VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY

HO CHI MINH CITY UNIVERSITY OF TECHNOLY

REPORT

LAB 2

Class: Microprocessors-Microcontrollers – CC01

Lecture: NGUYỄN THIÊN ÂN

No.	Full Name	ID Student
1.	Nguyễn Huy Tài	2110513

Ho Chi Minh City, Octocber 1st 2024

Contents

1 Exercise 1 2

1.1 Report 1 2

1.2 Report 2 2

2 Exercise 2 5

2.1 Report 1 5

2.2 Report 2 5

2.3 Answer the question 5

3 Exercise 3 6

3.1 Report 1 6

3.2 Report 2 6

4 Exercise 4 6

4.1 Report 1 6

5 Exercise 5 7

5.1 Report 1 7

6 Exercise 6 7

6.1 Report 1 7

6.2 Report 2 8

6.3 Report 3 8

7 Exercise 7 8

7.1 Report 1 8

8 Exercise 8 9

8.1 Report 1 9

9 Exercise 9 11

9.1 Report 1 11

9.2 Report 2 11

10 Exercise 10 12

10.1 Report 1 12

10.2 Report 2 13

11 Source 13

1 Exercise 1

1.1 Report 1

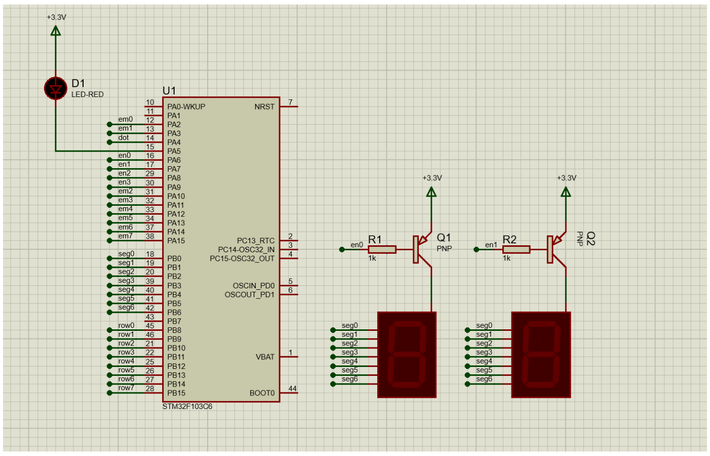

Figure 1: *The schematic of exercise 1*

1.2 Report 2

```
void display7seg(int nth, int value) {

  if (nth == 1)
  {
      HAL_GPIO_WritePin(en0_GPIO_Port, en0_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en1_GPIO_Port, en1_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(en2_GPIO_Port, en2_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en3_GPIO_Port, en3_Pin, GPIO_PIN_SET);
  }
  else if (nth == 0)
  {
      HAL_GPIO_WritePin(en0_GPIO_Port, en0_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(en1_GPIO_Port, en1_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en2_GPIO_Port, en2_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en3_GPIO_Port, en3_Pin, GPIO_PIN_SET);
  }
  else if (nth == 2)
  {
      HAL_GPIO_WritePin(en0_GPIO_Port, en0_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en1_GPIO_Port, en1_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en2_GPIO_Port, en2_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(en3_GPIO_Port, en3_Pin, GPIO_PIN_SET);
  }
  else if (nth == 3)
  {
      HAL_GPIO_WritePin(en0_GPIO_Port, en0_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en1_GPIO_Port, en1_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en2_GPIO_Port, en2_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(en3_GPIO_Port, en3_Pin, GPIO_PIN_RESET);
  }


    uint8_t segments = segmentMap[value];  // Get segment pattern for the value

    // Assuming 7-segment pins are connected to GPIOB Pins 0-6
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, (segments & 0x01) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_1, (segments & 0x02) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, (segments & 0x04) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, (segments & 0x08) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, (segments & 0x10) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, (segments & 0x20) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, (segments & 0x40) ? GPIO_PIN_SET : GPIO_PIN_RESET);
}

void clearLED() {
    // Turn off all segments (assuming GPIOB pins 0-6 control the segments)
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 |
                      GPIO_PIN_4 | GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_RESET);
```

```
}
```

Program 1: Function code in exercise1

```c
int counter = 100, nth = 0;
int buffer[2] = {1, 2};

void HAL_TIM_PeriodElapsedCallback (TIM_HandleTypeDef *htim) {
    counter--;
    if (counter == 50 || counter == 0) {
        if (counter == 0) {
            counter = 100;
            HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);  // Toggle GPIOA PIN 5
        }
        // Alternate between 0 and 1 for ’nth’
        nth = (nth == 0) ? 1 : 0;

        clearLED();  // Clear current display

        // Display the value on the 7-segment display
        display7seg(nth, buffer[nth]);
    }
}
```

