

Advanced Computer Architecture

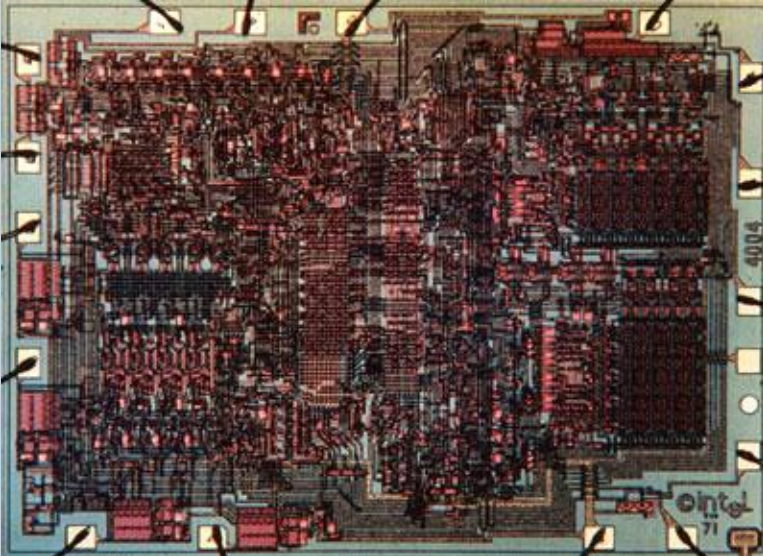
1

LECTURE 0

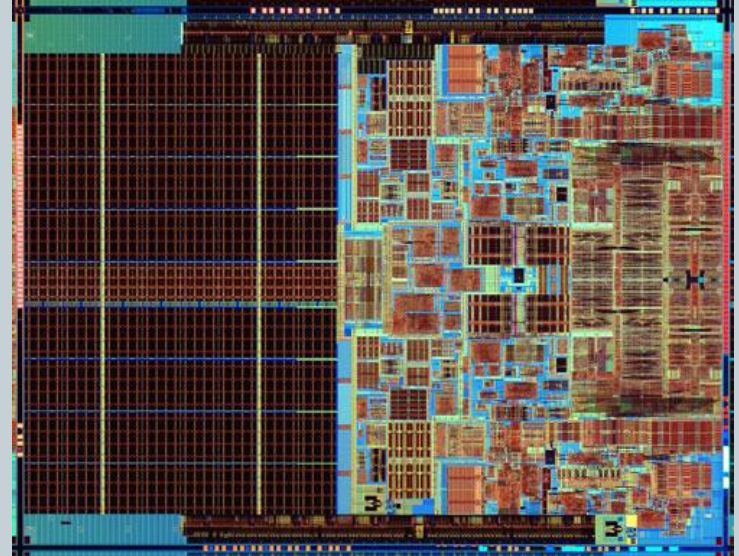
JAN LEMEIRE

Course Objectives

2



Intel 4004 – 1971 – 2.3K trans.



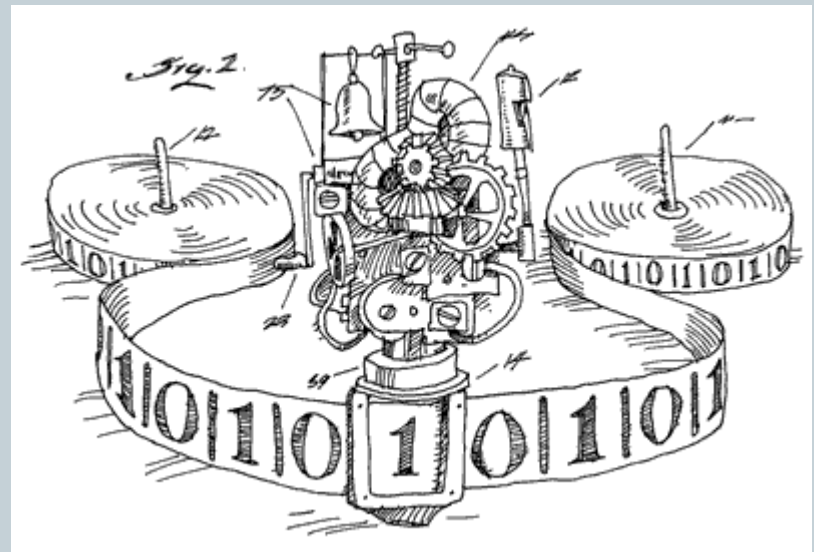
Intel Core 2 Duo – 2006 – 291M trans.

