Embedded OS Implementation, Fall 2022

Project #1 (due November 2nd, 2022 (Wednesday) at 12:00)

[PART I] Task Control Block Linked List

Objective:

Following the previous homework (HW1), please add some code to the μC/OS-II scheduler **in the kernel level** to observe the operations of the task control block (TCB) and TCB linked list.

※ The TCB address is dynamic.

The output results are shown below:

```
OSTick    created, Thread ID 15568
Task[63] created, TCB Address   bdf680
------After TCB[63] being linked------
Previous TCB point to address         0
Current  TCB point to address    bdf680
Next     TCB point to address         0

the file 'TaskSet.txt'was opened
Task[ 1] created, TCB Address   bdf6dc
------After TCB[ 1] being linked------
Previous TCB point to address         0
Current  TCB point to address    bdf6dc
Next     TCB point to address    bdf680

Task[ 2] created, TCB Address   bdf738
------After TCB[ 2] being linked------
Previous TCB point to address         0
Current  TCB point to address    bdf738
Next     TCB point to address    bdf6dc

==================TCB linked list==================
Task    Prev_TCB_addr   TCB_addr   Next_TCB_addr
 2              0        bdf738         bdf6dc
 1         bdf738        bdf6dc         bdf680
63         bdf6dc        bdf680              0
```

[PART II] RM Scheduler Implementation

Objective:

To implement the Rate Monotonic (RM) scheduler for periodic tasks and observe the scheduling behaviors.

Problem Definition:

Implement the following three task sets of periodic tasks. Add necessary code to the μC/OS-II scheduler **in the kernel level** to observe how the task suffers from the scheduler. We give the files for the parameter of the task.

Periodic Task Set = {τ_{ID} (ID, arrival time, execution time, period)}

Example Task Set 1 = {τ_1 (1, 1, 2, 4), τ_2 (2, 0, 4, 10)}

Example Task Set 2 = {τ_1 (1, 3, 4, 14), τ_2 (2, 0, 2, 8), τ_3 (3, 0, 4, 10), τ_4(4, 24, 2, 12)}

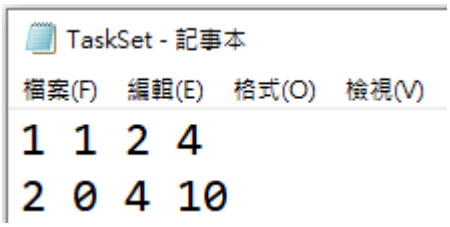
Example Task Set 3 = {τ_1 (1, 2, 2, 10), τ_2 (2, 1, 1, 5), τ_3 (3, 0, 8, 15)}

※ The priority of the task is set according to the RM scheduling rules.

The input file format:

Task ID	Arrive Time	Execution Time	Task Periodic
##	##	##	##

Example of task set file 1:



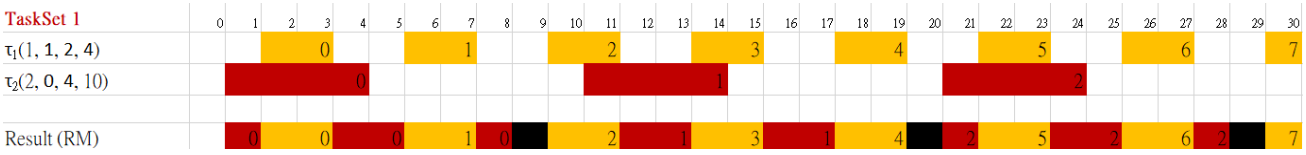
Evaluation:

The output format:

Tick	Event	CurrentTask ID	NextTask ID	Response Time	# of Context Switch	Preemption Time	OSTimeDly
##	**Preemption**	**task(ID)(job number)**	**task(ID)(job number)**				
##	**Completion**	**task(ID)(job number)**	**task(ID)(job number)**	##	##	##	##
##	**MissDeadline**	**task(ID)(job number)**	**----------**				

