# AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

## Faculty of Engineering

## Lab Report

### Experiment # 02

### Experiment Title: Familiarization with an STM32, the study of blink test and implementation of a light controlling system using microcontrollers.

| | | | |
|---|---|---|---|
| **Course Title:** | MICROPROCESSOR AND EMBEDDED SYSTEMS LAB | | |
| **Course Code:** | COE3104 | **Section:** | A |
| **Semester:** | Spring 2022-23 | **Degree Program:** | BSc in CSE/BSc in EEE |
| **Course Teacher:** | **Md. Ali Noor** | | |




**Group # 06**

| Sl No | Name | ID | Program |
|---|---|---|---|
| 1 | Syed Aftab Uddin | 19-41522-3 | EEE |
| 2 | Syeda Aynul Karim | 19-41829-3 | CSE |
| 3 | Md. Sumon | 20-42556-1 | CSE |
| 4 | Maimona Rahman Farjana | 20-42954-1 | CSE |
| 5 | Islam, Md. Rashedul | 20-43301-1 | CSE |
| 6 | Medha Chowdhury | 20-41930-1 | CSE |


***Faculty use only***

| | | |
|---|---|---|
| FACULTY COMMENTS | **Marks Obtained** | |
| | **Total Marks** | |

## Experiment Title:

Familiarization with an STM32, the study of blink test, and implementation of a light-controlling system using microcontrollers.

## Introduction:

The objective of this experiment is to get familiarized with STM Microcontroller. In our lab we use STM32-Nucleo-F401RE.
It features the ARMCortexM4 32-bit STM32F401RE microcontroller which is in the LQFP64 package. The Boards pinout is similar to Arduino UNO and has many other additional pins to expand performance. This board also comes with an integrated ST-LINK/V2-1 programmer and debugger; hence it is very easy to get started with this board.

## Objectives:

The objectives of this experiment are to Familiarize with the STM32CubeIDE and STM32F401RE microcontroller, implement a simple circuit to make an LED light to blink using ST32 and implementation of a light control system using STM32.

## Equipment List:

1. STM32F401RE
2. Breadboard
3. LED lights (red, yellow, green)
4. Jumper wires
5. Computer
6. Resistor (Three 200 ohms)
7. STM32Cube IDE (any version)

## Circuit Diagram:

The STM32 board is made up of the following

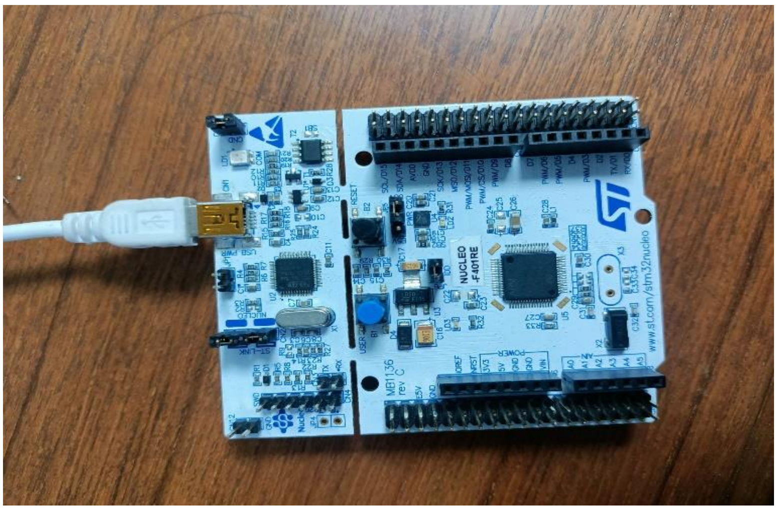

Figure 1: STM32 board components.

