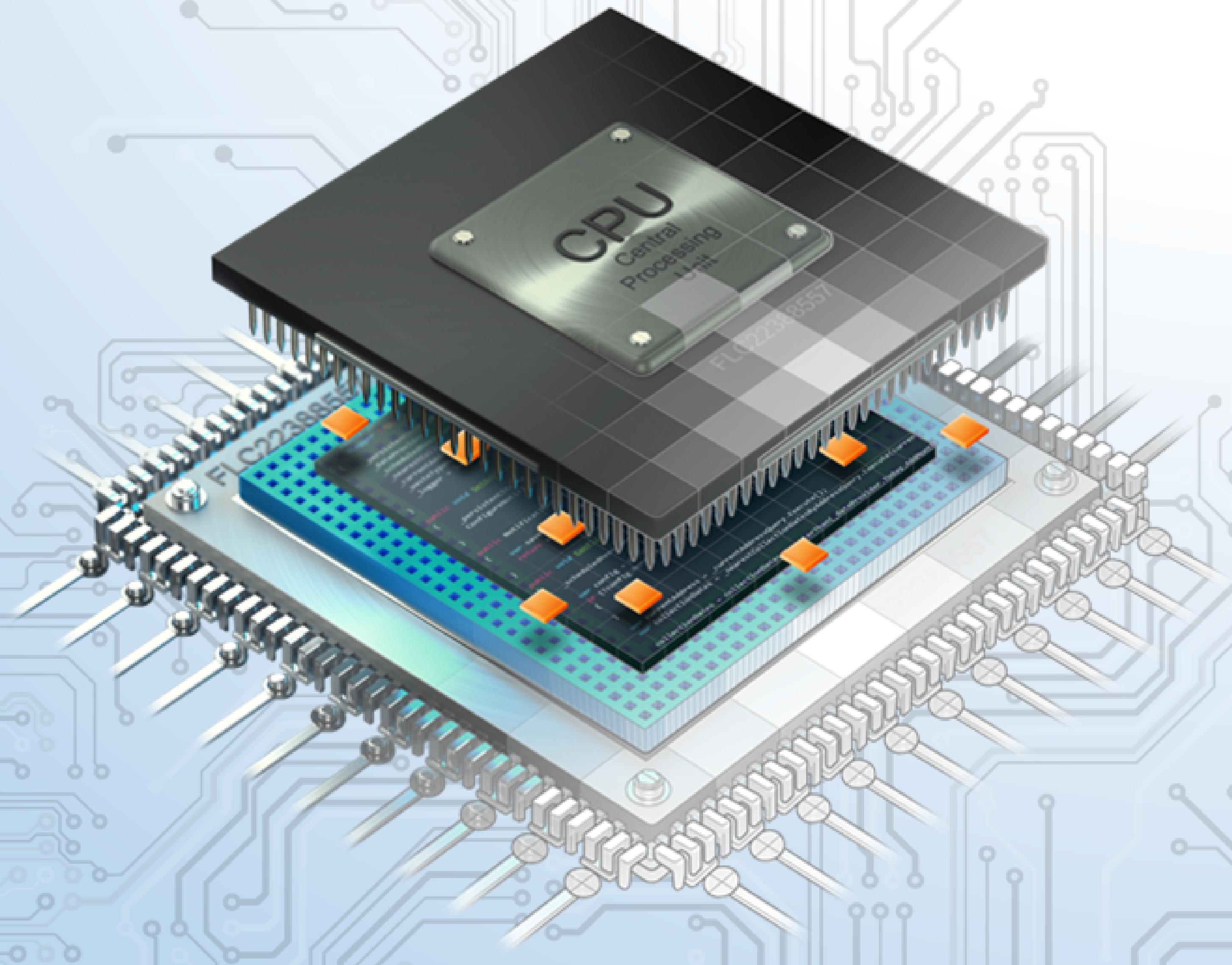
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
COMPUTER ENGINEERING

Microcontroller



Dr. Le Trong Nhan

VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

Microprocessor - Microcontroller (CO3009)

