Teaching guide

IDENTIFICATION DETAILS

Degree:	Computer Engineering
Scope	Computer and Systems Engineering
Faculty/School:	Higher Polytechnic School
Course:	COMPUTER ARCHITECTURE AND ORGANISATION
Type:	Compulsory
ECTS credits:	6
Year:	2
Code:	5621
Teaching period:	Third semester
Subject:	Computers and Systems
Module:	Common to the Computer Science Branch
Teaching type:	Classroom-based
Language:	Inglés
Total number of student study hours:	150

Teaching staff	E-mail
Daniel León González	daniel.leon@ufv.es
José Manuel Sánchez Mañes	jm.smanes@ufv.es
Olga Peñalba Rodríguez	o.penalba@ufv.es

SUBJECT DESCRIPTION

The Computer Architecture and Organization course provides a global vision of the structure and operation of a computer and the different components that are part of it: processor, memory system and peripheral devices. In addition, it enables students to develop low-level applications for a given processor and to acquire the ability to

learn new languages and development environments for other processors or microcontrollers.

This course belongs to the Computer Science branch and, within it, to the subject area of Computers and Systems. It is taught in the first semester of the second year of the Computer Engineering degree and requires a minimum of 150 hours of student work.
The course provides an overview of how a computer operates, focusing primarily on the processor and the low-level programming model (assembly language).
It begins with a review of various standards for representing information within the computer, especially numerical data (integers and real numbers), and continues by introducing the different types of modern computing systems: desktops, laptops, servers, mainframes, game consoles, mobile devices, and others.
The course then focuses on the study of the processor, both from the software perspective (i.e., its architecture), describing the features that define different instruction sets, and from the hardware perspective (i.e., its microarchitecture), explaining the main components, how they interact to execute instructions, and the parameters that influence performance.
It goes on to examine the other subsystems of a computer:
- the memory and storage subsystem, emphasizing technological differences between memory types and the fundamentals of hierarchical memory organization.
- the input/output system, with a focus on communication techniques between the processor and peripheral devices.
- interconnection buses, detailing their basic structure and the most commonly used types.
Finally, the course introduces key performance optimization techniques in processing and memory systems, such as cache memory, pipelining, superscalar and VLIW architectures, and multithreaded, multicore, and multiprocessor systems.
The course includes a significant hands-on laboratory component, focused on learning assembly programming for one or more processor architectures.

GOAL

The main goals of this course are to understand how a computer works from a hardware perspective—specifically, which components make up its structure and how they interact—to identify the criteria used to assess system quality and performance, and to grasp the principles of low-level application development for a given processor. These objectives provide the foundation for analyzing and evaluating system performance, as well as for developing the ability to select the most appropriate systems for specific needs.

The specific aims of the subject are:

Analyzing the Architecture of a Von Neumman Computer

Understand the main instruction set architecture (ISA) philosophies and their fundamental characteristics and functions.

Developing of software in assembly language.

Understand the concepts of performance, speedup and their laws

Calculation of the resulting speedup when implementing cache memory, parallelism and pipelining

PRIOR KNOWLEDGE

The main standards for representing information in binary, as well as number base conversions between decimal, binary, and hexadecimal systems.
Two's complement representation (as covered in the course Foundations of Computer Engineering).
Basic characteristics of combinational and sequential logic circuits (as introduced in the course Electronics and Computer Technology).
Fundamentals of algorithm design and coding, (as presented in the syllabi of Introduction to Programming and Object-Oriented Programming).

COURSE SYLLABUS

Topic 1. Introduction.

Basic concepts.
Types of computer systems.
Review of previous knowledge: representation of information.

Topic 2. Instruction Set Architecture (ISA).

Features of the machine instructions.
Addressing modes.
RISC and CISC architectures.
Machine format.
Assembly language.

Topic 3. Basic aspects of computer organization.

Von Neumann and Harvard architectures.
Stages in the execution of an instruction.
Internal structure of the processor: arithmetic-logical unit, registers, control unit.
Types of memory, organization of the memory system, interaction with the processor.
Communication between the processor and the peripherals: input/output system.

Topic 4. System performance.

Parameters for evaluating performance.
Program execution time.
Benchmarks.

Topic 5. Advanced aspects in computer organization.

