Real-Time Operating Systems and Languages

Real-Time and Embedded Systems (M)

Tutorial 4

UNIVERSITY
of
GLASGOW



Tutorial Outline

- Review of problem set 3
- Review of lectures
- Discussion and examples
- Question & answer

Copyright © 2006 University of Glasgow

Review of Problem Set 3: Question 1

- Question 1: Consider a system of three periodic tasks $T_1 = (4, 1)$, $T_2 = (5, 1)$ and $T_3 = (10, 2)$. Demonstrate that this system can be scheduled using:

  a) the rate monotonic algorithm

  b) the EDF algorithm

Copyright © 2006 University of Glasgow

Review of Problem Set 3: Question 2

- Question 2: The system from question 1 must also support the execution of three aperiodic jobs: A_1 which is released at time 1, A_2 which is released at time 8, and A_3 which is released at time 12. Each aperiodic job executes for 1 unit of time. The system is scheduled using RM, with a server task, $T_s = (3, 0.5)$ to schedule the aperiodic jobs. Is the system schedulable if T_s is a polling server? What are the response times of the aperiodic jobs?

Copyright © 2006 University of Glasgow

Review of Problem Set 3: Question 3

- Question 3: Is the system from question 2 schedulable if T_s is a deferrable server? What are the response times of the aperiodic jobs?

Copyright © 2006 University of Glasgow

Review of Problem Set 3: Question 4

- Question 4: Is the system from question 2 schedulable if T_s is a simple sporadic server? Explain your answer.

Copyright © 2006 University of Glasgow

Review of Problem Set 3: Question 5

- Question 5: A simple sporadic server can be defined for EDF systems, as well as for rate monotonic systems. Explain the difference between the consumption and replenishment rules for a simple sporadic server in a deadline driven system, compared to a rate monotonic system.

Copyright © 2006 University of Glasgow

Review of Lectures

- Implementing real time systems
  - Key concepts and constraints
  - System architectures:
    - Cyclic executive
    - Microkernel with priority scheduler
- Implementing scheduling algorithms
  - Jobs, tasks, and threads
  - Priority scheduling of periodic tasks
    - Rate monotonic
    - Earliest deadline first
  - Priority scheduling of aperiodic and sporadic tasks

Copyright © 2006 University of Glasgow

Review of Lectures

- Real-time operating systems and languages
  - Clocks and timing
    - Clocks and the concept of time
    - Delays and timeouts
  - Scheduling
  - C and POSIX, Real-time Java and Ada
- Real-time system as part of a larger system
  - Two level scheduler
    - Discussion of concepts
    - Advantages and disadvantages
    - Case study: RTLinux
- Real-time on general purpose systems
  - Need for flexible applications
  - Implementation strategies
  - Scheduling

Copyright © 2006 University of Glasgow

Key Learning Outcomes

- Understanding how real-time operating system schedulers are implemented; different implementation styles and their trade-off
- Understanding of how to use real-time operating systems; their features and the limits of practical scheduling
- Understanding of why flexible applications are needed; and some of the issues to consider when designing such applications

Copyright © 2006 University of Glasgow

Discussion and Examples (1)

- 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 advantages and disadvantages of the two approaches.

Copyright © 2006 University of Glasgow

Discussion and Examples (2)

- The POSIX 1003.1b API provides the ability to schedule processes at a range of fixed 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

Copyright © 2006 University of Glasgow

Discussion and Examples (3)

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

Copyright © 2006 University of Glasgow

Discussion and Examples (4)

- 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?

Copyright © 2006 University of Glasgow

Any Further Questions?

Copyright © 2006 University of Glasgow