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

Computer Engineering Department

Computer aided design of computer systems

(Syllabus)

Requisites of the Course

Cycle of Higher Education	***First cycle of higher education (Bachelor's degree)***
Field of Study	*12 Information Technologies*
Speciality	*123 Hardware engineering*
Education Program	*Computer Systems and Networks*
Type of Course	*Selective*
Mode of Studies	*full-time*
Year of studies, semester	*4 year (8 semester)*
ECTS workload	*4 credits (ECTS) .120 hours*
Testing and assessment	*Test, MCW*
Course Schedule	*http://rozklad.kpi.ua/*
Language of Instruction	*English*
Course Instructors	Lecturer: *D.Sc., Seniour Scientist, Anatoliy Sergiyenko, mobile +380688123376, email anat.srg@gmail.com, personal web page http://kanyevsky.kpi.ua* Teacher of practical work: *Assistant Anastasiia Anatoliivna Sergiyenko*
Access to the course	*Lessons: https://bbb.comsys.kpi.ua/b/ana-22m-tre* *Text books: https://kanyevsky.kpi.ua/студентам/*

Outline of the Course

1. Course description, goals, objectives, and learning outcomes

Teaching the basics of computer-aided design (CAD) of computer systems at the university has the following aspects:

Academic aspect. CAD is a science that provides applied knowledge. The goal is to acquaint students with the basic concepts and methods of automated design of modern complex technical systems.

Professional aspect. Today, the set of microprocessor architectures implemented on an integral circuit (IC) is rapidly expanding. Field programmable gate arrays (FPGAs) are becoming an integral attribute of embedded computing technology. There is a growing need for design engineers who are proficient in designing systems on a chip (SoC). They are required to be able to describe devices with a given functionality in VHDL or Verilog, including a microprocessor, reconfigurable intelectual property (IP) cores according to project requirements, connect modules to a SoC structure, and program the firmware support. The purpose of this course is to develop skills in using the VHDL language to describe hardware projects of medium complexity that are implemented in FPGAs. The basis of laboratory work is the design of microprocessor IP cores using the VHDL language and appropriate CAD tools, including a VHDL simulator and a logic circuit compiler-synthesizer. A student who performs a course of laboratory work studies not only the basics of microprocessor circuitry, but also the modern technology of their design for implementation in FPGAs.

Intellectual and educational aspects. Studying the basics of IC design automation contributes to the development of cognitive skills, which is an essential intellectual factor in the process of creating both hardware and software.

As a result of studying this discipline, the following learning outcomes are achieved.

Competencies:

•ZK1 - ability to abstract thinking, analysis and synthesis;

• ZK2 - ability to learn and acquire modern knowledge;

• ZK3 - ability to apply knowledge in practical situations;

• ZK7 - ability to identify, pose and solve problems.

• FK2 - ability to use modern methods and programming languages for development of algorithms and software.

• FK5 - ability to use the tools and systems of design automation to development of components of computer systems and networks, Internet applications, cyber-physical systems, etc.

• FK6 - ability to design, implement and maintain computer systems and networks of various types and purposes.

• FK7 - ability to use and implement new technologies, including technologies of smart, mobile, green and secure computing, participate in the modernization and reconstruction of computer systems and networks, various embedded and distributed applications, in particular with the aim of increasing their efficiency.

• FK13 - Ability to solve problems in the field of computer and information technologies, determine the limitations of these technologies.

• FK14 - ability to design systems and their components taking into account all their aspects. life cycle and task, including creation, configuration, operation, maintenance and disposal.

• FK15 - ability to argue the choice of methods for solving the specific problems, critically evaluate the obtained results, justify and defend the accepted decisions.

• FK16 - ability to design, implement and maintain the high-performance parallel and distributed computer systems and their components using FPGAs, modules and CAD systems.

Knowledge and skills:

• PRH 1 - know and understand the scientific principles that underlie the functioning of computer tools, systems and networks.

