# 计算机体系结构 第二章 数据表示与指令系统

***Summarized by Pengxiang Li***

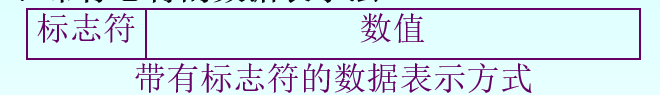
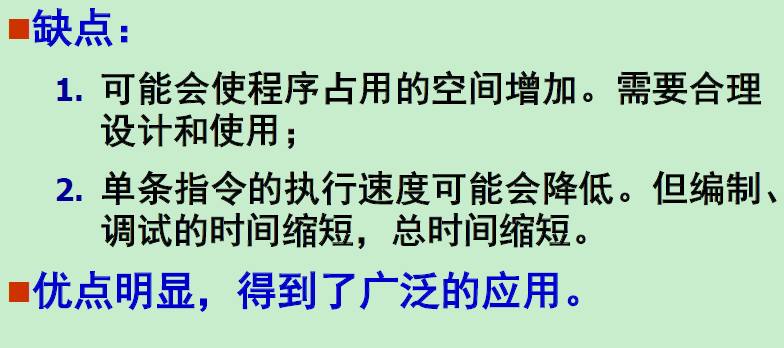
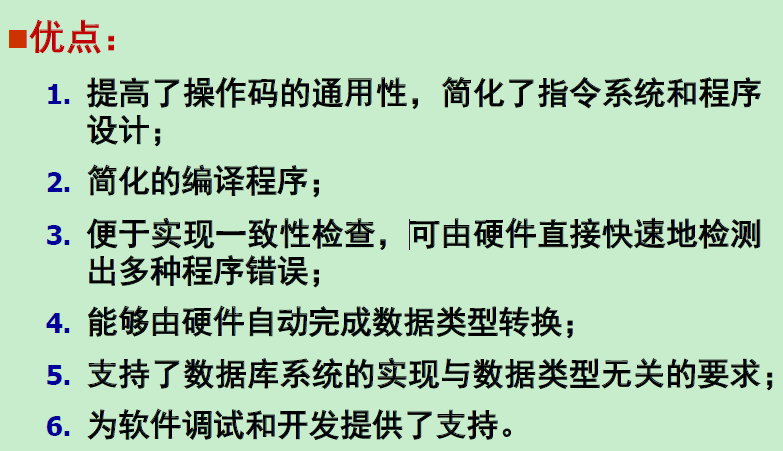
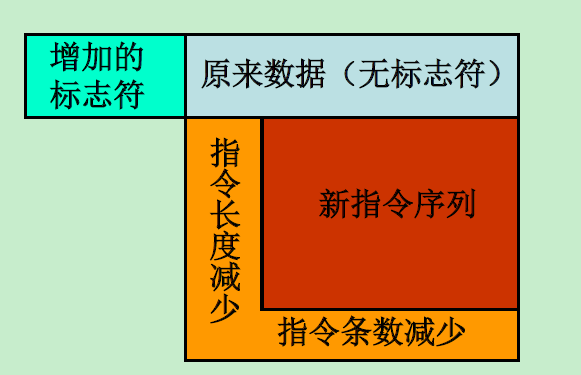
## 2.1 数据表示

1. 数据类型：包括数据表示和数据结构
   1. 数据类型是数据值的集合以及该集合的操作的集合
2. 数据表示： 计算机硬件能够直接识别，可以被指令系统直接调用的那些数据类型。

### 2.1.1 数据表示与数据结构

1. 数据结构
   1. 必须通过软件映像，变换成机器中所具有的各种数据表示来实现。
2. 数据表示与数据结构的关系
   1. 数据结构是通过数据表示实现的
   2. 数据表示是数据结构的组成元素
   3. 数据结构和数据表示都是数据类型的子集
   4. 哪些数据类型用数据表示实现，哪些用数据结构实现，事实上是一个软硬件取舍问题。

