ECG 700 Advanced Computer System Architecture Fall 2012

Lecture 1 – Fundamentals of Quantitative Design and Analysis

Mei Yang

Adapted from David Patterson's slides on graduate computer architecture

Outline

- Introduction
- Classes of Computers
- Defining Computer Architecture
- Trends in Technology
- Trends in Power and Calculation
- Trends in Cost and Calculation
- Dependability: Definition and Quantity
- Relative Performance Metrics

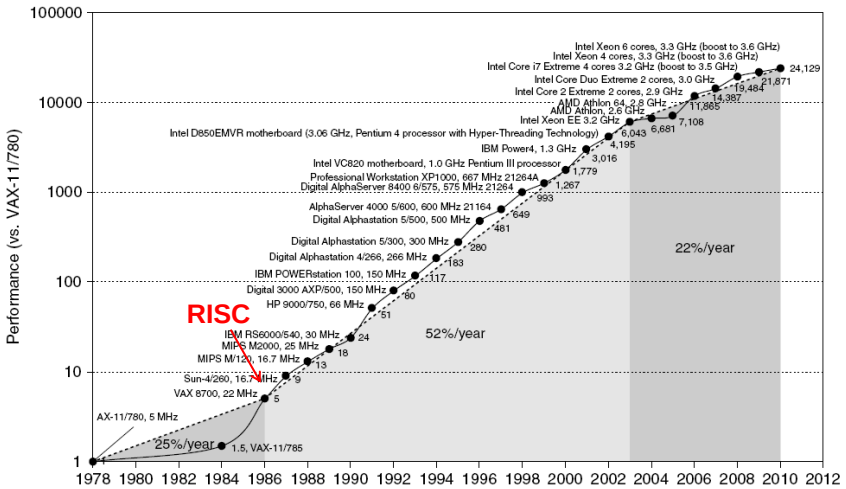
2

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

3

Single Processor Performance



- VAX : 25%/year 1978 to 1986
- RISC + x86: 52%/year 1986 to 2003
- multiprocessor: <22%/year 2003 to present

4

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

5

Classes of Computers

- Personal Mobile Device (PMD)
  - e.g. smart phones, tablet computers
  - Emphasis on energy efficiency and real-time performance

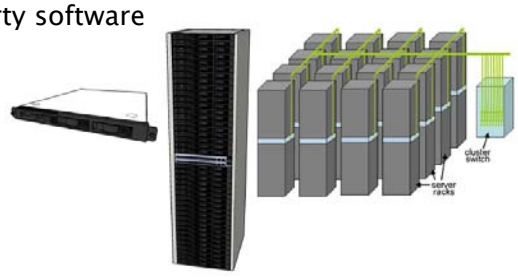
- Desktop Computing
  - Emphasis on price-performance

- Servers
  - Emphasis on availability, scalability, throughput

6

Classes of Computers

- 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
  - Cannot run third-party software
  - Emphasis: price



7

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

8

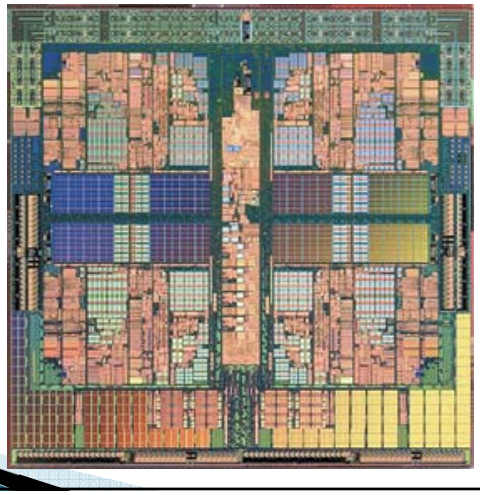
Flynn's Taxonomy

- 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

9

Sea Change in Chip Design

- Sea change in chip design: multiple "cores" (2X processors per chip / ~ 2 years)
  - More simpler processors are more power efficient



10

Problems with Sea Change

- Architectures not ready for 1000 CPUs / chip
- Algorithms, Programming Languages, Compilers, Operating Systems, Libraries, … not ready to supply Thread Level Parallelism or Data Level Parallelism for 1000 CPUs / chip

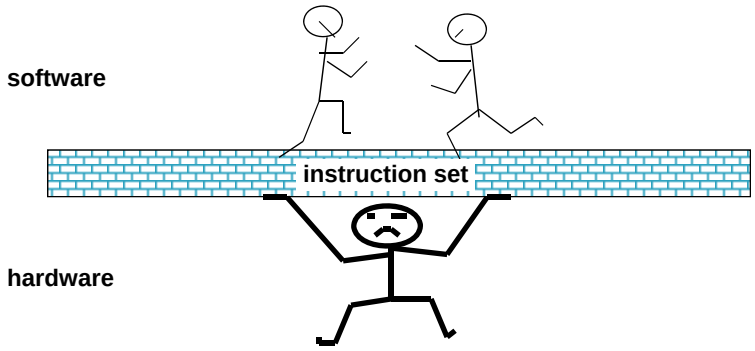
11

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

12

Instruction Set Architecture



- Properties of a good abstraction
  - Lasts through many generations (portability)
  - Used in many different ways (generality)
  - Provides convenient functionality to higher levels
  - Permits an efficient implementation at lower levels