Where have all the transistors gone?

Turing Machine

3

- Universal computing machine
 - Infinite tape
 - Number of states
 - Program
 - Syntax: $\langle \text{current state} \rangle \langle \text{current symbol} \rangle \langle \text{new symbol} \rangle \langle \text{direction} \rangle \langle \text{new state} \rangle$

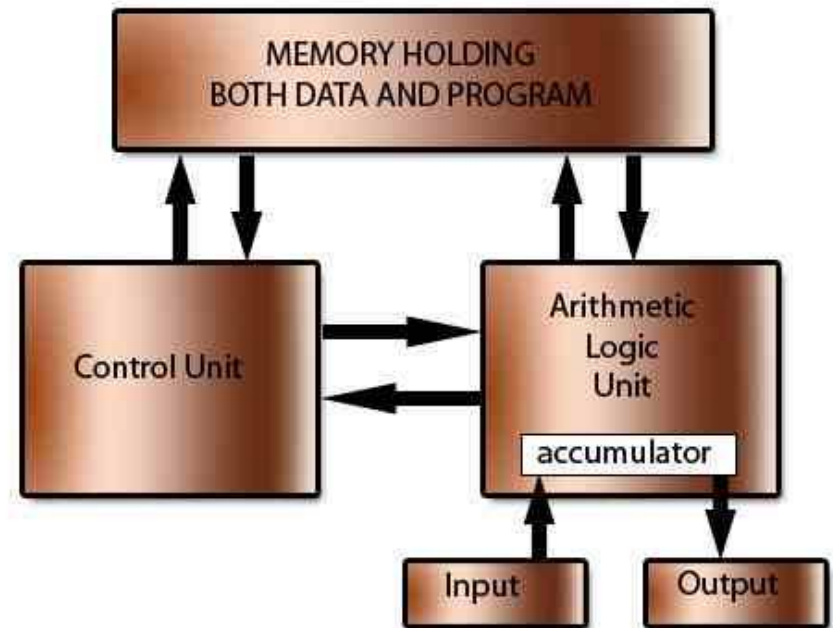


Von Neumann architecture

4

- Stored program
- Instruction sequence processed one-by-one
- ALU: basic instructions

The Von Neumann or Stored Program architecture



One goal

5

Minimize runtime

Iron law of Performance

6

- time/program =

time/cycle * cycles/instruction * instructions/program

= 1/frequency * cycles per instruction * instruction count

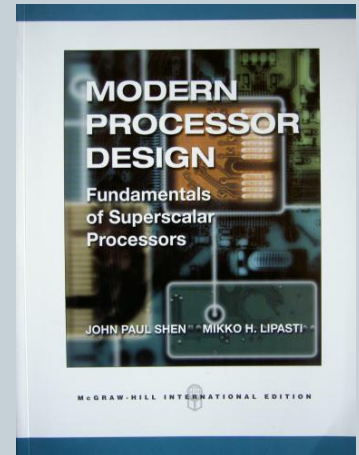
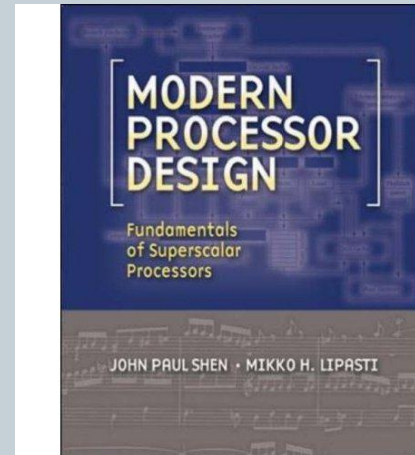
= 1/f * CPI * IC

- performance = program/time = $\frac{f * IPC}{IC}$

Course Material

7

- Modern Processor Design: Fundamentals of Modern Processor Design by Shen & Lipasti, McGraw-Hill, ISBN 0-07-056064-7, can be found at books.google.com
- Course slides / notes
- Research papers



Quote: “10GHz in 2010” (2005)

The Free Lunch is Over

8

Why You Don't Have 10GHz Today?

