Computer Organization and Design

Chapter 1

Computer Abstractions and Technology

Review:Classes of Computers

❑ Desktop and laptop computers

Designed to deliver good performance to a single user at low cost usually executing 3rd party software, usually incorporating a graphics display, a keyboard, and a mouse

❑ Servers

Used to run larger programs for multiple, simultaneous users typically accessed only via a network and that places a greater emphasis on dependability and (often) security

❑ Supercomputers

A high performance, high cost class of servers with hundreds to thousands of processors, terabytes of memory and petabytes of storage that are used for high-end scientific and engineering applications

❑ Embedded computers (processors)

A computer inside another device used for running one predetermined application

Review: Some Basic Definitions

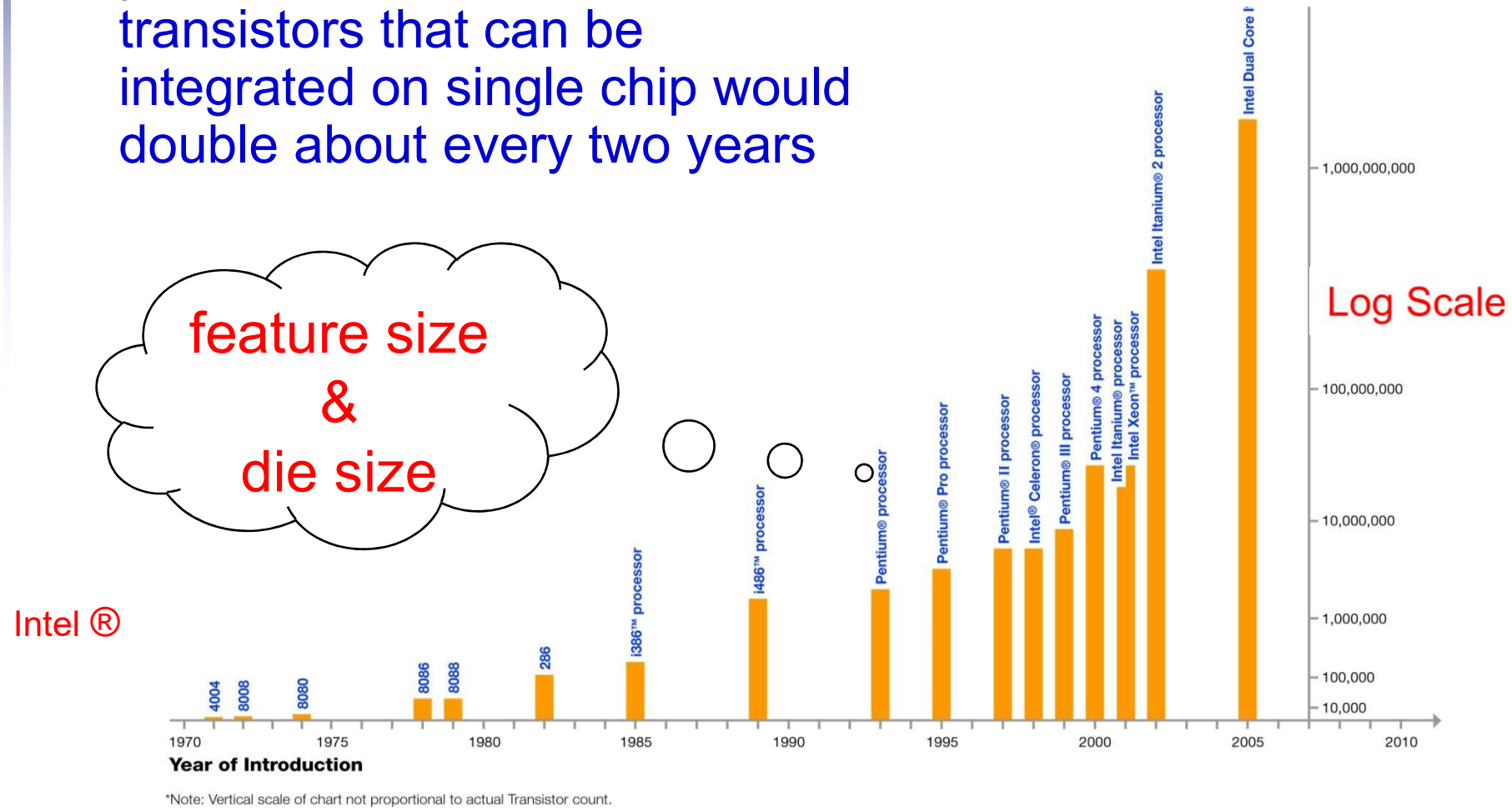
- Kilobyte – 2^{10} or 1,024 bytes
- Megabyte– 2^{20} or 1,048,576 bytes
  - sometimes “rounded” to 10^6 or 1,000,000 bytes
- Gigabyte – 2^{30} or 1,073,741,824 bytes
  - sometimes rounded to 10^9 or 1,000,000,000 bytes
- Terabyte – 2^{40} or 1,099,511,627,776 bytes
  - sometimes rounded to 10^{12} or 1,000,000,000,000 bytes
- Petabyte – 2^{50} or 1024 terabytes
  - sometimes rounded to 10^{15} or 1,000,000,000,000,000 bytes
- Exabyte – 2^{60} or 1024 petabytes
  - Sometimes rounded to 10^{18} or 1,000,000,000,000,000,000 bytes

Review: Moore's Law?

- In 1965, Intel's Gordon Moore predicted that the number of transistors that can be integrated on single chip would double about every two years

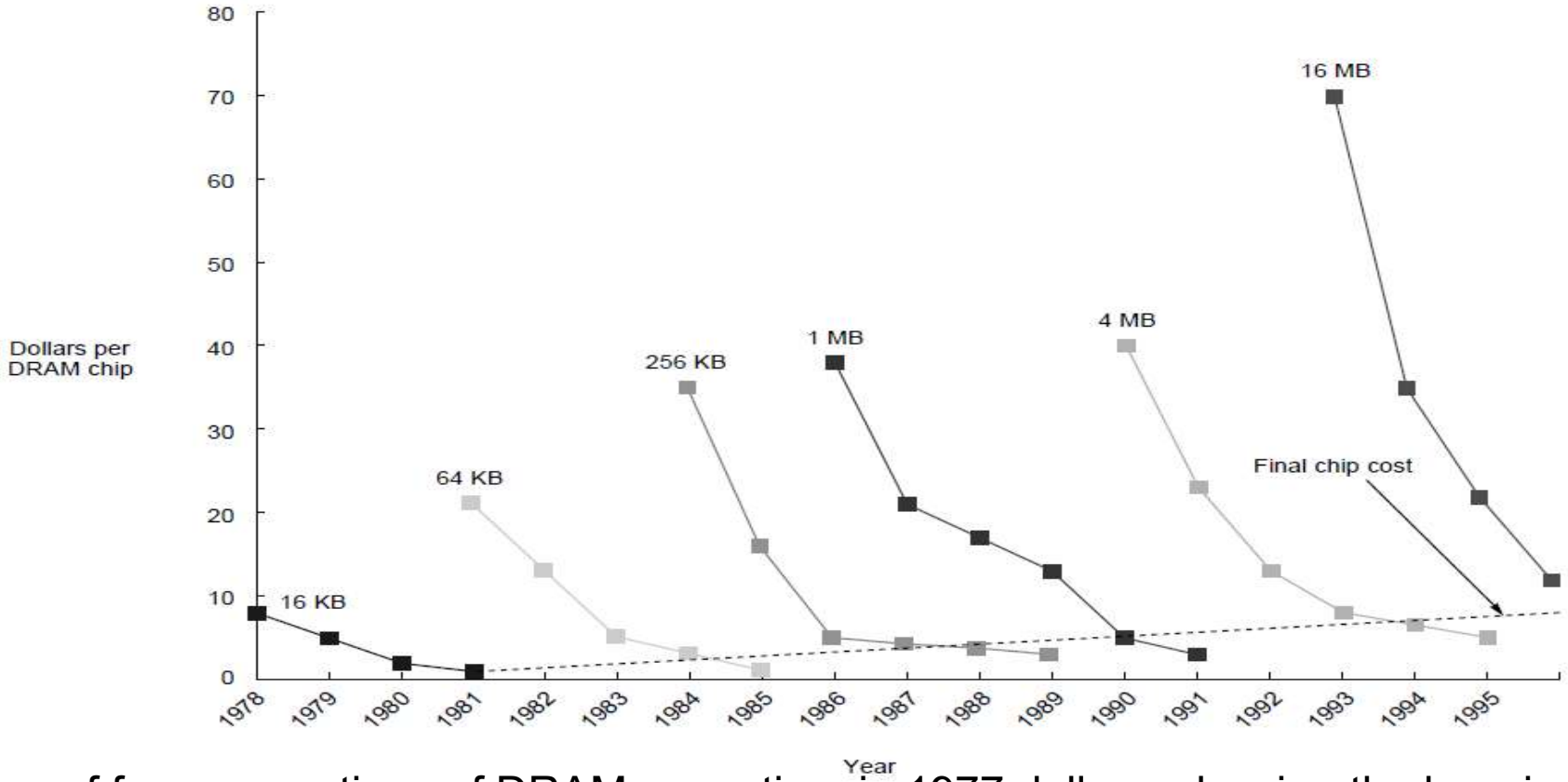
Dual Core Itanium with 1.7B transistors

feature size & die size



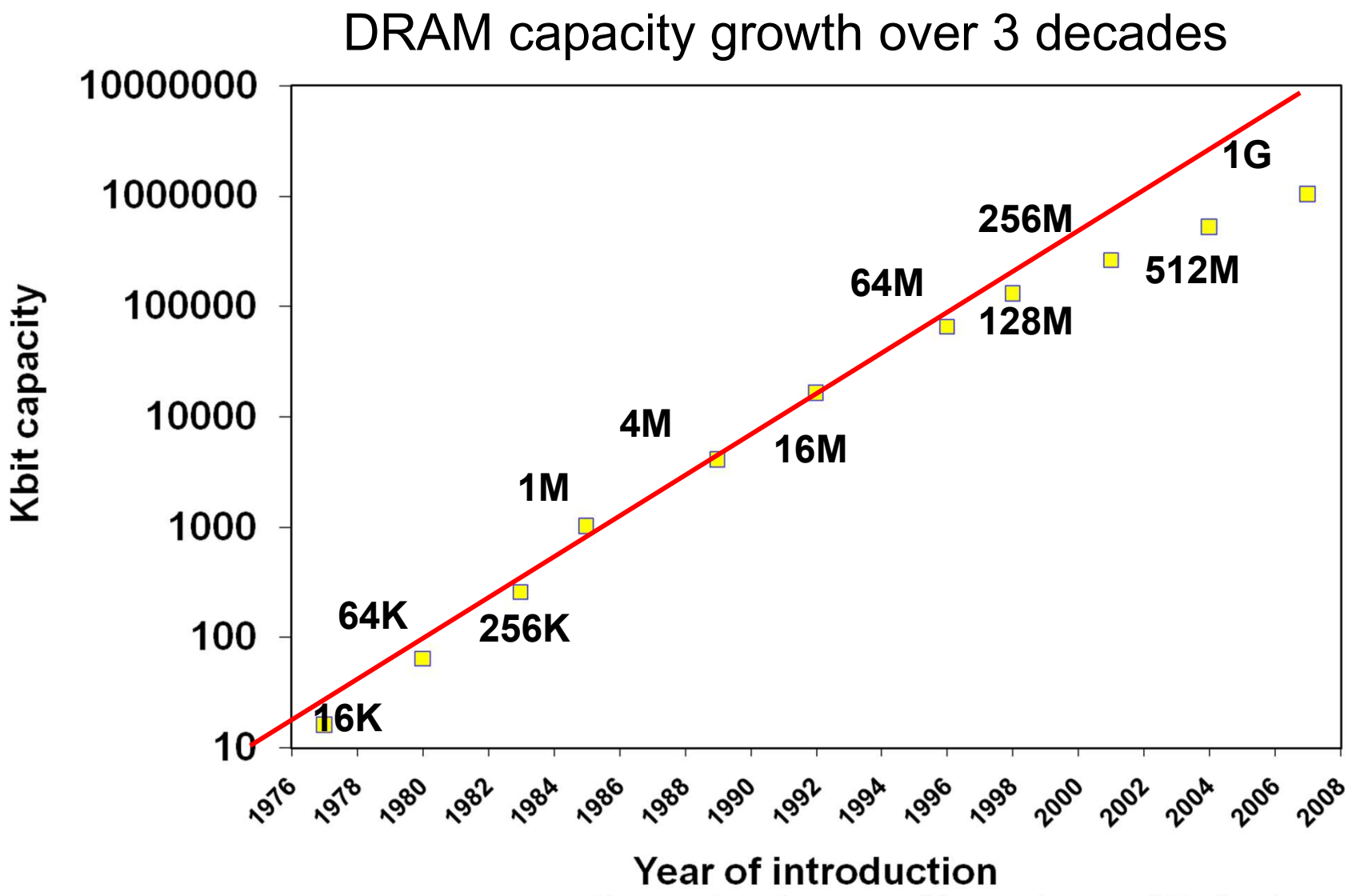
Intel ®

Prices of DRAMs over time in 1977 dollars



Prices of four generations of DRAMs over time in 1977 dollars, showing the learning curve at work. A1977 dollar is worth about $2.44 in 1995; most of this inflation occurred in the period of 1977–82, during which the value changed to $1.61. The cost of a **megabyte** of memory has dropped incredibly during this period, from over **$5000** in 1977 to just over **$6** in 1995 (in 1977 dollars)

Another Example of Moore's Law Impact

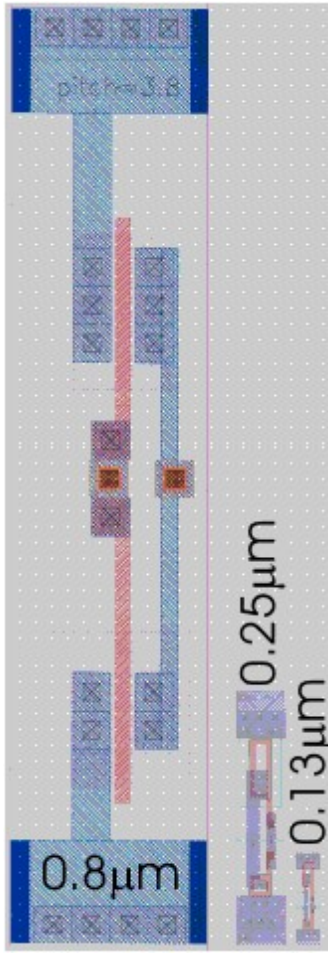


The Computer Revolution

- Progress in computer technology
  - Underpinned by Moore's Law
- Makes novel applications feasible
  - Computers in automobiles
  - Cell phones
  - Human genome project
  - World Wide Web
  - Search Engines
- Computers are pervasive
  - IOT

Technology Scaling

- Key enabler for smaller, faster and more power efficient computing systems
- Technology scaling has a threefold objective:
  - Increase the transistor density
  - Reduce the gate delay
  - Reduce the power consumption
    - Device dimensions (lateral and vertical) and voltages are reduced by $1/\alpha$ (~0.7)
- Technology generation spans 2-3 years



Technology Scaling Roadmap (ITRS)

- Electronics technology continues to evolve
  - Increased capacity and performance; reduced cost

Year	2006	2008	2010	2012	2014	2017	2019
Feature size (nm)	90	65	45	32	22	14/10	7
Intg. Capacity (BT)	2	4	6	16	32	?	?