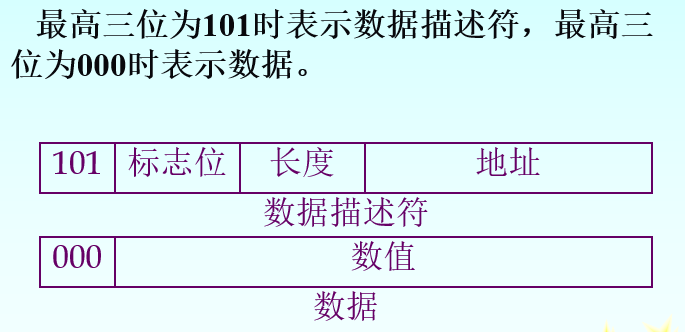
### 2.1.2 高级数据表示

1. 有时候引入一些高级数据表示比在指令系统中增设一条技巧性地指令的意义要大。
2. 有以下几种高级数据表示：
   1. 自定义数据表示
   2. 向量数组数据表示
   3. 堆栈数据表示
3. 自定义数据表示
   1. 由数据本身来表明数据类型，是计算机内的数据具有自定义能力。
   2. 目的：缩短机器语言与高级语言对数据属性说明之间的语义差异。
   3. 分为两类
      1. **带标志符的数据表示**
         1. 标识符由编译器或者其他系统软件建立，对程序员透明
         2. 
         3. 
         4. 

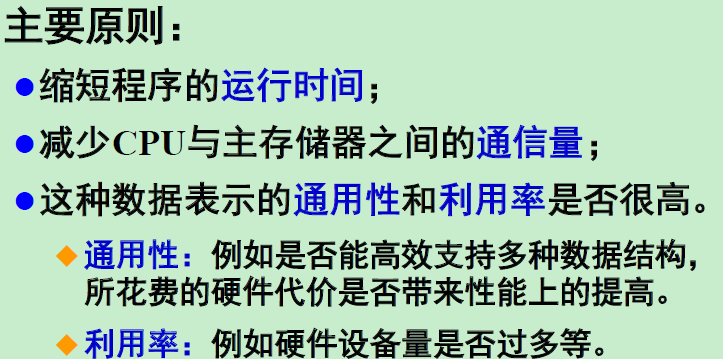
**当****面积小于****面积时，程序所占的空间减少**

* + 1. **数据描述符表示法**
       1. **描述数据块的属性，与数据分开存放**
       2. **数据描述符与标志符的区别：**
          1. **标志符与数据合存于一个存储单元中，用于描述单个数据的类型和属性（作用于一个数据）；**

**而描述符则和数据分开存放，主要用于描述成块数据的特征（作用于一组数据）。**

* + - * 1. 要访问数据集中的元素的时候，必须先访问描述符，这就至少要增加一级寻址
        2. 描述符可以看成程序的一部分，而不是数据的一部分，因为它是专门来描述要访问的数据的特性的。
      1. 

### 2.1.3 引入数据表示的原则



## 2.2 寻址方式

1. 寻址方式：寻找操作数及数据存放单元的方法。
2. 主要内容：寻址方式的设计思想和设计方法
3. 方法：分析各种寻址技术的优缺点，如何选择和确定寻址技术。

### 2.2.1 寻址方式分析

1. 编址方式
   1. 编制单位 = 访问单位
   2. 常用的编址单位： 字编址、字节编址、位编址、块编址
   3. 编制方式：分类编址、统一编址、隐含编址
2. 寻址方式
   1. 大多数计算机采用分类编址的方式
   2. 分类：
      1. 面向寄存器的寻址方式
      2. 面向主存的寻址方式
      3. 面向堆栈的寻址方式
3. 如何在指令中指明寻址方式？
   1. 在操作码中指明（隐式）
   2. 在地址码中指明（显式） 寻址灵活，操作码短；需要设置专门寻址访问位，指令长度相对变长。

