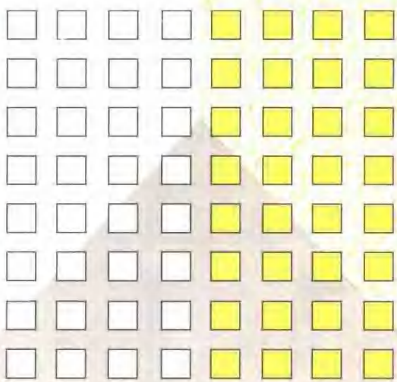
金地时代 π 中国计算机软件专业技术水平考试辅导书

程序设计

高级程序员级考试辅导书

刘 伟 编著

- 必考知识点
- 难点分析
- 例题详解
- 课后练习
- 模拟试题



科学出版社

中国计算机软件专业技术水平考试辅导书

程序设计

高级程序员级考试辅导书

刘 伟 编著

科学出版社

内 容 简 介

本书是中国计算机软件专业技术资格和水平考试中心按照 1999 年《程序设计》（高级程序员级）考试大纲组织编写的参考书。全书按照应试者应该掌握的有关程序设计内容的必考知识点进行了分析、归纳和总结，依据大纲对各章的常见问题、难点和例题进行了详细分析。并参照大纲要求安排了大量习题及解答，以便给应试者学习时提供辅导和启示。

本书既可以作为参加计算机程序设计（高级程序员级）水平考试的应试辅导材料，也可以作为从事计算机程序设计工作的技术人员的培训教材。

图书在版编目（CIP）数据

程序设计高级程序员级考试辅导书/刘伟编著. —北京：科学出版社，2000
（中国计算机软件专业技术水平考试辅导书）
ISBN 7-03-008028-9

Ⅰ.程… Ⅱ.刘… Ⅲ.程序设计-水平考试-教学参考资料 Ⅳ.TP311.1

中国版本图书馆 CIP 数据核字（2000）第 69244 号

科学出版社 出版
北京东黄城根北街 16 号
邮政编码：100717
北京双青印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2000 年 10 月第 一 版 开本：787×1092 1/16
2000 年 10 月第 一次印刷 印张：18 1/2
印数：1—5 000 字数：424 000

定价：26.00 元

（如有印装质量问题，我社负责调换〈环伟〉）

专家指点　轻松跨越

前　言

计算机软件专业技术资格和水平考试自 1991 年实行开始，至今已走过了 8 年的历程，共有近 40 万人参加考试，在国内外产生了较大的影响。为了满足我国计算机信息技术发展和企业对计算机软件人才的需求，1999 年新定考试大纲将资格和水平考试的范围和内容扩大为程序设计、软件工程、网络、多媒体和数据库五个方面，以适应社会对各种软件人才的需求。

编者受计算机软件专业技术资格和水平考试中心的委托，编写这套适用于程序设计方面考试要求的辅导教材，共分为初级程序员级、程序员级和高级程序员级三册。目的在于帮助参加计算机软件专业技术资格和水平考试的广大考生复习，使其掌握考试大纲的基本要求，适应考试的形式和试题类型。

本书属高级程序员级辅导教材，编写的依据为 99 年新定考试大纲，对高级程序员级考试应必备的知识和应具备的能力作了详尽的讲解。全书共分系统结构、数据结构、程序语言基础、操作系统、数据库、多媒体、软件工程、计算机网络和程序设计九个章节，每一章节均含有必考知识点、疑难点分析、例题详析，其中部分例题选自于以往的考题，并结合新大纲作了相应修改。书中还包含有大量的习题和模拟试题，选题内容、试题类型尽量与计算机软件专业技术资格和水平考试保持一致，符合大纲要求，难度适中。希望能对参加考试的广大读者起到帮助和指导作用。

全书由刘伟同志主编，第二、三、四、八章由刘伟同志编写，第五、六章由司志刚同志编写，第九章由鹤荣育同志编写，第一章由戴青同志编写，第七章由王秋生同志编写，。彭利凯、刘可同志在本书的编写过程中作了大量工作，在此向他们表示感谢。由于本书涉及内容广泛、编写时间紧促等原因，书中难免存在错误和不妥之处，衷心希望广大读者给予指正和提出宝贵的修改意见。

