高等学校 计算机教材

http://www.phei.com.cn

# Windows 98 IO结构分析教程

郭嵩山 等编著


電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

高等学校计算机教材

# Windows 98 IO 结构分析教程

郭嵩山　等编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

## 内 容 简 介

本书是由中山大学信息科学与技术学院计算机科学系郭嵩山教授编写的“操作系统结构分析”课程的教材。

该教材的特点是采用以模块主程序为主线，以数据结构为中心的系统软件分析方法，对 Windows 98 设备驱动模块（IO.SYS）进行深入的剖析，不仅使读者全面了解操作系统实现 I/O 的过程。同时，也学会并掌握了用汇编程序实现软件分析的方法。

本书可作为高等院校计算机有关专业的教材或教学参考书，也可作为在职培训教材及从事 PC 开发的工程技术人员常备的技术参考资料。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

**图书在版编目（CIP）数据**

Windows 98 IO 结构分析教程/郭嵩山等编著. —北京：电子工业出版社，2004.1
高等学校计算机教材
ISBN 7-5053-9510-6

Ⅰ.W… Ⅱ.郭… Ⅲ.窗口软件，Windows 98—结构分析—高等学校—教材 Ⅳ.TP.316.7

中国版本图书馆 CIP 数据核字（2003）第 121310 号

责任编辑：龚立董
印　　刷：北京大中印刷厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：17　字数：435 千字
印　　次：2004 年 1 月第 1 次印刷
印　　数：4 000 册　　定价：25.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

# 前　言

Windows 是家喻户晓的个人计算机操作系统（以下简称 OS）。Windows 的操作和使用是非常简便的，但其 OS 规模却是十分庞大，结构也十分复杂。如何深入分析、研究 OS 的结构是学习和理解 OS 如何实现资源管理的重要一环，尤其是对 OS 底层如何实现 I/O 的基本管理；对了解整个 OS 的运作；对相关的通信与接口；对计算机安全和病毒的防治等等，都有着十分普遍的意义。我们通过深入剖析 Windows OS IO 的结构，学习 OS 是如何设计和实现的，这些对于从事计算机应用开发人员，将会受益匪浅。

如何分析操作系统，笔者总结多年来剖析和研究 OS 和系统软件中所摸索到的经验后认为：采用以模块主程序为主线，以数据结构为中心的系统软件分析方法，是一种好的分析方法。强调以主程序为主线，可使读者快捷、省力地了解整个模块的总体结构，再逐层去剖析各个分支模块的结构。强调以数据结构为中心，是因为一个系统程序的设计，在其算法确定之后，关键就是数据结构的设计。在剖析 OS 时，往往遇到的难题是对表格（线性表、链表）、缓冲区（暂存区）、静态和动态堆栈等数据结构未能弄清；对各模块所使用的数据单元的意义和取值未能了解；从而大大降低了分析和阅读系统程序清单的速度，甚至使分析工作无法进行下去。

本书以 Windows 98 作为主要分析对象，对其设备驱动模块（IO.SYS）进行了全面、深入的剖析。同时，对于涉及 I/O 设备中 ROM-BIOS 的部分中断处理程序也进行了分析，以便于读者全面了解 OS 实现 I/O 的过程。为了方便读者学习、分析和研究，本书第 2~4 章的最后 1~2 节，都列出了该部分程序的详细注释清单。

本书力求从有利于教与学的角度，对一般人感到难度很大的操作系统内部结构和实现原理通过深入浅出的系统论述，让读者既能建立整体的概念，又能逐步深入，一层一层地剖析。为了帮助读者能结合原理读懂源程序清单，我们对于程序清单中的注释尽量详细，力求深入到每一条指令，对于重要的程序段，我们都在正文中给出了执行流程图。同时，本书每章后均附有习题和思考题，以帮助读者更好地理解和掌握。

