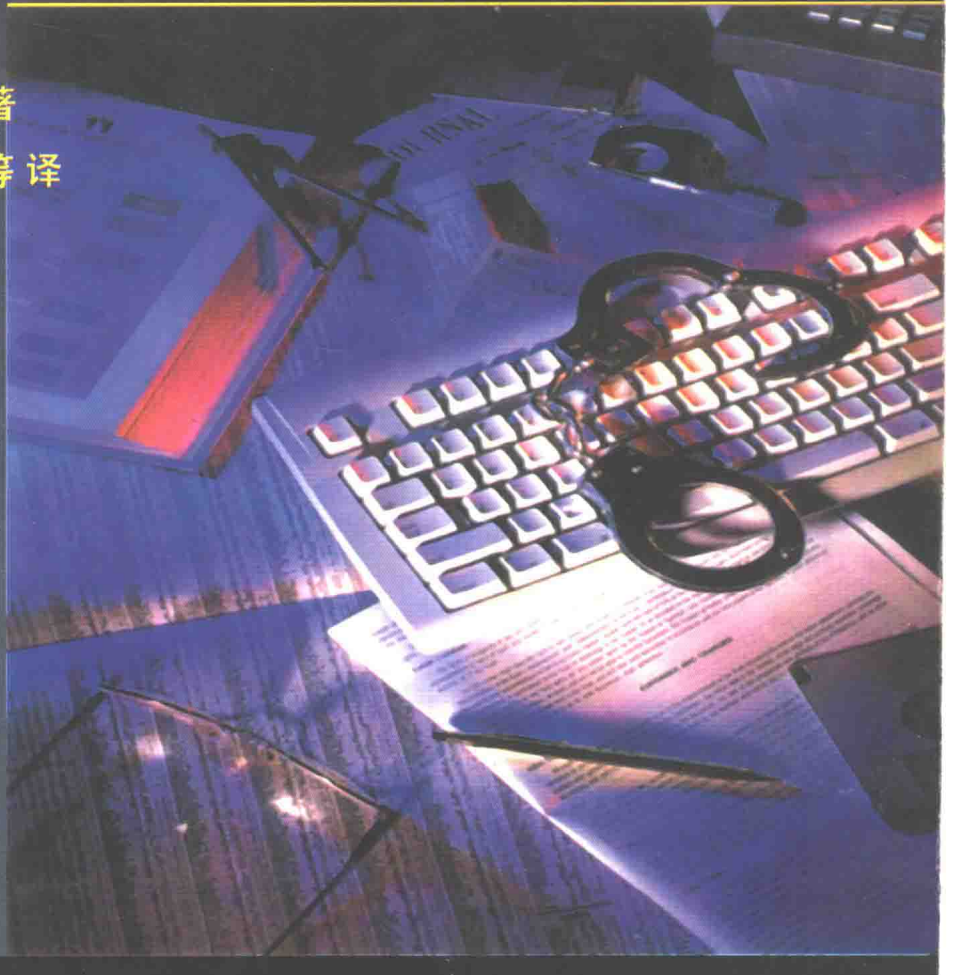
The Microsoft Windows 95 Developer's Guide

The Microsoft Windows 95

开发人员指南

(美) Stefano Maruzzi 著
周靖 王勇 宋玺如 等译



计算机软件开发
与程序设计
系列丛书

机械工业出版社 西蒙与舒斯特国际出版公司

计算机软件开发与程序设计系列丛书

The Microsoft Windows 95

开发人员指南

(美)Stefano Maruzzi 著

周靖 王勇 宋玺如 等译

机械工业出版社
西蒙与舒斯特国际出版公司

本书分为两部分。前一部分讲述了 API Win32 中的软件开发、开发工具以及应程序的开发。其中着重介绍了消息的重画模式、资源文件、菜单的运用、建立窗口的艺术，对话框的管理以及预定义窗口类在 API Win32 应用程序内的开发。后一部分讲述了 Win95 的通用控件、图形设备接口、非标准输入与输出、内存管理与 DLL、多线程 IPC 和 I/Windows 高级技术、Win95 外壳开发等新技术在 Win95 中的应用。

本书可供计算机软件开发人员及大专院校电子与计算机专业的师生使用与参考。

Stefano Maruzzi: The Microsoft Windows 95 Developer's Guide.
Authorized translation from the English language edition published by ZD.
Copyright 1996 by Ziff-Davis.
All rights reserved. For sale in Mainland China only.

