计算机体系结构--

基本概念

zhaofangbupt@163.com

北京郵電大學

BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

zhaofangbupt@163.com

主要内容

1. 计算机系统结构的概念
2. 计算机系统结构的发展
3. 影响体系结构的因素
4. 定量分析技术基础
5. 体系结构中并行性的发展

zhaofangbupt@163.com

计算机系统设计者的主要任务

- 计算机系统设计者的任务
  - 指令系统的设计、功能的组织、逻辑设计和物理实现
- 计算机系统设计者主要考虑因素
  - 确定所设计计算机系统的功能需求
  - 软、硬件的功能分配
    - 软件与硬件功能的合理分配才能设计出性能价格比最佳的计算机
  - 设计出符合未来技术发展的系统结构

zhaofangbupt@163.com

计算机系统设计者主要考虑因素

- 确定所设计计算机系统的功能需求
  - 应用领域：是专用还是通用？是面向科学计算还是面向商用处理？是桌面机、服务器还是嵌入式计算机？
  - 软件兼容层次：要求在高级语言层兼容？要求在目标代码或二进制代码层兼容？
  - 操作系统要求：为支持选定的操作系统所必需的特性
  - 标准：如浮点数标准，IEEE754；I／O总线标准有VME，Sbus，PCI，SCSI等

zhaofangbupt@163.com

计算机系统设计者主要考虑因素

- 软、硬件的功能分配
  - 提高硬件功能比例可提高性能，减少程序所需存储空间，但会降低硬件利用率和计算机系统的灵活性及适应性
  - 提高软件功能比例可降低硬件成本，设计容易、改进简单，但运算速度下降，增加软件设计费用和存储器容量
  - 有的特殊要求需要配置相应的硬件
  - 对设计方案的选择，必须考虑到设计的复杂性

zhaofangbupt@163.com

计算机系统设计者主要考虑因素

- 设计出符合未来技术发展的系统结构
  - 计算机实现的主要技术包括：
    - 集成电路、半导体DRAM、磁盘和网络实现技术
  - 软件技术发展重要趋势是：
    - 程序及其数据所使用的存储空间越来越大
    - 高级语言在很多应用领域取代了汇编语言，这使编译器的地位更加重要，它与系统结构之间的相互支持可以有效地提高处理器运行程序的效率