Program 2: Source code in the **HAL_TIM_PeriodElapsedCallback** function

2 Exercise 2

2.1 Report 1

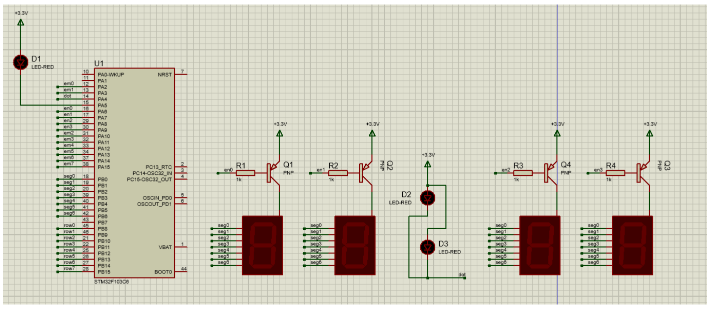

Figure 2: *The schematic of exercise 2*

2.2 Report 2

int counter

```

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim) {
  counter--;

  if (counter == 50 || counter == 0) {
    if (counter == 0) {
      counter = 100;
      HAL_GPIO_TogglePin(GPIOA, led_red_Pin | dot_Pin);
    }

    clearLED();
    display7seg(idx, buffer[idx]);

    idx = (idx + 1) % 4;
  }
}
```

Program 3: Source code in the **HAL_TIM_PeriodElapsedCallback** function

2.3 Answer the question

The frequency of the clock is 2 Hz

3 Exercise 3

3.1 Report 1

```c
void update7SEG(int index)
{
  int led_buffer [4] = {1 , 2 , 3 , 4};
  clearEnableVsLED();
  display7seg(index, led_buffer[index]);
}
```

Program 4: Source code in update7SEG(int index) function

3.2 Report 2

```c
int counter = 100, idx = 0;

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
  counter--;

  if (counter == 50 || counter == 0) {
    if (counter == 0) {
      counter = 100;
      HAL_GPIO_TogglePin(GPIOA, led_red_Pin | dot_Pin);
    }

//    clearLED();
    update7SEG(idx);

    idx = (idx + 1) % 4;
  }
}
```

Program 5: Source code in the **HAL_TIM_PeriodElapsedCallback** function

4 Exercise 4

4.1 Report 1

```c
/*
 * inorder to display 1hz for all 4 leds, which means each leds should bright within 250ms
 */
int counter = 50, idx = 0;

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim){
  counter--;

  if (counter == 25 || counter == 0) {
    if (counter == 0) {
      counter = 50;
      HAL_GPIO_TogglePin(GPIOA, led_red_Pin | dot_Pin);
    }

//    clearLED();

    idx++;

    if (idx >=4)
    {
      idx = 0;
      update7SEG(idx);
    }

    else update7SEG(idx);
  }
}
```

Program 6: Source code in the **HAL_TIM_PeriodElapsedCallback** function

5 Exercise 5

5.1 Report 1

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

Program 7: Source code in the **updateClockBuffer** function

6 Exercise 6

6.1 Report 1

If line 1 of the code is missed, what happens after that and why?

If line 1 of the code is missed, the value timer0_flag is kept at 0 and can not be set to 1, so the LED will not blink

6.2 Report 2

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

If line 1 of the code is changed to setTimer0(1), the LED will not blink, because if duration = 1, we get timer0_counter = 0 (since timer0_counter is of type int), then when executing timer_run(), the value of timer0_counter can not satisfy the if condition, thus timer0_flag is kept at 0 and can not be set to 1

6.3 Report 3

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

If line 1 of the code is changed to setTimer0(10), we get timer0_counter = 1, this value satisfy the if condition in timer_run() and the timer0_flag is set to 1 right away, so the LED will be invoked and start blinking properly.

7 Exercise 7

7.1 Report 1

```
int index = 0;
int hour = 12, min = 38, sec = 55;
int TIMEstate = 0;
int index_matrix = 0;
int count_dot = 0;
setTimer1(250);
int bufferClock[4] = {0,0,0,0};
updateClockBuffer(bufferClock, hour, min, sec);
display7seg(0, bufferClock[0]);
index++;

// flag[1] is trigger for second
// count_dot is trigger for dot_led

while (1)
{
    if (count_dot == 2)
    {
      // LED & DOT
      HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_4);
      HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
      count_dot = 0;
    }

    switch(TIMEstate)
    {
      case 0:
        sec++;
        setTimer0(1000);
        TIMEstate = 1;
        break;
      case 1:
        if (flags[0] == 1)
        {
          if (sec >= 60)
          {
            sec = 0;
            min++;
          }
          if (min >= 60)
          {
            min = 0;
            hour++;
          }
          if (hour >= 24)
          {
            hour = 0;
          }
          TIMEstate = 0;
        }
        break;
    }

    updateClockBuffer(bufferClock, hour, min, sec);

    if (index >= 4) index = 0;
}
```

