National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

Department of Computer Engineering

Complex Systems Design

Syllabus

Details of the academic discipline

Level of higher education	***First (bachelor's)***
Field of Study	*12 Information technologies*
Specialty	*121 Software Engineering*
Education Program	*Computer systems Software Engineering*
Type of Course	*Normative*
Mode of Studies	*daytime*
Year of studies, semester	*4th year, spring semester*
ECTS workload	*4, 5 credits (ECTS) , 135 hours*
Testing and assessment	*Final test*
Course Schedule	*Lectures - 36 hours, laboratory works - 18 hours, independent work of students - 81 hours*
Language of Instruction	*English*
Course Instructors	*Associate professor, Ph.D., Dolgolenko Oleksandr Mykolayovych, aleks.dolgolenko@gmail.com* *Associate professor, Ph.D. , Volokita Artem Mykolayovych, artem.volokita@kpi.ua*
Access to the course	*https://campus.kpi.ua/tutor/index.php?mode=mob*

Program of educational discipline

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

The discipline is aimed at studying the possibilities of innovative technologies implemented in modern Intel microprocessors . Studying this discipline by bachelors will allow them to use these technologies in their professional activities.

The purpose of studying the discipline "Design of complex systems" is to form students' abilities and skills in using the possibilities of innovative technologies implemented in microprocessors, which they will most likely deal with during their professional activities.

The subject of the discipline is:

- types of parallelism in the work of the core of a modern microprocessor ;
- *x86-64* architecture ;
- implementation of a limited data flow scheme in the microprocessor core ;
- *CISC* decoding to a *RISC set* ;
- principles of development of the branching prediction device;
- register renaming mechanism;
- principles of cache memory construction;
- associativity of cache memory and *TLB* buffers;
- m mechanisms of cache coherence;
- innovative technologies implemented in modern microprocessors.

According to the requirements of the National Academy of Sciences, applicants after mastering the discipline "Design of complex systems" must demonstrate the following program learning results:

knowledge:

- features of execution of a separate flow of commands in the core of a modern micro-processor;
- features of parallel program execution in a modern microprocessor;
- features of cache memory and cache coherence mechanism in modern microprocessors (PRN26);
- possibilities of innovative technologies implemented in modern microprocessors

skill:

- choose the optimal strategy for placing variables of a parallel program: in the stack, according to the principles of localization and immutability, or in the heap;
- analyze the effectiveness of the parallel algorithm;
- use the possibilities of innovative technologies implemented in microprocessors when developing software for high-performance computer systems

experience:

- construction and analysis of a parallel algorithm;
- creation and debugging of a parallel program;
- using the possibilities of innovative technologies.

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

To successfully master the discipline, knowledge of the following disciplines is required: "Algorithms and Data Structures", "Computer Systems and Networks Fundamentals", "High Performance Systems Software", "Multi-threading" programming in Java ".

Component based on the learning outcomes of this discipline: "Diploma Design".

Content of the academic discipline

Names of sections and topics	Number of hours				
	In total	including			
		Lectures	Practical (seminar)	Laboratory (computer workshop)	SRS
1	2	3	4	5	6
Chapter 1 . Architectural features of the core construction of a modern microprocessor					
Topic 1.1. Types of parallelism in the operation of the core of a modern microprocessor, bitwise parallelism and parallelism at the data level (scalar and vector instructions).	5	2	-	-	3
Topic 1.2. Parallelism at the command level .	5	2	-	-	3

1	2	3	4	5	6
Topic 1.3. Parallelism at the command thread level.	5	2	-	-	3
Topic 1.4. Architecture *x86-64* .	5	2	-	-	3
Together by chapter 1	**20**	**8**	**-**	**-**	**12**
Section 2 . **Superscalar architecture (*CISC* - *RISC* architecture)**					
Topic 2.1. Decoding *CISC* to *RISC set .*	5	2	-	-	3
Topic 2. 2 . Principles of cache memory construction.	5	2	-	-	3
Topic 2. 3 . Associativity of cache memory and *TLB* buffers.	5	2	-	-	3
Topic 2. 4 . Cache coherence mechanisms.	13	2	-	6	5
Topic 2. 5 . Principles of development of a branching prediction device.	5	2	-	-	3
Topic 2. 6 . Renaming registers .	5	2	-	-	3
Topic 2. 7 . Improved *Tomasulo algorithm* .	5	2	-	-	3
Together by chapter 2	**43**	**14**	**-**	**6**	**23**
Chapter 3 . **Innovative technologies implemented in modern microprocessors**					
Topic 3. 1 . Hardware bypass of locks.	12	2	-	4	6
Topic 3. 2 . Hardware implementation of transactional memory.	16	2	-	8	6
Topic 3. 3 . Intel® OoOE and Intel® QuickPath technologies	3	2	-	-	1
Topic 3.4 Logical segmentation of RAM. Multichannel RAM.	11	4	-	-	7
Topic 3.5. Intel® vPro™ technology.	5	2	-	-	3
Topic 3.6. Remote administration.	5	2	-	-	3
Together by section 3	**52**	**14**	-	**12**	**26**
Modular control work	**8**				**8**

