Computer Organization and Design

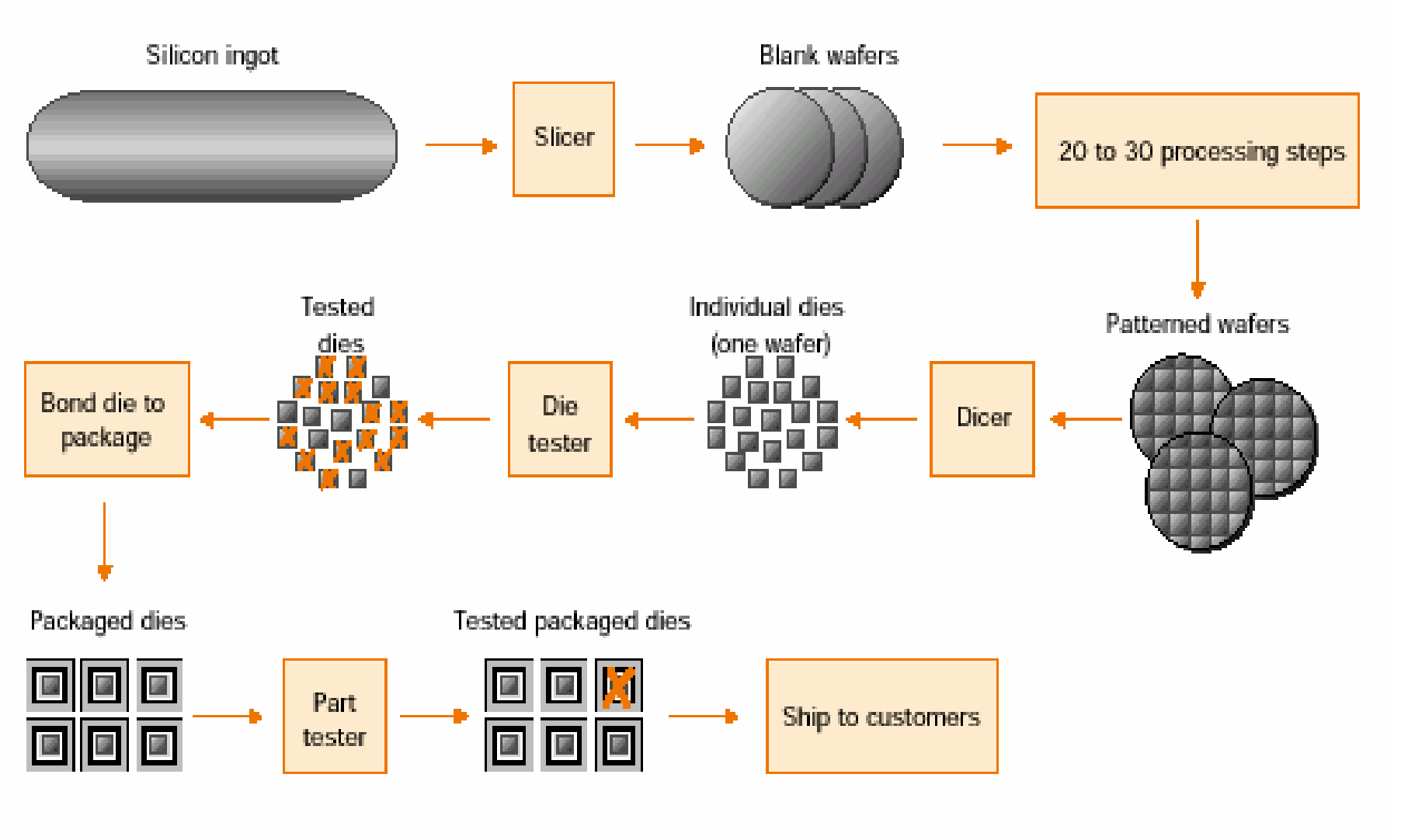
Ch4: Performance Evaluation

性能评价

主 要 内 容

- manufacturing cost
- Execution Time vs. Throughput
- CPU Time=cycle time x CPI x Instructions / program
- MIPS、MFLOPS
- Benchmark（基准程序）

