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

Microprocessor - Microcontroller

Lab Report - CO3010

Lab 2

Advisor(s): Phan Văn Sỹ
Student(s): Lương Đức Hiếu 2352324

HO CHI MINH CITY, SEPTEMBER 2025

Contents

1 Overall

The GitHub link for the Lab 2 project is at : https://github.com/hieuld1003/MPU-MCU

2 Exercise

2.1 Exercise 1

2.1.1 Report 1

Schematic for exercise 1 using Proteus:

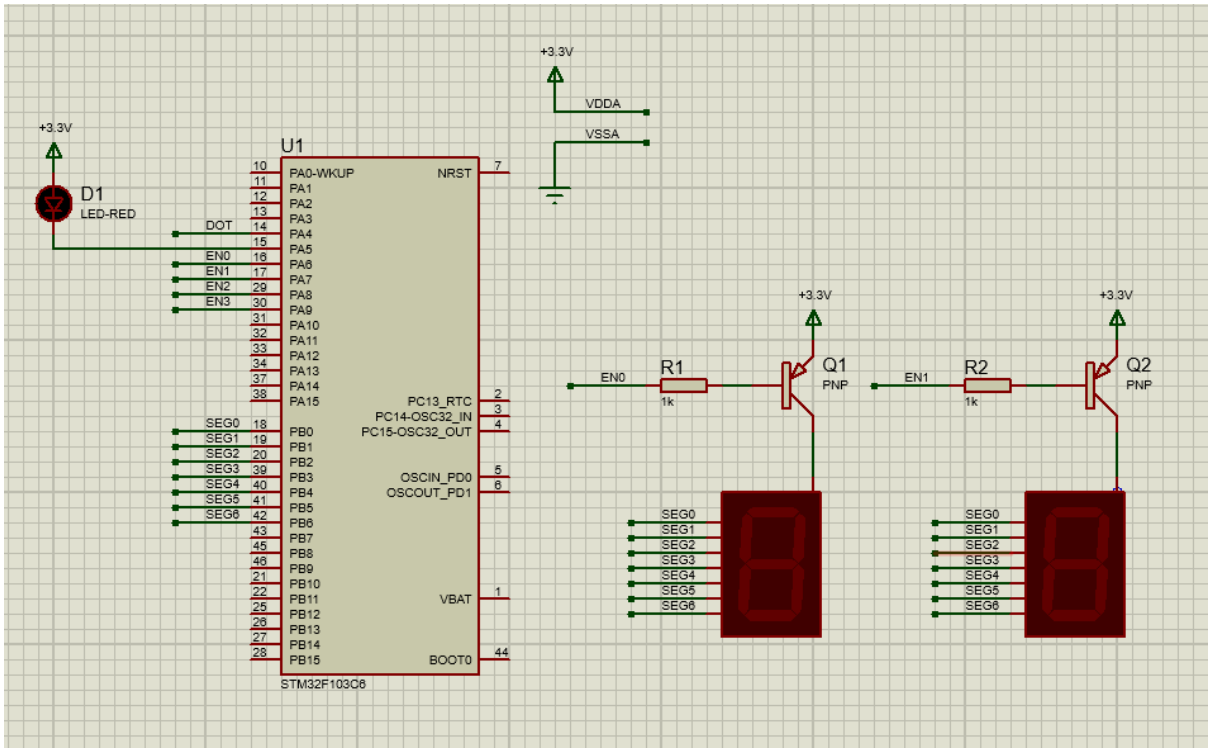

Figure 2.1: Exercise 1 schematic

2.1.2 Report 2

This is the source code for exercise 1:

```
void toggle() {
    HAL_GPIO_TogglePin(EN0_GPIO_Port, EN0_Pin);
    HAL_GPIO_TogglePin(EN1_GPIO_Port, EN1_Pin);
```

```
}

void ex1() {
    display7SEG(1);
    HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 0);
    HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 1);
}

int clock_1 = 50;
void HAL_TIM_PeriodElapsedCallback (TIM_HandleTypeDef * htim) {
    clock_1--;
    if (clock_1 <= 0) {
        switch (HAL_GPIO_ReadPin(EN0_GPIO_Port, EN0_Pin)) {
            case 0: {
                toggle();
                display7SEG(2);
                break;
            }
            case 1:{
                toggle();
                display7SEG(1);
                break;
            }
        }
    clock_1 = 50;
    }
}
```

