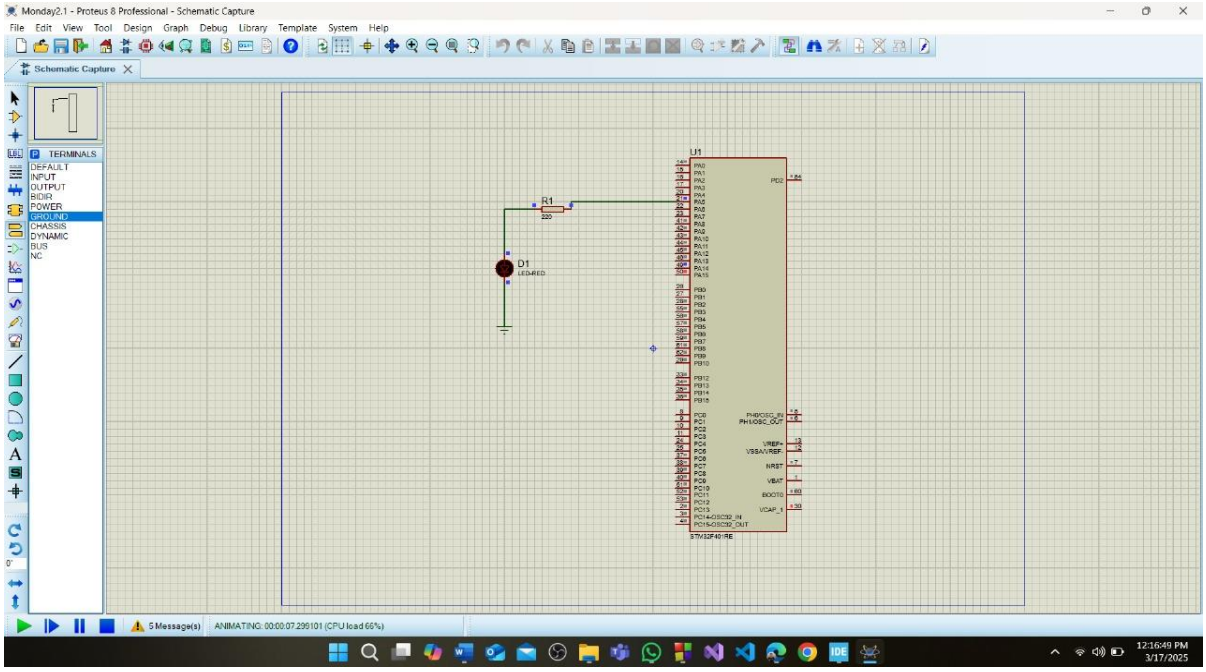

Figure 09: LED blink test on simulation (When LED is off)

