Thursday, May 11, 2006
2:30p.m. – 4.15p.m.

University of Glasgow

DEGREES OF M.Sc., P.G. Dip., M.Sci., M.Eng., B.Eng., B.Sc., M.A. and M.A. (Social Sciences)

COMPUTING SCIENCE M:
REAL TIME AND EMBEDDED SYSTEMS

(Answer 3 out of 4 questions.)

1. **(a)** Consider a system of four independent preemptable periodic tasks:

$T_1 = (8, 1)$, $T_2 = (4, 1)$, $T_3 = (12, 4)$, and $T_4 = (10, 1)$

All jobs have phase equal to zero, and relative deadline equal to their period. Can this system be scheduling using the least slack time algorithm? Explain your answer.

[2]

(b) Can the system from part (a) be scheduled using the deadline monotonic algorithm? Explain your answer.

[2]

(c) Can the system from part (a) be scheduled using the rate monotonic algorithm? How will the resulting schedule differ from that produced by the deadline monotonic algorithm? Explain your answer.

[2]

(d) It is not always possible to determine the schedulability of a fixed priority system using simple maximum schedulable utilisation tests. An alternative schedulability test can be used in some cases, based on the concept of a critical instant and timed demand analysis. Explain what are critical instants, and outline how they can be used with time demand analysis to provide a schedulability test for fixed priority systems. Explain why critical instants cannot be used as the basis for a schedulability test for systems using dynamic priority scheduling.

[8]

(e) A system of 15 independent preemptable periodic tasks is scheduled using the rate monotonic algorithm, but must run on an operating system that supports only eight priority levels. Explain how the limited number of available operating system priority levels might affect the performance of the tasks. Discuss how you would best map the rate monotonic priorities onto the available priority levels.

[6]

2. One approach to scheduling aperiodic or sporadic jobs on a priority-scheduled system is to run them within some form of periodic server task. Such servers are scheduled along with other periodic tasks in the system, with priority determined by the scheduling algorithm in use, but include the concept of a server execution budget, e_{PS}, along with their period, p_{PS}, and a set of rules to specify how the budget is consumed and replenished, and how the server executes when it has budget and available work.

(a) Two types of periodic server are the polling server and the deferrable server. For each of these, explain the budget consumption and replenishment rules.

[6]

(b) A third type of periodic server is the sporadic server. The budget consumption and replenishment rules for a sporadic server are more complex than for the polling or deferrable servers. Why, given their complexity, are sporadic servers used?

[2]

(c) POSIX 1003.1d defines a `SCHED_SPORADIC` scheduling policy, which provides a hybrid sporadic/background server. This acts as a standard sporadic server while it has budget, but runs as a low priority background server when the sporadic server budget is exhausted. How might this difference in behaviour affect the performance of tasks running within the server, compared to running those tasks within a standard sporadic server? How might a hybrid server such as this affect the schedulability of the system?

[4]

(d) You have been asked to design the scheduler for a new real-time operating system. This system is to support periodic tasks (which may block on resource access) using the earliest deadline first algorithm, and also sporadic tasks. With the aid of a diagram, outline the design of the scheduler, showing the various queues and any other important features of the design, and explaining how jobs are ordered in the various queues. Discuss the algorithmic complexity of your design compared to a naïve design using a single queue for all jobs, ordered by absolute deadline.

[8]

3. **(a)** Explain what is priority inversion? When does it occur?

[2]

(b) Two options for avoiding priority inversion are the basic priority inheritance protocol and the basic priority ceiling protocol. Discuss how these two protocols differ in terms of their scheduling, resource allocation and priority inheritance rules.

[6]

(c) Explain why the basic priority ceiling protocol is immune to deadlock.

[2]

(d) You have been asked to schedule a system of fixed priority tasks that contend for resources. Discuss the circumstances under which you would use each of the basic priority inheritance protocol, the basic priority ceiling protocol, or the stack-based priority ceiling protocol to control resource access, and explain the advantages of each for those circumstances.

[8]

(e) Can the basic priority ceiling protocol be used with dynamic priority scheduling? Would it be a good idea to do so? Explain your answer.

[2]

4. **(a)** The thesis of Edward A. Lee's paper "Absolutely Positively on Time: What Would it Take?" (IEEE Computer, July 2005, pages 85-87) is that "recent computing advances do more harm than good when embedded computing systems absolutely must meet tight timing constraints", and that we must "reinvent computing science" in order that real-time and embedded systems can realise their full potential, and move beyond the confines of C and POSIX. Do you agree with this thesis? Discuss how recent advances in hardware, in operating systems and in programming languages have affected the design of real-time systems, and whether these changes are beneficial or harmful.

[20]