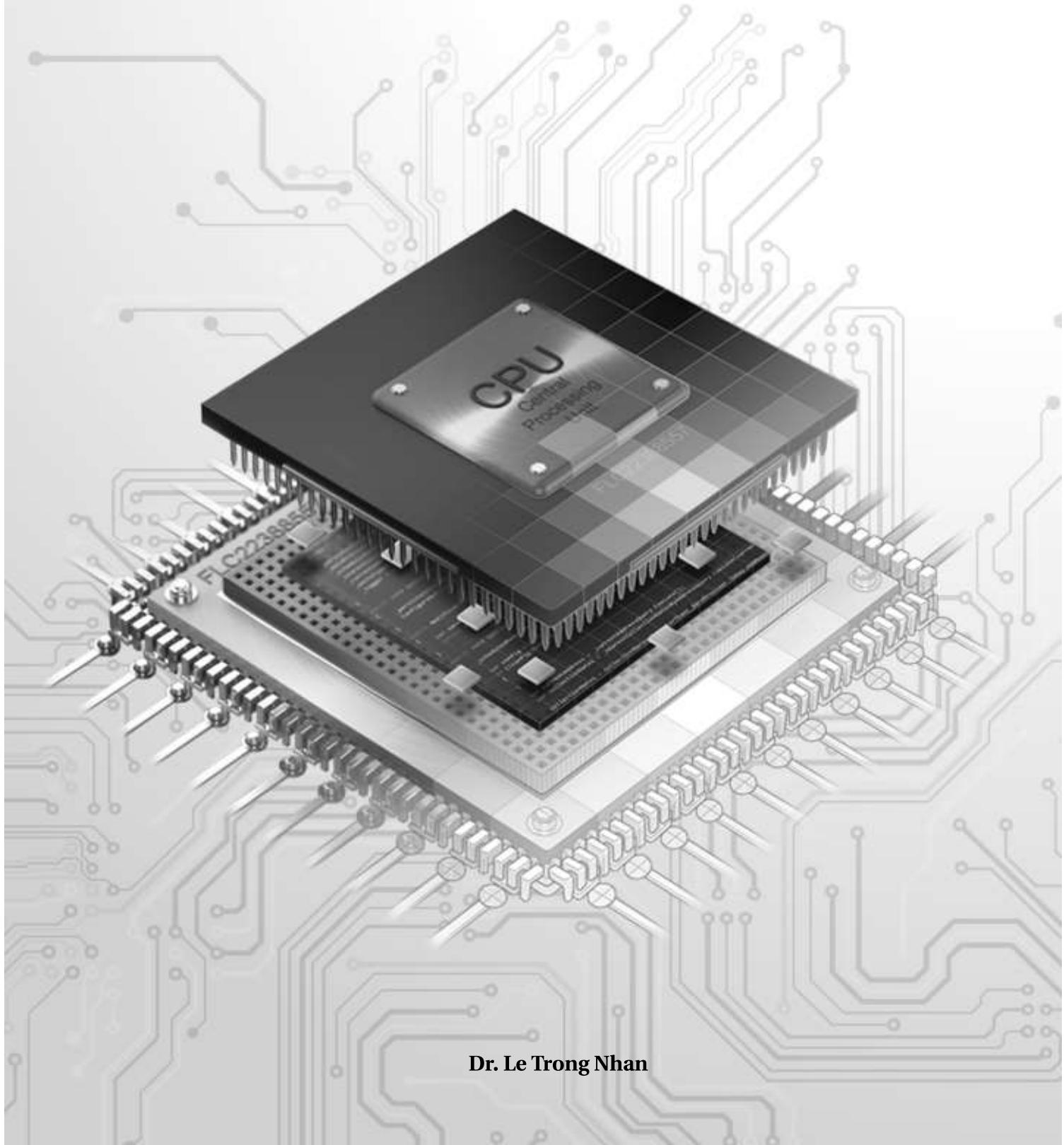




HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
COMPUTER ENGINEERING

Microcontroller



Dr. Le Trong Nhan

Mục lục

Chapter 1. LED Animations	9
1 Introduction	10
2 First project on STM32Cube	11
3 Simulation on Proteus	17
4 Exercise and Report	23
4.1 Exercise 1	23
4.2 Exercise 2	24
4.3 Exercise 3	26
4.4 Exercise 4	29
4.5 Exercise 5	34
4.6 Exercise 6	46
4.7 Exercise 7	48
4.8 Exercise 8	48
4.9 Exercise 9	49
4.10 Exercise 10	49
Chapter 2. Timer Interrupt and LED Scanning	53
1 Introduction	54
2 Timer Interrupt Setup	56
3 Exercise and Report	59
3.1 Exercise 1	59
3.2 Exercise 2	60
3.3 Exercise 3	61
3.4 Exercise 4	62
3.5 Exercise 5	62
3.6 Exercise 6	63
3.7 Exercise 7	65

3.8	Exercise 8	65
3.9	Exercise 9	65
3.10	Exercise 10	66
Chapter 3. Buttons/Switches		67
1	Objectives	68
2	Introduction	68
3	Basic techniques for reading from port pins	70
3.1	The need for pull-up resistors	70
3.2	Dealing with switch bounces	70
4	Reading switch input (basic code) using STM32	75
4.1	Input Output Processing Patterns	75
4.2	Setting up	76
4.2.1	Create a project	76
4.2.2	Create a file C source file and header file for input reading	76
4.3	Code For Read Port Pin and Debouncing	78
4.3.1	The code in the input_reading.c file	78
4.3.2	The code in the input_reading.h file	79
4.3.3	The code in the timer.c file	79
4.4	Button State Processing	80
4.4.1	Finite State Machine	80
4.4.2	The code for the FSM in the input_processing.c file	81
4.4.3	The code in the input_processing.h	81
4.4.4	The code in the main.c file	82
5	Exercises and Report	83
5.1	Specifications	83
5.2	Exercise 1: Sketch an FSM	84
5.3	Exercise 2: Proteus Schematic	84
5.4	Exercise 3: Create STM32 Project	84
5.5	Exercise 4: Modify Timer Parameters	84
5.6	Exercise 5: Adding code for button debouncing	84
5.7	Exercise 6: Adding code for displaying modes	85
5.8	Exercise 7: Adding code for increasing time duration value for the red LEDs	85

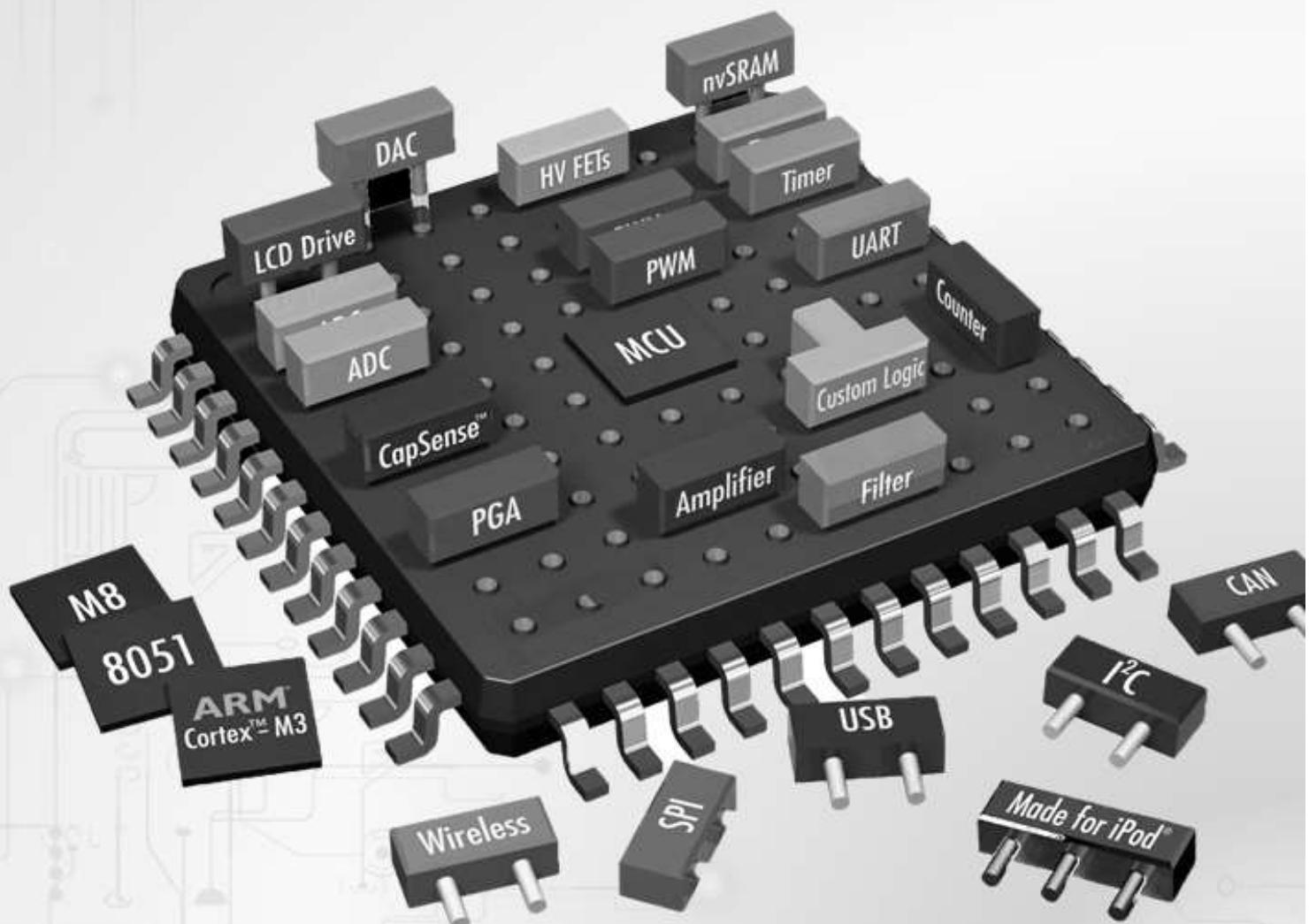
5.9	Exercise 8: Adding code for increasing time duration value for the amber LEDs	85
5.10	Exercise 9: Adding code for increasing time duration value for the green LEDs	85
5.11	Exercise 10: To finish the project	86
Chapter 4. Digital Clock Project		87
Chapter 5. A cooperative scheduler		89
1	Introduction	90
1.1	Super Loop Architecture	90
1.2	Timer-based interrupts and interrupt service routines	91
2	What is a scheduler?	92
2.1	The co-operative scheduler	92
2.2	Function pointers	93
2.3	Solution	94
2.3.1	Overview	95
2.3.2	The scheduler data structure and task array	96
2.3.3	The initialization function	97
2.3.4	The ‘Update’ function	97
2.3.5	The ‘Add Task’ function	98
2.3.6	The ‘Dispatcher’	99
2.3.7	The ‘Delete Task’ function	101
2.3.8	Reducing power consumption	101
2.3.9	Reporting errors	102
2.3.10	Adding a watchdog	103
2.3.11	Reliability and safety implications	104
2.3.12	Portability	104
3	Objectives	104
4	Problem	105
5	Demonstration	105
6	Submission	106
7	References	106
Chapter 6. Flow and Error Control in Communication		107
1	Introduction	108
2	Proteus simulation platform	109

