VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY

HO CHI MINH CITY UNIVERSITY OF TECHNOLY

REPORT

LAB 3

Class: Microprocessors-Microcontrollers – CC01

Lecture: NGUYỄN THIÊN ÂN

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

Ho Chi Minh City, November 12th 2024

Contents

1 Exercise 1

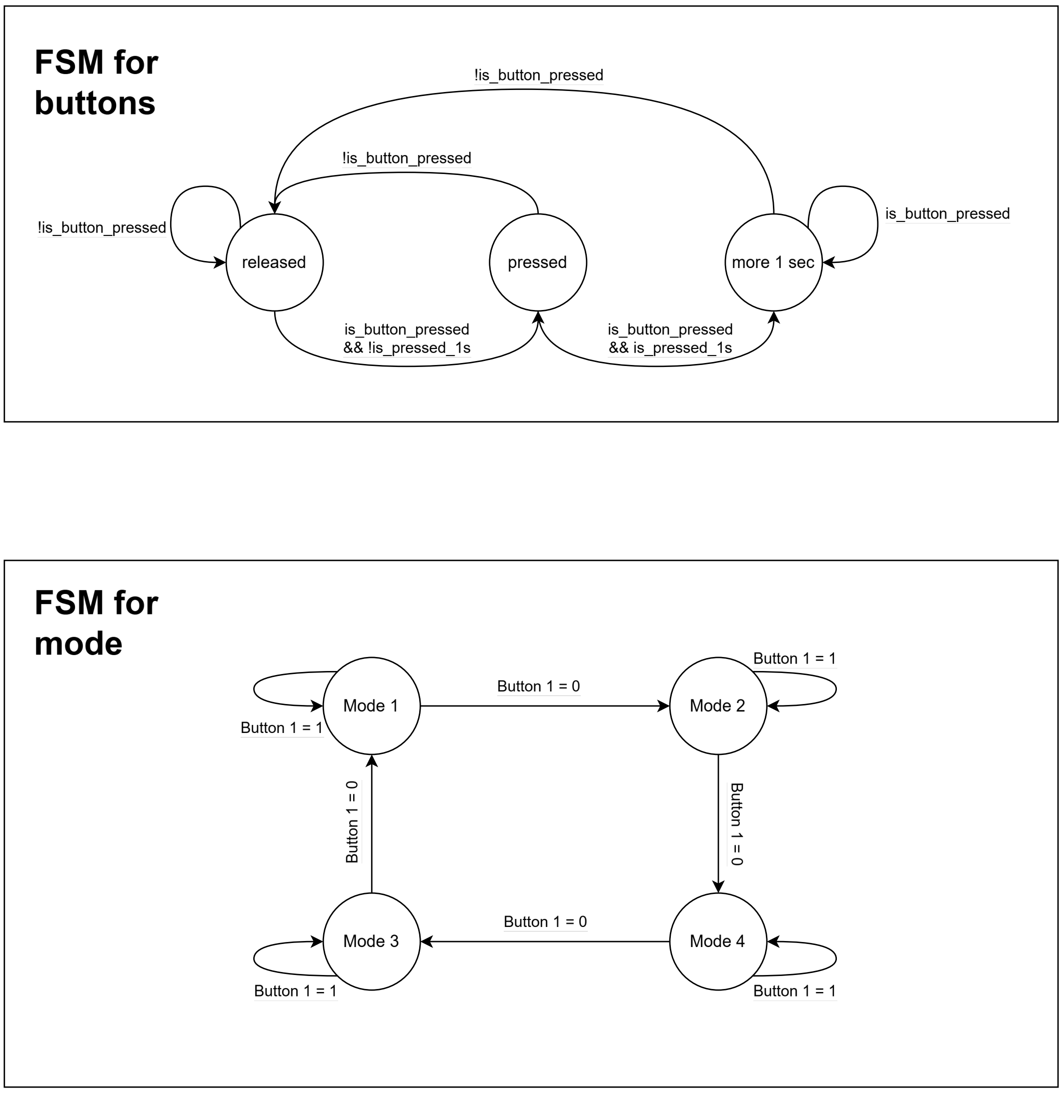


Figure 1: *The schematic of overall system*

In the FSM for mode:

- In mode 1, the system will run traffic lights normally.
- In mode 2, the red LEDs will blink at a frequency of 2 Hz. Additionally, the first two displayed LEDs will indicate the count of red LEDs, while the last two LEDs will display the mode. If button 2 is pressed, the counter increases by one unit. If button 3 is pressed, the counter is set to a new value determined by button 2.
- Mode 3 functions similarly to Mode 2, but for the amber LEDs.
- Mode 4 functions similarly to Mode 2, but for the green LEDs.

