



Operating System Practice– Final Project

Che-Wei Chang

chewei@mail.cgu.edu.tw

Department of Computer Science and Information
Engineering, Chang Gung University

Report

- ▶ Only four A4 pages
- ▶ 12 pt words
- ▶ Deadline is 23:00 2026/06/8
- ▶ File name: OSP-Project-TeamNumber.zip
- ▶ Required Files: The source code files and the report
- ▶ In the report, remember to provide your names and student IDs
- ▶ Upload to the e-learning system



The Requirements of Final Presentation

- ▶ Demonstration is on 6/11, and 4 students are in a team
- ▶ Presentation is only for 5 minutes
 - Quickly go through the implementation
 - Talk more about the problems you solved
 - Highlight your extra exercise
- ▶ Live demo is required
 - Explain and quickly go through your program
 - Show your source code, TAs might ask questions on your source code
- ▶ You will be asked for one or two questions
 - You have **only 30 seconds** to answer a question



Grading Rules

- ▶ Report: 35% (normal upper bound 30)
- ▶ Presentation: 35% (normal upper bound 30)
- ▶ Question Answer: A-30 B-25 C-20 D-15 E-10 F-5



Requirements

▶ Task Scheduling

- Adopt priority-driven scheduling
- The scheduler always schedules the highest priority ready task to run
- Modify the priority of each task
- Related code in uC/OS II
 - See OS_Sched() for scheduling policy
 - See OSTimeTick() for time management
 - See OSIntExit() for the interrupt management

▶ Provide the RM and EDF Scheduler

- Input: A task set, each task is with its execution time and period
- Output: The printed result of each task



Input

- ▶ The input format should be as follows
 - Your program should have the capability to create the assigned number of tasks and their corresponding period and execution time.
 - Example: taskset.txt

```
3 //number of task
1 3 // task 1: (execution time 1, period 1)
2 9 // task 2: (execution time 2, period 2)
4 12 // task 3: (execution time 3, period 3)
```
- ▶ The total utilization is no more than **65%**
- ▶ The number of tasks is no more than **7**

Input Example (1 / 2)

4

1 12

1 7

2 19

3 20



Input Example (2/2)

5

1 18

1 17

2 16

1 20

1 6



Output

- ▶ Your program output must shows the following information
 - A sequence of the running task over time
 - The time when context switch occurred
- ▶ A report to describe your implementation
 - Relationship of each function
 - Implementation flow chart
 - Implementation details



Hints (1 / 2)

- ▶ You can read three other example in the document and refer to the source code.
- ▶ In order to implement a new scheduler, we might have to modify the `os_tcb` data structure to include some new attributes.
- ▶ The function `OSTaskCreateExt()` is used to create tasks, and we can modify this function to input the execution time and the period to each task.
- ▶ Each task executes an infinite loop and uses `OSTimeGet()` to get the execution time, where `OS_TICKS_PER_SEC` is the number of ticks for a second.
 - Note that a task might be preempted during its execution.
- ▶ Use `OSTimeDly()` when the task finish its execution.



Hints (2 / 2)

- ▶ Modify the deadline of a task before it call `OSTimeDly()` (ex: `OSTCBCur->deadline=OSTCBCur->deadline+TaskPeriod`)
- ▶ When the delay of a task is completed, the function `OSTaskResume()` is called to put the task back to ready queue and reschedule.
- ▶ Modify the function `OS_Sched()` to pick the task with the shortest period or the earliest deadline.
- ▶ `OSStart()` is used to start the execution of tasks.
- ▶ `OSTaskChangePrio()` is used to change the priority of a task.



Bonuses

- ▶ Implementation and discussion of PIP on RM: 20%
- or
- ▶ Implementation and discussion of PCP on RM: 30%