十多年来，我们在操作系统结构分析教学中，采用学习老版本，研究新版本的教学方法。虽然操作系统版本更新很快，但每一次更新往往仅在原基础上对某些部分做了较大修改，而其他部分变化不大。这样，我们通过在课堂上精讲，课后让同学们在各自 PC 机上阅读、研究当前使用的 OS 版本，对于提高同学们的分析解决问题能力，有较大帮助。同时，在大作业中要求同学们，重新用汇编程序完成实现 OS 的某些功能的核心程序模块及一些工具软件，收到了较好的效果。

本书可作为高等院校计算机有关专业的教材或教学参考书，也可作为在职培训教材及从事 PC 开发的工程技术人员常备的技术参考资料。

中山大学计算机科学系软件和应用专业的同学参加了对 DOS、Windows 9X 不同模块的

剖析工作，他们是93级的郑嘉宁、张东宝、余晓仪，94级的陈悦、洪昕等。在老师指导下，98级的卢环震、卢良楷、蔡树彬、麦振刚同学（按参加编写的章节顺序排名）在《BIO结构分析教程》（见参考资料1）基础上进行修改、补充和版本更新；最后，由郭嵩山老师进行书稿的统编、修改、补充及审定，从而完成了本书的编写及两年的教学试用工作。这也是前面所说的学习老版本，研究新版本的教学方法的成果。

由于作者水平所限，本书会有不少缺点和错误，恳请读者批评指正。

编著者

2003年12月



此为试读，需要完整PDF请访问：www.ertongbook.com

# 目　录

# 第 0 章　操作系统结构概述

早期的个人计算机操作系统，由于系统规模较小，逻辑关系较简单。所以，设计者往往只注重功能设计和效率，而忽视了结构的设计。在个人计算机的发展史中，由于操作系统设计不当，引起错误而造成惨重的损失，使人们记忆犹新。因此，人们在总结经验的基础上，认识到系统结构直接影响到系统的性能，从而越来越重视结构的设计，并逐步采用结构化程序设计方法来设计操作系统，使之成为结构清晰、易读易懂、适应性强、可靠性高、易于修改、易于证明其正确性的系统。

采用结构化程序设计方法来设计操作系统，可以将操作系统看成一个整体模块，它由若干个模块按一定的结构方式组成。到目前为止，操作系统的结构大体上可以分为 3 类：即模块组合结构、层次结构和管程结构。

## 第 1 节　模块组合结构

模块组合结构也称无序模块结构、模块接口结构。所谓模块组合结构，就是将一个大系统分成若干个相对独立的模块，这些模块可以独立编制，为使每一模块不太复杂，又可把大模块划分成更小的模块，形成“积木式”的结构方式，并把这些模块按规定的接口（如转子、调用或借助通信区、工作单元等）连接起来。

模块组合结构的特点是：

① 模块是以功能而不是以程序或数据的特点来划分的；

② 数据作为全程量使用；

③ 不同模块间可以不加限制地互相调用或转移，模块间信息传递方式可以随意约定。

模块组合结构的优点是结构紧密，接口简单，系统效率高；缺点是模块独立性差，结构不清晰，不易读，不易理解，修改也不方便，往往动一处而牵动全体。此外，为了保证数据的完整性，通常采用全局封中断的方式，从而限制了系统的并发性。

早期的 UNIX 版本采用模块组合结构，UNIX 源程序划分为 44 个文件，这 44 个文件可看成模块，其中 14 个是全局变量说明，28 个是 C 语言文件，2 个是汇编语言文件。图 0.1 示出了 UNIX 操作系统，其中 7 个 C 语言文件间的依赖关系，其文件之间的调用是随意的，没有受一定的规律限制。

## 第 2 节　层 次 结 构

所谓层次结构，就是将整个操作系统分解成若干个基本模块，并按照一定的原则，将这些模块排列成若干层，各层之间只有单向依赖关系，也即低层为高层服务，高层依赖于低层，各层之间不能构成循环。层次结构避免了模块组合结构的缺点，减少了模块间的相互依赖关系，消除了循环调用现象。

