Lab.6 STM32 Keypad Scanning

0516076 施威綸

1. Lab objectives 實驗目的

- Understand the principle of STM32
- Use C code to control STM32
- design program for 7-seg LED and keypad

2. Steps 實驗步驟

2.1. Max7219 displayer

Modify your code in lab5.2 to make it callable by C. Add a C file to complete the code given below, display your student ID on 7-Seg LED.

Modify the assembly code into standard procedure format:

(a) Add push, pop into the function.
(b) Using r0~r4 for arguments.
(c) Declare the global function name.

```
    .syntax unified
    .cpu cortex-m4
    .thumb
.data
    student_id: .byte 0, 5, 1, 6, 0, 7, 6

.text
    //.global main
    .global GPIO_init
    .global max7219_init
    .global max7219_send

    .equ RCC_AHB2ENR, 0x4002104C
    .equ GPIOA_BASE, 0x48000000
    .equ GPIOA_MODER, 0x48000000
    .equ GPIOA_OTYPER, 0x48000004
    .equ GPIOA_OSPEEDR, 0x48000008
    .equ GPIOA_PUPDR, 0x4800000C
    .equ GPIOA_ODR, 0x48000014
    .equ GPIOA_BSRR_OFFSET, 0x18
    .equ GPIOA_BRR_OFFSET, 0x28

    .equ DECODE_MODE, 0x9
    .equ INTENSITY, 0xA
    .equ SCAN_LIMIT, 0xB
    .equ SHUTDOWN, 0xC
    .equ DISPLAY_TEST, 0xF
    .equ LOAD, 0x40
    .equ DATA, 0x20
    .equ CLK, 0x80
```

GPIO_INIT():

```
54 GPIO_init:
55 //TODO: Initialize three GPIO pins as output for max7219 DIN, CS and CLK
56     push {r0, r1, r2, lr}
57     // Enable AHB2 GPIOA clock
58     movs r0, #0x1
59     ldr r1, =RCC_AHB2ENR
60     str r0, [r1]
61
62     // Set PA5~7 as output mode
63     movs r0, #0x5400 // 0x010101, set pa5~7 as output
64     ldr r1, =GPIOA_MODER
65     ldr r2, [r1]
66     and r2, #0xFFFF03FF // Mask MODERs
67     orrs r2, r2, r0
68     str r2, [r1]
69
70
71     // Set PA5~7 as high speed mode
72     movs r0, #(0x2A<<10) // set them as 0x101010
73     ldr r1, =GPIOA_OSPEEDR
74     ldr r2, [r1]
75     and r2, #0xFFFF03FF // Mask SPEEDRs
76     orrs r2, r2, r0
77     str r2, [r1]
78
79     pop {r0, r1, r2, pc}
```

Max7219_init():

```
81 max7219_init:
82 // Initialize max7219 registers
83     push {r0, r1, lr}//save link register
84
85     //decode mode setting
86     ldr r0, =DECODE_MODE // 0xX9
87     movs r1, #0xFF // decode all digits
88     bl max7219_send
89
90     //intensity setting
91     ldr r0, =INTENSITY // 0xXA
92     movs r1, #0xA // from 0~F
93     bl max7219_send
94
95     //scan-limit setting
96     ldr r0, =SCAN_LIMIT // 0xXB
97     movs r1, #0 // display digit 0 only
98     bl max7219_send
99
100    //shutdown setting(normal operation)
101    ldr r0, =SHUTDOWN // 0xXC
102    movs r1, #1 // normal operation
103    bl max7219_send
104
105    //display-test setting
106    ldr r0, =DISPLAY_TEST // 0xF
107    //movs r1,#1 // display-test
108    movs r1, #0 // not display-test, just normal operation
109    bl max7219_send
110
111    //display initial 0
112    mov r0, #1
113    mov r1, #0
114    bl max7219_send
115
116    pop {r0, r1, pc}
```

