Blekinge Institute of Technology
Department of Computer Science

Revision: 1,1
Reg.no: BTH-4.1.14-0926-2023

COURSE SYLLABUS

Multiprocessorprogrammering

Multiprocessor Programming

7.5 credits (7,5 högskolepoäng)

Course code: DV2597
Main field of study: Computer Science
Disciplinary domain: Technology
Education level: Second cycle
Specialization: A1N - Second cycle, has only first cycle course/s as entry requirements

Language of instruction: English
Applies from: 2021-08-23
Approved: 2023-08-11

1. Decision

This course is established by Dean 2020-03-04. The course syllabus is approved by Head of Department of Computer Science 2023-08-11 and applies from 2021-08-23.

2. Entry requirements

Admission to the course requires 60 credits completed in the subject area Computer science or in the subject area Software engineering, including completed courses in programming, 12 credits, algorithm and data structures, 6 credits and operating system 6 credits and computer engineering 6 credits.

3. Objective and content

3.1 Objective

Parallelism has been an approach to achieve high performance in computer systems for many years. Today, computer systems incorporate parallelism at several levels: regular arrays (matrices) of SIMD processor cores as in GPUs, a smaller number of general CPU cores as in multicore processors, or as connected processing nodes as in a distributed system. To utilize the performance potential of the hardware, we need to write efficient parallel programs. In other words, parallel computer systems and parallel programming are fundamental building blocks for contemporary as well as future computer systems.

The aim of this course is to teach the students how to program computer systems consisting of many processors or processor cores to achieve high performance. We will look at programming of shared memory, both MIMD and SIMD, as well as programming of distributed systems. The focus in the course will be on programming of shared-memory multiprocessors.

3.2 Content

The course covers the following areas:
• Introduction to multiprocessor systems and parallel programming
• Design principles for parallel computer architectures and computer systems
• Programming models and design principles for parallel programs to achieve high performance, e.g., partition of work to fit the target architecture, communication, and synchronization
• Libraries for parallel programming, e.g., pthreads and CUDA
• General-purpose computing on GPUs (GPGPU)
• Example algorithms for a number of interesting application domains
• Practical training in development of parallel programs

4. Learning outcomes

The following learning outcomes are examined in the course:

4.1 Knowledge and understanding

On completion of the course, the student will be able to:
• Thoroughly describe the design and working conditions of different types of multiprocessors and parallel computer systems
• Thoroughly describe different programming models for multiprocessors and parallel computer systems

• Thoroughly describe different parallel program design principles to achieve high performance as well as how the architecture affects the performance of a program

4.2 Competence and skills

On completion of the course, the student will be able to:
• Practically apply different principles and techniques for developing and implementing parallel program
• Analyze the data flow in a problem in order to partition the program in parallel parts
• Perform simple performance measurements and performance analysis of parallel programs
• Implement a general algorithm / program for execution on a GPU
• Perform suitable optimizations of a parallel program, e.g., reducing communication and synchronization overhead, to achieve high performance

4.3 Judgement and approach

On completion of the course, the student will be able to:
• In writing and orally present and motivate their solutions
• Independently and critically evaluate and analyze their solutions

5. Learning activities

The teaching consists of a combination of lectures and laboratory exercises. The lectures present and explains the theoretical parts of the course. In addition, the students are expected to independently acquire the theoretical knowledge by self studies of relevant literature. The laboratory assignments focus on exercises that develop the student's knowledge in the domain.

The mandatory laboratory assignments are solved independently or in groups within given time limits. Information whether the laboratory assignments can be solved in groups is given at course start and in the course PM. The laboratory assignments are mainly performed in the programming languages C / C++. The student is expected to have enough knowledge of these languages to independently be able to apply the theory in practice to solve the laboratory assignments.

6. Assessment and grading

Modes of examinations of the course

Code	Module	Credits	Grade
2110	Written examination[1]	3 credits	AF
2120	Laboratory session 1	1.5 credits	GU
2130	Laboratory session 2	1.5 credits	GU
2140	Laboratory session 3	1.5 credits	GU

[1] Determines the final grade for the course, which will only be issued when all components have been approved.

The course will be graded A Excellent, B Very good, C Good, D Satisfactory, E Sufficient, FX Fail, supplementation required, F Fail.

The information before a course occasion states the assessment criteria and make explicit in which modes of examination that the learning outcomes are assessed.

An examiner can, after consulting the Disability Advisor at BTH, decide on a customized examination form for a student with a long-term disability to be provided with an examination equivalent to one given to a student who is not disabled.

7. Course evaluation

The course evaluation should be carried out in line with BTH:s course evaluation template and process.

8. Restrictions regarding degree

The course can form part of a degree but not together with another course the content of which completely or partly corresponds with the contents of this course.

9. Course literature and other materials of instruction

Main literature
• Wen-mei W. Hwu, David B. Kirk, and Izzat El Hajj, Programming Massively Parallel Processors: A Hands-on Approach, 4th Edition, 2022, ISBN-9780323912310 (paper back) ISBN-9780323984638 (ebook).
• T. Rauber and G. Rünger, Parallel Programming for Multicore and Cluster Systems, 2nd ed., Springer, 2013, ISBN 978-3-642-37800-3.
• Material from the department.

Reference literature
• Wolfgang Engel, GPU PRO 4: Advanced Rendering Techniques, ISBN-10: 1466567430 ISBN-13: 978-1466567436.

• Jason Zink, Matt Pettineo, Jack Hoxley, Practical Rendering and Computation with Direct3D 11, ISBN-10:1568817207, ISBN-13: 978-1568817200.
• A. Grama, A. Gupta, G. Karypis, and V. Kumar, Introduction to Parallel Computing, 2nd edition, Addison-Wesley, 2003, ISBN 0201648652.
• J.L. Hennessy and D.A. Patterson, Computer Architecture – A Quantitative Approach, 4th edition, Morgan Kaufmann Publishers, 2007, ISBN 0-12-370-490-1.
• Maurice Herlihy, Nir Shavit, Victor Luchangco, and Michael Spear, The Art of Multiprocessor Programming, 2nd edition, Morgan Kaufmann Publishers, 2020, Paperback ISBN: 9780124159501, eBook ISBN: 9780123914064.