3	Project configurations	110
3.1	UART Configuration	110
3.2	ADC Input	111
4	UART loop-back communication	111
5	Sensor reading	112
6	Project description	113
6.1	Command parser	113
6.2	Project implementation	114
Chapter 7. Real Time Operating System		115
Chapter 8. Bài Tập Giữa Kì		117
1	Giới thiệu	118
2	Nộp bài	118
3	Report	119
3.1	Mô phỏng trên Proteus	119
3.2	Thiết kế máy trạng thái	119
3.3	Lập trình trên STM32Cube	119
3.4	Module Timer	120
3.4.1	File timer.h	120
3.4.2	File timer.c	120
4	Video demo	121
Chapter 9. MIDTERM 2022		123
1	Introduction	124
2	Implement and Report	125
2.1	Proteus schematic - 1 point	125
2.2	State machine Step 1 - 2 points	125
2.3	State machine Step 2 - 2 points	126
2.4	State machine Step 3 - 2 points	126
2.5	Led Blinky for Debugging - 1 point	127
2.6	Github and Demo	127
3	Extra exercise - Engineer mindset -1 point	127
Chapter 10. GIỮA KÌ 2022		129
1	Giới thiệu	130

2	Hiện thực và Report	131
2.1	Sơ đồ nguyên lý trên Proteus - 1 điểm	131
2.2	State machine Step 1 - 2 điểm	131
2.3	State machine Step 2 - 2 điểm	132
2.4	State machine Step 3 - 2 points	132
2.5	Led Blinky for Debugging - 1 điểm	133
2.6	Github và Demo	133
3	Bài tập thêm - Engineer mindset - 1 điểm	133

CHƯƠNG 1

LED Animations



1 Introduction

In this manual, the STM32CubeIDE is used as an editor to program the ARM microcontroller. STM32CubeIDE is an advanced C/C++ development platform with peripheral configuration, code generation, code compilation, and debug features for STM32 microcontrollers and microprocessors.



Hình 1.1: STM32Cube IDE for STM32 Programming

The most interest of STM32CubeIDE is that after the selection of an empty STM32 MCU or MPU, or preconfigured microcontroller or microprocessor from the selection of a board, the initialization code generated automatically. At any time during the development, the user can return to the initialization and configuration of the peripherals or middleware and regenerate the initialization code with no impact on the user code. This feature can simplify the initialization process and speedup the development application running on STM32 micro-controller. The software can be downloaded from the link bellow:

https://ubc.sgp1.digitaloceanspaces.com/BKU_Softwares/STM32/stm32cubeide_1.7.0.zip

Moreover, for a hangout class, the program is firstly simulated on Proteus. Students are also supposed to download and install this software as well:

https://ubc.sgp1.digitaloceanspaces.com/BKU_Softwares/STM32/Proteus_8.10_SP0_Pro.exe

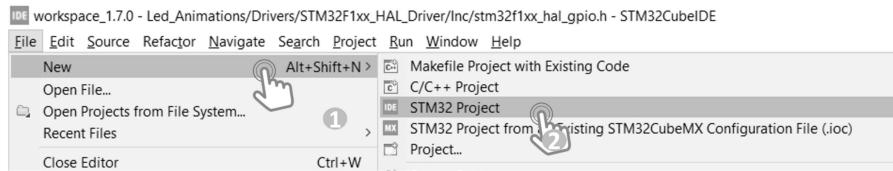
The rest of this manual consists of:

- Create a project on STM32Cube IDE
- Create a project on Proteus
- Simulate the project on Proteus

Finally, students are supposed to finish 10 different projects.

2 First project on STM32Cube

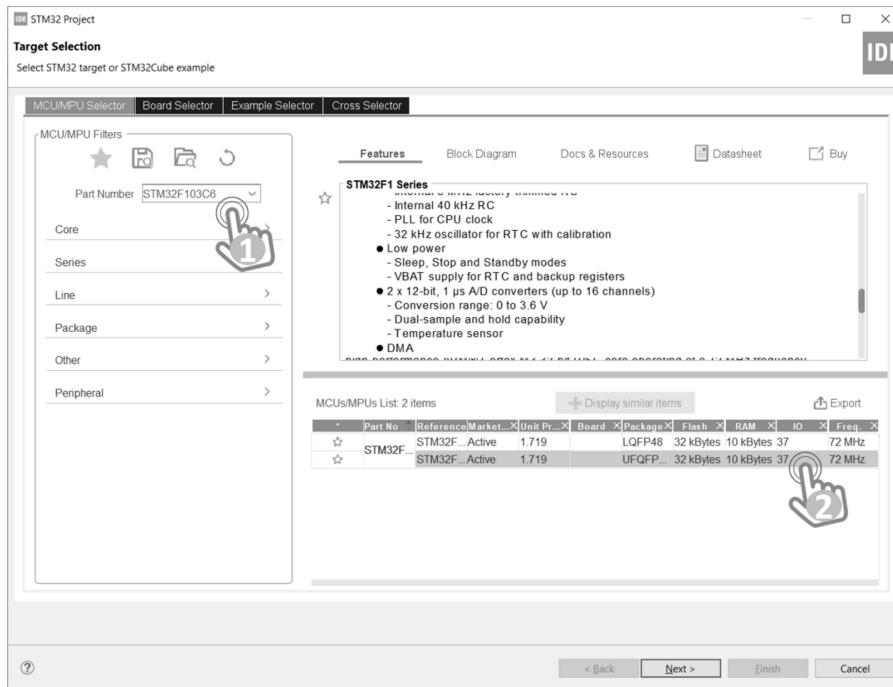
Step 1: Launch STM32CubeIDE, from the menu **File**, select **New**, then chose **STM32 Project**



Hình 1.2: Create a new project on STM32CubeIDE

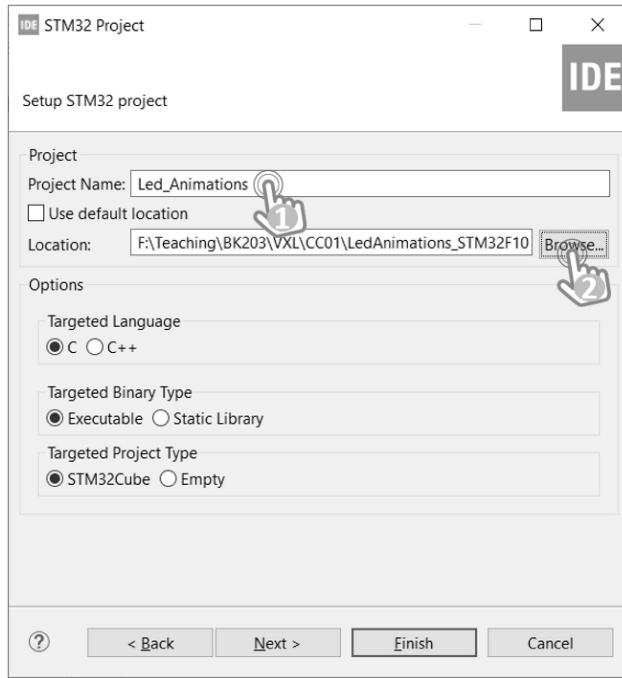
The IDE needs to download some packages, which normally takes time in this first time a project is created.

Step 2: Select the STM32F103C6 in the following dialog, then click on **Next**



Hình 1.3: Select the target device

Step 3: Provide the **Name** and the **Location** for the project.



Hình 1.4: Select the target device

It is important to notice that the **Targeted Project Type** should be **STM32Cube**. In the case this option is disable, step 1 must be repeated. The location path should not contain special characters (e.g. the space). Finally, click on the **Next** button.

Step 4: On the last dialog, just keep the default firmware version and click on **Finish** button.

Step 5: The project is created and the wizard for configuration is display. This utility from CubeIDE can simplify the configuration process for an ARM micro-controller like the STM32.

From the configuration windows, select **Pin configuration**, select the pin **PA5** and set to **GPIO Output** mode, since this pin is connected to an LED in the STM32 development kit.

Step 6: Right click on PA5 and select **Enter user label**, and provide the name for this pin (e.g. **LED_RED**). This step helps programming afterward more memorable.

Finally, save the configuration process by pressing **Ctrl + S** and confirm this step by clicking on **OK** button. The code generation is started.

Step 7: Implement the first blinky project in the main function as follow:

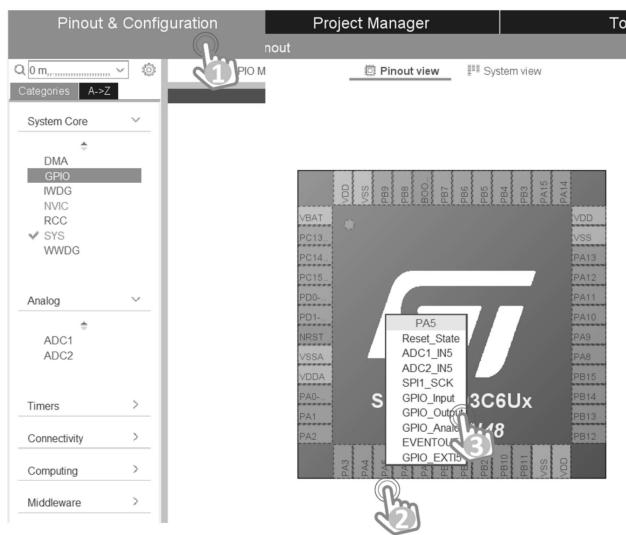
```

1 int main(void)
2 {
3     /* USER CODE BEGIN 1 */
4
5     /* USER CODE END 1 */

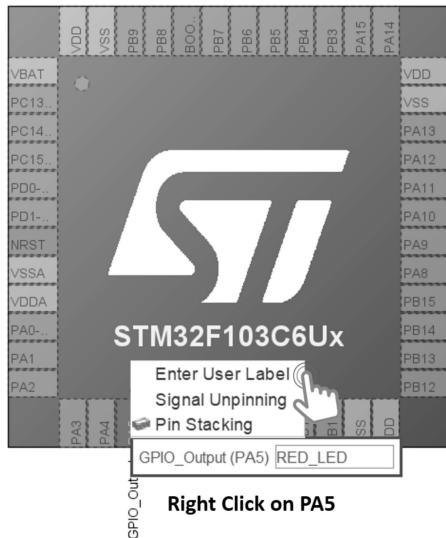
```



