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.

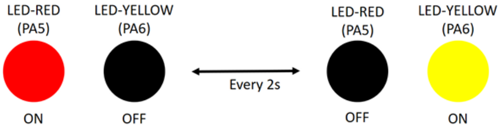
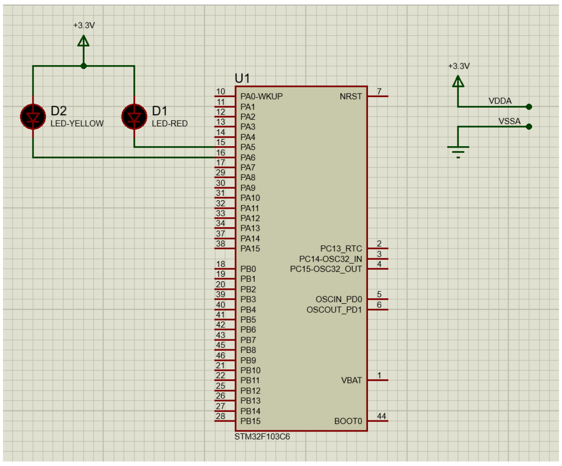


Figure 1.23: State transitions for 2 LEDs

1.1.1 Schematic



1.1.2 Source Code

```c
while (1)
{
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET);
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET);
    HAL_Delay(2000);

    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET);
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET);
    HAL_Delay(2000);
}
```

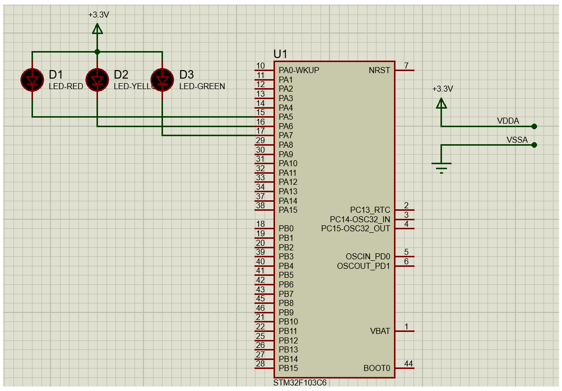
1.1.3 GitHub Repository

- 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

```c
while (1)
{
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET);
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET);
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET);
   HAL_Delay(5000);

   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET);
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET);
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET);
   HAL_Delay(2000);

   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET);
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET);
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET);
   HAL_Delay(3000);
}
```

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 bellow.

