CS 452 / 652 Real-Time Programming

Final Examination

Wednesday, December 11, 2002 19:00-22:00

(30 marks)

Question 1 (5 marks):

Memory management. The Buddy System combines dynamic and static memory partitioning. Given 1M of memory, draw the memory state for each of the following actions using the Buddy System scheme:

Process	Action
A	Request 100 K
B	Request 240 K
C	Request 60 K
D	Request 250 K
B	Release B
A	Release A
E	Request 75 K
C	Release C
E	Release E
D	Release D

Describe the advantages and disadvantages of the Buddy System in comparison to the system you used in your kernel. Include discussion on internal and external fragmentation, compaction and efficiency.

Question 2 (8 marks):

Priority inversion occurs when a higher priority task is forced to block and wait on a lower priority task. Outline a realistic scenario that illustrates the priority inversion problem. Make certain that the problem described in your scenario could not be solved simply by reassigning new static priorities to the tasks involved.

Priority inheritance, which allows the priority of tasks to be increased dynamically, is the standard solution to the priority inversion problem. Outline how your kernel could be modified to implement priority inheritance. Discuss the exact conditions under which a task's priority will be increased and the level to which it will be increased. Describe the specific changes to the kernel data structures and the changes to the algorithms for Send, Receive and Reply needed to support priority inheritance. How would these changes affect the maximum length of time spent in the kernel with interrupts off?

Question 3 (8 marks):

Add a process to your final application that re-issues the last command for each train (except reverse) every 2 seconds. Describe all of the software layers invoked to execute this process. Use diagrams and pseudocode as appropriate.

Question 4 (4 marks):

Rate Monotonic Priority Ordering (RMPO). A set of tasks will be executed on a uniprocessor system with pre-emption. For each task τ_i, C_i is the max time to execute the task, T_i is the max time allowed between executions (service period). Assuming all tasks start at time $t = 0$, graph the CPU load for the tasks from the following table using a rate monotonic priority ordering schedule. Label the critical instant. Is the the schedule feasible?

Task	T	C
τ_1	4	1
τ_2	6	2
τ_3	8	2
τ_4	12	1

Question 5 (5 marks):

Deadline Monotonic Priority Ordering (DMPO) Schedule. A set of tasks will be executed on a uniprocessor system with pre-emption. For each task τ_i, C_i is the max time to execute the task, T_i is the time between executions (service period) and D_i is the deadline for completion. Determine if the following task set is schedulable using the DMPO test function?

Task	T	C	D
τ_1	90	14	50
τ_2	320	50	140
τ_3	700	110	220