Source code in main func:

```
int main(void)
{
    SystemClock_Config();
    MX_TIM2_Init();
    MX_GPIO_Init();
    HAL_TIM_Base_Start_IT(&htim2);

    ex1();
    while (1) {}
```

```
}
```

2.1.3 Short question

What is the frequency of the scanning process?

The switching time between the two 7-segment LEDs is 0.5 seconds (500 ms) each, therefore.

$$f = \frac{1}{T} = \frac{1}{0.5\,(s) + 0.5\,(s)} = 1\,(Hz)$$

2.2 Exercise 2

2.2.1 Report 1

Schematic for exercise 2 using Proteus:

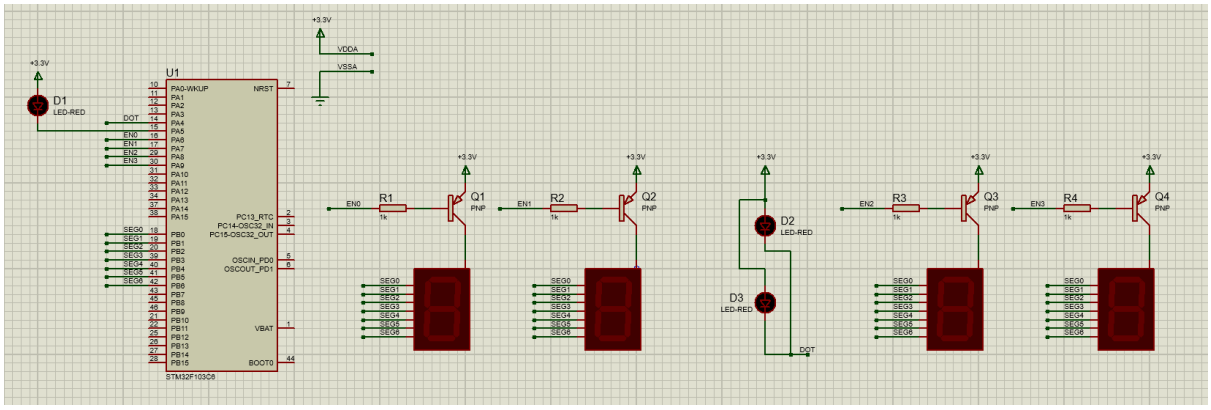

Figure 2.2: Exercise 2 schematic

2.2.2 Report 2

This is the source code for exercise 2:

```
int clock_1 = 100;
int clock7Seg = 1;
int stage = 0;

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef * htim) {
    clock_1 --;
    if (clock_1 <= 0) {
        HAL_GPIO_TogglePin(DOT_GPIO_Port, DOT_Pin);
```

```
        clock_1 = 100;
    }
    clock7Seg--;
    if (clock7Seg <= 0){
        switch (stage){
            case 0:
            {
                HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 0);
                HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 1);
                HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 1);
                HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 1);
                display7SEG(1);
                stage = 1;
                break;
            }
            case 1:
            {
                HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 1);
                HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 0);
                HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 1);
                HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 1);
                display7SEG(2);
                stage = 2;
                break;
            }
            case 2:
            {
                HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 1);
                HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 1);
                HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 0);
                HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 1);
                display7SEG(3);
                stage = 3;
                break;
            }
            case 3:
            {
                HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 1);
```

```
                HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 1);
                HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 1);
                HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 0);
                display7SEG(0);
                stage = 0;
                break;
            }
        }
        clock7Seg = 100;
    }
}
```

Source code in main func:

```
int main(void)
{
    SystemClock_Config();
    MX_TIM2_Init();
    MX_GPIO_Init();
    HAL_TIM_Base_Start_IT(&htim2);
    while (1) {}
}
```

2.2.3 Short question

What is the frequency of the scanning process?

