

VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
Faculty of Computer Science and Engineering



CC02 — Lab Report

Microprocessor - Microcontroller Lab 2

Supervisors: Nguyen Thien An
Students: Nguyen Thanh Tai 2252722

Ho Chi Minh City, October 1, 2024



Contents

1	Exercise	2
1.1	Exercise 1	3
1.1.1	Report 1	3
1.1.2	Report 2	3
1.2	Exercise 2	4
1.2.1	Report 1	4
1.2.2	Report 2	4
1.3	Exercise 3	6
1.3.1	Report 1	6
1.3.2	Report 2	6
1.4	Exercise 4	8
1.4.1	Report 1	8
1.5	Exercise 5	9
1.5.1	Report 1	9
1.6	Exercise 6	10
1.6.1	Report 1	10
1.6.2	Report 2	10
1.6.3	Report 3	10
1.7	Exercise 7	11
1.8	Exercise 8	12
1.8.1	Report 1	12
1.9	Exercise 9	14
1.9.1	Report 1	14
1.9.2	Report 2	14
1.10	Exercise 10	16
	References	17

1 Exercise

The GitHub link for the lab schematics is at [here](https://github.com/ThanhTaiNguyen24/mcu-mpu-lab1) or in this link: <https://github.com/ThanhTaiNguyen24/mcu-mpu-lab1>.

The schematic for the exercises from 1 to 10 is located here:

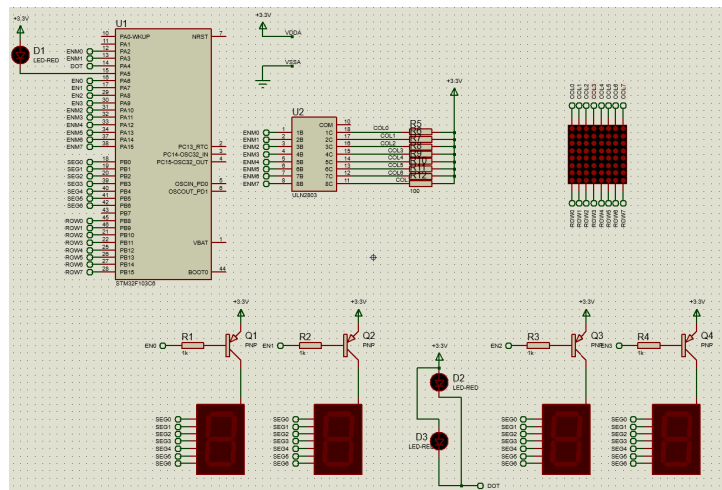


Figure 1: The schematic for the exercises from 1 to 10.



1.1 Exercise 1

1.1.1 Report 1

Can be found at 1.

1.1.2 Report 2

This is the header file library for the exercise 1 :

```
1 int counter = 100;
2 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
3
4     if(counter == 100){
5         led7segmentdisplay(1);
6         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
7         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
8     } else if (counter == 50){
9         led7segmentdisplay(2);
10        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
11        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);
12    }
13    counter--;
14    if (counter == 0) counter = 100;
15 }
```

Short question: the frequency of the scanning process is 1Hz.

1.2 Exercise 2

1.2.1 Report 1

Can be found at 1.

1.2.2 Report 2

This is the source code for the exercise 2:

```
1 int counter = 200;
2 int led_counter = 100;
3 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
4     if(counter == 200){
5         led7segmentdisplay(1);
6         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
7         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
8         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_SET);
9         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_SET);
10    } else if (counter == 150){
11        led7segmentdisplay(2);
12        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);
13        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
14        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_SET);
15        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_SET);
16
17    } else if(counter == 100){
18        led7segmentdisplay(3);
19        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
20        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
21        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_RESET);
22        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_SET);
23
24    } else if (counter == 50){
25        led7segmentdisplay(0);
26        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
27        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
28        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_SET);
29        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_RESET);
30    }
31    counter--;
32    led_counter--;
33    if (led_counter == 50){
34        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_SET);
35    }
36    if (led_counter == 0){
37        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET);
38        led_counter = 100;
39    }
40    if (counter == 0){
```



```
41     counter = 200;  
42 }  
43 }
```

