

**翻译**：先把N+1级程序全部转换成N级程序后，再去执行新产生的N级程序，在执行过程中N+1级程序不再被访问。

**解释**：每当一条N+1级指令被译码后，就直接去执行一串等效的N级指令，然后再去取下一条N+1级的指令，依此重复进行。

解释执行比编译后再执行所花的时间多，但占用的存储空间较少。

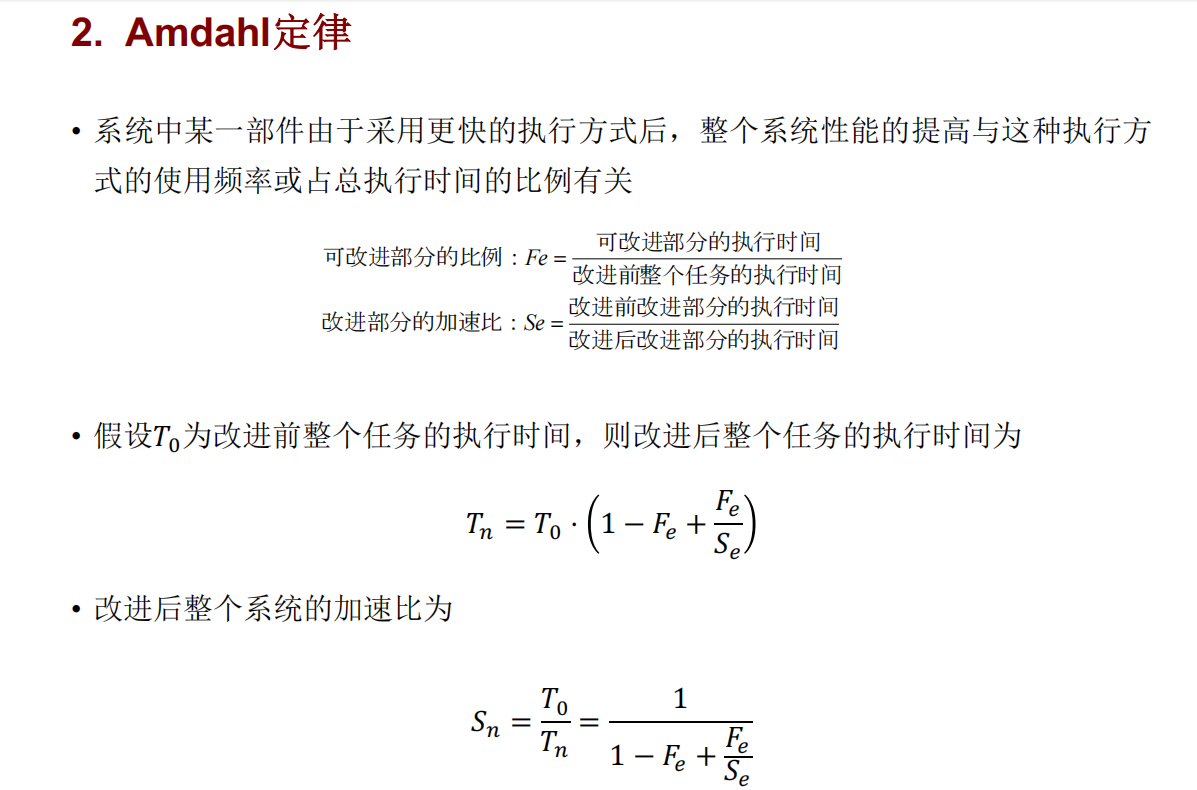
L0-L2级用解释的方法实现，L3-L5级则用翻译的方法实现

