

Chapter 1

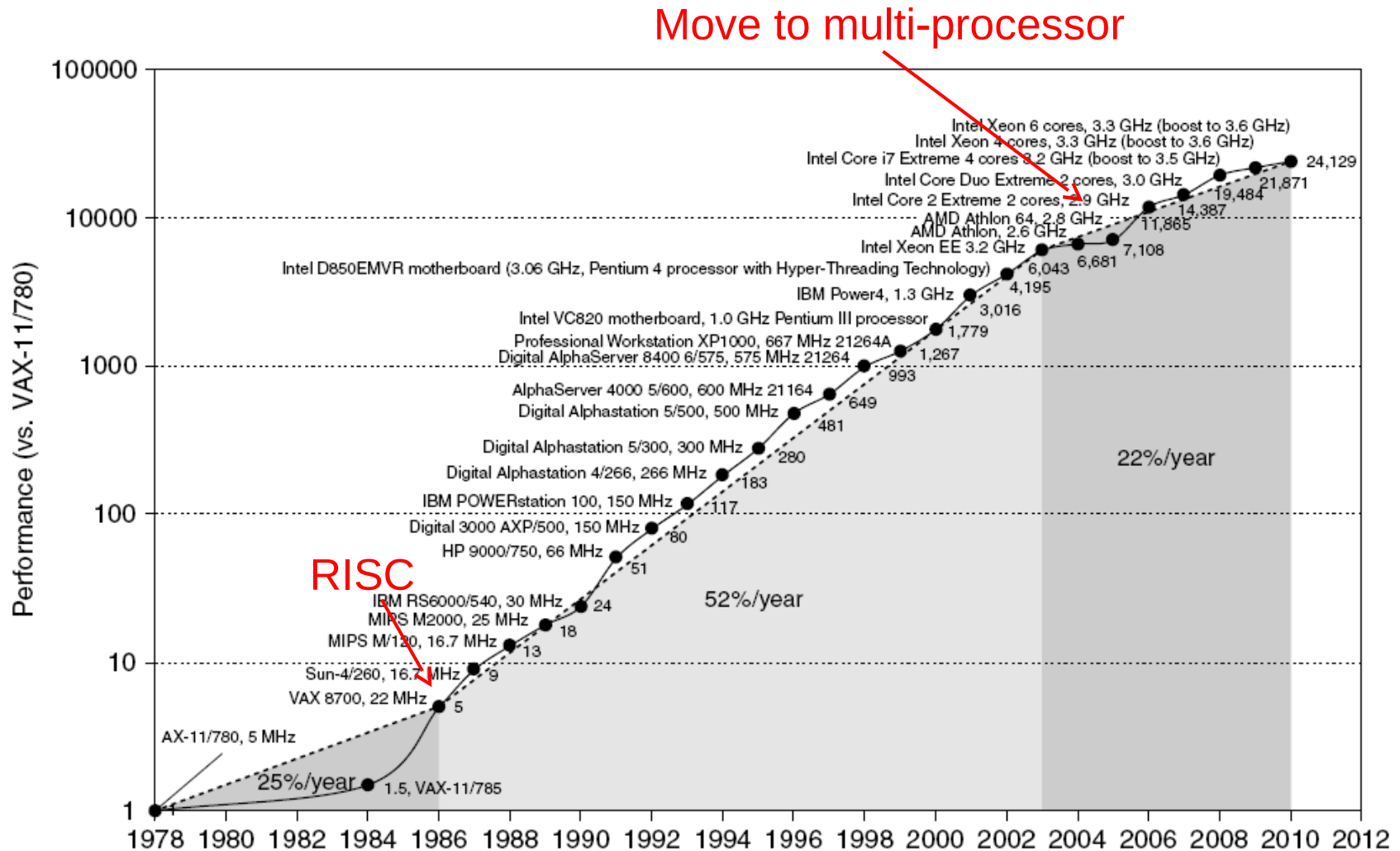
Fundamentals of Quantitative Design and Analysis

Computer Technology

- Performance improvements:
 - Improvements in semiconductor technology
 - Feature size, clock speed
 - Improvements in computer architectures
 - Enabled by HLL compilers, UNIX
 - Lead to RISC architectures

- Together have enabled:
 - Lightweight computers
 - Productivity-based managed/interpreted programming languages

Single Processor Performance



Current Trends in Architecture

- Cannot continue to leverage Instruction-Level parallelism (ILP)
 - Single processor performance improvement ended in 2003
- New models for performance:
 - Data-level parallelism (DLP)
 - Thread-level parallelism (TLP)
 - Request-level parallelism (RLP)
- These require explicit restructuring of the application