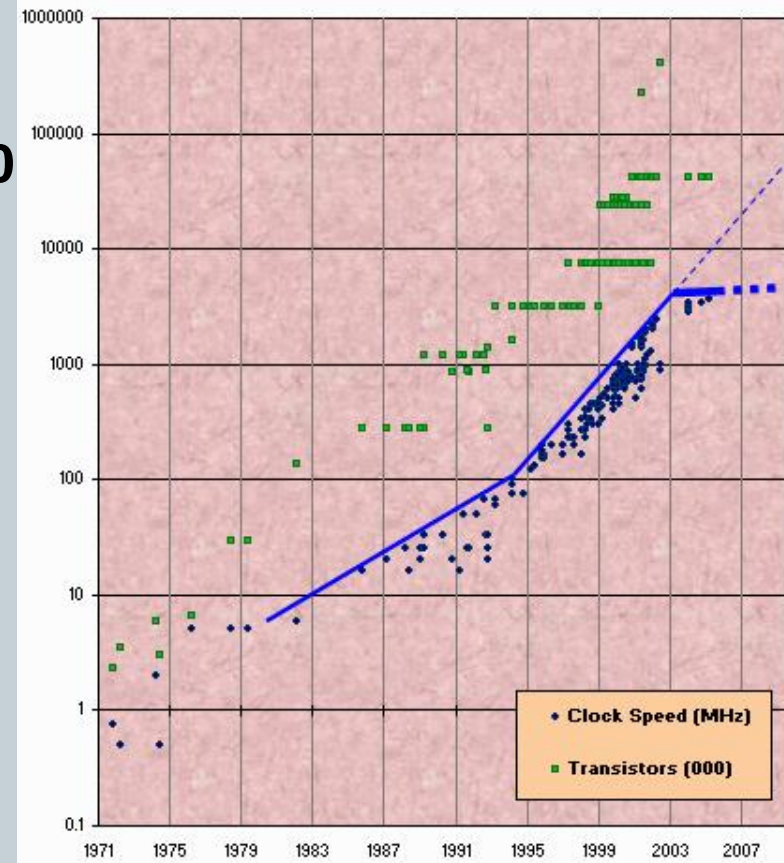
- heat/surface is the problem (power wall)
- 12 nm would mean electric paths of 10 atoms wide

Moreover:

- memory bottleneck
- instruction level parallelism (ILP) wall

What about Moore's Law?

