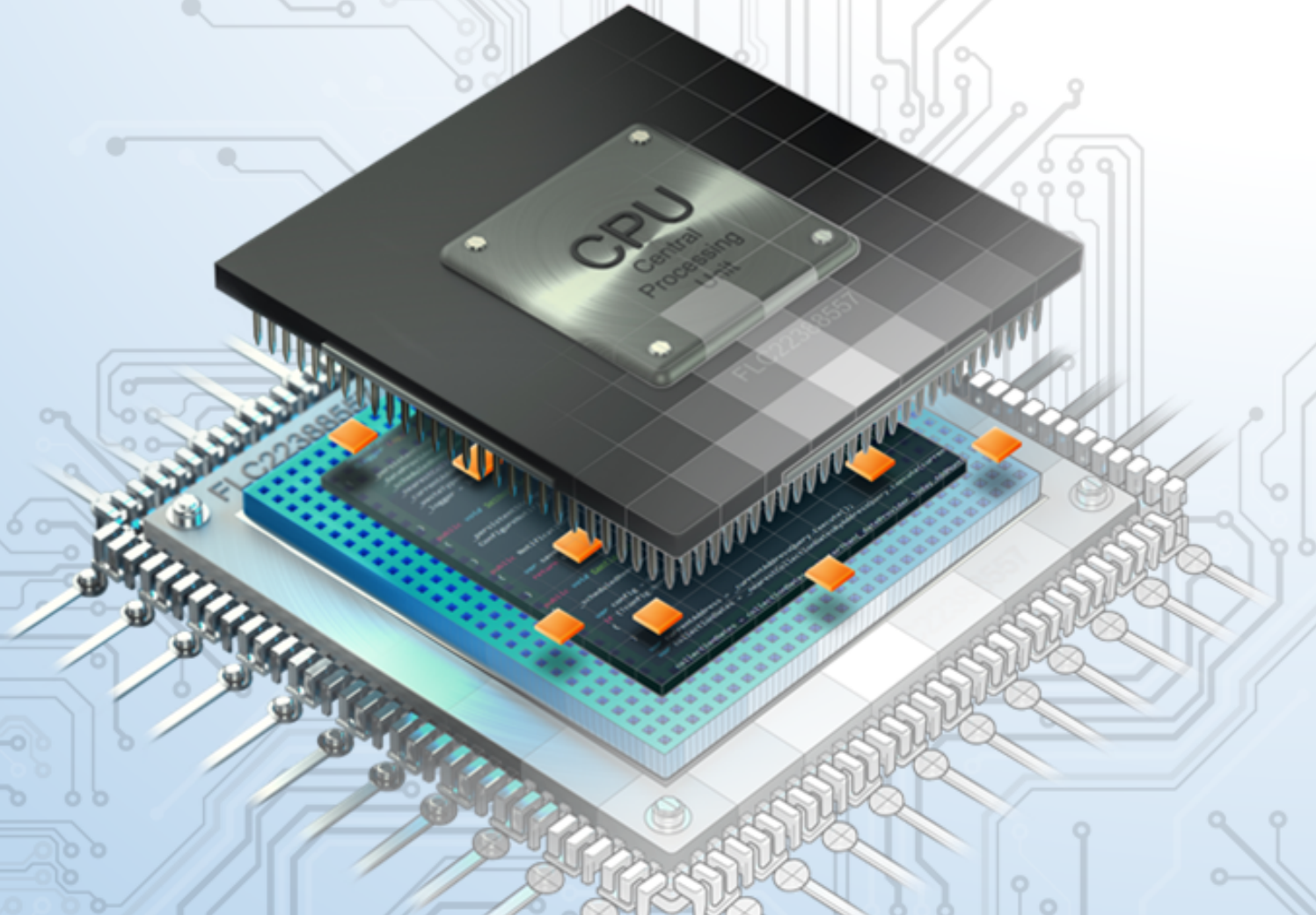
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
COMPUTER ENGINEERING