本书中文简体字版由机械工业出版社和美国西蒙与舒斯特国际出版公司合作出版，未经出版者书面许可，本书的任何部分不得以任何方式复制或抄袭。
本书封面贴有 Prentice Hall 防伪标签，无标签者不得销售。
版权所有，翻印必究。

本书版权登记号：图字：01-96-1222

图书在版编目(CIP)数据

The Microsoft Windows95 开发人员指南/(美)马入芝(Maruzzi, S.)著；周靖等译. -北京：机械工业出版社，1997.1
(计算机软件开发与程序设计系列丛书)
ISBN 7-111-05459-8

Ⅰ.T… Ⅱ.①马… ②周… Ⅲ.窗口软件，Windows95-软件开发-指南 Ⅳ.TP316-62

中国版本图书馆 CIP 数据核字(96)第 23780 号

出 版 人：马九荣(北京市百万庄南街 1 号 邮政编码 100037)
责任编辑：何伟新 东凌
北京牛山世兴印刷厂印刷·新华书店北京发行所发行
1997 年 1 月第 1 版第 1 次印刷
787mm×1092mm 1/16·49.5 印张 1236 千字
0001-5000 册
定价：86.00 元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

译者的话

自从美国微软公司（Microsoft）于1995年8月发布Microsoft Windows 95之后，各种个人计算机上的应用程序都开始从DOS平台向Windows95平台转移。

对于专业程序员，或者是将要成为专业程序员的人来说，为Windows95编写应用程序看来是必由之路。国际知名的软件开发和培训专家Stefano Maruzzi在《The Microsoft Windows95开发人员指南》一书内为使用C，C++以及Visual Basic的程序员展示了怎样建立全功能的32位Windows95应用程序。在这本书中，针对Win32API开发过程的第一步骤，作者都用多个示范文件及对应的.EXE文件（可以在附带的CD盘中找到）进行了补充说明。

为了帮助广大计算机用户学习怎样从头编写32位应用程序，如何运用一些基本的和高级的技巧，从而编写出出色的Windows95应用程序，我们翻译了这本《The Microsoft Windows95开发人员指南》。参加本书翻译工作的有周靖、吴卫华、王勇、宋玺如、冀惠刚、王丽梅、张丽霞、贾银萧、李晋宏、李竹华、潘旭燕、黄为、徐茜、尤晓东、徐晓梅、黄培林、王丽梅、陈琦、陈红梅、刘涛、李明辉、刘丽、杨淑英、朱锦鸾等同志。由于时间仓促，翻译过程中难免出现错误，欢迎广大读者指正。

译者

1996.8于北京

The Microsoft Windows 95 开发人员指南

《计算机软件开发与程序设计系列丛书》

- Windows 95 开发人员指南
- Visual Basic 4开发人员指南
- Visual Foxpro 3.0 开发指南
- Windows 95 API 程序设计
- Visual Basic 4 API 程序设计
- 按实例学 Delphi 2 程序设计
- 应用 Visual Basic 开发客户机/服务器
- Java编程指南
- I/O接口程序设计入门与应用
- 精通VISUAL BASIC 4.0 多媒体程序设计

本书特色

- 介绍API Win 32中的软件开发工具以及应用程序的开发，包括：消息的重画模式及资源文件，高级用户接口，诸如弹出式菜单、对话框、表格、向导和菜单条，窗口框架的拖拽高级内存管理与动态链接库，图形设备接口，非标准输入与输出，多线程IPC和I/Windows高级技术及Windows 95外壳开发
- 适用于具有一些C.C++或VisualBasic程序设计经验而又亟待提高的读者
- 书中有大量例程可直接引用到您的应用程序中