层次结构的优点是：

① 将整体问题局部化。由于层次结构是把一个大型系统分解成若干个具有单向依赖关系的层次。因此，将对整个系统的了解，进而分解成对各层模块的局部了解；将保证整个系统的正确性，分解成保证各层模块的正确性。

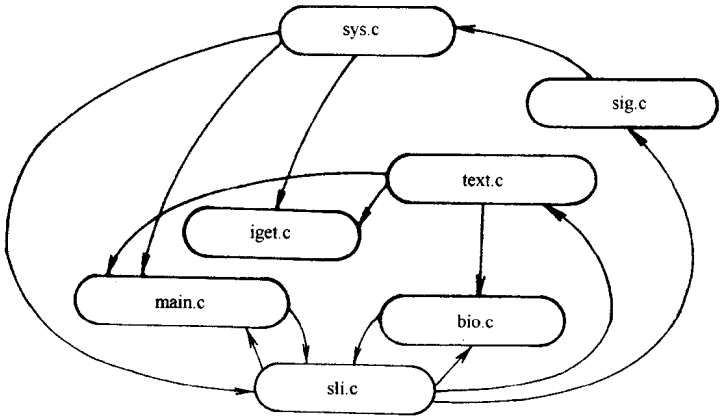



图 0.1 UNIX 部分文件的调用关系

② 层次结构的操作系统上层模块对下层模块的调用，通常设计成从统一的入口进入。也就是说，每层中各模块以统一的接口提供给上层模块调用。这样，就大大减少了接口量，从而使各层次间的调用更加清晰和规范。

③ 对于以进程作为层次中模块基本单位的层次结构（称为进程分层结构），能较好地体现操作系统的并发特征，能动态地描述系统的执行过程。

④ 各层次独立性强，灵活性高，易于维护、修改和移植。

⑤ 系统结构清晰，易于阅读和理解。

但层次结构也存在一些缺点。例如，在进程分层结构中，每个进程都要建立一个进程控制块，从而增加了系统的开销。此外，进程分层结构是由核心来统一管理控制转移、标志保留和层间的信息传递，故信息传递效率比模块组合结构低。而且，由核心统一管理，调度负荷重。

在层次结构的操作系统中，如果不仅层间是单向依赖关系，同一层间各模块也是相互独立、单向依赖和不构成循环。那么，这种层次结构称为全序结构；如果某些层次内部的模块存在着循环调用的关系。那么，这种层次结构称为半序结构。

Windows 98、MS-DOS 是典型的层次结构的操作系统，图 0-2 示出了 Windows 98、MS-DOS OS 的层次结构。

层次结构分层的原则是以单向依赖关系为总的原则，其他的原则为：

① 将一些直接依赖于硬件的模块（如中断处理、设备驱动等）放在最低层，这样改造后的操作系统，硬件特性消失。移植时，只需修改最低层就行。例如 Windows 98 的 IO.SYS 模块、MS-DOS 的常驻 IBMBIO.COM 模块等。

② 将操作系统中指挥整个系统运转的指挥控制中心，如处理机调度、进程控制、通信机构等模块放在最低层。

③ 将对操作系统的命令解释模块放在最高层，当要求改变操作系统的操作方式时，只

要改变最高层，例如 Windows 98 的图形界面模块、MS-DOS 的 COMMAND.COM 模块等。

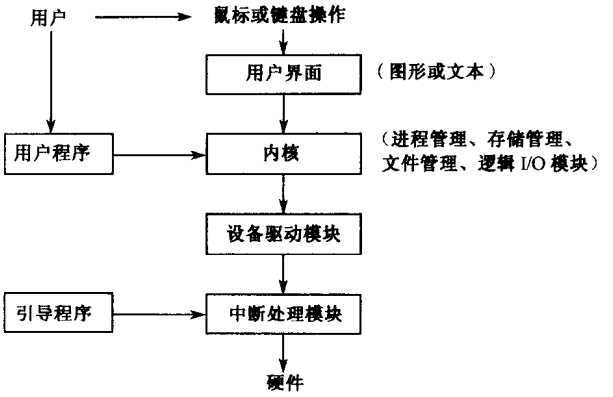



图 0.2　Windows 98、MS-DOS 的层次结构框图

④　将接受控制的模块放在较高层，并将非资源分配而又与硬件特性关系不大的文件管理模块放在中间层，例如 Windows 98 的存储管理、文件管理模块，MS-DOS 的 IBMDOS.COM 模块就安排在中间层。