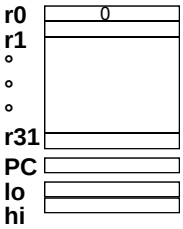
13

Instruction Set Architecture

- ISA
  - Class of ISA
    - General-purpose register architecture
  - Memory addressing
    - Byte addressing vs. word addressing
  - Addressing modes
    - Register, immediate, indirect, etc.
  - Types and sizes of operands
    - 8-bit, 16-bit, 32-bit, 64-bit
  - Operations
    - Data transfer, arithmetic, logic, control, and floating point
  - Control flow instructions
    - Conditional branch, unconditional jumps, procedure calls, and returns
  - Encoding an ISA
    - Fixed length vs. variable length

14

Example: MIPS



Programmable storage

- 2^{32} x bytes
- 31 x 32-bit GPRs (R0=0)
- 32 x 32-bit FP regs (paired DP)
- HI, LO, PC

Data types ?

Format ?

Addressing Modes?

Arithmetic logical

Add, AddU, Sub, SubU, And, Or, Xor, Nor, SLT, SLTU,
AddI, AddIU, SLTI, SLTIU, AndI, OrI, XorI, *LUI*
SLL, SRL, SRA, SLLV, SRLV, SRAV

Memory Access

LB, LBU, LH, LHU, LW, LWL,LWR
SB, SH, SW, SWL, SWR

Control

J, JAL, JR, JALR
BEq, BNE, BLEZ,BGTZ,BLTZ,BGEZ,BLTZAL,BGEZAL

32-bit instructions on word boundary

15

ISA vs. Computer Architecture

- Old definition of computer architecture = instruction set design
  - Other aspects of computer design called implementation
  - Insinuates implementation is uninteresting or less challenging
- Computer architecture covers all aspects of computer design: ISA, organization, and hardware
- Architect's job: design a computer to meet functional requirements as well as price, performance, and availability goals.
- What really matters is the functioning of the complete system
  - hardware, runtime system, compiler, operating system, and application

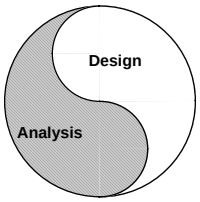
16

Functional Requirements

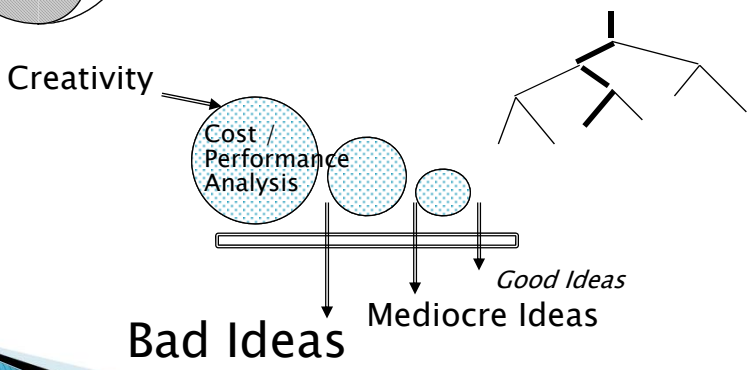
Functional requirements	Typical features required or supported
Application area	*Target of computer*
Personal mobile device	Real-time performance for a range of tasks, including interactive performance for graphics, video, and audio; energy efficiency (Ch. 2, 3, 4, 5; App. A)
General-purpose desktop	Balanced performance for a range of tasks, including interactive performance for graphics, video, and audio (Ch. 2, 3, 4, 5; App. A)
Servers	Support for databases and transaction processing; enhancements for reliability and availability; support for scalability (Ch. 2, 5; App. A, D, F)
Clusters/warehouse-scale computers	Throughput performance for many independent tasks; error correction for memory; energy proportionality (Ch 2, 6; App. F)
Embedded computing	Often requires special support for graphics or video (or other application-specific extension); power limitations and power control may be required; real-time constraints (Ch. 2, 3, 5; App. A, E)
Level of software compatibility	*Determines amount of existing software for computer*
At programming language	Most flexible for designer; need new compiler (Ch. 3, 5; App. A)
Object code or binary compatible	Instruction set architecture is completely defined—little flexibility—but no investment needed in software or porting programs (App. A)
Operating system requirements	*Necessary features to support chosen OS (Ch. 2; App. B)*
Size of address space	Very important feature (Ch. 2); may limit applications
Memory management	Required for modern OS; may be paged or segmented (Ch. 2)
Protection	Different OS and application needs: page vs. segment; virtual machines (Ch. 2)
Standards	*Certain standards may be required by marketplace*
Floating point	Format and arithmetic: IEEE 754 standard (App. J), special arithmetic for graphics or signal processing
I/O interfaces	For I/O devices: Serial ATA, Serial Attached SCSI, PCI Express (App. D, F)
Operating systems	UNIX, Windows, Linux, CISCO IOS
Networks	Support required for different networks: Ethernet, Infiniband (App. F)
Programming languages	Languages (ANSI C, C++, Java, Fortran) affect instruction set (App. A)