Hình 1.5: Keep default firmware version



Hình 1.6: Set PA5 to GPIO Output mode



Hình 1.7: Provide a name for PA5

```

6
7  /* MCU Configuration
8
9  /* Reset of all peripherals, Initializes the Flash
10   interface and the Systick. */
11 HAL_Init();
12
13 /* USER CODE BEGIN Init */
14
15 /* USER CODE END Init */
16
17 /* Configure the system clock */
18 SystemClock_Config();
19
20 /* USER CODE BEGIN SysInit */
21
22 /* USER CODE END SysInit */
23
24 /* Initialize all configured peripherals */
25 MX_GPIO_Init();
26 /* USER CODE BEGIN 2 */
27
28 /* USER CODE END 2 */
29
30 /* Infinite loop */
31 /* USER CODE BEGIN WHILE */
32
33 while (1)
{
```

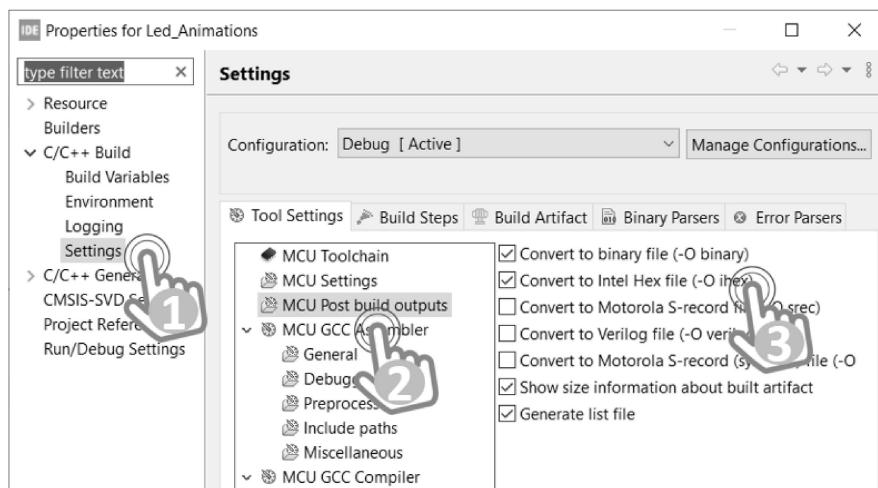
```

34     HAL_GPIO_TogglePin(LED_RED_GPIO_Port, LED_RED_Pin);
35     HAL_Delay(1000);
36     /* USER CODE END WHILE */
37
38     /* USER CODE BEGIN 3 */
39 }
40 /* USER CODE END 3 */
41 }
```

Program 1.1: First blinky LED project

Actually, what is added to the main function is line number 34 and 35. Please put your code in a right place, otherwise it can be deleted when the code is generated (e.g. change the configuration of the project). When coding, frequently use the suggestions by pressing **Ctrl+Space**.

Step 7: Due to the simulation on Proteus, the hex file should be generated from STM32Cube IDE. From menu **Project**, select **Properties** to open the dialog bellow:



Hình 1.8: Config for hex file output

Navigate to **C/C++ Build**, select **Settings**, **MCU Post build outputs**, and check to the **Intel Hex file**.

Step 8: Build the project by clicking on menu **Project** and select **Build Project**. Please check on the output console of the IDE to be sure that the hex file is generated, as follow:

```

22:36:06 **** Incremental Build of configuration Debug for project Led_Animations ****
make -j8 all
arm-none-eabi-size  Led_Animations.elf
      text    data     bss     dec   hex filename
      4596      20    1572    6188   182c Led_Animations.elf
Finished building: default.size.stdout

22:36:06 Build Finished. 0 errors, 0 warnings. (took 272ms)
```

Hình 1.9: Compile the project and generate Hex file

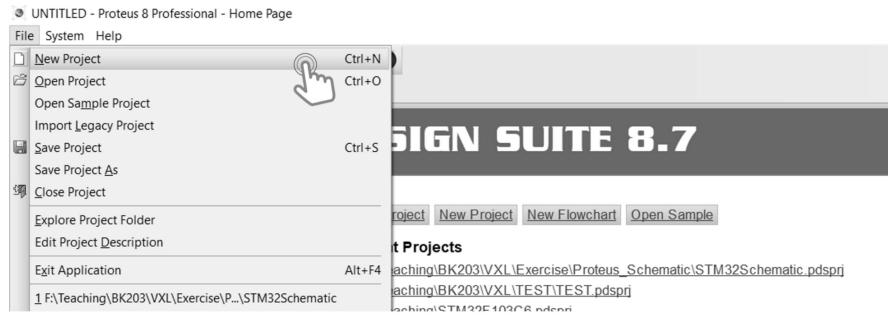
The hex file is located under the **Debug** folder of your project, which is used for the simulation in Proteus afterward. In the case a development kit is connected to your PC, from menu **Run**, select **Run** to download the program to the hardware platform.

In the case there are multiple project in a work-space, double click on the project name to activate this project. Whenever a project is built, check the output files to make sure that you are working in a right project.

3 Simulation on Proteus

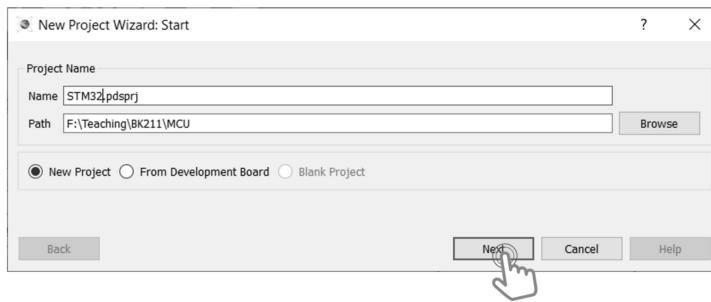
For an online training, a simulation on Proteus can be used. The details to create an STM32 project on Proteus are described below.

Step 1: Launch Proteus (**with administration access**) and from menu **File**, select **New Project**.



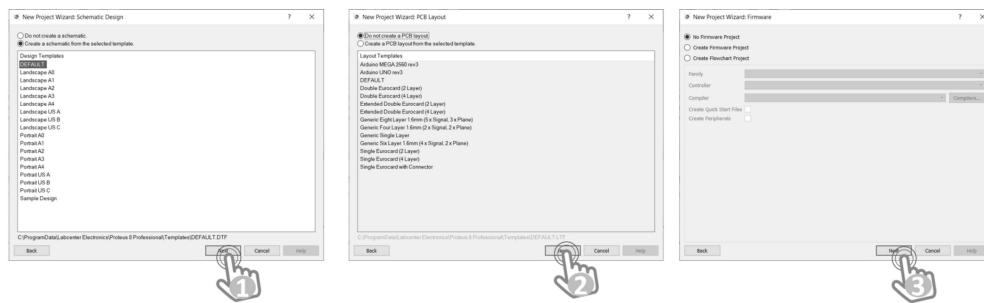
Hình 1.10: Create a new project on Proteus

Step 2: Provide the name and the location of the project, then click on **Next** button.



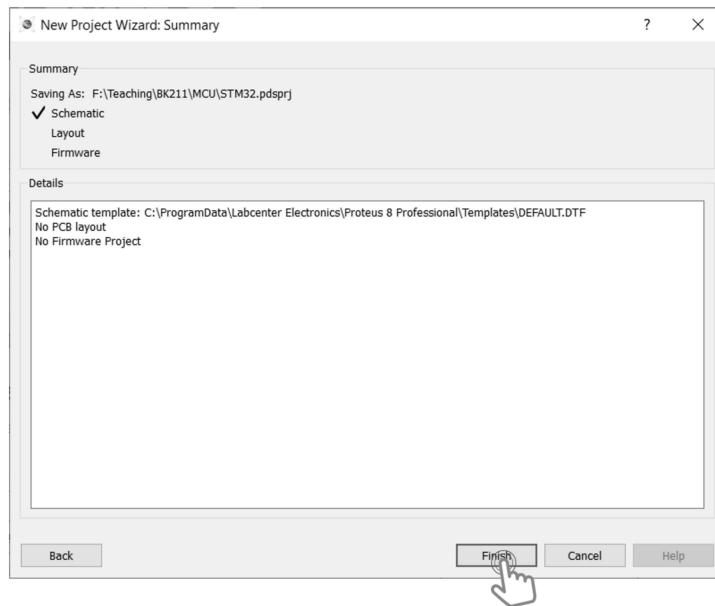
Hình 1.11: Provide project name and location

Step 3: For following dialog, just click on **Next** button as just a schematic is required for the lab.



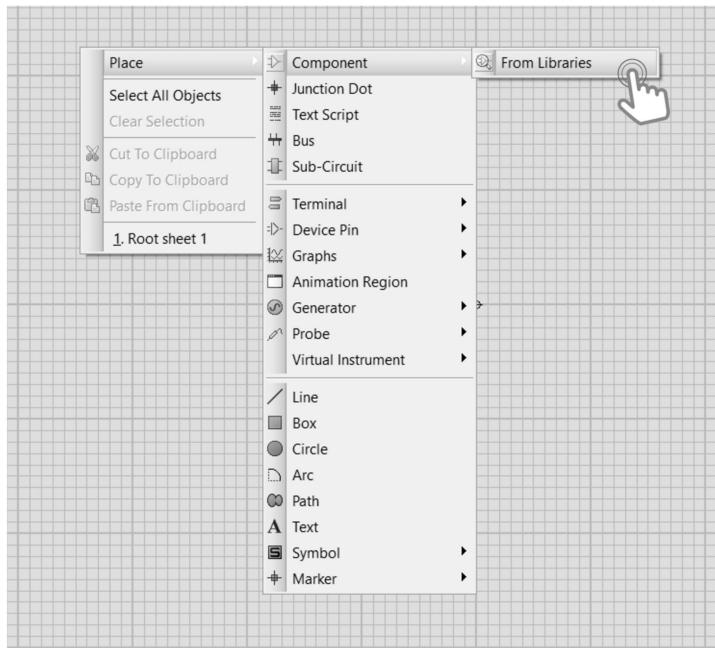
Hình 1.12: Keep the default options by clicking on Next

Step 4: Finally, click on **Finish** button to close the project wizard.