## Simulation diagram of STM32 board:

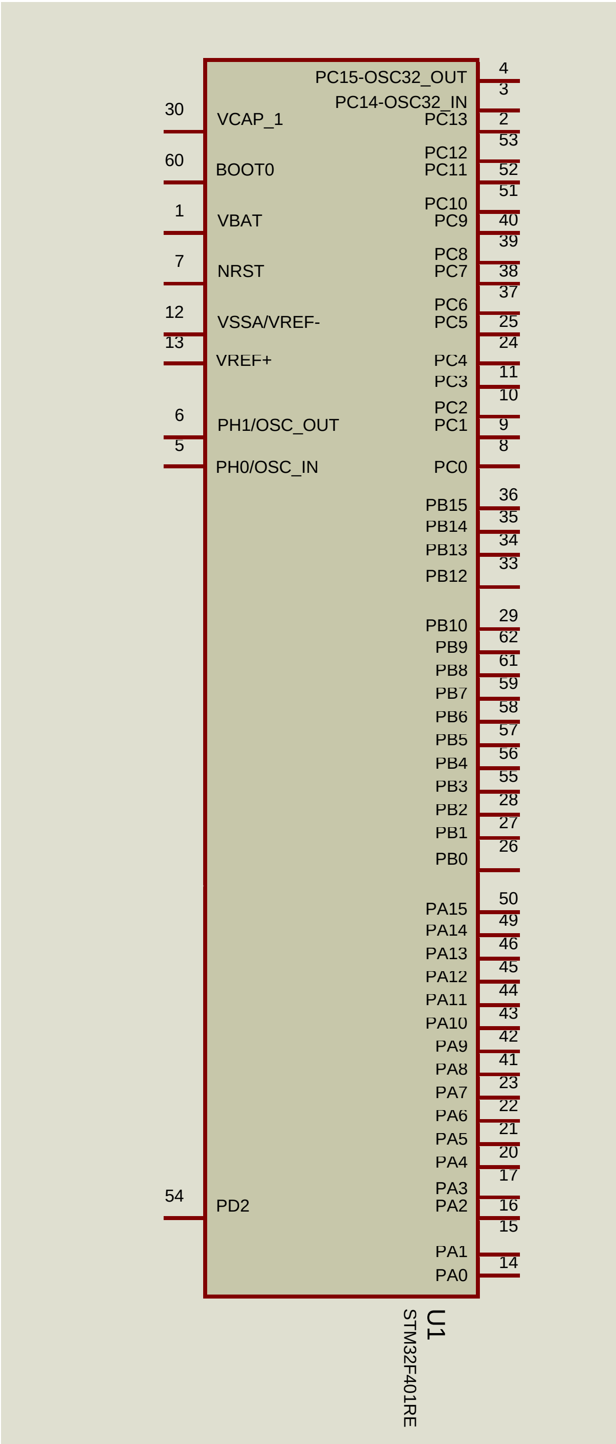


Figure 2: STM32 simulation board (Proteus)