2 Exercise 2

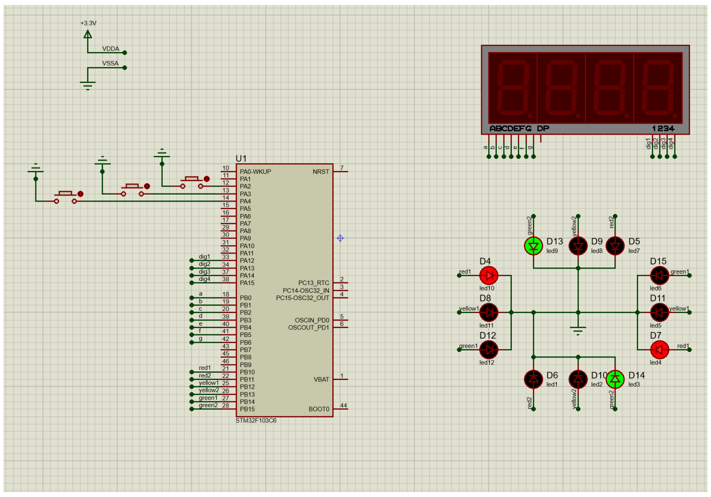


Figure 2: *The schematic of traffic light system*

3 Exercise 3

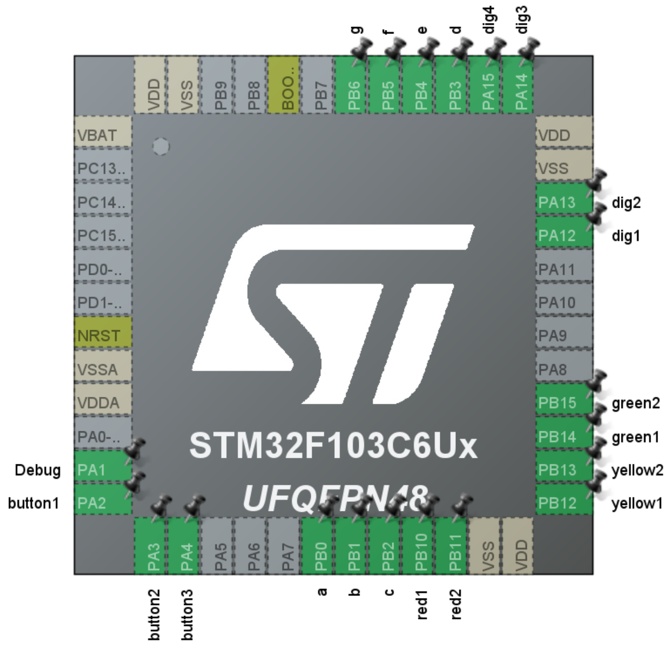
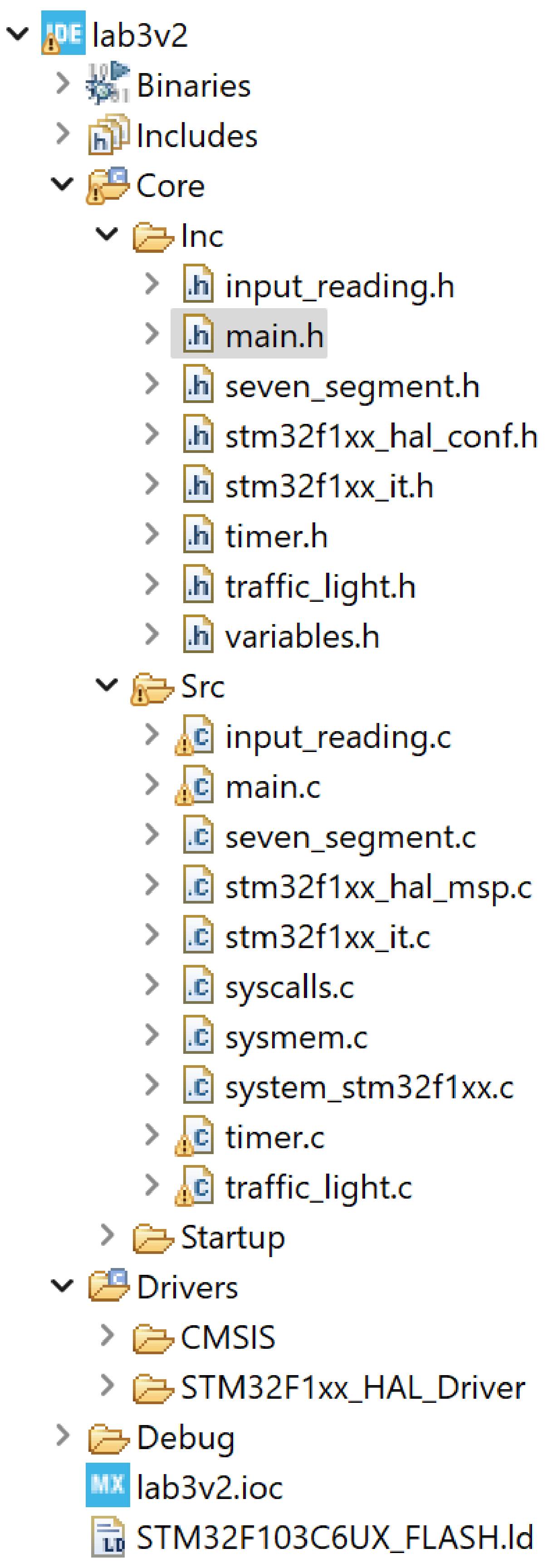


