



DOS 6.X



高级编程

肖力 编著

人民邮电出版社



7.23/6.6
XL/1

DOS 6. X 高级编程

肖 力 编著

人民邮电出版社

0030697

内 容 提 要

本书介绍在 386 和 MS-DOS 6.0 以上的微机系统上充分利用硬软件资源进行高级编程的各种技术。

全书分以下四部分内容:第一部分介绍硬件基础和 DOS 知识,分别讲解了 80386 的硬件特性、硬件调试手段、DOS 6 和 EMS 提供的调用。第二部分介绍 386 的实模式与保护模式互相切换技术和如何使用扩充内存和扩展内存技术。第三部分介绍直接存取、访问 CD-ROM 和使用 VESA 技术,讲解查询和事件驱动方式的鼠标程序、设备驱动程序和 TSR 程序的结构和设计方法。第四部分介绍 Windows 中的 DOS 程序与 Windows 通信的方法。

本书从基础知识讲到高级编程,内容丰富、叙述详尽,有大量实例 程序可供读者剖析和调用。

本书可供微机软件开发人员参考也可作为大专院校计算机和应用专业高年级师生和研究生的教学参考书。

35340/10

DOS 6.x 高级编程

肖 力 编著

责任编辑 吕晓春

*

人民邮电出版社出版发行

北京朝阳门内南竹杆胡同 111 号

北京顺义兴华印刷厂印刷

新华书店总店科技发行所经销

*

开本:787×1092 1/16 1996 年 4 月 第一版

印张:42.25 1996 年 4 月 北京第 1 次印刷

字数:1064 千字 印数:1—4 000 册

ISBN 7-115-05981-0/TP·268

定价:45.00 元

前 言

随着计算机技术的发展,计算机的应用越来越广泛。它不仅已深入到工业、农业、商业、金融、医学、国防、科研等各领域,而且已进入人们的日常生活。可以预见,计算机技术将在我国国民经济建设和人民生活当中发挥越来越大的作用。

人们在使用计算机时,除了要了解计算机系统的基本硬件结构外,还必须熟练掌握计算机的操作系统。操作系统是系统的核心和最基本部分。它是整个计算机系统的控制管理中心。操作系统统一控制、管理计算机硬件和软件;其中也包括对其它系统软件,如编辑程序、编译程序、连接程序等的管理。随着计算机体系结构和使用方式的发展,各种操作系统也随之产生和发展。目前在微型计算机上使用的操作系统有很多种,MS-DOS 是美国微软公司为 IBM PC 机研制的磁盘操作系统,从 1981 年开发了 MS-DOS 1.00 版本以来,至今已发展到 MS-DOS 6.22 版。

为了帮助计算机应用和开发人员更充分地利用计算机的硬件资源,我们编写了这本《DOS 6.x 高级编程》。它是从计算机硬件和 DOS 基础知识讲起,一直讲到高级编程。我们希望它作为微型计算机软件开发人员和大专院校计算机和应用专业高年级师生的参考书,能对大家的学习、工作与研究有所帮助。

对于本书中的疏漏和差错,敬请读者指正,我们不胜感谢!

编者
1996 年 1 月

目 录

第一章 PC 硬件概述	1
1.1 基本寻址	1
1.2 内存类型	2
1.3 寄存器	2
1.4 再谈寻址	6
1.5 端口	6
1.6 硬件中断	7
1.7 定时器	9
1.8 通用 I/O	9
1.9 键盘	9
1.10 显示	10
1.11 磁盘	11
第二章 DOS 编程必备知识	12
2.1 MS-DOS 的装入过程	12
2.1.1 BOOT 程序	12
2.1.2 IO.SYS 总体结构分析	13
2.1.3 SYS-INIT II 中 CONFIG.SYS 文件的处理过程	14
2.1.4 COMMAND.COM 的初始化流程	16
2.2 DOS 应用程序的类型	18
2.2.1 .EXE 文件	18
2.2.2 .COM 文件	18
2.2.3 TSR 程序	19
2.2.4 设备驱动程序	19
2.3 DOS 中断	20
2.4 BIOS 中断	21
2.5 BIOS 变量	22
第三章 对 DOS 的 C 语言编程	23
3.1 指针寻址	23
3.2 访问环境块	26
3.3 输入和输出	27
3.4 中断	27
3.5 中断服务	32
第四章 几个程序例子	34
4.1 ESCAPE	34
4.2 SPACE	36