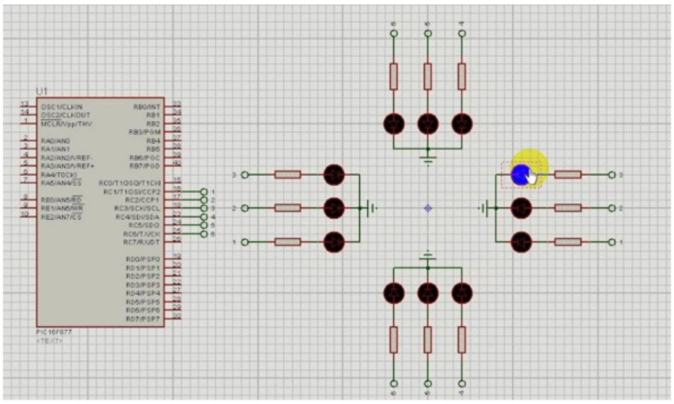
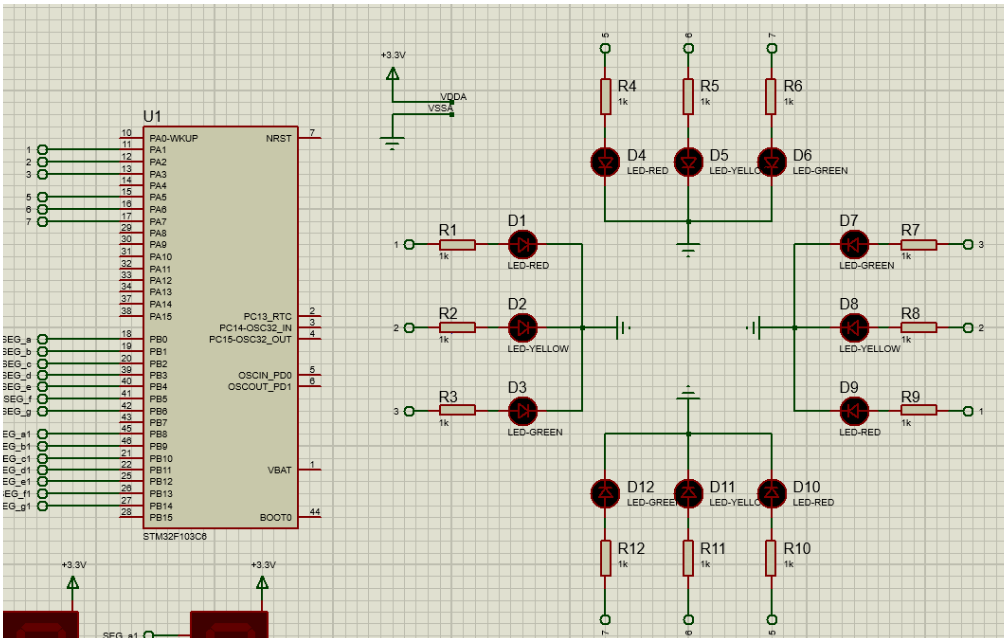


Figure 1.24: Reference design for a 4 way traffic light

1.3.1 Schematic



1.3.2 Source Code

```
void traffic_light(void)
{
   // Direction 1
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, RESET); // Red off
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET); // Red on
   for (int count = 5; count >= 0; count--) {
      if (count > 2) {
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, SET); // Green on
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET); // Yellow off
      } else {
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET); // Green off
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, SET); // Yellow on
      }
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
   HAL_Delay(1000);
   }

   // Direction 2
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, SET); // Red on
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET); // Red off
   for (int count = 5; count >= 0; count--) {
      if (count > 2) {
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET); // Green on
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
      } else {
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET); // Yellow on
      }
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET);
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET);
   HAL_Delay(1000);
   }
}
```

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 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:

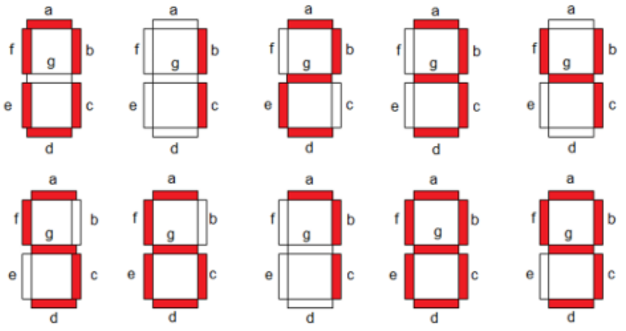
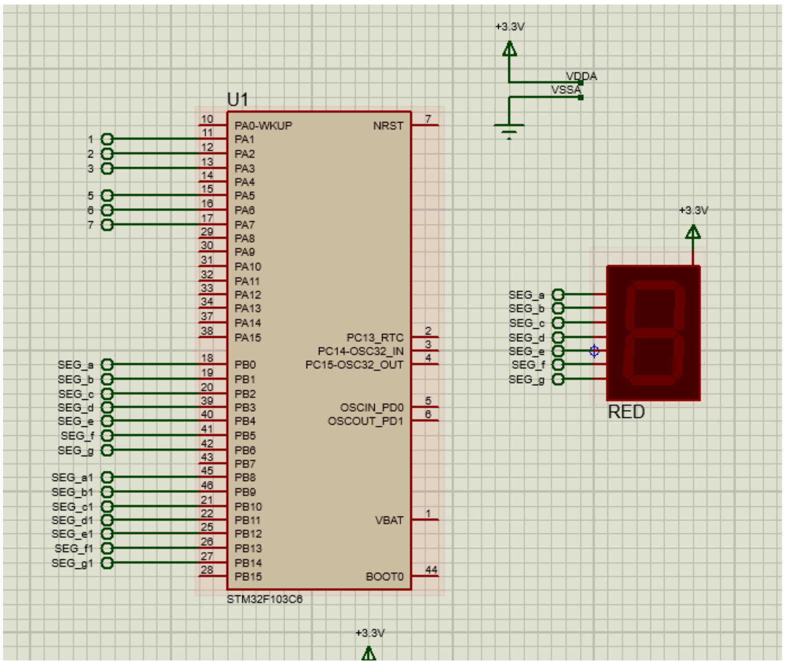