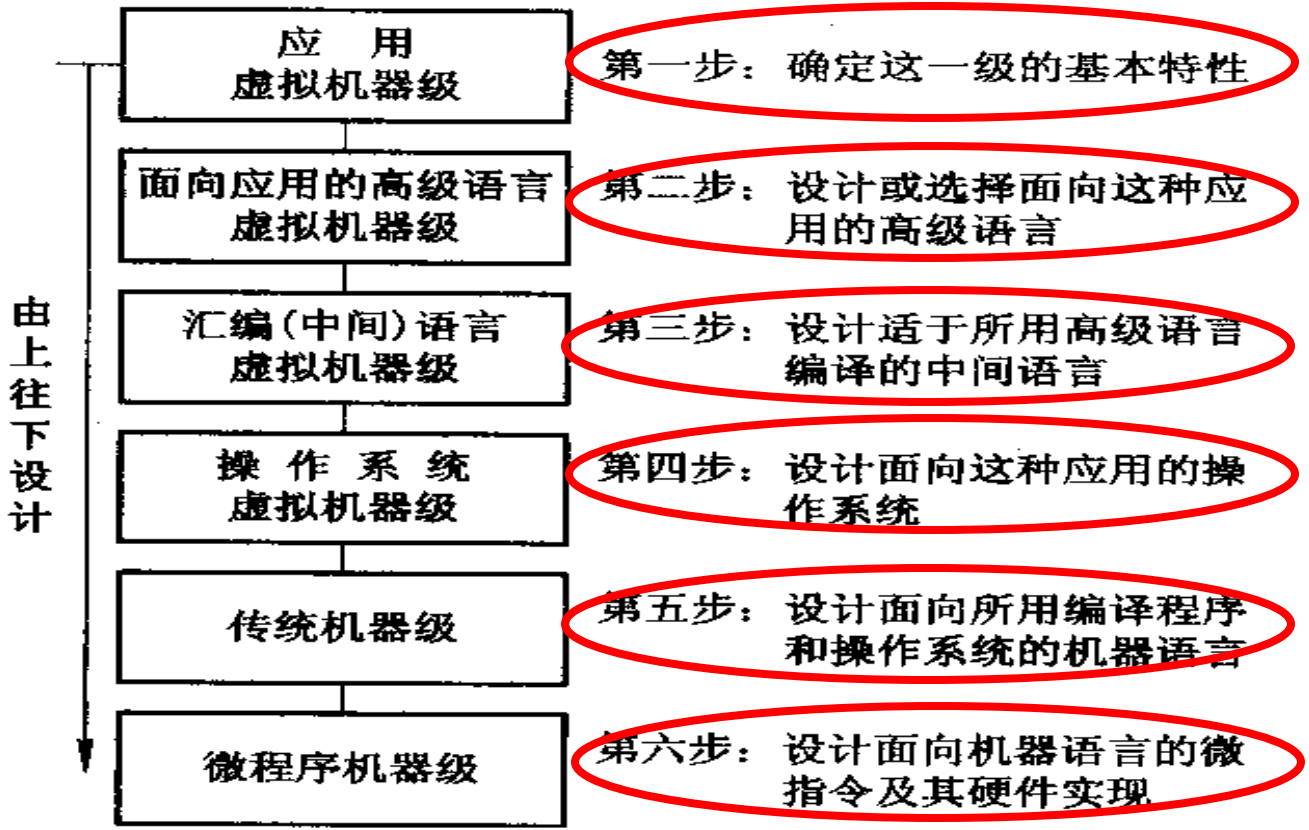
zhaofangbupt@163.com

计算机系统设计的主要方法

- “由上而下”设计
  - 从层次结构中最上面一级开始，逐层往下设计各层机器
- “由下而上”设计
  - 从层次结构中最下面一层开始，逐层往上设计各层机器
- “从中间开始”设计
  - 综合考虑软硬件的分工，从中间开始设计

