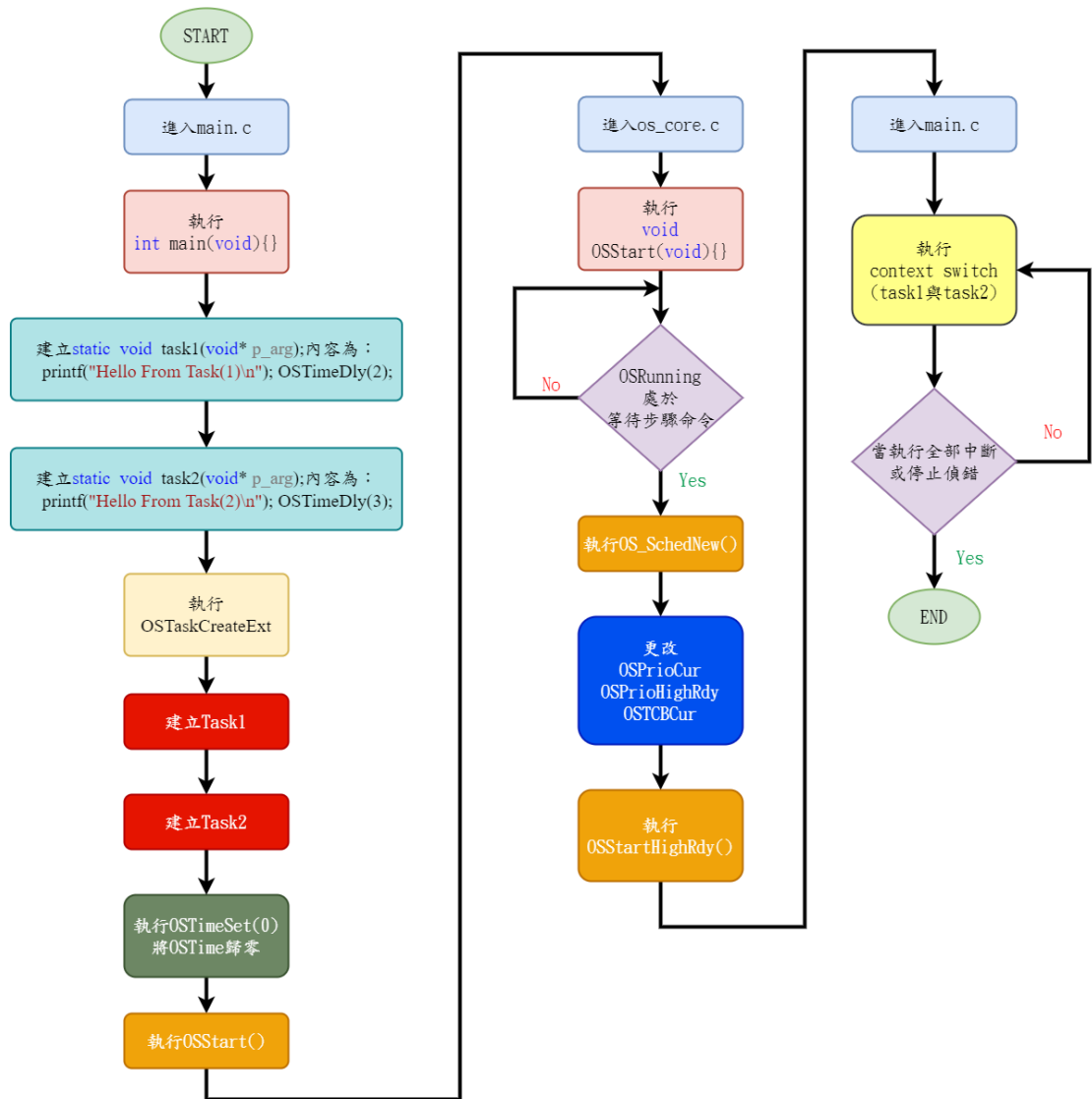


The system flow and the explanation of the process(functions). (45%)

1. 進入 main.c 。
2. 執行 `int main (void){}` 。
3. 建立 `static void task1(void* p_arg);`
內容為：`printf("Hello From Task(1)\n");`
`OSTimeDly(2);`
4. 建立 `static void task2(void* p_arg);`
內容為：`printf("Hello From Task(2)\n");`
`OSTimeDly(3);`
5. 執行 `OSTaskCreateExt`
6. 建立 task1 。
7. 建立 task2 。
8. 透過 `OSTimeSet(0);` 將 `OSTime` 歸零。
9. 執行 `OSStart()` 。
10. 進入 os_core.c 。
11. 執行 `void OSStart (void){}` 。
12. 如果 `OSRunning` 處於等待步驟命令時開始執行 `OS_SchedNew()` 。
13. 執行完 `OS_SchedNew();` 之後更改 `OSPrioCur`、`OSTCBHighRdy`、`OSTCBCur` 三個變數。
14. 執行 `OSStartHighRdy ()` 。
15. 進入 main.c 。
16. 執行 context switch
`(void task1(void* p_arg) {} ↔ void task2(void* p_arg) {})` 。
- 列印 "Hello From Task(1)"，執行 `OSTimeDly(2)` 之後再
列印 "Hello From Task(2)"，執行 `OSTimeDly(3)`
17. task1 與 task2 的 Delay 時間分別為 2 與 3 個單位，故此程式則會依照
`OSTime` 為 2 的倍數執行 task1，3 的倍數時執行 task2，6 的倍數時執行
task1、task2 。
18. 當執行全部中斷或停止偵錯時程式結束，否則重複執行 16 。