### 2.2.2 间接寻址方式与变址寻址方式的比较

1. 差别：间址寻址方式：间接地址在主存储器中，没有偏移量

变址寻址方式：基地址在变址寄存器中，有偏移量

1. 主要优缺点比较：
   1. 实现的难易程度：间址寻址方式较容易
   2. 指令的执行速度：间址寻址方式慢
   3. 对数组运算的支持：变址寻址较好

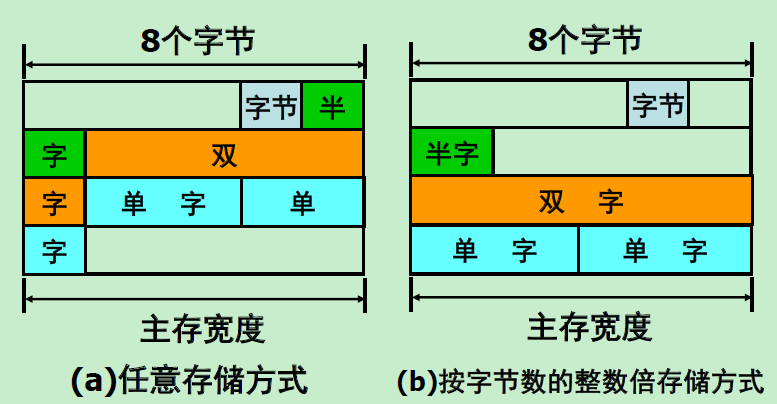
### 2.2.3 程序在主存中的定位技术

1. 程序需要定位的**主要原因：**

**程序的独立性；程序的模块化设计；数据结构在程序运行过程中，其大小往往是变化的；有些程序本身很大，大于分配给它的主存物理空间。**

1. 三种定位技术：
   1. **直接定位方式：**在程序**装入**主存储器**之前**，程序中的指令和数据的主存物理地址就已经确定了的称为直接定位方式**。**
   2. **静态定位：**在程序**装入**主存储器的**过程中**随即进行地址变换，确定指令和数据的主存物理地址的称为静态定位方式。
   3. **动态定位：**在程序**执行**过程中，当访问到相应的指令或数据时才进行地址变换，确定指令和数据的主存物理地址的称为动态定位方式。

### 2.2.4 信息分布



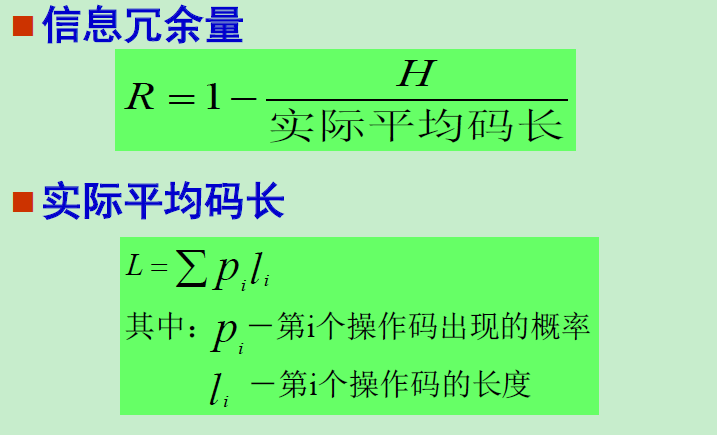
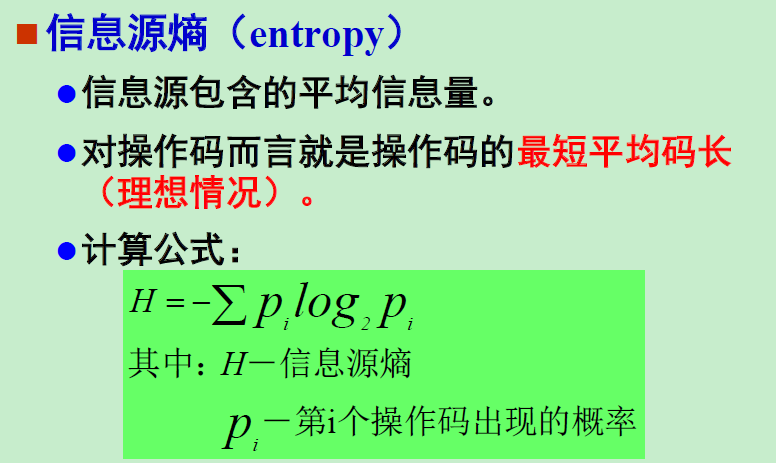
## 指令系统的设计和优化

