COURSE OUTLINE

(1) GENERAL

SCHOOL	ENGINEERING		
ACADEMIC UNIT	ELECTRICAL AND COMPUTER ENGINEERING DEPT.		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	**ECE_ ELE910**	**SEMESTER**	**9**
COURSE TITLE	EMBEDDED SYSTEMS II		

INDEPENDENT TEACHING ACTIVITIES *if credits are awarded for separate components of the course, e.g. lectures, laboratory exercises, etc. If the credits are awarded for the whole of the course, give the weekly teaching hours and the total credits*	**WEEKLY TEACHING HOURS**	**CREDITS**
Lectures	3	
Seminars / Practice exercises	1	
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (4).	4	5

COURSE TYPE *general background, special background, specialised, general knowledge, skills development*	Specialised
PREREQUISITE COURSES:	No. Students are advised to have already attended the course: EMBEDDED SYSTEMS I
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek
IS THE COURSE OFFERED TO ERASMUS STUDENTS	Yes
COURSE WEBSITE (URL)	https://www.ece.uop.gr/

(2) LEARNING OUTCOMES

Learning outcomes

The course learning outcomes, specific knowledge, skills and competences of an appropriate level, which the students will acquire with the successful completion of the course are described.

Consult Appendix A

- *Description of the level of learning outcomes for each qualifications cycle, according to the Qualifications Framework of the European Higher Education Area*
- *Descriptors for Levels 6, 7 & 8 of the European Qualifications Framework for Lifelong Learning*

and Appendix B

- *Guidelines for writing Learning Outcomes*

The goal of this course is to extend the knowledge the students acquired in course EMBEDDED SYSTEMS I and to introduce advance topics related to embedded system design. More specifically, the students will be taught modern techniques for embedded system design as well as validation, verification and testing techniques for systems comprising of hardware and software. In parallel, the basic compilation techniques for embedded systems as well as code optimization methods will be introduced. As part of the design process of an embedded system, the existing embedded operating systems and their features will be presented. In addition, the students will be taught how to develop hardware accelerators for embedded systems. Finally, the concept of networked embedded systems will be presented, which is the cornerstone of the next generation of systems called IoT/CPS (Internet of Things/Cyberphysical Systems).

Keywords: Embedded systems, embedded software, compilers for embedded systems, optimization techniques, networked embedded systems

Educational Results

After the successful completion of the course, the students will be able to:

At knowledge level:

1. To understand the modern embedded system design techniques as well as methods for validating, verifying and testing the systems under development
2. To evaluate and select the proper optimizations for improving the performance of an embedded system
3. To understand how we compile embedded software
4. To understand the basic principles of embedded operating systems
5. To understand the role of accelerators as part of an embedded system
6. To realize the role of networked embedded systems as part of a set of interconnected devices (IoT)

At skills' level:

1. Familiarize with the embedded system design development frameworks
2. Be trained in programming modern embedded systems using state-of-the art development boards
3. Be trained in using specialized software for designing embedded systems

At capabilities' level:

1. To apply proper design techniques during the design of an embedded system
2. To implement the right optimizations for improving the performance of an embedded system
3. To use effectively compilers and relevant tools for developing embedded software
4. To design accelerators for improving the performance of the final system
5. To solve practical problems that appear during embedded systems design and the design of interconnected embedded systems

General Competences

Taking into consideration the general competences that the degree-holder must acquire (as these appear in the Diploma Supplement and appear below), at which of the following does the course aim?

Search for, analysis and synthesis of data and information, with the use of the necessary technology	*Project planning and management*
Adapting to new situations	*Respect for difference and multiculturalism*
Decision-making	*Respect for the natural environment*
Working independently	*Showing social, professional and ethical responsibility and sensitivity to gender issues*
Team work	*Criticism and self-criticism*
Working in an international environment	*Production of free, creative and inductive thinking*
Working in an interdisciplinary environment	*……*
Production of new research ideas	*Others…*
	…….

- Search, analysis and synthesis of data and information using appropriate tools and technology
- Working individually
- Working in group
- Working in international environment
- Promoting free, creative and inductive thinking
- Promoting new research ideas

(3) SYLLABUS

Program design and design patterns. Program models like data flow graphs and control flow graphs. Introduction to compilation methods. Program optimization in terms of speed, size and power consumption. How do we test programs and verify their correctness? Process abstraction. Context switch between programs. Real time operating systems. Interprocess communication. Performance analysis and power consumption. Hardware accelerators. Architectural patterns. Architectural design: scheduling and allocation. Networked embedded systems. General network architectures and OSI layers. Various networks: I2C, CAN, Ethernet, Myrinet. Design techniques for distributed embedded systems. Embedded systems able to connect to the internet. Quality assurance. Design examples.

