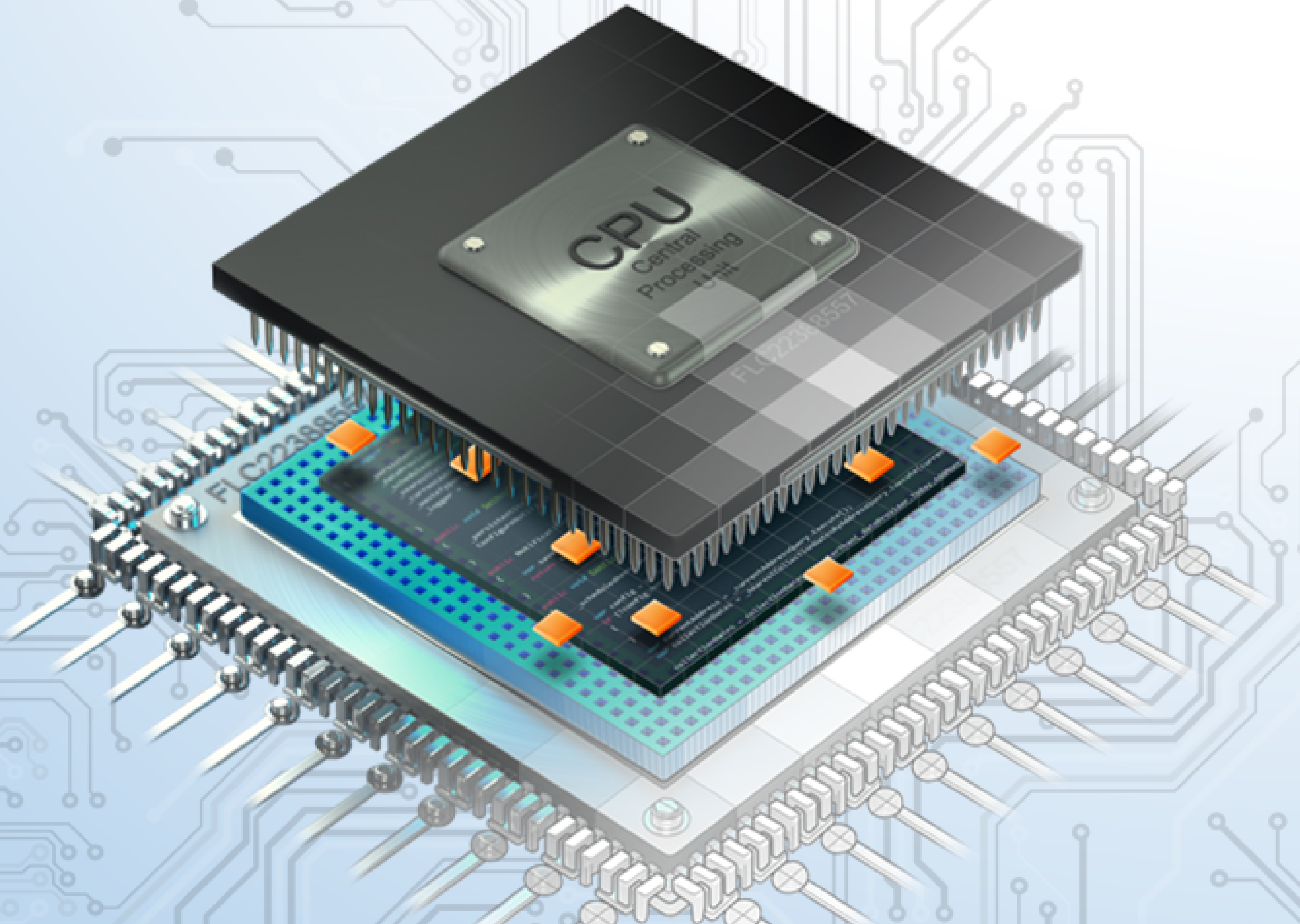
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
COMPUTER ENGINEERING