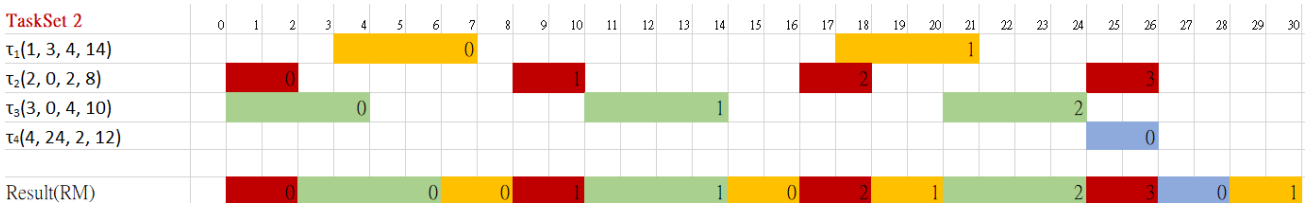
※ If the task is Idle Task, print "*task(priority)*".

The output results of Task Set 1:



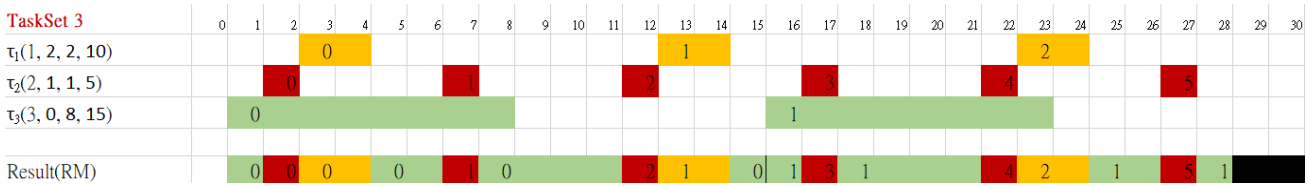
Tick	Event	CurrentTask ID	NextTask ID	ResponseTime	#of ContextSwitch	PreemptionTime	OSTimeDly
1	Preemption	task(2)(0)	task(1)(0)				
3	Completion	task(1)(0)	task(2)(0)	2	2	0	2
5	Preemption	task(2)(0)	task(1)(1)				
7	Completion	task(1)(1)	task(2)(0)	2	2	0	2
8	Completion	task(2)(0)	task(63)	8	5	4	2
9	Preemption	task(63)	task(1)(2)				
11	Completion	task(1)(2)	task(2)(1)	2	2	0	2
13	Preemption	task(2)(1)	task(1)(3)				
15	Completion	task(1)(3)	task(2)(1)	2	2	0	2
17	Completion	task(2)(1)	task(1)(4)	7	4	3	3
19	Completion	task(1)(4)	task(63)	2	2	0	2
20	Preemption	task(63)	task(2)(2)				
21	Preemption	task(2)(2)	task(1)(5)				
23	Completion	task(1)(5)	task(2)(2)	2	2	0	2
25	Preemption	task(2)(2)	task(1)(6)				
27	Completion	task(1)(6)	task(2)(2)	2	2	0	2
28	Completion	task(2)(2)	task(63)	8	6	4	2
29	Preemption	task(63)	task(1)(7)				

The output results of Task Set 2:



Tick	Event	CurrentTask ID	NextTask ID	ResponseTime	#of ContextSwitch	PreemptionTime	OSTimeDly
2	Completion	task(2)(0)	task(3)(0)	2	1	0	6
6	Completion	task(3)(0)	task(1)(0)	6	2	2	4
8	Preemption	task(1)(0)	task(2)(1)				
10	Completion	task(2)(1)	task(3)(1)	2	2	0	6
14	Completion	task(3)(1)	task(1)(0)	4	2	0	6
16	Completion	task(1)(0)	task(2)(2)	13	4	9	1
18	Completion	task(2)(2)	task(1)(1)	2	2	0	6
20	Preemption	task(1)(1)	task(3)(2)				
24	Completion	task(3)(2)	task(2)(3)	4	2	0	6
26	Completion	task(2)(3)	task(4)(0)	2	2	0	6
28	Completion	task(4)(0)	task(1)(1)	4	2	2	8
30	Completion	task(1)(1)	task(3)(3)	13	4	9	1

The output results of Task Set 3:



