第1章 计算机设计基础

1.1 引言
1.2 计算机的分类
1.3 计算机系统结构定义和计算机的设计任务
1.4 实现技术的趋势
1.5 集成电路功耗的趋势
1.6 成本的趋势
1.7 可靠性
1.8 测量、报告和总结计算机性能
1.9 计算机设计的量化原则
1.10 综合：性能和性价比

1.9 计算机设计的量化原则

- 利用并行性（parallelism）
- 局部性原理（Principle of Locality）
- 注重经常性事件（the common case）
- Amdahl's 定律
- CPU 性能公式

利用并行性

- 改善计算机性能最重要的方法
- 并行性的层次
  - 系统级: 多线程，多个处理器
  - 指令级:
    - 流水线、超标量（多发射）、Out Of Order等
  - 操作级:
    - 并行加法器
    - 组相联cache
    - 功能部件流水线

局部性原理

- 程序特性：趋向于重用最近用过的数据和指令
- 经验法则：
  - 一个程序90%的执行时间仅仅执行其10%的代码。
- 时间局部性（Temporal locality）
  - 最近访问过的项很可能近期将被访问。
- 空间局部性（Spatial locality）
  - 地址相近单元的内容趋向于在一定时间内被相近访问。

注重常用事件

- 计算机设计最重要和普遍原则
  - 功耗、资源分配、性能、可靠性
  - 经验法则：simple is fast.
  - 简化常用事件，速度能够更快

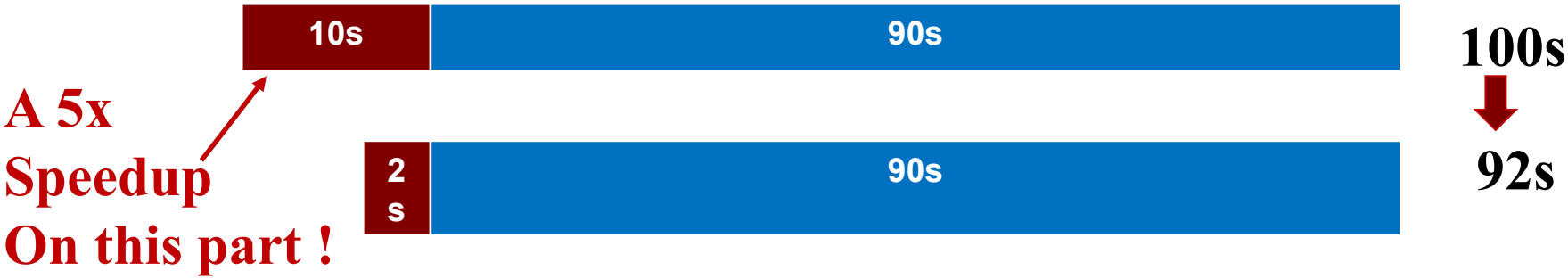
例如：CPU中两个数相加，结果可能产生溢出，溢出情况较少，不溢出才是常见情况。因此，可以通过简化不溢出相加的操作来提高机器的性能。

例如：处理器中的取指和译码单元要比乘法单元使用得更加频繁，因此，应注重这两个单元的性能设计。

- 量化这个原则的基本定律：*Amdahl's Law*

Amdahl's 定律

- 采用**更快的执行方式**后所获得的**系统性能提高**，与这种**执行方式**的**使用频率**或**占总执行时间的比例**有关。
- Example



$$\text{改进后的执行时间} = \frac{\text{改进部分的执行时间}}{\text{改进加速比}} + \text{不可改进部分的执行时间}$$

- 增加时钟频率**不会影响存储器访问时间**
- 使用浮点处理部件**不会加速整数ALU操作**

Amdahl定律定义了一台计算机系统采用某种改进措施所取得的加速比：

$$\text{加速比} = \frac{\text{采用改进措施后计算机的性能}}{\text{没有采用改进措施试计算机的性能}} = \frac{\text{没有采用改进措施时计算机的执行时间}}{\text{采用改进措施试后计算机的执行时间}}$$