1. 指令的组成：操作码 + 地址码
   1. 操作码： 操作种类、操作数描述
   2. 地址码： 地址、地址附加信息、寻址方式

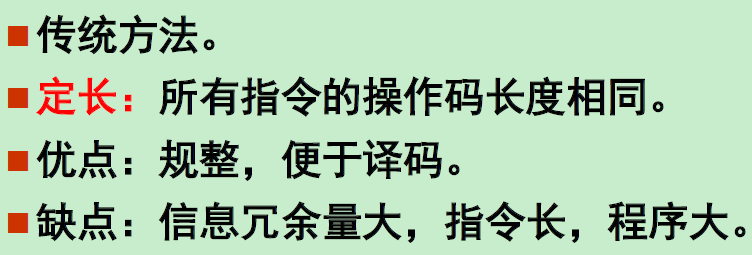
### 2.3.1指令操作码的优化

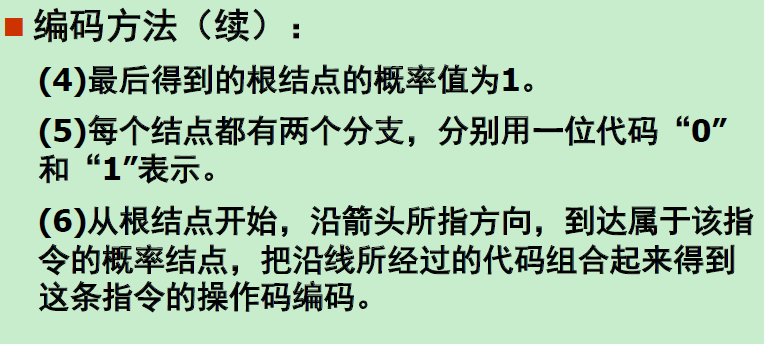
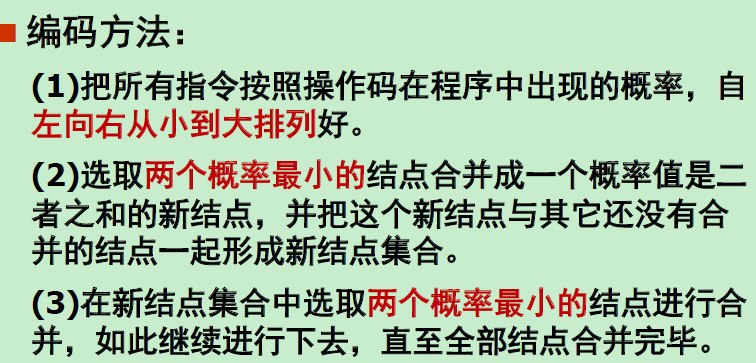
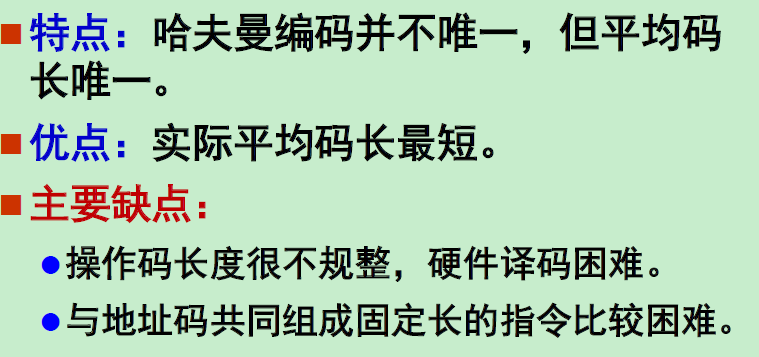
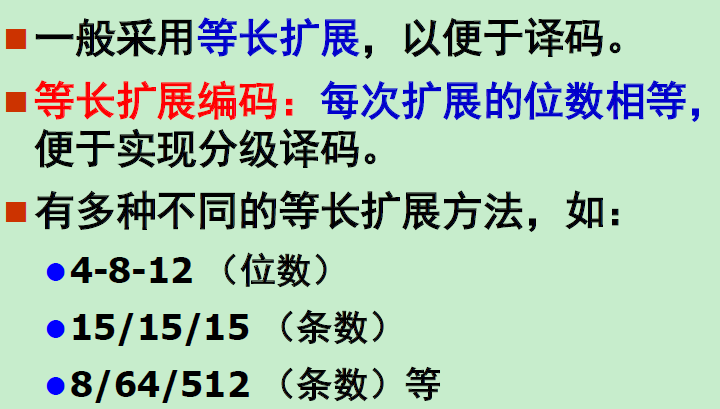
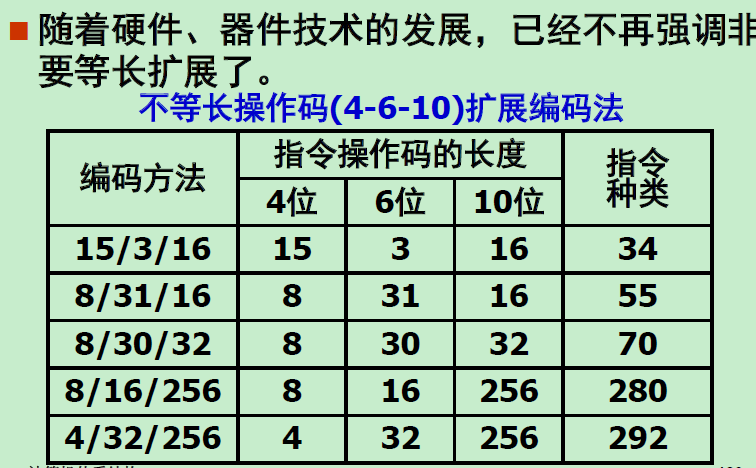
1. **操作码优化的目的：**节省程序的存储空间  
    指令格式尽量规整，便于译码
2. **操作码的三种编码方法（重点）**

