UTM
UNIVERSITI TEKNOLOGI MALAYSIA
RESEARCH UNIVERSITY

Lecture 1: Course Intro and Computer Performance

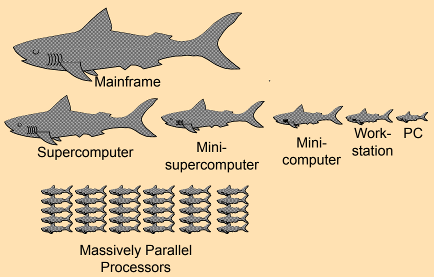
Dr. Norafida Ithnin
Advanced Computer System & Architecture
MCC 2313

Technology Trends

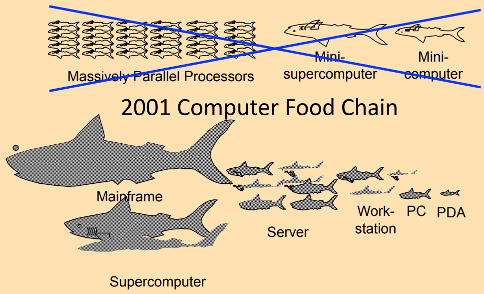
- Functionality Enhancements
  - Improvements in networking technology
  - Increase in communication and multimedia support
  - Support for larger programs
- Performance
  - Technology Advances
    - Decreases in feature size, increase in wafer size, lower voltages
  - Computer architecture advances improves low-end
    - RISC, superscalar, pipelining, …
- Price: Lower costs due to …
  - Simpler architectures
  - Higher volumes
  - More reliable systems
  - Improved technology

INSPIRING CREATIVE & INNOVATIVE MINDS

1988 Computer Food Chain



INSPIRING CREATIVE & INNOVATIVE MINDS



INSPIRING CREATIVE & INNOVATIVE MINDS

Processor Perspective

- Putting performance growth in perspective:

	Pentium 3(Coppermine)	Cray YMP
Type	Desktop	Supercomputer
Year	2000	1988
Clock	1130 MHz	167 MHz
MIPS	> 1000 MIPS	< 50 MIPS
Cost	$2,000	$1,000,000
Cache	256 KB	0.25 KB
Memory	512 MB	256 MB

INSPIRING CREATIVE & INNOVATIVE MINDS

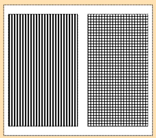
Where Has This Performance Improvement Come From?

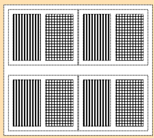
- Technology
  - More transistors per chip
  - Faster logic
- Machine Organization/Implementation
  - Deeper pipelines
  - More instructions executed in parallel
- Instruction Set Architecture
  - Reduced Instruction Set Computers (RISC)
  - Multimedia extensions
  - Explicit parallelism
- Compiler technology
  - Finding more parallelism in code
  - Greater levels of optimization

INSPIRING CREATIVE & INNOVATIVE MINDS

What is Ahead?

- Greater instruction level parallelism
- Bigger caches, and more levels of cache
- Multiple processors per chip
- Complete systems on a chip
- High performance interconnect





Computers in early 2000

- Technology
  - Very large dynamic RAM: 1 GB and beyond
  - Large fast static RAM: 1 MB, 10ns
  - Very large disks: Approaching 100 GB
- Complete systems on a chip
  - 20+ Million Transistors
- Parallelism
  - Superscalar, VLIW
  - Superpipeline
  - Explicitely Parallel
  - Multiprocessors
  - Distributed systems

Computers in the Early 2001

- Low Power
  - 60% of PCs portable by 2002
  - Performance per watt is now of interest
- Parallel I/O
  - Many applications I/O limited, not computation limited
  - Processors speeds increase faster than memory and I/O
- Multimedia
  - New interface technologies
  - Video, speech, handwriting, virtual reality, …
- Embedded systems extremely important
  - 90% of computers manufactured and 50% of processor revenue is in the embedded market (e.g., microcontrollers, DSPs, graphics processors, etc.)

Hardware Technology

	1980	1990	2001
Memory chips	64 KB	4 MB	256 MB
Clock Rate	1-2 MHz	20-40 MHz	700-1200 MHz
Hard disks	40 M	1 G	40 G
Floppies	.256 M	1.5 M	0.5-2 G

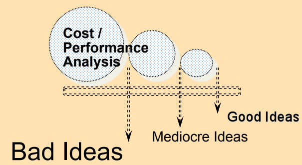
Computing in the 21st century

- Continue quadrupling memory about every 3 years
- Single-chip multiprocessor systems
- High-speed communication networks
- These improvements will create the need for new and innovative computer systems.