回顾: Integrated Circuits Costs --- manufacturing process

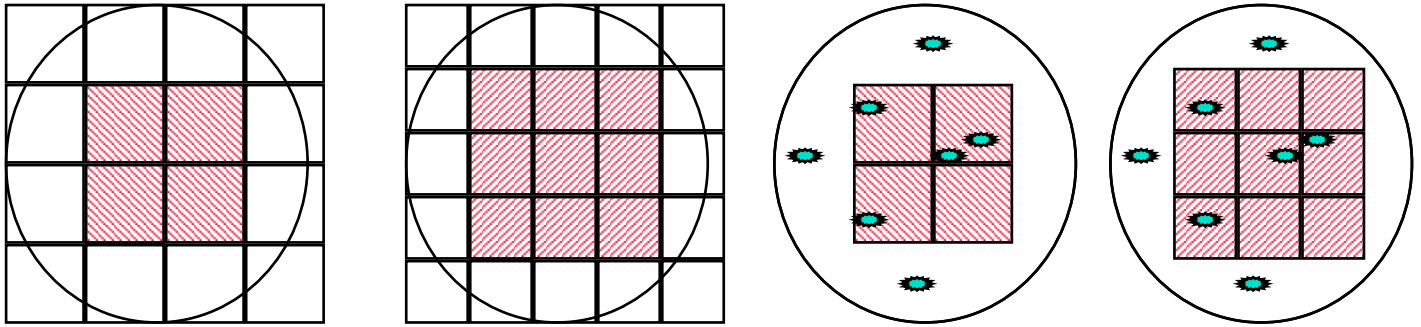


在考察性能前，先考察成本！

Integrated Circuits Costs --- formula

$$\text{Die cost} = \frac{Cost\_per\_wafter}{Die\_per\_wafer \times Yield}$$

$$\text{Dies per wafer} = \frac{wafer\_area}{Die\_area}$$



晶块合格率

$$\text{Die Yield} = \frac{1}{(1+(Defect\_per\_area \times Die\_area))^2}$$

由此看出：每个圆晶片上的晶块数、集成电路成本都与晶块面积有关！

Real World Examples

Chip	*Metal layers*	*Line width*	*Wafer cost*	*Defect /cm²*	*Area mm²*	*Dies/ wafer*	*Yield*	*Die Cost*
386DX	2	0.90	$900	1.0	43	360	71%	$4
486DX2	3	0.80	$1200	1.0	81	181	54%	$12
PowerPC 601	4	0.80	$1700	1.3	121	115	28%	$53
HP PA 7100	3	0.80	$1300	1.0	196	66	27%	$73
DEC Alpha	3	0.70	$1500	1.2	234	53	19%	$149
SuperSPARC	3	0.70	$1700	1.6	256	48	13%	$272
Pentium	3	0.80	$1500	1.5	296	40	9%	$417

From "Estimating IC Manufacturing Costs," by Linley Gwennap, *Microprocessor Report*, August 2, 1993, p. 15

Other Costs

$$\text{IC cost} = \frac{\text{Die cost} + \text{Testing cost} + \text{Packaging cost}}{\text{Final test yield}}$$

封装成本（Packaging cost）：取决于引脚数、散热性等

Chip	*Die cost*	*Package pins*	*Package type*	*Package cost*	*Test & Assembly*	*Total*
386DX	$4	132	QFP	$1	$4	$9
486DX2	$12	168	PGA	$11	$12	$35
PowerPC 601	$53	304	QFP	$3	$21	$77
HP PA 7100	$73	504	PGA	$35	$16	$124
DEC Alpha	$149	431	PGA	$30	$23	$202
SuperSPARC	$272	293	PGA	$20	$34	$326
Pentium	$417	273	PGA	$19	$37	$473

什么是计算机性能？

先考虑民航客机的“性能”：续航能力、速度、载客量、运输能力？

Airplane	DC to Paris	Range	Speed (m.p.h.)	Passengers	Throughput (p x m.p.h.)	Cost
Boeing 747	6.5 hours	4150	610	470	286,700	???
BAC/Sud Concorde	3 hours	4000	1350	132	178200	???
Douglas DC-8	7.3 hours	8720	544	146	79,424	???

