**嵌入式作業系統實作**

**Embedded OS Implementation**

**PA\_2**

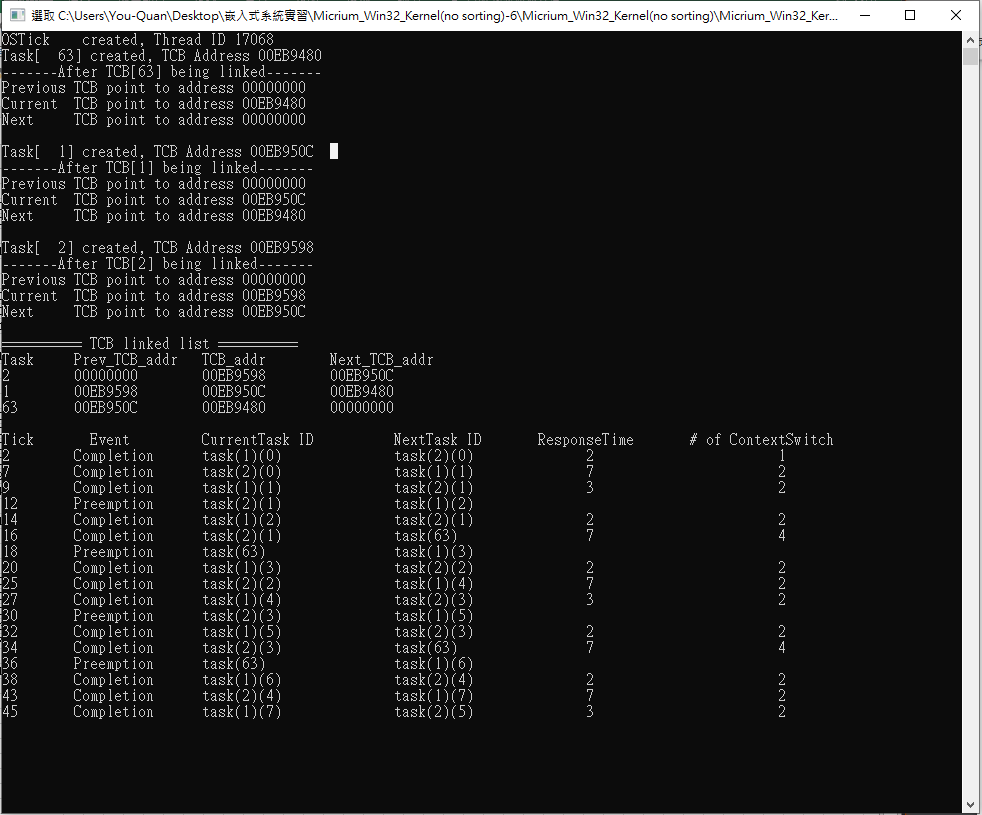
指導教授: 陳雅淑 教授

課程學生: M10907314張祐銓

**[ PART I] EDF Scheduler Implementation [60%]**

* The screenshot results (with the given format) of two task sets. (Tick 0 to tick 40 or the tick when a task missing the deadline) (10%)

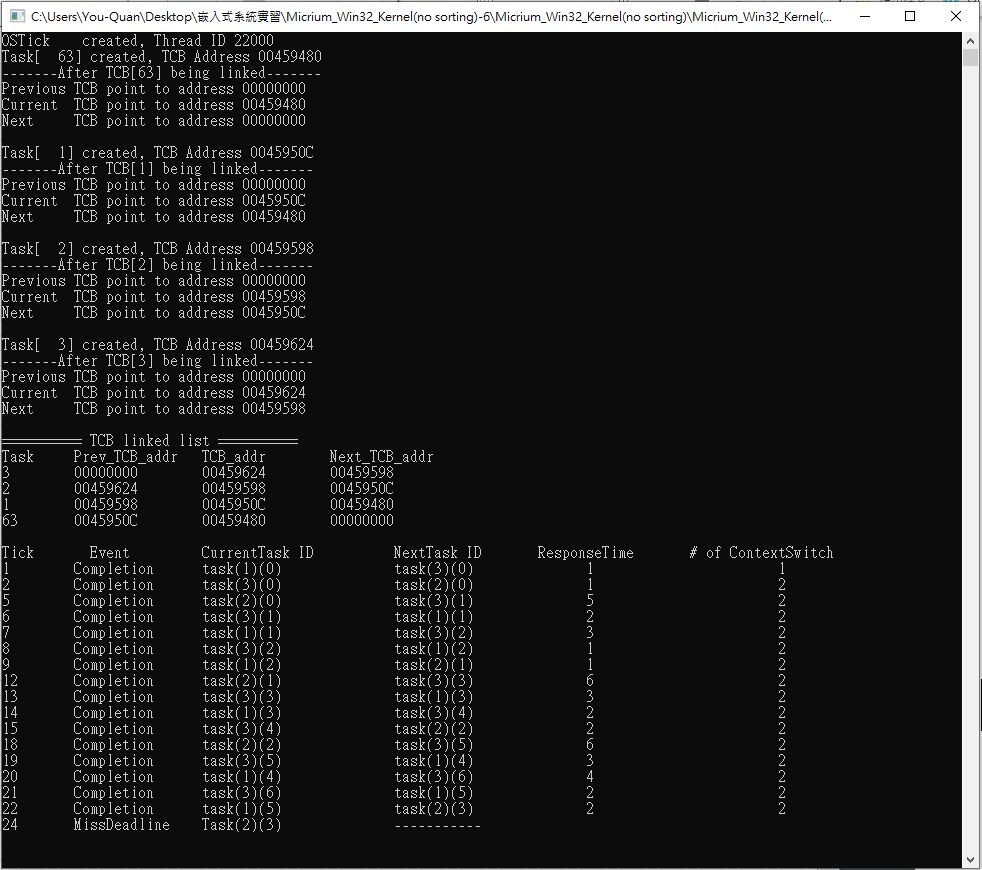
Task set 1 = {τ1 (0, 2, 6), τ2 (0, 5, 9)}