Tick	Event	CurrentTask ID	NextTask ID	ResponseTime	#of ContextSwitch	PreemptionTime	OSTimeDly
1	Preemption	task(3)(0)	task(2)(0)				
2	Completion	task(2)(0)	task(1)(0)	1	2	0	4
4	Completion	task(1)(0)	task(3)(0)	2	2	0	8
6	Preemption	task(3)(0)	task(2)(1)				
7	Completion	task(2)(1)	task(3)(0)	1	2	0	4
11	Preemption	task(3)(0)	task(2)(2)				
12	Completion	task(2)(2)	task(1)(1)	1	2	0	4
14	Completion	task(1)(1)	task(3)(0)	2	2	0	8
15	Completion	task(3)(0)	task(3)(1)	15	6	7	
16	Preemption	task(3)(1)	task(2)(3)				
17	Completion	task(2)(3)	task(3)(1)	1	2	0	4
21	Preemption	task(3)(1)	task(2)(4)				
22	Completion	task(2)(4)	task(1)(2)	1	2	0	4
24	Completion	task(1)(2)	task(3)(1)	2	2	0	8
26	Preemption	task(3)(1)	task(2)(5)				
27	Completion	task(2)(5)	task(3)(1)	1	2	0	4
28	Completion	task(3)(1)	task(63)	13	7	5	2
30	Preemption	task(63)	task(3)(2)				

[Bonus] FIFO Scheduler Implementation

Objective:

To implement the non-preemptive First In First Out (FIFO) scheduling for periodic tasks, and handle the miss deadline behaviors.

Problem Definition:

Implement the following task set of periodic tasks. Add necessary code to the μC/OS-II scheduler **in the kernel level** to observe how the task suffers the schedule delay.

Periodic Task Set = {τ_{ID} (ID, arrival time, execution time, period)}

Task set 1 = {τ_1 (1, 1, 2, 6), τ_2 (2, 0, 4, 10)}

※If tasks arrive simultaneously, the task with the longest execution time will be executed first.

Evaluation:

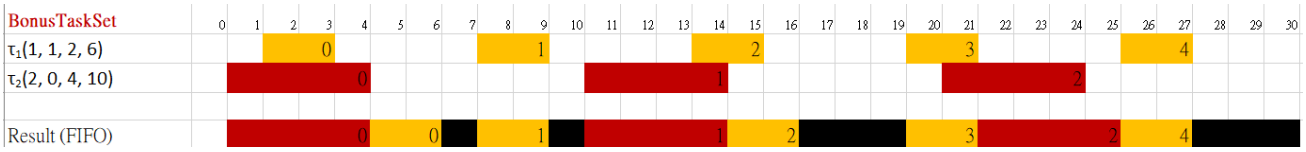
The output format:

Tick	Event	CurrentTask ID	NextTask ID	Response Time	# of Context Switch	Preemption Time	OSTimeDly
##	Preemption	task(ID)(job number)	task(ID)(job number)				
##	Completion	task(ID)(job number)	task(ID)(job number)	##	##	##	##
##	MissDeadline	task(ID)(job number)	----------				

※ If the task is Idle Task, print "*task(priority)*".

The output results of Task Set 1:

Consider two tasks **τ_1 (1, 1, 2, 6)** and **τ_2 (2, 0, 4, 10):**



Tick	Event	CurrentTask ID	NextTask ID	ResponseTime	#of ContextSwitch	PreemptionTime	OSTimeDly
4	Completion	task(2)(0)	task(1)(0)	4	1	0	6
6	Completion	task(1)(0)	task(63)	5	2	3	1
7	Preemption	task(63)	task(1)(1)				
9	Completion	task(1)(1)	task(63)	2	2	0	4
10	Preemption	task(63)	task(2)(1)				
14	Completion	task(2)(1)	task(1)(2)	4	2	0	6
16	Completion	task(1)(2)	task(63)	3	2	1	3
19	Preemption	task(63)	task(1)(3)				
21	Completion	task(1)(3)	task(2)(2)	2	2	0	4
25	Completion	task(2)(2)	task(1)(4)	5	2	1	5
27	Completion	task(1)(4)	task(63)	2	2	0	4
30	Preemption	task(63)	task(2)(3)				

Credit:

[PART I] Task Control Block Linked List [20%]

- The screenshot results. (10%)
- A report that describes your implementation (please attach the screenshot of the code and **MARK** the modified part). (10%)

[PART II] RM Scheduler Implementation [80%]

- The correctness of schedule results of examples. Note the testing task set might not be the same as the given example task set. (40%)
- A report that describes your implementation (please attach the screenshot of the code and **MARK** the modified part). (40%)

[Bonus I] FIFO Scheduler Implementation [10%]

- The correctness of schedule results of examples. Note the testing task set might not be the same as the given example task set. (5%)
- Implement FIFO and compare the schedule results with that of RM (please attach the screenshot of the code and **MARK** the modified part). (3%)
- Implement and describe how to handle the deadline missing situation under FIFO. (2%)

※ You must modify the source code!