4.3	EDISP	37
4.4	PRTSCRN	39
4.5	SPYS	40
第五章	DOS 服务	44
5.1	简单的 I/O 服务	46
5.2	磁盘控制操作	51
5.3	文件操作	56
5.4	FCB 文件服务	59
5.5	句柄服务	60
5.6	目录操作	67
5.7	日期和时间操作	69
5.8	IOCTL 操作	71
5.9	其它操作	74
5.10	其它 DOS 中断	80
第六章	ROM BIOS 服务	86
6.1	显示器服务	86
6.2	设备配置服务	92
6.3	读常规内存的大小	92
6.4	磁盘服务	93
6.5	串行口服务	97
6.6	键盘服务	100
6.7	打印机服务	101
6.8	时钟设备服务	102
6.9	BIOS 变量	103
第七章	DOS 的内存管理	105
7.1	DOS6 管理的各种内存	105
7.1.1	常规内存	105
7.1.2	扩展内存	105
7.1.3	扩充内存	106
7.1.4	高端内存	106
7.2	DOS 的内存管理方法	107
7.3	DOS 内存管理调用	107
7.4	直接访问 DOS 的内存管理	110
第八章	DOS 的进程管理	113
8.1	DOS 的进程管理功能	113
8.2	两个重要数据结构:EXE 文件头和 PSP 详解	116
8.2.1	EXE 文件头的结构详解	116
8.2.2	PSP 结构详解	117
第九章	直接存取技术	122
9.1	把文本写入屏幕存储器	122

9.2	中断规则	125
9.3	管理硬件中断	127
9.4	直接键盘存取	127
9.5	访问 CD-ROM	133
9.5.1	MSCDEX-DOS 访问 CD-ROM 的关键	133
9.5.2	寻找 MSCDEX	134
9.5.3	判断一个驱动器是否是 CD-ROM 设备	136
9.6	定时和声音产生	136
9.7	AT 的实时时钟	142
9.8	使用控制杆	144
9.9	并行口	147
9.10	串行口	149
9.10.1	串行口参数	150
9.10.2	直接 UART 存取	150
9.10.3	一个简单的终端仿真器	151
第十章	DOS6 的数据压缩	163
10.1	MRCI 介绍	163
10.2	使用 MRCI	165
10.2.1	检测 MRCI	165
10.2.2	MRCI 请求包	165
10.2.3	使用 MRCI	166
10.3	DoubleSpace 调用介绍	175
第十一章	构造完备的应用程序	177
11.1	Break 异常处理	178
11.2	严重错误处理	183
11.3	哪一种语言最好?	187
11.4	多任务研究	188
11.5	一个简单的程序 HEXDUMP	188
11.6	一个高性能 C 应用程序	197
第十二章	图形程序设计	212
12.1	模式选择	213
12.2	像素表示法	215
12.2.1	CGA 和 HGA 像素地址	216
12.2.2	EGA	218
12.2.3	VGA256 色模式	223
12.3	设置颜色	223
12.3.1	CGA 颜色	224
12.3.2	EGA 颜色	225
12.3.3	VGA 颜色	225
12.4	综合考虑	226