Microcontroller

Lab's Report



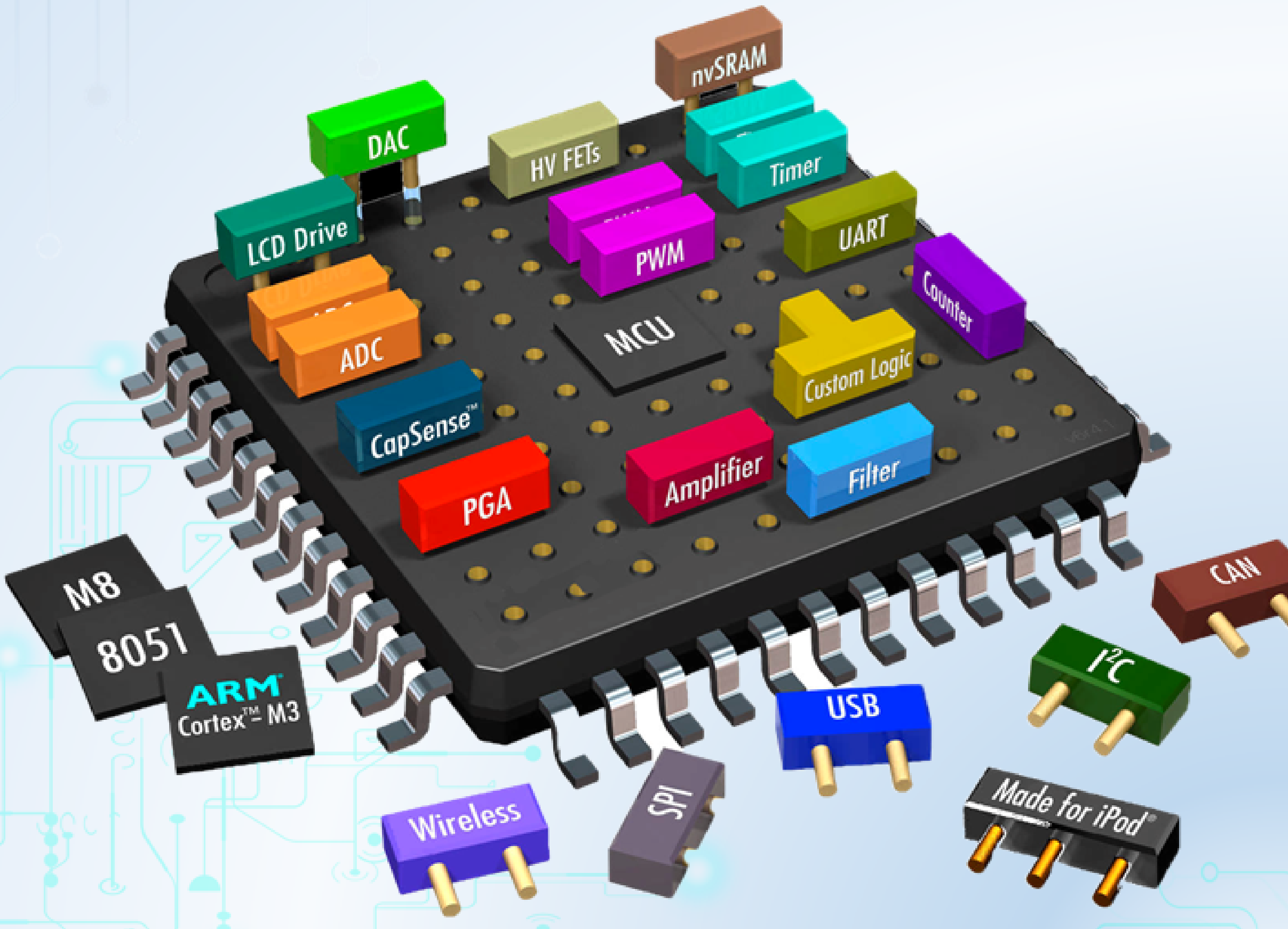
Date:15/09/2022

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.