We have 4 seven segment LEDs, so

$$f = \frac{1}{T} = \frac{1}{0.5\,(s) * 4\,(s)} = 0.5\,(Hz)$$

2.3 Exercise 3

2.3.1 Report 1

This is the source code of the update7SEG function for exercise 3:

```
const int MAX_LED = 4;
int index_led = 0;
int led_buffer[4] = {0, 1, 2, 3}; //change value in here

```

```
void ledNum(int idx){
    switch (idx){
        case 0:
        {
            HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 0);
            HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 1);
            HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 1);
            HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 1);
            break;
        }
        case 1:
        {
            HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 1);
            HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 0);
            HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 1);
            HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 1);
            break;
        }
        case 2:
        {
            HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 1);
            HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 1);
            HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 0);
            HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 1);
            break;
        }
        case 3:
        {
            HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 1);
            HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 1);
            HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 1);
            HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 0);
            break;
        }
        default:
        {
            HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, 0);
            HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, 0);
```

```
43              HAL_GPIO_WritePin(EN2_GPIO_Port, EN2_Pin, 0);
44              HAL_GPIO_WritePin(EN3_GPIO_Port, EN3_Pin, 0);
45              break;
46          }
47      }
48  }
49
50  void update7SEG (int index){
51      if (index_led >= MAX_LED) index_led = 0;
52      index = index % 4;
53      switch (index){
54          case 0:{
55              ledNum(index);
56              display7SEG(led_buffer[index]);
57              break;
58          }
59          case 1:{
60              ledNum(index);
61              display7SEG(led_buffer[index]);
62              break;
63          }
64          case 2:{
65              ledNum(index);
66              display7SEG(led_buffer[index]);
67              break;
68          }
69          case 3:{
70              ledNum(index);
71              display7SEG(led_buffer[index]);
72              break;
73          }
74          default:
75              ledNum(-1);
76              break;
77      }
78  }
```

2.3.2 Report 2

This is the source code of the HAL function for exercise 3:

```
int clock7Seg = 50;
int clockBlink = 100;
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef * htim){
    clock7Seg--;
    clockBlink--;
    if (clock7Seg <= 0) {
        update7SEG(index_led++);
        clock7Seg = 50;
    }
    if (clockBlink <= 0){
        HAL_GPIO_TogglePin(DOT_GPIO_Port, DOT_Pin);
        clockBlink = 100;
    }
}
```

Source code in main func:

```
int main(void)
{
    SystemClock_Config();
    MX_TIM2_Init();
    MX_GPIO_Init();
    HAL_TIM_Base_Start_IT(&htim2);
    while (1) {}
}
```

2.4 Exercise 4

2.4.1 Report 1

In exercise 2, the frequency of each 7-segment LED is 0.5 Hz. To set the frequency of those to 1 Hz, we simply reduce the clock7Seg by half.

This is the source code of the HAL function for exercise 4:

```
int clock7Seg = 25;
int clockBlink = 100;
```

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef * htim){
    clock7Seg--;
    clockBlink--;
    if (clock7Seg <= 0) {
        update7SEG(index_led++);
        clock7Seg = 25;
    }
    if (clockBlink <= 0){
        HAL_GPIO_TogglePin(DOT_GPIO_Port, DOT_Pin);
        clockBlink = 100;
    }
}
```

So the frequency of each 7-segment LED now will be:

$$f = \frac{1}{T} = \frac{1}{0.25\,(s) + 0.25\,(s) * 3\,(s)} = 1\,(Hz)$$

2.5 Exercise 5

2.5.1 Report 1

This is the source code for exercise 5:

