

EMBRY-RIDDLE AERONAUTICAL UNIVERSITY
Department of Computing and Mathematics
COURSE OUTLINE FOR

Course No.: CS450
Cr Hrs: 3

Title: Real Time Systems

Lecture Hours: 3

Laboratory Hours: 0

COURSE DESCRIPTION:

Introducing the concepts of real-time software systems from the user, designer, and programmer viewpoint. The connection of an external process to a digital computer by means of hardware and software interface is discussed. The structure, programming, and basic properties of real-time systems are described with an overview of system software. Related topics as interrupts, concurrent task synchronization, sharing resources, and reliability factors are discussed. Real-time software development in a host-target environment is presented and working on a prototype application will be encouraged. Required is knowledge of computer operations including input/output, interrupt interface, and familiarity with operating systems concepts, and software engineering lifecycle. Required is proficiency in a high-level language programming (working knowledge of C or Ada on UNIX system) including passing data between program modules, operating with input/output, and using memory resources.

GOALS:

The purpose of the course is to have students: understand the concepts of real-time process and control, and of a computer as a real-time machine; represent and implement a real-time software system using established software engineering methodologies and development platforms; understand the concepts of inter-process interface and multitasking; understand issues of time-critical computing; be familiar with a real-time system and application software; appreciate the role of real-time systems in target-based aviation/aerospace applications.

PERFORMANCE OBJECTIVES:

As the result of the course instruction the students will be able to:

1. Understand concepts of time-critical computing and identify real-time systems.
2. Get familiar with a host-target development environment for time-critical systems.
3. Write multitasking computer programs with inter-task communication and synchronization.
4. Apply concepts of inter-task communication and synchronization via shared memory, message queues, signals, semaphores, mailboxes.
5. Understand real-time kernels and task scheduling.
6. Have basic understanding of the interfacing hardware.

7. Understand concepts of reliability in relation to real-time software
8. Relate the course experience to aviation applications.

Department of Computing and Mathematics
COURSE OUTLINE FOR CS450, Continued

TEXTBOOK:

Real-Time Systems and their Programming Languages, by Alan Burns and Andy Wellings, Addison Wesley, 1997, ISBN 0-201-40365
Wind River System Tornado Manuals (HTML version on line at the workstations running Tornado, access to hard copies in the LB131 Real-Time Software Project Lab)

SUGGESTED SUPPLEMENTAL MATERIALS:

- a. *Real-Time Systems Design and Analysis*, P.A. Laplante, IEEE Press, 1993
- b. *POSIX.4: Programming for the Real World*, B.O. Gallmeister; O'Reilly & Associates, 1995
- c. *Real-Time Software Systems*, J.E. Cooling; PWS Publishing Company, 1997
- d. *Practical UNIX Programming*, K.A. Robbins, S. Robbins; Prentice Hall, 1996 with source examples on ftp://vip.cs.utsa.edu in pub/pup directory
- e. *Advanced Programming in the Environment*; W.R. Stevens; Addison Wesley, 1993
- f. *Software Design Methods for Concurrent and Real-Time Systems*, H. Gomaa; Addison-Wesley Publishing Company, 1993
- g. *Software Systems Construction with Examples Ada*, B. Sanden, Prentice Hall, 1994
- h. *Designing Large Real-Time Systems in Ada*, K.Nielsen, K.Shumate; McGraw Hill, 1989,
- i. *Real-Time Microprocessor Systems*, S. Savitzky; Van Nostrand Reinhold, 1985,
- j. *Strategies for Real-Time System Specification*, D.J.Hatley, I.A.Pirbhai; Dorset House Publ.,1987,
- k. *An Implementation Guide to Real-Time Programming*, L.Ripps; Prentice Hall, 1989,
- l. *Software Specification and Design: A Disciplined Approach for Real-Time Systems*, K. Shumate, M. Keller; John Wiley, 1992,
- m. *Real-Time Software*, edited by R.L. Glass; Prentice Hall, 1983,
- n. *Introduction to Real-Time Software Design*,S.T.Allworth, R.Zobel; Springer-Verlag NewYork Inc., 1987,
- o. *Structured Development for Real-Time Systems*, P.T.Ward, S.J.Mellor; Prentice Hall, 1985,
- p. An access to the Web to download the materials. An access to library with current articles in such magazines as: Embedded Computing, Transactions of ACM, IEEE Computer, Dr. Dobbs Journal, Real Times, etc.
- q. An access to a computer laboratory with a fast, user friendly, high-level language environment.

