GDAŃSK UNIVERSITY OF TECHNOLOGY

Subject card

Subject name and code	Multithreaded Programming, PG_00067969						
Field of study	Automatic Control, Cybernetics and Robotics						
Date of commencement of studies	October 2025		Academic year of realisation of subject	2026/2027			
Education level	first-cycle studies		Subject group	Optional subject group Subject group related to scientific research in the field of study			
Mode of study	Full-time studies		Mode of delivery	at the university			
Year of study	2		Language of instruction	Polish			
Semester of study	4		ECTS credits	3.0			
Learning profile	general academic profile		Assessment form	assessment			
Conducting unit	Department of Signals and Systems -> Faculty of Electronics Telecommunications and Informatics -> Faculties of Gdańsk University of Technology						
Name and surname of lecturer (lecturers)	Subject supervisor	dr inż. Piotr Fiertek					
	Teachers	dr inż. Piotr Fiertek					
Lesson types	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	15.0	0.0	15.0	15.0	0.0	45
	E-learning hours included: 0.0						
Learning activity and number of study hours	Learning activity	Participation in didactic classes included in study plan		Participation in consultation hours	Self-study		SUM
	Number of study hours	45		3.0	27.0		75
Subject objectives	The aim of the course is to introduce students to multithreaded programming and creating multithreaded software on various hardware platforms.						

Learning outcomes

Course outcome	Subject outcome	Method of verification
[K6_W03] knows and understands, to an advanced extent, the construction and operating principles of components and systems related to the field of study, including theories, methods and complex relationships between them and selected specific issues - appropriate for the curriculum	Student describes and is able to apply in practice multithreaded software development techniques.	[SW1] Assessment of factual knowledge
[K6_W10] knows and understands, to an advanced extent, the parameters, functions, and methods of analysis, design, and optimization of electronic circuits and systems, the definitions of error and measurement uncertainty, measurement methods, including time, frequency, and phase measurements, the properties of converters, and methods of digital signal processing, as well as the basic processes occurring in the life cycle of technical devices, objects, and systems, and methods of supporting processes and functions, specific to the field of study	Student describes and is able to evaluate the usefulness of various mechanisms used in concurrent programming and is able to apply them to solve practical problems.	[SW3] Assessment of knowledge contained in written work and projects [SW1] Assessment of factual knowledge
[K6_U04] can apply knowledge of programming methods and techniques as well as select and apply appropriate programming methods and tools in computer software development or programming devices or controllers using microprocessors or programmable elements or systems specific to the field of study	Student describes and is able to use in practice basic techniques used in multithreaded programming.	[SU1] Assessment of task fulfilment [SU4] Assessment of ability to use methods and tools

Subject contents

Course content – lecture

1. Introduction to multithreaded programming. Thread and process. Concurrent processing versus parallel processing.
2. Basic multithreaded algorithms. Fibonacci sequence generation, matrix multiplication, merge sort.
3. Concurrent data structures.
4. Design patterns in concurrent programming (ProducerConsumer, ReaderWriter, pipeline, Actor model).
5. Thread management methods and thread synchronization tools. Atomic operations, mutexes, semaphores, messages. Implementing mutual exclusion.
6. Thread scheduling and solving thread interaction problems. Interthread communication (IPC, queues, signals).
7. Thread priorities. Priority inheritance.
8. Implementing multithreading in single- and multiprocessor systems. Multithreading.
9. Problems with multithreading. Deadlocks and starvation. Causes and solutions.
10. Multithreading in high-level languages.
11. Security in concurrent applications, race condition attacks. Principles of secure design. Examples of real-world incidents.
12. Testing and debugging multithreaded applications (race condition detection, stress testing, deterministic error recovery).
13. Multithreading in microcontrollers: FreeRTOS as an example. 14. The Pthread library and its mechanisms. 15. Programming with the OpenMP library.

Course content – laboratory

The course includes six individual projects and their laboratory implementation. Project topics:

1. Configuring an environment containing FreeRTOS for the STM microcontroller. Running the first program containing two threads.
2. Creating threads to handle the lab kit components using semaphores and inter-thread communication.
3. Interrupt-handling events generated by the lab kit (e.g., the AD converter).
4. Controlling the lab kit components using a RaspberryPI computer.
5. Incorporating additional features into the developed software for the RaspberryPI using multithreading (OpenMP or Pthread).
6. Handling interrupts generated by the lab kit components (for the RaspberryPI).

Course content – project

1. Design of a multithreaded program (FreeRTOS) for an STM32 microcontroller using I/O registers.
2. Design of a multithreaded program for an STM32 microcontroller with asynchronous threads (e.g., supporting an AD converter).
3. Design of a program for a Raspberry PI microcontroller (Linux) using I/O registers.
4. Design of a program for a Raspberry PI using multithreading and the OpenMP or Pthread library.
5. Design of a program for a Raspberry PI using multithreading and the Pthread library.
6. Interrupt handling and asynchronous threads in LINUX (Raspberry PI).

Prerequisites and co-requisites

<table>
<tr><td>Assessment methods and criteria</td><td colspan="2"><table>
<tr><th>Subject passing criteria</th><th>Passing threshold</th><th>Percentage of the final grade</th></tr>
<tr><td>Midterm colloquium</td><td>51.0%</td><td>50.0%</td></tr>
<tr><td>Evaluation of projects and their implementation</td><td>51.0%</td><td>50.0%</td></tr>
</table></td></tr>
<tr><td rowspan="3">Recommended reading</td><td>Basic literature</td><td>"OpenMP Application Programming Interface", OpenMP Architecture Review Board;
"C++ Concurrency in Action: Practical Multithreading", A. Williams, HELION SA, 2013;
"Algorithms and Parallel Computing", F. Gebali, A John Wiley & Sons, Inc., 2011;</td></tr>
<tr><td>Supplementary literature</td><td>"Introduction to Parallel Computing", W. P. Petersen, P. Arbenz, OXFORD UNIVERSITY PRESS, 2004A. Williams, "The Art of Multiprocessor Programming", M. Herlihy N. Shavit, Elsevier Inc, 2008;</td></tr>
<tr><td>eResources addresses</td><td></td></tr>
<tr><td>Example issues/ example questions/ tasks being completed</td><td colspan="2">1. What are synchronous, asynchronous, and background threads? What are the main differences between them?
2. Discuss the methods and mechanisms of thread synchronization.
3. What is the phenomenon called a "race condition"? When does it occur, what are its effects, and how can it be eliminated?
4. Why can using too many threads result in a decrease in application performance?
5. Describe the multithreaded algorithm for creating Fibonacci sequence entries.
6. What is priority inheritance and what is its purpose?
7. What is a deadlock and when can it occur?
8. How does a semaphore work? How does it differ from a mutex?
9. List and briefly describe the basic methods of communication between threads.</td></tr>
<tr><td>Practical activites within the subject</td><td colspan="2">Not applicable</td></tr>
</table>

Document generated electronically. Does not require a seal or signature.