COM304 COMPUTER ARCHITECTURE

3 February 2020

Dr Noor Mahammad Sk

COM304 Course Evaluation

- Pre End Semester: 40% weightage
  - Quiz 1 – 20% weightage
  - Quiz 2 – 20% weightage
- End semester: 60% Weightage
  - End Sem Exam – 60% weightage
- Solution to Open Challenges will be graded ‘S’

Syllabus

- Fundamentals of computer design: Classes of computers, trends in technology, measurement of performance of a computer system, current issues in design of functional components of a computer system – Processor unit, memory unit, and secondary storage unit; Hardware/software tradeoff in computer design
- Fundamentals of processor design: Instruction set processor design, exploitation of instruction level parallelism, processor micro architecture, performance of a processor
- Pipelined processor architecture: Fundamentals of pipelining, arithmetic pipeline design – Carry look ahead adder, Wallace tree multiplier, Floating–point adder/subtractor;
- Instruction pipeline design; Balancing pipeline stages; Stalls in a pipeline; Methods for reductions of stalls in a pipelined processor

Syllabus Contd.,

- Superscalar processor architecture: Limitations of scalar pipelines, superscalar pipelines, dynamic exploitation of instruction–Level parallelism, register dataflow techniques, memory dataflow techniques, Instruction flow techniques, case studies of superscalar processors
- Advanced processor architectures: Multithreaded processors, multi core processors, reconfigurable instruction set processors
- Storage system architectures: RAID architecture, storage area networks, Network attached storage
- Large computer system architectures: Symmetric multiprocessor systems – Shared memory systems and shared bus architectures; cache coherency protocols – MESI protocol and coherence in multi–level cache systems; Internetwork architectures – Directory protocol for cache coherence

Reference Books

- Hennessy J.H and Patterson D.A, Computer Architecture – A Quantitative Approach, Morgan Kaufmann, 2003.
- Shen J.P and Lipasti M.H, Modern Processor Design – Fundamentals of Superscalar Processors, Tata McGraw Hill, 2003.