Advanced aspects of the processor: parallelism at the instruction level, parallelism at the thread level, parallelism at the process level.
Advanced aspects of the memory system: memory hierarchy, cache memory, virtual memory.

Laboratory sessions.

Assembler programming.
Simulation environments for cross-development.
Debugging.
Programming with arithmetic instructions.
Programming of conditional selection statements and loops.
Flags.
Indirect access, data tables and jump tables.
Stack management.
Input/output.
Subroutines.

EDUCATION ACTIVITIES

The subject is structured into three main components:

Theoretical: focused on mastering key concepts in the field of computer architecture and organization.
Practical: aimed at reinforcing and applying theoretical knowledge.
Laboratory: sessions focused on developing practical skills in assembly programming. A project-based learning methodology will be emphasized, with work carried out in groups.

To support these components, expository lessons are combined with practical classes and laboratory work, using participatory methodologies that go beyond traditional content delivery—such as problem-based and collaborative learning—designed to foster student-centered learning and encourage both peer-to-peer and student-instructor interaction.
Non-face-to-face activities, supervised by the instructor through regular classes and/or tutoring sessions, are intended to promote students' autonomous learning skills.
Specifically, the proposed activities and methodologies for the course include:

(AF1) Lectures and discussions: aimed at developing and presenting the theoretical foundations of the subject. A primarily expository approach will be used by the instructor, supported by audiovisual materials, and combined with interactive techniques—especially questioning—as a tool to foster reflection, dialogue, and deeper learning.
(AF2) Problem solving and case studies: sessions focused on applying optimization techniques and procedures in computer optimization. A problem-based learning approach will be used, promoting a collaborative and constructive learning environment, with active student participation and peer interaction at the core of the problem-solving process.
(AF4) Laboratory work with specific technological tools: students will develop an assembly programming project that integrates the knowledge and skills acquired throughout the course. These projects will use development environments that emulate real processors and will be carried out in groups to encourage collaboration.

Face-to-face instruction will be complemented by a substantial amount of autonomous student work, some of which may be completed collaboratively, to promote cooperative learning (AFA1). Planned non-face-to-face activities include:

Individual study and assignments: supporting the internalization of concepts presented in lectures, as well as the application methods practiced in labs and practical sessions.
Group work: primarily intended to prepare for laboratory sessions.

All student work will be supervised and guided by the instructor through individual or group tutoring sessions (AFE1). In some cases, students may be asked to present the conclusions of their work in class, enabling the exchange of knowledge and experiences among peers. Finally, to facilitate access to learning materials, support work planning, and enhance communication between students and the instructor, Canvas, an online learning platform offering various digital resources, will serve as the central hub for academic interaction throughout the course.

DISTRIBUTION OF WORK TIME

TEACHER-LED TRAINING ACTIVITIES	INDIVIDUAL WORK
60 Horas	90 Horas
• AF1 - Lectures and discussions 26h • AF2 - Problem solving and case studies 8h AF4 - Practical classes with specific technological resources 22h • AFE1 - Academic tutoring and assessment activities 4h	• AFA1 - Individual or group work and autonomous study 90h

LEARNING RESULTS

Ability to know, understand and evaluate the structure and architecture of computers, as well as the basic components that make them up.

SPECIFIC LEARNING RESULTS

Students can describe the basic structure of a computer and explain the operation of its main components—processor, memory system, input/output system, and interconnection buses—as well as the flow of information between them.

Students understand the low-level programming model of a processor and are able to develop assembly programs that implement medium-complexity algorithms, debugging them using the manufacturer's development tools and documentation.

Students are able to analyze, compute, and reason about key parameters related to system performance. They can also describe and evaluate the most relevant techniques used for performance optimization, as well as identify major technological and architectural trends in processor design.

Students are able to develop, document, and present topics related to the course, both independently and collaboratively, linking them to content from other subjects within the “Computers and Systems” area of

knowledge, thereby gaining a holistic understanding of the field.

LEARNING APPRAISAL SYSTEM

ORDINARY CALL:

(SE1) 60%: Theoretical-practical written tests.
(SE3) 30%: Assembly programming project, accompanied by an individual assessment to verify the skills acquired in assembly programming.
(SE2) 10%: Daily activities, work and individual and group exercises.