⑤　对被调用的模块尽量放在靠低层，对于专用模块（不被其他模块所调用）放在较高层，以符合高层向低层请求服务的单向依赖关系的原则。

⑥　将同类或相似功能的模块尽量放在同一层。例如，存储管理、文件管理、设备管理等模块，一般都分别放在同一层，尤其不要将有相近功能的模块放在相隔的两层中。

层次结构的操作系统设计方法通常有自底向上法（bottom-up）和自顶向下法（top-down）两种，前者是从硬件或经软件扩充的第 1 级虚拟机底部开始逐层扩充功能，直至到达目标系统，后者则恰好相反。

## 第 3 节　管 程 结 构

### 1. 管程和类程概念的引入

（1）管程（monitor）

由于各进程在共享临界资源时必须互斥，每次只允许一个进程进入临界区。为此，各个使用临界区的进程必须使用同步操作。此外，为实现异步环境下进程间的通信，也需要同步操作。这样，使大量同步操作分散于各进程中，而同一进程需要使用多个临界资源时也需要若干个同步操作。如此多的同步操作分散在各个进程中，往往会因为使用不当而发生死锁。为解决共享资源同步操作分散在各个进程而引起系统可靠性方面的问题，产生了将共享资源全部同步操作集中在一个程序单位的设想。在用数据结构抽象表示共享资源时，资源管理程序就可用在该数据结构上进行操作的一组过程来表示，从而引入管程（资源管理程序）的概念。所谓管程，是指共享资源的数据结构，以及在其上的能为并发进程所执行的一组操作。

一个管程由以下 4 大部分组成：

①　管程名 monitor；



此为试读，需要完整PDF请访问：www.ertongbook.com

② 局限于管程的数据说明；

③ 在该数据结构上进行操作的一组过程；

④ 对局部数据赋予初值的语句（初启语句）。

管程与进程的区别在于：

① 管程定义的是公用数据结构，进程定义的是私用数据结构；

② 管程定义的是在数据结构上的同步操作和初始操作，并将系统的同步操作相对集中起来，而进程定义的是顺序操作；

③ 管程是为解决进程共享资源的互斥而设置的，进程是为实现系统并发性而设置的；

④ 管程被进程所调用，管程和调用它的进程不能并行操作，而进程间可以并行操作；

⑤ 进程有生命期，由创建到撤消。管程是控制系统所固有的，不需由进程创建或撤消，只供进程调用。

（2）类程（class）

所谓类程，是指专用资源的数据结构，以及在其上规定的全部操作。由于类程是在专用资源上进行操作的一组过程，所以它不存在同步操作。

一个类程由以下 4 大部分组成：

① 类程名 class；

② 局限于类程的数据说明；

③ 在该数据结构上进行操作的一组过程；

④ 初启语句。

管程与类程的主要区别在于，管程是管理共享资源，将竞争共享资源的并发进程通过同步操作处理成顺序执行；类程是管理专用资源，类程被进程所调用，被看成进程的延伸，不同的进程调用各自的类程。

### 2. 管程结构操作系统

从系统功能和实现相结合的观点出发，从系统中提炼出管程、类程、进程等几种基本成分，将系统分解成由这些基本成分组成的模块，并将这些模块按一定的原则编入各层。核心在最内层，是管理 CPU 的专用管程，这种结构称为管程结构（也称为层次管程结构）。管程结构具有以下特点：

