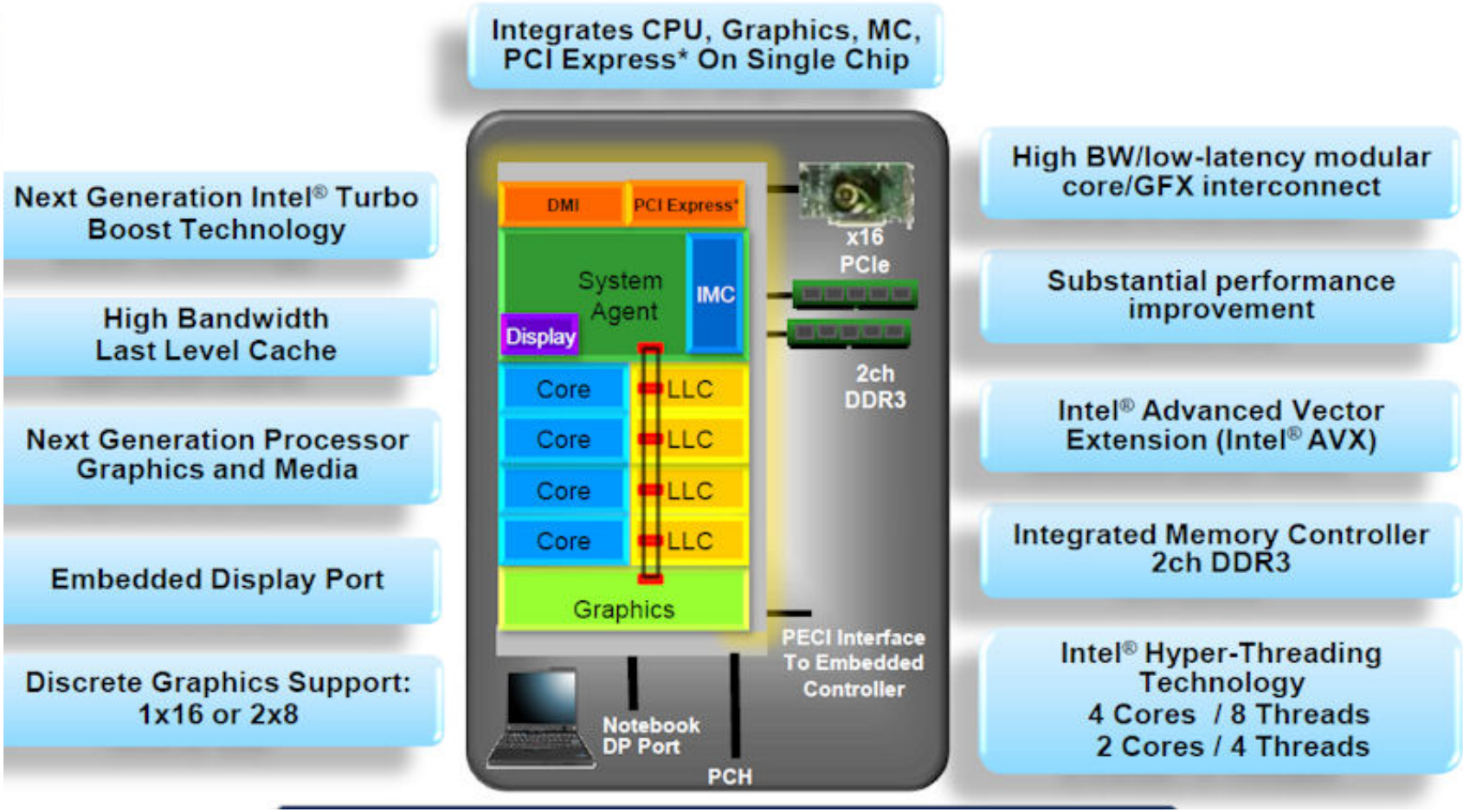
Computer Architecture

Berk Sunar Thomas Eisenbarth

WHY COMPUTER ARCHITECTURE?

Modern CPU Architecture



Checkout: http://ark.intel.com/ for full details of Intel devices

Outline

- **Administrative Details**
- Computer Architecture?
- Measuring Performance

Previous Knowledge

Expected background:

- Basic architecture
- Programming in C/C++/Java

Topics you should've seen before:

- Instruction sets, computer arithmetic, assembly programming, memory, I/O
- Pipelining, caches, virtual memory

Outline

- Administrative Details
- **Computer Architecture**
- Measuring Performance

What is Computer Architecture?