※ Standard input and output filenames in the project are necessary for the checker. Please check the file names before submitting.

```
#define INPUT_FILE_NAME "./TaskSet.txt"
#define OUTPUT_FILE_NAME "./Output.txt"
```

※ Please set the system end time as 30 seconds in this project.

```
#define SYSTEM_END_TIME 30
```

※ We will use different task sets to verify your code.

※ When the current task is completed, the completion information shall be printed even if there is one task missing its deadline.

Project submit:

Submit to Moodle2.

Submit deadline: November 2nd, 2022 (Wednesday) at 12:00

File name format: RTOS_Myyyddxxx_PA1.zip

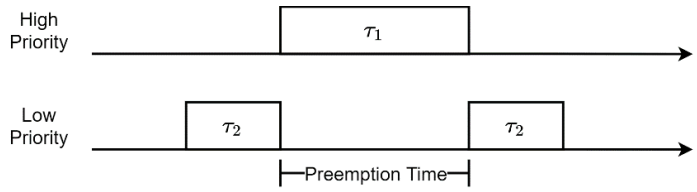
RTOS_Myyyddxxx_PA1.zip includes (The tree structure of files is shown as hints):

- The report (RTOS_Myyyddxxx_PA1.pdf).
- Folder with the executable µC/OS-II project (RTOS_Myyyddxxx_PA1).

※ Plagiarizing is strictly prohibited.

Hints:

1. Preemption time is introduced in multiple tasking.



2. RTOS_Myyyddxxx_PA1.zip include files as follows:

```
C:.
|   RTOS_Myyyddxxx_PA1.pdf
|
\---RTOS_Myyyddxxx_PA1
    |   ReadMe.txt
    |
    +---Micrium
    |   \---Software
    |       +---uC-CPU
    |       |   |   cpu_cache.h
    |       |   |   cpu_core.c
    |       |   |   cpu_core.h
    |       |   |   cpu_def.h
    |       |   |
    |       |   \---Win32
    |       |       \---Visual_Studio
    |       |               cpu.h
    |       |               cpu_c.c
    |       |
    |       +---uC-LIB
    |       |       lib_ascii.c
    |       |       lib_ascii.h
    |       |       lib_def.h
    |       |       lib_math.c
    |       |       lib_math.h
    |       |       lib_mem.c
    |       |       lib_mem.h
    |       |       lib_str.c
    |       |       lib_str.h
    |       |
    |       \---uCOS-II
    |           +---Ports
    |           |   \---Win32
    |           |       \---Visual Studio
    |           |               os_cpu.h
    |           |               os_cpu_c.c
    |           |
    |           \---Source
    |                   os.h
    |                   os_cfg_r.h
    |                   os_core.c
    |                   os_dbg_r.c
    |                   os_flag.c
    |                   os_mbox.c
    |                   os_mem.c
    |                   os_mutex.c
    |                   os_q.c
    |                   os_sem.c
    |                   os_task.c
    |                   os_time.c
    |                   os_tmr.c
    |                   os_trace.h
    |                   ucos_ii.c
    |                   ucos_ii.h
    |
```

```
    \---Microsoft
        +---BSP
        |   \---Windows
        |           bsp_cpu.c
        |
        \---Windows
            \---Kernel
                |   app_cfg.h
                |   cpu_cfg.h
                |   lib_cfg.h
                |
                \---OS2
                    |   app_hooks.c
                    |   main.c
                    |   os_cfg.h
                    |
                    \---VS
                            OS2.sln
                            OS2.vcxproj
                            OS2.vcxproj.filters
                            OS2.vcxproj.user
```