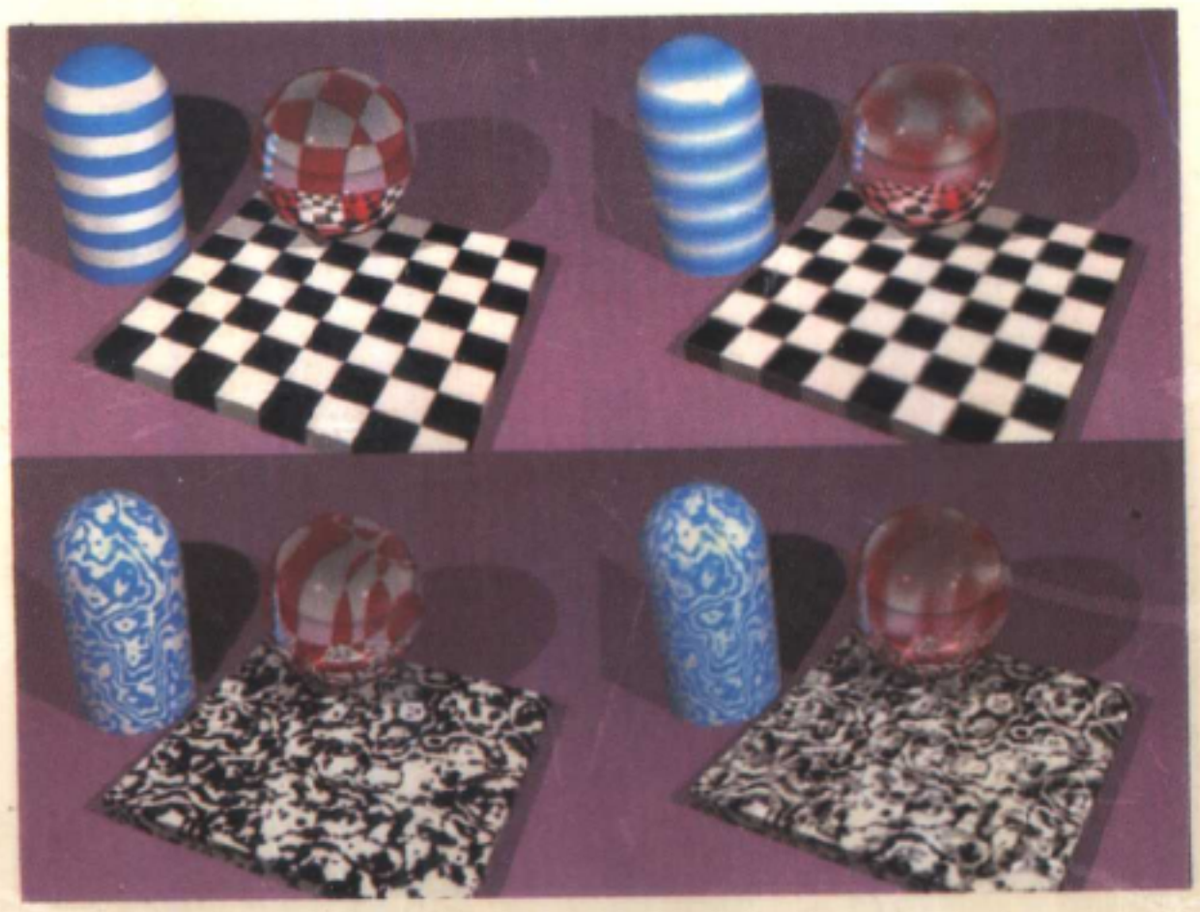
微机操作系统系列丛书

Windows 内核剖析

黄朝明　陈国栋　编著



希望

学苑出版社

台湾第3波文化事业股份有限公司
新近简体字版电脑图书一览表

序号	购书编号	书　　名	定价/册
1	1184	FoxPro 程序设计专辑	29.00 元
2	1166	Internet 学术网络资源	23.00 元
3	1216	电脑动画剖析	19.00 元
4	1269	QuickBasic 编程技巧	13.00 元
5	1251	细说 MS-DOS 6.2	59.00 元
6	1258	AutoCAD 软件探秘	39.00 元
7	1396	AutoCAD 入门——CNS 计算机辅助绘图	29.00 元
8	1303	Internet 实用手册	32.00 元
9	1273	OrCAD/SDT Ⅲ 电脑辅助电路设计彻底研究	38.00 元
10	1271	OrCAD/SDT Ⅳ 电脑辅助电路设计彻底研究	38.00 元
11	1288	Linux 的世界	15.00 元
12	1308	轻松学习 C 语言	28.00 元
13	1270	8051 系列单片机原理与实习	42.00 元
14	1274	例学 MS Word for Windows 5.0 中文版	29.00 元
15	1307	MS－DOS 6.2 实务传真	26.00 元
16	1344	DOS 批处理文件设计技巧	19.00 元
17	1394	中文 Excel 5.0 指令范例图解	29.00 元
18	1361	X－Windows 实用手册	17.00 元
19	1329	Borland C＋＋入门指南	32.00 元
20	1441	Power Point 4.0 for Windows 应用宝典	23.00 元
21	1563	CADKEY 3D 绘图设计实用手册	19.00 元
22	1562	MIDI 爱好者手册(音乐设备数字接口)	48.00 元
23		Protel 计算机辅助电路板设计实例	39.00 元
24	1568	中文版 PC DOS 6.3 实用指南	32.00 元
25		AutoCAD R12.0 绘图软件基础与提高	35.00 元
26	1555	看图例学 Page Maker 5.0	29.00 元
27		局域网络与 Novell 纵横集 安装与指令篇	32.00 元

台湾碁峰资讯股份有限公司新到中文简体电脑图书

序号	购书编号	书　　名	定价/册
1	1330	MS－Word 6.0 for Windows 中文版操作手册	23.00 元
2	1448	AutoCAD 3D 技巧与应用	29.00 元
3	1508	Visual Basic for Windows 3.X 入门与实例应用	25.00 元