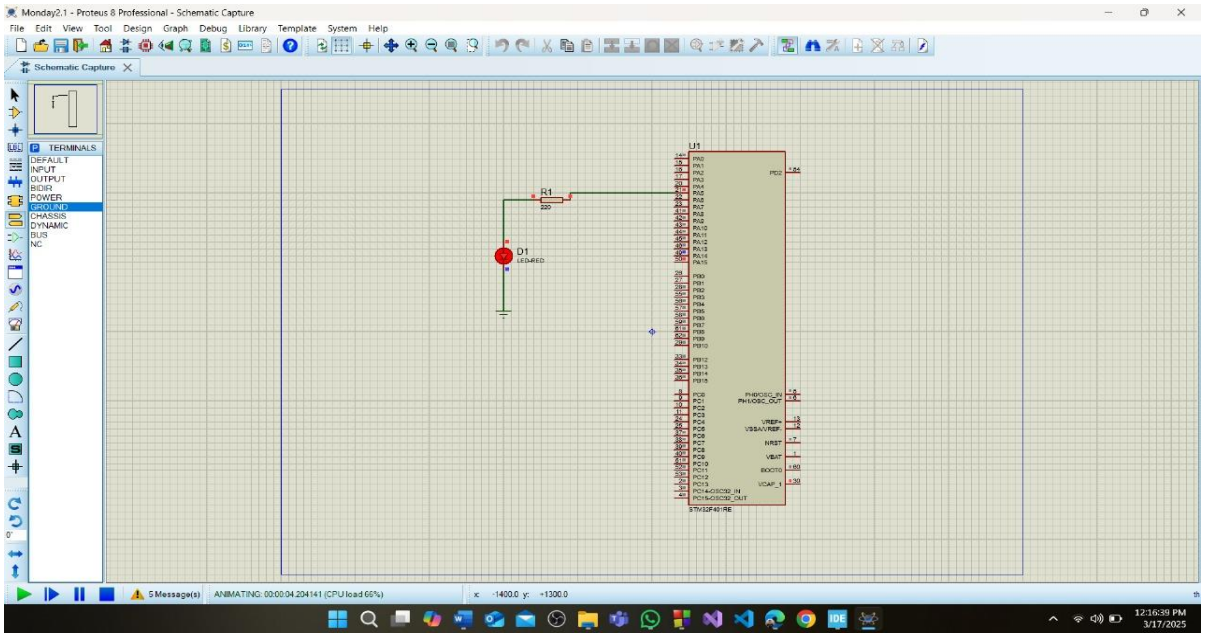

Figure 10: LED blink test on simulation(When LED is on)

Explanation: In this simulation, the STM32 microcontroller, 220 Ω resistor (R_1), and LED (D_1) were set up according to the hardware configuration used in the lab. The program was written and compiled in the STM32CubeIDE, generating a HEX file. This file was then imported into the Proteus simulation to simulate the behavior of the circuit. When the simulation was run, the LED's operation was observed based on the programmed logic to verify the performance of the system and ensure it behaves as expected compared to the actual hardware setup.

2.Push button LED control:

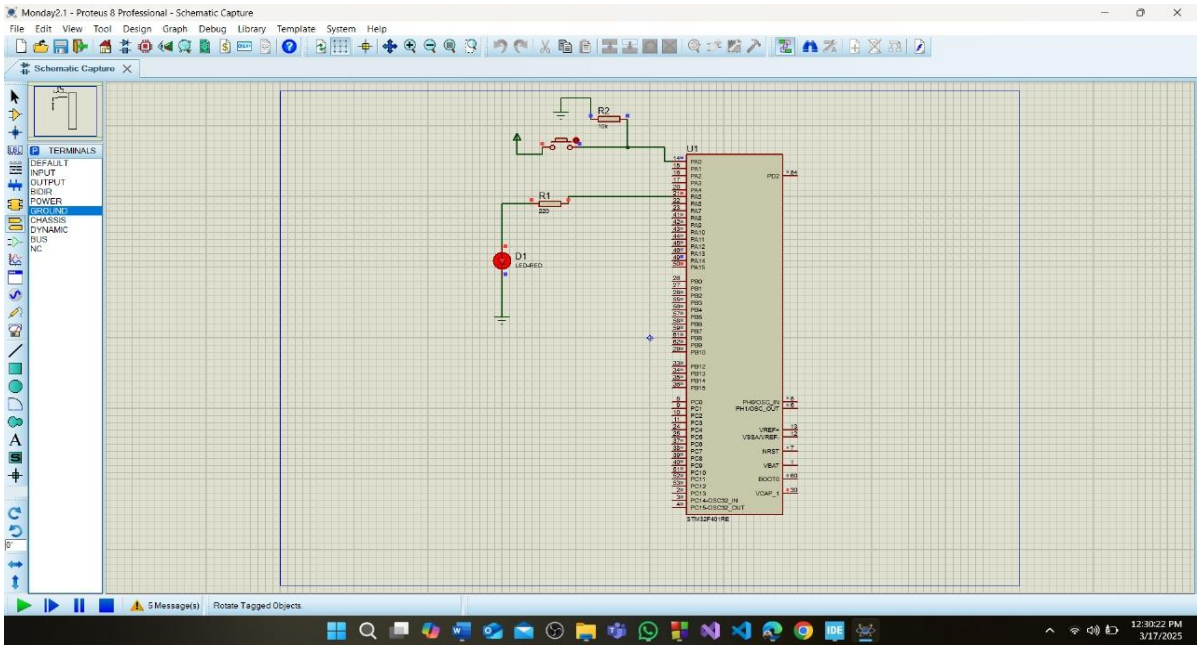

Figure 11: LED blink test using push button on simulation(When button is not pushed)

