CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Real-Time Systems

Lecture #15

Professor Jan Jonsson

Department of Computer Science and Engineering
Chalmers University of Technology

Real-Time Systems

Facing the written exam

Monday, March 18, 2019 @ 08:30−12:30
in the M building

Note: in case you need to take a re-exam in August 2019, please note that it takes place at Lindholmen campus

Facing the exam

Permitted to use during the exam:

- Chalmers-approved calculator
  - Useful tool while solving feasibility analysis problems
  - Approved models: Casio FX-82, Sharp EL-W531, Texas TI-30
- Compendium: ”Programming with the TinyTimber kernel”
  - Important aid describing the basic principles of TinyTimber
  - Useful if you need to explain how to construct, for example, software with periodic activities
- Please observe:
  - Markings, indexes or notes are <u>not</u> permitted in the compendium.
  - Electronic dictionaries may <u>not</u> be used during the exam.

Facing the exam

Reading guidelines:

- Lecture and Exercise notes (”PowerPoint hand-outs”)
  - All material is <u>very relevant</u>, and <u>may be examined</u>
- Excerpts from research articles and books:
  - Recommended reading, but <u>will not be examined</u>
- Exercise compendium
  - Recommended problem solving …
- Old written exams
  - For inspiration …

Facing the exam

Important knowledge areas:

- Design principles for real-time systems
  - Real-time systems: typical properties, misconceptions
  - Real-time constraints: origin, interpretation (soft/hard)
  - Design phases: specification, implementation, verification
  - Verification: methods, difficulties, pitfalls
  - Network communication: methods (CAN in particular)

Facing the exam

Important knowledge areas (cont'd):

- Principles of concurrent programming
  - Paradigm: reactive (event driven) programming
  - Parallelization: pros & cons
  - Resource management: mutual exclusion, critical region
  - Deadlock: definition, management
  - Starvation: definition, management

Facing the exam

Important knowledge areas (cont'd):

- Language support for concurrent programming
  - Mutual exclusion: protected objects, monitors, semaphores, synchronized methods, mutex'd methods
  - Machine-level mutual exclusion: disable interrupt, test-and-set
- Language support for real-time programming
  - Units of concurrency: task, thread, method
  - Scheduling support: clocks, time, delays, priorities
  - Device drivers: interrupts handlers, call-back functionality
  - TinyTimber: "how it's done"
  - WCET: purpose, required properties, analysis methods

Facing the exam

Important knowledge areas (cont'd):

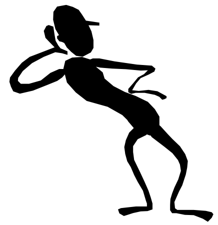
- Scheduling theory
  - Task model: WCET, deadline, period, offset
  - Scheduling: definitions, priorities, preemption
  - Feasibility tests: purpose, exactness (sufficient/necessary)
  - Complexity theory: time complexity, NP-completeness
- Scheduling with cyclic executives
  - Properties: time table, pros & cons
  - Scheduling: generation of time tables, run-time behavior
  - Feasibility test: hyper-period analysis

Facing the exam

Important knowledge areas (cont'd):

- Scheduling with pseudo-parallel execution
  - Properties: priority assignment, optimality, pros & cons
  - Scheduling: run-time behavior, construct timing diagram
  - Feasibility test: theory, assumptions, exactness, complexity
  - RM, DM, EDF: "how it's done"
  - RMFF, RM-US: "how it's done"

Facing the exam



What type of exam problems will there be?

- Real-time computing concepts
  - Will probe your general knowledge in real-time computing
- Programming concepts
  - Will probe your knowledge in how to design concurrent real-time programs (limited amounts of code programming)
- Scheduling concepts and theory
  - Will probe your knowledge in WCET analysis, scheduling, feasibility analysis and complexity theory

Let yourself be inspired, but not controlled, by the contents of old exams!

Facing the exam

Deriving the grade in course element 'Laboratory':

The grade (U, 3, 4, 5) will reflect your practical skills at the laboratory sessions as well as your presentation skills.

The grade is determined by the following:

- The quality of laboratory performance
  - sub-score is awarded based on a set of four criteria
  - sub-score sets a preliminary grade
- The quality of project report
  - sub-score is awarded based on a set of three criteria
  - sub-score can potentially adjust the grade

Facing the exam

Deriving the final grade in the course:

The final grade (U, 3, 4, 5) in the course will reflect your laboratory skills as well as your theoretical skills.

The final grade is influenced almost equally by:

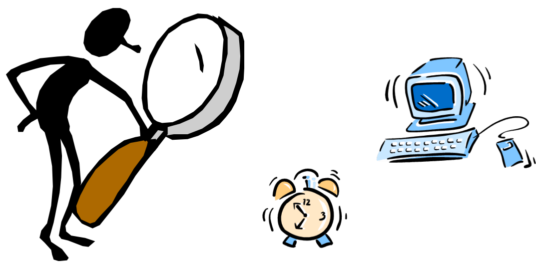
- Your results in course element ‘Laboratory’
  - based on a ‘Pass’ grade (3, 4, 5)
- Your results in course element ‘Examination’
  - based on a ‘Pass’ written exam score (24−60 points)

Note: GU students use Chalmers grading scale within Canvas, but get corresponding GU grades in Ladok.

Real-time systems

… and then ...

… what to do if you are curious and want to know more?



Real-time systems

What additional issues are there?

- How to generate schedules using search algorithms?
  - Branch-and-bound and simulated annealing algorithms
- How to achieve efficient multiprocessor scheduling?
  - P-fair scheduling and task-splitting algorithms
- How to handle system overload?
  - Which tasks to accept and which tasks to reject
- How to handle aperiodic tasks?
  - Server-based and server-less approaches

These issues (and more) are addressed in the advanced course in ”Parallel and Distributed Real-Time Systems” (EDA422/DIT172, quarter 4)