# *CO1904 Computer Architecture*

**Credits:** *20* **Convenor:** *Dr. R. Crole* **Semester:** *1st*

**Prerequisites:** *none*

| | | | |
|---|---|---|---|
| **Lectures:** | *22 hours* | **Independent Study:** | *106 hours* |
| **Tutorials:** | *22 hours* | | |

**Assessment:** *Coursework: 20% + Two hour exam in January: 80%*

| *Formative Coursework* | |
|---|---|
| **Assignments:** | *1 in total* |

| *Summative Coursework* | |
|---|---|
| **Class Tests:** | *2 in total* |
| **Assignments:** | *2 in total* |

## Learning Outcomes **Students should be able to:**

- Explain and discuss an overview of datapath and control of a modern processor; outline how it fits within a computer.
- Explain fundamental binary systems, digital hardware and logic and solve simple problems.
- Discuss and critique a high level view of current processors or a detailed view of a simple model processor, including the ISA. Solve simple problems.

**Explanation of Prerequisites** No specific knowledge is required, but a very rudimentary understanding of logic and discrete mathematics will be helpful. Some programming experience is also helpful, but not essential.

**Module Description** While modern **computer**s and computer-controlled devices are complex, there are key components from which these are built. In particular they have a *processor* which might be thought of as the heart of a computer. The way components fit together to form a processor is called the processor **architecture**. To understand a processor you need to know about hardware and software.

This module provides a broad picture of key hardware and software components: Very roughly speaking, hardware refers to physical artefacts such as a capacitive OLED screen or memory board, and software to programs which are stored using magnetic or electrical systems. And what are programs? These are instructions which are executed on the processor.

The module will teach details of computer arithmetic (arithmetic calculations take place when almost any program runs), the processor, and low-level software programs that execute on the processor hardware. In doing so we overview the hardware components such as memory, an arithmetic and logic unit, data-selectors and so on. We look at different formulations of the software and how to write simple programs. And we study an achitecture in detail.

At the end of the module you will have a good understanding of how state-of-the-art processors work, how they are built, and will be able to read about and understand more advanced topics in architecture. You will be able to learn more about operating systems and networks. And knowledge of binary, hex, and how computers work will be useful in many other applications areas, especially graphics and GPUs.

**Syllabus** *Examples throughout the course will be based on the MIPS and ARMv8 Instruction Set Architectures. See the module web page for industry details.*

The binary and hexary number systems. Binary arithmetic: basic definitions, algorithms for computing arithmetic operations. Signed and unsigned integers. Overflow and correctness conditions. Elementary logic and truthtables.

The top level view of a modern computer: memory, processor, I/O, the fetch, decode, execute cycle. Memory layouts and the Endian systems. The memory hierachy.

Hardware components: Gates for implementing simple logical propositions. Multiplexors, decoders, and related circuits. Clocks. Arithmetic Logic Units (ALUs). Simple memory circuits (bits and cells). Basics of register files and memory boards.

The MIPS and ARMv8 RISC instruction sets and simple programs. A subset of the MIPS/ARM languages treated in detail at the assembly and machine levels. Syntax and semantics. Translating assembly instructions into machine language fields including branch address calculations.

Construction of a simple (core) datapath via composition of atomic ALUs, register file(s), memory(ies) and other basic components. Description of control program. The interaction of the datapath and control to make a processor.

Introduction to advanced topics: these might include pipelining and parallelism, multi-core processors, GPUs, and cache memory (according to available time).

## Reading List

[B] Hennessy and Patterson, *Computer Organization and Design; 9780124077263, 9780124078864 (5th),* Morgan Kaufmann, 2013, MIPS fifth edition.

[B] Hennessy and Patterson, *Computer Organization and Design; 9780128017333, 9780128018354 (1st),* Morgan Kaufmann, 2016, ARM first edition.

[C] Tanenbaum, *Structured Computer Organization; ISBN-13: 9780132916523, 9780133061796,* Prentice Hall (Pearson), 2013 (sixth edition).

[C] Stallings, *Computer Organization and Architecture; ISBN-13: 9780132936330,* Prentice Hall, Pearson, 2013 (ninth edition - out of print?).

## Convenor's Notes

**Module Learning** This module is quite closely based on Hennessy and Patterson. If you wish to do preliminary reading, this would be a good book to look at; and this is also the best book to use as support during the module itelf. (For preliminary study use the MIPS edition.)

**Assessment** This module has a final examination, and ongoing coursework during term. There are 4 individual hand-in courseworks, CW1, CW2, CW3, CW4. These comprise two class tests and two independent study assignments. One is Formative and the others Summative. The 20% coursework mark breaks down as follows:

**CW1:** Assignment, F

**CW2:** Mini Test, S, 3% module mark

**CW3:** Assignment, S, 7% module mark

**CW4:** Test, S, 5% module mark

**CW5:** Test, S, 5% module mark

There will also be coursework preparation (CWP) assignments. These will not be assessed/handed-in and are designed to help you to achieve the module learning outcomes (in particular, prepare you for the Coursework and Examination).

Students taking CO1904 at level 2 will be required to take some additional learning and assessment since the module is worth 20 credits. The assessment will take the form of a CW5 take-home assignment.