A hardware perspective:

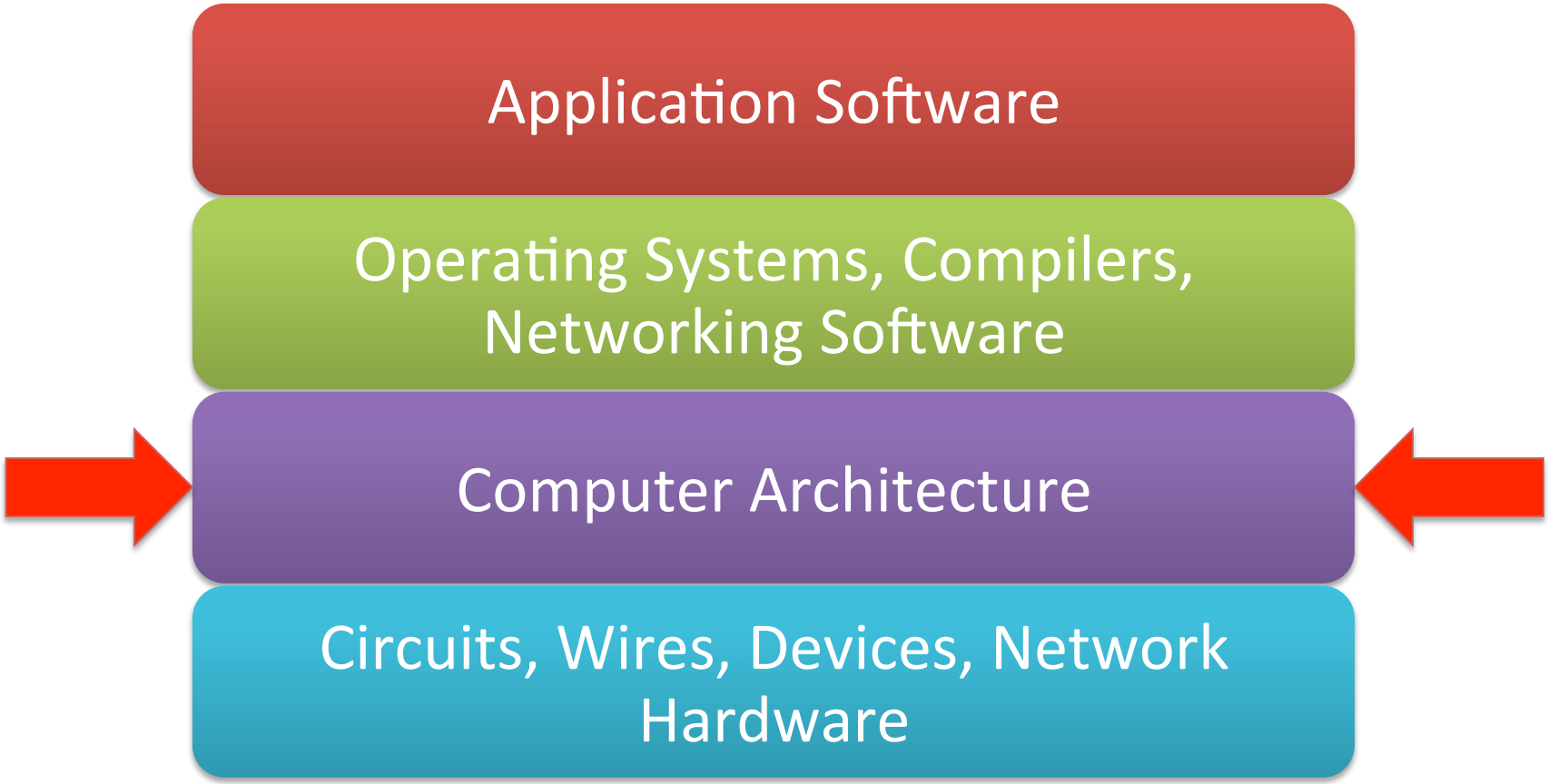
- Understand modern computer components and their interaction
- Learn to evaluate and compare computers
- Enable to make *own* design decisions

→Build your own computer

  - Instruction Set Design
  - Functional Organization
  - Logic Design
  - Implementation

Architecture and Other Disciplines

Architecture interacts with many other fields:



Levels of Computer Architecture

Architecture:

- functional appearance to software
- opcodes, addressing modes, architected registers

Microarchitecture (= focus of this course)

- logical structure that implements the architecture
- pipelining, functional units, caches, physical registers

Realization (circuits)

- physical structure that embodies the implementation
- gates, cells, transistors, wires

Why study Computer Architecture?

Requirements are always changing:

- Are computers fast enough?
  - Computation intense applications: AI, gaming, virtual reality, gaming, complex simulation, gaming, ...
- Any goals beyond speed?
  - Power: heat dissipation, battery life, utility bill
  - Cost
  - Reliability
  - …

Why study Computer Architecture?