- **Time of Concorde vs. Boeing 747?**
  - 从DC到巴黎，Concorde 比Boeing 747快3.5小时
  - 速度上Concorde 比Boeing 747快1350/610=2.2倍

  Concorde的性能更好！
- **Throughput of Boeing 747 vs. Concorde?**
  - 运载能力上Boeing 747 比Concorde大286,700/178200=1.6倍

  Boeing 747的性能更好！

考虑制造成本使性能评价更复杂

The bottom line: Performance (and cost)

- 不同的性能评价标准导致不同的结论
  - **Time to do the task (Execution Time)**
    - **execution time, response time, latency(等待时间或时延)**
  - **Tasks per day, hour, week, sec, ns. .. (Throughput)**
    - **throughput(吞吐率), bandwidth(带宽)**

- 基本的性能评价标准是**CPU**的执行时间

" X is n times faster than Y" means

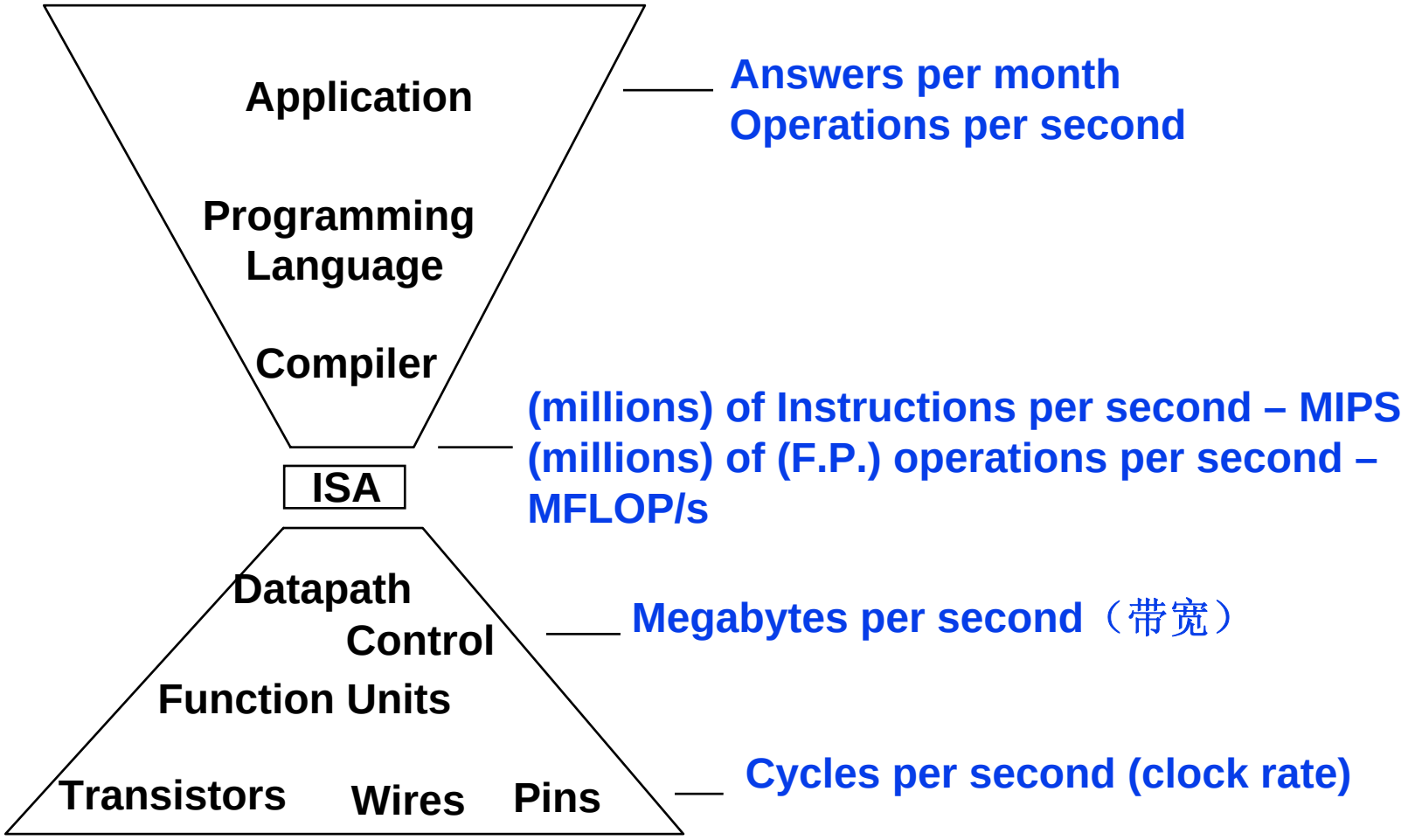
$$\frac{\text{ExTime(Y)}}{\text{ExTime(X)}} = \frac{\text{Performance(X)}}{\text{Performance(Y)}}$$

