

IBM PC XT AT 系列机

高级程序员 编程指南

H

中国科学院希望电脑技术公司

一九八九年六月

前 言

计算机编程者是当今杰出的发明家。但可惜在他们最出色的成果中，有些新技术却是耗費时间的。调试程序所需时间数十倍于编程时间并非耸人听闻。因编程者考虑不周，导致程序运行时需多次启动也屡见不鲜。而编程者应该熟练掌握的省时之术——了解计算机信息——却很少有人提及，耗費数小时确定一个简单的实据是新程序员的例行公事，令人烦躁得简直要把参考书撕烂。

一个典型的程序员一夜忙下来被CRT弄得眼睛酸痛，打印结果堆积如山，参考书散落遍地。这些书中有：硬件技术参考手册，DOS技术参考手册，语言参考手册，某些芯片的规格表，打印机和键盘的硬件手册，以及十几本计算机书籍。这些资料含有大量信息，要获得这些信息，需要你掌握复杂的规则，耗費大量时间。

由于很多人并不具备过目能诵的记忆力(某种意义上讲从事计算机工作会使你丧失这种能力)，所有这些书都是必不可缺的。因为一个已读过的东西要不时地翻看。初次查找时，也许要用一小时才能确定资料所在。一旦找到，还需花费许多步骤才能从冗长的写给初学者的说明中精选出你所需要的资料：倘若不巧这本手册是用斯瓦西里语写的话，又得花半个下午来翻译，如此等等。可见迫切需要这样一本大部头的书，它尽可能包罗万象，又不含无益于编程的资料，完全按中等程度编写，包含所有IBM微指令，编排合理，查找方便，可到哪里去找这样的书呢？

为此，我汇集多方资料编成这本参考书——教科书，用以满足那些渴望能编制出色的程序，但又缺少时间(或不忍花费600—800美元去购买其它手册和书籍)的人之需要。本书的编排有两个特色。其一：依硬件类型分为章，又按硬件特点细分为节，继而分为短条目，每个条目讨论一个特定的编程任务。例如视频一章的一个小节讨论光标，其中有各种条目说明怎样定光标，改变其形状，打开及关闭光标等等。

其二，每项讨论分为四个部分(有时少些)。开头一、二段讲这基础知识，接着从高级语言编程，中级的BIOS及DOS中断编程，初级的直接对微处理器辅助芯片编程等几方面讨论了所提的问题。并且，每章开头一、二页还讲述了了解本章各节内容所需的背景知识。这些概述旨在复习，但若是初次学习也不妨一读。

高级编程的讨论给出了用高级语言编写的任务。尽管所用概念也同样适用于PASCAL或C语言，给出的实例却是用BASIC写成的。之所以选用BASIC，一方面由于它是计算机领域内广泛使用的语言，每个IBM微计算机的用户都有BASIC；另一方面：Microsoft公司的BASIC在应用IBM硬件方面是其他程序语言所不能比拟的。即使是BASIC语言的初学者，也应能利用很多讨论结果。为扩展BASIC的功用，书中还给出了一些机器语言子程序，附录中讲述了如何把这些子程序编进你的程序中。使用这些子程序，你可方便地做很多事情，如键盘再编程及为单色显示卡增加分页功能。

中级编程讲述了如何使用操作系统提供的中断功能来完成编程任务。中断是功能很强的小例程，它能完成各种计算机中艰巨的工作。如移动光标，续磁盘目录。中断是汇编语言的中坚，中级编程的例子是用汇编语言写成的。但愿越来越多的为高级语言的编译器能够访

中断，使聪明的编程者能完成高级语言本身不能完成的工作，如绝对盘扇区读。因此，人们对中级编程的兴趣不断增长。本书只讨论了PC-DOS(MS-DOS)。若要为iPM-86或UCSDp系统编程，则需另找资料。

最后，初级编程的例子讲述了特定的任务是怎样在芯片一级上执行的。所有IBM系列的微计算机基本上具有相同结构。这是因为所有微机都基于相同的Intel系列集成电路。可用任何语言(包括BASIC)通过I/O端口访问芯片。书中讨论了所有对编程者关系重大的芯片，如定时器芯片，外围接口芯片，中断控制器芯片，CRT控制器，软盘驱动器及管理串/并行口的芯片。尽管IBM不鼓励编程者在初级水平上编程(因为考虑到这种程序在未来的机器上不能运行)，但人们不断发现机器的有些功能不用其他方法进行开发。