zhaofangbupt@163.com

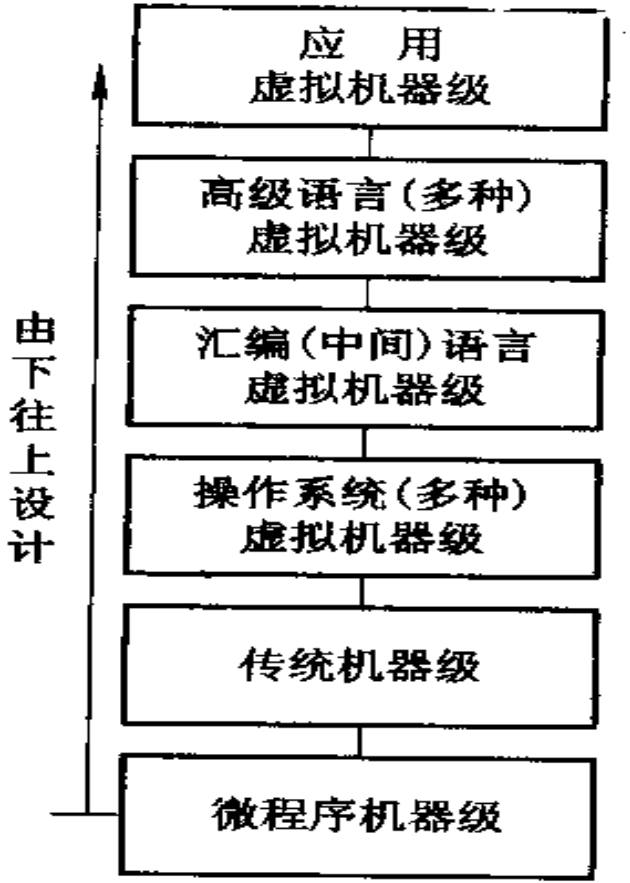
“由上往下”（top-down）设计



zhaofangbupt@163.com

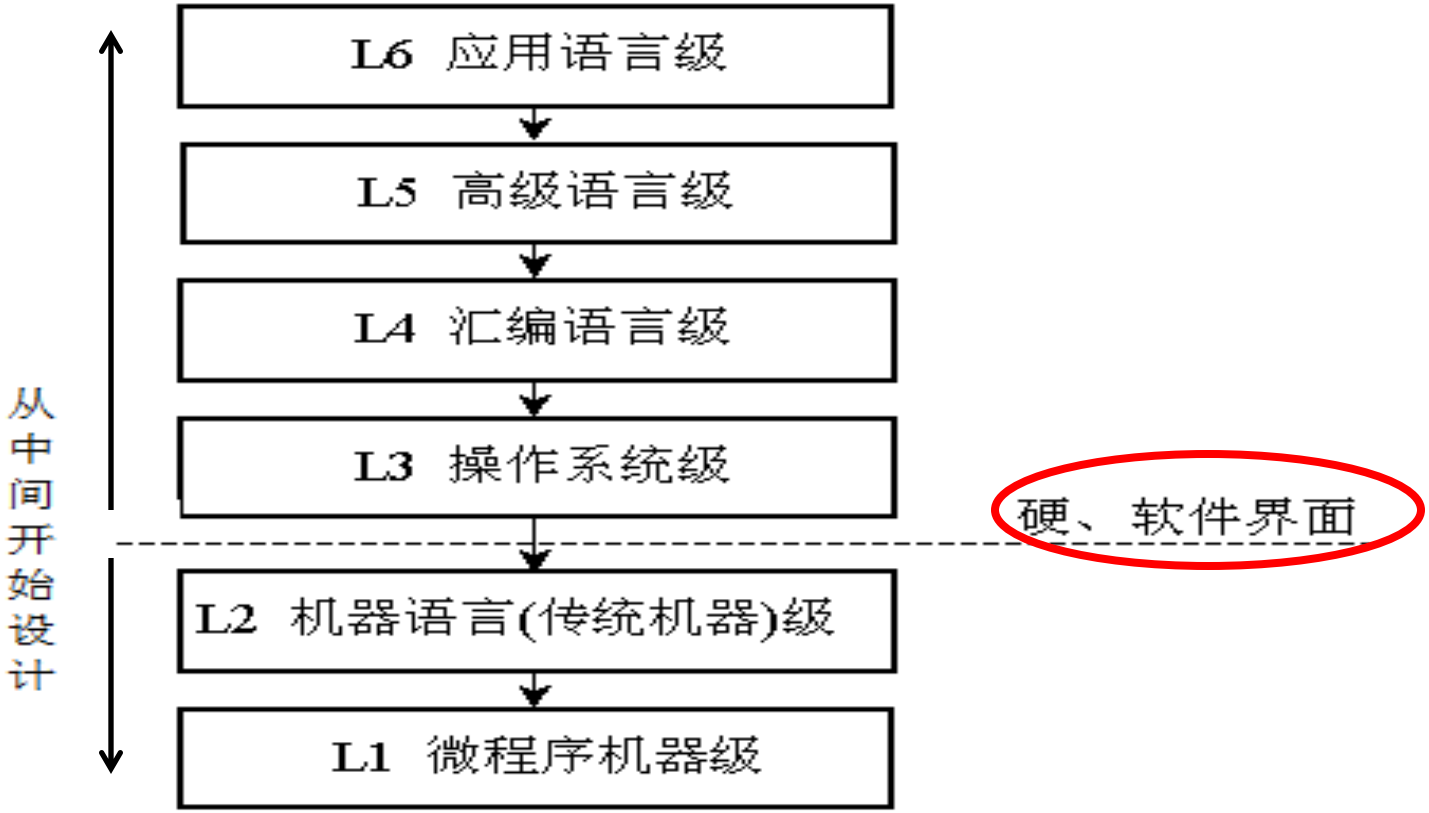
“由下往上”(bottom-up)设计

- 计算机发展的早期，这种设计方法被广泛采用
- 软件技术完全处于被动状态，这会造成软件和硬件的脱节，使整个系统的效率降低
- 目前该方法很少再采用



zhaofangbupt@163.com

从中间开始(middle-out)设计



zhaofangbupt@163.com

计算机系统设计的定量原理

- 加快经常性事件的速度
  - 是计算机设计中最重要且应用最广泛的设计原则
- Amdahl定律
  - 描述了对经常性优化实现能在多大程度上改进整个系统的性能，给出定量的说明
- CPU性能公式
  - 用于衡量CPU的性能
- 程序的局部性原理
  - 是构建存储体系和设计Cache的理论基础

zhaofangbupt@163.com

加快经常性事件的速度

- 在计算机设计中，经常性事件速度的加快能够显著提高整个系统的性能
  - CPU中的两个数相加时，相加结果可能产生溢出，出现溢出的情况比较少见，不溢出才是比较常见的情况
  - 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，应优先优化这两个单元
- 通常经常性事件处理比较简单，更容易优化实现
- 优化是指分配更多的资源、达到更高的性能或者分配更多的电能等

zhaofangbupt@163.com

Amdahl定律

- 加快某部件执行速度所获得的系统性能(加速比),与该部件在系统中的总执行时间（或重要性）的比例有关
- 加速比的定义:

$$\text{加速比} = \frac{\text{采用改进措施后的性能}}{\text{没有采用改进措施前的性能}} = \frac{\text{没有采用改进措施前执行某任务的时间}}{\text{采用改进措施后执行某任务的时间}}$$

- 加速比说明了改进后计算机比改进前快了多少倍

zhaofangbupt@163.com

Amdahl定律

- 加速比依赖于两个因素
- 执行某任务的总时间可分为能被改进和不能改进两部分
  - 可改进比例Fe：在改进前的系统中，可改进部分的执行时间在总的执行时间中所占的比例
    - 它总是小于等于1
  - 部件加速比Se：可改进部分改进以后性能提高的倍数
    - 它是改进前所需的执行时间与改进后执行时间的比
    - 一般情况下部件加速比是大于1

zhaofangbupt@163.com

Amdahl定律

- 设T_0 = 没有采用改进措施前执行某任务的时间

  T_n = 采用改进措施后执行某任务的时间
- 一个需运行60秒的程序中有20秒的运算可以加速

  则：可改进比例Fe = 20/60
- 如果该系统改进后，可改进部分的执行时间减少为5秒

  则：部件加速比Se = 20/5

zhaofangbupt@163.com

Amdahl定律

- 改进后程序的总执行时间

$$\text{总执行时间}_{\text{改进后}} = \text{不可改进部分的执行时间} + \text{可改进部分改进后的执行时间}$$

$$\text{总执行时间}_{\text{改进后}} = (1-\text{可改进比例}) \times \text{总执行时间}_{\text{改进前}} + \frac{\text{可改进比例} \times \text{总执行时间}_{\text{改进前}}}{\text{部件加速比}}$$