Laboratory Report

LAB 1

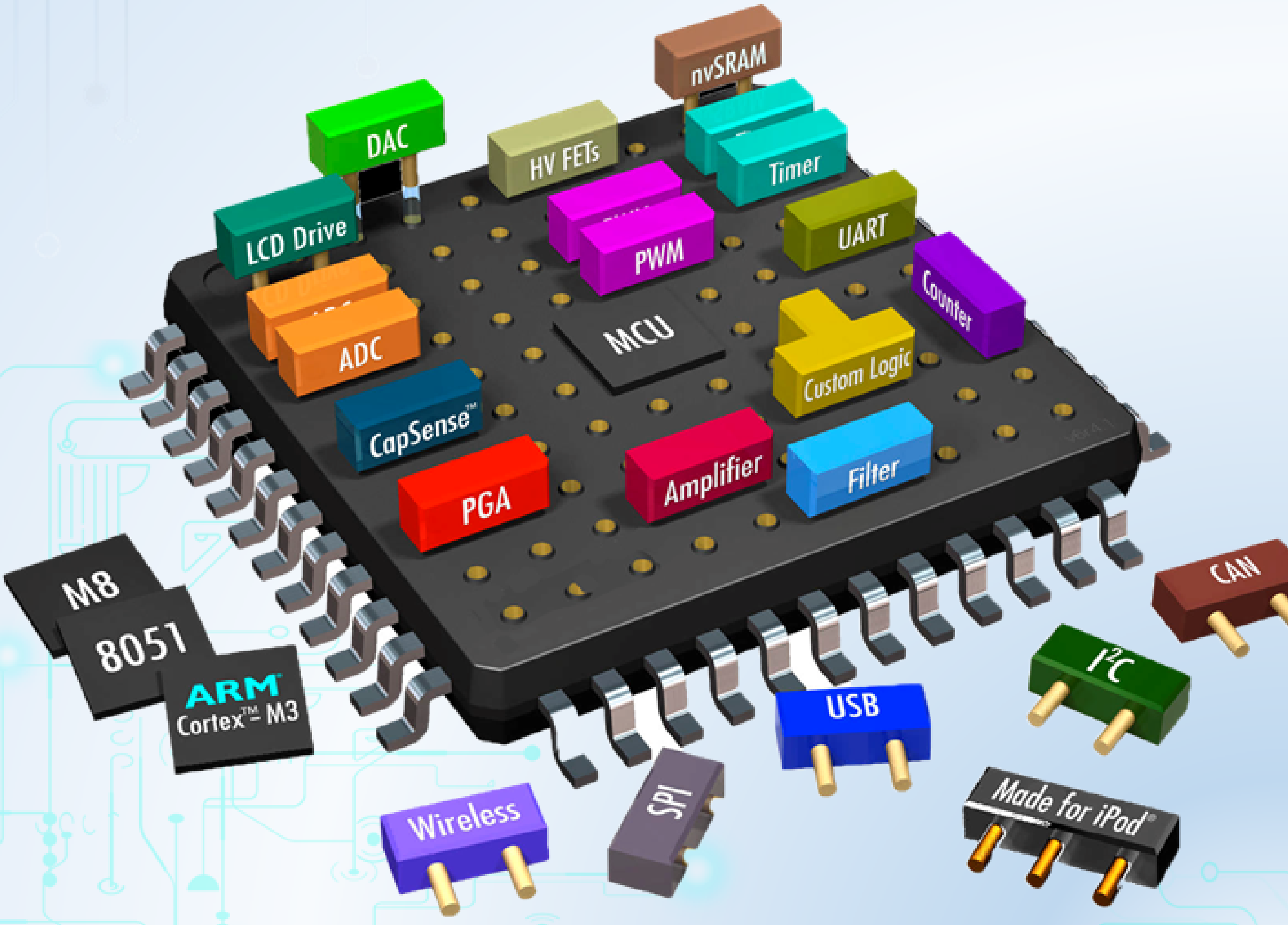
Teacher:	Le Trong Nhan Huynh Phuc Nghi
Class:	L03
Student:	Pham Van Nhat Vu (2110676)

HO CHI MINH CITY, 2023

Contents

CHAPTER 1

LED Animations



1 Exercise and Report

1.1 Exercise 1

From the simulation on Proteus, one more LED is connected to pin **PA6** of the STM32 (negative pin of the LED is connected to PA6). The component suggested in this exercise is **LED-YELLOW**, which can be found from the device list.

In this exercise, the status of two LEDs are switched every 2 seconds, as demonstrated in the figure bellow.

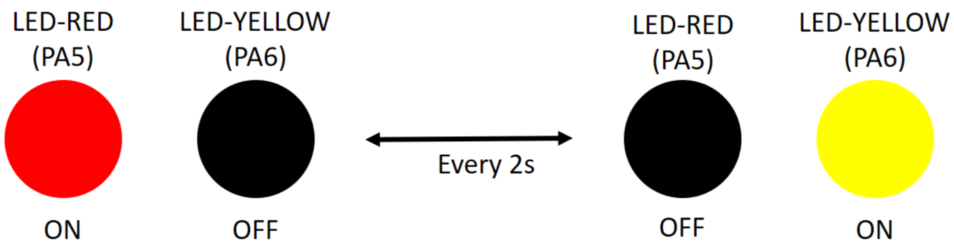


Figure 1.1: State transitions for 2 LEDs

Report 1: Depict the schematic from Proteus simulation in this report. The caption of the figure is a downloadable link to the Proteus project file (e.g. a github link).

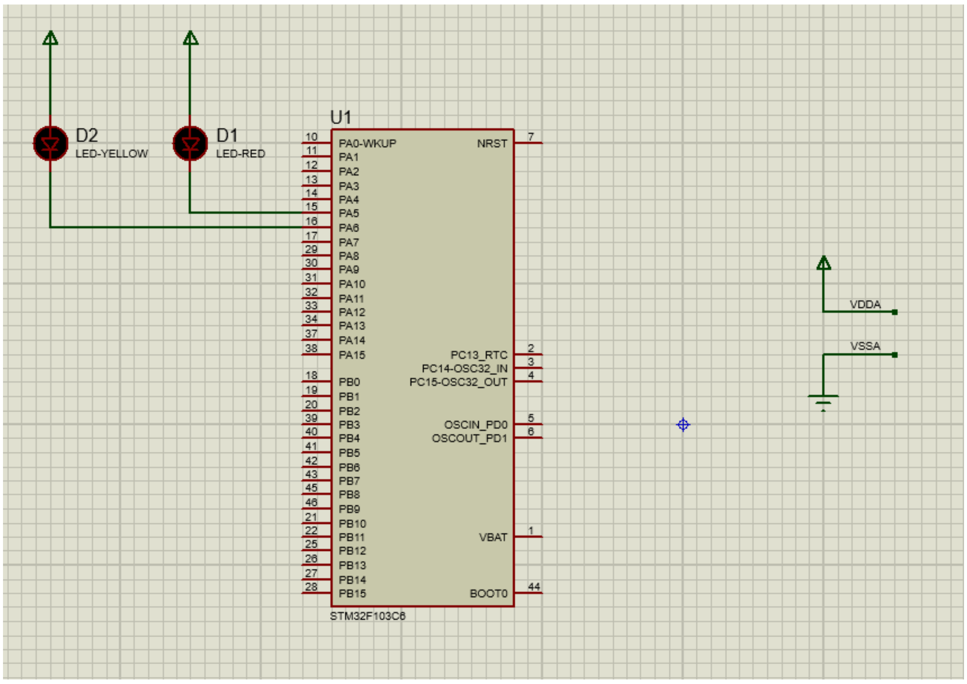


Figure 1.2: Schematic of Proteus simulation project

Report 2: Present the source code in the infinite loop while of your project. If a user-defined functions is used, it is required to present in this part. A brief description can be added for this function (e.g. using comments).

```c
#define TRANSITION_TIME 2
...
HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin,
    GPIO_PIN_RESET);
HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port, YELLOW_LED_Pin,
    GPIO_PIN_SET);
uint8_t count = TRANSITION_TIME;
while (1)
{
  if (count <= 0)
  {
      count = TRANSITION_TIME;
      HAL_GPIO_TogglePin(RED_LED_GPIO_Port, RED_LED_Pin);
      HAL_GPIO_TogglePin(YELLOW_LED_GPIO_Port,
    YELLOW_LED_Pin);
  }
  --count;
  HAL_Delay(1000);
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
}
```