① 从数据结构的角度来看，它是将数据结构及其上操作集中起来的一种抽象的数据类型；

② 从资源管理的角度来看，它将系统分成若干模块，用数据表示抽象的系统资源，并根据共享资源和专用资源在管理上的差别来定义模块的类型和结构，从而引出了管程和类程的概念。它使系统的同步操作相对集中，从而增加了模块的相对独立性。

## 习题和思考题

1. 何谓模块组合结构、层次结构和管程结构 OS？试述其特征，并比较这 3 种 OS 结构的优缺点。

2. 试述层次结构 OS 分层的原则。

3. 何谓管程？何谓类程？试述管程和进程的区别。

# 第 1 章　Windows 98 IO 模块总体概述

## 第 1 节　Windows 98 总体结构

Windows 98 已经是一个完整的 32 位 OS，Windows 98 更在 Windows 95 原有的即插即用功能上增加了新的功能。例如，Windows 98 具有完整的 32 位内核；具有抢占式多任务（preemptive multitasking）和多线程（multithreading）；具有完整的 32 位保护模式的文件系统；32 位可安装的文件系统（IFS：Installable File System）驱动程序，32 位驱动程序模型（WDM：Win32 Driver Model），允许 Windows 98 和 NT5.0 使用相同的驱动程序。

Windows 98 的总体结构如图 1.1 所示。

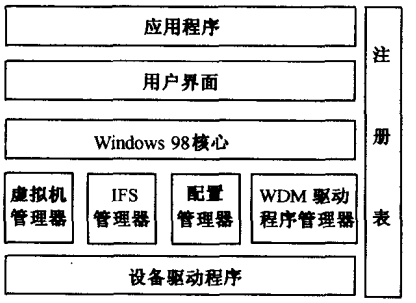



图 1.1　Windows 98 总体结构

### 1. 设备驱动程序

Windows 98 驱动程序分为通用驱动程序（universal driver）和微型驱动程序（mindriver）。前者由 OS 厂商（微软）提供，针对特定类别设备，每一类设备共享该类设备的通用驱动程序，而微型驱动程序由具体设备厂商开发。

### 2. Windows 98 核心

Windows 98 的核心包括 User（用户）组件、Kernel（核心）组件、GDI（图形用户界面）组件。

① User 组件负责管理键盘、鼠标及其他设备的输入，并将其输出到用户界面，User 组件同时也管理声音驱动程序、定时器、通信口等设备；

② Kernel 组件提供 OS 基本功能，包括文件 I/O 服务、虚拟内存管理、任务排队、例外处理（Exception handling）和运行 EXE、DLL 等文件；

③ GDI 组件负责管理屏幕显示的图形系统，将图形输出到屏幕、打印机或其他设备。

### 3. 虚拟机管理器

Windows 98 的应用程序运行在虚拟机（VM：Virtual Machine）上，由 OS 监视和分配

资源。虚拟机管理器（VM Manager）代表 OS 为每个应用程序和系统程序分配资源，并负责建立、维护虚拟机环境。虚拟机管理器提供程序排队与多任务、内存分页和 MS-DOS 方式支持的 3 种服务。

#### 4. 可安装文件系统管理器

Windows 98 文件系统由 3 部分组成：

① 可安装文件管理器：为 Windows 98 提供对不同文件系统的访问能力，它包括 IFSHLP.SYS 和 IFSMGR.VxD 两个文件；

② 文件系统驱动程序：包括 32 位 VFAT（提供保护模式驱动程序来处理磁盘上文件系统）、光盘文件系统（CDFS）、网络重定向程序（Network redirector），以及厂商提供的文件系统组件；

③ 块 I/O 子系统（Block I/O Subsystem）：用于访问实际设备。

#### 5. 配置管理器

配置管理器（Configuration Manager）负责管理 Windows 98 的即插即用功能，它包括：

① 列举器（Enumerator）：用来识别总线、设备的配置设置并收集设备驱动程序或 BIOS 信息；

② 仲裁器（Arbitrator）：负责为每个设备分配资源（IRQ、I/O 端口地址等）；

③ 设备驱动程序：包括实模式的 MS-DOS 驱动程序（*.SYS）、实模式的 Windows 驱动程序（在 SYSMEM.INI 中用*.DRV）和保护模式的 Windows 驱动程序（虚拟驱动程序，*.VxD 或*.386）。

