332
Advanced Computer Architecture
Chapter 1.1

Introduction
Is this course for you? How will it work? What will you learn?

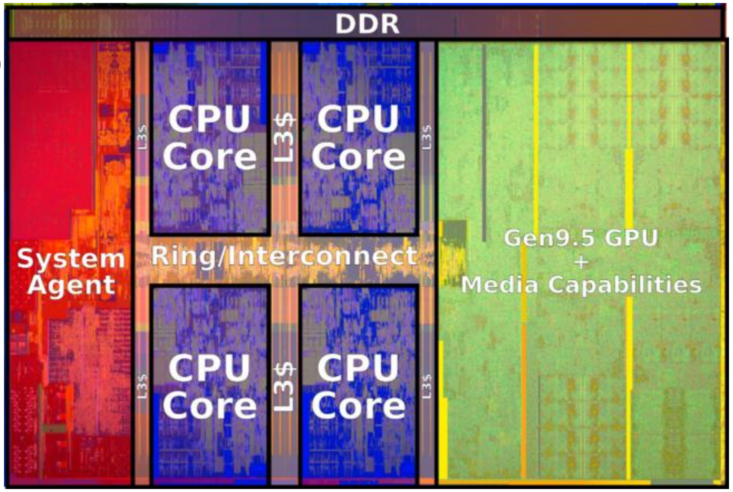
October 2023
Paul H J Kelly

These lecture notes are partly based on the course text, Hennessy and Patterson's *Computer Architecture, a quantitative approach (6th ed), and on the lecture slides of David Patterson's Berkeley course (CS252)*

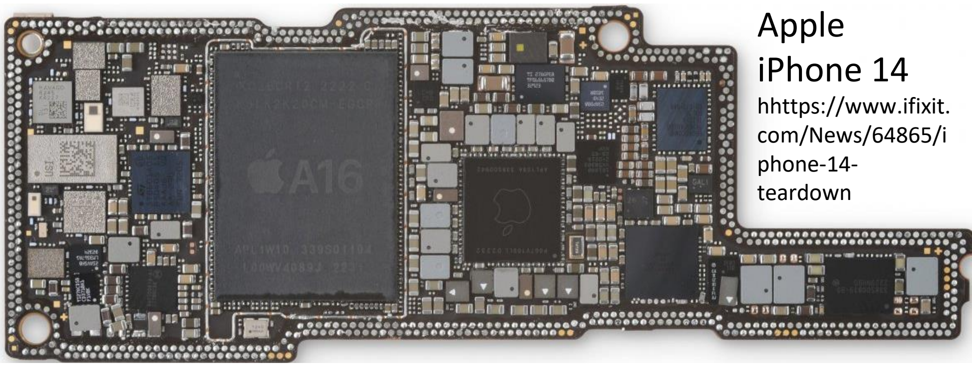
Course materials online on
https://scientia.doc.ic.ac.uk/2324/modules/60001/materials and
https://www.doc.ic.ac.uk/~phjk/AdvancedCompArchitecture/aca20/

What is this course about?

- How the latest microprocessors *work*
- Why they are built that way – and what are the alternatives?
- How you can make software that uses the hardware in the best possible way
- How you can make a compiler that does it for you
- How you can design a computer for *your* problem
- What does a *big* computer look like?
- What are the fundamental big ideas and challenges in computer architecture?
- What is the scope for theory?