Program 1.1: Source code for exercise 1

1.2 Exercise 2

Extend the first exercise to simulate the behavior of a traffic light. A third LED, named **LED-GREEN** is added to the system, which is connected to **PA7**. A cycle in this traffic light is 5 seconds for the RED, 2 seconds for the YELLOW and 3 seconds for the GREEN. The LED-GREEN is also controlled by its negative pin.

Similarly, the report in this exercise includes the schematic of your circuit and a your source code in the while loop.

Report 1: Present the schematic.

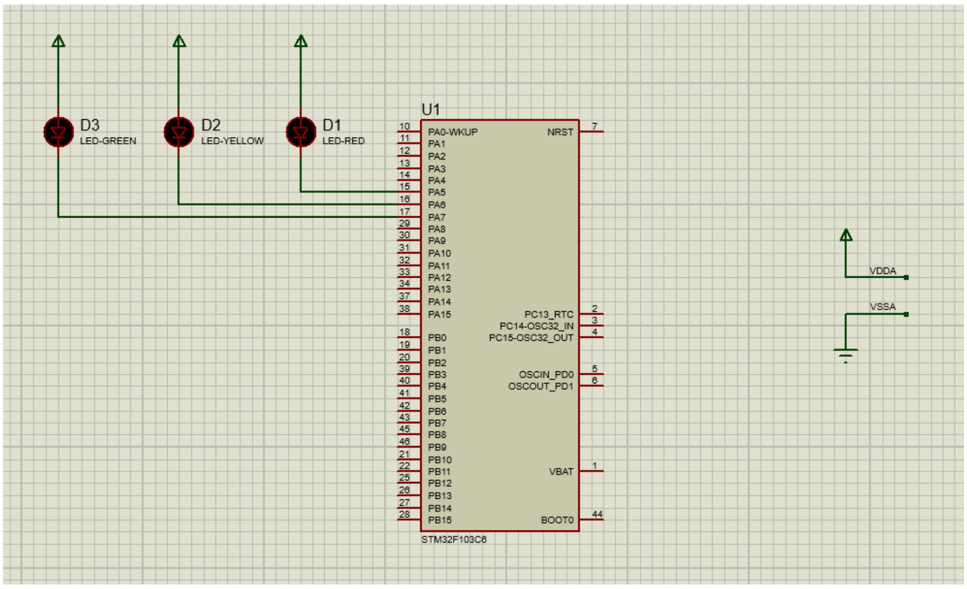


Figure 1.3: Schematic of Proteus simulation project

Report 2: Present the source code in while.

```
#define RED_DURATION 5
#define YELLOW_DURATION 2
#define GREEN_DURATION 3
#define RED_STATE 0
#define GREEN_STATE 1
#define YELLOW_STATE 2
...
HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin, GPIO_PIN_RESET);
HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port, YELLOW_LED_Pin, GPIO_PIN_SET);
HAL_GPIO_WritePin(GREEN_LED_GPIO_Port, GREEN_LED_Pin, GPIO_PIN_SET);
uint8_t count = RED_DURATION;
uint8_t state = RED_STATE;
while (1)
{
  switch (state)
  {
  case RED_STATE:
      HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin,
   GPIO_PIN_RESET);
      if (count <= 0)
      {
          count = GREEN_DURATION;
          state = GREEN_STATE;
          HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin,
    GPIO_PIN_SET);
          HAL_GPIO_WritePin(GREEN_LED_GPIO_Port,
   GREEN_LED_Pin, GPIO_PIN_RESET);
      }
      break;
  case GREEN_STATE:
      HAL_GPIO_WritePin(GREEN_LED_GPIO_Port, GREEN_LED_Pin,
    GPIO_PIN_RESET);
      if (count <= 0)
      {
          count = YELLOW_DURATION;
          state = YELLOW_STATE;
          HAL_GPIO_WritePin(GREEN_LED_GPIO_Port,
   GREEN_LED_Pin, GPIO_PIN_SET);
          HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port,
   YELLOW_LED_Pin, GPIO_PIN_RESET);
      }
      break;
  case YELLOW_STATE:
      HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port,
   YELLOW_LED_Pin, GPIO_PIN_RESET);
      if (count <= 0)
      {
          count = RED_DURATION;
          state = RED_STATE;
          HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port,
   YELLOW_LED_Pin, GPIO_PIN_SET);
          HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin,
    GPIO_PIN_RESET);
      }
      break;
  }
  --count;
  HAL_Delay(1000);
}
```

Program 1.2: Source code for exercise 2

1.3 Exercise 3

Extend to the 4-way traffic light. Arrange 12 LEDs in a nice shape to simulate the behaviors of a traffic light. A reference design can be found in the figure bellow.