配置管理器为每个设备加载相应的设备驱动程序。

## 第 2 节 Windows 98 引导过程总述

操作系统的引导是指装入并启动操作系统的过程，Windows 98 从硬盘引导入内存，大体上经历了 4 个阶段。在这 4 个阶段中，将由 Windows 98 不同的模块起主控作用。下面，将结合 Windows 98 各模块在初始化过程中内存位置的变化，来阐述初始化过程中各个阶段的特点。

#### 1. 装入 IO.SYS 阶段

这阶段的任务主要由引导记录程序来完成，其实现过程如下：

① 系统加电后，由硬件设备，控制自动指向内存 ROM-BIOS 的入口，进行硬件设备的自测，执行固化在 ROM-BIOS 中的系统自举中断处理（INT 19H），判别是硬盘启动还是软盘启动；

② 如是硬盘启动，将位于 0 头 0 柱 1 扇区的硬盘主引导记录（HBOOT）读到 0000:7C00H 开始的内存，并将控制权转移到硬盘主引导记录，硬盘主引导记录先将自身搬至 0:61B~7FFH 开始的内存，以便腾出 0:7C00H 开始的内存供装入分区 BOOT 程序，HBOOT 执行中先检查自举分区，并将该分区 BOOT 程序读入到 0000:7C00H 开始的内存，并将控制

权转移到分区 BOOT 程序；

③ 如是软盘启动，则直接将软盘 BOOT 程序读到 0000:7C00H 开始的内存，并将控制权转移到软盘 BOOT 程序；

④ BOOT 程序首先将逻辑 1 和逻辑 2 两个扇区（即分区 BOOT 后的第 2、3 扇区）读入到 0000:7E00H 开始的内存，然后转去执行第 3 个扇区的 BOOT 程序；

⑤ 检查盘目录中是否有 IO.SYS 文件，如有，将包含有重装入过程（属于 IO.SYS 的初始化程序 I（SYSINIT I））在内的 IO.SYS 文件头 4 个扇区读入到 70:0H 开始的内存，并将控制权转移到 IO.SYS 重装入过程。

### 2. IO 初始化阶段

这阶段的任务由 IO.SYS 的初始化程序（SYSINITI）来完成。

① 执行重装入过程，建立 FAT 表在内存的映像，然后根据 FAT 表把.IO.SYS 全部装入；

② 保存重装入过程传递过来的参数，对系统数据进行设置、保存，并对字符设备进行初始化；

③ 对软、硬驱进行检查，为每一个块设备生成块设备控制块，并将其链接；

④ 根据系统实际配置取舍系统；

⑤ 控制权转移到 SYSINIT 程序 II。

### 3. Windows 98 的初始化阶段（system 初始化阶段）

这阶段的任务先由 SYSINIT II 后再由 MSDOS.SYS 初始化程序来完成。

MSDOS.SYS 是一个十分重要的配置文件，用于确定启动方式（是 Windows 98 还是 DOS 7.0）。Windows 98 提供多重引导功能，用户能在 Windows 98、DOS 7.0 之间来回切换。

① 将初始化程序 II（SYSINIT II）移至内存高端，以腾出空间来定位 MSDOS.SYS；

② 为磁盘文件管理建立一系列参数表和控制块，建立 SYSINIT II 的运行环境；

③ 由 SYSINIT II 读入 MSDOS.SYS 文件并对其配置命令解释执行处理；

④ 将 MSDOS.SYS 定位于紧靠内存 IO 常驻部分之后，执行 MSDOS.SYS 的初始化程序，设置启动选项，只要不是安全模式启动，就处理系统注册，并构造 Windows 内核模块，加载并执行压缩磁盘文件的驱动程序（如 DRVSPACE、DBLSPACE 等）；