  - Fun facts about 45nm transistors
    1. 30 million can fit on the head of a pin
    1. You could fit more than 2,000 across the width of a human hair
    1. If **car prices** had fallen at the same rate as the price of a single transistor has since 1968, a new car today would cost about ?
    1. 1 cent

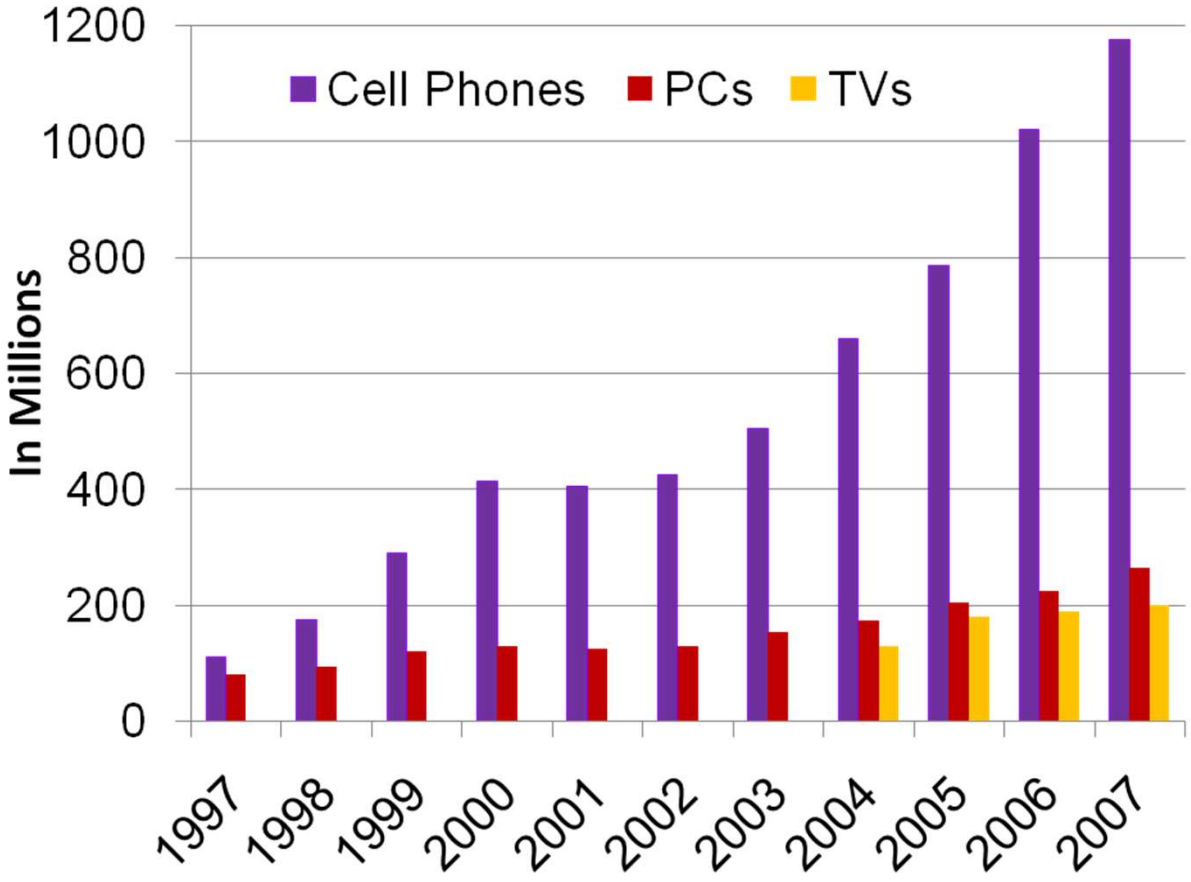
Embedded Processor Characteristics

The largest class of computers spanning the widest range of applications and performance

- Often have minimum performance requirements. Examples?
- Often have stringent limitations on cost. Examples?
- Often have stringent limitations on power consumption. Examples?
- Often have low tolerance for failure. Examples?

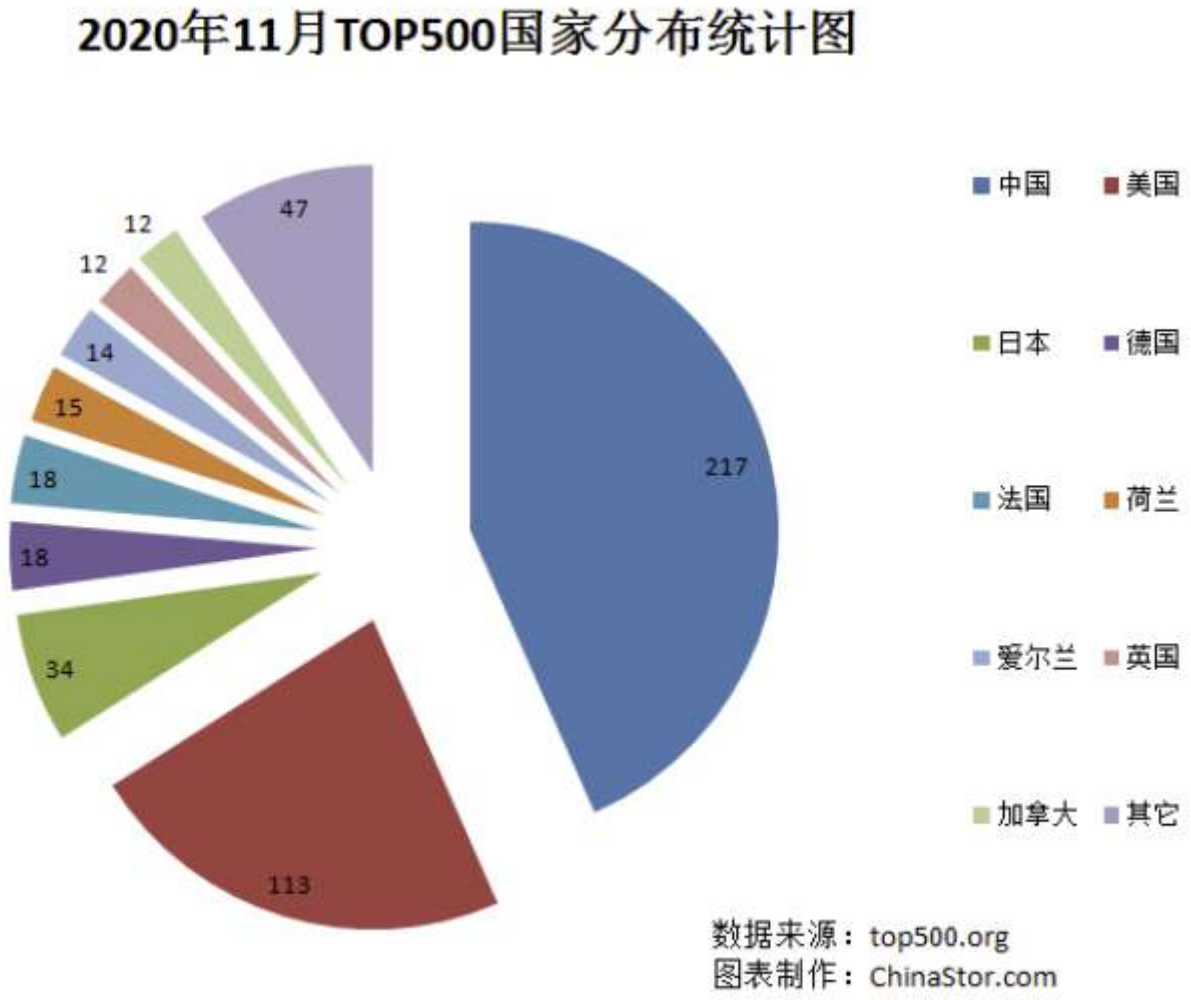
The Processor Market

embedded growth >> desktop growth



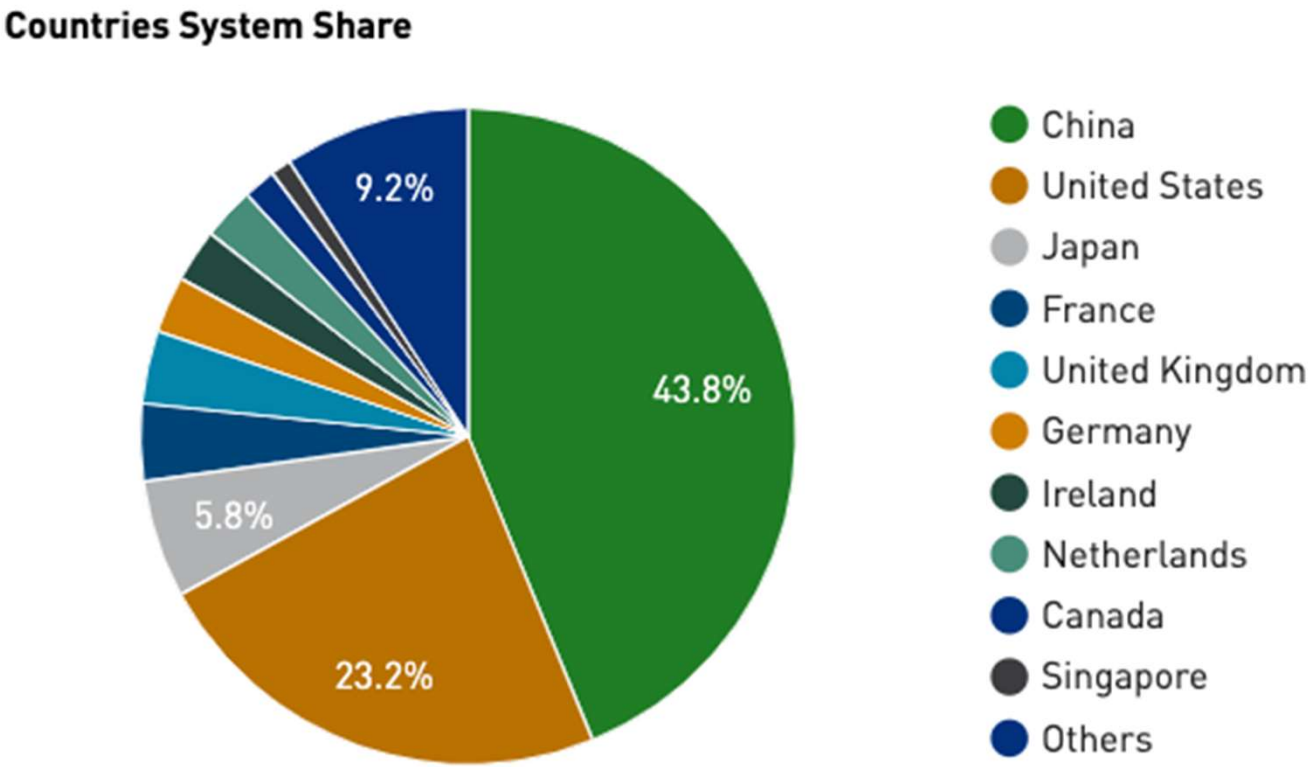
- Where else are embedded processors found?

超算TOP500 解析

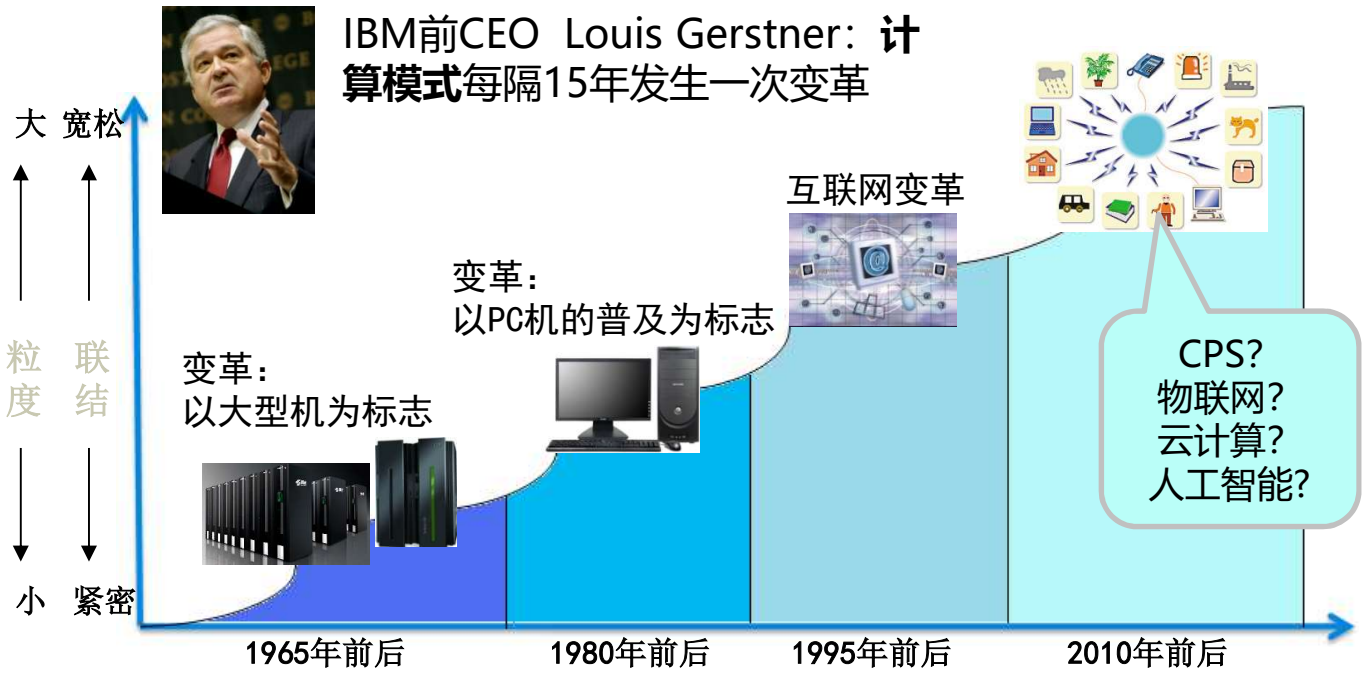


http://www.chinastor.com/hpc-top500/111USS2020.html

超算TOP500 解析


Countries System Share
9.2%
43.8%
5.8%
23.2%
China
United States
Japan
France
United Kingdom
Germany
Ireland
Netherlands
Canada
Singapore
Others