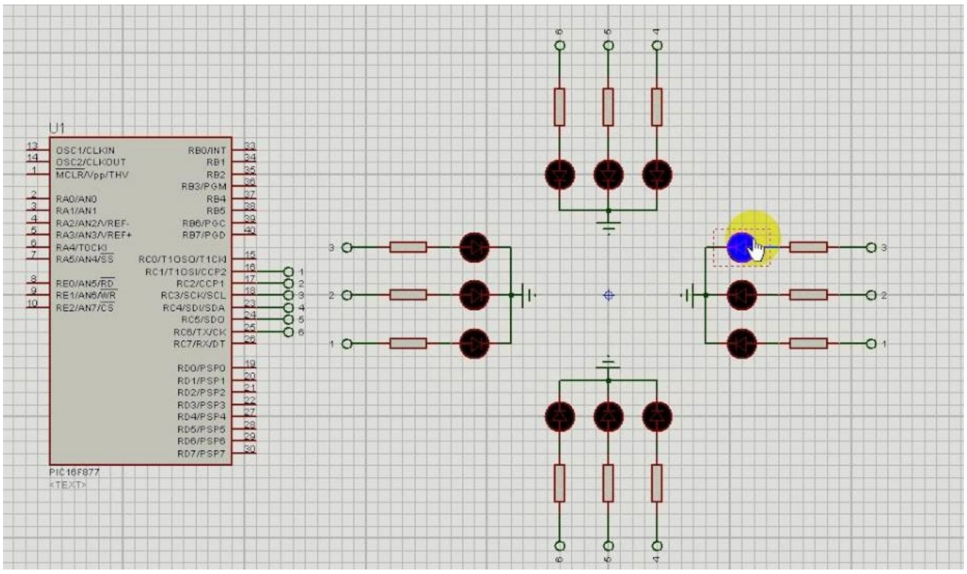


Figure 1.4: Reference design for a 4 way traffic light

Report 1: Present the schematic.

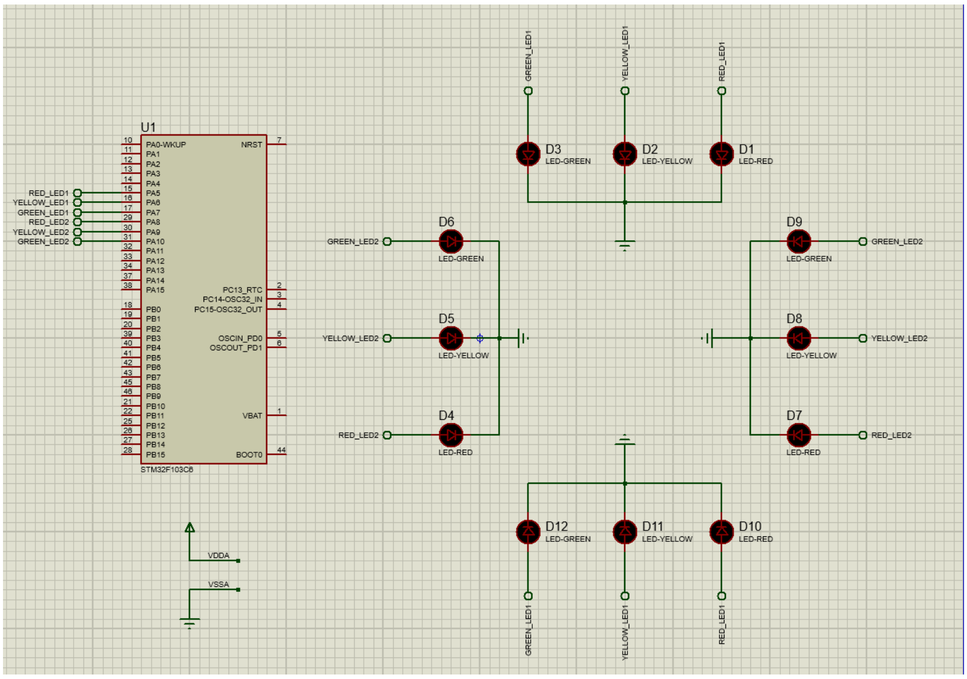


Figure 1.5: Schematic of Proteus simulation project

Report 2: Present the source code in while.

```
#define RED_DURATION 5
#define YELLOW_DURATION 2
#define GREEN_DURATION 3
```

```c
#define RED_STATE 0
#define GREEN_STATE 1
#define YELLOW_STATE 2
...
HAL_GPIO_WritePin(RED_LED1_GPIO_Port, RED_LED1_Pin, GPIO_PIN_SET);
HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port, YELLOW_LED1_Pin, GPIO_PIN_RESET);
HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port, GREEN_LED1_Pin, GPIO_PIN_RESET);
HAL_GPIO_WritePin(RED_LED2_GPIO_Port, RED_LED2_Pin, GPIO_PIN_RESET);
HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port, YELLOW_LED2_Pin, GPIO_PIN_RESET);
HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port, GREEN_LED2_Pin, GPIO_PIN_SET);
uint8_t count1 = RED_DURATION;
uint8_t state1 = RED_STATE;
uint8_t count2 = GREEN_DURATION;
uint8_t state2 = GREEN_STATE;
while (1)
{
  switch (state1)
  {
  case RED_STATE:
      HAL_GPIO_WritePin(RED_LED1_GPIO_Port, RED_LED1_Pin, GPIO_PIN_SET);
      if (count1 <= 0)
      {
          count1 = GREEN_DURATION;
          state1 = GREEN_STATE;
          HAL_GPIO_WritePin(RED_LED1_GPIO_Port, RED_LED1_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port, GREEN_LED1_Pin, GPIO_PIN_SET);
      }
      break;
  case GREEN_STATE:
      HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port, GREEN_LED1_Pin, GPIO_PIN_SET);
      if (count1 <= 0)
      {
          count1 = YELLOW_DURATION;
          state1 = YELLOW_STATE;
          HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port, GREEN_LED1_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port, YELLOW_LED1_Pin, GPIO_PIN_SET);
      }
```

```
        break;
    case YELLOW_STATE:
        HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port,
 YELLOW_LED1_Pin, GPIO_PIN_SET);
        if (count1 <= 0)
        {
            count1 = RED_DURATION;
            state1 = RED_STATE;
            HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port,
 YELLOW_LED1_Pin, GPIO_PIN_RESET);
            HAL_GPIO_WritePin(RED_LED1_GPIO_Port,
 RED_LED1_Pin, GPIO_PIN_SET);
        }
        break;
    }
    --count1;

    switch (state2)
    {
    case RED_STATE:
        HAL_GPIO_WritePin(RED_LED2_GPIO_Port, RED_LED2_Pin,
 GPIO_PIN_SET);
        if (count2 <= 0)
        {
            count2 = GREEN_DURATION;
            state2 = GREEN_STATE;
            HAL_GPIO_WritePin(RED_LED2_GPIO_Port,
 RED_LED2_Pin, GPIO_PIN_RESET);
            HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port,
 GREEN_LED2_Pin, GPIO_PIN_SET);
        }
        break;
    case GREEN_STATE:
        HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port,
 GREEN_LED2_Pin, GPIO_PIN_SET);
        if (count2 <= 0)
        {
            count2 = YELLOW_DURATION;
            state2 = YELLOW_STATE;
            HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port,
 GREEN_LED2_Pin, GPIO_PIN_RESET);
            HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port,
 YELLOW_LED2_Pin, GPIO_PIN_SET);
        }
        break;
    case YELLOW_STATE:
        HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port,
 YELLOW_LED2_Pin, GPIO_PIN_SET);
        if (count2 <= 0)
```

```
      {
          count2 = RED_DURATION;
          state2 = RED_STATE;
          HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port,
   YELLOW_LED2_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(RED_LED2_GPIO_Port,
   RED_LED2_Pin, GPIO_PIN_SET);
      }
      break;
  }
  --count2;
  HAL_Delay(1000);
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
}
```