编者

1999. 12

目　录

第 1 章　计算机系统结构

1.1　必考知识点

1.1.1　计算机系统结构概述

一、计算机系统结构、组成和实现

计算机系统结构这个名词从 70 年代开始广泛使用，但至今没有一个统一、确切的定义。它与软、硬件界面动态变化和器件集成度的迅速发展有关。

以多级层次结构分析，从不同级的使用者来看，其系统结构的含义、概念的着重点有所不同。

1．Amdahl 观点

Amdahl 等人于 1964 年提出，由程序设计者所看到的一个计算机系统的属性，即概念性结构和功能特性。本来是存在的事物或属性，但从某种角度看，好象不存在的概念，称为透明性概念。

系统结构指的就是对每个级的界面上、下的功能分配和对界面的定义，各级都有其系统结构。

它研究的是软、硬件功能分配以及对机器级界面的确定。这些属性包括：指令系统、数据结构、寻址方式、中断机构、寄存器定义和 I/O 结构（机器级的）。同时还包括机器工作状态的定义及切换对信息保护的支持。

2．计算机组成

定义：指的是系统结构的逻辑实现，包括机器内的数据流和控制流的组成及逻辑设计。

它的设计内容包括按希望所达到的性能价格比，如何最佳、最合理地把设备和部件组成计算机，以实现所定的系统结构。它的进展取决于如何提高速度，提高操作的并行度、重迭度及功能分散和专用功能部件的设置。

3．计算机实现

定义：指的是计算机组成的物理实现。

一种组成可以采用多种物理实现，但计算机实现并不是只能被动地实现，它和计算机组成是可以折中、权衡的。

为达到所要求的速度，简单的组成都是高速的实现，复杂的组成却是一般的速度。

计算机的组成和实现密切相关，二者往往难以明确区分，因而提出把二者统称为计算机实现，即计算机实现包括计算机系统结构的逻辑实现和物理实现。

优化：一般指在当时器件技术条件下，如何在价格不增或少增的前提下，尽可能

地提高速度。

二、计算机系统分类

共有 3 种不同的分类方法

1．Flyrn 分类法（1966 年提出如下定义）

指令流：机器执行的指令序列。

数据流：由指令流　用的数据序列，包括输入数据和中间结果。

多倍性：在系统最受限制的元件上同时处于同一执行阶段的指令或数据的最大可能个数。

根据指令流和数据流的不同组织方式，把计算机系统结构分为下以 4 类：

SISD：单指令流单数据流（传统顺序处理）。

SIMD：单指令流多数据流（降列式并行处理）。

MISD：多指令流单数据流（流水结构处理）。

MIMD：多指令流多数据流（多处理）。

2．冯氏分类法

冯泽云于 1972 年用最大并行度对计算机系统结构进行分类。

字串位串：WSBS（纯串行计算机）。

字并位串：WPBS　n>1，m=1（传统并行单处理机）。

字串位并：WSBP　n=1，m>1。

字并位并：WPBP　n>1，m>1。

3．Handler 分类法

Handler 于 1977 年根据并行度和流水线提出了分类法。这种分类法把计算机的硬件结构分成 3 个层次，并充分考虑它们的可并行——流水处理程度。这 3 个层次是：

（1）程序控制部件（PCU）的个数 k。

（2）算术逻辑部件（ALU）或处理部件（PE）的个数 d。

（3）每个算术逻辑部件包含基本逻辑线路（ELC）的套数 w。

可以把一个计算机系统的结构用公式表示为：

T（系统型号）＝（k,d,w）

为了进一步揭示流水线的特殊性，一个计算机系统的结构可用如下公式表示：

T（系统型号）＝（k×k',d×d',w×w'）

其中：k'表示宏流水线中程序控制部件的数量，d'表示指令流水线中算术逻辑部件的个数，w'表示操作流水线中基本逻辑线路的套数。

1.1.2　存储器系统

存储器系统主要用于保存数据和程序，没有任何一种单一的技术能够完全优化地满足计算机存储系统的要求：高速存取、大容量和低成本。因为在存储器技术中，存在以下的制约关系：