12.5	提高图形性能	239
12.6	Super VGA 编程	240
12.6.1	什么是 VESA VBE	240
12.6.2	使用 VESA	241
12.6.3	关于 VESA 模式	241
12.6.4	设置 VESA 模式	243
12.6.5	控制 VESA	243
12.6.6	关于 VESA 的总结	243
第十三章	鼠标编程	246
13.1	鼠标方式	246
13.2	鼠标屏幕	247
13.3	鼠标光标	247
13.4	鼠标灵敏度	247
13.5	重要的鼠标变量	248
13.6	基本的鼠标命令	248
13.7	一个基本的 C 语言鼠标库	254
13.8	查询鼠标	260
13.9	事件驱动程序设计	270
13.10	图形模式下使用鼠标	284
第十四章	扩充内存前景:EMS	286
14.1	EMS 如何工作	286
14.2	检测 EMS	287
14.3	选择 EMS 命令	288
14.4	维持兼容性	302
14.5	CEMS 程序库	302
14.6	使用 CEMS:DUP	305
14.7	在 EMS 中执行代码	316
第十五章	设备驱动程序	320
15.1	设备驱动程序的结构	320
15.2	装载设备驱动程序	324
15.3	设备驱动程序的类型	324
15.4	字符设备驱动程序命令	324
15.4.1	INIT	325
15.4.2	INPUT、OUTPUT 和 VERIFY、OUTPUT	325
15.4.3	INPUT (NO WAIT)	326
15.4.4	INSTATUS 和 OUTSTATUS	327
15.4.5	INFLUSH 和 OUTFLUSH	327
15.4.6	IOCTL INPUT 和 IOCTL OUTPUT	327
15.4.7	DEVICE OPEN 和 DEVICE CLOSE	327
15.4.8	GENERIC IOCTL	328

15.5	块设备驱动程序命令	328
15.5.1	INIT	328
15.5.2	MEDIA CHECK	330
15.5.3	BUILD BPB	331
15.5.4	INPUT,OUTPUT 和 VERIFY OUTPUT	331
15.6	任选命令	332
15.7	设备驱动程序的开发环境	333
15.8	一个字符设备驱动程序	339
15.9	一个完整的块设备驱动程序	343
15.10	调试设备驱动程序	352
15.11	进一步要考虑的问题	353
第十六章	TSR 程序设计	355
16.1	TSR 的体系结构	355
16.2	TSR 的接口 INT 2FH	356
16.3	WASTE0:一个简单的拦截器	356
16.4	WASTE1:改进版本	358
16.5	WASTE:最后版本	361
16.6	INTASM:一个拦截器开发环境	368
16.7	控制光标大小	379
16.8	关于拦截器的进一步工作	382
16.9	弹出式 TSR 基础	382
16.10	访问 DOS	383
16.11	临界区	383
16.12	上下文管理	384
16.13	TSRASM:一个弹出式 TSR 开发环境	384
16.14	一些弹出式 TSR 范例	411
16.15	如果 TSR 不工作	421
第十七章	80386 保护模式	423
17.1	保护模式的益处	423
17.1.1	访问 4GB 内存	423
17.1.2	虚拟存储	423
17.1.3	地址映射	424
17.1.4	改进的分段机制	424
17.1.5	内存保护	424
17.1.6	进程保护	424
17.1.7	寄存器	424
17.1.8	改进的寻址模式	424
17.1.9	多任务支持	424
17.1.10	硬件测试	425
17.2	分段机制	425

17.2.1	段选择符	425
17.2.2	表	425
17.2.3	保护模式特权机制	428
17.2.4	数据访问	430
17.2.5	代码段的特权级	430
17.3	多任务处理	430
17.4	再论代码段	432
17.5	异常情况	433
17.6	存储器管理	435
17.6.1	存储器管理	435
17.6.2	地址变换	435
17.6.3	页面故障	436
17.7	实模式和 V86 模式	437
17.8	V86 模式下处理中断	438
17.9	切换到保护模式	438
17.9.1	必需的表	439
17.9.2	切换模式	439
17.9.3	设置 TR	439
17.9.4	允许分页	440
17.9.5	返回实模式	440
17.10	PC 机的保护模式	440
第十八章	使用扩展内存	442
18.1	BIOS 调用	442
18.2	分配扩展内存	443
18.3	CEXT 库程序	444
18.4	扩展内存	450
18.5	XMS 内存的种类	450
18.6	调用驱动程序	451
18.7	常用的 XMS 调用	452
18.8	XMS 风格的虚存	454
第十九章	硬件调试技术	457
19.1	8086 的调试手段	457
19.2	80386 的硬件调试	457
19.3	调试寄存器	458
19.3.1	地址寄存器	458
19.3.2	控制寄存器	458
19.3.3	状态寄存器	459
19.4	恢复被中断的程序	460
19.5	BREAK386	460
19.5.1	实现基本功能	460