17

Computer Architecture is Design and Analysis



- Architecture design is an iterative process:
  - Searching the space of possible designs
  - At all levels of computer systems



18

Conventional Wisdom in Comp. Arch

- Old Conventional Wisdom: Power is free, Transistors expensive
- New Conventional Wisdom: "Power wall" Power expensive, transistors free (Can put more on chip than can afford to turn on)
- Old CW: Sufficiently increasing Instruction Level Parallelism via compilers, innovation (Out-of-order, speculation, VLIW, …)
- New CW: "ILP wall" law of diminishing returns on more HW for ILP
- Old CW: Multiplies are slow, Memory access is fast
- New CW: "Memory wall" Memory slow, multiplies fast (200 clock cycles to DRAM memory, 4 clocks for multiply)
- Old CW: Uniprocessor performance 2X / 1.5 yrs
- New CW: Power Wall + ILP Wall + Memory Wall = Brick Wall
  - Uniprocessor performance now 2X / 5(?) yrs

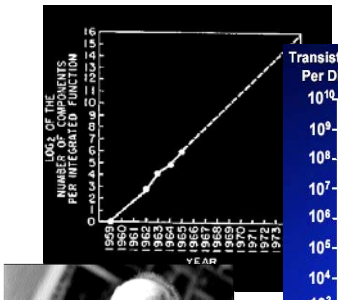
19

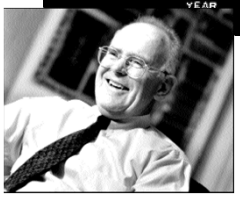
Trends in Technology

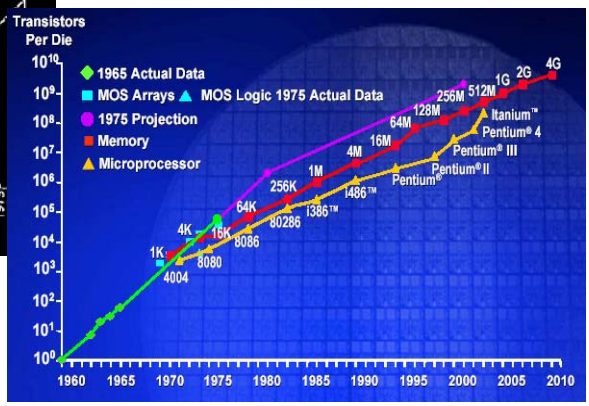
- Integrated circuit technology
  - Transistor density: 35%/year
  - Die size: 10-20%/year
  - Integration overall: 40-55%/year
- DRAM capacity: 25-40%/year (slowing)
- Flash capacity: 50-60%/year
  - 15-20X cheaper/bit than DRAM
- Magnetic disk technology: 40%/year
  - 15-25X cheaper/bit then Flash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

20

Integrated Circuit Design



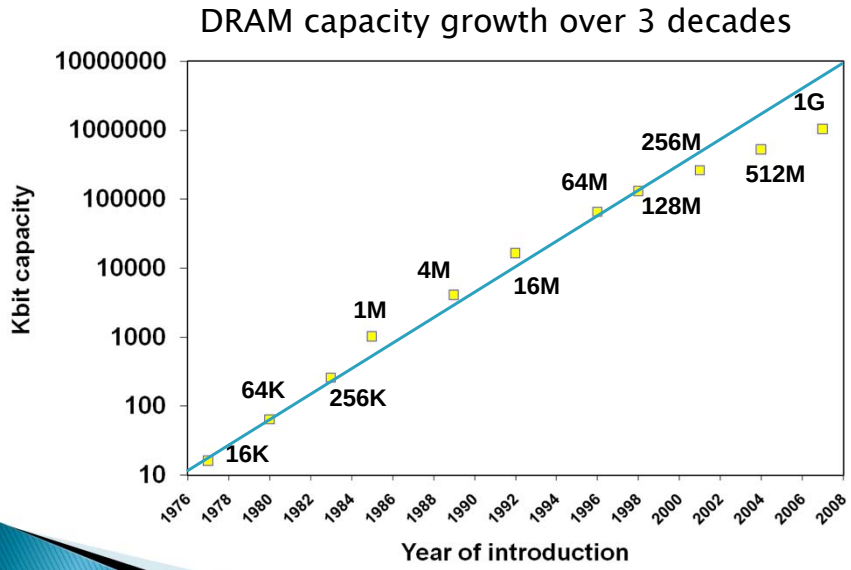




- "Cramming More Components onto Integrated Circuits"
  - Gordon Moore, Electronics, 1965
- # on transistors / cost-effective integrated circuit double every N months ($12 \leq N \leq 24$)