Program 1.3: Source code for exercise 3

1.4 Exercise 4

Add **only one 7 led segment** to the schematic in Exercise 3. This component can be found in Proteus by the keyword **7SEG-COM-ANODE**. For this device, the common pin should be connected to the power supply and other pins are supposed to connected to PB0 to PB6. Therefore, to turn-on a segment in this 7SEG, the STM32 pin should be in logic 0 (0V).

Implement a function named **display7SEG(int num)**. The input for this function is from 0 to 9 and the outputs are listed as following:

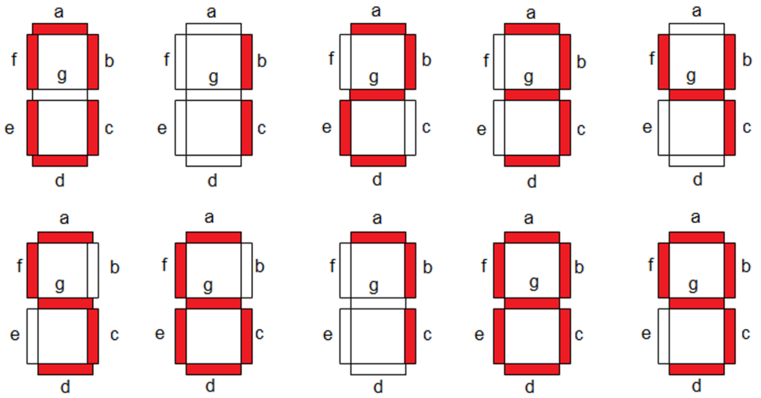


Figure 1.6: Display a number on 7 segment LED

This function is invoked in the while loop for testing as following:

```c
int counter = 0;
while (1){
    if(counter >= 10) counter = 0;
    display7SEG(counter++);
    HAL_Delay(1000);

}
```

Program 1.4: An example for your source code

Report 1: Present the schematic.