```
int hour,
minute,
second;
void updateClockBuffer(){
    led_buffer[0] = hour / 10;
    led_buffer[1] = hour % 10;
    led_buffer[2] = minute / 10;
    led_buffer[3] = minute % 10;
}
```

Source code in main func:

```
int main(void)
{
    SystemClock_Config();
    MX_TIM2_Init();
    MX_GPIO_Init();
    HAL_TIM_Base_Start_IT(&htim2);
```

```
7       hour = 1,
8       minute = 2,
9       second = 59;
10      updateClockBuffer();
11      while (1) {
12          HAL_Delay(1000);
13          if (second >= 60) {
14          second = 0;
15          minute++;
16          }
17          if (minute >= 60) {
18          minute = 0;
19          hour++;
20          }
21          if (hour >= 24) {
22          hour = 0;
23          }
24          second++;
25          updateClockBuffer();
26      }
27  }
```

2.6 Exercise 6

2.6.1 Report 1

If in line 1 of the code above is miss, what happens after that and why?

The LED will not blink because the trigger flag is set equal to 0 instead of less than or equal to 0. Consequently, the flag will never be triggered, and the clock will never be reset to its default value.

2.6.2 Report 2

If in line 1 of the code above is changed to setTimer0(1), what happens after that and why?

The LED will not blink because, in the setTimer0 function, the duration is divided by the cycle. Therefore, with a duration less than 10 (the current cycle in this exercise),

timer0 counter is set to 0. This effectively negates the purpose of the setTimer0 function, making it similar to not having it before the while loop.

2.6.3 Report 3

If in line 1 of the code above is changed to setTimer0(10), what is changed compared to 2 first questions and why?

The timer0 counter is initially set to 1. Within the first 10ms, the LED will toggle to the Off state. Subsequently, it will toggle on and off every 2 seconds within the while loop.

2.7 Exercise 7

2.7.1 Report 1

This is the source code in the while loop for exercise 7:

```
int main(void)
{
    SystemClock_Config();
    MX_TIM2_Init();
    MX_GPIO_Init();
    HAL_TIM_Base_Start_IT(&htim2);

    setTimer0(1000);
    while (1) {
        if (timer0_flag == 1){
            HAL_GPIO_TogglePin(DOT_GPIO_Port, DOT_Pin);
            second++;
            if (second >= 60) {
                second = 0;
                minute++;
            }
            if (minute >= 60) {
                minute = 0;
                hour++;
            }
            if (hour >= 24) {
                hour = 0;
```

```
            }
            setTimer0(100);
        }
        updateClockBuffer();
    }
}
```

2.8 Exercise 8

2.8.1 Report 1

This is the source code in the while loop for exercise 8:

```
int main(void)
{
    SystemClock_Config();
    MX_TIM2_Init();
    MX_GPIO_Init();
    HAL_TIM_Base_Start_IT(&htim2);

    setTimer0(1000);
    setTimer1(1000);
    while (1) {
        if (timer0_flag == 1) {
            HAL_GPIO_TogglePin(DOT_GPIO_Port, DOT_Pin);
            second++;
            if (second >= 60) {
                second = 0;
                minute++;
            }
            if (minute >= 60) {
                minute = 0;
                hour++;
            }
            if (hour >= 24) {
                hour = 0;
            }
            setTimer0(100);
        }
```