PREREQUISITE KNOWLEDGE AND TOPICS:

1. Required knowledge of computer operations including input/output and interrupt interface.
2. Required is proficiency in a high level language programming including passing data between program modules, operating with input/output, and using memory resouces.
3. Expected is familiarity with operating system concepts and software engineering lifecycle and process

TOPIC	CLASS HOURS	COURSE OBJECTIVES
1. Real-Time Basics	6	Understand concepts of time-critical computing and identify real-time systems. Get familiar with a host-target development environment for time-critical systems.
2. Real-Time Systems Development and Platforms	3	Get familiar with a host-target development environment for time-critical systems. Understand real-time kernels and task scheduling.
3. RT Programming and Timing Concepts	6	Understand concepts of time-critical computing and identify real-time systems. Apply concepts of inter-task communication and synchronization via shared memory, message queues, signals, semaphores, mailboxes.
4. Concurrency and Multitasking	6	Write multitasking computer programs with inter-task communication and synchronization. Apply concepts of inter-task communication and synchronization via shared memory, message queues, signals, semaphores, mailboxes.
5. Semaphores	3	Understand concepts of time-critical computing and identify real-time systems. Apply concepts of inter-task communication and synchronization via shared memory, message queues, signals, semaphores, mailboxes.
6. Shared Resources and Mutual Exclusion	3	Understand concepts of time-critical computing and identify real-time systems. Apply concepts of inter-task communication and synchronization via shared memory, message queues, signals, semaphores, mailboxes.
7. Message Queues	3	Understand concepts of time-critical computing and identify real-time systems. Apply concepts of inter-task communication and synchronization via shared memory, message queues, signals, semaphores, mailboxes.
8. Signals, Interrupts, and Timers	3	Understand concepts of time-critical computing and identify real-time systems. Apply concepts of inter-task communication and synchronization via shared memory, message queues, signals, semaphores, mailboxes. Have basic understanding of the interfacing hardware.
9. Task Scheduling	3	Understand concepts of time-critical computing and identify real-time systems. Understand real-time kernels and task scheduling.

TOPIC (cont.)	CLASS HOURS	COURSE OBJECTIVES (cont.)
10. RT Software Engineering Notation	6	Understand concepts of time-critical computing and identify real-time systems. Get familiar with a host-target development environment for time-critical systems.
11. Reliability and Exception Handling	3	Understand concepts of reliability in relation to real-time software.

LABORATORY AND COMPUTER USAGE:

Validated, user-friendly high-level language environment (ADA, C, Fortran) with possible real-time expansions, capable of interfacing with a low assembly-level system functions. The best solution is a single-user workstation or a dedicated real-time computer with an appropriate real-time operating system. Such crucial real-time issues as the timing, interrupts handling, and input/output interface, can be only resolved when all system resources are under user control. Thus it is necessary to have an assembler environment and an exhaustive computer system documentation (port addresses, interrupt numbers, reserved memory areas, special functions). The system should provide for an easy interface with external devices by the means of e.g. RS232 serial interface standard. The communication software should be available. The laboratory setting should provide for a close proximity of available external real-time systems (flight trainers, weather equipment, ATC console) with well defined interfaces.

GRADING SYSTEM:

- 4-6 individual laboratory assignments
- half-semester long project for teams of 6-8 students, including report and presentation
- tests: three midterm tests and optional comprehensive final make-up test.
- quizzes: 5-7 short (unannounced) quizzes will be given at the end of lecture to check the class progress.

grading percentages:

lab reports 20%, quizzes 10%, tests 30%, project 20%, final exam 20%

letter grade:

A - above 90%, B - above 80%, C - above 70%, D - above 60%, F - below 60%

ESTIMATED CONTENT:

Skills: 25 %
Content: 75 %