Max7219_send(int address, int data):

```
118 max7219_send:
119    //input parameter: r0 is ADDRESS , r1 is DATA
120     //TODO: Use this function to send a message to max7219
121     push {r0, r1, r2, r3, r4, r5, r6, r7, r8, r9, lr}
122     lsl r0, r0, #8
123     add r0, r0, r1
124     ldr r1, =#GPIOA_BASE
125     ldr r2, =LOAD // 0x40
126     ldr r3, =DATA // 0x20
127     ldr r4, =CLK // 0x80
128     ldr r5, =GPIOA_BSRR_OFFSET // 0x18
129     ldr r6, =GPIOA_BRR_OFFSET // 0x28
130     mov r7, #16 // r7 = i
131 .max7219send_loop:
132     mov r8, #1
133     sub r9, r7, #1
134     lsl r8, r8, r9 // r8 = mask
135     str r4, [r1,r6] //HAL_GPIO_WritePin(GPIOA, CLOCK, 0);
136     tst r0, r8
137     beq .bit_not_set //bit not set
138     str r3, [r1,r5]
139     b .if_done
140 .bit_not_set:
141     str r3, [r1,r6]
142 .if_done:
143     str r4, [r1,r5]
144     subs r7, r7, #1
145     bgt .max7219send_loop
146     str r2, [r1,r6]
147     str r2, [r1,r5]
148     pop {r0, r1, r2, r3, r4, r5, r6, r7, r8, r9, pc}
149     BX LR
```

C file:

```
 2 extern void GPIO_init();
 3 extern void max7219_init();
 4 extern void max7219_send(unsigned int address, unsigned int data);
 5
 6 void display(int* id, int num_digs) {
 7     int i = 0;
 8     max7219_send(0xB, num_digs-1); // set scan limit
 9     for(i=0; i<num_digs; i++) {
10         max7219_send(num_digs-i, id[i]);
11     }
12 }
13
14 int main() {
15     int id[7] = {0, 5, 1, 6, 0, 7, 6};
16     GPIO_init();
17     max7219_init();
18     display(id, 7);
19
20     return 0;
21 }
```

2.2. Keypad Scanning

Use 4 input GPIO pins and 4 output GPIO pins to connect with keypad. Show the corresponding number of pressed button on 7-SegLED.

(a) Include stm32l476xx.h, and initial GPIO in the c file.
(b) Scan the keypad input.
(c) Pass the display value into display(), and call max7219_send().

```c
#include <stm32l476xx.h>
//TODO: define your gpio pin
#define X0 13
#define X1 14
#define X2 15
#define X3 1
#define Y0 3
#define Y1 5
#define Y2 4
#define Y3 10

#define SCAN_LIMIT 0xB

//unsigned int x_pin[4] = {X0, X1, X2, X3};
//unsigned int y_pin[4] = {Y0, Y1, Y2, Y3};

extern void GPIO_init();
extern void max7219_init();
extern void max7219_send(unsigned int address, unsigned int data);

void keypad_init();
int keypad_scan(); // return keypad value
void display(int);


int main() {

    GPIO_init();
    max7219_init();
    keypad_init();

    while(1) {
        display(keypad_scan());
    }
    //display(12345678);
    return 0;
```

Keypad_init():

```c
void keypad_init() {
    // enable GPIOB clock
    RCC->AHB2ENR |= 0x2;

    // GPIO output (PB1, PB15, PB14, PB13)
    GPIOB->MODER &= 0x57FFFFF7; // output 0x01
    GPIOB->PUPDR |= 0x54000004; // pull-up 0x01
    GPIOB->OSPEEDR |= 0x54000004; // mid-speed 0x01

    // GPIO input (PB10, PB4, PB5, PB3)
    GPIOB->MODER &= 0xFFCFF03F; // input 0x00
    GPIOB->PUPDR &= 0; // pull-down 0x10
    GPIOB->PUPDR |= 0x00202A80;
    GPIOB->OSPEEDR |= 0x54000004; // mid-speed 0x01
}
```

keypad_scan():

```
// return keypad value
int keypad_scan() {
    int key;
    int flag_keypad, flag_debounce, flag_key;
    int table[4][4] = {{1, 4, 7, 15}, {2, 5, 8, 0}, {3, 6, 9, 14}, {10, 11, 12, 13}};
    int o_pin_offset[4] = {13, 14, 15, 1};
    int i_pin_offset[4] = {3, 5, 4, 10};
    int i, j;

    while(1) {
        GPIOB->ODR |= 0xE002;
        flag_keypad = GPIOB->IDR & 0x0438; // PB10, 4, 5, 3
        if(flag_keypad != 0){ // detected input -> debounce
            int k = 45000;
            while(k!=0) {
                flag_debounce = GPIOB->IDR & 0x0438;
                k --;
            }
            if(flag_debounce != 0) {
                for(i=0; i<4; i++) { // scan keypad columns (output)
                    // set all outputs low excepts the test col.
                    GPIOB->ODR = 0xFFFF0000 | (1<<o_pin_offset[i]);

                    for(j=0; j<4; j++) { // read keypad rows (input)
                        flag_key = GPIOB->IDR & (1<<i_pin_offset[j]);
                        if(flag_key != 0){
                            key = table[i][j];
                        }
                    }
                }
                return key;
            }
        }
    }
}
```