Program 8: Source code in the while loop

8 Exercise 8

8.1 Report 1

```
int index = 0;
int hour = 12, min = 38, sec = 55;
int TIMEstate = 0;
int index_matrix = 0;
int count_dot = 0;
setTimer1(250);
int bufferClock[4] = {0,0,0,0};
updateClockBuffer(bufferClock, hour, min, sec);
display7seg(0, bufferClock[0]);
index++;

```

```
// flag[0] is trigger for second
// count_dot is trigger for dot_led
// flag[1] is trigger for scanning

while (1)
{
    if (count_dot == 2)
    {
      // LED & DOT
      HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_4);
      HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
      count_dot = 0;
    }

   if (flags[1] == 1)
    {
        clearLED();
        display7seg(index, bufferClock[index]);
        index++;
        count_dot++;
        setTimer1(250);
    }

    switch(TIMEstate)
    {
      case 0:
        sec++;
        setTimer0(1000);
        TIMEstate = 1;
        break;
      case 1:
        if (flags[0] == 1)
        {
          if (sec >= 60)
          {
            sec = 0;
            min++;
          }
          if (min >= 60)
          {
            min = 0;
            hour++;
          }
          if (hour >= 24)
          {
            hour = 0;
          }
          TIMEstate = 0;
        }
```

```
            break;
        }

    updateClockBuffer(bufferClock, hour, min, sec);

    if (index >= 4) index = 0;
}
```

Program 9: Source code in the while loop

9 Exercise 9

9.1 Report 1

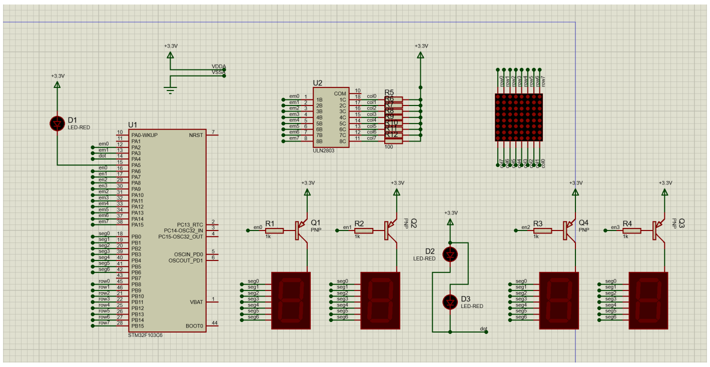

Figure 3: *The schematic of exercise 9*

9.2 Report 2

```
void setRow(int row)
{
    HAL_GPIO_WritePin(GPIOB, row0_Pin, (row == 0)? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, row1_Pin, (row == 1)? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, row2_Pin, (row == 2)? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, row3_Pin, (row == 3)? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, row4_Pin, (row == 4)? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, row5_Pin, (row == 5)? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, row6_Pin, (row == 6)? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, row7_Pin, (row == 7)? GPIO_PIN_SET : GPIO_PIN_RESET);
}
```

```

void setColumn(int value)
{
    HAL_GPIO_WritePin(GPIOA, em0_Pin, (value & 0x01) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOA, em1_Pin, (value & 0x02) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOA, em2_Pin, (value & 0x04) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOA, em3_Pin, (value & 0x08) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOA, em4_Pin, (value & 0x10) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOA, em5_Pin, (value & 0x20) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOA, em6_Pin, (value & 0x40) ? GPIO_PIN_SET : GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOA, em7_Pin, (value & 0x80) ? GPIO_PIN_SET : GPIO_PIN_RESET);
}

void updateLEDMatrix(int index)
{
    setRow(index);
    setColumn(matrix_buffer[index]);
}
```

Program 10: Source code of functions matrixled in functions.h

```
while (1)
{
    if (flags[3] == 1)
    {
        updateLEDMatrix(index_matrix);
        index_matrix ++;
        setTimer3(20);
    }
    if (index_matrix >= 8 )
    {
        index_matrix = 0;
    }
}
```

Program 11: Source code in the while loop

10 Exercise 10

10.1 Report 1

```c
void shiftLeft(uint8_t matrix_buffer[8])
{
    for (int i = 0; i < 8; i++) {
        uint8_t leftBit = (matrix_buffer[i] & 0x80) >> 7;
        matrix_buffer[i] = (matrix_buffer[i] << 1) |
   leftBit;
    }
}
```

Program 12: **shiftleft** function in functions.h

```c
while (1)
{
    if (flags[3] == 1)
    {
        updateLEDMatrix(index_matrix);
        index_matrix ++;
        setTimer3(20);
    }
    if (index_matrix >= 8 )
    {
        shiftLeft(matrix_buffer);
        index_matrix = 0;
    }
}
```

Program 13: Source code in the while loop

10.2 Report 2

I put make the shift left by using concatenation method and put in the if condition

11 Source

You can find the source code on my GitHub repository: **My GitHub Repository**.