相对性能用执行时间的倒数来表示！

性能测量

- 不同场合用户关心的问题不同，例如：
  - 要求吞吐率高的场合：如，多媒体应用（音/视频的播放要流畅！）
  - 要求响应时间短的场合：如，事务处理系统（存/取款的速度要快！）
  - 要求吞吐率高且响应时间短的场合：**ATM**、文件服务器、**Web**服务器等
- 比较计算机的性能时，用执行时间来衡量
  - 完成同样工作量所需时间最短的那台计算机就是性能最好的
  - 计算机处理器时间往往被多个程序共享使用，因此，用户感觉到的程序执行时间并不是程序真正的执行时间
  - 通常把用户感觉到的响应时间分成：
    - **CPU**时间，指**CPU**真正花在程序执行上的时间。又包括两部分：
      - ✓ 用户**CPU**时间：用来运行用户代码的时间
      - ✓ 系统**CPU**时间：为了执行用户程序而需要运行操作系统程序的时间
    - 其他时间，指等待**I/O**操作完成的时间或**CPU**花在其他程序的时间
  - 系统性能和**CPU**性能不等价，有一定的区别
    - 系统性能**(System performance)**：
      系统响应时间，与**CPU**外的其他部分也都有关系
    - **CPU**性能**(CPU performance)**：用户**CPU**时间
  - 本章主要讨论**CPU**性能

在不同层次上的计算机性能度量



Relating Processor Metrics

- CPU execution time = CPU clock cycles/pgm ÷ clock rate
- or CPU execution time = CPU clock cycles/pgm X clock cycle time
- or CPU execution time = CPI X instrs/pgm X clock cycle time
- CPU clock cycles/pgm = Instrs/pgm X avg. clock cycles per instr.
- CPI = CPU clock cycles/pgm ÷ Instructions/pgm
- CPI 用来衡量以下各方面的综合结果
  - Instruction Set Architecture
  - Implementation of that architecture
  - program

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. count	CPI	clock rate
Program			
Compiler			
Instr. Set Arch.			
Organization			
Technology			