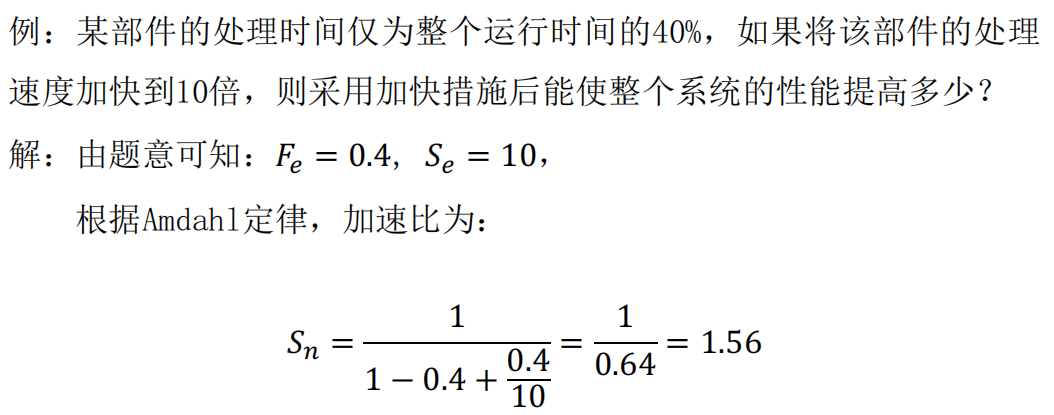
**计算机系统的定量原理**

哈夫曼压缩原理

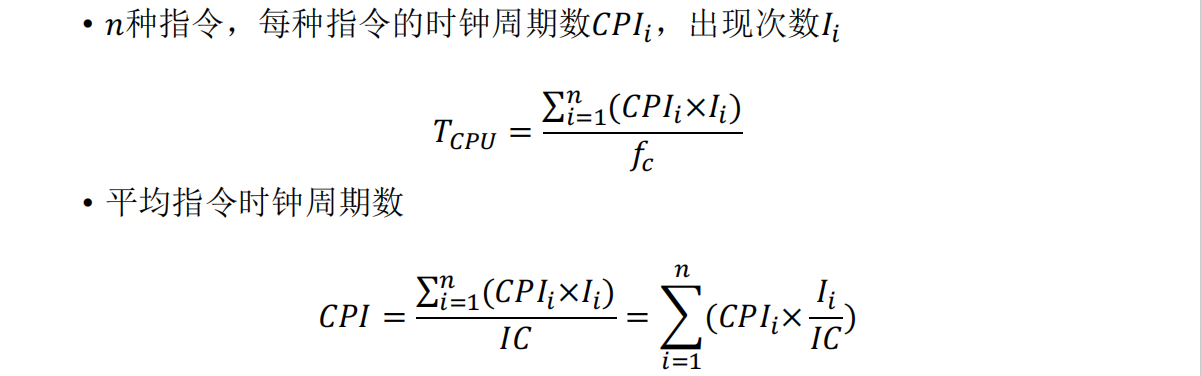
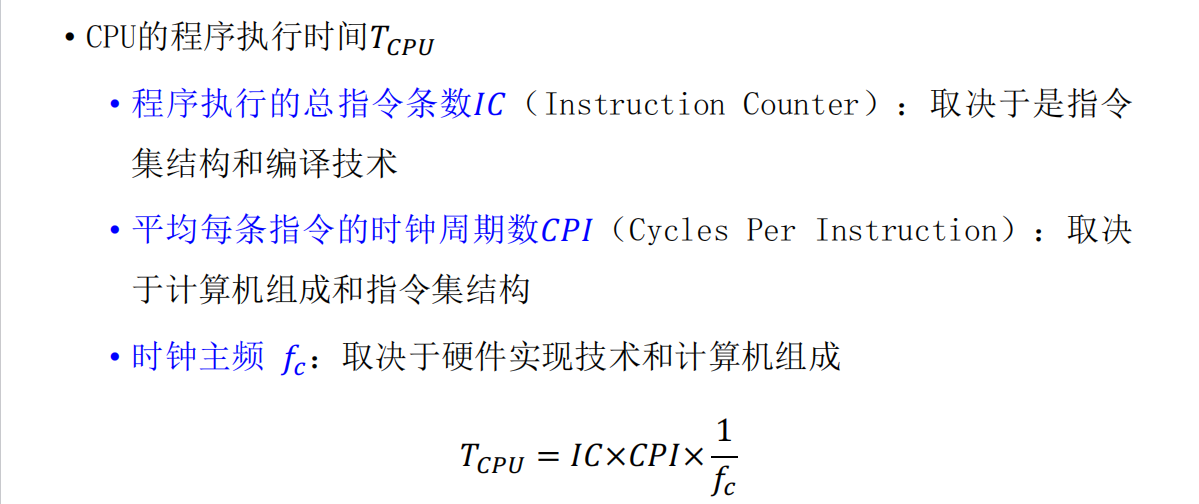
在计算机系统的设计中经常要在不同的方法之间进行折中，尽可能加速处理高概率事件远比加速处理低概率事件对性能的提高要显著。例如，CPU在运算中发生溢出的概率是很低的，为此，设计时可考虑加快不溢出时的运算速度，而对溢出时的运算速度不予考虑。