**几个概念：**



* 1. **固定长度编码**



* 1. **Huffman编码法（最小概率合并法）**
     1. 
     2. 
  2. **扩展编码**
     1. 由固定长操作码与Huffman编码法相结合形成的一种编码方式，操作码长度被限定使用有限的几种码长，仍体现高概率指令用短码，低概率指令用长码的思想，使操作码的平均码长虽大于Huffman编码，但小于等长编码，是一种实际可用的优化编码方法。
     2.  

## 指令系统的发展和改进

1. 进一步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。**复杂指令系统计算机，简称CISC**
2. 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，提高指令的执行速度。**精简指令系统计算机，简称RISC**。

### CISC 和RISC

1. CISC (Complex Instruction Set Computer)
   1. 增强指令功能，设置功能复杂的指令,面向目标代码、高级语言和操作系统,用一条指令代替一串指令
   2. 设置更为复杂，功能更强的新指令以取代原先由软件子程序完成的功能，实现软件功能的硬化。
   3. **CISC指令系统存在的问题：**

(1)指令系统庞大。

(2)指令执行速度低。

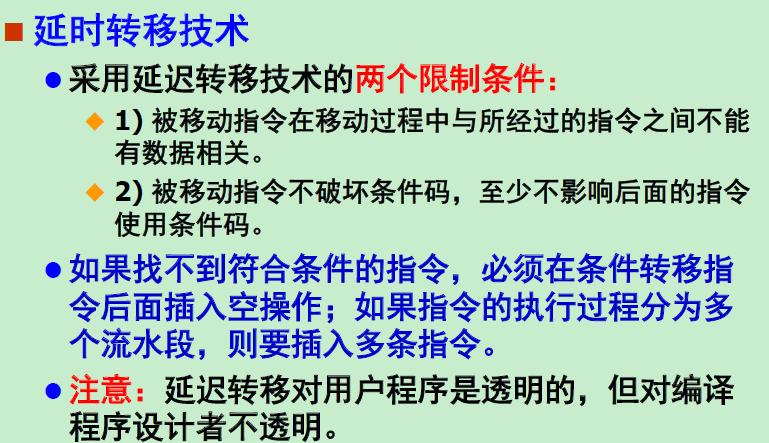
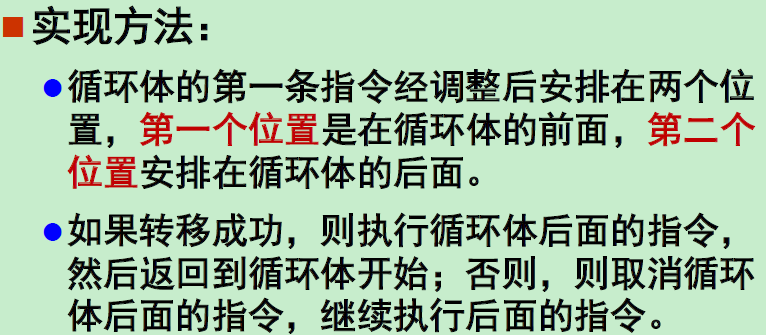
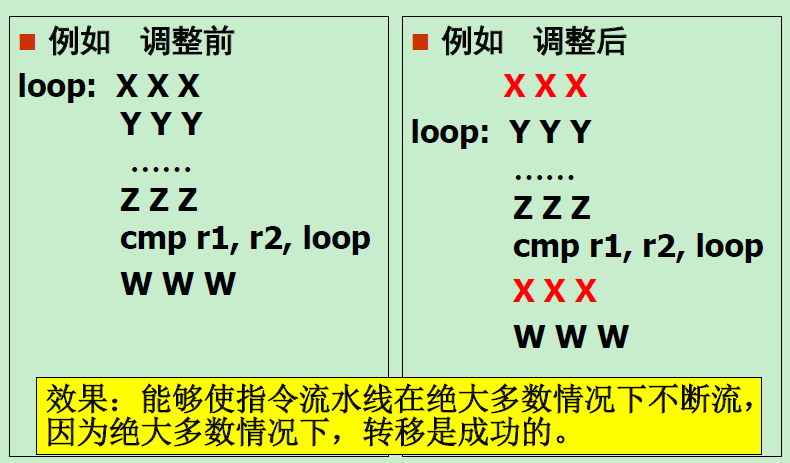
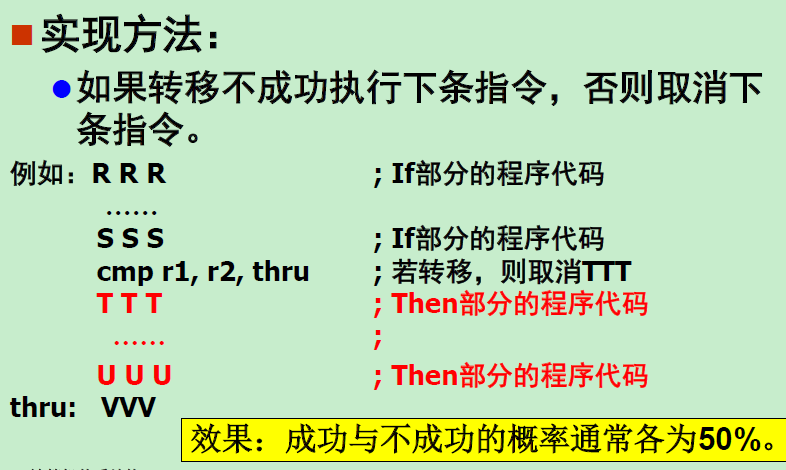
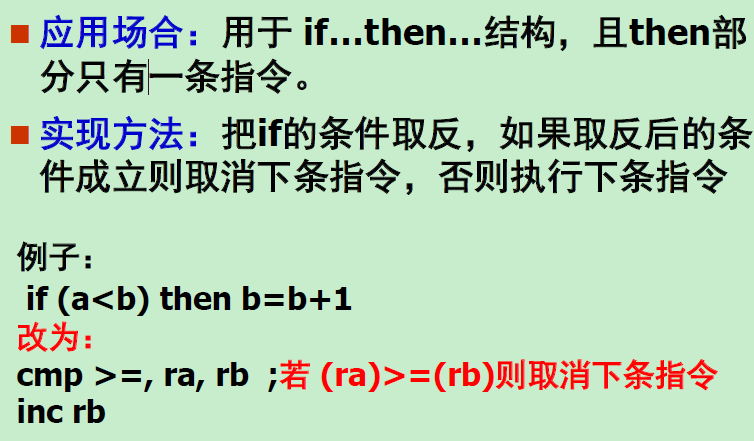
(3)编译程序本身太长、太复杂。