- 存储器读写速率越高，每位的成本也越高。
- 存储器容量越大，每位的成本也越高。

- 存储器容量越大，读写速率越低。

解决这一难点的方法就是采用多级存储体系结构(memory hierarchy)。

一、存储器体系

1．存储器系统的特征

（1）位置：存储器系统主要包括内部寄存器和外部存储器。内部存储器是可以被处理器直接存取的存储器，又称为主存储器(main memory)；外部存储器(external memory)需要通过 I/O 模块与处理器交换数据，又称为辅助存储器(auxiliary memory)。除了正在执行的程序和处理的数据存放在主存储器外，其他所有的信息都保存在辅助存储器中，当需要时再从辅存中调入主存。

为了提高性能，弥补 CPU 处理速度和主存存取速度之间的差异，还设置了 cache(高速缓冲存储器)。其容量小，但速度极快，位于 CPU 和主存储器之间，用于存放 CPU 正在执行的程序段和所需数据。

（2）存储器单元：衡量主存储器容量大小的单位通常是字节或字(word)(1 字＝8，16，32 位)，而外存储器的容量一般用字节表示。

（3）存取方式(method of accessing)：常用的存取方式有顺序存取、直接存取、随机存取和相联存取 4 种。

（4）性能：存储器系统的性能主要由存取时间、存储器带宽、存储器周期和数据传输率等来衡量。

（5）物理介质：常用的存储介质有半导体存储器(semiconductor memory)、磁介质存储器(magnetic surface memory)、光存储器 3 种。

（6）物理特性。

- 易失性存储器(volatile memory)：当断电时，保存在易失性存储器里的内容就会丢失。
- 非易失性存储器(nonvolatile memory)：保存在非易失性存储器里的内容不会因为断电而丢失。除非对其进行写入操作，否则数据不会被破坏。
- 不可改写存储器(nonerasable memory)：数据一旦写入，就无法更改。

2．多级存储器体系

计算机采用多级存储器体系的目的就在于能够获得尽可能高的存取速率，同时保持较低的成本。

从存储器体系顶端的寄存器和 cache 到底部的磁带存储器，其价格逐渐降低；而容量和访问时间则逐渐增加。通过合理地设计存储器体系结构，使得经常访问的数据保存在高速存储器中，而较少访问的数据保存在低速存储器中，即从高到低，数据的访问频率逐渐减少。这样就可能组成一个访问速度尽可能快，容量尽可能大而价格也较便宜的存储系统。

二、高速缓冲存储器 cache

设置 cache(高速缓冲存储器)的目的在于提高 CPU 数据输入输出的速率，突破所谓的“冯·诺依曼瓶颈”，即 CPU 与存储系统间的数据传送带宽限制。在计算机的存储系统体系中，cache 是访问速度最快的层次。

1．cache 存储器的原理

使用 cache 改善系统性能的依据是程序的局部性原理，即程序的地址访问流有很强的时序相关性，未来的访问模式与最近已发生的访问模式相似。在任一给定的时间间隔内，对不同的地址区域其访问概率是不同的，有的区域高，有的区域低。而另一种可能则是访问概率随着离当前执行指令的远近而变化，离当前执行指令越近，其概率也越高。

依据局部性原理，把主存储器中访问概率高的内容存放在 cache 中，当 CPU 需要读取数据时，就首先在 cache 中查找是否有所需内容。如果有，则直接从 cache 中读取；若没有，则再从主存中读取该数据，然后同时送往 CPU 和 cache。如果 CPU 需要访问的内容大多能在 cache 中找到(称为访问命中，hit)，则可以大大提高系统性能。

2．cache 存储器的结构

cache 存储器的组织结构与主存储器不一样，它以行(line)作为基本单元。每一行又分为标志项和数据域两部分。数据域中存放着若干项数据，而标志项则是这一块数据的地址标识。

3．cache 存储器的映射机制

当 CPU 发出访存请求后，存储器地址先被送到 cache 控制器以确定所需数据是否已在 cache 中，若命中(hit)，则直接对 cache 进行访问。这个过程称为 cache 的地址映射(mapping)。为了适应 cache 存储器的极高存取速率，映射也必须在极短的时间内完成。常见的映射方法有直接映射、相联映射和组相联映射.