4		AutoSurf 3D 曲面造型设计	25.00 元
5	1520	AutoCAD for Windows AME&AVE 技巧与应用	32.00 元
6	1524	中文 Word 5.0 for Windows 教程	35.00 元
7		dBASE 5 for Windows 图例入门	32.00 元
8		易学易用 CorelDRAW 5.0	45.00 元
9		中文版 Lotus 1－2－3 for Windows 5.0 教学手册	34.00 元
10		活用 Lotus 1－2－3 cc:Mail	17.00 元

欲购以上图书的朋友请与 010－2541992,2562329 书刊部联系,
或传真 010－2579874,2561057 书刊部

ISBN 7-5077-0804-7/TP・15

本册定价:32.00 元

微机操作系统系列丛书

Windows 内核剖析

黄朝明　陈国栋　编著
鑫万博　陈金凤　审校

学苑出版社

(京)新登字 151 号

内 容 简 介

本书深入探讨 Windows 内部运作机制，揭开 Windows 操作系统的神秘面纱，介绍 Windows 内部如何进行内存管理、如何启动进程、窗口系统如何运作、图形设备驱动程序界面的关键问题、调度程序的实现、消息系统和动态链接库的运行机制等。本书适合于设计低级 Windows 应用程序(特别是设备驱动程序)以及编写与操作系统密切相关的应用程序的读者参考。

欲购本书的用户，可与北京海淀 8721 信箱书刊部联系，电话 2562329，邮政编码 100080。

微机操作系统系列丛书

Windows 内核剖析

编　　著：黄朝明　陈国栋

审　　校：鑫万博　陈金凤

责任编辑：甄国宪

排　　版：万博图书创作社

出版发行：学苑出版社　　**邮政编码**：100036

社　　址：北京市海淀区万寿路西街 11 号

印　　刷：列电印刷厂

开　　本：787×1092　1/16

印　　张：30.625　**字　　数**：710 千字

印　　数：1～2000 册

版　　次：1994 年 6 月北京第 1 版第 1 次

ISBN7-5077-0804-7/TP・15

本册定价：32.00 元

学苑版图书印、装错误可随时退换

目　　录

试读结束：需要全本请在线购买：　www.ertongbook.com

试读结束:需要全本请在线购买: www.ertongbook.com

前　言

每天都有数以百万计的人在使用 Microsoft Windows 操作环境，有数以千计的人在 Windows 环境下编程，而且每天都有数以千计的 Windows 程序设计人员要寻思：在某个给定的条件下，Windows 将如何进行处理。Windows 是一个相当巨大、复杂和灵活的编程环境，Microsoft 公司的文档里给出的只是 Windows 的一些皮毛知识。