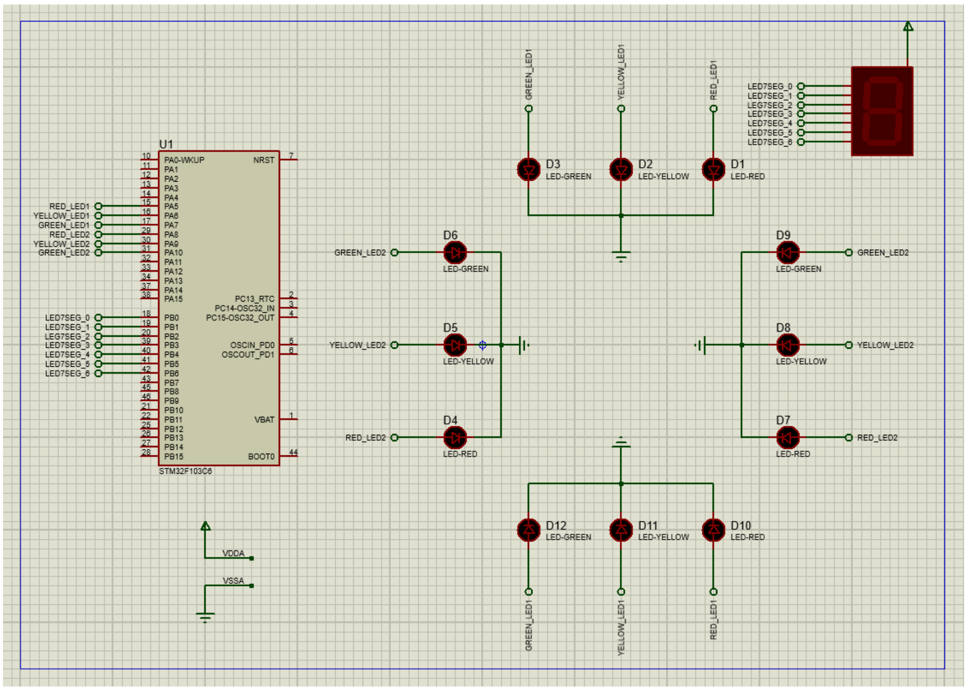


Figure 1.7: Schematic of Proteus simulation project

Report 2: Present the source code for display7SEG function.

```c
void display7SEG(int number)
{
  switch (number)
  {
  case 0:
    HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin,
   GPIO_PIN_RESET);
```

```c
      HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin, GPIO_PIN_SET);
      break;
    case 1:
      HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin, GPIO_PIN_SET);
      break;
    case 2:
      HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin, GPIO_PIN_RESET);
      break;
    case 3:
      HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin,
 GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin,
 GPIO_PIN_RESET);
    break;
  case 4:
    HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin,
 GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin,
 GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin,
 GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin,
 GPIO_PIN_RESET);
    break;
  case 5:
    HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin,
 GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin,
 GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin,
 GPIO_PIN_RESET);
    break;
  case 6:
    HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin,
 GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin,
 GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin,
 GPIO_PIN_RESET);
```

```
    HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin,
   GPIO_PIN_RESET);
    break;
  case 7:
    HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin,
   GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin,
   GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin,
   GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin,
   GPIO_PIN_SET);
    break;
  case 8:
    HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin,
   GPIO_PIN_RESET);
    break;
  case 9:
    HAL_GPIO_WritePin(LED7SEG_0_GPIO_Port, LED7SEG_0_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_1_GPIO_Port, LED7SEG_1_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_2_GPIO_Port, LED7SEG_2_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_3_GPIO_Port, LED7SEG_3_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_4_GPIO_Port, LED7SEG_4_Pin,
   GPIO_PIN_SET);
    HAL_GPIO_WritePin(LED7SEG_5_GPIO_Port, LED7SEG_5_Pin,
   GPIO_PIN_RESET);
    HAL_GPIO_WritePin(LED7SEG_6_GPIO_Port, LED7SEG_6_Pin,
   GPIO_PIN_RESET);
      break;
  }
}
```

Program 1.5: Source code of function `display7SEG()` for exercise 2

1.5 Exercise 5

Integrate the 7SEG-LED to the 4 way traffic light. In this case, the 7SEG-LED is used to display countdown value.

In this exercise, only source code is required to present. The function display7SEG in previous exercise can be re-used.

The source code of this exercise is almost the same as the source code of exercise 3, only an invocation of function `display7SEG(count1)` is added at line **54** to display the countdown value of the up and down traffic light.

```
1  #define RED_DURATION 5
2  #define YELLOW_DURATION 2
3  #define GREEN_DURATION 3
4  #define RED_STATE 0
5  #define GREEN_STATE 1
6  #define YELLOW_STATE 2
7  ...
8  HAL_GPIO_WritePin(RED_LED1_GPIO_Port, RED_LED1_Pin,
      GPIO_PIN_SET);
9  HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port, YELLOW_LED1_Pin,
      GPIO_PIN_RESET);
10 HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port, GREEN_LED1_Pin,
      GPIO_PIN_RESET);
11 HAL_GPIO_WritePin(RED_LED2_GPIO_Port, RED_LED2_Pin,
      GPIO_PIN_RESET);
12 HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port, YELLOW_LED2_Pin,
      GPIO_PIN_RESET);
13 HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port, GREEN_LED2_Pin,
      GPIO_PIN_SET);
14 uint8_t count1 = RED_DURATION;
15 uint8_t state1 = RED_STATE;
16 uint8_t count2 = GREEN_DURATION;
17 uint8_t state2 = GREEN_STATE;
18 while (1)
19 {
20   switch (state1)
21   {
22   case RED_STATE:
23       HAL_GPIO_WritePin(RED_LED1_GPIO_Port, RED_LED1_Pin,
      GPIO_PIN_SET);
24       if (count1 <= 0)
```

```
    {
        count1 = GREEN_DURATION;
        state1 = GREEN_STATE;
        HAL_GPIO_WritePin(RED_LED1_GPIO_Port,
 RED_LED1_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port,
 GREEN_LED1_Pin, GPIO_PIN_SET);
    }
    break;
case GREEN_STATE:
    HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port,
 GREEN_LED1_Pin, GPIO_PIN_SET);
    if (count1 <= 0)
    {
        count1 = YELLOW_DURATION;
        state1 = YELLOW_STATE;
        HAL_GPIO_WritePin(GREEN_LED1_GPIO_Port,
 GREEN_LED1_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port,
 YELLOW_LED1_Pin, GPIO_PIN_SET);
    }
    break;
case YELLOW_STATE:
    HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port,
 YELLOW_LED1_Pin, GPIO_PIN_SET);
    if (count1 <= 0)
    {
        count1 = RED_DURATION;
        state1 = RED_STATE;
        HAL_GPIO_WritePin(YELLOW_LED1_GPIO_Port,
 YELLOW_LED1_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(RED_LED1_GPIO_Port,
 RED_LED1_Pin, GPIO_PIN_SET);
    }
    break;
}
display7SEG(count1);
--count1;

switch (state2)
{
case RED_STATE:
    HAL_GPIO_WritePin(RED_LED2_GPIO_Port, RED_LED2_Pin,
 GPIO_PIN_SET);
    if (count2 <= 0)
    {
        count2 = GREEN_DURATION;
        state2 = GREEN_STATE;
        HAL_GPIO_WritePin(RED_LED2_GPIO_Port,
   RED_LED2_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port,
   GREEN_LED2_Pin, GPIO_PIN_SET);
      }
      break;
  case GREEN_STATE:
      HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port,
   GREEN_LED2_Pin, GPIO_PIN_SET);
      if (count2 <= 0)
      {
          count2 = YELLOW_DURATION;
          state2 = YELLOW_STATE;
          HAL_GPIO_WritePin(GREEN_LED2_GPIO_Port,
   GREEN_LED2_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port,
   YELLOW_LED2_Pin, GPIO_PIN_SET);
      }
      break;
  case YELLOW_STATE:
      HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port,
   YELLOW_LED2_Pin, GPIO_PIN_SET);
      if (count2 <= 0)
      {
          count2 = RED_DURATION;
          state2 = RED_STATE;
          HAL_GPIO_WritePin(YELLOW_LED2_GPIO_Port,
   YELLOW_LED2_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(RED_LED2_GPIO_Port,
   RED_LED2_Pin, GPIO_PIN_SET);
      }
      break;
  }
  --count2;
  HAL_Delay(1000);
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
}
```

Program 1.6: Source code for exercise 5

1.6 Exercise 6

In this exercise, a new Proteus schematic is designed to simulate an analog clock, with 12 different number. The connections for 12 LEDs are supposed from PA4 to PA15 of the STM32. The arrangement of 12 LEDs is depicted as follows.

