

CHALMERS

Computer Engineering

Computer Science and Engineering – Chalmers University of Technology and Göteborg University

EDA222 / DIT160 - Realtidssystem Real-Time Systems

7.5 higher education credits, Quarter 3, 2007/08

This web page functions both as a course description sheet and as a medium for distribution of new information regarding the course. The information will be updated dynamically, so please visit this web page at least once per week. Information regarding course syllabus, examination results, and submission of homework assignment is available at the [course's home page at the Student Portal](#).

Contents:

- [Staff](#)
 - [Course outline](#)
 - [Course organization](#)
 - [Course literature](#)
 - [Course documents](#)
 - [Lecture schedule](#)
 - [Reading guidelines](#)
 - [Miscellaneous \(relevant literature and Internet material\)](#)
-

News: **UPDATED** (January 15, 2008)

080115: Updated the official home page of the course in **Real-Time Systems** for 2007/08.

Lecturer and examiner:

Docent [Jan Jonsson](#)

Room: 4472 (EDIT house)

Phone: 031-772 5220

E-mail: janjo@chalmers.se

Course assistants:

Dr. [Arne Dahlberg](#)

4466 (EDIT house)

031-772 1705

dahlberg@chalmers.seDr. [Roger Johansson](#)

4455A (EDIT house)

031-772 5729

roger@chalmers.se

Course Outline:

A real-time system is a computer system in which the correctness of the system depends on the time when results are generated. Real-time systems interact with a more or less time-critical environment. Examples of real-time systems are control systems for cars, aircraft and space vehicles, manufacturing system, financial transaction systems, computer games and multimedia applications. This course is intended to give basic knowledge about methods for the design and analysis of real-time systems.

After the course the students shall be able to:

- Construct concurrently executing software for real-time applications that interface to input/output units such as sensors and actuators.
- Describe the principles and mechanisms used for designing real-time kernels and run-time systems.
- Describe the mechanisms used for time-critical scheduling of tasks.
- Apply the basic analysis methods used for verifying the temporal correctness of a set of executing tasks.

In the design of real-time systems it is practical to implement the application software as multiple concurrently executing processes, where each process is responsible for a given task in the system. The concept of multiple processes provides for an intuitive way of decomposing a complex system into smaller parts that are simple to comprehend and implement.

This course uses Ada as the main programming language because of its powerful support for programming of concurrent processes. In particular, the course demonstrates how language constructs such as rendezvous and protected objects are used for implementing communication/synchronization between processes, resource

management and mutual exclusion. Since other programming languages use monitors or semaphores to implement these functions, the course also contains a presentation of such techniques. In addition, the course demonstrates how to use low-level programming in Ada to handle interrupt-driven communication with input and output devices. To demonstrate the general principles in real-time programming, the course also gives examples of how these techniques are implemented in other programming languages, such as C and Java.

In order to execute a program containing multiple concurrent processes there is a real-time kernel (run-time system) that distributes the available capacity of the microprocessor among the processes. The course shows how a simple real-time kernel is organized.

The real-time kernel determines the order of execution for the processes by means of a scheduling algorithm. To that end, the course presents techniques based on cyclic time-table based scheduling as well as scheduling techniques using static or dynamic process priorities. In addition, protocols for the management of shared hardware and software resources are presented.

In real-time systems with strict timing constraints it is necessary to make a pre-run-time analysis of the system schedulability. The course presents three different analysis methods for systems that schedule processes using static or dynamic priorities: utilization-based analysis, response-time analysis, and processor-demand analysis. In conjunction with this, the course also gives an account on how to derive the maximum resource requirement (worst-case execution time) of a process.

Organization:

Lectures:

The course is organized as a series of lectures where fundamental theories and concepts are presented. Lectures are given at two occasions per week (except study week 1 and 2 where there is extra lectures on Wednesdays at 08.00-09.45 in room EL42):

Tuesdays at 10.00-11.45 in room EL43 Thursdays at 13.15-15.00 in room EL43

Exercise sessions:

As a complement to the lectures, there will be exercise sessions on the specific topics covered during the main lectures. At each session, a course assistant gives a mini lecture on selected parts of the course contents, and also discusses selected problems from the exercise compendium. The remaining time is for discussing solutions to the selected problems or the laboratory assignment. Exercise sessions are offered on one occasion per week:

Thursdays at 15.15-17.00 in room EL43

Laboratory assignment:

To achieve practical experience of the construction of a real-time system, a compulsory laboratory assignment is included in the course. The purpose of the assignment is to control a simulated train system with the aid of concurrent tasks on a microprocessor. The software tasks should be able to handle switches, sensors and speed control of two train sets. The assignment encompasses control of external units using interrupt handling. The software is programmed using Ada 95.