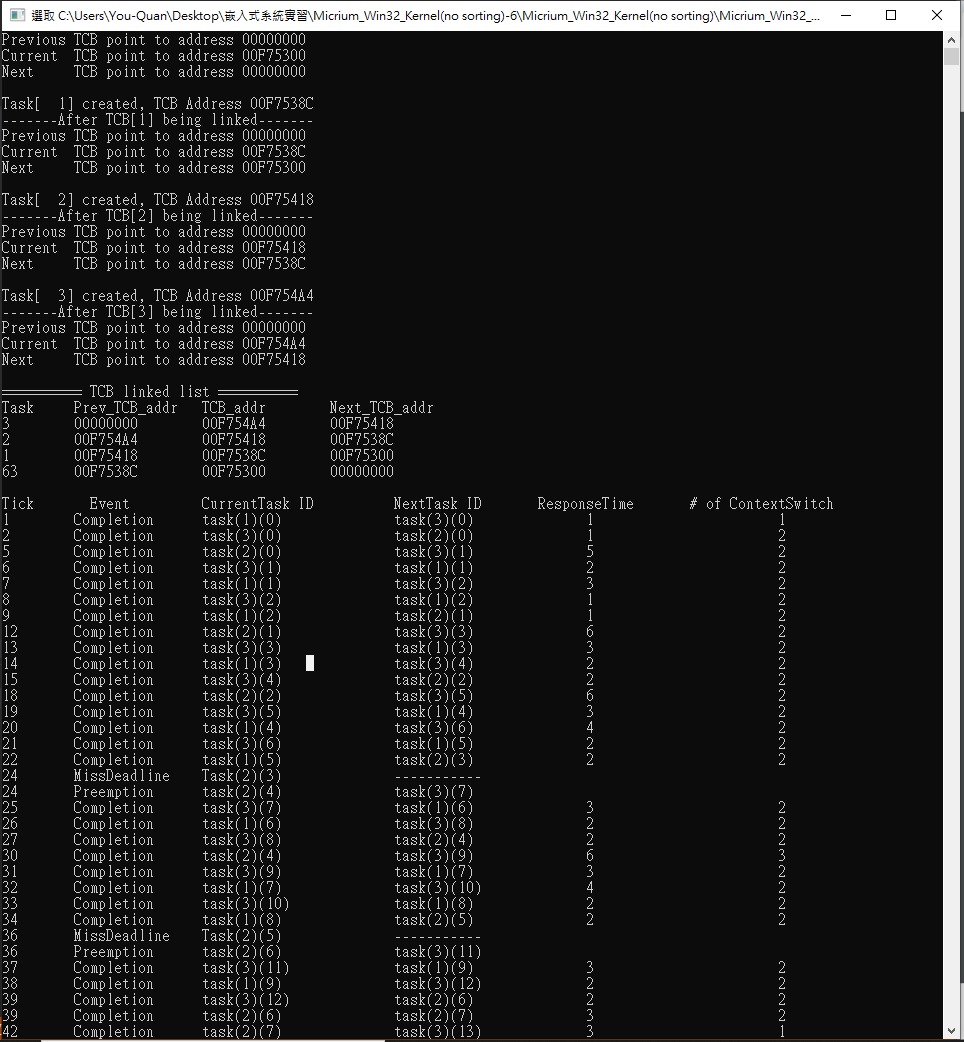
Task set 2 = {τ1 (0, 1, 4), τ2 (0, 3, 6), τ3 (1, 1, 3)}

做了兩個版本

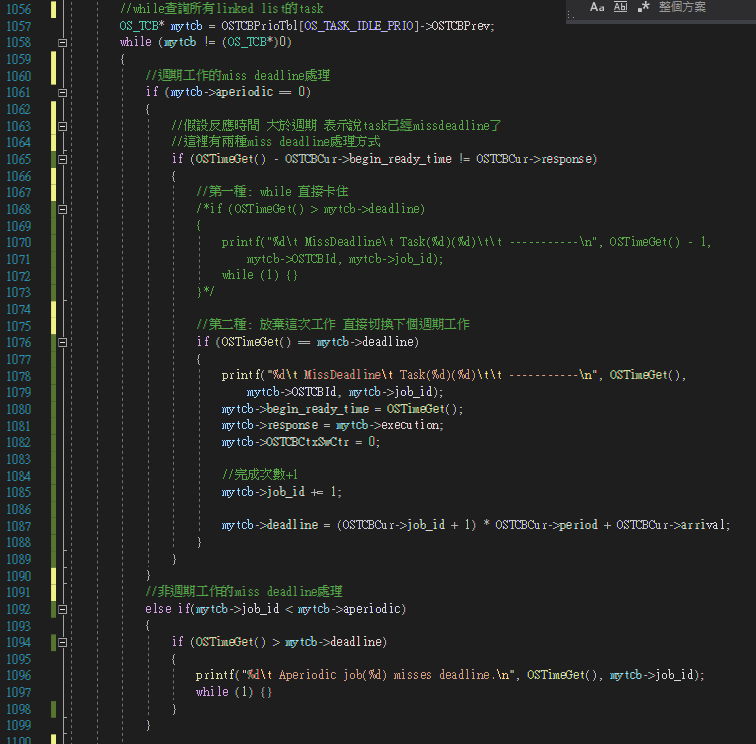
1. Missdeadline就卡while



2. 放棄掉這次的missdeadline task



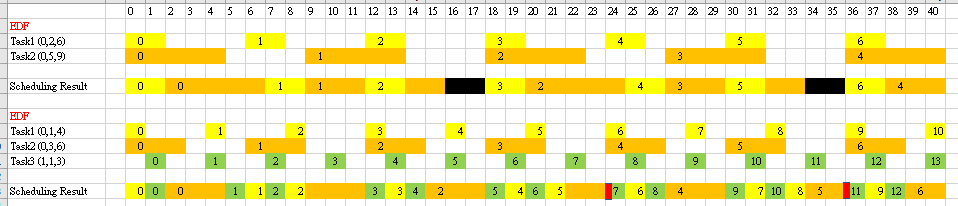
* Implement and describe how to handle the deadline missing situation under EDF. (10%)



判斷miss deadline的程式打在OSTimeTick()下，只要OSTimeGet() 大於 某個task的deadline就判斷為miss deadline，之後做了兩種處理miss deadline的版本。

1. 直接卡一個while，讓整個系統停下來。
2. 放棄這次的task工作，將OS\_TCB的變數設定為下個週期工作的數值，之後過OSTimeTick()會進入OSIntExit()進行context switch，讓系統繼續運作下去。

* A report that describes your implementation, including scheduling results of two task sets, modified functions, data structure, etc. (please ATTACH the screenshot of the code and MARK the modified part) (40%)



上次RMS的實現條列說明:

1. 在OS\_TCB中新增變數

INT32U begin\_ready\_time; //紀錄task變ready的時間點

INT32U response; //反應時間

INT32U arrival; //到達時間

INT32U execution; //執行時間

INT32U period; //週期

INT32U job\_id; //工作次數

1. 在OSTimeTick中去察看目前的ReadyTable有哪些Task正在Ready狀態，把所有正在Ready且不是目前正在執行的task的所有task的response都加一，因為task正在Ready卻沒有被執行表示被延後了一個tick，response一開始初始值是週期內的執行時間，所以task工作執行只要沒有執行滿response的時間就會一直卡在while迴圈內。
2. Task執行完會執行OS\_Sched()去切換給下個Task，所以在OS\_Sched()中顯示完成的字串。
3. 當TimeTick中斷產生之後會進OSIntExit()把低優先權的task中斷給高優先權的task，所以在這裡顯示中斷狀態的字串。
4. 當有task已經快要完成時有可能會被其他優先權高的task給搶占，所以會在OSIntExit()增加判斷程式迴避掉那次的context switch，讓快要做完的task先完成它的工作。
5. 當有一個task完成了一個周期內的工作後要做下個週期的工作時因為不會進入OS\_Sched()做context switch，所以在OSTimeDly(0)時表示task的反應時間等於週期，在OSTimeDly內去print完成的字串。
6. 判斷是否有task已經miss deadline，只要在OSTimeTick中去判斷reponse是否大於period，表示說反應時間已經大於週期就是miss deadline。