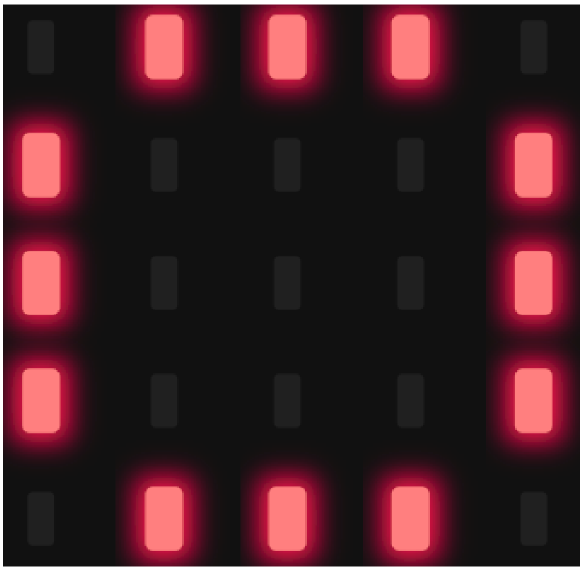


Figure 1.8: 12 LEDs for an analog clock

Report 1: Present the schematic.

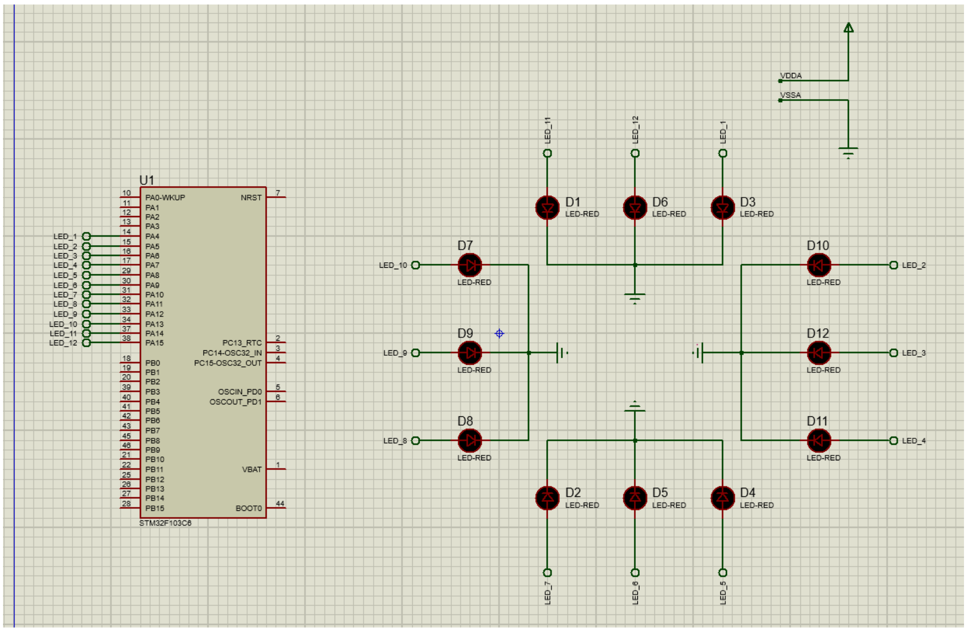


Figure 1.9: Schematic of Proteus simulation project

Report 2: Implement a simple program to test the connection of every single LED. This testing program should turn every LED in a sequence.

```
uint8_t count = 0;
while (1)
{
  ++count;
  switch (count)
  {
  case 1:
      HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin,
   GPIO_PIN_SET);
      break;
  case 2:
      HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin,
   GPIO_PIN_SET);
      break;
  case 3:
      HAL_GPIO_WritePin(LED3_GPIO_Port, LED3_Pin,
   GPIO_PIN_SET);
      break;
  case 4:
      HAL_GPIO_WritePin(LED4_GPIO_Port, LED4_Pin,
   GPIO_PIN_SET);
      break;
  case 5:
      HAL_GPIO_WritePin(LED5_GPIO_Port, LED5_Pin,
   GPIO_PIN_SET);
      break;
  case 6:
      HAL_GPIO_WritePin(LED6_GPIO_Port, LED6_Pin,
   GPIO_PIN_SET);
      break;
  case 7:
      HAL_GPIO_WritePin(LED7_GPIO_Port, LED7_Pin,
   GPIO_PIN_SET);
      break;
  case 8:
      HAL_GPIO_WritePin(LED8_GPIO_Port, LED8_Pin,
   GPIO_PIN_SET);
      break;
  case 9:
      HAL_GPIO_WritePin(LED9_GPIO_Port, LED9_Pin,
   GPIO_PIN_SET);
      break;
  case 10:
      HAL_GPIO_WritePin(LED10_GPIO_Port, LED10_Pin,
   GPIO_PIN_SET);
      break;
  case 11:
      HAL_GPIO_WritePin(LED11_GPIO_Port, LED11_Pin,
   GPIO_PIN_SET);
      break;
  case 12:
      HAL_GPIO_WritePin(LED12_GPIO_Port, LED12_Pin,
   GPIO_PIN_SET);
      break;
  default:
      count = 13;
      break;
  }
  HAL_Delay(1000);
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
}
```

Program 1.7: Source code for testing the connection of every single LED

1.7 Exercise 7

Implement a function named **clearAllClock()** to turn off all 12 LEDs. Present the source code of this function.

```
void clearAllClock()
{
  HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED3_GPIO_Port, LED3_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED4_GPIO_Port, LED4_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED5_GPIO_Port, LED5_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED6_GPIO_Port, LED6_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED7_GPIO_Port, LED7_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED8_GPIO_Port, LED8_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED9_GPIO_Port, LED9_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED10_GPIO_Port, LED10_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED11_GPIO_Port, LED11_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED12_GPIO_Port, LED12_Pin,
   GPIO_PIN_RESET);
}
```

