

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



Microprocessors-Microcontrollers

COURSE ID: CO3010 - HK251 CLASS: CC02

Lab 1 Report

Lecturer: Phan Văn Sỹ
Students: Nguyễn Thế Khang - 2352490

HỒ CHÍ MINH CITY, September 2025



Contents

1	Exercise and Report	2
1.1	Exercise 1	2
1.1.1	Schematic	2
1.1.2	Source Code	2
1.1.3	GitHub Repository	3
1.2	Exercise 2	4
1.2.1	Schematic	4
1.2.2	Source Code	4
1.2.3	GitHub Repository	4
1.3	Exercise 3	6
1.3.1	Schematic	6
1.3.2	Source Code	6
1.3.3	GitHub Repository	7
1.4	Exercise 4	8
1.4.1	Schematic	8
1.4.2	Source Code	9
1.4.3	GitHub Repository	11
1.5	Exercise 5	12
1.5.1	Schematic	12
1.5.2	Source Code	12
1.5.3	GitHub Repository	13
1.6	Exercise 6	14
1.6.1	Schematic	14
1.6.2	Source Code	15
1.6.3	GitHub Repository	15
1.7	Exercise 7	16
1.7.1	Source Code	16
1.7.2	GitHub Repository	16
1.8	Exercise 8	17
1.8.1	Source Code	17
1.8.2	GitHub Repository	17
1.9	Exercise 9	18
1.9.1	Source Code	18
1.9.2	GitHub Repository	18
1.10	Exercise 10	19
1.10.1	Schematic	19
1.10.2	Source Code	19
1.10.3	GitHub Repository	20

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.

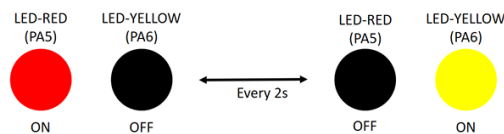
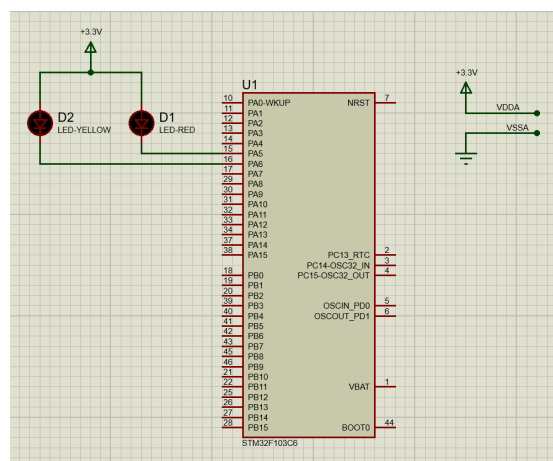


Figure 1.23: State transitions for 2 LEDs

1.1.1 Schematic



1.1.2 Source Code

```
1 while (1)
2 {
3     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET);
4     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET);
5     HAL_Delay(2000);
6
7     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET);
8     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET);
9     HAL_Delay(2000);
10 }
```



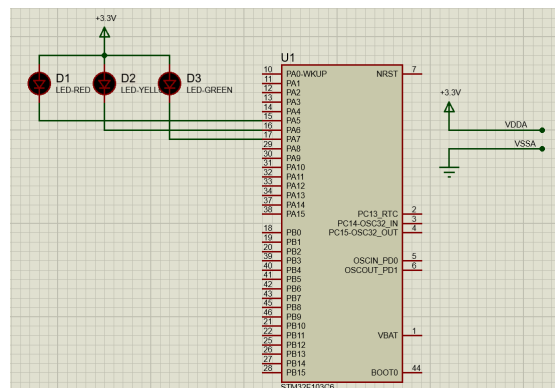
1.1.3 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/tree/main/Schematics/Lab%201/Exercise%201>
- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%201>

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.