EDF的實現條列說明:

1. 在OS\_TCB中新增變數

INT32U deadline; //EDF的最後期限

1. OS\_SchedNew()會去找下個要執行的task，將EDF找下個task的程式寫在OS\_SchedNew()中，根據所有task的deadline大小決定哪個task要優先執行，最後直接改動OSPrioHighRdy變成我們希望執行的task。
2. 在OSTimeDly(0)時為task剛好完成在deadline前，在OSTimeDly(0)時執行OS\_Sched做排程。
3. 在OSIntExit()在task快要完成時，將task的deadline先切換到下個deadline時間，判斷下個要換哪個task執行，先切換是因為task快完成時會迴避掉這次的context switch。

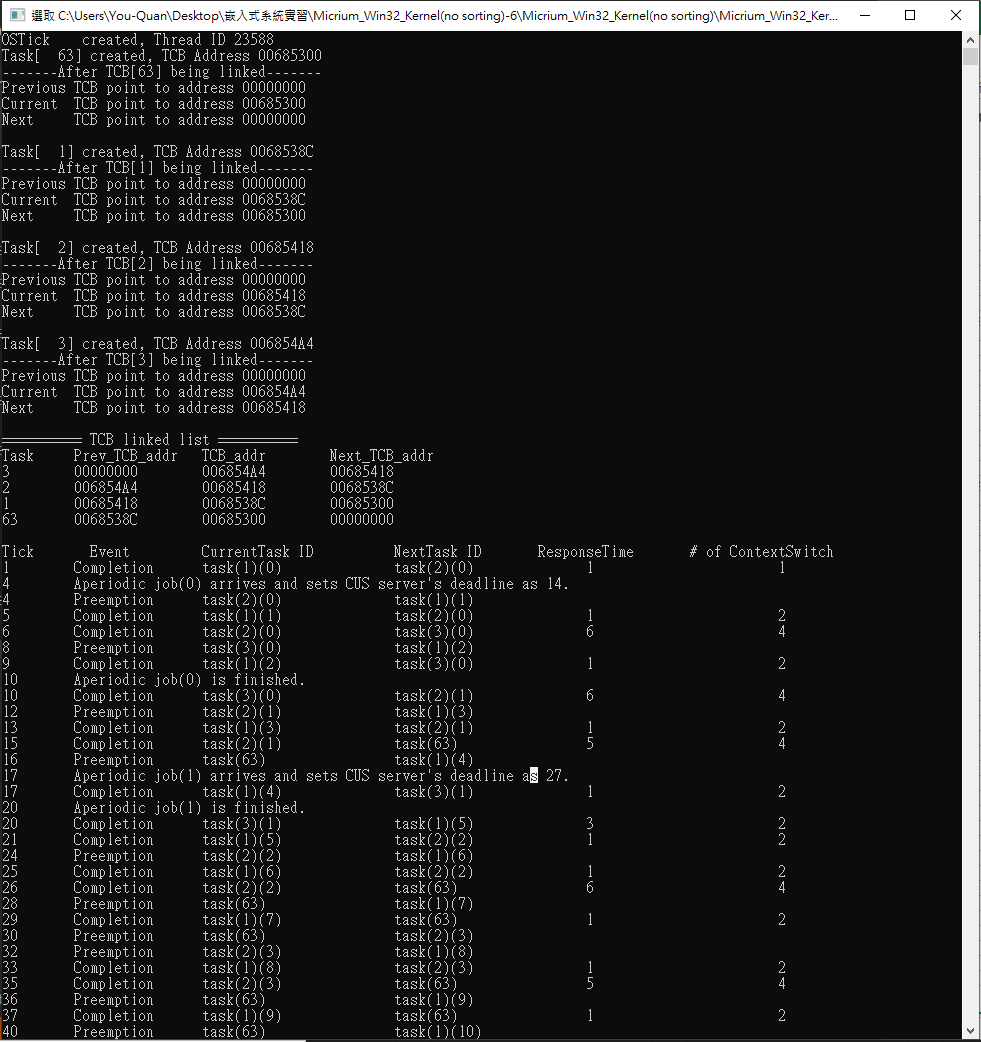
* 程式截圖部分跟CUS的程式截圖整合在一起

**[ PART II] CUS Scheduler Implementation [40%]**

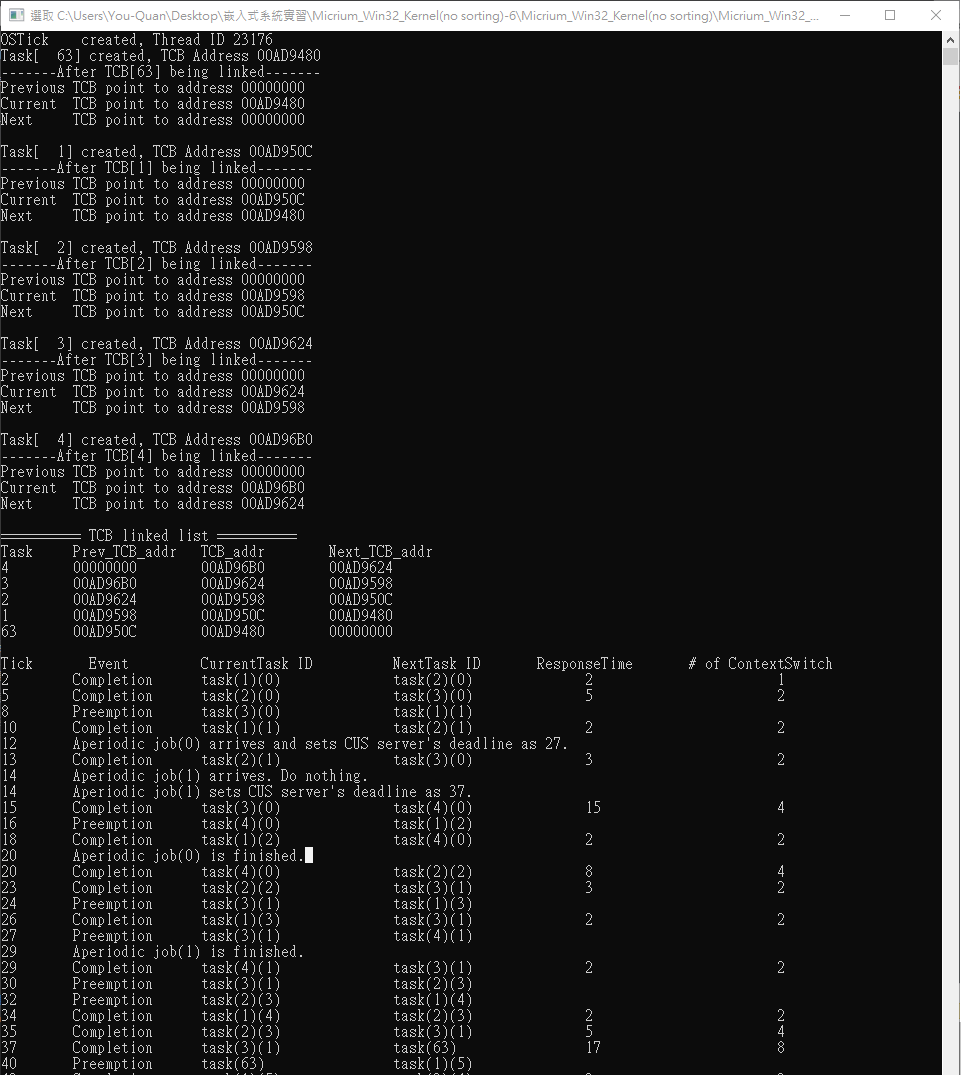
* The screenshot results (with the given format) of two task sets. (Tick 0 to tick 40 or the tick when a task missing the deadline). (10%)

