Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Department of Computer Engineering

Computer architecture 2 "Processors" Working program of the academic discipline (Syllabus)

Details of the academic discipline

Level of higher education	***First (bachelor's degree)***
Branch of knowledge	*12 Information technologies*
Specialty	*123 Computer engineering*
Educational program	*Computer Engineering*
Discipline status	*Normative*
Form of education	*intramural(daytime)*
Year of education, semester	*3rd year, autumn*
Scope of the discipline	*4.5 credits, 135 hours,* *Lectures 18 (36 hours), Laboratory 9 (18 hours)*
Semester control/ control measures	*Examination*
Lessons schedule	*According to the schedule for the spring semester of the current academic year at http://rozklad.kpi.ua*
Language of teaching	*Ukrainian*
Information about head of the course / teachers	Lecturer: Dr.Sc., associate prof., Klymenko Iryna Anatoliivna, ikliryna@gmail.com Laboratory: ass. Kaplunov Artem Volodymyrovych, art.kaplunov@gmail.com
Placement of the course	Lecture material: https://bbb.comsys.kpi.ua/b/iry-ped-qe9

Program of the Academic Discipline

1. Description of the educational discipline, its purpose, subject of study and learning outcomes

The purpose of the Computer Architecture discipline is to study the theoretical foundations, functional capabilities, and principles of interaction of the main computer components at the functional level. The subject of studying the credit module Computer Architecture 2 "Processors" is the architecture and functionality of modern processor cores and other computer hardware components; input/output systems; means and mechanisms for forwarding and storing data at the functional level of a computer; the organization of the architecture-dependent level of the operating system, familiarity with low-level languages and the basics of programming for processor cores, the modern element base and the practical basics of modern hardware design processes.

Hands-on experience with building, set-up and using of Linux-like operating systems on AVR processors; building executable open-source programs; basic low-level languages programming skills, in particular C, for AVR processor core.

Studying the discipline provides the following general and professional competencies:

- GC2. Ability to learn and master modern knowledge.
- PC1. Ability to apply legislative and regulatory frameworks, as well as national and international requirements, practices and standards, to implementation of professional activities in Computer Engineering field.

- PC5. Ability to use automation design tools and systems for the development of components of Computer Systems and Networks, Internet applications, Cyber-Physical Systems also.
- PC7. Ability to use and implement new technologies, including Smart, Mobile, Green and Secure Computing Technologies, to take part in the modernization and reconstruction of Computer Systems and Networks, a variety of Embedded and Distributed applications, in particular with the aim of increasing their efficiency.
- PC13. Ability to solve problems in the field of Computer and Information Technology, to determine the limitations of these Technologies.
- PC14. Ability to Design Systems and their Components taking into account all aspects of their life cycle and task, including creation, configuration, operation, maintaining and disposal.
- PC16. Ability to design, implement and administer High Performance parallel and distributed computer systems and their components using FPGA modules and Systems of Automated Design.

In accordance with aforecited, students will receive the following program learning outcomes:

- PLO1. Know and understand the scientific principles that underpin the functioning of Computer, Systems and Networks.
- PLO3. Know the latest technologies in Computer Science Engineering.
- PLO4. Know and understand the impact of Technical Solutions in a social, economic, social and environmental context.
- PLO5. Have knowledge of the basics of Economics and Project Management.
- PLO7. Be able to solve speciality-specific problems of methods of analysis and synthesis.
- PLO10. Be able to develop Software for Embedded and Distributed Applications, Mobile and Hybrid Systems; evaluate, operate specialty-specific equipment.
- PLO11. Be able to search for information in various sources to solve problems of Computer Engineering.
- PLO12. Be able to work effectively both individually and in a team.
- PLO13. Be able to identify, classify and describe the operation of Computer Systems and their components.
- PLO14. Be able to combine Theory and Practice, as well as make decisions and develop a Strategy for solving speciality-specific problems, taking into account Universal Values, Social, State and Industrial Interests.
- PLO15. Be able to perform experimental research on professional topics.
- PLO19. Ability to adapt to new situations, justify, make and implement relevant decisions.
- PLO22. Determine parameters of individual blocks of Computers, Computer Systems, Computer Networks.
- PLO24. Build, configure, and use Linux-like operating systems.

