Introduction to Real-time Systems

Syllabus

Web Page

http://www.ece.northwestern.edu/~dickrp/ece397-1

Instructors

Robert Dick
Department of Electrical and Computer Engineering
Technological Institute, Room Tech. L477
847-467-2298
dickrp@ece.northwestern.edu

Peter Dinda
Department of Computer Science
1890 Maple Avenue, Room 338
847-467-7859
Office Hours: Thursdays 2-4pm or by appointment
pdinda@cs.northwestern.edu

Teaching assistants

Ashish Gupta
Department of Computer Science
1890 Maple Avenue, Room 246
847-491-7060
ashish@cs.northwestern.edu
Office hours: Wed, Fri 11:30-1 or by appointment

Location and Time

Lecture: Tech. LR5, Tu, Th 12:30-2:00PM
Recitation: CS small classroom, 342, Tuesdays, 5:30-6:30pm.

Prerequisites

Required	CS 343 or equivalent operating systems course
Highly recommended	Familiarity with C/C++ systems programming
Recommended	Familiarity with microcontroller-based design

Textbook and other readings

Jane W. S. Liu, *Real-time Systems*, Prentice Hall, 2000 (Textbook)

- An in-depth introduction to real-time systems

Documentation on development board and tools

Survey papers and other handouts

Objectives, framework, philosophy, and caveats

Real-time systems are software/hardware systems in which timing constraints must be met for correctness. "Real-time" does not mean "really fast", but rather "get this done by the **deadline** or else you have failed." Such **hard real-time** systems are ubquitous and vital in a range of places from everyday life to estoteria:

- The ignition, fuel injection, and timing of most car engines is handled by a real-time system.
- The antilock brake system in most cars is a real-time system.
- The "avionics" in most commercial aircraft are real-time systems.
- The F117 stealth fighter and other modern fighter aircraft are designed to be aerodyamically unstable. They will fall out of the sky like a rock if the real-time control systems that stabilize them fail.
- The software of the Mars Rovers is architected around VxWorks, a commercial real-time operating system.

Most hard real-time systems are **embedded systems**, meaning the software is **codesigned** with the hardware, under often severe cost constraints, to accomplish a specific task. The user doesn't see the system as a general purpose computer.

Soft real-time systems are penalized by how much they miss the desired deadlines and by the variance of their latency. Such systems show up in many familiar places in general purpose computers, or should:

- Media playback
- Scientific visualization
- Games
- Interactive applications

Distributed real-time systems are those which have multple resources with non-negligible latency among them. Examples include the multiple processors in a car, or media being streamed over a network.

This course introduces the underlying concepts hard, soft, and distributed real-time systems.

You will learn:

- How to formally specify tasks and their constraints and dependencies
- How to use rate monotonic analysis (RMA), to tell you whether a set of tasks is schedulable
- How to program an embedded system with your tasks and executive.

- How to understand and use tools that make building an embedded hard real-time system easier.
- How general purpose real-time scheduling algorithms such as earliest deadline first (EDF) can schedule general sets of tasks, both period and aperiodic to meet their constraints.
- How we can combine bounded interrupt times, EDF, full preemption, priority inheritance, and admission control to achieve hard real-time performance in a general purpose operating system.
- How wired and wireless networks can provide hard and soft real-time guarantees, although very few do.
- How distributed real-time systems are architected and built on top of OS-level and network-level real-time primitives.
- How adaptive distributed real-time systems can provide some degree of real-time behavior even when such primitives are unavailable.

Our textbook provides an in-depth academic treatment of these topics and others. To make real-time systems concrete, however, we will couple the book and lecture with a series of labs in which you will construct several real-time systems on different hardware.

Be warned that this is the first iteration of this course. It covers the whole range of the real-time systems field, but it is certainly biased to some extent by your instructors' research interests in this area.

Projects

The projects in the course will use three different hardware platforms:

- Sensor platforms. Currently, we plan to use Crossbow Motes, tiny battery-powered embedded systems capable of wireless communication as well as sensing sound, light, and temperature.
- Handheld devices. Currently, we plan to use pocket PCs equipped with video cameras running the Windows CE real-time operating system.