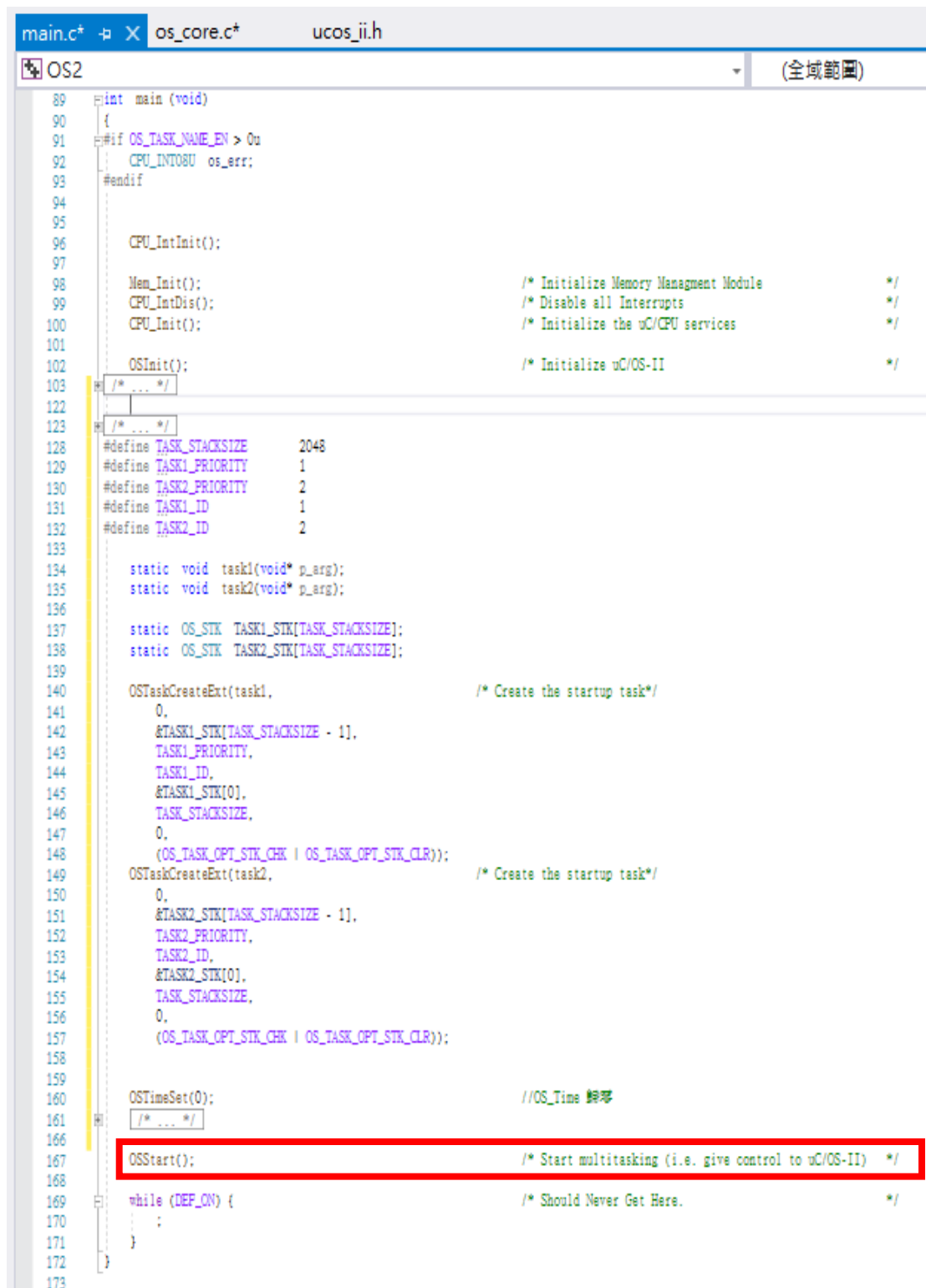


The screenshot of the result. (10%)

```
C:\Users\MMLAB\Desktop\20201014\test01\Mic
OSTick    created, Thread ID 1360
Task[ 63] created, Thread ID 8816
Task[  1] created, Thread ID 14156
Task[  2] created, Thread ID 10344
Tick      Form Task      To Task
0         *****      task(1)
0         task(1)        task(2)
0         task(2)        task(63)
3         task(63)       task(1)
3         task(1)        task(63)
6         task(63)       task(1)
6         task(1)        task(2)
6         task(2)        task(63)
9         task(63)       task(1)
9         task(1)        task(63)
12        task(63)       task(1)
12        task(1)        task(2)
12        task(2)        task(63)
15        task(63)       task(1)
15        task(1)        task(63)
18        task(63)       task(1)
18        task(1)        task(2)
18        task(2)        task(63)
21        task(63)       task(1)
21        task(1)        task(63)
24        task(63)       task(1)
24        task(1)        task(2)
24        task(2)        task(63)
27        task(63)       task(1)
27        task(1)        task(63)
30        task(63)       task(1)
30        task(1)        task(2)
30        task(2)        task(63)
33        task(63)       task(1)
33        task(1)        task(63)
```