4．cache 存储器的淘汰算法

当 cache 存储器产生了一次访问未命中之后，相应的数据应同时读入 CPU 和 cache。但是当 cache 已存满数据后，新数据就必须淘汰 cache 中的某些旧数据。最常用的淘汰算法有随机淘汰法、先进先出法(FIFO)和近期最少使用的淘汰法(LRU，least recently used)。

5．cache 存储器的写操作

因为需要保证缓存在 cache 中的数据与主存中的内容一致，相对读操作而言，cache 的写操作就显得比较复杂。常用的有以下 3 种方法：

（1）写直达(write through)：当要写 cache 时，数据同时写回主存储器。

（2）写回(write back)：CPU 修改 cache 的某一行后，相应的数据并不立即写入主存储器单元。而是当该行被从 cache 中淘汰时，才把数据写回到主存储器中。

（3）标记法：对 cache 中的每一个数据设置一个有效位。当数据进入 cache 后，有效位置为 1；而当 CPU 要对该数据进行修改时，数据只需写入主存储器并同时将该有效垃清 0。当要从 cache 中读取数据时，需要测试其有效位，若为 1 则直接从 cache 中取数；否则从主存中取数。

三、虚拟存储器

虚拟存储器的管理原则与 cache 基本相似：

- 存储系统由小容量的高速存储器和大容量的低速存储器组成。
- 把要经常访问的数据驻留在高速存储器中。
- 一旦这些数据访问频率下降则把它们送回低速存储器中。
- 设计有效的管理算法，以使系统访问速率接近高速存储器，而容量接近大容量的

低速存储器，有较高的性能价格比。

1．虚拟存储器概述

在使用虚拟存储器体系的系统中，由程序(CPU)使用的访存地址称为虚拟地址，程序（CPU）直接访问的存储空间称为虚拟地址空间；而主存储器的地址则称为物理地址。通常虚拟地址空间远远大于主存储器容量。

CPU 发出访存请求，其提供的地址为虚拟地址，需要经过地址映射以确定所需内容是否已在主存储器中。若已调入主存储器，则把虚拟地址映射为物理地址来访问主存；否则产生页故障，系统把相应的数据从辅存调入主存，同时再送往 CPU。若主存内容已满，则还需淘汰掉某些数据。

虚拟存储系统在产生页故障后通常不再把 CPU 的控制权移交给另一作业，而是让 CPU 等待从辅存调入新的页面以避免竞争处理器。

2．虚拟存储器的地址映射机制

地址映射的基本功能是实现从虚拟地址到物理地址的转换。页面管理系统把主存和虚存空间分割成固定大小的页面，分别称为物理页面和虚拟页面(逻辑页面)。虚拟地址分成两部分：虚拟页号 VP 和页内地址 VD。同样物理地址也分成两部分：物理页号 PP 以及页内地址 PD。通常虚拟地址空间的页面与主存储器空间的页面大小相等，但虚拟地址空间远大于主存空间。

3．分段存储系统

页面是虚拟存储器系统中使用的最小单元，因此为了访问某一存储单元(字，word)需要把包含该单元的整个页面全部调入主存。通常页面的划分由虚存管理系统负责，页面与程序逻辑结构无关。这样一页中的内容往往逻辑上无关，调入该页常会引入许多无用的内容，从而降低系统性能。

分段存储器以段作为基本单元，段是相关程序和数据的集合，形成一个子程序，所以分段存储器系统能够反映程序内部的逻辑关系。段与页不同，段长是不固定的，由程序员设计，而且分页地址空间是一维空间，所有地址都是连续的；分段存储器则是二维地址空间，由段号+段内位移量组成其地址，段内地址连续而段与段之间不连续。可变段长的分段存储管理比固定页面大小的分页存储管理要复杂得多。

可以把分页和分段技术结合起来组建虚拟存储器系统。具体做法是：用分段表示程序及数据的逻辑结构，然后把段再分成固定大小的页面。

4．页面调度算法

当主存已满而又需从辅存户调入新页面时，就要从主存中淘汰一个页面。页面淘汰算法对虚存系统性能影响很大，好的算法可以减少页故障率。当淘汰算法选择不当时，不仅可能增加页故障率，而且会造成系统在重负荷时产生颠簸(thrashing)——即某些页面频繁地在主存和辅存间调进调出。