Measurement and Evaluation

Architecture is an iterative process:
- Search the possible design space
- Make selections
- Evaluate the selections made

Good measurement tools are required to accurately evaluate the selection.



Measurement Tools

- Benchmarks
- Simulation (many levels)
  - ISA, RTL, Gate, Transistor
- Cost, Delay, and Area Estimates
- Queuing Theory
- Rules of Thumb
- Fundamental Laws

The Bottom Line: Performance (and Cost)

Plane	DC to Paris	Speed	Passengers	Throughput (pmph)
Boeing 747	6.5 hours	610 mph	470	286,700
BAD/Sud Concorde	3 hours	1350 mph	132	178,200

- Time to run the task (Execution Time)
  - Execution time, response time, latency
- Tasks per day, hour, week, sec, ns … (Performance)
  - Throughput, bandwidth

Performance and Execution Time

Execution time and performance are reciprocals

$$\frac{\text{ExTime(Y)}}{\text{ExTime(X)}} = \frac{\text{Performance(X)}}{\text{Performance(Y)}}$$

- **Speed of Concorde vs. Boeing 747**
- **Throughput of Boeing 747 vs. Concorde**

Performance Terminology

"X is n% faster than Y" means:

$$\frac{\text{ExTime(Y)}}{\text{ExTime(X)}} = \frac{\text{Performance(X)}}{\text{Performance(Y)}} = 1 + \frac{n}{100}$$

$$n = \frac{100(\text{Performance(X)} - \text{Performance(Y)})}{\text{Performance(Y)}}$$

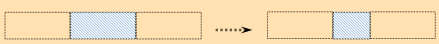
$$n = \frac{100(\text{ExTime(Y)} - \text{ExTime(X)})}{\text{ExTime(X)}}$$

Example: Y takes 15 seconds to complete a task, X takes 10 seconds. What % faster is X?

Amdahl's Law

Speedup due to enhancement E:

$$\text{Speedup(E)} = \frac{\text{ExTime w/o E}}{\text{ExTime w/E}} = \frac{\text{Performance w/ E}}{\text{Performance w/o E}}$$



Suppose that enhancement E accelerates a fraction $\text{Fraction}_{enhanced}$ of the task by a factor $\text{Speedup}_{enhanced}$, and the remainder of the task is unaffected.

What are the new execution time and the overall speedup due to the enhancement?

Amdahl's Law

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

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

Example of Amdahl's Law

- Floating point instructions improved to run 2X; but only 10% of the time was spent on these instructions.

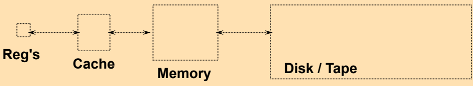
$ExTime_{new}$ =

=

$Speedup_{overall}$

Make The Common Case Fast

- All instructions require an instruction fetch, only a fraction require a data fetch/store.
  - Optimize instruction access over data access
- Programs exhibit *locality*
  - 90% of time in 10% of code
  - Temporal Locality (items referenced recently)
  - Spatial Locality (items referenced nearby)
- Access to small memories is faster
  - Provide a *storage hierarchy* such that the most frequent accesses are to the smallest (closest) memories.



Hardware/Software Partitioning

- The simple case is usually the most frequent and the easiest to optimize!
- Do simple, fast things in hardware and be sure the rest can be handled correctly in software.

Would you handled these in hardware or software:

- Integer addition?
- Accessing data from disk?
- Floating point square root?

Performance Factors

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

- The number of instructions/program is called the instruction count (IC).
- The average number of cycles per instruction is called the CPI.
- The number of seconds per cycle is the clock period.
- The clock rate is the multiplicative inverse of the clock period and is given in cycles per second (or MHz).
- For example, if a processor has a clock period of 5 ns, what is it's clock rate?