Class Timings

- 100% Attendance must

DAY	Time
Monday	*09:00 - 09:50*
Tuesday	*08:00 - 08:50*
Friday	*10:00 - 10:50*

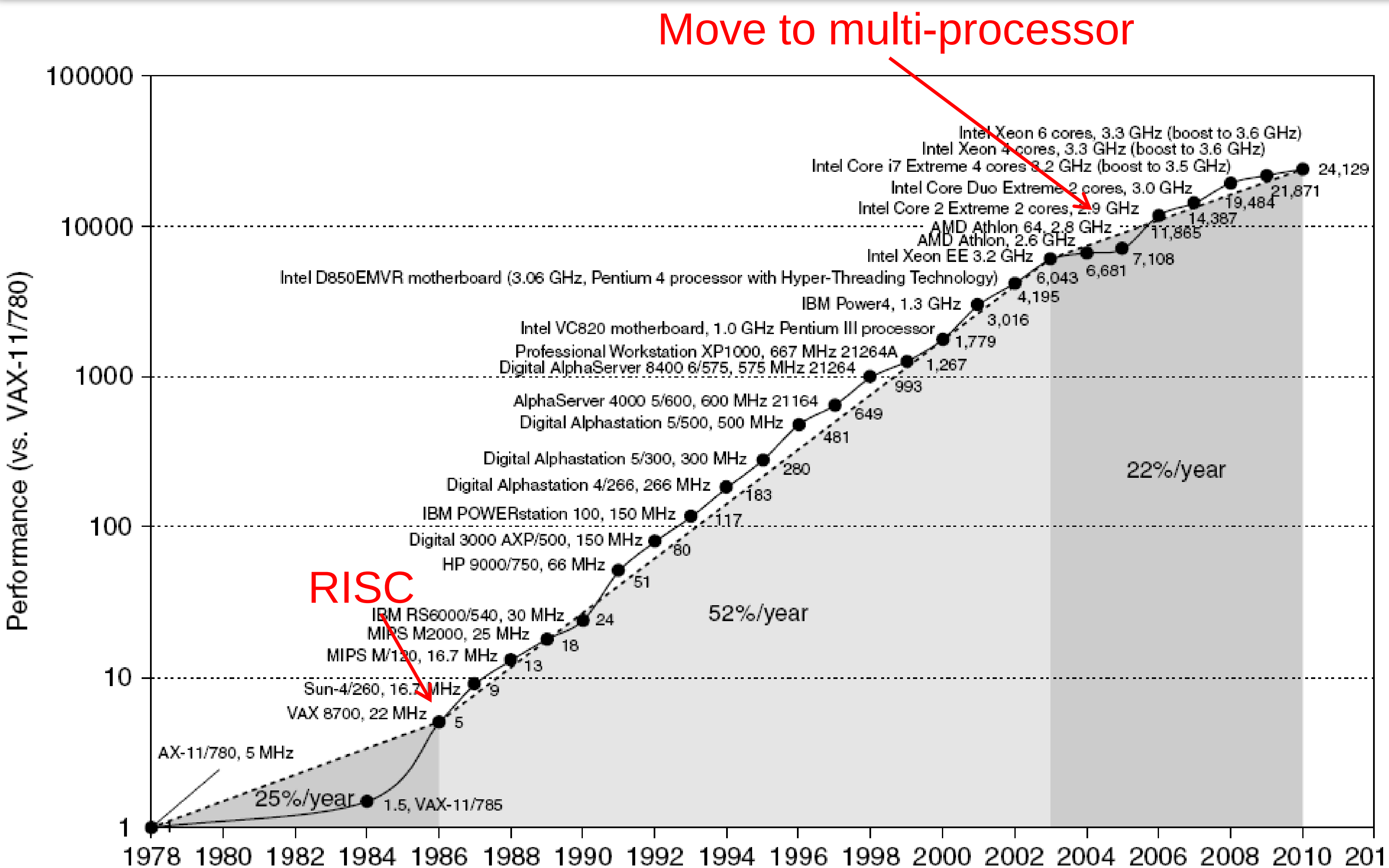
7

Fundamentals of Computer Design

Computer Technology

- Performance improvements:
  - Improvements in semiconductor technology:
    - Feature size, clock speed
  - Improvements in computer architecture
    - 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 Devices (PMD)
  - e.g., smart phones, tablet computers
  - Emphasis on energy efficient and real-time
- Desktop Computing
  - Emphasis on price-performance
- Servers
  - Emphasis on availability, scalability and throughput
- Clusters/warehouse scale computers
  - Used for "Software as a Service (SaaS)"
  - Emphasis on availability and price-performance
  - Subclass: Super computers
    - 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 architecture/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 instruction, instruction encoding
- "Real" computer architecture:
  - Specific requirements of the target machine
  - Design to maximize performance within constraints:
    - Cost, power and availability
  - Includes ISA, micro-architecture, hardware

Trends in Technology

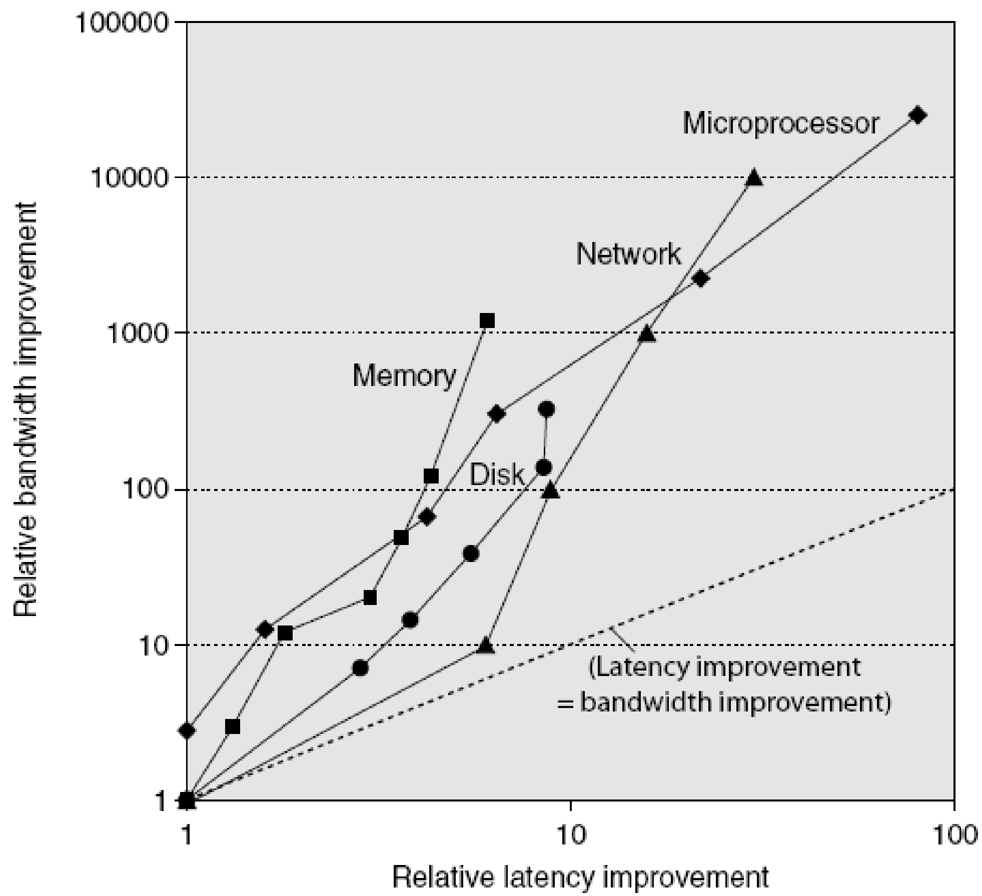
- Integrated circuit technology
  - Transistor density: 35%/year
  - Die size: 10-20%/year
  - Integration overall: 40 – 55%/year
- DRAM capacity: 25–40 % (slowing)
- Flash capacity: 50-60%/year
  - 15 – 20x cheaper/bit than DRAM
- Magnetic disk technology: 40%/year
  - 15 – 25x cheaper/bit than Flash
  - 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 micros in 1971 to 0.032 microns in 2011
  - Transistor performance scales linearly
    - Wire delay does not improve with feature size!
  - Integration density scales quadratically

