Embedded Systems

Code: 102791
ECTS Credits: 6

2025/2026

Degree	Type	Year
Computer Engineering	OB	3
Computer Engineering	OT	4

Contact

Name: Lluis Ribas Xirgo

Email: lluis.ribas@uab.cat

Teachers

Joaquin Saiz Alcaine

Teaching groups languages

You can view this information at the end of this document.

Prerequisites

For a full understanding of the contents of the course, it is convenient to have a basic ability in programming and a good knowledge of how programs execute in computers. For this, you should have completed Programming laboratory and Software engineering courses (programming), as well as Computer structure, Operating systems and Computer architecture (program execution model).

Objectives and Contextualisation

This course is the first in the subject of Design of application-oriented computing systems, which deals with the development of systems that run the algorithms of specific applications under a set of stringent requirements. For example, it is not enough for a mobile device to be capable of displaying a video, it must be done at 25 images per second, synchronizing it with audio and consuming as little energy as possible. Therefore, the main goal of the subject is to know how to design algorithms and the implications that each design has in the cost of the application depending on the very algorithm and how it is implemented, that is, in accordance with the chosen execution platform.

In this context, the Embedded systems' course objective is that students acquire the following competences:

- Understand the various fields of application of embedded systems.
- Have notions of the usual requirements of each application domain, including those in real time.
- Understand the aspects of security, reliability and robustness of the systems.
- Understand the methodology of the development of embedded systems.
- Understand the various systems' computational models.

- Have practical skills with the design and manipulation of state-based computational models.
- Understand the basic elements of embedded system architectures.
- Have acquired the rudiments of platform-based design.
- Know how to estimate implementation costs based on the computational models of the systems.
- Understand the problem of partitioning systems and be aware of the different approaches to doing it.
- Have the rudiments of programming and hardware description languages for the implementation of the systems.

Competences

Computer Engineering

- Acquire personal work habits.
- Capacity to design, develop, evaluate and ensure the accessibility, ergonomics, usability and security of computer systems, services and applications, as well as of the information that they manage.
- Communication.
- Have the capacity to define, evaluate and select hardware and software platforms for the development and execution of computer systems, services and applications.
- Have the capacity to develop specific processors and embedded systems and to develop and optimise the software of said systems.
- Work in teams.

Learning Outcomes

1. Accept and respect the role of the various team members, and its different levels of dependence.
2. Analyse the requirements of computer applications.
3. Compare and evaluate the possible platforms that can fulfil the requirements of applications.
4. Identify the needs of the specific application that must be resolved.
5. Identify the security needs that embedded systems have to fulfil.
6. Manage time and resources available. Work in an organized manner .
7. Use English as the language of communication and professional relations .
8. Work independently.

Content

1. Introduction
1.1. Embedded systems development process
1.2. Controllers based on state machines
1.3. State machine programming

2. Models of computation
2.1. Extended Finite State Machines (EFSMs)
2.2. Concurrent and Hierarchical Extended Finite State Machines (HCEFSMs)
2.3. Algorithmic State Machines (ASMs)
2.4. Data Flow Graphs (DFGs)
2.5. Control Data Flow Graphs (CDFGs)

3. Development of embedded systems
3.1. Architecture of complex systems
3.2. Formal verification of state-oriented systems
3.3. Software synthesis
3.4. Simulation
3.5. Real time systems

Activities and Methodology

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Laboratory: Course project development	12	0.48	2, 1, 3, 6, 8
Problem-solving: Problem solution proposals and discussion	12	0.48	2, 1, 3, 6, 8
Theory: Attendance and participation in theory classes	26	1.04	4, 5
Type: Supervised			
Course project follow-up tutoring	6	0.24	2, 3, 6, 8
Type: Autonomous			
Assignment: Project development and report writing	12	0.48	2, 1, 3, 6, 8, 7
Problem solving and report writing	24	0.96	2, 1, 3, 6, 8, 7
Study	23	0.92	4, 5, 8

Teaching is structured in the following face-to-face activities:

Theory classes: Presentations of course contents, with a first part that is devoted to the dissemination of the necessary knowledge for the analysis and the design of embedded systems, and to explain cases that situate in context the knowledge and the abilities that are acquired during the course. The second part is devoted to the discussion of problems that will be dealt with in the corresponding seminars.

