

COMP 252

System Architecture

Antoniu Pop

`antoniu.pop@manchester.ac.uk`

Lecturers

- ▶ Antoniu Pop – antoniupop@manchester.ac.uk
- ▶ Oliver Rhodes – oliver.rhodes@manchester.ac.uk

22 Lectures

- ▶ Mondays at 13:00 in Chem G.54
- ▶ Tuesdays at 17:00 in Simon 3.44A

Guest lecture

- ▶ *Multicore Systems (?)*
- ▶ John Goodacre

Labs

- ▶ 5 × 2 hour sessions
- ▶ Wednesdays at 9:00 (group G - Kilb G23) and Fridays at 12:00 (group F - Tootill)
- ▶ Lab results count for **20%** of the course final mark
- ▶ Extensions are automatic, and **included** in the following deadlines:

Ex 1 - Measuring Cache Performance	18:00 on 14 Feb
Ex 2 - Direct Mapped Cache	18:00 on 28 Feb
Ex 3 - Fully Associative Cache	18:00 on 13 Mar
Ex 4 - Measuring Computer Performance	18:00 on 27 Mar

- ▶ Labs **must** be marked at the latest in the following session

Recommended textbooks

- ▶ D.A. Patterson & J.L. Hennessy, *“Computer Organization and Design. The Hardware/Software Interface”*, Morgan Kaufmann, now in 4th Edition.
- ▶ J. Smith & R. Nair, *“Virtual Machines: Versatile Platforms for Systems and Processes”*, Morgan Kaufmann.

Further reading:

<http://syllabus.cs.manchester.ac.uk/ugt/COMP25212/>

Aims of this course

- ▶ To introduce architectural techniques which are used in modern processors and systems
- ▶ To understand how the specification of systems affects their performance and suitability for particular applications
- ▶ To understand how to design such systems

Course overview

- ▶ Architectural techniques – making processors go faster
 - ▶ Caches
 - ▶ Pipelines
 - ▶ Multi-Threading
 - ▶ Multi-Core

- ▶ How to make processors more flexible
 - ▶ Virtualization

- ▶ The architecture of permanent storage

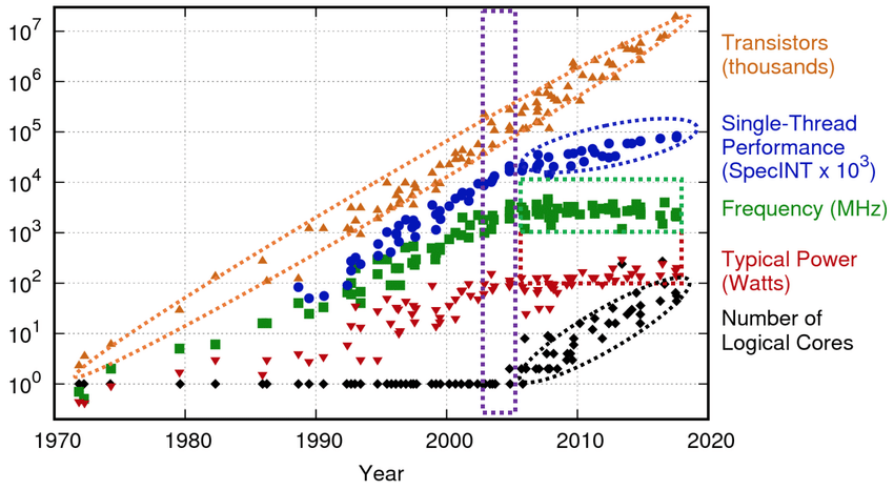
Motivation for performance

- ▶ There is always a demand for increased computational performance
- ▶ Current “microprocessors” are several thousand times faster than when they were first introduced 4 decades ago.
- ▶ But still lots of things they can't do in real time due to lack of speed – e.g. HD video synthesis, realistic game physics, machine learning problems