思考：三个因素与哪些方面有关？

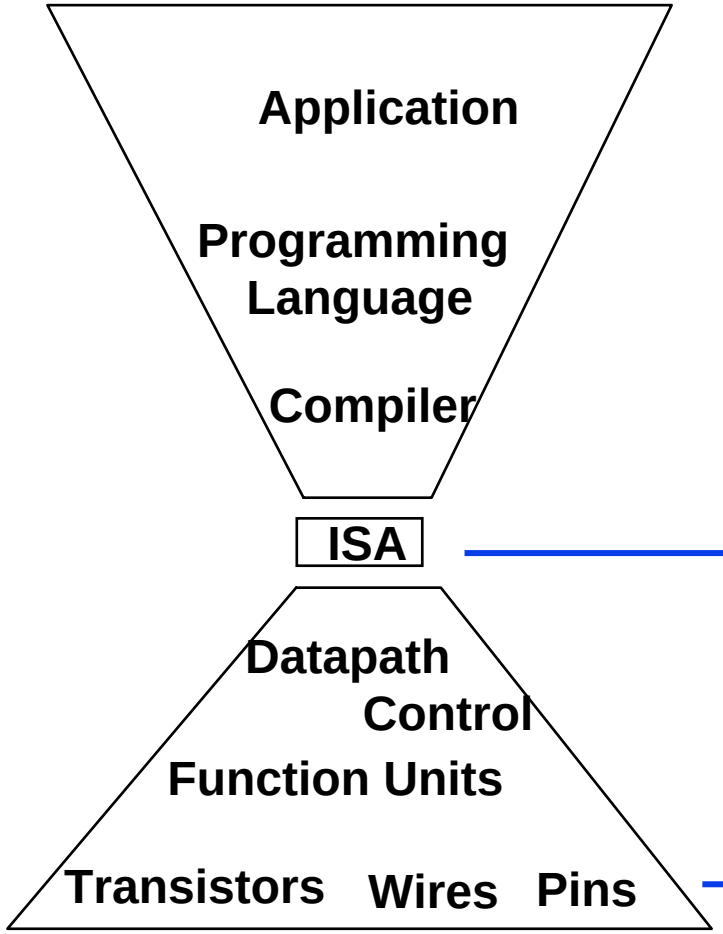
Aspects of CPU Performance

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

	instr. count	CPI	clock rate
Program	X	X	
Compiler	X	(x)	
Instr. Set Arch.	X	X	
Organization		X	X
Technology			X

Organizational Trade-offs

因此，必须在各方面进行权衡！



3 factors:How are they related?

- ✓ 时钟数的减少可能会增加时钟周期的长度
- ✓ 增加时钟频率会增加指令的条数
- ✓ 改变IS以减少指令的条数会使时钟周期变长
- ✓ 即使是同一台机器的同一个问题，最少指令条数的程序不一定执行最快

Instruction Count (Mix)

CPI

Cycle Time

3 factors:
Where are they?

如何计算CPI?

CPI = (CPU Time × Clock Rate) / Instruction Count
= Clock Cycles / Instruction Count

CPI **"Average cycles per instruction"**

CPU time = $ClockCycle\ \ Time \times \sum_{i=1}^{n} CPI_i \times C_i$

Ci **"instruction count"**

CPI = $\sum_{i=1}^{n} CPI_i \times F_i$ **where** $F_i = \frac{C_i}{Instruction\_Count}$

Fi **"instruction frequency"**

Example1

Our favorite program runs in 10 sec on machine A, which has a 400MHz clock. We are trying to design a machine B with faster clock rate so as to reduce the execution time to 6 sec.

The increase of clock rate will affect the rest of the CPU design, causing B to require 1.2 times as many clock cycles as machine A for this program. What clock rate should be?

Answer:

CPU time A = CPU clock cycle A / clock rate A

==> CPU clock cycle A = 10 sec x 400 x 10^6

Clock rate B = CPU clock cycle B / CPU time B

= 1.2*400*10^6 / 6 = 800 MHz

B's clock is 2 times as fast as A's clock, but B is not 2 times as fast as A.

Example2

Instruction type and Instruction frequencies in the execution of a program:

Op	Freq	Cycles
ALU	43%	1
Load	21%	2
Store	12%	2
Branch	24%	2

Question: What is the average CPI of the machine?