19.5.2	调试信息	477
19.5.3	使用断点中断	478
19.5.4	注意项	482
19.5.5	细节问题	483
19.5.6	恢复标志处理	483
19.6	使用 C 语言写中断处理程序	484
19.6.1	使用 C 语言的问题和解决方案	484
19.6.2	使用 C 语言实现具体处理操作	489
19.7	较深入的工作	493
第二十章	实模式下访问 4G 字节内存空间	494
20.1	实模式下访问 4G 字节原理	494
20.2	功能库的实现	495
20.3	语言的要求	505
20.4	使用功能库	505
20.5	实例	506
20.6	可能发生的问题	509
第二十一章	DOS 扩展器	511
21.1	关于 PROT	511
21.2	使用 PROT	512
21.2.1	段	512
21.2.2	写一个程序	514
21.3	综合考虑	517
21.4	动态连接模式	517
21.5	调试	521
21.6	确定故障原因	523
21.7	多任务处理	523
21.8	中断问题	524
21.9	如何管理中断	527
21.10	硬件中断	528
21.11	32 位世界中的 16 位工具	528
21.12	程序例子	528
21.13	PROT 的改进	529
21.14	商用 DOS 扩展器	650
21.14.1	兼容性	650
21.14.2	选择 DOS 扩展器	651
第二十二章	DOS 下程序如何与 Windows 联系	653
22.1	Windows 模式	653
22.2	进入 Windows	653
22.3	启动和退出 Windows	654
22.4	其它 Windows 调用	654

22.5	WINOLDAP 功能调用	655
22.6	使用 WINOLDAP	656
22.6.1	检测 DOS 盒	657
22.6.2	剪贴板基础	657
22.6.3	写入剪贴板	658
22.6.4	从剪贴板中读取数据	658
22.6.5	DOS 使用剪贴板的库函数	658
22.7	总结	663

PC 硬件概述

在一本关于软件的书中,第一章就来谈论硬件,这似乎有点不合逻辑。但是,为了有效地对 PC 编程,就必须懂得有关的硬件知识。在下面的几节中,我们将回顾 8086 系列处理器的一些基本的结构特点,并介绍 PC 机的硬件。如果你经常使用的高级语言隐藏了这些细节的话,那么你就会觉得开始的几节特别有帮助。

1.1 基本寻址

8086 型处理器把内存组织成 8 位的字节。如果涉及的数大于 8 位,8086 就把最低有效字节存放在最低内存地址。尽管这听上去很合理,但是在读内存卸出时常常会令人迷惑,因为数值似乎是倒序的。例如,计算机这样存放 B800H 这个字:先存放 00H,然后是 B8H。

Intel 系列处理器采用分段的内存寻址技术。简单地说,一个段就是一个内存区,计算机可以处理多个段。在实址模式(DOS 运行的模式)中,每个段是严格的 64KB 长,一共可以有 65536 个段。段与段之间可以重叠,一个段可以从上一个段的 16 字节后开始。这就是为什么 DOS 不能直接访问大于 1MB 内存的原因($65,536 * 16 = 1,048,576$ 即 1MB)。8086 和 8088(用于 XT 级机器上的处理器)只能寻址 1MB,286,386 和 486 处理器可以支持更大的内存,但 DOS 不能直接访问它们。

段的编号从 0000H 到 FFFFH。既然每个段都是 64KB 长,我们就可以用一个称为偏移量的值来定位要寻址的字节。一个完整的 8086 地址总是包括一个段值和一个偏移量。如果段值是 0040H,偏移量是 0102H,那么地址就被写作 0040:0102。因为相邻段之间间隔 16 个字节(10H),所以 0000:0010 与 0001:0000 表示同一地址。同样,0040:0000 与 0000:0400 是同一地址,0020:0200 也表示这一地址。计算机总是倒序地存放地址。例如,0040:1234 在计算机的内存(以 16 进制表示)中是这样的:

34 12 40 00

程序通过寄存器和常数的组合来形成地址。鉴于此,我们将在考查了处理器的寄存器之后再详细地讨论寻址问题。

1.2 内存类型

用户程序可以使用几种不同类型的内存。就像我们刚看到的,XT 级计算机只能寻址 1MB 内存,这种内存称为常规内存。XT 可以有 640KB 的 RAM,而 I/O 设备、BIOS ROM 和其它系统资源使用其余的 384KB 内存。

当程序需要的内存空间开始超出这个范围时, Lotus、Intel 和 Microsoft 一起建立了扩充内存规范(EMS)。你可以看到 EMS 通常和它后面的版本号一起出现,像 EMS 3.2 或 EMS 4.0(目前最新的版本)。绝大多数的 EMS 系统包括一个或多个 EMS 卡和一个称之为驱动程序的软件。程序可以通过驱动程序请求使用 EMS 内存,可是这种内存在某一时刻只能访问 64KB。EMS 驱动程序把这 64KB 块放在 1MB 地址空间的高端,通常就在紧邻着 BIOS 的下面。通过调用驱动程序,程序可以把 EMS 内存 16KB 的“页”映射到这个 64KB 的块中。

即使 EMS 可以处理的内存多达 32MB,但在某一时刻,程序只能访问有限数量的 16KB 的页。尽管这不太方便,但是扩充内存是 XT 级机器可以使用的唯一的一类附加内存。如果程序使用其它类型的附加内存,那它们就不能在基于 8088/8086 的机器上运行。

EMS 也可以由磁盘来模拟,这不需要额外的硬件,但是与硬件的 EMS 系统相比速度太慢。

在 286、386 和 486 上,1MB 的内存并不是上限。286 可以寻址多达 16MB;386 和 486 可以寻址多达 4096MB(4GB)。从技术上讲,这种扩展内存只能在保护模式下访问。但 BIOS 提供了一个功能,数据可以在任意的两个地址之间相互传递,包括扩展内存中的地址。

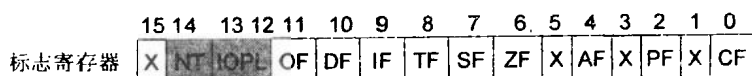
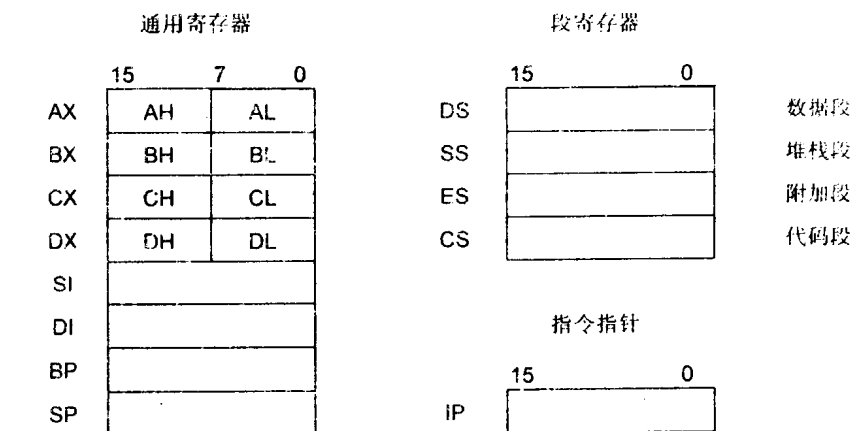
扩展内存规范(XMS)驱动程序用一个巧妙的寻址技术(将在后面讨论)允许 DOS 下的应用程序直接访问扩展内存的将近 64K 的区域。重要的是要注意在保护模式下,常规内存和扩展内存并没有区别,扩展内存只不过是从 1MB 地址开始。用软件可将扩展内存转换成扩充内存,在 286 上这一点不是十分有效,并且不能模拟所有的 EMS 特性。可是,386 和 486 有强大的内存管理功能,EMS 可以完整有效地用软件来实现。