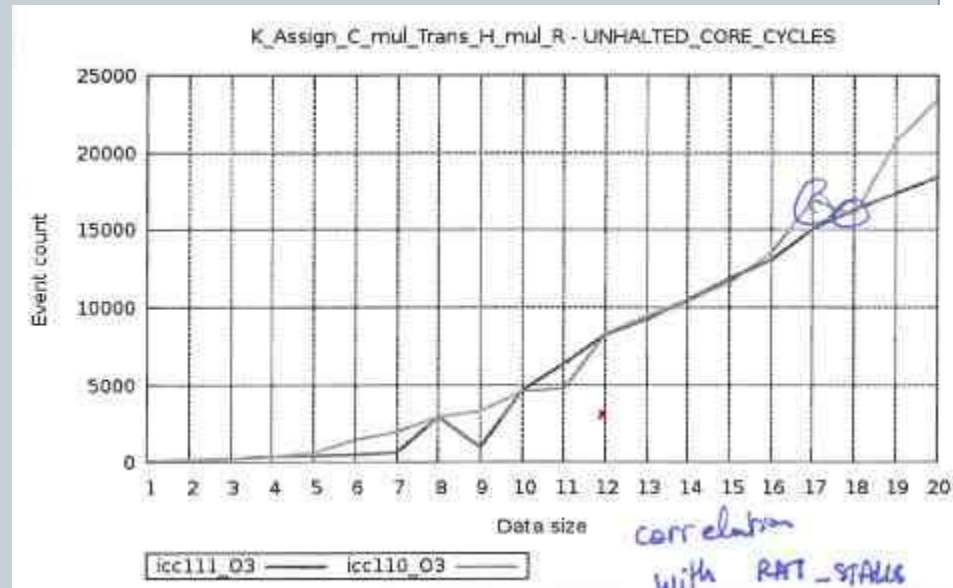
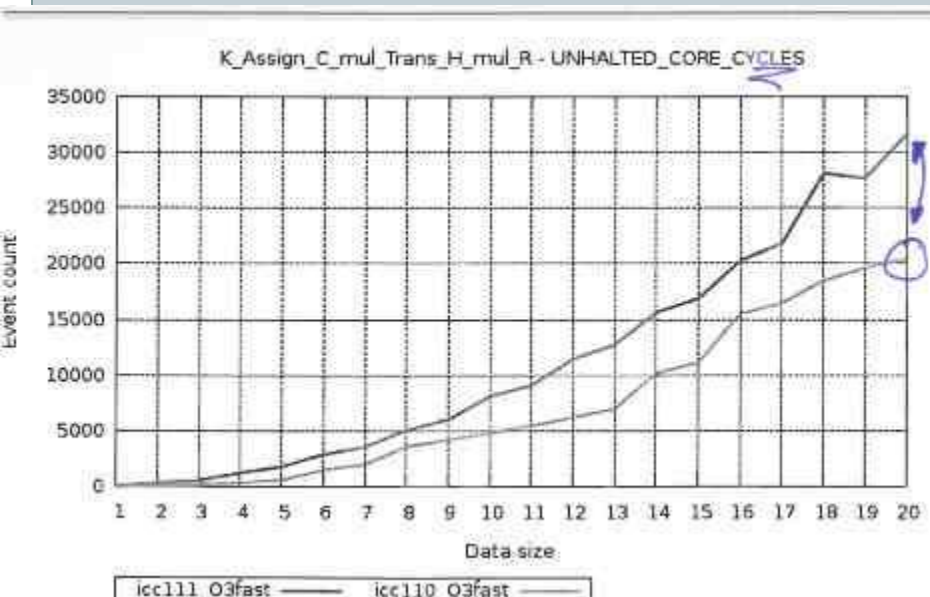
- increase of Clock speed: stopped
 - increase of Transistors: ongoing
- It's now about the number of cores!*



Programmer & compiler

9

- Instruction set of processor
- Write assembler ;-)
- Compiler optimization:



Programmer & compiler

10

- Computation:
 1. Fetch instruction
 2. Load data
 3. Perform instruction
 4. Store result
- Mission: optimize!
- Ultimate:
 - 'get' with no cost
 - Only ALU-calculations cost (the 'real' work)
 - Parallelize!

Quest for Performance

11

- **CMOS VLSI**

- Submicron feature sizes: 0.8u -> 0.6u -> 0.3u -> 0.25u -> 0.18u
- Metal layers: 3 -> 4 -> 5 -> 6 -> 7 (copper)
- Power supply voltage: 5v -> 3.3v -> 2.4v -> 1.8v

- **CAD Tools**

- Interconnect simulation and critical path analysis
- Clock signal propagation analysis
- Process simulation and yield analysis/learning

- **Microarchitecture**

- Superpipelined and superscalar machines
- Speculative and dynamic microarchitectures
- Simulation tools and emulation systems

- **Compilers**

- Extraction of instruction-level parallelism
- Aggressive and speculative code scheduling
- Object code translation and optimization