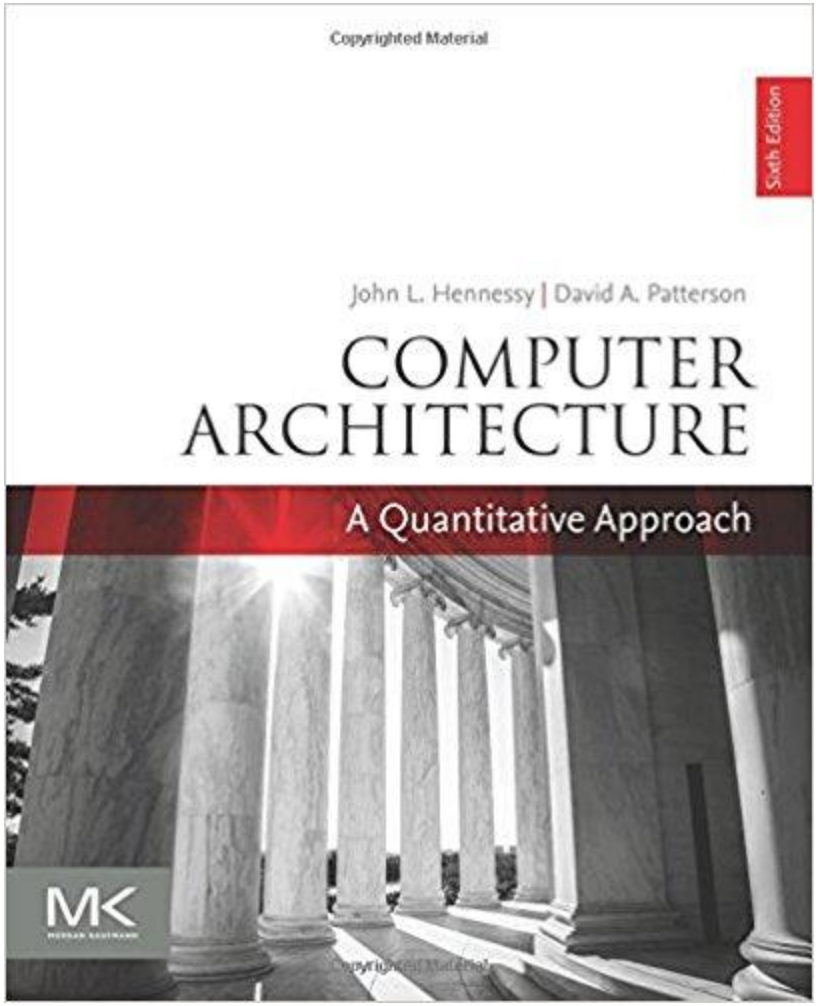
Intel Kaby Lake (my laptop) https://en.wikichip.org/wiki/intel/core_i7/i7-8650u



Apple iPhone 14 hhttps://www.ifixit.com/News/64865/iphone-14-teardown

Frontier (Oak Ridge National Labs, USA) https://en.wikipedia.org/wiki/Frontier_(supercomputer)

This is a textbook-based course



Computer Architecture: A Quantitative Approach (6^{th} Edition)

John L. Hennessy, David A. Patterson

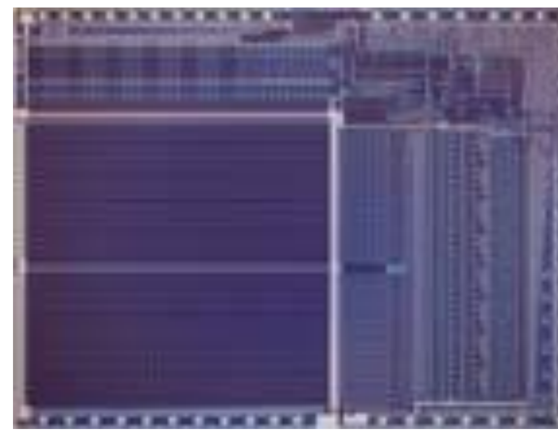
- 936 pages. Morgan Kaufmann (2017)
- ISBN: 9780128119051
- Price: around £70 (shop around!)
- Publisher's companion web site:
  - https://www.elsevier.com/books-and-journals/book-companion/9780128119051
  - Textbook includes some vital introductory material as appendices:
  - Appendix C: tutorial on pipelining (read it NOW)
  - Appendix B: tutorial on memory hierarchy (read it NOW)
- Further appendices (some in book, some online) cover more advanced material (some very relevant to parts of the course), eg
  - Networks
  - Parallel applications
  - Embedded systems
  - Storage systems
  - VLIW
  - Computer arithmetic (esp floating point)
  - Historical perspectives

Who are these guys anyway and why should I read their book?

- John Hennessy:
  - Founder, MIPS Computer Systems
  - President (2000-2016), Stanford University
  - Board member, Cisco, chair of Alphabet Inc (parent company of Google)
  - The “godfather of Silicon Valley” (Wikipedia)

RAID-I (1989) consisted of a Sun 4/280 workstation with 128 MB of DRAM, four dual-string SCSI controllers, 28 5.25-inch SCSI disks and specialized disk striping software.