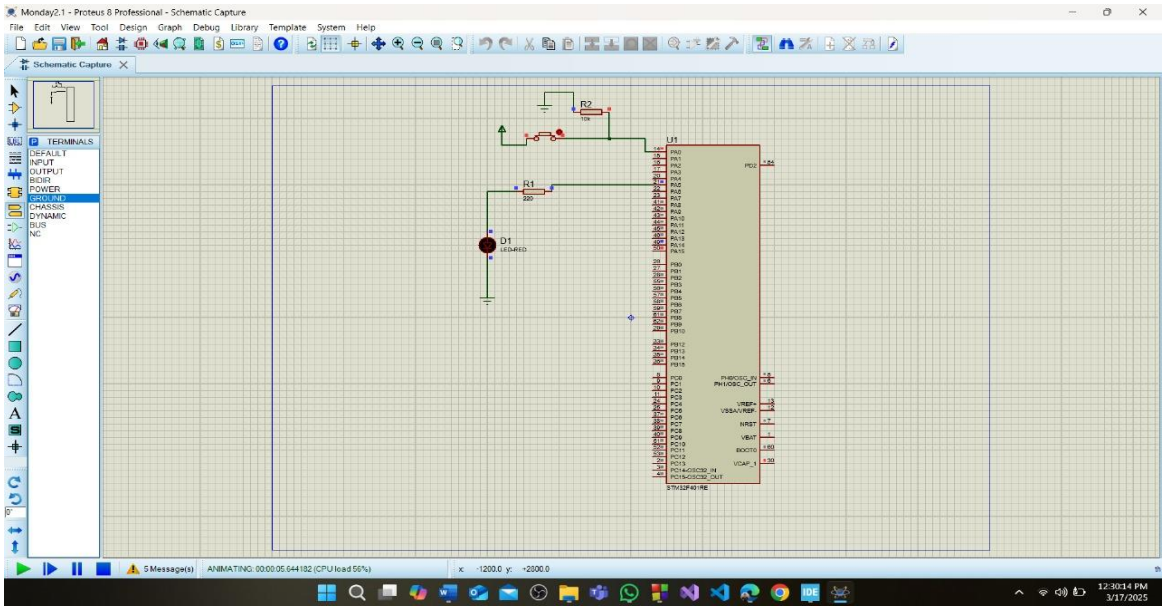

Figure 12: LED blink test using push button on simulation(When button is pushed)

Explanation: In this simulation, the STM32 microcontroller, 220Ω resistor (R1), LED (D1), and a push button were configured according to the hardware setup used in the lab. The push button was connected to PA5 of the microcontroller with a 10kΩ resistor to ensure a stable LOW state when the button is not pressed. The program was written and compiled in STM32CubeIDE, generating a HEX file that was imported into the Proteus simulation. The simulation was run to observe the LED's operation based on the button press. When the button is pressed, the LED connected to D13 turns ON, and when released, the LED turns OFF. The simulation results were compared with the actual hardware to verify the circuit's performance.

Answer to Question:
(Traffic light control)

