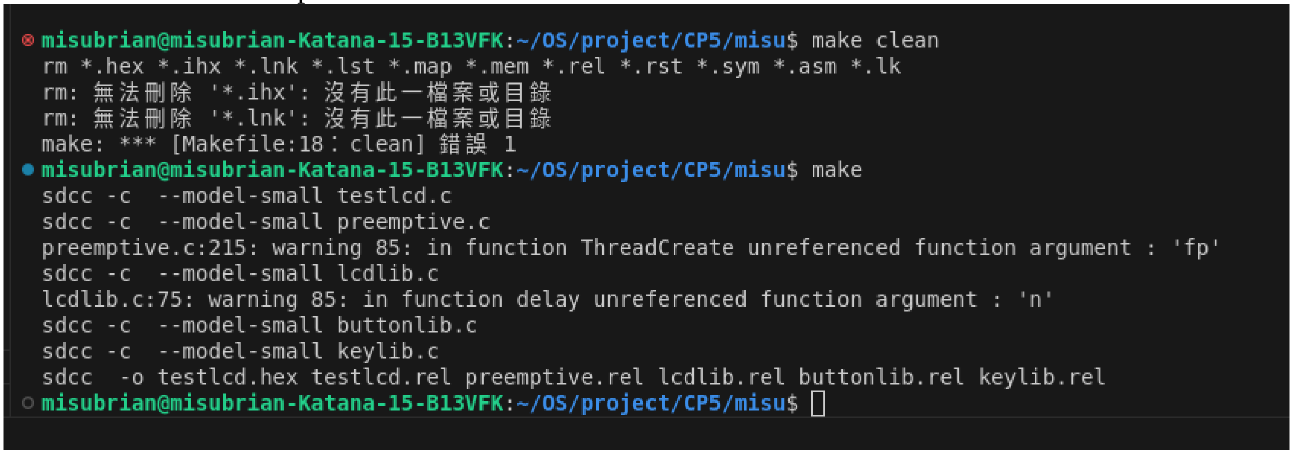
Report of Project Checkpoint 5

112062630 JING-YONG, SU 蘇敬詠

Part 1.

1. Screenshots for compilation

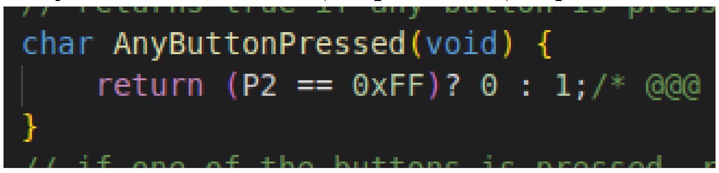


```
misubrian@misubrian-Katana-15-B13VFK:~/OS/project/CP5/misu$ make clean
rm *.hex *.ihx *.lnk *.lst *.map *.mem *.rel *.rst *.sym *.asm *.lk
rm: 無法刪除 '*.ihx': 沒有此一檔案或目錄
rm: 無法刪除 '*.lnk': 沒有此一檔案或目錄
make: *** [Makefile:18：clean] 錯誤 1
misubrian@misubrian-Katana-15-B13VFK:~/OS/project/CP5/misu$ make
sdcc -c  --model-small testlcd.c
sdcc -c  --model-small preemptive.c
preemptive.c:215: warning 85: in function ThreadCreate unreferenced function argument : 'fp'
sdcc -c  --model-small lcdlib.c
lcdlib.c:75: warning 85: in function delay unreferenced function argument : 'n'
sdcc -c  --model-small buttonlib.c
sdcc -c  --model-small keylib.c
sdcc  -o testlcd.hex testlcd.rel preemptive.rel lcdlib.rel buttonlib.rel keylib.rel
misubrian@misubrian-Katana-15-B13VFK:~/OS/project/CP5/misu$ 
```

2.