并非所有的任务都用三种编程方法进行讨论，因为，有些任务BASIC语言完成不了，有些功能操作系统不支持，有些工作采用初级编程很繁琐(如许多磁盘操作)，而且DOS的编写者已经作了这些工作，且完成得很好，所以不能也没必要对其进行讨论。但多数情况下，都用了三种编程方法，通过三种方法的比较，你可看到高级语言是如何调用中断的，而中断又是怎样在计算机的心脏—芯片上工作的。

这本书看上去也许令只熟悉BASIC或PASCAL等高级语言的人望而生畏。因讲述中，初级编程用的是汇编语言，表面上如同罗塞塔石一样神秘难懂。但对学习汇编语言的人来说，本书是一个理想的伴侣，但不要以为若不懂或不打算学习汇编语言的话，本书就只有三分之一的可用之处。至少有些编译器，如Turbo Pascal和Lattice C将使你建立并应用在中级编程中讲述的操作系统功能。并且事实上许多初级编程的过程可由高级语言完成。为使你能理解汇编语言实例的执行情况，附录D给出了简介，即使你从不使用汇编语言，通过了解它的执行过程，你也将对高级语言执行的过程及出错原因有进一步的认识。

几乎每个小条目都有各自的程序示例。许多只是短短几行。还有一些仅仅是复杂程序的开端，只有少数是完整的可执行程序。我只给出程序中你所需要的片段供你参考，而不是让好似完美的程序充斥全书，也并非要使所有示例都成为典范。事实上，许多程序有意写得不是很完善，以便于你仿效。示例的目的主要不是提供程序模式大全，而是给出思路促使你思考所做工作的方方面面。倘若愿意的话，你可将示例转化成实际程序的开端，并随心所欲地将其扩展成各种形式。由于所有示例都已测试过，它们可使你免犯那些因长时间编程而头昏脑胀时易犯的愚蠢错误。

这本书至少比散文难读些。但我尽量少用术语，并在书后附有计算机基本词汇表。除了一些非常专门的知识外，几乎所有IBM文献中有关编程的知识都已汇入本书。若能无所不包固然是件好事，但那样，本书就将长达千页，且会使人只见树木不见森林。因而对于特殊的编程要求，如软盘控制器的大量编程或AT键盘的重新编程，你还要查阅IBM技术参考手册或芯片厂家的性能表。但99%的程序仅据本书所给的IBM硬件资料即可编写。书中针对同一问题给了不同解决办法，并且比较了它们的优劣。本书还提供了ASCII代码，指令执行时间等各种常用表格，因此仅这一本书就能满足你所有的日常需要。

本书还讲述了许多IBM文献所忽略的细节问题。如某个控制代码是由哪个屏幕输出例程转化的，各类磁盘功能是怎样格式化文件的，有些条目讲述了怎样完成通用编程任

务。这些任务原本不是硬件的专职，但却应用了很多硬件的特性，如实时操作和水平翻卷。本书还用一定篇幅讲述了编程技巧，这些技巧虽非出自权威人士之手，但却能解决编程者所遇到的难题。而现在的情形是：每个编程者都要一遍又一遍地自己弄清这些概念。试想这是一件多么具有讽刺意义的事啊：信息时代的聪明人要象没有莎草纸时那样花费大量时间重新发明轮子。

本书也介绍了各类IBM机器间的差异。以标准PC为基准，当PCjr、XT或AT与之有相异之处时，则加以说明。有一点要说明的是：本书没有谈到AT机及DOS 3.0中涉及多用户系统的特点。这些特点将有专著论述。除了个别注明的例程外，许多程序是针对标准PC编制的：但除非另加说明，它们可在任何IBM微指令支持下正常运行。但有一点限制：本书读者须使用PC-DOS 2.1版本及相应的高级BASIC版本。

祝您编程顺利！

Robert Jourdain

目 录

第一章 系统资源

第一节 确定系统资源状况	(1)
1. 1. 1 访问8255外设接口	(1)
1. 1. 2 找出IBM微机之类型	(5)
1. 1. 3 判定PC-DOS版本	(6)
1. 1. 4 找出图形适配卡的数目及类型	(6)
1. 1. 5 找出磁盘驱动器的数目及类型	(8)
1. 1. 6 找出外部设备的数目及类型	(9)
1. 1. 7 确定RAM量	(11)
第二节 中断管理	(14)
1. 2. 1 8259中断控制器编程	(14)
1. 2. 2 允许/禁止特定的硬件中断	(16)
1. 2. 3 编制自己的中断	(17)
1. 2. 4 完善现有中断	(19)
第三节 程序管理	(21)
1. 3. 1 分配/撤销分配存储器	(22)
1. 3. 2 运行嵌套程序	(24)
1. 3. 3 在一个程序内使用DOS用户接口指令	(26)
1. 3. 4 程序执行完后常驻存储器	(27)
1. 3. 5 调入并运行程序覆盖	(29)
1. 3. 6 将程序由EXE型转接为COM型	(32)

第二章 定时器与发声

第一节 定时器的量数和读数	(37)
2. 1. 1 8253/8254定时器编程	(37)
2. 1. 2 置/读时间	(40)
2. 1. 3 置/读日期	(42)
2. 1. 4 置/读实时时钟	(43)
2. 1. 5 延时操作编程	(44)
2. 1. 6 定时操作编程	(45)

2. 1. 7	实时控制操作编程	(47)
2. 1. 8	用定时器芯片产生随机数	(51)
第二节	发 声	(53)
2. 2. 1	76496发声器的编程方法 (仅PCjr)	(53)
2. 2. 2	演奏	(54)
2. 2. 3	与其它操作同时进行的演奏	(57)
2. 2. 4	报警	(58)
2. 2. 5	演奏音字符串	(59)
2. 2. 6	在进行其它操作的同时演奏音字符串	(63)
2. 2. 7	产生滑音音调	(66)
2. 2. 8	产生音响效果	(67)
2. 2. 9	同时发声	(69)

第三章 键盘

第一节	键盘控制	(71)
3. 1. 1	清除键盘缓冲区	(72)
3. 1. 2	检查缓冲区中的键入字符	(74)
3. 1. 3	等待键入字符但不在屏幕上显示	(75)
3. 1. 4	等待键入字符并把它送向屏幕	(77)
3. 1. 5	立即获取字符	(78)
3. 1. 6	获取键入字符串	(79)
3. 1. 7	检查/设置双态键和换档键的状态	(81)
3. 1. 8	编写通用键盘输入例程	(82)
3. 1. 9	重编键盘中断	(86)
第二节	存取特殊键	(91)
3. 2. 1	Backspace, Enter, Escape和Tab键的使用	(91)
3. 2. 2	组合键的使用: Shift键, Ctrl键和Alt键	(91)
3. 2. 3	双态键的使用: NumLock, CapLock, Ins和ScrollLock	(92)
3. 2. 4	数字副键盘和光标键的使用	(93)
3. 2. 5	功能键的使用	(94)
3. 2. 6	重编单个键	(94)
3. 2. 7	对单个键分配键盘宏功能	(96)
3. 2. 8	建立Ctrl - Break例程	(97)
3. 2. 9	Prtsc键的重定义	(98)
第三节	各种键盘的编码及应用	(100)

3. 2. 1	各种键的使用	(100)
3. 3. 2	扫描码	(101)
3. 3. 3	ASCII码	(101)
3. 3. 4	框图编码	(101)
3. 3. 5	扩充码	(101)

第四章 视频显示

第一节	视频显示控制	(107)
4. 1. 1	6845视频控制器的编程	(108)
4. 1. 2	设置/检验屏幕显示模式	(110)
4. 1. 3	设置字符属性/颜色	(114)
4. 1. 4	设置屏幕边界颜色	(121)
4. 1. 5	清除全部/部分屏幕内容	(122)
4. 1. 6	视频适配器之间的转换	(124)
第二节	光标控制	(126)
4. 2. 1	设置光标于绝对位置	(126)
4. 2. 2	设置光标于相对位置	(129)
4. 2. 3	打开/关闭光标	(130)
4. 2. 4	改变光标形状	(131)
4. 2. 5	读/存恢复光标位置	(132)
4. 2. 6	产生交替光标类型	(134)
第三节	屏幕字符显示	(136)
4. 3. 1	在屏幕上“写”单个字符	(136)
4. 3. 2	在屏幕上“写”一串字符	(142)
4. 3. 3	读给定位置的字符及属性	(144)
4. 3. 4	建立特殊字符	(145)
4. 3. 5	查询块字符所需数据	(149)
第四节	描述点状图形	(153)
4. 4. 1	设置点阵图形三彩色	(155)
4. 4. 2	在屏幕上描绘点阵(对于单色卡、彩色卡、PC jr)	(159)
4. 4. 3	在屏幕上描绘点阵(对于EGA)	(162)
4. 4. 4	确定屏幕上一点的彩色	(171)
4. 4. 5	在屏幕上画线	(175)
4. 4. 6	屏幕区域填充	(179)
4. 4. 7	用块字符描绘图形	(183)

第五节 应用滚动 (SCROLLING) 及分页技术	(185)
4. 5. 1 文本屏幕垂直滚动	(185)
4. 5. 2 文本屏幕水平滚动	(187)
4. 5. 3 切换文本页	(188)
4. 5. 4 文本页间滚动	(192)

第五章 磁盘驱动器

第一节 监视磁盘分配	(194)
5. 1. 1 读文件分配表	(194)
5. 1. 2 确定可用磁盘空间	(198)
5. 1. 3 获取/设置文件长度	(199)
5. 1. 4 恢复盘空间不足错误	(200)
第二节 磁盘目录操作	(202)
5. 2. 1 读/改根目录	(203)
5. 2. 2 创立/删除子目录	(206)
5. 2. 3 读/改子目录	(207)
5. 2. 4 获取/设置当前目录	(208)
5. 2. 5 获取/设置文件的时间和日期	(209)
5. 2. 6 写保护或隐藏文件	(210)
5. 2. 7 读/改文卷标号	(212)
第三节 准备进行文件操作	(215)
5. 3. 1 设置/检查缺省驱动器	(217)
5. 3. 2 建立/删除文件	(217)
5. 3. 3 打开/关闭文件	(220)
5. 3. 4 重新命名文件/移动文件的目录位置	(224)
5. 3. 5 准备进行文件操作	(226)
5. 3. 6 分析来自命令行的信息	(229)
第四节 读写文件	(231)
5. 4. 1 对765软盘控制器和8237DMA芯片进行编程	(232)
5. 4. 2 读/写特定扇区	(241)
5. 4. 3 写到顺序文件	(244)
5. 4. 4 读顺序文件	(250)
5. 4. 5 写到随机文件	(255)
5. 4. 6 读随机文件	(259)
5. 4. 7 在读/写操作之后验证数据	(262)

5. 4. 8 确定/恢复磁盘错误	(262)
-------------------------	-------

第六章 打印机

第一节 控制打印机操作	(265)
6. 1. 1 初始化打印口和再初始化打印机	(266)
6. 1. 2 测试打印机是否联机	(267)
6. 1. 3 解释和排除打印机的错误	(269)
6. 1. 4 两个或多个打印机间的切换	(270)
第二节 设置打印参数	(272)
6. 2. 1 设定文本和图形打印方式	(272)
6. 2. 2 控制行距	(274)
6. 2. 3 控制走纸	(275)
6. 2. 4 控制打印头的位置	(275)
6. 2. 5 设置表格位置	(276)
6. 2. 6 改变打印字形	(276)
6. 2. 7 IBM各种打印机性能比较	(277)
第三节 向打印机发送数据	(280)
6. 3. 1 输出文本或图形数据	(281)
6. 3. 2 文本的右对齐	(284)
6. 3. 3 均衡间隙的文本打印	(287)
6. 3. 4 打印特殊字符	(288)
6. 3. 5 屏幕拷贝	(291)

第七章 输入输出

第一节 访问串行端口	(295)
7. 1. 1 8250 UART芯片编程	(295)
7. 1. 2 串行口初始化	(296)
7. 1. 3 设置当前通信端口	(301)
7. 1. 4 监视串行端口状态	(302)
7. 1. 5 初始化并监视调制解调器	(304)
7. 1. 6 数据发送	(308)
7. 1. 7 接收数据	(311)
7. 1. 8 用通信中断发送/接收数据	(315)
7. 1. 9 查找通信控制码	(318)

第二节 建立设备驱动程序	(320)
7. 2. 1 建立设备首部	(321)
7. 2. 2 建立设备策略	(322)
7. 2. 3 建立设备中断句柄	(323)
7. 2. 4 访问设备驱动程序	(327)
7. 2. 5 检查/分析设备错误	(328)
第三节 使用串行 I/O 设备	(332)
7. 3. 1 读/写磁带录音机	(332)
7. 3. 2 读光笔位置	(333)
7. 3. 3 从游戏端口取模拟输入值	(336)
7. 3. 4 从游戏口取数字输入值	(338)

附 录

附录 A: 二进制、十六进制及内存寻址	(342)
二进制数	(342)
十六进制数	(343)
内存地址和端口地址	(344)
附录 B: BASIC语言的位操作	(345)
附录 C: 汇编语言基础知识	(349)
附录 D: 将汇编例程组合成BASIC程序	(355)
附录 E: 应用ANSI.SYS设备驱动程序	(357)
附录 F: 8088指令集	(358)
附录 G: 80286指令集	(364)
附录 H: IBM PC 名词一览表	(369)

第一章 系统资源

第一节 确定系统资源状况

当输入程序时，首先应了解程序所处的环境：它在哪种IBM微计算机上运行？由哪种DOS版本支持？供其使用的内存有多大？必要的外部设备是否都齐备？了解这些情况的方法有三种：仅仅促使程序使用者去了解这些情况是下策（因为他不一定能得知答案）；尽可能多地从系统版上的DIP开关设置中了解情况则要好多，但这些设置有时仍不能提供足够的情况；因此可采取第三种方法——直接访问有问题的硬件或从BIOS数据区中了解情况。由于从DIP开关设置入手最利于了解情况，因此本节从8255外围接口芯片开始讨论，由该芯片可得知DIP开关设置信息。

一个程序只能以三种方式访问硬件。它可读/写任何与硬件相连的端口地址（只涉及65535个可用口地址中的一小部分），也可读/写多于百万个RAM地址空间中的任何地址。（7.3.0）中较详细地汇总了端口地址。图1-1列出了操作系统及程序在内存中的分布情况。（图见下页）

1.1.1 访问8255外设接口

若要了解外设的当前状态，最好从INTEL8255外设接口芯片入手。该芯片用途广泛。通过它可得知系统板上DIP开关的设置情况。它接收键盘输入的计算机信息，并控制包括8253时钟芯片在内的一系列外部设备。在各类IBM微机中，唯独AT机不用8255；相反，它将本机配置信息连同实时时钟一起存储于由特殊电池供电的芯片上。但AT机仍用8255的口地址进行键盘操作及控制时钟芯片。

8255有三个单字节寄存器，分别称为端口A、B、C，相应的口地址为60H-62H。对这三个端口都可进行读操作，但只能对端口B进行写操作。将PC机端口B的位7置为1，可改变端口A所含信息；同样，PC机位2的设置能决定端口C的低四位内容。XT机位3的设置也有相同的功能。寄存器内容如下：

端口A(60H)

当端口B的位7为0时：

位0-7 PC, XT, PCjr, AT: 键盘接收的八位扫描码

当端口B的位7为1时（对于PC）：

位	PC: 若为0表示无磁盘驱动器
1	PC: 未用
2-3	PC: 系统板上的RAM体
4-5	PC: 显示类型(11=单色, 10=80 X 25彩色, 01=40 X 25彩色)
6-7	PC: 磁盘驱动器数目

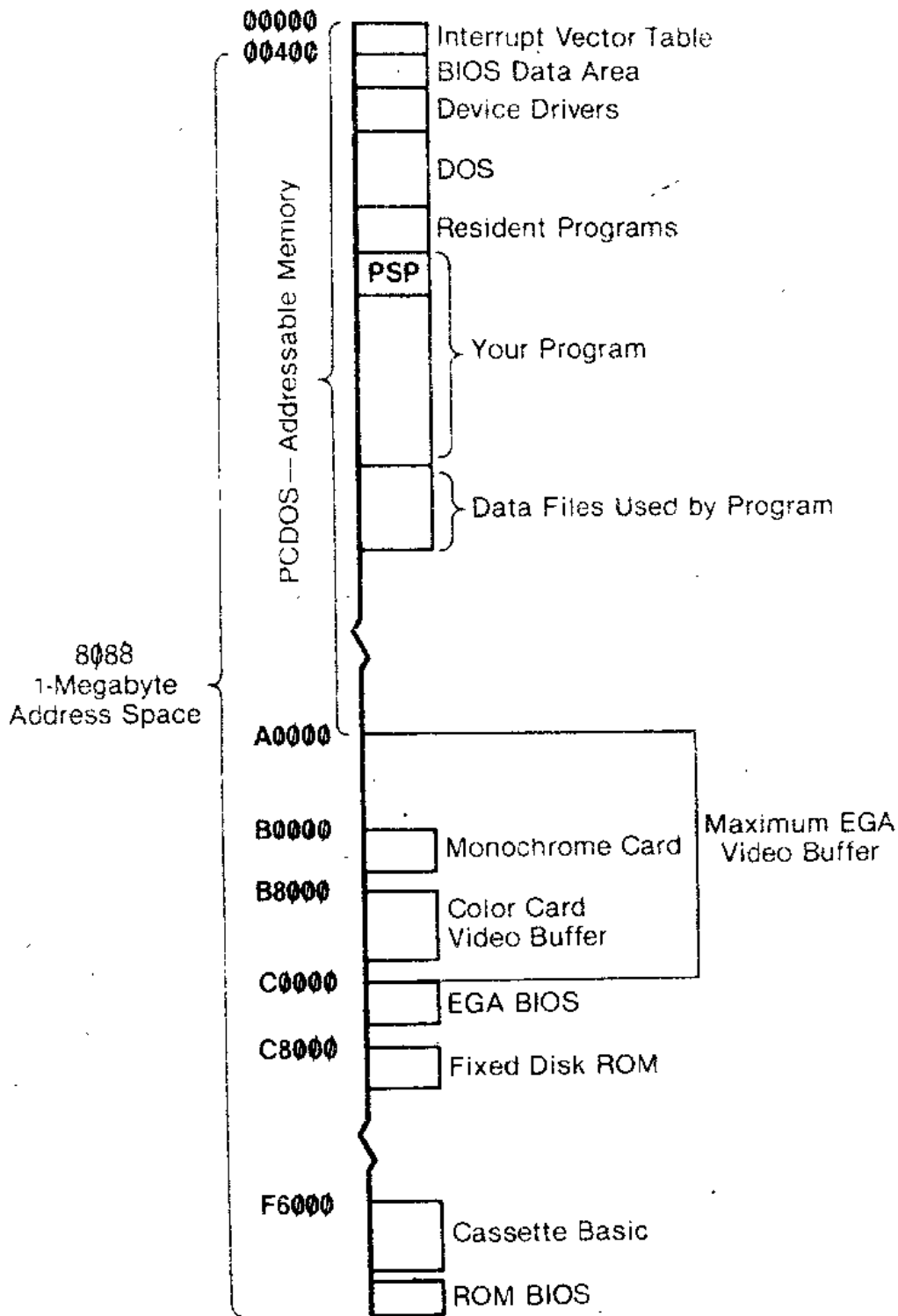


Figure 1-1. Use of the one-megabyte memory address space.

端口B(61H)

- 位0 PC, XT, PCjr: 8253定时器芯片通道2的控制门
- 1 PC, XT, PCjr: 输出至扬声器
- 2 PC: 选择端口C的内容
PCjr: 1为字符模式, 0为图形模式
- 3 PC, PCjr: 若为1表示关闭磁带机马达
XT: 选择端口C的内容
- 4 PC, XT: 若为0表示RAM使能
PCjr: 若为1表示禁止蜂鸣器及磁带机马达
- 5 PC, XT: 若为0表示使能扩展槽出错信号
- 6 PC, XT: 若为1表示使能键盘时钟信号
- 5-6 PCjr: 选择声源
(00=8253芯片, 01=磁带, 10=I/O, 11=76496芯片)
- 7 PC: 选择端口A的内容, 键盘应答
XT: 键盘应答

端口C(62H)

当PC端口B的位2为1或XT端口B的位3为1时:

- 位0-3 PC: 配置开关2的下半部分 (RAM在扩展槽中)
- 0 PCjr: 键盘输入值丢失
- 1 XT: 若为1表示安装了数学协处理器
PCjr: 若为0表示安装了调制解调器卡
- 2-3 XT: 系统板上的RAM体
- 3 PCjr: 0=128K RAM
- 4 PC, PCjr: 由磁带输入
XT: 未用
- 5 PC, XT, PCjr: 8253通道2的输出
- 6 PC, XT: 若为1表示扩展槽错误检验
PCjr: 若为1表示键盘数据
- 7 PC, XT: 若为1表示奇偶错误检验
PCjr: 若为0表示已连接键盘电缆

当PC端口B的位2为0及XT端口B的位3为0时:

- 位0-3 PC: 配置开关2的上半部分 (未用)
- 0-1 XT: 显示类型
(11=单色, 10=80×25彩色, 01=40×25彩色)
- 2-3 XT: 磁盘驱动器数目 (00=1, 等等)
- 4-7 PC, XT: 与端口B的位2为1时相同

注: 寄存器某一位的0相当于DIP开关上的一个“off”(关)设置。

AT机将其配置设置信息连同实时时钟一起存储在Motorola MC 146818芯片上。尽管AT机用相同的口地址控制时间芯片并接收键盘输入数据, 但它并没有8255芯片。

MC146818有64个寄存器：依次标为00-3FH。若要读一个寄存器，应首先将寄存器送入70H号口地址，然后从71H号口地址读出寄存器内容。下面讨论各种配置设置情况，概况如下：

寄存器号	用途
10H	软磁盘驱动器类型
12	硬盘驱动器类型
14	外设
15	系统板存储器（低字节）
16	系统板存储器（高字节）
17	全部扩展存储器（低字节）
18	全部扩展存储器（高字节）
30	超出1M的扩展存储器（低字节）
31	超出1M的扩展存储器（高字节）

高级编程

本书给出了几个访问上述端口的例子。先看一个确定PC机已装磁盘驱动器数目的BASIC程序。在读端口A的高两位之前，必须将端口B的位7置为1。而在执行前又需将该位恢复为0，否则键盘将会锁死，就得重新启动。BASIC不允许以二进制表示的数字，那样会使位类型的分析复杂化。用简单的子程序就能把一个大到255（口地址可达的最大值）的整数转化为一个八字符的二进制字符串。然后用诸如MID\$的字符串函数就能取出有关的位进行分析。有关BASIC位操作的基础知识请参阅附录B。

```

100 A=INP (&H61)           '取端口B数值
110 A=AOR 128               '位7置1
120 OUT &H61,A              '将该字节送返回端口B
130 B=INP (&H60)           '取端口A数值
140 A=A AND 127             '位7置零
150 OUT &H61, A             '将原始值再存入端口B
160 GOSUB 1000              '将其转化成二进制字符串
170 NUMDISK$=RIGHT$(B$,1)  '取第0位
180 IF D$="1" THEN NUMDISK=0:GOTO 230
                                '无磁盘系统

190 C$=LEFT$(B$,2)         '取字符串的高两位
200 TALLEY=0                '保持磁盘计数
210 IF RIGHT$(C$,1)="1" THEN TALLEY=2
                                '分离高位字节
220 IF LEFT$(C$,1)="1" THEN TALLEY=TALLEY+1
                                '加低位字节
230 TALLEY=TALLEY+1         '从1（而非从0）开始计数，得到
                                驱动器数目
1000 ''                     '下面的子程序将字节转化成二进

```

<pre> 1010 B\$ = "" 1020 FOR N=7 TO 0 STEP -1 1030 Z=B-2 N 1040 IF Z >= 0 THEN B=Z: B\$ = B\$ + "1" ELSE B\$ = B\$ + "0" 1050 NEXT 1060 RETURN </pre>	<pre> 制字符串 'B\$ 是字符串 '不断测试更小的2的幂 '将其与字节值相减 '组成字符串 '对每位重复操作 '完成 </pre>
---	---

初级编程

用汇编程序确定磁盘驱动器数目的方法与上例相同，但更简便。同样，必须恢复端口B的原始值。

IN AL,61H	:取端口B的值
OR AL,10000000B	:强置位7为1
OUT 61H,AL	:字节代换
IN AL,60H	:取端口A的值
MOV CL,6	:设置AL右移位数
SHR AL,CL	:将高两位右移六位
INC AL	:从1开始计数
MOV NUMDRIVES,AL	:得到驱动器数
IN AL,61H	:准备恢复端口B之值
AND AL,01111111B	:位7置0
OUT 61H,AL	:字节代换

1.1.2 找出IBM微机之类型

各类IBM微机间存在着兼容性问题。若要使一个程序在各类IBM微机上运行并充分发挥其作用，就必须确定运行程序的机器类型。这个信息可从存储器空间的倒数第二个字节得知。在BIOS ROM内，其地址是FFFFE，并使用如下代码：

计算机	代码
PC	FF
XT	FE
PCjr	FD
AT	FC

高级编程

使用BASIC语言时，只需用PEEK指令读值：

```

100 DEF SEG=&HFF000 '指向存储器顶部64K
110 X=PEEK(&HFFFE) '取倒数第二字节
120 IF X=&HFD THEN '那末是PCjr

```

初级编程

使用汇编语言确定计算机类型：

MOV AX,0F000H	:将ES指向ROM
---------------	-----------

MOV ES,AX	
MOV AL,ES:[0FFFEH]	;取字节
CMP AL,0FDH	;是PCjr?
JENITIALIZE—JR	;若是, 转初始程序

1.1.3. 判定PC-DOS版本

随着PC-DOS版本的更新, 其功能不断增强, 使得一些程序的编制更为简便。若要确保软件在任何DOS版本支持下都能运行, 应使程序只需DOS 1.0所提供的功能。DOS提供了能返回DOS版本号的中断功能。此号可用来检验软件的兼容性。程序能在刚开始执行时便显现出错误信息, 以提醒用户更换版本。当然, 这种情况并不多见。

中级编程

INT 21H的功能调用能读出DOS版本号。调用返回时, “主版本号”(2.10的2)在AL中, “次版本号”(2.10的10)在AH中(注意: 1次版本号以AH形式给出, 而不是1H)。AL值可为0, 指DOS 2.0以前的版本。这个中断将0返回到BX和CX中, 改变了其原值。

;; FIND DOS VERSION:	;; ---确定DOS版本号;
MOV AH,30H	;取10DOS版本号的功能调用号
INT 21H	;得到功能调用号
CMP AL,2	;检验版本是否为2.X
JL WRONG-DOS;	若小于2, 给出信息

1.1.4 找出图形适配卡的数目及类型

应确知一个程序是在单色卡、彩图卡, 还是在EGA(增强型彩图卡)上运行, 是否还有第二个适配卡。[4.1.6]讲述了如何用开关来选择适配卡。位于BIOS数据区0040:0010处的设备状态字节反映了DIP开关1的设置情况, 而该设置决定哪个卡是现行的。理想的情况应是: 若用单色卡, 位5—4将为11; 若是80×25的彩图卡, 则为10; 若是40×25的彩图卡, 则为01; 若是EGA卡, 则为00。但实际上如果EGA存在, 位5—4并不为00, 其具体值由其本身DIP开关的不同设置方式决定。因此, 你必须首先用其他的方法判定EGA是否存在, 若不存在, 则BIOS的数据就会指明现行的是单色卡还是彩色卡。若要检验EGA, 可能试0040:0087处的字节。若为0, 则没有EGA卡; 若非零, 当位3为0时, EGA是现存适配卡; 若为1时, 现行卡是另一个适配卡。

当存在EGA时, 可向6845芯片[4.1.1]的光标地址寄存器写入并回读一个数值并检验其是否相符。用此法可判定是单色还是彩色卡。对于单色卡, 将0FH送入3B4H口以指示光标寄存器, 然后由3B5H口读/写光标地址。彩色卡上的相应端口分别为3D4H及3D5H。当不存在适配卡时, 口地址返回值为0FFH; 但由于此值也可能是寄存器原有的, 所以只检验此值是不够的。

也许你还有两个有关EGA的问题有待解答: 现有多少存储容量? 需连换哪种显示器? 要判定显示器类型, 可测试0040:0087处的位1。当其为1时, 接的是单色显示器; 为0时, 则接彩色显示器。若你的程序用的是350线彩色模式, 则需要判断彩色显示器是IRGB型还是R'G'B'RGB型, 而后者对应于增强型IBM彩色显示器。这些情况可由EGA本身的四个DIP开关的设置情况得知。当调用INT10H的功能12H时, 这些设置信息就返

回到CL中。若低4位为0110，则为增强型彩色显示器。此功能还给出EGA的存储容量。若BL返回值为0，则有64K存储量；为1则有128K；2表有192K；3表全部256K内存。

高级编程

这些程序段检验了当前显示器类型及模式，并判定了主机用的是何种显示适配器。

```
100""                                     '确定现行适配器；
110 DEF SEG=&H40                         '指向BIOS数据区
120 X=PEEK(&H87)                         '检验是否为EGA
130 IF X=0 THEN 200若无EGA，前跳
140 IF X AND 8=0 THEN...                 'EGA取得控制权
200 X=PEEK(&H10)                         '取设备状态字节
210 Y=X AND 48                           '得到位4 及位5的结合值
220 IF Y=48 THEN...                     '是单显(00110000)
230 IF Y=32 THEN...                     '为80X25图形模式(00100000)
240 IF Y=16 THEN...                     '为40X25图形模式(00010000)
```

以下是当EGA或彩色卡为现行卡时检验单色卡的例子。若你端口地址H3D4和H3D5，该程序也可用于检验彩色卡。

```
100""                                     '确定单色卡是否存在；
110 OUT &H3B4,&HF                       '确立光标寄存器地址
120 X=INP(&H3B5)                         '读该地址并存储
130 OUT &H3B5,100                       '将判决值送入寄存器
140 IF INP(&H3B5)<>100 THEN...           '若值相同则单色卡存在
150 OUT &H3B5,X                         '若单色卡存在则恢复原值、
```

初级编程

下面的例子与上述BASIC程序完成的功能相同。

确定现行适配器：

```
MOV AX,40H                               '将ES指向BIOS数据区
MOV ES,AX
MOV AL,ES:[87H]                           '检验EGA是否存在
CMP AL,0
JE NO-EGA                                '若004H:0087H是0，没有EGA
TEST AL,00001000B                         'EGA存在，测试位3
JNZ EGA-NOT-ACTIVE                       '若位3为1，EGA不是现行的
[EGA-NOT-ACTIVE:MOV AL,ES:[10H]          '取视频状态字节
AND AL,00110000B                         '隔离位4 及位5
CMP AL,48                                '是单色卡吗？
JE MONOCHROME                           '若是，跳转
```

否则为彩色卡

假定为单色卡，确定是否装有（非现行的）彩色卡。

已安装彩色卡（非现行）否？