1.2.1 Schematic



1.2.2 Source Code

```
1 while (1)
2 {
3     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET);
4     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET);
5     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET);
6     HAL_Delay(5000);
7
8     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET);
9     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET);
10    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET);
11    HAL_Delay(2000);
12
13    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET);
14    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET);
15    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET);
16    HAL_Delay(3000);
17 }
```

1.2.3 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/tree/main/Schematics/Lab%201/Exercise%202>



- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%202>

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 below.

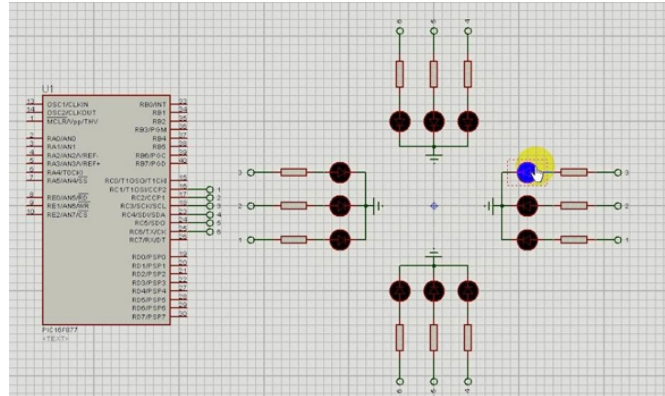
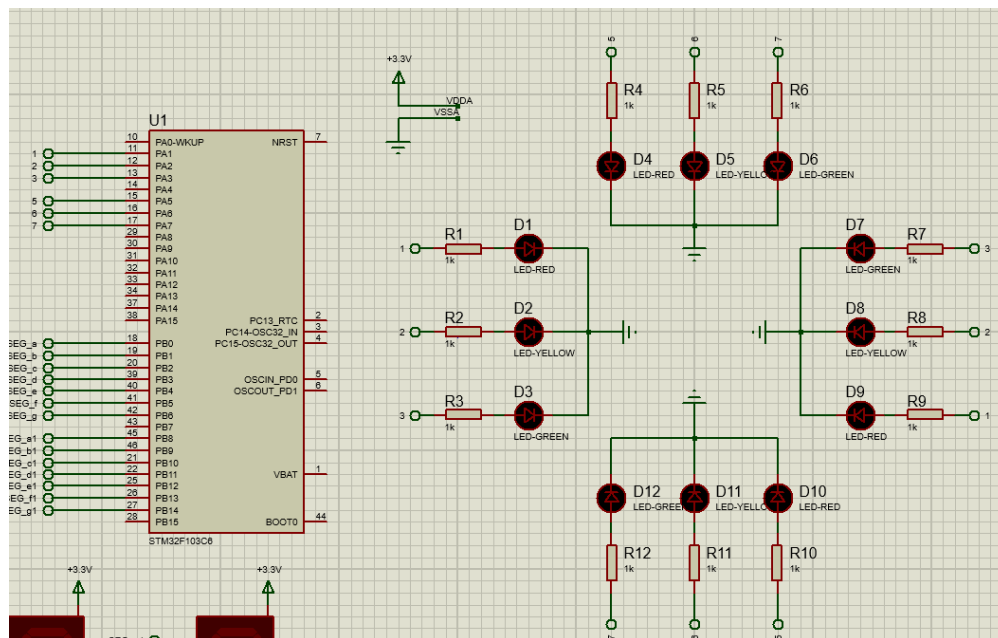


Figure 1.24: Reference design for a 4 way traffic light

1.3.1 Schematic



1.3.2 Source Code



```
1 void traffic_light(void)
2 {
3     // Direction 1
4     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, RESET); // Red off
5     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET); // Red on
6     for (int count = 5; count >= 0; count--) {
7         if (count > 2) {
8             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, SET); // Green on
9             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET); // Yellow off
10        } else {
11            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET); // Green off
12            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, SET); // Yellow on
13        }
14        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
15        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
16        HAL_Delay(1000);
17    }
18
19    // Direction 2
20    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, SET); // Red on
21    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET); // Red off
22    for (int count = 5; count >= 0; count--) {
23        if (count > 2) {
24            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET); // Green on
25            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
26        } else {
27            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
28            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET); // Yellow on
29        }
30        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET);
31        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET);
32        HAL_Delay(1000);
33    }
34 }
```

