Module Leader

**Dr A H Gee** [1]

Lecturers

Dr A H Gee and Dr P O Kristensson

Coursework Leader

**Dr P O Kristensson** [2]

Timing and Structure

Lent Term. 75% exam / 25% coursework

Prerequisites

Part 1 Digital Circuits and Computing assumed

Aims

The aims of the course are to:

- Describe the computer hardware that underlies modern information processing systems.
- Explain how to write multithreaded software that runs on such hardware.

Objectives

As specific objectives, by the end of the course students should be able to:

- Appreciate the basic components needed to construct a computer and the different ways to interconnect these components, including the various ways of exploiting parallelism.
- Compare the instruction sets, implementation issues and performance of CISC and RISC architectures.
- Design efficient hardware for computer arithmetic.
- Understand the operation of pipelined datapaths.
- Describe memory organisation, addressing schemes and the use of caches; and their effects on performance.
- Compare the various ways of handling input and output in a computer system.
- Understand the concept of a memory model.
- Understand basic concurrency concepts.
- Design and implement thread-safe algorithms in C++.

Content

Computer Systems (8L + 2 examples classes, Dr Andrew Gee)
Computer architecture, historical perspectives.
Instruction set architectures, RISC vs CISC.
ALU design, datapaths and control, pipelining.
Memory hierarchy, caches, virtual memory.
Input/output, bus organization, polling and interrupt-driven I/O, DMA.
Parallel processing, SIMD and MIMD architectures.

Assessment: examination (75%), candidates to attempt two questions from a choice of three

Parallel Programming (4L, Dr Per Ola Kristensson)

- C++11/14/17 memory model.
- Race conditions, mutual exclusion, synchronization, starvation.
- Thread-safe data structures.
- C++11/14/17 threading library

Assessment: coursework (25%)

Coursework

Multithreaded programming using the C++11/14/17 memory model and threading libraries. The programming exercise is an opportunity to experience how theoretical concepts from the lectures translate into actual working code using a state-of-the-art industry standard threading library. Time required: four hours programming plus 15 minutes demonstrating and discussing your code with an assessor. Please note that coursework assessment is not anonymous.

<table>
<thead>
<tr>
<th>Coursework</th>
<th>Format</th>
<th>Due date &amp; marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multithreaded programming</td>
<td>Individual</td>
<td>Software to be</td>
</tr>
<tr>
<td>Learning objectives:</td>
<td>Demonstrating your software</td>
<td>Lent Term or</td>
</tr>
<tr>
<td></td>
<td>Not anonymously marked</td>
<td>Assessment Term</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[15/60]</td>
</tr>
</tbody>
</table>

Booklists

Please see the Booklist for Group F Courses [3] for references for this module.

Examination Guidelines

Please refer to Form & conduct of the examinations [4].

Last modified: 04/10/2017 14:58