Microcontroller

Lab's Report



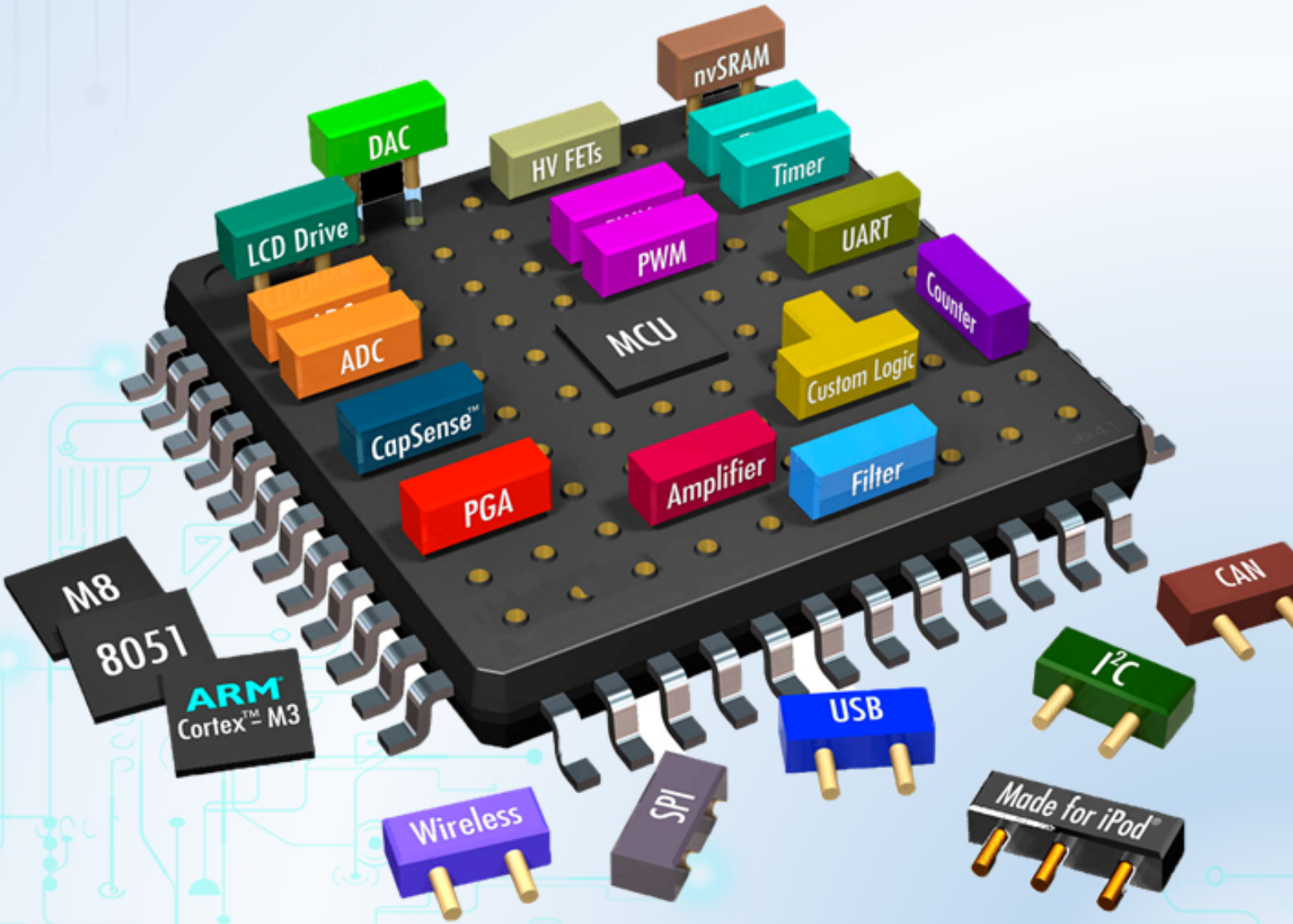
Date:15/09/2022

Cao Minh Quang - 2052221

Contents

CHAPTER 1

LED Animations



1 Github for version control

- Please access the following link to gain the full control of this lab.
- `https://github.com/cmq2002/Microcontroller_Microprocessor_Lab1.git`

2 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.

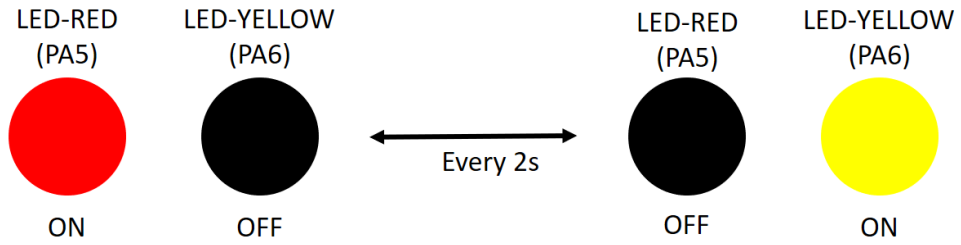


Figure 1.1: State transitions for 2 LEDs

Report 1: Depict the schematic from Proteus simulation in this report.

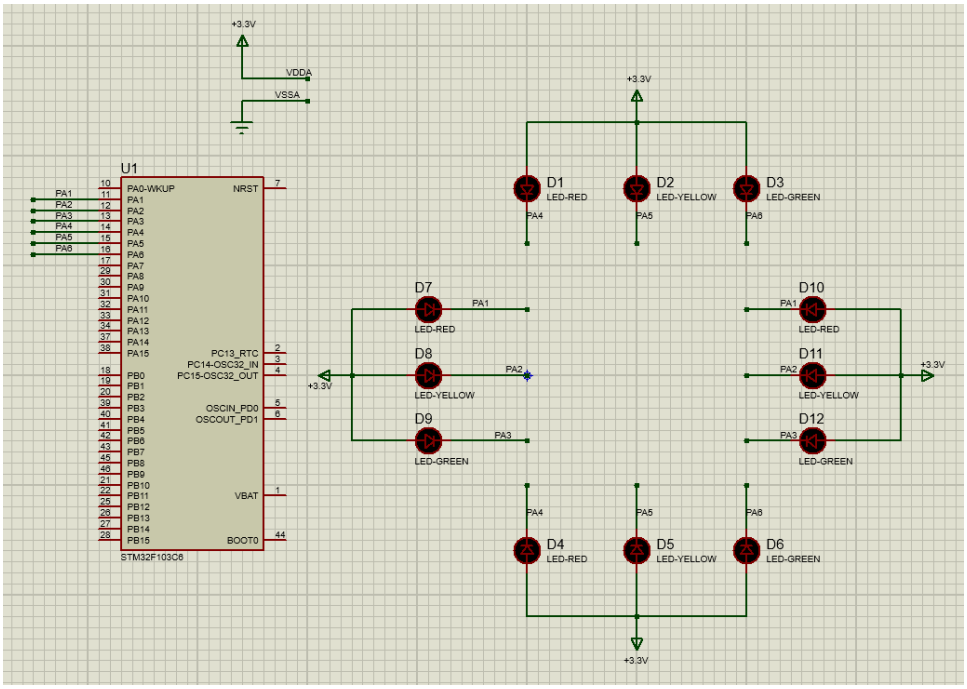


Figure 1.2: Schemetic capture in proteus

Report 2: Present the source code.

```
enum ledState {state0, state1};
enum ledState status = state1;
// state1 -> RED:on, Yellow: off
// state0 -> RED:off, Yellow: on
static int counter = 2;
const int threshold = 0;
while (1)
{
 switch (status){
   case state1:
     {
       counter--;
       if (counter == threshold){
         counter = setCounter();
         status = state0;
       }
       enableRED_LED();
     }
     break;

   case state0:
     {
       counter--;
       if (counter == threshold){
         counter = setCounter();
         status = state1;
       }
```

```
        enableYellow_LED();
      }
      break;
    default:
      break;
}
 HAL_Delay(1000);
/* USER CODE END WHILE */

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

Here is the support function:

```
/* Private user code
   ---------------------------------------------------*/
/* USER CODE BEGIN 0 */
int setCounter (void){
  return 2;
}

void enableRED_LED (void){
   HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin,
   GPIO_PIN_RESET);
   HAL_GPIO_WritePin(LED_YELLOW_GPIO_Port,
   LED_YELLOW_Pin, GPIO_PIN_SET);
}

void enableYellow_LED (void){
  HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin,
   GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_YELLOW_GPIO_Port,
   LED_YELLOW_Pin, GPIO_PIN_RESET);
}
/* USER CODE END 0 */
```

3 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.

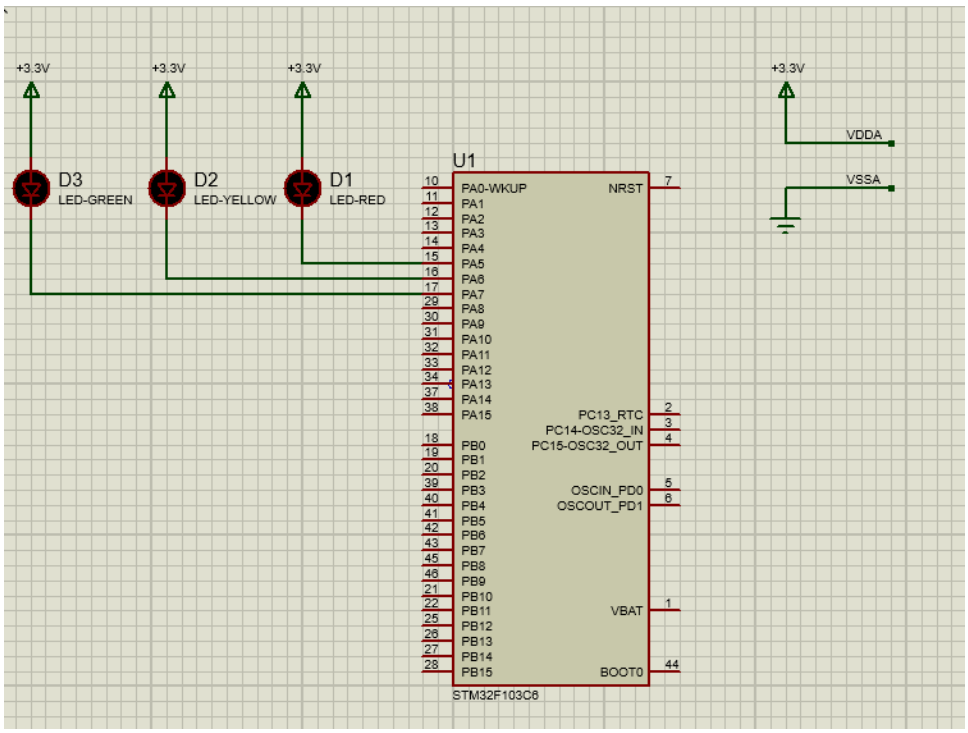


Figure 1.3: Schemetic capture in proteus

Report 2: Present the source code in while.

```
enum enableState {state0, state1, state2};
enum enableState status = state0;
// status = 0 -> Red:on, Yellow: off, Green: off
// status = 1 -> Red:off, Yellow: off, Green: on
// status = 2 -> Red:off, Yellow: on, Green: off
static int counterRED = 5;
static int counterYellow = 2;
static int counterGreen = 3;
const int threshold = 0;
while (1)
{
switch(status){
  case state0:
    {
      counterRED--;
      if (counterRED == threshold){
        counterRED = setCounterRED();
        status = state1;
      }
      enableRED_LED();
    }
    break;

  case state1:
    {
      counterGreen--;
```

```
      if (counterGreen == threshold){
        counterGreen = setCounterGreen();
        status = state2;
      }
      enableGreen_LED();
    }
    break;

  case state2:
    {
      counterYellow--;
      if (counterYellow == threshold){
        counterYellow = setCounterYellow();
        status = state0;
      }
      enableYellow_LED();
    }
    break;
  default:
    break;
}

HAL_Delay(1000);

/* USER CODE END WHILE */

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

Program 1.1: Source code

Here is the support fucntions:

```
/* Private user code
   ---------------------------------------------*/
/* USER CODE BEGIN 0 */
void enableRED_LED (void){
  HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin,
   GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED_YELLOW_GPIO_Port,
   LED_YELLOW_Pin, GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin,
    GPIO_PIN_SET);
}

void enableGreen_LED (void){
  HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin,
   GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_YELLOW_GPIO_Port,
   LED_YELLOW_Pin, GPIO_PIN_SET);
```

```
12    HAL_GPIO_WritePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin,
        GPIO_PIN_RESET);
13  }
14
15  void enableYellow_LED(void){
16    HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin,
       GPIO_PIN_SET);
17    HAL_GPIO_WritePin(LED_YELLOW_GPIO_Port,
       LED_YELLOW_Pin, GPIO_PIN_RESET);
18    HAL_GPIO_WritePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin,
        GPIO_PIN_SET);
19  }
20
21  static int setCounterRED (void){
22    return 5;
23  }
24
25  static int setCounterYellow (void){
26    return 2;
27  }
28
29  static int setCounterGreen (void){
30    return 3;
31  }
32  /* USER CODE END 0 */
```

4 Exercise 3

Extend to the 4-way traffic light. Arrange 12 LEDs in a nice shape to simulate the behaviors of a traffic light.