**======================== Task Set 1 ======================== Periodic Task Set1 = {τ1 (0, 1, 4), τ2 (0, 4, 10), τ3\_ServerSize (0.3)}**

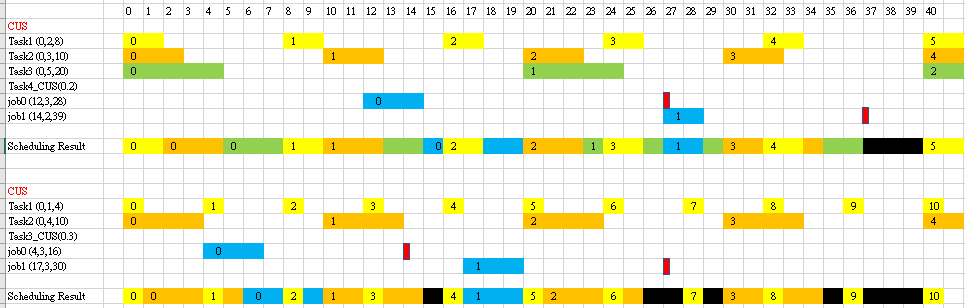
**Aperiodic Jobs Set1 = {j0 (4, 3, 16), j1 (17, 3, 30)}**



**======================== Task Set 2 ======================== Periodic Task Set2 = {τ1 (0, 2, 8), τ2 (0, 3, 10), τ3 (0, 5, 20), τ4\_ServerSize (0.2)} Aperiodic Jobs Set2 = {j0 (12, 3, 28), j1 (14, 2, 39)}**

****

* A report that describes your implementation, including scheduling results of two task sets, modified functions, data structure, etc. (please ATTACH the screenshot of the code and MARK the modified part). (30%)



CUS實現方式條列說明:

1. 在OS\_TCB中新增變數

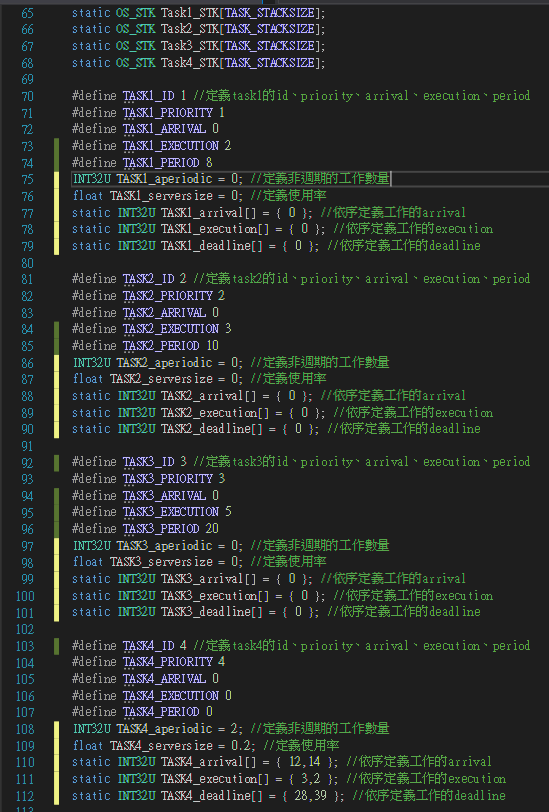
INT32U aperiodic; //非週期的工作數量

float aperiodic\_serversize; //非週期的使用率

INT32U \*aperiodic\_arrival; //非週期的到達時間陣列指標

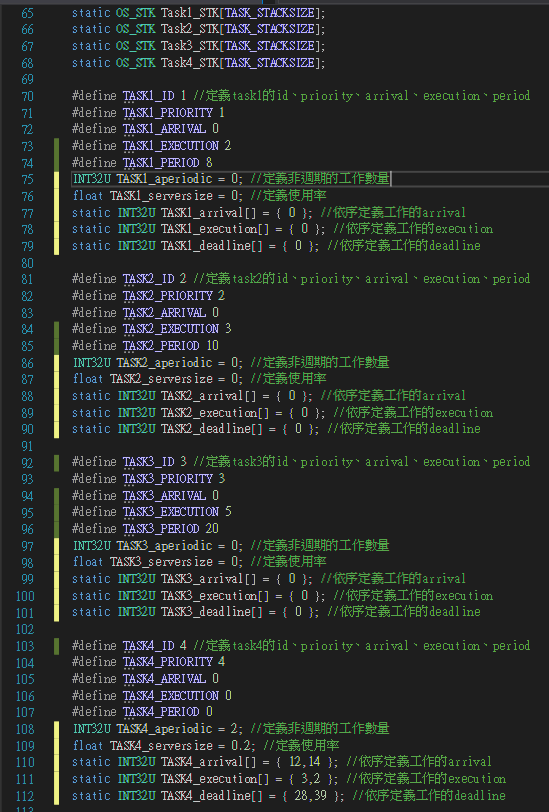
INT32U \*aperiodic\_execution; //非週期的執行時間陣列指標

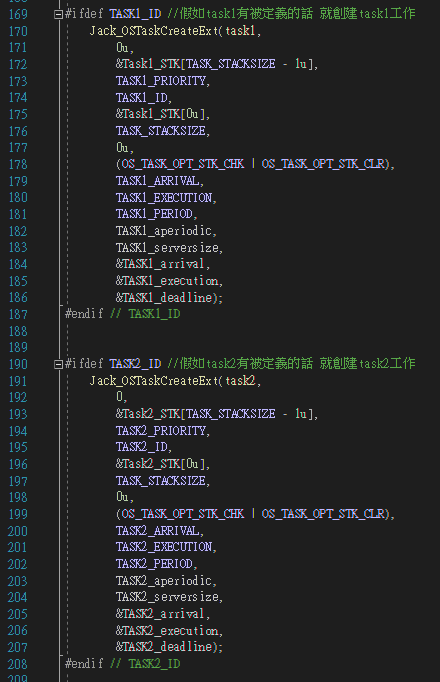
INT32U \*aperiodic\_deadline; //非週期給定的最後期限陣列指標