那么是不是有一些方法能让我们掌握在某个特定条件下 Windows 是如何处理各种各样的问题的呢？当然有，建议有关人员最好能读一读有关自己不理解的某个特征的源代码，而不是查阅那些无用的手册。

的确是这样的，程序员们询问的有关 Windows 问题的答案大部分都在 Windows 的源代码中，也就是说在诸如 Krnl386\winexec. c(或 Winexe. asm?)，user\wmvisrgn. c 和 gdi\meta. c 等等之类的文件中。

但问题是，这些源代码中的大部分都保存在 Microsoft 的 Redmond，WA 之类的我们手头没有的文档中，Microsoft 公司曾经讨论过要将其新的 Windows NT 操作系统的源代码发放给某些大学使用，但从来没有谈论过要将 Windows 的源代码提供给少数 OEM(原始设备生产商)和 ISVs(独立软件商)以外的人员使用。在 Windows 软件开发工具(SDK)中包含了两小段 Windows 的源代码(user\defwnd. c 和 user\def dlg)，大部分 Windows 设备驱动程序的源代码保存在 Windows 的设备驱动程序开发工具(DDK)中，但也仅此而已。

这一点很遗憾，但并不是致命的障碍。任何人只要走进一家服务态度良好的软件商店，都可以买到一份 Windows 软件包，然后带回家去反汇编 Windows。几乎任何一本讲有关计算机法律和商业秘密的教科书中都会讲到，软件购买者可以自由地反汇编一个计算机程序，这就是“通过逆向工程进行发现的能力”，特别是当 Windows 市场占有率越来越大的时候，具有这种能力特别有用。Microsoft 公司还提供诸如 EXEHDR，CodeView 和 Windows 的调试版本等等工具软件，这就使得反汇编变得很容易，甚至是不可避免的。

估计最大的问题将可能是：“需要这个信息吗?”当然当程序员在 Windows 环境下编程时，若不需要知道 Windows 内部的处理情况是再好也不过的了。在软件工程中，“信息隐藏”是最关键的概念之一，但遗憾的是，现在的软件工业离将 Windows API 这种编程界面当作黑箱元素来处理还相差甚远，所以 Windows 环境下的操作员会一遍遍地琢磨在给定条件下，诸如 CreateWindow，GetMessage 或 GlobalAlloc 之类的函数将如何进行处理。读完本书，他们会立即发现问题的答案。

举一个例子来说，看完 Microsoft 文档中有关 GlobalAlloc 函数的说明，就会知道可以用 GMEM_FIXED 选项调用 GlobalAlloc 函数，以通知 Windows 所分配的内存块不能在线性地址空间内移动。先让我们翻到第二章看看 GbTop 函数的伪代码(由 GlobalAlloc 函数调用 GbTop 函数)。这段伪代码非常直率地显示出，在 Windows 3.1 中，如果不是由 DLL 中的函数调用 GbTop 函数，则 GbTop 只是简单地将 GMEM_FIXED 标志置为 off。如果程序员不知道这些细节问题，在 Windows 环境下编程时还能感到安全吗?

想在这个层次上检测 Windows 还有一个理由，在计算机法律中，计算机程序被视为“文

学著作”，但遗憾的是这个思想没有被引伸到这样一个逻辑结论上：应该像读书一样读像Windows这样的计算机程序的源代码。事实上，只有极少一部分大型计算机程序被公开检查，并能达到文学评论家所称的“仔细阅读”这种深度，这个极少一部分就包括UNIX操作系统。