Power and Energy

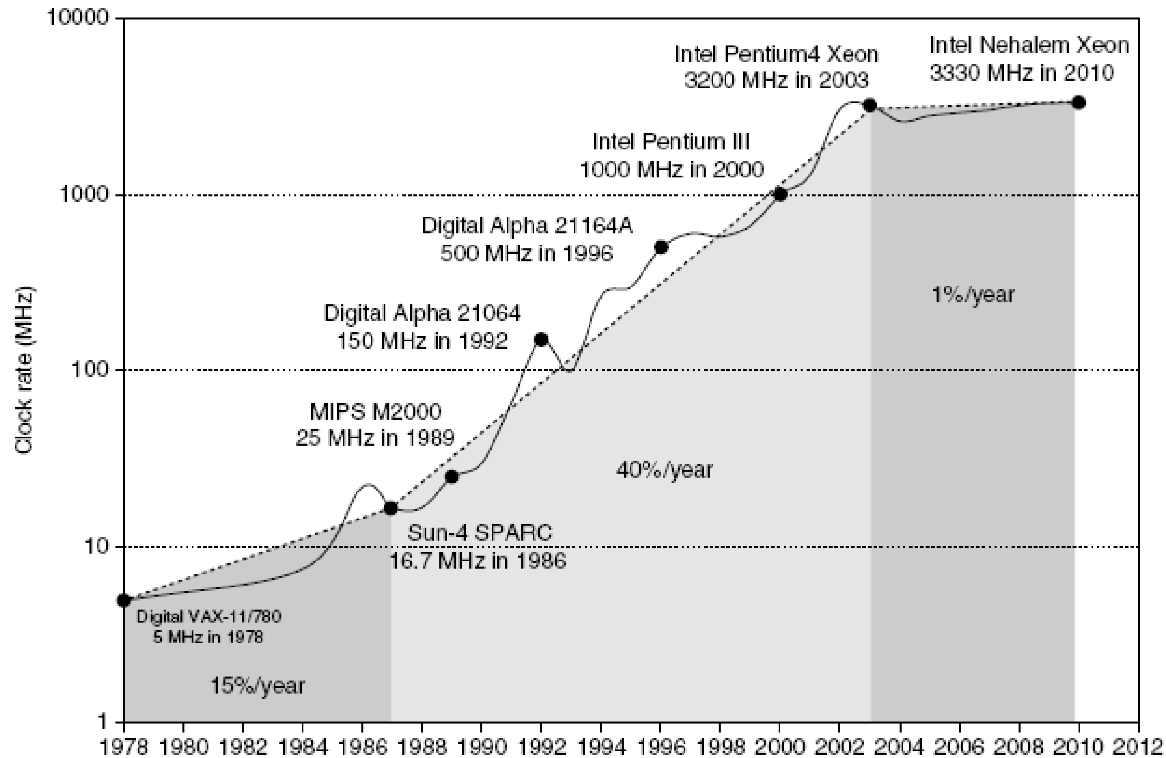
- Problem: Get power in, Get power out
- Thermal Design Power (TDP)
  - Characterizes sustained power consumption
  - Used as target for power supply and cooling system
  - Lower than peak power, higher than average power consumption
- Clock rate can be reduced dynamically to limit power consumption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

Dynamic Energy and Power

- Dynamic energy
  - Transistor switch from 0 → 1 or 1 → 0
  - ½ x Capacitive load x Voltage^2
- Dynamic power
  - ½ x Capacitive load x Voltage^2 x 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:
  - Dynamic voltage frequency scaling
  - Low power state for DRAM, disks
  - Overclocking, turning off cores

Static Power

- Static power consumption
  - $\text{Current}_{\text{static}}$ x 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

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

$$\text{Cost of die} = \frac{\text{Cost of Wafer}}{\text{Die per wafer x Die yield}}$$

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

- 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 (40mm, 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
  - Execution time_Y / 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

$$\text{CPU time} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$$

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

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

$$\text{CPU time} = \text{Instruction count} \times \text{Cycles per instruction} \times \text{Clock cycle time}$$

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

Principles of Computer Design

- Different instruction types having different CPIs

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

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