1.3.3 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/tree/main/Schematics/Lab%201/Exercise%203%20to%205>
- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%203%20to%205/Core/Src/Exercise%203.h>

1.4 Exercise 4

Add only one 7 led segment to the schematic in Exercise3. 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 be 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 out puts are listed as following:

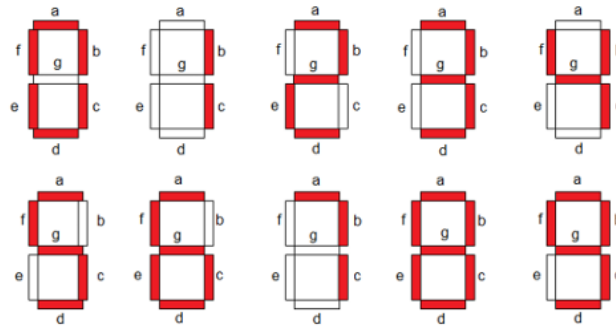
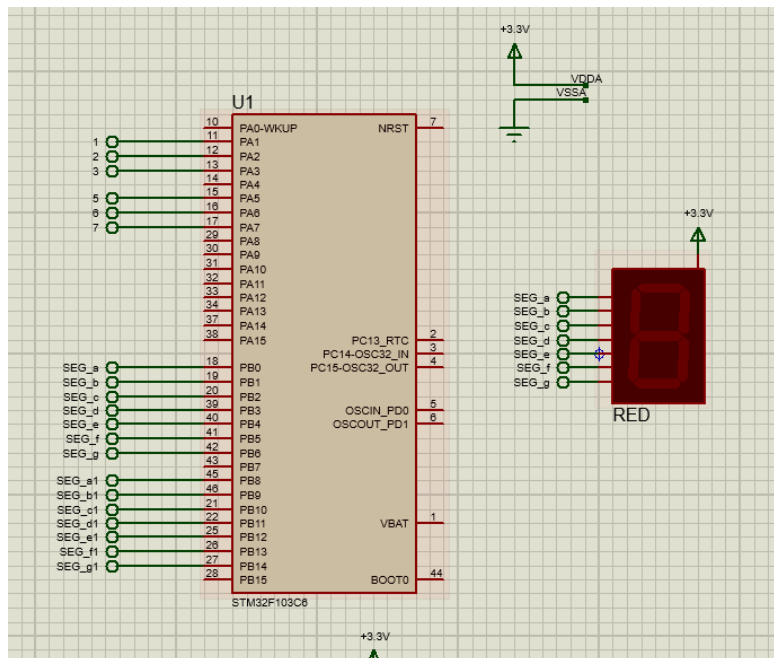


Figure 1.25: Display a number on 7 segment LED

1.4.1 Schematic



1.4.2 Source Code

```
1 void display7SEG(int num) {
2     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, SET); // a
3     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, SET); // b
4     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, SET); // c
5     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
6     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
7     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
8     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g
9
10    // Switch case for each digit
11    switch (num) {
12        case 0:
13            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
14            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
15            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
16            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
17            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
18            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
19            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g
20            break;
21        case 1:
22            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
23            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
24            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, SET); // a
25            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
26            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
27            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
28            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g
29            break;
30        case 2:
31            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
32            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
33            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
34            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
35            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
36            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, SET); // c
37            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
38            break;
39        case 3:
40            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
41            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
42            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
43            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
44            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
45            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
46            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
47            break;
48        case 4:
49            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
50            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
51            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
```



```
52     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
53     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, SET); // a
54     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
55     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
56     break;
57 case 5:
58     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
59     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
60     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
61     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
62     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
63     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, SET); // b
64     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
65     break;
66 case 6:
67     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
68     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
69     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
70     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
71     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
72     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
73     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, SET); // b
74     break;
75 case 7:
76     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
77     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
78     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
79     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
80     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
81     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
82     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g
83     break;
84 case 8:
85     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
86     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
87     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
88     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
89     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
90     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
91     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
92     break;
93 case 9:
94     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
95     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
96     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
97     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
98     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
99     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
100    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
101    break;
102 }
103 }
```