计算模式每15年周期性革命性变革



主观题 1分

设置

课堂讨论：

计算模式每15年周期性革命的规律的技术原因

是什么？

正常使用主观题需2.0以上版本雨课堂

作答

多选题 1分

设置

以下哪些超级计算机中国制造的

A Summit

B Sierra

C Sunway Taihulight

D Tianhe-2A

提交

2020年11月TOP500前十名 www.chinastor.com			
本届排名	上届排名	名称	国家
1	1	Supercomputer Fugaku	日本
2	2	Summit	美国
3	3	Sierra	美国
4	4	Sunway TaihuLight	中国
5	7	Selene	美国
6	5	Tianhe-2A	中国
7		JUWELS Booster Module	德国
8	6	HPC5	意大利
9	8	Frontera	美国
10		Dammam-7	沙特

单选题 1分

设置

可以通过以下的指标来说明计算机系统的性能

A CPU的主频

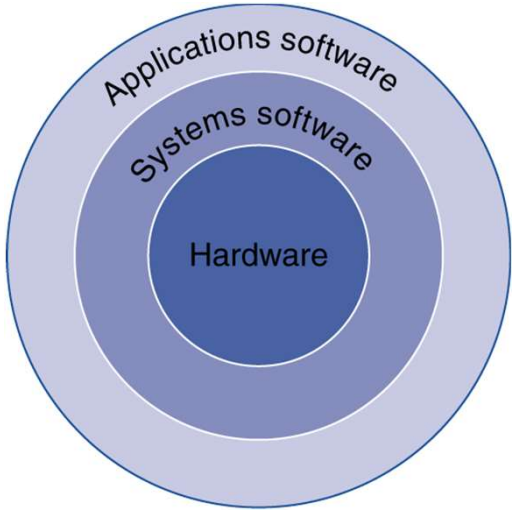
B 内存的大小

C 硬盘大小

D 以上均不能准确刻画

提交

Below Your Program



- Application software
  - Written in high-level language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers

Levels of Program Code

- High-level language program (in C)

```
void swap (int v[], int k)
{       int temp;
        temp = v[k];
        v[k] = v[k+1];
        v[k+1] = temp;
}
```

one-to-many

C compiler

- Assembly language program (for MIPS)

```
swap:   sll     $2, $5, 2
        add     $2, $4, $2
        lw      $15, 0($2)
        lw      $16, 4($2)
        sw      $16, 0($2)
        sw      $15, 4($2)
        jr      $31
```

one-to-one

assembler

- Machine (object, binary) code (for MIPS)

```
000000 00000 00101 0001000010000000
000000 00100 00010 0001000000100000
. . .
```

Advantages of Higher-Level Languages ?

- Higher-level languages
  1. Allow the programmer to think in a more natural language and for their intended use (Fortran for scientific computation, Cobol for business programming, Lisp for symbol manipulation, Java for web programming, …)
  2. Improve programmer productivity – more understandable code that is easier to debug and validate
  3. Improve program maintainability
  4. Allow programs to be independent of the computer on which they are developed (compilers and assemblers can translate high-level language programs to the binary instructions of any machine)
  5. Emergence of optimizing compilers that produce very efficient assembly code optimized for the target machine

- As a result, very little programming is done today at the assembler level

Instruction Set Architecture (ISA)

- ISA, or simply architecture – the abstract interface between the hardware and the lowest level software that encompasses all the information necessary to write a machine language program, including instructions, registers, memory access, I/O, …
  - Enables implementations of varying cost and performance to run identical software
- The combination of the basic instruction set (the ISA) and the operating system interface is called the application binary interface (ABI)
  - ABI – The user portion of the instruction set plus the operating system interfaces used by application programmers.
    - Defines a standard for binary portability across computers.

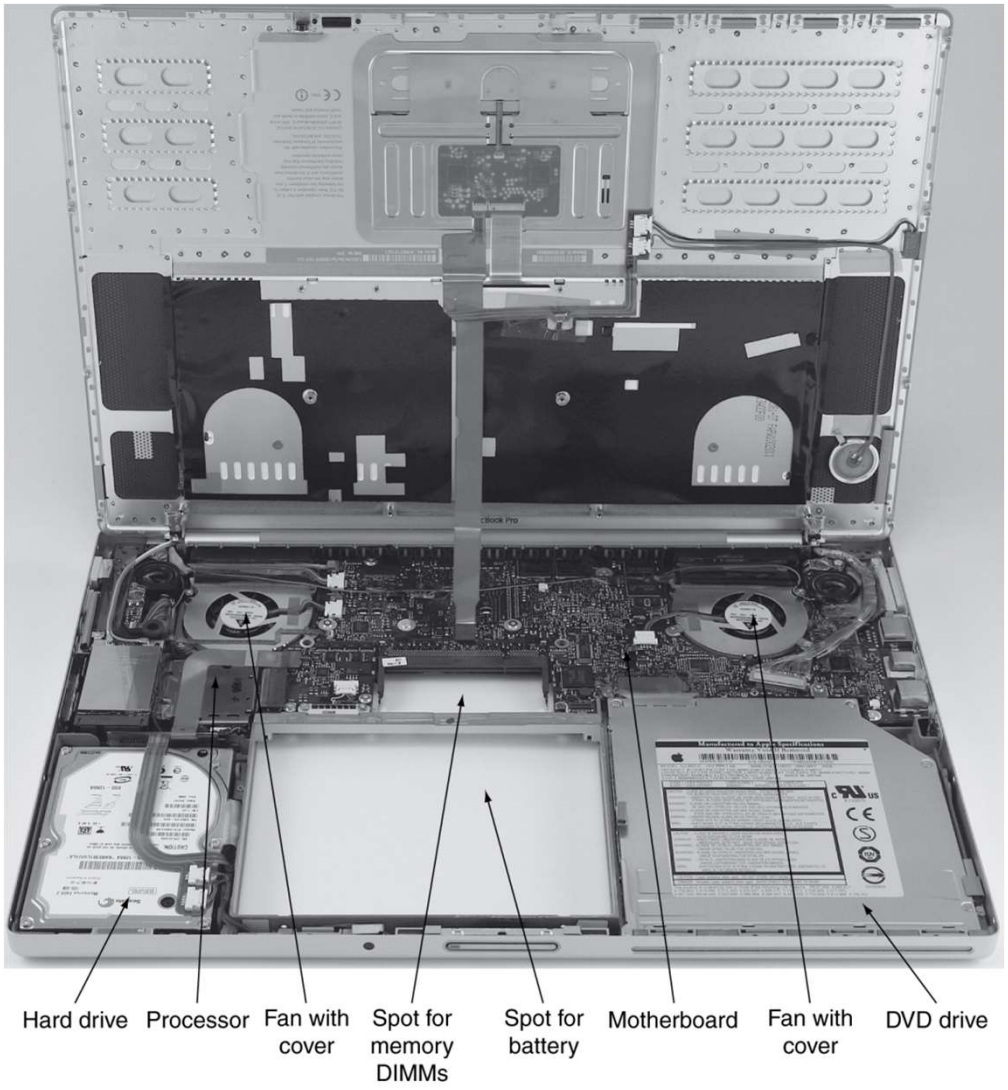
ABI实现了二进制代码级兼容性

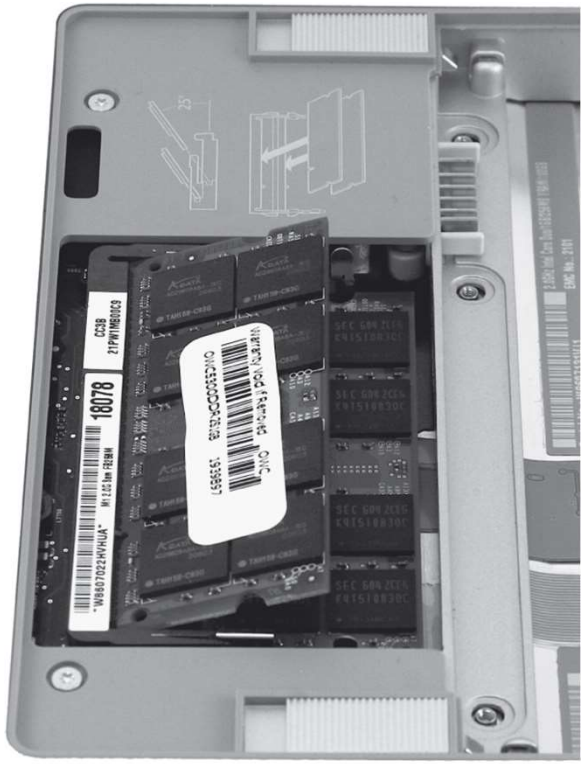
Application Binary Interface (ABI)

DOS OS

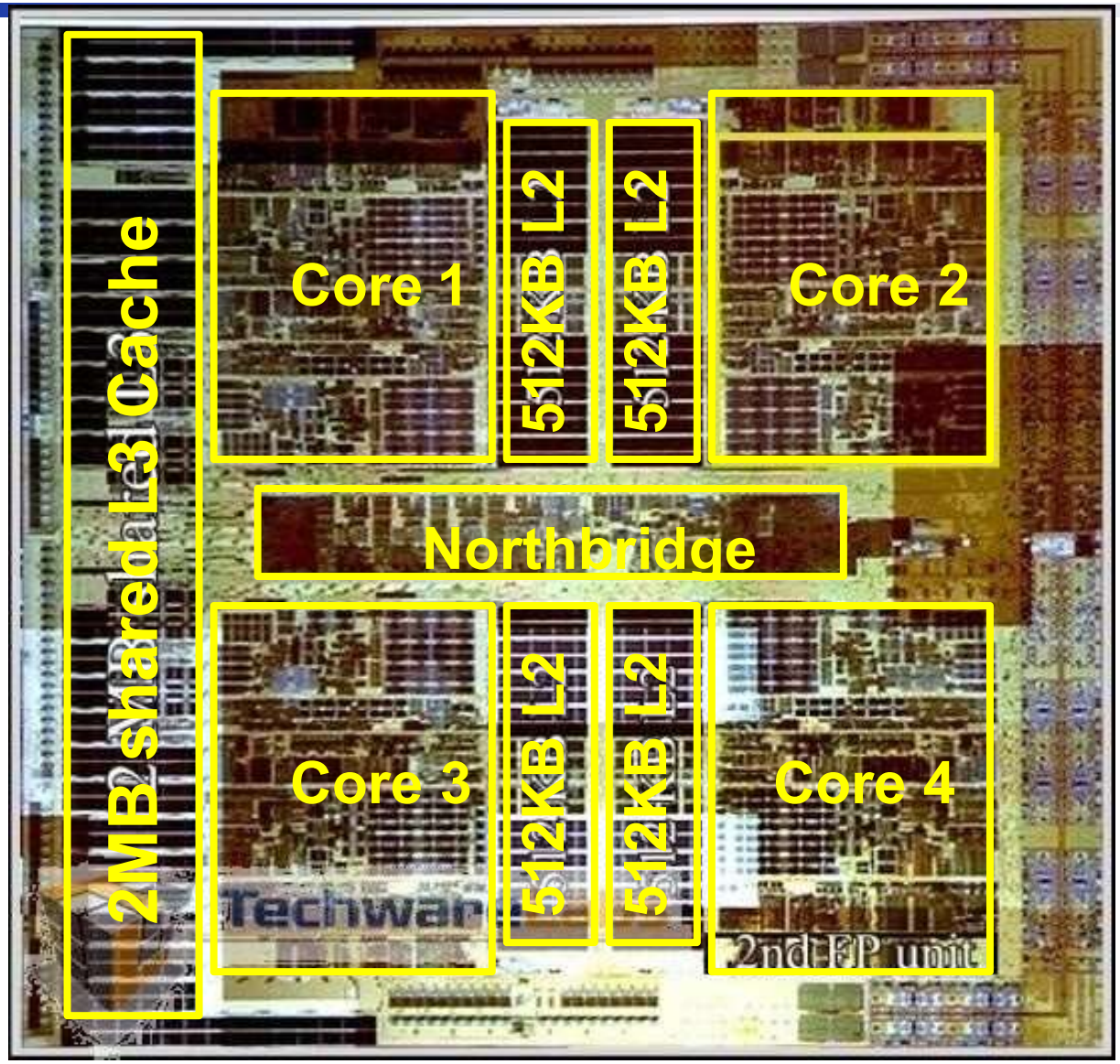
指令是如何执行的呢？ 指令是如何实现其对应的功能?

Opening the Box





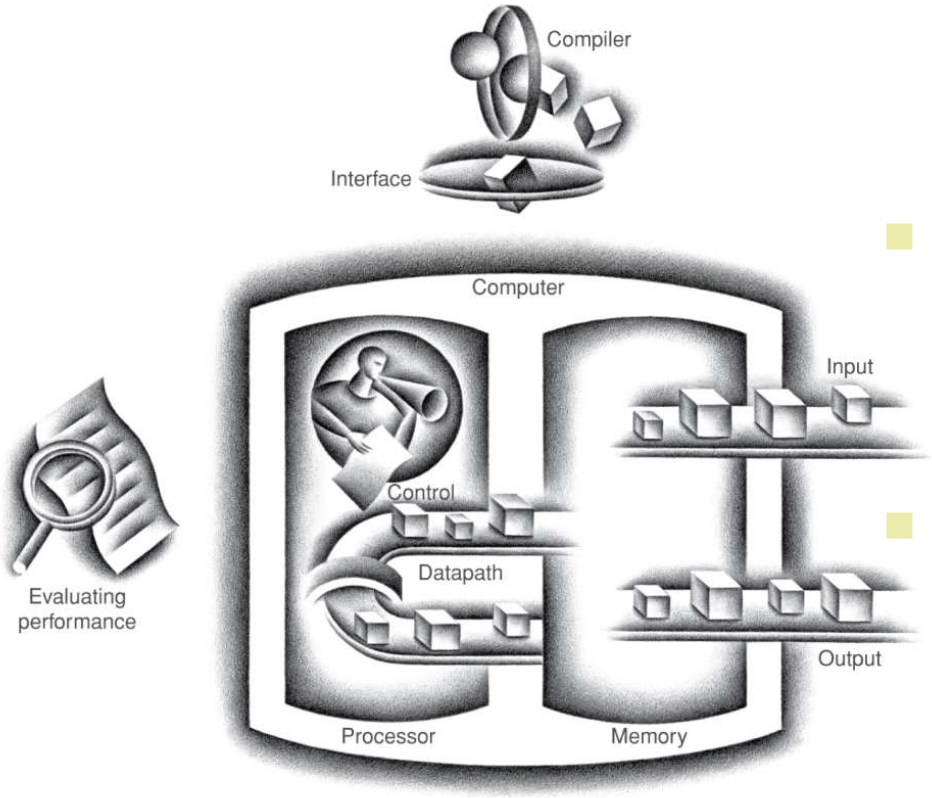
Inside the Processor



- AMD's Barcelona Multicore Chip
- Four out-of-order cores on one chip
- 1.9 GHz clock rate
- 65nm technology
- Three levels of caches (L1, L2, L3) on chip
- Integrated Northbridge