CPI = 1×43% + 2×21% + 2×12% +2×24% = 1.57

Example3

Suppose we have two implementations of the same instruction set.
Machine A has a clock cycle time of
10 ns and an average CPI of 2.0 for some program.

Machine B has a clock cycle time of 20 ns and an
average CPI of 1.2 for the same program.

Which is faster? And by how much?

相同**IS**对于同一个程序，其指令序列是一样的，当然条数相同！

Let n denote the number of instructions of the program
CPU time A = n * 2.0* 10 = 20n（ns）
CPU time B = n * 1.2 *20 = 24n（ns）

Machine A is 1.2 faster than B .

在此，又看到三个因素之间的相互影响。

Example 4

ISA has 3 kinds of instructions:

Instruction class	CPI for this instruction class
A	1
B	2
C	3

One program has 2 code sequences:

Code Sequence	Instruction counts for instruction class		
	A	B	C
1	2	1	2
2	4	1	1

Which code sequence has more instructions?
Which will be faster?
What is the CPI for each sequence?

S.1 has 5 instructions; S.2 has 6.
S.1 needs 2x1+1x2+2x3=10 cycles; S.2 needs 4x1+1x2+1x3=9 cycles.
S.1 has CPI=10/5=2; S.2 has CPI=9/6=1.5

选择性能评价程序（Benchmarks）

- 用基准程序来评测计算机的性能
  - 基准测试程序是专门用来进行性能评价的一组程序
  - 不同用户使用的计算机用不同的基准程序
  - 基准程序通过运行实际负载来反映计算机的性能
  - 最好的基准程序是用户实际使用的程序或典型的简单程序

- 基准程序的缺陷
  - 现象：基准程序的性能与某段短代码密切相关时，会被利用以得到不当的性能评测结果
  - 手段：硬件系统设计人员或编译器开发者针对这些代码片段进行特殊的优化，使得执行这段代码的速度非常快
    - 例1：**Intel Pentium**处理器运行**SPECint**时用了公司内部使用的特殊编译器，使其性能极高
    - 例**2**：矩阵乘法程序**SPECmatrix300**有**99%**的时间运行在一行语句上，有些厂商用特殊编译器优化该语句，使性能达**VAX11/780**的**729.8**倍！

用于性能估计的程序

- **(Toy) Benchmarks**（短小基准程序）
  - 程序短小容易编译，便于仿真或手工编译，因而可用于对新开发的机器进行性能评测.（因为**No compiler for novel machines**）。
  - 大小：**10-100 line**
  - 例：**sieve, puzzle, quicksort**
  - 缺陷：不是实际使用的程序，只用于新开发的计算机。
- **Synthetic Benchmarks**（综合基准程序）
  - 目的：试图用一个基准程序去涵盖一系列基准程序的特征。
  - 做法：使各种语句的执行频度与一系列基准程序中的频度一致。
  - 例： **Whetstone(Algol60→Fortran), Dhrystone(Ada→C)**
  - 缺陷：不是实际使用的程序，可作特殊优化能使评测结果优，但实际不是如此。
- **Kernels**（核心程序段）
  - 实际程序的时间关键片断
  - 通常用在科学计算领域测试某个特殊功能的性能
  - 例： **Livermore loops(21 loops), Linpack(linear algebra)**
- **Real programs**（频繁使用的实际程序）
  - **e.g., gcc, spice**

Successful Benchmark: SPEC

- 1988年，5家公司（ Sun, MIPS, HP, Apollo, DEC ）联合提出了Systems Performance Evaluation Committee (SPEC)
- SPEC给出了一组标准的测试程序、标准输入和测试报告。它们是一些实际的程序，包括 OS calls、 I/O等。
- 版本 89：10 programs = 4 for integer + 6 for FP, 用每个程序的执行时间求出一个综合性能指标
- 版本92：SPECInt92 (6 integer programs) and SPECfp92 (14 floating point programs)
  - 整数和浮点数单独提供衡量指标：SPECInt92和SPECfp92
  - 增加 SPECbase: 禁止使用任何与程序有关的编译优化开关
- 版本95： 8 int + 10fp
- 较新版本： include SPEC HPC96, SPEC JVM98, SPEC WEB99, SPEC OMP2001. SPEC CPU2000，See http://www.spec.org for more details
  - “benchmarks useful for 3 years”
  - Base machine is changed from VAX-11/780 to Sun SPARC 10/40

如何给出综合评价结果?

