HALMSTAD UNIVERSITY

Center for Research on Embedded Systems (CERES)

Embedded Systems Programming

Assessment Guide 2015

In this document, the type of questions that may be posed in the final examination of listed. In order to learn about these subjects, you need to study the handouts, extra information posted on the page (including the proof of Liu and Layland's theorem) and also go through your solutions to the practicals. Reading the recommended textbook is not necessary.

Knowledge:

Know and compare the following concepts:

- Memory-mapped IO vs. separate bus IO,
- Shadowing and shadow variable,
- Busy waiting vs. interrupt-driven IO (and time-slicing),
- Manual vs. automatic interleaving,
- Static vs. dynamic priorities,
- Rate Monotonic vs. Earliest Deadline First scheduling,
- Periodic task scheduling vs. polling servers
- Deadline monotonic vs. Rate Monotonic
- Activity vs. notification
- Service vs. worker thread

Application

Be able to implement or analyze a piece of code using the following programming techniques:

- Bitwise operations
- Synchronization using mutex
- Spawning, yielding and dispatching threads using setjmp and longjmp
- Periodic tasks and deadlines using send,
- Multithreading in Java using Runnable,
- Interaction with UI in worker threads,
- Starting services from activities,
- Communicating using sockets, and
- Notifications and linking them to activities.

Use Rate Monotonic and Earliest Deadline First scheduling to determine the execution of a set of periodic tasks.

Determine the schedulability of a set of tasks using utilization-based schedulability analysis for Rate Monotonic, Earliest Deadline First, Deadline Monotonic and Rate Monotonic with Polling Servers.

Be able to understand and explained different parts of the proofs for the utlilization-based schedulability analysis of Rate Monotonic (for 2 tasks).

Analysis

Know and explain the challenges regarding the following models of embedded systems programming, possibly by means of an example:

- Busy waiting
- Interrupt-Driven IO
- Interleaving by Hand
- Concurrency
- Android Programming