University of Glasgow

DEGREES OF M.Eng., B.Eng., B.Sc., M.A. and M.A. (Social Sciences)

COMPUTING SCIENCE 4H:
REAL TIME AND EMBEDDED SYSTEMS
Sample Examination

(Answer 3 out of 4 questions.)

1. Assume that you have a system of periodic, independent, preemptible tasks to be scheduled on a single processor,

$\mathcal{T} = \{T_i\}$ for i = 1..n, where $T_i = (\varphi_i, p_i, e_i, \text{and } D_i)$ for each i.

(a) Assume $p_i = D_i$ for each i. What is the schedulability condition of the Earliest Deadline First algorithm for this system? Is this a necessary and sufficient condition?

[3]

(b) Assume that $p_i \geq D_i$ for each i. What is the schedulability condition of the Earliest Deadline First algorithm for this system? Is this a necessary and sufficient condition?

[3]

(c) Assume that $p_i = D_i$ for each i. What is the schedulability condition of the Rate Monotonic algorithm for this system? Is this a necessary and sufficient condition?

[3]

(d) Assume that $p_i \leq D_i$ for each i. Under what conditions will the schedulability of the Rate Monotonic algorithm for this system be identical to that stated in response to part (a) above?

[3]

(e) You have been provided with the following system definition (all tasks are independent and preemptible, and to be scheduled on a single processor):

$\mathcal{T} = \{T_1 = (0, 2, 0.4, 2), T_2 = (1, 4, 1, 4), T_3 = (0, 5, 1.5, 5)\}$

(i) Is $\mathcal{T}$ schedulable using EDF? Explain your answer.

(ii) Is $\mathcal{T}$ schedulable using RM? Explain your answer.

The parameters of the system have changed, such that T_3 becomes (0, 8, 4, 8).

(iii) Is $\mathcal{T}$ schedulable using EDF? Explain your answer.

(iv) Is $\mathcal{T}$ schedulable using RM? Explain your answer.

[2+2+2+2]

2. You have been hired into a software engineering firm to replace a design engineer that has recently left. His legacy is the design of a real time embedded system. Your first task upon arrival is to critically review the design, since the implementation phase is to start within two weeks.

(a) The system consists primarily of independent, preemptible, periodic tasks that must meet their deadlines, along with random aperiodic jobs involved in the user interface of the system; it is desirable to minimize the average response times of the aperiodic jobs. Your predecessor had decided to use a simple deferrable server. The total utilization of the periodic jobs is U_p, and the maximum utilization permitted while still being able to meet all deadlines is $U_{max} > U_p$. Your predecessor has specified that the deferrable server size should be $U_{max} - U_p$.

Do you agree with his choice? If so, indicate why; if not, how would you change the design?

[5]

(b) Several environmental considerations lead to using an earliest deadline first scheduling algorithm for the periodic tasks and the server. Several of the periodic tasks compete for exclusive access to a shared resource, and it is essential that the system not deadlock. Your predecessor designed the system to use the priority inheritance protocol to minimize blocking due to resource contention.

Do you agree with his choice? If so, indicate why; if not, how would you change the design?

[5]

(c) We usually assume that the context switch time is negligible when determining the schedulability of a system. Your predecessor has instrumented a prototype of the running system, and has determined the maximum context switch time for jobs in execution to be T_{CS}. He has, therefore, increased the execution time for each of the periodic tasks (including the bandwidth-preserving server) by $2 * T_{CS}$.

Do you agree with this approach? If so, indicate why; if not, how would you approach this problem differently?

[5]

(d) Each job in periodic task T_i queries an array of sensors; each query to each sensor takes ~1ms to complete, and there are N sensors to be queried serially; after issuing the query, the job self-suspends until the I/O completes. Your predecessor has accounted for this situation by increasing the execution time e_i for jobs in the task by $N * T_{CS}$.

Do you agree with this approach? If so, indicate why; if not, how would you approach this problem differently?

[5]

3. **(a)** Many real-time systems comprise tasks which have to be executed periodically. An operating system can support such tasks through the abstraction of a "periodic thread", or they can be implemented using a standard thread that loops with an appropriate period. Compare the two approaches, commenting on their relative advantages and disadvantages.

[3]

(b) The POSIX 1003.1b API provides the ability to schedule processes at a range of priority levels, according to either FIFO or round-robin scheduling disciplines. Using a diagram of the scheduler's priority queues, explain how both real-time and non-real time tasks may be supported by a single operating system. Your diagram should show tasks in the ready, executing, sleeping and blocked states, and you should explain when transitions between states occur.

[8]

(c) Explain how the presence of non-real time tasks might disrupt the operation of real-time tasks running on the same system.

[4]

(d) You are debugging a soft real time networked multimedia application running on Linux. The application uses the POSIX real time extensions to select high priority FIFO scheduling, so that it has priority over non-real time tasks in the system and can achieve smooth video playback. Unfortunately, the application is faulty and goes into an infinite loop when trying to decode certain video sequences, never blocking or yielding the processor. What would be the effects of this on other tasks running on the system? How would you debug the program?

[5]

4. **(a)** You have been given the task of implementing a streaming audio application, as part of a media server to be used by an Internet radio station. You can implement this application using TCP/IP or UDP/IP. Which would you choose, and why?

[4]

(b) When testing the streaming audio application, which sends packets with constant spacing, you notice that the inter-packet timing is disrupted on reception. This is causing problems for your application, which tries to playout the audio as soon as each packet is received. Assuming you cannot change the network, how would you adapt the application to make it robust to this timing variation?

[2]

(c) Explain why the inter-packet spacing at the receiver may have been disrupted, describing the effects of each type of problem. Is disruption of inter-packet timing a problem with the sender, the network, or both?

[8]

(d) Flush with success from developing the streaming audio application, you take another job with a bank that wishes to stream real-time stock quotes to their customers. A key requirement of this new application is that all customers receive the quotes at approximately the same time, to prevent one customer getting an unfair advantage over others. Your employer is concerned that it may not be possible to build such a system using an IP network. Is he right to be concerned? Justify your answer.

[6]