$$= \left[(1-\text{可改进比例}) + \frac{\text{可改进比例}}{\text{部件加速比}}\right] \times \text{总执行时间}_{\text{改进前}}$$

zhaofangbupt@163.com

Amdahl定律

- 系统加速比为改进前与改进后总执行时间之比

$$\text{加速比} = \frac{\text{总执行时间}_{\text{改进前}}}{\text{总执行时间}_{\text{改进后}}}$$

$$= \frac{1}{(1-\text{可改进比例}) + \frac{\text{可改进比例}}{\text{部件加速比}}}$$

zhaofangbupt@163.com

Amdahl定律

- 按照Fe和Se描述模式，系统的加速比Sp为：

$$Sp = \frac{T_0}{T_n} = \frac{1}{(1 - Fe) + Fe / Se}$$

- 其中：
  - 当Fe为0，即没有可改进部分时，Sp为1
  - 可见：性能的提高幅度受改进部分所占比例的限制
  - 当Se→∞时，则Sp→1/(1-Fe)
  - 可获取性能改善的极限值受到Fe值的约束

zhaofangbupt@163.com

Amdahl定律-举例

- 【例1】某人一天完成A、 B、C三件任务，所用时间分别为：

任务	A	B	C
完成需要的时间	1	3	6

- 如果某件工作的效率单独提高一倍，则加速比为：

任务	A	B	C	T(执行时间)	加速比S_p
A效率提高一倍	0.5	3	6	9.5	1.05
B效率提高一倍	1	1.5	6	8.5	1.18
C效率提高一倍	1	3	3	7	1.48

zhaofangbupt@163.com

Amdahl定律-举例

- 【例2】将计算机系统中某一功能的处理速度提高到原来的20倍，但该功能的处理时间仅占整个系统运行时间的40%，则采用此提高性能的方法后，能使整个系统的性能提高多少？
- 解 由题可知，可改进比例 = 40% = 0.4

  部件加速比 = 20

zhaofangbupt@163.com

Amdahl定律-举例

- 根据Amdahl定律可知：

$$Sp = \frac{T_0}{T_n} = \frac{1}{(1 - Fe) + Fe / Se}$$

$$\text{总加速比} = \frac{1}{0.6 + \frac{0.4}{20}} = 1.613$$

- 采用此提高性能的方法后，能使整个系统的性能提高到原来的1.613倍

zhaofangbupt@163.com

Amdahl定律-举例

- 【例3】已知求浮点数平方根FPSQR的操作占整个测试程序执行时间的20%。比较两种设计方案:一种是采用FPSQR硬件，使FPSQR操作的速度加快到10倍。另一方法是使所有浮点数据指令FP的速度加快到2倍，而FP指令占整个执行时间的50%。

  解：Fe_FPSQR = 0.2，Se_FPSQR = 10，

  Fe_FP = 0.5，Se_FP = 2，

zhaofangbupt@163.com

Amdahl定律-举例

$$\therefore\ S_{FPSQR} = \frac{1}{(1-0.2)+\frac{0.2}{10}} = 1.22$$

$$\therefore\ S_{FP} = \frac{1}{(1-0.5)+\frac{0.5}{2}} = 1.33$$

zhaofangbupt@163.com

Amdahl定律-举例

□【例4】 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20倍，而系统运行某一程序的整体性能提高到原来的5倍，试计算该程序中浮点操作所占的比例。

解：由题可知，部件加速比 = 20，系统加速比 = 5

根据Amdahl定律可知

$$\text{加速比} = \frac{1}{(1-\text{可改进比例}) + \frac{\text{可改进比例}}{\text{部件加速比}}}$$

zhaofangbupt@163.com

Amdahl定律-举例

$$5 = \frac{1}{(1 - \text{可改进比例}) + \frac{\text{可改进比例}}{20}}$$

设：可改进比例 = Fe，则：

$$5 = \frac{1}{(1 - Fe) + \frac{Fe}{20}}$$

由此可得：可改进比例 = 84.2%

即程序中浮点操作所占的比例为84.2%

zhaofangbupt@163.com

Amdahl定律-举例

- 【例5】假定一执行部件改进后速度提高到10倍，改进后被改进部件执行时间占系统总运行时间的50%。问改进后系统的加速比Sp是多少？

  解：设T_0(T_n)改进前(后)计算机执行某任务的总时间，　Fe是被改进部分的时间百分比

  于是　　$Tn=[(1-Fe)+Fe/10]\times T_0$

  由题意　　$Fe/10=(1-Fe)$

  从而　　$Fe=10/11=0.\ 91$

  所以　　$Sp=T_0/Tn=5.5$　（1/0.182）

zhaofangbupt@163.com

Amdahl定律总结

- Amdahl定律：一种性能改进的递减规则
  - 如果仅仅对计算任务中的一部分做性能改进，则改进得越多，所得到的总体性能的提升就越有限
- 重要推论：如果只针对整个任务的一部分进行改进和优化，那么所获得的加速比不超过