display(int): display the integer through max7219_send()

```
void display(int num) {
    int arr[8];
    int num_digs = 0;

    if(num == 0) {
        max7219_send(SCAN_LIMIT, 0);
        max7219_send(1, 0);
    }

    while(num != 0) {
        arr[num_digs] = (num%10);
        num /= 10;
        num_digs ++;
    }
    max7219_send(SCAN_LIMIT, num_digs-1);
    while(num_digs != 0) {
        max7219_send(num_digs, arr[num_digs-1]);
        num_digs --;
    }
}
```

2.3. Single and multi buttons

Show pressed button of keypad on 7-Seg LED. Each value of corresponding button is given below.

	X0	X1	X2	X3
Y0	1	2	3	10
Y1	4	5	6	11
Y2	7	8	9	12
Y3	C	0	C	13

When multiple buttons are pressed, show the sum of values that buttons pressed representing. If shown value is greater than *99999999*, don't modify the number showing on 7-Seg LED until button C is pressed.

(a) Same as lab6.2.

(b) Modify keypad_scan() and return the sum of multiple buttons.

main():

```
28 int main() {
29     int input;
30     int display_num = 0;
31
32     GPIO_init();
33     max7219_init();
34     keypad_init();
35
36     while(1) {
37         input = keypad_scan();
38         if(input == -1) display_num = 0;
39         else if((display_num + input) <= 99999999) display_num += input;
40
41         display(display_num);
42     }
43     //display(12345678);
44     return 0;
45 }
```

keypad_scan():

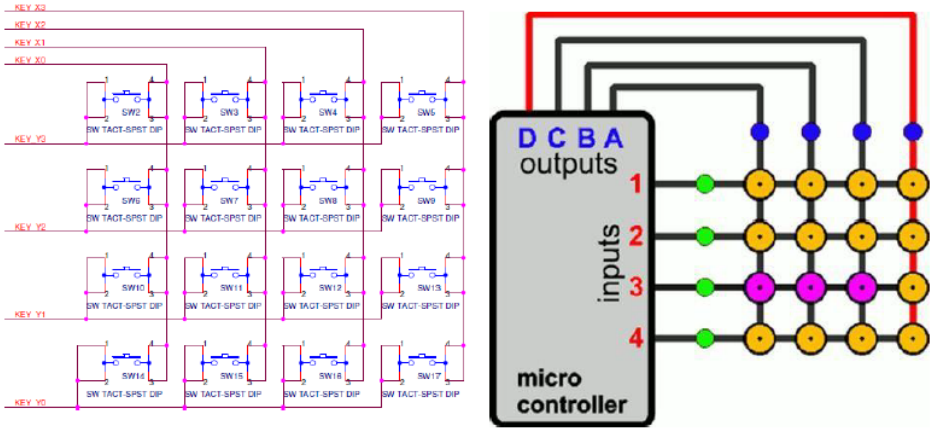
```
86              if(flag_debounce != 0) {
87                  // check from rows to cols
88                  for(j=0; j<4; j++) { // read each keypad rows (input)
89                      for(i=0; i<4; i++) { // scan each keypad columns (output)
90                          GPIOB->ODR = 0xFFFF0000 | (1<<o_pin_offset[i]);
91                          flag_key = GPIOB->IDR & (1<<i_pin_offset[j]);
92                          if(flag_key != 0){
93                              if(table[i][j] == -1) {
94                                  clear = 1;
95                              }
96                              sum += table[i][j];
97                          }
98                      }
99                  }
100                 return clear ? -1 : sum;
101             }
```

3. Results and analysis 實驗結果與分析

3.1. Max7219 displayer

At first, I forgot to delete the global declaration of main in the assembly code, which cause multiple define.

3.2. Keypad Scanning



For the keypad, the output pins are PB13, PB14, PB15, PB1, and the input pins are PB3, PB5, PB4, PB10.

3.3. Single and multi buttons

Because of the circuit design, the scanning of pressing multi-button on the same row is not successful. We can do it by switching the input and output pin, and scan twice to check if the button is pressed.

4. Conclusions and ideas 心得討論與應用聯想

In this lab, I have learned more about other input/output devices. Though writing C seems to be easier, compare to assembly code, this lab still cost me a lot of time, such as clarify the concepts or the code in Lab5. The only thing I hope is that the exam tonight will go well.