问题：如果用一组基准程序在不同的机器上测出了运行时间，那么如何综合评价机器的性能呢？

先看一个例子：

Program 1: 1 sec on machine A, 10 sec on machine B

Program 2: 1000 sec on A，100 sec on B

What are your conclusions?

- **A is 10 times faster than B for program1.**
- **B is 10 times faster than A for Program2.**

这个结论无法比较A和B的好坏

必须用一个综合的值来表示！

Total execution time是一个综合度量值，可以据此得出结论：

B is 1001/110=9.1 times faster than A

实际上，必须考虑每个程序在作业中的使用频度，即加权平均

综合性能评价的方法

- 可以用以下两种平均值来评价：
  - **Arithmetic mean(算术平均)：求和后除n**
  - **Geometric mean(几何平均)：求积后开根号n**
- 根据算术平均执行时间能得到总平均执行时间
- 根据几何平均执行时间不能得到程序总的执行时间
- 执行时间的规格化（测试机器相对于参考机器的性能）：
  - time on reference machine ÷ time on measured machine
- 平均规格化执行时间不能用算术平均来计算，而应该用几何平均
  - program A going from 2 seconds to 1 second as important as program B going from 2000 seconds to 1000 seconds.

（算术平均值不能反映这一点！）

综上所述，算术平均和几何平均各有长处，可灵活使用！

Impact of Means on SPECmark89 for IBM 550

	Ratio to VAX:		Time:		Weighted Time:	
Program	*Before*	*After*	*Before*	*After*	*Before*	*After*
gcc	30	29	49	51	8.91	9.22
espresso	35	34	65	67	7.64	7.86
spice	47	47	510	510	5.69	5.69
doduc	46	49	41	38	5.81	5.45
nasa7	78	144	258	140	3.43	1.86
li	34	34	183	183	7.86	7.86
eqntott	40	40	28	28	6.68	6.68
matrix300	78	730	58	6	3.43	0.37
fpppp	90	87	34	35	2.97	3.07
tomcatv	133	138	20	19	2.01	1.94
Mean	54	72	124	108	54.42	49.9
	Geometric		*Arithmetic*		*Weighted Arith.*	
	Ratio	1.33	Ratio	1.16	Ratio	1.09

注：“After”表示加了特殊编译开关后的结果。

好的评价标准应该对特殊处理不敏感！

该表反映了不同的均值计算法得到的不同结论。

Marketing Metrics （产品宣称指标）

MIPS = Instruction Count / Time * 10^6

= Clock Rate / CPI * 10^6

Million Instructions Per Seconds

- 不同机器的指令集不同
- 程序由不同的指令混合而成
- 指令使用的频度动态变化
- **Peak MIPS:** （不实用）

用**MIPS**数表示性能有没有局限**?**

所以**MIPS**数不能说明性能的好坏 **(see the next example)**

MFLOP/S = FP Operations / Time * 10^6

Million Floating-point Operations Per Second

- 与机器相关性大
- 不是程序中花时间的部分

用**MFLOPS**数表示性能也有局限！

Example: MIPS数不可靠！

Assume we build an optimizing compiler for the load/store machine. The compiler discards 50% of the ALU instructions.

1) What is the CPI ?

2) Assuming a 20 ns clock cycle time (50 MHz clock rate). What is the MIPS rating for optimized code versus unoptimized code? Does the MIPS rating agree with the rating of execution time?