1	2	3	4	5	6
Preparation of prepayment	**4**				**4**
Test	**8**				**8**
Total in the semester:	**135**	**36**	**-**	**18**	**81**

3. Educational materials and resources

1. John L. Hennessy, David A. Patterson. Computer Architecture. A Quantitative Approach, USA, Morgan Kaufmann, 6 th , 2017 – 665p.+ add-ins.

2. J. Shen, M. Lipasti. Modern Processor Design: Fundamentals of Superscalar Processors. Waveland Press, 2013 - 642 p.

3. Intel® 64 and IA-32 Architectures Software Developer Manuals (https://software.intel.com/en-us/articles/intel-sdm).

4. x86 and amd64 instruction reference (https://www.felixcloutier.com/x86/).

Educational content

4. Methodology

Lecture classes

lecture no	The name of the topic of the lecture and a list of the main questions (a list of didactic tools, references to the literature and tasks on the SRS)
1	**Architectural features of the construction of the core of a modern microprocessor** [1, p. 148-366]. Types of parallelism in the operation of the core of a modern microprocessor, bitwise parallelism and parallelism at the data level (scalar and vector instructions).
2	Parallelism at the command level: IPC level (instruction per clock cycle); Scoreboarding and Tomasulo algorithms as a means of achieving IPC >1.
3	Parallelism at the level of command threads: Hyperthreading, Speculation Multithreading, Tasks-Level Parallelism , Fiber parallelism , Thread pool.
4	x86-64 architecture: main registers, index registers, additional registers, command pointer, segment registers, XMM (YMM , ZMM , SSE) registers, state register, scalar and vector registers, introduction to register renaming mechanism.
5	**Superscalar architecture (*CISC* - *RISC* architecture)** , Data Flow , Restricted Data Flow , Front - end , CISC decoding to RISC set , RISC cache (L0), development models: tick - tock and process - architecture - optimization in the development of Intel microprocessors with CISC-RISC architecture .
6	Principles of cache memory construction: TLB buffers and virtual memory; page table data; multi-level page tables; flat memory model _ paged memory model _ X 86 segmented memory model _ virtual, linear and physical addresses; three modes of processing pages in the X86 - 64 architecture [1, p. 247-284].
7	Associativity of cache memory and TLB buffers, policy of replacing data blocks in the cache, influence of associativity on the operation of the processor core.
8	Cache coherence mechanisms, snooping , MESI, MOESI and MESIF protocols.
9	Principles of development of a branching prediction device, branching prediction using BTB , branching history, finite state machine for branching direction prediction

	based on its previous history.
10	Back - end , improved Tomasulo algorithm , data failure threats, reservation stations [1, p. 71-105; 3, p. 177-209, 246-261].
11	Renaming of registers, Allocator , reordering buffer of results [3, p. 217-245].
12	**Innovative technologies implemented in modern microprocessors** . Hardware lock bypassing, IntelTransactionalSynchronizationExtensions (TSX) - NI ® technology, speculativemultithreading , HardwareLockElision (HLE), emergence and development of HLE .
13	Hardware implementation of transactional memory, Restricted Transactional Memory , emergence and development of Hardware Transactional Memory , Haswell 's _ Transactiona lMemory .
14	Intel ® OoOE technology, Intel ® QuickPath technology.
15	Hierarchy of computer memory [1, p. 1158-1169], memory wall, logical segmentation of RAM.
16	Multi-channel RAM, built-in DRAM .
17	Intel® vPro™ technologies , Hyperthreading , VT-x , VT-d , Turbo Boost 3.0, Trusted Execution technology , Cisco Self Defending Network, Microsoft Network Access Protection, Execute disable bit.
18	Remote Administration, Intel Active Management technology, universal unique identifier, vPro - enabled gateway , Intel Management Engine , MINIX 3.

Laboratory work (computer workshop)

The main tasks of the cycle of laboratory work (computer workshop) are students' acquisition of the necessary practical skills to use the possibilities of innovative technologies implemented in modern microprocessors from Intel .