Problem-solving seminars: Discussion of case studies to consolidate theoretical knowledge regarding the analysis and design of embedded systems.

Laboratory practices: Teamwork at the laboratory, following a walk-through guide under the supervision of a teacher. Each session deals with a specific aspect regarding the implementation of embedded systems.

There is a very important part of teamwork outside the classroom and the laboratory. In this sense, each member of each team will have to assume different roles for each assignment. This also means having to work in an organized way and know how to work autonomously when appropriate.

TRANSVERSAL COMPETENCES

In this course, students are expected to acquire autonomy and organizational skills in their own work as well as basic competence in teamwork, and in English.

In this sense, there will be a part of the assessment of each of the corresponding learning results:

T03.02. Assume and respect the role of each member of the team, as well as the different levels of team dependence: The course project is teamwork, and the final report and defense must detail what each member has done.

T02.01. Work autonomously: Each person must solve problems and do the corresponding part of the project on its own.

T02.03. Manage time and available resources. As for the course project, teams must work in an organized way and each member must commit to his or her role. In any case, all late submissions have penalties dependingon the delay, which compels students to manage the time so that their team can meet the required time requirements.

T04.03. Use English as the language of communication and professional relationship of reference: Most of the material is in English to promote the learning of this language and it is positively valued that the reports of the project or the solutions are written in English.

Annotation: Within the schedule set by the centre or degree programme, 15 minutes of one class will be reserved for students to evaluate their lecturers and their courses or modules through questionnaires.

Assessment

Continous Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Continuous assessment assignments	25	5	0.2	2, 3, 6, 4, 5, 8, 7
Final exam	25	2	0.08	4, 5
Make-up exam	50	2	0.08	4, 5
Midterm exam	25	2	0.08	4, 5
Project follow-up reports (6)	25	24	0.96	2, 1, 3, 6, 8, 7

a) Procedure and activities' plan

The assessment is continuous with specific activities (exams and assignments) throughout the course. These assessment activities generate a series of grades that determine the final grade.

The calculation of the final grade *n* follows the following expression:

$$n = x \cdot 50\% + p \cdot 25\% + c \cdot 25\%$$

where *x* is the grade for the exams; *p*, that for the laboratory project, and *c*, that for the continuous assessment.

If $x < 5$ or $p < 5$, the final grade *n* is, at most, a 4.5. In other words, the average of the exams and the project must be passed separately.

Exam grade *x* is the average of the midterm exam and the final exam.

Project grade *p* is obtained from the weighted average of the grades corresponding to each lab session. Six are planned. In case of non-attendance, the absent person will receive a 0 as the grade for the corresponding session.

Continuous assessment grade *c* is obtained from a weighted average of the problem-solving assignments completed throughout the course. There is no minimum and, therefore, the course can be passed with $c = 0$ as long as $x \cdot 50\% + p \cdot 25\% \geq 5$.

b) Assessment activities schedule

The dates of all face-to-face activities, including assessment activities, and submission deadlines will be published on the virtual campus (CV) and may be subject to possible changes in programming for reasons of adaptation to possible incidents: they will always be previously informed through the CV since it is the usual mechanism for exchanging information between teachers and students outside the classroom.

c) Re-assessmentprocedures

In accordance with the coordination of the Degree and the deanship of the School of Engineering, the following activities are not recoverable:

- Project, 25% of the final grade

- Continuous assessment, 25% of the final grade

The average grade of the exams can be recovered with a specific make-up exam.

d) Assessment review procedure

Assessment activities can be reviewed any time after corresponding grades are published and before the deadline for the revision of the final exam.

Should the change of a grade be agreed upon, that grade may not be modified in a later review.

No reviews will be done after the closure of the reviews of the final exam, but for the make-up exam.

e) Grading

A "non-assessable" grade is assigned to students that have not participated in any assessment activity. In any other case, not participating in an assessment activity scores 0 in the weighted average computation.

Honors will be awarded to those who obtain grades greater than or equal to 9.0 in each part, up to 5% of those enrolled in descending order of final grades. They may also be granted in other cases if they do not exceed 5% and the final grade is equal to or greater than 9.0.

f) Irregularities, copies and plagiarism

Copies are evidence that the work or the examination has been done in part or in full without the author's intellectual contribution. This definition also includes attempts to copy in exams and reports, and violations of the norms that ensure intellectual authorship. Plagiarisms refer to the works and texts of other authors that are passed on as their own. They are a crime against intellectual property. To avoid plagiarism, quote the sources you use when writing the corresponding work reports or examinations.