1.3 寄存器

8086 有四个通用寄存器、一个标志寄存器、四个段寄存器、二个索引寄存器、一个堆栈段寄存器和一个堆栈指针寄存器、一个基址寄存器和一个指令指针寄存器。图 1-1 给出了这些寄存器以及 8086 中的其它寄存器;图 1-2 给出了在 386 和 486 中的寄存器。注意 286、386 和 486 有多种专用的寄存器。虽然它们在 DOS 下不是十分有用,但当我们讨论 DOS 的保护模式扩展时将会用到它们。

每个通用寄存器都是 16 位的。也可以把其中的任何一个用作二个 8 位寄存器。例如,AH 和 AL 事实上就是 AX。AX、BX、CX 和 DX 是通用寄存器:

- AX 有时也称为累加器,用于算术运算。
- BX 是唯一能在实址模式中参与构成地址的通用寄存器。
- CX 用于循环或重复操作的计数。



X=保留
 NT=任务嵌套
 IOPL=I/O 特权级
 OF=溢出
 DF=方向
 IF=允许中断
 TF=陷阱
 SF=符号
 ZF=零
 AF=辅助进位
 PF=校验
 CF=进位

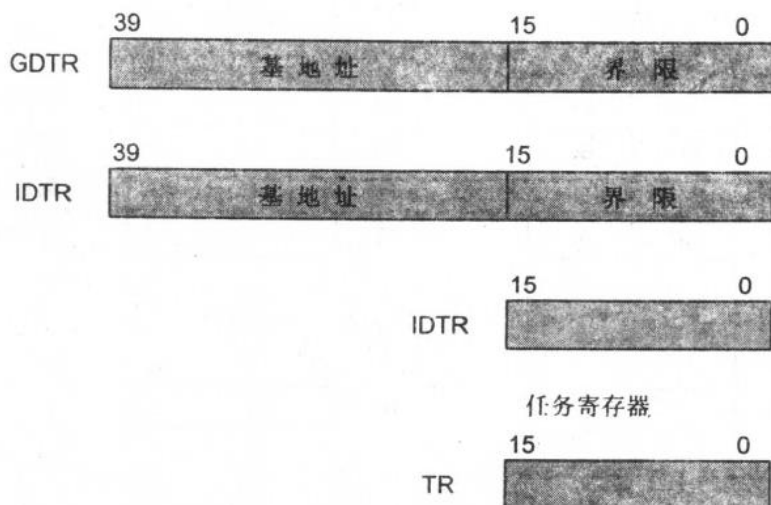


TS=任务切换
 E=模拟协处理器
 M=监视器处理器
 PE=允许特权保护模式

注：带阴影的寄存器和域仅在 286 上存在

图 1-1 8088-8086/80286 寄存器(a)

- DX 指定 I/O 地址,特别是大于 0FFH 的 I/O 地址。它还在乘法和除法指令中与 AX 一起使用。AX 寄存器的两个 8 位寄存器。在图 1-1 中 AX 是 8001H,这就意味着 AH=80H,AL=01H。386 和 486 有同样的通用寄存器,只不过那些寄存器是 32 位的。32 位的寄存器都以字母 E 开头。像图 1-2 当中,EAX 是 72018001H,那么 AX=8001H,