Hình 1.13: Finish the project wizard

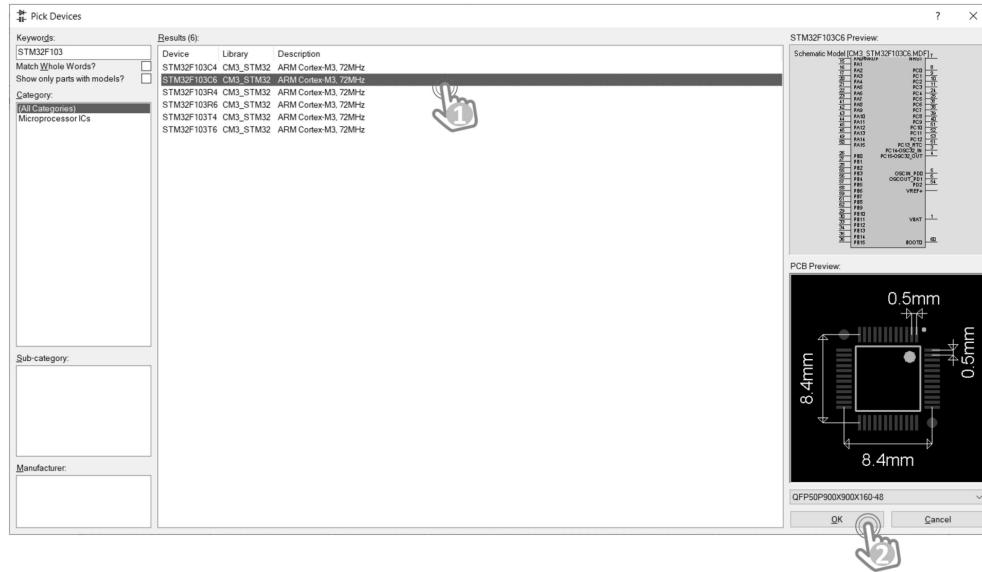
Step 5: On the main page of the project, right click to select **Place, Components, From Libraries**, as follows:



Hình 1.14: Select a component from the library

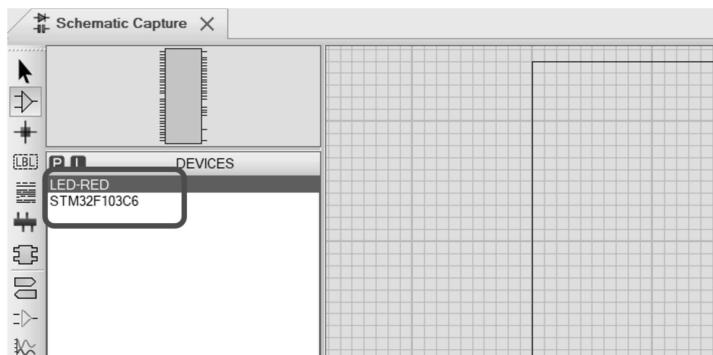
If there is an error with no library found, please restart the Proteus software with Run as administrator option.

Step 6: From the list of components in the library, select STM32F103C6, as follows:



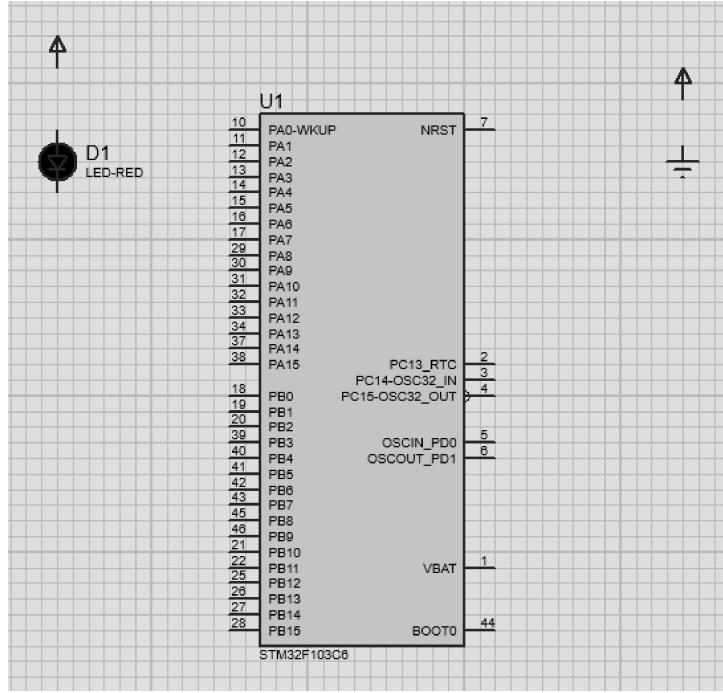
Hình 1.15: Select STM32F103C6

Repeat step 5 and 6 to select an LED, named **LED-RED** in Proteus. Finally, these components are appeared on the DEVICES windows, which is on left hand side as follows:



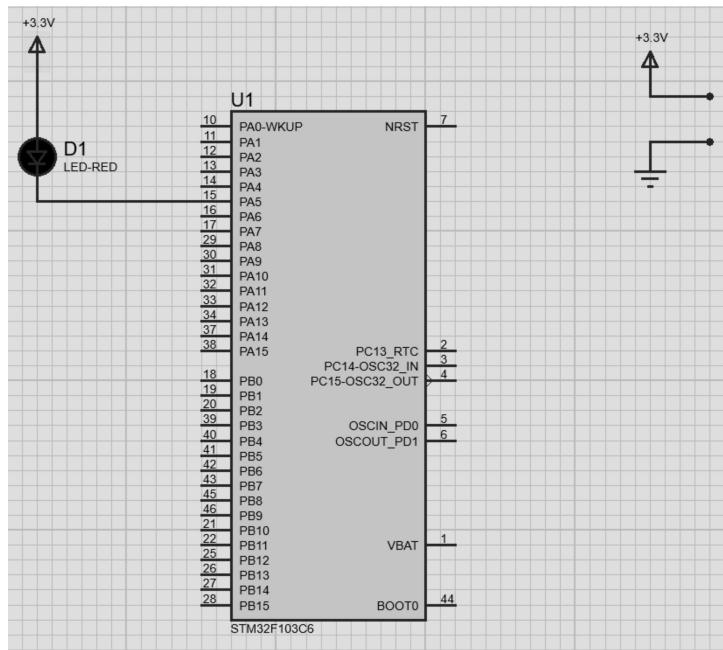
Hình 1.16: STM32 and an LED in the project

Step 7: Place the components to the project: right click on the main page, select on **Place, Component**, and select device added in Step 6. To add the Power and the Ground, right click on the main page, select on **Place, Terminal**. The result in this step is expected as follows:



Hình 1.17: Place components to the project

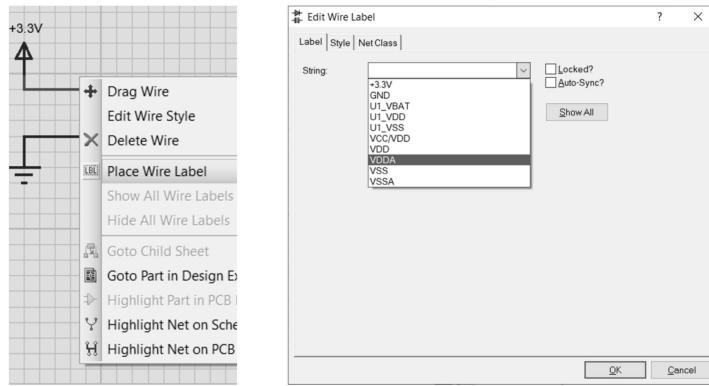
Step 8: Start wiring the circuit. The negative pin of the LED is connected to PA5 while its positive pin is connected to the power supply. For the power and the ground on the right, just make a short wire, which will labeled in the next step.



Hình 1.18: Connect components and set the power to 3.3V

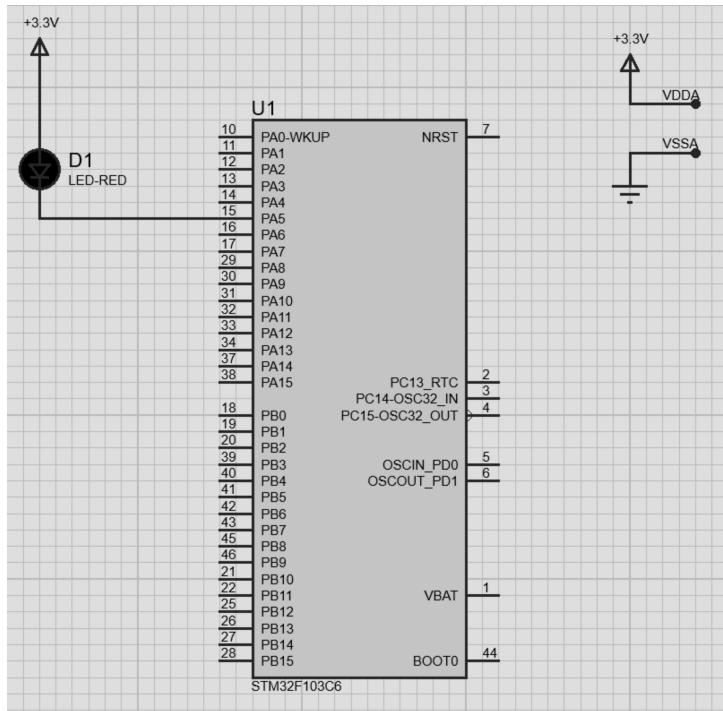
In this step, also double click on the power supply in order to provide the String property to **+3.3V**.

Step 8: Right click on the wire of the power supply and the ground, and select **Place wire Label**



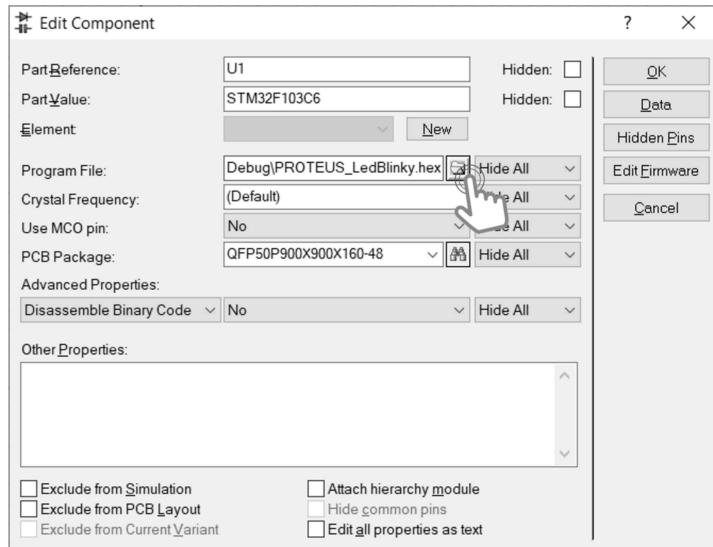
Hình 1.19: Place label for Power and Ground

This step is required as VDDA and VSSA of the STM32 must be connected to provide the reference voltage. Therefore, VDDA is connected to 3.3V, while the VSSA is connected to the Ground. Finally, the image of our schematic is shown bellow:



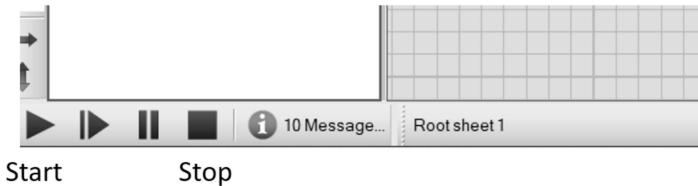
Hình 1.20: Finalize the schematic

Step 9: Double click on the STM32, and set the **Program File** to the Hex file, which is generated from Cube IDE, as following:



Hình 1.21: Set the program of the STM32 to the hex file from Cube IDE

From now, the simulation is ready to start by clicking on the menu **Debug**, and select on **Run simulation**. To stop the simulation, click on **Debug** and select **Stop VMS Debugging**. Moreover, there are some quick access bottom on the left corner of the Proteus to start or stop the simulation, as shown following:



Hình 1.22: Quick access buttons to start and stop the simulation

If everything is success, students can see the LED is blinking every second. Please stop the simulation before updating the project, either in Proteus or STM32Cube IDE. However, the step 9 (set the program file for STM32 in Proteus) is required to do once. Beside the toggle instruction, student can set or reset a pin as following:

```

1 while (1){
2     HAL_GPIO_WritePin(LED_RED_GPIO_Port , LED_RED_Pin ,
3         GPIO_PIN_SET);
4     HAL_Delay(1000);
5     HAL_GPIO_WritePin(LED_RED_GPIO_Port , LED_RED_Pin ,
6         GPIO_PIN_RESET);
7     HAL_Delay(1000);
8 }
```

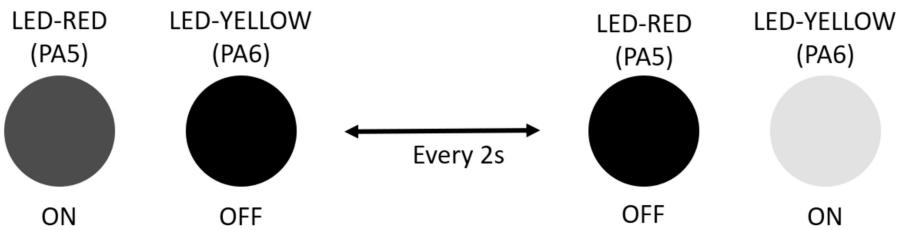
Program 1.2: An example for LED blinky

4 Exercise and Report

4.1 Exercise 1

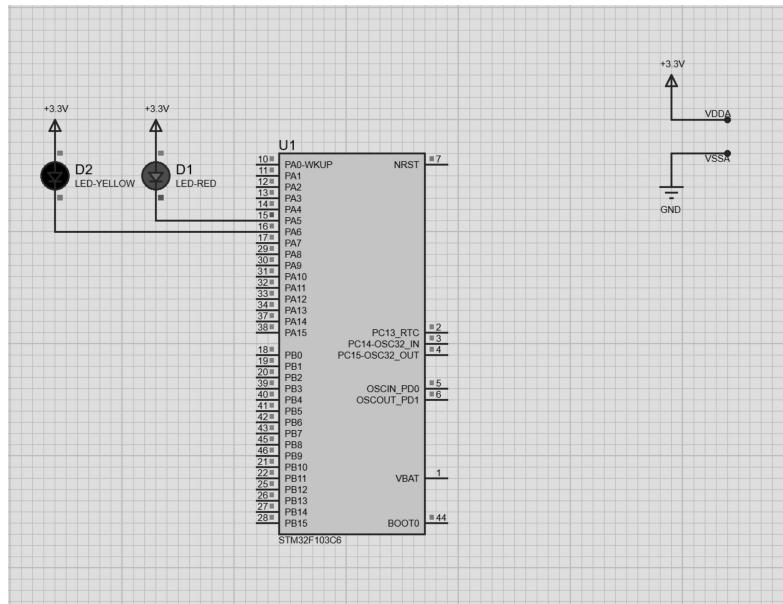
From the simulation on Proteus, one more LED is connected to pin **PA6** of the STM32 (negative pin of the LED is connected to PA6). The component suggested in this exercise is **LED-YELLOW**, which can be found from the device list.

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



Hình 1.23: State transitions for 2 LEDs

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



Hình 1.24: State transitions for 2 LEDs

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

```

1   int flag = 0;
2   while (1)
3   {
4       if (flag < 2)
5       {
6           HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port,
7 YELLOW_LED_Pin, SET);
8           HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin,
9 RESET);
10          flag++;
11      }
12      else
13      {
14          HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port,
15 YELLOW_LED_Pin, RESET);
16          HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin, SET
17 );
18          flag++;
19          if (flag == 4) flag = 0;
20      }
21      /* USER CODE END 3 */
22  }

```

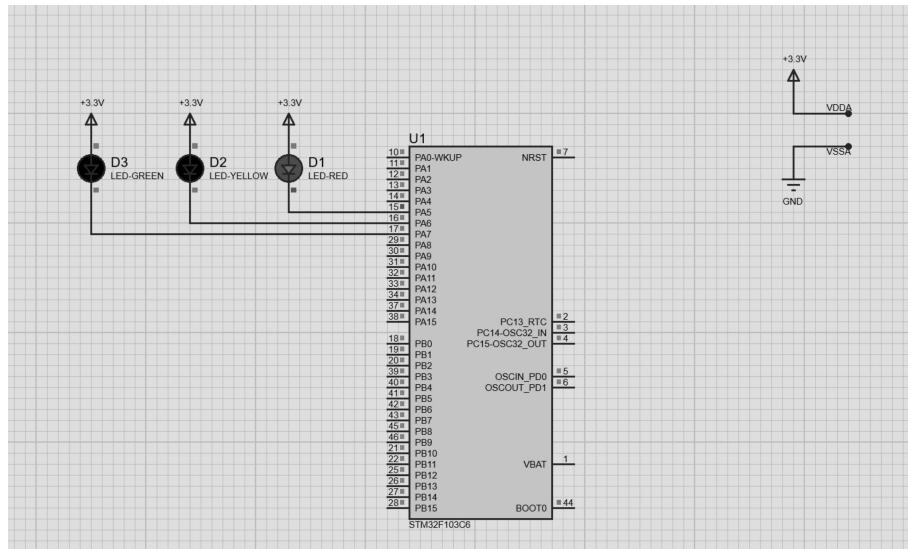
Program 1.3: Coding

4.2 Exercise 2

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

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

Report 1: Present the schematic.



Hình 1.25: State transitions for 3 LEDs

Report 2: Present the source code in while.

```

1   int flag_red = 0;
2   int flag_yellow = 0;
3   int flag_green = 0;
4   while (1)
5   {
6     //Red Led turns on
7     if (flag_red <= 5 && flag_yellow == 0 && flag_green == 0)
8     {
9       HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin, RESET);
10      HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port, YELLOW_LED_Pin, SET);
11      HAL_GPIO_WritePin(GREEN_LED_GPIO_Port, GREEN_LED_Pin, SET);
12      flag_red++;
13    }
14
15    //Green Led turns on
16    if (flag_red > 5 && flag_yellow == 0 && flag_green <=3)
17    {
18      HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin, SET);
19      HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port, YELLOW_LED_Pin, SET);
20      HAL_GPIO_WritePin(GREEN_LED_GPIO_Port, GREEN_LED_Pin, RESET);
21      flag_green++;
22    }
23

```

```

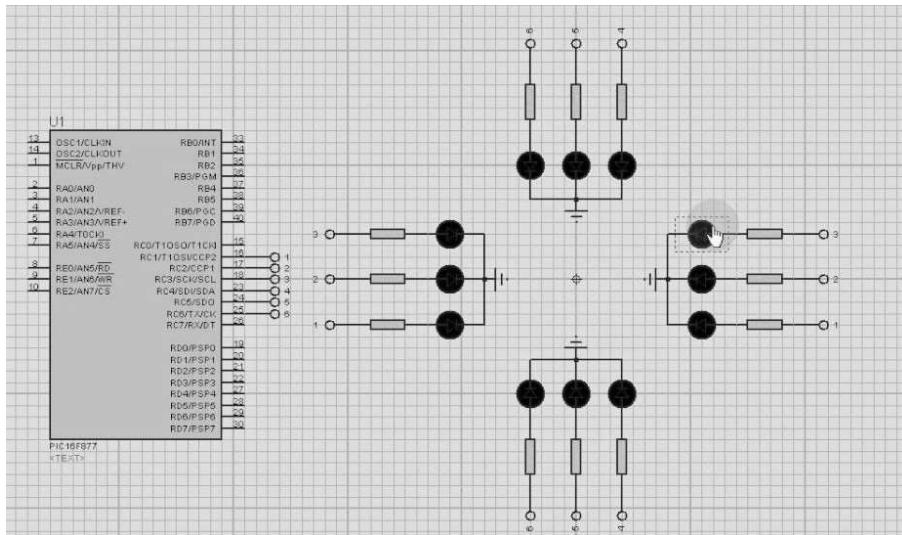
24 //Yellow Led turns on
25 if (flag_red > 5 && flag_yellow <= 2 && flag_green > 3)
26 {
27     HAL_GPIO_WritePin(RED_LED_GPIO_Port, RED_LED_Pin, SET
28 );
29     HAL_GPIO_WritePin(YELLOW_LED_GPIO_Port,
30 YELLOW_LED_Pin, RESET);
31     HAL_GPIO_WritePin(GREEN_LED_GPIO_Port, GREEN_LED_Pin,
32 SET);
33     flag_yellow++;
34     if (flag_red > 5 && flag_yellow > 2 && flag_green >
35 3)
36     {
37         flag_red = 0;
38         flag_yellow = 0;
39         flag_green = 0;
40     }
41 }
42 HAL_Delay(1000);
43 }

```

Program 1.4: Coding

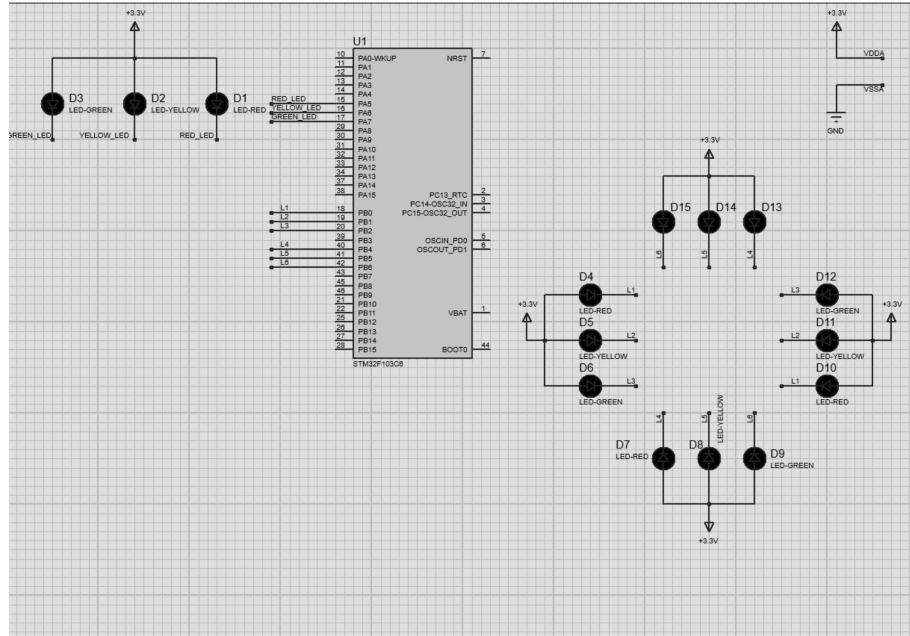
4.3 Exercise 3

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.