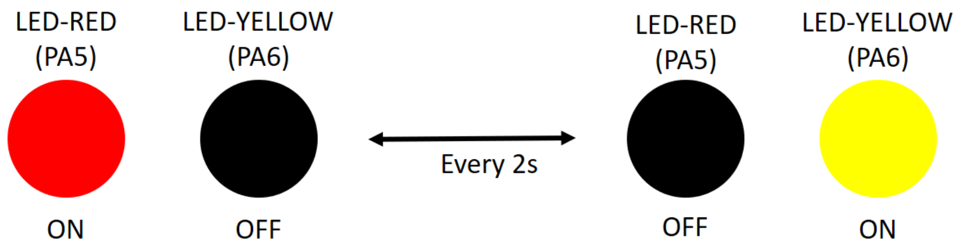


Figure 1.1: State transitions for 2 LEDs

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

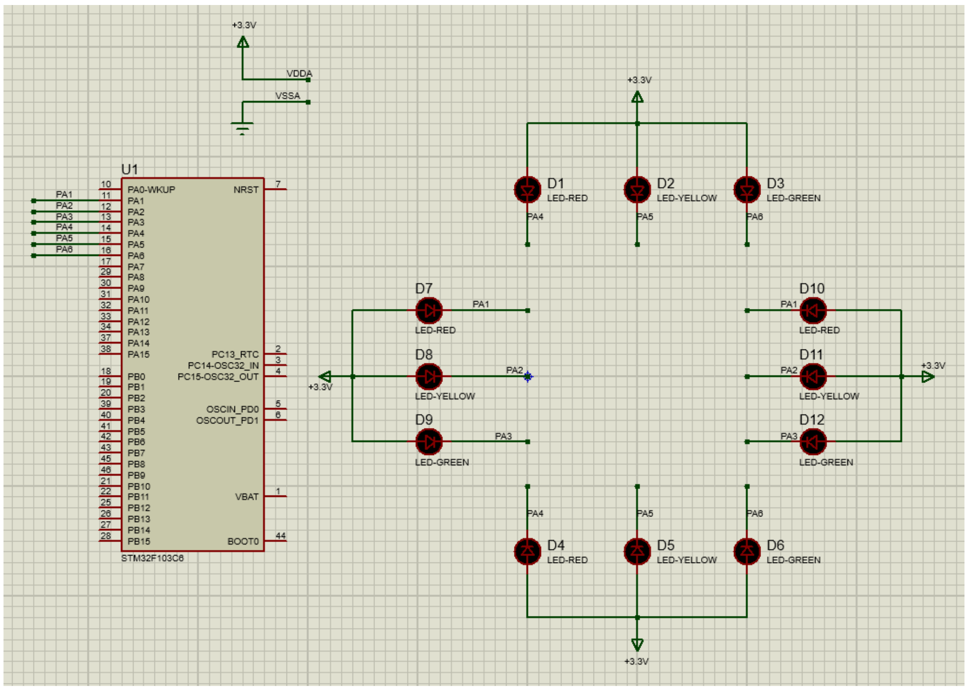


Figure 1.2: Schemetic capture in proteus

Report 2: Present the source code.

```
#define setCounter 2
#define threshold 0

int main(void)
{
  /* USER CODE BEGIN WHILE */
  enum ledState {state0, state1};
  enum ledState status = state1;
  // state1 -> RED:on, Yellow: off
  // state0 -> RED:off, Yellow: on
  static int counter = setCounter;
  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 */
    }
    /* USER CODE END 3 */
}
```