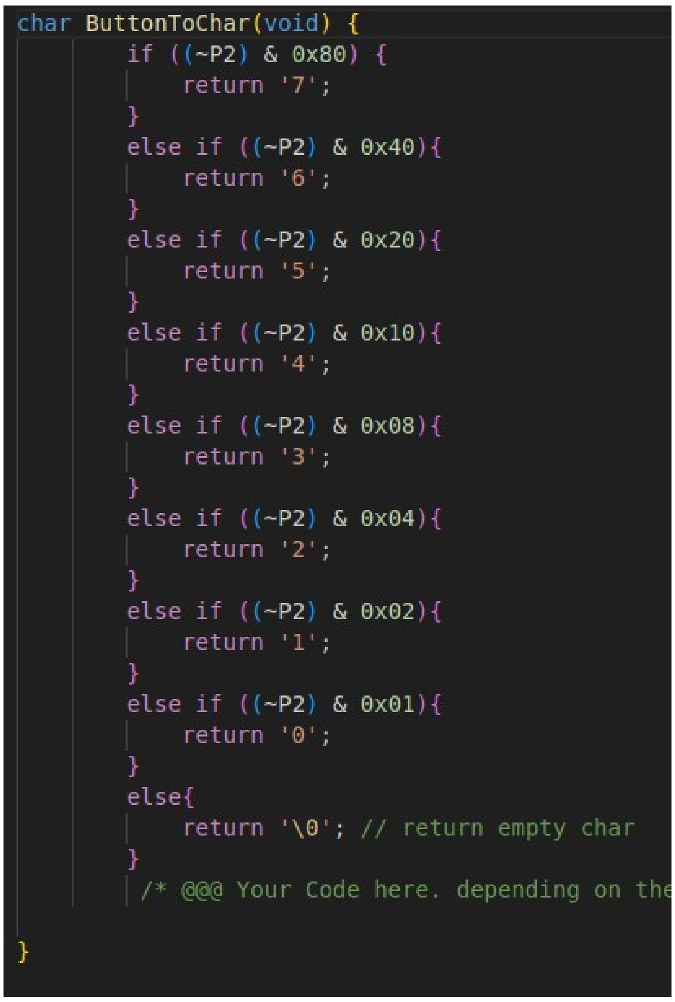
(a) Bottom Part

(i) Return TRUE if any of bit of P2 is zero(i.e. pull-down), equivalent with P2 != 0xFF.



```
char AnyButtonPressed(void) {
    return (P2 == 0xFF)? 0 : 1;/* @@@
}
```

(ii) ButtonToChar(): Check from MSB to LSB of P2 is 0 or not first. Therefore, we can always return the largest number even when several buttons are pushed.