21

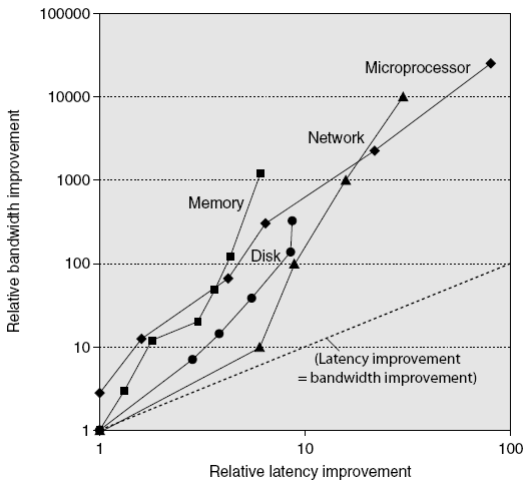
DRAM



22

Bandwidth over Latency

- Bandwidth: total amount of work done in unit time
  - E.g., M bits/second over network, M bytes/second from disk
- Latency: elapsed time for a single work
  - E.g., one-way network delay in microseconds, average disk access time in milliseconds



23

Milestones

Microprocessor	16-bit address/ bus, microcoded	32-bit address/ bus, microcoded	5-stage pipeline, on-chip I & D caches, FPU	2-way superscalar, 64-bit bus	Out-of-order 3-way superscalar	Out-of-order superpipelined, on-chip L2 cache	Multicore OOO 4-way on chip L3 cache, Turbo
Product	Intel 80286	Intel 80386	Intel 80486	Intel Pentium	Intel Pentium Pro	Intel Pentium 4	Intel Core i7
Year	1982	1985	1989	1993	1997	2001	2010
Die size (mm^2)	47	43	81	90	308	217	240
Transistors	134,000	275,000	1,200,000	3,100,000	5,500,000	42,000,000	1,170,000,000
Processors/chip	1	1	1	1	1	1	4
Pins	68	132	168	273	387	423	1366
Latency (clocks)	6	5	5	5	10	22	14
Bus width (bits)	16	32	32	64	64	64	196
Clock rate (MHz)	12.5	16	25	66	200	1500	3333
Bandwidth (MIPS)	2	6	25	132	600	4500	50,000
Latency (ns)	320	313	200	76	50	15	4
Memory module	DRAM	Page mode DRAM	Fast page mode DRAM	Fast page mode DRAM	Synchronous DRAM	Double data rate SDRAM	DDR3 SDRAM
Module width (bits)	16	16	32	64	64	64	64
Year	1980	1983	1986	1993	1997	2000	2010
Mbits/DRAM chip	0.06	0.25	1	16	64	256	2048
Die size (mm^2)	35	45	70	130	170	204	50
Pins/DRAM chip	16	16	18	20	54	66	134
Bandwidth (MBytes/s)	13	40	160	267	640	1600	16,000
Latency (ns)	225	170	125	75	62	52	37

24

Milestones (cont'd)

Memory module	DRAM	Page mode DRAM	Fast page mode DRAM	Fast page mode DRAM	Synchronous DRAM	Double data rate SDRAM	DDR3 SDRAM
Module width (bits)	16	16	32	64	64	64	64
Year	1980	1983	1986	1993	1997	2000	2010
Mbits/DRAM chip	0.06	0.25	1	16	64	256	2048
Die size (mm^2)	35	45	70	130	170	204	50
Pins/DRAM chip	16	16	18	20	54	66	134
Bandwidth (MBytes/s)	13	40	160	267	640	1600	16,000
Latency (ns)	225	170	125	75	62	52	37
Local area network	Ethernet	Fast Ethernet	Gigabit Ethernet	10 Gigabit Ethernet	100 Gigabit Ethernet		
IEEE standard	802.3	803.3u	802.3ab	802.3ac	802.3ba		
Year	1978	1995	1999	2003	2010		
Bandwidth (Mbits/sec)	10	100	1000	10,000	100,000		
Latency (µsec)	3000	500	340	190	100		
Hard disk	3600 RPM	5400 RPM	7200 RPM	10,000 RPM	15,000 RPM	15,000 RPM	
Product	CDC WrenI 94145-36	Seagate ST41600	Seagate ST15150	Seagate ST39102	Seagate ST373453	Seagate ST3600057	
Year	1983	1990	1994	1998	2003	2010	
Capacity (GB)	0.03	1.4	4.3	9.1	73.4	600	
Disk form factor	5.25 inch	5.25 inch	3.5 inch	3.5 inch	3.5 inch	3.5 inch	
Media diameter	5.25 inch	5.25 inch	3.5 inch	3.0 inch	2.5 inch	2.5 inch	
Interface	ST-412	SCSI	SCSI	SCSI	SCSI	SAS	
Bandwidth (MBytes/s)	0.6	4	9	24	86	204	
Latency (ms)	48.3	17.1	12.7	8.8	5.7	3.6	