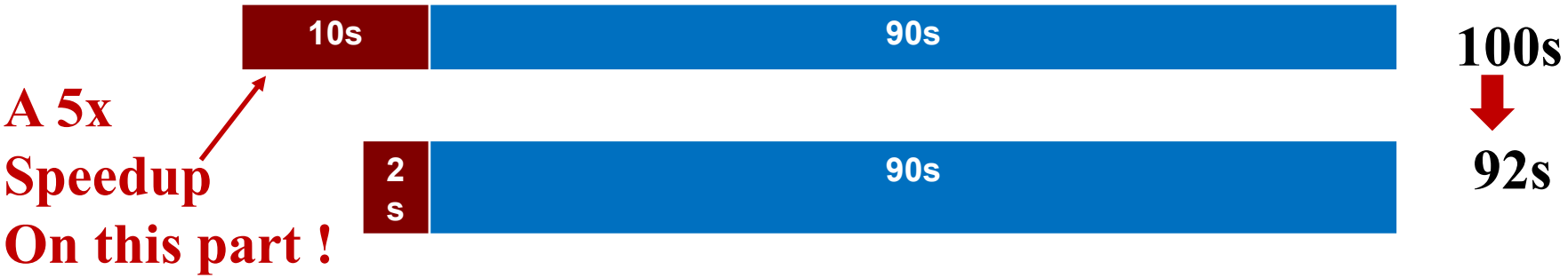
加速比反映了使用改进措施后完成一个任务比不使用改进措施完成同一任务加快的比率。

Amdahl定律中，加速比与两个因素有关：

a. 改进比例 Fe

$$Fe = \frac{\text{可改进部分的执行时间}}{\text{改进前整个任务的执行时间}}$$

下图中：Fe就是10 / 100=1/10。



b. 改进加速比 Se

$$Se = \frac{\text{改进前可改进部分的执行时间}}{\text{改进后可改进部分的执行时间}}$$

上图中，Se就是10 / 2=5

Amdahl定律公式：

$$\text{改进后的执行时间} = \text{不可改进部分的执行时间} + \frac{\text{改进部分的执行时间}}{\text{改进加速比}}$$

设改进后执行时间为T_{new}，

改进前的执行时间为T_{old}，则上式可以写为：

$$T_{new} = T_{old}(1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced} * T_{old}}{Speedup_{enhanced}}$$

简写为：

$$T_n = T_o(1 - F_e + F_e/S_e)$$

上式中 $1 - F_e$ 表示不可改进部分。

根据： $T_n = T_o(1 - F_e + F_e/S_e)$

上式中$1 - F_e$**表示不可改进部分。**

改进后整个系统的加速比S_n**为:**

显然当F_e**为0，即没有可改进部分时，** S_n**为1。**

当$S_e \to \infty$ **时，则**$S_n = \frac{1}{1-F_e}$ **，因此，可获取性能改善极限值受**F_e **值的约束。**

$$S_n = \frac{T_o}{T_n} = \frac{1}{1 - F_e + F_e/S_e}$$

提高改进比例F_e或改进加速比S_e，都可以提高S_n ， 但是F_e对 S_n的影响更大。

Amdahl定律举例1

浮点功能单元执行性能提高2倍；但是仅有10%的浮点指令。计算总体性能加速比

$$Time_{new} = Time_{old} \times (1 - 0.1 + 0.1/2) = 0.95 \times Time_{old}$$

$$Speedup_{overall} = 1 \div 0.95 = 1.053$$

【例2】 假定新的处理器采用了改进措施，新处理器处理的计算速度是原来处理器的10倍，同时假定老处理器有40%的时间用于计算，另外60%的时间用于I/O操作。那么改进性能后总的加速比是多少？

【例2】 假定新的处理器采用了改进措施，新处理器处理的计算速度是原来处理器的10倍，同时假定老处理器有40%的时间用于计算，另外60%的时间用于I/O操作。那么改进性能后总的加速比是多少？