ISBN 7-111-05459-8

9 787111 054597 >

北京华章图文信息有限公司
北京市百万庄南街14号
电话：68326444
邮编：100037

ISBN-7-111-05459-8/TP · 435
定价：86.00元

试读结束，需要全本PDF请购买 www.ertongbook.com

目　录

第 1 章　Win32 中的软件开发

Microsoft Windows 是微软公司针对配备 Intel x86 微处理芯片的个人计算机推出的第一种多任务解决方案。Windows 的第一个版本是 1.01，于 1985 年发布。在那个时候，正在使用的几乎所有个人计算机配备的都是 Intel 的 8088/86 微处理芯片。当时，第一套 80286 系统（IBM PC AT）也不过刚刚才发布（1984 年 8 月）。

不过，Windows 并没有很快地兴旺起来，这主要是由于硬件和 DOS 操作系统的内在限制决定的。另外，图形分辨率和处理器的速度也是一个很大的瓶颈。除此以外，当时的 CGA 标准只能显示 320×200 个象素，并且只有四种颜色，而且价格昂贵，这样就阻碍了它的广泛普及。另外，当时的软件开发手段落后，只有一些“高手”才会对编程感兴趣。这和近 80 年来工业化发展所取得高度生产力是很不相称的。Windows 的后续版本取得了稳定的进展，每个版本的功能都比以前更加强大，并且越来能充分发掘出硬件的潜力。

现在的情况已经完全不同了。越来越多的个人电脑用 80486 和 Pentium 芯片武装起来，80386 系统很快就成了昨日黄花。桌面计算机很容易就可以用至少 16 色达到 800×600 的分辨率。事实上，许多计算机系统都可以处理更高的视频分辨率和支持 256 色同屏显示，甚至能支持上百万种颜色同屏显示。对于新出厂的每台个人电脑来说，鼠标已经成为标准的和不可缺少的配置，几百兆的硬盘也随处可见。伴随着硬件性能的持续提高，价格的下降也是同步进行的。这样造成的结果就是：个人电脑大量充斥市场。

1992 年 4 月，微软推出了 Microsoft Windows 3.1。不久，1993 年 8 月，Microsoft Windows NT 3.1 也上市发行了。1993 年 12 月，微软公司正式宣布他们准备开发 Windows 环境的一种新的 32 位版本。Windows 环境下的第一个 32 位 API 是在 3.1 版本的 Windows NT 里出现的。要使 Win32 得到公众的接受，只有等到 Microsoft Windows 95 广泛铺开的时候才会实现。最初，这个 32 位版本的代码名叫作 Chicago（芝加哥），后来改名为 Microsoft Windows 95。在本书内，我们统一称呼为 Win95。

1.1　Microsoft Windows 的演变

最开始的时候，开发一个 Windows 应用程序意味着必须使用一种高级的编程语言，这几乎总是 C，并且还要和运行环境的软件开发包（SDK）交互作用。这些函数总合起来就是众所周知的“应用程序编程接口”（API）。

随着 NT 的降临，当 Windows API 升至 32 位的时候，它就演变成了著名的 Win32，而较老的 API 版本则叫作 Win16。通过表 1-1，我们可以大致看出 Microsoft Windows 的演变过程。

表格的“编码”栏是指构成 Microsoft Windows 环境的编码本质。API 栏是指软件开发人员在对应平台上开发应用程序时使用的应用程序编程接口。“应用程序”栏是指各 Microsoft Windows 版本所支持的程序的本质。