常用的页面淘汰算法有 FIFO 算法和 LRU 算法。

1.1.3 流水线技术

流水线技术是通过并行硬件来提高系统性能的常用方法，主要用于巨型机。

一、流水线技术的基本原理

流水线技术其实是一种任务的分解技术。把一件任务分解为若干顺序执行的子任务，不同的子任务由不同的执行机构负责执行，而这些机构可同时并行工作。在任一时刻，任一任务只占用其中一个执行机构，这样就可以实现多个任务的重叠执行，以提高工作效率。

二、计算机流水线技术概述

1．指令流水线

计算机中一条指令的执行需要若干步，通常采用流水线技术来实现指令的执行，以提高CPU性能。典型的指令执行共分7个阶段：

（1）计算指令地址，修改程序计数器PC。

（2）取指，即从存储器中取出指令。

（3）指令译码。

（4）计算操作数地址。

（5）取操作数。

（6）执行指令。

（7）保存结果。

流水线技术对性能的提高程度取决于其执行顺序中最慢的一步。流水线设计的基本思想是任何一个可以分解的任务都可以用流水线来做，可以设置多个处理机构，分别执行相应的子任务。为了提高流水线性能，有些处理时间长的步还需分解成更小的步，使流水线上所有步的处理时间相同。

2．运算操作流水线

以上讨论了指令级流水线技术的概况，解释了流水线技术是怎样提高计算机性能的。其实，计算机在执行各种运算操作时，也可以应用流水线技术来提高运算速度。

三、影响流水线性能的主要因素

流水线的关键之处在于“重叠执行”。为了得到高的性能表现，流水线应该满负荷工作，即各个阶段都要同时并行地工作。但是在实际情况中，流水线各个阶段可能会相互影响，阻塞流水线，使其性能下降。阻塞主要由两种情形引起：执行转移指令和共享资源冲突。

1．转移指令的影响

通常在顺序执行指令的情况下，当CPU取一条指令时，流水线的地址计算部件可以独立地把当前PC值加上当前指令长度来计算下一条指令的地址，从而可以并行地工作，但是当流水线执行一条转移指令时，就会引起流水线的阻塞。因为在该转移指令完成之前，流水线不能确定出下一条指令的地址。

所以为了保证指令的正确执行，必须把取指段和指令地址计算段互锁。在取出转移指令后，立即锁住指令地址计算段，直到转移指令执行完成。互锁阶段流水线处于等待状态，不能满负荷工作，从而性能下降。

2．共享资源访问冲突

当多条指令以流水线方式重叠执行时，由于可能会引起对共享的寄存器／存储器

资源访问次序的变化，因此将导致冲突——这种情况又称为数据相关。

（1）访问次序的变化

为了简单起见，设有顺序执行的两条指令 I1 和 I2。I1 的运算结果存放在存储单元 M1 中，I2 则需要从 M1 中取出 I1 的运算结果作为源操作数。其正常执行顺序为 M：I1 写—I2 读。然而，以流水线方式执行 I1 和 I2，如果不对流水线做某些控制，当 I2 取操作数时，I1 还没有保存结果。所以实际上对 M1 的访问顺序变成：I2 读—I1 写。

（2）访问冲突

两个操作访问共享资源的情况有 4 种：读—读、读—写、写—读和写—写。除了读—读组合不会因为使用流水线技术而产生冲突外(因为无论谁先读，其结果都一样)，其余 3 种都会产生冲突，因而产生错误。

为了避免冲突，需要把相互有关的指令进行阻塞，这样就会引起流水线效率的下降。一般说来，指令流水线级数越多，越容易导致数据相关，阻塞流水线。

四、流水线计算机的存储器结构

流水线技术使处理器能够并行地重叠执行若干条指令，从而提高了指令的执行速度。但这又要求其存储系统能够及时地响应流水线的访存需求，而且要求存储系统能够支持并发访问而不产生冲突。指令流水线技术其实是把处理瓶颈从 CPU 子系统转移给存储子系统。