Figure 1.25: Display a number on 7 segment LED

1.4.1 Schematic



1.4.2 Source Code

```
void display7SEG(int num) {
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, SET); // a
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, SET); // b
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, SET); // c
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g

    // Switch case for each digit
    switch (num) {
        case 0:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g
            break;
        case 1:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, SET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g
            break;
        case 2:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, SET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
            break;
        case 3:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
            break;
        case 4:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, SET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
            break;
        case 5:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, SET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
            break;
        case 6:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, SET); // b
            break;
        case 7:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, SET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, SET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, SET); // g
            break;
        case 8:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, RESET); // e
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
            break;
        case 9:
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET); // a
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, RESET); // b
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, RESET); // c
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, RESET); // d
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, RESET); // f
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, RESET); // g
            HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, SET); // e
            break;
    }
}
```

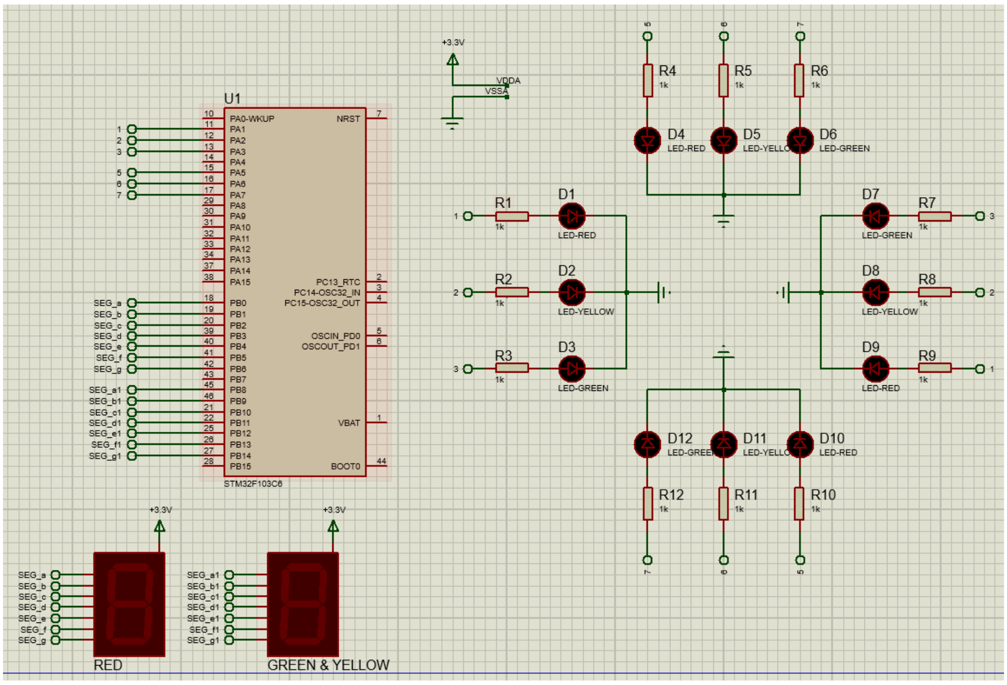
1.4.3 GitHub Repository

- 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

```
void completeTraffic_light(void)
{
   // Direction 1
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, RESET); // Red off
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, SET); // Red on
      for (int count = 5; count >= 0; count--) {
         if (count > 2) {
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, SET); // Green on
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET); // Yellow off
            display7SEGGreenYellow(count - 2);
         } else {
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET); // Green off
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, SET); // Yellow on
            display7SEGGreenYellow(count);
         }
```

```
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
            display7SEG(count);
            HAL_Delay(1000);
       }

   // Direction 2
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, SET); // Red on
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, RESET); // Red off
      for (int count = 5; count >= 0; count--) {
         if (count > 2) {
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, SET); // Green on
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, RESET); // Yellow off
            display7SEGGreenYellow(count - 2);
         } else {
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, RESET); // Green off
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, SET); // Yellow on
            display7SEGGreenYellow(count);
         }
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, RESET);
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, RESET);
            display7SEG(count);
            HAL_Delay(1000);
      }
}
```

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.

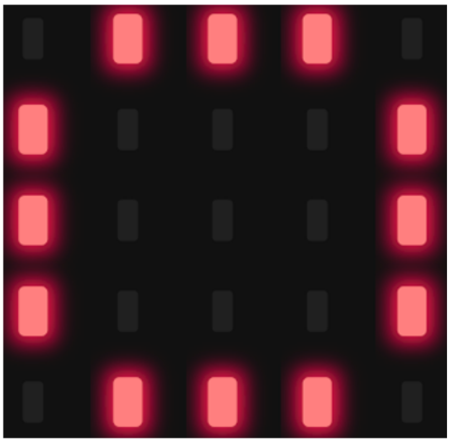
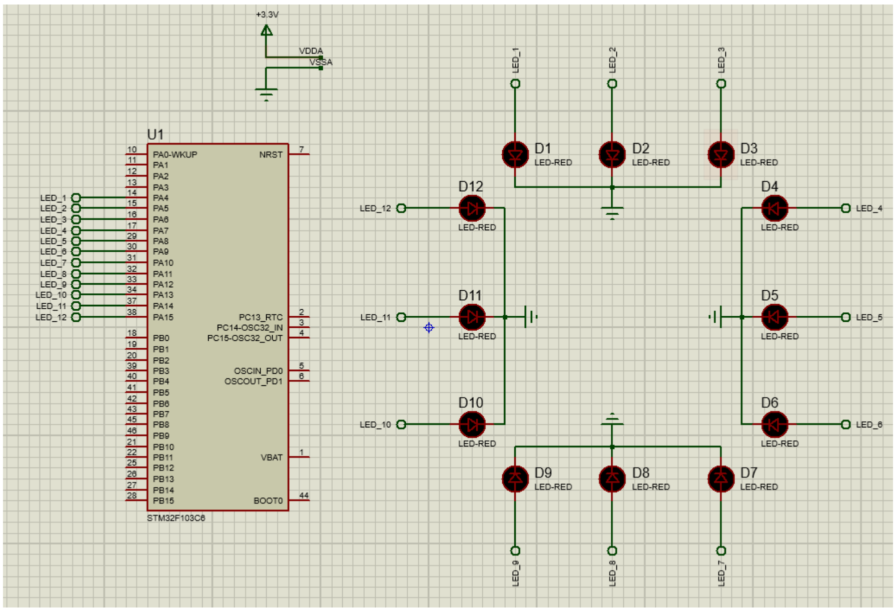