Amdahl定律（用于加速并行计算）





CPU性能公式



访问的局部性原理

时间局部性：程序中近期被访问的信息项很可能马上将被再次访问。

空间局部性：指那些在访问地址上相邻近的信息项很可能会被一起访问。

**计算机系统设计方法**

由上向下：

面向应用的数学模型→面向应用的高级语言→面向这种应用的操作系统→面向

操作系统和高级语言的机器语言→面向机器语言的微指令系统和硬件实现

**特点**：对于所面向的应用领域，性能和性能价格比很高。随着通用计算机价格降低，目前已经很少采用

由下向上：

根据当时的器件水平，设计微程序机器级和传统机器级**→**根据不同的应用领域设计多种操作系统、汇编语言、高级语言编译器等**→**最后设计面向应用的用户级

**特点：**容易使软件和硬件脱节，整个计算机系统的效率降低

中间开始：

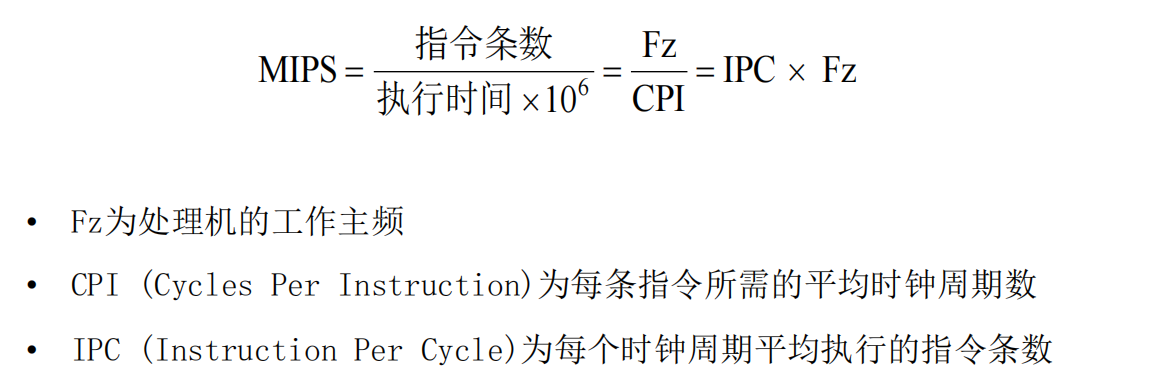
首先定义软硬件的分界面(指令系统、存储系统、输入输出系统、中断系统、硬件对操作系统和编译系统的支持等)，然后各个层次分别进行设计(软件设计人员设计操作系统、高级语言、汇编语言、应用程序等，硬件设计人员设计传统机器、微程序、硬联逻辑等)