Aspects of CPU Performance

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

	Instr. Cnt	CPI	Clock Rate
Program			
Compiler			
Instr. Set			
Organization			
Technology			

Aspects of CPU Performance

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

	Inst Count	CPI	Clock Rate
Program	X	X	
Compiler	X	X	
Inst. Set.	X	X	
Organization		X	X
Technology			X

Marketing Metrics

MIPS = Instruction Count / Time * 10^6 = Clock Rate / CPI * 10^6

- Not effective for machines with different instruction sets
- Not effective for programs with different instruction mixes
- Uncorrelated with performance

MFLOPs = FP Operations / Time * 10^6

- Machine dependent
- Often not where time is spent

Normalized MFLOPS:	
add,sub,compare,mult	1
divide, sqrt	4
exp, sin, . . .	8

•Peak - maximum able to achieve

•Native - average for a set of benchmarks

•Relative - compared to another platform

INSPIRING CREATIVE & INNOVATIVE MINDS

Programs to Evaluate Processor Performance

- (Toy) Benchmarks
  - 10-100 line program
  - e.g.: sieve, puzzle, quicksort
- Synthetic Benchmarks
  - Attempt to match average frequencies of real workloads
  - e.g., Whetstone, dhrystone
- Kernels
  - Time critical excerpts
- Real Benchmarks

INSPIRING CREATIVE & INNOVATIVE MINDS

Benchmarks

- Benchmark mistakes
  - Only average behavior represented in test workload
  - Loading level controlled inappropriately
  - Caching effects ignored
  - Buffer sizes not appropriate
  - Ignoring monitoring overhead
  - Not ensuring same initial conditions
  - Collecting too much data but doing too little analysis
- Benchmark tricks
  - Compiler wired to optimize the workload
  - Very small benchmarks used
  - Benchmarks manually translated to optimize performance

INSPIRING CREATIVE & INNOVATIVE MINDS

SPEC: System Performance Evaluation Cooperative

- First Round SPEC CPU89
  - 10 programs yielding a single number
- Second Round SPEC CPU92
  - SPEC CINT92 (6 integer programs) and SPEC CFP92 (14 floating point programs)
  - Compiler flags can be set differently for different programs
- Third Round SPEC CPU95
  - new set of programs: SPEC CINT95 (8 integer programs) and SPEC CFP95 (10 floating point)
  - Single flag setting for all programs
- Fourth Round SPEC CPU2000
  - new set of programs: SPEC CINT2000 (12 integer programs) and SPEC CFP2000 (14 floating point)
  - Single flag setting for all programs
  - Programs in C, C++, Fortran 77, and Fortran 90

INSPIRING CREATIVE & INNOVATIVE MINDS

SPEC 2000

- 12 integer programs:
  - 2 Compression
  - 2 Circuit Placement and Routing
  - C Programming Language Compiler
  - Combinatorial Optimization
  - Chess, Word Processing
  - Computer Visualization
  - PERL Programming Language
  - Group Theory Interpreter
  - Object-oriented Database.
  - Written in C (11) and C++ (1)
- 14 floating point programs:
  - Quantum Physics
  - Shallow Water Modeling
  - Multi-grid Solver
  - 3D Potential Field
  - Parabolic / Elliptic PDEs
  - 3-D Graphics Library
  - Computational Fluid Dynamics
  - Image Recognition
  - Seismic Wave Simulation
  - Image Processing
  - Computational Chemistry
  - Number Theory / Primality Testing
  - Finite-element Crash Simulation
  - High Energy Nuclear Physics
  - Pollutant Distribution
  - Written in Fortran (10) and C (4)

INSPIRING CREATIVE & INNOVATIVE MINDS

Other SPEC Benchmarks

- JVM98:
  - Measures performance of Java Virtual Machines
- SFS97:
  - Measures performance of network file server (NFS) protocols
- Web99:
  - Measures performance of World Wide Web applications
- HPC96:
  - Measures performance of large, industrial applications
- APC, MEDIA, OPC
  - Meausre performance of graphics applications
- For more information about the SPEC benchmarks see: http://www.spec.org.

INSPIRING CREATIVE & INNOVATIVE MINDS

UTM
RESEARCH UNIVERSITY

Conclusions on Performance

- A fundamental rule in computer architecture is to make the common case fast.
- The most accurate measure of performance is the execution time of representative real programs (benchmarks).
- Execution time is dependent on the number of instructions per program, the number of cycles per instruction, and the clock rate.
- When designing computer systems, both cost and performance need to be taken into account.

INSPIRING CREATIVE & INNOVATIVE MINDS

UTM
UNIVERSITI TEKNOLOGI MALAYSIA
RESEARCH UNIVERSITY

Discussions

INSPIRING CREATIVE & INNOVATIVE MINDS