The projects are geared to building a hierarchical distributed sensor network using these devices. Your sensor platforms will record light, sound, and temperature at their locations. Periodically, they will communicate to inform each other of the state of their corner of the world. Your handhelds will be able to tap into these communications and get the collective picture created by a group of sensor platforms. The handhelds will in turn talk to each other to form an even wider view of the world, sharing their sensor platforms' data and contributing their own.

Homework

Occasional, to reinforce the concepts in lecture.

Exams

There will be a midterm exam and non-cumulative final exam.

Grading

50 % Projects
10 % Homework
20 % Midterm
20 % Final

Late Policy

For each calendar day after the due date for a homework or a lab, 10% is lost. After 1 day, the maximum score is 90%, after 2 days, 80%, etc, for a maximum of 10 days. Homeworks that are four or more days late receive no credit.

Schedule

Num	Date	Topics	Recitation	Homework and Project	Readings
1	4 Jan	Introduction: Administrative details, applications for real-time systems, hard vs. soft real-time, distributed, communications (PD)	Intro to windows development environment (EVC++, PPC)	Introductory application for WinCE.	Liu, Chapter 1, Therac 25
2	6 Jan	Introduction (cont.): Defining costs, deadline violation cost functions, optimization formulation, real-time relationship to hardware, scheduling, operating systems (RD)			Liu, Chapter 2
3	11 Jan	Definitions: Graph representations, timing constraints, data dependencies, multirate scheduling, guarantee (hard vs. soft) (RD)	Intro to svn, mote development environment Introduction to nesc.	Svn repository for groups. Communication, local processing, transmission via RS232, print out on workstation.	Liu Chapter 3
4	13 Jan	Workload characterization,			Liu Chapter 3 and

		periodic, aperiodic, sporadic, deterministic, stochastic (PD)			handout
5	18 Jan	Scheduling: Off-line vs. on-line, independent vs. data dependent tasks, static, round-robbin, priority-driven, release times, theoretical results (RD)	Introduce Bluetooth, message libraries, audio processing concepts.	Walkie talkie with message abstraction.	Liu Chapter 4, Kwok survey paper
6	20 Jan	Off-line scheduling algorithms: list-based, force-directed, criticality-based, clock-driven (RD)			Liu Chapter 5
7	25 Jan	On-line scheduling issues: priority-driven, admission control, priority inversion, task dependencies (PD)	Introduction to Zigbee, Motes message manipulation . Changing message size.	Multi-hop streaming of audio with PPC-side playback.	Liu Chapter 6, Liu RMA paper
8	27 Jan	On-line scheduling admission control algorithms: RMA and others; scheduling algorithms: RMS, EDF, priority inheritance, and others (PD)			Liu Chapter 7, part of Chapter 8
9	1 Feb	RTOS definitions, comparison with general-purpose OSs, scheduling algorithms (RD)	Introduction to video processing.	React to multiple audio/video feed commands on PPCs.	Liu parts of Chapter 8, Ramamritha m paper
10	3 Feb	Bounded response time for device drivers, RTOS scheduling, case studies, light-weight vs. heavy-weight RTOSs (RD)			Ghosh survey paper, Liu Chapter 12
Midterm Exam: 7 Feb 7:00 PM (review session 4 Feb)					
11	8 Feb	Real-time networking	Reactive	React to	Liu Chapter

		(PD).	processing in TinyOS..	sense data frequency commands and alarm commands on motes.	11 and Tenet paper
12	10 Feb	Media processing, buffering, stream processing (PD)			Kurose and Ross
13	15 Feb	Routing, ad-hoc multi-hop networks (RD)	Help session.	Stress test, priority/mission control under load for PPC and motes.	Handouts
14	17 Feb	Distributed real-time systems (RD)			Dissertation introduction
15	22 Feb	Distributed real-time systems (PD)	Microcontroller hardware timers.	Mote data prioritization, rate adjustment.	Handouts
16	24 Feb	Queuing for real-time (PD).			Handouts
17	1 March	Hardware and real-time: power consumption, synthesis, prediction-hostile architectural features (RD).	Help session.	Queuing lab on PPC.	Handouts
18	3 March	Special topics (RD)			Handouts
19	7 March	Real-time for adaptive systems (PD).	Review session	Priority-based forwarding in ad-hoc network or power lab.	Handouts
20	9 March	Summary and review or special topics			
Final Exam: Monday, March 14, 7-9pm, Location TBD (Review Session: March 10)					