1.3 Exercise 3

1.3.1 Report 1

This is the source code for update7SEG function:

```
1 void update7SEG (int index){
2     switch(index){
3         case 0:
4             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
5             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
6             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_SET);
7             HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_SET);
8             led7segmentdisplay(led_buffer[0]);
9             break;
10        case 1:
11            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);
12            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
13            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_SET);
14            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_SET);
15            led7segmentdisplay(led_buffer[1]);
16            break;
17        case 2:
18            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_RESET);
19            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
20            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
21            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_SET);
22            led7segmentdisplay(led_buffer[2]);
23            break;
24        case 3:
25            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_RESET);
26            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_SET);
27            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
28            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
29            led7segmentdisplay(led_buffer[3]);
30            break;
31        default:
32            break;
33    }
34 }
```

1.3.2 Report 2

This is the source code for PeriodElapsedCallback function:

```
1 int counter = 50;
2 int led_counter = 100;
3 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
4     update7SEG(index_led);
5     counter--;
```



```
6  if(counter == 0){
7      index_led++;
8      counter = 50;
9      if (index_led > 3) index_led = 0;
10 }
11
12
13 led_counter --;
14 if (led_counter == 50){
15     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_SET);
16 }
17 if (led_counter == 0){
18     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET);
19     led_counter = 100;
20 }
21 }
```




1.4 Exercise 4

1.4.1 Report 1

This is the source code for the exercise 4:

```
1 int counter = 25;
2 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
3     update7SEG(index_led);
4     counter--;
5     if(counter == 0){
6         index_led++;
7         counter = 25;
8         if (index_led > 3) index_led = 0;
9     }
10    led_counter--;
11    if (led_counter == 50){
12        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_SET);
13    }
14    if (led_counter == 0){
15        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET);
16        led_counter = 100;
17    }
18 }
```



1.5 Exercise 5

1.5.1 Report 1

This is the source code for the exercise 5:

```
1 void updateClockBuffer (){  
2     led_buffer[0] = hour / 10;  
3     led_buffer[1] = hour % 10;  
4     led_buffer[2] = minute / 10;  
5     led_buffer[3] = minute % 10;  
6 }
```



1.6 Exercise 6

1.6.1 Report 1

Answer: if **setTimer0(1000)** is removed, **timer0_counter** always equal to 0, the LED will not toggle.

1.6.2 Report 2

Answer: **void setTimer0** receive input parameter named **duration** as an **integer**. If **duration** equal to 1, **timer0_counter** (which is an **integer** value equal to 0, the LED will not toggle.

1.6.3 Report 3

Answer: If **duration** equal to 10, **timer0_counter** (which is an **integer** value equal to 1, the **timer_run()** function will invokes in this case, the LED will toggle.

1.7 Exercise 7

This is the code for the exercise 7:

```
1 setTimer1(1000);
2 setTimer2(500);
3 while (1)
4 {
5     if(timer1_flag == 1) {
6         second++;
7         if (second >= 60){
8             second = 0;
9             minute++;
10        }
11        if (minute >= 60) {
12            minute = 0;
13            hour++;
14        }
15        if (hour >= 24) {
16            hour = 0;    24
17        }
18        updateClockBuffer();
19        setTimer1(1000);
20    }
21    if(timer2_flag == 1){
22        HAL_GPIO_TogglePin (GPIOA ,GPIO_PIN_4);
23        setTimer2(500);
24    }
25 }
```