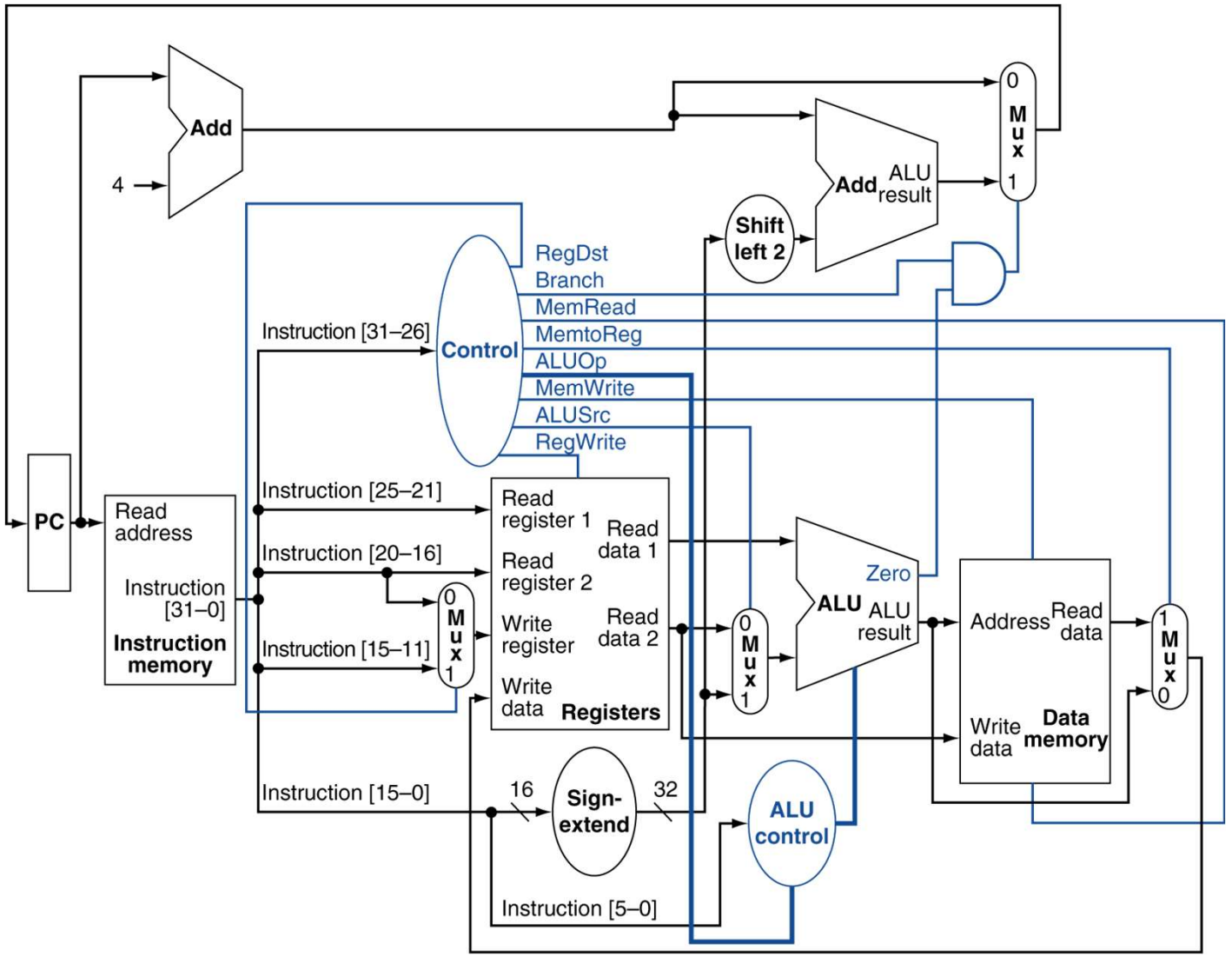
Components of a Computer

The BIG Picture

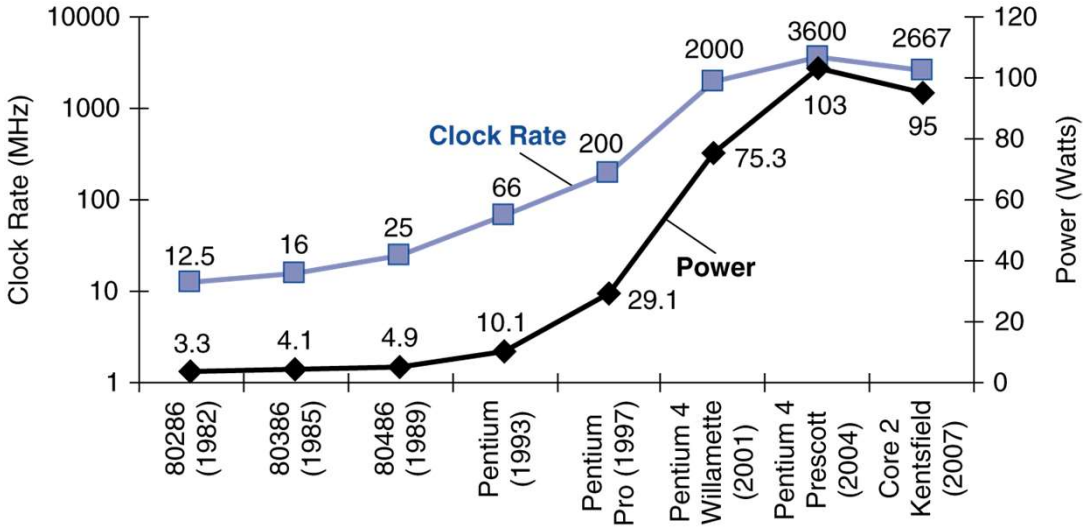


- Five main components
  - Input, output, memory, datapath, control
  - Datapath + Control = Processor
- Same components for all kinds of computers
  - Desktop, server, embedded
- Input/output includes
  - User-interface devices, storage devices, network adapters

Inside the Processor



Power Trends and Power Wall

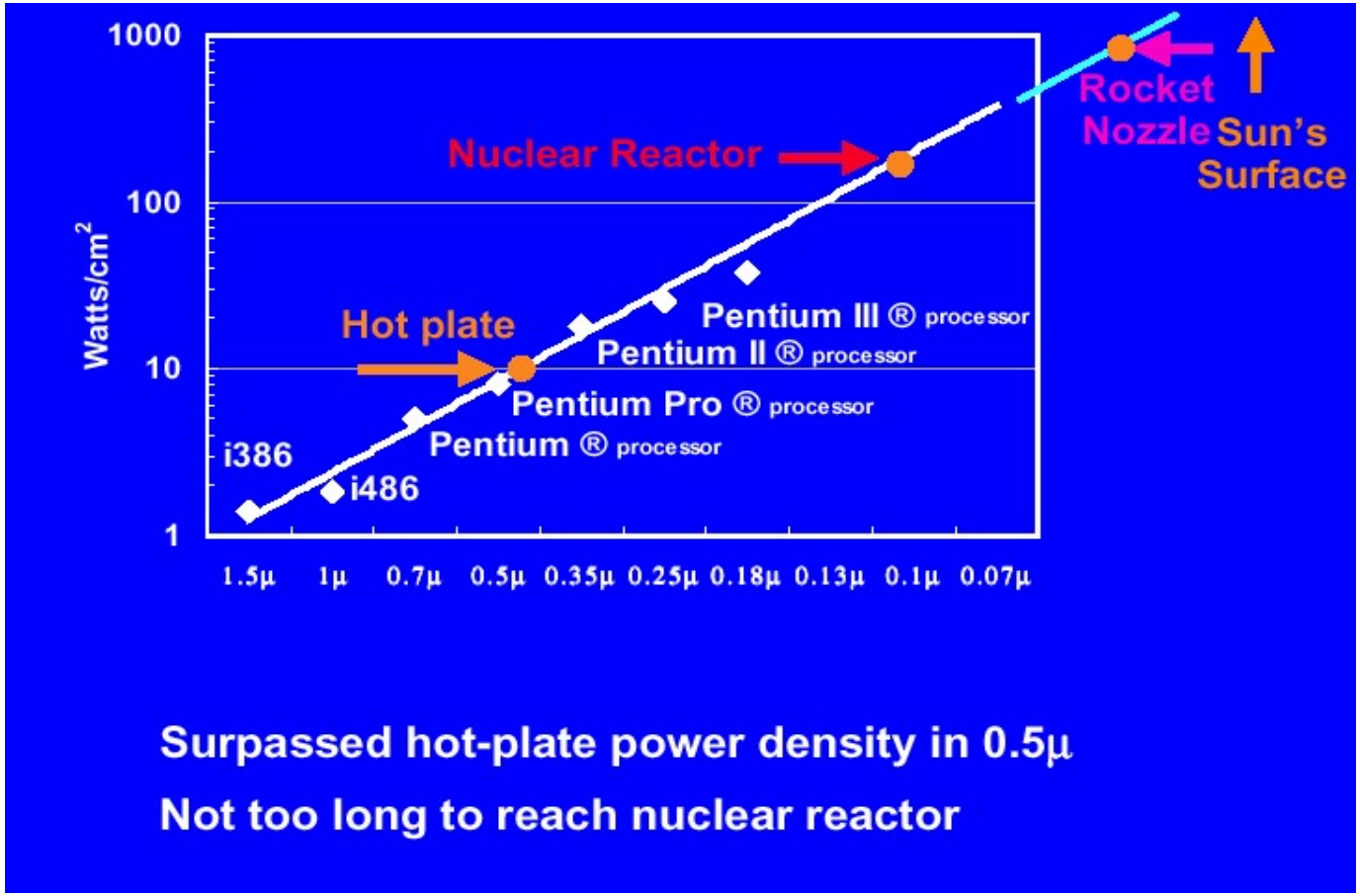


- In CMOS IC technology

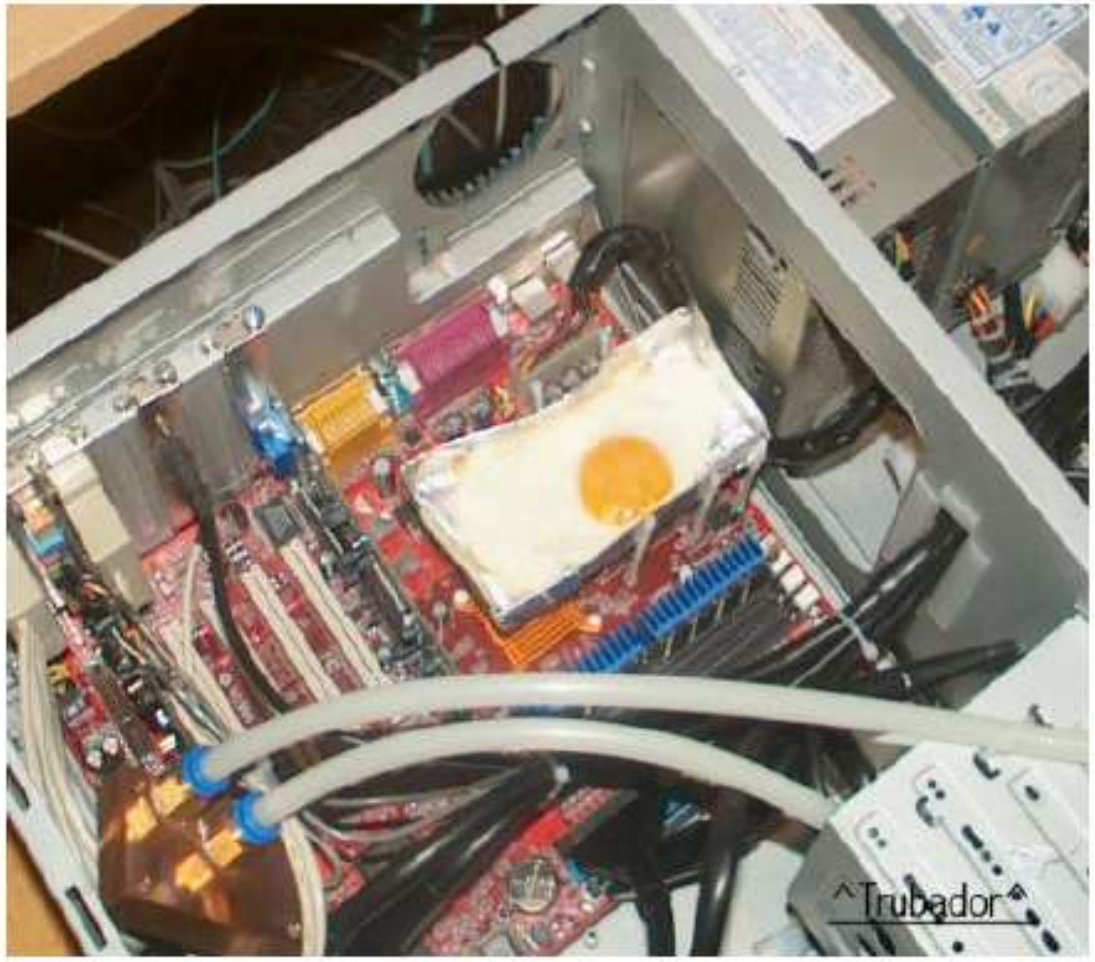
$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

$\times 30$ | $5\text{V} \rightarrow 1\text{V}$ | $\times 1000$

Power Density Getting Worse



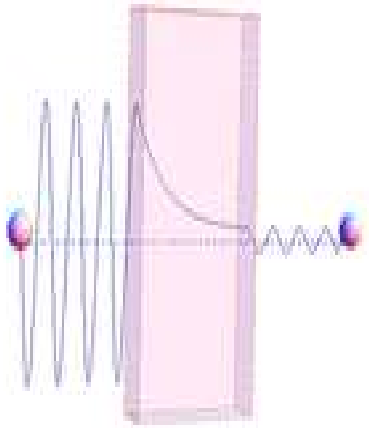
Surpassed hot (kitchen) plate … why not use it?