- David Patterson
  - Leader, Berkeley RISC project
  - RAID (redundant arrays of inexpensive disks)
  - Professor, University of California, Berkeley
  - President of ACM 2004-6
  - Served on Information Technology Advisory Committee to the US



http://www.cs.berkeley.edu/~pattrsn/Arch/prototypes2.html

RISC-I (1982) Contains 44,420 transistors, fabbed in 5 micron NMOS, with a die area of 77 mm^2, ran at 1 MHz. This chip is probably the first VLSI RISC.

Joint winners of the 2017 ACM Turing Award

“For pioneering a systematic, quantitative approach to the design and evaluation of computer architectures with enduring impact on the microprocessor industry”

Course organisation

- Lecturer:
  - Paul Kelly – Leader, Software Performance Optimisation research group
    - With help from PhD students
- Nominally four lecture hours per week, up to two hours “synchronous”
- In the last couple of lectures we will spend some time on exam preparation

- Assessment:
  - Exam
    - The exam will take place in last week of term
    - The goal of the course is to teach you how to think about computer architecture
    - The exam is designed to test your thinking, your understanding – not your memory – have a look at the past papers
  - Coursework
    - You will be assigned two coursework exercises
    - You will learn about using simulators, and experimentally evaluating hypotheses to understand system performance
    - You will get introduced to the research frontier in the field
    - You are welcome to bring laptops to class to get started and get help with lab work (we may go to the DoC labs when necessary)

Course overview (plan)

- Ch1
  - Review of pipelined, in-order processor architecture and simple cache structures
- Ch2
  - Dynamic scheduling, out-of-order
  - Register renaming
  - Speculative execution
- Ch3
  - Branch prediction
- Ch4
  - Caches in more depth
  - Software techniques to improve cache performance
  - Virtual memory and protection
- Ch5
  - Side-channel vulnerabilities
- Ch6
  - Static instruction scheduling
  - Software pipelining
  - instruction-set support for speculation and register renaming
- Ch7
  - Multi-threading
- Ch8
  - Data-parallelism, SIMD and vector
- Ch9
  - Graphics processors and manycore
- Ch10
  - Shared-memory multiprocessors
  - Cache coherency
  - Atomicity, consistency
  - Large-scale cache-coherency; ccNUMA. COMA

- **Lab-based coursework exercise:**
  - “Exploration”: Simulation study
  - “Evaluation”: summarise and evaluate a recent research paper in computer architecture
- **Exam:**
  - Partially based on recent processor architecture article, which we will study in advance (see past papers)

External Students – Registration for DoC Courses

1. Apply at: `https://dbc.doc.ic.ac.uk/externalreg/`
2. Then,
   - Your department's endorser will approve/reject your application
3. If approved,
   - DoC's External Student Liaison will approve/reject your application
4. If approved (again!),
   - Students will get access to DoC resources (DoC account, CATE, ...)
   - No access after a few days? Check status of approval and contact relevant person(s)

Key Dates

- Exams for DoC $3^{rd}/4^{th}$ yr. courses take place at the end of the Term in which the course is taught
- Registration for exams opens in November for Autumn courses and end January for Spring term courses

If in doubt, read the guidelines available at the link above ☺

Main points:

- If you have studied computer architecture before, you should be able to this course
  - Do you know what a pipeline stall is? Do you know what a cache miss is?
  - You will also need to do some C programming, a little Linux command-line and bash-scripting
- By the end you will understand the main features and design alternatives in computer architectures widely used today
  - The "microarchitecture" of single cores, for both low power and high performance
  - Multicore systems, including how the cores are connected and how memory consistency is maintained
  - Graphics processors – at least from a compute point of view (rather than graphics)
  - Large-scale computer systems, supercomputers
- The course's examinable content is defined by the lecture slides, but you will benefit from reading more widely
- The textbook provides both more depth and more breadth

- There will be two assessed coursework exercises:
  (1) "Exploration" – find a single-core microarchitecture configuration that runs a given program with the lowest total energy
  (2) "Evaluation" – write a brief summary and evaluation of an article from one of this year's main computer architecture conferences

- The final exam will be partly based on an article about a recent architecture, which we will study in detail in class