## Simulation Set-up:

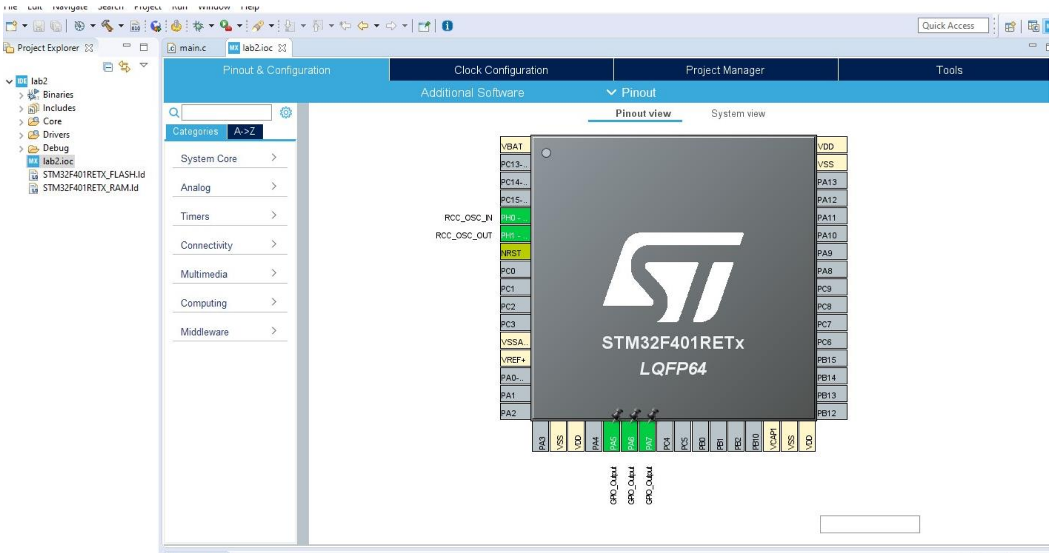


Fig 3: Setup PIN configuration from STM32 Cube IDE

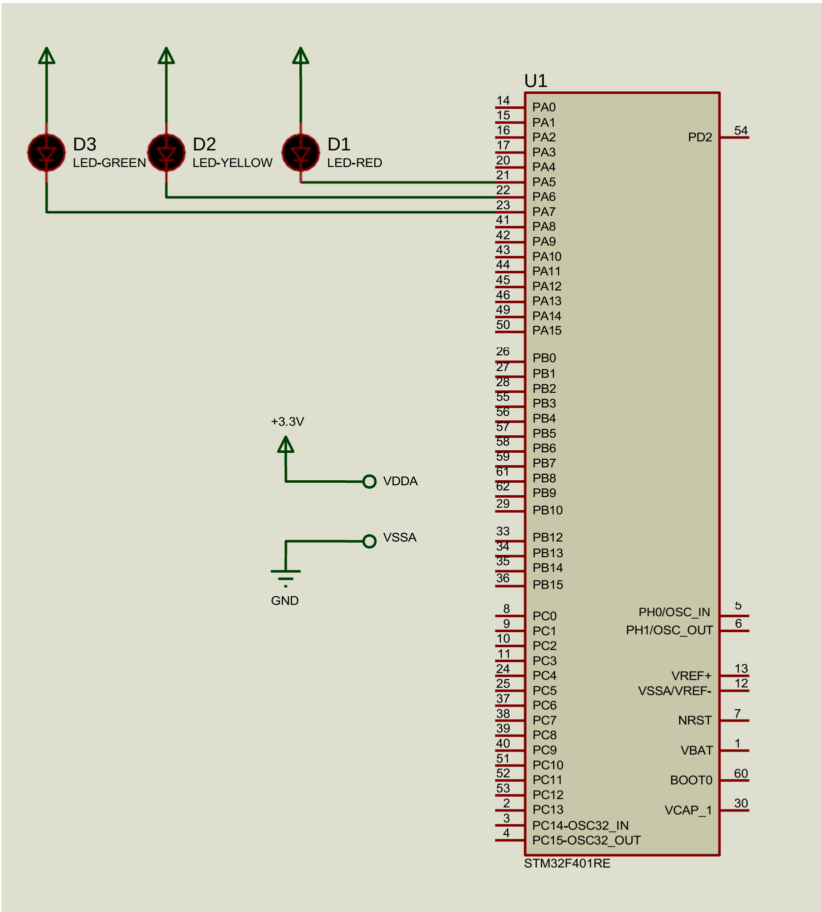


Fig 4: Setup traffic light system from proteus

## Source Code:

```
#include "main.h"
UART_HandleTypeDef huart2;
static void MX_GPIO_Init(void);
static void MX_USART2_UART_Init(void);
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  MX_USART2_UART_Init();
  while (1)
  {
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5,1);
		HAL_Delay(3000);
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5,0);

		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6,1);
		HAL_Delay(2000);
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6,0);

		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7,1);
		HAL_Delay(4000);
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7,0);
for(int i=0; i<3;i++){
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7,1);
		HAL_Delay(500);
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7,0);
		HAL_Delay(500);
}
  }
}

void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE2);

  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
  RCC_OscInitStruct.PLL.PLLM = 16;
  RCC_OscInitStruct.PLL.PLLN = 336;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV4;
  RCC_OscInitStruct.PLL.PLLQ = 7;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
```

```
                            |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
  {
    Error_Handler();
  }
}

static void MX_USART2_UART_Init(void)
{

  huart2.Instance = USART2;
  huart2.Init.BaudRate = 115200;
  huart2.Init.WordLength = UART_WORDLENGTH_8B;
  huart2.Init.StopBits = UART_STOPBITS_1;
  huart2.Init.Parity = UART_PARITY_NONE;
  huart2.Init.Mode = UART_MODE_TX_RX;
  huart2.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart2.Init.OverSampling = UART_OVERSAMPLING_16;
  if (HAL_UART_Init(&huart2) != HAL_OK)
  {
    Error_Handler();
  }

}

static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();

  HAL_GPIO_WritePin(GPIOA, LD2_Pin|GPIO_PIN_6|GPIO_PIN_7, GPIO_PIN_RESET);

  GPIO_InitStruct.Pin = B1_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(B1_GPIO_Port, &GPIO_InitStruct);
  GPIO_InitStruct.Pin = LD2_Pin|GPIO_PIN_6|GPIO_PIN_7;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

}
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
```

```
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
}

#ifdef  USE_FULL_ASSERT
void assert_failed(uint8_t *file, uint32_t line)
{
}
#endif
```