2. Pre-requisites and post-requisites of the discipline (place in the structural and logical scheme of training according to the relevant educational program)

When studying the discipline «Computer Architecture. Part 2. Processors» it is advisable to use the knowledge obtained during the study of previous disciplines: «Computer Logic. Part 1», «Computer Logic. Part 2. Computer Arithmetic», «Computer Electronics», «Computer architecture 1. Arithmetic and control devices», «System programming».

The discipline is basic for the courses: «Computer architecture. Part 3. Microprocessor devices», «Computer Architecture. Course work», «Computer systems»; courses of elective disciplines of F-catalogue, «Signal processing systems», «Testing and quality control (QA) of embedded systems», «C/Embedded programming technologies», «FPGA programming technologies», «Designing technologies of intellectual systems».

3. Content of the academic discipline

Introduction

Chapter 1. Introduction to the architecture of modern processors. Features of the architectural and functional organization of modern RISC and CISC processors.

Topic 1.1. Modern classification of processors by functional purpose. Universal and specialized processors. Overview of the modern element base. Systems on a crystal (SoC). Multi-core processors.

Topic 1.2. Von-Neumann architecture in modern processor cores. The AVR architecture is the flagship of the modern elemental base for processor cores aimed at installing universal operating systems.

Chapter 2. Introduction to the practical part of the course "Computer architecture 2. Processors".

Topic 2.1. Organization of the architecture-dependent level of the operating system in universal and specialized computers, controllers, microcontrollers.

Topic 2.2. Working in the Linux command line. An introduction to Git version control. Creating a project on GitLab.

Topic 2.3. Features of assembly, debugging and use of Linux-type operating systems on AVRm7 processors with Von Neumann architecture for implementation of embedded systems and computers.

Topic 2.4. Compilation and cross-compilation. Tools for cross-compilation (Toolchains). Collection of bootloaders (u-boot), kernel (Kernel), file system (RootFs) executables for AVRv7 processor.

Topic 2.5. Stages of deployment of the operating system. Architecture initialization processes in the kernel space of the operating system on the example of the architecture of x86 and AVRv7 processors.

Topic 2.6. Processor firmware. Implementation of a separate console using the UART protocol.

Chapter 3. Memory architecture. The issue of the interaction of the processor and memory during the execution of commands

Topic 3.1. Team management. Conveyors. Command system. Formats of commands, methods of addressing commands and operands in processors with different architectures. Addressing of data and transitions. Peculiarities of executing commands of various classes and working with routines in universal and specialized processors.

Topic 3.2. General issues of multi-level computer memory construction. Super-random register memory, random access memory, buffer (cache) memory. Interaction of all levels of memory among themselves.

Topic 3.3. Organization of cache memory in modern computers.

Topic 3.4. Organization of virtual memory. Memory management unit (MMU, Memory Management Unit).

Access to a common resource. Protection of memory partitions.

Topic 3.5. Management of data processing at the program level. Multi-program mode of operation of the processor.

Topic 3.6. Direct memory access (DMA, direct memory access).

Topic 3.7. Interfaces of storage devices for computers with different organization of the system bus.

Chapter 4. Architecture of the input/output system

Topic 4.1. General questions about the operation of computers with external devices. Organization of memory space and I/O.

Topic 4.2. Interfaces of external devices in the program mode of the readiness survey

Topic 4.3. Construction of interfaces of external devices (ZP). Interfaces in systems with modular organization.

Topic 4.4. Organization of the system of interruptions. Processing of interruptions at different levels (software, firmware, hardware)

Chapter 5. Directions of processor architecture development in the conditions of the 4th industrial revolution (Industry 4.0/IoT)

Topic 5.1. Development directions of processor architecture in the context of Industry 4.0 / IoT.

Topic 5.2. Using co-processors to increase computing performance. Using FPGA to implement reconfigurable modules of processor cores.
Topic 5.3. Overview of the RISC-V processors, Raspberry Pi.