• PRH 2 - have the skills of performing experiments, collecting data and modeling in computer systems.

• PRH 7 - to be able to solve problems of analysis and synthesis of devices which are relevant to the specialty.

• PRH - To be able to think systematically and apply creative abilities to the formation of new ideas.

• PRH 10 - to be able to develop software for embedded and distributed applications, mobile and hybrid systems, to design, and operate the typical application specific hardware.

• PRH 11 - to be able to search for information in various sources to solve the computer engineering problems.

• PRH 15 - to be able to perform experimental research on a professional topic.

• PRH 16 - to be able to evaluate the obtained results and defend the accepted decisions with arguments.

2. Prerequisites and post-requisites of the course (the place of the course in the scheme of studies in accordance with curriculum)

Prerequisites, i.e. disciplines, the study of which must precede the study of this discipline:

- Physics.
- Mathematical analysis.
- Linear algebra.
- Discrete Math.
- Fundamentals of electrical engineering.
- Programming.
- Data structures and algorithms.
- Computer logic.
- Computer electronics.
- Software engineering.
- Computer modeling.
- Digital signal processing.

Post-requisites, i.e. disciplines, the study of which must be preceded by the study of this discipline:

- hybrid computer systems;
- basics of computer design;
- reliability of computer systems;
- diploma design.

3. Content of the course

Theme 1: Flow of designing hardware using VHDL

Features of circuit design technology using VHDL. Design of SoC. The process of developing a computing devices. The complexity of SoC projects and the possibilities of design technology. Technology of IP cores.

Computer models for VHDL. Basics of VHDL. Representation of algorithms using VHDL. Synthesis of the logical scheme. Main features of VHDL. Sequential operators in VHDL. Parallel operators in VHDL. Procedures and functions. Packages. Attributes.

Topic 2: Technology of developing SoC.

The process of developing SoC on FPGA. The complexity of SoC projects and the possibilities of design technology. IP Cores technology. Interoperable SoC hardware and software design. SoC system synthesis. FPGA as an element base of modern electronics.

The structure and architecture of FPGA. IEEE library for designing devices. Representation of algorithms on VHDL. General requirements for designing SoC. VHDL constructions for synthesis. Standard VHDL packages for synthesis. Designing combination schemes. Design of arithmetic devices. Designing schemes with memory. Design of control circuits. Management of synthesis. Memory programming.

Topic 3: Design of pipelined computers.

Peculiarities of pipeline data processing. Mapping the algorithm to the hardware. Periodic algorithms. Graph of synchronous data flows (SDF). Signal graph. Classification of data flow graphs.

Designing RISC processors. Design of specialized computers. Methodology for the synthesis of a conveyor calculator. Optimization of the structure. Conveyor execution of the algorithm. Speedway of the SDF. Transformation of the graph of the algorithm into an acyclic SDF. Algorithm graph optimization. Perspectives of integrated technology. Prospects of FPGA.

4. Educational materials

Basic literature

1. Sergiyenko A.M., Korneychuk V.I. Digital Networks Design – К.: «Корнійчук», 2007. –80 p. http://kanyevsky.kpi.ua/wp-content/uploads/2017/11/Sergiyenko_DND.pdf

2. Sergiyenko A.M. Computer architecture. Part 1. К.:NTUU "KPI". 2017. 100 p. http://kanyevsky.kpi.ua/wp-content/uploads/2017/09/Apx_comp_eng.pdf

3. Sergiyenko A.M**.** Computer-aided design of computer systems. Laboratorium works.К.: Igor Sikorsky KPI. 2022. 20 p.

Additional text books

1. Ashenden P.J., Lewis J. The designer's guide to VHDL. 3-d Ed. Morgan Kaufman. 909 p. https://www.academia.edu/9455362/The_Designers_Guide_to_VHDL

2. Harris S., Harris D. Digital Design and Computer Architecture: RISC-V Edition. Morgan Kaufmann, 2021. 565 p. https://microelectronica.pro/wp-content/uploads/books/digital-design-and-computer-architecture-russian-translation.pdf