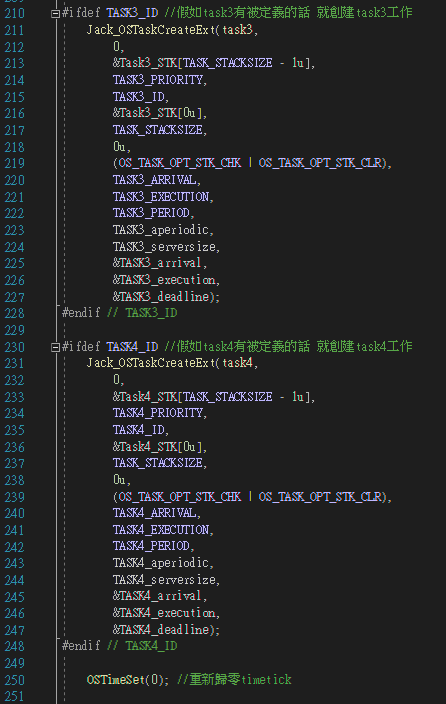
INT32U aperiodic\_count; //查看下個task是否要到達的變數

1. 非週期的period = execution / serversize。
2. 真正arraival的時間點 = max(上個工作deadline,這次預設的arrival)。
3. 當完成設定的工作數量後，OS\_SchedNew()就不會再把非週期的這個task排進去。
4. 當非週期的工作完成時，OS\_TCB內的變數會設定到下個工作變數。
5. 在OSTimeTick()內判斷當預設的arrival到達時，是否在前一個task的deadline前，再顯示對應的相關字串。

Main.c

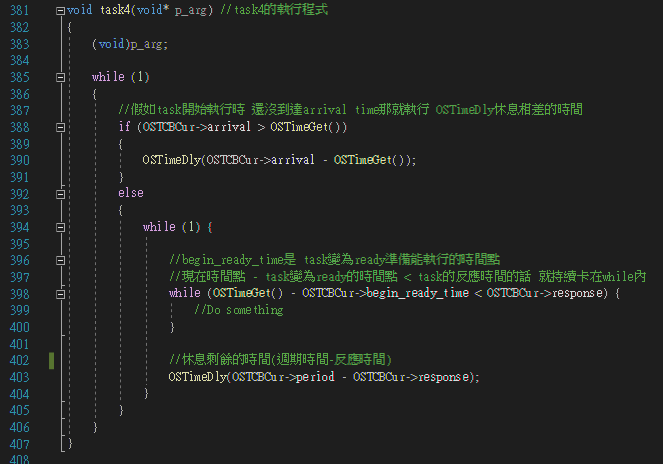
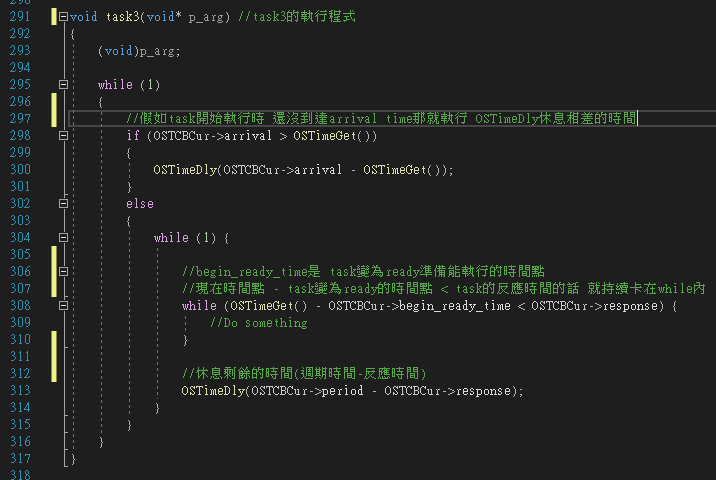