解：由题意可知：$F_e = 40\% = 0.4$ ， $S_e = 10$

$$S_n = \frac{1}{1 - F_e + F_e/S_e} = \frac{1}{1 - 0.4 + 0.4/10}$$
$$= 1.5625$$

【例3】 试分析采用哪种设计方案实现求浮点数平方根FPSRQ对系统性能提高更大。假定FPSRQ操作占原来整个测试程序执行时间的20%。

一种设计方案是增加专门的FPSRQ硬件，可以将FPSRQ操作的速度加快到10倍；

另一种设计方案是提高所有FP运算指令的执行速度，使得FP指令的执行速度加快为原来的1.6倍，设FP运算指令在原来总执行时间中占50%。

试比较这两种设计方案。

解：根据 $S_n = \dfrac{1}{1 - F_e + F_e/S_e}$

对这两种设计方案的加速比分别进行计算。

增加专门FPSRQ硬件方案：$F_e = 20\% = 0.2$，$S_e = 10$

$$S_{SFPSRQ} = \frac{1}{1-0.2+0.2/10} = \frac{1}{0.82} \approx 1.2195$$

提高所有FP运算指令速度方案：$F_e = 50\% = 0.5$，$S_e = 1.6$

$$S_{FP} = \frac{1}{1 - 0.5 + 0.5/1.6} = \frac{1}{0.8125} \approx 1.2308$$

根据结果判断，提高所有FP运算指令速度的方案要好一些，这是由于该测试程序中浮点操作所占比重较大。

上例需要知道FPSQR硬件方案和改进FP操作的时间，直接测量这些时间是比较困难。因此，也使用后面介绍的CPU性能公式来选择设计方案。

例题4　设一个磁盘子系统有如下组件和MTTF:

- 10个磁盘，每一个的MTTF是1 000 000（1百万）小时
- 1个ATA控制器，500 000（50万）小时的MTTF
- 1个电源，200 000（20万）小时的MTTF
- 1个风扇，200 000小时的MTTF
- 1条ATA电缆，1 000 000小时的MTTF

假设采用双电源，计算总故障率的改进比率。

例题4　设一个磁盘子系统有如下组件和MTTF:

- 10个磁盘，每一个的MTTF是1 000 000（1百万）小时
- 1个ATA控制器，500 000（50万）小时的MTTF
- 1个电源，200 000（20万）小时的MTTF
- 1个风扇，200 000小时的MTTF
- 1条ATA电缆，1 000 000小时的MTTF

假设采用双电源，计算总故障率的改进比率。

解：单电源

$$\text{系统故障率} = 10 \times \frac{1}{1\,000\,000} + \frac{1}{500\,000} + \frac{1}{200\,000} + \frac{1}{200\,000} + \frac{1}{1\,000\,000}$$

$$= \frac{23}{1\,000\,000}$$

Therefore, the fraction of the failure rate that could be improved is 5 per million hours out of 23 for the whole system, or 0.2174.

前面的例题计算了如果采用双电源，则电源系统的可靠性会提高到4167倍。因此总的改进比率为

$$Improvement_{power\ supply\ pair} = \frac{1}{(1 - 0.2174) + \frac{0.2174}{4167}} = \frac{1}{0.7827} = 1.2777$$

CPU 性能公式

- 处理器性能的“铁律”：
  - 要直接测量使用新改进措施的改进时间是困难的。
- CPU 性能公式

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

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

$$= \frac{Instruction\ Count\ * Cycles\ Per\ Instruction}{Clock\ rate}$$

计算CPU 时间

$$\text{CPU time} = \frac{Instruction\ Count * Cycles\ Per\ Instruction}{Clock\ rate}$$

Architecture --> Implementation --> Realization

Compiler Designer Processor Designer Chip Designer

Component of performance	Units of measure
CPU execution time for a program	Seconds for the program
Instruction count	Instructions executed for the program
Clock cycles per instructions (CPI)	Average number of clock cycles/instruction
Clock cycle time	Seconds per clock cycle

相关因素

- CPU 性能取决于3个特征：
  - clock cycle time (or clock rate) (CCT)
  - cycles per instruction (CPI)
  - instruction count (IC)

	Inst Count	CPI	Clock Rate
Program	√		
Compiler	√	√	
Instraction Set		√	√
Organization		√	√
Technology			√