Rapid change in technology: (approximate annual improvements)

- **IC logic Technology**
  - Transistor density +35%, die size +10-20%

  ➔together 40-55% doubling every 18 – 24 months (Moore's Law)
- **Memory**
  - DRAM (memory): density +25-40%
  - FLASH (memory): density +50-60%, cost >15x cheaper than DRAM
  - Magnetic disk (mem): density +40%, cost >20x cheaper than FLASH

➔ Parameters change *and* change relative to one another!

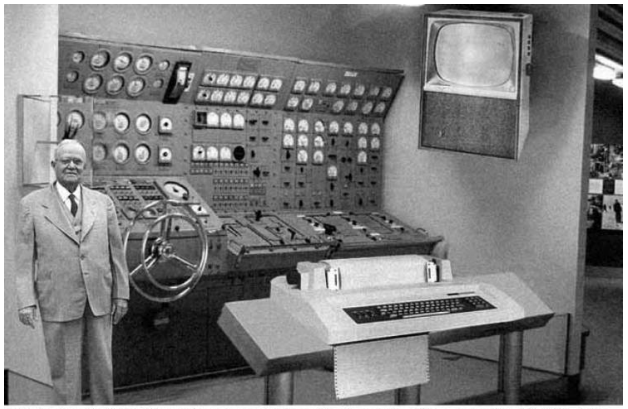
➔ Not even including technology jumps like nanotechnologies

➔ Designs change even if requirements fixed

➔ Computer architect must plan ahead for changing tech

Computing: Yesterday vs. Today

60's:
one Computer, many users



Scientists from the RAND Corporation have created this model to illustrate how a "home computer" could look like in the year 2004. However the needed technology will not be economically feasible for the average home. Also the scientists readily admit that the computer will require not yet invented technology to actually work, but 50 years from now scientific progress is expected to solve these problems. With teletype interface and the Fortran language, the computer will be easy to use.

Y2K:
one Computer per user

TODAY:
1 user, countless computers

Two trends:

- Internet of Things
- Cloud Computing

Moore's Law

Cramming More Components onto Integrated Circuits

[G.E. Moore, Electronics, 1965]

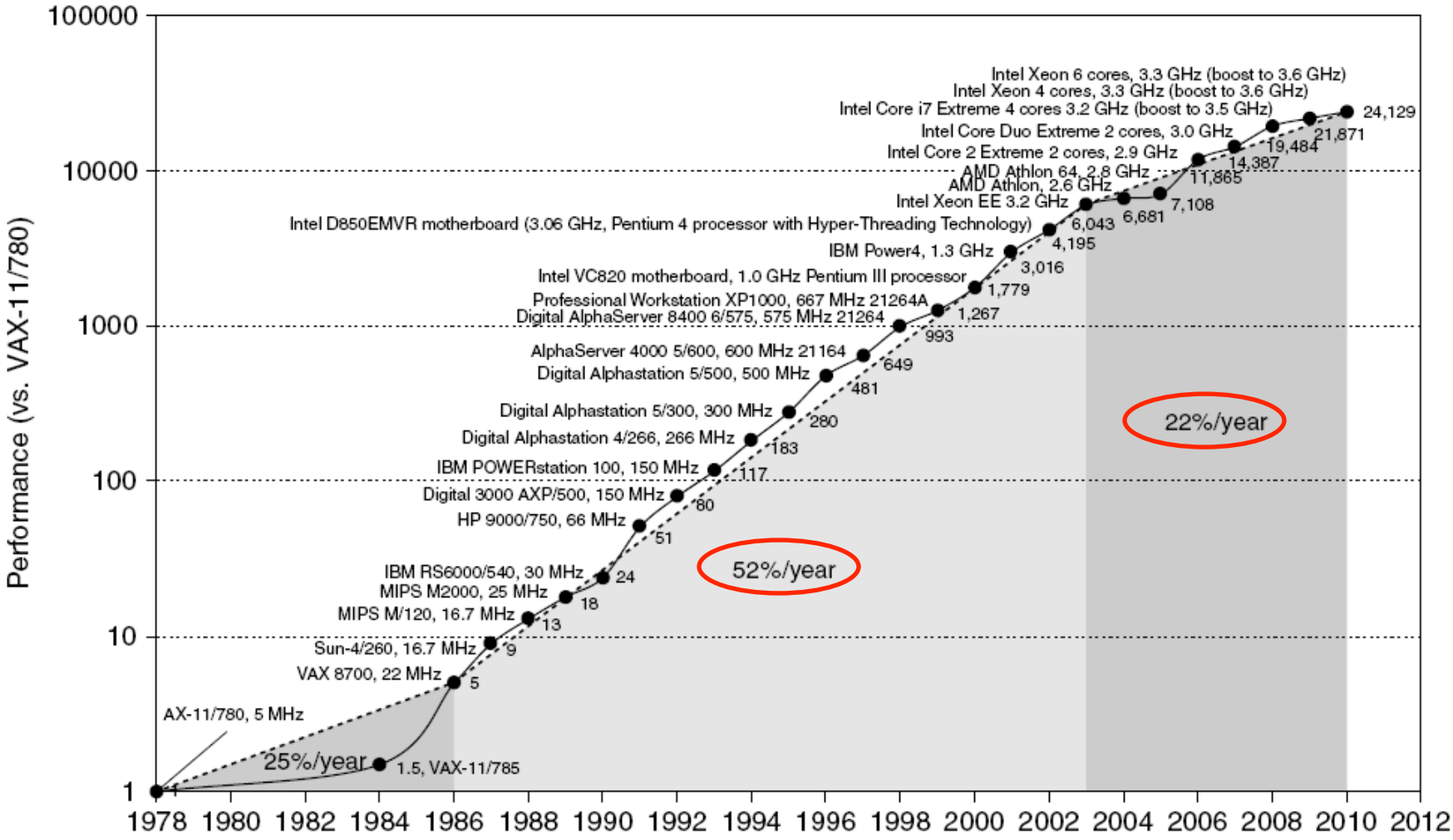
Observation:

- (DRAM) transistor density doubles annually (slightly off: doubles every 18 months)

Corollaries:

- Cost per transistor halves annually (18 months)
- Power per transistor decreases with scaling
- Speed increases with scaling
- Reliability starting to decrease with scaling

Growth in Processor Performance



Some examples of Intel Processor Generations

Year	Processor	transistors	Clock rate (MHz)	Performance in MIPS
1982	I 80286	134k	12.5	2
1985	I 80386	275k	16	6
1989	I 80486	1.2M	25	25
1993	I Pentium	3.1M	66	132
1997	Pentium Pro	5.5M	200	600
2001	Pentium 4	42M	1500	4500
2010	I Core i7	1.17B	3333	50000

→Performance improved by 25,000X

Power Consumption of CMOS

- Information stored as voltage levels –Hi =1/Lo=0
- Energy cost for a single switching event:

  Energy: $E_{dyn} = \frac{1}{2} \cdot C_{ld} \cdot V^2$
- Signal transitions dissipate power:

$$P = \underbrace{\alpha \cdot C_{ld} \cdot V^2 \cdot f}_{dynamic} + \underbrace{V \cdot I_{leak}}_{static}$$

Activity factor α is determined by data

- Power and energy can both serve as a metric.

Reducing Power Consumption

Problem:

Heat Dissipation: IC size is 1.5 $cm\hat{}2$ (@80W for I7)

→Freq. increase of 30% ended in 2003

Energy Efficiency improvements:

- Reduce Voltage (increases static power)
  - E.g. DVFS (Dynamic Voltage Frequency Scaling)
- Adjust clock frequency
- Turn off:
  - Quick power down/ race to halt
  - Stop clock (clock gating) or Power down (power gating) unused components

Better Metric: Energy

Example: race to halt

- Processor A needs 100% of power for 100% of run time
- Processor B needs 120% of power for 70% of run time

Q: Which processor is more energy efficient?

Answer:

Energy $= power \times run\ time$

- $E{\downarrow}A = 1 \times 1$
- $E{\downarrow}B = 1.2 \times .7 = .84$

➔B needs only 84% of the energy of A!

Outline

- Administrative Details
- Computer Architecture
- **Measuring Performance**

Performance Measurement

Much of the focus of this course is on improving performance

Topics:

- performance metrics
- CPU performance equation
- benchmarks and benchmarking
- reporting averages
- Amdahl's Law
- Little's Law
- concepts
  - balance
  - tradeoffs
  - bursty behavior (average and peak performance)

Performance Metrics

When is a computer fast?

Latency: response time, execution time

- good metric for fixed amount of work (minimize time)

Throughput: bandwidth, work per time

- = (1 / latency) when there is NO OVERLAP
- > (1 / latency) when there is overlap
- in real processors, there is always overlap (e.g., pipelining)
- good metric for fixed amount of time (maximize work)

Measuring Performance:

- Execution time: t measured as:
  - wall time/response time (includes I/O)or
  - CPU time
- Performance: Perf = $1/t$

Next time:

More on:

Measuring Performance