表 1-1 Microsoft Windows 的演变

产　品	发行日期	编码	API	应用程序
MS Windows 1.x	1985 年 11 月	16 位	Win16	16 位
MS Windows/386	1987 年 9 月	16 位/32 位	Win16	16 位
MS Windows 2.x	1987 年 12 月	16 位	Win16	16 位
MS Windows 3.0	1990 年 5 月	16 位/32 位	Win16	16 位
MS Windows 3.1	1992 年 4 月	16 位/32 位	Win16	16 位
MS Windows for Workgroups 3.1	1992 年 11 月	16 位/32 位	Win16	16 位
MS Windows for Workgroups 3.11	1993 年 11 月	16 位/32 位	Win16	16 位
MS Windows 3.11	1993 年 12 月	16 位/32 位	Win16	16 位
MS Windows NT 3.1	1993 年 8 月	32 位	Win32	32 位/16 位
MS Windows NT AS 3.1	1993 年 8 月	32 位	Win32	32 位/16 位
MS Windows NT Workstation 3.5	1994 年 10 月	32 位	Win32	32 位/16 位
MS Windows NT Server 3.5	1994 年 10 月	32 位	Win32	32 位/16 位
MS Windows NT Workstation 3.51	1995 年 7 月	32 位	Win32	32 位/16 位
MS Windows NT Server 3.51	1995 年 7 月	32 位	Win32	32 位/16 位
MS Windows 95	1995 年 8 月	32 位/16 位	Win32	32 位/16 位

Win32 代表了对 API 总体质量最显著的一次提升。它正逐渐成为应用广泛的一种目标平台，特别是发行了 Microsoft Windows 95 以后，人们对它更是刮目相看。微软的目标是发行一套统一的 Win32 开发包，使其能在 NT 和 Windows 95 支持的所有硬件平台上寻址。

微软现在这两套 32 位 API 子集会在以后几年内得以完全的合并。当前，针对 NT 和 Win95 进行 Win32 开发还存在一些差异，并且对这些差异必须引起足够的重视。NT 支持 Win32 的全集，这个全集最初是于 1992 年 7 月出版的。1993 年 8 月，随同 NT 的发行，又对它进行了一番修订。从另外一方面来说，由于 Windows 95 自身的限制，它只具有 Win32 一个子集的地位。这个子集就是我们常说的 Win32c。然而，微软的工程师已被授命在 Win95 里增加新的 Win32 API 函数，而这些函数在 NT 内却是没有的。之所以会取得这项进展，主要是由于在 Microsoft Windows 95 里引入了一种面向对象的用户界面。将来，甚至在 NT 里都会包含 Microsoft Windows 95 对 Win32 所作的全部扩展。这将在 NT 的下一个版本（当前的代码名为 Microsoft Cairo）发布时实现。表 1-2 为大家比较了当前两个 Win32 版本间的不同。

表 1-2 比较 Win32 API 在 Microsoft Windows 95 和 Microsoft WindowsNT 里的不同特性

Win32 特性	MS Windows 95	MS Windows NT
32 位内存管理	√	√
文件映射	√	√
联网	√	√
OLE 2.x	√	√
邮件槽和命名管道	√（只用于客户端）	√
Win32 线程管理	√	√
高级 GDI	√（大部分）	√
远程进程调用	√	√
MS Windows 95 UI	√	只有通用控件
GDI 转换		√
事件登录		√
安全保证		√
Unicode（统一编码）		√

试读结束，需要全本PDF请购买 www.ertongbook.com

以前，API 是和 Microsoft Windows 特定的版本紧密集成在一起的。但是如今，Win32 却通过两个操作系统间的微妙差别而实现了一体化。这是 Windows 编程史上最值得书写的一笔。

1.2 我们在哪里

到目前为止，我们还没有提到关于 Win32s 的任何事情。这是 Win32 的一个附加的子集，它的目标是开发出甚至有能力在 Microsoft Windows 3.x 系统上运行的 32 位编码（当然要有恰当的支持组件）。但是，这种设计方案引发了下面这个问题：应该选择什么平台来实施 Win32 开发呢？为了回答这个问题，我们可以先设计一个名为 Platform（平台）的小型的、简单的应用程序。这个应用程序的目的是告诉用户关于基础操作系统的信息。

有两个函数可以识别当前运行的是 32 位 Windows 的何种版本，它们分别是 GetVersion () 和 GetVersionEx ()。我们建议大家只使用第二个函数，它比第一个更清楚，功能也更强。下面是这两个函数的语法：

```
DWORD GetVersion (void);
DWORD GetVersionEx (LPOSVERSIONINFO osvi);
```