  1/（1—可改进比例）

zhaofangbupt@163.com

CPU性能公式

- 执行一个程序所需的CPU时间
  - CPU时间= 执行程序所需的时钟周期数×时钟周期时间
    其中，时钟周期时间是系统时钟频率的倒数
- 每条指令执行的平均时钟周期数CPI（Cycles Per Instruction）
  - CPI = 执行程序所需的时钟周期数 / IC
    IC：所执行的指令条数
- 程序执行的CPU时间可以写成
  - CPU时间 = IC ×CPI ×时钟周期时间

zhaofangbupt@163.com

CPU性能公式

- CPU的性能取决于3个参数
  - 时钟周期时间：取决于硬件实现技术和计算机组成
  - CPI：取决于计算机组成和指令集结构
  - IC：取决于指令集结构和编译技术
- 对CPU性能公式进行进一步细化
- 假设：计算机系统有n种指令
  - CPI_i：第i种指令的处理时间
  - IC_i：在程序中第i种指令出现的次数

  则　　CPU时钟周期数 $= \sum_{i=1}^{n} (CPI_i \times IC_i)$

zhaofangbupt@163.com

CPU性能公式

CPU时间 = 执行程序所需的时钟周期数×时钟周期时间

$$= \sum_{i=1}^{n} (CPI_i \times IC_i) \times \text{时钟周期时间}$$

CPI可以表示为：

$$CPI = \frac{\text{时钟周期数}}{IC} = \frac{\sum_{i=1}^{n} (CPI_i \times IC_i)}{IC} = \sum_{i=1}^{n} (CPI_i \times \frac{IC_i}{IC})$$

其中，(IC_i/IC)反映了第i种指令在程序中所占的比例

zhaofangbupt@163.com

CPU性能公式-举例

- 【例6】某系统中浮点操作FP的比例为25%，平均CPI为4。其他指令的平均CPI约为1.33。浮点平方操作FPSQR比例为2%，CPI为20。分别把浮点平方操作和其他所有浮点操作的CPI减为2，试比较两种方案对系统性能的提高程度？

解：未改进前，每条指令的平均时钟周期（CPI）为

$$CPI = \sum_{i=1}^{n} (CPI_i \times \frac{IC_i}{IC}) = (4*25\%) + (1.33*75\%) = 2.0$$

zhaofangbupt@163.com

CPU性能公式-举例

（1）采用第一种方案：FPSQR操作的CPI由CPI_{FPSQR} = 20 减至CPI'_{FPSQR} = 2，则整个系统的指令平均时钟周期数为：

$CPI_1=CPI-(CPI_{FPSQR}-CPI'_{FPSQR}) \times 2\%=2-(20-2) \times 2\% =1.64$

（2）采用第二种方案：所有FP操作的CPI由CPI_{FP} = 4 减至CPI'_{FP} = 2，则整个系统的指令平均时钟周期数为：

$CPI_2=CPI-(CPI_{FP}-CPI'_{FP}) \times 25\%=2-(4-2) \times 25\% =1.5$

第二种方案优于第一种方案

zhaofangbupt@163.com

CPU性能公式-举例

- 【例7】考虑条件分支指令的两种不同设计方法：

  （1）CPU_A：通过比较指令设置条件码，然后测试条件码进行分支。

  （2）CPU_B：在分支指令中包括比较过程。

在这两种CPU中，条件分支指令都占用2个时钟周期，而所有其他指令占用1个时钟周期。

zhaofangbupt@163.com

CPU性能公式-举例

对于CPU_A，执行的指令中分支指令占20%；由于每条分支指令之前都需要有比较指令，因此比较指令也占20%。由于CPU_A在分支时不需要比较，因此CPU_B的时钟周期时间是CPU_A的1.25倍。问：哪一个CPU更快？如果CPU_B的时钟周期时间只是CPU_A的1.1倍，哪一个CPU更快呢？

zhaofangbupt@163.com

CPU性能公式-举例

解：我们不考虑所有系统问题，所以可用CPU性能公式。占用2个时钟周期的分支指令占总指令的20%，剩下的指令占用1个时钟周期。所以

$$CPI_A = 0.2 \times 2 + 0.80 \times 1 = 1.2$$

则CPU_A性能为

$$\text{总CPU时间}_A = IC_A \times 1.2 \times \text{时钟周期}_A$$

根据假设，有

$$\text{时钟周期}_B = 1.25 \times \text{时钟周期}_A$$

在CPU_B中没有独立的比较指令，所以CPU_B的程序量

zhaofangbupt@163.com

CPU性能公式-举例

为CPU_A的80%，分支指令的比例为

$$20\%/80\% = 25\%$$

这些分支指令占用2个时钟周期，而剩下的75%的指令占用1个时钟周期，因此

$$CPI_B = 0.25 \times 2 + 0.75 \times 1 = 1.25$$

因为CPU_B不执行比较，故

$$IC_B = 0.8 \times IC_A$$

zhaofangbupt@163.com

CPU性能公式-举例

因此CPU_B性能为

$$\text{总CPU时间}_B = IC_B \times CPI_B \times \text{时钟周期}_B$$
$$= 0.8 \times IC_A \times 1.25 \times (1.25 \times \text{时钟周期}_A)$$
$$= 1.25 \times IC_A \times \text{时钟周期}_A$$