No. z/p	Name of laboratory work (computer workshop)	Number of aud. hours
1	**MESIF cache coherence protocol .** Develop two instances of your version of the Java multithreaded program . Develop the first instance of a multi-threaded program in accordance with the localization principle, while both scalar and vector variables should not be divided between threads - the MESIF cache coherence mechanism will not work in this case . Develop the second instance of a multi-threaded program using scalar and vector global variables and thread synchronization mechanisms when accessing these variables - the search for the current value of the variable will take place, in accordance with the MESIF cache coherence mechanism, in the L1D caches of all microprocessor cores. Build comparative graphs and draw conclusions on the relative speed of these instances of the program.	6
2	**Research on hardware lock bypass technology (Intel 's HardwareLockElision).** This work requires access to an Intel microprocessor that supports TSX-NI technology. Create threads using the *Callable interface* from the *java package . util . concurrent* (thread synchronization using the *Lock interface* from the *java . util . concurrent . lock package*). Explore the execution speed of your	4

	parallel program on this microprocessor. After that, disable Intel ® TSX-NI using the following registry fix (for Windows 10): regadd " HKEY _ LOCAL _ MACHINE \ SYSTEM \ CurrentControlSet \ Control \ SessionManager \ Kernel " / vDisableTsx / tREG _ DWORD / d 1 / f . Reboot the computer for these changes to take effect and re -examine the execution speed of your parallel program variants. Build comparative graphs and draw conclusions about the work. For the next enablement of Intel ® TSX - NI technology , do the same: regadd " HKEY _ LOCAL _ MACHINE \ SYSTEM \ CurrentControlSet \ Control \ SessionManager \ Kernel " / vDisableTsx / tREG _ DWORD / d 0 / f .	
3	**Study of the technology of hardware implementation of transactional memory.** This work requires access to an Intel microprocessor that supports TSX-NI technology. Deploy some DBMS on it. Organize access to the DBMS using a special API, for example, javax.sql. Develop a Java DBMS test program and explore the performance of your DBMS when using hardwaretransactionalmemory. After that, disable TSX-NI technology support on your microprocessor (see the second lab work) and deploy the same DBMS and Java on the same hardware (some libraries will be different). Investigate the performance of this version of the DBMS on the same test program when using softwaretransactionalmemory. Give comparative graphs and draw conclusions about the work.	8
	Together:	18

Instructions for performing laboratory work:

- for the dimension of vectors and matrices chosen by you (at least 100), develop a multi-threaded program that calculates your version of the task and the time of its solution;
- all initial data are positive numbers with a floating point;
- generate initial data, while ensuring that the orders of elements of matrices and vectors are different (the more the orders differ, the more noticeable it will be that the standard mathematical laws for floating-point arithmetic are not fulfilled);
- save the initial data in the file (in the first two laboratory works, enter the initial data from this file);
- organize the output of function calculation results to a file and their synchronized output to the console from the streams where they are calculated;
- to increase the accuracy of floating-point calculations, perform each addition of a sequence of numbers using the *Kahan- type algorithm* , or before performing the addition of a sequence of numbers, first sort it in ascending order and then sum it up, starting with the smallest numbers;
- using synchronization tools, develop a parallel program in Java to calculate two mathematical functions;
- Your variant of mathematical functions for calculating the first two laboratory works is determined by your number in the group list in Campus (where : MA , etc. – matrix; A , etc. – vector; a , etc. – scalar):

OPTION 1 OPTION 2

A=B*(M M +MZ)+E*MM; D=B*(M E +MZ)-E*(MM+M E);
M A =max(BE)*MM*MT-MZ*(MT+MM). M A = min(MM)*(ME+MZ)-ME*MM.

OPTION 3 OPTION 4

M G= MB *M M + MC * M O -MM; A=B*MS+D*MZ+E*MM;
A=D*MC-min(D)*B. M A =max(D+B)*M Z *MM-ML*MC*a.

OPTION 5 OPTION 6

A = B * MC +D*MT; M A = max (B+C)*MD*MT+MZ*MB;
M A = min(D)*MC*ME+MZ*MT. E= B * M D+C*MT*a.

OPTION 7 OPTION 8

A= C *MC+ D * MM * a - B * MT ; C = B*MS- D * MM ;
M A =max(BD)*MD*MT-MC*(MT+MM).MF=min(B+D)*MC*MZ+MM*(MC+MM)*a.

OPTION 9 OPTION 10

E= B * MC +D* min(MC); M G= MB *M O + M M*(M O-MB)d;
M A =b*MD*(MC-MX)+MX*MC*b. A= min(V)*DC*MO*d.

OPTION 11 OPTION 12

Y=D*MT+max(B)*D; A=B*MS+D*MZ+E*MM;
M A =MT*(MT+MZ)-MZ*MT.MG=min(D+E)*MM*MT-MZ*ME.

OPTION 13 OPTION 14

B = D*MO-min(D)*C;M A = max(MD)*(MT+MZ)-MT*MD;
S = SORT(MO*D+D+C). E =B*M D + D * MT .

OPTION 15 OPTION 16

M A =MD*MT+MZ-ME*MM; MG=MD*(MT+MZ)-MT*MZ;
A=D*MT-max(D)*B. A =min(B)*D*MT+B.

OPTION 17 OPTION 18

M A = min(D)*MD*MX+MZ*ME; M A = min(D+ B)*MD*MT+MX*ME;
E=B*M D -D*M Z .C=B*MT+D*MX*a.

OPTION 19 OPTION 20

M G= MB *M O + M S*(M B-MO); O = SORT (P)* SORT (MR * MS);
S = SORT(O*MO+B*MB). M G= M R*MS+ MC *(M R+MS).

OPTION 21 OPTION 22

A = B * MC +D*MZ+E*MT; M G= MB *MT+ MC *(M B*MM+MT);
MG=min(D+B)*MC*MT-MZ*ME. B = D*MT-min(D)*C.

OPTION 23 OPTION 24

MF=MD*(ME+MM)-ME*MM; A =min(C)*C*MT+C ;
E=B*ME+D*max(MM).MX=min(CD)*MD*MT-MC*(MC+MT).

OPTION 25 OPTION 26

M D = min(MT)*(MT+MX)-MT*MX; MG=max(A+C)*MB*MT-MZ*ME*a;
D= B *(M T+MX)-E*(MT- M X); X=A*MB-min(C)*C.

OPTION 27 OPTION 28

M A = MM*(ME-MX)+ME*MX*q ; MF=min(CD)*MC*MZ+MM*(MC+MM) ;
Y=B*ME+D*min(B). X = SORT(MC*M+DC).

5. Bachelor's independent work:

- preparation for lectures (recommended literature is given in the table " Lecture classes");
- preparation of laboratory work;
- preparation for the test.

Policy and control

6. Policy of academic discipline (educational component)

During classes in the discipline "Design of complex systems", bachelors must follow certain disciplinary rules:

- it is forbidden to be late for classes;
- extraneous conversations or other noise that interferes with classes are not allowed;
- leaving the classroom during the lesson is allowed only with the teacher's permission;
- the use of mobile phones and other technical means is not allowed without the teacher's permission.

7. Types of control and rating system for evaluating learning outcomes (RSO)

Current control:

During the semester, students perform 3 laboratory works.

The maximum number of points for each of the laboratory works: 20 points.

Points are awarded for the correctness of the preparation of the protocol of the laboratory work, the express survey on the topic of the laboratory work, and the correspondence of the content of the completed work to the task for it. For the first two works, there is a deadline for its submission, according to the table. "Laboratory works". The first computer workshop, according to the schedule, refers to the time of the first laboratory work. The maximum score for the first work can be obtained in 2 and 3 computer workshops. In the 4th computer workshop: it is possible to get no more than 17 points for 1 work; for the second - up to 20 points. The next step of the deadline begins with the 6th computer workshop: it is possible to get no more than 14 points for 1 work; for the second - no more than 17 points; for the third - up to 20 points. The completed work cannot be rated lower than 10 points.

In total, it is possible to score up to 60 points for laboratory work. Up to 10 points can be obtained for a modular control work.

Calendar control:

For bachelors, one calendar control is provided, which takes place according to the term defined in Campus , and is displayed in the item Campus \ "Calendar control". For its successful completion, you need to have at least one passed laboratory work. Grades for passed laboratory work are issued in the Campus \ "Current control" section.

Semester control:

Semester control includes assessment. It is possible to score up to 30 points on the test.
Conditions for admission to semester control: semester rating – 30 points and above.

Table of correspondence of rating points to grades on the university scale:

Scores	*ECTS assessment*
100-95	Excellent
94-85	Very good
84-75	Good
74-65	Satisfactorily
64-60	Sufficient
Less than 60	Fail
Admission conditions not met	Not allowed

8. Additional information on the discipline (educational component)

The list of questions to be submitted for semester control will be provided in the last lecture of the course.

Working program of the academic discipline (syllabus):

designed by Associate Professor, Ph.D., Seniour Scientist Dolgolenko Oleksandr Mykolayovych
adopted by the Department of Computer Engineering (No. 10 dated May 25 , 2022).
approved by the Methodical Commission of FIOT (protocol No. 10 of 09.06.2022) .