Report 1: Depict the schematic

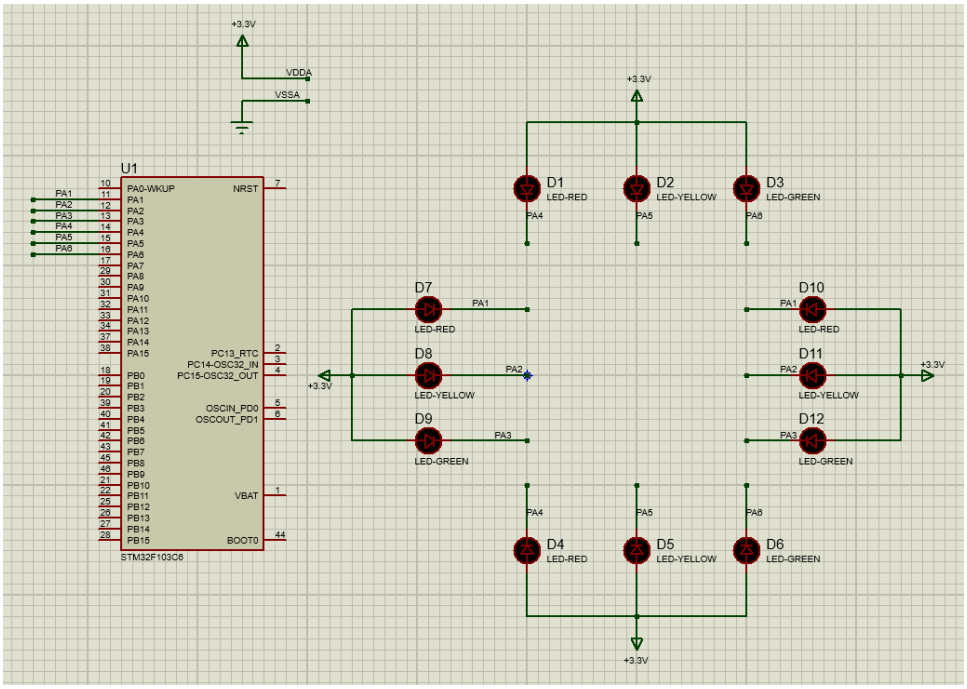


Figure 1.4: 4 way traffic light

Report 2: Present the source code

```
enum displayState {state0, state1, state2};
enum displayState status = state0;

// For West-East Direction:
// status = 0 -> Red1:on, Yellow1: off, Green1: off
// status = 1 -> Red1:off, Yellow1: off, Green1: on
// status = 2 -> Red1:off, Yellow1: on, Green1: off

// For North-South Direction:
// status = 0 -> Red2:off, Green2: on(3s) -> Yellow2:
   on(2s),
// status = 1 -> Red2:on, Yellow2: off, Green2: off
// status = 2 -> Red2:on, Yellow2: off, Green2: off

static int counterRED = 5;
static int counterYellow = 2;
static int counterGreen = 3;

const int threshold = 0;

while (1)
{
switch (status){
  case state0:
    {
      counterRED--;
      if (counterRED == threshold){
```

```
27          counterRED = setCounterRed();
28          status = state1;
29        }
30
31        //West-East Direction
32        enableRed1();
33
34        if (enableTerm_Green2(counterRED)==1){
35          //North-South Direction
36          enableGreen2();
37        }
38        if (enableTerm_Yellow2(counterRED)==1){
39          //North-South Direction
40          enableYellow2();
41        }
42      }
43      break;
44
45    case state1:
46      {
47        counterGreen--;
48        if (counterGreen == threshold){
49          counterGreen = setCounterGreen();
50          status = state2;
51        }
52
53        //West-East Direction
54        enableGreen1();
55
56        //North-South Direction
57        enableRed2();
58      }
59      break;
60
61    case state2:
62      {
63        counterYellow--;
64        if (counterYellow == threshold){
65          counterYellow = setCounterYellow();
66          status = state0;
67        }
68
69        //West-East Direction
70        enableYellow1();
71
72        //North-South Direction
73        enableRed2();
74      }
75      break;
```

```
}

HAL_Delay(1000);

/* USER CODE END WHILE */

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

Program 1.2: Source code

Here is the support functions:

```
/* Private user code
   ---------------------------------------*/
/* USER CODE BEGIN 0 */
void enableRed1 (void){
  HAL_GPIO_WritePin(LED_RED_1_GPIO_Port, LED_RED_1_Pin,
    GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED_YELLOW_1_GPIO_Port,
   LED_YELLOW_1_Pin, GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_GREEN_1_GPIO_Port,
   LED_GREEN_1_Pin, GPIO_PIN_SET);
}

void enableYellow1 (void){
  HAL_GPIO_WritePin(LED_RED_1_GPIO_Port, LED_RED_1_Pin,
    GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_YELLOW_1_GPIO_Port,
   LED_YELLOW_1_Pin, GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED_GREEN_1_GPIO_Port,
   LED_GREEN_1_Pin, GPIO_PIN_SET);
}

void enableGreen1 (void){
  HAL_GPIO_WritePin(LED_RED_1_GPIO_Port, LED_RED_1_Pin,
    GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_YELLOW_1_GPIO_Port,
   LED_YELLOW_1_Pin, GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_GREEN_1_GPIO_Port,
   LED_GREEN_1_Pin, GPIO_PIN_RESET);
}

void enableRed2 (void){
  HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
    GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port,
   LED_YELLOW_2_Pin, GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port,
   LED_GREEN_2_Pin, GPIO_PIN_SET);
}
```

```
26
27 void enableYellow2 (void){
28   HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
       GPIO_PIN_SET);
29   HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port,
      LED_YELLOW_2_Pin, GPIO_PIN_RESET);
30   HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port,
      LED_GREEN_2_Pin, GPIO_PIN_SET);
31 }
32
33 void enableGreen2 (void){
34   HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
       GPIO_PIN_SET);
35   HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port,
      LED_YELLOW_2_Pin, GPIO_PIN_SET);
36   HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port,
      LED_GREEN_2_Pin, GPIO_PIN_RESET);
37 }
38
39 static int setCounterRed (void){
40   return 5;
41 }
42
43 static int setCounterYellow (void){
44   return 2;
45 }
46
47 static int setCounterGreen (void){
48   return 3;
49 }
50
51 int enableTerm_Green2 (int value){
52   if (value > 1 && value < 5) return 1;
53   return 0;
54 }
55
56 int enableTerm_Yellow2 (int value){
57   if (value >= 0 && value <= 1) return 1;
58   return 0;
59 }
60 /* USER CODE END 0 */
```

5 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:

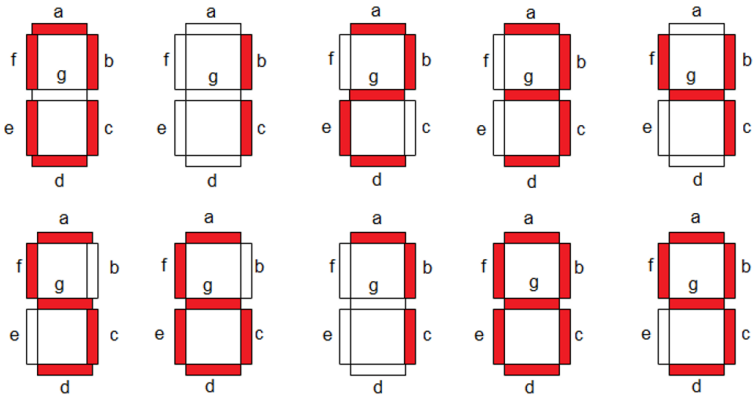


Figure 1.5: Display a number on 7 segment LED

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

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

}
```