In accordance with the UAB regulations, copies or plagiarisms or any attempt to alter the assessment result, for oneself or for others, like e.g. letting other copy, imply a final grade for the corresponding part (exam, continuous assessment or project) of 0 in the computation of the final score and failing the course. This does not limit the right to act against perpetrators, both in the academic field and in the criminal.

The use of Artificial Intelligence (AI) technologies as an integral part of the development of the work is permitted, but not recommended, provided that the result reflects a significant contribution by the student in the analysis and personal reflection. The student must clearly identify which parts have been generated with this technology, specify the tools used and include a critical reflection on how they have influenced the process and the result of the activity. The lack of transparency in the use of AI is considered a lack of academic honesty and entails a penalty in the grade of the activity, or greater sanctions in serious cases.

g) Assessment of repeaters

There is no differentiated treatment for repeaters, but they can take advantage of their own material from the previous year provided it is informed in the corresponding reports.

h) Single assessment

This course does not have a single assessment procedure.

Bibliography

Ll. Ribas Xirgo. (2014). *How to code finite state machines (FSMs) in C. A systematic approach.* TR01.102791 Embedded Systems. Universitat Autònoma de Barcelona. [https://www.researchgate.net/publication/273636602_How_to_code_finite_state_machines_FSMs_in_C_A_syste

Explains a method to program state machines in C which resembles the one presented in the course.

Ll. Ribas Xirgo. (2011). "Estructura bàsica d'un computador", Capítol 5 de Montse Peiron Guàrdia, Lluís Ribas i Xirgo, Fermín Sánchez Carracedo i A. Josep Velasco González: *Fonaments de computadors*. Material docent de la UOC. OpenCourseWare de la UOC. [http://openaccess.uoc.edu/webapps/o2/handle/10609/12901]
It describes the state-based machine model, the algorithmic machines, and the digital systems' basic architectures that are used in the course from a different perspective, though.

Edward A. Lee and Sanjit A. Seshia. (2017) *Introduction to Embedded Systems, A Cyber-Physical Systems Approach*, Second Edition, MIT Press.
A full course on embedded systems with much more theoretical background. (See also: https://ptolemy.berkeley.edu/)

M. J. Pont. (2005). *Embedded C*. Pearson Education Ltd.: Essex, England.
It shows how embedded systems are programmed, topic also treated in the course problem-solving part and laboratory. Therefore, it's an interesting complementary material.

Brian Bailey, Grant Martin and Andrew Piziali. (2007). *ESL Design and Verification. A Prescription for Electronic System-Level Methodology*. Elsevier.
It gives an overview of the embedded systems' synthesis process and situates the course material, thus it's a good complement.

Tim Wilmshurst. (2010). *Designing Embedded Systems with PIC Microcontrollers.Principles and Applications (Second Edition)*. Elsevier.
Complementary information to that of the course on an embedded system for controlling a robot.

Oliver H. Bailey. (2005). *Embedded Systems Desktop Integration*. Wordware Publishing.
Complementary information about the hardware-software communication aspect of embedded systems.

Software

CoppeliaSim, EDU Version, Coppelia Robotics [https://www.coppeliarobotics.com/]

ZeroBrane Studio, ZeroBrane [https://studio.zerobrane.com/]

Draw.io, diagrams.net [https://app.diagrams.net/]

Groups and Languages

Please note that this information is provisional until 30 November 2025. You can check it through this link. To consult the language you will need to enter the CODE of the subject.

Name	Group	Language	Semester	Turn

(PAUL) Classroom practices	431	Catalan	first semester	morning-mixed
(PAUL) Classroom practices	432	Catalan	first semester	morning-mixed
(PLAB) Practical laboratories	431	Catalan/Spanish	first semester	morning-mixed
(PLAB) Practical laboratories	432	Catalan/Spanish	first semester	morning-mixed
(PLAB) Practical laboratories	433	Catalan/Spanish	first semester	morning-mixed
(PLAB) Practical laboratories	434	Catalan/Spanish	first semester	morning-mixed
(TE) Theory	430	Catalan	first semester	morning-mixed