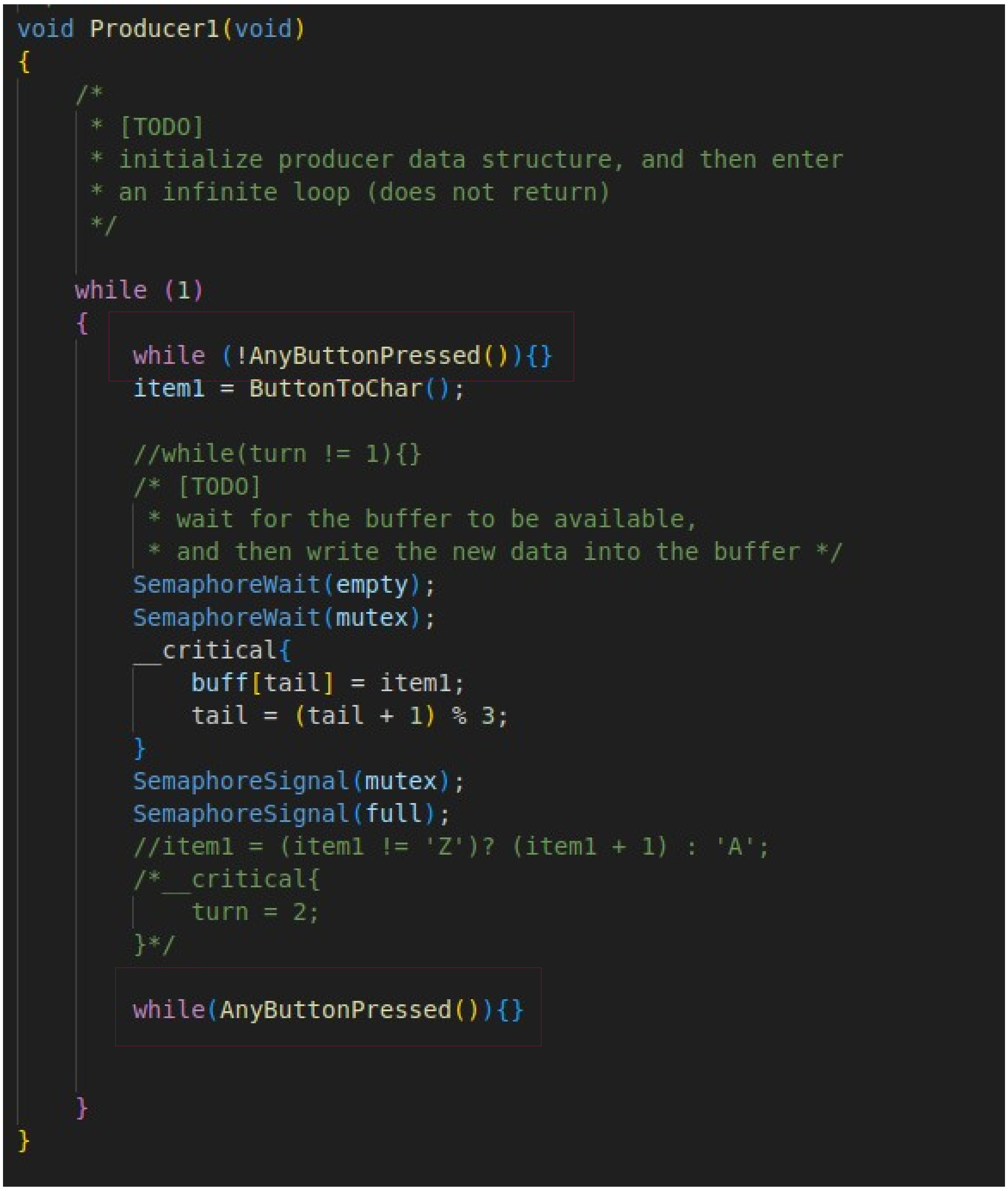
```c
char ButtonToChar(void) {
        if ((~P2) & 0x80) {
            return '7';
        }
        else if ((~P2) & 0x40){
            return '6';
        }
        else if ((~P2) & 0x20){
            return '5';
        }
        else if ((~P2) & 0x10){
            return '4';
        }
        else if ((~P2) & 0x08){
            return '3';
        }
        else if ((~P2) & 0x04){
            return '2';
        }
        else if ((~P2) & 0x02){
            return '1';
        }
        else if ((~P2) & 0x01){
            return '0';
        }
        else{
            return '\0'; // return empty char
        }
         /* @@@ Your Code here. depending on the

}
```

2. (b) Producer(1&2) function:

(I) Get input from button or keypad.