Program 1.3: An example for your source code

Report 1: Present the schematic.

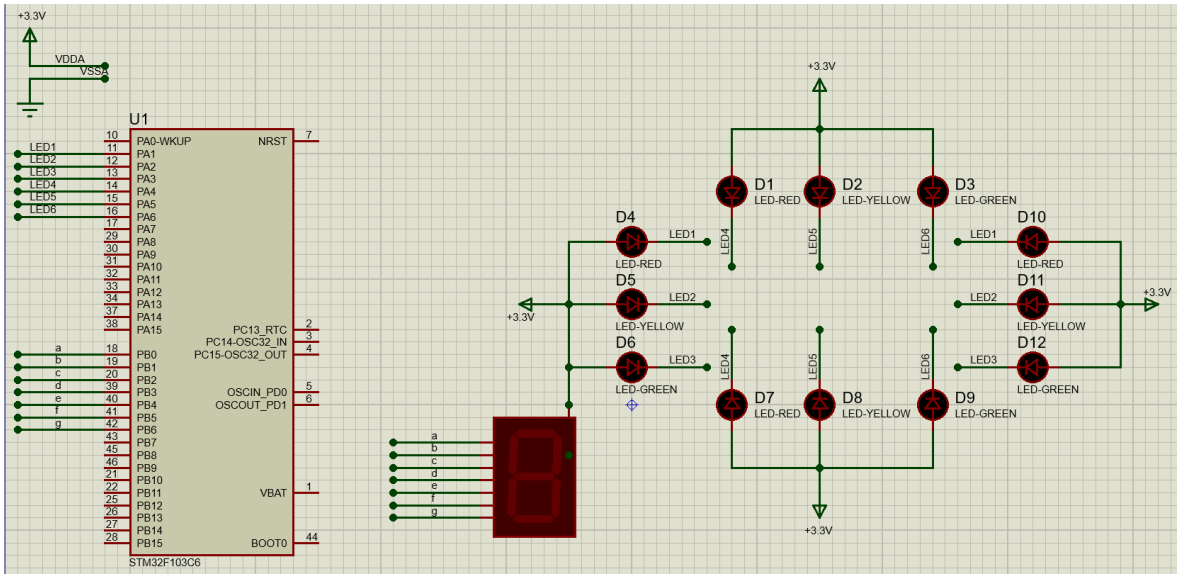


Figure 1.6: Display a number on 7 segment LED

Report 2: Present the source code for display7SEG function.

```
enum displayState {state0, state1, state2};
enum displayState status = state0;

// For West-East Direction:
// status = 0 -> Red1:on, Yellow1: off, Green1: off
// status = 1 -> Red1:off, Yellow1: off, Green1: on
// status = 2 -> Red1:off, Yellow1: on, Green1: off

// For North-South Direction:
// status = 0 -> Red2:off, Green2: on(3s) -> Yellow2:
   on(2s),
// status = 1 -> Red2:on, Yellow2: off, Green2: off
// status = 2 -> Red2:on, Yellow2: off, Green2: off

static int counterRED = 5;
static int counterYellow = 2;
static int counterGreen = 3;

const int threshold = 0;
while (1)
{
  switch (status){
    case state0:
      {
        //West-East Direction
        enableRed1();

        // Countdown number for W-E when red led on is
   the same as counterRED
        int temp1 = counterRED;
```

```
29          display7SEG1(temp1);
30
31          if (enableTerm_Green2(counterRED)==1){
32            //North-South Direction
33            enableGreen2();
34          }
35
36          if (enableTerm_Yellow2(counterRED)==1){
37            //North-South Direction
38            enableYellow2();
39          }
40
41          counterRED--;
42          if (counterRED == threshold){
43            counterRED = setCounterRed();
44            status = state1;
45          }
46        }
47        break;
48
49      case state1:
50        {
51          //West-East Direction
52          enableGreen1();
53
54          //Countdown number in W-E when green led on is
    same as counterGreen
55          int temp2 = counterGreen;
56          display7SEG1(temp2);
57
58          //North-South Direction
59          enableRed2();
60
61          counterGreen--;
62          if (counterGreen == threshold){
63            counterGreen = setCounterGreen();
64            status = state2;
65          }
66        }
67        break;
68
69      case state2:
70        {
71          //West-East Direction
72          enableYellow1();
73
74          //Countdown number for W-E when yellow led on
    is same as counterYellow
75          int temp3 = counterYellow;
```

```
        display7SEG1(temp3);

        //North-South Direction
        enableRed2();

        counterYellow--;
        if (counterYellow == threshold){
          counterYellow = setCounterYellow();
          status = state0;
        }
      }
      break;
    default:
      break;
  }
  HAL_Delay(1000);
/* USER CODE END WHILE */

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

Here is the support functions:

```
/* Private user code
   ---------------------------------------*/
/* USER CODE BEGIN 0 */
void display7SEG1 (int num){
  switch (num){
    case 0:
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, RESET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, RESET);
      HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, RESET);
      HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, RESET);
      HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, RESET);
      HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, RESET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, SET);
      break;
    case 1:
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, SET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, RESET);
      HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, RESET);
      HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, SET);
      HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, SET);
      HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, SET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, SET);
      break;
    case 2:
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, RESET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, RESET);
      HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, SET);
```

```
27        HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, RESET);
28        HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, RESET);
29        HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, SET);
30        HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
31        break;
32      case 3:
33        HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, RESET);
34        HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, RESET);
35        HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, RESET);
36        HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, RESET);
37        HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, SET);
38        HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, SET);
39        HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
40        break;
41      case 4:
42        HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, SET);
43        HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, RESET);
44        HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, RESET);
45        HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, SET);
46        HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, SET);
47        HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, RESET);
48        HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
49        break;
50      case 5:
51        HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, RESET);
52        HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, SET);
53        HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, RESET);
54        HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, RESET);
55        HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, SET);
56        HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, RESET);
57        HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
58        break;
59      default:
60        break;
61    }
62  }
63
64  void enableRed1 (void){
65    HAL_GPIO_WritePin(LED_RED_1_GPIO_Port, LED_RED_1_Pin,
        GPIO_PIN_RESET);
66    HAL_GPIO_WritePin(LED_YELLOW_1_GPIO_Port,
       LED_YELLOW_1_Pin, GPIO_PIN_SET);
67    HAL_GPIO_WritePin(LED_GREEN_1_GPIO_Port,
       LED_GREEN_1_Pin, GPIO_PIN_SET);
68  }
69
70  void enableYellow1 (void){
71    HAL_GPIO_WritePin(LED_RED_1_GPIO_Port, LED_RED_1_Pin,
        GPIO_PIN_SET);
```

```
72    HAL_GPIO_WritePin(LED_YELLOW_1_GPIO_Port,
       LED_YELLOW_1_Pin, GPIO_PIN_RESET);
73    HAL_GPIO_WritePin(LED_GREEN_1_GPIO_Port,
       LED_GREEN_1_Pin, GPIO_PIN_SET);
74  }
75
76  void enableGreen1 (void){
77    HAL_GPIO_WritePin(LED_RED_1_GPIO_Port, LED_RED_1_Pin,
        GPIO_PIN_SET);
78    HAL_GPIO_WritePin(LED_YELLOW_1_GPIO_Port,
       LED_YELLOW_1_Pin, GPIO_PIN_SET);
79    HAL_GPIO_WritePin(LED_GREEN_1_GPIO_Port,
       LED_GREEN_1_Pin, GPIO_PIN_RESET);
80  }
81
82  void enableRed2 (void){
83    HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
        GPIO_PIN_RESET);
84    HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port,
       LED_YELLOW_2_Pin, GPIO_PIN_SET);
85    HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port,
       LED_GREEN_2_Pin, GPIO_PIN_SET);
86  }
87
88  void enableYellow2 (void){
89    HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
        GPIO_PIN_SET);
90    HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port,
       LED_YELLOW_2_Pin, GPIO_PIN_RESET);
91    HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port,
       LED_GREEN_2_Pin, GPIO_PIN_SET);
92  }
93
94  void enableGreen2 (void){
95    HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
        GPIO_PIN_SET);
96    HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port,
       LED_YELLOW_2_Pin, GPIO_PIN_SET);
97    HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port,
       LED_GREEN_2_Pin, GPIO_PIN_RESET);
98  }
99
100 static int setCounterRed (void){
101   return 5;
102 }
103
104 static int setCounterYellow (void){
105   return 2;
106 }
```

```
107
108 static int setCounterGreen (void){
109   return 3;
110 }
111
112 int enableTerm_Green2 (int value){
113   if (value >=3) return 1;
114   return 0;
115 }
116
117 int enableTerm_Yellow2 (int value){
118   if (value <= 2) return 1;
119   return 0;
120 }
121 /* USER CODE END 0 */
```

6 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 and also the support functions.

```
1  enum displayState {state0, state1, state2};
2  enum displayState status = state0;
3  // For West-East Direction:
4  // status = 0 -> Red1:on, Yellow1: off, Green1: off
5  // status = 1 -> Red1:off, Yellow1: off, Green1: on
6  // status = 2 -> Red1:off, Yellow1: on, Green1: off
7
8  // For North-South Direction:
9  // status = 0 -> Red2:off, Green2: on(3s) -> Yellow2:
      on(2s),
10 // status = 1 -> Red2:on, Yellow2: off, Green2: off
11 // status = 2 -> Red2:on, Yellow2: off, Green2: off
12
13 static int counterRED = 5;
14 static int counterYellow = 2;
15 static int counterGreen = 3;
16
17 const int threshold = 0;
18 while (1)
19 {
20   switch (status){
21     case state0:
22       {
```

```
23        //West-East Direction
24        enableRed1();
25        // Countdown number for W-E when red led on is
  the same as counterRED
26        int temp1 = counterRED;
27        display7SEG1(temp1);
28
29        if (enableTerm_Green2(counterRED)==1){
30          //North-South Direction
31          enableGreen2();
32
33          // Countdown number for N-S when green led on
   must start at 3
34          int temp2 = counterRED - setCounterYellow();
35          display7SEG2(temp2);
36        }
37
38        if (enableTerm_Yellow2(counterRED)==1){
39          //North-South Direction
40          enableYellow2();
41
42          // When red led on W-E count to 2, it's also
  a countdown number for yellow led of N-S
43          display7SEG2(temp1);
44        }
45
46        counterRED--;
47        if (counterRED == threshold){
48          counterRED = setCounterRed();
49          status = state1;
50        }
51      }
52      break;
53
54    case state1:
55      {
56        //West-East Direction
57        enableGreen1();
58
59        //Countdown number in W-E when green led on is
  same as counterGreen
60        int temp1 = counterGreen;
61        display7SEG1(temp1);
62
63        //North-South Direction
64        enableRed2();
65
66        // Meanwhile, countdown in N-S when red led on
  must start at 5
```

