

**Course Syllabus**  
**COMP 737 [232] – Real-Time Systems**  
Spring 2008

**Meeting Place:** SN115

**Meeting Time:** 2:00 – 3:15, TuTh

**Course Web Page:** <http://www.cs.unc.edu/~anderson/teach/comp737>

**Powerpoint Transparencies:** In the public folder comp737 on my PC, Anderson1-cs

**Instructor:** Prof. Jim Anderson

**Telephone:** 962-1757

**Office:** SN362

**E-mail:** anderson@cs.unc.edu

**Office Hours:** If the class stays small, it will be by appointment: just send me email or stop by.

**Goal of the Course:** To study issues related to the design and analysis of systems with real-time constraints. The problem of ensuring such constraints is ultimately a scheduling problem, so much attention is devoted to such problems. This is a “must” course for anyone wanting to do real-time-systems research in this department.

**Primary Text:** *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, Second Edition, Giorgio Buttazzo, Springer, 2004.

**Supplemental Text:** *Real-Time Systems*, Jane Liu, Prentice Hall, 2000.

I recommend that you buy the supplemental text only if you think you will end up doing research in this area. We will also use some papers from the literature and a few chapters from other books. These will be copied for you.

**Prerequisites:** COMP 530 [142] (Undergrad Operating Systems). In addition, one of the things researchers in this area need to be comfortable with is computational-complexity issues pertaining to validating timing constraints. Because of this, for a class or two, some basic knowledge of NP-completeness, as covered in COMP 750 and some undergrad classes, will be useful. That said, a deep understanding of NP-completeness is not necessary. *No NP-completeness-related questions will be asked on exams* (though some easy ones may come up on homeworks). **Please note that I do not intend to zealously enforce the prerequisites.** Anyone with a decent background in algorithms and operating systems should be able to handle the material.

<b>Grading:</b> Homework	20%
Project	30%
Midterm Exam	20%
Final Exam	30%

We will probably have five or six homework assignments. Some of these will involve programming. *All homework assignments must be completed individually.* These assignments are designed to make sure you’re keeping pace: you should not find them extraordinarily time-consuming.

Each student must complete a class project. You are responsible for defining your own project. Your project can be either an experimental investigation or a survey or research paper. The project must be a fairly significant piece of work. Survey papers are viewed as an option of last resort — it will be more difficult to obtain top marks on the project if this is the option you choose. It is perfectly fine to use research from an RA position as the basis for your class project. However, your project may not be based on work from another course without the permission of me and the instructor for that course (permission

will be granted only if the total work involved is commensurate with the amount of effort expected in both courses combined). Two-person projects may be permitted, provided the total work involved is about twice that of the typical single-person project.

The final exam will cover the entire course.

**Class Participation:** This class will be far more enjoyable for everyone if all students come to class ready and willing to discuss the material to be covered. I plan to reward those who participate in class by increasing their final grade by up to half a letter grade. I also reserve the right to add a similar negative “reward” for those who routinely come to class late, read the newspaper, sleep, or surf the Internet in class, etc.

**Topics:** This year, I am switching from Liu’s book as the main text to Buttazzo’s. The list of topics I currently plan to cover is given below. However, given that we are switching texts, there may be some inevitable changes to this list later. (Chapter numbers refer to Buttazzo’s book, unless specified otherwise.)

## Part I: Uniprocessor Scheduling of Independent Tasks.

- **Introduction to real-time systems (1 week).**

- J. Stankovic, “Misconceptions About Real-Time Computing,” *IEEE Computer*, Vol. 21, No. 10, October 1988, pp. 10-19.
- Chapter 1: A general view.
- Chapter 2: Basic concepts.

