# Resource Access Control

Real-Time and Embedded Systems (M)

Tutorial 5

UNIVERSITY
*of*
GLASGOW



# Tutorial Outline

- Review of lectures
- Worked examples
- Question & answer

Copyright © 2006 University of Glasgow
All rights reserved

# Review of Lectures

- Definition of resources
- Resources access control protocols
  - Non-preemptable critical sections
  - Basic priority inheritance protocol
  - Priority ceiling protocol
    - Basic priority ceiling protocol
    - Stack-based priority ceiling protocol
    - Ceiling priority protocol
  - Resource access control for dynamic priority systems
- Implementing resource access control
  - Locking primitives
    - Semaphores and mutexes; priority inheritance using POSIX mutexes/ scheduling
  - Messages, signals and events
    - Priority inheritance for messaging

Copyright © 2006 University of Glasgow
All rights reserved

# Key Learning Outcomes

- Defined resources, explaining timing anomalies and the need for resource access control
- Illustrated operation of several resource access control protocols:
  - Non-preemptable critical section
  - Basic priority inheritance protocol
  - Basic and stack-based priority ceiling protocols
- Described some practical methods used to implement resource access control:
  - Use of POSIX real-time extensions and mutexes for locking, to directly implement the ideas described
  - Other mechanisms: semaphores, message queues, signals, etc.

Copyright © 2006 University of Glasgow
All rights reserved

# Example: Priority Ceiling Protocol

- Consider a system of 3 jobs:
  - $J_1$ released at 3.6, executes for 3.8, and has highest priority
  - $J_2$ released at 1.0, executes for 4.0
  - $J_3$ released at 0.0, executes for 5.6, and has lowest priority
- Resources:
  - $J_1$ requires resource red for 1.6 units after it has executed for 1.0 units
  - $J_2$ requires resource green for 2.0 units after it has executed for 2.0 units; 0.6 units after getting green it also requires blue for 0.8 units
  - $J_3$ requires resource blue for 2.0 units, after it has executed for 0.6 units; 1.0 units after getting blue it also requires green for 1.0 units
- Draw a diagram to show how the system is scheduled if the *priority ceiling protocol* is used to arbitrate resource access

Copyright © 2006 University of Glasgow
All rights reserved

# Example: Priority Inheritance Protocol

- Consider a system of 3 jobs:
  - $J_1$ released at 3.6, executes for 3.8, and has highest priority
  - $J_2$ released at 1.0, executes for 4.0
  - $J_3$ released at 0.0, executes for 5.6, and has lowest priority
- Resources:
  - $J_1$ requires resource red for 1.6 units after it has executed for 1.0 units
  - $J_2$ requires resource green for 2.0 units after it has executed for 2.0 units; 0.6 units after getting green it also requires blue for 0.8 units
  - $J_3$ requires resource blue for 2.0 units, after it has executed for 0.6 units; 1.0 units after getting blue it also requires green for 1.0 units
- Draw a diagram to show how the system is scheduled if the *priority inheritance protocol* is used to arbitrate resource access

Copyright © 2006 University of Glasgow
All rights reserved

# Choice of Access Control Protocol

- The choice of resource access control protocol impacts not just the blocking time, but also correctness of the system
  - The priority inheritance protocol does not prevent deadlock, although the priority ceiling protocols do
  - Must use an additional protocol – e.g. enforce lock ordering – to ensure correctness, if priority inheritance used
    - E.g. if a job needs both green and blue resources, it *must* lock green before it tries to lock blue, even if this increases the length of the critical section

Copyright © 2006 University of Glasgow
All rights reserved

# Any Further Questions?

Copyright © 2006 University of Glasgow
All rights reserved