Op	Freq	Cycle	Optimizing compiler	New Freq
ALU	43%	1	21.5/ (21.5+21+12+24)=27%	27%
Load	21%	2	21 / (21.5+21+12+24)=27%	27%
Store	12%	2	12 / (21.5+21+12+24)=15%	15%
Branch	24%	2	24 / (21.5+21+12+24)= 31%	31%
CPI		1.57		1.73
MIPS		31.8	50/1.57=31.8MIPS / 50/1.73=28.9MIPS	28.9

结果：因为优化后减少了**ALU**指令（其他指令数没变），所以程序执行时间一定减少了，但优化后的**MIPS**数反而降低了。

性能评价的重要定律：Amdahl定律

举例：某个程序在某台计算机上运行所需的时间是**100s**，其中**80s**用来执行乘法操作。那么，要使该程序的速度提高到原来的**5**倍，那么，乘法部件的速度应该是原来的多少倍？

乘法部件的速度是不是应该为原来的**5**倍呢？

$$\text{改进后的执行时间} = \frac{\text{改进部分的执行时间}}{\text{改进部分的改进倍数}} + \text{未改进部分执行时间}$$

$$20s = \frac{80s}{n} + (100s - 80s) \qquad 0 = \frac{80s}{n}$$

说明：当**80%**都是乘法运算的话，无论怎样对乘法部件改进，整体性能都不可能提高到原来的**5**倍。

陷阱：在对机器的某一部分进行改进后，整个系统的性能应该得到同样幅度的提高。

$$\textbf{Amdahl}\text{定律：整体改进倍数} = \frac{1}{\text{改进部分份额} / \text{改进部分的改进倍数} + \text{未改进部分份额}}$$

- **Amdahl**定律说明：整体性能在多大程度上得以提高，与改进部分在总的执行时间中所占的份额密切相关。

Example: 改进部分份额和改进倍数的关系

Suppose an enhancement runs 10 times faster than the original machine, but is only usable 40% of the time.

Question: what is the overall speedup?

Answer: 改进部分时间份额 = 0.4
改进部分的改进倍数 = 10
整体改进倍数 = 1/(0.6+0.4/10) = 1.56

若改进部分时间份额为0.6呢，结果如何？

整体改进倍数 = 1/(0.4+0.6/10) = 2.17

若改进部分时间份额为0.9呢，结果如何？

整体改进倍数 = 1/(0.1+0.9/10) = 5.26

是不是与想象的结果相差较大！

本章小结

- 性能的定义：一般用程序的响应时间或系统的吞吐率表示机器或系统整体性能。
- CPU性能的测量：
  - 一般把程序的响应时间划分成CPU时间和等待时间，CPU时间又分成用户CPU时间和系统CPU时间。
  - 因为操作系统对自己所花费的时间进行测量时，不十分准确，所以，对CPU性能的测量一般通过测量程序运行的用户CPU时间来进行。
- 各种性能指标之间的关系：
  - CPU执行时间=CPU时钟周期数 x 时钟周期
  - 时钟周期和时钟频率互为倒数
  - CPU时钟周期数 = 程序指令数 x 每条指令的平均时钟周期数CPI
- 性能评价程序的选择：
  - 采用一组基准测试程序来对机器的性能进行综合（算术（加权）平均/几何平均）评测。
  - 有些制造商会针对评测程序中频繁出现的语句采用专门的编译器，使评测程序的运行效率大幅提高。因此有时基准评测程序也不能说明问题。
- MIPS数不能说明问题，不具有可比性！
- 对于某种特定的指令集体系结构，提高计算机性能的主要途径有：
  - 提高时钟频率（第六章 流水线）
  - 优化处理器中数据通路的结构以降低CPI（第五章 单周期/多周期处理器）
  - 采用编译优化措施来减少指令条数或降低指令复杂度（第二章 ）

本章作业

- **4.1**
- **4.2**
- **4.3**
- **4.6**
- **4.7**
- **4.8**
- **4.11**
- **4.12**
- **4.14**
- **4.15**

下星期二（3月24号）交作业！