Classes of Computers

- Personal Mobile Device (PMD)
 - e.g. smart phones, tablet computers
 - Emphasis on energy efficiency and real-time
- Desktop Computing
 - Emphasis on price-performance
- Servers
 - Emphasis on availability, scalability, throughput
- Clusters / Warehouse Scale Computers
 - Used for “Software as a Service (SaaS)”
 - Emphasis on availability and price-performance
 - Sub-class: Supercomputers, emphasis: floating-point performance and fast internal networks
- Embedded Computers
 - Emphasis: price

Parallelism

- Classes of parallelism in applications:
 - Data-Level Parallelism (DLP)
 - Task-Level Parallelism (TLP)

- Classes of architectural parallelism:
 - Instruction-Level Parallelism (ILP)
 - Vector architectures/Graphic Processor Units (GPUs)
 - Thread-Level Parallelism
 - Request-Level Parallelism

Flynn's Taxonomy

- Single instruction stream, single data stream (SISD)
- Single instruction stream, multiple data streams (SIMD)
 - Vector architectures
 - Multimedia extensions
 - Graphics processor units
- Multiple instruction streams, single data stream (MISD)
 - No commercial implementation
- Multiple instruction streams, multiple data streams (MIMD)
 - Tightly-coupled MIMD
 - Loosely-coupled MIMD

Defining Computer Architecture

- “Old” view of computer architecture:
 - Instruction Set Architecture (ISA) design
 - i.e. decisions regarding:
 - registers, memory addressing, addressing modes, instruction operands, available operations, control flow instructions, instruction encoding
- “Real” computer architecture:
 - Specific requirements of the target machine
 - Design to maximize performance within constraints: cost, power, and availability
 - Includes ISA, microarchitecture, hardware

Trends in Technology

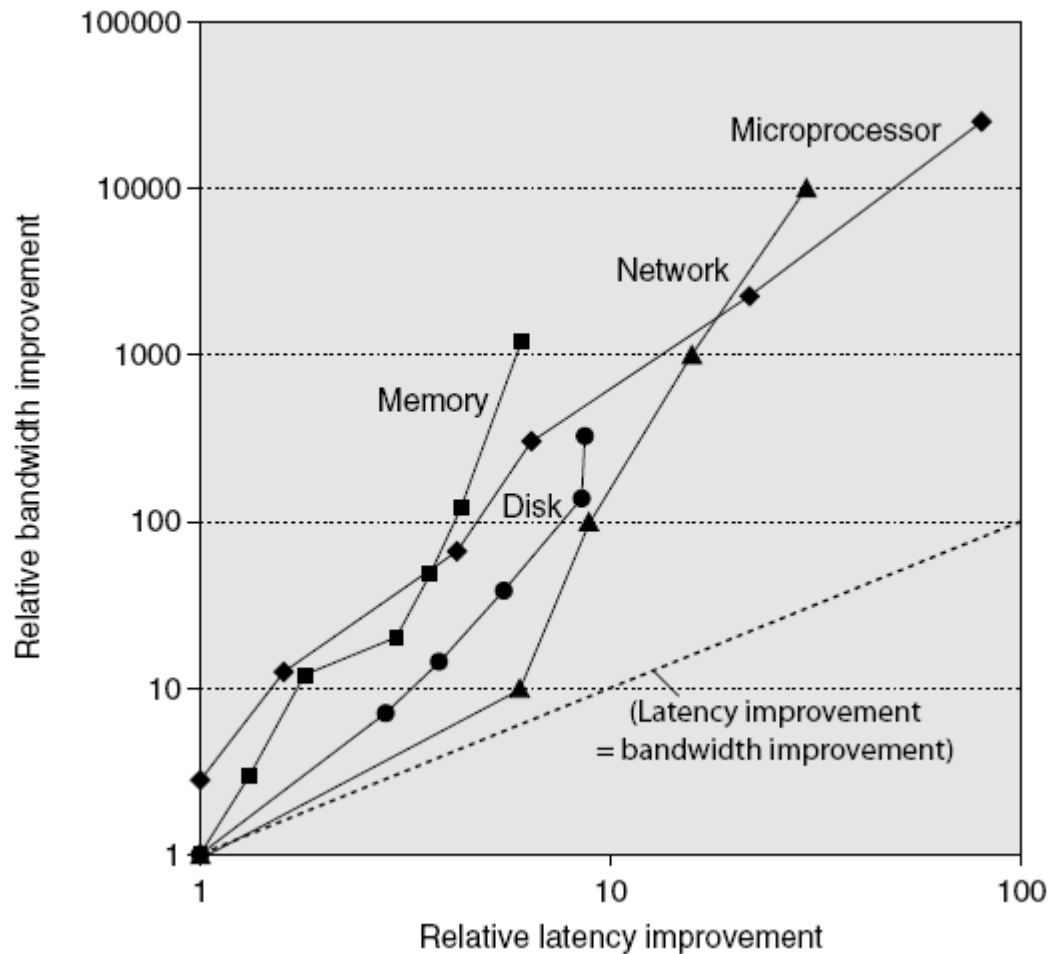
- Integrated circuit technology
 - Transistor density: 35%/year
 - Die size: 10-20%/year
 - Integration overall: 40-55%/year
- DRAM capacity: 25-40%/year (slowing)
- Flash capacity: 50-60%/year
 - 15-20X cheaper/bit than DRAM
- Magnetic disk technology: 40%/year
 - 15-25X cheaper/bit than Flash
 - 300-500X cheaper/bit than DRAM