Program 1.8: Function Implementation

1.8 Exercise 8

Implement a function named **setNumberOnClock(int num)**. The input for this function is from **0 to 11** and an appropriate LED is turn on. Present the source code of this function.

```c
void setNumberOnClock(int num)
{
  switch (num)
  {
  case 1:
    HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin,
   GPIO_PIN_SET);
    break;
  case 2:
    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin,
   GPIO_PIN_SET);
    break;
  case 3:
    HAL_GPIO_WritePin(LED3_GPIO_Port, LED3_Pin,
   GPIO_PIN_SET);
    break;
  case 4:
    HAL_GPIO_WritePin(LED4_GPIO_Port, LED4_Pin,
   GPIO_PIN_SET);
    break;
  case 5:
    HAL_GPIO_WritePin(LED5_GPIO_Port, LED5_Pin,
   GPIO_PIN_SET);
    break;
  case 6:
    HAL_GPIO_WritePin(LED6_GPIO_Port, LED6_Pin,
   GPIO_PIN_SET);
    break;
  case 7:
    HAL_GPIO_WritePin(LED7_GPIO_Port, LED7_Pin,
   GPIO_PIN_SET);
    break;
  case 8:
    HAL_GPIO_WritePin(LED8_GPIO_Port, LED8_Pin,
   GPIO_PIN_SET);
    break;
  case 9:
    HAL_GPIO_WritePin(LED9_GPIO_Port, LED9_Pin,
   GPIO_PIN_SET);
    break;
  case 10:
    HAL_GPIO_WritePin(LED10_GPIO_Port, LED10_Pin,
   GPIO_PIN_SET);
    break;
  case 11:
    HAL_GPIO_WritePin(LED11_GPIO_Port, LED11_Pin,
   GPIO_PIN_SET);
    break;
  case 0:
    HAL_GPIO_WritePin(LED12_GPIO_Port, LED12_Pin,
   GPIO_PIN_SET);
    break;
  }
}
```

Program 1.9: Function Implementation

1.9 Exercise 9

Implement a function named **clearNumberOnClock(int num)**. The input for this function is from **0 to 11** and an appropriate LED is turn off.

```
void clearNumberOnClock(int num)
{
  switch (num)
  {
  case 1:
    HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin,
   GPIO_PIN_RESET);
    break;
  case 2:
    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin,
   GPIO_PIN_RESET);
    break;
  case 3:
    HAL_GPIO_WritePin(LED3_GPIO_Port, LED3_Pin,
   GPIO_PIN_RESET);
    break;
  case 4:
    HAL_GPIO_WritePin(LED4_GPIO_Port, LED4_Pin,
   GPIO_PIN_RESET);
    break;
  case 5:
    HAL_GPIO_WritePin(LED5_GPIO_Port, LED5_Pin,
   GPIO_PIN_RESET);
    break;
  case 6:
    HAL_GPIO_WritePin(LED6_GPIO_Port, LED6_Pin,
   GPIO_PIN_RESET);
    break;
  case 7:
    HAL_GPIO_WritePin(LED7_GPIO_Port, LED7_Pin,
   GPIO_PIN_RESET);
    break;
  case 8:
    HAL_GPIO_WritePin(LED8_GPIO_Port, LED8_Pin,
   GPIO_PIN_RESET);
    break;
  case 9:
```

```c
    HAL_GPIO_WritePin(LED9_GPIO_Port, LED9_Pin,
   GPIO_PIN_RESET);
    break;
  case 10:
    HAL_GPIO_WritePin(LED10_GPIO_Port, LED10_Pin,
   GPIO_PIN_RESET);
    break;
  case 11:
    HAL_GPIO_WritePin(LED11_GPIO_Port, LED11_Pin,
   GPIO_PIN_RESET);
    break;
  case 0:
    HAL_GPIO_WritePin(LED12_GPIO_Port, LED12_Pin,
   GPIO_PIN_RESET);
    break;
  }
}
```

Program 1.10: Function Implementation

1.10 Exercise 10

Integrate the whole system and use 12 LEDs to display a clock. At a given time, there are only 3 LEDs are turn on for hour, minute and second information.

```c
#define SECONDS_PER_TICK 5
#define SECOND_CYCLES_PER_MINUTE 5
#define NUM_OF_TICKS 12
#define SECOND_DURATION 1
...
uint8_t counter_second_led = 0;
uint8_t counter_minute_led = 0;
uint8_t counter_hour_led = 0;
uint8_t counter_second_cycles = SECOND_CYCLES_PER_MINUTE;
uint8_t counter_second = SECONDS_PER_TICK;
uint8_t timer_counter = SECOND_DURATION;
clearAllClock();
setNumberOnClock(counter_second_led);
setNumberOnClock(counter_minute_led);
setNumberOnClock(counter_hour_led);
while (1)
{
  if (timer_counter <= 0)
  {
      timer_counter = SECOND_DURATION;
      --counter_second;
      if (counter_second <= 0)
      {
          counter_second = SECONDS_PER_TICK;
          clearNumberOnClock(counter_second_led);
```

```
     clearNumberOnClock(counter_minute_led);
     clearNumberOnClock(counter_hour_led);

     ++counter_second_led;
     if (counter_second_led == NUM_OF_TICKS)
     {
       counter_second_led = 0;
       --counter_second_cycles;
       if (counter_second_cycles <= 0)
       {
         counter_second_cycles = SECOND_CYCLES_PER_MINUTE;
         ++counter_minute_led;
       }
     }

     if (counter_minute_led == NUM_OF_TICKS)
     {
       counter_minute_led = 0;
       ++counter_hour_led;
     }

     if (counter_hour_led == NUM_OF_TICKS)
     {
       counter_hour_led = 0;
     }


     setNumberOnClock(counter_second_led);
     setNumberOnClock(counter_minute_led);
     setNumberOnClock(counter_hour_led);
   }
 }
 --timer_counter;
 HAL_Delay(1000);
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
}
```

Program 1.11: Source code for exercise 10