在存储系统中，也需要使用流水线技术以匹配采用了流水线技术的 CPU 系统的处理能力。

流水线计算机的存储器分成若干(4 个)独立存储体，以支持流水线方式并发访问。流水线计算机中也使用了 cache，通常分为指令 cache 和数据 cache，各自用于存放指令和操作数。在通常情况下该结构工作良好，但是如果系统允许程序在执行过程中修改其自身，就会大大地影响其性能。

五、流水线控制

为了尽可能地使流水线满负荷工作，在流入线入口处设置一个控制器，用以控制流水线入口处的各种操作。控制器在入口处不断接收新操作进入流水线，使流水线一直维持在最大的吞吐率，同时确保在流水线各段不会产生冲突。

六、流水线的中断处理

流水线计算机处理中断的方法通常有两种。

（1）不精确断点法。当发生中断后，计算机并不立即响应该中断，而是先禁止指令再进入流水线，然后待已在流水线中的所有指令执行完毕，才响应该中断。

（2）精确断点法。当有中断请求时，立即中止当前流水线的作业，去响应中断。这种方法的硬件电路比前一种复杂得多，但对中断响应的实时性却比前一种方法好。

1.1.4 输出输入系统

一、输出输入控制器

用以控制一个或多个外设与 CPU、主存储器进行数据交换。它通过系统接口(内部

接口)与主机 (CPU、主存储器) 进行交互，通过设备接口(外部接口)与各种外设打交道。

（1）控制与定时(control & timing)：I/O 系统必须对通信进行控制和定时，以协调外设和内部资源之间的通信流量，因为内部资源（如主存、系统总线等）被…系列的活动包括数据的 I/O 所共享。同时 I/O 模块和 CPU 之间的传递还需进行总线仲裁，以解决多个设备争用总线的问题。

（2）与 CPU 通信(CPU Communication)

命令：I/O 系统必须能够接收和翻译由 CPU 发来的控制命令。

数据：I/O 系统还需通过数据线与 CPU 交换数据。

状态：I/O 系统能够向 CPU 报告外设的状态，以便 CPU 根据该状态信息来完成数据交换。

地址：每…个外设都具有自己的地址，I/O 系统应该能够识别它所控制下的每一个外设的地址。

（3）与设备通信(devicecommunication)：I/O 系统与外设之间交换控制命令、状态信息以及数据。

（4）数据通信缓冲(databufferi，8)：I/O 系统需要用数据缓冲寄存器来暂存数据，以此来适应通信双方速率的不匹配，从而提高主机利用率。

（5）错误检测机制(errordetection)：外设和主机进行数据交换的过程中可能会产生错误，通常使用奇偶校验码、CRC 循环冗余校验码进行检测和纠正。

在主机和外设之间设置 I/O 系统，其目的在于隐藏各种外设的具体细节信息(如数据格式、时序等)，给 CPU 提供一个统一、简便的控制界面，用以完成数据交换，减轻 CPU 的 I/O 负担。

2．输入输出控制器基本结构

- 数据寄存器：用于缓冲从 CPU 来的或是从外设来的数据。
- 状态寄存器：记录设备当前状态，CPU 通过查询该寄存器可知道设备情况，以决定操作顺序。
- 控制寄存器：CPU 通过对该控制器进行访问来决定该外设执行的操作和控制（控制外设以及 I/O 系统）。
- 控制电路：I/O 系统的核心。负责控制 I/O 系统的运行，同时要完成外部设备地址的识别，以确定 I/O 操作将发生在哪个外设上。还可能需要中断控制逻辑，直接存储访问（DMA，direct memory access）控制逻辑，以支持中断方式或 DMA 方式通信。
- 外设接口控制：接受 CPU 发来的控制命令，控制外设与 I/O 之间的数据交换等。

二、输出输入的工作方式

输入输出系统主要有 3 种方式与主机交换数据。

（1）程序控制方式(programmed I/O)：输入输出完全由 CPU 控制，在整个 I/O 过程中 CPU 必须等待而不能进行其他工作，无法充分发挥高速的处理能力。

（2）程序中断方式(interrupt-driven I/O)：I/O 模块以中断的方式通知 CPU 开始传递数据，无需 CPU 主动查询和等待外设，从而把 CPU 解放出来做其他工作，以提高 CPU