frequency

IPC

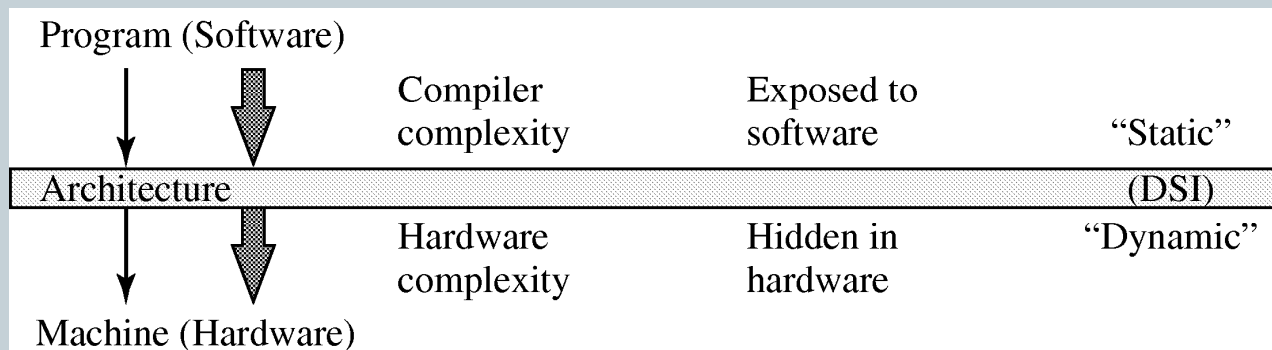
THIS COURSE

Instruction count

Static/Dynamic Interface

12

- Defines what of the micro-architecture is visible in the ISA
 - placed at best synergy between static (compiler) and dynamic (hardware) optimizations
 - lower SDI: more exposed, more optimization opportunities for compilers
 - higher SDI: easier for HW compatibility
- Depends on context (HW resources, application domain,...)



CISC > RISC > VLIW

Performance (2)

13

- **more complex instructions in ISA**
 - more computations per instruction: IC ⬇
 - higher cycle time: f ⬇
 - deeper pipelines may harm IPC: IPC ⬇
- **compiler optimizations, for example unrolling**
 - less instructions: IC ⬇
 - more cache misses: IPC ⬇
- **dynamic optimization of operations, for example dynamic instruction reuse**
 - less operations to execute: IC ⬇
 - less operations depend on each other (transitively): IPC ⬆
 - slower clock for taking decisions: f ⬇

Performance (3)

14

- reduce CPI through caching, pipelining, branch prediction, superscalar
 - higher fan-out of transistors, longer critical paths: $f \searrow$
- reduce cycle time through pipelining
 - shorter stages do less work: $f \nearrow$
 - bubbles in the pipeline (branch predictor, data dependencies, relatively slower memory): IPC \searrow
- take a look back at the end of this course

Performance and Parallelism (1)

15

- Parallelism is quest for higher IPC
 - CISC: $IPC < 1$
 - pipelined RISC: $IPC = 1$
 - superscalar RISC / VLIW : $IPC > 1$

Historical Perspective

16

70s	80s	90s	2000s
10K->100K transistors	100k -> 1 M transistors	1M - >100M trans.	1B
0.2 -> 2 MHz clock	2->20 MHz clock	20M -> 1 GHz clock	3 GHz
< 0.1 IPC	0.1 -> 0.9 IPC	0.9 -> 2.0 IPC	10 ???
<0.2 MIPS/MFLOPS	0.20 -> 20 MIPS/MFLOPS	20 -> 2K MIPS/MFLOPS	100.000
4 bit -> 8 bit microcontrollers	PCs instead of supercomputers debate on ideal ISA (compiler target and implementation) caches & pipelining	more focus on implementation branch prediction, out-of-order, superscalar, VLIWs mostly ISA-independent	multicores ASIPs (application-specific instruction-set processor)

Course Objectives

17

- to understand how we can use hundreds of millions of transistors to execute programs faster or more energy-efficient
 - design (what, how, why)
 - measurements
- to understand that “faster” essentially means “in parallel”
 - instruction-level parallelism
 - data-level parallelism
 - thread-level parallelism
- to understand why multi-cores are the credo of the industry

Why take this course?

18

- To become a computer designer
 - Alumni of this class helped design your computer
- To learn what is *under the hood* of a computer
 - Innate curiosity
 - To better understand when things break
 - To write better code/applications
 - To write better system software (O/S, compiler, etc.)
- Because it is intellectually fascinating!
 - What is the most complex man-made device?

End Note

19

- Want Performance?
- Use Multiple processors !
- “Parallel Systems” course