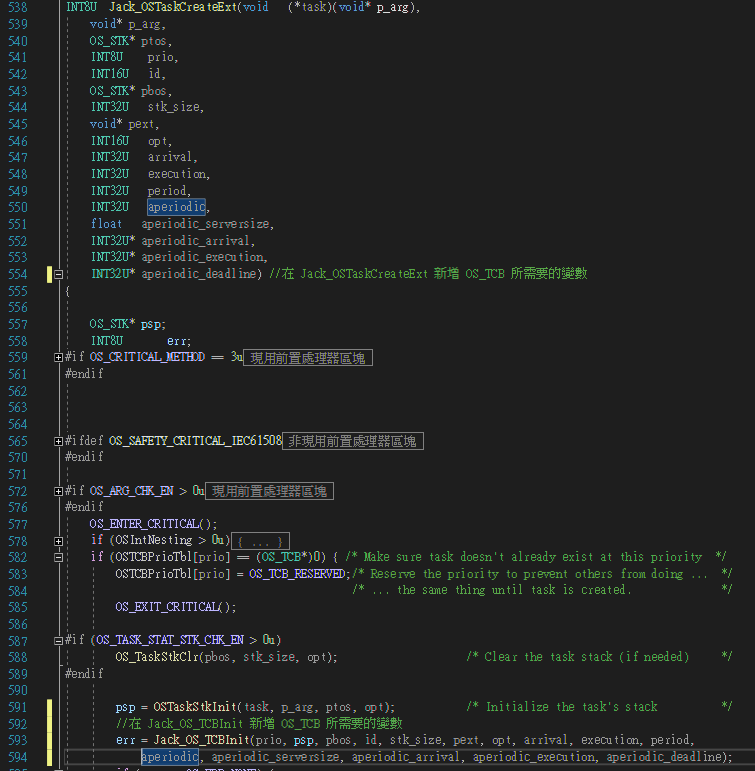




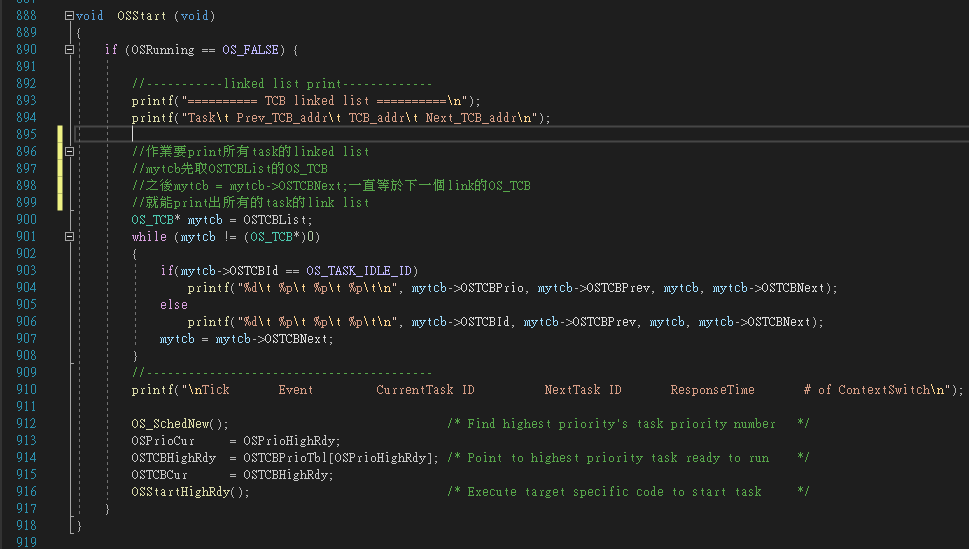
ucos\_ii.h

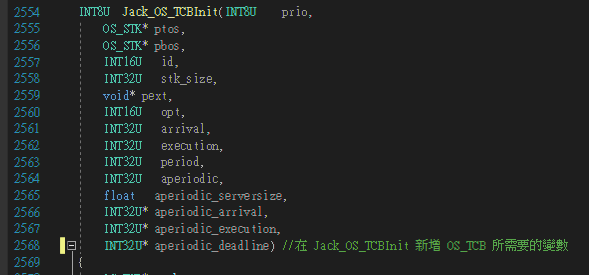


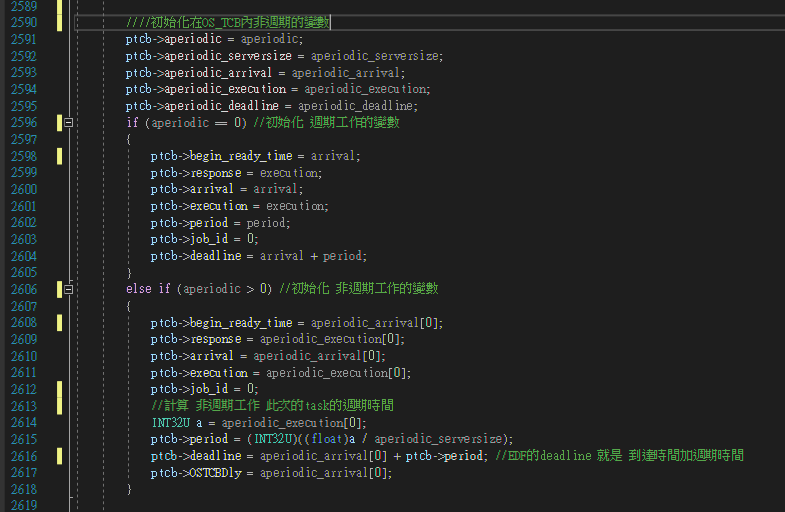
os\_task.c



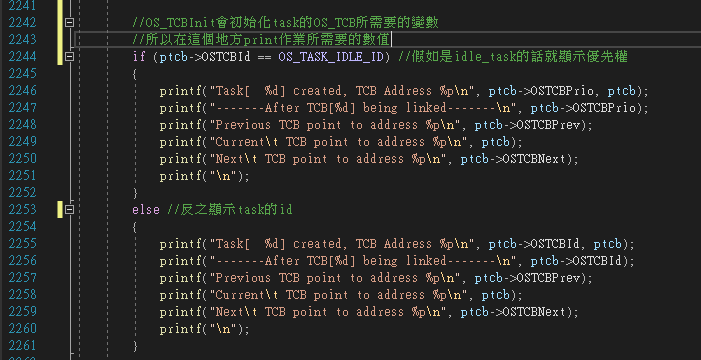
os\_core.c



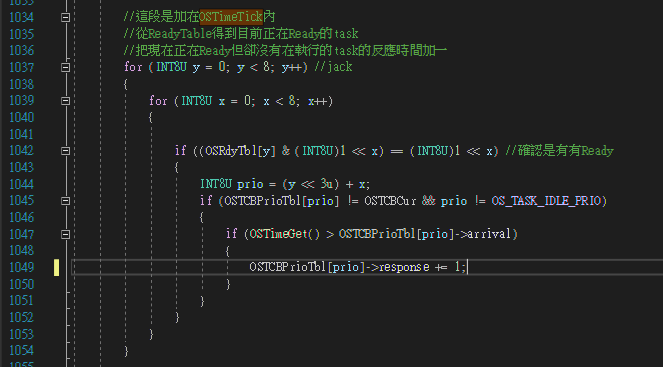




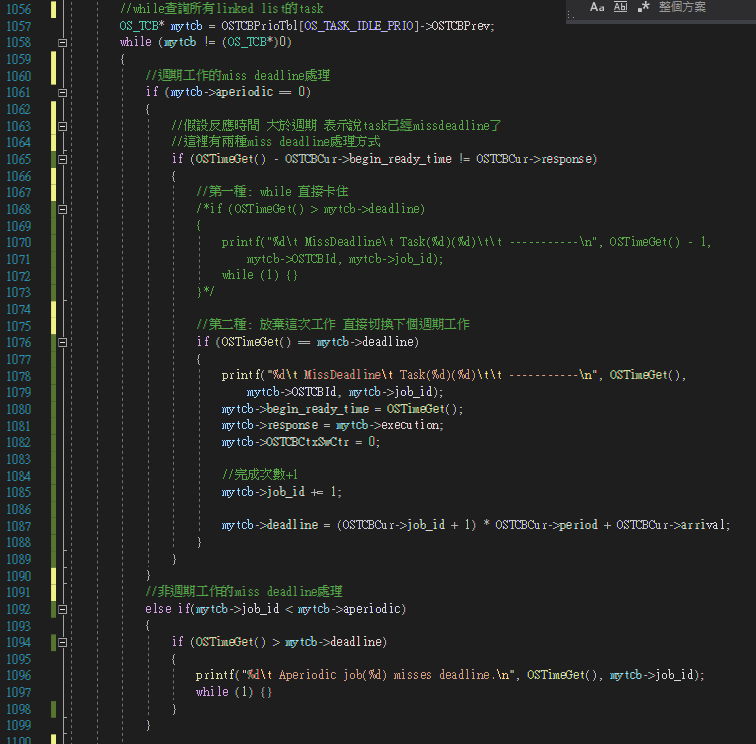
Jack\_OS\_TCBInit()