The course lectures cover the following thematic areas:

1. **Program design Design patterns** Design patterns for embedded systems. Program models. Data flow graphs. Control flow graphs.
2. **Basic compilation techniques (1)** Assemblers. Linking. Compilation. processes. Data structures. Dead code elimination. Embedding processes in code. Examples.
3. **Basic compilation techniques (2)** Loop transformations. Register allocation. Scheduling. Command selection. Understanding and using the compiler. Interpreters and JIT compilers. Examples.
4. **Optimizations for embedded systems (1)** Analysis and run time optimization: Elements influencing the program performance. Trace driven analysis, Loop optimizations, Cache optimizations. Examples.
5. **Optimizations for embedded systems (2)** Optimizations for increasing speed execution. Analysis and optimization of energy and power. Program power consumption and optimization. Examples.
6. **Program validation and testing** White box testing Black box testing. Evaluation of structural constructs. Performance testing. Design example.
7. **Processes in embedded systems** Multiple tasks and multiple processes. Multiple rate systems. Early multitasking technology: Co-routine. Processes. Context switch: cooperative multitasking, preemptive multitasking. Examples. Process status and scheduling.
8. **Operating systems for embedded systems (1)** Structure of operating system. Interprocess communication. Other operations of operating system Scheduling policies: rate monotonic scheduling, earliest deadline first scheduling, rate monotonic scheduling vs earliest deadline first scheduling.
9. **Operating systems for embedded systems (2)** Interprocess communication mechanisms: signals, shared memory communication, message-based communication. Evaluation of operating systems' performance. Strategies for optimizing power consumption of processes.
10. **Hardware accelerators** CPUs and accelerators. Why do we use accelerators? Designing accelerators.
11. **Designing embedded systems with accelerators** Performance analysis. Partitioning. Scheduling and allocation. System integration and debugging. Design example: video accelerator.
12. **Networks for embedded systems** Distributed embedded architectures. Why distributed? Hardware/software architectures. Programming message passing. I2C bus. CAN bus. Ethernet. Myrinet. Internet.
13. **Designing networked embedded systems** Analysis of communication. Performance analysis. Hardware platform design, partitioning and scheduling. Embedded system able to connect to internet.

(4) TEACHING and LEARNING METHODS - EVALUATION

DELIVERY *Face-to-face, Distance learning, etc.*	Face-to-face in-class lecturing
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY *Use of ICT in teaching, laboratory education, communication with students*	• Slides (ppt) of the presentation of the theoretical part of the course, which will be available from the beginning of semester through e-Class. • Guidelines for the exercises (one per exercise), which will

	be available from the beginning of the semester through e-Class. • Suggested solutions for each exercise will be provided following the completion of each exercise. • Support of teaching procedure through the e-Class platform (notification of the teaching procedure, distribution of slides, supplementary material, announcements, relative links and literature, provision of test and the final examination) • Specialized software relevant to the course.
TEACHING METHODS *The manner and methods of teaching are described in detail.* *Lectures, seminars, laboratory practice, fieldwork, study and analysis of bibliography, tutorials, placements, clinical practice, art workshop, interactive teaching, educational visits, project, essay writing, artistic creativity, etc.* *The student's study hours for each learning activity are given as well as the hours of non-directed study according to the principles of the ECTS*	see table below

Activity	***Semester workload***
Lectures	39
Practical examples and exercises – focusing on problem solving	13
Study of lectures and bibliography	53
Project implementation	20
Course Total	***125 hours (5 ECTS)***

STUDENT PERFORMANCE EVALUATION *Description of the evaluation procedure* *Language of evaluation, methods of evaluation, summative or conclusive, multiple choice questionnaires, short-answer questions, open-ended questions, problem solving, written work, essay/report, oral examination, public presentation, laboratory work, clinical examination of patient, art interpretation, other* *Specifically-defined evaluation criteria are given, and if and where they are accessible to students.*	A. Evaluation of theoretical part: Written exams take place throughout the semester and include: 1. Solving exercises 2. Multiple choice questions 3. Comparative evaluation of theory elements B. Evaluation of exercises/projects Final written exams that includes: 1. Solving exercises 2. Multiple choice questions Comments: • The final grade is the weighted result of the grades of theory and assignments. The weights will be defined and the beginning of each semester and they will be announced via e-Class. • The final exams are in Greek language • The examination process and the evaluation criteria are publicly available to the students through e-Class.

(5) ATTACHED BIBLIOGRAPHY

- Suggested bibliography:

1. Wayne Wolf, «High Performance Embedded Computing”, 2nd edition, Morgan Kaufman, 2014

2. Κωνσταντίνος Καλοβρέκτης, «Βασικές Δομές Ενσωματωμένων Συστημάτων», Εκδόσεις Βαρβαρήγου, 2012
3. Peter Marwedel, «Embedded System Design», Springer, 2011
4. Wayne Wolf, «Οι Υπολογιστές ως Συστατικά Στοιχεία», Εκδόσεις Ελληνικών Τεχνολογιών, 2008

- Related academic journals:

1. ACM Transactions on Embedded Computing Systems
2. ACM Transactions on Design Automation of Electronic Systems