To pass the subject in the ordinary call, ALL of the following conditions must be met:

The SE1 and SE3 marks must be 4 or higher (SE1>=4 and SE3>=4).
The simple average of the SE1 and SE3 marks must be 5 or higher ((SE1+SE3) /2 >=5).
The weighted average of all evaluable aspects of the course must be 5 or higher ((SE1*0.6+SE3*0.3+SE2*0.1) >=5).

Students with Academic Exemption in Ordinary Call:
Students granted academic exemption will be assessed based on the following components:

(SE1-D) 60%: Theoretical-practical written tests.
(SE3-D) 40%: Assembly programming project, accompanied by an individual assessment to verify the skills acquired in assembly programming.

To pass the course in the ordinary call, ALL of the following conditions must be met:

The SE1-D and SE3-D scores must be 4 or higher (SE1-D>=4 and SE3-D>=4).
The weighted average of all evaluable aspects of the course must be 5 or higher ((SE1-D*0.6+SE3-D*0.4) >=5).

For students with academic exemption, both the written tests and the group project will be on the same dates and with the same deadlines as the rest of the students in the ordinary call.

EXTRAORDINARY CALL:
It has the same elements as in an ordinary call with the following considerations:

The mark for participation and submission of exercises (SE2) is non-recoverable; the grade obtained during the ordinary call will be retained for this component.
If SE1>=5 in the ordinary call, this mark will be retanedi for the extraordinary call.
If SE3>=5 in the ordinary call, this mark will be retained for the extraordinary call.

Students with Academic Exemption in Extraordinary Call:
Students who have been granted academic exemption will be evaluated according to the same components described in the ordinary call, with the following considerations:

If SE1-D>=5 in the ordinary call, this mark will be retained for the extraordinary call.
If SE3-D>=5 in the ordinary call, this mark will be retained for the extraordinary call.

Total number of calls: Students have six opportunities to pass this course. All matters related to the evaluation process and the use of exam attempts are governed by the UFV Evaluation Regulations.
Academic Integrity: Any form of fraud or plagiarism by the student in an evaluable activity will be sanctioned in accordance with the UFV Coexistence Regulations. For these purposes, “plagiarism” shall be understood as any attempt to deceive the assessment system—for example, by copying exercises, exams, practical assignments, or any other type of submission from a peer or from unauthorized materials or devices—with the intent of making the teacher believe the work is their own.

ETHICAL AND RESPONSIBLE USE OF ARTIFICIAL INTELLIGENCE

1.- The use of any Artificial Intelligence (AI) system or service shall be determined by the lecturer, and may only be used in the manner and under the conditions indicated by them. In all cases, its use must comply with the following principles:
a) The use of AI systems or services must be accompanied by critical reflection on the part of the student regarding their impact and/or limitations in the development of the assigned task or project.
b) The selection of AI systems or services must be justified, explaining their advantages over other tools or methods of obtaining information. The chosen model and the version of AI used must be described in as much detail as possible.
c) The student must appropriately cite the use of AI systems or services, specifying the parts of the work where they were used and describing the creative process followed. The use of citation formats and usage examples may be consulted on the Library website(https://www.ufv.es/gestion-de-la-informacion_biblioteca/).
d) The results obtained through AI systems or services must always be verified. As the author, the student is responsible for their work and for the legitimacy of the sources used.
2.- In all cases, the use of AI systems or services must always respect the principles of responsible and ethical use upheld by the university, as outlined in the Guide for the Responsible Use of Artificial Intelligence in Studies at UFV. Additionally, the lecturer may request other types of individual commitments from the student when deemed necessary.
3.- Without prejudice to the above, in cases of doubt regarding the ethical and responsible use of any AI system or service, the lecturer may require an oral presentation of any assignment or partial submission. This oral evaluation shall take precedence over any other form of assessment outlined in the Teaching Guide. In this oral defense, the student must demonstrate knowledge of the subject, justify their decisions, and explain the development of their work.

BIBLIOGRAPHY AND OTHER RESOURCES

Basic

Jim Ledin Modern Computer Architecture and Organization 2020
Publisher: Packt. ISBN: 9781838984397

David Patterson & John Hennessy Computer Organization and Design: The Hardware/Software Interface 2011
Publisher: Morgan Kaufmann. ISBN 10: 0123747503 / ISBN 13: 9780123747501

William Stallings Computer Organization and Architecture : Designing for performance 2006
Publisher: Prentice Hall, 7th edition. ISBN: 0131856448, 9780131856448