引 言

也不过是代码而已

除了 Microsoft Windows 表面上的成功以外，它的内部工作原理对大多数程序员仍然是一个不解之谜。从宏观上看，Windows 使用的数据结构和算法与我们自己编制的代码差不多，但当我们面对数目惊人的 API、消息、句柄、回调函数等等的时候，就会觉得东西太多，不容易记住。Windows 看上去就像一个巨大的实体，用户通过命令发送自己的处理请求，如果不出问题，Windows 会正确处理这次处理请求。如果用户激怒了 Windows，Windows 将给用户程序回敬一个不可恢复的应用程序错误(Unrecoverable Application Error，简称为 UAE)。换句话说，Windows 程序员心里常常觉得 Windows 是一个洪水猛兽，而不仅仅是一个由用户程序调用，或者是调用用户程序的大型代码函数库。

本书准备剥去 Windows 神秘的外衣，并揭示出 Windows 内部的工作原理。本书的目的是给读者提供某种程度的条理性和预见性，如果读者能理解 Windows 是如何创建和维护对象的，则对自己程序中出的问题的理解程度也会加深，而不用再想出种种途径去解决问题并直到解决为止。

现在讲如何编制 Windows 应用程序的书非常多，当然其中也有一些好书，但它们无一例外地将注意力放在如何向一个“黑箱”进行输入上。如果程序想将一幅图形放在按钮上，只要把这个放在这儿，把那个放在那儿，转动一下手柄，按钮就画成了。本书则另辟一条蹊径，本书重点是打开黑箱，揭示出黑箱中的秘密，并向读者展示在别的书籍中描述的技巧是如何工作的，并解释一下为什么要按那种方式工作。

有这样一种见解，即没有必要知道黑箱内部的情况。这个见解的前提是：如果你设计了一个可靠的黑箱，用户不必知道黑箱中到底是如何处理的。但我们对这个前提的合法性产生疑问，就是说，我们都认为 Windows 这个黑箱并不是很完善的。为了使程序设计得更可靠，我们应该在更深的层次上理解问题，而不能仅仅局限于文档提供的资料。这并不是想贬低 Microsoft 公司文档编制者的工作成绩，相反，它只是数目惊人的“push the limit”的应用程序和古怪的 CPU 体系结构的副产品而已。

本书内容简介

在后续章节中，我们选取一些使用广泛的操作系统的概念，并详细讲述 Windows 实现这些概念的方法。例如，我们将看到磁盘上的一个可执行文件如何变成一个带有可与用户交互的窗口的当前运行进程。内存管理那一章特别长，因为这一章是 Windows 的基础。

这本书绝对不可能将 Windows 所有的内容都囊括进去，因此只能选取那些我觉得特别关键、最熟悉，而且对大多数的程序员都特别有用的内容来重点讲述。即使要求这样严格，涉及的内容还是很多，尤其是 Windows 中还有许多各式各样的变化，考虑下列这些变化：

- Windows 3.0 相对于 Windows 3.1 的区别
- Windows 标准版与增强版的区别
- Windows 的远东版本
- WINOS2(OS/2 2.0 环境下的 Windows 版本)
- Windows for Workgroups
- Win32s(在 Windows 3.1 中运行 32 位的应用程序)

将所有的这些组合都包括进去几乎是不可能的,所以作出了明智的选择:本书以 1992 年 3 月发行的 Windows 3.1 增强版作为背景。在适当的场合,也会讨论别的版本的一些情况(例如,我们将讨论当 Windows 3.1 版本的加载程序遇到 Win32 应用程序时将如何处理等问题)。尽管在后续章节中并不特别强调 Windows for Workgroups,但对于 Windows 3.1 的讨论,同样也适用于 Windows for Workgroups。本书所讲的 Windows"内核"对于 Windows 3.1 和 Windows for Workgroups 来讲没有什么区别。

因为 Windows 包罗万象,所以许多内容我们没有时间去讨论。例如,所有讲 Virtual Device Drivers(虚拟设备驱动程序),Virtual Machine Manager(虚拟机器管理程序)以及如何在 Windows 环境中运行 DOS 应用程序等内容,本书都未涉及到。站在更高的角度上,必须放弃对对话框、客户定制控制、DDE 和 OLE 的工作原理的讨论,这并不是因为不能讲述网络 API,多媒体扩充增加到 Windows 3.1 中的新功能以及 Windows for Workgroups 等等内容,这本书主要是为了集中讲述 Windows 内核,但幸运的是,其他一些书籍已经打算讨论本书未能涉及的内容。