Figure 1.26: 12 LEDs for an analog clock

1.6.1 Schematic



1.6.2 Source Code

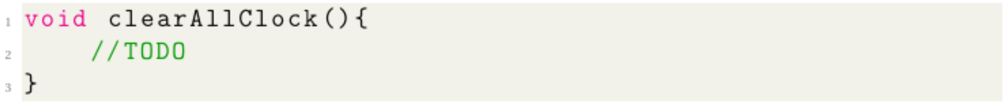
```c
uint16_t led_pins[12] = {
    LED_2_Pin, // Hour 1
    LED_3_Pin, // Hour 2
    LED_4_Pin, // Hour 3
    LED_5_Pin, // Hour 4
    LED_6_Pin, // Hour 5
    LED_7_Pin, // Hour 6
    LED_8_Pin, // Hour 7
    LED_9_Pin, // Hour 8
    LED_10_Pin, // Hour 9
    LED_11_Pin, // Hour 10
    LED_12_Pin, // Hour 11
    LED_1_Pin  // Hour 12
};

int count = 0;
void analog_clock(void)
{
   while(1)
   {
      if (count < 12) {
         HAL_GPIO_WritePin(GPIOA, led_pins[count], SET);
      } else {
         // All on for full circle flash
         HAL_GPIO_WritePin(GPIOA, LED_1_Pin|LED_2_Pin|LED_3_Pin|LED_4_Pin
                                |LED_5_Pin|LED_6_Pin|LED_7_Pin|LED_8_Pin
                                |LED_9_Pin|LED_10_Pin|LED_11_Pin|LED_12_Pin, SET);
      }

      HAL_Delay(1000);
      if (count == 12) {
         count = 0;
      } else {
         count++;
      }
   }
}
```

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.



```
void clearAllClock(){
    //TODO
}
```

Program 1.5: Function Implementation

1.7.1 Source Code

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

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 tun on. Present the source code of this function.

1.8.1 Source Code

```
void setNumberOnClock(int num)
{
   if (num >= 0 && num < 12)
       {
           HAL_GPIO_WritePin(GPIOA, led_pins[num], SET);
       }
}
```

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

```
void clearNumberOnClock(int num)
{
   if (num >= 0 && num < 12)
       {
           HAL_GPIO_WritePin(GPIOA, led_pins[num], RESET);
       }
}
```

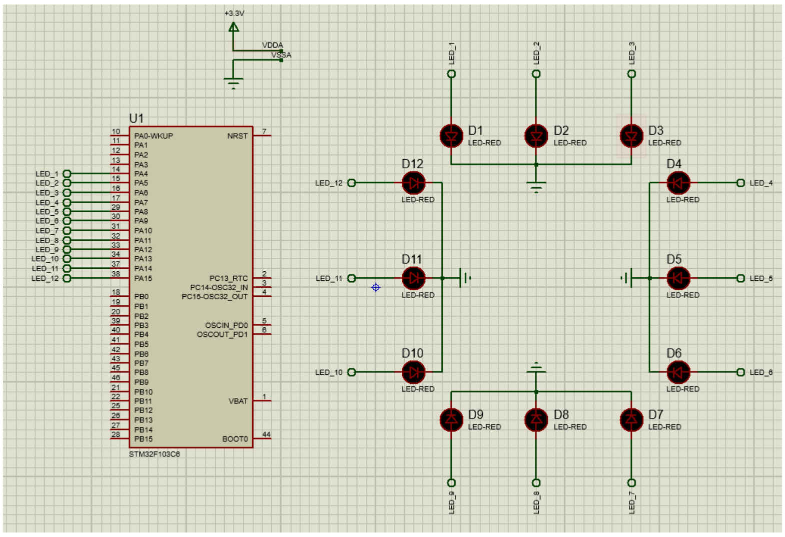
1.9.2 GitHub Repository

- 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

```
void completeAnalog_clock(void)
{
    static uint32_t total_seconds = 0;
    int hour, minute, second;

    while (1)
    {
        second = total_seconds % 60;
        minute = (total_seconds / 60) % 60;
        hour = (total_seconds / 3600) % 24;

        clearAllClock();
        setNumberOnClock(hour % 12);
        setNumberOnClock((minute / 5) % 12);
        setNumberOnClock((second / 5) % 12);

        total_seconds++;
        HAL_Delay(1000);
    }
}
```

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