4. Educational resources and materials

4.1. Basic literature

1. Computer Architecture 2. Processors: Theory and Practicum [Electronic resource]: tutorial for bachelor's study degree applicants under the educational program "Computer systems and networks" specialty 123 "Computer engineering" / I. A. Klymenko, A. V. Kaplunov, V. A. Taraniuk, V. V. Tkachenko; Igor Sikorsky KPI. – Electronic text data (1 file: XX MB). – Kyiv: Igor Sikorsky KPI, 2022. – 50 p.
2. Sergiyenko A.M. Computer Architecture (Архітектура комп'ютерів: підручник англійською мовою). - КПІ ім. Ігоря Сікорського, 2022. – 395 с.

4.2. Additional literature

1. Testing and quality control (QA) of embedded systems [Electronic resource]: tutorial for bachelor's study degree applicants under the educational program "Computer systems and networks" specialty 123 "Computer engineering" / I. A. Klymenko,V. A. Taraniuk, V. V. Tkachenko, O.O. Pysarchuk; Igor Sikorsky KPI. – Electronic text data (1 file: XX MB). – Kyiv: Igor Sikorsky KPI, 2022. – 58 p.
2. Sarah L. Harris, David Harris Digital Design and Computer Architecture / Elsevier Science & Technology, 2022. – 720 p.
3. Hamacher C., Vranesic Z., Zaky S. COMPUTER ORGANIZATION: 5th edition. – 2022. [Електронний ресурс] Hamacher:ComputerOrganization (mhhe.com), Computer Organization By Carl Hamacher 5th Edition | lulabi.live.
4. Tanenbaum A.S. Structured Computer Organization: 6th Edition – 2013. - [Електронний ресурс]
5. Mark A. Yoder, Jason Kridner, BeagleBone Cookbook: Software and Hardware Problems and Solutions / O'Reilly Media, 2015. – 346 p.
6. BeagleBone Black [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://beagleboard.org/black.
7. AM335x and AMIC110 Sitara™ Processors Technical Reference Manual (Rev. Q) [Електронний ресурс] // Texas Instruments. – 2022. – Режим доступу до ресурсу: https://www.ti.com/lit/ug/spruh73q/spruh73q.pdf?ts=1660308115041&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FAM3357.
8. Клименко І.А. Класифікація та архітектурні особливості програмованих мультипроцесорних систем-на-кристалі // Проблеми інформатизації та управління: Зб.наук.пр.– К.: Вид-во нац. авіац. ун-ту «НАУ-друк», 2012.– Вип.1(36). – С 90 – 103.
9. Клименко І.А. Тенденції застосування сучасної елементної бази для побудови високопродуктивних обчислювальних систем // Проблеми інформатизації та управління: Зб.наук.пр. – К.: Вид-во нац. авіац. ун-ту «НАУ-друк», 2010.– Вип.1(29). – С 90 – 103.

4.3. Information resources

1. Course of video lectures – https://bbb.comsys.kpi.ua/b/iry-ped-qe9

Educational content

5. Methodology of mastering an educational discipline (educational component)

Distribution of study time by types of classes and tasks in the discipline according to the working study plan. 135 hours and 4.5 credits are allocated to the credit module.

To achieve the goal of the educational discipline, the lecture material should focus on the features of the construction of the functional level of the computer, processor and other components. Particular attention should be paid to the features of computer design using a modern element base.

The purpose of the laboratory work is to acquire the skills and abilities to apply the principles of programming for modern processors and learning the architecture of their individual functional units in practice. To perform laboratory work, emulators of Linux-oriented architectures (CEMU ARM), microcontroller programming tools (vim, Eclipse, Cube IDE), hardware (Raspbery Pi, BeagleBone Black).

Topics of laboratory works:

Laboratory work 1. Introduction to Git version control system. Creating a project in Gitea KPI for laboratory works implementation.
Laboratory work 2. Introduction to the Kbuild Linux system for building projects from source codes. Compiling executable files for Linux OS on ARM processor architecture.
Laboratory work 3. Linux Kernel deployment in Qemu software emulator environment for ARM CORTEX A processor architecture.
Laboratory work 4. Creating a remote console using the UART interface. BeagleBone Black board firmware installation.
Laboratory work 5. Development of kernel modules on C.
Laboratory work 6. Development of kernel modules. Device Tree research.