尽管 GetVersionEx () 要以数据结构 OSVERSIONINFO 为基础，但这两个函数都会返回一个 DWORD（双字）。GetVersion () 的返回值包含了一些信息，通过这些信息就可以知道程序是在何种 32 位平台上运行的。但是只有进行了一番处理，提取出二进制位的含义后，才有可能真正实现。请参考下面这个代码段。

```
...
DWORD dwVer;
WORD wHi;
int i;

dwVer = GetVsrsion ();
wHi = HIWORD (dwVer);
i = wHi >> 14;
switch (i)
{
        case 0:
                strcpy (szString, "MS Windows NT");
                break;

        case 1:
                strcpy (szString, "MS Windows 3.x + Win32s");
                break;

        case 3:
```

```
                strcpy (szString, "MS Windows 95");
                break;
    }
    ...
```

识别的方法是检查返回高字节的最后两位。如果为二进制的 00，就表明是 Windows NT；如果为 01，就表明是 Win32s；如果为 11，就表明是 Win95。从另外一方面来说，GetVersionEx () 则返回了 dwPlatformId 项中需要的信息，如下所示：

```
OSVERSIONINFO osvi;
DWORD version;
osvi.dwOSVersionInfoSize = sizeof (osvi);
GetVersionEx (&osvi);

switch (osvi.dwPlatformId)
{
        case VER_PLATFORM_WIN32s:
        {
                strcpy (szString, "MS Win32s on MS Windows 3.1")
        }
                break;

        case VER_PLATFORM_WIN32_WINDOWS:
        {
                strcpy (szString, "MS Windows 95");
        }
                break;

        case VER_PLATFORM_WIN32_NT:
        {
                strcpy (szString, "MS Windows NT");
        }
                break;
}
...
```

运行 Platform 以后 (图 1-1)，大家可能会立即对 Windows 95 的新奇特性产生兴趣。你肯定会注意到一个彩色的光标、菜单和其他窗口组件的新外观，另外还有覆盖于客户区的背景位图。这些特性和许多其他的特性都会在后续的章节中进行详细介绍。

在本书附带的 CD 里，Listing 1.1 包含了 Platform 的源代码。它强调了 Win32——32 位程序的运行环境的运用。

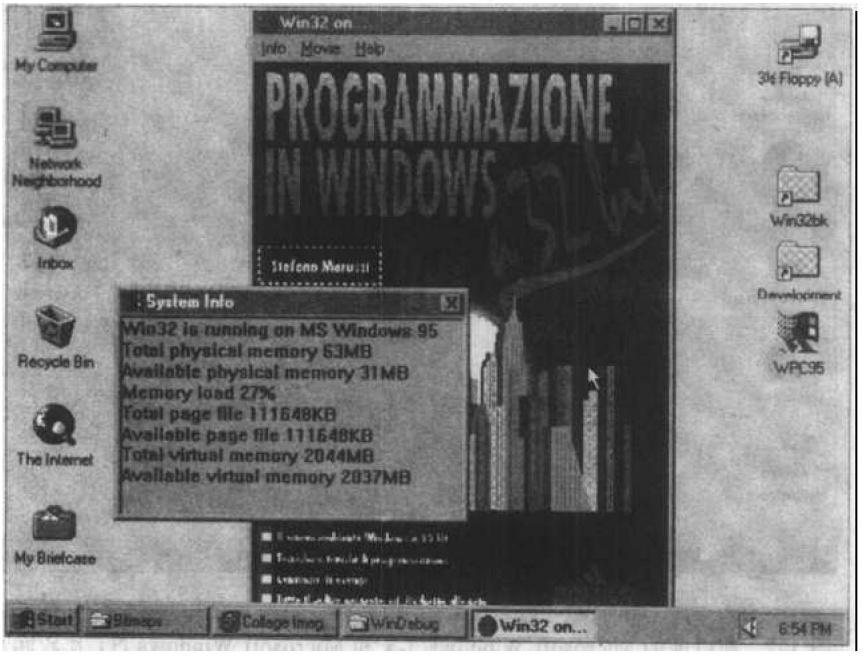


图 1-1　Platform 在 MS Windows 95 中的运行情况

1.3　32 位编程的引入

尽管出现了 Win32，同时它还在不断地进化，但是微软支持的主要 32 位平台还是 MS Windows 95 以及居于从属地位的 Windows NT。Microsoft Win32 是建立具有如下特性的程序的起点：