Bandwidth and Latency

- Bandwidth or throughput
 - Total work done in a given time
 - 10,000-25,000X improvement for processors
 - 300-1200X improvement for memory and disks

- Latency or response time
 - Time between start and completion of an event
 - 30-80X improvement for processors
 - 6-8X improvement for memory and disks

Bandwidth and Latency



Log-log plot of bandwidth and latency milestones

Transistors and Wires

- Feature size
 - Minimum size of transistor or wire in x or y dimension
 - 10 microns in 1971 to .032 microns in 2011
 - Transistor performance scales linearly
 - Wire delay does not improve with feature size!
 - Integration density scales quadratically

Power and Energy

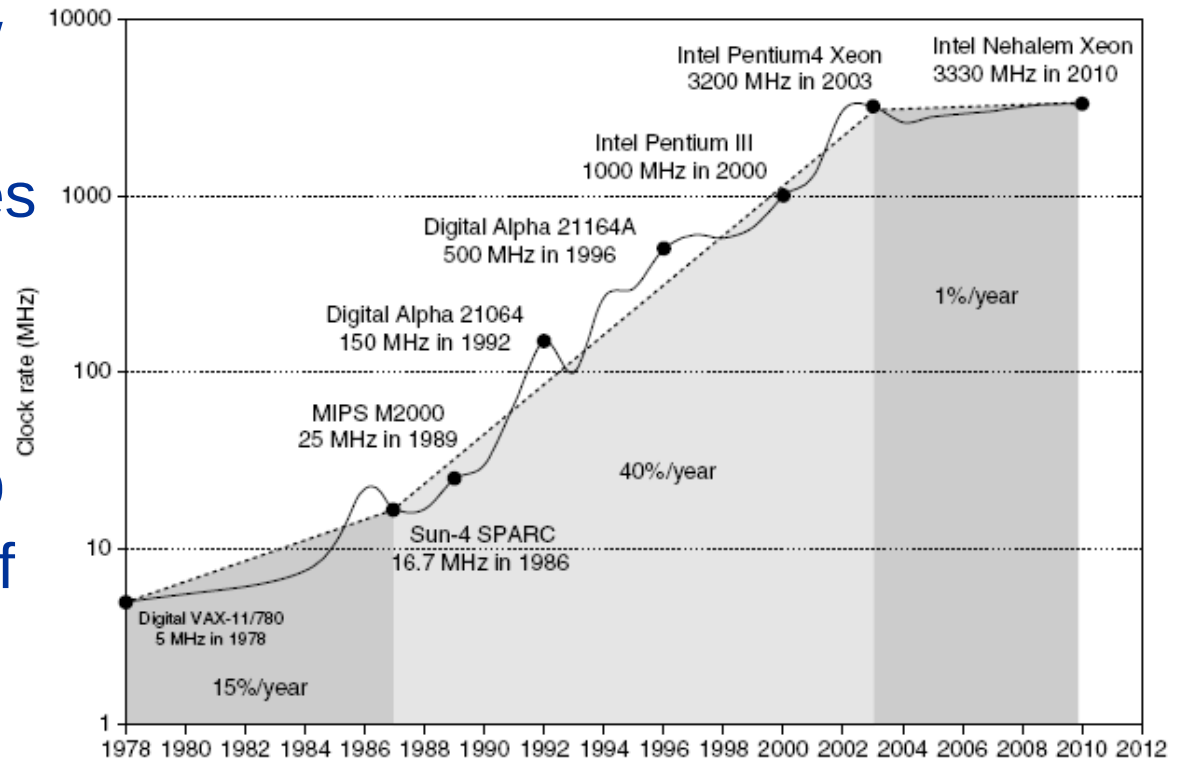
- Problem: Get power in, get power out
- Thermal Design Power (TDP)
 - Characterizes sustained power consumption
 - Used as target for power supply and cooling system
 - Lower than peak power, higher than average power consumption
- Clock rate can be reduced dynamically to limit power consumption
- Energy per task is often a better measurement

