

E&C

1992

●一九九二年 ●总期第85期

4

電子

ISSN 1000-7107

479

與電腦



• ELECTRONICS AND COMPUTERS •



- | | |
|---------------------|----------|
| ① 标准集成电路数据手册 运算放大器 | 平: 27.00 |
| | 精: 32.00 |
| ② 标准集成电路数据手册 PAL 电路 | 平: 56.00 |
| | 精: 61.00 |
| ③ 标准集成电路数据手册 TTL 电路 | 平: 35.00 |
| | 精: 40.00 |
| ④ 国内外功率晶体管实用手册 上册 | 平: 29.00 |
| | 精: 35.00 |
| ⑤ 国内外功率晶体管实用手册 下册 | 平: 63.00 |
| | 精: 68.00 |
| ⑥ 传感器敏感元件大全 | 平: 58.00 |
| | 精: 64.00 |

计算机病毒大事年表

北京理工大学 邓集锋 编译

- 从二次大战末到现在计算机与计算机病毒发展史上的主要大事记。
- 1945 世界上第一台电子数字计算机—ENIAC问世,二战期间科学家们用它进行弹道学方面的计算,这台计算机有100英尺长,耗电400瓦,但是它的数据处理容量却不及现在的一台膝上计算机。
- 是年,普林斯顿高级研究所的电子计算机项目中心聘请纽曼(Neumann)担任主任。正如我们现在所知道的,这有效地推动了计算机技术的发展。
- 1949 纽曼发表了《复杂自动机的理论与结构》一书,第一个提出复制机制的理论。
- 50年代 计算机在美国、欧洲及世界其它各地得到迅猛的发展、逐渐走向商品化。
- 1953年 世界上第一台商用计算机—Univac I,由斯伯雷·兰德(Sperry Rand)公司研制成功。
- 1954 贝尔电话系统公司公布了有关容许电话盗用者进入电话系统的频率值的技术细节。
- 1959 美国电报电话公司(AT&T)贝尔研究所的程序设计员编制“磁心战”游戏程序,以攻击对方的程序和破坏数据为乐。其它的研究机构 尤其是麻省理工学院(MIT)人工智能实验室和施乐公司(Xerox)帕洛阿尔托(Palo Alto)研究所的研究人员也开始编写磁心存储器杀手程序。
- 1965 狂热的“超级计算机迷”,特别是那些可以自己独立编制程序的人,开始被人们称作“恶作剧者”。具有突出特征的计算机文化随着计算机技术的发展逐渐形成。
- 1966 两个美国大学生编写一段可以自拷贝的程序,很可能它就是第一代病毒之一。该程序产生的病毒足以使计算机系统崩溃。
- 1969 美国国防部远景规划署开发出世界上第一个大型计算机网络(ARPANET)实现全美国防工程项目的计算机联网。该网络经历了二十多年不断发展的进程,同时也给“恶作剧者”提供更多的机会。
- 1971 计算机犯罪逐步升级,从最初的破坏数据发展到修改信用证利率,篡改库存记录以及挪用资金。
- 《绅士》(Esquire)杂志公布了有关电话盗用活动的第一批详细资料。此后,各出版物相继发表更详尽的消息。
- 1972 电话盗用者迫不及待地行动起来,不断发现免费进入电话系统的新途径,天才的“恶作剧者”则专门研究未经授权进入计算机系统存取的技术。
- 1973 有关证券融资公司假券案的第一份资料公诸于世。计算机犯罪显现出难以估量的潜力。在过去的十年中,计算机程序员伪造了64,000张彩票。
- 1974 施乐(Xerox)公司研制出第一份自复制代码。该公司研究机构的研究人员还有效地消除了“磁心战”游戏程序的破坏性。
- 1975 早期的微型机出现。
- 1976 赤旅(Red Brigade)恐怖组织对欧洲计算机设备系统连续发起十次突袭行动,第一场计算机风暴席卷全世界。
- 美国最高联邦政府机构向参议院一个委员会发出呼吁,希望注意计算机犯罪的严重性。
- 1977 Thomas, J. Ryan的科幻小说《Adolescence of P-1》轰动美国科普界。作者幻想出世界上第一个病毒,可以从一台计算机到另一台计算机传染流行,最终控制了7000台计算机的操作系统,酿成一场灾难。
- 1978 洛杉矶一家银行由于罪犯在电话中盗用口令和银行帐号而失窃现金一千零二十万美元。
- 1979 阿里桑那州首先实施计算机犯罪法。
- 1980 P C机日渐成为市场上的主要机型,价格便宜,发展潜力很大。施乐(Xerox)公司研究所研制蠕虫程序,可以防止破坏数据。
- 1981 据估计,本年度计算机“恶作剧者”人数翻了三番。
- 1982 设在蒙哥马利县的加利福尼亚州计算机图书情报系统中发现“逻辑炸弹”病毒。
- 1983 电影《战争游戏》(War games)开播,恶作剧者的行动愈演愈烈。克恩·汤姆森(Ken Thompson)在一次演讲中首次把自复制机制的秘密公诸于世。他是一位参与研制UNIX操作系统的软件工程师,也是全美制造业协会会员。
- 1984 《科学的美国人》(Scientific American)杂志公布了“磁心战”游戏程序的详细内容。此后,编制病毒的资料开始广泛传播。
- 有关自复制程序及其对系统破坏性的第一份科学文件曝光。加州的弗莱德·科恩(Fred Cohen)教授在一次计算机安全学术讨论会上首次提出“病毒”的概念,并针对病毒的破坏力进行实验演示。
- 1985 早期的病毒程序在美国大学里大肆肆虐。这使得库克·蒙斯特(Cookie Monster)之类的恶作剧者大为兴奋。
- 在日本,中间派政治集团通过侵袭20套计算机设备系统,致使可供一千万旅客往返的铁路系统处于瘫痪状态,从而战胜了左翼势力。
- 巴基斯坦智囊(Pakistani Brain)病毒在巴基斯坦的拉瓦尔(Lahore)市诞生,随后通过非法拷贝软件的途径在世界范围内传播。
- D. G. 伯勒森(Donald Gene Bursleson)被得克萨斯州的福特·瓦斯(Fort Worth)证券交易公司解雇,原因是他移植了一个蠕虫程序进而毁坏公司数据。
- 1986 第一个诱发全世界感染的病毒出现。
- 1987 勒海(Lehigh)病毒问世,并开始在美国大学里广泛传播。 (下转封三)

计算机病毒大事年表 (续)

北京理工大学 邓集锋 编译

在美国本土,巴基斯坦智囊病毒首先在宾夕法尼亚大学和怀俄明大学发现,并以强劲的势头传染开去。其它病毒也迅速繁殖。

圣诞节前夕,圣诞树病毒自德国穿越大西洋在美国传播,使350,000台IBM公司的国际电子信息网的网络终端染毒。这一大型计算机网络最终因大量复制的连锁信件而不堪重负,导致瘫痪。

1988 病毒侵袭大肆流行。以色列希伯来(Hebrew)大学的计算机被犹太人(Israeli)病毒感染。该病毒专门在巴勒斯坦国亡国日毁坏计算机数据。

乔治城大学经受了巴勒斯坦智囊(Pakistani Brain)病毒连续七个月的侵袭。总部设在罗得(Rhode)岛的《幸福》(Providence Journal)杂志社的300台计算机全部被感染。

评分(Scores)病毒感染NASA和美国其它政府机构的计算机,继而扩散到美国国会办公系统和成百上千个包括波音航空公司、福特航空公司在内的计算机系统中。稍后一点,福特航空公司的计算机系统又被nVir病毒感染。

MacMag病毒于是年3月2日—Mac II系统问世的周年纪念日发作。专家们估计它传染了250,000套计算机系统。

台湾发现首例计算机病毒侵权案件。

汉堡的计算机骚乱俱乐部(Computer Chaos Club)声称已将其研制的病毒感染NASA系统。尔后,该病毒的研究者在巴黎被拘捕。

8月,苏联政府机构的计算机网络发现病毒入侵。

9月,与日本电气公司联机的日本最大的计算机网PC—VAN网发生计算机病毒入侵用户计算机事件。

第一件著名的专利商用软件染毒案见报。MacMag病毒侵入爱尔德斯(Aldus)公司的“Freehand”图形程序并大量传播。这年晚些时候,“Freehand”的改良版—β测试版的程序染上nVir病毒,不过很快控制了传染。

全美软件开发委员会呼吁软件立法和研制反病毒软件。

计算机病毒工业协会(CVIA)收集了最为详细的病毒感染资料,并为反病毒软件的研制与营销建立一套标准。

D. G. 伯勒森因第一起病毒犯罪案件被判刑。

有史以来世界上最大的病毒侵袭事件发生于11月2日,导致美国最重要的两家计算机网络INTERNET和ARPANET的数千台计算机瘫痪或半瘫痪。其中有相当一部分计算机系统涉及到美国国防工程项目。

国会报告把计算机病毒部分放在显要位置,并说“美国在工业、医学、商业、教育和家庭生活等方面的计算机病毒蔓延状况表明,国会对于这一事态的发展引起关注是非常适宜的。”

到此为止,已可识别30多种病毒及其衍生的许多病毒变种。大约40个美国主要的工业企业确认有染毒经历。

美国《商业周刊》(BusinessWeek),《时代》周刊(Time)及其它新闻媒体开始长篇大幅报导病毒的威胁性,但仍然有许多计算机杂志对此不屑一顾。

反病毒软件广泛流行。

病毒变得“驯服”,并被有效地用于增强计算机系统的免疫功能。

病毒往恶性化方向发展。现有病毒版本经恶作剧者改良后,具有更大的破坏力。

1989 日本《朝日新闻》评选1988年十大科技新闻,“计算机病毒侵入日、美、苏计算机网络”以第二条重要新闻的显著位置入选。美国科学家也把“计算机病毒在欧美流行”作为88年重要国防科技十大新闻。

年初,大连市统计局计算中心所有M24微机全部感染小球病毒。计算机病毒很快在中国大陆许多地方蔓延。

三月初,重庆西南铝加工厂计算中心研制出国内第一份反病毒程序,可对染毒盘进行消毒处理。

9、10月间,上海、福建等地开展计算机病毒情况调查。结果表明,有50%以上的计算机染毒,机型大都是IBM PC、XT、长城0520系列及各种兼容机。

台湾计算机厂商警告说,台湾现有七十万台PC机中约有80%已潜伏病毒。

10月13日,“哥伦布日”(Columbus Day)病毒在全球发作,引起巨大恐慌。日本通产省着手制定对付此病毒的方法。

11月3日在昆明市召开的“第四次全国计算机安全技术交流会”上,专门讨论了“计算机犯罪,计算机病毒研究”。

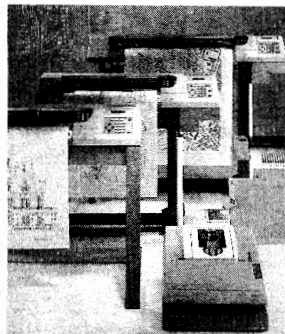
11月份《中国日报》报道,我国40多万套计算机系统中约有十分之一染有病毒,而且沿海各省市病毒蔓延范围比内地要广得多。

年末,美国人鲍伯因向全世界发放带毒的“艾滋病资料”软件包进行敲诈活动,被联邦调查局的特工人员逮捕。

中国大陆许多高等院校和科研机构纷纷推出病毒检测、消除和免疫的软件。其中以公安部的SCAN系列查毒软件和KILL系列消毒软件最为流行。

计算机病毒在种类不断增加的过程中迅速蔓延到全世界,成为威胁计算机安全的主要敌人。这促进了计算机安全领域中的一个重要分支—计算机病毒学的形成。

计算机技术的高度发展为人类提供高度的自动化和现代化。计算机网络及联网规模的不断扩大并向着现代化方向发展,使人类社会的信息交流得以更高速、更高效、更广泛的形式进行。因此,充分地利用人类信息资源这一社会财富成为迫切需要;相应地,社会的法律制度和伦理道德也需要进一步完善。总之,当代社会以计算机技术为基础的计算机文化文明已经形成。



一九九二年

总期第85期

電子與電腦

ELECTRONICS AND COMPUTERS

目 录

· 综述 ·

引人注目的微型机 戴伟高(2)

· PC 用户 ·

用 24 针打印机打印 CAD 图形 卢耀志(4)
编制菜单程序的辅助程序 @. COM

..... 张 红(7)

格式文件与 SET FILTER TO 命令的结合

..... 麦红波(8)

Azusa/2708 病毒与介质描述 何悦荣(9)

MS FORTRAN 程序的菜单设计技术

..... 曹国钧 王 健(10)

如何制做病毒标本 梁宇翀(10)

用 dBASE Ⅲ MEMO 字段开发题库系统

..... 陈卫平(11)

关于《dBASE Ⅲ 过程文件加密的发现》质疑

..... 张维仁(11)

在单台驱动器上如何作软盘备份 ... 杨栋林(12)

一种不打印数值零的方法 曹和平(12)

交互式的 DEL 命令 莫 琳(12)

· 学习机之友 ·

为 APPLE DOS 增加多目录功能 秦燕军(14)

验证四色猜想的 BASIC 程序 陈庆祥(15)

磁盘不抹掉文件的 35 轨改 40 轨方法

..... 钱仕宏(16)

拆解 C—WORDSTAR 五笔字型 陈晓乐(16)

BASIC 中“宏代换”的实现 许 斌(17)

改进的 UNNEW 程序 张 亨(18)

CEC—I 中华学习机磁带游戏软件的解密

..... 胡发新(18)

也谈 CEC—I 全功能造字 覃 敏(19)

ProDos 磁盘操作系统入门(续) 廖 凯(20)

· 6050 机器语言讲座 ·

第四章 6502MPU 的指令系统 朱国江(22)

· 初级程序员级软件水平考试辅导 ·

操作系统及 PC-DOS 宋丹颖(27)

· 学用单片机 ·

实现单片机最小系统与 PC 机的通信

..... 张培仁(30)

· 学装微电脑 ·

红外线遥控室内温度 易齐干(34)

· 电脑巧开发 ·

Z80 单板机与打印机巧相连 张慧成(37)

降低 MP-1 电脑的功耗 邓文超(40)

· 电脑游戏机 ·

第三章 F BASIC 的画面控制语句

..... 于 春(41)

· 维修经验谈 ·

中华学习机电源故障维修经验谈 ... 汪 晖(44)

袖珍计算机 PC 型 PB 型系统 RAM 扩展模块

常见故障与维修 朱 杰(46)

· 读者联谊 ·

普及型 PC 机个人用户软件交流联谊活动问题解

答(四) 王路敬(48)

封一:绘图仪

封二:病毒大事年表

封三:病毒大事年表(续)

封四:电子工业出版社图书

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部

(北京 173 信箱 邮政编码:100036)

印刷:北京三二〇九厂

国内总发行:北京报刊发行局

国内统一刊号:CN11-2199

邮发代号:2-888

国外代号:M924

出版日期:每月 23 日

主编:王惠民 副主编:王昌铭

责任编辑:施玉新

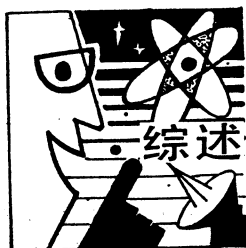
订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:0.95 元



引人注目的微型机

武汉 70005 信箱(430070) 戴伟高

一、引言

在微机运用取得广泛发展的今天,这种家族系列产品不足之处何在!事实上,在许多工业开发项目中,不得不转向单板机、单片机和嵌入式微控制器就是一个很好的说明。但是这些产品毕竟是专用的,它们的软硬件环境远远赶不上 PC 机。高档一点的产品批量上不去,价格就无法下降。再加上目前某些产品一些技术问题还没彻底解决,使用起来也实感不便。那么有没有一种产品,它既和世界上销量最大,软硬件环境最丰富的 PC 机兼容,又能作为单板机、单片机或嵌入式微控制器使用呢?美国 AMPRO 公司 1991 年推向市场的产品“Core Module”是一种成功的努力,被杂志誉为解决嵌入式处理器问题的开拓者,它的产品究竟有那些特色呢?

二、产品的主要特征

要使 IBM PC/XT 转向工业控制,首先要解决的一个问题就是体积和功耗(主要指发热情况)。只有体积小,才有可能谈得上嵌入,否则在某些运用中,不是它嵌入主方,而是主方嵌入它。原始的 PC/XT 底板面积显然太大,而 AMPRO 公司交给我们的“Core module/XT”产品,面积不到手掌那么大,尺寸仅为 $3.6'' \times 3.6'' \times 0.6''$,但却包括了 PC/XT 的全部功能,真可谓是一块大芯片。难怪公司美称它为“Core Module”。这种模块,由于全部采用表面封装的 CMOS 电路,整个功耗仅为一瓦左右,发热情况得到了彻底地解决,免去了许多运用中令人头疼的风冷电扇,但是芯型模块的整个技术内容不仅是体积和功耗。这不算特别新鲜,因为 1988 年,Intel 宣布的 wildcard-88,尺寸达到 $2'' \times 4''$ 。当时第一次拿到这个模块的用户,对已经在他手中的东西还不敢相信,但集成电路技术毕竟发展到了这个程度。

PC 机的总线靠插槽(SLOT)联接其他适配器(俗称卡)。这种结构既不抗振也不可靠,在办公桌上用问题不大,在某些工业环境中问题就比较突出。AMPRO 产品革新地采用了堆垛式(Stack)结构,总线靠堆垛式插头(Stack-through headers)联接。为了保持硬件上与 PC/XT 兼容,它也可连结插槽式 PC 机的各种 I/O 适配卡。以保证各种各样的 AMPRO 适配器没有开发出来之前,用户暂时可使用 PC 插槽式适配卡,或者在优越的环境中直接使用廉价的 PC 卡。这种堆垛式结构板,构造紧凑,牢固,并且具有极大的灵活性,用户

可根据具体情况,配置自己的系统,犹如小孩搭积木。这种结构形式,决定了它既可作为单板,又可作为 PC 微机系统。过去单板或单片机的开发一般在微机上配合开发板进行。即使微小的硬件变动也得把装置拆下,重新在开发系统下仿真。更不用说单板(单片)更新升级这样的工作。AMPRO 公司将在 1991 年全面推出的 Core module/XT, Core module/286 和 Core module/386 SX,因为和 IBM 全兼容,所有外设接口都有现成的软硬件,这种资源丰富得几乎能满足你所想解决的一切课题的大部份,而您的精力只要全部集中到你所想解决的特殊课题上。这样就大大减少了工作强度,便于早出成果。某些改动和调试甚至能方便地在现场进行。因为 AMPRO 公司为自己的产品配上了“Private eye”。这种个人视屏只有 1 平方英寸,戴在头上工作起来十分自如,但给人的效果却相当于一台 12 英寸的 720×280 的单显,它和 CGA 卡兼容。RT 公司(反射技术公司)之所以把它称为“眼睛”。看来有一定的道理。就象人的眼睛虽然小,但却能把万物装进去一样。

总之,AMPRO 公司的产品在开发阶段,配上所有的外设,它就构成了一台 PC 系统机,拆去所有外设,它就成了名符其实的嵌入式部件。它的体积和堆垛结构甚至允许人们把它当作一块大芯片来处理,真可谓地地道道的单片机。在网络中可以把它嵌入键盘或显示器就成了专用终端;把它嵌入打印机,就成了共享打印设备;嵌入自控设备,就成了大脑中枢等等,很有发展潜力。

三、第三方产业支援

在恶劣环境下,PC/XT 遇到的另一个问题就是机械式软硬盘,尤其是软驱的故障率问题。近年来工业控制机在这方面作出了一些努力,用得比较多的是靠电池备份的 RAM 盘。但这种方法价格并不便宜,又不宜仿真大容量盘,更谈不上仿真硬盘。同时更换电池,数据的丢失仍然是用户要关注的一个实际问题。近两年来,半导体存储技术的发展,使 flash 存储器走向了实用阶段。这种存储器能象 EEPROM 一样多次读写,周期使用次数高达 10000 多。不同于 EPROM,它可以对扇区进行擦写,价格又比 EEPROM 便宜,周期使用次数,远远大于 EPROM。目前,这种技术还在深入发展。权威预测,再过两年,价格将大大下降。AMPRO 采用这种器件,组成了它的 SSD(固态盘),较理想地解决了工业控制中信息的储存。公司在它的芯型模块上装有

两个字节宽度的芯座,每一个能支持 512K 字节 flash 存储器(如果需要,也可使用其它的 NOVRAM)。通过 mini Module/SSD(微型固态硬盘模块)还能进一步把容量提高到 2M 字节;如果需要,经 stack plane/SSD(SSD 堆垛板),它上面有 16 个字节宽度芯片座,定能满足你的实际需求。特别值得一提的是,为了配合 SSD 的管理,和在 ROM 中仿真软盘操作系统。AMPRO 通过第三方提供了实用软件和适合于 ROM 启动的 DR DOS3.41,它与 MS DOS 3.3 全兼容。同时数字研究公司(DR)还为 AMPRO 开发了最新版本的 DR DOS 5.0。Annabooks 公司的 PROM kit, XT BIOS kit, AT BIOS kit; Datalight 公司的 ROM-DOS, C—thru—ROM 等实用程序都为开发阶段提供了方便。Award ROM BIOS 是保持兼容性的关键,并且有它的独到之处,诸如:启动, SCSI 功能,在板 EMS 4.0 和 ROM DOS 功能。为了满足嵌入式系统的要求,与 PC/XT 不同, Core module/XT 加入了 1K 字节的 EEPROM,用于储存配置信息。

四、各种各样的 I/O 模块

AMPRO 公司为完善它的产品系列,正在积极开发它的各种模块。除上面提及的,已经开发出来的还包

括: Little Board/386(小板 386), Little Board/286, Little Board/PC, Little Board/86, Slot Board/386(槽板 386), Slot Board/286, mini Module/Modem(微型调制解调器模块), Mini Modute/SENFAX, mini Module/CGA, mini Module/EGA, mini Module/ LCD, mini Module/VGA, mini Module/ATDISK, mini Module/FSS, mini Module/AR-CNET, mini Module/488, mini, Modute/mini, Module/488, mini Module/SSP, Stacker/mini Module, Stack Plane/mini Module, Slot Board/SSD, Stack Plane/SSD, mini Backplane/AT, mini Backplane/PC, Stack Plane/PC, Backplane Adapter/PC 以及 ELD 显示控制器和 LCD 电源模块等等。这些产品,有些顾名思义,有些限于篇幅就不一一解释它们的作用和功能了。

五、结束

AMPRO 公司充分利用了近几年发展起来的新技术,革新地采用了堆垛式结构,使自己的产品填补了市场的空白,在保持与世界销售量最大的 PC 机兼容的基础上,使产品的平均故障时间达到 100000 至 200000 小时,从而使自己的产品具有强大的市场生命力。

简 讯

具有十二年发展历史的国家级新型高科技企业——中国计算机软件与技术服务总公司于 1992 年 3 月 3 日至 7 日在京举办了“92 计算机应用成果、新品发布展示会”。展出了四十余项科研、应用、生产和服务方面的新成果、新产品和新技术。主要有:

多用户财务系统、HPCCOM 通讯软件、CV-2 图纸读取系统、“译星”英汉和“汉译”汉英机器翻译系统、汉化的 UNIX SVR4.0 具有销售版权的操作系统等。

中软件公司还开发了 USPS 微机内置式开关电源、SL 单回路调节器、多功能维修测试卡、微机防病毒卡、打印机共享器、轮廓字库、多用户文档检索系统等产品,它以其崭新的面貌和独特的功能展示出了中软公司的雄厚实力,受到社会各界的广泛关注。

欲索取有关详细资料者,请直接与中软公司联系。地址:北京海淀区学院南路 55 号电脑大厦。邮编:100081。

光盘图书通向未来

本刊讯:4月28日上午,北京金盘电子有限公司在人民大会堂举办了光盘图书演示会。

北京金盘电子有限公司是中国第一家以经营、制作电子光盘图书(CD-ROM)为主的中外合资企业。CD-ROM 被称为信息时代的图书,它不仅存储量巨大,而集文字、图像、声音于一盘,可以用计算机非常迅速准确的查找到所需要的信息。因此,它在教育、科研、经济及国防诸方面有着广阔的应用前景。作为一种先进的信息传播媒介,CD-ROM 正在推动着新的出版潮流。

薄一波同志到会并为此项高新技术题词:

“光盘图书,通向未来。”

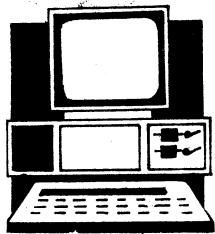
最新 FORTH 版本 PC35

中国 FORTH 应用研究会迄今已成立五年,一贯致力于推广第四代高级计算机语言 FORTH,拥有包括最新版本 PC/FORTH35 在内的多种 PC 及 APPLE 版本的软件以及各种 FORTH 资料,并经常举办函授及各种培训班。

欢迎有志之士及爱好者与我们联系。

地址:北京 西北北三条 10 号

邮编:100034 电话:655661



PC用户

用 24 针打印机打印 CAD 图形

长春 东北勘测设计院物探计算室(130062)卢耀志

目前, Auto CAD 绘图软件包的使用已很普遍, 用 Auto CAD 搞工程绘图很方便、高效。但是它不支持 24 针打印机绘图, 只支持 9 针打印机, 9 针打印机打印的图形质量比较差, 图幅也小, 一般不能用来绘正式的工程图纸, 所以, 若再没有绘图机, 就只好望图兴叹了。有些 24 针打印机如 AR-3240、LQ-1600K 等虽然也能直接在 CAD 状态下绘图, 但也不是真正的 24 针图形, 与 9 针打印机打的一样。本文介绍一种 24 针打印机打印 CAD 图形的方法, 此方法打出的是真正的 24 针图形, 完全可以代替绘图机满足工程绘图的需要, 经过实际使用本人认为, 就是有绘图机的用户, 打印机打印 CAD 图形也有其可取之处: 它操作简便; 绘图时无需人的看管; 不会出现现象绘图机笔不下水、走纸不畅等故障。对于复杂图形, 打印机绘图的效率会更高。程序用 Turbo Pascal 5.0 编写, 笔者毫不保留地将完整的程序奉献给大家, 只要正确无误地输入并编译, 即可运行。

一、方法原理

本方法的基本思路是: 先获得 CAD 图形数据, 再将其转化成点阵图形数据。送到打印机打印。

1. 获得 CAD 图形数据的方法

Auto CAD 的图形文件 *.DWG 是二进制文件, 不了解其结构不易打开取出有用的信息, 幸运的是, CAD 提供的绘图机绘图命令 PLOT, 可以输出至磁盘, 扩展名为 PLT, 且是文本文件, 其格式如下:

```
;;HALOEC1 UL0 V16 409 D3200 664 3200 1736
U1405 2165 UP0
```

;;HALOEC1 UL0 V16 是设定绘图机状态的命令及参数, 其中 EC1 设定绘图机分辨率为 0.001 英寸, 后面的数字是以 0.001 英寸为单位的移动绘图笔的绝对坐标, 数字前的 D 或 U 表示是落笔移动还是抬笔移动, 即是画线还是空移笔。上面这一串数字的意义是: 画线, 线的起点是 (409, 3200), 再落笔画线到 (3200, 664), 再画到 (3200, 1736), 再抬笔移动到点 (1405, 2165)。最后的 UP0 表示笔归位, 绘图结束。PLT 文件的格式是相对固定的, 只要绘图机配置一定, PLT 文件的头与尾就固定了, 只有中间的坐标数据因图形的变化而随之变化。那么我们就可以从中得到 CAD 图形的数据了, 方法是, 用假读将 19 个字符的文件头略去, 接着就读后面的坐标数据, 且剔除数字前的“D”或“U”。见子程序 readplt。其运行结果就是读出一对坐标值 X, Y 和确定是落笔坐标还是抬笔坐标。

2. 生成图形点阵数据

上述读取工作得到的是一对对坐标值, 也就是一

条条线的端点坐标, 而我们要用打印机打印图形, 这些坐标值是无法直接打印图形的, 必需将这些线的坐标数据化成打印机打印的点阵数据。

转换成点阵数据的方法是先将线分解成许多点, 再根据这些点的坐标把内存中某一单元某一位写“1”, 在内存中形成由二进制位“1”构成的图形, 而这每个单元的值正是打印机打印图形所需的数据。生成图形点阵数据的具体步骤是:

(1) 把从 PLT 中读出的坐标数据化成以点为单位的坐标值。

$$X := \text{Round}(X * 0.18); Y := \text{Round}(Y * 0.18);$$

从 PLT 文件中读出的数是以英寸为单位的数, 乘上 0.18 就变成了以打印机的点为单位的数了。以下的计算均是以点为单位。

(2) 将线分解成点, 就是根据线的端点坐标计算出一组代表该线的点的坐标。设 (X1, Y1) 和 (X2, Y2) 为一线的两端点坐标, 那么样点数为 $|X2 - X1| + |Y2 - Y1|$ 中较大的一个, 假设为 N, 则 X 的分解步长为 $DX := (X2 - X1) / N$; Y 的步长为 $DY := (Y2 - Y1) / N$, 各采样点的坐标为:

$$X := X + DX;$$

$$Y := Y + DY;$$

如此计算 N-1 次便将此线分解完毕。

(3) 写内存单元。开辟一个缓冲区, 其结构是一个二维数组, 其类型为字节, 总容量为 420K 字节, 用 PASCAL 语言可描述为:

```
TYPE POI = ^ DIM
```

```
DIM = ARRAY[0..1999] OF BYTE;
```

```
VAR P: ARRAY[0..209] OF POI;
```

点阵是 210 * 8X2000 大小, 相当于纸上面积 237 * 290 平方毫米, 若已知一个点的坐标是 (X, Y), 则应将字节 $P[Y \text{ DIV } 8]^X$ 的第 $(Y \text{ mod } 8)$ 位写“1”, 即:

$$P[Y \text{ DIV } 8]^X := P[Y \text{ DIV } 8]^X \text{ OR } ER[Y \text{ MOD } 8];$$

ER 是个表, 其内容与打印机针的编码有关, 对于 LQ-1600K 打印机这个表的内容为:

```
ER[0] := 128;
```

```
ER[1] := 64;
```

```
.....
```

```
ER[7] := 1;
```

ER[0] 是打印机上边第一针的编码, ER[1] 是上边第二针的编码, 以此类推。

3. 打印

点阵数据生成后, 就可顺次地读出送给打印机打

印了。见 PLOT 子程序。

4. 裁剪与拼接

由于内存容量的限制,一次只能处理 237 * 290mm 大小的图形,对于较大的图形要分段处理,方法是用一个 CAD 的命令文件由大图生成多个 PLT 文件,一个一个地进行点阵数据的生成和打印,命令文件名字为 P8.SCR,其内容见文后,注意:3 后面是二个空格,p1-p7 后面都有一个空格,p8 与 0 之间有二个空格,输入时一个空格也不能漏掉。本命令文件的作用是生成 P1.PLT、P2.PLT...P8.PLT 共 8 个文件。若其长度为 17 字节则是无用的没有图形数据文件,就是绘图窗口落到图形的外边了,处理时不管它,处理顺序是 P1.PLT、P2.PLT,一直遇到文件长度为 17 则退出。由于选择每个裁剪窗口长度正好是 70 个打印行,所以打完一页不动打印机再打印下一页图就很好地接上了。这些均由程序自动处理,无需人的干涉。

二、操作步骤

假若程序已正确地输入(名为 LQ1600.PAS)并编译,命令文件 P8.SCR 也已用行编辑程序输入,则可按下述步骤打印 CAD 图形:

(1)进入 CAD 绘图,图幅的高不要超过 28 个绘图单位,长度不限,并按一个绘图单位等于一个厘米的比例绘,绘好后存盘,这里为了叙述方便假定图形文件名为 TU.DWG,然后退出 CAD,假设您使用的是汉化 CAD,要重新启动机器,清除汉字系统。

(2)在 CAD 子目录下键入

ACAD TU P8 回车

运行完后再次键入

LQ1600 文件名 回车

假设文件名指定为 ABC,则运行完后准备好打印机,再键入

COPY ABC?.PIC PRN 回车

打印机便开始打印了,若想打印多份,就重复执行 COPY ABC?.PIC PRN。

在执行 LQ1600 时若没指定文件名,则程序提示: Out File or Prn;这时还给您一次指定文件名的机会,若仍不指定文件名,只打回车,则程序处理出的点阵数据不存盘,而是边处理边打印,只打印一份。

最后几点说明:

(1)本程序在 IBM/PC XT AT 及其兼容机上通过运行。

(2)若机器配有数学协处理器,将程序中 {\$E+} 去掉,可用 Turbo Pascal 4.0 编译,用 5.0 编译机器有无协处理器均可。

(3)CAD 应将绘图机配置成 DMP52MP。

附 1: P8.SCR 命令文件清单

```
3 w 0.00,0 23.70,28 y y * 1 * 0 * 16 x y m 0,0
max n 0.16 n n 10=1 pl
3 w 23.70,0 47.40,28 n p2
3 w 47.40,0 71.10,28 n p3
```

```
3 w 71.10,0 94.80,28 n p4
3 w 94.80,0 118.50,28 n p5
3 w 118.50,0 142.20,28 n p6
3 w 142.20,0 165.90,28 n p7
3 w 165.90,0 189.60,28 n p8 0
```

附 2: LQ1600.PAS 源程序清单

```
PROGRAM PRPLOT(INPUT,OUTPUT);
{$E+}{$N+}
uses crt,dos;
TYPE poi= ^ dim;
dim=array[0..2005]of of byte;
var
p:array[0..209] of poi;
er:array[0..7] of byte;
i,j,n,u,pl,pn,x,y,x1,y1,x2,
y2,s,key;integer;
a,ch;char;
F;file of char;
lst;text;
dose;byte;
Fm,us,ms,gos,name,ff:string;
dirinfo,searchrec;
ind;word;
label 1,2;
procedure readplt(var x,y;integer);
var
ss:string;
s;char;
n1,n2;integer;
label 10,20,30,40,50;
begin
ss:='';s:=chr(0);n1:=8;n2:=0;
10: read(F,s);
if s=' ' then goto 20;
ss:=ss+s;goto 10;
20: if ss='UP0' then begin gos:='END';
goto 50;end;
if pos('U',ss)=1 then begin
ms:='U';delete(ss,1,1);
end;
if pos('D',ss)=1 then begin
ms:='D';delete(ss,1,1);
end;
Val(ss,y,n1);y:=round(y*0.18);
s:=chr(0);ss:='';
30: read(F,s);
if s=' ' then goto 40;
ss:=ss+s;
goto 30;
40: Val(ss,x,n1);x:=round(x*0.18);
```

```

50;
end;
procedure pokemem(var x1,y1,x2,y2;integer);
var
a,b,n,i,jj,xx,yy,w;integer;
dx,dy,x0,y0;single;
begin
a:=x2-x1; b:=y2-y1;
if abs(a)>abs(b) then n:=abs(a)
else n:=abs(b);
if n<>0 then begin dx:=a/n;dy:=b/n;
end;
x0:=x1;y0:=y1;
for i:=0 to n do begin
xx:=round(x0);yy:=round(y0);
jj:=yy div 8;
w:=yy mod 8;
p[jj]^ [xx]:=p[jj]^ [xx] or er[w];
x0:=x0+dx; y0:=y0+dy;
end;
end;
procedure plot;
var
i,j;integer;
a,b,c;byte;
begin
GotoXY(14,12);Textcolor(7);
if name<>'prn' then write('Write disk ')
else write('PRPLOT...');
write(lst,chr(27),'3',chr(24));
j:=0;
while j<210 do begin
write(lst,chr(27),'*',chr(39),chr(208),chr(7));
for i:=0 to 1999 do begin
a:=p[j]^ [i];if a=26 then a:=24;
b:=p[j+1]^ [i];if b=26 then b:=24;
c:=p[j+2]^ [i];if c=26 then c:=24;
write(lst,chr(a),chr(b),chr(c));
end;
writeln(lst);
if keypressed then begin ch:=readkey;
if ord(ch)=27 then exit;end;
j:=j+3;
end;
end;
{-----Main Procedure-----}
begin
name:=ParamStr(1);
if length(name)=0 then begin
write('Out prn or file:');readln(name);end;

```

```

if length(name)=0 then name:='prn';
Textmode(3);ch:='a';
Window(20,1,80,25);
findfirst('P?.PLT',Archive,dirinfo);
dose:=doserror;
pn:=0;p1:=0;
if dirinfo.size>17 then
begin p1:=1;pn:=p1;end;
while(dirinfo.size>17 and(dose=0))do
begin
findnext(dirinfo);
dose:=doserror;
if(dirinfo.size>17)and(dose=0)then
Pn:=Pn+1;end;
er[7]:=1;
for i:=6 downto 0 do er[i]:=er[i+1]*2;
for j:=0 to 209 do new(p[j]);
clrscr;
GotoXY(5,15);
Textcolor(11);
writeln('LQ1600 PLOT (Press ESC End)91.7');
GotoXY(10,10);Textcolor(2);
writeln('*****');
GotoXY(10,11);Textcolor(2);
writeln('* * * * *');
GotoXY(10,12);Textcolor(2);
writeln('* * * * *');
GotoXY(10,13);Textcolor(2);
writeln('* * * * *');
GotoXY(10,14);Textcolor(2);
writeln('*****');
Textcolor(5);
for u:=p1 to pn do begin
gos:='';
GotoXY(10,9);Textcolor(12);
Writeln('Prplot ... Page ',U);
a:=chr(0);writeln(a);
for j:=0 to 209 do begin
fillchar(p[j]^ ,2006,a);
end;
str(u,us);
Fm:='P'+us+'.PLT';
if name<>'prn' then FF:=name+us+'.pic'
else ff:=name;
Assign(F,Fm);reset(F);
assign(lst,ff);rewrite(lst);
s:=0;
for i:=1 to 19 do read(f,a);
readplt(x1,y1);

```

(下转13页)

编制菜单程序的辅助程序@.COM

湖北省公路局科研所(430030) 张红

在较大的应用系统中,往往需要在 DOS 一级借助菜单程序将各个松散的分系统模块组织起来,以期形成一个紧凑、高效、对用户透明的整体系统。因此,向用户提供一个由彩色画面装饰的菜单屏幕就具有了一定的实用价值。

在文章[1]中,作者介绍了一种菜单组织技术。这种技术对需要经常增删子模块,从而必须不断修改菜单中的功能项和屏面编排格式的系统来讲,因要经常修改汇编源程序,并要进行再次的汇编、连接和转换,所以不太方便灵活;另外,屏幕的颜色是单一的,未充分利用丰富多彩的颜色。为此作者想到应充分利用许多微机上的彩色显示功能,并且应提供一种更加灵活方便的菜单组织技术。

我们知道在 dBASE Ⅲ 中分别提供了设置屏幕属性和显示字符串的功能,这给作者某些启示,可以从某种程度上模仿 dBASE Ⅲ 的功能。作者从而想到了可将光标定位、屏幕属性设置和字符串显示三种功能组合到一起,为菜单组织提供一种更灵活的技术,以便为操作人员提供一种更友好的用户界面。

由此作者利用汇编语言编制了能实现这些功能的程序@.ASM(附后)。在此程序中采用的思想可参见作者的文章[2][3]。读者通过汇编、连接和转换即可获得@.COM。程序的使用格式可见所附源程序前部的注释部分。批处理程序 XTGLO.BAT 为利用@.COM 编制一个彩色菜单提供了一个示范。若将@.COM 与文章[1]中采用的批处理程序一起使用即可以编制出用户界面好的多彩、灵活菜单。

@.COM 的使用条件是:必须在 DOS 版本至少为 2.00,且带彩色显示卡的微机上运行,另外还要求操作系统的配置文件 CONFIG.SYS 中应该有安装 ANSI.SYS 的语句:DEVICE=ANSI.SYS。

参考文献

- [1] 张红,一种在操作系统下组织菜单的实用技术,电脑学习,1990年第5期
 - [2] 张红,为 DOS 扩充设置屏幕属性外部命令,微计算机应用,1991年第4期
 - [3] 张红,直接在操作系统下设置彩色屏幕的方法,计算机时代,1990年第1期
- ; 能在操作系统一级模拟 dBASE Ⅲ 的 @ 功能的程序
; @.ASM
; 用法:@行号,列号 SAY“字符串” IN 前景色代码/
; 背景色代码 颜色代码表为:0:黑色,1:红色,2:绿

色,

;3:黄色,4:兰色,5:洋红,6:青色,7:白色

CR EQU 0DH

SPACE EQU 20H

SLASH EQU 2FH

DOLLAR EQU 24H

DISP MACRO MESSAGE

MOV DX,OFFSET MESSAGE

MOV AH,09H

INT 21H

ENDM

CODE SEGMENT PARA PUBLIC 'CODE'

ASSUME CS,CODE,DS,CODE,SS,CODE,ES,CODE

ORG 100H

START PROC NEAR

JMP MASTER

FLAG DB 0

CURSORLOC DB 1BH, '['

DB 8 DUP(?)

COLOR0 DB 1BH, "[0m", "\$"

COLOR DB 1BH, "[3;4 m", "\$"

MESS DB 40 DUP(?)

MASTER: CLD

MOV SI, 80H

XOR AX,AX

LODSB

CMP AX,0

JE EXIT

MOV CX,AX

LABEL1: LODSB

CMP AL,SPACE

JNE LABEL2

LOOP LABEL1

LABEL2: JCXZ EXIT

LEA DI,CURSORLOC+2

MOV CL,0

LABEL3: CMP AL, '0'

JB EXIT

CMP AL, '9'

JG EXIT

STOSB

INC CL

LODSB

CMP FLAG,1

JE LABEL4

CMP AL, ','

JNE LABEL3

CMP CL,2

JC EXIT

MOV AL, ','

STOSB

MOV CL,0

MOV FLAG,1

LODSB

JMP LABEL3

```

LABEL4:  CMP AL,SPACE
        JNE LABEL3
        MOV AL,'H'
        STOSB
        MOV AL,DOLLAR
        STOSB
        JMP NEXT
EXIT:    MOV AX,4C00H
        INT 21H
NEXT:    ADD SI,3
        LODSB
        CMP AL,SPACE
        JNE EXIT
        LODSB
        CMP AL,'"
        JNE EXIT
        LEA DI,MESS
LABEL5:  LODSB
        CMP AL,'"
        JE LABEL6:
        STOSB
        JMP LABEL5
LABEL6:  MOV AL,DOLLAR
        STOSB
        LODSB
        CMP AL,SPACE
        JNE EXIT
        INC SI
        INC SI
        LODSB
        CMP AL,SPACE
        JNE EXIT
        LODSB
        CMP AL,'0'
        JB EXIT
        CMP AL,'7'
        JG EXIT
        LEA DI,COLOR+3
        STOSB
        LODSB
        CMP AL,SLASH
        JNE EXIT
        LODSB
        CMP AL,'0'
        JB EXIT
        CMP AL,'7'
        JG EXIT
        LEA DI,COLOR+6
        STOSB
        DISP CURSORLOC
        DISP COLOR
        DISP MESS
        DISP COLOR0
        JMP EXIT
START   ENDP

```

```

CODE    ENDS
        END START
C>EDLIN XTGL0.BAT
End of input file
* L

```

```

1: * ECHO OFF
2: CLS
3: @3,27 SAY "总 系 统 目 录" IN 2/3
4: @6,28 SAY "1...计划管理系统" IN 3/2
5: @08,28 SAY "2...财务管理系统" IN 2/5
6: @10,28 SAY "3...人事管理系统" IN 6/3
7: @12,28 SAY "4...工资管理系统" IN 4/2
8: @14,28 SAY "5...结束系统操作" IN 6/5
9: ECHO ON

```

格式文件与 SET FILTER TO 命令的结合

云南昆明凉亭轧钢厂(650215)麦红波

在 dBASE III 和 FoxBASE 中,都提供了格式文件(扩展名为.FMT)和 SET FILTER TO 命令,笔者把格式文件与 SET FILTER TO 命令相结合,编制出一种新形式的数据输入程序。

示例中 SET FILTER TO 命令,用来对文件中的记录按命令中给出的逻辑表达式进行过滤,只读出满足条件的记录,与用 LOCATE 命令加 CONTINUE 命令所实现的功能相同,结构比其简洁。SET FILTER TO 命令的特点是,对满足逻辑表达式条件的记录在文件中的存放方式没有要求。

格式文件是用来生成一个屏幕格式,它只能用 APPEND, INSERT, CHANGE, EDIT 和 READ 命令来激活,当格式文件被激活后,所生成的屏幕格式具备了全屏屏幕编辑功能,尤其是可用 <PgUp> 键读出一个记录,用 <PgUp> 键读出一个记录, <Esc> 键结束输入操作,这比用命令语句来实现上述功能要方便,简化了程序。注意示例中有一条 SET CLEAR OFF 命令,它的作用是在敲键读记录时,屏幕不闪动,即显示新记录时,系统不清屏重显屏幕格式。(注意 dBASE III 中没有 SET CLEAR OFF 命令)

示例

1. 水电费数据库结构(SDF.BDF)

field	FieldName	Type	Width	Dec
1	编号	Numeric	4	
2	部门	Character	4	
3	姓名	Character	8	
4	水电费	Numeric	6	2

2. 格式文件(SDF.FMT)

@6,30,say '水 电 费 输 入 表'

(下转 13 页)

Azusa/2708 病毒与介质描述

华南师范大学计算机系(510631) 何悦荣

一、引言

近来,在某些 IBM/PC 系列微机上流行着一种操作系统病毒,此病毒在发作时,将使串行(Serial)和并行(Parallel)口不能正常使用,干扰软盘驱动器的操作。具体症状为:在打印文件时,会无故出现无纸或打印机没有准备好等假象;用 DIR 命令显示软盘目录时,经常是换了盘,但显示的是上一张盘的目录。

因为此病毒没有特征串,所以在社会上有着不同的名称(SP 病毒、IO 病毒、打印病毒、2708 病毒),根据美国 McAfee 协会 1991 年 2 月 15 日公布的世界各地发现的病毒命名为 Azusa/2708(阿珠沙)。

我们用中国公安部计算机监察处 90 年 8 月下发的 SCAN(检测 78 种病毒),不能查出此病毒。改用公安部 91 年 2 月下发的 SCAN 3.0 才能检测出来。

二、Azusa 病毒的特征

Azusa 病毒是 BOOT 区类病毒,它借鉴了大麻病毒的设计思想,传染 A、B、C 盘,病毒长度为 0170H。因病毒修改了 INT 13H,所以具有很强的传染力,只要有关 INT 13H 的操作,磁盘就可能被传染。用带毒系统启动机器时,病毒把系统内存减少 1KB,并进驻 RAM 高端,保存 INT 13H 入口地址,并修改指向高端 XXXX:000BH。对于硬盘,病毒仅替换了原来的自举记录,不需额外的扇区,但它保留了硬盘分区表信息,即主引导扇区后 42H 个字节的信息。硬盘感染后没有什么症状,当带毒硬盘启动系统 32 次后,把第一并行口和第一串行口的地址低八位改为 00,从而在通信和打印时主机误报故障。对于软盘,病毒先将传染盘的 0 道 0 面 1 扇区的引导记录转移到 27H 道 1 面 8 扇区,若成功,则在复制厂商名和版本号后,将病毒体代替原来的引导程序。因此该病毒将破坏软盘 27H 道 1 面 8 扇区的数据。

三、Azusa 病毒干扰软盘操作的原因

在被 Azusa 病毒感染的计算机上,用 DIR 命令显示软盘目录时,经常是换了软盘,但显示的依然是上一张盘的目录内容;有时在显示目录时会出现:

Not ready error reading drive A

Abort, Retry, Fail?

打入 R 后能继续等现象。通过分析 DOS 及 Azusa 病毒所设置的 INT 13H,发现了问题的原因。

有些 DOS 版本(如 DOS 3.0 和 DOS 3.1)的设备驱动程序支持可移动介质。DOS 3.1 版本设有介质描述符,对于 5.25 英寸 15 扇区的介质来说,描述符是

F9H, 5.25 英寸 9 扇区的介质描述符是 FDH。对于 DOS 3.1 版本,如果更换磁盘,介质检验设备驱动程序能使 DWORD 指针指向卷标识符。如果没有更换磁盘,输入/输出设备驱动程序也返回一个指向卷标识符的 DWORD 指针。当命令代码字段是 1 时, DOS 为驱动单元调用介质检查程序并给出当前介质描述符字节。介质检查程序能给出下列之一的信息:①未改变介质;②改变介质;③不确定;④指错代码。如果驱动程序将设备标题属性字的可移动介质位 11 置 1,且驱动程序返回 -1,则驱动程序必须将 DWORD 指针指向前一个卷标识字段(ID)符。

DOS 3.1 设置的 INT 13H,每次磁盘操作后, DOS 调用介质检查程序并给出当前介质描述字节。以后 DOS 在检索目录时,将根据介质描述符字节决定是否使用已读入 DOS 缓冲区的目录数据。

Azusa 病毒设置的 INT 13H,判断是否对软盘操作,若不是则转执行原 INT 13H,若是则读软盘引导扇区到高端 XXXX:200~3FFH,判断是否被感染(高端 289H 的字是否为 B190H),若已感染则转原 INT 13H,若未感染则传染软盘后转原 INT 13H。因此换了磁盘后执行 DIR 时,病毒先读盘检测磁盘是否已被感染等操作,读后驱动器中介质更换信息被清除,到转执行原 INT 13H 时,得不到介质更换的返回码,则 DOS 认为磁盘没有换过盘,继续显示缓冲区的目录,而没有真正去读磁盘目录。

出现目录检索异常的现象,是和 DOS 版本有关。因为不同的 DOS 有不同的设备驱动程序。如果 DOS 没有直接从 ROMBIOS 的 INT 13H 获得介质更换返回码,就不会发生异常情况。例如,启动受感染的 DOS 4.01 后,就不会出现上述异常现象。

由于 Azusa 病毒未复制磁盘 BIOS 参数块,故染毒软盘在 DOS 3.2, DOS 4.01 下给出 "Sector not found error"(扇区没找到)的信息,而不能正常读写,但在 DOS 2.0、2.1、3.0、3.1 和 3.3 下不影响使用。

四、Azusa 病毒的解毒与免疫方法

如果感染了 Azusa 病毒,可用公安部下发的 KILL V35.01 消除。如果没有上述消毒软件的用户,对于软盘可以用 DEBUG 把干净的 DOS 盘(一般要同一版本)的引导区覆写病毒盘的引导扇区。若无同版本的病毒引导扇区,可用 DEBUG 把染毒软盘 27H 道 1 面 8 扇区(即数据区逻辑 2CEH 扇区)的数据读到内存(注意看,是否还是引导区数据)然后再写到 0 面 1 扇区。

对于硬盘,由于病毒放弃了原来的主引导记录,必须恢复其首扇区原来的内容,可采用回写已备份的引导扇区的方法消毒。若没有备份,可借助同版本的 FDISK.COM 来重做与原来相同的硬盘分区的方法消除病毒。

消毒后的磁盘仍有被染毒的危险,因此应当为磁盘增加免疫功能。Azusa 病毒的免疫码在磁盘 089H 处,病毒传染前先检测此处的字是否为 B190H,若是则不再传染。所以把 B190H,放在首扇区 089H 处,就可免疫。但对于系统盘,由于首扇区 089H 处含有有用的程序,不能简单地将免疫字直接放在此处。免疫时需先搬动几个字节,以便给免疫码空出位置。

注:本文提及的软盘为 5.25 英寸 360K 磁盘。

MS FORTRAN 程序的 菜单设计技术

重庆医药设计院 曹国钧
渝州大学图书馆 王 健

当今,在 CCBIOS 汉字操作系统下,IBM PC/XT 微机上开发的 MS FORTRAN 程序采用菜单较多,增加了人机交互的能力。但是,MS FORTRAN 系统中并未提供这个方面的能力,因此,要实现这些菜单形式就比较困难。笔者在开发一些应用软件过程中,充分使用了 CCBIOS 2.13H 中文操作系统的特殊显示功能和扩展键盘系统文件 ANSI.SYS,较为完善地解决了这一问题。下面就一些方面谈谈笔者浮浅的看法。

一、屏幕的光标定位及清屏

在 CCBIOS 2.13H 系统中提供了一个键盘扩展文件 ANSI.SYS,它有清屏及光标定位的功能。笔者已将其写成 MS FORTRAN 语言形式。

1. 清屏语句:

```
WRITE(*,'(1X,A1,3H[2J])' ) CHAR(27)
```

其中 CHAR(27)就是控制码 ESC 的 ASCII 代码。

2. 光标定位语句:

```
WRITE(*,'(1X,A1,1H[,I2,1H[,I2,1HH)])' )
```

```
CHAR(27),10,20
```

为了方便,笔者已将清屏及光标定位功能编成了子程序 CLSLOCA(L,L1,L2),其中清屏控制变量 L=0 时表示清屏,L1,L2 是光标定位的行和列位置。

```
SUBROUTINE CLSLOCA(L,L1,L2)
```

```
IF (L.EQ.0) WRITE(*,'(1X,A1,3H[2J/])' )  
CHAR(27)
```

```
IF (L1.LT.10.AND.L2.LT.10) THEN
```

```
WRITE(*,'(1X,A1,1H[,I1,1H[,I1,1HH/])' )
```

```
CHAR(27),L1,L2
```

```
ELSE IF (L1.GE.10.AND.L2.GE.10) THEN
```

```
WRITE(*,'(1X,A1,1H[,I2,1H[,I2,1HH/])' )
```

```
CHAR(27),L1,L2
```

```
ELSE
```

```
IF (L1.LT.10) WRITE(*,'(1X,A1,1H[,I1,1H[,I2,  
1HH/])' ) CHAR(27),L1,L2
```

```
IF (L2.LT.10) WRITE(*,'(1X,A1,1H[,I2,1H[,I1,  
1HH/])' ) CHAR(27),L1,L2
```

```
ENDIF
```

```
RETURN
```

```
END
```

二、汉字大小显示程度:

在 CCBIOS 2.13H 系统中,提供了一种特殊显示功能,常常为开发软件人员所忽视。其中有一条功能就是在汉字系统中控制汉字大小显示(即能在屏幕上显示出 24×24 点阵汉字来)。

格式:chr[\$](14)+["@字体 a,b,c,...,A,B,C,...汉字或字符串"]

例如:如果要把“重庆医药设计院工程经济室”在屏幕上显示出 L 型字体,则可以写成下列 MS FORTRAN 语句:

```
WRITE(*,'(1X,A1,5H+["@L,A,2H"])' )
```

```
CHAR(14), '重庆医药设计院工程经济室'
```

注意:上述命令功能仅在 CGA、EGA、VGA 等彩显适配器上正确地显示。如果是 Hercules 卡(即单显 21 行),在运行上述 MS FORTRAN 程序之前,应先在 DOS 提示符下键入下列命令:

```
C>MODE BW80
```

使之转向黑白图形方式,然后在启动 CCBIOS 2.13H 系统的批处理文件的屏幕文件改为 CC25.com(原为 ch21.com),这样,也仍能够正确地显示 24×24 点阵汉字。

最后要说明的是:

1. 对于彩色图形显示器,还考虑汉字的颜色及屏幕颜色;

2. 在运行 MS FORTRAN 菜单程序之前,应在系统配置文件 CONFIG.SYS 中有一句:device=ansi.sys,确保上述命令串正确地运行。

如何制做病毒标本

华南师大计算机 90 级(510631) 梁宇翀

目前全世界约发现近三百种计算病毒。计算机病毒成为社会的新“公害”。

为了研究病毒的防治,我们经常要采集、制做一些病毒标本。本文将在这方面提出自己的一些看法。

病毒绝大部分属于操作系统病毒和外壳型病毒。例如小球、石头、智囊、磁盘杀手、广大、六四、打印等属于操作系统病毒;黑色星期五、维也纳、瀑布、杨基、毛毛虫属于外壳型病毒。这些都是国内很常见的计算机

病毒。

制做操作系统病毒标本最简单的一个方法是：用带毒盘启动，再对一张正常 DOS 盘进行感染即可。这种方法的好处主要是方便。但浪费空间，一张磁盘只能保存一种操作系统病毒；而且标本在遇到别的操作系统病毒攻击时容易丢失。

另一种制作操作系统病毒标本的方法是：用正常 DOS 启动，将整个病毒在 DEBUG 下调入内存，用 N、R 和 W 命令作为一个数据文件存盘，并做好这种病毒的说明。这种方法解决了上法的难题，但比较麻烦。笔者一般使用这种方法。

制做外壳型病毒标本最简单的方法是：把染上病毒的 EXE 或 COM 文件直接复制过来，进行保存。这种方法很简单，但由于有些 EXE 或 COM 文件比较长而会造成浪费磁盘空间，且分析起来较困难。

另一种制做外壳型病毒标本的方法是：先用 MASM 或 ASM 制做一个只有两条指令——“MOV AH, 4CH 和 INT 21H(返回 DOS)”的 EXE 或 COM 文件，再在病毒控制下，把这个文件感染即可。这种方法解决了上法的难题。

但有点要注意，做外壳型病毒标本时要在改变标本的扩展名使其非 EXE 或 COM 后保存，以防被别的外壳型病毒再次感染。

用 dBASE Ⅲ MEMO 字段开发试题库系统

长沙中建五局中专 陈卫平

各种试题库系统都有一个共同需要解决的问题，即试题正文和答案中的特殊字符和图形的处理问题。这是因为目前通用的各种汉字操作系统中，点阵字符字库很难面面俱到，即使使用造字程序，由于受点阵的影响，也难以达到满意的效果。笔者经过探索，发现在 dBASE Ⅲ MEMO 字段下，编辑 BASIC 程序，利用系统提供的打印控制字符（如上、下标、下划线、字体比例、字型、分数型式等）可以打印出较满意的试题型式。其技巧就是在 dBASE Ⅲ MEMO 字段下，键入 BASIC 程序，将所需打印的特殊字符加控制字。整个程序可以是一个试题，也可以是一组试题。程序编辑好后，退出 dBASE Ⅲ。然后在 CONFIG. DB 文件中加入 WP=BASIC。重新进入 dBASE Ⅲ，一旦进入 MEMO 字段，按 Ctrl + Pgdn 键，则自动运行 BASIC 程序，从而打印出试题。

在实际系统中，可以把一个试题视为一个记录，试题的正文用 MEMO 字段，其余参数或指标用其它类型字段管理，这样每一个记录就是一个完整的试题，可方便地在 dBASE Ⅲ 下进行各种操作。

至于图形处理，我们也可采取同样手段，在 BASIC

下先用各种键盘绘图程序作出图形，记住每个试题对应的图形文件名，然后按前面所述的方法编写一段 BASIC 程序（含调图和程序下硬拷贝等功能），这样打印机就既能打印出带特殊字符的正文，又能打印出与试题相对应的图形，生成的试卷就较满意了。

本方法在 LX-286 机和 LQ-1600 打印机下运行成功。

关于《dBASE Ⅲ 过程文件 加密的发现》质疑

浙江嘉兴民丰造纸厂(314000) 张维仁

《电子与电脑》91 年第 9 期刘人伟同志《dBASE Ⅲ 过程文件加密的发现》一文存在着多处问题。①文中提到的 CMIS V1.0(88.10)计量管理软件名称应叫作“工业企业通用计量管理信息系统”，(简称 CMMS)是由国家技术监督局组织开发和发行的，而 CCDOS V4.0 才是由机电部六所开发的，文中将两者弄混了；②CCDOS V4.0 与 2.13、UCDOS 及 XSDOS 等中文操作系统一样，并不能提高 dBASE Ⅲ 程序的运行速度；③文中提及的 DB05. PRG 等其实并非是过程文件，而是一般的命令文件。CMMS 的作者为防止内存配置较小的计算机装载程序时内存不够，在编制程序时除 V2.0 的 XZ36. PRG 外，均未采用过程文件这一形式；④CMMS V1.0 是一个完全开放的系统，无一程序进行过加密处理。进入时需将 #1 盘插入 A 驱动器是所采用的 dBASE Ⅲ 所致，并非程序编制时进行的加密处理。程序并没有加密过，用户完全可以根据需要自行修改，但从 dBASE Ⅲ 换到 FOXPLUS 环境下执行确是存在文中所提及的问题，到底是为什么呢？原因是当初作者编写 CMMS V1.0 时是使用 WS 的格式文件（或者叫文本文件，进入 WS 时选 D）编写的。WS 的格式文件每满一页均自动插入一个 ASCII 码 8AH(138)作为分页符，这就是刘文用 DEBUG 所见的 0D8AH。dBASE Ⅲ 环境下命令文件中这一分页符在执行时是予以忽略的，并不会引起出错，用 WS 或在圆点提示符下用 modi comm<命令文件名>查看时不能显示出 8AH 这一 ASCII 码，但可看到文件的全部内容，说明文件并未加密。在 FOXPLUS 环境下则将这 1 字符视为出错，这就是 dBASE Ⅲ 与 FOXPLUS 的一个不同之处。明白了 8AH 产生的机制，要除去它就相当简单了，不必费脑筋去编制程序，只要把这些命令文件用 WS 的编制非格式（文本）文件（选 N）调入后马上重新存一次盘，即可将这些每隔 55 行出现一次的 8AH 去掉了。④文中提及 CMMS V1.0 有些命令文件执行速度慢并可能出错，是由于 V1.0 编制仓促不够完善所致。因此换用 FOXPLUS 固然可

（下转 13 页）

在单台驱动器上如何作软盘备份

太原太行仪表厂研究所(630006) 杨栋林

市面上流行的各类微机中,一般都配接一台磁盘驱动器,或 3.5 英寸,1.44MB,或 5.25 英寸,1.2MB。即使配接双软盘驱动器,也是两种类型的,要么是 5.25 英寸 1.3MB+5.25 英寸,360KB,要么是 5.25 英寸,1.2MB+3.5 英寸,1.44MB。这给软盘的备份带来很大不便,例如,在一台 386 微机(配接一台 5.25 英寸,1.2MB 驱动器加一台 3.5 英寸 1.44MB 驱动器),备份 1.2M 高密磁盘上的软件,通常的作法是:先将 A: 盘的软件全部拷入硬盘 C: 中,然后格式化一张 1.2M 磁盘,再将 C: 盘中的内容 COPY 到 A: 上。这样作要占用 C: 盘的空间,而且要作两次拷贝,耗时、费力。

其实,DOS 的设计者为我们考虑了这个问题。在 MS-DOS3.2 和 MS-DOS3.3 中,有一个设备驱动程序 DRIVER.SYS 就是专门为我们解决这个问题的,下面介绍如何借助 DRIVER.SYS 来作磁盘备份。

DRIVER.SYS 使用户能为某一物理驱动器另外再指定一个驱动器标识字母,使用时用户必须在 CONFIG.SYS 文件中加以相应的说明。在上面的例子中,我们的微机已经有了两个软驱一个硬驱,已经分别指定

了字母 A,B 和 C,则可以通过下列语句给物理的第一台驱动器再赋给另一个驱动器字母 D。

```
DEVICE=DRIVER.SYS/D:0/T:80/S:15/H:2/C/F:1
```

在这个语句中,/D:0 就是给第一台物理驱动器,再赋给一个驱动器字母 D,第一台软驱称为驱动器 0,第二台软驱为 1,依此类推。选项/T,/S 和/H 分别说明物理驱动器的道数,每道扇区和磁头数。选项/C 只适用于 PC/AT 及其兼容机,在这些机器的驱动器上包含有可以检测驱动器门什么时候被打开和盘片什么时候已被更换的信号;选项/F 告诉 DRIVER.SYS 物理驱动器的类型,D 是单面(180K)或双面(360K),1 是 1.2M 字节的盘;2 是 3.5 英寸,720K 字节的盘。

有了上述语句,就可以用下列命令把一个或若干个文件从一个 1.2M 的盘上拷贝到另一个 1.2M 的盘片上,DOS 会在需要的时候提示用户更换盘片。

```
COPY A:filename1 D:filename2
```

或者用全盘拷贝命令作整盘的复制

```
DISKCOPY A: D:
```

同样,DOS 会在需要时提示用户更换盘片。

一种不打印数值零的方法

蚌埠石油站(233040) 曹和平

按照国内处理报表的习惯,当数值为 0 时,不打印,以求报表的清晰。在 FoxBASE+ 中有一个 IIF 函数,该函数的语法是:

```
IIF(<逻辑表达式>,<表达式 1>,<表达式 2>)
```

利用此函数可代替 IF...ENDIF 语句,方便实现。

例如:

:

```
P6=STR(独子女费,6,2)
```

```
P6=IIF(Val(P6)=0,"",P6)
```

:

```
@&H,B Say".....|独子女费|....."
```

```
@&H+1,B Say".....|&P6.|....."
```

在打印时,当数值为 0 时,即以空格取代。

交互式的 DEL 命令

国防科技大学计算机系 90 级(410073) 莫琳

DOS 的使用者有一个很大的烦恼便是不小心将磁盘中的有用的文件删除了,这种情况在使用全局文件名进行删除时更容易发生。倘若及时发现,还可以修复,否则便无法挽回了,想必用过 DOS 的人都有过这种担心和感受吧?

为此,我用 PASECAL 编了一个名为 KILL 的文件,可对你所要删除的每个文件提出询问,确认后再删去。应用这个程序,使得在一个全局文件中保留个别文件而删除其它的文件变得十分方便、安全。

程序中应用了一个技巧:用 exec 命令将所要删除的文件,用形如 dir *. * >f.dat 的 DOS 命令存放在 f.dat 文件中。然后从此文件中读取文件名而不是从磁盘目录区中读取,这样避免了程序的冗长,又充分利用了系统资源。程序运行要求 CommoD.com 文件在当前目录。

```
program kill_file;  
{ $m 32768,0,0 }  
uses dos,crt;
```



```

var
  fn:string[20];
  fi:string[12];
  i:byte;
  e:file;
  f:text;
  c:char;
  a:string[8];
  b:string[4];
begin
  repeat
    clrscr;
    writeln(' ',30,' * Erase file * ');
    write(' ',20,' Enter the erase — file-
name: ');
    readln(fn);
    if fn='' then fn:=' * . * ';
    exec(' \command.com', '/C' +' dir ' +fn
+' >f.dat ');
    window(25,4,50,20,);
    assign(f,' f.dat ');
    reset(f);
    for i:=1 to 4 do readln(f,a);
    read(f,a);

```

(上接 6 页)

```

1: readplt(x2,y2);
if ms='U' then begin
  X1:=x2;Y1:=y2;goto 1;end;
if gos<>'END' then begin
  POKEMEM(x1,y1,x2,y2);x1:=x2;y1:=y2;
if (s mod 64)=0 then begin
  gotoXY(14,12);
  write(' Vector:',s:6);
end;
s:=s+1;goto 1;

```

(上接 8 页)

```

@7,30 say '=====';
@9,35 say '编号:'
@9,41 SAY 编号
@11,41 SAY 部门
@13,35 SAY '姓名:'
@13,41 SAY 姓名
@15,33 SAY '水电费:' GET 水电费
READ

```

(上接 11 页)

可使程序运行速度得以加快,但并不能解决程序本身存在的问题,目前,这套计量管理软件已推出第二版,除采用了 FOXPLUS 加快速度外,还修正了 V1.0 存在

```

readln(f,b);
repeat
  fi:=a+'.'+copy(b,2,3);
  repeat
    write(fi,13,' (Y/N?) ');
    readln(c);
    c:=upcase(c);
  until c in ['N','Y'];
  if c='Y' then begin
    assign(e,fi);
    erase(e);
  end;
  read(f,a);
  readln(f,b);
until eof(f);
close(f);
window(1,1,80,25);
gotoxy(22,20);
write(' Erase another(Y/N)? ');
readln(c);
until (c='N') or (c='n');
assign(f,' f.dat ');
erase(f);
end.

```

end else PLOT;

close(1st);

if ord(ch)=27 then goto 2

end;

2: if name<>'prn' then begin

gotoxy(4,15);

writeln(' Printer plot command:copy',
name,' ?.pic prn');

end;

end.

3. 主程序(SDF. PRG)

CLEAR

USE SDF

SET FILTER TO 部门='车队'

GO TOP

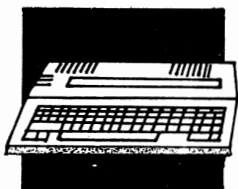
SET FORMAT TO SDF. FMT

EDIT

CLOSE DATA

SET FORMAT TO

的错误,对模块结构进行了调整,同时整个程序都进行优化。因此,将版本升级才能解决 V1.0 所存在的全部问题。



学习机之友

为 APPLE DOS 增加多目录功能

上海番禺中学(200052) 秦燕军

APPLE DOS 对磁盘文件的管理是通过在 \$11 道 \$0 区上的 VTOC 表进行的。在 VTOC 表中记录了磁盘的磁道和磁区使用状况(第 \$38—\$C3 字节),另外在第 \$01~\$02 字节中指出了第一个文件目录区的道号和磁区号。系统一共有十五个文件目录区,存放在 \$11 道 \$1~\$F 区上(即俗称的目录道),并以链式结构衔接。因此 DOS 在管理文件时,只需取得 VTOC 表中的第 \$01~\$02 字节内容就可以对磁盘上的全部文件执行存取功能,也很容易通过对 VTOC 表中第 \$01~\$02 字节的修改,使文件目录区存放在磁盘的其他位置,或可以保留标准的文件目录区,再开辟新的文件目录区,达到多目录的功能。

目前已有许多关于多目录问题的讨论,基本都是建立在上述方法上。但所用的方法都存在着一一些问题。首先是都修改了磁盘上原来的记录格式而增加了新的非自由磁区。对于 APPLE II 用户来说,存放用户程序的磁区原本已经很少,现在若再被新目录占用掉一部分,似乎有些得不偿失;其次各目录之间的相互转换比较麻烦,而制作多目录磁盘的手续就更复杂;再者还要兼顾 VTOC 表中磁区使用情况的协调统一问题等等。

针对上述问题,笔者对 APPLE DOS 进行了细致的剖析,终于找到了自认为比较满意的方法,即:利用 CATALOG 命令建立多目录及其间的转换。

一、使 CATALOG 命令允许携带 L 参数

修改 DOS 的一个原则是要求修改后的 DOS 与标准 DOS 完全兼容,否则修改的意义就不是很大。依照这一原则,笔者只是修改了 DOS 命令有效键语表,给 CATALOG 命令添加了一个新的参数:Ln。

使用 Ln 参数时,n 的取值范围是十进制数 1~11,它表示 11 个相对独立的新目录区(超出该范围系统会发出“范围过大”的错误信息)。CATALOG 命令中原有的 V、S、D 三个参数依然被保留,例如:

```
CATALOG D2,L4
```

表示显示 2 号驱动器中磁盘上第 4 目录里的文件名。当 n=1,或省略 L 参数时,CATALOG 命令显示主目录(即标准目录道)上的文件名。

在用 CATALOG Ln 命令进入新目录以后,所有的 DOS 命令都将在该目录下运行,不会影响其余目录。

二、在 CATALOG 命令处理程序中修改 VTOC 表缓冲区

DOS 对文件的管理是通过磁盘上的 VTOC 表实现的,即文件目录区道号和区号是“固化”在磁盘上的。在使用多目录功能时要反复对磁盘上的 VTOC 表进行修

改将是一件困难的工作,因此笔者利用 DOS 命令在管理文件时每次都先要把 VTOC 表读入内存然后再加以分析使用的过程,在 DOS 命令把 VTOC 表读入缓冲区后,立即着手对其进行修改,那么接下来 DOS 命令使用的就是已经按要求修改过的 VTOC 表,这样也就同样达到了对磁盘上 VTOC 表进行修改的目的。这种方法的优点,一是 VTOC 表的修改变得简单,二是保留了标准的 VTOC 表,使得磁盘的使用与原来一样。

对 VTOC 表修改的工作由 CATALOG Ln 命令完成,这也就是“用 CATALOG 命令建立多目录”的由来。

三、VTOC 表的恢复与磁盘使用状况的更新

在 DOS 命令中,有一类是向磁盘写入文件的命令(如 SAVE、BSAVE 等),这些命令的执行将会改变磁盘原来的磁区使用状况。因此该类命令在完成对文件的写入后,须再把缓冲区中已包含新的磁区使用状况的 VTOC 表重新写入磁盘的 \$11 道, \$0 区,以更新盘上的 VTOC 表。

可是由于在前文中已对缓冲区中的 VTOC 表关于文件目录区的字节内容作了修改,现若直接将其一起写入磁盘,势必会改变磁盘上的 VTOC 表。因此,当 DOS 命令需要把缓冲区中的 VTOC 表写入磁盘时还要先恢复缓冲区中 VTOC 表的文件目录道、区号,以满足更新后仍是标准的 VTOC 表的要求。

由于本法始终是使用了一个 VTOC 表,自然也就避免了多 VTOC 表磁盘磁区使用状况难以协调统一的问题。

四、使用 DOS 在磁盘的空区作为新的文件目录区

DOS 存放在磁盘上的 \$0、\$1、\$2 道,但是 \$2 道仅被使用了 \$0~\$4 区,而 \$5~\$F 区未被使用,因此可以将这部分磁区作为新目录的存储区,以达到不占用磁盘容量的目的。

五、多目录功能 DOS 的实现

由于笔者只是对 DOS 原有的处理程序,进行了一些修补,就实现了多目录功能,因此该 DOS 的制作十分简单,具体步骤如下:

```
]PR #6 ✓ 启动 DOS
]CALL-151 ✓ 进入监控系统
* A92A;78 ✓ 修 DOS 命令有效键语表
* AD98;20 B3 B6 ✓
* B6B3;AD 6C AA A2 11 C9 01 F0
13 18 69 04 C9 10 10 11
A2 02 8D D7 B 6 8E D8 B6
20 DC AB 60 A9 0F 4C C5
B6 4C C9 A0 0F 11 ✓
```

```
* B00E;4C D9 B6 ✓
* B6D9;C9 01 F0 0F 48 A9 11 8D
  BC B3 A9 0F 8D BD B3 68
  4C 52 B0 20 52 B0 AD D7
  B6 8D BD B3 AD D8 B6 8D
  BC B3 60 ✓
```

* ^ C ✓ 用 CTRL-C 退出监控系统

]INIT HELLO ✓

至此,就产生了一个多目录功能 DOS 的磁盘

六、多目录功能 DOS 的应用

多目录功能 DOS 的用途很多,现略举一、二:

1. 文件加密

将需要保密的文件存放在主目录(标准目录)以外的其他目录中,一般用户将因找不到文件目录而无法查看文件的内容。由于本法所存放的文件目录区是动态的,可用任意磁区作为文件目录区,例如,当你想把文件目录存放在 \$A 道 \$B 区,只需键入:

```
]POKE 44441,203;POKE 46807,11;POKE 46808,
10
```

而在磁盘上的 VTOC 表中却没有留下丝毫痕迹。因此在结合其他加密法,一定会大大增加破译的难度。

2. 隐藏 DOS 的引导文件

在磁盘上有一个 HELLO 文件,在启动 DOS 后将会去自动执行它。有很多加密法是在 HELLO 文件中加进了加密语句而起到保密作用。只要破译了 HELLO 文件,也就破译了整个软件。能否使 HELLO 文件在完成了自己的任务后自动消失,令解密者无从下手?一种方法是在 HELLO 文件中的最后加上 DELETE 命令。但这样做却有缺陷,一是如果磁盘写保护,就不起作用;二是该磁盘只能一次性使用。

现在这个问题可以比较圆满地解决了:先用 CATALOG Ln 命令进入非主目录,然后在该目录下 IINIT HELLO,再启动刚被格式化好的磁盘,你就会发现该磁盘也象标准 DOS 盘一样执行了 HELLO 文件,但如果你用 CATALOG 命令查看磁盘,却没有 HELLO 文件,它已经“消失”了!

同样地,你也可以将 HELLO 文件的目录区存放在任意磁区,或者在 HELLO 文件中加上 POKE 43306,112 语句,以破坏 CATALOG Ln 命令。

3. 提供相关文件的储存

有些文件比较重要,因而希望留个备份名,一般是在其他磁盘上或用其他文件名再存一遍,但这种方法容易引起混淆,不是因为备份不知放在哪里,就是因为遗忘了备份名。现在不用担心了,我们可以在同一张盘片上用相同的文件名存放一个内容相同的文件(有点类似 PC 机上子目录功能),只要在标准目录中存放文件后,再进入其它目录重新存放一遍就可以了。

在两个或多个文件名相同的文件,其中只有一个为真,其余为假时,这就是多文件名保密法。

验证四色猜想的 BASIC 程序

四川南充一中(637000) 陈庆祥

四色猜想最初是英国大学生弗朗西斯·格里思在 1852 年提出来的,其内容是:在给平面或球面上的任何地图着色时,最多用四种不同颜色就可以把所有相邻的区域分开。这个问题刚提出来时,许多数学家都不屑一顾,觉得它过于简单。但很快就发现这是一个极难证明的问题。在长达一百多年的时间里,一批又一批的杰出数学家和成千上万个数学爱好者都曾为四色问题的证明而绞尽脑汁,但是都未如愿以偿,因此只能称它为“四色猜想”。四色猜想是世界公认的近代数学三大难题之一。直到 1978 年,才由两位美国数学家阿佩尔和哈肯通力合作,利用当时最先进的三台 IBM360 型高速电子计算机,运行了 1200 小时,检验了所有可能的图形组合方式,终于使四色问题获得了完善的证明。有趣的是:对这次证明的审核,同样也是由电子计算机完成的。利用电子计算机使四色猜想成为四色定理,不仅是当时轰动世界的新闻,也开创了利用电子计算机证明数学难题的先例。

下面的 BASIC 程序可以验证四色定理,即在理论上可以实现用不多于四种颜色解决任何复杂地图的着色问题(所处理地图复杂程度仅受机器内存容量的限制),而且可以给出所有的四色着色方案,它会使你领略到电子计算机的神奇功能。

程序说明:

用二维数组 A 存放各相邻区域的邻接关系,比如 $A(1,2)=1$ 表示区域 1 与区域 2 相邻, $A(2,5)=0$ 表示区域 2 与区域 5 不相邻,称之为邻接矩阵。用一维数组 B 存放各区域的颜色号数,如 $B(5)=2$ 表示第 5 区域涂第 2 种颜色。

采用回溯算法来处理各区域的涂色问题。即用 1~4 号颜色逐一试探,凡是使已涂区域各相邻者颜色不同就被认可,否则用下一种颜色再试,直至使所有区域都被涂上合适的颜色。改变从 200 语句开始的 DATA 语句中的图形区域个数 N 的值和图形的邻接矩阵数据,就可以处理不同图形的涂色问题。

针对图 1 和图 2,由两组 DATA 语句给出不同数据,分别运行后可迅速给出两组不同的涂色方案。

程序清单及运行实例:

```
5 REM 利用电子计算机处理地图着色问题。
10 READ N;DIM A(N,N),B(N);PRINT
20 FOR I=1 TO N:PRINT LEFT$( "("+STR$(I)+
  ")",4);
30 FOR J=1 TO N:READ A(I,J);NEXT J,I:PRINT
40 FOR I=1 TO 4:B(I)=I
50 FOR J=2 TO N:C=0
60 C=C+1;IF C>4 THEN 120
```

```

70 FOR K=1 TO N
80 IF A(J,K)=1 AND C=B(K) THEN 60
90 NEXT K;B(J)=C;NEXT J
100 FOR K=1 TO N:PRINT LEFT$( " " +STR$(B
(K))+" ",4);;B(K)=0
110 NEXT K:PRINT:T=T+1
120 NEXT I
130 PRINT:PRINT "TOTAL=";T:END
200 DATA 8,
201 DATA 0,1,1,0,0,0,0,1
202 DATA 1,0,1,0,0,0,0,1
203 DATA 1,1,0,1,1,0,0,1
204 DATA 0,0,1,0,0,0,0,1
205 DATA 0,0,1,0,0,1,1,1
206 DATA 0,0,0,0,1,0,1,1
207 DATA 0,0,0,0,1,1,0,1
208 DATA 1,1,1,1,1,1,1,0
]RUN

```

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
1	2	3	1	1	2	3	4
2	1	3	1	1	2	3	4
3	1	2	1	1	2	3	4

TOTAL=3

修改从 200 语句开始的数据便可处理第二个图的着色问题:

```

]LIST200-
200 DATA 12
201 DATA 0,1,1,1,1,1,0,0,0,0,0,0
202 DATA 1,0,1,0,0,1,1,1,0,0,0,0
203 DATA 1,1,0,1,0,0,0,1,1,0,0,0
204 DATA 1,0,1,0,1,0,0,0,1,1,0,0
205 DATA 1,0,0,1,0,1,0,0,0,1,1,0
206 DATA 1,1,0,0,1,0,1,0,0,0,1,0
207 DATA 0,1,0,0,0,1,0,1,0,0,1,1
208 DATA 0,1,1,0,0,0,1,0,1,0,0,1
209 DATA 0,0,1,1,0,0,0,1,0,1,0,1
210 DATA 0,0,0,1,1,0,0,0,1,0,1,1
211 DATA 0,0,0,0,1,1,1,0,0,1,0,1
212 DATA 0,0,0,0,0,0,1,1,1,1,1,0
]RUN

```

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
1	2	3	2	3	4	1	4	1	4	2	3
2	1	3	1	3	4	2	4	2	4	1	3

TOTAL=2

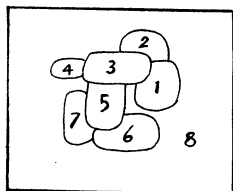


图 1

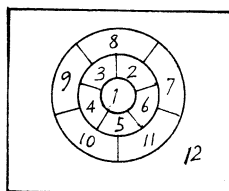


图 2

磁盘不抹掉文件的 35 轨 改 40 轨方法

浙江杭州(310015) 钱仕宏

在 APPLE II, CEC-I 上, 标准 DOS3.3 FORMAT 过的磁盘都是 35 轨的, 怎样能把 35 轨的磁盘不抹掉上面的文件, 而改成 40 轨或 41 轨呢?

下面介绍一下改成 40 轨的方法, 需准备以下工具:

1) COPY II PLUS 5.0 或 5.0 以上版本一张。(若没有, 可用任何扇区编辑工具加 40 轨 FORMAT 工具代替)

2) COPY 工具盘一张(最好 40 轨)

3) 空盘一张

现在就可动手了。1) 先用 COPY II PLUS(以下皆称 PS)上的 FORMAT 格式化一张空盘, 暂称复制盘。

2) 用 COPY 把复制盘第 \$23—\$27 轨(改成 41 轨则为 \$28), 拷到原盘上。(若没有 40 轨 COPY 工具, 可把原盘 \$00—\$22 轨拷到复制盘上, 此复制盘以后就当原盘操作。

3) 用 CPS 上的 SECTOR EDITOR 从原盘读入 \$11 轨 \$00 扇区, 将地址 \$34 处的 23 改为 28(41 轨则为 29), 再从地址 \$C4 处开始依次向后填 FF FF 00 00 FF FF 00 00 FF……填到 \$D7(41 轨则为 \$DB)为止。最后将这一扇区写回原盘。

对于想改成 41 轨的读者, 则 FORMAT 和 COPY 工具都需 41 轨的。

解拆 C—WORDSTAR 五笔字型

广东惠东职业中学
高一(5)班(516300) 陈晓乐

在中华机上使用 CEC—WORDSTAR 时, 觉得该系统中的五笔输入法非常实用, 因此我认真分析了 CEC—WORDSTAR 的结构, 想出一个简单易行的解拆方法, 把五笔输入法拆成两个单文件, 供以后使用。现将解拆方法介绍给大家。

〈一〉启动 CEC—WORDSTAR 1.0, 进入主菜单后按“X”退出。

〈二〉在西文状态下进入监控, 键入:

```

BCB2:0C 0F✓
BCB6:00 40✓
BC66:4C 59 FF
BC56G ✓

```

〈三〉光标出现后键入:


```
1536 IF Q$(5)="工程师"OR Q(3)>20 AND Q(17)>
=250 THEN FG=1
```

程序 3 是应用实例。说明如下:

3710:调用程序 2,设置筛选条件。

3730:打开工作文件。

3750:读出一个记录中的各字段。这里的数组变量名必须和程序 2 一致。

3775:调用程序 1,若条件满足,返回时 FG=1,否则 FG=0,据此可知当前记录是否符合给定条件,从而实现了库记录的筛选。

程序 3

```
3710 GOSUB 1550
3730 PRINT D$;"OPEN";W$
3740 FOR R=1 TO H
3750 PRINT D$;"READ";W$
3760 FOR J=1 TO XM:INPUT Q$(J):Q(J)=VAL(Q
$(J)):NEXT
3765 PRINT D$
3770 IF BT$=" " THEN 3780
3775 GOSUB 1530:IF FG=0 THEN 4000
3780
3785 REM 此处对符合条件者进行处理
3790
4000 NEXT
```

用本文方法模拟 dBASE 的宏代换效果很好,不必对程序作大的修改就能在已投入使用的软件中应用,由于条件可以任意指定及组合,弥补了常规方式的不足,扩展了 BASIC 语言程序的功能。但要注意的是:本方法直接修改程序的内存映像,必须细心谨慎地按要求去做,否则非但达不到预期目的,甚至还会破坏内存中的整个用户程序。

程序中所用变量说明:

H:数据文件中的记录个数。

XM:每个记录中所含字段数目。

N\$(I):第 I 字段的字段名。

LX\$(I):第 I 字段的数据类型。

"N"表示数值型,"C"表示字符串型。

改进的 UNNEW 程序

唐山白玉瓷厂 张亭

为追回误操作 NEW 掉的 BASIC 程序,笔者曾编制过一则 31 个字节长的 UNNEW 程序,发表在《中华学习机》杂志 1990 年第八期。考虑到很多 CEC-I 型机未配驱动器,而要做这种追回工作的又往往是一些初学者,因此不断地压缩改进还是有必要的。

改进后的 UNNEW 程序只有 25 个字节长。当误 NEW 掉 BASIC 程序后,只要未键入任何新程序,可进行如下操作(其中"␣"表示空格键):

```
]CALL-151
```

```
* 2E7:A001F36788A23B169
E669D02E66A90D0F2CA
D0F14CF2D4
```

```
* 2E7G
```

```
]
```

至此,BASIC 程序已完全恢复。

本文程序在原理上与原 UNNEW 完全一样。除更加简短之外,另外两个特点也都具备:适用性广,可直接用于 APPLE 机的软汉字、西文等多种系统;浮动性,能运行于其他地址而无须任何更改。

比起原 UNNEW 程序来,本文程序在循环部分更精炼,同时使用了一个 6502 怪指令码 F3,与后面的 67 实现"INC(\$67),Y"的功能。这在监控的反汇编"L"命令中是看不到的。6502 的用户不能用这个怪指令,只可根据道理稍加修改,也只须增加 1 个字节的长度。

CEC-I 中华学习机 磁带游戏软件的解密

中国农科院蜜蜂研究所(100093)胡发新

CEC-I 中华学习机的磁带游戏机软件,为了运行速度快,多数是采用机器语言编制的,极少数是用 BASIC 语言编写的,这些磁带游戏软件多数又是经过加密处理,进入游戏后不能中断,只有采取某种解密方法,才可读出程序内容。

机器语言程序的解密对于初学的爱好者,可能是一大障碍。笔者经过摸索剖析,对大多数磁带游戏软件可进行解密。现予以介绍供感兴趣者参考。

在键入 PLAY 后,读入磁带信号到刚要显示还未显示,并发出“嘟”一声时,立即按录音机暂停键,再按下学习机 CTRL-RESET 键,中断程序运行,屏幕显示 9DC1 和监控字符“*” (如不能显示 9DC1 和监控字符“*”,说明本游戏软件不适用此法解密),键入 800L,一直到找到赋值语句 STA \$FC,STA \$FD 的读入数据存放首地址指针和 STA \$FE,STA \$FF 的读入数据存放尾地址指针,并记下首尾地址数值。然后再键入 240L,找到无条件转移语句 JMP \$0300,往下找到第二个 JMP \$××××游戏入口地址,记下地址数值,并改为监控入口地址 JMP \$FF65 或 JMP \$FF69。JMP 是无条件执行程序转移指令,游戏软件正常运行到此指令的游戏入口地址时,就转到游戏入口地址并自动进入游戏。经过修改后,把磁带倒回少许(观察录音机记数计,稍有转动即可),键入 800G,重新按录音机 PLAY 键,并立即按学习机回车键,显示屏出现游戏名称等,然后出现读带显示符号(如无读带显示符号可重新倒回磁带少许,再进行上述操作,直到读带符号出现为止),当程序运行到 JMP \$FF65 或 JMP \$FF69

指令时,自动进入监控,并出现监控符号“*”,表示程序解密成功。键入游戏入口地址,用 G↙就可进行游戏或用 L↙读出程序。如要录制游戏程序,不能按查到的首、尾地址录制,要用游戏入口地址为首地址,游戏入口地址加上原来首地址到尾地址的长度为尾地址。

用以上方法对高速公路、空袭轰炸、大食客、打伞兵、两栖飞机、中国象棋、警察抓小偷等大多数磁带游戏软件均可进行解密。阅读游戏程序或重新录制游戏软件;从中也可了解到加密的奥妙,并学习到不少机器语言的知识。

也谈 CEC—I 全功能造字

广西巴马一中(547500) 覃 敏

怎样才能使得造出的字能够象固化汉字一样方便地使用呢?笔者在实际中找到了一种行之有效的方法,供大家参考;CEC—I 内部的汉字及符号分为 87 个区,每个区有 94 个位,理论上应该有 8178 个汉字及符号,但其中第 10 区至第 15 区留给用户作造字之用。遗憾的是 CEC—I 的设计者们并没有给用户提供造字的方法,当你输入这 6 个区的区位码时,屏幕上没有任何显示。是不是得到一个空串呢?不是的,用 LEN 函数可以测出它的长度和其他区的汉字的长度没有什么两样,在西文状态下也能看到它的 ASCII 码,说明用区位方式是能够输入第 10 区至第 15 区的汉字内码的,只是汉字处理程序不加以显示及打印罢了。经过对汉字处理程序的仔细分析后,发现在 \$ECCD~\$ED82 处是取汉字点阵的子程序,它首先判断要找的是汉字还是字符,是汉字时还要判断是否在第 10 区至第 15 区之间,若是则设定进位为 0 后返回,告诉显示及打印程序没有找到点阵,显示及打印程序就会将该字略过,这就是有内码而没有显示及打印的原因。如果能够使程序在判断出第 10 区至第 15 区之间时跳到我们的取造字点阵子程序,那么不就实现造字的显示及打印了吗?笔者想到了一种方法:先将汉字处理程序由辅存 2 复制到 RAM 卡的相同地址,再将 CSW 和 KSW 指向复制的汉字处理程序,汉字系统照样能正常工作,这样我们就得到一个 RAM 汉字系统,可以对它进行各种修改了。

根据上面的分析,笔者用机器语言编了程序 1。由于构思巧妙,程序仅占半页多一点的内存,其中 \$300~\$31B 将固化汉字处理程序复制到 RAM 卡的 \$EC00~\$FFFF, \$31C~\$331 修改取点阵子程序及建立 & 指针, \$332~\$345 进入中文状态并修改 CSW 和 KSW 指针, \$346 及 \$355 分别是新的 CSW 及 KSW 的进入点, \$35B~\$361 是置 RAM 卡读写有效的子程序, \$362~\$385 是取造字点阵的子程序。

\$6000 以后是用户造字点阵区,每个字占用 \$20 个单元,点阵结构与固化字库相同,具体可参看《中华学习机 CEC—I 技术参考手册(硬件)》一书。字是按区

位码由小到大排列的,即第一个字的区位码是 1001,第二个是 1002,依此类推。由于一个程序中不会用到很多造字,所以程序 1 只安排第 10 区作为造字区,并可同时使用 94 个造字。若读者不想用 \$6000 以后内存作为点阵区,可修改 \$36F 单元的值,如改为 \$D0 则可将点阵区移入 RAM 卡,提高内存的使用效率。

下面介绍一下使用方法:输入造字点阵并运行程序 1,机器便会工作于 RAM 卡中的汉字系统,先按 F3 进入区位方式,再键入 1001,便得到第一个造字(口字旁右边一个野字,广东话是“东西”的意思),再键入 1002 又得到第 2 个造字(一个丰满的空心感叹号),这些造字将与其他汉字一样可显示在屏幕上,也可由打印机用各种字型输出。要注意的是,执行 PR#3 或者在西文状态下按中文键后,机器进入的是固化汉字系统,这时造字是不能显示及打印的,要进入 RAM 汉字系统只要执行一下 & 即可。

顺便提一点,有了这个 RAM 汉字系统后,读者还可以对它进行各种修改,以满足不同的需要。比如,原汉字系统中有两个错误:一是封锁了“~”键,二是 PRINT CHR\$(29)功能失效。只要修改两个单元中的值便可改正过来;方法是:在监控状态下键入 C081↵C081↵ECE2: 7F↵F0A2: 8D↵

另外,没有磁盘机的读者须将程序 1 中 \$343 单元的值改为 \$60 方可正确运行。

程序 1

```
0300— A0 EC 84 27 A0 00 84 26
0308— 20 AB C3 B1 26 20 B9 C3
0310— 20 5B 03 91 26 C8 D0 F0
0318— E6 27 D0 EC A9 D5 8D F3
0320— EC A2 02 BD 86 03 9D C9
0328— EC BD 89 03 9D F5 03 CA
0330— 10 F1 20 00 C3 A9 03 85
0338— 37 85 39 A9 46 85 36 A9
0340— 55 85 38 4C EA 03 29 7F
0348— C9 11 F0 06 20 5B 03 4C
0350— 34 C3 4C 3A C3 20 5B 03
0358— 4C 06 C3 2C 83 C0 2C 83
0360— C0 60 A9 00 85 EE 85 EF
0368— A5 D7 E9 20 85 ED A9 60
0370— 8D 5E ED A9 B2 8D 71 ED
0378— 20 53 ED A9 40 8D 5E ED
0380— A9 A4 8D 71 ED 60 4C 62
0388— 03 4C 32 03
```

用户造字字库

```
6000— 00 02 E8 7F BE 42 AA 2A
6008— EA 13 AA 12 AA 7E EA 53
6010— 8A 30 8A 10 EA 13 8E 10
6018— 82 10 80 13 70 14 20 08
6020— 80 01 40 02 20 04 20 04
6028— 20 04 20 04 40 02 40 02
6030— 40 02 80 01 80 01 00 00
6038— 80 01 40 02 40 02 80 01
```

ProDOS 磁盘操作系统入门(续)

廖 凯

(五)、EXEC 命令

EXEC 命令将控制权从键盘转移到文本文件,而使计算机执行文件内的命令,就如同你从键盘打入一样。EXEC 不能被 CTRL-C 或 RESET 中断,要想中断只能重新启动系统。在执行 EXEC 命令时,不清除内存中的程序,而将 EXEC 文件内带行号的程序段加到内存中的程序上。EXEC 命令一次只能用在程序上。如果正在执行的命令中又有一个 EXEC 命令,则关闭第一个 EXEC 文件,执行第二个 EXEC 命令。EXEC 命令不被 NEW 或 CLOSE 命令所影响。监控命令在 EXEC 文件内不能执行。

EXEC 命令既可以执行文件中的命令,又可以合并两个程序。在合并程序时,相同的行号将被新的 EXEC 行号所代替。

在建立 EXEC 文件时,无行号的命令其后要有一个回车字符。EXEC 命令可以使用[,F#][,S#][,D#] 参数,它允你跳过数行或数个数据域。用法如下:

EXEC 路径名[,F#][,S#][,D#]

下面的程序可以生成任何类型的 EXEC 文件。一行文字最多可包含 239 个字符,当行满时要按回车键。在输入结束时要打入 END 和回车。

```
100 REM MAKE AN EXEC OR TEXT FILE
110 REM 238 CHARACTERS MAY BE INPUT AT ONE
    TIME
115 DIM L$(100)
120 D$ = CHR$(4)
125 PRINT D$;"PR#3"
130 A = A + 1
140 PRINT "Use RETURN when line is complete ... Type
    END when finished;"
150 PRINT "ENTER LINE "A
160 ONERR GOTO 320
170 INPUT " "; L$(A)
180 IF L$(A) = "END" OR L$(A) = "End" OR L$(A) =
    "end" THEN L$(A) = "END"; GOTO 200
190 GOTO 130
200 ONERR GOTO 370
210 INPUT "What is the name and path of the EXEC file ";
N$
220 PRINT D$;"OPEN"N$ : PRINT D$;"CLOSE"
230 PRINT D$;"DELETE"N$
240 PRINT D$;"OPEN"N$
```

```
250 PRINT D$;"WRITE"N$
260 FOR I=1 TO A
270 IF L$(I) = "END" THEN 300
280 PRINT L$(I)
290 NEXT
300 PRINT D$;"CLOSE"
310 END
320 POKE 216,0
330 PRINT "An ERROR has occurred at line "A
340 PRINT "Would you like to retype line "A" back in?"
350 GET A$ : IF LEFT$(A$,1) = "Y" THEN 170
360 GOTO 220
370 POKE 216,0 : HOME : PRINT "PATH NOT FOUND"
380 GOTO 200
```

行号 125 设置屏幕显示为 80 列方式;行号 130 用于计数,应避免使用 FOR 循环;行号 210 要求用户输入 EXEC 文件的路径和文件名;行号 260 将数组内各单元写到磁盘上。当遇到数据 END 时,关闭文件,结束程序。行号 320 至 380 为错误处理程序。

1. 实际应用

下面将举例说明 EXEC 命令在实际中的应用。若要将一些子程序转换到一个文本文件内,就要用 EXEC 命令来执行,转换后这些子程序随时可以被存取。下面的程序是一个转换程序,在运行此程序后,它将转换它自身到一个文本文件内。

```
1 D$ = CHR$(4)
2 F$ = "MAKE.TEXT"
3 PRINT D$;"OPEN"F$ : PRINT D$;"CLOSE"
  : PRINT D$;"DELETE"F$ : PRINT D$;"
  OPEN"F$ : PRINT D$;"WRITE"F$
4 LIST
5 PRINT D$;"CLOSE"
建立一个文本文件:
```

①装入你的子程序或程序到内存,确信行号 1 至 5 未被使用;如果已用,它们将被覆盖。当运行此程序时,首 5 行将加到文本文件内。

②EXEC MAKE.TEXT 文件到你的程序内。

③列印行号 2 并改变 MAKE.TEXT 为你所需的文件名。

④打入 RUN,程序将被转换到行号 2 所指定的文

本文件内。

将文本文件加到程序内：

①装入你的 BASIC 程序。

②打入 EXEC 文件名,F6。这 F6 参数会跃过文本文件的首五行。

(六)CHAIN 命令

这是 ProDOS 的一个新命令。当一个 BASIC 程序太长而难以装入内存时,可用 CHAIN 命令将程序分为两部分,分别执行。在运行任一部分时,其变量仍保存在存储器内。你没有必要在使用 CHAIN 之前关闭打开的文件或存储变量。系统的完整性不会被破坏。

在使用 CHAIN 命令时,所有变量和打开的文件将为链接程序而保留。你不能在链接程序内再定义一个用在初始程序内的数组,而要在初始程序内定义所有的变量及数组。若在随后的链接程序内定义数组,则会发生 OUT OF MEMORY 错误。这问题可通过在第五章第二节的数组清除器(Array Eraser 程序)来解决。

CHAIN 命令的使用很灵活,它可以从指定的行号处开始执行。链接命令用,@#参数来指定行号,用法如下:

```
10 PRINT D$;"CHAIN/卷名/文件名,@100"
```

在上面例子中,链接程序的执行将在行号 100 处开始。注意,在返回初始化程序时,不要再执行变量初始部分,否则会使数据和重要的计数器重置。

(七)、RAM 磁盘

ProDOS 的一个最好的特征是它具有将扩展 80 列卡内额外的 64KRAM 作为一个 RAM 磁盘的能力。RAM 磁盘的最大优点是它的工作速度非常快。

扩展 80 列卡具有 64K 字节的容量,可通过用 S3, D2(槽口及驱动器号)来指定它的卷名:/RAM。其目录表最多可存放 12 个文件或子目录。你可以更改其目录的名字,也可以通过卷名或槽口及驱动器号来列印/RAM 目录。在/RAM 磁盘上有 120 块的存储空间(近 60K 字节)可用于存放你的程序,剩余 4K 字节由 ProDOS 占用。

由于/RAM 会很快地传送资料,所以存储你的链接程序段到/RAM 内是一个好的方法。CHAIN 命令可以在不同的程序段之间交换使用。

(八)、系统启动

较新的 APPLE 计算机都具有自启动 ROM,要启动系统时,只要将系统盘插入 1 号驱动器内开机即可,而无论它是 DOS3.3,ProDOS 或 Pascal。当一个 ProDOS 磁盘被格式化时,操作系统不被存放在磁盘上,这可以使用户有 50 块的磁盘空间存放数据。ProDOS 操作系统是以文件方式存放在磁盘上的。

ProDOS 启动磁盘

ProDOS Filer Utility 程序可格式化 ProDOS 格式的磁盘并给磁盘指定一个名字(卷名)。作为启动的磁盘要将 Users. Disk 磁盘上的两个系统文件拷到你的新磁盘上:BASIC.SYSTEM 和 PRODOS。

在启动系统时,ProDOS Kernel(PRODOS 文件)被装入内存并运行,ProDOS 搜寻每个外围槽口并确定哪个槽口有磁盘控制卡,而后装入 BASIC.SYSTEM 解释程序并运行,最后寻找名为 STARTUP 的程序。若找到此程序则运行;若没找到,系统将显示:

```
PRODOS BASIC 1.1
```

```
COPYRIGHT APPLE,1983/84
```

STATRUP 程序可以是 BASIC,EXEC 或二进制程序。在 ProDOS User. Disk 磁盘上 STARTUP 程序是一个菜单式程序,它具有多项功能。

你可以设计一个实用程序作为自己的启动程序。在设计一个启动程序时,最好建立一个文本文件。你可用它来存储辅助程序段到 RAM 内,然后启动你的主程序。下面的例子将说明如何实现这功能。

```
LOAD/DATA.DISK/PROGRAM.B
```

```
SAVE/RAM/PROGRAM.B
```

```
LOAD/DATA.DISK/PROGRAM.C
```

```
SAVE/RAM/PROGRAM.C
```

```
RUN/DATA.DISK/PROGRAM.A
```

这程序作为 STARTUP 被保存,当磁盘启动时,PROGRAM.B 和 PROGRAM.C 被装入/RAM,并且 PROGRAM.A 被执行。PROGRAM.A 现在可以以惊人的速度调用其它的程序。另外,PROGRAM.B 会调用 PROGRAM.C,反之亦然。

(九)、绘图

ProDOS 没有改变 DOS3.3 的任何绘图指令。如果你要放置造型表在 HIMEM 之上,就必须小心设置 HIMEM 的位置。

ProDOS 分配 1K 字节的文件缓冲区供数据使用,并会在文件被打开或关闭时自动分配缓冲区空间。如果你的主机有 64K RAM,在启动系统时,HIMEM 将被定为 38400。用如下语句可查到 HIMEM 的当前值:

```
PRINT PEEK(115)+PEEK(116)*256
```

ProDOS 系统留给 BASIC 程序的空间在 2048 和 38400 之间。图形页 1 在 8192 和 16383 之间,图形页 2 在 16384 和 24575 之间。内存的 768 至 975 单元是一个空区,可用来存放短小的机器语言程序。

注意,HIMEM 必须在文件被打开或数据被装入内存之前设置。在 BLOAD 数据到高分辨率图形页 2 时,要先关闭 80 列卡。

(未完待续)

第四章 6502 MPU 的指令系统

南京大学大气科学系(210008) 朱国江

指令系统是程序设计的基础。任何一个程序设计总是以完成预定任务为目的,而就其功能来说,主要包括进行算术逻辑运算和各种控制,这些操作又都要用到各种数据和代码,因此,任何一种计算机的指令系统都应包括数据传送、程序控制及算术逻辑运算这三类指令。6502 微处理机共有 56 条指令,由于每条指令又可对应有不同的寻址方式,因而实际上一条指令形成了好几种不同的操作码,这样就相当于好几条指令,所以又可认为 6502 微处理机共有 151 条指令。

为了更好地学习机器语言程序设计,熟悉 6502 指令系统,了解 6502 中可供编程的寄存器情况,掌握各种寻址方式,就显得特别重要。本讲主要介绍 6502 指令系统,对其中每一条指令,不仅要熟悉其功能,而且要了解指令的执行对标志寄存器 P 的各位所产生的影响,这是学习本讲的要点。下面我们分三种指令类型对 6502 指令系统加以介绍。

一、数据传送类指令

6502 指令系统的数据传送类指令主要包括:数据由内存寄存器取到寄存器;由寄存器传送到内存单元;各寄存器之间的数据传送等三部份,共 20 条指令,分为取数、送数、传送、进栈出栈和移位 5 种类型。

应该特别指出的是,数的传送是计算机最基本、最经常、大量的操作,在整个程序中占有很大的比重。数的传送是否灵活,传送速度快还是慢,选用什么样的寻址方式,对整个程序结构是否精巧以及对程序的执行都起着重要的作用,因此,掌握数的传送和操作,是学习 6502 机器语言程序设计的基本功。

1. 取数指令

取数指令是指 MPU 某个内存单元取数到某个寄存器(A、X 或 Y)。它只有三条:LDA,LDX,LDY,指令执行后只对负数标志 N 和零标志位 Z 有影响。

①LDA——由存储器取数送入累加器, $M \rightarrow A$ 。

根据寻址方式的不同,LDA 分别有 8 种具体形式。这种指令的功能是将一个立即数或将某个存储单元中的内容送累加器 A。

例如,LDA # \$63,是采用立即寻址方式的指令,其机器码为 A9 63。执行本指令后除了将立即数 63 送累加器(原累加器值被破坏)外,还对处理器状态标志寄存器 P 中的 Z 和 N 两个标志位产生影响。本例中被传送的操作数 63 不是一个 8 位全为 0 的数,所以零标志位 Z 被置为 0;又由于 63 的最高位(第 7 位)是 0,因

为是正数,所以负数标志 N 被置为 0。

至于程序计数器 PC 内容由 nnnn(LDA # \$63 的首地址第 1 字节处)指向下一条指令所在地址 nnnn+2。

②LDX——由存储器取数至寄存器 X, $M \rightarrow X$ 。

LDX 指令是将一个立即数或某个存储单元的内容送寄存器 X,它有 5 种寻址方式。

例如,指令 LDX \$4000,Y,采用绝对 Y 寻址方式,设 Y 寄存器内容为 \$28,且(\$4028)=4F,则指令 LDX \$4000,Y 将 \$4000 加上 Y 寄存器内容得 \$4028,然后将该地址 \$4028 的内容 4F 送入 X 寄存器即(\$4028) \rightarrow X,最后 X 寄存器的内容为 \$4F,写成二进制码为 0100 1111,可见最高位为 0,负数标志位 N=0;又由于执行指令后结果非 0,所以零标志 Z=0。由于绝对 Y 寻址系 3 字节指令,所以程序计数器 PC 指向 nnnn+3。

③LDY——由存储器取数送寄存器 Y, $M \rightarrow Y$ 。其功能与 LDX 指令类似,亦包括 5 种寻址方式。

2. 送数指令

送数指令是指 MPU 将某个寄存器(A、X 或 Y)的内容放到存储器的某个存储单元中。它有三条:STA,STX,STY,各指令执行后对各标志位均不产生影响,即保持各指令执行前状态。

①STA——累加器 A 中的数存入内存, $A \rightarrow M$ 。

STA 指令执行后累加器内容不变,内存内容改变。例如,(A)=63,(003A)=40,则执行 STA \$3A 指令后,零页地址 003A 中的内容变为 63。STA 指令有 7 种寻址方式,或者说内存单元的地址可分别由七种寻址方式表达。

②STX——变址寄存器 X 中的数存入内存, $X \rightarrow M$ 。它有三种寻址方式。但运算中绝对不能有 X 寄存器。零页 Y 变址指令 STX \$nn,Y 是正确的,而 STX \$nn,X 却是非法的。

③STY——变址寄存器 Y 中的数存入内存 $Y \rightarrow M$ 。它也有三种寻址方式。同样,运算中绝对不允许用 Y 寄存器,如 STY \$nn,Y 是非法的。

3. 传送指令

传送指令是指在 MPU 内部的各个寄存器之间传送数据。这类指令均采用隐含寻址方式,指令的操作数隐含在操作码之中,因而均为一字节指令。传送指令共有 6 条,它们是:TAX,TXA,TAY,TYA,TSX,TXS。前

五条指令执行后均会改变标志寄存器 P 中的 Z 位和 N 位标志,但不影响 P 中的其余标志位;TXS 指令则不影响 P 中任何标志位。

①TAX——将累加器 A 的内容传送至变址寄存器 X,即 $A \rightarrow X$ 。TAX 指令为暂存累加器 A 中的数提供了方便。这是因为程序设计中累加器 A 使用十分频繁,在进行另一操作之前,常常要暂时保存累加器中的内容,如果用存储单元或堆栈来暂存累加器中的数,则不如用 TAX 指令更为灵活。

②TXA——将寄存器 X 的内容传送至累加器 A,即 $X \rightarrow A$ 。TXA 和 TAX 指令配对使用,可暂时保存累加器 A 的内容,也可用于改变变址寄存器 X 的内容。

③TAY——将累加器 A 的内容传送至寄存器 Y,即 $A \rightarrow Y$ 。TAY 与 TAX 指令的区别在于将变址寄存器 Y 代替后者所使用的 X,其作用与意义相同,不再赘述。

④TYA——将变址寄存器 Y 的内容传送至累加器 A,即 $Y \rightarrow A$ 。TYA 指令和 TAY 指令配对使用可灵活地暂存和恢复累加器中的内容。

⑤TSX——将堆栈指示器 S 的内容传送至变址寄存器 X,即 $S \rightarrow X$ 。TSX 指令便于对堆栈进行灵活的操作。堆栈操作的优点是动态存取,其操作方式基本上是顺序执行的。灵活运用 TSX 指令,可使堆栈的依次操作改变为变址操作。

⑥TXS——将寄存器 X 的内容传送至堆栈指示器 S,即 $X \rightarrow S$ 。TXS 可以方便地改变堆栈指针,与其它指令配合使用,可增加程序对堆栈存取的灵活性。

以上我们介绍了取数指令、送存指令和传送指令,从各类传送指令可见,立即数和任一存储单元中的内容都可通过不同的寻址方式相互传送数据和交换信息;直接往寄存器 A、X、Y 传送;寄存器 X、Y、A 的内容也可以直接往存储器传送;寄存器 A 和 X 之间、A 和 Y 之间、X 和 S 之间也都可以相互直接传送。但应注意,各存储单元之间却不能直接交换数据。此外,还应注意标志位因各指令执行情况而改变,每条指令对标志位的影响能力是不一样的。

4. 堆栈指令

堆栈指令是在 MPU 与堆栈之间传送数据。由 MPU 向堆栈区存数叫进栈或压入;由堆栈向 MPU 送数叫出栈或弹出。在介绍堆栈指令之前,我们先简单介绍一下堆栈。它是计算机软件中一个非常重要的概念。

计算机中的栈,是存放数据的,它是在随机存储器 RAM 中开辟的一个特殊存储区,即从一个固定地址开始的一片连续单元。在这个存储区中,数据的存(进)、取(出)必须遵从“先进后出”或者说“后进先出”的规则,这好像货物一样,最先放入的货物,总是放在最底下,再放入货物时,往上堆;而取货时,总是把最上面的货物先取走(后进先出),最后才能取最底下的货物(先进后出)。这种特殊结构的存储区叫做堆栈。6502 规定堆栈设置在第 1 页中,即只能把 $\$0100 - \$01FF$ 这 256 个单元作为堆栈使用。在堆栈中,数据的存放是从

地址码高的单元到地址码低的单元依次进行的。最后存放数据的那个单元称为栈顶,栈的另一端则为栈底。当一个栈没有存放任何数据时,栈底就是栈顶,即 $\$01FF$ 单元。为了对堆栈中的数据进行存取操作,6502 把邻近栈顶的一个空单元的地址码存入堆栈寄存器 S(称为堆栈指针)中,而 S 的初值通常由 6502 设定在栈底位置 $\$01FF$ 处(即空栈时),由于堆栈指针 S 表明允许进行存取操作的当前位置,所以 S 在数据进栈时有自动减 1 的功能;而在数据出栈时则有自动加 1 的功能,可见它的作用类似程序计数器 PC。从这里也可以看出栈顶的位置,是随数据的进栈和出栈而上、下浮动的,其当前位置则由堆栈指针 S 反映出来。

现在我们介绍进栈和出栈的指令。进栈,是指将寄存器的内容送到堆栈指针所指的存储单元;出栈,是将堆栈指针所指向的存储单元的内容送入寄存器。进出栈指令只有四条,它们都采用隐含寻址方式,因而均为单字节指令。

①PHA——累加器进栈指令

把累加器 A 中的内容送入栈指针 S 所指向的栈顶空单元 M_s 中,然后栈指针 S 自动减 1。即 $A \rightarrow M_s, S - 1 \rightarrow S$ 。本指令不影响标志寄存器 P 的状态。

②PHP——标志寄存器 P 进栈指令。

把标志寄存器 P 的内容送入栈指针 S 所指向的栈顶空单元 M_s 中,然后栈指针 S 自动减 1。即 $P \rightarrow M_s, S - 1 \rightarrow S$ 。本指令也不影响标志寄存器 P 本身的状态。

③PLA——累加器出栈指令

它先使栈指针 S 加 1,得到栈顶的地址码,然后将此地址单元的内容送累加器 A。即 $S + 1 \rightarrow S, M_s \rightarrow A$ 。执行 PLA 指令要影响标志寄存器 P 中的 N、Z 位。

④PLP——标志寄存器 P 出栈指令

它先使 S 自动加 1,得到栈顶地址,然后从栈顶中取出内容送 P。即 $S + 1 \rightarrow S, M_s \rightarrow P$ 。由于标志寄存器 P 中加入了新的内容,所以该指令执行后, P 中的内容由出栈的那个单元内容决定。

进出栈指令的几点说明:

- 进栈指令 PHA, PHP, 主要用于保护现场,即保存 CPU 各寄存器的状态。

- 出栈指令 PLA, PLP, 主要用于恢复现场,即恢复寄存器 A、P 原来的状态。

- PHA 指令要与 PLA 指令配对使用。

- PHP 指令要与 PLP 指令配对使用。

- 寄存器 X、Y 没有堆栈指令,它们的进、出栈操作只能通过累加器 A 进行。

5. 移位指令

移位指令是将累加器或存储单元中的各位依次左、右移动或循环左、右移动。

①ASL——算术左移指令

此指令将累加器或选定的存储器字节内各位依次向左移一位,最高位(第 7 位)值移入标志位 C 中,最低位(第 0 位)补 0。ASL 指令执行后相当于把移位前的数乘以 2,因而称算术左移。由于最高位移进了 C,所以

ASL 指令要影响标志寄存器 P 中的 C 标志位,此外它还影响 N、Z 标志。ASL 指令有 5 种寻址方式。

如累加器 A 之内容为 7A,则执行累加器寻址指令 ASL A 后,累加器的内容变为 F4;若存储单元 \$ 3F86 的内容为 CB,则执行绝对寻址指令 ASL \$ 3F86 后,\$ 3F86 单元的内容改为 96。显然,执行一次 ASL 指令,相当于完成一次乘 2 操作。

②LSR——逻辑右移指令

LSR 的功能是将累加器或存储单元中各位依次向右移一位,最低位移入标志位 C 中,最高位补 0。

执行 LSR 指令对于字节中的无符号数或正数相当于除以 2,但对于负数并不适用,因最高位补 0,会把负数变成正数。所以 LSR 指令不能称为算术右移而称为逻辑右移。

例如,(\$ 04FA)=0D,则执行 LSR \$ 04FA 后,(\$ 04FA)=06,这是 \$ 0D 除以 2 后所得商的整数部分,而小数部分存放在 C 中,即 C=1。

由于最低位移进了 C,LSR 指令必然影响 P 寄存器中的 C 标志;由于最高位补 0,所以使标志 N=0;它还影响 Z 标志位。LSR 指令也有 5 种寻址方式。

③ROL——循环左移指令

ROL 的移位功能是将字节内容各位连同进位 C 一起依次循环左移一位,共有 5 种寻址方式,指令执行后影响标志位 N、Z、C。

例如,设(X)=16,(\$ 004A)=2E,标志位 C=1,则执行 ROL \$ 34,X 零页 X 变址后,存储器 \$ 004A 之内容变为 \$ 5D,而标志位 C=0。

④ROR——循环右移指令

ROR 的移位功能是将字节内容各位连同进位 C 一起依次循环右移一位,也有 5 种寻址方式,指令执行后对标志位影响同 ROL 指令。

例如,设(X)=14,(\$ 0114)=ED,标志位 C=0 则执行 ROR \$ 0100,X 绝对 X 变址后,存储器 \$ 0114 之内容变为 \$ 76,且 C=1。

至此,我们将数据传送类指令介绍完毕。

二、算术逻辑运算类指令

6502 内部的算术逻辑运算单元 ALU,是完成各种算术运算和逻辑运算的基本部件,它可以实现加、减、与、或、异或以及比较、移位,加 1,减 1 等操作。它的输入一个是累加器 A,另一个是存储器经由数据总线来的八位二进制数,此外,由于变址寄存器 X、Y 可以完成加 1、减 1 及比较等简单运算,所以数据和信息的传递主要在 A、M、X 及 Y 中间进行。运算过程中标志寄存器 P 的有关各位,将对运算结果直接发生不同的影响,又是实现条件跳转的依据,所以对 P 寄存器也应特别关照。

算术逻辑运算类指令共有 20 条,分为加法、减法、比较、加 1、减 1、标志位、逻辑运算和位测试等 8 种类别,现一一叙述如下。

1. 加法指令

ADC——累加器与存储器带进位相加,结果送入

累加器,即 $A+M+C \rightarrow A$ 。共有 8 种寻址方式,指令执行后,除 I、D、B 三个状态标志不变外,余下状态标志均根据上述相加的结果而变动。

例如,立即寻址方式指令 ADC # \$ 33,设相加前(A)=55,C=1,则执行 ADC # \$ 33 指令后,累加器 A 的内容变为 89。而 C 变为 0(因相加结果最高位上无进位)。其它标志位情况变动如下:结果不为 0,因而 Z=0;结果最高位是 1,则使负数标志 N=1;又由于相加过程中第 6 位和第 7 位中的一个发生进位(此例是第 6 位进位而第 7 位无进位),V 标志=1。即如果把两数视为带符号的正数,相加结果超出了—个字节所能表示的最大正数 7F,产生溢出。

ADC 是 6502 中唯一的一条加法指令,它会把进位位列入运算(6502 中没有不带进位的加法),因此,开始加法时,应在 ADC 指令之前予置进位标志 C=0(如安排一条 CLC 指令),才不会影响运算的结果。

ADC 指令既可以执行二进制加法,也可以做十进制加法。在 ADC 指令前再加一条指令 CLD,将十进制运算标志 D 置 0,则进行二进制运算;若加 SED,将十进制运算标志 D 置 1,则进行十进制运算。

2. 减法指令

SBC——累加器减去存储器及进位位的反码,结果送累加器,即 $A-M-\bar{C} \rightarrow A$ 。它有 8 种寻址方式,影响 P 寄存器各位情况同 ADC 指令。 \bar{C} 表示进位标志 C 取反, $\bar{C}=1$ 即说明有借位。

例如,零页寻址指令 SBC \$ 08,设(A)=01,(\$ 008)=02,C=1(即 $\bar{C}=0$,低字节无借位),则执行 SBC \$ 08 指令后,累加器 A 中为 FF(1111 1111,即-1 的补码)。由于最高位为 1,说明结果为负,因而使 N=1;结果不为 0,所以 Z=0;结果-1 未超出一个字节所能表示的范围,即无溢出,所以 V=0;又由于结果为负,说明不够减,最高位向高字节借位,因而使 $\bar{C}=1$,即 C=0。总之,SBC \$ 08 指令执行后各标志位是:N=1,Z=0,V=0,C=0。

注意,在开始执行单字节减法运算之前,应先执行 SEC 指令(它将进位标志 C 置 1),则 $\bar{C}=0$ 表示没有借位(清除借位)。这样,才不影响运算结果。

3. 比较指令

①CMP——累加器内容与存储器内容比较,结果不送累加器,即 $A-M$ 。该指令有 8 种寻址方式,指令执行后影响 N、Z 及 C 标志。

CMP 指令实际执行减法运算,但它与减法指令有两点区别;第一借位标志 C 不参加运算,因此,在使用 CMP 指令前不必置位 C 标志;第二运算结果不送入累加器 A,因而指令 CMP 执行后不改变 A 中内容。

CMP 指令常用来判断 A 和 M 中的内容是否相等或孰大孰小,因为执行 CMP 指令后,标志位 C、N、Z 将发生变化,特别是 C 的状况可以作为判别大小的标准,若 $C=1(\bar{C}=0)$ 表示够减无借位,说明 $A \geq M$;若 $C=0$ 表示不够减有借位,说明 $A < M$ 。

本指令最常出现在条件分支指令的前面,影响 N、

Z、C 标志位的状态,作为实现条件跳转的依据。

②CPX——变址寄存器 X 内容与存储器内容比较,结果不回送,即 $X-M$ 。其功能和 CMP 指令相似,只不过把累加器 A 换成 X。CPX 的寻址方式只有三种,影响标志位情况也是 N, Z, C。

③CPY——变址寄存器 Y 内容同存储器内容比较结果不回送,即 $Y-M$ 。其功能和 CPX 指令相同,只不过把变址寄存器 X 换成 Y。本指令也只有三种寻址方式,影响标志位情况同 CPX。

比较指令常与条件转移指令配合使用,实现程序的条件转移,在分支程序设计中重要的作用。

4. 加 1 指令

①INC——存储器的内容加 1,即 $M+1 \rightarrow M$,本指令有 4 种寻址方式,执行后仅对 N, Z 标志产生影响。

②INX——寄存器 X 内容加 1,即 $X+1 \rightarrow X$,本指令为隐含寻址方式单字节指令,影响标志位 N, Z。

③INY——寄存器 Y 内容加 1,即 $Y+1 \rightarrow Y$,本指令亦为隐含寻址方式指令,影响标志位 Z, N。

加 1 指令同将要介绍的减 1 指令,可以在多种应用上做为计数器用。

5. 减 1 指令

①DEC——存储器内容减 1,即 $M-1 \rightarrow M$,本指令有四种寻址方式,执行结果影响 N, Z 标志。

②DEX——变址寄存器 X 内容减 1,即 $X-1 \rightarrow X$,它采用隐含寻址方式,影响 N, Z 位。

③DEY——变址寄存器 Y 内容减 1,即 $Y-1 \rightarrow Y$,它采用隐含寻址方式,影响 N, Z 标。

6. 标志位指令

条件转移指令与状态寄存器 P 各标志位的值密切相关。P 寄存器各标志位一般由程序执行中有关指令操作情况自动产生。但它们也可以用指令设置。

①SEC——将进位标志 C 置 1: $1 \rightarrow C$ 。

②SED——将十进制运算标志 D 置 1: $1 \rightarrow D$,当 $D=1$ 时,执行十进制位运算。

③CLC——将进位标志 C 置 0: $0 \rightarrow C$

④CLD——将十进制运算标志 D 置 0: $0 \rightarrow D$,此时执行 2 进制运算。

⑤CLV——将溢出标志 V 置 0: $0 \rightarrow V$ 。6502 没有置 V 为 1 的指令,但可通过 BIT 指令来影响 V 标志。

7. 逻辑运算指令

逻辑运算指令共有三条,它们有三个共同点:逻辑运算操作是各位独立进行的,这是由逻辑运算指令本身的特有性质决定的,彼此不存在借位、进位关系;均有 8 种寻址方式;指令的执行只影响 N 和 Z 标志。

①AND——累加器与存储器内容逻辑“与”

AND 指令将累加器 A 的内容与选定存储器内容按位相“与”,结果送累加器,而存储器中的内容不变,即 $A \wedge M \rightarrow A$ 。这里 M 可以是一立即数,也可以是某一存储单元的内容。

AND 指令常用于屏蔽某些位或者取出某些位。

例如,已知 $(\$040)=3D$,要求将其最低 4 位存放

至 $\$0041$ 单元中,而最高 4 位清 0,则用程序:

```
0300—A5 40    LDA $40
0302—29 0F     AND # $0F
0304—85 41     STA $41
0306—60        RTS
```

又如,已知 $(\$4000)=AD$,要求将其 d_5 位存于 A 中,则可用下列指令实现:

```
0300—AD 00 40  LDA $4000
0303—29 20     AND # $20
0305—00        BRK
```

②ORA——累加器与存储器内容逻辑“或”

ORA 指令将累加器 A 的内容与指定存储器内容按位“或”,并将结果送累加器,存储器中的内容不变,即 $A \vee M \rightarrow A$ 。M 的意义同上。

ORA 指令常用于使某些位置 1,或求混合信息。例如,执行 $ORA \# \$80$,会无条件地将累加器 A 的最高位置为 1。欲将 $\$8000$ 单元中的高位置为 1,低四位不变,可用 $LDA \# \$8000$ 和 $ORA \# \$80$, $STA \# \$8000$ 三条指令实现。

③EOR——累加器内容与存储器内容“异或”

EOR 指令将累加器内容与指定存储器内容按位“异或”,并将结果送累加器 A,而存储器内容不变,即 $A \oplus M \rightarrow A$ 。

EOR 指令可用于求反码,6502 没有专门求反码指令。

例如,执行 $EOR \# \$FF$ 指令,可使累加器内容的每一个“1”变“0”,而“0”变“1”。若将 $\$7000$ 单元的内容求反后送 $\$8000$ 单元,则可用 $LDA \# \$7000$, $EOR \# \$FF$ 及 $STA \# \$8000$ 三条指令实现。

8. 位测试指令

BIT——用累加器检测存储器各位

BIT 指令将累加器 A 中的内容与指定存储单元的内容按位相“与”,存储器的最高位送 N 标志,次高位送 V 标志,并根据相“与”结果是否为 0 来设置 Z 标志。当 $A \wedge M=0$ 时, $Z=1$;当 $A \wedge M \neq 0$ 时, $Z=0$ 。注意指令 BIT 执行后,累加器与存储器中的内容均不改变。该指令只有二种寻址方式。

如 $(A)=A6$, $(\$1641)=E0$,则执行 $BIT \# \$1641$ 指令后, $N=1$, $V=1$, $Z=0$ 。

至此,以上对 20 条算术逻辑运算指令作了介绍。

三、程序控制类指令

程序控制类指令包括程序转移指令和 CPU 控制指令两个主要部分。它们的共同特点是能改变程序顺序执行的次序,从而使程序计数器 PC 不再沿着不断自动加 1 的操作向下运行,而是使 PC 的内容发生跳转变,转入新的地址去执行存放的指令。转移指令的设置体现了分支程序设计的要求。程序控制类指令共有 16 条,是程序设计中十分有用的操作,灵活和巧妙地使用这些指令,体现了程序设计的技巧。下面按条件转移、无条件转移、子程序及返回、中断和空操作等几种类型逐一介绍。

1. 无条件转移指令

无条件转移指令只有 JMP 一条,它有两种寻址方式。JMP 指令不参考标志寄存器 P 中的条件,因此称为无条件转移指令。它用于控制程序执行的路线,将程序由通常的执行顺序转另一个指定的目标地址。

如,对于绝对寻址方式指令 JMP \$ 6000,CPU 执行:00→PC_L,60→PC_H 操作,而转向 \$ 6000 单元执行指令。对于间址寻址方式指令 JMP(\$ 6000),CPU 执行:(\$ 6000)→PC_L,(\$ 6000+1)→PC_H 操作,而转向 PC_H PC_L 所指向的单元执行指令。

2. 条件转移指令

条件转移指令有四个特点:

- 条件转移指令共有 8 条,它们实际上是完成对指令计数器 PC 重新赋值的操作。

- 所有条件转移指令都是根据标志寄存器 P 中的某一状态值来决定是否执行转移操作。

- 条件转移指令均采用相对寻址方式,一律都是两字节指令。前者操作码,后者偏移量。

- 条件转移指令都是根据前一条指令的执行结果而决定程序的走向,执行后不改变各标志位状态。

①BEQ——结果为 0(Z=1)时转移,否则继续。

②BNE——结果非零(Z=0)时转移,否则继续。

③BCC——进位标志 C 为 0 时转移,否则继续。

④BCS——进位标志 C 为 1 时转移,否则继续。

⑤BPL——结果为正数(N=0)时转移,否则继续。

⑥BMI——结果为负数(N=1)时转移,否则继续。

⑦BVS——有溢出(V=1)时转移,否则继续。

⑧BVC——无溢出(V=0)时转移,否则继续。

3. 子程序指令

①JSR——转子指令

执行该指令后,CPU 转到该指令操作数所指出的地址去执行子程序。它只有绝对寻址一种方式,格式为 JSR \$ nHnL,执行过程第一步是将 JSR \$ nHnL 机器码第三字节在存储器中的地址保存到堆栈中,以备执行子程序返回指令时取出;第二步是把 \$ nHnL 送 PC,CPU 根据 PC 的指示转移到子程序入口,执行子程序。

转子程序也是无条件转移指令,但它与 JMP 指令有区别。JMP 指令控制程序转出后不再返回,而转子指令在子程序执行完毕时还要返回主程序被打断处(断点)。

②RTS——返主指令

执行该指令后,由子程序返回到主程序断点处。它是采用隐含寻址方式的单字节指令,通常放在子程序的末尾,以便子程序执行完后正确返回主程序。

RTS 指令执行过程是:从堆栈中连续取出栈顶两个单元的内容,它们正是执行 JSR 指令时保存在堆栈中的断点地址,将此地址送 PC,PC 自动加 1 修正后得到返回地址(JSR 指令后面一条指令的地址),CPU 根据 PC 的内容,返回主程序继续执行。

4. 中断指令

中断指令共有 4 条:即 CLI,SEI,BRK,RTI。它们均

采用隐含寻址方式。

①CLI——开中断指令

将中断禁止标志 I 置 0。当外设有中断请求时,如果 I=0,则 6502 响应中断而转入中断服务子程序。所以 CLI 的功能就是使中断请求被允许。

②SEI——关中断指令

将中断禁止标志置 1。此时,6502 不响应外设的中断请求,即处于关中断状态。因此,指令 SEI 的功能就是关中断,即中断请求被禁止。

③RTI——中断返回指令

6502 响应外设中断请求时,要将断点地址 PC 和 P 寄存器的内容进栈保护起来,然后转移到中断服务程序。等到服务程序执行完毕也要返回主程序,这个返回是依靠在服务程序的最后放一条 RTI 指令来实现。RTI 指令的功能是从堆栈中取回在响应中断时保存的 P 及 PC 值,从而恢复了标志状态及返回断点。

④BRK——软件中断指令

BRK 指令能强使 CPU 中断程序的执行,故称它为软件中断命令。执行 BRK 指令时 B 标志置 1,此时 CPU 处于软件中断状态;同时 I 也置 1,即 CPU 禁止对外设中断申请的响应。

5. 空操作指令

空操作 NOP 也是单字节隐含寻址方式指令。CPU 执行 NOP 指令,除了使 PC+1→PC,并占用两个时钟同期外,不做其它操作,故称为空操作指令。

NOP 指令主要用于调试程序,例如,在调试一段含有转子指令 JSR 的主程序时,为了调试方便,可暂用三条 NOP 指令代替主程序中 JSR 指令所占用的三个字节,而当主程序调好,再重新换上 JSR 指令。又如暂时删除某条指令,可用 NOP 指令替代,等程序统调好时再换上。这样可避免调试中由于地址变化而造成的混乱和错误。

各条指令加上不同的寻址方式后,总共 151 条指令,它们的指令格式、功能、寻址方式、机器码、影响标志位情况、执行周期等情况,请参看 6502 指令表。

电 脑 报

●集实用性、知识性、资料性、趣味性于一体,可读性强,可长期保存、查阅。

●内容丰富、取材新颖、通俗易懂、寓教于乐,由浅入深地把你引入奇妙的电脑世界。

●主要栏目:电脑写作、电脑与信息、信息与社会、名人谈、软件服务台、维修小窍门、购机指南、现代办公设备、热门软件、实用电脑资料……

●全国邮局均可订阅,月价:0.68 元,年价:8.16 元

●社址:重庆市双钢路 3 号,邮编:630013

操作系统及 PC-DOS

北京电脑天地学校(100051) 宋丹颖

自测题

一、判断下列叙述是否正确,请在相应的题号前画√(正确)或×(错误)。

1. 操作系统是键盘命令的集合,它用于控制和管理计算机系统的硬件资源和软件资源。

2. 为提高计算机处理机和外部设备的利用率,把多个程序同时放入主存储器,这种方法称为多道程序设计。

3. 接收和完成用户对数据库的存取请求的系统称为数据库系统。

4. 系统响应时间的重要性超过系统资源的利用率,被广泛地用于生产过程控制、武器控制、情报检索、飞机订票业务等领域的是分时操作系统。

5. 兼容机之间指令系统基本上是相同的,但硬件实现方法可以不同。

6. 一般 1.2MB 的软盘驱动器既能对 1.2MB 软盘进行操作,也能对 360KB 软盘进行操作。

7. 在多级目录结构中,不允许两个不同的文件具有相同的名字。

8. 主要用于 PC 系列机上的微机操作系统 DOS 是单用户、单任务操作系统,它可以用于实时处理,具有批处理功能,一般不能分时。

9. 内部命令是在系统启动时由装入程序读入内存并常驻内存的命令。

10. DOS 规定,指定目录路径有两种方法,其中相对路径是指从子目录开始到文件所在目录的路径。

二、从供选择的答案中选出正确答案的编号填写在横线上。

1. 操作系统是对计算机系统的全部资源进行控制与管理的系统软件。系统资源指的是_____。

供选择的答案:

- A. 软件、数据、硬件、存储器
- B. 处理机、存储器、输入/输出设备、信息
- C. 程序、数据、输入/输出设备、中央处理机
- D. 主机、输入/输出设备、文件、外存储器

2. PC-DOS 系统的命令处理程序 COMMAND.COM 的功能是_____。

供选择的答案:

- A. 负责接受和解释用户输入的命令及出错处理,具有全部内部命令的处理程序
- B. 负责出错处理,具有全部内部命令、外部命令的处理程序

C. 负责接受和解释用户输入的命令,具有全部内部命令、外部命令的处理程序

D. 负责出错处理,具有全部外部命令的处理程序

3. 当系统显示如下错误信息:Bad command or file name 时,应从两方面进行检查_____。

供选择的答案:

A. 磁盘驱动器是否关好小门;磁盘是否贴有写保护

B. 磁盘是否贴有写保护;命令字拼写及格式是否正确

C. 查看命令及格式拼写是否正确;查看磁盘上是否有该文件

D. 内存中是否有该文件,是否为可执行文件

4. AUTOEXEC. BAT 文件是自动批处理文件,它的主要内容是_____;CONFIG. SYS 文件内容是_____。

供选择的答案:

A. 建立或编辑文件的一系列子命令 B. 设置系统结构的命令清单 C. 启动后希望立即自动执行的一系列命令 D. 合并单独产生的目标模块,控制被调试程序运行的命令

三、从供选择的答案中选出与下列叙述关系最密切的字句。

1. 在操作系统的存储管理中,覆盖和交换是[A],用以进行存储扩充;设备管理中的缓冲技术是[B],用以提高 CPU 和外设的并行工作能力。对磁盘的管理属于[C];对磁盘中信息的管理属于[D]。

A, B: ①以空间换取时间 ②以时间换取空间
③充分利用时间 ④充分利用空间

C, D: ①设备管理 ②文件管理 ③驱动器管理
④作业管理

2. 硬盘和软盘是目前常见的两种存储媒体,在第一次使用时[A]。在进行格式化时要注意,认为[B]是错误的。

在操作系统中使用删除目录命令删除一个目录时,欲删除的目录应该是[C]的目录。

在操作系统中文件复制命令从甲目录复制一个文件到乙目录,那么甲目录和被复制的文件都必须是[D]的,乙目录必须是[E]的。

供选择的答案:

- A: ①可直接使用,不必进行格式化
- ②只有硬盘才必须先进行格式化

③只有软盘才必须先进行格式化

④都必须先进行格式化

B: ①“不同操作系统下格式化的软盘是不可通用的”

②“写保护装置起作用的磁盘无法被格式化”

③“格式化一个磁盘将破坏磁盘上的所有信息”

④“在 IBM-PC 机 DOS 下被格式化过的磁盘不能再在其它种类的操作系统下被格式化”

C: ①有文件 ②有子目录 ③有多级子目录

④空

D: ①可读 ②可写 ③写保护

④读保护

E: ①可读 ②可写 ③写保护

④覆盖

四、设当前系统提示符为 C>, 请建立批处理文件, 它具有的功能如下:

1. 自动在用户盘(放在 B 驱动器内)上建立一个名为 A1 的子目录;

2. 把系统盘(放在 A 驱动器内)上的 WB1. TXT 文件复制到 A1 子目录下;

3. 在用户盘上再建立一个名为 A2 的子目录;

4. 把系统盘上的 WB2. TXT 文件复制到 A2 子目录下;

5. 把两个子目录下的. TXT 文件连接为一个 BIG. TXT 文件放在 A1 子目录下, 要求 WB1. TXT 在前, WB2. TXT 在后;

6. 在屏幕上显示本批处理文件自身的内容;

五. 现有 A 盘目录结构如下:

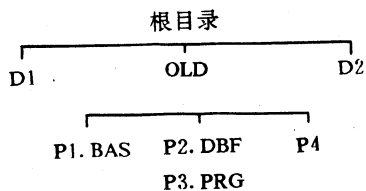


图 1

请完成以下操作, 把子目录 OLD 下的文件分别复制到新建的各子目录下, 并删除旧的子目录 OLD。请按下列步骤写出命令序列。

1. 在子目录 D1 下建立一个名为 BAS 的子目录, 把 OLD 下的 P1. BAS 文件复制到新建的子目录下;

2. 在子目录 D1 下建立一个新的子目录 DB, 把 OLD 下的 P2. DBF 和 P3. PRG 文件复制到新建的子目录下;

3. 在子目录 D2 下建立一个名为 WS 的子目录, 把 OLD 下的 P4 文件复制到新建的子目录下;

4. 删除 OLD 子目录下的全部文件;

5. 删除子目录 OLD;

6. 画出更改后的树形目录结构图。

分析及答案

一、分析

• 28 •

1. 错误。操作系统是控制和管理计算机硬件和软件资源、合理地组织计算机工作流程以及方便用户的程序的集合。它为用户提供的接口一般有两种:一种是程序一级的各种服务, 提供给用户一组系统调用; 另一种是提供各种键盘操作命令和作业控制语言, 这只是一方面, 因此操作系统不仅仅是键盘命令的集合。

2. 正确。

3. 正确。

4. 错误。本叙述中描述的特点及应用都适用于实时系统而不是分时系统。实时系统的特点是时间性强, 响应快, 可靠性高。

5. 正确。

6. 正确。

7. 错误。在多级目录中, 对不同子目录下的同名文件可根据用户给出的检索路径进行查找, 不会造成混乱。而在一级目录结构中, 所有文件说明都登记在目录表中, 用户仅根据文件名使用文件存储空间, 不允许两个不同的文件具有相同的名字。

8. 正确。

9. 正确。

10. 错误。DOS 规定指定目录路径有两种方法, 其中相对路径是指从当前目录开始到文件所在目录的路径。为缩短检索时间, 操作系统为用户在内存中开辟了目录文件区域——当前目录, 作为查找文件时无须特别指出的现行目录。叙述中说相对路径是指从子目录开始到文件所在目录的路径是不正确的。

答案: 1. × 2. ✓ 3. ✓ 4. × 5. ✓

6. ✓ 7. × 8. ✓ 9. ✓ 10. ×

二、分析

1. 从资源管理的观点看, 操作系统是控制和管理计算机资源的软件。通常计算机系统的资源分为: 处理机、存储器、输入/输出设备(这些是硬件资源)和数据信息(软件资源)四类。相应的操作系统包括处理机管理、存储管理、设备管理和信息管理等几部分。

2. PC-DOS 采用层次模块结构。它的核心由三个层次模块和一个引导程序组成。三个层次模块中的命令处理程序 COMMAND. COM 由暂驻内存、常驻内存和初始化三部分构成。COMMAND. COM 的功能是接受和解释用户输入的命令, 错误中断和键盘中断处理, 并具有全部内部命令的处理程序。对外部命令, 它负责把相应的命令文件读入内存, 然后把控制权交给被调用的程序。

3. 当系统显示: Bad command or file name 的错误信息时, 表明键入的命令不是一条合法的 DOS 命令。这时应检查命令的拼写是否正确, 若有错, 应重新键入。如果命令没有拼写错误, 可再查看该驱动器内的磁盘上是否有用户要执行的外部命令或批处理文件。因为外部命令在 DOS 启动后并未装入内存, 而是以程序文件的形式存在磁盘上, 当执行该命令时才由磁盘装入内存。若盘上根本不存在这个文件而要用这个命令, 就会出现以上错误信息。磁盘贴有写保护胶带时并不影

响从盘上读出文件。

4. AUTOEXEC. BAT 文件是自动批处理文件,它是一个特殊的自动型批文件。当系统启动时,PC-DOS 会检查盘上是否有这个文件,如果有,则执行该文件中的命令,自动建立系统所需要的状态;如果没有,则提示输入系统日期、时间。由于它具有一启动就自动执行的特点,所以它的主要内容是启动后希望立即执行的一系列命令,用以建立开机就需要的系统状态。CONFIG. SYS 是一个系统结构设置文件。每当 DOS 启动时,都会查找盘上是否有这个文件。若有, DOS 就按照文件中的命令设置系统;若没有,则 DOS 把系统的默认值赋予各配置命令。所以, CONFIG. SYS 文件的内容是设置系统结构的命令清单。用户在应用中感到默认值不能满足需要时,可建立或修改 CONFIG. SYS 文件予以重新设置。

答案: 1. B 2. A 3. C 4. C, B

三、分析

1. 操作系统中的内存扩充是通过采用覆盖、交换、虚拟存储等技术实现的。覆盖是解一个问题时在不同阶段反复使用同一内存区的技术。交换是指允许一个作业装入内存还能把它调入或调出内存。这两种方法都是把要运行的程序装入内存,把准备运行的程序放在外存。基本指导思想是以时间换取空间,虽然内、外存的交换花费了时间,但却换来了空间,使用户可以运行比内存还要大的程序。设备管理中的缓冲技术通常采用在内存中划出具有 n 个单元的区域充当缓冲器来实现。它的指导思想是以空间换取时间,通过建立暂存数据的存储装置以解决高速的 CPU 与低速的输入/输出设备之间速度不匹配的问题。对磁盘的管理属于操作系统中的设备管理,而对磁盘中存储信息的管理属于文件管理的范畴。

2. 软盘价格便宜,携带、使用方便;硬盘容量大,运行速度快,它们都是目前微机上常用的存储媒体。无论软盘还是硬盘,在第一次使用时都必须先进行格式化。所谓格式化,就是在磁盘上建立起文件目录表文件分配表,按照 DOS 的要求,将软盘划分为若干磁道和扇区或将硬盘划分为若干柱面和扇区,为接收信息做好准备。“空白”盘经过格式化才能记录信息。旧的软盘,若有损坏的磁道,为了把损坏的磁道剔除,也必须对该盘进行格式化。由于不同的操作系统对磁盘设定格式、划分区域不同,所以不同操作系统下格式化的软盘不能通用。如在 PC-DOS 下被格式化过的磁盘要在其它种类操作系统下使用则需要在该操作系统下重新进行格式化。

在具有多级目录结构的系统中若要删除一个目录时,要求被删除的目录必须为空(即该级目录不包含任何文件及下一级子目录)。换句话说,在删除目录之前,必须先删除该目录中的所有文件,然后再删除目录。

使用操作系统中的文件复制命令把一个文件从甲目录复制到乙目录,那么称甲目录中被复制的文件为源文件,复制到乙目录中的副本为目标文件。对于

“源”来说,是从中读取信息,传出信息;而“目标”则是要把信息传输到其上,要写入信息。所以,被复制的甲目录和该目录下的文件必须是可读的,而乙目录必须是可写的。

答案:

1. A② B① C① D②
2. A④ B④ C④ D① E②

四、分析

批处理文件是把若干条要执行的命令像编程序一样写入该文件,使用时只要键入批文件名便能自动执行文件中的一批命令。编写批文件时要根据每一条功能要求选用相应的 DOS 命令去完成。例如,题中第五点要求把两个子目录下的. TXT 文件连接成一个文件放在子目录 A1 下,并要求连接后的文件要按指定的先后次序存放。首先确定连接文件要用 COPY 命令,并选用连接复制格式。在写参加连接的源文件时,要注意文件的前后次序。同理,可写出其它命令。

解: C>COPY CON B:P1. BAT

```
MD B:\A1
COPY A:WB1. TXT B:\A1
MD B:\A2
COPY A:WB2. TXT B:\A2
COPY B:\A1\WB1. TXT + B:\A2\
WB2. TXT B:\A1\BIG. TXT
TYPE B:P1. BAT
^ Z
```

C>TYPE B:P1. BAT

五、解:

```
A>MD \D1\BAS
A>COPY \OLD\P1. BAS \D1\BAS
A>MD \D1\DB
A>COPY \OLD\P2. DBF \D1\DB
A>COPY \OLD\P3. PRG \D1\DB
A>MD \D2\WS
A>COPY \OLD\P4 \D2\WS
A>CD OLD
A>DEL *.*
A>CD..
A>RD \OLD
```

更改后的树形目录结构图:

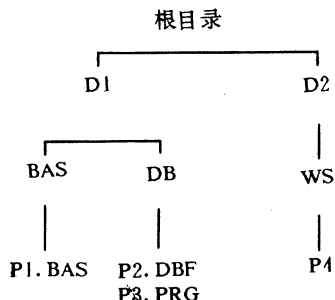


图 2



实现单片机最小系统与 PC 机的通信

合肥中国科技大学计算中心(230026) 张培仁

PC 机和多个单片机也可以组成多级控制系统,一般以 PC 机为后级,多个单片机最小系统为前级。一般它们通信是由两个独立的模块组成,即单片机通信模块和 PC 机通信模块。

单片机通信模块的设计:

MCS-51 单片机内串行口部分有两个物理上相互独立的数据缓冲器 SBUF。但两个缓冲器的地址是一个。地址是 99H。该缓冲器用来发送和接收数据。专用寄存器 SCON 和 PCON 控制串行口的工作方式和通信的波特率设定。定时器工作为波特率发生器。CPU 向 SBUF 写数据即是发送数据,CPU 从 SBUF 读数据就是接收数据。

单片机 MCS-51 串行口是全双工串行通信口。

串行口控制寄存器是 SCON。

SCON

D7	D6	D5	D4	D3	D2	D1	D0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0, SM1, 工作方式选择位,如下表。

SM0	SM1	工作方式
0	0	串行口作为移位寄存器使用
0	1	8 位 UART, 波特率可变
1	0	9 位 UART, 波特率固定二种
1	1	9 位 UART 波特率可变

表中 UART 是异步串行通信。

SM2: 允许方式 2, 3 多机通信。

REN: 允许串行接收。

TB8: 在方式 2 和方式 3 时是发送的数据的第 9 位。

RB8: 在方式 2 和方式 3 时接收数据的第 9 位。在方式 1 时,若 SM2=0, RB8 为接收到数据的停止位。

TI, RI: 发送,接收中断标志。表示是否已发送完或接收到数据。由硬件置位,软件清 0。

因为 PC 机的 RS-232 接口是由 8250 芯片控制的,所以一般 PC/XT, AT 机用 RS-232 与其他设备通信时所使用是 8 位数据位(可以少于 8 位数据),由此决定单片机必须设置方式 1。

工作方式 1: 当 SM0, SM1=0 时,串行接口被选择为工作方式 1,此时为可变波特率的 8 位异步通信方式。

发送数据 TXD 端输出,每一帧信息为 10 位,一位起始标志位 0, 8 位数据位和一位停止位 1。发送时数

据先送到发送缓冲器 SBUF,然后启动发送。数据发送完后,将中断标志 TI 置“1”。

接收时(REN=1),数据从 RXD 输入,当采样到第一个 1 到 0 的下跳沿时启动接收器,确认起始位后,接收一帧信息。当 RI=0,停止位为 1 或 SM2=0 时,停止位进入 RB8,中断标志位 RI 置“1”。接收了一帧数据。

波特率的设定:

定时器工作为波特率发生器。为此须设定定时器 1 是工作方式 2,即自动重装模式。TL1 作为 8 位计数器,TH1 作为常数缓冲器。当 TL1 计数溢出时,将 TH1 中常数送到 TL0 中,使 TL1 再次从初值开始重新计数。定时器 1 产生固定频率占空比是 1:1 的脉冲,波特率可由设定时间常数确定。

波特率=(定时器 1 的溢出率) * 1/32(或 * 1/16)

1/32, 1/16 选择由波特率选择寄存器 PCON 的第七位确定。PCON. 7=0 是乘 1/32, PCON. 7=1 乘 1/16。

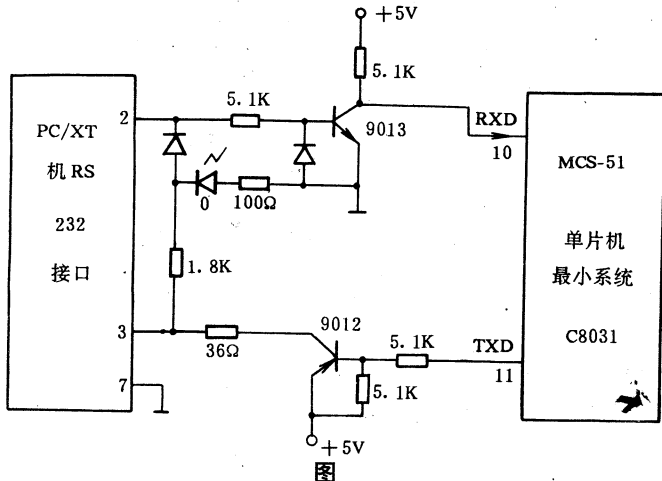
定时器 1 的溢出率=计数速率/(256-定时器计数常数)

计数速率=fosc/12。其中 fosc 是主振荡频率,我们选用是 6MHz。

当波特率是 1200 时,这时 TH1=TL1=0F3H

通信所用线是一般双绞电源线。通信距离一般在 30 米以内。如果通信距离比较远,要考虑用光电隔离或用电流环来驱动,这时一般 1000 米,波特率 300 时问题不大,通信是可靠的,对通信线也无特殊要求。

通信的硬件环境和接口电路如图所示。



PC/XT 机 RS-232 接口输入输出使用 MC1488, MC1489。我们用 9013, 9012 两只晶体管把 RS-232 的标准电平 -12V 表示 1, +12 表示 0 转变成 0 至 5V 的

电平,以便和 8031 的最小系统中 8031 进行联接。D 是发光二极管,PC 机和单片机通信时,D 有明暗变化,表示正在传送数据。这种电路是准 RS—232 接口电路。

单片机通信模块包含三个子模块。

初始化模块:置定时器 1 的工作方式,置计数初值,定义波特率,定义串行口的工作模式等。

发送模块:将累加器 ACC 中的内容发送到串行口。

接收模块:接收串行口上的数据,送到累加器 ACC 中。

PC/XT 机通信模块简介如下:

PC/XT 机上的 RS—232 标准串行口中是由 8250 异步通信控制芯片控制的。端口地址为 3F8—3FE。直接用 IN,OUT 指令对口地址操作,访问 8250。

8250 中有数据发送保持器和数据接收缓冲器,通信线控制寄存器,通信线状态寄存器,除数锁存器等。这些分别由软件置数。

PC 机通信模块也分 3 个子程序模块。

初始化模块:定义波特率,定义串行口数据格式。

接收模块:将数据接收缓冲器的内容写入指定文件中。

发送模块:将指定的文件内容送给数据发送保持器。

PC 机和单片机通信的运行步骤如下:

当单片机是以最小系统为基础的开发机时,首先把汇编好的单片机通信程序送入单片机片外 RAM 某地址处如 2000H 为开始的内存中。并运行这个单片机通信程序(文件名 DP. ASM),这时单片开发机的监控程序已停止工作。

其次在 PC 机上编辑一个很直观的数据文件,例如 ABC. ASM 文件。内容如下:

```
ORG 2100H
DB 00H, 11H, 22H, 33H, 44H, 55H
DB 66H, 77H, 88H, 99H, 0AAH, 0BBH
DB 0CCH, 0DDH, 0EEH, 0FFH
END
```

把 ABC. ASM 进行汇编产生 ABC. OBJ 文件。

第三在 PC 机运行 PC 机通信模块程序。文件名是 ZHANKAN. PAS。这时在 PC 显示屏上出现提示,并要选择(1—3)。功能 1 是 PC 机向 8031 送数据。2 是 8031 向 PC 机传送数据。3 是返回 DOS。我们先选择 1,把 ABC. OBJ 文件传到开发机(或最小系统)的 2100H 为开始的地址处。PC 机传送完 ABC. OBJ 后又返回到(1—3)通信的提示状态。这时选择 2,即把单片机的外存 RAM 中 2100H 处的 16 个(十进制数)字节内容传回 PC 机,并形成另一个文件。文件名是 ABC1. OBJ。最后再返回 DOS。这时用 DOS 的 TYPE 命令分别显示 ABC. OBJ 和 ABC1. OBJ 二个文件的内容,从而比较是否一致。以便证明通信过程是否正常。

如果通信不在开发机上进行,只要把单片机的通信程序写入 EPROM 中,并通过按键或启动复位执行

通信程序,其他步骤同上,即可完成相应通信功能。

这个 PC 机和单片机的通信模块应用是很广的。如果再增加一些命令控制,将是一个很好的控制系统。KDC—Ⅲ 型开发机的在线仿真机的通信程序就是在此基础上扩充的。

DP. ASM, ZHANKAN. PAS 两个通信程序全部清单如下:

```
FILE NAME DP. ASM

org 2000h
anl tmod, # 0f0h
orl tmod, # 020h ;定时器 1 置方式 1
mov th1, # 0f3h
mov tl1, # 0f3h ;置时间常数
clr pcon. 7 ;清除波特率倍增位
setb tr1 ;启动定时器 1
clr es
clr etl ;禁止中断
mov scon, # 052h ;置串行口工作方式
pd: lcall load ;判断接收
jz frompc ;结束
cjne a, # 01h, pd ;发送
ajmp topc

load: jnb ri, load ;如果 n=0 则等待
clr ri ;清除接收中断请求位
mov a, sbuf ;接收数据送 a
ret

rd: lcall load ;将起始地址送地址寄存器
mov dph, a
lcall load ;
mov dpl, a ;
lcall load
subb a, dph ;用末址减首址得总字节数
mov r0, a ;送 r0, r1
inc r0 ;
lcall load
subb a, dpl
mov r1, a
ret

frompc: lcall rd
jnc f1 ;无进位转 f1 否则 r0-1
dec r0 ;
f1: lcall load ;
movx @dptr, a ;接收的数据送外存
inc dptr ;指向下一个地址
djnz r1, f1 ;
djnz r0, f1 ;文件没有接收完,
ajmp pd ;继续接收

topc: lcall rd
jnz t1 ;进位位=0 转 t1 否则
dec r0 ;r0-1→r0
t1: movx a, @dptr ;发送
lcall send
inc dptr
djnz r1, t1
djnz r0, t1
```

```

    ajmp pd
send:    jnb ti,send
        clr ti
        mov sbuf,a
        ret
good:    end

```

FILE NAME ZHANKAN.PAS

```

program ps(input,output);
uses crt;
label 10;
var
    file;file of char;
    st:string[12];
    a,b,x,c3:byte;
    ch,m,n:char;
    cl,c2,i,h:word;
procedure ws;
begin
    port[$3fb]:=$80;    ;定义波特率
    port[$3f9]:=$0;    ;数据格式
    port[$3f8]:=$60;
    port[$3fb]:=$03;
end;
procedure change(j:char;var k:byte) ;字符⇒字节
begin
    if j in ['0','1','2','3','4','5',
             '6','7','8','9'] then k:=ord(j)-ord('0')
    else if j in ['a','b','c','d','e','f']
        then k:=ord(j)-ord('a')+10
    else if j in ['A','B','C','D','E','F']
        then k:=ord(j)-ord('A')+10
    end;
procedure send(a:byte);    ;发送数据
begin
    ;⇒保持器
    while (port[$3fd] and $40)<>$40 do;
    port[$3f8]:=a;
end;
procedure ww(j:byte);    ;字节⇒字符
const yy:array[0..15]of    ;写入文件中
    char=('0','1','2','3','4','5','6',
          '7','8','9','A','B','C','D','E','F');
begin
    a:=j div $10;
    write(file,yy[a]);
    b:=j mod $10;
    write(file,yy[b]);
end;
procedure hh(var h:word);    ;读地址
begin
    readln(st);
    h:=0;
    for i:=1 to length(st) do
        begin

```

```

            change(st[i],x);
            h:=h*$10+x;
        end;
end;
procedure cc(var a:byte);    ;接收数据
begin
    while (port[$3fd] and $01)<>$01 do;
    a:=port[$3f8];
end;
procedure pctomcs;    ;PC 机发送模块
begin
    clrscr;
    gotoxy(20,10);
    write('Please input object file name:');
    readln(st);
    gotoxy(20,20);
    write('Waiting...');
    {$i-}
    assign(file,st);
    reset(file);
    {$i+}
    x:=ioresult;
    if x=0 then
        begin
            while not eof(file) do
                begin
                    read(file,m);
                    if not eof(file) then
                        begin
                            read(file,n);
                            change(m,a);
                            change(n,b);
                            a:=a*$10+b;
                            send(a);
                        end;
                    end;
                gotoxy(40,20);
                writeln('O. K. ');
                close(file);
            end else
                begin
                    gotoxy(25,23);
                    writeln('File',st,'can not be opened. ');
                    ch:=readkey;
                end;
            end;
        end;
procedure mstopc;    PC 机接收模块
begin
    clrscr;
    gotoxy(20,10);
    write('Please input first address:');
    gotoxy(20,12);
    write('Please input end address:');
    gotoxy(20,14);
    write('Please input receive object

```

```

file name:');
gotoxy(48,10);
hh(c1);
writeln;
gotoxy(46,12);
hh(c2);
gotoxy(60,14);
readln(st);
gotoxy(20,22);
write('Waiting...');
assign(filel,st);
rewrite(filel);
ww(hi(c1));
ww(lo(c1));
ww(hi(c2));
ww(lo(c2));
send(hi(c1));delay(50);
send(lo(c1));delay(50);
send(hi(c2));delay(50);
send(lo(c2));
for i:=1 to c2-c1 do
begin
cc(c3);
ww(c3);
end;
gotoxy(40,23);
writeln('O. K. ');
close(filel);
end;
BEGIN
10: clrscr;           ;主程序
gotoxy(22,2);        ;菜单选择
write(' * * * * * ');
gotoxy(22,3);
write(' * * * * * ');
gotoxy(22,4);
write(' * * * * * ');
gotoxy(25,4);
writeln('1. Move date from pc to 8031');

```

```

gotoxy(22,5);
write(' * * * * * ');
gotoxy(22,6);
write(' * * * * * ');
gotoxy(25,6);
writeln('2. Move date from 8031 to PC');
gotoxy(22,7);
write(' * * * * * ');
gotoxy(22,8);
write(' * * * * * ');
gotoxy(25,8);
writeln('3. Exit');
gotoxy(22,9);
write(' * * * * * ');
gotoxy(22,10);
write(' * * * * * ');
gotoxy(25,14);
write('Please select(1..3):');
ch:=readkey;
if ch<>'1' then if ch<>'2' then
if ch<>'3' then goto 10;
ws;
case ch of
'1':begin
send($00);
pctomes;
goto 10;
end;
'2':begin
send($01);
mcstopc;
goto 10;
end;
'3':begin
send($11);
halt;
end;
end;
END.

```

(上接43页)

上运动。可在同一个画面上出现16个卡通图案，八个静止的，八个运动的。

(2)解除 DEF MOVE 语句的作用使用 CAN 语句。解除 DEF SPRITE 语句的作用使用 SPRITE 语句，因为 SPRITE OFF 语句虽能令卡通图案消失，但它没有选择消失的功能，所以必须使用 SPRITE n 语句。

(3)设定卡通图案的显示座标，即显示用 DEF SPRITE 定义的卡通图案用 SPRITE。设定卡通图案运动座标用 POSITION。

(4)我们学习了三个设定座标语句 LOCATE、POSITION、SPRITE，必须分清三个语句的适用对象和取值范围。LOCATE 语句设定打印座标，取值范围 X:0~27,Y:0~23;POSITION 设定运动卡通图案的显示座标，取值范围 X:0~240,Y:0~220;SPRITE 语句定义静止卡通图案的显示座标，取值范围同 POSITION 语句。

(5)用 DEF SPRITE 语句定义的卡通图案只有 SPRITE、SPRITE ON、SPRITE OFF 对它起作用，其它语句不能控制。

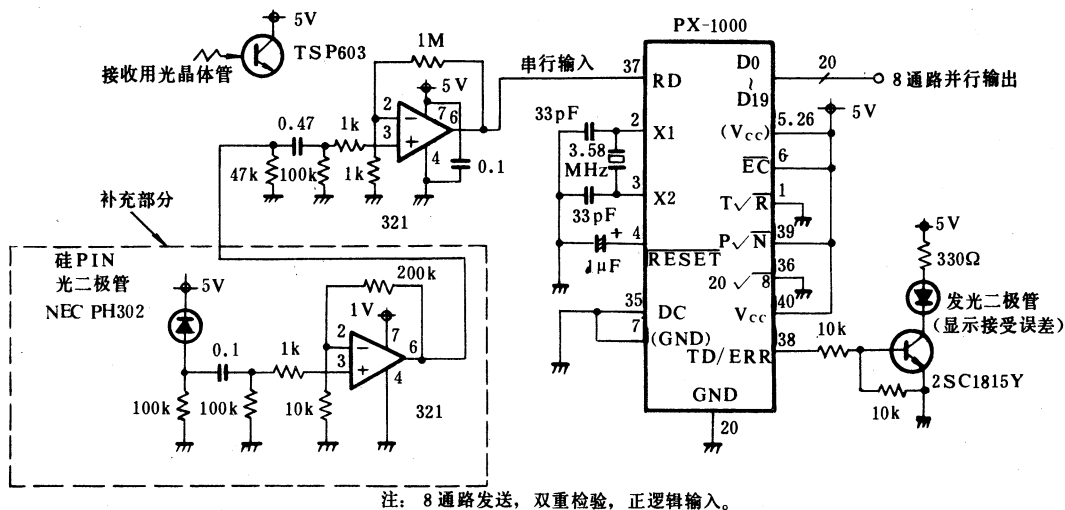


图 3 接收部分的电路图

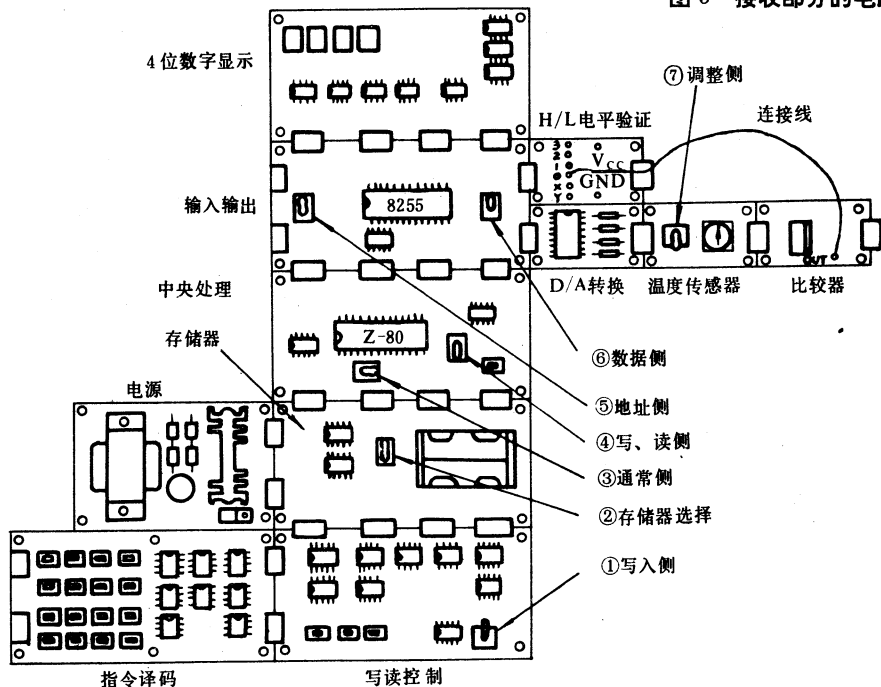


图 4 太平洋微电脑教育部件速接图

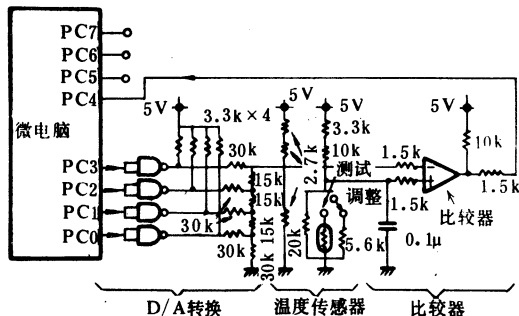


图 5 输入输出部件端口 C 的电路图

编制程序的要点：

输入输出部件端口 C 低位向 D/A 转换部件输出 0FH~00H，端口 C 第 4 位 PC₄ 检验比较器的输出。输出的数据反转之后加上 15H，即为室温。

寄存器 B 暂时存放 D/A 转换数据，寄存器 C 暂时存放现在室温。

4 位数显部件的低两位（端口 B）显示温度，数值为 15℃~30℃（15℃ 以下为 15℃，30℃ 以上为 30℃）。

程序流程图和程序清单如表 2、表 3 所示。

表 2

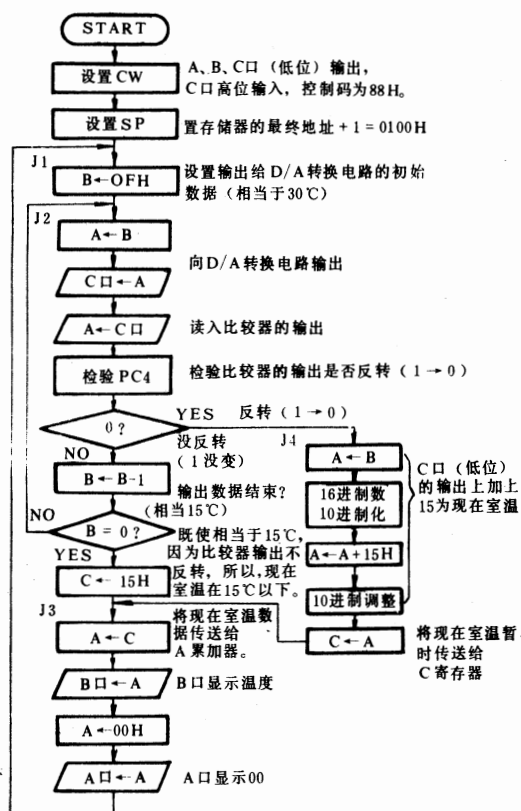


表 3

标号	助记符	地址	机器语言	注释
START	LD A, 88H	00	3E 88	设置 CW
	OUT(03H), A	02	D3 03	设置 SP
	LD SP, 0100H	04	31 00 00	
J1	LD B, 0FH	07	06 0F	设置 D/A 转换 初始数据
J2	LD A, B	09	78	
	OUT(02H), A	0A	D3 02	向 D/A 转换电路输出
	IN A, (02H)	0C	D8 02	
	BIT 4, A	0E	CB 67	检验比较器的反转输出
	JP Z, J4	10	CA 23 00	若反转跳入 J4
	DEC B	13	05	若不反转数据减 1
	JP NZ J2	14	C2 09 00	B ≠ 0 跳入 J2
	LD C, 15H	17	0E 15	B = 0 设置现在室温数据
J3	LD A, C	19	79	
	OUT(01H), A	1A	D3 01	向 B 口输入现在室温
	LD A, 00H	1C	3E 00	
	OUT(00H), A	1E	D3 00	A 口显示 00
	JP J1	20	C3 07 00	跳入 J1
J4	LD A, B	23	78	
	ADD A, 00H	24	C6 00	
	DAA	26	27	
	ADD A, 15H	27	C6 15	计算现在室温
	DAA	29	27	
	LDC, A	2A	4F	
	JP J3	2B	C3 19 00	跳入 J3

将程序写入微电脑，检验写入是否正确。执行程序。

温度传感器部件的切换开关开始应在“调整”侧，转动该部件的可调电阻，4 位数显部件显示 22 时，应该将切换开关搬至“测试”侧。

用手指触摸温度传感器部件的热敏电阻，温度显示发生变化，说明微电脑工作正常。

上述检验完毕，输入输出部件端口 B 连接发送部件，4 位数显部件的端口 B 连接接收部件，端口 A 连接电源部件。如图 6、图 7 所示。

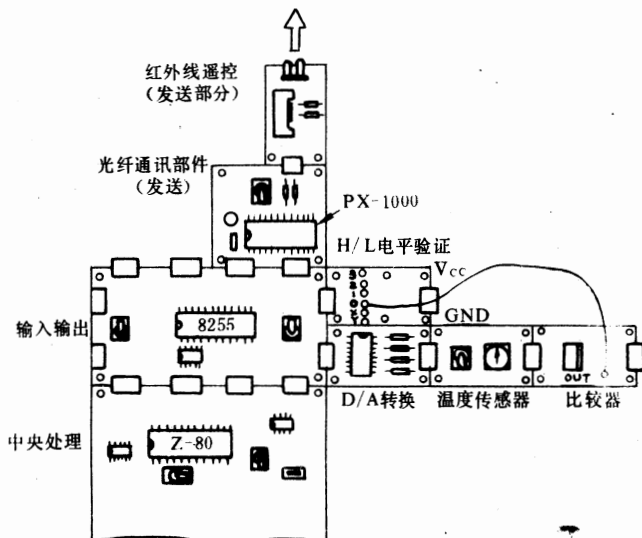


图 6 发送部分连接图

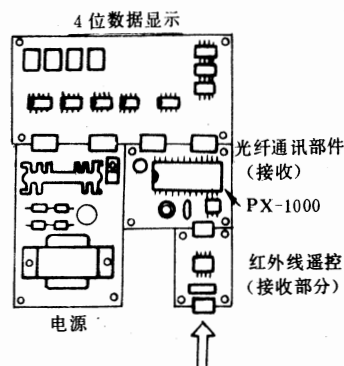


图 7 接收部分连接图

发送部分的红外发光二极管与接收部份的光敏晶体管相对应，两管位置正好对应时，接收部位检验位置的 LED 灯不亮。

再用手触摸热敏电阻，4 位数字显示发生变化，说明红外线遥测发送实验成功。

发送距离随周围条件而变化，日光灯照明条件下，能发送 4 米距离。

注：询问 D/A 变换部件、温度传感器、比较器部件等事宜请与天津纺织工学院机械系高殿斌同志联系，邮编 300160



Z80 单板机与打印机巧相连

北京信息工程学院(450002) 张慧成

Z80 单板机价格便宜,性能良好,适用于自动控制、机床改造、智能仪器、生产过程监测等工作。但是,由于内存和数码显示方式的限制,不可能配备比较完善的汇编反汇编和动态调试程序。对于用户来说,程序的汇编和调试仍是件很麻烦的工作,因此很多用户都利用现有的微机系统如 PC/XT 微机作为 Z80 的开发系统,利用交叉汇编把麻烦的汇编工作用微机来完成。但是在动态调试时,仍需在单板机上一步步调试。若能把打印机与单板机连接起来,利用点阵式行打印机把程序动态执行的过程、每一步的结果和有关状态直接打印出来,就会大大提高工作效率。

实际上,只用三个组件就能把单板机与打印机连接起来。可以把它们做在一块很小的电路接口板上附加在单板机的布线区上。在单板机的输入/输出(I/O)接口部分,其 I/O 译码电路的输出一般都留有空余未用的引脚以利扩展,可利用这些引脚作为选通信号。例如选 PS7(口地址 9CH—9FH,可只用 9CH)。电路原理如图 1。

由原理图 1 可以看出:单板机来的数据总线通过 74LS244 驱动与打印机数据线相连。其中 D₀ 线有双重作用:写操作时,D₀ 为数据信息,读操作时,D₀ 为打印机的状态。读打印机状态时的读信号由 IOR+PS₇ 组成, IOW+PS₇ 作为选通打印机的写信号。当单板机向打印机送打印信息时,数据总线上出现的是写向打印机的待打印信息。当向打印机接口板发读命令时,若读得结果的 D₀ 位为“1”,则打印机处于“忙”状态,否则处于空闲状态。据此可以编写出打印字符程序如图 2:

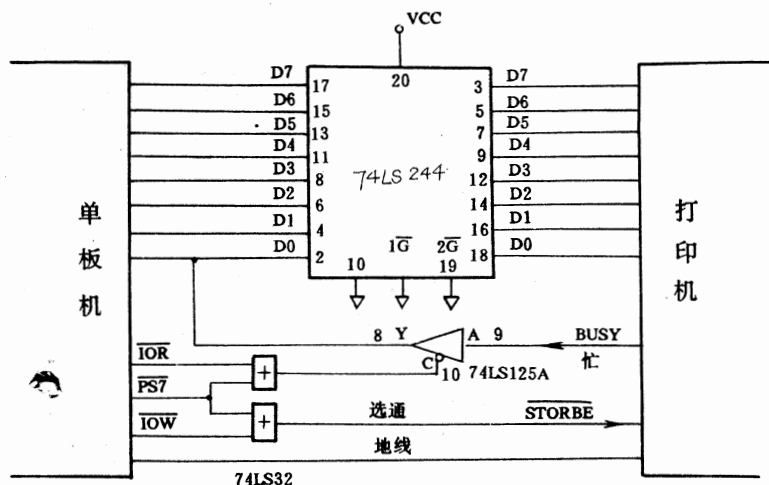


图 1. Z80 单板布线区上外接的扩展打印机接口板

入口: A——待打印字符的 ASCII 码
PCHR: PUSH AF
PR: IN A,(9CH) ;读打印机状态
AND 01H ;判断
JR NZ,PR ;若打印机忙,则转 PR
POP AF ;打印机电空
OUT (9CH),A ;送待打印的字符
RET

图 2 打印字符子程序

若要以十六进制数据形式打印出数据内容,相应子程序如图 3。以这两个子程序为基础可以编写各种应用程序,如打印信息程序和打印 Z80 CPU 寄存器状态子程序见图 4、图 5。

入口 A——待打印的数据

PHEX: PUSH AF ;保存数据
AND 0FH ;取高 4 位
RLCA
RLCA
RLCA ;移到字节的低 4 位
RLCA
ADD A,30H ;变为 ASCII 码
CP 3AH ;判是小于 9 的数字吗?
JR C,PH1 ;是,转 PH1
ADD A,07H ;数字为 10—15,
变为 A—F 的 ASCII 码
PH1: CALL PCHR ;打印输出
POP AF ;恢复数据
AND 0FH ;取低 4 位
ADD A,30H ;变为 ASCII 码
CP 3AH ;判是小于 9 的数字吗?
JR C,PH2 ;是,则转 PH2
ADD A,07H ;否,变为 A—F 的 ASCII 码
PH2: CALL PCHR ;打印输出
RET

入口: (HL)——信息首地址指针,数据信息
以控制字符 03H 表示结尾

PMSG: PUSH AF
PUSH HL ;保护现场
PMSG1: LD A,(HL) ;数据信息送 A
CALL PCHR ;打印输出
INC HL ;指向下一个
LD A,(HL) ;取信息到 A

```

CP 03H      ;是结束标志吗?
JR NZ,PMSG1 ;否,转回
POP HL
POP AF      }是,恢复现场
RET

```

图 4 打印信息子程序

```

PRTREG  PUSH BC
        PUSH DE }保护现场
        PUSH HL
        PUSH AF
        PUSH AF ;保存 AF 寄存器内容
        LD A,'B'
        CALL PCHR
        LD A,'C'
        CALL PCHR }打印字符"BC="
        LD A,'='
        CALL PCHR
        LD A,B
        CALL PHEX }打印 BC 寄存器的内容
        LD A,C
        CALL PHEX
        LD A,' '
        CALL PCHR
        LD A,'D'
        CALL PCHR }打印字符"DE="
        LD A,'E'
        CALL PCHR
        LD A,'='
        CALL PCHR
        LD A,D
        CALL PHEX }打印 DE 寄存器的内容
        LD A,E
        CALL PHEX
        LD A,' '
        CALL PCHR
        LD A,'H'
        CALL PCHR }打印字符"HL="
        LD A,'L'
        CALL PCHR
        LD A,'='
        CALL PCHR
        LD A,H
        CALL PHEX }打印 HL 寄存器的内容
        LD A,L
        CALL PHEX
        LD A,' '
        CALL PCHR
        LD A,'A'
        CALL PCHR }打印字符"AF="
        LD A,'F'
        CALL PCHR
        LD A,'='
        CALL PCHR
        POP BC ;将 AF 寄存器的内容弹至

```

```

BC 寄存器
LD A,B
CALL PHEX }打印出 AF 寄存器的内容
LD A,C
CALL PHEX
LD A,0DH
CALL PCHR }回车换行
LD A,0AH
CALL PCHR
LD A,03H
CALL PCHR }送信息结束标志
POP AF
POP HL }恢复现场
POP DE
POP BC
RET

```

图 5 打印 Z80—CPU 寄存器内容子程序

```

=====
(上接 39 页)
wbs:  cmp     ah,75
      jz      wbs1
      cmp     ah,77
      jnz     key
      mov     ah,R4
      dec     al
      dec     al
      mov     R4,al
wbs1:  mov     dx,03d4h
      mov     al,04h ; 04h, 04h, 02h, 02h (第
                        60 行)
      out     dx,al
      mov     dx,3d5h
      mov     al,R4
      inc     al
      out     dx,al
      mov     R4,al
      jmp     key
wbs    ends
      end     strt

```

该源文件经汇编后形成 AT40. EXE 或 AT40. COM 程序。
源文件第 5, 8, 9, 11, 43, 60 行注释中的数据依次为
AT40. ASM, AT80. ASM, AG40. ASM, AG80. ASM 中相应指令
中的值。例如:

```

第 5 行  AT40. ASM 中为 piont db 40 dup(' ')
          AT80. ASM 中为 piont db 80 dup(' ')
          AG40. ASM 中为 piont db 40 dup(' ')
          AG80. ASM 中为 piont db 80 dup(' ')
第 60 行 AT40. ASM 中为 mov al,04h
          AT80. ASM 中为 mov al,04h
          AG40. ASM 中为 mov al,02h
          AG80. ASM 中为 mov al,02h

```

除了上述 6 行有不同之外,这四个源文件的其他部分是
完全相同的。

AT40. ASM, AT80. ASM, AG40. ASM, AG80. ASM 四个源
文件经汇编后将分别形成
AT40. EXE(或 AT40. COM),
AT80. EXE(或 AT80. COM),
AG40. EXE(或 AG40. COM),
AG80. EXE(或 AG80. COM)。

PAL 卡使用中的几个问题

本刊在 92 年第三期向读者推荐了使 PC 微机接彩色电视机的 PAL 卡。

该卡在使用中可能出现一些问题,解决办法如下:

1. 场不同步

画面上下滚动或为两三个静止画面。

此时,应调整电视机的垂直同步旋钮或电位器(V—HOLD)。

2. 行不同步

荧光屏上是许多向左或向右倾斜的条子。

此时,应调整电视机的水平同步旋钮或电位器(H—HOLD)。

3. 画面左右偏移

可调整 PAL 卡上的电位器 W2。

4. 无彩色

微调 PAL 卡上的可变电容器 C1 和电位器 W1。

5. 色调失真

画面颜色与实际颜色不符。

微调 PAL 卡上的电位器 W1。

6. 场不同步且电视机无垂直同步旋钮或电位器可调。此时可运行下列程序:

(1)AT40.COM

用于 40X25 字符方式。

(2)AT80.COM

用于 80X25 字符方式。

(3)AG40.COM

用于 320X200 图形方式。

(3)AG80.COM

用于 640X200 图形方式。

上述四个程序运行后,荧光屏上将显示由“.”组成的网

格。按 ↑ ↓ ← → 四个键可使行场同步且使画面上下左右偏移。

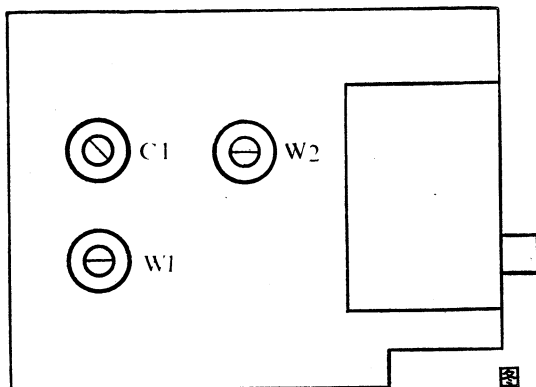
按 ESC 键退出该程序。

AT40. ASM 源文件如下:

```
wbs segment
    assume cs:wbs,ds:wbs
    org 0100h
strtr: jmp zz
piont db 40 dup(' ') ;40,80,40,80(第5行)
db '$'
asd DB "Please press ESC exit $"
R4 DB 28H ;28H,28H,3CH,3CH
; (第8行)
R7 DB 20H ;20H,20H,30H,30H
; (第9行)
zz: mov ah,0
```

```
mov al,00 ;00,02,05,06(第11行)
int 10h
mov dx,60h
mov ah,2
int 10h
mov cx,25
disp: mov dx,offset piont
mov ah,9
int 21h
loop disp
mov dh,12
mov dl,0
mov ah,2
int 10h
mov dx,offset asd
mov ah,9h
int 21h
key: mov ah,0
int 16h
cmp al,1bh
jnz wbsv
mov ah,4ch
int 21h
wbsv: cmp ah,72
jz wbsv1
cmp ah,80
jnz wbsv
mov al,R7
dec al
dec al
mov R7,al
wbsv1: mov dx,03d4h
mov al,07h
out dx,al
mov dx,3d5h
mov al,R7
inc al
out dx,al
mov R7,al
jmp key
```

(下转 38 页)



图

降低 MP-1 电脑的功耗

广西苍梧县佛子冲铅锌矿机动科(543100) 邓文超

作为计算机的后备直流电源,通常使用的都是干电池。干电池容量小、价格高。因此,降低计算机的功耗就显得十分重要。

为了方便编程,MP-1 计算机设置有 18 个发光二极管指示数据和地址,计算机运行时 18 个发光二极管绝大部分都在工作(发光)。事实上,在计算机投入正常工作后,这些发光二极管完全没有必要再发光,而白白消耗能源,因为我们实际需要的仅仅是计算机的 I/O 口信息。

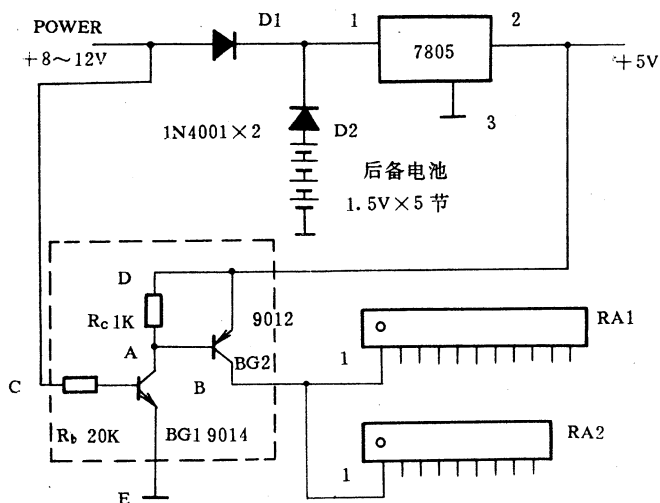


图 1

笔者经过尝试,增加四个元件(见图 1 虚线框内),将这些发光二极管工作方式改为:当使用市电时就发光;使用后备电源时就停止工作(发光)。这样对延长电池的使用寿命起到了很好的作用。因为 18 个发光二极管消耗的总电流接近 90 毫安,而干电池的容量仅为几百个毫安时。

工作原理:(对照图 1)当有市电时,市电经配套的电源变换器形成约 10 伏的直流电供到 POWER 插座,经 Rb 提供基极电流给 BG1, BG1 导通, A 点电压近似为零, BG2 导通, B 点电压约等于 5 伏,也就是说 RA1、RA2 电阻排的第 1 脚电压约等于 5 伏,于是发光二极管工作如常;当市电断开后,后备电池经 D2 投入工

书讯

潍坊计算机公司北京办事处尚有少量《中华学习机》1990 年合订本(上、下册),定价 12.00 元,《苹果园》1986、1987、1988 年合订本,每年定价 6.00 元。以上定价均含邮资在内。

欲购者速从邮局寄款至:北京市劳动人民文化宫内,潍坊计算机公司北京办事处收。邮政编码:100006。

作,由于 D1 的隔离, Rb 没有电流流过, BG1 截止, A 点电压呈高电位, BG2 截止, B 点电压约为 0, 因此各发光二极管得不到工作电流而停止工作。

改装步骤:

1. 将电阻排 RA1、RA2 焊下(若没有专用拆焊集成电路的工具可借助于 12 号注射针头, 逐个脚拆焊), 并将它们的第 1 脚折弯起来。
2. 将 RA1、RA2 的各脚除第 1 脚之外焊回原处。
3. 用细导线将 RA1、RA2 的第 1 脚焊接连通。
4. 用刀片将 POWER 与 7805 稳压集成块的第 1 脚之间连接的印刷线路割断(插元件面和焊点面均须割断), 并在断口处跨焊一只 4001 二极管(即图 1 中的 D1)。

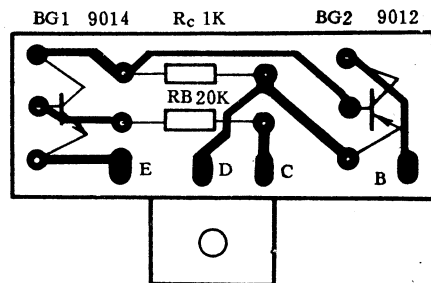


图 2

5. 对照图 1 虚线部分设计一小块电路板如图 2, 并焊上 Rb、Rc、BG1、BG2 四个元件; 在 B 点引线连接到 RA1、RA2 电阻排的第 1 脚; C 点接到主机电路板上的 POWER 焊点(即图 1 中 D1 正极); D 点接到 7805 的 2 脚(即 5 伏), E 点接到 7805 的 3 脚(即地)。然后利用固定 7805 的螺钉将此小电路板固定(也可不用小电路板, 而直接将四个元件粘在主机板上)。

调试:

由于元件少, 线路简单, 只要确保元件完好, 安装无误, 一般无需调试就可成功。成功的标志是: 使用市电时有发光二极管亮; 而断开市电使用后备电池时发光二极管全灭。如果不是这样则应检查电路是否安装错误, 例如三极管极性接反或 BG1、BG2 型号对调等。

敬告作者

因本刊编辑部人力有限, 来稿量大, 难以做到不用稿件一一退回, 请作者务必自留底稿。在给我刊搞稿半年之后未见刊登者, 可再向他刊投稿。



电脑游戏机

第三章 F BASIC 的画面控制语句

山东 于 春

六. 配色语句(COLOR, CGSET, PALET)

F BASIC 系统中固化了卡通图案3板12组颜色配合, 背景图案2板8组颜色组合供配色使用, 还可以使用 PALET 语句从52种颜色代码中任选三种组成新的一组色彩组合。有关配色语句只有三条, 虽然语句不多, 但比较生涩难懂, 随机手册介绍的又过于简单, 从而使初学者很难掌握弄通。希读者根据本文的介绍和示例加强练习, 在应用中, 逐步深入理解。

1. 置色语句(COLOR)

COLOR 简写 COL.

COLOR 语句的功能是在屏幕的某位置显示某种颜色。语句格式为

COL. X, Y, n

式中 X: 屏幕水平方向的位置座标0~27.

Y: 屏幕垂直方向的位置座标0~23

n: 配色号码0~3

COLOR 语句可使每组 X, Y 决定的点的相邻四个位置染上由 n 决定的色彩。因此, 利用 COLOR 语句, 可使背景面产生千变万化的色彩。

例23.

```
10 F. I=0 TO 671
20 P. CH. (255);
30 N.
40 X=RND(28):Y=RND(24):E=RND(4)
50 COL. X, Y, Z
60 G. 40
```

RUN 屏幕先连续打印一兰色底面, 然后产生各种色彩块的千变万化。

2. 规定 BG、SP 面颜色组合语句(CGSET)

CGSET 简写 CG.

CGSET 语句的功能是规定背景及卡通画面的颜色组合。语句格式为:

CG. m, n

式中 m—背景用的色块代码0~1

n—卡通用的色块代码0~2

有了 COLOR, CGSET 语句就可以对背景和卡通进行配色、具体方法如下:

A. 卡通图案的配色

(1) 通过 CGSET 语句 n 选定 SPRITE 的面板代码0~2(见随机手册封底)。

(2) 通过 DEF SPRITE 语句中的 A 选定由 CGSET 语句已选定面板中的颜色组合。即配色号码0~3。

B. 背景图案的配色

(1) 通过 CGSET 语句中的 m 选定 BG 的面板代码0~1。

(2) 通过 COLOR 语句中的 n 选定由 CGSET 语句已选定面板中的配色号码0~3。

在具体运用中, 要首先弄清面板代码和配色号码这两个概念。然后再弄懂每个配色号码所代表的一组颜色组合中的三种色彩的代码。

例24.

```
10 CLS:SP. O.
20 F. A=0 TO 1:F. B=0 TO 2
30 CG. A, B
40 F. I=0 TO 671:P. CH. (255);:N.
50 F. C=0 TO 3
60 COL. 5+C*3, 5+C*2, C:N.
70 F. D=0 TO 3
80 DE. SP. D, (D, 1, 0, 0, 0)="@ABC"
90 SP. D, 50+D*50, 60+D*50
100 N.:N.:N.
```

RUN 可以看到背景、色块、飞鸟的各种颜色变化。其变化顺序为

- (1) 背景配色面板代码为0, 卡通面板代码为0, 打印三种色块, SP 变化0面板的四种颜色组合。
- (2) BG 面板代码为0, 卡通面板代码为1, SP 变化1面板的四种颜色组合。
- (3) BG 面板代码为0, SP 面板代码为2, ……
- (4) BG 面板代码为0, SP 面板代码为3, ……
- (5) BG 面板代码为1, SP 面板代码为0……

⋮

3. 重新定义配色号码语句(PALET)

PALET 语句的功能是将配色代码中的颜色组合重新设定为任意颜色组合, 以满足程序中配色的需要。该语句使用的不多, 随机手册中介绍的较细, 不再赘述, 仅举一例。

例25. 用 PALET 语句重新设定卡通图案中0号面板中第三组颜色组合。选定 SP 为玛丽和丽莎。

```
10 CLS:CG. 1, 0:SP. O.
20 DE. SP. 0, (3, 1, 0, 0, 0)=CH. (40)+CH.
(41)+CH. (42)+CH. (43)
30 DE. SP. 1, (3, 1, 0, 0, 0)=CH. (12)+CH.
(13)+CH. (14)+CH. (15)
40 SP. 0, 100, 150:SP. 1, 70, 100
50 PAU. 300
60 PAL. S3, 0, 18, 22, 41:PAU. 300
```

70 PAL.S3,0,1,21,33:PAU.300

80 G.10

RUN 显示玛丽,丽莎两个卡通图案三种色彩的反复变化。

到此为止,本章介绍了有关 BG、SP 控制语句28条。这些语句大部分比较陌生,尤其对初学者,接受难度较大。但只要按本文介绍的语句顺序,在领会掌握示例设计思路的基础上勤奋练习,定能熟练掌握,灵活运用。

另外,在 F BASIC 系统中固化了四个不完全的游戏程序,读者可以通过 F₁~F₄键调出,也可以键入 GAME 指令调出。用 GAME 指令调用时,其格式为 GAME n(GAME 简写 GA. 本机 n 为 0~3)。调出程序,按 STOP 中断,用 LIST 列表,可进行补充和阅读。通过阅读别人的成熟程序,开拓思路,然后融汇贯通,举一反三,编出自己的游戏程序。

综合已介绍的语句,下面列一个游戏程序片断。由于编制一个完整的游戏程序,程序量是很大的,不可能列出完整的程序清单刊载。为使读者窥一斑而知全豹,示例模仿采蘑菇游戏,用 1# 操纵器控制玛丽左、右运动和向左、右跳。同时,有三个乌龟自右向左运动,碰到玛丽则消失,在屏幕右上角打印得分。若玛丽跳到屏幕外边,则自动重新启动开始。

为便于阅读和理解,示例有关部分加进中文注释。为减少篇幅,删去了音乐程序。

例26. 游戏程序片断

```
1 K=STRI.(0):IF K=0 T.1 } (按 START 钮,
2 IF K=1 T.LOC.9,10: } 打印 START!!
P."START!!":PAU.300 } 延时300)
10 CLS:CGEN(2):SP.O.: (初始状态设定。)
CGSET 0,1:Q=0:
X2=56:Y2=150
20 BG. (转 BG 画面)
30 F.I=4 TO 6 (定义三个乌龟
自左向右移动。)
40 DE.M.(I)=SP.(13,7,3,255,0,I-3)
50 POS.I,145+I*15,150
60 M.I:N.
70 K=STRI.(0):G=SRI.(0) (定义1#操纵器
方向功能)
IF K=0 AND G=0 T.70
80 IF K=8 T.380 (按 A 键,转跳子程序)
90 IF G=1 T.S=3 (左,右运动)
100 IF G=2 T.S=7
110 X1=X2:Y1=Y2 (左,右跳判断变量)
120 DE.M.(0)=SP. (0,S,1,2,0,0)
130 POS.0,X2,Y2 (定义0#卡通)
140 M.
150 IF M.(0)=-1 T.150
```

160 GOS.440 (转 SP 相遇处理子程序)

170 X2=XP.(0):Y2=YP.(0):G.70

(测试动作结束的座标。)

```
180 I=0
190 I=I+1:X2=X2+I/2: } (设定右跳座标)
Y2=Y2-I*2:
IF I>7 T.230
200 GOS.410 (转出界处理子程序)
210 DE.SP.1,(2,1,0,1,0)=
GH.(13)+CH.(12)+CH.(15)
+CH.(14)(定义1#动作面向右)
220 SP.1,X2,Y2:PAU.3: (令1#显示)
SP.1:G.190 (令1#消失,反复.)
```

(180~220定义玛丽向右跳上去。)

```
230 PAU.6:I=0
240 I=I+1:X2=X2+I/2:Y2=Y2+I*2:
IF I>7 T.270
250 GOS.410
260 SP.1,X2,Y2:PAU.3:SP.1:G.240
270 PAU.6:SP.1: (令1#SP消失,设定
0#显示座标,启动)
POS.0,X2,Y2:M.0:G.70
```

(230~270为玛丽向右跳下。

180~270为玛丽向右跳一次。)

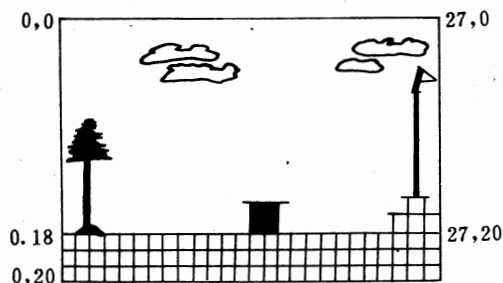
```
280 I=0
290 I=I+1:X2=X2-I/2: } (定义左跳座标。)
Y2=Y2-I*2:
IF I>7 T.330
300 G.410
310 DE.SP.2,(1,1,0,0,0)=
CH.(12)+CH.(13)+CH.(14)
+CH.(15) (定义2#SP。)
320 SP.2,X2,Y2:PAU.3:(显2#SP、消2#SP)
SP.2:G.290
330 PAU.6:I=0 (定义2#跳下)
340 I=I+1:X2=X2-I/2:
Y2=Y2+I*2:
IFI>7 T.370
350 GOS.410
360 SP.2,X2,Y2:PAU.3:
SP.2:G.340
370 PAU.6:SP.2:
POS.0,X2,Y2:M.0:G.70
(280~370为SP向左跳一次子程序)
380 ERA 0 (令0#SP消失。
390 IF X1<X2 T.180 (转右跳子程
400 IF X1>X2 T.280 (转左跳子程
(380~480为左,右跳判断子程序)
```

```

410 IF X2<16 OR X2>240 (判断 SP 是否出界
    及出界处理子程序。)
    T. X2=100:G. 10
420 IF Y2<24 OR Y2>220
    T. Y2=150:G. 10
430 RE.
440 IF CRA. (0)=4 T. CAN4:(判断0#SP 是否与
    Q=Q+1 4#、5#、6#相遇及相遇
    处理子程序。)
450 IF CRA. (0)=5 T. CAN5:Q=Q+1
460 IF CRA. (0)=6 T. CAN6:Q=Q+1
470 LOC. 12,0:P. "SCORE:"; } (打印得分)
480 LOC. 20,0:P. Q
490 IF M. (4)=0 AND M. (5)=0 (若 SP4#、5#、
    6#运行完毕重新启动)
    AND M. (6)=0 T. 30
500 RE.

```

程序输入完毕, RUN 后, 按1#操纵器的开始键 START, 进入 BG 画面, 按附图各点座标设计背景后, 返回 BASIC 状态, 键入 LIST 20, 消去 BGTOOL, 换为 VIEW. 即可运行。



背景图案设计图:

读者可以体会到一个完整游戏程序的复杂性。调试设计程序往往一次不能完成, 而游戏机又不能总是通电, 因此, 读者可参阅随机手册中有关用录音机读写资料的指令, 将已设计的程序存入磁带, 在下次设计时再调出来继续进行。但大多数收录机写入容易, 读出困难。这并非你的收录机或电脑有问题, 而是游戏机的磁带输入接口与收录机的输出不匹配。有关磁带读写的硬件改造以后再介绍。

最后, 列举一个完整的、简单的游戏程序示例。

例27. 打飞鸟

游戏开始, 一只飞鸟在天空自右向左飞行, 下面有一个发射台, 按动1#操纵器上的 A 键可以发射子弹, 击中飞鸟, 累计击中的数量每10发子弹为一局, 局末显示命中率。同例26相同, 程序后加进中文注释。

```

10 CLS:SP. 0.:CGS. 1,0
20 LOC. 0,20:P. "Fa She"
30 LOC. 20,20:P. "J:Zhong"
40 PLAY"V15T1" (定义发声)

```

```

50 DE. M. (0)=SP. (12,1,1,1,0,0)(定义子弹)
60 DE. SP. 2,(0,1,0,0,0)=
    CH. (172)+CH. (173)+
    CH. (174)+CH. (175): } (定义发射台
    SP. 2,120,225 } 并显示)
70 CGS. 1,2:DE. M. (1)=
    SP. (15,7,1,255,0,3): } (定义飞鸟
    POS. 1,250,20:M. 1 } 并使其运动。)
80 K=STRI. (0):IF K=0 OR (定义1#操纵器,
    只有按 A 键起作用。)
    K=1OR K=2OR K=4 T. 170
90 IF K=8 T. T=T+1
100 IF T=11 T. 230 (每局10发弹)
110 LOC. 1,22:P. T (打印发射数)
120 PLAY"O4C0"CD"DEE"FG"G"(发射子弹声)
130 F. I=220 TO 0 ST. -5
140 POS. 0,125,I:M. 0 } (子弹运行并判断
    PAU. 1:IF CRA. (1)=0 T. 190 } 是否击中飞鸟。)
150 N.
160 ERA 0 (子弹消失)
170 IF M. (1)=0 T. M (若飞鸟停止再启动。)
180 G. 80
190 ERA0,1:PLAY"O0C0EGCEG"(击中飞鸟声)
200 CGS. 1,1:
    DE. SP. 3,(3,1,0,0,0)= (定义爆炸)
    CH. (180)+CH. (181)+
    CH. (182)+CH. (183)
210 SP. 3,125,20:PAU. 30: (显示爆炸并消失)
    SP. 3:
    S=S+1:LOC. 22,22:P. S (打印击中数)
220 G. 70
    (190~210为击中飞鸟处理子程序)
230 ERA 0,1:C=S*10: (结束处理子程序
    LOC. 5,10:
    P. "Ni De Ming Zhong Lu="; 打印命中率)
    C;"%":
    LOC. 0,23
240 END

```

程序中使用了发声语句 PLAY, 以前没有介绍, 读者可参阅随机手册。

在游戏画面上, 中间留了很大空间, 读者可使用 BG 做图法, 自己设计蓝天、白云, 陆地、江河、山岭、树林等。以使游戏画面生动逼真, 丰富多彩。

最后说明几点:

(1) DEF SPRITE 和 DEF MOVE 两个语句中的卡通图案的编号不能混淆。这两个语句, 各自独立使用自己的编号。因此, 在一个画面上既可以用 DEF SPRITE 语句定义显示8个卡通图案, 又可以用 DEF MOVE 语句定义8个卡通图案在画面

(下转33页)



中华学习机电源故障维修经验谈

浙江嘉兴市化工厂企管办(314021) 汪晖

CEC—I 中华学习机是我国自行研制的一种灵巧的八位微型计算机。由于其功能强,造价低而倍受微机爱好者和广大中小学生的青睐。然而随着时间的推移,其故障的维修问题就愈显得重要了。

中华学习机中硬件部分的电源系统是故障的多发部位。笔者根据自己几年来的维修经验,在整理了有关方面资料的基础上实测并绘制了上海无线电四厂生产的凯歌牌 CEC—I 中华学习机电路原理图(详见附图),想就具体的故障现象谈谈维修的经验以供读者参考。

一、电路工作原理

在谈具体实例前先对 CEC—I 电源的原理和其特征性作一番剖析,以益在举例中起到触类旁通的效果。此机为西南电子(蛇口)有限公司的产品,属无工频变压器自激式单端反激脉宽频率混合调制型开关电源。有 $\pm 5V \pm 12V$ 四路输出,总输出功率约 25W。

1. 主电路

从图中可见 R_1 为限流电阻, C_1 、 L_1 、 C_2 、 C_3 为低通滤波器,主要起滤除电网交流高频干扰和消除电源振荡电路产生的高频干扰信号作用。主变换电路由 R_6 、 R_7 、 Q_1 、 Q_2 、 R_3 、 D_7 、 C_8 、 T_1 的绕组 N_1 和 N_3 组成。它们构成单管自激多谐振荡器。当电源接通后,在 R_7 、 R_6 的偏压下 Q_1 开始导通,集电极电流 I_c 流过 T_1 的 N_1 绕组,在绕组 N_3 上感应出③正⑤负的电压,此电压经 R_3 、 D_7 、 R_4 加到 Q_1 基极。并同时 C_8 进行充电。此正反馈电压使 Q_1 进一步导通达到饱和。 Q_1 饱和后, I_c 达到最大值,并不再继续增大, T_1 各绕组上的感应电压消失。此时 C_8 已充至 N_3 绕组的感应电压值,极性为上负下正的电压加到 Q_1 使 I_c 减小,并在 T_1 的 N_3 绕组上感应出③负⑤正的电压。 N_3 绕组的电压与 C_8 上的电压相迭加,给 Q_1 的基极加上了一负偏压,可改善波形的上升沿。同时 Q_1 在 N_3 绕组的正反馈作用下,又很快进入了截止状态。此时 I_c 最小, N_1 、 N_3 上的感应电压为零, Q_1 的基极又在 R_6 、 R_7 的偏压下开始了新一轮循环。如此往复就形成了自激振荡,其振荡的频率取决于 Q_1 的 I_c 值和 N_1 绕组的电感量。 R_4 是用于限制 N_3 的感应电压向 Q_1 提供的基流,使 Q_1 不致饱和过深。 R_3 、 D_7 起改善振荡波形上升沿和下降沿的作用。 C_8 既有减小 Q_1 的存储时间作用,又有在短路保护后使 Q_1 重新起振的效应。 T_1 是开关电源的核心部件,兼有变压器和储能的双重作用。它的质量和参数将直接影响电源的性能和 Q_1 的工作状态。经 T_1 变换得到的回路高频电压经 $D_{11} \sim D_{15}$ 整流, $C_{12} \sim C_{19}$ 、 L_2 、 L_3 滤波后输

出平滑的直流电压。 $R_9 \sim R_{11}$ 是开关电源所必须的负载。

2. 控制电路

图中虚线框中的部分为控制电路。 T_2 左边部份为调频、调宽电路,右边部份为取样比较放大电路。 $+5V$ 和 $+12V$ 通过 R_{12} 、 R_{13} 、 R_{14} 、 R_T 分压后得到一直流电压,并与 D_{17} 稳压管的基准电压 V_z 进行比较,得一误差信号,经 Q_4 、 Q_5 放大后,得到经 T_2 的 W_2 绕组的控制电流,通过互感控制 W_1 绕组,交变阻抗达到控制 T_1 的 N_3 绕组加在 Q_3 基极上的电压高低的目的。也就是说通过控制 Q_3 与 Q_1 基极电流的分流作用,来改变 Q_1 振荡的频率和导通时间,最终改变输出电压的高低起到稳压的作用。

3. 保护电路

此种电源设有多种保护功能,如对 Q_1 管的保护、过载保护,短路保护, $+5V$ 过压保护等。这里只略讲二种:对 Q_1 管的保护是 R_2 、 D_5 、 C_6 、 C_7 组成的能量吸收电路来完成,它们主要吸收 T_1 的 N_1 绕组在 Q_1 截止期间的剩余能量,以减小脉冲尖峰起保护 Q_1 管作用。过载保护由 Q_2 、 Q_3 、 R_5 等组成,当输出功率达到额定输出功率的 1.2 倍时, Q_1 的 I_c 增大, R_5 上的压降也增大,当压降大于 0.3V 时, Q_2 导通,使 Q_3 导通, Q_1 基极电流减小,此正反馈作用使 Q_1 截止,实现了保护。

二、电路的优缺点

优点:无电源变压器体积小,效率高,稳压范围宽,有多种保护功能,振荡、调整部分与输出取样部分隔离,干扰少,电源采用了冷盘设计、整流地线悬浮与整机地线分开,使主机不带电,使用安全。

缺点:电路较复杂,相互牵扯多,维修相对较困难,对各元件要求较高。

三、举例谈维修

下面结合两例维修实例,谈修理技法。

例 1. 开机屏幕上无任何显示,键盘右上方指示灯不亮。

分析与检修:这种现象是典型的电源部分故障,首先要判断一下故障在输入部分还是在输出部分。一般检查保险丝 F_1 有否烧断输出回路电压是否正常。保险丝烧断,回路输出全无为输入部分故障,反之为输出部分故障。打开机盖查如保险丝烧断,换上新的开机又烧断,检查 Q_1 和 Q_2 两管均击穿, R_1 也烧断,换上新管 R_1 后电源正常。对这种问题应注意 Q_1 、 Q_2 击穿后应检查一下 D_5 、 D_7 、 D_8 、 D_9 、 D_{10} 等有否损坏,否则新换上的 Q_1 、 Q_2 管又有被击穿损坏的可能。 Q_1 管原型号不

好买时,可用 BU406、2SC2832A、2SC3150 等开关管代换, Q_2 可用 9012 代换。

例 2. 故障现象同上

分析与检修:先检查保险丝完好,再测量回路输出电压,只有 +5V 没有电压输出,其余三路电压均正常。这是输出部分的故障,检查稳压管 D_{18} 好的,用万用表欧姆档测量 D_{13} 、 D_{14} ,发现 D_{13} 已击穿短路,电源保护电路作用使 Q_1 停振而无电压输出。换上新的二极管后故障排除。

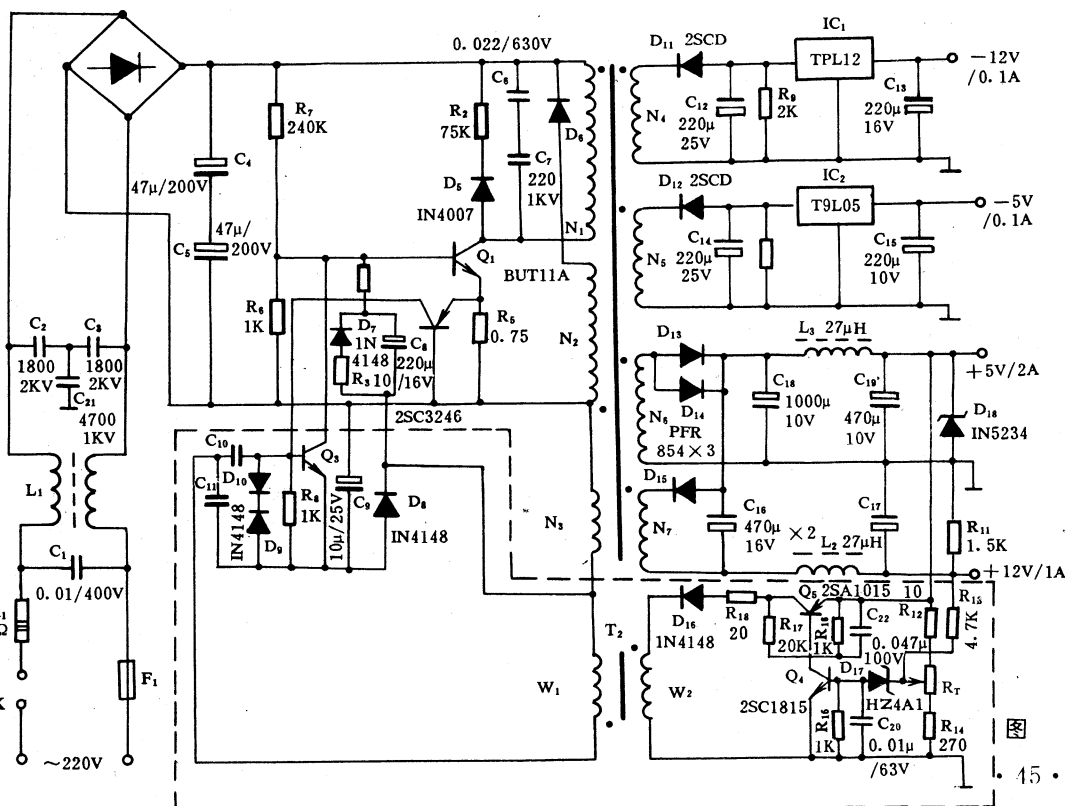
四、小结:

从以上两例维修实例中可见中华学习机电源部分的故障并非十分复杂,不外乎有三种可能性,见表三。只要了解和掌握了开关电源的原理、电源变化与故障之间的关系后,维修就便利。表一和表二是笔者

实测了上无四厂凯歌牌 CEC—I 中华学习机电源部分各三极管的电压值和各管在断电时的正确电阻数据,供读者参考,其中静态值是指电源空载时的实测数据。另外再将一组中华学习机电源断电路,各组输出电压组件的内阻数据供读者在维修时参考。MF64 型表 $R \times 100$ 档,红笔接地时 +5V 端内阻 $400\Omega \sim 500\Omega$,反向时为 $350\Omega \sim 400\Omega$; -5V 端红笔接地时内阻 $1.5K \sim 1.7K$,反向时 $1.5K \sim 2K$; +12V 端红笔接地时内阻 $1K \sim 1.1K$,反向时 $900\Omega \sim 1K$; -12V 端红笔接地时内阻 $1.2K \sim 1.4K$,反向时 $3K \sim 3.4K$ 。

电源部分的故障有时是很简单明了的,有时却复杂隐蔽,上述为一般常用的方式及经验,读者不可墨守陈规,用固定模式去套用。

编 号	型 号	电 压 值					
		V_c (伏)		V_b (伏)		V_e (伏)	
		静态	动态	静态	动态	静态	动态
Q_1	BUT11A	270	260	-3.3	-3.1	0	0
Q_2	2SA1015	-7.4	-8.7	0	0	0	0
Q_3	2SC3246	-3.3	-3.1	-7.4	-8.7	-4.6	-5.8
Q_4	2SC1815	4.4	4.4	0.7	0.6	0	0
Q_5	2SA1015	1.4	2.6	4.4	4.4	5.1	5.1
备 注		以上数据为 MF64 型万用表实测					



图

表二

MF64 型 R×100

管 脚	Q ₁		Q ₂		Q ₃		Q ₄		Q ₅	
	红地黑测	黑地红测	红地	黑地	红地	黑地	红地	黑地	红地	黑地
R _c	20K	1K	1.7K	∞	1K	∞	1.5K	1.5K	1.1K	1.5K
R _e	0	0	0	0	1.5K	∞	200Ω	200Ω	0	0
R _b	700Ω	700Ω	0	0	1.7K	900Ω	1.1K	1.15K	950Ω	850Ω

表三

常见电源部分故障	维修时方法和检测部位
无输出电压	1. 查 F ₁ 是否完好? 完好测有无输出电压? 查 D ₁₃ 、D ₁₄ 、D ₁₅ 、D ₁₈ 、D ₇ 、D ₈ 、Q ₃ 、Q ₁ 等 2. F ₁ 烧断查测 D ₆ 、Q ₁ 、Q ₂ 、Q ₃ 、D ₅ 、D ₇ 、D ₈ 、D ₉ 、D ₁₀ 元件有无损坏。
输出电压偏高	1. 是 -5V 或 -12V; 查 79L05、79L12 2. 是 +5V 或 +12V; 调 RT, 查 D ₁₇ 、R ₁₁ 、R ₁₂ 、R ₁₃ 、R ₁₄ 、D ₁₈ 、Q ₄ 、Q ₅ 。
输出电压偏低	1. 仅 -5V 或 -12V, 查 C ₁₂ 、C ₁₄ 、79L05、79L12 2. 调 R _T , 将 D ₁₆ 、D ₁₈ 同时断开电压升高重点查 T ₂ 右边 Q ₅ 、Q ₄ 、D ₁₆ 、D ₁₇ 若无变化重点查 Q ₃ 、Q ₂ 、D ₈ 、C ₉ 等元件

袖珍计算机 PC 型 PB 型系统 RAM 扩展 模块常见故障与维修

湖北沙市饲料公司微机室(271000) 朱 杰

1. 不执行读写指令,通过 MEM 命令查询 RAM 的剩余容量,只显示主机本身的 RAM 容量。向主机输入程序时,显示 ERROR 10 错误,而程序并不长。

常见原因:

(1)模块内的 RAM 存储器由多块集成电路组成,由三线八选一译码分配器控制片选。当片选电路 TC40H138 损坏,模块内的 RAM 存储量无效,可用国产的集成电路 CC54138 代换。

(2)模块内部隔离缓冲电路 50H000P 损坏或模块电路板铜箔被腐蚀。首先,清理被腐蚀的铜箔线,镀上一层焊锡。如果 50H000P 损坏,可用国产器件 CC4050 代替更换。

(3)V_{GG} 电源隔离二极管 1S598 开路,片选电路没有加上工作电源。对此可启开模块,重新焊好即可。

(4)RAM 存储器电路损坏。可选用片状扁平封装的 HM6116 替换。

2. 显示内容与预存内容不符或丢失,主机显示紊乱,出现无规则显示点。

常见原因:

(1)模块内装锂电池因使用年久失效,漏出电解液,使集成电路,铜箔线短路,可用四氯化碳清洗模块基板及电路引脚,再更换相同规格的锂电池。

(2)模块插头与主机适配插口接触不良。插头有四十根引出脚,容易氧化,产生接触不良故障,用四氯化碳擦洗后,重新插入主机。

(3)模块内电源滤波电容漏电,使 RAM 供电不稳定。

(4)DAN202 片状双二极管损坏,当模块从主机拔下时,锂电池电源没有加到整个模块电路。使储存的信息因断电丢失。DAN202 二极管的作用是节省锂电池电源,当模块插入主机模块仓后,主机 V_{GG} 电源高于锂电池电源,没有加到整个模块电路。使储存的信息因断电丢失。DAN202 二极管的作用是节省锂电池电源,当模块插入主机模块仓后,主机 V_{GG} 电源高于锂电池电源,使 DAN202 反偏置关断,由 V_{GG} 保证模块工作。片状双二极管难以找到同规格器件,可用国产芝麻状二极管替代。

3. 执行 NEW0 操作后,通过 MEM 指令查询剩余容量,主机和 8K 模块的 RAM 容量只有 7994B 字节,比正常时的容量(10042B 字节)整整少了 2KB 字节,但主机运算、编程正常,这是 CE-155 模块内有一片 RAM 电路的存储容量没有加上的表现。

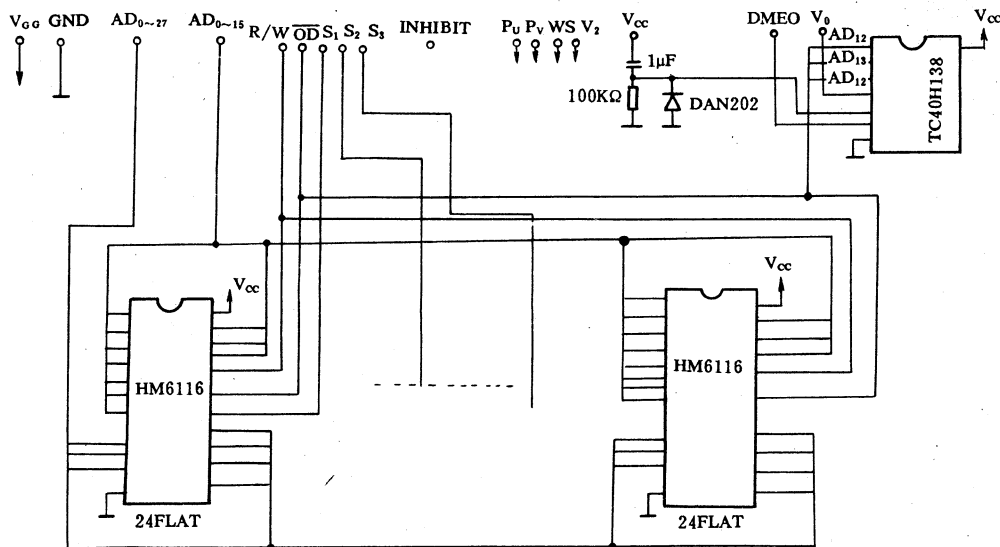
常见原因:

(1)CE-155 模块由 4 片 RAM 芯片 MH6116 组成,其中有三块的片选由主机 PC-1500 机的三线八选一译码分配电路 40H138 控制而另一片则由模块内的片选电路 40H138 控制片选,该芯片由主机 DME0, Y₀ 控制。当 40H138 损坏, Y₇ 输出低电平,受控的 MH6116 CS 端电平为低,使该片容量挂不上内存。可用相同的芯片代换。

(2)主机模块仓或模块插头接触不良,可用清洁剂擦洗插头部分。

(3)主机的 DME0 Y₀ 信号没有送到模块的片选电路,如因铜箔线被腐蚀,隔离元器件损坏。

(4)RAM 电路损坏,可用片状扁平封的 MH6116 替换。



图

(上接 48 页)

器上的 ROM 提供了进入系统的方便条件。系统为寻找有效控制器上的 ROM,在 C800 : F400H 地址段中,一个有效的 ROM 程序,其程序前几个字节的内容为:

C800:0000 55;第 0 字节内容

C800:0001 AA;第 1 字节内容

C800:0002 XX;第 2 字节为 ROM 程序长度

C800:0003 YY;第 3 字节从该字节开始为硬盘控制

制板上 ROM 程序及参数

对硬盘进行低级格式化的具体操作如下:

在 DEBUG 状态下:

(1)用显示内存单元内容的命令 D 把 C800:0003—000F 单元中的内容显示出来,以确认适配器 ROM。

(2)用反汇编命令 U 把 C800 : 0003—000F 之间的目标码以反汇编为源程序,查出跳转到硬盘低级格式化程序段的 JMP 指令所在存储地址的首地址的段内偏移量××××

(3)用运行命令 G 运行入口地址为 C800 : ××××开始的硬盘低级格式化程序。

—G=C : 800××××

即正式进入并且开始利用控制器上 ROM 程序中低级格式化程序对硬盘进行低级格式化。

例如长城 0520CH,20MB 的硬盘,利用 ROM 实现

硬盘低级格式化的过程是:

A>DEBUG

-D C800:0000 000F 55 AA EB E9 52 08-5B
42 58 44 30 36 28

-U C800:0003 000F

C800:0003 EBIJ JMP 0023

C800:0005 E95208 JMP 085A

C800:0008 5B POP BX

C800:0009 42 INC DX

C800:000A 58 POP AX

C800:000B 44 INC SP

C800:000C 30362028 XOR 12820],DH

-G=C800:0005

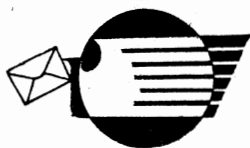
Hard Disk Formatting Utility v3.0

This program will erase all data on the specified hard disk

Enter a hard disk number (1 to 8) or press "9" to quit

此时只要正确回答硬盘的数目然后打回车键即开始对硬盘进行低级格式化操作。

不过有一点需要说明的是,不同厂家生产的硬盘适配器,其 ROM 中低级格式化程序入口地址××××可能有所不同,一般来看在 2—3 条 JMP 命令中 JMP 跳转 08××,往往是 ROM 中硬盘格式化的入口地址。在上例中××××是 0005。



普及型 PC 机个人用户软件交流联谊活动

读者联谊

问题解答(四)

中国农科院计算中心(100081) 王路敬

11. PC 机系统板上的 ROM BIOS 从结构上有哪几部分组成?

根据 ROM BIOS 所提供的功能,它有以下七部分组成:

(1)程序数据区

它包括:定义设备地址;在 0 段置 8088 中断向量号;在 40 段设置 ROM BIOS 数据区,包括系统显示参数及数据区、键盘数据区、软盘数据区、视频显示数据区、定时器数据区以及硬盘数据区、在 50 段设置附加数据区。

(2)自测试程序

包括如下内容:

键盘的测试;8088 处理器测试;ROM BIOS 测试;8237 DMA 控制器测试;基本 16K ROM 测试;CRT 适配器和显示 RAM 测试;建立 8088 中断向量表;ROM BASIC 测试;软盘驱动器的测试;扩展 RAM 的测试等。

(3)自举装入程序 INT 19H

(4)主要输入/输出设备的驱动程序

这些驱动程序主要有:

RS 232 I/O——INT 14H

键盘 I/O——INT 16H

软盘 I/O——INT 13H

打印机 I/O——INT 17H

显示器 I/O——INT 10H

(5)系统配置分析程序

它包括:

确定 ROM 的容量——INT 12H

确定系统配置的设备——INT 12H

(6)其他中断服务程序

这些服务程序包括:

不可屏蔽中断处理程序;

系统的时钟——INT 1AH;

屏幕打印——INT 05H;

系统使用的临时中断服务程序。

(7)图形方式的字符发生器图形提供 128 个字符 8*8 点阵图形。

12. 硬盘 ROM BIOS 由哪几部分组成?

在 PC 机系统中硬盘适配器上也有一片 ROM 组件。硬盘 ROM BIOS 主要提供磁盘输入输出驱动程序,它由以下五部分组成:

(1)硬盘 I/O 参数的定义

其中包括:

错误代码参数

设备地址端口

硬盘控制命令字

(2)硬盘数据区

它包括:

在 0 段设置磁盘中断向量号以及 40 段设置磁盘状态字。

(3)硬盘 I/O 的初始化程序

该程序的作用是使硬盘输入/输出驱动程序进入系统。

(4)自举装入程序——INT 19H

(5)硬盘 I/O 驱动程序。

在微机使用和维修中,常发现硬盘适配器(硬卡)出故障,例如,在诊断硬盘的过程中,若硬盘适配器内部诊断有故障或者控制器 RAM 诊断有故障其出错代码“1701”。故障形式多种多样,原因很多,可以从适配器 ROM 中得到帮助。为此,应首先了解它具有如下特点:

1. 硬盘适配器上的 ROM 参加了整个系统 BIOS 的统一编址,系统规定它的起始地址为 C800:0000(16 进制)。

2. 一个有效的 ROM BIOS 程序应具有如下的规定:

(1)第一字节内容为 55H

(2)第二字节内容为 AAH

(3)第三字节内容是长度指示器。ROM 的大小是以 512 字节块为单位。

(4)第四字节:EBH 一定是一条 JMP 跳转指令。

了解到这些特点,可以想办法检测它的正确性,为维护硬盘提供有用信息。例如,在硬盘的维护中,有时需要对硬盘进行低级格式化,则可以调用 ROM 中的低级格式化程序,既方便,操作又简单。但是,不同厂家生产的不同型号的硬盘适配器上 ROM 的对其硬盘进行低级格式化程序的入口不同。

IBM-PC/XT 及其兼容机的硬盘在必要的时候运行硬盘适配器提供的 ROM 程序中对硬盘进行低级格式化时,由于不同厂家生产的硬盘适配器其低级格式化程序的入口地址不同。所以有必要了解一种寻找低级格式化程序入口地址的方法。长城 0520 系列及其兼容机,由于配置了硬盘驱动器,该驱动器上的 ROM 组件,必须要进入系统。系统板的 ROM BIOS 为硬盘控制

(下转 47 页)