在这些假设下，尽管CPU_B执行指令条数较少，CPU_A因为有着更短的时钟周期，所以比CPU_B快

zhaofangbupt@163.com

CPU性能公式-举例

如果CPU_B的时钟周期时间仅仅是CPU_A的1.1倍，则

$$时钟周期_B = 1.10 \times 时钟周期_A$$

CPUB的性能为

$$总CPU时间_B = IC_B \times CPI_B \times 时钟周期_B$$

$$= 0.8 \times IC_A \times 1.25 \times (1.10 \times 时钟周期_A)$$

$$= 1.10 \times IC_A \times 时钟周期_A$$

因此CPU_B由于执行更少指令条数，比CPU_A运行更快

zhaofangbupt@163.com

程序的局部性原理

- 对大量程序运行过程统计和分析发现，程序经常会重复使用它最近使用过的指令和数据，具体而言，程序花费90%的执行时间来执行10%的代码
- 程序的时间局部性
  - 程序即将用到的信息很可能就是目前正在使用的信息
- 程序的空间局部性
  - 程序即将用到的信息很可能与目前正在使用的信息在空间上相邻或者临近

zhaofangbupt@163.com

计算机系统的性能评测

- 性能评测包括“评估”与“测试”两种方法
- 性能评估是基于一些原始数据进行逻辑推算，典型的评估指标有MIPS（million instructions per second）、MFLOPS（million floating point operations per second）
- 性能测试是用“尺子”来度量计算机的性能，作为尺子的程序称为基准测试程序（Benchmark Program）

zhaofangbupt@163.com

执行时间和吞吐率

- 如何评测一台计算机的性能，与测试者看问题的角度有关
  - 用户关心的是：单个程序的执行时间（执行单个程序所花的时间很少）
  - 数据处理中心的管理员关心的是：吞吐率（在单位时间里能够完成的任务很多）

zhaofangbupt@163.com

执行时间和吞吐率

- 假设两台计算机为X和Y，“X比Y快”的意思是：
  - 对于给定任务，X的执行时间比Y的执行时间少
  - X的性能是Y的n倍，即

$$\frac{\text{执行时间Y}}{\text{执行时间X}} = n$$

  - 而执行时间与性能成反比，即

$$\frac{\text{执行时间Y}}{\text{执行时间X}} = \frac{\frac{1}{\text{性能Y}}}{\frac{1}{\text{性能X}}} = \frac{\text{性能X}}{\text{性能Y}}$$

zhaofangbupt@163.com

执行时间和吞吐率

- 执行时间可以有多种定义：
  - 计算机完成某一任务所花费的全部时间，包括磁盘访问、存储器访问、输入/输出、操作系统开销等
  - CPU时间：CPU执行所给定的程序所花费的时间，不包含I/O等待时间以及运行其他程序的时间
    - 用户CPU时间：用户程序所耗费的CPU时间
    - 系统CPU时间：用户程序运行期间操作系统耗费的CPU时间

zhaofangbupt@163.com

CPU性能评估指标

- 两个典型的性能评估指标：
  - MIPS：适合评估标量机
  - MFLOPS：用来评估浮点操作
- MIPS（million instructions per second）：
  - 表示每秒百万条指令数。对于一个给定的程序，MIPS定义为 $$\mathrm{MIPS}=\frac{\text{指令条数}}{\text{执行时间}\times 10^6}=\frac{\text{时钟频率}}{\mathrm{CPI}\times 10^6}$$
  - 程序的执行时间Te为： $$T_e=\frac{\text{指令条数}}{\mathrm{MIPS}\times 10^6}$$

zhaofangbupt@163.com

CPU性能评估指标

- 在使用MIPS评估机器的性能应该注意以下问题：
  - MIPS只适用于评价标量机的性能
  - MIPS依赖于指令系统
- 为了对不同指令系统的计算机进行更合理的性能比较，可以采用相对MIPS指标，选择一个参照计算机进行性能比较：

$$MIPS_v = \frac{T_{ref}}{T_v} \times MIPS_{ref}$$

- 现代计算机一般采用GIPS和TIPS来衡量

zhaofangbupt@163.com

CPU性能评估指标

- MFLOPS（million floating point operations per second）
  - 每秒百万次浮点操作次数

$$\text{MFLOPS}=\frac{\text{程序中的浮点操作次数}}{\text{执行时间}\times 10^{6}}$$

- MFLOPS比较适用于衡量有大量浮点运算操作的高性能计算机的性能
  - MFLOPS用浮点操作次数来评估性能，而不是用指令来衡量的
  - MFLOPS可以用来比较两种不同的机器的浮点性能