25

Reasons

- Moore's Law helps BW more than latency
  - Faster transistors, more transistors, more pins help bandwidth
  - Smaller, faster transistors but communicate over (relatively) longer lines: limits latency
- Distance limits latency
  - Size of DRAM block ⇒ long bit and word lines ⇒ most of DRAM access time
- Latency helps BW, but not vice versa
  - Lower DRAM latency ⇒ More access/second (higher bandwidth)
  - Higher linear density helps disk BW (and capacity), but not disk Latency
- Bandwidth hurts latency
  - Adding chips to widen a memory module increases Bandwidth but higher fan-out on address lines may increase Latency
- Operating system overhead hurts latency more than bandwidth
  - Long messages amortize overhead; overhead bigger part of short messages
- Bandwidth easier to sell ("bigger=better")
  - E.g., 4400 MB/s DIMM ("PC4400") vs. 50 ns latency

26

Transistors and Wires

- Feature size

Year	2004	2006	2008	2010	2012
Feature size (nm)	90	65	45	32	22
Intg. Capacity (BT)	2	4	6	16	32

  - 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!

$$T = 0.39rcl^2$$

- Integration density increases quadratically with a linear decrease in feature size

27

Trends in 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

28

Dynamic Power

- For CMOS chips, traditional dominant energy consumption has been in switching transistors, called *dynamic power*

$$Power_{dynamic} = 1/2 \times CapacitiveLoad \times Voltage^2 \times FrequencySwitched$$

- For mobile devices, energy better metric

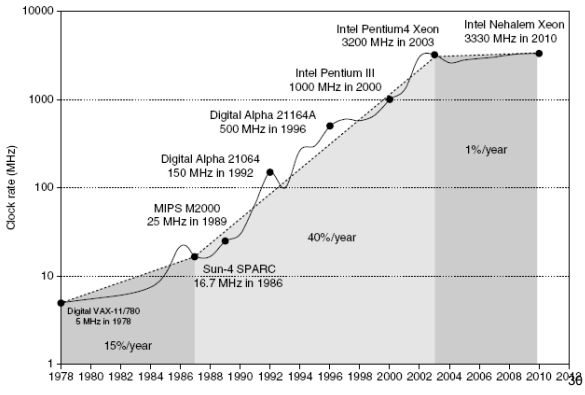
$$Energy_{dynamic} = CapacitiveLoad \times Voltage^2$$

- For a fixed task, slowing clock rate (frequency switched) reduces power, but not energy
- Capacitive load is a function of number of transistors connected to output and technology, which determines capacitance of wires and transistors
- Dropping voltage helps both, so went from 5V to 1V
- To save energy & dynamic power, most CPUs now turn off clock of inactive modules (e.g. Fl. Pt. Unit)

29

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



30

Example of quantifying power

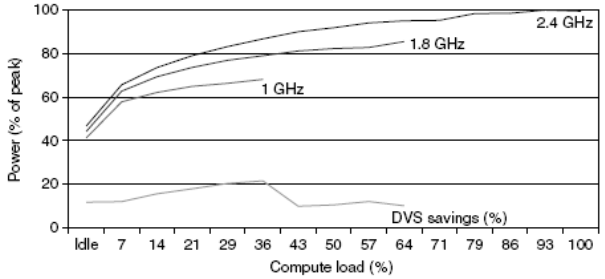
- Suppose 15% reduction in voltage results in a 15% reduction in frequency. What is impact on dynamic power?

$$
\begin{aligned}
Power_{dynamic} &= 1/2 \times CapacitiveLoad \times Voltage^2 \times FrequencySwitched \\
&= 1/2 \times .85 \times CapacitiveLoad \times (.85 \times Voltage)^2 \times FrequencySwitched \\
&= (.85)^3 \times OldPower_{dynamic} \\
&\approx 0.6 \times OldPower_{dynamic}
\end{aligned}
$$

31

Reducing Power

- Techniques for reducing power:
  - Do nothing well
  - Dynamic Voltage-Frequency Scaling
  - Design for typical cases
    - using low power modes for DRAM and storage
  - Overclocking, turning off cores