⑤ 解释系统配置文件（CONFIG.SYS），建立供用户运行用的 OS 运行环境；

⑥ 执行有关 HMA 的中断处理，将 Windows 98 IO 模块、Windows 98 内核模块搬至高内存处（1MB 外），腾出基本内存空间为加载最高级别用户处理程序做准备。

### 4. 加载最高级别的用户界面处理程序阶段

本阶段前半部分工作由 SYSINIT II 来完成，后半部分工作由用户界面处理程序初始化程序完成。最高级别的用户界面处理程序可能包括 3 个不同的文件：用户自定义的最高级别命令处理程序、Windows 98 默认的图形界面处理程序及 DOS 默认的命令处理程序（COMMAND.COM）。

① 填写有关参数，以区别是否有交互启动、压缩驱动器或加载图形用户界面等操作；

② 检查最高级别命令处理程序格式，计算该程序的长度，确定内存空间是否足够；

③ 填写 EXEC 参数块（EXEC 加载头），将某一个最高级别的用户界面处理程序加载入内存；

④ 执行用户界面处理程序的初始化工作，显示有关工作界面，等待用户输入（选择）命令或图标。

## 第 3 节 Windows 98 IO 模块的数据结构

数据结构的设计是操作系统设计的关键之一，Windows 98 的 IO 模块数据结构是操作系统的核心部分，大部分 Windows 98 的 I/O 操作都是围绕这些数据结构进行的，Windows 98 的 IO 模块是由文件 IO.SYS（在 MS-DOS 启动下改名为 WINBOOT.SYS）建立的，现在让我们对其数据结构进行分析。

### 1. 堆栈运行环境

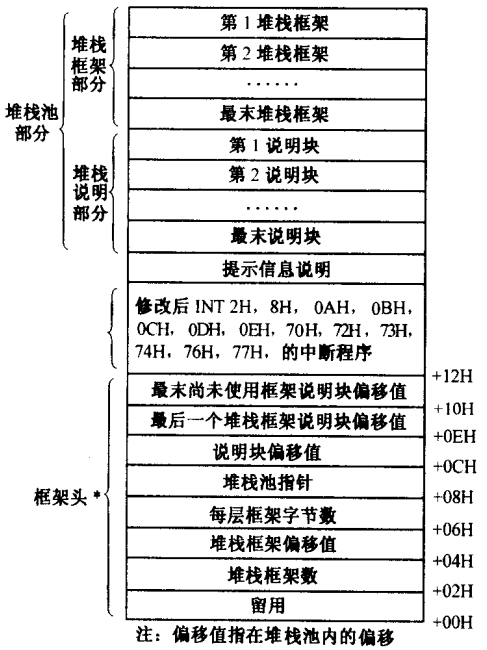



图 1.2（a） 堆栈运行环境的结构

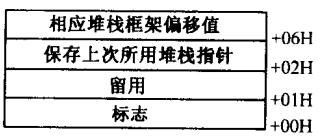



图 1.2（b） 堆栈框架说明块的结构

堆栈运行环境是由 SYSINIT II 在解释 CONFIG.SYS 中（或使用默认值）STACKS 命令时作为 OS 运行环境之一建立在内存 Windows 98 内核之后的区域，堆栈运行环境由堆栈框架头、中断修改部分、提示信息说明部分和堆栈池 3 部分组成。其中堆栈框架头有 12H 个字节，包括有关堆栈池的信息；中断修改部分存放由 SYSINIT 修改的 14 个中断服务程序；提示信息说明部分包括有关堆栈的出错信息；堆栈池由堆栈框架部分和框架说明部分所组成，每个框架说明块长度为 8BH 字节，堆栈运行环境结构如图 1.2（a）所示，堆栈框架说明块结构如图 1.2（b）所示。