为了正确对待本书所讨论的专题,作了一系列的假设。在编写这本书时,假定读者应对分段、选择器和保护模式的概念有所了解,不打算详细讨论这几个问题,宁愿利用这个时间去讨论别的专题,或者是将问题讨论得更透彻一点。

另外还假设读者有 Windows SDK 文档,或者有 SDK 文档完整的替代物(例如,Borland C++的 Windows 文档)。例如,在"内存管理"一章中,对于如何实现 GlobalAlloc( )函数的描述文本,如果在 SDK 文档中能找到,就不准备再重复了。本书的目的是能给读者提供新信息。

最后,如果读者能理解操作系统的一些基本概念,也很有用(例如,多任务管理是什么意思,虚拟内存在更高层次上的操作原理等等),当然读者没有必要为了读这本书而专门去攻读计算机学方面的学位,只要求读者不必在一般操作系统概念上耽误太多的时间,因为我们的目的是研究如何实现 Windows 中的这些概念。

简而言之,这本书将讲述如何编制 Windows 程序,它只告诉读者 Windows 是如何工作的,以及如何加深读者现有的知识。

分析方法

与 Unix(或其变种)之类的操作系统不一样,Windows 的源程序都由 Microsoft 公司保存起来,所以要想研究 Windows 的工作原理是要费一番脑筋的。本书主要采用两种方法来解剖 Windows 和判断 Windows 的工作原理。

采用的主要方法是分析 Windows 文件列表中的汇编语言。这些列表都是由自己开发的

程序 WINDIS 产生的。

现在以 KRNL386.EXE 中的 IsTask( )函数为例来分析由 WINDIS 程序生成的列表。IsTask( )函数是一个已经公开的 Windows 3.1 函数，这个函数接收一个 hTask 参数，并判断该参数是否为一个任务句柄：

```
ISTASK proc
881E:    MOV      BX,SP
8820:    MOV      AX,WORD PTR SS:[BX+04]    ; 从堆栈中取参数，
                                            ; 并赋给 AX
8824:    OR       AX,AX                     ; 参数==0 吗?
8826:    JE       8840                      ; 如果失败就跳转到 8840 处
8828:    LSL      BX,AX                     ; 取选择器极限值，并放到 BX 中
882B:    JNE      8840                      ; 如果不是一个合法选择器则返回 0
882D:    CMP      BX 00FC                   ; 应确保段限值足够大，以允许读
8831:    JL       8840                      ; 取下面的值
8833:    MOV      ES,AX                     ; 在偏移地址 00FA 处
8835:    CMP      WORD PTR ES:[00FA],4454   ; 查找 TDB 特征('TD')
883C:    JNE      8840                      ; 如果未找到则返回 0
883E:    JMP      8842                      ; AX 中仍然保存参数值
                                            ; 跳转到下一条语句处
                                            ; 这条语句将 AX 置为 0
                                            ; 使得 ISTASK( )函数返回
                                            ; FALSE
8840:    XOR      AX,AX                     ; 如果都失败了，就跳转到
                                            ; 此处
8842:    RETF     0002                      ; 返回 AX 中的值(一个 TDB 或 0)
ISTASK   endp
```

如果 WINDIS 能增加如此详细的注释的话就太好了，但遗憾的是现在它还不行。读者刚才看到的程序就是用 WINDIS 程序输出的，注释和操行符是添加的，这样做可使程序的可读性强。

由于现在很少有人用汇编程序编制 Windows 应用程序，所以觉得如果能将汇编语言翻译成 C 语言伪代码，那么这本书的可读性将大大增强。对于上面给出的 IsTask( )代码，将其翻译成下面的 C 语言源代码：

```
Pseude code for IsTask( ) -- CONTEXT.OBJ
//参数
//        WORD    hTask      // 一个潜在的 hTask(一个 TDB 选择器)
//局部变量：
          WORD    segLimit   // 被传递的选择器参数的极限
    if (hTask==0)
        return 0
```

试读结束：需要全本请在线购买：　www.ertongbook.com