32

Static Power

- Because leakage current flows even when a transistor is off, now *static power* important too

$$Power_{static} = Current_{static} \times Voltage$$

- Leakage current increases in processors with smaller transistor sizes
- Increasing the number of transistors increases power even if they are turned off
- To reduce: power gating

33

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

34

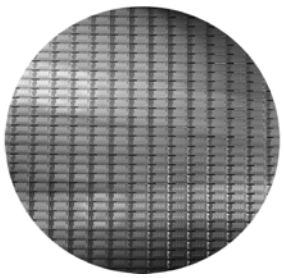
Integrated Circuit Cost

- 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)

35

Example on cost of die

- Find the cost of die of a 300 mm (30 cm) wafer a die that is 1.5 cm on a side. Assume the wafer cost is at $5.500.

$$\text{Dies per wafer} = \frac{\pi \times (30/2)^2}{2.25} - \frac{\pi \times 30}{\sqrt{2 \times 2.25}} = \frac{706.9}{2.25} - \frac{94.2}{2.12} = 270$$

$$\text{Die yield} = \left(1 + \frac{0.4 \times 2.25}{4.0}\right)^{-4} = 0.44$$

Cost of die = 5500/(270*0.44)≈$46

36

Dependability

- How decide when a system is operating properly?
- Infrastructure providers now offer Service Level Agreements (SLA) to guarantee that their networking or power service would be dependable
- Systems alternate between 2 states of service with respect to an SLA:
  - Service accomplishment, where the service is delivered as specified in SLA
  - Service interruption, where the delivered service is different from the SLA
- Failure = transition from state 1 to state 2
- Restoration = transition from state 2 to state 1

37

Define and Quantity Dependability

- Module reliability = measure of continuous service accomplishment (or time to failure).
  - Mean Time To Failure (MTTF) measures Reliability
  - Failures In Time (FIT) = 1/MTTF, the rate of failures
    - Traditionally reported as failures per billion hours of operation
- Mean Time To Repair (MTTR) measures Service Interruption
  - Mean Time Between Failures (MTBF) = MTTF+MTTR
- Module availability measures service as alternate between the 2 states of accomplishment and interruption (number between 0 and 1, e.g. 0.9)
- Module availability = MTTF / (MTTF + MTTR)

38

Example calculating reliability

- If modules have exponentially distributed lifetimes (age of module does not affect probability of failure), overall failure rate is the sum of failure rates of the modules
- Calculate FIT and MTTF for 10 disks (1M hour MTTF per disk), 1 disk controller (0.5M hour MTTF), and 1 power supply (0.2M hour MTTF):

$$\begin{aligned} FailureRate &= 10 \times (1/1{,}000{,}000) + 1/500{,}000 + 1/200{,}000 \\ &= 10 + 2 + 5/1{,}000{,}000 \\ &= 17/1{,}000{,}000 \\ &= 17{,}000\,FIT \\ MTTF &= 1{,}000{,}000{,}000/17{,}000 \\ &\approx 59{,}000\,hours \end{aligned}$$

39

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

40

What to Measure

- Benchmarks
  - Kernels (e.g. matrix multiply)
  - Toy programs (e.g. sorting)
  - Synthetic benchmarks (e.g. Dhrystone)
  - Benchmark suites (e.g. SPEC06fp, TPC-C)
- SPECRatio: Normalize execution times to reference computer, yielding a ratio proportional to performance =

  time on reference computer /
  time on computer being rated

41

Summarizing Performance Results

- If program SPECRatio on Computer A is 1.25 times bigger than Computer B, then

$$1.25 = \frac{SPECRatio_A}{SPECRatio_B} = \frac{\frac{ExecutionTime_{reference}}{ExecutionTime_A}}{\frac{ExecutionTime_{reference}}{ExecutionTime_B}}$$

$$= \frac{ExecutionTime_B}{ExecutionTime_A} = \frac{Performance_A}{Performance_B}$$

- **Note that when comparing 2 computers as a ratio, execution times on the reference computer drop out, so choice of reference computer is irrelevant**

42

Summarizing Performance Results

- Since ratios, proper mean is geometric mean (SPECRatio unitless, so arithmetic mean meaningless)

$$GeometricMean = \sqrt[n]{\prod_{i=1}^{n} SPECRatio_i}$$

1. **Geometric mean of the ratios is the same as the ratio of the geometric means**
2. **Ratio of geometric means = Geometric mean of performance ratios ⇒ choice of reference computer is irrelevant!**

- **These two points make geometric mean of ratios attractive to summarize performance**

43

Summarizing Performance Results

- Does a single mean well summarize performance of programs in benchmark suite?
- Can decide if mean a good predictor by characterizing variability of distribution using standard deviation
- Like geometric mean, geometric standard deviation is multiplicative rather than arithmetic
- Can simply take the logarithm of SPECRatios, compute the standard mean and standard deviation, and then take the exponent to convert back:

$$GeometricMean = \exp\left(\frac{1}{n} \times \sum_{i=1}^{n} \ln\left(SPECRatio_i\right)\right)$$

$$GeometricStDev = \exp\left(StDev\left(\ln\left(SPECRatio_i\right)\right)\right)$$