zhaofangbupt@163.com

CPU性能评估指标

- 问题：
  - MFLOPS只能用来衡量机器浮点操作的性能，而不能体现机器的整体性能
  - MFLOPS评价不同机器并非完全可靠，不同机器上浮点运算指令集不同
  - 不同浮点运算的复杂性是不同的
- 通常采用GFLOPS和TFLOPS来衡量，分别表示每秒执行十亿次和万亿次浮点操作
- 用MIPS/W表示单位功耗能达到的MIPS指标

zhaofangbupt@163.com

CPU性能评估指标

- 计算机性能通常用峰值性能（Peak performance）和持续性能（Sustained performance）来评价
  - 上述分析方法计算得到的MIPS和MFLOPS指标都是峰值性能指标
  - 峰值性能是指在理想情况下计算机系统可获得的最高理论性能，它不能反映出系统的实际性能
  - 实际性能又称持续性能，其值只有峰值性能的5%~35%
  - 实际程序运行时会受到硬件结构、操作系统、算法设计和编程等因素的影响

zhaofangbupt@163.com

基准测试程序

- 用于测试和比较性能的基准测试程序的最佳选择是真实应用程序：
  - 例如编译器
- 以前常采用简化了的程序：
  - 核心测试程序：从真实程序中选出的关键代码段构成的小程序
  - 小测试程序：简单的只有几十行的小程序
  - 合成的测试程序：人工合成出来的程序
    - Whetstone与Dhrystone是最流行的合成测试程序

zhaofangbupt@163.com

基准测试程序

- 从测试性能的角度来看，上述测试程序现在已经不可信：
  - 这些这些程序比较小，具有片面性
  - 系统结构设计者和编译器的设计者可以“合谋”把他们的计算机面向这些测试程序进行优化设计，使得该计算机显得性能更高
- 性能测试的结果除了和采用什么测试程序有关以外，还和在什么条件下进行测试有关

zhaofangbupt@163.com

基准测试程序

- 基准测试程序设计者对制造商的要求
  - 采用同一种编译器
  - 对同一种语言的程序都采用相同的一组编译标志
- 是否允许修改测试程序的源程序，有三种不同的处理方法：
  - 不允许修改
  - 允许修改，但因测试程序很复杂或者很大，几乎是无法修改
  - 允许修改，只要保证最后输出的结果相同

zhaofangbupt@163.com

基准测试程序

- 基准测试程序套件：由各种不同的真实应用程序构成
  - 能比较全面地反映计算机在各个方面的处理性能
- 不同的基准测试程序，侧重点不一样：
  - 有的测试CPU性能
  - 有的测试文件服务器性能
  - 有的测试科学计算性能
  - 有的测试输入输出性能
  - 有的测试网络通讯速度
  - ……

zhaofangbupt@163.com

基准测试程序

- SPEC基准测试程序集
  - SPEC推出第一组基准测试程序SPEC89，包含10个程序
- SPEC原来主要是测试CPU性能
- 现在强调开发能反映真实应用的基准测试程序集
- 已推广至测试高性能计算机系统、网络服务器和商业应用服务器等

zhaofangbupt@163.com

基准测试程序

TPC基准测试程序

- TPC的功能是制定商务应用基准测试程序的标准规范、性能和价格度量，并管理测结果的发布
- TPC-C：在线事务处理（OLTP）基准测试程序
- TPC-D：决策支持的基准程序
- TPC-E：在线事务处理（OLTP）基准测试程序，具有较高的事务处理率

zhaofangbupt@163.com

基准测试程序

- 针对服务器性能的测试：
  - SPEC测试注重全面衡量Web应用中Java企业级应用服务器性能
  - 而TPC测试体系则注重在线处理能力和数据库查询能力
  - SPEC测试针对的是服务器硬件，则TPC测试针对的是一整套系统，它体现了服务器厂商在高端关键领域的方案开发和优化能力，因此，对高端用户选型整套系统来说，TPC测试无疑更具参考价值

zhaofangbupt@163.com

基准测试程序

- LinPACK基准测试程序
  - LinPACK是线性系统软件包（Linear system package）用于求解稠密线性代数方程组的数学软件，包含大量浮点加法和浮点乘法运算，该软件包是浮点性能的测试基准程序，用以评价高性能计算机的浮点性能
- LinPACK测试包括：
  - LinPACK100求解规模为100阶的稠密线性代数方程组
  - LinPACK1000求解规模为1000阶的线性代数方程组
  - HPL用于高度并行计算基准测试，它对数组大小N没有限制，求解问题的规模可以改变

zhaofangbupt@163.com

性能比较

- 计算机系统的持续性能可以用各种基准测试程序运行结果的某种统计平均值来表示，不同机器的性能就可以通过某种平均值来进行比较
- 常用的统计平均值表示法有三种：
  - 算术性能平均值（arithmetic mean）
  - 调和性能平均值（harmonic mean）
  - 几何性能平均值（geometric mean）

zhaofangbupt@163.com

性能比较

算术性能平均值

- 算术性能平均值是n个测试程序运算时间或运算速度的算术平均值
- 设计算机运行各测试程序运算时间测量结果分别为T_1、T_2、…、T_n，则该计算机的执行时间算术平均值的计算公式为：