```
27          if (timer1_flag == 1) {
28                  update7SEG(index_led++);
29                  setTimer1(100);
30          }
31          updateClockBuffer();
32      }
33  }
```

2.9 Exercise 9

2.9.1 Report 1

Schematic for exercise 9 using Proteus:

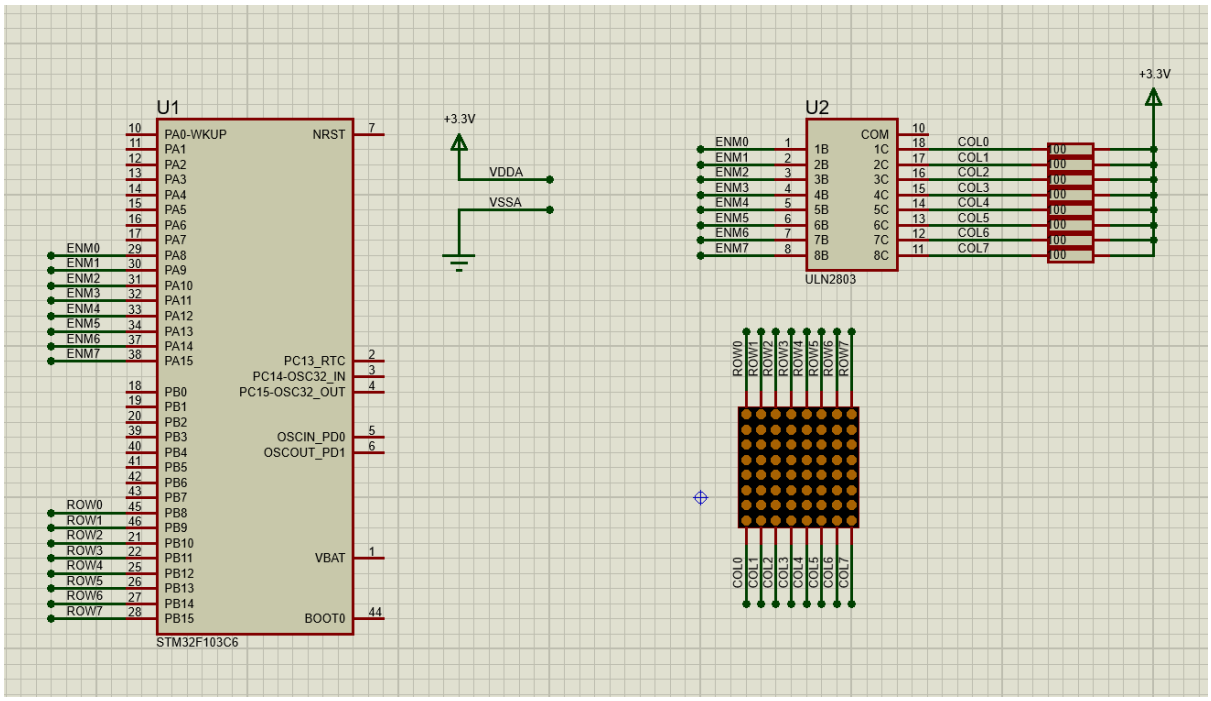

Figure 2.3: Exercise 9 schematic

2.9.2 Report 2

This is the source code for exercise 9:

```
1 int matrix_idx = 0;
2 uint8_t matrix_buffer_A[8] = {0xE7, 0xDB, 0xBD, 0xBD, 0x81, 0xBD,
      0xBD, 0xBD};
```

```
3
4 /* TIMER STARTS HERE */
5 int counter = 0, flag = 0;
6
7 int const cycle = 1;
8
9 void set(int duration){
10     counter = duration / cycle;
11     flag = 0;
12 }
13
14 void run(void){
15     counter--;
16     if (counter == 0) flag = 1;
17 }
18
19 /* TIMER ENDS HERE */
20 int convertedBinary[8] = {0,0,0,0,0,0,0,0};
21
22 void convertToBinary(uint8_t num) {
23     for(int i = 7; i >= 0; i--) {
24         convertedBinary[i] = num % 2;
25         num = num / 2;
26     }
27 }
28
29 void updateLEDMatrix(int index){
30     uint16_t enm_pins[] = {ENM0_Pin, ENM1_Pin, ENM2_Pin, ENM3_Pin
       , ENM4_Pin, ENM5_Pin, ENM6_Pin, ENM7_Pin};
31     if (index < 8) {
32         convertToBinary(matrix_buffer_A[index]);
33         for(int i = 0; i < 8; i++){
34             HAL_GPIO_WritePin(GPIOA, enm_pins[i], (
                   convertedBinary[i] == 0) ? GPIO_PIN_RESET :
                   GPIO_PIN_SET);
35         }
36     }
37 }
```