6. Independent work of a student of full-time higher education

Types of independent work for students of full-time education (81 hours):
– preparation for auditorium classes, work on current homework and study of lecture materials (1 hour x 18 lectures = 18 hours);
– implementation of individual tasks for laboratory works, exercise solving, protocol formatting, preparation and processing of calculations based on primary data obtained in practical classes, formatting of laboratory work report (4 hours x 6 laboratory works = 24 hours);
– preparation and completion of module test (8 hours);
– preparation for the examination (9 hours);
– topics for self-study (11 topics x 2 hours = 22 hours).

Теми на самостійне опрацювання (денна форма навчання)

Chapter 1. Introduction to the architecture of modern processors. Features of the architectural and functional organization of modern RISC and CISC processors.
Topic 1.1. Overview of multi-core and multi-processor computer technologies. Specifics of command execution on multi-core processor cores. Multi-core in RISC ARM processors. (2 hours).
Chapter 2. Introduction to the practical part of the course "Computer architecture 2. Processors".
Topic 2.2. Commands for working in the Linux command line (2 hours).
Topic 2.3. Introduction to Git version control system (2 hours).
Chapter 3. Memory architecture. The issue of the interaction of the processor and memory during the execution of commands
Topic 3.1. Concept of a conveyor on processor core level. Command execution on processors of various architecture. Addressing of data and transitions (2 hours).
Topic 3.3. Cache memory organization in modern computers (2 hours).
Topic 3.4. Memory Management Unit, MMU. Memory partitions protection (2 hours).
Topic 3.5. Multi-program processor operation mode (2 hours).
Topic 3.6. Direct memory access, DMA (2 hours).
Chapter 5. Directions of processor architecture development in the conditions of the 4th industrial revolution (Industry 4.0/IoT)
Topic 5.1. Development directions of processor architecture in the context of Industry 4.0 / IoT (2 hours).
Topic 5.3. Overview of RISC-V processor, Raspberry Pi. (2 hours).

Policy and Control

7. Policy of academic discipline (educational component)

Deadlines are set for the completion of laboratory work and modular control work.

Completion of laboratory work outside of established deadlines is accompanied by penalty points, which are deducted from the grade for the protocol. Module test is not accepted outside of established deadlines.

Module Test is carried out independently according to an individual task.

Each laboratory work is preceded by the completion of an individual task and its preparation in the form of a protocol. A student who came to class without a formatted protocol is not allowed to do laboratory work. At the first stage, a student defends the results received while working on the individual task for the laboratory work, at the second stage – defends theoretical knowledge by the means of oral questioning or testing. The majority of laboratory works are accompanied with test for evaluation of theoretical and practical material studied for the laboratory work. Points received for the completion of a laboratory work, for the test and protocol are included in the score for the laboratory work. Testing is conducted during the laboratory class after checking the results of completing laboratory works. A student who did not complete the individual task is not allowed to the test and laboratory work.

Individual lecture topics are accompanied by short express tests (for 15 minutes), which include the material of the studied topic and questions that are asked for independent study. The points obtained for the test are included in the semester rating. Recurring tests are not retaken.

The module test is conducted during the auditorium class without the usage of auxiliary means (mobile phones, tablets etc.); the result is sent to the corresponding directory of Google Drive through a Google Form.

The module test is not rewritten in case of a negative mark, a negative mark on the Module Test (less than 9 points (<60%)) is equaled to 0 points, in which case the Module Test is not passed.

Completion of laboratory works is mandatory for admission to the semester exam. The condition for admission to the exam is the completion of all laboratory work and a starting rating of at least 30 points.

The points for the exam work are added to the points for the laboratory work and Module Test and make up the semester rating.

Bonus points are awarded for: active participation in lectures; completing recurrent homework, keeping lecture notes, preparing a presentation on one of the self-study topics of the discipline, etc. Maximum available bonus points are no more than 10.

Penalty points are issued for: untimely submission of laboratory work.