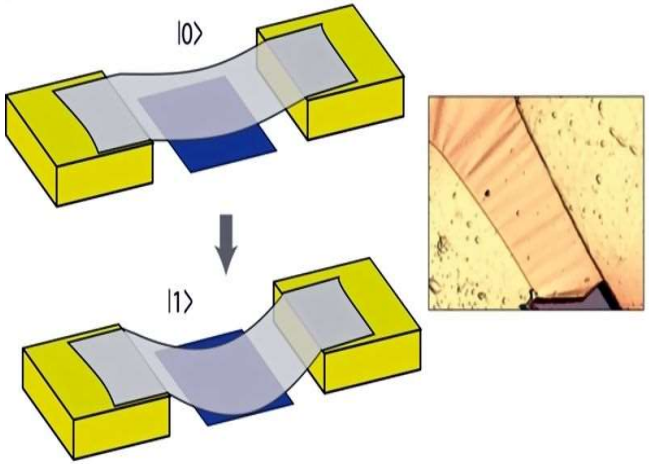
散热\隧道效应

1 风扇空气散热

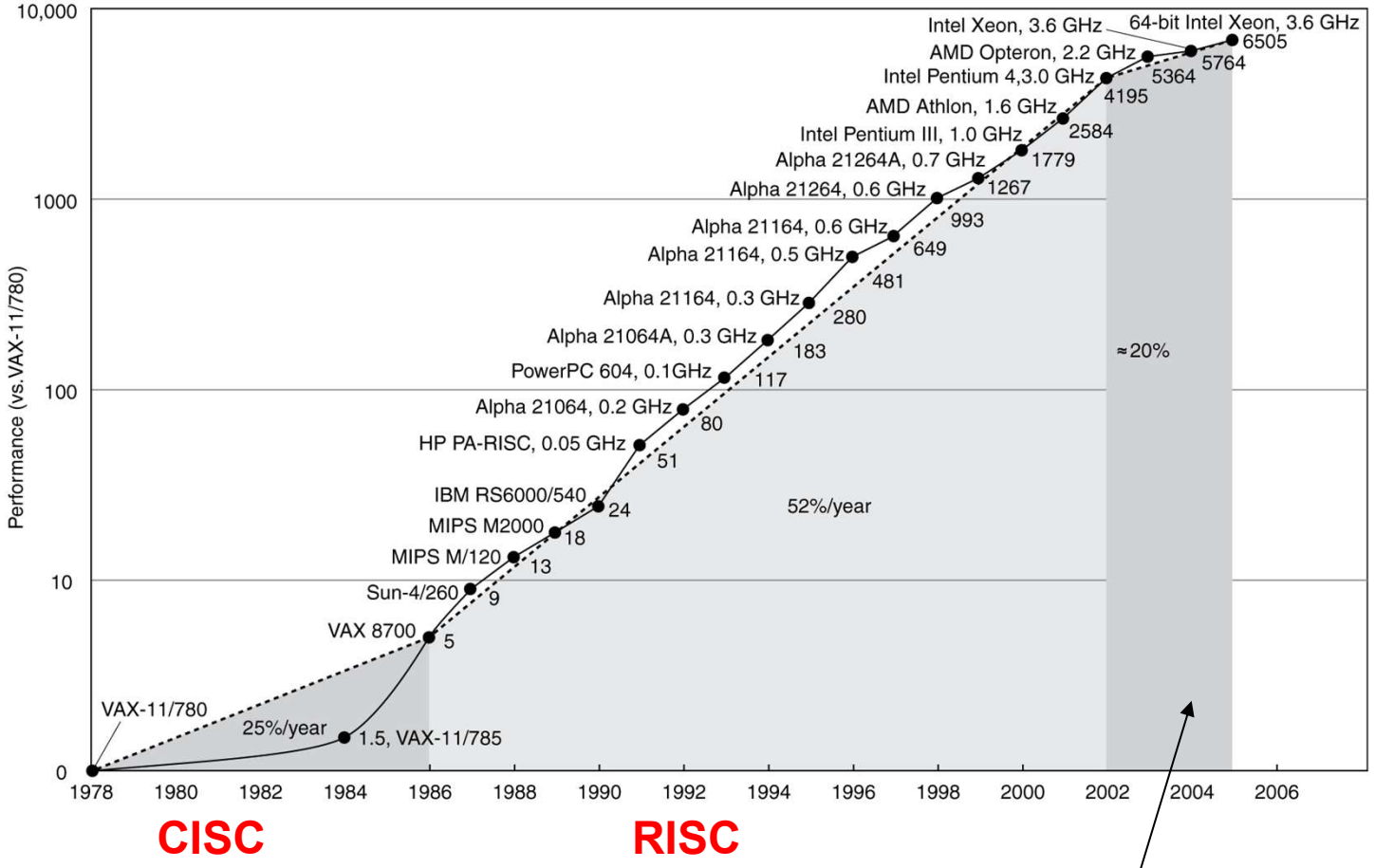
2 水冷散热方式



3 超级玩家的液氮散热



Uniprocessor Performance



Constrained by power, instruction-level parallelism, memory latency

The Sea Change

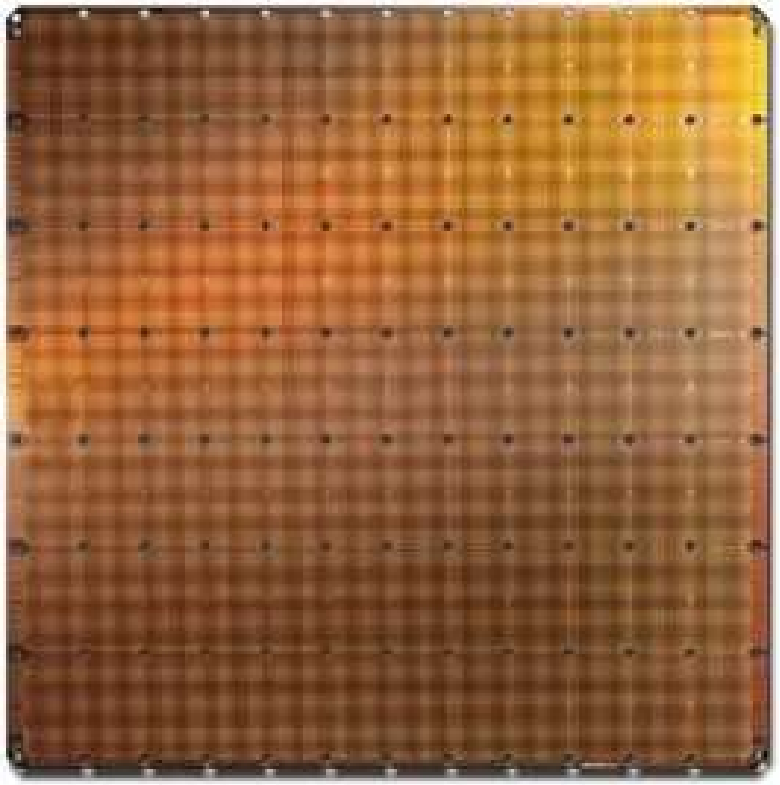
- The power challenge has forced a change in the design of microprocessors
  1. Since 2002 the rate of improvement in the response time of programs on desktop computers has slowed from a factor of 1.5 per year to less than a factor of 1.2 per year
- As of 2006 all desktop and server companies are shipping microprocessors with **multiple** processors – cores – per chip

Product	AMD Barcelona	Intel Nehalem	IBM Power 6	Sun Niagara 2
Cores per chip	4	4	2	8
Clock rate	2.5 GHz	~2.5 GHz?	4.7 GHz	1.4 GHz
Power	120 W	~100 W?	~100 W?	94 W

- Plan of record is to double the number of cores per chip per generation (about every two years)

参考材料

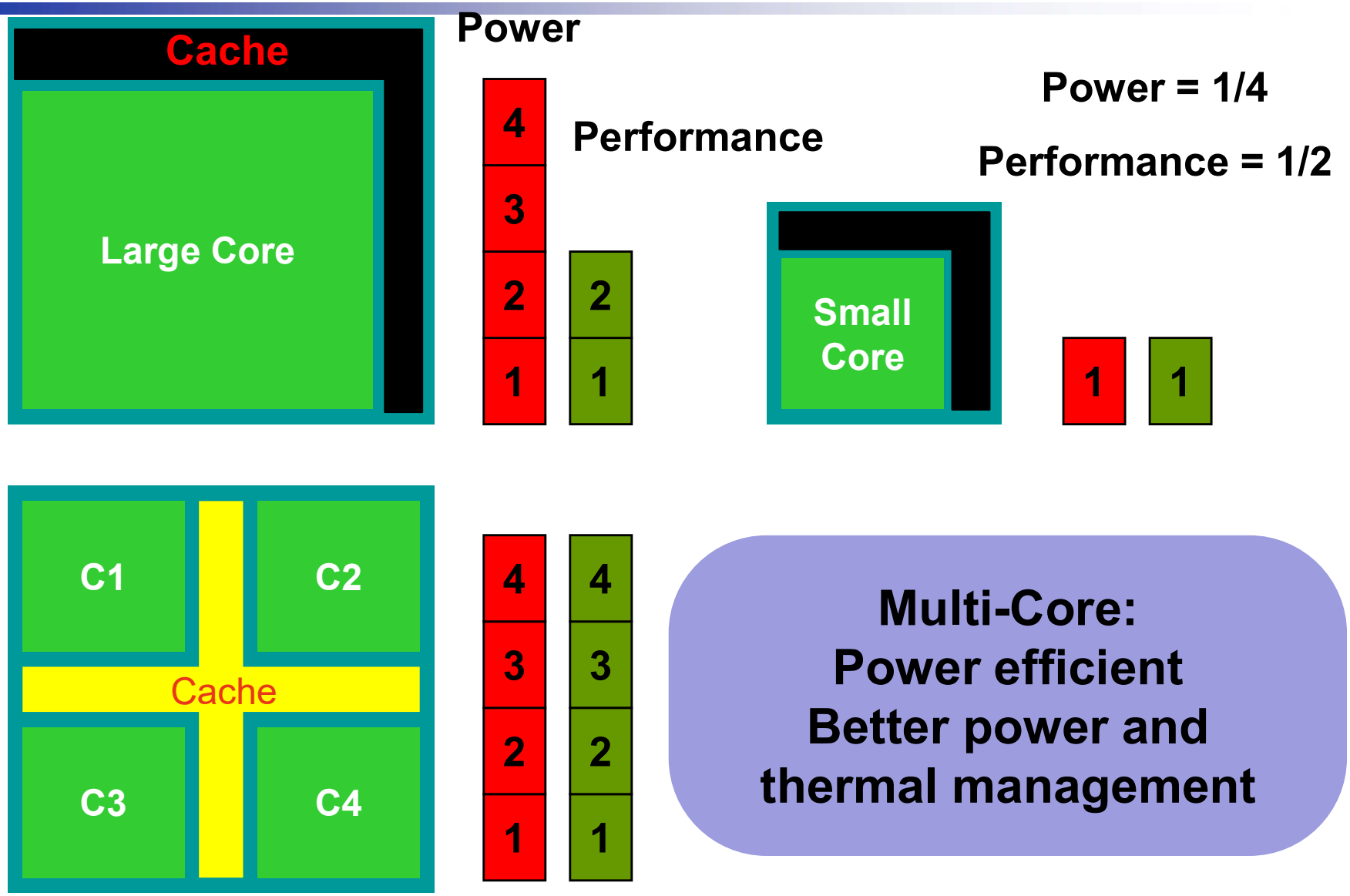
1.2 万亿晶体管：全球最大 AI 芯片超越 GPU 1 万倍（2020年11月21日）



Cerebras WSE
1.2 Trillion transistors
46,225 mm² silicon

Largest GPU
21.1 Billion transistors
815 mm² silicon

Why Multiple Cores on a Chip?



Multi-Core:
Power efficient
Better power and thermal management

主观题 1分

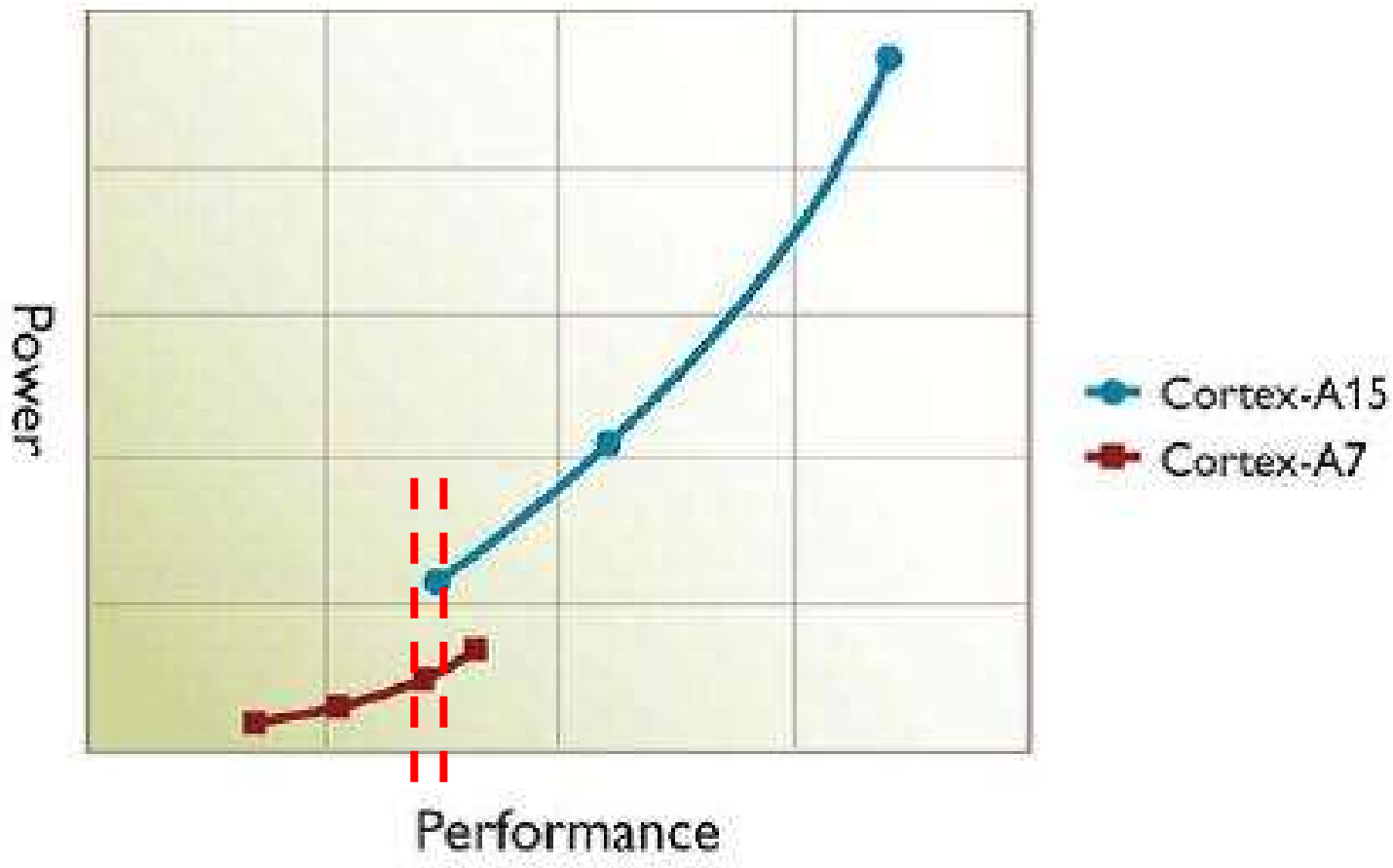
设置

为什么小的CPU核可具备更高的性能/功耗比?