在图 1.2 中，堆栈框架的个数及每个框架所占用的字节数是由 CONFIG.SYS 中配置命令 STACKS = N，S 所设置的。其中 N 为堆栈框架数，其值从 8~64，S 为每层堆栈框架的字节数，其值从 32~512。如果不使用 STACKS 配置命令，N、S 的值根据机器默认值选取，在 Windows 98 下，N，S 隐含值分别为 9 和 256 字节。

在图 1.2（a）中，标志位为 00H 时表示当前说明块可用（即当前堆栈框架可用），为 01H 时，表示该说明块已被使用；为 03H 时表示该堆栈框架内数据已被破坏。说明块

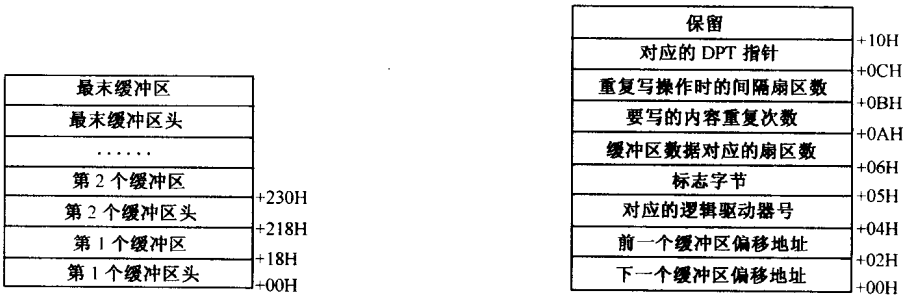
02H~05H 字段存放上次的 SS:SP，以便中断返回时恢复。

在每次发生硬中断时，Windows 98 就从堆栈池中取出一个尚未使用的堆栈框架供系统使用，待中断处理完毕，再将这个堆栈框架释放到堆栈池中。

采用这种动态堆栈结构的意义在于，当一系列中断发生时，能使系统每次中断独自使用一个堆栈区，从而避免了由于一系列中断发生而导致可能产生的中断竞争。由于采用动态堆栈管理，提高了用户界面的透明度。

### 2. 缓冲区链（BUFFERS 运行环境）

BUFFERS 运行环境是由 SYSINIT II 根据系统配置命令在 RAM 的低端建立的磁盘缓冲组，用于存放读写盘数据，每个缓冲区可存放一个扇区长（200H = 512 个字节）数据。采用 BUFFERS 运行环境这种缓冲区链结构，可以减少短期内重复读（写）同一扇区的操作。因为 BUFFERS 运行环境为每个缓冲区设置一个缓冲区头，链首指针指向第 1 个缓冲区头，而第 1 个缓冲区头保存下一个缓冲区的指针和最末缓冲区的指针，最末缓冲区头则保存第 1 个缓冲区的指针和前一个缓冲区的指针，其余位于中间的缓冲区的缓冲区头分别保存下一个缓冲区的指针和前一个缓冲区的指针，因此 BUFFERS 运行环境是具有双向循环链结构的。每个缓冲区头长度为 18H 字节。通过各缓冲区头，将各个缓冲区连接成一个双向链表。BUFFERS 运行环境结构如图 1.3（a）所示；缓冲区头结构如图 1.3（b）所示。




（a） BUFFERS 结构　　（b） 缓冲区头结构

图 1.3　BUFFERS 运行环境的结构

在 Windows 98 中，缓冲区设置与读写盘效率有密切关系，缓冲区越多，读写盘次数越少，但相应的空间开销就会增大，所以缓冲区个数的设置要适当，其值是由 SYSINIT II 解释 CONFIG.SYS 中的 BUFFERS 配置命令的参数值决定。如果用户未使用该配置命令，则系统将选取其默认值，Windows 98 取值是

512KB＜内存容量：BUFFERS = 15；

当 Windows 98 用不同的启动方式启动时，缓冲区的结构有所不同。Windows 98 在硬盘启动时有 7 个选项，分别是：

① Normal

② Logged（\BOOTLOG.TXT）

③ Safe

④ Step-by-step confirmation



此为试读，需要完整PDF请访问：www.ertongbook.com