The laboratory sessions are administrated via the Student Portal and is done in the following way. First, you register for a **group** ([link to group registration](#)) and will thereby be assigned a group name (which is also the name of your account on the laboratory computer system). Then, you **book** time slots in the laboratory using the assigned group name ([link to booking of laboratory sessions](#)). Please note that you must be logged on to the portal in order to be able to use these services.

The laboratory sessions are done in groups of at most two students, and take place in the course laboratories of the Division of Computer Engineering (southern part of the EDIT building, floor 4) during study weeks 2 through 7.

The booking of time slots for the laboratory sessions can be done on February 1 at the earliest.

The following tentative time slots are available for the laboratory sessions (at most 12 groups per time slot):

Monday 17.00-21.00 Wednesday 08.00-12.00 (study week 3-7)

To avoid a shortage of laboratory session time slots at the end of the course, every group must book (and attend) one time slot per week during study weeks 2, 3, 4, 5 and 6. Since the number of time slots per week are limited,

not all groups will be able to book more than one time slot per week. Therefore, it is not allowed to book more than one time slot in advance. Booking of additional time slots may be done on the day before the current time slot at the earliest. Booked time slots that are not being used will not be compensated for.

Approval of the laboratory assignment is done in the following way. The student demonstrates a functioning program for the course assistant. At the same time, the student should explain how the software is constructed, using access graphs and program listings. Finally, a laboratory report should be written and electronically submitted via the Student Portal ([link to report submission](#)).

Deadline for submitting the laboratory report is March 7, 2008. Deadline for final approval is March 28, 2007.

Student reception and consultation:

Questions related to course contents and laboratory assignment are primarily answered in conjunction with the lectures, exercise sessions and laboratory sessions. Otherwise, questions are answered by the course responsible at the following reception time:

Wednesdays at 13.15-13.45 in room 4472 (EDIT house)

Examination:

The student is evaluated through a final written exam. An approved laboratory assignment and passing the written exam yields a final grade (scale is U, 3, 4, 5). The following material is permitted to use during the written exam:

- Excerpt from *Ada 95 Reference Manual*
- *Ada Distilled*
- *Ada vs. Java* (Quick Reference)
- Type approved calculator.

For the final exam the following rules of thumb apply:

- The lecture notes (PowerPoint hand-outs + blackboard notes) represent the level of which algorithms and methods should be known and understood for the final exam and for your future role as an engineer. Thus, it is more important to understand concepts and principles rather than specific and complicated implementation details. You will acquire implementation skills in due time through your practical work as an engineer.
- Let the list of "suggested reading" provided below be a guide rather than a cook book for using the textbook. Use the lecture notes, the textbook and some common sense to avoid spending significant time on less relevant material. The student should become comfortable with the material such that general concepts and principles are well understood and can be applied.
- Older written exams indicate the type of problems that can possibly appear on a written exam. The focus on the written exam will be on principles and theory, while a smaller number of problems requires solutions in the form of program code. In the latter case, it is more important to describe the program structure and functionality than producing a syntactically correct code.

The ordinary examination takes place on Wednesday, March 12, 14.00 - 18.00 in the V building.

The first re-examination will take place on August 12, 2008 at 08.30 - 12.30 in the V building.

IMPORTANT! As of the academic year 2005/06, there are compulsory sign-ups for final written exam. You can sign up as early as six weeks before but no later than two weeks prior to the beginning of the examination period. Cancellation of a sign-up must be made one week or more prior to the date of the exam. Examination sign-up is made via the Student Portal ([link to examination sign-up](#)).

Literature:

- Alan Burns and Andy Wellings, *Real-Time Systems and Programming Languages*, 3rd edition, Addison-Wesley, 2001, ISBN 0-201-72988-1 (can be purchased at Cremona and at DC)

What parts of the book that are relevant are listed in the following [reading guidelines](#). (Reading guidelines for the old course book *Concurrency in Ada* are available [here](#).)