正常使用主观题需2.0以上版本雨课堂

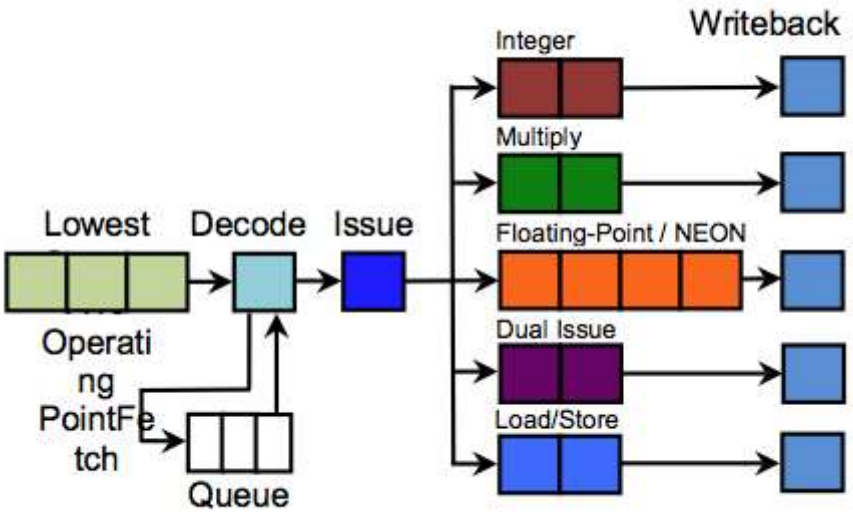
作答

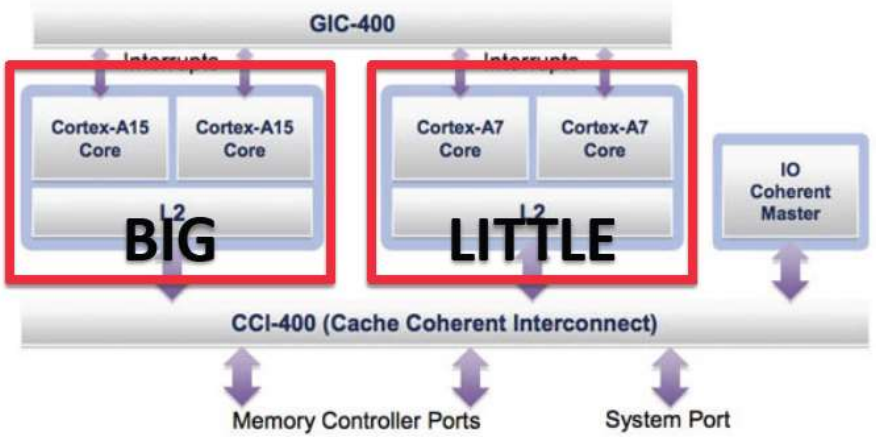
Big Core vs Little Core

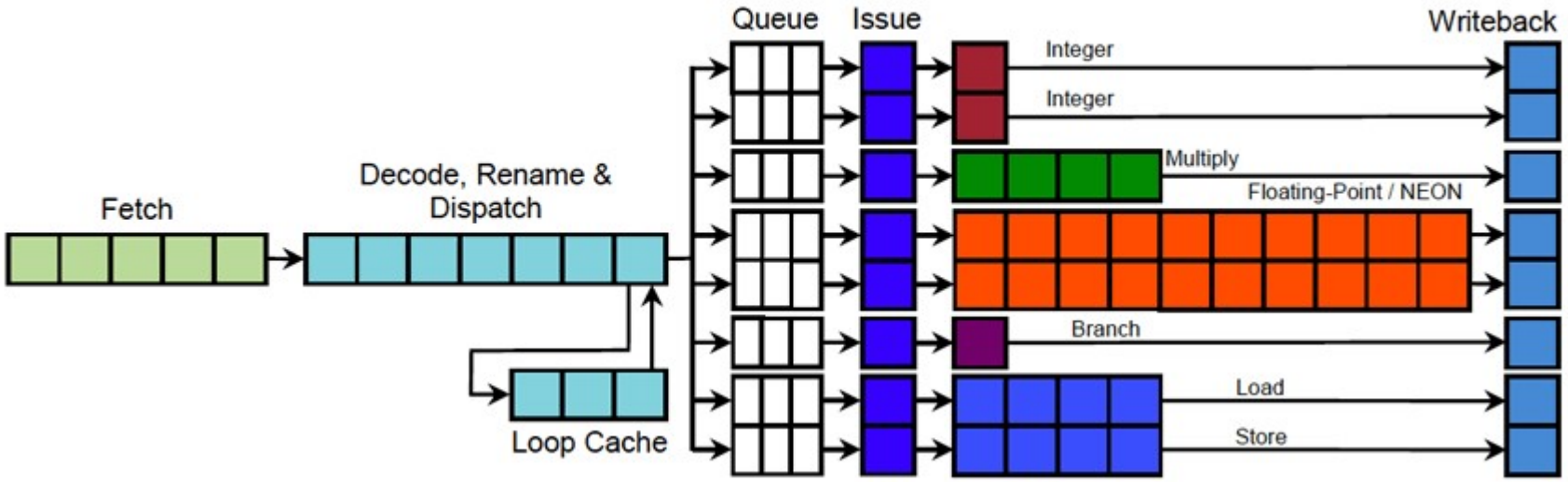


$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

Inside the ARM big.little Core







单选题 1分

设置

只要两台计算机的采用相同的CPU，在其中一台计算机上可以运行的程序就一定可以迁移到另一台计算机运行

A 正确

B 错误

提交

Performance Metrics

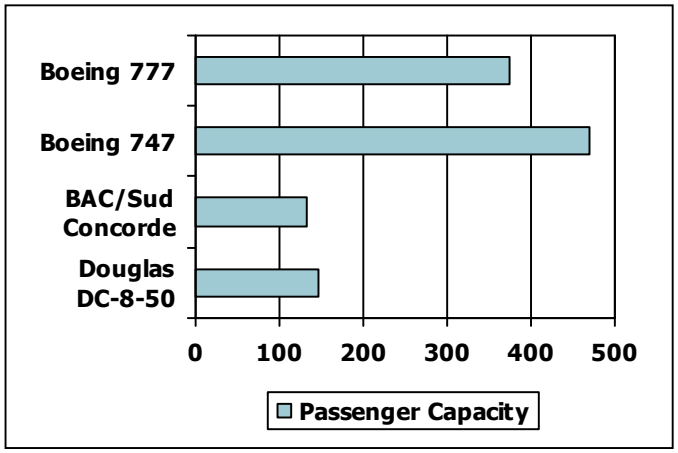
- Purchasing perspective
  - given a collection of machines, which has the
    - best performance ?
    - least cost ?
    - best cost/performance?
- Design perspective
  - faced with design options, which has the
    - best performance improvement ?
    - least cost ?
    - best cost/performance?
- Both require
  - basis for comparison
  - metric for evaluation
- Our goal is to understand what factors in the architecture contribute to overall system performance and the relative importance (and cost) of these factors

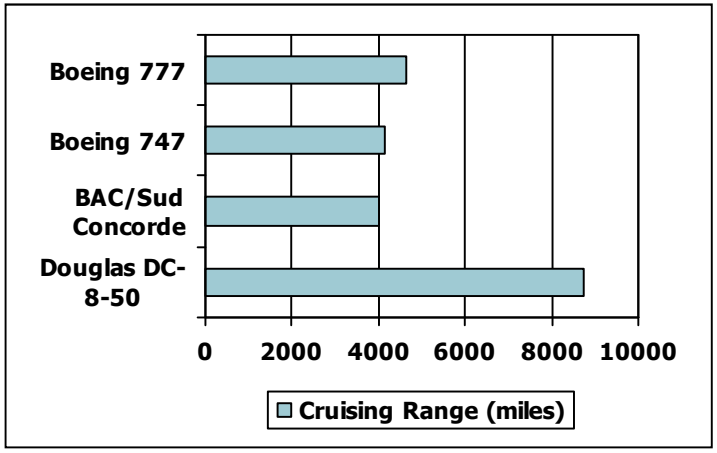
Understanding Performance

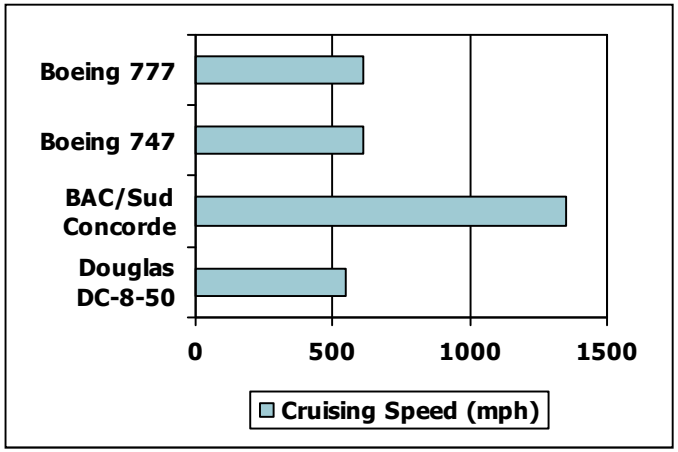
- Algorithm
  - Determines number of operations executed
- Programming language, compiler, ISA
  - Determine number of machine instructions executed per operation
- Processor and memory system
  - Determine how fast instructions are executed
- I/O system (including OS)
  - Determines how fast I/O operations are executed

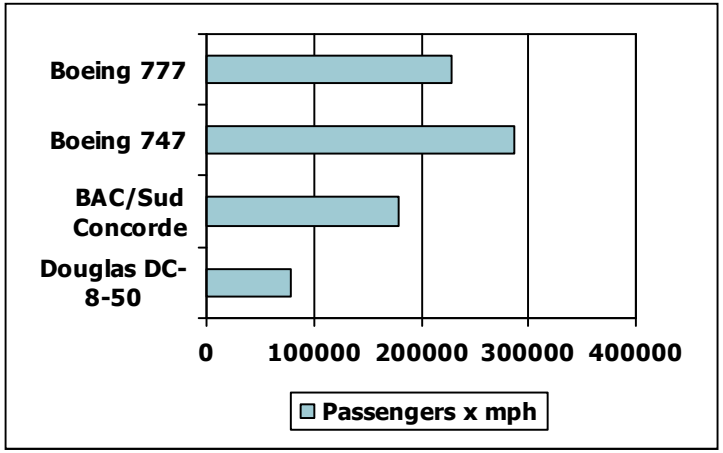
Defining Performance

- Which airplane has the best **performance**?









Response Time and Throughput

- Response time
  - How long it takes to do a task
    - Important to individual users
- Throughput
  - Total work done per unit time
    - e.g., tasks/transactions/… per hour
    - Important to data center managers
- How are response time & throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?
- We'll focus on response time for now…

Relative Performance

- Define Performance = 1/Execution Time
- "X is n time faster than Y"

$$\text{Performance}_X / \text{Performance}_Y = \text{Execution time}_Y / \text{Execution time}_X = n$$

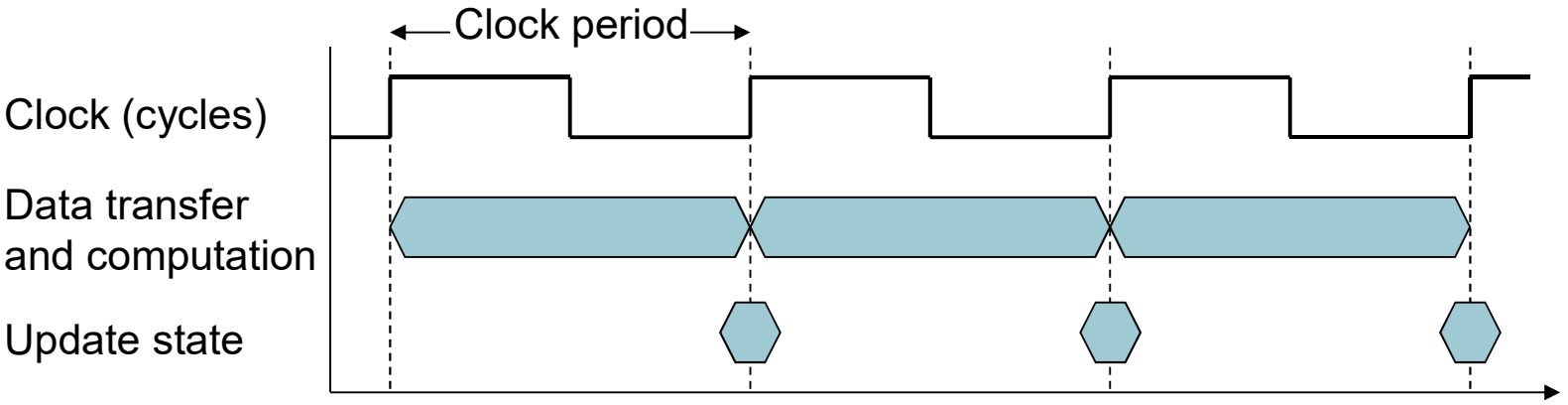
- Example: time taken to run a program
  - 10s on A, 15s on B
  - Execution Time_B / Execution Time_A = 15s / 10s = 1.5
  - So A is 1.5 times faster than B

Measuring Execution Time

- Execution time: seconds/program
- Elapsed time (wall clock time)
  - Total response time, including all aspects
    - Processing, I/O, OS overhead, idle time
  - Determines system performance
- CPU time
  - Time spent processing a given job
    - Discounts I/O time, other jobs' shares
  - Comprises user CPU time and system CPU time
  - Different programs are affected differently by CPU and system performance

CPU Clocking: Review

- Operation of digital hardware governed by a constant-rate clock



- Clock period: duration of a clock cycle
  - e.g., 250ps = 0.25ns = 250×10^{-12}s
- Clock frequency (rate): cycles per second
  - e.g., 4.0GHz = 4000MHz = 4.0×10^{9}Hz

CPU Clocking: Review

- Clock rate (clock cycles per second in MHz or GHz) is inverse of clock cycle time (clock period)

$$CC = 1 / CR$$

10 nsec clock cycle => 100 MHz clock rate

5 nsec clock cycle => 200 MHz clock rate

2 nsec clock cycle => 500 MHz clock rate

1 nsec (10^{-9}) clock cycle => 1 GHz (10^{9}) clock rate

500 psec clock cycle => 2 GHz clock rate

250 psec clock cycle => 4 GHz clock rate

200 psec clock cycle => 5 GHz clock rate

CPU Time

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

- Performance improved by
  - Reducing number of clock cycles
  - Increasing clock rate
- Hardware designer must often trade off clock rate against cycle count
  - Many techniques that decrease the number of clock cycles also increase the clock cycle time

CPU Time Example

- A program runs on computer A with a 2 GHz clock in 10 seconds. Unfortunately, to accomplish this, computer B will require 1.2 times as many clock cycles as computer A to run the program. What clock rate must computer B run at to run this program in 6 seconds?

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6\text{s}}$$

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$

$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

$$\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6\text{s}} = \frac{24 \times 10^9}{6\text{s}} = 4\text{GHz}$$

Instruction Count and CPI

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix

CPI Example

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster, and by how much?

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$
$$= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$

A is faster…

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$
$$= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2$$

…by this much

CPI in More Detail

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

  - Weighted average CPI

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \left(\text{CPI}_i \times \underbrace{\frac{\text{Instruction Count}_i}{\text{Instruction Count}}}_{\text{Relative frequency}} \right)$$

CPI Example

- Alternative compiled code sequences using instructions in classes A, B, C. What is avg. CPI?