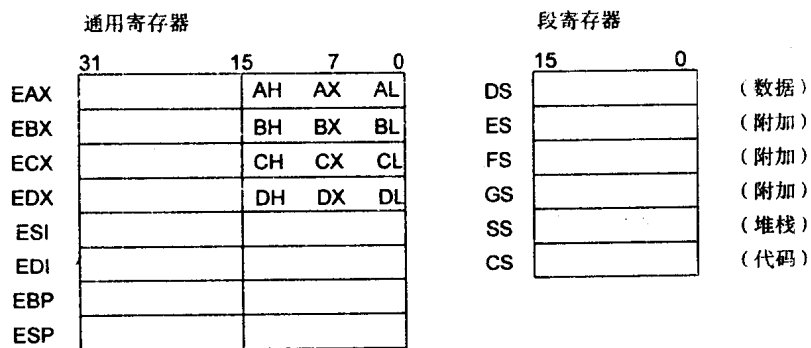
注：带阴影的寄存器和域仅在 286 上存在

图 1-1 8088/8086/80286 寄存器(b)

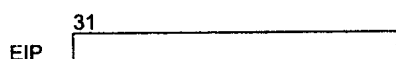
AH=80H, AL=01H。不能直接访问 32 位寄存器的高 16 位。

理论上讲,通用寄存器可以用于任何操作。但是每个寄存器都有某些特定功能。有时对于某种特定的操作必须使用某个寄存器,有时使用某个寄存器会比用其它寄存器更有效。

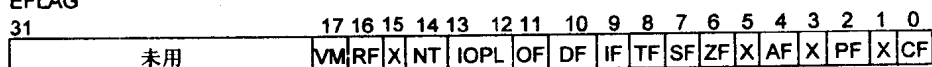
- 段寄存器(CS、DS、SS 和 ES)用来指出当前代码段(CS)、数据段(DS)和堆栈段(SS), ES 是个附加段,用于段间数据传输。这些段像下面描述的那样参与地址计算。386 和 486 处理器还有二个附加段寄存器:FS 和 GS。
- 二个索引寄存器:SI 和 DI。主要是参与形成地址。与通用寄存器不周,SI 和 DI 是不能分成二半的。在 386 和 486 上,可以使用 ESI 和 EDI 来形成 32 位地址,或用 SI 和 DI 形成 16 位地址。
- 8086 系列提供一个堆栈供 CPU 和程序员使用。堆栈作为数据临时存储区,在子程序调用和中断功能中起重要作用。堆栈段寄存器(SS)和栈指针(SP,或 386/486 中的 ESP)给出栈顶地址。用 SS 寄存器作段值,用 SP 寄存器作偏移量,就能确定当前堆栈的栈顶。
- 基址寄存器 BP(或 386/486 中的 EBP)指向栈中的项。在许多方面,它与 SI、DI 寄存器相似,但我们对寻址方式稍加观察就能看出两者之间有明显的差别。
- 指令指针寄存器(IP)给出下一条指令在当前代码段中的偏移量。尽管跳转指令或子程序调用会修改 IP,但用户不能直接修改它。当然,在 386 和 486 中,32 位方式时使用 EIP。
- 标志寄存器包含一系列确定处理器状态的标志位。某一位可以表示上次算术运算是否溢出,另一位可以控制处理器对外部事件(或称为中断)的响应。程序通常用特定的指令设置或清除某一位来改变这个寄存器。



指令指针

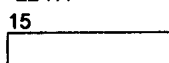


EFLAG

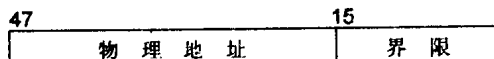


VM=V86 模式 DF=方向 AF=辅助进位
 RF=重新开始标志 IF=中断允许 PF=校验
 NT=嵌套任务 TF=陷阱 CF=进位
 IOPL=I/O 特权级 SF=符号 X=未用
 OF=溢出 ZF=零

LDTR



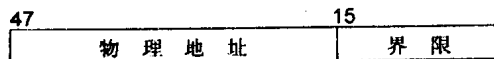
GDTR



TR



IDTR

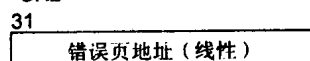


CR0

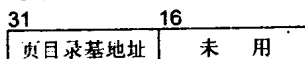


ET=协处理器类型 MP=协处理器保护
 TS=任务切换 PE=保护模式允许
 EM=协处理器仿真

CR2



CR3



注：图中不包含调试和测试寄存器

图 1-2 80386/80486 寄存器