Dynamic Energy and Power

- Dynamic energy
 - Transistor switch from 0 -> 1 or 1 -> 0
 - $\frac{1}{2} \times \text{Capacitive load} \times \text{Voltage}^2$
- Dynamic power
 - $\frac{1}{2} \times \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency switched}$
- Reducing clock rate reduces power, not energy

Power

- Intel 80386 consumed ~ 2 W
- 3.3 GHz Intel Core i7 consumes 130 W
- Heat must be dissipated from 1.5 x 1.5 cm chip
- This is the limit of what can be cooled by air



Reducing Power

- Techniques for reducing power:
 - Do nothing well
 - Dynamic Voltage-Frequency Scaling
 - Low power state for DRAM, disks
 - Overclocking, turning off cores

Static Power

- Static power consumption
 - $\text{Current}_{\text{static}} \times \text{Voltage}$
 - Scales with number of transistors
 - To reduce: power gating

Trends in Cost

- Cost driven down by learning curve
 - Yield
- DRAM: price closely tracks cost
- Microprocessors: price depends on volume
 - 10% less for each doubling of volume

Integrated Circuit Cost

- Integrated circuit

$$\text{Cost of integrated circuit} = \frac{\text{Cost of die} + \text{Cost of testing die} + \text{Cost of packaging and final test}}{\text{Final test yield}}$$

$$\text{Cost of die} = \frac{\text{Cost of wafer}}{\text{Dies per wafer} \times \text{Die yield}}$$

$$\text{Dies per wafer} = \frac{\pi \times (\text{Wafer diameter}/2)^2}{\text{Die area}} - \frac{\pi \times \text{Wafer diameter}}{\sqrt{2} \times \text{Die area}}$$

- Bose-Einstein formula:

$$\text{Die yield} = \text{Wafer yield} \times 1 / (1 + \text{Defects per unit area} \times \text{Die area})^N$$

- Defects per unit area = 0.016-0.057 defects per square cm (2010)
- N = process-complexity factor = 11.5-15.5 (40 nm, 2010)

Dependability

- Module reliability
 - Mean time to failure (MTTF)
 - Mean time to repair (MTTR)
 - Mean time between failures (MTBF) = $MTTF + MTTR$
 - Availability = $MTTF / MTBF$

Measuring Performance

- Typical performance metrics:
 - Response time
 - Throughput
- Speedup of X relative to Y
 - $\text{Execution time}_Y / \text{Execution time}_X$
- Execution time
 - Wall clock time: includes all system overheads
 - CPU time: only computation time
- Benchmarks
 - Kernels (e.g. matrix multiply)
 - Toy programs (e.g. sorting)
 - Synthetic benchmarks (e.g. Dhrystone)
 - Benchmark suites (e.g. SPEC06fp, TPC-C)

Principles of Computer Design

- Take Advantage of Parallelism
 - e.g. multiple processors, disks, memory banks, pipelining, multiple functional units
- Principle of Locality
 - Reuse of data and instructions
- Focus on the Common Case

- **Amdahl's Law**

$$\text{Execution time}_{\text{new}} = \text{Execution time}_{\text{old}} \times \left((1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}} \right)$$

$$\text{Speedup}_{\text{overall}} = \frac{\text{Execution time}_{\text{old}}}{\text{Execution time}_{\text{new}}} = \frac{1}{(1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

Principles of Computer Design

■ The Processor Performance Equation

CPU time = CPU clock cycles for a program \times Clock cycle time

$$\text{CPU time} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

$$\text{CPI} = \frac{\text{CPU clock cycles for a program}}{\text{Instruction count}}$$

CPU time = Instruction count \times Cycles per instruction \times Clock cycle time

$$\frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}} = \frac{\text{Seconds}}{\text{Program}} = \text{CPU time}$$

Principles of Computer Design

- Different instruction types having different CPIs

$$\text{CPU clock cycles} = \sum_{i=1}^n \text{IC}_i \times \text{CPI}_i$$

$$\text{CPU time} = \left(\sum_{i=1}^n \text{IC}_i \times \text{CPI}_i \right) \times \text{Clock cycle time}$$