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5
  - Clock Cycles
    = 2×1 + 1×2 + 2×3
    = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles
    = 4×1 + 1×2 + 1×3
    = 9
  - Avg. CPI = 9/6 = 1.5

Performance Summary

The BIG Picture

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

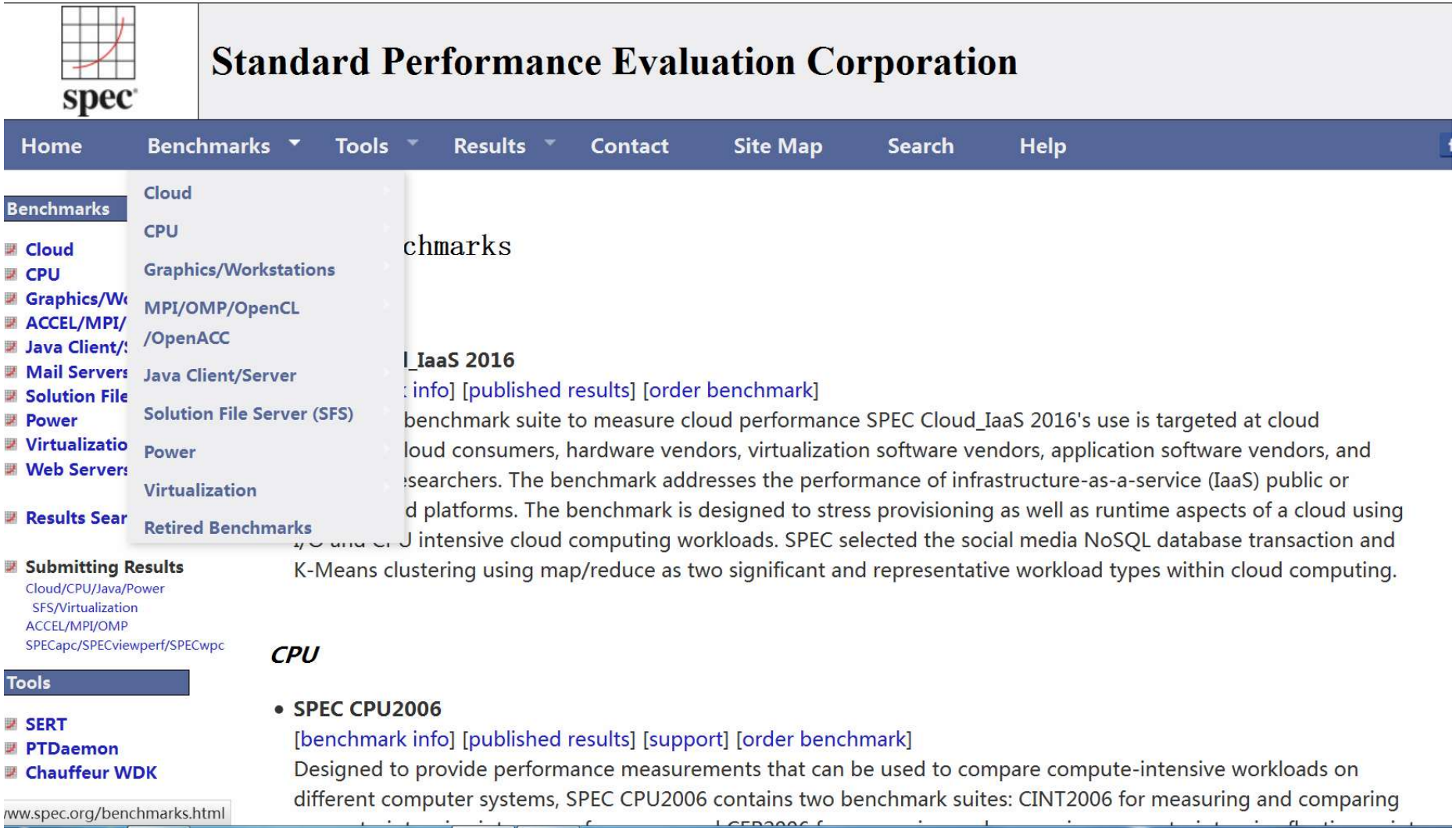
	Instruction_ count	CPI	clock_cycle
Algorithm	X	X	
Programming language	X	X	
Compiler	X	X	
ISA	X	X	X
Core organization		X	X
Technology			X

Workloads and Benchmarks

- Benchmarks – a set of programs that form a “workload” specifically chosen to measure performance
- SPEC (System Performance Evaluation Cooperative) creates standard sets of benchmarks starting with SPEC89. The SPEC CPU2006 which consists of 12 integer benchmarks (CINT2006) and 17 floating-point benchmarks (CFP2006).

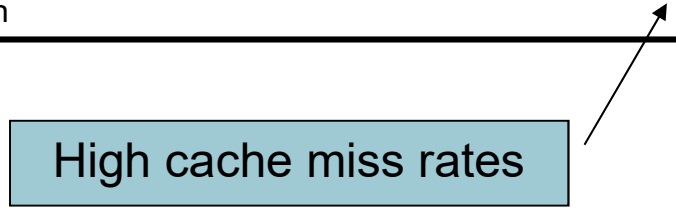
  www.spec.org

- There are also benchmark collections for power workloads (SPECpower_ssj2008), for mail workloads (SPECmail2008), for multimedia workloads (mediabench), …


Standard Performance Evaluation Corporation
Home
Benchmarks
Tools
Results
Contact
Site Map
Search
Help
Cloud
CPU
Graphics/Workstations
MPI/OMP/OpenCL /OpenACC
Java Client/Server
Solution File Server (SFS)
Power
Virtualization
Retired Benchmarks
Benchmarks
Cloud
CPU
Graphics/W
ACCEL/MPI/
Java Client/
Mail Server
Solution File
Power
Virtualizatio
Web Server
Results Sear
Submitting Results
Cloud/CPU/Java/Power
SFS/Virtualization
ACCEL/MPI/OMP
SPECapc/SPECviewperf/SPECwpc
Tools
SERT
PTDaemon
Chauffeur WDK
ww.spec.org/benchmarks.html
chmarks
l_IaaS 2016
info] [published results] [order benchmark]
benchmark suite to measure cloud performance SPEC Cloud_IaaS 2016's use is targeted at cloud
loud consumers, hardware vendors, virtualization software vendors, application software vendors, and
esearchers. The benchmark addresses the performance of infrastructure-as-a-service (IaaS) public or
d platforms. The benchmark is designed to stress provisioning as well as runtime aspects of a cloud using
intensive cloud computing workloads. SPEC selected the social media NoSQL database transaction and
K-Means clustering using map/reduce as two significant and representative workload types within cloud computing.
CPU
SPEC CPU2006
[benchmark info] [published results] [support] [order benchmark]
Designed to provide performance measurements that can be used to compare compute-intensive workloads on
different computer systems, SPEC CPU2006 contains two benchmark suites: CINT2006 for measuring and comparing

CINT2006 for Opteron X4 2356

Name	Description	IC × 10^9	CPI	Tc (ns)	Exec time	Ref time	SPECratio
perl	Interpreted string processing	2,118	0.75	0.4	637	9,777	15.3
bzip2	Block-sorting compression	2,389	0.85	0.4	817	9,650	11.8
gcc	GNU C Compiler	1,050	1.72	0.4	24	8,050	11.1
mcf	Combinatorial optimization	336	10.00	0.4	1,345	9,120	6.8
go	Go game (AI)	1,658	1.09	0.4	721	10,490	14.6
hmmer	Search gene sequence	2,783	0.80	0.4	890	9,330	10.5
sjeng	Chess game (AI)	2,176	0.96	0.4	37	12,100	14.5
libquantum	Quantum computer simulation	1,623	1.61	0.4	1,047	20,720	19.8
h264avc	Video compression	3,102	0.80	0.4	993	22,130	22.3
omnetpp	Discrete event simulation	587	2.94	0.4	690	6,250	9.1
astar	Games/path finding	1,082	1.79	0.4	773	7,020	9.1
xalancbmk	XML parsing	1,058	2.70	0.4	1,143	6,900	6.0
Geometric mean							11.7



Comparing and Summarizing Performance

- How do we summarize the performance for benchmark set with a single number?
  - First the execution times are normalized giving the "SPEC ratio" (bigger is faster, i.e., SPEC ratio is the inverse of execution time)
  - The SPEC ratios are then "averaged" using the geometric mean (GM)

$$GM = \sqrt[n]{\prod_{i=1}^{n} \text{SPEC ratio}_i}$$

- Guiding principle in reporting performance measurements is reproducibility – list everything another experimenter would need to duplicate the experiment (version of the operating system, compiler settings, input set used, specific computer configuration (clock rate, cache sizes and speed, memory size and speed, etc.))

单选题 1分

设置

可以通过一个大型的矩阵分解计算任务完成时间来比较AMD和Intel两种CPU的性能

A 可以比较

B 不能比较

C 无法确定

提交

Pitfall: Amdahl's Law

- Amdahl's law: performance enhancement possible with a given improvement is limited by the amount that the improved feature is used
- Pitfall: Improving an aspect of a computer and expecting a proportional improvement in overall performance

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

- Example: multiply accounts for 80s/100s
  - How much improvement in multiply performance to get 5× overall?

$$20 = \frac{80}{n} + 20$$

  - Can't be done!
- Corollary: make the common case fast

A Simple Example

Op	Freq	CPI_i	Freq x CPI_i			
ALU	50%	1	.5	.5	.5	.25
Load	20%	5	1.0	.4	1.0	1.0
Store	10%	3	.3	.3	.3	.3
Branch	20%	2	.4	.4	.2	.4
			Σ = 2.2	1.6	2.0	1.95

- How much faster would the machine be if a better data cache reduced the average load time to 2 cycles?
  CPU time new = 1.6 x IC x CC so 2.2/1.6 means 37.5% faster
- How does this compare with using branch prediction to shave a cycle off the branch time?
  CPU time new = 2.0 x IC x CC so 2.2/2.0 means 10% faster
- What if two ALU instructions could be executed at once?
  CPU time new = 1.95 x IC x CC so 2.2/1.95 means 12.8% faster

Pitfall: MIPS as a Performance Metric

- MIPS: Millions of Instructions Per Second
  - If used as a metric to compare computers:
    - Doesn't account for differences in instruction complexity
      - Different ISAs may lead to different instruction counts for same program
    - MIPS varies between programs on the same computer!
      - Computer cannot have a single MIPS rating

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

$$= \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

  - e.g. CPI varied by 13x for SPEC2006 on AMD Opteron X4, so MIPS does as well

SPEC Power Benchmark

- Power consumption of server at different workload levels
  - Performance: ssj_ops/sec (server side java ops/sec)
  - Power: Watts (Joules/sec)

$$\text{Overall ssj\_ops per Watt} = \left(\sum_{i=0}^{10} \text{ssj\_ops}_i\right) \Big/ \left(\sum_{i=0}^{10} \text{power}_i\right)$$

SPECpower_ssj2008 for X4

Target Load %	Performance (ssj_ops/sec)	Average Power (Watts)	Per/Power
100%	231,867	295	786
90%	211,282	286	739
80%	185,803	275	676
70%	163,427	265	617
60%	140,160	256	548
50%	118,324	246	481
40%	920,35	233	395
30%	70,500	222	318
20%	47,126	206	229
10%	23,066	180	128
0%	0	141	0
Overall sum	1,283,590	2,605	
∑ssj_ops/ ∑power		493	

Fallacy: Low Power at Idle

- Look back at X4 power benchmark
  - At 100% load: 295W
  - At 50% load: 246W (83%)
  - At 10% load: 180W (61%)
- Google data center
  - Mostly operates at 10% – 50% load
  - At 100% load less than 1% of the time
- Consider designing processors to make power proportional to load

主观题 10分

设置

请思考一下为什么现在基于虚拟化技术的云计算中心相对于传统数据中心的计算效能（能效比）更高，更加绿色节能?

正常使用主观题需2.0以上版本雨课堂

作答

Integrated Circuit Cost

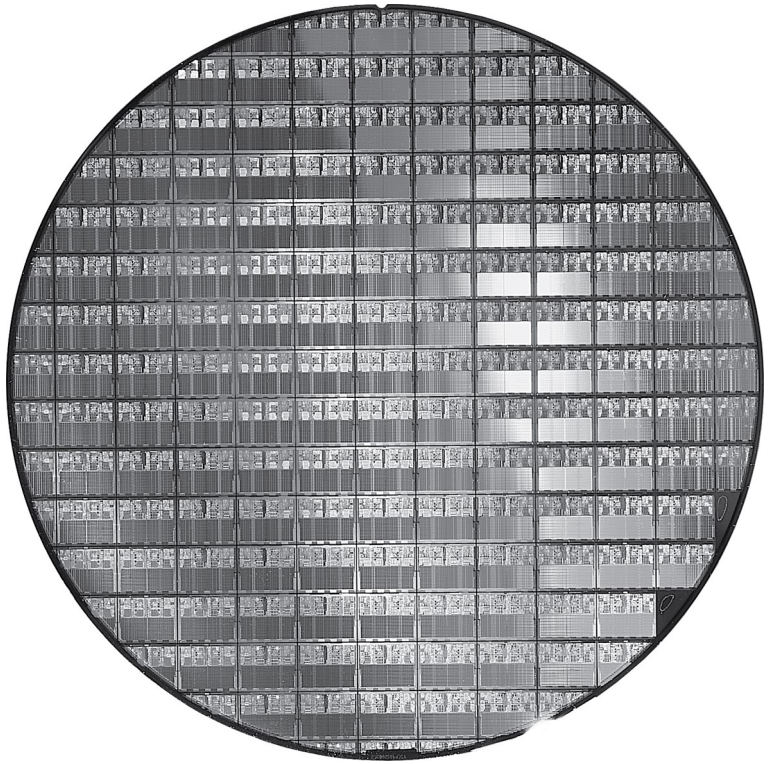
$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{Yield}}$$

$$\text{Dies per wafer} \approx \text{Wafer area}/\text{Die area}$$

$$\text{Yield} = \frac{1}{(1+(\text{Defects per area} \times \text{Die area}/2))^2}$$

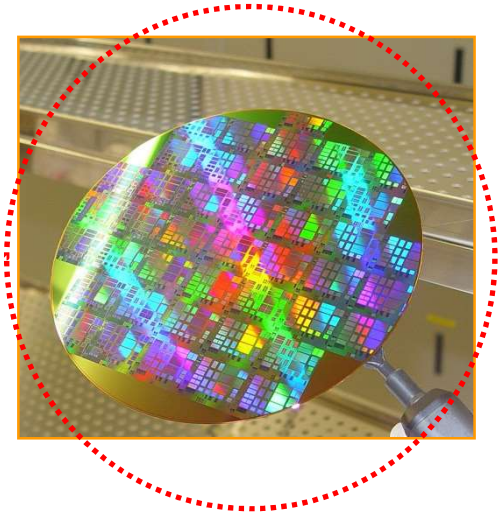
- Nonlinear relation to area and defect rate
  - Wafer cost and area are fixed
  - Defect rate determined by manufacturing process
  - Die area determined by architecture and circuit design