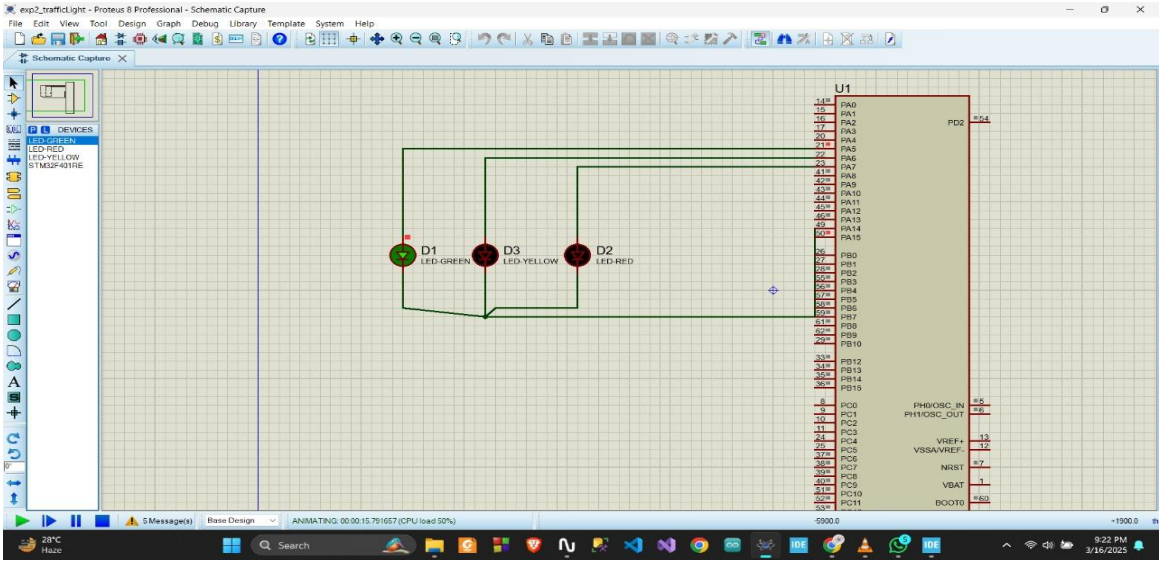

Figure 13: Traffic light control system (When Green LED is on)

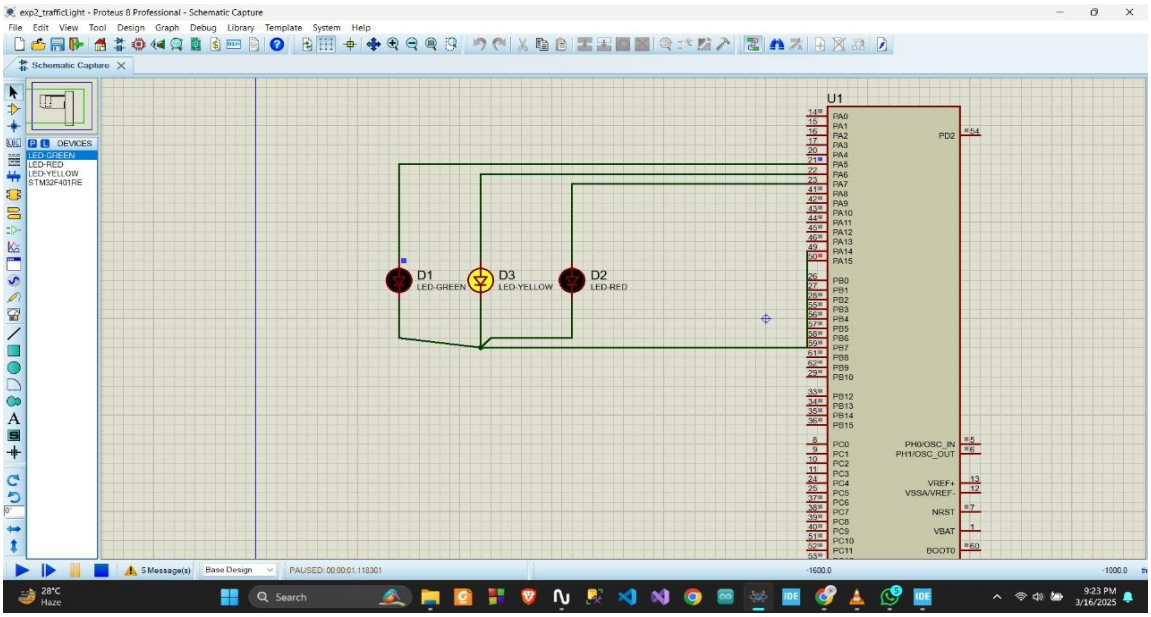

Figure 14: Traffic light control system (When Yellow LED is on)