如：cache选择直接映像还是2-路组相联，其时钟周期时间是不同的。

如程序中有n类指令时，处理器设计用下式计算总的CPU时钟周期数：

$$\text{CPU的时钟周期数} = \sum_{i=1}^{n} I_i * CPI_i$$

其中I_i表示i类指令在程序中执行的次数，CPI_i表示i类指令所需的平均时钟周期数。以下式子表示CPU时间：

$$\text{CPU时间}=\left(\sum_{i=1}^{n} I_i * CPI_i\right) * \text{时钟周期}$$

程序总CPI也可以表示为：

$$CPI = \frac{\left(\sum_{i=1}^{n} I_i * CPI_i\right)}{IC} = \sum_{i=1}^{n}\left(\frac{I_i}{IC} * CPI_i\right)$$

其中，IC表示程序总执行指令数，I_i/IC表示执行i 类指令数在程序总执行指令数中所占的比例（即*i*类指令执行频度，可统计获得）。

*CPI*需要通过测量获得，它与流水线效率、Cache的命中率以及其他存储器效率等有关。

下面是对前面例题的两种设计方案，改用指令的执行频度和指令CPI来分析，在实际情况中可以通过仿真或使用硬件仪器来测量相应的指标。下面通过例子来说明上述CPU性能公式。

假设针对一个测试程序有如下的测量值：

FP指令（包括FPSRQ指令）的执行频度 = 25%

FP指令的平均CPI = 4.0

其他指令的平均CPI = 1.3333

FPSRQ指令的执行频度 = 2%

FPSRQ指令的CPI = 20

假定有两种备选的设计方案:

一种是将FPSRQ的CPI减至2；（减少至20/2=10倍）

另一种是将所有FP的CPI减至2.5。（减少至4/2.5=1.6倍）

下面用CPU性能公式比较这两种方案。

解：由题意可知，方案中只有CPI发生了变化，指令执行频度保持不变。首先计算没有任何改变的原始$CPI_{original}$ ：

$$CPI_{original} = \sum_{i=1}^{n}(\frac{I_i}{IC} * CPI_i) = 25\% * 4 + 75\% * 1.3333 = 1.999975 \approx 2.0$$

用原始的 $CPI_{original}$ 减去改进了FPSRQ功能所节省的时间，就可以计算出改进FPSRQ方案的 ：

$$CPI_{with\ new\ FPSRQ} = CPI_{original} - 2\% * (CPI_{oldFPSRQ} - CPI_{ofnewFPSRonly})$$

$$= 2.0 - 2\%*(20-2) = 1.64$$

或 $CPI_{\text{with new FPSRQ}} =$

$1.3333 \times 75\% + (4*0.25-20*0.02) + 2 \times 2\%$

$=0.999975+0.6+0.04 \approx 1.64$

可以用同样的方法计算改进所有FP方案的CPI，或通过将改进FP的CPI值和非FP的CPI值加起来得到。

$$CPI_{newFP} = CPI_{original} - 25\%（CPI_{oldFP} - CPI_{newFP}）$$

$$= 2.0 - 25\% \times （4-2.5）= 1.625 \quad 或$$

$$CPI_{newFP} = (25\% * 2.5) + (75\% * 1.3333) \approx 1.625$$

因为改进所有FP方案的CPI较小，对CPU的性能提高更多。

下面我们来分析改进所有FP方案的加速比：

$$\frac{CPU时间_{original}}{CPU时间_{new}} = \frac{IC * 时钟周期 * CPI_{original}}{IC * 时钟周期 * CPI_{new}} = \frac{CPI_{original}}{CPI_{new}}$$

$$S_{newFP} = \frac{CPI_{original}}{CPI_{newFP}} = \frac{2.0}{1.625} \approx 1.2308$$

$$S_{with\ new\ FPSQR} = \frac{CPI_{original}}{CPI_{with\ new\ FPSQR}} = \frac{2.0}{1.64} \approx 1.2195$$

可见，以上结果与前面用Amdahl定律计算出的结果是相同的。

Classroom Test

- 某计算机系统采用浮点运算部件后，使浮点运算速度提高到原来的25倍，而系统运行某一程序的整体性能提高到原来的4倍，试计算该程序中浮点操作所占的比例。