- ▶应用程序的执行独立于所有硬件设备以外。
- ▶图形用户界面。
- ▶Microsoft Windows 95 和 Microsoft Windows NT 之间透明的移植能力，可以移植至支持 Microsoft Windows NT 的 RISC 硬件平台。
- ▶面向对象的用户界面，减轻了用户学习的负担。
- ▶高性能的抢先式多任务和多线程。
- ▶高级的多媒体支持（声音、图形、影象等等）。
- ▶通过 OLE 技术实现的多个应用程序的对象定位。

通过这段对 Win32 的简要总结，32 位程序开发的前途应该是很明朗的了。Win32 可以应用于特定的操作系统，这种系统应能直接控制和处理 PC 硬件资源，而不必依赖于 MS-DOS 系统服务。通过图 1-2，大家可以看到系统引导以后出现的 Windows 95 面向对象的用户界面。

在 PC 的世界里，活跃着 Windows 16 位程序的许多编程高手。对于这些开发者来说，把自己的程序转换成 32 位是一件让人激动的事情。这就有机会重温一下自己的编程经历；在其中注入一些新鲜血液以后，甚至还能提高它们的性能。然而，许多新的开发者则是由于受到

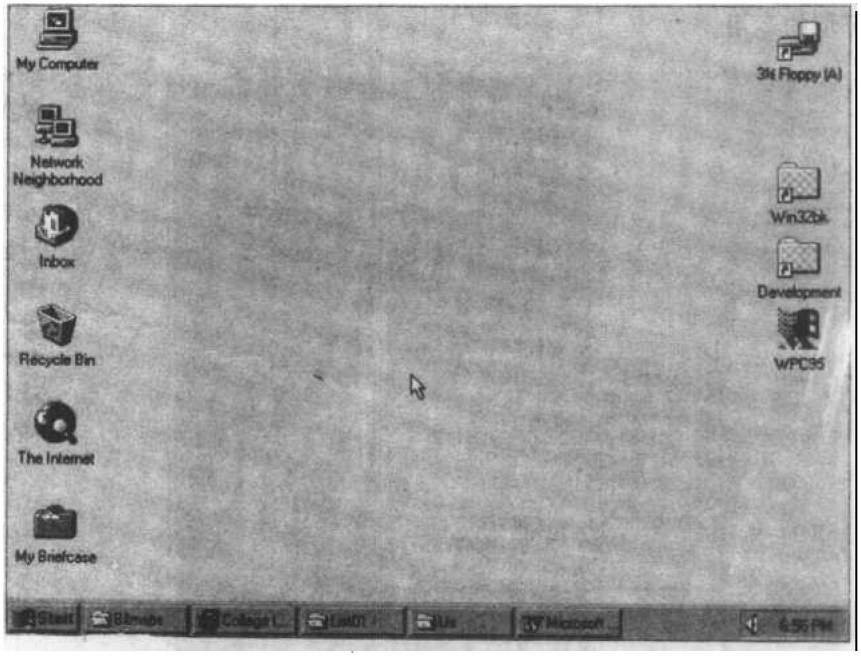


图 1-2　和以前的 Microsoft Windows 3.x 和 Microsoft Windows NT 比较起来，用 Microsoft Windows 95 装备的系统在外观上看起来颇不相同

Windows 的 32 位版本以及新 Win32 API 的强大功能的吸引，所以才投身到 Windows 编程行列中来的。无论你的背景如何，过渡至 Win32 都需要掌握一些学习技巧。

最后，Win32 是真正意义的抢先式多任务、多线程以及线性寻址内存管理（平坦内存——flat memory）。Windows 95 的新界面使你不得不重新斟酌应用程序“外观和感觉”，这通常暗示必须修正以前对用户交互进行支配的逻辑。

1.4　Windows 的硬件需求

抢先式多任务是 Win32 最具有吸引力的一个方面。发行 Windows 第一个版本的时候，几乎所有微处理芯片都是建立在 8088/86 的基础上的。Microsoft Windows 的 1.x 和 2.x 版本紧密依赖于实模式体系，几乎没有办法能够跨越这种体系内部固有的局限。随着 3.0 版本的发布，Windows 逐渐转向 Intel 的保护模式 16 位和 32 位微处理芯片。更近的 3.1 和 3.11 版本不再提供对老式的实模式的支持，那种模式已经太陈旧了。随着 Windows 的发展，以前的 8088/86 处理器已被完全淘汰。Windows for Workgroups 3.11 推出以后，80286 微处理器也面临着相同的命运。无论 Microsoft Windows 95 还是 Microsoft Windows NT，它们都是专门为 80386 处理器以及更高档的处理器设计的；换句话说，它们针对的都是 32 位保护模式。

1.4.1　Intel x86 微处理器家族

为了真正理解 16 位和 32 位 Windows 版本是如何实现多任务机制的，让我们先来看一看 Intel 微处理器的内部结构。iAPX86 家族的所有产品——8088，8086，80286，80386，80486 和 Pentium——最初都要用实模式进行自举。对于 8088 和 8086 处理器来说，这是唯一的选

择。为什么更高级的微处理器也要这样做呢？这完全是出于对整个产品系列兼容性的考虑。表1-3描述了iAPX86产品家族的不同特征。

表 1-3　在个人计算机中使用的 Intel iAPX86 家族的处理器

iAPX86	自然模式	仿　真	RAM	虚拟内存	段
8088	实模式	无	1MB	不支持	64K
8086	实模式	无	1MB	不支持	64K
80286	保护模式 16 位	实模式	16MB	1GB	64K
80386	保护模式 32 位	实模式，保护模式 16 位	4GB	64TB	4GB
80486	保护模式 32 位	实模式，保护模式 16 位	4GB	64TB	4GB
Pentium	保护模式 32 位	实模式，保护模式 16 位	4GB	64TB	4GB

在这儿，我们有必要解释一下表1-3中“虚拟内存”栏的含义。80286系统的值是1GB，这意味着286处理器可以在这样大的一个内存区域内寻址。1GB这个值很容易就可以推算出来：我们只需要注意到LDT和GDT的每个描述符最多只能访问长度为64K的段。这样一来，64K×8192个描述符×2个表，结果就是1GB。

正如大家在表1-3中看到的那样，iAPX86家族的所有成员都是先从实模式开始运行的，以后再根据安装的操作系统改变运行模式。MS-DOS是完全以实模式为基础的。所以，MS-DOS不能利用高级处理器提供的任何一种增强特性，比如虚拟内存和大内存寻址空间等等。

1. 16位实模式

对实模式进行管理的规则是比较简单的（特别是和保护模式比较起来），80286，80386，80486和Pentium处理器都可以使用实模式。在实模式里，寻址空间被限制在1MB以内，这是由于地址总线的宽度为20位（2^{20}=1MB）。实际寻址是通过“段地址：偏移地址”对来实现的。通过成对使用不同的值，就可以访问1MB内存内的任何位置。例如，通过下面这张表可以看出：不同的“段地址：偏移地址”对可能指向内存的同一个物理位置。

“段地址：偏移地址”对	物理位置
1000：40	10040
1001：30	10040
1002：20	10040

对于这种寻址方案来说，一个重要的注意事项在于，它没有提供任何一种内存保护机制。如果想实施一个有效的多任务运行环境，内存保护是不可缺少的。上面这三个“段地址：偏移地址”对明确揭示了这个概念。

通过图1-3，我们可以看到Intel处理器是如何利用20位总线来执行地址引用的。

2. 16位保护模式

80286处理器设计工作于16位保护模式下。实模式和保护模式的根本区别在于用于访问内存位置的逻辑不同。即使在这种情况下，我们看到的也是一对16位值——“段地址：偏移地址”。

就像刚才看到的那样，在实模式下，我们获得了一个由20位组成的基准地址。整个操作简单、快速，这就是MS-DOS系统性能比较高的原因。相反，在16位保护模式下，一个地址

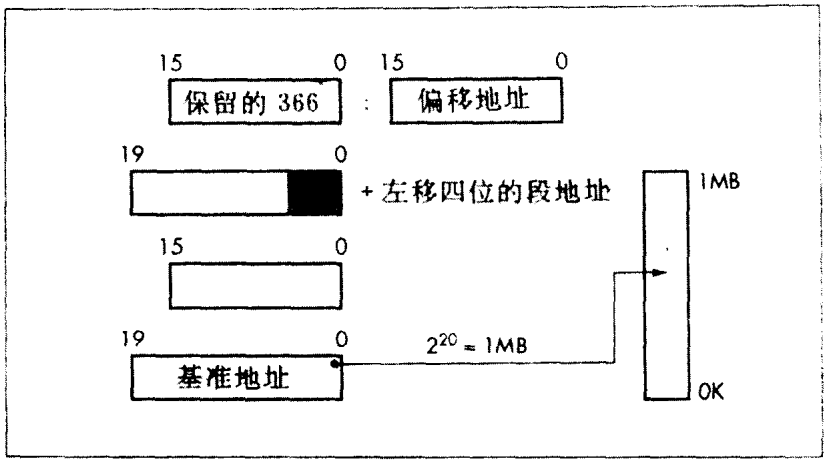


图 1-3　运行于实模式的 Intel iAPX86 处理器体系的寻址方案

的“段地址”部分被系统解释成三部分不同的信息：依次占用 2，1 和 13 个二进制位。

开头两位称为 RPL（请求符优先级，Requestor Privilege Level），它显然有四种不同的二进制组合：00，01，10 和 11。RPL 指示处理器和操作系统对正在执行的处理进行检查，看看它们的优先级之间是不是有不一致的地方。这是对系统执行的处理进行保护的一种简单方式。

段地址的第二个部分只有一个二进制位，我们把它叫作“表指示符”(Table Indicator，或者 TI)——可以是 0 或 1。表指示符的工作原理就像铁路上用的一个扳道叉。当它的值为 1 的时候，就表明更进一步的信息可以在 LDT（局部描述表，Loacal Description Table）里找到。否则，假如为 0，就会到 GDT（公共描述表，Global Description Table）里寻找信息。

段地址中的剩余 13 个二进制位叫作“选择符”(Selector)。选择符起着对表指示符生成的表进行索引的作用。描述表是一个对象，它不能超出 64K 的长度限制，因为它是由 8192 个描述符组成的，每个描述符有 8 个字节长。选择符也是一个对象，它有 13 个二进制位的长度。所以，一共有 8192 种可能的组合（$2^{13}=8192$）。一旦表指示符已经选中了一个表，就可以通过读取选择符访问对应的描述符。无论 LDT 还是 GDT 都是系统内存的一部分，和普通的内存段一样，它们的最大长度也不能超过 64K。有两个寄存器（LDTR 和 GDTR）总是随同当前使用的 GDT 和 LDT 一起载入的。

这些寄存器的长度可以达到 40 位（24 位用于地址，16 位用于描述表的尺寸）。LDT 和 GDT 的结构是类似的，但是操作系统对它们的用法却是各不相同的。OS/2 1.x 是真正利用了 16 位保护模式的唯一一种操作系统。在 OS/2 1.x 里，GDT 的任务是对和标准操作系统组件（系统内核和设备驱动程序）有关信息和引用进行跟踪，并且为每次活动的操作保留一个描述符。为了进行比较，大家可以把一个 LDT（或者 GDT）想象成一幢 8192 层的摩天大楼。编入一个段地址的选择符可以想象成电梯间的按钮，为了到达希望的楼层——或者说描述符，必须按下一个对应的按钮。如图 1-4 所示。

每个描述符的头两个字节未能在 80286 处理器中使用。接着的一个单字节叫作“存取字节”(access byte)，它描述了正在引用的内存区域的一些特征。这两个八位二进制包含了我们

试读结束，需要全本PDF请购买 www.ertongbook.com