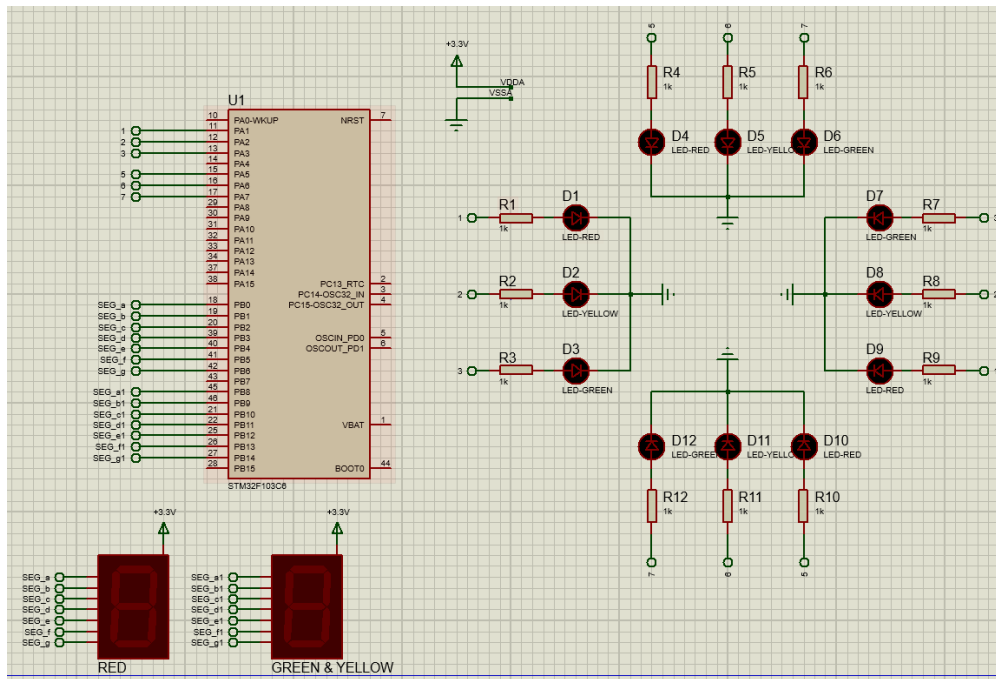
1.4.3 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/tree/main/Schematics/Lab%201/Exercise%203%20to%205>
- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%203%20to%205/Core/Src/Exercise%204.h>

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.

1.5.1 Schematic



1.5.2 Source Code

```
1 void completeTraffic_light(void)
2 {
3     // Direction 1
4     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, RESET); // Red off
5     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET); // Red on
6     for (int count = 5; count >= 0; count--) {
7         if (count > 2) {
8             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, SET); // Green on
9             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET); // Yellow off
10            display7SEGGreenYellow(count - 2);
11        } else {
12            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET); // Green off
13            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, SET); // Yellow on
14            display7SEGGreenYellow(count);
15        }
16    }
17 }
```



```
16     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
17     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
18     display7SEG(count);
19     HAL_Delay(1000);
20 }
21
22 // Direction 2
23 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, SET); // Red on
24 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET); // Red off
25     for (int count = 5; count >= 0; count--) {
26         if (count > 2) {
27             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET); // Green on
28             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
29             display7SEGGreenYellow(count - 2);
30         } else {
31             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
32             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET); // Yellow on
33             display7SEGGreenYellow(count);
34         }
35         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET);
36         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET);
37         display7SEG(count);
38         HAL_Delay(1000);
39     }
40 }
```

1.5.3 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/tree/main/Schematics/Lab%201/Exercise%203%20to%205>
- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%203%20to%205/Core/Src/Exercise%205.h>

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.