Architecture & the future

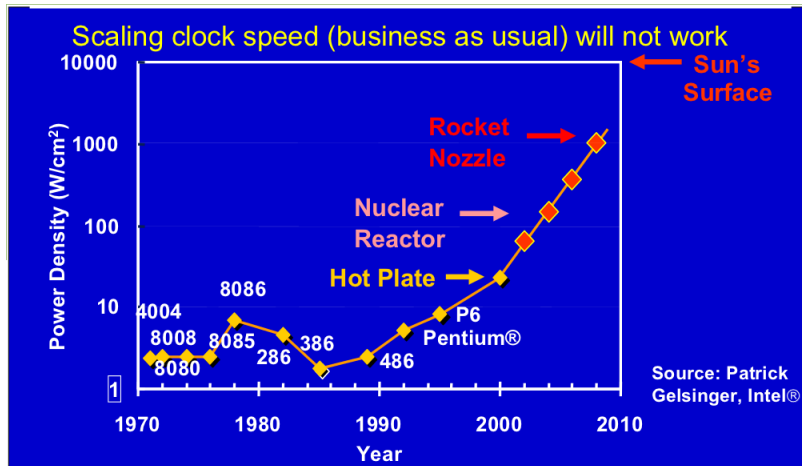
- ▶ Speed improvements due to technology have slowed since around 2004/5
 - ▶ Physical production limits
 - ▶ Power Dissipation
 - ▶ Device Variability

- ▶ Architecture plays a larger role in future performance gains
 - ▶ Parallelism (multi-core, many-core, GPGPU, various accelerators)
 - ▶ Hardware implementation (ASIC, FPGA)

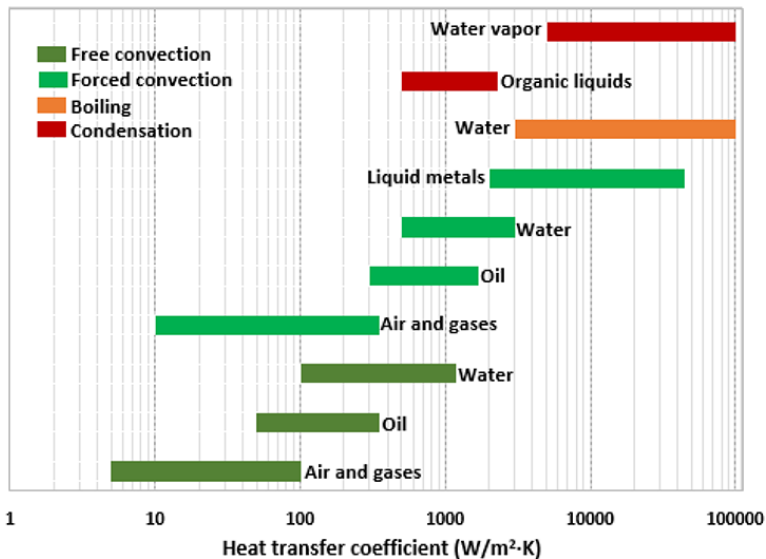


Source: M.Rusanovsky et al. (2019). *BACKUS: Comprehensive High-Performance Research Software Engineering Approach for Simulations in Supercomputing Systems*.

Power [density] Wall: end of Frequency Scaling



$$P = A \cdot C \cdot V^2 \cdot f$$

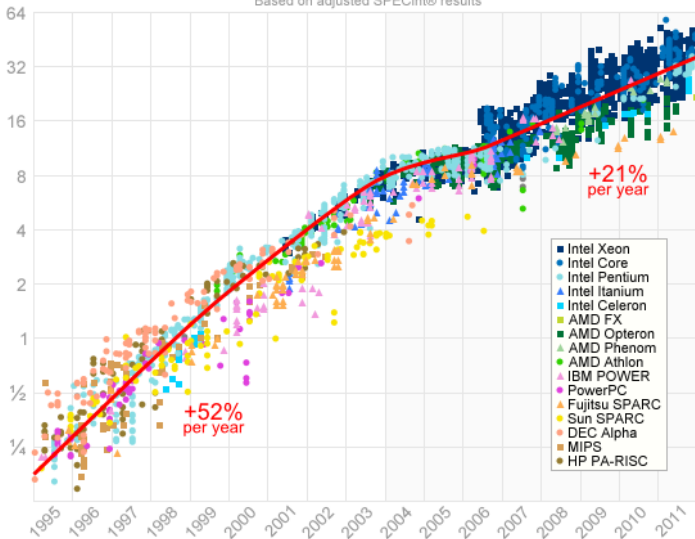


Credit: Computer and Information Science "Computer Science and Engineering – Electronics Cooling", book edited by S M Sohel