- **Classic uniprocessor scheduling results (4 weeks).**

- Static scheduling.
  - \* Chapter 5 of Jane Liu’s book: Cyclic executives.
- Dynamic scheduling.
  - \* Dynamic-priority scheduling:
    - Chapter 3, Section 3: Optimality of EDF.
    - Chapter 4, Section 4: Utilization-based schedulability test for EDF.
    - Nonpreemptive EDF from:  
K. Jeffay, D. Stanat, and C. Martel, “On Non-Preemptive Scheduling of Periodic and Sporadic Tasks,” *Proceedings of the 12th IEEE Real-Time Systems Symposium*, San Antonio, TX, December 1991, pp. 129-139.
  - \* Static-priority scheduling:
    - Chapter 4, Sections 3 and 5: Optimality of RM and DM.
    - Chapter 4, Section 3: Utilization-based schedulability test for RM.
    - Chapter 4, Section 5, supplemented by material from Jane Liu’s book: Demand-based scheduling conditions for static-priority systems.
  - \* Dealing with complexities arising in real systems.
    - Chapter 6, Section 8 from Jane Liu’s book: Practical considerations. (Skip 6.8.6 – 6.8.7.)
    - Timing analysis, from:  
C.M. Krishna and K.G. Shin, *Real-Time Systems*, McGraw Hill, pages 25-37.

- **Intractability results (1.5 weeks).**

- Preemptive systems.

- \* Dynamic-priority systems, from:  
S. Baruah, R. Howell, and L. Rosier, “Feasibility Problems for Recurring Tasks on One Processor,” *Theoretical Computer Science*, Vol. 118, pp. 3-20, 1993.
- \* Static-priority systems, from:  
J. Leung and J. Whitehead, “On the Complexity of Fixed-Priority Scheduling of Periodic, Real-Time Tasks,” *Performance Evaluation*, Vol. 2, No. 4, pp. 237-250, 1982.
- Nonpreemptive systems.
  - \* Dynamic-priority systems, from:  
K. Jeffay, D. Stanat, and C. Martel, “On Non-Preemptive Scheduling of Periodic and Sporadic Tasks,” *Proceedings of the 12th IEEE Real-Time Systems Symposium*, San Antonio, TX, December 1991, pp. 129-139.
  - \* Static-priority systems (no good reference here).

## Part II: Beyond Uniprocessor Independent Task Models.

### • Resource sharing (2 weeks).

- **Motivation:** “What Really Happened on Mars Rover Pathfinder,” by Mike Jones.
- Chapter 7: Priority inheritance and priority ceiling protocols, stack resource protocol.
- Resource sharing under EDF, from:  
K. Jeffay, “Scheduling Sporadic Tasks with Shared Resources in Hard-Real-Time Systems,” *Proceedings of the 13th IEEE Real-Time Systems Symposium*, Phoenix, AZ, December 1992, pp. 89-99.
- Lock-free approach, from:  
J. Anderson, S. Ramamurthy, and K. Jeffay, “Real-Time Computing with Lock-Free Objects,” *ACM Transactions on Computer Systems*, Vol. 15, No. 6, pp. 388-395, May 1997.  
  
J. Anderson and S. Ramamurthy, “A Framework for Implementing Objects and Scheduling Tasks in Lock-Free Real-Time Systems,” *Proceedings of the 17th IEEE Real-Time Systems Symposium*, pp. 94-105, December 1996.

### • Multiprocessors and distributed systems (1 week).

- Chapter 9, Section 3 from Jane Liu’s book: Multiprocessor priority ceiling protocol.
- Chapter 9, Section 4 from Jane Liu’s book: End-to-end scheduling.

### • Mixing real-time and non-real-time (2 weeks).

- Chapter 5: Fixed-priority servers.
- Chapter 6: Dynamic-priority servers.

### • Fairness (1 week).

- Proportional-share scheduling, from:  
I. Stoica, H. Abdel-Wahab, K. Jeffay, S. Baruah, J. Gehrke, and C.G. Plaxton, “A Proportional Share Resource Allocation Algorithm for Real-Time, Time-Shared Systems,” *Proceedings of the 17th IEEE Real-Time Systems Symposium*, pp. 288-299, 1996.

### • A quick look at “systems issues” (1.5 weeks — hopefully, some of the prior topics will go more quickly and we can spend more time on this topic).

- Chapter 9: Kernel design issues.
- Chapter 10: Application design issues.
- Chapter 11: Real-time operating systems and standards.

**Part III: Beyond This Class.** We have a weekly “real-time lunch” meeting where ongoing research in our real-time group is discussed. You are welcome to attend these meetings. Also, Sanjoy periodically teaches special topics courses on some aspect of real-time systems. Watch for those.