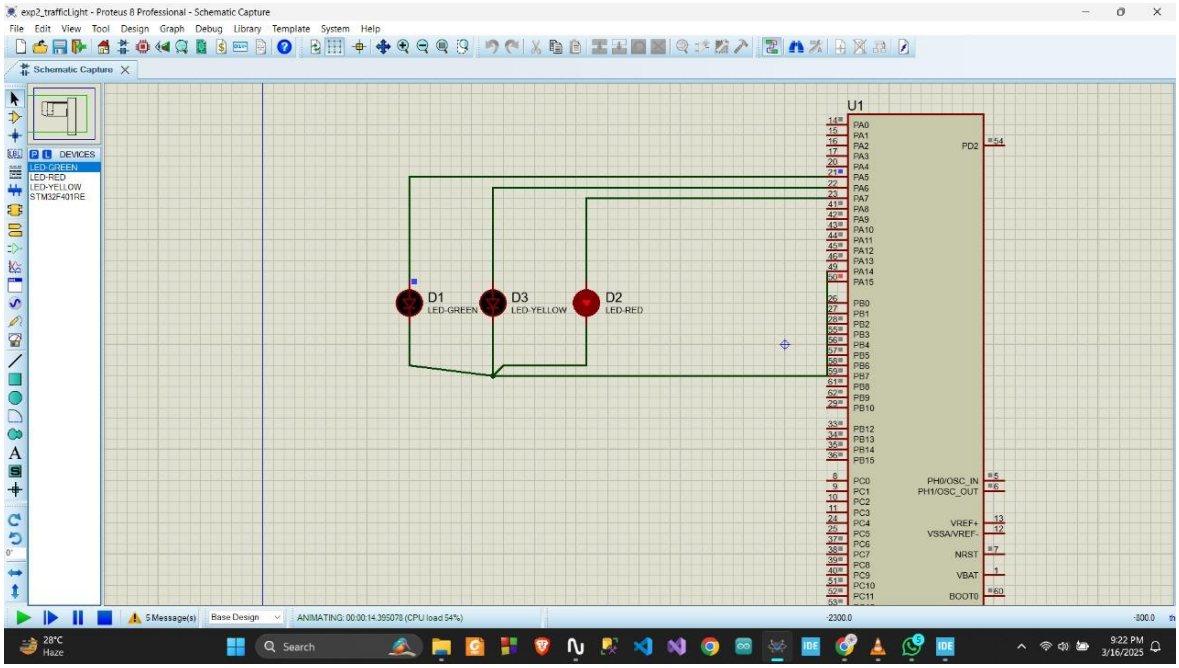

Figure 15: Traffic light control system (When Red LED is on)

Explanation: In the traffic light control system simulation, we used Proteus for circuit design and STM32CubeIDE for programming. The circuit consists of three LEDs (Red, Yellow, and Green) connected to STM32 GPIO pins through current-limiting resistors. The microcontroller controls the LEDs in a predefined sequence to simulate real traffic light operation. The Green LED turns ON for 5 seconds, followed by the Yellow LED for 2 seconds, and then the Red LED for 5 seconds before repeating the cycle. The program uses HAL_Delay() functions or timers in STM32CubeIDE to implement time delays for switching between LED states. The simulation in Proteus allows us to verify the correct operation of the system, ensuring proper sequencing of the LEDs.

Discussion:

This experiment provided a comprehensive understanding of the STM32 microcontroller board and its basic functionalities. Initially, we explored how to configure the board and execute a simple LED blink test, which helped us understand GPIO (General-Purpose Input/Output) control, timing mechanisms, and fundamental coding principles in embedded systems. By implementing the LED blink test, we observed how delays affect microcontroller operations and how precise timing is essential in embedded applications.

The second phase of the experiment involved creating a push button LED activation circuit. This part required understanding input handling and implementing logic-based control mechanisms. We had to ensure that button presses were correctly detected, which introduced us to concepts such as debouncing—preventing unintended multiple detections from a single press due to mechanical switch noise. This practical approach allowed us to observe the real-world challenges of working with physical inputs and microcontrollers.

Real-Life Scenarios

1. **Home Automation:** The concept of using push buttons to control LEDs can be extended to smart home applications, such as controlling lights, fans, or even security systems with a simple button press.
2. **Traffic Light Systems:** Microcontrollers are widely used in traffic signal control systems where LEDs represent stop, go, and caution signals. The timing mechanisms we explored in the LED blink test are similar to those used in such systems.
3. **Consumer Electronics:** Many household devices, such as remote controls, microwave ovens, and digital clocks, rely on push-button interfaces similar to the circuit we implemented.
4. **Medical Equipment:** Devices like patient monitoring systems and ventilators use microcontrollers for user input, status indication, and sensor integration.
5. **Automotive Applications:** In modern vehicles, microcontrollers control dashboard indicators, touch panels, and various user input systems that function similarly to our push-button LED circuit.

Conclusion:

In conclusion, the experiment provided valuable hands-on experience with the STM32 microcontroller board, focusing on implementing an LED blink test and a 2-push button LED activation circuit. The LED blink test demonstrated the basic process of programming microcontrollers for output control using timing functions. Integrating push buttons allowed manual control over the LED, enhancing the understanding of input handling and interrupt management. This experiment successfully demonstrated how simple microcontroller operations can be applied to build more advanced systems, providing essential knowledge for future projects involving embedded systems and automation.

References:

[1] https://en.wikipedia.org/wiki/STM32
[2] https://www.labcenter.com
[3] *Aiub lab manual*