Murshed – CC BY 3.0 license – DOI: 10.5772/63321

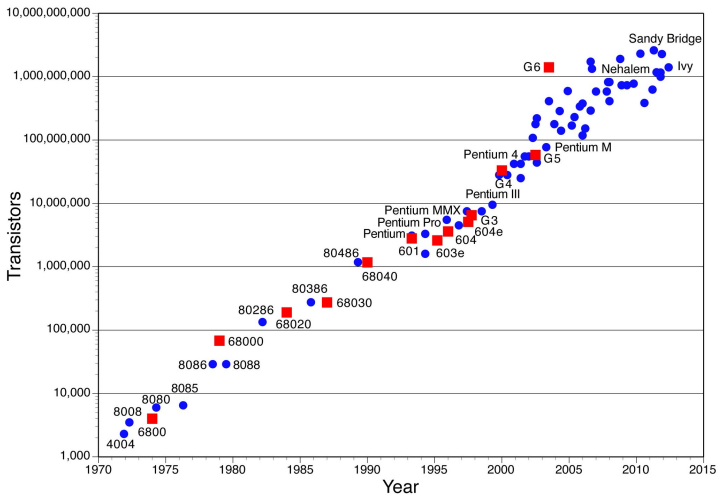
Single-Threaded Integer Performance

Based on adjusted SPECint® results



Source: <http://preshing.com/20120208/a-look-back-at-single-threaded-cpu-performance/>

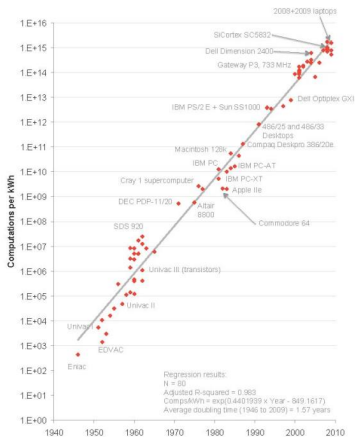
Moore's Law



Source: "What lies beneath? 50 years of enabling Moore's Law", Mike Czerniak.

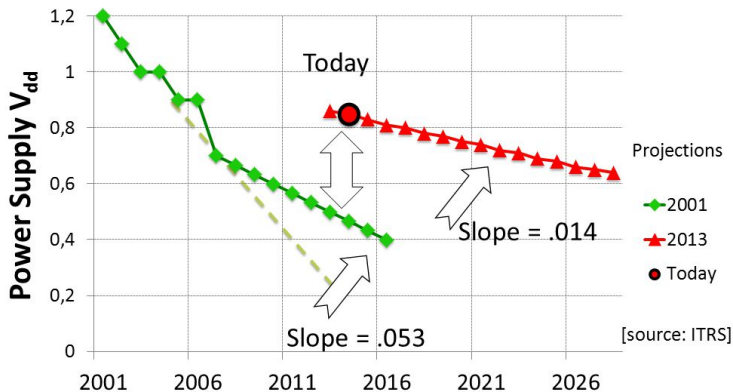
Ref: "Cramming More Components onto Integrated Circuits", Gordon Moore.

Koomey's Law



Source: Jonathan Koomey, <http://www.koomey.com/post/14466436072>

End of Dennard Scaling



The fundamental energy silver bullet is

Source: Babak Falsafi, "Heterogeneous Memory & Its Impact on Rack-Scale Computing", EcoCloud project presentation.