## Hardware Set-up:

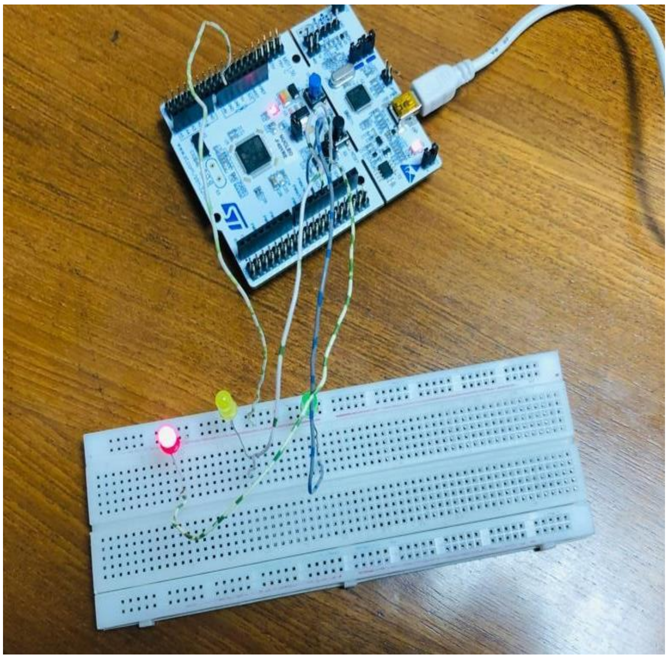

Fig 5: Red-light turned-on

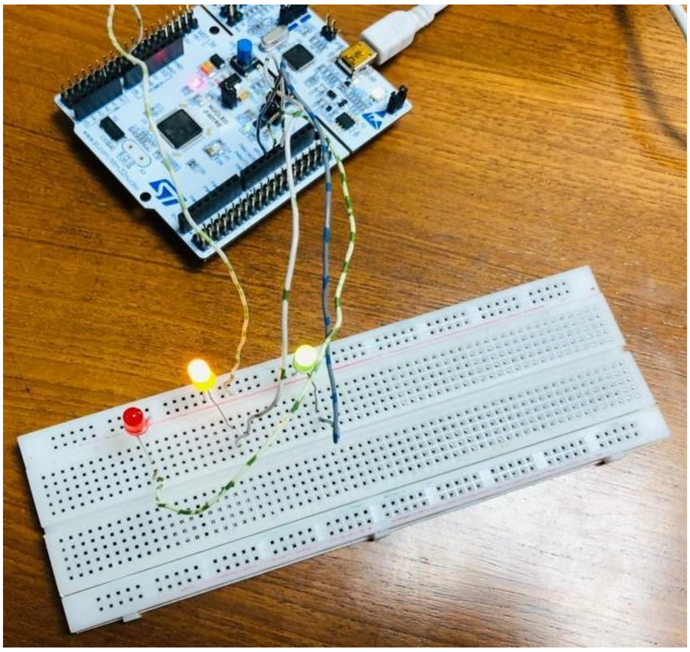

Fig 6: Yellow light turned on

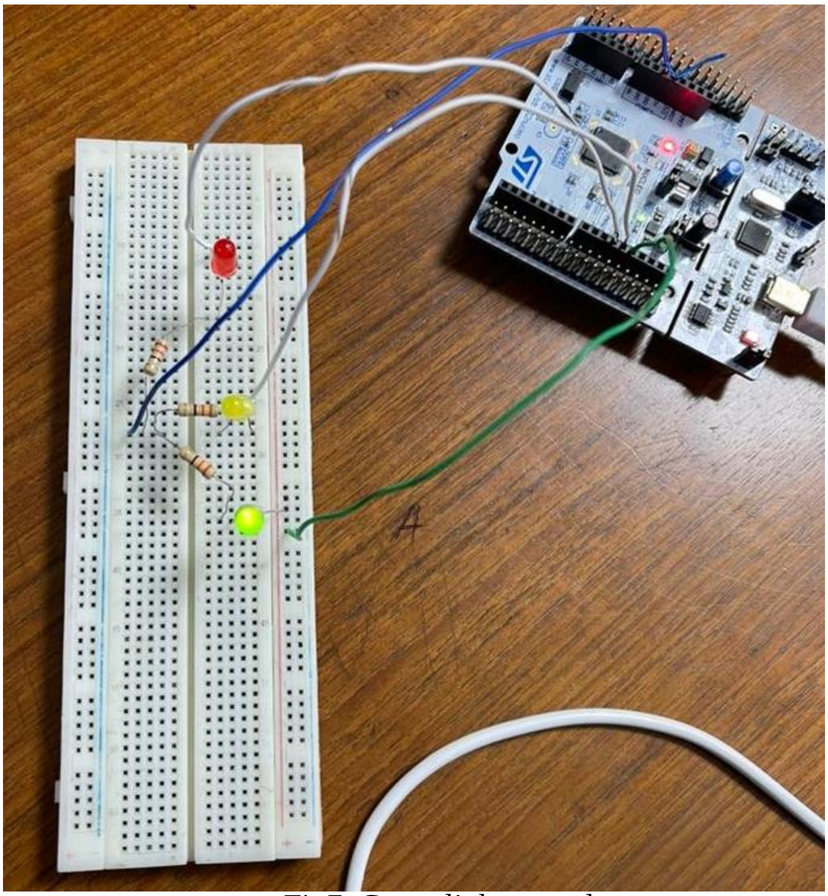

Fig7: Green light turned on

## **Simulation Output Results:**

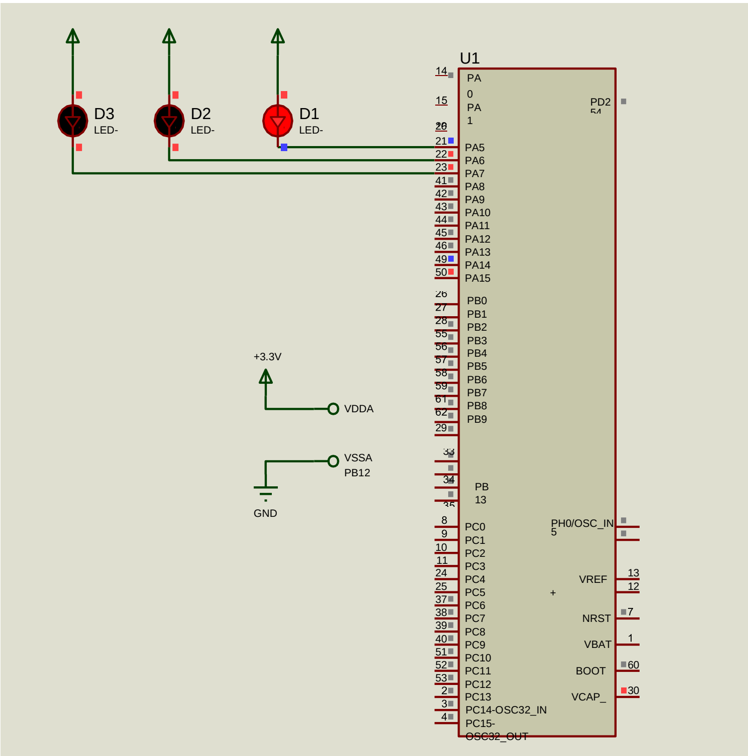


Fig 8: Red light turned on

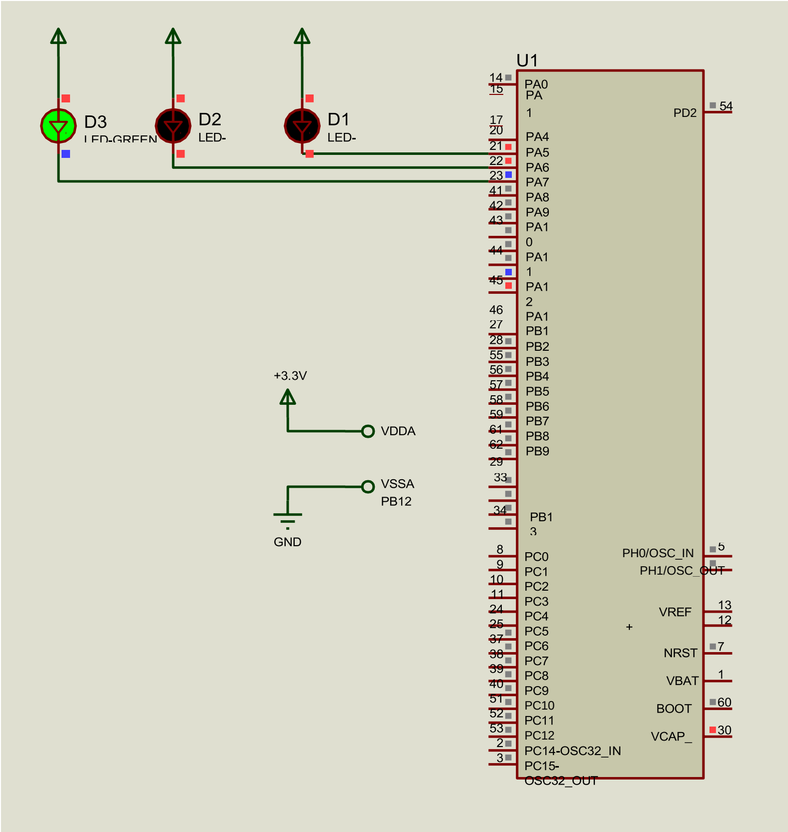


Fig 9: Green light turned on

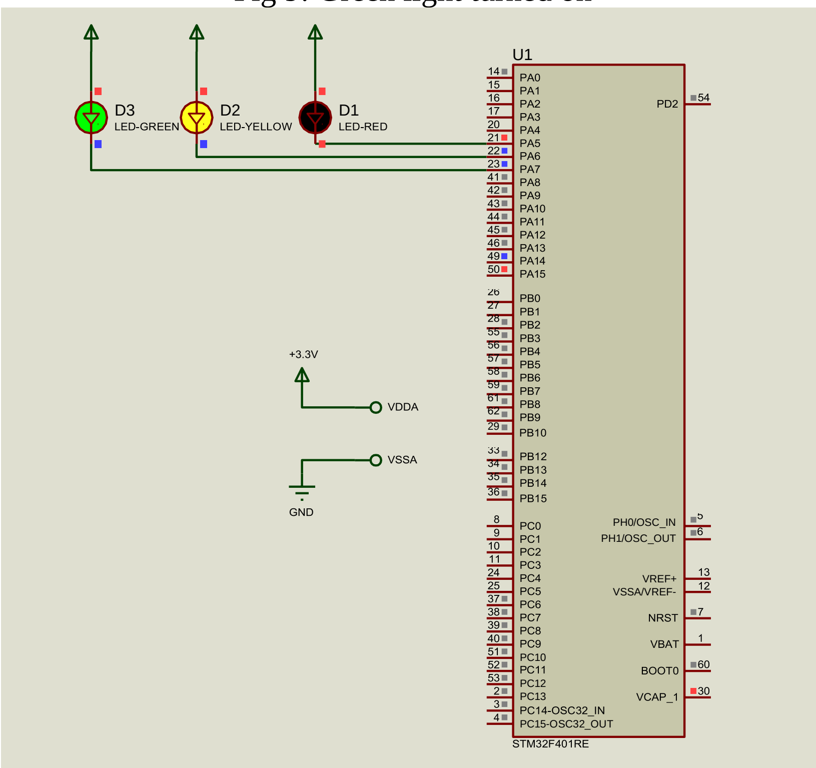


Fig 10: Yellow light turned on

**Discussion and Conclusion:** In the following experiment, the primary target was to acquire knowledge of a microcontroller STM32 and assemble a demo light control system. The experiment with three LED lights (RED, Green, and Yellow) an STM32 board, and some wires to prepare a light system.
After connecting the LEDs and wires to the board as shown in the image, it was time to connect them to the STM32 microcontroller. STM32 had 28 pins. Wires are gained in PA5, PA6, and PA7 at the end. Once circuit construction has been done, the USB should be inserted into the computer to develop program commands on the STM32 board. The required code was written in a "while loop" which was 4 to 5 lines asshown in the "Source code" section. After building and running the program from STM32 cube IDE, theboard showed results to blinking LED lights.
Then fixed the problem and the desired result was obtained as the light control system was successfully established and working by using the STM32 microcontroller.

## References:


1) Lab manual
2) STM32 Cube IDE https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html
3) Proteus 8 Professional v8.11