**应用场合**：用于系列机的设计

**特点**：软硬件人员结合、同时设计，软硬件功能分配合理。

计算机系统的评价标准：

1. MIPS



优点：直观、方便。目前还经常使用

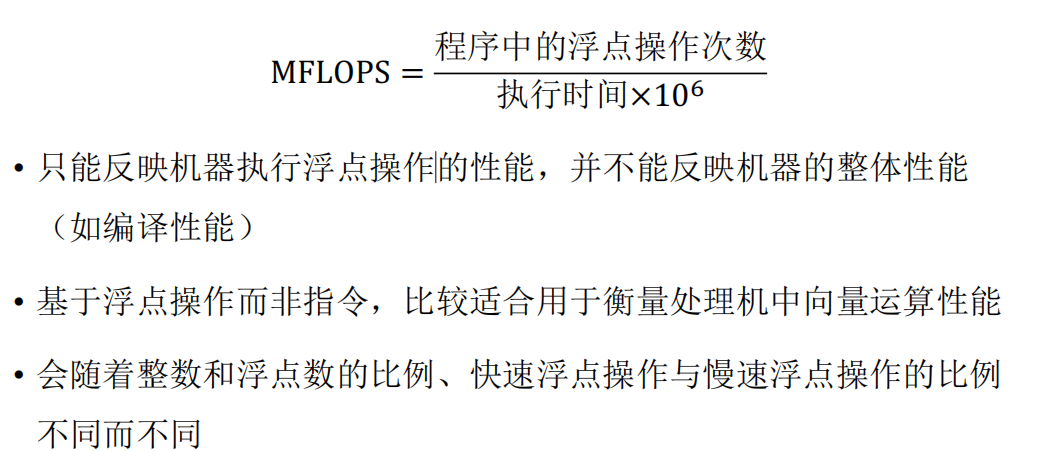
主要缺点:

(1) MIPS依赖于指令集，用MIPS来比较指令集不同的机器的性能好坏是很不准确的

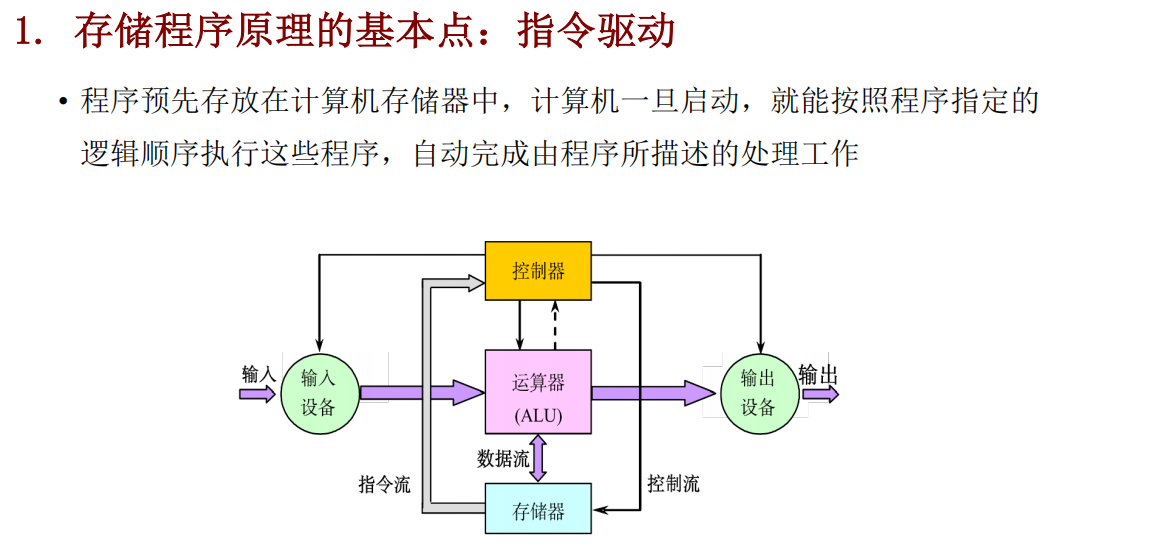
(2) 在同一台机器上，由于指令使用频度差别很大，MIPS会因程序不同而变化

(3) MIPS可能与性能相反。例如，具有可选硬件浮点运算部件的机器，具有优化功能的编译器

1. MFLOPS



冯诺依曼架构：



以运算器为中心,集中控制。

1.在存储器中，指令和数据同等对待。指令和数据一样可以进行运算，即由指令

组成的程序是可以修改的。

2.存储器是按地址访问、按顺序线性编址的一维结构，每个单元的位数是固定的。

3.指令的执行是顺序的。

一般是按照指令在存储器中存放的顺序执行。

程序的分支由转移指令实现。

由指令计数器PC指明当前正在执行的指令在存储器中的地址。

4.指令由操作码和地址码组成。

5.指令和数据均以二进制编码表示，采用二进制运算。

**第2章 指令系统**

机器上直接运行的程序是由指令组成的，指令系统是软件与硬件之间的一个主要分界面，也是他们直接互相沟通的一座桥梁。

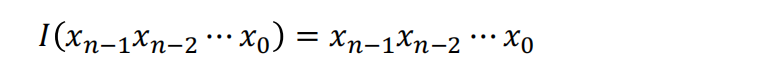
指令系统的发展和改进：

按CISC方向：增强原有指令的功能以及设置更为复杂的新指令，取代原先由软件子程序完成的功能，实现软件功能的硬化。

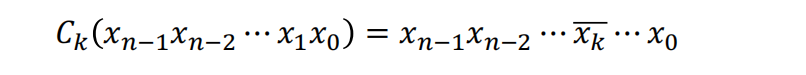
按RISC方向：通过减少指令种类和简化指令功能来降低硬件设计的复杂度，提高指令执行速度。