Here is the support function:

```
/* 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);
}

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.

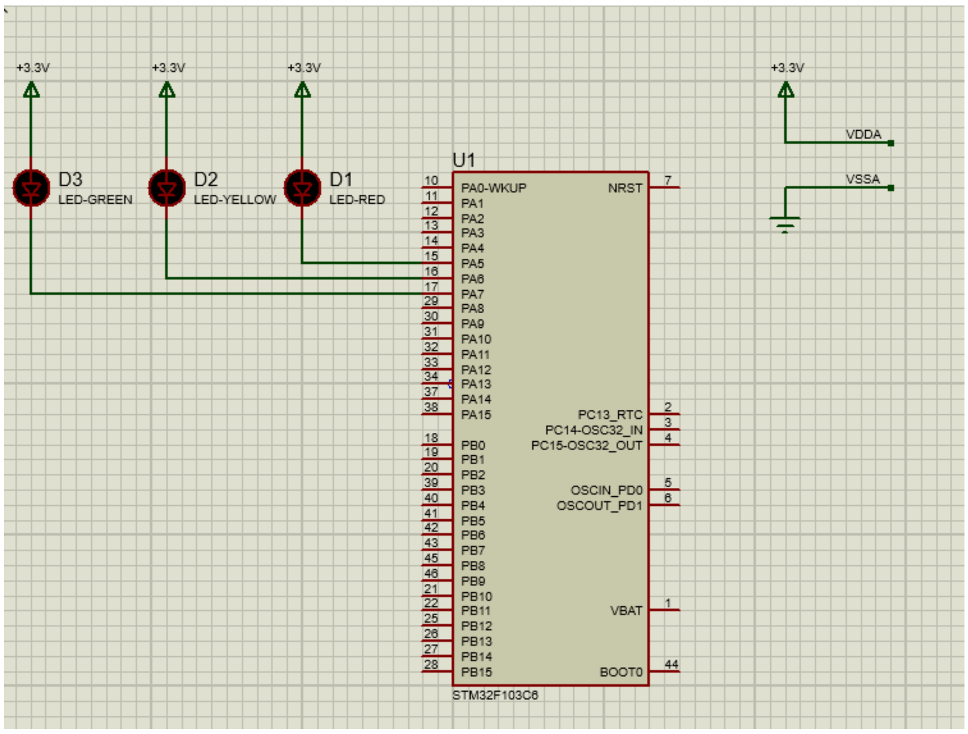


Figure 1.3: Schemetic capture in proteus

Report 2: Present the source code in while.

```
#define setCounterRed 5
#define setCounterYellow 2
#define setCounterGreen 3
#define threshold 0

int main(void)
{
  /* USER CODE BEGIN 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 = setCounterRed;
  static int counterYellow = setCounterYellow;
  static int counterGreen = setCounterGreen;
  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 */
  }
  /* USER CODE END 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);
  HAL_GPIO_WritePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin,
    GPIO_PIN_RESET);
}

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);
  HAL_GPIO_WritePin(LED_GREEN_GPIO_Port, LED_GREEN_Pin,
    GPIO_PIN_SET);
}
/* 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

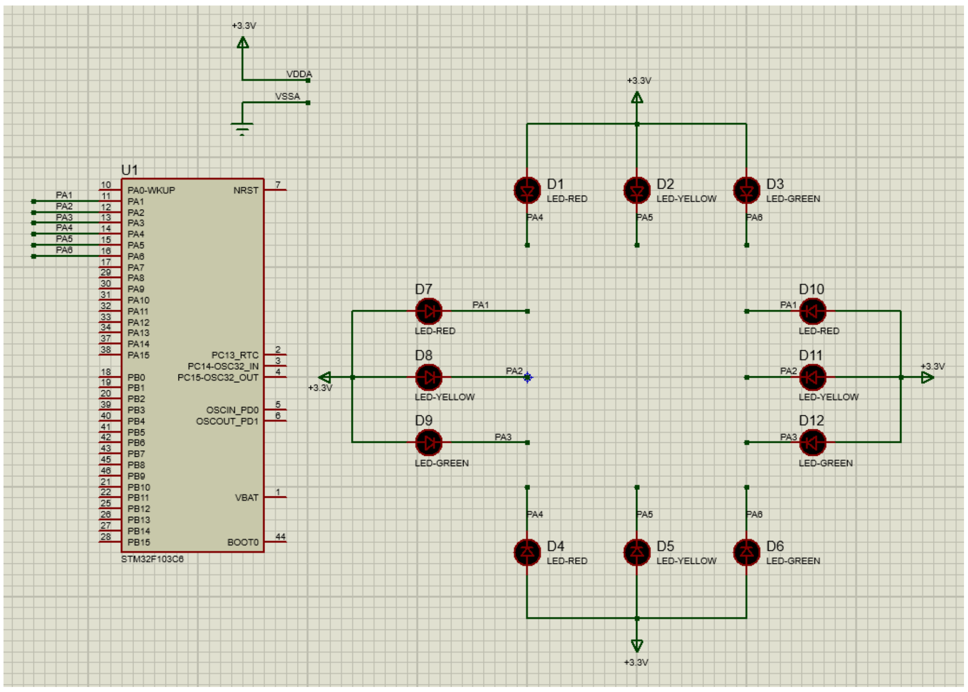


Figure 1.4: 4 way traffic light

Report 2: Present the source code

```c
#define setCounterRed 5
#define setCounterGreen 3
#define setCounterYellow 2
#define threshold 0

int main(void)
{
  /* USER CODE BEGIN WHILE */
  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 = setCounterRed;
  static int counterYellow = setCounterYellow;
  static int counterGreen = setCounterGreen;

  while (1)
  {
  switch (status){
    case state0:
      {
        //West-East Direction
        enableRed1();

        if (enableTerm_Green2(counterRED)==1){
          //North-South Direction
          enableGreen2();
        }
        if (enableTerm_Yellow2(counterRED)==1){
          //North-South Direction
          enableYellow2();
        }

        counterRED--;
        if (counterRED == threshold){
          counterRED = setCounterRed;
          status = state1;
        }
      }
      break;

    case state1:
      {
        //West-East Direction
        enableGreen1();

        //North-South Direction
        enableRed2();

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

    case state2:
      {
        //West-East Direction
        enableYellow1();

        //North-South Direction
        enableRed2();

        counterYellow--;
        if (counterYellow == threshold){
          counterYellow = setCounterYellow;
          status = state0;
        }
      }
      break;
  }

  HAL_Delay(1000);

  /* USER CODE END WHILE */

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

Program 1.2: Source code

Here is the support functions:

```
/* Private user code
   -------------------------------------*/
```

```c
/* 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);
}

void enableYellow2 (void){
  HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
     GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port,
   LED_YELLOW_2_Pin, GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port,
   LED_GREEN_2_Pin, GPIO_PIN_SET);
}

void enableGreen2 (void){
  HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin,
     GPIO_PIN_SET);
  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_RESET);
}

int enableTerm_Green2 (int value){
  if (value >= setCounterGreen) return 1;
  return 0;
}

int enableTerm_Yellow2 (int value){
  if (value <= setCounterYellow) return 1;
  return 0;
}
/* 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:

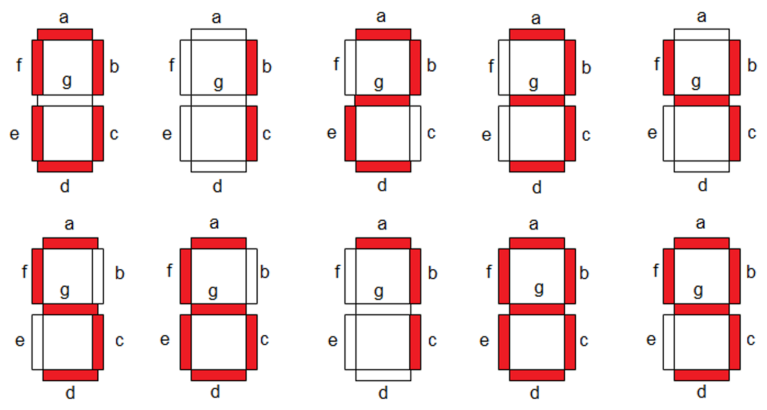


Figure 1.5: Display a number on 7 segment LED

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.

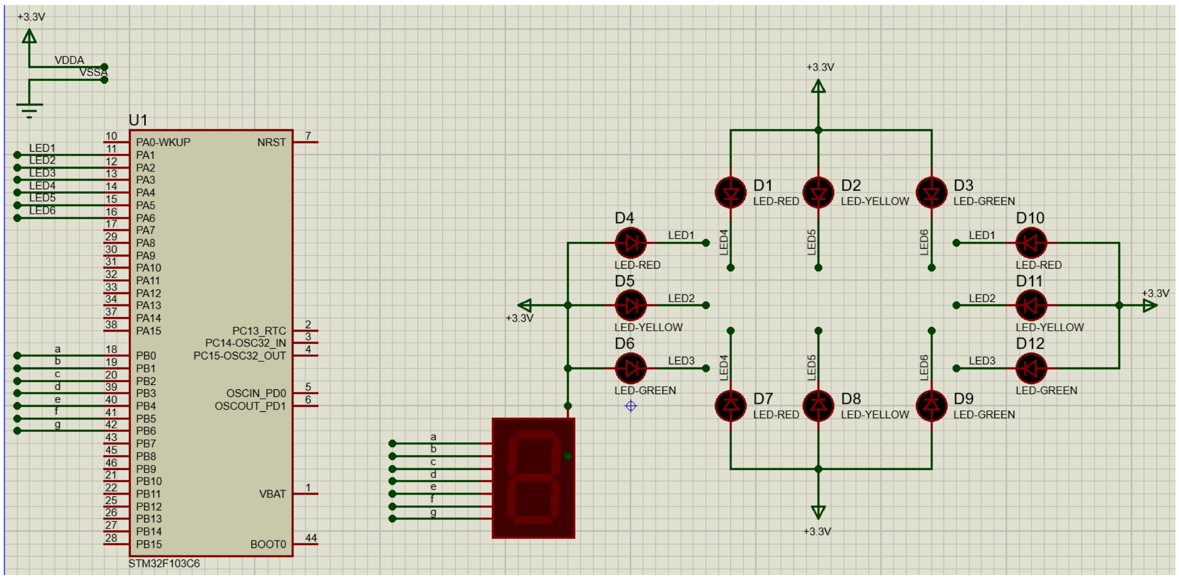


Figure 1.6: Display a number on 7 segment LED

Report 2: Present the source code for display7SEG function.

```c
#define setCounterRed 5
#define setCounterGreen 3
#define setCounterYellow 2
#define threshold 0

int main(void)
{
  /* USER CODE BEGIN WHILE */
  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 = setCounterRed;
  static int counterYellow = setCounterYellow;
  static int counterGreen = setCounterGreen;

  while (1)
  {
    switch (status){
      case state0:
        {
          //West-East Direction
          enableRed1();

          // Countdown number for W-E when red led on is the same as counterRED
          display7SEG1(counterRED);

          if (enableTerm_Green2(counterRED)==1){
            //North-South Direction
            enableGreen2();
          }

          if (enableTerm_Yellow2(counterRED)==1){
            //North-South Direction
            enableYellow2();
          }

          counterRED--;
          if (counterRED == threshold){
            counterRED = setCounterRed;
            status = state1;
          }
        }
        break;

      case state1:
        {
          //West-East Direction
          enableGreen1();

          //Countdown number in W-E when green led on is same as counterGreen
          display7SEG1(counterGreen);

          //North-South Direction
          enableRed2();

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

      case state2:
        {
          //West-East Direction
```

```
            enableYellow1();

            //Countdown number for W-E when yellow led on
    is same as counterYellow
            display7SEG1(counterYellow);

            //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);
```

```c
      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);
      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, SET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
      break;
    case 3:
      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, SET);
      HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, SET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
      break;
    case 4:
      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, RESET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
      break;
    case 5:
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, RESET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, SET);
      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, SET);
      HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, RESET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, RESET);
      break;
    default:
      break;
  }
}

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);
```

```c
}

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);
}

void enableYellow2 (void){
  HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin, GPIO_PIN_SET);
  HAL_GPIO_WritePin(LED_YELLOW_2_GPIO_Port, LED_YELLOW_2_Pin, GPIO_PIN_RESET);
  HAL_GPIO_WritePin(LED_GREEN_2_GPIO_Port, LED_GREEN_2_Pin, GPIO_PIN_SET);
}

void enableGreen2 (void){
  HAL_GPIO_WritePin(LED_RED_2_GPIO_Port, LED_RED_2_Pin, GPIO_PIN_SET);
  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_RESET);
}

int enableTerm_Green2 (int value){
  if (value >= setCounterGreen) return 1;
  return 0;
}

int enableTerm_Yellow2 (int value){
  if (value <= setCounterYellow) return 1;
  return 0;
}
/* 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.

```
#define setCounterRed 5
#define setCounterGreen 3
#define setCounterYellow 2
#define threshold 0

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

  enum displayState {state0, state1, state2};
  enum displayState status = state0;
  enum displayState status1= state1;
  // 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 = setCounterRed;
  static int counterYellow = setCounterYellow;
  static int counterGreen = setCounterGreen;

  static int counterRED1 = setCounterRed;
  static int counterYellow1 = setCounterYellow;
  static int counterGreen1 = setCounterGreen;
```

```c

while (1)
{
  switch (status){
    case state0:
      enableRed1();
      display7SEG1(counterRED);
      counterRED--;
      if (counterRED == threshold){
        counterRED = setCounterRed;
        status = state1;
      }
      break;
    case state1:
      enableGreen1();
      display7SEG1(counterGreen);
      counterGreen--;
      if (counterGreen == threshold){
        counterGreen = setCounterGreen;
        status = state2;
      }
      break;
    case state2:
      enableYellow1();
      display7SEG1(counterYellow);
      counterYellow--;
      if (counterYellow == threshold){
        counterYellow = setCounterYellow;
        status = state0;
      }
      break;
    default:
      break;
  }

  switch (status1){
    case state0:
      enableRed2();
      display7SEG2(counterRED1);
      counterRED1--;
      if (counterRED1 == threshold){
        counterRED1 = setCounterRed;
        status1 = state1;
      }
      break;
    case state1:
      enableGreen2();
      display7SEG2(counterGreen1);
      counterGreen1--;
```

```
        if (counterGreen1 == threshold){
          counterGreen1 = setCounterGreen;
          status1 = state2;
        }
        break;
      case state2:
        enableYellow2();
        display7SEG2(counterYellow1);
        counterYellow1--;
        if (counterYellow1 == threshold){
          counterYellow1 = setCounterYellow;
          status1 = state0;
        }
        break;
      default:
        break;
    }
    HAL_Delay(1000);
    /* USER CODE END WHILE */

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

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.

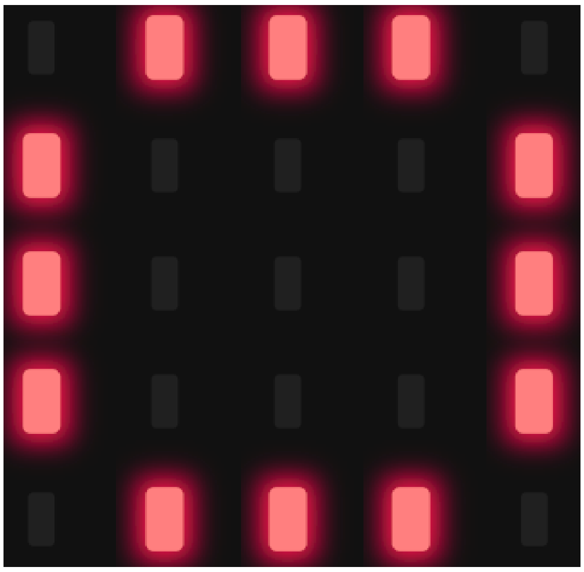


Figure 1.7: 12 LEDs for an analog clock

Report 1: Present the schematic.

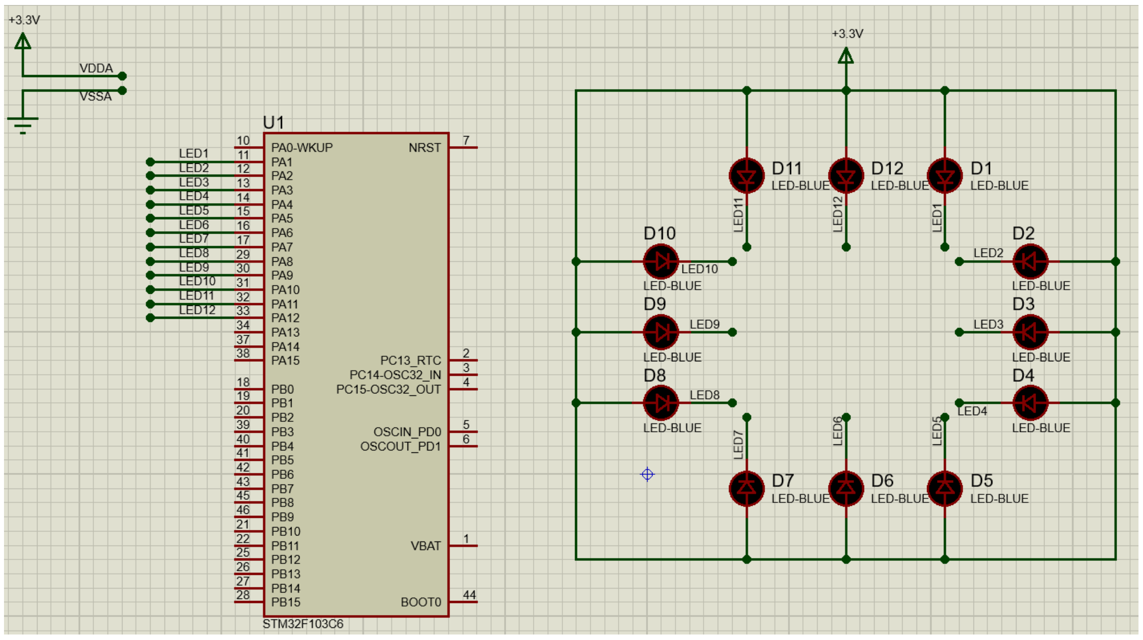


Figure 1.8: 12 LEDs for an analog clock

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

```c
//Invoking this function in while loop of main to test
   all LEDs in sequence
void testingSystem(){
  HAL_GPIO_TogglePin(LED_1_GPIO_Port, LED_1_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_2_GPIO_Port, LED_2_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_3_GPIO_Port, LED_3_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_4_GPIO_Port, LED_4_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_5_GPIO_Port, LED_5_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_6_GPIO_Port, LED_6_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_7_GPIO_Port, LED_7_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_8_GPIO_Port, LED_8_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_9_GPIO_Port, LED_9_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_10_GPIO_Port, LED_10_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_11_GPIO_Port, LED_11_Pin);
  HAL_Delay(100);
  HAL_GPIO_TogglePin(LED_12_GPIO_Port, LED_12_Pin);
  HAL_Delay(100);
}
```

8 Exercise 7

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

```c
void clearAllClock (){
  HAL_GPIO_WritePin(LED_1_GPIO_Port, LED_1_Pin, SET);
  HAL_GPIO_WritePin(LED_2_GPIO_Port, LED_2_Pin, SET);
  HAL_GPIO_WritePin(LED_3_GPIO_Port, LED_3_Pin, SET);
  HAL_GPIO_WritePin(LED_4_GPIO_Port, LED_4_Pin, SET);
  HAL_GPIO_WritePin(LED_5_GPIO_Port, LED_5_Pin, SET);
  HAL_GPIO_WritePin(LED_6_GPIO_Port, LED_6_Pin, SET);
  HAL_GPIO_WritePin(LED_7_GPIO_Port, LED_7_Pin, SET);
  HAL_GPIO_WritePin(LED_8_GPIO_Port, LED_8_Pin, SET);
  HAL_GPIO_WritePin(LED_9_GPIO_Port, LED_9_Pin, SET);
  HAL_GPIO_WritePin(LED_10_GPIO_Port, LED_10_Pin, SET);
  HAL_GPIO_WritePin(LED_11_GPIO_Port, LED_11_Pin, SET);
  HAL_GPIO_WritePin(LED_12_GPIO_Port, LED_12_Pin, SET);
}
```

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.

```c
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.

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

int main(void)
{
  /* USER CODE BEGIN WHILE */
  int hr = 0, min = 0, sec = 0;
  while (1)
  {
    //Ex10 - Integrate the whole system
    clearAllClock();
  sec++;
  if (sec == 60){sec = 0; min++;}
  if (min == 60){min = 0; hr++;}
  if (hr == 12) {hr = 0; min = 0; sec = 0;}

  display(hr,min,sec);
  HAL_Delay(1000);
  /* USER CODE END WHILE */

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