1.8 Exercise 8

1.8.1 Report 1

This is the code for the exercise 8:

```
1 setTimer1(10);
2 setTimer2(10);
3 setTimer3(20);
4 while (1)
5 {
6     if(timer1_flag == 1) {
7         second++;
8         if (second >= 60){
9             second = 0;
10            minute++;
11        }
12        if (minute >= 60) {
13            minute = 0;
14            hour++;
15        }
16        if (hour >= 24) {
17            hour = 0;
18        }
19        updateClockBuffer();
20        update7SEG(index_led);
21        HAL_GPIO_TogglePin (GPIOA ,GPIO_PIN_4);
22        index_led++;
23        if (index_led > 3) index_led = 0;
24        setTimer1(1000);
25    }
26 }
27
28 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
29     timerRun();
30 }
31
32 int timer1_counter = 0;
33 int timer1_flag = 0;
34 int TIMER_CYCLE = 10;
35
36 void setTimer1(int duration){
37     timer1_counter = duration/TIMER_CYCLE;
38     timer1_flag = 0;
39 }
40 void setTimer2(int duration){
41     timer2_counter = duration/TIMER_CYCLE;
42     timer2_flag = 0;
43 }
44 void setTimer3(int duration){
45     timer3_counter = duration/TIMER_CYCLE;
```



```
46     timer3_flag = 0;
47 }
48
49 void timerRun(){
50     if (timer1_counter > 0){
51         timer1_counter--;
52         if(timer1_counter <= 0){
53             timer1_flag = 1;
54         }
55     }
56     if (timer2_counter > 0){
57         timer2_counter--;
58         if(timer2_counter <= 0){
59             timer2_flag = 1;
60         }
61     }
62     if (timer3_counter > 0){
63         timer3_counter--;
64         if(timer3_counter <= 0){
65             timer3_flag = 1;
66         }
67     }
68 }
```



1.9 Exercise 9

1.9.1 Report 1

Can be found at 1.

1.9.2 Report 2

This is the source code for the exercise 9:

```
1 uint8_t matrix_buffer [8] = {0x18,0x3c,0x66,0x66,0x7e,0x7e,0x66,0x66};
2 void updateLedMatrix(int index){
3     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_8, GPIO_PIN_SET);
4     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_9, GPIO_PIN_SET);
5     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_10, GPIO_PIN_SET);
6     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_11, GPIO_PIN_SET);
7     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_SET);
8     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_SET);
9     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_SET);
10    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, GPIO_PIN_SET);
11
12    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, GPIO_PIN_SET);
13    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_SET);
14    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_10, GPIO_PIN_SET);
15    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_11, GPIO_PIN_SET);
16    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_12, GPIO_PIN_SET);
17    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_13, GPIO_PIN_SET);
18    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_14, GPIO_PIN_SET);
19    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_15, GPIO_PIN_SET);
20
21    uint8_t numberCode = matrix_buffer[index];
22    uint8_t value[8];
23    switch(index){
24        case 0: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_8, GPIO_PIN_RESET); break;
25        case 1: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_9, GPIO_PIN_RESET); break;
26        case 2: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_10, GPIO_PIN_RESET); break;
27        case 3: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_11, GPIO_PIN_RESET); break;
28        case 4: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_RESET); break;
29        case 5: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_RESET); break;
30        case 6: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_RESET); break;
31        case 7: HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, GPIO_PIN_RESET); break;
32        default: break;
33    }
34    for (index_led_matrix = 0; index_led_matrix < MAX_LED_MATRIX; index_led_matrix
        ++){
35
36        value[index_led_matrix] = (numberCode >> (7 - index_led_matrix)) & 0x01;
37        value[index_led_matrix] = (value[index_led_matrix] == 1) ? 0 : 1;
38    }
39    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, value[0]);
```



```
40 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, value[1]);  
41 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_10, value[2]);  
42 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_11, value[3]);  
43 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_12, value[4]);  
44 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_13, value[5]);  
45 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_14, value[6]);  
46 HAL_GPIO_WritePin(GPIOA, GPIO_PIN_15, value[7]);  
47 }
```


1.10 Exercise 10

This is the code for the exercise 10:

```
1  setTimer1(10);
2  setTimer2(10);
3  setTimer3(20);
4  while (1){
5      if(timer2_flag == 1){
6          updateLedMatrix(index_led_matrix);
7          index_led_matrix++;
8          if (index_led_matrix > MAX_LED_MATRIX) index_led_matrix = 0;
9          setTimer2(40);
10     }
11     if(timer3_flag == 1){
12         HAL_GPIO_TogglePin (GPIOA, GPIO_PIN_5);
13         ledShiftRight();
14         setTimer3(500);
15     }
16 }
17 void ledShiftRight(){
18     for (int i = 0; i < MAX_LED_MATRIX;i++){
19         uint8_t leftToRight = (matrix_buffer[i] & 0x01) << 7;
20         matrix_buffer[i] = (matrix_buffer[i] >> 1) | leftToRight;
21     }
22 }
```

Explanation: The letter 'A' is shifted to the right. If borderless, the letter 'A' will appear in the first left column of the LED MATRIX again.



References