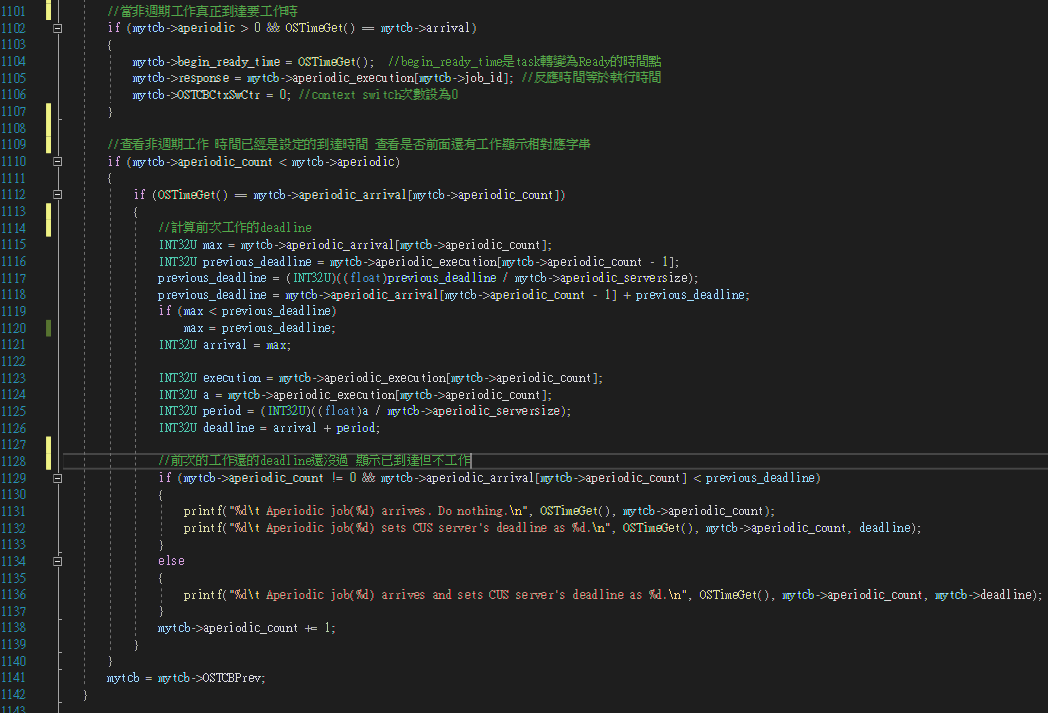
Jack\_OS\_TCBInit()



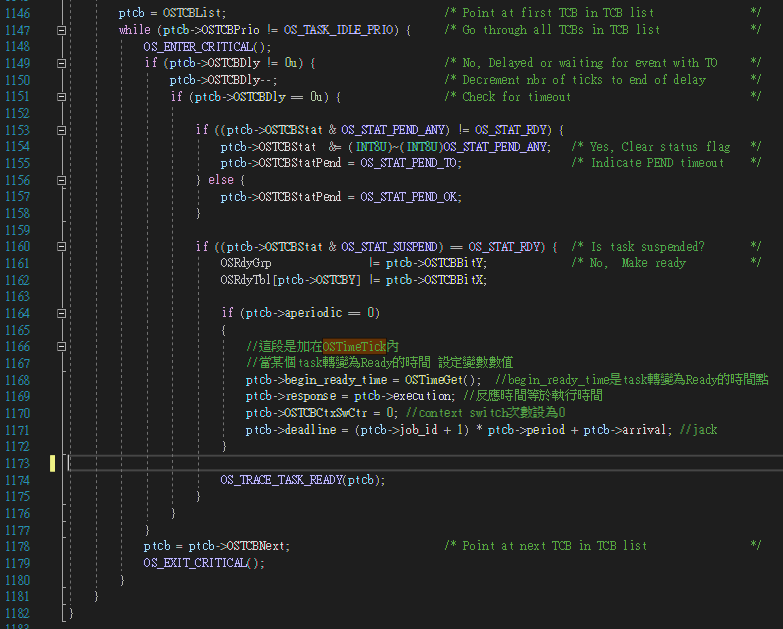
OSTimeTick()



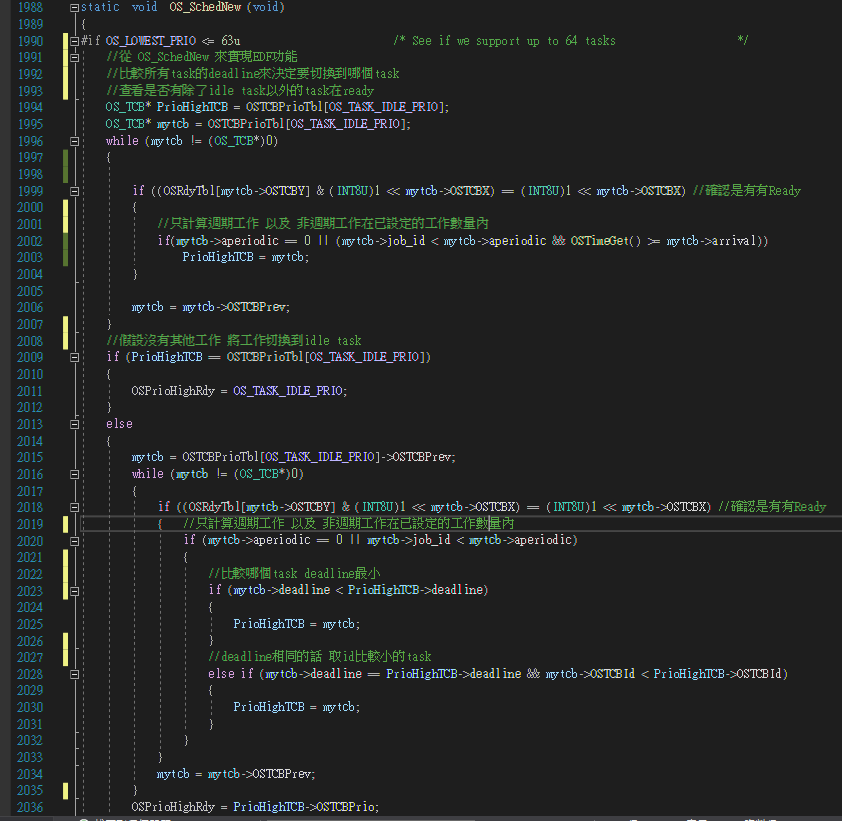
OSTimeTick()