Figure 3: *The picture shows ioc file of system*



```
lab3v2
  Binaries
  Includes
  Core
    Inc
      input_reading.h
      main.h
      seven_segment.h
      stm32f1xx_hal_conf.h
      stm32f1xx_it.h
      timer.h
      traffic_light.h
      variables.h
    Src
      input_reading.c
      main.c
      seven_segment.c
      stm32f1xx_hal_msp.c
      stm32f1xx_it.c
      syscalls.c
      sysmem.c
      system_stm32f1xx.c
      timer.c
      traffic_light.c
    Startup
  Drivers
    CMSIS
    STM32F1xx_HAL_Driver
  Debug
  lab3v2.ioc
  STM32F103C6UX_FLASH.ld
```

Figure 4: *The tree view of directory of system*

4 Exercise 4

5 Exercise 5

5.1 Report 1

```c
int setCounter[3] = {0};

// We aim to work with more than one button
#define NO_OF_BUTTONS 3

// Timer interrupt duration is 10ms, so to pass 1 second, we need to jump to the interrupt service routine 100 times
#define DURATION_FOR_AUTO_INCREASING 100
#define BUTTON_IS_PRESSED GPIO_PIN_RESET
#define BUTTON_IS_RELEASED GPIO_PIN_SET

// The buffer that the final result is stored at debouncing
static GPIO_PinState buttonBuffer[NO_OF_BUTTONS];

// We define two buffers for debouncing
static GPIO_PinState debounceButtonBuffer1[NO_OF_BUTTONS];
static GPIO_PinState debounceButtonBuffer2[NO_OF_BUTTONS];

// We define a flag for a button pressed more than 1 second.
static uint8_t flagForButtonPress1s[NO_OF_BUTTONS];

// We define a counter for automatically increasing the value after the button is pressed more than 1 second.
static uint16_t counterForButtonPress1s[NO_OF_BUTTONS];

static uint16_t pinsButtons[NO_OF_BUTTONS] =
{
    button1_Pin,
    button2_Pin,
    button3_Pin
};

void button_reading(void) {
    for (char i = 0; i < NO_OF_BUTTONS; ++i) {
        debounceButtonBuffer1[i] = debounceButtonBuffer2[i];
        debounceButtonBuffer2[i] = HAL_GPIO_ReadPin(button1_GPIO_Port, pinsButtons[i]);

        if (debounceButtonBuffer1[i] == debounceButtonBuffer2[i]) {
            buttonBuffer[i] = debounceButtonBuffer1[i];

            if (buttonBuffer[i] == BUTTON_IS_PRESSED) {
                // If a button is pressed, we start counting
                if (counterForButtonPress1s[i] < DURATION_FOR_AUTO_INCREASING) {
                    counterForButtonPress1s[i]++;
                } else {
                    // The flag is turned on when 1 second has passed since the button is pressed.
                    flagForButtonPress1s[i] = 1;
                    // todo
                }
            } else {
                counterForButtonPress1s[i] = 0;
                flagForButtonPress1s[i] = 0;
            }
        }
    }
}
```