8. Types of control and rating system for evaluating learning outcomes (RSE)

The student's semester rating for the discipline is calculated based on a 100-point scale. The semester rating consists of the starting (current) grade R_S and examination R_E. Starting rating consists of points which a student receives for completing of 6 laboratory works R_L, module test R_{MT} and points for express-tests and bonus points.

The maximum number of points for laboratory works is 40 points, i.e., $R_L = 40$.

The criteria for evaluating laboratory works are as follows:

– timeliness of preparation of the protocol for the laboratory session, completeness of the theoretical or practical task in the protocol, the protocol is posted on GitLab on time: 0 - 2 points (penalty points can be deducted from the grade for late delivery of the protocol 0-2 points);

– correct functioning of the developed models on software or hardware, demonstration of own repository on GitLab with laboratory work materials and presence of commits: 0 - 2 points;

– a survey/testing on the subject of laboratory work for accepting the practical part of the work, defense of the results obtained in the work, answers to additional theoretical questions of the teacher, completeness of the report/protocol on the work on GitLab: 0 – 3(4).

The maximum number of points for Module Test R_{MT} = 15 points.

The criteria for evaluating Module Test at four levels:

- correct and meaningful answer with explanations in the terms of the subject area: 13 - 15 points;
- correct answer, incomplete explanations: 11 - 12 points;
- the answer contains mistakes: 9 - 10 points;
- the answer contains significant mistakes, there are no explanations: 4-8 points;
- no answer: 0 points.

The score for MKR is reduced by:

- incorrect design of schemes and drawings, lack of digitization of registers, buses, non-compliance with GOST standards;
- lack of comments in the program code and design of algorithms;
- absence of comments and explanations during calculations.

The details of the points for the current works for the semester are given in the following table.

The name of the class	**Form of control**	**Scores**	**Admission to the exam by automatic evaluation**	**Total points**
Laboratory work 1	Protocol on GitLab	2	5	7
	Task completion	2		
	Survey/test (Linux introduction, Git)	3		
Laboratory work 2	Protocol on GitLab	2	4	7
	Task completion	2		
	Survey/test	3		
Laboratory work 3	Protocol on GitLab	2	4	7
	Task completion	2		
	Survey/test	3		
Laboratory work 4	Protocol on GitLab	2	4	7
	Task completion	2		
	Survey/test	3		
Laboratory work 5	Task completion	2	2	4
	Protocol on GitLab	2		
Laboratory work 6	Task completion	2	5	8
	Protocol on GitLab	2		
	Survey/test on laboratory works 5 - 6	4		
Module Test	MT2	15	9	15
Express-tests		5		5
Total points		60	30	60

The maximum number of points for the exam is **R_E = 40 points**.

The examination ticket contains 4 tasks (one theoretical and three practical) on the subject of lectures and laboratory work performed during the semester. Each question is evaluated from 0 to 10 points.

The criteria for evaluating each question at four levels:

- correct and meaningful answer: 9 - 10 points;
- correct answer, incomplete explanations: 7 - 8 points;
- the answer contains mistakes: 5 - 6 points;
- no answer or the answer is incorrect: 0 points.

Calendar certification of students (for 8 and 14 weeks of semesters) in the discipline is carried out according to the value of the student's current rating at the time of certification. If the value of this rating is at least 50% of the maximum possible at the time of certification, the student is considered certified. Otherwise, the attestation information is marked as "uncertified".

A necessary condition for a student's admission to the exam is the completion and defense of all laboratory work with a total of at least 30 points.

The number of points a student receives per semester is determined by the formula

$$R = (R_L + R_{MT}) + R_E = R_S + R_E$$

The maximum number of points per semester does not exceed RS = 100.

Taking into account the received sum of points, the final grade is determined by the following table:

Scores	*Rating*
100-95	Excellent
94-85	Very good
84-75	Good
74-65	Satisfactorily
64-60	Enough
Less than 60	Unsatisfactorily
Admission conditions not met	Not allowed

Working program of the academic discipline (syllabus):
Made by, Ph.D., docent, professor of the department of CE Klymenko Iryna Anatoliivna.
Approved by the Department of Computing Engineering (Protocol №10 dated 25.05.2022).
Agreed by the methodical commission of FICT (protocol №10 dated 09.06.2022).