3. Khan S.A. Digital Design of Signal Processing Systems. A practical Approach. Wiley, 2011. 608 p. https://www.academia.edu/34580646/A_Practical_Approach_DIGITAL_DESIGN_OF_SIGNAL_PROCESSING_SYSTEMS_DIGITAL_DESIGN_OF_SIGNAL_PROCESSING_SYSTEMS_A_Practical_Approach

Lesson slides in https://kanyevsky.kpi.ua/ студентам/презентації-лекційного-матеріалу-з-в /

Equipment needed for classes.

Lecture classes are held in a classroom equipped with a projector, practical classes are held in a computer classroom. The student version of ActiveHDL CAD is used for laboratory work.

Educational content

5. Methodology

The educational content of the discipline consists of lessons and laboratory exercises.

Lessons:

Theme 1: Flow of designing hardware using VHDL

Lesson 1. Introduction.

A new golden era of computer architectures. Textbooks, presentations of lecture material, laboratory works. Rating system. Features of circuit design technology using VHDL. Comparison of schematic and model design. CAD with VHDL. Design of systems on a chip (SoC). The process of developing a computing device. Labor productivity of the SoC developer. The complexity of SoC projects and the possibilities of design technology. Technology of IP cores.

Lesson 2. Computer models in VHDL.

Basics of VHDL. Data types supporting hardware description. Operators, attributes supporting system behavior over time. Computational model for VHDL implementation. VHDL simulator

architecture. Hardware model for VHDL implementation. Representation of algorithms using VHDL. Synthesis of the logical network.

Lesson 3. Main features of VHDL. Project object and architecture. Objects, types and expressions of the VHDL language. Entity as the interface description. Architecture as a description of the model behavior. Behavioral model of SoC. Data types in VHDL. Type Standard Logic from package 1164. Signal, variable, constant.

Lesson 4. Sequential operators in VHDL.

Process and sequential operators. Sensitivity list. VHDL language operations. Expressions of the VHDL language. Signal assignment operator. Operators waiting for the wait event. Variable assignment operator. Conditional operators. The selection operator. The loop operator. Procedure call and return from it. Assert and Report operators.

Lesson 5. Parallel operators in VHDL.

Computational model for VHDL implementation. Parallel operators in VHDL. Process operator. Sensitivity list. Signal assignment. Parallel procedure call. Inserting a component instance. The GENERATE statement. BLOCK operator. ASSERT parallel operator and testbench.

Lesson 6. Procedures and functions. Packages.

Subroutines. Function call. Procedure call. Package. Subroutines. Type conversion. Computational model for VHDL implementation. Program structure. Entity and Architecture. Configuration announcement.

Lesson 7. Attributes.

Attributes for a scalar type. Attributes for a regular type. Attributes of signals. User attributes. Aliases. Labels in the program.

Theme 2: Technology of SoC development.

Lesson 8. Technology of developing SoC.

The process of developing SoC in FPGA. The complexity of SoC projects and the capabilities of design technology. IP Cores technology. Interoperable SoC hardware and software design. SoC system synthesis. FPGA is an element base of modern electronics. Advantages of using FPGA. Examples of effective FPGA application.

Lesson 9. Structure and architecture of FPGA.

Xilinx Virtex FPGA hardware resources. Logical table. Trigger. Configurable logic block. Implementation of multi-input multiplexers. Shared memory. Shift register with adjustable delay. Block memory. Block DSP48. I/O buffers. Synchronous signal distribution network. The block of frequency multiplication and division with digital or phase lock.

Lesson 10. IEEE library for designing devices.

Process and sequential operators. Computational model for VHDL implementation. Hardware model for VHDL implementation. Representation of algorithms in VHDL. Synthesis of the logical network. General requirements for designing SoC. The principle of two-clock synchronization. Designing with asynchronous triggers. The principle of single-clock synchronization. Distortion of single clock synchronization. VHDL constructions for synthesis. Package std_logic_1164. Multiple signal sources and resolution function. Functions of the package std_logic_1164. Packages std_logic_arith, std_logic_signed and std_logic_unsigned. Packages numeric_bit and numeric_std.

Lesson 11. Design of combinational circuits.

Combinational circuit according to the Boolean equation. Selective signal assignment. Conditional signal assignment. Combination network by conditional assignment. Combinational network by subroutine call. Combinational network according to the sequential operator. An example of the development of a combinational network. Using three-state logic. Arbitrary state in Std_Logic. Unpredictable asynchronous triggers. Design of arithmetic devices.

Lesson 12. Designing networks with memory.

Unpredictable asynchronous triggers. The principle of single-cycle synchronization. Distortion of single clock synchronization. Synthesis of networks with memory. Registers in behavioral VHDL. Rising/falling functions. The WAIT statement. Variables in process. Process with register. Inserting a register type component. An example of designing a network with memory.

Lesson 13. Project management and project management control.

The concept of operating and controlling automata. Moore's FSM. Miley's FSM. An example of an automaton design. Ways of coding states of the automaton. Examples of automaton design. An example of designing a counter on a shift register. Design of complex FSMs. Microprogram automaton. An example of designing a special function calculation block. Design management. Synthesis control attributes. Placement Constraint Attributes. Initialization attributes. Time constraints.

Lesson 14. Memory programming

Shared memory. Shift register with adjustable delay. Block memory. FIFO. Memory with correction codes. Memory with increased capacity. Distributed memory programming. Programming BlockRAM. RAM initialization. Programming of non-volatile memory.

Topic 3: Design of pipelined computers.

Lecture 15. Periodic algorithms and pipelined computers.

Acceleration of calculations using data processing on FPGA. Sequential and parallel data processing. FPGA-based system. Peculiarities of pipeline data processing. Mapping the algorithm to the hardware. Periodic algorithms. The algorithm given by the cycle. Periodic algorithm graph. Mapping the graph of the periodic algorithm. Graph of synchronous data flows (SDF). Heterogeneous and homogeneous SDFs. Reduced graph of the algorithm. Signal graph. Classification of data flow graphs.

Lecture 16. Design of RISC processors

Basics of RISC processors. Pipelining. Register memory. Executing instruction while processing a branch. Simplified addressing. Simple instruction format. Microprocessor core from laboratory works. An example of the synthesis of a RISC processor. An example of the synthesis of a stacked RISC processor.

Lecture 17. Design of application specific computers

Methodology for the synthesis of a pipelined processor. Optimization of the structure. An example of the synthesis of a block for calculating a special function. Pipelined execution of the algorithm. Speed parameters of SDF. Transformation of the graph of the algorithm into an acyclic SDF. Algorithm graph optimization. Retiming. Folding SDF. Unfolding SDF.

Lecture 18. Prospects for the implementation of FPGA. Modular control work.

Programming languages for hardware description. Perspectives of integrated technology. Prospects of FPGA.

Laboratory works

Laboratory work 1.

Arithmetic and logical device.
Laboratory work 2.
Command counter
Laboratory work 3.
A non-volatile memory device.
Laboratory work 4.
Register memory
Laboratory work 5.
Special function calculation block.
Laboratory work 6.
Arithmetic device.
Laboratory work 7.
Microprocessor core.

6. Self-study of the student

The self-study includes the independent work of students and is as follows:

• preparation for lectures by studying the previous lecture material as well as literary sources on which the material of previous lectures is based (the list of sources and the list of sections is provided together with the lecture material);

• preparation for the laboratory exercise by getting acquainted with the task and guidelines for laboratory work, including the study of theoretical material needed to answer control questions for laboratory work;

Each laboratory exercise is intended to be prepared and executed for two weeks.

Policy and Assessment

1. Course policy

The system of requirements for students:

• the student is obliged to attend lectures and laboratory classes, and actively work on mastering the material taught at them;

• at the lecture the lecturer uses his own presentation material;

• laboratory works are defended in two stages: the first stage is - students perform tasks, draw up an electronic report and send to the teacher; the second stage is - defence of the laboratory work in the laboratory. The control of knowledge in the laboratory exercises is carried out by checking the report on laboratory work, as well as through the implementation of modular tests.

• modular test is written in a lecture using all available materials;

2. Monitoring and grading policy

At the first class the students are acquainted with the grading policy which is based on Regulations on the system of assessment of learning outcomes https://document.kpi.ua/files/2020_1-273.pdf. The student's rating in the course consists of points that he/she receives for defended laboratory exercises (R1), the modular control work (R2), and, finally, the final test (R3).

Rs=R1+R2+R3=100 points

As a result, the maximum average weight score is equal to:

7 laboratory exercises x 7 points = 49 points

modular control work = 11 points

final test = 40 points.

According to the university regulations on the monitoring of the student's academic progress (https://kpi.ua/document_control) there are two assessment weeks (attestation), usually during 7th/8th and 14th/15th week of the semester, when students take the Progress and Module tests respectively, to check their progress against the criteria of the course assessment policy.

The condition of the first attestation is to receive at least 10 points (at the time of certification). The condition of the second attestation is to receive at least 30 points (at the time of certification).

The scoring criteria are:

• Execution of laboratory works:

- impeccable work values 7 points;
- there are certain shortcomings in the decorated work - 6-5 points;
- there are some shortcomings in the implementation of the work program - 4-3 points;

Work not performed or not defended - 0 points.

• Final kontrol work is evaluated with 40 points. The control work consists of 16 test questions, as well as a practical task (to design a program) from the list provided in the appendix to the work program.

For each correct answer to the test question 2 points are awarded. The answer to the practical task is evaluated with 8 points according to the following criteria:

- "excellent" - the correct text of the program with comments - 8 - 7 points;
- "good" - the text of the program, in general, correct, but not used optimization techniques - 6-5 points;
- "satisfactory" - there are some fundamental errors in the text of the program - 4 - 3 points;
- "unsatisfactory" - unsatisfactory answer - 0 points.

The students whose finally score the required number of points (≥60) can:

- get their final grade according to the rating score;
- perform a Fail/Pass test in order to increase the grade.

Students can receive up to 6 incentive points for performing creative works from the credit module (compiling abstracts, participating in competitions, in research, etc.).

Students whose final score Rs is below 60 points but more than 30 are required to complete a Fail/Pass test. If the grade for the test is lower than the grade, which the student gets for his semester activity, a strict requirement is applied - the student's previous rating is canceled and he receives a grade based on the results of the Fail/Pass test. Students whose score is below 30 are not allowed to take the Fail/ Pass Test.

The Fail/Pass test is estimated at 60 points. The control task of this work consists of three questions. Each of the 3 questions is evaluated with 20 points according to the following criteria:

- "excellent" - a complete answer (at least 90% of the required information), provided appropriate justifications and personal opinion - 20 - 18 points;
- "good" - a fairly complete answer (at least 75% of the required information), performed in accordance with the requirements for the level of "skills", or minor inaccuracies) - 17… 15 points;

- "satisfactory" - incomplete answer (not less than 60% of the required information, which is performed in accordance with the requirements for the "stereotypical" level and some errors) - 14… 12 points;

- "unsatisfactory" - unsatisfactory answer - 0 points.

The final performance score R_S is adopted by university grading system as follows:

Score	*Grade*
100-95	Excellent
94-85	Very good
84-75	Good
74-65	Satisfactory
64-60	Sufficient
Below 60	Fail
Course requirements are not met	Not Graded

Syllabus of the course

Is designed by teacher D.Sc., Senior Scientist, Anatolii Sergiyenko

Adopted by Department of Computing Engineering (protocol № 10 , 25.05.2022)

Approved by the Faculty Board of Methodology (protocol № 10 , 13.06. 2022)

. ..