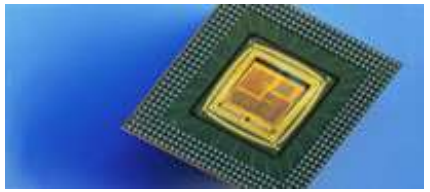
AMD Opteron X2 Wafer



- X2: 300mm wafer, 117 chips, 90nm technology
- X4: 45nm technology

芯片的产生与应用示意



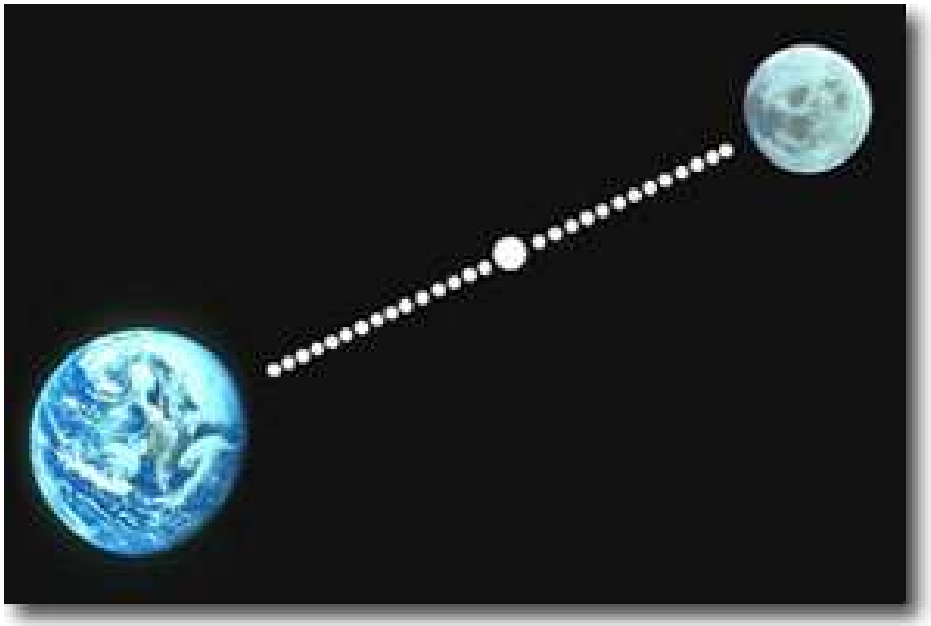


硅: 来源于沙子

Purity of Silicon of chips

- 硅体材料纯度达到 99.9999999%

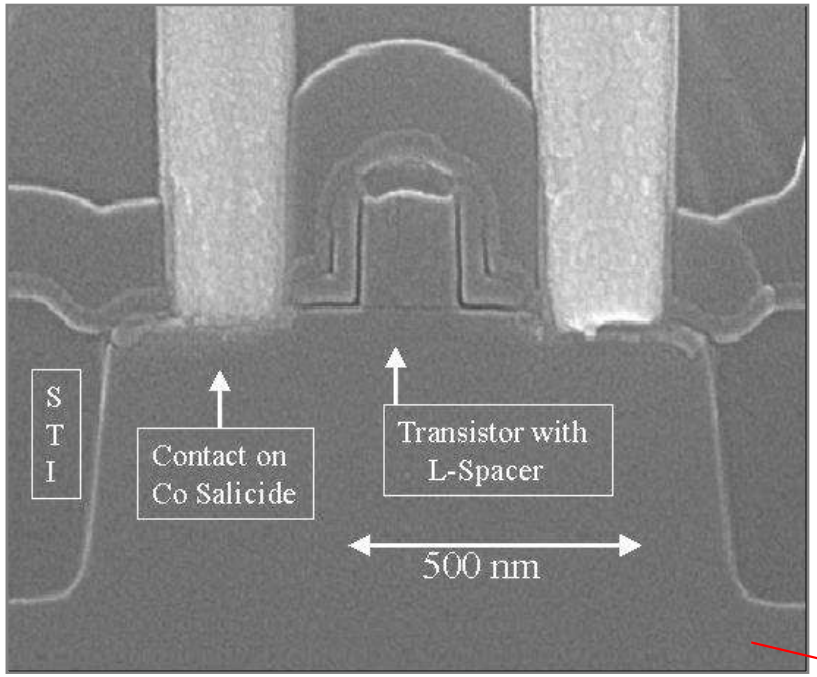
  Only allow one purity in 10B atoms —— equivalent to one defect in the Pin-Pon balls filled in between Earth and Moon.

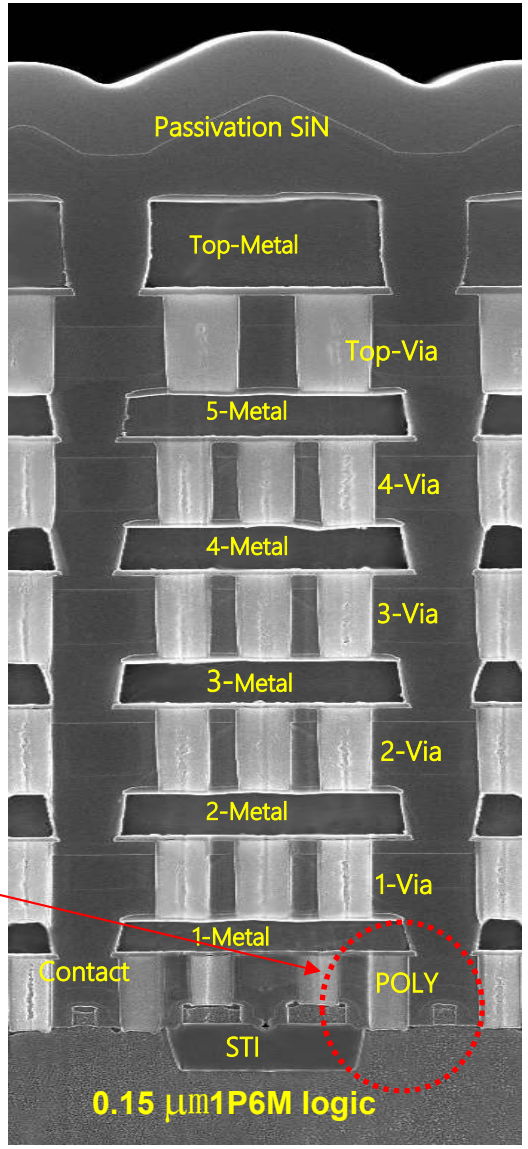


Purity of Manufacturing enviroments

- 洁净度是药品制造的1000倍。

分层结构



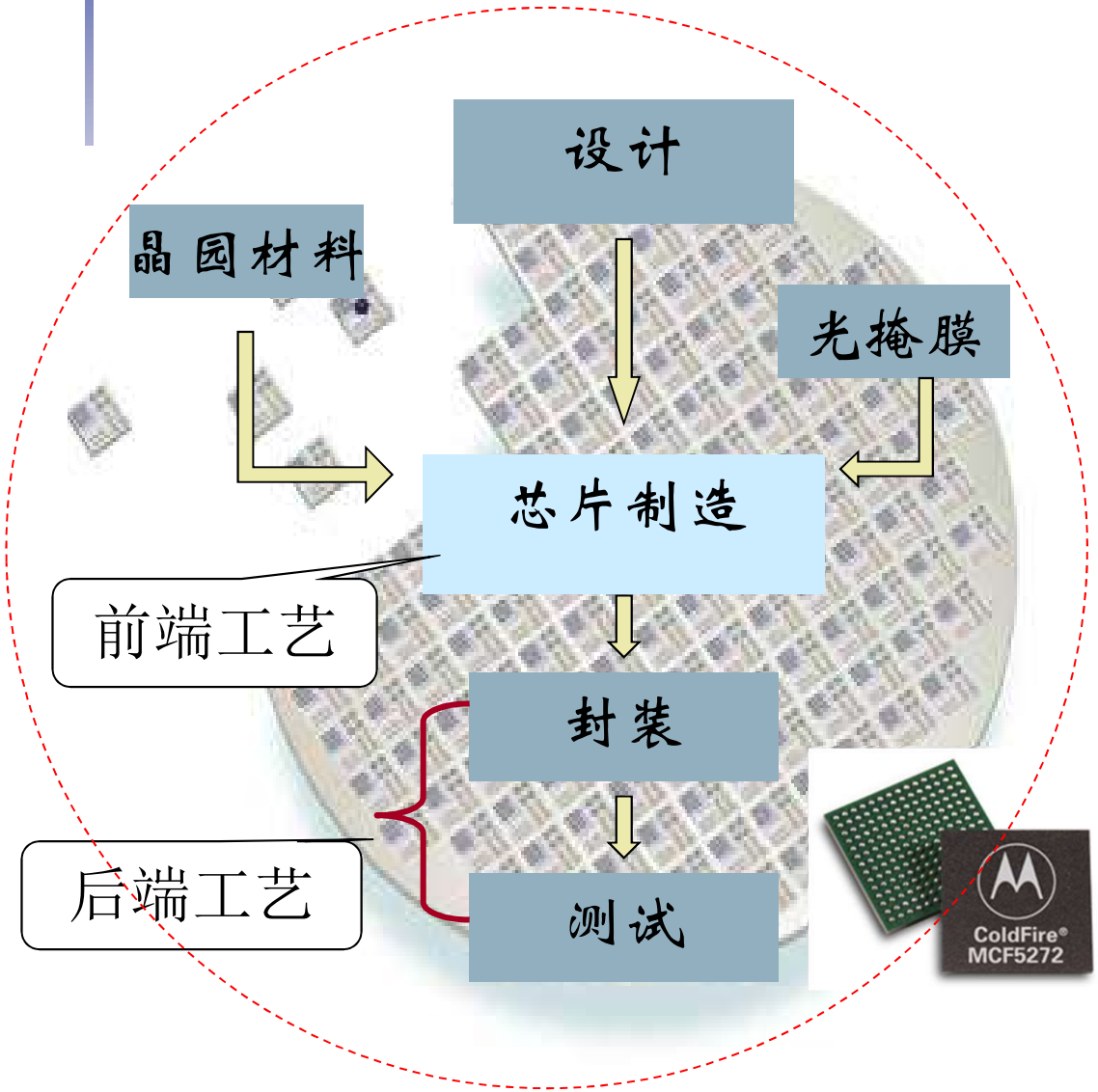


0.15 μm 逻辑电路分层结构:

共30层光掩膜

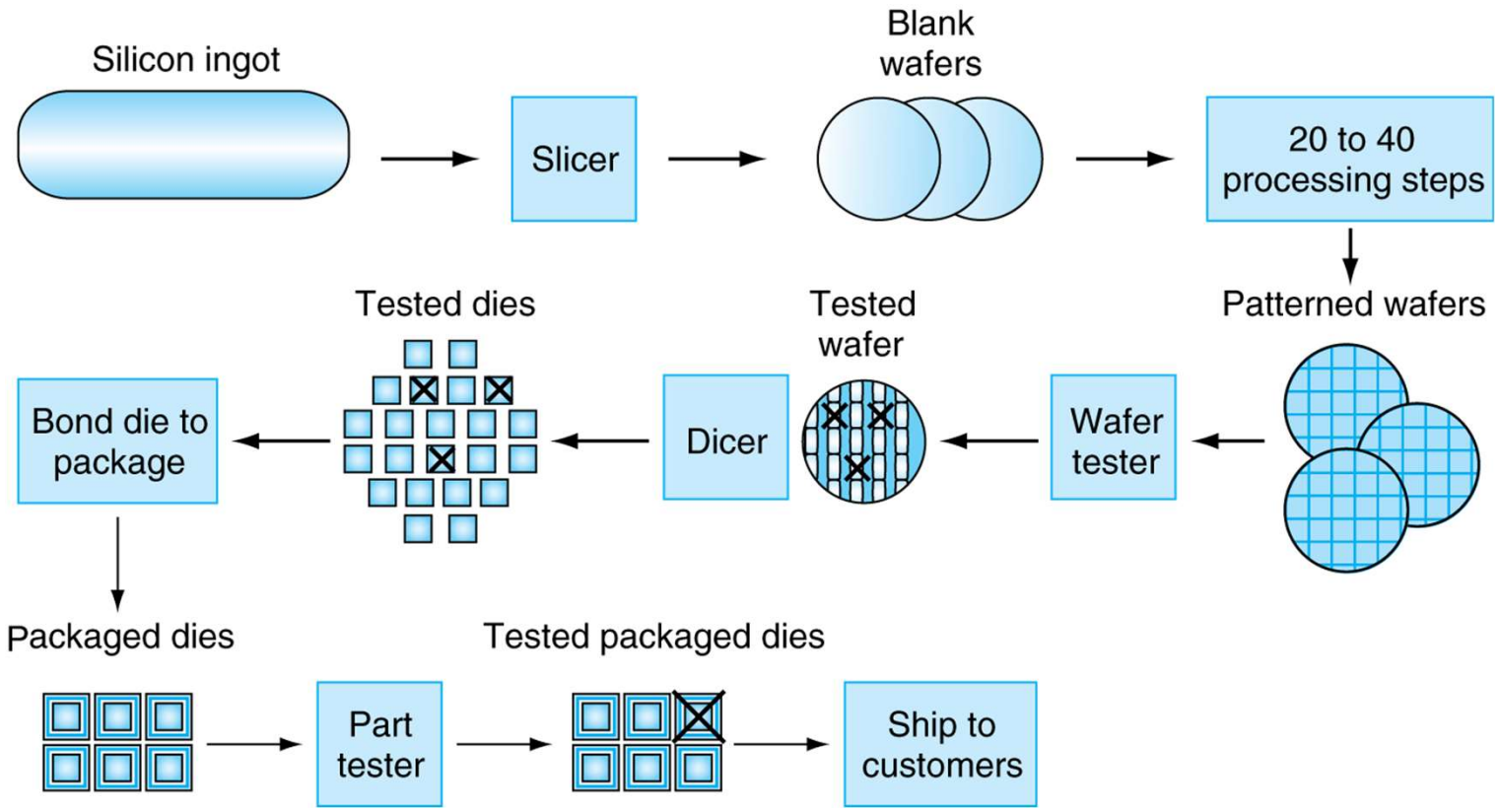
总共需600多步骤制造

IC Industrial Chain



系统制造商

Manufacturing ICs



- Yield: proportion of working dies per wafer

Concluding Remarks

- Cost/performance is improving
  - Due to underlying technology development
- Hierarchical layers of abstraction
  - In both hardware and software
- Instruction set architecture
  - The hardware/software interface
- Execution time: the best performance measure!
- Power is a limiting factor
  - Use parallelism to improve performance

主观题 10分

设置

课堂讨论：

请仔细阅读教材第一章内容，理解可以通过综合多种措施来改进计算机系统，使得系统性能可以提升200倍以上。

正常使用主观题需2.0以上版本雨课堂

作答

Assignment 1

- Homework assignment

  1.2 ~1.5 , 1.6 ~1.8 , 1.13

- To be submitted on Sakai

  File Format： Word or PDF

- Read

  Chapter 1 of P&H

  The First Draft Report on the EDVAC