```
void resetState(){
    HAL_GPIO_WritePin(GPIOB, ROW0_Pin|ROW1_Pin|ROW2_Pin|ROW3_Pin|
       ROW4_Pin|ROW5_Pin|ROW6_Pin|ROW7_Pin, GPIO_PIN_SET);
}
void displayLEDMatrix(){
    if (matrix_idx >= 8) {
            matrix_idx = 0;
        }
        resetState();
        HAL_GPIO_WritePin(GPIOB, ROW0_Pin << matrix_idx,
           GPIO_PIN_RESET);
        updateLEDMatrix(matrix_idx);
}
```

Source code in main func:

```
int main(void)
{
    SystemClock_Config();
    MX_TIM2_Init();
    MX_GPIO_Init();
    HAL_TIM_Base_Start_IT(&htim2);

    set(100);
    while (1)
    {
        if (flag == 1) {
        displayLEDMatrix();
        matrix_idx++;
        set(20);
        }
    }
}
```

2.10 Exercise 10

2.10.1 Report 1

This is the source code for exercise 10:

```
int matrix_idx = 0;
uint8_t matrix_buffer_A[8] = {0xE7, 0xDB, 0xBD, 0xBD, 0x81, 0xBD,
    0xBD, 0xBD};

/* TIMER STARTS HERE */
int counter = 0, flag = 0;

int const cycle = 1;

void set(int duration){
    counter = duration / cycle;
    flag = 0;
}

void run(void){
    counter--;
    if (counter == 0) flag = 1;
}

/* TIMER ENDS HERE */
int convertedBinary[8] = {0,0,0,0,0,0,0,0};

void convertToBinary(uint8_t num) {
    for(int i = 7; i >= 0; i--) {
        convertedBinary[i] = num % 2;
        num = num / 2;
    }
}

void shiftLeft(){
    for (int i = 0; i < 8; i++) {
        matrix_buffer_A[i] = (matrix_buffer_A[i] << 1) | (
           matrix_buffer_A[i] >> 7);
    }
}


```

```
36 void updateLEDMatrix(int index){
37     uint16_t enm_pins[] = {ENM0_Pin, ENM1_Pin, ENM2_Pin, ENM3_Pin
           , ENM4_Pin, ENM5_Pin, ENM6_Pin, ENM7_Pin};
38     if (index < 8) {
39         convertToBinary(matrix_buffer_A[index]);
40         for(int i = 0; i < 8; i++){
41             HAL_GPIO_WritePin(GPIOA, enm_pins[i], (
                   convertedBinary[i] == 0) ? GPIO_PIN_RESET :
                   GPIO_PIN_SET);
42         }
43     }
44 }
45 void resetState(){
46     HAL_GPIO_WritePin(GPIOB, ROW0_Pin|ROW1_Pin|ROW2_Pin|ROW3_Pin|
           ROW4_Pin|ROW5_Pin|ROW6_Pin|ROW7_Pin, GPIO_PIN_SET);
47 }
48 void displayLEDMatrix(){
49     if (matrix_idx >= 8) {
50             matrix_idx = 0;
51             shiftLeft();
52         }
53         resetState();
54         HAL_GPIO_WritePin(GPIOB, ROW0_Pin << matrix_idx,
               GPIO_PIN_RESET);
55         updateLEDMatrix(matrix_idx);
56 }
```

Source code in main func: the same like ex9.