COURSE DESCRIPTION CARD - SYLLABUS

Course name
Computer System Architecture [N1Inf1>ASK]

Course

Field of study
Computing

Year/Semester
2/4

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
Polish

Form of study
part-time

Requirements
compulsory

Number of hours

Lecture
16

Laboratory classes
24

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

5,00

Coordinators

dr inż. Rafał Klaus
rafal.klaus@put.poznan.pl

Lecturers

Prerequisites

Students starting this module should have basic knowledge regarding of binary systems, boolean algebra, digital circuits and low level programming. Students should be able to solve basic problems in the range of design, checking the correctness and implementing digital circuits and the ability to acquire information from the indicated sources. Students should also understand the necessity to broaden own competences/ be ready to cooperate within the team. In addition, in the field of social competence, the student must present such attitudes as honesty, responsibility, perseverance, cognitive curiosity, creativity, personal culture, respect for other people.

Course objective

Teach students basic knowledge of the construction, principles and practical aspects of operation of computer systems, including microprocessor architectures, memory systems, interrupts, direct memory access systems, communication interfaces, system buses, microcontrollers, parallelism at the level of architecture and efficiency of computer systems. Develop students" ability to solve simple problems of low-level programming of microprocessor systems,programming iterfaces and chipsets, designing, building and testing of simple robots controlled by a microprocessor system. Teach students" the teamwork principles and creative thinking skills through the use of an original training system (Academy of Creative Action).

Course-related learning outcomes

Knowledge:
1. the student has an expanded and deep knowledge in the field of computer system architectures, embedded systems as well as hardware support of operating systems.
2. the student has an expanded and deep knowledge in the field of low-lewel programming languages and hardware aspects of human-computer interfaces.
3. the student knows the basic techniques, methods and tools used in the process of solving IT problems in the field of computer system architecture and implementation of programming languages.

Skills:
1. the student can, by formulating and solving IT tasks, apply properly selected analytical and experimental methods.
2. the student can design, build and test simple programs for microprocessor and embedded systems.
3. the student is able to formulate simple algorithms and implement them using low-level programming languages.
4. the student is able to specify and implement analysers using known tools.

Social competences:
1. the student is aware of the importance of knowledge in solving engineering problems and knows examples and understands the reasons for malfunctioning IT systems that led to serious financial and social losses or to serious health conditions or even to death.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Learning outcomes presented above are verified as follows:
Verification of assumed learning objectives related to lectures:
- evaluation of acquired knowledge on the basis of the written exam (a test includes 8-12 close questions and 3-4 open questions, 50% points needed to pass).
Verification of assumed learning objectives related to laboratory classes:
- evaluation of preparation for individual laboratory sessions (initial test) and assessment of skills related to solving laboratory exercises,
- continuous assessment, during each class (verbal answers),
- assessment of skills related to the project/laboratory task implemented by a team of students (each students" grade is evaluated based on the quality of his/hers part as well as answering to several project related questions),
- assessment of the project implementation report.

Programme content

Lectures cover the following topics:
1. Introduction to the architecture of computer systems
2. Structure of the computer system
3. Interrupt systems and DMA
4. Communication interfaces
5. Processor architecture
6. Cache memory
7. System buses
8. Architecture of multi-core processors
9. Families of popular microcontrollers including the 8051 kernel, the AduC842 microcontroller
10. Arduino and Raspberry Pi systems
11. Assembly language programming, C and Pythton
During laboratory classes, students solve tasks related to:
- learn the software environments and hardware tools
- inspecting the code written in low-level programming language
- searching for errors in hardware projects
- learn the rules of programming seven-segment displays, LCD displays, motor control, hardware resources of microcontrollers, other sensors and actuators
- in team work, students design, build and program an autonomous microprocessor robot and prepare didactic trainings and presentation of their robots for secondary schools (RoboDay event)

Course topics

Lectures cover the following topics:
1. Introduction to the architecture of computer systems
2. Structure of the computer system
3. Interrupt systems
4. Communication interfaces
5. Processor architecture
6. Cache memory
7. System buses
8. Architecture of multi-core processors
9. Families of popular microcontrollers including the 8051 kernel, the AduC842 microcontroller
10. Arduino and Raspberry Pi systems
11. Assembly language programming,

During laboratory classes, students solve tasks related to:
- learn the software environments and hardware tools
- inspecting the code written in low-level programming language
- searching for errors in hardware projects
- learn the rules of programming seven-segment displays, LCD displays, motor control, hardware resources of microcontrollers, other sensors and actuators
- in team work, students design, build and program an autonomous microprocessor robot and prepare didactic trainings and presentation of their robots for secondary schools (RoboDay event)

Teaching methods

1. lecture: multimedia presentation, presentation illustrated with examples shown on the blackboard
2. laboratory classes: solving tasks, discussion, multimedia presentation, projects demonstration

Bibliography

Basic
1. Organizacja i architektura systemu komputerowego, W. Stallings, WNT, Warszawa, 2004
2. Struktura organizacyjna i architektura systemów komputerowych, L. Null, J. Lobur, Helion, Gliwice, 2004
3. Anatomia PC, P. Metzger, Helion, Gliwice, 2007

Additional
1. Architektura komputerów, J. Biernat, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2005
2. Computer Organization and Design, D. Patterson, J. Hennessy, Morgan Kaufmann, 2008
3. Klaus R., Szymaniak P.: “Prototypowanie 3D robota pirotechnicznego”, Mechanika z.103 nr 351/2014, Zeszyty Naukowe Politechnika Opolska Opole 2014; ISBN 978-83-64056-49-9
4. Klaus R. Agilecoach na Wydziale Informatyki Politechniki Poznańskiej, http://biuletyn.pti.org.pl/BiuletynPTI_2016-04.pdf
5. Klaus R. RoboDay na Wydziale Informatyki Politechniki Poznańskiej, http://biuletyn.pti.org.pl/BiuletynPTI-2016-03.pdf

Breakdown of average student's workload

	Hours	ECTS
Total workload	125	5,00
Classes requiring direct contact with the teacher	42	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	83	3,00