Architecture & Technology

- ▶ A lot of performance improvements over 40 years have been due to technology
- ▶ Mainly due to smaller, faster circuits (Dennard scaling, frequency scaling)
- ▶ But it isn't that simple e.g.
 - ▶ CPU speed increased more than $1000\times$
 - ▶ Main memory speed less than $10\times$
 - ▶ **Memory wall**

An Introduction to Caches

COMP 252 - Lecture 1

Antoniu Pop

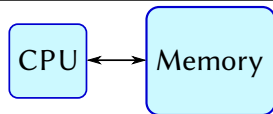
antoniu.pop@manchester.ac.uk

Processor Cache Memory

- ▶ A very important technique in processors since about mid 1980s
- ▶ Purpose is to overcome the speed imbalance between fast internal processor circuitry
e.g. ALU & registers (fast) vs. main memory (slow)
- ▶ No modern processor could perform anywhere near current speeds without caches

Understand Caches: prerequisites

- ▶ CPU connected to memory



- ▶ CPU fetches a sequence of instructions from memory and executes them
- ▶ Some instructions are loads or stores which read or write values from/to memory addresses

What is a Cache?

- ▶ Cache: “A secret hiding place”
- ▶ General principle
 - ▶ If something is far away and/or takes a long time to access, keep a local copy
 - ▶ Usually limited, but fast, local space
- ▶ Not just processors
 - ▶ Web pages kept on local disc
 - ▶ Virtual Memory is a form of caching
 - ▶ Web Content Delivery Networks (e.g. akamai.com)

What is a Processor Cache?

- ▶ Small amount of very fast memory used as temporary store for frequently used memory locations (both instructions and data)
- ▶ Relies on locality
 - ▶ during any short period of time, a program uses only a small subset (working set) of its instructions and data.
- ▶ Locality: **space** and **time**
 - ▶ Spatial locality: data located close to recently accessed addresses is likely to be used as well
 - ▶ Temporal locality: data used recently is likely to be re-used

Processor Cache Memory

- ▶ Example: 2.8 GHz processor
 - ▶ 32KB L1 data cache and 32KB L1 instruction cache
 - ▶ 256KB on-chip L2 cache (L2 cache is 4 way set associative)

- ▶ What does it mean?

- ▶ Why is it there?

- ▶ What is “good”?

Why is Cache Needed?

- ▶ Modern processor speed: above 1GHz
- ▶ More than 1 instruction / nsec (10^{-9} sec)
- ▶ Every instruction needs to be fetched from main memory
- ▶ Many instructions (1 in 3?) access main memory (R/W data)
- ▶ But RAM memory access time typically more than 50 nsec!

67× too slow!

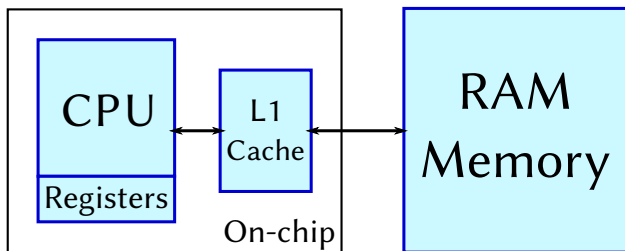
Facts about memory speeds

- ▶ Circuit capacitance is the thing that makes things slow (needs charging)
- ▶ Bigger things have bigger capacitance
- ▶ So large memories are slow
- ▶ Dynamic memories (storing data on capacitance) are slower than static memories (bistable circuits)

Interconnection Speeds

- ▶ External wires also have significant capacitance
- ▶ Driving signals between chips needs special high power interface circuits
- ▶ Things within a VLSI “chip” are fast – anything “off chip” is slow
- ▶ Put everything on a single chip?
 - ▶ e.g., 3D integration, memory stacked on top of the chip...

Basic Level 1 (L1) Cache



- ▶ Compiler makes best use of registers – they are the fastest
- ▶ Anything not in registers – must go (logically) to memory
- ▶ But is data (a copy!) in cache?

Cache Requirements

- ▶ Main memory is big
 - ▶ e.g. potentially 2^{32} bytes (4GB) if 32 bit address (or 2^{48} if x64)
- ▶ Cache is small (to be fast)
 - ▶ e.g. 32k bytes, can only hold a very small fraction ($\times 10^{-17}$ or $\times 10^{-33}$) of all possible data
- ▶ But cache must be able to store and retrieve any one of 2^{32} (or 2^{48}) addresses/data
 - ▶ In practice would not hold single bytes, but in principle ...)
- ▶ Special structures needed
 - ▶ Not simple memory indexed by address

Memory Hierarchy