44

Summarizing Performance Results

- Standard deviation is more informative if know distribution has a standard form
  - bell-shaped normal distribution, whose data are symmetric around mean
  - lognormal distribution, where logarithms of data--not data itself--are normally distributed (symmetric) on a logarithmic scale
- For a lognormal distribution, we expect that

68% of samples fall in range $[mean / gstdev, mean \times gstdev]$

95% of samples fall in range $[mean / gstdev^2, mean \times gstdev^2]$

- Note: Excel provides functions EXP(), LN(), and STDEV() that make calculating geometric mean and multiplicative standard deviation easy

45

Example of mean and gstdev

- Using the data in Figure 1.14, calculate the geometric standard deviation and the percentage of the results that fall within a single standard deviation of the geometric mean. Are the results compatible with a lognormal distribution?

Benchmarks	Ultra 5 Time (sec)	Opteron Time (sec)	SPECRatio	Itanium 2 Time (sec)	SPECRatio	Opteron/Itanium Times (sec)	Itanium/Opteron SPECRatios
wupwise	1600	51.5	31.06	56.1	28.53	0.92	0.92
swim	3100	125.0	24.73	70.7	43.85	1.77	1.77
mgrid	1800	98.0	18.37	65.8	27.36	1.49	1.49
applu	2100	94.0	22.34	50.9	41.25	1.85	1.85
mesa	1400	64.6	21.69	108.0	12.99	0.60	0.60
galgel	2900	86.4	33.57	40.0	72.47	2.16	2.16
art	2600	92.4	28.13	21.0	123.67	4.40	4.40
equake	1300	72.6	17.92	36.3	35.78	2.00	2.00
facerec	1900	73.6	25.80	86.9	21.86	0.85	0.85
ammp	2200	136.0	16.14	132.0	16.63	1.03	1.03
lucas	2000	88.8	22.52	107.0	18.76	0.83	0.83
fma3d	2100	120.0	17.48	131.0	16.09	0.92	0.92
sixtrack	1100	123.0	8.95	68.8	15.99	1.79	1.79
apsi	2600	150.0	17.36	231.0	11.27	0.65	0.65
Geometric mean			20.86		27.12	1.30	1.30

46

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

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

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

47

Example

- Suppose FP square root (FPSQR) is responsible for 20% of the execution time of a critical graphics benchmark. Two proposals for improving the performance:

  1) Enhance the FPSQR hardware and speed up this operation by a factor of 10.

  2) Make all FP instructions in the graphics processor run faster by a factor of 1.6; FP instructions are responsible for half of the execution time for the application.

  Compare these two design alternatives.

$$\text{Speedup}_{\text{FPSQR}} = \frac{1}{(1-0.2)+0.2/10} = 1/0.82 = 1.22$$

$$\text{Speedup}_{\text{FO}} = \frac{1}{(1-0.5)+0.5/1.6} = 1/0.8125 = 1.23$$

48

Example on reliability

- How much improvement of the reliability of whole system that can result in if the power supply's MTTF is improved via redundancy from 200,000-hour to 830,000,000-hour MTTF, or 4150X better for the previous example on slide 29?

$$FailureRate = 10 \times (1/1{,}000{,}000) + 1/500{,}000 + 1/200{,}000$$
$$= 10 + 2 + 5/1{,}000{,}000$$
$$= 17/1{,}000{,}000$$
$$= 17{,}000\,FIT$$
$$MTTF = 1{,}000{,}000{,}000/17{,}000$$
$$\approx 59{,}000\,hours$$

$$\text{Speedup}_{\text{power supply pair}} = \frac{1}{(1-5/17)+0.29/4150} = 1/0.71 = 1.41$$

49

Relative Performance Metrics

- The most important program property is the principle of locality
  - The 90-10 Rule: a program spends 90% of its execution time in only 10% of the code
  - Temporal locality: the recently accessed items are likely to be accessed in the near future
  - Spatial locality: whose addresses are near one another tend to be referenced together in time
- Focus on the common case
  - Works for power as well as for resource allocation and performance

50

Principles of Computer Design

- The Processor Performance Equation

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

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

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

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

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

51

Processor Performance

CPU time = Instruction_count x CPI x clock_cycle

	Instruction_ count	CPI	clock_cycle
Algorithm			
Programming language			
Compiler			
ISA			
Core organization			
Technology			

52

Processor Performance

CPU time = Instruction_count x CPI x clock_cycle

	Instruction_ count	CPI	clock_cycle
Algorithm	X	X	
Programming language	X	X	
Compiler	X	X	
ISA	X	X	X
Core organization		X	X
Technology			X