互连函数

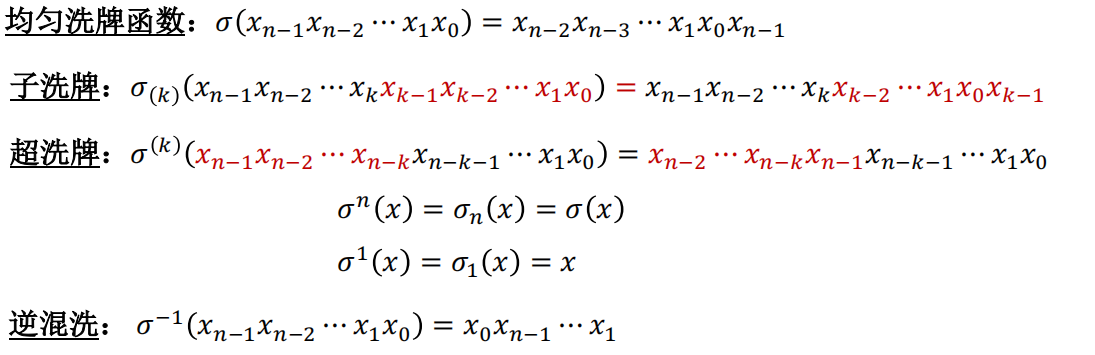
1.恒等函数：相同编号的输入端与输出端一一对应



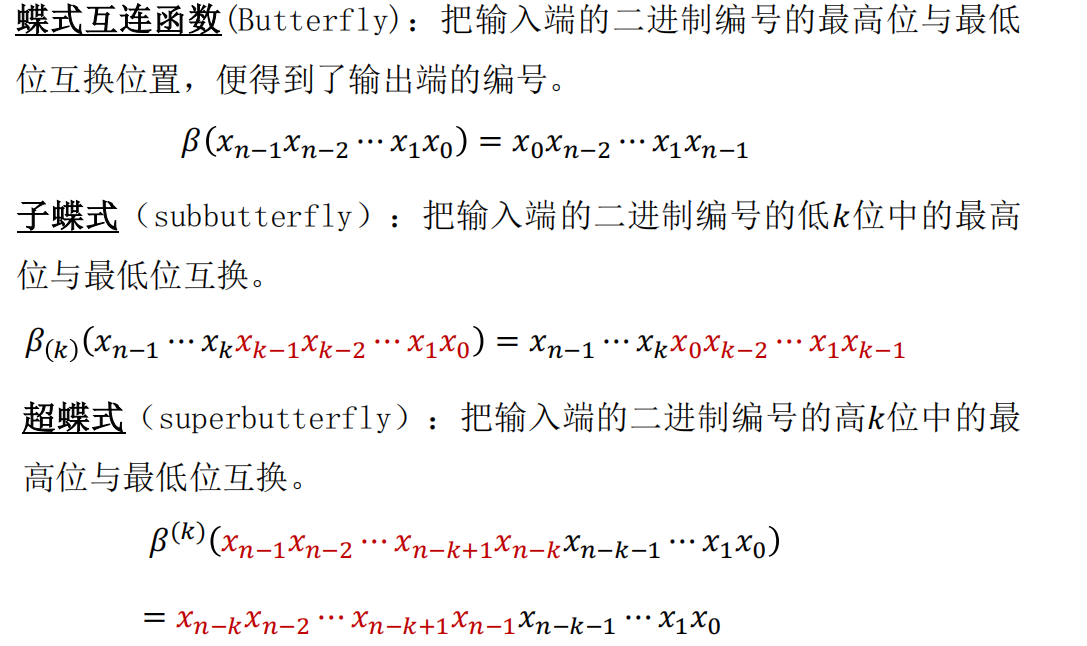
2.交换函数：实现二进制地址编码中第k位互反的输入端与输出端之间的连接



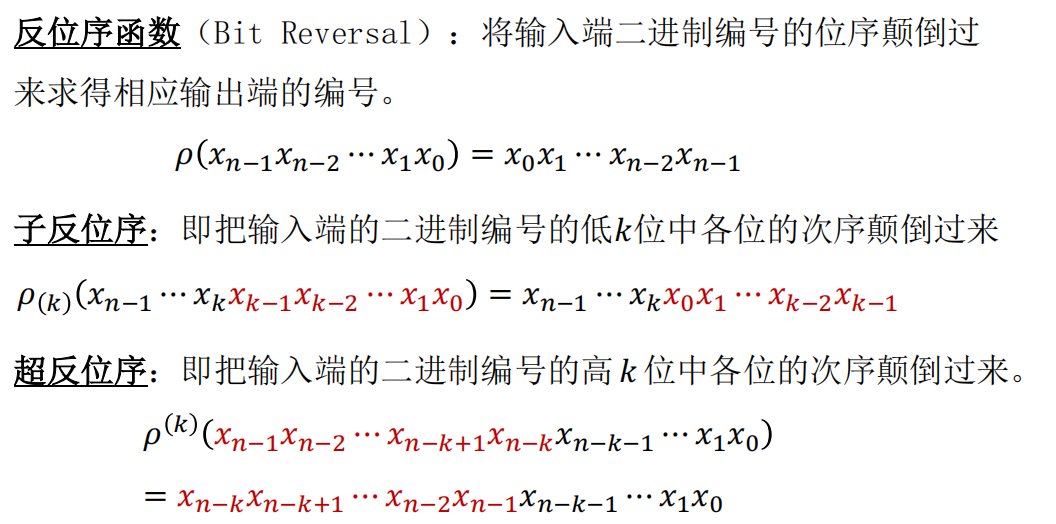
3.均匀洗牌函数：将输入端分成数目相等的两半，前一半和后一半按类似均匀混洗扑克牌的方式交叉地连接到输出端



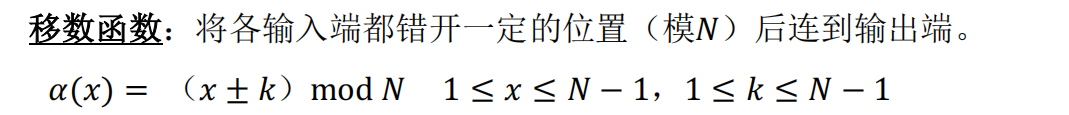
4.碟式互连函数



5.反位序函数



6.移数函数



7.PM2I函数

