COURSE DESCRIPTION CARD - SYLLABUS

Course name

PO 2.5.2 Programowanie systemów przetwarzania równoległego - EC 2.5.2 Parallel processing systems programming

Course

Field of study
Teleinformatics

Year/Semester
1/2

Area of study (specialization)

Profile of study
general academic

Level of study
second-cycle studies

Course offered in
Polish

Form of study
full-time

Requirements
elective

Number of hours

Lecture
30

Laboratory classes
15

Other (e.g. online)

Tutorials
0

Projects/seminars
0/0

Number of credit points

4

Lecturers

Responsible for the course/lecturer:
dr hab. inż. Olgierd Stankiewicz

Responsible for the course/lecturer:

Prerequisites

1. Knows basic data structures and algorithms used in programming languages.
2. Has a working knowledge of programming methodologies and techniques in high-level languages.
3. Has basic knowledge of digital signal processing methods in data communications.

4. Is able to acquire information from literature, databases and other sources in Polish or English.
5. Is able to use programming mechanisms and programming environments of object-oriented languages and available library software.
6. Knows the limitations of his/her own knowledge and understands the need to update it.
7. Understands the influence of own work on the team results and the necessity of obeying the rules of teamwork and taking responsibility for the tasks performed together.

Course objective

Learning the basic features of programming parallel processing systems. To get acquainted with the existing technical solutions concerning the methods of designing parallel processing algorithms. Prepare your own implementations of selected algorithms.
To shape and develop the ability to acquire knowledge of current developments in parallel processing systems.

Course-related learning outcomes

Knowledge

Understands the capabilities and limitations of parallel processing systems. Knows advanced data structures and algorithms used in parallel processing systems. Has a working knowledge of programming techniques for parallel processing systems.
Knows advanced data structures and algorithms used in programming languages and has a working knowledge of programming methodologies and techniques in high-level languages.

Has advanced knowledge of the design and architecture of parallel processing systems. Knows the basic libraries used in the development and design of software for parallel processing systems. Has advanced knowledge of object-oriented design and programming, architecture of object-oriented programming systems and basic object-oriented libraries in various programming languages, including libraries for programming mobile terminals; has extended knowledge of programming project management.

Knows the correct terminology for parallel processing systems. Has an expanded English vocabulary of data communications and technology.

Skills

Be able to critically evaluate the architecture of a parallel processing system. Indicate its basic features and parameters. Be able to think critically and argue a position.

Able to independently acquire knowledge in the development of parallel processing systems. Able to self-educate.

Can analyze data from literature on parallel processing systems, analyze latest recommendations and normalization documents. Can obtain data from literature, databases and other sources in Polish or English, analyze standardization recommendations, integrate obtained information, interpret it, draw conclusions and formulate and justify opinions.

Be able to use computational algorithms, data structures and techniques used in parallel processing systems. Be able to solve data communication problems using parallel processing systems. Be able to use advanced computational algorithms, data structures and high-level programming languages to solve technical problems related to data communications.

Knows the available libraries and application interfaces used to develop parallel processing systems software. Knows advanced programming mechanisms for creating parallel processing systems software. Is able to use advanced programming mechanisms, programming environments of object-oriented languages, available library software including program application interfaces.

Can design and execute entirely parallel processing systems software necessary to solve the technical issues posed. Can design and fully execute software according to the art of software engineering in solving simple technical issues, can apply software engineering principles to solve a part of a complex computer project.

Social competences

Is aware of a wide range of technical solutions for parallel processing and their continuous development. Knows the limitations of his/her own knowledge and understands the necessity of updating it. Is open to possibilities of continuous education and improvement of professional, personal and social competences.

Understands that parallel programming, including GPU programming, often requires teamwork. Understands the impact of own work on the team results and the necessity of submitting to team work rules, taking responsibility for tasks performed together, sees benefits from the exchange of experiences also in multicultural environment.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Lectures: verification of the assumed effects of learning is realized through assessment of knowledge shown on the test. The test consists of answering questions and solving problems.
A minimum score of 50% is required to receive a grade of 3.0; 3.5 - 60%; 4.0 - 70%; 4.5 - 80%; 5.0 - 90%.

Laboratories: on the basis of evaluation of the current progress of the tasks and activity in class.

Programme content

Architectures for parallel processing systems.
Building computing clusters.
Multithreading.
GPU architectures.
Parallel processing (division of tasks into subtasks, memory and task synchronization methods).
An overview of libraries for developing parallel processing software libraries:
- CPU: C++ threads, pthreads, MPI, OpenMP.
- GPU: OpenCL, Cuda, Vulcan.

Laboratory classes will consist of the preparation by students of programs implementing selected algorithms of parallel processing together with experimental verification of their correctness.

Teaching methods

1. Lectures: multimedia presentation, supplemented with current examples and additional explanations on the blackboard.
2. Laboratories: solving tasks, programming.

Bibliography

Basic

Z. Czech, "Introduction to parallel computing," PWN, Warsaw 2013.
Foster I., "Designing and Building Parallel Programs," book available online at http://www-unix.mcs.anl.gov/dbpp
M. Herlihy, N. Shavit "The Art of Multiprocessor Programming" Elsevier, 2008 (Polish edition "Sztuka programowania wielooprocesorowego", PWN 2010)

Additional

Grama A. et al, "Introduction to Parallel Computing" (2nd ed.), Addison-Wesley, 2003
Websites: www.openmp.org, www.mpi-forum.org

Breakdown of average student's workload

	Hours	ECTS
Total workload	86	4.0
Classes requiring direct contact with the teacher	45	2.0
Student's own work (preparation for tests, preparation for laboratory classes, literature studies)	41	2.0