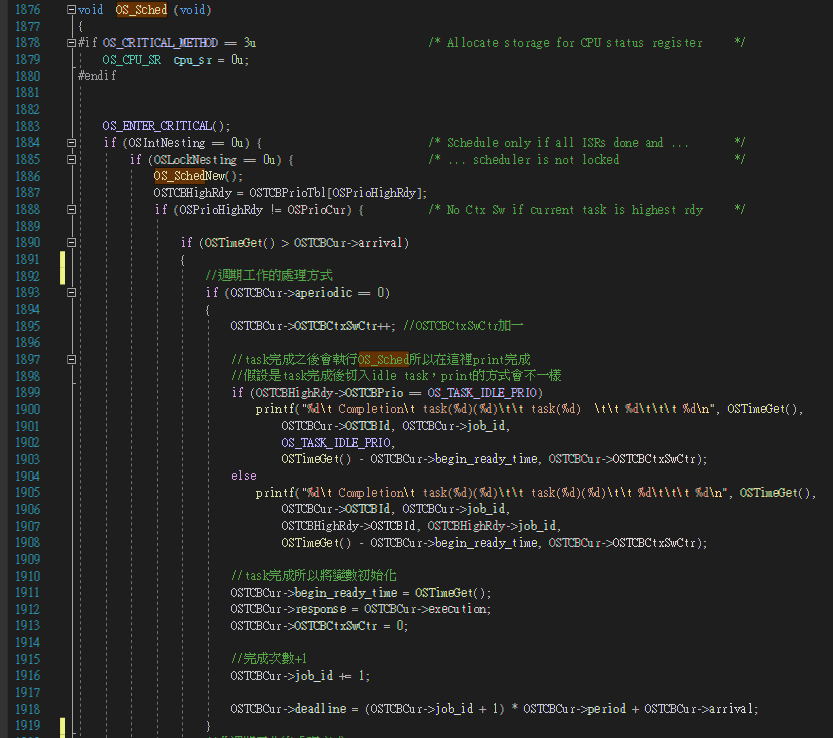


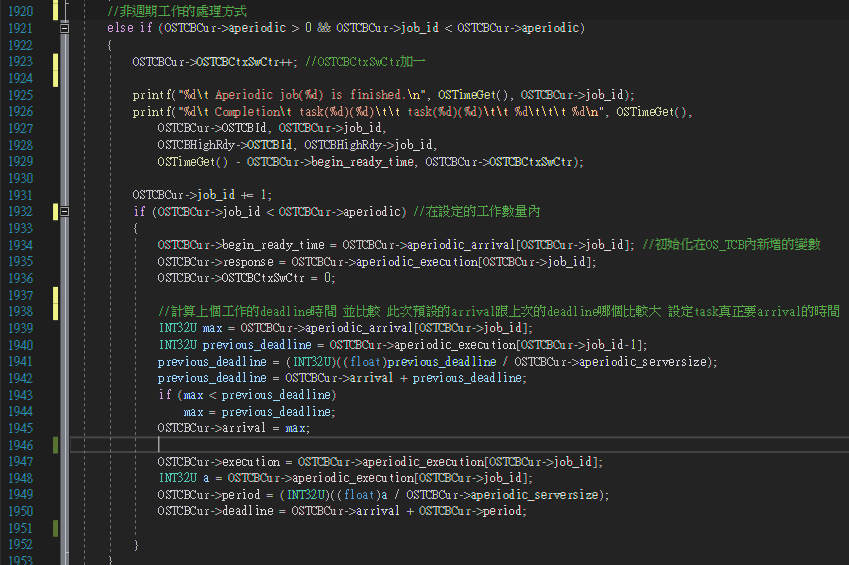
OSTimeTick()

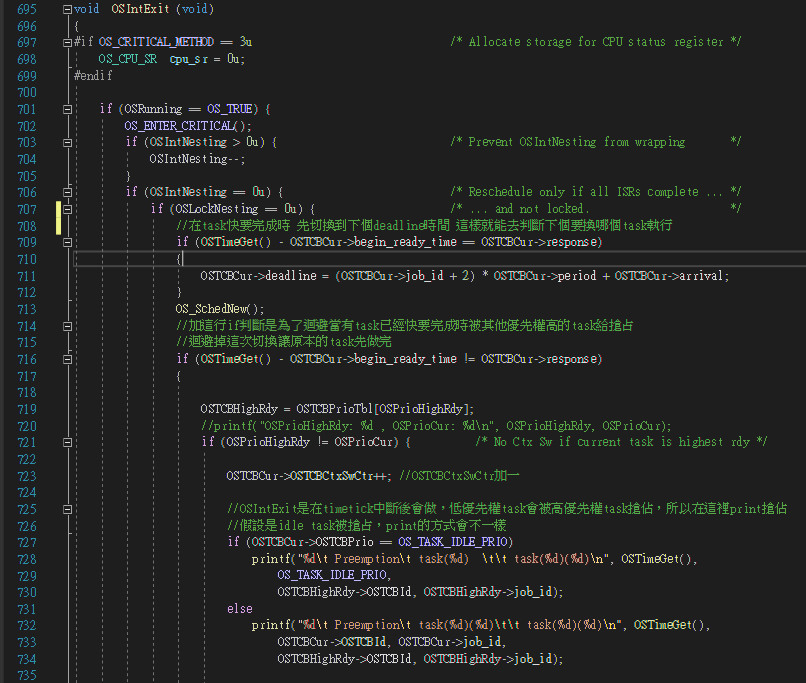


OSTimeTick()

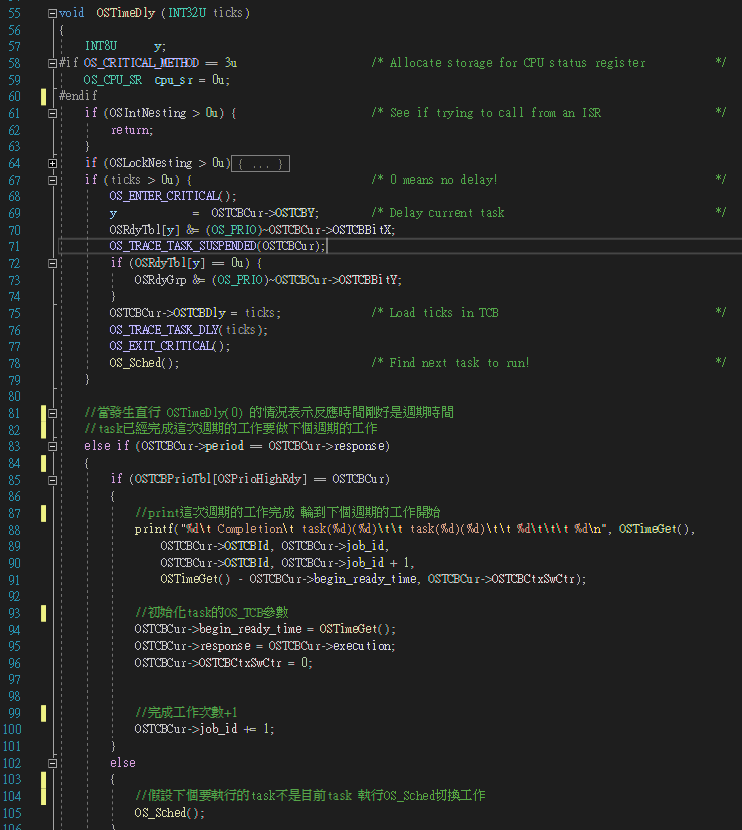




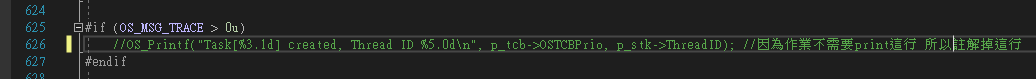




os\_time.c



os\_cpu\_c.c



os\_cpu\_c.c、main.c、os\_time.c、ucos\_ii.h、os\_task.c、os\_core.c