53

Calculation of CPI

- Computing the overall effective CPI is done by looking at the different types of instructions and their individual cycle counts and averaging

$$\text{CPI} = \frac{\sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i}{\text{Instruction count}} = \sum_{i=1}^{n} \frac{\text{IC}_i}{\text{Instruction count}} \times \text{CPI}_i$$

  - Where IC_i is the count (percentage) of the number of instructions of class i executed
  - CPI_i is the (average) number of clock cycles per instruction for that instruction class
  - n is the number of instruction classes

54

Example

- Suppose we have made the following measurements:
  Frequency of FP operations = 25%
  Average CPI of FP operations = 4.0
  Average CPI of other instructions = 1.33
  Frequency of FPSQR= 2%
  CPI of FPSQR = 20
  Assume that the two design alternatives are to decrease the CPI of FPSQR to 2 or to decrease the average CPI of all FP operations to 2.5. Compare these two design alternatives using the processor performance equation.

55

Example

$$\text{CPI}_{\text{original}} = \sum_{i=1}^{n} \text{CPI}_i \times \left(\frac{\text{IC}_i}{\text{Instruction count}} \right)$$
$$= (4 \times 25\%) + (1.33 \times 75\%) = 2.0$$

$$\text{CPI}_{\text{with new FPSQR}} = \text{CPI}_{\text{original}} - 2\% \times (\text{CPI}_{\text{old FPSQR}} - \text{CPI}_{\text{of new FPSQR only}})$$
$$= 2.0 - 2\% \times (20 - 2) = 1.64$$

$$\text{CPI}_{\text{new FP}} = (75\% \times 1.33) + (25\% \times 2.5) = 1.62$$

$$\text{Speedup}_{\text{new FP}} = \frac{\text{CPU time}_{\text{original}}}{\text{CPU time}_{\text{new FP}}} = \frac{\text{IC} \times \text{Clock cycle} \times \text{CPI}_{\text{original}}}{\text{IC} \times \text{Clock cycle} \times \text{CPI}_{\text{new FP}}}$$
$$= \frac{\text{CPI}_{\text{original}}}{\text{CPI}_{\text{new FP}}} = \frac{2.00}{1.625} = 1.23$$

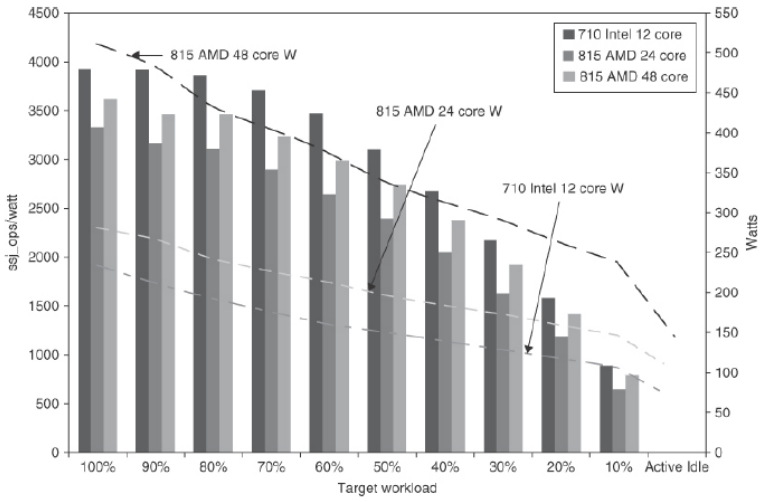
56

Performance, Price, and Power

	System 1		System 2		System 3	
Component		Cost (% Cost)		Cost (% Cost)		Cost (% Cost)
Base server	PowerEdge R710	$653 (7%)	PowerEdge R815	$1437 (15%)	PowerEdge R815	$1437 (11%)
Power supply	570 W		1100 W		1100 W	
Processor	Xeon X5670	$3738 (40%)	Opteron 6174	$2679 (29%)	Opteron 6174	$5358 (42%)
Clock rate	2.93 GHz		2.20 GHz		2.20 GHz	
Total cores	12		24		48	
Sockets	2		2		4	
Cores/socket	6		12		12	
DRAM	12 GB	$484 (5%)	16 GB	$693 (7%)	32 GB	$1386 (11%)
Ethernet Inter.	Dual 1-Gbit	$199 (2%)	Dual 1-Gbit	$199 (2%)	Dual 1-Gbit	$199 (2%)
Disk	50 GB SSD	$1279 (14%)	50 GB SSD	$1279 (14%)	50 GB SSD	$1279 (10%)
Windows OS		$2999 (32%)		$2999 (33%)		$2999 (24%)
Total		$9352 (100%)		$9286 (100%)		$12,658 (100%)
Max ssj_ops	910,978		926,676		1,840,450	
Max ssj_ops/$	97		100		145	

57

Performance, Price, and Power



58