A report that describes your implementation (please attach the screenshot of the code and MARK the modified part). (45%)

1. 先從 OSStart();的程式開始查看定義



```
main.c* x os_core.c* ucos_ii.h
OS2 (全域範圍)
89 int main(void)
90 {
91     #if OS_TASK_NAME_EN > 0u
92         CPU_INT08U os_err;
93     #endif
94
95     CPU_IntInit();
96
97     Mem_Init(); /* Initialize Memory Management Module */
98     CPU_IntDis(); /* Disable all Interrupts */
99     CPU_Init(); /* Initialize the uC/CPU services */
100
101     OSInit(); /* Initialize uC/OS-II */
102     /* ... */
103
104     /* ... */
105
106     #define TASK_STACKSIZE 2048
107     #define TASK1_PRIORITY 1
108     #define TASK2_PRIORITY 2
109     #define TASK1_ID 1
110     #define TASK2_ID 2
111
112     static void task1(void* p_arg);
113     static void task2(void* p_arg);
114
115     static OS_STK TASK1_STK[TASK_STACKSIZE];
116     static OS_STK TASK2_STK[TASK_STACKSIZE];
117
118     OSTaskCreateExt(task1, /* Create the startup task*/
119         0,
120         &TASK1_STK[TASK_STACKSIZE - 1],
121         TASK1_PRIORITY,
122         TASK1_ID,
123         &TASK1_STK[0],
124         TASK_STACKSIZE,
125         0,
126         (OS_TASK_OPT_STK_CHK | OS_TASK_OPT_STK_CLR));
127     OSTaskCreateExt(task2, /* Create the startup task*/
128         0,
129         &TASK2_STK[TASK_STACKSIZE - 1],
130         TASK2_PRIORITY,
131         TASK2_ID,
132         &TASK2_STK[0],
133         TASK_STACKSIZE,
134         0,
135         (OS_TASK_OPT_STK_CHK | OS_TASK_OPT_STK_CLR));
136
137     OSTimeSet(0); //OS_Time 歸零
138     /* ... */
139
140     OSStart(); /* Start multitasking (i.e. give control to uC/OS-II) */
141
142     while (DEF_ON) { /* Should Never Get Here. */
143         ;
144     }
145 }
```

2. 在 `if (OSRunning == OS_FALSE) {}` 中看見了本次作業需要的參數 `OSPrioCur` 與 `OSTCBHighRdy`

```

if (OSRunning == OS_FALSE) {
    OS_SchedNew();
    OSPrioCur = OSPrioHighRdy;
    OSTCBHighRdy = OSTCBPrioTbl[OSPrioHighRdy]; /* Point to highest priority task ready to run */
    OSTCBCur = OSTCBHighRdy;
    OSStartHighRdy(); /* Execute target specific code to start task */
}

/*
*****
*
* HWOO find
*
*/
OS_EXT INT8U OSPrioCur; /* Priority of current task */
OS_EXT INT8U OSPrioHighRdy; /* Priority of highest priority task */
/*
*****
*
* HWOO find
*
*/

```

3. 開始一行一行觀看 `if (OSRunning == OS_FALSE) {}` 中函數的定義
4. 從 `static void OS_SchedNew (void){}` 中可以發現 `OSTCBHighRdy` 的更新位置在這裡

```

static void OS_SchedNew (void)
{
    #if OS_LOWEST_PRIO <= 63u /* See if we support up to 64 tasks */
        INT8U y;

        y = OSUnMapTbl[OSRdyGrp];
        OSPrioHighRdy = (INT8U)((y << 3u) + OSUnMapTbl[OSRdyTbl[y]]);

        /*
        *****
        *
        * HWOO
        *
        */

        if (OSPrioCur == 0) { /*如果OSPrioCur==0 也就是還沒有開始執行 */
            printf("Tick\t Form Task\t To Task\n"); /*print表頭 */
            printf("%d\t %s\t task(%d)\n", OSTimeGet(), "*****", OSPrioHighRdy); /*print第一行 */
        }
        else if (OSPrioCur != OSPrioHighRdy) { /*如果OSPrioCur不等於最高優先前的 */
            printf("%d\t task(%d)\t task(%d)\n", OSTimeGet(), OSPrioCur, OSPrioHighRdy);
        }

        /*
        *****
        *
        * HWOO
        *
        */
    }
}

```

5. 再來就是經由不斷嘗試得出最精簡的 code 以及符合題目要求的排版
6. END