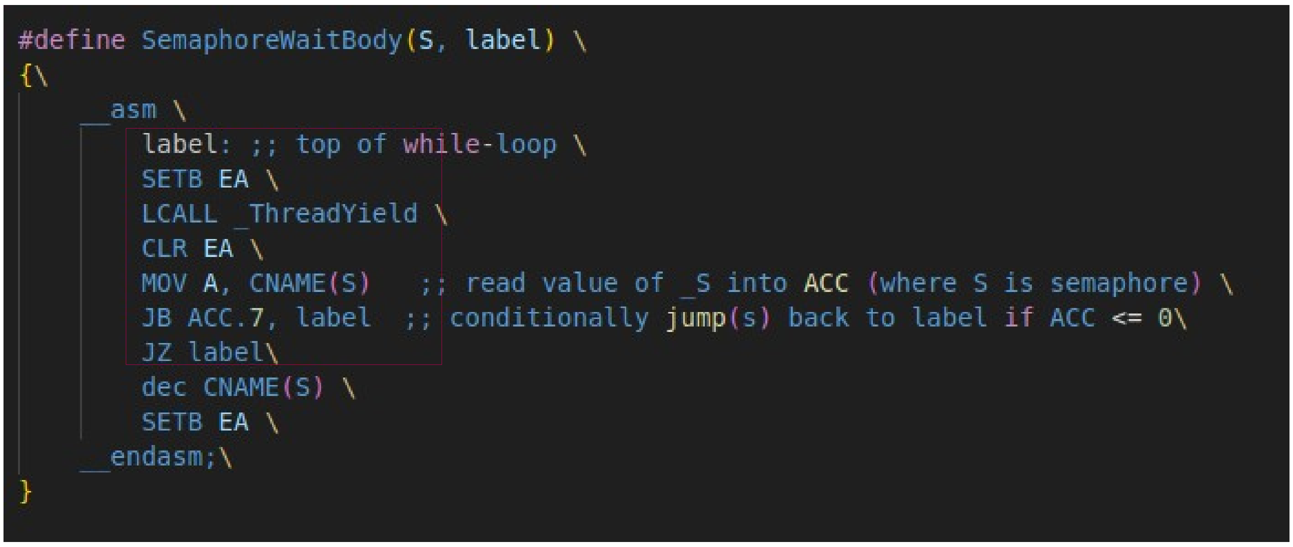
(ii) To avoid from repeatedly enqueue an number when button haven’t been released, I add the spin lock in front and end of the code.

(iii) Cancel the “turn” method done in CP4 because we can control the order and fairness by how to push the buttons.



```c
void Producer1(void)
{
    /*
     * [TODO]
     * initialize producer data structure, and then enter
     * an infinite loop (does not return)
     */

    while (1)
    {
        while (!AnyButtonPressed()){}
        item1 = ButtonToChar();

        //while(turn != 1){}
        /* [TODO]
         * wait for the buffer to be available,
         * and then write the new data into the buffer */
        SemaphoreWait(empty);
        SemaphoreWait(mutex);
        __critical{
            buff[tail] = item1;
            tail = (tail + 1) % 3;
        }
        SemaphoreSignal(mutex);
        SemaphoreSignal(full);
        //item1 = (item1 != 'Z')? (item1 + 1) : 'A';
        /*__critical{
            turn = 2;
        }*/

        while(AnyButtonPressed()){}


    }
}
```

3. Modification on SemaphoreWaitBody:
Disable the interrupt to avoid race condition. Moreover, yield when it failed to get the semaphore.
The red rectangle below represents the section of while loop.



```
#define SemaphoreWaitBody(S, label) \
{\
    __asm \
        label: ;; top of while-loop \
        SETB EA \
        LCALL _ThreadYield \
        CLR EA \
        MOV A, CNAME(S)    ;; read value of _S into ACC (where S is semaphore) \
        JB ACC.7, label  ;; conditionally jump(s) back to label if ACC <= 0\
        JZ label\
        dec CNAME(S) \
        SETB EA \
    __endasm;\
}
```

4. Notes:
My assignment runs properly with a single producer, and I can see the keys I press displayed on the LCD. However, I regret that my program does not function correctly with two producers. Nevertheless, I have submitted it for evaluation, leaving it to the discretion of the teaching assistants. I respect their decisions and am very grateful to the TAs who actively and patiently assisted and discussed the issues with me.