Program 1: Code for debouncing buttons **input_reading.c**

6 Exercise 6

```c
enum Mode {MODE1, MODE2, MODE3, MODE4};
enum Mode mode = MODE1;
// .
// .
// .
// Handle mode switching
switch(mode) {
    case MODE1:
        fsm_traffic_light();
        if (flag[1] == 1)
        {
            setTimer_blinkly(50);
        }
        previous_mode = 1;
        setTimer_blinkly(50);
        setCounter[0] = counterLightBuffer[0];
        setCounter[1] = counterLightBuffer[1];
        setCounter[2] = counterLightBuffer[2];
        break;

    case MODE2:
        if (previous_mode == 1)
        {
            HAL_GPIO_WritePin(GPIOB, red1_Pin, 0);
            HAL_GPIO_WritePin(GPIOB, red2_Pin, 0);
        }
        if (flag[1] == 1)
        {
            HAL_GPIO_TogglePin(GPIOB, red1_Pin);
            HAL_GPIO_TogglePin(GPIOB, red2_Pin);
            setTimer_blinkly(50);
        }
        HAL_GPIO_WritePin(GPIOB, yellow1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, yellow2_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, green1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, green2_Pin, 0);

        updateDigitBuffer(setCounter[0], 02);// displaying
    mode 02

        previous_mode = 2;
        break;

    case MODE3:
        HAL_GPIO_WritePin(GPIOB, red1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, red2_Pin, 0);
        if (flag[1] == 1)
        {
            HAL_GPIO_TogglePin(GPIOB, yellow1_Pin);
            HAL_GPIO_TogglePin(GPIOB, yellow2_Pin);
            setTimer_blinkly(50);
        }
        HAL_GPIO_WritePin(GPIOB, green1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, green2_Pin, 0);
        updateDigitBuffer(setCounter[1], 03);// displaying
    mode 03
        previous_mode = 3;
        break;

    case MODE4:
        HAL_GPIO_WritePin(GPIOB, red1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, red2_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, yellow1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, yellow2_Pin, 0);
        if (flag[1] == 1)
        {
            HAL_GPIO_TogglePin(GPIOB, green1_Pin);
            HAL_GPIO_TogglePin(GPIOB, green2_Pin);
            setTimer_blinkly(50);
        }
        reset_state();
        updateDigitBuffer(setCounter[2], 04); // displaying
    mode 04
        previous_mode = 4;
        break;
}
```

Program 2: Code for switching mode and displaying it in **input_reading.c**

7 Exercise 7 - 8 - 9

```
/*****************Beginning handle button
   2****************/

if (buttonStates[1] == BUTTON_PRESSED || buttonStates[1] ==
    BUTTON_PRESSED_MORE_THAN_1_SECOND)
{

    HAL_GPIO_WritePin(GPIOA, Debug_Pin, 1);
    // debug on if button1 pressed

    if (buttonStates[1] ==
   BUTTON_PRESSED_MORE_THAN_1_SECOND)
    {
        if (flag[4] == 1)
        {
            if (mode == 1) setCounter[0] ++;
            else if (mode == 2) setCounter[1] ++;
            else if (mode == 3) setCounter[2] ++;

            //////////////////////////////////////////
            if (setCounter[0] == 100) setCounter[0] = 0;
            else if (setCounter[1] == 100) setCounter[1] =
   0;
            else if (setCounter[2] == 100) setCounter[2] =
   0;

            setTimer_increasing_num(10);
        }
    }

    else
    {
        if (flag[4] == 1)
        {
            if (mode == 1) setCounter[0] ++;
            else if (mode == 2) setCounter[1] ++;
            else if (mode == 3) setCounter[2] ++;

            //////////////////////////////////////////
            if (setCounter[0] == 100) setCounter[0] = 0;
            else if (setCounter[1] == 100) setCounter[1] = 0;
            else if (setCounter[2] == 100) setCounter[2] = 0;

            setTimer_increasing_num(100);
        }
    }
}

else
{
    HAL_GPIO_WritePin(GPIOA, Debug_Pin, 0);
}

/**************Ending handle button2**************/

/**********Beginning handle button 3************/

if (buttonStates[2] == BUTTON_PRESSED || buttonStates[2] == BUTTON_PRESSED_MORE_THAN_1_SECOND)
{
    if (flag[4] == 1)
    {
        if (mode == 1) counterLightBuffer[0] = setCounter[0];
        else if (mode == 2) counterLightBuffer[1] = setCounter[1];
        else if (mode == 3) counterLightBuffer[2] = setCounter[2];
        setTimer_increasing_num(5);
    }
}

else
{
    HAL_GPIO_WritePin(GPIOA, Debug_Pin, 0);
}

/*************Ending handle button 3************/
```
Program 3: Code for code for increasing time duration value for the red LEDs in **input_reading.c**

8 Exercise 10

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