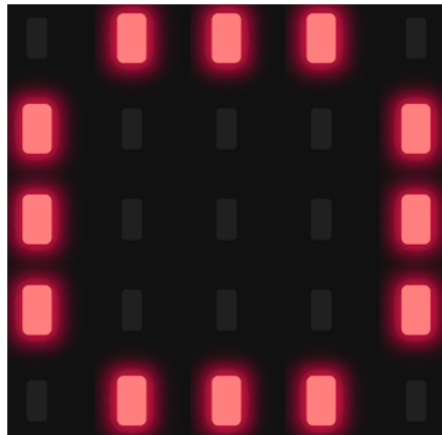
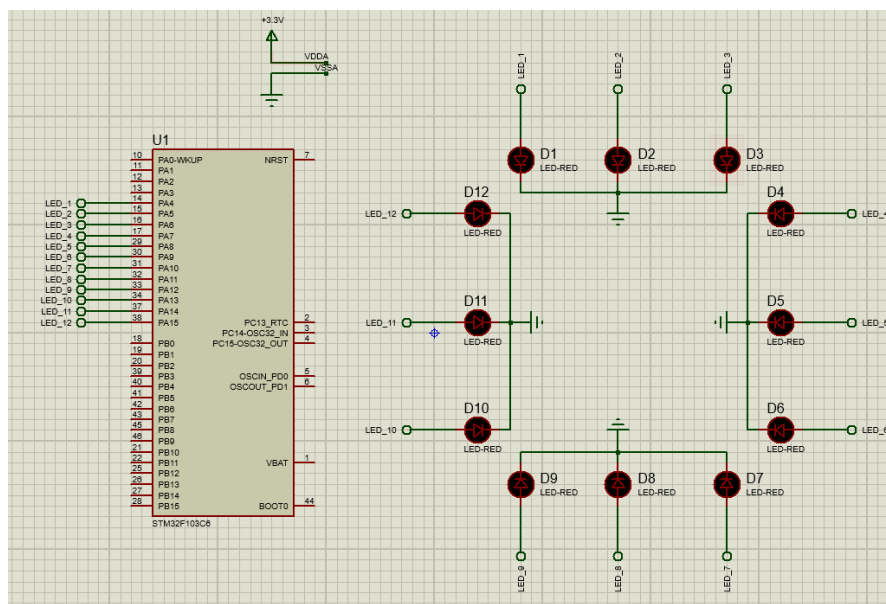


Figure 1.26: 12 LEDs for an analog clock

1.6.1 Schematic



1.6.2 Source Code

```
1 uint16_t led_pins[12] = {
2     LED_2_Pin, // Hour 1
3     LED_3_Pin, // Hour 2
4     LED_4_Pin, // Hour 3
5     LED_5_Pin, // Hour 4
6     LED_6_Pin, // Hour 5
7     LED_7_Pin, // Hour 6
8     LED_8_Pin, // Hour 7
9     LED_9_Pin, // Hour 8
10    LED_10_Pin, // Hour 9
11    LED_11_Pin, // Hour 10
12    LED_12_Pin, // Hour 11
13    LED_1_Pin  // Hour 12
14};
15
16 int count = 0;
17 void analog_clock(void)
18 {
19     while(1)
20     {
21         if (count < 12) {
22             HAL_GPIO_WritePin(GPIOA, led_pins[count], SET);
23         } else {
24             // All on for full circle flash
25             HAL_GPIO_WritePin(GPIOA, LED_1_Pin|LED_2_Pin|LED_3_Pin|LED_4_Pin
26                                 |LED_5_Pin|LED_6_Pin|LED_7_Pin|LED_8_Pin
27                                 |LED_9_Pin|LED_10_Pin|LED_11_Pin|LED_12_Pin, SET);
28         }
29
30         HAL_Delay(1000);
31         if (count == 12) {
32             count = 0;
33         } else {
34             count++;
35         }
36     }
37 }
```