- Recommended course material (available in electronic formats)

- Ken Tindell, *Real Time Systems and Fixed Priority Scheduling* [PDF]
 - Exercise compendium [PDF]
 - Laboratory assignment compendium [PDF] (+ skeleton source files [.zip])
 - GNU Ada95 cross compiler for M68xx0 [PDF]
 - Excerpt from *Ada95 Reference Manual (ARM)* [PDF] Permitted to use at the written exam.

 - Additional course material related to Ada (available in electronic formats)
 - Ada Distilled [PDF] Permitted to use at the written exam.
 - Quick Ada [html, PDF]
 - Quick Reference Ada vs. Java [PDF] Permitted to use at the written exam.
 - GNAT GPL Edition Ada compiler [.exe]
-

Course Documents:

- Course information sheet ([here](#))
- Lecture notes (available [here](#))
- Written exams (available [here](#))
- Status of approval for laboratory assignment: (available [here](#))

*The course information and lecture notes are available as both Postscript and PDF files.
Click here if to download a reader for these formats: ([Ghostview](#), [Acrobat Reader](#))*

Related information:

- Ada related links:
 - [Ada Home: The Web Site for Ada](#)
 - [GNAT GPL Edition Ada compiler](#)
 - [GRASP Ada Editor](#)
 - Java related links:
 - Article in IEEE Computer about [Real-Time Java](#)
 - Article in IEEE Spectrum about [The Risks with Parallel Programming in Java](#)
 - Links related to the problem with Mars Pathfinder:
 - Mike Jones' [report from RTSS'97](#)
 - Glen Reeves' (JPL) [comments](#)
 - Links to international archival journals that cover the topic of real-time systems design:
 - [Real-Time Systems: The International Journal of Time-Critical Computing Systems](#)
 - [IEEE Transactions on Computers](#)
 - [IEEE Transactions on Parallel and Distributed Systems](#)
 - [IEEE Transactions on Software Engineering](#)
 - Books that deal with the topic of real-time systems design:
 - C. M. Krishna and K. G. Shin, *Real-Time Systems*, McGraw-Hill, 1997, ISBN 0-07-114243-6.
 - H. Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications*, Kluwer Academic Press, 1997, ISBN 0-7923-9894-7.
 - G. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, Kluwer Academic Press, 1997, ISBN 0-7923-9994-3.
 - Links to research activities in real-time systems design:
 - in Sweden ([ARTES](#) och [SNART](#))
 - internationally ([IEEE CS Technical Committee on Real-Time Systems](#))
 - Link to advanced course in [Parallel and Distributed Real-Time Systems](#)
-

Last modified January 15, 2008 by [Jan Jonsson](#)

CHALMERS

Computer Engineering

Computer Science and Engineering – Chalmers University of Technology and Göteborg University

EDA222 / DIT160 - Realtidssystem Real-Time Systems

Lecture schedule:

| Activity | Week | Day | Time | Room | Topic |
|----------|------|-----------|-------------|------|---|
| L1 | 4 | Tuesday | 10.00-11.45 | EL43 | Introduction; Design methods for real-time systems |
| L2 | 4 | Wednesday | 08.00-09.45 | EL42 | Concurrent programming; Cooperating tasks; Rendezvous in Ada 95 |
| L3 | 4 | Thursday | 13.15-15.00 | EL43 | Shared data structures; Mutual exclusion; Protected objects in Ada 95 |
| L4 | 5 | Wednesday | 08.00-09.45 | EL42 | Clocks, time and task priorities in Ada 95 |
| L5 | 5 | Thursday | 13.15-15.00 | EL43 | Low-level programming in Ada 95 |
| L6 | 6 | Tuesday | 10.00-11.45 | EL43 | Resource management; Deadlock and starvation |
| L7 | 6 | Thursday | 13.15-15.00 | EL43 | Monitors and semaphores; Mutual exclusion (cont'd) |
| L8 | 7 | Tuesday | 10.00-11.45 | EL43 | Design methods for real-time kernels |
| L9 | 7 | Thursday | 13.15-15.00 | EL43 | Fault-tolerance and network communication |
| L10 | 8 | Tuesday | 10.00-11.45 | EL43 | Task model; Worst-case execution times |
| L11 | 8 | Thursday | 13.15-15.00 | EL43 | Scheduling: terminology, cyclic executives |
| L12 | 9 | Tuesday | 10.00-11.45 | EL43 | Scheduling: static and dynamic priorities, utilization-based analysis |
| L13 | 9 | Thursday | 13.15-15.00 | EL43 | Scheduling: response-time analysis |
| L14 | 10 | Tuesday | 10.00-11.45 | EL43 | Scheduling: processor-demand analysis; Summary and reading hints |