Hình 1.26: Reference design for a 4 way traffic light

Report 1: Present the schematic.



Hình 1.27: State transitions for 3 LEDs

Report 2: Present the source code in while.

```

1 #define RED      0
2 #define YELLOW   1
3 #define GREEN    2
4
5 int flag = 0;
6 int led_color = RED;
7
8 while (1)
9 {
10     switch(led_color)
11     {
12         case RED:
13             HAL_GPIO_WritePin(RED_1_GPIO_Port, RED_1_Pin,
14             RESET);
15             HAL_GPIO_WritePin(YELLOW_1_GPIO_Port,
16             YELLOW_1_Pin, SET);
17             HAL_GPIO_WritePin(GREEN_1_GPIO_Port,
18             GREEN_1_Pin, SET);
19
20             HAL_GPIO_WritePin(RED_2_GPIO_Port, RED_2_Pin,
21             SET);
22             HAL_GPIO_WritePin(YELLOW_2_GPIO_Port,
23             YELLOW_2_Pin, SET);
24             HAL_GPIO_WritePin(GREEN_2_GPIO_Port,
25             GREEN_2_Pin, RESET);

```

```

20         flag++;
21
22         if(flag > 3)
23         {
24             HAL_GPIO_WritePin(RED_2_GPIO_Port,
25 RED_2_Pin, SET);
26             HAL_GPIO_WritePin(YELLOW_2_GPIO_Port,
27 YELLOW_2_Pin, RESET);
28             HAL_GPIO_WritePin(GREEN_2_GPIO_Port,
29 GREEN_2_Pin, SET);
30         }
31
32         if (flag == 5)
33         {
34             led_color = GREEN;
35             flag = 0;
36         }
37         break;
38
39     case YELLOW:
40         HAL_GPIO_WritePin(RED_1_GPIO_Port, RED_1_Pin,
41 SET);
42         HAL_GPIO_WritePin(YELLOW_1_GPIO_Port,
43 YELLOW_1_Pin, RESET);
44         HAL_GPIO_WritePin(GREEN_1_GPIO_Port,
45 GREEN_1_Pin, SET);
46
47         HAL_GPIO_WritePin(RED_2_GPIO_Port, RED_2_Pin,
48 RESET);
49         HAL_GPIO_WritePin(YELLOW_2_GPIO_Port,
50 YELLOW_2_Pin, SET);
51         HAL_GPIO_WritePin(GREEN_2_GPIO_Port,
52 GREEN_2_Pin, SET);
53         flag++;
54
55         if (flag == 2)
56         {
57             led_color = RED;
58             flag = 0;
59         }
60         break;
61
62     case GREEN:
63         HAL_GPIO_WritePin(RED_1_GPIO_Port, RED_1_Pin,
64 SET);
65         HAL_GPIO_WritePin(YELLOW_1_GPIO_Port,
66 YELLOW_1_Pin, SET);
67         HAL_GPIO_WritePin(GREEN_1_GPIO_Port,
68 GREEN_1_Pin, RESET);

```

```

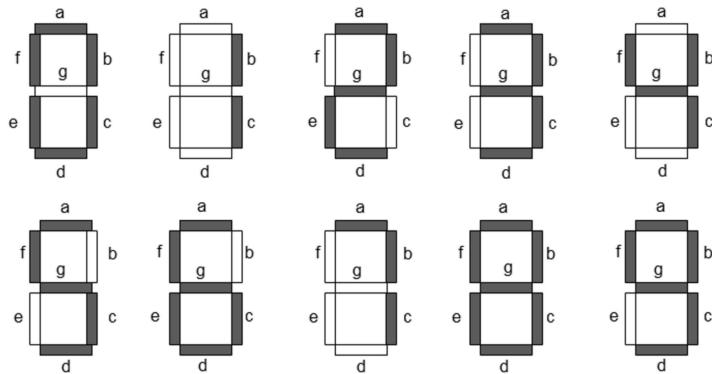
57
58     HAL_GPIO_WritePin(RED_2_GPIO_Port , RED_2_Pin ,
59     RESET) ;
60     HAL_GPIO_WritePin(YELLOW_2_GPIO_Port ,
61     YELLOW_2_Pin , SET) ;
62     HAL_GPIO_WritePin(GREEN_2_GPIO_Port ,
63     GREEN_2_Pin , SET) ;
64     flag++ ;
65
66     if (flag == 3)
67     {
68         led_color = YELLOW ;
69         flag = 0 ;
70     }
71     break ;
72 }
73 HAL_Delay(1000) ;
74 }
```

Program 1.5: Coding

4.4 Exercise 4

Add **only one 7 led segment** to the schematic in Exercise 3. This component can be found in Proteus by the keyword **7SEG-COM-ANODE**. For this device, the common pin should be connected to the power supply and other pins are supposed to 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 outputs are listed as following:



Hình 1.28: Display a number on 7 segment LED

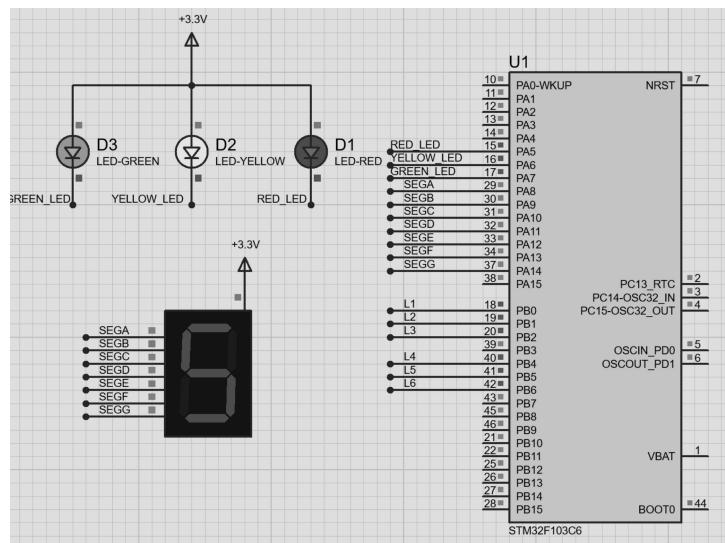
This function is invoked in the while loop for testing as following:

```

1 int counter = 0;
2 while (1){
3     if(counter >= 10) counter = 0;
4     display7SEG(counter++);
5     HAL_Delay(1000);
6
7 }
```

Program 1.6: An example for your source code

Report 1: Present the schematic.



Hình 1.29: Display a number on 7 segment LED

Report 2: Present the source code for display7SEG function.

```

1 #include "7SEG.h"
2 int counter = 0;
3 void display_7SEG (int num)
4 {
5     //Turn off all of the segment first
6     HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin, SET);
7     HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin, SET);
8     HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin, SET);
9     HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET);
10    HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET);
11    HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET);
12    HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET);
13
14    switch(counter)
15    {
16        case 0:
```

```

17         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
18             RESET);
19         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
20             RESET);
21         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
22             RESET);
23         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
24             RESET);
25         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin,
26             RESET);
27         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
28             RESET);
29         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET
30     );
31         break;
32
33     case 1:
34         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin, SET
35     );
36         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
37             RESET);
38         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
39             RESET);
40         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET
41     );
42         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
43     );
44         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
45     );
46         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET
47     );
48         break;
49
50     case 2:
51         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
52             RESET);
53         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
54             RESET);
55         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin, SET
56     );
57         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
58             RESET);
59         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin,
60             RESET);
61         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
62     );
63         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
64             RESET);
65         break;

```

```

45
46     case 3:
47         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
48             RESET);
48         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
49             RESET);
49         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
50             RESET);
50         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
51             RESET);
51         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
52 );
52         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
53 );
53         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
54             RESET);
54         break;
55
56     case 4:
57         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin, SET
58 );
58         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
59             RESET);
59         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
60             RESET);
60         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET
61 );
61         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
62 );
62         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
63             RESET);
63         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
64             RESET);
64         break;
65
66     case 5:
67         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
68             RESET);
68         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin, SET
69 );
69         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
70             RESET);
70         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
71             RESET);
71         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
72 );
72         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
73             RESET);
73         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,

```

```

        RESET);
        break;

    case 6:
        HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin, SET
    );
        HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
        RESET);
        break;

    case 7:
        HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET
    );
        HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
    );
        HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
    );
        HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET
    );
        break;

    case 8:
        HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin,
        RESET);
        HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,

```

```

        RESET);
103    HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
RESET);
104    break;
105
106    case 9:
107        HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
RESET);
108        HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
RESET);
109        HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
RESET);
110        HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
RESET);
111        HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
);
112        HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
RESET);
113        HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
RESET);
114    break;
115
116    default:
117        break;
118    }
119
120}
121
122/* USER CODE BEGIN WHILE */
123while (1)
124{
125    if (counter >= 10) counter = 0;
126    display_7SEG(counter++);
127    HAL_Delay(1000);
128}
129/* USER CODE END 3 */
130}

```

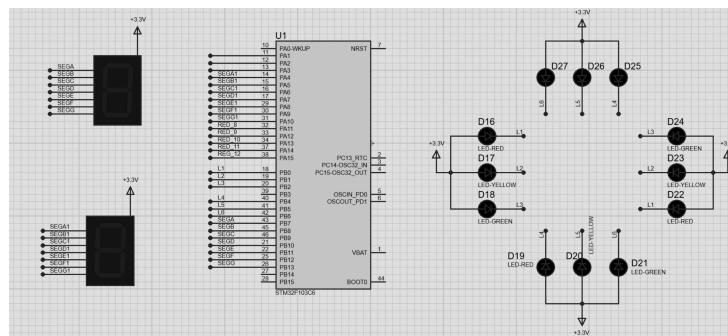
Program 1.7: An example for your source code

4.5 Exercise 5

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

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

Report 1: Present the schematic.



Hình 1.30: Display a number on 7 segment LED

Report 2: Present the source code for display7SEG function.

```

1 void display_7SEG ( int num )
2 {
3     //Turn off all of the segment first
4     HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin, SET);
5     HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin, SET);
6     HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin, SET);
7     HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET);
8     HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET);
9     HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET);
10    HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET);
11
12    switch(num)
13    {
14        case 0:
15            HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
16 RESET);
16            HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
17 RESET);
17            HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
18 RESET);
18            HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
19 RESET);
19            HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin,
20 RESET);
20            HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
21 RESET);
21            HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET
22 );
22            break;
23
24        case 1:

```

```

25         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin, SET
26     );
27         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
28     RESET);
28         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
29     RESET);
29         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET
30     );
30         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
31     );
31         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
32     );
32         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET
33     );
33         break;
34
34     case 2:
35         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
36     RESET);
36         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
37     RESET);
37         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin, SET
38     );
38         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
39     RESET);
39         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin,
40     RESET);
40         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
41     );
41         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
42     RESET);
42         break;
43
43     case 3:
44         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
45     RESET);
45         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
46     RESET);
46         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
47     RESET);
47         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
48     RESET);
48         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
49     );
49         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
50     );
50         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
51     RESET);
51         break;

```

```

53
54     case 4:
55         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin, SET
56 );
57         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
RESET);
58         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
RESET);
59         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET
60 );
61         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
62 );
63         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
RESET);
64         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
RESET);
65         break;
66
67     case 5:
68         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
RESET);
69         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin, SET
70 );
71         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
RESET);
72         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
RESET);
73         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
74 );
75         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
RESET);
76         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
RESET);
77         break;
78
79     case 6:
80         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
RESET);
81         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin, SET
82 );
83         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
RESET);
84         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
RESET);
85         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin,
RESET);
86         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,
RESET);
87         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,

```

```

        RESET);
        break;

83
84     case 7:
85         HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
86             RESET);
87         HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
88             RESET);
89         HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
90             RESET);
91         HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin, SET
92 );
93         HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
94 );
95         HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin, SET
96 );
97         HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin, SET
98 );
99         break;

100
101    case 8:
102        HAL_GPIO_WritePin(SEG_A_GPIO_Port, SEG_A_Pin,
103             RESET);
104        HAL_GPIO_WritePin(SEG_B_GPIO_Port, SEG_B_Pin,
105             RESET);
106        HAL_GPIO_WritePin(SEG_C_GPIO_Port, SEG_C_Pin,
107             RESET);
108        HAL_GPIO_WritePin(SEG_D_GPIO_Port, SEG_D_Pin,
109             RESET);
110        HAL_GPIO_WritePin(SEG_E_GPIO_Port, SEG_E_Pin, SET
111 );
112        HAL_GPIO_WritePin(SEG_F_GPIO_Port, SEG_F_Pin,

```

```

        RESET);
111     HAL_GPIO_WritePin(SEG_G_GPIO_Port, SEG_G_Pin,
112     RESET);
113     break;
114
115     default:
116     break;
117 }
118 }
119
120 void display_7SEG1 (int num1)
121 {
122     //Turn off all of the segment first
123     HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin, SET);
124     HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin, SET);
125     HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin, SET);
126     HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin, SET);
127     HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin, SET);
128     HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGFF1_Pin, SET);
129     HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin, SET);
130
131     switch(num1)
132     {
133         case 0:
134             HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin,
135             RESET);
136             HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
137             RESET);
138             HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
139             RESET);
140             HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin,
141             RESET);
142             HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin,
143             RESET);
144             HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGFF1_Pin,
145             RESET);
146             HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin, SET
147 );
148         break;
149
150         case 1:
151             HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin, SET
152 );
153             HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
154             RESET);
155             HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
156             RESET);
157             HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin, SET
158 );
159     }
160 }
```

```

    );
148     HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin, SET
    );
149     HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGFI_Pin, SET
    );
150     HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin, SET
    );
151     break;
152
153
154     case 2:
155         HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin,
156             RESET);
156         HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
157             RESET);
157         HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin, SET
    );
158         HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin,
159             RESET);
159         HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin,
160             RESET);
160         HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGFI_Pin, SET
    );
161         HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin,
162             RESET);
162         break;
163
164     case 3:
165         HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin,
166             RESET);
166         HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
167             RESET);
167         HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
168             RESET);
168         HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin,
169             RESET);
169         HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin, SET
    );
170         HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGFI_Pin, SET
    );
171         HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin,
172             RESET);
172         break;
173
174     case 4:
175         HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin, SET
    );
176         HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
176             RESET);

```

```

177         HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
178             RESET);
179         HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin, SET
180 );
181         HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin, SET
182 );
183         HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGF1_Pin,
184             RESET);
185         HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin,
186             RESET);
187         break;
188
189     case 5:
190         HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin,
191             RESET);
192         HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin, SET
193 );
194         HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
195             RESET);
196         HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin,
197             RESET);
198         HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin, SET
199 );
200         HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGF1_Pin,
201             RESET);
202         HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin,
203             RESET);
204         break;
205
206     case 7:
207         HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin,

```

```

        RESET);
207      HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
        RESET);
208      HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
        RESET);
209      HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin, SET
    );
210      HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin, SET
    );
211      HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGF1_Pin, SET
    );
212      HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin, SET
    );
213      break;
214
215      case 8:
216          HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin,
        RESET);
217          HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
        RESET);
218          HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
        RESET);
219          HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin,
        RESET);
220          HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin,
        RESET);
221          HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGF1_Pin,
        RESET);
222          HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin,
        RESET);
223          break;
224
225      case 9:
226          HAL_GPIO_WritePin(SEGA1_GPIO_Port, SEGA1_Pin,
        RESET);
227          HAL_GPIO_WritePin(SEGB1_GPIO_Port, SEGB1_Pin,
        RESET);
228          HAL_GPIO_WritePin(SEGC1_GPIO_Port, SEGC1_Pin,
        RESET);
229          HAL_GPIO_WritePin(SEGD1_GPIO_Port, SEGD1_Pin,
        RESET);
230          HAL_GPIO_WritePin(SEGE1_GPIO_Port, SEGE1_Pin, SET
    );
231          HAL_GPIO_WritePin(SEGF1_GPIO_Port, SEGF1_Pin,
        RESET);
232          HAL_GPIO_WritePin(SEGG1_GPIO_Port, SEGG1_Pin,
        RESET);
233          break;
234

```

```

235         default:
236             break;
237     }
238 }
239 }
240
241
242 #define RED      0
243 #define YELLOW   1
244 #define GREEN   2
245
246 int flag_1 = 0;
247 int flag_2 = 0;
248 int countdown_1 = 0;
249 int countdown_2 = 0;
250 int led_color_1 = RED;
251 int led_color_2 = GREEN;
252
253 while (1)
254 {
255     // B       n      1
256     switch(led_color_1) // Kim tra trang thi ca
257     {
258         case RED:
259             HAL_GPIO_WritePin(RED_1_GPIO_Port, RED_1_Pin,
260             RESET);
261             HAL_GPIO_WritePin(YELLOW_1_GPIO_Port,
262             YELLOW_1_Pin, SET);
263             HAL_GPIO_WritePin(GREEN_1_GPIO_Port,
264             GREEN_1_Pin, SET);
265
266             countdown_1 = 5 - flag_1;
267             display_7SEG(countdown_1);
268
269             flag_1++;
270             if(flag_1 == 5)
271             {
272                 led_color_1 = GREEN;
273                 flag_1 = 0;
274             }
275             break;
276
277         case YELLOW:
278             HAL_GPIO_WritePin(RED_1_GPIO_Port, RED_1_Pin,
279             SET);
280             HAL_GPIO_WritePin(YELLOW_1_GPIO_Port,
281             YELLOW_1_Pin, RESET);
282             HAL_GPIO_WritePin(GREEN_1_GPIO_Port,

```

```

        GREEN_1_Pin , SET);

278
279     countdown_1 = 2 - flag_1;
280     display_7SEG(countdown_1);

281
282     flag_1++;
283     if (flag_1 == 2)
284     {
285         led_color_1 = RED;
286         flag_1 = 0;
287     }
288     break;

289
290     case GREEN:
291         HAL_GPIO_WritePin(RED_1_GPIO_Port , RED_1_Pin ,
SET);
292         HAL_GPIO_WritePin(YELLOW_1_GPIO_Port ,
YELLOW_1_Pin , SET);
293         HAL_GPIO_WritePin(GREEN_1_GPIO_Port ,
GREEN_1_Pin , RESET);

294         countdown_1 = 3 - flag_1;
295         display_7SEG(countdown_1);

296
297         flag_1++;
298         if (flag_1 == 3)
299         {
300             led_color_1 = YELLOW;
301             flag_1 = 0;
302         }
303         break;

304
305     default:
306         break;
307     }

308 }

309
310
311     switch(led_color_2)
312     {
313
314         case RED:
315             HAL_GPIO_WritePin(RED_2_GPIO_Port , RED_2_Pin ,
RESET);
316             HAL_GPIO_WritePin(YELLOW_2_GPIO_Port ,
YELLOW_2_Pin , SET);
317             HAL_GPIO_WritePin(GREEN_2_GPIO_Port ,
GREEN_2_Pin , SET);

318             countdown_2 = 5 - flag_2;
319             display_7SEG1(countdown_2);

```

```

320
321     flag_2++;
322     if(flag_2 == 5)
323     {
324         led_color_2 = GREEN;
325         flag_2 = 0;
326     }
327     break;
328
329     case YELLOW:
330         HAL_GPIO_WritePin(RED_2_GPIO_Port, RED_2_Pin,
331                         SET);
331         HAL_GPIO_WritePin(YELLOW_2_GPIO_Port,
332                         YELLOW_2_Pin, RESET);
332         HAL_GPIO_WritePin(GREEN_2_GPIO_Port,
333                         GREEN_2_Pin, SET);
334
335         countdown_2 = 2 - flag_2;
336         display_7SEG1(countdown_2);
337
338         flag_2++;
339         if (flag_2 == 2)
340         {
341             led_color_2 = RED;
342             flag_2 = 0;
343         }
344         break;
345
346     case GREEN:
347         HAL_GPIO_WritePin(RED_2_GPIO_Port, RED_2_Pin,
348                         SET);
348         HAL_GPIO_WritePin(YELLOW_2_GPIO_Port,
349                         YELLOW_2_Pin, SET);
350         HAL_GPIO_WritePin(GREEN_2_GPIO_Port,
351                         GREEN_2_Pin, RESET);
352
353         countdown_2 = 3 - flag_2;
354         display_7SEG1(countdown_2);
355
356         flag_2++;
357         if (flag_2 == 3)
358         {
359             led_color_2 = YELLOW;
360             flag_2 = 0;
361         }
362         break;
363
364     default:
365         break;

```

```

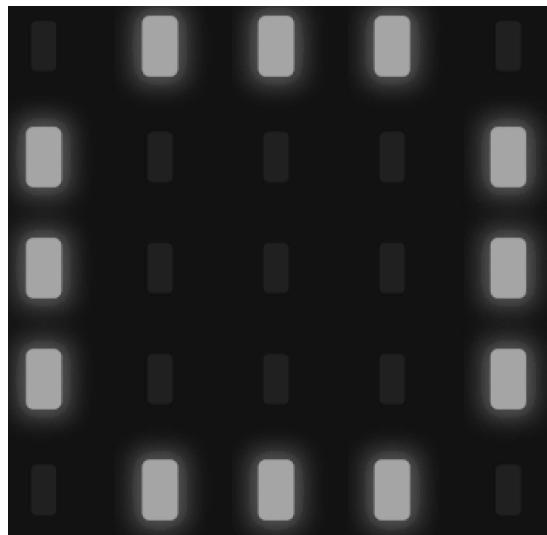
363 }
364
365 HAL_Delay(1000);

```

Program 1.8: Source code

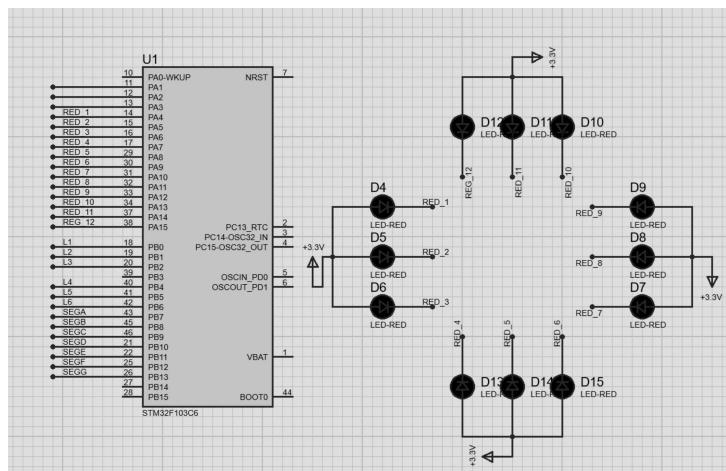
4.6 Exercise 6

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.



Hình 1.31: 12 LEDs for an analog clock

Report 1: Present the schematic.



Hình 1.32: Display a number on 7 segment LED

Report 2: Present the source code

```
1 int main(void)
2 {
3     HAL_Init();
4     SystemClock_Config();
5     MX_GPIO_Init();
6     while (1)
7     {
8         for (int pin = 4; pin <= 15; pin++)
9         {
10             // Turn on the current LED (PA pin)
11             HAL_GPIO_WritePin(GPIOA, 1 << pin, GPIO_PIN_RESET);
12         }
13         HAL_Delay(1000); // 1s delay to observe the LED
14         // Turn off the current LED
15         HAL_GPIO_WritePin(GPIOA, 1 << pin, GPIO_PIN_SET);
16     }
17 }
18 }

19
20
21
22 static void MX_GPIO_Init(void)
23 {
24     GPIO_InitTypeDef GPIO_InitStruct = {0};
25     /* USER CODE BEGIN MX_GPIO_Init_1 */
26     /* USER CODE END MX_GPIO_Init_1 */

27     /* GPIO Ports Clock Enable */
28     __HAL_RCC_GPIOA_CLK_ENABLE();

29
30     /*Configure GPIO pin Output Level */
31     HAL_GPIO_WritePin(GPIOA, RED_1A_Pin|RED_2A_Pin|RED_3_Pin|
32     RED_4_Pin
33                                     |RED_5_Pin|RED_6_Pin|RED_7_Pin|
34     RED_8_Pin
35                                     |RED_9_Pin|RED_10_Pin|RED_11_Pin|
36     RED_12_Pin, GPIO_PIN_SET); //Initial State is OFF.

37
38     /*Configure GPIO pins : RED_1A_Pin RED_2A_Pin RED_3_Pin
39     RED_4_Pin
40             RED_5_Pin RED_6_Pin RED_7_Pin
41             RED_8_Pin
42             RED_9_Pin RED_10_Pin RED_11_Pin
43             RED_12_Pin */
44     GPIO_InitStruct.Pin = RED_1A_Pin|RED_2A_Pin|RED_3_Pin|
45     RED_4_Pin
46                                     |RED_5_Pin|RED_6_Pin|RED_7_Pin|
```

```

    RED_8_Pin
41                                | RED_9_Pin|RED_10_Pin|RED_11_Pin|
    RED_12_Pin;
42 GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
43 GPIO_InitStruct.Pull = GPIO_NOPULL;
44 GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
45 HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
46
47 /* USER CODE BEGIN MX_GPIO_Init_2 */
48 /* USER CODE END MX_GPIO_Init_2 */
49 }

```

Program 1.9: Source code

4.7 Exercise 7

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

```

1 void clearAllClock()
2 {
3     for (int pin = 4; pin <= 15; pin++)
4     {
5         // Turn off the current LED (PA pin)
6         HAL_GPIO_WritePin(GPIOA, 1 << pin,
7             GPIO_PIN_SET);
8         HAL_Delay(1000); // 1s delay to observe the
9         LED
10    }
11 }
12
13 while (1)
14 {
15     clearAllClock();
16 }

```

Program 1.10: Function Implementation

4.8 Exercise 8

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

```

1     int pin = 4;
2     void SetNumberOnClock(int num)
3     {

```

```

4     pin = 4 + num; // Map num to the corresponding GPIO
5     pin (PA4 to PA15)
6     HAL_GPIO_WritePin(GPIOA, 1 << pin, GPIO_PIN_RESET);
// Turn on the corresponding LED
7     HAL_Delay(1000); // Delay to observe the LED
8 }
9 while (1)
{
10    for (int num = 0; num < 12; num++)
11    {
12        SetNumberOnClock(num);
13    }
14}

```

Program 1.11: Function Implementation

4.9 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     int pin = 4;
2     void clearNumberOnClock(int num)
3     {
4         pin = 4 + num; // Map num to the corresponding GPIO
5         pin (PA4 to PA15)
6         HAL_GPIO_WritePin(GPIOA, 1 << pin, GPIO_PIN_SET); // 
Turn off the corresponding LED
7     }
8     while (1)
9     {
10        for (int num = 0; num < 12; num++)
11        {
12            clearNumberOnClock(num);
13        }
14    }

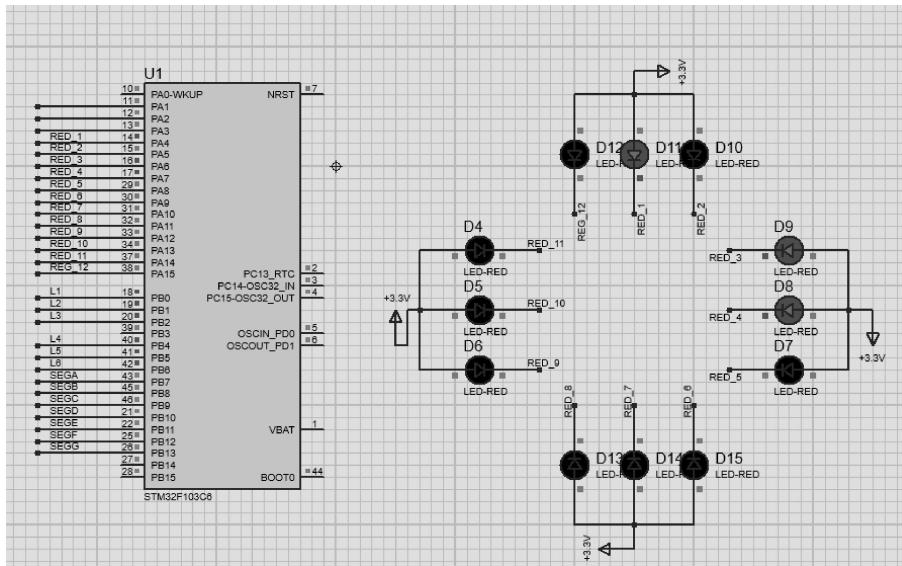
```

Program 1.12: Function Implementation

4.10 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.

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



Hình 1.33: State transitions for 2 LEDs

Report 2: Present the source code

```

1
2
3
4 // Global variable for tracking the current LED position
5 // (used for simplicity)
6 int pin = 4; // Start with pin 4 (PA4)
7
8 void SetNumberOnClock(int num)
9 {
10     pin = 4 + num; // Map num to the corresponding GPIO
11     pin (PA4 to PA15)
12     HAL_GPIO_WritePin(GPIOA, 1 << pin, GPIO_PIN_RESET);
13     // Turn on the corresponding LED
14 }
15
16 void clearNumberOnClock(int num)
17 {
18     pin = 4 + num; // Map num to the corresponding GPIO
19     pin (PA4 to PA15)
20     HAL_GPIO_WritePin(GPIOA, 1 << pin, GPIO_PIN_SET); // Turn off the corresponding LED
21 }
22
23 // Function to map a value (hour, minute, or second) to a
24 // 12-position clock
25 int mapToClockPosition(int value, int maxValue) {
26
27 }
```

```

21         return (value * 12) / maxValue; // Maps 0-maxValue
22         to 0-11
23     }
24
25     // Function to turn on appropriate LEDs for hour, minute,
26     // and second
27     void displayTime(int hour, int minute, int second) {
28         // Clear all LEDs first
29         for (int i = 0; i < 12; i++) {
30             clearNumberOnClock(i); // Turn off all LEDs
31         }
32
33         // Map hour, minute, and second to clock positions
34         // (0-11)
35         int hourPosition = mapToClockPosition(hour, 12);
36         int minutePosition = mapToClockPosition(minute, 60);
37         int secondPosition = mapToClockPosition(second, 60);
38
39         // Set the LEDs for hour, minute, and second
40         SetNumberOnClock(hourPosition);
41         SetNumberOnClock(minutePosition);
42         SetNumberOnClock(secondPosition);
43     }
44
45     int hour = 0;
46     int minute = 0;
47     int second = 0;
48     while (1)
49     {
50         // Update the display with the current time
51         displayTime(hour, minute, second);
52         // Simulate time progression
53         HAL_Delay(1000); // 1 second delay
54         second++;
55
56         if (second >= 60) {
57             second = 0;
58             minute++;
59         }
60         if (minute >= 60) {
61             minute = 0;
62             hour++;
63         }
64     }
65 }
```

Program 1.13: Source Code

- **Name:** HOANG VAN PHI
- **MSSV:** 2252608
- **Class:** CO3010_CC01
- **Teacher:** NGUYEN THIEN AN
- **Lab:** 1
- **Link GitHub:** https://github.com/PhiutheWind/LAB1_VXL.git