```
67          int temp2 = counterGreen + setCounterYellow();
68          display7SEG2(temp2);
69
70          counterGreen--;
71          if (counterGreen == threshold){
72            counterGreen = setCounterGreen();
73            status = state2;
74          }
75        }
76        break;
77
78      case state2:
79        {
80          //West-East Direction
81          enableYellow1();
82
83          //Countdown number for W-E when yellow led on
    is same as counterYellow
84          int temp = counterYellow;
85          display7SEG1(temp);
86
87          //North-South Direction
88          enableRed2();
89
90          // The 2 seconds duration of yellow led in W-E
    is also the last 2 of red led on N-S
91          display7SEG2(temp);
92
93          counterYellow--;
94          if (counterYellow == threshold){
95            counterYellow = setCounterYellow();
96            status = state0;
97          }
98        }
99        break;
100     default:
101       break;
102   }
103   HAL_Delay(1000);
104 /* USER CODE END WHILE */
105
106 /* USER CODE BEGIN 3 */
107 }
```

7 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.

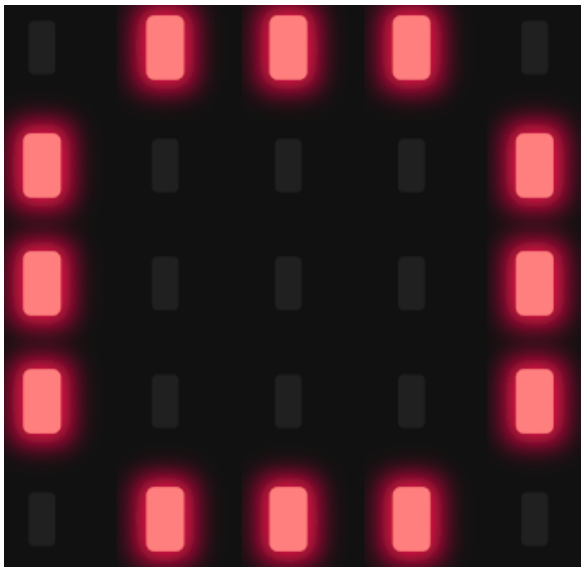


Figure 1.7: 12 LEDs for an analog clock

Report 1: Present the schematic.

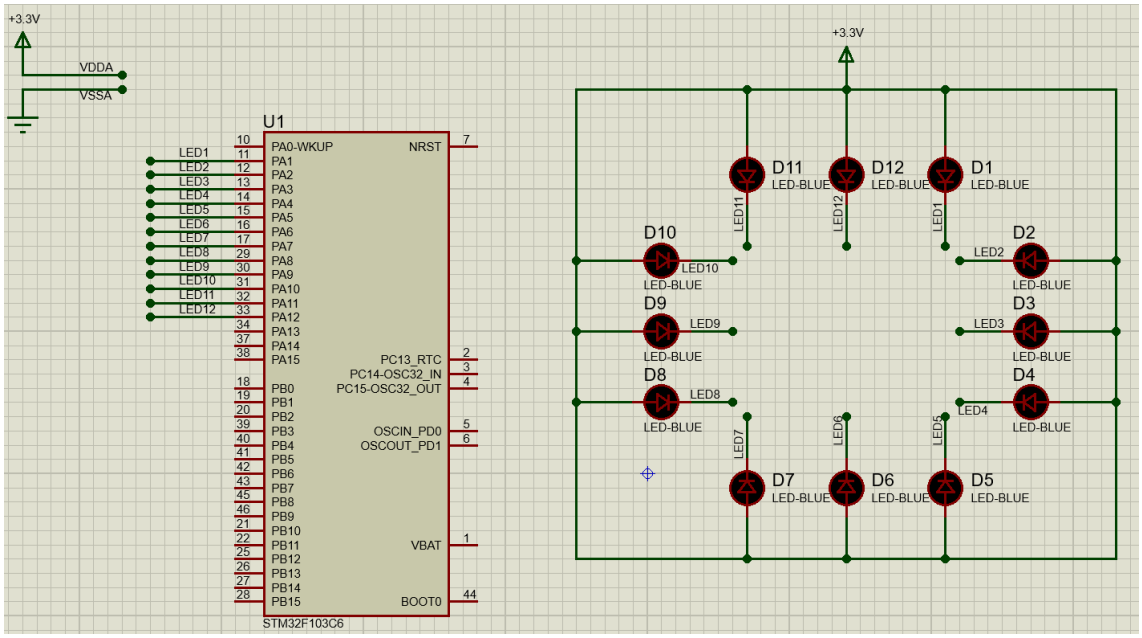


Figure 1.8: 12 LEDs for an analog clock

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

```
//Invoking this function in while loop of main to test
   all LEDs in sequence
void testingSystem(){
  HAL_GPIO_WritePin(LED_1_GPIO_Port, LED_1_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_1_GPIO_Port, LED_1_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_2_GPIO_Port, LED_2_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_2_GPIO_Port, LED_2_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_3_GPIO_Port, LED_3_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_3_GPIO_Port, LED_3_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_4_GPIO_Port, LED_4_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_4_GPIO_Port, LED_4_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_5_GPIO_Port, LED_5_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_5_GPIO_Port, LED_5_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_6_GPIO_Port, LED_6_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_6_GPIO_Port, LED_6_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_7_GPIO_Port, LED_7_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_7_GPIO_Port, LED_7_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_8_GPIO_Port, LED_8_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_8_GPIO_Port, LED_8_Pin, SET);
  HAL_Delay(100);

  HAL_GPIO_WritePin(LED_9_GPIO_Port, LED_9_Pin, RESET);
  HAL_Delay(100);
  HAL_GPIO_WritePin(LED_9_GPIO_Port, LED_9_Pin, SET);
  HAL_Delay(100);
```