(4)各种指令使用频度都不会太高，且差别很大。

1. RISC (Reduced Instruction Set Computer)
   1. 只保留功能简单的指令,功能较复杂的指令用子程序来实现
   2. **RISC的定义与特点**

卡内基梅隆大学（Carnegie Mellon）论述RISC的特点：  
1、大多数指令在单周期内完成  
2、LOAD/STORE结构  
3、硬布线控制逻辑  
4、减少指令和寻址方式的种类  
5、固定的指令格式  
6、注重编译优化技术

* 1. RISC为使优化编译器便于生成优化代码，应具有：**三地址指令格式、较多的寄存器、对称的指令格式。**
  2. RISC的优点：
     1. 简化了指令系统设计，适合于VLI实现
     2. 提高了执行速度和效率
     3. 降低了成本，提高了可靠性
     4. 可以提供直接支持高级语言的能力，简化了编译程序的设计
  3. 缺点：
     1. 指令少，加重了汇编程序员的负担
     2. 对浮点运算和虚拟存储器的支持很轻大，但不够理想。
     3. 比CISC的编译程序难写

1. **RISC的关键技术**
   1. 重叠寄存器窗口技术
      1. **原因：RISC中，子程序比CISC中多, 因传送参数而访问存储器的信息量很大。**
      2. **设置一个数量比较大的寄存器堆，并把它划分成很多个窗口。在每个过程使用的几个窗口中有一个窗口是与前一个过程共用，还有个窗口是与下一个过程共用。**
      3. 每个过程使用一个有限数量的寄存器窗口，并让各个过程寄存器窗口部分重叠
      4. 设置一个数量比较大的寄存器堆，并把它划分成很多个窗口。在每个过程使用的几个窗口中有一个窗口是与前一个过程共用，还有个窗口是与下一个过程共用
   2. 流水线技术和**延迟转移技术**
      1. 延迟转移技术
         1. 定义：为了避免无条件转移指令和成功条件转移指令造成预取浪费，指令流水线断流等情况，在转移指令之后**插入一条不相关**的有效的指令，而转移指令被延迟执行，这种技术称为**延迟转移技术。**采用指令延迟转移技术时，指令序列的调整由编译器自动进行。
   3. 
2. **指令取消技术**
   1. 采用指令延时技术，在许多情况下找不到可以用来调整的指令，故有些RISC采用指令取消技术，分为三种情况：
   2. 向后转移（循环程序）
      1.  
   3. 向前转移(if-then)
      1. 
   4. 隐含转移技术
      1. 
3. **指令流调整技术**
   1. 目标：通过变量重新命名消除数据相关，提高流水线执行效率
4. **以硬件为主固件为辅**
   1. 固件的主要缺点是：**执行速度低**。主要优点是：**便于实现复杂指令，便于修改指令系统**。
   2. **以硬联逻辑为主来实现指令系统,对于复杂指令，也使用微程序技术实现.**