Exercise session schedule:

| Activity | Week | Day | Time | Room | Topic |
|----------|------|----------|-------------|------|---|
| E1 | 4 | Thursday | 15.15-17.00 | EL43 | Introduction to Ada 95, tasks and protected objects |
| E2 | 5 | Thursday | 15.15-17.00 | EL43 | Low-level programming and exception handling in Ada 95 |
| E3 | 6 | Thursday | 15.15-17.00 | EL43 | Interrupts in Ada 95; The laboratory assignment: model, description and specification |
| E4 | 7 | Thursday | 15.15-17.00 | EL43 | Low-level synchronization |
| E5 | 8 | Thursday | 15.15-17.00 | EL43 | Interfacing Ada 95 to C and assembly language |
| E6 | 9 | Thursday | 15.15-17.00 | EL43 | Worst-case execution times; Scheduling |
| E7 | 10 | Thursday | 15.15-17.00 | EL43 | Scheduling (cont'd) |

Last modified January 15, 2008 by [Jan Jonsson](#)

EDA222 / DIT160 - Realtidssystem Real-Time Systems

Reading guidelines:

Burns & Wellings, "Real-Time Systems and Programming Languages":

| Chapter | Contents | Recommended | Overview |
|---------|---|---|-----------------------------|
| 1 | Introduction to real-time systems | 1.1-1.3 | |
| 2 | Designing real-time systems | 2.1-2.3, 2.6-2.9 | 2.4-2.5 |
| 3 | Programming in the small | 3.1-3.6 | |
| 4 | Programming in the large | 4.1-4.3, 4.4 (pp. 81-83), 4.5 (pp. 91-94) | |
| 5 | Reliability and fault tolerance | 5.1-5.5, 5.8, 5.10 | 5.6-5.7, 5.9 |
| 6 | Exceptions and exception handling | 6.2, 6.3.1 | |
| 7 | Concurrent programming | 7.3.6, 7.4 | 7.1-7.2, 7.3.1-7.3.4, 7.3.8 |
| 8 | Protected objects | 8.6 (pp. 246-248), 8.6.5, 8.7 | 8.1-8.3, 8.4.1-8.4.5, 8.4.7 |
| 9 | Message based synchronization and communication | 9.1-9.2, 9.3.2, 9.4.2-9.4.3 | 9.3.3, 9.4 (pp. 292-293) |
| 11 | Resource control | 11.1-11.2, 11.7 | 11.3-11.4 |
| 12 | Real-time facilities | 12.1, 12.2 (pp. 411-412), 12.2.2, 12.3, 12.5-12.6 | 12.4, 12.7.1, 12.8.1 |
| 13 | Scheduling | 13.1-13.7, 13.8 (pp. 481-482), 13.8.1, 13.9, 13.11 (pp. 490-491), 13.11.1-13.11.2, 13.14 (pp. 504-505), 13.14.1 | |
| 14 | Distributed Systems | 14.1-14.2, 14.7.2 (pp. 567-569) | 14.7 |
| 15 | Low-level programming | 15.1-15.2, 15.4, 15.8 | |
| 16 | The execution environment | 16.1 | 16.2-16.3 |

We recommend that, for all chapters mentioned above, the student should also read the associated Introduction and Summary texts. Students who are familiar with the programming languages Java or C may also find it helpful to consult the parts of the book that describe solutions written in these languages in conjunction with studying the corresponding solutions in Ada 95.

Tindell, "Real-Time Systems and Fixed Priority Scheduling":

| Chapter | Contents | Recommended | Overview |
|---------|------------------------------------|-------------|-----------|
| 1 | Introduction | 1.1 - 1.5 | |
| 2 | Estimating program execution times | 2.1 - 2.5 | |
| 3 | Two scheduling approaches | 3.1 - 3.4 | |
| 4 | Fixed-Priority Scheduling | 4.1 - 4.5 | 4.6 - 4.7 |

Last modified January 15, 2008 by [Jan Jonsson](#)