1.6.3 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/tree/main/Schematics/Lab%201/Exercise%206%20to%2010>
- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%206%20to%2010/Core/Src/Exercise%206.h>

1.7 Exercise 7

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

```
1 void clearAllClock(){  
2     //TODO  
3 }
```

Program 1.5: Function Implementation

1.7.1 Source Code

```
1 void clearAllClock(void)  
2 {  
3     HAL_GPIO_WritePin(GPIOA, LED_1_Pin|LED_2_Pin|LED_3_Pin|LED_4_Pin  
4         |LED_5_Pin|LED_6_Pin|LED_7_Pin|LED_8_Pin  
5         |LED_9_Pin|LED_10_Pin|LED_11_Pin|LED_12_Pin, RESET);  
6 }
```

1.7.2 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%206%20to%2010/Core/Src/Exercise%207.h>



1.8 Exercise 8

Implement a function named `setNumberOnClock(intnum)`. The input for this function is from 0 to 11 and an appropriate LED is turned on. Present the source code of this function.

1.8.1 Source Code

```
1 void setNumberOnClock(int num)
2 {
3     if (num >= 0 && num < 12)
4     {
5         HAL_GPIO_WritePin(GPIOA, led_pins[num], SET);
6     }
7 }
```

1.8.2 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%206%20to%2010/Core/Src/Exercise%208.h>



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.

1.9.1 Source Code

```
1 void clearNumberOnClock(int num)
2 {
3     if (num >= 0 && num < 12)
4     {
5         HAL_GPIO_WritePin(GPIOA, led_pins[num], RESET);
6     }
7 }
```

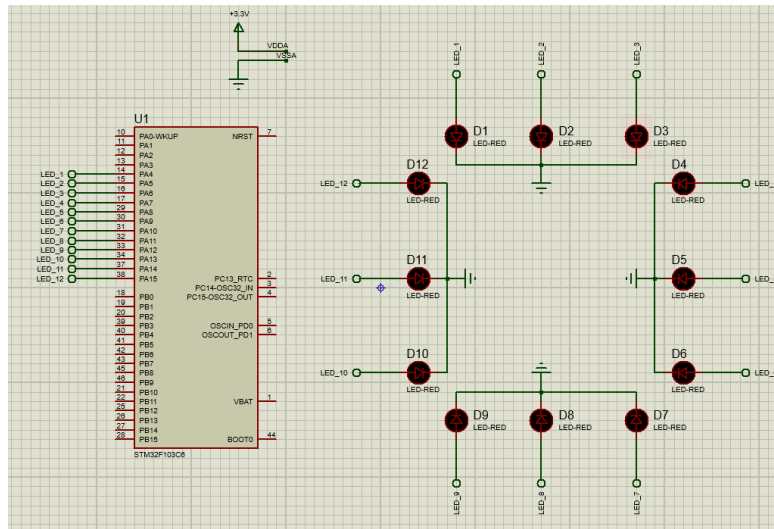
1.9.2 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%206%20to%2010/Core/Src/Exercise%209.h>

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.

1.10.1 Schematic



1.10.2 Source Code

```
1 void completeAnalog_clock(void)
2 {
3     static uint32_t total_seconds = 0;
4     int hour, minute, second;
5
6     while (1)
7     {
8         second = total_seconds % 60;
9         minute = (total_seconds / 60) % 60;
10        hour = (total_seconds / 3600) % 24;
11
12        clearAllClock();
13        setNumberOnClock(hour % 12);
14        setNumberOnClock((minute / 5) % 12);
15        setNumberOnClock((second / 5) % 12);
16
17        total_seconds++;
18        HAL_Delay(1000);
19    }
20 }
```



1.10.3 GitHub Repository

- <https://github.com/iiTatoman/MCU-MPU/tree/main/Schematics/Lab%201/Exercise%206%20to%2010>
- <https://github.com/iiTatoman/MCU-MPU/blob/main/Source/Lab%201/Exercise%206%20to%2010/Core/Src/Exercise%2010.h>