```
47
48    HAL_GPIO_WritePin(LED_10_GPIO_Port, LED_10_Pin, RESET
      );
49    HAL_Delay(100);
50    HAL_GPIO_WritePin(LED_10_GPIO_Port, LED_10_Pin, SET);
51    HAL_Delay(100);
52
53    HAL_GPIO_WritePin(LED_11_GPIO_Port, LED_11_Pin, RESET
      );
54    HAL_Delay(100);
55    HAL_GPIO_WritePin(LED_11_GPIO_Port, LED_11_Pin, SET);
56    HAL_Delay(100);
57
58    HAL_GPIO_WritePin(LED_12_GPIO_Port, LED_12_Pin, RESET
      );
59    HAL_Delay(100);
60    HAL_GPIO_WritePin(LED_12_GPIO_Port, LED_12_Pin, SET);
61    HAL_Delay(100);
62  }
```

8 Exercise 7

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

```
1  void clearAllClock (){
2    HAL_GPIO_WritePin(LED_1_GPIO_Port, LED_1_Pin, SET);
3    HAL_GPIO_WritePin(LED_2_GPIO_Port, LED_2_Pin, SET);
4    HAL_GPIO_WritePin(LED_3_GPIO_Port, LED_3_Pin, SET);
5    HAL_GPIO_WritePin(LED_4_GPIO_Port, LED_4_Pin, SET);
6    HAL_GPIO_WritePin(LED_5_GPIO_Port, LED_5_Pin, SET);
7    HAL_GPIO_WritePin(LED_6_GPIO_Port, LED_6_Pin, SET);
8    HAL_GPIO_WritePin(LED_7_GPIO_Port, LED_7_Pin, SET);
9    HAL_GPIO_WritePin(LED_8_GPIO_Port, LED_8_Pin, SET);
10   HAL_GPIO_WritePin(LED_9_GPIO_Port, LED_9_Pin, SET);
11   HAL_GPIO_WritePin(LED_10_GPIO_Port, LED_10_Pin, SET);
12   HAL_GPIO_WritePin(LED_11_GPIO_Port, LED_11_Pin, SET);
13   HAL_GPIO_WritePin(LED_12_GPIO_Port, LED_12_Pin, SET);
14 }
```

Program 1.4: Function Implementation

9 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 0:
      HAL_GPIO_WritePin(LED_12_GPIO_Port, LED_12_Pin, RESET
   );
      break;
    case 1:
      HAL_GPIO_WritePin(LED_1_GPIO_Port, LED_1_Pin, RESET);
      break;
    case 2:
      HAL_GPIO_WritePin(LED_2_GPIO_Port, LED_2_Pin, RESET);
      break;
    case 3:
      HAL_GPIO_WritePin(LED_3_GPIO_Port, LED_3_Pin, RESET);
      break;
    case 4:
      HAL_GPIO_WritePin(LED_4_GPIO_Port, LED_4_Pin, RESET);
      break;
    case 5:
      HAL_GPIO_WritePin(LED_5_GPIO_Port, LED_5_Pin, RESET);
      break;
    case 6:
      HAL_GPIO_WritePin(LED_6_GPIO_Port, LED_6_Pin, RESET);
      break;
    case 7:
      HAL_GPIO_WritePin(LED_7_GPIO_Port, LED_7_Pin, RESET);
      break;
    case 8:
      HAL_GPIO_WritePin(LED_8_GPIO_Port, LED_8_Pin, RESET);
      break;
    case 9:
      HAL_GPIO_WritePin(LED_9_GPIO_Port, LED_9_Pin, RESET);
      break;
    case 10:
      HAL_GPIO_WritePin(LED_10_GPIO_Port, LED_10_Pin, RESET
   );
      break;
    case 11:
      HAL_GPIO_WritePin(LED_11_GPIO_Port, LED_11_Pin, RESET
   );
      break;
    default:
      break;
  }
}
```

10 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 0:
      HAL_GPIO_WritePin(LED_12_GPIO_Port, LED_12_Pin, SET);
      break;
    case 1:
      HAL_GPIO_WritePin(LED_1_GPIO_Port, LED_1_Pin, SET);
      break;
    case 2:
      HAL_GPIO_WritePin(LED_2_GPIO_Port, LED_2_Pin, SET);
      break;
    case 3:
      HAL_GPIO_WritePin(LED_3_GPIO_Port, LED_3_Pin, SET);
      break;
    case 4:
      HAL_GPIO_WritePin(LED_4_GPIO_Port, LED_4_Pin, SET);
      break;
    case 5:
      HAL_GPIO_WritePin(LED_5_GPIO_Port, LED_5_Pin, SET);
      break;
    case 6:
      HAL_GPIO_WritePin(LED_6_GPIO_Port, LED_6_Pin, SET);
      break;
    case 7:
      HAL_GPIO_WritePin(LED_7_GPIO_Port, LED_7_Pin, SET);
      break;
    case 8:
      HAL_GPIO_WritePin(LED_8_GPIO_Port, LED_8_Pin, SET);
      break;
    case 9:
      HAL_GPIO_WritePin(LED_9_GPIO_Port, LED_9_Pin, SET);
      break;
    case 10:
      HAL_GPIO_WritePin(LED_10_GPIO_Port, LED_10_Pin, SET);
      break;
    case 11:
      HAL_GPIO_WritePin(LED_11_GPIO_Port, LED_11_Pin, SET);
      break;
    default:
      break;
  }
}
```

11 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.

```
//Ex10
void display(int hr, int min, int sec){
  setNumberOnClock(hr);
  setNumberOnClock(min/5);
  setNumberOnClock(sec/5);
}

int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration
   ---------------------------------------------------------
   */

  /* Reset of all peripherals, Initializes the Flash
   interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  int hr = 0, min = 0, sec = 0;
  while (1)
  {
    //Ex10 - Integrate the whole system
```

```
42      clearAllClock();
43    sec++;
44    if (sec == 60){sec = 0; min++;}
45    if (min == 60){min = 0; hr++;}
46    if (hr == 12) {hr = 0; min = 0; sec = 0;}
47
48    display(hr,min,sec);
49    HAL_Delay(1000);
50    /* USER CODE END WHILE */
51
52      /* USER CODE BEGIN 3 */
53    }
54    /* USER CODE END 3 */
55  }
```