$$S_m = \frac{1}{n}\sum_{i=1}^{n} T_i$$

算术平均值也可以采用计算机执行各种测试程序的速度指标的平均值

zhaofangbupt@163.com

性能比较

- 调和性能平均值
  - 如果性能是用速度表示，则可以采用调和平均值
  - 设计算机运行各测试程序的速度测量指标分别为R_1、R_2、…、R_n，则该计算机的调和平均值的计算公式为：

$$Hm = \frac{n}{\sum_{i=1}^{n} \frac{1}{Ri}} = \frac{n}{\sum_{i=1}^{n} Ti}$$

- 调和平均值与测试程序总的执行时间成反比。它和算术平均值一样，性能比较的结果与参考计算机的选择有关

zhaofangbupt@163.com

性能比较

加权调和平均值

- 如果考虑工作负荷中各程序不会以相等的比例出现，则可以使用加权调和平均值公式：

$$Hm = \frac{1}{\sum_{i=1}^{n} \frac{Wi}{Ri}} = \frac{1}{\sum_{i=1}^{n} WiTi}$$

- 其中：$\sum_{i=1}^{n} w_i = 1$

zhaofangbupt@163.com

性能比较

几何性能平均值

- 基本思想来源与性能规格化的方法，即以某台计算机的性能作为参考标准，其他计算机性能则除以该参考标准而获得一个比值。如果比值相等，则可以认为这些计算机具有相同的性能
- 一台计算机的几何平均值的计算公式为：

$$G_m = \sqrt[n]{(\prod_{i=1}^{n} R_i)} = \sqrt[n]{(\prod_{i=1}^{n} \frac{1}{T_i})} = R_1^{\frac{1}{n}} \times R_2^{\frac{1}{n}} \times \ldots \times R_n^{\frac{1}{n}}$$

zhaofangbupt@163.com

性能比较

- 几何平均值的一个重要特点是，两个计算机速度的几何平均值之比等于速度比的几何平均值

$$\frac{G_m(X)}{G_m(Y)} = G_m\left(\frac{X}{Y}\right)$$

- 由此可得：

$$\frac{G_m(X/Z)}{G_m(Y/Z)} = \frac{G_m(X)/G_m(Z)}{G_m(Y)/G_m(Z)} = \frac{G_m(X)}{G_m(Y)} = G_m\left(\frac{X}{Y}\right)$$

- 在进行性能比较时，与参考计算机的选择无关

zhaofangbupt@163.com

性能比较

- 【例8】下表列出了两个程序在A、B、C3台机器上的执行时间，如何比较三台机器的性能呢?

执行时间 (秒)	A	B	C
程序1	1	10	20
程序2	1000	10	20

zhaofangbupt@163.com

性能比较

- 从该表可以得出：
  - 执行程序1：
    - A机的速度是B机的10倍
    - A机的速度是C机的20倍
    - B机的速度是C机的2倍
  - 执行程序2：
    - B机的速度是A机的100倍
    - C机的速度是A机的50倍
    - B机的速度是C机的2倍

zhaofangbupt@163.com

性能比较

- 总执行时间：
  - 计算机执行所有测试程序的总时间
- 计算结果：
  - B机执行程序1和程序2的速度是A机的50.05倍
  - C机执行程序1和程序2的速度是A机的24.02倍
  - B机执行程序1和程序2的速度是C机的2倍
- 人们还经常用三种平均值来比较
  - 算数平均值
  - 调和平均值
  - 几何平均值

zhaofangbupt@163.com

性能比较

- 两个程序在A、B、C三台计算机上的执行时间

	A机	B机	C机	W（1）	W（2）	W（3）
程序1	1.00	10.00	20.00	0.50	0.909	0.999
程序2	1000.00	10.00	20.00	0.50	0.091	0.001
加权算术平均值A_m（1）	500.50	10.00	20.00			
加权算术平均值A_m（2）	91.91	10.00	20.00			
加权算术平均值A_m（3）	2.00	10.00	20.00			

zhaofangbupt@163.com

内容小结

- 计算机定量分析技术基础
  - 计算机系统设计需要考虑的因素
  - 计算机系统设计方法及其优缺点
  - 计算机系统定量原理
    - 加快经常性事件的速度
    - Amdahl定律（重点内容）
    - CPU性能公式
    - 程序的局部性原理

zhaofangbupt@163.com

内容小结

- 计算机系统性能评测
  - CPU性能指标
  - 基准测试程序
  - 性能比较方法

知识要点

“由上往下”设计	“由下往上”设计	“从中间开始”
Amdahl定律	CPU性能公式	程序局部性原理
CPU性能指标	基准测试程序	性能比较方法

zhaofangbupt@163.com

思考题

- 1. 计算机设计有哪几种方法，各有什么优缺点？
- 2. 计算机系统“从中间开始”方法中的“中间”指的是什么地方？这样设计的好处是什么？
- 3. 计算机系统设计中经常使用的4个定量原理是什么？并说出它们的含义。

Thank You !

zhaofangbupt@163.com

北京郵電大學

BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS