電子與電腦

一九九一年合订本

《电子与电脑》编辑部

《电子与电脑》
计算机爱好者之友

欢迎订阅　　欢迎投稿

ISBN7-5053-1540-4/TP. 289
定价：15. 50 元

电 子 与 电 脑

1991年合订本

《电子与电脑》编辑部编

電子工業出版社

(京)新登字 055 号

内 容 提 要

本刊是普及电子和电脑知识的科普刊物。设有 PC 用户、学习机之友、C 语言初级讲座、学用单片机、学装微电脑、电脑巧开发、维修经验谈、初级程序员级水平考试辅导讲座、软件介绍等栏目。本刊内容软硬兼顾、突出实用、资料透明、便于借鉴。

合订本将全年学习机和学用单片机栏目中(因期刊篇幅有限而未登出)的程序清单一并登出。

本刊是初、中级水平的计算机爱好者之友。

《电子与电脑》1991 年合订本
《电子与电脑》编辑部编
电子工业出版社出版(北京市万寿路)
电子工业出版社发行　各地新华书店经售
机电部情报研究所印刷
开本:787×1092 毫米 1/16 印张:39.5 字数:1400 千字
1992 年 2 月第 1 版 1992 年 2 月第 1 次印刷
印数:16000 册　　定价:15.50 元
ISBN7-5053-1540-4/TP.289

《电子与电脑》1991年总目录

·综述·　　期　页

为《电子与电脑》而作——中国计算机学会向广大计算机爱好者拜年 …… 1 2
今天我们相识——本刊91年致读者 …… 1 3
PAL和GAL …… 朱世鸿 2 2
磁盘和磁盘机 …… 林　兼 3 2
90年代的电脑技术和采购策略(上)…… 殷广济 4 2
90年代的电脑技术和采购策略(下)…… 殷广济 5 2
CGA,EGA,VGA …… 余　红 5 3
计算机技术的产物——条形码技术 …… 于江苇 5 4
80年代微机发展回顾及90年代展望(上) … 陈志雄 6 2
80年代微机发展回顾及90年代展望(下) … 陈志雄 7 2
永不过时的电脑软件——自生成软件系统… 郑　佑 8 2
参加91'第三届国际信息学奥林匹克(IOI'91)竞赛的体会 …… 吴文虎　薛宏熙 9 2
众说纷云:8位学习机是否已过时?
专家认定:君在教育界仍是主力军! …… 9 3
论电脑写作 …… 彭明光 10 2
在中华机上扩充中文输入码的探索 …… 胡志群　胡志广　胡沛云 11 2
新型信息录入技术-条码技术及其应用 …… 王枞等 12 2

·PC用户·

单色卡仿真CGA视频标准的图形功能 …… 朱大公 1 4
西文Wordstar文件在中文状态下的操作 …… 尤建忠 1 6
试论磁盘文件的最佳存储方法 …… 陈宏明 1 8
断电保护打印程序 …… 郭河伟 1 9
将三角函数引进DBASE系统 …… 何管略 1 10
经验与技巧两则 …… 吴仲文 1 11
中国计算机学会1990年度全国青少年信息学(计算机)通信赛试题 1 12
通用打印机状态检测程序 …… 夏伟文 2 3
dBASEⅢ与BASICA之间的数据传送问题 … 严桂兰 2 5
dBASE运用经验与技巧 …… 孙　鹏 2 7
如何解决在运行FOXBASE内存不够的问题 …… 周日初 2 9
COMPRESS在硬盘集中存储中的应用 …… 许　鹰 2 9
怎样在IBM-PC机上实现汉字的动态显示… 李晓华 2 10
DOS I/O重定向、管道功能及其使用 …… 徐国辉 2 11
在低牌本DOS下安装虚拟盘 …… 周健勇 2 12
五笔CCDOS中一字节修正 …… 吴帮忠 2 12
一个快速实用的字串替换程序 …… 金林樵 2 12
PC机磁盘文件的加密与解密 …… 刘忠达 2 13
硬盘正确格式化一例 …… 袁士彬 2 14
数据库通用维护模块设计 …… 徐维祥 3 4
字典库的编辑和使用 …… 黄焕如 3 7
用DOS命令设计菜单…… 陈　凯 3 8
在IBM PC/AT机异步通信适配器的自检方法 …… 杨　沂　杨耘华 3 10
产生真正的随机数 …… 应海涛 3 11
利用屏幕缓冲区实现语言的三重调用 …… 张建新 3 11
谈谈子目录加密与解密的一种方法 …… 李连德 3 12
PC机显示氢原子轨道波函数 …… 陈可中　曾庆涛　郭冰莹 3 12
BASIC程序P加密的简易解密法 …… 李志刚 3 13
用随机文件建立数据新探 …… 纪京松 4 5
英文Wordstar文件的标准ASCII化 …… 赵　毅 4 7
BASIC程序P保护在文件加密中的应用 …… 何　骏 4 7
在dBASEⅢ中实现多维数组功能 …… 石供非 4 8
为DOS增加定义功能键外部命令…… 张　红 4 9
设置MS-DOS的功能键 …… 何管略 4 11
编译BASIC运行文件CCDOS不相容的解决方法 …… 刘亚兵 4 12
硬盘的加锁和解锁 …… 张　文 4 12
键控图形处理程序 …… 徐维科 4 13
彩色的高分辨图形 …… 潘麟元 4 14
回溯法及其在编程中的应用 …… 陈　磊 4 15
通用打印机状态检测程序的改进 …… 夏伟文 5 5
通用制表程序 …… 赵　欣 5 6
也谈CdbaseⅢ的纵向送数方式 …… 王传信 5 9
PCTOOLS的使用技巧 …… 刘建明 5 11
程序常驻内存(TSR)的实现方法…… 李　华 5 12
防止非法使用DBASE程序的一种方法 …… 黄世刚 5 14
也谈IBM机图形存取 …… 焦小友 5 14
侦销核对程序 …… 张永辉 5 14
汉字同音字查询的实现 …… 李凤明 6 4
如何提高CdBASEⅢ数据库汇总速度 …… 刘昌斌 6 6
文件目录修复一例 …… 唐银红 6 6
多功能卡日历/时钟功能的开发 …… 张翠玲 6 8
Turbo C问答 …… 施悦华 6 8
就Turbo C谈充分利用系统内存 …… 宋卫东 6 9
女子健美体型电脑咨询程序 …… 黎彦才等 6 9
使清屏画面变美妙 …… 王　存 6 11
技巧二则 …… 汉　神 6 12
利用DEBUG进行反汇编存盘的方法 …… 金林樵 6 13
90年全国计算机通讯赛题分析 …… 汪五一 6 14
用未公布的dBASEⅢ命令开发图形功能 …… 金林樵 7 4
谈dBASEⅢ"模糊"查询 …… 班艳玲 7 5
FOXBASE编程技巧两则 …… 贺文华 7 5
恢复误删除文件的原理和方法 …… 朱大公 7 6
微机绘制平面曲线 …… 魏新民 7 9
若干子目录下共用一个dBASEⅢ(或WS)软件包 …… 龚文彪 7 11
小议PC-DOS的拷贝命令…… 刘　杰 7 11
能读多位数的计算机程序 …… 周礼森 7 12
模拟钢琴演奏 …… 冯剑秘 7 12
如何对中文DOS的汉字库进行直接读写 … 刘旭华 7 13
NC—新型文件管理工具软件 …… 杨　进 7 14
通用下拉式菜单的实现方法 …… 张志远 8 3
附录　程序清单

如何改变数据记录在库中的绝对位置 …… 李海飞 8 4
dBASE Ⅱ过程文件加密的发现 …… 刘人伟 8 5
对 SET CARRY ON 的一点补充 …… 李志刚 8 5
IBM PC/XT 磁盘扇区读取方法分析 …… 吴兆熊 8 6
使用设备驱动程序 ANSI·SYS 扩充
屏幕和键盘 …… 曹长英 李 杰 8 8
用 PC TOOLS 修改软件的汉字提示 …… 李开新 8 10
巧用批命令定义功能键 …… 邢观锋 8 10
CCDOS 下屏幕死锁的解决 …… 成 刚 杨明秋 8 11
VDISK 的妙用 …… 叶 舸 8 11
IBM PC/XT 定时中断服务程序设计 ……
…… 张 辉 李东升 李奎贤 9 5
通用数据库修改程序 …… 周日初 9 7
PC-DOS 命令 WHEREIS 的改正 …… 李 敏 9 8
巧算 1000! …… 李 帆 9 9
又发现一种新病毒 …… 王家春 9 10
杨基病毒快速解毒法 …… 李凤明 9 10
设备驱动程序 ANSI.SYS 的使用 …… 王立勇 9 11
巧用虚拟盘 …… 李焕民 9 12
用递归解九连环 …… 钟 武 9 12
BROWSE 命令中 FIELDS 参数的菜单选择 … 张述健 10 4
给 dBASE 程序加入活用功能 …… 贺映文 10 5
在 dBASE Ⅲ中实现多维数组功能的另一方法 … 程林 10 6
集成字处理软件 CCED 的使用技巧 …… 许 鹰 10 6
如何自编 Park? …… 葛芝宾 10 7
需 ROM BASIC 支持的软盘改造 …… 曾礼伟 10 8
最新病毒检测及清除软件 VIRUSCAN Ver7.2V77
…… 梁伟杰 10 9
Goldbach 奇数猜想的 BASIC 程序 …… 陈君佐 10 10
快速排序"快"在何处? …… 应 宏 10 11
两个高效率的排序程序 …… 李军杰 10 11
利用 PASCAL 扩充 dBASE Ⅲ的函数功能 ……
…… 王 越 张秀邦 11 4
如何动态生成有记忆功能的菜单 …… 苏士俊 11 6
在 dBASE Ⅲ下实现菜单的光带键选 …… 阮自力 11 7
双列分页打印程序 …… 陈 锋 11 8
广州大学一号病毒解除与免疫 …… 张文锋 11 10
谈谈 2.13 汉字系统在软盘上的使用 …… 廖红旭 11 11
PC/XT 机病毒报警程序 …… 王传智 刘秉瀚 11 12
考生姓名编码自动生成程序 …… 任绥海 11 12
dBASE Ⅲ下实现 Foxbase 中 prompt 的菜单功能 ……
…… 毛维平 12 4
信息系统中模糊查找的设计 …… 戚靖山 朱玉环 12 5
高密软盘的单驱动器拷贝 …… 李 毅 12 6
用汇编程序显示硬盘主引导记录 …… 李日林 12 7
PC DOS 内存管理 …… 陈 宏 12 8
CC BIOS 2.13F 的小补钉 …… 严仲武 12 9
中文弹出式窗口设计 …… 姜咏江 12 10
一种实用新颖的排序方法 …… 朱国江 12 12
·学习机之友·
APPLE Ⅱ高分辨率绘图与文字编辑技术 … 邹培国 1 15
向量图形表自动生成程序 …… 万维侠 1 18
学生成绩管理程序 …… 朱 江 1 20
CEC 控制键的调整及保护 …… 张保田 1 21
磁带小汇编系统自制简法 …… 石 勇 张优方 1 22
巧用 FID 解密 …… 杨 克 1 22
CALL 新用法 …… 姜 路 1 23
APPLE Ⅱ的口令设置 …… 成 鹰 1 23
性格类型测试程序 …… 叶 平 1 23
程序出错处理与 RESUME 的功能扩充 …… 丁志伟 2 15
特殊 6502 指令 …… 李 铁 2 17
四舍六入法程序 …… 韦肖鸿 2 17
"怪"指令在软件加密中的应用 …… 傅 剑 2 18
双显示电子钟 …… 万 斌 2 19
苹果机汉字造型表自动生成程序 …… 刘士明 2 20
高效率的排序方法 …… 陈庆祥 2 21
演示抛物面的光学效应 …… 王晓林 2 23
LASER QZT/C 应用经验 …… 张海翔 2 24
BASIC 语言实现递归 …… 苏亚华 3 14
数据处理原理和技巧 …… 朱国江 3 15
哥德巴赫偶数猜想的 BASIC 程序 …… 陈君佐 3 18
梵塔移动的动态模拟 …… 刘 闯 3 19
查找内存中的代码 …… 陈 宇 3 20
奇妙的图案 …… 李 明 3 20
最简数据盘生成法 …… 江 东 3 20
超级 8—40 单元打印 …… 王 壮 3 21
将 16 进制代码翻译成 DATA 语句的数据 … 张振堂 3 21
关于多目录磁道问题 …… 张剑平 3 22
电子表键盘输入方式的改进 …… 田洪瑜 3 22
LASER 301 兼做数字石英钟 …… 郑嘉琦 3 23
中华学习机使用技巧 …… 任 义 4 16
实用改良的 NEWCODE-POKES 程序 …… 黄影昌 4 17
巧解"八皇后"问题 …… 柳 杨 4 17
对《APPLE 和 CEC—I 设计者的疏忽》一文的看法 ……
…… 刘 峰 4 18
APPLE 缺陷的解决办法 …… 陈盛施 4 18
数值计算中存在问题的解决办法 …… 朱洪波 4 18
无数据磁盘格式化程序 …… 姜宏岩 4 19
跟踪机器语言程序的运行 …… 田俊成 4 20
局部清屏程序 …… 程 华 4 20
扩展 BLOAD 命令的功能 …… 蒋 阳 4 21
从"重赏"谈起 …… 卢保堂 卢玉堂 4 21
在 APPLE SOFT 下使用小汇编 …… 稽克威 4 22
工资票面张数计算程序的简化 …… 孙承灿 4 22
让 CEC-BASEC 也有自动行号功能 …… 沈东山 5 17
图形屏幕坐标的对应存储单元 ……
…… 葛海东 5 18
神奇的 $COBO 单元 …… 李 峰 5 18
怎样解读 CEC-I 学习机的汉字系统 …… 刘文勤 5 18
用机器语言解八皇后问题及其推广 ……
…… 卢祖兴 刘超宏 5 19
也谈汉字平移显示 …… 李茂学 5 19
前台显示后台绘图的秘密 …… 谢吉华 5 20
演示抛体运动 …… 白梅兰 5 20
多单元电视接收无线设计程序 …… 史文忠 5 22
一个实用的造型表编辑程序 …… 左震宇 5 23
CEC-I 全功能造字 …… 傅叔平 6 16
梵塔问题和奇数的幻方的一行程序 …… 陈 纲 6 18
APPLE-Ⅱ机打印格式的控制 …… 张冠玉 6 19
将 BASIC 程序变成 B 型文件并以 BRUN 运行 ……
…… 杨 峰 6 20

APPLE DOS 的一处错误 …… 尹进渝 6 20
ProDOS 系统盘的修改 …… 廖 凯 6 21
制作 40 磁道磁盘的最简方法 …… 瞿 波 6 21
PC-1500 机新监控程序…… 沈玉波 6 22
关于四舍五入程序的讨论 …… 梁 放 叶德祥 6 25
为无驱动器系统设置虚拟磁盘 …… 刘庆丰 7 15
巧用 LOGO 解难题 …… 廖大勇 7 16
修改 CEC 汉字系统程序 …… 周 易 7 16
浅谈计算机在中专数学中的应用… 王志平 顾正刚 7 17
数据块的后移 …… 冯端品 7 18
从 QXC 卡读取汉字字模 …… 牛海峰 7 18
ProDOS 磁盘操作系统入门 …… 廖 凯 7 19
在 APPLESOFT 系统下实现 HSCRN 功能 … 李 齐 7 21
BASIC 语言中几个命令功能的扩展 …… 那履弘 7 21
谈谈 FORTH …… 丁志伟 8 12
再谈特殊 6502 指令 …… 李永前 8 13
扩展监控系统命令 …… 蒋建一 8 14
MZ－731 机绘制的财务计划执行分析报告… 黄浩珍 8 14
机器语言程序计算阶乘 …… 易鹤龄 8 17
ProDOS 磁盘操作系统入门(续) …… 廖 凯 8 18
用堆栈思想处理任意 N 个元素的排列组合问题……
…… 索素文 8 20
简单有趣的解密办法 …… 张 亭 8 22
将 CEC－I 中华学习机改制成专用机 …… 姜良夫 9 14
用栈结构优化程序 …… 陈合力 9 17
寻找无理数 e 的约率和密率 …… 施为钢 9 17
怎样节省 CEC-I 存储空间 …… 尹向阳 梁 曦 9 18
ProDOS 磁盘操作系统入门(续) …… 廖 凯 9 19
数制转换简法 …… 李 峰 9 20
也谈扩展 COPY Ⅱ PLUS4. 0 …… 关炳坤 9 21
从 CP/M 转移文件到 APPLE DOS 下 …… 方福兴 9 21
多功能 CEC-I 高速排序 …… 王庆顺 9 22
小学生速算训练 …… 李永升 9 23
CEC-I 高级动画制作技术 …… 黄一禾 10 13
警察抓小偷 …… 李 明 10 15
ProDOS 磁盘操作系统入门(续) …… 廖 凯 10 16
高性能的 CEC-I 自定义组合键 …… 杨勇佳 10 18
活泼的光标 …… 殷瑞冬 10 19
CEC-I 机汉字使用第十一行 …… 任 义 10 19
三页图形竖向并打 …… 陈 钢 10 20
电子打字机 …… 王 蔚 10 20
把 APPLE DOS 移入 RAM 卡 …… 傅 剑 11 14
将程序 CHAIN 升级为 APPLE DOS 命令 ……
…… 赵江龙 孙杰亮 11 15
向天坛机移植中华机汉字软件的一些体会与经验 ……
…… 张 斌 11 17
FORTH 问答堆栈和逆波兰表示法 …… 丁志伟 11 18
中华学习机图书管理程序 …… 王二珠 11 19
ProDOS 磁盘操作系统入门(续) …… 廖 凯 11 21
多层次全功能阴影菜单 …… 黄文忠 12 13
虚线 …… 刘 裕 12 14
技巧两则 …… 宋正荣 12 15
改进 MUSIC 的发音 …… 熊 波 12 15
中华学习机 RAM 监控及其应用 …… 朴一兵 12 16
CEC-I 键控光标作图程序 …… 陈国良 12 17
ProDOS 磁盘操作系统入门(续) …… 廖 凯 12 18

·C 语言初阶·

《C 语言讲座》第一讲 TurboC …… 李文兵 1 25
第二讲 Turbo C 基本数据 …… 李文兵 2 25
第三讲 命令行 Turbo C 和实用程序…… 李文兵 3 24
第四讲 C 运算符 …… 李文兵 4 23
第五讲 程序控制结构(一) ……
…… 李文兵 王玉华 蓝智斌 5 25
第五讲 程序控制结构(二) ……
…… 李文兵 王玉华 蓝智斌 6 23
第六讲 宏指令的用法 …… 李文兵 7 23
第七讲 结构数据(一) …… 李文兵 8 22
第七讲 结构数据(二) …… 李文兵 9 24
第八讲 指针与数组 …… 李文兵 10 21
第九讲 指针与函数、结构 …… 李文兵 11 23
第十讲 文件管理 …… 李文兵 12 20

·学用单片机·

单片机化产品中的键盘和显示器的设计…… 吕锡庆 1 29
EPROM 及 8751 安全智能编程法… 翁甲辉 李 柯 1 31
普及型单片机开发工具的硬件设计 ……
…… 张培仁 刘振安 2 29
普及型单片机开发工具的软件设计 ……
…… 张培仁 刘振安 3 28
智力竞赛抢答装置 …… 宋济华 4 27
单片机电子音乐编程实践 …… 张培仁 张新宇 4 30
简易单片机电子琴 …… 张培仁 张新宇 5 30
MCS－51 单片机在听力播放、发射系统中
的应用 …… 沈乃才 宋振环 5 32
单片机最小应用系统 …… 张培仁 6 26
电脑坦克—KDC 最小系统应用实例之一 ……
…… 张培仁 严 琦 阮永光 6 27
单片机旅馆客房门卫系统 …… 傅士淦 7 27
单片机汉字打印驱动程序的设计思想 …… 尹 征 8 25
MCS－51 单片机应用中二种 LED 显示方法… 傅篱 9 27
多功能电子钟 …… 张培仁 刘振安 10 27
汽车转弯信号灯控制 …… 张培仁 刘振安 11 26
SCB－I 型单板单片机的存贮器改进…… 江 琪 12 26
MCS－51 单片机端口结构图示中的一个普遍错误 ……
…… 王修才 12 27

·学装微电脑·

学装 μp-80 手动 EPROM 写入器 …… 易齐干 1 33
40 支开关输入部件的制作 …… 易齐干 2 34
8 位 A/D 变换部件的制作 …… 易齐干 3 31
数字游标卡尺的制作 …… 易齐干 4 34
用微电脑检查印刷电路板 …… 易齐干 5 37
微电脑控制电梯模型的实验与制作 …… 易齐干 6 33
微电脑控制跨接弯制机 …… 易齐干 8 29
相位控制部件的制作 …… 易齐干 9 29
Z80 CPU 与单片微处理器 8031 的交换 … 高殿斌 11 28
串行 I/O 部件的制作与实验 …… 易齐干 12 28
“微电脑控制技术”函授班总结 …… 易齐干 12 30

·电脑巧开发·

微电脑控制电饭锅 …… 万国泰 1 35
MP－I 汉字及图形显示装置 …… 陈名则 2 37
APPLE-Ⅱ彩灯自控电路 …… 张孝玖 2 38

MP－I 微电脑加装编码键盘 …… 李志平 3 36
扫描式红外线十路遥控器 …… 王刚勤 3 37
语言报时钟的原理与实现 …… 蔡莲红 5 28
8031 与 PP40 描绘器接口技术 …… 皇甫文忠 6 29
解决 TEC-B1 不能使用 AP51－I 卡
问题的方法 …… 刘希栎 6 32
MSX-EPROM 读写/控除器 …… 刘 园 7 32
如何用 EPROM 读写器对 EEPROM 编程 ……
…… 闫怀兵 陈立山 8 35
LASER-310 机加接数字音响发生器 …… 王建锋 8 36
怎样用电脑检修录像机 …… 李玉全 9 32
用娃娃机开发单片机程序 …… 张 涛 9 32
简易精确的 APPLE Ⅱ时针卡 … 姜仕仁 施青松 10 30
用 EPROM 制作多路报警器 …… 王洪伟 10 32
在 APPLE 机接口卡中固化 BASIC 应用程序 ……
…… 解武杰 11 30
设计微电脑控制电路的小常识 …… 王 林 11 31
型号不同的绘图仪绘制 TANGO
软件图形的方法 …… 黄 健 陈 军 11 32
将 MP－I 电脑的时基信号为晶体振荡…… 邓文超 12 31

·维修经验谈·

DM－2214 高分辨显示器无字符的检修及
提高字符清晰度的研究 …… 杨鼓行 1 37
王安机 RAM 故障简易检修法 …… 董纯坚 2 40
IBM286 维修一例 …… 叶志斌 2 41
中华学习机典型故障一例的分析 …… 杨瑞华 2 42
APPLE Ⅱ主机常见故障的检修 …… 陈 鸿 2 42
软盘驱动器机械故障的分析与维修 …… 魏学则 4 44
更换集成块一法 …… 吴广元 4 44
APPLE－Ⅱ机内存故障维修方法 …… 腾 达 4 45
巧修 CEC－I 受损键盘 …… 黄文贵 4 45
彩色显示器亮度失控的探讨 …… 胡野红 6 45
微计算机的维修工具－8088 系统板故障诊断卡……
…… 董 珊 6 46
任天堂系列游戏机检修实例 …… 林华春 7 34
让 PC 机接上家用彩电——CGA-PAL 卡即将问世 … 7 35
微计算机维修工具－8088 系统板故障诊断卡(续)……
…… 董 珊 7 36
磁盘驱动器修复一例 …… 雷 勇 7 38
任天堂系列游戏机检修实例(续) …… 林华春 8 38
“8088 系统板测试卡”对系统板总线部分的诊断测试……
…… 梁杰熙 8 39
一九九一年计算机初级程序员有奖竞赛试题 …… 8 41
一九九一年计算机初级程序员有奖竞赛试题答卷 … 8 48
TH8070 打印机接口电路工作原理与故障分析 ……
…… 李燕华 9 35
长城 0520CH 机打印机适配器故障维修…… 邱 槟 9 37
AT(286)微机硬盘处理一例 …… 刘文峰 10 35
从屏幕显示异常判断苹果主机板故障 …… 张艾美 10 36
M2024 型打印机故障检修一例 …… 王 峰 10 37
计算机设备故障检修表 …… 晋金省 宋靖涛 10 38
IBM PC/XT 微型计算机检修一例…… 平 凡 10 39
MP－I 微电脑快速检修法 …… 李 红 10 40
如何使九针打印机字迹更清晰 …… 陆秀忠 10 41
YOKO747 型教学语言学习机的修理…… 郭长福 10 41
微机软盘驱动器“卡盘”的简易修复 …… 乐冬青 10 42
CEC－I 软驱故障维修一例…… 潘 冬 10 42
“坏”软盘的再生 …… 张 智 10 43
GW286BH 系统板维修一例 …… 董纯坚 10 43
怎样使用清洗盘 …… 瞿新国 10 43
驱动器修理一例 …… 张红平 11 35
硬磁盘驱动器专用接口电路 501262 的测试 ……
…… 卜建辉 11 36
IBM－PC/XT 一种系统错误的分析 …… 姜金友 11 41
用非格式化方法修复硬盘一例 …… 周建辉 11 42
检修 IBM AS/400B35 故障二例…… 王志远 11 42
自己学修计算机 …… 陈 勇 11 43
TEC-I 机磁盘驱动器 I/O 接口板故障检修一例 ……
…… 崔时珍 玄勇活 11 45
3070 打印机断针定位方法及 8088 汇编程序 ……
…… 龙兵生 11 46
打印机故障维修－以 STAR 或 GX-15 打印机为例……
…… 王 威 王 候 11 47

·初级程序员级水平考试辅导问答·

1991 年初级程序员(有奖学习)水平考试辅导讲座
第一讲 计算机基础知识(上) …… 王路敬 1 39
计算机基础知识(下) …… 王路敬 2 43
微型计算机操作系统使用问答 …… 王路敬 3 40
微型数据录入操作回答 …… 刘 英 张茂芝 4 40
BASIC 语言程序设计回答 …… 李志刚 吕 品 5 44
汉字 dBASE Ⅲ使用技巧(上) …… 王路敬 6 37
汉字 dBASE Ⅲ使用技巧(中) …… 王路敬 7 39
汉字 dBASE Ⅲ使用技巧(下) …… 王路敬 9 39
91 年第三届国际信息学奥林匹克(IOI'91)试题 ……
…… 吴文虎 9 46
Z80 的“新”指令 …… 章永得 9 47

·新书与软件·

ORACLE 关系数据库管理系统 … 王学民 魏 国 1 44
Turbo CAD 软件简介…… 2 46
VIRUSCAN 病毒处理软件包功能简介 …… 朱 禹 4 46
Microsoft Windows 3.0 窗口软件简介 …… 4 46
最新版本 PC TOOLS V6.0 …… 4 47
TANGO 印刷线路板自动布线软件包 …… 7 45
电子工业出版社新书介绍 …… 7 45
电子工业出版社软件部发行 CCED V3.0 中文字
表编辑软件 …… 7 46
给 TANGO 软件增加汉字元件库 …… 黄 健 陈 军 8 47

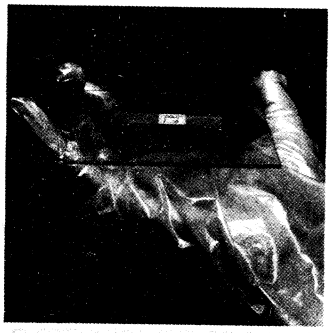
·读者联谊·

普及型 PC 机个人用户软件交流联谊活动 …… 1 46
湖北地区中华学习机联谊活动情况 …… 2 48
致参加“普及型 PC 个人用户软件交流联谊活动”
的用户 …… 农科院计算中心 7 47

·游戏机电脑·

键盘电视游戏机的 FBASIC 语言及程序设计(上) ……
…… 于春 10 33
键盘电视游戏机的 FBASIC 语言及程序设计(下) ……
…… 于春 11 33
第二章 FBASIC 的基本语句 …… 于春 12 32
巧用电脑作幻方 …… 郭高惠 2 47

E&C 電子與電腦



一九九一年
总期第70期

電子與電腦

目　录

·综述·
为《电子与电脑》而作——中国计算机学会向广大计算机爱好者拜年 …… (2)
今天我们相识——本刊91年致读者 …… (3)

·PC用户·
单色卡仿真CGA视频标准的图形功能 …… 朱大公(4)
西文Wordstar文件在中文状态下的操作 … 尤建忠(6)
试论磁盘文件的最佳存储方法 …… 陈宏明(8)
断电保护打印程序 …… 郭河伟(9)
将三角函数引进DBASE系统 …… (10)
经验与技巧两则 …… (11)
中国计算机学会1990年度全国青少年信息学(计算机)通信赛试题 …… (12)

·学习机之友·
APPLE Ⅱ高分辨率绘图与文字编辑技术 … 邹培国(15)
向量图形表自动生成程序 …… 万维侠(19)
学生成绩管理程序 …… 朱　江(20)
CEC控制键的调整及保护 …… 张保田(21)
磁带小汇编系统自制简法 …… 石勇　张优方(22)
巧用FID解密 …… 杨　克(22)
CALL新用法 …… 姜　路(23)
APPLE－Ⅱ的口令设置 …… 成　鹰(23)
性格类型测试程序 …… 叶　平(23)

·C语言初阶·
《C语言初阶》讲座
第一讲　Turbo　C …… 李文兵(25)

·学用单片机·
单片机化产品中的键盘和显示器的设计 … 吕锡庆(29)
EPROM及8751安全智能编程法 … 翁甲辉　李柯(31)

·学装微电脑·
学装μP－80手动EPROM写入器 …… 易齐干(33)

·电脑巧开发·
微电脑控制电饭锅 …… 万国泰(35)

·维修经验谈·
DM－2214高分辨显示器无字符的检修及提高字符清晰度的研究 …… 杨鼓行(37)

·初级程序员级水平考试辅导问答·
1991年初级程序员(有奖学习)水平考试辅导讲座
第一讲　计算机基础知识(上) …… 王路敬(39)

·新书与软件·
ORACLE关系数据库管理系统 …… 王学民　魏国(44)

·读者联谊·
普及型PC机个人用户软件交流联谊活动 …… (46)

·信息与服务· …… (47)

·封面·
未封装的单片微电脑芯片

·封三·
LASER 310参考电路(资料) …… 李敬茂　供稿

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:北京市邮政局
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:张　丽
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

为《电子与电脑》而作

——中国计算机学会向广大计算机爱好者拜年

1991年来到了。这一年正是我国"八五"计划和十年规划开始执行之时。九十年代是实现我国社会主义现代化建设总体目标的关键阶段。这将是我们伟大祖国奋进的关键阶段，也是计算机为促进国家的经济和社会的发展大有用武之地的九年。在这个10年中，我们将迎来电子计算机——人类最重要的先进技术工具——诞生的五十周年(1946——1996)和我国计算机界的学术团体、中国计算机学会成立三十周年(1962——1992)。在此，我们向全国计算机爱好者祝贺新年，祝大家在新的一年里取得新的收获，祝你们在学习和运用计算机的知识和技术为祖国的四个现代化建设事业做出新的贡献。

九十年代计算机将成为人类认识世界和改造世界的高效得力助手，计算机将会愈来愈显示出它是人类的重要资源和财富。因此，我们无论是在什么工作岗位上或是从事什么职业，了解和学习计算机知识犹如你必需学好政治，语文和数理化等文化知识一样，要学好计算机这样一门第二文化知识课。我们这样说，绝不是因为我们是从事计算机这一行的，而片面强调计算机的重要性。这是40多年自从电子计算机诞生以来而被世界各地的经济技术进步的事实所证明了的。计算机及其应用领域也不局限在计算机这一行，它已是诸家百行的工具了。就象前几世纪强调机床是促进技术进步的重要条件一样。

从计算机是一个工具的角度来看，九十年代计算机发展的总趋势是什么呢？是更好地解决人机接口技术，改善人机界面。使得人们能够更高效、更方便地使用计算机，而不必需掌握计算机内部的硬软件科学技术，只需了解计算机这一工具的外特性和功能，正是从这样的概念出发，计算机才象政治、语文等是第二文化，是人们普遍需要学习的。

机电部电子工业出版社主办的《电子与电脑》刊物正是为满足人们，特别是年青人对计算机知识和应用技术的需要而创办的。其读者对象是广大计算机和电子技术爱好者，是面向初学者和初中级技术人员的。这部分人中间应当说绝大部分是青年人。中国计算机学会对于青年科技工作者的工作是十分重视的，学会的两位理事长胡启恒和张效祥教授都多次说过：学会要办成年青活泼面向未来的学会，要给青年人更多的机会，每届理事会更选时，希望有更多年青同志加入我们这个队伍，我们这个学会要奖掖后进，注意吸收青年人参加学会的工作与活动，使他们得到更宽广的午台，更快地成长为一代年青的计算机科技工作者。

大家知道，我国的计算机事业开创于五十年代中期。创建计算机事业的老一辈专家现在大部已超过了70岁。就是五十年代大学毕业的学生进入计算机界的这一批人，到了本世纪末，也早已超过了退休年龄。因此，我国计算机的这一行能不能真正起到成为祖国四化建设的有力工具，重担已经历史地落在年青一代的肩上了。这就是我们在20世纪最后九年开始的时候，要想说出的心里话，并以此作为写给《电子与电脑》的新年祝词。从国际计算机界的统计，不少在计算机及其应用领域有所建树、有所成就的，大都是30岁左右的人们所建造的。因此，使我国的年青一代尽快地涌现在国内国际午台上，成为知名专家、管理行家和技术工人，乃是我国计算机事业兴旺发达的重要标志。

同时，我们也看到我国年青一代计算机科技工作者在老一辈专家的培养指导下，正在可喜地成长，在我国和国际学术会议上，年青人发表了许多具有国际水平的论文报告，受到了国际计算机界的重视。也还看到了我国的青少年在计算机知识的学习和普及方面的进步。在1989年和1990年首届和第二届奥林匹克信息学(计算机)的国际大赛上取得了好成绩，共获得金牌1枚，银牌2枚，铜牌4枚，而且是在世界各国参赛队中间，唯一全部参赛选手都拿到奖牌的一个队。受到了世界各国和联合国教科文组织的重视与关注。

中国计算机学会将会努力支持和配合《电子与电脑》刊物做好这些方面的工作。我们相信，我国的年青一代人一定能够把我国的计算机事业继往开来，开创一个新的振兴局面。

祝广大计算机及其应用工作者和爱好者新年好！

(中国计算机学会办公室供稿)

今天我们相识

——本刊91年致读者

在91年到来之际，又有众多的“电脑迷”们加入到我们《电子与电脑》的读者和作译者行列中来了。我们为有数万名关心和支持我们的老朋友，为又结识越来越多的新朋友，感到非常欣慰和鼓舞。

今天，电脑作为一种“智能型工具”，已经和正在深入的渗透到社会生活的各个领域。电脑的使用程度如何?从某种意义上说，可以反映出一个国家科技和生产发展的水平。大家知道，要学好和用好电脑，是需要掌握一些基础知识、使用方法和技巧的。对于广大没有机会接受电脑系统教育而要应用电脑的朋友来说，对于已具备一定电脑知识而缺乏实用经验的朋友来说，都希望有一个知己的学友，《电子与电脑》将会成为您最好的搭挡和帮手。

本刊在90年第9期上，刊出了“《电子与电脑》读者、作译者意见征询表”后，已陆续收到数百封信函的反馈意见。对于大家提出的许多宝贵建议，编辑部进行了认真的综合研究，安排了91年的工作，这里，特向本刊广大海内外读者汇报，希望得到各位的支持和帮助。

《电子与电脑》读者对象：初、中级水平的计算机爱好者。

《电子与电脑》办刊风格：软硬兼顾，突出实用、资料透明、便于借鉴。专栏与函授结合、期刊与专辑结合，将刊物办成开放式社会大学校。

《电子与电脑》主要栏目有：

“综述”拟请计算机界的专家，撰写有关计算机科学、应用、教育等方面的文章。

“PC用户”、“学习机之友”结合国产0520系列机、普及型PC、IBMPC及其兼容机，CEC－I、Apple、Laser、PC－1500等国内企事业单位、大中小学及个人常用机型，刊登有关操作系统的深入剖析、编程方法和技巧、软件汉化、字表处理、图形设计、加密解密、消除病毒等实用程序。注意刊登中小学计算机辅助教学及青少年的作品。

“学装微电脑”结合MP－I、μP－80等教育套件、为广大青少年和非计算机专业的技术人员提供以组装方式、从硬件入手学习微电脑技术。

MP－I以6502为主。为进一步开发利用或维修CEC－I，Apple等机型的读者提供学习硬 件手段。μP－80以Z80为主。为学习微电脑控制技术的读者提供自学的系列套件。91年该栏目将刊出有关MP－I的初级应用及μP－80用于机电一体化控制方面的实例。

结合此栏目，中国计算机学会以MP－I为主、继续组织开办第三期“学装微电脑”函授班；本刊委托天津纺织工学院开办以μP－80为主的“微电脑控制技术”函授班，为愿意深入，系统学习的读者提供一个保证学会的条件。

“学用单片机”结合目前国内常用的MCS－51系列单片机最小系统，选登一些典型的应用及开发资料。使初级用户可以拿来就用，为广大电子爱好者快速学会如何将单片电脑用于自己设计的电子产品中去。

结合此栏目，我刊委托中国单片机学会和南京大学计算机系开办“单片机技术”函授学习班。

“电脑巧开发”结合各类初级电脑的功能扩充和性能改进，刊登有关电路设计，以及有关接口电路、电脑控制家电等方面的文章。

“初级程序员级水平考试辅导”“C语言初阶讲座”使想系统学习计算机知识的读者，通过自学考试的途径，完成从初级→中级→高级程序员级全国统一考试，获得资格证书。

结合此栏目，我刊同中国软件行业协会高档微机协会举办“全国初级程序员级考试辅导”，讲座及有奖竞赛。

“读者联谊”这是应读者要求开办的新栏目，此栏将介绍各地电脑爱好者的联谊活动情况；读者软硬件的交流信息；编辑部难以及时回复的读者提问；读者或单位求援的信息等。

结合此栏目，今年将试办“普及型PC个人用户软件交流联谊”活动，委托中国农业科学院计算中心培训部作为此活动的联谊点。其目的在于针对目前国内逐渐增多的普及型PC个人用户，普遍感到缺乏软件的情况提供一个相互交流，开发及共享软件的机会。

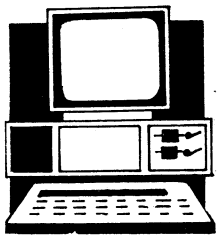
读者有关CEC－I、Apple等学习机软件登录、评测、出版工作，由电子工业出版社软件部负责办理。

为了更好的为读者服务，我们将努力办好“读者服务部”。由于缺乏经验、加之人力财力所限，许多工作不能等充分搞好了才开展，而往往是读者提出要求后，我们立即增办的，因而可能出现服务信息不能及时反馈给读者的情况。今后，我们会应读者要求，力争将服务项目增的多一点，水平提的高一点。

在新的一年到来之际，我们会使自己的工作再上一个台阶，以不断前进的面貌出现在读者面前。希望广大计算机科研、教育、产业界，有关的学会、协会、广大的计算机爱好者，不断给我们以指导和帮助。

在新的一年到来之际，《电子与电脑》编辑部全体工作人员，向关心和支持我们的各界朋友、致以新春的问候，祝各位事业成功，合家幸福。

《电子与电脑》编辑部
全体工作人员
1991.元旦.



单色卡仿真 CGA 视频标准的图形功能

朱大公

目前，国内流行一种性能/价格比高；且与 IBM－PC 兼容的准十六位微型计算机。这种微型计算机大都配置单色卡(Her cules)和单色显示器。在这种配置下，大多数具有图形功能的软件，包括许多由 CCDOS 支持、在 IBM－PC 机上运行的汉字软件无法在该类微型机上运行。编制一份实现单色卡仿真 CGA 视频标准的图形功能的软件便可以解决这一问题。本文从三个方面对此方法加以详细介绍。

一、CGA 视频标准

众所周知，CGA 视频标准支持两种显示接口卡(显示适配器)、八种视频显示工作模式。其中，彩卡(彩色适配器)支持七种工作模式，而单色卡(单色适配器)则只支持一种工作模式。现将这八种工作模式列表于下：

工作模式号	工作模式	说明
00	40×25	单色字符模式(40 字符/行，25 行/屏)
01	40×25	彩色字符模式(40 字符/行，25 行/屏)
02	80×25	单色字符模式(80 字符/行，25 行/屏)
03	80×25	彩色字符模式(80 字符/行，25 行/屏)
04	320×200	中分辨率彩色图形模式
05	320×200	中分辨率单色图形模式
06	640×200	高分辨率单色图形模式
07	80×25	黑白字符模式(用于单色卡)

系统工作于哪种模式是由系统的硬件配置和 ROM BIOS 中的显示管理模块(10H 中断)的 00 号功能调用决定的。当系统配置单色卡和单色显示器时，如不采取特殊措施，只能工作于 07 号模式。这是在此种配置下不能运行图形软件的关键之所在。IBM－PC 机上的汉字应用软件绝大多数都工作于 06 号模式。要想在此种配置下运行工作于 06 号模式的汉字软件，必须使其工作模式由 07 号改变为 06 号。简单地调用 10H 中断的 00 号功能是不能实现这种转变的。

二、CGA 视频标准的实现

在 IBM－PC 机的显示接口卡上有一块大规模集成电路芯片，称为 CRTC 6845。显示接口卡的硬件电路和 ROM BIOS 中的 10H 中断均是以它为核心进行设计的。CGA 视频标准的实现，主要取决于对 CRTC 6845 的初始化编程和使用不同的显示缓冲区。

CRTC 6845 有 19 个内部寄存器，其中 1 个为地址索引寄存器，其余 18 个为数据寄存器。18 个数据寄存器编号为 R0～R17，与显示模式有关的是 R0～R11。按照常规，作为外设端口的 18 个数据寄存器要占用 18 个 I/O 端口地址，但 CRTC 6845 采用了一种特殊的设计，以地址索引寄存器表示选中 CRTC 6845，以地址索引寄存器的内容表示选中 18 个数据寄存器中的某一个。这样，CRTC 6845 只占用两个 I/O 端寄地址。无论是彩色卡，还是单色卡，向数据寄存器 R0～R11 写入不同的参数，是使显示器工作于不同的显式模式必不可少的操作。在单色卡上，CRTC 6845 的地址索引寄存器使用端口地址 3B4H，18 个数据寄存器公用一个 I/O 端口地址 3B5H。在彩色卡上，CRTC 6845 的地址索引寄存器使用 I/O 端口地址 3D4H，而 18 个数据寄存器则公用 I/O 端口地址 3D5H。下面一段程序说明怎样向 CTRC 6845 的数据寄存器写入参数。其中，名为 CUANSHU 的变量存放有对 CRTC 6845 初始化编程所需的参数。

```
        LEA     SI,CUANSHU
        MOV     DX,3B4H
        MOV     CX,12
        XOR     AH,AH
LOOP1:  MOV     AL,AH
        OUT     DX,AL
        OUT     DX,AL
        INC     DX
        LODSB
        OUT     DX,AL
        INC     AH
        DEC     DX
        LOOP    LOOP1
```

显示接口卡上除了核心芯片 CRTC6845 外，还有一定数量的 RAM 芯片。这些芯片所处的内存地址区域称为显示缓冲区。在彩色卡上，显示缓冲区的首地址为 B800:0000，其容量不小于 16KB；在单色卡上，显示缓冲区的首地址为 B000:0000，其容量不小于 4KB。在完成对 CRTC6845 的初始化编程后，为了得到一帧清洁的幕面，可以清屏。清屏的实质是向显示缓冲区写入清屏的数据。通常的办法是，对于字符模式向显示缓冲区写入空格的 ASSII 码 20H；对于图形，模式则用全零充满显示缓冲区。需要注意的是，彩色卡和单色卡所使用的显示缓冲区，其起始地址和容量大小都是不相同

的。下面一段程序说明怎样清屏。

```
MOV     CX,16384
MOV     AX,0B800H
CLD
MOV     ES,AX
XOR     DI,DI
XOR     AX,AX
REPZ    STOSB
```

三、单色卡仿真 CGA 视频标准的图形功能的实现

单色卡上有一个 I/O 端口 3BFH，其作用是打开单色卡，使用方法是：

```
MOV     AL,03
MOV     DX,3BFH
OUT     DX,AL
```

在对 CRTC 6845 进行初始化编程前应先用上述指令打开单色卡。

按照 IBM/PC 机的内存分配，A000：0000～B000：FFFF 共 128KB 内存分配给显示缓冲区。配置彩色卡和中分辨率彩色显示器时，显示缓冲区地址为 B800：0000～B800：3FFF 共 16KB；配置单色卡和高分辨率单色显示器时，显示缓冲区地址为 B000：0000～B000：0FFF 共 4KB。其余内存地址预留不用。

目前，出于性能/价格比、环境保护、应用需求等诸多因素，市场上出现了许多以 8088 为 CPU、配置单色卡和高分辨单色显示器的准 16 位微型计算机。与 IBM/PC 推出的初期相比，其单色卡有了很大改进，主要表现在显示缓冲区的容量增大至 64KB 或更大。这是单色卡仿真 CGA 视频标准的图形功能的物质基础。就安装 64KB 的单色卡而言，显示缓冲区的地址为 B000：0000～B0000：FFFF。在使用上，这 64KB 显示缓冲区分成两页，每页 32KB，第一页起始地址为 B000：0000，第二页起始地址为 B800：0000。这种分页方法为单色卡仿真 CGA 视频标准的图形功能提供了方便，即只要使用第二页就行了。具体方法是将端口 3B8H 的最高位置 1。对于 64KB 显示缓冲区，只使用第二页的前 16KB 就可以了。

根据上述分析我们可以设想，如果把 CGA 视频标准的 06 号模式所需的初始化参数写入 CRTC6845 的数据寄存器 R0～R11，使用 B800：0000 开始的 16KB 内存空间为显示缓冲区，借助于 1NT10H 的 00 号功能调用，便可以实现单色卡仿真 CGA 视频标准的图形功能。

下面的程序便是根据上述思想设计的一个既简单又完整的仿真程序。程序用 8088 汇编语言写成，经汇编、连接后便可生成可执行的 .EXE 文件。如果用 DOS 的实用程序 EXE2B1N.EXE 将其转换成 .COM 文件，则执行速度更快。

```
1: CSG          SEGMENT
2:              ASSUME    CS:CSG,DS:CSG
3:              ASSUME    ES:CSG,SS:CSG
4:              ORG       0100H
5: START        PROC      FAR
6:              PUSH      DS
7:              XOR       AX,AX
8:              PUSH      AX
9: ;保存硬件设备配置
10:             MOV       AX,0040H
11:             MOV       ES,AX
12:             MOV       AL,0EDH
13:             AND       ES:[0001H],AL
14:;打开单色卡
15:             MOV       AL,03
16:             MOV       DX,3BFH
17:             OUT       DX,AL
18:;初始化      CRTC6845
19:;数据寄存器
20:*            PUSH      CS
21:             POP       DS
22:             LEA       SI,CUANSHU
23:             MOV       DX,3B4H
24:             MOV       CX,12
25:             XOR       AH,AH
26:LOOP1:       MOV       AL,AH
27:             OUT       DX,AL
28:             INC       DX
29:             LODSB
30:             OUT       DX,AL
31:             INC       AH
32:             DEC       DX
33:             LOOP      LOOP1
34:;显示缓冲区置
35:;全 00,清屏
36:             MOV       CX,16384
37:             MOV       AX,0B800H
38:             CLD
39:             MOV       ES,AX
40:             XOR       DI,DI
41:             XOR       AX,AX
42:             REPZ      STOSB
43:;初始化控制寄存
44:;器,允许显示
45:             MOV       DX,3B8H
46:             MOV       AL,02
47:             ADD       AL,88H
48:             OUT       DX,AL
49:;置 06 号模式
50:             MOV       AX,0006H
51:             INT       10H
52:EXIT:        RET
53:CUANSHU      DB        38H,28H,2DH
54:             DB        0AH,7FH,06H
55:             DB        64H,70H,02H
56:             DB        01H,06H,07H
57:START        ENDP
58:CSG          ENDS
59:             END       START
```

西文 Wordstar 文件在中文状态下的操作

浙江大学出版社　尤建忠

随着计算机信息资料的相互交流与应用，往往会遇到西文 Wordstar 状态下编辑的纯英文文件要在中文状态下加入中文内容这一类情况，这就需要对西文 Wordstar 状态下编辑的纯英文文件进行适当的处理，否则将难以达到目的。

西文 Wordstar 状态下编辑的纯英文文件在 DOS 状态下用 TYPE 命令在屏幕上显示，或在 CCDOS 状态下用 C－Wordstar 编辑，都会出现原文件中一些内容变成 IBM PC ASCII 码表中值大于 80H 的字符，这种情况主要出现在空格前的一个字符处。但用命令：

＞TYPE 文件名＞PRN＜CR＞

则会在打印机上打印出正确的内容。如不对原文件进行处理，则无法用 C－Wordstar 进行编辑。

用 Debug 程序装入文件，则会发现那些 ASCII 码值大于 80H 的字符均用不可显示字符（句点）表示，且为原码值的基础上加上 80H，即最高位（bit）为 1，对原文件进行处理，以便使它能在中文状态下编辑的方法很多，在此介绍两种。一种是专门对容量小于 64K 字节的文件进行简易处理的方法；另一种则是可以处理任意容量的文件的专用程序。它们的原理是相同的，即将 ASCII 码值的最高位改为 0。不过，在处理之前切记要处理的文件是纯英文文件，否则对一个含中文的文件进行处理，就会使中文内容受到破坏。

1. Debug 动态处理方法

先用命令：

＞Debug 文件名＜CR＞

装入文件，然后用命令：

－R＜CR＞

检查文件长度，即 BX，CX 寄存器值（高位在 BX，低位在 CX）。如 BX 不等于零，则表示文件长度大于 64K，故需将文件分成几个小文件（用 CWS 的 ctrl－KW 命令）。在 CX 值上加上 100H，即可找到文件末尾后的内存地址。一般可在文件结束符 1AH 后找个地址。举例，如 CX＝00F0H，BX＝0000H，则可找到该内存地址为 01F0H，我们将这个地址选为调试程序的首地址。接着打入汇编命令：

－A 首地址＜CR＞

```
××××:首地址      push bx
××××:××××      push cx
××××:××××      mov bx,100
××××:××××      mov cx,首地址－1
××××:××××      sub cx,bx
××××:循环地址    mov al,[bx]
××××:××××      shl al,1
××××:××××      shr al,1
××××:××××      mov [bx],al
××××:××××      inc bx
××××:××××      loop 循环地址
××××:××××      pop cx
××××:××××      pop bx
××××:末地址      ＜CR＞
```

上面程序中的地址要根据实际情况填入，然后用反汇编命令：

－U 首地址　末地址＜CR＞

进行反汇编。最后用命令：

－G＝首地址　末地址＜CR＞

－W＜CR＞

－Q＜CR＞

即可处理完毕，并存盘回到 DOS 状态下。此时可用 TYPE 命令显示或在中文状态下用 C－Wordstar 对文件进行处理。

2. 软件方法

本处理软件主要是在一个复制文件的程序上稍加修改，即加入 MASK 子程序而编制成的。而 MASK 子程序，就是用来屏蔽 ASCII 码的最高位的。

在编制该程序时，首先用 Wordstar（中西文均可）的 N 命令或 Edlin 输入源程序，然后用 MASM 汇编和 LINK 连接而生成 CONVERT.EXE 文件。使用时，仅需打入：

＞convert＜CR＞

屏幕上即会提示打入需处理的文件名和处理好后的文件名（不能为同一文件名）即可。下面是该软件的源程序 CONVERT.ASM：

```
stack     segmentpara stack'stack'
          db 100 dup(9)
stack     ends
data      segment
stile     db64
          db?
          db64dup(' ')
dfile     db64
          db?
          db64dup(' ')
ask1      db0ah,0dh,'Please input source'
          db'file name:','$'
ask2      db 0ah,0dh,'please input'
          db'destination file name:','$'
err1      db0ah,0dh,'createrror','$'
err2      db0ah,0dh,'open err','$'
err3      db0ah,0dh,'read err','$'
err4      db0ah,0dh,'write err','$'
err5      db0ah,0dh,'close source'
          db'file error','$'
```

```
err6        db 0ah,0dh,'close destination'
            db 'file error',' $ '
bufr        dw?
data        ends
code        sgment
            assume cs:code,ds:data,es:data
start       proc far
            push ds
            sub ax,ax
            push ax
            mov ax,data
            mov ds,ax
            mov es,ax
            lea dx,ask1
            call disp
            lea dx,sfile
            call (inpt)
            mov cl,sfile+1
            xor ch,ch
            mov si,cx
            mov [si+sfile+2],0
            lea dx,ask2
            call disp
            lea dx,dfile
            call inpt
            mov cl,dfile+1
            xor ch,ch
            mov si,cx
            mov[si+dfile+2],0
            call settle
            ret
start       endp
disp        proc near
            mov ah,9
            int 21h
            ret
disp        endp
inpt        proc near
            mov ah,0ah
            int21h
            ret
inpt        endp
mask        proc near
            mov di,offset sfile+2
            push cx
again:      mov al,[di]
            shl al,1
            shr al,1
            mov[di],al
            inc di
            loop again
            pop cx
            ret
mask        endp
settle      proc near
            mov ah,3ch
            lea dx,dfile+2
            mov cx,0020h
            int21h
            lea dx,err1
            mov bx,ax
            jc err
            mov bufr,ax
            mov ah,3dh
            mov al,0
            leadx,sfile+2
            int21h
            leadx,err2
            mov bx,ax
            jc err
deal        mov cx,0020h
            mov ah,3fh
            leadx,sfile+2
            int21h
            lea dx,err3
            jc err
            or ax,ax
            je ok
            call mask
            lea dx,sfile+2
            mov ah,40h
            xchg bufr,bx
            int21h
            lea dx,err4
            jc err
            xchg bufr,bx
            jmp deal
ok:         mov ah,3eh
            int21h
            lea dx,err5
            jc err
            xchg bufr,bx
            mov ah,3eh
            int21h
            lea dx,err6
            jc err
            ret
err:        movah,3eh
            int21h
            xchg bufr,bx
            mov ah,3eh
            int21h
            call disp
            ret
settle      endp
code        ends
            end start
```

试论磁盘文件的最佳存储方法

长沙水利电力师范学院计算中心　陈宏明

在MIS管理应用中，最常用的语言是dBASE，设计程序的方法是采用模块化设计方法，即“从上到下、从整到零、从粗到细、分而治之”，将管理系统分成很多小模块，分别编制和调试，并且相应的设计一些数据库、索引文件、内存变量文件、格式文件等，使得管理系统由许多小文件组成。特别是软件开发者在设计程序时，不注意优化数据库的设计，程序没有合并的情况下，一个大的管理系统，可能系统实际总长度有几M字节，但文件数可能有几千个。这样当管理系统推广应用时，经常会遇到这样的问题，虽然整个系统仅占几M空间，但在带有20M硬盘的机器上，当磁盘自由空间大于该系统所占空间（例如十几M）的情况下，有的计算机能够运行，有的却不能运行。笔者在推广财务管理系统时，就遇到过此类问题。为什么会出现这种现象，关键是在磁盘文件的存储方法，本文给出磁盘文件的最佳存储方法。

众所周知，磁盘在使用前必须先进行格式化，硬盘还必须先分区，然后才能进行格式化。分区和格式化是在磁盘上首先建立面（柱）、道、族、扇区的储存结构，以便文件存储和管理。在DOS下每扇区固定为512个字节，但多少个扇区组成一个族，这在不同容量的硬盘不同版本的DOS下是不同的。文件存储在格式化好的扇区中，但文件分配不是按扇区分配的，也不是按实际长度分配的，而是按族分配的，即族是文件分配的最小单位。为了管理文件，DOS在磁盘的0道建立存储文件名、文件长度、性质的目录区和文件在磁盘的实际位置的文件分配表FAT。FAT中存放文件在磁盘中的起始族号和族的数目。据此可以得知文件分配表FAT决定着磁盘为每个文件分配的实际空间，族的大小决定着磁盘空间的利用率。在DOS2.XX系列中，FAT用12位表示，即占1.5个字节，因此与之相应的磁盘族总数不能超过4096(2^{12})，当磁盘容量大时，只能提高每族所包含的扇区数。在DOS3.XX系列中，FAT用16位表示，即占2个字节，因此磁盘族的上限为65536(2^{16})。这样对于不同的DOS版本，不同的磁盘，族所包含的扇区数目也是不同的。请见下表：

机型	DOS版本	磁盘容量	每族扇区数	磁盘族总数	磁盘扇区总数
IBMPC/XT	2.10	10M	8	2586	20739
IBMPC/XT	3.30	10M	8	2586	20739
IBMPC/XT	2.10	20M	16	2591	41547
IBMPC/XT	3.30	20M	4	10354	41531
GW0520CH	2.10	20M	16	2590	41531
GW0520CH	3.30	20M	4	10354	41531
AST286	3.30	40M	4	16326	65467
AST 386	3.30	40M	4	16326	65467
IBMPC/XT	2.10	360K	2	354	720
IBMPC/XT	3.30	360K	2	354	720
GW0520CH	2.10	360K	2	354	720
GW0520CH	3.30	360K	2	354	720

根据以上表可以得知，对于软盘无论用什么版本的DOS格式化，无论在什么样的机型下，每个族所含扇区数是一定的。例如360K软盘均为2个扇区，1024个字节，即当一个文件存储在磁盘上，无论它实际多么小，那怕它只有几个字节，软盘也至少给它分配1024个字节。对于硬盘，每个族所包含的扇区数，与计算机的机型无关，与DOS的版本有关，与硬盘的容量有关。

根据上表及分析，当带有20M硬盘的计算机用DOS2.XX格式化时，每族包含16个扇区，8K个字节，即任意一个文件如需存到磁盘上，无论它实际多么小，在磁盘上至少占8096个字节。而用DOS3.XX格式化时，每族仅含4个扇区，2K个字节。那么例如某管理系统有2000个文件，平均每个文件实际长度为1024个字节，系统总长度为2M字节，存放在软盘上大约为6张。但拷贝到硬盘时，如硬盘由DOS 2.XX格式化时，由于每个文件虽然仅有1K，硬盘也必须分配给它8K字节，即该系统在硬盘上占16M字节；如硬盘由DOS3.XX格式化时，硬盘给每个文件分配2K字节，整个系统仅占4M字节，少12M字节。所以磁盘文件的优化存储方法是，在超过10M的硬盘上如DOS为2.XX版本，首先提高DOS版本，方法是：用DOS 3.30从A盘启动后，先删除硬盘上原有DOS分区，再重新建立DOS分区，然后将硬盘带系统的格式化，这样可以将每族扇区数由16个降为4个。注意：必须先删除硬盘上已有分区，然后重新建立分区，否则DOS版本虽然为3.30，但每族扇区数仍为16。其次，根据以上分析，在管理系统中，应尽量合并小文件。例如，用FOXPLUS可以将系统的所有PRG文件结合在一个文件中，这样可以大大提高硬盘的利用率。另外文件的长度也应尽量为2024的倍数，这样也可充分利用磁盘空间。不过事物都是两方面的，在DOS 3.XX版本下，族包含的扇区数少，但读写磁盘的速度也因之而下降，在286机、386机，由于机器主频速度快，感觉不出，在

（下转36页）

断电保护打印程序

河南禹州红星电瓷厂　郭河伟

微机应用于事务处理，提供给用户丰富的打印输出功能是很重要的。当打印一份包含较多(几百甚至上千)记录的报表时，如果在打印过程中突然断电，那么等来电后再次开机打印时，不是接着上次打印，而是从头开始打印。这样，不仅造成纸张浪费，而且降低工作效率。为了克服这种现象，我们结合实际，设计了一种具有断电保护功能的打印程序。应用该程序，断电后再次开机时，就能接着上次继续打印。本程序应用 dBASE Ⅲ 编制而成，在 IBM－PC/XT 主机，M2024 打印机上调试通过。

设数据库为 xj. dbf，包含60个记录，结构如下：

场变量名	场变量类型	场宽	小数位
S1(学号)	数字型	4	0
S2(年龄)	数字型	2	0

要求程序能实现分页打印，自第二页开始，每页控制打印21个记录。且能实现断电保护功能。

先建立变量文件 dym. mem，其中变量 e 为记录号，初值为1。每打印完一页报表，e 就变化一次。e 的取值与记录号的关系见下表：

e 取值	页数	记录号
1	1	1～20
21	2	21～41
42	3	42～62
63	4	63～83
84	5	84～104
……	……	……

自打印第一页后，e 的取值均为21的整数倍，即 e＝21，42，63，84……

打印主程序 dy. prg，打印子程序 dy0. prg 见后。

关于程序执行情况的说明：

1. 当程序运行第一次打印报表时，预先建立的变量文件 dym. mem 中 e＝1，此时程序提示“正在打印……”。当打印完第20个记录时，程序又提示“调整打印纸，按任一键继续打印……”，然后接着上次打印。以后，每页控制打印21个记录，直至循环打印结束。全部记录打印结束时，变量 e 被重新赋值为1，这样下次打印仍从头开始。

2. 当程序中断(如突然断电或按 ESC 键强制中断)后重新运行时的打印情况：

①若第一页未打印完就中断，则再次运行程序仍从第一页开始打印，屏幕不提示“接着上次打印吗?”。

②若已打印若干页之后，再次运行程序时屏幕就提示“接着上次打印吗?”，请用户选择。用户可选择接着上一次继续打印，也可选择退出，从头开始打印。

```
. use xj
. display structure
数据库结构 — 数据库：A：xj. dbf
数据库中的数据记录个数：60
数据库的最后更新日期：01/01/80
字段      字段名      类型        宽度      小数
1         S1          数字型      4
2         S2          数字型      2
* * 总计   * *                    7

. type dy0. prg
set devi to print
set print on
if e=1. or. e/21=int(e/21)
    @ 4,10 say'
    @ 5,10 say'
    @ 6,10 say'
    @ 7,10 say'
endif
@ prow()+1,10 say ' |' +str(s1,4)+' |' +str(s2,2)+' |'
@ prow()+1,10 say ' |----|--|'
set devi to scre
set print off
retu

. type dy. prg
a=' 1'
do while a=' 1'
rest from dym
use xj
do while . not. eof()
  f=' 1'
  if e/21=int(e/21)
    acce ' 接着上次打印吗(1. 是  0. 否)：' to c
    if c=' 1'
      f=' 2'
    else
      e=1
      save to dym
    endif
  endif
  do while . not. eof()
    if e/21<>int(e/21). or. f=' 2'
      if f=' 2'
        go el
      endif
      if e=1. or. e/21=int(e/21)
```

学　年
号　龄

```
        clear
        @ 5,10 say '正 在 打 印，请 稍 等……'
        endif
        do dy0
        f='1'
        e=e+1
        skip
      else
        set devi to print
        set print on
        @ 0,0 say '      '
        set print off
        set devi to scre
        clear
        ? chr(7)+chr(7)
        @ 5,10 say '请 调 整 打 印 纸!'
        f='2'
        el=e
        rele c
        save to dym
        wait '按 任 一 键 继 续…'
      endif
    enddo
enddo
e=1
set devi to print
set print on
@ 0,0 say '      '
set devi to scre
set print off
save to dym
? chr(7)
acce '还打印吗(1.打印  0.退出):' to a
use
enddo
retu
```

将三角函数引进 DBASE 系统

广东汕头 金山中学 何管略

我们知道，在 dbase Ⅲ 系统中配有 37个内部函数，大致可以分为五种类型：数学函数5个，字符串操作函数7个，类型转换函数4个，测试函数11个和日期函数10个。其中5个数学函数是：EXP(指数函数)，INT(取整函数)，LOG(自然对数函数)，ROUND(舍入函数)和SQRT(平方根函数)。这里没有三角函数。

有时，我们的确需要用到三角函数。例如：统计学生的考试成绩，由于各科试题的难易程度不同，原始分的分布状态，一般说来是随机的；据此作出的比较和分析，往往不会令人满意。如果将随机分布的原始分，转换为按正态分布的标准分，统计结果，就会比较符合实际。这个转换过程，由于数据众多，工作量很大，最好运行 dbase 系统，建立数据库，采用三角函数，编程解决。

如何将三角函数引进 dbase 系统呢？按照三角函数的级数展开公式，编程计算，是有可能实现的。以余弦函数为例，它的级数展开式为：

$$\cos(x)=1-x^2/2!+x^4/4!-x^6/6!+\cdots$$

这列级数的特点是，相邻两项为一组；越靠左边(前面)的组，对计算结果的影响越大；越靠右边(后面)的组，对计算结果的影响越小。

下面介绍由已知角度值，求弧度值，余弦值以及绘制直方图的编程思路。首先，建立一个数据库COSB，含角度、弧度、余弦、图象四个字段，其中图象字段为字符型，余为数值型。从键盘逐个输入角度值。例如从90°到－90°，每隔15°输入一个角度。将角度值(d)转换为弧度值(x)时，用 x=d * π/180。按照单精度数值的要求，公式中的圆周率，取 π=3.141593，准确到小数点后六位。

为使求得的余弦值，也相应地准确到小数点后六位，我们累计了级数展开式的前面5组，即计算到 x^18/18!的数据项。所用算式，显然相当繁复。由于程序多次使用中间变量(w、y)，尽量减少重复运算；实践表明，运行速度，还是不慢。

将余弦值放大10倍，并四舍五入取整；然后按此整数的多少，用 * * 标记作出直方图，就可形象地显示余弦值随其角度而变化的规律。详细情况，请参阅程序清单及其数据库的具体结构。

```
· type cos.prg
set talk off
use COSB
brow
go 1
set head off
p = 3.141593 / 180
do while .not. eof()
    *将 角 度 化 弧 度
    x = 角度 * p
    repl 弧度 with x
    * 计 算 余 弦 函 数
    w = x * x
    y = 1 * w / 1 / 2
    z = 1 - y
    y = y * w / 3 / 4
    z = z + y
    y = y * w / 5 / 6
    z = z - y
    y = y * w / 7 / 8
    z = z + y
    y = y * w / 9 / 10
```

```
z = z - y
    y = y * w /11 /12
    z = z + y
    y = y * w /13 /14
    z = z - y
    y = y * w /15 /16
    z = z + y
    y = y * w /17 /18
    z = z - y
    repl 余弦 with z
    * 绘制余弦函数直方图
    repl 图象 with space(10)
    if z > 0
      z = int( z * 10 + .5)
      j = 1
      do while j <= z
      repl 图象 with trim(图象) + "*"
        j = j + 1
      enddo j <= z
    endif
    disp off
    skip
    enddo .not. eof()
    set head on
    ? chr(7)
    go 1
    brow
    set talk on
    return
    •
    • list off
```

角度	弧度	余弦	图象
90	1.570796	0.000000	
75	1.308997	0.258819	* * *
60	1.047198	0.500000	* * * * *
45	0.785398	0.707107	* * * * * * *
30	0.523599	0.866025	* * * * * * * * *
15	0.261799	0.965926	* * * * * * * * * *
0	0.000000	1.000000	* * * * * * * * * *
-15	-0.261799	0.965926	* * * * * * * * * *
-30	-0.523599	0.866025	* * * * * * * * *
-45	-0.785398	0.707107	* * * * * * *
-60	-1.047198	0.500000	* * * * *
-75	-1.308997	0.258819	* * *
-90	-1.570796	0.000000	

经验与技巧两则

珠海市拱北海关　吴仲文

一、如何进入带密码的子目录

计算机用户为存放私人文件而在硬盘建立子目录。为保护数据起见，通常在子目录名中置入密码，例如区位码的空位、半个汉字以及按 ALT 键＋小键盘数字键所得的 ASCII 码等。假如遗忘密码或试图访问他人目录，而又没有 PCTOOLS 等工具软件，要进入这些子目录仍是感到困难的。现介绍一种简单的方法，可轻而易举地进入带密码的子目录。例如要进入子目录 SUBD□（□表示区位码的空位等），步骤如下：

1. 在根目录下键入命令 DIR SUBD *. ＞ML. BAT，得到含有该子目录名的文件 ML. BAT；

2. 以字处理软件（如 WS）编辑 ML. BAT，令该文件的内容为“CD\SUBD□”；

3. 执行批处理文件 ML. BAT，即可进入该子目录。

其关键在于应用了 DOS 的重定向功能，将 DIR 命令列出的文件目录清单存放到 ML. BAT 中。同理，也可以利用本法对文件名带有密码的磁盘文件进行操作。本例说明，灵活运用 DOS 命令，对解决一些问题可得到事半功倍的效果。

二、巧获汉字的 HEX 机内码及区位码

无论是 ASCII 字符或汉字字符，均以16进制数(HEX)的形式储存于计算机内，ASCII 字符占用一个字节，而汉字则占用两个字节，且两字节二进制数的最高位均为1，以区别于 ASCII 字符。如何方便地获得汉字的机内码是许多计算机用户所关心的问题；实际上，就 FOXBASE 而言，利用 ASC()函数就可分别求得汉字第一及第二字节机内码的10进制数，再转化为16进制数即可。又，通过对照汉字的机内码(10进制)及其相应的区位码，不难看出，汉字第一字节的机内码减去160，即相应区位码的区号，第二字节减去160，即其位号，因此，知道汉字的机内码，即可求得其区位码，反之亦然。现以一 FOXBASE 示范程序说明如何通过汉字字符求得其 HEX 机内码及相应的区位码。

```
* 示范程序
clear
set talk off
hex= '0123456789ABCDEF'
hz=space(2)
@ 12,12 say '请输入汉字：' get hz
read
```

（下转第45页）

中国计算机学会 1990 年度

全国青少年信息学(计算机)通信赛试题

编者按：中国计算机学会为普及青少年计算机知识教育，每年举行一次全国性青少年计算机程序设计竞赛。90 年度采用通信赛方式进行并于 11 月 25 日结束。比赛将决出一等奖三名、二等奖六名、三等奖十二名，获奖选手将于 91 年 1 月到北京参加复赛。

此次比赛为 1991 年度我国参加国际第三届奥林匹克信息学(计算机)竞赛选手的选拔创造了一定的条件。

试　题

由 27 个小方块组成一个类似魔方的立方体，每一个小方块有一个标志。图 1 示出了该立方体处于目标状态时小方块之间的邻接关系。图中的字符为小方块的标志。

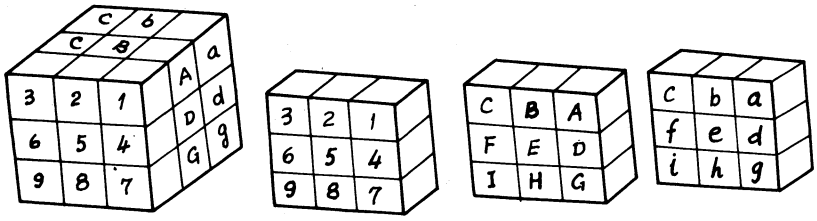


图 1　立方体的目标状态

为了给立方体的小方块定位，设置了如图 2 所示的一个三维坐标系，空间有 27 个点分别编号为 1、2…27，它们在空间是不动的。X、Y、Z 轴的交点的编号为 14。将标志为 E 的小方块的中心放在编号为 14 的 X、Y、Z 轴的交点上，其它小方块的中心分别落 在除 14 点以外的其他 26 个空间上。

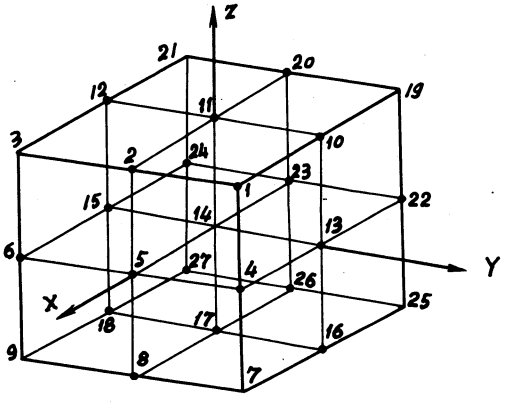


图 2　含 27 个点的空间坐标系

在确定了小方块所在的空间位置之后，就可以按照空间编号的顺序来描述这个立方体了。比如图 1 的目标状态可以用如下的一个字符串来表示：

1 2 3 4 5 6 7 8 9 A B C D E F G H I a b c d e f g h i

这个字符串中的字符是小方块的标志，字符排列顺序就是小方块在空间的排列顺序。也就是说标志为 1～9 的小方块分别落在空间位置 1～9 上，标志为 A～I 的小方块分别落在空间位置 10～18 上，标志为 a～i 的小方块分别落在空间位置 19～27 上。

对于图 2，我们用眼睛对着 X 轴看，叫“正视”；对着 Y 轴看，叫“侧视”；对着 Z 轴看，叫“俯视”。当“正视”时，X 轴通过空间三个点 23、14、5。以这三个点为中心构成垂直于 X 轴的三“片”。

1、2、3、4、5、6、7、8、9 为片 1 的空间位置；

10、11、12、13、14、15、16、17、18 为片 2 的空间位置；
19、20、21、22、23、24、25、26、27 为片 3 的空间位置。

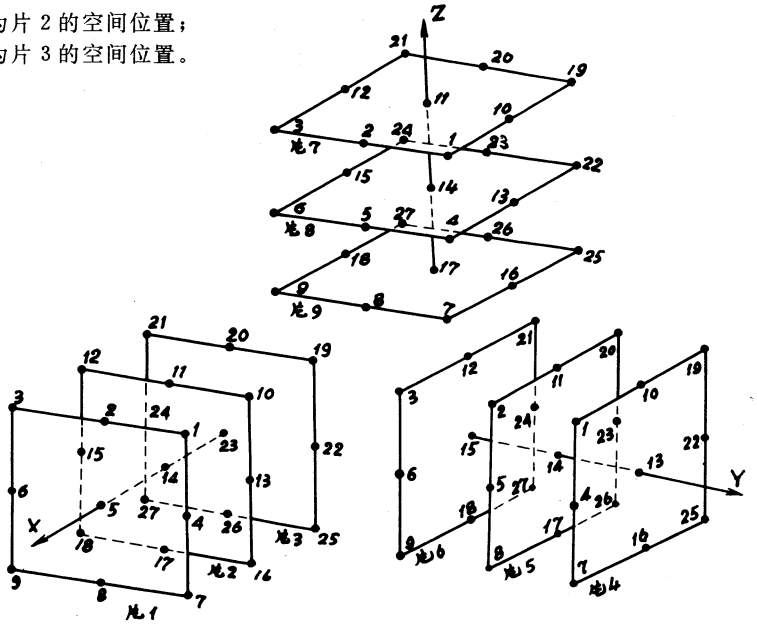


图 3　片 1— 片 9 空间位置图

每个片都由 9 个落在这些空间点上的小方块组成。这 9 个小方块组成的片以 X 为轴线，可顺时针旋转，或逆时针旋转，每转一步为 90 度。

当“侧视”时，Y 轴通过空间 3 个点：15、14、13 构成片 4、片 5、片 6。

当“俯视”时，Z 轴通过空间 3 个点：17、14、11 构成片 7、片 8、片 9。

片 1—片 9 的空间位置见图 3。

从图 3 可见片 1—片 3 可绕 X 轴转动；片 4—片 6 可绕 Y 轴转动；片 7—片 9 绕 Z 轴转动。片是以空间位置来划分的。小方块转到该空间位置就可归属于这一片。例如图 4，组成片 4 的小方块为 1、4、7、A、D、G、a、d、g，当片 4 绕 Y 轴逆时针转 90 度后，片 4 所包含的小方块空间位置有了变化，影响到片 1、片 2、片 3、片 7、片 8 和片 9 中小方块的组成。比如片 1，原来的成员是小方块 1、2、3、4、5、6、7、8 和 9，当片 4 绕 Y 轴逆针转过 90 度之后，成员变成 a、2、3、A、5、6、1、8 和 9。这时再让片 1 绕 X 轴逆时针转 90 度，转过之后片 1 的成员不变但成员所处空间位置有所变化。这就引起片 4、片 5、片 6、片 7、片 8、片 9 的成员有所变化。在图 4 中，立方体的三种状态可由下列字符串序列来描述：

变化前的起始状态为

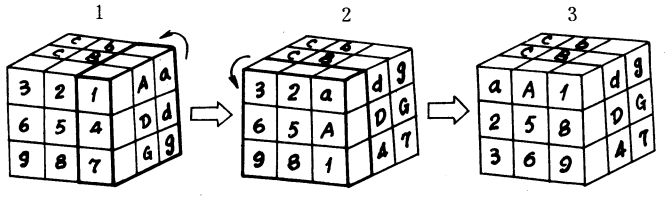
1 2 3 4 5 6 7 8 9 A B C D E F G H I a b c d e f g h i

片 4 绕 Y 轴逆时针转过 90 度后的状态为

a 2 3 A 5 6 1 8 9 d B C D E F 4 H I g b c G e f 7 h i

片 1 绕 X 轴逆时针转过 90 度后的状态为

1 A a 8 5 2 9 6 3 d B C D E F 4 H I g b c G e f 7 h i



Y 轴上的片 4 逆时针转　　X 轴上的片 1 逆时针转　　两次转动后的状态

图 4 立方体状态变化例图

请你在看懂上述的内容之后编写一个程序并上机通过。程序要求如下：

1. 含 27 个小方块的立方体用字符串来描述。目标状态为

1 2 3 4 5 6 7 8 9 A B C D E F G H I a b c d e f g h i

2. 从键盘输入初始状态。限于内存容量初始状态距目标状态的步数不用太多，三步以内即可。

3. 程序应能判断输入是否有误。

4. 寻找一种旋转方案，使立方体由初始状态达到目标状态。输出结果应按如下格式：

start :1 A a 8 5 2 9 6 3 d B C D E F 4 H I g b c G e f 7 h i

step 1:a 2 3 A 5 6 1 8 9 d B C D E F 4 H I g b c G e f 7 h i

step 2: 1 2 3 4 5 6 7 8 9 A B C D E F G H I a b c d e f g h i

end.

这个结果就是图 4 的例子，但是要倒过来，以③做为初始状态，②为第一步旋转后的结果，①是第二步旋转之后达到的目标状态。

5. 在 4 的基础上，寻找最佳方案，使转动步数最少。

6. 写出必要的文档材料，包括：

(1)打印程序清单；

(2)对程序中的变量、函数作必要的说明；

(3)解释编程思路；

(4)画出程序框图；

(5)讨论程序运行结果；

(6)其他。

此 6 项内容须在 4 小时内完成。

测 试 数 据

说明：该测试数据仅供评委检测选手程序时使用，不得事先向选手公布：

测试项目与相应得分：

1. 输入 1 2 3 4 5 6 7 8 9 E A B C F D G H I a b c d e f g h i (10 分)
2. 输入 1 2 3 4 5 6 7 8 9 A B C D E F G K J a b c d e f g h i (10 分)
3. 输入 1 2 c 4 5 C 7 8 3 A B f D E F G H 6 a b i d e I g h 9 (20 分)
4. 输入 1 2 c 4 5C 7 8 3 A B f D E F G H 6 i I 9 b e h a d g (20 分)
5. 输入 1 8 3 4 H 6 7 h 9A 5 C D E F G el a 2 c d B f g b i (20 分)
6. 输入 1 8 3 4 H 6 7 h 9 A 5 C D E F G e I c f i 2 B b a d g (20 分)

六项测 试共 100 分。文档 20 分。总分为 120 分。

(上接 32 页)

```
1505H  E5 2B       MOV    A,2BH      ;A←EP 代码
1507H  B4 42 24    CJNE   A, # 42H,  ;是 8751 加密则转
                                152EH
150AH  71 D0       ACALL  13D0H      ;送 1ms 编程脉冲
150CH  05 F0       INC    B          ;X←X+1
150EH  75 90 FF    MOV    P1, # FFH  ;置 P1 为输入方式
1511H  D1 59       ACALL  1659H      ;A←读 EP 数据
1513H  B5 01 OD    CJNE   A,01H,     ;与源数据不同则转
                                1523H
1516H  91 4F       ACALL  144FH      ;相同，加 6Xms 脉
                                冲
1518H  E9          MOV    A,R1       ;A←数据
1519H  91 60       ACALL  1460H      ;显示一字节
151BH  71 35       ACALL  1335H      ;地址结束?
151DH  60 0F       JZ     152EH      ;是，则转结束
151FH  71 66       ACALL  1366H      ;否，则地址加一
1521H  A1 0A       AJMP   150AH      ;进行下一字节编程
1523H  F9          MOV    R1,A       ;暂存出错 EP 数据
1524H  E5 F0       MOV    A,B        ;A←X
1526H  B4 19 E1    CJNE   A, # 19H,  ;X≠25，则转再加
                                150AH
1529H  E9          MOV    A,R1       X=25，A←出错数
                                据
152AH  91 60       ACALL  1460H      ;显示出错
152CH  80 FE       SJMP   152CH      ;持续显示
152EH  02 00 00    LJMP   0000H      ;结束，返监控
```

APPLE Ⅱ 高分辨率绘图与文字编辑技术

同济大学　邹培国

本文介绍在 APPLE Ⅱ 高分辨率图形区中直接由键盘控制实现绘图与文字编辑的方法，着重介绍图形与字符混合的基本原理，图形库的构造方法，编程技巧和实际应用。

由于 APPLESOFT 设有提供在高分辨率图形区显示字符的语句，因而给图表、曲线的数字标注与文字说明带来了不便。有些计算机杂志介绍了几种在高分辨图形区显示字符的程序，但大多采用机器语言或较复杂的绘图程序，对一般的用户仍有所不便。

本文采用 APPLESOFT 的图形库技术和简单的绘图语句，来实现高分辨率图形与字符混合的功能。

一、高分辨率图形与字符混合原理

利用高分辨率图形数据存储区与屏幕的对应关系，我们可以使用机器语言或 POKE 语句，将所需图形直接显示在屏幕上。许多游戏程序和固定图形的显示等都是通过这种方式进行的。但这种方式技巧性强，编程复杂，不够灵活，难以推广使用。下面我们介绍怎样利用 APPLESOFT 提供的绘图语句 DRAW，XDRAW 和图形库技术来编制 BASIC 程序，实现绘图与字符显示和编辑功能。

为了实现上述功能，我们事先将常用的字符和图形点阵编制成绘图向量存入图形库。在 APPLESOFT 中，每个绘图向量用三位二进制数表示，意义如下表所示。

绘图向量	图　示	意　　义
000	↑	上移不画(仅移动不画点)
001	→	右移不画
010	↓	下移不画
011	←	左移不画
100	↑	上移画(画点且移动)
101	→	右移画
110	↓	下移画
111	←	左移画

利用上面八种向量，就可以设计出所需的图形来。譬如，将字符 2 设计成绘图向量，如图 1 所示。在内存中，一个字节可以表示三个绘图向量，分别用 A、B、C 来表示，具体规定如图 2 所示。

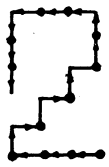
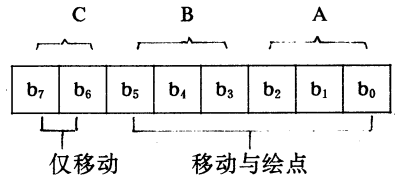


图 1　“2”的绘图向量表示



(C＝00 时 C 不动作，CB＝00000 时 C、B 均不动作，CBA＝$00 表示向量到此结束)

图 2　绘图向量的动作关系

由此可见，C 只能表示移动→、↓、←，不能表示↑；C＝00 时，B 也不能表示↑；而当 C、B、A 均为 0 时，表示一个图形的绘图向量到此结束。绘图向量的动作次序为：A→B→C。

有了绘图向量，就可以设计图形库了。图形库就是内存中的一个二进制数据表，格式如图 3 所示。

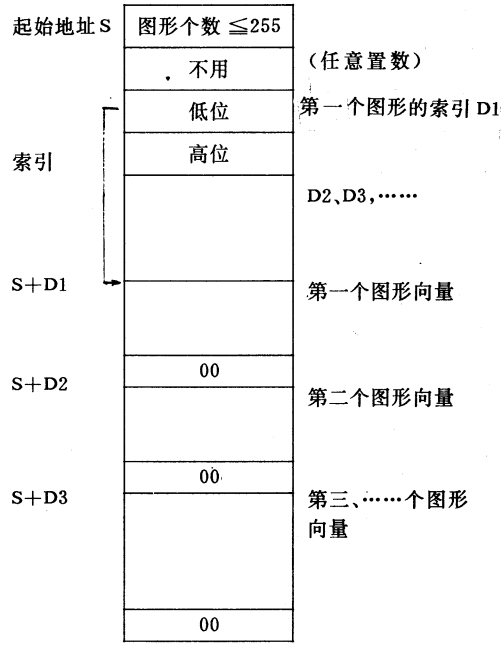


图 3　图形库格式

造图形库的步骤如下：

(1)在方格纸上画出所需图形；

(2)设计绘图向量，并转换成 16 进制数；

(3)用 CALL-151 进入监控状态，按图 3 的格式键入图形库(也可用 POKE 语句)；

(4)用 DOS 命令将图形库存入(BSAVE)磁盘；

(5)以后使用时，先把图形库装入(BLOAD)内存，并在零页区的 $E8，$E9 两个单元设置图形库的起始地址 S。

图形库可以放在内存的任何一个合适的区域，但为了避免内存冲突，一般将图形库存放在$6000以上的单元中。

以图1的字符2为例，设计绘图向量如下：

序　号	C　B　A	C　B　A	十六进制
1		00111111	3F
2	→	00110111	67
3	→	00001100	0C
4	→	00001100	0C
5	←	11100100	E4
6	↓	10111111	BF
7		00110110	36
8	（结束）	00000000	00

如果该图形库从$6000开始存放，假设字符2的序号为19，则该图形库在内存中的形式为：

开始地址：$6000:43——图形个数＝67
$6001:00——不用（任意置数）
$6026:0E　字符2的索引
$6027:01　（D19＝010E）
.....
$610E:3F　字符2的图形向量
$610F:67
.....
$6114:36
$6115:00
.....

图形库输入内存后，取名CHAR存入磁盘：

BSAVE CHAR,A$6000,L$2BF

这样，在以后应用中，装入CHAR，并在$E8，$E9单元设置图形库起始地址$6000即可。程序如下：

```
10 PRINT CHT$(4);"BLOAD  CHAR"
20 POKE 232,0:POKE233,96
```

图形库一旦建立，就可以用DRAW，XDRAW语句调用库中图形，从而实现在高分辨率图形区绘图与显示字符功能。下面提供两个实用程序以说明绘图程序的编制和应用。

二、实　例

[例1]绘图与文字编辑程序(DRAW-CHAR)

本程序具有与文本区类似的字符显示、光标移动、屏幕编辑与修改等功能。并且，可以由用户按控制键，实现以绘点线形式的画图功能。程序的编制、功能和使用简要说明如下：

(1)本图形库共有67个键盘字符(4×6点阵)，以其(ASCII码－31)为序号存放，第1个为空格，第63个为"^"。另设两个光标：第65号为字符光标　，第66号为绘图光标"."。

(2)用户按下列键盘即可直接在屏幕上绘图或编辑文字。

●功能键(控制键)

→，←，CTRL-A，CTRL-Z：光标右、左、上、下移动不画；

CTRL-K，CTRL-J，CTRL-I，CTRL-M：光标右、左、上、下移动画点；

CTRL-R：子程序返回(本程序表示结束)；

ESC：由光标□变成光标"."(绘图状态)，并使移动距离为一点间距(画图时使用)；

REPT：重复键意义不变，可结合使用。

●字符键：按下字符键，显示字符(点阵4×6)，光标自动右移一格，满行时自动换行。

●修改：将光标移回原处，再按所需字符键或空格键即可修改或擦除之。

●例子：按I，显示I□，若要改为J，再按←，J即可，显示J□；

按ESC，CTRL-K REPT，显示——；再按CTRL-J REPT即可擦除之。

(3)本程序的绘图部分以子程序形式给出，用户可方便地调用(GOSUB 6000)。

程序第1、5句调用图形库，并设置图形库起始地址。第10、20句设置闪动光标的位置(10、10)，间距(字符点阵4×6)，并在第一页(HGR)工作，白色显示。

第6000句实现光标闪动；第6010句读键盘，第6020句清键盘。第6022句绘点(XDRAW)，第6060句写字；第6057句实现空格显示。

第6025—6055、6465—6482句实现移动功能。

程序清单和图形库附后。

[例2]画正弦曲线并标注

本程序用绘线的方法，画出一条正弦曲线，并调用绘图子程序进行标注。程序和运行结果附后。

程序中，第10、20句调用图形库，设置图形库起始地址。第30、40句进入第二页绘图区(HGR2)，并选用蓝色画座标轴；第50－60句用白色画正弦曲线。第100－200句对正弦曲线标注数据和字符。

第5000－5060句是调用库中图形并显示的子程序，可作为通用子程序使用。本程序也可以调用例1的子程序(GOSUB6000)，直接由键盘控制在屏幕上标注。

图形库CHAR

```
6000- 43 00 02 00 88 00 8E 00
6008- 96 00 A4 00 B3 00 BD 00
6010- C7 00 CC 00 D2 00 D7 00
6018- E2 00 EA 00 EE 00 F3 00
6020- F6 00 FE 00 07 01 0E 01
6028- 16 01 20 01 2A 01 33 01
6030- 3E 01 46 01 50 01 5A 01
6038- 5E 01 63 01 6B 01 73 01
6040- 7C 01 84 01 84 01 8D 01
6048- 98 01 A0 01 A9 01 B2 01
6050- BA 01 C3 01 CC 01 D4 01
6058- DC 01 E5 01 EB 01 F7 01
```

```
6060- 01 02 0A 02 13 02 1D 02
6068- 27 02 31 02 39 02 42 02
6070- 4C 02 57 02 61 02 69 02
6078- 72 02 79 02 7F 02 86 02
6080- 8C 02 8F 02 AC 02 AE 02
6088- 1B 04 20 24 04 00 18 18
6090- 18 24 0D 36 06 00 FB 6C
6098- 0C 3F 3F 0C 0D 0C 3F 3F
60A0- 0C 6C 36 00 18 37 1F 28
60A8- 0C 0D 1C 3F 1C 0D 3C 0C
60B0- 15 2D 00 3C FE 63 0C 0C
60B8- 0C DF 37 2D 00 1F 27 0D
60C0- 05 F8 0C E4 17 36 00 C0
60C8- C0 0C 04 00 1B 1C 24 64
60D0- 06 00 63 24 24 23 00 DF
60D8- 0C 6D 38 3F 3F 4C 2D F8
60E0- 3B 00 1B 24 3F 0D 20 95
60E8- 2D 00 18 1E 07 00 18 28
60F0- 1F 3F 00 1B 3B 00 DB 63
60F8- 0C 0C 0C 0C 04 00 3B 07
6100- 20 24 0C AD 36 36 00 3B
6108- 2F 20 24 BC 06 00 3F 67
6110- 0C 0C E4 BF 36 00 20 FC
6118- 0C 05 38 3F 96 72 2D 00
6120- 23 2C 1F 1F 25 0C 0C 36
6128- 06 00 3B 1C 4D E4 3F 24
6130- 2D 2D 00 3B 07 20 2C 1C
6138- 0C 2D D6 0E 36 00 1B 24
6140- 0C 05 20 3F 3F 00 20 1C
6148- 0C 1C BF 0E 1E 76 2D 00
6150- 3B 27 08 1C 64 AD 36 77
6158- 06 00 1B 04 20 00 DB 0C
6160- 04 20 00 1C 1C 1C 0C 0C
6168- 0C 04 00 18 28 04 38 3F
6170- 16 2D 00 DB 0C 0C 0C 1C
6178- 1C 1C 04 00 1B 04 20 65
6180- 1C 3F 07 00 24 3F D6 24
6188- 24 0C AD 36 00 3B 27 24
6190- 24 2D 15 1E 7F 89 36 00
6198- 3F 07 20 24 0C 2D 05 00
61A0- 3B 67 24 3C 28 AD 36 36
61A8- 00 3F 27 24 24 2D 35 1A
61B0- 3F 00 DB 24 24 2C 2D D6
61B8- 3F 00 3F 27 24 24 2D B5
61C0- 77 36 00 24 24 BC 3A C4
61C8- 36 36 36 00 3B 27 21 24
61D0- 07 28 2D 00 3B 07 28 09
61D8- 24 24 04 00 DF 24 24 6C
61E0- F1 1E AE 05 00 3F 27 24
61E8- 24 04 00 DF 23 24 24 15
61F0- 15 C5 61 36 36 06 00 DF
61F8- 24 24 AC 56 0E 24 24 04
6200- 00 3F 27 24 24 2D 35 36
6208- 36 00 DB 24 24 2C 2D 32
6210- 1E 3F 00 C4 21 E4 BF 36
6218- 76 C4 A9 36 00 DF 24 24
6220- 2C AD 1E 37 0E 06 00 3B
6228- 1C 4D 1C 1C 1C 0C AD 06
6230- 00 1B 24 24 1F 28 2D 2D
6238- 00 3B 1C 24 24 4D 36 36
6240- 06 00 1B 1C 1C 24 6C 09
6248- 36 36 33 00 3C 1C 17 1E
6250- 24 24 6C 09 36 36 00 DF
6258- 6C E1 27 1C 6C 31 1E 06
6260- 00 1B 24 E4 1C 4D F1 06
6268- 00 3F 27 05 28 28 20 3F
6270- 3F 00 3B 27 24 24 2D 04
6278- 00 18 1C 1C 1C 04 00 3F
6280- 4C 24 24 3F 04 00 E0 1C
6288- 17 17 07 00 3F 27 00 31
6290- 3F 3F 27 2D 2D 05 38 3F
6298- 3F 2C 2D 2D 3C 3F 3F 2C
62A0- 2D 2D 3C 3F 3F 2C 2D 2D
62A8- 96 92 03 00 04 00 58 58
62B0- 58 58 58 58 58 39 3F 3F
62B8- 37 36 36 36 E7 1C 00
```

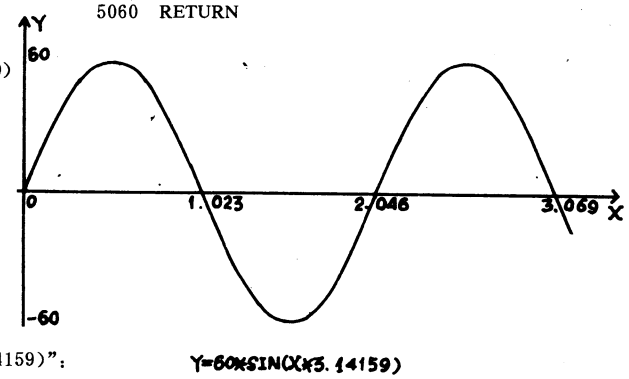
[例1]DRAW-CHAR

```
1  PRINT CHR$ (4);"BLOAD CHAR"
5  POKE 232,0:POKE 233,96
10  X=10:Y=10:M3=4:M4=6:G=1:W=1
20  HGR : HCOLOR=3:SCALE=1:ROT=0:GOSUB 6000:
    END
6000  XDRAW 66-W AT X,Y:XDRAW 66 -W AT X,Y
6010 V=PEEK(49152):IF V<128 THEN 6000
6020  POKE 49168,0
6022  IF ABS(V-139)<=2 AND V<  >140 THEN XDRAW
      66 AT X,Y
6025  IF V=137 OR V=129 THEN 6475
6030  IF V=138 OR V=136 THEN 6470
6035  IT V=139 OR V=149 THEN 6465
6040  IF V=141 OR V=154 THEN 6480
6045  IF V=155 THEN M1=0:M2=-1:W=0:GOTO 6000
6050  IF V=146 THEN RETURN
6055  IF V<160 THEN 6000
6057  IF V=160 THEN DRAW 65 AT X,Y:XDRAW 65 AT
      X,Y:W=1:GOTO 6465
6060  M1 = M3: M2 = M4: W = 1: DRAW 65 AT X, Y:
      XDRAW 65 AT X,Y:DRAW V-159 AT X,Y
6465  X=X+M1+1:IF X>278-M1 THEN X=0:Y=Y+
      M2+2
6466  IF Y>125+32 * G-M2 THEN Y=0
6468  GOTO 6000
6470  X=X-M1-1:IF X<0 THE X=278-M1:Y=Y-M2
      -2
6471  IF Y<0 THEN Y=125+32 * G-M2
6472  GOTO 6000
6475  Y=Y-M2-2:IF Y<0 THEN Y=125+32 * G-M2:
      X=X+M1+1
6476  IF X>278-M1 THEN X=0
6478  GOTO 6000
```

```
6480  Y=Y+M2+2:IF Y)125+32*G-M2 THEN Y=0:
      X=X-M1-1
6481  IF X(0 THEN X=278-M1
6482  GOTO 6000
```

[例2] SIN(X)

```
10  PRINT CHR$(4);"BLOAD CHAR"
20  POKE 232,0:POKE 233,96
30   HGR2:HCOLOR=6
40  HPLOT 0,80 TO 279,80:HPI     10,0 TO 10,160
50  HCOLOR=3:HPLOT 10,80
60  FOR I=0 TO 260
70  HPLOT TO I+10,80-60*SIN(I*3.14159/180)
180  NEXT
100 X=15:Y=87:N$="0":GOSUB 5000
110 X=108:N$="1.023":GOSUB 5000
120 X=183:N$="2.046":GOSUB 5000
130 X=268:N$="3.069":GOSUB 5000
140 X=20:Y=20:N$="60":GOSUB 5000
150 X=25:Y=140:N$="-60":GOSUB 5000
160 X=12:Y=4:N$="^":GOSUB 5000
170 X=279:Y=83:N$=")":GOSUB 5000
180 X=180:Y=160:N$="Y=60*SIN(X*3.14159)":
    GOSUB 5000
190 X=279:Y=90:N$="X":GOSUB 5000
200 X=18:Y=5:N$="Y":GOSUB 5000
1000  END
5000  HCOLOR=3:SCALE=1:ROT=0
5010  L= LEN (N$)
5020  FOR U=L TO 1 STEP-1
5030 A$=MID$ (N$,U,1)
5040  DRAW ASC(A$)-31 AT X,Y
5050 X=X-5:NEXT
5060  RETURN
```



向量图形表自动生成程序

贵州大学　计科系82级　　万维侠

CEC-Ⅰ和APPLEⅡ的BASIC中有向量作图法，这种方法绘图速度快且能旋转、放大，对编制教学、游戏等软件很有用，但是，编制向量图形表却很繁复，需花大量时间，还易出错。我编了这个CEC-Ⅰ机向量图形表生成程序，能方便地绘制图形并能生成正确的向量图形表。程序稍加修改就可以在APPLEⅡ上使用。

一　程序功能

1. 任意编图。可在屏幕上编绘图形，为了方便，图形点为放大的点，所编图形要求在64×64点以下。

2. 菜单提示。程序在运行时，每一过程都使用了菜单提示，方便操作。

3. 实现续编。图形表既可连续编完也可中途存盘关机，下次再调出程序和未完成的图形表继续编制。

另外，当所编图形同上一个图形相似时，可以保留上个图形方便编图。

二、程序使用

程序运行时有三个过程：编图、路径和生成。

在编图过程，以"十"为光标，可用键盘右边的四个方向键控制光标上、下、左、右移动，用Q、W、A、S键控制光标左上、右上、左下、右下移动，空格键改变当前绘图状态(画点或不画)，P键进入擦除状态，光标所过点都被擦除。编图完成后按ESC键结束，进入路径过程，按下ESC键时的光标位置就是路径的起点。

在路径过程，用四个方向键使光标走过每一个光点，如走错可按P键退回一步，也可连续按P键，连续退回，每个光点都走完后，按ESC键进入生成过程。

在生成过程，程序自动按路径生成向量图形，并询问所生成的向量图形是否有效，若有效，则可以编绘下一个图形，若回答N，则程序回到编图过程，并保留原图形，这时可以更改该图形直至生成满意的向量图形。

当要续编时，在程序开始的提示下，按下续编选择项2，这时询问待续编的图形表文件名，输入后程序自动由磁盘调入到内存。如果待续编的图形表本来就在内存中($6000开始)，当程序询问时直接按回车键，这种情况存在于当程序由CTRL-RESET中断或CTRL-C中断后要续编时。

另外请注意，图形表表头第二个字节本是无用单元，现作为续编使用，存储已编好的图形数目，当这个数目等于表头第一个字节值时就不能再续编了。对于不是由本程序生成的图形表要续编，必须相应改动表头第二个字节为已有图形数目才可。

当菜单提示时，第一个选项可按除其它选项键的任意键就可进入。

三、程序说明

10～40，初始化；

100～490，主程序；

500，每个图形相对地址的计算存储子程序；

600～620，图形表各分量的计算、存储子程序；
700～710，绘光标子程序；
750～760，绘光点子程序；
800～830，保留图形重绘子程序；
900～910，方向键控制光标移动子程序；
1000～1120，编图子程序；
2000～2080，路径子程序；
3000～3080，自动生成向量表子程序。

四、变量说明

W%，存图形数组；X%，Y%，存路径数组；Z，当前图形号；E，图形表基地址，定在$6000，即十进制24576；F，图形表分量存储地址指针；M、N，光标的位置坐标；B，编图的状态变量。

向量图形表编制完成后，就可在应用程序中使用DRAW语句或XDRAW语句调用了。向量图形的调用规则详见有关资料。

```
10 DIM W%(63,63),X%(700),Y%(700),B$(2)
20 SCALE=1:ROT=0
30 E=24576:POKE 232,0:POKE 233,96
40 B$(0)="不画":B$(1)="擦除":B$(2)="画点"
100 TEXT:PR#3:HOME:HGR2
110 PRINT"[向量图形表生成程序]":PRINT
120 PRINT"[1]编制新表":PRINT"[2]续编"
130 GET A$:PRINT:ON VAL(A$)GOTO 200,250
200 PRINT"编制新表":INPUT"有几个图形?";D
210 POKE E,D:FOR Z=2 TO 2*D+1:POKE E+Z,0:NEXT
220 F=E+2*(D+1):W=F:R=1
230 Z=0:GOSUB 500
240 GOTO 300
250 INPUT"待续编图形表文件名?";A$
260 IF A$<>""THEN  PRINT CHR$(4);"BLOAD";A$
270 IF PEEK(E+1)=PEEK(E) THEN PRINT"已满！":END
280 R=PEEK(E+1)+1:F=E+PEEK(E+R*2+1)*256
    +PEEK(E+R*2):W=F
300 FOR Z=R TO PEEK(E)
310 IF Z>R THEN VTAB 10:PRINT"请稍等";:FOR M=0 TO
    S:FOR N=0 TO S:W%(M,N)=0:NEXT:NEXT
320 HOME:VTAB 4:PRINT"第";Z;"个图形":PRINT
330 PRINT "[1] 24×24".PRINT"[2]38×38":PRINT"[3]
    64×64"
340 GET A$:S=24+(A$="2")*14+(A$="3")*40
350 T=INT(192/S):S=S-1
360 HGR2:HOME
370 GOSUB 1000:GOSUB 2000:GOSUB 3000
380 IF A$="N" THEN 370
390 POKE E+1,Z:IF Z<PEEK(E) THEN GOSUB 500
400 HOME:VTAB 3
410 IF Z<PEEK(E) THEN PRINT"[1]不用上次图形":PRINT
    "[2]使用上次图形"
420 PRINT"[3]存盘"
430 GET A$:ON VAL(A$) GOTO 440,450,470
440 NEXT Z:PRINT:PRINT"编制结束":END
450 IF Z=PEEK(E) THEN 440
460 GOSUB 800:Z=Z+1:GOTO 370
470 PRINT:INPUT"该图形表文件名:";A$
480 IF A$<>""THEN PRINT CHR$(4);"BSAVE";A$;",
    A";E;",L";F-E+1
490 GOTO 400
500 A=F-E:POKE E+(Z+1)*2,A-INT(A/256)*256:
    POKE E+(Z+1)*2+1,INT(A/256):RETURN
600 IF G<2 OR(V>0 AND V<4) THEN H=H+8^G*V:G=
    (G<2)*(G+1):POKE F,H:F=F+(G=0):H=H*(G>
    0):RETURN
610 IF H>0THEN F=F+1:G=(H<8):H=0:GOTO 600
620 POKE F,128,POKE F+1,128:F=F+2:G=1:H=V:
    RETURN
700 P=INT(T/2)-(S<24):X=M*T+85+P:Y=N*T+P
710 HPLOT X-P,Y TO X+P,Y:HPLOT X,Y-P TO X,Y+
    P:RETURN
750 X=M*T+85:Y=N*T:J=T-1-(S<24)
760FOR I=0 TO J:HPLOT X,Y+I TO X+J,Y+I :NEXT:
    RETURN
800 HGR2:HOME:HCOLOR=3
810 FOR M=0 TO S:FOR N=0 TO S
820 IF W%(M,N)THEN GOSUB 750
830 NEXT:NEXT:RETURN
900 Q=ASC(A$):M=M+(Q=21 AND M<S)-(Q=8
    AND M>0)
910  N=N+(Q=10 AND N<S)-(Q=11 AND N>0):RE-
    TURN
1000 PRINT "No.";Z:PRINT"[编图]":PRINT"Esc 结束":
    PRINT"[P]擦除":PRINT"Spc 状态"
1010 PRINT" Q      W":PRINT" A      S"
1020 M=INT(S/2):N=M:A$="  ":B=1
1030 HCOLOR=3*W%(M,N):GOSUB 700
1040 IF A$="P" THEN B=0
1050 IF A$="  " AND B=0THEN B=1
1060 IF A$="  " OR B=0 THEN B=-1*B:VTAB 9:
    PRINT B$(B+1)
1070 IF B>=0 THEN HCOLOR=B*3:GOSUB 750:W%(M,
    N)=B
1080 GOSUB 900:IF Q=27 THEN RETURN
1090 M=M+(Q=87 OR Q=83)*(M<S)-(Q=81 OR Q=
    65)*(M>0)
1100 N=N+(Q=65 OR Q=83)*(N<S)-(Q=81 OR Q=
    87)*(N>0)
1110 HCOLOR=3*NOT W%(M,N):GOSUB 700
1120 GET A$:GOTO 1030
2000 VTAB 2:HTAB 2:PRINT"路径":PRINT:HTAB 5:
    PRINT"退回":FOR I=1 TO 5:PRINT SPC(8):PRINT:
    NEXT
2010 X%(0)=M:Y%(0)=N:SM=700
2020 FOR K=1 TO 700
2030 HCOLOR=3*NOT W%(M,N):GOSUB 700
2040 GET A$:IF A$="P" AND K>1 THEN HCOLOR=3*
    W%(M,N):GOSUB 700:K=K-2:M=X%(K):N=Y%
    (K):NEXT
2050 GOSUB 900:X%(K)=M:Y%(K)=N
```

```
2060 IF Q=27 THEN SM=K:RETURN
2070 IF X%(K)=X%(K-1) AND Y%(K)=Y%(K-1)
   THEN 2040
2080 NEXT K:RETURN
3000 HCOLOR=3:H=0:G=0
3010 FOR K=1 TO SM
3020 M=X%(K):N=Y%(K):X=X%(K-1):Y=Y%(K
   -1)
3030 V=W%(X,Y)*4+(M)X)+(N)Y)*2+(M(X)*3
3040 GOSUB 600:POKE F,H:POKE F+1,0:DRAW Z AT
   64,128
3050 NEXT K:F=F+2
3060 VTAB 4:INPUT"有效吗?";A$
3070 IF A$="N" THEN F=W:GOSUB 800:RETURN
3080 W=F:RETURN
```

学生成绩管理程序

西宁　青海师大附中　朱江

对学生的成绩进行管理、统计是一项复杂的工作，随着计算机的普及，这项工作也越来越多地交给了计算机。此程序可完成对学生成绩的保存、修改、排序、打印等项工作，起到了用 DBASE Ⅱ 的基本作用，且不用再外加 Z80卡等外设。程序功能介绍如下：

1. 建立成绩库：此项可在磁盘上建立一个文本文件，保存输入的学生成绩。

2. 修改成绩库：此项可将已在磁盘上的成绩进行重写，可以修改一人或多人的成绩。

3. 扩充成绩库：可增加磁盘上成绩库的人数。

4. 删除成绩库：此项可将废弃不用的成绩库从磁盘上删去。

5. 打印(排序)成绩报表：此项可按照排好的名次逐个打印出所有学生的成绩报表，打印完后，可裁下交给学生保留。程序自动对齐数据，美观大方

6. 退出本系统：此项可将程序从内存中清掉，回到 BASIC 状态。

以上只是对程序功能的简单介绍，程序采用全汉化菜单提示。程序就说明到这里，以下是关于此程序的几点需注意的地方：

1. 程序在运行之前，应先让计算机处在中文状态下，否则将看不懂提示。

2. 程序是对学生的语文、数学、英语、物理、化学、生物、政治成绩进行管理，如有改动，只需修改第4、57句。

3. 存放学生成绩库的磁盘不能贴写保护。

```
1 REM     学生成绩管理系统
2 D$=CHR$(4)
3 HOME:VTAB 4:HTAB 6:PRINT
" ========================= ":
  HTAB 6: INVERSE: PRINT"* * * 学生成绩管理系统
  * * *":HTAB 6: NORMAL:PRINT
"========================="
4 VTAB 7:HTAB 6: PRINT"【语文|数学|英语|物理】":
  HTAB 9:PRINT"【化学|生物|政治】";CHR$(19)
5 HOME:PRINT"1:建立成绩库;":PRINT"2修改成绩库;":
  PRINT"3:扩充成绩库;":PRINT"4:删除成绩库;":PRINT
  "5:打印(排序)成绩报表;":PRINT"6:退出本系统。":IN-
  VERSE:PRINT"请选择:";:INPUT X:NORMAL:IF X(1 OR
  X)6 THEN 5
6 ON X GOSUB 7,8,20,23,25,26:HOME:PRINT"请稍候
  …":GOSUB 51:GOSUB 37:GOSUB 48:GOSUB 37:GOSUB
  39:GOTO5
7 HOME:INPUT"请输入人数:";N:DIM N$(N),A(N+1,
  10):FOR I=1 TO N:GOSUB 49:NEXT I:INPUT"请输入成
  绩库名称:";T$:RETURN
8 GOSUB 36:GOSUB 45
9 HOME:PRINT"1:修改一人成绩;":PRINT"2:修改部分人
  成绩;":PRINT "3:修改所有人成绩。"
10 INVERSE:PRINT"请选择:":NORMAL:INPUT X:IF X(1
  OR X)3 THEN 10
11 IF X=1 THEN INPUT"请输入学号或姓名:";N$:H1=
  VAL(N$):H2=H1:T=(H1=0):ON T +1 GOTO 14,17
12 IF X=2 THEN INPUT"请输入学号范围;";H1,H2
13 IF X=3 THEN 16
14 FOR I=1 TO N:FOR H=H1 TO H2:IF A(I,1)=H THEN
  GOSUB 50:IF H1=H2 THEN RETURN
15 NEXT H,I:RETURN
16 FOR I=1 TO N:GOSUB 50:NEXT:RETURN
17 FOR I=1 TO N:IF N$(I)=N$ THEN 19
18 NEXT I
19 GOSUB 50:RETURN
20 GOSUB 36:GOSUB 45
21 HOME:INPUT"增加多少名学生的成绩";N2:N1=N+1:
  N=N+N2
22 FOR I=N1 TO N:GOSUB 49:NEXT I: RETURN
23 POP:HOME:INPUT"请输入成绩库名称";T$:INTUT"按
  [Y]删除,按[N]返回。";A$:IF A$(  )"Y" THEN 5
24 GOSUB 48:GOTO 5
25 POP:GOSUB 36:GOSUB 45:GOSUB 27:GOTO 5
26 POP:HOME:VTAB 5:HTAB 7:PRINT"谢谢使用本系统,
  再见!":NEW
27 PRINT"打印机准备打印……":PRINT"按任一键开始打
  印:";CHR$(19)
28 POKE 1659,1:POKE 1915,0:POKE 2043,70
29 FOR I=1 TO N:PRINT" ┌———┬———┬———┬
```

```
┼──┼──┼──
```

其进行分析：

```
*2442L↘
2442- LDA $C000
……
2452- CMP #$95
2454- BEQ $24B2
```

可以看出$2453单元中存放“←”的键码$95，现用

```
*2453:A7↘
```

存入“'”的键码A7，再把软件重新存带或存盘，更改按键工作就完成了，以后再调出软件可以用并列在一起的“;”和“'”键方便准确地控制图形左右运动。依同样方法还可以把控制键改为其他组合，如“M”和“N”键。定期更换控制键使键盘上各键使用机会趋向平均相等有利于保护键盘。

```
0300- AD 63 03 85 40 AD 64 03
0308- 85 41 C6 40 10 02 C6 41
0310- E6 40 D0 0A E6 41 A5 41
0318- CD 65 03 D0 01 60 A0 00
0320- B1 40 CD 60 03 D0 E9 C8
0328- B1 40 CD 61 03 D0 E1 C8
0330- B1 40 CD 62 03 D0 D9 A6
0338- 40 A4 41 20 96 FD 4C 10
0340- 03
```

磁带小汇编系统自制简法

浙江新昌中学　石勇　张优方

在APPLE Ⅱ及其兼容机上要配有磁盘机才能使用INTBASIC中的小汇编。但是许多学校计算机小组只有教师机才配有磁盘机，所以将小汇编存入磁带是很有必要的。我们使用了一点小技巧，将INTBASIC中的小汇编存入磁带，这样就使得无磁盘机的APPLE Ⅱ及其兼容机也能享用小汇编，并且在BASIC中可用CALL语句调用自己汇编成的汇编程序。由于存取时间仅需二十多秒钟，故使用很方便。操作过程如下：

一、存带

1. 在教师机（有磁盘机）中插入DOS3. 3系统主盘，开机，待引导结束后，键入：

```
]INT↘（进入INTBASIC，“↘”表示按回车键，下同）
>CALL-151↘（进入监控）
*300:8D 87 C0 4C 66 F6 8D 82 C0 4C 3C D4↘
```

2. 准备好磁带机及放好磁带，键入：

```
*300. 30BWF425. FFFFW↘（按回车前先按下磁带机上的RECORD和PLAY两个键）
```

待光标出现（大约需三十秒钟）即告成功。

二、使用

]接通主机电源，键入：

]CALL-151↘（进入监控）

*300. 30BRF425. FFFFR（暂时不按下回车键，按下亦可）

2. 将已录制好的小汇编磁带准备好，把磁带倒回原处，按下PLAY键，即按回车键。待光标出现，键入：

300G↘（即进入小汇编系统）

本系统使用方法与INTBASIC中的小汇编完全一致。小汇编与其它状态的转换方法如下：

```
!$FF69G↘进入监控
!$306G↘进入SOFTBASIC
*300G↘进入小汇编系统
]CALL768↘进入小汇编系统
```

本系统中的300至30B一段汇编程序可放在您自己认为合适的地方。本系统需在16KRAM卡的支持下使用。请读者们不妨试一试。

巧用FID解密

武汉钢铁公司　第六子弟中学　杨克

众所周知，在APPLE Ⅱ及其兼容机上使用DOS3. 3操作系统时，可以用在文件名中插入控制字符的方法达到加密的目的。而解密时用TOOL KIT盘中的LOADAPA程序，便可找出暗藏的控制字符。但是，这样做比较麻烦（要先后调入四个文件），而且较费时间。对于没有TOOL KIT的用户，更是无能为力。

经过多次实践，我找到了一种利用DOS 3. 3主盘中的FID程序解密的方法，方法是这样的：

1. 运行FID程序，从主菜单选择“1”进行拷贝。

2. 屏幕提示输入源盘和目标盘所在驱动器的槽号和驱动器号，这时输入同一台驱动器的槽号和驱动器号。如：SOURCE SLOT?6↘

DRIVE?1↘

DESTINATION SLOT?6↘

DRIVE?1↘

3. 屏幕提示：FILENAME?键入“=”后回车，屏幕又提示：DO YOU WANT PROMPTING?，键入“Y↘”，表示要有选择地拷贝文件。

4. 按照屏幕提示，插入加密文件所在的盘敲任意键开始。

5. 屏幕依次显示出盘内各文件名，键入“N↘”跳过未加密文件，到加密文件时键入“Y↘”。

6. 屏幕提示插入目标盘，这时不要换盘，敲任意键继续。

7. 屏幕提示盘中已有同名文件，这时有三种方案供选择：①键入一个新文件名作为拷入文件的文件名；②键入“CTRL—C↘”跳过这个文件，放弃拷贝；③直接敲“RETURN”键，冲掉原同名文件强行拷贝。选择第1种方案，键入一个新文件名（未加密），并按回车

键，则盘中便拷入了一个内容与原加密文件相同，而文件名不同的新文件。

8. 若盘中还有其它加密文件，则可按此方法依次解密。

9. 解密完毕后，回到主菜单，选择“9”退出本程序，回到 Applesoft 状态。至此，文件解密工作结束。

CALL 新用法

北京铁道附中　　姜路

APPLESOFT 的 CALL 语句，用于 BASIC 程序中调用机器语言子程序。但当机器语言子程序的运行需要一些参数时（例如发声程序的音高和音长），通常是先用 POKE 语句，把参数先写进指定内存单元再进入调用，比较麻烦。

我发现在机器子程序开始处加入一个或多个 JSR $E74C，便可把 CALL 地址后的一个或多个“，参数”存入变址寄存器 X，供机器语言子程序使用。

以下是我编的示范程序。

该程序10—15句是把我改编的机器发声子程序放入内存。20－50是模拟“倒水”的发声程序，可以看到40句 CALL768后直接跟上音高和音长，十分方便。

```
10 FOR I=768 TO 796:READ A: POKE I,A:NEXT
15 DATA 32,76,231,134,6,32,76,231,134,7,173,48,192,
   202,208,5,198,7,208,1,96,136,208,245,164,6,76,
   10,3
20 FOR I=255 TO 1 STEP -1
30 CALL 768,I,10
40 NEXT
50 END
```

APPLE-Ⅱ的口令设置

沈阳教育学院数学系八八一班　　成鹰

本人在编 BASIC 程序时，利用随机文件编制了一口令，此口令用起来比较方便，易于修改口令内容，使他人不易发现，使程序功能得到保密，（暂记随机文件名为 ABC），在主程序之前应有如下小程序：

```
5 POKE 43225,0:POKE 43226,0:POKE 43227,0
10 NOMON:D$=CHR$(4):ON ERROR GOTO 100
15 PRINT D$;"OPEN ABC"
16 PRINT D$;"READ ABC"
17 INPUT A$
18 PRINT D$;"CLOSE ABC"
20 INPUT"Please input number!";Q$
21 IF Q$<>A$ THEN NEW:END

  主程序

100 NEW:END
```

说明：5句破坏 MON 入口，10句消除 DOS 指令的跟踪功能，使得在执行15－18句时不在屏幕上显示 A$ 值并防止 CTRL-C 中断。20、21句是输入码值并判断是否正确。

本程序若与《电子与电脑》5期的程序简易加密相结合，或在随机文件名中加入控制符号，可使得程序功能更好的保密。

性格类型测试程序

湖北省教育科学研究所　　叶平

在 PC 机上流传甚广的所谓“算命”软件，除去某些荒诞的成分，其中有一个测量性格的项目有一定的价值。笔者在中华机上编写了一个类似的程序，程序中采用了阴影菜单式选项和趣味清屏等方法，使操作一目了然，可供学校对学生的性格类型作简单的测试分类。

瑞士的一位心理学家在对人的性格类型进行多年研究后，按其特点分为四类，它们是敏感型、感情型、思考型和想像型。他发现相同类型的人容易交往，了解自己的性格属于哪一类，可以在生活与工作中扬长避短，使群众关系更为和谐。“性格类型测试”程序要求被试者逐一思考屏幕显示的若干问题，并根据每一问题下面列出的四个答案，按照符合自己情况的优先顺序给予应答。操作时，按空格键选项，用回车键应答。程序会自动按你应答的顺序对各答案赋分。当全部问题回答完毕后，屏幕将显示你在各类性格上的得分并给出你的性格类型。

由于程序中用“顺序文本文件”来储存问卷，所以必须编一简短的文本文件写入程序，把问题及答案上每一项作为一个域储存，并以 XL. DAT 为名存盘。该写入程序如下：

```
10 D$=CHR$(4):PRINT D$;"PR # 3":PRINT:
   HOME
20 PRINT D$;"OPEN XL. DAT":PRINT D$
30 INPUT A$
40 PRINT D$;"WRITE XL. DAT"
50 PRINT A$:INPUT"NEXT(Y/N)";Y$
60 IF Y$<>"N"THEN 30
70 PRINT D$;"CLOSE":END
```

T 文件“XL. DAT”中应写入的内容如下：

一. 我给别人留下的深刻印象可能是——
经验丰富
热情
灵敏
知识丰富
二. 当我按计划工作时总希望这个计划能够——
取得预期效果而不浪费时间精力
有趣并能和其他人一起进行
计划性强
能产生有价值的新成果
三. 我的时间宝贵所以总是首先确定要做的事——
有无价值
能否使别人感到有趣
是否安排得当按计划进行
是否考虑好了下一步计划
四. 对我来说最满意的情况是——
比原计划做得多
对别人有帮助
通过思考解决了一个问题
把一个想法和另一想法联系起来
五. 我喜欢别人把我看成是一个——
能完成工作任务的人
充满热情与活力的人
办事胸有成竹的人
有远见卓识的人
六. 当别人对我无礼时我往往——
立即表现出不快
心情不快但能很快消除
遣责对方
不去理他而考虑自己的事

对于没有磁盘机的用户，当然可以把上述内容用 DATA 语句储存，并适当修改程序。关于四类性格的特点，原文中还有若干解释，若能写在程序中将更能说明问题，限于篇幅不再赘述。用这一程序的设计方法，还可以改写成其他一些心理测量程序，如测试气质、性格倾向等等。

```
10 REM 性格类型测定(XL1. 3)
20 D$ =CHR$ (4):PRINT D$ ;“PR#3”:PRINT
30 DIM S(4),T(4),N$ (4),N1$ (4),X$ (4),X1$ (4),B
   $ (4)
40 FOR I=1 TO 4:READ N1$ (I),X1$ (I):NEXT
50 HOME:VTAB 9:HTAB 4:PRINT“[用空格键选择后，压
   下”;CHR$ (15);“RETURN”;CHR$ (14);”]”
60 FOR I=1 TO 4:T(I)=0:X$ (I)=X1$ (I):NEXT
100PRINT D$ ;“OPEN XL. DAT”
110 FOR I=1 TO 6
120 PRINT D$ ;“READ XL. DAT”
130 INPUT A$
140 FOR J=1 TO 4:INPUT B$ (J):NEXT J
150 PRINT D$
160 GOSUB 1000:GOSUB 2000
170 FOR J=1 TO 4:T(J)=T(J)+S(J):NEXT J
180 GOSUB 3000
190 NEXT I
200 PRINT D$ ;“CLOSE”
210 GOSUB 4000
220 VTAB 9 :HTAB 8 :PRINT “还要做下一人吗(Y/N)”;:
   GET Y$
230 IF Y$ =“Y” THEN 50
240 END
1000 VTAB 2:HTAB 4:PRINT A$
1010 FOR J=4 TO 7:VTAB J
1020 HTAB 1:PRINT N1$ (J-3):B$ (J-3)
1030 NEXTJ:RETURN
2000 G=1 :W=4
2010 FOR K=1 TO 4:S(K)=0:N$ (K)=N1$ (K):NEXT K
2020 HTAB 1:VTAB(G+3):PRINT CHR$ (15):N$ (G):B
   $ (G):CHR$ (14)
2030 GET Y$ :Y=ASC(Y$ )
2040 IF Y=32 THEN 2070
2050 IF Y=13 THEN PRINT CHR$ (7):GOTO 2100
2060 GOTO 2030
2070 HTAB 1:VTAB(G+3):PRINT N$ (G):B$ (G):G=G
   +1
2080 IF G>4 THEN G=1
2090 IF S(G)<>0 THEN 2070
2095 GOTO 2020
2100 FOR K=1 TO 4
2110 IF S(K)=0 THEN 2130
2120 NEXT K:RETURN
2130 N$ (G)=STR$ (W)+“  ”:S(G)=W
2140 W=W-1:IF W=0 THEN MUSIC 100,100 :RETURN
2150 GOTO 2070
3000 HTAB 1
3010 FOR J=7 TO 2 STEP -1
3015 VTAB J
3020 PRINT CHR$ (26)
3025 FOR T=1 TO 100:NEXT T
3030 NEXT J:RETURN
4000 HOME
4010 FOR I=1 TO 3:FOR J=I TO 4
4020 IF T(I)>T(J) THEN 4040
4030 T=T(I):T(I)=T(J):T(J)=T
4035 T$ =X$ (I):X$ (I)=X$ (J):X$ (J)=T$
4040 NEXT J,I
4050 FOR I=1 TO 4:VTAB(I+1):HTAB 8
4060 PRINT “(”;X$ (I);“)得分=”;T(I)
4070 NEXT J
4080 VTAB 7:HTAB 5:PRINT CHR$ (15);“您的性格类型
   为：”;X$ (1);CHR$ (14)
5090 RETURN
5000 DATA“A ”,敏感型,“B ”,感情型,“C ”,思考型,“D ”,
   想像型
```

《C 语言初阶》讲座

第一讲　Turbo　C

天津师大计算机系　李文兵

编者的话：C 语言是描述、表达能力极强的结构化语言。它不仅适合于通用的程序设计，也是发达国家开发系统软件的主流语言。大家熟知的 UNIX 操作系统、dBASE Ⅲ 等，就是应用 C 编制的。有专家预计，C 语言不久将取代 PASCAL。鉴于 C 语言日益引起广大软件工作者的注意和学习兴趣，我们特邀天津师范大学计算机系李文兵老师主持这个入门讲座，期望能为学习 C 语言的读者提供一点帮助。

本讲座计划共十讲：

1. Turbo C
2. Tcc 编译和实用程序
3. C 的数据仅有数据类型之分吗？
4. 什么是函数？
5. C 运算符知多少？
6. C 的语句多吗？
7. C 预处理程序之功能
8. 结构数据可构造数据结构
9. 掌握了指针才算掌握了 C
10. 文件管理

为配合本讲座的学习，本刊将在 91 年上半年出版“C 语言程序设计 300 题”专辑。

我们在为计算机编制程序时究竟使用哪种语言好呢？这要根据实际情况来确定，也就是根据资源条件（包括技术力量）和要实现的功能来确定，即因地制宜。

目前，C 语言受到人们的普遍喜爱。因为，它既是成功的系统描述语言，又是通用的程序设计语言。作为系统描述语言，，用它成功地编写出 UNIX、FOX-base 等。C 语言的可读性和可移植性比起汇编语言来要好得多，而代码效率却只比汇编语言低 10～20%，因此，人们给 C 起了个美称，叫便携汇编语言。作为通用的程序设计语言，越来越多的人使用它，替代其它高级语言，编写信息系统软件。目前，C 语言已成为世界发达国家软件开发的主流语言。

1. C 语言与 Turbo C

C 语言是 70 年代贝尔（Bell）实验室为描述 UNIX 操作系统和 C 编译程序，而开发的一种系统描述语言。

1969 年，美国贝尔实验室的两个研究人员 Ken Thompson 和 Dennis M. Ritche，开始开发 UNIX，用了不到两个人年的时间就研制成功了，当时的 UNIX 版本是用汇编语言编写的。

1970 年，K. Thompson 为了提高 UNIX 的可读性和可移植性，在 BCPL 语言的基础上，开发了一种新的语言，起名叫“B”，之所以叫这个名字，据说其中一个因素就是根据 BCPL 的字头。由于 B 语言存在着一些缺点，例如，它没有定义数据类型，这就无法支持多种数据类型。因此，该语言没有流行起来。

从 1971 年开始，D. M. Ritche 用了一年左右的时间，在 B 的基础上开发了第一个 C 编辑程序，1972 年投入使用，C 语言就是这样诞生的。因为这个编译程序是在 B 语言的基础上开发的，无论是在英文字母序列中也好，还是在 BCPL 这个名字中也好，排在 B 后面的均为 C，故起名为 C 语言。

1973 年，K. Thompson 和 D. M. Ritche 两人合作，用 C 语言把 UNIX 又写了一遍，这就为 UNIX 的移植和发展奠定了基础。

1975 年 UNIX 第 6 版本公诸于世，C 语言即被举世瞩目。

1978 年 Briam W. Kernighan 和 D. M. Ritche 合写了名为《The C Programming Language》的著作，为 C 语言在全世界范围内的推广与普及提供了一本很好的教科书，被世人誉为标准版本。1981 年日本著名学者石田晴久把这本书译为日文。1982 年我国 UNIX、C 的著名专家孙玉方、孟庆昌先生把这本书编译为中文教材。从此，C 语言便越来越受到软件工程人员的喜爱，他们放弃原来已熟悉的 FORTRAN、PASCAL 等语言，而使用 C 语言开发软件。

C 语言开始是附属于 UNIX，运行在 PDP—11 机上，但到 1978 年以后，就移植到各种微机上，能在各种操作系统下运行。这就出现了各种 C 编译系统。这些 C 编译系统的规格是不尽相同的，因此，用户编辑的源程序要与所使用的编译系统对应，即各种编译系统下所编的源程序不具有互换性。

为提高互换性，美国标准协会（ANSI）的信息系统委员会（X3 委员会）中的 J11 组对 C 语言进行了规格化，提出了美国标准 C（ANSI C）方案，得到了各国的承认。如果该标准能得到国际标准化组织（ISO）的批准，C 语言将真正成为移植性高的语言。最近发表的一些 C 编译系统几乎都符合 ANSI C 标准。

Turbo C 编译系统是 Borland International 公司的产品，它支持 C 标准版本，支持 ANSI C 提案，备有与硬件有关的特殊函数。该编译系统深受国内外软件人员的欢迎。它有漂亮的编辑程序；运行速度较快；系统不算很大且功能较强，又具有现代 C 的风格。

2. 可执行程序的产生过程

从要解决的问题到可执行程序要经过如下 6 步：

(1)把要解决的问题分为几个小问题，明确它们的联系及顺序；

(2)说明每个小问题的详细内容，以及有关算法；

(3)使用编辑程序，用 C 语言编写源程序；

(4)使用 C 编译程序，编译由(3)得到的 C 源程序；

(5)如果编译发生错误，则要修正源程序，反复进行(4)和(5)，直到没有错误为止。

(6)使用链接程序，把编译后的目标文件变为可执行文件。如果发生错误，则要变更源文件或是编译和链接的环境，重新编译和链接，直到没有错误为止。

Turbo C 是在 MS-DOS 操作系统上使用的 C 编译系统。因此，若使用 Turbo C 就必须能熟练使用 MS-DOS，这就要掌握 MS-DOS 的基本知识。

在 Turbo C 中，编辑程序、编译程序、链接程序等全部都是独立的程序。然而，Turbo C 提供了统一调配这些程序的开发环境，使得 Turbo C 用起来很方便。可以说，Turbo C 是编辑、编译、链接、调试各功能的一体化、有机结合的产物。

3. Turbo C 2.0 **特点与配置要求**

Turbo C 2.0 是 Turbo C 的最新版本。

(1) Turbo C 2.0 的特点

该版本在 1.5 版的基础上增加了许多功能，具有如下特点：

- 编辑、编译、调试、运行一体化
- 综合调试程序具有单步执行、单步跟踪、断点设置、表达式监视和求值等功能。
- 支持 Turbo Debugger 独立调试程序
- 具有更快的编译、链接程序(快 20～30%)
- 具有更快的内存分配函数和串函数
- EMS(扩展内存规范)用作编辑缓冲区
- 能仿真 80x87、浮点运行速度更快
- 新增加了 Signal 和 Raise 函数
- 新增加的—emit—函数允许用户在编译时向程序插入机器代码
- 高级图形库中增加了许多新函数，包括可安装的驱动程序和字体。
- 支持命令行上的匹配符 * 和？等
- 可连接生成小模式的 COM 文件，运行速度高
- 支持 Long double 常数和变量
- 能自动进行快缩进和回退及优化填充
- MAKE 实用程序可自动进行依赖关系检查
- 新增加一些实用工具，如：

THELP：用于在 DOS 下得到与 Turbo C 2.0 版本内容相关的帮助系统的内存驻留程序

CINSTXFR · EXE：用来把 Turbo C 1.5 版综合开发环境的配置文件转换成 2.0 版的对应文件

OBJXREF · EXE：用于按指定目录顺序搜索所有目标文件和库文件

(2)Turbo C 2.0 的配置要求

Turbo C 2.0 有如下配置要求

- 适于 IBM PC 系列机，包括 XT、AT、PS/2 及其它兼容机
- 需要 2.0 或更高版本 DOS 支持
- 至少需要 448K RAM
- 80 列彩/单监视器
- 至少一个软盘驱动器，建议使用两个软盘或一硬盘带一软盘
- 能仿真 80x87 协处理器，若系统中有该芯片，将大大加快浮点运算速度

4. Turbo C 2.0 **内容**

Turbo C 2.0 共有 6 张盘，各盘内容如下：

#1 盘 INSTALL/HELP(安装/帮助盘)

INSTALL · EXE 安装程序

README · COM README 文件阅读程序

TCHELP · TCH 帮助文件

THELP · COM 读取 TCHELP · TCH 的驻留程序

THELP · DOC THELP · COM 文件的文档

README 有关 Turbo C 最新信息的文件

#2 盘 INTEGRATED DEVELOPMENT ENVIRONMENT(综合开发环境盘)

TC · EXE Turbo C 综合编译程序

TCCONFIG · EXE 配置文件转换程序

MAKE · EXE 程序管理工具

GREP · COM 文件检索

TOUCH · COM 时间记录器

#3 盘 COMMAND LINE/UTILITIES(命令行编译程序/实用工具盘)

TCC · EXE 命令行编译程序

CPP · EXE 预处理程序

TCINST · EXE TC · EXE 配置设置工具

TLINK · EXE 链接程序

HELPME ! · DOC 一般问题和答案

#4 盘 LIBRARIES(库程序盘)

COS · OBJ 小型模式启动程序

COT · OBJ 微型模式启动程序

COL · OBJ 大型模式启动程序

MATHS · LIB 小型模式数学库

MATHL · LIB 大型模式数学库

CS · LIB 小型模式运行库

CL · LIB 大型模式运行库

EMU · LIB 8087 仿真库

GRAPHICS · LIB 图形库

FP87 · LIB 8087 库

TLIB · EXE 库管理工具

#5 盘 HEADER FILES/LIBRARIES(标题文件

和库程序）

????????．H　标题文件

〈SYS〉　SYS\＊．H 标题文件目录

COC．OBJ　紧缩模式启动程序

COM．OBJ　中型模式启动程序

MATHC．LIB　紧缩模式数学库

MATHM．LIB　中型模式数学库

CC．LIB　紧缩模式运行库

CM．LIB　中型模式运行库

#6 盘　EXAMPLES/BGI/MICS（例子/BGI 图形库/MICS 文件盘）

UNPACK．COM　打开．ARC 文件的工具

OBJXREF．COM　目标文件交差引用工具

COH．OBJ　巨型模式启动代码

MATHH．LIB　巨型模式数学库

CH．LIB　巨型模式运行库

GETOPT．C　命令行选择分析器

HELLO．C　源程序例子

MATHERR．C　数学库例外情况处理源程序

SSIGNAL．C　ssignal 和 gsignal 函数源程序

CINSTXFR．EXE　传送 1．5 版配置到2．0版的工具

INIT．OBJ　连接 Prolog 时的初始化代码

BGI．ARC　BGI 驱动程序和字体

BGIOBJ．EXE　字体和驱动程序转换工具

ATT．BGI　ATT400 图形卡驱动程序

CGA．BGI　CGA 图形驱动程序

EGAVGA．BGI　EGA 和 VGA 图形驱动程序

HERC．BGI　Hercules 图形驱动程序

IBM8514．BGI　IBM8514 图形卡驱动程序

PC3270．BGI　PC3270 图形驱动程序

GOTH．CHR　哥特式字符集

LITT．CHR　小字符集

SANS．CHR　sans serif 字符集

TRIP．CHR　立体字符集

BGIDEMO．C　图形演示程序

STARTUP．ARC　启动源程序的 ARC 文件和其它相关文件

RULES．ASI　和 Turbo c 接口的汇编 include 文件

CO．ASM　启动代码的汇编源程序

SETARGV．ASM　命令行分析的汇编源程序

SETENVP．ASM　环境处理的汇编源程序

BUILD-CO．BAT　建立启动代码模块的批处理文件

MAIN．C　交互式 C 主文件

EMUVARS．ASI　仿真程序的汇编变量说明

WILDARGS．OBJ　匹配符参数扩充模块的目标码

EXAMPLES．ARC　各种例子

CPASDEMO．PAS　Turbo Pascal 4．0 和 Turbo C 2．0 接口演示 Pascal 源程序

CPASDEMO．C　Turbo Pascal 4．0 和 Turbo C 2．0 接口演示 C 源程序

CTOPAS．TC　在与 Turbo Pascal 4．0 程序链接时为产生正确格式的 Turbo C 模块所必须的 TC．EXE 配置文件

CBAR．C　PBAR．PRO 文件中用到的函数例子

PBAR．PRO　Turbo Prolog 和 Turbo C 接口演示 Prolog 程序

WORDCNT．C　源程序级调试演示例子

WORDCNT．DAT　WORDCNT．C 中用到的数据文件

5. Turbo C 2.0 **的安装**

Turbo C 2.0 版有两种编译程序，即综合环境编译程序 TC 和命令行编译程序 TCC。其安装和启动按如下方法进行。

(1)安装前准备工作　先用 DOS 的 DISKCOPY 命令对六张盘作一备份，把源盘保存好，使用备份工作。

(2)安装方法有两种：

①使用 DOS 的 copy 命令　可把源程序中必需的文件拷贝到工作软盘或硬盘的某一子目录下，如使用如下命令：

A＞copy　a：TC．EXE　b：↙

A＞copy　a：TC．EXE　c：\TC↙

可把 TC．EXE 文件分别拷贝到 B 盘上或 C 盘的 TC 子目录下。

②利用安装程序　使用第一张盘中的 INSTALL．EXE 文件，也可安装 Turbo C 2.0 版。命令为：

A＞INSTALL↙

INSTALL 有三种选择：

(1)Hard Disk 把 Turbo C 2.0 各部分安装到硬盘的各子目录下，并根据各子目录创建 Turbo C．CFG 文件。

(2)update from TC 1.5 把 Turbo C 1.5 版升级为 2.0 版，并传送当前 TC．EXE 中的配置选择项。

(3)Floppy Disk　在软盘中建立 Turbo C 2.0,必须事先准备好三张格式化了的空盘,每次运行 INSTALL 时,都让你安装一种存储模式的 Turbo C ,如果想安装几种模式,就得有几套盘,每一套对应一种模式。

6.综合开发环境 TC

安装好 Turbo C 2.0 以后,在其目录下,打入 TC 或 TCC 即可分别启动 Turbo C 2.0 综合开发环境或命令行编译程序,然后,使用这两种方式之一就可以进行程序设计了。本讲先介绍 TC 的用法。

(1)TC——综合开发环境

TC 是 Turbo C 的综合开发环境执行程序。在该环境中,开发程序所必须的所有工具并非七零八落,为方便用户,而把它们综合为一体。其中除了编辑程序、编译程序、链接程序之外,还有 MAKE(重新编译、链接)、编译的状态设定、debug 等强有力的工具,构成一个功能很强,使用极为方便的系统。Borland 公司把该环境叫做 Integrated Development Environment(综合开发环境)。

(2)TC 启动方法

装入 Turbo C 2.0 后,在 DOS 下,键入如下命令即可启动 TC。

A>TC↙

TC 启动后,出现主屏幕和版本信息,按任意键,版本信息消失,按 shift-F10,版本信息重现。主屏幕由四部分组成,即主菜单、编辑窗口、信息窗口和功能键提示,如图 1.1 所示。

(3)TC 命令行开关

TC 有如下命令行开关

/c——加载配置文件　其用法如:

TC　/c myconfig

该命令使 Turbo C 在当前目录下查找名为 myconfig 的配置文件。注意/c 与文件名之间无空格。

/b——重新编译 project 里的所有文件,把编译信息打印在输出设备上,返回 DOS。其用法如:

TC　/c myconfig • tc /b

或

TC　/b

/m——只编译、键接过时文件

/d——在检测到合适的硬件时,使用双监视器。

(4)TC 热键

TC 中可使用的热键如表 1.1 所示。

表 1.1　TC 热键

键	功　　能
F1	激活帮助窗口,提供有关当前位置的信息
F2	编辑的文件存盘
F3	加载文件(出现输入框)
F4	程序运行到光标所在行
F5	放大、缩小活动窗口
F6	开关活动窗口
F7	在调试模式下运行程序,跟踪到函数内部
F8	在调试模式下运行程序,跳过函数调用
F9	执行 Make
Ctrl-F1	调用有关函数的上下文帮助
Ctrl-F3	显示调用栈
Ctrl-F4	计算表达式
Ctrl-F7	增加监视表达式
Ctrl-F8	断点开关
Ctrl-F9	运行程序
Alt-F1	显示上次访问的帮助
Alt-F3	选择文件加载
Alt-F6	开关活动窗口里的内容
Alt-F7	定位上一错误
Alt-F8	定位下一错误
Alt-F9	把文件编译为 OBJ 文件
Alt-B	转到 Break/Watch 菜单
Alt-C	转到 Compile 菜单
Alt-D	转到 Debug 菜单
Alt-E	转到 Edit 菜单
Alt-F	转到 File 菜单
Alt-O	转到 Option 菜单
Alt-P	转到 Project 菜单
Alt-R	转到 Run 菜单
Alt-X	退出 TC,返回到 DOS

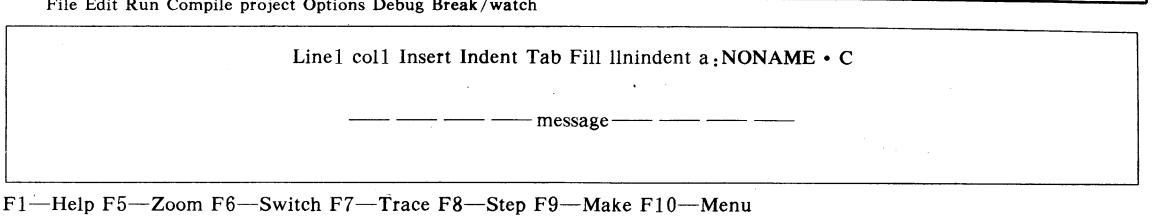


图 1.1　主屏幕

学用单片机

单片机化产品中的键盘和显示器的设计

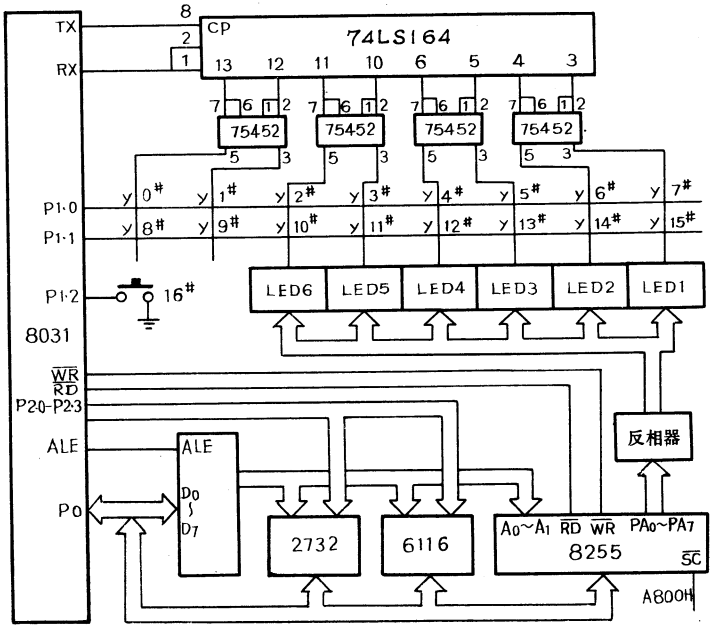
安徽大学电子工程与信息科学系　吕锡庆

在单片机化产品中，为了实现参数预置、功能转换和现场参数的显示等，通常需配置键盘和显示器。在软件设计中，一般也都是先设计键盘和显示程序，然后再根据键盘各键的定义，设计相应的键处理程序。因此，在单片机化产品的设计中，键盘和显示器的设计是首先要考虑的问题。本文就该问题介绍一个设计实例，给出的电路图和程序清单均经过实际检验。

一、硬件电路设计

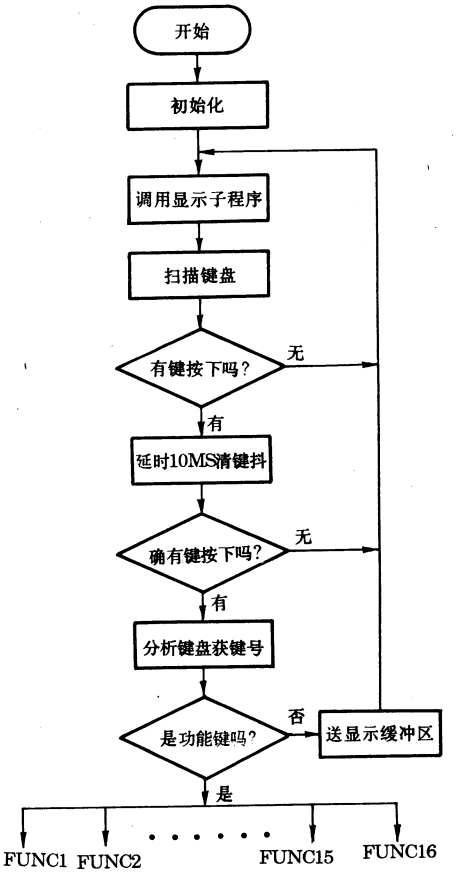
在51单片机的应用系统中，若串行口不作通讯用，则可用串行口作控制键盘和显示器的接口。图1给出了某一单片机化产品中的键盘和显示器的电路图。

图中，74LS164为8位串行移位寄存器，它将8031单片机送来的8位串行数据变成8位并行数据，经驱动器75452送往键盘和显示器，作键盘的列扫描信号或显示器的位控信号。键盘的两根行线分别与单片机的P1.0、P1.1相连接，向单片机提供行信号。为了减小面板的面积和节约成本，整个键盘只设置了17只键，其中16#键为复用控制键，它单独与单片机的P1.2相连接，向单片机提供复用控制信号。0#至15#键为复用键。当16#键不按下时，0#至15#键分别作0至F数据键用。当16#键按下时，0#至15#键分别作16个功能键用。这足以满足一般的应用系统的需要。扩展的8255的A端口输出的数据经反相器送往显示器，作段码信号。



二、软件设计

1. 程序框图



（1）一开机，首先执行主程序，完成初始化。然后调用显示子程序，在显示器上给出提示符rd，告诉操作者机器已准备好。

（2）当显示器上出现rd提示符后，立即调用键盘子程序，这时，操作者可从键盘输入数据或命令，经键盘扫描和键盘分析，得键号于R2中，返回主程序。

（3）在主程序中判断按下的键是数据键还是功能键，若是数据键，将键号送往地址显示缓冲区（安排在内部RAM的52H至55H单元），并在显示器的高4位上显示。若是功能键，则转入相应的功能键处理程序。

（4）在功能键处理程序中，若需要，可从地址显示缓冲区中取出先前输入的数据，作内部RAM或外部RAM的显示、修改地址，也可作某些特定程序的启动地址。

2. 程序清单

（1）主程序

```
START:MOV SP,#18H      ;设置堆栈指针
```

```
        MOV A,#80H      ;8255 控制字送 A
        MOV  DPTR,  #
        0A803H          ;8255 控制口地址送 DPTR
        MOVX @DPTR,A    ;置 8255 工作于方式 0,A 口为
                        输出状态
        MOV R2,#06H     ;显示缓冲区的长度送 R2
        MOV R0,#50H     ;显示缓冲区首址送 R0
        MOV  DPTR,  #
        CTAB            ;初始显示字符表首址送 DPTR
LOOP:   CLR A
        MOVC  A,  @A +
        DPTR            ;取出初始显示字符
        MOV @R0,A       ;初始显示字符送显示缓冲区
        INC R0
        INC DPTR
        DJNZ R2,LOOP
        MOV R4,#04H     ;设置允许从键盘连续送数的最
                        大次数
        MOV R3,#00H     ;设置送数次数计数器
LOOP1:  LCALL DISP      ;调用显示子程序
        LCALL KEYDE     ;调用键盘子程序
        JNZ LOOP2       ;若 A 的内容不为 0,则有键按
                        下,获键号于 R2 中,转 LOOP2
        SJMP LOOP1      ;无键按下,转 LOOP1
LOOP2:  MOV A,R2        ;键号送 A
        JB ACC.4,FUNC   ;若键号大于 FH,则为功能键,
                        转 FUNC
        INC R3          ;是数字键,计一次数
        LCALL SHFT      ;调用地址显示缓冲区移位子程
                        序,将内部 RAM 的 52H 至 55H
                        单元的内容由低向高移动一个
                        字节,再将 R2 中的键号送入 52H
                        单元
        DJNZ R4,LOOP1   ;按数字键次数不满 4 次则转
        MOV R4,#04H     ;按数字键满 4 次后,重新初始化
                        R4、R3
        MOV R3,#00H;
        SJMP LOOP1
CTAB:   DB 11H,11H,
        11H, 11H,0DH,
        10H             ;初始化显示字符表显示空白,
                        空白,空白,空白,d,r
FUNC:   MOV A,R2        ;功能键号送 A
        RL A            ;功能键号左移 1 位
        CLR C
        SUBC #20H
        MOV  DPTR,  #
        FTAB            ;功能键入口表首址送 DPTR
        JMP @ A+DPTR    ;转功能键入口表
FIAB:   AJMP FUNC1
        AJMP FUNC2
        ⋮
        AJMP FUNC16
```

(2) 键盘子程序

```
KEYDE:MOV A,#OFFH
        MOV  DPTR,  #
        0A800H          ;8255A 口地址送 DPTR
        MOVX @DPTR,A    ;8255A 口输出全“1”,使显示
                        器显示空白
        MOV P1,#OFFH    ;使 P1 口输出全“1”
        MOV  SBUF,  #   ;使键盘列线输出全“0”,
        OFFH            扫锚键盘
WAIT:   JNB TI,WAIT     ;等待串行口数据发送完
        CLR TI          ;发送完,请串行口中断标志
KEY1:   JNB P1.0,SCA1   ;第 1 行有键按下则转
        JB P1.1,KEY2    ;1,2 两行均无键按下转 KEY2
SCA1:   ACALL DLIOMS    ;有键按下,延时 10MS 消除
                        键抖动
        JNB P1.0,SCA2   ;第 1 行确有键按下转 SCA2
        JB P1.1,KEY2    ;1,2 行均无键按下,是键抖动
                        失误,转 KEY2
SCA2:   MOV R6,#01H     ;为先扫锚第 1 列做准备
        MOV R2,#00H
        MOV R7,#08H
KEY5:   MOV SBUF,A      ;逐列扫描
KEY3:   JNB TI,KEY3     ;等待串行口数据发送完
        CLR TI          ;发送完了,请串行口中断标志
        JNB P1.0 SCA0   ;第 1 行有键按下则转
        JB P1.1,NEXT    ;第 2 行无键按下转 NEXT
        MOV R1, #08H    ;第 2 行有键按下,行值 08H
                        送 R1
        SJMP SCA4
SCA0:   MOV R1,#00H     ;第 1 行有键按下,行值 00H
                        送 R1
SCA4:   JB P1.2,SCA3    ;复用控制键未按下转 SCA3
        MOV A,R1
        ADD A, #10H     ;若复用键按下,则行值加 10H
        MOV R1,A
SCA3:   MOV A,#OFFH
        MOV SBUF,A      ;令列线输出全“0”
KEY4:   JNB TI,KEY4     ;等待串行口数据发送完
        CLR TI          ;发送完,请中断标志
KEY6:   JNB P1.0,KEY 6  ;等待按键释放
        JNB P1.1,KEY6
        JNB P1.2,KEY6
        MOV A,R1
        ADD A,R2        ;行号加列号得键号
        MOV R2,A        ;键号送 R2
        MOV A,#01H      ;使 A 不等于 0
        RET             ;返回主程序
DLIOMS: MOV R7,#0AH     ;延时 10MS 子程序
DLA0:   MOV R6,#OFFH
DLA1:   DJNZ R6,DLA1
        DJNZ R7,DLA0
        RET
```

(3)显示子程序

```
DISP:   SETB PSW.4      ;选 2 区寄存器
        MOV R6,#06H     ;显示缓冲区长度送 R6
        MOV R7,#20H     ;显示器的位控码送 R7
        MOV R1,#50H     ;显示缓冲区首址送 R1
DISP1:  MOV DPTR,
        CSTAB           ;段码表首址送 DPTR
```

```
        MOV A,@R1       ;取显示缓冲区字符
        MOVC  A, @A +
        DPTR            ;查表得对应段码送 A
        MOV   DPTR,  #
        OA800H          ;8255A 口地址送 DPTR
        MOVX @DPTR,A    ;段码送 8255A 口
        MOV A, R7
        MOV SBUF,A      ;发送位控码
        RR A            ;位控码右移 1 位,为扫描显示
                        邻位做准备
        MOV R7,A
WAIT2:JNB TI,WAIT2      ;等待位控码发送完
        CLR TI          ;发送完,请中断标志
        ACLL DL2MS      ;调用 2MS 延时子程序
        INC R1          ;修改显示缓冲区地址指针
        DJNZ R6,DISP1
        CLR PSW.4       ;扫描显示完,恢复原寄存器区
        RET             ;返回主程序

CSTAB:DB C0H,F9H,A4H,B0H,99H,92H,
        DB 82H,F8H,80H,98H,88H,83H,
        DB C6H,A1H,86H,8EH,CEH,FFH;段码表分别对应
        显示 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F,r,空白
DL2MS:MOV R3,#02H       ;延时 2MS 子程序
DLA2:  MOV R4,#FFH
DLA3:  DJNZ R4,DLA3
        DJNZ R3,DLA2
```

(4) 地址显示缓冲区移位子程序

```
SHFT:  MOV 56H,R3       ;从键盘送数的次数送 56H 单元
        MOV R6,#03H
        MOV R0,#54H
        MOV R1,#55H
SHFT1: MOV A,@R0        ;52H 至 55H 单元的内容
                        由低向高移动一个字节
        MOV @R1,A
        DEC R0
        DEC R1
        DJNZ R6,SHFT1
        MOV A,R2        ;键号送 52H 单元
        MOV @R1,A
        RET             ;返回主程序
```

由于单片机的应用场合千差万别,其软、硬件的差别虽然较大,但就其键盘和显示器的软、硬件设计而言,一般差别比较小。本文所述的软件和硬件,只要稍加修改,即可用于一般的单片机应用系统中。

EPROM 及 8751 安全智能编程法

郑州市解放军信息工程学院二系　翁甲辉　李　柯

近几年来,我们在单片机开发与 EPROM 编程器的科研实践中,摸索试验成功一种行之有效的 EPROM 安全智能编程法,现将其基本工作原理介绍一下:

一、常用 EPROM 编程法简介

目前国内外常用的 EPROM 编程法主要有三种:

1. 一般编程法(Standard Programming)

通常的 EPROM2716,2732,2764,27128 及单片机 8751 均采用这种编程法. 其编程原理是:将 Vpp 直流高压直接加在芯片的 Vpp 引脚上(2732 例外),编程脉冲宽度为 50±5ms. 每个字节 EPROM 经 50ms 编程后进行校验,若出错则判为芯片失效. 2K 字节的 2716 编程时间约需 100 秒。

2. 智能编程法(Inteligent Programming)

Intel 公司的智能编程法可以用来对 EPROM 2764,27128,27256,27512 等芯片的编程。其编程原理是:将 Vcc 电压由+5V 提高到 6±0.25V,Vpp 直流高压直接加在芯片的 Vpp 引脚上(27512 例外);先加持续时间为 1ms 的"起始编程脉冲"进行起始编程,然后进行字节校验;若这个字节编程写入尚未成功,则再加 1ms 的起始编程脉 冲,直至这个字节校验成功为上,这时起始编程脉 冲累计有 Xms(X≤25);最后再加 3Xms 的"附加编程脉冲"来加固原来 Xms 的编程效果。这样,对于不同厂家、不同工艺、不同芯片、不同的 V_{PP}编程高压以及不同的字节单元,其实际编程时间是不同的。

显而易见:智能编程法对编程时间进行的智能控制的优点是十分明显的,它既能保证每个字节编程的可靠性,又能大幅度节省编程时间。

3. 快脉冲编程法(Quick—Pulse Programming)

Intel 公司最新的快脉冲编程法可用来对 EPROM 2764、27128、27256、27512 等芯片的编程。其编程原理是:将芯片的 V_{CC}电压由+5V 提高到 6.25±0.25V,并将 V_{PP}直流高压加在芯片的 V_{PP}引脚上(27512 例外);然后用脉宽为 100μs 的编程脉冲进行试写入,经字节校验后,若发现这个字节编程写入尚未成功,则继续加 100μs 的编程脉冲,直至字节校验成功为止;若编程脉冲已经加了 25 次(即 2500μs),这个字节的编程写入仍未成功,则判为芯片失效。

显然,快脉冲编程法的编程速率极高,一片 64K 字节的 EPROM27512 的编程时间只需要 8 秒钟左右!

二、安全智能编程法的提出

上述几种国内外常用的 EPROM 编程法大都是将直流编程高压直接加在芯片的 V_{PP} 引脚上。大量实践证明：当编程开关接通 V_{PP} 高压时，会产生一个持续时间约为 60μs、峰值比 V_{PP} 高出约 10V 的过冲电压，然而不少厂家制造的 EPROM 芯片的 I_{PP}～V_{PP} 特性不能忍受这个过冲电压，而导致芯片永久性击穿，这是 EPROM 编程废品率极高的根本原因。单片机 8751 片上 EPROM 编程时，则严格要求 V_{PP} 高压不得有超过 0.5V 的小毛刺。否则会烧毁单片机。

为了限制接通 V_{PP} 编程高压时产生的过冲电压的幅度，国内不少单位采用的措施主要有：

1. 降低 Vpp 电压编程：例如将 Vpp 降为 20～24V 对 EPROM2716 进行编程；

2. 芯片 Vpp 引脚与地(GND)间并接一个吸收电容(50～100μF)；

3. 用电子开关代替手动 Vpp 开关；

上述这些限制 Vpp 过冲电压的措施取得了一定的效果，使 EPROM 编程废品率大为下降，但烧毁芯片的现象仍时有发生，不能根除。

另外一般编程法未考虑不同厂家、不同工艺、不同芯片及不同存储单元的电气特性差异，千篇一律地规定为 50ms/字节的编程时间，必然导致“编程过头”或“编程不足”现象的产生，而且编程时间也十分冗长，效率太低。

智能编程法和快脉冲编程法具有编程时间合理和编程速率高的明显优点，但是需要将 Vcc 电压提高到 6V～6.25V，实践证明：在这种电压下，有些厂家生产的 EPROM 却十分容易烧毁。

由此可见，完全有必要研究一种适用于各种 EPROM 和 8751 的安全、可靠、快速的通用编程方法。

我们在大量实验研究的基础上，在硬件上增加了可靠的芯片安全保护措施；在软件方面吸取了智能编程法的优点，试验成功一种通用 EPROM 安全智能编程法。

三、芯片安全保护硬件措施

我们经过反复实验，主要采取以下两个芯片安全保护硬件措施：

1. 不提高 Vcc 电压，仍采用＋5V，以确保各种厂家的 EPROM 芯片均能安全编程。

2. 芯片 Vpp 引脚不直接加直流编程高压，而是通过脉冲放大器来加脉冲编程高压进行编程，其原理电路如图 1 所示. 这样，芯片的 Vpp 引脚上就不会有过冲电压。

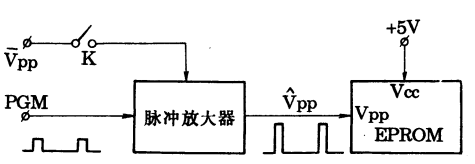


图 1

采用以上两种芯片保护措施之后，就能有效地避免 Vcc 击穿或 Vpp 击穿，芯片烧毁现象基本根除。

四、安全智能编程软件的设计考虑

我们在 Intel 智能编程法的基础上，在编程软件的设计上作了以下三点改进：

1. 采用通用 EPRG 编程软件，能对 8751 及各种 EPROM 进行编程，大大节省了软件开销；

2. 由于 Vcc 仍保持为＋5V，为了保证编程效果的长期可靠性，将“附加编程脉冲”由 3Xms 改为 6Xms；

3. 为了便于观察编程进程，采用每编程一个字节就显示一个字节的方式。

这种通用的安全智能编程法软件 EPRG 主流程图如图 2 所示. 采用 MCS－51 单片机作为主机的通用 EPRG 软件主程序清单请见附录。

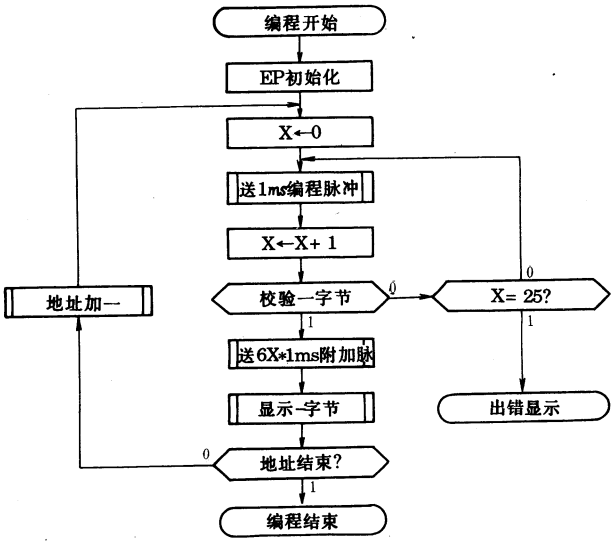


图 2

一年多来，安全智能编程法已经在 RICE 型通用微机开发系统、ROM 仿真型 MCS－51 单片机开发应用系统及通用 EPROM 编程器中实际使用。硬件保安措施能够确保编程时的芯片安全；而通用 EPROM 编程软件能够自动按 8751 单片机及各厂的 EPROM 2716，2732，2764，27128，27256，27512 的编程要求进行编程时间智能控制；比一般编程法能提高编程速率五倍左右，例如 EPROM2716 的编程时间只需要 15 秒左右。

实际使用结果证明：安全智能编程法具有安全、可靠和快速的优点。

附录：通用 EPROM 智能编程软件主程序清单

(8751 或 EPROM 通用)

```
1500H   51 F4      ACALL   12F4H   ;EP 初始化
1502H   75 F0 00   MOV     B,  #   ;X←0
                                00H
```

(下转 14 页)

学装 μp-80

手动 EPROM 写入器

易齐干

学装微电脑

编者按：本刊 90 年开始推出 μp－80 微电脑教育套件。今年将继续刊出有关 μp－80 的开发与应用资料及配套硬件，希望大家能参加这方面的自学。

μp－80 套件的应用开发，EPROM 写入器是不可缺少的。

本文介绍的手动 EPROM 写入器，与 μp－80 套件的电源部件、键盘部件和写读检制部件连接对 2716 或 2732EPROM 手动写入、读出操作。

1. EP－ROM2716、2732 引脚配置。

EP－ROM2716、2732 的引脚配置如图 1 所示。A_0～A_{11}：地址线；D_0～D_7：数据线；$\overline{CE}$：片选；V_{PP}：控制写入 25V 电源引脚；V_{CC}：5V 电源。

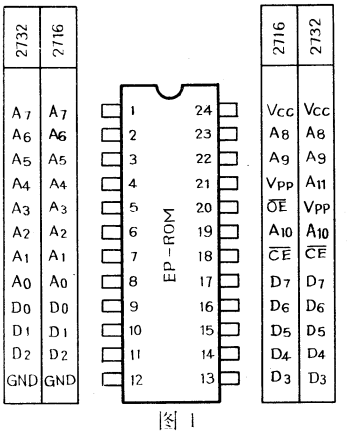


图 1

EPROM 写入

EPROM2716、2732 出厂时，内部所有的位均为 1，写入程序前请务必检查。

写入过程基本如下：

(2716)

(1)引脚㉑V_{PP}为 OFF(5V)，EPROM 为准备写入状态。

(2)引脚⑱$\overline{CE}$ 为"L"电平、引脚⑳$\overline{CS}$ 为"H"电平，接着引脚㉑V_{PP}为 ON(25V)状态。

(3)数据进入数据线。

(4)地址进入地址线。

(5)向引脚⑱施加 50ms 正脉冲

(6)电步骤 3. 重复，将程序全部写入。

(7)如果写入完毕，引脚㉑V_{PP} OFF(5V)之后，取下 EP－ROM2716。

(2732)

(1)引脚⑳V_{PP}为 OFF(OV)，EPROM 为准备写入状态。

(2)引脚⑱$\overline{CE}$ 为"H"电平，引脚⑳V_{PP}为 ON(25V)。

(3)数据进入数据线。

(4)地址进入地址线。

(5)向引脚⑱$\overline{CE}$ 施加 50ms 的负脉冲。

(6)由步骤 3. 重复，写入全部程序。

(7)如果写入完毕，引脚⑳V_{PP}为 OFF(OV)后，取下 EPROM。

如果写满全部地址，2716(2KB)大约需 100S、2732(4KB)约需 200S。

2. 手动写入器电路原理。

手动写入器电路原理图如图 2 所示。

(1)V_{PP}电源。

在写入过程中，需要向 V_{PP}引脚施加 25V 电压。目前，国内出售的 EPROM 写入器均采用外部提供 25V 直流电压，这样不方便使用。

该写入器仅使用 μp－80 套件的 5V 电源部件，通过 TL497 IC 芯片，即可获得 25V 直流电压。TL497IC 芯片外部电路简单，功能可靠。

TL497 的引脚⑬与⑩之间连接 120μH 电感线圈。

由于 2716 与 2732 的引脚⑳与㉑的区别，TL497 的输出引脚⑥经过晶体管放大后为 25V，由双刀开关 A 控制进入 EPROM 的 V_{PP}引脚。

2. $\overline{CE}$ 引脚的 50ms 脉冲。

采用 74LS123IC 与电阻、电容组合起来成为宽脉冲发生器，由 $\overline{Q}$ 引脚、Q 引脚分别提供 50ms 负脉冲、正脉冲。

引脚 1A 的触发脉冲由 μp80 的"写读控制部件"的"写入"按键提供。

(3)地址线的扩展。

μp－80 套件只有 A_0～A_7 8 条地址线。EPROM2716 的地址线为 A_0～A_{10}。EPROM2732 的地址线为 A_0～A_{11}，该写入器上添加一片 74LS193、一片 74LS126，来解决高位地址线。

该写入器的地址信息有复位功能和加 1 功能，即只能使用"写读控制部件"的"复位"按键和"写入"按键。要求引脚 CL 和引脚 CU 分别和"写读控制部件"的相应引脚连接。

(4)地址与数据显示。

地址信息显示由 12 支红色发光二极管代表 A_0～A_{11}。

数据信息显示由 8 支绿色发光二级管代表 D_0～D_7。

(5)读出检查。

对 EPROM2716 进行读出检查时，双刀开关 B 应搬向 2732 侧，反之开关 B 应搬向 2716 侧。

写入过程中，μp－80 套件的“键盘部件”、“写读控制部件”的操作不变，只是应该预先在“写读控制部件”印刷线路板上照图 3 所示，焊接三条跨接线。

有关手动 EPROM 套件的邮购事宜，见本期邮购消息栏。

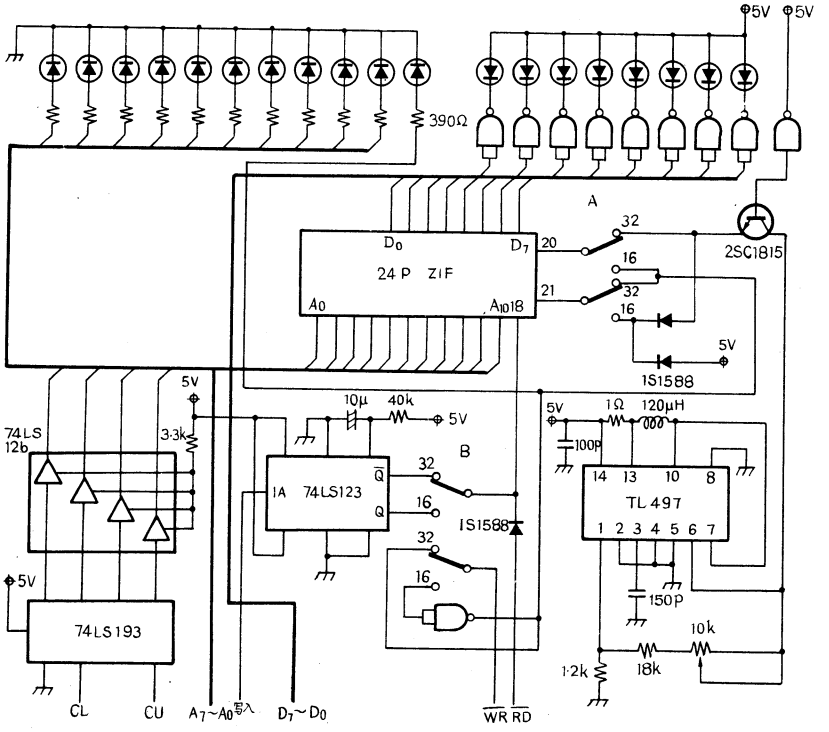


图 2

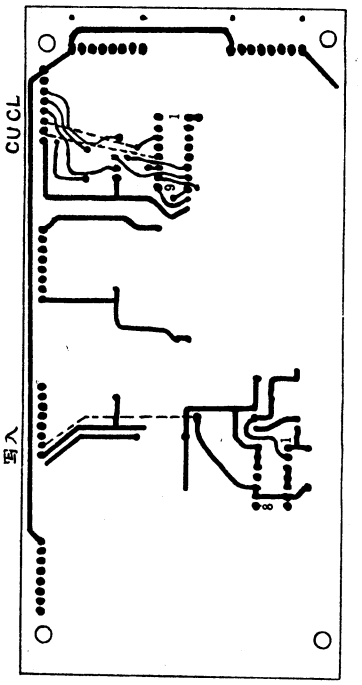


图 3

封底说明

我院庄志荣副教授发明的专利技术产品，用于 LASER310、200、500 等系列机庄氏万用 310、500 模块和用于夏普 PC—1500 等系列机。庄氏万用 PC—1500 模块使低档机更新换代，升级腾飞。其体积小、插接方便，与电视机、显示器相连，分辨率提高 6 倍，高分辨显示 192 个以上汉字，保持主机磁盘机、磁带机正常工作，直接按出 68 个功能字，保持全屏幕编辑，即光标在整个屏幕内上下左右移动，一个语句行内可写入 128 个以上字符，即 64 个以上汉字。模块自身可不断实现功能升级，其为主机扩充十几个系统和 250 多项功能，扩充 512KB 存贮空间，有 A/D、D/A 转换、异步串行通讯接口、自带打印卡，可同时配接两类不同性质的打印设备(点阵机与绘图机)。

用万用模块配接的 LASER310、PC—1500 等系列计算机，是融汉字、图形、通讯、声像合为一体的庄志荣全能中西文电脑。采用拼音、区位码、新字元组合、人机对话手写汉字等方法输入汉字，可造简体字、繁体字、书法字、美术字、空心字等；有国家标准点阵一、二级汉字及符号 8000 多个和万个以上矢量汉字；用拼音输入汉字，每个汉字可直接显示区位码，每个汉字可连续重复调用；键入一个国标区位码，一次便可同时显示 16 个相邻汉字，汉字可任意放大到 192 倍；称植扩展 IBM-PC 机 BASIC 语言、动画语言、绘图语言，独创动画绘图坐标语言，表格生成语言，汉字放大显示打印控制语文；首创与中华机、苹果机、IBM—PC 机兼容的标准汉化 LOGO 语言(MITLOGO)，可与结构 BASIC 语言混合使用，采用虚拟空间，在内存极小的环境下调用 8000 多个汉字和运行 LOGO 语言；增设 wordstar 系统；电子作图，画人物像，书写汉字、符号、美术字等等；异步串行通讯系统，实现与 IBM—PC 机连网通讯；首创声像同步合成系统，让计算机说话；在模块体内，设置 32KB、64KB、128KB、192KB 等类型的电子记录体，它象磁盘驱动器工作一样，实现数据文件、程序文件存取和格式化等工作。增设的 A/D，D/A 转换卡，开拓了新的应用领域。

万用 310 模块是 1988 年问世的，在 1988 年 3 月和 12 月初两次记者新闻发布会上，公布了以 LOGO 语言为核心，以高分辨显示国家标准一、二级汉字和矢量汉字为基础的万用 310 模块将以强大功能系统改造落后低档 LASER310、200 等系列计算机。

万用 310 模块发展很快，广泛应用在工业、农业、文教、卫生、医药、体育等等行业。庄氏多功能运动员心理素质测试仪、多功能 A/D、D/A 转换器、医疗器械、图声教育机等，被称为精湛的技术、卓越的功能、可靠的质量、低廉的价格，操作简便，使用方便，深受各行各业的欢迎。(感兴趣的读者，请汇 1.5 元邮资索取产品功能介绍。)

微电脑控制电饭锅

万国泰

装有微电脑的电饭锅具有“智能”，它能自动烧饭，不但烧出的饭质量好而且节能省时间。

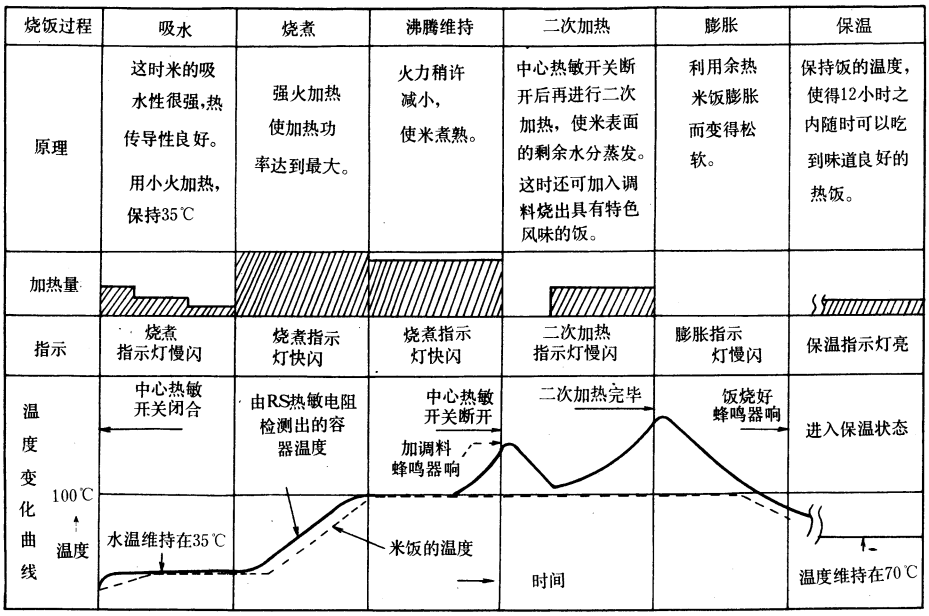
微电脑控制烧饭的原理是根据安装在电饭锅上的热敏温度传感器检测温度变化，自动调节加热功率使烧饭全过程处于最佳状态。

烧饭过程各阶段的耗电状况。米饭的温度变化均见图 1。在吸水阶段，电脑控制加热器的通电率使浸泡在水中的米维持在 35℃，让米粒充分吸水，经过一段时间后加热器才获得全电压进入烧煮阶段，从 35℃开始直至温度升到 100℃，电脑积算这段时间，根据这段时间的长短自动计算出锅内有多少米并确定最适当的加热控制。米煮成饭以后锅子底部的水干了，温度升至 100℃以上，锅中心的热敏开关转为断开，停止烧煮，然后进行二次加热，最后经过膨胀（俗称闷饭）阶段，饭就完全烧好了。以后是保温阶段，通常在 12 小时之内可以保持饭的品质不变。发生异常情况时温度一旦超出正常范围，超温传感器将使电脑发出停止加热的指令，并发出声光报警。

图 2 是原理框图。8035 的 P_1 口作为输出口，输出双向可控硅触发信号及 LED 指示灯信号，如果使用高效发光二极管的话可以不用驱动三极管，双向可控硅的触发信号要用光耦隔离。INT、T_0、T_1、P_{24}～P_{26}作输入口。在每个机器周期 8035 都要检测 INT、T_0、、T_1 这三个口的电平，并且有相应的转移指令可直接使用。超温信号经比较后注入 INT，正常情况下比较器输出高电平，温度超出正常范围时比较器输出转为低电平，CPU 切断加热电源并发出报警信号。温度极限由比较器中的微调电位器设定。T_0 口作测温用，555 时基电路在这里作为一个电阻/频率变换器，烧煮热敏电阻 RS 的阻值变化引起变换器输出频率的相应变化，CPU 据此求出实际温度。变换器应使用品质较好的阻容元件。T_1 口输入电网电压的过零信号用来控制双向可控硅的触发。P_{24}接中心热敏开关，这是一个属剩磁性热敏开关，温度升高至额定值断开。P_{25}接二次加热设定开关，这个双金属片热敏开关的动作值可调节，便于沸腾维持阶段以后在米饭中加入鸡丁与肉末等调料，经二次加热烧出具有风味特色的米饭。P_{26}接蜂鸣器按钮，按下按钮，沸腾保持阶段结束后蜂鸣器响，表示可以加调料了，再按一下则消声。整个硬件单元安装在锅体旁的胶木电气盒并与锅体保持一定的距离以免受高热。

控制装置的程序框图见图 3。程序采用 CPU 查询方式。在延时子程序中含有 INT＝0 时的转向指令及测温与测电压过零等操作指令，这样充分利用延时时间并确保延时时间内超温保护功能有效。在烧煮阶段 CPU 根据米饭温度的上升速率来计算米的多少是程序中的关键，不同容量的电饭锅即使烧煮相同米量的饭温升速率仍然不同，用理论方法推导比较困难因此可以用实验方法。在烧煮不同量米饭的时候测出实际温升速率，并用内插法排出一和张表放在程序第三页，CPU 将用查表方式根据温升速率查出米量从而确定以后的加热控制状况。

图1. 电脑控制烧饭程序



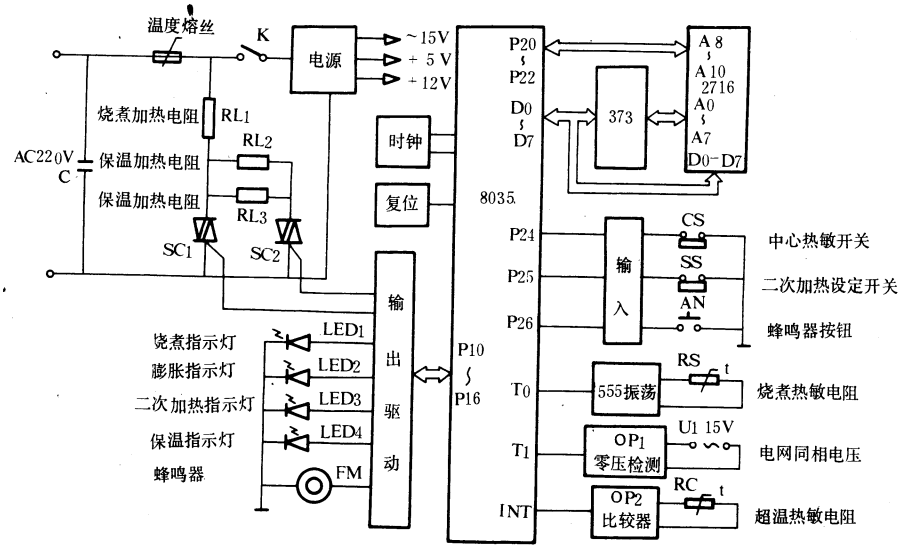


图2. 原理框图

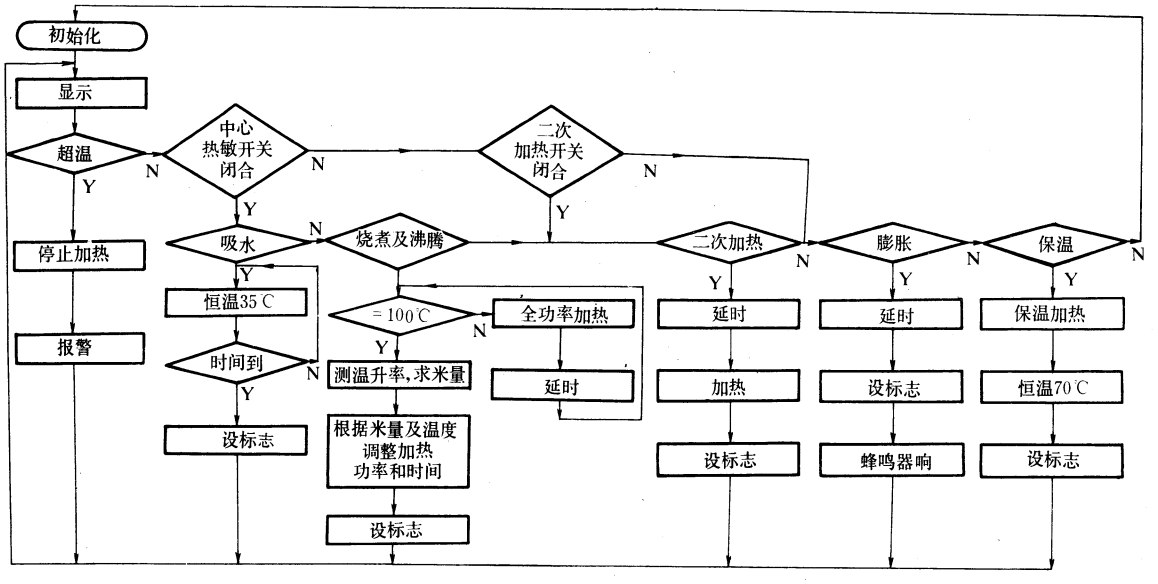
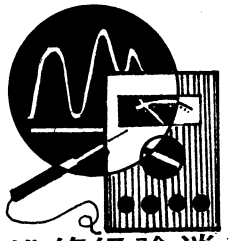


图 3 程序流程图

(上接第 8 页)

8088CPU 机上可以感觉出，尤其当文件存储时族比较离散时更加明显。所以应经常用 PCTOOLS 的整理文件功能，整理磁盘，使得一个文件在存储时族的分配尽可能的连续，这样可以提高读写速度。另外还应指出，虽然一般书上均认为当硬盘包含了子目录后，硬盘文件的数目不受目录区的限制，即文件数可以不受限制，实际它的数目受硬盘上族的数目限制，不能大于族的总数。

总之，根据以上分析，在 DOS 下族是文件在磁盘存储分配的最小单位。磁盘文件的最佳存储方法是提高族的数目，降低族包含的扇区数。即使在 8088CPU 类型机上也应装入 DOS 3.XX 版本。另外要尽量减少文件的数目，文件的长度尽量接近族长度的倍数。



DM－2214高分辨显示器无字符的检修及提高字符清晰度的研究

杨鼓行

本文分析DM－2214高分辨显示器无字符的故障修理及分析了字符边缘带色的原因，从而提高了字符的清晰度。

一、无字符显示的检修

DM－2214高分辨显示器是台湾最近几年的产品。现对其中一台的无字符显示故障进行分析，即打开显示器电源开关，显示器出现正常光栅，但与计算机连好后，打开计算机电源，显示器仍只有光栅。

一般的显示器几乎都没有线路图，而且厂家不同，线路的差别也较大，因此必须根据实物，画下有关部分的线路图、如图1所示，是红色通道的部分线路图、蓝色、绿色的线路与红色的一样。

由于无字符显示，故用示波器查找来自显示卡的三种颜色的脉冲信号。

脉冲在U_{103}(74LS05)的13腿是正常的，见图一，但在A点，则波形很小，以至于在R_{169}下端无波形。

再查蓝色通道(B)，绿色通道(G)两路在相当于R_{169}的下方处，也无波形，故屏幕上不会显示任何颜色的字符。

由于R_{169}下方无波形，故先分析Q_{108}，Q_{111}的工作点，光查A点电位，其电位为1.006伏，(B)点为1.7伏，似乎Q_{105}是正常的，(A)点电位正常时应是多少不知道，但初步印象似乎是低了一些，因为Q_{108}的基极在U_{103}(74LS05)为高电平时，接近5伏，而Q_{108}的集电极电位，却只有1.006伏，但从$U_{be}=0.6\sim0.7$伏来看，Q_{105}为正常，Q_{108}、Q_{111}的U_{be}也为0.6～0.7伏。

除了(A)点的电位偏低外，另一个不正常之处是R_{151}有51伏的压降，也就说，有40mA的电流流过R_{151}，进入哪儿呢？正常时，流过R_{151}的电流应流过Q_{105}到Q_{108}或Q_{111}，目前是否也是这样呢？下面断开Q_{105}，以判断一下40mA电流的去向，如果40mA电流是流过Q_{105}这个方向，则断开Q_{105}后，R_{151}上的51伏压降应消失，但断开Q_{105}后，R_{151}上仍有51伏的压降。

下面进而断开Q_{117}与Q_{120}，则51伏的压降消失了，查此两个管子、查出Q_{120}(B1109)的ec短路。Q_{120}是PNP管，进一步再查绿色通道(G)，发现相应的PNP管Q_{121}也短路烧坏。

因此经过R_{151}的电流不走Q_{107}而被Q_{120}短路，造成(A)点的电位只有1.006伏，(B)点的电位为1.7伏，由于(B)点的电位是三色共用的，故蓝色也被截止。

显示器无任何字符显示、排除了Q_{120}、Q_{121}的短路

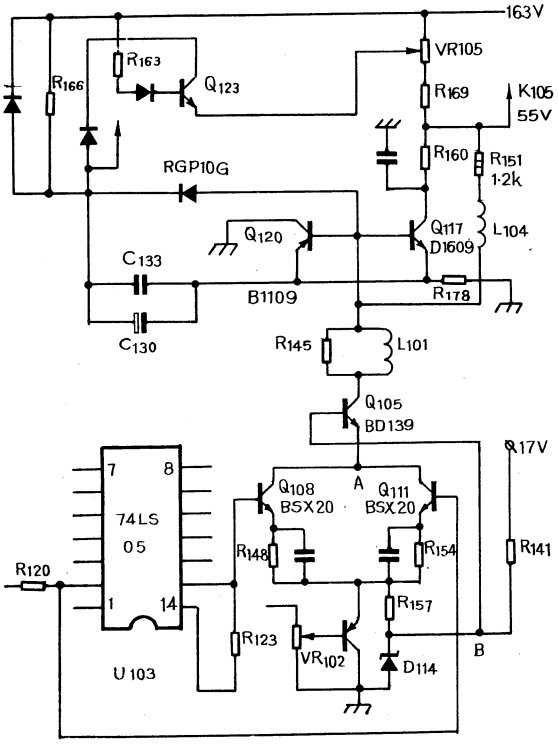


图1　Red通道的部分线路图

后，(A)点电位变为5.5伏，(B)点电位变为5.9伏，前面正说过，Q_{108}的基极电位，在U_{103}(74LS05)的12脚为高电平时，可能达到4.8伏左右，因此(A)点电位必须在5伏以上才合理，当用两个B546代换Q_{120}、Q_{121}后，显示器屏幕便有字符显示，但字符为蓝色字符，并有红色和绿色拖尾，其拖尾长度超过了3、4个字符的位置，并且R_{151}仍有5伏压降，也就是说，故障从无字符显示，变为有字符显示，但有严重的红色，绿以拖尾。

下面讨论一下R_{151}上的5伏压降来自何方，在绿色通道对应的这个电阻上也有5伏压降，蓝以通道对应的这个电阻上则无5伏压降，再次断开Q_{105}、断开后此电压降仍存在。那末此电流必定与Q_{117}这一路有关，但Q_{117}是好的。从电位上看：如图2所示，$U_{ce}=0$，则可能是R_C开路，则e的电位是R_1，三极管b_{1e}电阻与R_2的分压，而R_C正常时，e的电位是R_1，R_{be}与βR_2的分压，其中β为三极管的电流放大系数。

最后查出R_{160}(图1)烧断了，有4mA(5V/R_{151}=

5V/1,2K＝4mA)电流通过 Q_{117}基极,并通过 R_{178},实际检查表明,R_{160}电阻表面的漆已烧成黑色,电阻上的碳膜也已烧断,绿色通道对应的这个电阻也烧断,换掉烧断的电阻,R_{151}上 5 伏的压降便消失了,但字符仍有绿色与红色的长拖尾。

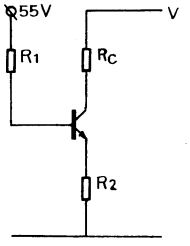


图 2　电位示意图

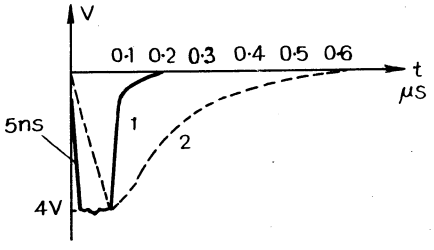


图 3　B117 基极波形

观察 Q_{117}基极的波形,如图 3 所示,Q_{120}如用蓝色通道上的 B1109 作为红色通道上的 Q_{120},则波形如图 3 中的 1,其前沿为 5ns,脉宽 100ns,后沿 25ns,如用 B546 作为 Q120,则波形为 2,前沿为 100ns,后沿长达 400ns,故造成红色与绿色有长达 4.5 个字符长的拖尾,从这分析可看到代替 Q_{120}的管子必须挑上升时间小,延迟时间小的管子。

综上所述此显示器无字符显示的故障来自 Q_{120}(红色通道)、Q_{121}(绿色通道)烧坏短路,来自 R_{160}(红色通道),R_{161}(绿以通道)的烧坏。

图 4　红色在字符边缘出现示意图

二、提高字符清晰度的研究。

由图 3 可以看到字符要清晰,蓝、绿、红三个通道的脉冲波形的上升时间和延迟时间较小,并且三色脉冲应尽量重合、则字符的边缘便清晰,无拖尾,若其中一色的后沿有些延迟,则出现该色滞后,在字符的左边出现该色,字符的边界不清晰。

试验过几种能购到的管子,如表 1 所示。

可以看出 B1109 为最佳,LM8550 上升时间,延迟时间,存贮时间均合格,但不足的是耐压只有 40 伏,从屏幕上的字符来看 LM8550 作为 Q_{120}时,字符相当清晰、无拖影和字符边缘出现其它颜色。但因耐压不够,不敢正式应用它,3CG8 的耐压刚好合格,但其上升时间和延迟时间均稍长了一点,从字符上看如图 4 所示,1 字在其右边的边缘上出现红色、当然如果绿色波形滞后了,便会出现绿色的边缘,因此在其它条件正常的情况下,字符如不清晰,字符右边有红色或绿色则是该色的波形的延迟,大概小于 10ns,如有拖影,则延迟时间太大,上升时间也太大,需更换合格的管子。

综上所述,字符要清晰,无蓝、红、绿色在字符的边缘出现,则三色波应尽量接近理想矩形波,并且三个波形应尽量重合,三个波形越接近矩形波,越重合在一起,则字符越清晰、越漂亮。

表 1　不同型号管子作为 Q_{120}时波形参数

管子型号	上升时间 ns	延迟时间 ns	后沿 ns	耐压 伏
PNP　B546	＞100		400	200
PNP　B940	＞100		400	200
PNP　B968	＞100		400	50
PNP　3CA1	20	20		80
PNP　LM8550	4	6		40
PNP　3CG8	6	10		50
PNP　B1109	4		25	160

计算机初级程序员（学习）水平考试辅导讲座

编者按：为提高我国初级计算机技术队伍的素质，给广大计算机初级工作人员提供自学和参加水平考试辅导的机会（见本刊90年12期《全国计算机软件水平考试初级程序员函授辅导班招生简章》）特设立本辅导讲座。由中国软件行业协会考试指导中心负责组稿和咨询应答工作，欢迎读者向本讲座提出问题和要求。来信请寄"中国软件行业协会考试指导中心"（北京学院路31号，邮编100083）。

问：水平考试的由来？

答：为适应我国电子信息产业发展的需要，中国软件行业协会考试中心在劳动部、机电部及有关省市劳动、人事部门的领导和支持下，自1988年起在北京及部分其他省市进行了初级计算机技术人员水平考试的试点工作。

问：水平考试是由何单位组织的？

答：是由中国软件行业协会考试中心组织的。

问：1988－1990年间有多少省市参加了初级程序员水平考试？

答：1988年有北京和湖北省参加了水平考试。1989年发展到包括有广东、辽宁、福建、云南、深圳等十二省市进行了联合考试。而1990年参加水平考试的十来个省市中已包括有澳门地区。

问：水平考试的意义和目的？

答：实行初级计算机技术人员水平考试，为有志于从事计算机软件工作的同志提供了一个检验自己知识水平和应用能力的机会，有利于激励他们不断学习、努力提高自己的技术业务水平；水平考试也为用人单位提供了一个客观依据，有利于正确录用人才和发现人才，有助于考核和正确使用在岗人员；同时，由于水平考试具有国际上的可比性，便于进行国际合作和人才交流。

问：初级程序员水平考试的对象是什么？

答：主要是各行各业中能在微型计算机上熟练操作、并能使用BASIC（C或汇编）语言和dBASE数据库作初级开发应用的技术工人、技术员或学生以及广大计算机的初学者。

问：参加初级程序员级水平考试的报名办法？

答：采用向全社会开放、自愿报名参加的办法。报名时间及报名费标准由各地自行决定。各地报名地址将由考试中心在《计算机世界》报上刊登公布（约在5、6月）。

问：水平考试的考试时间、考场和发证工作？

答：水平考试的考试时间在每年9月的第一个星期日（如90年为9月2日），考场由各地指定。

凡参加水平考试、成绩达到合格标准者，由中国软件行业协会考试中心统一颁发考试合格证明（各地可自行决定加发各地考委会或有关政府的合格证书）。

中国软件行业协会考试中心公布的九〇年初级程序员级水平考试大纲（试行）说明，北京的考试是在北京市劳动部门的领导下进行的，北京软件水平考试考委会将作为北京市劳动部门认可的一个工人考核点。凡考试合格又符合晋级条件者可由各单位根据此证书到劳动部门办理手续换取等级证书。

问：考试大纲准备改吗？何时公布？

答：中国软件行业协会考试中心正在酝酿考试大纲的修改方案。予计在91年2月公布。

问：《全国计算机软件人员水平考试初级程序员函授辅导班》的主办单位是谁？

答：中国软件行业协会考试指导中心。

问：《全国计算机应用软件人员水平考试函授、电视辅导班》的协办单位有哪些？

答：协办单位有：中央电视台、清华大学计算机系、清华大学出版社、北京大学计算机系、中国科技大学研究生院、北京信息工程学院、北京电脑天地学校、电子工业出版社等。

问：水平考试函授辅导班的办学宗旨是什么？

答：由考试指导中心主办的《全国计算机应用软件人员水平考试函授、电视辅导班》（以下简称《函授辅导班》）（第三期）是包含初程、程序员、高程和系统分析员等四个级别在内的面向全国的函授辅导班。今年对程序员和高程将通过中央电视台（二频道）进行辅导。

其中初级程序员班的办学宗旨是为提高我国初级软件人员的素质和帮助参加全国初级程序员级水平考试的同志准备应试。

问：初级程序员函授（面授）辅导班（第三期）的报名日期及办法？

答：报名日期：即日起至91年3月。

报名办法：填报名表（函索）、交二张一寸相片及报名费5元、学费（含教材费、函授费、邮费）90元，面授费由各面授点自定。对中学生自费者70元优惠（汇款附言请注"中学生优惠"），开学前寄出教材、学员证。

问：报名和汇款地点？

答：报名地点在北京学院路31号，中国软件行业协会考试指导中心，邮码100083，电话：2012233－322/529。

银行汇款：北京工商银行海淀区东升分理处，帐号：891377－07高档微机协会。

邮局汇款：北京学院路31号，中国软件行业协会高档微机协会（邮政编码：100083）。

问：采用何种教学方式？

答：主要通过《软件函授通讯》和《电子与电脑》月刊等资料进行函授教育，指导学员围绕全国软件水平考试大纲和辅导丛书进行自学；还将通过函授月刊定期向学员提供有关（水平考试）信息、解答共同性的疑难问题。在北京及学员集中的省市设面授辅导站利用假日进行面授辅导，对边远地区没有条件进行面授的学员，通过函授布置作业、解答疑难。

问：辅导内容有什么？

答：将围绕91年考试大纲进行，有计算机基础知识、微机操作系统、数据录入技术、BASIC、C和dBASE Ⅲ程序设计及应用等。

问：函授辅导与水平考试的关系应怎么看？

答：考试和辅导是相辅相成的，要考试就要搞好辅导；但搞辅导不能只为考试，应不限于考试，应杜绝猜题押宝，旨在努力提高学员的理论水平和实际工作能力。二年来的统计数字证明，函授辅导班的学员参加全国软件人员水平考试的合格率较高。

问：函授班结业考试的时间和方式？

答：函授班于《电子与电脑》杂志第7期刊登考题，进行开卷考试。凡学员按期寄回答卷，成绩合格者，统一由中国软件行业协会考试指导中心发给结业证书以及有奖学习纪念奖品。优秀学员将被送报参加有奖竞赛复赛。

问：函授辅导班主办单位——中国软件行业协会考试指导中心的组织归属和负责工作？

答：中国软件行业协会属机电部领导，考试指导中心是中国软件行业协会下属的全民性质的事业单位。考试指导中心受机电部计算机司委托负责协调组织全国计算机软件水平考试辅导工作。它从85年开始联合了清华大学、北京大学、中国科技大学研究生院、北京工业大学、北京师范大学、上海交通大学、南京大学、南京东南大学、北京电脑天地学校等院校的计算机专家编写了成套的软件水平考试辅导教材，组织了二期全国计算机软件水平考试函授辅导班。为了提高软件水平考试的科学性，考试指导中心与中国计算机学会普及工作委员会于90年十月底共同召集了全国各地关心我国软件事业发展的高等院校和计算机界同行专家、学者，召开了全国首届软件水平考试研讨会，对如何进一步改进和完善我国计算机应用软件人员水平（任职资格）考试提出了积极建议。

问：1990年《四通集团杯》全国初级人员有奖竞赛（以下简称有奖竞赛）是谁倡办的？其宗旨呢？

答：有奖竞赛是由中国软件行业协会考试指导中心、中国高档微机协会和《电子与电脑》杂志编辑部联合倡办的。

旨在鼓励广大初级计算机软件人员为振兴中华而勤奋学习和发现人才、培养人才，以提高我国软件人员的素质和壮大软件产业的队伍。热心于社会教育事业的四通集团公司等单位支持和赞助了九〇年的有奖竞赛。

问：去年有奖竞赛是怎样进行的？何时？何地？

答：九〇年有奖竞赛是在北京进行的。

竞赛分两轮进行。

1. 初赛：第一轮参赛者是函授辅导班学员（读者）。采用开卷笔试方式。试题公布在《电子与电脑》月刊第7期。凡九月一日前将答卷寄给中国软件行业协会考试指导中心而又成绩及格者均发结业证和纪念品。

2. 复赛：第二轮参赛者25名，他们是初赛合格者中的优秀选手和由各省市推荐参加了全国软件人员水平考试（初程级）的优胜者（应包括港、台等地区软件协会推荐的优秀选手）。

复赛包含笔试与操作比赛二部分，笔试在清华大学举行，操作在北京电脑天地学校举行。由国内著名计算机专家、教授等组成评奖委员会，由清华大学计算机系吴文虎教授担任评奖委员会主任委员。最后评出一等奖1名，二等奖3名，三等奖5名，四等奖8名，鼓励奖7名。

授奖大会11月24日在北京清华大学隆重举行。国内计算机界许多知名的专家、教授出席了授奖大会，四通集团公司副总裁、中国软件行业协会考试中心主任，电子工业出版社社长等分别将奖品授予优胜选手。

竞赛活动得到了社会各界的关注，新华社、光明日报、科技日报、计算机世界、中国计算机报、北京电视台等均到会采访并报道。这说明了全社会关心我国计算机软件事业的发展，关心我国软件人员的成长。

这次竞赛是首届，可能很多优秀选手还未被发现。此外，台湾软件协会也因时间仓促未及选拔选手参加这次竞赛（有关九〇年有奖竞赛的试题与答案详见《电子与电脑》杂志90年12期）。

问：91年继续举办《有奖竞赛》吗？

答：91年将继续举办《有奖竞赛》，而且将包含初级程序员和程序员两个级别。欢迎全国计算机初、中级软件人员都来踊跃参加有奖学习和竞赛。我们热忱期望今后有更多的企业关心我国计算机软件事业的发展，为共同培养和发掘我国计算机软件人才、提高我国计算机的应用水平而努力。

（中国软件行业协会考试指导中心供稿）

第一讲　计算机基础知识(上)

王　路　敬

问题 1　计算机系统是由哪两部分组成?

答:由下述两部分组成。

(1)硬件

硬件是指计算机系统中设备本身,是能够收集加工与处理数据及产生输出数据的各种固定装置的总称,硬件提供了处理数据的物质基础。具体地说,是由运算器、控制器、存储器、输入设备、输出设备组成了计算机的硬件系统。

运算器——直接完成各种算术运算和逻辑运算的装置。它是由电子线路构成。

计算机的运算有两大特点:其一简单,就本质而言计算机只会做最简单的二进制加法运算;其二　,每秒钟做上百万次、千万次运算已是很平常的事了。

、内存储器——存放数据和程序的装置。它具有记忆的能力,设置在计算机主机内。为使用方便将其划分为若干单元,每个单元为一个字节可存放一个八位二进制数。每个单元有一个编码,称之为地址码。地址码也是用二进制数表示的。例如 IBM PC 及其兼容机为 16 位机,其地址码一般有 20 位二进制位,可编码到 1024K 即 1M。

内存储器通常分随机存储器 RAM(Random Access Memory)和只读存储器 ROM(Read Only Memory)二种。RAM 是一种既可以从其读取代码,又可向其写入代码的存储器,是内存储器的主体。它的特点是打开电源时,其中没有有用数据,一旦写入数据,只要电源不切断且计算机处于正常工作状态,数据就能保持。断电后,全部数据消失。ROM 是一种只能从中读取代码,而不能以一般方法向其写入代码的存储器。不管关机或停电,里面的信息永远不变。一般 RAM 和 ROM 在内存储器中统一编址。RAM 处于低地址段,ROM 处于内存储器的最高地址段。IBM PC/XT 及其兼容机 512K 的 RAM 地址编码为 00000H～7FFFFH。

输入设备——向计算机送入数据、程序以及各种字符信息的设备。这种设备可以把数据和程序转换成电信号,以二进制代码输入到计算机里。计算机的输入设备很多,微型机的键盘就是一种标准的输入设备。键盘一般分成三个区:功能键区,标准打字机键盘区以及小键盘区。按功能分为专用字符键,控制键,编辑键。其工作原理是将所按键的扫描码用查表的方法转换成 ASCII 码,存入键盘缓冲区供使用。

输出设备——把计算机工作的中间结果或最后结果表示(打印或显示)出来的设备。这种设备把内存储器所存的内容按一定的转换规则变为相应的字符或汉字输出。微型计算机常用的输出设备为显示器和打印机。

控制器——计算机的指挥系统。控制器通过向机器的各个部分发出控制信号来指挥整个机器自动地、协调地进行工作。它是根据人事先编好的程序来进行工作的。计算机自动工作的过程实质上就是自动执行程序的过程。

运算器,内存储器和控制器称为计算机的主机,主机中的运算器和控制器称为中央处理器(CPU),把各种输入、输出设备,外存储器等统称为计算机的外围设备。

(2)软件

是指使用计算机所必备的各种程序的总称。它的任务是发挥和扩大机器的功能,以及提高机器的使用效率,便于用户掌握使用。计算机软件是由一些程序组成的,这些程序通常放在计算机存储器里,看不见,摸不着,相对硬设备来说它是软设备,即软件。

总的来说,软件主要包括两大类:

①系统软件——它是用于计算机的管理、维护、控制和运行,以及计算机程序的翻译、装入、编辑和运行的程序。包含有操作系统、语言编译系统和常用服务程序等。

②应用软件——指的是为方便某种应用,或解决某类问题(如科学计算、数据处理及实时控制等)所必须的各种程序。应用软件包和面向用户的程序设计语言等都属于应用软件。

计算机和别的设备不同,不只是需要有适当的硬设备和训练有素的操作人员使它得以运行,它还需要有存放在机器里的程序,告诉它要做什么,怎样进行运算。衡量一台计算机的功能、能力,不仅要根据硬件的性能,还要看计算机上配备的软件的丰富程度,即要把软、硬件综合起来衡量。所以,软件在计算机系统中的意义,不仅是不可缺少的一部分,而且软件的价值随着软件的发展已超过了硬件本身,软件在计算机系统中的地位越来越重要。用户购买计算机,既要购买计算机的各种硬件,又要购买使用计算机必不可少的软件。

问题 2　什么叫计算机的指令、指令系统和程序?

答:指挥计算机进行基本操作的命令称为指令。一台计算机能够识别的所有指令的组合称为指令系统。指令系统有简单的,有复杂的,大小不同,形式各异。它表示一台机器处理能力的高低。把要解决的问题按处理的步骤编成一条条指令,能够完成一定处理功能的指令集合称为程序。

问题 3　显示器怎样分类?显示方式有哪些?显示标准有哪些?

答：显示器是将计算机中的信息在屏幕上显示出来，是微型计算机标准输出设备之一。

显示器按色彩分单色和彩色；按分辨率可分为低、中和高分辨率显示器。显示方式分为字符和图形显示。

①字符显示：屏幕显示 40 列×25 行或 80 列×25 行；字符区点阵 8×8，字符点阵 5×7 或 7×9；可进行黑白反相、闪烁、高亮度显示，彩色时有前景 16 色，背景 8 色；显示缓冲区可存放 8 帧(40×25)或 4 帧(80×25)的显示信息。

②图形方式：中分辨率时屏幕显示 320×200 点或 40×25 个字符，4 色；高分辨率时屏幕显示 640×200 点或 80×25 个字符，两色；显示缓冲区可存放一帧图形信息。

按显示标准分：

(1)有 CGA—— 彩色图形显示标准。IBM－PC/XT，PC/AT 显示系统就是采用这种显示标准。这种显示标准的图形分辨率有：

320×200

(2)EGA—— 增强型图形显示标准。例如，这种显示标准的图形分辨率有：

320×200

640×350 等。

(3)VGA—— 图形显示标准。IBM PS/2－50、60、80 型微机显示系统均采用这种显示标准。这种显示标准的图形分辨率有：

320×200

640×350

640×480 等。

(4)CEGA—— 中文增强型显示标准。新一代长城系列机 GW0520DH，GW286B，GW286BH，GW286EX 以及 GW386 均采用这种显示标准。这种显示标准的图形分辨率有：

320×200

640×350

640×480 等。

(5)CMGA—— 中文单色多灰度显示标准。长城 GW0520EM 就是采用这种显示标准。这种显示标准的图形分辨率有：

320×200

640×480

720×350 等。

(6)014＋015—— 长城系列中文彩色显示标准。长城 GW0520CH、长城 GW286 采用这种显示标准。这种显示标准的图形分辨率如：

长城 GW014　单色：640×504

彩色：640×480(8 色)

其中 014 是 GW0520CH 以及 GW286 机中高分辨显示适配卡编号 GW－014 的简称。015 为中分辨显示适配卡 GW－015 的简称。

CGA、EGA、VGA 为国际流行的显示标准。CEGA 是使长城系列机更加靠近国际流行的标准。

问题 4　微机中最常用的外存储器有哪几种？

答：微机系统中软盘和硬盘两种磁盘存储器是微机中最常用的外存储器。从尺寸上有 5.25 英寸、3.5 英寸软盘；面数有单面和双面两种；其密度有单密度、双密度和高密三种。软盘的存储容量，视软盘种类不同而异。例如 5.25 英寸软盘有 160KB、180KB、320KB、360KB、1.2MB；3.5 英寸软盘有 720KB、1.44MB 等。

使用软盘时必须考虑软盘与软盘驱动器的兼容性。具体规定：

(1)160KB/180KB 单面 5.25 英寸驱动器

可以在这种驱动器上读写的软盘片只有 160KB/180KB 单面、倍密度一种盘片。

(2)320KB/360KB 双面 5.25 英寸驱动器

可以在这种驱动器上读写的软盘片有 160KB/180KB 单面、倍密度和 320KB/360KB 双面、倍密度两种盘片。

(3)1.2MB 高密度 5.25 英寸驱动器

可以在这种驱动器上读写的盘片有：160KB/180KB 单面、倍密度，320KB/360KB 双面、倍密度和 1.2MB高密度三种盘片。

(4)720KB 双面 3.5 英寸驱动器

可以在这种驱动器上读写的只有 720KB 双面盘片。

(5)1.44MB 双面 3.5 英寸驱动器

可以在这种驱动器上读写的有 720KB 双面和 1.44MB双面两种盘片。

所以，在使用软盘时，高密度的软盘片如果不是空白盘，只可放在高密度软盘驱动器上读或写，而普通密度的盘片既可在普通密度的软盘驱动器上读或写，也可以在高密度驱动器上读或写。但一般普通密度的软盘片不要当高密度软盘片用。

IBM PC、长城系列及其兼容机的硬盘为 5.25 英寸，IBM PS/2 系列机是 3.5 英寸的硬盘。其存储容量最常见的 10MB、20MB、30MB、40MB、80MB、90MB、110MB 等。一个硬盘的存储容量＝磁头数×道数×每道扇数×每扇的字节数。硬盘是微机系统重要的硬件资源，为保护硬盘，在关机之前应先做固定磁头的工作。

问题 5　软盘片外框上的一个矩形缺口的作用是什么？

答：在软盘片右边有一个缺口，该缺口为读写缺口。如果盘上记有重要信息，为了防止因误操作而破坏这些信息，此时希望整个盘片只许读，不允许写就要用胶纸把缺口封住，这就达到了写保护的目的。

问题 6　打印机分哪两大类？有哪几种打印方式？

答：打印机是微型计算机系统的主要输出设备之

一。在微型机系统中，打印机是作为一个独立的部件与主机分离存在的。主机中都含有一个或多个打印机接口，这种接口多采用并行方式传送数据，即用八根数据线每次将一个字节的数据同时送出。计算机以中断或查询方式控制着打印机的动作，打印机服务程序是作为操作系统的一个组成部分常驻于主机内存中。

打印机分击打式和非击打式两大类。一般使用的大多是击打式打印机，它的打印头由若干根针组成，常用的有 9 针、24 针等。通过打印驱动程序控制各个不同位置的针动作或不动作，打印出各种字符或图形。

象屏幕显示器一样，打印机在微型机系统中的工作方式，就其接受来自主机数据信号类型的不同，也可分为字符方式和图形方式。

所谓字符方式，是指主机在发送打印数据时，只传送字符的 ASCII 码，而字符的形状是从装于打印机内部的只读存储器(ROM)中发出的。汉字的打印也可以在字符方式下进行，这要以打印机内部具备全部汉字字模为前提。字符方式可以获得较快的打印速度，是当前西文打印中最常用的方法，中文打印如采用这种方式，打印机的成本就要相应提高。

在图形方式下，主机所传送的不是字符代码，而是经过软件编辑的图形象素的电信号。图形方式既可以打印西文字符，也可以打印汉字字形或任意形态的图形。主机所输出的西文字符和汉字字形的图形信号，其字模都要在主机中存储着。此时字模不仅能存储于只读存储器 ROM 中，也可存储于随机存储器 RAM 或磁盘存储器中。图形方式可以打印出丰富多彩的字形和任意形态的图形，但它要以降低打印速度为代价。

在现代的微型机系统中，上述两种打印方式往往是共存的，到底使用哪一种要视具体情况而定。有时，用户可用键盘输入命令或通过程序中给定的指令来选择其一；有时由系统规定而不能改变。

求答

• 何处有售关于 MZ－731 程序集

江苏苏州类门外苏州交频瓷厂

(215001)汪波

• 请告知 COMX PL－80 四色打印机引线功能。

安徽马鞍山市东风村 48 栋 3 号

(243021)刘德超

• 本人有一台 NF－500A(H－01)电脑，我很想扩展其功能，希望有经验的朋友提供资料和帮助。

辽宁大连市碧流河水库发电厂

(116221)张国俊

山东莱芜第一纺织机械厂

(271100)田映华

电子工业出版社软件部向社会各界致意

电子工业出版社软件部专心致力于计算机软件出版事业，以促进软件产品的推广与应用，充分发挥其社会效益与经济效益，保护软件开发者的合法权益。

本软件部独家承担出版国家教委、机电部和中国科协所属的“全国中华学习机教育软件评审委员会”评审通过的辅助教学软件。

本软件部向社会广泛征集有推广应用价值的各种微型计算机软件，征集范围包括系统工具类、应用类、辅助教学类、幼儿教育类和益智性游戏类。应征软件稿包括不加密的软件母盘和使用说明书。经本部评审合格的软件，按国家出版署 的规定，给予标准书号，颁发出版证书，实行版权保护，通过新华书店向全国发行；同时也自办发行，在大中城市设有软件经销点。

本软件部出版的软件品种丰富，质量可靠提供售后服务。

本软件部愿同海内外同仁友好合作，跟踪软件技术发展成果，竭诚为社会各界服务！

欢迎各地软件开发者涌跃投稿！

欢迎广大软件用户订购！

欢迎各界朋友合作！

电子工业出版社软件部

地址：北京 173 信箱

邮码：100036

电话：815342

联系人：张丽华　吕　迈　徐海波

开户银行：北京工商行翠微路分理处

户名：电子工业出版社杂志编辑部

帐号：891333－59

软件目录

软件名称	适用机型	盘片数	售价
CTE 通用汉字表格编辑器	PC	2	200 元
打印报表程序自动生成工具	PC	1	260 元
LD 磁盘文件查询	PC	1	50 元
模拟电子技术教学及试题习题库	PC	6	500 元
汉化 FOXBASE 2.1	PC	3	350 元
单盘五笔字形	PC	1	95 元
25 行 dBASE+	PC	2	400 元
中华学习机集成工具软件 V1.0 版	CEC－I	1	90 元
新型积木式工具软件	CEC－I	4	145 元
中华学习机图文工具	CEC－I	2	140 元
CEC－I 文件编辑系统	CEC－I	2	180 元
CEC－I DOS 系统	CEC－I	2	140 元
三维图形制作编辑系统	CEC－I	2	100 元
中华学习机工具箱	CEC－I	1	25 元
ESS4.0 新概念英语	CEC－I	6	80 元

新书与软件

ORACLE 关系数据库管理系统

王学民　魏　国

ORACLE 数据库管理系统是美国 ORACLE 公司的产品，是目前世界上最流行的数据库管理系统。ORACLE1.0 于 1978 年推出，目前市场上流行的是第六版。国内大量引进 ORACLE 还是近两年的事，一些大学，企事业单位已开始研究和使用 ORACLE。

一、ORACLE 的特点

ORACLE 之所以流行开来，是因为它有如下优点：

1.1 广泛的适应性

ORACLE 支持各种类型的大中型机、小型机和微机系统，如可在大中型机 IBM 系统/370、4300、30××系列机上运行，可在小型机 DEC　PDP-11、DEC VAX-11、8600、8800、AT&T　、3B5、3B15、3B20、APOLLO、DOMAIN　、DGMV 系列机，HP 9000/500、HONEWELL　DPS-6、IBM 系统/88、Sperry5000、7000 等上运行，可在微型机如 IBM PC/XT、PC/AT、RTPC、AT&T、PC6300、7300、3B2、3B1、DEC Rainbow、MICRO-VAX、SUN(所有型号)WANG PC 、NCR 、TOWER 等上运行。

ORACLE 可用于多种操作系统，如 VM/CMS MVS UTS、UNIX、VMS、ULTRIX、RSX-11M^{+}、Aegis AOS/VS DG/VX、HP/VX、GCOS、VOS、DOS、XENIX 、AIX 等。

ORACLE 是众多关系数据库管理系统中唯一具有很宽范围的硬件兼容性和操作系统独立性的系统。

1.2 很好的兼容性

ORACLE 与 DB2、DB3、SQL/DS 完全兼容，能够直接使用 IBM 现有数据库系统的数据和应用程序。

1.3 较强的可移植性

ORACLE 为不同机型、不同操作系统上的用户提供一个以标准数据库语言 SQL 为基础的、统一的软件环境，具有相同的用户界面，用户可以从一个机器环境将应用项目移到另一完全不同的机器环境。这样，用户在更换或升级系统时都不会影响已开发的软件，达到源程序一级完全兼容，保证了应用软件的可移植性。

1.4 采用标准的 SQL

1976 年 IBM 公司的 E. F. Codd 博士发表了关系数据库 System　R 的基本概貌及其 SQL 语言。参照这个蓝图，第一个以 SQL 为基础的商品化关系型 DBMS 于 1979 年由 ORACLE 公司在小型机 PDP-11 上推出，并取名为 ORACLE。

SQL 已被 ANSI(美国国家标准局)所采纳，SQL 是完善的关系数据库语言，它建立在关系模型之上。

ORACLE 用户通过 SQL 语言可以完成数据定义，数据操纵和数据控制。ORACLE 提供给用户的一切工具都是以 SQL 为基础的。

1.5 提供一体化的数据字典

ORACLE 支持一个基于关系模型的动态联机目录(又叫字典)。字典定义关系的列，关系本身和关系视图，它也包含了从实表到单个数据项的详细解释。字典维护数据和系统用户的保密和控制信息。各个授权用户都可对数据字典进行查询，就如同查询一般数据一样，每当 ORACLE 数据库中发生表的增、删、改及授权变化时，系统都能自动对数据字典表进行相应更新。

1.6 采用先进的技术

当出现程序错误或系统故障时，数据库可以恢复到错误故障前的状态。它是借助于前象文件和后象文件表实现的。

ORACLE 采用按级封锁技术，用来保证多用户的并行存取和数据的完整性和一致性。ORACLE 能自动发现并解决死锁。

1.7 提供了丰富的工具

ORACLE 公司十分重视改善用户的应用环境，对此花了大量的精力。ORACLE 具有比任何其它数据管理系统都丰富的面向最终用户的特性。

ORACLE 为用户提供一套包括格式化处理，菜单管理，表格图形，报表生成等第四代语言工具在内的支撑工具环境，另外还有很好的决策支持系统。

ORACLE 可通过三种类型的接口将数据库相互连接：

1. EASY *，面向初学者和不熟练用户。

2. SQL * 命令接口，面向熟练的经过培训的用户及系统专业人员。

3. PRO * 编程接口，面向使用高级语言的程序调用或预编译的 MIS 专业人员，每种接口都产生 ORACLE 处理的 SQL 查询。

ORACLE 系统提供了全面的决策支持产品，包括 SQL * Graph(图形软件)；SQL * Cale(电子报表)Easy * SQL(利用光标定量选择技术的 SQL 语句生成器)。

二、SQL 语言简介

SQL 分为数据定义语言(DDL)，数据操纵语言(DML)，查询语言和数据控制语言。

1. 数据定义：数据定义包括建表、建索引、建视图三类，由 CREATE 命令实现。

例：建立一个部门表 DEPT 的命令如下：

CREATE TABLE DEPT

(DEPTNO NUMBER(2) NOT NULL,部门号

DNAME CHAR(14),部门名

LOC CHAR(12));地点

建完表后,系统为该表分配初始数据空间,同时也可对表的列的定义进行修改或增加新列,建立索引的目的是为了提高检索速度。索引为ORACLE提供数据库存取的候选路径、系统中的优化器,能有效地利用索引改进查询性能建索引命令CREATE、INDEX

视图(又叫虚表),是由基表派生出来的逻辑表,在目标数据库中并未实际存储它,只是在字典表中汇有形成该视图的SQL命令。当访问该视图时,系统将自动执行相应的SQL命令,建立视图命令格式为CREAT VIEW〈视图名〉AS〈检索块〉。

视图可象基本表一样来操作,但通常仅限于对由一个表中的直接的行和列的子集所构成的视图进行修改。

2.数据操纵

数据操纵指对表中的数据进行插入、删除更新操作,命令分别是INSERT、DELETE和UPDATE。

3.数据查询

数据查询命令SELECT是SQL功能最丰富,使用频率最高的命令,其格式也最复杂。

SELECT的命令格式为

SELECT〈列名〉FROM〈表名〉WHERE〈条件〉

ORACLE支持255个不同判断条件的单个数据库查询,返回的数据可在255个不同的分类字段上分类。

用户可以实现连接查询,根据指定的查询条件,联接两个或多个表,检索出多个表的数据。

此外,用户还可以对查询结果进行排序和分组,可以进嵌套子查询,还可以查询进行优化。

4.数据控制

通知数据控制命令可以建立新用户,对用户进行授权,保证数据的安全性。

三、ORACLE产品

除了RDBMS之外,为了方便应用软件的开发和不断开拓应用范围。ORACLE公司还陆续推出了具有第四代语言特征的各种开发工具和组合软件。这不但方便了软件的开发,而且保证了软件的质量。

3.1 SQL * PLUS

SQL * PLUS是ORACLE交互命令的驱动接口,它接收用户键入的SQL语句,并把执行结果显示在屏幕上,适用于进行定义、查询、产生简单格式报表等操作,还可以建立用户,授予或撤消权限等。

3.2 SQL * FORMS

是一个交互式、多用户,基于格式的应用开发工具。它提供全屏幕描述工具来设计屏幕应用,用户可通过生成的应用在设计好的屏幕格式上进行数据的录入,实例查询、修改、删除等操作,交互式屏幕设计程序使用户可以自由地设计屏幕和格式中数据项的位置。SQL * FORMS可以作为一个原型开发工具来使用。

3.3 SQL * Cale

是一个集电子数据表,数据库、制图为一体的多功能组合软件,与LOTUS 1-2-3完全兼容,可对ORACLE RDBMS数据库进行存取、维护,它易学易用,是决策支持者的得力助手。

3.4 SQL * Graph

SQL * Graph是个图形接口,能把数据信息图表化,用户输入SQL查询语句后,输入图形显示方式,就可把查询结果以圆图、直方图、线图方式显示在屏幕上或输出到绘图机。

3.5 SQL * MENU

是一个动态菜单程序,使程序设计者能对任何软件产品构造一个用户良好的菜单接口。所谓"动态",是指用户可在任何时候从菜单撤掉某些可选项,也即在一个系统里随时装入新的模块或取消已有的模块。

3.6 SQL * NET

是一种独立于机器的交互式通信软件,它允许机器在同等基础上相互通信。依靠SQL * NET可以在一个特定的计算机网络内或在系统网络之间通信。

3.7 SQL * Report

是一个ORACLE报表生成工具。它将正文格式化功能和SQL查询功能结合起来,用户可以使用数据库中已有的信息以及其它任何希望出现的正文来生成各种各样的文档报告。

3.8 PRO * 主语言接口

PRO * 主语言产品包括对所选的高级语言处理程序的一系列综合编译程序,目前可用的有C,COBOL,FORTRAN,PASCAL,PL/1,BASIC,ADA等高级语言。在高级语言程序里嵌入SQL语句,使得用户可以利用传统的程序设计语言和SQL语言各自的优势进行应用开发。

基于ORACLE数据库上述优点及其配套软件之齐全。可以想见,ORACLE数据库也将会在我国得到广泛的使用,并加速我国信息化社会的建设。

(上接第21页)

```
asc1=asc(hz)
asc2=asc(subs(hz,2,1))
jnm1=subs(hex,int(asc1/16)+1,1)
jnm2=subs(hex,mod(asc1,16)+1,1)
jnm3=subs(hex,int(asc2/16)+1,1)
jnm4=subs(hex,mod(asc2,16)+1,1)
qh=asc1-160
wh=asc2-160
qwm1=iif(qh<10,subs(str(100+qh,3),2,2),str(qh,2))
qwm2=iif(wh<10,subs(str(100+wh,3),2,2),str(wh,2))
@ 13,12 say '16进制机内码:' +jnm1+jnm2+jnm3+jnm4
@ 14,12 say '区位码:' +qwm1+qwm2
return
```

普及型 PC 机个人用户软件交流联谊活动

目前，市场上普及型 PC 机较多，如海华 PC、北方 PC、新潮 PC 等，由于各公司产品的硬件，特别是显示卡的设计并非完全一致，因而以上各型之间，普及型 PC 与 IBM PC 之间，普及型 PC 与 0520 之间软件并不完全兼容，为了使 PC 个人用户可以较方便的获得应用软件，不断提高自己的应用水平。本刊应读者要求特组织“普及型 PC 机个人用户软件交流联谊活动，联谊点设在农科院计算中心培训部。

宗旨：交流 PC 软件，共同开发，共享成果。

成员的责任和义务：申请加入联谊活动的读者，需填表登记，即为联谊活动成员。成员有责任向联谊点无偿提供软件，也可无偿从联谊点交换所需软件（不带盘片的读者收取盘片费和每片 2 元的拷盘费）。外地读者另收邮挂费 5 元（10 片以内）。

参加联谊活动的成员，不得将所交换的软件，用于经营性活动。暂不交流游戏类软件。

登记表如下：（可用此表或复制件）

<table>
<tr><td>姓　名</td><td>性别</td><td>年　龄</td><td colspan="2">联系地址及电话（邮编）</td><td>职　业</td><td rowspan="3">照片</td></tr>
<tr><td></td><td></td><td></td><td colspan="2"></td><td></td></tr>
<tr><td colspan="2">主机型号</td><td colspan="2">显示器（彩色、单色、分辨率）</td><td>有硬盘否</td><td>软驱（几个）</td></tr>
<tr><td colspan="2"></td><td colspan="2"></td><td></td><td></td><td></td></tr>
<tr><td colspan="3">已有软件</td><td colspan="4">还需要什么软件</td></tr>
<tr><td colspan="3"></td><td colspan="4"></td></tr>
<tr><td colspan="7">您在使用中，有什么困难和要求</td></tr>
<tr><td colspan="7"></td></tr>
</table>

联系地址：北京海淀区白石桥路 30 号中国农业科学院计算中心培训部。

邮政编码　100081　　电话　896531——241　　联系人　王路敬。

電子與電腦 2

一九九一年

总期第71期

電子與電腦

ELECTRONICS AND COMPUTERS

目 录

·综述·

PAL 和 GAL …… 朱世鸿(2)

·PC 用户·

通用打印机状态检测程序 …… 夏伟文(3)

dBASE Ⅲ 与 BASICA 之间的数据传送问题 …… 严桂兰(5)

dBASE 运用经验与技巧 …… 孙 鹏(7)

如何解决在运行 FOXBASE 内存不够的问题 …… 周日初(9)

COMPRESS 在硬盘集中存储中的应用 …… 许 鹰(9)

怎样在 IBM－PC 机上实现汉字的动态显示 …… 李晓华(10)

DOS I/O 重定向、管道功能及其使用 …… 徐国辉(11)

在低版本 DOS 下安装虚拟盘 …… 周健勇(12)

五笔 CCDOS 中一字节修正 …… 吴邦忠(12)

一个快速实用的字串替换程序 …… 金林樵(12)

PC 机磁盘文件的加密与解密 …… 刘忠达(13)

硬盘正确格式化一例 …… 袁士彬(14)

·学习机之友·

程序出错处理与 RESUME 的功能扩充 …… 丁志伟(15)

特殊 6502 指令 …… 李 铁(17)

四舍六入法程序 …… 韦肖鸿(17)

“怪”指令在软件加密中的应用 …… 傅 剑(18)

双显示电子钟 …… 万 斌(19)

苹果机汉字造型表自动生成程序 …… 刘士明(20)

高效率的排序方法 …… 陈庆祥(21)

演示抛物面的光学效应 …… 王晓林(23)

LASER QZT/C 应用经验 …… 张海翔(24)

·C 语言初阶·

第二讲 Turbo C 基本数据 …… 李文兵(25)

·学用单片机·

普及型单片机开发工具的硬件设计 …… 张培仁 刘振安(29)

·学装微电脑·

40 支开关输入部件的制作 …… 易齐干(34)

·电脑巧开发·

MP－I 汉字及图形显示装置 …… 陈名则(37)

APPLE－Ⅱ彩灯自控电路 …… 张孝玖(38)

·维修经验谈·

王安机 RAM 故障简易检修法 …… 董纯坚(40)

IBM 286 维修一例 …… 叶志斌(41)

中华学习机典型故障一例的分析 …… 杨瑞华(42)

APPLE Ⅱ主机常见故障的检修 …… 陈 鸿(42)

·初级程序员级水平考试辅导问答·

计算机基础知识(下) …… 王路敬(43)

·新书与软件·

Turbo CAD 软件简介 …… (46)

·游戏通·

巧用电脑作幻方 …… 郭高惠(47)

·读者联谊·

湖北地区中华学习机联谊活动情况 …… (48)

·信息与服务· …… (48)

·封二、封三· 6502 指令表 …… 李 铁

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:北京市邮政局

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:张 丽

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

PAL 和 GAL

朱世鸿

微电子技术和计算机技术是一对孪生姐妹。自1959年美国德克萨斯公司和仙童公司研制出在半导体硅片上制造出有特定功能的电子线路，即目前统称的集成电路(IC—Integrated Circuit)之后，微电子技术的高速发展超出了人们的预料，使现在的计算机科学技术每隔5～10年就有一次重大突破，机器的工作速度提高了10倍，而体积和成本却缩小10倍。目前，相当一台早期庞大计算机功能的电子线路，可全部集成在面积仅为几个 mm^2 的硅片上(见本刊91年第1期封面图片)。

构成计算机的IC从59年至今，经历了一个从采用小规模IC(SSI)→中规模IC(MSI)→大规模IC(LSI)→甚大规模IC(VLSI)→超大规模IC(SLSI)的过程。随着半导体工艺VLSI和SLSI技术的不断成熟，自然就可将各种越来越复杂的电子线路集成到硅片上，从而出现了各种专用IC即(ASIC—Application Specific IC。

关于ASIC目前尚无统一严格的定义，通常，有全定制和半定制二种之分。

全定制电路是厂家根据用户的具体要求而单独设计的某种特定功能的电路。一般凡大量生产的产品，例如通用的微处理器Z80,Z8000,8086,MCS51等均采用此方法设计。

半定制电路是在芯片上设计制作好一些元件和元件组，用户仅只考虑电路逻辑功能和各功能模块之间的连接就可以了，因此，人们对半定制的ASIC产生了极大的兴趣。

半定制ASIC按其内部结构可分为门阵列(Gate Array)、标准单元(Standard Cell)和可编程逻辑器件(PLD—Programmable Logic Device)。

(1)门阵列是在硅片上制成标准逻辑门的矩阵，是不封装的半成品，生产厂家根据用户设计要求选择合适的阵列规模，在阵列中制作出互连的图案(码点)进行定制，最后封装为成品售给用户。

(2)标准单元由一个预先定义好的功能单元库组成，它包括微处理器和存储器阵列，甚至包括线性电路，功放电路等。用户可从标准单元库中选取合适的逻辑块，对逻辑电路具有可优化的充分条件，因而设计自由度高。

上述两种ASIC电路均要由用户向生产厂家提出逻辑功能进行定做，用户无法自行开发。

(3)PLD又分可编程只读存储器(PROM、EPROM、E^2PROM)、可编程逻辑阵列(PLA—Programmable Logic Arrag)以及可编程逻辑阵列(PAL—Programmable Arrag Logic)和可编程通用阵列逻辑(GAL—Generic Arrag Logic)。

有关PROM、EPROM和E^2PROM的知识和用法，大家都比较熟悉。它们在计算机中一般多用来存储监控程序和表格常数。而PLA器件其基本结构与可编程只读存储器相似，在功能上又有区别，PLA器件可通过编程方法实现其一定的逻辑功能，可以单独使用，因而在实现各种逻辑功能上具有灵活性和较强的功能，同其它可编程器件一样，若阵列的连线采用掩膜方式制成，称工厂编程的PLA，若阵列连线由用户烧制(可熔断丝)而成称现场可编程逻辑阵列FPLA。

随着电子技术，计算机技术对IC的要求，又开发出集中了具PLA编程和PROM的低成本等于一身的PAL器件。它以可编程的灵活性、可加密、传输速度快、降低器件总功耗和减少外围芯片数、减小电路板面积和加速系统设计、提高可靠性等众多优点而迅速地打入市场。

目前，市场上有29种型号的PAL，可替代90%以上约400余种标准的TTL IC的逻辑功能，通过适当的编程可使一片PAL等效地替代多片TTL IC的逻辑功能，留心观察计算机主机板的读者一定会发现，目前新型的各种PC机的主机板上的IC数目，远远少于几年前生产的PC机。其原因就是采用了PAL器件替代了原来的TTL IC的缘故。

在技术发展的过程中，往往是由于某种产品的不完善，而继续推出更高性能的产品。关于PAL器件，虽然它有着上述许多优点，但它是一次编程定终身，不能重复改变功能使用。于是，就应运而生了GAL器件。GAL是美国品格(LATTICE)半导体公司1985年研制的最新一代PLD器件，它可在一秒钟内完成对芯片的电擦/写(无须用紫外线擦除)，具有加密功能，可重复并随时可修改其逻辑功能。所以，GAL器件已占领了PLD市场。国内对GAL器件的应用已引起广大开发与应用工作者的广泛而浓厚兴趣，一个普及应用GAL的“热潮”一定会在国内出现。

当然要用好GAL，还要掌握有关GAL器件的特性和设计方法，由于篇幅所限；《电子与电脑》将会不定期的对GAL及其编程器等知识进行介绍。本文仅给出GAL的概念，为了便于读者认识GAL，下面给出几种在国内较常用的GAL型号：GAL16V8，GAL20V8，GAL16V8A，GAL20V8A，ISPGAL16Z8，GAL39V8等。打算使用GAL器件或者有条件逛电子商行的读者，不妨先去找找看，先增加些感性认识。

本期封面我们选用了GAL器件的雕塑模型照片，希望您能在第一次认识GAL的时候，要记住GAL是可以由您任意塑造(编程或擦除)其逻辑功能的一种通用IC。

亲爱的读者，您要想开发新型的电子或计算机产品吗？您一定要逐步学会用GAL。

通用打印机状态检测程序

夏伟文

在PC机的DOS环境下，如果打印机未准备好而程序要求打印时，屏幕上会出现一个不优雅的提示“Abort,Retry,Ignore?”在这种情况下，有的程序不得不被迫中止运行乃至死机。因此，在打印以前通常要检测打印机的状态。在FOXbase+系统中，可以用SYS(13)函数来确认打印机是否正常；在BASIC中可以用ON ERROR GOTO与IF ERR=24 OR ERR=27 THEN语句配合来检测打印机是否正常；借助8088汇编语言调用BIOS中断的INT 17H的功能之一就是读出打印机的状态。下面用三段程序来说明这三种用法。

FOXbase+：

```
DO WHIL .T.
    IF SYS(13)='READY'
        EXIT
    ENDI
        CLEA
    ?? CHR(7)+CHR(7)
        SET COLO TO 6
    @10,25 SAY '打印机没准备好,请检查!'
        @12,27 SAY '按任意键再试!'
    SET COLO TO 7
    SET CONS OFF
        WAIT
    SET CONS ON
    ENDD
```

BASIC：

```
        10 ON ERROR GOTO 1000 '出现错误则转到第
        1000语句处理
        ……
        1000 REM 打印机出错则转5000执行
        1010 IF ERR=24 OR ERR=27 THEN GOSUB
        5000:RESUME
        5000 PRINT "打印机未准备好!":
             PRINT "按任意键再试一次……" '通知
        用户打印机出错
        5010 IF INKEY$="" THEN 5010 '等待键盘输
        入一个字符
        5020 RETURN '返回主程序继续打印
```

在汇编语言中调用BIOS中断17H，

```
    MOV AH,02     ;请求读端口
    MOV DX,00     ;选定默认的打印机端口
    INT 17H       ;调用BIOS中断
```

执行完这段汇编语言的操作以后，寄存器AH中的某些位设置成1。该寄存器一共7位，各位含义如下：7：打印机准备好；6：对最后一个字符的确认；5：无纸；4：选择打印机；3：I/O错；0：超时错。

以上三种方法，可以对打印机是否准备好作出简单的判断，但只能对打印机的错误给出笼统的提示。还有一种方法，可以区分出打印机错误的类型：是未联机、未通电、还是缺纸。下面先简述其原理：我们知道，DOS能够处理三个并行设备(LPT1～3)。每个并行设备通过相应的适配器与主机相联。对适配器的操作则是通过对其中的三个I/O寄存器的读写实现的，每个I/O寄存器都有自己的口地址，在BIOS数据区中分配二个字节存放适配器的基地址，所谓基地址即三组口地址中每一组的最低地址。默认的打印机LPT1的基地址存放在0040:0080。CPU通过状态寄存器和控制寄存器来进行打印操作。状态寄存器有7位：

(0～2位未用)

第3位：0=打印机出错　1=打印机正常

(未通电)

第4位：0=打印机脱机　1=打印机联机

第5位：0=打印机有纸　1=打印机无纸

第6位：0=打印机接到字　1=正常

符的回答信号

第7位：0=打印机工作　1=打印机不工作

要判断打印机是什么错误，只要读状态寄存器的第3、4、5位数值即可。例如，当寄存器的7位二进制位组合如下时：11011111(十进制数值为223)，表示打印机就绪。但不同的打印机尽管处于相同状态，状态寄存器位组合却不同。因此，可以用下面的程序1判定用户的打印机的状态参数，也就是分别在正常、断电、缺纸和未联机等情况下运行该程序，即可在屏幕上显示出不同情况下状态寄存器的7位二进制位组合的数值(十进制)。然后，将得到的参数依照给定的次序写进第二个程序的1000 DATA语句中，注意次序不要颠倒，就可以使用程序2对打印机状态随时测试。

程序1：

```
10 REM 测定参数
20 DEF SEG=&H40              '指向BIOS数据区
30 B=PEEK(9)*256+            '求状态寄存器的地址
PEEK(8)+1
40 A=INP(B)                  '求状态参数
50 PRINT A                   '显示参数
```

程序2：(以TH 3070打印机为例)

```
5 REM 程序名:CHPR.BAS
10 SCREEN 0,0,0:WIDTH 80:CLS:KEY OFF
20 READ A1,A2,A3,A4          '读入参数
30 DEF SEG=&H40              '指向BIOS数据区
40 B=PEEK(9)*256+            '求状态寄存器的地址
PEEK(8)+1
```

```
50 IF INP(B)=A1 THEN   '如果打印机准备好则
110                    转到 110 句退出 BASIC
60 BEEP:COLOR 7:K=1    '否则报警并提示
70 IF INP(B)=A2 THEN COLOR 7,0:CLS:LOCATE 10,21:
COLOR 20,1:PRINT "注意";:
COLOR 2,0:PRINT "请按打印机面板上的 SELECT(联机)
键!":GOSUB 130:BEEP:GOTO 70
80 IF INP(B)=A3 THEN COLOR 7,0:CLS:LOCATE 10,21:
COLOR 20,3:PRINT "注意";:
COLOR 4,0:PRINT "请打开打印机电源开关":GOSUB 130:
BEEP:GOTO 80
90 IF INP(B)=A4 THEN GOSUB 120:GOTO 90
100 IF INP(B)<>A1 THEN COLOR 7,0:CLS:LOCATE 10,25:
COLOR 20,1:PRINT "注意";:
COLOR 15,0:PRINT "打印机尚未准备好……":GOSUB 130:
K=K+1:IF K>7 THEN 50 ELSE 100
110 CLS:COLOR 7:SYSTEM
120 COLOR 7,0:CLS:LOCATE 10,27:COLOR 20,2:PRINT
"注 意";:COLOR 14,0:PRINT "请给打印机装好纸!":
PRINT:PRINT TAB()12;:PRINT "然后看面板 SELECT(联
机)和 PAPER END(缺纸)指示灯是否正常!":GOSUB 130:
BEEP:RETURN
130 FOR I=1 TO 3200:NEXT I:RETURN
1000 DATA 223,79,135,255' 数值含义依次为:正常,未联
机,未通电,缺纸
```

该打印机检测程序可以在 DBASE 或 FOXbase+ 系统下用 RUN GWBASIC CHPR 来调用。当打印机正常时,什么也不显示,继续执行下面的程序。出错时,则用声音告警和给出相应的彩色提示,直至用户按提示将打印机准备好为止。程序 1 和程序 2 已经在 GW-386/20 和 GW0520CH-Ⅱ型机上调试通过。

[通讯]

近几年来,我国的计算机变普及教育活动有了很大的发展,许多学习了计算机语言,并已具备了编制程序的能力。但是,如何进一步利用计算机的接口电路作一些开发应用,以及计算机出了故障敢于动手修理,许多人还缺乏硬件方面的知识。

为在计算机爱好者中培养具有一定计算机硬件基础知识和动手能力的人才,中国计算机学会与电子工业出版社联合举办了两期"学装微电脑"函授班。每期经过四个半月的学习,根据学员的作业,考试成绩,由中国计算机学会发给结业证书。

目前,第三期函授班的招生工作正在进行。

(接第 28 页)

这说明,若在文件 file2 中使用外部存储变量,则须在该文件中进行外部变量说明。

②性质　外部存储变量有如下性质:

- 作用域的全局性　全部存储变量是全局变量。
- 生存期的永久性。
- 未初始化的外部存储变量的值为 0。

(4)寄存器存储变量

用关键字 register 定义,如:

```
register int e;
```

寄存器变量值存储在 CPU 中。设定这种变量是为了提高操作速度。它们常用于循环控制。Turbo C 允许同时定义两个寄存器变量。

新书邮购消息
电子工业出版社

代号	书名	定价(含)邮费
B0011	针式打印机	8.70
B0012	电源	8.70
B0013	5.25 英寸磁盘机	7.80
B0016	微型计算机电路分析与维修丛书电路图集(一)	43.70
B0017	微型计算机电路分析与维修丛书电路图集(二)	203.50
B0202	IBM PC 编程指南	8.10
B0203	dBASE Ⅲ应用解疑	5.80
B0204	MS-DOS 高水平程序设计	6.40
B0205	DOS/BIOS 使用详解	9.20
B1430	中华学习机使用技巧与检修 360 问	12.70

邮购办法:

- 邮局汇款　地址:北京市万寿路电子工业出版社邮购科(邮编 100036)
  请将购书清单填写在汇款单的"附言栏"内
- 银行汇款　开户行:北京市工商银行翠微路分理处
  帐号:661036-40
  请将购书清单随银行汇单一并寄来,不必另附信
- 注意事项　填购书清单,请务必写明"书代号"及所购册数,书名可作省略;收书人地址要写详细,字迹清晰
- 联系电话　81.3693(电报挂号:3101)

dBASE Ⅲ与BASICA之间的数据传送问题

福建泉州华侨大学　严桂兰

在大量的应用软件中，人们已广泛地使用各种高级语言进行混合设计。这既可以充分调用各语言的优点，相互补充，相互完善；也可达到优化设计，提高软件质量的目的。本文仅就dBASE Ⅲ与高级BASIC之间的数据传送的隐含问题作些技术性的阐述。

一、dBASE Ⅲ多字段数据被BASICA完整地接受。

在dBASE Ⅲ系统中，每条数据记录可由高达128项字段组成，其字节数可高达4000。这对大、中型企业的事务管理项基本上是够用的。当将多个字段数据传送给BASICA时，却遇上了麻烦。首先是BASICA的语句行中限制字符数不超过255，而来自dBASE Ⅲ多字段数据往往在BASICA接受数据的书写行中大大超过255个字符数。虽然在编辑输入BASICA程序行时，可无止境地键入多字段的数据格式，屏幕并无出错提示，可实际能接受的字符数仍是255个，超过部分都将被截除，引起数据丢失。为保证数据的完整性，可采取将原dBASE Ⅲ的数据库文件生成二个(或者更多个)数据文件的方案。第一个数据文件可保留库中全部数据；第二个数据文件则截取库文件中后半部分数据。如此在BASICA中便可以两个数据文件来接受数据。原来由一个语句行接受不了的数据，现改为由两个语句行来接受，取第一个数据文件的前半部分数据(书写字符数小于255)，剩余部分数据由第二个数据文件提供。为节省空间，第一个数据文件也可取对应的前半部分数据，但要从BASICA或者从两个数据文件返回库文件形成一个完整的记录数据时，又会引起麻烦，因而，采用稍多的内存开销来保持数据的完整性及数据双向传送是值得的。

例如，在zdq.dbf数据库中有39个字段，其字节长度为211，若用一个数据文件传送数据给BASICA，其LEN＝211＋2＝213：

```
……
140 OPEN "ZDQ.TXT" AS#1 LEN=213
150 FIELD#1,3 AS A1$,12 AS A2$,8 AS A3$,5 AS A4$,16 AS A5$,16 AS A6$,6 AS A7$,4 AS A8$,8 AS A9$,16 AS A10$,4 AS A11$ ……, 4 AS A37$,4 AS A38$,5 AS A39$
```

在键入150行号内容时，形式上能按上述进行，无出错信息，但实际能被接受是：

```
"FIELD#1,3 AS A1$……
4 AS A14$,……,4 AS A24$,4 AS A25$,4"
```

(共255个字符)

从A26$项到A39$项数据没能被接受。在BASICA中又无继续行的语法，因此，BASICA本身是无法解决这一矛盾的。如果将zdq.dbf生成zdq1.txt及zdqq1.txt文件：

```
Use zdq
Copy to zdq1 sdf
Copy to zdqq1 fileds a26,a27,…a39 sdf
```

则在BASICA中采用：

```
……
140 z1="zdq1.txt";z2="zdqq1.txt"
145 OPEN Z1$ AS#1 LEN=213:OPEN Z2$ AS#2 LEN=59
150 FIELD#1,3 AS A1$,12 AS A2$……,4 AS A25$
155 FIELD#2,4 AS A26$,4 AS A27$……4 AS A38$,5 AS A39$
```

注意：用两个数据文件传送数据时，每个打开文件的语句中的LEN值要比各数据文件数据项字节数之和加2。如此处理后，BASICA程序就能完整地接受来自dBASE Ⅲ数据库中多个字段数据。

二、同一结构，不同数据的产生与传送

在实际使用中，往往随着时间的推移或者不同的对象，其数据完全不同，但所对应的数据项名、结构却完全相同，例如，银行存款单，其存款种类、帐号、户名，存款金额，期限、利率…等，不同的户名，就有一组对应数据；又如，外销企业，其外商的订单：订单号、订货日期、货号、订货数量、单价、金额、交货日期…也因不同的外商，其对应数据发生变化；在海关中的报关单，税务局的税款单等等，都属于结构相同，数据不同的类型。对于此类数据若采用dBASE进行管理，其输入、查询、检索是十分方便的，而且，还可利用同一数据库结构，对不同对象的数据进行各种操作，然后，转成数据文件保存各对象的不同数据，数据库便可腾出进行另一对象数据组的操作。如此往返使用同一数据库对数据进行管理，而后转变产生各自的数据文件的办法，可节省空间，程序结构也十分清晰，同时也为与其它语言接口作出必要的准备，实属一举多得。由于不同对象仅是数据不同，此时可按顺序建立同名(与数据库同名更易操作)仅以最后序号不同来区别，例如，新的一张订单数据，可根据已存在的最后的一个数据文件序号加1，而产生新的数据文件名：

```
use zdq                          ;打开订单库
zap                              ;删除上一个订单数据
:                                ;输入新的订单数据
:
n=1
nn=str(n,1)
do while file("'zdq'+nn  +'.txt'")
   n=n+1
    if n<10
```

```
        nn=str(n,1)
      else
        nn=str(n,2)
      endif                    ;以序号在 100 之内为例
enddo
z="zdq"+nn
copy to &z sdf                 ;建立新订单的数据文件
```

若已有两个订单数据文件 zdq1.txt 和 zdq2.txt，则按上面程序段再输入一个新的订单数据，便产生 zdq3.txt 文件，如此类推。

现在若要将同一结构，不同对象的数据传送到 BASICA 进行数据处理，即将遇到的第二问题是：BASICA 中到底要设置多少行对应的接受语句？（每一个数据文件都必须采用两个语句行——一个语句行用来打开该数据文件及参数设置，另一行则要取其各数据项对应数据。）由于订单数量是随机的，无法确定或限死订单数量，即数据文件的个数，BASICA 程序也就无法编写相应的接受语句。然而，通过上述由 dBASE Ⅲ 产生同一结构不同数据文件方法的启示，也可在 BASICA 中采用相应的公式化接受语句：

```
……
125 INPUT "输入订单号:";N
130 IF N>=10 THEN NN$=RIGHT¥(STR(N),2)
135 NN$=RIGHT$(STR$(N),1)
135 Z1$="zdq"+NN$+".txt":
    Z2$="zdqq"+NN$+".txt"
145 OPEN Z1$ AS#1 LEN=213:OPEN Z2$ AS #2 LEN=59
    (以下 150－155 行同上)
```

上述七个语句行（一般五个）便可接受来自 dBASE Ⅲ 数据库中传送来的任意多个数据文件的任意字段数据，因此，用此公式化的接受语句便可解决同一结构多个数据文件的数据传送问题。

三、数据传送中的软件接口

各高级语言间传送数据的软件接口，一般采用数据文件。BASICA 接受来自 dBASE Ⅲ 的数据，有两种软件接口形式，一种是由 dBASE 库文件产生数据文件，另一种是数据库文件本身。前者传送的数据较为简单、直观，但要多开销内存，它除数据库文件外，还有对应的数据文件存在；后者传送的数据直接来自数据库文件，但取数据的方式较前者麻烦，需经过计算才能得到，对于中文数据还面临着数据保真性问题。

上面描述了由数据库产生数据文件，而后被 BASICA 取其数据的全过程。但这是数据传送的特例，因为它并不是每一个数据文件都对应一个数据库，而是多个数据文件共用一个数据库结构，较为经济，数据文件是无结构的，因此，多个数据文件加一个数据库所占内存必然比对应多个数据库文件所占内存要少。对于不同结构不同数据的传送，若也采取数据文件方式，则多个数据库必然对应有多个数据文件，这势必要开销更多的内存，为此，若直接利用数据库传送数据较为合适，它可省去对应多个数据文件。例如：

```
:
:
240 OPEN "ao.dbf" AS#3 LEN=66
```

由于数据库带有结构，要读出库中的数据，需跨越文件标识信息及结构，因此，从什么字段开始取数据，取多长字符，需根据下列公式计算而得：

$N=32+32\times n+2$

其中 n 为库中的字段数

N/LEN＝整数…余数

LEN 为各字段长之和再加 1，此 1 的字段号设为 0 字段，专为安排记录作删除记号"＊"而设的字符长度。

依照余数大小，从 0 字段开始，顺序将各字段长累加，直到某个字段时，其字段长与前累加之和大于或等于余数为止，然后，对此特定的字段长分成两部，前半部分与前累加字段长之和正好等于余数值，BASICA 便从此字段剩余的后半部分开始取数据，继而顺序向后字段取数据，直到最后字段，再接着从 0 字段开始顺序地取数据，一直取到特定字段的前部分字段长数据为止（此时字段序号相同，序号前字符另取）。

例如，ao.dbf 库中有 11 个字段．其各字段长之和为 65，所以

```
LEN=65+1=66
字段号 0 1 2 3 4 5 6 7 8 9 10 11
字段长 1 4 6 4 4 4 4 4 8 8 10 9
          66
                        终止 9 1 起始
          56
```

$N=32+32\times 11+2=386$

$386/66=5\cdots 56$

因此，BASICA 开始从库文件的 10 号字段取数据，其数据字符长度为 1，即从此字段数据的最后一个字节取起，接着再取 11 号字段数据，而后再从 0 号字段取数据，直到 10 号字段的另一部分数据取完（见上示意图）。其 BASICA 取 ao.dbf 中数据的完整形式为：

```
……
240 OPEN "ao.dbf" AS#3 LEN=66
245 FIELD#3,1 AS D10$,9 AS D11$,1 AS D0$,4 AS D1$,6 AS D2$,4 AS D3$,4 AS D4$,4 AS D5$,4 AS D6$,4 AS D7$,8 AS D8$,8 AS D9$,9 AS E10$
```

在 245 行中，起始字段与终止字段都是 10 号字段，其前后数据字节数和为原字段长度，字段序号均为 10，仅名称不同，起始处要与其它字段名同，终止字段名可另取（如 E10$），这样，DBASE Ⅲ 数据库的数据便可直接传送到 BASICA，不再需要软件接口过渡。值得一提的是中文数据的传送，汉字占两个字符宽度。当采用库文件传送汉字时，按上述公式组计算，某特定字段长要被分成两部分，此时若前部分字段长出现

奇数，则会使汉字传送失真。这是不允许的，为此，需返回 dBASE Ⅲ 中修改数据库结构。为凑成被分部分字段长为偶数，宁可在某些字段中或前或后加入空白字符来加大字段长度。这一问题在程序设计时就要考虑与确定，避免在程序调试中才来处理。

dBASE 运用经验与技巧

孙　鹏

一、浅谈汉字 dBASE Ⅲ 与高级语言的接口问题

尽管汉字 dBASE Ⅲ 具有很强的数据处理能力，但它对复杂的计算、图象处理、功能扩充等方面就显得不太适应，比不上 BASIC、PASCAL、FORTRAN、C 语言等。倘若能将汉字 dBASE Ⅲ 与高级语言结合起来，相互取长补短，融为一个有机整体，是比较理想的，这就要解决汉字 dBASE Ⅲ 与高语言的接口问题。dBASE Ⅲ 调用高级语言是通过 RUN 命令实现的，被调用的高级语言可通过编译、连接形成. COM 或. EXE 文件（解释 BASIC 除外）以外部文件存在磁盘上，只要高级语言已形成独立的可执行文件，汉字 dBASE Ⅲ 总可以用语句 RUN 来调用它。

由于汉字 dBASE Ⅲ 没有提供直接与高级语言的接口，因而它与高级语言的数据交换只能间接进行，但汉字 dBASE Ⅲ 的文本文件是 ASCII 码文件，而某些高级语言都有处理 ASCII 码文件的能力，因此，实现汉字 dBASE Ⅲ 与高级语言的数据传递，文本文件. TXT 起着关键作用，实际上是汉字 dBASE Ⅲ 与高级语言的数据接口。如果弄清了高级语言数据文件的存储格式，那么与汉字 dBASE Ⅲ 的数据交换是可以实现的。比如：PASCAL 语言有文本文件和顺序文件两种类型，那么我们就要选择文本文件，因为顺序文件是以二进制格式存储的，文本文件是以 ASCII 形式存储的；COBOL 语言有顺序、索引、直接文件三种类型，其中顺序文件是 ASCII 文件，数据之间没有分隔符，其存储形式与汉字 dBASE Ⅲ 的标准文本文件格式一致，因此，COBOL 语言与汉字 dBASE Ⅲ 交换数据时选择顺序文件比较合适；FORTRAN 语言分有格式和无格式输入/输出两种文件，而有格式输入/输出文件是以 ASCII 码形式存储的，所以选有格式输入/输出比较合适。下面主要以 PASCAL 语言为例进行说明。

PASCAL 语言数据文件是以空格作为数据分隔符的，因此，在 dBASE Ⅲ 处理时，数据库的数据应使用 DELIMETED WITH BLANK 数据格式进行复制，下面的 PASCAL 语言程序是将汉字 dBASE Ⅲ 的 SPP1. TXT 文件数据取出，经过处理后存于 SPP2. TXT 中，供汉字 dBASE Ⅲ 程序使用。

汉字 dBASE Ⅲ 程序：

```
    :
    USE SS
    COPY TO SPP1 DELIMITED WITH BLANK
    RUN SP
    USE SS1
    APPEND FROM SPP2 SDF
    :
```

PASCAL 语言程序：

```
    PROGRAM SP(INPUT,OUTPUT);
      VAR
        SP1,SP2:TEXT;
            :
        BEGIN
            :
          ASSIGN(SP1,'SPP1.TXT');RTSET(SP1);
          ASSIGN(SP2,'SPP2.TXT');REWRITE(SP2);
            :
          READ(SP1,XM,NL);READLN(SP1);
            :
          WRITE(SP2,XM:6,NL:2);WRITELN(SP2);
            :
          CLOSE(SP1);CLOSE(SP2);
        END.
```

综上所述，弄清高级语言数据文件的存储格式，是实现汉字 dBASE Ⅲ 与高级语言数据交换成败的关键，不同语言有不同的数据存储格式，掌握了数据的存储格式，就能实现与汉字 dBASE Ⅲ 的数据交换。

二、关于汉字 dBASE Ⅲ 中 CONFIG. SYS 和 CONFIG. DB 文件的建立

1、CONFIG. SYS 文件的建立

在调试汉字 dBASE Ⅲ 命令文件的过程中，打开 3 个以上的数据库文件，甚至有时只打开 1 个数据库文件，系统便发出打开文件太多的信息，产生此原因和 CONFIG. SYS 文件有关。因为不论在什么时候，系统启动时都要检测 CONFIG. SYS 文件，若不设定就按默认值执行，即同时打开文件数为 8 个，缓冲区为 2 个。而系统本身占用 5 个文件，为了在调试、使用汉字 dBASE Ⅲ 的过程中，可打开足够的文件个数，需要对 CONFIG. SYS 文件重新加以设置，一般只需要写入如下 3 个语句：

DEVICE＝ANSI. SYS

FILES＝20

BUFFERS＝15

因为在汉字 dBASE Ⅲ 中，规定可以同时打开 10 个数据库文件和 15 个命令文件，所以文件的参数设置 20 个比较合适，缓冲区设置 10～15 比较合适，最多可不超过 20 个。为了加快系统运行速度，用户往往需要增加 DOS 缓冲区个数，但缓冲区设置过多并不是一件好事，一是，用户可用的内存空间就小了，因为每增加 1 个缓冲区就会使 DOS 在内存多占 528 个字节，故在内存容量较小情况下应设置较小的缓冲区；二是缓冲区的数量设置如果大于 20 个，系统可能会出现运行速度减慢的现象，原因是由于设置缓冲区过多，造成 DOS 在所有缓冲区中检索记录所用的时间要比直接从磁盘中读取记录的时间长。实践证明：对于内存是 640K(或 512K)的，设置 10～15 个缓冲区，系统就可以运行较快了，对于内存是 256K 的，设置 4 就可以了。总之，BUFFERS 的设置大小取决于用户计算机的内存容量，

2、CONFIG. DB 文件的建立

汉字 dBASE Ⅲ 提供了内部的字处理程序来编辑命令文件，但它的功能有限，不能编辑超过 4K 的程序，但是，汉字 dBASE Ⅲ 提供了另一种方法，在不退出 dBASE 的情况下，调用 EDLIN 和 WORDSTAR，不但可编辑大于 4K 的程序，而且还提供了一系列汉字 dBASE Ⅲ 字处理程序所不具备的灵活编辑手段，这种方法是通过对 CONFIG. DB 文件的设置来实现的，如果在该文件中设置：

TEDIT＝EDLIN

WP＝WS

两条命令，这样用户可直接在汉字 dBASE Ⅲ 中用行编辑 EDLIN 建立和修改命令文件，编辑 MEMO 字段时，装入字处理程序 WORDSTAR。

在 CONFIG. DB 文件中，可设置功能键、SET 命令的 ON/OFF 状态、系统可用的最大内存空间、编辑方式等。下面是笔者建立的 CONFIG. DB 文件：

TALK＝OFF	执行结果不送屏幕
CARRY＝ON	将上一记录复制到当前记录
COLOR＝3,4,6	设置屏幕属性
DEFAULT＝A	当前驱动器为 A
DEVICE＝SCREEN	@…SAY 命令送屏幕
ECHO＝OFF	ESC 键终止命令文件
F2＝MODIFY COMMAND	设置功能键 F2
MAXMEM＝540K	可用最大内存为 540k
MVARSIZ＝5K	内存变量空间为 5k
PATH＝\SP	设置路径
SAFETY＝ON	设文件重写保护
PRINT＝OFF	关闭打印机
TEDIT＝EDLIN	编辑命令文件调 EDLIN
WP＝WS	编辑 MEMO 调 WORDSTAR
COMMAND＝DO SSS	进入 DBASE Ⅲ 后自动执行 SSS. PRG

当汉字 dBASE Ⅲ 启动时，系统自动查找 CONFIG. DB 文件，若找到自动装入，按该文件的设置来装配，执行后回到命令状态。

三、快速查找命令文件中错误的方法与技巧

在汉字 dBASE Ⅲ 中，对发生的语法错误(标点符号错、命令名错、内存变量未初始化等)、结构错误(IF 和 ENDIF、DO WHILE 和 ENDDO 不配对等)，系统会自动给出提示，而对逻辑错误，系统不会给出提示，这要给调试工作带来不便，我们不妨通过以下两个方面来调试程序。

1、使用汉字 dBASE Ⅲ 中以下的几个 SET 命令

SET TALK ON 将命令执行结果显示在屏幕上

可用该命令判断逻辑执行是否正确。

SET ECHO ON 将正在执行的命令显示在屏幕上或在打印机上输出

用该命令便于跟踪程序的执行流程。

SET STEP ON 执行一条命令后暂停

用该命令可把出错的语句孤立出来。

SET DEBUG ON 将 SET ECHO ON 输出送打印机

以防止调试中产生的屏幕显示和程序运行中应该产生的屏幕显示相混淆。

但使用以上几个 SET 命令比较浪费机时，特别是 SET STEP ON 命令，因而一旦调通，找到错误所在，就将它们置成 OFF 状态，最好的方法是将它们用在可能出错的程序段，以便节省调试时间。

2、采用 *、IF…ENDIF、DO WHILE…ENDDO 等，临时去掉程序的某些部分，使这部分程序段不被执行，来确定程序出错的根源所在。或者在程序适当的地方插入一些命令，如：?、??、@…SAY 或 WAIT 命令等，以便观察程序运行结果，有利于分段进行调试。

KD－GAL 编程器

KD－GAL 编程器通过和 IBM－PC 及兼容机的串行口进行通讯，全部编程过程是在“菜单”结构的提示下进行 100%的高速编程，并具有自动比较，擦除等功能，操作灵活方便、快速。

编程器体积小，携带及使用方便，是当今电子产品研制、生产理想的工具。

该产品实行三包，终身保修。

我室在超声应用、微机软、硬件应用方面具有强有力的工程技术力量。可替代各用户单独设计用 GAL 芯片实现的各种规模的数字逻辑。

欢迎来人来函洽谈。

KD－GAL 编程器每台 1100 元(大专院校可优惠 10%)。

联系地址：安徽合肥科大无线电系

联系人：朱世鸿　邮政编码：230026

帐号：10514461865　开户行：合肥工商行望办

单位：中国科大科技开发总公司

如何解决运行 FOXBASE 内存不够的问题

湖北天门市 121 信箱微机室　　周日初

运行 FOXBASE 要求内存容量≥375k，但对于只有 640k 内存的 IBM PC/XT 机，在装入 CCDOS(CCLIB 约为 232k)后，所剩内存不过 365k，虽然能够进入 FOXBASE，并且可以执行 FOXBASE 命令文件，但运行速度还不及 DBASE Ⅲ，而且用 MODI COMM 来编辑文件时会出现无足够的内存空间。为了克服在运行汉化 FOXBASE 时因内存不够而产生的诸多毛病，可以减少内存占用量。减少内存开销主要是考虑如何缩小汉字字库。

在字库中，二级汉字有 3008 个，这些汉字平时用得极少，为了减少内存，可以删除这些字，这样 CCLIB 的长度就变为：237632－3008＊32＝141376

知道了 CCLIB 的长度，就可用 DEBUG 来改变 CCLIB 文件的长度，具体步骤如下：

```
debug cclib↙
－r↙
AX＝0000 BX＝0003 CX＝A040 DX＝0000 SP＝FFEE
BP＝0000 SI＝0000 DI＝0000
DS＝567D ES＝567D SS＝567D CS＝567D NV UP EI PL
NZ NA PO NC
567D:0100 0100      ADD [BX＋SI],AX          DS:
0003＝00A0
－rbx↙
BX 0003
:0002↙
－rcx↙
CX A040
:2840↙
－W↙
Writing 22840 bytes
－q↙
```

若你不懂得寄存器 BX、CX 中数的含义可用下法求得：BX ＝ INT (141376/65536) ＝ 2　　CX ＝ HEX (141376－INT(141376/65536))＝2840H

注：1、笔者所使用的机器为 IBM PC/XT，内存 640k、25 行汉显，用这种方法改完运行 FOXBASE 效果较满意。

2、若二级字库中有一部分汉字是你所必需的，那么可以先使用笔者在去年十月份开发的软件修改字库，再按上述步骤进行。该软件有一功能可把字库中的某个字移到字库的非汉字区，需此源程序者，请与本编辑部联系。(在运行此程序前，还要用 EDLIN 编辑两个长度均为 25 字节的 HIGHTER 和 HIGHTER. ZH 文件，其中 HIGHTER. ZH 为只读文件)

COMPRESS 在硬盘集中存储中的应用

常州 102 医院　许　鹰

由于信息的不断更新和增减，一些文件，尤其是较大的文件，其存储的簇块往往分散在硬盘多个地方。当读取这些文件时，就造成磁头的来回寻道读取，使系统的运行时间大大增加，同时也加快了磁头的磨损。文件增删次数越多，文件数据变化越大，存储的分散现象就越严重。因此，有必要对这些文件及数据的存储区进行归并处理，尽量使同一文件的信息存储在相连的簇块中。下面介绍一种简单而有效的方法：利用 pctools 工具盘上的一个专用于压缩磁盘存储区，处理零星分散簇块的实用程序 compress. exe 来处理。该程序在 pctools4. 11－5. 1 版上都有。主要操作步骤如下：

1. 运行前先把硬盘上无用的文件删除掉。

2. 在硬盘上执行 CHKDSK/F 的 DOS 命令，将散失簇块收集到一个文件中，使压缩达到最佳效果。

3. 将带有 COMPRESS 的软盘插入 A 驱动器，然后键入 COMPRESS，这时出现主菜单。键入需要处理的驱动器号“C”。接着进行功能选择，键入“5”，即进行磁盘文件压缩处理。然后选择“C”。回车后，屏幕上即出现所要压缩的磁盘物理图。可以明显地看到文件目录区后的各个文件已开始一个簇接一个簇地顺序重写，而原来的文件则被暂时移到盘上空余的其它地方。等到上一文件完整地写完后，这些“拆迁户”才被安排回来接下去写，就这样直至所有文件重写完。最后自动清除掉散失的簇块。上述过程对 20 兆硬 盘来说需要十几分钟。由于上述过程重排了文件在磁盘上的物理位置，因此如果使用的汉字系统是 CCBIOS2. 13A－F 的话，需要对涉及文件在磁盘存储位置有关的高点阵字库的索引文件重新索引，即重新执行一下 WORK24. EXE 文件，以产生新的 24 点阵字库索引文件，这样才能正确打印 24 点阵汉字。

怎样在IBM—PC机上实现汉字的动态显示

李晓华

所谓动态显示就是指在屏幕上中的画面，能够按一定规则(通过编程)在屏幕上进行移动的图形。由于人眼的反应往往比较迟纯，将一些跳跃的运动看成是平稳的。因此，可应用这种视幻觉使图形在屏幕上移动。其具体方法是：首先画出该图形，接着消除该图形，然后在一个微小变化的位置处再显示出该图形。通过这一系列操作，就可以在屏幕上达到使得图形移动。由于在屏幕上显示的汉字是作为图形处理的特例，因此在屏幕上显示的所有的图形都能适用这一原理。

一、怎样使汉字(图形)在屏幕上"动画"

首先是屏幕上的光标定位，接着便是在该位置上显示出汉字，之后通过一段延时程序，使得汉字移动起来有一定的连贯性。延时完成后应马上把该位置的汉字消除。最后通过一定的计算，改变其汉字的位置，再显示汉字……。重复上述过程，这样汉字便在屏幕上移动起来了。

根据上述原理，利用BASIC语言的PRINT语句和LOCATE就能够使汉字在屏幕上移动起来。例一：

```
10 X=10
20 Y=1
30 LOCATE X,Y
40 PRINT "云"
50 FOR I=1 TO 100
60 NEXT I
70 LOCATE X,Y
80 PRINT " "
90 Y=Y+2
100 IF Y>80 THEN Y=1
110 GOTO 30
```

10,20句给X,Y坐标赋初值。30,40句光标定位，显示字符串。50,60句延时。70,80句清除被显示的字符。90句对Y坐标进行累加，改变Y坐标的位置。100句的判断使汉字移动时不超出屏幕范围。

二、怎样使汉字'字符串'在屏幕上"动画"

为使屏幕设计具有动态画面，使得汉字从不同角度一个个显示出来，增加图形的真实感，在程序中利用BASIC的MID$函数对字符串进行分解。在应用时要注意汉字的完整性，其余同上。例二：

```
20 H=20
30 X=2
40 CH$="在屏幕上实现汉字的动态显示"
50 FOR N=1 TO 26 STEP 2
55 Y=80
60 CH1$=MID$(CH$,N,2)
70 LOCATE X,Y
80 PRINT CH1$
90 FOR I=0 TO 10
100 NEXT I
110 LOCATE X,Y
120 PRINT " "
130 Y=Y-2
140 IF Y=H THEN GOTO 160
150 GOTO 70
160 H=H+2
170 LOCATE X,H
180 PRINT CH1$
190 NEXT N
200 END
```

三、具有一定速度和加速度的汉字(图形)"动画"

上面两个例子使我们看到了产生动态汉字的方法。通过这些"画—不画—移动—再画"的操作，就可以使汉字动态地移动。只要通过对运动路线进行计算就能使汉字图形沿更复杂的路径运动。但对于移动来说它必须和速度有关。对于显示器我们看成是三维的，因此定义了三种速度，即水平方向速度、垂直方向速度和加速度。例如，定义水平方向速度为XV，垂直方向速度为YH，水平方向加速度为XVS，垂直方向加速度为YHC。这样不仅可改变汉字的位置，而且也能改变移动速度，改变后的位置通过下面进行计算：

```
X=X+XVS
Y=Y+YHC
XVS=XVS+A
YHC=YHC+B
```

其中A,B分别为水平和垂直加速度。例三：

```
30 SCREEN 0:CLS
40 X=1:Y=1:V=1:H=1:A$="汉字动显示"
50 COLOR 7,0:LOCATE X,Y:PRINT A$;
60 LOCATE X,Y:COLOR 7,0:PRINT "          ";
70 X=X+V
80 Y=Y+H
90 IF X=5 AND Y=32 THEN PRINT A$;:GOTO 130
100 IF X=23 OR X=1 THEN V=-V
110 IF Y=80 OR Y=1 THEN H=-H
120 GOTO 50
130 END
```

四、用C语言实现汉字动态移动

利用C语言实现汉字动态显示的方法与上述原理相同。首先定义一个外部变量DXY(该变量为X、Y坐标，通过改变该变量从而使得汉字移动)。该程序中有一个主程序(MAIN())和一个函数(DISP(DXY))，程序MAIN()用于显示汉字、消除汉字、计算X(Y)的坐标值；DISP()函数用于光标的定位，该函数是通过调用BIOS的10H中断实现光标定位的。例四：

```
int dxy;
main()
{
```

```
    extern int dxy;
    for (dxy=0x0501;dxy<0x0546;dxy++)
    {
disp(dxy);
printf("汉字动显示");
disp(dxy);
printf("          ");
    }
}
disp(dxy)
{
    extern int dxy;
    struct{int ax,bx,cx,dx,si,di,ds,es;}rv;
    rv.ax=0x0200;
    rv.dx=dxy;
    rv.bx=0x0000;
    sysint(0x10,&rv,&rv);
    return(dxy);
}
```

DOS I/O 重定向、管道功能及其使用

江西省经济信息中心 徐国辉

DOS 版本 2 增强的功能之一是支持类似于 UNIX 的 I/O 重定向和管道。如果用户能够熟悉并灵活使用它们，将会带来极大的方便。

I/O 重定向

DOS 数据源设备和目的设备叫做标准输入和标准输出，通常键盘为标准输入，显示器为标准输出，它们为两个预定的 I/O 通道，可被任何程序存取。在 I/O 重定向中，用户可以改变上面用于正常输入和输出的源及目的，即通过 DOS 命令行中的参数使这些通道之一重定向到某一设备(如打印机)和磁盘文件上。

"<"表示重定向程序的输入符，">"表示重定向程序的输出符，">>"是把重定向的程序输出加到已建立的文件上。例如命令 DIR>PRN，它不是在显示器上显示，而是输出到打印机上。

管道

管道是伴随 I/O 重定向而来的。一个管道是一条连接两个程序的计算机处理线，就是把一个程序的输出变为另一个程序的输入。"|"表示管道操作符。如 TYPE LOCK|DEBUG 这条命令表示把 TYPE 显示的内容作为 DEBUG 的输入。

筛选程序

DOS 提供了 SORT、FIND、MORE 三个筛选程序。它们是 DOS 支持 I/O 重定向和管道环境下运行的。筛选程序是将一字符流作为输入，对其进行处理并将处理后的字符流作为输出。文本流源和目的也可为任何字符设备，文件或系统所知的程序。SORT 对文本数据排序；FIND 寻找输入流中与指定字符串匹配的字符串；MORE 一次显示一屏数据。例如：

DIR|MORE

表示一次显示一屏 DOS 执行 DIR 信息，按空格键显示下一屏信息。

SORT<WORDS|MORE

表示 DOS 从 WORDS 文件中接收字，并对这些字进行排序，得出的结果，每次通过 MORE 显示整屏行。

DIR|FIND"01－01－90"|SORT>PRN

表示 DOS 在 DIR 中查找"01－01－90"字符串的行，结果送给 SORT 排序且把输出送给打印机打印。

综合使用实例

例 1：把 COMMAND.COM 反汇编程序转为磁盘文件，以便转换到高速打印机(如 IBM 3200)打印出来进行分析。

1. 建立文件 CFILE

```
n command.com
l
u 100 4680
q
```

2. 键入下面命令

TYPE CFILE|DEBUG>UCOMM

UCOMM 就是 COMMAND.COM 的反汇编程序的磁盘文件。

例 2：对 XGH 子目录进行加密和解密。其原理是利用 DEBUG 修改子目录开始簇号，即第 26、27 字节。

1. 用 DEBUG 调磁盘目录区记下 XGH 子目录的第 26－27 字节在段内偏移和数值，它们分别为 1FA，1FB 2D，00

2. 建立下面两个文件

LOCK	KEY
1 100 2 11 10	1 100 2 11 10
elfa 00 2d	elfa 2d 00
w 100 2 11 10	w 100 2 11 10
q	q

3. 加密时键入 TYPE LOCK|DEBUG>S。解密时键入 TYPE KEY|DEBUG>SS。这两个命令是把 TYPE 的输出作为 DEBUG 的输入，修改和恢复子目录开始族号，同时把显示的内容输出到给定的磁盘文件上使 LOCK 和 KEY 文件在 CRT 上不显示。

DOSI/O 重定向和管道功能除了上述用法外，还可以用来调试程序，程序员可以通过建立用于程序的"回答"磁盘文件(代替键盘输入)来调试程序，运行程序时把输入重定向到该磁盘文件上，可减少调试时间。用户也可编写自己的筛选程序，从而更好地利用 DOS 环境下提供的 I/O 重定向和管道的强大功能。

在低版本 DOS 下安装虚拟盘

江苏江阴　　周健勇

虚拟盘素有执行速度快、磁盘与驱动器磨损小等优点，但只在 DOS 3.0 及以上的系统才具有虚拟盘功能，因为它提供了一个外部命令文件 VDISK.SYS。我们也能利用它在 PC/XT 机的低版本 DOS 下安装虚拟盘。具体方法是：

①将 VDISK.SYS 文件拷到低版的 DOS 系统盘上。如 DOS 2.1 盘。

②在 DOS 2.1 系统盘上的 CONFIG.SYS 文件中增加一条 DEVICE＝VDISK.SYS bbb[,sss][,ddd]命令。

bbb—— 虚盘的容量，单位 k，默认值 64k。

sss—— 虚盘的扇区大小，单位为字节，可选 128、256、512，默认值 128。

ddd—— 虚盘目录项数，可选 2～512，默认值 64。

例如：建立一个 120k 容量、扇区为 128、目录项为 64 的虚拟盘，则将：

DEVICE＝VDISK.SYS 120,128,64

写入 CONFIG.SYS 文件中。

③用该 DOS 2.1 系统盘重新启动机器，屏幕显示虚拟盘的容量、扇区大小及目录项数：

```
VDISK Version 3.2 virtual disk
Buffer size:     120KB
Sector size:     128
Directory entries:     64
Current date is Tue 1－01－1980
Enter new date:
```

至此，已安装成功，就可以在虚盘上进行操作了。若机器已装有硬盘，则虚拟盘提示符为 D＞。

本例在 Super PC/XT－Ⅲ、IBM PC/XT 286 两种机器上通过。

五笔 CCDOS 中一字节修正

上海现代信息技术研究所　　吴邦忠

五笔字型 CCDOS 汉字操作系统以其独特的编码优点被越来越多的国内外微机用户所采用。然而在实际使用该操作系统中我们发现，当用组合键 ctrl－F10 进入改变打印字号、打印行宽状态时，如果在已经键入打印字号后（即：此时光标停在等待接收行宽参数位置），先从键盘打入退格键（backspace 键）就会发现，此后系统只接收键盘输入的字母（A－P）、回车（Enter）及退格键，而不接收数字键，而且当键盘输入有效字符（A－P）和回车或直接键入回车后，系统就结束定义打印字号、行宽的过程，返回提示符 C＞状态，造成不能改变打印行宽。我们经过分析 CCBIOS 中键盘管理模块（int 16H）后发现，该模块在处理 ctrl－F10 组合键的程序段（CS:3198－CS:32B2）中，主要是由于以下二个指令：

```
CS:31f9 CMP byte ptr[2896],02
CS:31fe JB 325b
```

判断有误，其修改方法如下：

```
1. C＞ren peng4.exe peng4
2. C＞debug peng4
－E cs:34fd 03↙
－W ↙
－Q↙
3. C＞ren peng4 peng4.exe
```

将经过修改后的 peng4.exe 文件引导装入，就不会产生本文上面所述的问题。如果先前已经进入汉字操作系统，则须重新启动（热启动或冷启动）计算机（PC－XT 或其它兼容机），再将 peng4.exe 文件引导装入。

注：单元[CS:2896]为汉字输入码缓冲区计数器。

一个快速实用的字串替换程序

杭州浙江电子学校　　金林樵

用手工进行字串的替换，是件头痛的事，搞不好，漏掉其中的一个字串没有修改，则程序的正常运行就会受到影响，特别是替换变量名时，就使程序产生错误的运行结果。下面的程序可为你在这方面排忧解难，使用它可以进行快速批量替换，既准确又迅速，它可使原文中的所有相同短字串替换成所需的长字串，也可使原文中的所有相同长字串均改为所需的短字串。

运行此程序时，输入的文件必须是 ASCII 码文件。按提示输入要编辑的文件名、替换和被替换字符串后，程序就将此文件读入内存，然后快速替换，替换的结果在屏幕上显示出来，以利检查。替换结束后，在当前盘上产生一个扩展名为“XRF”的同名文件，而原文件保留不变。新文件运行通过后，可将扩展名改为原类型。

```
100 REM 字符替换
110 CLEAR:KEY OFF:CLS
120 J＝0:M＝999:OPTION BASE 1:DIM P$(M),P1$(M)
130 INPUT “请输入文件名:”,FILE$
140 OPEN FILE$ FOR INPUT AS #1
150 WHILE NOT EOF(1)
160 COUNT＝COUNT＋1
170 LINE INPUT #1,P$(COUNT)
180 WEND
```

PC 机磁盘文件的加密与解密

上海中国纺织大学计算机系　刘忠达

在讨论 PC 机磁盘文件的加密与解密以前，先讲述一下磁盘结构。双面的 $5\frac{1}{4}$ 英寸软盘每面各有 40 个磁道，每道又分九个扇区，每个扇区有 512 个字节。DOS 把所有扇区划分为四个部分：

1. 引导程序区(IPL)

它占有 0 扇区，存放格式字、规格表和自举程序。

2. 文件分配表(FAT)

文件分配表占有 1－4 扇区，纪录文件占有空间，在 FAT 中存放每个文件占有簇(Cluster)的链接表。PC－DOS 规定每两个扇区为一簇，每簇占 1024 字节空间。FAT 共有一式两份文件分配表，各占两个扇区。

3. 文件目录表(DIR)

文件目录表从 5 扇区到 B 扇区，占有 7 个扇区，用来存放磁盘文件目录。

4. 资料区

从 C 扇区开始，共 608 个扇区，362496 个字节。这个数字也就是用不加 S 参数的 FORMAT 命令格式化后得到的磁盘空间。资料区用来存放文件正文。

文件目录表共可存放 112 个文件目录。PC－DOS 对目录的每个字节的含意都作了明确的规定如下：

0—7：磁盘文件名

8—A：扩展名

B：文件档案属性，标准规定如下：

01：表明该文件为只读文件；

02：表明该文件为隐含文件；

10：表明为一个子目录档案；

20：正常文件属性

C—15H：未用

16H—17H：文件建立时间

18H—19H：文件建立日期

1AH—1BH：起始簇号

1CH—1FH：文件长度

明白了上述磁盘结构，借助 DEBUG 调试工具便可做一些简单的加密与解密工作。

许多人掌握的简单加密方法有以下三种：

1. 在文件名中加入隐含字符；

2. 在文件名中插入半个汉字；

3. 修改文件属性，达到加密目的。

对于方法 1 和方法 2 的解密，可利用 DEBUG 的 E 命令将目录项中的 0—A 字节改为一般字符的 ASCII 码(十六进制)即可。对于方法 3，也可用 E 命令将文件属性改为正常文件。这样就完成了解密工作。

下面介绍一种新的加密方法。我们知道目录项中的 C—15H 字节未用，我们可利用其中的 C 和 D 两字节来进行文件的加密工作。方法如下：

1. 进入 DEBUG

2. 利用 L 命令将磁盘的目录区调入内存

3. 利用 D 命令找到欲加密文件的目录项

4. 读该目录项 1AH 和 1BH 字节的内容，设为 0E，00，用 E 命令将它们写入该目录项的 C 和 D 字节。

5. 将该目录项的 1A 和 1B 字节置为任意数值(不同于原来数值)。

6. 用 D 命令察看，此时目录项中的起始簇号已被改变。

7. 利用 W 命令将目录区写回磁盘。

8. 退出 DEBUG。

若要加密多个文件，只要重复步骤 3，4，5。

当执行被加密的文件时，计算机无法正常读盘，给出出错信息。

解密是加密的逆过程，步骤如下：

1. 进入 DEBUG

2. 利用 L 命令将磁盘的目录区调入内存。

3. 将起始簇号(存于目录项的 C 和 D 字节)写回目录项的 1A 和 1B 字节。

4. 利用 W 命令将目录区写回磁盘。

5. 退出 DEBUG。

详细操作过程附后。

```
190 CLOSE #1
200 FILE$=LEFT$(FILE$,INSTR(FILE$,"."))+"XRF"
210 INPUT "键入原字串:",SEARCH$
220 INPUT "替换成:",AP$
230 FOR I=1 TO COUNT
240 PTR=INSTR(P$(I),SEARCH$)
250 IF PTR=0 THEN 290
260 ZC$=RIGHT$(P$(I),LEN(P$(I))-PTR-LEN(SEARCH$)+1)
270 P1$(I)=P1$(I)+LEFT$(P$(I),PTR-1)+AP$:P$(I)=ZC$
280 J=J+1:GOTO 240
290 P1$(I)=P1$(I)+ZC$:ZC$=""
300 IF P1$(I)<>"" THEN PRINT P1$(I) ELSE P1$(I)=P$(I)
310 NEXT I
320 OPEN FILE$ FOR OUTPUT AS #1
330 FOR I=1 TO COUNT
340 PRINT #1,P1$(I)
350 NEXT I
360 CLOSE #1
370 END
```

```
A:\>debug
-l 100 0 5 7
-d 1a0
56DE:01A0  41 53 43 49 49 20 20 20-42 41 53 20 00 00 00 00  ASCII   BAS ....
56DE:01B0  00 00 00 00 00 00 F3 08-21 00 1F 00 F9 00 00 00  ......?!...y...
56DE:01C0  51 55 57 45 49 4D 41 20-42 41 53 20 00 00 00 00  QUWEIMA  BAS ....
56DE:01D0  00 00 00 00 00 00 54 00-21 00 20 00 BD 00 00 00  ......T.!..?...
56DE:01E0  4B 45 59 32 20 20 20 20-42 41 53 20 00 00 00 00  KEY2    BAS ....
56DE:01F0  00 00 00 00 00 00 55 0A-21 00 21 00 37 01 00 00  ......U.!.!.7...
56DE:0200  42 41 53 49 43 41 20 20-43 4F 4D 20 00 00 00 00  BASICA   COM ....
56DE:0210  00 00 00 00 00 00 00 60-68 06 0E 00 80 65 00 00  .......`h....e..
-e 20c 0e 00
-e 21a 22 22
-d 200 21f
56DE:0200  42 41 53 49 43 41 20 20-43 4F 4D 20 0E 00 00 00  BASICA COM.....
56DE:0210  00 00 00 00 00 00 00 60-68 06 22 22 80 65 00 00  .......`h."".e..
-w 100 0 5 7
-q
A:\>basica
Sector not found error reading drive A
Abort,Retry,Ignore? a
A:\>debug
-l 100 0 5 7
-e 21a 0e 00
-w 100 0 5 7
-q
A:\>basica
```

硬盘正确格式化一例

袁士彬

微机在使用过程中，若不小心被病毒严重感染，系统不能正常运行时，需要对硬盘格式化；有时在编制涉及系统调用的程序时，往往会破坏系统，这时亦需要对硬盘格式化。有时因操作不小心而致使系统丢失，屏幕显示：

SYSTEM HALTED

这时你只要插入与原 DOS 匹配的系统盘重新启动微机，打入如下命令：

A>SYS C:↙

即可恢复硬盘系统。如果你用了不匹配的系统盘，特别是用了较原版本高的系统盘进行如上操作的话，就必须对硬盘进行格式化。根据我们在实际工作中的经验，如果格式化方法不当，是不能装入 DOS 系统重新使用硬盘的，下面给出实际工作中正确格式化硬盘一例。

根据判断认为硬盘需要格式化后，先用完好的 DOS 软盘启动微机，用 FDISK. COM 在硬盘上建立 DOS 分区，FDISK 给出以下五种选择：

①Create DOS Partition
②Change Active Partition
③Delete DOS Partition
④Display Partition Data
⑤Select Next Fixed Disk Drive

选①建立 DOS 分区，屏幕显示：

DOS Partition Has Created

再对硬盘格式化：

A>Format c:/s/v↙

这时屏幕提示敲入 C 盘的卷标，但无论你敲入的卷标如何，均不对硬盘进行格式化。我们分析可能是因为 DOS 分区实际上已被破坏的缘故，这时需要使用 LOWFORM. EXE 文件进行低级格式化，但经常对硬盘进行低级格式化是很不利的，而且一般系统盘中不配有 LOWFORM. EXE 文件。既然 DOS 分区被破坏了，那么我们用 FDISK. COM 文件先将原 DOS 分区删除，再重新建立一个 DOS 分区，因而得到硬盘正确格式化方法如下：

1. 用系统盘启动微机后，键入如下命令：

A>FDISK↙

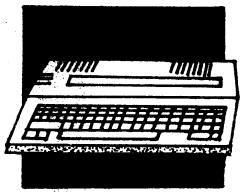
在屏幕给出提示后选③，删除原 DOS 分区。待再次给出五种选择时，再选①重新建立 DOS 分区。

2. 对硬盘进行格式化并装入系统。

A>FORMAT C:/S/V↙

3. 用 COPY 命令装入常用的 DOS 文件。

至此，即可得到一个正常可用的微机系统。



学习机之友

程序出错处理及 RESUME 的功能扩充

北京机工印刷厂　丁志伟

在有错误处理的 BASIC 程序中，常会遇到 ONERR GOTO 和 RESUME 语句，下面介绍功能和用法。

1. ONERR GOTO。设置一个转向行号，常放在程序开始。执行这个语句之后，运行时遇到程序有错，就转到它指定的错误处理程序，例如：

```
10 ONERR GOTO 1000
20 A=3;B=5
……
1000……
1020 RESUME
```

第 10 句执行过之后，20 句本意是要执行两个赋值语句，但误把冒号打成分号，出现了语法错误，这时就会转到 1000 句的错误处理程序。如果程序中没有发生错误，ONERR GOTO 则不会产生影响。

2. 上面的小程序中，从 1000 句开始为错误处理程序。它应能根据错误性质做相应的处理。最后一条 RESUME 语句类似子程序的 RETURN，它使程序运行返回到发生错误的位置。如果处理得当，程序就象没发生过错误一样，继续运行。

3. 错误处理程序不应再出错。它的第 1 条语句常是 POKE216,0。其作用是清除 ONERR GOTO 设置的标志。此后再发生错误时，程序就中断运行，避免因错误处理部分发生错误而陷入死循环。

下面重点讨论 RESUME 语句。

前面提到，RESUME 语句的作用是返回到发生错误的位置。但是有些错误难以处理，比如磁盘存取错误；或者其它原因，希望在处理错误之后（也可能不处理错误）转到其他行号，RESUME 就难以满足要求了。这时可以想到用其他语句转出，比如用 GOTO 语句。但是，用其他语句脱离错误处理程序在某些情况下是行不通的。下面举例说明，见程序 1：

```
10 ONERR GOTO 1000
500 A=3/B
505 PRINT A
510 STOP
1000 POKE 216,0
1005 B=1
1010 RESUME

]TRACE↙
]RUN↙
#10#500#1000#1005#1010#500#505 3#510
Break in 510
```

第一次执行 500 句除数 B 为 0，计算发生错误，因为有了 10 句的设置，程序接下来会转到 1000 句。1000 句清除出错转向标志。此后程序再出错，BASIC 解释程序会发出错信息并停止运行。1010 句的 RESUME 返回发生错误之处，也就是 500 句。以上执行情况可以看出，程序运行正常。如果把 1010 句改为 GOTO500 或 GOSUB500，效果也相同。在这个小程序中，RESUME、GOTO、GOSUB 这三个语句做为错误处理程序的结尾都能使程序正常运行。再看下面的程序 2：

```
10 ONERR GOTO 1000
20 GOSUB 500
30 STOP
500 A=3/B
505 PRINT A
510 RETURN
1000 POKE 216,0
1005 B=1
1010 RESUME
```

程序 2 所要完成的工作和程序 1 相同，只是把除法改由子程序做。再看执行情况：

```
]RUN↙
#10#20#500#1000#1005#1010#500#505
3#510#30
Break in 30
]1010 GOTO 500↙（改变 1010 句）

]RUN↙
#10#20#500#1000#1005#1010#500#505
3#510
? RETURN without GOSUB error in 510
```

当 1010 句是 RESUME 时，子程序工作正常，而改为 GOTO500 后，就会使 GOSUB 好象没有执行过。RESUME 到底做了些什么工作呢？看看它的处理程序就清楚了：

```
F318-  A5 DA   LDA   $DA
F31A-  85 75   STA   $75   恢复出错时行号
F31C-  A5 DB   LDA   $DB
F31E-  85 76   STA   $76
F320-  A5 DC   LDA   $DC   恢复出错时扫描指针
F322-  85 B8   STA   $B8
F324-  A5 DD   LDA   $DD
F326-  85 B9   STA   $B9
F328-  A6 DF   LDX   $DF   恢复堆栈指针
F32A-  9A      TXS
F32B-  4C D2 D7JMP   $D7D2 执行下一条语句
```

这段程序中使用了几个零页存储单元：

$DA、$DB：存放发生错误时的行号；

$DC、$DD：存放发生错误时的程序扫描指针；

$DF：存放发生错误时的堆栈指针。

$75、$76　存放正在执行语句的行号。

$B8、$B9　执行程序时用来扫描程序的指针。

RESUME 的功能就是恢复出错时的三个指针，使程序能返回到原来出错的位置上继续运行。$F318～$F327 的功能完全可以由 GOTO 语句完成。关键在于 $F328～$F32A 这两条，即恢复栈指针。

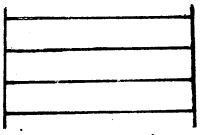


图 1

图 1 是程序 2 使用堆栈情况的示意。执行 GOSUB 语句时，用到了堆栈，错误发生时，也用到了堆栈；堆栈的功能，是后进先出，进栈和出栈要匹配。1010 句如果改为 GOTO500，直接转到 500 句，继而执行 510 句 RETURN 时，堆栈指针却还指向出错时栈顶位置，出栈的内容不是 20 句 GOSUB 入栈的内容，入栈和出栈不匹配，BASIC 解释程序就会认为运行有误。

程序 1 之所以能正常运行，是因为出错之前没有调用含错误的子程序，不存在入栈、出栈不匹配问题。为了使问题更加明显，再请看程序 3：

```
10 ONERR GOTO 1000
20 A=3/0
30 STOP
1000 PRINT PEEK (223),
1005 IF PEEK (223)< 90 THEN STOP
1010 GOTO 20
```

地址 223($DF)存放出错时栈指针。程序出错和进入子程序时，都会有数据入栈，栈指针数值变小。

```
NO TRACE↙
]RUN
248     239     230
221     212     203
194     185     176
167     158     149
140     131     122
113     104     95
86
Break in 1005
```

由于程序不断出错，错误处理程序又没使用 RESUME，数据不断入栈。6502 的堆栈只能存放 256 个字节，一旦堆栈装满，就会造成难以恢复的混乱。如果把程序 3 的 1005 句去掉再执行，就会看得更清楚。

综上所述，想让程序安全运行，在没把握时，出错处理的结束不要轻易用 GOTO 代替 RESUME。

若不想返回原来出错位置，又该怎么办呢？为此我编了一段机器语言子程序(程序 4)：

```
0300-   20  0C  E1     JSR    $E10C
0303-   A5  A1         LDA    $A1
0305-   85  DA         STA    $DA
0307-   85  50         STA    $50
0309-   A5  A0         LDA    $A0
030B-   85  DB         STA    $DB
030D-   85  51         STA    $51
030F-   20  1A  D6     JSR    $D61A
0312-   38            SEC
0313-   A5  9B         LDA    $9B
0315-   E9  01         SBC    #$01
0317-   85  DC         STA    $DC
0319-   A5  9C         LDA    $9C
031B-   E9  00         SBC    #$00
031D-   85  DD         STA    $DD
031F-   60             RTS
```

这段程序的作用是，在执行 RESUME 之前，先把出错处理后返回的行号和扫描指/针计算出来，修改出错指针($DA～$DD 内容)。执行 RESUME 时，程序就会返回指定的位置。子程序 $EI0C 将 USR(n)送到浮点累加器 $9D～$A2 的值转换成二字节整数后存入 $A0(高位)和 $A1(低位)。子程序 $D61A 根据 $50、$51 单元提供的返回行号；找到扫描指针后存入 $9B、$9C。最后将扫描指针减一，调整到上一 BASIC 语句行的结束标志处，再存入 $DC、$DD。

程序 4 在使用时要和 BASIC 程序配合，用 USR(n)函数传递出错处理后返回的行号到浮点累加器 $9D～$A2。在执行 USR(n)前，应先在 $0A～$0C 中放一条 JMP 指令，以便转到程序 4。例如程序 5：

```
1000 POKE 216,0
1005 B=1
1010 HH=505
1015 POKE 10,76:POKE 11,0:POKE12,3
1020 HH=USR(HH)
1030 RESUME
```

1010 句的 HH 被赋予执行 RESUME 时想转去的行号，这里定为 505。1015 句将处理 USR(n)函数的进入地址放入 $0A～$0C 单元。这是一条跳转指令 JMP $300。1020 句通过 USR(n)函数将 HH 的值放入浮点累加器($9D～$A2)，然后执行 $0A～$0C 单元的指令 JMP $300 进入程序 4。通过程序 4，$DA～$DD 的内容已被修改，以后执行 RESUME 时，就会返回行号 505。如想转去其它行号，可对 1010 句的 HH 赋予相应的值。

值得注意的是，如果是子程序出错，最好不要让错误处理子程序直接转出子程序，尤其是子程序嵌套的情况下更是如此。应当先返回到子程序内部由 RETURN 语句退出子程序或者执行完 RESUME 语句后再执行一个 POP 语句。原因也是为了堆栈的进栈和出栈相匹配。例如想在出错后转到第 10 句，可以这样：

```
1010 HH=2000
1015 HH=USR(HH)
……
1030 RESUME
2000 POP:GOTO 10
```

特殊 6502 指令

李 铁

笔者 87 年编了一个 6502 跟综程序，在调试中，无意跟综了一个“???”指令，发现这个指令长度并不是一个字节，而且对寄存器也有影响。经分析，发现这是 6502 设计师在设计 6502 时，未禁止除合法指令外的其它代码而产生的误动作。

最近见一些刊物上介绍这些指令，但多不全面。如有的称“6502 未公布指令”就不妥，因为并不是 6502 设计师有意设计的。为使读者能正确使用这些指令，现将研究资料整理发表。（见封二、三表①－④）

这些代码存在着规律性，这种规律性是由 6502 译码电路决定的。产生误动作的原因（原理）如下：

指令码读入指令寄存器后，指令译码器将 D_0VD_1，为零触发 A 类（表①）指令处理，$D_0=1$ 触发 B 类（表②）指令处理，$D_1=1$ 触发 C 类（表③）指令处理。尔后再进一步处理，如 B 类，由 $D_7D_6D_5$ 确定操作性质，$D_4D_3D_2$ 确定寻址方式（周期，指令长度）等。

$D_0\wedge D_1$ 非零未定义，但也未禁止。所以当 $D_0\wedge D_1=1$ 时，$D_0D_1=11$，同时触动了 BC 两类指令的一些操作，进而产生了误动作。基本原理就是这样。

由于未禁止非指令码，所以几乎所有非指令码都能产生误动作，但有明显作用的只有 66 条。另还有 0×××0010，1××10010 两类指令，也有作用（死机），其原因是这些代码触发了 CPU 循环执行一些操作。

关于误动作指令执行所需周期，表④所列的，由 $D_4D_3D_2$ 三位决定，可参照表②对应的指令（有移位动作＋2，否则相同）。

误动作指令可用于加密；编程可节省空间；执行一些特殊操作等。但这些指令只能用于 6502，6502A 芯片，而对 65C02（Apple Ⅱ C 机芯片）却不能用，因为在 65C02 中，这些误动作代码已被定义为新的指令。

（注：表中 6 条指令含意保留，因为这 6 条指令动作复杂，实用性不强，所以留做加密用。）

四舍六入法程序

浙江金华粮食学校　　韦肖鸿

四舍五入是大家熟悉的一种数字修约规则。但大量的实践表明，采用四舍五入法修约数据，会造成系统误差，使结果的误差增大，因为采用四舍五入法处理数据，进位的项数有 5 个，舍去的项数则是 4 个，数字修约时进位的可能性比舍去的可能性要大。因此，修约后数值的平均值会出现偏离原来实际数值的平均值的情况。为了克服四舍五入中的上述缺陷，人们提出采用四舍六入法来修约数据，其具体规则如下：

四舍六入五考虑，五后非零则进一，五后皆零看奇偶，五前为偶应舍去，五前为奇则进一。

这种修约规则可使有的 5 进位，有的 5 舍去，从而让因修约而引起的正负误差有相互抵销的机会，达到进舍项数相平衡，使舍入误差总和等于零的目的。

在 BASIC 语言中，对四舍五入法修约数据，一般就直接通过 INT(X)函数实现。如给出下式：

INT(X＊10^N＋0.5)/10^N

就可对数据 X 保留 N 位小数，并对小数点后第 N＋1 位进行四舍五入的处理。

对数据进行四舍六入的修约没有专门的函数能够实现，这时可编制一个专门的子程序来摸拟四舍六入的功能。下面的子程序供大家参考。

```
100  X$=STR$(X):L=LEN(X$)
105  FOR I=1 TO L
110  IF MID$(X$,I,1)="." THEN 125
115  NEXT I
120  GOTO 170
125  Y$=MID$(X$,I+N+1,1)
130  IF VAL(Y$)>=6 THEN X=X+10^(-N):GOTO
     170
135  IF VAL(Y$)<=4 THEN 170
140  FOR J=I+N+2 TO L
145  IF VAL(MID$(X$,J,1))<>0 THEN X=X+10^
     (-N): GOTO 170
150  NEXT J
155  Y1$=MID$(X$,I+N,1)
160  IF VAL(Y1$)/2=INT(VAL(Y1$)/2) THEN170
165  X=X+10^(-N)
170  X=INT(X＊10^N)/10^N
175  PRINT X
180  RETURN
```

以上程序中的变量 N 是要求保留小数的位数，X 是要求修约的数据。将该子程序插入到一些有关的数据处理程序中，就可实现对数字的四舍六入修约，从而可保证数据处理结果的更合理与更高准确度。

“怪”指令在软件加密中的应用

舟山市电力公司变电工区　傅剑

本刊90年第8期(总第65期)《利用BOOI1对磁盘进行机密》一文,虽然能起到保护效果,但笔者认为此文加密过程有如下四点缺陷:

1. 该文加密程序不但占用了DOS3.3程序修补区中的APPEND和VERIFY程序修补区,而且覆盖了LDX $2B指令(取得槽号*16,以便进入BOOT2),这样的BOOT1进入内存后,将会使DOS命令APPEND、BSAVE、SAVE出错。

2. 必须使用NIBBLES AWAY、ZAP等扇区修改软件,没有这些软件的用户就无法用该方法加密。

3. 对每片加密磁盘都需修改0轨0区的内容,因此加密过程繁琐,不适合大量制作。

4. 该文作者只考虑了设置保密口令,而对RESET和起动跟踪都没有防范措施,因此很容易被破解。

笔者针对上述四点的缺陷,同样对BOOT1进行一些修改,但不同的是直接修改BOOT1在内存中的映象,并且避开了DOS3.3程序修补区,这样不但无需利用工具程序来修改软盘0轨0区中的BOOT1,而且在DOS3.3进入内存后,不影响APPEND、BSAVE、SAVE等命令功能。修改后的BOOT1映象可以用DOS命令中INIT直接写入被加密磁盘,操作简便,适于大量制作。笔者在BOOT1中也设置了口令保护(口令是:1968111),而且封锁了RESET功能,并用目前知者不多的6502怪指令编写关键加密程序,使起动跟踪破密者不明究竟,而放弃跟踪。具体做法如下:

1. 开机,引导正常的DOS3.3

2. 输入CALL－151命令,进入监控模式。

3. 输入如下机器语言程序:

```
B645－  AB        ???                ;地址B645～B649
B646－  B3        ???                 为6502怪指令,
B647－  8F        ???                 相当于正常指令
B648－  4B        ???                 LDA#B3和STA
B649－  08        PHP                 $84B
B64A－  4C 5E 08  JMP  $085E

B6B3－  AB        ???                ;地址B6B3～B6C6
B6B4－  5E 8F 4B  ???  $4B8F,X        为6502怪指令,
B6B7－  08        LSR                 相当于正常指令:
B6B8－  AB        PHR
B6B9－  00        ???                 LDA#5E
B6BA－  8F        BRK                 STA$84B
B6BB－  F2        ???                 LDA#00
B6BC－  03        ???                 STA$3F2
B6BD－  AB        ???                 LDA#C6
B6BE－  C6 8F     DEC  $8F            STA#3F3
B6C0－  F3        ???                 LDA#63
B6C1－  03        ???                 STA$3F4
B6C2－  AB        ???
B6C3－  63        ???
B6C4－  8F        ???
B6C5－  F4        ???
B6C6－  03        ???
B6C7－  20 58 FC  JSR  $FC58
B6CA－  AD FC 08  LDA  $08FC
B6CD－  20 ED FD  JSR  $FDED

B6D0－  A9 00     LDA  #$00
B6D2－  A2 00     LDX  #$00
B6D4－  20 0C FD  JSR  $FD0C
B6D7－  DF        ???
B6D8－  F5 08     SBC  $08,X          ;地址B6D7～
B6DA－  DD F5 08  CMP  $08F5,X         B6D9为6502
B6DD－  D0 D4     BNE  $B6B3           怪指令,相当于
B6DF－  DF        ???                  正常指令:DEC
B6E0－  F5 08     SBC  $08,X           $8F5,X
B6E2－  E8        INX
B6E3－  E0 07     CPX  #$07
B6E5－  D0 ED     BNE  $B6D4          ;地址B6DF～
B6E7－  20 2F FB  JSR  $FB2F           B6E1为6502怪
B6EA－  CF        ???                  指令,相当于正
B6EB－  EF        ???                  常指令:DEC
B6EC－  08        PHP                  $8F5,X
B6ED－  A6 2B     LDX  $2B
B6EF－  6D FD 08  ADC  $08FD
B6F2－  CF        ???
B6F3－  EF        ???                 ;地址B6EA～
B6F4－  08        PHP                  B6EC为6502怪
B6F5－  B2        ???                  指令,相当于正
B6F6－  BA        TSX                  常指令:DEC
B6F7－  B7        ???                  $8EF
B6F8－  B9 B2 B2  LDA  $B2B2,Y
B6FB－  B2        ???
B6FC－  BF        ???
B6FD－  00        BRK
B6FE    B6 09     LDX  $09,Y
```

地址B6F5～B6FB为密码值,现设置为1968111,可任意修改为你想设置的任何口令,但必须注意输入口令字符的ASCII码时要加上1。(因为加密程序中的6502怪指令将会使密码减1处理,这是为了防止被启动跟踪而设置的)

4. 取出DOS3.3系统盘,插入需加密盘,键入INIT命令对磁盘进行格式化。当格式化完成后,就制

成了一张具有口令、防 RESET 和起动跟踪的加密盘。

经过上述处理的系统盘启动后，屏幕的左上角首先出现一个“?”，要求输入口令，当你输入正确的口令，才能启动 DOS，否则系统将封锁 DOS，确保安全，你必须重新启动该磁盘才能再输入口令。

编者的话：我们结合《6502 特殊指令》一文向读者介绍傅剑同志这篇应用文章。有关“怪”指令的操作的准确性质及稳定可靠问题，仍需通过大量的实践证明，欢迎大家提供这方面的材料。

双显示电子钟

宁夏中卫县中学　高二班　万斌

在 1989 年第一期《青少年计算机》杂志上曾发表过一篇《双显示电子钟》。该程序十分冗长，虽便于阅读，却给读者输入带来极大不便。并且该程序所实现的功能并不十分理想。例如：没有设置秒针，并且分针每一分钟才跳动一下。为此，我编了一个“双显示电子钟”程序。为了便于输入，须对程序进行优化、简化的工作。这就要对时、分、秒针在每一时刻的位置进行算法分析。经过精打细算，我确定了算法，并运用大量的逻辑运算来实现。程序中极少运用 IF…THEN 语句。如遇到秒数到零，分数需加 1 时，程序仍运用逻辑运算。

程序运行后会询问：“HOUR MINUTE SECOND (时、分、秒)”，这时用户输入即时时间，然后按任何一键，程序钟即开始工作。屏幕左上角有一电子表，正中间有一机械钟，时、分、秒针不停地运转。用户可对时钟误差进行调整，方法是对空循环的终值进行调整。

有兴趣的读者，不妨对此程序进行剖析，利用其中的逻辑判断进行其他程序的编制，并可对程序进行修改，使机械和电子表移至高分辨率第一页混合显示。

```
5  PR#3:HGR2 :HOME :R(1)=45:R(2)=60:R(3)=75:HCOLOR=3
10  INPUT "HOUR, MINUTE, SECOND"; H, M, S: HGR2:HOME:HH=H+M/60:MM=M+S/60-.1
20  PI=3.14159265:R=80:FOR I=0 TO 2*PI STEP PI/6:GOSUB100:HPLOT X,Y:NEXT:P=PI/2:K(1)=HH/3:K(2)=MM/15:K(3)=S/15:GET A$
30  FOR I=1 TO 3:A(I)=P*K(I)-P-(K(I))1)*2*PI:B(I)=2*PI-A(I):NEXT:C(1)=PI/21600:C(2)=PI/1800:C(3)=PI/30
40  FOR J=1 TO 3:I=A(J):R=R(J):GOSUB 100:HCOLOR=0:HPLOT139,95 TO X(J),Y(J):HCOLOR=3:HPLOT 139,95 TO X,Y:X(J)=X,Y(J)=Y:NEXT:MUSIC 100,1
50  FOR I=1 TO 3:A(I)=A(I)+C(I)-(A(I)+C(I))B(I))*2*PI:NEXT
60  VTAB 1:HTAB 1:PRINT H;":";M;":";S;SPC(3);:S=S+1-(S=59)*60:M=M+(S=0)-(M=59 AND S=0)*60:H=H+(M=0 AND S=0)-(H=11 AND S=0 AND M=0)*12
70  FOR I=1 TO 385:NEXT:GOTO 40
100  X=139+R*COS(I):Y=95+R*SIN(I)*99/100:RETURN
```

（上接第 42 页）

四、其它 IC 故障：

现象：1. 开机后，喇叭不响，屏幕无任何反应，指示灯亮。查 A_2、B_2、B_{13}、C_1、C_2、A_{14}。

2. 开机后，喇叭会响，但屏幕没有反应。查 A_2、A_8-A_{10}、B_2、B_{10}、B_{13}、C_2、$C_{11}D_{11}$-D_{14}。

3. 开机后，喇叭不响，没有进行 RESET 设定，屏幕出现杂乱的字形。查 A_{13}、B_5-B_8、B_{11}、C_1、C_2、C_{14}、E_{12}、E_{13}、$F_{12}-F_{14}$、G_1-G_5、$G_{10}-G_{12}$、C_3-C_{10}、D_3-D_{10}、E_3-E_{10}RAM。

4. 文字正常，但图形和画面故障。查 A_8-A_{11}、B_4、B_5、B_8、B_9、$B_{12}B_{13}C_3-C_{10}$、C_{12}、C_{14}、E_2、E_{11}-E_{14}、F_{14}、G_1。

5. 字体错误或变形。查 A_3、A_5、B_5-B_8。

6. 按键与出现的文字不相符。查 A_3、B_2、B_3、B_{11}、B_{13}。

7. 喇叭不响，其它正常。查 F_{13}、J_{13}、Q_4、C_{11}、C_{12}电容及喇叭连线与插头。

8. 光标不正常。查 A_3、B_2、B_3、B_{11}、B_{13}。

9. 磁带机存取信息故障。查录音磁带机的输出输入与主机连接是否正确，磁带机的音量控制是否适当，板音磁头与原录音头不正。F_{13}、H_{14}、J_{13}、K_{13}。

10. 不按键时，会出现文字。或按一次键会出现两个以上的字符，健盘工作不正常。这种情况可查 B_6、B_7、F_{13}、A_{12}、B_{10}、C_{11}。另外键盘本身的故障以及排线中容易受外界干扰也会造成以上所述的故障现象。

11. 无彩色信号。查主机板上晶振边的 50P 微调电容。调整彩电的色饱和钮以及 B_{12}、B_{13}。

按以上所列的现象，维护人员不必掌握较深的维修技术，采用替代法或比较法以及常规的电子测量仪器，就可把损坏的主机修复。

苹果机汉字造型表自动生成程序

济南西站　刘士明

为解决苹果机在高解析度状态无法显示汉字及字符的问题，我编制了这个程序，可将您在屏幕上所造的汉字及其他一切图形、字符自动转换为向量造型表，并可存入磁盘文件备用。使用时，只需将该造型表读入内存，在高解析度状态下，用 DRAW 命令调用您所造的汉字，并且能够以任意大小，角度显示。

程序执行时，首先询问您所要造的汉字个数，然后在屏幕左侧显示一个 15×16 的点阵，左上角闪烁的小方框为光标。使用下列功能键，可在点阵中移动光标和置光点，帮您排出要造的汉字点阵字形：

按 I、J、K、M 键移动光标；

按 S、X、Z、Y 键上下左右置光点。

为造出优美的字形，在点阵右侧开辟了一个反向显示的“小窗口”：当您在点阵中造字的同时，按 D 键，它同步显示您已排的字形，使您知道该在点阵中何处置光点，才能造出最理想的汉字。

在程序中，还定义了下列功能键，帮助您制作汉字造型表。当所排点阵不满意时，按 H 键可随意清除点阵中的任何光点；按 Q 键清除所有光点；按 E 键结束当前汉字点阵的输入，转入下一个汉字。直至最后一个汉字造型生成，则询问是否保存该造型表，及程序是否继续，可按提示进行选择。若选择保存该造型表，则提示您输入一个文件名后，将所造的汉字造型存入磁盘。

对于生成的汉字造型，使用时，按下列步骤操作：

＜1＞将造型表文件调入内存：BLOAD　文件名

＜2＞使计算机进入高解析度状态：

HGR：POKE 232,0：POKE 233,96

＜3＞定义颜色、旋转角度和放大倍数。

＜4＞显示造型表中的汉字：

DRAW N AT X,Y

第＜4＞步显示表中第 N 个汉字于屏幕坐标(X,Y)处，颜色、旋转角和放大倍数由第＜3＞步定义。

```
10  TEXT :HOME
20  BL$ = CHR$(7)
30  VTAB 22 : INPUT"ENTER SHAPE NUMBER:";N: IF N<1 OR N>255 THEN 30
50  ST = 24576:BYTE = 0:ADDR =ST + 2* N + 2:S = ADDR
60  POKE ST ,N:POKE ST+ 1,0 :POKE ST+2,N * 2 +2: POKE ST + 3,0
70  BYTE = 0:ADDR =S:A =0: P = 1
80  POKE 768,1: POKE 769,0: POKE 770,4: POKE 771,0: POKE 772,44: POKE 773,62: POKE 774,0
90  POKE 232,0:POKE 233,3
100  HGR :POKE -16302,0:HCOLOR =3:SCALE=5: ROT=64
110  FOR I =15 TO 155 STEP 10
120  FOR I2=15 TO 165 STEP 10
130  HPLOT I,I2
140  NEXT I2,I
150  X2 = 1:Y2 = 1:I=1
160  T = PEEK (-16384):POKE -16368,0
170  X3 = 10 * (X2-1)+15:Y3 = 10 * (Y2-1)+15:XDRAW 1 AT X3,Y3:XDRAW 1 AT X3,Y3
175  IF T<127 THEN 160
180  IF T= 217 AND X2<15 THEN X2=X2+1: POKE ADDR,67: POKE ADDR +1,1: POKE ADDR+2,6: ADDR = ADDR+3: GOTO 380
190  IF T=218 AND X2 > 1 THEN X2=X2-1: POKE ADDR,67: POKE ADDR+1,3:POKE ADDR+2,6:ADDR = ADDR+3: GOTO 380
200  IF T =202 AND X2 > 1 THEN X2=X2-1: POKE ADDR,3: GOTO 370
205  IF T=196 THEN HCOLOR =3: POKE ADDR,0: FOR I2= 203 TO 223:HPLOT I2,50 TO I2,75: NEXT :HCOLOR=0:POKE 233,96:SCALE= 1:DRAW P AT 205,55:POKE 233,3:SCALE =5:HCOLOR=3: GOTO 160
210  IF T=203 AND X2<15 THEN X2 =X2+1: POKE ADDR,1: GOTO 370
220  IF T=205 AND Y2<16 THEN Y2 =Y2+1: POKE ADDR,2: GOTO 370
230  IF T=197 THEN 440
240  IF T=211 AND Y2>1 THEN Y2=Y2-1: POKE ADDR,67: POKE ADDR +1,67: POKE ADDR+2,6: ADDR = ADDR+3:GOTO 380
250  IF T =216 AND Y2<16 THEN Y2 =Y2+1: POKE ADDR,67:POKE ADDR +1,2: POKE ADDR+2,6: ADDR= ADDR+3:GOTO 380
260  IF T =201 AND Y2>1 THEN Y2 =Y2-1: POKE ADDR,67:GOTO 370
270  IF T=209 THEN 70
280  IF T<> 200 THEN GOTO 160
290  HCOLOR= 0: GOSUB 390:HCOLOR=3 : HPLOT X1,Y1:ADDR = ADDR -1:M1 = PEEK (ADDR):M2 = PEEK (ADDR-1):M3=PEEK (ADDR-2)
300  X2 = X2+(((M3=67) AND (M2=3) AND (M1 =6) OR (M1=3))-(((M3=67)AND (M2=1) AND (M1 =6)) OR (M1=1))
310  Y2=Y2+(((M3=67)AND (M2=67) AND (M1 =6) OR (M1=67))-(((M3=67)AND (M2=2) AND (M1 =6)) OR (M1=2))
320  IF Y2< 1 THEN Y2 = 1
330  IF X2 < 1 THEN X2 = 1
350  IF M1 <>6 THEN 160
```

```
360  ADDR = ADDR-2: GOTO 160
370  ADDR = ADDR+1:GOTO 160
380  HCOLOR=3: GOSUB 390:GOTO 160
390  X1 = 10*(X2-1)+15:Y1= (Y2-1)*10+15
400  SCALE=2:A=1
410  FOR I I=1 TO 64:ROT=I I:DRAW 1 AT X1,Y1:
NEXT
420  SCALE=5:RETURN
440  POKE ADDR,0:ADDR=ADDR+1:HCOLOR=3
460  BYTE=ADDR-ST:S=ADDR
480  P=P+1:IF P) N THEN PRINT BL$:GOTO 520
490  A= INT (BYTE/256):B =BYTE-A*256:POKE
ST+2*P,B:POKE ST+2*P+1,A
500  BYTE = 0: GOTO 80
520 HOME :POKE -16301,0:VTAB 22:PRINT "BSAVE
TABLE TO DISK(Y/N):";:GET Y$:IF Y$<>"Y" AND Y
$<>"N" THEN 520
530  IF Y$="N" THEN 570
540  HOME: VTAB 22: INPUT "ENTER TABLE
NAME:";NA$: IF NA$=""THEN 540
550  LE = ADDR-ST
560  PRINT CHR$ (4);"BSAVE" NA$",A";ST;",
L";LE
570  HOME :VTAB 22:PRINT"END PROGRAM (Y/
N)?";:GET Y$: IF Y$="Y"THEN HOME :TEXT :END
580  IF Y$<>"Y"AND Y$<>"N" THEN 570
590  HOME:GOTO 30
```

高效率的排序方法

陈 庆 祥

一、问题的提出.

在大多数实际问题中，排序的对象并非是一个单纯的数组，而是具有密切联系的若干数组的集合。例如某中学一次英语竞赛的有关统计资料如下表：

考号	姓 名	性别	年龄	班级	笔试	口试	总分
0001	郑 丽	女	15	102	92	77	
0002	肖剑文	男	16	201	84	79	

上表中总分按笔试的60%与口试的40%之和计算。现要求能按任一数据关键项排序。以下是按总分由高到低排序(设有5个学生)：

```
10  INPUT "N=";N: REM 学生竞赛记录分类程序 1.
20  DIM A%(N),B$(N),C$(N),D%(N),E%(N),F
(N),G(N),S(N)
25  FOR I = 1 TO N:A%(I) = I
30  READ B$(I),C$(I),D%(I),E%(I),F(I),G(I)
40  S(I) = INT (.6*F(I)+.4*G(I)+.5): NEXT I
50  FOR I = 1 TO N-1:FOR J=I+1 TO N
70  IF S(I)>= S(J) THEN 120
80  T=A%(I):A%(I) =A%(J):A%(J)=T
85  T$=B$(I):B$(I) =B$(J):B$(J) = T$
90  T$ = C$(I):C$(I) =C$(J):C$(J) = T$
95  T = D%(I):D%(I) = D%(J):D%(J) = T
100  T= E%(I):E%(I) = E%(J):E%(J) = T
105  T= F(I):F(I) = F(J):F(J) = T
110  T= G(I):G(I) = G(J):G(J) = T
115  T= S(I):S(I) = S(J):S(J) = T
120  NEXT : NEXT
130  PRINT"考号  姓  名  性别  年龄  班级  笔试
口试  总分  名次"
140  FOR I = 1 TO N
150  PRINT RIGHT$("000"+STR$(A%(I)),4)"  "B
$(I)"  ";
160  PRINT C$(I)"  "D%(I)"  "E%(I)"  "
F(I)"  "G(I)"  "S(I)"  "I
170  NEXT I:END
1000  DATA "郑 丽","女",15,102,92,77
1005  DATA "肖剑文","男",16,201,84,79
1010  DATA "唐学英","女",14,101,94,86
1015  DATA "陈大勇","男",16,303,95,90
1020  DATA "李 涛","男",17,301,80,85
]RUN
N=5
```

考号	姓 名	性别	年龄	班级	笔试	口试	总分	名次
0004	陈大勇	男	16	303	95	90	93	1
0003	唐学英	女	14	101	94	86	91	2
0001	郑 丽	女	15	102	92	77	86	3
0005	李 涛	男	17	301	80	85	82	4
0002	肖剑文	男	16	201	84	79	82	5

程序1排序的效率很低，其原因是所有数据项必须随总分的交换而交换，其次，程序1还缺乏处理并列名次的能力，比如同为82分的两名学生名次却不相同。

二、利用下标嵌套进行索引排序

减少数据交换量是提高排序效率的关键。利用下标嵌套的方法进行索引排序(以记录号即考号为索引项)，可以避免记录号以外的任何数据交换，从而简化处理过程并大大提高排序的效率，程序如下：

```
10  INPUT "N=";N: REM 学生竞赛记录分类程序 2.
20  DIM A%(N),B$(N),C$(N),D%(N),E%(N),F
(N),G(N),S(N),H%(N)
25  FOR I = 1 TO N:A%(I) = I
30  READ B$(I),C$(I),D%(I),E%(I),F(I),G(I)
40  S(I) = INT (.6*F(I)+.4*G(I)+.5): NEXT I
50  FOR I =1 TO N -1:L = I
60  FOR J = I +1 TO N
70  IF S(A%(L))<S(A%(J))THENL= J
```

```
80  NEXT :T=A%(I): A%(I) = A%(L) :A%(L) = T
90  GOSUB 300: NEXT : GOSUB 300
100  PRINT TAB(10)"表  一:(按总分名次打印)":PRINT
110  PRINT "考号  姓  名  性别  年龄  班级  笔试
口试  总分  名次"
120  FOR I = 1 TO N:M = A%(I):P = H%(M):GOSUB
200:NEXT: PRINT
130  PRINT TAB (10)"表  二:(按原来顺序打印)":PRINT
140  "考号  姓  名  性别  年龄  班级  笔试  口试
总分  名次"
150  FOR I = 1 TO N:M =I:P = H%(I):GOSUB 200:
NEXT:PRINT
160  END
200  REM 打印各条记录
210  PRINT RIGHT $ ("000"+STR $ (M), 4)"    "B
$ (M)"    ";
220  PRINT C $ (M)"    "D%(M)"    "E%(M)
"    "F(M)"    "G(M)"    ";
230  PRINT S(M) SPC(6-LEN(STR $ (S(M))))P: RE-
TURN
300  REM 处理并记录名次
310  K=K+ NOT (S(A%(I))=(S(A%(I-1)))
320  H%(A%(I)) = K: RETURN
1000  DATA "郑  丽","女",15,102,92,77
1005  DATA "肖剑文","男",16,201,84,79
1010  DATA "唐学英","女",14,101,94,86
1015  DATA "陈大勇","男",16,303,95,90
1020  DATA "李  涛","男",17,301,80,85
]RUN
N=5
    表    一:(按总分名次打印)
考号  姓  名  性别  年龄  班级  笔试  口试  总分
名次
0004  陈大勇  男  16  303  95  90  93  1
0003  唐学英  女  14  101  94  86  91  2
0001  郑  丽  女  15  102  92  77  86  3
0005  李  涛  男  17  301  80  85  82  4
0002  肖剑文  男  16  201  84  79  82  4
          表    二:(按原来顺序打印)
考号  姓  名  性别  年龄  班级  笔试  口试  总分
名次
0001  郑  丽  女  15  102  92  77  86  3
0002  肖剑文  男  16  201  84  79  82  4
0003  唐学英  女  14  101  94  86  91  2
0004  陈大勇  男  16  303  95  90  93  1
0005  李  涛  男  17  301  80  85  82  4
```

程序 2 根据各记录号为自然数的特点,在比较总分时,仅交换各存贮记录号的下标变量 A%(L)与 A%(J)的值(第 70、80 程序行),其它数组一律不动,在数组 A%中记载了排序结果。这个程序比程序 1 的数据交换量减少了近 90%,效率高出好几倍,同时,不仅解决了并列名次问题,还可以两种不同方式输出排序结果,以满足不同需求。

三、高效率的映射索引排序方法

索引排序能大大减少排序时的数据交换,提高排序效率。但是,绝大多数的排序方法都离不开关键项数据的反复比较过程,因此当记录条数较多时仍然要花费较多的时间。利用映射索引排序则可以同时避免数据交换和比较这两个过程,因而具有极高的排序效率。

利用映射索引排序的程序如下:

```
10  INPUT "N=";N: REM 学生竞赛记录分类程序 3
20  DIM M(100)
30  FOR I =1 TO N:READ A $ ,B $ ,C%,D%,X1,X2
40  S = INT (.6*X1 + .4*X2 + 0.5)
50  M(S) = M(S)+ 1:NEXT
60  FOR I = 100 TO 0 STEP -1
70  K=K+ SGN (M(I)):M(I) =K
80  NEXT :RESTORE
90  PRINT "考号  姓  名  性别  年龄  班级  笔试  口
试  总分  名次
100  FOR I =1 TO N:READ A $ ,B $ ,C%,D%,X1,X2
105  S = INT (.6*X1 + .4*X2 + 0.5)
110  PRINT RIGHT $ ("000"+STR $ (I),4)"  ";
115  PRINT A $ "    "B $ "    "C%"    "D%
"    "X1"    "X2"    "S"    "M(S)
120  NEXT :END
1000  DATA "郑  丽","女",15,102,92,77
1005  DATA "肖剑文","男",16,201,84,79
1010  DATA "唐学英","女",14,101,94,86
1015  DATA "陈大勇","男",16,303,95,90
1020  DATA "李  涛","男",17,301,80,85
]RUN
N=5
考号  姓  名  性别  年龄  班级  笔试  口试  总分
名次
0001  郑  丽  女  15  102  92  77  86  3
0002  肖剑文  男  16  201  84  79  82  4
0003  唐学英  女  14  101  94  86  91  2
0004  陈大勇  男  16  303  95  90  93  1
0005  李  涛  男  17  301  80  85  82  4
```

程序 3 仅用单循环就完成了排序过程,无需任何数据交换和比较,其效率高于其他任何排序方法。程序 3 的主要特点是数组 M 的使用。程序的第 50 行中,以总分 S 为索引项,用数组 M 存放各种分数的个数;而在第 70 行中又用它存放各个分数的名次,同时也解决了并列名次的问题。这种利用同一数组先后存放两类相关数据的技巧,充分说明了灵活地使用下标去管理数组的意义。这种排序方法在用 DATA 语句提供数据时,只需定义一个固定容量的数组 M(100)。这样当 N 较大时,可以节省大量的存储空间;而且以简单变量取代数组还可进一步提高处理效率(对 1000 条记录排序所用时间不到 60 秒)。当作为索引的分数含有一位小数时,可扩大 10 倍使之成为整数,再将数组容量扩大 10 倍。

演示抛物面的光学效应

成都市东城区教师进修校　王晓林

抛物面卫星信号接收天线、太阳灶反光罩以及汽车灯反光罩等都应用了抛物面的一个重要物理原理，即它能把一组平行光线反射汇聚于焦点，又能把焦点处的点光源散射光反射成一组平行光线。这是中学物理和数学中要介绍的重要内容。但是，在上课时很难直观演示这两个可逆的过程。我编了下面的程序，不仅可以在屏幕上直观地看到这两个过程的“慢镜头”，而且还可看到，一组同步推进的平行光线，虽然被抛物面反射的先后不同，但却是同时汇聚于焦点。这一现象在教材中往往被忽视。但是，正因为有这一效果，才能确保电波所携带的信息汇聚于焦点时不会失真。此外，由于参数 P 是由用户自由输入的，因此可以看到各种开口大小的抛物面的反射和汇聚效应是一致的。

本程序显示的实际上仅是抛物面的轴截面上的现象。程序的关键技巧在于能保证各条光线不论是平行状态还是散射状态下都要能“等速”同步推进。而且由于各条线与抛物面相遇是不同时的，因而还要准确处理每条光线何时开始反射及反射后的方向，而且要保证反射和未反射的光线仍能同步推进，其原理如下：

各条光线上每次只显示一个推进点；然后利用直线的参数方程($x=x_0+l\cos t$，$y=y_0+l\sin t$)计算下一个点，并判断该点是否已达到抛物面，若已达到，则计算光线反射后的前进方向，再算下一个点的坐标。若未达到则显示该点，如此循环，直至全光路显示完毕。

程序运行时，输入参数 P($y^2=2px$ 中的 P)。然后依次显示两次可逆过程。显示中可由空格键控制。显示完后，敲 ESC 键可继续。

本程序是在 CEC—I 中文状态下通过。若要在 APPLE 机上实现，只需去掉所有含汉字的程序行(8900,9080,9019,9005,9006,9157)。

```
8900  PR# 3: PRINT :HGR2
9000  DIM B0(4),Y1(4),X1(4),Y(4),X(4),A(4),C
      (4)
9003  HOME :VTAB 2:PRINT "Y^2=2PX"
9005  VTAB 4: PRINT "抛物线物理意义"
9006  PRINT "对声光的反射和汇聚作用"
9010  VTAB 6: PRINT"0< P<10"
9015  INPUT "P=";P:P0 =P/2: IF P<1 THEN 9010
9016  IF P>10 THEN 9010
9019  A$ = "平行光汇":B$ ="聚至焦点"
9020  F = 0:M0 =277:N0 =190 :M=M0/3:N=N0/
      2
9025  K=M0/8:XL = 2 * M/K:L0 = XL+P0
9030  GOSUB 10000
9050  FOR J=1 TO 4:X(J) = -1:Y(J)=0
9055  Y=J * N/K/5:Y1(J) = Y:X=Y * Y/P0:B0(J)
      =XL-X:X1(J)=XL
9060  R=X+P0:A(J)= -Y/R:C(J)=(P0-X)/R
9065  NEXT J
9070  FOR Y= 0 TO N/K STEP .03
9073  X=Y * Y/P0: IF X>XL THEN 9076
9075  HPLOT M+K * X,N-K * Y:HPLOT M +K * X,
      N+K * Y
9076  NEXT Y
9080  VTAB 2:PRINT A$ :PRINT B$
9090  FOR I= 0 TO L0 * K/2
9092  FOR J = 1 TO 4
9095  IF I/K <B0(J)/2 THEN XS = X(J):YS = Y(J):
      GOTO 9105
9100  YS = A(J):XS = C(J)
9105  X2 = X1(J)+2 * XS/K:Y2 = Y1(J)+2 * YS/
      K:HPLOT M+K * X2,N-K * Y2:HPLOT M+K
      * X2,N+K * Y2
9107  V = PEEK (49152): IFV = 160 THEN 9107
9110  X1(J) = X2:Y1(J) =Y2
9112  NEXT J
9115  NEXT I: F0 = F0+1
9130  IF F0=2 THEN 9160
9140  FOR J =1 TO 4:B0(J)= L0-B0(J)
9145  X(J)= -C(J):Y(J)= -A(J)
9150  A(J) = 0:C(J) =1:X1(J) =P0:Y1(J)=0
9155  NEXT J: GOSUB 10000
9157  A$ ="焦点光源":B$ ="变为平行光"
9158  GOTO 9070
9160  V= PEEK (49152): IFV= 141 THEN 9003
9165  GOTO 9160
10000  HGR2:HCOLOR=3
10002  HPLOT 1,N TO 279,N TO 273,N-2 TO 273,N
       +2 TO 279,N
10004  HPLOT M,190 TO M,1 TO M +2,5 TO M-2,5
       TO M,1
10006  IF K< 3 THEN 10020
10008  F = M-K * INT (M/K):F1 = N-K * INT
       (N/K)
10010  FOR I = F TO 279 STEP K:HPLOT I,N-1 TO
       I,N+2:NEXT
10012  FOR I =F1 TO 190 STEP K:HPLOT M-1 ,I TO
       M+2,I:NEXT
10020  RETURN
```

LASER QZT/C 应用经验

北师大附属实验中学 张海翔

QZT/C 汉化高显系统经《电子与电脑》杂志于八八年九月首先介绍给读者后，经一年多使用，摸索了几点经验，现介绍给大家。

一、用外存保存变量

主机的 CSAVE 只能把程序录到磁带上，而程序的运行结果即变量值的保存，一直是一个难题。QZ 系统可胜任此项工作，见示例程序一、二。程序一先给 A、B(9)、C% 等变量运算赋值。40 行利用 QZ 特有的 NAME 语句。NAME 既可转录 BASIC 程序，也可转录二进制文件，而且可以用中文作文件名。40 行作用是取变量区首尾地址把整个变量区作二进制文件转录，并给该文件起了“变量库”这样一个直观明确的中文名字。程序二是调回“变量库”的例子。由于程序一、二长度不同，所以利用 BLOAD 的“变址调入”功能以使变量区调入后准确定位。应注意程序二中数组需定义(但不必赋值)而简单变量应先赋值，如统一赋为 0。调入“变量库”后 A、C% 等已不是 0 而是程序一的运行结果。

二、可显示的倒时钟

QZ 的时钟功能有趣又有用，它不仅不受干扰显示在屏幕右上角，而且还有一个专用保留字 TIMS $ 可随时在 BASIC 程序中调用显示时钟。QZ 还有一个“倒时钟”，TIME $ 调不出它，但有时又希望在程序中，如棋类程序作限时记秒用。这只要在程序开头加上程序三的语句就可随时用字符串变量 TR $ 调出倒时钟，其中使用了变量地址函数 VARPTR。因倒时钟长五位，所以 10 行给 TR $ 赋初值时也是五位。有趣的是如 10 行字串长 14 位，如 TR $ = “A B C D E F G H I J K L M N”则以后用 TR $ 可同时调出“倒时钟”和“顺时钟”。TR $ 是串变量，与保留字 TIME $ 性质不同，只能用在程序中，修改删除程序都会给 TR $ 赋空串值。

三、高显彩色绘图

QZ 的 V8 软件增加了高显彩色功能。SET、RESET、POINT 都和 MODE(1)用法一致。用 GET 和 PUT 可把一个区域的彩色图形方便快速地移到另一区。也可用 LINE 画彩色线，如

LINE(X,0)-(X,191)

其规律是当坐标 X 为偶数时画蓝色线，X 为奇数时画黄线，用相邻偶奇数执行两次 LINE 则为红线。在高显彩色屏幕上不显示 ASCII 字符，但仍可打入任何命令，只要击键正确就会被执行。如不执行还可以打入 SC1 或 SC0 命令转到高显文字状态，你前次打入的命令会显示在屏幕上以便检查错在什么地方。

四、中文词组加锁

QZC 有全拼音、区位、偏旁部首和词组四种汉字输入方式。在词组方式下用户可根据需要预先建立常用词组库。只需击几个英文键，就可快速输入几个到几十个汉字。QZ 词组库格式别致，原来就是普通 BASIC 程序，人人都是“不学自会”。但 BASIC 词库容易被 NEW 掉。为了保护词库，可以用 QZ 的加锁命令 LOCK，如

60000 LOCK 60000:END

这样 60000 行以后的词库就被保护了，加锁情况下要调入新程序可以用拼接命令。

NEW:MERGEL“文件名”(可省)

这样词库拼在新调入程序后面不会丢失。万一在未加锁情况下使用 NEW 命令，及时发现后执行 KEY 或 CALL&H4E19 或 MONE4E19，都能救回词组库。

程序一

```
10  DIM B(20):B(9)=3.45
20  A=10/3:C%=-10
30  PRINT“请启动磁带机，击空格键开始转录”:CALL &H3B2C
40  NAME “变量库”PEEK(&H78F9)+256*PEEK(&H78FA),PEEK(&H78FD)+256*PEEK(&H78FE):SCREEN
```

程序二

```
10  DIM B(20):B(9)=0
20  A=0:C%=0
30  BLOAD PEEK(&H78F9)+256*PEEK(&H78FA)“变量库”:PRINT
40  PRINT A;C%;B(9)
```

```
3.33333 -10 3.45
```

程序三

```
10  TR$=“12345”
20  A%=VARPTR(TR$)
30  POKE A%+1,&H97:POKE A%+2,&H79
40  PRINT TR$
```

第二讲　Turbo C 基本数据

李文兵

1. 基本数据类型

Turbo C 中使用的基本数据类型有 5 种(表 2.1)：

表 2.1　Trubo C 基本数据类型

数据类型	长度	数值范围
int(整型)	16 位	−32768～32767
float(单精度浮点型)	32 位	3.4E−38～3.4E+38
double(双精度浮点型)	64 位	1.7E−308～1.7E+308
char(字符型)	8 位	−128～127
void(无值型)	0 位	无值

2. **修饰符**

数据类型除 void 外，可依用途作如下修饰说明：

signed　有符号　unsigned　无符号

long　长整型　short　短型

这四个修饰符还可按如下组合使用：

signed　long　signed　short

unsigned　long　unsigned　short

与 char、int、double 组合可得，如下六种类型：

signed　short　int　unsigned　short　int

signed　long　int　unsigned　long　int

signed　double　unsigned　double

signed　char　unsigned　char

无符号字符和有符号字符的取值范围如下：

表 2.2　字符型数据的分类

类　型	取值范围
char	−128～127
unsigned char	0～255
signed char	−128～127

3. **常量**

常量是指其值固定不变的量，有如下五种：

(1)**字符型常量**　用单引号引起来的单个字符，如'a'，' '等。

(2)**整型常数**　可用 10、8 和 16 进制表示。用 8 和 16 进制表示时，在常数前要有前导词 0 和 0X，如：

123…10 进制数

0123…8 进制数

0X123…16 进制数

长整型常数的后面要加上 l 或 L，如 123L。

浮点型常数　它有 2 种表示方法，即：

①小数表示法　如 1.23；

②科学表示法，如 1.23e−3。

(4)**字符串常量**　用双引号引起来的字符序列，如"abc"。在存储器内部表为字符序列 abc\0。\0(Null)为结尾符。可见，字符与字符串是不同的。

(5)**控制字符常量**　ASCII 码值在 0X00～0X1F 之间的字符，即控制用字符，需采用反斜杠与特定字符组合表示，才能输入。控制字符如表 2.3 所示。

表 2.3 控制字符表

符号	ASCII 码	功能
\o	0x00	Null
\a	0x07	响铃
\b	0x08	退格
\t	0x09	水平制表
\f	0x0c	走纸
\n	0x0a	回车换行
\v	0x0b	垂直制表
\y	0x0d	回车
\\	0x5c	反斜杠
\'	0x2c	单引号
\"	0x22	双引号
\?	0x3f	问号
\DDD	0DDD	DDD 为 3 位及其以下 8 进制数
\xHHH	0xHHH	HHH 为 3 位及其以下 16 进制数

4. **变量及其初始化**

所谓变量是其值可变化的量。在程序中使用变量时，必须首先说明其数据类型，其语句格式如下：

数据类型　变量表；

注意；C 语句末尾必须有一个分号(;)，例如：

```
char a, b;
int i,j,k;
```

为使用这些变量，还必须先给出其值的大小，这叫变量初始化。例如

```
int i=10,j=20,k;
K=i+j;
```

变量名的组成规则如下：

①以字母或下线符(_)开始。

②能使用的字符有 a～z,A～Z,0～9,_和 $。

③有效长度为 32 个字符。

④大小写是有区别的。C 语言是以小写字母为主，书写的小写化是 C 程序的一大特点。

⑤不能使用 Turbo C 的关键字。

5. **函数**

结构的模块化，是 C 程序的又一大特点。函数是程序的基本模块，相当于子程序，定义形式如下：

函数类型说明　函数名(参数表)

```
参数类型说明
{
  说明语句
  执行语句
}
```

(1)**函数类型**　用前面所介绍的关于基本数据类型的关键字来说明，指出函数值的数据类型。

(2)**函数名**　标识符后紧跟一对园括号构成。

(3)**函数体**　函数体的界线符是一对花括号，其内容大致为两大类：说明语句和执行语句。

函数的定义中，必不可少的部分是：

```
函数名()
{
}
```

这是最小C程序。由于它的函数体没有实在内容，故什么也不执行。程序员常用它在程序中事先占据一个位置，好将来扩充一个有实际功能的函数。

6. 表达式

(1)**表达式的组成**　表达式是用运算符把操作数连接构成的式子。操作数可以是常量、变量和函数。

(2)**表达式的解**　指表达式中的操作数按运算符优先级进行运算，最终得到的表达式的值。

(3)**表达式的副作用**　在表达式中使用赋值运算符和递增(++)、递减(--)运算符时，将把值赋给某个变量，这就叫表达式的副作用(side effects)。

(4)**表达式语句**　表达式的后面加上一个分号(;)就构成一个表达式语句。C程序主要是由表达式语句构成的，语句的表达式化是C程序的又一大特点。

(5)**简单语句和复合语句**　C语言中的分号是语句的结尾符。含一个分号的语句叫简单语句，如：

```
K=i+j;
```

多个简单语句借助一对花括号可组成复合语句，如：

```
{ i=10;
j=20;
K=i+j;
}
```

7. 程序、文件、函数三者的关系

C程序的编译单位是文件。一个文件由一个或若干函数组成。一个可执行文件必须有一个名为main( )的函数，叫主函数。不管该函数放在什么地方，程序总是从它开始执行，到它执行完毕，整个程序也就执行完毕。其它函数由main( )或别的函数调用。

下面介绍一个简单的程序，以使读者了解C程序的风格，尽快掌握C程序的设计方法。

[练习 2.1]A> type li1.c

```
main( )
{ int i=10,j=20,k;
k=i+j;
print f("k=%d\n",k);
}
```

执行：A>li1

```
K=30
```

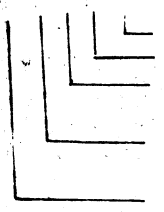
(1)该程序只含一个文件，文件名叫li1.c；注意，C源文件的扩展名必须是C。该文件只含有一个函数，这就是主函数main( )。

(2)2行为说明语句，说明i、j、k为整型变量；同时对i和j进行了初始化。

(3)3行是计算i+j的值的执行语句。该语句就是一个表达式语句。符号=是赋值运算符，表示把i+j的值赋给变量K，因此，该语句也叫赋值语句。

(4)4行是C语言中输出方式——调用输出函数。C语言的输入输出功能是靠调用C编译系统的库函数实现的。输入输出的函数化是C程序的又一大特点。函数printf( )是按格式输出函数，其功能是按照给定格式显示(或打印)表达式的值。打印格式如下：

```
                转换控制字符串   自变量K
printf("K=%d\n",k);
```



自变量，其内容为要显示的值
回车换行
转换控制字符，表示按10进制显示所对应的K的值
转换控制开始符，指定要显示的值的位置
原样显示的部分

printf( )函数的转换控制字符如表2.4所示。

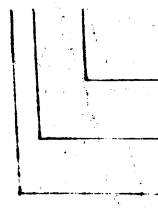
表 2.4　printf( )转换控制字符

转换控制	所指定的打印格式
%d	按10进制输出
%o	按无符号8进制输出
%x	按无符号16进制输出
%u	按无符号10进制输出
%c	按字符输出
%s	按字符串输出
%e	以men形式输出
%f	以小数形式输出
%g	采用%e和%f中较短的输出
%p	打印地址

在%与转换控制字符之间还可加上宽度说明。

无宽度说明时，若输出字符，则按字符的实际个数输出；若输出浮点数则按6位小数处理。

```
%-××××.××××∅
```



转换控制字符
小数点后面的位数
整数部分，表示输出的最小位数(小数占一位)
有负号靠左边输出，否则靠右边输出
自变量值存放开始位置

8. 按格式输入函数 scanf( )

(1)**功能和用法**　scanf( )也是一个系统库函数，

其功能是从键盘输入一个字符，用法是

scanf（转换控制字符串，变量1指针，变量2指针，……）；

[练习 2.2]

```
A>type li2.c
main()
{  int i,j
scanf("%d,%d",&i,&j);
printf("input:%d,%d\n",i,j);
}
```

执行：

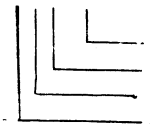
```
A>li2
246,48
input:246,48
```

该练习中scanf()各参数的名称和功能如下：

第1参数　第2参数　第3参数

（转换控制字符串）（变量指针）（j变量指针）

scanf("%d,%d",&i,&j);



变量j的指针

变量j的指针

转换控制字符

表示转换控制开始

数据分割符，可为逗号，空格、制表符、换行符

(2)转换控制字符

d——把输入看作是10进制整型数；

f——把输入看作是浮点数；

e——同f；

o——把输入看作是8进制数；

x——把输入看作是16进制数；

c——把输入看作是一个字符；

s——把输入看作是字符串；

h——把输入看作是短整型数；

i——把输入看作是10进制整型数；

p——把输入看作是指针；

n——输入视为整型数，其值为读入字符个数。

(3)scanf()函数的一些具体用法

①转换控制字符前的数字，用于指定数据位数；

②转换控制字符前面的L，用来表示读双精度浮点数或长整型数据；

③转换控制开始符%后面的*用来禁止赋值；

④如若转换控制字符串中有非转换控制字符，则输入数据时要在相应的部分输入与此相同的字符；

9.函数调用形式

(1)C程序结构　C程序总的看，是由函数定义组成的。注意，函数不能嵌套定义，但能嵌套调用。

(2)函数的调用关系　分为内部调用和外部调用。内部调用指一个文件内函数间的调用。外部调用指一个文件的函数调用另一个文件的函数。

(3)函数调用形式　最基本的是如下三种形式：

①函数语句调用　把被用函数的函数名直接写出，并以实参替换形参，圆括号后加以分号。被调用函数将作为一个独立的语句出现，如：

[练习 2.3]

```
A>type li3.c
main()
{ printf("I'm in main.\n");
aia();
}
aia()
{printf("Now I'm in aia..\n");}
```

执行：

```
A>li3
I'm in main.
Now I'm in aia..
```

②函数表达式调用　这种调用的特点是被调用函数出现在调用函数的表达式中。被调用函数要有返回值。在调用函数中，要说明被调用函数的数据类型，若其数据为整型，则无须进行类型说明。

[练习 2.4]

```
A>type li4.c
main()
{ float i=1.5;
double square();
i=square(i);
printf("%f\n",i);
}
double square(x)
double x;
{return(x*x);}
```

执行：

```
A>li4
2.250000
```

③函数参数调用　被调用函数以函数参数形式出现，其调用条件和注意事项同函数表达式调用。

[练习 2.5]

```
A>type li5.c
int i=2,j=5;
main()
{printf("%d,%d,%d\n",i,sum(),sub());}
sum()
{  int s;
   s=i+j;
   return(s);
}
sub()
{  int e;
   e=i-j;
   return(e);
}
```

执行：

```
A>li5
2,7,-3
```

10　数据的存储类别

在C语言中，变量除了有类型之分，还有四种存储类别，即：自动存储变量、静态存储变量、外部存储变量和寄存器存储变量。这些存储类别用来说明诸如变量的作用域、变量的生存期等存储性质。变量的作用域是指能访问该变量的场所。变量的生存期是指变量值

所能保留的时间。

(1)**自动存储变量**

①性质　是指其值在某一范围内得以保留,这一范围以外则消失的变量,它有如下性质:

·作用域的局部性　其作用域为该变量定义所在的模块(一对花括号围起来的部分)内或函数内。

·生存期的暂时性　其生存期为该变量定义所在函数或模块的执行周期,即一旦进入该函数或模块,C就自动地为该变量建立存储区,而一旦退出该函数或模块,C就自动地收回此存储区。

·未初始化的变量其值不确定,是无意义的。

②定义　定义自动存储变量用关键字auto,如

```
auto int i;j;
```

[练习 2.6]

```
A) type li6.c
main()
{int i=100,k=80;
printf("i=%d\n",i);
{int i=200;
printf("        i=%d k=%d\n",i,k++);
}
printf("i=%d k=%d\n",i,k);
}
```

执行:A> li6

```
i=1 00
    i=200 k=80
i=100 k=81
```

(2)**静态存储变量**

①定义　可按如下形式加以定义及初始化:

static　数据类型　变量名=初值;

②种类　有局部和全局两种静态存储变量。

·静态局部存储变量　仅能在所定义的模块内使用。与auto变量不同,它可保留原值不变。如函数:

```
count__up()
{static int number=0;
number+=25;
return(number);
}
```

其返回值为25;以后每调用一次,其值就增加25。这是由于在编译时就给静态存储变量number分配了存储空间,而以后该变量就永久地存在下去的缘故。

但变量number不能在它所在的函数外进行行访问,说明以这种方式定义的静态存储变量其作用域是局部性的。因此这种变量叫做内部静态存储变量。

·静态全局存储变量　如果把变量定义在模块外部,则所有函数都可以使用这个变量,例如:

```
static int number;
count__up()
{number+=25;
return(number);
}
reset()
{number=0;
}
```

这样定义的存储变量叫做外部静态存储变量。

③性质　静态存储变量有如下性质:

·作用域　内部静态存储变量的作用域为函数或模块内;外部静态存储变量的作用域为整个程序。

·生存期的永久性。

·未初始化的静态存储变量的值为0。

下面说明静态存储变量和自动存储变量的区别。

[练习 2.7]

```
A)type li7.c
main()
{count();
count();
printf("\n");
add();
add();
}
count()
{static int num0=0;
num0+=25;
printf("%d\t",num0);
}
add()
{auto int num1=0;
num1+=25;
printf("%d\t",num1);
}
```

执行:　A>li7

```
25      50
25      25
```

(3)**外部存储变量**

①用途　用于把大程序分割为若干程序单元(文件)而开发的场合。说明外部存储变量使用关键字extern。例如把一个程序分为两个单元来编译:

```
A>type file1.c
int i,j;
float  x;/* 定义外部存储变量i、j和x */
main()
{       :
func1();     /* 函数调用 */
func2();     /* 函数调用 */
}
A>type file2.c
extern int i,j;
extern float x;     /* 对外部存储变量的说明 */
func1();     /* 函数定义 */
{     :
}
func2();     /* 函数定义 */
{     :
}
```

(转第4页)

学用单片机

普及型单片机开发工具的硬件设计

中国科学技术大学计算中心　张培仁　刘振安

编者的话：单片机已经广泛用于机电产品、家用电器等社会生产、生活等各个领域。由于单片机的独特优点、已为广大电子爱好者所喜爱，为他们开发电子产品的智能应用，提供了无限广阔的天地。

大家知道，在实际应用中，特别是对爱好者来说，使用单片机，只要设计一个最小系统或者选用一个最小系统就可以了。关键是要结合自己的应用项目，装入控制该"应用项目"程序的 EPROM 芯片和相应的接口电路。对于业余爱好者来说，在编制某项专用程序时，不能纸上谈兵，必须有一个调试此程序（或叫系统软件）的工具。正如，电子爱好者用万用表、示波器等工具、调试收音机、电视机及其它电子产品一样，单片机的应用，也要通过开发工具进行调试，所不同的是，前者是调整硬件的工作状态；后者确要调试所需要的应用系统的硬软件。

如果购买一台专用单片机开发工具，其售价是爱好者无法接受的。为此，本刊特约中国科学技术大学计算中心微机室张培仁、刘振安同志为本刊组织并撰写有关单片机开发工具及典型最小系统应用方面的技术文章。他们选用 KDC—Ⅲ型单片机开发机介绍给读者，硬软件资料全部透明，并愿意为本刊读者提供咨询服务。对于他们为繁荣我国单片机应用事业和无私奉献的精神，我代表本刊工作人员和广大读者向他们表示深深的谢意。相信会有更多的作者和读者同我们一起办好本刊"学用单片机"专栏。

单片机开发工具（或称开发机）从使用角度上看，主要是解决编程、排错、仿真三方面功能。

1. 编程功能

单片机缺乏自身编程的能力，需要借助开发机来进行编程，所以，开发机应有机器语言、汇编语言、高级语言三种方式编程的能力。

2. 排错功能

任何比较复杂的程序常常不会一次编程就完全编好，而经常需要排错并修改程序。排错时首先要知道错在什么地方，这就要求开发机必须提供以下排错手段：

（1）单步：用户可以一次只执行一条指令。执行一条指令后即返回监控程序。

（2）运行：用户程序可以从任何一条地址启动，然后全速运行。

（3）断点运行：用户可以设置断点，当程序执行到断点时，控制返回到监控程序。

（4）检查和修改存储器的内容。

（5）跟踪：它能跟踪单片机运行时的每一指令周期中地址，数据，I/O 端口和控制总线上的信息。

3. 仿真功能

仿真就是指开发机能通过仿真器的硬件和软件"真实"地模拟所开发的应用系统的运行情况。开发机不但仿真单片机的 CPU，还能仿真存储器和 I/O。端口，也要仿真单片机的中断系统的运行，使用户能利用单片机尽量多的资源。在调试用户应用系统时，监控程序和用户程序都要使用单片机中的资源，从而完成相应任务。这时有一个控制权相互转移的问题，常常是监控程序启动用户程序，又在某种条件下（如碰到断点）从应用程序回到监控程序，因此在设计开发机监控程序时要尽可能少占单片机的资源。

开发机的种类：目前大致可分为三种。一种普及型开发系统即象单板机那样，有近 30 个键，可以和 PC 联机形成一种联机调试系统。具有编程、排错、仿真三大功能。第二种是独立型仿真器，可直接用汇编或高级语言编程，是一种在线实时仿真开发系统，能在同一种机型中开发多种芯片的单片机，系统扩充也比较方便。第三种纯软件单片机模拟仿真系统，无硬件设备，投资最少。它是用软件方法模拟仿真单片机硬件环境，指令系统应用及开发系统。它给开发单片机提供新的手段。

下面我们详细介绍一种普及型 KDC—Ⅲ型单片机开发机，供读者学习或选用，或根据我们所提供的硬软件资料自行制作。KDC—Ⅲ单片机开发机和目标系统共用一个 CPU8031。整个硬件成本较低，而性能上又可完全满足用户开发单片机的需要。

一、硬件结构

KDC—Ⅲ电原理图见图 1，它采用 8031 作为处理器，外接一片 2764（8KEPROM），一片 6264（8K 的 RAM）一片的 I/O 接口芯片。基本上是在单片机最小系统的基础上加扩充而构成。系统时钟采用内部时钟方式，即利用 8031 内部振荡电路，在 XTAL1、XTAL2 引脚外接 6MHz 晶体。内部的振荡电路处于自激振荡。复位开关采用电平方或复位，复位电路中电阻，电容参数和 CPU 的时钟频率及芯片最小复位时间有关。电源开启时可以自动进行复位。

开发机利用 8031 的 P0 口和 P2 口构成地址总线和数据总线。P2 口作为外部存储器的高 8 位地址的输出口。P0 口既是输出低 8 位地址又输出（或输入）相应数据的端口。P0 口在输出低 8 位地址时由 ALE 打入外部地址锁存器（74LS373）中，使地址锁存输出，在整个指令周期期间形成稳定的低 8 位地址。

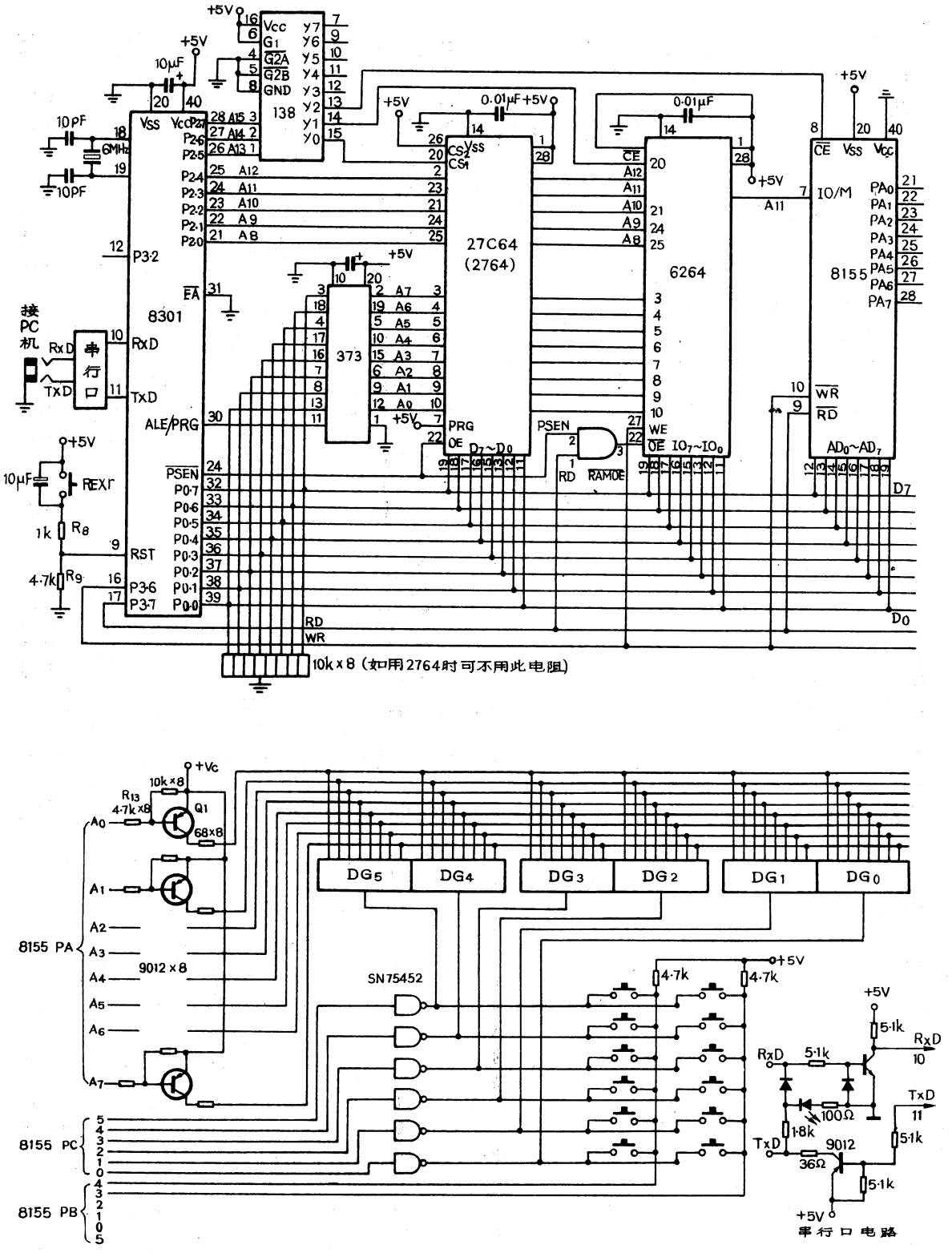

8301
138
373
27C64
(2764)
6264
8155
接PC机
串行口
RxD
TxD
RST
ALE/PRG
PSEN
RD
WR
10k×8 (如用2764时可不用此电阻)
8155 PA
8155 PC
8155 PB
9012×8
SN75452
DG5
DG4
DG3
DG2
DG1
DG0
串行口电路

图 1

8031 外部程序存储器由 $\overline{PSEN}$ 信号选通。8031 外部扩展 I/O(如 8155)由 $\overline{WR}$,$\overline{RD}$ 及地址译码器的信号选通。

开发机的地址分配如下:

Y0: 0000H ～ 1FFFH; Y1: 2000H ～ 3FFFH; Y2: 4000H ～ 5FFFH; Y3: 6000H ～ 7FFFH; Y4: 8000H ～ 9FFFH; Y5: A000H～BFFFH; Y6: C000H～DFFFH; Y7: E000H～FFFFH。

2764 的地址是 Y 数,作为程序存储器;6264 作为数据存信器,地址为 Y1。8155 的内 PRAM 地址是 4000～40FFH。8155 的 I/O 口,A 口地址:4101H,B 口地址:4102H,C 口:4103H,控制口::4100H。

开发机和 PC 通信通过 PCRS232 口(三条线)和开发机三芯插座相联接。通信波特率是 1200。

以下我们从三个方面,分析 KDC—Ⅲ开发机电路图。

1. 最小系统的扩充;

2. 最小系统与键盘、显示器的接口;

3. 最小系统与 PC 的通信。

一、最小系统的扩充

大家知道,MCS—51 最小系统应用时,经常需要加以扩充,因为扩充方法比较典型和规范,又都是选用常规芯片,所以,扩充是比较容易的。

1. 总线的扩充应用

从总线观点来看,单片机扩充主要是片外的地址总线,数据总线,控制总线三方面的扩充。

地址总线(AB)16 条,可寻址 64K。单片机用 P0 分时输出低 8 位的地址,P2 口输出高八位地址。因为低八位地址是分时输出的,所以要用锁存器锁存。在最小系统中 P0、P2 已经作为地址口使用,就不能再作为 I/O 口使用了。

八位数据总线(DB)由 P0 口输入输出。对外接的扩展片,一般由地址线控制片选信号。

控制总线(CB),主要扩展控制线有:$\overline{WR}$,$\overline{RD}$,$\overline{PSEN}$,ALE,$\overline{EA}$ 等。

在单片机最小应用系统扩充时,首先要注意是否还有未用的地址口。另外要考虑扩展口(P0～P3)本身负载能力。从理论上讲 P0 可以接 8 个 LSTTL 电路,其它口只能驱动 4 个 LSTTL 电路。实际上我们使用过的大部分 8031,负载能力要比理论上小。一般小于 5mA 时,I/O 的电平基本正常。在系统扩充时要注意电平的转换。一般接口是 CMOS 电路要接上拉电阻。在和高电压接口时最好通过光电耦合器(如 4N25)来完成。

2. 存储器的扩充

MCS—51 的存储器可分程序存储器和数据存储器。这两个存储器地址可以重叠,用单独的控制信号和指令可以把两者区分开。一般程序存储器由 $\overline{PSEN}$ 控制,读取数据用 MOVC 的指令。数据存储器用 $\overline{RD}$,$\overline{WR}$ 控制信号和 MOVX 指令,进行读写。

KDC—Ⅲ开发机,监控程序只有 2K 左右。而我们选用 2764(8K)作程序存储器的主要原因是为了以后扩充(实际上我们把一个通用系统的监控程序和子程序库扩充到了余下的 6K 中)。另一方面目前 2764 的市场价格还低于 2716,2732。此外 2764 和 27128、27256 都是 28 脚芯片,再扩充很方便。三种芯片区别如下表所示:

管　脚	2764	27128	27256
27	$\overline{PGM}$	$\overline{PGM}$	A14
26	N·C	A13	A13

从图中看出 2764 和 27256 只有二个脚有所区别,如需将 2764 换用 27256 时,可用跳线来解决 27,26 管脚的选线问题。

片选信号(20 管脚)$\overline{CE}$ 接法的考虑,在最小系统中如果只扩充 EPROM 时,可把 $\overline{CE}$ 接地。如果还有其它一些芯片也要扩充时,可以用最高位 P2.7 作为选通的控制信号。也可以将 P2 口高位用 74LS138 的输出 Y0～Y7 作选通信号。

数据存储器与程序存储器地址(0000H—FFFFH)重叠,可使用不同指令和控制信号来解决扩充中选址问题。6264 控制信号如下表:

6264 方式选择	$\overline{CE}$	$\overline{WE}$	$\overline{OE}$	功能说明
写	0	0	1	数据写入 6264
读	0	1	0	数据从 6264 读出
非选中	1	×	×	数据线高阻

6264 选的地址是 2000H—2FFFH

经常采用两种方法对扩充 6264(8K×8)进行读写。

其一,读写指令用 R0 和 R6 作低 8 位地址指针,高位地址由 P2 口单独输出。如下程序向 6264 中以 2100H 为开始地址的内存写数据。

程序如下:

```
        ORG 2000H
        MOV R0,#00H
        MOVA, #01H
RE1:    MOV P2, #21H
        MOVR1, #00H
        MOVX @R1,A
        INC R0
        DJNZ R0,RE1
HE1:    SJMJ HEI
```

其二,用 DPTR 作为 6264 的地址指针(KDC—Ⅲ监控程序中用此方法)。

```
        ORG 2000H
        MOV R0, #00H
RE2:    MOVA,#02H
        MOV DPTR, #2100H
        MOVX @DPTR,A
        INC DPTR
```

```
        INC R0
        DJNZ R0,RE2
HE2:    SJMP HE2
```

在最小系统中，数据存储器和程序存储器都是分开的，但有些场合希望两者合用。在KDC—Ⅲ开发机中把6264既作为数据存储器，也做为程序存储器。当PC机把用户的程序装入6264时，是把6264作为数据存储器使用，而在执行用户程序时，6264作为程序存储器使用。我们是用$\overline{PSEN}$和$\overline{RD}$信号相"与"的方法达到两者合用(见图1)的。当$\overline{PSEN}$与$\overline{RD}$都作为正与门的输入信号，产生低电平的读选通信号。可以用MOVX指令把"程序代码"写入6264中(利用$\overline{WR}$低电平有效信号)，还可用MOVX指令全读出修改之(利用$\overline{RP}$和$\overline{WR}$信号)，然后作为程序存储器使用，执行已写入的程序(利用$\overline{PSEN}$信号)。

最小系统扩充数据存储器(RAM)时，一般用静态存储器而较少地用动态存储器。主要原因扩充方便，不需要刷新，价格也较便宜。

3. I/O口的扩充

最小系统由于片外有EPROM，常用P2、P0口作为高低位地址使用。只有P1、P3两个I/O口可使用。如果I/O不够用，可进一步扩充I/O口，主要有两个方法：

1. 总线扩展方法。即用P0口作为数据口，而选片信号采取与数据存储器相同寻址方法。扩充I/O连接的外围设备，均与片外数据存储器统一编址。每一个I/O芯片占用一个片外ROM地址或一个片外ROM区域，而这时与程序存储器无关。在KDC—Ⅲ开发机中，373、6264、8155都用这个方法扩充的。

2. 串行口扩展方法。因为最小系统中利用串行口移位寄存器功能扩展I/O口。这种方法主要优点是占单片机的资源很少，接口简单，但相对并行口来说速度比较慢。

二、最小系统与键盘显示器的接口

某些最小系统在应用时，需要外接显示器和键盘。现以KDC—Ⅲ开发机为例，采用一片8155解决键盘扫描，数码管显示和最小系统的连接问题。有关电路图见图1。

1. 程控扫描显示电路

8031通过扩展接口8155的A口输出段码，0电平有效。C口输出位码，1电平有效。当六个数码管共阴极数码管全显示时，可逐次把所需的字符显示在规定字位上，每点亮一个数码管之后，稍停一段时间(1ms～2ms)，使之发光稳定，之后点亮下一个。这样巡回扫描显示速度较快，每秒可重复多次。虽然在不同时刻只有一个数码管工作，但利用人眼的视觉暂留效应和发光二极管灭时余辉效应，所以看到的是六个字符同时显示。这种巡回扫描显示需要靠程序来完成。下面给出KDC—Ⅲ开发机的扫描显示程序框图(见图2)。

2. 程控键盘扫描电路

在图1中，程序不断从PC口反门(即SN75452输出端)输出扫描码01H、02H、04H、08H、10H、20H。这些扫描码分别控制DG0～DG5六个数码管(共阴极)点亮。这个扫描码同时也扫描KDC—Ⅲ的12个键的小键盘。一旦用户有键按下，这时8155PB口的PB3、PB4就有0电平出现，单片机不断从PB口取走数据判断是否有键按下，如有键按下进入键分析程序和键处理程序，最后执行相应子程序。这种方法叫接口扫描法。

键盘扫描子程序KEYSCRN.ASM的程序框图(见图3)。

三、最小系统与PC的通信

由于单片机内(或外接)可以固化一定的控制程序，并带有相应的数据存储器和通信用的串行口，十分适合在小型通信系统中应用。可以最小系统为核心，连接多个最小系统的分布式系统，也可以连成以PC为主机和多个最小系统组成的分布式主从系统(或多级控制系统)。

结合KDC—Ⅲ加以说明，见电路图1中串行接口电路部分。PC/XT的RS232接口的输入输出分别采用MC11488和MC1489。我们采用9013，9012两只晶体管把RS232的标准电平－12V表示1，＋12V表示0转变为0～5V电平，以便和最小系统中的8031连接。D是发光二极管，当PC与最小系统通信时，有明暗变化。这种电路是一种准RS—232接口电路。

通信所用连线为双绞线，通信距离在30M以内。如果距离较远可用光电隔离或用电流环来驱动。由此可知，通信接口电路主要是解决电平转换问题，而通信的实施则是靠软件进行的，它们分别是单片机通信模块和PC机通信模块。

这二个模块分别由初始化模块，接收模块、和发送模块等三个子模块组成。下面介绍一下

PC与最小系统通信(如KDC—Ⅲ开发机)的过程。首先，把编译好的单片机通信程序送入单片机片外数据存储器某地址处(如2000H)开始的内存中。并运行这个通信程序(文件名DP.ASM)，这时开发机的监控程序已停止工作。

其次，在PC上编辑一个数据文件，如ABC、ASM文件。

```
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进行编译产生ABC.OBJ文件。

最后，在PC上运行PC通信模块(文件名是ZHAN KAN.PAS)。在PC上显示出提示，并要选择(1～3)。

```
1:PC向8031送数据
2:8031向PC传送数据
3:返回DOS。
```

应先选1，把ABC.OBJ文件送到单片机以2100H为起始的地址处。送完又返回提示。为了检查一下传

送情况，可再选 2。单片机外存将 2100H 处的 16 个(十进制数)字节内容传回 PC，形成一个文件名是 ABC1. OBJ，最后返回 DOS。此用 DOS 的 TYPE 命令分别显示 ABC. OBJ 和 ABC1. OBJ 二个文件的内容，即可证明通信过程的正常。

以上从硬件角度，结合 KDC—Ⅲ介绍了单片机最小系统及最小系统的扩充、最小系统与键盘、显示器的接口、单片机最小系统与 PC 通信等问题。

下讲从软件角度，结合 KDC—Ⅲ的监控程序的设计，通信模块的设计，并向读者提供全部程序清单。

本刊第三期刊载“普及型单片开发工具的软件设计”。

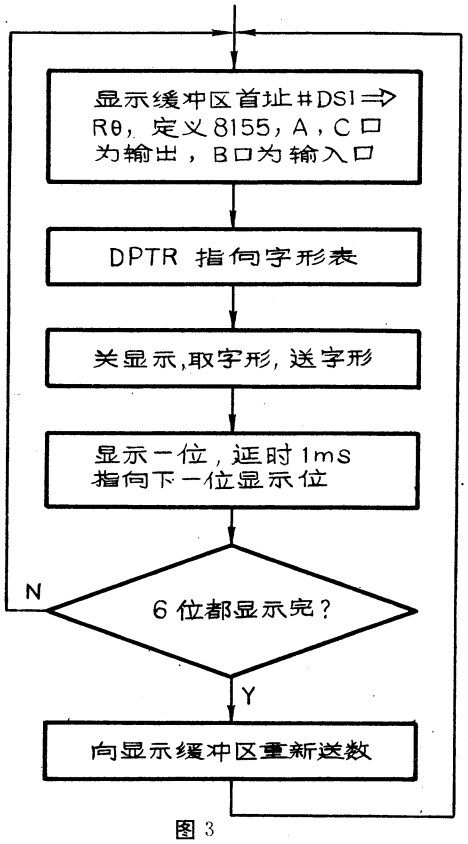


图 3

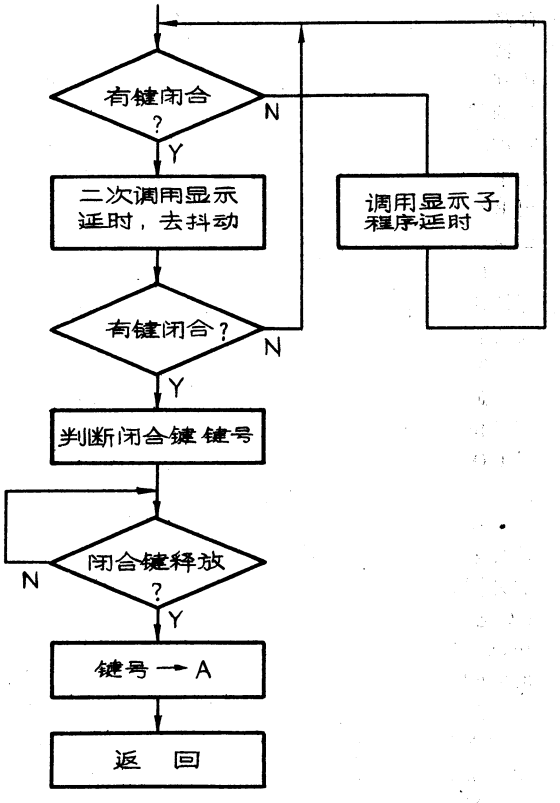
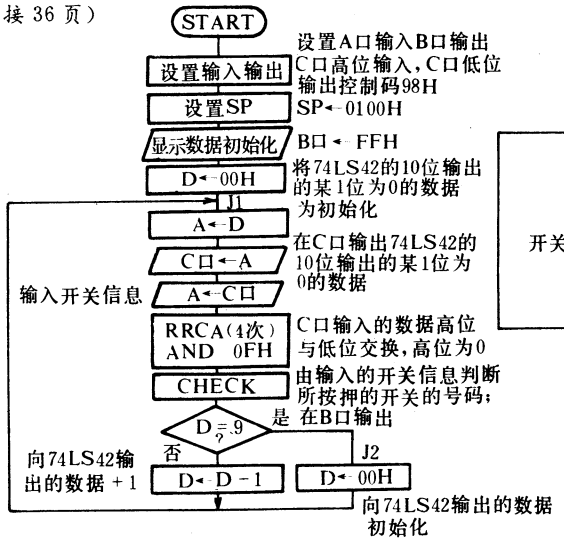


图 4

(接 36 页)



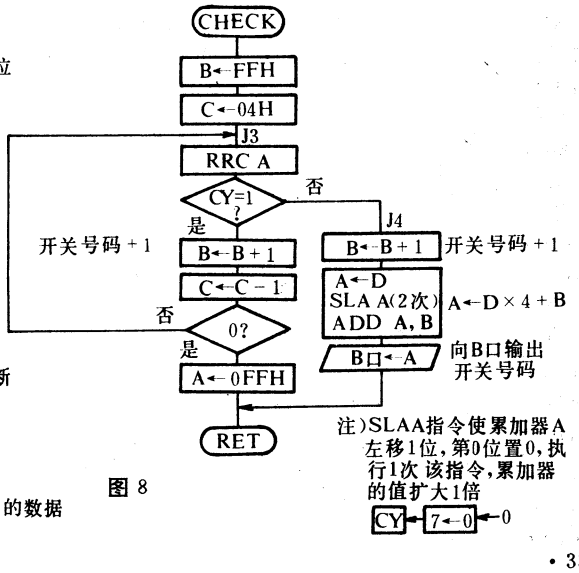


图 8

学装微电脑

学装 μp-80

40 支开关输入部件的制作

易齐干

微电脑控制机器工作时往往要接很多微动开关、按键开关、继电器与电磁铁等。通过 8255 将开关状态输入给微电脑，或驱使负载工作。

一片 8255 有 A、B、C 三组输入输出端口只能处理 24 个输入输出信息。如何增加 8255 的输入输出信息呢？这是本文要讨论的问题。

1. 动态开关扫描

本文介绍使用 μp—80 微电脑教育部件 8255 的端口 C，通过 74LS42 输入 40 支开关状态的方法——采用矩阵方式，即把全部开关划分为几个组，按组平均轮流输入开关状态，这种方法也称为动态开关扫描。仅用端口 C 开关输入电路如图 1 所示。端口 C 高位 PC7～PC4 为输入口，低位 PC3～PC0 为输出口，在不按动任何一支开关时，输入端为高电平（使用电阻对端口 C 高位进行上拉）。

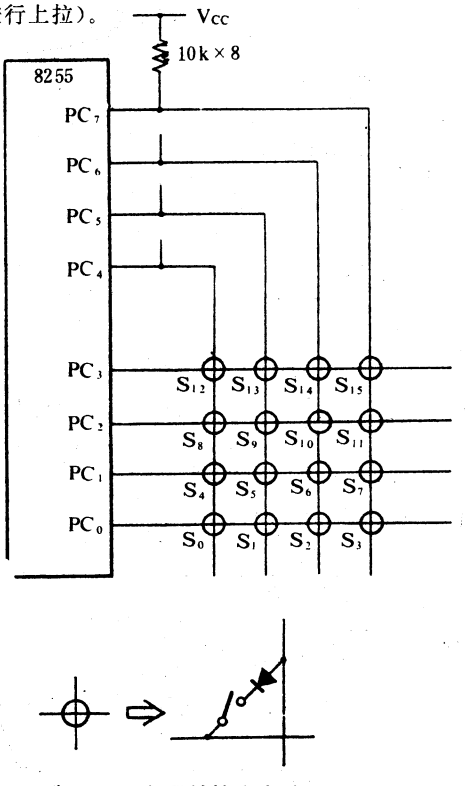


图 1　端口 C16 支开关输入电路

首先，由端口 C 输出 11111110（这时仅 PC0 为低电平。高 4 位设置成输入端之后，其信号决定于其它端口输出，所以，设置为 0 或 1 都可以）。当端口 C 高位输入低位的信息时，用程序查询 PC4、PC5、PC6、PC7 为 0 或是为 1。如 S0 为“通”，（按键）PC4 为低电平，故可查出 S0 被按动。如果 S2 为“通”，因为 PC6 为低电平，故查出 S2 被按动。查询按动的开关序号的程序流程图如图 2 所示。用 4×4 矩阵输入 16 支开关状态，是常用的键联接方法。

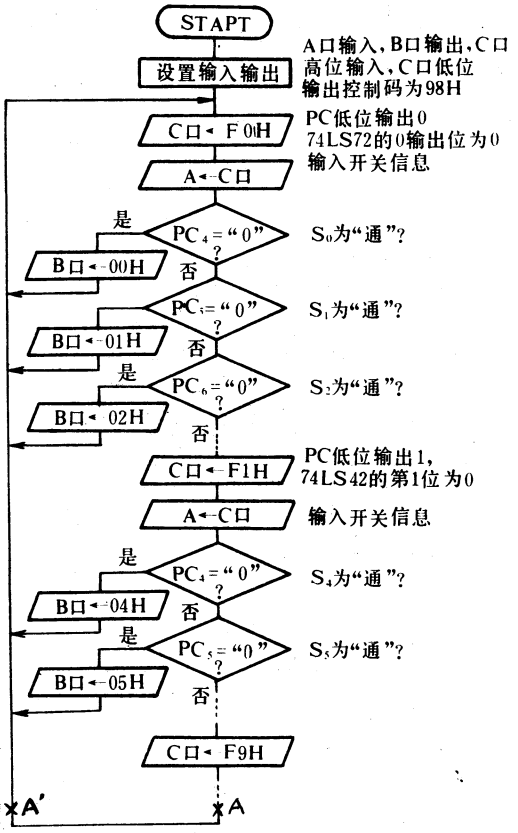


图 2　输入 16 支开关状态的流程图

2. 仅用 C 口，输入 40 支开关状态的方法

前面已经介绍了矩阵方式，要增加输入开关的数目。可使用 74LS42（BCD→十进制译码器）功能如图 3 所示，根据 A、B、C、D4 个输入端的状态，输出端 0～9 可选择其中之一为低电平。使用这种 IC 如图 5 所示，端口 C 低 4 位为输出端，与 74LS42 的 A、B、C、D 输入端相连，分组数目可达 10 组，所以能输入 40 支开关状态。

No.	输入				输出									
	D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	L	L	L	L	L	H	H	H	H	H	H	H	H	H
1	L	L	L	H	H	L	H	H	H	H	H	H	H	H
2	L	L	H	L	H	H	L	H	H	H	H	H	H	H
3	L	L	H	H	H	H	H	L	H	H	H	H	H	H
4	L	H	L	L	H	H	H	H	L	H	H	H	H	H
5	L	H	L	H	H	H	H	H	H	L	H	H	H	H
6	L	H	H	L	H	H	H	H	H	H	L	H	H	H
7	L	H	H	H	H	H	H	H	H	H	H	L	H	H
8	H	L	L	L	H	H	H	H	H	H	H	L	H	
9	H	L	L	H	H	H	H	H	H	H	H	H	H	L

图 3

开关 $S_{1n+m}=S_{4\times2+3}=S_{11}$

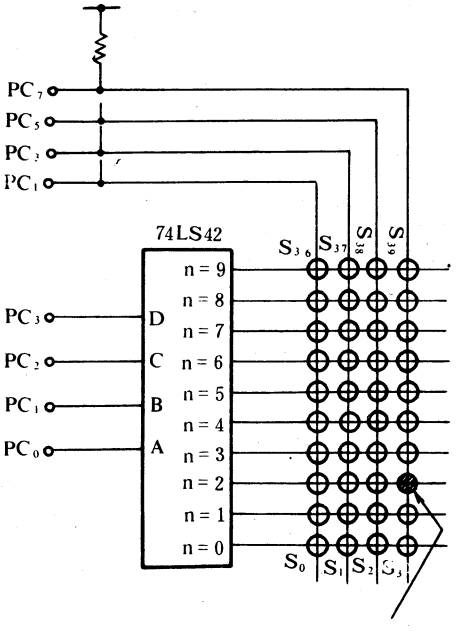


图 4　40 支开关输入电路

3. 与微电脑教育部件的连接

位数字显示部件显示所按动的开关序号 N_0。

图 6 给出扫描 40 支开关输入的流程图(其中×部分与图 2 联接),可以看出 74LS42 10 位输出中仅有 1 位轮流为低电平,端口 C 低位输出数据,同时向端口 C 高位输入开关状态,判断哪支开关被按动,由端口 B 输出开关编号数据。照图 7 所示的流程图则能编制出这种程序。将程序写入微电脑教育部件,如果没有差错,则能较好地执行动作。但此程序编出后很冗长。

如何考虑出比较短的程序呢? 方法如图 7 所示的流程图。以图 4 的电路图为基础,例如,假设 S0 为

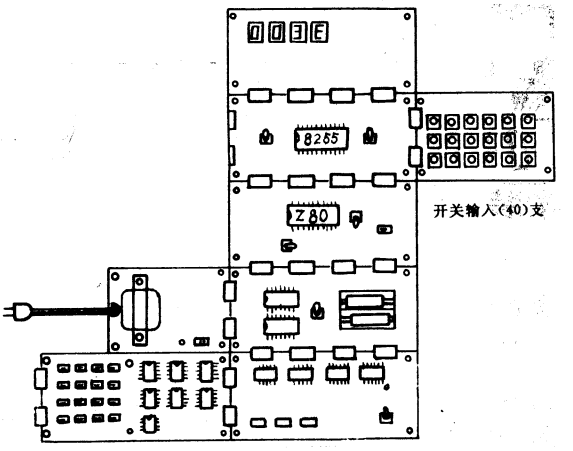


图 5

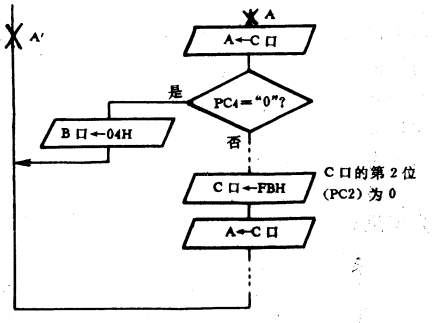


图 6

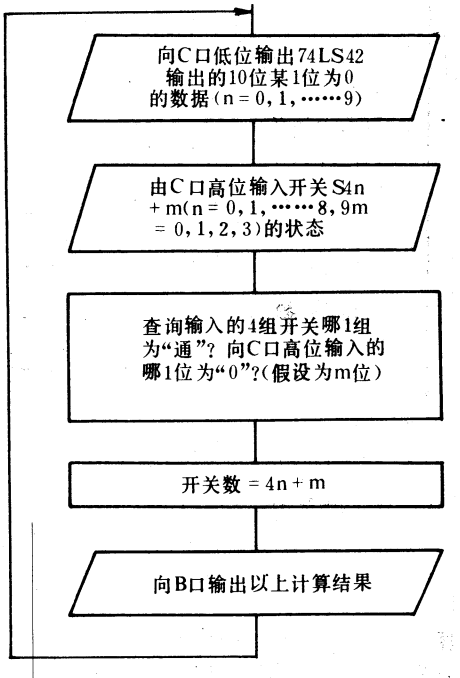


图 7

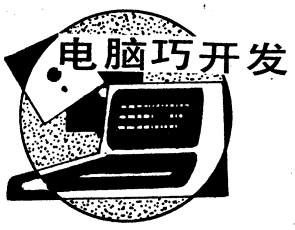
助记符		地址	机器语言	注释
STARY	LD A,98H	00	3E 98	设置 CW
	OUT(03H),A	02	D3 03	
	LD SP,0100H	04	31 00 01	设置 SP
	LD A,FFH	07	3E FF	显示数据初始化
	OUT(01H),A	09	D3 01	
	LD D,00H	0B	1600	初始设定
J1	LD A,D	0D	7A	向 74LS42 输出数据
	OUT(02H),A	0E	D3 02	
	IN A,(02H)	10	DB 02	输入开关信息
	RRC A	12	CB 0F	高位与低位交换、高位为 0
	RRC A	14	CB 0F	
	RRC A	16	CB 0F	
	RRC A	18	CB 0F	
	AND OFH	1A	E6 0F	
	CALL CHECK	1C	CD 40 00	判断所按押开关号码与输出显示
	LD A,D	1F	7A	D=9?
	CP 9	20	FE 09	
	JP Z,J2	22	CA 29 00	
	INC D	25	14	向 74LS42 输出的数据+1
	JP J1	26	C3 0D 00	
J2	LD D,00H	29	16 00	
	JP J1	2B	C3 0D 00	
CHECK	LD B, FFH	40	06 FF	设置累加器 A 中的第 0 位数据初始值
	LD C,04H	42	0E 04	设置检验位的数据
J3	RRC A	44	CB 0F	右转
	JP NC,J4	46	D2 51 00	如果为 0 跳向 J4
	INC B	49	04	B←B+1
	DEC C	4A	0D	C←C+1
	JP NZ,J3	4B	C2 44 00	检验位全部结束跳向 J3
	LD A,FFH	4E	3E FF	
	RET	50	C9	
J4	INC B	51	04	B←B+1
	LD A,D	52	7A	A←D×4+B
	SLA A	53	CB 27	
	SLA A	55	CB 27	
	ADD A,B	57	80	
	OUT (01H),A	58	D3 01	输出开关号码
	RET	5A	C9	

图 9　程序清单

"通"时,S1 为"通"时,进行分析很容易理解。详细的流程图如图 8 所示,程序清单如图 9 所示。与图 6 的流程图相比较,相当短而简炼(但是,对于不熟练的读者难于理解),不能很快理解的读者,首先要进行模仿,在模仿之中逐步搞懂。

本文,介绍了仅用端口 C 输入 40 支开关状态的动态开关扫描方法。由所介绍可知,如果具备 74LS42 IC 知识(硬件知识)与动态开关扫描的程序知识(软件知识),微电脑的应用能向深度、广度扩展,所以说硬件(尤其是 IC 的使用方法)与程序的研究是极其重要的。

(本文图 8 转第 33 页)



MP-I 汉字及图形显示装置

铁道部第四勘测设计院　陈名则

MP-I 微电脑具有价格低，易于开发，应用范围广的特点，本人开发的 MP-I 汉字及图形显示装置，可以在用 LED 组成的屏幕上显示汉字及各种图形，汉字及各种图形可通过程序控制和变化，适合用于作广告及招牌，有一定的实用价值。本装置扩展了原机的接口，并对原机的地址译码电路作了适当改动，现分别介绍于后。

1. 接口扩展电路

MP-I 机原来的输出接口只能显示 8 位二进制数的状态，即 8 个点，而要显示一个字符，一般需 7×9 个点。本装置对原机的输出接口进行了扩展，共可显示 32×16 个点，每屏可显示 4 个 7×9 点阵的汉字。根据本装置的原理，还可扩展到 64×16 点阵或 32×32 点阵。接口扩展电路见图 1。

扩展电路采用 1/2 块 2 线—4 线译码器 T3139（74LS139）及 4 块 4 线—16 线译码器 T1154（74LS154）组合成 6 线—64 线译码器，将 T3139 的片选端①脚接 IC19 的⑥脚，并将 IC19⑥脚与原机 IC21（74LS374）的⑪脚断开，使在选中输出接口页面地址时，扩展的接口电路总是处于工作态。64 块 74LS374 组成 64 个 8 位数据锁存器，每块 374 连接 8 个 LED。其工作原理是，当程序选中输出接口时，T3139 的①脚 $\overline{S}$ 处于低电位，6 线—64 线译码电路处于工作态，根据地址线 A_5～A_0 提供的地址码中的高两位 A_5A_4，先由 T3139 译码。选中 T1154 中的一块，再根据低四位 A_3～A_0 选中 T1154 的 16 条输出线的一条，由于 T1154 选中的输出线为低电平，这个信号正好作为相应的 74LS374 的使能信号，进入 74LS374 的 CLK 端，使数据线的信息进入相应的 374 锁存。由于 74LS374 的 $\overline{OE}$ 端接地，其内部的三态门总是处于工作态，锁存的信息通过 LED 显示出来，当 CLK 端变为高电平后，这一组发光二极管显示的内容保持不变。这样，通过程序控制机器发出接口页面不同的输出地址码和按一定要求编制的数据码，就达到了在 32×16 的点阵上显示汉字和图形了。

2. 原机译码电路的改动

原机译码电路改动的电路见图 2。具体做法是，将 IC18 的⑤脚与 A_8 断开，改接至 A_9；IC18 的②、③脚与 A_9 断开，改接至 6502 的 A_10 端，再利用原机计数器 IC11 的空闲部分，在 IC11（74LS93）的⑧脚接一个三态门，其控制端接 IC4 的①、⑲脚，输出端与 A_{10} 相接，并在

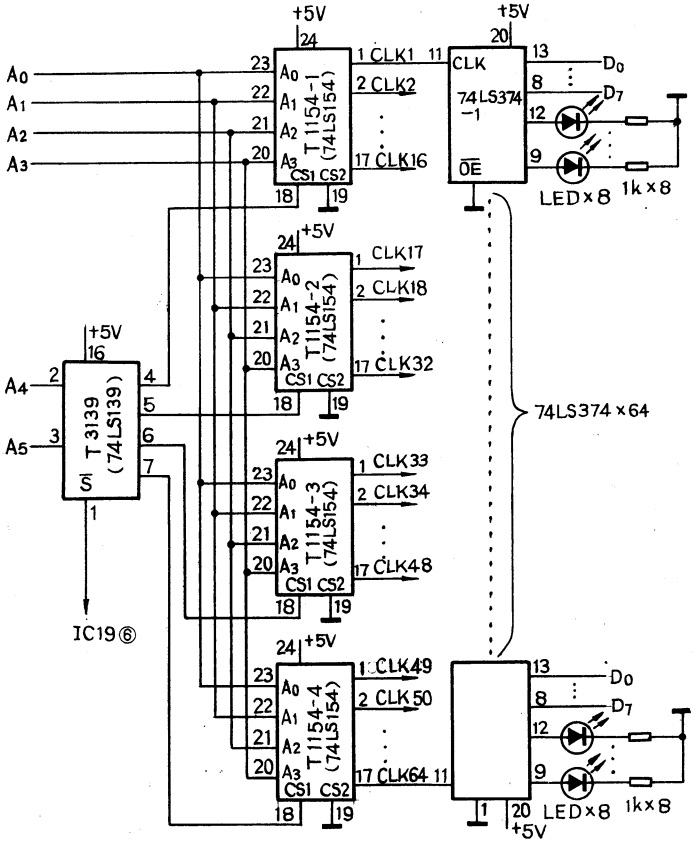


图 1

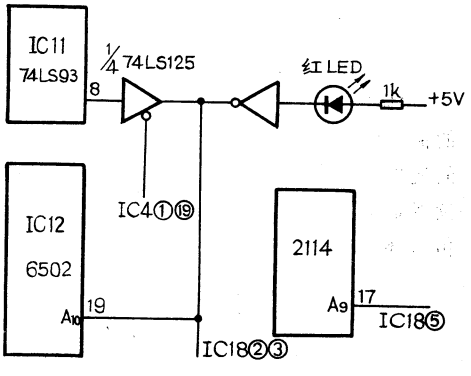


图 2

A_{10}端按原机的显示方法加一个红色发光二极管，用以显示 A_{10}位的状态。改动后，输出端口的地址为 $401～$4FF 的奇数地址，每一个地址对应于一组 8 个数据显示 LED。这样，不但扩大了显示的范围，同时也将整个第 2 页面的地址让给了 RAM2114，增加了存储器的存储范围。

3. 使用方法

将发光二级管 8 个一组，每行排 4 组，共排 16 行，则组成了 32×16 的点阵。将要显示的字符和图案在 32×16 的方格纸上描好，编出相应的 8 位二进制码，按一定的顺序编入程序，输入机器，运行后，即可显示出相应的字符和图案。图 3 是显示“中”字的图形及其相应的编码和汇编程序。

要注意的是，由于原机所配稳压电源容量小，因此，显示器及接口扩展电路要另设单独的电源并将扩展电路的地端与原机的地端连接。

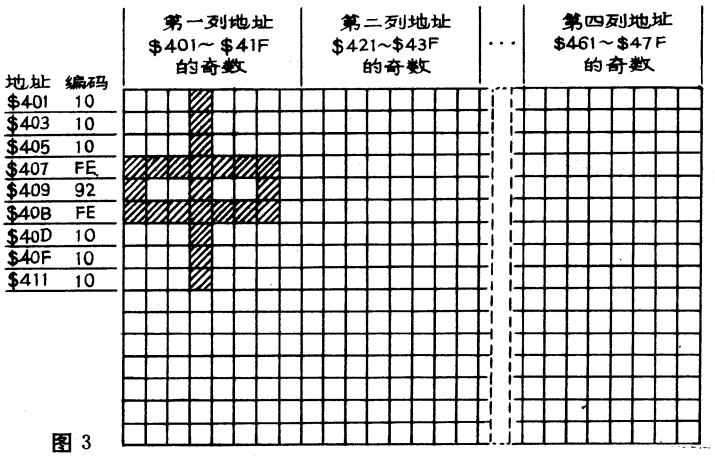


图 3

程序：

```
LDA  #$10
     STA  $401
     LDA  #$10
     STA  $403
     LDA  #$10
     STA  $405
     LDA  #$FE
     STA  $407
     LDA  #$92
     STA  $409
     LDA  #$FE
     STA  $40B
     LDA  #$10
     STA  $40D;
     LDA  $1F;这三个地址传送的数据相
同。
     STA  $411;
```

APPLE— Ⅱ 彩灯自控电路

重庆南开中学　张孝玖

由图 1 所示的 APPLE— Ⅱ 微机的游戏接口插座和图 2 所示的接口电路，在微机程序的控制下，可组成一个具有四个自动开关的控制电路。用三十二个彩色小灯泡分成四组，连接成如图 3 所示的电路，可用它来模拟晚会舞台彩灯或节日彩灯。将图 3 中的 1,2,3,4 端和图 2 中的 1,2,3,4 端分别连接在一起，并将图 2 中的 K 端和图 3 中的 A 端分别接在灯泡电路供电源的两极上。在运行程序时，小彩灯会产生多种形式的变化灯光。

一、APPLE— Ⅱ 微机游戏接口插座简介：

在微机的主机底板上，有一个十六脚未插集成电路块的插座，其旁标有 GAMEI/O 字样，这个插座即游戏接口插座。这里只介绍与本文有关的几个引脚，见图 1。8 脚是接地端，电源负极。1 脚＋5 伏，为电源正极。15、14、13、12 脚分别叫 AN_0，AN_1，AN_2，AN_3，是逻辑电平输出端，处于高电平时约为＋5 伏，低电平时约为 0 伏。这四个输出端电平的高、低由 POKE 语句控制。其控制地址见下表。

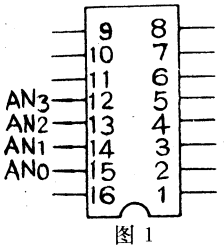


图 1

访问地址与逻辑电平状态之间的关系

输出端	信号状态	控制地址	
		十进制	十六进制
AN_0	低电平	49240　－16296	$CO58
	高电平	49241　－16295	$CO59
AN_1	低电平	49242　－16294	$CO5A
	高电平	49243　－16293	$CO5B
AN_2	低电平	49244　－16292	$CO5C
	高电平	49245　－16291	$CO5D
AN_3	低电平	49246　－16290	$CO5E
	高电平	49247　－16289	$CO5F

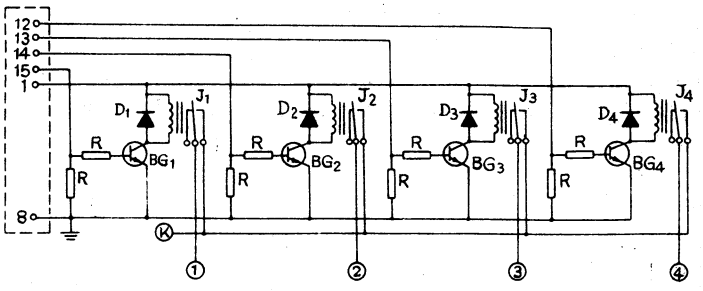


图 2

二、接口电路：

接口电路如图 2 所示，是由三极管 BG_1—BG_4 组成的四个开关电路。J_1—J_4 是微小型继电器。三极管集电极电源由 GAMEI/O 插座的 5 脚供给。当 AN_0 为高电平时，BG_1 的 b—e 极间加上一个正脉冲触发信号而导通，J_1 的簧片吸合；反之，AN_0 为低电平时，b—e 极间电压为 0 而截止，J_1 的簧片释放。其余三个开关电路的工作状态与此相同。这样，J_1—J_4 的公共接点 K 与 1、2、3、4 端就组成了可由程序控制的具有四个开关特点的自动开关电路。每个三极管处于开或关状态的时间长短是与 AN_0—AN_3 相应输出端高、低电平的时间相同的。对 APPLE— Ⅱ 微机，每空循环 750 次，即 FOR T=1 TO 750 : NEXT T 约 1 秒钟。

电路中 BG_1—BG_4 用 3DK 型三极管，D_1—D_4 为普通二极管，所有的电阻值为 5.6K，J_1—J_4 用约小于 +5 伏就能吸合的继电器。电路安装完毕后，一般无需调整，必要时，可调整接在三极管基极上的电阻，以使三极管处于良好的导通和截止状态。

三、彩色小灯泡电路：

彩灯电路如图 3 所示

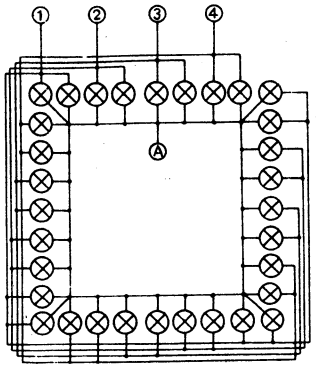


图 3

四、彩灯控制程序举例：

控制彩灯发光形式可依自己喜好设计，我编制的 BASIC 语言程序清单如下。运行该程序时，先是让所有彩灯全亮全灭地闪烁几下，接着是四组灯光匀速跑动，接着是匀加速跑动，接着是匀减速跑动。并且，上述每种情况都是先按顺时针后按逆时针方向进行的。最后，全部彩灯再闪烁几下便结束。

图 2 可作为一个简易通用控制电路，并具有制作简单、使用灵活方便、实用性强的优点。不仅可直接用于负载功率较小的被控电路，也可作为大功率被控电路的驱动电路。所以，它在多种控制设备中有较大用途。

```
]LIST
10 A=49240 : B=49246 : FOR K=A
     TO B STEP 2:POKE K,O : NEXT

20 N=8 : GOSUB 110
30 FOR TT=1 TO 1500 : NEXT
40 P=1 : T1=100 : GOSUB 170
50 FOR TT=1 TO 1500 : NEXT
60 P=2 : T1=10 : T=4 * T1 * N : T
     A=T : GOSUB 170
70 FOR TT=1 TO 1500 : NEXT
80 P=3 : T=0 : T1=40 : GOSUB 170
90 GOSUB 110
100 END
110 REM SUB -1
120 FOR I=1 TO 10
130 IF 1/2<> INT (I/2) THEN
     A=49241 : B=49247 : FOR J=
     A TO B STEP 2 : ROKE J,0 : NEXT
     J : FOR TT =1 TO 300 : NEXT :
     GOTO 150
140 A=A - 1:B=B - 1 : FOR J=A TO B STEP
    2 : POKE J,0 : NEXT J : FOR TT=1 TO
    100:NEXT
150 NEXT I
160 RETURN
170 REM SUB-2
180 A=49241 : B=49247 : C=2 : GOSUB 250
190 FOR TT=1 TO 750 : NEXT
200 LET A=49247 : B=49241 : C=-2
210 IF P=2 AND C=-2 THEN T=TA: GOSUB
    250 : GOTO 240
220 IF C=-2 THEN T=0
230 GOSUB 250
240 RETURN
250 REM SUB-3
260 FOR I=1 TO N
270 FOR J=A TO B STEP C
280 IF P=1 THEN T=T1 : GOTO 3 10
290 IF P=2 THEN T=T - T1:GOTO 310
300 T=T+T1
310 POKE J,0 : FOR TT=1 TO 300 : NEXT
320 POKE J-1,0 : FOR TT=1   TO T : NEXT
330 NEXT J
340 NEXT I
350 RETURN
```

维修经验谈

王安机 RAM 故障简易检修法

董纯坚

WANG－PC 机中存储器 RAM 是微机芯片中最易损坏的集成块之一，维修起来即费时又费钱，但用户只要根据本文介绍的方法，不需要任何仪器设备和电路图，用随机诊断盘就能自行修理，这是一种经济、实用、快速的检修方法。

下面介绍一下 WANG－PC 机 RAM 芯片的分布情况，在主机箱内有 RAM 芯片的板，共有 3 块，它们是系统板，扩展内存板，显示板。

1. 系统板

系统板上有 256KB RAM，它分为四组（BANK），RAM 芯片在系统板上的位置标号及代码见表 1：

下面举具体维修实例对表 1 加以说明。

例 1：故障现象

主机运机一段程序后“死机”屏幕显示：

? SYSTEM ERROR：I/O ERRORON OPTI ON BOARD.

诊断：插入随机诊断盘进入软盘驱动器，选择系统卡项，诊断屏幕显示：DATA READ＝0010.

表 1

RAM 标号	代码	RAM 标号	代码	RAM 标号	代码	RAM 标号	代码
L6	8000	L5	0080	L4		L3	
L12	4000	L11	0040	L10		L9	
L16	2000	L15	0020	L14		L13	
L22	1000	L21	0010	L20		L19	
L28	0800	L27	0008	L26		L25	
L34	0400	L33	0004	L32		L31	
L38	0200	L37	0002	L36		L35	
L44	0100	L43	0001	L42		L41	
L50	0000	L49	0000	L48		L47	

根据读数据为 0010 从上表对应关系可确定是 L21 芯片损坏，换该芯片主机恢复正常运行。系统板另外两排 RAM 如有损坏，则会引起主机不能正常启动，这两排 RAM 故障的排除可采用“芯片叠背法”。

2. 512K 扩展内存板

扩展内存板共有 8 排 RAM 芯片，每排有四个 4164 动态存储芯片，包括 8 个数据位和一个奇偶校验位，RAM 芯片的排布位置，段地址和代码形式见表 2：

例 2：故障现象

主机运行一会，停机屏幕显示内存 RAM 出错。

诊断：插入诊断盘，选择扩展内存卡项，屏幕显示：

OPENED OR SHORTED ADDRESS LINE-（LOC A000：0012H）

PARITY ERROR-（LOC A000：0008H. GOOD/BAD DATA FB/FFH）

表 2

RAM 位置																段地址
106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	E000：×××× 8000：××××
86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	C000：×××× C000：××××
66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	A000：×××× A000：××××

续表 2

46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	80000：×××× 8000：××××
02	04	08	10	20	40	80	80/ /80	80/ /80	01	02	04	08	10	20	40	80	
FD	FB	F7	EF	DF	BF	7F	FF/ /FF	FF/ /FF	FE	FD	FB	F7	EF	DF	BF	7F	
奇					校验位				偶								

根据屏幕显示 ADDRESS 和 RAM 的段地址都是(A000：××××)，可从表 2 中确定损坏的 RAM 芯片在 L65—L82 号这一排。根据代码(FB/FFH)可推算是 L67 号或是 L77 号，具体是哪一位要看(LOC A000：0008H GOOD/BAD DATA FB/FFH)中的 0008H 最后一位 16 进制数是偶数还是奇数来确定，这里 0008 是偶数，那么肯定无疑将是 L77 号芯片损坏，换去该芯片，主机运行正常。需注意的一点是：有的内存卡诊断时，将显示数字代码，芯片的寻找方法同上。

3. 图形板

图形卡 RAM 是 4116 芯片，每个芯片是 16K 位，有 2 排，每排 8 个芯片，各芯片的位置及代码见表 3：

例 3：故障现象

开机屏幕上有许多雪花点或竖杠杠，影响操作人员视线诊断：插入随机诊断软盘进入软盘驱动器。

屏幕显示：DATA DETACTED DATA＝2000，对应于 L19 号芯片换之，屏幕恢复正常。

以上王安机 RAM 芯片的诊断及维修是笔者在维修中的探索和经验，供各位读者参考。

表 3

RAM 位置	28	29	30	31	32	33	34	35
代码	0001	0002	0004	0008	0010	0020	0040	0080
RAM 位置	14	15	16	17	18	19	20	21
代码	0100	0200	0400	0800	1000	2000	4000	8000

IBM 286 维修一例

叶志斌

故障现象：机器运行 dBASE Ⅲ程序过程中突然死机，热启动启失效，冷启动后显示 162——System Options Not Set—(RUN Setup)

(Resume＝"F1"key)

按 F1 键后 C 盘不能自举，进入 ROM BASIC，但软盘仍可自举，再运行 Setup 现象依旧。

故障分析：CPU 是 80286 的各类微机一般采用"Setup"软件以菜单的形式来选择系统配置的，运行它则把所选的有关配置信息通过系统板上的 RT/CMOS RAM 端口地址写入这一 RAM 中，一旦 RT/CMOS RAM 电路或 RAM 片子出现故障，Setup 的信息就读不出来，或已丢失。再次运行 Setup 也写不进去信息，而这类错误最常见的原因是给 RT/CMOS RAM 供电的干电池电压不足所至，遇到这类问题应先测量干电池的电压，若不足则更换之，当你一时买不到，可拉出两根引线，在机箱外用两节高效 5 号电池取代，看能否排除故障。

故障排除：本例是电池电压不足，更换新电池后，运行 Setup，故障排除。

中华学习机典型故障一例的分析

杨瑞华

故障原因：开机无“哗”声。屏幕闪一下，然后什么也没有，主机指示灯亮，同时按其它的键无任何反应。

分析：根据上述反应，可能引起故障的原因有：视(射)频、系统时钟影响整机工作。落实到具体器件是Q_1、Q_2、B_2等元件组成的主频振荡电路，有关的PAL电路及其它主要部件。如CPU、MMU，IOU，U_{13}，U_{16}，U_{17}等有关电路。

打开机盖，取下键盘，主机板上的电路及元件一清二楚。其功能分布很明了。检修工具用的是MODEL－101万用表。首先对CPU的控制总线进行直流电压的测量，(6502地址总线和数据总线，在读写时其引脚的电压不一至，它所提供的电压不能判断电路本身是否正常，所以可以不去测量地址和数据总线)，测6502CPU，主要测1脚，3脚，8脚，37脚，39脚，21脚等，首先测量1脚V_{SS}，8脚V_U，21脚V_{SS}均正常。进一步测量与故障有关的脚，3脚，37脚，39脚。测得3脚$\overline{OUT\ Q_1}$为零伏，实际应为2.4伏左右。39脚OUT Q_2为4.6伏，正常值在2.2伏左右，3脚与39脚均是6502微处理机对输入的定时信号Q。经过处理后的输出定时信号。测37脚$\overline{INQ}$。为4.6伏左右，正常值为2伏左右，用交流档测没有交流信号，Q_0是6502微处理机的基本脉冲定时信号接入单，没有交流信号证明没有外来脉冲信号，故障可以比较肯定的确认为是PAL或者是晶体振荡部份。而6502的时钟信号由PAL电路中的U_{17}、晶体振荡部分由Q_1，Q_2，B_2等构成。

由于PAL电路是一块可编程逻辑部件，用来产生脉冲信号和控制信号，它是一个20脚封装的引线，查手册得知20脚为V_{DD}，10脚为V_{SS}。测得两脚电压正常，继续测1、2、3脚均正常。测14、15脚时发现14脚为零伏，而15脚又有4.6伏，查得正常时均为2.1伏左右。用万用表串电容测交流电压没有变化，这时将U_{17}拔下放在工作正常的机器上试验，开机正常，证明芯片没有问题，分析只有14脚没有脉冲信号。引起这个交流信号的可能在Q_1、Q_2，B_2晶振脉冲上发生问题。重点查找这三个元件及有关元件。用表测Q_1，和Q_2发现Q_2的三个脚电压一样高。断开电源用欧姆档测，发现E脚与其它两脚不通，用小功率烙铁加吸锡器快速焊下，果然三极管已开路。相同换上同型号β的三极管，再开机一切正常。这个问题可能是管子本身的质量差引起的。

APPLE Ⅱ 主机常见故障的检修

陈 鸿

APPLE Ⅱ机在我国中小学中已相当普及，由于主机板的工作条件较恶劣，承受着浪涌冲击，温升高及操作不当等，使得集成块损坏率较高。笔者根据主机板的工作原理以及积累的实际维修经验，将主机板的常见故障整理归纳如下几条，供大家维修时参考。

APPLE Ⅱ机主机板主要由6502中央微处理器ROM(只读存贮器)、RAM(随机存贮器)，输入输出装置及产生视频文字及图像信号的线路等几部分组成。根据屏幕出现的故障现象，就能判断出故障的位置，从而把故障排除掉。

一、6502中央微处理器故障：

现象：开机后，指示灯亮，喇叭不响，无光标，屏幕无任何反应，不能复位，各种冷、热启动均无效。

二、RAM故障：

现象：1.开机后，指示灯亮，喇叭不响，屏幕出现杂乱文字或黑白小方块，按键无效。此类故障多数为C_3－C_{10}(地址为$0000－$3FFF)的RAM。

2.驱动器工作不正常。

3.程序运行一会就停止。

4.高解像页区图像有斑点不能消除。

5.驱动器与接口卡完好，插入主机后，主机不能工作。

6.开机后，直接进入监控状态，屏幕出现“*”的系统光标。

2－5条多数为D_3－D_{10}或E_3－E_{10}(高地址$4000－$7FFF，$8000－$BFFF)RAM故障。

三、ROM故障：

现象：1.开机后，没有任何反应，屏幕全白或全为“?”，按键无效。一般为第6个ROM，用2732的则为第3个ROM。

2.开机后，直接进入监控状态。

3.开机后，直接进入监控状态，按复位键后屏幕出现APPLESOFT的状态，但任何命令仍不能执行。

4.开机后，执行程序时会出现错误信息。

以上2－4条多为第1－3个或第5个ROM。

(转第19页)

计算机基础知识(下)

王路敬

7. 目前编写计算机程序所用的语言可分哪几类？各有何特点？

可分三类：

(1)直接和机器打交道，用计算机的指令表达的机器语言。这种语言是计算机硬件系统所能识别的、不需要翻译直接供机器使用的程序语言，机器型号不同，机器语言通常不同。机器语言中的每一条指令是一条二进制形式的指令代码，该代码由操作码和地址码组成。机器语言程序编写和调试修改都比较麻烦，但执行速度快。

(2)用机器指令的助记符表达的汇编语言。这种语言是一种面向机器的程序设计语言，不同系列的计算机其汇编语言不同。例如苹果机的 6502 汇编语言，IBM PC 系列机及其兼容机的 8088 汇编语言等。汇编语言是用助记符来代替操作码，用地址符号来代替地址码的语言。所以汇编语言也叫“符号”语言。汇编语言与机器语言关系密切，它的指令和翻译成的机器语言之间的关系基本上是一一对应的。这种语言好理解，好记忆，便于阅读，而且保持了机器语言编程质量高、执行速度快，占用内存空间小的优点。通常汇编语句由标号、操作码、操作数和注释四部分组成。使用标号可方便查询和修改，便于转移指令的书写；使用注释可增强程序的可读性。操作码和操作数一起构成一个机器指令，指定计算机完成一个特定功能。操作码规定机器执行什么操作，操作数提供执行操作时的数据或数据所在的地址。为了得到操作数需要了解寻址方式，寻址范围和使用方法。不同的指令系统采用的寻址方式不同，常用的有立即寻址、直接寻址、间接寻址、变址寻址等。汇编语言在系统开发、实时检测、实时控制、实时处理中发挥着巨大的作用。

(3)独立于机器，用不依赖于机器的具体指令表达的高级算法语言。例如 BASIC，PASCAL、FORTRAN 等。这种语言无需了解计算机的内部构造。使用高级语言编写程序方便，易于查错、验证、阅读和修改；同时由于高级语言编写的程序符合人们的习惯，能自然地表达各种问题的有关概念，所以可大大提高程序的可移植性和通用性。

目前世界上已有数百种高级语言，用的最多的也有数十种，其中 BASIC 语言是微型机上使用最普遍的一种高级语言。

使用高级语言要注意掌握语法和程序设计方法。一般高级语言的语法成分主要有基本元素(数、变量、串等)，表达式和语句。从功能上可分输入输出语句，逻辑运算和算术运算语句、程序控制结构语句和传输语句四类。只要掌握了基本语句的功能和使用，再熟悉一些常用算法，就可以编写一些实用程序。

除机器语言程序可以直接为机器所识别外，汇编语言和高级语言编写的程序，我们称之为源程序它必须经过“翻译”变成机器语言才能被机器识别。汇编程序把汇编语言源程序“翻译”成机器语言程序，该过程叫汇编。高级语言源程序“翻译”成机器语言程序有两种方式：一种是编译方式。这种方式是编译程序把高级语言源程序，经过语言编译器，统一地进行一次翻译工作，同时也进行查错，然后产生出一个目标码(机器码)组成的目标程序，在这个过程中计算机不执行任何源程序内所规定的动作。使用编译程序对源程序进行编译时，有的系统中形成的目标程序可直接执行，但在 IBM PC 系统中，目标程序不能直接运行，还要将生成的以 · OBJ 为扩展名的目标程序经过连接程序生成以 · EXE 为扩展名的可执行文件后才能执行。第二种是解释方式。这种方式直接将源程序引入内存，然后语言系统本身逐行地读取它，通过解释器对其进行解释和查错。如果没有语法错误，便根据解释结果，把该行语句翻译成相应目标程序，让机器去执行。

两种“翻译”方式各有优缺点。解释方式的缺点是执行速度慢，优点是调试程序很方便。因为在解释方式下，一般都是单步执行，产生错误后便立即停止执行，显示出错误信息。编译方式的优点是产生的独立的可执行文件运行速快，保密性好。源程序经过编译、连接后得到的 · EXE 文件无须加密，任何解密程序也不能破密而得到源程序。缺点是调试比较困难。

8. 在计算机内部采用什么数制？这种数制有何特点？

在计算机内部一切信息包括数字、字母、字符和汉字等的存放，处理和传送均采用二进制数的代码组合，这是认识计算机的根本出发点。数在计算机中是以器件的物理状态来表示的，一个具有两种不同的稳定状态且能相互转换的器件，就可以用来表示一位二进制数。所以在计算机内用二进制数表示简单可靠，二进制运算规则也是最简单的。因此，目前在计算机中广泛采用二进制数来表示数字和进行运算。

二进制数具有两个基本特点：(1)具有两个不同的数字符号即 0 和 1；(2)逢二进位。由于是逢二进位，所以同一个数字符号在不同的数位所表示的值是不同的。

例如 1111.111

小数点左边的第 1 位的"1"代表的值就是本身，小数点左边的第 2 位的"1"是由第 1 位逢二进上来的，所以它的值为 1×2^1；则左边的第三位、第四位的值依次应当是 1×2^2 和 1×2^3；而小数点右边第 1 位的 1 则代表 1×2^{-1}；右边第 2 位、第 3 位的值应当是 1×2^{-2}和 1×2^{-3}……。

应当指出的虽然在计算机内部使用二进制数进行工作，但是，对于人来说，使用二进制是很不方便的。二进制数比起等值的十进制数来，位数要多得多，写起来，读写来也不方便。为此，人们通常用八进制和十六进制做为二进制的缩写方式。十进制数是人们通常习惯采用的计数制，也是人们最熟悉的计数制，要想用计算机处理十进制数必须要用二进制为十进制数编码并按二进制逻辑来实现十进制运算法则。同理，计算结果应从二进制数转换成十进制数再从计算机输出。这就产生了不同计算制之间相互转换的问题。

9. 在计算机中区别数的正负是如何规定的？

区别数的正负用符号位，符号位规定在数的最前面。用"1"表示负数；用"0"表示正数。也就是说，数的符号在机器中也数码化了。我们把一个数在机器中的表示形式叫机器数。而把这个数的本身叫真值。

例如：设有

N1＝＋1100111

N2＝－1100111

则 N1 和 N2 在计算机中表示为：

N1：01100111

N2：11100111

带有符号的 N1，N2 为真值；在机器中表示的 N1，N2 为机器数。由于计算机硬件设备限制，机器数有位数限制。通常机器数位为 2 的倍数。

10. 在计算机中数有哪几种表示方法？

有三种表示法：原码、补码和反码。正数任何时候都用原码表示。只是负数可以用原码、补码或反码表示。

原码表示法是一种简单的机器数表示法。其符号位用数码 0 表示正号；用数码 1 表示负号。数值部分按一般二进制形式表示。

例如：

N1＝＋1001010

N2＝－1001010

则

〔N1〕原＝01001010

〔N2〕原＝11001010

补码表示法规定：正数的补码和原码相同。负数的补码则先对原码除符号位外各位取反，然后末位加 1。

例如：

NN＝－1001010

则

〔NN〕原＝11001010

〔NN〕补＝10110101＋1＝10110110

引进补码概念之后，加减法运算都可以用加法来实现，即用求"和"来代替求"差"，数的符号位也可以当做数值处理，一道参加运算，且真值的补码之"和"等于真值"和"的补码。这为加减法运算带来很多方便，因此，在近代计算机中加减法多采用补码运算。

反码表示法规定：正数的反码，或者负数的反码都是这个正数的原码或者这个负数的原码除符号位外按位求反得到。

倒如：

N1＝＋1001010

N2＝－1001010

则

〔N1〕原＝01001010

〔N1〕反＝00110101

〔N2〕原＝11001010

〔N2〕反＝10110101

反码通常做为求补过程的中间形式。

但是不论用哪一种方法表示数，当数的绝对值超过机器允许表示的最大值时，就要发生溢出，从而造成运算错误。例如 8 位机，即字长为 8 位，则最大值为 $(255)_{10}$；若超出此值，就会产生溢出。16 位机即字长为 16 位，则最大值为 $2^{15}-1$；如果运算结果大于此值，就发生溢出。这个溢出刚好进到符号位，占据了符号位的位置，从而使结算结果发生错误。使用中应注意这个问题。

11. 十进制数与二进制数之间相互转换时应注意什么问题？

不同计数制之间的转换是根据如两个有理数相算，则两数的整数部分和分数部分一定分别相等的原理进行的。

十进制整数转换成二进制整数采用除 2 取余法；十进制纯小数转换成二进制小数采用乘 2 取整法；十进制混合小数转换为二进制数则将整数部分和纯小数部分按上述原则分别进行转换，然后再将其组合起来即可。二进制数转换为十进制数方法更为简单，只要将二进制数用计数制通用形式表示出来，相加计算出结果，便得到相应的十进制数。

不论是十进制数转换二进制数还是二进制数转换为十进制数，其整数部分和小数部分要采用不同的转换方法。不要误认为一个整数和一个小数形式一样，则转换后形式也一样。例如 10111 是十进制的 23，但 0.10111 却是十进制的 0.71875；19 是二进制数的 10011，但 0.19 却不是二进制数的 0.10011。这是第一点要注意的。

第二点，十进制小数不一定都能转换成完全等值的二进制小数，所以有时要取近似值。

从初等数学中我们知道：任何有限位的小数，都能用分数表示，但是任何一个分数却未必能用有限位的

小数表示，例如 1/3 就是这样。所以两种数制的转换也存在类似情况。一个二进制小数，能够完全准确地转换成十进制小数，但是一个十进制小数，却时常不能完全准确地转换成二进制小数。例如十进制数的 0.1 就是这样。

不能用有限位的二进制小数去表示任一个有限位的十进制小数，这是二进制的一个缺点。但对一般科学计算，这个缺点是可以容忍的。因为科学计算或多或少都具有近似计算的性质。彻底解决这个问题，计算机多采用二进制编码的十进制数进行计算。即一位十进制数用四位二进制编码来表示，表示的方法很多，较常用的是 8421BCD 码。

12. 微机使用的西文字符是如何编码的？

微机使用的西文字符常用的编码是 ASCII 码。每个 ASCII 码由 8 个二进制位组成，它是计算机代码的最基本单位，即一个字节。一个字节表示的范围十进制数为 0～255。所以全部 ASCII 码最多可表示 256 种不同的字符，但在 ASCII 码中，并不把 256 种二进制数组合都作为字符的代码，而是在 ASCII 中，把二进制位的最高位为 0 数字称为基本 ASCII 码。它的范围是 0～127。即基本 ASCII 码共 128 个，其中 0～31 作为控制代码，32－127 为显示字符。控制代码在计算机中不当作字符显示，而是作为计算机进行某一特定的动作的功能代码。例如代码 13 的功能是把光标移到行的最左端，代码 10 的功能是使光标移到下一显示行等等。32－127 共 96 个编码是字符码，其中包括英文字母的大小写一共 52 个。供书写程序和描述命令用。

在 ASCII 码中，把二进制位的最高位为 1 的数字称之为扩充 ASCII 码，它的范围 128～255。扩充 ASCII 码也是 128 个。各国都把扩充 ASCII 码规定为自己国家语言的字符代码。我国就把扩充的 ASCII 码作为汉字的代码。一个汉字的机内码就是由两个扩充的 ASCII 组成，这两个内码的选取与汉字区位码的规则相关。

13. 什么是汉字的外码？

所谓汉字的外码是指计算机与人进行交换的字形符号的一种代码。它与汉字的输入方式有直接关系。同一个汉字，选择不同的输入方式，其汉字的外码不同。例如“啊”，在区位码输入方式下其外码为“1601”；拼音码输入方式下，其外码为“a”；而五笔字型输入方法，其外码为“kbsk”。汉字输入的过程就是外码转换为内码的过程，即将汉字或一些符号的外码键入计算机后，计算机把它转换为可识别的内码后再存储内存。而汉字的输出过程则是内码转换为外码的过程，即计算机把内存中的内码转换为约定字型后输出到显示器或打印机。

（接 46 页）

硬件配置：

Turbo CAD 要求至少配置下列硬件与操作系统：

1. IBM PC/XT、AT 及其兼容机

2. PC DOS（或 MS－DOS）操作系统 2.0 以上版本

3. 内存（至少 256K）

4. 360K 软盘驱动器 2 个

5. 图形显示器。

以上是所需的基本配置，而为了提高 Turbo CAD 程序的运行效率，应该增加下列硬设备。

1. 增加硬盘容量：可改善整个操作系统环境，增快文件输入/输出速度，免除抽换软盘的麻烦、能利用 DOS 子目录来配置图形文件的存贮空间。

2. 输入设备：Turbo CAD 的标准输入设备是键盘、同时也可利用其它输入设备、如鼠标器、数位板等。这些可在 Turbo CAD 中设定所需的输入设备。

3. 打印机：Turbo CAD 能把图形文件输出给许多不同的打印机进行打印。通常、打印机的分辨率比屏幕的高，如果所使用的打印机在 Install 程序中并不列出、可配置兼容的打印机。

4. 绘图机：如要得到质量更佳、分辨率更高的图形，最好使用绘图机。Turbo CAD 提供了许多机型的绘图机供选用。

5. 系统存贮器：Turbo CAD 可以完全利用系统内存 640K，如果用户在图形中加上更多的图素，Turbo CAD 就会用到更多的内存；如果绘制的图形很复杂，则占用内存可能超过 256K（在任何时候，Turbo CAD 绘图屏幕左下角的状态区内均会显示出内存的剩余空间）。若系统配置有内存扩充卡、可将其设定为随机存取磁盘、以减少磁盘存取时间，加快绘图速度。

以上是 Turbo CAD 系统运行的软、硬件环境。读者可根据各单位的具体情况，加以选用。

购书消息

《电子与电脑》90 年合订本已出版发行，每本定价 13.5 元（另加邮费 15%）。需要者可汇款至北京 173 信箱，电子工业出版社发行部，邮购科　邮政编码 100036

新书与软件

Turbo CAD 软件简介

Turbo CAD 是美国 Borland International Inc. 推出的能在 IBM PC/XT 及其兼容机上实现的一种崭新的计算机辅助设计系统。它是以主菜单(主功能表)和下拉式菜单(下拉式功能表)的功能选择项提供各种绘图命令,它的绘图功能完整,开发环境良好,使用方便,执行速度快,拥有 128 层,100 种线型,255 的线宽以及 32 种箭头和屏幕计算器供用户绘图与计算使用,其绘图精度很高,能任意设定绘图比例,能任意填图,能自动标示尺寸,能测量、画图与计算,且能自动计算面积,长度,周长等。能注意标示所需字型大小(5 种)、颜色(16 种)的文字;能对图形进行任意放大、缩小、旋转,映射与拉长;能对图形进行任意删除、增添、移动与拷贝;能将图形输出至绘图机或打印机进行绘制或打印。

Turbo CAD 可用于绘制各式各样的图形,适用于诸如电子、电气、电器、土木、建筑、机械、造船、航空、化工、纺织、美术、服装设计等各行各业的计算机辅助设计,应用范围十分广泛。

Turbo CAD 的软件配置与硬件要求

软件配置:Turbo CAD 共有系统盘 2 片、其内容包括 Turbo CAD 程序、支援程序和文件。

第 1 盘的内容为:

TURBOCAD COM	368244—17—88	7:57p
TURBOCAD 000	40961—03—88	5:57p
TURBOCAD 001	74243—02—88	
TURBOCAD 002	7683—02—88	
TURBOCAD 003	107523—02—88	
TURBOCAD 004	20483—02—88	
TURBOCAD 005	33283—02—88	
TURBOCAD 006	17923—02—88	
TURBOCAD 007	10243—02—88	
TURBOCAD 008	66563—02—88	
TURBOCAD 009	30723—02—88	
TURBOCAD 010	97283—02—88	
TURBOCAD 011	197123—02—88	
TURBOCAD 012	87043—02—88	
TURBOCAD 013	35843—02—88	
TURBOCAD 014	23043—02—88	
TURBOCAD 015	94723—02—88	
TURBOCAD 016	660483—02—88	
TURBOCAD 017	28163—02—88	
TURBOCAD 018	281603—02—88	
TURBOCAD 019	53763—02—88	
TURBOCAD 020	122883—02—88	
TURBOCAD 021	61443—02—88	
TURBOCAD 022	71683—02—88	
TURBOCAD 023	46083—02—88	
TURBOCAD 024	166403—02—88	
TURBOCAD 025	112643—02—88	
TURBOCAD 026	120323—02—88	
TURBOCAD 027	23043—02—88	
TURBOCAD 028	43523—02—88	
TURBOCAD 029	84484—17—88	7:57p
TURBOCAD 030	30723—02—88	
TURBOCAD 031	135683—02—88	
TURBOCAD 032	25603—02—88	
14X9 FON	35842—04—88	
4X6 FON	2852—04—88	
LINETYPE LIN	502—04—88	
NORMAL ARO	432—04—88	
NORMAL DRV	10081—01—80	12:06a
TC DRV	1701—01—80	12:05a
CONFIG CAD	1561—01—80	12:15a
41 File(s)		0 bytes free

第 2 盘的内容为:

TURBOCAD	OVL	11360	8—29—88	3:47p
TURBOCAD	HLP	20252	3—02—88	
INSTALL	COM	60788	3—02—88	
COMPLEX	FNT	11260	3—02—88	
ITALIC	FNT	12468	3—02—88	
NORMAL	FNT	4428	3—02—88	
SIMPLEX	FNT	6828	3—02—88	
TXT	FNT	4016	3—02—88	
TCUTIL	COM	56636	3—02—88	
MASTER	DRV	17388	9—26—88	1:30p
BILLOFM	PRG	6311	3—02—88	
READ	ME	2176	1—01—80	12:02a
README	BAT	24	1—01—80	4:59a

13 File(s) 141312 bytes free

(转第 45 页)

巧用电脑作幻方

郭高惠

电脑的出现离不了数字，处理数字的能力自然也就无需置疑了。将其投入数字海洋，并利用那些满足要求的数字，为我们拼造出一个个赋有魔力和趣味的数字方阵——幻方，正是本文的目的所在。

所谓幻方，其实是指由 1 到 n^2 个连续自然数排列成的方阵；它有 n 行 n 列，并且每行上 n 个数字之和与每列上 n 个数字之和相等，同时还与两条对角线上的 n 个数字之和相同，这个和可由式 $n(n^2+1)/2$ 确定；n 就称为该幻方的阶（或称 n 阶幻方）。

幻方的构造虽然遵循某种规律，但要靠人工来构作（特别是高阶幻方）并不那么容易办到，所以最好还是由我们手头的电脑来代劳。为此，笔者设计了下面的幻方制作程序可在一般家用电脑上直接运行供大家使用，利用它可以得到任意阶数的幻方。

由于奇阶幻方与偶阶幻方（偶阶中又将阶数能被 4 整除的幻方另归一类，即 4 的倍数阶幻方）的构成规律不同，所以在程序中使用了不同的语句段。下面就对程序略作说明。在该程序中，用变量 I、J 分别作为幻方的行、列坐标值；G 用于区别幻方类型（即奇或偶幻方）。N 是幻方的阶数，程序运行前会提示你从键盘输入。S 数组用来存放幻方数据，其它变量是一些辅助临时性变量。

70 语句定义数组为输出幻方作准备；

90～110 语句用于完成 4 的倍数阶幻方的构造。

150～180 语句用于完成其它情况的幻方设计。

185～190 语句为输出整齐美观的幻方作准备。

195～220 语句用于幻方的输出打印。如果是偶阶幻方，就不能直接输出，要转 300 语句作处理后再输出。300～350 语句可按给定规律对偶阶幻方进行合成并完成输出。

230～240 语句用于判断是否还要继续运行。

```
LIST
10 CLS'     这是一个求取 N 阶幻方的实用程序
40 PRINT:INPUT"请输入阶数 N[N>2]";N
50 PRINT:I=1:P=1:G=N AND 3
55 IF N<3 THEN PRINT"阶数小于 3. 重新输入.":RUN
   40
60 L=N/(1+ABS(G=2)):J=INT(L/2+ .5):IF G=0
   THEN J=1
70 DIM S(L,L):FOR T=1 TO L * L:R=INT(T/L)
80 S(I,J)=T:IF G>0 THEN 150
90 IF T/L=R THEN I=I+1:J=(1+L) * SGN(I AND
   2)
100 P=1-P:IF P=0 THEN I=L-I+1
110 J=J+1-(R+1 AND 2):GOTO 180
150 IF T/L=R THEN I=I+1:GOTO 180
160 I=I-1+ABE(I=1) * L:
170 J=J+1-ABS(J=L) * L
180 NEXT:T=T-1:P=LEN(STR$(T)):K=P * N/2-8
185 K=ABS(N>6) * K+1:FOR I=1 TO P:Y$=Y$+
   CHR$(35)
190 NEXT I:K=K+ABS(N>L) * N/2:PRINT TAB(K);
195 PRINT"下面是";N;"阶幻方":IF N>L THEN 300
200 FOR I=1 TO N:FOR J=1 TO L
210 PRINT USING Y$;S(I,J);:NEXT :PRINT:NEXT
220 PRINT TAB(K);"其特征值为:";(N * N+1)/2 * N
230 PRINT:INPUT"还想打印请敲 Y 键";Y$
240 IF Y$="Y" OR Y$="y" THEN RUN 40 ELSE END
300 R=INT(L/2)+1:FOR S=1 TO N:I=S-ABS(S>L) *
   L
310 FOR P=1 TO N:J=P-ABS(P>L) * L:G=(J>1 OR I
   <>R)
320 G=3 * T * (ABS(S>L)<>G * (J<R)+(I=R) * (J
   =R))
330 IF P>L THEN G=T+ABS((S>L)=1+J<R)) * T
340 PRINT USING Y$+"#";S(I,J)+ABS(G);:
350 NEXT :PRINT:NEXT:G0T0 220
RUN
请输入阶数 N[N>2]? 5
下面是 5 阶幻方
17 24 1 8 15
23 5 7 14 16
4 6 13 20 22
10 12 19 21 3
11 18 25 2 9
其特征值为:65
还想打印请敲 Y 键? Y
请输入阶数 N[N>2]? 4
下面是 4 阶幻方
1 14 15 4
8 11 10 5
12 7 6 9
13 2 3 16
其特征值为:34
还想打印请敲 Y 键? Y
请输入阶数 N[N>2]? 6
下面是 6 阶幻方
35 1 6 26 19 24
3 32 7 21 23 25
31 9 2 22 27 20
8 28 33 17 10 15
30 5 34 12 14 16
4 36 29 13 18 11
其特征值为:111
还想打印请敲 Y 键? N
OK
```

湖北地区中华学习机联谊活动情况

本刊讯：湖北省中华学习机应用沙龙是湖北地区长年坚持中华学习机普及活动的一个群众团体，沙龙的联合发起单位有湖北省青少年科技活动中心，长江日报社，武汉市青少年宫、武昌区青少年计算机协会、湖北省电子器材物资公司电脑开发部，沙龙的具体活动由市青少年宫和武昌青少年计算机协会负责。联合这些发起单位的目的是便于解决沙龙需要的活动阵地、宣传报导、骨干力量（包括活动积极分子和技术力量）的组织、器材经费的筹措等一系列问题。

我们理解的沙龙，就是计算机爱好者的俱乐部，它应该不以赢利为目的，宗旨就是为广大青少年和拥有中华学习机的个人服务，组织他们切磋电脑应用经验，交流学习体会，互相交换各类软件，提供各种服务：组织系列软件讲座，提供资料和信息，帮助解决用户在使用中遇到的各种困难，组织会员与生产厂家对话等一系列活动。我们开展的这些活动博得了广大计算机爱好者的欢迎和支持，沙龙从去年（八九年）六月成立至今我们长年坚持活动。会员也是来自四面八方，有工人、学生、机关干部、教师、工程师，会员里面有十几岁的娃娃，也有退休的老人，有为了增加自己的知识面而来的，也有是为了教孩子先来当学生的，甚至连远在湖北荆门炼油厂的同志也找上门来要参加我们的活动，鄂西自治州的同志来信索取资料。

活动的形式也不是一成不变的，沙龙成立之初，我们采取上大课办讲坐的方式，这种方式对一部分初学者可能很有帮助，但对另外一些人如有些已涉足硬件开发的同志显然是不能满足他们的要求，这时候我们从沙龙活动的积极份子中组织了一批骨干，成立会员常委会，由这些常务会员开会研究确定每月活动的具体内容，每个常务会员都有宣讲义务。为了照顾不同层次的要求，改变过去一次活动一个专题，单打一的方式，每次安排三个内容（兼顾普及和提高），每个专题尽量短小精干，言欲未尽的会下再和有兴趣的一部分同志进一步探讨。这种做法深得广大会员的赞尝。另外沙龙还同社会上其它一些组织如湖北省电子学会联合举办一些讲坐，以提高沙龙的知名度，让更多的计算机爱好者参加到我们沙龙里面来。

沙龙成立至今仅仅一年多，虽然做了一些工作，但也存在不少问题，它毕竟是一种新事物，没有可循的常规，要靠我们不断总结完善，使其更有生命力。

浙江省现代微电子技术发展公司国内最优价推出

计算机类 万国 WG-900 超级多功能办公机集电脑、轻印刷系统、打字机为一身，融文字处理、传真、联网功能为一体，首批办公机在浙江省委宣传部等机关、企事业单位中运行。基本配置 Super PC/XT 加 M1724 打印机。CPU NEC V20 12MHz RAM 640KB 硬盘 20MB 软驱 360K×2，101 键标准键盘，14″双频直角平面显示器，国际流行机箱 11000 元/套。Super PC/XT 5200 元/套 Super PC 3400 元/套 中华机 CEC—I 790 元--950 元/台（陕产）CEC－PC/10 3950 元/套 AST 286（140）18500 元/套 AST 386 SX 21000 元/套 供应各类进口驱动器、打印机、PC、中华机软件。

单片机类 MCS－51 标准型单片机学习开发系统 8031 芯片，时钟频率 6－12MHz RAM 6264 EPROM 27256 EEPROM 2864 两路 16 位定时/计数器；可编程 I/O 为 8255 提供三个并行口，可直接驱动打印机或与 IBM PC 并行口连接；八路 8 位 A/D 0809 一路 8 位 D/A 0832。该系统用读者熟悉的 Z80 单板机模式开发单片机，同时为用户提供了一个完整的数据采集、处理及控制系统。适宜大中专院校、科研、工矿企业学习、开发 MCS－51 系列单片机及作为单片微机函授班教学、实验用机，散件 325 元/套 整机 375 元/套 附有实验指导书一套。

集成电路类 特价供应各种规格进口集成电路 8031 15 元，6264 24 元，2864 80 元，27256 18 元，8255 12 元，0809 18 元，0832 18 元，8279 18 元。

电脑学习机类 科特 FCS－90 电脑学习机 电子游戏与家用电脑两机合一，具有中小学阶段 BASIC 程序、幼儿教育、英文打字、绘图、电子琴演奏、自编节目、数据读/写及电子游戏八大功能 675 元（上海市场价 750 元、北京市场价 725 元）。

电子游戏机类 小天才游戏机 IQ－701 豪华型 675 元，IQ－501 实用型 445 元，IQ－301 经济型 395，IQ－201 普及型 375 元。胜天 9900 豪华型 440 元，9000 高档型 390 元，8800 普通型 340 元。天马牌 TM939 型 245 元，TM828 型 225 元，TM737 型 205 元。最新面市日本原装世嘉五代立体游戏机、立体场景、形象逼真、具有大型游艺机效果 965 元。

游戏卡类 超级玛莉、坦克大全 24 元，魂斗罗 30 元、魂斗罗二代 40 元、空中魂斗罗 60 元、赤影战士、星际魂斗罗、四合一强卡均 70 元，八合一强卡 130 元，31 合一 95 元，42 合一 105 元，52 合一 135 元，63 合一 155 元，110 合一综合大全 315 元。世嘉五代立体游戏机适用卡单卡 95 元，二合一卡 135 元等百余种。

电子游戏指南（随卡手册）国内第一本详细介绍任天堂、电脑学习机、世嘉五代立体游戏机及大型游艺机 100 种最受欢迎的游戏节目，包括故事梗概、基本打法及攻关秘诀。50 万字与上下两册，邮购每套 10 元，购机一台或购卡贰盒赠书一册。电子游戏指南月刊（内部资料）每期 2 元，介绍最新游戏卡节目及近期机卡报价。接受九一年订户。

游戏机专用配套电脑学习键盘（BASIC 语言、人机对话、计算、学习英语、作曲等）290 元/台。

快译通：中英文双向翻译电子字典，家庭、企业、学生、科技、涉外人员理想的随身翻译工具，每台 380 元。

索要各种资料、集成电路、游戏卡目录及报价单请附邮费 0.4 元。

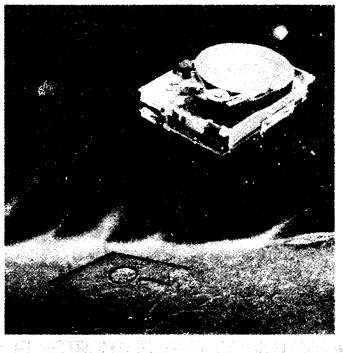
邮购地址：杭州环城北路 57 号三立大厦南二楼

邮编：310006 电话：574439—384 联系人：谭启仁

電子與電腦

3

一九九一年
总期第 72 期



電子與電腦

目　录

·综述·
磁盘和磁盘机(一) …… 林兼(2)
·PC 用户·
数据库通用维护模块设计 …… 徐维祥(4)
字典库的编辑和使用 …… 黄焕如(7)
用 DOS 命令设计菜单 …… 陈凯(8)
在 IBM PC/AT 机上配置虚盘 …… 何管略(9)
IBM PC/XT 机异步通信适配器的自检方法 …… 杨沂　杨耘华(10)
产生真正的随机数 …… 应海涛(11)
利用屏幕缓冲区实现语言的三重调用 …… 张建新(11)
谈谈子目录加密与解密的一种方法 …… 李连德(12)
PC 机显示氢原子轨道波函数 …… 陈可中　曾庆涛　郭冰莹(12)
BASIC 程序 P 加密的简易解密法 …… 李志刚(13)
·学习机之友·
BASIC 语言实现递归 …… 苏亚华(14)
数据处理原理和技巧 …… 朱国江(15)
哥德巴赫偶数猜想的 BASIC 程序 …… 陈君佐(18)
梵塔移动的动态模拟 …… 刘闯(19)
查找内存中的代码 …… 陈宇(20)
奇妙的图案 …… 李明(20)
最简数据盘生成法 …… 江东(20)
超级 8～40 单元打印 …… 王壮(21)
将 16 进制代码翻译成 DATA 语句的数据 …… 张振堂(21)
关于多目录磁道问题 …… 张剑平(22)
电子表键盘输入方式的改进 …… 田洪瑜(22)
LASER310 兼做数字石英钟 …… 郑嘉琦(23)
·C 语言初阶·
第三讲　命令行 Turbo C 和实用程序 …… 李文兵(24)
·学用单片机·
普及型单片机开发工具的软件设计 …… 张培仁　刘振安(28)
·学装微电脑·
学装 μp-80 8 位 A/D 变换部件的制作 …… 易齐干(31)
·电脑巧开发·
MP－I 微电脑加装编码键盘 …… 李志平(36)
扫描式红外线十路遥控器 …… 王刚勤(37)
·初级程序员级水平考试辅导问答·
微型计算机操作系统使用问答 …… 王路敬(40)
·封三·
任天堂游戏机参考电路

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:北京市邮政局
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:张　丽
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

磁盘和磁盘机（一）

林　兼

近来，IBM-PC 和它的兼容机（如长城 0520，ST-PC，HH-PC）逐步进入学校和家庭，引起了越来越多的人的兴趣，许多人也想了解它所用的各种外部设备。今天，我们首先介绍 IBM-PC 所用的磁盘和磁盘机。

所有各种计算机，不论大小，都要有存储设备。就象人的头脑一样，记住各种公式和数据。有些数据需要长期保存，有些数据需要在计算机间交换，就要把它存储在一个单独的“地方”，就象人把数据写在纸上保存一样。这个“地方”在 Apple 机和中华学习机中用磁带和软盘片，在 IBM-PC 中，不用录音磁带，因为它存取速度太慢，可靠性也低。大部分管理程序，如 MS－DOS 操作系统，编好的程序，游戏等等都是记录在软盘片上，然后输入到机器中去执行。计算的结果也是记录在软盘片上取走。这样，输入速度快，交换也方便。要注意的是，Apple 机所用的软盘片和软盘机，和 IBM-PC 所用的是不一样的，并不通用。差别下面我们会讲到。

除了软盘外，IBM-PC 还可以配置硬盘机。目的是扩大存储容量和提高工作速度。

软盘片的种类

软盘片是目前微机最常用的存储介质。它由圆形的聚酯薄膜作片基，两面涂复有磁性材料。外面有一个方形的塑料保护套。早期盘片直径是 8 英寸（200mm），这种盘片现在不常用。最常见的是 $5\frac{1}{4}$ 英寸（130mm）和 3.5 英寸（90mm）的两种。前一种用于 Apple 机和 IBM-PC 机，后一种用于 PS/2 和手提式计算机。它们的外形尺寸如图 1。除了尺寸大小不同外，$5\frac{1}{4}$ 英寸盘片保护套是低质、柔软的。3.5 英寸盘片保护套是塑料的，硬的，很好区别。

除了上述这几种外，新研制的还有 3 英寸，2 英寸等小型软盘片。但目前都没有广泛使用。

$5\frac{1}{4}$ 英寸软盘的种类

$5\frac{1}{4}$ 英寸软盘按用途分有三种：

1. 工作（记录）用盘片，即通常存储数据用的盘片。

从图 1 可以看到，这种盘片保护套上有几个孔。中间大圆孔是主轴孔，当软盘片放进软盘机时，主轴把盘片夹紧，带动软盘片在保护套内旋转。保护套是不动的。中间主轴孔旁有一个小孔，叫做索引孔，它用来确定磁道的起始位置，检查盘片转速。下面有一长圆孔，是供磁头运动进行读写的。侧边有一缺口。当用胶纸把缺口封上时，就不能进行写入，以避免因误操作把软盘上重要的数据破坏了。通常叫它“写保护”。

2. 校准用盘片，用来调整和检查软盘机。通常称为 CE 盘，通俗叫法是“猫眼盘”。

这种校准盘片上面写有特殊的信号，不是用一般复制的办法可以做出来的。因此价钱比普通的盘片贵几十倍。可是它在外形上和普通盘片没有差别，如果不注意，把它当普通盘片使用，就破坏了它的功能，造成不应有的损失。

3. 清洗盘片。由于软盘机的磁头是暴露在周围空气中，难免沾上灰尘或其他脏物。最好要定期清洗。清洗有许多方法。采用清洗盘片是一种方法。这种盘片在外观颜色上与普通盘片不同，因此不易混淆。

不同规格的工作盘片

我们可以把一张软盘片看成是一页书，盘片的两面好比是一页书的两面。盘片的每面分成许多磁道，一条磁道好比是书的“一行”。只不过磁道是一个个同心圆，而不是一行行直线。每条磁道由许多位组成，好比是每行中的一个个字。数据就保存在这些位上，每个位存储一个数据。在计算机中，存储容量通常用字节（Byte）或兆字节（MB）表示。一个字节有 8 位。

对使用者来说，最关心的事是一片盘能保存尽量多的数据。用计算机的述语说就是存储容量尽可能大。但软盘片存储面积是一定的，要存储多必须提高单位面积的存储量。在每条磁道上单位长度上存储的位数叫做“位密度”，单位是每毫米位数或每英寸位数（bpi）。而半径方向单位长度上的磁道数称为“道密度”，单位用每毫米道数或每英寸道数（tpi）。

从存储容量上考虑，当然是位密度和道密度越大越好。但是，由于软盘片要互换，要求在一台软盘机上写好的软盘片在其他同类型的软盘机上也能读出，必须有统一的标准。目前，国际标准规定了每个磁道的位置（同心圆的半径），也规定了道密度。目前 $5\frac{1}{4}$ 英寸盘片只有两种道密度 48tpi 和 96tpi。位密度也只有两种常用，采用改进调频制的 5876bpi，称为“倍密度”。采用 9646bpi 的称为“高密度”。

总的说来，软盘片有单面和双面，有倍密度和高密度，有 48tpi 和 96tpi，排列组合，可以有好几种不同的规格。

Apple 机和中华学习机采用的是单面软盘片，IBM－PC 机只用双面盘片。这是它们间不能互换使用的原因之一。

对于高密度，只用 96tpi。因此，IBM－PC 机所用的盘片实际上只有三种：

双面　倍密度，48tpi
　　倍密度，96tpi
　　高密度，96tpi

存储容量的计算

当我们打开一本书时，可以看到每面有多少行，每行有若干字。这两个数相乘，就得到每面可排下的最大的字数。但由于空行空格，标题等等占去不少位置，实际的字数要比最大字数少。不同的开本最大字数也不一样。

软盘片也一样，我们可以用每面磁道数乘上每磁道存储字节数得到它的“最大存储容量”。这个容量通常称为“非格式化容量”。例如，双面 48tpi 的盘片，每面 40 个磁道，每磁道存储 6250 字节，它的非格式化容量就是：

$2\times40\times6250=500000$ 字节 $=500$ 千字节(KB)

在使用时，不同的操作系统把软盘分成不同的格式。在 IBM－PC 机中通常用的 PC－DOS，把一条磁道分成 8 个或 9 个扇区，每个扇区有 512 字节。因此，一张软盘片可存储的数据量：

当用 8 个扇区时：

$8\times512\times2\times40=328$KB

当用 9 个扇区时：

$9\times512\times2\times40=368$KB

这个容量称为“格式化容量”。

由此可见，格式化容量与采用的格式有关。如果我们把这样的软盘片放在 Apple 机上，它会得出另一种结果来。因此，当我们选择盘片时要注意到非格式化容量，格式化容量的差别。目前，由于互换的要求，大多数盘片采用 PC－DOS 的每磁道 9 个扇区的标准。格式化容量都写明是 360KB。这是 368KB 取整的结果。

对于双面倍密度 96tpi 的盘片，非格式化容量是 1000KB，格式化容量是 720KB。

对于双面高密度 96tpi 的盘片，非格式化容量是 1.6MB(兆字节)。格式化容量是 1.2MB。

软盘片的选择

既然有这么多规格，在使用中如何选择呢？

第一，要采用和徽机规格一致的盘片。如 Apple 机还是 IBM－PC 机。当然，双面盘当单面盘用是可以的，就是价钱贵一些。但单面盘一般不能用作双面盘。因为不用的那面缺陷多，可靠性差。

高密度盘片不能当倍密度用。因为磁特性不同，在倍密度的软盘机上写不进去。

盘片的规格在保护套上都有标明。如倍密度用 DD(Double Density)，高密度用 HD(High Density)，双面用 DS(Double Side)，48tpi 或 96tpi 直接标明，只不过有的用大写字母 TPI 字样。另外，有的厂家不同盘片的标签颜色也不同，容易区别开来。

在购买盘片时，可以用肉眼观察盘片表面的情况。从保护套的长圆孔处可以看到盘表面的磁性层，应当平整，光滑，无小凸起物，也没有小针孔。表面应没有划伤痕迹，也应没有霉斑。颜色应均匀，没有深浅不一。

注意不要用手或其他物体触摸盘片表面。

在外观检查后，应把盘片放到 IBM－PC 机上进行格式化。新盘片格式化后应没有缺陷。

三种 $5\frac{1}{4}$ 英寸盘片规格汇总如下：

规格	位密度 bpi	道密度 tpi	非格式容量 KB	格式化容量 KB
双面倍密度 48tpi (DS,DD)	5876	48	500	360
双面、倍密度，96tpi (DS,DD)	5876	96	1000	720
双面、高密度，96tpi (DS,HD)	9646	96	1600	1200

软盘片使用注意事项

软盘片使用环境要求：

温度 10℃～44℃(50℉～112℉)

相对湿度 20%～80%(不结露)

这些要求一般都能达到。也有个别情况会超出上述范围。例如北方的冬天，相对湿度可能低于 20%，这时盘片因过于干燥，摩擦产生静电，使用时会出错。又例如在南方夏天，机箱内温度会高于 44℃，盘片受热后变形，当然也会出错。

除了环境条件外，使用中还应注意：

1. 要保持盘片使用环境和保存环境清洁。以免污染盘片表面。盘片划伤报废多半是灰尘造成的。因此，微机使用的地方应保持清洁。盘片用后放回纸套内。

2. 不要把软盘片放在靠近有磁场的地方，如：电机、变压器，显示器等设备附近。过大的磁场会把记录的数据抹去。

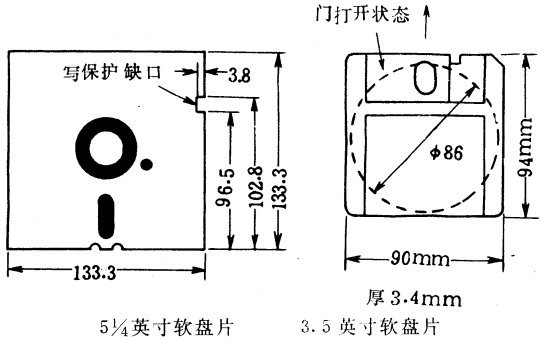
3. 不要弯曲、折叠，重压盘片。因为盘片上有折痕或变形，都会使读写出错。

4. 严禁用手触摸盘片表面。也不能用酒精或其他溶剂清洗盘片表面。

5. 不要把盘片放在高温处或在烈日下爆晒。

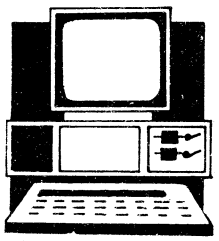
6. 标签应用软水笔写好后贴上去，不要用铅笔或园珠笔写。也不能用橡皮擦标签。因为粉末或油性物会落到盘片表面上，造成读写出错。

软盘片注意使用，一方面延长使用寿命。按标准，软盘片可使用三百万次。但由于使用不小心，使盘片过早报废，没有达到这个指标。另一方面，盘片损坏造成数据丢失，使辛辛苦苦编成的软件或工作结果，毁于一旦。这种情况，也是常常见到的。



5¼英寸软盘片　　3.5 英寸软盘片

图 1



PC用户

数据库通用维护模块设计

锦州铁路运输学校　徐维祥

绝大多数的数据库在实际运用中都要经常进行数据的更新，因而数据库的维护是一项经常性工作。目前不外乎采用两类维护方法：一是为每个数据库编制一套维护模块。对于不同的数据库要设计不同的维护模块，这种重复劳动浪费了程序设计人员很多时间；二是把维护工作留给通晓 DBASEⅢ命令的人员去完成。这样提高了对用户的要求，客观上限制了数据库的普及使用。

能否提供一个通用维护模块，既不要求操作者了解程序原理和 DBASEⅢ命令，又能完成对各种数据库的维护工作呢？笔者进行了尝试，本文把一个操作简便，功能齐全的通用维护程序奉献给读者。

一、通用维护模块应当具备的功能

维护数据库的工作可以归结为：1. 删除一条或一批记录；2. 修改一条或一批记录；3. 替换一批记录；4. 中间插入记录；5. 末尾添加记录。有了这五种基本操作就可以满足对数据库更新的各种需求。

对单条记录的处理以采用记录号定位的方法最简便；处理一批记录则以条件定位的方法最灵活。

为了通用，维护模块应能自动向操作者提供要维护的数据库的结构信息。

为使操作尽可能简便，应对字段名编号，使操作者通过选号而不必输入汉字就能使用字段名。

维护模块应在记录范围、记录应满足的条件及摘录出必要的字段加以处理等方面向用户完全敞开。

二、主要技术问题的处理

首先要解决的问题是能提供任意选定的数据库的结构信息。对此采用的方法是，在第一工作区打开要维护的数据库，用拷贝命令把库的结构拷贝成结构描述库 JBJ。DBF，结构描述库的记录就是要维护的数据库各字段的有关信息。然后在第二工作区打开 JBJ。DBF 库，准备随时提供有关信息。

DBASEⅢ命令的一般形式为：＜命令字＞[范围][字段名清单][条件]。由于要针对未知结构的数据库进行设计，方括号中的三个可选项必须是通用的。只有充分运用这三个选择项，才能保证在任何情况下都能满足维护操作的要求。

在构成定位记录的条件时，要组成＜字段名＞＜关系符＞＜字段值＞的条件表达式。在需要使用字段名时，屏幕提示子模块采用循环方式将结构描述库中的字段序号和字段名一一显示出来，操作者一经指定字段序号，程序就用变量传递及宏代换替换成字段名，免除了输入汉字字段名时既麻烦又易错的弊端。

为了组成灵活的条件表达式，关系符可以采用多种形式。例如，对数值型和日期型字段可采用＝，＜，＞，＜＝，＞＝，＜＞；对字符型字段还可采用＄。＄引进了包含功能，只要字段值中包含给定值，条件就成立，选择性极好。为了操作方便，对这些关系符也做了编号处理，选号即可输入关系符。

不同类型的字段构成表达式时有不同的要求。为了通用，在指定字段之后，程序要通过结构描述库了解该字段的类型，然后用相应的方法接受手工输入的字段值，组成合法的条件表达式。譬如，对数值型字段只接受数码；对日期型字段提示 MM/DD/YY，将数据转换成日期型等。这样防止了字段值类型出错，也避免了输入的内容超过允许的长度。

程序还能将简单的条件表达式复合起来，构成＜简单条件 1＞.AND./.OR.＜简单条件 2＞.AND./.OR.……的形式，使操作者对数据拥有极强的控制力，方便灵活地筛选出要处理的记录。

为了实现在某一范围内进行更新操作，设置了范围选择子模块。自动提示指定库内现有记录数，并接受处理范围的起止记录号，选择满足条件的记录进行处理。对于单条记录，给出相同的起止记录号就可以越过范围和条件选择进入实质性处理。

有时只需修改少数几个字段，为此使用屏幕提示子模块将字段序号和字段名全部显示出来，操作者只需指定序号就可以取出字段名。这段程序包含在循环中，循环一次取出一个字段名，并自动将其用逗号连接起来，再经宏代换处理，放在修改命令的[字段名清单]位置。于是，修改时只列出点到的字段，非常清晰。考虑到有时要对一批记录中的某一字段作同样的修改，程序设置了用输入值一次更改一批记录的替换功能。

实用系统中维护模块内使用的格式文件，通常是针对具体数据库设计的。在通用维护中设计通用格式文件比较复杂。为避开这个难点，程序巧妙地借用了 CHANGE 命令提供的格式，解决了这个问题。

```
* 通用维护模块(TYWH.PRG)
SET TALK OFF
ACCE "请输入数据库名：" TO KM
USE &KM
COPY TO JBJ.DBF STRU EXTE
SELE 2
USE JBJ
SELE 1
H=" "
DO WHIL H<>"0"
CLEA
@1,28 SAY "数据库维护模块功能："
@3,28 SAY "1 2 3 4 5 0"
@4,28 SAY "追插替修删退"
```

```
@5,28 SAY "加入换改除出"
@7,31 SAY "请选号(0-5):" GET H
READ
CLEA
STOR 0 TO B,C,D,E,J,T,Z
STOR " " TO G,MB,TJ
DO CASE
CASE H="0"
LOOP
CASE H="1"
APPE BLAN
CHAN NEXT 1
CASE H="2"
INPU "新记录插入位置:" TO B
GO B
INSE BLAN BEFO
CHAN NEXT 1
CASE H="3"
DO TFW
DO TTJ
SELE 2
DO WHIL E<10
@E+11,0 CLEA
E=E+1
ENDD
@12,5 SAY "要替换哪个字段:"
@12,21 GET B PICT "@Z 999" RANG 1,T
READ
GO B
MB=TRIM(FLELD--NAME)
@12,21 SAY "&MB"
SELE 1
APPE BLAN
MD=&MB
@13,13 SAY "替换成:" GET MD
READ
DELE
PACK
GO C
LOCA NEXT D FOR &TJ
DO WHIL RECN()<D+C-1
REPL &MB WITH MD
CONT
ENDD
CASE H="4"
DO TFW
IF D=2
GO C
CHAN NEXT 1
LOOP
ENDI
SELE 2
COUN TO T
DO TPM
DO WHIL E<=T
B=0
@18,5 SAY "指定要改的字段,回车退出:"
@18,32 GET B PICT "@Z 999" RANG 0,T
READ
IF B=0
EXIT
ENDI
GO B
MB=MB-G-FIELD-NAME
@12,5 SAY MB
G=","
ENDD
MB=TRIM(MB)
DO TTJ
GO C
LOCA NEXT D FOR &TJ
DO WHIL RECN()<D+C-1
CHAN FIEL &MB NEXT 1
SKIP-1
CONT
ENDD
CASE H="5"
DO TFW
IF D>2
DO TTJ
ELSE
TJ="RECN()=C"
ENDI
GO C
LOCA NEXT D FOR &TJ
DO WHIL RECN()<D+C-1
CLEA
DISP
ACCE "拟删除对否(Y/N)?" TO X
IF UPPE(X)="Y"
DELE
ENDI
CONT
ENDD
PACK
ENDC
ENDD
CLOS DATA
DELE FILE JBJ. DBF
SET TALK ON
RETU
*屏幕提示子模块(TPM. PRG)
CLEA
STOR 0 TO I,M,N
GO TOP
DO WHIL . NOT. EOF()
M=INT(I/5)
N=16*(I-5*M)
@M+1,N SAY RECN() PICT "999"
@M+1,N+3 SAY ":"+FIELD-NAME
```

```
I=I+1
SKIP
ENDD
RETU

*条件选择子模块(TTJ.PRG)
SELE 2
A1="="
A2="<"
A3=">"
A4="<="
A5=">="
A6="<>"
A7="$"
SET FILT TO FIELD_TYPE()"M"
COUN TO T
DO TPM
@11,2 SAY "记录需满足的条件:"
DO WHIL J<6
STOR "  " TO Q,K,Z,L
B=0
@18,0 CLEA
@18,8 SAY "指定字段名代号:"
@18,24 GET B PICT "@Z 999"RANG 1,T
READ
GO B
Q=Q-FIELD_NAME
@11+J,20 SAY Q
F=SPAC(FIELD_LEN)
B=0
Y=6
DO CASE
CASE FIELD_TYPE="C"
Z="  7:$"
Y=7
R="@R"
DO TFH
IF A&K="$"
Q=["&F"]-A&K-Q
ELSE
Q=Q-A&K-["&F"]
ENDI
CASE FIELD_TYPE="N"
P="@Z  ##########"
DO TFH
Q=Q-A&K-[VAL("&F")]
CASE FIELD_TYPE="D"
P="@D"
DO TFH
Q=Q-A&K-[CTOD("&F")]
CASE FIELD_TYPE="L"
@18,8 SAY "请输入期望值(T/F):" GET L
READ
@ 11+J,30 SAY "=" +"&L"
IF L="F"
Q=". NOT. "+Q
ENDI
ENDC
B=0
@18,6 SAY "选择{1:与 2:或 3:修改 0:结束}:"
@18,36 GET B PICT "@Z 9" RANG 0,3
READ
Q=TRIM(Q)
DO CASE
CASE B=1
TJ=TJ-Q-  ". AND. "
@12+J,15 SAY  ". AND. "
CASE B=2
TJ=TJ-Q-". OR. "
@12+J,15 SAY  ". OR. "
CASE B=3
LOOP
CASE B=0
TJ=TJ-Q
EXIT
ENDC
J=J+1
ENDD
SET FILT TO
SELE 1
RETU

*关系符子模块(TFH.PRG)
@18,0 CLEA
@18,0 SAY"{1:=  2:<  3:>  4:<=  5:>=  6:<>"
@18,27 SAY Z+"}选号"
@18,37 GET B PICT "@Z 9" RANG 1,Y
READ
K=STR(B,1)
@11+J,30 SAY A&K
@18,0 CLEA
@18,25 SAY "输入期望值:"
@11+J,32 GET F PICT "&P"
READ
F=TRIM(F)
RETU

*范围选择子模块(TFW.PRG)
CLEA
COUN TO S
@11,10 SAY "共有"+STR(S,5)+"条记录"
@13,10 SAY "范围:从第"
@13,20 GET C PICT "@RZ 99999 条" RANG 1,S
READ
@13,28 SAY "至第"
@13,33 GET D PICT "@RZ 99999 条" RANG C,S
READ
D=D-C+2
RETU
```

字典库的编辑和使用

江西拖拉机发动机厂　黄焕如

在 DBASE 应用软件中，经常遇到大量的汉字词组输入。为减少数据冗余，节约存储空间，提高输入速度，一个有效途径就是利用字典库。

所谓字典库就是把可能出现的频率较高的信息，分门别类地存入一系列数据库中，然后把原库或其他库中的这些信息以相应的代码替换。例如，在一个财务凭证传票数据库中，科目名称一般需要 15 个汉字左右（30 字节），如果每个记录都占有这 30 个字节，记录数增加时不仅浪费存储空间，而且输入费时又易出错。于是可以单独建立科目字典库 PZKM. DBF，结构和部分数据如下：

。display structure

字段名	类型	宽库	
DH	字符	6	（科目代号）
KM	字符	30	（科目名称）

Record #	DH	KM
1	101000	在途出口商品
2	102000	库存出口商品
3	103000	委托加工商品
	……	

在原凭证传票数据库中就不需要描述科目名称这个字段了，保留科目代号字段，利用它读取相应的科目名称。为提高查询速度，在建字典库的同时也建立相应的索引文件。索引的关键字为同主库的同一字段。如上例以关键字 DH 建立索引文件 PZKM. NDX。

字典库的内容应采取既可另行编辑，也可输入时现场编辑的方法。单独编辑字典库比较简单，使用时要注意输入字典内容不应重复或太全，以免一些根本用不到或不常用的内容进入字典库而降低查找速度。而现场编辑字典库变化比较多，应结合实际灵活使用。如上例，在一个输入财务凭证传票过程中，往往仅键入科目代号而调用科目字典库内相应的科目名称，由于科目数量太多且各单位常用科目各不相同，显然仅采用单独编辑科目字典库是不适宜的，应结合采用现场编辑、自动进入字典库的方法。以下程序可插入在输入部分程序中，其主要功能是：当输入任一科目代号时，首先在科目字典库内查找该代号，如有此代号则显示相应的科目名称；否则提示是否重建该科目名称，如重建将这新的科目设置在科目字典库中。由于其索引文件和数据库被同时打开，新建立的科目又可随时被调用，这就实现了现场编辑字典库的功能。（参考程序一）

字典库的使用很广泛，其中最重要的就是汉字词组的选择输入。即通过程序在屏幕上显示一定数量的汉字词组和对应的数值，输入某一数值，则相当于输入了相应的汉字词组，大大地提高了输入汉字的速度和准确性。程序二给出了示范，该程序的特点是：字典库内容显示在屏幕右上角窗口内，每页显示五个词组（可根据需要修改），既可向前翻又可向后翻页。选择某一数值，其对应的词组就可调出，另外还可随时退出选择。当字典库内记录数不是其显示个数的整数倍，下翻最后一页时，按最后五条词组排列显示，充分利用了屏幕，且新颖别致。程序二可作为一子程序供输入、检索等模块调用。

附程序一

```
……
SET TALK OFF
CLEAR
USE PZKM INDE PZKM
DH1=SPACE(6)
KM1=SPACE(30)
JJ=.T.
DO WHILE JJ
@2,8 SAY "科目代号:" GET DH1
READ
SEEK DH1
FF="N"
IF EOF( )
@24,0
@24,14 SAY "字典库内无此代号,重建吗?(Y/N)"
GET FF PICT "!"
READ
@24,14
ELSE
JJ=.F.
LOOP
ENDIF EOF( )
IF FF="Y"
@24,14 SAY "请输入科目名称:" GET KM1
READ
@24,0
APPE BLAN
REPL KM WITH KM1,DH WITH DH1
JJ=.F.
ENDIF FF="Y"
ENDDO JJ=.T.
@2,40 SAY "科目名称:"+KM
……
```

附程序二

```
SET TALK OFF
USE PZKM
CLEAR
```

```
@24,0
KK=.T.
DO WHILE KK
N=1
A="  "
@2,40 CLEAR
@0,0
DO WHILE N<6
IF .NOT.EOF( )
@ROW( )+1,40 SAY STR(N,4)+". "+KM
SKIP
ENDIF
IF EOF( ).AND.N<5
N=0
SKIP-5
EXIT
ENDIF
N=N+1
ENDDO
IF N=0
LOOP
ENDIF
SKIP-1
M=RECNO()
DO WHILE .T.
@24,10 say  "请选择:1-5 或 U/D-上/下换页  Q-退出"
SET CONS OFF
WAIT TO A
SET CONS ON
@24,0
DO CASE
CASE VAL(A)<6 .AND. VAL(A)>0
GO M
SKIP VAL(A)-5
@14,0 SAY DH+"  "+KM  (取出选择的词组)
KK=.F.
EXIT
CASE UPPER(A)="U"  (上翻一页)
GO M
SKIP-9
EXIT
CASE UPPER(A)="D"  (下翻一页)
GO M
IF .NOT.EOF()
SKIP
ENDIF
EXIT
CASE UPPER(A)="Q"  (退出)
KK=.F.
EXIT
ENDCASE
ENDDO
ENDDO KK
RETURN
```

用 DOS 命令设计菜单

陈凯

灵活地使用 DOS 的批文件命令、设置提示命令 PROMPT 和显示命令 TYPE，可以设计出用户界面友好的菜单。兹介绍一种设计菜单的方法和技巧。

一、建立标准的 ASCII 码菜单选择项文件。例如，使用 WORDSTAR、EDLIN 等建立《通用软件设计工具包》菜单选择项文件，其名称是 GJ.ACD，内容为：

【通用软件设计工具包】

G1－制表准备	G5－字母转换
G2－打印输出	G6－软件保护
G3－居中输出	G7－拷贝扩充
G4－格式转换	G8－通用编辑

二、用批文件命令（扩展名 BAT）、设置提示符命令和显示命令将菜单选择项与操作提示信息相连接形成一个完整的菜单。批文件命令的文件名是“GJ.BAT”，设置 DOS 系统的提示信息为“选择 G0(G0－退出)～8 后，按 Enter 键：”

```
ECHO OFF
REM 文件名是:GJ.BAT
PROMPT 选择 G0(G0-退出)～8 后,按 Enter 键:
TYPE GJ.ACD
```

三、分别建立各功能模块的批文件命令。制表准备子模块批文件命令的文件名是：G1.BAT”，退出《通用软件设计工具包》子模块批文件的文件名是“G0.BAT”，其它功能模块的文件名的定义类推。

```
ECHO OFF
REM 文件名是:G1.BAT
DBASE G1.PRG
TYPE GJ.ACD
```

```
ECHO OFF
REM 文件名是:G0.BAT
PROMPT $p$g
CLS
```

四、批文件命令 GJ.BAT 和 G0.BAT 的运行结果：

C:\>GJ↙

键入回车后即显示上面的菜单，并在其下显示：

选择 G0(G0－退出)～8 后，按 Enter 键：

由此可知，这种菜单主要作用是显示操作提示的信息，对于一切可以执行文件（扩展名为“.BAT”、“.COM”和“.EXE”）功能的实现没有任何影响。

在 IBM PC/AT 型机上配置虚盘

广东 汕头 金山中学　何管略

DOS 直接使用 IBM PC/AT 型机的内存主区，占 RAM 640KB。运行汉字 dBASE Ⅲ 时，如果采用 CCDOS 2.10，其内存分配大致如下（误差不超过±10KB）

MS-DOS	50KB
CC-DOS	70KB
HZK16	260KB
DBASE－Ⅲ	260KB

它将 260KB 的两级字库，全部装入内存，运行速度确实不慢；但是，由于内存空间用完，没有一点自由空间，以致 dBASE Ⅲ 不能调用 DOS 的外部命令（如 Edlin），使用起来，甚为不便。

CCDOS 4.0 对此作了改进。要装入多少汉字，可由用户指定。通常是装入一半，这样就剩余 120KB 左右的自由空间，可供 dBASE Ⅲ 调用 DOS 的外部命令。但是，对于未装入内存中的汉字的 I/O 操作，每次都要读盘，降低了运行速度，难免美中不足。

有没有既活且快、两全其美的方案呢？有的，这就是下面我们所要介绍的“配置虚盘方案”。

IBM PC/AT 型机一般配有总量达 1MB 的 RAM。其中，主区 640KB，扩展区 384KB。可以开发这 384KB 的扩展内存区，建立虚盘，将 260KB 的全部汉字库，装在虚盘里。于是，对于任何汉字的各种 I/O 操作，都不必去读硬盘或软盘；只读虚盘，可以明显提高运行速度。另外，在 640KB 的主内存区中，又可剩余 250KB 左右的自由空间，供 dBASE Ⅲ 调用 DOS 的各种外部命令；例如长达 170KB 的 PCTOOL 工具软件，也可由 dBASE Ⅲ 的 RUN 命令随时调用，非常灵活方便。怎样建立虚盘？以 CCDOS 2.13E 操作系统为例：

(1)在系统配置文件 Config.SYS 中，增写一项设备驱动程序：Device＝Vdisk.SYS 384 512 32/E。当然，建立虚盘的系统文件 Vdisk.SYS，必须存在根目录中。参数 384 规定虚盘的大小(KB)，512 规定扇区的字节数(B)，参数 32 规定根目录的项数，开关/E 规定虚盘建在扩展内存区(否则虚盘建在主区)。

DOS 对设备驱动程序的项数没有限制，所以在 Config.SYS 文件中，原有的 Device＝Ansi.SYS 可以保持不动，它的加强键盘和屏幕管理的功能，也不受影响。

(2)运行 CCDOS 2.13E 系统，选用将 16 点阵汉字库全部装入虚盘(其盘符规定为 D:)的相应项目。

(3)考虑到虚盘的 384KB 容量，装了 260KB 的字库之外，还有 120KB 的空间，可以用来装载一些常用的命令文件；如果键入 Path D:\，又可提高常用命令文件的运行速度。此项工作，重复性大，宜写成批处理文件，每次建盘后运行一遍。

(4)在虚盘的建立、装载和调试过程中，有时难免丢失 Setup 参数，导致下次开机失灵。建议首先运行 Setup 程序，记录本机的各项参数，特别是硬盘的型号。在关机之前，再次运行 Setup 程序，检查有关参数；如果发现丢失，就及时键入正确参数。

配置虚盘的具体情况，项目如下：

```
VDISK Version 3.30 virtual disk D:
Transfer size adjusted
Buffer size:        384 KB
Sector size:        512
Directory entries:   32
Transfer size:       8
Strike a key when ready……
C:\>
C:\>dir *.SYS
Volume in drive C is JIN－ZHONG
Directory of C:\
CONFIG     SYS      86      4-19-90     10:08p
ANSI       SYS      1664    7-11-86      9:07a
VDISK      SYS      3455    3-17-87     12:00p
        3 File(s)   21495808 bytes free
C:\>type config.sys
Device=Vdisk.sys 384 512 32/E
Device=Ansi.sys
Files=20
Buffers=24

C:\>dir d:
Volume in drive D is VDISK V3.3
Directory of D:\
HZK16              261696    3-19-88     8:59a
EDLIN      COM      7526    3-17-87     12:00p
BASICA     COM     36403    3-17-87     12:00p
SETUP      EXE     15793    1-07-88      5:29p
MODE       OCM     15440    7-24-87     12:00a
CC11       COM      3991    2-11-89      8:53a
CE21       COM      3991    2-11-89      4:38p
CE25       COM      3991    2-14-89      1:25p
CS         COM        98    8-21-87      1:36p
        9 File(s)       38912 bytes free

C:\CS
内存自由空间 517424 字节

.Run CS
内存自由空间 251312 字节
```

IBM PC/XT 机异步通信适配器的自检方法

中国金融学院　杨沂　　中科院地理所　杨耘华

微机的异步通信方式，也就是串行通信方式。通信的发送方和接受方之间的数据信息的传输是在单根数据线上，以每次一个二进制位传送。但是在联机的过程中，一旦数据未能传输成功，就需对每个微机的异步通信适配器(UART)进行检查。一般是采用物理接线的方法，使计算机发出的信号，变成自身接收的信号，以此来检验 UART 的工作情况。除上述方法，还可用软件检验的方法而无需物理接线。

IBM PC/XT 系列微机使用的 UART 是 8250 器件，采用电压和电流环接口，电压接口采用 EIA RS232C 串行通信标准，用－3～－15V 之间的电压表示二进制 1，＋3～＋15V 之间表示二进制 0，允许最大数据传输率为 20kbps，最长可驱动电缆 15 米。8250 有两个信道，即 COM1 和 COM2。我们只对 COM1 进行讨论，对 COM2 只需将端口改一下即可。

在编制通信软件时，对芯片的通信速率、奇偶校验和字长等初始化之后，就可采取程序查询方式或中断请求方式实现通信。查询式通信比较直观，原理也很简单，只要通过对 8250 通信线路状态寄存器进行访问，由其内容来判断能否向 8250 输出数据或由 8250 输入数据。它的主要缺点是要占用大量的机时去查询 8250 的状态，既不经济，效率也低。

另一种方式就是通过硬件中断请求来为输入或输出建立一个循环队列，以完成串行通信的工作。8250 一旦产生了符合用户规定的条件，就通知 CPU。在 CPU 允许后，就将现在运行的程序挂起，先运行中断服务程序，然后再运行原来的程序。这种方式可大量节约机时，提高计算机使用效率。

使用 8250 中断方式进行通信的编程，除了通常的初始化，还需做以下几步：

1. 对 8259A 中断控制器的中断屏蔽寄存器初始化，即允许第 4 级中断。

2. 将 8250 中断服务程序的入口地址填入中断向量表的向量单元中。

3. 初始化 8250 芯片，包括置成中断工作方式。

4. 根据需要，设置中断允许寄存器相应的允许/屏蔽位。8250 允许四级中断：(一)数据接受就绪，(二)发送保持寄存器空，(三)允许接受线路中断(四)允许 MODEM 状态中断，其优先级由(一)至(四)。

IBM PC/XT 系列微机的 UART 有一种特殊的功能，即在调制解调器寄存器初始化时将第 4 位置成 1，这样就使 8250 发出的数据变成了自己接收的数据，而无需外部接线。我们利用这种特殊功能和通信硬件中断的特性，编制了一个小程序，来完成对 UART 的自我检查。程序的主要特点是：

1. 无接线情况下，利用 UART 的循回特性；

2. 使用了 8250 数据接收就绪中断。

程序的主要流程如下：选择是 COM1 口还是 COM2 口；取出原来中断对应的段地址和偏移值，加以保存；然后对 8250 进行初始化；接着利用 MS－DOS 的 INT 21H 的 25H 号中断，将我们所编的中断服务程序入口通知 CPU；并接过异步通信中断向量，写入 8259A 的中断屏蔽寄存器；在调制解调器寄存器初始化时，将第四位置成 1，以建立 UART 的循回方式；并在屏幕上建立两个窗口；从而打开中断，程序进入一个循环。它自动查看是否有信息从键盘输入，有则在键盘输入窗口显示出来，并由 8250 发出，接着再看串口输入缓冲区是否收到新的码，有则在串口输入窗口显示出来。这两个窗口是分别滚动的，一旦按了 ESC 键，则程序恢复串行口中断向量、8259A 中断屏蔽器的原来状态，从而关中断并返回 DOS。通过两个窗口的数据可判断 UART 的工作情况，如果两个窗口的数据能一一对应，就证明 8250 工作正常，否则需进一步诊断。

由于篇幅关系，我们只给出对 8250 的 COM1 方式中断器的建立过程及中断服务程序；如对此有兴趣者，可与编辑部联系。

```
……
mov     ah,35h                  ;取原中断段址偏移址
mov     al,0ch
int     21h
mov     int－seg,es
mov     int－offs,bx
mov     dx,offset com－int      ;设新中断
mov     ah,25h
mov     al,0ch
int     21h
in      al,21h                  ;在 8259A 中打开 COM1
and     al,not(10h)             ;中断
out     21h,al
mov     dx,03fch                ;在 8250 中置循回特性
mov     al,11011b               ;和收码中断
out     dx,al
mov     dx,03f9h
mov     al,1
out     dx,al

……
com－int proc  far
        sti                     ;保存现场
```

```
        push  ax
        push  bx
        push  dx
        push  ds
        mov   ax,cs              ;建立可寻址性
        mov   ds,ax
        mov   dx,03f8h           ;取数据
        in    al,dx
        cli
        mov   bx,con-in
        mov   [com-buf+bx],al    ;将数据放入缓充区
        inc   bx                 ;确定下一数据在缓充区
        cmp   bx,com-buf-len     ;的位置
        jne   com-int1
        xor   bx,bx
com-int1:
        mov   com-in,bx
        sti
        mov   al,20h                    ;中断结束
        out   020h,al
        pop   ds                        ;恢复现场
        pop   dx
        pop   bx
        pop   ax
        iret                            ;中断返回
com-int endp

……

int-offs dw  0                  ;保存原中断地址
int-seg dw   0
com-in dw    0                  ;数据位置
com-buf-len  equ 100            ;缓充区长度
com-buf  db 100 dup('0')        ;缓充区

……
```

产生真正的随机数

上海机械学院　应海涛

程序设计中常用到随机数。虽然 BASICA 语言中有 RND 函数可用来生成随机数，但每一次运行程序，它总是生成同一系列的“随机数”，往往不能达到某些应用目的。本人使用了一点小技巧解决了这个问题，只要在程序前部加上如下语句：

```
10 PRINT  "Press any key"
20 IF INKEY$ =""  THEN A=RND:GOTO 20
```

运行程序时，只要按下任意键，即可生成难以预测的某一系列随机数供使用。20 句利用了运行程序到按一任意键之间的时差(足以让第 20 句循环无数遍)的不确定性，使随机系列的第一个数无法确定。这样就达到每次运行程序，产生的随机数系列难以雷同。

利用屏幕缓冲区实现语言的三重调用

淄博市建设银行　张建新

PC 机分配给屏幕缓冲区的内存为 16K。在段地址 47104(&HB800)中，偏移地址为 0 至 16383。屏幕的分辨率为 640×200。缓冲区每一个字节的一位对应屏幕上的一个点。因此，一显示行的 640 点需要 640/8=80 字节。屏幕显示分偶数行和奇数行，其缓冲区各分配 8K 字节，各为 100 显示行。但偶数行和奇数行实际需要的内存量各为 100×80=8000 字节，比 8K(8×1024)各少 192 字节。这样，分配给屏幕缓冲区的内存比实际使用的要多出 384 字节。偶数行分配地址为 0 至 8191。其中 8000 至 8191 为空余区。奇数行分配地址为 8192 至 16383。其中 16192 至 16383 为空余区。

在混合语言程序设计中，常会遇到数据传递问题。数据传递速度最快、手段最简捷的方法便是内存中转法，即把参与传递的数据置入一足够安全的内存区，进入另一语言系统时，再从此内存区读取。综上分析，在传递数据较少，不影响屏幕美观的情况下，可以利用屏幕缓冲区空余的 384 字节实现数据的中转，也可以存放简短的机器语言子程序。

以下二程序展示利用屏幕缓冲区从 dBASE 调用 BASIC 程序，BASIC 程序又调用机器语言子程序，最后返回 dBASE，同时进行数据相互传递诸功能。

```
10 DEF SEG=&HB800 :P=PEEK(8000)
110 CLS:KEY OFF:SCREEN 1,0:COLOR 1,0
120 LOCATE 1,12
130 LINE(10,16)-(289,36),2,B
140 LINE(11,15)-(290,37),2,B
150 PRINT  "从 DBASE 中取 P 值"
185 LOCATE 5,12:PRINT P
187 LOCATE 7,12:INPUT  "输入 1 数";A
190 SCREEN 2,0
195 PRINT  "调机器语言子程序,做屏幕硬拷贝!"
220 POKE 8000,A
230 PRINT  "把 A 的值送入 DBASE"
320 DATA &H55,&Hcd,&H05,&H5D,&HCB
335 DEF SEG=&HB800:FOR I=8003 TO 8007
340 READ J:POKE I,J:NEXT I
350 X=8003:CALL X
400 SYSTEM
```

```
C)type qwe。prg
clear
set talk off.
SET SEGMENT TO 47104
input  “输入 1 数:”  TO A
I=0
STOR “POKE”+STR(I+8000)+”,“+STR(A) TO L
&L
@3,12 say  “A 的值送入 BASIC 程序 AAA”
run basica AAA
set segment to 47104
@4,12 SAY  “从 BASIC 程序 AAA 中取数”
k=peek(8000)
? k
return
```

谈谈子目录加密与解密的一种方法

黑龙江鸡西市公共汽车公司　李连德

为了使子目录中的程序不被非法用户进入，下面介绍一种用半个汉字对目录名加密的方法：

1. 用 CCDOS 启动，使机器进入汉字操作。

2. 打入 MD 密。

3. 用退格键将“密”字删去一半，然后回车。

此时再执行 DIR 命令，则只能看到＜DIR＞而看不到子目录名。这就使半个汉字成为了“保密字”，从而达到保密目的。

如果遗忘了保密字，子目录就无法进入。

解密方法如下：

1. 用 PCDOS 启动，使其进入西文操作系统。

2. 用 DIR 或显示磁盘目录命令：

```
C)DIR *.〈CR〉
 ╙      〈DIR〉89-09-27 8.25
        1File(1) 19555529 bytes free
```

此时就能看到“密”字的半个汉字的字符“╙”，然后在 ASCII 码表中查出“╙”所对应的值 211。

3. 启动 CCDOS 进入汉字系统，并进入 BASIC。

4. 用 PRINT CHR$(211)+CHR$(A)：

A 是 161 到 254 之间的任意 ASCII 码值。

假设 A 为 178 则：

```
PRINTCHR$(211)+CHR$(178)＜CR＞
硬
OK
```

5. SYSTEM〈CR〉返回 CCDOS

6. C)CD 硬，然后用退格键删去一半＜CR＞回车。

此时便进入到保密字为“密”的子目录中。

一个汉字是由 161 到 254 之间的两个 ASCII 码组成的，当一个汉字被删半后就变成了单 ASCII 码，只有在 PCDOS 才能显示半字的 ACSII 符。可见一个半字所建立的子目录可用多个汉字的半字进入。利用汉字的这种组字规律，可将所有汉字库打印出来。

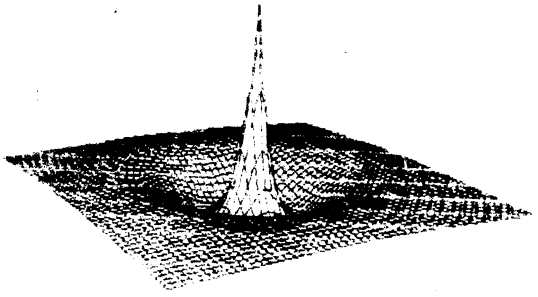
PC 机显示氢原子轨道波函数

陈可中　曾庆涛　郭冰莹

我们在 IBM-PC 机上用编译 BASIC 成功地模拟了氢原子中电子轨道波函数的图象，一则可以帮助同学们澄清一些物理概念和模糊的图象；二则可以激发同学们学习物理理论和电脑的兴趣。现将氢原子中 3S 电子轨道波函数立体图及程序附后：

```
10 SCREEN 2:CLS:PI=3.14159:DIM YH(2800),YL(2800)
20 UM=1000:VM=1000:A=-58*PI/180:A1=COS(A):A2=SIN(A)
30 DU1=40:DV1=40:DU2=10:DV2=10:UP=5500:VP=0:WP=1600:US=600
40 K=.015:W0=6001000!*3.14159∧(-.5)
50 DEF FNF(U,V)=W0/1.732/81*K∧1.5*(27-18*K*(U∧2+V∧2)∧.5+2*K∧2*(U∧2+V∧2))*EXP(-K*(U∧2+V∧2)∧.5/3)-400
60 GOSUB 170:FOR U=UM TO -UM STEP -DU1
70 FOR V=VM TO -VM STEP -DV2:GOSUB 150
80 IF V=VM THEN PLOT=0 ELSE PLOT=2
90 GOSUB 180:GOSUB 330:NEXT V:NEXT U
100 GOSUB 170:FOR V=VM TO -VM STEP -DV1
110 FOR U=UM TO -UM STEP -DU2:GOSUB 150
120 IF U=UM THEN PLOT=0 ELSE PLOT=2
130 GOSUB 180:GOSUB 330:NEXT U:NEXT V
140 K$=INPUT$(1):END
150 UU=U*A1-V*A2:VV=V*A1+U*A2:TRAN=(UP-US)/(UP-UU)
160 W=FNF(U,V):X=VP-TRAN*(VP-VV):Y=WP-TRAN*(WP-W):RETURN
170 FOR I=0 TO 2800:YH(I)=-1000:YL(I)=1000:NEXT I:RETURN
180 XNOW=1300+CINT(X):IF PLOT=0 THEN XLAST=XNOW
190 IF Y)YH(XNOW)THEN GOSUB 230
200 IF Y(YL(XNOW)THEN GOSUB 280
210 IF Y(YH(XNOW)AND Y)YL(XNOW)THEN XLAST=XNOW:PLOT=0
220 RETURN
230 IF XNOW)XLAST THEN XSMA=XLAST:XBIG=XNOW
240 IF XNOW(=XLAST THEN XBIG=XLAST:XSMA=XNOW
```

```
250 YH(XNOW)=Y:FOR SCAN=XSMA+1 TO XBIG-1
260 YH(SCAN)=Y+(YH(XLAST)-Y)*(SCAN-XNOW)/(XLAST-XNOW)
270 NEXT SCAN:XLAST=XNOW:RETURN
280 IF XNOW>XLAST THEN XSMA=XLAST:XBIG=XNOW
290 IF XNOW<=XLAST THEN XBIG=XLAST:XSMA=XNOW
300 YL(XNOW)=Y:FOR SCAN=XSMA+1 TO XBIG-1
310 YL(SCAN)=Y+(YL(XLAST)-Y)*(SCAN-XNOW)/(XLAST-XNOW)
320 NEXT SCAN:XLAST=XNOW:RETURN
330 IF PLOT=2 THEN LINE-(.24*X+320,100-Y/10):RETURN
340 PSET(.24*X+320,100-Y/10),POINT(.24*X+320,100-Y/10):RETURN
```



BASIC 程序 P 加密的简易解密法

潍坊市产品质量监督检验所　李志刚

BASIC 程序可以加 P 参数存盘，这类程序被调入内存后将不能用 LIST 列出，从而达到加密的目的。以前本杂志上介绍过不少解密方法，但大都较为复杂。现向读者介绍一种十分简单的解密方法。

先用 DEBUG 建立一个名为 PJ. BAS 的文件：

```
A>DEBUG
-E DS:0100 FF01
-N PJ. BAS
-R CX
CX 0000
: 0002
-W
-Q
```

需要解密时，在 BASIC 状态下用 LOAD "文件名" 调入要解密的程序，之后用 LOAD "PJ"调入 PJ. BAS，用 LIST 就可列出程序的内容。

我们还可以利用 PJ. BAS 恢复被 NEW 清除的程序，方法是：当执行 NEW 后又想恢复时，就可执行 LOAD "PJ"这样的操作，被清除了的程序便又恢复。

（接第 27 页）

• $<（全文件名宏）　展开成含扩展名的目标名，例如

```
target. obj:target. c
    copy $< A:\usrobj
    tcc -c $*
```

可展开成：

```
target. obj:target. c
    copy target. obj A:\usrobj
    tcc -c target
```

• $:（文件路径名宏）　可展开成如下路径名：当文件名为 A:a\b\c\. c 时，可展开成 A:a\b\。

• $.（带扩展名的文件名宏）　可展开成带扩展名的文件名。如用 $. 可把\a\b\c. c 展开成 c. c。

• $&（文件名宏）　该宏可展开成不带扩展名的文件名。例如用 $& 可把 A:\a\b\c. c 展开成 C。

4. Touch

用于修改一个或多个文件的时间和日期为当前时间和日期，使比依赖于他们的文件更新。用法如下：

Touch　文件名表

文件名可用符 * 或?。对于执行过 Touch 的文件名，执行 MAKE 时，是要重新构造的。

5. Grep

Grep 是功能很强的查找实用程序，用法如下：

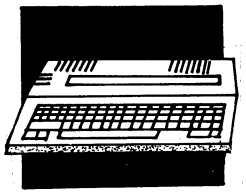
Grep　任选项　查找对象　指定文件

Grep 任选项如下表所示。

-c	只输出匹配行数目	-o	UNIX 输出格式
-d	查找目录	-r	正则表达式查找
-i	忽略大小写区别	-u	更新选择项
-l	列出匹配文件	-v	打印不匹配的行
-n	行前有标号	-w	字查找
		-z	输出查找的文件名

〔练习 3.5〕

```
C>dir *.c
L14     C     102     1-01-80   12:03a
L15     C     46      1-01-80   12:29a
A       C     38      1-01-80   12:35a
        3 File(s) 13687072 bytes free
C>grep printf *.c
File L14. C:
  printf("Tianjin Shida Computer");
  printf("Turbo C  2. 0 exercise");
File A. C:
  printf("Now is a. c");
```



学习机之友

BASIC 语言实现递归

福建宁德师专　苏亚华

一个子程序中，如果包含对自己的调用，则称为递归调用。一般的结构化程序设计语言中都支持递归。递归是解决许多带有回溯性质的算法问题的有效方法。递归的应用还使得程序具有良好的结构和可读性。然而，BASIC 语言是一种非结构化语言，一般都规定子程序不能调用自己，其原因有二：

1. BASIC 语言中没有局部变量的概念，程序中所有变量均为全局变量。同时，子程序的调用不带参数，子程序内部的变量与外部变量只要同名，就是同一变量，不产生屏蔽。如果允许递归，则必然导致许多同名变量出现在不同的各层次子程序中，造成子程序内部变量的混乱。

2. 子程序的嵌套深度有限制，如 APPLE 机 BASIC 语言规定：子程序嵌套深度不得超过 24。递归调用如控制不当，极易造成嵌套深度的越界。

囿于此，在 BASIC 语言的编程中，带有递归性质的问题以往总是靠大量 GOTO 语句来解决，从而使程序结构性、可读性都很差。那么，能不能将结构化编程思想引入 BASIC 语言，实现递归调用呢？事实上，只要我们了解机器对于子程序嵌套的处理过程，对上述两点作适当控制，妥善解决好子程序内部的变量保护和恢复等问题，递归调用是可能的。

为解决 BASIC 语言无局部变量的问题，在递归时，可用数组来区分不同层次中的同名变量。数组的下标表示子程序嵌套的深度，在每层调用之前，先设置好深度值，返回后再恢复。这样，在不同层次子程序中，变量各自独立，互相屏蔽。

我们用一道例题来具体分析如何在 BASIC 语言编程中实现递归。此题是一九八八年福建省青少年计算机程序设计竞赛 BASIC 语言组试卷的最后一题。

[试题]　试编程找出下列字母方阵(图一)中所有隐藏的单词 FUJIAN，单词中字母的走向可以是图二中编号为 1——8 的八个方向，要求打印格式为：

(X,Y),a1,a2,a3,a4,a5

(X,Y)为首字母“F”所在的行、列坐标，a1,a2,a3,a4,a5 是后续每个字母相对前一个字母的走向代号。例如，图一中箭号所示的 FUJIAN 是其中的一种，其打印结果应为

(1,1),3,5,4,6,8

[分析]　题目要求找出所有满足条件的解。因此，必须对所有可能的起点和走向进行搜索，判断是否符合条件。具体过程是：对每一个起点，先选择某一方向前进一步，如果这一步可以接受，就找出下一个字符开始匹配的所有解与其结合，然后返回去再搜索另一方向，直

列坐标

行坐标	1	2	3	4	5	6
1	F	U	J	U	A	N
2	J	J	F	I	U	N
3	N	U	I	J	U	F
4	U	A	N	J	N	A
5	I	A	J	F	U	I
6	J	N	N	I	N	F

(图一)

方向代号表

8	1	2
7		3
6	5	4

(图二)

至八个方向走完。“找出下一个字符开始匹配的所有解”这一过程可以用一个子程序来实现，它是这样描述的：对当前搜索到的点，选择每个走向前进一步，如果这一步可接受，再“找出下一个字符开始匹配的所有解”与其结合，否则换另一走向。这就体现了递归的思想。

设计子程序时，我们用变量 K 表示搜索深度，用数组 T(N)来存放走向代号。每次调用子程序时变量 K 加 1，而把该层所选择的走向代号放入 T(K)中。待调用返回时，再将变量 K 减 1。这样，就能有效地保护有关数据(相当于被屏蔽)。程序如下：

```
10 F$="FUJIAN":N=LEN(F$):DIM H$(6,6),A(8),B(8),X(N),Y(N),T(N)
20 FOR I=1 TO 6:FOR J=1 TO 6:READ H$(I,J):NEXT I,J
30 FOR I=1 TO 8:READ A(I),B(I):NEXT I
50 FOR I=1 TO 6:FOR J=1 TO 6:X(1)=I:Y(1)=J
60 IF H$(X(1),Y(1))=MID$(F$,1,1)THEN K=2:T(2)=0:GOSUB 500
70 NEXT J,I:END
101 DATA "F","U","J","U","A","N":DATA"J","J","F","I","U","N"
102 DATA "N","U","I","J","U","F":DATA"U","A","N","J","N","A"
103 DATA "I","A","J","F","U","I":DATA"J","N","N","I","N","F"
104 DATA -1,0,-1,1,0,1,1,1,1,0,1,-1,0,-1,-1,-1
500 REM 子程序入口
510 T(K)=T(K)+1:IF T(K)>8 THEN 590
520 X(K)=X(K-1)+A(T(K)):Y(K)=Y(K-1)+B(T(K))
530 IF X(K)<1 OR X(K)>6 OR Y(K)<1 OR Y(K)>6 THEN 510
540 IF H$(X(K),Y(K))<>MID$(F$,K,1)THEN 510
550 IF K<N THEN K=K+1:T(K)=0:GOSUB 500:K=K-1:GOTO 510
```

```
560 PRINT "(";X(1);",";Y(1);")";:FOR I=2 TO N:
PRINT ",";T(I);:NEXT
570 PRINT:GOTO 510
590 RETURN
```

60 语句判断首字符的选择。若正确则转子程序执行 500 语句；否则选择下一个首字符位置。由于子程序采用递归算法，主程序的思想十分简单。

子程序的 510 语句进入下一个走向，八个走向都走完后返回。520－540 语句判断该走向是否可接受。如不可接受则转 510 语句选择下一个走向。在可以接受的情况下，550 语句判断是否到达终点：如到达，则打印出一个解；否则再调用子程序（递归），找出下一个字符开始匹配的所有解。

［结论］ 在非结构化语言 BASIC 编程中，一定程度地体现程序的层次关系和模块结构是可能的。程序的模块化，使得设计思想易于理解，程序可读性强，调试和修改也比较容易。

数据处理原理和技巧

南京大学　朱国江

本文介绍数据处理的四种基本操作：数据插入、删除、修改和合并。读者可从中体会菜单技术的应用；模块化程序结构的基本思想；在程序中加注中文说明的方法以及信息处理的设计原理。

一、数据处理原理

1. 数据插入

插入数据的方法比较简单。例如，有下列数组：

33　44　55　61　29

若要在 55 和 61 之间插入 99 这一个数，首先应设法在 55 和 61 之间空出一个空位来，为此，必须把 55 后面的两个数相应地后移一个存储单元。

这种移动必须是最后面一个数据先向右移，然后倒数第二个数据再向右移，这样就空出 55 以后的位置，并保留了 55 以后的其它数据。反之，如果 55 以后的第一个数据先向右移动一个存储单元，则最后一个数据（本例是 29）将被冲掉。

用 BASIC 语句来处理，上述思想简单归结为：D(I＋1)＝D(I)。这个语句的作用是将第 I 个存储单元之值，移动到第 I＋1 个存储空间去。

为保存数据并依次向右移，用循环语句来控制将是十分方便的。由于要插入新的数据，原有数组开辟的内存就不够用了。故 DIM 说明语句可用 DIMD(20)来解决（设原有 10 个数，增加到最多 20 个数）。

插入数据的实现，见程序中 140－360 句程序段。

2. 数据删除

删除数据的概念是不难理解的。如删除下列数据中的一个记录 66

98　97　66　53　42

首先将 66 删去，然后让 53 填入 66 的位置，最后将 42 再填补到原来 53 的地方。

上述思想是将后面的数据向前移，从而删除了原有的信息。但是，数据删除必须考虑：

・除要删除的数据外，所有原来在被删除信息后面的全体数据，应该依次向前顺移；

・删除一个数据后，原数据组的最后一个记录也应删去。

总之，删除数据（或信息）的思路，可以概括为一个语句：即 D(I)＝D(I＋1)。为了实现上述思想，合理安排循环控制是至关重要的。

删除数据的原理，见程序段 370－550 句。

3. 数据修改

数据修改的问题相对于信息的插入、删除要简单得多。而利用下标变量或者字符串变量更为方便。例如，要更改字符串，只要设法指出存储在那一个单元的信息要改，然后重新赋给修改的内容即可。更简单地说是这样的：若 D$(5)中存有 PPI，今要改成 RHI，只要键入 N＝5 和 N$＝RHI，那么，D$(N)＝N$ 即可完成信息更改。

数字修改更为方便，可令 D(N)＝M，其中 N 指修改的位置，而 M 为修改的数据。

完成修改数据的程序段，见 560－740 句。

4. 数据合并

数据合并是指将若干组数据合并在一起。为叙述方便，暂定一组数为 8 个，另一组为 10 个，各自已按数值由大到小顺序排好。现在要求把这两组数合并在一起，并按从大到小的顺序重新存储。

用数组 A(9)、B(11)作为原来两组数的存储单元，每组数组的最后一个存储单元作为附加单元备用。用数组 X(19)作为合并后全部数据记录的存储单元。

设置三个指针：I、J、N，让它们分别指向数组 A、B、X 的位置。开始时令 I＝1，J＝1，N＝0，这就表示两组数据尚未合并时，I 指向 A 数组首地址的数据 A(1)，J 指向 B 数组的首地址数据 B(1)，而 N＝0，则表示指向 X 数组的地址下标为零的位置，此时 X(0)＝0，表示没有数据存入。

程序设计的主要思路：

开始，让每个数组中各自最大的一个值进行比较，那一个值大，就把最大值存入 X 数组，并把对应数组

的指针加 1。例如 A(I)>B(J),则大数 A(I)的值存入 X(N)中,并使指针 I=I+1;若 A(I)<B(J),大数是 B(J)里的值,应让 B(J)的值存入 X(N)中,同时使指针 J 加 1。如此不断循环,直至实现两组数据的合并,而合并后的数据也是顺存排列的。为判断合并是否完成,可以安排 N 指针作为计数器,本例中 N=18 时,表示合并完成。

数据合并的程序段安排在 750—910 句。

二、几点技巧说明

1. DIM 语句安排

各功能块所用数组说明,全部放在 20 句,这样可防止多次运行功能块时,出现数组重复说明的错误。

2. RESTORE 设置

30 句中设置了恢复数据语句 RESTORE,这是整个程序连接中必不可少的,否则各模块完成该模块功能后,返回总控时,其它各模块无数可读。

3. HOME 语句作用

130 句用了清屏语句 HOME,目的是将屏幕清除干净,否则菜单提示和运行结果混在一起。但在打印程序运行结果时,应删去 HOME 语句,改 130 句为 PRINT:PRINT:GOTO 30,否则发生打印格式不统一,以及打印机空走等情况。中华学习机就属此例。

4. 循环延时技巧

730 句和 900 句各安排一个空循环,起延时作用,否则由于迅速返回总控显示菜单而看不清运行结果,这是 130 句 HOME 语句造成的。

5. 虚读技术

760 句安排了一个虚读技术,这是因为 750—910 句是处理两组数据的合并问题,两组数据放在 880—890 句两个 DATA 语句中,如果不安排 760 句的条件判断和虚读技巧,则发生 350 句的数据和 880 句的数据合并,从而造成程序错误并中断。

6. 防止出错处理

80 句中 INPUT K$ 是等待用户响应,由操作者输入希望完成的功能代号。110 句是防止程序乱跑(BOMB PROOFING)。程序中四种操作是用 1 到 4 这四个功能代号来响应的,若使用者按错数字键,则程序可能"爆炸",所以安排了"防爆"措施。这样做实际上是增加了是否在 1 到 4 以内的逻辑检查,如出错,则自动返回重新输入。它属于提高程序质量的方法。

7. 输入采用字符串变量

程序中的输入采用字符串变量 K$ 而不用数值变量,有两方面的考虑:其一是防止和处理的数值混淆;其二是输入方式灵活方便,既可输入字符,也可输入数字。例如,要结束可键入"#"号,经 90 句结束处理。而输入"1","2","3","4",也可以理解是字符,经 100 句 VAL(K$)函数变成数值。

8. 模块化的程序结构

这是编制程序常用的方法。120 句是在分析用户的要求后进行分支处理,显然用开关控制语句 ON/GOSUB 最为合适。120 句后面的行号,则是本程序相应 4 个功能模块的入口。每个独立的模块,分别完成注释中的相应功能,在每个模块结尾处都设置了 RETURN 语句,以使在完成本模块功能后,正确返回总控,根据需要再作其它操作。

```
10 REM 数据处理原理
20 DIM D(20),A(9),B(11),X(19)
30 RESTORE
40 PRINT"1.数据插入"
50 PRINT"2.数据删除"
60 PRINT"3.数据修改"
70 PRINT"4.数据合并"
80 INPUT K$
90 IF K$="#"THEN END
100 MO=VAL(K$)
110 IF MO<1 OR MO>4 THEN 80
120 ON MO GOSUB 140,370,560,750
130 HOME:GOTO 30
140 REM 数据插入
150 FOR I=1 TO 10
160 READ D(I)
170 PRINT D(I);"";
180 NEXT I
190 PRINT
200 K=0
210 INPUT  "N=";N:REM 数据插入位置
220 IF N=-1 THEN 360
230 FOR I=10+K TO N STEP -1
240 D(I+1)=D(I)
250 NEXT I
260 K=K+1
270 INPUT "A=";A:REM 插入的数据
280 D(N)=A
290 PRINT
300 FOR I=1 TO 10+K
310 PRINT D(I);"";
320 NEXT I
330 PRINT
340 GOTO 210
350 DATA 12,47,35,26,61,83,90,55,27,88
360 RETURN
370 REM 数据删除
380 FOR I=1 TO 10
390 READ D(I)
400 PRINT D(I);"";
410 NEXT I
420 PRINT
430 K=0
440 INPUT "N=";N:REM 删除的位置
450 IF N=-1 THEN RETURN
460 FOR I=N TO 9
470 D(I)=D(I+1)
480 NEXT I
490 K=K+1
500 PRINT
510 FOR I=1 TO 10-K
```

```
520 PRINT D(I);"";
530 NEXT I
540 PRINT
550 GOTO 440
560 REM 数据修改
570 FOR I=1 TO 10
580 READ D(I)
590 PRINT D(I);"";
600 NEXT I
610 PRINT
620 REM 连续修改
630 FOR I=1 TO 10
640 INPUT "N=";N:REM 修改的位置
650 IF N=-1 THEN 690:REM 修改结束,显示,返回
660 INPUT "M=";M:REM 修改的数据
670 D(N)=M
680 NEXT I
690 PRINT
700 FOR I=1 TO 10
710 PRINT D(I);"";
720 NEXT I
730 FOR I=1 TO 2000:NEXT I
740 RETURN
750 REM 数据合并
760 IF MO=4 THEN FOR I=1 TO 10:READ S:NEXT I
770 FOR I=1 TO 8:READ A(I):NEXT
780 FOR I=1 TO 10:READ B(I):NEXT
790 I=1:J=1:N=0
800 N=N+1
810 IF A(I)>B(J) THEN LET X(N)=A(I):I=I+1:GO-
TO 830
820 X(N)=B(J):J=J+1
830 IF N<18 THEN 800
840 FOR I=1 TO 18
850 PRINT X(I);",";
860 NEXT
870 PRINT
880 DATA 98,97,89,85,84,71,69,63
890 DATA 96,94,92,90,87,83,72,69,60,55
900 FOR I=1 TO 2000:NEXT I
910 RETURN
]RUN
1.数据插入
2.数据删除
3.数据修改
4.数据合并
? 1
12 47 35 26 61 83 90 55 27 88
N=3
A=5555

12 47 5555 35 26 61 83 90 55 27 88
N=-1

1.数据插入
2.数据删除
3.数据修改
4.数据合并
? 2
12 47 35 26 61 83 90 55 27 88
N=3

12 47 26 61 83 90 55 27 88
N=-1

1.数据插入
2.数据删除
3.数据修改
4.数据合并
? 3
12 47 35 26 61 83 90 55 27 88
N=3
M=7777
N=-1

12 47 7777 26 61 83 90 55 27 88
1.数据插入
2.数据删除
3.数据修改
4.数据合并
? 4
98,97,96,94,92,90,89,87,85,84,83,72,71,69,69,
63,60,55,

1.数据插入
2.数据删除
3.数据修改
4.数据合并
? #
```

(接第 48 页)

31. 使用打印汉字驱动程序应当注意哪些问题?

应当注意下面三个问题:

(1)打印汉字驱动程序应与打印机的型号配套。不同型号的打印机,其驱动程序不同。

(2)打印汉字驱动程序应与要求的汉字库配套。例如长城 0520CH 机,在 GWBIOS 3.00 支持下,3070 打印机的打印驱动程序 3070C·EXE 要求 CLIB24 字库文件的长度为 588K,而 3·COM 则要求 CLIB24 字库文件为 607K。

(3)打印汉字驱动程序应与中文操作系统 CCDOS 版本配套。IBM PC/XT,GW0520A 型机等在 CCDOS 2.0 支持下,3070 打印机的驱动程序是 ALL24P·EXE,而在 CCDOS 4.0 版本下,3070 打印机的驱动程序是 3070P·COM 或 TH3070E·COM。

哥德巴赫偶数猜想的 BASIC 程序

汕头四中　陈君佐

哥德巴赫(Goldbach，1690－1764)是德国人。他在1725年当选俄国彼得堡科学院院士。1742年，他在和大数学家欧拉的几次通信中，提出了关于正整数和素数之间关系的两个推测，用现在的话来说，就是：

1. 每一个不小于6的偶数都是两个奇素数之和；

2. 每一个不小于9的奇数都是三个奇素数之和。

二百多年来，这两个猜想吸引了许多数学家的兴趣。我们以符号{a,b}表示任一个大偶数都可表为一个不超过a个素因子的乘积和另一个不超过b个素因子的乘积之和。据潘承洞教授《哥德巴赫猜想》介绍，1920年前后，有人证明了命题{9,9}。1924年{7,7}被证明。1932年{6,6}被证明。1937年{5,7}，{4,9}，{3,15}被证明。1938年{5,5}被证明。1939年{4,4}被证明。1956年我国数学家王元证明了{3,4}，1957年又证明了{2,3}。1962年潘承洞证明了{1,5}。1962年王元和潘承洞又分别证明了{1,4}。直到1966和1978年陈景润证明了{1,2}，人们正一步步向{1,1}逼近。

下面，我用BASIC语言，编出一个快速寻求{1,1}素数对的程序：

```
10  INPUT A
20  B=INT(A/2)
30  M=0
40  FOR Y=B TO A
50  X=A--Y
60  IF Y=2*INT(Y/2)THEN 250
70  IF X=2*INT(X/2)THEN 250
80  IF X=3 THEN 200
90  IF X=1 THEN 250
100 SY=INT(SQR(Y+0.5))
110 SX=INT(SQR(X+0.5))
120 IF SX=2*INT(SX/2)THEN 150
130 SS=SX+2
140 GOTO 160
150 SS=SX+1
160 FOR N=3 TO SX STEP 2
170 IF Y=N*INT(Y/N) THEN 250
180 IF X=N*INT(X/N) THEN 250
190 NEXT N
200 FOR N=SS TO SY STEP 2
210 IF Y=N*INT(Y/N)THEN 250
220 NEXT N
230 M=M+1
240 PRINT M,X;"    ";Y
250 NEXT Y
280 PRINT
290 PRINT A,M
300 GOTO 10
```

运行程序后，输入100即得出

1	47	53
2	41	59
3	29	71
4	17	83
5	11	89
6	3	97
100	6	

用《哥德巴赫猜想》介绍的把偶数A表成的不同素数对的函数，就是：

$$D(100)=6$$

表示偶数100可表为6种不同数值的Goldbach素数对。

当分别输入200、300、400，可得D(200)＝8，D(300)＝21和D(400)＝14。

在汕头大学计算中心的VAX－Ⅱ机以及两台中型机上收集了大量的D(x)值，发现，有65%的偶数x，满足：

$$D(x)\leqslant\frac{[\pi(x)]^2}{x} \qquad (1)$$

如100以内有25个素数，用数论函数表示，就是$\pi(100)=25$，代入知：

$$D(100)=6\leqslant\frac{[\pi(100)]^2}{100}=\frac{25^2}{100}=6.25$$

$$D(200)=8\leqslant\frac{[\pi(200)]^2}{200}=\frac{46^2}{200}=10.58$$

也即含素因子“2”和“5”的偶数，满足(1)式。

有30%左右的含素因子“3”的偶数，满足：

$$\frac{[\pi(x)]^2}{x}\leqslant D(x)\leqslant 2\times\frac{[\pi(x)]^2}{x} \qquad (2)$$

如：

$$D(300)=21\leqslant 2\times\frac{[\pi(300)]^2}{300}=2\times\frac{62^2}{300}$$

$$=2\times 12.81=25.62$$

$$D(600)=32\leqslant 2\times\frac{[\pi(600)]^2}{600}$$

$$=2\times\frac{109^2}{600}=2\times 19.8$$

$$=39.6$$

只有极少数的富含素因子的偶数，才满足：

$$D(x)\geqslant 2\times\frac{[\pi(x)]^2}{x} \qquad (3)$$

如$x=2\times3\times5\times7\times11\times13\times17=510510$这偶数

$$D(510510)=9493\geqslant 2\times\frac{[\pi(510510)]^2}{510510}$$

$$=2\times\frac{42331^2}{510510}=2\times 3510=7020$$

根据我们电算结果，认为1920年Hardy的渐近公式：

$$D(x)\sim 2C(x)\frac{x}{\log^2 x} \qquad (A)$$

系数 2C(x)偏大，这里

$$C(x)=\prod_{P>2}\left(1-\frac{1}{(p-1)^2}\right)\prod_{\substack{P\mid N\\P>2}}\frac{p-1}{p-2}$$

而陈景润 1978 年给出：对充分大的 N，有

$$D(N)<7.928C(N)\frac{N}{\log^2 N}$$

就更与实际计算有很大的偏离了。

1990 是一个偶数，我把它送入 LASER310 机，在 2 分钟 20 秒内，她就给出 42 对 Goldbach 素数对来。

```
? 1990
1          977          1013
2          971          1019
3          941          1049
4          929          1061
5          887          1103
6          881          1109
7          839          1151
8          827          1163
9          809          1181
10         797          1193
11         773          1217
12         761          1229
13         701          1289
14         683          1307
15         617          1373
16         563          1427
17         557          1433
18         509          1481
19         503          1487
20         491          1499
21         479          1511
22         467          1523
23         431          1559
24         419          1571
25         389          1601
26         383          1607
27         353          1637
28         293          1697
29         281          1709
30         269          1721
31         257          1733
32         179          1811
33         167          1823
34         113          1877
35         101          1889
36         89           1901
37         83           1907
38         59           1931
39         41           1949
40         17           1973
41         11           1979
42         3            1987
```

我从大量计算发现，偶数越大，能找到的 Goldbach 素数对越多。但不知为什么？至今，世界还有些数学家，却坚持说大偶数，找不到(1＋1)的素数对。

梵塔移动的动态模拟

天津市大港一中　刘闯

1990 年七月举行的天津市计算机团体赛的第五题是：动态模拟梵塔的移动，要求打印移动过程如“A→B”等，且移动速度可调。我编写了如下程序，较好地完成了整个过程的模拟。为使移动过程看得更清楚并可暂停，把移速可调改为响铃提示按某键继续。这里对程序稍作说明：

B 数组和 B,F 数组和 F,S 数组分别表示源塔，目标塔，中间塔；

X 数组记录三塔位置；P 数组记录三个塔上的塔盘数；A(a,b)数组中 a 记录塔号，b 为层数，整个记录各塔各盘的宽度。

程序 10～40 设置必要参量。

60 画初始时的 A 柱上各盘。

190～210 打印移动步数及方法，模拟移动过程。

290～320 为画图子程序。

120～280 是求解移动路径。

```
5 REM 河内塔的动态模拟
10 INPUT "塔盘个数:";N
20 INPUT "塔盘高度:";H
30 X(1)=40:X(2)=140:X(3)=239:Y=140:P(1)=N:A(1,1)=48:B=1:CALL 5576:HPLOT 0,141-HTO 279,141-H
35 VTAB 10:PRINT TAB (6)"A"TAB (18)"B"TAB(30)"C"
40 FOR I=2 TO N:A(1,I)=A(1,I-1)-2:NEXT
60 FOR I=1TO N:P=I:GOSUB 290:NEXT
70 DIM N(20),B(20),F(20),S(20)
80 I=0:N(1)=N:J=1
90 B(1)=1:F(1)=3:S(1)=2
100 GOSUB 120
110 END
120 I=I+1
130 IF N(I)=0 THEN I=I-1:RETURN
140 N(I+1)=N(I)-1
150 B(I+1)=B(I)
160 F(I+1)=S(I)
170 S(I+1)=F(I)
180 GOSUB 120
190 VTAB 2:HTAB 16:PRINT J:J=J+1:B=B(I):F=F(I):PRINT CHR$(7):GETA$
200 VTAB 3:HTAB 14:PRINT CHR$(64+B)"→"CHR$(64+F):HCOLOR=4:P=P(B):GOSUB 300:P(F)=P(F)+1
210 A(F,P(F))=A(B,P(B)):P(B)=P(B)-1:B=F:P
```

```
=P(B):GOSUB 290
    220 N(I+1)=N(I)-1
    230 B(I+1)=S(I)
    240 F(I+1)=F(I)
    250 S(I+1)=B(I)
    260 GOSUB 120
    270 I=I-1
    280 RETURN
    290 HCOLOR=3
    300 HPLOT X(B)-A(B,P),Y-H*P TO X(B)-A(B,
P),Y-H*(P+1) TO X(B)+A(B,P),Y-H*(P+1) TO X
(B)+A(B,P),Y-H*P
    320 RETURN
```

查找内存中的代码

陈　宇

将程序清单的代码键入内存指定地址后，以“MEMORY SCAN”为名存盘，备以后使用。然后，再在监控状态下键入 300G 即可。将要寻找的字串长度值放入 $0 单元，要找的字串依次从 $1 单元放起，如要寻找“A9 00”字串，可键入“0:2 A9 00”，接着键入“首地址。末地址 CTRL－Y”，程序便可在首地址至末地址之间的内存中为你寻找这一字串，若找到一处，便印出地址数值。在寻找过程中，可按 CTRL－C 退出。

```
0300— A9 4C 8D F8 03 A9 10 8D
0308— F9 03 A9 03 8D FA 03 60
0310— A9 00 85 FD 85 FE A0 00
0318— A5 00 85 FF B1 3C D9 01
0320— 00 D0 11 C6 FF F0 04 C8
0328— 4C 1C 03 20 66 03 E6 FD
0330— D0 02 E6 FE AD 00 C0 C9
0338— 83 F0 13 E6 3C D0 02 E6
0340— 3D A5 3D 18 C5 3F 90 CE
0348— A5 3C C5 3E 90 C8 A9 8D
0350— 20 ED FD A0 05 B9 76 03
0358— 20 ED FD 88 10 F7 A5 FE
0360— 85 3D A5 FD 85 3C A5 3D
0368— 20 DA FD A5 3C 20 DA FD
0370— A9 A0 20 ED FD 60 A0 BA
0378— C4 CE C9 C6
```

奇妙的图案

宁夏中卫县中学　高一班　李明

我自己根据一个智能绘图仪的原理，编了下面这个程序，别看它很短，画出的图案却非常奇妙。

```
10 HGR:HCOLOR=3:INPUT"R,L=";R,L:D=
75/R:T=3.14159/180
20 HPLOT 140,5+R-L
30 A=A-15:A=A-INT(A/360)*360:I=I+
15/D:I=I-INT(I/360)*360
40 X=140+SIN(I*T)*(75-R)+SIN(A*T)
*L:Y=80-COS(I*T)*(75-R)-COS(A*T)*L
50 HPLOT TO X,Y :GOTO 30
```

此程序在 CEC－I（中华学习机）上通过。程序参数要求：R≤75，L≤R＋5，下面参数可供参考：R，L＝50，45 R，L＝35，25 R，L＝35，15 R，L＝35，40 R，L＝45，30 R，L＝55，45 R，L＝25 30 R，L＝30，35 R，L＝30，15 R，L＝15，10 R，L＝19，13 R，L＝37，42

此程序原理如下：一个大圈，圈内侧有齿轮，还有许多大小不同的小圈，圈外侧也有齿轮，当两圈互相咬合，小圈在大圈做圆周运动时，小圈半径上任一点所走的轨迹就是图。

注：图案的精密程度，可改 30 行语句 A＝A－5 和 I＝I＋15/D 中常数的大小（越小越精密）来调整。

最简数据盘生成法

江　东

利用微机进行各种事务管理，常把系统盘和数据盘各用一张磁盘，以获得最大存储空间。利用下面这个一行程序，可以生成一个不含 HELLO 和 DOS 的 40 磁道纯数据盘。经过两年多的运行使用，在中英文状态下效果良好。附程序清单如下：

```
0 HOME: POKE42344, 76: POKE44723, 4: POKE
46922, 96: POKE44725, 164: POKE46063, 40:
POKE48894,40:PRINT CHR $(4);"INIT HELLO,D2"
```

若去掉最后三个 POKE 语句，则可生成 35 磁道的纯数据盘（均能用 CATALOG 列目录）。

超级 8～40 单元打印

青岛九中微机室　王壮

在 APPLE 与中华学习机中，监控状态下对内存单元的列印只有一种形式，即一行列出八个单元的内容。在显示器上看很清楚，但输出至打印机时，却特别浪费纸。为此本人编写了这个打印程序以节省打印纸不必要的消耗。

本程序使用了内存中 $2000～$20A6 单元，主要是为了避免与其他机器语言程序发生冲突。程序设置了五种打印格式，分别为一行打印 8、16、24、32、40 个内存单元内容。您可以根据所要打印的程序的长短采用不同的打印格式。程序使用前不必用 PR＃1 接通打印机。下面介绍使用方法：

首先运行本程序进行最初的设置，然后在您需要打印时键入如下命令：

首地址. 末地址 CTRL Y 一行打印内存单元个数

除以 8。例如您想打印内存中 $2000～$20A6 的内容，并且一行打印 40 个单元，在监控状态下可键入

2000.20A6 CTRL Y 5

在一行打印单元个数超过 24 时，程序会自动设置打印压缩字体。本程序适用于 CP80、DP80 等 EPSON 兼容打印机。

```
2000-A9 0B 8D F9 03 A9 20 8D   FA 03 60 A2 04 B5 3B 95
2020-FD A5 34 85 B8 20 B7 00   E6 B8 49 B0 F0 07 8D 84
2040-FD E6 08 A2 00 A0 00 A5   06 29 07 F0 02 A2 01 A5
2060-20 DA FD A9 A0 20 ED FD   E6 06 A5 06 F0 21 4C 91
2080-ED FD E8 E0 04 D0 D7 A2   00 20 8E FD 4C 4F 20 E6
20A0-FD 20 93 FE 4C EA 03      05 CA D0 F9 A9 C1 A2 36   20 A3 FE 20 EA 03 20 8E
                               20 C9 06 30 03 4C 99 20   C9 04 30 05 A9 0F 20 ED
                               07 20 DA FD A5 06 20 DA   FD A9 AD 20 ED FD B1 06
                               20 A5 06 C5 08 F0 22 A5   06 29 07 D0 E1 A9 A0 20
                               07 A5 07 C5 09 30 E0 F0   D8 20 8E FD A9 12 20 ED
```

将 16 进制代码翻译成 DATA 语句的数据

天津第七印刷厂设备科　张振堂

在 BASIC 语言编程过程中，经常需要设置一些 READ/DATA 语句，把一段机器语言子程序或一段图形数据 POKE 到指定单元。现将上述过程反过来，即把指定单元的数据由 16 进制变为 10 进制，并存放在 DATA 语句中。

键入程序代码，并以“MAKE DATA”为名存盘：

```
BSAVE MAKE DATA, A$300,L$CE
```

使用时用 BRUN 将其调回内存运行。

也可根据需要将本程序调往其它单元运行，如：

```
BRUN MAKE DATA, A$9500
```

此后，在编制程序过程中，可随时使用 & 命令将内存中代码转换为 DATA 语句。命令格式为

```
&m,n
```

m 为代码的首地址，n 为长度。所生成的 DATA 语句行号为原 BASIC 程序最大行号再加 10。

例一，机器语言代码首地址 1653，长度 34

```
&1653,34
```

本程序生成的 DATA 语句不应超过 249 个字节。对于过长的机器语言代码，可将其分为数段，再通过多次运行本程序生成多条 DATA 语句。

例二，机器语言代码首地址 4000，长度 179

可分为三段：

序　号	首　地　址	长　度
1	4000	60
2	4000＋60＝4060	59
3	4060＋59＝4119	60

分别运行本程序，可生成三条 DATA 语句。

```
&4000,60
&4060,59
&4119,60
0300— D8 18 AD 72 AA 69 18 8D
0308— F6 03 AD 73 AA 69 00 8D
0310— F7 03 A9 4C 8D F5 03 60
0318— 20 46 E7 86 F9 A9 00 85
0320— FC 85 FE 85 FF A9 FC 85
0328— FB A5 67 85 69 A5 68 85
0330— 6A A0 01 B1 69 F0 13 48
0338— 88 B1 69 A6 69 86 FB A6
0340— 6A 86 FC 85 69 68 85 6A
0348— D0 E7 98 91 69 C8 18 B1
0350— FB 69 0A 91 69 C8 A9 00
0358— 71 FB 91 69 C8 A9 83 91
0360— 69 18 98 65 69 85 69 90
0368— 02 E6 6A A9 60 8D 61 AE
0370— A4 FF B1 50 85 44 A2 00
0378— A9 02 85 FD A4 FD 20 44
0380— AE 84 FD D0 06 A4 FE E0
0388— 00 F0 09 E6 FE A4 FE 09
0390— 30 91 69 E8 C6 FD 10 E4
0398— E0 00 D0 05 A9 30 C8 91
03A0— 69 A9 2C C8 91 69 84 FE
```

```
03A8— E6 FF C6 F9 D0 C2 A9 00
03B0— 91 69 C8 91 69 C8 91 69
03B8— C8 98 18 65 69 85 69 90
03C0— 02 E6 6A A9 09 8D 61 AE
03C8— 20 74 D6 4C F2 D4
```

关于多目录磁道问题

天津市南开中学高二　张剑平

贵刊一九九〇年第八期刊登关炳坤同志的《如何在 APPLE Ⅱ 及其兼容机上建立多个目录磁道》(下称《关》文)中,曾有编者按说,此文没有考虑各 VTOC 间协调管理问题。的确,《关》文提供的方法不完善,它未在各 VTOC 中把各目录磁道都置为已使用,因而可能遭到破坏。我对此进行了研究,发现可下列方法解决。

第一种方法:每一次存盘后,把相应的 VTOC 内容读入内存,利用《关》文所附 RWTS 程序,重复其操作步骤 2 即可。

第二种方法是利用本文所附程序对 VTOC 进行改制。其实根本不需每个目录道都有一个 VTOC,只要编一个小程序不断改变 VTOC 中 $01 和 $02 字节中的目录表指针即可。本程序即利用了这种思路。

程序在使用前要根据实际建立多目录道的情况改写 0 语句行的 N 值,然后按照 200 句中 DATA 的形式输入目录道中目录表首区所在磁道与扇区位置。运行后会自动显示已建的各目录道所在磁道位置,并让您选择使用哪一道,再进行 VTOC 的改制。

最好能以本程序作为 HELLO 程序初始化一张新盘,然后按《关》文中的方法建立目录(当然不需每个子目录都拥有一个 VTOC,即不用执行操作步骤 2),接着在 VTOC 中把各目录道所占磁道的自由扇区图置为 $00(表示已全部使用。)或 $01(表示除第 0 区外其它扇区已被占用)。为了在任意目录道下都能正确调用 HELLO 程序,可把标准目录道的 $0F 扇区的内容分别拷入各目录道的 $0F 扇区中,当然不要忘了对各区中的 $01～$02 字节进行相应改动!

第三种方法最简单且行之有效。此法就是把磁道分为若干区域,使每个子目录分区自治,互不干涉。具体方法如下(请参考《关》文来阅读):

1. 初始化一张新盘。

2. 读出 $11 道 0 区,把 VTOC 中的 $84 字节后所有字节都置为 $00,表示这些磁道已使用,再写入 $11 道 0 区。

3. 接着把 $01～$02 改为 $1E,$0F。把 $38～$83 中所有内容置为 $00。把 $84 字节后的所有字节按以下形式改制:FFFF0000FFFF0000,改到 $C4 字节即可。然后把它写入 $1E 道 $00 扇区。(注意:$A8～$AB 中仍为 00)。

4. 读出 $11 道 $0F 扇区,把 $01～$02 字节改为 $1E,$0E 再写到 $1E 道 $0F 扇区中。

5. 把 $1E 道第 $0E～$02 扇区的 $01～$02 字节分别改为 $1E,$0D;…;$1E,$01,再写回到原扇区。

至此一张有 2 个目录道的磁盘就建立好了。

```
0 BUFP=8192:TRK=788:SEC=789:
    CMD=796:N=3
5 HOME:FLASH:HTAB 13:PRINT
    "CHANGE DIR TRACK":NORMAL
10 HTAB 4:FOR I=1 TO 32:PRINT
    "*";:NEXT:PRINT
20 FOR I=768 TO 807:READ A:POKE
    I,A:NEXT
30 DATA 169,3,160,16,32,217,3,169,0,133,72,96,
   0,0,0,0,1,96,1,0,17,0,37,3,0,32,0,0,1,0,0,
   96,1,255,255,255,0,1,239,216
35 PRINT:PRINT
40 FOR K=1 TO N
45 READ C(K,1),C(K,2)
50 PRINT TAB(11);K;"==> SUB T
   RACK:";C(K,1)
55 NEXT:PRINT:PRINT:PRINT
60 HTAB 8: FLASH : INPUT "USE WH
    ICH DIR TRACK?";X: NORMAL
70 POKE TRK, 17:POKE SEC,0:POKE
    CMD,1:CALL 768
80 POKE BUFP+1,C(X,1):POKE BU
    FP+2,C(X,2)
90 POKE CMD, 2:CALL 768
100 NEW
200 DATA T1,81,T2,82,Tn,Sn
```

电子表键盘输入方式的改进

上海市新昌中学　田洪瑜

《电子与电脑》90 年第 7 期《具有电子表功能的键盘输入方式》一文(以下简称《表》文),提供了一种对键盘输入过程进行计时、限时和显时的方法。然而经用秒表测定发现,该程序还存在着一个根本的问题:无键盘输入时的计时虽准确,而有键盘输入时的计时误差却很大。例如,用标准时间 10 秒连续按某数字键输入,电脑电子表的显示却只有 5 秒,如此大的误差,严重影响了实际应用。

经分析发现,《表》文误差产生的原因是:有键盘输入时,程序要多运行 3030 行～3050 行,需要一定的时间,而 3015 行中的计时累加增量 0.079 却没有变化,从而产生了误差。改进的方法是使计时累加增量根据是否有键盘输入而变化,即增加

3008 N=0.079

3012 IF B＞127 THEN N＝0.279

3032 N＝0.079

并将 3015 句改为

3015 K＝K＋N:IF K＜60 THEN 3025

经如此改进，电脑电子表就能准确计时了。

另外，《表》文有三处排版失误，请作如下更正：

3005 VTAB 1:HTAB 3:PRINT“分　”

3015 K＝K＋.079:IF K＜60 THEN 3025

3025 VTAB1:HTAB 1:PRINT K0:VTAB 1:HTAB 5:PRINT INT(K);“秒”

LASER－310 兼做数字石英钟

北京六十一中　郑嘉琦

在 LASER310 上，不必增加硬件，只用软件方法就能使它象一台数字石英钟一样，在屏幕右上角显示时、分、秒。有趣的是，在走时的同时，计算机仍然可以照常使用。感兴趣的诸君不妨一试。其方法如下。

输入并运行下面程序。屏幕第二行将显示“START TIME:?”，这时请键入开始走时的时、分数。例如再过一两分钟就到 9 点 25 分，就键入 9.25。之后屏幕右上角将以反白方式显示出 09.25:00 并同时在第四行显示出“PRESS RETURN TO START!!”。这时请对照手表，等待 9 点 25 分到点时立即按下回车键，屏幕右上角的显示开始走时。再键入 NEW 将程序清除就可以象往常一样使用计算机了。计算机的使用与走时显示将互不相干，“并列”进行。只是不要运行有 MODE (1)的程序，不要使用内存中的高分辨度显示区。

走时的快慢可以用改变两个内存单元中的数值进行调节。29340 单元管粗调，29361 单元管细调。数值加大走时变慢，数值减小走时变快。具体做法可先用 PEEK 查阅这两个单元的当前值，然后再根据实际走时的快慢用 POKE 来改变其中的数值。通过调节可以达到每天走时误差不超过 0.5 秒。一经调定，走时将非常稳定。这是由于走时的稳定程度决定于主机频率稳定度极高的石英晶体振荡器。

```
5 POKE 30845,201
10 DATA 62,50,237,71,62,114,50,127,120,62,101,50,126,120
11 DATA 62,195,50,125,120,201,0,0,0,0,0
12 DATA 71,203,63,203,63,203,63,203,63,246,48,119,35
13 DATA 120,230,15,246,48,119,201,0,0,0,0,0,0
14 DATA 33,24,112,58,1,116,205,75,114,35,54,46,35,58
15 DATA 2,116,205,75,114,35,54,58,35,58,3,116,205,75,114
16 DATA 237,87,61,237,71,192,62,50,237,71,58,3,116,60
17 DATA 39,50,3,116,214,96,39,192,50,3,116,62,61,237,71
18 DATA 58,2,116,60,39,50,2,116,214,96,39,192,0,0
19 DATA 50,2,116,62,59,237,71,58,1,116,60,39,50,1,116
20 DATA 214,19,39,192,0,0,62,1,50,1,116,201
100 POKE 30862,50
105 POKE 30863,114
110 FOR I=29234 TO I+149
115 READ A:POKE I,A:NEXT I
120 CLS:PRINT:INPUT“START TIME:”;A:T=A:E=0
130 IF A＞=13 THEN T=A－12
135 IF A－INT(A)＞=0.6 OR A＞=24 THEN PRINT“REDO”;:GOTO 120
140 IF T＞=10 THEN E=1
150 T=T＊100:A$=STR$(T)
152 IF T＜100 THEN A$=“H”＋A$
153 IF T＜10 THEN A$=“H”＋A$
155 FOR J=1 TO 4:A$(J)=MID$(A$,J＋E,1):NEXT J
160 S=VAL(A$(1))＊16＋VAL(A$(2)):F=VAL(A$(3))＊16＋VAL(A$(4))
170 POKE 29697,S:POKE 29698,F
175 POKE 29699,0:N=0
180 FOR I=28696 TO I+7
190 IF I=28698 THEN 205
195 IF I=28701 THEN 205
200 N=N＋1:POKE I,VAL(A$(N))＋48
205 NEXT I
210 PRINT:PRINT:INPUT“PRESS RETURN TO START!!”;B$
220 A=USR(0)
240 END
```

第三讲　命令行 Turbo C 和实用程序

天津师大计算机系　李文兵

本讲介绍用 DOS 命令行可执行的命令行 Turbo C 和实用程序 CPP、MAKE、TOUCH、GREP 的用法。

1. **命令行** Turbo C

命令行 Turbo C 包括如下程序：TCC——C 编译程序；TLINK——链接程序；TLIB——库管理程序。

命令行 Turbo C 还具有直接插入汇编指令的功能和汇编源程序输出功能(TC 没有这些功能)。

(1)TCC——**命令行编译程序**

TCC 比 TC 有更多的选项(表 3.1)，其用法是：

TCC　任选项　要编译的文件名

表 3.1　TCC 任选项

−I	80/86/286 指令集	−gN	N 次警告后停止
−A	禁止非 ANSI 扩展	−iN	最大标识符长度
−B	通过汇编程序编译	−jN	N 次错误后停止
−C	允许注释嵌套	−k	标准堆栈结构
−Dxxx	定义宏	−I×	传×选项到链接程序。
−Exxx	选择汇编程序名 ·	−mc	紧缩模式
−G	按时间优化	−mh	巨型模式
−Ixxx	包含文件目录	−ml	大型模式
−K	缺省字符无符号型	−mm	中型模式
−L	库文件目标	−ms	小型模式 *
−O	优化转移指令	−mt	微型模式
−S	产生汇编输出	−nxxx	输出文件目录
−M	产生链接映射	−oxxx	目标文件名
−N	堆栈溢出检查	−p	pascal 调用
−Uxxx	未定义宏	−r	寄存器变量 *
−Z	寄存器优化	−u	外部名下打下线符 *
−a	产生字对齐	−v	源级调试 ·
−c	只编译，不链接	−w	允许所有警告
−d	合并重复字符串	−Wxxx	允许×××警告
−exxx	执行文件名	−w−xxx	不允许×××警告
−f	浮点仿真	−y	产生行号信息
−f87	8087 浮点运算	−zxxx	设置段名

几点说明：

①打 * 者为缺省项；②打 · 者为 2.0 版新增任选项；③−×−表示任选项×之否定；

表 3.2　−Wxxx　任选项

(不符合 ANSI 标准)	
−wdup	重定义的标识符不一致
−wret	同时使用了 return 与 return of value
−wstr	标识符不是结构部分
−wstu	标识符是未定义的结构
−wsus	可疑的指针变换
−wvoi	void 函数不能返回值
−wzst	长度为零的结构
(常见错误)	
−waus	赋予标识符一个未用过的值
−wdef	标识符已定义过
−weff	无效代码
−wpar	标识符为从未用过的参数
−wpia	象是不正确的赋值
−wrch	不能实现的编码
−wrvi	函数应返回值
(不常见错误)	
−wamb	二义性运算符(要加括号)
−wamp	函数或数组有多余的 & 运算符
−wnod	函数未定义
−wpro	无函数原型
−wstv	用结构传值
−wuse	标识符未使用
(移植性错误)	
−wapt	无互换性指针赋值
−wcln	常数太长
−wcpt	无互换性指针比较
−wdgn	比较的常数超出范围
−wrpt	返回值转换中无互换性
−wsig	类型变换中丢失位数
−wucp	Signed 与 unsigned char 型指针相混合

〔练习 3.1〕是 DOS 下使用 TCC 的一个实例：

```
C>tcc−IA:\include−IA:\lib−etest li.c
Turbo C Version 2.0 Copyright (c)……
li.c:
Turbo Link Version 2.0 Copyright (c)……
                Available memory 408688
C>type li.c
main()
{printf("abcd.");}
```

(2)TLINK——**链接程序**

TLINK 是不受操作系统控制的。它把目标模块、库模块等链接成可执行的文件。用法如下：

Tlink 目标文件，执行文件，列表文件，库文件

表 3.3　TLINK 任选项

/c	在公共的和外部的符号中大小写有意义
/d	发现库中符号重复的警告
/i	初始化全部段
/l	插入源文件行号
/m	映象文件含公共符号
/n	无缺省库
/s	详尽的段映象
/x	无映象文件

关于 TLINK 的用法，请注意以下几点：

①被链接文件的扩展名可缺省。

②执行文件和列表文件可缺省，但其后的逗号不能缺省；执行文件缺省表示与目标文件同名；列表文件缺省表示与执行文件同名。

③任选项可放在命令行任何位置，如不想产生映象文件，且把源程序行号放入可执行文件时：

TLINK /x /l c0s MyFILE,,,cs

④多个目标文件用空格隔开；第一个目标文件必须是与编译该程序时所用存储模式一致的初始化模块名。初始化模块名与相应存储模式类型见表 3.4。

⑤库表用空格分开。标准库和数学库也须与编译程序所用存储模式一致，其对应关系也见表 3.4。

如果程序使用浮点数，须在命令行写入 EMU.Lib（无 8087/80287，使用浮点仿真库），或 FP87.Lib（有 8087/80287，使用 80x87 浮点库）。

。

表 3.4 存储模式及其相应的文件

存储模式	初始化模块	标准库	数学库
微型	C0T.obj	CT.lib	MATHt.lib
小型	C0S.obj	CS.lib	MATHs.lib
紧凑型	C0C.obj	CC.lib	MATHc.lib
中型	C0M.obj	CM.lib	MATHm.lib
大型	C0L.obj	CL.lib	MATHl.lib
巨型	C0H.obj	CH.lib	MATHh.lib

例如：使用小型存储模式，浮点仿真，链接名为 li 的目标文件，其 TLINK 命令行为：

Tlink c0s li ,,, EMU CS MATHS

(3)TLIB——**库管理程序**：

①功能　为建立目标形式的函数库提供方便。它有增加模块到库、从库中消除模块和从库中抽出一个.obj 文件三种功能。其命令的一般形式是：

TLIB 库名〔OP〕模块名〔OP〕模块名…

(所指定的库)(操作符)(操作模块)

②注意事项：

• 库扩展名为.Lib，操作模块扩展名为.obj。

• 允许使用的操作符如表 3.5 所示

表 3.5 Tlib 的操作符

操作符	功　能
+	增加模块到指定库
−	从指定库中消去模块
*	从指定库中抽取.obj 文件
−+或+−	消失同时又增加(即重新拷贝)
−*或*−	抽取同时又消去

• TLib 只能对整个模块进行操作

下面通过实例综合练习 Tcc、Tlib、Tlink 三个程序的用法。练习 3.2 中给出的序号为操作顺序。

〔练习 3.2〕

(1)编辑求 x^y 的文件 sqr.c

```
C>type sqr.c
long sqr(x,y)
int x,y;
{ int i;
  long j=1;
  for(i=0;i<y;i++)
  j=(long)j*x;
  return(j);
```

(2)用 Tcc 编译该函数

```
C>tcc -c sqr.c
Turbo C Version 2.0 Copyright(c)...
sqr.c:
    Available memory 419722
```

(3)用 dir 查看编译结果

```
C>dir sqr.*
SQR     OBJ     260     1-01-80     12:05a
SQR     C       104     1-01-80     12:33a
        2 File(s) 13703168 bytes free
```

(4)用 Tlib 把 sqr.lib 加入到 maths 库

```
C>tlib c:\turboc\lib\maths+sqr
TLIB Version 2.0 Copyright (c)...
```

(5)编辑调用函数 sqr()的文件 test.c

```
C>type test.c
#include <stdio.h>
#include "sqr.c"
main()
{ int x,y;
  printf("Please Input:\n");
  scanf("%d,%d\n,",&x,&y);
  printf("%d ^ %d=%ld\n",x,y,sqr(x,y));
}
```

(6)编译文件 test.c

```
C>tcc -c test.c
Turbo C Version 2.0 Copyright (c) ....
test.c:
        Available memory 85560
```

(7)查看编译结果

```
C>dir test.*
TEST    C       162     1-01-80     12:14a
TEST    OBJ     459     1-01-80     12:17a
        2 File(s)  13711360 bytes free
```

(8)用 Tlink 对 test.obj 文件进行链接

```
C>tlink c:\turboc\lib\c0s test,test,,c:\turboc\lib
\emu
c:\turboc\lib\maths c:turboc\lib\cs
Turbo Link Version 2.0 Copyright (c)...
```

(9)查看链接结果

```
C>dir test.*
TEST    C       162     1-01-80     12:14a
TEST    OBJ     459     1-01-80     12:17a
TEST    MAP     514     1-01-80     12:30a
TEST    EXE     9544    1-01-80     12:30a
        4 File(s) 13709312 bytes free
```

(10)执行文件

```
C>test
```

```
Please Input:
2,4
2,4
2 ^ 4=16
```

对于源文件 test.c，若使用如下任选项编译，则可直接得到执行文件 test.exe

```
C>tcc -Ic:\turboc\include -Ic:\turboc\lib -ea:test a:test.c
Turbo C Version 2.0 Copyright (c) ....
a:test.c:
Turbo Link Version 2.0 Copyright (c) ....
        Available memory 404038
C>dir a:test.*
TEST   OBJ   562    1-01-80   1:08a
TEST   C     162    1-01-80   12:14a
TEST   EXE   9544   1-01-80   12:04a
        3 File(s)     71680 bytes free
```

2. CPP——预处理程序

CPP 也是不受操作系统控制的应用程序，其功能是对源程序预处理。它最初由编译程序启动。使用 TC 时，我们完全意识不到它。其用法如下：

CPP　　文件名

源文件经 CPP 预处理后，其中的宏指令均被预处理变为扩展名为.i 的文件，如练习 3.3：

(1)编辑含有宏指令的程序

```
C>type exp3_3.c
#define MAX(x,y) (x>y)? x:y
main()
{int a=5,b=3
  printf("MAX=%d\n",MAX(a,b));
}
```

(2)用 CPP 预处理

```
C>CPP exp3_3.c
CPP version 2.0 copyright (c) ....
exp3_3.c
        Available memory 525430
```

(3)用 type 命令查看预处理后的文件

```
C>type exp3-3.i
exp3_3.c 1:
exp3_3.c 2:main()
exp3_3.c 3:{int a=5,b=3;
exp3_3.c 4:printf("%d\n",(a>b)? a:b);
exp3_3.c 5:}
exp3_3.c 6:
```

表 3.6　CPP 任选项

-A	禁止非 ANSI 扩展	-ml	大型模式
-C	允许注释嵌套	-mm	中型模式
-Dxxx	定义宏	-ms	小型模式
-Ixxx	Include 文件目录	-mt	微型模式
-P	包含源文件信息	-nxxx	输出文件目录
-xxx	未定义宏	-oxxx	目标文件名
-gN	N 次警告后停止	-p	pascal 调用
-iN	最大标识符长度	-w	允许所有警告
-jN	N 次错误后停止	-wxxx	允许×××警告
-mc	紧凑模式	-w-xxx	不允许×××警告
-mh	巨型模式		

3. MAKE——程序管理工具

MAKE 也是不受操作系统控制的软件工具。

(1)功能　自动重新编译修改过的文件，避免对所有文件重新编译而造成时间浪费。

(2)Make 的启动　启动 MAKE 的命令行是：

```
A>MAKE↙
```

该命令将编译所需的模块，建立可执行文件。当不指定其它文件且 MAKE 存在时，就执行其内容。要想使用其它文件(如 MYMAKE)，就要用任选项-f，如：

```
A>MAKE -fMYMAKE
```

(3)任选项　MAKE 命令行的选项见下表。

任选项	含　　义
-Didentifier	定义标识符 identifier
-Diden=string	把标识符定义为字串(不能含空格与制表符)
-Idirectory	在 directory 目录内检索包含
- identifier	解除前面所定义的标识符
-s	无该选项，MAKE 显示全部命令；否则不显示。
-n	显示命令但不执行。用于观察 make 文件。
-ffilename	把 filename 作为 MAKE 文件使用。
-?,-h	显示帮助信息。

〔练习 3.4〕

```
C>make -s -fsample.mak
MAKE Version 2.0 Copyright (c) ....
Available memory 247615 bytes
```

(4)MAKE 文件　由目标文件、源文件和命令表组成。目标文件由它所依存的源文件经编译产生。MAKE 文件的一般形式如下：

```
目标文件 1:源文件 1
        命令序列
目标文件 2:源文件 2
        命令序列
   :       :
```

①目标文件须从最左列写起，其后紧跟冒号；

②命令形式为：若干空格〔词头〕命令

目标文件的词头有三个，即@、-num 和-。

@：表示执行命令时，不显示该命令

-num：若指定数值 num，则当退出状态大于 num 时，MAKE 停止执行；若未指定，则当退出值不为零时，MAKE 停止执行，且删除目前的目标文件。

-：不检验退出状态，因此，继续执行 MAKE。

命令能够使用 MS-DOS 的全部命令，但改道命令和滚边命令例外。命令序列中的命令、任选项、文件名等用空格或制表符隔开。

③注释以 # 开头，可跟在源文件表或命令序列之后；若注释单独占一行，就必须从最左列开始。

④目标文件定义与下一定义至少间隔为一空行。

(5)MAKE 文件组成要素　由注释、法则、隐含法则、宏定义、指令等五个基本要素组成。

①注释　是为便于阅读 MAKE 文件，而设置的以 # 为开头的说明，并不被 MAKE 执行。

②法则　用来描述文件间的依存关系与命令，其形式如下：

目标文件〔目标文件…〕:〔源文件·源文件…〕
〔命令〕
〔命令〕

例如：C.obj:c.c d.h
tcc -c -mm -fc.c

其中 C.obj 为目标文件，c.c 为源文件，d.h 为源文件，tcc -c -mm -f c.c 为命令。

法则执行过程如下：

·若目标文件不存在则生成目标文件。

·若目标文件存在，则检查其生成日期与时间，并与对应的每个源文件的生成日期与时间比较。若目标文件较新，则不执行命令；而当源文件较新时，则执行命令，制作相应的新目标文件。

③隐含法则　请看下面这些法则：

```
a.obj:a.c
    tcc -c -mm -f a.c
C.obj:c.c
    tcc -c -mm -f c.c
```

它们的形式完全相同，目标文件名与对应的源文件也分别相同，可以用如下一条隐含法则来表示。

```
-C.obj:
    tcc -c -mm -f $<
```

这条隐含法则表示，所有.obj 文件皆依存于对应的.C 文件，且通过 tcc 命令实现由.C 到.obj 的转变。这种隐含法则也适用于不用法则指定命令的情况。

④宏定义　MAKE 文件用宏定义的形式是：

宏名＝宏展开字符

宏名用不含空格的英文字母、数字序列指定。宏展开字符可用英文字母、数字、空格等构成，末尾是回车换行符。所定义的宏名可按 $(宏名)形式用在 MAKE 文件中。MAKE 将把 $(宏名)置换为宏展开字符。如果宏名未定义，将置换为 Null 字符。例如下面的法则，第一行为宏定义。第 3 行为该宏定义的使用，MAKE 将把 $(MCR)置换为 S。

```
MCR=S
a.obj:a.c
    tcc -c -m$(MCR) -f a.c
```

要想改变存储模式，只要改变宏定义即可。

⑤控制指令　MAKE 的控制指令(directives)有：

·包含指令(!include)　该指令的功能是把 MAKE 文件所指定的文件包含进来，其形式为

!include　“文件名”

有了这条指令就可把不同程序共用的部分放入一个专门文件，这样不管哪个 MAKE 文件就都能使用。

·条件指令(!if,!elif,!else,!endif)

〔用法 1〕	!if	表达式	若表达式成立
		语句 1	则执行语句 1、
		语句 2	语句 2…
	!endif		
〔用法 2〕	!if	表达式	若表达式成立，
		语句 1	则执行语句 1，
	!else		否则执行语句 2
		语句 2	
	!endif		
〔用法 3〕	!if	表达式 1	表达式 1 为真则执
		语句 1	行语句 1；表达式 1
	!elif	表达式 2	为假，表达式 2 为
		语句 2	真则执语句 2；若
	!endif		表达式 1、表达式 2
			皆假，则什么也不
			执行

语句 1、语句 2 包括宏定义、法则、隐含法则、include 指令等。!if 和 ! elif 中的表达式是 32 位带符号的整型表达式，由操作数、运算符构成。操作数可用 10、8、16 进制表示。可用运算符如下：

单项运算符有－、～、!；

双项运算符有＋、－、＊、/、%、≫、≪、&、|、^、&&、||、＞、＜、＞＝、＜＝、＝＝、! ＝；

三项运算符有？:。

·错误指令(! error)　功能是使 MAKE 停止执行，显示文本。一般与条件命令组合使用，如：

```
!if !$d(MCR)
!error MACRO not defined
!endif
```

·解除命令(!undef)解除以宏名定义的宏，用法：

!undef　宏名

⑥包含宏定义　MAKE 的包含宏定义用法有：

·$d(宏名)　宏名定义过，$d(宏名)返回 1；否则返回 0。该宏可与条件命令组合使用，如：

```
!if !$d(MCR)
MCR=S
!endif
```

表示若 MCR 未定义，则把 MCR 定义为 S。

—$＊(基文件名宏)　它可以展开为不带扩展名的目标名。如：

```
OBJ=a.obj  b.obj  c.obj
target.exe= $(OBJ)
  tcc c0$(MCR)a b c, $*,$*,\emu
    math$(MCR)  c$(MCR)
```

其中，$＊用不带扩展名的 target.exe 替代即可。该例中的 $(OBJ)用宏所定义的 OBJ 的目录文件替代即可。此外，该宏也适用于隐含规则，如：

```
.c.obj
  tcc $*
```

(转第 13 页)

学用单片机

普及型单片机开发工具的软件设计

中国科学技术大学　张培仁　刘振安

上期刊出了 KDC-Ⅲ的硬件结构，这里介绍其软件设计。

单片机开发机软件由在 IBM PC 运行的一个组合软件包和 KDC—Ⅲ上的监控程序两部分组成。

KDC—Ⅲ的软件结构如下：

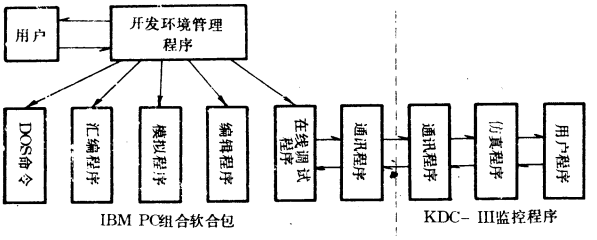


图 1

一、IBM PC 组合软件包主要由编辑程序，汇编程序，模拟调试程序，通信程序，在线调试程序共同组成。编辑程序用于用户汇编语言程序的输入；汇编程序将用户的源程序汇编成可执行的机器码文件；调试程序提供开发机调试命令，用户可通过这些命令控制开发机从而完成一系列的调试功能；组合软件包的各程序均可以直接在 DOS 系统下独立直接运行。

本软件动态地，通过人机对话的方式监督，控制和改变单片机系统的硬软件之间关系，为用户调试用户程序提供了灵活的调试手段。它允许用户检查，修改系统中 8031 内部寄存器 RAM。可单步，连续地执行用户程序。并可随意启停程序的执行；可采用设置断点的方法灵活实现硬软件的查错。

IBM PC 主机方面软件模块框图如下：

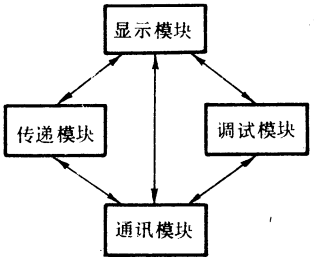


图 2

二、下发机的监控程序

监控程序主要由通信程序和仿真程序组成。通信程序是接收 PC 发来的各种命令，并根据命令把相应的信息返回给 PC；仿真程序主要包括单步，断点，全速运行等命令的处理程序和中断处理程序。

单片机的总监控框图如图 5 所示。

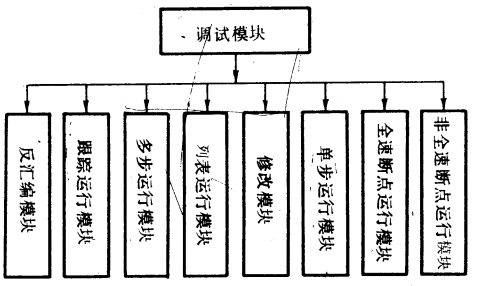


图 3

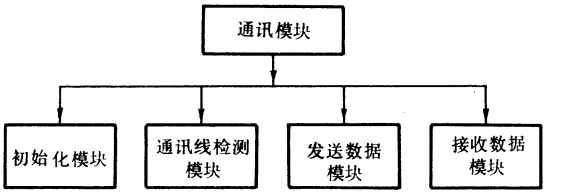


图 4

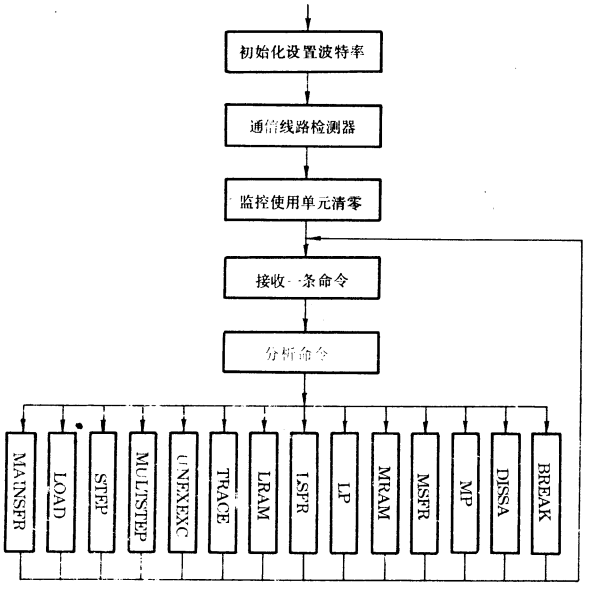


图 5

单片机调试程序框图如图 6

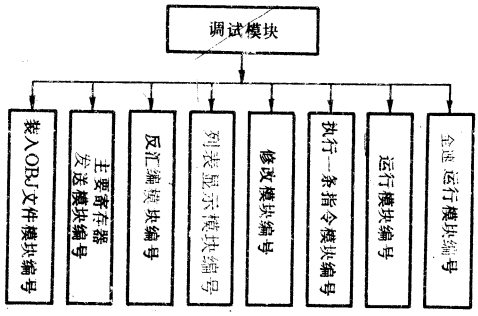


图 6

这里讲的调试程序就是仿真程序的用户界面，也可以讲仿真程序是调试程序的服务程序。调试程序的每一命令都要通过仿真程序中的相应模块在开发机上执行而实施。

(三)监控程序一些设计方法和说明。

监控程序主要由初始化模块，接收分析命令模块，以及各个命令相对应各个执行模块组成和 PC 通信的模块。

初始化模块是仿真器加电或复位后立即执行模块。它包括通信初始化模块和系统环境初始化模块组成。系统环境初始化主要负责系统堆栈，监控使用单元清零。

接收分析命令模块，主要通过调用通信程序的接收模块接收调试程序发送过来的命令码，并根据此命令码调不同的模块完成调试操作。各个命令子模块如 OBJ 文件加载模块，修改模块，单步模块，多步模块，跟踪模块，非全速断点运行模块，全速断点运行模块等。这些子模块都将调用两个最基本最低层两个通用子模块。即是一个是将 8031 当前状态要内容送到 PC 的模块，另一个是控制 8031 执行一条指令的模块。

监控程序在设计时要解决主要问题如下：

1. 单片机资源的争夺的问题。因为我们开发机是单 CPU 的处理机。监控程序(即仿真处理机)和用户程序(目标系统处理机)合用一个 CPU。开发机一般工作方式是用户通过开发机的监控程序来控制用户程序运行和调试，一般情况下运行或调试一段用户程序后又返回监控。这样就产生了监控程序和用户程序运行时都必然会用一些寄存器如累加器 ACC，DPTR 等。为了确保用户程序正常运行。必须在 8031 内部(也可以在片外)设两个堆栈，从而把监控和用户程序有关数据分开。如果共用一个堆栈，就可能产生不可予知的现象。另外在任何时候都要使开发机知道目前是执行用户程序还是监控程序，为此必须有明确的标志。因为监控和用户在中断矢量，寄存器内容，执行指令都会有所不同，所以我们在 8031 内部开了一区域即 6AH～7FH 作为保存用户寄存器的内容和系统堆栈区的区域。从而较好解决单 CPU，监控和用户争夺资源的矛盾。这个方法的缺点是占用 8031 内部 22 字节 RAM。

2. 监控程序要快速地执行各个子命令的模块。

为了提仿真程序的速度，减少代码量，当监控通过接收分析模块接收到某个命令，经过查表跳转的方法完成控制代码分析和相应模块的调用。

我们把命令按顺序编好码如下表所示：

00	传送主要寄存器的内容
01	OBJ 文件加载
02	单步
⋮	⋮
OE	非全速断点运行
OF	仿真程序复位
10	汇编

在仿真程序中再建立一个 17 个长跳转语句组成的跳转散表，(长跳转语句是 LJMP)TABE1 表中每一个语句分别指向一个模块的入口。

由于长跳转指令占三个字节，因此当接收到调试命编码以后乘以 3 再用散转指令 JMP@A＋DPTR 来实现相应模块的调用。

此模块程序如下：

```
BEGIN: LCALL RECEIV
       MOV DPTR,#TABE1
       MOV B,#03H
       MUL AB
       PUSH ACC
       MOV A,B
       ADD A,DPH
       MOV DPH,A
       POP ACC
       JMP @A+DPTR
TABE1: LJMP MAINSFR
       LJMP……
         ⋮
```

(三)单步的实现。

对于一个调试程序控制用户程序一次执行一条指令是必不可少的基本手段。开发机通过单步可进一步实现多步运行，跟踪运行等调试手段。MCS—48 单片机有一个 SS 引脚，可用控制程序单步执行，MCS—96/98 有一条陷井软中断指令 TRACE 可用于控制单步执行。但 MCS—51 没有单独控制单步的引脚。但是利用 MCS—51 中断结构也可以实现单步。

在 MCS—51 中断系统中当另一个同级中断还在进行时，一个中断请求是不会得到响应的，而且 RETI 指令之后，至少要执行一条其它指令(非 RETI，或控制中断的指令)，这个中断请求才会得到响应。具体方法是开始确定单片机 CPU 控制权是否在监控程序手中，而后执行一段监控程序做好执行单步准备工作(包括置中断允许，置中断优先级，置外部中断 0 激发方式是电平方式等)准备工作完成之后，便激活外部中断 0，紧接其后是一条返回指令 RETI。在执行这条指令时那个中断不会被响应，在执行这条指令之后，还要执行一条用户的程序，才又会响应那个中断，从而又一次返回监控。流程图如图 7。

多步，跟踪是基于单步基础上执行的。

多步运行步骤：

①接收程序开始地址。

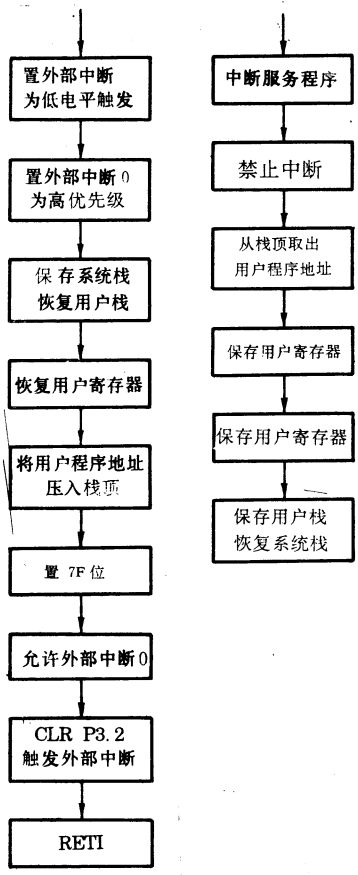


图 7

②接收执行步数。
③调单步操作模块。
④执行步数减 1。
⑤步数不等 0 转到③
⑥传送主要寄存器的值到 PC。
⑦结束。

跟踪运行步骤：
①接收程序的起始地址。
②调单步操作模块。
③传送主要寄存器的值到 PC。
④调反汇编模块。
⑤接收是否继续命令，若继续转到②。
⑥结束。

(四)断点运行的管理与实现。

断点运行对于一个完善的调试系统而言同单步一样是必不可少的调试手段。断点实现有两种方法即硬断点和软断点。我们为了降低成本使用软断点。

用软件方式实现断点运行，一种方法是每执行一条指令(由单步模块完成)就比较一下当前地址是否为断点地址，若已到断点地址就停下来。监控中仿真程序就用这个方法实现非全速断点运行。非全速断点运行由于每执行一条用户指令就需要有一系列的比较工作和监控，用户之间控制权转换的工作要进行。所以相对来说比较慢，我们称它为非全速运行。

非全速断点流程如下：

①接收程序起始地址。
②接收并保存断点地址。
③调单步操作模块。
④比较当前指令地址是否等于断点。若不等转③。
⑤传送主要寄存器的值到 PC 机。
⑥结束。

用软件实现断点运行还有一个方法。这个方法是参照 8086 系统中断点管理方式实现的。即先将断点处内容保存起来，取而代之一条转移到断点服务的程序的跳转指令，然后将控制权交给用户程序，并运行用户程序。直到用户程序运行到断点处为止，通过预先设置好的跳转指令返回监控。此方法为了区别前一种方法，所以称全速断点运行。

流程图如下：
①接收程序起始地址。
②接收并保存断点地址。
③保存断点起的三个字节的用户信息，将长转移指令 LJMP 0016 放入用户设置断点处。
④保存系统栈，恢复用户栈，恢复用户寄存器。
⑤程序起始地址进栈。
⑥执行 RET 指令，启动用户程序。

用户程序运行到断点处，便遇到 LJMP 0016 指令，返回到 0016 监控程序中去，从而控制权又回到监控程序中。

0016 处全速断点运行服务程序流程图如下：
①保存用户寄存器，保护用户栈，恢复系统栈。
②恢复断点处三个字节的用户信息。
③传送主要寄存器到 PC 机。
④结束。

全速断点运行速度快，而且用户程序完全脱离仿真程序(监控)的控制，故用户可占用全部的资源。这是全速断点运行的特点。但是由于这一方法修改断点处的内容，所以用户程序执行时不能和断点处后三个字节有联系，否则可能出错。

(五)用户程序的中断矢量的确定。

监控程序和用户程序都会利用中断执行相应的中断服务程序。MCS—51 规定五个中断源的中断矢量分别在 0003H，000BH……5 个相应地址中。这些地址已由 ROM(2764)构成。也就是不能修改了，所以进入中断以后，首先确定是监控使用中断还是用户使用中断。如果是用户，这时监控程序将用户中断服务程序的入口地址转到 2003H，200BH……相应地址处。2000H 开始给用户装入文件的 RAM 区，所以可以用户写入自己所需要中断服务程序入口地址。

总之单片机开发机的仿真程序还要解决一些其他模块问题，如 OBJ 文件装载，小汇编模块等问题。这里不再一一介绍了。

KDC-Ⅲ开发机在线仿真软件文件名 ASI. PAS，单片机的监控程序文件名 P4. PRT。两个文件全部程序清单可向本刊编辑部函索，请寄印、邮费 15 元。

学装 μP—80

8 位 A/D 变换部件的制作

易齐干

当连续变化模拟量的数据输入给微电脑时，必须把模拟量变换为数字量，本文介绍使用摸拟量/数字量(A/D)变换 IC(东芝 TC5090AP)，将模拟量变换为 8 位数字量的方法。

表 1　TC5090AP 特性

·高精度　±1LSB Max
·功　耗　10mW $\left(\begin{matrix}V_{DD}=5V\\ fosc=1MHz\end{matrix}\right)$
·电　源　5±1.5 V
·变换时间 2ms(Max)(fosc=1.5MHz)
·内部有 CR 基准振荡电路
·三态锁存输出
·输入/输出端为 TTL/CMOS 兼容
·有自动补偿

1. TC5090AP—低功耗积分型 8 位 A/D 变换器。特性如表 1 所示，管脚分配如图 1，管脚说明如表 2 所示。

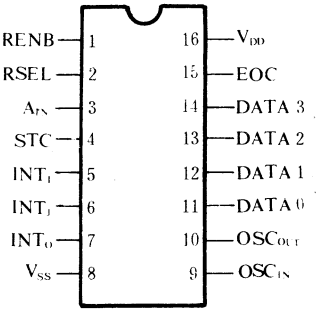


图 1　TC5090AP

表 2　TC5090AP 管脚信号及说明

No	符号	名称与功能	No	符号	名称与功能
1	RENB	(Read Enable) 数据读出信号 "H"：DATA 0～3输出 "L"：DATA 0～3高阻抗	9	OSC IN	Oscillator Inpnt Oscillator Output 输入基准时钟 能使用石英振荡器或外部电阻产生时钟信号。 OSCIN为输入，也可由外部供给时钟信号。
2	RSEL	(Read Select) 4位数据输出是选择高4位或是选择低4位 "H"：高4位输出。 "L"：低4位输出。	10	OSC OUT	
3	A_{IN}	(Analog Input) 模拟量输入 电压输入范围 V_{SS}～V_{DD}	11	DATA 0	3 State Parall l Data Output 输出变换数据 DATA 0 为LSB DATA 3 为MSB
4	STC	(Start Conversion) 变换开始信号，STC下降沿变换开始。在STC下降过程中，RSEL以及RENB至少其中之一为"H"电平。	12	DATA 1	
5	INT I	Integrator Input Integrator Junction integrator Output 外部电阻R_I与电容C_I组成积分器。	13	DATA 2	
6	INT J		14	DATA 3	
7	INT 0		15	EOC	(End of Conversion) 变换结束信号 如果STC下降，则为"L"电平，如果变换结束，返回"H"电平
8	Vss	(Digital Ground) 通常为0V	16	V_{DD}	(Powev Supply) 5V±1.5V

2. 8 位 A/D 变换部件的制作

8 位 A/D 变换部件的电路如图 2 所示。

采用外接电阻可获得时钟频率，外接电阻 $R_f=47k\Omega$，时钟频率为 500kHz。两种关系如下：

10kΩ 为 1.5MHz，20kΩ 约为 1MHz；30kΩ 约为 800kHz。

积分电路的电阻 R_I 与电路 C_I 应满足下式：

$$R_I \cdot C_I=(1.2\sim1.75)\cdot 1/fosc$$

式中：R_I：电阻(MΩ)

C_I：电容(μF)

fosc：振荡频率(kHz)

本文为 $R_I=1.5M\Omega$，$C_I=0.0015\mu F$

模拟量输入出脚 A_{IN} 连续切换开关，可以输入由电位器得到的 0～5V 电压(切换开关在检验侧)，也可输入由热敏电阻温度传感器获得的模拟电压(切换开关在测试侧)。

8 位 A/D 变换数据与 RSEL 端的高电平(H)、低电平(L)相对应(因为高 4 位和低 4 位均在 4 根数据线上输出)。控制端与数据线出脚能够与 μP—80 输入输出部件的端口 C 连接。

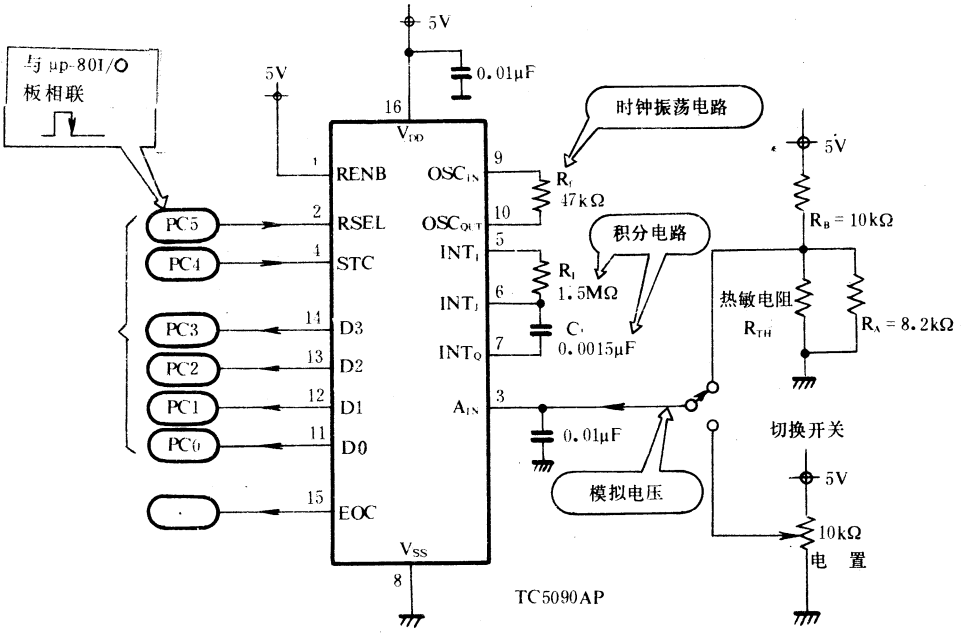


图 2

表 3　温度、热敏电阻与模拟电压的关系

温度　T(℃)	0	10	20	30
热敏电阻　R_{TH}(kΩ)	13.3	8.8	6.0	4.2
合成电阻　R_T(kΩ)	5.1	4.2	3.5	2.8
模拟电压　E_A(V)	1.7	1.5	1.3	1.1
每度的模拟电压ΔE_A(V)	$\frac{0.2}{10}=0.02$	$\frac{0.2}{10}=0.02$	$\frac{0.2}{10}=0.02$	

注　①热敏电阻为石冢电子(株)的5DT型。

②合成电阻R_T是热敏电阻R_{TH}与电阻R_A的合成

$$R_T=\frac{R_{TH}\cdot R_A}{R_{TH}+R_A}$$

③模拟电压由下式求出

$$E_A=5\times\frac{R_T}{R_T+R_B}$$

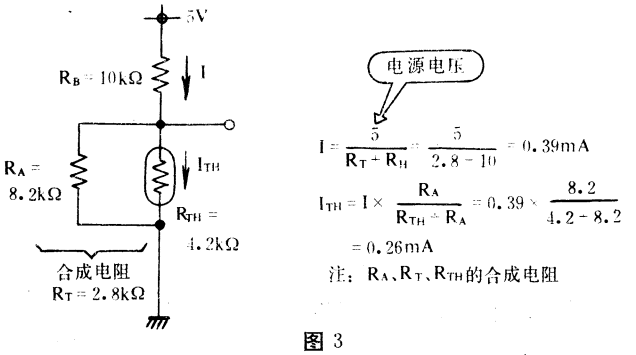


图 3

表 4　温度与 A/D 变换数据对应表

温度(℃)	0	10	20	30
模拟电压(V)	1.7	1.5	1.3	1.1
16进制数	57H	4DH	43H	39H

注：①模拟电压的计算方法参照表A

②温度0℃时讨论16进制计算方法如下式。

因为A　D变换IC为8位，所以平均

1H等于5 256(V)，故下式成立

$$x\times\frac{5}{256}=1.7\ \blacktriangleright\ x=\frac{256\times1.7}{5}\fallingdotseq87$$

将87化为16进制数87 ÷ 16 = 5…7故为57H。

严格地讲，微电脑通过检测 EOC 输出电平，如果变换结束则向微电脑输入数据，STC 脉冲信号的下降沿是变换开始，到变换结束需要 6ms 时间(振荡频率为 500kHz 时)之后，要用软件向微电脑输入数据。

3. 应用举例

本实验 8 位 A/D 变换部件应用于输入温度并显示其结果。

热敏电阻本身温度与电阻值的变化不是线性关系，为谋求线性关系，要并联附加电阻，由热敏电阻获得电压的变化，还要串联电阻 RB(见图 2)。

TC5090AP 的分辨率为 5÷256≒0.02V 即温度变化 1 度电压变化 0.02V，按照这原则决定 R_A 和 R_B 的大小，给以后程序编制带来很大方便。首先，计算出各个温度值的模拟电压这样即可求出 R_A 和 R_B 的大小。现在，R_A=8.2kΩ，R_B=10kΩ，与各个温度相对应的模

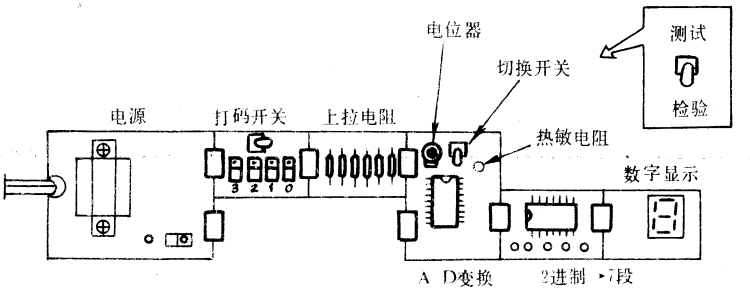


图 4

表 5　A/D 变换部件工作检验表

No	操作电位器	操作打码开关 S_2	操作结果	
1	最左端 (模拟电压为0V)	ON →OFF →ON (STC信号L →H →L)	S_1 OFF	变换数据的高4位显示0
			S_1 ON	变换数据的低4位显示0
2	中间 (模拟电压约2.5V)	〃	S_1 OFF	变换数据的高4位显示7
			S_1 ON	变换数据的低4位显示F
3	最右端 (模拟电压为5V)	〃	S_1 OFF	变换数据的高4位显示F
			S_1 ON	变换数据的低4位显示F

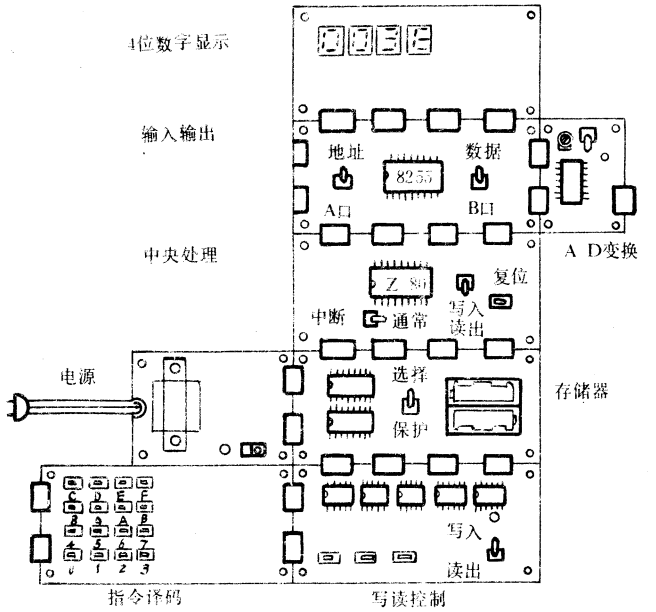


图 9

拟电压如表 3 所示。另外，为获得模拟电压的绝对值，要串联 $R_B=10k\Omega$ 的电阻，其实，$R_B<10k\Omega$ 也可以，但是，如果 R_B 值过小，流过热敏电阻的电流变大，本身发热造成误差，这点决不能忽视，一般流过热敏电阻的电流不超过 0.5mA，按图 3 所列出公式可以计算出流过热敏电阻的电流(30℃时)。

然后，计算各代表性温度所对应的 8 位数据。结果如表 4 所示。

制作完毕，要进行硬件检验。确认没有连线错误、虚焊、跨接等问题之后，将微电脑教育部件的打码开关、上拉电阻、2 进制—7 段、数字显示等部件。按图 4 所示连接。将 A/D 变换部件的切换开关扳至检验侧，操作打码开关 S0(变换开始)与 S1(高位输出数据与低位输出数据切换)，数字显示结果如果与表 5 所示一致，说明工作正确，这样，预先对最初的基本动作进行检验，会给以后带来方便。

4. A/D 变换程序

以上所述表 4 绘制温度与 A/D 变换数据对应表，发现随温度上升，A/D 变换数据变小，这样，使用起来很困难。因此，要如表 6 所示，将 A/D 变换数据进行反转，随温度由低变高，A/D 变换数据也与此相对应由小变大。由表 6 可知，A/D 变换的任何一个数据减去 A8H 之后，进行 10 进制调整，即为温度。

表 6　温度与 A/D 变换数据对应表

温度(℃)	模拟电压(V)	A/D 变换数据(16H)	
		正码	反码
0	1.7	57	A8
1		56	A9
2		55	AA
3		54	AB
4		53	AC
5		52	AD
6		51	AE
7		50	AF
8		4F	B0
9		4E	B1
10	1.5	4D	B2
11		4C	B3
12		4B	B4
13		4A	B5
14		49	B6
15		48	B7
16		47	B8
17		46	B9
18		45	BA
19		44	BB
20	1.3	43	BC
21		42	BD
22		41	BE
23		40	BF
24		3F	C0
25		3E	C1
26		3D	C2
27		3C	C3
28		3B	C4
29		3A	C5
30	1.1	39	C6

现在，由流程图依次考虑温度测试显示程序。简略流程图与 RAM 区域图如图 5 所示。主要子程序的详细流程图如图 7～图 8 所示。要熟练地使用 A/D 变换

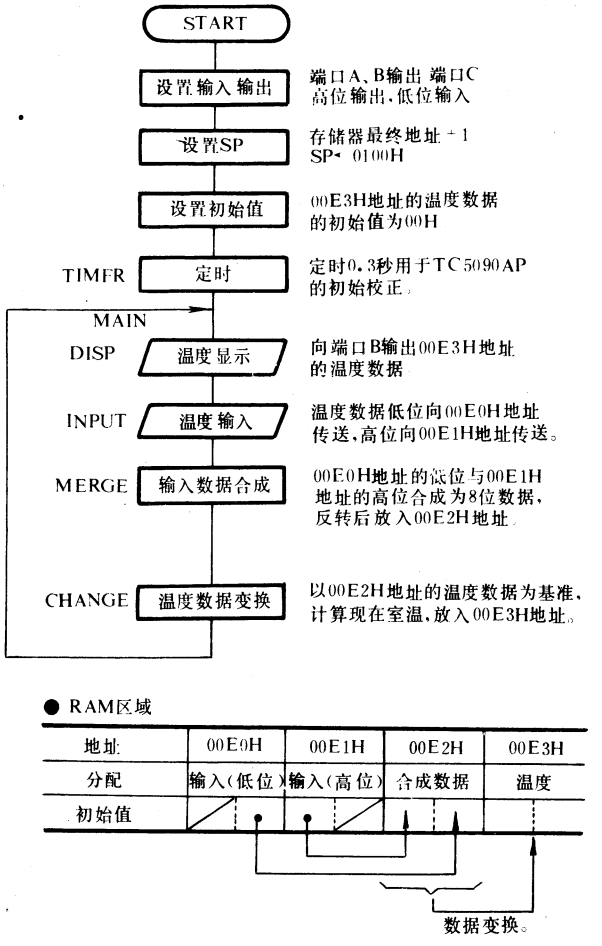

START
设置输入输出
端口A、B输出 端口C
高位输出，低位输入
设置SP
存储器最终地址＋1
SP◂ 0100H
设置初始值
00E3H地址的温度数据
的初始值为00H
TIMER
定时
定时0.3秒用于TC5090AP
的初始校正。
MAIN
DISP
温度显示
向端口B输出00E3H地址
的温度数据
INPUT
温度输入
温度数据低位向00E0H地址
传送，高位向00E1H地址传送。
MERGE
输入数据合成
00E0H地址的低位与00E1H
地址的高位合成为8位数据，
反转后放入00E2H地址
CHANGE
温度数据变换
以00E2H地址的温度数据为基准，
计算现在室温，放入00E3H地址。
● RAM区域
地址
00E0H
00E1H
00E2H
00E3H
分配
输入(低位)
输入(高位)
合成数据
温度
初始值
数据变换。

图 5

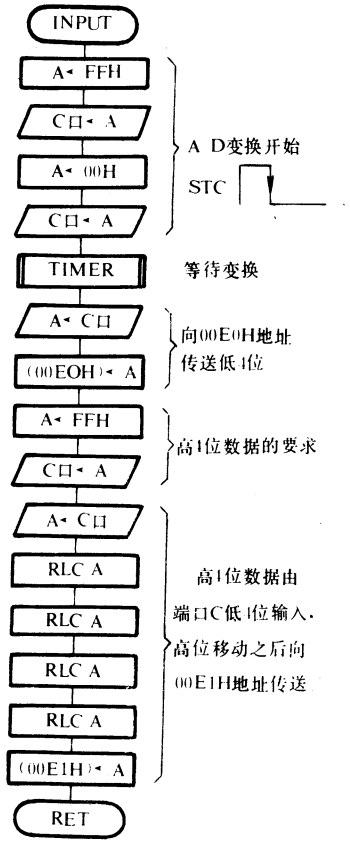

INPUT
A◂ FFH
C口◂ A
A◂ 00H
C口◂ A
A D变换开始
STC
TIMER
等待变换
A◂ C口
(00E0H)◂ A
向00E0H地址
传送低4位
A◂ FFH
C口◂ A
高4位数据的要求
A◂ C口
RLC A
RLC A
RLC A
RLC A
(00E1H)◂ A
高4位数据由
端口C低4位输入，
高位移动之后向
00E1H地址传送
RET

图 6

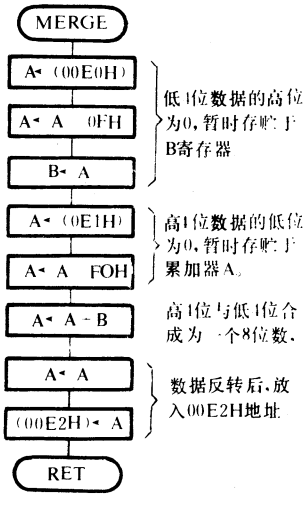

MERGE
A◂ (00E0H)
A◂ A 0FH
B◂ A
低4位数据的高位
为0，暂时存贮于
B寄存器
A◂ (00E1H)
A◂ A F0H
高4位数据的低位
为0，暂时存贮于
累加器A。
A◂ A＋B
高4位与低4位合
成为一个8位数。
A◂ A
(00E2H)◂ A
数据反转后，放
入00E2H地址
RET

图 7

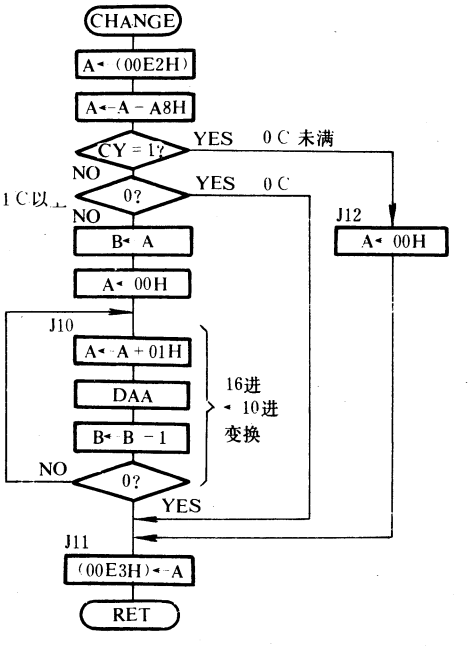

CHANGE
A◂ (00E2H)
A◂－A－A8H
CY＝1?
YES
0C未满
NO
1C以上
0?
YES
0C
NO
J12
A◂ 00H
B◂ A
A◂ 00H
J10
A◂ A＋01H
DAA
B◂ B－1
16进
◂10进
变换
NO
0?
YES
J11
(00E3H)◂ A
RET

图 8

表 7　程序清单

标记	助记符	地址	机器语言	注释
START	LD A,8H	0000	3E81	设置输入输出
	OUT(03H),A	0002	D3 03	
	LD SP,0100H	0004	31 00 01	设置 SP
	LD A,00H	0007	3E 00	设置初始值
	LD(00E3H)A	0009	32 E3 00	
	CALL TIMER	000C	CD 20 00	
MAIN	CALL DISP	000F	CD 30 00	
	CALL INPUT	0012	CD 40 00	
	CALL MERGE	0015	CD 70 00	
	CALL CHANGE	0018	CD 90 00	
	JP MAIN	001B	C3 0F 00	
TIMER	LD D,5CH	0020	16 5C	延时子程序
J40	LD E,5FH	0022	1E 5F	
J41	DEC D	0024	1D	
	JP NZ,J41	0025	C2 24 00	
	DEC D	0028	15	
	JP NZ,J40	0029	C2 22 00	
	RET	002C	C9	
DISP	LD A,(00E3H)	0030	3A E3 00	温度显示
	OUT(01H),A	0033	D3 01	子程序
	RET	0035	C9	
INPUT	LD A,FFH	0040	3E FF	
	OUT(02H)A	0042	D3 02	
	LD A,00H	0044	3E 00	
	OUT(02H)A	0046	D3 02	温度输入子程序
	CALL TIMER	0048	CD 20 00	
	IN A,(02H)	004B	DB 02	
	LD(00E0H),A	004D	32 E0 00	
	LD A,FFH	0050	3E FF	
	OUT(02H),A	0052	D3 02	
	IN A,(02H)	0054	DB 02	
	RLCA	0056	CB 07	
	RLCA	0058	CB 07	
	RLCA	005A	CB 07	
	RLCA	005C	CB 07	
	LD(00E1H),A	005E	32 E1 00	
	RET	0061	C9	
MERGE	LD A,(00E0H)	0070	3A E0 00	
	AND 0FH	0073	36 0F	
	LD B,A	0075	47	
	LD A,(00E1H)	0076	3A E1 00	
	AND F0H	0079	E6 F01	输入数据
	ADD A,B	007B	80	合成子程序
	CPL	007C	2F	
	LD(00E2H),A	007D	22 E2 00	
	RET	0080	C9	
CHANGE	LD A,(00E2H)	0090	3A E2 00	
	SUB A8H	0093	D6 A8	
	JP C,J12	0095	DA A9 00	
	JP Z,J11	0098	CA A5 00	
	LD B,A	009B	47	温度数据变换子程序
	LD A,00H	009C	3E 00	
J10	ADD 01H	009E	C6 01	
	DAA	00A0	27	
	DEC B	00A1	05	
J11	JP NZ,J10	00A2	C2 9E 00	
	LD(00E3H),A	00A5	32 E3 00	
J12	RET	00A8	C9	
J12	LD A,00H	00A9	3E 00	
	JP J11	00AB	C3 A5 00	

部件，必须认真理解 INPUT：温度输入子程序（图 6）、MERGE：输入数据合成子程序（图 7）、CHANGE：温度数据变换子程序（图 8）。以流程图为基础，编制程序清单。如表 7 所示。

5. 执行程序

首先，参照图 9、连接 μP－80 与 8 位 A/D 变换部件。然后，写入表 7 的程序，将 A/D 变换部件的切换开关扳在检验侧，旋转电位器，端口 B 显示应该由 00 变到 87（输入输出部件的显示切换开关在端口 A、端口 B 一侧）。

检验结束，A/D 变换部件的切换开关在测试侧，接通代替热敏电阻的 6kΩ 电阻，它相当于 20℃。此时，应该显示 20℃左右（电源电压与电阻平衡，可能有±2 度误差）。这时，要调整温度输入电路的串联电阻 RB 的数值，希望显示为 20℃，如果显示 20℃，连接热敏电阻。

本实验温度显示以 1 度为单位，如果使用运算放大器放大模拟电压，平均 1 度的电压变化量由 20mV 到 40mV 时，测试精度可达到 0.5 度。在模拟量输入端，如果连接热敏电阻以外的传感器，也能显示由该传感器决定的测试量。有兴趣的读者不妨动手实验。

首届电子版图书全国大联展即将开幕

为了贯彻中央十三届七中全会关于加速发展电子工业的方针，进一步落实中央关于“一手抓整顿、一手抓繁荣”的指示精神，繁荣文化图书市场，缓解“卖书难和买书难”的矛盾，电子工业出版社定于 1991 年 4 月 25 日至 5 月 10 日在北京、上海、天津、广州、南京、西安、武汉、沈阳、重庆、济南、杭州等 30 多个大中城市的 40 多家新华书店同时举办“电子工业出版社图书首届全国大联展”。

本届联展共将展销电子版图书十五大类，近千个品种。其中九〇年以来出版的新书及重印书 300 余种。这些图书基本上覆盖了电子科学技术的各个专业；兼顾了不同层次读者的需求，除了有大批实用性很强的工程技术图书和科普读物参展外，还有不少具有较高学术水平的著作和读者见面。在展销期间，电子工业出版社软件出版部将提供一百余种计算机辅助教学软件和近百种 PC 机应用软件供读者选购。此外还有若干音像出版物上市。

电子工业出版社软件部

举办软件优惠酬宾活动

为促进计算机软件的推广应用，软件部决定在我社图书首届全国大联展期间举办“软件优惠酬宾活动”，即在 1991 年 4 月 1 日～5 月 31 日期间，实行特优价销售软件：CEC—I 和 Apple—Ⅲ软件优惠价为目录定价的 75%，IBM PC 及其兼容机软件为目录定价的 85%。欲购者从速函索“软件优惠酬宾”卡和“软件目录”。联系地址：北京万寿路 173 信箱软件部　邮码：100036

联系人：谈众安　电话：815342

电脑巧开发

MP—1 微电脑加装编码键盘

李志平

MP—1 学习机结构简单，实用性强，经过开发能胜任复杂的工作。只是原机的数据键为 8 位拨动开关，输入程序时颇费心思，且容易出错。本人给 MP—1 机加装了一个简单的编码键盘，输入程序时甚为简便。

一、工作原理

电路见图一，共有十六个键，可以打出 0 到 F 十六个数据，当没有按下按键时 S_3，S_2，S_1，S_0 均为高电平，当一个键被按下时，即有一根 y 线和 x 线被短接，对应的三极管有导通电压而导通，于是被短接的 y 线通过 be 极接地，变为低电平，x 线也因三极管导通而变为低电压，也就是说每按下一个键，都导致和此键相连的 x 线和 y 线变为低电压，把 x 线和 y 线的电位变化通过二极管矩阵编码，即获得与被按键的键面数字相符的对应的二进制数码（高电平为 0），同时 x 电位变化也引起 D 点变化，D 为按键检测点，各按键按下时 S_3，S_2，S_1，S_0 D 输出电位及代码如表所列。

S_3，S_2，S_1，S_0 输出电位通过 IC_1，（施密特触发器）整形反相输入到 IC_3，IC_4 存储起来。IC_3 和 IC_4 为 4 位 D 锁存触发器，D 的电位通过整形作为 IC_5（JK 触发器）的时钟信号。每来一次 D 信号，JK 就翻转一次，JK 的 Q 和 $\bar{Q}$ 电位通过电容器，二极管变为正尖脉冲，以控制两组 D 触发器轮流触发，按下键时 S_3，S_2，S_1，S_0 的信号在两组 D 触发触器输入端等候，假定按键前 JK 的 Q=1，按下键时 $\bar{Q}$=1，高 4 位 D 触发器有时钟脉

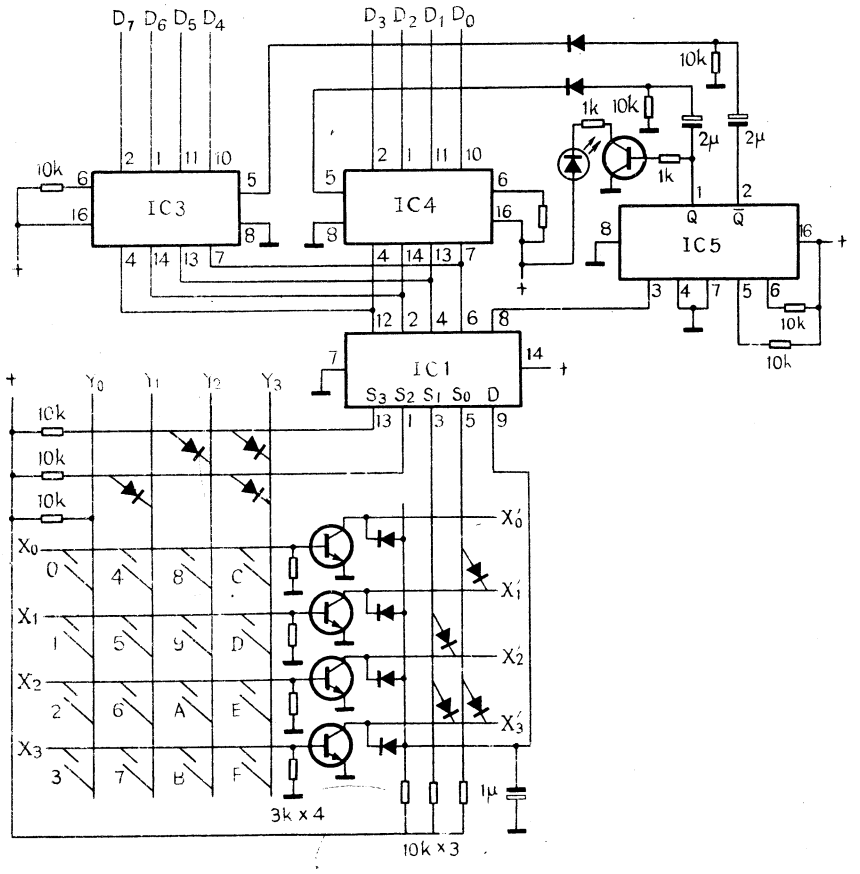


图 1

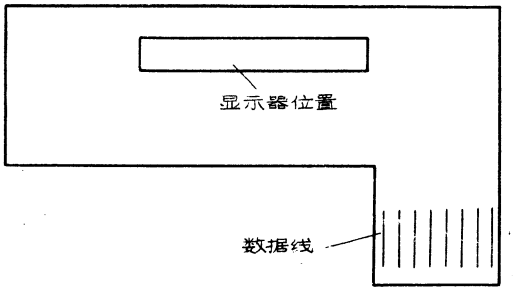


图 2

冲，被触发(即存入了 S_3,S_2,S_1,S_0。再按一次键时，低 4 位 D 触发器被触发，第三次按键时，高 4 位又工作，如此反复。在 JK 的 Q 端有一指示灯，灯亮时即可开始输入数据，第一次输入是高 4 位。

另外，多数程序输入时都按顺序从小到大输入的。在 A_1 键上并联一个三极管开关，由 WR 键控制输入程序就方便多了。输入程序时用 A_1 或 A_n 找到需要的入口地址，按数据键使指示灯亮，输入高 4 位，接着输入低 4 位，按 WR 键数据就输入内存单元，放开 WR 键，程序计数器加 1，再输入下一个数据。

二、安装与元件选择

电路板设计如图二形状，以便于安装在原机上，按键时宜选用微型微动开关，IC_1 为 CD40106，IC_4 和 IC_3 为 CD4042，IC_5 用 CD4027，三极管可用 3DG201 等廉

	S_3	S_2	S_1	S_0	D	代码
0	△	△	△	△	↓	0000
1	△	△	△	↓	↓	0001
2	△	△	↓	△	↓	0010
3	△	△	↓	↓	↓	0011
4	△	↓	△	△	↓	0100
5	△	↓	△	↓	↓	0101
6	△	↓	↓	△	↓	0110
7	△	↓	↓	↓	↓	0111
8	↓	△	△	△	↓	1000
9	↓	△	△	↓	↓	1001
A	↓	△	↓	△	↓	1010
B	↓	△	↓	↓	↓	1011
C	↓	↓	△	△	↓	1100
D	↓	↓	△	↓	↓	1101
E	↓	↓	↓	△	↓	1110
F	↓	↓	↓	↓	↓	1111

△ 代表高电平　↓ 代表低电平

图 3

价管子。安装时不能带电焊接 IC。最好能用 IC 插座安装。电路板上还可装一个 8 位插座，可把数据输出到 MP－1 机的输入口进行“现场”控制。

扫描式红外线十路遥控器

王刚勤

本遥控器是采用电子扫描而实现对音频译码器中心频率的捕捉，且只用一个锁相环音频译码器，就可分离出十种不相同的中心频率信号，从而大大减化了控制电路。该遥控电路设计新颖、实用性强，与其它遥控电路相比，具有体积小、重量轻、性能价格比高等优点。可对各种家用电器及其它需要多路控制的装置和器具进行遥控。

一、工作原理

图 1 和图 2 是扫描式红外线十路遥控器的接收电路。图 3 是扫描式红外线十路遥控器的发射电路。现把以上两种电路的工作情况分别叙述如下：

1. 发射电路　由 IC 等元件组成一个多谐振荡器，其振荡频率由 $1.443/(r+2R11)C_2$ 决定，r 为 R_1～R_{10} 其中某一个电阻。AN_1～AN_{10}为 10 个按键开关。倘若按下某一个按键遥控开关时，使得 IC 所组成的多谐振荡器工作，则振荡信号从③脚输出，故红外发射管 FG 发出一定频率的红外光。只要轮流按一下 AN_1～AN_{10} 开关，它便产生出十种不相同的红外线遥控发射频率。

2. 接收电路　主要由红外线接收探头、放大电路、音频译码器、脉冲产生器、十进制计数器/脉冲分配器、模拟数字传输开关、双稳态电路、与非门和非门电路、电子开关、复位电路、执行电路等组成。LM567 是一种锁相环(PLL)音频译码器，它具有高噪声抑制、窄通频带、抗干扰性好、中心频率稳定等特点。LM567 的中心频率由 $1/1.1RC_9$ 决定(R 为 R_{20}～R_{29}其中某一个电阻)；工作频率可在 0.01～500 千赫；工作电压 4.75～9 伏；第③脚输入端，要求输入信号＞25 毫伏；第⑧脚输出端，允许最大灌电流为 100 毫安；静态工作电流为 8 毫安。LM567 的基本工作过程如下：当在器件③脚输入＞25 毫伏的振荡信号，且输入信号频率与 LM567 的中心频率相同时，则⑧脚由高电平跳变到低电平。

平时，红外接收管 HG 不受到红外光作用时，其音频译码器 IC_4 的输出端⑧脚呈高电平，IC_1、IC_2、IC_5～IC_7 均处于工作状态，并且快速向 LM567 的 10 路遥控

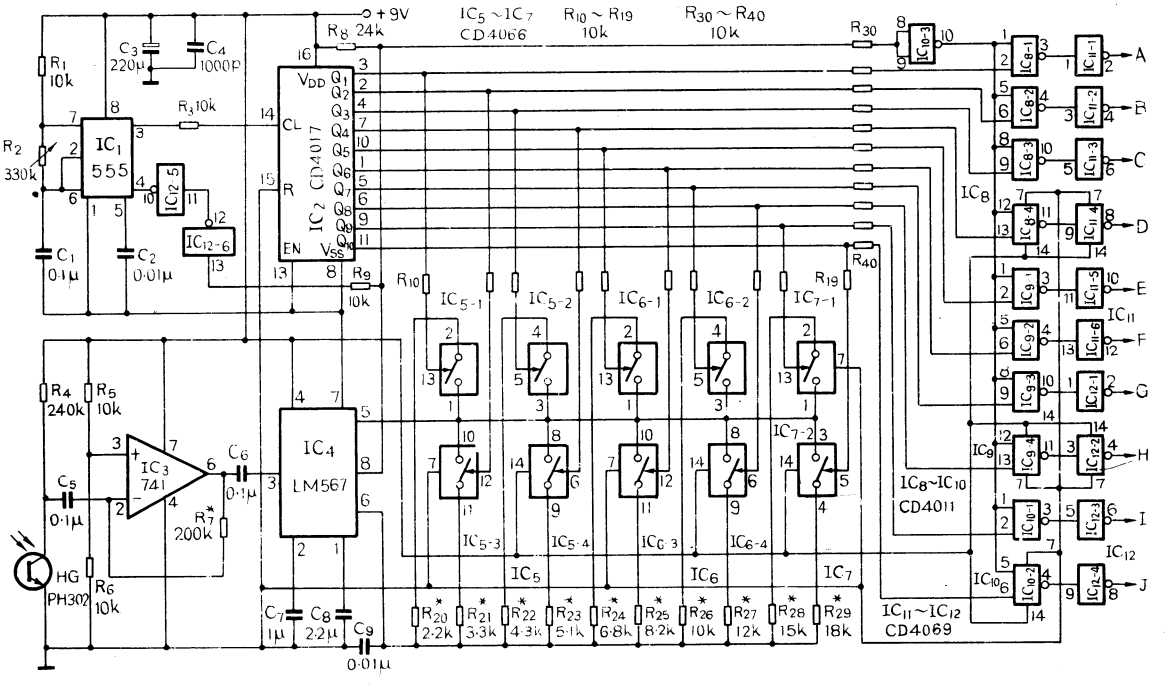


图 1

管 FG 的微弱红外光信号，经 C_5 耦合加到运算放大器 IC_3 的②脚上进行放大。被放大后的输入振荡信号从 IC_3 的⑥脚输出，由 C_6 耦合加至音频译码器 IC_4 的输入端③脚上去捕捉 IC_4 的中心频率。与此同时，IC_1 产生的脉冲振荡信号从③脚输出，其振荡频率由 $1.443/(R_1+2R_2)C_1$ 决定，频率 f 约为 20 赫左右。然后，由 R_3 加到十进制计数器/脉冲分配器 IC_2 的⑭脚上进行计数译码输出，使得 Q_1～Q_{10}轮流的输出正脉冲信号。由于 IC_1 的振荡周期 T 约 0．05 秒，故使 Q_1～Q_{10}的输出端分别在 0．5 秒时获得正脉冲信号一次。当 IC_2 的③脚输出正脉冲信号，使得 IC_{5-1}模拟开关接通，相应接通 IC_4 的⑤脚与⑥脚之间的电阻 R_{20}，随之 IC_4 产生的中心频率与接收到的发射频率相同，此时⑧脚由高电平跳变到低电平。然后，IC_{10-3}的⑩脚输出高电平，其 IC_{8-1}的两个输出端①和②脚均为高电平，③脚输出低

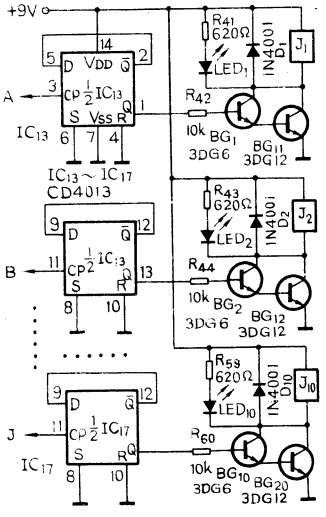


图 2

通道中心频率进行扫描。由于输入振荡信号未加到 LM567 上，其输出端⑧脚为高电平，则 IC_{10-3}的⑩脚输出低电平，使得 IC_{8-1}～IC_{8-4}、IC_{9-1}～IC_{9-4}、IC_{10-1}、IC_{10-2}上的一个输入端分别为低电平，故 A～J 端均输出低电平，致使继电器 J_1～J_{10}都为释放状态。

当按下 AN_1 时，红外接收管 HG 接收到红外发射

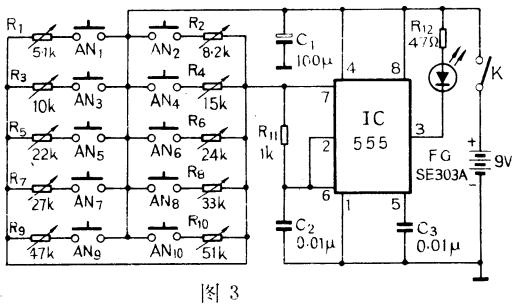


图 3

电平，使得 IC_{11-1}的②脚（A 端）输出高电平信号（由于 IC_{8-2}～IC_{8-4}、IC_{9-1}～IC_{9-4}、IC_{10-1}、IC_{10-2}上有一个输入端分别为低电平，故使 B～J 端均为低电平）。就在 IC_{11-1}的②脚输出高电平的瞬间，使得 IC_{13}的③脚获得正脉冲信号，致使 IC_{13}其中一个双稳态电路翻转，其①脚由低电平跳变到高电平，且 BG_1 和 BG_{11}均导通，J_1 通电吸合，从而使第一路的电器工作。当 IC_4 的⑧脚由高电平跳变到低电平的同时，其低电平信号经电阻 R_9 使 IC_{12-6}的⑫脚输出高电平，IC_{12-5}的⑩脚输出低电平，且 IC_1 自动复位，使得 IC_2 不再计数译码输出，而③脚保持在高电平状态。当 AN_1 一旦松开（接点被断开），IC_4 的⑧脚又恢复到高电平状态，但继电器 J_1 仍保持吸合状态。这时 IC_4 无输入信号，既使扫描电路继续工作亦是无效的，不能使其它的电器工作，除非按一下相应的按键开关。若不使电器工作，只要按一下相应的按键开关即可。同理，分别按一下 AN_2～AN_{10}开关，则能使相应的其它电器通电工作。其工作原理与上相同，这里不再赘述。

二、主要元器件选择

发射电路中，IC 使用 NE555 时基集成电路，亦可用 5G1555、FX555、LM555 等。红外发射管 FG 使用 SE303A，或 HG503 等。AN_1～AN_{10}均用小型按键开关。

接收电路中，IC_1 使用时基集成电路 NE555，其它的 555 型号亦可。IC_2 使用十进制计数器/脉冲分配器 CD4017 或 C187、MC4017 等，其外引出脚排列和功用见图 4。IC_3 为通用型运算放大器 μA741，亦可用 CF741、5G24 等。IC_4 使用锁相环音频译码器 LM567，

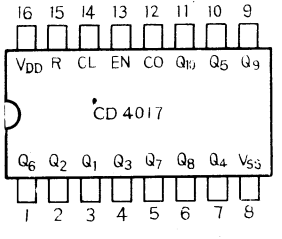


图 4

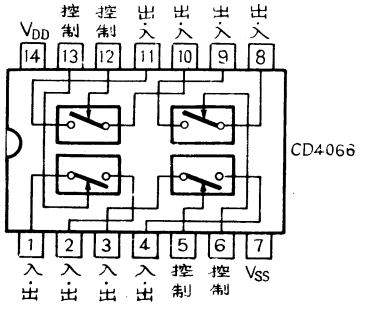


图 5

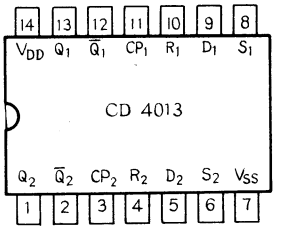


图 6

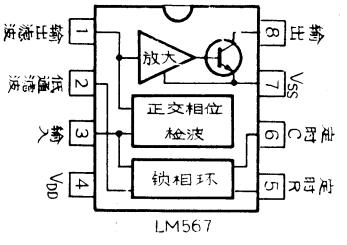


图 7

其外引出脚排列和功用见图 7。IC_5～IC_7 均使用四双向模拟数字传输开关 CD4066，亦可用 5G811、CH4066、C544 等，其外引出脚排列和功用见图 5。IC_8～IC_{10}均使用四与非门集成电路 CD4011，或用 5G801、CH4011 等。IC_{11}、IC_{12}均用六非门集成电路 CD4069，或 C033、5G806 等。IC_{13}～IC_{17}均选用双 D 触发器 CD4013 或 CH4013、5G822 等，其外引出脚排列和功用见图 6。BG_1～BG_{10}均选用放大系数大于 60 的硅 NPN 型晶体管，如 3DG6 等。BG_{11}～BG_{20}均选用放大系数大于 80 的硅 NPN 型晶体管，如 3DG12 等。HG 为红外线接收管 PH302，若此管没有可用一般的光敏三极管代替。J_1～J_{10}均使用直流工作电压 12 伏、线圈的直流电阻 450 欧、吸合电流≤20 毫安的 JQ×－4 小型灵敏继电器。

三、调试及使用

整机安装完毕，经检查无误方可通电调试。先不接用电器，把一支 620 欧的电阻与一支普通发光二极管串联后接在 IC_4 的⑧脚和电源正端之间，再把发射电路中的第一路遥控按键开关 AN_1 用导线短接，然后分别接通接收电路和发射电路的电源，将红外发射管 FG 对准红外接收管 HG 的受光面，缓慢的转动可调电阻 R_1，同时观察接在 IC_4 的⑧脚上的发光二极管，当该发光二极管刚好点亮时，则停止调整 R_1，此时第一路发射频率与 IC_4 相应的第一通道中心频率相同。最后，取下按键开关上的短接导线，第一路遥控通道就算调整好了。同理，用相同的方法可调整好 2～10 遥控通道，使其 IC_4 的 2～10 通道中心频率与发射电路的 2～10 通道发射频率相对应。发光二极管 LED_1～LED_{10}分别作 1～10 路电器的通电工作指示。倘若接收机 1～10 通道的某个中心频率偏移相对应的发射频率的话，可适当调整 R_{20}～R_{29}与之相对应的电阻的阻值，使其每一路遥控通道均能可靠的对电器进行遥控。

初级程序员水平考试辅导

微型计算机操作系统使用问答

王路敬

一台计算机安装起来后，怎样使这套系统正常地运转起来呢?计算机的各组成部分又是怎样相互配合、协调一致有效地进行工作呢？这一切都是通过操作系统去控制的。操作系统是计算机所有硬件资源和软件资源的组织者和管理者，是对计算机系统进行统一管理、分配和调度的系统软件，是由许多具有控制和管理功能的子程序组成。

任何一个用户都是通过操作系统使用计算机的。它是人机交互的接口，是所有软件编制的基础，所有应用软件的使用方法及原理都与操作系统有着许多内在的联系。因此，了解操作系统的一般工作原理以及有关的基本概念，基本组成及使用方法对操作人员和软件工作人员都是十分必要的。随着 IBM PC 机及其兼容机的广泛普及，PCDOS、CCDOS 操作系统已成为一种最流行的单用户操作系统。作为使用微型计算机的初级程序员，迅速正确的解决使用操作系统中碰到的一些问题显然是极为重要的。下面就将大家在使用 PCDOS 和 CCDOS 过程中常见的而又具有普遍性的问题解答如下。对报考初级程序员的考生进行复习提供一些帮助。

1. PCDOS 的发展过程分为哪几阶段？

到目前为止，DOS 的发展过程可以分为两个阶段：

(1)第一个阶段是指 DOS1.00 及 DOS1.10 版时期。这些早期的版本是由微软件公司(Microsoft)将所购置的 8 位 CP/M 操作系统予以改进而来。事实上，早期的 DOS 可以说是 8086/8088 CP/M 的扩充版。

(2)第二个阶段则始于 2.00 版的发表。这个版主要改革部分是将原来版本再加上结构上的扩充。主要增加的部分是提供 UNIX 的部分使用者界面。而其他结构上的重大改革是提供了硬盘驱动器、树形文件目录系统、输入/输出定向及管道功能、可增设的磁盘机、提供更方便操作系统与应用程序的文件输入/输出界面等。

PC-DOS2.10 版仅对系统内部作了修改，使能适用于 PC/JR 及 PC Portable(轻便型)的半高型软盘驱动器特性。没有强加任何新命令。长城 0520CH 就使用了该版本。

PC-DOS3.00 版本为 PC/AT 而发表的。它除了提供 PC/AT 的 1.2MB 的软盘驱动器功能外，外部较明显的改进是命令可以加上一个前列字串，用以描述命令所在的子目录。这一点改进在 3.00 及后续版本中体现得比较明显。在内部，文件分配表是(FAT)做了重大的改变，文件分配表的每一项以 16 位二进制数来表示，而不是以 12 位二进制数来表示。所以一个磁盘中最多可以有 65536 簇，而不是以前的 4096 簇。簇的数量增加意味着在同一大小的磁盘中，每一个簇可以有较小的空间，因此，磁盘空间的使用会更有效率。

PC-DOS3.10 实现了网络功能，它还提供了 SUBST 命令和 JOIN 命令。SUBST 命令可以将一个磁盘机字母代替任一路径。JOIN 命令，它可以把整个磁盘机处理成为其他装置的子目录。除此之外，PC-DOS3.10 改正了 PC-DOS3.00 的 BACKUP 命令。早期版本的 BACKUP 在处理到超过磁盘容量限制的文件时会失去它的磁盘计数，而新版的 BACKUP 则完全正常。

PC-DOS3.20 版是为轻便型 PC 机所发表的，现在已广泛的应用到长城 0520DH，长城 286 及其兼容机上。这种版本主要加强部分是提供对 3.5 英寸 720KB 软盘驱动器的驱动功能，以及 IBM 的 Token-Ring 的网络功能。也有一般性的改进，如提供 REPLACE、XCOPY 等，它们是 COPY 命令的补充和加强。这种版本的其他特点是提供一个称为 DRIVER·SYS 的设备驱动程序。该程序的主要功能是提供外部的 3.5 英寸磁盘机驱动功能，也可以处理其他的磁盘机。增加了一些命令，扩充了一项新的输入/输出控制。

PC-DOS3.30 除了可以用于现有的 PC/XT，PC/AT 及长城系列微机以外还可以用于 IBM PS/2 微型机。在 PC-DOS3.30 中引进了一些新的概念，如码页的概念；也配备了一些新的设备驱动程序和外部命令程序，如 DISPLAY·SYS、APPEND 等。

DOS4.00 是最新版本，它包括许多新命令，还包括许多较以前各版本有所增强的命令。它的新功能支持大容量存储介质(硬盘分区可以超过 32 兆字节)。提供了全屏幕交互式安装实用程序 SELECT 以及由鼠标器或键盘驱动的全屏幕用户接口程序 Shell，使你能建立用户定义的菜单系统。4.00 版加强了对命令语法的检查，出错信息，而且作了更详尽的区分和说明。DOS4.00 中较大的文件分配表将加速对文件的访问。

可见，DOS 的版本不断变化，其功能不断增强。综上所述，除 PC-DOS3.10 版是为了网络的需要，其他版本变更的主要原因是因为磁盘的升级。

2. PCDOS 的各个版本之间的兼容性是什么含义？

目前社会上使用了 PCDOS 的各种版本，较多的是 PCDOS3.××版。这些不同的 PCDOS 版本之间基本上是兼容的。所谓兼容是指低版本上能操作的命令，在高版本上同样可以操作。版本号越高，功能越强。选用

PCDOS 的版本要根据机型，低档次的微机不要追求高版本的 PCDOS。

3. 组成 PCDOS 的四个模块各有哪些主要功能？

PC-DOS 由引导程序、隐含文件 IBMBIO・COM 基本输入输出管理程序、隐含文件 IBMDOS・COM 中断 21H 的系统功能调用和用户文件管理程序以及命令处理程序 COMMAND・COM 四部分组成。它们各自的功能是：

(一)引导程序

系统加电后，执行 ROM 区地址在 FFFF:0000 的启动程序，进行初始化，并自检。然后进入 ROM BIOS 的 INT 19H 检查驱动器插盘否。若插有 DOS 磁盘，则读入引导程序，否则再进入 ROM BIOS。引导程序放在内存地址 0000:7C00。

其引导过程是：

(1)检查 0 道 5 扇区的目录块，应有 IBMBIO・COM 和 IBMDOS・COM

(2)将 IBMBIO・COM 和 IBMDOS・COM 两个文件放在地址 0060:0000 处。

(3)转到 0060:0000 处执行 IBMBIO・COM

(4)以上没有，则为非系统盘显示错误，等待换盘。

(二)IBMBIO・COM

该程序依赖机器硬件，扩充了 ROM BIOS 与 IBMDOS・COM 接口，处理同全体外设的通讯。它的功能是：

第一，初始化功能，包括：

(1)建立磁盘参数表(INT-1EH)入口；

(2)初始化 RS232 端口和打印机口；

(3)建立 01H,03H,04H,1BH 类中断入口；

(4)建立通信区 0050 段中的标志单元；

(5)把 IBMDOS 从当前位置移到 BFH 段，要复盖一些 BIOS・COM 程序；

(6)确定磁驱动器数和 RAM 的大小；

(7)调 IBMDOS 初始化程序，建立用户区段；

(8)填写 25H,29H 类中断入口地址；

(9)把 COMMAND・COM 读到用户区段 0100 处；

(10)建立磁盘缓冲区(DAT)地址，偏移量为 80H；

(11)执行 COMMAND・COM 程序。

第二，接口功能，有 11 个设备驱动程序：

05H, 08H, 0AH, 10H, 11H, 12H, 13H, 14H, 16H, 17H, 19HB

(三)IBMDOS・COM

该程序不依赖机器硬件，与 IBMBIO、COMMAND 和应用程序接口。它的功能是：

第一，初始化功能

(1)按照 IBMBIO・COM 的磁盘参数表 DPT，再建立新的磁盘参数表 DPT；

(2)每个驱动器建立一个文件分配表(FAT)；

(3)在最后一张文件分配表末尾建立用户区段地址；

(4)检查 RAM 的大小；

(5)填写中断向量 0H,20H,22H,23H,24H,25H,26H,30H 的入口；

(6)在程序段建立前缀控制块 PSP。

第二，中断 21H 功能调用

共有 12 组 57 个功能块，调用时 AH 放功能块号。

(四)COMMAND・COM

COMMAND・COM 由暂驻内存、常驻内存和初始化部分组成。

第一，暂驻部分

在 COMMAND・COM 建立段前缀 PSP 后，由 IBMBIO 的初始化程序将 COMMAND・COM 的常驻部分调入 CS:0100，由此开始执行 COMMAND・COM 的初始化。暂驻部分含：命令处理程序本身、所有的内部命令处理程序和批命令处理程序。对 DOS2.1 还包括装入外部命令程序，执行外部命令程序。

第二，COMMAND・COM 的内存常驻部分的功能是：

(1)与 IBMDOS 用户区程序，COMMAND・COM 的暂存部分通讯；

(2)建立 INT22H 管理程序终止，INT23H 中断退出，INT24H 出错处理，INT27H 结束并保留程序；

(3)在用户程序终止后，检查暂存部分被用户程序覆盖否，若覆盖了则重新调入。

第三，初始化部分。这部分包含 AUTOEXEC・BAT 文件处理程序的设置程序。它决定被 装入程序的段地址。

第四，COMMAND・COM 还产生 DOS 提示符(如 A＞或 C＞)，读来自键盘(或批命令文件)的命令并执行之。对于外部命令，它建立一个命令行并发出 EXEC 功能调用，以装入并把控制传送给这个外部命令处理程序。

4. 启动 PCDOS 的含义是什么？

组成 PCDOS 的文件都装于磁盘之中，只有将磁盘的文件按一定的规则装入内存，才能使 PCDOS 操作系统激活，从而把系统的硬件资源和软件资源管理起来。所谓 PCDOS 的启动就是把系统盘上基本文件：IBMBIO・COM 基本输入输出系统、IBMDOS・COM DOS 核心、COMMAND・COM 命令处理程序装入内存的过程。

启动 PCDOS 有两种方式：一种是冷启动，即加电开机的启动方式。第二种是热启动，这是系统主机已经加电，或你想摆脱某种困境而要回到 PCDOS 状态的启动方式。建议你最好从硬盘系统启动 PCDOS(或 CCDOS)，这样做操作简单，启动速度快，而且可减少计算机病毒感染的机会。

5. PCDOS 启动的过程是怎样进行的？

PCDOS 的启动过程如下：

(1)当系统加电或热启动后，程序从 FFFF:0000H 开始执行，进入自检程序和 ROM 引导装入程序。

(2)ROM 引导装入程序，从磁盘的第 1 扇区(引导扇区)读入磁盘引导装入程序到 0000:007CH，然后将控制权转交给磁盘引导装入程序。

(3)磁盘引导装入程序检查盘上有无 DOS 系统。检查过程为:读入根目录的第 1 扇区,然后检查其中前两个文件是否为 IBMBIO·COM 和 IBMDOS·COM,若盘上无这两个文件,系统将提示用户插入另一磁盘,然后按下任一键继续。若发现了这两个文件,磁盘引导装入程序便将 IBMBIO·COM 读入内存,并将控制转向 IBMBIO·COM 的初始化入口。

(4)IBMBIO·COM 由两个模块组成:第 1 个模块是 BIOS,它由一组设备驱动程序组成;第 2 个模块是 SYSINIT。

BIOS 模块的初始化部分将 IBMDOS·COM 加载到内存。

(5)SYSINIT 由 BIOS 的初始化代码调用。该程序确定连续的内存空间的大小,并将自身重新装入内存高端。然后,SYSINIT 将 DOS 核心(IBMDOS·COM)从初始化位置重新加载到最后位置,并将 IBMBIO·COM 中可覆盖的初始化程序和内存低端的 SYSINIT 覆盖掉。

(6)随后,SYSINIT 调用 IBMDOS·COM 的初始化程序。作为系统初始化工作的一部分,DOS 核心检查由驻留的块设备驱动程序返回的磁盘参量,确定系统所使用的最大扇区的大小,建立一些磁盘参数块(DPB)并设立磁盘缓冲区。

执行完 DOS 核心的初始化程序后,SYSINIT 便可调用 DOS 的文件服务程序(INT21H)去打开 CONFIG·SYS 文件,用户可在 CONFIG·SYS 中指定新加的设备驱动程序,确定磁盘缓冲区的数目,最多可同时打开的文件数目以及定义命令处理程序的文件名等。

(7)加载完所有可安装的设备驱动程序后,SYSINIT 关闭所有的文件指针,重新打开控制台(CON)、打印机(PRN)以及辅助设备(AUX)。此时允许用户安装的字符设备驱动程序覆盖 BIOS 为标准设备驻留的设备驱动程序。

最后,SYSINIT 调用 DOS 的 EXEC 功能将命令处理程序(COMMAND·COM)加载到内存,然后将控制转交给命令处理程序。

命令处理程序显示 DOS 提示符,等待用户输入命令。此时 DOS 进入正常工作状态,SYSINIT 消失。

6. 为什么有时热启动不起作用?此时应采取什么办法?

主要原因有两个:其一,突然改变了键盘中断处理程序的入口地址,使中断程序不能正确地被唤醒。由 PCDOS 中断向量表可知,地址 0000:0036～0000:0039 这 4 个字节是 PCDOS 的 9 类中断入口地址,它对应的是一个键盘中断处理程序。当由于某种原因改变了该类中断入口地址时,9 类中断程序不能正确执行,因而热启动也不能实现。其二,由于程序停留在关中断状态。当一个程序由于某种原因而突然停止运行时,如果程序正好停止在 CLI((关中断)指令处而使系统处于禁止中断状态,那么一切可屏蔽的中断都被挂起,9 类中断是一个可屏蔽中断,因此也不再起作用了。

当遇到上述两种情况时,要重新加电冷启动系统。

7. 在什么情况下需要启动 PCDOS(或 CCDOS)?

一般在下述情况下需要启动 DOS?

(1)系统从休息状态转入到工作状态。

(2)因软件故障或操作不当而使系统无法继续运行,出现系统"死锁"(或称"死机")。

(3)当改变系统结构设备而必须对系统结构设备文件 CONFIG·SYS 进行修改或重新建立时。

8. 在 PCDOS(或 CCDOS)下当前盘与启动盘有什么关系?系统启动后能改变启动盘吗?系统在什么情况下才会用到启动盘?

操作系统启动后,在未改变当前行定义的情况下,当前盘和启动盘是指同一个磁盘驱动器。以后当前盘可以任意改变,而启动盘不可改变。系统在"COMMAND·COM"的暂驻部分被覆盖时(即软件的数量过大),才会用到启动盘,以便把覆盖部分的代码读入内存。

例如长城 0520 或 IBM PC/XT 的启动盘是 A 或 C,当前盘可为 A、C 也可以是 B。

9. 长城 0520CH 系统启动后,为什么有时显示的日期和时间为标准时间,而有时则不是?

长城 0520CH 主机内有一个可在长时间切断电源后仍能保持准确走时的时钟,称之实时时钟。只有使用了外部命令 REALTIME 后,才能将实时时钟的内容传送到系统时钟,从而得到标准时间。系统时钟是操作系统及各种应用软件所使用的时钟。如果每次启动后希望使用正确时钟,就应使用 REALTIME 命令进行传送,或在自动执行批处理文件中增加 REALTIME/C 这一命令。

10. 什么是批处理文件?文件的扩展名为·BAT 的 ASCII 码文件称之为批处理文件。

对于经常要执行的一系列命令,可以把这些命令语句象编程序一样写入到一个批处理文件中。然后此文件可以作为外部命令被调用,这时系统将自动地依次执行该文件中的全部命令语句。

对于已存在磁盘上的批处理文件,可以用下列方式调用:

>批处理文件名↙

这时,系统将自动地依次执行当前盘上的命令语句。系统在执行批处理文件时,并不是把所含的命令文件全部装入内存而是读入一条并执行后再读入下一条执行,这样一直到最后一条命令语句执行完为止。

批处理文件可用 COPY CON:〈文件名〉·BAT 或用 EDLIN〈文件名〉·BAT 命令来建立。

例如,我们希望系统连续执行如下操作:

先把 A 盘的所有·BAS 文件复制到 B 盘,然后显示 A 盘的所有·BAS 文件目录,再显示 B 盘的所有·BAS 文件目录,最后打印出批处理文件自身的内容。

建立如下的批处理文件可达到上述要求:

```
A>COPY CON:WW·BAT
    COPY A:*·BAS B:
```

```
DIR A:*.BAS
DIR B:*.BAS
TYPE WW.BAT>PRN
^Z
```

^Z 即〈CTRL〉+Z 结束 COPY CON:命令并将 WW.BAT 文件存盘。按下述方式便可执行 WW.BAT 文件：

```
A>WW
```

11. 在批处理文件中可以使用哪些形式参数？为什么要使用？

PCDOS 系统规定，用户可使用 10 个形式参数。它们是%0，%1，%2，……%9。其中%0 表示批处理文件自身。值得注意的是"%"与数字之间不能有空格。调用时如果实际参数个数少于形式参数，则会显示出"Syntaxerror"的出错信息；如果实际参数个数多于形式参数，系统将不理睬多余的实际参数。

在批处理文件中使用形式参数目的是为了增加批处理文件的灵活性。

例如批处理文件 BATN.BAT 内容如下：

```
A>TYPE BATN.BAT
    DIR A:
    DISKCOPY A:B:
    DISKCOMP A:B:
    DIR B:
```

在该批处理文件内的盘符"A:"和"B:"用两个形式参数来代替，即：

```
A>COPY CON:BATN.BAT
    TYPE %0.BAT    ;显示该批处理文件内的
                    全部内容
    DIR %1    ;显示由第一个参数所定义的磁盘
              的目录
    DISKCOPY %1 %2
    DISKCOMP %1 %2
    DIR %2        ;显示由第二个参数所定义的磁
                   盘目录
    ^Z
```

在批处理文件中不引入形式参数，由于各条命令的参数都是固定的，故每次运行批处理文件只能执行固定的操作。引入形式参数后，就可以在调用批处理文件时再决定命令中的参数，给批处理文件赋予了一定的灵活性。

12. 在批处理文件中可包含哪些内容？

可以包含 PCDOS 内部命令、外部命令、可执行的程序、形式参数以及专用于批处理文件的子命令。这些子命令是专用于批作业的内部命令和语句的标号，以实现稍微复杂一些的作业结构。

这些子命令主要有：

(1)ECHO—禁止或允许屏幕的命令显示语；

命令格式：

```
ECHO
```

或：ECHO ON/OFF

或：ECHO 〈显示信息〉

(2)REM—注释语句

命令格式：

```
REM 〈注释语〉
```

(3)PAUSE—暂停语句

命令格式：

```
PAUSE 〔〈注释语〉〕
```

(4)GOTO—转向语句

命令格式：

```
GOTO 〈语句标号〉
```

(5)IF—条件判断语句

命令格式：

```
IF 〔NOT〕〈条件〉〈DOS 命令〉
```

(6)FOR—重复执行语句

命令格式：

```
FOR %%〈变量名〉IN〈文件名集〉DO〈DOS 命令〉
```

(7)SHIFT—左移参数语句

命令格式：

```
SHIFT
```

上述 7 个批处理子命令虽然也可以在操作系统的提示符下作为命令输入，但这样执行不是没有实际意义，就是不引起真正的执行。一般与 PCDOS 的命令，可执行程序结合起来组成一个批处理作业，使该处理文件更具有人机交互功能和友好性。

例如 CCDOS2.13G 中文操作系统的自动执行批处理文件 AUTOEXEC.BAT 的内容如下：

```
ECHO OFF
CLS
CD\213
MENUFF
ECHO                PLEASE WAIT
IF ERRORLEVEL 52 GOTO E
IF ERRORLEVEL 51 GOTO C
IF ERRORLEVEL 50 GOTO B
IF ERRORLEVEL 49 GOTO A
:E
FILE1 2
GOTO D
:A
FILE0 82
GOTO D
:C
IF NOT EXIST D:HZK16 COPY HZK16 D:
FILE3 D2
:D
CCCC
CC11
PRT
FILE16B
FILE24 1SFHK
FILE40 1FHK
FILE49 1
```

```
ZF24 3
YX1
;B
PATH C:\;C:\213
KEY
CD\
```

13. 为什么说 AUTOEXEC·BAT 是 CCDOS 启动的必备文件?

CCDOS 是在 PCDOS 的基础上开发而成的中文操作系统。它既保留了 PCDOS 原有的全部功能,又增加了有关汉字方面的处理能力。作为一个汉字操作系统,必须在系统启动后调用一些处理汉字的系统软件实现汉字的输入、显示和打印。如果不使用 AUTOEXEC·BAT 文件,就要在每次系统启动后手工调用,这样会使不了解情况的用户无法进入中文操作系统,建立这个文件后,每当用户启动 PCDOS 时,命令处理程序 COMMAND·COM 就在启动盘的根目录中搜索 AUTOEXEC·BAT,并在找到之后立即执行,从而使系统自动处在中文操作系统下。

例如 IBM PC/XT 机使用的 CCDOS2.0/2.10,在该系统下 AUTOEXEC·BAT 一般由下列基本语句组成:

```
ECHO OFF
CLS
FILE1
CCCC
ALL24P
```

其中 FILE1,CCCC 分别为 CCDOS2.0/2.1 的第 1 个和第 2 个启动文件是必不少的。ALL24P 是 3070 打印机打印汉字的驱动程序,其他型号的打印机可选取相适应打印驱动程序。

AUTOEXEC·BAT 是文本型文件可由用户建立和修改。在使用中需要注意的是当 AUTOEXEC·BAT 的内容一旦进行修改,需要重新启动系统。

14. DOS 为用户提供哪些实用功能?

DOS 为用户提供了大量的实用功能,包括设置一组中断和各种功能调用,其中有键盘输入程序,终端和打印机输出程序,形成文件控制块程序,内存管理程序,日期和时间程序以及各种磁盘、目录、文件的控制管理程序。作为文件管理功能,DOS 提供了两种类型的功能调用,它们是:

(1)文件控制块(FCB)功能调用。

(2)扩充控制功能调用。

15. DOS 在磁盘上分配分哪些区域?它们的含义是什么?

由 DOS 格式化的软盘或者磁盘,其扇区的大小均为 512 字节。DOS 将磁盘划为如下几个区:

(1)引导记录区,占 1 个扇区。它位于软盘的 0 面,0 磁道,1 扇区。如果试图用驱动器 A 中的非系统盘启动系统,它被用来产生出错误信息。对硬盘来说,引导记录位于 DOS 部分和第 1 个扇区。

(2)文件分配表 FAT 区,可变长。

FAT 用来为文件分配空间,每次一组。对磁盘上的每一个 FAT 由 12 位(1.5 字节)的表项或者 16 位(2 个字节)的表项组成。凡是少于 20740 个扇区的软盘或硬盘都用 12 位 FAT,多于 20740 个扇区的,用 16 位 FAT。FAT 总是位于引导记录接着的扇区。如果 FAT 大于一个扇区,则占用下面连续的扇区号。

(3)备份文件分配表 FAT 区,可变长。

(4)根目录区,可变长。

紧挨着文件分配表 FAT 存放。

FORMAT 命令,在磁盘上建立根目录。不同种类的软盘片,根目录的个数不同。

(5)数据区

文件的空间分配(在数据区)只有在需要的时候才做。每次分配一个簇。一个簇总是一个或多个连续的扇区号,一个文件所有的簇在 FAT 中链在一起。数据区排在备份文件分配表区之后。

16. DOS 怎样使用文件目录表和文件分配表 FAT?

DOS 在分配磁盘空间时是以簇为最小单位进行的,而不是以扇区为单位。对于不同形式的磁盘,每簇所拥有的扇区数是不同的。对一个长度仅为一个字节的文件来说,在双面软盘上要占用 1024 字节的空间,而在硬盘上就要占用 4096 字节的空间。读文件时,DOS 首先在目录中扛到该文件的目录项,根据目录项中第 25～27 字节的起始簇号值,求出文件的起始相对扇区号和在 FAT 的入口。起如扇区号与起始簇号之间的关如下(十进制):

磁盘形式	起始扇区号 S 与起始簇号 C
单面 8 扇区	S=C+5
双面 8 扇区	S=2C+6
单面 9 扇区	S=C+7
双面 9 扇区	S=2C+8
10M 硬盘	S=8C+33

FAT 表中的入口地址即为字节的顺序号(从 0 开始编号)。计算方法:

(1)将簇号乘以 1.5

(2)若所得结果为一整数,比如说是 6,则 FAT 中的第 6 个字节的 8 位二进制数作为 b7～b0 位,将相邻的下一个字节的低 4 位作为 b11～b8,组成一个 12 位二进制数,这个 12 位的二进制数即表示对应簇的状态。若所得结果是一小数,比如是 7.5,则 FAT 中的第 7 个字节的高 4 位作为 b3～b0 位。将下邻的一个字节作为 b11～b4,组成一个 12 位的二进制数。

写文件时过程正好相反,DOS 首先在目录中检索是否有同名文件,若无则将该文件名写入第一个找到的作过删除标记的目录中,否则开辟一新的未用目录项。然后,顺序检索 FAT 中的每个表项。找到第一个值为 000H 的表项后,将文件写入该表项对应的簇中,并将其簇号写入一个表项中。如果文件写完,则该表项位置为 FFFH。

一个文件可以是放在互相衔接的几个连续簇中，也可能是断续放在几个不相邻的簇中，中间隔着其他文件占用的簇或空簇。但不论哪种情形，一个文件所占用的簇，其簇号一定是从小到大顺序递增的，除非该文件不是采用正常的拷贝命令和编辑手段建立的。当用COPY命令将一些文件拷贝到一张空盘上时，所有的文件一定是放在互相衔接的连续簇中的。一旦对文件进行了多次增删和修改后，这种连续的情形就不存在了。有时在某个文件中间还会出现释放的空簇，这些情况都给恢复文件带来极大困难，甚至使恢复成为不可能的。

17. 文件分配表FAT向用户提供哪些信息？

FAT是磁盘上一个相当重要的区域，向用户提供以下4个方面的信息：

(1)磁盘性质；

(2)扇区封锁信息表；

(3)扇区使用情况；

(4)指示一个文件的后续簇。

FAT中每一个表项的值由一个12位或16位二进制数组成。第1～2字节总是FFH，第0字节(首字节)表示磁盘的性质，其意义如下：

FEH—单面每磁道8扇区软盘。

FFH—双面每磁道8扇区软盘。

FCH—单面每磁道9扇区软盘。

FDH—双面每磁道9扇区软盘。

F8H—硬盘

从第3字节开始的第2表项表示磁盘上数据区各簇的情况。数据区一定是从002H簇开始的。第3～5字节为第2～3表项，表示磁盘上002H簇和003簇的情况。表项号，簇号，磁盘上的簇域是一一对应的。这里要注意表项的值在字节中的位置。例如第4字节一半属于第2表项，而另一半属于第3表项。具体分配时其高4位应属于第3表项的低4位，而其低4位则是第2表项的高4位。

磁盘在格式化时如发现损坏的扇区便在对应的扇区表项中写入FF7H，表示该扇区所在簇不能使用。当表项的值为000H时表示对应的簇是空的，可以使用。FF8H～FFFH表示该簇为文件中最后一簇。FF0H～FF6H表示保留的簇，其他在表项中出现的指示文件中的一个簇簇号，也是该文件在FAT中的下一个入口。

18. 文件目录表向用户能提供哪些信息？

文件目录表是存放文件目录的地方，每个文件的目录由32个字节组成，包含了有关该文件的重要信息，各字节的意义如下：

(1)第0～7字节：

文件名，当第0字节的值为：

00H——表示该目录项从未使用过。DOS在有关的操作中遇到00H后即停止向下继续检索。

E5H——表示该文件已被删除，DOS跳过该目录项继续向下检索。

2EH——表示目录表为一子目录表。

除此之外出现的其他值为文件名第一个字符的ASCII代码。

(2)第8～10字节：

文件的扩展名。

(3)第11字节：

表示文件的属性，其含意如下：

01H—表示只读文件，拒绝COPY、ERAES命令。

02H—表示隐含文件，拒绝COPY、ERASE、DIR命令。

04H—表示系统文件，拒绝COPY、ERASE、DIR命令。

08H—表示该目录项前11个字符为磁盘卷标，在FAT中没有入口，只有存在于根目录中。

10H—表示该文件名为一子目录的父名。

20H—表示档案位。在建立或修改文件时，该值加到文件属性中表示关闭，对硬盘上文件使用BACKUP命令后，该位被打开，从文件属性中减去该值。归档位主要和BACKUP命令中的M开关及RESTOR命令中的P开关有关。

(4)第12～21字节：

保留未用。

(5)第22～23字节：

文件生成或最后一次更新的时间。

(6)第24～25字节：

文件生成或最后一次更新的日期。

(7)第26～27字节：

文件的起始簇号，也是指向FAT的入口地址。第26字节是低位字节，第27字节是高位字节。当第0字节为2EH时表示子目录所在的簇号。当第0字节与第1字节均为2EH时，表示该子目录表的父目录所在的簇号。

(8)第28～31字节：

表示文件长度，以字节为单位。第28字节为低位字节，第31位字节为高位字节。

19. CCDOS系统盘上的DONFIG·SYS文件有何用途？

CONFIG·SYS为系统结构设置文件。在CCDOS启动过程中，CCDOS自动在系统盘中检索这个文件。如该文件存在，则将该文件中各语句的内容对系统进行一些初始化设备，如果不存在，CCDOS就按约定值进行初始化设备。

以CCDOS 2.0/2.10为例，CONFIG·SYS文件包含如下7条系统设备命令完成以下功能：

(1)BREAK=ON\OFF(约定状态:OFF)

该语句用于对中止键〈CTRL〉+〈BREAK〉进行开(ON)关(OFF)设置。

当BREAK=ON时，无论系统在执行什么操作，都可以用〈CTRL〉+〈BREAK〉键中止系统的操作。

当BREAK=OFF时，只有当系统正在执行外设操作(屏幕、键盘、打印机和异步通讯适配器操作)时，才能用〈CTRL〉+〈BREAK〉键中止系统的操作。当系统执

行其它操作时，〈CTRL〉＋〈BREAK〉键无效。

也可以用内部命令 BREAK 设置〈CTRL〉＋〈BREAK〉的状态。但需要注意的是，在某些应用软件中（例如 dBASE Ⅲ），〈CTRL〉＋〈BREAK〉键是无效的。因此，BREAK 的设置对这类软件是无效的。

(2)BUFFERS＝n

其中 n 可在 1～99 之间选择，约定值为 2，表示设值缓冲区的数量。

缓冲区是磁盘文件管理的重要设置。当 DOS 读磁盘时，先将磁盘上的内容读入缓冲区中，然后再对其中的信息进行处理。在 DOS 进行反复读盘操作时，它先去缓冲区中检查要读的内容是否已在缓冲区中，若在，则勿须再执行读盘操作，从而节省了大量的读盘时间。

当 DOS 写磁盘时，要在缓冲区中存放写的信息，DOS 将缓冲区中的数据组织好后将其写入磁盘。

每个缓冲区要占用 528 个字节的内存。在某些应用中，多几个缓冲区可以提高程序的执行速度，但要以多开销内存为代价。并不是说缓冲区越多，速度就一定会更快。对于要求执行大量随机磁盘存取的应用程序（例如基本的数据库应用程序）缓冲区应多一些（例如在 10～20 之间可以取得较好的效益）；而对于执行大量顺序磁盘存取的应用程序（例如对整个文件读写），系统默认的两个缓冲区是足够的了，没有必要再去设置 BUFFERS 的数量。

(3)FILES＝n

FILES 用于规定 DOS 同时可打开的文件的数量，其中 n 可在 8～99 之间选择。系统的默认值是 8。但这 8 个 FILES 并不是都交给用户使用，用户只能使用其中的 3 个，另外 5 个 DOS 早已分配给标准输入、标准输出、标准错误输出、辅助设备和标准打印机了。用户如果需要同时打开多个文件，必须对 FILES 重新设置。

FILES 也是 DOS 进行文件管理的依据。DOS 每打开一个文件，就要在内存中为该文件建立一个文件控制块（简称 FCB），FILES 实质上规定了 DOS 所能建立的 FCB 的个数。

每个 FCB 在内存中要占用 39 个字节。一区 39 个字节对几百 K 的内存来说是无足轻重的，却赢得了可打开多个文件的方便，所以用户多设置几个 FILES 是明智的作法。

(4)DEVICE＝〈文件名全称〉

DOS 规定了 5 种标准设备文件，它们是：标准输入设备；标准输出设备；出错信息输出设备；异步通讯控制器及打印机。如果用户想要扩充系统的设备，就要用 DEVICE 语句来定义扩充设备的驱动程序文件的名字。这样，在 DOS 启动时，系统就能把这个文件装入内存作为 DOS 基本设备的扩充。例如，在 0520CH 上使用的 GWBIOS 3.00 在支持高级语言作图功能时，要把 DEVICE 设置为：

DEVICE＝GRD・SYS

其中 GRD・SYS 是一个系统图形驱动文件，该文件作为系统的虚拟图形设备，当系统启动时，将自动把 GRD・SYS 作为系统设备驱动文件装入内存。这样，用户在使用高级语言中的打开（OPEN）设备文件，写（WRITE）数据到该设备等语句时就可以进行画图操作了。

在一个 CONFIG・SYS 文件中可以使用多条 DEVICE 语句来定义多个扩充设备。DEVICE 通常是按应用软件的要求而设置的。

(5)SHELL＝〈文件名全称〉

此语句可以用用户自己开发的控制台命令处理程序代替系统所规定使用的 COMMAND・COM 文件。用户在使用这条语句时一定要谨慎从事，除非确有必要使用另行开发的控制命令处理程序，一般是不使用此语句的。

(6)AVAILDEV＝F

AVAILDEV 命令用于关闭字符设备名检查，其缺省值是 T(开)。当字符设备名检查开关被打开(T)时，每当进行指针型文件操作时，要检查文件名是否为字符设备名，否则(F)，则不对其进行检查。

(7)SWITCHAR＝〈开关引导符〉

SWITCHAR 命令用于设置命令行参数开关引导符，其缺省值是“/”。开关引导符是 COMMAND・COM 所特需的。

综上所述，CONFIG・SYS 文件是十分有用的。特别是在某些应用领域，例如 dBASE Ⅲ，它要求 FILES 和 BUFFERS 必须大于约定值，如若不设置就不能正常使用 dBASE Ⅲ。而 CONFIG・SYS 是在 DOS 启动时使用的，用户容易忽略，所以要提醒用户注意。

20. PCDOS 的内部命令和外部命令是按什么划分的？使用外部命令时注意什么问题？

DOS 的命令有两类：一类是 DOS 的内部命令，另一类是 DOS 的外部命令。内部命令与外部命令是按此命令是否常驻内存中来划分的。DOS 任何命令的执行即执行一段程序，DOS 的内部命令的程序已随 DOS 的启动而装入内存，其命令动词为系统保留词，如 DIR，COPY，TYPE，ERASE 等等均为内部命令。其外部命令的程序存于命令文件中，调用时装入内存，执行完一般所占用的内存释放，其命令动词为命令文件的文件名。使用外部命令时应注意：

(1)外部命令的执行依赖于命令文件的存在，即执行任一个外部命令，系统首先要从磁盘读取相应的文件，如此命令不存在，则命令不可执行，因此在当前盘上应当存在要执行的命令文件。

(2)在调用外部命令时，命令文件的扩展名是不必给出的。系统约定的扩展名为：

・COM

・EXE

・BAT

三种类型的文件的文件名 DOS 可直接调用。

(3)当调用外部命令时，若系统显示“Bad command or file name”错误信息，这时可以从两个方面检查：一是命令拼错了没有；一是当前盘是否存在该命令文件。

21. ANSI・SYS 程序有何功能？怎样设置？

PCDOS 装载的标准设备驱动器支持标准输入、标准输出、标准打印机、软盘以及硬盘设备。PCDOS 支持这些设备不需要用户用任何命令指定，如果要扩充输入设备、输出设备的控制特性，就需要把编写的有关设备的驱动程序，在启动 PCDOS 时把它装入，而且是通过在系统结构设置文件 CONFIG・SYS 中加入一个 DEVICE＝〈命令〉这种形式指定。其中 DEVICE＝ANSI・SYS 就是一种指定形式。

ANSI・SYS 是 PCDOS 提供的一个系统文件，在此文件的控制下可以实现屏幕和键盘的功能扩充，使 PCDOS 用扩展功能取代标准输入和标准输出。中文操作系统 CCDOS 2.0/2.1 是在 PCDOS 2.0/2.1 基础上扩充汉字输入和输出功能后形成的，键盘和屏幕输入和输出功能的扩充就是利用了 ANSI・SYS 的控制特性。如果在 CONFIG・SYS 文件中不加入 DEVICE＝ANSI・SYS 语句，IBM-PC/XT，长城 0520A 及其兼容机，使用 CCDOS2.0/2.1，在显示 11 行汉字状态下，PCDOS 的清屏命令 CLS 执行后光标不显示，屏幕上什么信息也没有，给使用带来不便。相反如果在 CONFIG・SYS 文件中加入语句 DEVICE＝ANSI・SYS 就很好地解决了清屏命令执行后屏幕无光标显示的毛病。

22. IBMBIO・COM、IBMDOS・COM 和 COMMAND・COM 版本不同时，会发生什么现象？怎样解决？

随着 DOS 功能的不断增加，其操作系统的大小也不断增加。在不同版本下，随版本号提高，其文件的长度也明显加大。对于 PCDOS 的基本文件：IBMBIO・COM、IBMDOS・COM、COMMAND・COM 的长度，以及它们在磁盘上占用的空间，系统程序占用区，暂态区，所占内存空间的情况在不同版本下其文件长度不同。

当 DOS 中两个隐含文件 IBMBIO・COM、IBMDOS・COM 和外部文件 COMMAND・COM 不是同一版本时，往往出现引导失败，或屏幕提示 DOS 版本不对，或自举后不能进入硬盘，运行应用软件出错等现象。产生这种问题的原因绝大多数是由于软盘之间，或者软盘与硬盘之间全盘拷贝造成的。例如，把整个软盘拷贝到硬盘，这时若盘上有 COMMAND・COM 文件也将它拷贝到了硬盘，而与它同版本的两个隐含文件 IBMBIO・COM 和 IBMDOS・COM 却没有拷贝到硬盘，造成上述现象。

每当出现这些现象时，不要急忙使用 FORMAT 命令格式化硬盘，或用检测盘测试微机，更不要把微机送出去修理。否则，既浪费人力、物力、财力、机时，又容易丢失大量数据。

在应用中可以采取一定措施来防止这种问题的发生。例如，所有应用程序只能拷贝到各自子目录下，少用或不用全盘拷贝命令。如果万一发生这种问题，可以针对不同情况，采取不同的解决办法。例如，当能自举时，可以将自举后屏幕显示的 COMMAND 中的版本号与用命令 VER 显示的两个隐含文件中的版本号相比较，看是否匹配。因为 DOS 的版本不同 IBMBIO・COM、IBMDOS・COM 和 COMMAND、COM 文件长度不同，通过命令文件的大小可以确定版本号是否匹配。当不能自举或屏幕提示 DOS 版本不对时，可以用比较高版本 DOS 从软盘自举，然后查看产生问题的原因。

23. 使用 FORMAT 命令时应注意什么问题？

磁盘格式化命令 FORMAT 是 DOS 的一个重要外部命令，执行该命令时它删除磁盘上所有的数据。因此在决定对一个磁盘，尤其是硬盘进行格式化时应当格外小心，在格式化之前需要备份又能够备份的文件最好备份下来，以便格式化后恢复。如果不能肯定哪个驱动器是默认的，在没有驱动器字母标号的情况下，就不要输入 FORMAT，否则就有可能把默认盘上的所有数据删除，造成损失。为保险起见，在 FORMAT 命令行中明确指出要格式化的驱动器的字母标号。

24. 为什么 FORMAT 命令不能消除硬盘上的大麻病毒？

因为大麻病毒存放在硬盘的主引导扇区，不属于 PCDOS 分区，所以与机器使用的 DOS 无关，故对硬盘格式化不能改变主引导扇区，寄生在主引导扇区的大麻病毒不能被消除。

利用下面两种方法可以人工消除硬盘感染的大麻病毒。

方法之一：

对硬盘进行初级格式化。如果系统带有初级格式化程序，可以在操作系统下直接执行；若没有初级格式化程序可用 DEBUG 程序调用 ROMBIOS 中的初始化程序并执行之。缺点是破坏硬盘的全部数据。

操作如下：

```
A>DEBUG
-G=C800:0005
```

该操完成后，即把硬盘主引导扇区的大麻病毒清除。然后进行硬盘分区，建立硬盘系统。

方法之二：

利用 DEBUG 程序进行如下操作：

```
A>DUBUG
-R IP                ;        设置 IP 指针
:100
-A 100               ;        开始汇编
××××:0100   MOV  DX,0080
             MOV  CX,0007
             MOV  BX,0200
             MOV  AX,0201
             INT  13H        ;读硬盘 0 柱面 0 磁头
                              7 扇区到内存 200 处读
                              原硬盘主引导程序
             MOV  DX,0080
             MOV  CX,0001
             MOV  BX,0200
             MOV  AX,0301
××××:011A   INT  13H        ;写硬盘 0 柱面 0 磁头
                              1 扇区恢复硬盘主
                              引导程序
```

```
-G=100 11C                    ; 执行上面程序即可
                                完成消除工作
```

25. FORMAT 和 SYS 两个命令在使用中有何区别?

FORMAT 命令选用 S 参数和 SYS 命令都可以复制系统文件,但在使用中有区别表现在:

(1)若想在已保存有数据的盘上再复制系统文件,应该用 SYS 命令,因为格式化时会删除磁盘上原有数据。

(2)使用 FORMAT /S 可以按如下顺序将 IBMBIO·COM、IBMDOS·COM 和 COMMAND·COM 复制系统文件,而使用 SYS 命令只能复制前两个隐含的系统文件,不能传送 COMMAND·COM。

(3)使用 SYS 命令传送系统文件,要求格式化时最好选用/B 参数,这样可以存入任一 DOS 版本的系统文件,否则只能存入指定的版本,还有一点要注意的是若格式化时选用了/V 参数,则不能用 SYS 命令传送系统文件,此时目标盘上没有为 DOS 系统保留存储空间。

26. 使用 BACKUP 和 RESTORE 命令应当注意什么?

BACKUP 和 RESTORE 这两个命令必须配合使用。具体来说,用 BACKUP 命令把硬盘根目录下的数据文件备份到软盘,用 RESTORE 命令还原备份文件时要还原到硬盘的根目录下;若用 BACKUP 命令将硬盘子目录下的文件备份到软盘,用 RESTORE 还原备份文件时要还原到硬盘相应的子目录下,否则不会成功。

27. COPY 和 DISKCOPY 两个命令使用上的不同点有哪些?

COPY 和 DISKCOPY 都是复制类命令,都能完成指定磁盘或文件的拷贝,它们的不同点是:

(1)工作方式不同。COPY 命令是以文件对文件的形式进行复制,而 DISKCOPY 则以磁道对磁道的方式进行复制。由于增、删和修改,一个文件常处于磁盘上不连续的扇区。读、写磁盘上不临接的文件要多花费时间。这时可利用 COPY 命令工作方式的特点。将原盘中的文件一个个复制到新盘上去,这样新盘上的文件就都处于连续的扇区中,即通过拷贝可以整理文件的存放空间。

(2)COPY 命令可复制全盘文件,也可以复制某个或某些指定的文件,而 DISKCOPY 只有全盘复制的功能。

(3)COPY 命令既可以在软盘和软盘之间,也可以在软盘和硬盘之间复制文件;DISKCOPY 只适用于软盘之间的复制。

(4)使用 COPY 命令复制前,目标盘必须已格式化,而用 DISKCOPY 若目录盘未格式化成与源盘一致的格式,它可在复制操作时格式化该盘。

(5)COPY 命令不能复制隐含文件(列目录时不显示的文件),故复制 DOS 系统盘(有两个隐含文件)时应选用 DISKCOPY 命令。

值得注意的是:COPY 命令只能从当前目录或指定目录中复制文件,若要复制包括子目录的所有文件,就要用 BACKUP 命令。该命令既可以将硬盘上的指定文件复制到软盘作为备份存档,也能传递长度大于一张软盘容量的文件。但用 BACKUP 命令得到的文件只能作为备份,调用时必须通过 RESTORE 命令将其恢复。DOS 3.20 和 DOS 3.30 版本还增加了 XCOPY 命令,用这个命令可以拷贝包括较低层次子目录在内的整个目录,XCOPY 复制的文件可以直接调用。

28. DIR 和 TREE 命令在使用上有何不同?

DIR 是 PCDOS 的内部命令,TREE 则属 PCDOS 的外部命令。两个命令都可以显示磁盘上文件的目录,但二者在使用中有所不同。

DIR 命令列出的是根目录或当前目录上的所有目录项或只列出指定文件的目录项。我们知道,从 DOS2.00 版起提供了一种新的目录结构(即由根目录和多级子目录一层层展开形成的目录结构),当磁盘上建有子目录且子目录为当前目录时,执行 DIR 命令显示会有:

·〈DIR〉 日期 时间

··〈DIR〉 日期 时间

这样两项。一个圆点标志的项表示该项是一个子目录;两个圆点标志的项表示该子目录归属于那个上级目录。如果你想知道每一个子目录的名称并了解磁盘上的全部目录结构,则要用 TREE 命令。要使用好文件目录,必须搞清楚根目录、子目录、当前目录、目录路径这几个基本概念,认清它们的区别。对子目录进行操作,有专门的建立、删除、显示和改变子目录的命令。

29. 删除子目录时应当注意那几点?

应当注意以下几点:

(1)被删除的子目录必须是空目录。如果子目录下存在文件先用删除文件的命令 ERASE(或 DEL)将其全部文件删除,再进行删除子目录的操作。

(2)被删除的子目录不为当前目录。

(3)根目录不可删除。子目录下有子目录,要由下向上逐级删除。

30. 使用 TYPE 命令显示文件内容时,为什么有时使屏幕显示发生紊乱而得不到要显示的结果?

TYPE 命令是在屏幕上显示 PCDOS 操作系统的 ASCII 码的文件的内容。所谓 ASCII 码文件即文本文件。如各种编译型语言的源程序(COBOL 语言,PASCAL 语言,FORTRAN 语言等),解释型语言的源程序(BASIC 语言,dBASE Ⅱ,Ⅲ等),批处理文件(·BAT),系统结构设置文件(CONFIG·SYS),应用软件的说明书,用编辑软件写的文章,建立的表格和书信等。若试图用 TYPE 命令显示非 ASCII 码的文件,例如文件扩展名为·EXE,·COM 等类型的文件,则使屏幕显示紊乱,或显示一些不可标识的图形,而不能得到要显示的文件内容。

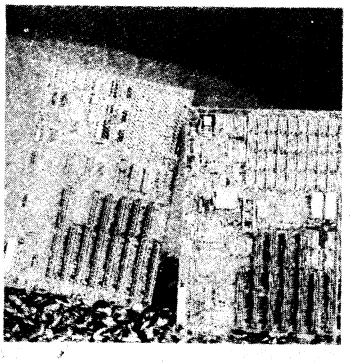
(转第 17 页)

電子與電腦

4

一九九一年

总期第73期



電子與電腦

目 录

·综述·

90年代的电脑技术和采购策略(上) …… 殷广济(2)

·PC用户·

用随机文件建立数据新探 …… 纪京松(5)

英文Wordstar文件的标准ASCII化 …… 赵 毅(7)

BASIC程序P保护在文件加密中的应用 … 何 骏(7)

在dBASEⅢ中实现多维数组功能 …… 石洪泰(8)

为DOS增加定义功能键外部命令 …… 张 红(9)

设置MS—DOS的功能键 …… 何管略(11)

编译BASIC运行文件CCDOS不相容的解决方法 …… 刘亚兵(12)

硬盘的加锁和解锁 …… 张 文(12)

键控图形处理程序 …… 徐维科(13)

彩色的高分辨图形 …… 潘麟元(14)

回溯法及其在编程中的应用 …… 陈 磊(15)

·学习机之友·

中华学习机使用技巧 …… 任 义(16)

实用改良的NEWCODE—POKES程序 …… 黄影昌(17)

巧解"八皇后"问题 …… 柳 杨(17)

关于APPLE和CEC—I机数计算误差的讨论——对《APPLE和CEC-I设计者的疏忽》一文的看法 …… 刘 峰(18)

——APPLE缺陷的解决办法 …… 陈盛施(18)

——数值计算中存在问题的解决办法 …… 朱洪波(18)

无DOS磁盘格式化程序 …… 姜宏岩(19)

跟踪机器语言程序的运行 …… 田俊成(20)

局部清屏程序 …… 程 华(20)

扩展BLOAD命令的功能 …… 蒋 阳(21)

从"重赏"谈起 …… 卢保堂 卢玉堂(21)

在APPLE SOFT下使用小汇编 …… 穆克威(22)

工资票面张数计算程序的简化 …… 孙承灿(22)

·C语言初阶·

第四讲 C运算符 …… 李文兵(23)

·学用单片机·

智力竞赛抢答装置 …… 宋济华(27)

单片机电子音乐编程实践 …… 张培仁 张新宇(30)

·学装微电脑·

数字游标长尺的制作 …… 易齐干(34)

·初级程序员级水平考试辅导讲座·

微机数据录入操作回答 …… 刘英 张茂芝(40)

·维修经验谈·

软盘驱动器机械故障的分析与维修 …… 魏学则(44)

更换集成块一法 …… 吴广元(44)

APPLE-Ⅱ机内存故障维修方法 …… 腾 达(45)

巧修CEC-I受损键盘 …… 黄文贵(45)

·新书与软件·

VIRUSCAN病毒处理软件包功能简介 …… 朱 禹(46)

Microsoft Windows 3.0窗口软件简介 …… (46)

最新版本PC TOOLS V6.0 …… (47)

·封二、封三· IBM PC主板芯片故障查寻表

IBM PC主板光器件排列图

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:北京市邮政局

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:徐云鹏

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

90年代的电脑技术和采购策略(上)

北方电脑公司　殷广济

自从1981年美国IBM公司推出PC机以来,至今仅有十年历史。目前,PC机市场上品种五彩缤纷,到处皆是。从主流机种的心脏,即称作CPU的芯片发展历史来看也还不到20年的历史。曾记得1984年8月当新的IBM PC/AT首次推出时,IBM的代表声称:这部新的电脑功能太强了,不能只将它视作单用户个人电脑,同时,他们向在场的人士保证,PC/AT的主要用途将是可以连接2—3台终端机的多用户系统的核心。当时,各方面的专家也赞成这种评估,并认为没有人真正需要象PC/AT般强大的计算能力。

然而,从那时候起,事情发生了巨大变化,拥有512KB内存,6MHz的PC/AT不再是个人电脑市场上的宠儿了。几年后的今天,PC/AT已进入其生命周期的衰退期,大有取代PC/XT作为电脑入门机器的趋势,32位电脑成为主流的时代已经到来。

采用80386,80486作CPU的电脑系统(以下称386,486系统)和采用80286作CPU的286系统最显著的差异在于386,486系统CPU是32位的,目前386CPU的时钟速率有16MHz,20MHz,25MHz,33MHz四种,486CPU的时钟速率有25MHz,33MHz两种,速率越快的处理机,指令执行速度亦快得多,从而使386、486系统的整体功能比286系统为佳。

386系统的运算速度达5MIPS,486系统的运算速度达20MIPS,基本上接近过去大型机的水平。采用33MHz 80486作CPU的个人电脑创造了速度记录,快,确实是快。但从市场和电脑系统设计角度来看,仍然会因BIOS及硬件兼容性,数学协处理器的运用,总线、扩充能力的不同而影响系统功能。要把90年代电脑技术变成采购策略是困难的。

好比一个健康的人,除头脑、四肢强壮之外,身体各器官的功能也需要正常。同样,若386、486系统只靠CPU运算速度快并不一定有用,存储器及输入输出(I/O)设备存取速度的配合也相当重要,若存储器及I/O装置的速度太慢,CPU就会时常被迫停留在静候的状态(Wait States),整体系统功能无法发挥。

造成CPU访问存储器时有等候现象发生的原因是通常使用DRAM作为主存储器,这种主存储器每隔一段时间就需要重新充电一次,否则电荷漏光,原本应为"1"的信息变为"0",影响输出结果的正确性。所以,当CPU要访问存储器时,通常会对存储器充电一次,而充电时间越长,就会造成CPU的等待现象。

在1985年,最好的处理器是80286,时钟频率仅有6/8MHz,8MHz的80286访问存储器时需要在125ns时间增量内完成,在250ns,375ns内完成,相对于零等待就是1个,2个等待状态。在处理器和存储器子系统间等待状态数被当成恒量系统性能的尺度。

到了1986年,80386—16MHz CPU问世,恒量等待状态的时间增量成了62.5ns,再使用传统的DRAM,系统就不可能达到真正的零等待,一些厂家如果按8MHz的80286作为恒量的标准,为了提高自己产品的性能,自称(错误的)他们的386系统运行在零等待状态,实际上是1/2个等待状态,他们担心的是他们的机器不是运行在零等待状态,比8MHz的AT零等待状态的机器运行还慢,系统功能也随之降低。

解决办法之一是将系统的DRAM全部换为速度较快(<80ns)的DRAM,当然,系统的价格会稍贵些。

第二种解决办法是大家看到的Cache memory方式,就是以一部分SRAM(Static Random Access Memory)来取代DRAM。其原理是按照统计,程序在大部分时间只是访问较少范围内的内存地址,采用SRAM,就是把经常存取的数据放在数量虽少,但速度很快的SRAM上(这种存储装置不需要充电,在同等条件下,访问速度比DRAM快2—3倍),不常用的数据放在DRAM上,一样可以增加系统的存取速度,由于SRAM比DRAM贵得多(在相同价格前提下,DRAM能提供的存储容量差不多是SRAM的4倍),所以通常只能以一部分SRAM来代替DRAM,而不是全部采用SRAM,一般采用32KB或64KB的SRAM,这样,访问数据的命中率可达到90%以上,而系统造价仍较便宜。

还有些优化DRAM存储系统结构的方法,如交互式和猝发转换结构等,来改进系统性能。

所谓交互的意思是在不同的DRAM体的存储地址交替进行预充电,使得对一个DRAM体进行访问时,对另一个DRAM体预充电,导致系统存储器循环时间等于DRAM访问时间,可以不计预充电时间(差不多占DRAM访问时间的一半),和采用SRAM差不多。

猝发转换是把DRAM体上的数据移到高速缓存SRAM上提高性能方法。

因此,在采购电脑系统时应注意DRAM的速度指标及结构,是否与CPU相匹配。

在PC/XT、AT时代,相容性并不是一个很严重的问题,或许说根本不是问题,因为蓝色巨人"IBM"提供了标准版本,兼容机厂商都以此版本为依据设计产品,只要支付IBM公司总产值的1%的许可证费即可,而不必支出大量的开发费,进行低成本的生产,低价在市场出售,1988年兼容机的产量达570万台,远

远超过 IBM 公司 180 万台的产量，大大威胁 IBM 公司市场。

为此，IBM 公司采用新策略，开发了新的微型机系统 PS/2，独自的操作系统 OS/2，IBM386 系统使用微通道（micro channel）结构，首先使用 3.5"软盘驱动器。同时，把生产许可证收费提到 5%，兼容机公司走与 IBM 兼容道路，利益风险太大，于是，COMPAQ，AST，Award，HP，NEC 等发表了与 IBM 公司 MCA 结构（micro channel architecture）相抗衡的扩展工业标准（Extended，industry standard architecture），1989 年 7 月，Intel 公司按 EISA 标准设计的 80386 芯片推出，同年 11 月推出 EISA 微机，90 年 2 月，EISA 标准的 80486 微机问世。所以，采用 Intel80x86 系列芯片的微电脑有三种结构，即 PC 及兼容机，MCA、EISA。在缺乏标准版本且厂商数目众多的情况下，用户对这些产品的相容性是值得注意的。

相容性的问题可以分为两个层次：BIOS（Basic Input/output System）相容性和硬件相容性。BIOS 的相容性是较容易做到，一般用户写的程序通常由键盘或磁盘驱动器存取信息，这通常是由系统提供的子程序来完成的，如果各系统提供的 BIOS 都能接受相同的变量，就可以说是 BIOS 兼容，对目前的 386 系统，若能执行 DOS 下的软件，则 BIOS 相容性是没有问题的。

硬件的相容性相对地困难，有人写程序习惯不通过 BIOS，而直接到某个地址访问信息，以争取时间，这时两个硬件系统兼容，则要每个地址扮演完全一样功能，在不涉嫌仿制的前提下是相当困难的。

目前，各种 386 系统之间相容性的问题就在于扩展卡的相容性，除了 IBM 的 PS/2－80 外，所有 386 系统均有 8 位，16 位，32 位扩展槽。毫无疑问，PS/2－80 的扩展卡不可用于其它系统上，因为总线的结构不一样，而其它系统的 8 位，16 位扩展槽，其目的在于让用户使用以前 XT，AT 的扩展卡，以降低成本。但 80386 的速度比 80286，8088 快得多，所以在使用这些扩展卡时，系统的速度必须放慢，也就是系统整体功能降低，有些厂家把系统速度限制减慢到某一限度，如 12MHz，此时某些扩展卡就不能在该系统上使用。

缺乏标准的 32 位扩展槽是另一个问题，每个厂家产品的 32 位扩展槽的接脚可能都不同，用户在购买前，务必弄清可以接哪些 32 位扩展卡，或者一次买齐，否则，买了卡以后而无法扩充系统功能，徒劳着急。

在购买电脑之前，应该审视整个系统，如果采用速度较慢的硬盘系统和显示系统，即使你用世界上最快的 CPU，你的系统也会象一条蜗牛。

硬盘驱动器一开始便与计算机事业同步发展。在 PC/XT 的时代，硬盘驱动器的存取时间大约为 85ms，AT 的硬盘驱动器存取速度大约为 40ms，由于个人电脑技术不断推进，磁盘驱动器技术也随之向前进步，为了配合 80386 的高速度，目前市场上硬盘驱动器的存取速度大约是 28ms，有的产品甚至提高到 16ms。目前，一般个人电脑的硬盘驱动器容量已进入 20MB 至 40MB 的范围，且大量采用 3.5 英寸的半高硬盘驱动器，平均的标准读写时间为 20 多 ms。由于网络技术及桌面印刷系统等发展，市场对大容量硬盘驱动器的需求增加，相继出现多种新产品，包括 300MB，600MB，1.2GB 的大容量硬盘驱动器。随着膝上型，笔记本型电脑的问世和普及，新一代 2.5 英寸 20MB 薄型（一英寸厚）的硬盘驱动器乘时而起。

3.5 英寸软盘驱动器较 5.25 英寸软盘驱动器设计较好，生产技术日渐成熟，成本下降，价格便宜，适应个人电脑微型化趋势并受膝上型电脑的普及所帮助，3.5 英寸 1.44MB 软磁盘驱动器亦将取代传统的 5.25 英寸 1.2MB 软盘驱动器的地位，2.5 英寸小型软盘驱动器也已面世多时。

现在，已有一些 4MB 的磁盘，一些制造商正在开发特殊的技术以适应 10M 容量及更高容量产品的需求。

由于光盘存储器记录密度高，存储容量大，随机存取，盘片可更换，信息保存寿命长，环境要求低，工作稳定可靠，是一个有前途的新兴外存储器设备，在市场上光盘存储器大约有三种类型：只读型，一次写入型和可抹型，容量均为 1000/3000MB。可改写的光盘系统也已面市，光磁盘容量虽然大大超过所有的硬磁盘，但缺点是存取时间稍慢，约为 40MS，传输速率也小，约为 2MB/S，但其成本价格却低于硬磁盘，且盘片可以更换，特别适用于图象处理，印刷系统，办公室自动化的文件存档系统和 CAD 系统，但由于技术及价格原因，短期内未必能普及。

除了存储介质以外，显示屏也是用户必须考虑的重要因素。即使你的电脑拥有再高的处理能力，如果不能将信息准确地传达给使用它的人便毫无意义。因此，对电脑画面的显示要求具有更高的清晰度。IBM 公司的 PS/2 产品未问世之前，彩色显示器都以 EGA 为标准，现在有了解析度更高，色彩更丰富的 VGA，PVGA 显示器，通过提高点距离密度和高对比度实现了高度清晰的显示，无论多复杂的图表，三维几何图形以及笔划繁杂的汉字均可色彩鲜艳地显示出来，甚至微妙差别亦清楚表示，分辨率有 640×480，800×600，1024×768，1280×1024 甚至更高，这取决于要你的电脑如何生动活泼，选择适当的显示器。

在 80386 族谱中，还有个小兄弟 80386SX，也是 Intel 公司 32 位产品之一。内部结构与 80386 相同，采用 100 接脚方形扁平塑封，体积比 80286、80386 都小，能耗也小，可以采用表面安装技术。芯片的外部数据总线为 16 位，能使电脑厂家以不太贵的 80286 元件来设计主板，可以 16 位机的价格来推销 32 位系统。

在许多公司计划开发 386SX 机时，对其前途莫衷一是。有的视其为闪闪发光的明星，有人说它对 16MHz 的 286 和 386 电脑市场冲击不大，有人认为正是令最终用户难以选择的另一类产品。

因为标准的 386 是以 32 位递增访问存储器，但 386SX 则以 16 位递增进行访问，因而芯片通过速度较慢。不去和 25MHz、33MHz 的 80386 芯片相比较，那样比也太公平，16MHz 的 386SX 和 16MHz 的 80386 相比，平均慢 23%，你要买真正的 32 位电脑系统，还是用 80386 好。虽然 386SX 和速度相当的，甚至和还要快一点的 286 芯片相比，也没有压倒优势，价格也较贵，但在网络服务器的应用中 386SX 就显得特别重要。如果你要运行 DESQ View 或 Lotus 1－2－3，买 80386SX 比买 80286 系统合适，如果你不想运行全 32 位软件，简单地说只是运行目前 DOS 和 OS/2 下的软件，买较快的 80286，比如 20MHz 或 25MHz 的系统，有更快运行速度并与 DOS 或 OS/2 软件兼容，价格也相宜。

80X86 系列芯片的新一代新星是 80486，它已开始用于个人电脑领域以及以性能为关键的工程工作站和小型计算机市场。（待续）

加快软件商品化进程　推动计算 机广泛应用

本刊记者

机电部计算机司委托中国计算机软件与技术服务总公司（简称：中软总公司，英文缩写 CS&S）举办“91 全国第三届计算机软件交流交易会。”会议于 91 年 4 月 4 日～9 日在北京国贸中心举行。来自全国各地的近百家单位近千种应用软件进行展示交流，形成了一定规模的软件商品市场。

所展示的软件大致可分为三类：一类是国家级，部委，大企事业单位开发的用于国家、部委、城市系统的管理软件，工具软件和专用软件包。二类是许多通用性的软件，如多种财务方面的管理软件，文字处理（照排），以及有关的改进 DOS 汉字处理功能的软件等。三类是引进和汉化的国外先进软件，如 SQL 语言、PCTOOL V6.0，汉化 window3.0 等。

用著作权法保护软件已成国际潮流，我国的著作权法将于 91 年 6 月实施，软件的出版发行工作势在必行，在这一新形势下，许多大学出版社的软件出版部纷纷在展示会上亮相，如北大出版社软件部；南大出版社软件部；华中理工大学出版社软件部等在会场设置摊位，进行试探性的促销活动，提高自己的知名度。可以预言将来大部分个人和集体开发的优秀软件将会以出版社软件出版物的形式进行推广并得到保护。

注：本文标题为机电部曾培炎副部长为“软交会”的题词。

全国青少年校外计算机活动研讨会

征文通知

由中国科协发起，并纳入 UNESCO IIP 项目总体计划，决定于 1991 年 8 月中旬在福建省厦门市举行《全国青少年校外计算机活动研讨会》。具体日期将在 1991 年 8 月全国青少年信息学（计算机）竞赛期间。该会议由中国计算机学会和中国科协青少年科技中心主办，厦门市承办。现将征集论文事项通知如下：

一、会议主题：计算机活动对青少年教育的影响

二、征文的主要内容：

1. 计算机普及与未来人才；

2. 中小学普及计算机知识的内容、方式探讨；

3. 计算机普及教育对其它中小学课程教育的促进与辨证关系；

4. 从认知心理科学角度探讨计算机普及教育的迁移作用；

5. 青少年校外计算机活动中心的活动模式与功能分析；

6. 综述性论文。这一类论文要求有具体统计资料、事实例证、学术观点和科学分析。不可以投寄空洞无物或工作总结式文章；

7. 开发先进的计算机教育软件（含示范表演）。

征文要求

文章一般限 5000 字，前面附有 300 字摘要（摘要用中英两种文字），书写在 400 字稿纸上，力求文字语言、标点符号正确通畅，图表规范，标出图题和表题及序号并在文中放在合适的位置上。一式二份于 1991 年 6 月 10 日前寄到：北京师范大学　计算中心　谢文杰同志处

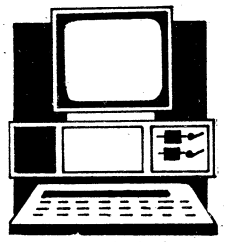
（邮编：100875　电话：(01)2012288－2708）

四、重要日期

1. 截稿日期　　1991 年 6 月 10 日

2. 发出录用通知　1991 年 6 月 30 日

3. 会议统一编印论文集。



PC用户

用随机文件建立数据库新探

纪京松

很多单位大都用 DBASE Ⅲ 建立数据库管理数据。但 DBASE Ⅲ 资源开销大，速度又慢，所以，一些系统完全可以用 BASIC 语言来实现。我所近几年组建了一些大、中型水利自动化管理系统，都是用 BASIC 语言建立随机文件对水情数据进行存储管理的。

水情关心的只是数据的时间、站名，各数据间没有网状联系。所以，我们以时间为基础换算出各数据存取所需的记录号。例如：以一个小时为时段。

CODE＝MONTH＊31＊24＋DAY＊24＋TIME

7 月 7 日 8 时的数据记录号便是：

CODE＝7＊31＊24＋7＊24＋8

按此记录号，便可以存取所需要的水情数据，文件中也无需存储时间数据，可节省硬盘资源。各水文站站名、水库名也无需存储，只要各记录中各字段次序一定下来，就可以按位置找到各水库的水情数据。下面举几个简单的例子说明一下：

（一）存放四个水库的雨量数据：

程序一

```
5 DIM A1$(4),A$(4)
10 OPEN "S" AS #1
20 FIELD #1,10 AS A$(1),10 AS A$(2),
       10 AS A$(3),10 AS A$(4)
30 INPUT "月:";MONTH
40 INPUT "日:";DAY
50 INPUT "时:";TIME
60 INPUT "A 水库雨量:";A1$(1)
70 INPUT "B 水库雨量:";A1$(2)
80 INPUT "C 水库雨量:";A1$(3)
90 INPUT "D 水库雨量:";A1$(4)
100 CODE=MONTH*31*24+DAY*24+TIME
110 FOR I=1 TO 4:LSET A$(I)=A1$(I):NEXT I
120 PUT #1,CODE
130 CLOSE:END
```

运行：RUN↙

月：7↙

日：7↙

时：8↙

A 水库雨量：264.33↙

B 水库雨量：32.1↙

C 水库雨量：5.00↙

D 水库雨量：102.51↙

这样 7 月 7 日 8 时，四个水库的当时雨量数据就存入随机文件“S”中，记录号为 5384。

（二）取出四个水库 7 月 7 日 8 时的雨量数据

程序二

```
5 DIM A1$(4),A$(4)
10 OPEN "S" AS #1
20 FIELD#1,10 AS A$(1),10 AS A$(2),
       10 AS A$(3),10 AS A$(4)
30 INPUT "月:";MONTH
40 INPUT "日:";DAY
50 INPUT "时:";TIME
60 CODE =MONTH*31*24+DAY*24+TIME
70 GET #1,CODE
80 PRINT "A 水库雨量:";A1$(1)
90 PRINT "B 水库雨量:";A1$(2)
100 PRINT "C 水库雨量:";A1$(3)
110 PRINT "D 水库雨量:";A1$(4)
120 CLOSE:END
```

运行：RUN↙

月：7↙

日：7↙

时：8↙

显示：

A 水库雨量：264.33↙

B 水库雨量：32.1↙

C 水库雨量：5.00↙

D 水库雨量：102.51↙

以上例子只管理了四个水库的数据，每一项定了 10 个字节。如果要管理成百上千个水库的数据，既浪费资源且无法实现，因随机文件一个记录最长只有 32767 字节。因此需修改数据库结构，进一步节约硬盘和内存资源开销，才得以管理上万个站的水情数据。

下面还是以 4 个水库的例子来说明。原来每个记录缓冲区占 40 字节，现在把它压缩成 8 字节。

程序三

```
10 REM 新思维建立数据库
20 DIM A$(4),NAM$(4),A1$(4),A(4)
30 COLOR 15,1:CLS
40 LOCATE 6,9:PRINT "输入日期: 月 日 时"
50 LOCATE 6,20:INPUT MON
60 LOCATE 6,27:INPUT DAY
70 LOCATE 6,34:INPUT TIM
80 FOR I=1 TO 4:READ NAM$(I):NEXT I
90 CLOSE:OPEN "Y.DAT" AS #2
100 FIELD #2,2 AS A$(1),2 AS A$(2),
       2 AS A$(3),2 AS A$(4)
110 FOR I=1 TO 4
120 PRINT NAM$(I);:INPUT A1$(I)
130 NEXT
140 FOR I=1 TO 4
150 A(I)=VAL(A1$(I))*100:A1$(I)=MKI$(A
(I))
```

```
160 LSET A$(I)=A1$(I)
170 NEXT I
180 CODE=MON*31*24+DAY*24+TIM
190 PUT #2,CODE
200 CLOSE:OPEN "Y.DAT" AS #2
210 FIELD #2,2 AS A$(1),2 AS A$(2),
        2 AS A$(3),2 AS A$(4)
220 GET #2,CODE
230 FOR I=1 TO 4
240 A(I)=CVI(A$(I))/100:A$(I)=STR$(A(I))
250 LON=LEN(A$(I)):LON=LON-1
260 A$(I)=MID$(A$(I),2,LON)
270 PRINT NAM$(I);A$(I)
280 NEXT I
290 CLOSE:END
300 DATA "A 水库雨量:","B 水库雨量:",
        "C 水库雨量:","D 水库雨量:"
```

运行时输入

月:7↙
日:7↙
时:8↙
A 水库雨量:264.33↙
B 水库雨量:32.01↙　　雨量单位 mm
C 水库雨量:5.00↙　　保留两位小数
D 水库雨量:102.51↙

输出:A 水库雨量:264.33
B 水库雨量:32.01
C 水库雨量:5.00
D 水库雨量:102.51

其中,150 句:A(I)=VAL(A1$(I))*100 把 A1$(I)转换成相应的 2 个字节的内部代码字符串。

这样,存入随机文件中的数据,实际上是各雨量数据相对应的 2 个字节的内部代码,因此可大大地节省机器资源开销,完成大量数据的存储。

240 句:将内部字符串还原成原来的字符串,为以后的定位修改,显示再存储提供条件。

250～260:消除符号位,水情数据都为正数。

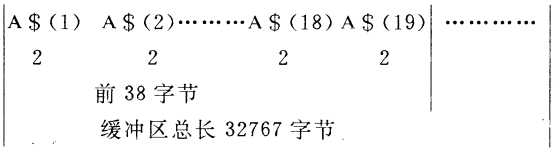
以上的程序,表明此方法比用 DBASE Ⅲ 建库优越得多,而且编译 BASIC 速度也快。由此推广开去就能建更大型的数据库,每个记录存取更多的数据。

随机文件一个记录最长为 32767 字节,这就意味着一个记录最长可以存取 16383 水情数据。

一个很棘手的问题是如何把一个记录缓冲区划分成 16383 个区段。用一个 FIELD 语句肯定是无法实现的。因为 BASIC 每个语句的长度最多不能超过 256 个字节,其中还包括行号。经过实验一个 FIELD 最多可划分 19 个区段。即

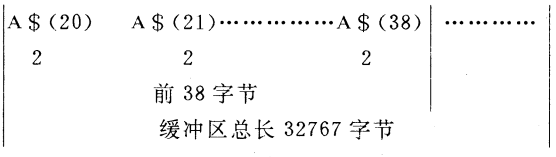
```
:
100 FIELD #1,2 AS A$(1),2 AS A$(2),…
        ……,2 AS A$(18),2 AS A$(19)
110 FIELD #1,2 AS A$(20),2 AS A$(21)…
        ……,2 AS A$(38)
:
```

如此划分下去不但麻烦,而且也不对。经过多次反复研究试验:发现 110 句的 FIELD 语句并不象某些教科书上所写继续从第 39 个字节开始划分以后的区段,而是又从头开始,如图一、图二说明。



图一

图一是执行 100 句的结果,前 38 个字节划分成 19 个区段,分别为 A$(1),A$(2)……A$(19)。



图二

图二是 110 句的执行后结果,又把前 38 个字节划分给了 A$(20),A$(21)……A$(38)。

因此,要使 110 句从第 39 个字节开始划分,则 110 句 2 AS A$(20)前面必须加一个偏移量,就是把前 38 个字节划给一个区段 A$。即

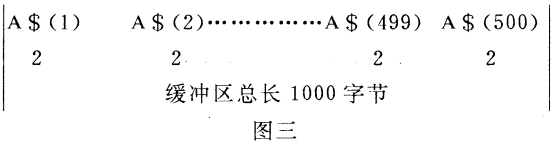
```
110 FIELD #1,38 AS A$,2 AS A$(20),……
        ……,2 AS A$(37)
```

为了方便,我们采用循环语句完成。以下是建立一个 A.DAT 随机文件,需要存放 500 个数据的程序。

程序三

```
:
90 CLOSE:OPEN "A.DAT" AS #1 LEN=1000
100 FOR I=1 TO 500
110 FIELD#1,2*(I-1) AS A$,2 AS A$(I)
120 NEXT I
:
```

110 句 A$ 为偏移量,执行结果如图三。



图三

另外,BASIC 有良好的绘图功能和通信功能,系统中的地图绘制等雨量线、水库剖面、各测站的通信都能用高级编译 BASIC 完成。这样的管理系统是 DABSE Ⅲ 所无法实现的。

如果数据过大,可采用单、双精度,转换成 4、8 个字节的内部代码时用 MKS$,MKD$ 函数。

当然,用户也可以根据自己的实际情况计算记录号,安排数据的存取,建立所需的数据库。

工作中得高昆高级工程师的大力帮助,在此深表感谢。

英文 WordStar 文本文件的标准 ASCII 化

河北师范学院　赵　毅

英文 WordStar(4.0 版)是文字处理中常用的软件。它的功能丰富,简便易学。由于配备了一个主辞典和一个用户辞典,我们可以用机器来检查和更正输入文本的拼写错误。最具特点的是与之配套的收有 22 万词条的同义词词典,机器响应迅速,对于经常用英文写作的用户来说,的确是个好帮手。然而,经它编辑出的文本文件采用的不是标准 ASCII 码,无法用其它的系统来显示,重新编辑或打印。笔者谨为新用户介绍解决这一问题的三种常用方法。

(1)在主菜单(Opening Menu)下打 N,选用非文本方式,该方式采用的是标准 ASCII 码。输入欲处理文件的文件名,打回车键。打开文件后,键入命令 ^ B 或 ^ QU。处理完后存盘即可。注意此法不会删掉文件中已设定的打印控制命令字符。

(2)在主菜单下打 P,进入打印操作状态。在最后一个打印提示语句后键入 ASCII。这样,WordStar 会动处理你的文本文件并以 ASCII.WS 的文件名存盘。你也可以自己规定文件名,具体方法是在键入 ASCII 后打一个 > 符号,然后输入文件名。注意该法保留原文件,但会删除新的文件中的控制字符。

(3)有些用户需要写作中英文两种文字夹杂的文件。他们通常用中文 WordStar 来输入汉字并排版,用英文 WordStar 来处理英文。下面为大家介绍笔者用 BASIC 编写的一个小程序,该程序在进行标准 ASCII 化处理的同时,能够在屏幕上显示处理结果:

```
10 CLS:LOCATE 5,20:INPUT "原文件名",S$
20 LOCATE 8,20:INPUT "新文件名?",T$:CLS
30 OPEN S$ FOR INPUT AS #1:OPEN T$ FOR OUTPUT AS
#2
40 IF EOF(1)=-1 THEN 140
50 X$=INPUT$(1,#1):M=ASC(X$)
60 IF M<128 THEN 80
70 IF M<=160 AND M>=128 THEN X$=CHR$(M-128)
80 PRINT #2,X$;:PRINT X$;:GOTO 40
90 Y$=INPUT$(1,#1):N=ASC(Y$)
100 IF N<128 THEN X$=CHR$(M-128)
110 PRINT #2,X$;Y$;:PRINT X$;Y$;:GOTO 40
120 IF N>160 THEN110
130 X$=CHR$(M-128):Y$=CHR$(N-128):GOTO 110
I40 CLOSE #1,#2:CLS
150 END
```

BASIC 程序 p 保护在文件加密中的应用

何　骏

BASIC 程序有三种方式保存文件。

1. SAVE "文件名"↙——压缩二进制格式。

2. SAVE " 文件名",A↙——ASCII 格式。

3. SAVE "文件名",P↙——编码二进制格式。这是一个保护性选择项,当程序经过 P 保护后,试图对它发出 LIST,EDIT 命令都会失败。

笔者编了一个程序 KEY20,利用这一点来对文件进行加密,其主导思想是把 ASCII 码文件的每一句加上行号及注解符"'",使之能在 BASIC 状态下用 LOAD 命令调入。

加密时先运行 FKEY20.BAS,输入需加密的文件名,然后返回操作系统状态,把文件扩展名改成.BAS,再进入 BASIC 状态,用 LOAD 调入,用第三种方式保存文件,最后恢复原先的文件扩展名。这样加密后就不能在 EDLIN 或 WS 下进行编辑。

解密时,在 BASIC 状态下用 LOAD 把加密的文件调入,先破 P 保护(注:有关破 P 保护的方式有几种,在此从略),然后用上述方式 2 保存文件,再运行 FKEY20.BAS,最后恢复原先文件名即可。

```
10 REM FILE-KEY PROGRAM VERSION 2.0
20 KEY OFF:CLS:DIM A$(1000)
30 PRINT "FILE-KEY PROGRAM VERSION 2.0"
40 PRINT:PRINT "PROGRAMMED BY HEJUN":PRINT
50 INPUT "PLEASE INPUT FILENAME (EXIT ---
ENTER):",F$
60 IF F$="" THEN SYSTEM
70 INPUT "LOCK OR UNLOCK (1 OR 0)?",K
80 IF K<>1 AND K<>0 THEN 70
90 OPEN "I",#1,F$:I=0
100 IF EOF(1) THEN CLOSE #1:GOTO 120
110 I=I+1:LINE INPUT #1,A$(I):GOTO 100
120 OPEN "O",#1,F$
130 IF K=0 GOTO 180
140 FOR J=1 TO I
150 C$=STR$(J):L=LEN(C$):C$=MID$(C$,2,L-
1)
160 A$(J)=C$+" '"+A$(J):PRINT #1,A$(J)
170 NEXT J:GOTO 220
180 FOR J=1 TO I
190 P=INSTR(A$(J)," '":L=LEN(A$(J))
200 A$(J)=RIGHT$(A$(J),L-P):PRINT #1,A$(J)
210 NEXT J
220 CLOSE #1:ERASE A$:GOTO 20
```

在 dBASE Ⅲ 中实现多维数组功能

内蒙哲里木盟饲料厂　　石洪泰

由于 dBASE Ⅲ 没有数组语句，使得许多计算问题难以解决，但是人们巧妙地运用 & 函数，很容易地实现了一维数组，如程序一，A&J 相当于 BASIC 语言中的 A(J)语句，只要注意循环变量 I 为“N”型，应用 J=STR(I,1)将其转化为“C”型才能使用 & 函数。

对于 BASIC 语言中的 A(J,N)，则不能简单地用 A&J&N 表示，这样表示在 J、N 小于 10 程序能够正常运行(如程序二)。在 BASIC 语言中的 A(1,11)或 A(11,1)，也就是在 J=11 或 N=11 时，dBASE Ⅲ 中 A&J&N 都表示为 A111，分不清是 A(1,11)还是 A(11,1)，这就不能正常进行数组计算。

笔者曾想到在 dBASE Ⅲ 中数组 A&J&N 中间加字母 B 的办法即 A&JB&N，但程序运行后出现了不可认字符，即 &JB 不可认，为此笔者用 &X 代替字母 B，即 A&J&X&N(相当于 A2B3 的形式)，实现了二维数组在 dBASE Ⅲ 中的计算的通用性(如程序三)。

同理用 A&J&X&N&Y&E(其中 &Y 为 C 型，&E 为数值变量)相当于 A1B2C3G 形式，代替 BASIC 语言中的三维数组 A(1,2,3)实现了多维数组在 dBASE Ⅲ 中应用的通用性。

程序一：

```
SET TALK OFF
CLEA
I=0
DO WHIL I<9
I=I+1
J=STR(I,1)
A&J=I * 20
ENDDO
I=0
DO WHIL I<9
I=I+1
J=STR(I,1)
? A&J
ENDDO
```

程序二：

```
SET TALK OFF
CLEA
I=0
DO WHIL I<9
I=I+1
M=0
DO WHIL M<9
M=M+1
J=STR(I,1)
N=STR(M,1)
A&J&N=M * 20+I * 100
ENDD
ENDD
I=0
DO WHIL I<9
I=I+1
M=0
DO WHIL M<9
M=M+1
J=STR(I,1)
N=STR(M,1)
? A&J&N
ENDD
ENDD
```

程序三：

```
SET TALK OFF
CLEA
I=0
X="B"
DO WHIL I<15
I=I+1
IF I>=10
J=STR(I,2)
ELSE
J=STR(I,1)
ENDIF
M=0
DO WHIL M<15
M=M+1
IF M>=10
N=STR(M,2)
ELSE
N=STR(M,1)
ENDIF
A&J&X&N=M * 20+I * 100
ENDD
ENDD
I=0
X="B"
DO WHIL I<15
I=I+1
IF I>=10
J=STR(I,2)
ELSE
J=STR(I,1)
ENDIF
M=0
DO WHIL M<15
M=M+1
IF M>=10
N=STR(M,2)
ELSE
N=STR(M,1)
ENDIF
? A&J&X&N
ENDD
ENDD
```

为 DOS 增加定义功能键外部命令

湖北省公路局科研所　张　红

许多高级程序设计语言，如 BASIC、DBASE Ⅲ 都提供了对功能键进行定义的功能。这种“软键”功能使用户能根据需要将一些经常使用的关键字或字符串赋予功能键。这样，在要输入这些关键字和字符串时，用户只需按一次功能键就能完成，不必逐个字符输入，从而简化了操作，减少输入错误，提高操作人员的键盘输入效率。然而在 MS-DOS 没有提供具有此功能的命令，这是一个缺憾。为此作者为 MS-DOS 增添一条外部命令 DEFINE，以弥补这一空白。

本文详述 DEFINE. COM 的设计要求、设计思路、具体实现、使用举例和要求。此程序不仅为用户提供了方便，而且还提供一个借助汇编语言对操作系统的资源进行扩充的开发实例。

一、扩充键盘的控制

2.00 版本以上的 MS-DOS 提供了扩充键盘控制功能，允许借助设备驱动程序 ANSI. SYS 对键盘上的任意键字进行重定义。将功能键或其它键字的控制符及所要定义的内容按一定的标准格式组合起来，利用输出定向到屏幕的各种 DOS 输出功能调用，即可完成键字的重新定义。由于 ANSI. SYS 提供的功能，可在由 2.00 以上版本 MS－DOS 支持的 IBM PC/XT 及其所有兼容机上实现，借助此功能开发出的软件有很好的兼容性。

对键盘重新定义所用的控制序列有以下格式：

(1)ESC［#；#；…… #p

(2)ESC［#；“string”；p

(3)ESC［#；“string”；#；#；…… #p

其中 # 代表一个 ASCII 码，ESC 是代码为 1BH 的控制符。第一个 # 是要定义的键字的 ASCII 码。功能键 F1～F10 对应的扩充 ASCII 码为 0；59～0；68。后面的“string”和 # 分别表示定义的内容和其他键字的 ASCII 码，尾部的小写 p 是控制序列结束标志。

例如，ESC［65；81p 定义键字 A(ASCII 码为 65)为 Q(ASCII 码为 81)；ESC［98；115p 定义键字 b(ASCII 码为 98)为 s(ASCII 码为 115)；而 ESC［0；68；“dir/p”；13p 则定义功能键 F10 为 dir/p 命令后跟一个回车符(ASCII 码 13)。

二、DEFINE 的基本要求及实现

作为 DOS 的外部命令，要求 DEFINE 执行速度快、少占用内存空间；另外，要充分利用操作系统现有的功能和资源。这些只有通过汇编语言程序才能达到。为此，作者选用宏汇编程序设计语言 MASM，借助 DOS 功能调用，进行 DEFINE 的开发。为了减少程序的长度，加快运行时装入内存的速度，汇编程序按. COM 文件格式要求编写。

DOS 的每条命令作为一个专用的处理程序，其处理步骤是：1. 获取必需的命令执行参数(含缺省情况)；2. 对参数进行分析；3. 当参数符合程序设定的要求时，程序按设计的功能继续执行；否则显示出错信息，然后回到操作系统待命状态。DEFINE 的流程正是按照这些工作步骤设计的。

三、具体实现

根据上述设计思路，结合 ANSI. SYS 的功能，作者编写了汇编源程序 DEFINE. ASM。读者可以使用任何一种编辑程序输入 DEFINE. ASM，然后按顺序执行以下操作：

```
C>MASM DEFINE
C>LINK DEFINE
C>EXE2BIN DEFINE. EXE DEFINE. COM
```

生成的 DEFINE. COM 即可作为一个外部命令在 DOS 下直接使用。

四、命令格式和使用举例

DEFINE 使用的命令格式如下：

DEFINE/FUNCNAME/parm1/parm2

符号/是命令参数的分隔符，不能缺少。参数 FUNCNAME 是要定义的功能键的名字，采用的形式为字母 F(大、小写均可)后紧随一功能键的序号，如 F1、f10，注意序号的值必须是 1～10 的整数。参数 parm1 是要定义的内容，它可以是任何字符串(包含空格符)或关键字，但不能含斜线符/，此参数的最大长度设定为 69。参数 parm2 是一个标识参数，它表示是否在定义的内容后面接回车符，通常它只有 Y 和 N(大、小写均可)两种选择。

在操作系统下，很多功能键都有其特定的定义。有时为了去除用户的定义，有必要恢复操作系统对这些功能键的原定义，DEFINE 还提供了满足这一要求的功能，只需将参数 parm1 和 parm2 均置为空即可。

下面举例说明命令的使用：

(1)DEFINE/F10/DIR A：/Y

将功能键 F10 定义成 DIR A：并且后跟一个回车；

(2)DEFINE/F1/功能键定义外部命令/N

将功能键 F1 定义成字符串“功能键定义外部命令”；

(3)DEFINE/F1//

此命令将功能键 F1 恢复到操作系统的原定义；

下面还有一些其它的例子：

```
DEFINE/F10/TYPE A：DEFINE. ASM/y
DEFINE/F7/张红/N
DEFINE/F3/mov ax，/N
DEFINE/F5/DBASE JHTJ. PRG/n
```

当未按要求的格式输入参数时，DEFINE 显示出错信息，并给出一个正确的示例。

运行 DEFINE 后，所设的功能键定义一直保持到

再次使用 DEFINE 重新定义或再次起动 DOS 为止。

五、命令使用要求

在使用 DEFINE 前必须确保：

(1)用户的 MS-DOS 版本号为 2.00 或更高；

(2)系统盘上必须有设备驱动程序 ANSI.SYS，并且在配置文件 CONFIG.SYS 中应该有安装 ANSI.SYS 的命令语句：DEVICE＝ANSI.SYS。

```
LF          EQU 0AH             ;换行的 ASCII 码
CR          EQU 0DH             ;回车的 ASCII 码
SPACE       EQU 20H             ;空格符的 ASCII 码
SLASH       EQU 2FH             ;/符的 ASCII 码
DOLLAR      EQU 24H             ;$符的 ASCII 码
;定义宏体
CLS         MACRO               ;清屏
            MOV AL,3
            MOV AH,0
            INT 10H
            ENDM
;
DISP        MACRO NOTE          ;字符串显示
            LEA DX,NOTE
            MOV AH,09H
            INT 21H
            ENDM
;程序代码段
CODE        SEGMENT PARA PUBLIC 'CODE'
            ASSUME CS:CODE,DS:CODE,SS:CODE,ES:
                   CODE
            ORG 100H
START       PROC NEAR
            JMP MASTER
BELL        DB  07H,07H,"$"
ERROR       DB  CR,LF,'命令格式错误！正确命令格式
            (例):DEFINE/F10/定义功能键/N',"$"
CPYRIGHT DB '(C)Copyright 1989 年湖北省公路局科研所
                张红',LF,CR,"$"
FUNCTION DB  1BH,'[0;'"
            DB  30 DUP(?)
MASTER:
            CLS                 ;清屏
            DISP CPYRIGHT       ;显示信息
            CLD                 ;设定串操作方向
            MOV SI,80H          ;置命令行参数块起始地址
            XOR AX,AX           ;初始化 AX 寄存器
            LODSB               ;取命令行参数长度存于
                                AL
            CMP AX,0            ;长度为 0 吗?
            JE ERRDS            ;长度为 0 转错误处理
            MOV CX,AX           ;取命令行参数长度
LABEL1:     LODSB               ;去除命令参数块前的空格
            CMP AL,SPACE        ;
            JNE LABEL2          ;
            LOOP LABEL1         ;
LABEL2:     JCXZ ERRDS          ;命令参数全是空格,出错
            CMP AL,SLASH        ;
            JNE ERRDS           ;参数块前导标志错误
            LODSB
            OR AL,20H           ;参数串第一字符化为小写
            CMP AL,'f'
            JNE ERRDS           ;第一字符不是 F 或 f,出错
            LODSB
            CMP AL,'1'          ;判断功能键的序号
            JNE LABEL3          ;判知序号不是 1 也不是
                                10
            MOV BL,AL
            LODSB
            CMP AL,'0'          ;功能键的序号是 10 吗?
            JE LABEL4           ;判知功能键的序号是 10
            CMP AL,SLASH
            JNE ERRDS           ;功能键的序号＞10,出错
            MOV AL,BL
            DEC SI
            JMP LABEL4
LABEL3:     CMP AL,'1'          ;
            JB ERRDS            ;功能键序号＜1,出错
            CMP AL,'9'          ;
            JG ERRDS            ;功能键序号＞9,出错
            JMP LABEL4
ERRDS:      DISP ERROR          ;显示出错信息
            DISP BELL           ;响铃
            JMP EXIT            ;
LABEL4:     LEA DI,FUNCTION+4
            MOV BL,AL
            CMP AL,'1'          ;
            JE LABELL2          ;功能键 F1
            MOV AL,'6'          ;
            JMP LABELL3         ;其他功能键(F2～F10)
LABELL2:    MOV AL,'5'          ;
LABELL3:    STOSB
            MOV AL,BL
            LEA DI,FUNCTION+5
            CMP AL,'0'          ;
            JE LABEL5           ;功能键 F10
            CMP AL,'1'          ;
            JE LABELL4          ;功能键 F1
            SUB AL,02H          ;其他功能键(F2～F9)
            JMP LABEL6          ;
LABELL4:    MOV AL,'9'          ;
            JMP LABEL6          ;
LABEL5:     MOV AL,'8'          ;
LABEL6:     STOSB
```

```
        LODSB                   ;
        CMP AL,SLASH            ;
        JNE ERRDS               ;未找到第二分隔符/,出错
        LEA DI,FUNCTION+8
        LODSB
        CMP AL,SLASH            ;后接的是/吗?
        JE LABEL8               ;是,转恢复原定义处理
LABEL7: CMP AL,CR               ;
        JE ERRDS                ;后接的是回车,出错
        CMP AL,SLASH
        JE LABEL9               ;定义已完,转后续处理
        STOSB
        LODSB
        JMP LABEL7
LABEL8  LEA DI,FUNCTION+7
        LEA SI,FUNCTION+2
        MOV CX,4                ;
LABELL9: LODSB                  ;
        STOSB                   ;
        LOOP LABELL9
        JMP LABEL12
LABEL9: PUSH AX                 ;
        MOV AL,'"'              ;
        STOSB                   ;
        POP AX
LABEL10: LODSB
        OR AL,20H               ;将回车标识符转换成小写
        CMP AL,'n'              ;
        JE LABEL12              ;不接回车符
        CMP AL,'y'
        JNE ERRDS               ;后随参数不是 Y(y)或 N
                                (n),出错
LABEL11: MOV AL,';'             ;进行后接回车符处理
        STOSB
        MOV AL,'1'
        STOSB
        MOV AL,'3'
        STOSB
LABEL12: MOV AL,'p'             ;p 是控制序列结尾标志
        STOSB
        MOV AL,DOLLAR
        STOSB
        DISP FUNCTION           ;进行功能键定义
EXIT:
        DISP BELL               ;响铃提示用户操作完毕
        MOV AX,4C00H            ;返回到操作系统
        INT 21H
START   ENDP
CODE    ENDS
        END START
```

设置 MS－DOS 的功能键

广东汕头　金山中学　何管略

众所周知，在 IBM 型机的 BASICA 系统中，有一组字串功能键，如 Alt＋L＝Locate，Alt＋S＝Screen 等，非常灵活、方便。我们从实践中体会到，运用 Prompt 命令，也可以在 MS－DOS 3.3 系统下，设置字串功能键。下面介绍具体制作程序：

(1)在 CONFIG.SYS 文件中写一条 DEVICE＝ANSI.SYS 命令，并保证根目录中含有 ANSI.SYS 文件，使系统支持一系列以 ESC 字符领头的字元组的特殊功能。

(2)键入 Prompt ng$e[0;18;"Edlin"p 命令。其中 $n＝当前的盘符(也可改用 $p＝当前的目录路径)，$g＝大于号，即">"。这个部分放前放后都可以；不过，从整体来说，这个部分却是必不可少的。如果缺少的话，其它引数的功能便无从实现。

$e＝ESC 字符。[0;18;＝Alt＋e 的组合键码。这个码值，大小写状态都相同，仅由扫描位置确定。从左向右，逐个加 1，如 Q＝16；从上向下，逐行增加 14，如 A＝30。字串"Edlin"就是我们所要定义的符号，大、小写都可以，显示时一律原样照抄；但不接受汉字。字符 p 表示字串功能键的定义结束。

注意：以 $ 领头的特殊字符，如 e,n,g,p 等，都用小写，如果改为大写，其相应的功能自行消失。

(3)Prompt 命令的引数最多可达 127 个字符，所以在同一条命令行的后面，还可继续定义其它的功能键，如 $e[0;68;"Dir";13p 等。其中[0;68;是 F10(不是 Alt＋F10)键的扩展码，13 是回车键的 ASCII 码。

(4)如果设置的字串功能键较多，可再键入一条或几条类似的 Prompt 命令。这些工作，键盘操作难免出错，而且断电关机后功能消失，下次开机又得重做，令人厌烦，因此最宜写成批处理程序。

值得特别注意的是，在同一个批处理程序中，如有两条以上的 Prompt 命令，则只有最后一条有效，在此之前的 Prompt 命令全都无效。因为它们在尚未实现之前，就被后者取而代之了。

怎样解决这个矛盾呢？有多少条 Prompt 命令，就写多少个批处理程序，每个程序只有一条 Prompt 命令。开机之后，依次键入.BAT 程序的主名就行了。

本期特向读者介绍在 DOS 状态下定义功能键的两篇文章。两文的设计风格不同，但都富启发性和教益，相信大家会喜欢。我们还希望来稿介绍设备驱动程序 ANSI.SYS 的详细资料。

编者

编译 BASIC 运行文件与 CCDOS 不相容的解决方法

刘亚兵

IBM－PC 编译 BASIC 由于与解释 BASIC 基本兼容，且运行速度快、占用内存少，深受用户喜爱。然而，它与 CCDOS 却不相容，表现在：其运行文件在运行和返回操作系统后均不能显示汉字。通过分析，笔者找到了解决这一问题的方法。

在带 O 参数编译 BASIC 运行文件.exe 中，均有如下两段指令：

```
5899:064E0 B40F          MOV AH,0F
5899:0650 E8B0FF         CALL 0603
5899:0653 BB6900         MOV BX,0069
5899:0656 D7             XLAT
5899:0657 50             PUSH AX
5899:0658 B400           MOV AH,00
5899:065A E8A6FF         CALL 0603
5899:065D 58             POP AX
5899:065E A27200         MOV [0072],AL

5899:0603 55             PUSH BP
5899:0604 56             PUSH SI
5899:0605 57             PUSH DI
5899:0606 CD10           INT 10
5899:0608 5F             POP DI
5899:0609 5E             POP SI
5899:060A 5D             POP BP
5899:060B C3             RET
```

其中，064e～0650 处指令完成取 CRT 当前显示模式，模式代码存于寄存器 AL 中；0658～065a 处指令完成 CRT 显示模式设置。通过 DEBUG 的运行命令 g 和单步命令 t 跟踪发现：在 0656 处 XLAT 指令将寄存器 AL 置零。这样，再通过 0658～065a 处指令就将 CRT 显示模式设置成了 40×25 黑白字符模式，而 CCDOS 操作系统却要求 CRT 处于图形模式下。因此，只要将 XLAT 指令改成 NOP（空操作）指令，系统就不会使 CRT 进入字符显示模式了。这样，编译 BASIC 的.exe 文件在运行和返回操作系统后就能显示汉字，从而解决了两者不相容的问题。实践证明：修改是切实可行的，而且对原来程序功能的实现无任何不良影响。

修改步骤（假定文件名为 example.exe）：

1. 将 example.exe 换名为 example；

2. 通过 DEBUG 将 example 调入内存，用 S 命令检索出 XLAT 指令的地址，按此地址将其内容由 D7（XLAT）修改为 90（NOP）；

3. 存盘、退出 DEBUG；

4. 将 example 换名为 example.exe。

附修改过程屏幕显示信息（带下划线内容表示应由用户键入）。

```
A)REN EXAMPLE.EXE EXAMPLE
A)DEBUG EXAMPLE
-r
AX=0000 BX=0000 CX=475C DX=0000 SP=00EE
DS=5854 ES=5854 SS=5854 CS=5854 IP=0100
5854:0100 4D      DEC BP
-S 100 485C D7 50 B4 00        注：检索范围上限为
5854:0A36                      CX+100h
-E A36 90
-W
Writing 475C bytes
-Q
A)REN EXAMPLE EXAMPLE.EXE
```

硬盘的加锁和解锁

大庆石油学院　　张文

为保护硬盘中的数据和文件的安全，人们想出了很多加密方法。最简单而又安全可靠的是锁住硬盘，使非法者不能进入，而合法人员只要解锁即可使用。

本文给出的程序，是通过修改 DOS 分区表中第 5 字节（操作系统代号），来实现硬盘加锁解锁的。将操作系统代号改为 00H（表示无操作系统），即可锁住硬盘。解锁时，将其改为 01H（12 位 FAT 的硬盘）或 04H（16 位 FAT 的硬盘）即可。

程序中 MOV Buff＋1C2H，0H 的 1C2H 表示 DOS 分区表中操作系统代号偏移地址，它也可能是 1D2H、1E2H 或 1F2H（请查看所用硬盘分区表）。解锁程序中 MOV buff＋1c2H，04H 的 1c2H，也可能为 1D2H，1E2H 或 1F2H。

1. 加锁：运行加锁程序。当再次启动计算机时，就进入不了硬盘。

2. 解锁：用软盘启动，运行解锁程序，然后热启动，就能使用硬盘了。

1. 加锁程序

```
code segment
    assume cs:code,cs:code,ss:code,es:code
    org 100H
begin:JMP start
buff DB 200H dup(?)
start:MOV AX,CS
      MOV DS,AX
      MOV ES,AX
      MOV SS,AX
```

```
        MOV DX,80H
        MOV CX,1
        MOV BX,offset buff
        MOV AX,201 H
        INT 13H
        MOV buff+1C2H,0
        MOV DX,80H
        MOV CX,1
        MOV BX,offset buff
        MOV AX,301H
        INT 13H
        RET
code ends
     end begin
```

2. 解锁程序

```
code segment
      assume cs:code,ds:code,ss:code,es:code
      org 100H
begin:JMP Start
buff DB 200H dup(' ')
start:MOV AX,CS
      MOV DS,AX
      MOV ES,AX
      MOV SS,AX
      MOV DX,80H
      MOV CX,1
      MOV BX,offest buff
      MOV AX,201H
      INT 13H
      MOV buff+1C2H,04H
      MOV DX,80H
      MOV CX,1
      MOV BX,offset buff
      MOV AX,301H
      INT 13H
      RET
code ends
      end begin
```

键控图形处理程序

徐维科

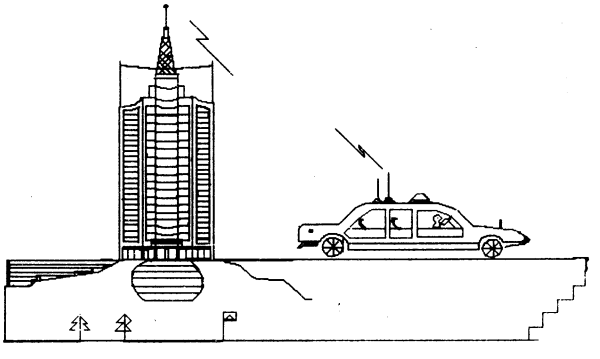
本程序用 BASIC 语言编写，它采用菜单与键盘控制相结合，可在屏幕任意位置进行彩色绘图，改图，并能将绘制好的图形存盘、或调出重新修改、或打印出来。程序运行的开始及各功能调用结束均返回菜单。操作非常简单方便。

在 GWBASIC 下运行此程序后，先进入菜单提示区。菜单中有下列功能供选择。选择"1"进入全屏幕绘图状态。程序定义了键盘上的十三个键为绘图专用键，按其它键均无效。小键盘上周围的数字键为方向键，它可使光点向相应方向运动。Ins 键控制画点或不画点。DEL 键控制是否在屏幕左上角显示光点坐标植。按下 C 键可调换八种颜色。Q 键放弃图形编辑返回菜单状态。E 键是存储键，按下此键后在屏幕左上角提示输入存盘文件名，作出响应后应按回车键。长城 0520C-H 型机的彩色显示缓冲区分红，绿，蓝三个区。所以图象存盘要首先定义文本寄存器为图形区，再分别对三种颜色区的象素以二进制文件的形式分别存入三个文件中。当在菜单提示区中选择"2"则显示"请输入图形文件名?"，输入完文件名后按回车键。本程序采用陷阱功能，防止因输入不存在的文件名而使程序中断运行。图形调入后的修改方法同"1"中所述。当选择"3"或者"4"同样要求输入文件名，若文件存在则显示或打印。当处于显示状态时，按任意键则返回菜单。打印图形时，屏幕硬拷贝结束后自动返回菜单。

用本程序建立的图形文件所占的存储空间比较大，每幅图需 144K 的存储空间，所以最好用硬盘操作。对于只要求单一颜色的图形可采用一个色区对其操作。这样每幅图只要 48K 的存储空间，也提高了存取速度。对程序的参数稍加修改即可用于 IBM-PC/XT 型机及兼容机。本程序在长城 0520C-H 型计算机和紫金 3070 型打印机上通过。

```
10 KEY OFF:CLS:DEFINT A-Z:SCREEN 2:ON ERROR GOTO 500
20 K$=SPACE$(35)
30 COLOR 7,0:CLS:LINE (0,0)-(640,450),3,BF
40 LINE (164,39)-(466,321),2,BF
50 LOCATE 4,30 :PRINT "键盘图形处理程序"
70 LOCATE 7 ,30:PRINT "1 绘制新图形"
80 LOCATE 9, 30:PRINT "2 修改图形"
90 LOCATE 11,30:PRINT "3 显示图形"
100 LOCATE 13,30:PRINT "4 打印图形"
110 LOCATE 15,30:PRINT "5 退出"
130 LOCATE 19,24:PRINT "请选择:";
140 A$=INPUT$(1):LOCATE 19,45:PRINT A$
150 IF A$="1" THEN CLS:GOTO 230
160 IF A$="5" THEN COLOR 7,0:CLS:KEY ON:GOTO 520
170 IF VAL(A$)<1 OR VAL(A$)>5 THEN 50
180 LOCATE 21,24:INPUT "请输入图形文件名";B$:CLS
190 DEF SEG=&HB800:OUT &H3DD,2:OUT &H3DE,1:BLOAD B$+".DTB",0
200 OUT &H3DE,3:BLOAD B$+".DTG",0:OUT &H3DE,5:BLOAD B$+".DTR",0
210 IF A$="3" THEN C$=INPUT$(1) :GOTO 10
220 IF A$="4" THEN 480
230 X=320:Y=225:Q=0:E=1:C=3:M=7:N=0
240 D$=INKEY$ :K=VAL(D$):IF K>0 AND K<4 THEN C=1
```

```
250 IF D$="C" OR D$="c" THEN C=C+1
260 IF C>7 THEN C=1
270 IF D$="E" OR D$="e" THEN 400
280 IF D$="Q" OR D$="q" THEN 10
290 IF LEN(D$)<>2 THEN 370
300 K=ASC (RIGHT$(D$,1))
310 IF K>70 AND K<74 THEN Y=Y-1
320 IF K>78 AND K<82 THEN Y=Y+1
330 IF K=71 OR K=75 OR K=79 THEN X=X-1
340 IF K=73 OR K=77 OR K=81 THEN X=X+1
350 IF K=82 THEN SWAP E,Q
360 IF K=83 THEN SWAP N,M:COLOR M,0
370 PSET(X,Y),C:LOCATE 1,1:PRINT X;Y
380 IF E=0 THEN PRESET(X,Y)
390 GOTO 240
400 COLOR 7,0
410 LOCATE 1,1:INPUT "输入存盘文件名";E$
420 LOCATE 1,1:PRINT K$
430 DEF SEG=&HB800:OUT &H3DD,2
440 OUT &H3DE,1:BSAVE E$+".DTB",49152!
450 OUT &H3DE,3:BSAVE E$+".DTG",0,49152!
460 OUT &H3DE,5:BSAVE E$+".DTR",0,49152!
470 GOTO 30
480 DEF SEG=&H7000:POKE 0,&HCD:POKE 1,&H5:
POKE 2,&HCB
490 A=0:CALL A:GOTO 30
500 IF ERR=53 THEN LOCATE 11,30:PRINT"***
文件没找到***"
510 FOR A=0 TO 400 :B=1:NEXT A:RESUME 30
520 ON ERROR GOTO 0:END
```



彩色的高分辨率图形

四川省自贡市川西南矿区劳动服务公司
潘麟元

通常情况下，用 SCREEN 2 进入 BASICA 的高分辨率作图方式，画面是黑白的。但我们知道，CCDOS 中的汉字就是在高分辨率图形方式下显示的，并且用 CTRL 和 F7 键可以改变屏幕颜色。那么，我们不进入 CCDOS，在西文 DOS 控制下能否实现彩色的高分辨率图形呢？答案是肯定的。

控制显示器的彩色图形适配板中，有一个彩色选择寄存器，它是一个 6 位的只写寄存器，它的输入输出端口地址是 &H3D9(十进制的 985)。我们可以用 BASIC 中的 OUT 语句来对它进行重写。在高分辨率状态下，我们只用到这个寄存器的低 4 位，d0～d3，其中 d0 控制蓝色，d1 控制绿色，d2 控制红色，d3 控制亮度。如果我们要在高分辨率状态下用红色画图，则色彩选择寄存器的低 4 位为 0100，即将十进制的 4 写入色彩选择寄存器即可达到目的。下面的程序就是在高分辨率状态下从左上角到右下角用红色画一斜线。

```
10 SCREEN 2:KEY OFF:CLS
20 OUT 985,4
30 LINE(0,0)-(639,199),1
40 WHILE INKEY$="":WEND
50 END
```

浙江温州市精密电子仪器经营部

地址：温州市飞霞南路龟湖路
邮编：325004
电挂：5072　　电话：336072
开户：工商五马辨　　帐号：1567892

供应(单价)EPROM2716/32/64/128 均 18 元；27256，25 元；RAM6116，15 元；6117，10 元；6246，32 元；8085/8031，15 元；Z80CPU，7.8 元；CTC/PIO 均 8.8 元；SIO，12 元；DMA，20 元；8039/8035/8255，均 10.5 元；8155，15 元；8243，12.5 元；8253，14 元；8279，22.5 元；8257/8259，18 元；8251，16.5 元；CD4508/4514/4515，4.8 元；光隔 P541－1/521－1，2 元；P541－2，3.5 元；P521－3，6 元；P521－4，7.5 元；40P 集成插座，0.65 元；晶振 2M，2.8 元；3.579545M，2 元；4M，3.2 元；2N3055，6.9 元 ；2SC1008，0.65 元；全桥 5A/50V，2.6 元；10A/50V，8.32 元；2μF/1KV，9 元，数字电压表 250V/电流表 5A，均 143 元。以上邮资 2 元/次，另有数字式实验多用仪，功能有(1)1MHZ 频率计(2)100PF～10000μF 电容计(3)三路可调稳压电源 1.2V～26V/1.5A 数显及干 5V～干 15V/1.5A，价 496 元，邮资 12.5 元/台，另有系列 74LS，CD4000 电主数百种其它晶体管，集成电路，继电器，阻容等均为进口器件，函索价目表者，请用贴足邮票信封与清回邮地址寄我部。

回溯法及其在编程中的应用

杭州第四中学　高三(3)　陈　磊

有相当多的问题；如迷宫问题，跳马问题以及求一些轨迹问题，并不能运用公式推导或采用循环、递归等方法求得它们的解。解答这类题目只能采用穷举、试探等方法，这就是我们所说的回溯法。

回溯法的基本思想是：为了求得问题的解，先选择某一种可能的情况向前搜索，在搜索的过程中，一旦发现原来的选择是错误的，就返回去加以修改，反复进行这种试探性的选择与返回纠错的过程，直到找到问题的解或者得出问题无解的判断为止。

回溯法是很重要的一种方法，在各类程序设计竞赛中常有它的一席之地，本文试图通过迷宫设计的例子介绍它的简单应用。

既然是个迷宫，当然希望有一定难度，不应该一眼就可以找到通路，最好只有唯一通路。此外，应该充分利用迷宫的面积；如果有一块地方是随便怎样走都走不到的，那等于浪费了一块区域。

古希腊神话传说中曾给出走迷宫的一个办法，即手拿着一个线团按下面的规律边走边放线：

(1)凡是走过的路都铺一条线；

(2)每到一个岔口先找没走过的路线；

(3)碰到死胡同，就在返回时铺上第2条线作为标记，以后凡遇到有两条线的路，就不走了。

这个办法可能要走许多弯路，但总能在有限步内走出迷宫。这种方法正是编程时采用的回溯过程。回溯算法是以试探搜索为优先的，即不考虑是否可以求解，先以搜索深度(即找到最终点)为主。

我们结合实例来看看程序的设计。我们把迷宫四周称为边墙，中央的墙称为隔墙，每一个小方格称为小间。好的迷宫应具有如下特征：

1. 每个小间的四个方向中至少有一堵隔墙。

2. 隔墙的每一处都通过其它隔墙与四个边墙之一相连。若几处隔墙不和任一边墙相通，它们必然形成“孤岛”，其结果是在这里出现两条通路。

另外，隔墙的任一处只与四周边墙中的一面相连，如果与两面边墙相连，则将阻断所有从左下至右上角的通路，或是将迷宫面积中的一块游离出去。

根据上面的分析，可以用下面的算法生成合乎条件的迷宫：首先假设迷宫范围内所有的隔墙存在，从迷宫左下角一间出发随机地选一个方向，前进一格，然后要判断两条，一是是否出了边墙，二是前方的这一间是否来过。如果两个回答均为否，就撤除这次穿过的那一小堵隔墙，并在进入的小间内留下“到此一游”的标志，如果两个回答有一成立(不可能两者皆成立)，就换个方向试试。倘若四个方向都不行，就以退为进，退到左下方一间再走，如果这一间都不行，再退到第二间……直到每个小间(共N^2间)都写上标记为止。

程序变量说明：

A(I)　$1\leqslant I\leqslant 2*N*(N+1)$。表示各间与各墙之编号。排列自左到右，自上而下。A(I)=0表明对应墙存在；A(I)=1，对应墙被撤除。

B(I,J)　I,J分别为小间行、列数。B(I,J)=0，此间未走到；B(I,J)=1此间已去过。

C(I),D(I)　表示第I种前进方向，$0\leqslant I\leqslant 6$，I=0、4表示向上；I=1或5向下；I=2、6向左；I=3向右。

U(L),V(L)当前所在小间坐标。

X、Y　前方欲进入一间坐标。

由于采用回溯法，速度不免受到影响，因此在设计时绘制了表示设置情况的指针。N为格数，N越大所需时间越长。

综上所述，利用回溯法必须弄清楚

1. 确定搜索的初始状态，即开始状态。

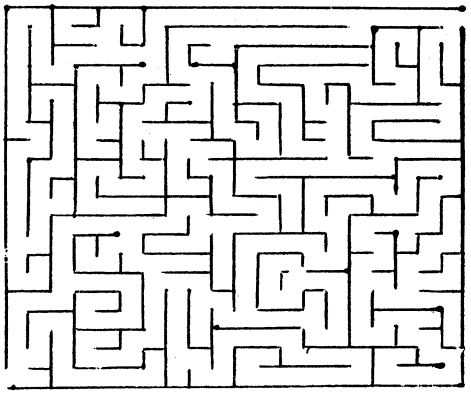
2. 当搜索到T步时，第T+1步的各种可能的选择将如何表示。

3. 确定回溯的条件是什么？并且回溯到前一步之后，如何做另一种选择。

4. 确定有解或无解的条件及如何求出所有解。

```
10 TEXT:HOME:VTAB 12:HTAB 14:PRINT "NUMBER-:";:INPUT "";N
15 IF N < 2 OR N>22 THEN PRINT CHR$ (7):GOTO 10
20 HGR2:HCOLOR=3:FOR I=0 TO 6:READ C(I),D(I):NEXT
30 DATA 0,-1,0,1,-1,0,1,0,0,-1,0,1,-1,0
40 M=N+1:P=M+N:Q=P/2:R=N * N:S=2 * M * N
45 GOSUB 300
50 DIM A(S),B(M,M),U(R),V(R)
60 B(N,1)=1:U(0)=N:V(0)=1
70 I=U(L):J=V(L)
80 E=INT (4 * RND (1))
90 FOR H=E TO E+3:X=I+C(H):Y=J+D(H)
100 IF X< =0 OR X>=M OR Y<= 0 OR Y > =M OR B(X,Y)< > 0 THEN 140
110 CO%=0: GOSUB 400:K=K+1:DW%=279/R * K:CO%=3:GOSUB 400
120 U(K)=X:V(K) =Y:B(X,Y)=1
130 A(P * I+J+Q * C(H) +D(H)/2-Q)=1:L=K:T=0:GOTO 70
140 NEXT
150 F=F+T:T=1:L =F
```

```
160 IF K<R -1 THEN 70
200 HGR2:HCOLOR=3:A(P)=1:A(S-P+1)=1
210 FOR I=1 TO S
220 IF A(I)< > 0 THEN 260
230 Y=INT ((I-1)/P)
240 X=I-P * Y:K=X>N:X=X-N * K
250 HPLOT 8 * (X+1-K),8 * (Y+K) TO 8 * X,8
* Y
260 NEXT:GET A$
270 TEXT:HOME:VTAB 12:PRINT "RUN AGAIN ? (Y/
N)";
280 VTAB 12:HTAB 34:GET A$:IF A$< >"Y" AND
A$ < > "N" THEN PRINT CHR$ (7):GOTO 280
290 IF A$ ="Y" THEN RUN
295 HOME:END
300 HPLOT 1,100 TO 278,100
310 FOR II=4 TO 274 STEP 30
320 HPLOT II,101 TO II,105
330 NEXT II
340 RETURN
400 IF DW% < 5 OR DW% >274 THEN 450
410 HCOLOR=CO%
420 HPLOT DW%,90 TO DW%,99
430 HPLOT DW% -3,97 TO DW%,99
440 HPLOT DW% +3,97 TO DW%,99
450 RETURN
```



中华学习机使用技巧

北京一零一中　任义

一、CATALOG 功能的改进：

使用 CATALOG 命令时，一次可显示 18 个文件名，如果盘中文件超过 18 个，则按任意键可继续列目录。但这样我们有时会忘记以前列过的目标文件名(特别是文件名较长时)。如果对 DOS 系统做些修改，便可使 CATALOG 命令的功能有所增强，即找到目标文件名后按回车键，系统进入 BASIC 状态，停止列目录，如按其他键则继续列目录。

```
300:20 E3 03 20 D9 03 A9 00 85 48 60
B7EB:00 01 0D
B7F4:01
B7F0:00 20
300G
2039:4C DF BC
B7F4:02
300G
B7EB:00 00 06
B7F4:01
300G
20DF:20 0C FD C9 8D D0 03 4C 2C AE 4C 3C AE
B7F4:02
300G
```

二、巧编音乐程序：

初学编音乐程序的人，往往对音阶、音长代码不熟悉而深感不便。现在向您介绍一种比较方便的方法：先定义两个数组 A(15)，B(5)，用 A 数组存放音阶代码，用 B 数组存放音长代码，然后 READ 出两个数组的下标，最后 MUSIC A(X)，B(Y)就可以了。程序如下：

```
10 DIM A(15),B(5)
20 FOR I=1 TO 5:READ B(I):NEXT:DATA 30,70,
110,160,255
30 FOR I=0 TO 15:READ A(I):NEXT:DATA 1,192,
171,152,140,128,114,102,255,228,205,95,84,75,68,62
40 READ X,Y:IF X<0 THEN END
50 MUSIC A(X),B(Y):GOTO 40
60 DATA 音阶下标(0-15)，音长下标(1-5)，音阶下标，
音长下标，……，-1，-1
```

三、DOS 失灵后……

(一)在中华学习机上按复位键后会使 DOS 命令失灵，这时我们只须在 BASIC 状态下键入 CALL 1002 或在监控状态下键入 3EAG 即可恢复。

(二)LISA 使用者在 DOS 失灵后，可用 BRK 进入监控，记住 $1800 的内含值，重新启动 LISA 系统：C600G，再将 $1800 的值存回，这样程序就不会被清掉。

四、其它

1. 在 LOGO 状态下按复位键进入监控，这时如果键入 CTRL-Y 回车就会回到 LOGO 状态。

2. 有时需要通过 GET 命令进入中文，这样要先有一个空打印，但这会破坏西文格式(屏幕上卷)，这时用 PR # 3:CALL 1002 进入中文，既进入了中文系统，又能继续使用 DOS 命令。

3. 从中文系统直接进入低分作图状态，如果您想进入全屏幕作图状态，用 POKE－16298,0:POKE－16302,0:POKE－16300,0:CALL－1998。如果您想进入图形、文本混合状态时用 POKE－16298,0:POKE－16301,0:POKE－16300,0:CALL－1994。

实用改良的 NEW CODE-POKES 程序

黄影昌

当我们在 CEC-I 或 APPLE 机上行进磁盘操作时，有时很想把一个 B 类文件转换成 A 类文件而用 CODE-POKES 程序(很多有关书籍都有介绍)来进行转换。但我在大量实践中发现，该程序最大的缺点是一个本来就不长的文件一经转换就成了一个很长的 A 类文件，因此占用较多的磁盘空间。针对该程序的缺点，我自己试着改良 CODE-POKES 程序，开发了一个新程序：NEW CODE-POKES，经过实验，效果挺不错。下面是该程序的几点说明：

20 句，输入转换后文件的文件名。

30 句，输入 B 类文件的首地址。

40 句，输入 B 类文件的末地址。

50 句，输入转换后 A 类文件的起始行号。

55－190 句，是本程序的转换部分。

具体的操作过程如下：

1. 调入需转换的 B 类文件，查出其首，末地址。

首地址：PRINT PEEK (43635) * 256 + PEEK (43634)↙

末地址：PRINT PEEK (43635) * 256 + PEEK (43634)+PEEK(43617) * 256+PEEK(43616)－1↙

2. 运行 NEW CODE-POKES，光标出现后，再 EXEC 文件名↙

待光标再现后，内存中就是转换后的 BASIC 程序了，此时你可将它存入磁盘中。

```
10 D$ = CHR$ (4) :C=1
20 TEXT:HOME:VTAB 4:INPUT "FILENAME:";N$
30 INPUT "SHOU DI ZHI:";A
40 INPUT "MO DI ZHI:";B
50 INPUT "QI SHI HANG HAO:";LN
55 PRINT D$ "OPEN";N$
60 PRINT D$ "DELETE";N$
70 PRINT D$ "OPEN";N$
75 PRINT D$ "WRITE";N$
80 FOR P=A TO B
100 IF C=61 THEN P=P+1:C=1
110 IF C<> 1 THEN 150
120 PRINT:PRINT LN;"DATA";
130 LN=LN+1
150 PRINT PEEK (P);:C=C+1:IF C=61 THEN 100
160 IF P<B THEN PRINT",";
170 NEXT P
175 PRINT
180 PRINT LN + 1;"FORI=";A;"TO";B;":READC:
POKEI,C:NEXTI"
190 PRINT D$ "CLOSE";N$
200 DEL 10,200
```

巧解"八皇后"问题

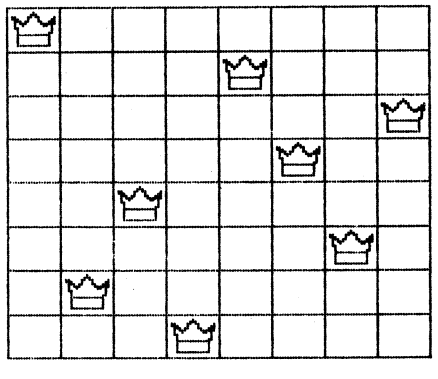
南京师范大学附中高一(1)　柳杨

"八皇后"问题是伟大的数学家高斯提出来的。它要求在国际象棋盘中摆下的八个皇后互不相吃。

本程序求解"八皇后"问题，并找出所有 92 种解答。程序运行过程中还配有生动的移子画面。

附程序清单和一种解答的拷贝。

```
5 DIM C(8),T(8),S(15),P(15):D=1
10 HOME:HGR2:HCOLOR=3
20 FOR I=0 TO 160 STEP 20:HPLOT I+60,0 TO I+60,160:HPLOT 60,I TO 220,I:NEXT
30 K=1
40 IF T(K)=1 THEN 80
50 IF S(8+D-K)=1 THEN 80
60 IF P(K+D-1)=1 THEN 80
70 C(D)=K:T(K)=1:S(8+D-K)=1:P(K+D-1)=1:CO=3:GOSUB 600:D=D+1:IF D>8 THEN TA=TA+1:GET A$ :GOTO 90
75 GOTO 30
80 K=K+1:IF K<=8 THEN 40
90 D=D-1:IF D<1 THEN TEXT:PRINT "TOTAL=";TA:PRINT" * * * END * * * ":END
100 K=C(D):CO=0:GOSUB 600:T(K)=0:S(8+D-K)=0:P(K+D-1)=0: GOTO 80
600 X=64+(K-1) * 20:Y=12+(D-1) * 20
610 HCOLOR=CO
620 HPLOT X,Y TO X+12,Y TO X+12,Y+5 TO X,Y+5 TO X,Y TO X-2,Y-8 TO X+2,Y-4 TO X+6,Y-10 TO X+10,Y-4 TO X+14,Y-8 TO X+12,Y
640 RETURN
```



关于 APPLE 和 CEC-I 机数值计算误差的讨论

对《APPLE 和 CEC-I 设计者的疏忽》一文的看法

成都人防电器设备厂　　刘峰

贵刊 90 年 8 期发表了刘光明的《APPLE 和 CEC-I 设计者的疏忽》一文，作者在打印数字“8.13”时产生了误差。这种误差常常是难以避免的，它并非是设计者的疏忽，而是由于计算机内采用二进制数，而高级语言均使用十进制数，所以程序和数据输入计算机就有一个数制转换的问题。一般的整数只要不超过机器允许的长度，在数制转换时不会发生误差，但是小数就不一样了，小数的数制转换多数只能取近似值，比如十进制小数 0.1 化为二进制小数是 0.00011001100……。这是一个无限循环小数，必须经取舍后成为近似值。其次，计算机用固定的字节数存放每一个实型数，而在输出时又仅仅以前几位有效数字的形式打印出来，这常常使得二者不可能完全一致。APPLESOFT 以 5 个字节存放一个实型数，输出最多 9 位有效数字，因此在两数相减时，只要差的有效数字不足 9 位，内存中的隐含误差部分便向前递补，凑足 9 位数字输出。当然，并非所有实型数在数制转换时都会发生误差。当其小数部分为 0.5 的正整数次幂时，数制转换时不会发生误差。另外，误差极小时，机器也会忽略不计。

其实，乘方与开方等运算均可导致误差，特殊情况下误差还相当大。当相差很大的两个整数相加减时，大数会“吃掉”小数。例如计算机计算“2.54E9+5”时，答案还是 2.54E9。运算乘方与开方时所产生的误差是由于计算机的系统软件采用与人工计算完全不同的近似算法（迭代逼近法）所引起的，而相差很大的两个整数相加减时则是因为两个数的阶码值相差较大引起的，如把第一个数换为定点数，情况就不会一样了。

很明显，无论是实型小数的数制转换引起的误差，还是整实数的近似计算误差，都会给运算带来不良影响，甚至使程序无法继续运行。那么，进行实型小数的计算和乘方开方的计算，都要设法消除误差。一般的方法是测出原始数据的小数位数中最长的一个（用对数和字符串处理），在输出前利用 INT(X)函数实现必要的舍入。其次是在用小数作累加时，可利用整数取代小数作为循环变量值并增加运算，这样误差值可显著降低。对于避免等值比较时逻辑值为零的情况判断失误，可采用限定误差范围的方法，如用 IF INT(SQR(X))−SQR(X)<1E−6 THEN PRINT X 可顺利地判断 X 是否为完全平方数。例如打印“8.13”时，可采用 PRINT (INT(8.13*100)/100)或 PRINT “8.13”等。

APPLE 缺陷的解决办法

福建邵武市闽北地质队　　陈盛旋

产生类似问题的数还有 8.104、−8.13 等等，这个问题虽无大错，却破坏了正确的打印格式、给编程者造成不大不小的麻烦。一个简易的解决办法，是在打印输出之前对数据进行四舍五入和小数点截位处理。具体公式如下：

A=−INT(−1*A*1E2+0.5)/1E2

在打印变量 A 之前，先进行上述公式计算，即可将 A 的小数点经四舍五入后只保留两位，即使 8.13000001 也将打印成 8.13 出来。这就解决了刘光明同志亟待解决的问题。

这个公式对正数负数和零均有效，因而也可用于普遍的数据四舍五入处理。如果要保留三位小数或四位小数，只需将公式中的 1E2 改成 1E3 或 1E4 即可。这一公式解决了 APPLE 机没有四舍五入语句的缺陷。

附：示范程序：

```
10 A=8.13
20 PRINT“A=”;A
30 A=−1NT(−1*A*1E2+0.5)/1E2
40 PRINT “A=”;A
50 END

JRUN↙
A=8.13000001(原来的 A 值)
A=8.13(经过处理的 A 值)
```

数值计算中存在问题的解决办法

吉林市水泵厂　　朱洪波

笔者在实践中，找出了解决问题的简单办法。只要在程序中增加一条语句：

```
10 A=8.13
20 A=INT(A*100)/100
30 PRINT A
RUN
8.13
```

在下面的计算中也会出现类似现象

```
10 M=60.07:N=4
20 PRINT M+N
```

```
RUN
60.0700001
```

解决办法也跟前面一样，增加下面一条语句：

```
P=INT((M+N)*100)/100
```

经过上述处理后，就不会出现错误的结果了。

笔者在实际工作中，发现 APPLE 及其兼容机还存在下面问题，即在小数部分取舍时，出现错误的计算结果。例：

```
10 P=34.555
20 PRINT(P*100+0.5)/100
RUN
34.55
```

打印的结果应该是 34.56，而经过上述计算却是 34.55。这种现象的出现，给软件的编制带来一定的麻烦，特别是在财务管理中，更是不允许出现。笔者曾看到用下列方法来解决，例如：

```
10 P=0.525
10 PRINT INT(P*100+0.5000001)/100
RUN
0.53
```

但此方法没有从根本上解决小数的取舍问题。例：

```
10 P=34.555
20 PRINT INT(P*100+0.5000001)/100
RUN
34.55
```

显然结果还是不对。笔者在编制程序中使用下面的改进方法，取得了较好的效果，该程序消除了一切在保留小数中出现的问题，从根本上解决了小数取舍时产生的错误。程序清单如下：

```
10 INPUT"A,N=";A,N
20 GOSUB100:PRINTA:GOTO10
100 A$=STR$(A):L=LEN(A$)
110 FORI=1 TO L
120 IF MID$(A$,1,1)="."THEN
I1=I:GOTO 130
125 GOTO 140
130 I=I1+N+1:A1=VAL(MID$(A$,1,I))+5/10^(N+1):A=VAL(MID$(STR$(A1),1,I-1)):RETURN
140 NEXT:RETURN
```

在上面的程序中，A 为要取舍的小数，N 为保留小数的位数，120 语句是测量在第几位出现小数点，130 语句是根据测得的位数计算出要取舍的结果。例：

```
RUN
A,N=? 34.555,2
34.56
A,N=? 0.525,2
0.53
A,N=? 0.5265,3
0.527
```

该程序虽然较长，但是只要将 100～140 语句作为一个子程序使用就比较方便了，例：

```
1^A=X:GOSUB100:X=A
```

可以看到，上面的例子是非常简单的，对于一个或多个数组变量也能方便地解决如：

```
10 FOR I=1 TO 30:A=A(I):GOSUB100:
A(I)=A:A=B(I):GOSUB100:B(I)=A:
NEXT
```

前面提出的问题，是 APPLE 及其兼容机上固有的。对于现有的机器，要想使之运行出准确的结果，只有从编程上采取措施。这样，苹果机还是能在各行各业中发挥作用的。

无DOS磁盘格式化程序

呼和浩特市第一中学九二(3)班　姜宏岩

当我们使用 DOS 的 INIT 命令对磁盘初始化时，命令的处理过程分为以下四个步骤：

(1)格式化磁盘。将磁盘划分为 $0～$22 共 35 条磁道，每条磁道有 $0～$F 共 16 个扇区的格式。

(2)把 $11 磁道的 $0 扇区定义为盘卷内容表，或称 VTOC 表。VTOC 表上记载了磁盘中每个扇区是“空”还是“被使用”的状态。将 $11 磁道的其余扇区 $F～$1 定义为供磁盘文件目录使用，然后把磁道 $0～$2 及 $11 在 VTOC 表中置为“被使用”。

(3)从内存中把 DOS 代码写入 $0～$2 磁道。

(4)把内存中的 BASIC 程序存入磁盘，程序名要在 INIT 命令中写明(一般起名 HELLO)。

因此，在已初始化的磁盘中，DOS 占了 3 个磁道，VTOC 表和文件目录共占 1 个磁道，再扣除 HELLO 程序所占的空间后，实际剩下的空间就更少了。事实上，有时并不需要每个磁盘都存放 DOS 代码，尤其是一些用于存放数据的磁盘。下面所提供的程序，运行后将修改内存中 DOS 的 INIT 命令程序，只对磁盘进行格式化，不写入 DOS 及 HELLO 程序，而将 $1、$2 磁道释放出来给用户使用，$0 磁道由于有其它原因仍然不能使用。

```
100 TEXT:HOME:VTAB 3
110 INVERSE:HTAB 8
120 PRINT "DATA DISK FORMATTER"
130 NORMAL:PRINT
140 POKE 42347,96:POKE 44723,4
160 POKE 44799,76:POKE 44800,127:POKE 44801,179
170 PRINT:PRINT
180 PRINT"INSERT DISKETTE TO BE INITIALLIZED"
190 PRINT"PRESS [RETURN] TO PROCEED"
200 PRINT "[ESC] TO EXIT"
```

```
210 PRINT "BEGIN(Y/N)?"
220 GET A$
230 IF A$ ="Y" THEN 250
240 IF A$ ="N" THEN 260
250 PRINT CHR$(4);"INIT XXX,254"
260 POKE 42347,76:POKE 44723,12:POKE 44799,32:
POKE 44800,194:POKE 44801,183:END
```

跟踪机器语言程序的运行

天津 武清 杨村一中 田俊成

苹果机在监控状态下没有跟踪运行命令，这给调试机器语言程序带来一定程度的不便。为此我们用6502机器语言编了一段程序，在监控状态下跟踪机器语言程序的运行，可以观察机器语言的执行过程，特别对程序出现死循环等问题，可以很快找到。

使用时，首先在监控下键入 3F9:00 10，然后每次调用只要键入：〔被执行程序首地址〕 CTRL-Y↙则从指定地址开始每执行完一条指令后，显示该指令。在执行过程中用 CTRL-S 使之暂停，再次使用 CTRL-S 则继续执行，如此可停下来查看当前提示。

```
1000— 20 75 FE A9 B1 85 48 A9
1008- EA 8D CA 10 8D CB 10 20
1010- D0 F8 A0 00 B1 3A C9 10
1018- F0 2F C9 30 F0 2B C9 50
1020- F0 27 C9 70 F0 23 C9 90
1028- F0 1F C9 B0 F0 1B C9 D0
1030- F0 17 C9 F0 F0 13 C9 4C
1038- F0 1A C9 6C F0 27 C9 20
1040- F0 3B C9 60 F0 57 4C BB
1048- 10 8D C9 10 A9 15 8D CA
1050- 10 4C C6 10 C8 B1 3A 85
1058- 3C C8 B1 3A 85 3B A5 3C
1060- 85 3A 4C 07 10 C8 B1 3A
1068- 85 3C C8 B1 3A 85 3D A0
1070- 00 B1 3C 85 3A C8 B1 3C
1078- 85 3B 4C 07 10 18 A5 3A
1080- 69 02 85 3C A5 3B 69 00
1088- 48 A5 3C 48 C8 B1 3A 85
1090- 3C C8 B1 3A 85 3B A5 3C
1098- 85 3A 4C 07 10 68 85 3A
10A0- 68 85 3B E6 3A D0 02 E6
10A8- 3B A5 3A C9 85 D0 09 A5
10B0- 3B C9 FF D0 03 4C 85 FF
10B8- 4C 07 10 88 C8 B1 3A 99
10C0- C9 10 C4 2F D0 F6 20 3F
10C8- FF AE EA EA 08 85 45 86
10D0- 46 84 47 68 85 48 20 53
10D8- F9 85 3A 84 3B 4C 07 10
10E0- E6 3A D0 02 E6 3B A0 00
10E8- B1 3A 85 2F 20 53 F9 85
10F0- 3A 84 3B 4C 07 10
```

局部清屏程序

解放军后勤工程学院 程华

用户在编制绘图程序时，时常要清除部分图形而保留另一部分图形。遇到这类情况一般是用 BASIC 作图语句逐点清除，运行速度受到限制，不利于实时应用。本文提供一段 6502 机器语言子程序，能方便的实现局部清屏，清屏面积的大小可由用户在指定单元内输入数据来实现。它可完成小到一个光点，大到整个屏幕的清除。此程序在 APPLE Ⅱ 及其兼容机中均可使用。现将使用方法简介如下：

由于本程序中各种转移指令均采用相对寻址，所以程序可根据需要置于内存任何位置。

应用此程序时如果在第一图形页清屏，需将 $20（等于十进制 32，以下用 L 表示十进制），第二页为 $40(L64)置于 $E6 单元(L230)。然后将需清屏的起始水平坐标值高位置于 $FA(L250)，低位置于 $FB(L251)。所谓高位是指水平坐标值除以 256 时商的整数部分，余数即为低位值。将终点水平坐标值高位置于 $FC(L252)，低位置于 $FD(L253)，将起点的垂直坐标值置于 $FE(L254)，终点的垂直坐标值置于 $FF(L255)。完成后，在 BASIC 程序中，可用 CALL768 调用此程序；在汇编语言中可用 JSR $300 调用。

请注意：坐标原点设在屏幕的左上角(0,0)。应用时起点坐标值必须小于或等于终点坐标值。另外终点坐标值不要大于图形页的尺寸，即水平不大于 279，垂直不大于 191。下面举例子说明程序的用法：

如果已定义好图形页并已调入一图形，现想清除中间一部分，其水平坐标起点是 100，终点是 170，垂直坐标起点是 50，终点是 85。可编制下列 BASIC 程序：

```
1000 POKE 250,0:POKE 251,100:POKE 252,0:
POKE 253,170:POKE 254,50:POKE 255,85:CALL
768
```

运行 1000 句后可看到，所要求清除部分的图形已被清除，而其余部分完好无损。

```
0300- A5 E4 48 A2 00 86 E4 E6
0308- FF A5 FE 85 F2 A2 00 B5
0310- FA 95 06 E8 E0 04 90 F7
0318- A6 07 A4 06 A5 F2 20 57
0320- F4 A5 06 C5 08 90 14 A5
0328- 07 C5 09 90 0E E6 F2 A5
0330- F2 C5 FF 90 D8 C6 FF 68
0338- 85 E4 60 E6 07 D0 D9 E6
0340- 06 18 90 D4
```

扩展 BLOAD 命令功能

重庆大学　蒋阳

在 DOSS3・3 磁盘操作系统中，有一个 BLOAD 命令，其功能是将存于磁盘上的二进制(B 类)文件的内容调入内存。对一个不知其功能或加密的 B 类文件，我们为了读懂或解密，首先必须知道它存放于内存的首地址、长度或者末地址，而 BLOAD 命令不能直接提供给用户。为此本人对 BLOAD 命令作了改进，当用户使用改进后的 BLOAD 命令调 B 类文件时，既完成原 BLOAD 的功能，同时自动清屏，并在屏幕的第一行上从左至右显示出文件放于内存的首地址、长度、末地址。

```
5 HOME
10 FOR I=48351 TO 48369:READ A:POKE I,A:NEXT
13 FOR I=46771 TO 46841:READ B:POKE I,B:NEXT
15 FOR I=47721 TO 47762:READ C:POKE I,C:NEXT
20 PROKE 41861,76:POKE 41862,179 :POKE 41863,182
30 DATA 165,7,233,0,32,218,253,165,6,32,218,253,40,174,114,170,76,136,163
40 DATA 8,32,88,252,169,211,32,240,253,169,189,32,240,253,169,164,32,240,253,173,115,170,32,218,253,173,114,170,32,218,253,169,160,32,240,253,169,204,32,240,253,169,189
45 DATA 32,240,253,169,164,32,240,253,173,97,170,32,218,253,173,96,170,32,218,253,169,160,32,240,253,76,105,186
60 DATA 169,197,32,240,253,169,189,32,240,253,169,164,32,240,253,24,173,114,170,109,96,170,133,6,173,115,170,109,97,170,133,7,56,165,6,233,1,133,6,76,223,188
```

从“重赏”谈起

山东金乡五中　卢保堂
中国人民公安大学　卢玉堂

印度有一个饶有趣味的故事。传说舍罕王打算重赏国际象棋的发明人——宰相西萨・班・达伊尔，国王问他有何要求，这位宰相说，只需在棋盘第一格放一粒麦子，第二格放两粒，第三格放四粒，这样依次加倍地放下去，直到放完第六十四格，把这些麦子都赏给他就可以了。国王认为这是区区赏金，微不足道，就满口答应，结果当然是绝对办不到的。如果我们估算一下，就会发现这个数字很大，可以用电脑帮助解决这个问题。程序一运行结果表明，此数是全世界两千年内生产的全部麦子。由于受计算机位数的限制，小麦粒数的后几位无法显示。程序二能精确地显示出小麦粒数。

程序和变量简单说明如下：

第 5 句：输入棋盘格数

第 6 句：计算总麦粒数的位数

第 10～80 句：清变量

第 100 句：赋初值

第 100～300 句：计算第 I 格内与前 I 格内总和的麦粒数

第 500～700 句：打印输出

S(I,J)表示第 I 格内麦粒数第 J 位的数字。

H(J)表示总麦粒数第 J 位的数字

SJ,JS 分别表示第 I 格内第 J 位的进位标志和进位

HJ,JH 分别表示总麦粒数第 J 位的进位标志和进位。

程序一

```
5 INPUT “QI PAN GE SHU?”;N
10 S=1:A =1:N=N-1
15 IF N=0 THEN 50
20 FOR I=1 TO N
30 A=2 * A:S =S+A
40 NEXT I
50 PRINT “S=”;S
60 END
]RUN
QI PAN GE SHU ? 64
S=1.84467441E+19
]RUN
QI PAN GE SHU ? 73
S=9.44473297E+21
```

程序二

```
5 INPUT“QI PAN GE SHU?”;N
6 N1=INT (LOG (2^ N)/LOG (10))
7 N=N-1
10 DIM S(N,N1),H(N1)
20 FOR I=0 TO N
30 FOR J=0 TO N1
40 S(I,J)=0
50 NEXT J:NEXT I
60 FOR J=0 TO N1
70 H(J)=0
80 NEXT J
100 S(0,0)=1:A=0:H(0)=1
105 IF N=0 THEN 500
110 FOR I=1 TO N
120 K=I-1
130 FOR J=0 TO N1
140 IF S(K,J)<5 THEN SJ =0:S0 =S(K,J):GOTO 160
150 SJ=1:S0 =S(K,J)-5
160 S(I,J)=S0 * 2+JS
170 IF SJ=0 THEN JS=0:GOTO 190
```

```
180 JS=1
190 NEXT J
200 FOR J=0 TO N1
210 H0=S(I,J)+H(J)
220 IF H0<10 THEN LET HJ=0:GOTO 250
230 HJ=1:H0=H0-10
250 H(J)=H0+JH
260 IF HJ=0 THEN LET JH=0:GOTO 280
270 JH=1
280 NEXT J
290 HJ=0:SJ=0
300 NEXT I
500 PRINT
550 PRINT "S=";
600 FOR J=N1 TO 0 STEP -1
610 PRINT H(J);
620 NEXT J
630 PRINT
700 END

]RUN
QI PAN GE SHU ? 64
S=18446744073709551615
]RUN
QI PAN GE SHU ? 73
S=9444732965739290427391
```

在 APPLESOFT 下使用小汇编

南京市五十中　嵇克威

大家都知道，在苹果机的整数 BASIC 下可以使用小汇编。只要机器从整数 BASIC 状态进入监控，再键入 F666G 即进入小汇编。小汇编系统短小精悍，简单实用，深为广大用户喜爱。可是目前苹果机多在 APPLESOFT BASIC 下工作，无法使用小汇编。为了解决这个问题，我将小汇编程序代码搬移到较低地址的 RAM 空间，对程序内部绝对寻址指令的高位地址作相应修改，具体步骤如下：

①引导 DOS3.3 主盘，载入 DOS 和 INTBASIC

②键入 INT 进入整数 BASIC；CALL－151 进入监控。

③在监控下键入：7500<F500.F63CM

④键入以下命令：

＊7537:75 N 755B:75 N 75BF:76 N 75DD:76N 75E7 :76 N 7633:75↙

⑤键入 CTRL-C 回到整数 BASIC，然后输入 FP 转到 APPLESOFT 状态。

现在可以在 APPLESOFT 下使用小汇编了。要用时，只要进入监控，再输入 7592G。为了更方便使用，可把 & 向量指向改造后的小汇编，即键入：3F5:4C 92 75↙

以后只要键入"&"即可进入小汇编。

⑥为了保存改造后的小汇编，先键入 74FD:4C 92 75↙然后存盘

BSAVE MINIASM,A$74FD,L$140

以后只要 BRUN MINIASM 就能进入小汇编。

工资票面张数计算程序的简化

浙江奉化一中　孙承灿

本刊 90 年第四期的《一个简单实用的工资票面张数计算程序》一文的程序，本人认为过于繁杂。其实，当一个单位 N 个人的应发工资数存放在数组 N(I)中时，只要在程序的开头加上第 5 行，在适当的位置插入一句 GOSUB 1000 即可。程序如下：

```
5 DIM Y(12),X(12):FOR I=1 TO 12:READ X(I):NEXT:DATA 10000,5000,1000,500,200,100,50,20,10,5,2,1
100 GOSUB 1000
:
:
1000 FOR I=1 TO N:A=N(I)*100:FOR J=1 TO 12:Y=INT (A/X(J)):Y(J)=Y(J)+Y:A=A-Y*X(J):NEXT:NEXT:RETURN
```

(上接 26 页)

(3)C 语言除了有优先级外，还有结合性问题。所谓结合性是指当表达式中同时出现多个优先级相同的运算符时的求解顺序。如果运算顺序是从左到右，叫左结合性；从右到左，叫右结合性。表中最右边一项，说明了各级运算符的结合性。

(4)C 语言中有 5 个运算符的结合性是可交换的。它们是＊、+、&、|、^。

(5)此外，注意 Turb。c 表达式中允许随意增加空格和使用多余的圆括号。

〔练习 4.15〕

```
main ()
    {  printf("%d\n",5>&&! (1<5)||2<=3);
       printf("%d,%d\n",5+3,3+5);
       printf("%d,%d\n",5|3,3|5);
    }
A>EXP4-15
1
8,8
7,7
```

第四讲　C 运算符

李文兵

运算符较多，是 C 语言的一个显著特点。那么，C 运算符究竟有多少，又如何使用，请看本讲。

1　算术、逻辑运算符

(1)算术运算符　　标准算术运算符见表 4.1。

表 4.1　　算术运算符

运算符	操　作	项　数	优先级
－	负	单项	1
*	乘	双项	2
/	除	双项	2
%	余	双项	2
＋	加	双项	3
－	减	双项	3

〔练习 4.1〕

```
A>type exp4_1.C
main()
    {   int i=2,j=3;
        printf("add=%d\n",i+j);
        printf("mod=%d\n",(i+j)%3);
    }
A>exp4_1
add=5
mod=2
```

该程序在(i+j)%3 中使用了运算符()，它是优先级别最高的运算符，常用它改变表达式的运算顺序。

(2)关系运算符　是用来判断两个值大小的运算符，共有六种，如表 4.2 所示。

表 4.2　　关系运算符

运算符	操　作	项　数	优先级
>	大于	双项	1
>=	大于或等于	双项	1
<	小于	双项	1
<=	小于或等于	双项	1
==	等于	双项	2
! =	不等于	双项	2

这些运算符的用法是：

操作数 1　　关系运算符　　操作数 2

该式成立时，运算结果为 1(代表真)；否则为 0(代表假)。注意，这里的 1 或 0 不是布尔量，是整型常数 1 或 0。C 语言里没有定义布尔量。

〔练习 4.2〕

```
C>TYPE EXP4_2C
main()
   {  int i=5,j=3,a=3,b=5;
      printf(" < <= > >= ==!=\n");
      printf(" %d %d %d %d %d %d\n",
      i<j,i<=j,i>j,i>=j,i==j, i!=j);
      printf(" %d %d %d %d %d %d\n",
      5<3,5<=3,5>3,5>=3,5==3,5!=
      3);
      printf("%d %d %d %d %d %d\n"
      a<b,a<=b,a>b,a>=b,a==b,a!=b);
   }
C>EXP4_2
 <   <=   >   >=   ==   !=
 0    0    1    1    0    1
 0    0    1    1    0    1
 1    1    0    0    0    1
```

(3)逻辑运算符　基本的逻辑运算符有逻辑与(AND)、逻辑或(OR)和逻辑非(NOT)。在 C 语言中其运算符号，如表 4.3 所示。

表 4.3　　逻辑运算符

运算符	操　作	项　数	优先级
&&	逻辑与	双项	2
||	逻辑或	双项	3
!	逻辑非	单项	1

C 语言无布尔量，故逻辑运算结果也用整型数 1 代表真，整型数 0 代表假，而对于操作数来说，若其值不是 0，为真；否则为假。

〔练习 4.3〕

```
A>type exp4_3.c
main()
      {  printf("%d\n",2&&1);
         printf("%d\n",2|| 3);
         printf("%d\n",! 2);
      }
A>exp4_3
1
1
0
```

(4)位逻辑运算符　有六种，如表 4.4 所示

表 4.4　位逻辑运算符

运算符	名　称	操　作	项　数	优先级
&	位积	按位与	双项	3
|	位和	按位或	双项	5
^	位差	按位异或	双项	4
~	位非	按位取反	双项	1
≪	左位移	按右值位数左移左值	双项	2
≫	右位移	按左值位数右移左值	双项	2

〔练习 4.4〕

```
C>type exp4_4.c
main()
        {   int x=7,y=5;
            printf("& | ^  ~  ≪ ≫\n");
            printf("%x %x %x %x %x %x",
                    x&y,x|y,x^y,~x,x≪y,x≫y);
        }
    c>exp4_4
 &   |   ^       ~       ≪     ≫
 5   7   2      fff8      e0     0
```

2. 递增、递减、赋值运算符

这三种运算符用在表达式中，使其具有副作用。

(1)递增与递减运算符　这两种运算符是 C 语言特有的，它们都是单项运算符，用法如下：

x++　等价于 x=x+1

x--　等价于 x=x-1

这两种运算符可放在操作数的前面，叫前置运算，也可放在操作数之后，叫后置运算。

对于前置运算，例如：

x=10;y=++x;

运算结果为：

x=11;y=11。

对于后置运算，例如：

x=10;y=x++;

运算结果为：

x=11;y=10。

〔练习 4.5〕

```
C>TYPE EXP4_5.C
main()
    {   int i=10,j;
        j=i;
        printf("   i\t j\n");
        printf("----\t----\n");
        printf(" %d\t %d\n",i++,++j);
        j=i++;
        printf("%d\t %d\n",i++,j++);
    }
C>EXP4_5
  i        j
----     ----
10        11
12        11
```

(2)赋值运算符　我们已经知道，C 语言的赋值运算符是"="，其用法是：

变量=表达式；

此外，在 C 语言中，假定 a 为变量，e 为表达式，op 为运算符，则表达式：

a=a op e;

可用如下表达式替代。

a op=e;

其中 op=叫做复合赋值运算符。op 可以是+、-、*、/、%、&、|、^、≪、≫，共 10 个。这样，赋值运算符又扩展了 10 个。

〔练习 4.6〕

```
C>TYPE EXP4_6.C
main()
    {   int a=100,b=15;
        a+=b;
        a-=b;
        a*=b;
        a/=b;
        a%=b;
        a≪=4;
        prnitf("shift left %d\n ",a);
        a≫=4;
        a&=b;
        a|=b;
        a^=0xffff;
        printf("exclusive or %d\n",a);
    }
C>EXP4_6
shift left 160
exclusive or -16
```

3　求字节数、条件运算、逗号

(1)sizeof 运算符　其功能是计算变量或类型的大小，其用法如练习 4.7 所示。

〔练习 4.7〕

```
C>TYPE EXP4_7.C
main()
    {   int a;
        int b[100];
        printf("1: sizeof int variable=%d\n",
                sizeof(a));
        printf("2: sizeof int=%d\n",
                sizeof(int));
        printf("3: sizeof array of int=%d\n",
```

```
                sizeof(b));
        printf("4: sizeof float=%d\n",
                sizeof(float));
        printf("5: sizeof double=%d\n",
                sizeof(double));
    }
```

C>EXP4_7

1: sizeof int variable=2

2: sizeof int=2

3: sizeof array of int=200

4: sizeof float=4

5: sizeof double=8

(2)条件运算符　C 语言特有,也是 C 语言中唯一的三项运算符,功能相当于一段程序,用法如下:

表达式 1? 表达式 2:表达式 3

其含义是:

若表达式 1 为真,则整个表达式的解就是表达式 2 的解,否则,为表达式 3 的解 。

〔练习 4.8〕

C>TYPE EXP4_8.C

```
main()
    {  int a=5,b=3;
       printf("(a>b? a:b=%d\n",
              (a>b)? a:b)
    }
```

C)EXP4_8

(a>b)? a:b=5

(3)逗号运算符　该运算符用在括号中有多个表达式的场合。其功能是:从左到右求解各个表达式,而整个表达式的值为最后所求解的表达式的值。例如:下面逗号表达式的解为 x 的值等于 4。

x=3,x=4;

逗号运算符有效地用于 while 语句和 for 语句中。

〔练习 4.9〕

C>type exp4_9.c

```
main()
    {  int x,y;
       printf(" %d\n",(x=3,y=4));
    }
```

C)EXP4_9

4

4　指针的概念及其运算符

(1)指针的概念　指针也是一种变量,是用来指定另一个变量(允许是结构变量,另一个指针变量)所在场所的变量,其内容为所指定变量的地址,指针变量的定义格式如下所示:

数据类型　*指针变量名;

例如:

char *p; p 为指向字符型数据的指针。

int *var var 为指向整型数据的指针。指针变量 p 和 var 的内容及其功能如图 4.1 所示。

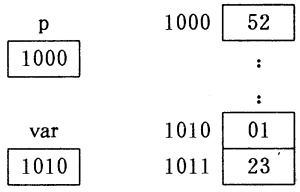


图 4.1　指针的概念

(2)指针有关的运算符　与指针有关的运算符有 & 与 * 两种。& 用来求变量的地址,叫地址运算符。* 用来访问指针所指变量的内容,例如:

int i=10; 定义变量 i 且赋初值 10

int *p; 定义 p 为指针变量

p=&i; 给 p 赋上变量 i 的地址,这时,就可以说指针 p 指向了变量 i。

*p=21; 给 p 所指的变量内赋上数据 21。

说明上述分析正确性的程序,见练习 4.10

〔练习 4.10〕

c>TYPE EXP4_10.C

```
main()
    {  int i=10,j;
       int *p;
       p=&i;j=*p;*p=21;
       printf("address of p j i *&i \n");
       printf("%p       %d %d %d \n",
              p,j,i,*&i);
    }
```

C)EXP4_10

```
address of p   j     i     *&i
FFDA          10    21    21
```

5　类型显示转换运算符

C 语言中的数据类型转换分为隐式(自动进行的)和显式(强制性的)两种。

(1)隐式转换　C 编译程序对表达式中出现的不同数据类型,按如下规则自动进行类型转换。

①表达式中的所有 float 类型数据,在任何操作之前均被转换为 double 类型。只有在数组和结构中 float 类型才真正按 float 型处理

②在算术表达式里,所有 short 和 char 型数据都转换为 int 型。

③参加运算的两个操作数类型不同时,要按图 4.2 所示把优先级低的类型转换为优先级高的类型。

double 型

long 型　　↑

unsigned 型　　优先级高

int　型

图 4.2　数据类型的优先级别

④在赋值表达式中,要把右边表达式的类型转换成左边表达式的类型,如练习 4.11 所示。

〔练习 4.11〕

C)TYPE EXP4_11.C

```
main()
    {  int i ,j;
       char c;
       float f;
       c=50;
       printf("c=%c\t c=%d\n",c,c);
       i=2.6;f=2;
       printf("i=%d\t f=%f\n",i,f);
       i='a';j='a'+1;
       printf("i=%d\t j=%d\n",i,j);
       printf("c=%d\t c=%c\n",c-'0'c);
    }
C>EXP4_11
 c=2      c=50
 i=2      f=2.000000
 i=97     j=98
 c=2      c=2
```

(2)显式转换　一个表达式 exp 可用显式转换表达式

```
(type) exp
```

把其值转换为园括号中 type 所指定的数据类型如：

```
int i=10;
double f;
f=sqrt(double)i);
```

其中园括号及其内的类型名即类型显式转换运算符。

读进 double 型数据，进而显示其值及截断后的值的程序，如练习 4.12 所示。

〔练习 4.12〕

```
C>TYPE EXP4_12C
main ()
    {  double f;
       scanf("%lf",&f);
       printf("the float value %10.2f\n",f);
       printf("truncated is %d\n",(int)f);
    }
C>exp4_12
12.2345678
the float value        12.23
truncated is 12
```

(3)溢出问题　在进行运算时，时常会遇到结果产生溢出现象，如练习 4.13 所示。

〔练习 4.13〕

```
C>TYPE EXP4_13.C
main()
    {  int x,y,z;
       x=400;y=200;
       z=x*y;
       printf("z=%d\n",z);
    }
C>EXP4_13
z=14464
```

在这个练习里，x、y、z 被定义为整型变量，x * y 的结果明显超出 int 型的数据范围。即使把变量 z 定义为 long 型，终因 x 和 y 的相乘是在整型范围内进行的，是不会有正确结果的。解决办法只有用类型显式转换，把 x、y 中的一个数转换为 long 型，这时，根据隐式转换规则，则另一个数也就自动转换为 long 型的了；然后再把 long 型的结果赋给事先已定义为 long 型的变量 z，如练习 4.14 所示。

```
C>TYPE EXP4_14.C
main ()
    {  int x,y;
       long z;
       x=400;y=200;
       z=(long)x*y;
       printf("%ld",z);
    }
C>EXP4_14
  80000
```

6 运算符优先级

在 C 语言中，各运算符是有其运算的优先级别的。运算符的优先级决定了表达式的求解顺序。C 语言运算符的优先级如表 4.5 所示。

表 4.5　C 语言运算符优先级

运　算　符	分　类	结合性
()　[]　→　.		从左至右
!　~　++　--　-(type)　*　&sizeof	单项运算符	从左至右
*　/　%		
+　-		
≪　≫		
〈　〈=　〉　〉=		
==　!=	双项运算符	从左至右
&		
^		
|		
&&		
||		
? :	条件运算符	从右至左
=　+=　-=　*=　/=　%=　≫=　≪=　&=　^=　|=	赋值运算符	从右至左
,	逗号运算符	从左至右

从该表我们可以看出：

(1)C 语言有 44 个运算符。

(2)这些运算符分 15 个优先级别；优先级自上向下变低；其中()运算符优先级最高；逗号运算符优先级最低；赋值运算符的优先级也比较低。　(转 22 页下)

学用单片机

智力竞赛抢答装置

宋济华

单片微机是近几年来在国内计算机行业崛起的一颗新星。由于它具有体积小、价格低、编程效率高、控制能力强等特点，被广泛地应用在工业自动控制和仪器仪表智能化领域，是微机应用产品化的理想机型。本人利用 MCS－48 系列单片微机 8035 制作了一台智力竞赛抢答装置。希望能给初学单片机的朋友以抛砖引玉的作用。

市售的智力竞赛抢答器大都是由分离式电子原件组装的。由于设计和布线困难，机器本身一般只具有抢答单一功能，举行智力竞赛还必须同时配备专门的计时器和计分器，机器的功能和可靠性都受到了极大的限制。本文介绍的智力竞赛抢答装置由于采用了先进的微电脑技术，使机器的性能更趋于完善。它集抢答器、计时器、计分器、音响提醒和数字显示于一体，具有较高的性能价格比，是举行智力竞赛的理想工具。

一、功能说明

本智力竞赛抢答装置（下文简称“装置”由控制台、显示台和抢答器三部分组成。控制台设抢答、回答、复位、加分和减分五个按键式开关以及显示计时、抢答号和得分的六个数码显示器，是本装置的核心控制单元，显示台为八组共二十四个大型数码显示器，显示八个抢答组的当前得分。抢答台由八个按键式开关组成，参赛组各执一个。

装置上电后，初始化数据项并扫描控制台。当主持人出完题启动抢答键后，20 秒定时器开始计时并开放抢答台，计时显示器从 20 开始以秒为单位递减显示抢答剩余时间。为了暄染竞赛的紧张气氛，扬声器每秒都发出音长为 1/4 秒的短笛声。如果在 20 秒内无抢答，抢答号和计分显示器始终显示零，20 秒到后，扬声器发出三次“嘟嘟”声，以示抢答时间到，计时显示器回零且立即关闭抢答台，回到初始态。假如在 20 秒内有一组先抢答，扬声器中马上发出“叮咚”声，计时显示器跳回零，抢答号和计分显示器立即显示抢答组的组号和当前成绩。为了不致于其他组再抢答，有一组抢答后，装置立即关闭抢答台。此时，主持人可启动回答键，给回答者计时，计时显示器也从 20 开始以秒为递减单位显示回答剩余时间，不过此期间扬声器并不发出音响，目的是不致于影响观众听回答者的答题。20 秒后，扬声器发出三次“嘟嘟”报警声，以示回答时间到，计时显示器回零，主持人可根据回答的正误情况按动加分、减分键，并将此成绩计入到该组的总得分中，从显示器上反映出来。一旦按动了复位键，装置便回到初始态。

在智力竞赛中，参赛各组的基分数一般为 100 分，以 10 为递增减单位。但在本装置中，为了简化电路和程序起见，以 10 为基分数，以 1 为递增减单位，而在显示器上反映出来的还是以 100 为基分数，10 为增减单位，这是因为在各组计分显示器的尾部多添了一个“0”。因此从效果上与前者是一致的。

二、硬件简介

本装置主要由单片微机 8035、程序存储器 E-PROM（2716）、地址锁存器 8212、数据输出锁存器（74LS273）、输入缓冲电路（74LS244）、显示驱动器、电源和外围器件组成。8035 内部集成了中央处理器（CPU）、数据存储器（RAM）、地址和数据总线（BUS）、定时器/计数器（CTC）以及两个并行的输入输出接口电路（P1 和 P2）是本装置的核心部件。2716 是本装置的 2K×8Bit 程序存储器；8121 是地址锁存器，由于 CPU 分时传输地址和数据信息，当 CPU 执行指令期间用它锁存地址的低八位信息；在显示电路中用的四片八 D 锁存器 74LS273，通过 8035P1 口提供的选片信号为后继的显示驱动器锁存 BUS 送来的段和位信息，两片 74LS244 是控制台和抢答台的输入缓冲电路接口；本装置中的六个显示驱动器为大型数码管显示器提供大功率的电流驱动。电源电路在本刊九〇年第 3 期《MCS48 交通信号自动控制》中已详细说明，不再赘述。

三、工作原理

图 1 是本装置的电原理图，图 2 是监控程序结构框图。当机器上电自举或复位后，8035 执行初始化，程序，P1 口各位依次变为低电平，为显示器数据锁存器提供选片信息，分时将各位数码显示器点亮并循环扫描控制键盘。当控制台有键闭合时，CPU 执行查询程序，译出键码，确定键意。若为抢答，监控启动 20 秒定时程序，并且每秒都在 P25 上输出一个高电平脉冲，触发短笛音电路。同时使 P17 变为低电平，选中抢答台键盘输入缓冲电路并循环扫描之。若在 20 秒定时期间，抢答台上有键闭合，监控立即使 P17 变为高电平，关闭抢答台，从而使以后的抢答失效，并在 P26 输出一个高电平脉冲，触发“叮咚”音响电路发声，尔后查询组号，并将抢答号和该组的成绩送控制台上的数码显示器且使计时器回零。定时到且无抢答，则 CPU 使 P27 输出高电平脉冲，触发“嘟嘟”电路发声三次，使 P17 为高电平关闭抢答台，计时器回零，操作返回初始态。

抢答台有抢答时，系统扫描控制台等待键盘输入，若回答键闭合，则监控重新启动 20 秒定时程序且等待控制台加分、减分或复位键闭合，以应答回答的正误，

并通过程序将加减的分数计入总成绩中显示出来，启动复位键，监控将操作返回初始态。

本装置的显示驱动器原理图见图 3，它是用晶体管和阻容元件放大数据锁存器后拖的 TTL 信号为大型数码管显示器的段信号和位信号提供足够功率的电流驱动。

本装置的监控程序的设计采用了“模块化”结构，在执行时采用子程序调用的方法来实现指定模块的功能。限于篇幅，监控程序表从略下面将几个主要模块的功能说明如下：

1. INT $：中断处理模块，它的作用是为装置提供精确的 20 秒定时。

2. DIGS：显示模块，指令序列将 CPU 提供的显示器段和位信息送显示器显示出来。

3. CZ：拆字模块，是将计数值、抢答组号以及成绩值八位二进制数变成两位十进制数送显示模块(DIGS)。

4. ZMB：字模表模块，存储在 300～309 单元中的数字 0～9 的显示字模表。监控程序根据累加器 A 的值利用 MOVP3 A，@A 指令在程序存储器的第三页(300～3FF)单元内的 300～309 的字模信息。

5. DELAY：延时 20ms 模块，配合键盘输入缓冲电路，该模块提供 20ms 延时，消除按键的抖动。

以上这些模块构成监控程序的基础，各种不同的控制功能都由这些模块堆积而成。

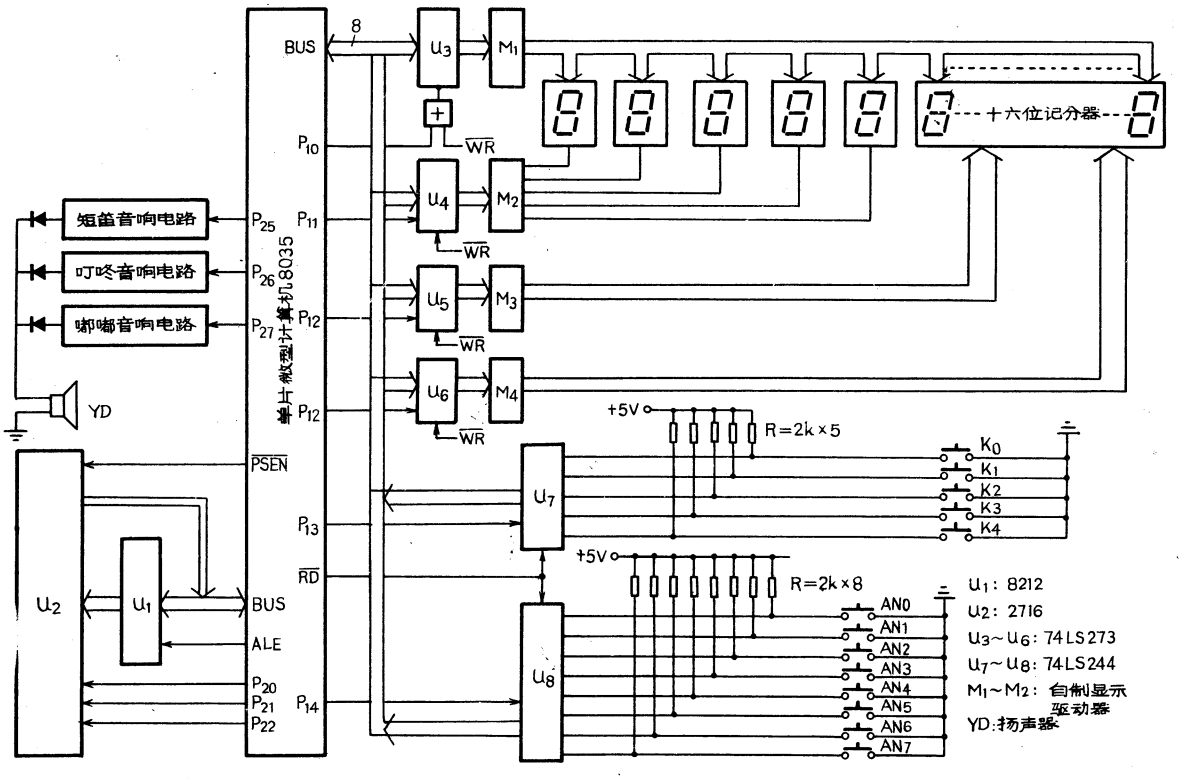


图 1

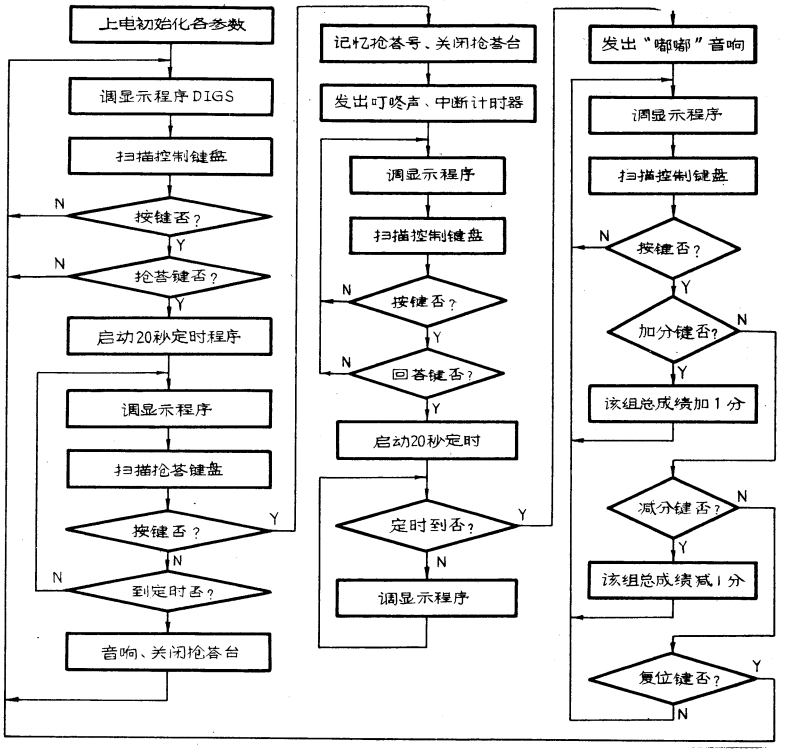

上电初始化各参数
调显示程序 DIGS
扫描控制键盘
按键否？
抢答键否？
启动20秒定时程序
调显示程序
扫描抢答键盘
按键否？
到定时否？
音响、关闭抢答台
记忆抢答号、关闭抢答台
发出叮咚声、中断计时器
调显示程序
扫描控制键盘
按键否？
回答键否？
启动20秒定时
定时到否？
调显示程序
发出“嘟嘟”音响
调显示程序
扫描控制键盘
按键否？
加分键否？
该组总成绩加1分
减分键否？
该组总成绩减1分
复位键否？
N
Y

图 2

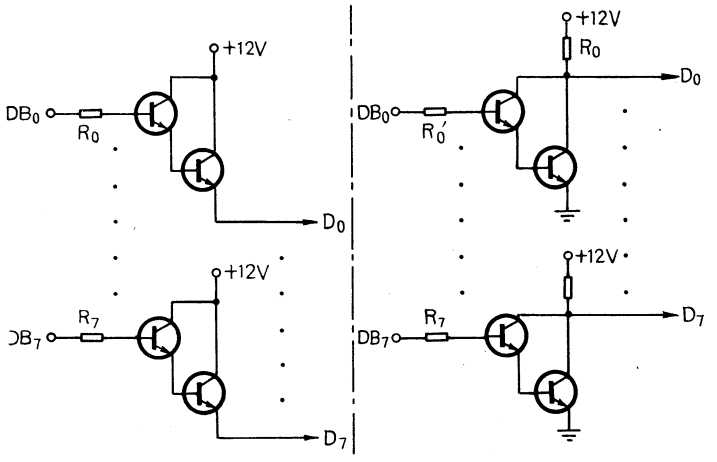

+12V
DB0
R0
D0
DB7
R7
D7
R0′

图 3

单片机电子音乐编程实践

张培仁　张新宇

音乐由一系列特定频率的声音组成。单片机能输出不同频率的方波信号形成音频，通过延时形成音长（节拍），这样，单片机就实现音乐演奏功能。

一、原理

通常采用十二平均律，将一个八度音程按十二半音频率按等比数列排列（见表一）。

对应着音名，可以很方便地写出各个音调音频的的简谱符号。规则是：不论低中高音，1～2（1 跟 2），2～3，4～5，5～6，6～7，各相差一个音程，3～4，7～i 各相差半个音程。例如 G 调的 ! 的音频为 196HZ，这一规律将用到程序设计之中。

为了用 8051 实现这些音频，我们用了定时器 TO。定时器溢出后产生中断，将连接喇叭的输出口（P1.7）取反，即可产生方波形成音频。

置 TO 为工作方式 1，即 THO，TLO 构成 16 位定时器，假定机器频率为 0.5MHz，则对应于每个音频的时间常数见表一。

表一. 时间常数（机器频率为 0.5MHz

NO. T	I	音名	音频 (HZ)	时间常数	简谱符号 C	D	E	F	G	A	B
1	22	C	131	F88D	1						
2	23	C#	139	F8F8							
3	24	D	147	F95D	2	1					
4	25	D#	156	F9BC							
5	26	E	165	FA16	3	2	1				
6	27	F	175	FA6B	4			1			
7	28	F#	185	FABC		3	2				
8	29	G	196	FBO7	5	4		2	1		
9	30	G#	208	FB4F			3				
10	31	A	221	FB92	6	5	4	3	2	1	
11	32	A#	234	FBD2				4			
12	33	B	248	FC0E	7	6	5		3	2	1
13	34	C	262	FC46	1			5	4		
14	35	C#	278	FC7C		7	6			3	2
15	36	D	294	FCAE	2	1		6	5	4	
16	37	D#	312	FCDE			7				3
17	38	E	330	FD0B	3	2	1	7	6	5	4
18	39	F	350	FD35	4			1			
19	40	F#	371	FD5E		3	2		7	6	5
20	41	G	393	FD83	5	4		2	1		
21	42	G#	416	FDA7		3			7	6	
22	43	A	441	FDC9	6	5	4	3	2	1	
23	44	A#	467	FDE9				4			7
24	45	B	495	FE07	7	6	5		3	2	1
25	46	C	525	FE23	1			5	4		
26	47	C#	556	FE3E		7	6			3	2
27	48	D	589	FE57	2	1		6	5	4	
28	49	D#	624	FE6F			7				3
29	50	E	661	FE85	3	2	1	7	6	5	4
30	51	F	700	FE9A	4			1			
31	52	F#	742	FEAF		3	2		7	6	5
32	53	G	786	FEC1	5	4		2	1		
33	54	G#	833	EED3			3			7	6
34	55	A	882	FEE4	6	5	4	3	2	1	
35	56	A#	935	FEF4				4			7
36	57	B	990	FF03	7	6	5		3	2	1
37	58	C	1049	FF11				5	4		
38	59	C#	1112	FF1F		7	6			3	2
39	60	D	1178	FF2B				6	5	4	
40	61	D#	1248	FF37			7				3
41	62	E	1322	FF42				7	6	5	4
42	63	F	1400	FF4D							
43	64	F#	1484	FF57					7	6	5
44	65	G	1572	FF60							
45	66	A	1665	FF69						7	6
46	67	A#	1764	FF72							
47	68	B	1869	FF7A							7

硬件配置见图 1

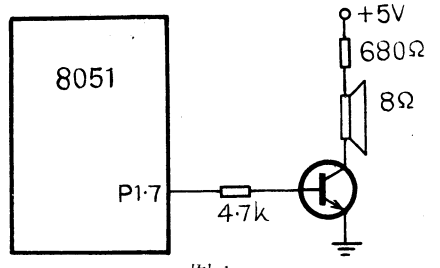
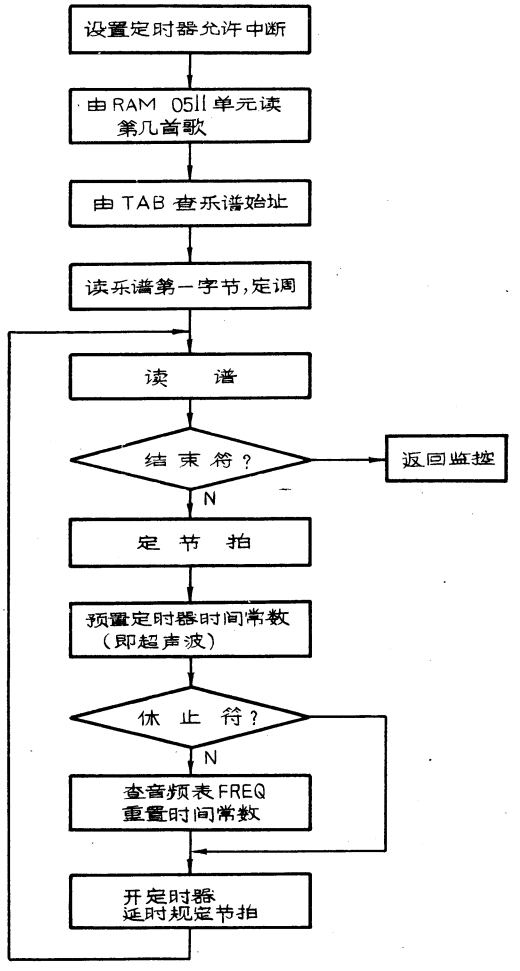


图 1

二、自动奏乐程序

将编好的若干个乐曲程序，存放在 EPROM 中，即通过查表音符数据输出一段音乐。从乐曲程序中可以看出，此程序能表现 C、D、E、F、G、A、B 中的任意曲调，能表现从 1 到 7 的二十一个音频、休止符和十种节拍。最关键是的，采用压缩数据，一个字节即包含音频和音长两个信息，节省很大内存。

程序结构框图如图二。



图二

自动奏乐程序清单

```
1   ORG 2000H
2   SPK EQU P1.7
3   MOV IE,#82H          ;允许中断
4   ANL 89H,#0F0H        ;置定时器为工作方式1
5   ORL 89H,#01H
6   AJMP START
7   ORG 200BH            ;中断服务程序
8   CPL SPK              ;取反,产生方波
9   MOV TH0,R7
10  MOV TL0,6R
11  STEB TR0
12  RETI
13  START:MOV R5,05H     ;选歌
14  star:MOV DPTR;#TAB   查乐谱首址
15  MOV A,R5
16  DEC A
17  RL A
18  MOVC A,@A+DPTR
19  PUSH A
20  MOV A,R5
21  DEC A
22  RL A
23  INC A
24  MOVC A,@A+DPTR
25  PUSH A
26  POP DPL
27  POP DPH
28  MOV A,#00H           ;定调
29  MOVC A,@A+DPTR
30  MOV R3,A
31  INC DPTR
32  SONG:CLR SPK
33  MOV A,#00H           ;读乐谱
34  MOVC A,@A+DPTR
35  MOV R2,A
36  SWAP A
37  ANL A,#0FH
38  JZ EXIT              ;结束符,转EXIT
39  MOV B,#05H
40  DIV AB
41  XCH A,B
42  JNZ MOM
43  MOV A,#05H
44  DEC B
45  MOM:MOV R0,A
46  MOV R1,B
47  MOV A,R2
48  ANL A,#0FH
49  MOV B,#08H
50  DIV AB
51  MOV R2,B
52  MOV R4,A
53  MOV R7#0FFH          ;预置为休止符
54  MOV R6,#0F9H
55  MOV A,R2
56  JZ YANS              ;休止符则转
57  CLR C                ;转化为余数处理数
58  RLC A                ;余数:1234567
59  DEC A                ;余数处理数:13568AC
60  MOV R2,A
61  CLR C
62  SUBB A,#06H
63  JC AN
64  DEC R2
65  AN:MOV A,#0CH
66  MOV B,R1
67  MUL AB
```

```
68  ADD A,R2
69  YANS:PUSH DPH
70  PUSH DPL
71  JZ TM                    ;休止则转 TM
72  DEC A                    ;否则,查音频表 FREQ
73  ADD A,R3                 ;重置时间常数
74  RL A
75  MOV DPTR,#FREQ
76  MOV R1,A
77  MOVC A,@A+DPTR
78  MOV R7,A
79  MOV A,R1
80  INC A
81  MOVC A,@A+DPTR
82  MOV R6,A
83  TM:MOV TH0,R7
84  MOV TL0,R6
85  SETB TR0                 ;开定时器
86  MOV DPTR,#LONG           ;查延时常数
87  MOV A,R0
88  DEC A
89  MOVC A,@A+DPTR
90  MOV R0,A
91  MOV A,R4
92  JZ TTT                   ;使用节拍 1
93  MOV A,R0                 ;使用节拍 2
94  CLR C
95  RRC A
96  MOV R0,A
97  TTT:ACALL DELAY          ;延时指定节拍
98  DJNZ R0,TTT
99  POP DPL
100 POP DPH
101 INC DPTR
102 AJMP SONG                ;继续演奏
103 EXIT:CLR SPK             ;退出
104 CLR TR0
105 LJMP 0000H
106 DELAY:MOV R4,#40H        ;延时子程序
107 LOOP1:MOV R1,#35H
108 LOOP2:NOP
109 DJNZ R1,LOOP2
110 DJNZ R4,LOOP1
111 RET
112 FREQ:DB,0F8H,8DH,0F8H,0F8H,0F9H,5DH,
    0F9H,0BCH,0FAH,16H,0FAH,6BH
113 DB,0FAH,0BCH,0FBH,07H,0FBH,07H,0FBH,
    4FH,0FBH,92H,0FBH,0D2H,0FCH,0EH
114 DB 0FCH,46H,0FCH,7CH,0FCH,0AEH,0FCH,
    0DEH,0FDH,0BH,0FDH,35H
115 DB 0FDH,5EH,0FDH,83H,0FDH,0A7H,0FDH,
    0C9H,0FDH,0E9H,0FEH,07H
116 DB 0FEH,23H,0FEH,3EH,0FEH,57H,0FEH,6FH,
    0FEH,85H,0FEH,9AH
117 DB 0FEH,0AFH,0FEH,0C1H,0FEH,0D3H,0FEH,
    0E4H,0FEH,0F4H,0FFH,03H
118 DB 0FFH,11H,0FFH,1FH,0FFH,2BH,0FFH,37H,
    0FFH,42H,0FFH,4DH
119 DB 0FFH,57H,0FFH,60H,0FFH,69H,0FFH,72H,
    0FFH,74H,0FFH,81H
120 LONG:DB 08H,20H,30H,80H,0C0H ;延时常数,
    代表 1/4/,1,3/2,4,6 拍
121 TAB:DB 21H,22H,21H,47H,21H,81H,21H,0B7H,
    21H,0DFH,22H,46H,22H,72H
122 ORG  2122H
123 T1:DB 09H ;A 小调 4/4 日本歌曲
124 DB 73H,7BH,7AH,71H,79H,2FH,36H,26H,4EH;
    四季歌
125 DB 74H,7CH,7BH,7AH,79H,7AH,7CH,93H
126 DB 74H,7CH,7BH,72H,7AH,7CH,73H,7BH,79H,
    26H,26H
127 DB 27H,73H,7AH,79H,2FH,79H,46H,00H
128 ORG 2147H
129 T2:DB 05H, ;F 调 2/4 苏格兰民歌
130 DB 2DH,89H,69H,79H,7BH,7AH,69H,72H,73H,;
    友谊地久天长
131 DB 89H,69H,7BH,7DH
132 DB 76H,7EH,8BH,6BH,7BH,79H
133 DB 8AH,69H,7AH,7BH,89H,6EH,2EH,2DH
134 DB 8EH,7EH,8DH,6BH,7BH,79H
135 DB 8AH,69H,7AH,7EH,8DH,6BH,7BH,7DH
136 DB 86H,0C9H,8DH,6BH,7BH,79H
137 DB 8AH,69H,7AH,7BH,89H,1EH,2EH2DH,81H,
    00H
138 ORG 218H
139 T3:DB 09H ;A 调 4/4  日本民歌
140 DB 73H,73H,76H,77H,0C1AH,77H,9EH, ;
    荒城之月
141 DB 74H,74H,73H,72H,0ABH,70H
142 DB 73H,73H,76H,77H,0C1H,77H,9EH
143 DB 74H,74H,73H,73H,5EH,70H
144 DB 81H,69H,27H,26H,74H,74H9BH
145 DB 72H,73H,74H,74H,0ABH,70H
146 DB 73H,73H,76H,77H,0CH1,77H,9EH
147 DB 74H,74H,73H,73H,5EH,70H,00H
148 ORG 21B7H
149 T4:DB07H ;G 调 4/4 日本歌曲
150 DB 81H,7BH,7AH,79H,2FH,79H,72H,72H,9AH ;
    你叫什么名字?
151 DB 82H,7CH,7BH,7AH,79H,7AH,73H,9BH
```

```
152 DB 85H,7EH,7DH,7BH,7CH,7DH,76H,76H,9EH
153 DB 85H,7CH,7BH,7BH,7AH,7AH,79H,79H,79H,
    79H,99H,00H
154 ORG 21DFH
155 T5:DB 07H ;G 调 4/4 美国歌曲
156 DB 7DH,0CBH,0CAH,0C9H,85H,7DH,7DH
    铃儿响叮当
157 DB 7DH,0CBH,0CAH,0C9H,86H,7EH
158 DB 7EH,0CCH,0CBH,0CAH,87H,7FH
159 DB 0CDH,0CDH,0CCH,0CAH,0D3H,0C9H
160 DB 7DH,0CBH,0CAH,0C9H,85H,7DH
161 DB 7DH,0CBH,0CAH,0C9H,86H,7EH
162 DB 7EH,0CCH,0CBH,0CAH,0CDH,0CDH,0CDH,
    0CDH
163 DB 0CEH,0CDH,0CCH,0CAH,0D1H,28H
164 DB 0CBH,0CBH,0C3H,0CBH,0CBH,0C3H
165 DB 0CBH,0CDH,0D9H,0BAH,0EBH
166 DB 0CCH,0CCH,0DCH,0BCH,0CCH,0CBH,0CBH,
    0CBH
167 DB 0CBH,0CAH,0CAH,0C9H,0C2H,0C5H
168 DB 0CBH,0CBH,0C3H,0CBH,0CBH,0C3H
169 DB 0CBH,0CDH,0D9H,0BAH,0EBH
170 DB 0CCH,0CCH,0DCH,0BCH,0CCH,0CBH,,0CBH,
    0CBH
171 DB 0CDH,0CDH,0CCH,0CAH,0D1H,28H,00H
172 ORG 2246H
173 T6:DB 09H ;E 调 2/4《桥》插曲
174 DB 2BH,2EH,2FH,79H,4EH ;啊,朋友再见
175 DB 2BH,2EH,2FH,79H,4EH
176 DB 2BH,2EH,2FH,71H,2FH,2EH
177 DB 71H,2FH,2EH,73H,73H,73H,7AH,7BH
178 DB 7CH,9CH,7EH,7DH,7CH,7EH,9BH
179 DB 7BH,7AH,79H,27H,73H,71H,27H
180 DB 3EH,30H,3EH,28H,00H
181 ORG 2272H
182 T7:DB 04H ;E 调 3/4 爱的罗曼史
183 DB 0C3H,0C3H,0C3H,0C3H,0C2H,0C1H
184 DB 0C1H,77H,76H,76H,0C1H,0C3H
185 DB 0C6H,0C6H,0C6H,0C6H,0C5H,0C4H
186 DB 0C4H,0C3H,0C2H,0C2H,0C3H,0C4H
187 DB 0C3H,0C4H,0C3H,0C5H,0C4H,0C3H
188 DB 0C3H,0C2H,0C1H,0C1H,77H,76H
189 DB 77H,77H,77H,77H,0C1H,77H
190 DB 76H,76H,76H,00H
```

1. 选歌

在通用程序中可以输入多首乐曲(以 0×H 结尾),而将每首乐曲乐谱首地址按顺序放入表 TAB 中(见程序清单)。我们从 RAM 某个单元,此为 05H 单元读出第几首歌,查表 TAB 即可得到乐曲乐谱首址。演奏乐曲直到结束字 0×H。本程序写有七首歌。

2. 选调

乐曲乐谱表第一字节是用来定调的(存入 R_3)。也就是在查音频表 FREQ 时须加上的偏移量。基准是 C 调,即偏移量为 0(见表二)。其它音调各简谱符号相对于 C 调都有一个偏移量,D 调偏移量是 2,故用 02H 来定 D 调。

表二:

音调	C	D	E	F	G	A	B
偏移量	0	2	4	5	7	9	B

3. 音频和音长(节拍)

我们设计了一种算法,使压缩数据能表现低中高音,不同的音长、休止符和结束符。

乐曲中一个字节为一个单位,即可表现音频和音长。字节高四位取自表三,低四位取自表四。高四位数包含音长和音高(即是低音,中音还是高音)两个信息。音长在高四位数据中尚不明确,使用拍节 1 还是拍节 2 还需由低四位数肯定。高四位为 0 表示结束符。低四位指出使用的节拍和音符。不论是 1、1 或 1,低四位都应是 1 或 9。低四位为 0 或 8 表示休止符。具体是:高四位数除以 5 的余数指出拍节 1 我们的算法假使 5 除以 5 得 0 余 5,10 除以 5 得 1 余 5……),查表 LONG(见程序清单)得延时常数。然后低四位数除以 8,如商为 0 表示延时常数不变,即拍节 1。否则,延时常数除以 2,.即拍节 2。拍节 2 是拍节 1 的一半。

表三. 高 4 位值

拍节 1	1/4	1	3/2	4	6
拍节 2	1/8	1/2	3/4	2	3
低　音	1	2	3	4	5
中　音	6	7	8	9	A
高　音	B	C	D	E	F

表四. 低 4 位值

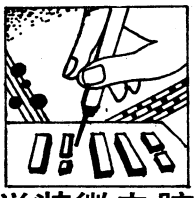
音符	1	2	3	4	5	6	7	0
拍节 1	1	2	3	4	5	6	7	0
拍节 2	9	A	B	C	D	E	F	8

LONG 中的延时常数分别代表 1/4、1、3/2、6 拍,延时时间可由程序延时部分(标号 TTT)算出。实际上还应加上数据处理过程的时间,所以实际延时比表五中的长。

表五. 延时时间

拍节	1/8	1/4	1/2	3/4	1	3/2	2	3	4	6
延时(s)	0.08	0.16	0.33	0.49	0.66	0.98	1.31	1.97	2.62	3.93

(转至 39 页下)



学装微电脑

数字游标长尺的制作

易齐干

在微电脑控制技术中，往往要测量物体的长度，这时，必须把测得的尺寸变换为微电脑能理解的数字量。

本文以数字游标长尺为例，介绍微电脑测长度的原理，它是利用滑动电位器，将长度转换为电压，再用模拟量/数字量(A/D)变换电路将电压变换为 8 位数据，通过 UP－80 显示其测量值，亦用打印机打印出结果。

1. 数字游标卡尺的结构

数字游标卡尺如图 1 所示。由两部份组成：一是滑动电位器为主体的机械部份，它把长度尺寸转换为电压。另一部份是接口，它把电压变换为 8 位数字量。

该数字游标卡尺的滑动式电位器可变范围最大为 40mm，所以改变定程档块的位置，能测量的范围分别为：0～40mm；20～60mm；40～80mm；60～100mm；80～120。

A/D 变换电路 TC5090P 可将滑动式电位器测得的电压变换为 8 位数字量。它是将 0～5V 模拟电压变换为 00H～FFH 的 8 位数字量的 IC。分辨率为 256(电压为 5/256V 分辨率)，即，0.5mm～52AH 对应；0.0mm～528H 对应；400mm－SC8H 对应。这样，若在 00H～FFH 范围内选择较为合适。不按上述原则也可以。

尺寸测量值、8 位数据与电压的关系如表 1 所示。

表 1　测量值和 8 位数据与电压的关系

测量值(mm)	0.0	0.5	1.0	……	39.0	39.5	40.0
8 位数据(16 进制)	28H	2AH	2CH	……	C4H	C6H	C8H
电压(V)	0.78	0.82	0.86	……	3.83	3.87	3.91

注 1)测量值为 0.0mm 时电压为(16×2＋8)×5/256 V＝0.78V

注 2)每 0.5mm 的电压变化为 5/256×2＝0.04V

表 2　STC，RSEL L，H 信号工作关系

工作 \ 端	STC	RSEL
A/D 转换开始	H	H
要求高位数据	L	H
要求低位数据	L	L

例如，滑动式电位器得到的电压为 3.87V，TC5090P 输出数据 C6H，将它输入给微电脑，则数字显示为 39.5mm。

2. A/B 转换电路(图 2)

有各种各样的方法可把模拟量转换为数字量。本文使用东芝制造的 TC5090P 据有 A/D 转换功能。来自微电脑的低电平(0V)或高电平(5V)分别出现在 STC 端或 RSEL 端，或开始 A/D 转换，或将转换后的 8 位数据分为高 4 位、低 4 位，由 D_4～D_1 端输出。

STC、RSEL 高电平、低电平与工作关系如表 2 所示。EOC 端为低电平时，该 IC 正在进行 A/D 转换；EOC 为高电平时，表示转换结束。

图 2 所示的电路中，有两片型号为 74LS157，其作用是将较多的数据输入给具有较少输入端的微电脑。

这两片 IC 各具备四位的输入端口 A、B，以 SEL 端为低电平、高电平决定/IC 的输出。SEL 的高电平、低电平与信息的流动如表 3 所示。

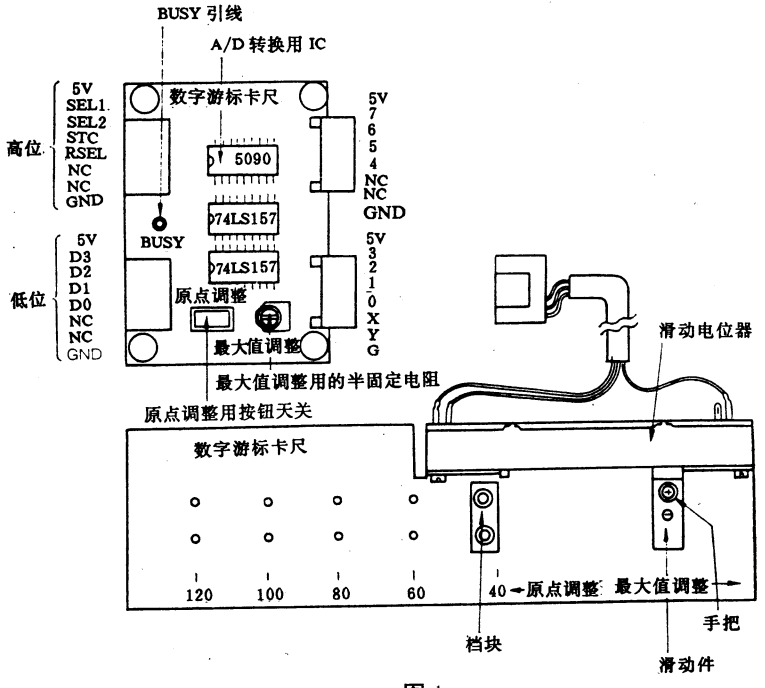
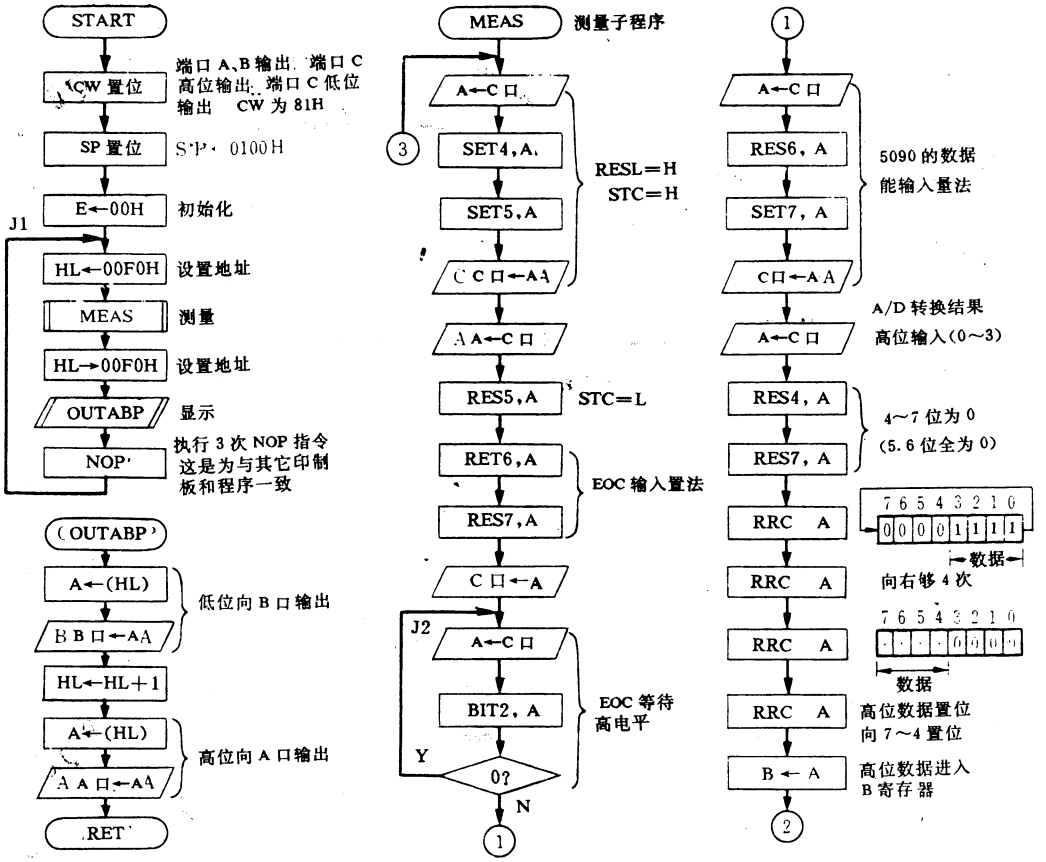


图 1

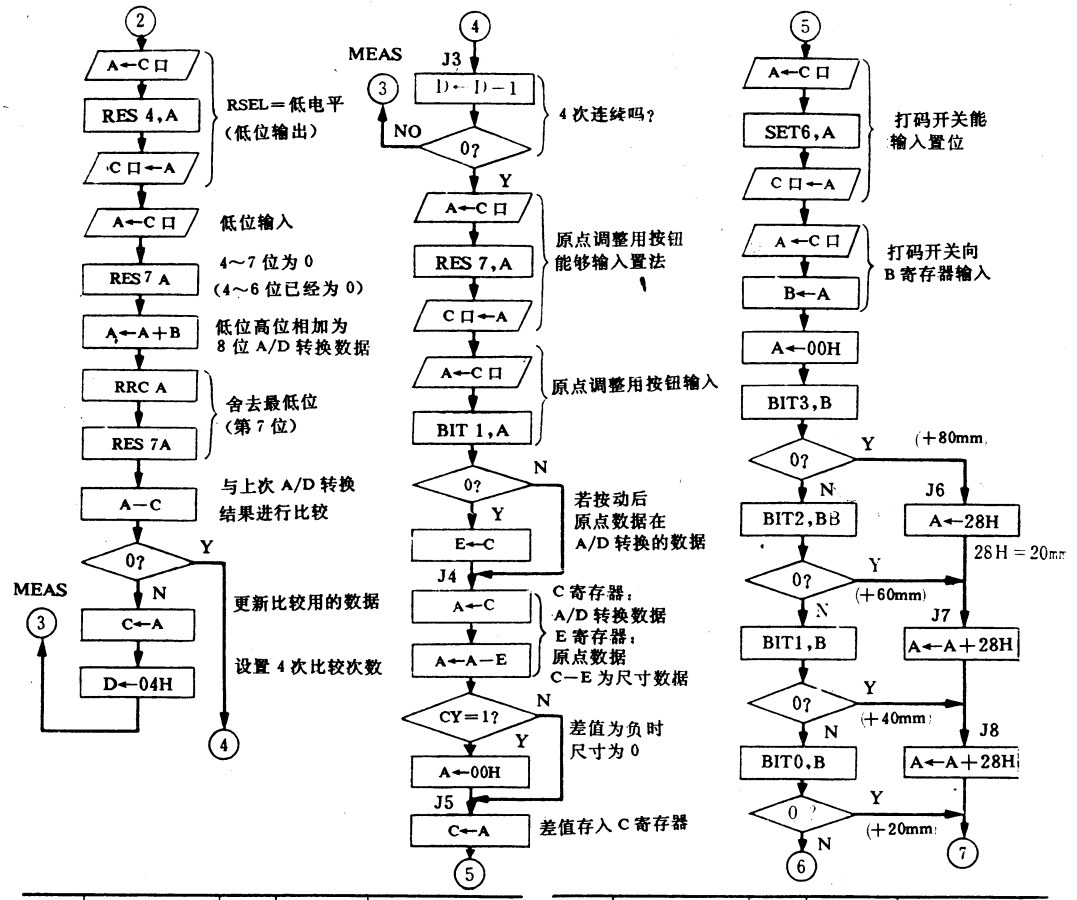
表 4.(1)



标号	助记符	地址	机器语	注释
START	LD A,81H	0000	3E 81	输入、输出置位
	OUT (03H),A	0002	D3 03	
	LD SP,0100H	0004	31 00 01	SP←0100H
	LD E,00H	0007	1E 00	初始化
J1	LD HL,00F0H	0009	21 F0 00	设置地址
	CALL MEAS	000C	CD 1B 00	测量子程序
	LD HL,00F0H	000F	21 F0 00	设置地址
	CALL OUTABP	0012	CD E6 00	显示
	NOP	0015	00	
	NOP	0016	00	什么也不做
	NOP	0017	00	
	JP J1	0018	C3 09 00	
MEAS	IN A,(02H)	001B	DB 02	
	SET 4,A	001D	CB E7	RESL=H
	SET 5,A	001F	CB EF	STC=H
	OUT (02H),A	0021	D3 02	
	IN A,(02H)	0023	DB 02	
	RES 5,A	0025	CB AF	STC=L
	RES 6,A	0027	CB B7	EOC 输入上置位
	RES 7,A	0029	CB BF	
	OUT(02H),A	002B	D3 02	

标号	助记符	地址	机器语	注释
J2	IN A,(02H)	002D	DB 02	EOC=高电平
	BIT 2,A	002F	CB 57	
	JP Z,J2	0031	CA 2D 00	
	IN A,(02H)	0034	DB 02	5090 的数据纯输入置位
	RES 6,A	0036	CB B7	
	SET 7,A	0038	CB FF	
	OUT(02H),A	003A	D3 02	
	IN A,(02H)	003C	DB 02	高位输入
	RES 4,A	003E	CB A7	4～7 位为 0
	RES 7,A	0040	CB BF	
	RRC A	0042	CB 0F	
	RRC A	0044	CB 0F	高位数据向 4～7 位置位
	RRC A	0046	CB 0F	
	RRC A	0048	CB 0F	
	LD B,A	004A	47	
	IN A,(02H)	004B	DB 02	
	RES 4,A	004D	CB A7	RSEL=低电平
	OUT (02H),	004F	D3 02	
	IN A,(02H)	0051	DB 02	输入低位 4～7 位为 0 定 8 位数据
	RES 7,A	0053	CB BF	
	ADD A,B	0055	80	

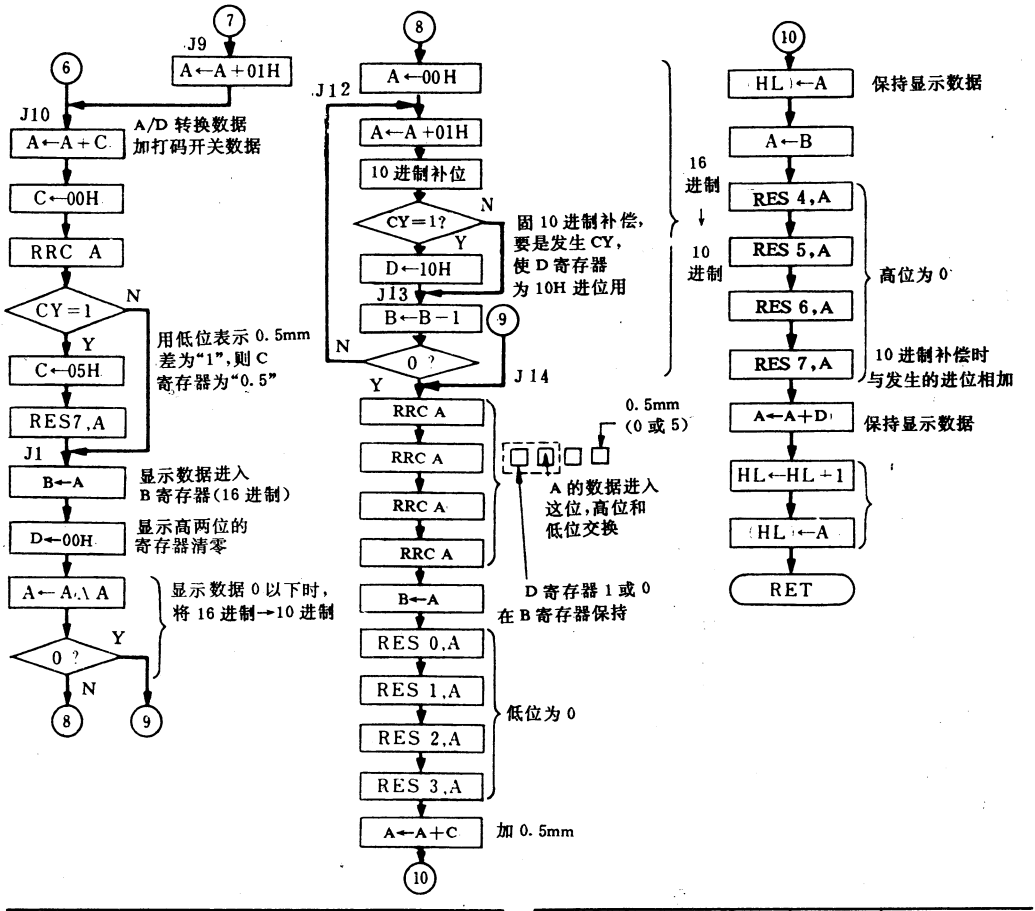
表 4.(2



标号	助记符	地址	机器语	注释
	RRC A	0056	CB 0F	舍掉最低位
	RES 7,A	0058	CB BF	
	CP C	005A	B9	比较前次数据
	JP Z,J3	005B	CA 64 00	
	LD C,A	005E	4F	更新比较数据
	LD D,04H	005F	16 04	比较次数置位
	JP MEAS	0061	C3 1B 00	
J3	DEC D	0064	15	连续 4 次吗?
	JP NZ,MEAS	0065	C2 1B 00	
	IN A(02H)	0068	DB 02	原点调整用按钮
	RES 7,A	006A	CB BF	开关输入置位
	OUT (02H),A	006C	D3 02	
	IN A,(02H)	006E	DB 02	按钮开关
	BIT 1,A	0070	CB 4F	输入检验
	JP NZ,J4	0072	C2 76 00	
	LD E,C	0075	59	原点数据
J4	LD A,C	0076	79	尺寸数据
	SUB E	0077	93	
	JP NC,J5	0078	D2 7C 00	负号尺寸为 0
	XOR A	007B	AF	
J5	LD C,A	007C	4F	

标号	助记符	地址	机器语	注释
J5	IN A,(02H)	007D	DB 02	打码开关
	SET 6,A	007F	CB F7	输入置位
	OUT(02H),A	0081	D3 02	
	IN A,(02H)	0083	DB 02	
	LD B,A	0085	47	
	XRO A	0086	AF	
	BIT 3,B	0087	CB 58	检验打码
	JP Z,J6	0089	CA 9E 00	开关 S_3
	BIT 2,B	008C	CB 50	检验打码
	JP Z,J7	008E	CA A0 00	开关 S_2
	BIT 1,B	0091	CB 48	检验打码
	JP Z,J8	0093	CA A2 00	开关 S_1
	BIT 0,B	0096	CB 40	检验打码
	JP Z,J9	0098	CA A4 00	开关 S_0
	JP J10	009B	C3 A6 00	
J6	LD A,28H	009E	3E 28	+80mm
J7	ADD A,28H	00A0	C6 28	+60mm
J8	ADD A,28H	00A2	C6 28	+40mm
J9	ADD A, 28H	00A4	C6 28	+20mm
J10	ADD A,C	00A6	81	A/D 转换数据加
	LD C,00H	00A7	0E 00	打码开关数据

表 4.(3)



标号	助记符	地址	机器语	注释
	RRC A	00A9	CB 0F	
	JP NC,J11	00AB	D2 B2 00	
	LD C,05H	00AE	0E 05	最低位为 0.5
	RES 7,A	00B0	CB BF	(保持显示数据)
J11	LD B,A	00B2	47	(16 进制)
	LD D,00H	00B3	16 00	
	AND A	00B5	A7	显示数据为 0 时
	JP Z,J14	00B6	CA C6 00	16 进制→10 进制
	XOR A	00B9	AF	
J12	ADD A,01H	00BA	C6 01	
	DAA	00BC	27	16 进制→10 进制
	JP NC,J13	00BD	D2 C2 00	
	LD D,10H	00C0	16 10	
J13	DEC B	00C2	05	
	JP NZ,J12	00C3	C2 BA 00	
J14	RRC A	00C6	CB 0F	
	RRC A	00C8	CB 0F	高位与低位交换
	RRC A	00CA	CB 0F	
	RRC A	00CC	CB 0F	
	LD B,A	00CE	47	
	RES 0,A	00CF	CB 87	低位为 0

标号	助记符	地址	机器语	注释
J14	RES 1,A	00D1	CB 8F	低位为 0
	RES ?,A	00D3	CB 97	
	RES ?,A	00D5	CB 9F	
	ADD A,C	00D7	81	0.5mm 数据相加
	LD (HL),A	00D8	77	保持显示数据
	LD A, B	00D9	78	
	RES 4,A	00DA	CB A7	
	RES 5,A	00DC	CB AF	高位为 0
	RES 6,A	00DE	CB B7	
	RES 7,A	00E0	CB BF	
	ADD A,D	00E2	82	加进位
	INC HL	00E3	23	保持显示数据
	LD (HL),A	00E4	77	
	RET	00E5	C9	
OUT ABP	LD A,(HL)	00E6	7E	B 端口
	OUT (01H),A	00E7	D3 01	输出低两位
	INC HL	00E9	23	
	LD A,(HL)	00EA	7E	端口 A
	OUT(00H),A	00EB	D3 00	输出高两位
	RET	00ED	C9	

3. 测量尺寸及显示方法

现在讨论将 UP－80 与数字游标卡尺连接起来(图 3),对其测量尺寸进行数字显示的方法。

从本刊上期的讨论,对照简单的流程图,读者能够理解 A/D 转换的方法。

表 3　SEL L,H 信号工作关系

	PC_6(SEL1)	PC_7(SEL2)
TC5090P 的 D_4 ～ D_{11}	L	H
打码开关	H	低电平高电平均可
TC5090 的 EOC 按键开关 BUSY	L	L

数字游标卡尺测量范围在 0～120.0mm,精度 0 为0.5mm 的显示程序示例,如表 4 所示。

上述程序中,MEAS 子程序有它较方便的特点。即初始化将 00H 置于 E 寄存器,调用子程序之前,将存放 A/D 转换结果的地址置于 HL 寄存器对时,测量值自动地进入 HL 寄存器对所指定的两个地址。例如,HL 寄存器对指定 00F0H 地址,测量结果 10.25mm 时,00F0H 地址低位存 25,00F1H 地址高位存 10。

实际操作方法说明如下:

(1)数字游标卡尺原点调整与最大值调整这里讨论 0～40mm 测量范围的操作方法。此时,打码开关部件的打码开关全部为不接通状态。测量范围改变后,要接通打码开关部件,请参考表 5 并按下述要点进行。

①各部件的肘式开关方向现图 3 所示一致。

②照图 4 程序清单顺序写入程序

③接通电源

④若写入完毕,将写入、读出控制部件的肘式开关①搬至读出侧,检查写入的程序是否有差错。

⑤各部件的肘式开关与表 6 所示一致。

⑥按动中央处理部件的复位开关,时间为 1 秒左右。将位于写入。读出侧的肘式开关④搬至实行侧。

⑦搬动滑动电位器的滑动件,将它移动到原点调整侧。

⑧按动接口印刷线路板上的原点调整开关,于是,4 位数字显示组件显示 0 0 0 0 0。

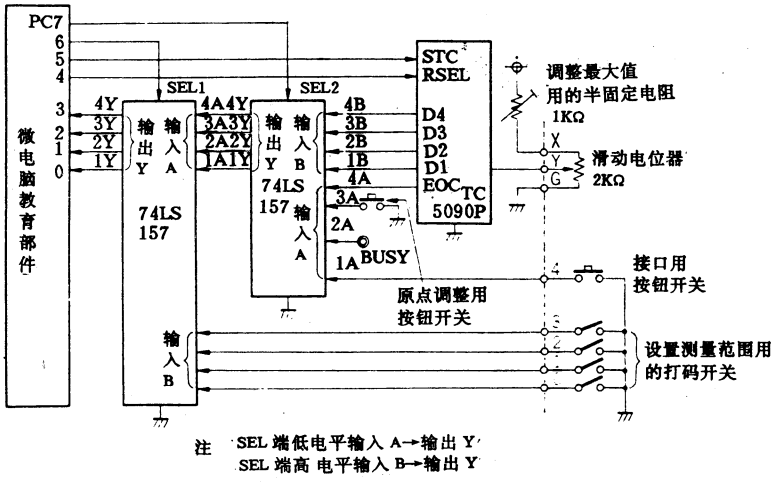


图 2

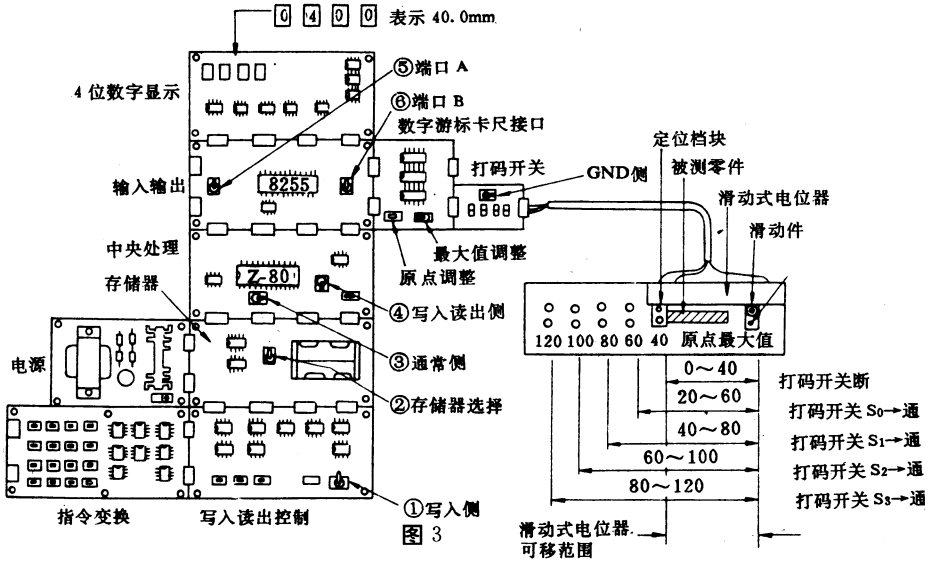


图 3

表 5　测量范围与打码开关的关系

测量范围 (mm)	打码开关的状态 (0→通 X→断) S_3	S_2	S_1	S_0	挡块的位置	原点调整显示	量大值调整显示
0～40	×	×	×	×	40	0000	0400
20～60	×	×	×	○	60	0200	0600
40～80	×	×	○	×	80	0400	0800
60～100	×	○	×	×	100	0600	1000
80～120	○	×	×	×	120	0800	1200

表 6　开关的关系

部件名称	开关 No	开关方向
写入读出控制	①	读出侧
存储器	②	存储器选择
中央处理	③	通常测
	④	写入、读出侧
输入输出	⑤	端口 A 侧
	⑥	端口 B 侧

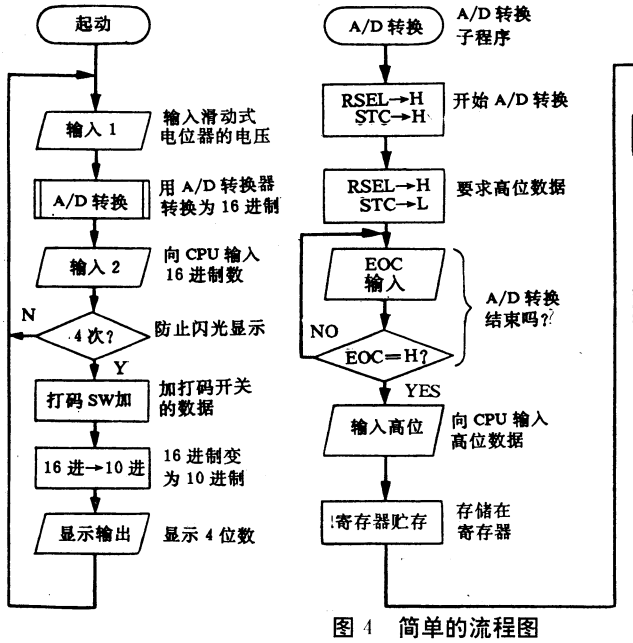


图 4　简单的流程图

⑨搬动滑动电位器的滑动件，将它移动到最大值调整侧。

⑩调整接口印刷线路板上的调整最大值半固定电阻，直至 4 位数字显示部件显示出 H 0400 为止。

(2)测量

①如图 3 所示，将被测零件与挡块重址放置。

②松动手柄，将滑动电位器的滑动件朝被测零件方向慢慢移动

③滑动件接触被测另件时，4 位数字显示部件显示测量尺寸。

该数字游标卡尺分辨率为 0.5mm，略为粗糙一些。不加改进将它应用于实际测量存在一定问题。但是，可以认为对于我们学习把测量尺寸输入给微电脑，进而数字显示还是具有充分肯定的价值。

要提高数字游标卡尺的分辨能力，一是要求滑动电位器具有较高的可靠性，二是要提高电源的稳定程度。对此有兴趣的读者不妨请进行制作尝试。

＊＊

(上接 33 页)　高四位数除以 5 的商乘以 12，加上低四位数除以 8 的余数的处理数，再加上音调偏移量，即可以从音频表 FREQ 中查出时间常数，送定时器产生音乐。需要说明的是，表一中每个音符是分散的。为了查出同一调下各音符的时间常数，需要把低四位除以 8 的余数转换为查表值，暂称为余数处理数。它等于同一调下，同一音高中在表 1 中的位置。譬如下调的 5，余数处理数是 8，显然，不论什么调，不论音高，“哆”音的余数处理数均为 1，“咪”的均为 3，等等。

如果低四位数除以 8 的余数为 0，则表示休止符，刺叭不响。为表现得更自然，不采用关定时器的方法。而是预置时间常数为 FFF9H。如为休止符，时间常数不变，产生超声波。否则根据音频表，产生音乐。

4. 停止

高四位为 0 表示停止。程序指向 0000H 如有监控则返加监控，重置 RAM05 单元，奏另外的歌曲。事实上，很简单的改动即可使程序连奏成重奏。比如，此时的低四位为 0 表示退出，非 0 表示重奏次数，为 F 表示连奏(奏下一首歌)。

5. 如何编写乐谱表

(1)乐谱表首址存放在表 TAB 中

(2)定调。由表二查出偏移量作为第一个字节。

(3)由表三、表四将乐谱译成数据。举例说明：

|3̲5̲ 1̲·2̲ 3 · 0̲|

其中的“1”为中音，节拍 3/4，由表三查出高四位为“8”；使用节拍 2，从表四查出低四位数为“9”，故为 89H。整段乐谱数据是：7BH，7DH，89H，62H，83H，28H。

(4)结尾必须是 0×H。

初级程序员水平考试辅导讲座
微机数据录入操作问答

刘英　张茂芝

1. 微机数据录入操作的含义是什么？

数据录入的全称是计算机辅助数据录入(CADE——Computer Aided Data Entry)，又称数据转换(data conversion)。数据是信息的载体。由于常见的数据是文字、数字、图表、符号等，所以用计算机处理数据就必须进行数据录入——将数据转换成机器的可读形式。这里数据录入与传统打字不完全相同，它必须按照一定的要求，遵循某种约定(如格式的约定、控制字约定等，)借助于计算机进行操作。

随着计算机技术的迅速发展，数据录入越来越显得重要，从六十年代开始数据录入迅速发展，至今已形成一个数据录入行业，为建立和更新数据库、办公室自动化管理、出版印刷业的电子排版印刷等方面服务。

数据录入常用设备主要有：数据录入专用设备、微型计算机、中英文电子打字机等。作为初级程序员主要指能在微机上(如 IBM PC 及其兼容机)进行数据录入操作。

2. 数据录人操作的基本要求是什么？

数据录入操作的基本要求是完成用户对作业(录入任务称为作业)的要求，保证录入数据的正确性。

为此要求录入人员具有很好的素质。例如，具有一定的中文水平和英文阅读能力；了解计算机的软硬件基本知识，能熟练地进行键盘操作；能够识别中英文书写体；熟悉校对符号和排版规则；能熟练使用常用的录入软件；以及具有紧张、细致、认真、负责的工作态度等。

3. 数据录人操作的要领主要有哪些？

数据录入操作是一门技术。掌握这门技术需要注意下面几点。

(1)培养对键盘录入的兴趣，树立不怕艰苦的工作作风。数据录入这项工作在初学时，难免感到单调乏味，可惜又无捷径可循。所以要培养对此项工作的兴趣，精力集中，不怕苦，从易到难，由简入繁，持之以恒，循序渐进地按正确的操作方法和顺序坚持练下去。当能独立上岗进行录入工作时，要不断总结经验，提高录入的速度和质量。

(2)保持良好的录入姿势，减轻工作的疲劳程度。在进行数据录入时，录入员的较好姿势应该是平坐在高度合适的椅子上，两腿平放在桌下，光线从左面过来。录入材料最好放在带有滑尺的架子上。录入员两肘悬空手腕平放，手指自然下垂，轻放在键位上，小臂与大臂间略小于 90°。只要桌椅适宜、姿势正确，那么就可以减轻录入中的疲劳，提高录入速度。

(3)掌握击键要领，眼睛不看键盘。打字的要点不在理解而在熟练。如果击键要看键盘，那么就会出现看了稿件看键盘，击键后眼睛又回稿件寻找内容，目光反复移动的情况。这种情况除浪费时间外，还容易造成眼睛疲劳，丢字串行等问题的出现。因此在录入时应做到眼睛不看键盘，手能不加思索地，自动地把所看字符“弹击”出去。这里说的“弹击”键码是击键的要领。录入时两手指尖接触键码的部位不是指甲，而是指端的圆肚部分。看清字符后就应用手指果断地“弹击”相应的键码。弹击键时点到为上，不要用力太大，弹击键码后手指迅速退回原位。击键要均匀有节奏。

(4)掌握基本指法及键位。通常人们把字母键 ASDFJKL 和标点；键称为基本键位(也有称作原位键或导键的)。其中 ASDF 四个键是左手的小指、无名指、中指及食指的原位键；JKL；是右手的食指、中指、无名指和小指的原位键。开始击键时，每个手指应轻放在该指对应的原位键上，每击一键后，手指应迅速返回原键位。每个手指除有它的原位键外，还有其范围键。按键盘码分布，左手小指(A 指)击键范围在 ZAQ1 和左边的其它键，左手无名指(S 指)应打 XSW2，左手中指(D 指)打 CDE3，左手食指(F 指)打 VFR4 和 BGT5；右手食指(J 指)打 MJU7 和 NHY6，右手中指(K 指)打，K!8，右手无名指(L 指)打。LO9，右手小指(；指)打/；PO 和右边的其它键。空格键根据各自的习惯由两手的大拇指来打。上述是指法的基本内容，只要掌握好，就可提高击键速度和准确性。

4. 数据录人的全过程包括哪些步骤？

数据录入全过程大致可分为六步：

(1)明确录入要求

接受录入任务，首先要弄清录入要求。这里所指录入要求主要包括：估计字符数量，作业成品的准确率、作业交货日期、文件类型与格式、输出作业成品的方式等。只有明确录入要求，才能制订合理的作业流程，把握住作业质量和作业进度。

(2)选用合适的录入程序

对于不同格式的数据应选用不同的录入程序，如

果没有合适的录入格式要编制录入程序。一般情况下，①对于输出的成品是打印纸，原始材料是一般公文资料的录入作业，可借助于中文(或英文)Wordstar 等字处理软件来录入或编辑，打开文件的命令采用 D。②对于非公文资料的录入作业，如计算机语言的源程序，则应使用 Wordstar 中的 N 方式打开文件，或考虑使用行编辑软件 EDLIN 来录入。③对于由多个字段构成的记录，应采用关系数据库 dBASE 来完成录入。④对于录入作业中的一些特殊的或专用的格式要求，可在使用上述软件录入后进行单独处理，或自己编制专用录入软件来达到要求。

总之，录入软件的选用，既要考虑录入时，使用方便，提高速度，减少错误，又要考虑后期数据处理和输出成品的格式要求等。

(3)键盘录入

键盘录入是录入的主要工作。为此录入人员要熟悉键盘、指法熟练、严格按录入要求击键。在键盘录入中应注意以下几点：

①对大批量作业，可拆成多批，要编好顺序号，起好文件名，分批录入后再合并。

②文稿注意保存完好，不要丢失，作好每天的工作任务、文件名、字符数等项记录，便于以后核对。

③做好软盘备份。

④保持稳定愉快的工作情绪，克服环境不利因素的影响，集中精力进行键盘录入。

(4)校对

校对是数据录入不可缺少的一环。常用的校对方法有键校、目校、“盘校”等。

①键校是指录入员在已经输入的文件基础上再输入一遍。这个过程借助于一种校核软件来实现，一般的专用录入设备都配有这种软件。此校对方式最适用于英文或数字型作业。②目校是指用人工进行阅读检测，这种方法目前在中文录入时使用较多，但校对速度慢，准确率不太高。③“盘校”是指“对盘”校核，就是由两个录入员各打一遍文稿，分别存放在两张软盘上，然后借助软件来进行校核。这种方法的本质上是文件比较。常用的比较方式有两种：一种是按记录比较。两个文件的同一记录有不一致的地方就显示出错信息，通知录入员检查改正。记录划分可以是一定数目的字符，也可以是一行文稿。另一种比较方式是按字节为单位，把一个文件看成一个数据流，对字符逐个比较。当两个文件中的某一字符不一致时，光标就停在该字符位置上等待修改。以字节为单位进行比较的方法比以记录为单位进行校对速度要快质量要好。一般情况下进行“盘校”后文件大面积错误率可控制在万分之一左右，能满足字符准确率方面的要求。

(5)数据处理

数据处理是指对作业的记录结构、章节控制符、字体控制符、表格控制符、排版格式等方面的格式查错处理及文件合并等工作，通常可借助排版软件或专用程序来完成。这是作业录入中质量控制的最后一个关口，所以要仔细认真。

(6)输出成品

按作业输出要求，将录入好的数据输出。常用的输出成品方式主要有：①存入软盘、磁带等磁介质中供使用；②由打印机打印成文稿供使用。

以上是数据录入工作的主要步骤，在实际中有的步骤要重复或交叉执行。例如，经校对后发现较多错误，有时仍要再进行键盘录入，因此录入人员需注意提高素质和技能，减少错误，降低误码率，尽量避免或减少重复工作。

5. 数据录入中常用哪些字符编码？

数据录入业说的“数据”包括数字、字母、汉字和各种符号，统称字符。对于计算机系统来说不能“直接处理”这些字符，而是处理代表它们的数码，即代码。把这些字符按照某种规律和约定，编成一组数码，就是字符编码。在数据录入中常用的字符编码有中西文之分。西文字符常见字符编码是 ASCII 码、EBCDIC 码；汉字常见字符编码，不同国家有不同代码，我国汉字国家标准字符编码是 GB2312－80 等。

(1)ASCII 码是英文 American Stardard Code for Information Interchange 的缩写，意思是美国标准信息交换代码。标准 ASCII 代码是 7 位二进制代码，能表示 128 种字符，取值范围用十进制数值是 0 到 127，对应十六进制数值是 0 到 7F。ASCII 码的编码方式是有序的，0 至 9 十个数字在英文字母前面，52 个英文字母按先大写后小写及字典顺序排列；其余还包括 32 个控制符和 34 个专用符号。编码和字符之间是一一对应关系。扩展的 ASCII 码为 8 位二进制码，十进制数值从 0 至 255，对应十六进制数值 0 到 FF，共 256 个。IBM PC 及其兼容微型计算机均采用 ASCII 代码。

(2)EBCDIC 码是英文 Extended Binary－Coded Decimal Interchange Code 的缩写，意思是扩充的二－十进制交换代码。它可以表示英文字母、数字、符号及控制字符等 256 种字符的代码，EBCDIC 码也是有序代码，作用类似 ASCII 码，主要用于美国 IBM 公司设计的计算机。

(3)GB2312－80 是我国 1981 年颁布的《信息交换用汉字编码字符集——基本集》的国标号码，它是我国用两个字节代表一个汉字和非汉字图形字符，用以作为计算机系统之间汉字信息交换用的标准代码。该代码表纵向分成 94 个区，横向将每个区分成 94 个位置，区号由第一字节标识，位号由第二字节标识。十进制数表示的区号和位号构成了每个汉字或汉字图形字符集的“区位码”。字符集的排列顺序是：1－9 区为 682 个符号；10－15 区空，未使用；16－55 区为一级汉字，3755 个，属常用字，按汉语拼音排列；56－87 区为二级汉字，3008 个，属非常用字，按部首排列；88－94 区空，未使用。国家标准 GB2312－80 是汉字信息处理的基础代码标准之一，目前已广泛用于我国通用汉字系统的信息交换及硬软件设计工作中。

6. 汉字输入方法是如何分类的？

汉字输入计算机的方法有多种，但可以分为两大类：一是通过计算机键盘人工输入，二是让计算机自动识别。计算机自动识别可以是利用计算机抽取汉字字形特征，实现对汉字的手写体或印刷体的自动输入；也可以是利用语音分析技术，抽取汉语语音特征，实现对汉语语音的自动识别。这种自动识别都由机器将字形或读音转换成代码，然后加以识别，以达到将文字或读音输入计算机的目的，这种方法目前在我国使用还不普遍。

通过计算机键盘，人工键入汉字编码进行输入是目前应用最广泛的汉字输入方法。根据数据录入人员获取汉字编码信息特征又可将汉字输入编码分为音码、形码、音形码三类。①音码是指采用声、韵、调的信息特征，用全拼、双拼、简拼、混拼等方式键入汉字，通过词、词组乃至语法、语义、语用知识区分同音词。例如CCDOS中使用的缩简拼音法、联想汉卡下使用的全拼音及双拼双音法等属于此类音形码。这种码对于熟悉汉语拼音的人是非常容易使用的，但重码多，不易盲打。②形码是指采用整字、字根、笔画、笔画组、字形结构等信息特征，利用专用键盘或通用键盘将汉字输入计算机中。五笔字型、长城笔形码等属于这一类。③音形码是指采用音和形两方面的信息给汉字编码。例如声韵部形码、见字识码等属于此类。这些编码方案各有千秋，所以使用时应注意选用较好的输入方案。

7. 字处理软件 Wordstar 中英文版本有何异同点？

英文版 Wordstar 是一个较成熟的字处理软件，适用于各种文书资料的输入、编辑、存贮和打印工作。具有①编辑功能，对输入文本的任何部分可随时进行修改、插入和删除；②查找和替换功能，对正在编辑的全文可快速查找或替换指定的字或短语；③显示功能，能在屏幕上打入文本的任何一部分供编辑修改；④文本块功能，能将单字、句子、段落、整个文件进行块移动、复盖、删除等；⑤屏幕格式编排功能，能设定边界和行宽，将标题居行中位置，重编段落等；⑥打印功能，能进行页面格式设计；⑦菜单提示和求助功能，根据用户需要提供详略不同的菜单及操作说明信息；⑧文件存贮功能，可在操作的不同时刻将打入的文本等以文件形式存入磁盘；⑨邮件合并功能，可由同一封信产生发往不同地址和不同收件人的信件，并能进行文本插入及文件连接；⑩校对拼写功能，能利用字典对全文的单字自动检查拼写错误，作错误标记及改错等。因此，英文版 Wordstar 是在西方国家和我国西文字处理工作中使用最早，应用最广泛的字处理软件之一。

为适应我国汉字信息处理的需要，机电部六所等单位西文 WS 进行了适当改造和汉化工作，产生了 C—Wordstar 汉字文字处理软件，简称 CWS。CWS 的基本功能和操作命令与西文 WS 基本相同，其差异主要有以下几点：①CWS 不具有汉化校对拼写功能；②由于 CWS(11 行)屏幕上显示 1 个汉字，需占用西文显示时的 4 个字符位置，所以屏幕显示功能不如西文 WS 方便；③由于一个汉字在屏幕上显示宽度占西文显示的两个字符位置，所以 CWS 的标尺行宽度的一半等于该行汉字的个数，并避免半个汉字的出现，标尺行宽度应取偶数；④西文文本中单字由字母组成，单字间由空格或标点隔开。中文文本中字、词间不一定有间隔，所以在使用 CWS 中的 ctrl＋A，或 ctrl＋F 等以字为单位的移动光标命令时容易产生错误的光标定位，上述命令不易使用；⑤由于汉字打印驱动程序的使用的差别，使得在西文 WSF 控制特殊打印效果的标记的操作命令有的失效，有的有较大变化，因此在打印汉字信息时，要注意参考所使用打印驱动程序的有关说明。

8. 字处理软件 WS 使用中常见术语的含义是什么？怎样使用？

中英文 Wordstar 有许多特性可以使文本输入工作简单化。为叙述方便，操作手册上采用了一些术语，这些常见术语的含义和简单使用方法如下。

(1)文本文件——在编辑中，需要遵循通常的文字书写规则(如自动换行、分页等)的文字资料文件在 Wordstar 中称为文本文件。常见的有书信、文章、论文、报告等。

(2)非文本文件——在编辑中，不能根据通常格式规则自动处理的数据资料文件在 Wordstar 中称非文本文件。常见的有表格、源程序等。

由于 Wordstar 将通常的文书资料分成文本文件和非文本文件两类，所以在打开文件时方式也不一样，在 WS 开启菜单中使用 D 打开文本文件，使用 N 打开非文本文件。两类文件的存盘命令没有区别。

(3)字绕转——输入文本时，当输入字符超过标尺右界时，WS 自动处理换行的过程称为“字绕转”。CWS 的标尺长度省缺值是 70，即每行可书写满 70 个字符或 35 个汉字，超过时实现“字绕转”。标尺长度可使用 ctrl＋O＋R 或 ctrl＋O＋L 改变，改变长度时要注意。输入中文标尺长度为偶数，否则产生字绕转会出现半个汉字。

(4)对齐——当输入一段文本时，字绕转在字之间自动插入适当空格，使文本右边成一直列，此过程称为“对齐”。这里值得注意的是：在使用 CWS 时，进行汉字和非全中文方式的 ASCII 字符混合输入，行末的 ASCII 英文单词或数字不实现“对齐”，需要采用人工干涉。

(5)硬空格——按空格键产生的空格称为硬空格。硬空格是文本文件的一个组成部队，只有使用删除命令才能删除。段落开头的空格可由硬空格产生。

(6)软空格——软空格是在字绕转或文本重排时产生的，WS 能有选择地自动插入或删除这些软空格。

(7)硬回车——每段结束时按回车键产生的回车称为硬回车。WS 屏幕右边标志列上有标记“<”。使用

ctrl+B 命令重编文本时，是以硬回车为单位进行的。

(8)软回车——由字绕转自动形成的，WS 屏幕右边标志列为空白。

编辑文本文件时，在输入自然段落结束时打回车键，即使用硬回车；段落内的换行由字绕转产生的软回车实现换行。编辑非文本文件时不能用软回车，由字绕转实现自动换行，必须每完成一行文本输入就打一个回车键。

(9)重排文本——又称重排段落，是指按屏幕标尺将修改排列不整齐的段落进行格式排版。两个硬回车之间为一个段落，排版后每行宽度由按下重排命令 ctrl+B 时的屏幕标尺给定。使用 ctrl+B 前应将光标移到要排段落首字符前，使用 ctrl+B 后光标自动停在下一段开始位置，可连续使用 ctrl+B 重排若干段落。

(10)圆点命令——插在文本中的打印命令。它的标志是用圆点(·)开始于文本行的第 1 列，点后跟两个英文字母，及一个空格或适当参数。圆点命令占一行，由硬回车结束，其功能对每一打印页都起作用。

9. Wordstar 有哪两种重要的编辑功能？

Wordstar 的两种重要的编辑功能是指对文本内容进行查找与替换和文本块操作。

(1)文本内容的查找与替换

在编辑文本过程中进行查找是指从文本中找出指定的字符串，字符串总长度不超过 30 个字符(或 15 个汉字)，使用 ctrl+Q+F 命令实现；替换是指从文本中先找出指定字符串，再用给定的内容替换查找到的字符串，使用 ctrl+Q+A 命令实现。查找和替换命令中有六个选择项，可灵活组合使用。

(2)文本块操作

文本块是预先用命令定义的文本内容的一部分。一个文本块可以是一个句子、一段文章或整篇文章。文本块简称块。块大小是指以块首标记到块尾标记所包含的文本内容。块命令可以简化许多编辑工作。Wordstar 允许使用块命令对文本块进行移动、复制、删除等。

10. 数据录入用哪个软件最好？

该问题最常见。有人说 WS 好，有人说 HW 好，也有人说 PE 好，……，众说不一。

在数据录入中究竟选用哪个软件最好，我们认为不能一概而论，建议从以下几方面去考虑。①所选软件要能满足录入作业的要求，特别要考虑利于作业录入后期的数据处理工作。②所选软件要操作方便灵活，编辑功能强，增加、删除、修改、打印等性能好。③如果录入中文作业所选软件汉化程度要好，适合使用录入人员所选用的汉字输入方案。④如果录入表格方面的作业所选软件制表功能要强，操作要方便。⑤所选软件价格是否便宜，录入人员是否熟悉等。

目前在 IBM PC 及其兼容机上有多种编辑软件、字处理软件、窗口软件等可用来做数据录入，如前面提到的 WS、HW、PE、EDLIN、dBASE 等都可以回来做录入软件，究竟用哪个还要根据录入作业的要求和录入人员使用的录入环境确定。如：长城系列机使用 HW 非常方便，而没有安装长城的微机只能考虑用 WS 等软件；又如联想汉字系统上配备的汉化 PE，编辑功能强，打印文稿方便自如，但没装联想汉卡也就不能使用；又如有些数据库使用 dBASE 作录入软件十分方便。总之，初级程序员应在熟悉两种编辑软件的基础上，尽可能地去多了解和掌握一些其他常见编辑软件、字处理软件、窗口软件、工具软件的使用方法和技巧，这样对复杂的数据处理作业就可以方便地确定使用一个软件或使用几个软件交叉完成数据录入。

个人转让 Apple－Ⅱ 计算机 1 台

配置：主机，软盘驱动器及卡，彩色显示器(NEC)，软件及资料。保证质量。

价格：2200 元

联系人：湖南省长沙市省电子分析所　万大范

邮码：410001

转让 CEC－Ⅰ 中华学习机

配置：CEC－Ⅰ型中华学习机(陕西产)　1 台

软盘驱动器(日本原装)　1 台

Z80 卡　1 只

可编程音响合成器(CEC－PSG)　一只

(有使用说明书)

价格：1750 元(购机时间一年，如有意者赠送全套有关书籍，数张软件)

联系人：陕西省安康市张岭水电三局科研所　李宗卫

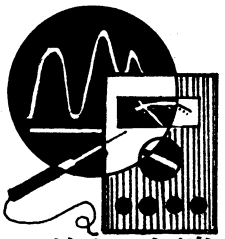
邮码：725011

(上接 45 页)

内部的两个小槽中，此时恢复弹簧键体的原状再用。“3”的方法，用万用表检查弹簧键体是否已修理好。

7. 如果还有其它失灵的键体，都可照以上办法进行修复。全部修好后，照拆卸的相反顺序依次装回主机，你就能正常使用。

只要发生在弹簧键体内的毛病，如：弹簧弹性变差等，都可用此方法进行修复。对于不能修复的弹簧键体，可用不常用的键体进行对换，如 TAB 键。注意：只换键体，不换键帽。



维修经验谈

软盘驱动器机械故障的分析与维修

广西桂林工商银行计算中心　魏学则

软盘驱动器故障大体上分为机械部件故障和电路器件故障，而机械故障率占软驱故障的90%，也就是说软盘驱动器故障有90%是属于机械方面出现异常。

这些故障归纳起来有以下几个方面：

（一）磁头脏

由于长期使用不按时用清洗盘清洗磁头，或使用质量较差的软盘和受潮，发霉的软盘，造成磁头上积磁，污物太多，影响软驱的读写功能。在读写软盘时主机显示器上显示该软驱读错误或写错误，出现这种故障后再去用清洗盘清洗磁头已不能解决问题。应将软驱从主机上取下，用棉签蘸上录音机或录像机清洗磁头用的高级磁头清洗剂，反复轻擦上下磁头，上磁头可用小镜子观察清洗情况，下磁头很直观，擦至光亮无污物即可。注意擦洗过程用力不易过重过猛，以免使磁头变位。

（二）磁头偏

磁头偏一般都是上磁头偏，下磁头偏的情况较少。这种故障现象表现为对一块装有DOS内容的盘片，DIR可以正常显示目录，但执行某一文件时，出现该软驱读错误，按A、R、I键绕过错误，这种故障也是因为使用发霉或较脏的软盘，这些软盘表面有污物已失去光滑感，磁头落在上面，盘片高速转动将磁头带偏。

判断软驱读写错误故障是否属于磁头偏这一类的方法是：找一块新软盘插入有故障的软驱，格式化该软盘，之后拷贝一些可执行文件，再去执行某一文件。如正常则说明该写电路没问题，此故障属磁头偏。

借助猫眼盘或磁头定位程序将磁头重新调试定位是最好不过。如果没有这些工具，可凭经验手动定位。我们知道盘片是顺时针旋转，那么磁头多偏向右边，用手指扶到软驱上磁头，轻轻往左带，边调边试，直至正常读写标准盘

如果上述方法不能解决，就放松磁头小车上固定磁头支架的两颗螺丝，将磁头向左推0.5mm左右，然后试读写盘片，要细心调试，动作不可太大。

（三）上磁头不到位

当软驱插入盘片压下仓门杠杆后，磁头落不到盘片上，出现这种故障其现象与磁头偏相类似，只能列出目录而不能读，写其中任意文件，或格式化时只有0磁头工作，显示“0”头某某道，而不显示“1”头某某道。出现这种故障时，把软驱拆下，不插盘片关上磁盘仓门，可以以软驱侧面观察到上磁头在下落到一定位置后就不再下落，与下磁头保持有5－7mm的距离，这样上磁头是不可能读到信息的。

我们从软驱上面观察，当你将软盘仓门杠杆压下后，联动上磁头起落杆也落下，使上下磁头相吻合。出故障时上磁头起落杠下落不到正常位置，致使上磁头落不到位。

处理方法是：在软盘仓门杠杆压下的情况下，用手去压磁头起落杆至上磁头刚好落到下磁头上。然后在杠杆联动的各轴上加注少量缝纫机油。正常情况下，上下磁头与盘体之间的压力是靠磁头与支架之间的弹性簧体来调节的。不必耽心磁头对盘片的用力过大或过小。但两磁头在读写时要相吻合。

（四）读、写时，运行到内道出错：

软驱在读写时，1.2MB软盘走到70道左右，360KB软盘走到30道左右开始出错或磁头小车出现“咔咔”响声。

这种故障出在磁头小车运行轨道上。在软驱被面可以看到有两条光亮的园轴，磁头支架在其上面前后运动。这轴上积有灰尘后，被磁头小车推到轨道的顶端，久而久之轨道顶端的灰尘将磁头小车卡住在某一部位，使其不能运行到轨道的顶端。造成读错误，或写不满盘等。

清理两轴上的灰尘，加注少量缝纫机油，故障即可排除。

软盘驱动器，无论任何型号，其电路部分各有所异，但机械部分都相类似，所以大家在检查一台有故障的软盘驱动器时，首先判断是电路故障还是机械故障，然后把故障压缩到某一部位，修起来也就方便了。

更换集成块一法

吴广元

在维修过程中，更换印刷电路板上的集成块是一件很麻烦的事。下面介绍的方法，无需专门设备，就能很容易地更换集成块，且不会损坏电路板。

一、在离电路板尽可能高的位置剪断待换下集成块的全部管脚。

二、用烙铁逐个溶化电路板上已剪断管脚的焊锡，并用尖嘴镊子将管脚拔出。

三、准备若干根不沾锡且溶点高的小细棍（木、竹等），如果手头暂时没有，用大头针代替也可以。用烙铁

依次溶化电路板焊孔中的锡，将准备好的小细棍插入电路板上的焊孔中。

四、锡凉后，拔去电路板焊孔中的小细棍，焊孔就完全清理干净了。

五、插入待换上的集成块，用烙铁焊好并剪齐管脚即可。

APPLE-Ⅱ机内存故障维修方法

滕　达

通常APPLE-Ⅱ机损坏的最大可能情况有两种，一种是主机电源损坏，一种是内存集成块损坏。

内存集成块损坏是指动态RAM(4116)的损坏。损坏轻者造成图形显示不完整，DOS调不进内存；重者一开机就启动不了。下面介绍一种方法能准确地查出损坏的集成块(4116)，使计算机恢复正常。

APPLE－Ⅱ机的RAM分为三排：

第一排C3－C10　　地址：$0000～$3FFF

第二排D3－D10　　地址：$4000～$7FFF

第三排E3－E10　　地址：$8000～$BFFF

C3-C10,D3-D10,E3-E10,是主机板上集成坏的坐标。

具体方法如下：

①打开计算机电源，如果不能启动，屏幕上出现杂乱图形，那么应先检查第一排(C3-C10)，最简单的方法是替换法，即将C3-C10八块4116用好的4116替换，直到故障排除为止，没有其它简便方法。如果仍不能排除故障，则应检查第六支ROM(F3)

②如果打开计算机电源后能够启动，但是在作图状态下，屏幕不能完全清屏，那么就说明第二排RAM有损坏，地址($4000～$7FFF)

在APPLESOFT状态下输入HGR2，将屏幕消除，然后CALL－151进入监控状态。在监控状态下查看$4000的值，如果第二排RAM损坏，则$4000的值就是非0，将$4000的内容由16进制转化为二进制，例如$4000的值为$15转化为二进制后则为00010101。这个二进制数的八位数分别用第二排的八块4116 RAM来贮存。具体的对应如下：

母板坐标：D3 D4 D5 D6 D7 D8 D9 D10

内存单元：第0位 第1位 第2位 第3位 第4位 第5位 第6位 第7位

以$15为例：1 0 1 0 1 0 0 0

出现1的位置所对应的集成块就是损坏的RAM，如上例可知。D3 D5 D7这三块4116损坏，只要将其拆下换上了新的集成块即可，其余的类推。

③如果计算机启动后，DOS调不进内存，使得驱动器不能使用，则有可能是第三排的RAM损坏，地址$8000－$BFFF

在APPLESOFT状态下输入HGR和HGR2，然后用CALL-151进入监控状态。在监控状态下输入，$B000＜2000，2FFFM $8000＜2000，4FFFM，将$8000－$BFFF清为0，然后读出$8000的值，如果非0，则说明第三排RAM(4116)损坏，具体判断损坏集成块位置和②相同。

巧修CEC－Ⅰ受损键盘

黄文贵

中华学习机CEC－Ⅰ型键盘上的69个键都是弹簧键体。在使用过程中，人们频繁地按动部分键造成个别键失灵，或者发生意外冲击。现象是：按动这些失灵的键时，屏幕上无任何显示。这些失灵的键，完全可以自己修复。现将修理步骤介绍如下：

1.接通学习机电源，同时按下CTRL—RE SET—TEST三个键，最后释放TEST键，对主机进行自检，自检内容在《用户使用手册》、《中华学习机CEC－Ⅰ技术参考手册(硬件)》书有所介绍。当自检全部正常时，可进一步确定故障发生在键盘上。

2.断开学习机电源，旋下外壳上的六颗螺钉，取下外壳上盖，拿出键盘。键盘右侧有一根26线的扁平电缆，此缆线与主机相连。小心取下接线卡，使键盘脱离主机，以利于修理。

3.在键盘的印刷电路板上，每个键的名称都标得非常清楚。用万用表的欧姆档进行检测，两表笔放在失灵键的两个焊点上，注意接触应良好。正常的键，没按动时电阻值R应为无穷大(即万用表指针不动)，按动时电阻值应为零值。失灵的键，电阻值始终为无穷大，用此方法进一步确定故障发生在弹簧键体上。

4.取下失灵键上的键帽，即标有A、B、C、……字母，向外稍用力拉就能取下。用电烙铁、医用的吹耳气球(注：搞摄影的人一般都具有此工具)。去掉弹簧键体两脚上的焊锡。用电烙铁加温，气球吹去焊锡。注意：温度应合适，时间不宜太长，以防损坏印刷板。焊锡一定要去干净，使弹簧键体两脚脱离印刷板。然后，又在键盘表面上用两把中等型号的扁平螺丝改刀放在弹簧键体的两旁，适当用力向上挑，就能取下弹簧键体。

5.此弹簧键体是密封的，仔细观察形状，你就能很容易拆下键体的外壳。一定要小心拆卸，里面装有一根弹簧和两块铜片，还要注意内部的形状和安装位置，以利于顺利安装。即使搞错，你只需稍加观察，也能装上。

6.拆开弹簧键体后，你会见到两块铜片，一片像“Π”型，一片像“山”型。据自己的修理经验，认为拆动“山”型铜片较合适，把“山”型铜片的两边向外拨动，目的是使两铜片能良好接触。然后分别把两铜片装入到

(转至43页下)

新书与软件

VIRUSCAN 病毒处理软件包功能简介

朱　禹

VIRUSCAN V61 版病毒处理软件包功能强大，它可免疫、检测、清除到目前为止已发现的 82 类病毒及它们的许多变种病毒，是目前处理计算机病毒最有效的软件。

该软件包是由美国 McAfee Associates 发行的，它包括十六个文件(8 个执行文件)，它们功能如下：

1. SCANRES——计算机病毒免疫程序，它可以被放置在批处理文件(AUTOEXEC・BAT)中，并驻留内存，开机时它将以 BOOT 扇区、硬盘分区表、隐含文件及各子目录下的所有文件包括其自身进行检测，看是否带有病毒。在计算机正常工作中，每当你装入程序时它对被装入的程序也进行同样的检查；若检查到有病毒存在，它将显示警告信息和所感染病毒的名字。若在你的批处理文件中装入了 SCANRES 程序，你的计算机将免受病毒感染，也就是说你的计算机对病毒具有免疫能力。

2. SCAN——计算机病毒检测程序。它可以检测到已发现的所有病毒。检查的范围可以包括硬盘分区表、BOOT 扇区、文件分配表、子目录下的所有文件及隐含文件。在检查时有多个选择项可供选择。比如：只对内存进行检测，只对 COM 文件进行检测或只对某一病毒进行检测等等。这些选择有助于提高你检测的效率。

3. CLEAN—UP 病毒清除程序。该程序可以清除 SCAN 发现的文件中的所有病毒。在清除过程中有多个选择项可供选择。这些选择可以用一个命令清除多块磁盘上的病毒，也可以清除一个具体程序上的病毒，除此外还介绍了自 57 版以来所发现的各种病毒的症状及主要危害。

4. MDISK——BOOT 扇区、分区表上的病毒清除程序，有些病毒感染了 BOOT 扇区、分区表是不能用 CLEAN 来清除的，对于不同的操作系统要使用不同的 MDISK。它可以是 MD40、MD33、MD32 或 MD30 之一，它们分别适用于 DOS4. 0、DOS3. 3、DOS3. 2、DOS3. 1(或 3. 0)。

5. NETSCAN 是 VIRUSCAN 的网络检测程序，它可以检测 PC 网络或工作站上的病毒。NETSCAN 有多种选择项可供选择，它可以检测多个驱动器上的多种病毒，也可以对一个驱动器或一个文件进行单独检测，检测出病毒后可以用手工单独的清除，也可以自动地清除多个病毒。

6. VALIDATE——证实程序，它是用于证实上述五个程序的完整性的。VALIDATE 可以检查上述五个程序是否被窜改过，该程序以两种不同的方法产生循环检查以确保其高度安全性，产生的结果可与 McAfee Associates 所公布证实数据进行比较，以确定上述五个程序的完整性。

7. VIRLIST——病毒特性表，在该表中给出了已发现的 PC 机所有病毒的名字，它们传染的范围，被感染文件增加的长度、清除它们的方法和它们的破坏性。

8. REGISTER——注册说明程序。这一程序中给出了使用上述软件的注册方法，从该程序中可以看出美国软件的使用注册的方法和有关价格，对我们了解美国软件市场的情况有一定的帮助。

VIRUSCAN 的每个执行程序都有说明与使用方法，想进一步了解的读者可与电子工业出版社软件部联系(北京 173 信箱软件部，邮码：100036)。

Microsoft Windows 3. 0 窗口软件简介

Window 3. 0 是美国 Microsoft 公司 90 年在原 2. 0 基础上推出的窗口软件，它扩充了 DOS 的操作方式，将改变 PC 机用户使用个人计算机的方式，解脱用户记忆理解 DOS 命令以及各种选项的负担。Windows 用下拉菜单和对话框代替了 DOS 的提示符，将各种实用程序表示为图标画面，许多命令可以通过鼠标的移动很直观的完成。尤其对计算机的初学者，提供了非常友好的界面。

Windows 为 PC 机操作系统提供了一个很好的多任务环境。它充分利用 CPU 空闲时间，允许同时运行多年程序，并为每个程序各开辟一个窗口，且各窗口之间可以方便地交换信息(图象、图形和文字)，通过这种信息交换的方式，可以将在不同应用程序中所做的工作合并到一个单独的文件里。

Windows 非常适用于办公事务处理，它提供了许多办公经常要做的功能，如资料卡、计算器、日程表及模拟时钟等，完成了办公的自动化。所以，也有人称 Windows 是办公桌系统。

Windows 程序包提供了三个基本软件：write（文书处理）、paintbrush（彩色绘图）和 terminal（终端）。这三个软件较充分地显示了 Windows 的特点。Write 是一个具有图形功能的字处理系统，它提供了多种字体、多种字形，可将图形、文字同时在屏幕上显示，并且与打印结果完全一致，即“所见即所得”。Paintbrush 是一个多功能彩色画面设计软件。使用 paintbrush 可以绘制出具有多种线型、多种背景、多种色彩并可与字符同时显示的画面。这些画面可以在屏幕上被任意移动、翻转、放大和缩小。Terminal 提供了终端仿真、文件的传递与接收功能等。这种功能能使用户的微机可以成为其它计算机的终端，并可以灵活地与其它计算机进行信息交换。

Windows 有着优越的内存管理功能，并且在其环境中的程序设计是与外部设备完全独立的。Windows 不象 DOS 系统，只有经过专门编程才可以使用 640K 以外的扩展内存，在 Windows 环境中，用户可以直接使用扩展内存，最大可以使用到 16MB。在 Windows 环境下编程，可以直接使用该环境提供的各种图形元素（如出口、下拉菜单、控制、对话框、图标、字形变换、字体变换和 Bitmap 等），用在帮助信息生成程序及其它工具，可编写出具有在线帮助的交互式图形界面程序。用 Windows 写出的程序，具有对外设（如显示器、打印机等）的独立性。即同一个程序，可以不加修改地直接用于 EGA、VGA 或 CGA 等显示适配卡上，也可以不加修改地直接使用不同的打印机、硬盘等，而一般的 DOS 程序，总要注明适用于何种显示适配卡、打印机、硬盘等。

总之，Windows 不仅可以使您充分利用计算机的潜力，赋予更多的新功能、新特性，而且其丰富的图形界面将在 PC 机上为您提供更直观、更高效的工作环境。

最新版本 PC TOOLS V6.0

美国第十三大软件公司（Central Point Software）最新推出 PC TOOLS V6.0 版本，这是一个功能强大，界面友好，使用方便的工具软件，是微机管理与办公自动化必备之工具！

PC TOOLS V6.0 由三大部份功能组成：

1）数据管理与维护

2）硬盘备份

3）桌面管理系统

数据管理与维护包括下列程序模块：

PC SHELL：这是一个在简便易用的窗口环境下，具有全部 DOS 维护命令功能的实用程序，提供拷贝，删除和比较文件功能，文件和磁盘编辑功能，以及目录管理功能，系统盘和数据盘格式化功能。PC SHELL 可按原格式显示大多数数据库建立的文件，制表软件和字处理软件建立的文件。另外，PC SHELL 还可恢复已删除的文件，显示常驻内存分布图，以及在 PC SHELL 状态下运行其它可实用程序，PC SHELL 可以常驻内存，或做为一个单独的实用程序使用。

DiskFix：这是一个可以修复软/硬盘各种问题的实用程序，其功能强大，易于使用，修复工作全部自动完成。

Mirror/Rebuild：用于防止意外删除数据和硬盘格式化。Mirror 在一个专用的隐藏文件中保存文件分配表和硬盘根目录，一旦意外地格式化了硬盘，用户可以快速恢复数据。Rebuild 可以尽可能地恢复 Mirror 所记录的硬盘状态，以便有效地挽回数据删除和格式化磁盘所造成的损失。PC SHELL 的 Undelete 功能可以利用 Mirror 建立的删除跟踪信息，自动恢复被意外删除的文件。

PC－Cache：通过把最常用的信息存在计算机中来提高硬盘的存取速度。

Compress：一个独立的实用程序，用于检查并消除硬盘的存储碎片，它也可以检查硬盘坏区，并把文件移出这些坏区。

PC Format：可格式化硬盘和各种密度的软盘并可用 Rebuild 恢复这些盘中的数据。

PC Secure：用于保密机密数据和程序。PC Secure 可以在磁盘上加密，解密，压缩和隐藏文件。PC Secure 使用 DES 加密系统，使数据完全随机化，没有密钥实际上是不可能解密的。若使用压缩选项，PC Secure 可以使绝大部份文件所占磁盘空间减少 25％到 60％。

硬盘备份功能由 PCBACKUP 模块组成，可对各种磁盘或磁带进行备份，支持对数据的压缩及自动进行备份的功能。

桌面管理系统具有强大的文字处理及办公自动化的功能，可同时支持 15 个窗口的操作，其中的 Notepads 字处理器提供了类似 Wordstar 的字处理功能，包括文本的剪裁，粘贴和拷贝，可进行拼写检查。Outlines 功能提供了对信息进行结构化管理的手段。数据库功能提供了数据库管理与查询的手段，实现了与 Dbase 和 Foxbase 的数据共享。

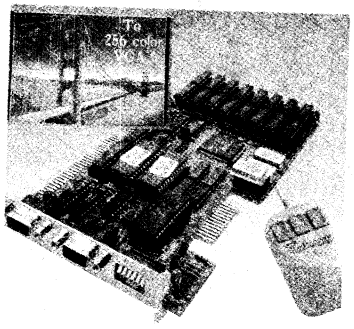
活动计划功能对于有效地安排工作日程计划，准时赴约可以有很大帮助。

桌面系统还支持远程通信功能，可使用 Modem 实现文件传送，也可使用 FAX 接口板进行远程传真服务。同时具有 HP－11 科学运算型，HP－12C 函数型，HP－16C 程序设计员和财会型计算器功能。

電子與電腦 5

一九九一年

总期第74期



電子與電腦

目录

·综述·
90年代的电脑技术和采购策略(下) …… 殷广济(2)
CGA,EGA,VGA …… 余 红(3)
计算机技术的产物——条形码技术 …… 于江苇(4)
·PC用户·
通用打印机状态检测程序的改进 …… 夏伟文(5)
通用制表程序 …… 赵 欣(6)
也谈Cdbase Ⅲ的纵向送数方式 …… 王传信(9)
PCTOOLS的使用技巧 …… 刘建明(11)
程序常驻内存(TSR)的实现方法 …… 李 华(12)
防止非法使用DBASE程序的一种方法 … 黄世刚(14)
也谈IBM机图形存取 …… 焦小友(14)
侦错核对程序 …… 张永辉(16)
·学习机之友·
让CEC—BASEC也有自动行号功能 …… 沈东山(17)
图形屏幕坐标的对应存储单元 …… 葛海东(18)
神奇的$COBO单元 …… 李 峰(18)
怎样解读CEC—I学习机的汉字系统 …… 刘文勤(18)
用机器语言解八皇后问题及其推广
…… 卢祖兴 刘超宏(19)
也谈汉字平移显示 …… 李茂学(19)
前台显示后台绘图的秘密 …… 谢吉华(20)
演示抛体运动 …… 白梅兰(21)
多单元电视接收天线设计程序 …… 史文忠(22)
一个实用的造型表编辑程序 …… 左震宇(23)
·C语言初阶讲座·
第五讲 程序控制结构(一)
…… 李文兵 王玉华 蓝智斌(25)
·电脑巧开发·
语言报时钟的原理与实现 …… 蔡莲红(28)
·学用单片机·
简易单片机电子琴 …… 张培仁 张新宇(30)
MCS—51单片机在听力播放、发射系统中的应用
…… 沈乃才 宋振环(32)
·学装微电脑·
用微电脑检查印刷电路板 …… 易齐干(37)
·初级程序员级水平考试辅导讲座·
BASIC语言程序设计问答 …… 李志刚 吕品(44)
封二、APPLE IIe维修资料 …… 香港余健康
封三、中国电子器材公司特区公司集成电路价目表

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:北京市邮政局
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:徐云鹏
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

90年代的电脑技术和采购策略(下)

北方电脑公司　殷广济

80486微处理器由一个高度优化的和80386处理器兼容的处理器和80387兼容的数学协处理器及8KB的数据与程序同一的cache合为一体,共有120万个晶体管(80386为25万个),采用1微米的CHMOS 1μ工艺制作。具有32位的内部体系结构以及32位的地址总线和数据总线。在单个周期常用的指令,如存取指令的执行,使用流水线作业及RISC技术,采用一种新的脉冲串数据传送机制,由存储器顺序阅读4个32位字,以保持芯片上数据及指令高速缓存经常装满,由于晶体管之间讯号传送距离大为缩短,80486的运算速度比80386芯片快5倍,工作于33MHz时,达到20VAXMIPS,可充分满足如AUTO CAD和Lotusl—2—3等软件的数学运算中所需的浮点运算功能,其性能足以与IBM3090主机媲美。

80486芯片是与80X86系列产品高度兼容,换言之,它可以执行为IBM PC机编写的,价值估计高达150多亿美元的众多通用软件及工程应用软件,并为DOS,Windows/386,OS/2,UNIX Systemv/386提供高水平的性能。与DOS兼容是Intel公司为80 X86系列微处理器发展重心,所以,80486的基本设计要求会有效地应用于IBM PC系列及兼容机中。

据称,Intel公司花在80486芯片上的投资超过3亿美元和4年时间,虽然并不是每个人都看好这颗芯片,认为它必能在众多的微处理器芯片中脱颖而出,成为小型机和工程工作站的中流砥柱,但无论如何,大部分专家都同意,80486将会是半导体发展历史上的一个精心杰作,而且将会赋予明天的台式电脑真正易于使用的运行环境。你要买高档的32位电脑时,应认真考虑80486系统。

总之,微处理器技术不断朝着集成度高,速度快、功能多、体积小的高档领域迈进。

由于世界交通发达,人员往来增多,人们企望有一种比台式电脑小的、能携带,可使用直流电源、能进行文字及事务处理的电脑。80年代开发膝上型(Laptop)及书本型(notebook)袖珍型(handhold)等便携式电脑并开始普及使用,正以爆炸速度赶上台式电脑,也许会成为电脑世界的骄子。

便携式电脑要求体积小巧,主板尺寸也必须相对缩小,故对芯片集成度要求较高,另一个更重要的要求在电力消耗方面,因此,此类芯片组都必须具备电源管理功能。

芯片组必须对电脑的耗电元件包括磁盘机、显示器、CPU以及一些机械动作进行完善的电源管理,电脑系统操作时,系统内部的各元(组)件,并非时时都处于运行状态,如果对这些不在运行状态的组件进行供电控制,就可延长整个系统的操作时间,此类电脑是从各元、组件一点一滴地把电力节约下来的。

液晶显示器模块所消耗的电力几乎是整个系统的一半,通常以外围芯片来侦测其使用状态,进而控制对液晶显示器的供电。

CPU是另一个耗电元件,通常以切断对其供电及降低频率两种方法来达到省电目的。将CPU的电源切断,CPU内的资料必须映射到存储器,以便在对CPU加电后继续执行工作,降低CPU频率是另一种省电方法,且内部资料亦能保存下来。

便携式电脑的有些特性是台式电脑所没有的,故标准的BIOS并不完全适用于便携式电脑,必须加入标准BIOS的欠缺的部分,便携式电脑具有如下特性:

CPU档次从8088、80286到80386 SX,内存1MB左右并可扩充,采用3.5英寸软磁盘和2英寸硬磁盘,其容量在20MB以上,通常为40MB,液晶显示,EGA/VGA标准,亦有采用气体/等离子技术,电源大多采用高效镍镉电池,充电后使用寿命均在4—8小时,重量在3公斤左右,膝上型稍重些。目前国内销售价格在8—9000元到2000多元之间,视其品种PC,XT,286,386不同而异。

中国情况有些特殊,也许不会有那么多人拿着便携式电脑转来转去,但在地质勘探,远航交通,办公室自动化及军队方面可能会有一定用途和相当大的驰骋空间。联想到近几十年来电脑工业的飞速发展及集成电路的更大规模集成,购买便携式电脑也许5—10年的期限内,不会为你使用的电脑的优越功能失去而感欣慰。

由于人工智能,特别是计算机辅助设计(CAD),计算机辅助工程(CAE)的兴起和软件开发的需要,要求有一种作为大型计算机在网络中的节点机,它具有良好的图形处理功能以便设计人员使用,这就是工作站。工作站的产品可依照其处理机的性能,存储器容量,显色能力,解析度,网络输入输出速度等分为初级、传统、超级工作站和绘图超级工作站等四大类。参见下表:

性能 \ 类别	初级工作站	传统工作站	超级工作站	绘图工作站
CPU 性能(MIPS)	2—12	4—15	8—20	20—40
FPU 性能(Mflops)	0.5—1.5	0.5—2.0	2.0—16	16—50
存储器(MB)	4—16	8—64	8—128	16—1024
硬盘容量(MB)	40—600	80—600	600—1300	1024 以上
显示屏尺寸(英寸)	12—19	12—19	15—19	19
显色能力	16—256	256—4096	256—16700	256—16700
解析度	1024*768 —1280*1024	1100*900 —1280*1024	1280*1024	1280*1024
网络速度 mbits/sec	10	10	10—125	10—1000

工程工作站主要应用领域如下：

应用化学：分子模型、量子化学、计量模型、程序工程、化学设计

工业自动化：离散程序控制、连续程序控制、自动测试装置

设计自动化：机械、电子/电机、建筑、工程

地球资源：地震分析、油田探测、地图

实时系统/资料收集与控制：建筑物自动控制、交通控制、大气监控、水质和污水控制、领空控制

绘图/动画/影像：电视/影片/动画/影像处理、电视传输

软件开发：电脑辅助软件工程(CASE)，人工智能及其它软件开发

电脑辅助排版：电脑辅助技术排版、电脑辅助电子排版系统

医药：身体扫描、病人监控

从用户角度看，工程工作站比起大型、中型、小型电脑系统来价格低廉，并可以通过网络共用系统资源，减轻大、中、小型电脑系统的负荷，所以愿意购买工作站。

目前工作站市场主导者是厂家还不是用户。主要的供应商有 SUN，HP/APDLLO，DEC，Inter Graphic 和 Silicon Graphic。然而未来超大规模集成电路技术，开放性系统，电脑辅助软件工程以及面向目标处理技术的发展将会改变目前市场的势态，高性能工作站功能会更强，中级工作站价格会下降，以 80486 为基础的 PC 系统对目前的工作站在价格上会取得优势，所以廉价的 PC 是否也会成为工作站系统是很值得人们关注的，其关键在于某些特殊工程用工作站的价格能否大幅度下降与高性能 PC 竞争，或是以 PC 为基础的廉价系统的功能能否和工作站一样令人满意，人们期待 PC 将成为明日工作站的开始。未来充满挑战和期待。

可以预见未来工作站的特点如下：(1)MIPS 将转换成 BIPS(每秒执行数十亿指令)；(2)存储容量由 MB 转换成 GB(数十亿字节)；(3)外存装置将由 GB 转成 TB(数百亿字节)；(4)用户使用软件时与硬设备无关；(5)工作站将有标准的用户界面；(6)工作站与多媒体及视听装置相结合；(7)UNIX 系统将成为工作站的套装软件之一。

用户不仅会考虑优异的性能/价格比，也会对应用软件的多寡，升级特性，可扩充性及厂家的可信度，服务及是否采用标准作为采购准则。

CGA、EGA、VGA

余　红

随着第一台微型计算机的诞生，市场对显示系统的需求永不满足，其性能在计算机科技人才的“勤奋＋科学＋应用”创新精神的努力下，一代代翻新，永无止境。目前，众多的电脑公司，纷纷推出高性能的显示卡。例如：

最早的 IBM PC 仅提供两种显示卡，即“单色显示卡”(简称 MDA)和 CGA 卡。MDA 只为单色显示器设计的，可显示 25×80 的文(数)字。分辨率为：720×350，而 CGA 可显示分离的红、蓝、绿信号，也可显示混合的视频信号，还可以显示以点绘制的图形以及 ASCII 码字符，其分辨率为 640×200。由于 MDA 只能显示文(数)字，而许多程序必须在屏幕上显示色彩，所以使用者选择了有彩色绘图功能但文字的清晰度较低的 CGA 卡。随着高层次微型计算机的推出，IBM 又发展出另一种提供较高文字及绘图分辨率的 EGA 卡，可以模拟 MDA 和 CGA，还可以在单色显示器上显示以点绘制的图形，可产生 16 色文(数)或 640×350 图形图象，其分辨率比 CGA 要高，更重要的是 EGA 同时显示的文字和图形远比 CGA 清晰，能使许多高级软件发挥其功能。以后又推出了“显示图形阵列”即 VGA 卡，

VGA 的程序设计界面和 EGA 相似，许多对 EGA 写的程序不经修改即可在 VGA 上运行，其分辨率比 EGA 高，文本方式下为 720×400，图形方式为 640×480，它可以从调色板中同时产生 256 种颜色。其他 800×600/16 色，800×600/256 色，1024×768/16 色的 VGA 卡花样翻新，图形功能强，使微计算机可显示的彩色画面，色彩缤纷。

由于 IBM PC 显示卡不支持汉字显示，国产长城计算机根据 IBM 的几种显示卡，研制出适合汉字显示的长城显示卡。如早期的长城微型计算机采用 IBM 的 CGA 卡和 014 汉字显示方式，原长城 014 板的显示方式有两种：字符方式，分辨率为 640×504，80×28 字符，16 种颜色。图形方式：分辨率为 640×480，8 种颜色。它主要配置于 GW0520CH 上，显示 25 行汉字外加三行汉字提示行。由于 014 方式与国际上流行的 CGA 及 EGA 方式有较大差别，不兼容。长城又开发出了另一种 CEGA 显示卡，它继承了 014 方式的汉字字符方式显示，并且与国际上流行的显示标准 CGA，EGA，VGA 高度兼容。其分辨率为 640×350，16 色图形显示方式，可以共享更多的软件，它还可以实现中西文图文叠加。CEGA 板也包括两种工作方式：方式 1 和方式 2。其中：

方式 1：兼容原长城高分辨率彩色图形适配器(014 板)的汉字，图形显示以及 IBM EGA 的中低分辨率的图形显示。

方式 2：兼容 IBM EGA 的所有显示方式，并增加显示图形汉字的功能。

CEGA 卡配置于长城 286，386 机型，通过 CEGA 板上的开关以及功能调用等两种途径，可以灵活地切换这两种工作方式，

近一年长城新近推出的 CVGA 卡，全兼容 IBM VGA，并具有图形 640×480/256 色，800×600/16 色，800×600/256 色，1024×768/16 色等增强功能。全兼容长城 CEGA 汉字显示标准：支持字符/图形迭加功能，支持 16×16 点阵/24×24 点阵汉字字库的软切换。具有新型的单区 4 字节汉字方式，使汉化工作变得更简便，特别适于 DOS4.0，WINDOW 等高级多窗口软件，其配置于长城 386 以上的机型。

综上所述：

IBM PC 常见的显示卡有：

CGA 卡：彩色图形卡，英文名：Colar Graphice Array.

EGA 卡：加强型图形卡，英文名：Enhanced Graphics Adapte.

VGA 卡：显示图形阵列，英文名：Video Graphics Array.

长城微型计算机中对应的显示卡有：

014 板：长城高分辨率彩色图形适配器

CEGA 卡：中文增强型图形卡，Chinese Enhanced graphics Adapter.

CVGA 卡：中文显示图形阵列，Chinese Video Graphics Array

显示卡作为计算机的“视觉神经”，将会越来越丰富发达。

(封面上给出的是外国公司有关 VGA 卡的图片资料)

计算机技术的产物——条形码技术

山东省科学院情报所　于江苇

许多人可能已注意到，商店的货架上有许多商品上印有条形码标志。这种标志的作用是什么？为什么要使用这种标签呢？我想这个问题应该从计算机的数据输入方法谈起。

大家知道，计算机的运行速度相当可观，现已达到每秒钟几亿次；另一方面，由于计算机数据输入技术发展还不尽人意，因此人们一直在寻找一种快速、准确的自动输入技术，为了适应这种需要，条形码技术就从此诞生了。

一、条形码符号的基本结构及种类

一个完整的条形码符号是由空白区、起邕位、信息位、校验位(可选)、终止位和空白区组成。如图所示：

空白区	起始位	信息们	校验位	终止位	空白区

空白区：顾名思意即没有任何印刷符号和信息的区域。它用来提示扫描器准备扫描条形码。

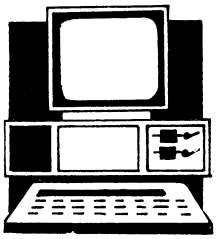
起始位：在扫描阅读时，当判定有起始位存在时，扫描器才会继续处理起始位以后的一系列信息。起始位的使用，给条形码增加了保险。

终止位：使用终止符，可避免不完整信息的输入。

校验位：它是校对读入的信息是否有效所采取的一种手段。

以上我们所谈到的是各种条形码制所具有的共性，而具体到每一种码制本身的具体特性来说，比如其密度、长度、连续性、宽条的个数、奇偶性等京有其不同之处了。条形码的种类在近四十年内逐渐增加起来，但就其本身的特性及实用性来讲，常见的有：

1. UPC 码(Universal Product Code)：它是一种只包含数字的商品零售代码，它被广泛应用于各种(下转 43 页)



PC用户

通用打印机状态检测程序的改进

夏伟文

本刊在一九九一年第二期上发表了《通用打印机状态检测程序》一文，对于不了解程序原理的人，在实际应用时很容易把参数搞错。为了解决这个问题，作者又把程序作了进一步的修改。

程序的原理在上面提到的文章中写得很详细，这里就不介绍了。新程序的改进之处是应用了数据文件，简化了判断用户打印机参数的过程，具体方法如下。

（一）用下面的程序 1 来判定用户所使用的打印机的状态参数，也就是运行该程序时根据提示将打印机分别置于正常、断电、缺纸和未联机等状态下即可将不同的状态下状态寄存器的数值（十进制）存入数据文件 PRCH. DAT。

（二）将 PRCH. DAT 拷入当前目录中，就可以使用下述的程序 2 对打印机状态随时测试。

程序 1：

```
10 OPEN"CHPR. DAT"FOR OUTPUT AS#1'打开数据文件
20 FOR I=1 TO 4
30 READ A$
40 CLS:LOCATE 10,18:PRINT A$;"然后按任意键…"
50 IF INKEY$="  "THEN 50
60 REM 测定参数
70 DEF SEG=&H40          '指向 BIOS 数据区
80 B=PEEK(9)*256+PEEK(8)+1'求状态寄存器的地址
90 A=INB(B)              '求状态参数
100 WRITE #1,A           '将参数写入数据文件
110 NEXT I
120 CLOSE #1             '关闭数据文件
130 SYSTEM
140 DATA"请将打印机置于正常状态，"
150 DATA"请将打印机置于未联机状态，"
160 DATA"请将打印机置于断电状态，"
170 DATA"请将打印机置于缺纸状态，"
```

程序 2：（以 TH 3070 打印机为例）

```
5 REM 文件名 CHPR. BAS
10 SCREEN 0,0,0:WIDTH 80:CLS:KEY OFF
20 OPEN"CHPR. DAT"FOR INPUT AS #1'打开数据文件
30 FOR I=1 TO 4:INPUT #1,A(I):NEXT I'从数据文件读入参数
40 CLOSE #1'关闭数据文件
50 DEF SEG=&H40'指向 BIOS 数据区
60 B=PEEK(9)*256+PEEK(8)+1'求状态寄存器的地址
70 IF INP (B)=A(1) THEN 130'如果打印机准备好则转到 110 句退出 BASIC
80 BEEP:COLOR 7:K=1'否则发出警报并在屏幕上显示出提示
90 IF INT(B)=A(2) THEN COLOR 7,0:CLS:LOCATE 10,21:COLOR 20,1:PRINT"注意";:
COLOR 2,0:PRINT"请按打印机面板上的 SELECT(联机)键!":GOSUB 150:BEEP:GOTO 90
100 IF INP(B)=A(3) THEN COLOR 7,0:CLS:LOCATE 10,21:COLOR 20,3:PRINT"注意";:
COLOR 4,0:PRINT"请打开打印机电源开关!":GOSUB 150:BEEP:GOTO 100
110 IF INP(B)=A(4) THEN GOSUB 140:GOTO 110
120 IF INP(B)<>A(1) THEN COLOR 7,0:CLS:LOCATE 10,25:COLOR 20,1:PRINT"注意";:
COLOR 15,0:PRINT"打印机尚未准备好…":GOSUB 150:K=K+1:
IF K>7 THEN 70 ELSE 120
130 CLS:COLOR 7:SYSTEM
140 COLOR 7,0:CLS:LOCATE 10,27:COLOR 20,2:PRINT"注意";:COLOR 14,0:
PRINT"请给打印机装好纸!":PRINT:PRINT TAB(12);:
PRINT"然后看面板 SELECT(联机)和 PAPER END(缺纸)指示灯是否正常!":GOSUB 150:
BEEP:RETURN
150 FOR I=1 TO 3200:NEXT I:RETURN
```

该程序的应用主要是面向于那些对计算机原理一无所知的录入员和初级操作员，他们在使用应用软件时，如果打印操作有误，操作系统会给出一些英文指示，这时往往令操作者们手足无措。有了这个程序，可以用汉字信息帮助他们改正错误，以免由于误操作而造成死机或丢失信息；尤其是在应用广泛的 dBase 系统中开发应用软件时，调用该程序可确保打印功能的正常实现。

上述程序已在 GW－386/20、GW0520－CHII 型机上调试通过。

通 用 制 表 程 序

济南　山东省机械工业学校　赵欣

一个管理程序要具有实用性，应使用户享有充分的“自主权”。例如打印数据库中哪些字段，打印符合什么条件的记录，应由用户决定。我在为某单位编制管理程序时，编制了一个具有上述功能的通用制表程序，它可以对任意以汉字作字段名的数据库进行制表打印，数据库的结构改变时，仍能按改变后的库进行打印，而制表程序本身无需修改。现将该程序介绍大家，愿能起到抛砖引玉的作用。

一、程序的功能及设计思想

首先介绍几个数据库：1 区是需制表打印的数据库，其库名由用户键入，在程序中为 &na（下简称原库）。2 区是打印库 PRDB2. PRG，该库是全 C 型字段，各记录值与原库完全相同，但字段宽度有所改变。3 区是打印库的结构库 STRU3. DBF，其中 FIELD__NAME 字段存放的是打印库（也是原库的）字段名，FIELD__LEN 字段是打印库各字段的宽度。4 区是表线库 LINE4. DBF，字段名与打印库完全相同，各字段值都是一段表线，库中只有 3 个记录，分别用来产生上表线 la、中表线 lb 和下表线 lc。5 区是条件库 COND5. DBF，只有一个字段名为“条件”的 C 型字段，存放各种打印条件。

1. 生成打印库和表线库

这一功能是由过程 PRLI 来实现的。它依据原库生成其结构库，再对结构库做三点改造。第一：描述类型的 FIELD__TYDE 字段值全改为 C；第二：描述字段宽度的 FIELD__LEN 字段值改为值宽度和栏目名（即原库字段名）宽度中较大的。例如：性别字段，字段值“男”或“女”只占 2 字符宽，而栏名“性别”占 4 字符宽，因此相应的 FIELD__LEN 值取 4；第三：由于制表符占 2 字符宽度，所以还要把 FIELD__LEN 的值都改为偶数。这样修改后的结构库就成了打印库所需的结构库。在此基础上又生成了表线库。为了条件打印的需要，还将原库和打印库建立了关联。

2. 产生打印条件

调用过程 PCOND 可产生打印条件，也可以无条件打印。用户可以自己键入打印条件，也可用条件库中已有的条件。在设计程序时一旦用户提出增加或修改打印条件，只需修改条件库就可以了。

3. 分页制表

该程序可分页制表，使用宽或窄打印纸。每页行数、起始页号、页标题及制表时间全由用户决定。

4. 用户任选打印字段

主程序第 31 至 44 行将原库的所有字段（也是打印库的字段）均显示出来，供用户按项号选择需打印的字段。（该项号也是 STRU 库的记录号）并且可以随时显示当前的打印宽度，当用户不想再选项号或已超过纸的宽度时会自动停止选项，开始打印。

二、几点说明

1. 关于打印库的形成

一般使用 APPEND FROM 语句要求当前库的字段名和类型与源数据库要一致，但实验证明，用该语句可以生成打印库，不管原库中相应字段是 N、C、L、D 都不会丢失数据。

2. 非汉字字段名的处理

由于本制表程序是用原库中的字段名作为表中的各栏目名，若是非汉字字段名可以修改 STRU3. DBF，在该库中再增加一个 C 型字段存放所有原库字段的栏目名，再对程序稍加修改就可以了。

3. 打印内容的取中

为了使各项打印内容能打印在本栏的中间位置，还可再修改过程 PRLI，由于各栏的宽度均存在 STRU3. DBF 库的 FIELD__LEN 字段中，而打印库各字段实际宽度又可用 TRIM 和 LEN 函数配合求出，只要根据这两个宽度就可将打印库中的各字段值取在中间位置。

4. 打印条件 &COND

主程序中 LOCATE FOR 语句中的打印条件是通过宏代换产生的，但有些版本的 CDBASE－Ⅲ不能对这种宏代换产生的逻辑表达式进行正确解释。这时可按下面的思路修改程序。第一步：先建一个语句库，其中只有一个 C 型字段，库中的第一条记录的值是“LOCATE FOR”＋COND，这式中 COND 是由过程 PCOND 产生的打印条件。第二条记录值是“RETURN”。第二步：由语句 COPY TO C:DTOP. PRG SDF 就可以产生一个外部过程。这种用数据库文件生成一个程序文件的方法，经实验证明完全可行。第三步：在主程序适当处调用 C:DTOP. PRG 就可以了。

```
1:* *用户任选字段、条件的通用制表程序 c:prna.prg
2:clear
3:clear all
4:set talk off
5:set device to screen
6:set print off
7:set procedure to c:proc
8:do prli
9:p="y"
10:do while p="y" .or. p="Y"
11:clear
12:@2,5 say "请开打印机，针头对准纸的第一行"
13:a=1
14:w=170
```

```
15;pn=1
16;pa=66
17;@3,5 say "1 宽纸 2 窄纸;请输入 1 或 2" get a pict "99"
18;@4,5 say "请输入分页行数(默认每页 66 行)" get pa
19;@5,5 say "起始页号(默认值 1)" get pn
20;read
21;if a=2
22;w=80
23;endif
24;set margin to 0
25;bt=space(60)
26;ti=space(11)
27;@6,1 say "请输入制表时间" get ti
28;@7,1 say "请输入报表的标题" get ti
29;read
30;bt=trim(bt)
31;sele 3
32;r=0
33;clear
34;go 1
35;do while .not. eof()
36;c=0
37;do while(.not. eof()) .and. ((c+len(trim(FIELD__
NAME)))<=76)
38;s=str(recno(),2)+trim(FIELD__NAME)+"  "
39;@r,c say s
40;c=col()
41;skip
42;enddo
43;r=r+1
44;enddo
45;@ 7,0 clear
46;@ 8,1 say "请按项号选择需打印的项目(输入 0 结束)"
47;z="|"
48;p="|"
49;s="z+"
50;sd="|"
51;sc=0
52;la"┌"
53;lb="├"
54;lc="└"
55;do while .t.
56;h="  "
57;@ 9,10 say "输入需打印的项号" get h pict "999"
58;read
59;if h=" 0".or.h="0".or.h=" 0".or.h="000"
60;@ 7,0 clear
61;@ 8,10 say "请停止输入项号"
62;exit
63;else
64;go val(h)
65;f=FIELD__LEN
66;f1=len(trim(FIELD__NAME))
67;st=trim(FIELD__NAME)+"+q+"
68;if f=f1
69;sbt=trim(FIELD__NAME)+"|"
70;else
71;f=(f-f1)/2
72;sbt=space(f)+trim(FIELD__NAME)+space(f)+"|"
73;endif
74;sc=sc+FIELD__LEN+2
75;12=len(sb+sbt)
76;if 12>=w
77;@ 7,0 clear
78;@ 8,10 say "已超过打印纸的宽度,停止输入项号"
79;exit
80;else
81;s=s+st
82;sb=sb+sbt
83;fn=trim(FIELD__NAME)
84;sele 4
85;go 1
86;la=la+&fn
87;go 2
88;lb=lb+&fn
89;go 3
90;lc=lc+&fn
91;sele 3
92;@ 7,0 clear
93;@ 7,1 say sbt
94;@ 8,1 say "目前的打印宽度是:"+str(sc+2,4)
95;endif
96;endif
97;enddo
98;ls=len(s)-3
99;s=substr(s,1,ls)+"+z"
100;12=len(sb)-2
101;sb=substr(sb,1,12=+"|"
102;j=len(la)-2
103;la=substr(la,1,j)+"┐"
104;lb=substr(lb,1,j)+"┤"
105;lc=substr(lc,1,j)+"┘"
106;cond="  "
107;do pcond
108;set print on
109;set device to print
110;L=len(la)
111;cb=(1-2*len(bt))/2
112;sele 1
113;locate for &cond
114;do while.not.eof()
115;eject
116;@ prow()+1,cb say chr(27)+"ID"+bt
117;?? chr(27)+"IA"
118;@ prow()+1,L/2 say "第"+str(pn,3)+"页"
119;@ prow(),L-15 say ti
120;@ prow()+1,0 say la
121;@ prow()+1,0 say sb
122;do while (.not.eof()) .and. (prow()<=pa-5)
123;sele 2
```

```
124:@ prow()+1,0 say lb
125:@ prow()+1,0 say &s
126:sele 1
127:continue
128:enddo
129:@ prow()+1,0 say lc
130:@ prow()+1,0 say "  "
131:*下一句是为不致漏打最后一行而设
132:?? chr(13)+chr(10)
133:pn=pn+1
134:enddo
135:if prow()<pa-1
136:@ pa-1,0 say "  "
137:?? chr(13)+chr(10)
138:endif
139:set print off
140:set device to screen
141:@ 7,0 clear
142:@ 9,10 say "继续打印吗? Y/N"
143:set console off
144:wait to p
145:set console on
146:enddo
147:close proce
148:clear all
149:clear
150:set talk on
151:return

1:**过程文件 c:proc.prg
2:procedure pcond
3:*生成打印条件
4:sele 5
5:use c:cond5
6:store "  " to c,cl
7:m="y"
8:do while m $ "Yy"
9:clear
10:@ 3,10 say "1.条件打印 2.无条件打印 请输入 1 或 2" get c
11:pict "9"
12:read
13:*无条件打印时 cond="recno()>0"
14:cond="recno()>0"
15:if c="1"
16:@ 5,10 say "1.自己输入条件 2.使用已固化的条件" get c1 pict
17:"9"
18:read
19:if c1="2"
20:sele 5
21:disp all
22:input "请输入您选中条件的记录号" to rno
23:go rno
24:cond=条件
25:else
26:cond=space(50)
27:@ 6,10 say "请输入打印条件" get cond
28:read
29:endif
30:cond=trim(cond)
31:endif
32:wait "以上各项输入需修改吗 Y/N?" to m
33:enddo
34:return
35:procedure prli
36:*生成打印库 c:prdb2.prg 和表线库 c:line4.dbf
37:*原库与打印库建立关联
38:set safe off
39:clear
40:sele 1
41:accept "请输入需打印的数据库名" to na
42:use &na
43:@ 4,15 say "正在生成打印库和表线库,请耐心等待"
44:copy to c:stru3 structure exte
45:use
46:sele 2
47:use c:stru3
48:repl all FIELD__ TYPE with "C"
49:repl all FIELD__ LEN with len(trim(FIELD__ NAME))for
50:FIELD__ LEN<len(trim(FIELD__ NAME))
51:repl all FIELD__ LEN with FIELD__ LEN+1 for
52:int(FIELD__ LEN/2)<>FIELD__ LEN/2
53:creat c:prdb2 from c:stru3
54:append from &na
55:sele 3
56:use c:stru3
57:copy to c:stru6
58:sele 4
59:use c:stru6
60:repl all FIELD__ LEN with FIELD__ LEN+2
61:creat c:line4 from c:stru6
62:append blank
63:append blank
64:append blank
65:sele 6
66:use c:stru6
67:index on FIELD__ LEN to c:st6
68:go bott
69:*生成两条最长的粗、细实线 s1,s2
70:k=(FIELD__ LEN-2)/2
71:store "" to s1,s2
72:i=1
73:do while i<=k
74:s1=s1+"—"
75:s2=s2+"—"
76:i=i+1
77:enddo
78:set index to
```

(下转第 15 页)

也谈 CdBASE Ⅲ 的纵向送数方式

王传信

“电子与电脑”90 年 11 期所载“dBASE Ⅲ 中常规送数方式的改造”一文(以下简称“改”文),提出了一个纵向送数的课题。这个课题提得好,带普遍性。只可惜,那段程序还有毛病,故在 PC 机上无法运行。下面就三个方面的看法,共同探讨一下:

一、“改”文程序的毛病

该程序有两种错误(语法错误与逻辑错误),故在 PC 机上无法运行。此外,还有一些不足之处,这当然是不影响运行的。这里分九点,简要地谈一下。

1. 前三个 STORE 语句的毛病

(1)变量 CONDITION 是字符型的,故 T 就要写成“T”才行。若用逻辑型的,则 T 要写成 .T. 的。

(2)FIELDNAME、A1 等九个字符型变量,用空串‘’(或一个字符‘ ’)来定义,语法上是对的,但不实用。因为 GET 句遇到空串,就一闪而过,不让操作者送数。即使是一个字符的串,我们也无法送入多字符的数据。

2. DO WHILE . CONDITION. 与@1,10 SAY ‘继续吗(Y/N)?’ GET CONDITON 不配套。其中 Y 与 T 不一致,故程序总是中断停运。这样,每输一个学科成绩,就得重新 DO…一下。

3. DATABASE1. DBF 文件名的长度>8 个字符,故在命名上有毛病,但还可在机上凑合用。

4. REPLACE FIELDNAME WITH A&P 中的 FIELDNAME 不是字段名,故要加 & 宏代换符号。

5. USE DATABASE1 的位置不当,故每输入一个学科名,就要重新开一次库。此外,在程序结束前,没有 USE 的关库语句,不安全。

6. 屏幕显示,新老数据混杂在一起,不清晰,看上去非常伤神。

7. DATABASE1 库的库结构,最好事先交待一下,以利阅读。

8. 库内各学科成绩,宜用数字型字段。若用字符型而又要让数字排齐的话,那未送数的要求就高,白白浪费时间。

9. 用功能键直接输入学科字段名,做得对。但要注意,F1 是不准再定义的。

PROG1. PRG 是修改后的程序,可在 PC 机上运行。

二、关于充分利用 dBASE Ⅲ 本身的功能问题

谈到横向与纵向送数,dBASE Ⅲ 本身就有此功能,只是人们很少注意。BROWSE 就是既可用直接命令方式又可用程序方式进行全屏幕的横向与纵向输入的命令语句。BROWSE 纵向送数的操作方法如下:

1. 打开 DATABASE 库,键入 BROWSE 命令。屏幕横向显示全部库字段。纵向显示记录。

2. 按 CTRL－HOME 键,屏幕顶行就显示下列五种功能的提示(光标停在 BOTTOM 处):

BOTTOM TOP LOCK RECORD# FREEZE

3. 把光标移到 FREEZE 处,回车,屏幕顶行就提问 FREEZE 的字段名。

4. 键入学科字段名(可用已定义的 F2－F10 功能键),回车,屏幕光标就停在该字段的第一个记录处。

5. 现在就可以纵向送数了。光标只会上下移动而不横走。待输送完毕后(若有必要,还可增加新的记录),按 CTRL－W 键,然后关库。

假如学科太多,BROWSE 后,要送数的学科却躲在屏幕右侧的外端时,则可以用 LOCK 功能把学生姓名字段锁住,然后用 CTRL－→把该学科字段移入屏内,再进行 FREEZE 等操作。

PROG2. PRG,就是用 BROWSE 操作的程序(比上述直接命令操作法,还要简单)。

三、关于纵向间接方式送数程序的编写

“改”文程序的特点是采用了间接方式送数(用 A1－A8 内存变量间接送数,一次 8 个),比直接开库送数要好一点,但屏幕的利用率太低。另外,当第 8 个数据输错时,有可能无法当场纠正。PROG3. PRG 采用了逐个显示,逐个送数,逐个检查,逐个置换,每屏可容纳 7×5 即 35 个学生的成绩的间接送数方式。同时,程序还避免了新送数据混杂的毛病,屏幕显示清晰。估计它是可以满足运行的要求的。

上述三个方面的看法,不一定全对,仅供商讨之用。

附上两种数据库的库结构与三种程序的清单。

数据库结构　: A:DATABASE. dbf
记录个数　　: 18
最后更新日期: 01/14/91

字段	字段名	类型	宽度	小数
1	学生姓名	Character	8	
2	S1	Numeric	6	2
3	S2	Numeric	6	2
4	S3	Numeric	6	2
5	S4	Numeric	6	2
6	S5	Numeric	6	2
7	S6	Numeric	6	2
8	S7	Numeric	6	2
9	S8	Numeric	6	2
10	S9	Numeric	6	2
** 总计 **			63	

数据库结构　: A:DATABAS1. dbf
记录个数　　: 18

最后更新日期：01/14/91

字段	字段名	类型	宽度	小数
1	学生姓名	Character	8	
2	S1	Character	8	
3	S2	Character	8	
4	S3	Character	8	
5	S4	Character	8	
6	S5	Character	8	
7	S6	Character	8	
8	S7	Character	8	
9	S8	Character	8	
10	S9	Character	8	
＊＊总计＊＊			81	

```
＊PROG1. PRG＊
SET TALK OFF
STOR . T. TO CONDITION
STOR '          ' TO FIELDNAME
USE DATABASE
DO WHIL CONDITION
CLEA
@ 1,10 SAY '请输入学科名：' GET FIELDNAME
READ
GO TOP
do whil . NOT. EOF()
@ 2,0 CLEA
STOR '          ' TO A1,A2,A3,A4,A5,A6,A7,A8
X=0
DO WHIL X<8 . AND. (. NOT. EOF())
X=X+1
P=STR(X,1)
@ X+1,17 SAY 学生姓名 GET A&P
SKIP
ENDD
READ
SKIP -1 * X
X=0
DO WHIL X<8 . AND. (. NOT. EOF())
X=X+1
P=STR(X,1)
REPL &FIELDNAME WITH VAL(A&P)
SKIP
ENDD
endd
@ 1,10 SAY '继续吗(. T. /. F. )?' GET CONDITION
READ
ENDD . T.
USE
RETU

＊PROG2. PRG＊
SET TALK OFF
USE DATABASE
BROW
USE
RETU

＊PROG3. PRG＊
SET TALK OFF
DY= 'Y'
SUBJNAME=SPAC(8)
USE DATABASE
DO WHIL UPPE(DY)= 'Y'
CLEA
@1,30 SAY '请输入学科名：' GET SUBJNAME
READ
CLEA
COL=0
do whil COL<65
@1,COL SAY '学生姓名'+subjname
COL=COL+16
endd
COL=0
ROW=2
GO TOP
do whil . NOT. EOF()
DX= 'N'
A=SPAC(6)
DO WHIL UPPE(DX)# 'Y'
@ ROW,COL SAY 学生姓名 GET A
@9,COL SAY '对吗(Y/N)?' GET DX
READ
ENDD
REPL &SUBJNAME WITH VAL(A)
ROW=ROW+1
IF ROW>8
ROW=2
@9,0 CLEA
if COL<64
COL=COL+16
else
COL=0
@2,0 CLEA
endi
ENDI
SKIP
endd
CLEA
? CHR(7)
@5,30 SAY '继续吗(Y/N)?' GET DY
READ
ENDD
CLEA
USE
RETU
```

PCTOOLS 的使用技巧

刘建明

PCTOOLS 是 PC 机最为流行的磁盘工具软件之一，它为广大用户提供诸多的 DOS 功能。若能熟练地使用 PCTOOLS 无疑会给您的工作带来莫大的方便，本文是笔者的一些体会。

1. POTOOLS 常驻内存

PCTOOLS 为用户提供了常驻内存的功能，使用方法是：键入

PCTOOLS /R128 (Enter)

此时屏幕提示用户 PCTOOLS 已常驻内存等信息。此后，用户可以随时启动 PCTOOLS，只要键入(Ctrl)＋(Esc)即可，即先按(Ctrl)不放，再按一下(Esc)键。其中的/R128 选项说明了 PCTOOLS 常驻内存时将占用 128K 的空间。另外 PCTOOLS 还有两个选项/BW 和/Fn。/BW 选项说明 PCTOOLS 将在黑白显示状态下工作，/Fn 选项中的 n 为 1－9 的数字，它定义了一个功能键来代替上述的(Esc)键，如：键入

PCTOOLS/R128/F4/BW (Enter)

后，用户可以随时按下(Ctrl)＋(F4)键启动黑白显示的 PCTOOLS。

2. 两个窗口

为了便于下面的叙述，首先介绍一下 PCTOOLS 的两个窗口，它们是针对文件操作的 File Function 窗口和针对磁盘操作及特殊操作的 Disk and Special Function 窗口。除了早期的版本，绝大多数 PCTOOLS 都是基于这两个窗口进行操作的。在窗口的下方，有一个命令提示区，用户只要键入命令中的大写字母，就可以执行该命令。两个窗口可以用(F3)键方便地切换。在文件操作窗口中，用户可以用上移键、下移键和(Enter)键选择需要的文件，文件被选择后由 PCTOOLS 给出入选序号，并由特殊色显示其文件名和入选序号。

3. 修改文件的属性

文件的属性有只读(Read Only)、隐含(Hidden)、系统文件(System)、文档文件(Archive)等。如果文件属性被修改为只读，则用 DEL 命令不能将其删除。如果文件属性被修改为隐含，则用 DIR 命令无法列出文件名。虽然许多 DOS 提供了修改文件属性的外部命令，但用 PCTOOLS 修改文件属性更方便。方法是进入文件操作窗口，用户选择好需要修改的文件后，键入“A”，执行“Attrib”命令，屏幕给出相应的窗口提示，根据提示，用上移，下移和(Enter)键将相应属性的开关由 OFF 改为 ON，最后键入“U”将其写入磁盘。一个文件修改好后，可用左移和右移键继续修改其它入选文件的属性。

4. 隐含子目录

许多 DOS 都为用户提供了修改文件属性的功能，却没有提供修改子目录属性的功能。而用 PCTOOLS，我们可以在任何一级子目录上完成这项工作。首先，记住需要修改属性的子目录名，然后进入磁盘和特殊操作窗口，键入“F”，执行“Find”命令，PCTOOLS 提示您输入要查找的字符，此时，键入子目录名，然后回车，则 PCTOOLS 帮您查找该子目录名在磁盘上的位置。查到以后，键入“E”，进入编辑状态，光标停留在目录名的第 0 个字节上，移动光标至同一行的第 11 个字节上，将其值修改为 17，再按下(F5)，就将其写入磁盘。最后，按(Esc)键，退回磁盘和特殊操作窗口，修改就完成了。

由于 PCTOOLS 是在整个磁盘上查找子目录名，因此子目录名不能和磁盘上的其它文件及子目录同名，否者将会引起意想不到的错误。

5. 进入加密的子目录

子目录名中含有不可显示的 ASCII 码或半个汉字时，或者子目录被隐含时，其他人想进入该子目录是很困难的。用 PCTOOLS 就可以不受这些限制。每当按下功能键(F10)时，PCTOOLS 等待用户输入一个驱动器号并回车以后，PCTOOLS 就开始查询磁盘，一但发现磁盘上存在子目录，就显示一个包括各个子目录的树形表，用户可以方便地移动光标进入该子目录，而不需要键入该子目录名。进入子目录后，就可以对其中的文件进行 COPY 等操作。

6. 文件目录的排序

在大型软件的开发过程中，往往要对大批量的文件进行存储、删除、修改。文件在目录中的位置不固定，令人眼花缭乱，甚至找不到文件。PCTOOLS 能将文件目录按照某种规律排序，让同一类文件排在一起，这样就不会显得零乱了。

进入 PCTOOLS 的 File Function 窗口，键入“S”，执行“Sort”命令，PCTOOLS 提示：(F7)按文件名排序，(F8)按扩展名排序，(F9)按文件所占空间大小排序，(F10)按文件生成时间排序，“A”正向排序，“D”倒排序，“U”排序结果存盘等信息。按照提示，您可以根据需要排序。

7. 恢复被删除的文件

如果因误操作而删除了没有备份的文件，将是非常遗憾的。用 PCTOOLS 可以快速、方便地恢复这些被删除的文件。进入 PCTOOLS 的磁盘和特殊操作窗口，键入“U”，执行“Undelete system”命令，屏幕提示用户选择一个子目录。用户选择子目录后，屏幕上列出该目录中所有被删除文件的文件名。从屏幕上可以看出这些文件名的第一个字符已被删除标志所覆盖，用户移动光标选择好需要恢复的文件。键入“G”，PCTOOLS 开始

恢复文件。首先，PCTOOLS 提示用户恢复文件名的第一个字符，再按下(F1)键，由 PCTOOLS 自动恢复文件的簇等信息。照此，依次恢复其他被删除的文件。

除此之外，PCTOOLS 还有许多功能和用法，而且随着 PCTOOLS 版本更新，功能不断增强，完善。由此可见，熟练地使用 PCTOOLS 对使用 PC 机的专业人员和非专业人员都是非常有益的，本文仅仅是抛砖引玉，希望大家都来讨论 PCTOOLS 的使用。

程序常驻内存(TSR)的实现方法

北京工业大学计算机科学系 8771 班　李华

常驻内存(Terminate and Stay Resident)，以下简称 TSR，意即程序虽然结束，但其内容还保留在 RAM 中。当重新按下触发键时，即可马上启动程序，无需从磁盘中再把程序装到 RAM。Borland 公司的软件产品 SK(Sidekick)，就是一个极为有名的 TSR 实用程序。本文是以 Turbo C 为背景，简单介绍如何实现 TSR 的设计。

大部分 TSR 都透过某一个中断启动。常见的中断有时钟、键盘及打印屏幕(prtsc)。Intel 8086 系列微处理器可提供 256 个向量驱动中断，中断服务程序 ISR 从中断向量表所指定的地址开始执行。向量表中的 256 个条目，每个四字节长，其中的地址指向处理特定中断类型的中断程序。向量表从 0000:0000 开始，中断 0 的服务程序地址在 0000:0000～0000:0003，中断 1 在 0000:0004～0000:0007，…依次类推。

TSR 程序设计包括两部分。第一部分用来启动，然后透过 TSR 的 DOS 功能呼叫跳出。此过程必须包括把 TSR 程序进入点的地址，置于向量表的正确位置内。第二部分是实际应用的浮现。在大多数软件产品中，这一部分采用视窗技术来进行操作应用。

TSR 启动部分：这里我们采用中断 9，即按键中断来启动。第一步把目前向量表中断 9 的地址移往 DOS 所不用的一个中断，在此我们使用中断 60，第二步把 TSR 的进入点地址置入向量表中断 9 的位置。当启动时，首先透过中断呼叫 TSR，然后检查读入字符是否为热键。如果是则执行 TSR 应用部分，否则关闭 TSR。使用按键中断的优点为(1)没有损失已有的功能。(2)允许多个 TSR 应用同时存在，由不同的热键启动。

在给出正式的程序前，我们要了解 BIOS 输入按键的方法。标准 DOS 设有 15 个字符长的缓冲区，以容纳从键盘敲入的字符。这种方式许可“打在前头”(type-ahead)等待处理。每当按下一键时，即产生中断 9。按键输入的中断服务程序 ISR 从键盘读入字符并置于缓冲区内，当呼叫 BIOS 的键盘字符输入功能时，只需查看缓冲区的内容。按键缓冲区位于 0000:041E(十进制 1054)；每一个按键产生 16 位的扫描码，15 个字符加上在末尾被自动加上的回车键扫描码，共 32 字节。

缓冲区如同一个环形队列，由一个首指针与一个尾指针控制其读写。首指针指向最近一次敲入的字符，尾指针指向将为 DOS 或 BIOS 的请求而送回的下一个字符。首指针存于 0000:041A(十进制 1050)位置。尾指针存于 0000:041C(十进制 1052)。当队列为空时，首尾指针之值相等。了解了这些内容，对于下面应用部分中判断是否为热键的程序就容易看明白了。

启动部分程序如下。此程序以 F2,F3 为热键，F2 启动 notepad(记事卡)，F3 启动视窗 window，在此对这两个过程的完整程序就不写出了，有兴趣的同学可以自己添加完全。

set__vid__mem()为屏幕模式函数，使公用指针 vid__mem 指向屏幕 RAM 的地址，以便视窗程序块中读写的应用。

tsr()过程调用中断 21H 使程序留驻内存。参数 size 用来告诉 DOS 保留多少空间给 TSR 程序，其单位为段，每一段为 16 字节。一般把程序的.EXE 文件大小除以 16，再乘以 2，作为 size 的大小将是安全的。

main()执行后，程序仍留驻 RAM，可“随时待命”。

```
/* Terminate and stay resident using
    keyboard interrupt */

#include "dos.h"
#include "stdlib.h"
#include "ctype.h"

void interrupt tsr__ap();

char far *vid__mem;

/* busy is set to 1 when program is active,
    0 otherwise */
char busy=0;

main()
{ struct address
        { char far *p;
        } temp;
    /* pointer to interrupt 9's address */
    struct address far *addr=
        (struct address far *) 36;
    /* pointer to interrupt 60's address */
    struct address far *int9=
        (struct address far *) 240;

    /* Move the keyboard interrupt routine's address
```

```
        to int 60. If int 60 and int 61 contain the same
        address,then the TSR program has not been
        installed * /
    if ( int9->p==(int9+1)->p)
        { int9->p=addr->p;
addr->p=(char far * ) tsr__ap;
printf( "tsr installed -f2 for notepad,");
printf( "f3 for window");
        }
    else { printf(" tsr already initialized\n");
    exit(1);
}

    set__vid__mem();
    tsr(1600);
}
    set__vid__mem()
    { int vmode;

        vmode=video__mode();
        if ((vmode!=2) && (vmode!=3) && (vmode!=
7))
        { printf("video must be in 80 column text mode");
    exit(1);
        }
    /* set proper address of video ram */
    if (vmode==7)
        vid__mem=(char far * ) 0xb0000000;
    else
        vid__mem=(char far * ) 0xb8000000;
}
    /* terminate but keep resident */
    tsr(size)
        unsigned size;
        {union REGS r;
        r.h.ah=49; /* terminate and stay resident */
        r.h.al=0; /* return code */
        r.x.dx=size; /* size of program/16 */
        int86(0x21,&r,&r);
    }

    /* return the current video mode */
    video__mode()
    { union REGS r;
        r.h.ah=15; /* get the video mode */
        return int86(0x10,&r,&r,) & 255;
    }
```

TSR 应用部分：

```
/* this is the entry point into the TSR application code */
void interrupt tsr__ap()
{ char far *t=(char far * ) 1050;/* address of head pointer
*/
    geninterrupt(60);/* read the character */
    if (*t!=*(t+2)) /* if not empty */
        {t+=*t-30+5; /* advance to the character posi-
tion */
        if (*t==60 || *t==61)
    { bioskey(0); /* clear the f2/f3 key */
        if (! busy)
        { busy=! busy;
        application(*t);

        busy=! busy;
        }
    }
        }
}
application(w)
int w;
{ switch(w)
    { case 60: notepad();
break;
    case 61: window();
break;
    }
}
```

interrupt 为 Turbo C 的保留字，为中断函数定义符，允许把 C 函数当作 ISR 使用。函数 geninterrupt() 也是 Turbo C 具有的。它是以要产生的中断编号来呼叫的。中断 60 的功能为仅由标准输入读取按键。(main ()中已将中断 9 的地址放在中断 60 的地址上)。从中断 60 返回后，检查是否首指针所指内容为热键(F2/F3)，二者位置码为 G0 及 G1。若按下二热键，则必须把它读进来，以免被应用读到。公用参数 busy 用来阻止两个应用同时启动。应用过程 application()中的 notepad()与 window()函数，在此就不写出了。

通过以上介绍，读者可对常驻内存程序的实现方法有一个大概了解。但由于 TSR 程序要破坏原系统正常的 DOS，所以使用与设计时，要特别小心。

(上接第 20 页)

功能：清除“后台”为第一页的图形。

② POKE 230,64: CALL—3086:RETURN

功能：清除“后台”为第二页的图形。

③ POKE—16299,0: POKE 230,32:RETURN

功能：第二页做“前台显示”，第一页做“后台绘图”。

④ POKE—16300,0: POKE 230,64:RETURN

功能：第一页做“前台显示”，第二页做“后台绘图”。

防止非法使用 dBASE 程序的一种方法

贵州水城钢铁公司　机修厂　黄世刚

程序员通常在 dBASE Ⅲ 程序中设置口令来防止非法人员启动程序。但非法人员只要查看源程序或者修改口令部分就能运行保密程序。我厂的标准化管理系统采用了一块引导软磁盘(保密盘),使得无此软磁盘的人员无法运行程序。全部程序在 IBM PC/XT 型机上通过。下面将保密方法简单介绍一下。

1. 首先建立一个引导盘,该盘仅供允许人员使用,在引导盘上建立一个引导文件:

文件名 BZH. BAT

```
ECHO OFF
A:
COPY A:BZH. PRG C:BZH. PRG
C:
ECHO ON
DBASE BZH
A:
ECHO OFF
DEL C:BZH. PRG
ECHO ON
```

2. 在该引导盘上建立一个 dBASE Ⅲ 程序的引导文件,这个引导文件也是主程序:

文件名 BZH. PRG

```
SET TALK OFF
CLOSE DATA
CLEA
? "  "
? "机修厂标准化管理系统"
? "1. 标准资料数据库"
? "2. 厂标准体系表"
? "3. 标准化工作文件"
? "4. 标准化管理机构"
? "5. 退出"
WAIT "请选择:" TO AC
DO CASE
CASE AC="1"
DO BZZL
CASE AC="2"
⋮
RETU
ENDCASE
ENDDO
```

3. 每次使用时在 DOS 状态下插入引导盘,打入 BZH↙后 DOS 就将主程序拷贝到 C 盘,并自动调用 C 盘上的 dBASE Ⅲ 及主程序 BZH. PRG,程序运行结束后引导程序自动删除 C 盘上的主程序。这样无引导盘的非法人员就无法运行 C 盘上的 dBASE Ⅲ 程序了。

为了进一步提高程序的保密度,还可以将引导盘上的 BZH. BAT 和 BHZ. PRG 隐藏起来。

也谈 IBM 机图形存取

上海工程技术大学　焦小友

本刊 89 年第 6 期刊登的"IBM 微机图形的另一种存取方法"中,作者采用的方法是将屏幕上图形信息经"GET"存入数组,再打开一个文件,将该数组读入文件。经本人多次试用,发觉有三个缺点。

一、调用文件后,图形并不能直接"PUT"在屏幕上,必须在调用前"GET"划取一块与存图时同样大小的数组。通常程序编制者并不能记住过去某个时候曾用过多大的数组,更不用说在屏幕上精确地划取了。若每次都记录图形文件"GET"了多大尺寸及范围,既烦琐又不可靠。

二、存取速度太慢。笔者作了一个测试,结果如下:(在 PC/XT 机上,用该文的范例程序)

整形数组大小	存入软盘时间
3000	26.929 秒
5000	44.99 秒
8002(全屏)	1 分 17.78 秒

由上表可见,存不到半屏图形将近半分钟,存全屏图形竟达 1 分 17 秒多,这段时间计算机可以从一累加到溢出!

三、文件所占的字节数太大,经测试如下:

整形数组大小	图形文件大小(字节)
3000	9001
5000	15001
8002(全屏)	24007

一幅全屏图形比用常规"BSAVE"存入的要大 8K!

综上所述,这种图形存取方法至少在 BASIC 领域内是一种不宜采用的方法。

下面向 BASIC 用户介绍一种方法,它既避免了上述缺点,又比用"BSAVE"灵活,笔者开发的一个绘图软件《ORIONARTS》均采用此方法存取图形。范例如下:(用 QUICK BASIC)

```
SCREEN 2: LINE (10 ,10)-(100. 100),,B  'Random graphics
DIM P%(16004)  'Define array
GET (0,0)-(639,199),P%  'Get full screen
DEF SEG = VARSEG(P%(0))  'Define segment
```

```
BSAVE "pic",VARPTR(P%(0)),16004   '存全屏
DEF SEG
```

这段程序把全屏存盘，各测试数据如下：

整形数组大小	存盘时间	文件大小
16004(全屏)	8.23 秒	16019 字节

全屏存盘仅用 8 秒，文件所占字节比"BSAVE"存屏(16K)还少。以下是调图的例子：

```
DIM L%(16019)      ;数组大小即为文件大小
DEF SEG = VARSEG(L%(0))
BLOAD "pic",VARPTR(L%(0))
DEF SEG
PUT (0,0),L%
```

调图时，须看一下图形文件的字节数，并据此定义数组的大小。调出后随时可"PUT"在屏幕上，由于是"PUT"到屏幕上的，所以也消除了以往使用常规调图时由于奇偶线原因造成的"二次成像"缺陷。

如果使用的是 BASICA 或 GWBASIC，只须把上述二程序加上行号并删去 DEF SEG = VARSEG(数组(0))"这句即可。

(上接第 8 页)

```
79;go 1
80;do while . not. eof()
81;fn=trim(FIELD_NAME)
82;fw=FIELD_LEN-2
83;sele 4
84;go 1
85;repl &fn with substr(s1,1,fw)+"+"
86;go 2
87;repl &fn with substr(s1,1,fw)+"+"
88;go 3
89;repl &fn with substr(s1,1,fw)+"+"
90;sele 6
91;skip
92;enddo
93;use
94;sele 1
95;use &na
96;set relation to recno() into b
97;set safe on
98;return

. use c;stru3
. list to print
```

记录号	FIELD_NAME	FIELD_TYPE	FIELD_LEN	FIELD_DEC
1	分类号	C	20	0
2	书名	C	34	0
3	作译者	C	10	0
4	出版社	C	12	0
5	出版时间	C	8	0
6	版次	C	4	0
7	检索号	C	6	0
8	总册数	C	6	0
9	现存数	C	6	0
10	页数	C	6	0
11	单价	C	6	2

图书库全部记录

第 2 页　　1990.12.5

分类号	书名	作译者
73.87053 2589	计算机应用教学大纲	中华教育部
73.87054 2308	电子计算机讲义	王玉龙
73.87055 2819	计算机通考试题	日本研究所
73.87051 2650	电子计算机实验	白中英

北京安华科技市场西城经营部

——主要经营：电子计算机、电子元器件

各种微机：

• 中华学习机：　CEC－1　￥900.00　小蜜蜂－I　￥920.00

• 家庭 PC 机：￥3150.00

主频：4.77/12 兆　RAM：640KB　软驱：360KB×2，101 键盘

14"双频单显

Smper286：￥6950.00

主频：10/21 兆　RAM：1MB 可扩 5MB　5¼软驱：1.2MB. 360KB 各 1

硬盘 3.5"40MB.　101 键盘　14"双频单显　数显机箱

选配：14"CGA 彩显加 1500 元　14"1024×768　VGA 彩显加 3800 元

各种：APPLE Ⅱ、CEC－Ⅰ、PC 机接口板；九针、24 针打印机

还承接各种微机的修理业务

对教育部门、个人购机实行优惠

地址：北京西四北大街 120 号　电话：65.6342　邮码：100034

联系人：孙义强　宋函华　BB 机：126 呼 45198　42616

侦错核对程序

西安供电局　张永辉

在港刊杂志上常见有程序中附有侦错核对码，对用户的键入程序是很有帮助的，其形式是每一程序行或监控下每显示一行数据都显示一个核对码，如键入此程序，比较一下核对码便可得知有无错处。笔者试着编写了一个较为简单的程序 EXBM(见程序二)，对监控还有所加强。

EXBM 运行后程序放在辅存，CPU 的控制权也在辅存。为区分是否在 EXBM 控制下，BASIC 的提示符改为“>”，监控的提示符为“?”。如果要退出 EXBM 状态，可按 CTRL—RESET；在辅存没被破坏的前提下键入 POKE49283,0 或监控状态下键入 C083 可再行进入。有两点需要说明：一是因 CEC 汉字系统自动置主存方式，禁止辅存，所以退出 EXBM；二是起动 EXBM 前系统内不能有使用辅存的程序。

在 BASIC 下，输入程序的常见错误有多字、少字、字符输入错误，其它还有夹带看不见的控制字符、行号错误等。当出现字符输入错误时，计算出的核对码和正常的不同，但不影响下一行的核对码，如出现多字少字现象，不但影响该行的核对码，还使下一行的核对码出现错误。当误键入了控制字符时，除光标控制符外均发出响声，如同按了 CTRL-G。LIST 时同样发出声响。使用者应掌握这些规律，以便迅速查出错误所在。

使用 EXBM 时先键入一行程序，回车后将显示一个核对码，与正确的核对码比较，无误后可继续键入，否则改正错误。为方便查看，LIST 后程序左起上下对齐，空出核对码和行号。核对码是两位十六进制数，在每个程序行的行首，用一个底横线分隔行号(见程序一)。BASIC 语句的显示方式和通常不一样，在 EXBM 状态下如果保留字在行尾，有可能保留字的后几个字符显示在下一行，而通常是不能分开显示的。EXBM 运行后一些 BASIC 语句将被禁止使用，特别是 RUN、GOTO、CONT 等命令的使用有可能破坏系统，最好不要使用这些命令。EXBM 只是为了侦错除错。DOS 的一般读写命令是允许执行的。

显示“?”符是进入监控的标志，在 EXBM 状态下可使用小汇编，对监控也无不良影响，控制符的处理同 BASIC 状态。在使用数据(块)显示指令时，每八个(或八个以下，视起始地址而言)数据有一个核对码，在数据输入时最好也采用显示时的统一地址，以便得到正确的核对码。因核对码的计算和数据的地址有关系，如果把数据输入到其它地址，即便是数据无误也得不到正确的核对码。数据输入完一行并回车后，将显示一个核对码，以供比较。在显示核对码的同时还将显示下一个要输入数据的地址，方便于数据块的输入。另外在数据块的显示中除按 CTRL-S 可暂停外，还可用空格键暂停，按 EXC 退出或按任一键继续显示。

EXBM 占用零页的 $19——1F 单元，此时用户不得使用这些单元。

一行程序或数据按其不同排列关系、不同大小而得出的核对码是大大超过 256 种状态的，就是说一个核对码可为多个程序行得出，这也是 EXBM 的不足之处。它是针对常见的、一行中只有少数错误，它的盲区是一行中出现多处错误，并且这些错误对核对码的影响相互抵消为零。不过这是极个别的情况。

程序一

```
19__ 10 HOME
00-20 HTAB (10):VTAB (7):PRINT "EXBM1.0 SYSTEM"
AC__ 30 END
```

```
6000- A2 00 8D 81 C0 8D 81 C0          $45
6008- BD 00 D0 9D 00 D0 E8 D0          $A5
6010- F7 EE 0A 60 EE 0D 60 D0          $04
6018- EF 8D 83 C0 8D 83 C0 A9          $8F
6020- BE 8D 40 D4 A9 13 8D 0D          $2A
6028- D5 A9 D9 8D 0E D5 A9 5C          $85
6030- 8D FA D6 A9 D9 8D FB D6          $5F
6038- A2 FF E8 BD 00 61 9D 12          $67
6040- D9 E0 F5 F0 0C BD A3 60          $B1
6048- E0 5D B0 EE 9D 66 D7 90          $0D
6050- E9 A9 66 8D 03 D7 8D 53          $A2
6058- D7 8D 5B D7 8D 60 D7 A9          $BC
6060- D7 8D 54 D7 8D 04 D7 8D          $60
6068- 5C D7 8D 61 D7 A9 51 8D          $B5
6070- 08 D7 A9 60 8D DB FE 2E          $61
6078- DB FB A9 20 8D 69 FF A9          $82
6080- 78 8D 8C FD A9 D9 8D 8D          $94
6088- FD A9 EA 8D 6C FF A9 80          $30
6090- 8D 6A FF A9 D9 8D 6B FF          $64
6098- A9 BE 8D BE FD A9 D9 8D          $B0
60A0- BF FD 60 A6 24 D0 0B C6          $FE
60A8- 25 48 20 8E FD 68 A6 1A          $15
60B0- 86 24 E4 1A D0 04 C9 20          $ED
60B8- F0 03 20 5C DB 60 AD 00          $4E
60C0- C0 C9 A0 D0 18 2C 10 C0          $62
60C8- AD 00 C0 10 FB C9 9B D0          $86
60D0- 09 A9 87 20 ED FD 68 68          $22
60D8- 68 68 8D 10 C0 60 B1 1C          $8A
60E0- 29 0F A8 B9 F5 D9 48 65          $82
60E8- 1C 29 0F A8 68 18 79 F5          $75
60F0- D9 A0 00 18 71 1C 45 1C          $4F
60F8- 45 1D 18 65 1F 85 1F 60          $8B
6100- 60 20 1C D9 20 4B D9 4C          $C4
6108- 3C D4 A5 9B 85 1C A5 9C          $9A
```

```
6110- 85 1D 20 3D D9 20 3D D9    $58
6118- A0 00 84 1F 20 39 D9 20    $FE
6120- 39 D9 B1 1C D0 F9 60 18    $33
6128- 20 A1 D7 E6 1C D0 02 E6    $26
6130- 1D 60 20 9C FC A9 1D 85    $2C
6138- 24 20 48 F9 A9 A4 20 5C    $79
6140- DB A5 1F 20 DA FD 20 9C    $F6
6148- FC 60 48 8A 48 20 1C D9    $21
6150- A5 1F 20 DA FD A9 DF 20    $F6
6158- 5C DB 68 AA 68 20 24 ED    $52
6160- A4 24 C8 84 1A 60 A5 24    $8C
6168- 85 1E 20 9C FC 60 A5 31    $A5
6170- C9 BA D0 33 A5 3C 85 1C    $4A
6178- A5 3D 85 1D A9 A0 20 ED    $C4
6180- FD A5 41 20 DA FD A5 40    $18
6188- 20 DA FD A9 BA 20 ED FD    $14
6190- 20 20 FC A5 1E 85 24 A0    $BB
6198- 00 84 1F 20 39 D9 A5 1C    $9E
61A0- C5 40 D0 F7 20 44 D9 A9    $1C
61A8- BF 85 33 60 20 DA FD A5    $D4
61B0- 3C 85 1C C5 40 D0 08 A5    $E2
61B8- 3D C5 41 D0 02 84 1F A5    $E0
61C0- 3D 85 1D 20 39 D9 A5 3C    $62
61C8- C5 3E D0 06 A5 3D C5 3F    $26
61D0- F0 06 A5 1C 29 07 D0 0A    $3A
61D8- 20 44 D9 A0 00 84 1F 20    $C1
61E0- 81 D7 60 1A F9 48 C6 14    $52
61E8- 99 5A CC 09 EE 71 39 B9    $5B
61F0- 30 2E 96 B1 2E B0 60       $DF
```

让 CEC—BASIC 也有自动行号功能

云南地矿局物质供应处　沈东山

中华学习机的 BASIC 没有自动行号的功能。《苹果机工具箱》一书中介绍了一个自动行号程序 AUTO LINE。该程序从 100 开始产生行号。如在 BASIC 程序区中已有一个 BASIC 程序，并且有大于 100 的行号时，就必须用 POKE 命令在指定内存单元预置行号，否则就有可能使内存中原有的 BASIC 程序被破坏，使用起来不太方便。

本人经过对 AUTO LINE 进行分析，设计了一个 AUTO 程序，该程序占用 $9000—$9161 内存单元，具有自动寻找程序尾行，并将尾行号取整为整 10 后给出行号，解决了 AUTO LINE 需人工预置行号的不便之处，在内存中没有 BASIC 程序时从 10 给出行号。

使用方法：在监控下将程序输入内存，然后用 BSAVE AUTO,A $9000,L $161 命令存盘备用。使用时只需 BRUN AUTO 命令便可实现自动行号功能。按 F5 键或 Ctrl—F 键便可退出自动行号功能。再次进入时可用 CALL 36864 调用。在运行中如果不想要正在输入的当前行，可按 Ctrl—X 键即可清除当前行，重新进行输入。如果将 $90F4 单元的内容改为 $0B，可在行号后自动生成 DATA 语句。这样可以很方便地在程序尾部加入程序所需要的数据。

```
9000- A9 90 85 74 A9 00 8D 60
9008- 91 8D 61 91 A5 67 85 69
9010- A5 68 85 6A A0 00 B1 69
9018- D0 05 C8 B1 69 F0 1C A0
9020- 02 B1 69 8D 60 91 C8 B1
9028- 69 8D 61 91 A0 00 B1 69
9030- AA C8 B1 69 86 69 85 6A
9038- 4C 14 90 20 3F 90 60 A9
9040- 00 8D 4E 91 8D 4F 91 8D
9048- 50 91 8D 51 91 8D 52 91
9050- A9 F0 8D 53 91 A9 14 8D
9058- 54 91 A9 11 8D 55 91 A9
9060- 24 8D 56 91 A9 11 8D 57
9068- 91 A9 F0 8D 58 91 AC 61
9070- 91 D0 15 4C 79 90 88 D0
9078- 0F AD 60 91 F0 35 C9 0A
9080- B0 1E D0 2F 4C E0 90 60
9088- A2 04 A9 06 20 C7 90 A9
9090- 05 A2 03 20 C7 90 A9 02
9098- A2 02 20 C7 90 4C 76 90
90A0- A2 03 A9 01 20 C7 90 AD
90A8- 60 91 38 E9 0A 8D 60 91
90B0- 4C 79 90 A2 04 20 C7 90
90B8- A9 00 8D 52 91 A2 03 A9
90C0- 01 20 C7 90 4C 84 90 18
90C8- 7D 4E 91 9D 4E 91 C9 0A
90D0- 10 01 60 E9 0A 8D 4E 91
90D8- 18 CA A9 01 4C C8 90 00
90E0- 68 AA 68 48 A9 F1 85 38
90E8- A9 90 85 39 20 EA 03 68
90F0- 60 48 8A C9 06 10 0A 68
90F8- 91 28 BD 4E 91 18 69 B0
9100- 60 68 2C 00 C0 10 FB 91
9108- 28 AD 00 C0 2C 10 C0 48
9110- C9 86 D0 11 68 A9 98 48
9118- A9 1B 85 38 A9 FD 85 39
9120- 20 EA 03 68 60 C9 8D F0
9128- 02 68 60 8A 48 A2 03 FE
9130- 4E 91 BD 4E 91 C9 0A 30
9138- 08 A9 00 9D 4E 91 CA 10
9140- EE 90 05 A9 01 9D 4E 91
9148- 68 AA 68 60
```

图形屏幕坐标的对应存储单元

葛海东

在APPLE Ⅱ和CEC—Ⅰ的高分辨率图形模式下，直接用机器语言中的送存指令作图，具有速度快的优点。但是图形点地址与我们所熟悉的直角坐标并不一致。本文提供一个能把直角坐标转换为图形点地址的程序。

执行前，要把纵坐标放入$1E，把横坐标放入$1B，若横坐标超过了$FF(255)那就要置$1A为$01，反之置为$00。在图形第一页时置$303为$08，第二页时则置为$10。程序执行完后，$FE、$FF中已按低在前高在后的顺序放入了图形点所在单元的地址。这样便于用间接变址指令如：STA ($FE),Y等来存取。$CF单元的数表示此点在存储单元的七个连续点中的位置。例如：若$CF中的值为$03，那么它就位于七个点中的d_3位(最小位元为d_0)，右移或循环右移三次就可用来改变此点的颜色，从而达到画点的目的。

程序中使用了$FF、$FE、$CF、$E8、$1B、$1E、$1A七个零页单元，并改变了寄存器A的内容，其它均不受影响。由于采用相对寻址，因而可以把它放到内存的任何位置执行。

```
300- 84 18 A9 10 85 FF A5 1E
308- A0 A0 0A B0 08 A0 50 0A
310- B0 04 A0 00 4A 0A 0A 84
318- FE 26 FF 0A 26 FF 0A 66
320- FE 4A 4A 4A 65 FF 85 FF
328- A4 1A A5 1B 38 85 CF E9
330- 07 E6 FE B0 F8 88 F0 F4
338- C6 FE A4 18 60 00 00 00
```

神奇的$C0B0单元

福建　南平师范学校　李锋

CEC—Ⅰ的新用户时常对字库的自检感到疑惑不解，机子怎么一而再，再而三地出现同一批地址$4000～$7FFF。原来，汉字的字库容量为256K，由于CEC—Ⅰ的中央处理器6502仅有64K的寻址能力，因此在汉字处理电路中，采取了存储器切换技术。存储体的切换，是通过对$C0B0单元置值来实现。单元$C0B0中各位的功能情况如下：

$D_7D_6D_5D_4D_3D_2D_1D_0$	功　能
××××0000	选择空间$4000～$7FFF第1体
××××0001	选择空间$4000～$7FFF第2体
××××0010	选择空间$4000～$7FFF第3体
××××0011	选择空间$4000～$7FFF第4体
××××0100	选择空间$4000～$7FFF第5体
××××0101	选择空间$4000～$7FFF第6体
××××0110	选择空间$4000～$7FFF第7体
××××0111	选择空间$4000～$7FFF第8体
××××1000	选择空间$4000～$7FFF第9体
××××1001	选择空间$4000～$7FFF第10体
××××1010	选择空间$4000～$7FFF第11体
××××1011	选择空间$4000～$7FFF第12体
××××1100	选择空间$4000～$7FFF第13体
××××1101	选择空间$4000～$7FFF第14体
××××1110	选择空间$4000～$7FFF第15体
××××1111	选择空间$4000～$7FFF第16体
×××0××××	禁止彩色显示
×××1××××	允许彩色显示
××0××××××	选择空间$8000～$BFFF第1体
××1××××××	选择空间$8000～$BFFF第2体

地址空间$4000～$7FFF的每一体的容量为16K字节，汉字库共16体，因此汉字库的容量为16×16＝256K。自检时，是一体一体地自检，故相应地址重复16次，约需一分钟！

怎样解读CEC—Ⅰ学习机的汉字系统

刘文勤

CEC—Ⅰ型中华学习机有较完善的汉字处理能力，许多拥有中华学习机的爱好者在二次开发应用中都希望详细了解该机汉字系统对汉字和ASCII字符的具体处理过程。但由于汉字系统固化在辅存中，要想研究它也要费点脑筋。笔者为此编了一个小程序，可以将辅存中D000～FFFF之间的12K字节汉字处理程序传送到内存1000～3FFF区间，然后返回监控系统。现在您可以对汉字系统的任意段落进行反汇编，仔细分析研究了，不过要记住，这段程序现在在内存中的高位地址，要加上$C0才是该机汉字系统的真实地址。

```
0800- A9 00 AA A8 85 06 85 08
0808- A9 D0 85 07 A9 10 85 09
0810- 8D 80 C0 8D 09 C0 B1 06
0818- 91 08 C8 D0 F9 E6 07 E6
0820- 09 E8 E0 30 D0 F0 8D 08
0828- C0 8D 82 C0 4C 69 FF
```

用机器语言解八皇后问题及其推广

广西柳州　龙城中学　卢祖兴　刘超宏

八皇后问题是指：在8×8的方棋盘上放置8个国际象棋的皇后，使之彼此不同行，不同列，不同对角线，以免互相吃掉。这是高斯在1850年首先提出来的数学趣题。由于计算量很大，以致这位著名数学家在有生之年也未能找出问题的全部解答。

自从计算机出现之后，八皇后问题自然也成了计算机爱好者讨论的热门话题。由于通常都是在8位机上用BASIC语言来求解，仍然费时较多。由于算法不同，有的竟长达8、9小时，快的也要近30分钟，若用汇编语言来解，就能极大地节省时间。

我们用6502机器语言编写的程序一，就能在22秒钟给出八皇后问题的92组解和解法布局图。方法是，在监控状态下，准确地输入程序一，然后300 G或在BASIC状态下，CALL 768即可。若事先接通打印机，还可把解法及布局图印在纸上。(0表示空白，*表示皇后)。为推广到N皇后的情形，输入程序二，就可以很方便地设定N值(3<N≤16)，用以解N皇后问题。

当N值较大时，可打印解法，而省略布局图，以节省时间和打印纸。在监控下输入程序一后，运行程序二，输入N值及选择解法(B)或布局图(F)即得出结果。

程序一

```
0300- A9 01 85 E0 A9 00 85 09
0308- 85 0A 85 18 E6 18 A6 18
0310- A9 01 95 E0 A6 18 CA 86
0318- 19 A6 19 B5 E0 85 1A A6
0320- 18 B5 E0 38 E5 1A F0 5D
0328- 10 04 49 FF 69 01 85 1A
0330- 8A 38 E5 19 C5 1A F0 4F
0338- C6 19 10 DD A5 18 C9 07
0340- 90 CA 85 08 E6 09 D0 02
0348- E6 0A A0 08 24 0B D0 18
0350- A6 08 98 D5 E0 08 A9 AA
0358- 28 F0 02 A9 B0 20 ED FD
0360- A9 A0 20 ED FD 88 D0 E8
0368- A6 08 B5 E0 20 E3 FD 24
0370- 0B F0 05 20 48 F9 F0 03
0378- 20 8E FD C6 08 10 CB 20
0380- 8E FD 20 8E FD A6 18 B5
0388- E0 C9 08 B0 04 F6 E0 D0
0390- 83 C6 18 10 F0 60
```

程序二

```
5 POKE 11,0
10 INPUT "N= ";N: POKE 831,N-1: POKE 843,N: POKE 906,N
15 INPUT "Full of Brief? (F/B)"; A$
17 IF A$ = "B" THEN POKE 11,1
18 CALL 768
20 PRINT "TOTAL="; PEEK (10) * 256+ PEEK (9)
30 POKE 831,7:POKE 843,8:POKE 906,8
]RUN
N =8
Full or Brief? (F/B)F
      0  0  0  0  *  0  0  0  4
      0  0  0  0  0  0  *  0  2
      0  *  0  0  0  0  0  0  7
      0  0  0  0  0  *  0  0  3
      0  0  *  0  0  0  0  0  6
      *  0  0  0  0  0  0  0  8
      0  0  0  *  0  0  0  0  5
      0  0  0  0  0  0  0  *  1

      0  0  0  *  0  0  0  0  5
      0  0  0  0  0  0  *  0  2
      0  0  0  0  *  0  0  0  4
      0  *  0  0  0  0  0  0  7
      0  0  0  0  0  *  0  0  3
      *  0  0  0  0  0  0  0  8
      0  0  *  0  0  0  0  0  6
      0  0  0  0  0  0  0  *  1

]RUN
N= 8
Full of Brief? (F/B) B
4  2  7  3  6  8  5  1
5  2  4  7  3  8  6  1
3  5  2  8  6  4  7  1
```

也谈汉字平移显示

四川铜梁县中学　李茂学

贵刊九〇年第八期上介绍汪五一同志的"CAI课件制作辅助工具——文字平移显示"，对汉字的平移作了较清楚的论述，读了深受启发。本人根据该文提供的思想方法，在中华学习机上重新设计了文字平移显示制作程序。本文的BASIC和机器语言程序都大大短于汪文的程序，操作简单，使用方便，在此呈献给大家供参考。下面介绍使用方法：

1. 输入机器语言程序2和程序3，分别以LMX·SET和LMX·PY为文件名存盘。

2. 运行BASIC程序1。输入平移汉字和行数(指刚输入的汉字显示在屏幕上有几行)。

按提示逐次回答后输入存盘(文件名如：PYHZ)

3. 在程序中或键盘上打入如下命令：

BLOAD PYHZ↙

BRUN LMX・PY

就会看到输入的汉字在指定的行上向左平移显示。

本程序最大限度可移动显示 400 个汉字，也能在 APPLE 机上运行。机器语言程序全是浮动地址，可放在内存中的适当地方。

程序一

```
1 REM 汉字平移点阵信息提取
2 POKE 24578,8: POKE 24580,8: POKE 24579,96: POKE 24581,96
5 PRINT CHR$(4);"BLOAD LMX.SET"
10 INPUT "输入平移汉字:";A$:PRINT A$
15 INPUT "有几行?";B: HGR2
20 VTAB 1: PRINT A$
30 FOR I = 1 TO B: VTAB 9: CALL 8192
40 PRINT CHR$(13)
50 NEXT I
60 INPUT "显示在第几页(1/2)?";P: IF P>2 OR P<1 THEN 60
65 INPUT "在第几行平移(1--12)?";N: INPUT "存入文件名";M$
70 A = PEEK(24579)*256+PEEK(24578):B = PEEK(24581)*256+PEEK(24580)
80 L = B-A+1:B = 8192*(1+(P=2))
85 N=N-1:A = B+40*INT(N/4)+256*(N-INT(N/4)*4):B = A+128
90 POKE 24577, INT(A/256): POKE 24576, A-256*PEEK(24577): POKE 24583, INT(B/256): POKE 24582,B-256*PEEK(24583)
100 PRINT CHR$(4);"BASVE";M$;",A$6000,L";L
```

程序 2

```
2000- AD 04 60 85 FE AD 05 60
2008- 85 FF A0 00 A9 00 85 FA
2010- A9 80 85 FC A9 40 85 FB
2018- 85 FD A2 00 B1 FA 84 06
2020- A0 00 91 FE 18 A9 04 65
2028- FB 85 FB E6 FE D0 02 E6
2030- FF A4 06 E8 E0 08 D0 E4
2038- A2 00 B1 FC 84 06 A0 00
2040- 91 FE 18 A9 04 65 FD 85
2048- FD E6 FE D0 02 E6 FF A4
2050- 06 E8 E0 08 D0 E4 C8 C0
2058- 28 D0 B1 A5 FE 8D 04 60
2060- A5 FF 8D 05 60 60
```

程序 3

```
3000- 2C 10 C0 AD 02 60 85 FE
3008- AD 03 60 85 FF 20 38 30
3010- 18 A5 FE 69 10 85 FE A5
3018- FF 69 00 85 FF AD 00 C0
3020- 10 01 60 A5 FF CD 05 60
3028- 30 E3 CD 05 60 D0 D4 A5
3030- FE CD 04 60 10 CD D0 D5
3038- A0 00 B1 FE 48 98 AA 68
3040- 95 50 C8 C0 10 D0 F3 A9
3048- 07 85 06 AD 00 60 85 FA
3050- AD 01 60 85 FB AD 06 60
3058- 85 FC AD 07 60 85 FD A2
3060- 00 B5 50 6A 08 95 50 A0
3068- 27 B1 FA 0A 28 6A 6A 08
3070- 0A 38 6A 91 FA 88 10 F1
3078- 28 18 A9 04 65 FB 85 FB
3080- E8 E0 08 D0 DC A2 00 B5
3088- 58 6A 08 95 58 A0 27 B1
3090- FC 0A 28 6A 6A 08 0A 38
3098- 6A 91 FC 88 10 F1 28 18
30A0- A9 04 65 FD 85 FD E8 E0
30A8- 08 D0 DC C6 06 D0 9C 60
```

“前台显示、后台绘图”的秘密

谢吉华

许多游戏软件运行时，能快速地更换画面且不露出任何绘图过程，令人赞叹不已。初看起来，似乎 CEC—I 机有“前台”、“后台”两部分。那么，“前台显示，后台绘图”的功能是如何实现的呢？

CEC—I 机有两页高分辨图形区，计算机执行 HPLOT 语句时，零页存储单元 $E6 为高分辨率作图的页面指针，当 $E6(十进制 230)的值为 $20(十进制 32)时，HPLOT 语句的执行结果画在高分辨第一页上；$E6 单元的值为 $40(十进制 64)时，HPLOT 的执行结果画在高分辨第二页上。因此，我们在观看高分辨第一页图形时，可用命令 POKE230,64 使高分辨第二页图形区作为“后台”，接着运行的绘图程序就会将图形画在第二页上，这样，“前台显示、后台绘图”的功能就实现了。第二页图形画好后，只要用 POKE—16299,0 就可以观看画好的第二页。这幅在“后台”画好的图形就在“前台”上显示出来了，而刚才为“前台”的第一页只要执行 POKE230,32 后又可转为“后台绘图”了。

细心的读者会发现，当第二页图形作为“前台”，而作为“后台”的第一页需要清除画面时该怎么办呢？如果使用 HGR 命令不但会使第一页转到“前台”而且还会让大家看到清除画面的动作，这样“后台绘图”就徒有虚名了。笔者发现 HGR、HGR2 有一段公用程序可供借用，它的入口是 $F3F2(十进制—3086)，所以要清除“后台”为第一页的图形，可执行如下命令：

POKE 230,32: CALL—3086↙

为了便于大家掌握“前台显示，后台绘图”的技巧，现将几个关键的子程序及功能说明如下：

① POKE 230,32: CALL—3086:RETURN

(下转第 13 页)

演示抛体运动

北方工业大学　白梅兰

应用微机的作图功能模拟物理实验是经济、方便又准确的现代化手段。本文介绍一个有趣的抛体运动问题的微机模拟演示。

1. 问题

设从某一点 O 以同样速率 V_0 沿各个不同方向，同时抛出 N 个物体(视为质点)，验证：

(1)任意时刻 t，各质点总是落在某一园周上；

(2)园心自由下落，半径与时间 t 成正比增大。

2. 问题分析(采用国际单位)

设抛射点 O 为坐标原点，水平向右为 X 轴正方向，竖直向上为 y 轴正方向，抛射体的初速率为 V_0。任意一个质点 i(i=0,1,……,N−1)的抛射角为 θ_i(抛射初速度与 X 轴正方向的夹角)，又设抛射时刻为计时起点。若不计空气阻力，质点的运动方程为

$$X=V_0\cos\theta_i \cdot t \qquad ①$$

$$Y=V_0\sin\theta_i \cdot t-\frac{1}{2}gt^2 \qquad ②$$

式中 g 为重力加速度，其值取为 9.8，t 是抛射体运动的时间。为使屏幕显示效果较好，本程序设计初速度 V_0 为 16。由①②两式消去角度 θ_i 得到

$$X^2+(Y+\frac{1}{2}gt^2)^2=(V_0t)^2 \qquad ③$$

由方程③可以看出：

(1)在某一时刻，t 值一定，③式为一园方程，其园心坐标为 $(0,-\frac{1}{2}gt^2)$，半径为 V_0t。因此，在任意时刻 t，从同一点 O，以同样的初速率 V_0，不同的抛射角 θ_i 抛出的 N 个质点总是散落在以 V_0t 为半径，以 $(0,-\frac{1}{2}gt^2)$ 为园心的同一园周上。

(2)园心 $(0,-\frac{1}{2}gt^2)$ 从抛射点 O 自由下落，园的半径 V_0t 与时间 t 成正比增大。

以上两点，通过微机模拟可以得到验证。

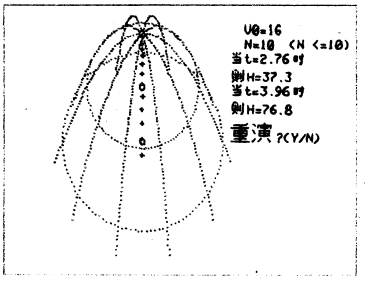
3. 模拟程序简介

本程序显示的汉字、图形和字符，都是用造型表建立的。通过 XDRAW M AT X,Y 语句将编号为 M 的图形显示在坐标为 X,Y 的位置。键入程序后，再在监控状态下键入造型表代码便可运行。

4. 模拟过程说明

程序首先在显示屏上方印出坐标原点和初速。依照提示，键入抛体个数 N(本程序设计 N≤10)再按回车键，电脑便从 O 处以同样的初速率 V_0，不同的抛射角 $\theta_i(\theta_i=\frac{2\pi}{N}i)$，同时抛出 N 个质点。随后各个质点分别作抛体运动。

在抛体运动过程中，只要按一下 T 键，电脑就画园。可以看到，任一时刻各质点确实处在同一园周上，画完园后，电脑又适时地给出抛体运动的时间 t，园



心下落距离 h 和半径 R 之值。屏幕上每一瞬时显示的小园表示园心的瞬时位置，“+”表示园心运动的等时间隔轨迹，很容易看出园心是作自由落体运动的。

相继按两次 T 键，从先后画两次园，给出两次 t、R 之值，均能表明半径 R 与时间 t 成正比增大。

演示到最后，电脑提示“重演？(Y/N)”。若要重演则键入 Y 重新开始；键入 N 则退出。

```
100 POKE 232,0 :POKE 233,96
300 HGR2 : HCOLOR= 3: ROT = 0: SCALE=1
310 HPLOT 0,0 TO 279,0 TO 279,191 TO 0,191 TO 0,0
320 LET W=2 * 3.14159265:X0 =110:Y0 = 20:G = 9.8:
V0 =16: C= 190: XDRAW 7 AT X0−9,Y0+7
325 HW = −2:HT = C:VT = 20:ST$ = "V0=16": GO-
SUB 10000
330 HT = C:VT = VT+10:ST$ = "N=? (N<=10)":
GOSUB 10000
335 INPUT N
336 IF N < 0 OR N > 10 THEN HT =C:ST$ = "N=? (N
<=10)": GOSUB 10000:VT = VT−10: GOTO 330
338 XDRAW 31 AT C+12,VT
339 HT = C + 12:VT = VT:ST$ = STR$(N): GOSUB
10000
360 AX = V0 * COS (W/N)
370 AY = V0 * SIN (W/N)
380 J = .06: FOR K= 1 TO 72:T =K * J
400 FOR I=1 TO N
410 X(I) = X0 + V0 * COS(I * W/N) * T
415 Y(I) = Y0 − V0 * SIN (I * W/N) * T+4.9 * T *
T
419 NEXT I
420 LET RX = AX * T:RY=AY * T: R = SQR (RX * RX
+ RY * RY):H = 4.9 * T * T
423 IF INT(K/5) = K/5 THEN XDRAW 6 AT X0−2, Y0 +
1 +G * T * T /2
425 FOR I=1 TO N:HPLOT X(I),Y(I): NEXT
```

```
427 IF PEEK (- 16384) = 212 THEN POKE -16368,0:
GOSUB 510
490 NEXT K
501 VW = 1:HT = C - 10:VT = VT+ 20: XDRAW 1 AT
HT,VT: XDRAW 4 AT HT +17,VT
502 HT = HT +35
503 ST$ = "? (Y/N)":GOSUB 10000
504 GET A$
507 IF A$ = "N" THEN TEXT : END
508 IF A$ < > "Y" THEN 504
509 GOTO 300
510 XDRAW 3 AT C - 10, VT + 10:HT = C: VT = VT +
10: ST$ = "@=" +STR$ (T) :GOSUB 10000
511 XDRAW 5 AT C + 40, VT+ 2
512 DRAW 7 AT X0 - 9,Y0 + 7 + G *T *T/2
513 FOR P = 0 TO W STEP 3.1415926 *3 /180
514 HPLOT X0 + R * COS (P) ,Y0 +H-R * SIN (P)
516 NEXT P:HW = -2
520 XDRAW 2 AT C - 10,VT +13:HT = C +2:VT = VT
+13: ST$ = "H=" +STR$ (INT (H * 10 +.5)/10):
GOSUB 10000
530 RETURN
10000 FOR I = 1 TO LEN (ST$)
10020 CH = ASC (MID$ (ST$,I,1)) -32
10030 HH = 8 + HW: IF CH = 0 THEN 10060
10050 XDRAW CH AT HT, VT
10060 HT = HT + HH
10070 NEXT I
10090 RETURN
```

```
6000-39 00 74 00 AE 00 CD 00
6008-E3 00 19 01 39 01 3E 01
6010-52 01 5A 01 61 01 61 01
6018-61 01 61 01 72 02 62 01
6020-69 01 78 01 82 01 92 01
6028-A2 01 AE 01 BB 01 CA 01
6030-D8 01 E7 01 61 01 61 01
6038-F7 01 01 02 61 01 09 02
6040-16 02 61 01 61 01 61 01
6048-61 01 61 01 61 01 61 01
6050-21 02 61 01 61 01 61 01
6058-61 01 61 01 3D 02 61 01
6060-61 01 61 01 4D 02 61 01
6068-61 01 61 01 5A 02 61 01
6070-61 01 69 02 2D 2D 2D 2D
6078-2D 2D 2D C4 3B 3F 3F E6
6080-3F 3F 04 28 2D 2D 66 2D
6088-2D 24 24 3F 3F 3F 3F 3F
6090-36 66 2D 2D 26 70 2D 25
6098-40 18 49 3F 3F 3F 37 1C
60A0-3F 3F 3F 44 2D 2D 2D 36
60A8-40 2D 2D 05 20 00 01 40
60B0-03 24 24 64 2D 15 36 36
60B8-FE 24 24 96 D2 1E 4D 70
60C0-0D 40 03 24 24 44 31 36
60C8-36 36 1E 04 00 29 2D 2D
60D0-25 24 3C 3F 3F B7 29 2D
60D8-25 C0 23 FC 73 4D 71 40
60E0-03 04 00 05 28 20 40 18
60E8-1C 1C 04 40 18 A8 0E 4E
60F0-24 2D 2D 25 0E 2D 2D 36
60F8-1F 3F 3F 3F 96 28 2D 74
6100-2D 35 3E 3F 27 1E 3F 27
6108-B4 32 2D 2D 74 2D 25 96
6110-73 0E DF DB DB 2B 28 20
6118-00 40 18 24 24 2D 36 77
6120-9D 6F 40 18 08 2D E5 16
6128-36 F6 07 20 B0 40 03 96
6130-70 F5 07 20 40 03 0E 04
6138-00 28 E5 16 04 00 49 49
6140-09 40 18 40 18 40 18 BF
6148-76 73 2D 0E 40 03 24 1C
6150-04 00 49 1C 1C 24 0C 0C
6158-04 00 09 05 28 20 E4 1C
6160-04 00 C9 0C 0C 0C 0C 04
6168-00 01 20 24 64 2D 15 36
6170-36 1E 3F 04 60 0C 04 00
6178-09 25 24 24 EC 13 8D 92
6180-22 00 01 40 18 40 03 0C
6188-2D 15 F6 C6 1E 1E 2E 2D
6190-25 00 01 40 18 40 18 2D
6198-2D 1E 1E 0E 15 F6 3F 07
61A0-20 00 49 21 24 24 3C 1E
61A8-1E 36 2D 0D 04 00 01 A8
61B0-46 2D 20 3C 38 3F 24 2D
61B8-2D 04 00 09 2D 05 20 1C
61C0-3F 37 26 40 03 0C 0C 2D
61C8-04 00 01 40 18 4D 1B 2D
61D0-2D A6 1A 1E 1E 36 04 00
61D8-09 2D 05 20 1C 3F 17 26
61E0-40 03 64 2D 15 26 00 01
61E8-40 18 20 0C 2D 15 B6 38
61F0-3F 4D F2 1E 3F 04 00 49
61F8-C9 07 38 38 60 0C 0C 04
6200-00 01 28 2D 25 38 3F 27
6208-00 01 40 18 40 03 0C 2D
6210-15 F6 BF 16 04 00 49 09
6218-B8 3F 20 E4 2D E5 23 04
6220-00 21 24 24 6C 09 36 3E
6228-3F 4E 31 26 00 21 24 24
6230-6C 09 36 36 36 DF 40 18
6238-78 11 0E 04 00 21 24 24
6240-6C 09 36 36 36 DF 40 18
6248-78 11 0E 04 00 21 24 24
6250-2C 2D 15 F6 3F 0E 0E 0E
6258-04 00 01 40 18 40 18 36
6260-36 0E 15 05 28 20 24 24
6268-00 49 24 E4 1C 6C 09 F6
6270-04 00 49 04 00
```

多单元电视接收天线设计程序

新疆石河子农垦中专　　史文忠

电视天线的种类很多，本程序是多单元定向天线的 VHF 和 UHF 频带的工作设计。(1—60 频道)

定向天线制做简单，增益高，是一种常用天线。此程序设计的天线可对单一频道接收，也可接收同一方向的多频道电视节目。

使用方法：①当程序运行后，输入所需接收的最低频道值，回车后，再输入所需接收的最高电视频道值，回车后，屏幕立即显出天线的各种数据。如需接收单一频道，两次的输入为同一频道值即可。②数组表中数据不能变更③天线的引向振子长度取相等，引向振子间距也取相等。

```
10 HOME (CLS)
20 INPUT "接收最低频道值(D)";M
```

(下转第 24 页)

一个实用的造型表编辑程序

北京一〇一中学　初三(1)班　左震宇

造型表是APPLE机的一个良好的绘图工具。但是造型表编制步骤繁琐，虽然已有不少制作造型表的辅助软件，但这些程序总不能使人十分满意。

经过本人多次实践，编制成功了一个实用的造型表编辑程序。本程序制表速度快，精度高，功能齐全，输入方便。程序由输入和扫描两部分组成。程序开始运行，首先输入造型数和存入内存的首址(必须在$6000以上)。接着在屏幕左侧有一个47×47的点阵，这是作图区。在右边有一个方框，框内可直接映射出作图区中的图形，以便直观地看到屏幕的效果。在点阵中央有一条短横线做为光标，用I、J、K、L键控制光标上下左右移动。对于点的画和擦，本程序采取了两种输入方法：第一种是用U键画上或擦去一个点；另一种是用I、J、K、L直接擦点或画点，C键用来切换擦或画两种模式。这两种绘图方式由P键切换。为了更好的利用作图区和让作图更准确，程序提供了座标输入。接T键后，输入横座标、纵座标，便可将光标移至对应处。若按Q键可退出编辑，进入扫描部分。扫描部分就是由用户将作图区中的造型扫描一遍(用I、J、K、L键移动光标)。如果想修改造型，返回编辑状态，可按B键。如果扫描完毕，想退出编辑状态，可按Q键。使用过其它造型表编辑程序的用户，可能有这样的发现，有时一个造型表打入时没有什么地方可挑剔，可打出来与原造型大不相同，这是由于APPLESOFT对于$01－$07的造型编码运行有误。本程序对于这个问题做了完善的处理。使打出的造型完全正确。当你退出编辑状态后，计算机会询问你是否要把这幅图拷贝到下一幅图中。若是键入Y则在下一个造型编辑时，作图区中仍然保留上一幅图，便于稍作修改，造出相差不多的第二幅图，这样可提高制表效率。若键入N则在编辑下一个造型时，先将作图区清除干净。在所有造型都编辑完之后，计算机将询问是否存盘。若键入Y，则在键入文件名后可将造型表存盘。

```
10  GOTO 80
20  X2=X:Y2=Y:X=X+(A=76 AND X<185)*4-(A=74 AND X>3)*4:Y=Y+(A=75 AND Y<184)*4-(A=73 AND Y>3)*4:RETURN
30  POKE -16304,0:POKE -16297,0:POKE -16302,0:POKE -16299,0:RETURN
40  POKE N2,S(S):N2=N2+1
50  IF S=0 OR S=4 THEN POKE N2,8:N2=N2+1
60  RETURN
70  POKE 232,B1:POKE 233,B2:DRAW I/2 AT X3,Y3:POKE 232,0:POKE 233,3:RETURN
80  FOR I=768 TO 825:READ A:POKE I,A:NEXT:POKE 232,0:POKE 233,0:DATA 2,0,6,0,10,0,62, 36,53,0,47,0
85  POKE 49238,0
90  DATA 24,165,107,105,9,133,6,133,8,165,108,133,7,144,2,230,7,105,18,133,9,160,0,169,0,145,6,230,6,208,2,230,7,165,6,197,8,208,240,165,7,197,9,208,234,96
100  DIM A%(47,47):SCALE=1:ROT=0
110  FOR I=0 TO 7:READ S(I):NEXT :DATA 24,8,16,24,28,12,20,28
120  HOME:VTAB 1:INPUT "How many shapes?";N:VTAB 5:INPUT "Start address in RAM?";M
130  N2=M+2+2*N:D=N2-M +1:B2=INT(M/256):B1= M-B2*256:POKE M,N:POKE M+1,0
140  FOR I=2 TO 2*N STEP 2
160  POKE M+I+1,INT ((N2-M )/256):POKE M+I,N2-PEEK (M+I+1)*256-M:POKE N2,0:POKE N2+1,0
170  IF CPY=1 THEN GOSUB 30:GOTO 190
180  HGR2:HCOLOR=1:FOR P=0 TO 191 STEP 4:HPLOT 0,P TO 188 ,P:HPLOT P+1,0 TO P+1,188:NEXT :CALL 780
190  X=95:Y=94:X1=24:Y1=24
200  HCOLOR=3:HPLOT 203,69 TO 255,69 TO 255,119 TO 203,119 TO 203,69
210 HCOLOR=3-A%(X1,Y1):DRAW 2 AT X,Y
220 GET A$
230 HCOLOR=A%(X1,Y1):DRAW 2 AT X,Y:HCOLOR=3
240  A=ASC(A$):GOSUB 20
260  X1=(X+1)/4:Y1=(Y+2)/4
270  IF A=85 THEN A%(X1,Y1)=3 -A%(X1,Y1)
280  IF PUT=1 THEN A%(X1,Y1)=C2
290  HCOLOR=A%(X1,Y1):DRAW 1 AT X,Y
300  IF A=81 THEN X3=205+X1 :Y3=140+Y1:GOTO 380
310  IF A=80 THEN PUT=ABS(PUT-1)
320  IF A=67 THEN C2=ABS(C2 -3)
330  HPLOT 205+X1,70+Y1
340  IF A=84 THEN TEXT:VTAB 20 :INPUT "INPUT X,Y";X3,Y3:X1 =X3:Y1=Y3:X=X1*4-1 :Y=Y3*4-2:GOSUB 30
370  GOTO 210
380  POKE -16368,0:S=0
390  FOR J=0 TO 3 STEP 3
400  POKE N2,0
410  HCOLOR=3-A%(X1,Y1):DRAW 2 AT X,Y
420  GET A$
430 HCOLOR=A%(X1,Y1):DRAW 2 AT X,Y:HCOLOR=3
440  A=ASC(A$):GOSUB 20
460  IF X=X2 AND Y=Y2 AND A<>66 AND A<>81 THEN 410
```

```
470  IF A=81 THEN GOTO 590
480  COLOUR=A%(X1,Y1)*4/3
490  X1=(X+1)/4:Y1=(Y+2)/4
500  S=(((Y<Y2)*0+(X>X2)+(Y>Y2)*2+(X<
X2)*3)+CO)*INT(2^J)+S
510  IF A=66 THEN N2=M+D:HCOLOR=0:GOSUB 70:
GOTO 190
520  HCOLOR=3:GOSUB 70
530  NEXT
540  POKE N2,S
550  IF S>7 THEN N2=N2+1:GOTO 380
560  GOSUB 40:GOTO 380
570  POKE N2,S:IF S>7 THEN N2= N2+1:GOTO 590
590  POKE N2,0:N2=N2+1
600  IF I=2*N THEN 630
610  TEXT:VTAB 9:PRINT "Copy this shape to next one?";:
GET A$:IF A$="Y" THEN CPY= 1:GOTO 630
620  CPY =0
630  N1=N1+2:D=N2-M
640  HCOLOR=0:GOSUB 70:NEXT I
650  TEXT:VTAB 12:PRINT "Save to disk? (Y/N)";:GET A
$:IF A$="Y" THEN VTAB 16:INPUT "File name?";N$:
PRINT CHR$ (4);"BSAVE";N$;",A";M;",L";D
660  END
```

(上接第 22 页)

```
30 INPUT "接收最高频道值(G)";N
40 FOR PS = 1 TO 60
50 READ PL
60 IF M = PS THEN PD = PL
70 IF N = PS THEN PG = PL
80 NEXT PS
100 PO = SQR (PD * PG)
110 DE = INT (0.3 * PG+0.5)
120 DI = INT (0.6 * DE)
130 DR = INT (0.19 * PD-0.5)
140 IE = INT (0.42 * PG)
150 IR = INT (0.5 * PD+0.5)
160 EI = INT (0.95 * PO/2)
170 B = INT (0.003 * PO+0.5)
180 HOME : PRINT"          ";M;"频道至";N;"频道的天
线数据"
200 PRINT "引向振子的间距          =";DE;"cm"
210 PRINT "引向振子与馈电振子间距  =";DI;"cm"
220 PRINT "反射振子与馈电振子间距  =";DR;"cm"
230 PRINT "引向振子长度            =";IE;"cm"
240 PRINT "反射振子长度            =";IR;"cm"
250 PRINT "馈电振子长度            =";EI;"cm"
260 PRINT "馈电振子宽度            =";B;"cm"
270 PRINT "馈电振子开口距          =";INT(0.027 *
                                    PO+0.5);"cm"
300 DATA 571,496,438,375,340,175,168,160,154,148,
142,137,63.3,62.2
310 DATA 61.2,60.2,59.3,58.4,57.5,56.6,55.8,55,54.
2,53.4,49.2,48.5
320 DATA48,47.3,46.7,46.2,45.6,45.1,44.5,44,43.5,
43,42.5,42,41.6
330 DATA 41.1,40.7,40.2,39.8,39.4,39,38.6,38.2,37.
8,37.4,36.66
340 DATA 36.32,35.97,35.63,35.29,34.97,34.64,34.32,
34.01,33.71
```

(上接第 31 页)

```
114          RET
115  HIGH:   MOV R0,#02H
116          RET
117  FREQ:   DB 0FCH,43H,0FCH,0ACH,0FDH,82H,
             0FDH,0C7H;音频表
118          DB 0FDH,09H,0FDH,33H,0FEH,05H
119          DB 0F8H,8BH,0F9H,05BH,0FBH,04H,
             0FBH,8FH
120          DB 0FAH,14H,0FAH,06BH,0FCH,0BH
121          DB 0FEH,21H,0FEH,56H,0FEH,0C1H,
             0FEH,0E3H
122          DB 0FEH,84H,0FEH,99H,0FFH,02H
123  KEYBL:  DB 41H,11H,12H,21H;键值表,代表:
             0 1 2 3
124          DB 22H,42H,51H,52H;
             4 5 6 7
125          DB 32H,62H;低音键
126          DB 31H,61H;高音键
```

以上程序是在 KDC－Ⅲ开发机上调试成功的。这时 2764(EPROM)地址 0000H,8155 的命令/状态寄存器地址为 4100H,输入控制字 0DH 时,设置功能是 C 口(地址为 4103H)为输出,B 口(地址为 4102H)为输入。

此程序较好解决了音频和音长的统一,比一般方法压缩了一半数据。

在硬件也可以脱离 KDC－Ⅲ单片开发机,单独地成为一个应用系统。可以用 P1.0～P1.6(8031 P1 口)来完成扫描键盘和输入键值,用 P1.7 驱动刺叭。整个系统可由 8031,2764,373 三片组成的最小应用系统组成。这时 2764 片选信号接地。从而真正构成一个独立电子琴。也可有 2764(8K)存储近 50 首乐谱的数据,使其具有自动演奏的功能。

当然要逼真地模拟音乐,还仍有一些问题要解决。音乐包括四要素:音频,音长,音色和音强。音频和音长用单片机比较好解决。音强可以用 D/A 转换控制,而音色的模拟将还有一定的问题。尽管如此,用单片机产生音乐将是十分有意义的工作。它将是一个无任何转动的机构的数字信号产生音乐系统,正向各类游戏机一样,音乐是存在 EPROM 中的,对于需要设计带有自动音响功能的电子产品,采用单片机音乐系统,不失为一种可行的技术方案。

C 语言初阶讲座

第五讲　程序控制结构(一)

李文兵　王玉华　蓝智斌

本讲介绍 C 程序控制结构。在 C 程序中,控制程序流程,使用控制结构。控制结构是由控制语句构成的。控制结构大致分为三种,即:

- 条件分支结构
- 循环控制结构
- 中断与转移结构

使用这三种控制结构,就能够编制出结构化程序。

本讲的最后,介绍递归函数,这是一种特殊的程序结构形式。

1　分支结构

(1)if **结构**　其结构形式如下所示:

```
if(条件表达式)  语句;
```

例如,求 a,b 两个数中大者的程序,就可以使用 if 结构编写,如练习 5.1 所示。

(练习 5.1)

```
A>TYPE EXP5__1.C
main()
    { int a,b,x;
      a=5;b=3;
      x=b;
      if (a>b) x=a;
      printf("max=%d\n",x);
    }
A>EXP5__1
max=5
```

(2)if～else **结构**　其结构形式如下:

```
if (条件表达式)  语句 1;
else   语句 2;
```

(练习 5.2)

```
A>TYPE EXP5__2.C
main()
    { int a=5,b=3,x;
      if (a>b) x=a;
        else x=b;
      printf("max=%d\n",x);
      }
A>EXP5__2
max=5
```

(3)**嵌套** if～else **结构**　if～else 结构中还可以放进 if 结构或 if～else 结构,这叫做嵌套结构:

```
①  if(条件表达式 1)
   ②  if(条件表达式 2)
       ③  if(条件表达式 3)  语句 3;
          else   语句 4;
       else   语句 2;
   else   语句 1;
```

在这种嵌套结构中,为使嵌套层次关系清晰,我们提倡程序员把程序写成如上所示的齿状模样。

(练习 5.3)

```
A>TYPE EXP5__3.C
main()
{ int number=55;
  /* nested if statement 1 */
    if (number>0){
      if(number%2==1)
        printf("%d positive odd\n",number;
    else
      printF("%d negative\n",number);
    /* nested if statement 2 */
      number=-number;
      if (number>0){
          if (number%2==1)
            printf("%d positive odd\n",number);
        else
            printf("%d positive even\n",number);
        }
      else{
        if (number%2==0)
            printf("%d negative even\n",number);
        else
            printf("%d negative odd\n",number);
        }
      }
A>EXP5__3
55 positive odd
-55 negative odd
```

(4)**多分支** if～else if **结构**　其结构形式如下:

```
if(条件表达式 1)  语句 1;
else if(条件表达式 2)  语句 2;
else if(条件表达式 3)  语句 3;
:
:
else if(条件表达式 n)  语句 n
else   语句 n+1;
```

在这种结构形式中,程序只执行满足条件的语句。如果哪个条件都不成立,就执行语句 n+1。

（练习 5.4）

```
A>TYPE EXP5__4.C
main()
{ unsigned char c;
printf("enter char:");
scanf("%c",&c);
if (c>=0x00 && c<0x20)
    printf("%x,control code",c);
else if (c>=33 && c<47)
    printf("%x [%c]:special char(group 1)\n",c,c);
else if (c>=48 && c<=57)
    printf("%x [%c]:digits\n",c,c);
else if (c>=58 && c<=64)
    printf("%x [%c]:special char(group 2)\n",c,c);
else if (c>=65 && c<=90)
    printf("%x[%c]:uppercase letter\n",c,c);
else if (c>97 && c<=122)
    printf("%x [%c]:lowercase letter\n",c,c);
else if (c>91 && c<=96)
    printf("%x [%c]:special char(group 3)\n",c,c);
else if (c>=123 && c<=126)
    printf("%x [%c]:special char(group 4)\n",c,c);
else
    printf("%x [%c]:non ascii char\n",c,c);      }

A>EXP5__4
enter char:1
31 [1]:digits

A>EXP5__4
enter char:$
24 [$]:special char(group 1)

A>EXP5__4。
enter char:&
26 [&]:special char(group 1)
```

(5)Switch **结构**　Switch 语句可实现多分支结构：

```
switch(变量)
  { case  常量表达式 1:
            语句 1;
            break;
    case 常量表达式 2:
            语句 2;
            break;
              ⋮
    case  常量表达式 n:
            语句 n;
            break;
    default:
            语句 n+1;
    }
```

• 当 switch 括号中的变量值与某 case 中常量表达式的值相同时，就执行该 case 中的语句。若该语句后跟有 break 语句，则跳出 switch 结构。若无 break 语句，则顺序执行下一个 case 中的语句。若变量的值与任何 case 的常量表达式的值都不相同，则执行 default 中的语句。使用 switch 结构需要注意以下几点：

(1)switch(变量)中的变量只限字符型(char)、整型(int)、枚举型(enum)。

(2)switch 语句也可嵌套使用。

(3)若干个 case 可公用一个语句，如练习 5.5 所示。

（练习 5.5）

```
A>TYPE EXP5__5.C
main()
      { int i;
        printf("enter i=");
        scanf("%d",&i);
        switch(i)
          { case 1:
                printf("case 1\n");
                break;
            case 2:
                printf("case 2\n");
                break;
            case 3:
            case 4:
            case 5:
                printf("case 3-5\n");
                break;
            default:
                printf("default");
            }
      }
A>EXP5__5
enter i=1
case 1

A>EXP5-5
enter i=9
default
```

2　循环结构

用下面的循环控制语句，便可实现循环结构。

- for
- while
- do—while

(1)for **结构**　for 结构的形式如下所示：

for(表达式 1;表达式 2;表达式 3)　语句；

其中表达式 1 一般用来初始化循环控制变量；表达式 2 为条件表达式，以此控制循环次数；表达式 3 用来修改循环控制变量。例如：

```
int i;
for (i=0;i<100;i++)
    printf("%d",i);
```

和

```
int i;
for (i=100;i>0;i--)
```

```
    printf("%d,i);
```

此外，可以使用两个循环控制变量，如：

```
int i,j;
for (i=0,j=100;i<j;i++,j--)  语句;
```

（练习 5.6）

```
A>type exp5__6.c
main()
    { int i,j=0;
      /* for statement 1 */
         for (i=0;i<100;i++) j+=i;
         printf("summation=%d\n",j);
    }
A>exp5__6
summation=4950
```

(2)**无限循环的 for 结构**　for 语句的三个表达式中任何一个都可以缺省。当表达式 2 缺省时，C 编译系统则认为表达式 2 恒为真，故 for 结构就变成无限循环结构，其特例如下所示：

```
for (;;) 语句;
```

为使无限循环的 for 语句在某条件下终止，可加入 break 语句，如练习 5.7 所示。

（练习 5.7）

```
A>TYPE EXP5__7.C
main()
    { int i,j;
      i=0,j=1;
      for (;;)
      { int k;
        k=j,j=i+j,i=k;
        if (k>100) break;
        printf("%d ",i);
      }
    }
A>EXP5__7
1  1  2  3  5  8  13  21  34  55  89
```

(3)**while 结构**　其结构形式如下所示：

```
while(条件表达式)  语句;
```

功能是：只要条件表达式成立，就执行条件表达式后所跟的语句；该语句可以是空语句、简单语句、或复合语句。例如：

```
int  len=10;
while (len>0)
{ printf("*");len--;}
```

只要 len 为正，该程序就打印 * 号。

while 的条件表达式中可引进字符输入库函数，如：

```
while ((ch=getchar())! ='\n');
```

其中 getchar()为字符输入库函数。注意：该 while 语句的执行语句就是空语句。这样，只要从键盘上输入的字符不是\n，输入操作就可以连续进行下去。

（练习 5.8）

```
A>TYPE exp5__8.c
#include <stdio.h>
main()
{ char ch;
  int len=10;
  char s[80];
  printf("while statement test 1\n");
  while (len>0)
  { printf("*");
  len--;
  }
  printf("\nwhile statement test 2\n");
  while (scanf("%d",&len)! ='\n')
      printf(":len=%d\n",len);
}
A>exp5__8
while statement test 1
* * * * * * * * * * *
while statement test 2
2  3  4  5  (仿效返回)
:len=2
:len=3
:len=4
:len=5
^C
```

执行结果中注有仿效返回的行，是为使操作者及时了解输入情况而设的操作系统功能。紧跟在仿效返回下面的，才真正是程序执行结果。

(4)**无限循环 while 结构**　其结构形式为：

```
while(1){……}
```

只要圆括号中的值非 0，就是无限循环 while 结构。

（练习 5.9）

```
A>TYPE EXP5__9.C
main()
    { int len;
      printf("while statement test\n");
      len=10;
      while(1)
      { printf("%d ",len--);
        if (len<=0)
          break;
      }
  }
A>EXP5__9
while statement test
10  9  8  7  6  5  4  3  2  1
```

(5)**do－while 结构**　其一般结构形式是：

```
do{语句;}
while (条件表达式);
```

若条件表达式成立，则执行语句。与 while 结构不同的是，不管条件成立与否，首先要执行一次 do 后面的语句。如练习 5.10 所示

```
A>type exp5__10.c
#include <stdio.h>
main()
```

```
{ char ch;
int len;
printf("do-while statement test 1\n");
len=10
do { printf(" * ");
    } while((len--)>=0);
printf("\ndo-while statement test 3\n");
len=50;
do { printf("%d ",len-=5);
    if (len<=0) break;
  } while(1);
}
A>exp5__ 10
do-while statement test 1
* * * * * * * * * * * * *
do-while statement test 3
45  40  35  30  25  20  15  10  5  0
```

（待续）

语音报时钟的原理和实现

清华大学　蔡莲红

石英电子表走时准确，使用方便，它使钟表工业向前跨了一大步。近年来，语音合成技术的进步，使语音闹时、报时的愿望得以实现。比如，时钟到了设定的时间，它用话音告诉你："时间到了。"有的时钟能随时向您报告准确时间。本文将简单介绍一下语音报时的基本原理和实现。

一、语音报时钟简介

图 1 表示了语音报时钟的原理框图。它是在电

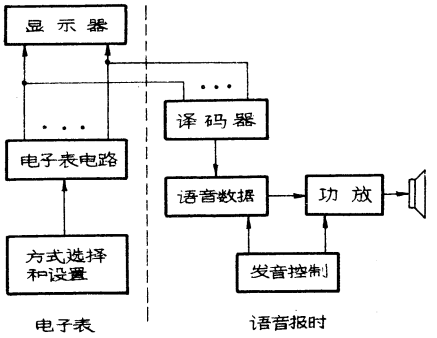


图 1　语音报时钟原理框图

子表的基础上，增加了语音报时功能。市场上已有大量的电子表集成块，本文不再讨论。语音报时钟的方式选择和设置跟一般电子表相同。如应具有设置月、日、时、分，设置闹时的时、分功能。因而它应有一个方式选择开关——选择闹时、时间、日期。还应有两个按钮：时(HR)、分(MIN)。它们组合起来实现方式选择和设置功能，如表 1。

语音报时部分由译码器、语音数据、发音控制和功效组成。语音数据是数字化了的语音波形。若以语音报时，首先要知道当时的时间。在不改动电子表集成电路的情况下，以连到显示器的连线中截获信号，并进行译码。译码器输出的是待发声的语音数据的地址。然后读出数据，报出时间。发音控制的作用是实现声音强/弱、报时开/关、报时按钮的控制。

表 1　　电子表方式选择和设置功能

		方式选择器位置			所用键	功　能
		闹时	时间	日期		
闹　时	小时	↑			HR	按HR键，小时加1
	分	↑			MIN	按MIN键，分钟加1
设置时间	小时		↑		HR	按HR键，小时加1
	分		↑		MIN	按MIN键，分钟加1
设置日期	小时			↑	HR	按HR键，小时加1
	分			↑	MIN	按MIN键，分钟加1

二、语音报时的基本原理

1. 语音数字化表示

语音信号处理技术的发展，把模拟语音信号变成数字量存储起来，需要时再恢复的应用日益广泛。事实上，处理数字化语音更为经济、其放音也易于控制。好语音圣诞卡、语音玩具、语音闹时和报时就是最简单的应用。

数字化语音的获取和重放原理如图 2 所示。从话筒输出的模拟语音信号，经放大和滤波后，送模数转换器(A/D)。A/D 按一定时间间隔(按一定的抽样率)，将模拟语音信号变成数字量。这些有序排列的数据。其数值大小表示语音抽样的幅度，就是数字化的语音。A/D 每完成一次转换，都把数据存入存储器中。需要放音时，先从存储器中取出数据，经数模转换变成模拟量，再经功率放大和滤波，即可放音了。在上述过程中应解决好以下几个问题：

(1)滤波器的截止频率和语音抽样率：首先根据系统对发音质量的要求，确定语音抽样率。在一些简单的应用系统中，语音抽样率为 6K—10KHz。根据采样定理，相应滤波器的截止频率为 3K—5KHz。

(2)语音抽样的码位：在数字表示中，采用二进制编码是方便的。这里码位指二进制编码的码位，也就是指 A/D 转换的位数。一般取 8—12 位。

语音的数字化过程，通常在计算机或专用开发系统中完成。

2. 数字化语音的放音：

如果已经获得语音数据，可将其写入 EPROM 中，

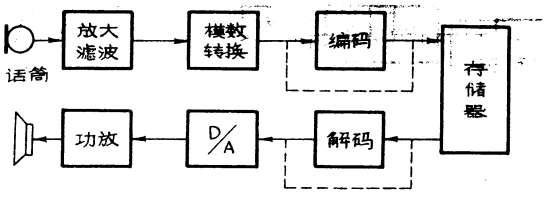


图2　语音数字化和放音

利用图 3 的电路放音。图中，三片 74LS161 是 EPROM 的地址计数器。开关 K 未接通时，地址计数器输出 0 地址。开关 K 接通一次，单稳电路 74LS123 输出一正脉冲，地址计数器在 555 供给的时钟作用下，开始计数。1408 是数模转换器，LM386 是功率放大器。值得指出的是：555 输出的时钟频率应与原 A/D 转换的频率相等。

3. 语音的编码和解码

按照上述的原理实现语音数字化和放音，线路简单，但数据量大。如果语音抽样率为 8KHz，每个抽样

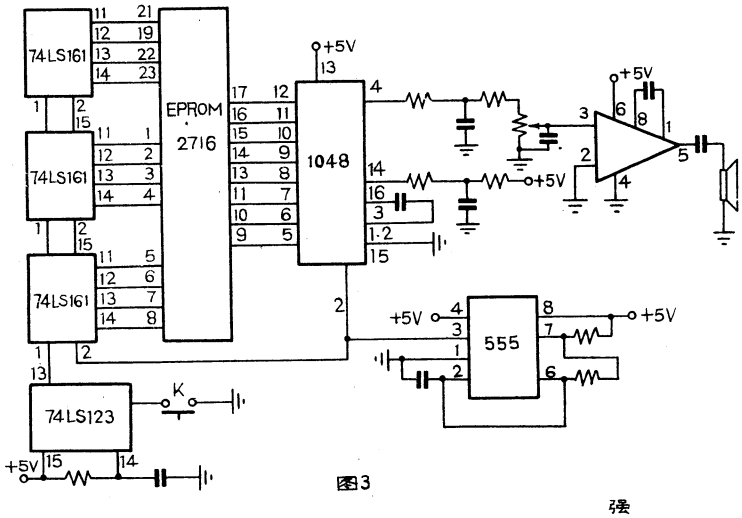


图3

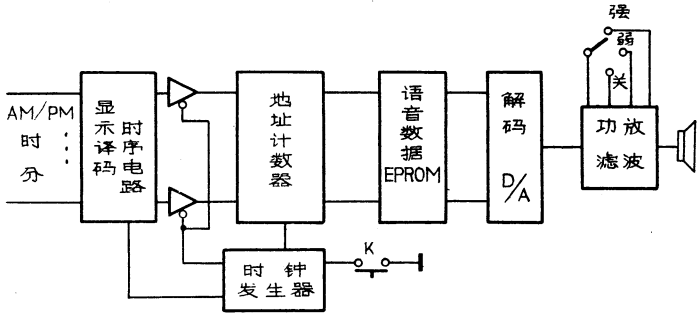


图4　语音报时的实现框图

用 8 位二进制表示，那么每秒钟得到 8KB(64K 位)语音数据。若想讲“早上五点五十八分”，大约需 3 秒钟，也就是说需 24KB 存储器。若要能报任何时和分，需要的存储器就更大了。研究发现，可以对语音数据进行压缩编码，而基本不影响发音质量。带来的益处是减少了数据量。因此在实用系统中，几乎都采用了压缩编码，或由硬件实现，或由软件实现。放音时，再进行解码。这就是图 2 中编码和解码的作用。

在语音的编码/解码技术中，最常用的是自适应差值脉冲编码(Adaptive Delta Pulse Code Modulation 简称 ADPCM)。市场上已有多种 ADPCM 编解码的集成电路，下面仅列出几种：OKI 公司的：MSM 5205、MSM5218、MSM5248、MSM6295，TI 公司的 TSP50C41(42/43/ 44)，Toshibar 的 T6831，Harris 公司的 HC55536、HC55564 和 SamSung 的 KS5911 和 KS5912 等。采用 ADPCM 编码，可将原抽样的 8 位数据，压缩到 4、3、2 或 1 位，大大减少了数据量。其放音电路的组成与图 3 类似，只是将数模转换集成电路换成专用 ADPCM 集成电路。

三、语音报时的实现

要用语音报时，先确定要报的音，如：月、日、点、分、1—59、早上、上午、下午和晚上等。然后建立标准数字音库(写好 EPROM)、各发音的首地址索引，最后按

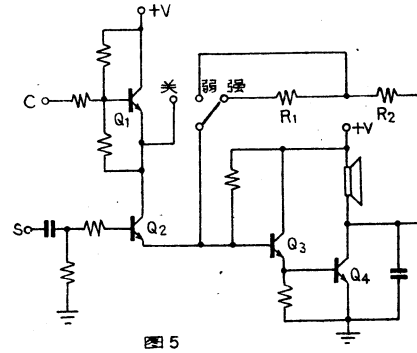


图5

图 4 的框图设计具体电路。其中显示译码的作用是把送显示器的 AM/PM、时、分、月、日信号取来，译出待发音的 EPROM 首地址。由于每次报音需由几个短音组合而成，因此还需设计一时序电路，顺序定时送出 EPROM 地址。这部分电路可由 TTL 集成电路或门阵电路实现。图 4 中其它部分都已熟悉。这里仅介绍一简单廉价的功放滤波电路，如图 5。图中 S 是语音信号输入端。C 是控制端。一般情况下(报时)，C 为高，Q1 导通，关掉报时开关时，C 为低，Q1 截止，Q2 不能传送语音信号。Q3 和 Q4 构成反相功率放大器，一个三掷开关，选择发音强、弱和无音。实际上声音强弱的控制是靠改变功放反馈电阻实现的。反馈电阻增大为 R1+R2 时，声音强，为 R2 时，声音弱。当开关拨到“无音”时，线路中将语音信号短路，故无声音输出。

图 4 中，除功放滤波控制开关和扬声器外，均可做成一块中规模集成电路，若再配一块电子表电路和显示器，就可形成一个语音报时钟。

学用单片机

简易单片机电子琴

张培仁　张新宇

在本刊91年第二、三期上，向大家介绍了KDC－Ⅲ的软硬件设计。在第四期上介绍了自动奏乐系统。

本期我们介绍一个让单片机变成电子琴的程序。硬件装置仍采用KDC－Ⅲ开发机，利用其按键进行演奏。该电子琴框图如图1所示。

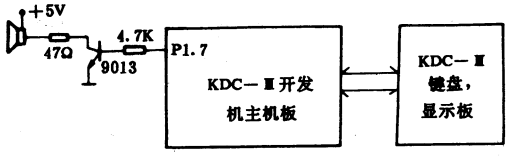


图1

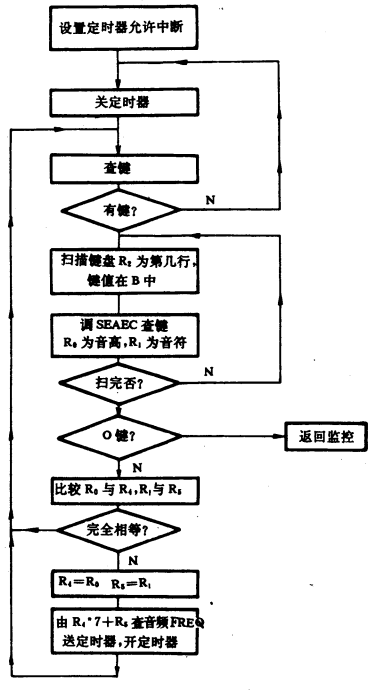


图2

键盘译码表和功能表如下：

0 C3B2	1 C0B3	2 C0B4	3 C3B4
4 C4B3	5 C1B3	6 C1B4	7 C4B4
8 C5B3	9 C2B3	A C2B4	B C5B4

键盘上1～7代表音符1～7。同时按下8(或9)和1～7键表示高音。同时按下A(或B)和1～7键表示低音。按键0则退出本程序。

电子琴程序设计思路是：查键盘，根据键值表的频率送定时器，并由8031的P1.7口输出去驱动喇叭发音。程序流程图如图2所示。

首先检查键盘。如有键，先延时1ms，逐行扫描，并调用SEARC子程序将键值转换成音符键(1～7)或者高键(低、中、高)。然后判断所得键值与原先键值是否相等。相等即按键不变，继续查键盘。不相等，查音频表FREQ，设置新的定时常数，产生新的声音。继续查键盘直至按下0键。

电子琴程序清单：

```
1          ORG 2000H
2          MOV IE,#82H          ;允许中断
3          ANL 89H,#0F0H        ;置定时器为工作方式1
4          ORL 89H,#01H
5          AJMP START
6          ORG 200BH            ;中断服务程序
7          CLR TR0
8          CPL P1.7             ;取反，产生方波
9          MOV TH0,R6
10         MOV TL0,R7
11         SETB TR0
12         RETI
13  START: CLR TR0
14  KEYBD: MOV DPTR,#4100H      ;置8155为：
15         MOV A,#0DH           ;C口输出
16         MOVX @DPTR,A         ;B口输出
17         MOV A,#1BH           ;查键盘
18         MOV DPTR,#4103H
19         MOVX @DPTR,A
20         MOV DPTR,#4102H
21         MOVX A,@DPTR
22         CPL A
23         ANL A,#18H
24         JZ START             ;无键转START
25         MOV R5,#0EH          ;有键，延时1ms
26  DE-
    LAY:   ACALL DIMS
27         DJNZ R5,DELAY
```

```
28          MOV R3,#01H
29          MOV R0,#00h
30  KSCAN:  MOV A,R3            ;逐行扫描键盘
31          MOV DPTR, #4103H
32          MOVX @DPTR,A
33          MOV DPTR, #4102H
34          MOVX A,@DPTR
35          CPL A
36          ANL A,#18H
37          JZ NOKEY            ;扫描行无键
38          RL A                ;有键,键值在 B
39          SWAP A
40          MOV B,A
41          MOV A,R3
42          MOV R2,#01H
43  DD:     CLR C
44          RRC A
45          JZ NPP
46          INC R2
47          AJMP DD
48  NPP:    MOV A,R2
49          SWAP A
50          ADD A,B
51          MOV R2,A
52          ACALL SEARC         ;调用 SEARC 查键
53  NOKEY:  MOV A,R3            ;扫描行无键处理,
54          RL A
55          MOV R3,A
56          CJNE R3, # 40H, KSCAN   ;未完扫下行
57          MOV A,R1
58          JZ EXIT             ;0 键转 STT
59          XCH A,R5            ;与原键比较
60          XRL A,R5
61          JZ UNCH1            ;即比较:R5—R1
62          MOV A,R0            ;  R4—R0
63          MOV R4,A
64          AJMP CHAN           ;变转 CHAN
65  EXIT:   LJMP 0000H          ;0 键退出程序
66  UNCH:   AJMP KEYBD
67  UNCH1:  MOV A,R0
68          XCH A,R4
69          XRL A,R4
70          JZ UNCH             ;键未变继续查键
71  CHAN:   MOV A,R4            ;键改变重置时间常数
72          MOV B,#07H
73          MUL AB
74          ADD A,R5
75          DEC A
76          RL A
77          MOV DPTR,#FREQ
78          MOV R2,A
79          MOVC A,@A+DPTR
80          MOV R6,A
81          MOV A,R2
82          INC A
83          MOVC A,@A+DPTR
84          MOV R7,A
85          MOV TH0,R6
86          MOV TL0,R7
87          SETB TR0
88          AJMP KEYBD          ;继续扫键直至 0 键
89  DIMS:   MOV R2,#64H         ;延时子程序
90  DLAY:   NOP
91          NOP
92          NOP
93          DJNZ R2,DLAY
94          RET
95  SEARC:  MOV B,#00H          ;查键值子程序,
96          MOV DPTR,#KEYBL     ;R1 为音符(1~7)
97  SEA:    MOV A,#00H          ;R0 为音高,
98          MOVC A,@A+DPTR      ;R0=0,1,2
99          XRL A,R2            ;表示:中音,低音,高音
100         JZ YES
101         INC B
102         INC DPTR
103         AJMP SEA
104 YES:    CLR C
105         MOV A,B
106         SUBB A,#08H
107         JNC LOW
108         MOV R1,B
109         RET
110 LOW:    MOV A,B
111         SUBB A,#0AH
112         JNC HIGH
113         MOV R0,#01H
```

(转至 24 页)

MCS—51 单片机在听力播放、发射系统中的应用

天津纺织工学院　沈乃才　宋振环

目前，国内各大专学校为了提高学生的外语水平，先后兴建了许多语言实验室、听力室等专用教室。但都因其投资费用高，能建成的教室数量有限，远远不能满足广大学生平时外语学习的要求。尤其是大量的听力练习。为此，我们先后采用 TP801 单板机和 MCS-51 单片机对有线发射、无线接收方式的听力播放、发射系统进行了全天候实时控制。较好地解决了上述问题，减轻了外语教师、管理人员的困难，取得较好的效果。

一、系统构成

我院语音听力，每天定时播放四次，每次一小时。因此，该系统要求能在规定的时间内准时完成：通电→放音→停放→倒带→放音→停放→倒带→断电等操作。以期达到系统无人值班、自动定时播出节目。

整个系统以 8031 为 CPU，系统硬件如图 1 所示。配置 2KB 程序存储器(2716)。系统还配置了 6 位 LED 数字显示器用以显示时间值，12 个键的键盘。共用一片 8155 可编程 I/O 接口芯片作输入/输出接口。8155 的 B 口控制显示器的阴极电位。称之为扫描器。8155 的 A 口控制 LED 显示器的各段来显示字形，称之为字型口，以逐位扫描工作方式，实现显示器的动态显示。播放系统的全部控制信号分别由 8031 的端口 P1 输出，经光电隔离和功率放大后，驱动发射系统的各相应继电器动作。接通或切断被控系统中相应的功能开关，从而达到自动定时控制的目的。第一次通电开机后的初值时间由键盘键入。全部控制程序固化于 2716 之中。

二、软件

系统软件由主程序、中断服务程序、显示子程序和键入子程序组成。

1. 主程序

主程序流程图如图 2 所示，完成工作单元初始化、清显示缓冲区。设定 8031 定时器 0 的工作方式、中断方式、启动定时器 0、显示时间并判定预定时间是否到?若时间到，则发出相应的控制信号，启动被控对象进行工作。

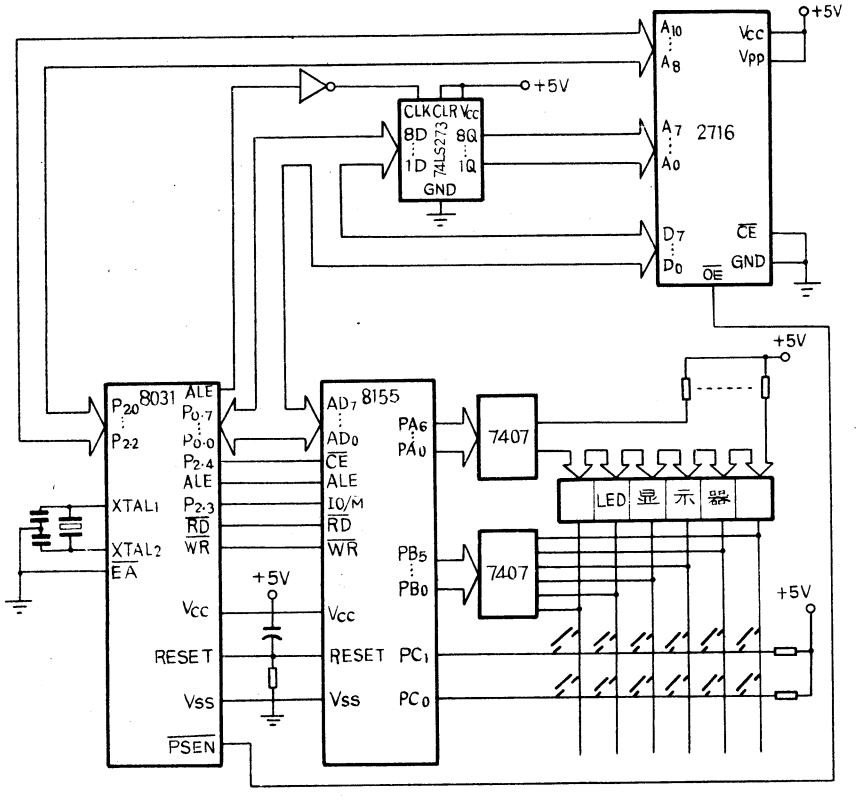


图 1　系统硬件电路图

2. 中断服务程序

中断服务程序流程图如图 3 所示。

定时器 T0 工作在方式 1，每隔 100ms 请求中断一次，每执行一次中断服务程序，溢出次数计数器单元初值减 1，判其减 1 后是否为零，如不为零，恢复现场，从中断返回。减足 10 次后为零。秒的个位值增 1。如此反复，并按秒、分、时的顺序逐位判别，按正常时间规律进行进位或清零，然后中断返回。

3. 键入子程序

为了在开机时，使其显示的时间与标准时间相吻合，必须进行标

准时间值的输入。本系统采用程控键扫描方式。键输入子程序流程图如图 4 所示。程序首先对 8155 初始化，继而判别键盘有无键闭合，利用二次调用显示子程序延迟 12ms 消除键抖动。为了保证键的一次闭合只作一次处理，必须等待输入键释放后才对输入键进行处理。最后将键号值送往 A 中。

4. 显示子程序

显示子程序流程图如图 5 所示。显示数据缓冲区

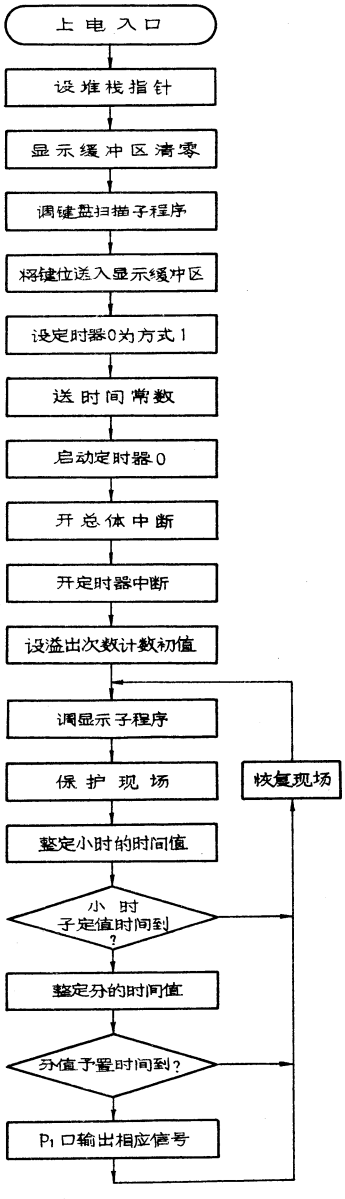


图 2

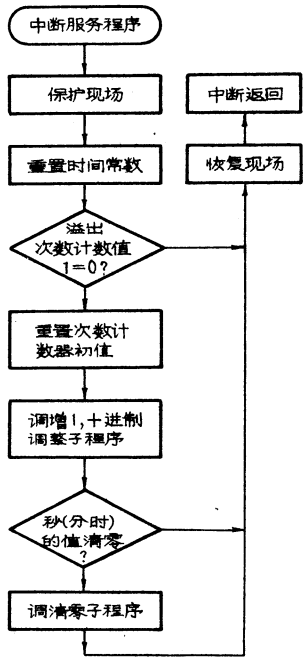


图 3

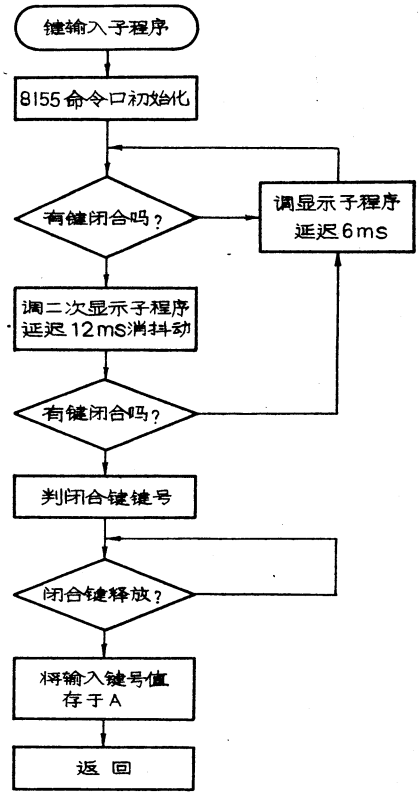


图 4

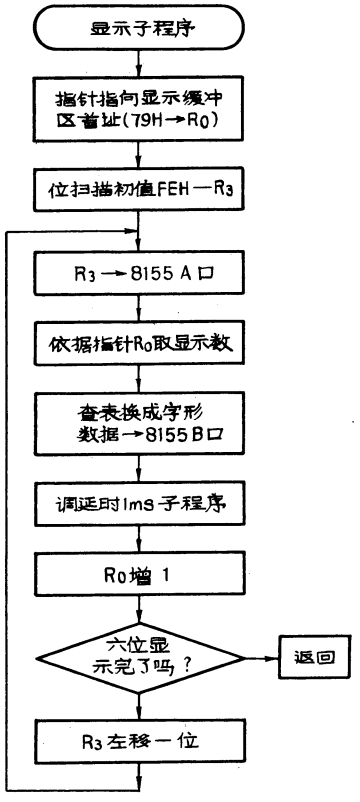


图5　显示子程序流程图

为 79H～7EH。主程序对 8155 的初始化之后，将欲显示的数据写入 79H～7EH。循环调用此显示子程序，显示器将时间值显示出来。直到各位都扫描一次，返回主程序。

该系统在我院已运行三年之久。实践证明，运行效果良好，工作可靠。不仅保证了学生有充分的课外听力练习时间，而且还节省了外语教师的大量课时。本系统于 1988 年底也通过了有关专家的技术鉴定。

本系统亦可在原有基础上稍加修改，成为各种实时控制器使用。

程序清单：

```
A>type ss.asm
            ORG 8000H
START:  AJMP MAIN
            ORG 800BH
        AJMP PIT0
            ORG 8010H
MAIN:   MOV SP,#60H
        MOV R0,#79H
        MOV R7,#06H
ML1:    MOV @R0,#00H
        INC R0
        DJNZ R7,ML1
        MOV R0,#7EH
        MOV R7,#06H
ML2:    PUSH PSW
        SETB PSW.3
        LCALL 078DH
        POP PSW
        MOV @R0,A
        DEC R0
        DJNZ R7,ML2
        MOV TMOD,#01H
        MOV TL0,#B8H
        MOV TH0,#3CH
        SETB 8CH
        SETB AFH
        SETB 0ASH
        MOV 30H,#0AH
ML0:    LCALL 0745H
        PUSH PSW
        PUSH ACC
        MOV A, 7EH
        SWAP A
        ORL A,7DH
        CJNE A,#06,NEXT1
        SJMP F1
NEXT1:  CJNE A, #07, NEXT2
        SJMP F2
NEXT2:  CJNE A,#15,NEXT3
        LJMP F3
NEXT3:  CJNE A,#16,NEXT4
        LJMP F4
NEXT4:  CJNE A,#18,NEXT5
        LJMP F5
NEXT5:  CJNE A,#19,NEXT6
        LJMP F6
NEXT6:  CJNE A,#20,NEXT7
        LJMP F7
NEXT7:  CJNE A,#21,HFXC
        LJMP F8
HFXC:   POP ACC
        POP PSW
        LJMP ML0
F1:     MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#7EH
        CJNE A,#44,NEXTA1
        LJMP WORK
NEXTA1: MOV R5,#0FCH
        CJNE A,#45,NEXTA2
        LJMP WORK
NEXTA2: MOV R5,#FEH
        CJNE A,#46,NEXTA3
        LJMP WORK
NEXTA3: LJMP HFXC
```

```
F2:     MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#FAH
        CJNE A,#17,NEXTB1
        LJMP WORK
NEXTB1: MOV R5,#FCH
        CJNE A,#18,NEXTB2
        LJMP WORK
NEXTB2: MOV R5,#FEH
        CJNE A,#19,NEXTB3
        LJMP WORK
NEXTB3: MOV R5,#FAH
        CJNE A,#50,NEXTB4
        LJMP WORK
NEXTB4: MOV R5,#FFH
        CJNE A,#51,NEXTB5
        LJMP WORK
NEXTB5: LJMP HFXC
F3:     MOV A,7CH
        MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#FEH
        CJNE A,#49,NEXTC1
        LJMP WORK
NEXTC1: MOV R5,#FCH
        CJNE A,#50,NEXTC2
        LJMP WORK
NEXTC2: MOV R5,#FEH
        CJNE A,#51,NEXTC3
        LJMP WORK
NEXTC3: LJMP HFXC
F4:     MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#FAH
        CJNE A,#22,NEXTD1
        LJMP WORK
NEXTD1: MOV R5,#FCH
        CJNE A,#23,NEXTD2
        LJMP WORK
NEXTD2: MOV R5,#FEH
        CJNE A,#24,NEXTD3
        LJMP WORK
NEXTD3: MOV R5,#FAH
        CJNE A,#55,NEXTD4
        LJMP WORK
NEXTD4: MOV R5,#FFH
        CJNE A,#56,NEXTD5
        LJMP WORK
NEXTD5: LJMP HEXC
F5:     MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#FEH
        CJNE A,#09,NEXTE1
        LJMP WORK
NEXTE1: MOV R5,#FCH
        CJNE A,#10,NEXTE2
        LJMP WORK
NEXTE2: MOV R5,#FEH
        CJNE A,#11,NEXTE3
        LJMP WORK
NEXTE3: MOV R5,#FAH
        CJNE A,#42,NEXTE4
        LJMP WORK
NEXTE4: MOV R5,#FCH
        CJNE A,#43,NEXTE5
        LJMP WORK
NEXTE5: MOV R5,#FEH
        CJNE A,#44,NEXTE6
        LJMP WORK
NEXTE6: LJMP HFXC
F6:     MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#FAH
        CJNE A,#15,NEXTF1
        LJMP WORK
NEXTF1: MOV R5,#FFH
        CJNE A,#16,NEXTF2
        LJMP WORK
NEXTF2: LJMP HFXC
F7:     MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#FEH
        CJNE A,#00,NEXTG1
        LJMP WORK
NEXTG1: MOV R5,#FCH
        CJNE A,#01,NEXTG2
        LJMP WORK
NEXTG2: MOV R5,#FEH
        CJNE A,#02,NEXTG3
        LJMP WORK
NEXTG3: MOV R5,#FAH
        CJNE A,#33,NEXTG4
        LJMP WORK
NEXTG4: MOV R5,#FCH
        CJNE A,#34,NEXTG5
        LJMP WORK
NEXTG5: MOV R5,#FEH
        CJNE A,#35,NEXTG6
        LJMP WORK
NEXTG6: LJMP HFXC
```

```
F8:     MOV A,7CH
        SWAP A
        ORL A,7BH
        MOV R5,#FAH
        CJNE A,#06,NEXTH1
        LJMP WORK
NEXTH1: MOV R5,#FFH
        CJNE A,#07,NEXTH2
        LJMP WORK
NEXTH2: LJMP HFXC
WORK:   MOV 90H,R5
        LJMP HFXC
PIT0:   PUSH PSW
        PUSH ACC
        SETB PSW,3
        MOV TH0,#3CH
        MOV TL0,#B8H
        MOV A,30H
        DEC A
        MOV 30H,A
        JNZ RET0
        MOV 30H,#0AH
        MOV R0,#7AH
        ACALL DAAD1
        MOV A,R2
        XRL A,#60H
        JNZ RET0
        ACALL CLR0
        MOV R0,#7CH
        ACALL DAAD1
        MOV A,R2
        XRL A,#60H
        JNZ RET0
        ACALL CLR0
        MOV R0,#7EH
        ACALL DAAD1
        MOV A,R2
        XRL A,#24H
        JNZ RET0
        ACALL CLR0
RET0:   POP ACC
        POP PSW
        RETI
DAAD1:  MOV A,@R0
        DEC R0
        SWAP A
        ORL A,@R0
        ADD A,#01H
        DA A
        MOV R2,A
        ANL A,#0FH
        MOV @R0,A
        MOV A,R2
        INC R0
        ANL A,#F0H
        SWAP A
        MOV @R0,A
        RET
CLR0:   CLR A
        MOV @R0,A
        DEC R0
        MOV @R0,A
        RET
```

(上接 47 页)

```
10 FOR I=1 TO 5
20 GOSUB 100
30 PRINT S
40 NEXT I
100 FOR I=1 TO 20
110 S=0
120 INPUT B(I)
130 S=S+B(I)
140 NEXT I
150 RETURN
```

答:(1)缺少数组说明语句:5 DIM B(20)

(2)主程序和子程序使用了同一个变量,应把 10 行和 40 行变量 I 改成 K。

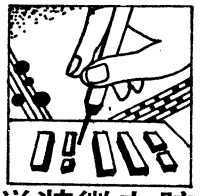
(3)第 10 行和第 100 行的循环终值应调换。

(4)语句 S=0 应放在循环外;即改为:

```
100 S=0
110 FOR I=1 TO 5
```

正确的程序如下:

```
5 DIM B(20)
10 FOR K=1 TO 20
20 GOSUB 100
30 PRINT S
40 NEXT K
50 END
100 S=0
110 FOR I=1 TO 5
120 INPUT B(I)
130 S=S+B(I)
140 NEXT I
150 RETURN
```



用微电脑检查印刷电路板

易齐干

怎样使用微电脑自动检查电子线路板的焊接情况，是保证电路板质量的重要手段。现在，使用 μp－80 教育部件，介绍检查诊断印刷线路工作状态是否正常的基本方法。被检的电子线路板是以基础篇中的反相器与锁存器板为例。

1. 检查反相器板

(1)反相器板的印刷线路图及电源理如图 1 所示。

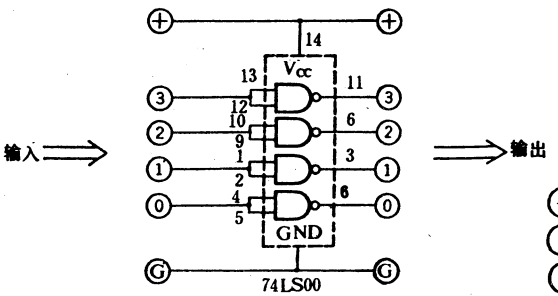


图 1

使用 μp－80 教育部件检查的反相器 74LS00 板，如图 2 所示。

74LS00 为“四与门”电路，因将每个与门输入端合二为一，故实为四反相器功能。

(2)μp－80 输入输出板上各口的分配如图 3 所示。

端口 A、B 与 C 高位为输出口，端口 C 的低位为输入口。

①端口 C 的高位向反相器输出检查数据。

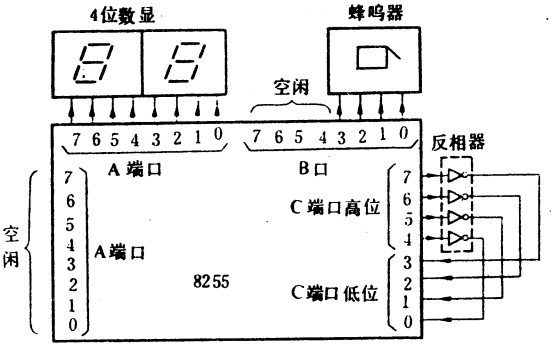


图 3

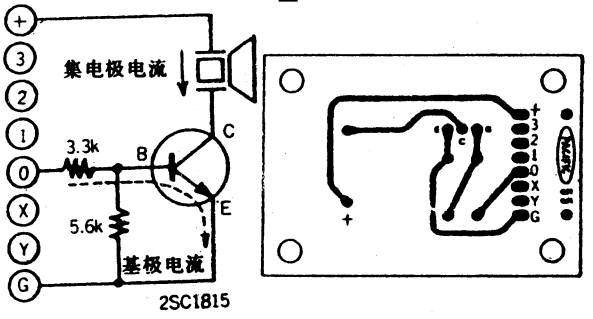


图 4

②反相器反转后的数据输入给端口 C 的低位。

③端口 A 输出检查结果进行 2 位数字显示。

当显示 FF，检查结果为正常；当显示其它数据，检查结果为不正常。与端口 B 相连的蜂鸣器发出蜂鸣，压电蜂鸣器的印刷板图及原理图如图 4 所示

(3)检查反相器板的流程图如图 5 所示。

由 00E0H 地址开始，顺序地设置并存储用于检查的输出数据和验证数据。如表 1 和图 6 所示。

①反相器由第 0 位到第 3 位依次输出 1(高电平)为输出数据。(即：10，20，40，80)。

②预先设置的验证数据与反相器反转后的数据进行比较，看其是否一致。

同样，可以求出第 1 位到第 3 位相对于输出数据的验证数据。另外，00E8H 地址数据 FFH 是检查结束数据。

参考上述内容，编制成的反相器部件检查程序如

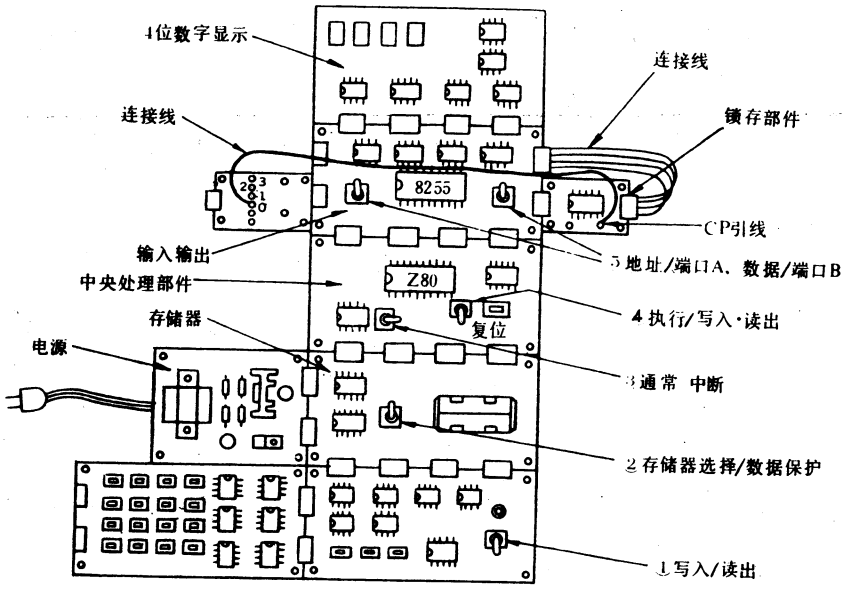


图 2

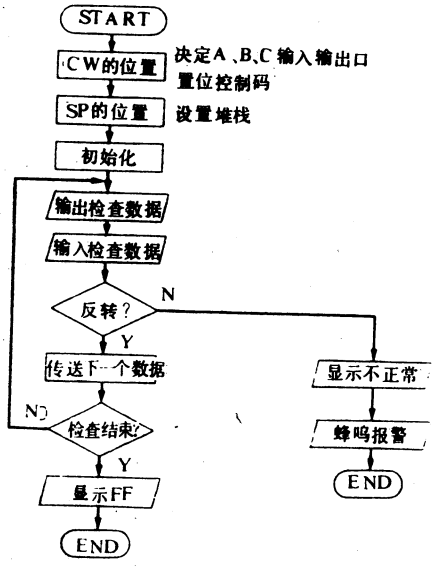


图 5

表 1

地址	数据	注释
E0	10	第0位的输出数据
E1	1E	第0位的验证数据
E2	20	第1位的输出数据
E3	21	第1位的验证数据
E4	40	第2位的输出数据
E5	4B	第2位的验证数据
E6	80	第3位的输出数据
E7	87	第3位的验证数据
E8	FF	存入FF为检查结束数据

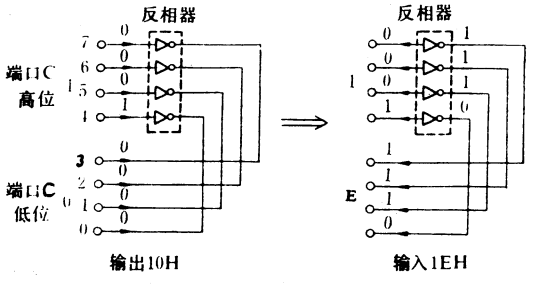


图 6

表 3

部件名称	开关No	开关方向
写入·读出控制	1	读出侧
存储器	2	存储器选择侧
中央处理	3	通常侧
	4	写入·读出侧
输入输出	5	端口A、端口B侧

表 4

显示	原因
10	第0位断线
20	第1位断线
40	第2位断线
80	第3位断线

表 2 所示。

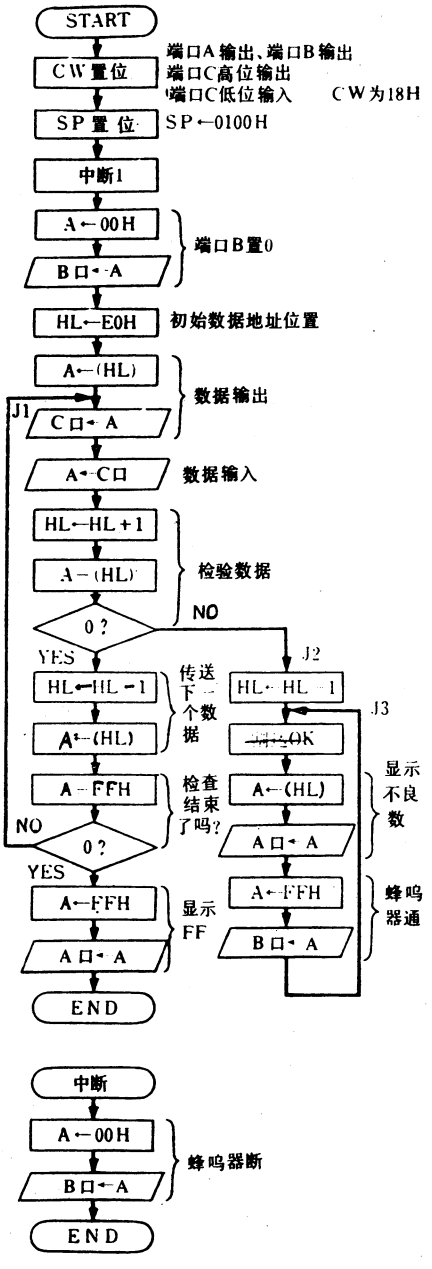
(4)检查方法

①表 2 所示的检查程序写入完毕，并进行检验之后，照图 2 连接各部件。

②各开关的位置与表 3 一致。

③按动中央处理部件的复位开关，其时间为 1 秒

表 2



标号	助记符	地址	机器语言	注释
START	LD A, 81H	00	3E 81	CW置位
	OUT(03H), A	02	D3 03	
	LD SP, 0100H	04	31 00 01	SP置位
	IMI	07	ED 56	中断
	LD A, 00H	09	3E 00	B口置于0
	OUT(01H), A	0B	D3 01	
	LD HL, E0H	0D	21 E0 00	初始数据地址置位
	LD A, (HL)	10	7E	数据输出
J1	OUT(02), A	11	D3 02	
	IN A, (02H)	13	DB 02	数据输入
	INC HL	15	23	检验数据
	CP (HL)	16	BE	
	JP NZ, J2	17	C2 26 00	
	INC HL	1A	23	传送下一数据
	LD A, (HL)	1B	7E	
	CP FFH	1C	FE FF	检查结束了吗?
	JP NZ, J1	1E	C2 11 00	
	LD A, FFH	21	3E FF	显示FF
	OUT(00H), A	23	D3 00	
	HALT	25	76	
J2	DEC HL	26	2B	
J3	EI	27	FB	可以中断
	LD A, (HL)	28	7E	显示不良数
	OUT(00H), A	29	D3 00	
	LD A, FFH	2B	3E FF	蜂鸣器通
	OUT(01H), A	2D	D3 01	
	JP J3	2F	C3 27 00	
中断	LD A, 00H	38	3E 00	用中断，蜂鸣器断
	OUT(01H), A	3A	D3 01	
	HALT	3C	76	
DATA		E0	10	检验第0位
		E1	1E	
		E2	20	检验第1位
		E3	2D	
		E4	40	检验第2位
		E5	4B	
		E6	80	检验第3位
		E7	87	
		E8	FF	检验结束

钟左右，将处于写入读出侧的开关④搬至实行侧，于是，4 位数字显示部件的右 2 位显示 FF 或其它数据。

若显示 FF，表示反相器合格。若为其它数据，说明不合格。此时，与端口 B 连接的蜂鸣器发出蜂鸣。若参照表 4 的检验表，则很容易查出故障所在。μp－80 中央处理部件的开关③搬向中断侧，则中止蜂鸣。

2. 检查锁存器板

(1)锁存器电路如图 7 所示。

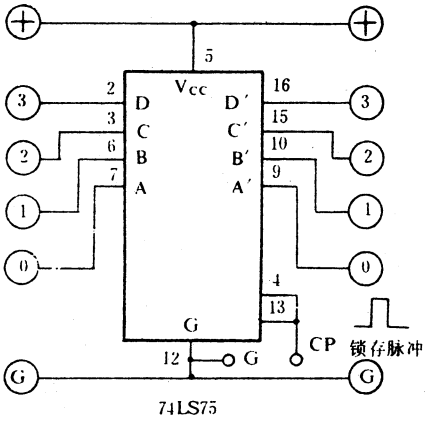


图 7

当锁存器板输入信息时，按动某个按键锁存住这一变化的信息，并使之不变。，74LS75 具有这种锁存功能，见图 7。

实际上是将 Ω 型锁存脉冲加在 CP 脚上，(为高电平时)74LS75 的输入数据将在输出端保持不变。当 CP 为低电平时，即使输入数据变化，输出数据也不变化。

(2)使用 μp－80 教育部件检查锁存器板的连接图如图 8 所示。

μp－80 中输入输出的分配如图 9 所示。端口 A、B、C 的低位为输出口。端口 C 的高位为输入口。

①端口 C 的低位向锁存器板输出检查数据。

②向端口 C 的高位输入锁存的数据。

③由端口 B 向显示 4 位数字部件输出检查结果。当显示 FF，检查结果为正常。显示其它数据，检查结果不正常。

④由端口 A 向锁存部件的 CP 输出锁存数据，并由显示 4 位数字部件显示该数据。

(3)检查锁存部件的板的流程图如图 10 所示。简要说明如下。

①由端口 C 低位输出锁存的数据。同时将该数据保存在 CPU 的 C 寄存器中。

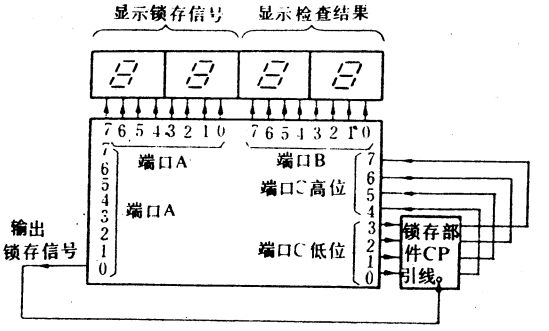


图 9

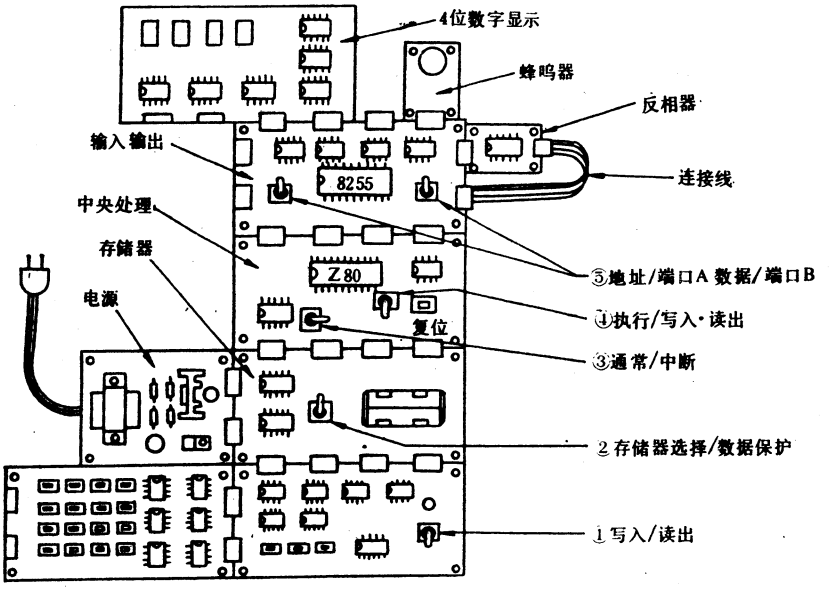


图 8

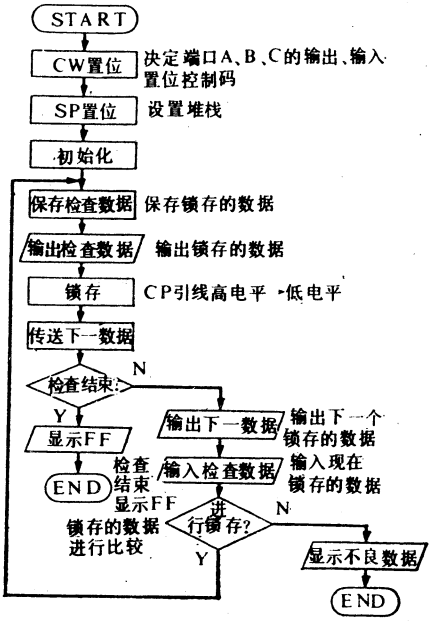


图 10

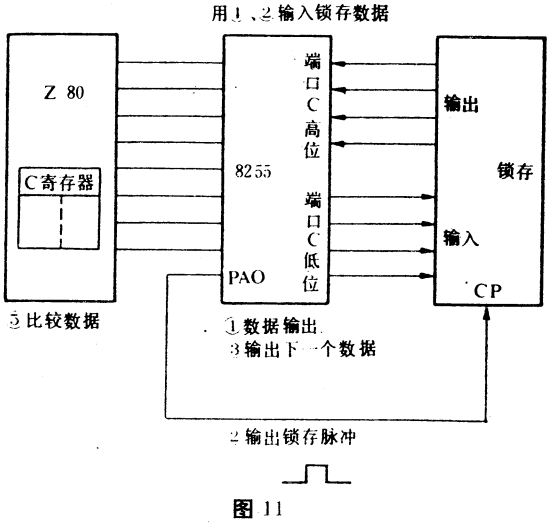


图 11

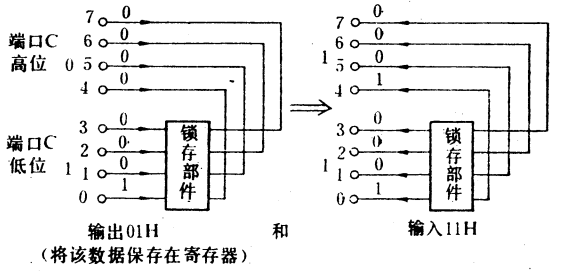


图 12

②端口 A 第 0 位的输出由高电平变为低电平时，控制 LS75 锁存。

③由端口 C 的低位输出下一个锁存的数据。

④将②锁存的数据，从端口 C 的高位输入。

⑤μp－80 比较由④输入的数据和由①保持在 C 寄存器中的数据。此时，若锁存，①输出的数据，则与 C 寄存器的数据一致。若没锁存，则与 C 寄存器的数据不一致。其原因是原封不动地输入③所输出的数据，或是通过焊锡点的情况变化输入了其它的数据。上述关系如图 11 所示。

⑥数据比较结果若一致，则进行下一位；若不一致，仍显示这一位。

由端口 C 输入的数据与 C 寄存器中原来的数据进行比较，如图 10 所示。当数据不一致时，可按下述方法用移位法变换输入数据，再和 C 寄存器的数据进行比较。

		端口C高位				端口C低位			
①端口C输入数据(11H)		0	0	0	1	0	0	0	1
		X	X	X	X	X	X	X	X
②AND F0H		1	1	1	1	0	0	0	0
③向右移位4次	①	0	0	0	0	1	0	0	0
	②			0	0	0	1	0	0
	③	0	0	0	0	0	0	1	0
	④	0	0	0	0	0	0		
		‖	‖	‖	‖	‖	‖	‖	‖
		0	0	0	0	0	0	0	1

④与 C 寄存器的数据(01H)进行比较

同样方法由第 1 位到第 3 位变换端口 C 输入数据，与 C 寄存器数据进行比较为合适。

参考上述内容编制的锁存部件检查程序，如表 5 所示。

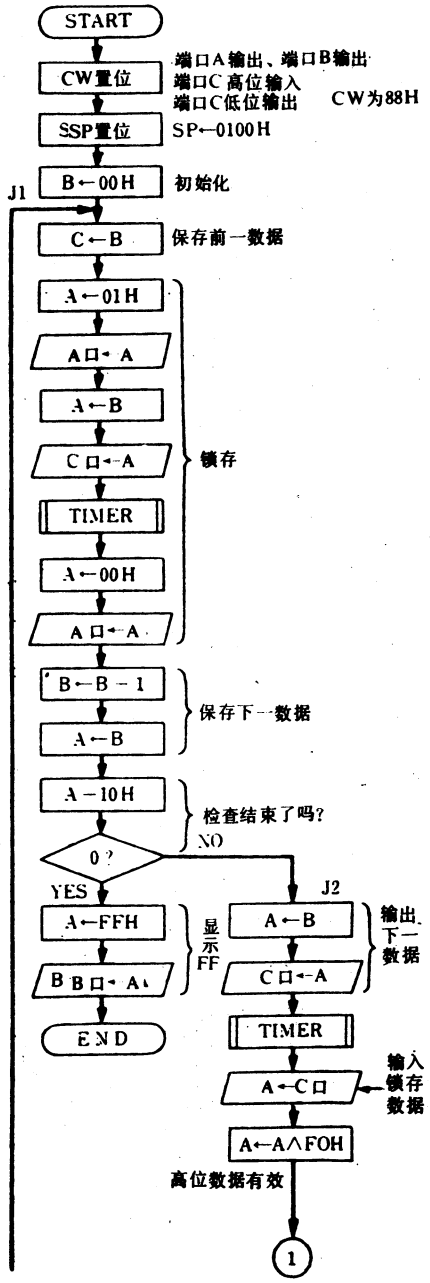
表 6

部件名称	开关N。	开关方向
写入·读出控制	1	读出侧
存储器	2	存储器选择侧
中央处理	3	通常侧
	4	写入·读出侧
输入输出	5	端口A 端口B侧

表 7

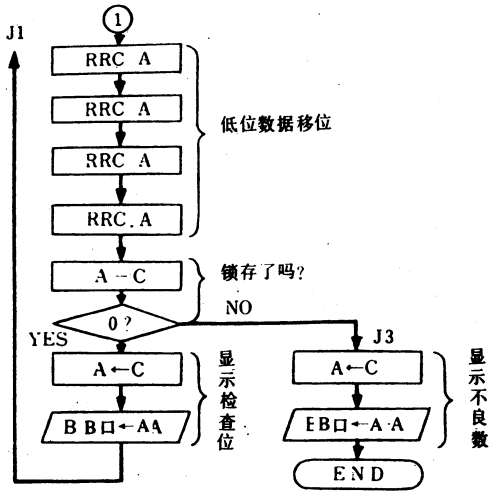
显示	原因
1 0	第0位断线，
2 0	第1位断线，
4 0	第2位断线，
8 0	第3位断线，

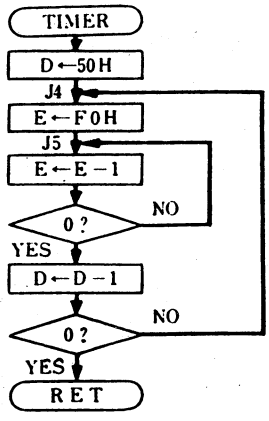
表 5



标号	助记符	地址	机器语言	注释
START	LD A, 88H	00	3E 88	CW置位
	OUT(03H), A	02	D3 03	
	LD SP, 0100H	04	31 00 01	设置堆栈SP
	LD B, 00H	07	06 00	初始化
J1	LD C, B	09	48	保存前一数据
	LD A, 01H	0A	3E 01	
	OUT(00H), A	0C	D3 00	
	LD A, B	0E	78	进行锁存
	OUT(02H), A	0F	D3 02	
	CALL TIMER	11	CD 44 00	
	LD A, 00H	14	3E 00	
	OUT(00H), A	16	D3 00	
	INC B	18	04	保存下一数据
	LD A, B	19	78	
	CP 10H	1A	FE 10	检查结束了吗?
	JP NZ, J2	1C	C2 24 00	
	LD A, FFH	1F	3E FF	检查结束
	OUT(01H), A	21	D3 01	显示FF
	HALT	23	76	
J2	LD A, B	24	78	输出下一数据
	OUT(02H), A	25	D3 02	
	CALL TIMER	27	CD 44 00	
	IN A,(02H)	2A	DB 02	输入锁存数据
	AND F0H	2C	E6 F0	高位数据有效
	RRC A	2E	CB 0F	数据在低位移位
	RRC A	30	CB 0F	
	RRC A	32	CB 0F	
	RRC A	34	CB 0F	
	CP C	36	B9	锁存了吗?
	JP NZ, J3	37	C2 40 00	
	LD A, C	3A	79	显示检查位
	OUT(01H), A	3B	D3 01	
	JP J1	3D	C3 09 00	
J3	LD A, C	40	79	显示不良位
	OUT(01H), A	41	D3 01	
	HALT	43	76	
TIMER	LD D, H	44	16 50	
J4	LD E, H	46	1E F0	
J5	DEC E	48	1D	
	JP NZ, J5	49	C2 48 00	
	DEC D	4C	15	
	JP NZ, J4	4D	C2 46 00	
	RET	50	C9	

接表 5





⑦ 检查方法

(1)照图 11 连接各部件。写入表 5 所示的检查程序,并进行检验。

(2)各部件的开关位置与表 6 所示一致。

(3)按动中央处理部件的复位开关,时间为 1 秒左右,将写入、读出一侧的开关④搬至实行侧。于是,显示 4 位数字部件的右边 2 位由 00 依次变化到 0F。最后,若显示 FF,说明锁存器板合格;若显示其它数据,则说明不合格。参考表 7 检验图表,能较容易找出故障原因。

该程序是,用延时方法延长各位的检查时间,如果将程序中"CALL TIMER"语句处,换成执行 3 次"NOP"(00H)指令时,能加快检查速度。

以上讨论了反相器板和锁存器板的检查系统,这仅是用微电脑自动检查的基本事例。用微电脑检查复杂电路板的原理是一样的,最关键的问题是要理解被检查部件的功能、工作原理、并编制出程序。

(上接第 4 页)品,诸如食品、音像磁带、出版物、金属制品以及其它许多物品。它还通常用于包装、销售计账和数据处理等方面。

EAN 码(European Artical Number):它是一种世界各国广泛采用的通用物品条形码。它的符号表示法和用途与 UPC 相似,且相互兼容。

3. 39 码:它是一种包含字母、数字和专用符号的通用条形码。主要用于文献图书、工业生产线、仓贮、货运、票证、邮政及医院等领域。

4. 25 码:是一种只包含数字的通用条形码。主要用于各种类型的文件、货运及工业生产线。

二、条形码系统的工作原理

条形码系统的基本组成部分包括:条形码扫描器、译码器、微型计算机这三大部分。

下面我们以经常使用的光笔为例,来介绍一下条形码系统的工作原理。

光笔作为扫描器的一种,是由发光二极管来产生光源的,经一球型兰宝石透镜聚焦在条形码表面上,其反射光线强弱与条形码中条和空的颜色深浅、宽窄大小成对应关系。反射光线被光笔中的光每元件接收并转换成模拟电信号,经放大电路放大后,送至比较器,由比较器产生相应的窄脉冲输出,比窄脉冲信号送到译码器。而译码器是由微处理器及其它控制芯片组成,其主要功能是将条形码读入设备送来的条形码电信号进行译码,变成计算机可识别的 ASCII 码,这样就可将 ASCII 码直接通过计算机键盘的输出口送至计算机,利用已编好的计算机应用程序,完成对数据的录入、统计、管理、控制等功能。必要时,计算机将发出相应的控制信号,经数模转换→放大驱动后,控制机械手操作相应的机械,从而达到生产过程自动控制的目的。

综上所述,条形码技术是在计算机应用的实践中应运而生的一种非常适用的数据自动采焦技术。近年来,它在实践中不断得到完善和发展。为了加速我国的对外经济贸易、促进我国经济的繁荣和发展,我们要大力推广使用条形码技术。

初级程序员级水平考试辅导讲座
BASIC 语言程序设计问答

李志刚　吕品

问题 1　什么是 BASIC 语言？

答：1965 年，由美国 Dartmonth 学院的 JohnG・Kemeny 和 Thomas E・Kurtz 两位教授以 FORTRAN 语言为蓝本，设计出适合于计算机程序编制人员使用的一种高级程序设计语言，这就是 BASIC 语言。它是 Beginner's ALL－purpose Symbolic Instruction Code（初学者通用符号指令代码）的缩写。在我国，BASIC 语言已是各类学习机、PC 机乃至高档微机上的宿主语言。由于 BASIC 语言具有易学易用的特点，已受到广大计算机用户的重视，并用它编制出涉及各个领域里大量的优秀应用软件，在计算机应用中发挥了重要作用。

问题 2　BASIC 语言有什么主要特点？

答：BASIC 语言是一种通用的面向算法的高级程序设计语言。它是一种适合于初学者使用而又实用的语言。BASIC 语言有以下几个主要特征：

1. BASIC 语言好学易懂。基本 BASIC 一共只有 17 种语句。基本 BASIC 命令和语句中使用的词以及运算符号与英语中使用的词以及数学中的符号差不多，因此比较直观、易于理解和记忆。

2. BASIC 语言是会话式语言。当 BASIC 命令和程序输入计算机后，计算机立即进行语法检查，并在出错时给出提示。用户可在计算机上边运行边修改，直至程序正确为止。

3. 一般 BASIC 语言有立即执行方式。即从键盘输入语句后，立即给出结果，给检查和调试程序带来很大方便。

4. BASIC 语言系统虽小但功能齐全。除科学计算外有一定的文字、图形和音响的处理能力。可用来编制中小规模的应用程序和进行各种辅助设计、辅助教学以及游戏软件的编制等。

5. BASIC 语言是小型计算机语言，它允许的数的范围、变量个数、数据类型、子程序调用等都有一定限制。大部分 BASIC 是解释型的，程序运行速度较慢。

问题 3　什么是 BASIC 程序？BASIC 程序的逻辑行与物理行是指什么？

答：一个 BASIC 程序是由若干个程序行组成。每个程序行由行号、语句组、行结束符三部分组成。BASIC 规定每个程序行中的最大字符个数（含空格）≤255。

行号是程序行的标志，写在每个程序行之首。BASIC 程序是按行号升序次序在内存储器中顺序存放，并按此顺序执行程序。行号不仅指示了程序的执行次序，同时也可由行号改变执行顺序。PC BASIC 中的行号允许使用 0～65529 之间的整数。在程序中行号可以不连续（间隔 10 为好），在输入程序时行号次序不限，删改某行时，其它行不受影响。

语句组是程序行的核心，它规定了计算机在执行该程序行时应做的操作。当语句组中只含一条语句时该行程序称为简单程序行，否则称为复合语句程序行。复合语句行中语句组内各语句之间用冒号分隔。每个语句由语句定义符和语句体两部分组成。语句定义符是指定该语句执行何种功能的动词，又称为 BASIC 的系统保留字，它位于语句体之前。语句体是该语句所执行操作的对象。语句定义符和语句体的书写格式由语言本身做出规定，BASIC 语言允许空语句体存在。

行结束符是通过打一个回车键来产生的。输入或编辑某一程序行结束时必须按回车键。在书写程序时一般可用"↙"表示回车键，通常可省略不写。

一个完整的程序一般由输入数据部分，计算处理部分，输出结果部分和结束部分组成。

物理行是指显示屏上的一行或打印机上的一行。逻辑行是指 BASIC 程序的一个完整程序行，它是由行号、语句组和行结束符构成。由于每个程序行中字符个数最大可达 255 个，因此一个逻辑行就可能对应一个或多个物理行。

问题 4　有几种提供数据的语句？它们的主要用途各是什么？

BASIC 语言中主要有三种提供数据的语句，即 LET、INPUT、和 READ 语句。

LET 语句主要用于表达式的计算及少数变量的赋值。

INPUT 语句主要用来在程序运行中从键盘给少量需改变值的参变量赋值。

READ 语句主要用于对较多变量的已知数据进行赋值。

问题 5　BASIC 语言中常用函数有几类？用户怎样引用？

答：BASIC 语言中的函数分为标准库函数和用户自定义函数两类。标准函数主要分为数值函数、字符串函数、输入输出函数及其它辅助函数。标准函数库是 BASIC 语言系统为完成特定的功能而提供的一组标准子程序。它们包含在 BASIC 语言中，并由该语言本身

规定每个标准函数的名字，用户在使用时只需写出它的函数名和自变量名就可以直接引用。在表达式中它们可象变量一样参与运算，但不能对其赋值。如要将 X＋Y 的平方根赋值给变量A，只要写出 LET A＝SQR(X＋Y)即可。

自定义函数是由用户为解决自己的特定问题而定义的非标准函数。自定义函数在程序中一经定义后，就同标准函数一样可在表达式中引用。

上述两类函数都可被程序多次调用，不同处在于：标准库函数独立于用户程序。自定义函数则包含在用户程序中，由用户自己编写和定义。

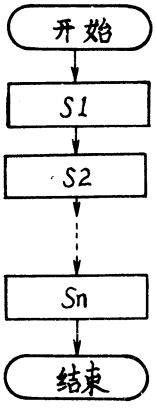
问题 6　BASIC 程序的基本控制结构是什么？由哪些语句实现？

答：BASIC 程序的基本控制结构有三种：顺序结构、选择结构和循环结构。

1. 顺序结构

顺序结构是程序的基本结构。也是最简单的结构。在此结构中，程序按行号排列的先后次序执行，其流程图如图一所示。

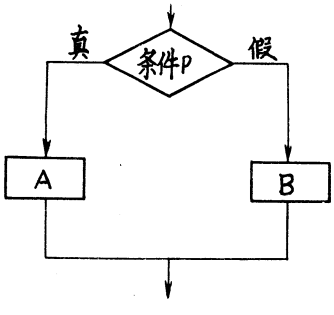
其中 S1，S2……Sn 为 n 个程序行。只要确定好各程序行执行的先后顺序就可较容易地完成程序设计。



图一、顺序结构

2. 选择结构

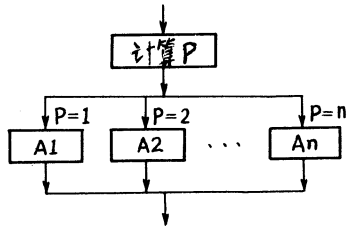
选择结构又称分支结构，它是按一定的条件在多个可能的计算或处理步骤中选取其中之一去执行。最简单的选择结构如图二所示。当条件 P 成立时(为真)，执行 A 框，否则执行 B 框。一般情况下，选择结构可以嵌套或为多路选择，如图三所示。



图二、选择结构

3、循环结构

循环结构又称重复结构。在程序设计中需要连续重复执行某一过程时，大多采用循环结构。常见的循环结构有三种类型：

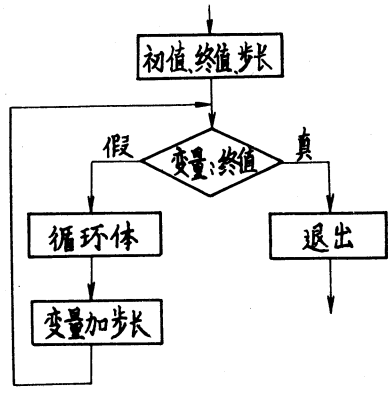


图三、多路选择

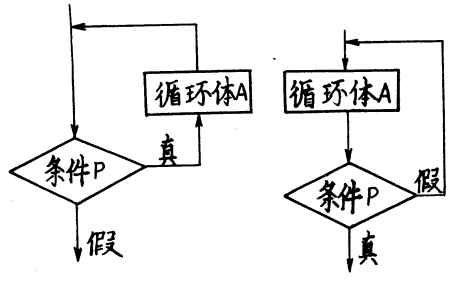
a、计数循环：可用 FOR/NEXT 语句实现，一般用于已知循环的次数、循环的初值、终值和步长的情形，其工作流程图如图四所示。

b、当型循环结构：可用 WHILE/WEND 语句实现，一般用于不知道循环次数的情况，只有在某个条件不满足时退出循环。其工作流程图如图五所示。

c、直到型循环结构：可用 IF/THEN 语句实现，一般也是用于不知道循环次数的情况，在条件满足时退出循环。其工作流程图如图六所示。



图四　计数循环



图五　当型循环　　图六　直到型循环

从流程图可看出，当型循环与直到型循环都是在一定条件下重复执行循环体 A，所不同的是，当型循环判断条件满足时，执行循环体 A，然后再次判断条件是否满足，以决定是继续执行循环体 A，还是退出循环。而直到型循环先执行一次循环体 A 再判断条件是否满足，以决定是重复执行循环体 A，还是退出循环。

通常这三种循环结构之间是可以互相转换的，例如用三种结构所编的求 $\sum_{n=1}^{100}\frac{1}{n(n+1)}$ BASIC 程序，其繁简程度不一。在程序设计中采用哪种循环结构，要视题目要求及已知条件而定，不能一概而论。

(1)计数循环

```
10  S=0
20  FOR K=1 TO 100
30  S=S+1/(K×K+K)
40  NEXT K
50  PRINT "S=";S
60  END
```

(2)当型循环

```
10  S=0:K=1
20  WHILE K<=100
30    S=S+1+/(K×K+K)
40    K=K+1
50  WEND
60  PRINT "S=";S
70  END
```

(3)直到型循环

```
10  S=0:K=1
20  S=S+1/(K×K+K)
30  K=K+1
40  IF K<=100 THEN 20
50  PRINT "S=";S
60  END
```

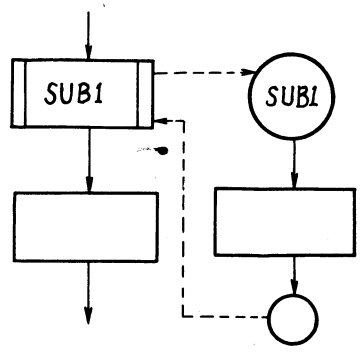
在 BASIC 中，由于上述三种结构都只有一个入口和一个出口，结构易于掌握，同时用它们又可以组成其它各种各样的程序结构，因此，熟练地掌握这三种基本结构，才能用 BASIC 语言编制出质量较高的程序。

4. 子程序结构

BASIC 还为程序设计者提供了子程序这一重要的程序结构，如图七所示。在程序设计中可将某些常用语句序列组合成一段程序，作一种特定的处理。该段程序就称为子程序段。如图中 SUB1 是子程序段。在主程序中需要用该特定处理功能时，就由主程序来调用，完成处理后又回到主程序调用处的下一条语句继续往下执行。从而大大地提高了程序编制的效率，减少了多余的程序段，提高了程序的可读性和可移植性。

问题 7　FOR/NEXT 与 WHILE/WEND 有什么区别？

答：FOR/NEXT 语句通常称为计数循环，而 WHILE/WEND 称为条件循环，也称当型循环。



图七　子程序结构

1. 按给定的次数重复执行一系列语句，这类循环称为计数循环。一般用于循环次数、循环控制变量的初值、终值和步长已知的情形。在 BASIC 语言中，计数循环一般是用 FOR/NEXT 语句来实现的。它的格式如下：

FOR 控制变量=X TO Y STEP Z
循环体
NEXT　控制变量

其中 X 是一个算术表达式，它产生循环控制变量的初值。Y 是算术表达式，是循环控制变量的终值。Z 是做控制变量增量的算术表达式，也叫步长。

这种递增(减)计数循环，通常是达到某一个值时终止。如果 Z 的值是正(负)的，每执行一次循环体，控制变量值增(减)一个步长，直至大于(小于)终值为止。

有两类计数循环；一类是执行循环体之前先检测循环变量，条件不满足则退出循环。这叫做先判定后执行。另一类是先执行循环体后检测循环控制变量。叫做后续判定结构，此种情况循环体至少要执行一次。究竟是哪一类取决于 BASIC 的版本。

2. 多数情况下，有些问题的求解，只知道控制条件，事先又无法知道重复循环的次数的情形，此时可使用条件控制循环的当型结构 WHILE/WEND。它的格式如下：

WHILE　条件表达式
循环体
WEND

是先判定后执行类型。先检测条件表达式是否成立(真)，如果成立则执行一次循环体，否则循环终止。

计数循环和条件循环两种方法在设计循环结构时均可替代使用。条件循环适用范围较广，是一种常用的循环结构。

问题 8　IF/THEN 语句与 FOR/NEXT 语句嵌套使用时，有什么规则？

答：1. 在 IF 语句中嵌套了 FOR/NEXT 语句。

IF 语句的格式是：IF 表达式 THEN 语句组 1 [ELSE 语句组 2]。必须注意，IF 语句只能出现在一个逻辑程序行中，不能分开写在一个以上的程序行中。如

```
110 IF A>B THEN PRINT A
120 ELSE PRINT B
```

是非法的，应该写成

```
110 IF A>B THEN PRINT A ELSE PRINT B
```

在语句组 1 和语句组 2 中可出现任意 BASIC 语句。当 FOR/NEXT 嵌套在 IF 语句中时，要注意不能将 IF 语句分割成几个程序行。必须保证在嵌入 FOR/NEXT 结构后，原 IF 结构仍在一个行号下。如

```
100 IF A>100 THEN FOR I=1 TO 20:A=A+I×I:NEXT I
```

如果嵌入后程序行超长，可采用调子程序的方法解决。

2. 在 FOR/NEXT 中嵌套 IF/THEN 语句。

FOR/NEXT 结构是由 FOR 和 NEXT 语句为语句括号构成一个语句组或程序段。因此，一般对出现在 FOR/NEXT 结构中的 IF/THEN 语句没有特殊的限制。

问题 9　使用子程序有什么好处？

答：BASIC 语言提供了用 GOSUB/RETURN 语句调用程序内部子程序的方法。

内部子程序的编写方法是将程序中多处需要调用的一般程序提出，编写成一个公用子程序段。在主程序内的任何地方都可对它重复调用。大量使用子程序后会使系统开销（进/出栈操作增加，程序执行速度下降）略有增加。然而这很小的代价换来了较大的好处，比如程序的结构化程序提高了。源程序代码量的减少也为程序换来了较大的工作空间。同时，程序结构清晰，不仅编程效率提高了，也改善了程序的可读性，为程序的维护和移植带来了很大的方便。

问题 10　下列程序段各完成什么功能？

(1)
```
10 INPUT "N=";N
20 IF N<=0 OR N>50 THEN 10
```

(2)
```
10 S=0
20 FOR I=1 TO 10
30 READ A:S=S+A
40 NEXT I
50 DATA 5,7,8,3,1,6,12,23,35,22
```

(3)
```
10 FOR I=1 TO 4
20 FOR J=1 TO I
30 PRINT TAB(4*J+1);I*J;
40 NEXT J
50 PRINT
60 NEXT I
```

(4)
```
10 P=1
20 FOR I=1 TO 20
30 P=P*I
40 NEXT I
```

(5)
```
10 INPUT A$
20 L=LEN(A$):DIM C$(L)
30 FOR I=1 TO L
40 C$(I)=MID$(A$,I,1)
50 NEXT I
```

(6)
```
10 A=RND(1)*100
20 B=INT(RND(1)*90+10)
30 PRINT A,B
```

(7)
```
10 INPUT "N=";N
20 IF N<100 OR N>999 OR INT(N)<>N THEN 10
30 A=INT(N/100)
40 B=INT(N/10)-A*10
50 C=N-A*100-B*10
60 PRINT C;B;A
```

答：(1)从键盘接收[0,50]之间的实数 N。

(2)求十个数的和。

(3)打印三角形数阵：

```
1
2  4
2  6  9
4  9  12  16
```

(4)求 1×2×3×……×10 的积(即求 10!)

(5)把从键盘输入的字符串分离成单个字符，并存入数组 C$。

(6)产生一个(0,100)之间的随机数 A 和一个[10,99]之间的随机整数 B。

(7)从键盘输入一个三位数，把这三位数字分离出来，并按逆序打印。

问题 11　下列程序中空白处应填入什么内容？该程序是求下面二维数组中最大元素所在的行和列。

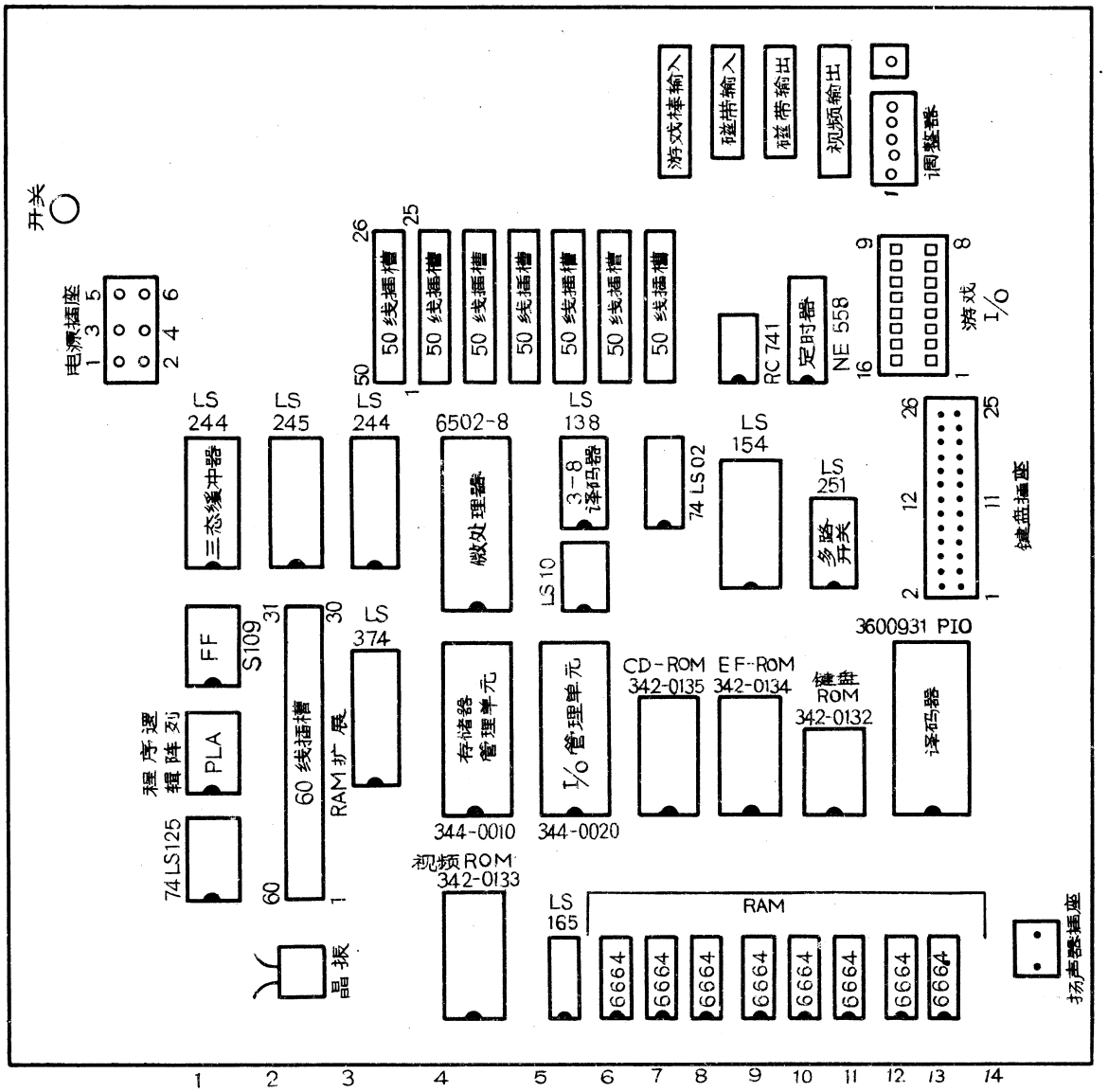
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 3 & 7 & 5 & 4 \\ 2 & 6 & 2 & 3 \end{bmatrix}$$

```
10 DIM _____
20 FOR I=1 TO _____
30 FOR J=1 TO _____
40 READ A(I,J)
50 NEXT _____
60 NEXT _____
70 FOR I=1 TO _____
80 FOR J=1 TO _____
90 IF MAX>A(I,J) THEN _____
100 MAX=_____ :II=I:JJ=J
110 NEXT J
120 NEXT I
130 PRINT II;JJ;MAX
140 END
150 DATA 1,2,4,5,3,7,5,4,2,6,2,3
```

答：(按行号顺序)A(3,4)、J、NEXT I、4、3、110、A(I,J)

问题 12　下面程序是求 20 名同学每个同学的五门学科考试总成绩，程序有哪些错误？(转至 36 页下)

Apple Ⅱe 主板元件排列图



把发展电子工业放到突出位置

中共中央关于制定国民经济和社会发展十年规划和“八五”计划的建议

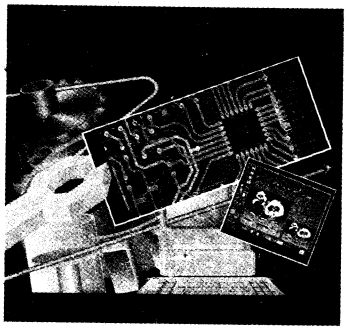
电子工业是促进我国产业结构现代化的带头产业，未来十年要从投资分配、技术开发、设备更新、产业政策和组织管理等各个方面，为电子工业的迅速发展和推广应用创造条件。集中力量开发以大规模集成电路为中心、计算机为主体的投资类电子产品，大力加强微电子技术、计算机与软件、传感器的开发和在国民经济中的普遍应用，积极利用电子技术改造传统产业，促进新兴产业的成长。努力发展光纤、卫星、微波等通信产品和消费类电子产品，以满足生产发展和人民生活提高的需要。

（转载《人民日报》1991年1月29日头版头条）

E&C 電子與電腦 6

一九九一年

总期第75期



電子與電腦

ELECTRONICS AND COMPUTERS

目　录

·综述·

80年代微机发展回顾及90年代展望(上) …… 陈志雄(2)

·PC用户·

汉字同音字查询的实现 …… 李凤明(4)
如何提高cdBASEⅢ数据库汇总速度 …… 刘昌斌(6)
文件目录修复一例 …… 唐银红(6)
多功能卡日历/时钟功能的开发 …… 张翠玲(8)
Turbo C问答 …… 施悦华(9)
就Turbo C谈充分利用系统内存 …… 宋卫东(9)
女子健美体型电脑咨询程序 …… 黎彦才等(10)
使清屏画面变美妙 …… 王　存(11)
技巧二则 …… 汉　神(12)
利用DEBUG进行反汇编存盘的方法 …… 金林樵(13)
90年全国计算机通讯赛题分析 …… 汪五一(14)

·学习机之友·

CEC-I全功能造字 …… 傅叔平(16)
梵塔问题和奇数幻方的一行程序 …… 陈　纲(18)
APPLE-Ⅱ机打印格式的控制 …… 张冠玉(19)
将BASIC程序变成B型文件并以BRUN运行 …… 杨　峰(20)
APPLE DOS的一处错误 …… 尹进渝(20)
ProDOS系统盘的修改 …… 廖　凯(21)
制作40磁道磁盘的最简方法 …… 瞿　波(21)
PC-1500机新监控程序 …… 沈玉波(22)
关于四舍五入程序的讨论 …… 梁　放　叶德祥(25)

·C语言初阶讲座·

第五讲　程序控制结构(二) …… 李　斌　王玉华　蓝智斌(23)

·学用单片机·

单片机最小应用系统 …… 张培仁(26)
电脑坦克
——KDC最小系统应用实例之一 …… 张培红　严　琦　阮永光(27)

·电脑巧开发·

8031与PP40描绘器接口技术 …… 皇甫文忠(29)
解决TEC-B1不能使用AP51-I卡问题的方法 …… 刘希栋(32)

·学装微电脑·

学装μP-80微电脑控制电梯模型的实验与制作 …… 易齐干(33)

·初级程序员级水平考试辅导问答·

汉字dBASEⅢ使用技巧(上) …… 王路敬(37)

·维修经验谈·

彩色显示器亮度失控的探讨 …… 胡野红(45)
微计算机的维修工具—8088系统板故障诊断卡 …… 董　珊(46)

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:北京市邮政局
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:徐云鹏
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

80年代微机发展回顾及90年代展望

艺高电脑国际有限公司总经理　陈志雄

微型计算机是指单芯片微型处理器(microprocessor)为中央处理中心构成的计算机，早期的微处理器功能有限，所以都用于要求低的个人应用计算方面，所以微机同个人计算机(Personal Computer)便常常相提并论。在80年代微处理器技术不断的进步，不少特别设计的微机在工程应用、设计方面有突破性的发展，并一步步代替了不少大型及小型计算机在工程应用、设计上的工作，因为在这一方面的微机好象代替了大型或小型计算机上联网的终端机或工作站，所以该系列的工程应用微机便被称为工作站，与个人电脑在不同的技术及市场领域里发展。

虽然微型机是一个通用名称，但为免得再增加混乱，在这里我以微型机代表个人计算机系列，而工作站则代表为工程应用而设计的计算机系列。

微处理器在70年代初期已制作成功，但微机的发展则到80年代才真正的普及化，自美国IBM公司推出的PC系列之后，市场心理趋使硬件及软件公司相信IBM PC系列将会成为微机标准，因而大量资金投资于PC系列的硬件及软件上。而位于东南亚区的电子厂亦一窝蜂的投资生产IBM PC兼容器，大量的供应迫使价格下降，使微机更容易普及化。在大量软件及低价格的相互作用之下，IBM PC系列微机便发展成一公认的微机标准。80年代便是利用这公认的微机标准不断的发展并普及化，但这标准同时亦带来技术的局限，因为要达至兼容的问题，微机的主要技术指标差不多全部控制于美国Intel公司的手里，在80年代Intel公司推出了8088,80286,及80386系列微处理器，而微机的中央处理器则只能根据Intel公司的发展而进步，直至今天还没有其他比Intel更强的微处理器跟Intel分庭抗礼。

在工作站方面发展跟微机可说是完全不同，工作站到现时还未有一个公认的标准，它的发展要靠针对用户的直接需要，不停的开发新技术，以先进的产品争取市场，很有趣的是微机是以大公司作为领导，但在工作站上比较小的公司反而取得领导地位，在80年代大型计算机公司如IBM,DIGTAL等在工作站的市场上一败涂地，而在不同时间上比较少的公司此起彼落，从早期的Computer Vision,Calma,Apollo,Mentor Graphics,到Daisy,silicon Graphics及近期的HP/Apollo,SUN,MIPS等等都是工作站市场的佼佼者。在技术方面，因不受一个标准所限，各施各法，都以更高的技术及产品取胜，单是微处理器便品类繁多，而且大都比当时的Intel微处理器功能更强，从早期的Motoroal 68000系列，National Semiconductor的32000系列，到现时Motorola的88000新系列，Intel的i860/i960，MIPS的R2000/R3000及SUN的SPARC都是功能强运算速度快的微处理器。除了微处理器外，工作站在其他技术指标方面亦远胜微机如总线速度，网络功能及图形发生及显示器等。

90年代的发展，可能有戏剧性的突破，在微机方面，虽然Intel在1990年推出的80486，垄断可能会被打破，如微机技术的发展不受到标准的局限，而在同一标准的范筹下，微机在性能价格比方面将会直追工作站。现时工作站的领导公司亦正尽再大的努力希望能把本身的技术及产品发展成一公认的标准，如标准化能成为事实90年代工作站的价格亦会大幅回落，直接向微机市场挑战。

90年代最终的发展是怎样，现时还言之过早，本文将对微机及工作站以往的市场技术作分析，并对最近将来的发展作一预测，读者可作自己的结论。

一、微处理器的发展

1. 微机的微处理器发展

在IBM推出PC系列以前，Z80及6502微处理器都是最为普通而分别于S—100总线CP/M操作系统的微机及苹果Ⅱ型微机之上。在IBM推出PC系列之后，PC系列的Intel 8088微处理器便从此变为主流，在80年代后期Intel推出16位的80286，32位的80386，而在90年推出的80486其实只是在80386的基础上加上协处理器(Co—processor)及8K缓冲存储，在功能上8088只是一个简单适合单用户的8位微处理器，而16位的80286对多任务要求在设计上作出安排，而32位的80386DX是一个同步32字节能支持并行处理的微处理器，80386SX是80386DX的简化体，386SX的内部结构基本上和386DX一产。只是在数据传达/接收时需要两个时钟循环，而不是386DX的一个时钟循环，所以在数据运算速度方面，386SX跟80286基本上相差不远，两者相差的是386SX内部是一个完整的32位微处理器可以运行专为386DX编写的多用户系统软件，而80286则不能。

Intel公司在90年代的发展方向已很明显，所有以后为微机开发的微处理器都会以80386DX为基础，加上外围功能，这些外围功能将包括网络、多媒介传讯(multi——media)如声音、图形辩认、输入/输出、人功智能等等。现时的80486芯片上已集成了一百二十万

个晶体管，但到公元 2000 年则 Intel 公司估计他的 Microchip2000 已可能成功集成一亿个晶体管在一个单芯片上。

Intel 的技术发展方向使很多半导体设计/生产厂家明白到如果他们不能生产一个跟 80386DX 兼容的微处理器，他们在 90 年代便完全没有机会在这个市场上跟 Intel 竞争。问题当然是在要设计一个跟 80386DX 兼容但又不会侵犯 Intel 专利权的微处理器，不管可行性如何，市场上估计最少已有 6 个厂家正在研制 80386DX 的兼容产品，他们能否成功还需拭目以待，但美国的 AMD 公司(Advanced Micro Devices)则成功推出了他的 80386 芯片，但 AMD 的成功并不代表其他厂家可同时取得成功，因为 AMD 跟 Intel 是签订过二线供应合同，即 Intel 与 AMD 根据合同需各自提供本身的设计的产品给对方生产，作为对方的二线供应商。Intel 在 80286 之后已停止供应资料给任何半导体厂家，但 AMD 仍然自行开发 80386DX 微处理器，虽然 Intel 在美国法庭控告 AMD，但市场人仕相信因 AMD 过往跟 Intel 签订的合同仍有效，所以 AMD 胜诉的机会很高，而 Intel 将不能以侵犯专利为理由禁止 AMD 生产及销售的 80386DX 芯片。只要 AMD 可以生产 80386DX，80486 便不成问题，而且理论上 AMD 已可以设计比 80486 更多接口功能的微处理器而仍然跟所有标准软件兼容，这可说是直接挑战 Intel 公司十年来的垄断及控制局面。

假定其他正在研制 80386DX 兼容的公司能在技术上或在法庭上解决 Intel 的专利权问题，可以想象不单 80386 和 80486 芯片的价格将会大幅回落，而且更新更强的微处理器将会不断推出，情况将跟工作站用的微处理器的发展一样。

2. 工作站的微处理器发展

象 Intel 的 8088/286/386/486 占领着微机市场一样，早期 Motorola 的 M68000 系列亦差不多占领了整个工作站的市场，现时大部份的工作站厂家都设计微处理器，现将一些标准型而在市场上可购买的微处理器简单作一介绍，介绍暂不包括专用的微处理器(如 IBM RISC，DEC VAX 及 HP/PA 等)。

工作站微处理器可分为两大类：普通指令系统计算机(CISC)及精简指令系统计算机(RISC)。

(1)CISC 普通指令系统计算机

(Ⅰ)Motorola M68020/030/040

以往大多数的工作站(如 Sun，Apollo，HP 等)都是采用 M68020/030。但 68030 的速度太慢，而 68882 浮点运算速度亦没法跟 RISC 竞争，所以工作站的设计者已一一停止使用 Motorola 的 68000 系列，现时只有 HP/Apollo 还是 Motorola 的唯一最大用户，最新 Motorola 的 68040 含浮点运算器，从 Motorola 提供的数据来看，它确实可与 SPARC 和 MIPS 并驾齐驱。

(Ⅱ)intel 80286/386/486

虽然 Intel 的微处理器在微机中处于垄断地位，但没有一个工作站厂家能成功的推广 386 为基础的工作站，SUN 的 386i 可说是完全失败的，但 80486 加上缓冲存储有差不多 20 个 Mips 的速度，所在廉价的无盘工作站(diskless workstation)亦可能会有新的生命力。

(Ⅲ)National Semiconductors 32032/32332/32536

NS 的 32000 系列微处理器是市面上第一个 32 位微处理器产品，性能优良，但它在工作站市场上则从来没有成功过，反之它作为接口器件的控制器则十分成功(如激光打印机和传真机的控制器)。

(2)RISC 精简指令系统计算机

(Ⅰ)Sparc

这是目前工作站用的 RISC 处理器中货源最广的一种，至少有五家半导体厂可供应 Sparc 晶片。包括 Fujitsu(门阵列)，Cypress(标准元)，Bipolar Integrated Technology(放射连结逻辑)，LSI Logic(优化门阵列)和 TI(和 SUN 合作的专用 IC)。PHILIPS 与 MATRA 亦宣布加入 Sparc 供应行列。Matsushita 和 Solboarne 正合作开发 64 位的 Sparc，而砷化镓(GaAs)的 Sparc 亦将面世，其速度可达至 200Mips。

(Ⅱ)MIPS R2000/R3000

MIPS 计算机公司的 R2000/R3000 系列是 Sparc 的强劲竞争对手，在 16—17MHz 主频时速度是 20Mips。该系列起源于斯坦福大学，目前已有很多主要的工作站供应商采用，例如 Digital 已用于他的 DEC station，而 silicon Graphics 更已使用在他全部的系统中。MIPS 微处理器的主要弱点是在市场上的数量少，软件的基础较差，但它的长处是内存分配速度快(Memory mapping)，而且它的集成度也比较高，每套的芯片数目较少。

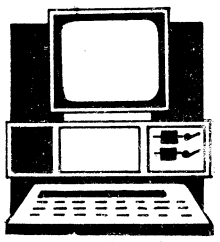
(Ⅲ)Intel i860/i960

Intel 公司的 RISC 产品主要有两个系列：i960 主要用作接口器件的控制器，而 i860 则是设计用于工作站的。本文将只讨论后者，从性能来说 i860 是众多 RISC 芯片中性能最强的其中一个，它构成了一个整体数运算器(Integer unit)，一个浮点运算器(floating point unit)，一个图形处理器(Graphics Processing unit)，一个在芯片上内存管理器(memory management unit)和缓冲内存(cache)。其速度大约为 25-30Mips 而其浮点运算速度最高可达 40Mflops。单从数字来看 i860 是一种很有竞争能力的产品，但它的问题是缺乏现成的软件。目前 i860 只在 HP/Apollo 的 4000/VRX 中用作图形处理器，在 Floating Point System 公司的 FPS 5000EA 用作阵列处理器。如 i860 的软件充分，它是一个功能极强的工作站中央处理器。

(Ⅳ)Motorola M88000

Motorola 的 M88000 RISC 还未在工作站市场上有任何影响，Unisys 公司现时是唯一选择了 M88000 作为工作站的公司，估计 Motorola 的 M88000 将不会成为主流产品。

可以看得到工作站的微处理器发展得比微机的 286/386/486 快，这是在没有标准局限的结果，所以 90 年代的微机(PC 机)微处理器的发展(下转 28 页)



PC用户

汉字同音字查询的实现

绵阳市职工社会保险事业管理处　　李凤明

在人事档案、户籍、工资等管理系统中，通过输入姓名或其它汉字信息进行查询是较常用的查询方法。这类查询一般先按查询内容进行索引，然后通过定位函数定位来实现查询，它要求输入的查询内容必须和索引内容一致才能正确定位查询。在查询时，如果输入了同音汉字，它就无能为力了。

这是因为使用索引进行查询时，是将查询内容的ASCII码与索引内容的ASCII码进行比较，两者一致时才认为是查询的记录，而同音字的机内码各不相同，按这种查询方法，若只知道姓名或其它汉字信息，将会出现漏查。在实际工作中，经常会出现同音的情况，这就需要一种汉字同音字查询技术。

根据《通讯用汉字字符集(基本集)及其交换码国家标准(GB2312-80)》，汉字是按拼音字母顺序排列的，同音字基本上在同一区中。每个汉字的机内码为两个字节，其高字节部分确定了它的区号。对汉字的区号进行比较，区号相同即可能为同音字，比较的汉字个数越多，同音的情况也就越确定。

根据这个思想，现介绍一种用Foxbase+2.00版本实现的，可对数据库中的姓名字段进行同音查询程序XMTYCX.PRG(程序附后)。程序为一个过程，可被不同的数据库调用。另外函数XMCL.PRG(程序附后)对姓名进行处理，将输入的姓名中的空格去掉，在数据录入时应调用该函数处理姓名。

假定现有职工工资数据库GZK.DBF(库结构附后)，已按姓名字段建立了索引文件GZK.IDX；显示某人具体情况的程序名为QKXS.PRG。通过下面的语句即可实现对GZK库中姓名的同音查询：

DO XMTYCX WITH "GZK","GZK","姓名","QKXS"

程序只需稍加修改，就可实现对其它汉字信息的同音字查询，也可用其它语言实现。程序在SPIC 286微机和浪潮0520-D微机上运行通过。

功能说明：1、可对姓名进行同音字查询，一屏显示18个记录，也可前后翻页。

2、输入记录号可显示某人的具体情况。

调用参数：1、查询库名；2、查询库按姓名索引文件名；

3、姓名字段名；4、显示具体情况子程序。

调用形式：

DO XCTYCX "查询库名","姓名索引文件","姓名字段名","显示情况子程序"

调用注意事项：本程序将选择J工作区打开数据库，对其他工作区无影响。

同音查询程序(XMTYCX.PRG)：

```
para x,y,z,w
dime cxm(4)
priv oldse,i,j,n,xxm,re ne,con
set stat off
set talk off
set esca off
oldse=ltri(str(sele(),2,0)) && 保护原工作区号
sele j
use &x inde &y
do whil .t.
xxm=spac(8)
clea
@24,0 say "请输入姓名(按 ESC 键退出)" get xxm
read
if read () =12   && 按 ESC 键退出
exit
endi
if len(trim(xxm))=0
loop
endi
xxm=xmcl(xxm)  && 调姓名处理函数处理查询姓名
cxm=""
cxm(1)=left(xxm,1)
seek cxm(1)  && 定位姓同音开始记录
if eof()
@24,0 clea
@24,0 say "查无此人，按任一键继续查询……"
    read
    loop
endi
n=len(xxm)
i=3
j=2
do whil i<=n  && 取查询姓名每个字的高字节定区号
cxm(j)=subs(xxm,i,1)
i=i+2
j=j+1
endd
ne=18  && 定义每屏显示记录个数
con="subs(&z,3,1)=cxm(2).and.subs(&z,5,1)=cxm(3)"
&& 定义姓名同音条件
do whil .t.
@2,0 clea to 24,79
@24,0,say "正在查询，按 ESC 键停止查询……"
@2,0 say "记录号------姓　名"
i=1
do whil &z=cxm(1).and.i<=ne
```

```
if inke()=27  && 按 ESC 键停止查询
exit
endi
if &con  && 显示满足条件的记录
@2+i,0 say str(recn(),6)+"              "+&z
i=i+1
re=recn()
endi
skip
endd
if i=1  && 文件结束未查到
@24,0 say  "查无此人,请按任一键重新输入……"
read
exit
endi
i=0
@24,0 say "请输入 ↑——上翻 ↓——下翻 ESC——退出;
或输入记录号" get i pict "999999" rang 1,recc()
read
if read()=12  && 按 ESC 键退出
exit
endi
do case
case read()=4  && 上翻
skip -ne * 2
case i>0  && 显示某人具体情况
go i
on erro exit  && 出错退出
do &w
go re
clea
othe
go re
endc
if &z<>cxm(1)
seek cxm(1)
endi
endd
endd
on erro  && 取消出错判断
use
sele &oldse  && 恢复原工作区号
retu
```

```
* 姓名处理函数(XMCL.PRG)
* 去掉字与字之间的空格
para xx
priv i,j,n,nxm
xx=ltri(trim(xx))
n=len(xx)
i=1
nxm=""
do whil i<=n
if subs(xx,j,1)<>"  "
nxm=nxm+subs(xx,i,1)
endi
i=i+1
endd
retu nxm
```

```
* 情况显示子程序(QKXS.PRG)
clea
disp off
wait
retu
```

职工工资数据库 GZK.DBF 结构:

Structure for database: E:\GZK.DBF
Number of data records: 90
Date of last update : 03/08/91

Field	Field Name	Type	Width	Dec
1	姓名	Character	6	
2	基础工资	Numeric	5	2
3	职务津贴	Numeric	6	2
4	粮贴	Numeric	2	
5	燃贴	Numeric	2	
6	综合补贴	Numeric	2	
7	交通费	Numeric	2	
8	书报费	Numeric	2	
9	副贴	Numeric	2	
10	肉贴	Numeric	2	
11	洗理费	Numeric	2	
12	合计	Numeric	5	1
** Total **			39	

程序运行结果

```
. do xmtycx with "gzk","gzk","姓名","qkxs"
请输入姓名(按 ESC 键退出)李奉鸣

正在查询,按 ESC 键停止查询……
记录号  ------    姓  名
    12            李凤明
请输入↑——上翻 ↓——下翻 ESC——退出 或输入记录号
12
姓名    基础工资  职务津贴  粮贴  燃贴  综合补贴
李凤明     39.00     29.00     0    15        50
        交通费  书报费  副贴  肉贴  洗理费  合计
             5       6     5    10       6  165.0
Press any key to continue……
```

如何提高 CdBASE Ⅲ 数据库汇总速度

广西玉林建设银行电脑室　　刘昌斌

目前，大部分管理机构每月、每年均要用 CdBASE Ⅲ 程序汇总各个下级机构的报表数据。我们建行系统每月、每年都要定期汇总下级行资金平衡表，这些下级行的资金平衡表数据库结构、记录个数和顺序各自相同，而且他们与汇总库的结构、记录个数和顺序也相同。汇总各下级行的数据实质就是把各个下级行的数据库按科目号相同的记录的数据累加到汇总库中。

CdBASE Ⅲ 可用于汇总数据的语句和方式很多，但是目前被广泛使用的汇总方法基本是：首先打开过渡汇总库，然后用语句“APPE FROM(数据库名)”的形式把各下级行的数据库记录一个一个地添加到过渡汇总库，这样就产生了一个庞大的过渡性数据库，对其按某个关键字建立索引后，再打开索引，此后再用“TOTAL TO（汇总库名）ON（关键字）ALL”语句对过渡汇总库具有相同科目号的记录汇总到汇总库内。用这种方法汇总数据速度很慢，因为用 APPE FROM 语句较为耗时，对庞大的数据库建立索引就是耗时的致命的弱点，用这种方法把结构为 14 字段、241 个记录的 14 个下级行数据库汇总需要 1 小时。

虽然用户都知道上述数据汇总方法的速度较慢，但是有不少的用户忽视了应用 REPL 语句进行数据库汇总的技巧。由于各下级行的数据库结构都与汇总库相同，这样就可以建立公共索引，然后在两个不同的工作区分别打开汇总库及其索引文件和下级行库及其索引文件(用语句“SET RELA TO（关键字）INTO(数据库别名)”，把汇总库和下级行库建立了关系(根据关键字连接两个数据库)，此时仅用一个语句“REPL ALL 字段名 WITH 字段名＋别名－＞字段名，…”，就可把一个下级行的数据汇总到汇总库内，速度极快。例：应用本方法汇总 14 个下级行的数据，汇总库名为 JPYSUM. DBF，下级行数据库名为 JPYn. DBF(n＝1——14)，他们的记录个数都是 241，他们的结构是：

字段名	类别	宽度	小数
FKMH	C	4	0
FKMM	C	32	0
FYCYE	N	14	2
FBYS	N	14	2
FBYF	N	14	2
FYE	N	14	2

应用程序(一)的技巧方法汇总数据，不但可极大地提高数据汇总速度，而且避免了产生庞大数据库，同时节省大量的外存空间，简直是一举三得。该数据库汇总方法具有通用性，适用于各个管理机构，现提供各用户借以抛砖引玉。

程序(一)

```
* * 技巧汇总数据库程序 JPYHZ. PRG
sele 1
use jpysum
rep all fycye with 0,fbys with 0,fbyf with 0,fye with0
inde on fkmh to jpy11
use jpysum inde jpy11
store 1 to n
do while n<15
@ 6,32 say "汇总第"+str(n,2)+"下级行"
if n<10
store "jpy"+str(n,1) to jpyn
else
store "jpy"+str(n,2) to jpyn
endif
sele 2
use &jpyn inde jpy11
sele 1
set rela to fkmh into &jpyn
repl all fycye with fycye+b－>fycye,fbys with fbys+b－>
fbys,fbyf with fbyf+b－>fbyf,fye with fye+b－>fye
sele 2
clos inde
n=n+1
enddo
clos data
retur
```

文件目录修复一例

湖北远安县财政局　唐银红

由于病毒侵袭，破坏了硬盘上的文件目录区，用 DIR 命令显示硬盘文件目录时(不带任何参数)，目录区前面的绝大部分目录项显示正常，到最后部分显示有 4—5 行乱七八糟的字符，并且每行都要显示到 80 列，除有能正常显示的英文字母、数字及普通字符外，还有大量的控制字符，整个硬盘上有十几个文件丢失。

以上现象表明：虽有十几个文件目录丢失，但系统认为这些文件仍然存在，继续占用硬盘空间。为了验证这一现象，用 PCTOOLS(R4。30 版)显示硬盘根目录，被破坏的部分的文件名、文件长度、建立日期等信息，竟然好端端地显示在目录区的最后，当试图对这些文件进行访问(读写、拷贝操作)时，PCTOOLS 的反应是系统中没有这些文件，访问失败。只能将这些文件名一

一抄录下来。

这些文件既占用了硬盘空间，又不能对其进行读写操作，必须对其进行删除操作。显然，这些文件的文件名都不能正常显示，就无法用 DOS 的命令 DEL FILENAME 进行删除操作。当然可以用 DEL *.* 命令或对硬盘重新进行格式化工作，但却要对硬盘进行备份，恢复后还要重建有关必要数据，不仅显得麻烦，工作量大，而且耗时长。我们知道，DOS 对文件目录的管理，是以文件名的第一个字符来区分一个文件是否是有效文件的，若文件被删除后，DOS 就将该文件名的第一个字符改为十六进制的 E5H。因此，只要将被破坏的文件名的第一个字符改为 E5H 即可。用以下两种方法都可以达到修复的目的。

方法一：用调试程序 DEBUG 修复。

1、用 DIR 命令显示文件目录，记下不能正确显示的文件前面的最后一个文件名，本例为 DBASE.EXE。

2、DEBUG＜ENTER＞进入调试程序。

3、将硬盘根目录区调入内存。

— L100 2 11 20 ＜ENTER＞

100 为装入内存中的起始地址，2 是 C 盘（硬盘）的代号，11 是 20M 硬盘根目录区的相对扇区起始号，20 是要读入的扇区数。

4、— S 100 4100 'DBASE　　EXE' ＜ENTER＞

装入的目录长为 16KB（32 个字节表示一个文件名的全部信息，装入的硬盘目录共有 512 个），用十六进制数表示为 4100H，因此要在 100H－4100H 范围内搜寻名为 DBASE.EXE 的文件。注意，文件名必须大写，文件名不足八个字符的部分用空格键补齐，文件名与扩展名之间不能有任何分隔符。

当 DEBUG 找到 DBASE.EXE 文件名后，立即显示文件名在内存中的地址，本例为：

4285：1CA0

5、—D1CA0 1DBF

从下个目录开始，观察不能正常显示的文件个数，也可省略这一步，根据前面用 PCTOOLS 观察时记录的文件个数。

6、进行修改。具体是从第一个不能正常显示的文件目录开始，用 E 命令修改。

— E1CC0 ＜ENTER＞

4285：1CC0 18.E5＜ENTER＞

修改第二个不能显示的文件名时，将地址值在前一个文件的地址值基础上，加上 20H 进行修改，以此类推，下一个文件的首地址均在上一个文件名首地址值的基础上加 20H，直到修改完毕。

7、— W 100 2 11 20 ＜ENTER＞

将修改后的目录内容写回硬盘。注意，读盘时用什么参数，写盘时一定要一致。

8、— Q ＜ENTER＞ 退出 DEBUG，返回 DOS。

用 DEBUG 进行修改，不需要工具软件，但不方便，对不熟悉 DEBUG 使用方法的用户来说，稍有差错，损失就更大了，建议用第二种方法修改。

方法二：用工具软件 PCTOOLS 进行修改（PCTOOLS 版本号为 R4.30）。

1、PCTOOLS ＜ENTER＞

2、F3 进入磁盘服务功能，选择根目录区。

3、E 进入磁盘编辑功能，此时，显示器上显示磁盘数据信息，从绝对扇区 0 开始显示，每行显示 16 个 ASCII 码，左边为十六进制数，右边显示对应字符，按 PgUp、PgDn 两个键，可以使显示内容上移一屏或下移一屏。首先显示的是三个系统文件的内容，然后是根目录区，连续翻动，直到找到文件 DBASE.EXE 为止。

4、F3 打开编辑开关，允许对磁盘文件进行编辑（读写操作）。用四个光标键（UP ARROW、DOWN ARROW、LEFT ARROW、RIGHT ARROW）移动光标，到不能正常显示的文件目录的第一个字符处，改成 E5H（在左边十六进制区修改，按 F1 键可调换修改的区域为十六进区或 ASCII 字符区），每二行（32 个字符）显示一个文件名及其有关信息，重复上述操作，直到修改完所有的不能正常显示的文件目录为止。一次可修改两页，按 PgUp 、PgDn 键可交换上下页进行修改。

5、F5 进行修改确认，保证修改后的内容有效（F6 放弃，使修改作废）。

6、＜ESC＞退出磁盘编辑功能，再按一次＜ESC＞键，确认退出 PCTOOLS。

7、DOS 控制下，运行带 F 参数的外部命令 CHKDSK/F，将系统丢失的簇号收回。

8、根据前面所列丢失文件名的清单，将删除的文件逐个从软盘拷回。

至此，整个文件修复工作完成。

注一：修改过程中，多次用 DIR 命令检测硬盘可用空间。修复工作开始前，用 CHKDSK/F 命令，不能检测到系统有丢失的簇，证明系统认为那些文件占用的磁盘空间是合法的；至第六步结束，用 DIR 命令列硬盘目录，显示的硬盘可用空间没有发生变化，表明删除的文件占用的磁盘空间不能正常释放，至第七步结束，用 DIR 命令列硬盘目录，发现硬盘上增加四十多个 FILE*.CHK 文件，将这些文件删除后，硬盘可用空间增加 300KB 之多，表明修复成功。

注二：因为文件目录信息已全部丢失，包括文件名、文件长度、首簇号、建立日期、文件属性、修改标志等，所以修复时不能直接改为原文件名，而必须进行删除操作。

注三：文中带下划线的部分由用户输入，＜ ＞表示为一个键，例如＜ENTER＞表示回车换行键。

多功能卡日历/时钟功能的开发

辽宁公路专科学校　张翠玲

目前的绝大多数 PC/XT 兼容机和家用 PC 机上，均安装有多功能卡。它为用户提供了五种功能，即软盘控制器接口、并行打印机接口、串行通信接口以及游戏适配器等。在卡上藏有一枚 3 伏锂电池，当计算机关机时，时钟芯片可以通过电池的能量来维持时间和日期。但由于缺少相应的驱动程序，使得日历/时钟功能没有得到充分的发挥，白白浪费了硬件资源。为此，有必要开发日历/时钟驱动程序，以省去每次开机时输入日期和时间的麻烦。

一、多功能卡的硬件特性

在多功能卡的上方有两排各十脚的插座，脚与脚之间的不同连接，可以改变多功能卡的功能。对于实时时钟（参照下图），当第 4 脚与第 3 脚连接时，为时钟 1 有效；而第 4 脚与第 6 脚连接时，为时钟 2 有效；如果第 3、4、6 脚均断开，则时钟功能无效。因此，为了能使时钟正常工作，必须保证 4、3 脚或 4、6 脚有一对是相连的。

2	4	6	8	10	12	14	16	18	20	上
1	3	5	7	9	11	13	15	17	19	下

插脚连接方式的不同，在访问时钟芯片时所采用的地址也是不同的。多功能卡时钟芯片上的硬件地址规定如下：

时钟 1：&H240

时钟 2：&H340

二、对时钟芯片的编程

多功能卡上的时钟芯片内包含有时间和日期各部分计数器，见表 1。每一个计数器可以通过时钟芯片地址的相对偏移量来访问。

表 1：时钟芯片相对偏移地址的内容

时钟芯片口地址	说明
+0	万分之一秒计数器
+1	百分秒计数器
+2	秒计数器
+3	分计数器
+4	小时计数器
+5	星期计数器
+6	日计数器
+7	月计数器
+8	年计数器
+9	上一个月份数
+A	上一年数

为了能对时钟芯片进行编程，首先必须确定出时钟芯片的端口地址。这可以根据插脚的连接方式来判断，但这种方法显得有些笨拙，因为不同的多功能卡其插脚的连接方式可能有所不同。寻找芯片较好的方法是读一些通常被分配给时钟的口地址，如 240H 和 340H 口，与硬件无关的口将返回值 0FEH 或 0FCH。例如要检验秒计数器时，其返回值不能大于 59，否则说明所检验的不是时钟口地址。

寻找时钟芯片的程序段如下：

```
CLKTAB    LABEL   BYTE
          DW      240H
          DW      340H
          .
          .
          LEA     SI,CLKTAB     ;SI 指向时钟表地址
          MOV     CX,2          ;共两个口地址
FIND:     MOV     DX,[SI]       ;第一口地址送 DX
          ADD     DX,2          ;加 2 指向秒计数器
          IN      AL,DX         ;取秒计数器值到 AL 中
          TEST    AL,80H        ;高位设置?
          JZ      GO            ;NO,非空口,继续执行
          ADD     SI,2          ;否则指向下一口地址
          LOOP    FIND          ;
          LEA     DX,MESS       ;显示信息
          MOV     AH,09H        ;
          INT     21H           ;
          INT     20H           ;
GO:       .
          .
MESS      DB      'No clock chip found! $',0DH,0AH
```

虽然时钟芯片本身没有自动增加年数的功能，但从表 1 中可以看到，时钟芯片上的 9 口和 A 口是用来存储上个月和前一年的信息的，我们可以利用这两口的信息对年数进行处理：首先判断上个月和本月两个计数器值的关系，若上个月数大于本月数，则年数增 1，将增加后的年数写回到芯片上，否则年数不变。处理年数的程序段如下：（程序中 MONTH 变量为芯片上 7 口中的内容）

```
MOV DX,24AH          ;从 24AH 或 34AH 口取前一年
IN AL,DX             ;存储器的信息放在 AL 中
MOV AH,AL            ;AL 送给 AH
DEC DX               ;指向 249H 口
IN AL,DX             ;取前一个月的信息放在 AL
CMP AL,MONTH         ;与本月份数比较
JBE CONT             ;不大于本月数,转 CONT
INC AH               ;否则,前一年数加 1
MOV AL,AH            ;
OUT DX,AL            ;写入 24AH 口
```

CONT;

有了上述了解，就可以编写时钟驱动程序了。编程时应注意时钟芯片是采用 BCD 码来存储信息的，因此使用这些信息时需进行一些转换，如要显示日期和时间，需将读出的 BCD 数转换成十六进制数，再转换成 ASCII 码；而在写芯片时，又需将从系统中得到的日期和时钟转换成 BCD 数。考虑到要缩短机器的启动时间及以后使用上的方便，应分别编写出读时钟和写时钟程序，读时钟程序的执行过程大致如下：从时钟芯片端口分别取出时间和日期内容，转换成十六进制数后，再调用 21H 中断的 2DH 号（置系统时间）和 2BH 号（置系统日期）功能，将芯片上的信息传到系统时钟上。写时钟正好与之相反，它先调用 21H 中断的 2CH（取系统时间）和 2AH（取系统日期）号功能，再将时间、日期转换成 BCD 数写到芯片上。

首次使用时，需给计算机输入正确的时间和日期，之后执行写时钟程序，并在系统盘或硬盘上建立 AUTOEXEC. BAT 文件，将读时钟程序的文件名及 DOS 命令 DATE 和 TIME 加入到该文件中，这样每次开机后，系统便会自动显示准确的日期和时间，既利用了硬件资源，又省去输入日期和时间的麻烦。

限于篇幅，源程序清单从略。

Turbo C 问答

天津继电器厂　施悦华

微机上的程序设计语言 TurboC 以它的功能强大和使用方便而深受欢迎，但有些初学者和已能用 TurboC 编程的用户对某些问题没有给予足够的注意，这里就列出几个问题并给出答案：

TC. EXE 与 TCC. EXE 有何不同？

TurboC 带有一个综合编译系统 TC. EXE 和一个行编译系统 TCC. EXE。可以说 TC. EXE 是带有编辑、连接和跟踪功能的 TCC. EXE。两者的配置文件分别为：TCCONFIG. TC 和 TURBOC. CFG。

为什么 TurboC 找不到用户的 #include 文件？

编译系统只在\include 目录下寻找包含文件，这个目录可以通过 Option/Directory 菜单指定。可以把用户的包含文件都放入\include 目录。

为什么出现 Unable to open include file 'stdarg. h'？

因为超过 DOS 允许同时打开的文件数，把 FILE=20 加入配置文件 CONFIG. SYS 中即可。

如何在 TurboC 编辑状态下打印源程序？

用键 CTRL<K><P>，可以打印标记块，如果没有标记，则打印整个源程序。

怎样产生打印输出？

TurboC 在 STDIO. H 中把打印机定义为一个文件，在使用前用户不必打开 stdprn，只须：

```
#include<stdio. h>
main( )
{ printf(stdprn,"Hello. \n");}即可
```

为什么 printf()和 puts()不能显示彩色字符？

必须使用控制台输入/输出语句 cprintf()和语句 cputs()。

如何创建. COM 文件？

用户手中如有 EXE2BIN. EXE，则可以在 Tiny 模式下建立. EXE 文件，然后用它转换为. COM 文件。如果没有，也可使用 TLINK 程序：

```
TCC－mt－lt－tiny↙
```

它建立 tiny. com 代替 tiny. exe（浮点运算不能转为. COM 文件）。

就 Turbo C 谈充分利用系统内存

天津海洋技术研究所　宋卫东

在各种计算机语言中，由于硬件的原故，突破 64K 内存是不容易的（特别是高级语言）。这就使用户受到了限制。特别是目前各种机型的内存正迅速扩大；不充分利用很可惜，也是很大的浪费。下面就谈一下在 TurboC 语言中，如何使用大于 64K 的数据。

例如我们要定义一个 1024×128 的二维数组。您也许会写成：

```
char array[1024][128]
```

但因为尺寸超过了 64K，它必须直接从系统堆去分配：

```
#include(alloc. h)
char (huge * array)[128]
main()
{……
array=farcalloc(sizeof( * array),1024);
……}
```

这一个数组的用法同其它直接分配的数组一样。注意，在数组定义过程中必须使用关键字"huge"因为只有"huge"才能寻址到 64K 以外。这样就扩大了内存的使用范围，提高了计算机的使用效率。

女子健美体型电脑咨询程序

四川泸州医学院　黎彦才　李晓阳　高明扬

近年来，健美运动在我国青少年中得到越来越广泛的开展。在健美训练中，人们希望随时了解自己训练的效果和体型是否符合标准，以及那些部位尚需加强训练等。利用电脑进行健美体型的咨询，无疑是一种高效率而方便的方法。我们根据有关资料在 IBM PC 机上编制了女子健美体型（国内标准）电脑咨询程序，在用户键入测量数据后，电脑将根据输入数据进行计算并打印出咨询单，用户可以从咨询单上看出根据自己身高计算出的自己身体各部位应有的比例以及自己现在体型各部位的实际比例，科学地进行健美训练。

本程序使用简便，只要按计算机屏幕显示的汉字提示，即可方便地进行操作，得到咨询。

女子健美体型标准为：

1. 体重＝（身高[厘米]－100）×0.85[公斤]
2. 胸围：应为身高的一半
3. 腰围：较胸围小 20 厘米
4. 髋围：较胸围大 4 厘米
5. 大腿围：较腰围小 10 厘米
6. 小腿围：较大腿围小 20 厘米
7. 足颈围：较小腿围小 10 厘米
8. 手腕围：较足颈围小 5 厘米
9. 上臂围：等于大腿围的一半
10. 颈围：与小腿围相等
11. 肩宽：应为胸围的一半减 4 厘米
12. 上、下身比例（以肚脐为界）：5∶8

```
5 CLS
15 PRINT"女子健美体型电脑咨询"
17 PRINT"请按提示键入测量数据(厘米)"
20 PRINT"身高＝";:INPUT A
25 PRINT"体重(公斤)＝";:INPUT A1
30 PRINT"胸围＝";:INPUT A2
40 PRINT"腰围＝";:INPUT A3
50 PRINT"髋围＝";:INPUT A4
60 PRINT"大腿围＝";:INPUT A5
70 PRINT"小腿围＝";:INPUT A6
80 PRINT"足颈围＝";:INPUT A7
90 PRINT"手腕围＝";:INPUT A8
100 PRINT"上臂围＝";:INPUT A9
110 PRINT"颈围＝";:INPUT A10
115 PRINT"肩宽＝";:INPUT A11
118 PRINT"脐高＝";:INPUT A12
120 X=(5*A12)/(A-A12):B1=(A-100)*.85
130 B2=A/2:B3=B2-20:B4=B2+4:B5=B3-10
135 B6=B5-20:B7=B6-10:B8=B7-5:B9=B5/2
140 B10=B6:B11=B2/2-4
150 A$="身高":A1$="体重":A2$="胸围"
160 A3$="腰围":A4$="髋围":A5$="大腿围"
170 A6$="小腿围":A7$="足颈围"
175 A8$="手腕围":A9$="上臂围"
180 A10$="颈围":A11$="肩宽"
190 A12$="上身下身比例":CLS
230 LPRINT TAB(18);STRING$(62,"*")
250 LPRINT TAB(34)"女子健美体型电脑咨询"
252 LPRINT TAB(34)"(国内女子健美体型标准)"
260 LPRINT TAB(18);STRING$(62,"*")
275 LPRINT TAB(20)"您的身高";A;"(厘米),";
280 LPRINT TAB(53)"您的体型健美标准应是:"
285 LPRINT TAB(20)"体重:  ";B1;"(公斤)";
290 LPRINT TAB(40)A2$":";B2;"(厘米)"
295 LPRINT TAB(60)A3$":  ";B3;"(厘米)"
305 LPRINT TAB(20)A4$":  ";B4;"(厘米)"
310 LPRINT TAB(40)A5$":";B5;"(厘米)"
315 LPRINT TAB(60)A6$":";B6;"(厘米)"
325 LPRINT TAB(20)A7$":";B7;"(厘米)"
330 LPRINT TAB(40)A8$":";B8;"(厘米)"
335 LPRINT TAB(60)A9$":";B9;"(厘米)"
345 LPRINT TAB(20)A10$":";B10;"(厘米)"
350 LPRINT TAB(40)A11$":";B11;"(厘米)"
355 LPRINT TAB(60)A12$":  5:8"
365 LPRINT
368 LPRINT TAB(20)"您体型的实际情况是"
370 LPRINT TAB(20)"体重:  ";A1;"公斤"
375 LPRINT TAB(40)A2$":";A2;"(厘米)"
380 LPRINT TAB(60)A3$":";A3;"(厘米)"
390 LPRINT TAB(20)A4$":";A4;"(厘米)"
392 LPRINT TAB(40)A5$":";A5;"(厘米)"
395 LPRINT TAB(60)A6$":";A6;"(厘米)"
400 LPRINT TAB(20)A7$":";A7;"(厘米)"
405 LPRINT TAB(40)A8$":";A8;"(厘米)"
410 LPRINT TAB(60)A9$":";A9;"(厘米)"
420 LPRINT TAB(20)A10$":";A10;"(厘米)"
425 LPRINT TAB(40)A11$":";A11;"(厘米)"
435 LPRINT TAB(60)A12$":5;"USING  "#"X
440 LPRINT TAB(18);STRING$(62,"-")
455 LPRINT TAB(43)"测  量  方  法"
470 LPRINT TAB(18)"1. 胸围:由腋下沿胸上方最丰满处测量";
475 LPRINT TAB(54)"7. 颈围:在中部颈最细处"
480 LPRINT TAB(18)"2. 上臂围:在肩关节与肘之间的中部关节";
485 LPRINT TAB(54)"8. 足颈围:在足颈的最细部位"
500 LPRINT TAB(18)"3. 髋围:在体前耻骨平行于臀部最大部位";
510 LPRINT TAB(54)"9. 手腕围:在手腕的最细部位"
520 LPRINT TAB(18)"4. 腰围:在正常情况下,腰的最细部位";
525 LPRINT TAB(54)"10. 肩宽:两肩峰之间的距离"
530 LPRINT TAB(18)"5. 大腿围:在大腿的最上部位,臀
```

```
折线下";
    540 LPRINT TAB(54)"11. 上,下身比例:以肚脐为界"
    550 LPRINT TAB(18)"6. 小腿围:在小腿最丰满处"
    560 LPRINT TAB(54)"12. 脐高:站立时肚脐至地面高"
    610 LPRINT TAB(18);STRING$(62,"*")
    630 PRINT:PRINT"是否继续咨询(Y/N)";
    640 INPUT C$
    645 IF C$="y" OR C$="Y" THEN GOTO 5
    650 END
```

使清屏画面变美妙

邯郸市财政局　王存

很多应用软件,都颇为重视屏幕静态画面的设计,却往往忽视屏幕动态画面的制作。象属于屏幕动态画面之列的清屏过程等,设计者却总习惯采用单一的黑底色将屏幕一清了之,很少考虑清屏过程潜在的积极意义。其实,清屏画面设计得优与劣,不仅体现软件设计水平的高与低,也直接影响软件的应用价值,关联用户的上机情趣。

为了提高应用软件的自身价值,使用户对所用软件一见则喜,百用不烦,始终感到耳目一新,兴味盎然,就需要考虑将清屏过程加以改造,使其变得图案新颖别致,设色鲜艳明丽,节奏欢快和谐。下面给出的几种清屏程序,是我们在实践中创造的几个代表作。各程序都很短小,但清屏效果很好,其特色可概述如下:

程序一:挂帘垂落　一幅五彩挂帘随着悦耳的声响逐渐垂落,把原先的屏幕内容清除与覆盖。

程序二:帷幕衔启　象戏剧换场一样,伴着音乐节奏,漂亮华丽的帷幕先从两侧合拢一起,继而又被拉开,形成一种可供推出新的屏幕内容的环境。

程序三:百叶窗关合　模拟百叶窗徐徐关合之过程,以达到清除原屏幕内容之目的。

程序四:激光万束　利用产生随机数的函数,生成激光辐射般的万道光束。这些光束自左上向右下照射,参差不齐,错落有致,绚丽多彩,耀眼夺目。

程序五:凤凰展翅　斑斓的凤凰,在烈火中诞生。翎翅,象朝霞殷红;毛羽,似晨晕金黄。当她凌空飞翔之时,旧的屏幕内容便被遮蔽。

程序六:彩蝶翩翩　似在花丛中翻飞,又象在百草间飘旋,翩翩起舞的花蝴蝶舒展艳丽的翅膀,遮掩屏幕上原有的一切。

程序七:碧波涟漪　一泓碧水,明澈如镜。突然,一串石子从天而降,把水面的平静砸得破碎。溅落处,水翻涡旋,波声清脆,美丽的涟漪涌向四围。

程序八:红旗漫卷　红艳艳的旗帜,迎风飘扬;金灿灿的旗穗,飞舞飘荡;《国旗国旗真美丽》的歌声,优美亲切,给人力量。

程序九:土星光环　模拟宇宙微粒排斥与吸引的旋转运动,在屏幕上展现了奇异灿烂的土星光环及土星本身的生成过程。

程序十:星云变幻　如同宇宙中进行不规则运动的网状星云,运动物质以光条形式从几个不同方位同时交织叠加,逐步弥漫膨胀而演变成一幅色彩纷呈,结构奇特的星云变幻图。

本文所述程序可作为 BASIC 子程序,套用时,将它们插入到应用程序中需改变屏幕画面的地方。

程序一

```
10 CLS:KEY OFF:SCREEN 2,0
20 I=3:Y=0:FOR J=0 TO 37:IF 6<I THEN I=0
30 PLAY"mb t250 o4 124fcd"
40 FOR K=0 TO 300:NEXT K:I=I+1
50 LINE(9,Y)-(630,Y+10),I,BF:Y=Y+12
60 LINE(9,Y-2)-(630,Y),0,BF:NEXT J
```

程序二

```
10 CLS:KEY OFF:SCREEN 2,0:I=4
20 X=0:FOR J=0 TO 19:IF 7<I THEN I=I-7
30 LINE(X,1)-(X+15,450),I,BF
40 LINE(640-X,1)-(625-X,450),I,BF:I=I+1
50 X=X+16:PLAY"mb t180 o3 124eg":NEXT J
60 FOR X=-7000 TO 0:NEXT X:FOR J=0 TO 20
70 LINE(320+X,1)-(305+X,450),0,BF
80 LINE(320-X,1)-(305-X,450),0,BF
90 X=X+16:PLAY"mb t180 o4 112ac":NEXT J
```

程序三

```
10 CLS:KEY OFF:SCREEN 2,0
20 X=635:Y=0:FOR J=1 TO 28:LINE(5,Y)-(X,Y),
3
30 LINE(5,Y+90)-(X,Y+90),4:LINE(5,Y+180)-
(X,Y+180),5
40 LINE(5,Y+270)-(X,Y+270),6:LINE(5,Y+360)-
(X,Y+360),2
50 PLAY"mb t150 o4 124ef":Y=Y+3:NEXT J
```

程序四

```
10 CLS:KEY OFF:SCREEN 2,0
20 I=0:FOR J=0 TO 120:IF 15<I THEN I=1
30 LINE-(RND*639,RND*450),I
40 LINE(RND*777,RND*444)-(RND+999,RND-
333),I
50 I=I+1:PLAY"mb t150 o4 112ab":NEXT J
```

程序五

```
10 CLS:KEY OFF:SCREEN 2,0
20 X=40:Y=-1000:FOR J=0 TO 47
```

```
30 LINE(X,Y)-(640-X,450-Y),4:Y=Y+50
40 PLAY"mb t150 o4 l12cd":NEXT J
50 X=100:Y=-1000:FOR J=0 TO 47
60 LINE(X,Y)-(640-X,450-Y),6:Y=Y+50
70 PLAY"mb t150 o4 l12gf":NEXT J
```

程序六

```
10 CLS:KEY OFF:SCREEN 2,0:Y=25
20 FOR J=3 TO 83:LINE(620,450-Y)-(20,Y),J
30 Y=Y+5:PLAY"mb t180 o4 l12d":NEXT J
40 CIRCLE(320,225),150,4,,6.28,6
50 PLAY"mb t120 o2 l12be-fag-d-"
60 Y=-70:PAINT(320,135),4
70 FOR J=0 TO 59:K=INT(J/12+2)
80 LINE(200,450-Y)-(440,Y),K:Y=Y+10
90 PLAY"mb t180 o4 l24f":NEXT J
```

程序七

```
10 CLS:KEY OFF:SCREEN 2,0
20 LINE(0,0)-(640,450),3,BF:X=3
30 FOR J=0 TO 26:CIRCLE(320,225),X,7,,,.3
40 PLAY"mb t150 o3 l12df":Y=X+6
50 CIRCLE(320,225),Y,1,,,.3:X=X+12
60 IF J-4*INT(J/4)=0 THEN 90
70 IF J-4*INT(J/4)=2 THEN 100
80 PAINT(320,225),7:GOTO 110
90 PAINT(320,225),3:GOTO 110
100 PAINT(320,225),1
110 PLAY"mb t120 o4 l24gae"
120 CIRCLE(320,225),J:NEXT J
```

程序八

```
10 CLS:KEY OFF:SCREEN 2,0:P=6.28
20 PLAY"mb t120 o3l4gegeo4co3al2g l4ecegel2d"
30 DRAW"BM-320,-224:C7 E10 F10 G10 H10"
40 CIRCLE(14,27),6,7,,.3,P:PAINT(14,27),7
50 LINE(9,39)-(19,450),6,BF
60 LINE(20,42)-(20,432),4:PLAY"l3el6dl4cdegl2a"
70 LINE(20,42)-(320,62),4
80 LINE(20,432)-(320,450),4
90 LINE(320,62)-(630,32),4
100 LINE(320,450)-(630,420),4
110 LINE(630,30)-(600,225),4
120 LINE(600,225)-(630,420),4:PAINT(30,43),4
130 PLAY"l8ageal2gl4gl8del2c"
140 LINE(18,39)-(605,60),6
150 LINE(605,55)-(635,65),2,BF
```

程序九

```
10 CLS:KEY OFF :SCREEN 2,0
20 X=RND*110:Y=RND*255:FOR J=5 TO 31
30 CIRCLE(320,225),X,J,,6.28,.7
40 X=RND*318:PLAY"mb t150 o4 l12dae"
50 CIRCLE(320,225),Y,J,,6.28,.7
60 Y=RND*193:NEXT J:PAINT(320,225),6
70 PLAY"mb t150 o2 l12cdefl1g-l8a"
```

程序十

```
10 CLS:KEY OFF:SCREEN 2,0:X=10
20 FOR J=0 TO 80:LINE(X,X)-(X-640,X-450),6
30 LINE(640-X,450-X)-(X,X),4
40 LINE(X,900-X)-(640-X,X-450),3
50 LINE(320+X,X)-(640--X,X-225),5
60 LINE(X-400,X)-(X,X-600),2:X=X+10
70 PLAY"mb t150 o2 l24agd":NEXT J
```

技 巧 二 则

广空 汉神

一、怎样使您的文件隐含或显示？

亲爱的读者，您大概会有这样一种感受吧：每当您编写一个程序或编辑一篇信息文件时，总想使自己的成果或信息文件不被其它人操作（如列文件目录<DIR>、查看文件内容<TYPE>以及拷贝文件<COPY>等）。为了达到此目的，请您不妨试试把文件隐含起来，也许会收到意想不到的效果。运行下面一小段程序就可以把文件隐含或显示。此程序在 IBM-PC/XT 机上运行通过。

说明：

一、本程序是用汇编语言编写（文件名 HAN. ASM），因此，首先必须经过宏汇编（MASM）和连接程序（LINK）后，产生 HAN. EXE 可执行文件，才能直接运行。

二、程序的 12 行至 17 行是提示"请输入您的文件名"。输入您要隐含或显示的文件名后，打回车；第 22 行至 26 行显示"是稳含还是显示"。如要隐含打"Y"，显示打"N"。

三、运行此程序，相当于置一个隐含/显示软开关。

```
1;CODE          SEGMENT PUBLIC
2;              ASSUME  CS:CODE,DS:CODE
3;              ORG  100H
4;START;        JMP  BEGIN
5;BUFFER        DB 15
6;              DB  ?
7;              DB 15 DUP(?)
8;FILENAME      DB'PLEASE INPUT YOUR FILE NAME:
                    $'
9;HIDEDISP      DB 0DH, 0AH,' HIDE ENTER "Y", DIS
                    PLAY "N", $'
10;BEGIN;       MOV AX,CS
11;             MOV DS,AX
12;             MOV DX,OFFSET FILENAME
13;             MOV AH,09H
14;             INT 21H
15;             MOV AH,0AH
```

```
16:             MOV DX,OFFSET BUFFER
17:             INT 21H
18:             MOV BL,BUFFER+1
19:             ADD BL,2
20:             MOV BH,0
21:             MOV BUFFER [BX],0
22:             MOV AH,09H
23:             MOV DX,OFFSET HIDEDISP
24:             INT 21H
25:             MOV AH,01
26:             INT 21H
27:             CMP AL,59H
28:             JZ A1
29:             CMP AL,79H
30:             JZ A1
31:             MOV CX,00H
32:             JMP A2
33:A1:          MOV CX,02H
34:A2:          MOV AX,4301H
35:             MOV DX,OFFSET BUFFER+2
36:             INT 21H
37:             MOV AH,4CH
38:             INT 21H
39:CODE         ENDS
40:             END START
```

二、怎样给磁盘加上汉字卷标?

我国于1983年推出专为IBM-PC机开发的汉字操作系统CCDOS,使得IBM-PC及其兼容机具备了汉字功能,从而在我国得到广泛使用。通常在机上由键盘输入汉字,并在屏幕上显示出来。实际上这是CCDOS把由键盘输入的汉字输入码(如区位码、拼音码等)转换成汉字机内码(如异形国标码),并将汉字的机内码转换成汉字字模输出显示的过程。汉字的机内码是用国标码最高位置1表示(称为异形国标码),以区别于一般的ASCII码,故一个汉字的机内码用两字节表示。

为了得到汉字卷标,首先我们必须得到汉字的机内码。一般情况下我们采用汉字编辑软件WS编辑一份所需的汉字信息文件,然后用DEBUG把汉字信息文件调入内存。用D命令显示出来的数据即为汉字的机内码(每两个字节为一个汉字机内码)。

修改磁盘卷标信息的操作,可分以下三种情况:

1. 磁盘在格式化时用FORMAT A:/V↙

```
C>DEBUG↙
-L100 0 5 1↙
-E100↙
CE C4 CF C8 C9 FA 30 31 C5 CC 20 08 20 20 20 20↙
-W100 0 5 1 ↙
-Q↙
```

2. 磁盘在格式化时用FORMAT A:/S/V↙

```
C>DEBUG↙
-L100 0 5 1↙
-E160↙
CE C4 CF C8 C9 FA 30 31 C5 CC 20 08 20 20 20 20↙
-W100 0 5 1↙
-Q↙
```

3. 磁盘在格式化时用FORMAT A:↙

```
C>DEBUG↙
-L100 0 5 7↙
-D↙
```

此时用D命令来查看目录,如果盘上的文件数目不超过112个,我们就在最后一个文件目录的地址后面加上卷标。假设盘上有50个文件,那么第51个文件目录的地址为:0B00

```
-E0B00↙
CE C4 CF C8 C9 FA 30 31 C5 CC 20 20 08 20 20 20 20
↙
-W100 0 5 7 ↙
-Q↙
```

用DIR查看目录时,就显示:

```
C>DIR A:↙
Volume in drive A is 文先生 01 盘
Directory of A:\↙
COMMAND   COM     17664   3-08-83    12:00P
FORMAT    COM      6016   3-08-83    12:00P
CHKDSK    COM      6400   3-08-83    12:00P
DEBUG     COM     11904   3-08-83    12:00P
        4  File(s)      318464 bytes free
```

如果是硬盘,上述的L、W命令分别改为:L100 2 11 10↙,W100 2 11 10↙,其余步骤不变。

利用DEBUG进行反汇编存盘的方法

浙江电子工业学校　金林樵

用DEBUG的U命令,可方便地反汇编出源程序的清单,但它只能显示和打印,却不能进行反汇编存盘操作,不便于程序的修改和再加工。下面介绍的方法,将为你提供这方面的帮助。

本方法利用DOS系统的输入输出重新定向功能,通过可变元的批处理程序(见程序一),将需反汇编源程序的文件名(包括扩展名)、反汇编起始地址、终止地址及一个退出DEBUG的q命令输出到T.TXT文件中,然后调用DEBUG将需汇编程序从起始地址到终止地址的反汇编清单,通过输出重新定向功能输出到输出源文件名存盘,以后就可利用字处理软件对其输出文件进行修改了,使用十分方便。使用方法如下:

A>UASM 需反汇编文件名　起始地址　终止地址　输出源文件名

本方法在IBM PC/XT机上调试通过。

A>type uasm.bat

```
ECHO OFF
ECHO U %2 %3 > T. TXT
ECHO Q >> T. TXT
DEBUG %1 < T. TXT > %4
DEL T. TXT
```

90年全国计算机通讯赛题分析

西安交大附中　汪五一

1990年全国青少年计算机通讯赛的试题是一道模拟魔方转动的题目。下面就该题的算法作一探讨(试题详载于本刊91年第1期,因太长,此处从略)。

综观全题,主要解决两大方面的问题:1.输入魔方的初始状态,并进行判错处理。2.若魔方的初态合理,则要求以最少步骤达到目标状态。

判错处理应包括有检查所输入的初始状态是否包含1～9,A～I,a～i 27个字符。另外根据魔方转动的特点,还应检查E小块是否在魔方的中心,即应在题目给定的编号点14上;1、3、7、9、a、c、g和i这8小块应落在魔方的8个角上的位置,即编号点1、3、7、9、19、21、25、27上;5、B、D、F、H和e这6小块应在魔方6个可见面的中心,即在编号点5、11、13、15、17和23位置上;其余的各小块应落在魔方其它点位置上。凡上述判断中只要有一个不符合要求,则应作出错处理。这一段检查出错处理程序较简单。

魔方从初始状态到目标状态的移动过程,实质就是从初始态不断按题目要求进行转动搜索,最终找到结果。由于魔方从当前状态转到下一状态有18种不同的转法,因此可直接采用广度优先搜索方法。把初始状态作为一个根结点,往下生成18个结点,再由这18个结点出发,再分别生成其它结点……这样边生成结点,边检查所生成的结点是否到达目标状态,直到目标态为止。这种盲目搜索方法必须把所生成的结点保存在数组中。若从初始状态到目标状态最少要转N次,则最少需要生成$18^{N-1}+1$个结点,最多则要生成18^N个结点才能达到目标。这对APPLE Ⅱ类微机,从初始态距目标态的步数在一、二步还可行。但三步以上的搜索,就很快由于记录的结点过多,而造成溢出的错误。

本题也可用人工智能中的A*算法来处理(1990年第二届国际信息学奥林匹克竞赛中,我国4名选手均采用该算法求解第一道赛题,有关算法可参见《学生计算机世界》报总143期或其它有关人工智能方面的书籍)。这种算法主要采用了一个启发函数来估价哪个结点最有希望达到目标,从而优先扩展该结点。因此它仍需要建立数组来记录已扩展的结点。另外这个启发函数要构造得比较准确,才能节约内存,以最快速度达到目标。但对本题由于小方块的转动不是独立的,转动某小块将涉及到该片其它小块也跟着一起转动,要建立一个较好的启发函数比较困难。正由于这个原因,就不能准确地优先发展能到达目标的结点。故用A*算法对本题所能解决的魔方移动步数也极为有限。

其实本题可用解答树的方法来处理。其最大特点是可用最少的内存空间来处理任意步数(原则上可以这么说)的魔方转动,但它也有不足之处,就是这种方法是以时间来换取空间的。有关解答树在数据结构书籍中均有介绍,现简述如下:

求解解答树的过程是:1.构造解答树;2.系统地检查解答树的结点,看它们是否对应于解答位置(所谓解答位置就是符合题目要求的结点)。

为了加快第2步中检查结点的速度,通常约定:任何一棵子树,只要它的根不满足条件,它就不会含有解答位置,因而不必继续考虑它,并且在第1步构造解答树时就不构造这种子树。

由于所构造的解答树的规模通常很大,因此把第2步对结点的检查与第一步构造解答树的工作并行进行,在逐步构造和检查解答树的过程中,对已处理过的结点,若以后不会再用,就不必保留它。

如果逐步构造和检查解答树的工作是从左到右逐步进行,那末一般来说,为检查长度为n的树枝T,只要保留n个结点就够了。在T的左边的一切结点都已检查过了;T的右边的一切结点尚未产生。按这种方式得到一条到达树叶的树枝T后,得到T右边的下一个树枝的方法是:沿T回溯到第一个尚未用过的向右的分支点,并由此向右走一步,随后仍然向左走,直至到达树叶为止。

因为本题要求以最少的步数转到目标状态,所以构造树的层数应以N=2开始搜索。若1步转动不能达到目标,则应构造N=3的树,即做2步转动试探,以此类推,直到符合目标要求为止。构造检查本题解答树的程序逻辑及BASIC程序如下:

1.输入魔方初始状态作为根结点。
2.设置生成树层数N初值为2。
3.设置堆栈指针I1为1。
4.栈指针I1等于树层数N时转去12。
5.置S=0。
6.S=S+1生成下一个扩展结点号。
7.若当前结点已扩展了18个结点则转12。
8.将当前结点状态及扩展的枝叶号S存入栈,栈指针I1加1。
9.根据S值扩展生成一个结点。
10.若生成的新结点未达到目标态则返回4。
11.完成魔方转动。
12.栈指针I1减1。
13.若已下越栈底,生成树层次N加1后返回3。
14.从栈中退出一个结点状态及S值,返回6。

```
1 CLEAR
5 DIM G$(50),G(50),E(18),D(9,5),F(10,2),C$(12):REM 数组 G$,G 作为堆栈,存放魔方转动的各状态及转动的方向,数组 E,D,F 为魔方转动的位置数据,数组 C$为存放判错所用数据
10 INPUT "INPUT A STRING:";A$:IF LEN (A$)<>27 THEN 10:REM 判断输入的初始态字符个数是否为 27 个
20 IF MID$(A$,14,1)<>"E" THEN PRINT "WRONG!":GOTO 1:REM 判断 E 是否在魔方中心
25 FOR Q=1 TO 8:READ C$(Q):NEST Q:DATA 1,3,7,9,a,c,g,i
30 FOR I=1 TO 8:READ J:R$=MID$(A$,J,1)
35 FOR Q=1 TO 8:IF R$=C$(Q) THEN C$(Q)="":GOTO 45
40 NEXT Q:PRINT "WRONG!":GOTO 1
45 NEXT I:DATA1,3,7,9,19,21,25,27:REM 判断 1,3,7,9,a,c,g,i 是否在魔方的 8 个角上
50 FOR Q=1 TO 6:READ C$(Q):NEXT Q:DATA 5,B,D,F,H,e
55 FOR I=1 TO 6:READ J:R$=MID$(A$,J,1)
60 FOR Q=1 TO 8:IF R$=C$(Q)THEN C$(Q)="":GOTO 75
65 NEXT Q:PRINT "WRONG!":GOTO 1
75 NEXT I:DATA 5,11,13,15,17,23:REM 判断 5,B,D,F,H,e 是否在魔方可见面的中心
80 FOR Q=1 TO 12:READ C$(Q):NEXT Q:DATA 2,4,6,8,A,C,G,I,b,d,f,h
90 FOR I=1 TO 12:READ J:R$=MID$(A$,J,1)
100 FOR Q=1 TO 12: IF R$=C$(Q) THEN C$(Q)="":GOTO 110
105 NEXT Q:PRINT "WRONG!":GOTO 1
110 NEXT I:DATA 2,4,6,8,10,12,16,18,20,22,24,26
111 REM 以上同时判断输入小方块的标志是否正确
120 B$="123456789ABCDEFGHIabcdefghi":REM B$存放目标状态
130 FOR I=0 TO 5:FOR J=1 TO 9:READ D(J,I):NEXT J:NEXT I
140 DATA 7,4,1,8,5,2,9,6,3,3,6,9,2,5,8,1,4,7
150 DATA 19,10,1,20,11,2,21,12,3,3,12,21,2,11,20,1,10,19
160 DATA 7,16,25,4,13,22,1,10,19,19,10,1,22,13,4,25,16,7
170 FOR I=1 TO 18:READ E(I):NEXT I:DATA 0,9,18,0,9,18,0,3,6,0,3,6,0,1,2,0,1,2
180 FOR I=0 TO 2:FOR J=1 TO 9:READ F(J,I):NEXT J:NEXT I:DATA 1,2,3,4,5,6,7,8,9,1,2,3,10,11,12,19,20,21,1,4,7,10,13,16,19,22,25:REM 数组 D,E,F 读入魔方转动的位置数据
200 IF A$=B$ THEN PRINT "RIGHT!":GO TO 380:REM 判断初始态是否为目标态
210 N=2:REM 设置树层数 N 的初值
220 I1=1:REM 设置栈指针初值
240 IF I1=N THEN 400:REM 栈指针等于树层数 N,则转 400 进行回溯处理
250 S=0
260 S=S+1:IF S>18 THEN 400:REM 判断当前结点是否已扩展了 18 个树叶
270 GOSUB 2000:I1=I1+1:REM 当前结点及将扩展的叶子号 S 入栈,栈指针加 1
275 J=1:R$=""
280 FOR I=1 TO 27
285 R=F(J,INT((S-1)/6))+E(S)
290 IF I=R THEN RE$=MID$(A$,D(J,INT((S-1)/3))+E(S),1):J=J+1:GOTO 305
300 RE$=MID$(A$,I,1)
305 R$=R$+RE$:NEXT I:REM 魔方某片实现转动
310 A$=R$
315 IF A$<>B$ THEN 240:REM 魔方未到目标,继续处理
320 GOSUB 2000:REM 魔方旋转成功,打印处理
329 REM 打印魔方转动过程
330 FOR Q=1 TO I1
340 IF Q=1 THEN PRINT "START:";:GOTO 360
350 PRINT "STEP";Q-1;":";
360 PRINT G$(Q)
370 NEXT Q:PRINT "END"
380 END
399 REM 回溯处理
400 I1=I1-1:IF I1=0 THEN N=N+1:GOTO 220:REM 栈指针减 1,判断是否到栈底
410 A$=G$(I1):S=G(I1)
420 GOTO 260:REM 结点状态及叶子号 S 出栈,转 260,扩展其它枝叶
1999 REM 结点及 S 入栈子程序
2000 G$(I1)=A$:G(I1)=S:RETURN
]RUN
INPUT A STRING:3G12FCahid58HEB4efg69IDA7bc
START:3G12FCahid58HEB4efg69IDA7bc
STEP1:36125CaGidD8HEB4Ffgb9IeA7hc
STEP2:369258aG1dDAHEB4FCgbcIef7hi
STEP3:a23G56189dDAHEB4FCgbcIef7hi
STEP4:123456789GDAHEBIFCabcdefghi
STEP5:123456789ABCDEFGHIabcdefghi
END
```

但这个程序仍有不足之处,需要大量的搜索时间。从以上分析可看出其探索时间是按几何级数递增的。若有一个需转 n 步才能到达目标的初始态,程序最多运行时间为 $T=t(1+18+18^2+\cdots\cdots+18^n)=t(\frac{18^n-1}{18-1})$。其中 t 为从一个结点出发,生成 18 个树叶所需的时间。本程序在 APPLE 机上测试 t 约为 20 秒。这样一个需转动五步才能达到目标的初始状态,最多需 $20(\frac{18^5-1}{18-1})=2223021$ 秒≈26 天才能完成。

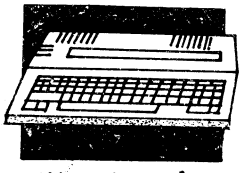
改进措施设想:

1. 从初态开始,除第一步有 18 种转动方法外,以后每转一步,从该步出发,虽也有 18 种转法,但其中有

一转法又转回到原状态。所以除第一步需扩展 18 个枝叶外，以后的结点只需扩展 17 个枝叶。

2. 由于解答树是按前序生成及搜索的，因此根据题目要求应先生成一棵层次 N 为 2 的树，看其能否达到目标。若不成，然后重新生成一棵层次 N 为 3 的树……这样就造成了搜索时间是按几何级数递增。若能构造一个估价函数，大概估价出魔方需 N 步达到目标，那么一开始就向 N+1 层树方向生成与搜索。若所生成的 N+1 层树仍未达到目标，就再用估价函数，估价出所生成的树叶离目标最少还需 n 步，下次就生成一棵 N+1+n 层树，再进行搜索。经这样处理也可以加快程序执行速度。

以上讨论不一定正确，希望各位读者提出更为优良的算法。



CEC-I 全功能造字

成都市成人教育学院　傅叔平

学习机之友

CEC-I 的字库包含了国标一、二级汉字 6763 个和一些外文字母和常用符号，通常情况下已够使用。有时候遇到一些冷僻的汉字或在有的场合需用一些特殊符号，这就需要造字了。但 CEC-I 的汉字管理程序和汉字库都是固化的，不能改变，这就给造字（特别是在打印输出中造字）带来了较大的困难。本文要解决的问题是要在显示和打印两种输出方式中均能造字并且不影响原有的功能。

一、实现方法

1. 显示造字

CEC-I 的汉字显示用的是高分辨率图形页。汉字系统的输出程序根据汉字或字符的代码在汉字库中查到点阵，再将点阵按高分率图形屏幕上点与其映射区地址的对应关系，将点阵数据置入合适的地址，便能在屏幕上显示出汉字来。问题是固化的汉字系统的输出程序只会从系统字库中查找点阵，没法加以改变让它接受用户提供的点阵数据。所以输出自造的字时不能直接调用汉字系统的输出程序，可采用如下较为简单的方法来解决：

在屏幕上显示用户造的字时不用汉字系统的输出程序，而直接调用系统内部的显示子程序来显示，这样就可以由用户程序将所需字的点阵直接送到系统的点阵缓冲区供显示之用。

2. 打印造字

当考虑还要在打印输出中造字时问题就复杂得多了，因为九针打印机打印 16×16 点阵的汉字时，不能象显示输出那样以一个汉字或字符为单位，必须以一个打印行为单位。造的字的点阵必须排在打印队列中其它字的点阵之间，还要考虑九针打印机打印一行汉字时必须分为上半行和下半行两次进行，还有字型、字间距、行间距等等一系列问题。

为了能将造的字插入其它的字中间打印出来，在显示时每显示一个字便将该字（包括造的字）的代码排入一个输出队列，打印时有如下几个步骤：

①根据输出队列的长度、字型和字间距计算出打印所需的位图形数据的个数，并将打印机置成图形打印状态；

②到输出队列中依次取代码，若代码属国标一、二级汉字或 ASCII 字符，仍到汉字系统的固化字库中取点阵，若是用户造字则到用户字库中取点阵；

③取完输出队列的点阵后打印出上半行；

④控制行距为 0 换行；

⑤类似①～③的步骤打印出下半行；

⑥按行间距走纸。

从上述看要作的工作是很多的，特别是打印输出，和重写一个汉字打印驱动程序也差不了多少。明智的作法是利用固化的汉字管理程序这个丰富的资源，直接调用其中有用的子程序给我们提供便利，这样可以减少编程工作量并使程序大大地缩短。

二、程序注释

程序 1 是按上述方法编制的一个功能较完善的造字程序，它既能在屏幕上显示用户造的字，也能打印出包括用户造的字在内的输出内容，其字型、字距、行距等仍用原来的命令决定。

该程序调用的汉字系统内部子程序主要有：

$C392——内码到国码转换；

$ECCD——查找系统字库中的点阵，调用前须将汉字的国标码高字节放入 $D6 单元，低字节放入 $D7 单元，若是 ASCII 字符则 $D6 单元置 0，$D7 单元放它的 ASCII 码；

$F0B5——显示一个汉字，显示前须将汉字的 16×16 点阵数据放入 $94D0 开始的点阵缓冲区；

$F0AD——显示 16×16 点阵的左半边字形。当显示 ASCII 字符时，就要用到该子程序来显示，若用前一个显示子程序则字符后将跟一个空格；

$F0E5——根据一个打印行的字数及字型和字间距计算出一个打印行的位图形数据个数；

$F630——将打印机设置成位图形打印方式；

$F66B——打印完上半行汉字后走纸。

程序 1 中 $6091～$60B9 是显示子程序，它在显示的同时也把字的代码排入输出队列。$60BA～$610A 是打印子程序。$610B～$6145 是取点阵子程序。为统一起见，用户造的字被定义为区位码表中的一个未定义区——10 区中的字。该子程序首先判断区

码是否 10(相应的国标码 $2A),若是则到用户字库区查找点阵。在将点阵送入点阵缓冲区之前还有一个工作要做,就是为了和系统字库的点阵数据规律一致,须将点阵数据的每个字节的各位左右颠倒,这由 $6146～$6152 的子程序来完成。

造字程序由 & 命令的入口进入,程序中 $6000～$6034 即是解释执行新定义的几个 & 命令。

三、应用

读者可在监控状态下输入程序 1,然后存盘待用。使用时应先将程序 1 和用户字库调入内存。用户字库起始地址为 $7000,每 32 个字节为一个字的点阵数据,即 $7000～$701F 为用户定义的 1 号字,$7020～$703F 为用户定义的 2 号字,依此类推。然后设置 & 命令的入口向量,使其指向 $6000。

造字程序由如下新定义的 BASIC 语句来调用:

①&C——清除打印队列,在初次打印之前或打印新内容之前须用此语句;

②&"<汉字串>"——显示汉字串前将其代码加入打印队列;

③&n——显示第 n 个用户造字前将其代码加入打印队列,n 对应用户字库区的第 n 个自造字;

④&p——将②、③两种命令形成的输出队列打印出来,和 PRINT 语句一样,字型、字距行距等的确定仍用原来的命令,若 1659 单元的值为 0 则不打印,此时此命令无效。在未用命令①清除打印队列之前,可多次使用此命令重复打印同一内容。

从上述可知,若要打印自造字,必须先显示再打印;若只显示则不用打印命令。另外,PRINT 语句的功能照旧,所以,若打印行中没有用户造的字,仍用 PRINT 语句输出即可。当然,PRINT 语句也可和这些新定义的 BASIC 语句混合使用。

程序 2 是调用造字程序的实例,运行结果中第一行字是由 50 行的 PRINT 语句输出的,第二行字是由 60 行输出的,其中第四个字(木旁右边一个童字)不但国标一、二级汉字中没有,就连《新华字典》上也查不到。第三行字中用 16×16 点阵造了一个符号,是由 70 行输出的。

程序 2 的 20 行中"CREATE. CHN. OBJ0"是程序 1 在磁盘上的文件名;30 行中的"CREATE. CHN. LIB"是用户字库在磁盘上的文件名。用户字库如下:

```
7000- 10 80 10 48 17 FC 11 10
7008- FC A4 17 FE 30 08 3B FC
7010- 52 48 53 F8 92 48 13 F8
7018- 10 40 13 F8 10 40 17 FE
7020- 01 00 01 00 01 00 01 00
7028- 03 80 03 80 03 80 07 C0
7030- 07 C0 27 C8 2F E8 2F E8
7038- 3F F8 3F F8 70 1C 80 02
```

程序 1

```
6000- A0 00 B1 B8 C9 43 D0 07
6008- A9 00 8D 53 61 F0 74 C9
6010- 22 F0 22 C9 50 F0 52 C9
6018- 30 90 6C C9 3A B0 68 20
6020- 7B DD 20 FB E6 8A 18 69
6028- 20 85 D7 A9 2A 85 D6 20
6030- 91 60 4C 86 60 20 8A 60
6038- A0 00 B1 B8 C9 22 F0 43
6040- C9 7F D0 19 20 8A 60 B1
6048- B8 20 92 C3 85 D6 20 8A
6050- 60 A0 00 B1 B8 20 92 C3
6058- 85 D7 4C 63 60 85 D7 A9
6060- 00 85 D6 20 91 60 4C 35
6068- 60 AD 7B 06 F0 03 20 BA
6070- 60 AD 7B 06 48 A9 00 8D
6078- 7B 06 A9 0D 20 2B C3 68
6080- 8D 7B 06 20 8A 60 60 4C
6088- C9 DE E6 B8 D0 02 E6 B9
6090- 60 AE 53 61 A5 D6 9D 55
6098- 61 E8 A5 D7 9D 55 61 E8
60A0- 8E 53 61 20 AB C3 20 0B
60A8- 61 A5 D6 F0 06 20 B5 F0
60B0- 4C B6 60 20 AD F0 20 B9
60B8- C3 60 AD 7B 06 F0 0C 20
60C0- AB C3 20 CC 60 20 8F F6
60C8- 20 B9 C3 60 A9 00 20 DA
60D0- 60 20 6B F6 A9 01 20 DA
60D8- 60 60 8D F5 94 AD 53 61
60E0- F0 28 8D F6 94 20 E0 F5
60E8- 20 30 F6 A2 00 EC 53 61
60F0- B0 18 BD 55 61 85 D6 E8
60F8- BD 55 61 85 D7 8A 48 20
6100- 0B 61 20 B7 F5 68 AA E8
6108- D0 E3 60 A5 D6 C9 2A F0
6110- 06 20 CD EC 4C 45 61 A9
6118- 00 85 06 A9 70 85 07 A5
6120- D7 38 E9 20 AA CA F0 10
6128- A5 06 18 69 20 85 06 A5
6130- 07 69 00 85 07 4C 25 61
6138- A0 1F B1 06 20 46 61 99
6140- D0 94 88 10 F5 60 A2 08
6148- 0A 6E 54 61 CA D0 F9 AD
6150- 54 61 60 A0 A0 C4
```

程序 2

```
1 REM 造字举例
5 B$ =CHR$ (13)+ CHR$ (4)+"BLOAD"
10 PRINT B$ ;"CREATE. CHN. OBJ0"
20 PRINT B$ ;"CREATE. CHN. LIB,A $7000"
30 POKE 1013,76:POKE 1014,0:POKE 1015,96
40 POKE 1659,6:POKE 1787,2:POKE 1915,6
50 PRINT "造字举例:"
60 & C;&"成都梓";&1 : & "桥街";& P
70 & C;& "您能打印出符号"; & 2;& "吗?";& P
90 POKE 1659,0
99 DEL 5,20:END
```

梵塔问题和奇数幻方的一行程序

北京大学　物理系　陈纲

编辑同志：

我是一名北大新生，是贵刊今年的新读者。感谢你们办出这么好的杂志，它已成了我学计算机的好帮手。现在冒昧地寄上两个一行程序，望多指教。

第一个程序解决了梵塔问题(即金片移动问题)。梵塔问题是计算机算法学中的一个老问题，历来是用递归方法解决的，而众所周知 BASIC 语言不易实现递归，即使用子程序自我调用的方法实现了，若金片太多也会造成堆栈溢出而出错，且不可能写成一行程序。本程序采用了一种非传统的算法，亦即将最小的金片按 A→B，B→C，C→A，(A、B、C 表示三根金针)的顺序移动，每移一步，比较一下剩下两根金针顶端的金片，能怎么移就怎么移，如此反复，移动 2^N-1 步(N 表示金片数)后必能将 N 片金片都移到另一根金针上。程序中用数组 X(N,2)存放三根针上的金片号码，用数组 A(2)存放三根针上的金片数目，并用逻辑代数的技巧，避免了使用 IF……THEN……语句，使此算法能在一行中实现。

第二个程序是一个打印奇数幻方程序。它采用了可能大家很熟悉的构造奇数幻方的数学方法，只是用了一些 SGN 函数以避免使用 IF……THEN……语句，使之能写在一行中。两个程序分别用两种不同的方法替代了 IF……THEN……语句，希望能对读者编写一行程序有些启发。

程序一

```
5 INPUT N:DIM X(N,2):FOR I=1 TO N:X(I,0)=I:NEXT:A(0)=N:FOR I=1 TO 2^N-1:F=E+1-(E)1)*3:A=X(A(E),E):B=X(A(F),F):M=(A)B)*E+(A(B)*F:F=F+E-M:PRINT"("I") move from "CHR$(65+M)" to "CHR$(65+F):A(F)=A(F)+1:X(A(F),F)=X(A(M),M):A(M)=A(M)-1:E=E+2-(E)0)*3:NEXT
]RUN
? 3
(1)  move from A to B
(2)  move from A to C
(3)  move from B to C
(4)  move from A to B
(5)  move from C to A
(6)  move from C to B
(7)  move from A to B
```

程序二

```
5 INPUT P:DIM P(P,P):X=(P+1)/2:Y=P:FOR I=1 TO P*P:P(X,Y)=I:M=X*SGN(P-X)+1:N=Y*SGN(P-Y)+1:A=SGN(P(M,N)):X=M+A*(X-M):Y=N+A*(Y-1-N):NEXT:FOR I=1 TO P:FOR J=1 TO P:PRINT RIGHT$("  "+STR$(P(J,I)),4);:NEXT:PRINT:NEXT
]RUN
? 5
   11   18   25    2    9
   10   12   19   21    3
    4    6   13   20   22
   23    5    7   14   16
   17   24    1    8   15
```

原稿还附有梵塔 N=5，幻方 P=31 的运行结果，限于篇幅从略　　编者。

(上接第 20 页)因为对顺序文件进行读写时，系统就把读写指针从该顺序文件的首字节向后移动了 p+1 个字节，并把这个位置作为顺序文件的首字节，前面的字节就无法读出或写入了，这样一来，读出的内容肯定不正确，写入的内容亦不能正确读出。

修正办法

修改 DOS 的命令执行过程，使系统先转去对 $AA6E，$AA6F 进行初始化置入 0，以确保消除 R_p 参数的遗留影响，修改步骤如下：

```
*A01B:20 C0 B6        JSR $B6C0
*B6C0:A2 00           LDX #$00
*B6C2:8E 6E AA        STX $AA6E
*B6C5:8E 6F AA        STX $AA6F
*B6C8:0E 5F AA        ASL $AA5F
*B6CB:60              RTS
```

然后用 INIT 命令格式化成一张具有修正 DOS 的系统盘。经过修改后 DOS 系统就不会有问题了。

更　正

本刊 1990 年 12 期《CEC 机汉字放大与缩小显示》一文中，由于笔者的疏忽，在程序三第 1000 行中遗漏了 POKE 8,A。正确的写法是：1000 A=2*LEN(A$)/3+4:POKE 8,A:POKE 230,64:CALL-3086:VTAB1:PRINT A$:RETURN。特更正，并向读者表示歉意。

苏州景范中学　赵旭

本刊 1991 年第 2 期《如何解决运行 FOXBASE 内存不够的问题》一文，由于本人不慎，致使第 34 行文字中的式子有误，应改为：

(141376 − 65536 * (INT (141376/65536)) = 2840H

谨向读者道歉

湖北天门市 121#信箱　周日初

APPLE-Ⅱ机打印格式的控制

无锡商业学校　张冠玉

APPLE机的荧屏每行显示40字符，而打印机一般都可印80字符。如果对打印机不加任何格式控制，那么打印机输出实际就是将荧屏显示原封不动的拷贝，仅利用了半边纸面。另外，我们时常要打印一些表格，要求有一定的宽度。因此，很有必要研究如何控制打印机的输出格式问题。

不同打印机都有自己的一套打印格式的命令，控制打印位置、行距、页长等。我们以83A—80型打印机为例，讨论80字符输出及行距等控制方法。

一、用TAB(X)函数输出80字符

TAB(X)函数是BASIC专为荧屏显示而设置的函数，无法直接在打印机上进行80字符输出，举例说明：

```
10 PRINT TAB(5)"A";TAB(25)"B"
;TAB( 45)"C";TAB( 65)"D"
RUN
    A                   B                   C
                        D
```

可以发现，字符A,B,C打印在同一行的预定位置，但字符D未能打在同一行的第65个位置，而是打在下一行的第25个位置。其原因是：若当前打印位置＞40，那么，在输出下一TAB(X)函数后面的打印项目内容时，计算机会从打印纸的右半边纸面的第一个位置（即第41个打印位置）开始，对自变量X进行计数，确定打印项的位置。若X≤40，则打印项的内容输出在同一行的右半边纸面；若X＞40，则打印项的内容是从下一行的第X－40个位置开始输出。

理解了上述原因，不难把上例加以修改：

```
10 PRINT TAB(5)"A";TAB(25)"B"
   ; TAB(41);TAB(45－40)"C";
   TAB(65－40)"D"
```

同样，不难推出打印80字符的格式：

〈行号〉 PRINT 〈打印内容组1〉;TAB(41);
TAB(X1－40)〈打印项1〉;…;TAB(X_n－40)〈打印项n〉

格式中自TAB(X1－40)以后为〈打印内容组2〉

说明：1.〈打印内容组1〉中最后一个字符的输出位置必须小于等于40。

2. $X1,\cdots,X_n$必须大于40，小于等于80。

二、用软控制开关输出80字符

APPLE机的打印机接口驱动软件利用某些地址单元的内容进行识别处理，来控制打印机的动作。我们称之为软控制开关。在这些软开关中，有一个地址1528＋槽口号（打印机通常插入1号槽，相应槽号为1），只要对此单元分别置入255或0，就可以控制打印机执行禁止换行或允许换行的动作，即：

```
POKE    1529,255    禁止换行
POKE    1529,0      复原
```

将软开关与TAB(X)函数联合使用，即可实现80字符输出。

格式：〈行号〉 POKE 1529,255
〈行号〉 PRINT TAB(X)〈打印项1〉
⋮ ⋮
〈行号〉 PRINT TAB(X)〈打印项N〉
〈行号〉 POKE 1529,0

使用这种格式时，一个PRINT语句只能含一个TAB(X)函数，且句尾不留分号或逗号，如：

```
10 POKE 1529,255
20 FOR I=0 TO 4
30 READ A$(I)
40 PRINT TAB(5+I*10)A$(I)
50 NEXT I
60 POKE 1529,0
70 DATA ####,$$$$,%%%%,****,
&&&&
```

这种格式的特点是使用方便，尤其适合于循环中的输出格式控制。

三、用控制码输出80字符

一般打印机都有ESC D与CHR$(9)这一对控制码。

ESC D规定横向打印时打印头的停留位置，ESC D后面最多能跟28个停留标号，停留位置为1～80。

CHR$(9)这个控制码必须与ESC D联合使用，它执行ESC D的规定。格式：

〈行号〉 PRINT CHR$(27);"D";
CHR$(n1);…;CHR$(n_k);CHR$(0)
〈行号〉 PRINT 〈打印项0〉;CHR$(9);
〈打印项1〉;CHR$(9);…;CHR$(9);
〈打印项K〉

说明：$n_1,\cdots,n_k$是规定打印头要停留的列位置，每一个CHR$(9)按顺序执行一个停留。例如：

```
10 PRINT CHR$(27);"D";CHR$(5);CHR$(25);CHR
   $(45);CHR$(65);CHR$(0);
20 PRINT "<";1;">";CHR$(9);"A";CHR$(9);"B";
   CHR$(9);"C";CHR$(9);"D"
30 PRINT "<";2;">";CHR$(9);"E";CHR$(9);"F";
   CHR$(9);"G";CHR$(9);"H"
```

四、打印机的其它常用控制码

1. 行距控制码

(1)ESC A＋(n)

设定行距为n/72英寸。1≤n≤85。

格式：〈行号〉 PRINT CHR$(27);"A";CHR$(n);

(2)ESC 0

设定行距为 1/8 英寸。

格式:〈行号〉PRINT CHR $(27);"0";

(3)ESC 1

设定行距为 7/72 英寸。

格式:〈行号〉PRINT CHR $(27);"1";

(4)ESC 2

设定行距为 1/6 英寸。

格式:〈行号〉PRINT CHR $(27);"2";

2. 页长控制码

ESC C+(n)

设定每页的长度,以行数为单位,每页最长不超过 127 行。

格式:〈行号〉PRINT CHR $(27);"C";CHR $(n);

3. 打印机复原码

ESC @

这个控制码使打印机复位、初始化,即设定页始端,页长 66 行,行宽 80 字符等。

将 BASIC 程序变成 B 型文件并以 BRUN 运行

北京师范学院数学系 87(3)班 杨峰

《电子与电脑》杂志曾介绍过将 BASIC 程序以 B 型文件存盘的方法,其缺点是无法以 BRUN 运行,虽起到保密作用,但调盘运行则比较繁。这里介绍一种可以把 BASIC 程序以 B 型文件存盘并且可以 BRUN 运行的方法。

首先介绍一个重要的入口——$A4FC,这是 DOS 运行 BASIC 程序即 RUN 的执行入口。我们可以在 BASIC 程序内存区($800)以前存入一段机器码程序,用以设定 BASIC 简单变量区及指针,然后 JMP $A4FC 来运行 BASIC 程序。

详细操作过程如下:

1. 键入或调入一个 BASIC 程序到内存区

2. CALL-151↙进入监控

3. AF. B0↙查看程序尾指针,设 AF 内容为 LL,设 B0 内容为 HH,计算未来程序长度:

长度=$HHLL-$0777

4. ESC @↙ 清屏,使光标回左上角

5. 键入以下机器码程序:

*778:A9 LL 85 69 4C F8 7 N 7F8:A9 HH 85 6A 4C FC A4

6. 用 BSAVE 存盘:BSAVE〈文件名〉,A $778,L $长度↙

好,现在你的 BASIC 程序以 B 型文件存盘了并且你可以试试是否可以 BRUN 运行呢?

如果在 BASIC 程序中加入 CTRL-C 和 RESET 保护,则以 B 型文件存盘后,本身已经是被加密的了,因为在 $800 以前的机器码处于文本显示区以外部分,不用特殊工具是无法看到其内容的。

APPLE DOS 的一处错误

重庆市职业技术教育中心 尹进渝

错误现象

在用 READ f[,R_p][,B_b]或 WRITE f[,R_p][,B_b]访问了随机文件后,若接着对顺序文件用 READ f[,B_b]或 WRITE f[,B_b]进行带 B_b 参数的读或写时,将使 READ 后的第一个 GET 或 INPUT 语句,WRITE 后的第一个 PRINT 主句不能正确按 B_b 参数指定的位置读写数据。

错误原因

在程序中访问文本文件时,必须先用 OPEN 命令打开文件,尔后读写。OPEN 命令格式为:

OPEN f[,V_v][,D_d][,S_s][,L_j]

j 表示记录长度,DOS 区分打开的文件是顺序的或是随机的,就是依靠参数项 L_j,系统内部并不区分对这两种文件的操作,而且在文件上不做任何标记。

当对文件进行访问时,如为写,则用 WRITE 命令确定文件写指针位置,格式为:

WRITE f[,R_p][,B_b]

若为读,则用 READ 命令确定文件读指针位置,格式为

READ f[,R_p][,B_b]

这两个命令既可对随机文件,亦可对顺序文件进行读写。读写随机文件时,命令中必须带 R_p 参数,R_p 指文件中的第 p 号记录。DOS 将把 R_p 参数值送入地址 $AA6E,$AA6F 中。这些地址里的值在下一次被 R_p 参数项修改前一直有效,即对 p 号记录访问完毕后系统并不对 $AA6E,$AA6F 进行初始化(置 0)。当随后接着对顺序文件进行读写操作时,READ 或 WRITE 命令不能带 R_p 参数,此时系统将把这个顺序文件视为记录长度为 1,即 L 参数等于 1 的随机文件,并把读写指针定位于 $AA6E,$AA6F 所指定的"记录号"位置,再根据 R_p 参数确定字节数。问题就出在这个因使用随机文件后遗留下来的"记录号"上,(转 18 页)

ProDOS 系统盘的修改

廖　凯

编者按　与 APPLE DOS3.3 相比，ProDOS 的性能更为优越，并在许多方面接近 16 位微机的磁盘操作系统，但由于推出较晚，国内介绍、剖析的文献也少，多数中华机和 APPLE 机的用户对它尚了解不多，甚至不知如何操作应用。从本期开始，我们将连载系统介绍 ProDOS 的专文，并特邀北京铁路局的廖凯先生撰稿。

ProDOS 系统以其卓越的功能而受到 APPLE Ⅱ及 CEC-I 用户的青睐。由于原版 ProDOS 系统中有一识别机型的子程序，所以只能在 APPLE Ⅱ原装机上运行，要想在 APPLE Ⅱ兼容机及 CEC－I 上运行，就必须修改 ProDOS 系统。另外，ProDOS 系统只使用 35 磁道，而现在的驱动器已可以使用 40 磁道甚至更多磁道。因此，本人对三种版本的 ProDOS 系统，即 ProDOS V1.0.1、ProDOS V1.1.1 及 ProDOS 8 V1.5 进行了修改，使其系统具有管理 40 磁道或 44 磁道的功能，磁盘的逻辑块总数将达到 320 块或 352 块，大大地扩充了磁盘的容量，经修改过的 ProDOS 系统可以在 APPLE Ⅱ兼容机及 CEC-I 上运行。

具体修改如下：

一、系统文件 PRODOS 的修改

1、ProDOS V1.0.1 和 V1.1.1 的修改

在启动 ProDOS 时，由于不是原装机而锁死，此时按 CTRL-RESET 便进入监控状态，然后针对各版本键入以下数据：

V1.0.1 版　23D4:EA EA A9 40 EA EA

V1.1.1 版　2428:EA EA A9 40 EA EA

然后 2000G，启动 ProDOS 系统，再进入 BASIC 状态，键入 BLOAD PRODOS,TSYS,A＄2000，按 CALL－151 进入监控状态，然后针对各版本键入以下数据：

V1.0.1 版　23D4:EA EA A9 40 EA EA
265B:EA EA
520C:C9 40(40 道)或 C9 60(44 道)

V1.1.1 版　2428:EA EA A9 40 EA EA
269E:EA EA
56E0:EA EA C9 40(40 道)或 EA EA C9 60(44 道)

按 CTRL-C 回到 BASIC 状态，键入 BSAVE PRODOS,TSYS,A＄2000。若 PRODOS 已上锁，就先解锁再存盘。

2、ProDOS 8 V1.5 的修改＊

用具有磁道扇区编辑功能的软件，如 ZAP、DISK FIXER 或 COPY][PLUS 中的 SECTOR EDITOR 等，读取＄01 磁道＄0A 扇区，将＄45－－＄4A 单元内值改为 EA EA A9 40 EA EA，再写回＄01 道＄0A 区。再读取＄04 磁道＄07 扇区，将＄E0－－＄E3 单元内的值改为 EA EA C9 40(40 道)或 EA EA C9 60(44 道)，再写回＄04 道＄07 区。

＊注：以上参数是以 PRODOS 文件为盘中第一个文件为基准的。

经修改过的各版本 ProDOS 便具有处理 40 磁道或 44 磁道的功能。

二、文件 FILER 的修改

目前 FILER 文件有两种版本，即 FILE V1.0 与 FILER V1.1。修改如下：

启动 ProDOS 系统进入 BASIC 状态，键入 BLOAD FILER ,TSYS,A＄2000，按 CALL－151 进入监控状态，然后针对各版本键入以下数据：

V1.0 版　4243:A9 40(40 道)或 A9 60(44 道)
79F3:C9 28(40 道)或 C9 2C(44 道)

V1.1 版　4269:A9 40(40 道)或 A9 60(44 道)
79F3:C9 28(40 道)或 C9 2C(44 道)

按 CTRL-C 回到 BASIC 状态，键入 BSAVE FILER,TSYS,A＄2000。若 FILER 已上锁，就先解锁再存盘。

经修改过的 FILER 文件，与修改过的 ProDOS 配合，就可以管理 40 磁道或 44 磁道的磁盘。

最后重新启动修改过的 ProDOS 盘，选择 FILER，格式化一张磁盘，将修改过的 PRODOS、FILER 以及其它文件拷入刚格式完的磁盘，此磁盘便为 40 磁道或 44 磁道的 ProDOS 系统盘。

制作 40 磁道磁盘的最简方法

湖北荆州师专计算机室　瞿波

APPLE 机用户为了增大磁盘容量，常将磁盘由 35 磁道扩充为 40 磁道，但所使用的方法大都较繁。本文给出制作 40 磁道磁盘的最简方法。在键入 INIT 格式化命令之前，先键入如下三条 POKE 语句：

```
]POKE 44725,160↙
]POKE 46063,40↙
]POKE 48894,40↙
```

再使用 INIT 命令格式化出的磁盘就具有 40 磁道，其 VTOC 表中的有关参数在格式化时已设定好，用不着再去进行修改、释放等工作。用 FID 程序的第 3 项功能，可看到该磁盘的扇区数增加了 80 个。格式化出的 40 磁道盘将工作得很正常，而且，用这张盘作源盘去格式化其它盘，得到的也是 40 磁道的磁盘。

PC—1500 机新监控程序

沈玉波

监控程序是用户编制机器语言程序的有力工具。PC—1500 机原未配监控程序，后来厂家提供了一个 0.5K 的多功能监控程序，使用该程序可观察和修改 64K 内存中任一单元内容（当然对 ROM 及无 RAM 的空缺区修改无效），并可检验 32 个内存单元是否有变码。使用时各状态间可相互转 换，也可退出监控，返回 BASIC。在使用过程中，我们感到该监控程序还有一些不尽如人意之处，如在监控状态不能直接关机，须返回 BASIC 方能关机，且关机后再开机时，须重新输入地址，给机器语言程序的调试带来不便。另外，原监控程序不具备增、删功能。

这里向读者介绍我们新编制的监控程序。它除具备原程序的全部功能外，增加了监控下关机和一字节增、删功能，使用更加方便。在监控模式下，按 OFF 键可直接关机；按 ON 键开机后，可保存关机前的显示内容、工作模式和各寄存器（CPU 内）内容，反复调试程序更加方便。在编辑状态，用 SHIFT DEL 键可在光标位置删除一字节，其余后续字节依次前移。按 SHIFT 和 INS 键后，可在光标处加入一条 NOP 指令（38H），其余后续字节依次后移。

监控程序全长 634 字节，可浮动。如使用 PC—1500A 可全部存入 7C01H 为首址的内存中，不受 NEW 或 NEW N 指令影响。如是 PC—1500 机应当首先用 NEW &4340 开辟机器语言保护区，然后再以 40C5H 为首址存入。使用时首先用 CALL &7C01 或 CALL &40C5 指令启动监控，这时屏幕显示 ADR. &__，应输入 4 位十六进制地址，如按下/、*、—、+、=及·键时，将由监控程序译码为 A、B、C、D、E、F，可快速输入数据。如输入地址有误，可重新输入或用 CL 键清除显示后重输。输入正确地址后，按 ENTER 键进入内容显示状态，一次显示首址及 8 个字节内容，按↓、↑键可显示后、前 8 字节内容；如按▶、◀键则进入编辑状态，这时可输入机器语言程序或对内存进行修改、增、删；如按 & 键可在光标处复制前一字节内容。按▲键则进入校验和状态，可一次显示 32 字节内容，校验和状态按↑、↓键可快速显示校验和。在各状态均可按 F1 键进入地址输入状态，也可按 ON 键返回 BASIC。不能对 BASIC 区首址后内容进行增删处理。

新监控程序清单如下：

```
7C00— 0D 6A 45 58 7B 5A 10 E9
7C08— 78 9B 00 FD 58 B5 04 8E
7C10— 46 94 BE 7B 15 14 FD C8
7C18— F1 BE 7B 26 BE ED 4D FD
7C20— 8A BE 7B 26 BA ED 4D B9
7C28— 0F F9 B3 30 B7 3A 81 02
7C30— B3 06 9A BE E2 43 48 7B
7C38— 4A 45 6A 05 F7 8B 03 88
7C40— 05 9A 24 B3 40 9A 2E 3D
7C48— 2B 2D 2A 2F 41 44 52 2E
7C50— 26 20 20 20 20 5F 0D FD
7C58— CA F5 88 03 48 7B 4A 4B
7C60— 58 7B 5A B0 6A 0A F5 88
7C68— 03 E9 78 80 00 BE E8 CA
7C70— BE 7B 32 48 7B 4A B5 B7
7C78— 20 81 0B FD 5A 44 F5 F5
7C80— F5 1E 9E 1B 9E 2A B7 18
7C88— 9B 2E B7 0E 8B D7 B7 0F
7C90— 8B 06 B7 0D 99 2D 8E 05
7C98— BE E3 3F 9E 2A BE ED 95
7CA0— 91 46 18 BE ED 95 91 4C
7CA8— 1A F2 E9 78 75 00 B5 41
7CB0— AE 7B 0E FD 98 14 B9 F8
7CB8— ED 78 9B 40 89 AD 1A BE
7CC0— 7B 10 EF 78 75 04 6A 07
7CC8— EF 78 75 04 55 FD A8 BE
7CD0— 7B 15 FD 2A 88 0E FD 1A
7CD8— ED 78 7C 01 8B 64 EB 78
7CE0— 7C 81 15 ED 78 9B 80 89
7CE8— 01 F1 BE 7B 26 AE 78 7D
7CF0— 14 B9 07 DD DD F1 ED 78
7CF8— 9B 80 8B 02 B3 06 BE EE
7D00— 22 CA 7E B5 7F CD 8A BE
7D08— 7B 32 BE ED 7D 81 18 ED
7D10— 78 9B 80 89 05 F1 59 0F
7D18— 8E 02 59 F0 1B 1E EF 78
7D20— 9B 80 91 7A 54 9E 7D B7
7D28— 0C 9B 0D B7 08 89 DB EF
7D30— 78 9B 80 93 8B 56 9E 8E
7D38— EB 78 9B 40 E9 78 7C 00
7D40— 9E 99 BE E2 43 B7 0C 8B
7D48— 81 B7 08 8B B4 B7 0A 8B
7D50— 87 B7 0B 8B 87 B7 09 9B
7D58— 21 B7 11 9B D9 B7 0F 8B
7D60— 05 B7 0E 99 BB 9A BE E3
7D68— 3F 9E 26 B9 F0 1A B5 F0
7D70— FD DA FD 52 BE 7B 10 B5
7D78— 7E BE ED 4D B5 00 6A 0F
7D80— D9 13 54 88 05 FD C8 56
7D88— BE 7B 10 EF 78 75 06 FD
7D90— 8A BE 7B 15 ED 78 75 40
7D98— 8B 07 EF 78 75 0C 54 9E
7DA0— 2D BE E2 43 FD 1A B7 0A
7DA8— 8B 38 B7 0B 8B 38 B7 08
7DB0— 8B 45 B7 0C 8B 13 B7 0F
7DB8— 8B 0A B7 09 99 65 E9 78
7DC0— 9B 3F 9E 88 BE E3 3F 9E
7DC8— 70 F2 E9 78 9B 3F 14 B9
7DD0— F8 1A EB 78 7C 81 9E A2
7DD8— B5 08 8E 13 B5 F8 FD 52
7DE0— 8E 0D 68 20 8E 04 68 E0
7DE8— FD 52 14 B9 F0 1A A4 FD
7DF0— DA E9 78 7C 00 9E C1 14
7DF8— B9 F0 1A 56 F2 E9 78 9B
7E00— BF EB 7C 9B 80 14 BB 07
7E08— 9E 39 B7 0F 89 05 BE E3
7E10— 3F 9E EC B7 1D 8B 11 B7
7E18— 1C 8B 36 B7 16 99 D2 56
7E20— 55 51 E9 78 9B 04 9E F2
7C28— 14 AE 40 00 94 AE 40 01
7E30— 54 54 56 57 1E 54 54 54
7E38— 94 A7 78 65 99 0C 14 A7
7E40— 78 66 99 12 56 B5 00 1E
7E48— A5 40 00 1A A5 40 01 18
7E50— 9E 2C 44 04 AE 40 00 84
7E58— AE 40 01 A5 78 65 08 A5
7E60— 78 66 0A 46 46 45 0E 04
7E68— 16 99 08 84 96 99 0C B5
7E70— 38 1E A5 40 00 04 A5 40
7E78— 01 08 9E 57
```

《C 语言初阶》讲座

第五讲　程序控制结构(二)

李文兵　王玉华　蓝智斌

3. 中断结构

(1)中断循环结构　这种结构是由中断语句实现的,这里介绍两个用于 C 程序各种循环结构中的中断语句:

break;——跳出循环结构

continue;——返回到循环开头

①break 语句　该语句除了用在 switch 结构中,还常用于各种控制结构中,其功能是从其所在的循环结构中跳出,如练习 5.11 所示。

〔练习 5.11〕

```
C>type exp5-11.  c
main ()
   {  int i, j, k;
      for (j=0; j<10; j++)
         {  k=i+j;
            if (k>15) break;
         }.
      printf ( " i     j     k     \n");
      printf ( " %d %d    %d\n", i, j, k);
  }
C>exp5-11
  i     j     k
  0    10    9
```

②continue 语句用于各种循环结构中,常作 if 的执行语句,其功能是返回到循环结构的开关,如练习 5.12 所示:

〔练习 5.12〕

```
C>type a:exp5-12.c
main()
    {  int i,j;
    /* continue test 1 */
      for (j=4;j<10;j++)
      {  for (i=0;i<50;i++)
           {  if (i%j) continue;
              printf("%d ",i);
           }
        printf("\n");
      }
      /* continue test 2 */
        i=0;
        while (++i<=9)
          {if (i==5) continue;
             printf("%d ",i);
          }
      }
```

```
C>exp5-12
0  4  8  12  16  20  24  28  32  36  40  44  48
0  5  10  15  20  25  30  35  40  45
0  6  12  18  24  30  36  42  48
0  7  14  21  28  35  42  49
0  8  16  24  32  40  48
0  9  18  27  36  45
1  2  3  4 6  7  8  9
```

(2)中止函数执行结构　这种程序结构使用 return 语句实现。该语句有两个功能:一是它将立即中断含有它的那个函数的执行,返回到调用此函数的语句处往下执行;二是它可以用来回送一个数值。C 程序中大多数函数是采用 return 语句来终止运行的,原因是有时函数须返回一个数值;有时需要在函数中设置多个终止点以提高效率。这就是说,一个函数可以有多个返回语句,如练习 5.13 所示。

```
C>type exp5-13.c
main()
  {  char str1[]="abcdefghijkl.";
     char str2[]="efg";
     printf("str2 in str1= %d\n",substr( *str1, *
                str2));
  }
  substr(s1,s2)
  char  *s1, *s2;
  {  register int t;
      char  *p2, *p;
      for (t=0;s1[t];t++)
      {  p=&s1[t];
         p2=s2;
         while( *p2&& *p2== *p)
           {  p++,p2++;
           }
         if( ! *p2) return(t);
      }
      return(-1);
  }
C>exp5-13
str2 in str1=4
```

(3)无条件转移结构　使用 goto 语句可实现无条件转移,其结构形式如下:

goto　标号;

功能是:它能转去执行标有该标号的语句。所谓标号,同变量一样,也是用标识符来表示,只是其后面跟有冒号而已。用 goto 语句转去执行的语句前面要放有标号。

goto 语句可从多重循环的某层次转到任何层次;

而 break 语句只能从里向外跳转一层。这是 goto 语句的优点。但是必须注意，①goto 语句只能在 goto 所在函数内跳转，不能转到函数外。②goto 语句不适于结构化程序，因此，奉劝大家尽量少使用，以免造成程序难读难懂。

这里，介绍一个使用 goto 结构从最里层 for 结构中跳出的程序，如练习 5.14 所示。

〔练习 5.14〕

```
A>type exp5-14.c
main()
    {  int i,j,k;
       for (i=0;i<10;i++)
       {  printf("i=%d\n",i);
          for (j=0;j<10;++j);
             {  printf("i=%d,j=%d\n",i,j);
                for (k=0;k<10;k++)
                {  goto stop;
                   printf("%d,%d,%d\n",i,j,k);
                }
             }
       }
       stop:printf("from stop jmp.");
    }
A>exp5-14
i=0
i=0,j=10
from stop jmp.
```

4. 递归结构

从 2 讲中，我们知道，一个函数可以调用另一个函数。此外，函数还可以调用其本身。这样的函数，就叫递归函数，递归函数就是一个递归结构。阶乘的算法就可用递归结构来实现，如练习 5.15 所示。

〔练习 5.15〕

```
A>type exp5-15.c
main()
    {  int i;
       long int fact(int);
       for (i=1;i<10;i++)
          printf("fact(%d)=%8ld\n",i,fact(i));
    }
    long int fact(i)
    int i;
    { if (i==1)
        return((long)i);
      else
        return((long)i * fact(i-1));
    }
A>exp5-15
fact(1)=          1
fact(2)=          2
fact(3)=          6
fact(4)=         24
fact(5)=        120
fact(6)=        720
fact(7)=       5040
fact(8)=      40320
fact(9)=     362880
```

这里，函数 fact()就是递归函数。

下面，再介绍 2 个递归结构的用法。

第 1 个是求两个正整数最大公约数的程序，如练习 5.16 所示。

〔练习 5.16〕

```
A>type exp5-16.c
#include <stdio.h>
main()
{  int i,j,k;
   char s[80];
   for (k=0;k<4;k++)
    {  i=atoi(gets(s));
    if (i<0)
      exit(0);
    j=atoi(gets(s));
    printf("hcf(%d,%d)=%d\n",i,j,hcf(i,j));
    }
}
hcf(i,j)
int i,j;
{  int r;
   r=i%j;
   if(r==0)
   return(j);
   else
   return(hcf(j,r));
}

A>exp5-16
9
21
hcf(9,21)=3
8
2
hcf(8,2)=2
40
16
hcf(40,16)=8
5
2
hcf(5,2)=1
```

这里用的是欧几里得算法，即对于正整数 p 和 q，设 p 除以 q 的余数为 r，就有：

若 r 为 0，则最大公约数就是 q；

若 r 不为 0，则用 q 和 r 分别置换 p 和 q，重复进行上述判断。

第 2 个例子是求费班纳赛(Fibonacci)级数的程序。该级数为：

0,1,1,2,3,5,8,13,21,34,……

可见，每个元素是由相邻的前两个元素的和组成的，如：

```
f(4)=f(3)+f(2)
    ={f(2)+f(1)}+{f(1)+f(0)}
    ={f(1)+f(0)+f(1)}+{f(1)+f(0)}
    ={1+0+1}+{1+0}
    =3
```

其一般形式为：

n=0,1 时，f(n)=n；

n>=2 时，f(n)=f(n-1)+f(n-2)

实现该级数的程序如练习 5.17 所示

[练习 5.17]

```
A>type exp5-17.C
main()
    {  int i=6;
       for(i=0;i<10;i++)
       printf("fib(%d)=%d\n",i,fib(i));
       }
       fib(i)
       int i;
       { if(i<=1)
           return(i);
         else
         return(fib(i-1)+fib(i-2);
       }
A>exp5-17
fib(0)=0
fib(1)=1
fib(2)=1
fib(3)=2
fib(4)=3
fib(5)=5
fib(6)=8
fib(7)=13
fib(8)=21
fib(9)=34
```

关于四舍六入程序的讨论

辽宁地震局地震大队　梁　放

贵刊一九九一年第二期上刊登了韦肖鸿同志的关于数据 4 舍 6 入处理程序，我对该程序进行了实际使用，在使用中发现该程序随然能解决数据 4 舍 6 入问题，但是由于采用字符处理方式，使得处理方法比较繁琐，且程序较长、运行速度慢。我根据自己的工作经验，在不改变数据取舍原则的基础上编写了如下程序。从程序清单可以看出，该程序语句很少，步骤简洁明了，特别适合于内存容量有限的微型计算机做为子程序使用。

```
1000 INPUT "A=";A:INPUT "P=";P
1001 B=A+0.5/10^P:C=INT(B*10^P)/10^P
1002 IF B*10^P=INT (B*10^P) AND INT ((A*10^P)/2)*2-INT (A*10^P)=0THEN LET C=INT (A*10^P)/10^P
1003 PRINT C
```

说明：

A：为要处理的数据。

P：为需要保留的小数位数。

C：为处理后的结果。

若 A 为负数，可先将 A 变为|A|，再进行处理，处理后还原。

浙江嵊县中学　叶德祥

对正负数的“舍”和“入”的理解应有区别，如保留小数 1 位，对于 1.55 是“入”变成 1.6，其值比原数大，而对－1.55 是“入”变成－1.6，其值比原数小，所以“入”不一定数值变大，同样“舍”数值不一定变小。如果用韦肖鸿同志的程序对 1.55 和－1.55 进行保留 1 位小数的修约，则分别变成 1.6 和－1.5；原来互为相反数的两数修约之后就不互为相反数了，这不合理，并有可能使程序潜伏难以发觉的错误，绝不能掉以轻心。此外，这样的程序不用字符串也能解决，而且运算速度更快，下面是我编写的程序，供大家参考。

```
100 Y=X*10^N:Z=INT(Y+0.5)
110 IF Y+0.5>Z THEN 140
120 IF Z/2=INT(Z/2) THEN 140
130 Z=Z-1
140 X=Z/10^N
150 RETURN
```

程序说明：数据 X 保留 N 位小数，对小数点后第 N+1 位进行舍入处理。如小数点后第 N+1 位数字不是 5，或小数点后第 N+1 位数字是 5，但后面的数字不全是零，则与平常的舍入法则相同，此时 Y+0.5>INT(Y+0.5)由 110 句判断。如小数点后第 N+1 位数字是 5，且后面的各位数字皆为零，则舍入后要保证小数点后第 N 位数字为偶数，即 Z 为偶数，由 120、130 句实现。

学用单片机

单片机最小应用系统

张培仁

MCS－51 系列中，8051 片内有 4K 掩膜 ROM，8031 无 4K 掩膜板 ROM，8751 片内有 4KEPROM。从国内市场买到 8051 时，用户无法把程序写到掩膜 ROM 中。因此，国内组成单片机最小应用系统对 8051（或 8031）来说只能外接一片 EPROM（如 2764）作为用户程序。8051（或 8031）从 2764 中取指令时，必须从 P2 口和 P0 口分别输出 16 位高位地址和低位地址。而 P0 口又是数据输入，输出的口。因此 P0 要在指令控制下分时输出地址，再输入或输出数据。这样先输出的地址要锁存起来，2716，2732，2764 等都是不带地址锁存器的。为了从 EPROM 中取出稳定的正确的程序必须在 8051 外再外接一片 74LS373 作为地址锁存器，由 3 片组成最小系统。如果 EPROM 是带地址锁存器时只要 2 片就可以组成最小的应用系统。一般为了便于扩充，常选用 4 片组成最小系统（增加一片 74LS138），作为最高 3 位地址的译码器。最小应用系统的电原理图如图 1 所示。印刷电路板图为 1：1，见本期封三。

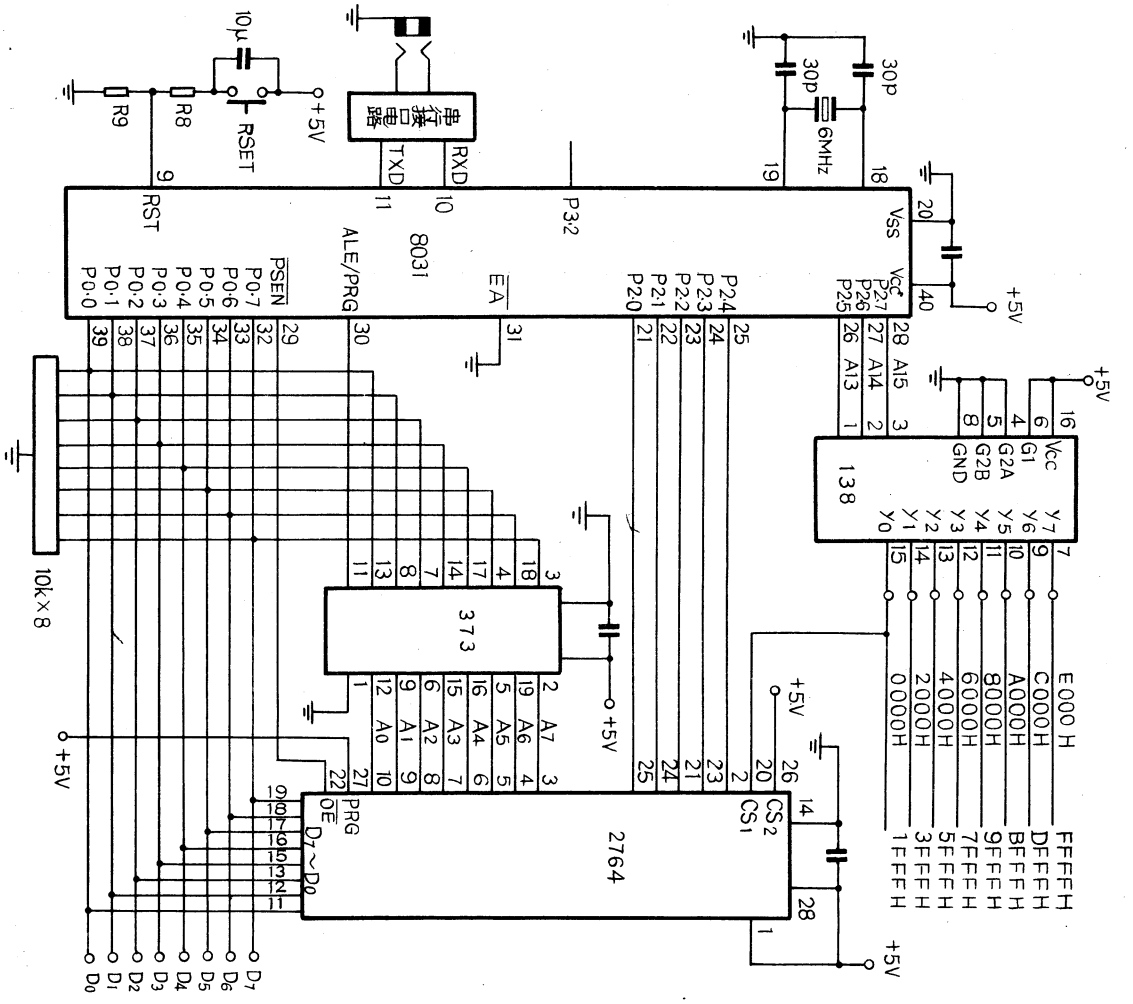


图 1

原理图中一部分控制信号说明如下：

RST/VPD：复位/掉电时备用电源。复位信号“1”电平有效。片内内接下拉电阻(10KΩ)。

ALE：允许地址锁存。“1”电平有效。此信号是周期出现，频率为主振荡频率的 1/6。

$\overline{PSEN}$：此输出是外部程序存储器的读选通信号。它的产生是由程序存储器的取指令来控制。在访问外部数据存储器时此信号不出现。“0”电平有效。

$\overline{EA}$：在最小应用系统中它永远是低电平。因为总要不断地从外部 EPROM 取指令。

在最小的应用系统中，P0 口已不能再作 I/O 口使用，它只能在指令控制下分时输出地址和输入输出数据。P2 口也只能作为 2764 高位地址的输出，也失去作为 I/O 使用的可能。因此可供用户使用只有 P1 口和 P3 口。P3 口是多功能口。

如图所示，此最小应用系统包括 8031，2764，373，138 各 1 片。以及＋5V 稳压电源的整流、滤波、稳压电路。还包括用二只晶体管组成非标准的 RS232 接口电路和三芯插座一只。为了便于扩充还留下 40 芯接口插座一个。

当最小系统由三片组成时，省掉 138，把 2764 的 CS1(片选信号)接地。其他不做任何改动。如果 CPU 用 80C31 时要在 P0 口接下拉电阻。

最小系统中用户可用资源还有二个 16 位定时器/计数器，一个可编程的串行口，8031 片内可用 128 字节 RAM 等。用户可以在这样硬件环境下创造出丰富多彩的自己的单片机应用系统。(本刊今后将陆续介绍 KDC 最小应用系统的文章供读者参考——编者注)

电 脑 坦 克

(KDC 最小系统应用实例之一)

张培仁　严　琦　阮永光

KDC 最小系统稍加扩充可构成一个实用的功能强的单片机应用系统。

玩具坦克是两个直流电机驱动的，原来用遥控开关转换电源的极性，以控制电机的正转反转，从而使坦克前进，后退、转弯等。我们改用单片机来控制其运动，并接上小喇叭和发光二极管，可模拟枪炮及音乐音响。具体电路见图 1。驱动电路是这样工作的：8031 的 P1.0～P1.3 口，经 9012 驱动，控制光电耦合器 4N25 的导通与截止，4N25 实现了单片机与功率驱动部分的电气隔离，最小系统与驱动电路可采用不同的电源，4N25 完成了单片机＋5V 到驱动电机±5V 电源的电平转换，8751 与 8778 是同一类型的功率集成开关电路，4N25 的第 4 脚输出接到集成开关的控制端(8751 的第 1 脚，8778 的第 5 脚)，8751 输出接电机，它导通时电机正转(图中电流 I_1)，同样 8778 导通时电机反转(图中流过电流 I_2)。另外喇叭和发光二极管也是由一片 8751 驱动的。

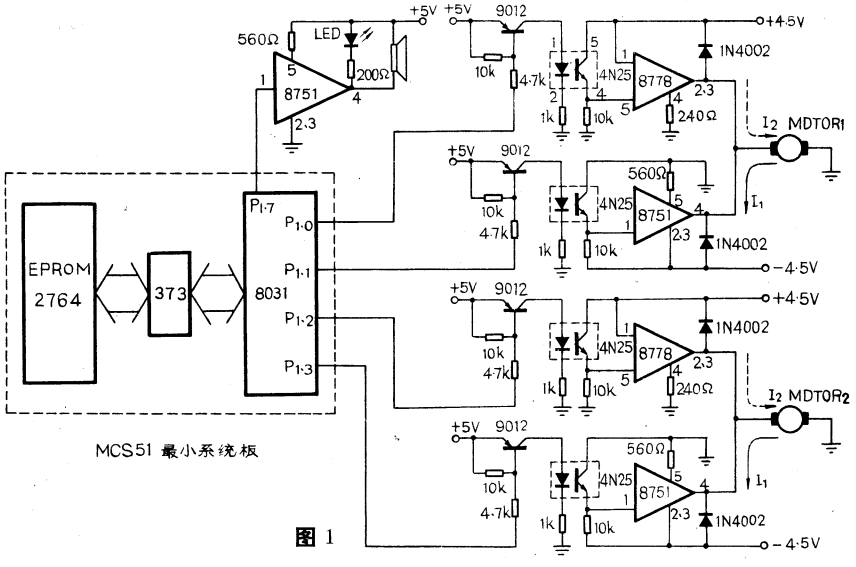


图 1

TWH8751,TWH8778 都是功率开关集成电路。这是专为逻辑电路输出作接口而设计的直流功率开关，可由 TTL、CMOS 等数字电路直接驱动。该器件工作可靠，开关速度快，工作效率高，寿命长。这是一种广泛应用自控系统、计算机终端接口，测量仪器中微电机控制等电路。TWH8751,8778 是一种集放大，比较、选通，整形和功率输出于一体的大规模集成开关功率器件。并自带散热片。TWH8751 是五端器件。①是控制选通端(ST)与输入端和输出端相差 180°。②是输入端 IN。③端是地。④端功率输出端(OUT)。⑤供 TWH8751 放大，比较，驱动电路电源端。电压不超过＋6V。超过＋6V 时要接降压电阻。

TWH8751 性能特点：1. 器件设计有死区，有滞回特性，低噪声、抗干忧性能好。2. 有自我保护，不易损坏。控制电平输入＜0.3V 导通，＞1.1V 时输出功率管截止。3. 工作频率可达 1.5MHZ。4. 开关性能好，边沿延时毫微秒级。5. 控制功率大，功率开关管击穿为 100V，加散热片可达 3A。典型应用如图 2。

TWH8778 和 TWH8751 不同的是 8778 输出功率管是 PNP 型，8751 是 NPN 型。其功能性能基本相似。

软件部分主要是对 P1 口的控制。程序主模块是表演部分(Persorm)用于设置坦克行走路线与发声控制，下面有二个子模块，即方向控制子模块，包括前进 Forward，后退 Backward，左转 LTurn，右转 RTurn，发声子模块包括音乐 MUSIC(含五首曲子)，枪炮，详细注解见程序清单。

将程序写入 EPROM 后，插入最小系统插座，再将 P1 口接到控制板上，坦克就可按预想的方式运动，并演奏美妙的乐曲，在停顿时发出激烈的枪炮声。

稍加改动，该系统还可扩展许多功能，如增加一超声测距装置，可使坦克探测到障碍物，并自动走出迷宫；接上发光管和光敏接收管，可实现坦克与目标间的对抗性演习，加上红外发射接收管，很容易实现遥控。单片机为这些功能的扩展提供了极大的便利。

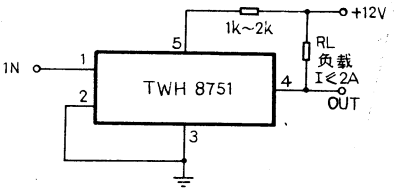


图 2 放大、比较、功放电路

该系统虽是为一玩具坦克设计的，但实际上，它有很大的通用性与实用性，8751 与 8778 属功率集成片。我们曾试用二个 8751 并接可以控制 24V，3.5A 步进电机可靠工作。用软件改变 P1 口输出脉冲的宽度，也可方便地控制电机速度，如果加上各种传感器，如光栅，速度计等，再通过单片机 T0、T1 口输入则可构成一个很实用的闭环的反馈控制系统。

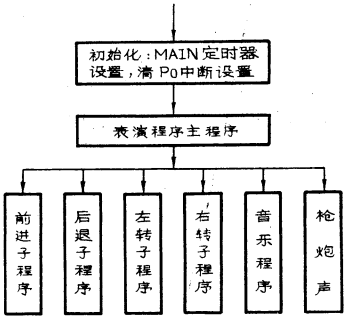


图 3 程序框图

程序框图如图 3 所示。有兴趣的读者可向本刊编辑部函索程序清单，付复印及邮寄费 3.00 元。

(接第 3 页)要看 Intel 的垄断能否被打破。

二、总线的发展

微机的标准总线为 ISA(Industry Standard Architecture)是 IBM 在 1981 年推出 IBM PC 时订下的，它是一个颇为简单的八位同步总线含校验位保护(Parity protection)及以边缘触发中断指令(edge—triggered interrupt)。1984 年 IBM 再推出了 16 位的 AT，为了要跟以前的 8 位 ISA 往下兼容，IBM 在原来的 62 针总线插座下加多一个 38 针的插座，但事实证明 AT 总线需比 8 位的总线有所进步，但它仍是微机在技术上发展的一个最大障碍。ISA 的 AT 总线仍是一个低速度(输入/输出)，及不适合多个微处理器并行工作的总线。

1987 年 IBM 再推出了一系列全新的微机系列(PS/2 系列)，而且采用了 IBM 自己订立的微通道(Micro Channel Architecture)总线。IBM 希望能以 PS/2 系列增加它不断减少的市场占有率，同时为了要保护他在市场的控制权，IBM 注册的微通道总线的版权，凡是要生产微通道总线的厂家，均需付出极为可观的版权费用，而且追索至厂家以前生产 ISA 总线的所有微机。IBM 的政策使大部分的微机生产商没有兴趣设计及生产微通道微机，最致命的是 IBM 的微通道总线跟以往的 ISA 在硬件及软件上并不兼容。在前景不明朗的情况下硬件及软件制造商并未在微通道上大量投资，这是做成微通道总线至今仍未普及或变为主流之原因。

1988/89 年 9 间美国及日本微机制造商联合创制了一种全新的总线 EISA(Extended Industry Standard Architecture)。EISA 的主要目的是解决 ISA 在功能上的不足，EISA 可同步的以 33MHz 速度进行输入/输出数据，而且从设计上容许多处理器并行工作。EISA 的插座有两排插针可往下兼容所有 ISA 的插板，这使在未有专为 EISA 而设计的插板时，用户仍可用现时的 ISA 硬件(当然，功能便跟现时 ISA 总线没有分别)同时 EISA 总线的设计及专利使用权以合理的价格供应给所有微机制造商。这政策看来颇为成功。现时设计 EISA 总线的厂商已远远超过设计 IBM 微通道的厂商。现时支持 EISA 的硬件及软件还未觉丰富，估计 EISA 还需数年才能普及化及成为主流标准。IBM 现时还很积极的宣传及推广 MCA(微通道)，但它的成功未容乐观。(待续)

8031 与 PP40 描绘器接口技术

皇甫文忠

LASER PP40 是一种体积小，价格低的四色描绘式打印机。可用来描绘字符及图形，可以打印 89 个标准 ASCII 字符和 52 个特定字符，并且具有较强的绘图功能。可在各种智能仪器及控制系统中作为微型绘图机使用。当然，在单片机开发系统中配接 PP40 描绘器是一种很实用的选配方案。

一、LASER PP40 技术参考

1. LASER PP40 接口信号及时序

PP40 的接口插座是 36 芯的，实际上只使用了 13 芯，各芯位信号说明如下表：

表一、PP40 接口插座芯位、信号及功能

芯 位	信 号	功 能
1	$\overline{\text{STROBE}}$	选通信号
2～9	DATA1～8	8 位并行数据总线
10	$\overline{\text{ACK}}$	应答信号，表示 PP40 准备好接收下一批数据
11	BUSY	“忙”信号，表示 PP40 正在工作中，不能接收信号数据
12、15	GND	接地
其余	不接	

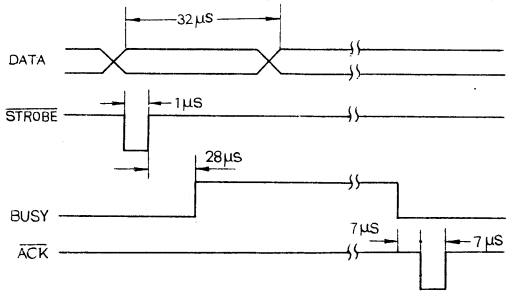
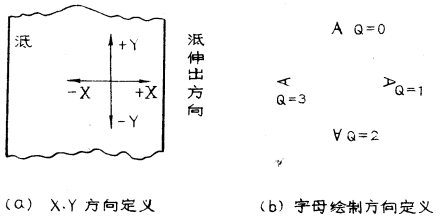


图 1　PP40 接口信号时序



(a) X.Y 方向定义　　(b) 字母绘制方向定义

图 2　X、Y 方向及字母描绘方向定义

表二、绘图命令格式及功能

命 令	格 式	功 能
线形式	Lp（p 由 0～15）	所绘划线的形式，实线 p＝0，点线 p＝1～15
重置	A	笔架返回 X 轴最左方，而 Y 轴不变动。返回文本模式，并以笔架停留作为起点。
回档	H	笔架升起返回起点。
预备	I	以笔架位置作为起点
绘线	Dx，y，… X_n，Y_n（-999≤x，y≤999）	由现时笔嘴位置(x,y)连线
相对绘线	J △x，△y … △x_n，△y_n（-999 ≤△x，△y≤999）	由现时笔嘴位置划一直线至笔嘴点△x，△y 距离之点上
移动	Mx，y（－999≤x，y≤999）	笔嘴升起，移动至起点相距 x，y 之点上
相对移动	R△x，△y（－999≤△x，△y≤999）	笔嘴升起，移动至现时笔架相距△x，△y 之新点上
颜色转换	C_n（n＝0～3）	颜色转换由 n 指定 0：黑，1：兰，2：绿，3：红
字符尺码	S_n（n＝0～63）	指定字符尺码
字母绘制方向	Q_n（n＝0～3）	指定文字编印方向（只在图案模式下适用）
编印	PC，C …… C_n	编印字符（C 为字符，n 无限制）
划轴线	Xp，q，r（p＝0～1）（q＝－999～999）（r ＝ 1 ～255）	由现时笔架位置绘划轴线 Y 轴：p＝0，x 轴：p＝1 q＝点距，r＝重复次数

PP40 接口信号时序如图 1 所示，在单片机应用系

统中，实际使用时，一般不使用 $\overline{ACK}$ 信号。

2. PP40 的工作方式

PP40 具有文本模式和图案模式两种绘图工作方式，加电后，初始状态为文本模式，在文本模式下，如果主机将回车控制字符 CR(0DH)和绘图控制字符 DC2(12H)写入 PP40，则 PP40 由文本模式变为图案模式；在图案模式下，再将回车控制符 CR(0DH)和文本模式控制符 DC1(11H)写入 PP40，则 PP40 又回到文本模式。

1)文本模式

PP40 工作于文本模式时，可描绘所有 ASCII 字符及一些特定字符，另外在 PP40 的字符编码中，有以下几个控制字符控制 PP40 的动作。

• 回位 BS(08H)：使笔回到前一个字符位置，若描图笔已处于最左边位置，该命令失效。

• 进纸 LF(0AH)：将纸推进一行

• 退纸 LU(0BH)：将纸倒退一行

• 回车 CR(0DH)：描图笔返回到最左边位置上。

• 转色 NC(1DH)：笔架转动一个位置至另一颜色笔。

当超过一行的字数后，PP40 自动回车并进纸一行。

2)图案模式

PP40 工作于图案模式时，可通过各种绘图操作命令，绘出各种图形、表格、曲线、绘图命令及格式、功能如表二所示。

X、Y 方向定义、字母描绘方向定义如图 2 所示。

绘图命令的编排有下列约定：

• 参数跟在命令符号后面，参数之间以“,”作分隔符。

• 单字符命令后可直接跟其它命令(返回文本命令除外，它后面必须跟回车符 0DH)

• 一个参数的命令(L,C,S,Q 四条命令)，可以在参数后面加“,”后跟其它命令。

• 二个以上参数的命令必须以回车符 CR(ODH)结束。

二、PP40 与 8031 单片机的接口

PP40 与 8031 单片机的接口形式很多，在具体应用时，可根据不同情况加以选择，大致有以下几种接口形式：采取 I/O 口，扩展 I/O 或总线口的连接方法。

1. 采取 I/O 口的接口方法

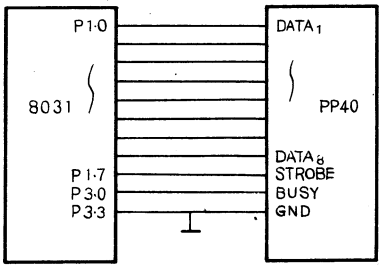


图 3　PP40 通过 I/O 口与 8031 的接口方法

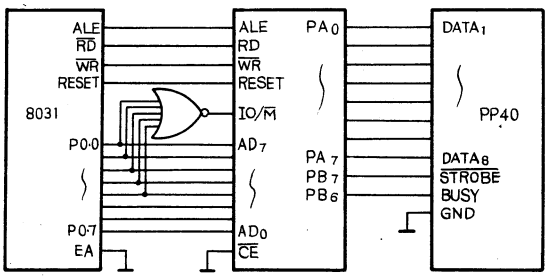
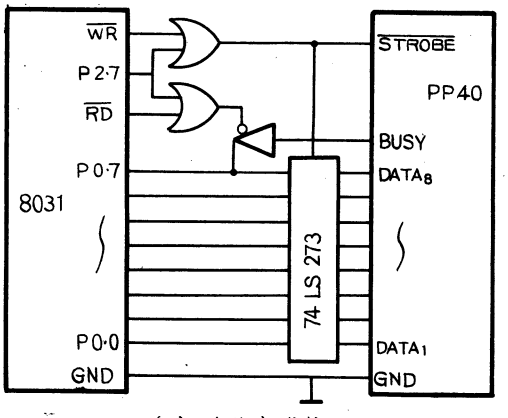
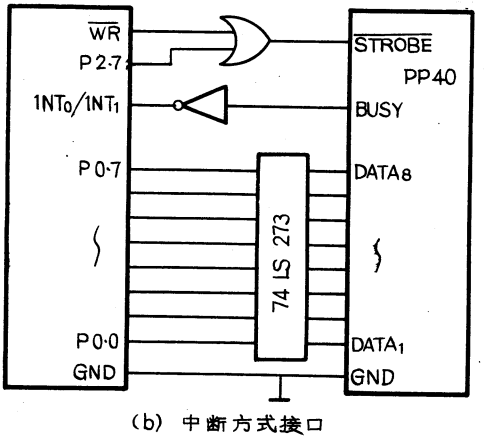


图 4　PP40 通过 8155 与 8031 的接口方法



(a)　查询方式接口



(b)　中断方式接口

图 5　PP40 与 8031 通过 P0 口的接口方法

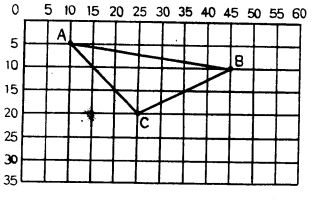


图 6

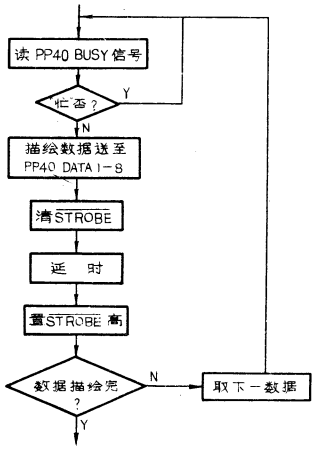


图 7　查询方式程序框图

通过 I/O 口的连接方法比较简单，但要占用 I/O 口线，这里占用 P1 口作为数据通道。

2. 采取扩展 I/O 口的接口方法

这里是用 8155 的 PA 口作为数据通道，也可用 8255 代替 8155，一般来说，只需再加上一对握手线，扩展 I/O 口可以用应用系统中任何并行 I/O 口线。

3. 通过总线口的接口方法

这里在 P0 口的输出端加一锁存器 74LS273，使用 $\overline{WR}$ 信号进行锁存控制。

三、PP40 描绘驱动程序设计

1. 文本模式及图案模式的编码设计

在设计 PP40 的字符及图案描绘程序之前，要对绘制的字符或图案进行编码设计。

首先将所要绘制的字符、图案变成一系列由命令码，控制码及文字字符组成的字符串，然后再将它们“翻译”成相应的以十六进制表示的数据串，作为提供给 PP40 使用的数据表。

举例如下：

1）文本模式的编码设计

绘制：COMPUTER SYSTEM

可采取的编码如下：

DC2,S3,C1 CRLF

DC1 CR LF

COMPUTE2 SYSTEM CRLF

其中 DC2 是图案模式编码，文本模式中可利用图案模式选择字符尺寸，颜色转换等，S3 表示选 3 号尺寸，C1 为选蓝色笔，DC1 是文本模式命令。

在程序中将上述编码转换成下列十六进制数据，存放在存储器中，然后逐个送入 PP40 中，即可绘出要求的字符。

12H，2CH，53H，33H，2CH，43H，31H，0DH，0AH，11H，0DH，0AH，20H，20H，43H，4FH，4DH，50H，55H，54H，45H，52H，20H，53H，59H，53H，54H，45H，4DH，0DH，0AH

2. 1 图案模式的编码设计

绘制如图 6 所示的三角形

编码设计如下：

CR DC2 CR HM10，－5CR；笔嘴抬起先回起点再移至 A 点

L0，J35，－10，－20，－5，－15，15 CR；描绘 AB、BC、CA 线段。

翻译成 16 进制数据如下：

0DH，12H，0DH，48H，4DH，31H，30H，2CH，2DH，35H，0DH，4CH，30H，2CH，4AH，33H，35H，2CH，2DH，31H，30H，2CH，2DH，32H，30H，2CH，2DH，35H，2CH，2DH，31H，35H，2CH，31H，35H，0DH

将上述数据送入 PP40，即可绘出如图 6 的三角形。

2. 描绘程序设计

1）中断方式接口程序设计

以图 3 接口电路为例，PP40 接口采用中断方式

主程序：

```
MAIN:   ⋮
        ORL PSW,#18H      ;描绘工作初始化，选用
                           工作寄存器区 3
        MOV R7,#20H       ;20H 个 ASC Ⅱ 代码数据
        MOV R5,#00H       ;描绘数据区首址 3000H
        MOV R6,#30H
        ANL PSW,#07H      ;恢复工作寄存器区 0
        MOV P1,#20H       ;启动 PP40 描绘空格符
        SETB P3.0         ;选通信号
        CLR P3.0
        SETB IT1          ;置 INT1 为下降沿触发方
                           式
        MOV IE,#84H       ;允许 INT1 中断请求
        MOV PSW,#00H      ;CPU 可做其它工作
        ⋮
```

中断服务程序：

```
PRINT:  PUSH ACC          ;保护 8031 现场
        PUSH PSW
        PUSH DPH
        PUSH DPL
        ORL PSW,#18H      ;选用工作寄存器 3
        MOV DPL,R5        ;送数据区地址
        MOV DPH,R6        ;
        MOVX A,@DPTR      ;送数据至 PP40
        MOV P1,A
        CLR P3.0          ;选通信号
        NOP               ;延时
        NOP
        SETB P3.0
        INC DPTR          ;指向下一数据
        MOV R5,DPL
        MOV R6,DPH
        DJNZ R7,PIRI      ;数据未完返回继续
        SETB 00H          ;设置结束标志供主程序查
                           询
        CLR EX1           ;关 INT1
PIRI:   POP DPL           ;恢复 8031 现场
```

```
        POP DPH
        POP PSW
        POP ACC
        RETI
```

说明：

主程中已将 20H 个描绘数据送入首址为 3000H 的数据存贮器中。

2)查询方式接口程序设计

查询方式接口程序的主要思想是利用软件查询 PP40 是否处于"忙"状态，是则等待，否则将数据送至 PP40，置 $\overline{\text{STRUBE}}$ 低电平，选通 PP40，完成一次描绘，主框图如下：

以图 4 的接口电路为例，程序清单如下所示，其中 8155 命令口地址 4100H，PA 口地址 4101H，PB 口地址 4102H。

```
MAIN:   MOV DPTR,#4100H   ;置 8155 工作方式
        MOV A,#0DH        A 口输入,B 口输出
        MOVX @DPTR,A
        MOV DPTR,#4102H   ;置 STRUBE 高电平
        MOVX A,@DPTR
        SETB A.7
        MOVX @DPTR,A
        ⋮
PRINT:  MOV DPTR,#4102H
        MOVX A,@DPTR
        JB A.6,PRINT      ;BUSY 为 1 则转 PRINT
                          等待
        MOV A,R3          ;R3 中为欲描绘数据
        MOV DPTR,#4101H   ;数据送至 PP40
        MOVX @DPTR,A
        MOV DPTR,#4100H   ;置 8155 工作方式
        MOV A,#0FH        A 口输出,B 口输出
        MOV @DPTR,A
        MOV DPTR,#4102H   ;清 STROBE,选通 PP40,
        MOVX A,@DPTR
        CLR A.7           启动一次描绘
        MOVX @DPTR,A
        PUSH A
        MOV A,#08H        ;延时
PPP:    DEC A
        JNZ PPP
        POP A
        SETB A.7          ;置 STROBE 高
        MOVX @DPTR,A
        MOV DPTR,#4100H
        MOV A,#0DH
        MOVX @DPTR,A
        RET
```

说明：本程序清单中只完成一个数据的描绘，用户可在主程序中将数据送入 R3，然后调用 PRINT，取下一个数据送入 R3，调用 PRINT，……即可完成一批数据的描绘。

解决 TEC—B1 不能使用 AP51—I 卡问题的方法

大庆新华发电厂　刘希栋

《电子与电脑》90 年第 2 期封底邮购消息介绍的 AP51 卡，是用于 APP LE Ⅱ及其兼容机的接口卡，但笔者在使用中发现该卡不能用于天坛机 TEC—B1，天坛机插上该卡引导 AP51DEBUG 后，即死机，无法运行。开始认为是卡出了毛病，但插在 CEC—Ⅰ中华学习机和 APPLE Ⅱ机上，能正常运行，证明卡是好的。

把 AP51 卡分别插在 TEC 和 CEC 机上用示波器查 $\overline{\text{DMA}}$ 信号，可以看出信号是一样的，在引导 DEBUG 后都有 $\overline{\text{DMA}}$ 信号产生。观查 TEC 和 CEC 机上 CPU37 脚(时钟)，信号就不一样了，CEC 机在 $\overline{\text{DMA}}$ 低电平期间 CPU 的 37 脚无时钟，而 TEC 机在 $\overline{\text{DMA}}$ 低电平期间 CPU37 脚时钟是连续的，不受 $\overline{\text{DMA}}$ 控制，这就是 TEC 机无法运行 AP51 卡的根本原因所在。

中华学习机 CEC—Ⅰ中信号对时钟信号的控制过程见图 1，在 $\overline{\text{DMA}}$ 高电平时 CPU37 脚有时钟，在 $\overline{\text{DMA}}$ 为低电平时切断时钟，6502 等待。而 TEC 机 CPU37 脚时钟信号从 U31 74LS04 第 6 脚引出不受 $\overline{\text{DMA}}$ 控制，AP51 卡在 TEC 机上由 DEBUG 启动后，发出 $\overline{\text{DMA}}$ 信号，6502 即应将控制权交给 MCS—51，但因时钟不受 $\overline{\text{DMA}}$ 控制继续提供，造成 6502 与 MCS—51 同时工作，产生了冲突而死机。

针对上述原因，可按图 2 方法使 AP51 卡在 TEC 机上运行。图中 74LS08 是增添的 2 输入 4 与门(只用其中一个与门)，把它放在 TEC 主板上适当位置，7 脚接 0V，14 脚接十 5V，1 脚连续 I/O SLOT22 脚 $\overline{\text{DMA}}$，2 脚连接 U31 74LS04 第 6 脚，3 脚连接至 6502 37 脚，断开原主板上 CPU37 脚到 U31 74LS04 第 6 脚之间的连线。

改动完成后，插上 AP51 卡，引导 DEBUG，AP51 卡将在 TEC 机上正常运行，TEC 机用户要使用 AP51 卡不妨一试。

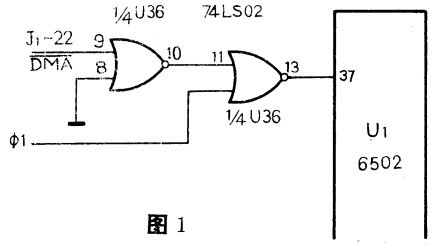


图 1

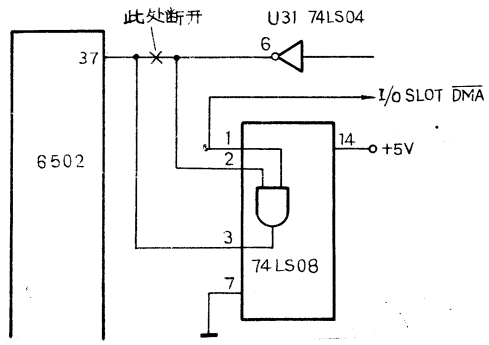


图 2

学装 μp-80

微电脑控制电梯模型的实验与制作

易齐干

该电梯模型假想用于四层楼房。主要动作与实际电梯相同。如：按动指定楼层的按钮，电梯门关闭，电梯起动，到达后，稍停片刻电梯门自动打开。与实际电梯不同之处是不接受一次按押 2～3 个楼层按钮，也不能预约。

该模型大体分为升降，门开闭和基体三部分。

1. 卷绕部分

电梯模型可用厚度为 5～10mm 的有机玻璃制成，强度薄弱处以及电机的固定使用不锈钢板加固。

驱动源使用任意型号的小型直流(DC)电机，由齿轮传动装置组成三级减速器(减速比约 3000:1)带动装在该轴上的滑轮卷绕钢丝绳提升电梯轿厢。轿厢的下降由本身自重实现。轿厢的重量较轻，使用 φ1mm 的钢丝绳就足以保证强度。

“现在，到了几层？”这种位置的检测由安装在各层的光传器完成。在轿厢的下方突出一块板，遮断光传器的光路进行位置检测(图 1 未画出)。

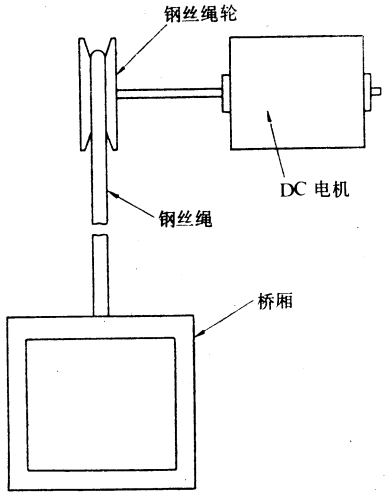


图 1

2. 轿厢门的开闭

轿厢门是用有机玻璃制成。各层楼的横梁是轿厢门的轨道。各层楼都装有直流(DC)电磁铁，电磁铁的吸引力通过厢门与电磁铁之间的杠杆，拉动打开厢门。厢门的关闭是靠弹簧的作用力完成。

DC 电磁铁衔铁的行程(伸出长度)约 15mm，由于各层轿厢门长度是 40mm，所以，利用杠杆的原理实现行程加大。

3. 基体

四层楼的模型也是由有机玻璃制成，各层设有轿厢门，最上方装卷绕部份，最下面为轿厢。在楼的正面与卷绕部份平行方向并排四支 LED(发光二级管)作为监控器，表示电梯所处位置。

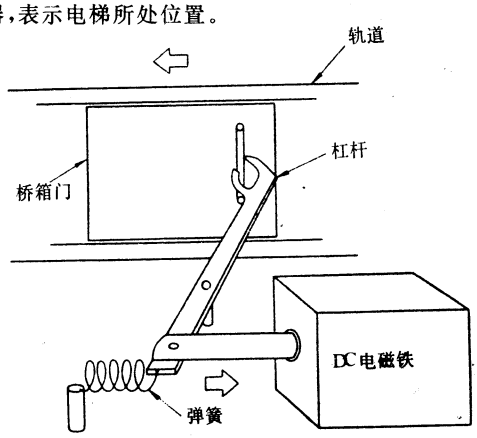


图 2

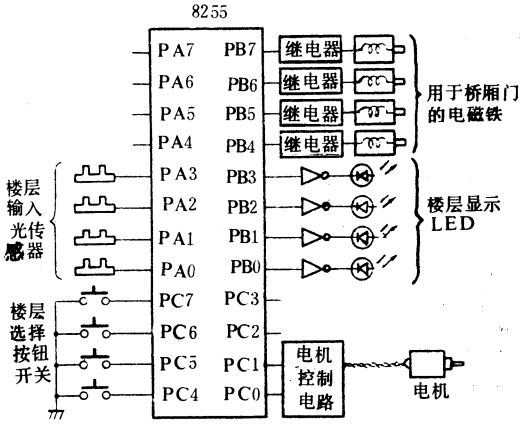


图 3

3、控制

微电脑控制电梯模型可使用 μp-80 单板机，端口分配如图 3 所示。读者可根据端口分配，设计出各接口电路。如光传感器、按钮开关、电磁铁电路、楼层显示 LED 亮灯电路及 DC 电机驱动电路如图 4～图 8 所示。亮灯电路、电机驱动电路，以及 12V 直流电源。

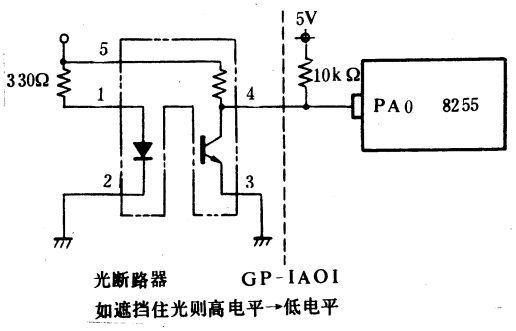


图 4

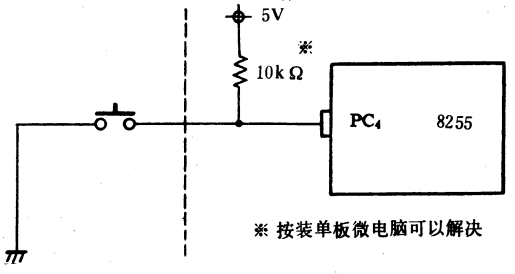


图 5

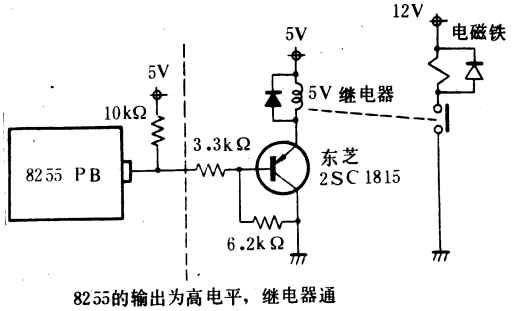


图 6

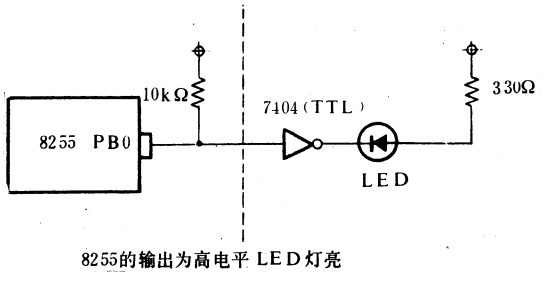


图 7

4. 软件设计

控制电梯的简单流程图如图 9 所示。

该软件是比较按钮开关的状态(由 PC7～PC4 读入,存储于 Z80 的 D 寄存器)与光开关的状态(由 PA3～PA0 读入,存储于 Z80 的 E 寄存器),选择电机应该正转还是停止或是反转。电机停止旋转时(希望轿厢

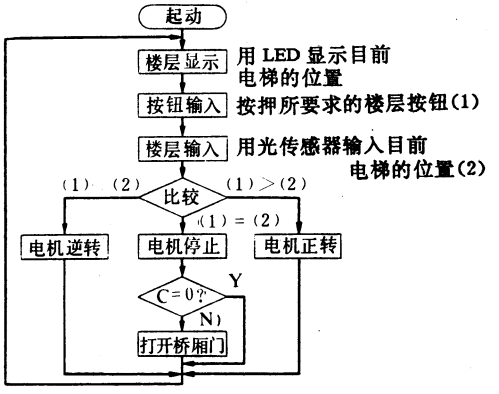


图 9

处于某一层),继电器电路驱动电磁铁,打开该楼层的轿厢门。轿厢门之所以打开,仅是电机停止旋转和判断是正转或反转之故。电机停止旋转由 01H 进行判断。

Z80 各寄存器的分配如表 1 所示。程序清单与流程图如表 2 所示。

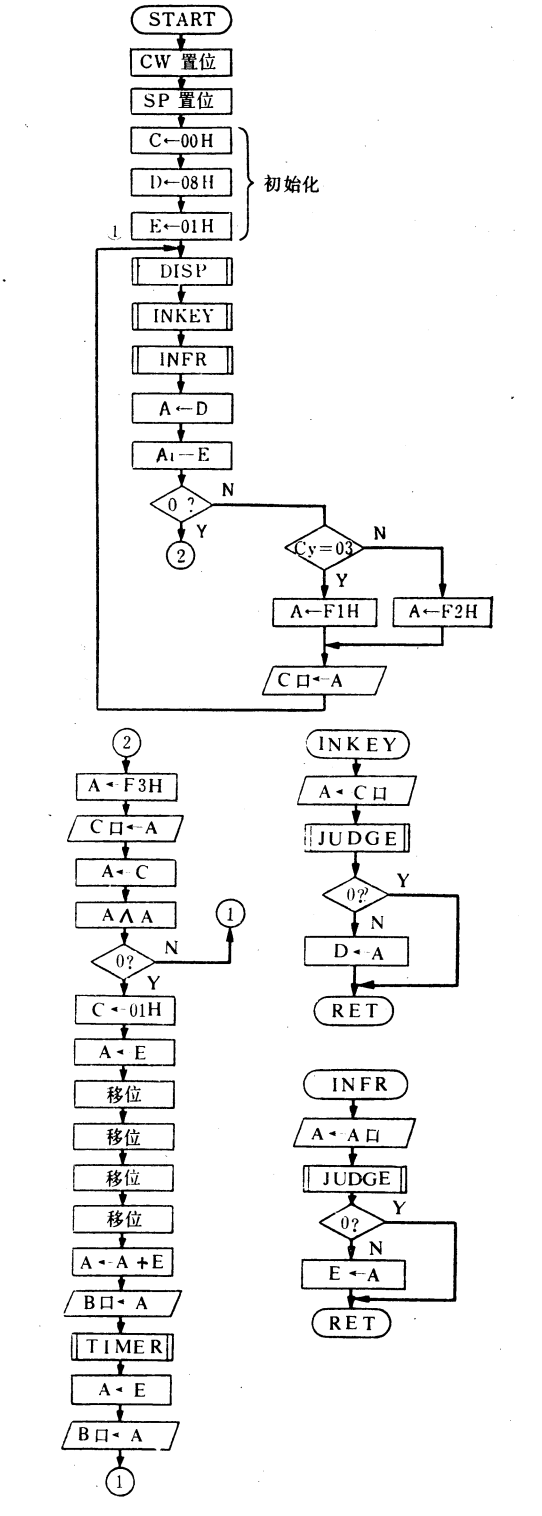
表 1 Z80 各寄存器的分配表:

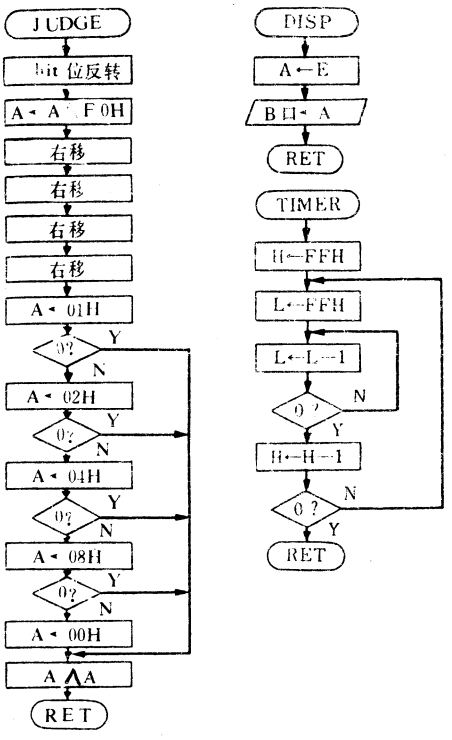
A 寄存器	与 8255 端口交换信息
B 寄存器	不使用
C 寄存器	打开桥厢门的标志
D 寄存器	存储按钮开关的输入值
E 寄存器	存储当前电梯处在的楼层
H,L 寄存器	用于延时的计数器

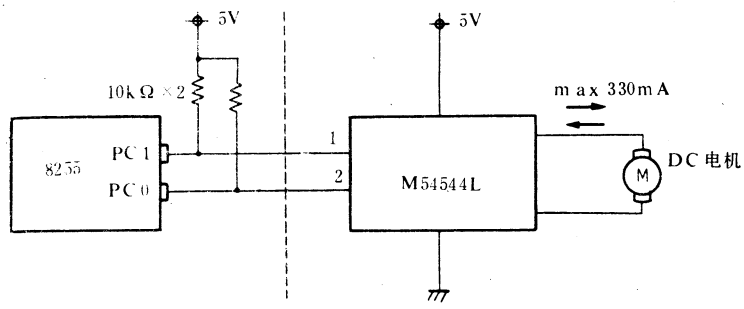
表 2 流程图及其程序清单

标号	助记符	地址	机器语言	注释
START	LD A,98H	00	3E 98	CW 置位
	OUT(03H),A	02	D3 03	
	LD SP 8100H	04	310081	SP 置位
	LD C,00H	07	OE 00	初始化
	LD D,08H	09	16 08	
	LD E,01H	0B	1E 01	
MAIN	CALL DISP	0D	CD 7F 00	楼层显示
	CALL INKEY	10	CD 4C 00	按钮输入(D)
	CALL INFR	13	CD 56 00	当前楼层输入(E)
	LD A,D	16	7A	比较
	CP E	17	BB	
	JPZ,STOPM	18	CA ZC 00	
	LD C,00H	1B	0E 00	门标志复位
	JP C,DOWN	1D	DA 27 00	
UP	LD A,F1H	20	3E F1	电机正转
J1	OUT (02H), AMA	22	D3 02	
	JP MAIN	24	C3 0D 00	
DOWN	LD A,F2H	27	3E F2	电机反转
	JP J1	29	C3 22 00	
STOPM	LD A,F3H	2C	3E F3	电机停转
	OUT(02H),A	2E	D3 02	

```
        LD A,C          30    79      }门标志是
        AND A           31    A7      }否为 0
        JP NZ,MAIN      32    C2 0D 00
        LD C,01H        35    0E 01   置门标志
        LD A,E          37    7B
        RLC A           38    CB 07   }
        RLC A           3A    CB 07   }左移 4 位，
        RLC A           3C    CB 07   }A 的低位
        RLC A           3E    CB 07   }移进高位
        ADD A,E         40    33
        OUT(01H),A      41    D3 01   }
        CALL TIMER      43    CD 83 00}用电磁铁
        LD A,E          46    7B      }打开门
        OUT 01H,A       47    D3 01   }
        JP MAIN         49    C3 0D 00
INKEY   IN A,(02H)      4C    DB 02   由 C 口输入
        CALL JUDGE      4E    CD 60 00
        JP Z,J2         51    CA 5F 00
        LD D,A          54    5F      按钮输入(D)
J2      RET             55    C9
INFR    IN A(00H)       56    DB 00   由 A 口输入
        CALL JUDGE      58    CD 60 00
        JP Z,J3         5B    CA 5F 00
        LD E,A          5E    5F      楼层输入(E)
J2      RET             5F    C9
JUDGE   CPL             60    2F      位反转
        AND FOH         61    E6 F0   低位无效
        RRA             63    1F      }
        RRA             64    1F      }高 4 位进
        RRA             65    1F      }入低位
        RRA             66    1F      }
        CP 01H          67    FE01    1 层?
        JP Z,J4         69    CA 7D 00
        CP 02H          6C    FE 02   2 层?
        JP Z,J4         6E    CA 7D 00
        CP 04H          71    FE 04   3 层?
        JP Z,J4         73    CA 7D 00
        CP 08H          76    FE 08   4 层?
        JP Z,J4         78    CA 7D 00
        LD A,00H        7B    3E 00
J4      AND A           7D    A7      A=0?
        RET             7E    C9
DISP    LD A,E          7E    7B      }
        OUT(01H),A      80    D3 01   }端口 B 输
        RET             82    C9      }出(E)的值
TIMER   LD H,FFH        83    26 FF   延时
J5      LD L,FFH        85    2E FF
J6      DEC L           87    2D1
        JP NZ,J6        88    C2 87 00
        DEC H           8B    25
        JP NZ,J5        8C    C2 85 00
        RET             8F    C9
```







输入		动作
1	2	
1	1	制动
1	0	正转
0	1	逆转
0	0	停止

1：高电平
0：低电平

图 8

上述软件表达的动作有：按动一次指定楼层开关；用光传感器检测电梯所处位置；与指定楼层进行比较；电梯升、降；厢门的开、闭等动作。对于多层楼的呼叫记忆与依次的动作；上升、下降时加速、减速的微妙控制；厢门的开闭消除不适感的同步方式等等还应该下功夫解决。

尤其是考虑对多架电梯群控管理，会深刻地体会到电梯控制的实用化是一件很了不起的事情。

编者注：M54544L 为日本三菱公司生产的 DC 电机控制电路，读者可根据图 8 状态表，可用开关三极管电路替代。

LD 磁盘文件查询程序

软件开发者：杜西延

LD 是英文 LABEL DISPLAY 的缩写，意为标识显示。曾经荣获一九八九年度军队级科技进步四等奖。它能在诸多的文件中迅速、准确、方便地找到自己所需的(ASCII 码)文件，是提高文件管理必不可少的工具软件。LD 不但能显示文件名、文件标示内容、文件字节数，而且显示文件具体内容、文件的个数和磁盘空间，它还有文件删除、子目录查询和标识打印等功能。LD 的操作采用屏幕对话方式，使用起来简单方便。

本软件已由电子工业出版社软件部出版

上述软件已由电子工业出版社软件部出版，欲购者请与北京 173 信箱软件部联系，邮编 100036。

CTE 通用表格编辑软件

软件开发者：张伟

CTE 是全屏幕编辑软件，用于表格和文书的编辑。它除具有同类文字编辑软件的各项功能外，还有如下特点：

1. 表格编辑是以一个窗口方框为一个基本操作单元，制表过程中操作人员始终不需要与表格线打交道，他所关心的只是方框中要输入什么内容以及方框所在的位置、方框的大小随输入文字的多少而变化。制表难度不随表格复杂程度的增加而增加，可编辑任意复杂的表格。

2. 可打印超过打印机宽度的表格。

3. 可对文件进行排版，版面宽度可在 40～240 间任意选择。

4. 提供了用光标画线抹线的方法。

初级程序员级水平考试辅导问答
汉字 dBASE Ⅲ 使用技巧(上)

王 路 敬

1. 运行 dBASE Ⅲ 以前,操作系统配置要作哪些改变?

在 MSDOS 或 PCDOS 操作系统下,必须对操作系统的两个缺省值进行改变。在一个名为 CONFIG · SYS 的 ASCII 文本文件中重新设定两个系统参数就可以做到这一点。

CONFIG · SYS 是启动 DOS 时自动执行的,它在 AUTOEXEC · BAT 之前执行,必须放在系统盘上。要改变的两个系统参数是:DOS 可以使用的缓冲区数目和 DOS 可以同时打开的文件数目。CONFIG · SYS 文件在 dBASE Ⅲ 文件应用中应包括这样两行:

BUFFERS=24

FILES=20

应当注意的是,每增加一个缓冲区就会使 DOS 在内存多占 528 个字节。由于这样会使 RAM 的开销增加,故在内存容量较小的情况下应设置较少的缓冲区。然而,如果想充分使用 dBASE Ⅲ,计算机的内存容量应允许设置一定数量的缓冲区。实践证明,设置 10～24 个缓冲区系统就可以运行较快了。

DOS2.0 以上各版本一般设置可打开的文件个数为 20。当超过约定值 8 以后,每增加一个文件就使 DOS 增加 39 个字节。在这些打开的文件中,DOS 保留 5 个分别用作标准输入、标准输出、标准错误输出、辅助设备和标准打印机。这就意味着,在整个系统里各种类型的文件最多可以同时打开 15 个,包括 dBASE Ⅲ 在内。dBASE Ⅲ 应用程序只可以打开 13 个文件。这是因为,dBASE Ⅲ 本身用了两个,一个供主程序用,另一个供覆盖文件用。

dBASE Ⅲ 运行时没有这个配置要求,但当缓冲区数目增加时,它的速度也会大大提高。文件数的调置并不影响 dBASE Ⅲ,因为它是直接处理文件控制块 FCB。因而允许在 dBASE Ⅲ 内同时打开 16 个文件,不包括 dBASE Ⅲ 本身在内。

汉字 dBASE Ⅲ 的运行环境在中文操作系统支持下,在启动操作系统时,CONFIG · SYS 文件同时被执行,从而完成了对系统的初始化设置。

2. 在 dBASE Ⅲ 系统盘上 CONFIG · DB 文件起什么作用?怎样建立和使用该文件?

在 dBASE Ⅲ 系统盘上的 CONFIG · DB 文件是对 dBASE Ⅲ 的运行环境作初始设置。CONFIG · DB 中的内容是一些具有特殊格式的命令行,当 dBASE Ⅲ 启动时,系统自动查找 CONFIG · DB 文件,若能找到,就执行 CONFIG · DB 文件中的命令,执行后回到命令状态;若找不到 CONFIG · DB 文件,则系统按既定(默认)初始化 dBASE Ⅲ,然后进入命令状态。

CONFIG · DB 是标准文本文件,可以用任一字处理或编辑程序建立,也可以在操作系统的提示符下键入以下命令来建立:

```
C>COPY CON:CONFIG · DB↙
    DEFAULT=C:↙
    TEDIT=EDLIN↙
    WP=WS↙
    ⋮
    ^Z↙(存盘)
```

CONFIG · DB 中的命令可以根据需要自行选择。

在实际应用中,CONFIG · DB 可以实现下面的功能:

(1)设置用户文件所存放的默认盘

命令格式:

DEFAULT=(盘符)

这样设置后,在 dBASE Ⅲ 的所有操作中,如果文件名前不加盘符,系统都将在所定义的磁盘上进行文件的存取。

(2)设置屏幕的颜色

命令格式:

COLOR=<标准显示>[,<加强显示>][,<边框>]

例如希望把屏幕设置成:标准部分为蓝色,其背景为黄色,加强显示时显示红色,背景为白色,并且处于闪烁方式下,边框为绿色,则:

COLOR=B/GR,*R/W,G

或者

COLOR=1/6,*4/7,2

(3)设置功能键

命令格式:

<功能键名称>=<命令>[;<命令>;…]

其中<功能键名称>指的是键盘上左面 F2 至 F10 键位,<命令>为 dBASE Ⅲ 命令,分号代表回车键。例如:

F2=USE CTSU;LIST;

该命令行定义按 F2 键一次等效于执行下列两条语句:

USE CTSU

LIST

可以用类似的方法定义其他功能键(F1 键不能定

义),以使得以后的键盘操作方便。

(4)设置 SET 命令的 ON/OFF 状态

例:TALK=ON

该命令使 TALK 为开,也就是说进入 dBASE Ⅲ 时就已能自动执行 SET TALK ON 语句。

(5)设置 dBASE Ⅲ 系统的最大可用内存

命令格式:

MAXMEM=<数值>

其中<数值>范围为 200K～720K,该命令为 dBASE Ⅲ 运行时规定了最大可用内存数范围。

例:设 IBM PC/XT 配置有 640K 的内存容量,希望给 dBASE Ⅲ 系统 540K,以便在 dBASE Ⅲ 系统中运行较大的外部高级语言程序。CONFIG · DB 命令行中可设置一条语句。

MAXMEM=540K

这种命令的使用随情况而定,有时使用 540K 内存运行外部文件时,系统还显示"无足够的内存空间"的提示。此时可以用 MAXMEM=640K 命令将可用内存设置到最大限度。

(6)规定内存变量可占用的最大内存空间。

命令格式:

MVARSIZ=<数值>

其中<数值>范围为 1K～31K,该命令为 dBASE Ⅲ 运行时规定了内存变量可占用的最大空间范围。

dBASE Ⅲ 允许内存变量为 256 个,空间为 6000 字节,但由于有时某些内存变量占据字节数太多,所用变量个数虽未超过 256 个,但总字节数已经超过了 6000 字节,在这种情况下需要首先扩大内存变量所允许使用的内存空间数目。

例:MVARSIZ=20K

此命令使内存变量可使用的内存扩大到 20480 个字节。保证程序的顺利执行。

(7)规定 dBASE Ⅲ 使用的编辑方式

命令格式:

TEDIT=<程序名>

WP=<程序名>

规定 MODIFY 命令所用的编辑程序和建立或修改明细型字段所使用的编辑程序。

我们知道 dBASE Ⅲ 可以用三种方法进行编辑(EDLIN),第三种是字处理程序(WORDSTAR)。未规定时,dBASE Ⅲ 默认编辑为它本身所带字处理程序,但我们可以用上述命令重新规定编辑程序。

例如:TEDIT=EDLIN

WP=WS

这样执行"MODIEY COMMAND<文件名>时,自动调 EDLIN 行编辑,编辑明细型字段时,装入字处理程序 WORDSTAR。

(8)进入 dBASE Ⅲ 系统后立即执行特定的命令。

命令格式:

COMMAND=<命令>

功能是执行列出的各<命令>

例:COMMAND=CLEAR

系统进入 dBASE Ⅲ 自动执行 CLEAR

在 CONFIG · DB 文件中,如果将上述命令放在一起就形成一个命令串。例如:用 EDLIN 建立 CONFIG · DB 文件,其内容如下:

```
DEFAULT=A
DECIMALS=3
F2=USE CTSU;LISI;
MAXMEM=640K
MAVARIZS=20K
TEDIT=EDLIN
WP=WS
COMMAND=CLEAR
```

进入 dBASE Ⅲ 后,执行 CONFIG · DB 文件,使得系统配置为:指定当前磁盘驱动器号为 A,小数点后取三位;F2 功能键代替 USE、LIST 命令;dBASE Ⅲ 最大可用内存为 640K;内存变量内存区为 20480 字节,修改命令文件时用 EDLIN 文本编辑程序。最后自动执行 CLEAR 命令。

3. 汉字 dBASE Ⅲ 在硬盘 C 的根目录上,信息管理系统在 A 盘上,如何设置才能正常工作?

当从 A 驱动器启动 CC－DOS 或者 PC－DOS 时,当前工作驱动器系统默认为 A。驱动器号可以为 A:、B:或 C:。

汉字 dBASE Ⅲ 系统若在 C 盘的根目录上,信息管理系统若在 A 盘上,为了调试 A 盘上的程序,我们利用定义默认盘的命令 SET DEFAULT TO 来实现,进行如下的操作:

```
C>DBASE↙
. SET DEFAULT TO A:↙
```

此后,A 盘即为系统 dBASE Ⅲ 的默认盘,调试程序时无需再指明盘号。

或者在 CONFIG · DB 文件中进行设置。设置的方法用字处理程序或编辑程序建立,或用其它方法建立。

```
例如:C>EDLIN CONFIG · DB↙
* I↙
1:DEFAULT=A:↙
2:↙
⋮ ^ Z↙
* E↙
C>
```

一旦对 CONFIG · DB 文件进行改变,需要重新启动 dBASE Ⅲ。

4. 在进行信息查询时,被查询的内容确实存在,但有时就是查不到,为什么?

出现这种情况往往是由于当前盘和系统定义的默认盘不一致而造成的。每当汉字 dBASE Ⅲ 被启动之后,汉字 dBASE Ⅲ 即自动地在其所在盘上寻找是否有 CONFIG · DB,若有这个文件,则首先执行 CONFIG ·

DB 所包含的各条命令，执行完，进入汉字 dBASE Ⅲ 命令状态；如果没有，便立刻进入汉字 dBASE Ⅲ 命令状态。若被查询的内容确实存在，但又查不到，解决的方法是将当前盘和系统默认盘取得一致，或者被查询的文件放在默认盘上，就不会发生查找不到的现象。

5. 运行汉字 dBASE Ⅲ 程序时，在什么情况下会出现“内存空间不够”的提示信息？

汉字 dBASE Ⅲ 的运行环境，最好要有 640KB 内存空间的支持，但在 512K 内存空间的条件下，基本上也可以运行。不过在这种情况下，必须小心谨慎地进行程序设计，否则，在程序运行过程中，系统会提示“内存空间不够”，程序再无法向下运行。在信息管理系统实际开发过程中，出现“内存空间不够”的提示，归纳起来有以下三种情况：

(1)利用 RUN 命令执行 CC－DOS 命令。因为 CC－DOS 命令是编译过的目的程序，它的执行需要将目的程序一次全部调入内存空间，因为内存空间已经存放了汉字 dBASE Ⅲ。

(2)命令文件中，出现语法错误。汉字 dBASE Ⅲ 再将“错误信息提示”模块调入内存空间，这种复盖模块要比其他复盖模块的容量大，因此，也往往出现“内存空间不够”的错误信息提示。

(3)同时打开多个文件，特别是又要打开索引文件，这时也会出现上述情况。

所有这三种情况，只要我们在程序设计过程中认真处理好，在 512K 内存空间的条件下，仍然是可以运行汉字 dBASE Ⅲ 的。

6. 当需要打开多个数据库时，dBASE Ⅲ 系统显示“打开文件太多”的提示信息是怎么回事？如何解决？

出现这个问题主要原因是没有修改中文操作系统 CC－DOS 中的系统结构设置文件 CONFIG · SYS 内容。

若 CC－DOS 中的 CONFIG · SYS 原来的内容为：

DEVICE＝ANSI · SYS

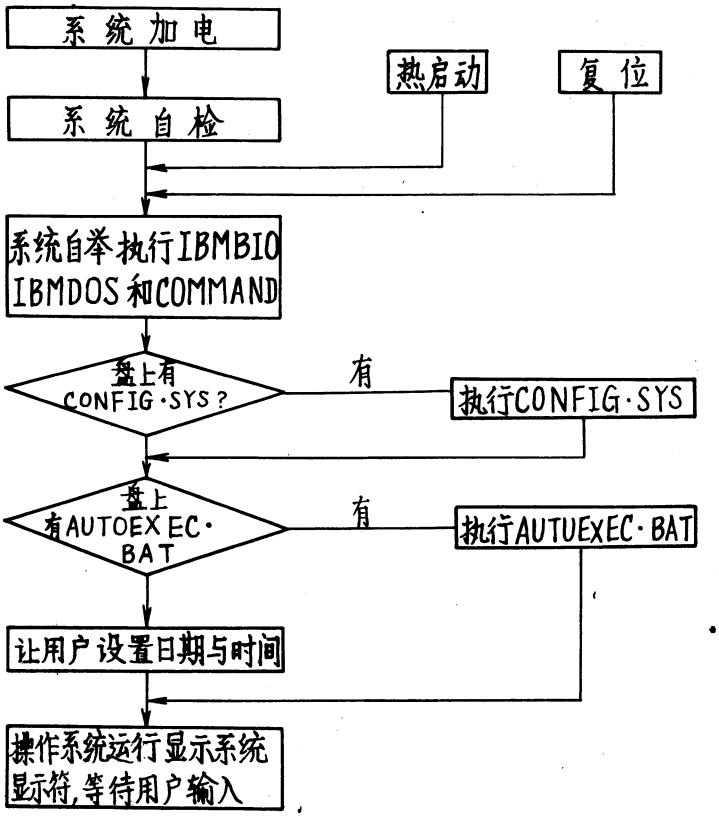
应修改为：

DEVICE＝ANSI · SYS

FILES＝20

BUFFERS＝24

在未经设置的情况下，DOS 同时可以打开的文件个数为 8，而扩充的输入输出操作将占去 5 个文件，留给 dBASE Ⅲ 用的文件个数只有 3 个，显然这是不够的。运行 dBASE Ⅲ 需要设置 FILLES＝20，此时汉字 dBASE Ⅲ 可以打开各类文件 15 个；不仅如此，修改后的 CONFIG · SYS 还必须和 CC－DOS 放在一张磁盘上，否则，CONFIG · SYS 仍然不起作用，这是由于 CC－DOS 的启动过程所决定的。CC－DOS 的启动过程如图所示。



从上图可以清楚地看到，CONFIG·SYS 放在其他磁盘上，例如放在汉字 dBASE Ⅲ 系统盘上，CC－DOS 启动过程中并不执行此文件，必须放在 CC－DOS 启动盘上。

7. 在硬盘不同子目录下运行 dBASE Ⅲ 命令文件有什么缺点？怎样克服？

在操作系统下先进入某个子目录，调用 dBASE Ⅲ 系统，然后再运行另一个子目录下的 dBASE Ⅲ 命令文件。若需要继续运行另一个子目录下的 dBASE Ⅲ 命令文件，往往是退出当前的 dBASE Ⅲ 状态，退出该子目录，然后再进入另一个子目录，调入 dBASE Ⅲ，再运行该子目录下的 dBASE Ⅲ 命令文件。这样多次调用 dBASE Ⅲ 会浪费用户的大量时间，而且在不同的子目录下至少要存放两个 dBASE Ⅲ 系统文件：DBASE.EXE 和 DBASE.OVL，占去了硬盘不少的空间(约 260K)。

解决这种问题的方法是调用根目录下的 dBASE Ⅲ。进入 dBASE Ⅲ 状态，然后进入子目录，运行该子目录下的 dBASE Ⅲ 命令文件。若继续运行另一子目录下的命令文件，退出当前目录状态，进入该子目录，运行该子目录下的 dBASE Ⅲ 命令文件。但在操作时，为保证 dBASE Ⅲ 系统只调用一次，在进入和退出子目录操作中要在 dBASE Ⅲ 系统下进行，为了做到这点使用 dBASE Ⅲ 的 RUN/! 执行 dBASE Ⅲ 外部程序的命令。这样就要求内存容量大于 640KB 的机器。

8. 多个子目录怎样公用 dBASE Ⅲ？

可以这样进行操作：

(1)把 dBASE Ⅲ 系统盘插入 A 驱动器，并将 dBASE Ⅲ 文件复制到硬盘的根目录下。

C>COPY A:*.* C:

(2)进入 dBASE Ⅲ

C>DBSAE

(3)进入所要的子目录

.RUN CD/子目录名

这样即进入了 dBASE Ⅲ 又进入了所要的子目录中。很显然要使硬盘上的若干子目录公用一个 dBASE Ⅲ，借助于 RUN 命令即可解决。

9. 在 dBASE Ⅲ 系统下如何对子目录文件进行操作？

为了避免一机多人使用造成数据和文件丢失的现象，常使用的办法是建立子目录文件。使用子目录对子目录文件操作运行速度也较快。在 dBASE Ⅲ 命令状态下对子目录文件进行操作可采用以下几种方法：

(1)在 dBASE Ⅲ 园点"·"提示符下，建立子目录操作：

.RUN MD\一级子目录文件名\二级子目录文件名

假若一级子目录文件名为 DBASE，二级子目录文件名为 ABC，其操作方法是：

.RUN MD\DBASE\ABC

(2)使子目录上的文件或运行程序变为当前目录操作：

.RUN CD\DBASE\ABC

若对文件直接进行编辑，可进行如下的操作：

.MODIFY COMMAND 文件名

若运行程序，可进行如下操作：

.DO 文件名

(3)直接对子目录进行操作

若编写程序文件操作如下：

.MODIFY COMMAND\DBASE\ABC\文件名

若运行程序，其操作如下：

.DO\DBASE\ABC\文件名

若打开数据库文件，其操作如下：

.USE\DBASE\ABC\文件名

.LIST

(4)使用批处理直接进入子目录

如果当前目录没有批处理文件时，可以随时建立批处理文件。其操作如下：

.MODIFY COMMAND 文件名.BAT

CD\DBASE\ABC

CLS

批处理文件建立好之后，在以后的使用中十分方便，只需在 dBASE Ⅲ 命令状态下键入 RUN 命令，批处理文件的文件名，而后打回车键就可进入子目录。其操作如下：

.RUN 文件名(这里的文件名指批处理文件名)

10. 使用建库命令时，文件的扩展名是否必须为 .DBF？

不必。在使用建库命令 CREATE 时，若省略扩展名，则系统自动为其加上扩展名.DBF。用户也可以用别的扩展名，但在使用库文件时，扩展名部分不能省略。为简便起见，用户一般不必另起扩展名。

11. 数据库文件结构的特点是什么？

DBF 文件由两大部分组成。第一部分是文件结构的说明，第二部分是实际的数据。

(1)文件结构说明部分

文件结构说明部分的长度为：(文件字段数＋1)×32＋2。前 32 个字节存放文件结构中除字段描述信息外的其它数据。有关字节的内容及含义见表 1－1。

表 1－1　数据库文件结构

字　节	内　容	含　义
1	83H 或 03H	DBF 文件标识符，含 M 型字段为 83H，否则为 03H
2－4	日期	最后一次修改 DBF 文件的日期，依次为年、月、日
5－8	记录数	DBF 文件中当前记录个数，低位在前，高位在后
9－10	地址	DBF 文件实际数据的相对起始地址
11－12	记录数	存放记录长度，低位在前，高位在后

从第 33 字节开始，存放各段说明部分，每字段占 32 个字节，有关字节内容及含义见表 1－2。

表 1－2　字段说明部分有关字节内容及含义

字　节	内　容　及　含　义
1－10	存放字段名，不足 10 个后补 00
12	类型标志(N,C,D,L,M)
17	字段宽度
18	若字段为实型，存放小数位数，否则为 00

文件结构说明部分除字段名和类型标志用 ASCII 码表示外，其余均为 16 进制数。文件说明部分以 ODH,OOH 为结束标志。

(2)数据部分

DBF 文件的实际数据紧接 ODH,OOH 之后存放。每个记录的第一个字节为记录删除标记，若用 DELETE 命令删除记录(未用 PACK 命令)，则其内容为 2A(＊的 ASCII 码)，否则为 20(空格)，其后连续以 ASCII 码形式按顺序和长度存放各字段数据，其中字符型的数据是左对齐，数值型的数据是右对齐，不足字段长部分用空格对齐。每个字符占一个字节，数据部分以 1A(CTRL－Z)结束。

知道 dBASE Ⅲ DBF 文件中数据安排后，就可以用调试程序(DEBUG)查看或修改有关字节的内容，达到恢复 DBF 文件(用 USE 可以打开)和恢复数据的目的；也可用改变 DBF 文件标识字，记录计数器或其它有关字节的内容给 DBF 文件加密。

12. **建立数据库可通过哪些途径？**

建立数据库可通过多种途径，归纳起来有三种方法，即直接用 CREATE 命令建立新库；用 COPY 命令将已建旧库改造成新库；将各高级语言建的数据文件变换成新库。分 6 种途径：

(1)第一次建新库

使用 CREATE 命令即可，这是大家比较熟悉的一种方法。

(2)由已建立的旧库改造成新库

①照旧库原样复制出新库

格式：

. USE＜待复制的旧库名＞

. COPY TO ＜新库名＞

②复制旧库的结构到新库

格式：

. USE＜被复制的旧库名＞

. COPY TO ＜新库名＞ STRU

③仅复制旧库中的几个字段到新库

格式：

. USE＜被复制的旧大库名＞

. COPY TO ＜新库名＞ STRU FIELDS ＜字段名清单＞

④复制部分结构连同部分记录到新库

格式：

. USE＜旧库名＞

. COPY TO ＜新库名＞ FIELDS ＜字段名清单＞＜FOR(条件)＞

⑤保留旧库全部内容，同时需增加字段名

格式：

. USE ＜旧库名＞

. COPY TO ＜备份库名＞

. MODIFY STRU

. APPEND FROM ＜备份库名＞

. ERASE ＜备份库名＞

⑥保留原有数据，并改变字段名

格式：

. USE ＜旧库名＞

. COPY TO ＜暂存外部文件名＞SDF

. MODIFY STRU

. APPEND FROM ＜暂存外部文件名＞. TXT SDF

. ERASE＜暂存外部文件名＞. TXT

(3)由两个相关的数据库联接产生第三个新库。

格式：

. SELECT 2

. USE＜第二个库名＞

. SELECT 1

.USE ＜第一个库名＞

.JOIN TO ＜新库名＞FOR＜条件表达式＞

(FIELDS ＜字段名清单＞)

(4)由一个旧库产生一个同类项合计新库

格式：

.USE＜旧库名＞

.INDEX ON ＜关键字段名＞ TO ＜索引文件名＞

.USE ＜旧库名＞ INDEX ＜索引文件名＞

.TOTAL ＜新库名＞ ON ＜关键字段索引＞(FOR ＜表达式＞)(FIELDS＜字段名表＞)

(5)从间接数据库产生新库

格式：

.CREATE＜新库名＞ FROM ＜间接库名＞

间接库名可用如下格式建立：

.USE ＜旧库名＞

.COPY TO ＜间接库名＞ STRU EXTE

(6)更改旧库名而变成新库

格式：

RENAME ＜旧库名＞ TO ＜新库名＞

建立数据库的时候，用户可根据应用的具体情况，选择上述建库的途径。

13. 有时打开 dBASE Ⅲ 数据库时屏幕显示“不是数据库文件”的信息，而用 DOS 的 DIR 列目录时仍为数据库文件，扩展名为 DBF，原因何在？

产生这现象的原因很可能是：

在用 USE 命令打开此数据库前，曾用过 dBASE Ⅲ 的 EDIT 命令修改过记录内容，或用 APPEND 命令追加了记录，亦即往数据库中写入了内容，随后即没有用 USE 命令关闭此数据库文件，也没有用 QUIT 命令退出 dBASE Ⅲ。那末在下一次打开此数据库时，dBASE Ⅲ 就可能不认为这是数据库文件了。

顺便说明一下，USE 命令不带任何参数时，为关闭原先打开的数据库文件。QUIT 命令用于关闭所有数据库文件和命令文件，并回到操作系统。

14. 在使用汉字 dBASE Ⅲ 时数据库数据的丢失是怎样造成的？如何解决？

数据库数据的丢失常常是由于下列几种情况造成：

(1)没有关闭数据库文件

在 dBASE Ⅲ 中，输入和修改数据常用的命令有：APPEND (BLANK)，INSERT (BLANK) (BEFORE)，BROWSE，DELETE，PACK 等。这几条命令有一特点，就是每执行一条新的命令后，才将前一命令的执行结果写盘。如果不了解这一点，只顾修改，输入操作，而没考虑到数据是否真正存盘，就会出问题。比如，若最后执行 APPEND 命令，执行完后就抽出盘片或关机，那么由 APPEND 输入的的数据没有写盘，这部分数据就将丢失。解决的办法是在最后增加一个关闭现有数据库文件的操作。

(2)使用 INSERT(BEFORE)命令时，对 dBASE Ⅲ 1.0 版会丢掉最后一条记录。

解决办法是在执行 INSERT(BEFORE)之前，先执行 APPEND BLANK 命令。

(3)使用 INSERT(BEFORE)没有建立正确文件结束标记。

dBASE Ⅲ 1.00 版本中，执行 INSERT(BEFORE)不能建立正确的文件结束标记。

例如：

.USE＜库文件名＞

.GOTO BOTTOM

.DISPLAY 姓名，性别

.APPEND BLANK

.INSERT BEFORE

.GO BOTTOM

屏幕上显示：

end of file encounted

goto bottom?

Do You Want Some help(Y/N)?

由上可知，在执行 INSERT BEFORE 后，发出 GO BOTTOM，显示出错误信息。因为 dBASE Ⅲ 对所有非索引文件，在执行 GO BOTTOM 后，指针指向最后一条记录，但 EOF()为.F.。EOF()为真的条件是指向最后一条记录+1。解决办法是在执行 INSERT(BEFORE)后，再执行一条 PACK。

(4)有时用 DIR 查目录时，看到文件的记录数正确，但又找不出全部记录，指定记录号查询或显示时，又出现“记录超出范围”的错误信息。解决办法是用 APPEND BLANK 命令找回数据。

15. 打开库后发现记录数少若干怎样处理？

一般可采用这样的步骤：

.USE ＜库文件名＞

.GO BOTTOM

.SKIP

若能用 LIST 显示下面的记录则修复成功：

若不能奏效可以重复执行 SKIP 命令。若仍不成功时则可用：

.USE ＜库文件名＞

.GO BOTTOM

.DELE

.PACK

若还不成功，调用 DEBUG 调试程序，查看被破坏的数据库的内存映象，从中常常可以找到问题的所在。操作步骤：

(1)将破坏的数据库拷贝到一个格式化好的空白盘上，并将该文件改名。

(2)调用 DEBUG 程序，查看，找出问题所在，修改、存盘、退出。

(3)复原文件名。

(4)进入 dBASE Ⅲ。打开库文件,显示,即可检查是否修复成功。

16. 向数据库录入数据有哪些方法?各有何特点?

数据库文件结构建立之后大量的工作是向库中输入数据,录入数据的方法有以下几种:

(1)建立数据库文件结构的同时录入数据

当数据库结构建立完毕,dBASE Ⅲ系统询问:现在就输入数据记录?(Y/N)。这时如果按"Y"键,dBASE Ⅲ则显示第一个记录的全部字段名,光标停留在第一个字段名后,我们就可以一个字段,一个字段地,一个记录,一个记录地输入数据,当输到明细型字段时,按 CTRL+PgDn 键则进入 dBASE Ⅲ字处理,在字处理方式下,便可给明细型字段录入数据,录入完毕,按 CTRL+W 则退出文字处理状态,输入下一个字段数据,如此这般进行下去。这种方法适用于初学者。

(2)简易录入法

所谓简易录入法就是利用 APPEND 命令在当前工作库文件或索引文件末尾增加一些新的记录,一次增加记录的数目是任意的,只要总的库文件内容中超出允许的最大容量就行。

操作:

. USE <库文件名>

. APPEND

采用全屏幕方式向库文件或索引文件添加新的记录。

(3)调用格式文件录入法

利用 dBASE Ⅲ的格式文件(. FMT)的方法,就是采用 APPEND 等命令调用格式文件的方法。这种方法好象以卡片形式向库文件添加新的记录,这样,显示的画面比较直观、舒适、漂亮。

操作的步骤:

①打开库文件。

. USE<库文件名>

②打开格式文件

. SET FORMTA TO <格式文件名>

③用 APPEND 命令调用格式文件录入数据。填完最后一个数据后,按回车键,表示一一个记录输入完毕,dBASE Ⅲ则显示下一个记录,再重复上述输入数据的操作。

(4)修改录入法

修改录入法就是将 APPEND BLANK 和 READ 配合起来使用,先向库文件追加一个空记录,然后再用 READ 命令读入键盘输入的数据,并用此数据修改空记录的内容。

操作步骤:

①打开库文件

. USE<库文件名>

②追加一个空记录

. APPEND BLANK

③打开格式文件并用 READ 命令读入一组数据

. SET FORMAT TO <格式文件名>

. REAN

如果还要追加数据,再追加一个空记录,重复上一次的操作过程。这样追加的记录数可以是任意的,如果编写一个循环程序,就可实现数据的自动录入。

这种方法要注意的是用 READ 命令不能进入 dBASE Ⅲ字处理状态,所以无法向明细型字段录入数据。

(5)传递快速录入法

这种方法是把上一个记录的内容传递到下一个记录里。如果这两个记录的内容完全相同,那么,把上一个记录传递过来以后,只要将不同的部分稍作修改,就实现了一个记录的数据输入。从而提高了数据的输入速度。

操作步骤:

①打开库文件

. USE<库文件名>

②设置记录传递开并

. SET CARRY ON

. APPEND

这种传递功能很有实用价值,如果两个相邻记录有很多数据项内容相同,只要对上一个记录作少许修改,就可得到下一个记录。

如果将记录传递开关设置成关闭状态,dBASE Ⅲ将不进行记录的传递,在屏幕上显示的是空记录。

例如进行如下的操作:

. SET CARRY OFF

. SET FORMAT TO <格式文件名>

. APPEND

在屏幕上仅显示一个空记录。

17. 加快数据输入有哪些措施?

一个数据库文件的建立,归纳起来无非是做两个方面的工作:一是建立数据库文件的结构;二是输入数据。当输入数据多,尤其输入汉字信息多时,加快数据输入的速度感到尤为必要。实践证明可以从以下几个方面考虑:

①使用 SET FUNCTION 命令

这个命令用来定义功能键的功能。键盘的 F1~F10 10 个功能键除 F1 不能改变其功能外,其他几个功能键均可由用户使用 SET FUNCTION 命令来改变。该命令使用的格式:

SET FUNCTION <字数> TO <字符串>

每个被改变的功能键的功能可以定义成含有 30 个字符的序列,这些字符可以是汉字,数字及各种符号。这样按一次功能键便可以输入一串预先定义的字符。9 个功能键即可以预先定义成 9 个常用字符串。例如,若数据库文件有"单位名称"这个字段,那么可以预先把 9 个常出现的单位名称定义在 9 个功能键上,当光标在字段"单位名称"下时,按下某一定义后的功能键,即可一次输入这个字段。功能键所定义的内容可根

据需要及时更换。

②利用替换命令 REPLACE

输入数据时，常常有许多字段内容相同，若输入的数据是以汉字形式出现，输入工作就更麻烦，可以用简单的符号来代替，最后用 REPLACE 命令来更改。

例如，“大豆品种资源数据库”中有一字段“花色”，在这字段下的数据要么输入“紫花”，要么输入“白花”，在输入时“紫花”处用“Z”代表；“白花”处用“B”代替，2000 个记录的数据都输入完后，最后，用 REPLACE 命令进行替换。这样可节省很多输入数据的时间。

上述问题的实现可以这样做：

USE＜库文件名＞

REPLACE ALL 花色 WITH “紫花” FOR 花色＝“Z”

REPLACE ALL 花色 WITH “白花” FOR 花色＝“B”

③应用 SET CARRY ON 命令

此命令可以免去输入相邻两个记录字段内容相同的数据。执行 SET CARRY ON 后，当执行 APPEND 命令时，将把前面一个记录的数据带到新记录中，当执行插入命令 INSERT(BLANK)时，将把当前记录的数据带到新记录中去。这时输入新记录的操作就变成了修改记录中的有关数据了。

④应用 APPEND FROM 命令

APPEND FROM ＜文件名＞ (FOR＜条件＞),(SDF)

(DELIMITED)命令是从其他文件追加记录到当前数据库文件之中。利用这一特性，对需要输入数据的数据库中某些字段及其内容与原有的数据库的对应字段及其数据相同时，不必重新输入这些数据，只需用上述命令，将这部分数据追加到已打开的数据库中，然后对其他字段输入相应的数据即可。

18. 明细文件的扩展名是否一定为. DBT?

是的。明细文件的扩展名也是系统自动加的。当用户所建的库中有明细型字段时，存盘后就自动在盘上生成一个与库文件名同名其扩展名为. DBT 的明细文件。

19. 明细字段里的空格是否占用空间?

是的。明细字段里的 1 个空格占 1 个字符的空间。因为明细字段的数据均可以用一段文字来表示，而这段文字是按 512 个字节为一个数据块来存放的。明细字段的数据最多允许 4K 字符，即 8 个数据块。当写入的文字为 512 个字节时，系统就为此明细字段分配一个数据块。当写入的内容大于 512 个字节而小于 1024 个字节时，系统就分配两个数据块。

20. 使用 Memo 明细型字段节省存储空间吗?

这要看 Memo 型字段中的内容而言。如果 Memo 型字段中的内容较少，则并不节省存储空间。这是因为，一旦定义一个 Memo 型字段，则不管其中有无内容，系统都要开销 512 个字节的存储空间。也就是说，又要有一个. DBF 文件，哪怕它是空的，也至少要占踞 512 个字节的盘空间。dBASE Ⅲ 为 Memo 型字段分配磁盘空间是以块为单位进行的。每个块为 512 个字节的 1 个扇区。最多可有 8 块，即 4096 个字节。

21. 怎样输出 Memo 明细型字段的内容?

dBASE Ⅲ 在数据库文件中增加了 Memo 型字段，这对处理内容较多且长短相差较大的字符数据提供了方便。但是，dBASE Ⅲ 提供的输出 Memo 字段的手段较少，能显示或修改 Memo 型字段的命令只有 6 条，即 EDIT，CHANGE，?，DISPLAY，LIST 和 REPORT。其中，EDIT 和 CHANGE 主要用于编辑和修改 Memo 型字段的内容。?、DISPLAY 和 LIST 主要用于输出 Memo 字段的内容。这 3 条命令中，? 命令更适合于在程序设计中使用。只要在 ? 命令中给出 Memo 型字段的名字，便可输出其内容。如果用 ? 命令造表，则可以在表格中输出 Memo 型字段的内容。REPORT 命令是专用于输出报表的。在用该命令建立报表格式时，只需针对要求输入栏目内容的提问键入需要输出的 Memo 型字段的名字，便可在报表中输出 Memo 型字段的内容。

22. 怎样使用行编辑程序或字处理软件 WORDSTAR 编辑 Memo 字段?

dBASE Ⅲ 提供了自己内部的字处理程序用以编辑 Memo 型字段，但它的功能有限，最多只能编辑 4096 个字节的文件。dBASE Ⅲ 还提供了另一种方法：在不退出 dBASE Ⅲ 的情况下调用 EDLIN 或 WORDSTAR。这是通过在 CONFIG · DB 文件中设置如下两个参数来实现的：

WP＝＜外部程序名＞

TEDIT＝＜外部程序名＞

WP 和 TEDIT 定义了用于编辑 Memo 型字段的外部程序。在用 EDLIN 或 WORDSTAR 编辑 Memo 型字段的情况下，这两个参数设置如下：

WP＝EDLIN

或

TEDIT＝WORDSTAR

但是，一旦进行了上述设置，在 dBASE Ⅲ 系统下打入：

MODIFY COMMAND ＜文件名＞

正常情况下应当进入到相应的编辑状态，或者是行编辑状态，或是字处理状态。不过有时候出现这样一种现象，执行上述命令后没有进入到所指定的编辑状态，而是又返回 dBASE Ⅲ 命令状态，即圆点提示符的状态。这是为什么呢？当出现这种现象时候要检查所指定的编辑软件是否在指定磁盘上存在，若指定的磁盘上不存在该软件，就返回 dBASE Ⅲ 圆点提示符状态等待。

维修经验谈

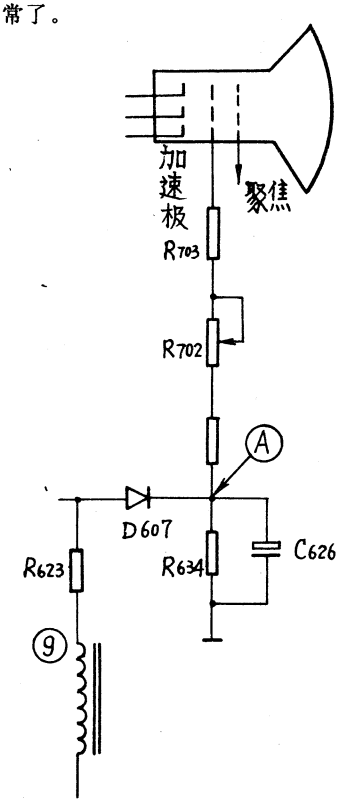
彩色显示器亮度失控的探讨

中国人民大学信息中心　胡野红

例 1:亮度失控

亮度失控是显示器常见故障之一,而造成这种故障的原因大部分都是忽视了机器的使用环境如:机房温度过高或湿度过高。也有少量的显示器故障是由于设计不合理或使用了质量不好的元器件所致。本文对后一种原因所引起的故障结合两个实例予以分析,供大家参考。例 1. 故障现象为亮度下不去,调节亮度电位器不能使屏幕的亮度发生变化此故障多次出现在 BRI—UP 彩显中。

分析与维修:首先检查了亮度控制电路没有发现异常现象,再检查显象管加速极电路发现该极电压高达 700V,而正常工作时该极电压为 420V,由以下附图可以看出,加速极电压出自高压包 T 的⑨端经 R623 限流,D607 限流 R634 和 R701 分压而得。进一步检查发现 A 点电压为 640V,而正常时应为 200V。这时可以明显看出,故障范围在 A 点以下电路,结果是 R634 开路和 C626 失效。换上同等参数的元件后该彩显工作正常了。



该彩显修好用了一段时间后旧戏重演,再次出现亮度失控现象,经查还是 C626 失效,原电路中 C626 为 4.7μf/200V 而在实测中 A 点正常工作电压为 200V。很明显 C626 耐压不够。由电工原理而知 C626 的耐压应选$\sqrt{2}$ 200(V)=282(V),故选用标称值为 4.7μf/300V 的电容换到该显示器上后,彩显一直正常工作至今未发现这种故障。

仔细分析这部分电路可以发现:①高压包 T⑨端的另一端接行输出管,而行输出的行脉冲虽经 D607 整流,但到达 A 点时还是脉动直流,对 C626 形成冲击。②由于高压包 T 是储能元件,每次开关机时均会产生数倍于正常工作电压的反电势,形成对 C626 的冲击。又由于反电势虽然电压高,但能量小,不至于数次开关机就损坏 C626,而长而久之的冲击使 C626 失效而起不到滤波作用。所以由以上事实可以得出结论:由于原线路对 C626 的耐压选择不当,造成 C626 失效、分压电阻 R634 开路 A 点电压升高,使显像管加速极电压脱离工作范围而引起显示器亮度失控。

例 2:故障现象为刚开机时光栅抖动呈浅绿色且光栅行幅不满,调节亮度电位器光栅亮度无变化,几分种后工作正常。

显示器牌号为 BRI—UP

分析与维修:先检查亮度控制电路未发现异常现象,再用万用表测机内直流电源输出端,发现各挡电压均在低于额定电压 30V 内随光栅抖动而变。故怀疑电源带载能力差。用替换法处理证明电源无问题。又怀疑行扫描电路有问题,结果行管、逆程电容等有关元器件均完好无损,断开行输出后电源电压工作正常,这说明短路现象发生在行输出以后部分。而行输出的最大负载在显像管,因为屏幕为浅绿色故怀疑 G 极(显像管)线路有问题,经仔细检查,G 极线路无问题。当把 G 级引线从显像管尾部线路板上焊下时,光栅满幅电源电压也工作正常了。所以怀疑显像管本身有问题。用摇表测 R、G、B 极对地电阻,发现 G 极电阻明显小于其它两极。换同一规格的显像管后,彩显工作正常工作了。可以看出,由于显像管 G 极对地绝缘性能变差,当开机初始、呈短路状态,使电源电压下降、光栅不满、亮度失控。随着时间的推移,显像管 G 极极板温度升高,电阻逐步增大到正常值使彩显正常工作,而关机后显像管极板逐渐变凉电阻也逐渐变小,而再次开机时,重现故障。

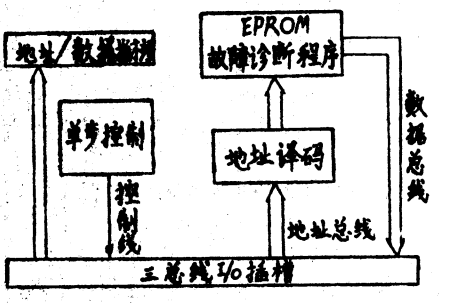
微计算机的维修工具——8088系统板故障的诊断卡

董 琳

微型计算机的系统板是整个计算机的核心。板上有8088CPU、ROM和RAM、DMA控制器，中断控制器，定时控制器，并行接口控制器及总线控制等多种功能，这些功能大多采用集成电路芯片实现。当系统板出现故障时绝大多数情况要造成"死机"，一旦"死机"，系统板上多数芯片的输入、输出状态都不正常，尤其对那些大规模集成电路芯片很难确定故障点，给故障修复工作造成很多困难，本刊第5期推荐的"8088系统板故障诊断卡"是维修IBM PC/XT及其兼容机系统板的实用工具，对于初次从事维修的同志，它将使你很快胜任常见故障的修复工作，对于已从事过系统板维修的同志，它将会提高你的工作效率。

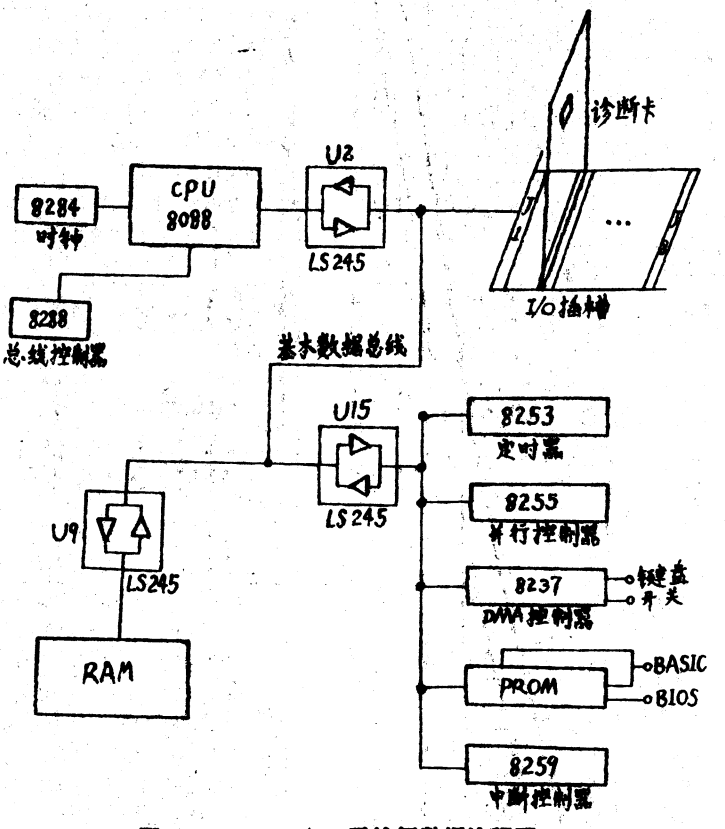
一、工作原理

8088系统板故障诊断卡(以下简称诊断卡)主要包括三部分电路，(一)存放诊断程序的EPROM芯片；(二)地址译码电路，(三)单步控制电路。三部分框图如一所示。



图一 诊断卡原理图

其工作过程是：从I/O总线插槽取得首地址，经地址译码器译码，选中故障诊断程序相应的地址单元，然后读取该地址的内容，这一定是诊断程序第一条指



图二 IBM PC/XT系统板数据流程图

令的内容，经数据总线送往 I/O 插槽，CPU 读取并执行该指令又送出下一个地址，取出第二条指令…如此循环。这样 EPROM 中的诊断程序将自动地诊断系统板上各部分的故障情况，并且直观地显示在屏幕上。当系统板上某些故障（约占系统板故障的 25%）影响到诊断程序正常运行时，为了进一步查找故障源，诊断卡上的单步控制线路可完成这一工作，在单步工作方式下，每执行一个总线周期停机一次，这样我们可以从地址/数据指示灯上清楚地看到地址/数据总线的故障，也可从中分析控制总线的故障。只要排除了总线故障，便可重复上述连续工作方式，全面地检查系统板上其它各部分故障。

二、从维修角度看系统板的结构及诊断卡位置

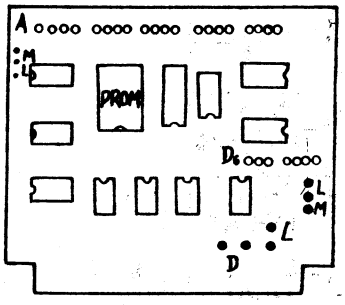
从图二中我们可得到哪些信息呢？首先看 U2，在非 DMA 方式下，数据传输由 8088CPU 控制，所有的数据输入，输出都要经过 U2；CPU 与 I/O 插槽中控制卡的信息联系要经过 U2，CPU 对 RAM 的数据读，写首先经过 U2 再经 U9 到达存储器，同样，CPU 对 PROM，8253、8255 的访问也要经 U2 和 U15，就是在 DMA 操作过程的开始，CPU 也必须经 U2 和 U15 对 8237DMA 控制器进行初始化。可见，U2 的好坏将影响整个系统板上其它各模块的正常工作，故该芯片尤其重要。

从诊断卡在整个图二中的位置可以看到，只要保证基本总线是好的，诊断卡就可以正常工作，直接对 8253、8255、8237、8259、RAM 等各部分进行故障诊断。但是必须封锁 CPU 对原 PROM 的访问。

利用图二我们还可以分析某些故障现象的故障源。如对系统板的诊断结果为 8237、8259、8255 均有故障，一般说几个芯片同时损坏的情况机会不多，可能性较大的是 U15，这在实际维修工作中会少走些弯路。

三、诊断卡的运行启动

诊断卡是系统板的维修工具，因此，在维修有故障的系统板之前先要保证诊断卡本身是完好的，所以有必要先在一个好系统板上学会正确地使用诊断卡的方法，并验证诊断卡是否完好。



图三　诊断卡外形图

在诊断卡插入 I/O 槽之前，先检查一下两种工作方式跳线开关的位置。若将跳线连接右下角 L 两点，则为单步工作方式；若跳线连接的是 D 两点，这时将工作在连续诊断方式。如图三所示。

左上角若连接跳线在 M 两点，地址发光二级管灭，若连接跳线在 L 两点，发光二级管亮。右中下位置若连接跳线在 L 位置，数据发光二级管灭，若连接跳线在 M 位置，数据发光二级管亮，一般情况要保证发光二级管亮，以便观察地址、数据总线的状态。

在跳线开关接好之后，诊断卡可以插在系统板的任一个 I/O 槽中（PC/XT 的 I/O 插槽 J8 除外）。但要注意：诊断卡的元件面要与其它卡的元件面方向要一致，千万不要插反，以免烧坏元件。

如果诊断卡上的 EPROM 芯片是 CGA 方式显示的，这时系统板上应插一块 CGA 卡，并连接相应的显示器；如果诊断卡上的 EPROM 芯片是单色图形方式显示，则应在系统板上插一块单色图形卡并连接单色显示器。两种 EPROM 在拔插替换时也应特别注意：EPROM 芯片不要插反，否则将破坏里面的诊断程序。

为了保证诊断卡上的 EPROM 能正常工作，还应该将系统板上原 PROM 芯片“置死”。实现方法是：PC 机，在系统板上将 U46(LS138)－5 脚或 U64(LS20)－6 脚断开；在 PC/XT 机上，位于系统板上靠近键盘插座附近，有两个用白色框线围起来的焊点，标称 E1(1、2)，在 PC/XT 系统板图纸上的第三项标以 E7(1、2)，将两点短路即可；在 0520CH 机上，将系统板上原 B105 芯片标有 01 的 PROM 芯片拔下，换上专为 0520CH 所写的 EPROM 芯片，即可完成对 0520CH 系统板的测试诊断。其它一些 XT 兼容机，并可采用这一方法，如台湾产的 PC/XT 机等。

在完成上述准备工作之后，现在可以进行加电测试，当电源开关接通后，显示器的屏幕上应立即现出大字“8088TEST”字样，之后换屏，在 PC/XT 机系统板条件下，出现如下的标准显示结果：

```
CPU ..... OK          SW1 1 2 3 4 5 6 7 8
8237 REG ..... OK          0 1 0 0 1 0 0 1
8237 CHAN0 ..... OK   SW2 1 2 3 4 5 6 7 8
8253 CHAN1 ..... OK        0 1 0 0 1 0 1 1
8259 REG ..... OK     * RS232(1)PORT TEST RESULT;
* MEMORY TEST 512KB OK   8250 REG ..... OK
* MEMORY.....80000 55 ERROR   8250 REG(STATUS) ..... OK
SI=0000                  8250 DATA TRANSFER ..... OK
                         8250 INTR ..... OK
8259 INTRR FLAG OK
8253－0 ..... OK      * RS232(2) PORT TEST RESULT;
                         8250 REG ..... OK
P10 DATA CHAN ..... OK   8250 REG ..... OK
P10 CONTROL PORT ..... OK   8250 DATA TRANSFER ..... OK
P10 STATUS ..... OK      8250 INTR TEST ..... OK
* P10 INTR TEST ..... ERROR
```

对于上面结果中打“＊”号的部分作如下说明

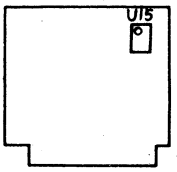
1　MEMORY TEST 512 KB…OK

　　MEMORY…8000 55 ERROR

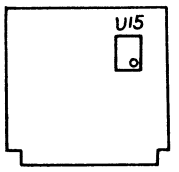
此项为系统板内存实际容量的测试结果，若被测系统板实际容量发生变化，显示结果将相应地改变，如实际容量为 256KB，则该项显示变为：

MEMOR TEST 256 KB OK

MEMOR…4000 55 ERROR



图四串卡



图五串卡

2 P10 INTR TEST ERROR

此项内容表示对 P10 的中断部分进行诊断后的结果，这部分的测试程序是针对某一并行接口编号的，故对通用的并口报错，我们准备对该部分的程序作进一步修改，满足对通用并口的测试。

3 RS232(1)PORT TEST RESULT：

要在这个测试结果区显示串卡一口的测试情况，应将串行卡的 U15 跳接器按图四所示方法接插。

4 RS232(2)RORT TEST RESULT：

要在该区显示测试串行卡 2 口的结果，只要将串行卡中 U15 的跳接器取出变换方向再插入，如图五所示。若有两个 RS232 卡同时测试，只要按上述图四、图五分别插入 U15 的位置，即可对二个串行口同时测试，其中一个测的是 RS232(1)另一个检测的是 RS232(2)，恰如标准结果所示。

需要对并行卡和串行卡做故障诊断时，应分别在并卡、串卡的插座插头位置上，插上诊断插头、插座，这样才可以使诊断程序正常工作，完成对并行卡、串行卡的测试，否则串、并卡显示的测试结果将都是 ERROR，测试是无效的。

这里给出的系统板诊断的标准结果，它的全部内容都直观地显示在一个屏幕上，如果发现某一项或几项与标准结果不同，则说明该部分发现了故障，如何针对具体的显示内容排除故障，我们下边分别给以介绍。

在对有故障的系统板进行加电检测之前，一般先要测一下系统板＋5V 与地之间的电阻值，一般应在 300Ω 左右，最低不小于 100Ω，然后交换表笔测反向电阻，其阻值不应有太大的差弃。

四、系统板上内存储器故障点的定位

为了准确地判断内存储器中的故障点，我们再仔细的分析一下内存显示信息：

MEMORY TEST 512 KB OK

MEMORY …… 80000 55 ERROR

该项信息中的 8 表示第 8 体。它是这样计算的，每 64K 的存储空间定义为一个体，现在共有 512K 的存储空间 OK，即有第 0－第 7 体共 8 个体 OK，这里出现的第 8 体已超出内存的实际容量、故报出 55 ERROR 信息，这是正常的超量错误信息。

如果待测的存储器是 512K，而测试信息显示的是：MEMORY TEST 256 KB OK

MEMORY …… 40000 ×× ERROR

按 64K 为一个体计算，有第 0－第 3 体共 4 个体 OK，第四体则表示为有故障的体位置。所以这一位数字表示的是从 0 开始计数的有故障的内存体。

因为每个体都有 9 个芯片组成，所以只知道哪个体有故障，并没有最后找到故障点。标准显示结果的“55”两位数字位置，将根据实际测试的结果给出不同的数字，表示出错体中，哪一位芯片出了故障，这些数字及其含意如表 1 所示。

表 1 内存故障定位表

显示数字	出错芯片
0 0	奇偶校验位
0 1	D0
0 2	D1
0 4	D2
0 8	D3
1 0	D4
2 0	D5
4 0	D6
8 0	D7

例如：当看到下面信息：MEMORY TEST 000KB OK

MEMORY …… 00000 04 ERROR

我们便知道是 0 体、D2 位芯片出了故障，一般情况下，为了慎重起见，把 0 体 D2 位芯片取出换到该体的其它位置上再加电测试，如果又在其它位置上给出了错误信息，则该片无疑是有故障的芯片。再比如出现：60000 00 ERROR 可以知道是第六体的奇偶校验信息片出了故障。

如果 PC/XT 机内存芯片是 256KB 的 41256 芯片，体号的计算方法不变，但坏片的物理位置应拆合为每一排芯片包括 4 个体。

当见到如下的错误信息：

MEMORY TEST 000 KB OK

MEMORY …… 00000 FF ERROR

这里报告 0 体 8 位芯片匀有故障。但一般情况，我们知道一个体中 8 个 RAM 存储器芯片突然同时损坏是很少见的，较大的可能是内存控制电路出了故障，我们可以检查行迭通($\overline{RASX}$)，列迭通($\overline{CASX}$)，等公用部分的电路。

利用诊断卡，我们在 PC 机上排除过这样一个故障：MEMORY TEST 000 KB OK

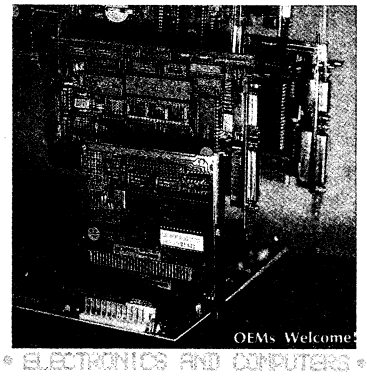
MEMORY …… 00000 00 ERROR

这里给出的信息表示 0 体的奇偶校验位坏但实际上该芯片没坏，我们从 PC 图纸上分析，引起奇偶校验错的原因还可能有两个，其一是 PCK 信号变成高电平，可能是 U96、U27、U97、U37 坏，其 2 是 I/O CHCK 信号变为高电平，可能为 U52 坏，经检查 U52－11 脚 I/O CHCK 信号“死记”，焊下 U52 芯片检查，发现 U52－13 对地短路，换 U52 芯片(LS00)，机器恢复正常。

電子與電腦 7

一九九一年

总期第76期



電子與電腦

ELECTRONICS AND COMPUTERS

目　录

·综述·

80年代微机发展回顾及90年代展望(续) …… 陈志雄(2)

·PC用户·

用未公布的dBASEⅢ命令开发图形功能 … 金林樵(4)
谈dBASEⅢ"模糊"查询 …… 班艳玲(5)
FOXBASE编程技巧两则 …… 贺文华(5)
恢复误删除文件的原理和方法 …… 朱大公(6)
微机绘制平面曲线 …… 魏新民(9)
若干子目录下共用一个dBASEⅢ(或WS)软件包 …… 龚文彪(11)
小议PC－DOS的拷贝命令 …… 刘　杰(11)
能读多位数的计算机程序 …… 周礼森(12)
模拟钢琴演奏 …… 冯剑秘(12)
如何对中文DOS的汉字库进行直接读写 …… 刘旭华(13)
NC—新型文件管理工具软件 …… 杨　进(14)

·学习机之友·

为无驱动器系统设置虚拟磁盘 …… 刘庆丰(15)
巧用LOGO解难题 …… 廖大勇(16)
修改CEC汉字系统程序 …… 周　易(16)
浅谈计算机在中专数学中的应用 …… 王志平　顾正刚(17)
数据块的后移 …… 冯端晶(18)
从QXC卡读取汉字字模 …… 牛海峰(18)
ProDOS磁盘操作系统入门 …… 廖　凯(19)
在APPLESOFT系统下实现HSCRN功能 …… 李　齐(21)
BASIC语言中几个命令功能的扩展 …… 那履弘(21)

·C语言初阶讲座·

第六讲　宏指令的用法 …… 李文兵(23)

·学用单片机·

单片机旅馆客房门卫系统 …… 傅士淦(27)

·电脑巧开发·

MSX－EPROM读写/擦除器 …… 刘　园(32)

·维修经验谈·

任天堂系列游戏机检修实例 …… 林华春(34)
让PC机接上家用彩电——CGA－PAL卡即将问世 …… 编辑部(35)
微计算机维修工具——8088系统板故障诊断卡(续)…… 董　珊(36)
磁盘驱动器修复一例 …… 雷　勇(38)

·初级程序员级水平考试辅导讲座·

汉字dBASEⅢ使用技巧(中) …… 王路敬(39)

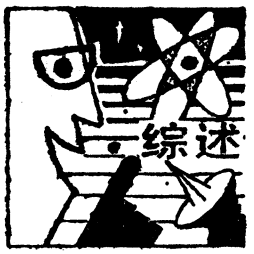
·新书与软件·

TANGO印刷线路板自动布线软件包 …… (45)
电子工业出版社新书介绍 …… (45)
电子工业出版社软件部发行CCED V3.0中文字表编辑软件 …… (46)

·读者联谊·

致参加"普及型PC个人用户软件交流联谊活动"的用户 …… 农科院计算中心(47)

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



80年代微机发展回顾及90年代展望（续）

艺高电脑国际有限公司总经理　陈志雄

工作站的总线在功能以及速度上是工作站和微机分别最大的地方。工作站的总线亦如它的微机处理器一样没有一个标准，SUN的S—BUS及VME BUS都同时存在SUN不同的系统里，有趣的是IBM最新推出的System6000是采用MCA(微通道总线)。System6000在性能上已可以跟其它的主流工作站竞争。而System6000配上MCA对它在市场上的影响则虽多作观察才能作出分析。

三、显示方式的发展

显示及图形功能是微机及工作站分别最大的地方，早期IBM的单色显示及彩色显示标准性能较差，单色图形显示方面已被美国Herculus公司开发的单色图形显示(720×350)所代替，而在彩色图形显示方面，IBM亦不停推出新的标准，从开始时的CGA(640×200)到EGA(640×400)至现时最流行的VGA(640×480,800×600,1024×768)都是在分辨率及显示颜色数量上作出改善，而图形的发生全靠以微机中央处理器来完成，所以在速度上微机图形发生比工作站的逊色得多。

反之工作站的生产厂家都有各自的图形标准，而这些图形的发生都是以独立的图形处理器来完成，为了不同的需要多数工作站的生产厂家都有其自己设计的大规模图形处理器（如SUN，DEC，IBM，Silicon Graphics等)，但新一代现成的图形处理器(如Intel的i860及Texas Instrument的TMS340系列)都有很强的性能，所以图形处理将会转向标准化器件而不用专用芯片，如HP/Apollo的VRX图形处理卡便是用上了Intel的i860，而IBM用在它的X终端上的图形处理便是Texas Instrument的TMS34010。

在90年代微机的显示及图形功能可能会向工作站的方向迈进，这是在改善微机总线后的合理结果。IBM PS/2的微通道及ESIA总线的设计是便利多个处理器在同一总线上并行运算，IBM于90年10月便首先推出在微通道上操作的XGA(Extended Graphics Array)显示及图形处理标准。新的XGA包含了一个高速的VGA，一个图形辅助处理器及总线控制功能，所以XGA除了可以往下跟VGA的软件兼容外，新开发的软件可以利用它的辅助处理器来增加图形发生的速度，而XGA的总线控制功能更能在中央处理器工作的当中，直接访问主机内存，提高了整个形功能的速度。

在ELSA总线上操作的图形处理标准估计亦在今后两三年内推出，它的功能相信会跟XGA大同小异，但最关键的还是谁能创造一个大家都公认的标准。

四、其他关键技术的发展

1. 显示器技术

以往显示器都是以12英寸及14英寸为主流，而以后的趋势将会多元化，从9英寸至19英寸以适合不同的需要，而RGB模拟彩色监示器因价格关系会一步步取代TTL数字彩色监示器。90年代最大的发展可能会是大屏幕彩色液晶显示器，这是世界关注的一个发展，因为现时彩色LCD技术仍然是掌握于日本厂家的手中，如彩色LCD的价格能够回落至合理水平的话它便极有可能取代了阴极射线管了。

2. 存储技术

光盘一直是磁盘的一种威协，但直至现在光盘技术还未有可以取代磁盘的突破，光盘现时在价格、性能、体积上都未能取胜磁盘，反之磁盘不断有新的发展，如现时的2 1/2英寸40兆字节硬盘在便携式计算机上已极为流行，市场上亦传言新的1 1/2英寸硬盘很快便出现。

在脱机后备方面，光盘还受到8mm录象带技术的威胁。现在，一个小小的8mm盒带已可以记录上千兆字节的信息，所以它正开始成为网络后备资料存储的标准。

光盘技术中，现时唯一胜出的要算是CD－ROM了，许多厂家利用CD－ROM来发送技术手册，文件和软件。在90年代，我们将可以见到CD－ROM在工作站及部份微机中成为一标准配件，使用CD－ROM来传递印刷品的年代已为期不远了。

3. 网络技术

IBM在80年代中期推出的“Token Ring”网络系统已证明不成功而现时最流行的差不多都是“以太网”(Ethernet)，但：“以太网”的传输速度只有10M bit/s，在90年代多媒介通讯开始在计算机上实现时，网络在速度上遇到的压力将会很大，尤其是计算机将不单是传送文本，而且还要传送语音，图形和图象。相信传送速度达100M bit/s的光纤网络会被推动而形成规模经济，但在此举未成之前，Ethernet还会继续维持其领导地位。

4. 微机的设计技术

半导体技术的进步不单是微处理器功能增加，对整台计算机的微型化有很大的影响。除了人所共知的存储芯片从80年代开始时的64K，256K到现时的1M及4M外，环绕着中央处理器的外围IC如DMA

(Direct Memory Access)，存储管理(Memory Management)，总线控制，中断控制等等，都得到集成化。早期的 8088PC/XT 及 80286AT 都需要上百片的集成电路组成，因为母板上基本计算机功能所需面积太大，所以许多外围功能如并/串联、显示器、软/硬盘控制器等都需要以插卡的形式来完成。

1985 年美国的 Chips and Technology 公司开始把 286AT 微机的外围芯片集成在 5 片大规模集成电路上，这方向证明很成功，很多半导体设计公司便因此而成立加入竞争行列，如 SUNTAC，VLSI，Zymos，Acer，Twin Head，Opti，Head land 等等。而一些大型公司如 Texas Instrument INTEL 和 Motorola 亦不甘人后相继加入竞争。结果是集成度不断的提高，从五芯片一组的变成最新有单芯片的 286AT 及 386SX 微机(微处器，内存 IC，EPROM 及一些零散的逻辑 IC 不包括在内)。

踏进 90 年代，有能力的半导体公司如 Intel，AMD 等更会进一步把微处理器和外围芯片都集而为一。估计 91 年中美国 AMD 公司将会推出创新的 80286 "Motherchip"，即把 80286 和它的外围电路集而为一的"母片"。可以看得到的是母板上的零件会大量减少，计算机设计及制造厂家将会把更多的外围功能放在母板上变成基本功能如并/串联接口、软盘控制器、软盘控制器、硬盘接口(IDE 接口)等，这对母板上扩充槽数量上的要求便减少了，因此计算机的外形可设计得更加小和更加美观。

五、市场的发展

根据美国市场研究公司 Datquest 的资料显示，1990 年全世界微机的总销量为 2500 万台，估计在 1994 年全世界微机的总销量会增加到 4000 万台总值 630 亿美元即每年有 12%左右的增长。

在市场占有率方面，除了超级计算机之外，大型及小型计算机的市场占有率将会一步步减少，估计它们的增长只有 6－7% 左右。所以到 1994 年工作站加上微机在市场的总值将会跟小型、大型和超级计算机的总值扯平。相信在 1995 年以后，以微处理器为基础的计算机在市场总值上将会越过小型、大型和超级计算机。

在微机本身的发展方向，便携式的增长将会比坐台式的大。在 1989 年坐台式销售量为 2000 万台左右占微机市场的 90%，估计 1994 年坐台式微机的销售量为 2500 万台占整个微机市场的 63%，而便携式微机更会发展得品种繁多，大致可分为：

1. 膝上形(重量在 5 公斤左右)——在 1994 年市场占有率大概为所有微机的 8%左右，总机量 310 万台。

2. 记事簿形(重量在 2 公斤左右)——在 1994 年市场占有率预测为所有微机的 16%，总机量 640 万台左右。

3. 掌上形(重量在 1 公斤之下)——在 1994 年市场占有率预测为所有微机的 13%，总机量 520 万台左右。

在微处理器方面，8088/8086 一向被人认为会被淘汰，但因为掌上形微机的掘起，在 92 年开始会重纳增长轨道，而 80286 微机将会在 90 年代步入负增长，数量最大而增长亦最高的。据美国 Dataquest 的估计是 80386SX 微机，而 386DX 和 80486 虽然有稳定的增长，但在数量上会远远不及 386SX。

以上预测是 Dataquest 公司前一段时间作出的，但他可能未能及时把 AMD 生产的 80386DX 和 Intel 公司宣布的 80486SX 可能带来的影响估计在内。相信 AMD 在 91 年或 92 年会开始把 80386DX 降价，而 Intel 的对策是推出 80486SX(注：80486SX 是 80486DX 减去了辅助处理器)。在两者竞争的情况下 386SX 能否继续处于领先地位上要看以后的发展，但得益的最后都是计算机的用家。

经过十年的市场及技术发展，可以预见的是在 90 年代微机将会更普及化，而其技术发展亦趋向更微型化及多元化。

在一个统一标准的局限下微机在技术上的发展在 80 年代是追不上工作站，但在 Intel80386 32 位微处理器推出之后估计标准微处理器已定型，只要半导体厂家能生产兼容 80386 的话，以后的发展可能不会被统一标准所局限。在微机总线有所改良而其他辅助器件在不停的改进下，微机开始有条件进军工作站市场。但工作站的领导公司亦不停地努力将自己的产品变成一个标准，但要推进这方面的工作可能比发展技术来得更困难。

无论如何我们在 90 年代将会看到的微机的发展不会比 80 年代的逊色，差不多可以断言 90 年代微机的功能价格比仍然有一到两个数量级的改善。

会 议 消 息

中国计算机用户协会 IBM 微机协会定于今年 10 月 21 日至 25 日在成都望江宾馆召开协会第四届年会(全国微机应用技术交流会。会议交流内容为：

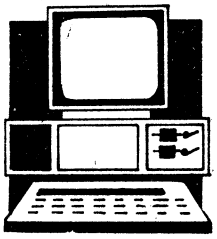
1. 计算机网络与通讯
2. 管理信息系统
3. 办公自动化系统
4. 电子、机械 CAD
5. 机电一体化产品
6. 计算机文字处理系统

会间作有关技术报告并评选优秀论文，欢迎协会理事单位、会员单位、微机用户以及从事微机研究开发、生产销售、技术服务的单位派人参加会议。

凡确定参加会议者请务于 8 月底前同 IBM 微机用户协会办公室联系。

通讯地址：北京学院南路 55 号 1314 室

电　　话：8316539



PC用户

用未公布的dBASE Ⅲ命令开发图形功能

金林樵

人们普遍认为dBASE Ⅲ本身没有图形功能，因此总是求助于PASCAL等有强大图形功能的软件，从dBASE Ⅲ数据库中间接或直接读得绘图数据后，将其显示出来。其实，我们可以借助dBASE Ⅲ的未公布命令来开发它的图形处理能力。

一、未公布的命令及函数

在使用dBASE Ⅲ时，我们发现了若干条在dBASE Ⅲ使用手册、参考书等资料中未见公布的命令和函数。借助于它们可在dBASE Ⅲ系统下直接绘出所需的图形。这些命令和函数为：

1. SET SEGMENT TO〈段地址〉

它用来定义其后的POKE，PEEK函数中内存地址所使用到的段地址。

2. POKE〈内存地址〉,〈数据〉

它用来将数据写入所指定的内存地址单元中。

3. PEEK（〈内存地址〉）

它将指定的内存地址单元的内容读出。

后两个函数中内存地址的段地址，由其前的最后一个SET SEGMENT命令中所指定的段地址决定。命令中的段地址，内存地址和数据均为十进制数，且不能是变量或表达式。

二、屏幕绘图的基本原理

在PC/XT的彩色图形显示器下，分配给屏幕的缓冲区为16K内存。它位于段地址47104(&HB800)中，偏移地址为0到16383。屏幕的分辨率为640×200。缓冲区中每个字节的每一位对应屏幕上的一个图形点。因此，每行640个点占用80个字节。屏幕显示分偶数和奇数行。偶数行对应的显示缓冲区地址为0～7999；奇数行分配地址为8192～16191。因此，在绘制图形时，应注意以下的换算关系：

将显示行除2后，结果为整数(设值为P)时，则其内存地址为：P＊80＋INT(列数/8)；若不为整数，则说明为奇数行，这时将该数取整(设值为P1)后，其对应地址为：8192＋P1＊80＋INT(列数/8)。例如POKE 80,255将在(2,0)至(2,7)间画一水平线。

三、图形功能的实现

根据下列数据库结构，绘出直方图。数据库JIN.DBF的结构如下：

字段	字段名	类型	宽度	小数位
1	班级	C	4	
2	平均成绩	N	4	

数据库中数据如下：

记录号	班级	平均成绩
1	8911	80
2	8912	75
3	8921	89
4	8922	71
5	8931	85

下附的程序就可根据上述数据库绘出直方图。本例在IBM PC/XT上调试通过，并可用屏幕硬拷贝打印出图形。

```
SET TALK OFF
CLEAR
SET SEGMENT TO 47104
Y=140
X=20
N=0
I=0
Y1=100
@ 0,24 SAY "浙江电子工业学校数学成绩示意图"
DO WHILE N<Y1
  AD=(INT(Y/2)-I-1)*80+X
  IF INT(N/2)=N/2
    STOR "POKE"+STR(AD+8192,6)+",3" TO PO
  ELSE
    STOR "POKE"+STR(AD,6)+",3" TO PO
    I=I+1
  ENDIF
  &PO
  N=N+1
ENDDO
I=0
X1=40
AD=INT(Y/2)*80+X
DO WHILE I< X1
  STOR "POKE"+STR(AD+I)+",255" TO PO
  &PO
  I=I+1
ENDDO
@ (Y-Y1)/16-1,X-3 SAY "平均成绩"
@ Y/16-1,X+X1+2 SAY "班级"
USE JIN
L=LEN(班级)+1
DO WHILE .NOT. EOF()
    STOR 平均成绩 TO J
    STOR 班级 TO BJ
    N=1
    I=0
    DO WHILE N< J
    AD=(INT(Y/2)-I)*80+X+L
    IF INT(N/2)=N/2
    STOR "POKE"+STR(AD+8192,6)+",255" TO PO
    ELSE
    STOR "POKE"+STR(AD,6)+",255" TO PO
```

```
    I=I+1
    ENDIF
    &PO
    N=N+1
    ENDDO
    X=X+L
    @ Y/16,X-1 SAY BJ
    SKIP
ENDDO
USE
RETURN
```

浅谈 dBASE Ⅲ“模糊”查询

黑龙江友谊县红兴农隆技校　班艳玲

关于汉字 dBASE Ⅲ 的模糊查询方法很多，但这些方法都不能满足用户的全部要求。如：有的用户要求把查到的信息从打印机上输出，如果用屏幕硬考贝，则效果不好。为了解决这类问题，本人在编程中发现用 SET FILTER TO〈条件〉这条命令，可以满足用户的打印要求，程序如下：

```
set print off
set talk off
clear
use〈数据库〉
go top
acce “请输入要查人的姓名(只需一字)” to h2
set filter to '&h2.' $ 姓名
copy to bh.dbf
*(注：可在此处插入屏幕显示)
wait “您需打印吗(y/n)?” to h3
if h3=“y” .or. h3=“Y”
clear
wait “按任一键开始打印…”
set device to print
set print on
use bh
go top
? line1
? line2
do while .not.eof()
? line3
? jl
skip
enddo
? line5
?
endif
set print off
retu
```

注：此种方法在 IBM PC/XT 机上通过，其中变量 line1，line2，line3，line4，line5 为表格变量，变量 jl 为记录信息。

FOXBASE 编程技巧两则

湖南娄底师专　贺文华

一、FOXBASE 中 ON 命令的巧用

FOXBASE 中的 ON 命令在指定的条件出现后将引起程序执行的转移。这种条件可能是在程序的执行过程中按了 ESC 键或其它键而引起的。为避免这种程序执行的转移引起对打开的数据库文件及其索引文件造成破坏致使数据丢失，我们可巧用 ON 命令，编制一个有意义的中断处理子程序：

```
SET BELL ON
SET CONSOLE OFF
?? CHR(7)
@ 8,20
@ 8,20 SAY '警告：'
@ 8,30 SAY '系统运行时不得强行中断!'
I=INKEY(5)
@ 8,20
SET CONSOLE ON
RETURN
```

调用这一子程序，只需在主程序的开头设置一个

```
ON ESCAPE DO ESC KEY
```

命令即可。当用户企图通过 ESC 键中断程序运行时，此程序将发出警告，稍停后继续程序的运行。

二、打印机未准备就绪的预处理

在数据库管理系统中，由于打印机未准备就绪而导致程序出错是一个常见的问题，有趣的是我们可以利用 FOXBASE 中提供的 SYS 函数来预防。

FOXBASE 中函数 SYS 可返回当前打印机的状态。在 MS-DOS 下，如果打印机没有联机或接通电源，将返回信息“OFFLINE”；如果打印机在线，则返回信息“READY”。据此，编制如下程序段来预防由于打印机故障而导致程序运行出错的情况。

```
* * * * 打印机未准备好处理子程序 * * * *
DO WHILE .T.
PASS= 'P'
SET CONSOLE OFF
IF SYS(13)=“OFFLINE”
SET BELL ON
?? CHR(7)
DO WHILE UPPER(PASS)<>'Y' AND UPPER(PASS)<>'N'
? “打印机未准备好，重试吗(y/n)：”
ACCEPT TO PASS
ENDDO
ENDIF
DO CASE
CASE UPPER(PASS)= 'N'
SET CONSOLE ON
```

```
EXIT                                    EXIT
CASE UPPER(PASS)='P'                    ENDCASE
SET CONSOLE ON                          ENDDO
DO 打印程序                             RETURN
```

恢复误删除文件的原理和方法

朱大公

磁盘文件被误删除是令人沮丧的事。本文提供的源程序，经汇编和链接生成可执行文件后，用户只需键入文件名，根据屏幕提示进行简单操作，便可恢复被误删除的文件。

为使读者对程序有较好的理解，本文拟从文件目录表(FDT)和文件分配表(FAT)谈起。

DOS 对文件的管理功能，主要是通过 FDT、FAT、磁盘参数表和设备驱动程序实现的，而与恢复被误删除文件有直接关系的是 FDT 和 FAT。

对于容量 360K 字节的软磁盘，DOS 把从相对扇区 5 开始的 7 个连续扇区分配给 FDT。FDT 包含 112 个目录项，因此根目录下最多可存放 112 个文件。一个目录项占 32 字节，分成下面 8 部分(表 1)。

位移	域	长度(字节)
00H－07H	文件名	8
08H－0AH	扩展名	3
0BH	文件属性	1
0CH－15H	保留	10
16H－17H	文件建立或最后修改日期	2
18H－19H	文件建立或最后修改时间	2
1AH－1BH	文件首簇号	2
1CH－1FH	文件长度(字节数)	4

表 1　目录项结构(位移相对于目录项首字节)

图 1 是某软盘上 FDT 的一部分，现结合表 1 和图 1 说明目录项各部分的含义。

```
3F70:0000  49 42 4D 42 49 4F 20 20-43 4F 4D 27 00 00 00 00    I B M B I O     C O M `....
3F70:0010  00 00 00 00 00 00 00 00-60 54 07 02 00 80 12 00 00  ........`T......
3F70:0020  E5 45 42 55 47 20 20 20-43 4F 4D 20 00 00 00 00    e E B U G       C O M  ....
3F70:0030  00 00 00 00 00 00 00 00-60 68 06 2A 00 80 2E 00 00  ........`h.*.....
3F70:0040  43 4F 4D 4D 41 4E 44 20-43 4F 4D 20 00 00 00 00    C O M M A N D   C O M  ....
3F70:0050  00 00 00 00 00 00 00 00-60 54 07 24 00 80 45 00 00  ........`T.$..E..
3F70:0060  00 F6 F6 F6 F6 F6 F6 F6-F6 F6 F6 F6 F6 F6 F6 F6    .vvvvvvvvvvvvvvv
3F70:0070  F6 F6 F6 F6 F6 F6 F6 F6-F6 F6 F6 F6 F6 F6 F6 F6    vvvvvvvvvvvvvvvv
```

图 1 文件目录表 FDT(部分)

文件名域　最多可存放 8 个字符的文件名，不足 8 个字符则以空格码 20H 填充。图 1 中 3F70:00 00 处开始的 8 个字节便存放文件名 IBMBIO，后跟 2 个空格。文件名域的首字节有可能出现下列情况：

00H　表示该目录项未使用过，其后的各目录项均不再被查找。空目录项的其余各字节一般填入 F6，如图 1 的 3F70:0061～007F。

E5H　文件删除标志。该目录项可在建立新文件时被使用。图 1 中 3F70:0020 处的 E5H 表示 eEBUG.COM 文件已被删除。

扩展名域　最多可放 3 个字符的扩展名。不足 3 个字符时以空格码 20H 填充。

文件属性域、DOS 保留区、日期域和时间域，它们与恢复误删除文件关系不大。

文件首簇号域　填入分配给文件的首簇号(按低、高字节顺序)。图 1 中 eEBUG.COM 的首簇号为 02AH。

文件长度域　填入文件长度字节数(按低、高顺序)，图 1 中 eEBUG.COM 的长度为 2E80 字节。

DOS 把 FAT 安排在相对扇区号 1 开始的两个连续扇区，还在随后的两个连续扇区安排一份副本，供 FAT 损坏时使用。分配给文件的磁盘空间以簇为单位，每一簇为连续的两个扇区。根据簇号可找到对应的扇区。文件的簇号信息都包含在 FAT 中。

软盘的 FAT 约有 680 个登记项，每项占 1.5 字节，即 12 位，并从第一项开始按 000H 簇、001H 簇、002H 簇等顺序编号。000H 簇的首字节为磁盘介质标志，对于软磁盘是 FDH，其余两字节保留为 FFH。因此可分配给文件的第一个可用簇为 002H 簇。磁盘上尚未存放文件时，除 000H 簇外，其余各字节均填入 00H。使用 FAT 的困难在于每个登记项的值并非实际簇号。DOS 按下列算法处理以获取实际簇号：

1 刚用过的簇号乘以 1.5；

2 取乘积的整数部分作为 FAT 中的位移；

3 从 FAT 表中指定位移处取出一个字；

4 若刚用过的簇号为偶数，取该字的低 12 位；若

刚用过的簇号为奇数，则取高 12 位。

5 上述结果若为 000H，表明该簇为空，便是可分配给文件的下一簇号；若结果为 FF7H，表示该簇已损坏不可用；若结果为 FFFH，表示刚用过的簇号是文件的最后一簇。其它情况则表明该簇已经分配。

←i 簇→			←i+1 簇→		
2	5	6	0	0	2

图 2 文件分配表 FAT 中的簇号

参看图 2，设第 i－1 簇为偶数，实际取出的字为 6025H，簇号取低 12 位，为 025H 簇。下一簇号应为 026H（取出的字为 0260H，取高 12 位）。千万不能认为第 i 簇的簇号为 256H，i＋1 簇簇号为 002H。

DOS 建立新文件时先扫描 FAT，跳过已分配的簇或坏簇，找出第一个空簇（其值为 000H），将该簇号填入 FDT 中相应目录项的文件首簇号域，再继续查找下一个空簇，将其簇号填入第一个可用簇的位置，依此类推，直到分配给文件的簇数满足文件长度的要求，并在最后一个可用簇的位置填入 FFFH。这样在 FAT 中便形成一条簇号链，无论文件在磁盘上的物理地址如何分散，均可正常读写。图 3 是 FAT 表的一部分：

```
3F70:0000   FD FF FF 03 40 00 05 60-00 FF 8F 00 09 A0 00 0B
3F70:0010   C0 00 0B E0 00 0F 00 01-11 20 01 13 40 01 15 60
3F70:0020   01 17 F0 FF 19 A0 01 1B-C0 01 1D E0 01 1F 00 02
3F70:0030   21 20 02 23 F0 FF 25 60-02 27 80 02 29 B0 03 00
3F70:0040   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
3F70:0050   00 37 80 03 39 A0 03 FF-CF 03 3D E0 03 3F 00 04
3F70:0060   41 20 04 43 40 04 45 60-04 FF 8F 04 49 A0 04 FF
3F70:0070   CF 04 FF EF 04 4F 00 05-51 F0 FF 00 00 00 00 00
```

图 3 文件分配表 FAT（部分）

现在，让我们借助图 1 和图 3 来查找文件 COMMAND. COM 的簇号链。

1 由图 1 可知，文件的首簇号为 024H；

2 024H×1. 5＝036H；以 036H 为偏移量（位移），在 FAT 中取出一个字，为 6025H；

4 由于 024H 为偶数，所以取 6025H 的低 12 位，为 025H。这就是分配给文件的下一簇号；

5 025H×1. 5，取整得 037H 作为偏移量，在 FAT 表中取出一个字，为 0260H；

6 由于 025H 为奇数，所以取 0260H 的高 12 位，为 026H，它是分配文件的又一个簇号；

7 重复 2～7 步，当取出的簇号为 FFFH 时，表示文件结束。

由计算得知，文件 COMMAND. COM 的簇号链为 024H、025H、026H、027H、028H、029H、03BH、03CH、03DH、03EH、03FH、040H、041H、042H、043H、044H、045H、和 046H 共 18 个簇。注意 029H 簇以后跳至 03BH 簇，表明文件在磁盘上的存放是不连续的。

删除文件时，DOS 先在 FDT 中找到要删除的文件名，将该目录项的首字节改为 E5H，并根据文件首簇号域提供的首簇号，在 FAT 中查找文件的簇号链，将簇号链的每一登记项填入 000H，而这时该文件的内容仍完整无损地保留在磁盘上，只是不能读出罢了。

将本文提供的恢复误删除文件的源程序通过行编辑程序 EDLIN. COM 输入计算机并存盘后，再经过汇编和链接便可生成. EXE 的可执行文件。设文件名为 HFWJ. EXE，使用时在 DOS 下键入 HFWJ，然后按程序提示将含误删除文件名的软盘插入 B 驱动器，键入待恢复的文件名（大小写均可）并回车。如果恢复文件成功则返回 DOS；若读写出错或找不到待恢复的文件名，程序都有相应的提示和处理。

根据 DOS 的文件管理机制，文件被删除后，一旦在磁盘上建立了新文件，或某些文件长度增加，则不仅簇号链的恢复可能出错，而且误删除文件在磁盘上的信息也将被新文件或其它信息所取代。因此要想使恢复误删除文件获得成功，应在建立新文件或增加某些文件长度以前进行。另外，当磁盘上有多个被删除文件时，恢复误删除文件的成功率也不大。

程序流程如下：

1 将 DOS 返回地址入栈、设置 DS 和 ES 段、提示在 B 驱动器插入软磁盘，等待键盘输入。

2 读 FDT，若反复读至规定次数后仍不成功则提示读 B 驱动器出错，返回 DOS。

3 提示输入误删除文件名，转换成 FDT 目录项的标准格式后存入文件名缓冲区。将文件名中的小写字符换成大写。

4 目录项超过 112 或其首字节为 00H 时，提示无此文件并询问是否重输入误删除文件名。若键入 Y 或 y 则返回 3，否则返回 DOS。

5 若目录项首字节不是 E5H，修改目录项指针后返回 4。

6 若误删文件名与目录文件名不同，指针指向下一目录项后返回 4。

7 修改恢复目录项文件名的首字节。

8 将 FDT 写回软磁盘，若反复写至规定次数仍不成功，则提示写 B 驱动器出错后返回 DOS。

9 计算误删除文件所需簇数并读入 FAT。

10 取刚用过的簇号乘以 1. 5，保留该偏移地址。

11 簇号增一得新簇号并乘以 1. 5，得新偏移地址。

12 新簇号为奇数则按新偏移地址取一个字的高12位；新簇号为偶数则取低 12 位。若所得之值为000H，则返回 11。

13 簇数减 1。若未减至零，根据刚用过簇号的奇偶性，在指定的偏移地址填入新簇号后返回 10。

14 所需簇数已减至零。根据刚用过的簇号的奇偶性，在指定的偏移地址填入 FFFH，将 FAT 写回 B 驱动器后返回 DOS。

```
SSG SEGMENT PARA     STACK 'STACK'
        DB      32 DUP('S')
SSG     ENDS
CSS     SEGMENT
        ASSUME DS:CSS,ES:CSS,CS:CSS
BEGIN   PROC    FAR
        PUSH    DS
        XOR     AX,AX
        PUSH    AX
        MOV     AX,CSS
        MOV     DS,AX
        PUSH    DS
        POP     ES
        LEA     DX,TISICP
        CALL    TISI
        MOV     AH,08
        INT     21H
        MOV     CX,05
DUB:    PUSH    CX
        CALL    DXCS
        INT     25H
        POP     CX
        POP     CX
        JNC     TSWJM
        LOOP    DUB
CUO:    LEA     DX,DUBCUO
        CALL    TISI
        JMP     JIESU
TSWJM:  LEA     DX,TISIM
        CALL    TISI
        LEA     DX,WJM0
        MOV     AH,0AH
        INT     21H
        MOV     CX,08
        MOV     BX,00
        LEA     SI,WJM2
        LEA     DI,WJM
QUZIFU: MOV     AL,[SI]
        CMP     AL,2EH
        JZ      CNSP1
        CMP     AL,0DH
        JZ      CNSP2
        MOVSB
        INC     BX
        JMP     QUZIFU
CNSP1:  INC     SI
        SUB     CX,BX
        MOV     AL,20H
        REPZ    STOSB
        MOV     CX,03
        MOV     BX,00
        JMP     QUZIFU
CNSP2:  SUB     CX,BX
        MOV     AL,20H
        REPZ    STOSB
        MOV     CX,11
        LEA     SI,WJM
        MOV     DI,SI
QUZF:   LODSB
        CMP     AL,61H
        JB      CNZF
        CMP     AL,7AH
        JG      CNZF
        SUB     AL,20H
CNZF:   STOSB
        LOOP    QUZF
        LEA     SI,ML-FAT
        LEA     BX,WJM0
        MOV     ML-112,BX
PAN112: CMP     SI,ML-112
        JB      PAN00
        JMP     WUWJ
PAN00:  MOV     AL,[SI]
        CMP     AL,00
        JNZ     E5
        JMP     WUWJ
E5:     MOV     AL,[SI]
        CMP     AL,229
        JZ      ZSI
ADDSI:  ADD     SI,32
        JMP     PAN112
ZSI:    MOV     WJMDZ,SI
        INC     SI
        LEA     DI,WJM
        INC     DI
        MOV     CX,10
        REPZ    CMPSB
        JZ      HFWJM
        MOV     SI,WJMDZ
        LEA     DI,WJM
        JMP     ADDSI
HFWJM:  MOV     SI,WJMDZ
        LEA     DI,WJM
        MOV     AL,[DI]
        MOV     [SI],AL
        MOV     CX,05
XIEB:   PUSH    CX
        CALL    DXCS
        INT     26H
        POP     CX
        POP     CX
        JNC     LIN
        LOOP    XIEB
        LEA     DX,XBCUO
        CALL    TISI
        JMP     JIESU
LIN:    JMP     QIUCUS
WUWJ:   LEA     DX,TISIW
        CALL    TISI
        MOV     AH,01
        INT     21H
        CMP     AL,59H
        JZ      JMP0
        CMP     AL,79H
        JNZ     JMP1
JMP0:   JMP     TSWJM
JMP1:   JMP     JIESU
QIUCUS: MOV     SI,WJMDZ
        ADD     SI,28
        MOV     AX,[SI]
        INC     SI
        INC     SI
        MOV     DX,[SI]
        MOV     BX,1024
        DIV     BX
        CMP     DX,00
        JZ      CNCUS
        INC     AX
CNCUS:  MOV     CUS,AX
        SUB     SI,04
        MOV     AX,[SI]
        PUH     AX
        LEA     BX,DXCS
        MOV     BYTE PTR [BX+07],02
        MOV     BYTE PTR [BX+10],01
        MOV     CX,05
ZDUB:   PUSH    CX
        CALL    DXCS
        INT     25H
        POP     CX
        POP     CX
        JNC     CUZAN
        LOOP    ZDUB
        JMP     CUO
CUZAN:  POP     AX
CUNCU:  MOV     QCUHAO,AX
        CALL    PYDZ
        MOV     FATDZ,BX
        MOV     AX,QCUHAO
        MOV     XCUHAO,AX
ZENCU:  INC     XCUHAO
        MOV     AX,XCUHAO
        CALL    PYDZ
        TEST    XCUHAO,01
        JNZ     J1
        ADD     BX,OFFSET ML-FAT
        MOV     AX,[BX]
        AND     AX,0FFFH
        CMP     AX,00
        JNZ     ZENCU
        DEC     CUS
        CMP     CUS,00
        JZ      XIEFFF
        CALL    TIANCH
        JMP     CUNCU
XIEFFF: CALL    TIANJS
        JMP     XIEFAT
J1:     ADD     BX,OFFSET ML-FAT
        MOV     AX,[BX]
        AND     AX,0FFF0H
        CMP     AX,00
        JNZ     ZENCU
        DEC     CUS
        CMP     CUS,00
        JZ      XIEFFF
        CALL    TIANCH
        JMP     CUNCU
XIEFAT: MOV     CX,05
ZXFAT:  PUSH    CX
        CALL    DXCS
        INT     26H
        POP     CX
        POP     CX
        JNC     JIESU
        LOOP    ZXFAT
        LEA     DX,XBCUO
```

```
        CALL    TISI
JIESU:  RET
BEGIN   ENDP
TISI    PROC
        MOV     AH,09
        INT     21H
        RET
TISI    ENDP
DXCS    PROC
        MOV     AL,01
        LEA     BX,ML-FAT
        MOV     CX,0007H
        MOV     DX,0005H
        RET
DXCS    ENDP
PYDZ    PROC
        MOV     BX,AX
        SHR     AX,01
        ADD     BX,AX
        RET
PYDZ    ENDP
TIANCH  PROC
        MOV     BX,FATDZ
        ADD     BX,OFFSET ML-FAT
        MOV     AX,[BX]
        TEST    QCUHAO,01
        JNZ     JI-SHL
        OR      AX,XCUHAO
        MOV     [BX],AX
        AND     AX,OFFFH
        JMP     RE-TCH
JI-SHL: MOV     CL,04
        SHL     XCUHAO,CL
        OR      AX,XCUHAO
        MOV     [BX],AX
        SHR     AX,CL
RE-TCH: RET
TIANCH  ENDP
TIANJS  PROC
        MOV     BX,FATDZ
        ADD     BX,OFFSET ML-FAT
        MOV     AX,[BX]
        TEST    QCUHAO,01
        JNZ     JIFFF
        OR      AX,0FFFH
        JMP     FFF
JIFFF:  OR      AX,0FFF0H
FFF:    MOV     [BX],AX
        RET
TIANJS  ENDP
ML-FAT  DB      1792 DUP('ML')
WJM0    DB      13
WJM1    DB      00
WJM2    DB      13 DUP(00)
WJM     DB      11 DUP(00)
WJMDZ   DW      00
ML-112  DW      00
CUS     DW      00
FATDZ   DW      00
QCUHAO  DW      00
XCUHAO  DW      00
DUBCUO  DB      0AH,0DH,'读 B:驱错! $'
TISIM   DB      0AH,0DH,'请输入
                误删除的文件名:- $'
TISIW   DB      0AH,0DH,'无此文件!'
        DB      0AH,0DH,'要重新输入
                文件名吗? (Y/N)- $'
TISICP  DB      0AH,0DH,'在 B:驱插入
                含误删除文件的磁盘!'
        DB      0AH,0DH,
                '按任意键继续… $'
XBCUO   DB      0AH,0DH'写 B:驱错! $'
CSS     ENDS
        END     BEGIN
```

微机绘制平面曲线

长春财政学校微机室　　魏新民

用微机绘制平面曲线时，往往要花很多时间设计程序。笔者在此给出一个用 BASIC 语言设计的的通用程序，可在配有中分辨(640×200)彩色显示器的 IBM-PC、XT、286 及兼容机上使用，现将程序的功能及使用方法说明如下。

此程序的通用性很强，主要体现在：1、对于任一方程，只要能用 BASIC 中的自定义函数来表示，都能画出图象；2、此程序适用于直角坐标方程 y＝F(X)、极坐标方程 ρ＝ρ(θ)及参数方程 X＝φ(t)，y＝ψ(t)三种类型的曲线；3、能在同一坐标系中同时画出多条(不超过 30)曲线，而且对于每一个方程，变量的取值区间可以任选。

用此程序画出的曲线准确直观，在坐标轴上都标有刻度，相邻两刻度的距离 I 由方程中变量的取值区间决定，比如，对于方程 $y=x^2$，当 x∈[－1,1]时，I＝0.1，当 x∈[－5,5]时，I＝1，当 x∈[-9,9]时，I＝5，图象画出后在屏幕的左上角显示出 I 的值。

此程序使用方便，只要在运行前，先根据曲线的方程依次在程序中的 101～130 程序行之间写入 m(m 为方程个数)个自定义函数语句与 GOTO 语句(行号必需连续)，运行后在屏幕的提示下输入“描点数(隐含值为 60)”、“图象个数(隐含值为 1)”、“变量取值区间”等信息即可，具体方法如下：

1、直角坐标方程，比如 y＝sinx，y＝xsinx 与 y＝x＋sinx，需写入如下三行：

```
101 DEF FNY(X)=SIN(X):GOTO 200
102 DEF FNY(X)=X*SIN(X):GOTO 200
103 DEF FNY (X)=X+SIN(X):GOTO 200
```

图 1 是画出的三条曲线(描点数 60，区间[-8,8])

2、极坐标方程，如 ρ＝θ 与 ρ＝4SIN4θ，需写入如下两个程序行：

```
101 DEF FNY(T)=T:GOTO 200
102 DEF FNY(T)=4*SIN(4*T):GOTO 200
```

对于直角坐标与极坐标方程，函数名必需是 FNY。图 2 为描点数 100，区间[0,8]和[0,6.284]的两条曲线。

3、参数方程，比如 $\begin{cases}x=t^2-2\\y=t\end{cases}$，需写入如下程序行：

```
101 DEF FNX(T)=T*T-2:DEF FNY(T)=T:
GOTO 200
```

函数名必需是 FNX 与 FNY。

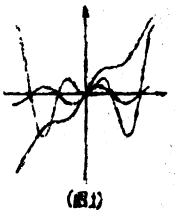
如方程中含有可变参量而且要画出当参量取不同值时方程所对应的曲线，或者要画出含有间断点的函数在其连续区间上的图象时，需要在 190 行中写一个自定义函数语句，对于前者，还需在 190 行中加一个 INPUT 语句。比如，对于含两个可变参量 A、B 的极坐标方程 ρ＝Acosθ＋B，应将 190 行写成

```
190 DEF FNY(T)=A*COS(T)+B:INPUT A,B
```

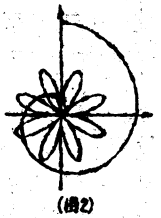
最后要说明的是，图象画出后，反复敲 A 可使屏幕在中分辨率与高分辨率之间转换，敲 C 可改变颜色，敲 S 可将图象缩小，敲 R 重新运行，敲 Q 退出，若敲 L 键，则将图象纵向伸长，这时如进行屏幕拷贝，在打印机上得到的是标准图象。

```
5 KEY OFF:SCREEN 2:CLS
10 LOCATE 4,20:INPUT"描点数(30--300)";N%:IF N%<30 OR N%>300 THEN N%=60
15 LOCATE ,20:INPUT "图象个数";M%:IF M%<1 THEN M%=1
20 LOCATE ,20:INPUT "方程类型(1--直角坐标;2--参数方程;3--极坐标)";LX:LX=INT(LX):IF LX<1 OR LX>3 THEN LX=1
25 LOCATE ,20:INPUT "如方程中含可变参数或间断点请键入 y";FT$
30 DIM X(M%,N%),Y(M%,N%),U(M%,2),X%(M%,N%),Y%(M%,N%),O(2)
40 I%=0:II=0:CC=0:FF=0:IF M%=1 THEN 60
50 LOCATE ,20:INPUT "在同一区间画图吗 y/n";A$:IF A$<>"y"AND A$<>"Y" THEN 70
60 LOCATE ,20:INPUT "变量取值区间[u,v]";U,V:H=V-U:GOTO 70
70 I%=I%+1 :IF FT$="Y" OR FT$="y"THEN 180
80 ON I% GOTO 101,102,103,104,105,106,107,108,109,110,111,112,113,114,115,116,117,118,119,120,121,122,123,124,125,126,127,128,129,130
180 LOCATE ,20:PRINT "第";I%;"组参量    ";
200 IF M%=1 OR A$="y" OR A$="Y" THEN 280
210 LOCATE ,20: PRINT"区间    ";I%;: INPUT U(I%,1),U(I%,2)
220 IF LX<>1 THEN 280
230 IF I%=1 THEN U1=U(1,1):V1=U(1,2)
240 FOR J%=1 TO 2:IF U(I%,J%)<U1 THEN U1=U(I%,J%)
250 IF U(I%,J%)>V1 THEN V1=U(I%,J%)
260 NEXT J%
280 IF A$="Y" OR A$="y" OR M%=1 THEN 290 ELSE U=U(I%,1):V=U(I%,2):H=V-U
290 ON LX GOTO 300,310,320
300 FOR J%=0 TO N%:X(I%,J%)=J%*H/N%+U:Y(I%,J%)=FNY(X(I%,J%)):NEXT:GOTO 330
310 FOR J%=0 TO N%:T=J%*H/N%+U:X(I%,J%)=FNX(T):Y(I%,J%)=FNY(T):NEXT:GOTO 330
320 FOR J%=0 TO N%:T=J%*H/N%+U:L=FNY(T):X(I%,J%)=L*COS(T):Y(I%,J%)=L*SIN(T):NEXT
330 IF I%=1 THEN P=Y(I%,0):Q=Y(I%,N%):IF LX<>1 THEN U1=X(I%,0):V1=X(I%,N%)
340 FOR J%=0 TO N%: IF Y(I%,J%)<P THN P=Y(I%,J%)
350 IF Y(I%,J%)>Q THEN Q=Y(I%,J%)
360 IF LX=1 THEN 390
370 IF X(I%,J%)<U1 THEN U1=X(I%,J%)
380 IF X(I%,J%)>V1 THEN V1=X(I%,J%)
390 NEXT
400 IF I%<M% THEN 70
410 IF A$="N" OR A$="n" OR LX<>1 THEN U=U1:V=V1:H=V-U
420 IF U>=0 AND V>0 THEN A$=60:UA%=0:GOTO 440
430 IF U<0 AND V<=0 THEN A%=600:VA%=638:ELSE A%=-U/H*540+60
440 IF -U/(A%-20)>V/(618-A%) THEN K=(A%-20)/(-U) ELSE K=(618-A%)/V
450 HL%=638
460 IF P<0 AND Q<=0 THEN B%=20:QA%=0:GOTO 480
470 IF P>=0 AND Q>0 THEN B%=160:PA%=180:ELSE B%=Q/(Q-P)*140+20
480 IF B%-5*K*P/12<=180 AND B%-5*K*Q/12>=10 THEN KX=K:KY=5*KX/12:GOTO 510
490 IF -P/(180-B%)>Q/(B%-10) THEN K=(180-B%)/(-P) ELSE K=(B%-10)/Q
500 KY=K:KX=12*KY/5
510 A1=A%:UA1=UA%:VA1=VA%:HL1=HL%:KX1=KX:KY1=KY:KX2=KX:KY2=KY
520 CLS: IF U<0 AND V>0 THEN UA%=A%+KX*U-16:VA%=A%+KX*V+16:GOTO 540
530 IF U<0 THEN UA%=A%+KX*U-16:ELSE VA%=A%+KX*V+16
540 H1%=(HL%-VA%+UA%)/2: UB%=UA%:UA%=H1%:A%=A%-UB%+H1%: VA%=VA%-UB%+H1%
550 IF P<0 AND Q>0 THEN PA%=B%-KY*P+8:QA%=B%-KY*Q-8:GOTO 570
560 IF P<0 THEN PA%=B%-KY*P+8 ELSE QA%=B%-KY*Q-8
570 H1%=(190-PA%+QA%)/2:UB%=QA%:QA%=H1%:B%=B%-UB%+H1%: PA%=PA%-UB%+H1%
580 IF ABS(U)>ABS(V) THEN US=ABS(U) ELSE US=ABS(V)
590 DW=1:O(1)=5:O(2)=2:CLS
600 II=0:WHILE DW*KY<5:DW*O((II MOD 2)+1):II=II+1:WEND
610 II=0:WHILE DW*KY>20:DW=DW/O(2-(II MOD 2)):II=II+1:WEND
620 IF V<>0 THEN FOR I%=A% TO VA% STEP DW*KX*V/ABS(V):PSET(I%,B%-1):NEXT
630 IF U<>0 THEN FOR I%=A% TO UA% STEP DW*KX*U/ABS(U):PSET(I%,B%-1):NEXT I%
640 IF ABS(P)>ABS(Q) THEN US=ABS(P) ELSE US=ABS(Q)
650 PRINT "I=";DW
660 LINE(A%,QA%)-(A%,PA%):LINE(A%-2,QA%+4)-(A%,QA%):LINE(A%+2,QA%+4)-(A%,QA%)
670 LINE(UA%,B%)-(VA%,B%):LINE(VA%-8,B%-1)-(VA%,B%):LINE(VA%-8,B%+1)-(VA%,B%)
680 IF P<>0 THEN FOR I%=B% TO PA% STEP -DW*KY*P/ABS(P):PSET(A%+2,I%):NEXT I%
690 IF V<>0 THEN FOR I%=B% TO QA% STEP -DW*KY*Q/ABS(Q):PSET(A%+2,I%):NEXT I%
```

```
700 FOR I%=1 TO M%:FOR J%=0 TO N%:X%(I%,J%)=A%+KX*X(I%,J%):Y%(I%,J%)=B%-KY*Y(I%,J%):NEXT
710 IF A2$<>"Z" AND A2$<>"z" THEN 730
720 FOR J%=0 TO N%-1:LINE(X%(I%,J%),Y%(I%,J%))-(X%(I%,J+1),Y%(I%,J%+1),3-(I% MOD 3):NEXT :GOTO 740
730 FOR J%=0 TO N%-1:LINE(X%(I%,J%),Y%(I%,J%))-(X%(I%,J%+1),Y%(I%,J%+1)):NEXT
740 NEXT I%
750 BB$=INKEY$:IF BB$<>"" THEN 750
760 BB$=INKEY$:IF BB$="" THEN 760
770 IF A2$<>"z" THEN 790
780 IF BB$="C" OR BB$="c" THEN CC=CC+1:CC1=INT((CC1+1)/2-1):COLOR 1+(CC1 MOD 15),CC MOD 2:GOTO 750
790 A%=A1:UA%=UA1:VA%=VA1:HL%=HL1:KX=KX1:KY=KY1
800 IF BB$<>"a" AND BB$<>"A" THEN 830
810 IF A2$="z" THEN A2$="g":SCREEN 2 ELSE A2$="z":KX=KX/2:A%=A%/2:VA%=VA%/2:UA%=UA%/2:HL%=319:SCREEN 1:COLOR 0,1
820 GOTO 520
830 IF BB$="S" OR BB$="s" THEN FF=FF+1:IF FF(5 THEN KX=KX/1.2^FF:KX1=KX:KY=KY/1.2^FF:KY1=KY:IF A2$="z" THEN KX=KX/2:HL%=319:GOTO 520 ELSE 520 ELSE 840 ELSE 850
840 FF=0:A%=A1:UA%=UA1:VA%=VA1:HL%=HL1:KX=KX2:KY=KY2:KX1=KX2:KY1=KY2:A2$="g":SCREEN 2:GOTO 520
850 IF BB$<>"L" AND BB$<>"l" THEN 900
860 IF A2$="z" THEN SCREEN 2:A2$="g"
870 IF B%-KY*P*6/5(=189 AND B%-KY*Q*6/5)10 THEN KY=KY*6/5: GOTO 520
880 IF -P/(180-B%))Q/(B%-10) THEN K=(180-B%)/(-P) ELSE K=(B%-10)/Q
890 KY=K:KX=2*K:GOTO 520
900 IF BB$="r" OR BB$="R" THEN RUN ELSE IF BB$="q" OR BB$="Q" THEN END ELSE 750
```



(图1)



(图2)

若干子目录下共用一个 dBASE Ⅲ (或 WS)软件包

湖北供销学校　　龚文彪

利用 DOS3.3 中的 APPEND 命令和 PATH 命令配合设置就能实现在若干子目录下共用一个 dBASE Ⅲ (或 WS)软件包。

假如在 PC/XT 机的根目录下有许多子目录，如 DOS、DB3、GZ(工资)、RS(人事)、CW(财务)等，只要把 DBASE Ⅲ 系统文件(DBASE.EXE 和 DBASE.OVL)放到子目录 DB3 中，其他子目录只放 dBASE Ⅲ 编的各种应用程序(如工资程序、人档程序、财务程序等)，只要在 AUTOEXEC.BAT 文件中添加如下两条命令：

```
PATH C:\;C:\DOS;C:\DB3;C:\GZ;C:\RS;C:\CW
C:\DOS APPEND C:\;C:\DOS;C:\DB3;C:\GZ;C:\RS;C:\CW
```

修改后存盘，重新启动机器，这时你就可在任意子目录下运用 dBASE Ⅲ 编的各种应用程序。即达到在若干子目录下共用一个 dBASE Ⅲ 软件包的目的。

说明：(1)APPEND.EXE 是 DOS3.3 外部命令，放在 C 盘的子目录 DOS 中。

(2)把 WORDSTAR 字处理软件放在 DB3 子目录中，在任意子目录下用 WS 编辑各种文稿或编制、修改程序，甚感方便。其他软件包也可这样做。

小议 PC-DOS 的拷贝命令

辽宁师范大学物理系　　刘杰

PC-DOS 有关文件拷贝的命令有：

DISKCOPY，COPY，SYS，BACKUP，RESTORE，其中以前二者用得最多，各命令的使用不尽相同。

(1)DISKCOPY 磁盘拷贝命令是将源磁盘一模一样的复制到目的磁盘上，磁盘上的每条磁道和每个磁区的内容都一样。若目的盘未格式化，则在拷贝的同时也对目的盘格式化。它不仅能拷贝显含文件，也能拷贝隐含文件 IBMDOS.COM，IBMBIO.COM。该命令只能用于同性质磁盘间，如软盘之间，硬盘之间，不能用于软硬盘之间或不同性质驱动器之间拷贝。

(2)COPY 拷贝命令可把源磁盘显含文件的全部或部分拷贝至目的磁盘，不能拷贝隐含文件 IBMDOS.COM，IBMBIO.COM，也不能将目的盘格式化，COPY 命令能重新安排各文件在磁盘上的位置并可用于不同性质磁盘间的拷贝。

(3)SYS 是将隐含文件 IBMDOS.COM，及 IBM-

BIO.COM 系统程序从第一个磁盘转移至第二个磁盘。在进行格式化时，如果要预先留下位置给系统程序，可使用 FORMAT /B。如果系统程序位置已被其它程序占用，该命令无效。

(4)BACKUP、RESTORE 是配对使用的。BACKUP 是将一个磁盘上的一些文件，转移至第二个磁盘作为备份，通常从硬盘转移至软盘，以防硬盘损坏而作备用。隐含、显含文件均可备分。BACKUP 形成的目的文件，加上了一些控制数据，这些数据是为 RESTORE 而设计的，备份盘上的文件要由 RESTORE 命令恢复后，方可正常使用，而由 COPY 命令形成的目的文件与源文件一模一样。这一点与 COPY 命令不同。

能读多位数的计算机程序

贵州　中国人民银行剑河支行　　周礼森

贵刊在创刊号刊出的《"会念数"的程序》以及在八七年第二期刊出的《计算机能读出你输入的数》，为计算机摸拟人读数的程序设计提供了思路。但所给程序均有其局限：一是两文均用拼音方式读数不直观，二是读数范围仅局限于 1～99 或 1～999 之间。

下面是笔者设计的程序，利用汉字方式能读出 10^{-15}～10^{16}之间的任意整数和小数，读数规则完全符合人们的读数习惯。当你用键盘输入一个数时，计算机立即在屏幕上用汉字显示这个数的读法，如键入"10801.45"这个数，屏幕立即显示：一万零八百零一点四五。程序可在任何由 CCDOS 支持的微机上运行。

```
10 INPUT "请输入要判读的数";X#:A#=INT(X#)
20 IF A#=0 THEN X#=X#/10:GOTO 30
30 X$=STR$(X#):A$=STR$(A#):K=LEN(A$)-1:L=LEN(X$)-1:B=16-K
40 B$="零一二三四五六七八九":C$="千百十万千百十亿千百十万千百十个"
50 DIM C$(L),C(L),D$(L),B$(L)
60 IF A#=X# THEN E$=A$:Q=K:GOTO 80
70 E$=X$:Q=L
80 GOSUB 800
90 GOSUB 1000
100 IF A#=X# THEN 120
110 PRINT "点";:FOR I=K+1 TO Q:PRINT D$(I);:NEXT
120 END
800 FOR I=1 TO Q
810 IF I=K+1 THEN 850
820 C$(I)=MID$(E$,I+1,1):C(I)=VAL(C$(I)):D$(I)=MID$(B$,2*C(I)+1,2)
830 IF I>K THEN 850
840 B$(I)=MID$(C$,2*(B+I)--1,2)
850 NEXT:RETURN
1000 FOR I=1 TO K
1010 IF K=1 THEN 1090
1020 IF C(I)=0 THEN 1050
1030 IF I=K THEN 1090
1040 PRINT D$(I)+B$(I);:GOTO 1100
1050 IF K>8 THEN IF B$(I)="万" AND C(I-1)=0 THEN IF C(I-2)=0 AND C(I-3)=0 THEN 1100
1060 IF B$(I)="亿" OR B$(I)="万" THEN PRINT B$(I);:GOTO 1080
1070 IF I=K THEN 1100
1080 IF C(I+1)=0 THEN 1100
1090 PRINT D$(I);
1100 NEXT:RETURN
run
请输入要判读的数？900000000000189
九百万亿零一百八十九
run
请输入要判读的数？0.000876
零点零零零八七六
```

模拟钢琴演奏

重庆电力学校计算机八一班　　冯剑秋

在 PC BASIC 中，一般演奏音乐的方式，是采取按预定的简谱编程序来实现的。换一首歌谱，就要另编一个程序，显得很麻烦。为了能充分利用 IBM PC 的丰富音乐功能，我编了一个小程序，使计算机能用键盘按简谱"弹"出悦耳的乐音，就像钢琴一样。

在本程序中，如果运行时你键入 C、D、E、F、G、A、B（相当于 C 调中的多、瑞、咪、发、梭、拉、西）七个键中的任意一个，便能发出相应的简谱的音。假如你要演奏高音多，那么按一下 M 键，再按 C 键，那么便是高音了。现在又要演奏中音咪，那么按一下 N 键，便回到了中音段，再按一下 E，便就是中音咪。假如又要演奏低音，那么便又按一下 N，就到了低音段，再按相应的键便可以发出低音了。要结束演奏，按字符"O"键便结束了。程序中 C$ 中的字串"L16"可以改变，它表示音乐的音长为 1/16 拍。读者可以按简谱把"L16"中的"16"改为其它的音长。便可以控制发音的长短。

可以看出，使用这种方法，可以随心所欲演奏任意的曲子，具有很高的趣味性，您不妨一试。

```
30 A$=INKEY$
40 IF A$="O" THEN END
50 IF A$="M" THEN B$="〉":L$="T":GOTO 80
60 IF A$="N" THEN B$="〈":L$="T":GOTO 80
70 L$="F"
80 IF L$="T" THEN A$=INKEY$:C$=B$+"L16"+A$
90 IF L$〈〉"T" THEN C$="L16"+A$
100 PLAY C$
110 GOTO 30
```

如何对中文 DOS 的汉字库进行直接读写

湖南　刘旭华

笔者为了在西文 DOS 下直接显示汉字，对中文 DOS 的汉字库进行了分析，发现 CCDOS2.12H 的汉字库的记录号与汉字区位码存在下面关系。

将区码减 1 再乘以 94，然后加位码，就是本字对应库中的记录号，比如汉字“师”，区位码为 4206，对应的记录号 R＝(42－1)×94＋6＝3860

对于 CCDOS2.1 等汉字系统的库，与上略有不同，对应关系如下(QW 为汉字区位码)：

$Q_1=INT(QW/100)$

$Q_2=QW-Q_1\times 100$

$$Q_1=\begin{cases}6 & Q_1=9\\ Q_1-8 & Q_1>15\\ Q_1\end{cases}$$

$R=(Q_1-1)\times 94+Q_2$　(R 为记录号)

根据这些关系，我们可以直接在西文 DOS 下读取对应的汉字，并将其显示出来。

程序(一)是一个汉字显示程序，可以任意放大，并变换出许多花样。当不放大时，本程序采用了一个小技巧，显示速度很快(见 390 句)。

程序(二)是将 CCDOS2.1 的汉字显示字库转换到 CCDOS2.13H 的字库中去的一个小程序，喜欢 CCDOS2.1 的粗体字的人不妨一试。

程序一

```
1 XX=10 :YY=10:YK=YY
3 INPUT KL,YL
20 KEY OFF:CLS
30 OUT 985,10
40 DIM A%(18):A%(0)=16:A%(1)=16
50 OPEN"C:\213\hzk16" AS#1 LEN=32
60 FIELD #1,32 AS A$
70 LOCATE 10,10:QW=4206:GOSUB 310
75 XX=XX+4*KL
80 LOCATE 10,12:QW=2083:GOSUB 310
85 XX=XX+4*KL
90 LOCATE 10,14:QW=2429:GOSUB 310
95 XX=KL+4*KL+XX
100 LOCATE 10,16:QW=5448:GOSUB 310
265 YY=YY+1
280 K$=INKEY$:IF K$="" THEN 280
290 CLOSE:END
310 XC=XX
320 Q=INT(QW/100):W=QW-Q*100
330 REM )49 THEN JL=(Q-1)*94+W
340 JL=(Q-1)*94+W
350 GET #1,JL
360 FOR I=1 TO 16
361 XX=XC
365 YY=YY+(YL-1)*2+1
370 A%(I+1)=CVI(MID$(A$,2*I-1,2))
371 IF KL=1 AND YL=1 THEN 380
375 GOTO 1000
380 NEXT
381 YY=YK
382 IF KL=1 AND YL=1 THEN 390
385 RETURN
390 PP=POS(0)*8-8:CC=CSRLIN*8-8:PUT(PP,CC),A%
400 RETURN
1000 X=A%(I+1):C$=""
1005 IF X<0 THEN X=(X)+2^15
1010 B$=STR$(X-INT(X/2)*2):B$=MID$(B$,2,1):C$=B$+C$
1020 X=INT(X/2)
1030 IF X>=2 THEN 1010
1040 IF X=1 THEN C$="1"+C$
1045 IF LEN(C$)<15 THEN FOR U=1 TO 15-LEN(C$):C$="0"+C$:NEXT
1047 F$=MID$(C$,1,7):G$=MID$(C$,8,8):C$=G$+F$
1050 LL=LEN(C$):FOR O=1 TO LL
1060 SS$=MID$(C$,O,1)
1065 IF KL=1 THEN RR=0
1066 IF KL>1 THEN RR=KL*2
1070 XX=XX+2*(KL-1)+1:IF VAL(SS$)=1 THEN CIRCLE(XX,YY),RR/2,'IF KL>1 THEN PAINT(XX+RR/2-1,YY),1
1080 NEXT
2000 GOTO 380
```

程序二

```
10 SCREEN 2,0
15 DIM A%(18):A%(0)=16:A%(1)=16
30 WIDTH 80:CLS
40 NA1$="C:\213\hzk16":NA2$="c\213\cclib"
41 LOCATE 20,40:PRINT "WAITING……"
45 OPEN NA2$ AS#1 LEN=32
46 OPEN NA1$ AS#2 LEN=32
50 FOR I= 1601 TO 8793
60 IF INT(I/100)=I/100 THEN 420
80 FIELD #1,32 AS A$
90 Q=INT(I/100):W=I-Q*100
100 IF Q=9 THEN Q=6
340 IF Q>15 THEN Q=Q-8
350 GET #1,(Q-1)*94+W
380 FIELD #2,32 AS B$
385 LSET B$=A$
390 Q=INT(I/100):W=I-Q*100
400 PUT #2,(Q-1)*94+W
420 NEXT I:CLOSE
```

NC—新型文件管理工具软件

杨 进

NC—Norton Commander 是由 Peter Norton Computing 于 1986 年推出的适用于 IBM—PC 及其兼容机的新型文件管理工具软件。以其小巧的体积，灵活的使用，方便的操作而独树一帜，某些功能甚至超过了长期居统治地位的 Pctools。

NC 由四个文件组成：NC. EXE(主运行文件)，NC. INI(运行环境设置文件)，NC. MNU(用户定义功能键菜单文件)，NC. EXT(用户定义非执行文件处理文件)，总长度约为 102K，比起 Pctools4. 0 以上的版本要小得多，但在文件管理上完全可以和 Pcshell 5. 0 媲美，现将其主要特点介绍如下：

一、双窗口显示，直观方便。NC 在屏幕上开辟了两个窗口，可同时显示，两个磁盘或路径的文件目录，这两个窗口的显示行数，显示方示皆可方便调节，可满足不同的需要。用 TAB 键可转换当前窗口，用 CTRL—U 可交换两个窗口的内容，还有其他一些功能键可通过 F1 获得提示帮助。值得一提的是，NC 还支持独特的 EGA 状态下可显示 43 行，更加方便对大量文件的同时处理。

二、对子目录的操作简便迅速。用过 Pctools 的人都知道，当目标盘的树目录结构十分庞大时，要想改变当前路径往往是十分费时费力的，相比之下 NC 读取目录树的速度就快得多，同时进入子目录也只需把光标移到子目录名上，然后一按回车即可，而不必象 Pctools 那样，每次改变都要查找甚至重读目录树，节省了大量时间。

三、良好的 DOS 界面。当屏幕左下角有光标闪烁时，可直接输入 DOS 命令或者运行文件名，如 DIR，TYPE，BASIC 等。但此时应保证在 NC 的同一目录中或 A 盘或 C 盘根目录中有 COMMAND. COM 文件，另外要保证运行文件的存在，否则 NC 将给出出错提示。(若事先用 PATH 命令指明了 COMMAND. COM 的路径亦可。)

四、灵活的用户自定义功能，这也是 NC 最独到之处。NC. MNU 是一个由用户定义功能键的文件。其基本格式为：

```
功能键名:注释
        命令
```

我们可以用 NC 的 EDIT 功能或 COMMAND 菜单中的 Menu file edit 功能编辑一个如下的 NC. MNU 文件：

```
D:      Directery of drive A
        dir a:/p
F1:     Pctools
        c:\tools\pc
F2:     Personal Edit II
        c:\tools\pe2
F3:     Norton Utilities
        d:\ni\nu
```

编辑完毕，存盘退出，再按 F2 键进入 USER 菜单，即可使用自定义的功能键。此文件的含义为：按"D"键则列 A 盘目录，按"F1"键则运行 C 盘 TOOLS 目录中的 Pctools，余类推。应注意，用户自定义的功能键只有在进入 USER 菜单以后才有效。

NC. EXT 是另一个由用户自定义的文件，它的作用在于告诉 NC 如何处理非执行文件。在使用 NC 时，当把光标移到一个可执行文件上时，一按回车，NC 就自动执行该文件。当把光标移到一个非执行文件上时，一按回车，NC 就查找 NC. EXT 文件，根据这一类非执行文件在 NC. EXT 中的定义进行处理，其基本格式为：

后缀:命令(运行文件名)!. 后缀(当命令不要求时可省略)

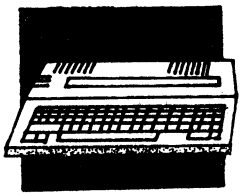
我们可以利用 COMMAND 菜单中的 extension file edit 功能编辑这样一个 NC. EXT 文件：

```
BAK: TYPE !. BAK
BAS: C:\BAS\GWBASIC !
PRG: C:\DBA\DBASE!
  * : ren no runfile for this extension
```

编辑完存盘退出，就可使 NC 按这一定义进行处理。其含义为：当文件为. BAK 文件时，则显示其内容；为. BAS 文件时，则调用 C 盘 BAS 目录下的 GWBASIC 运行此文件；为. PRG 文件时，则调用 C 盘 DBA 目录下的 DBASE 运行此文件；为其它未被定义的后缀时，则显示"rem no runfile for this extension"。

需要说明的是：如果用 NC 的 EDIT 功能编辑 NC. EXT 文件，在编辑完存盘后，必需先退出 NC，然后重新进入 NC，才能使 NC 按新的定义处理，否则将仍然按照原来的定义执行。

另外，对于 NC 的显示方式，用户可根据自己的需要或习惯进行设置，设置完毕后，利用 NC 的 OPTION 菜单中 Save setup 功能将其存入 NC. INI，下次使用 NC 时，即可直接按新设置显示。



学习机之友

为无驱动器系统设置虚拟磁盘

刘庆丰

目前与苹果机兼容的各类学习机已大量进入家庭。但不少用户均未配置驱动器，只用录音机作外存，这种结构使全机的性能大为降低，且使大量在 DOS3.3 系统下开发的软件均不能运行。但这类学习机均有扩充 16k 存储器，完全可以用软件的方式将其生成为一台虚拟磁盘机。

下面即是我参考有关资料编写的虚拟磁盘程序。首次使用前先将在有驱动器机子上录下的 $9D00.$BFFF 间的数据读入内存，再键入或读入程序一，并继续键入：B7B7:20 9A BE NADC0:20 61 BD NB3AF:BA E5 E5 F2 C6 A0 D3 CF C4 AD CE D8 N9CFD:4C 00 BD。将 $9CFD～$BFFF 间的数据录下来，就成为一盒“虚拟磁盘引导带”了。

用法：将此虚拟磁盘引导程序读入内存并从首址运行，屏幕将显示“DOS-XNP W:BDAD”等字符，此时整机已在 DOS3.3 系统管理下工作，可以使用所有的 DOS 命令了，使用方法与效果和在驱动器上完全相同且速度快得多。本系统为用户提供了 60 扇区(15k)容量(每次 CATALOG 时均会自动显示剩余扇区数值)，共可容纳 21 个文件。虚拟盘储满后将显示 DISK FULL，此时及关机前若需将“盘”内信息脱机保存，可键入 CALL48557 或 BDADG，屏幕将显示 BSAVE “DOS”,A $3FFD,L $4057 字样。移动光标划过这些字符并回车，同时启动录音机，就将全“盘”内容整体存入了磁带。这个命令是为小蜜蜂(XMF-1)学习机设置的，读回磁带时只需使用 BRUN“XNP”命令就可将所有信息重新装入虚拟磁盘。对于 CEC 等其他苹果兼容机，只需对程序作如下改动就可在监控状态下存带，读回时改用 3FFD.8054R 命令并从首址运行：

BE75:A0 B3 C6 C6 C4 AE B8 B0 B5 B4 D7 A0

程序一

```
BD00-20 58 FC A2 18 BD 21 BD
BD08- 9D 05 04 CA D0 F7 A2 25
BD10- BD 3A BD 9D CF 03 CA D0
BD18- F7 20 51 A8 20 D4 A7 4C
BD20- 19 BF AA AA AA A0 D8 CE
BD28- D0 AD C4 CF D3 A0 AA AA
BD30- AA A0 A0 A0 D7 BA C2 C4
BD38- C1 C4 A0 4C BF 9D 4C 84
BD40- 9D 4C FD AA 4C B5 B7 AD
BD48- 0F 9D AC 0E 9D 60 AD C2
BD50- AA AC C1 AA 60 4C 51 A8
BD58- EA EA 4C 59 FA BF 9D 38
BD60- 00 A9 00 85 40 20 86 BD
BD68- 18 B9 F2 B3 F0 0E 0A 90
BD70- FB 48 E6 40 D0 02 E6 41
BD78- 68 18 90 F0 88 D0 E9 A6
BD80- 40 A5 41 4C 24 ED 85 41
BD88- AD EF B3 0A 0A 85 06 A4
BD90- 06 60 B1 3C 91 42 E6 42
BD98- D0 02 E6 43 A5 3C C5 3E
BDA0- A5 3D E5 3F E6 3C D0 02
BDA8- E6 3D 90 E6 60 A9 00 85
BDB0- 42 85 3C A9 40 85 43 A9
BDB8- D0 85 3D A9 FF 85 3E 85
BDC0- 3F A0 00 2C 80 C0 20 92
BDC8- BD 2C 88 C0 A9 00 85 42
BDD0- 85 3C A9 70 85 43 A9 D0
BDD8- 85 3D A9 FF 85 3E A9 DF
BDE0- 85 3F A0 00 20 92 BD 2C
BDE8- 82 C0 A2 00 BD 75 BE 20
BDF0- F0 FD E8 E0 19 90 F5 A9
BDF8- 00 85 42 A9 80 85 43 A9
BE00- 22 85 3C A9 BE 85 3D 85
BE08- 3F A9 74 85 3E A0 00 20
BE10- 2C FE A9 4C 8D FD 3F A9
BE18- 00 8D FE 3F A9 80 8D FF
BE20- 3F 60 8D 81 C0 8D 81 C0
BE28- A9 00 85 42 85 3C A9 D0
BE30- 85 43 A9 40 85 3D A9 FF
BE38- 85 3E A9 6F 85 3F A0 00
BE40- 20 2C FE 8D 89 C0 8D 89
BE48- C0 A9 00 85 42 85 3C A9
BE50- D0 85 43 A9 70 85 3D A9
BE58- FF 85 3E A9 7F 85 3F A0
BE60- 00 20 2C FE 8D 82 C0 A2
BE68- 00 BD 8F BE 20 F0 FD E8
BE70- E0 0A 90 F5 60 A0 C2 D3
BE78- C1 D6 C5 A2 D8 CE D0 A2
BE80- AC C1 A4 B3 C6 C6 C4 AC
BE88- CC A4 B4 B0 B5 B7 A0 D2
```

NBDF4:0C

有了这套虚拟磁盘系统，在很大程度上相当于配置了一台驱动器，不仅可进行各种顺序、随机文件的实际操作，更重要的是大量在 DOS 系统下开发的软件或操作系统，几乎都能够使用了。比如 DOS3.3 系统主盘上的一些非常有用的工具软件，如 RENUMBER(更改行号及程序合并)、CHAIN(程序链接)等，都可将其存放于虚拟盘中，供随时调用。若软件占内存较大，虚拟盘无法容纳时，还可只将中途需读盘的部分存于虚拟磁盘内，而将主体程序仍存于内存，这样能运行的软件范围就很广泛了。

由于本系统并不存在一个实际的“磁盘”，所以在同别人交换软件时仍需通过磁带进行。至于如何将 A 类 BASIC 文件和 B 类二进制文件存入磁带，相信大家都已很熟悉，但存储 T 型文本文件的方式却很少有人提到，而 T 型文件在 DOS 系统中又是一类非常有价值的文件。这里给出两段小程序，程序二可将 T 型顺序文件变成内存中的二进制信息并给出首末地址，以便按 B 类方式存储，程序三则将内存中的二进制信息重新按指定的文件名生成 T 型文件。

使用本系统还需注意以下几点：1. 由于 16k 卡已为虚拟磁盘所占用，所以如整数 BASIC、MIT LOGO 等一些需占用 16K 卡的系统将不能载入，DOS 的 INT 命令也不能使用，否则将显示 LANGUAGE NOT AVAILABLE 信息；2. 从首址起动及 INIT 命令都将使虚拟盘全部清零，所以使用这两个命令一定要慎重，当然也可利用这种特性来快速清“盘”；3. 本程序存放于 DOS 系统的 RWTS 区，不另占任何空间，故对各种系统的运行均无任何影响。但有个别软件运行中直接调用 RWTS 来驱动磁盘，在本系统中运行将有一定困难。

```
BE90- C1 CD AD D8 CE D0 A0 A1
BE98- A1 00 84 48 85 49 AD F4
BEA0- B7 F0 58 C9 04 D0 03 4C
BEA8- 0C BF 48 AD EC B7 C9 13
BEB0- 90 04 E9 04 B0 F8 C9 0F
BEB8- B0 04 69 04 90 F8 8D 78
BEC0- 04 E9 0F AA AD F0 B7 85
BEC8- 3E AD F1 B7 85 3F BC 76
BED0- BF B9 83 C0 B9 83 C0 BD
BED8- 7A BF 0D ED B7 85 47 A9
BEE0- 00 85 46 68 4A B0 0B A0
BEE8- 00 B1 3E 91 46 88 D0 F9
BEF0- F0 09 A0 00 B1 46 91 3E
BEF8- 88 D0 F9 2C 82 C0 AD 75
BF00- BF 8D F7 B7 AD 97 BF 8D
BF08- F6 B7 18 60 20 19 BF A9
BF10- 00 85 48 20 EA A2 4C 8B
BF18- 9F A9 FF 85 47 A0 00 84
BF20- 46 98 2C 83 C0 2C 83 C0
BF28- 91 46 2C 8B C0 91 46 88
BF30- D0 F3 C6 47 A5 47 C9 D0
BF38- B0 E7 A0 0D B9 7E BF 99
BF40- 74 E0 88 10 F7 A0 07 B9
BF48- 8C BF 99 30 E0 88 10 F7
BF50- A9 7A 8D 27 E0 A0 03 B9
BF58- 94 BF 99 00 E0 88 10 F7
BF60- A9 11 A0 02 8D 01 E3 8C
BF68- 02 E3 88 8D 01 E2 8C 02
BF70- E2 2C 82 C0 60 60 00 08
BF78- 08 08 D0 D0 E0 F0 FF FF
BF80- 00 00 FF FF 00 00 FF F0
BF88- 00 00 FF FF 11 01 00 00
BF90- 23 10 00 01 04 11 03 00
```

程序二

```
10 FOR I=768 TO 781:READ A:POKE I,A:NEXT:DATA 230,6,208,2,230,7,96,165,7,166,6,76,65,249
20 ONERR GOTO 100
30 POKE 6,0: POKE 7,32:D$=CHR$(4)
40 INPUT "NAME? ";A$
50 PRINT D$"OPEN"A$
60 PRINT D$"READ"A$
70 GET B$:B=ASC(B$)+128
80 GOSUB 150:GOTO 70
100 PRINT D$"CLOSE"
110 B=0:GOSUB 150
120 PRINT"$2000-$";:CALL 775
130 END
150 POKE (PEEK(6)+PEEK(7)*256),B:CALL 768:RETURN
```

程序三

```
10 D$=CHR$(4):A=8192
20 INPUT "NAME? ";A$
30 PRINT D$"OPEN"A$
40 PRINT D$"WRITE"A$
50 B=PEEK(A):A=A+1:IF B=0 THEN 80
60 PRINT CHR$(B);:GOTO 50
80 PRINT D$"CLOSE"
```

巧用 LOGO 解难题

四川　廖大勇

一九八六年全国青少年计算机程序设计竞赛题中，有一道“四狗互追”的难题，原题如下：

有一个正方形场地，在四个角上有四条猎犬，分别为 A，B，C，D，鸣枪之后它们同时起跑，并以相同速率 A 追 B，B 追 C，C 追 D，D 追 A，试编程在屏幕上显示它们跑出的轨迹（即跑过的路线）。由于 BASIC 语言无递归功能，编写的程序都要牵涉到较多较深的数学知识，难度较大，不易为学生理解掌握。

用 LOGO 语言的递功能来解此题，只需两个简单的过程，简洁明了，直观易读，充分体现出了 LOGO 语言简单易懂、化难为易的神奇本领！程序如下：

```
TO DOG
ASPECT 1 MAKE "A 120 PU FD 60 LT 90 FD 60 PD REPEAT 4 (LT 90 FD :A) RT 180 HT DOG1 :A
TO DOG1 :A
IF :A<3 THEN STOP
REPEAT 4 (FD 3 PU FD :A-3 PD RT 90) FD 3
RT (ATAN 3 :A-3) DOG1 SQRT(:A-3)*(:A-3)+3
END
```

修改 CEC 汉字系统程序

四川　泸州二中　周易

CEC-I 中华学习机的汉字系统管理程序被固化在辅存 ROM 的 $EC00～$FFFF 一段地址中，无法用监控命令 L 进行查看，更无法对其进行修改，使 CEC 机汉字系统的扩展很不方便。

下面这段机器语言程序专门为有志于 CEC 汉字系统扩展的微机爱好者而设计，它将辅存 ROM 中的汉字管理程序移到了 RAM 内存中的 $2C00～3FFF 这段地址中，并将其中的绝对地址改成了相对于 2C00 的地址，使用户更容易使用。现在不仅可查看它，更可以大胆地修改调用它为自己所用了。

利用这个方法，可使 CEC 具有同 STC 汉字系统相似的汉字处理功能，还可配合 F_4，F_5 键的扩展功能，使 CEC 汉字系统得到进一步的强化和扩充。

运行方法：在监控下键入 21CDG 即可。

```
21CD- A9 00 85
21D0- 06 85 08 A9 EC 85 07 A9
21D8- 2C 85 09 8D 0B C0 8D 06
21E0- C0 20 AB C3 20 ED 21 20
21E8- B9 C3 4C 00 22 A0 00 A2
21F0- EC B1 06 91 08 C8 D0 F9
21F8- E6 C7 E6 09 E8 D0 F2 60
2200- A9 00 85 06 A9 2C A6 EC
2208- A0 00 B1 06 C9 20 F0 0D
```

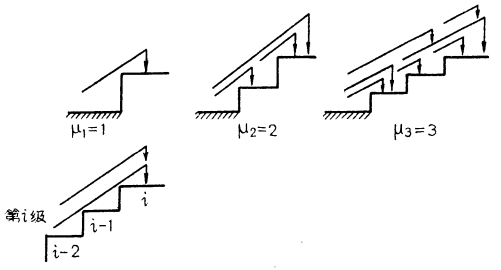
```
2210- C9 4C F0 09 C8 D0 F3 E6
2218- 07 E8 D0 EE 60 A5 06 48
2220- A5 07 48 98 48 8A 48 C8
2228- D0 06 E6 07 E8 D0 01 60
2230- C8 D0 06 E6 07 E8 D0 01
2238- 60 B1 06 C9 EC 90 0B 29
2240- 3F 91 06 68 68 68 68 4C
2248- 14 22 68 AA 68 A8 68 85
2250- 07 68 85 06 4C 14 22
```

浅谈计算机在中专数学中的应用

无锡无线电工业学校　王志平　顾正刚

在中等专业学校，如果将计算机课程的学习穿插在一些基础课教学中，尤其是有机地与数学结合起来，可增加它的趣味性和实用性。

例如在学习 BASIC 语言时，可以结合数学中一些较为繁琐和头痛的问题来讲解。这样，既可巩固所学的计算机知识，又可使学生感到学有所用，从而进一步激发学习热情。下面举两个简单的例子。



（一）在加深理解变量的概念的时候，希望学生学会使用变量循环更新的方法，我们就结合了中专数学中计算数列某项这一繁琐的问题。为了增加它的趣味性，设置了如下例题：

某数学教师上课时，问一同学他每次上楼梯一步跨几个台阶？同学回答：有时一步跨一个台阶，有时一步两个台阶。老师又问：按这种走法，从一楼走到三楼一共 40 级台阶，可有多少种方式走到三楼？学生哑口无言。这时老师就分析了这一题目：

跨上第 i 级只能有两种方式，即从第 i－1 级跨一步；从第 i－2 级跨二步。因此，$\mu_i=\mu_{i-1}+\mu_{i-2}$。这样，实际得到了如下一个数列：

1，2，3，5，8，……

该数列从第三项起，每一项都等于前两项之和。这一问题就是要求出该数列的第 40 项。这是件令学生头痛的问题。下面利用计算机分析解答。

设置两个变量 A 和 B，开始令 A＝1，B＝2。采用变量更新 A＝A＋B，B＝B＋A。更新一次后，A 和 B 的值分别为 3 和 5，正好是数列的第三，四项。如此更新下去可得数列的各项。因此，本问题的程序如下：

```
5 LET A=1:B=2
10 FOR I=1 TO 19
20 LET A=A+B:LET B=B+A
25 NEXT I
30 PRINT "THE WAY=";B
35 END
RUN↙
THE WAY=1.6558E+08
```

（二）在线性方程组一章中，为了学会线性方程组的解法，结合中专教材《数学》第四册，可以讲解用高斯消元法解线性方程组的程序。

［例］用高斯消元法解联立方程组：

$$3X+2Y+Z=14$$
$$X+Y+Z=10$$
$$2X+3Y-Z=1$$

在这里可与同学们讲解高斯消元法的程序：

```
10 DIM A(3,4)
20 FOR I=1 TO 3:FOR J=1 TO4
40 READ A(I,J)
50 NEXT J:NEXT I
70 FOR J=1 TO 3
80 IF A(J,J)<>0 THEN 150
90 FOR I=J TO 3
100 IF A(I,J)<>0 THEN 140
110 NEXT I
120 PRINT"No unique solution or no solution "
130 GOTO 610
140 GOSUB 300
150 LET C=1/A(J,J):GOSUB 400
170 FOR I=1 TO 3
180 IF I=J THEN 210
190 LET C=-A(I,J):GOSUB 500
210 NEXT I
220 NEXT J
230 FOR I=1 TO 3
240 PRINT "X(";I;")=";A(I,4)
250 NEXT I
260 GOTO 610
300 FOR K=J TO 4
310 LET B=A(J,K):A(J,K)=A(I,K):A(I,K)=B
340 NEXT K:RETURN
400 FOR I=J TO 4
410 LET A(J,I)=A(J,I)*C
420 NEXT I:RETURN
500 FOR S=J TO 4
510 A(I,S)=A(I,S)+C*A(J,S)
```

```
520 NEXT S:RETURN
600 DATA 3,2,1,14,1,1,1,10,2,3,-1,1
610 END
RUN↙
X(1)=1
X(2)=2
X(3)=7
```

对于任意元线性方程组都可以用以上程序求解，只需将数组 A(3,4)及相应的 FOR-NEXT 语句中循环终值修改为合适的数据即可。

通过以上的简单例子，我们体会到将计算机和数学有关内容有机地结合在一起，一定可以收到良好的教学效果。实际上，也可以将中专的计算机教学渗透到其它一些学科中去，这样可以使学生学会利用计算机解决一些实际问题，为走上工作岗位后普及应用计算机打下良好的基础。

数据块的后移

成都商业学校　冯端品

在监控系统中有一个移动数据的 M 命令，用它把数据快向前移动是没有任何问题的，但是用它把数据块向后移动，当源地址和目标地址发生交叉时，会丢失数据。分析 M 命令的执行过程，原来它在移动数据块时，先移第 1 个数据，然后移第 2 个数据，……直到最后一个数据。这样向前移是对的，向后移就可能发生问题。那么我们只要想办法在向后移数据块时，把次序反过来，先移最后 1 个数据，然后移倒数第 2 个数据……直到第 1 个数据，就不会发生问题了。

我用机器语言编了一段程序，解决了把数据块向后移动的问题。为了应用方便，程序地址是浮动的。

把程序调入内存任何方便的位置，然后把程序的开始地址送入 $3F9、$3FA 单元，低位在前，高位在后。当需要把数据块向后移动时，在监控状态下输入

目标地址＜源数据块首址．末址 CTRL-Y↙

```
0300- A5 3C 85 FA A5 3D 85 FB
0308- A5 3F 85 FD A5 3E 85 FC
0310- 38 E5 FA 85 08 A5 FD E5
0318- FB 85 09 A5 44 18 65 08
0320- 85 08 A5 45 65 09 85 09
0328- A0 00 B1 FC 91 08 A5 FC
0330- C5 FA D0 07 A5 FD C5 FB
0338- D0 01 60 A6 FC D0 02 C6
0340- FD C6 FC A6 08 D0 02 C6
0348- 09 C6 08 D0 DD F0 DB
```

从 QZC 卡读取汉字字模

北京机械工程学院附中　牛海峰

QZC 汉卡的功能很强，加装后许多指标已经略高 CEC 机一筹。它还留了两个语句，使用户自行定义其功能。这两个语句分别是 KEY 和 PLAY，开机初始化后被定义为：KEY 是追回刚被 NEW 掉的程序；PLAY 是用来在 PST 表内查找某行程序所在地址。下面介绍用 PLAY 语句显示出汉字字模的方法

用 MON A9000↙进入汇编过程，键入以下程序：

```
9000-  CALL 2B0E
       PUSH HL
       LD A,(7894)
       CALL 4E6C
       LD HL,7AB2
       LD B,20
900F-  LD A,(HL)
       CALL 5080
       INC HL
       DJNZ 900F
       POP HL
       RET
```

然后修改 PLAY 的指针，执行：MONW7991；C3，00，90↙指针即修改完毕。进入高显状态，执行 PLAY 16，01↙即看到显示出："啊"，然后显示啊的点阵字模。扩展后的 PLAY 语句也可用在程序中。

我在使用 QZC 卡编程中陆续发现了一些该卡的机器语言子程序入口，现公布如下：485CH：进入 SC 状态，使用寄存器 A，B。程序中利用 HALT 命令防止闪烁，原理是在场消隐期间改变状态。

5080H：把 A 中的二进制数换成 ASCII 字符印出

5189H：设原 A 寄在器存有 mm，L 寄存器存有 nn，本子程序把 A 和 L 的值以"mmnn－"的形式印出。

4E12H：进行 SC 的状态切换。(SC1↔SC0)

40CCH：把 ASCII 代码串换算成数值，入口时 HL 指向代码串首，处理后 HL 指向串尾＋1，结果存于 WRA1 和 DE 中。

4E6CH：汉字输出。入口时，A 内存入区码值，E 内存入位码，调用后，对应汉字将打印在当前位置上。

48A1H：把对应汉字字模从汉卡读出，送入以 7AB2H 开始的缓冲区中，区长 32 字节，A，E 同上。

2B0EH：此程序是在原驻机系统中的，用户用来扩展语句非常方便，功能是专用于测试："数据①，数据②"形式的代码串，并将数据①存于 7894 和 7897 中，数据②存于 A，E 中。

ProDOS 磁盘操作系统入门

廖　凯

前　言

目前中华学习机已经普及，广大用户对 DOS3.3 已比较熟悉，而对 ProDOS 磁盘操作系统却很陌生。我应编辑部的要求，将向大家系统地介绍 ProDOS 系统，使大家能够迅速地掌握 ProDOS 系统。此介绍只适合于中华学习、APPLE 机及其兼容机的用户。

第一章　ProDOS 简介

一　ProDOS 的功能

ProDOS(Professional Disk Operating System 专业磁盘操作系统)是 1984 年由苹果公司为克服 DOS3.3 所存在的缺陷而设计的一种高级磁盘操作系统。它是目前 APPLE 机功能最强的磁盘操作系统，它不仅包括了 DOS3.3 的一切功能，而且还具有以下功能：

1. 快速磁盘存取——8K 字节/秒，是 DOS3.3 的 8 倍。

2. 大容量存储——支持磁盘容量可达到 32M 字节，每一文件容量可达到 16M 字节。

3. 多层次文件结构——使用与 UNIX 相同的树形分层目录结构。

4. 文件类型可达 256 种，包括用户自定义类型。

5. 支持时钟/计时卡，自动给文件加上日期和时间。

6. 加强的 BASIC 指令。

7. 提供用户可改变的四个中断进入点。

8. 方便的 MLI(机器语言接口)调用。

9. 支持 RAM 磁盘。

10. 与 APPLE III 的 SOS 操作系统相兼容的文件结构。

二　ProDOS 的运行条件

ProDOS 系统必须具备以下条件才能真正运行：

1. 驻机 ROM 中必须是 APPLESOFT BASIC 语言解释程序。

2. 至少有 64K RAM 的内存。

＊3. 必须是 APPLE 公司的原装机。

注：这是指原版 ProDOS 系统盘而言，如果用户使用的是中华学习机或苹果兼容机，可参阅《电子与电脑》1991 年第 6 期的《ProDOS 系统盘的修改》一文，将 ProDOS 进行修改。修改后的 ProDOS 系统盘可在满足前二个条件的任何兼容机上运行。

三　ProDOS 的结构

ProDOS 系统经核心程序及解释程序提供有效的存储管理和中断处理。这核心程序即 MLI(Machine Language Interface 机器语言接口)，它包含四个部分：

1. 命令处理程序：由解释程序接收信息并确定所使用的 ProDOS 命令类型，如果命令有效，那么处理程序发送命令到操作系统内的另一部分，如果命令无效，则得到一个错误信息。

2. 分层文件管理程序：由 MLI 执行命令的发送、记载引导磁盘的磁道、发送数据到磁盘驱动程序、处理打开的文件及进行磁盘的管理。分层文件管理程序按逻辑次序在磁盘上存储数据。

3. 磁盘驱动程序：通过磁盘驱动设备进行数据读写，其磁盘驱动设备有 DISK Ⅱ，DuoDISK 和 RAM 磁盘。

4. 中断处理程序：ProDOS 允许将四级中断处理子程序同时装入内存，并按优先级别予以响应，独立工作。这给联网和通信以及实时控制创造了有利条件。

ProDOS 系统盘在市场上有两种版本：ProDOS USERS. DISK V1. 0. 1 及 ProDOS USERS. DISK V1. 1. 1

在系统盘上主要有四个系统文件，它们是：

1. PRODOS 是 ProDOS 系统的核心，它用于控制计算机与输入/输出(I/O)设备信息的流程，并经 BASIC 解释程序与用户沟通。

2. BASIC. SYSTEM 是磁盘操作命令解释程序，它提供加强的 BASIC 命令。BASIC. SYSTEM 解释程序根据缓冲区空间的自动分配情况，提供存储管理，随着文件的打开与关闭，它将自动向上或向下移动用户可使用的内存地址上界 HIMEM。

3. FILER 是磁盘及文件管理程序。它用于磁盘或文件的拷贝、比较以及磁盘的格式化和文件的上锁、改名等操作。

4. CONVERT 是转换程序，它可将文件在 ProDOS 和 DOS3.3 之间进行转换。

四　ProDOS 的启动

ProDOS 系统盘具有自启动功能。当你将系统盘插入驱动器后，开机即可进入主菜单。或者你可以使用 PR＃或 IN＃命令来启动系统盘. 在启动系统盘时，ProDOS 首先将 PRODOS 文件装入内存(首地址为 $2000)并执行。MLI 确定计算机存储容量并移动 MLI 到存储器较高的位置，而后确定所有在槽口上的设备并建立系统整体页面。随后 MLI 将 BASIC. SYSTEM 文件装入内存(首地址为 $2000)并执行。BASIC. SYSTEM 于是移动自己本身并更新系统整体页面。系统整体页面是解释程序与操作系统之间的通信环节。在系统整体页面内部是一个系统位元图(bit map)，它记载存储器从 $0000 到 $BFFF 各页的状态。解释程序决定各页是否显示或使用。在一个 64K RAM 的计算机上，ProDOS 系统占用内存 25K 字节，留给用户 35.5K 字节用于编程。

第二章　文件目录结构

ProDOS 使用多级目录结构，这给用户带来很多方便，它可以使用户按任务的不同层次、不同领域，建立多层次的分目录。如学校由校、系、年级、班级四层次组成，一个学校可以按其层次组织方式，采用多级目录结构形式，方便管理其所属的全部文件，这种多层次的树形结构有一个根目录，它是整个目录结构的基础.在根目录下，既可以是目录文件，也可以是普通文件。

ProDOS 这种分层文件结构，对于存储容量大的磁盘（例如 5M 温式硬盘）特别有用，而它在 5 寸软磁盘上用途则有限，因为这种软磁盘容量不大。

一　文件目录

每个 ProDOS 磁盘都有一个卷名，是在磁盘被格式化时所给予的。在列磁盘目录和进入主菜单时，磁盘卷名会被显示出来。在根目录下共可存放 51 个文件。

DOS3. 3 采用的是一级目录结构。所谓一级目录结构是指把系统中的所有文件都建立在一张目录表中，结构比较简单。但是一级目录结构存在不少的缺点，首先如果系统中有很多文件，要查找一个文件需要描扫整个目录表，使得查找目录时间增加；其次是不方便用户对文件的访问。一级目录结构的文件系统规定，用户不能给文件起相同的名字，给多用户带来很大的不便，因此在多用户的情况下，采用二级目录结构较为方便。二级目录结构是由一个主目录和它所管理的若干子目录组成。文件系统采用二级或多级目录结构后，就能够为用户提供文件可以同名和一个文件可以有多个不同名字的这种便利。

现在让我们看一个 ProDOS 使用子目录的例子：

/DATA

NAME	TYPE	BLOCK	MODIFIED
SPREADSHEET	DIR	1	NO DATE
WORD. PROCESSOR	DIR	1	NO DATE
DATA. BASE	DIR	1	NO DATE

这磁盘的卷名为 DATA，用子目录把资料分为 SPREADSHEET，WORD. PROCESSOR 和 DATA. BASE。我们可以在 SPREADSHEET 目录内存放空白表格文件。其它两个子目录也同样，你可以由 TYPE 项目得知这些文件是目录文件（类型为 DIR）。此时你可能需要一个字处理文件的列表，并找三个名为 CORP. POLICY，FRED 和 GENERAL. DATA 的文件，如下：

/DATA/WORD. PROCESSOR

NAME	TYPE	BLOCK	MODIFIED
CORP. POLICY	TXT	6	NO DATE
FRED	TXT	2	NO DATE
GENERAL. DATA	TXT	4	NO DATE

要在目录结构内观看其它文件，你必须按顺序给出正确的文件名。在上面的例子中，我们通知 ProDOS 寻找名为 DATA 的卷目录和名为 WORD. PROCESSOR 的子目录，计算机在 WORD. PROCESSOR 目录内读取指定文件并显示出来。

注意在上述的例子（/DATA）中，文本文件并未列出，列出的只是子目录，这是因为 ProDOS 一次只能显示一级目录。在这种情况下，它是目录结构的第一层。

二　路径名（PATHNAME）

当用户要访问多级目录中的某个文件时，需要用该文件的“路径名”来标识文件。所谓文件的路径名是指从主目录（根目录）出发，一直到所要找的文件，把途径的各目录名连接一起而形成的。每个目录名之间必须用“/”分开。例如：

/DATA/WORD. PROCESSOR/FRED

路径名前面的第一个“/”代表主目录（根目录）。该例子表示 ProDOS 首先寻找卷目录 DATA，然后找子目录 WORD. PROCESSOR，最后找到文件 FRED。如果你输入的路径名不对或文件名不在这儿，那么将会显示：PATH NOT FOUND。

三　部首（PREFIX）

在许多实例中，你要存取一个多级目录的文件时，需要连续打入路径名，这很麻烦而又费时。由于这个缘故，所以 ProDOS 提供一个部首命令 PREFIX。命令 PREFIX 用于定义一个路径名（PATHNAME）。当使用相同的路径名时，我们可以用 PREFIX 命令定义路径名：PREFIX/DATA/WORD. PROCESSOR/

这允许我们在 WORD. PROCESSOR 子目录内存取任何文件，除输入整个路径名的情况外。

例如我们要访问 FRED 文件，只要输入 FRED 就可以了。换句话说，在使用 PREFIX 以后，我们只要输入文件名即可。

注意在输入的文件名前面没有“/”。

如果在 WORD. PROCESSOR 目录中加入一个子目录 CONTRACTS，我们仍能在部首下存取所有文件或使用一个部分路径名，例如 PREFIX 仍设为/DATA/WORD. PROCESSOR/，要从 CONTRACTS 子目录中访问 LABOR 文件，我们就需输入：CONTRACTS/LABOR

注意在子目录 CONTRACTS 前面没有“/”，如果再加一个“/”，那么路径名将为 / DATA/WORD. PROCESSOR//CONTRACTS，这是错误的，即使在设置部首时后面未附“/”，ProDOS 也会自动加上。

在存取一个磁盘或文件时，ProDOS 只寻找卷名而不是驱动器号，这在写数据到数个驱动器时会很有用。不过 ProDOS 仍保留 DOS3. 3 的存取方式。

四　目录名和文件名的限制

ProDOS 要求一个目录名和文件名必须以字母开头，并且长度不能超过 15 个字符。文件名和目录名内的字符必须是相连的。ProDOS 将允许在一个文件名和目录名内使用大小写字母、数字和一个句号。

ProDOS 路径名的最长为 64 个字符，包括斜线。

注意：你不能 RUN 或执行目录文件。（未完待续）

在 APPLESOFT 系统下实现 HSCRN 功能

四川绵阳师专附中　李齐

在 CP/M 操作系统支持下运行的 GBASIC 语言为高分辨作图提供了一条 HSCRN 语句，它可以检查屏幕上的点是否显示。这对我们编写动画和游戏程序无疑是一大帮助。

可是在无 CP/M 操作系统下能否实现 HSCRN 功能呢？我们知道，APPLE－Ⅱ计算机的高分辨率图形方式有两页。它们的内存区分别是 $2000—$3FFF 和 $4000—$5FFF。下面我们主要就第一页进行讨论。

高分辨率图形的内存区中的一个字节对应屏幕上的一个显示单元，一个显示单元含有七个点。字节的低七位同七个点一一对应。如果某一点显示。那么这个点的对应位为1，否则为0。字节的最高位是彩色代码，我们这里不必深究。

另外还需要注意，显示单元与内存单元的地址排列对应关系并不是完全按照顺序的。内存单元的内容在显示时的次序是把屏幕分成三个部分，然后依次填入。所以不能直接用 X、Y 坐标得出与显示单元对应的内存单元，但可以用下面的公式求出。

AD＝8192＋[INT(Y/8) mod8]×128＋INT(Y/64)×40＋(Ymod8)×1024＋INT(X/7)

可以看出，要实现 HSCRN 功能，必须进行二进制数处理和地址计算。由于汇编语言处理二进制数的功能较强；而用 BASIC 语言进行复杂计算较为方便，所以采用汇编语言和 BASIC 语言相结合的方式。

以下是程序清单。读者可先将程序1输入内存，程序必须从 $0300开始装入，而将程序2做为子程序放在自己的程序中，需要时调用。AD 是点的显示标志，程序运行后 AD 的值如为1则表示显示，如为0则表示未显示。所以通过 AD 的值就可以看出点是否显示，从而实现了 HSCRN 的功能。在调用子程序前，应先将需要判定的点的纵坐标和横坐标的值分别赋给变量 Y 和 X。不要忘了在程序2后面加上 RETURN。

程序1

```
0300—   AD  13  03      LDA   $0313
0303—   AE  14  13      LDX   $0314
0306—   F0  05          BEQ   $030D
0308—   4A              LSR
0309—   CA              DEX
030A—   4C  06  03      JMP   $0306
030D—   29  01          AND   #$01
030F—   8D  13  03      STA   $0313
0312—   60              RTS
```

程序2

```
1000  AD＝8192＋(INT(Y/8)－8＊INT(Y/64))＊128＋
      INT(Y/64)＊40＋(Y－8＊INT(Y/8))＊1024＋INT
      (X/7)
1010  AD＝PEEK(AD):POKE 787,AD:AD＝X－7＊INT
      (X/7):POKE 788,AD:CALL 768:AD＝PEEK(787)
```

BASIC 语言中几个命令功能的扩展

那振弘

中华学习机和苹果Ⅱ微机（以下简称学习机）的 BASIC 语言中 GOTO，GOSUB，ON…GOTO，ON…GOSUB 以及 RESTORE 命令，可以通过一段短短的解释程序进一步扩展其功能。扩展功能之后，不影响原来命令的用法，不占用 BASIC 语言通常所用的空间，而且可以增加编程技巧，使之完成一些特殊的功能。本文旨在介绍扩展上述命令功能的实用方法。

1. 原解释程序的局限性

在学习机的浮点 BASIC 语言中，GOTO 语句是由 GOTO 命令和数值行号两部分构成的。由于它不能像整数 BASIC 语言那样用表达式表示行号，因此 GOTO 语句仅能实现绝对转移，而不存在使程序产生分枝的功能。GOSUB 命令也仅能调用以数值表示的、固定行号的子程序，这样就限制了程序的灵活性。

ON…GOTO(GOSUB)语句虽然可以实现程序的分枝，但是这种分枝仅仅是单结点的多枝树结构，分枝的数量增多表达起来不太方便，语句也较长。而且在 ON…GOTO(GOSUB)语句中作为判别分枝的开关变量的取值仅能是自然数。(例如 ON K GOTO…中的 K。)如果开关变量的值较为离散，那么就不得不再次为开关变量赋值，或是用空行号添补行号序列中与自然数序列不对应的部分。实际上这样的程序结构只要用 GOTO 命令和变量行号组成的语句就可以替代。

ON…GOTO(GOSUB)语句中的数值行号序列也可以扩展成为变量序列。这样通过开关变量、行号变量两级变化，可以构造出具有多结点、多分枝树和网结构的程序。同样，在条件转移语句 IF…THEN GOTO(GOSUB)语句中，用变量表示行号，可使该语句由条件单分枝结构(二叉树)变为条件多分枝结构。

在学习机的 BASIC 语言中，恢复数据命令 RESTORE 不能带参数。如果要想有选择地重读某些数据，势必要空读很多数据，且空读的次数不得有误。在一些微机中(例如 IBM－PC 机)，RESTORE 命令可以同其后的行号组成语句，这样就可根据需要来设置数

据指针，这在分析处理数据的程序中具有重要意义。我们可以对此语句的功能进行扩展，完成与 IBMPC 机类似的功能；而且 RESTORE 命令后面的行号还可以用表达式来表示，这样就会给编程带来很大的方便。

2. 解释程序的建立及说明

扩展上述功能的解释程序共用56条汇编指令，占用第3页起始的119个内存单元；反汇编如下

```
0300—   C9  B4        CMP     # $ B4
0302—   D0  2F        BNE     $ 0333
0304—   20  B1  00    JSR     $ 00B1
0307—   20  F8  E6    JSR     $ E6F8
030A—   C9  AF        CMP     # $ AF
030C—   F0  03        BEQ     $ 0311
030E—   4C  EF  D9    JMP     $ D9EF
0311—   20  B1  00    JSR     $ 00B1
0314—   48            PHA
0315—   A5  A1        LDA     $ A1
0317—   85  06        STA     $ 06
0319—   C6  06        DEC     $ 06
031B    D0  04        BNE     $ 0321
031D—   68            PLA
031E—   4C  33  03    JMP     $ 0333
0321—   20  B1  00    JSR     $ 00B1
0324—   20  7B  DD    JSR     $ DD7B
0327—   20  52  E7    JSR     $ E752
032A—   20  B7  00    JSR     $ 00B7
032D—   C9  2C        CMP     # $ 2C
032F—   F0  E8        BEQ     $ 0319
0331—   68            PLA
0332—   60            RTS
0333—   48            PHA
0334—   20  B1  00    JSR     $ 00B1
0337—   20  7B  DD    JSR     $ DD7B
033A—   20  52  E7    JSR     $ E752
033D—   68            PLA
033E—   C9  B0        CMP     # $ B0
0340—   D0  1A        BNE     $ 035C
0342—   A9  03        LDA     # $ 03
0344—   20  D6  D3    JSR     $ D3D6
0347—   A5  B9        LDA     $ B9
0249—   48            PHA
034A—   A5  B8        LDA     $ B8
034C—   48            PHA
034D—   A5  76        LDA     $ 76
034F—   48            PHA
0350—   A5  75        LDA     $ 75
0352—   48            PHA
0353—   A9  B0        LDA     # $ B0
0355—   48            PHA
3056—   20  41  D9    JSR     $ D941
0359—   4C  D2  D7    JMP     $ D7D2
035C—   C9  AB        CMP     # $ AB
035E—   F0  14        BEQ     $ 0374
0360—   C9  AE        CMP     # $ AE
0362—   F0  03        BEQ     $ 0367
0364—   4C  95  DD    JMP     $ DD95
0367—   20  1A  D6    JSR     $ D61A
036A—   A5  9B        LDA     $ 9B
036C—   A4  9C        LDY     $ 9C
036E—   38            SEC
036F—   E9  01        SBC     # $ 01
0371—   4C  50  D8    JMP     $ D850
0374—   4C  41  D9    JMP     $ D941
```

这段解释程序可利用 BASIC 语言的 & 命令，因此在使用前要将本段程序的首地址送入3F5等单元，即

```
03F5—   4C  00  03    JMP     $ 0300
```

这段解释程序可在监控状态下键入，亦可在汇编状态下键入。对监控及小汇编不很熟悉的读者，可用 BASIC 语言将其输入。程序如下：

```
400   FOR I = 1 TO 119:READ A
410   POKE 767+I,A:NEXT
420   POKE 1013,76:POKE 1014,0
430   POKE 1015,3:END
440   DATA 201,180,208,47,32,177,0,32
450   DATA 248,230,201,175,240,3,76,239
460   DATA 217,32,177,0,72,165,161,133
470   DATA 6,198,6,208,4,104,76,51
480   DATA 3,32,177,0,32,123,221,32
490   DATA 82,231,32,183,0,201,44,240
500   DATA 232,104,96,72,32,177,0,32
510   DATA 123,221,32,82,231,104,201,176
520   DATA 208,26,169,3,32,214,211,165
530   DATA 185,72,165,184,72,165,118,72
540   DATA 165,117,72,169,176,72,32,65
550   DATA 217,76,210,215,201,171,240,20
560   DATA 201,174,240,3,76,149,221,32
570   DATA 26,214,165,155,164,156,56,233
580   DATA 1,76,80,216,76,65,217
```

这段 BASIC 程序最好写在 HELLO 程序中，以便一开机就可获得扩展这些命令的功能。

3. 解释程序的语法及说明

我们附加的这段解释程序使用了 & 命令作为引导符。因此在程序中用到扩展的功能时，相应的语法如下：

& GOTO〈表达式〉
& GOSUB〈表达式〉
& ON〈表达式〉&GOTO〈表达式〉
& ON〈表达式〉&GOSUB〈表达式〉
& RESTORE〈表达式〉

注1. GOTO 和 GOSUB 后面的表达式可以是可表达行号的任何形式的变量、表达式或数值行号，如果采用数值行号则与原来的功能相同。如果变量为小数，那么将截去小数部分；如果变量对应的行号不存在，那么仍给出原错误提示。

注2. &RESTORE 命令后的表达式不可省略，恢复全部数据时仍用原命令或给出第一个 DATA 语句行号；表达式值无对应行号时，将设置该值后面的行。

注3. &ON…&GOTO(GOSUB)语句中的第二个 & 命令如果省略，那么后面只能用数值行号。

C 语言初阶讲座

第六讲　宏指令的用法

李文兵

典型的 C 源程序的编译过程是：用预处理程序处理带 # 号的宏预处理指令（简称宏指令）；进行语法分析，产生目标代码；进行代码优化处理。

本讲，我们介绍 C 宏指令的用法。

1. #define 与 #undef

(1) #define　这条宏指令叫宏定义。它用得比较多，也非常有用。其使用格式如下：

#define 〈标识符〉〈字符串〉

这里出现的标识符也叫宏名，#define 的功能是：用字符串置换宏名。

```
A>type exp6-1.c
    #define message "This is a message"
  main()
    { printf(message);}
A>exp6-1
    This is a message
```

宏名可以是带参数的宏

```
A>type exp6-2.c
    #define dmax(x,y) (x>y)? x:y
    main()
    { int a=3,b=2;
      printf("%d\n",dmax(a,b));}
A>exp6-2
    3
A>type exp6-3.c
    #dedfine begin {
    #dedfine end }
    #dedfine DMAX(x,y) x>y? x:y
    main()
    begin
        doubie x=1000.0,y=200.0;
        long lx=11,ly=21;
        int i;
        printf("d-max=%lf\n",DMAX(x,y));
        printf("l-max=%ld\n",DMAX(lx,ly));
    end
A>exp6-3
d-max=1000.000000
l-max=21
```

宏定义经预处理的结果叫宏展开，练习 6.3 的宏展开程序是

```
main()
{ double x=1000.0,y=200.0;
  long lx=11,ly=21;
  int i;
  printf("d-max=%lf\n",x>y? x:y);
  printf("l-max=%ld\n",lx>ly? lx:ly);
}
```

宏展开程序再经过语法分析，产生目标代码，代码优化，之后，经过链接程序 link 把程序的目标码和集中存放诸如 printf()、scanf()等函数目标码的库链接起来，才使程序变为可执行程序。

带参数的宏的调用只是宏置换，因此，执行速度比较快；而预处理展开后的程序要占存储空间，故带参数的宏要比函数需要较多的存储空间。

宏名还可以嵌套使用，如练习 6.4 所示。

```
A>type exp6-4.c
    #dedfine a 10
    #dedfine b 20
    #dedfine c 30
    #dedfine v a*b*c
    main()
    { printf("staiseki %d\n",v); }
A>exp6-4
staiseki 6000
```

使用宏定义应注意以下几点：

①# 前不能有空格或标识符。

②宏指令不是 C 语句，故其末端不能有分号(;)，否则，分号被看作是字符串的一部分。

③宏名不能用双引号引起来，否则不能置换。

④宏指令一般放在文件开头部分，且只在该文件内有效；出了这文件，该宏指令全失去作用。

(2) #undef　取消宏定义，其用法是

#undef 标识符

其中标识符是由宏定义指令定义过的宏名。预处理程序检测到取消宏定义后，便使对应的宏定义失效。

```
A>type exp6-5.c
    #define PAI 3.14159
    main()
    { float r,l,a,v;
      r=4.0;
      l=2.0*PAI*r;
      #undef PAI
      a=PAI*r*r;
      v=(3.0/4.0)*PAI*r*r*r;
      printf("R=%6.2f\n",r);
      printf("L=%f\n A=%f\n V=%f\n",l,a,v);
      }
```

从宏展开可以看出，在取消宏定义指令 #undef PAI 后的宏名 PAI 不能展开。因为预处理程序遇到这条宏指令后，便使宏定义 #define PAI 3.14159 失效。

2. #include

这条宏指令叫宏包含，其用法有两种：

#include "文件名"

#include 〈文件名〉

遇到这两条指令之一，预处理程序就把文件名所指定的文件引进到本指令所在的源文件中，即用文件名所指定的文件内容置换#include 指令行。

这两种宏包含指令是有区别的。遇到前者，预处理程序首先检索当前的文件目录有无该文件，若没有检索到，再检索 C 编译系统中的目录。遇到后者，预处理程序只检索 C 编译系统的目录。宏包含的实例如下：

```
A>type u.h
#define begin{
#define end }
#define DMAX(x,y) x>y? x:y
C>type a:exp6-6.c
    #include "a:u.h"
    main()
    begin
        double x=1000.0,y=200.0;
        long lx=11,ly=21;
        int i;
        printf("d-max=%lf\n",DMAX(x,y));
        printf("l-max=%ld\n",DMAX(lx,ly));
    end
A>exp6-6
d-max=1000.000000
l-max=21
```

我们可以事先编好具有各种功能的标题文件(扩展名为.h 的文件)或其它文件，然后，再用#include 把它们包含到当前文件里使用。

stdio.h 是用得最多的标题文件，内容大致包括：符号定义、标准输入设备的定义及常数、函数、类型的定义等。在根目录下可用如下命令查看。

```
C>CD TC\include
C>type stdio.h
```

标准输入输出函数 getchar() 和 putchar() 就是在该文件里定义的，故凡是要使用这两个函数的文件，其开头部分必须有宏包含：

```
#include <stdio.h>
```

或

```
#include "stdio.h"
```

getchar() 的功能是从操作系统确定的标准输入设备输入一个字符。其特点是：没有参数；类型为整型；返回值为字符的 ASCII 码。用法：

```
int c;
c=getchar();
```

或

```
char c;
c=getchar();
```

putchar()的功能是从操作系统所确定的标准输出设备上输出一个字符，其特点是：必须把输出字符作为参数给出；其类型为整数；没有返回值。

下面是用这两个函数输入输出的最简单的例子：

```
A>type exp6-7.c
    #include "stdio.h"
    main()
    {   int c;
        c=getchar();
        putchar(c);
    }
A>exp6-7
3 (仿效返回)
3 (执行结果)
A>exp6-7
w
w
```

下面是使用这两个函数连续输入输出的程序：

```
A>type exp6-8.c
    #include "stdio.h"
    main()
    { for(;;)
      putchar(getchar());
    }
A>exp6-8
abc. (仿效返回)
abc. (执行结果)
```

3. 条件编译

这类宏指令有：#if、#ifdef、#ifndef、#endif、#else、#elif。它们按一定方式组合，构成条件编译程序结构。下面介绍这些程序结构。

(1) #if～#endif 结构　这种结构的用法如下：

```
#if 常量表达式
    语句;
#endif
```

如果常量表达式的值不为 0，则 Turbo C 编译该程序模块；否则，不编译该程序模块。如练习 6.9 所示。

```
C>type exp6-9.c
#define MAX 10
#if MAX==0
#define MAX 5
#endif
main()
    { int i=6;
  do  {
        printf(" i=%d\n",i++);
      } while (i<MAX);
    }
```

由于 MAX 先用#define 定义为 10，其条件编译模块不被编译，故 MAX 仍保持为 10，执行结果如下：

```
C>exp6-9
i=6
i=7
i=8
i=9
```

如把#define MAX 10 删去，则如练习 6.10。

```
A>type exp6-10.c
#if MAX==0
#define MAX 5
#endif
main()
    { int i=2;
      do { printf("i=%d\n",i++);
```

```
        } while(i<MAX);
    }
A>exp6-10
i=2
i=3
i=4
```

(2) #if～#else～#endif 用法如下：

```
 #if  常量表达式
        语句 1;
 #else
        语句 2;
 #endif
```

如果常量表达式的值不为 0，则编译语句 1；否则，编译语句 2，如练习 6.11 所示。

```
A>type exp6-11.c
#define MAX 100
#if MAX>10
  #define MAX 10
#else
  #define MAX 5
#endif
main()
    { int i=6;
      do { prinft(" i=%d\n",i++};
        } while(i<MAX);
    }
```

因为设定 MAX=100，根据 #if～#else～#endif 的功能，使 MAX=10，故程序执行结果如下

```
A>exp6-11.exe
i=6
i=7
i=8
i=9
```

(3) #if～#elif～#endif 用法如下：

```
 #if  常量表达式 1
        语句 1;
 #elif 常量表达式 2
        语句 2;
 #endif
```

这里，#elif 可进行多次重复，如练习 6.12。

```
A>type exp6-12.c
#define MAX 4
#if MAX >10
#define MAX 10
#elif MAX >3
#define MAX 3
#elif MAX >2
#define MAX 2
#endif
main()
    { int i=0;
      do{ printf(" i=%d\n",i++);
        } while(i<MAX);
    }
A>exp6-12
i=0
i=1
i=2
```

(4) #ifdef、#ifndef　这两条宏指令是用来识别宏名是否被定义过了。它们均可取代 #if，构成条件编译结构。对于 #ifdef，当其后跟的标识符被 #define 定义过时为真；对于 #ifndef，只有其后跟的标识符没有被 #define 定义过才为真。如练习 6.13 和 6.14；

```
A>type exp6-13.c
#define MAX 4
main()
    { #ifdef MAX
        printf("define MAX\n");
      #else
        printf("not defined MAX\n");
      #endif
    }
A>exp6-13
define MAX
A>type exp6-14.c
#define MAX 4
main()
    { #ifndef MAX
        printf("define MAX\n");
      #else
        printf("not defined MAX\n");
      #endif
    }
A>exp6-14
not defined MAX
```

4. #line

这条宏指令可用来修改 Turbo C 中所定义的--LINE--和--FILE--等宏名的内容，用法如下：

```
    #line  行号  "文件名"
```

注意该行号为源程序中当前行号，文件名为当前编译的文件名。

```
A>type exp6-15.c
main()
    { printf("line=%d\n",--LINE--);
      {
       #line 1000 "sub-block"
printf("line=%d %s\n",--LINE--,--FILE--);
      }
    }
A>exp6-15
line=2
line=5  A:\EXP6-15.C
```

从执行结果可以看出，尽管我们在该程序 4 行中所指定的文件并非当前编译的文件 exp6-15.c，但编译程序还是把--FILE--和--LINE--分别按当前编

译文件名和源程序中当前行号进行处理的。再做一个练习。

```
C>type exp6-16.c
#line 100
main()
    { printf("LINE=%d\n",--LINE--);
    }
C>exp6-16
LINE=3
```

5. #pragma　　用法：

#pragma　名字

在 Turbo C 中有两个名字可使用，即：

(1) #pragma inline　该宏指令对于 Turbo C 来说，表示程序内含有 inline 汇编码，等效于指定编译任选项-B。

(2) #pragma warn　该宏指令与许可编译任选项-wxxx 所指定的 xxx 相同，但由 #pragma warn 所指定的 xxx 优先。例如，若编译如下程序，则会因变量 i 不确定而发生警告。

```
C>type exp6-17.c
/* pragma warn -def */
main()
{   int i;
    if (i>=100) printf("i=%d\n",i);
}
```

编译如下程序则不会出现警告。

```
C>type exp6-18.c
#pragma warn -def
main()
{   int i;
    if(i>=100) printf("i=%d\n",i);
}
```

6. #error

这条宏指令的用法是：

#error　错误信息

在编译过程中，一旦遇到这条宏指令，便会中止编译，读该宏指令，显示出错误信息后而停止。

```
C>type exp6-19.c
main()
{ printf("Before #error");
  #error error-message
    printf("After #error");
}
```

信息窗口显示内容如下：

```
Compiling C:\EXP6-19.C:
Error C:\EXP6-18.C  3: Error directive:
    error-message in function main
```

7. 内部宏名

ANSI 标准规定了 5 个内部宏名，即：--LINE--，--FILE--，--DATE--，--TIME--，--STDC--。

Turbo C 增补了 10 个内部宏名，即：--CDECL--，--COMPACT--，--HUGE--，--LARGE--，--MEDIUM--，--MSDOS--，--PASCAL--，--SMALL--，--TINY--，--TURBOC--。

下面说明各内部宏名的定义值。

--LINE--存放当前编译的行号

--FILE--存放当前编译的文件名

--DATE--存放源文件被编译为目标文件的日期，存放形式为月/日/年。

--TIME--存放源文件被转换为目标文件的开始时间，以时:分:秒形式存放。

--STDC--若存有 10 进制常数 1，则代表使用的环境工具是标准工具程序；否则，为非标准工具程序。

--CDECL--使用标准 C 调用协议，即没有使用 pascal 选择项时，--CDECL--有定义；否则无定义。

--TINY--、--SMALL--、--COMPACT--、--MEDIUM--、--LARGE--、--HUGE--这 6 个内部宏名是表示存储模式的，因此，它们当中只有对应于编译中定义的存储模式的一个才被定义。

--MSDOS--使用 MS-DOS 版 Turbo C 时被定义为 1。

--PASCAL--使用 pascal 调用协议时，--PASCAL--被定义；否则不被定义。

--TURBOC--存有使用的 Turbo C 的版本号。

```
A>type exp6-20.c
main()
{    printf("%s  %d  %s  %s\n",--FILE--,
              --LINE--,--DATE--,--TIME--);
     printf("Program being compiled using the ");
#ifdef --TINY--
     prinft("tiny model");
#endif
#ifdef --SMALL--
     prinft("small model");
#endif
#ifdef --COMPACT--
     prinft("compact model");
#endif
#ifdef --MEDIUM--
     prinft("medium model");
#endif
#ifdef --LARGE--
     prinft("large model");
#endif
#ifdef --HUGE--
     prinft("huge model");
#endif
     prinft("\n");
}
C)exp6-19
EXP6-19.C  3  Jan  01  1980  00:16:20
Program being compiled using the small model
```

学用单片机

单片机旅馆客房门卫系统

杭州旅游职业学校　傅士淦

旅馆饭店的客房安全服务素为管理者所重视，也为住宿旅客所关切。客房服务员为搞好安全服务付出了辛勤劳动。但有时旅客出入客房频繁，稍为怠慢即被认为有服务不周之嫌；倘使偶有疏忽，客房财物短少，即造成企业和个人的损失。因此，客房安全管理实为一项繁琐又重要的工作。

现用单片微电脑设计控制的门卫系统，可以对客房实行有效的安全管理。它忠于职守，不徇私情，能可靠地代替人们的劳动。

旅馆客房门卫系统功能：

旅客进入客房时，需在系统键盘上键入六位数字密码。经电脑核实无误，指示灯亮，房门自动打开；十五秒钟后房门自动关闭，指示灯灭。如有外力阻碍开门或关门，系统将发讯号报警。出室者只须按室内开门键，门即自动打开十五秒钟并关闭。内存密码 100 组，可供管理者调换选用。

一、系统硬件介绍：

系统硬件电原理图如图 1。

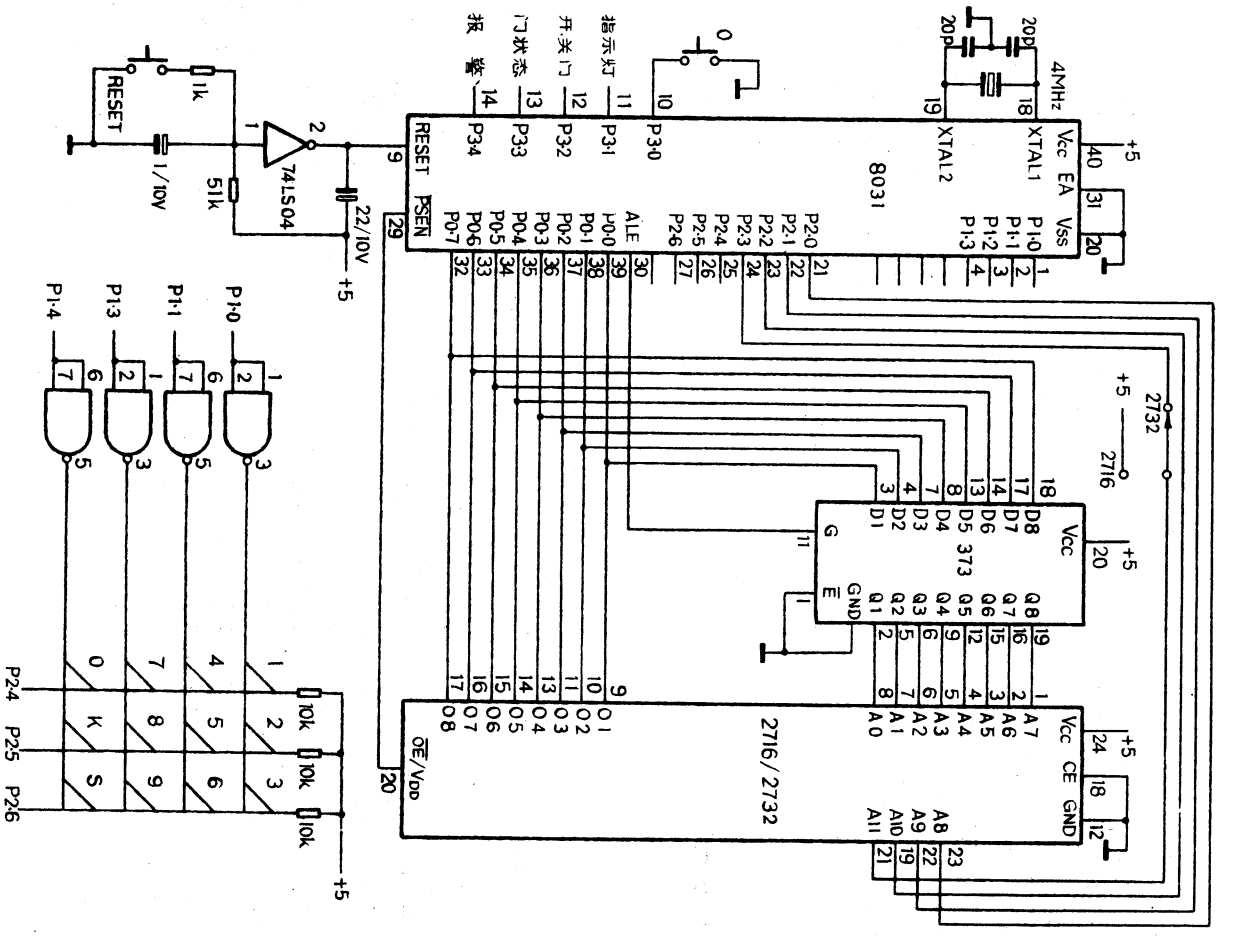


图 1

本系统由单片机 MCS－51 系列 8031 单片机作微处理机。单片机的 P0 口作地址和数据总线输入、输出口 P2 口低四位作外储器地址高四位输出口，P3 口作外设控制或检测用。P0 口 0,1,2,3 位用作位扫描，P2 口 4,5,6 位用作键输入口。ALE 打入外部地锁存器 74LS373，锁存器输出作为地址总线低八位。由$\overline{PSEN}$信号选通外存储器 EPROM 2716 或 2732，外存储器存放系统程序和密码数据。

键盘用矩阵联接编码方式。键盘上设置十三个按键。其中一个复位键，十个数字键，二个命令键。

二、系统软件介绍：

系统软件流程框图如图 2。

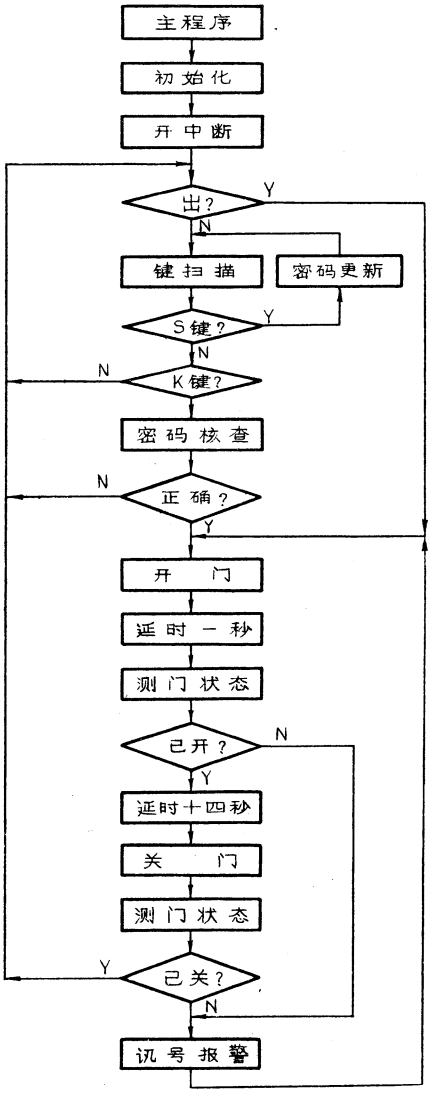
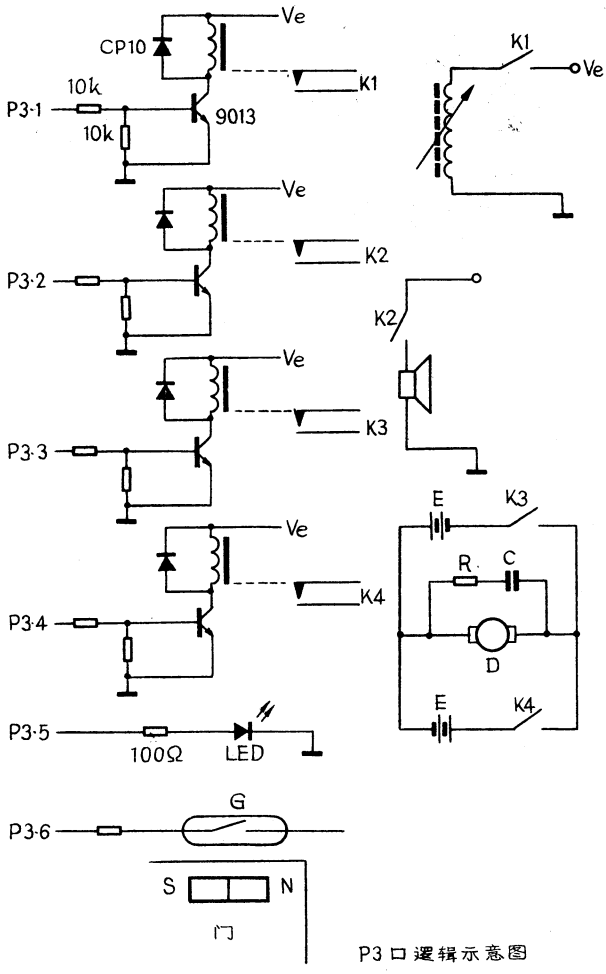


图 2

系统程序由主程序、键盘扫描子程序密码输入子程序密码核查子程序及中断服务子程序组成。



P3 口逻辑示意图

利用单片机机内定时器，设置 100SM 溢出中断一次。中断服务子程序对中断次数计数，并把计数值置于 R6 中，用来延时计时。

三、系统工作说明：

复位键 RESET 可中断程序运行，使程序返回入口地址 0000H，也可用此键中止报警。

用户首先选定内存密码，可用命令键 S。把已写入 EPROM 的 100 组密码，选择任一组作当前密码用。在键盘上输入 0～99 数字，按下 S 键，便确定当前密码组的首址。此键由管理人员专用。（装置此键不必外露）。

命令键 K 是执行键，当六位数字密码输入后，按下 K 键，系统就进行核查工作。若输入密码数字正确无误，置 P3.1、P3.3、P3.5 高电平，执行启锁、开门、亮指示灯动作。K1 吸合，接通电磁铁 M，启开锁舌。K3 吸合，直流电机 D 正转，经传动机构变速，打开门户。延时一秒钟，置 P3.3 低电平，关电机。此时检测 P3.6 位，若是高电平，因舌簧管 G 仍被装在门上的磁铁吸合，门未被打开。置 P3.2 高电平，发讯号报警。若检测到 P3.6 位低电平，则门已被打开。延时十五秒，置 P3.4 高电平，K4 吸合，直流电机反转驱动门户关闭。延时

一秒再次检测门的状态，若门已关，置 P3.1、P3.4、P3.5 低电平，执行以下工作；释放电磁铁使锁舌合上、关电机、关指示灯。若门未关上则发信号报警，直至外界干预。

系统另有一个出门键 OUT，装于客房内。按下此键，P3.0 位接地。系统检测到 P3.0 低电平状态，即转入自动开门、关门等动作。

四、核查控制程序清单：

```
TEXT:   MOV R0,#10H          ;输入密码存放首址
        MOV DPTR,#0300H      ;内存密码组存放首址
        MOV R2,#06H          ;核查六次
        MOV A,20H            ;密码组代码
TX0:    MOVC A,@A+DPTR       ;取密码
        XRL A,@R0            ;核查
        JZ TX1               ;相符继续核查
        SJMP MAIN            ;不相符返回循环
TX1:    INC R0               ;
        INC DPTR             ;
        DJNZ R2,TX0          ;
TX2:    SETB P3.1            ;开锁
        SETB P3.3            ;开门
        STEB P3.5            ;亮指示灯
        CLR A                ;
        MOV R6,A             ;
TX3:    MOV A,R6             ;延时一秒钟
        CJNE A,#0AH,TX3      ;
        CLR P3.3             ;关电机
        JNB P3.6 TX4         ;检测门状态
        SJMP TX8             ;未开报警
TX4:    MOV A,R6             ;门已开延时十五秒
        CJNE A, #0A0H,TX4    ;
        SETB P3.4            ;关门
TX5:    MOV A,R6             ;延时一秒
        CJNE A,#0AAH,TX5     ;
        CLR P3.4             ;关电视
TX6:    JB P3.6,TX7          ;测门状态
        SJMP TX8             ;未关报警
TX7:    CLR P3.1             ;门已关上锁
        CLR P3.5             ;熄指示灯
        SJMP MAIN            ;返回循环
TX8:    SETB P3.2            ;报警讯号
        SJMP TX8
```

Addrs	Opcode	Line		Source Line
		1		ORG 0000H
0000H	010D	2		AJMP MAIN
		3		ORG 000BH
000BH	2112	4		AJMP JSW
000DH	C2B1	5	MAIN:	CLR P3.1
000FH	C2B1	6		CLR P3.1
0011H	C2B3	7		CLR P3.3
0013H	C2B4	8		CLR P3.4
0015H	C2B5	9		CLR P3.5
0017H	C2B6	10		CLR P3.6
0019H	758160	11		MOV SP,#60H
001CH	758901	12		MOV TMOD,#01H
001FH	E4	13		CLR A
0020H	FE	14		MOV R6,A
0021H	F520	15		MOV 20H,A
0023H	758AB0	16		MOV TL0,#0B0H
0026H	758C3C	17		MOV TH0,#3CH
0029H	753010	18		MOV 30H,#10H
002CH	D28C	19		SETB TR0
002EH	20B002	20	MP0:	JB P3.0,MP1
0031H	0167	21		AJMP TXZ
0033H	1195	22	MP1:	ACALL KENY
0035H	FC	23		MOV R4,A
0036H	740B	24		MOV A,#0BH
0038H	9C	25		SUBB A,R4
0039H	4002	26		JC MP2
003BH	8004	27		SJMP MP3
003DH	11CA	28	MP2:	ACALL INP
003FH	80ED	29		SJMP MP0
0041H	1195	30	MP3:	ACALL KENY
0043H	B40C02	31		CJNE A,#0CH,MP4
0046H	800D	32		SJMP TEXT
0048H	1195	33	MP4:	ACALL KENY
004AH	B41406	34		CJNE A,#14H,MP5
004DH	7811	35		MOV R0,#11H
004FH	7920	36		MOV R1,#20H
0051H	11DC	37		ACALL JUB
0053H	012E	38	MP5:	AJMP MP0
0055H	11E8	39	TEXT:	ACALL JPC
0057H	7811	40	:	MOV R0,#11H
0059H	7A06	41		MOV R2,#06H
005BH	E520	42		MOV A,20H
005DH	93	43	TX0:	MOVC A,@A+DPTR
005EH	66	44		XRL A,@R0
005FH	6002	45		JZ TX1
0061H	80CB	46		SJMP MP0
0063H	08	47	TX1:	INC R0
0064H	A3	48		INC DPTR
0065H	DAF6	49		DJNZ R2,TX0
0067H	D2B1	50	TX2:	SETB P3.1
0069H	D2B3	51		SETB P3.3
006BH	D2B5	52		SETB P3.5
006DH	E4	53		CLR A
006EH	FE	54		MOV R6,A
006FH	EE	55	TX3:	MOV A,R6
0070H	B40AFC	56		CJNE A,#0AH,TX3
0073H	C2B3	57		CLR P3.3
0075H	20B602	58		JB P3.6,TX4
0078H	8017	59		SJMP TX8
007AH	EE	60	TX4:	MOV A,R6
007BH	B4A0FC	61		CJNE A,#0A0H,TX4
007EH	D2B4	62		SETB P3.4
0080H	EE	63	TX5:	MOV A,R6
0081H	B4AAFC	64		CJNE A,#0AAH,TX5
0084H	C2B4	65		CLR P3.4
0086H	20B602	66	TX6:	JB P3.6,TX7

```
0089H  8006    67          SJMP TX8
008BH  C2B1    68   TX7:   CLR P3.1
008DH  C2B5    69          CLR P3.5
008FH  809D    70          SJMP MP0
0091H  D2B2    71   TX8:   SETB P3.2
0093H  80FC    72          SJMP TX8
               73
0095H  7A08    74   KENY:  MOV R2,#08H
0097H  7B80    75          MOV R3,#80H
0099H  EB      76   EN0:   MOV A,R3
009AH  F590    77          MOV P1,A
009CH  C3      78          CLR C
009DH  13      79          RRC A
009EH  FB      80          MOV R3,A
009FH  E5A0    81          MOV A,P2
00A1H  F4      82          CPL A
00A2H  5470    83          ANL A,#70H
00A4H  7002    84          JNZ EN1
00A6H  DAF1    85          DJNZ R2,EN0
00A8H  03      86   EN1:   RRA
00A9H  03      87          RRA
00AAH  541F    88          ANL A,#1FH
00ACH  4A      89          ORL A,R2
00ADH  FA      90          MOV R2,A
00AEH  9000BE  91          MOV DPTR,EN5
00B1H  7B0C    92          MOV R3,#0CH
00B3H  EB      93   EN2:   MOV A,R3
00B4H  93      94          MOVC A,@A+DPTR
00B5H  B50A02  95          CJNE A,0AH,EN4
00B8H  EB      96   EN3:   MOV A,R3
00B9H  22      97          RET
00BAH  DBF7    98   EN4:   DJNZ R3,EN2
00BCH  80FA    99          SJMB EN3
               100  EN5:   DB #08H
               101         DB #05H
               102         DB #09H
               103         DB #11H
               104         DB #06H
               105         DB #0AH
               106         DB #12H
               107         DB #07H
               108         DB #0BH
               109         DB #13H
               110         DB #0CH
               111         DB #14H
               112
00CAH  7830    113  INP:   MOV R0,#30H
00CCH  E6      114         MOV A,@R0
00CDH  F9      115         MOV R1,A
00CEH  EC      116         MOV A,R4
00CFH  F7      117         MOV @R1,A
00D0H  09      118         INC R1
00D1H  E9      119         MOV A,R1
00D2H  F6      120         MOV @R0,A
00D3H  B41602  121         CJNE A,#16H,IN1
00D6H  8003    122         SJMP IN2
00D8H  753016  123  IN1:   MOV 30H,#16H
00DBH  22      124  IN2:   RET
               125
00DCH  E6      126  JUB:   MOV A,@R0
00DDH  540F    127         ANL A,#0FH
00DFH  FB      128         MOV R3,A
00E0H  18      129         DEC R0
00E1H  E6      130         MOV A,@R0
00E2H  C4      131         SWAP A
00E3H  54F0    132         ANL A,#0F0H
00E5H  4B      133         ORL A,R3
00E6H  F9      134         MOV R1,A
00E7H  22      135         RET
               136
00E8H  E520    137  JPC:   MOV A,20H
00EAH  540F    138         ANL A,#0FH
00ECH  FB      139         MOV R3,A
00EDH  EB      140         MOV A,R3
00EEH  A4      141         MUL AB
00EFH  FB      142         MOV R3,A
00F0H  75F010  143         MOV B,#10H
00F3H  C5F0    144         XCH A,B
00F5H  CA      145         XCH A,R2
00F6H  A4      146         MUL AB
00F7H  2A      147         ADD A,R2
00F8H  CB      148         XCH A,R3
00F9H  5320F0  149         ANL 20H,#0F0H
00FCH  7820    150         MOV R0,#20H
00FEH  26      151         ADD A,@R0
00FFH  CB      152         XCH A,R3
0100H  3400    153         ADDC A,#00H
0102H  FA      154         MOV R2,A
0103H  7903    155         MOV R1,#03H
0105H  7F00    156         MOV R7,#00H
0107H  EB      157         MOV A,R3
0108H  2F      158         ADD A,R7
0109H  FD      159         MOV R5 A
010AH  EA      160         MOV A,R2
010BH  39      161         ADDC A,R1
010CH  FC      162         MOV R4,A
010DH  8D82    163         MOV DPL,R5
010FH  8C83    164         MOV DPH,R4
0111H  22      165         RET
               166
0112H  758AB7  167  JSW:   MOV TL0,#0B7H
0115H  758C3C  168         MOV TH0,#03CH
0118H  0E      169         INC R6
0119H  32      170         RETI
               171
               172         ORG 0300H
               173         DB #01H
               174         DB #02H
               175         DB #03H
               176         DB #04H
```

```
177          DB #05H                              182          DB #00H
178          DB #06H          ERROR    0
179          DB #07H
180          DB #08H
181          DB #09H
```

中华学习机汉字处理(CPC)卡使用说明

中华学习机汉字处理卡(以下简称CPC卡)是专为CEC－I型中华学习机设计的五笔字型输入、打印机驱动和汉字全屏幕编辑的多功能卡。插上该卡后,中华学习机在汉字状态下,除可以使用原有的"区位"、"拼音"两种汉字输入方法外,还可使用"五笔"方式输入。同时还可使用CPC卡上的汉字全屏幕编辑软件以及打印驱动功能。CPC卡虽然使中华学习机增加了三项功能,但未改动和减少原中华学习机的任何功能。

CPC卡具体使用方法如下:

安装办法

将中华学习机电源关掉。把CPC卡插入扩充槽内(器件面向用户)。如使用CPC卡上的汉字全屏幕编辑软件,还需在磁盘驱动器中插入一张带有DOS3.30系统的软盘。然后打开显示器和中华学习机的电源即可使用下述功能。

一、打印功能(PR#1):

为方便用户,中华学习机CPC卡在中西文方式下均保留原单一打印卡的全部功能,且操作方法不变。

在西文方式下,当准备好打印机之后,只需直接键入"PR#1"便可接通打印机。此时CPC卡为用户提供了字符打印及高分辨率图形拷贝功能。

※打印西文字符

用于打印程序清单、程序运行结果以及其它信息。

在此方式下,原打印控制符均不变。如PRINT CHR$(20)可使输出的西方字符在正体/斜体之间转换;PRINT CHR$(23)可使输出的大写字母变为小写。其它如加重打印、缩小打印、行间距控制等【ESC】序列控制符均可照常使用。

※打印图形

西文方式下接通打印机后,直接按CTRL－Q(【西文】)键或用BASIC语句PRINT CHR$(17)即可按用户设置的参数打印出高分辨率图形。

参数设置方法如下:

N=P+B+I+S+L

其中:

P=0	不打印
P=1	打印第一页
P=2	打印第二页
P=3	两页并排打印
B=0	直接打印原图
B=4	两页与(AND)运算后打印
B=8	两页或(OR)运算后打印
B=16	两页异或(EOR)运算后打印
I=0	正相打印
I=32	反相打印
S=0	一倍大小打印
S=64	二倍大小打印
L=0	打印全屏图形
L=128	打印一行图形

用户按需要选择各参数并相加后,使用POKE1913,N即可确定打印参数。在打印过程中,可按CTRL－C中断打印。

汉字方式下的打印设置也保持不变:

※设置打印方式

POKE1659,N

N=0为不打印,以屏幕作为输出显示设备。N=1～15为打印,以打印机作为输出设备,所打印字型与N的值有关。共有15种大小字型,N=1最小,N=15最大。

※设置字间距

POKE1687,N

N为所打印的字符及汉字之间的字间距离,N的取值为0～255,以点为单位。

※设置行距

POKE1915,N

N为打印行之间的点距离,N的取值为0～255。

※设置行打印字数

POKE2043,N

N为每行所打印的汉字个数。每一汉字占两个字符位置。N的取值为0～255。

二、五笔字型输入功能(PR#8)

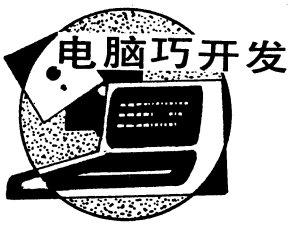
用户键入"PR#8"后,中华学习机自动进入汉字状态,并将屏幕编辑软件调入16K扩充RAM。此时中华学习机可使用五笔字型输入法进行汉字输入,并可使用屏幕编辑软件进行文字编辑。

在汉字状态下,按【F4】键便可进入五笔字型输入方式。此时屏幕底端的状态提示行显示"五笔!"。除"Z"键外的25个小写字母,将作为五笔字型编码输入而显示在屏幕底端的状态提示行上。大写字母和数字符号将作为普通输入而显示在屏幕光标位置上。

CPC卡提供了国标GB－2312(80)16区～87区两级汉字(共计6763个汉字)编码,并有一、二、三级简码功能。例如:"啊"字可用编码:KBSK(原码)、KBS(三级简码)、KB(二级简码)三个编码进行汉字输入;"国"字可用编码:LGYI(原码)、LGY(三级简码)或L(一级简码)三个编码进行汉字输入。同时,该卡还提供了二千六百多个双字词组供用户使用。这是其它中华学习机软、硬五笔系统所不具备的。

当在输入过程中出现重码的情况时,该卡将会在屏幕底端显示出所有的重码字,用户可使用数字键选择所需要的字。如果用户所要选择的为第一个重码字,不必用数字键选择,

(下转47页)



MSX—EPROM 读写/擦除器

刘　园

MSX 微电脑在我国有较多的用户。为了扩充其功能，使之用于开发应用特设计 MSX—EPROM 写入器与擦除器。

1. **硬件结构**　本写入器是利用了具有三个八位端口的可编程接口芯片 8255 和相应的控制程序构成的，其硬件电路如图 1 所示。其中 8255 的数据线分别与 MSX 总线的 D_0～D_7 相连，A_0A_1 作为内部寄存器的选择线也接至 MSX 总线的 A_0A_1，$\overline{IO\ RQ}$和 A_7 经过二极管 D_1D_2 相或后接至 8255 的片选端作为选片信号，这样就有效地避免了和 MSX 机内的端口发生选址冲突。8255 的读写控制线分别接至 MSX 的总线 $\overline{WE}$ 和 $\overline{RD}$ 端，以便 CUP 对其进行相应的操作。RESET 是 8255 的复位信号，高电平有效，故接地。端口 A 接至待写 EPROM 的数据线 D_0～D_7 构成写入器的数据总线。端口 B 接至待写 EPROM 的 A_0～A_7 构成写入器的低八位地址线。端口 C 的 PC_0～PC_6 接至待写 EPROM 的 A_8～A_{14}，构成写入器的高七位地址线，而端口 C 的 PC_7 经过 BG_1、BG_2 产生相应的编程脉冲加入 V_{PP} 端，其中 D_4 是为了保证读片时 V_{PP} 为高电平而设置的。

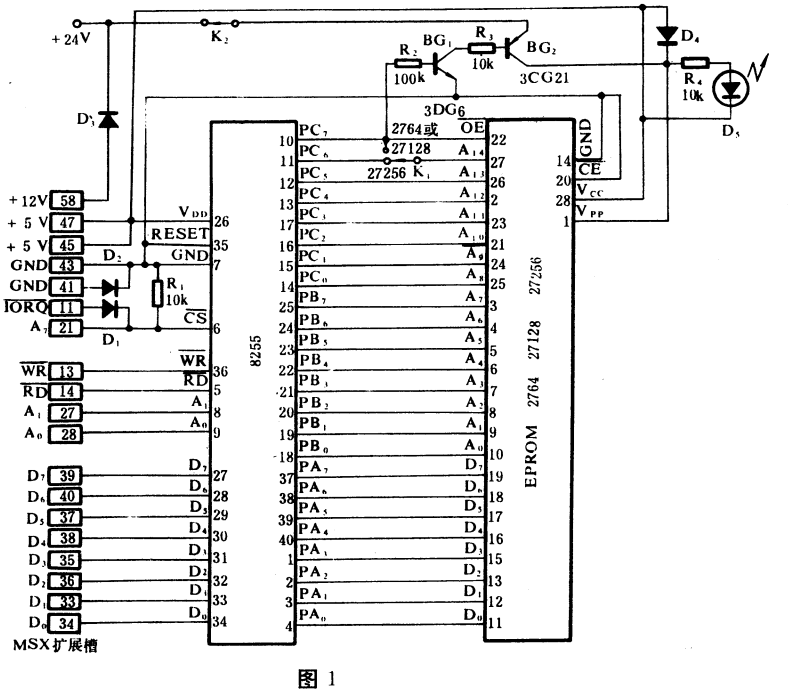


图 1

D5 为发光二极管，当对 EPROM 进行写操作时发光，用来指示编程情况，D_3 为芯片所用编程电压不同而设计。当要写入的芯片为 12V 编程电压时无需外加编程电压，此时 12V 电压由 MSX 总线提供。当要写入的芯片为 24V 编程电压时，需在 24V 端加上电压，此时机内 12V 电压因 D_3 反编被切断，24V 将做为编程电压提供给芯片。开关 K_2 用来接通或断开编程电压以防误操作而带来不必要的麻烦。K_1 用来选择 27256，27128 或 2764 图 1 中所置位置的读写 27256，这样本写入器就可完成对 2764，27128，27256 三种不同的大容量 EPROM 进行读写和校验。

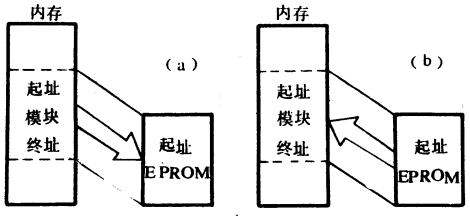


图 2

2. **软件**　本写入器是在磁带版《监控汇编》的支持工作，当要对 EPROM 进行读写操作时，首先要 MSX 机用 RUN“CAS：”调入监控汇编，然后用“CMDMON”命令进入监控，用“X”命令在 BC 寄存器对中放入要操作的 EPROM 首址；在 DE 寄存器对中放入模块区首址；在 HL 寄存器对中放入模块区的尾地址，为以下各种操作做好准备。

在 A 寄存器中放入 1H 表示要对 EPROM　进行“写”操作，此时若敲入“GD400”就可把模块中的数据写入到 EPROM 中，如图 2(a)其工作过程见图 3 左边框，首先由 PB_0～PB_7 和 PC_0～PC_6 输出地址信号，把要写入的数据通过 PA_0～PA_7 送到待写的 EPROM，然后通过 PC_7 输出约 2ms 左右的编程控制脉中，一个单元即完成了写操作，修正指

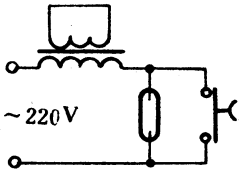


图 4

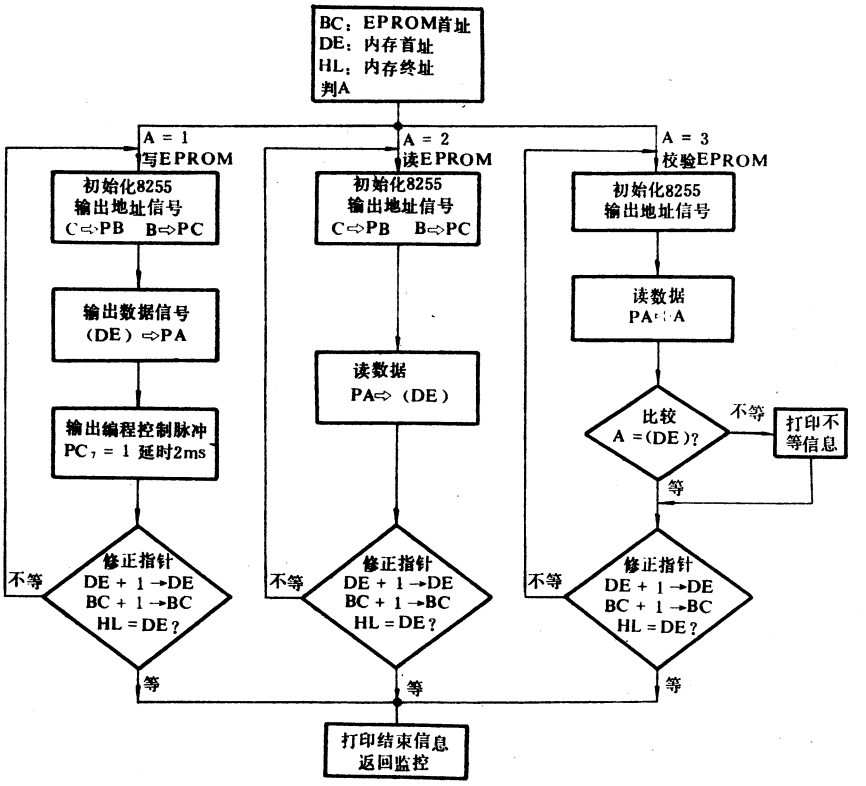


图 3

针后循环上述操作，直至把所有的数据写完。

在 A 寄存器中放入 2H 表示要对 EPROM 进行读操作，此时若敲入“GD400”就可把 EPROM 中的数据装入模块如图 2(b)。工作过程参见图 3 中框，也是先通过 PB_0～PB_7 和 PC_0～PC_6 发出地址信号而后从 PA_0～PA_7 上读取 EPROM 的数据放入指定的模块单元中，修正指针后循环读到指定的长度。

在 A 寄存器中放入 3H 表示要对 EPROM 进行校验。此时若敲“GD400”即检查 EPROM 中的内容和指定模块中的内容是否完全相等，当有不等内容时将打印出不等单元的地址和该单元的内容，如此反复循环到结束。工作过程见图 3 右框，和读出 EPROM 时大致相同，只是读出的数据不是放入指定的单元。而是和该单元进行比较后做判别处理。

以上所说的模块是指内存的某一区域。写入器所用软件主要由 ADDRES，V_{PP}，ERROR 和 LDINC 四个子程序构成，它们的功能分别是输出地址信号；产生编程脉冲控制信号；打印出错单元地址及该单元内容；修改指针判结束。

3. 擦除器 实际上就是一个普通的日光灯电路，电原理如图 4，考虑到体积和价格等因素没有加入时间控制，但完全可满足业余条件下擦除 EPROM。其中镇流器可用 XD7 信号灯的变压器将其次级短路后代用，紫外管可采用 250W 水银灯泡改制，首先将灯炮外壳轻轻敲破，取出其中的芯子，即可代替紫外管。而后用白铁皮制作一个大小合适，便于放置 EPROM 和紫外管的支架，安好变压器和起动按钮，按图检查无误即可通电试验。正常后可把铁架装入一个铝制小菜盒中，这样一个体积小巧，又很实用的擦除器即告制作完毕。使用时，把要擦除的芯片放在支架上正对紫外管，启动紫外管发光，然后盖上盒盖约 30 分钟左右（不同 EPROM 擦除时间不同）即可把 EPROM 擦除干净。写入芯片时，首先要检查是否擦除干净（即检查所有单元是否为 FFH）然后即可按上述方法进行写入等操作。

有关 MSX—EPROM 读写/擦除器的程序清单，可向编辑部函索，收复印及邮挂费 2.00 元。

维修经验谈

任天堂系列游戏机检修实例

林华春

目前各种型号任天堂系列游戏机，虽然其型号繁多，但从电路原理上区别大致可分为三类。以 CPU、PPU 时钟晶振频率为例：①26.601712MHz 单时钟晶振，此类机线路简单，图像质量好，游戏卡兼容性好；②26.601712MHz 和 21.47727MHz；双时钟晶振，此类机线路略为复杂，图象质量较好，游戏卡兼容性稍差；③21.251465MHz，单时钟晶振，此类机线路复杂，增加了制式变换电路，图像质量较好，游戏卡兼容性好，但维修因缺少配件，比较困难。本文结合实际检修例子，用简单仪器、工具（如万用表、逻辑笔、示波器等）加实际经验的检修方法，来判断、排除游戏机电路故障，以此来介绍如何掌握游戏机的检修方法。在检修中还要配备一盘性能良好的游戏卡，作为测试卡使用（文中的故障说明均以第一代“魂斗罗”为例）。

例一、故障现象：有光栅，无图象，无伴音（单时钟晶振）。

分析与检修：联机通电后，电视机荧光屏上有宽屏幕光栅出现，此时可确定 5V 供电电源及 PPU 电路基本正常，再用万用表检测电脑板上各 IC 电路电源端是否有 5V 电压，经检测正常。因为有光栅，所以可确定晶振电路正常，然后用示波器检测 CPU65272 的㉙脚是否有晶振脉中，经检测正常，用万用表测 CPU③脚电压为 4.1V，用手按一下复位按键 RESET，表针回摆后复位，证明复位端正常。断电后测与 CPU 联接的 6116 静态存储器和 74LS139 译码器 IC 电路各脚对地电阻值，（万用表放 RX1K 档），检测后发现 74LS139④脚对地电阻仅为十几欧，与正常值相差较多，断开其外围电路再测量，仍为十几欧，换一片 74LS139 后开机，图像、伴音恢复正常。

例二、故障现象：与例一相同。

分析与检修：用上述方法检测后，没有发现故障点。反复按动 RESET 复位按键后，电视荧光屏上出现一些不规则的图案或颜色变化，此时可确定图像处理 PPU 电路正常，故障是由 CPU 及存储器电路引起，当用逻辑笔测 74LS139 第④脚时，为高电平，而测 6116 的⑱脚 $\overline{CS}$ 片选信号端时，为开路状态，断电后用万用表测两脚引线间电阻值，发现已开路，用细软线将两脚之间联接后开机，一切恢复正常。

例三、故障现象：一台灰白游戏机，无型号，与例一相同。

分析与检修：用上述方法检测后，没有发现异常，更换 CPU 后故障依然存在，此时可确认 CPU 电路正常，仔细观察电视荧光屏上的光栅，发现并不十分清晰，随即更换 PPU6528PIC 电路，换新后开机，故障排除。

例四、故障现象：一台红白游戏机，游戏程序可正常进行，但在图像画面中产生一些多余的图案。

分析与检修：图像的出现是由 PPU 电路执行了 CPU 电路传输的信号后而产生的，从画面上看只有一些多余图案，但主画面并不破碎，可确定是 CPU 数据口电路损坏，换新 CPU87007 后，故障排除。

例五、故障现象：图像破碎，静画面游戏程序能正常进行，（背景画面称为静画面，手柄控制的人物为动画面。）

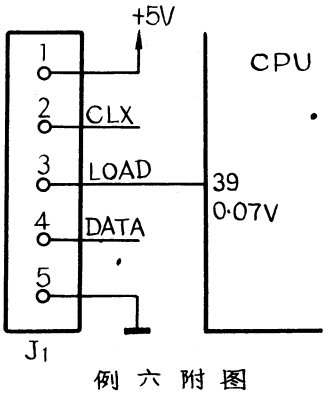
分析与检修：由上例故障分析可知，图像是由 PPU 电路及所属外围电路产生的，故其损坏后，图像即破碎，首先检查游戏机上联接游戏卡的插座，测量时发现有几个脚接触不良，用烙铁重新焊接后，故障排除。除此之外，74LS373 锁存器，视频随机静态存储器 6116 和其联接印刷电路板损坏，都能产生该故障。

例六、故障现象：一台小泰山 301 型游戏机，自动游戏程序正常，画面清晰，但不能启动。

分析与检修：按常规方法检查，用新手柄控制盒替代旧手柄控制盒，仍不能启动，用万用表电压档测 I 号手柄插座各脚电压值，发现第③脚无 0.07V 电压，参见附图。顺其引线测量 CPU 的㊴也无 0.07V 电压，将㊴脚外围电路断环节后测量仍无 0.07V 电压，故确定是 CPU 电路损坏，换新后故障排除。出现上述故障的原因是其它修理人员带电拔插手柄插头。

例七、故障现象：一台红白“小太阳”737 游戏机，故障现象同例六相同。

分析与检修：首先测 I 号手柄插座各脚电压值，发现③脚无 0.70V 电压，但测量 CPU㊴脚却有 0.07V 电压，断电后用万用表测插座至 CPU㊴脚间的联线，发现



例六附图

断线，联接后故障排除。此故障是人为将机器摔在地上把引线摔断。

例八：故障现象：一台任天堂 LT—900A 游戏机，故障现象同例六。

分析检修：正常情况下，把 I 号手柄拔下，插在 II 号手柄插座上，游戏节目能够启动。用此方法处理后，游戏节目能启动，这说明 I 号手柄及电路正常，由此可判断是电脑板上的 74HC368 损坏，换新后故障排除。也可以用上面方法检测 II 号手柄是否正常，此时可在 II 号手柄中加一个启动信号。

例九、故障现象：一台红白游戏机，按 I 号手柄上启动按键 STAR 后，游戏节目不能启动，按其它键无反应，只有按 B 键后图象上选择单双人游戏的符号上下跳动。

分析与检修：为区别是手柄电路还是电脑板上 74HC368 损坏，可先换一个新手柄试验一下，经试验故障如初，可确定是 74HC368 损坏，测试后发现 74HC368 已损坏，换新后用原机手柄试验，故障仍然出现，所以怀疑是手柄有故障，经测量发现手柄电缆中有一根引线断线，联接后故障排除，对于这种故障，要分开检查，否则会造成故障判断困难。

让 PC 机接上家用彩电

CGA—PAL 卡即将问世

本刊接到不少打算购 PC 机的读者来函。寻求 PC 机接彩电的 PAL 卡。鉴于目前国内尚无厂商供货。为了及时满足经济条件并不富裕的购机者的需要。本刊组织多年来一直支持本刊工作的朋友，开发出 CGA—PAL 卡。目前已完成设计工作，有待组织制作。为了较准确的进行投入，请有兴趣购买的读者速来函同我们联系，以便有的放矢地安排制作。。

有关 CGA—PAL 卡的功能，简介如下：

CGA—PAL 适配器是一种复合卡，可直接接 CGA (640×200)彩色显示器；也可以接 PAL 制式或 NTSC 制式的彩色电视机(从电视机的视频输入端口输入；如果彩电上没有视频口，则需加射频调制器)。

CGA—PAL 卡有两种基本工作方式：字母数字方式(A/N)和可寻址所有点的图形方式(APA)。在 A/N 方式中，显示器可以工作于 40 列×25 行方式，以适应低分辨率显示器，也可以工作于 80 列×25 行方式，用来驱动中分辨率监视器。在上述两种方式中，均使用 8×8 的字符框定义字符，小写字符使用 7×7 的字符框定义，比大写字符低一行，大写，小写字符适用于所有的工作方式。每个字符都可以使用 16 个前景颜色和 8 个背景颜色中的一种组合。

CGA—PAL 卡包括 16KB 的存储器，可存储多达 8 屏显示信息。在 80 列×25 行显示方式时，可存储四屏显示信息。全部 16KB 存储器均可直接由 CPC 直接寻址，使软件在管理屏幕时具有极大的灵活性。

在 APA 工作方式时，有两种可用的分辨率：一是中分辨率彩色图形方式(320 图素×200 行)。在中分辨率分式时，每个图素可以在四种颜色中选择一种使用。背景颜色可以是 16 种颜色之一。

在 A/N 方式时，字符由一个 ROM 字符发生器形成，字符发生器中包括了 256 种不同字符的点阵映象。

CGA—PAL 卡也可工作在黑白显示方式，此时能得到高分辨率方式，字符属性包括负象显示，闪耀和增亮功能，因为接口中全部 16KB 存储器都被用来定义图象的亮与不亮状态。

估计售价：350.00 元(注意：请勿汇款，仅来函表示购买意向及要求，以供我们估计市场需要量参考)。

介绍一种理想的“单片机教学实验系统”

“北京市单片机应用技术协会为进一步做好单片机的推广服务工作，近期集中了一批高校教师及有经验的技术干部研制了“单片机教学实验系统”。目前“BJS-51 系列单片机教学实验系统”的基本部分已经完成。“BJS-98 系列单片机教学实验系统”及功能扩展、综合应用部分正进行中。

这种教学实验系统区别于目前市场上一些立足于某仿真器产品开发的教学系统，它完全从教学的实际需要出发，具备实验内容多、覆盖范围宽、结构灵活、性能可靠、价格低廉等优点，是一种理想的教学实验设备。

凡对该产品有兴趣者，请与北京虎坊路十三号北京市单片机应用技术协会盛志纯联系。

邮编：100052

微计算机维修工具
—8088 系统板故障诊断卡(续)

董　珊

五　8253 定时/计数控制器的检查

PC/XT 机系统板中,8253 定时/计数控制器完成三项工作:①为电钟提供一恒定的时间基准,由通道 0 每隔 0.055 秒产生一个输出信号(每秒产生 18.2 次),用以请求时钟中断,8088 响应该中断,用以计算一天的时间。②定时对 RAM 刷新,每隔 15.12us 由通道 1 发出一输出信号,向 8237 的通道 0 提出申请,8237 的通道 0 负责实施刷新工作。③控制扬声器的音频,由通道 3 实现。

在测试卡的标准显示结果中,可以看到诊断程序对 8253 定时器的通道 0,通道 1 进行了测试,其中"8253　CHAN1……OK",是 8253 通道 1 的测试结果,"8253,CHAN0……OK",表示对 8253 通道 0 的检测结果。如若发现错误,8253 的诊断结果会有若干不同的情况,我们大体上可以将这若干情况分为两类:"并发性故障"和非并发性故障。所谓"并发性故障,是指在 8253 显示出故障信息的同时,其它部分也同时报告出错误信息,特别是对 8237 和开关值部分,例如下述情况:

CPU……OK		SW1	1	2	3	4	5	6	7	8
8237	REG…ERROR		1	0	0	1	1	1	1	1
8237	CHANO…OK	SW2	1	2	3	4	5	6	7	8
8253	CHAN1…ERROR		1	1	1	1	1	1	1	1

在 8253 通道 1 有故障时,8237 的寄存器也出现了故障,并且开关值 SW1 的值亦不正确,这是一种典型的"并发性故障"。一般说来,这么多部分因损坏的可能性不大,所以首先检查它们的公用电路,例如 8237 和 8253 在数据传送时都经过的数据缓冲驱动器 U_{15}(LS－245),片选信号的输入控制器 U_{45}(LS－138)和控制传送驱动器 U_{17}(LS－244)。所谓"非并发性故障,是指在显示的信息中,只有 8253 一项出现错误。这种情况,一般是 8253 芯片本身或它的输入、输出控制部分有故障。首先,反复启动几次机器,查看故障信息是否固定不变,若现象固定,可检查 8253 的 9 脚、15 脚、18 脚它们分别是通道 0、1、2 的时钟输入,匀应有脉冲信号,若为恒定电平,可能为 U_{21}损坏。若输入正常,可查 8253 的 21 脚片选信号($\overline{CS}$)是否有负脉冲,若没有,可能是 U_{45}有故障。当 8253 的时钟输入,片选信号都正常以后,可查 U_{73},因为即使 8253 发出了正常信号刷新内存,但 U_{73}不好,诊断程序也无法运行。假如这些匀是好的,可断定 8253 芯片损坏。在替换 8253 时,要选用 8253－5 的型号,这是因为其它型号的 8253 芯片时钟频率不一样,在 PC/XT 机上不能正常工作。

六　8237DMA 控制逻辑的维修

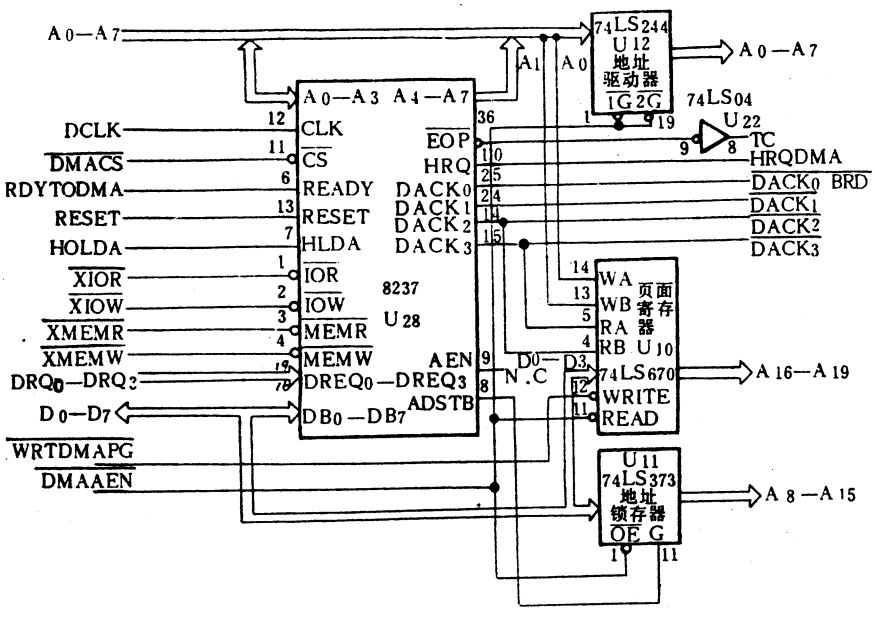
IBM PC/XT 的 DMA 控制逻辑由 DMA 控制电路和 DMA 应答电路两部分组成。DMA 控制电路由 8237DMA 控制器(U_{28}),DMA 页面寄存器(U_{10}),DMA 地址锁存器(U_{11})和 DMA 地址驱动器(U_{12})组成如图六所示,8237 有四个通道,通道 0 用于动态 RAM 刷新;通道 1 未用;通道 2 用于软磁盘的 DMA 操作,通道 3 用于硬磁盘的 DMA 操作。故只有通道 0 用在系统板上所以诊断卡的程序只对通道 0 作了全面的检查。

在发现 8237 报出故障信息时,一般应该首先检查 8237 和输入信号,8237 的 13 脚为 RESET 信号,在开机时应有一个正向脉冲信号,用以消除 8237 的内部寄存器。8237 的 1 脚($\overline{IOR}$)2 脚($\overline{IOW}$)应有脉冲信号。上述三个信号出现故障,不是短路就是有断路。8237 的另一个输入端是 11 脚($\overline{CS}$),这里应为负脉冲信号,否则可查 U_{45}和 U_{17}。8237 的 12 脚为时钟输入信号,它的频率为 4.77MHZ,但它高电平的保持时间为 77ns,若时钟输入为恒定电平或波形不好,8237 出现故障,这时可查一下 U_{74}和 U_{89},若二者都好,可能是延时芯片 TD2 有故障。

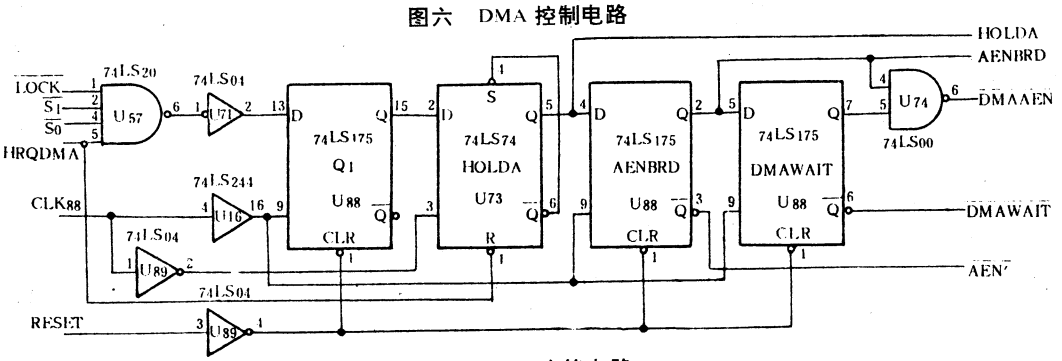
同时还应检查 8237 的 6 脚即 READY 信号,这里应该是脉冲信号,否则应检查 U_{88}的引脚 8 和 U_{70}是否正常。

若上述部分未见异常,应对 DMA 应答电路进行检查,诊断程序是通过通道 0 的响应情况来观查它的,当 8237 的引脚 19 有高电平或脉冲时,8237 的引脚 10 应有高电平或脉冲信号输出,否则就是 8237 坏,若 8237 的 10 脚正常,应检查 8237 的 7 脚是否收到高电平的 HLDA 保持请求响应信号,若 HLDA 是低电平,说明 DMA 应答电路有故障,可查 U_{57}引脚 2、4,它们是 8088 送来的状态信号,应为脉冲,U_{57}引脚 1 为 $\overline{LOCK}$信号应为高电平或脉冲,这时 U_{57-6}应为低电平或脉冲,U_{88}的引脚 13 应为高电平或脉冲,U_{73}的 5 脚应输出高电平或脉冲,所以当 HOLD 信号不正常时,一般是 U_{57}、U_{71},U_{88}、和 U_{73}四个芯片中某一个损坏,如图七所示。

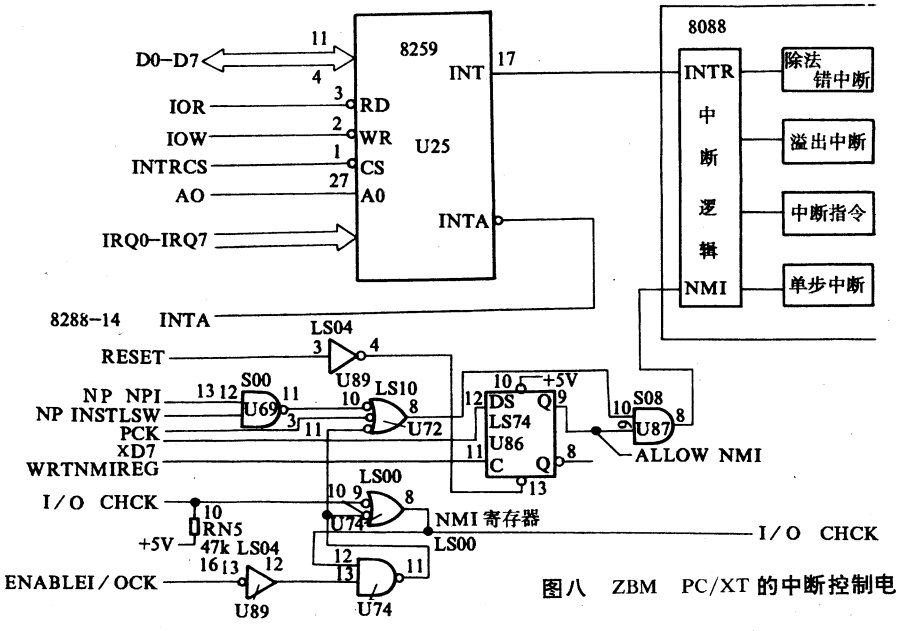
若 8237 的输入信号和 DMA 应答信号都是好的,则 8237 有故障的可能性很大。注意在更换 8237 芯片时,应使用 8237A－5 型号的芯片。



图六　DMA 控制电路



图七　DMA 应答电路



图八　ZBM　PC/XT 的中断控制电路

8237 给出的错信息可能在查找故障点时比较复杂，应仔细的测量上述各点。这里我们举一个实例。从显示屏幕上见到如下信息：

8237　　CHAN0……ERROR

MEMORY　……0000 FF ERROR

8237　　STADUS　bit　ERROR

内存给出的 0 体 FF 不太可能 8 个芯片都坏，8237 的 0 通道错，表示内存刷新电路不正常，进一步说明不是 RAM 本身错，我们根据这些信息发现在 DMA 应答电路中 U_{57-6}脚输出波型不对，低电平在 2V 左右，而 U_{57}的输入匀正常，换掉 U_{57}后，马上恢复正常。

七　对 8259 中断控制部分的检查

8259 中断控制器管理的是可屏蔽中断 INTR，这是由外部设备申请的中断，负责选择优先级最高者向 CPU 发出中断申请。利用诊断卡可以检查 8259 的内部寄存器，及中断请求响应情况。

例如发现错误信息："8259　REG…ERROR"便可知这是测试程序对 8259 的内部寄存器进行检查时，发现有错误，这时应查一下 8259－1 脚(CS)是否为低电平或页脉冲，引脚 27 的地址线 A。也应有脉冲，同时 8259 的 2 脚($\overline{WR}$)和 3 脚($\overline{RD}$)也必须有负脉冲。若 $\overline{CS}$ 恒为高电平，应检查 U_{45}和 U_{17}；若 $\overline{WR}$ 信号为恒高电平，可检查 U_{13}是否有故障。PC/XT 的中断控制电路见图八。

当基本 16K 内存测试通过后，测试卡中的检测程序将设置中断向量，并进行中断响应情况的测试，若发现错误会有如下的信息显示：

"8259　INTRR　FLAG　ERROR"

这时可以检查 8259_{-17}脚 INT 中断信号是否有高电平(或脉冲)输出，若只有低电平则 8259 损坏；若有高电平或脉冲输出，则可检查 8288 的引脚 14 是否有低电平或脉冲信号输出，若无，则一般是 8288 损坏，少数情况也可能是 8088 故障。

在更换 8259 时也要注意选用 8259AC－2 型号的芯片。

在对 8259 进行检查时同样会发生"并发性故障"例如显示：

8237　REG……　ERROR

8237　CHAN0……　ERROR

8253　CHAN1……　ERROR

8259　REG……　ERROR

造成这种现象的原因一般是公用部分故障例如 U_{15}损坏的可能性最大。凡属于并发性故障，应首先在图纸中找到它们的公用电路，仔细测量，效果更好。

八、开关读取不正常的检查

在测试卡测试结果的显示中，有 SW1 和 SW2 二项开关值的检查结果，其中 PC/XT 机只有 SW1，SW2 在 PC 机检测时才有实际意义。

这一部分的检测结果，反映的是 8255 并行接口控制器的工作情况，在 PC/XT 中，8255 共有三个接口，PA 口负责读取键盘输入扫描码；PB 口负责扬声器的控制、奇偶校验控制；PC 口负责 SW1 开关状态的读取。

在开关值读取不正常时，首先应检查开关本身是否有故障，方法是改变开关的设置状况，再观查测试结果，便可判断开关是否损坏。当开关值报出错误信息而且其它部分显示均正常时，例如：

CPU……		OK	SW1	1	2	3	4	5	6	7	8
8237	REG…	OK		1	1	1	0	1	0	0	1
8237	CHAN0…	OK	SW2	1	2	3	4	5	6	7	8
8253	CHAN1…	OK		1	1	1	1	0	0	0	0
8259	REG…	OK									

这种情况下可以检查一下 U_{41}(SL244)，如果开关和 U_{41}都正常则一般是 8255 的 PC 口有故障。

至此，我们可以看到 8088 系统板诊断卡在维修系统板时的作用，它几乎覆盖了系统板上的全部功能，并且在信息报告中将故障按功能分类，对缩小故障围范，提高工作效率有很大帮助，而且你越是对机器的电路熟悉，使用该诊断卡的时间越长，你越感到它确实是一个得力的系统板维修工具。

除此之外，从标准显示信息报告中还可知道，诊断卡还可对打印机并行卡和 RS232 串行卡进行类似的故障诊断，这里不再一一赘述。

磁盘驱动器修复一例

雷　勇

一台驱动器与主机正常连接，开机后驱动器呼呼转动不停，红灯不熄，无法正常引导 DOS，修理步骤如下：

1. 换上一台好驱动器，结果一切正常，这说明问题出在驱动器上。

2. 打开驱动器，发现磁头小车不能顺利平稳地滑动，仔细观察，注意到有两片写保护纸粘住了，以致不能正常工作，用镊子将其小心夹出，驱动器即恢复正常工作。

反思：现在驱动器和写保护片被广泛地使用，但是有些写保护片质量有问题，很容易卡在驱动器里面，这些又是不易被人注意的细节，很容易造成事故。

因此，在这里敬请广大用户注意：在每次使用盘片时应检查写保护片是否牢固，这样能够防止不出意外事故。

初级程序员级水平考试辅导讲座
汉字 dBASE Ⅲ 使用技巧(中)

王路敬

23. 检索命令 FIND、SEEK 的功能上存在的共同缺陷是什么？怎样解决？

无论是 FIND 还是 SEEK 命令，在功能上都不尽完善。它们存在的共同缺陷是：

(1)只能查找首记录，而不能进行连续的顺序查找；

(2)只能在索引文件上进行检索，而不能查找库文件。

利用顺序查找命令 LOCAT 和 CONTINUE 配合起来使用，可以进行连续的顺序查找。这种查找既可以在索引文件内进行查找，也可以在库文件内进行查找。在索引文件内进行查找时，是按着索引文件的记录逻辑顺序进行查找的；而在库文件中则是按照库文件的记录进行查找的；且可以实现连续的查找。

24. 在数据库文件及在索引文件中进行查询时，FIND、SEEK 和 LOCATE、CONTINUE 命令在功能上有何不同？

不同的地方见表 1—1 所示

表 1—1 FIND、SEEK 和 LOCATE、CONTINUE 功能对比

功能 / 命令	能否在库文件中进行查询		能否在索引文件中进行查询						
	能否定域查询	能否连续查询	能否定域查询	能否连续查询	非索引关键字	字符型关键字	数字型关键字	日期型关键字	组合关键字
FIND	不能	不能	不能	不能	不能	能	能	不能	能
SEEK	不能	不能	不能	不能	不能	能	能	能	能
LOCATE CONT—INUE	能	能	能	能	能	能	能	能	能

25. 使用快速检索命令 FIND 应注意什么问题？

检索命令 FIND 只能对索引文件进行各种信息查找，其目的是为了提高信息检索速度。使用该命令时应注意下列几个问题：

(1)查找的字符串无需加引号

用户查找的目标信息可以是字符串，在用 FIND 命令时，字符串可不加界限符‘ ’、“ ”、〔〕，而将字符直接写在 FIND 命令后面。

例如：

· FIND 高级工程师

DISP

(2)FIND 命令只能查找满足条件的首记录

(3)FIND 命令可以查找部分字符串

例如：

· FIND 高级

· DISP

(4)查找内存变量时，必须先用宏代换命令 & 将内存变量替换成字符串之后，才能进行查找。

例如：

· 条件＝“高级工程师”

高级工程师

· FIND & 条件

· DISP

(5)不能查找非索引关键字

如果在操作中 FIND 查找的是没有索引的关键字，dBASE Ⅲ将回答“未发现”的错误信息。

(6)不能查找日期型索引关键字。

26. 检索命令 SEEK 比 FIND 功能上有哪些扩展？

SEEK 是 dBASE Ⅲ 新增添的命令，可以在索引的数据库文件中，检索关键字与给定表达式相一致的首记录。它克服了 FIND 命令查找数字型关键字要经过多次变换，手续繁琐和不能查找日期型索引关键字的二大缺点。

例 1. 当索引关键字为数字型时，查找内存变量，不需进行任何变换，而直接进行查找，十分方便。

· 条件＝45

45

· SEEK 条件

· DISP

同一问题，而 FIND 命令必须先用 STR()函数将数字型内存变量转换成字符型内存变量，然后再用宏代换 & 命令将字符型内存变量转换成字符串。最后再用 FIND 命令进行查找。

· 条件＝45

· 条件＝STR(条件,2)

· FIND & 条件

· DISP

例 2，索引关键字为日期型时，SEEK 命令扩大了 FIND 命令的功能。

表 1—2 SEEK 和 FIND 功能比较一览表

异同 / 命令	字符型索引关键字		数字型索引关键字		日期型索引关键字	
	常量	内存变量	常量	内存变量	常量	内存变量
FIND	直接引用不加引号	要宏代换	直接引用不加引号	先将数字转为字符型再用宏代换	不能	不能
SEEK	不能	不用宏代换	直接引用不加引号	无需宏代换	不能	能够

· 日期＝CTOD(‘09/12/81’)

09/12/81

• SEEK 日期

• DISP

而 FIND 根本没有上述功能。

但是 SEEK 命令不能查找字符串，只能查找内存变量，这点在使用时应予注意。

借此机会小结 SEEK 和 FIND 命令在功能和应用上的异同点。见表 1—2。

27. **索引文件有哪些特点？**

归纳起来有以下几点：

(1)索引文件占用磁盘空间少。

(2)可实现多种不同形式的快速查询。

(3)建立不同数据库之间的关联。

(4)打开一个数据库的同时可以打开七个与它相关的索引文件，录入数据时索引文件同时增加，并按逻辑顺序排列。

(5)当索引文件打开后，删除数据库全部记录时，对应的索引文件记录也一同删除，但建立的逻辑顺序不变。

28. **索引文件和排序文件在使用上有何不同？**

排序也称分类，按照某一关键字段来重新排列数据库中的数据，可递增或递减。使用的命令是 SORT，这个命令的作用是由打开的数据库文件再产生一个新的数据库文件，该文件中的记录包括了原文件中符合范围条件的所有记录，这些记录在新的数据库文件中首先按命令给的第一个关键字段的顺序进行排列，如果第一个关键字段的数据值相同，记录又按第二个关键字段进行排列等等。在使用排序文件时，排序文件并不改变原数据库，而是以原数据库文件为蓝本，按照给出的关键字段及用户指定的排序方法，建立一个新的，按某一关键的顺序排列的数据库。使用几次 SORT 命令就建立几个这样的数据库，所以它要占用许多存储空间。

与排序文件相比，索引文件是根据一个选定的关键字建立起来的，"关键字"是一个字段，或者是几个字段的组合。由于各记录均有与此关键字对应的项，便可根据这些项的值而对各记录按逻辑顺序进行排列。索引文件就是按此逻辑顺序号记录着各关键字项及对应的记录号的，当数据库文件打开时，若对应的索引文件也已打开，则对数据库的记录进行列表输出时，便可根据索引文件的内容将数据库内的记录不按原来的物理顺序，而由关键字规定的逻辑顺序列出来。

由于索引文件是依赖于数据库文件而存在的，所以检索某个索引文件时，应该在打开该索引文件的同时也必须要打开与之相对应的数据库文件。INDEX 命令产生的索引文件，所占空间比原数据库文件要小。

29. **打开索引文件有哪二种方法？二者在使用上有何区别？**

有两种方法打开索引文件，一种是用 UES 命令，另一种是用 SET INDEX TO 命令。

使用 USE 命令打开与库相关的索引文件，一次可以打开一个，也可以打开多个，但最多能打开 7 个索引文件。被打开的 7 个索引文件可以一次打开，也可以分多次打开。

使用 SET INDEX TO 命令同样可以打开索引文件。二者不同的是：USE〈库文件名〉INDEX〈索引文件名〉是将库文件与其相关的索引文件一次打开，而 SET INDEX TO〈索引文件名表〉可先打开库文件，其后当需要时，另行打开索引文件。

30. **删除标志对库文件索引有什么影响？**

在 dBASE Ⅲ 中有一删除标志记入开关的设置命令：

SET DELETED ON/OFF

当为 SET DELETED ON 状态时，删除标志就起作用了，被加删除标志的记录就好象不存在一样，对记录计数、查找、复制等操作都不考虑这些加删除标志的记录。要是对其索引，对加删除标志记录也进行索引，但操作时不考虑这些加了删除标志的记录。

如果断开删除标志开关，则索引文件又恢复其原来的面貌。

所以删除标志开关接通状态，带删除标志的记录被隐蔽起来，当删除标志开关断开之后，这些被隐蔽起来的记录又重新被显露出来。但是不管删除标志是处于接通或断开状态，当对库文件进行索引时，带删除标志的记录和未加删除标志的记录一样，都被索引了。

31. **DELETE，ZAP，DELETE FILE，ERASE 在使用上有什么区别？**

DELETE 是给要删除的记录加删除标记，这样做的目的显然是为了慎重，真正地从库文件中删除 DELETE 指定的这些记录，还必须调用其他命令。

ZAP 是快速清除库文件内容，这种删除是不可恢复的。但库文件结构仍然存在。

DELETE FILE 可以删除任何文件，但不能删除已打开的文件。

而 ERASE 是 dBASE Ⅲ 新增加的一条命令，其功能与 DELETE FILE 一样。但在 dBASE Ⅱ 中 ERASE 是清屏命令。

32. **COPY，COPY TO，COPY FILE 在使用中有什么不同？**

COPY 命令是 PC—DOS 的内部命令。它可在 dBASE Ⅲ 系统下用 RUN 或！命令来调用，进行文件复制，运行速度比在 PC—DOS 系统要快。

COPY TO 是在 dBASE Ⅲ 系统里的命令它可以进行：

(1)库文件结构的拷贝

操作：

• USE〈库文件名〉

• COPY TO〈新库文件名〉STRUTURE

(2)库文件内容的拷贝

操作：

• USE〈库文件名〉

• COPY TO〈新库文件名〉〔范围〕〔FOR〈条件〉〕〔FILEDS〈字段名清单〉〕

(3)生成数据格式文件

操作

• USE〈库文件名〉

• COPY TO〈数据格式文件名〉SDE/DELIMITED

〔范围〕〔FOR〈条件〉〕〔FIELDS〈字段名清单〉〕

而 COPY FILE 对 dBASE Ⅲ 定义的各种文件进行复制。但不能对已打开的文件进行复制。

33. **DISPLAY，DISPLAY MEMORY 和 DISPLAY FILE 在使用上有何不同？**

DISPLAY 是信息显示命令之一。它可以显示库文件结构，也可显示库文件或索引文件的内容。该命令的最大特点具有分页显示功能。即一次只显示一个屏幕画面，叫一页。一页的容量相当于一个屏幕显示的信息，10 个记录。

DISPLAY MEMORY 用来显示已定义的内存变量。DISPLAY FILE 用来显示磁盘上的文件目录。

例如显示当前盘上的库文件

操作：

• DISPLAY FILE

34. DELETE,PACK,ZAP **在功能上的区别是什么**?

DELETE 命令是给记录加删除标志“*”,并不是真正的删除 DETETE 命令指定的记录,这样做的目的显然是为了慎重。这种逻辑上的删除可以用 RECALL 命令还原成不带删除标志的记录。

PACK 命令则是从数据库文件中清除掉所有带删除标志的记录,真正实现了对库文件记录的最后删除,但速度较慢。

ZAP 命令则是快速清除库文件内容,从已打开的库文件中删除所有记录,这种删除是不可恢复的。

35. SUM **和** TOTAL **命令有什么区别**?

SUM 命令可以在当前工作库文件或当前工作索引文件内,对用户指定范围内满足给定条件的记录的数字型字段进行求和运算。

它可以对所有记录的所有数字型字段求和;也可按给定的条件对满足条件的记录的数字型字段求和。在求和时可以利用索引文件或排序文件来加快求和速度。

而 TOTAL 命令是将已打开的库文件的某些数字型字段,按照关键字相同的原则,进行汇总,汇总结果放在一个汇总库内,而汇总库也是一个库文件。它必须是按照关键字进行排序或索引过,换言之,TOTAL 命令执行之前必须按照关键字段建立排序或索引文件。而 SUM 命令没有这个要求。TOTAL 命令的结果是一个库文件,SUM 命令的结果是一个数值。

36. dBASE Ⅲ **的** RUN **命令应用在哪些方面**?

dBASE Ⅲ 的 RUN 命令是在 dBASE Ⅲ 状态下,允许用户在 dBASE Ⅲ 命令文件中执行扩展名为“. EXE”、“. COM”的文件和 CC—DOS 的一些命令,一旦执行结束便返回到 dBASE Ⅲ 状态下。使用该命令,要求用户内存有足够空间,若有效内存空间不够,则外部程序调不到内存中来。一般要求内存空间为 64KB 较好。

该命令格式:

· RUN〈外部文件名〉〈参数〉

或

!〈外部文件名〉〈参数〉(“!”之后不能有空格)

其中〈外部文件名〉指以“. EXE”、“. COM”为扩展名文件或 DOS 的内部、外部命令。〈参数〉指源程序名,扩展名或者子目录名等。

RUN 命令为 dBASE Ⅲ 提供了十分方便的操作系统接口,为用户提供了一个良好的处理环境。该命令可应用于:

(1)在 dBASE Ⅲ 状态下编辑源程序

在 RUN/! 命令下直接调用 EDLIN 或 WS 等编辑程序。使用方法如下:

RUN EDLIN 源程序名·扩展名

这样便进入了行编辑程序状态,即达到了编辑源程序的目的,又避免了数据丢失(因 dBASE Ⅲ 的字处理命令所编辑的文件最大长度 4KB)及用其他方法时来回进入。退出 dBASE Ⅲ 状态所造成的时间上的浪费。

(2)可使用多个目录共享一个 dBASE Ⅲ 版本

如 dBASE Ⅲ 在一子目录 USER1 中,现欲在与 USER1 同层次 USER2 子目录中使用 dBASE Ⅲ,则可用 RUN 命令调用:

· RUN CD\USER1

· RUN CD\USER2

这样便可达到子目录 USER1 与 USER2 共享一个 dBASE Ⅲ 版本的目的。

(3)与外部语言程序连接

可以通过 RUN 命令调用其他高级语言程序。对编译类高级语言,经过链接可得到 · EXE 文件,故可直接调用。而对解释型语言不能直接运行其程序,但是可以先调用该语言的解释程序再运行其有关应用程序。

例如,要调用名为 A1 · BAS 的 GWBASIC 程序,可进行如下操作:

· RUN GWBASIC A1

若要返回到 dBASE Ⅲ 状态下,则在 A1 程序的出口处使用 SYSYTEM 即可。

(4)执行 DOS 的一些命令速度快

在 dBASE Ⅲ 状态下能很方便地使用 DOS 的一些命令,速度比在 DOS 系统下要快得多。

37. **数据库文件用完为什么要关闭**?

如果忘记关闭库文件,一般情况下不会出现什么大问题,因为 dBASE Ⅲ 具有自动关闭库文件的功能。但在某些特殊情况下,不关闭文件,就会出现一些我们意想不到的“怪现象”。

例如,对一个软盘上的库文件进行数据修改,修改完后,未关闭文件就从驱动器抽出,再重新进入 dBASE Ⅲ,结果会发现要修改的记录内容并未变化。其原因就是因为对库文件内容的修改是在计算机内存中进行的,并非修改完即自动转入磁盘,只有在关闭文件之后,才能将修改内容存到磁盘文件中。

再例如,删除操作:

· USE〈库文件名〉

· DELETE ALL FOR RECNO() 〉=4

3 记录 deleted

· PACK

×××记录　　已拷贝

上述操作过程,似乎 3 个记录已被删除,但这种删除也只是在内存中进行的,并未真正地写到磁盘上,因为还未关闭文件。如果我们误认为记录真的被删除了,拿走了磁盘后就退出系统,或换另一张盘,作其他数据操作,我们再打开经过删除操作的该文件,发现记录号〉=4 的记录并未删除掉。

因此,当对库文件内容进行过某些数据操作之后,特别是变动了库文件内容,应及时关闭库文件,以确保这种数据操作的有效性。

38. **关闭数据库常用的方法有哪些**?

在汉字 dBASE Ⅲ 中,输入和修改数据经常使用的命令:APPEND〔BLANK〕,INSERT〔BLANK〕〔BEFORE〕,BROWSE,DELETE,PACK 等。有关这几条命令的执行特点是,每执行一条命令后,才将前一条命令执行的结果真正写入磁盘。

关闭数据库常用的方法有:

(1)在单工作区方式下,打开一个数据库文件,利用打开新文件时,自动关闭以前已打开的数据库的功能,关闭前一个文件。

(2)使用不带参数的 USE 命令,关闭现用数据库文件。

(3)使用 CLOSE DATABASE 命令,关闭所有打开数据库文件、索引文件和格式文件。

(4)使用 CLEAR ALL 命令,关闭所有打开的数据库文件,同时释放所有的内存变量。

(5)使用 QUIT 命令。该命令对数据进行一次清理,该存盘的存盘,所以在退出 dBASE Ⅲ 之前应执行该命令,而不应强行关机。

以上几种办法,可根据不同的需要选择使用。

39. **调试程序过程中要进行其他操作为什么必须首先关闭数据库**?

在 dBASE Ⅲ 状态下调试命令文件的过程中，若将数据盘放在软盘驱动器中，并且未关闭所有的数据库，这时若在该驱动器中换入另外的盘片，执行其他的操作，可能会将该盘上所有的内容全部删除(包括子目录)，而会写上刚刚插入的数据盘上的全部内容，但无真正内容。尤其是运行 DOS 命令时很容易发生这一现象，造成数据丢失。

存在这一现象的主要原因是在对数据库进行操作时，打开数据库后，便可对它进行存入、修改、删除、显示等操作，这一系列工作不是在当前使用的驱动器上进行的，而是在微机的内存中进行的。操作完毕，所有的结果还在内存中，必须在完成关闭数据库操作时，才将该内容存到盘上。否则盘上的内容不变。当然，如果退出 dBASE Ⅲ 状态，微机将自动关闭所在数据，释放所有内存变量，然后回到 DOS 状态，这一过程一般用户是感觉不出来的。

万一发生上述现象，将被破坏的磁盘不要再进行其他操作，想办法进行恢复。

40. **退出 dBASE Ⅲ 系统时为什么一定要执行 QUIT 命令？**

主要原因是由于 dBASE Ⅲ (包括 dBASE Ⅱ)的许多命令并不是立即执行的，尤其是涉及外部设备的命令。比如 CTRL+W 的修改存盘，有时不马上执行操作，即是参照内存储器 RAM 的使用等情况由系统随机管理。dBASE Ⅲ 退出系统命令 QUIT 对系统环境进行一次清理，该存盘的存盘，对原有的信息还要进行一次反存处理。可见，退出 dBASE Ⅲ 时一定要执行 QUIT 命令，切不可为了一时的方便不执行它而退机，或是直接把机器的电源关掉，否则常常会致使一些不该发生的事情发生。有时数据的丢失就是因为忽略了这一点。

41. CLOSE DATABASES **和 USE 命令在功能上有何区别？**

CLOSE DATABASES 命令可以同时关闭所有工作区内的所有库文件。如果现在已经开辟 10 个工作区，要关闭 10 个工作区内的 10 个库文件只要执行：

· CLOSE DATABASES

命令一次就可以了。

而 USE 命令只能关闭当前工作区的库文件或索引文件。所以，要关闭 10 个工作区内的 10 个库文件，则要调用 10 次 USE 命令。

显然，CLOSE DATABASES 要比 USE 效率高。

42. **为什么当执行 QUIT 命令时，有时系统显示：**

"Invalid command · com

Insert command · com disk in drive A

Abort，Retry，Ignore?"**错误信息？**

出现这个问题往往是内存空间小所引起的。对于一般 512KB 的内存，当调入汉字库和打印驱动程序后，进入 dBASE Ⅲ，这时内存空间仅剩 10KB 左右，当运行程序时很可能将 COMMAND · COM 的高端已破坏。因为当开机启动时，首先从系统盘中读出引导记录，然后将 IBMBIOS · COM 和 IBMDOS · COM 两个隐含文件读入内存，进行 DOS 初始化，最后将 COMMAND · COM 读入内存，并赋之于控制权。COMMAND · COM 对本身进行初始化后，便 将其核心部分自举到内存的高地址端，这部分约占内存 12KB 左右，而在内存低地址端仅留一小部分，这部分包括处理 INT 22H，INT 23H，INT 24H 以及检查高端部分是否被覆盖。当用户应用程序调入时，系统并不对 COMMAND · COM 部分进行保护，若应用程序较大，则可能将其 COMMAND · COM 的高端部分覆盖掉。这样的内存分布可使用户得到最大的可能的内存空间。当退出用户应用程序返回到操作系统之前，COMMAND · COM 的驻留部分检查其高端部分是否被覆盖，若已覆盖，则重新从当前盘读出 COMMAND · COM 的高端部分自举到内存的高地址端，然后回到系统提示。同样的道理，从 dBASE Ⅲ 退出时，系统发现若 COMMAND · COM 的高端已被破坏，则重新在当前盘调入 COMMAND · COM 的高端，如果系统启动盘和当前 dBASE Ⅲ 系统盘不是同一版本的操作系统，因此就出现问题里的显示信息。

43. **宏代换函数一般应用在哪些方面？**

(1)代替一条命令

当程序中要重复用到某一条命令时，可将该命令存放到一个内存变量中，在需要时用宏代换引用即可，从而省去了反复书写同一命令的麻烦。例如程序中要反复用到删除某一条记录的命令，则可先将删除命令存入内存变量之中：

· STORE "DELECTE RECORD" TO D

在需要删除第 5 条记录的位置，使用宏代换

· &D 5

在需要删除第 2 条记录的位置，使用宏代换

· &D 2

这等效于使用了 DELECTE RECORD 5 或 DELECTE RECORD 2

(2)代替字段名

当字段名是一个变化的量时，应把字段名存入内存变量之中，在需要的时候用宏代换引用。例如，对于一个图书管理系统，应该在给出书名，或者给出作者名，或者给出编号的情况下都能找出要找的书。那么是否要为每一种情况设计一段程序呢？实际上没有这个必要，只要一段程序就可以解决问题：

ACCEPT "请输入要查找的字段名：" TO F

ACCEPT "请输入要查找的字段内容：" TO S

LOCATE ALL FOR & F=S

若某甲输入的字段名是"书名"，输入的字段内容是"dBASE Ⅲ 综合应用"，则 LOCATE 命令的条件表达式经代换后就变成 FOR 书名="dBASE Ⅲ 综合应用"。若某乙输入的字段名是"作者名"，输入的字段内容是"王秉湖"，则代换后的条件表达式为 FOR 作者="王秉湖"。可见，用宏代换可以将一个表达式变为多个。

(3)代替文件名

当文件名是一个变化的量时，就必须用到宏代换。例如，根据用户需要在程序运行中打开某个数据库文件时，则应该使用如下命令：

· ACCEPT "请输入库文件名"TO FILE

· USE & FILE

(4)将字符型数据转换为数据型数据

& 函数可将一个字符型数字转化为一个数字型量，具有类似 VAL()函数的功能。

例如：· STORE "123.45" TO X

· Y=&X

· ? TYPE("Y")

N

(5)代替字符串

对程序中重复出现的字符串，用 & 函数可减少程序量；人机对话时，& 的使用可增加程序的灵活性。例如：主菜单有 8 个功能模块供用户选择，有两种编写方式：一种用 CASE 语句，8 个模块就必须对应 8 段 CASE 语句；第二种用 & 函数方法，仅用很少语句就能实现上述功能：

```
DO WHILE .T.
PH=" "
@ 5,1 SAY "请送功能模块号:" GET PH
IF PH<="5"
DO 文件名 &PH
EXIT
ELSE
LOOP
ENDIF
ENDDO
```

这种宏代换的方法,当功能模块增加后,程序也无须做什么修改。

(6)形成多维数组

编制程序过程中,使用数组变量能大幅度地缩减程序。尤其是编制任意二维统计模块时没有A(I,J)数组的应用,程序的实现会相当复杂。在dBASEⅢ中,系统用"."区分宏代换变量结束。

例如:

```
A12="HOW ARE YOU?"
I="1"
J="2"
? A&I.&J
```

这里A&I.&J表示了A12变量,用"."分割了I和J两个变量的代换。多维数组也同样可用此法实现。

(7)在FIND命令中使用变量

FIND命令中不能直接使用变量,必须用宏代换来引用。

例如:

```
.STORE "张平" TO NAME
.FIND &NAME
```

通过这种变通即可在命令中使用变量。

尽管宏代换的用途很多,但只要了解它的本质,就可以掌握要领。它的本质就是去掉字符型内存变量的引号。去掉引号的对象,若在命令行的位置上,则被当作一条命令;若在字段名的位置上,则被当作一个字段名;若在文件名的位置上,则被当作一个文件名等等。

宏代换尽管很有用,但它要降低程序的执行速度。这是因为程序在执行包含宏代换的语句行时是先代换而后执行。

44. 使用宏代换应注意哪些问题?

(1)宏代换的字符型变量不允许有前置空格,否则,系统认为语法错误,而与后续的其他变量间,一定要留空格。

例如:

```
.运算符="*"
.STORE 44 &运算符 4 TO A
44*4
```

(2)变量宏代换后,原变量内容是数字,就变成数值;是字符串,就失去了字符串的引号。某些情况下,变量宏代换后仍需要有引号,为解决此矛盾,采用双层引号将变量内容括起来,但双层引号不能相同。

(3)在DO WHILE语句的条件表达式中不能用宏代换。

(4)&函数后必须紧跟一个已定义过的字符型存储变量,而不能是D型,N型和L型。

45. 调用dBASEⅢ的函数时需要注意什么问题?

dBASEⅢ有36种函数,函数的一个重要特性就是函数类型。函数可以被当作一个常数、变量或表达式在程序中进行各种操作运算。但是任何一种数据操作运算都必须遵循着一条原则:即只有相同类型的数据才能进行操作。在函数调用时需注意以下几点:

(1)函数本身的意义。每种函数都完成一定的功能,回送一个确定的数值因此可以把函数看成一种常数,变量或表达式,参加各种数据操作运算。

例如:

```
.? SQRT(4)
```

函数值为2,这里2是函数的回送值。

(2)每种函数都具有一定的数据类型或属性,在对这些函数进行各种数据操作时,一定要注意每种函数的数据类型,否则就要出现操作语法错误。

(3)每种函数都要求一定数据类型的参数,不能出错。

例如:SQRT(B)就是错误的,因为参数"B"不是数字型,而是字符型数据。

(4)如果回送值是一个简单数值,它可以和其他数据进行运算。

例如:2+SQRT(4)

46. 人机会话命令WAIT、ACCEPT、INPUT在用法上有什么不同?

人机会话命令WAIT、ACCEPT、INPUT三个命令在命令文件中的使用不同的地方是:

WAIT:作暂停或输入单个字符,不需要回车,快速响应命令。

例如:.WAIT "打印报表否?(Y/N)" TO W

显示:打印报表否?(Y/N)。于是,根据需要按Y或N键快速响应。

如按Y,则打印。

.? W (变量W显示Y的结果)

ACCEPT:输入多于一个字符时使用。

例如:.ACCEPT "请输入姓名:" TO N

显示:请输入姓名:(可根据具体姓名输入,然后在变量N中存储)

INPUT:除能输入字符和数字型数据外,还可输入逻辑型数据。但字符型需加引号,不如ACCEPT方便。INPUT常用于输入数字型或输入逻辑型数据。

例如:.INPUT "请输入(0~5)选择功能:" TO M

显示:请输入(0~5)选择功能:

47. READ命令和GO命令怎样配合,对指定记录进行屏幕修改?

要修改某个记录,先用GO命令将记录指针移至该记录处,然后再用READ命令对其进行修改。用READ可以修改一条记录中的所有数据项,也可以只修改部分数据项,只要把格式文件设计好,操作是比较容易的。

操作步骤:

(1)打开数据库文件 .USE〈库文件名〉

(2)打开格式文件 .SET FORMAT TO〈格式文件名.FMT〉

(3)将记录定位 .GO RECORD n

(4)用READ命令调用格式文件

.READ

(5)利用全屏幕编辑键进行屏幕编辑修改

48. RETURN,RETURN TO MASTER,QUIT,CANCEL命令有什么区别?

在dBASEⅢ命令文件结束方式的控制中,RETURN为程序逻辑终止命令,当命令文件执行到RETURN命令时,自动地逻辑终止本次程序的运行。

若该程序是由其他主程序调用的，那么该子程序终止后，则自动地返回主程序；如果该程序是由操作员通过键盘输入调用的，那么该子程序终止之后，又自动返回键盘操作工作模式。

RETURN TO MASTER，在这种命令方式下，使得子程序无论在哪一层，当执行到 RETURN 时，都自动返回到最外层的主程序，而不是返回调用该程序的相邻主程序。最外层主程序是 dBASE Ⅲ 模式下调用的第一个主程序。

QUIT 命令是退出 dBASE Ⅲ 系统的命令。QUIT 命令的执行，将关闭 dBASE Ⅲ 系统中的一切文件，其中包括库文件、命令文件等，并自动返回操作系统。

而 CANCEL 命令则是取消程序运行。在程序执行过程中，如果遇到 CANCEL 命令，就自动终止命令文件的执行，返回键盘操作工作模式。

49. SET TALK，SET STEP，SET ECHO，SET DEBUG **等命令在用法上有什么区别？如何配合使用？**

SET TALK ON/OFF 命令是 dBASE Ⅲ 将执行的每一条命令结果显示/不显示在屏幕上，作为人机会话的开关控制。系统默认状态是 ON。

SET STEP ON/OFF 命令是 dBASE Ⅲ 的一条程序调试命令。在程序运行过程中，每执行一条命令都自动停止/不停止，系统默认状态是 OFF。

SET ECHO ON/OFF 命令是 dBASE Ⅲ 程序调试工具之一，其作用是可将运行中的命令文件的命令清单，通过屏幕显示或通过打印机打印出来。ON 状态时显示，OFF 状态时不显示。

而 SET DEBUG ON/OFF 命令也是 dBASE Ⅲ 调试工具之一。它具有自动接通或切断打印机的功能。

SET DEBUG 应该和 SET ECHO、SET STEP、SET TALA 配合使用，使 ECHO、STEP、TALK 回送的命令、信息通过打印机打印机出来。

例如：

· SET DEBUG ON

· SET ECHO ON

此时，ECHO 给出的程序命令清单是送给打印机打印出来的。在此种状态下，当执行某个程序时，就可以对命令文件进行错误诊断。诊断方法是 ECHO 命令在程序运行的同时，给出命令文件中的每条命令，供检查是否出错。因为 DEBUG 是处于 ON 状态，所以 ECHO 送出的程序命令是通过打印机打印出来，而运行的结果，却是通过屏幕显示出来。一旦发现运行的结果有错，就可以从程序命令清单中查找哪些命令可能有错。

50. dBASE Ⅲ **变量有哪几种？它们之间有什么异同点？**

dBASE Ⅲ 有二种变量：内存变量和字段名变量。

内存变量被用来存放程序运行过程中所要用到的各种中间数据。因此内存变量是一种临时工作单元，需要用时可以临时定义，不用时又可以释放掉，每个内存变量都是独立的。

内存变量有四种类型：

(1)字符型内存变量——用来存储字符串。字符变量长度 ⟨=254。

(2)数字型内存变量——用来存储数字。

(3)逻辑型内存变量——只能存储逻辑"真"或逻辑"假"。

(4)日期型内存变量——存储日期。日期总长度是 8 个字节。

字段名变量是构成库文件的最基本数据单元。同一个库文件中的各个数据项在物理意义上是有关联的。字段名变量的数据类型有字符型、数字型、逻辑型、明细型以及日期型。

51. **内存变量的主要用途是什么？**

内存变量是独立存储于数据结构之外的有名数据项，存储于内存变量区。内存变量区中的数据用来控制程序执行过程中各个进程的执行，并且每个数据项都有一个内存变量名指向它。这样一组内存变量就好象一个总是打开的没有任何字段结构的数据库文件。

内存变量最主要的用途有如下一些：

(1)保存经常使用的常数；

(2)保存操作者键入的信息；

(3)作为数据进入数据文件的缓冲区；

(4)保存表达式和函数的结果；

(5)作为计数器的暂存器。

内存变量的建立和赋值是在同一命令内完成的。内存变量的类型就是它们的数据类型，可分为字符型、日期型、逻辑型、数值型。

52. **建立和赋值内存变量的命令有哪些？**

表 1—3 归纳了建立和赋值内存变量的命令。

表 1—3 建立和赋值内存变量的命令

命令	所能处理的内存变量的类型
=(赋值命令)	C,D,L,N①
ACCEPT TO	C
AVERAGE TO	N
COUNT TO	N
INPUT TO	C,D,L,N
RESTORE FROM	C,D,L,N
STORE	C,D,L,N
SUM TO	N
WAIT TO	C

表中 C 表示字符型内存变量，D 表示日期型内存变量，L 表示逻辑型内存变量，N 表示数字型内存变量。

53. **使用内存变量时应注意什么？**

当许多内存变量被来回地建立和恢复时，内存变量区可能会被划成碎片。这样会使实际可用的内存变量区越来越小。更为严重的是，当没有内存变量区而又企图建立内存变量时，就会导致危急情况。如果你的内存变量出了毛病而又找不出原因，就应该考虑到这一点。此时，应把所有活动内存变量保存到一个内存变量文件上去，然后从这个文件上重新调入内存，这样就可以重新构造内存变量区，消灭碎片现象。相应的操作命令及格式如下：

SAVE TO 〈内存变量文件名〉

RESTORE FROM 〈内存变量文件名〉

在汉字 dBASE Ⅲ 中，内存变量分为两大类，一类为局部型变量，一类为全局型变量。在圆点"·"提示符状态下设定的所有变量都被自动定义为全局型变量。在命令文件中设定的变量，如果不加说明，则一律被认定为局部变量。局部变量只在同一级命令文件中有效。一旦遇到 RETURN 命令，局部变量将全部自动释放。因此，对于在程序各级命令中都要使用的变量，都必须事先用 PUBLIC 命令说明为全局变量。

若已把一个变量设定为全局变量，则不能先给定初值，否则系统将给出错误信息。但对一个已经说明为全局型的且已赋过值的变量，可以在另一级命令文件中重新用 PUBLIC 说明之。

新书与软件

TANGO 印刷线路板自动布线软件包

在计算机迅速发展的今天，各种应用软件也越来越多，已成为科研、生产等工作中不可缺少的重要工具。计算机辅助设计和设计自动化是当今电子工程技术必不可少的重要支柱之一，印制线路板图的设计自动化是其中重要的一部份，它彻底解决了传统手工绘图效率低、错误多、集成化低、质量不高的缺点。

印制线路板在电气设备中，从人造卫星的航天电子设备到家庭最小最简单的电器都无法离开印制线路板，它起着各种规模的电子元件和机械结构的双重连接作用，因而印制电路板的正确设计是电子工程中一项极为重要的基础部件，它必须体现优良的电气性能、又具有合理的工艺结构，二者缺一不可，否则达不到预期目的。

TANGO 印制线路板自动布线软件包是美国 ACCEL Technologies Inc 在 1987 年推广的最新一代自动布线软件包，该软件包的设计者具有多年印制线路板图设计的工作经验，并充分吸取了其它印制板图设计 CAD 的优点，因而 TANGO 软件包可以满足众多工程技术人员在实际工作中的要求。它可以在有限的面积内，根据设计者的要求设计出数量庞大和多种线宽的电气连接线、多种规格和形状的焊盘和连接孔，精确地设计出符合高频中要求的特定阻抗的印制线路板和微带电路。

TANGO 软件包包括电原理图编辑 TANGO—SCHEMATIC、电路板设计 TANGO—PCB、线路板自动布线设计 TANGO—ROUTE 等。

为了便于广大电子工程技术人员迅速掌握有关 TANGO 的用法。本刊将出版《印制线路板图的计算机辅助设计》专辑，供 TANGO 用户参考（并可提供 24 针打印机，如 1724，3070……等打印驱动软件）。

电子工业出版社新书介绍

《**电子测量仪器选购指南**》定价 35.00 元

本《指南》汇集了我国和美、英、法、日、西德等十多个国家的主要电子测量仪器厂商生产的有关产品，给出每一种仪器的生产厂商型号、主要技术性能。

《**最新简明中外集成电路互换型号手册**》定价 18.00 元

本手册收集中国、美国、日本、西德等 150 家集成电路和元件厂家，45000 个最新型号互换信息及技术参数、管脚应用图，提供中外各种集成电路型号互换资料、接口电路，运算放大器脚图，国内外集成电路型号互换表，中外电阻、电位器、电容器。适合电子产品设计，家电维修的计算机和半导体研究、元器件供销工作人员及大专院校师生参考。

《**家用电器元器件手册**》（精装）　定价 56.00 元

《**传感器敏感元器件大全**》（精装）　定价 64.00 元

《**标准集成电路数据手册**——TTL **集成电路**》（精装）定价 31.50 元

《**标准集成电路数据手册**——**运算放大器**》（精装）定价 32.00 元

《**国内外功率晶体管实用手册**》上（精装）定价 21.00 元

《**国内外功率晶体管实用手册**》下（精装）定价 45.00 元

以上六种图书是大型的、完整的电子工具系列丛书。可供研究、设计、生产、技术开发等部门有关专业的工程技术人员、大专院校有关专业教师、学生、实验室科研及广大无线电、电子爱好者、家用电器、电子修理行业的同志阅读、查阅。

《dBASE **应用解疑**》定价 5.00 元

《IBM PC **编程指南**》定价 7.00 元

《DOS/BIOS **使用详解**》定价 8.00 元

《MS—DOS **高水平程序设计**》定价 5.50 元

以上四种图书是 dBASE 应用、IBM PC 机、C 语言和汇编语言使用必备用书、该书编排合理、启发性强、由浅入深、为计算机系统开发提供了得力工具、是开发人员开阔思想解决疑难、提高水平的必读之书。

邮购办法：邮局汇款　北京万寿路电子工业出版社发行部邮购科

银行汇款　开户行北京市工商银行翠微路分理处帐号 661036—40

注：购书清单写在汇款单"附言栏"内写明书名、册数、收书人详细地址和邮政编码。

本部电话：81.3693　电挂：3101　邮政编码：100036

电子工业出版社软件部发行
CCED V3.0 中文字表编辑软件

软件开发者：朱崇君

CCED 是针对中文编辑的特点开发而成的编辑软件。特点：除吸收了各种字处理软件的优点外，还增加了表格处理功能和数据运算功能。使用户在同一文件中既可做表格，又可写文章。CCED 还能直接调用 dBASE 的数据文件，形成各种表格输出。

在文字处理方面，CCED 除了具备一般编辑软件的基本功能（如：删除、恢复、搜索、替换、复制、对中等）外，还包括了 EDLIN、PE 及汉字 Wordstar 的全部优点，集 PE 中跨越文件的各种块操作功能与 Wordstar 中的排版功能、信函批处理功能于一身。CCED 中的块操作可以实现文件中或文件之间文字块的复制、移动或覆盖。块打印可帮助您完成诸如根据通信录打印贺年卡的任务。段重组可支持中英文混合排版，在保持单词完整性的前提下，排版结果可使行的首尾对齐，并妥善地处理了行首、尾的标点符号。CCED 还具有 EDLIN 中的上行复制功能，并将其扩展到全屏幕操作。

在表格编辑方面，用户可在通过光标画线制表，也可以利用程序本身的功能自动地生成表格。对于画好的表格，可以调整宽度。在表格中插入一行时，纵向表格线能自动保持连贯。在字处理中所用的编辑功能均可以在表格编辑中使用。更必表内数据时，采用线保护状态，可使原表格保持不变。为了能同其它软件交换数据，CCED 所制作的表格文件为纯粹的文本文件，而不带任何隐含控制符。

在数据计算方面，用户可在编辑版面的任何位置列算式，进行计算。对于表格中的数据还可以列写公式按行或列成批计算。

CCED 的另一个突出特点是能够调出 dBASE 数据库，产生各种报表输出。dBASE 是微机上常用的数据库系统，其报表打印一直很困难，往往需要用户编制许多繁琐的程序才能实现。而所编的程序常常是针对某一种报表，一旦要改动或产生新的报表，就得修改或新编程序。使用 CCED 可使 dBASE 的报表自动输出。无论结构多么复杂的报表，用户只要用 CCED 画出一个报表的样子（称之为样本表格），就能从数据库中提取数据对其填表输出。样本表格可以长期保存，亦可随时修改。最初的样本表格还可以由程序自动生成，其操作十分简单。生成样本表头时，字段名能够自动排放整齐。由于采用了这种样本表格技术，更为管理系统开发提供了方便。

联系地址：北京 173 信箱电子工业出版社软件部
邮政编码：100036

北京华源技贸商行
经营：中华学习机、苹果机系列产品

中华学习机　910—930 元（保修三年）
天坛学习机　995 元（保修一年）
软盘驱动器：东芝　590 元
Superb　640 元
中华学习机各种软件 800 余种
软盘：（学生盘）、ACT、DATACE、3M、Maxell janus）
2.5～7.5 元
单色显示器　520—550 元
中华学习机扩充卡　打印卡　Z80 打印卡
驱动卡　双驱动卡等
游戏棒（开关量，模拟量）　47—72 元
中华学习机各种资料、书籍　30 余种

凡在本部购买学习机者，将得到保修、免费培训的优惠，欢迎来函索取目录，本部代修学习机、驱动器、并可为学校配置中华学习机网络。批发面议。

地址：北京东城沙滩五四大街 52 号　邮政编码：100009
电话：512.7308　开户行：工商行崇外分理处中都信用社
帐户：华源技贸商行　帐号：中都 03040
联系人：于民　张小键

浙江温州市精密电子仪器经营部

地址：温州市飞霞南路龟湖路
邮编：325004　电挂：5072　电话 336072
开户：工商五马辨　帐号：1567892

供应（单价）EPROM2716/32/64/128 均 18 元；27256，25 元；RAM6116，15 元；6117，10 元，6264，32 元，8085/8031，15 元；Z80CPU，7.8 元；CTC/PIO 均 8.8 元；SIO，12 元；DMA，20 元；8039/8035/8255，均 10.5 元；8155，15 元；8243，12.5 元；8253，14 元；8279，22.5 元；8257/8259，18 元；8251，16.5 元；CD4508/4514/4515，4.8 元；光隔 P541—1/521—1，2 元；P541—2，3.5 元；P521—3，6 元；P521—4，7.5 元；40P 集成插座，0.65 元；晶振 2M，2.8 元；3.579545M，2 元；4M，3.2 元；2N3055，6.9 元；2SC1008，0.65 元，全桥 5A/50V，2.6 元；10A/50V，8.32 元；2μF/1KV，9 元，数字电压表 250V/电流表 5A，均 143 元。以上邮资 2 元/次，另有数字式实验多用仪，功能有（1）1MHZ 频率计（2）100PF～10000μF 电容计（3）三路可调稳压电源 1.2V～26V/1.5A 数显及 ±5V～±15V/1.5A，价 496 元，邮资 12.5 元/台，另有系列 74LS，CD4000 电路及数百种其它晶体管，集成电路，继电器，阻容等均为进口器件，函索价目表者，请用贴足邮票信封写清回邮地址寄我部。

求　援

- PC—8100 图纸
  联系地址：新疆哈密桥梁厂薄如兰转
  张子芳
  邮政编码：839001
- 有关 MZ—731 选配磁盘驱动器的方法及资料
  联系地址：江苏油田运输船一、李志明
  邮政编码：225261

致参加“普及型PC个人用户软件交流联谊活动”的用户

农科院计算中心

自从今年一月份“电子与电脑”杂志刊登组织“普及型PC机个人用户软件交流联谊活动”以来，得到了社会各界人士，尤其是“电子与电脑”杂志用户大力支持和充分的肯定，作为联谊点的挂靠单位，我们中国农科院计算中心培训部全体同志能有这样一个为广大用户直接服务的机会，能为PC机个人用户获得自己需要的软件，提高用户的应用水平作一些力所能及的工作感到十分高兴。

到目前为止，我们先后收到来自各行各业PC机个人用户要求参加该项活动的已达120多人。大家积极主动地提供了自己拥有的软件清单；反映了希望通过联谊活动得到自己所缺乏的软件愿望和要求；提出了在使用中碰到的许多困难和问题；对搞好联谊活动提了殷切的希望。大家的积极支持是我们办好联谊活动的根本条件。针对大家的来信、来函、来电，我们已组织人将有关材料进行了计算机管理。归纳起来大家普遍的要求：一是有了机器，由于硬件资料缺乏，对其性能了解较少，操作使用尚有一定困难，所以希望提供有关的硬件资料信息。二是有了机器，由于缺乏软件，机器的功能不能很好的发挥，利用率低，所以希望相互交流交换软件。三是有了软件由于缺乏软件使用说明的资料，致使软件不能得到利用，所以希望提供有关软件使用的资料来源。针对上述问题和大家的普遍要求，下一步联谊活动的搞法我们准备作如下的安排：

1. 关于PC机系统硬件资料，哪些用户需要什么样的资料，希望直接来函给我培训部说明，目前我部拥有120种微机各种硬件、软件资料，北京市用户可直接来我培训部选购，外地用户可邮购，但资料的名称邮政编码、单位地址、收件人姓名务必写清楚。

2. 关于软件的交流，大家所需的软件覆盖面较大，既有系统类软件，也有应用软件类，既有文字处理方面的软件，也有统计、制表、制图等方面的软件，既有输入方法，也有教学软件，甚至不少用户需娱乐方面的游戏软件等等。所以单靠我们这里一个点很难满足大家的要求，但作为大家常用的软件我们陆续公布常用软件清单(附后见48页)，供大家交流。原则仍然是北京市用户可持软盘直接由我部获取。外地用户写明所需软件名称，办理邮寄，这是一种办法。我们已初步与《电子与电脑》杂志编辑部商量决定在今年年底举办一次北京地区用户的联谊会，通过联谊活动交流使用的经验、交流软件。借此摸索经验，然后扩大联谊会范围和内容。什么时候举办，以通知为准。

3. 关于使用中的问题。我们正在把大家使用中的问题进行整理、归纳，准备将带有普遍性的问题组织有关方面专业人员，通过《电子与电脑》杂志这一园地回答给用户。

“第一批常用软件清单”附后：

联系人：中国农业科学院计算中心培训部王路敬

邮政编码：100081

通讯地址：北京海淀区白石桥路30号

(上接第31页)

只要继续输入下一字的编码，此字就会被自动选中作为输入。

用CPC卡上的五笔字型输入法输入的汉字，不需经任何处理即可方便地用于编程、打印或用于CPC卡上的屏幕编辑软件。

三、汉字全屏幕编辑功能

使用PR#8进入汉字后，按【F5】键即可进入汉字全屏幕编辑状态，进行文字的插入、删除等编辑，并可存盘或打印。编辑功能如下：

光标上移

↑

光标左移← →光标右移

↓

光标下移

【CTRL－N】	在光标所在行开一新行
【CTRL－D】	删除光标所指字符
【CTRL－X】	删除光标所在行
【CTRL－I】或【TAB】	插入/履盖状态转换键
【CTRL－Q】【西文】	快速/正常编辑转换键
【QUIT】、【T】	设置字号
【QUIT】、【D】	删除整个文件
【QUIT】、【S】	文件存入磁盘
【QUIT】、【L】	文件调入内存
【QUIT】、【P】	打印盘上文件
【QUIT】、【QUIT】	退出屏幕编辑

当打印文件时，系统会询问打印参数。打印参数如下：

T：字号　C：字距　N：行距　H：行内字数

回答时，各参数用“,”分开。取值范围为：1≤T≤15；0≤C≤254；0≤N≤254；0≤H≤254。软件默认值为：T＝6、C＝4、N＝4、H＝40。另外字号可在编辑文章时直接设置，从而实现同一文章多种字号。

在屏幕编辑软件中也可使用五笔字型输入法。使用方法同前介绍。

中华学习机CPC卡五笔字型输入法采用独特的检索方式，使时间和空间的占用都减到最低。从时间来说，当用户输入完五笔码后，基本感觉不到输入后的编码检索时间即可出字，为实现高速盲打创造了条件；从空间来说，CPC卡只占用了用户空间高端的$9000～$91FF的512个字节，并用HIMEM保护起来。对用户来说，不会有什么明显影响。

中华学习机CPC卡集三种重要功能为一体，使用户能在中华学习机上以低代价换得高效的文字处理功能。随着CPC卡今后不断的改进，中华学习机也将被升级为中文处理机。

常用交流软件清单

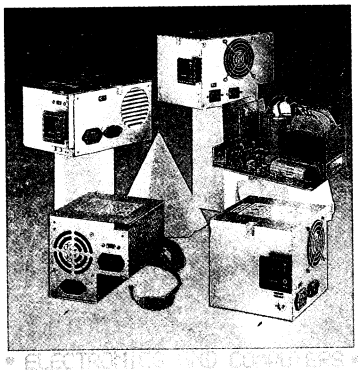
	版本号	适用机型	所需盘数		版本号	适用机型	所需盘数		版本号	适用机型	所需盘数
CCDOSS	V2.00	A	1					MSC		A&C	9
CCDOS	V2.10	A	1	硬盘加密		A&C	1	COBOL Ⅱ	V2.0	A&C	9
CCDOS	V2.41	A	1	加密、拷贝		A&C	1	FORTRAN		A&C	2
CCDOS	V3.0	C	1	解密软件(专用)		A&C	1	MS—FORTRAN		A	2
CCDOS	V4.0	A&C	5	WS(字处理)		A	1	FOXBASE PLUS	V2.00	A&C	2
GWDOS	V2.13(A—G)	A&C	7～8	WS		C	1	DBASE Ⅱ		A	1
LLDOS(双音双拼)		A	2	WS		D	1			C	1
		C	1	WS		286	1	DBASE Ⅲ PLUS		A&C	6
MSDOS	V4.01	A&C	6	XE	V2.8	A&C	2	全汉化 DBASE Ⅲ	V1.01	A&C	1
CLIB24＊24(汉字库)		A&C	11	XE		CEGA	2	汉字编译 DBASE Ⅲ			2
CLIB32＊32		A&C	12	HW(字处理)		A&C	1	编译 DBASE Ⅲ		C	2
CLIB48＊48		A&C	25	HW		CEGA	1	DBASE Ⅲ 辅助系统		C	4
各种打印驱动程序		A&C	1	CW(词语输入方法)		CEGA	2	DLLTIL		C	7
高级打字驱动程序(PS)		A&C	1	CVICH SINER			5	INFORMIX		C	4
LP24		A&C	6	LOTUS(图形管理系统)		A	1	DOFRMAT		C	1
对折打印		A&C	1	LOTUS		C	3	编译 PASCAL			2
IBM 检测程序		A&C	1	LOTUS(教学盘)		C	1	R:BASE 4000(库管理系统)	V1.11	C	2
PCTOOLS(汉化)	V1.0	A&C	1	MAKER(报表汉化)	V2.1	A	1	日常事务处理及生物统计		A	4
PCTOOLS(汉化)	V4.21	A&C	1	汉化 BCT 制表		A	3	DUTIL		C	1
PCTOOLS(西文)	V4.11	A&C	1	CCED 制表编辑软件		A&C	1	STATPAK 统计	V3.11	A&C	2
PCTOOLS(西文)	V5.5	A&C	4	高级智能制表软件(80元)	V2.2	A&C	1	STATISTICS 多功能统计		A&C	1
BOX(工具箱)		A&C	5	AUTO HCAD(汉化计算机辅助设计)		A&C	10	科研管理软件		A&C	2
DMA(磁盘管理恢复文件)		A&C	1	AUTO CAD	V2.17	A&C	5	日常管理软件		A&c	2
五笔字型		A	2	ACAD(汉化计算机辅助设计)		A&C	8	通用工资管理软件		A&c	1
五笔字型造字软件		A	2	OFFICE		A	2	财务报表管理软件		A&c	1
五笔字型导硬盘		A	1	OFFICE		C	2	卫生统计程序软件包		A&c	1
五笔字型		C	1	OFFICE		D&286B	2	人事软件		A&c	2
五笔字型学习盘		单显	2	OFFICE		286	2	中文干部管理档案		A&c	1
		彩显	2	OFFICE		CEGA	2	通用财会记帐;银行对帐软件		A&c	2
表形码学习软件			1	GCLISP(汉化)		A&C	8	检测 77 种;消 12 种病毒	V1.0	A&c	1
长城笔形码输入方法		GWBX—CGEA	1	TURBO PROLOG(系统盘)		A&C	1	检测 78 种;消 13 种病毒	V1.0	A&c	1
COPYWRITE	V5.01	A&C	2	TURBO PASCAL	V5.0	A&C	3	检测 79 种;消 17 种病毒	V1.0	A&c	1
GW87		A&C	1	TURBO C	V2.0	A&C	7	WORDPERFECT(从国外引进)			5

注:适用机型一栏内凡标有“A”均适用于 IBM—PC、PC/XT,标有“C”的为彩色显示器高分辨率下运行的软件。

ISSN 1000-1077

電子與電腦

8



一九九一年

总期第77期

電子與電腦

目　录

·综述·

永不过时的电脑软件—自生软件系统 …… 郑佑(2)

·PC 用户·

通用下拉式菜单的实现方法 …… 张志远(3)
如何改变数据记录在库中的绝对位置 …… 李海飞(4)
dBASE Ⅲ 过程文件加密的发现 …… 刘人伟(5)
对 SET CARRY ON 的一点补充 …… 李志刚(5)
IBM PC/XT 磁盘扇区读取方法分析 …… 吴兆熊(6)
使用设备驱动程序 ANSI.SYS 扩充屏幕和键盘 …… 曹长英　李杰(8)
用 PC TOOLS 修改软件的汉字提示 …… 李开新(10)
巧用批命令定义功能键 …… 邢观锋(10)
CCDOS 下屏幕死锁的解决 …… 成刚　杨明秋(11)
VDISK 的妙用 …… 叶舸(11)

·学习机之友·

谈谈 FORTH …… 丁志伟(12)
再谈特殊 6502 指令 …… 李永前(13)
扩展监控系统命令 …… 蒋建一(14)
MZ－731 机绘制的财务计划执行分析报表 …… 黄浩珍(14)
机器语言程序计算阶乘 …… 易鹤铃(17)
ProDOS 磁盘操作系统入门(续) …… 廖凯(18)
用堆栈思想处理任意 N 个元素的排列组合问题 …… 索素文(20)
简单有趣的解密办法 …… 张亭(22)

·C 语言初阶讲座·

第七讲　结构数据(一) …… 李文兵(22)

·学用单片机·

单片机汉字打印驱动程序的设计思想 …… 尹征(25)

·学装微电脑·

微电脑控制跨接线弯制机 …… 易齐干(29)

·电脑巧开发·

如何用 EPROM 读写器对 EEPROM 编程 …… 闫怀兵、陈立山(35)
LASER－310 机加接数字音响发生器 …… 王建锋(36)

·维修经验谈·

任天堂系列游戏机检修实例(续) …… 林华春(38)
"8088 系统板测试卡"对系统板总线部分的诊断测试 …… 梁杰熙(39)
一九九一年计算机初级程序员有奖竞赛试题 …… (41)
一九九一年计算机初级程序员有奖竞赛试题答卷 …… (48)

·新书与软件·

给 TANGO 软件增加汉字元件库 …… 黄健、陈军(47)

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

永不过时的电脑软件——自生软件系统

郑 佑
(WORLD-WIDE INSTITUTE OF VALUATION, U.S.A)

一个具有永久生命的软件系统(称为自生软件系统)已在近期研制成功,它是基于一个软体自动化的新概念,其目的在于将目前所有的软件都转化为一种永久的形式——这种形式将使软件永远不会过时或随着电脑科技的新发展而被淘汰。以这种形式写出来的软件一旦完成,就永远不需要重写,因为电脑可以自动将它们修正或改进。

目前,软件是寄生于电脑技术。例如,当电脑硬件系统,操作系统或语言系统改变时,在这些系统环境下写出来的软件系统就会被淘汰。一般来讲,所有现在的软件,迟早都是会随着电脑技术的不断变化而被淘汰,这一点是不容置疑的。

在这种永不过时的软件系统中,所有的软件都是由电脑生成的,都可以在每一次再生成的过程中,自动的由电脑来解释和修正。在技术上,软件的重新生成是由向软件生成器输入一种特殊的描述问题特性的信号文件。此信号文件就是键盘输入的直接记录,软件生成器可以直接认识这种信号文件,因而也可以自动地将它所代表的软件加以修正。软件生成器和信号文件生成器本身也都必需有它们自己的信号文件,因而也可以被重新生成。因此,所有的在这一永不过时的软件系统中的软件都是由电脑生成,它们都有描述它们自己的信号文件。

这种自我生产的软件系统与生物细胞的结构极为相象,为电脑设计的基本原理提供了一个新的概念:以自我生产(简称为自生)细胞科技为基础设计新型电脑。能够自我繁殖的生物细胞同时具备许多优点,自生细胞科技也同时具有相类似的优点:

1. 目前大多数的软件都是由专业程序员直接用电脑语言写出来的,而在自生软体系统中,所有的程序(正如生物细胞中的蛋白质)都是经向电脑中的软件生成器(正如细胞中的蛋白质生成器,如核糖体)输入一种特殊的描述问题特性的信号文件(正如细胞中的脱氧核糖核酸,DNA)而产生的。目前的电脑系统是一种不成熟的粗糙的产品,因为在最初发明时,只想到把产品尽快的造出来,而结果是在这种系统下产生出来的软件,就象是直接制造出的很容易腐化的蛋白质,由于没有生命力而被淘汰。

2. 上述的那种特殊的描述问题特性的信号文件,正如细胞中的脱氧核糖核酸(DNA)有永远生存的特性,它不但可以用于任何电脑系统,而且不受任何将来电脑科技新发展的影响。由于这种信号文件是使用者与电脑用人类语言对话的方式所产生的。因此所有只要受过一般教育的人都能够使用电脑,以上二点使电脑的使用范围普遍增加。

3. 由自生软件系统所产生的自生软件细胞单元可以代替电子神经网(neural network)中的电子神经细胞(neuron),这种神经细胞可以自我繁殖并相互传递信息。

4. 人工智能的发展正需要多元,多层次的自生软件细胞系统,人工生物(Artificial life)也需要自生细胞科技的发展。

5. 现今的电脑发展已经到了一个相当的程度,应该有足够的适应性,或者说是复杂性,没有任何理由使电脑与其使用者(包括程序员在内)在脱离人类语言的情况下相互隔绝。自生软件系统将使高级电脑语言全部作废。

6. 目前的电脑结构是一个初期,草率而伧促(从自然创造生物的眼光来看)的设计结果。就连它的发明者之一 Johnvon Neumann,也认为不满意,在他生命的最后几年中,一直致力于研究一个和生物细胞类似的系统。

7. 一个可以由最初的软件自我生成同样软件的示范系统已经设计成功。

大多数目前的科技都是由简单部份,结合构成复杂系统。而自生细胞科技的主要概念是,从包含一切抗能和所有基本知识的复杂的细胞元开始。自我生成同样复杂的新细胞,新细胞可以在生成时改变它的性能来适应各种新的环境的需要。

这一再生细胞科技将是自动化的最终形式。它是解决和控制高复杂性问题所必不可少的。同时也是生命的特征。细胞观念使多元化细胞设计成为可能。细胞的再繁殖使程序的复杂性成几何级数增长。正如人类的大脑可增加 10^{12} 的复杂性,如果没有自我繁殖的能力,如此巨大的数字是不可能用人工制造出来的。

用科技术语来讲,生物生命也可以被称作是自动化的自生细胞科技。可能正是由于它的过份复杂,使得外来的全面控制成为不可能,而最终的控制就让给此细胞为基础的生命体自已掌握。从技术上来讲,生命是由天文量级复杂程度的程序完成的,是真正最先进的科学科技,尽管至今为止,还没有人能够将此程序破译。现在二十世纪的无生命科技,落后于生命科技至少几千年。不过,第一个自生软件系统的产品正在设计、计划于明年问世。

今后,软件系统将分成二种:一种是永不过时的,如自生软体系统,另一种是迟早会被淘汰的,如目前大多数的软体系统。照理来讲,所有现在非自生的以及将来的软体系统将都得改写成或写成自生软件系统。

PC用户

通用下拉式菜单的实现方法

山东计算机服务公司临沂分公司　张志远

一个设计良好的菜单不仅富有艺术情趣，令人耳目一新，而且能减轻用户记忆负担，减少错误操作，提高效率。通常的菜单设计方法是根据编号或字母选择相应的功能，使用者要全神贯注于屏幕信息，寻找按键的位置，容易出错。近来流行的菜单形式有下拉式、上弹 式及开窗口等，较好的解决了上述问题。它使用移动键上下左右移动光棒，光棒所在的位置即要选择的菜单项，按回车后进入该项功能，使用方便，不易出错，且保留主菜单画面，子菜单以下拉对应方式层层进入，层次分明。这种菜单虽然好，但大多是在工具软件中采用汇编语言或其它语言编写，一般用户不熟悉。笔者通过多次实践，在 Foxbase＋2.0 和 DBASE Ⅲ＋下，实现了下拉式菜单的设计，现介绍给大家参考。

1. 主菜单(程序清单见 CD1・PRG)

主菜单采用在屏幕的顶部一行显示各菜单项，其中有一菜单项的底色与其它菜单项不同，并进行闪烁。这一底色被称作光棒，此光棒可以通过按键左右移动，按回车键即选定了光棒所在的菜单项。调用方法：

主程序将各菜单项的内容赋给变量 G1～Gn；

定义 Z 为全局变量；

调用 DO CD1 WITH 参数 1，参数 2，参数 3，参数 4(参数 1：显示行、参数 2：起始列、参数 3：列距、参数 4：列数)；

根据变量 Z 的值进入相应的功能子程序(其中：Z＝“　”退出，Z＝“1”菜单项 1，Z＝“2”菜单项 2，……Z“n”菜单项 n)。

2. 子菜单(下拉式菜单 CD2・PRG)

主菜单选定后，便进入相应的子菜单，子菜单的列座标与相应的主菜单项对齐，且采用下拉方式，保留主菜单。光棒可通过按键上下移动或退出，回车即选择。只是 DO 语句中参数的意义不同，参数 1：起始行；参数 2：起始列，参数 3：行距，参数 4：行数。

另外需要说明的一点，此通用菜单程序，根据需要变换行坐标，可变成通用上弹式菜单程序。

```
*CD1.PRG 主菜单程序
PARA A,B,C,D
*A:显示行,B:起始列,C:列距,D:列数
SET COLO TO W+/R+
@24,0 SAY"提示:左移←右移→选择 Enter 退出 Esc"
SET COLO N/GR+
N=1
L=B
DO WHILE N< =D
X=LTRIM(STR(N,3))
@A,L SAY G&X　&& 显示各菜单项
N=N+1
L=L+C+LEN(G&X)
ENDD
L1=L-LEN(G&X)-C
X="1"
L=B
DO WHILE . T.
SET COLO TO W+*/B+
@A,L SAY G&X　&& 显示被选菜单
X1=X
L1=L
W=INKEY(0)　&& 返回击入的键值
SET COLO TO W+/B+
DO CASE
CASE W=19　　&&←左移
X=IIF(VAL(X)=1,STR(D),STR(VAL(X)-1,3))
X=LTRIM(X)
L=IIF(X=LTRIM(STR(D)),L1,L-LEN(G&X)-C)
CASE W=4　&&→右移
X=IIF(VAL(X)=D,"1",LTRIM(STR(VAL(X)+,3)))
L=IIF(L="1". AND. LEN(X)=1,B,L+LEN(G&X1)+C)
CASE W=13　&&Enter 选择
@A,L SAY G&X
Z=X
EXIT
CASE W=27　　&&Esc 退出
Z=" "
EXIT
ENDC
SET COLO TO N/GR+
@A,L1 SAY G&X1　　&& 恢复上次被选菜单
ENDD
RETU

*CD2.PRG 子菜单(下拉式)程序
PARA A,B,C,D
*A:起始行,B:起始列,C:行距,D:行数
SET COLO TO R+/G+
@24,0 SAY"提示:上移↑下移↓选择 Enter 退出 Esc"
SET COLO TO R+/W+
N=1
DO WHILE N< =D
X=LTRIM(STR(N,2))
@A+(N-1)*C,B SAY G&X　　&& 显示各菜单项
N=N+1
ENDD
L=A
DO WHILE .T.
```

```
SET COLO TO GR+ * /N
X=LTRIM(STR((L+C-A)/C,2))
@L,B SAY G&X      && 显示被选菜单
X1=X
W=INKEY(0)      && 返回击入的键值
SET COLO TO N/W+
DO CASE
CASE W=5      && ↑上移
L=IIF(L=A,(D-1) * C+A,L-C)
CASE W=24      && ↓下移
L=IIF(L=(D-1) * C+A,A,L+C)
CASE W=13      &&Enter 选择
Z=X
EXIT
CASE W=27      &&Esc 退出
Z=" "
EXIT
ENDC
SET COLO TO R+/W+
@(&X1-1) * C+A,B, SAY G&X1  && 恢复上次被选菜单
ENDD
RETU
```

如何改变数据记录在库中的绝对位置

河北涿州市物探局研究院软件所　　李海飞

dBASE Ⅲ提供的'COPY TO〈FILENAME〉STRUCTURE EXTENDED'命令可以用来把当前库文件的各个域变成〈FILENAME〉文件中的记录。〈FILINAME〉包括 FIELD_NAME,FIELD_TYPE,FIELD_LEN,FIELD-DEC 四个域。对我们来说,FIELD_NAME 最为重量。通过对〈FILENAME〉的记录的循环可以达到对当前库文件的整个域进行操作的目的。

再利用 dBASE Ⅲ的宏替换功能,可以使本程序具有通用性。在编程过程中还用到了一些临时库文件,如 TempFile.dbf、sef.dbf,用完以后即把它们删除了。

程序如下:

```
set heading off
set talk off
con=.T.
 * con 为控制循环的逻辑型变量
do while con
clear
accept '请输入数据库文件名:' to Processfil
FName=ProcessFil+'.dbf'
if .not. file('&FName')
    ? '数据库文件不存在,返回'
    return
endif
sele 1
use &FName
 * &FName 为需要处理的数据库文件
copy to sef structure extended
sele 2
use sef
 * sef.dbf 为 &FName 的结构伸展文件
sele 1
accept '要浏览整个数据库吗(Y/N)?' to Ans
if upper (Ans)='Y'
list all
endif
input '请输入准备转移的记录的记录号:' to SouRec
goto SouRec
copy next 1 to TempFile
sele 3
use TempFile
 * TempFile.dbf 为只有一个记录的临时文件
sele 1
input '请输入转移到的记录号:' to DesRec
goto DesRec
if SouRec<DesRec
    insert blank
 * 若源记录号小于目的记录号,则在目的记录后面插入
else
insert blank before
 * 若源记录号大于等于目的记录号,则在目的记录前面插入
endif
sele 2
go top
do while .not.eof()
 * 通过对结构伸展文件 sef.dbf 的循环来达到对 &FName 文件循环的目的
FN=FIELD_NAME
sele 1
replace &FN with c->&FN
sele 2
skip
enddo
sele 1
if SouRec<DesRec
dele record SouRec
else
dele record SouRec+1
endif
pack
accept '还要继续吗(Y/N)?' to Ans
if upper (Ans)='N'
    con=.F.
endif
use
sele 2
```

```
use
erase sef.dbf
sele 3
use
erase TempFile.dbf
enddo
set talk on
set heading on
return
```

dBASE Ⅲ过程文件加密的发现

大港石油管理局炼油厂　刘人伟

我厂现用由电子工业部第六研究所1988年10月出版的计量管理软件包《CMIS V1.0》，当数据库录入到三千五百以上时，再由此库进行其它方面的管理和应用，结果发现软件包的运行速度越来越慢，且出错率逐渐增加，造成部分数据录不进去或串库，以及数据行缩小等，使得打印出的台帐格式不齐，数据丢失。虽然该软件包已经采用CCDOS 4.0系统来提高运行速度，但由于受到dBASE 1.0B版本的限制，无法正确完成计量管理的大量工作，使该软件内的功能、速度等均受到很大限制。例如：机内每做一项工作的进入和退出，都需要很长时间方可完成，从而有可能造成不必要的错误。因此要经常对数据库进行备份处理，以保证不因意外而发生错误。这给正常工作带来很多麻烦和困难，及要求用户要了解该软件包中部分数据的作用，否则要整盘拷贝（对所有数据库），对一般用户来说有一定难度。为了提高软件的速度和可靠性，以充分利用我厂现有AST286机的特性，我把《CMIS》软件包在FOXBASE＋2.0上试运行，结果发现该软件包上的过程文件有局部地方加密，而无法利用FOXBASE＋的高速，高容量等特性。

在对此dBASE Ⅲ过程文件解密时，我首先试用EDLIN对加密文件内容进行修改，但未成功。原因是加密的行语句不能显示而无法修改，若进行重新输入编辑此行，却出现该行语句重合叠加等，而无能为力。又试用Wordstar等文本编辑软件解密也无效。虽然调用dBASE Ⅲ的过程文件编辑程序，进行有密语句重写存盘，可以解决。可要把上百个过程文件中的加密文件进行修改，工作量很大很烦琐。但解密是可行的，同时发现此加密方法为行语句尾加密。

针对行语句尾加密，我首先利用DEBUG对其中一个有密文件XB05.PRG进行反汇编，找出正常语句行尾与有密语句行尾的区别。结果发现正常无密语句行尾标记为0D0A（回车、换行），而有密语句行尾标记为0D8A，所以做相应修改即可完成。

经修改后的XB05.PRG文件，可用Wordstar、EDLIN进行文本编辑修改，也可以完整地输出程序清单。同时，能顺利地在Foxbase＋上运行，只要把Foxbase＋上的config。FX做适当修改即可。

通过上述发现与修改，我决定用Pctools R1.0软件，对整个CMIS软件的过程文件进行全面解密。使用Pctools的search功能，对过程文件进行有密处查找修改，找出含有十六进制码0D8A处，把8A的8改为0，即可完成该语句行的解密工作。此方法简单易行，任何人都能在短时间内，对上百个加密文件进行统一的解密工作。在解密时，发现有个别行语句行尾为0D73，这些在解完0D8A后，再对0D73用同样的方法，把它改为0D0A即可。

对CMIS软件解密后，再在Foxbase＋2.0上运行，收到良好效果，使CMIS软件包运行速度更快，更理想，也解决了由于dBASE Ⅲ所带来的软件不足，从而保证该软件的正常运行。解了密的CMIS软件包，还可以在dBASE Ⅲ Plus上运行，运行结果也很理想。

上述加密方法也可用于自编的dBASE Ⅲ过程文件中去，以防止他人修改。只要用DEBUG或Pctools等工具软件把dBASE Ⅲ的过程文件的行语句尾0D0A中的0A改为其它十六进制数码即可。

若把0D0A改为0D1A，（1A为文件结束符^Z的代码），该被加密的语句以后内容，再也不能通过type和文本编辑（包括dBASE Ⅲ的文本编辑）等方法看到和修改原程序清单，所以此法更能起到保密和保护程序作用。上述方法再结合DOS和dBASE Ⅲ的其它加密方法，能达到更理想的加密效果，而不被他人所利用。

对SET CARRY ON的一点补充

常德科技情报研究所　李志刚

人们在用dBASE Ⅲ编制程序时，为了加快数据输入速度，常使用SET CARRY ON命令来使输入当前记录时携带上一记录。然而人们不曾注意到，假如上一记录是做了删除标记“*”的，那么携带时，删除标记被带上，而且若没有经过PACK命令压缩，则以后增加的记录均被带上删除标记。因使用PACK命令占时较长，在平时删除时，一般不会即时使用PACK，而是待有了一定的删除标记记录时，才集中整理数据，所以稍不注意使用PACK命令，就有可能引起大量数据的丢失，而让人感到莫名其妙，因此本人提出了注意的方法，供参考。即：在使用APPEND命令，并输完当前记录后，紧补上RECALL命令，以唤醒当前记录，因不管记录有无删除标记，RECALL命令均将其变为无删除标记，这样即避免了误删记录的可能性。

IBM PC/XT 磁盘扇区读取方法分析

重庆渝州大学计算机教研室　　吴兆熊

系统结构研究及程序编制人员，常有必要读取某些程序的机器码及反汇编文本。由于保密、覆盖、不常驻内存等因素，要从内存直接获得完整的反汇编文本往往很困难。这种情况下，只好直接从磁盘上按文件的扇区分布，把文件完整地读出。要做到这一点，首先须了解 IBM PC/XT 提供的几种读磁盘扇区方法。本文拟通过对磁盘扇区结构和 ROM BIOS 的分析，来阐明这些读磁盘扇区方法的各自特点。

磁盘结构及扇区分布

双面软盘驱动器有两只读写磁头，固定在存取臂的头部，各用于对磁盘的一个面进行数据读写。存取臂可沿磁盘径向移动；数据是在磁盘旋转过程中写入磁盘上。所以，对应于读写磁头所在位置的数据记录部分为一个圆圈。这一圆圈就是可记录数据的，称为磁道的轨迹。一个磁道读写完毕，存取臂带动磁头沿盘面径向移到另一位置。因此，在盘面上形成了多条同心圆的磁道。PC/XT 机用的双面倍密度软盘，每面有 40 个磁道，两面共 80 磁道。

硬盘机的结构与上述类似。在 IBM PC/XT 机中常用的是二块盘片 4 个磁头的 10M、20M 字节容量的硬盘机，其结构如图 1 所示。

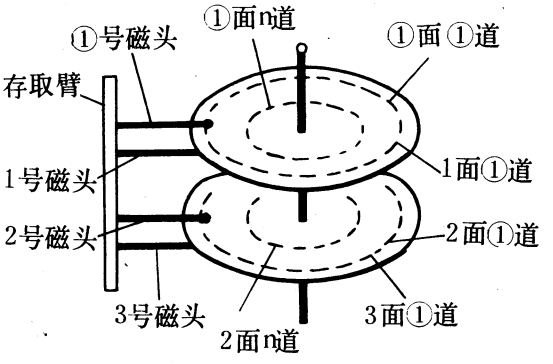


图 1　磁盘驱动器结构示意

标准 10M 硬盘每面有 305 磁道，增强 20M 硬盘每面有 614 磁道。

每一盘面上的磁道均从零开始编号。因为所有磁头均固定在存取臂上，当存取臂停留在某一位置时，所有磁头均驻留在各对应面的同一编号磁道上。如图 1 所示，当 0 号磁头处于 0 面 0 道位置时，其余三个磁头也分别处于对应盘面的 0 道上。我们把各个盘面上的同编号磁道重叠起来，组成一个圆筒，称之为柱面。即是说，0 号柱面指的是 0～3 面的 4 个 0 号磁道，1 号柱面指的是 0～3 面的 4 个 1 号磁道，等等。这意味着 20M 硬盘具有 614 个柱面。与此类似，双面软盘每一柱面包含 2 个同一编号的磁道，这两磁道组成一柱面，共有 40 个柱面。

双面倍密软盘的每一磁道分为 9 个扇区，每扇区可记录 512 字节数据，一张盘片可记录的数据量为：

磁头数×每面磁道数×每道扇区数×每扇区记录字节数＝2×40×9×0.5K＝360K

标准 10M 和增强 20M 硬盘，每磁道分为 17 个扇区。每扇区可记录数据量同样为 512 字节。按上述公式，标准 10M 盘共可记录 4×305×17×0.5K＝10370K 字节；增强 20M 盘可记录 20876 字节。软盘及硬盘每面的磁道及扇区分布如图 2、3 所示。

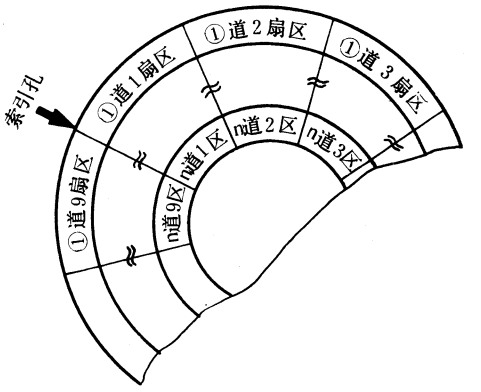


图 2　软盘的磁道及扇区分布

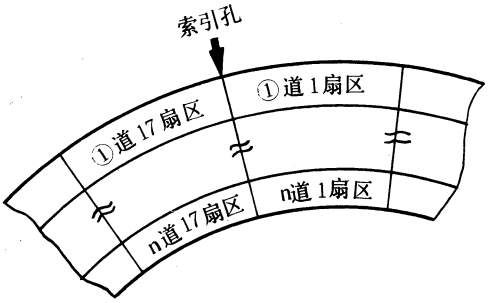


图 3　硬盘的磁道及扇区分布

磁盘数据读取

在 DOS 系统层，可通过软中断命令 INT 25 读取磁盘扇区数据。在 ROM BIOS 则通过硬盘和软盘各自的软中断 INT 13 读取。三种方法各有什么不同呢？我们先根据前述磁盘扇区结构，来讨论扇区读取方法。

IBM PC/XT 采用两种方法：

1. 按顺序扇区号读取。

从磁盘组的 0 面 0 道 1 扇区开始，直到最末一面，末道的最后一个扇区，依序连续编号，按这方式分配给每一扇区的编号，称为逻辑扇区号。要读某一扇区，只须指定该扇区的逻辑扇区号即可。DOS 系统中的软中断 INT25，就是用的这种读取方式。这时，利用 DX 来指出准备读的起始逻辑扇区号，CX 指出须读出的扇区数目。最大可读出扇区数不超过 7EH。执行 INT 25 时。从 AL 指定的驱动器磁盘上读出的数据，被依次存入由 DS:BX 指定的内存缓冲区内。

这里，有必要介绍一下逻辑扇区号的编排原则。因为所有读写磁头均固定在同一存取臂上，磁道的改变要靠存取臂的机械移动来完成。在同一磁道上改换扇区，只须检索该磁道上各扇区的标志即可；所需时间较移动存取臂短得多。因此，记录数据时，希望把每一磁道上的所有扇区都记录满后，才移动存取臂至下一磁道继续记录，即把每一柱面录满后，才改由下一柱面续录。据此，扇区是按柱面（磁道号）的顺序连续编号。以双面软盘为例，其编号顺序是从 0 面 0 道 1 扇区开始，依序编至第 9 扇区；接下来是 1 面 0 道 1 扇区，然后是 0 面 1 道 1 扇区……。逻辑扇区号的起始扇区编号为零，即 0 面 0 道 1 扇区为 0 号逻辑扇区，如表 1 所示。

磁 道 号	0		1		…	39	
磁 头 号	0	1	0	1	…	0	1
物理扇区号	1…9	1…9	1…9	1…9	…	1…9	1…9
罗辑扇区号	0…8	9…17	18…26	27…35	…	702…710	711…719

表 1 360K 双面倍密软盘物理与逻辑扇区号对照表

与软盘不同的是，硬盘将 0 面 0 道的 17 个扇区设为隐含扇区（各型硬盘机的隐含扇区数不尽相同，请参阅有关资料）。因此，硬盘是以 1 面 0 道 1 扇区作为 0 号逻辑扇区，按上述软盘编号方法，依序编号直至最后一个磁道第 3 面的 17 扇区。

2. 按扇区所在物理位置读取。

这种方法是按磁盘的面号、磁道号及扇区在该磁道中的顺序号读取，即按图 2、图 3 所示扇区位置读取。PC/XT 机的 ROM BIOS 中有 2 个按这种方式读取扇区的子程序，即软盘 DISKETTE_IO 和硬盘 DISK_IO。它们都是靠软中断指令 INT 13 调用。尽管都是 INT 13，但硬盘和软盘的入口参数存在如下差别：

a、软盘 INT 13

该子程序入口绝对地址在 ROM 区的 FEC59H。

(AH)＝0～6；(AH)＝2 时，执行读磁盘扇区功能。

(AL)＝1～9；要读的扇区数。

(CL)＝1～9；起始扇区数。

(CH)＝0～27H；要读扇区所在磁道号

(DH)＝0，1；面号。

(DL)＝0～3；驱动器号（0＝A 盘，1＝B 盘……）

ES:BX；指向输入缓冲首地址。

调用软盘 INT 13 时，每次只能读出指定磁道内的扇区数。那就是说，(AL)和(CL)中的参数是互相制约的。如 CL 指定该磁道的 7 号扇区为始读扇区；那末，可读扇区数最多只能到该道的末扇区——9 扇区。这时(AL)最大值不能超过 3，否则将出错。

b、硬盘 INT 13

入口绝对地址在硬盘 ROM 区的 C8256H。

(AH)＝0～14H；(AH)＝0～5 与软盘 INT 13 的调用功能相同。

(AL)＝1～80H；要读的扇区数（长读为 1～79H）。

(CL)＝1～11H；起始扇区号（CL 的 b5～b0 位）。

（CL 的 b7b6 位）＋(CH)＝0～3FFH；柱面（磁道）号。

(DH)＝0～7H；磁头号（面号）

(DL)＝80～87H；驱动器号（80＝C，81＝D，……）

ES:BS；指向输入缓冲区首地址。

硬盘 INT 13 的最大可读扇区数与面号、道号及起始扇区号无关。

几点说明

1、INT 25 不能读出硬盘的隐含扇区（0 面 0 道 1～17 扇区），因隐含扇区未编入逻辑扇区号序列。为了读出隐含扇区的内容，应该使用硬盘 INT 13，按隐含扇区所在物理位置读出。

2、若只知道扇区号，用 INT 25 读出是十分方便的。例如，根据 FAT 查出了文件分布的逻辑扇区号，直接送入 DX 再调用 INT 25 即可。在这种情况下，若采用 INT 13，则需要把逻辑扇区号换算为物理扇区号。这牵扯到磁盘的规格和参数，十分麻烦，且易出错。因此，只有已知欲读扇区的物理位置，或必需读隐含扇区（如清除硬盘病毒）时，才使用 INT 13。

3、INT 25 的输入缓冲区段地址是用 DS 寄存器来指出。驱动器 0～1 号为软盘，2 号为硬盘，而 INT 13 是用 ES 寄存器指示缓存区段地址。驱动器 0～3 号为软盘，80～87H 为硬盘。这些细微差别都要特别注意。

4、软盘 INT 13 和硬盘 INT 13 尽管调用号都相

同，但在结构和功能上却有很大的差别。使用时，除了要仔细研究前面介绍的参数或有关资料外，还需要了解你使用的是哪一个 INT 13，否则很容易出错。为此，有必要在这里对机器的启动过程，概略地作一介绍。

机器启动后，执行 ROM BIOS；由子程序 CLR_STG（FE18EH）填入中断矢量。这时，在 0000：004C～004F 单元填入的是软盘 INT13 的入口地址 FEC59H。此后，ROM_SCAN 子程序（FE518H）检查有无硬盘 ROM 存在。若有，则转至硬盘 ROM（C800：0003）。在此，首先把软盘中断的入口地址值 FEC59H，转移到 INT 40 的位置 0000：0100～0103 单元；而用硬盘 INT 13 取代，把入口地址 C800：0256H 填入到 0000：004C～004FH 单元中。往后，软盘 I/O 功能由 INT 40 完成，软中断指令 INT 13 被用于硬盘 I/O 了。

但是，在系统有硬盘 ROM 的情况下，仍然可利用软中断指令 INT 13 来实现软盘 I/O 功能。因为，在硬盘 INT 13 子程序一开始，就检查 DL 中的盘号。若（DL）＜80H，则属于软盘 INT 13，转而调用 INT 40 指向 FEC59H，执行软盘 I/O 功能。否则，按硬盘 INT 13 处理。那就是说，用户在 DL 中置入的是软盘号，那末调用的就是硬盘 INT 13；置入的是硬盘号，调用的就是硬盘 INT 13。当然，其余参数也应根据置入的盘号，按前面介绍的范围来选用，否则将会出错。

结束语

在 IBM PC/XT 机中，软中断指令的实质是调用一组专用处理子程序。本文仅对其要点作了一些分析。具体使用中，还要考虑该子程序对某些寄存器及堆栈的破坏等因素。为了不发生错误，最好采取一些措施，如另设一用户堆栈，对受影响寄存器进行保护，有关程序用 DEBUG 验证一下等等。

文中仅对软中断读盘功能作了介绍。按理类推，对软中断写盘功能也就能够理解，故不再赘述。

使用设备驱动程序 ANSI. SYS 扩充屏幕和键盘

解放军兽医大学计算机教研室　曹长英　李杰

PC DOS 装载的标准设备驱动程序支持标准输入、标准输出、标准打印机、软盘以及硬盘设备，支持键盘和屏幕输出。系统提供的这些常规的屏幕和键盘控制功能是由常驻的‘CON’设备驱动程序实现的，然而它还有如下不足：

（1）每一键符定义是固定的，无法改变。

（2）屏幕显示类似电传打字机的输出，无法改变光标位置及显示的色彩等。

为此，还提供了一个扩展屏幕和键盘控制的驱动程序，命名为 ANSI. SYS 文件，附加在 DOS 系统盘上。DOS 支持这些设备不需要用户任何命令指定，而是在启动 DOS 时自动把它装入，方法是通过在设置文件 CONFIG. SYS 时，在其中加入一个 DEVICE＝＜命令＞这种形式指定的。在此即：DEVICE＝ANSI. SYS。

ANSI. SYS 文件除了保持原常驻的 CON 驱动程序的全部功能外，还扩展了如下的功能：

（1）随意指定光标位置。

（2）控制光标的移动。

（3）清除屏幕。

（4）指定屏幕显示规格。

（5）设置显示色彩参数。

（6）随意对任一键符赋予新的定义。

这些功能虽然只是直接控制显示器硬件功能（ROM BIOS 的 INT）的一部分，但能确保调用上述功能的应用程序可在所有运行 DOS 系统的机器上执行。

为使 ANSI. SYS 加载内存后，取代常驻的 CON 驱动程序，在编制 ANSI. SYS 程序的设备头时，其设备名字域内仍取“CON”名。这样，当 DOS 开工初始化期间，该驱动程序作为可安装的字符设备驱动程序，被连接在常驻的 CON 程序之前，成为设备链中仅次于空设备 NUL 的第一个字符设备驱动程序。当应用程序对键盘或屏幕请求一个 I/O 操作时，DOS 自然在设备链中，首先匹配有扩展功能的 CON 驱动程序。

下面讨论 ANSI. SYS 具备哪些扩展屏幕和键盘控制的功能。

在 DOS 版本 2 和 3 下，当通过“DEVICE＝ANSI. SYS”命令加载内存后，任意一个应用程序，在调用 INT 21H 的 01H、02H、06H 和 09H 等四个子功能时，可发出特殊的字符序列控制屏幕光标的位置、设置色彩参数以及对任一键符赋予新的含义。

特殊字符序列的格式如下：

ESC［#字符；---

其中：（1）ESC 是一个字符的代码（十六进制数是 1B，十进制数是 27，）而不是三个字符---“ESC”。

（2）ESC 之后是一字符串，以［为前导字符。

（3）［之后是＃，它表示一个数字参数，若是字符则必须用十进制数表示出其 ASCII 码值。

（4）参数之间用；分隔开。

光标控制功能：

ESC ＃；H　　设置光标位置

将光标移到参数所指定的位置。第一个＃指定行号，第二个＃指定列号。缺省值为 1，即让光标“回家”（左上角）。

ESC［＃A　　控制光标上移

光标上移#指定的行数，列位置不变。#缺省为1，即上移一行。若光标已在顶行，则不予理睬。

ESC [#B　　控制光标下移。

光标下移#指定的行数，列位置不变。#缺省为1，即下移一行。若光标已在底行，则不予理睬。

ESC [#C　　控制光标前移。

光标前移#指定的列数，行位置不变。#缺省为1，即前移一列。若光标已在最前列，则不予理睬。

ESC [#D　　控制光标后移。

光标后移#指定的列数，行位置不变。#缺省为1，即后移一列。若光标已在最后列，则不予理睬。

ESC [#;#f　　控制光标位置

光标移到参数指定的位置。第一个#指定行号，第二个#指定列号。缺省值为1，即让光标"回家"(左上角)。

ESC [#;#R　　报告光标位置

将当前光标位置通过标准输入设备报告DOS，第一个#指定行号，第二个#指定列号。

ESC [6n　　返回光标位置

ESC [s　　保存光标位置

ESC [u　　恢复光标位置

屏幕清除功能：

ESC [2j　　清除屏幕，光标"回家"(左上角)

ESC [k　　清除自光标位置开始到行尾的全部信息。

屏幕显示规格：

ESC [=#h　　设置屏幕显示规格　参数#：

0——40×25黑白；

1——40×25彩色；

2——80×25黑白；

3——80×25彩色；

4——320×200彩色；

5——320×200黑白；

6——640×200黑白；

7——在行尾处键入将产生一新行，不上移。

ESC[=#ll　　复位屏幕显示规格　参数#：

0～6同上

7--行尾不卷绕(丢掉超过行尾的字符)

图形显示控制：

ESC [#;…;#m　　设备图形显示的特征

在图形方式下，字符显示将具有由下述参数所指定的特征，直到有另一个控制序列改变。

参数#	含　义
0	全部特征关闭(常规的黑底白字)
1	粗体(高亮度)
4	下划线(仅适用单色显示)
5	闪烁
7	反转(白底黑字)
8	消隐(不可见)
30	黑字
31	红字
32	绿字
33	黄字
34	蓝字
35	深红色字
36	蓝绿色字
37	白字
40	黑底色
41	红底色
42	绿底色
43	黄底色
44	蓝底色
45	深红底色
46	蓝绿底色
47	白底色

键符的新定义控制：

ESC [#;#;#p　　赋予参数指定的键符新定义

第一个#指定替换的键符，其后的#定义替换后的新定义。若第一个#为0，则与其后的第二个#共同构成被替换的扩展ASCII键符。

例如：

ESC [65;81p　　表示A新定义为Q

ESC [0;68;97p　　表示F10的新定义为a

ESC [#;"STRING";…#p　　赋予参数#指定的键符以双引号内的字符串定义

或者"String"与#任意的组合

例如：

ESC [0;68;"dir";13P

该序列意味着F10定义为dir命令(带回车键)。

以上这些功能，用户可以自己编制一个外部命令文件或在批处理中使用，根据需要选用有关命令。在系统设置CONGIF.SYS文件中写上这条DEVICE=ANSI.SYS命令，并且保证根目录中含有ANSI.SYS文件，才能使系统支持一系列以ESC字符领头的字元组的特殊功能。

中文操作系统CCDOS2.0/2.1是在PC DOS 2.0/2.1的基础上扩充汉字输入和输出功能后形成的，键盘输入和屏幕和输出功能的扩充就是利用ANSI.SYS的控制特性。此命令文件非常重要，在实际使用中，我们碰到除了在长城本身所带的操作系统外，如果在CONFIG.SYS文件中不加入ANSI.SYS命令，如：(1)在2.13F系统结构中无此命令，则系统将无法启动，在启动的中途死机。(2)在286、0520CH等长城系列机的2.13D系统结构下调用长城系统结构下装入的dBASE管理系统，在修改程序文件存盘后，会出现屏幕不能向上滚动，数据总是在第25行显示的现象，给使用带来不便。相反如果在CONFIG.SYS中加入了该命令就很好地解决了这些问题。

我们可以根据需要，在实际中灵活使用设备驱动程序ANSI.SYS来扩充屏幕和键盘，弥补DOS命令的某些不足。

用 PC TOOLS 修改软件的汉字提示

山东济宁市煤气公司调度室　李开新

对于改变软件的提示信息一般用 DEBUG 进行修改，但对于一个不太懂 8088 汇编语言的人来说比较麻烦，且由于 DEBUG 本身的局限，不能修改 *.EXE 文件(或 *.HEX 文件)，需将后缀改变，才能修改。而用 PC TOOLS 修改则简而易行，而且可以直接修改 *.EXE 文件(或 *.HEX 文件)，下面以 CDBASE Ⅲ ver 1.0 为例说明如下：

DBASE 启动后的提示信息存在于 DBASE.OVL 文件中，执行 DBASE.EXE 文件时将其显示在屏幕上，其中有“山东省济宁市计算机实业公司软件部”。设欲将其改为：山东省济宁市煤气公司调度室李开新”，修改只对 DBASE.OVL 文件进行即可。具体修改如下：

一、查取“山东省济宁市计算机实业公司软件部”的机内码为：C9BD B6AB CAA1 BCC3 C4FE CAD0 BCC6 CBE3 BBFA CAB5 D2B5 B9AB CBBE C8ED BCFE B2BF，又查得“煤气公司调度室李开新”的机内码为：C3BA C6F8 B9AB CBBE B5F7 B6C8 CAD2 C0EE BFAA D0C2。查取汉字的机内码的方法有多种，可以编辑一文件，只含以上文字，根据汉字编码的高低字节最高位都为“1”的特点，用 PC TOOLS 的“F”命令查取；可以从国标区位码表查得；可以编程序或用其它方法查得。

二、启动 PC TOOLS，对 A 驱动器的软盘(含 DBASE.OVL 文件)操作，进入文件功能后，选择 DBASE.OVL 文件，按“F”键进入查找字符串模块，按“F1”键进入机内码模式，输入要查的汉字的机内码，键入回车后自动查找，查找结束后，按“E”键进入“观察编辑”，则在找到的机内码前显示闪烁的光标，然后用新的汉字机内码输入以覆盖原码。具体修改前后如下：

修改前显示

一……BCC6(计)CBE3(算)BBFA(机)CAB5(实)D2B5(业)B9AB(公)CBBE(司)C8ED(软)BCFE(件)B2BF(部)……

修改后显示：

——……C3BA(煤)C6F8(气)B9AB(公)CBBE(司)B5F7(调)B6C8(度)CAD2(室)C0EE(李)BFAA(开)D0C2(新)……修改结束后，按“F5”存盘，按“ESC”键退出 PC TOOLS。

三、运行 DBASE.EXE 文件，即可显示新的提示信息。

四、用此方法也可修改其它提示信息，或对其它西文软件的提示信息进行汉化。笔者已在 PC－286 机上对几个中西文软件进行提示的汉化，新用操作作系统为西山 CCDOS－4。PC TOOLS 版本为 DeluxeR4.3。

五、注意：修改时尽量使新的提示信息量少于原有信息量，余下的填“20”(空格)，以免误改操作码，引起文件执行错误。

巧用批命令定义功能键

福州大学计算机系 12901 信箱　邢观锋

在上机操作实践中，我们常常要用到一些使用频度较高的关键字符串，若每一次都一个个字符输入，势必增加击键次数，容易发生错误，降低了工作效率。如果能把功能键定义成一些使用频度较高的字串，无疑能提高效率，简化机时。本文介绍的方法是采用 DOS 的批文件功能，可直接在 DOS 提示符之下使用，只要具备磁盘操作系统基本知识的人就可运用。

步骤：先用 EDLIN 或其他文本编辑程序建立批文件 DY1.BAT

```
A>EDLIN DY1.BAT
1:*^[[0;59;"dir A:";13p
2:*^[[0;60;"dir B:";13p;"chkdsk";13p
```

然后执行 type dy1.bat，这时屏幕上没有显示任何文本内容，但是功能键 F1 已被定义成 dir A:，F2 已被定义成 dir B:chkdsk，这时按下 F1 即执行 dir A:功能，按下 F2 先执行 dir B:紧接着自动执行 chkdsk 命令。建立文本 dy1.bat 时应注意以下几点：

1.“^”字符是用 ^V(即 CTRL—V)键输入；

2.59,60 分别是 F1 和 F2 键的扩展 ASCII 码，引号内是要定义的内容，13 表示回车，p 表示此行结束(p 要小写)；

下面提供一段小程序，原理同上，由于采用了批文件中的哑参数，能任意地改变 F1～F9 功能键的定义。

```
A>EDLIN DY2.BAT
1:*ECHO ^[[0;59;"%1";13p
2:*ECHO ^[[0;60;"%2";13p
3:*ECHO ^[[0;61;"%3";13p
4:*ECHO ^[[0;62;"%4";13p
5:*ECHO ^[[0;63;"%5";13p
6:*ECHO ^[[0;64;"%6";13p
7:*ECHO ^[[0;65;"%7";13p
8:*ECHO ^[[0;66;"%8";13p
9:*ECHO ^[[0;67;"%9";13p
```

执行时只要键入 dy2，后面按顺序跟上要定义的内容，相互之间至少要一个空格。执行结果屏幕上也是不显示文本内容，但各功能键已被定义，若 dy2 后不跟任何定义内容，即可取消 F1～F9 原有的定义。

(注：以上所述方法在 CONFIG.SYS 中须有 DEVICE=ANSI.SYS 此条命令)。

CCDOS 下屏幕死锁的解决

桂林工商银行电子中心站　成刚　杨明秋

一、现象：

使用 CCDOS(如 CCDOS2.1,CCDOS3.0 等)时,有时会出现正常的数据信息(指非提示行的信息)掉进提示行(第 10 行)的情形。一旦掉进提示行,则屏幕的正常内容不再滚动,如 dir 列目录命令的所有显示都在最低行。每一行的输出都将前一行的内容覆盖掉。这种现象破坏了屏幕的正常显示,影响了用户的使用,我们把这种现象称之为屏幕死锁。

二、分析：

在 CCDOS 的显示(CRT)控制模块中,采用了实屏和虚屏两个概念。操作系统对实屏中的最末一行(第 10 行)进行了特殊的处理,使它成为提示行。它恒对应虚屏的最后一行(第 24 行)。系统对实屏内容进行滚动时,只滚动前面的 10 行(0～9 行),而不滚动第 10 行。所以,一旦信息进入第 10 行,就会造成屏幕死锁。

屏幕死锁的原因在于系统盘的根目录的系统配置文件 config.sys 中使用了配置命令 device＝ansi.sys,使得 ansi.sys 代替了系统的标准屏幕和键盘程序。

ansi.sys 是一个用来扩充屏幕及键盘功能的设备驱动程序,它把 25 行看作是屏幕的一页,即 0～24 行。这样,当显示内容进入到最后一行(第 24 行)时,就造成了屏幕死锁——前面我们讨论过,CCDOS 实屏的提示行(第 10 行)恒对应虚屏的第 24 行。

三、解决屏幕死锁的方法

解决屏幕死锁的方法有很多,下面简单谈几种。

①第一种方法：

最简单的方法是将文件 config.sys 中的 device＝ansi.sys 命令删去,即可消除屏幕死锁现象。这种方法的缺点是用户不能使用屏幕和键盘的扩充功能。

②第二种方法：

用 debug 命令对造成屏幕死锁的 ansi.sys 稍作修改。进行如下操作(下划线处为用户修改处)。

0278 处的指令 CMP BYTE PTR[0102],19 和 027F 处指令 MOV BYTE PTR[0102],18 决定了光标定位的范围是第 0～24(18H)行,现在我们把光标定位范围修改为 0～23(17H)行,那么显示的内容就不会进入最后一行(第 24 行),从而消除了屏幕死锁现象。

```
A>debug ansi.sys↙
-e 27c↙
-xxxx:027c 19.18↙
-e 283↙
-xxxx:0283 18.17↙
-w↙
-q↙
```

③第三种方法：

用 CCDOS 原配盘上的 ansi.sys 替换掉引起屏幕死锁的 ansi.sys 也可消除屏幕死锁。

实际上,CCDOS 原配置的 ansi.sys 也象第 2 种方法那样作了修改。

方法二和方法三既可消除屏幕死锁,又可使用户使用屏幕和键盘的扩充功能。

VDISK 的妙用

北京农业工程大学 244#信箱　叶舸

DOS3.0 以上版本提供了一个名为 VDISK.SYS 的虚拟磁盘驱动程序,从而使用户能自由地按自己需要建立适当大小的 RAM 虚拟盘。使用中发现,2.X 的 DOS 版本利用 3.0 所提供的驱动程序也可以使用 RAM 虚拟盘。

对双软驱无硬盘系统的用户,在从应用程序中退出时,常会碰到"Insert disk with COMMAND in drive A and strike any key when ready"这个提示。这时你不得不中断手头繁忙的工作,从也许是一大堆磁盘中找出系统盘。之后也许还得再将工作盘换入,这的确是一件让人头痛的事。

随着 DOS 版本的升级,系统文件在磁盘中所占容量也越来越大。DOS 3.0 已达约 58K,而 DOS3.3 与 CCDOS 汉字库已不能并存于同一张盘上。因此一般作法是除启动盘外通常将盘格式化为工作盘(即不含系统文件)。这样,前面所说情况更容易发生。

本人在上机实践中发现一种利用 RAM 虚拟盘巧妙地解决这一问题的方法,实现过程如下：

一、确认你的启动盘上有 VDISK.SYS 文件

二、建立或编辑该盘上 CONFIG.SYS 文件,使之包含如下一行：

DEVIDE＝VDISK.SYS,nk

其含义是设置一大小为 nk 的虚拟盘,视自己需要设置 n 大小。若使用本方法,建议 n 至少为 25K。

三、建立或编辑 AUTOEXEC.BAT 文件,使之包含如下两行：

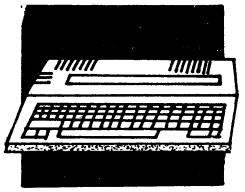
COPY COMMAND　COM C:(C:虚拟盘盘号)

SET COMSPEC＝C:COMMAND.COM(设置系统重新装入命令处理程序的路径为 C:)若所设置虚拟盘盘号不同,则需相应改动此两处。

用这个盘启动后,你就再也不会碰到要求插入系统盘的提示,当你从应用程序退出,不会再停顿,而是立即回到系统提示符下。

RAM 虚拟盘最大特点是高速度,其存取速度即内存存取速度,比磁盘操作快几十倍。因此,使用上述方

(下转 13 页)



学习机之友

谈 谈 FORTH

北京机工印刷厂　　丁志伟

FORTH 语言，从软件设计的角度看，比起 BASIC 来，更适合于中华学习机这样的八位机。鉴于人们普遍对它缺乏了解，下面用问答形式向大家介绍。

甲：你怎么想起要介绍 FORTH 语言来的？

乙：据了解，中华学习机已经生产了十几万台，加上苹果机及兼容机，还有各种以 Z80 为 CPU 的微机，我国八位微机的社会拥有量相当大。近年来 16 位 PC 机越来越便宜，八位机面临被淘汰的境地。这么大一笔社会财富要被浪费，真是太可惜了。我们还没富裕到如此迅速地对微机更新换代的程度，应该充分发挥这些微机的效能，延长其生命周期。

甲：这和你要介绍的 FORTH 语言有什么关系？

乙：我先问你，你使用的是中华学习机，又有一定应用经验，开发应用程序时用什么语言呢？

甲：一般用 BASIC，如果要求速度高，或对某些特殊要求的场合，就用汇编。大多数人都如此，你呢？

乙：过去我也和你一样。BASIC 一般还可以用，但它毕竟是初学者语言，对已经跨过初学门槛的人来说，常会感到不足。比如说速度慢，程序长而变量较多时，编程和排错相当麻烦，维护和修改也较困难。汇编就更麻烦了，设计周期长，对编程者要求也很高。

甲：你说得不错。那 PASCAL、FORTRAN、LOGO 等语言是否会好些？

乙：当然也能用。可惜受机器内存、速度限制，八位机上这些语言的版本都不完善，使用者很少。

甲：不论用什么语言，在八位机上编一些大程序，常有捉襟见肘，顾此失彼之感，总不知出路在哪里。

乙：我过去也有这种感觉，直到找到 FORTH，问题才基本解决了。这是一个值得推荐的语言。

甲：真的吗？你先告诉我，FORTH 是什么意思？

乙：查克·摩尔(Churk Moore)是美国的一个程序设计师。1969 年，他在美国佛吉尼亚国立无线电天文台工作时发明了这种语言，用来控制巨大的天文望远镜，当时在第三代计算机 IBM1130 上运行。由于这个语言具有一系列不同寻常的特点，效果又非常好，被命名为 FOURTH，也就是英文第四代(计算机语言)的意思。1130 机上规定标识符只能用 5 个字符，于是去掉了 U，就成了 FORTH。顺便说一句，由于这个成功，FORTH 被国际天文学会定为天文台自动化的标准计算机语言。

甲：噢，是这么回事。那这个语言都有哪些不同寻常的特点呢？

乙：FORTH 是面向堆栈的可扩展的语言系统。大多数操作都是通过堆栈进行的，很少或不用变量就能对数据进行操作。运行时占内存少，速度快，由于通过堆栈进行操作，程序和表达式都是用一种特殊的被称为逆波兰的表达法表示的。表达式不用括号，不必考虑优先级。系统在结构上采用链式结构，又被称为词典，各功能块通过链表被串连在一起，功能块被称为动词，相当于其他语言中的语句、函数或命令。这样的结构非常易于扩展和压缩。

甲：你说的扩展是怎么回事？

乙：不管用什么语言编程，经常会遇到这种情况，所需要的功能，语言系统并没提供，这时你怎么办？

甲：这一般要通过编程解决。有些功能实现起来很麻烦，常要剖析所使用的语言系统，和原有系统协调也不太容易。不过好在《电子与电脑》经常登一些这方面的小程序。当然，有时用一两个扩展功能还行，多用几个就难了，可能会发生冲突。

乙：出现这种情况的原因在于：不论一个语言版本功能多么强，总不可能面面俱到，适合每个用户的各种需要。而且这些语言系统本身结构上比较复杂，多不对用户开放，不提供或很少提供扩展功能，靠打补丁式的扩展方法难免顾此失彼，更何况众多扩展功能的作者互不通气。

甲：那 FORTH 是怎么解决这个难题的呢？

乙：FORTH 遇到不够时，通常也是靠编程来解决。不同的是，FORTH 有一套扩展机制，程序经编译后就成为系统的一部分，是向词典增添一个新动词。

甲：新词和原有系统关系怎么样。能协调一致吗？

乙：只要编程时别出错，这个新词就好象是系统原来就具备的一样，它也能象系统中其他动词一样被调用，地位完全相同。其实系统本身很多动词就是用扩展的方法编出来的，如果对原有系统不满意，也可以重新定义有关动词，我自己就时常这么做。

甲：很有意思。这样的新词能定义多少呢？

乙：只要内存够用，想编多少都行。实际应用中可以看做没有限制。反过来，也可以删除一些不需要的动词，以便使系统保持在适当的规模上，FORTH 系统是能够拼接裁剪的。

甲：是否可以认为，扩展是通过编程进行的？

乙：不错，编程和扩展可看成一回事。通过编程，可以把系统逐步改造成符合自己需要和习惯的样子。这样的系统，主动权在使用者手里，越用就越得心应手。从这个角度看，FORTH 具有一些“学习”能力。

甲：真没想到还有这种编程语言。有这种能力，FORTH 是否很复杂呢？

乙：恰恰相反，FORTH 的功能是建立在简明的基

础之上的，结构简单，也不难理解。

甲：占内存大吗？

乙：不论是系统本身，还是源程序，或编译好的动词都相当短。一般情况下比 BASIC、PASCAL 等语言短得多，甚至经常比汇编还短。这是 FORTH 的特点之一。我用的版本，包括各种基本功能、编辑、磁盘存取等，有 160 多个词，还不到 8K 字节。后来我进行了一些改造，增加了不少基本功能，还增加了虚拟磁盘，反编译等，也只有不到 12K 字节。其中很多动词都是编程辅助工具，不是必需的，还可以删除。

甲：这对内存紧张的八位机来说，是相当理想的。

乙：据介绍，某些应用的目标程序，可以少于 1K 字节，由于占内存少，整个语言中的解释、编译、编辑、磁盘存取、汇编等功能可以同时常驻内存，成为一体。除了源程序，很少用得着磁盘存取。

甲：在编程、调试时一定很方便。

乙：确实，我对此深有体会。

甲：那速度怎么样？

乙：FORTH 速度快是公认的。我用的版本，在中华学习机上比 BASIC 一般要快 2～5 倍。对于要求速度高的场合，还可以用 FORTH 中的汇编来编动词。编译速度也相当快，其他高级语言根本无法与之相比。

甲：用 FORTH 编程是否也要快些？

乙：有人称 FORTH 是天然的结构化语言，比 PASCAL 的结构化水平还高，非常适合于软件工程学倡导的自顶向下的程序设计方法。具体编程时，用系统扩充的方法，每个动词相当于子程序，都可以独立调试，编程和调试都较简单。子程序可以通用，编制程序当然要快得多。我使用 FORTH 一年多，除了编一些小程序外，还对系统做了不少改造、扩充，编了 6502 高级反汇编、FORTH 反编译、虚拟磁盘等较大的程序。用 BASIC 来编，这种进度是不可能的。

甲：真是太迷人了，FORTH 好学吗？

乙：开始有些难。对于初学者，由于 FORTH 用逆波兰表示法，要有一个适应过程。较高水平的应用则要求对硬件系统有所了解和掌握汇编语言。跨过开始了门槛，随着对系统的适应，加上对系统的扩展、改造，就会越来越容易。怎么样，想学吗？

甲：当然想学，我以后有问题还要来麻烦你。

乙：别客气，有问题尽管来问。

甲：谢谢你介绍这么多，再见！

乙：再见！

再谈特殊 6502 指令

浙江金华二中　李永前

特殊 6502 指令，笔者最早见于《最新苹果尔机加解密大全》(关永健著)，于是一直在寻找 6502 指令的规律。以后在各杂志上相继见到几篇关于特殊 6502 指令的文章，但本刊 91 年第 2 期刊登的 6502 指令表最完整最有规律。对此表笔者再作如下三点补充：

1. 表①②③中现有指令(除两个保留外)，均为普通 6502 指令；表④中的指令均为特殊 6502 指令。

2. 在各表中尚有许多空格，能填上的还有：表③中第 000 列 000～011 行及 100 列 000～111 行的十二个空格均为死机。表①中 000 列 100 行；表②中 010 列 100 行；表③中 000 列 100、110、111 行的五个空格均为 NOP。其余空格还有待各位验证后填上。

3. 表①中的 000 列 001 行应为 JSR $nnnn；表④中的 000 列 110 行应为 CMP($nn,x)。

特殊 6502 指令可用于软件加密。一个特殊指令通常包含两个普通指令，而执行时间却与一个普通指令执行时间相等。因此如能很好地加以利用，将能起到缩短程序，提高运行速度的功效。为此笔者编写了一个高分辩第一页快速清屏的程序，以供参考。

```
0300-   A2 00      LDX   # $ 00
0302-   A9 00      LDA   # $ 00
0304-   9D 00 3F   STA   # 3F00,X
0307-   E8         INX
0308-   D0 FA      BNE   $ 0304
030A-   A9 1F      LDA   # $ 1F
030C-              CF    ???
                   06    ASL
        030D-      03
$ 03               D0    BNE
        030F-      F1
$ 0302             60    RTS
        0311-
```

030C～030D 为特殊 6502 指令，相当于普通指令：

DEC $ 0306

CMP $ 0306

本程序比同类程序减少了三个字节，速度加快了 48 个节拍。

(上接 12 页)

法还有一优点：当你使用目前一些流行软件均提供的功能 OS Shell(如 BASIC3.0，Turbo 系列)时，也需读入 A 盘上 COMMAND 文件，一方面，若 A 盘中无该文件则出错而使操作失败；另一方面读入至少需 3－5 秒的时间。而在使用本方法后，既能保证正确执行，并且在一瞬间即可完成，非常便利，充分发挥软件的功能。

另外，当程序需进行大量磁盘读写操作时，将磁盘读写转向 RAM 盘，可大大提高效率。如本人自编的一个汉化工具软件，需对目标文件进行逐字节读取，以提取英文提示。当目标文件较大时，速度慢得让人难以忍受。改进后方法为先将目标文件拷入虚拟盘(相当于一次完成所有的磁盘操作)，再对虚拟盘中文件进行读写。实测表明，在原方法下需 6 分钟读取完成的目标文件在改进方法下仅需 40 多秒，速度提高近十倍。

扩展监控系统命令

蚌埠医学院　蒋建一

中华学习机的监控系统，有一个提供给用户的监控扩展命令(CTRL-Y 命令)。每当键入该命令，监控程序便会跳转到内存第三页的 $3F8 单元去执行，通常 $3F8 处存有一条 JMP $FF69 指令返回监控系统，若我们重新设置该处的跳转地址，使其转向自己编制的程序入口，便可获得监控系统的控制权。利用这一特点，我编制了几个新的监控命令，以弥补原监控命令的某些不足。新增的监控命令及其功能简介如下：

一、CTRL-Y 命令。在监控系统下键入 CTRL-Y↙ 可进入小汇编状态；在小汇编状态下键入 $CTRL-Y↙ 便可退回到监控系统。

二、CTRL-YK 命令。同时以十六进制数和 ASCII 字符两种形式显示内存中的内容，格式如下：

1：〈地址〉CTRL-YK 显示指定地址单元内容。

2：〈地址 1〉·〈地址 2〉CTRL－YK 显示从地址 1 到地址 2 间的全部内存单元内容。

3：·〈地址 2〉CTRL-YK 显示从当前地址到地址 2 之间的内容。

4：CTRL-YK 显示当前地址到该行行尾内容。

若与 1CTRL-P 联用，例如 1 CTRL-P〈地址 1〉·〈地址 2〉CTRL-YK 所显内容可输出到打印机。

三、〈地址 1〉·〈地址 2〉CTRL-YL 命令。将内存地址 1 到地址 2 中的机器指令反汇编并以双列输出到打印机。该命令自动联打印机，并能自动分页打印。按空格键可开始或暂停打印。当打印的程序多于 2 页，按 P 键可使其打印完一页后暂停，等待换纸或调整纸等。按 ESC 键可终止打印。

四、〈地址 1〉<〈地址 2〉·〈地址 3〉CTRL－YM 命令。该命令功能为在搬移内存中机器码程序的同时能够自动修正其绝对转移地址等。

在键入该程序前，须首先做好如下工作：

1. *AA57:2↙设置 DOS 文件缓冲区个数为 2。

2. *9D00:80　9A↙将 DOS 缓冲区文件名栏地址改到 $9A80。

3. *A7D4G↙　重新设定 DOS 文件缓冲区。

然后再按照程序清单正确键入全部程序，并以 MON(*^YKLM)为名存盘。本程序放在 DOS 的一号缓冲区中，其首址为 A$9AA6，长度为 L$237。为了方便调用我们可以用一个 BASIC 程序来设置并装入监控扩展程序。以后只要开机运行一次这个 BASIC 程序，便会自动设置好其扩展入口，并将程序保护起来。这时我们便可以进入监控，方便地选用各扩展命令了。

```
20  POKE 43607,2:POKE 40192,128:POKE 40193,154
30  CALL 42964
40  PRINT CHR$(4);"BLOAD MON(*^YKLM)"
50  POKE 1017,166:POKE 1018,154
```

```
9AA6— A4 34
9AA8— B9 00 02 C9 CB F0 1E C9
9AB0— CC F0 14 C9 CD F0 6E C9
9AB8— 8D F0 03 4C 65 FF A5 33
9AC0— C9 AA D0 06 4C 50 D3 20
9AC8— 86 9B 4C 56 D3 C0 02 B0
9AD0— 19 A5 3C 85 3A 09 07 85
9AD8— 3E A5 3D 85 3B 85 3F 90
9AE0— 09 A5 3C 29 07 D0 0D 20
9AE8— 0A 9B A2 01 20 78 FE 20
9AF0— 92 FD 84 47 A9 A0 20 ED
9AF8— FD B1 3C 20 DA FD E6 47
9B00— 20 BA FC 90 DC 20 0A 9B
AB08— F0 79 A9 1F 85 24 B1 3A
9B10— 09 80 C9 FF F0 04 C9 A0
9B18— B0 02 A9 AE 20 ED FD C8
9B20— C4 47 D0 EA 60 A0 00 A5
9B28— 42 85 3A A5 43 85 3B 20
9B30— 2C FE 38 A5 40 E5 44 85
9B38— FE A5 41 E5 45 85 FF A2
9B40— 00 20 8C F8 A5 2F C9 02
9B48— D0 28 A0 01 B1 3A C5 3C
9B50— C8 B1 3A E5 3D B0 1B 88
9B58— B1 3A C5 40 C8 B1 3A E5
9B60— 41 90 0F A0 01 B1 3A E5
9B68— FE 91 3A C8 B1 3A E5 FF
9B70— 91 3A 20 53 F9 85 3A 84
9B78— 3B A5 3A C5 42 A5 3B E5
9B80— 43 90 BC 4C 69 FF A5 3E
9B88— 85 18 A5 3F 85 19 A9 36
9B90— 0A 85 1E A2 FF 86 1B A2
9B98— 4B 86 1C 20 A4 9B A2 00
9BA0— 86 1B 86 1C 09 80 85 1F
9BA8— A5 3C A4 3D 85 3A 84 3B
9BB0— E0 00 D0 03 20 67 9C 85
9BB8— 06 84 07 20 67 9C A5 1A
9BC0— D0 01 60 4A 85 1D 90 02
9BC8— E6 1D A5 06 A4 07 85 3A
9BD0— 84 3B 20 97 9C C6 1D D0
9BD8— F9 85 08 84 09 E6 1B E6
9BE0— 1B 20 DD FB 20 58 FC 20
9BE8— 80 F2 A2 05 BD CD 9C 20
9BF0— 5C DB CA D0 F7 A6 1B A9
9BF8— 00 20 24 ED 20 73 F2 24
9C00— 1F 10 07 29 7F 85 1F 20
9C08— C0 9C A9 01 20 95 FE 20
9C10— 8E FD A6 1C 20 50 F9 A9
9C18— AE 20 ED FD A6 1B A9 00
9C20— 20 24 ED A9 AE 20 ED FD
9C28— 20 8E FD 20 8E FD A5 06
9C30— A4 07 20 83 9C 85 06 84
9C38— 07 C6 1A F0 1B A2 26 86
9C40— 24 A2 02 20 50 F9 A5 08
9C48— A4 09 20 83 9C 85 08 84
9C50— 09 20 A4 9C C6 1A D0 D3
9C58— A9 8C 20 ED FD 20 93 FE
9C60— A5 3A A4 3B 4C B4 9B A2
9C68— 00 86 1A C4 19 90 08 D0
9C70— 11 C5 18 90 02 D0 0B 20
9C78— 97 9C E6 1A A6 1A E4 1E
9C80— D0 E9 60 20 9F 9C A6 3A
9C88— 20 99 FD 20 48 F9 20 8C
9C90— F8 20 D3 F8 4C 9C 9C A2
9C98— 00 20 8C F8 20 53 F9 85
9CA0— 3A 84 3B 60 AD 00 C0 10
9CA8— 23 2C 10 C0 C9 9B D0 06
9CB0— 20 93 FE 4C 65 FF C9 D0
C9B8— D0 02 85 1F C9 A0 D0 0C
9CC0— AD 00 C0 10 FB 2C 10 C0
9CC8— C9 A0 D0 F4 60 BA C5 C7
9CD0— C1 D0 00
```

MZ—731 机绘制的财务计划执行分析报表

广东　花县宾馆　　黄浩珍

夏普 MZ—731 微机的四色绘图打印机可输出 80 列字符。我用 BASIC 语言编了这个程序，用它来做财务计划执行情况分析报告，提高工作效率很多，对于及时提供分析资料，指导经济活动，有一定的实用价值。

本程序只要在在 DATA 语句中置入本单位各个部门本期实际发生的营业额和利润，以及对应的计划数、去年同期数，就会自动完成汇总和求出与财务计划执行分析有关的百分比，并绘制出下列图表：

（一），《本期计划执行报告表》。分别列出各单位本期实现的营业和利润的计划数、完成数、完成计划的百分比和利润率。

（二），《总额比重图》。以园形百分图描出各部门营业额与利润额在总额中所占的比重。

（三），《同期比较图》。以直方图绘出各部门营业额、利润额与去年同期变化的比较。

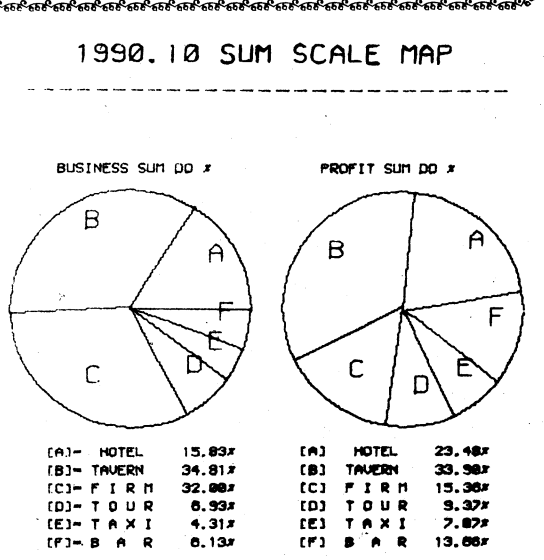
本程序用模块式的结构编制，每执行完一项功能后，都先回到主菜单程序，所以功能是任选的，既可以只打印某一个表或图；也可以重复打印某一个表或图。如把单位名称改为月份，在数据置入中把 B$ 改为 1—N 月，就可成为《期中计划执行情况累计表》；同理，如置入 1—12 月份的数据，就可以编制出本单位的《全年分月计划执行情况报告表》。另外，绘图机所绘出的图形比例精确，用放大尺放后，可作为办公室的悬挂图表。

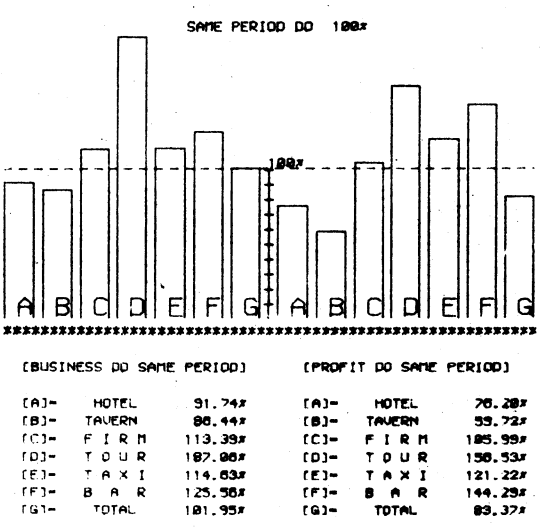
本程序头段为计算程序。(1)为主菜单程序；(2)为打印《计划执行报告表》程序；(3)为绘制《总额比重图》程序；(4)为绘制《同期比较图》程序；(5)为打印星状线条子程序；末段为数据。

为使图面完整合理，某一单位发生亏损时，《总额比重图》的利润部份会自动剔除该单位不计，《同期比较图》会将其置零，但注释部份仍标明其实际数。

为便于校对及保全资料，数据统一用 READ 读入。主程序(10—1300 行)与数据应分别存带。数据部份统一用 3000 行号开始，以时间作注释及存带文件名称，先在 3010 行号的 DATA 中置入 B$(时间)和单位数(N)。从 3020 行号开始，依次置入表列各数。每一部门占用一个行号，校对好后先存带。使用时先 LOAD 载入数据文件，再用“ME・”指令连接主程序运行。

```
10 REM ---PLAN CARRY REPORT---
20 READ B$,N:M=N+1:Z=6:W=1:C$"----------
--------------------"
30 IF N>=10 THEN Z=4:W=0
40 DIM A$(M),I$(M),A1(M),A2(M),A3(M),A4(M),B1
(M),B2(M),B3(M),B4(M),B5(M)
50 FOR I=1 TO N:I$(I)=CHR$(64+I)
60 READ A$(I),A1(I),A2(I),A3,B1(I),B2(I),B3
```






```
70 A3(I)=A2(I)/A1(I)*100
80 A4(I)=A2(I)/A3*100
90 B3(I)=B2(I)/B1(I)*100
100 B4(I)=B2(I)/B3*100
110 B5(I)=B2(I)/A2(I)*100
120 A=A+A1(I):B=B+A2(I):C=C+A3
130 D=D+B1(I):E=E+B2(I):F=F+B3
140 NEXT:RESTORE
```

```
150 A$(M)=" TOTAL ":I$(M)=CHR$(64+M)
160 A1(M)=A:A2(M)=B:A3(M)=B/A*100:A4(M)=B/
C*100
170 B1(M)=D:B2(M)=E:B3(M)=E/D*100:B4(M)=E/F
*100:B5(M)=E/B*100
180' ---[1]---
190 PRINT"  ":PRINTTAB(10)"MENU PRO-
GRAM!":PRINT
200 PRINT,"[1] TOTAL LIST":PRINT
210 PRINT,"[2] SCALE MAP":PRINT
220 PRINT,"[3] SAME PERIOD COMPARE MAP":PRINT
230 PRINT,"[4] END":PRINT
240 PRINT,"          PLEASE  SELECT KEY";
250 INPUT K:IF K>4 THEN 250
260 ON K GOTO 270,500,920,1280
270 ' ---[2]---
280 PRINT"  ";:PRINT,"THE PRINTER
LIST!":PCOLOR 1
290 PRINT/PTAB(7)B$;" PLAN CARRY REPORT":PCOL-
OR2
300 PRINT/PTAB(8)C$:GOSUB1300:PCOLOR1:PRINT/P
310 PRINT/PTAB(21)"B U S I N E S S ";:PRINT/PTAB(51)
"P R O F I T";:PRINT/PTAB(72)"PROFI-":PRINT/P"
NO  N A M E"
320 PRINT/PTAB(15)"  PLAN      FULFILL
DO  %      PLAN      FULFILL    DO  %:;PRINT/
PTAB(72)"TRATE%"
330 GOSUB1300:PRINT/P:PCOLOR0
340 FOR I=1 TO M
350 PRINT/P" [";I$(I);"]";
360 PRINT/PTAB(6)A$(I);
370 PRINT/PTAB(14)A1(I);
380 PRINT/PTAB(25);A2(I);
390 PRINT/PTAB(35)" ";
400 PRINT/PUSING"###。##";A3(I);
410 PRINT/PTAB(45)B1(I);
420 PRINT/PTAB(55)B2(I);
430 PRINT/PTAB(63)"  ";
440 PRINT/PUSING"###。##%";B3(I);
450 PRINT/PTAB(70)"  ";
460 PRINT/PUSING"###.##%";B5(I)
470 PRINT/P:NEXT I
480 GOSUB1300:PCOLOR0
490 MODE TN:PRINT/P:PRINT/P:GOTO 180
500 ' ---[3]---
510 PRINT"  ":PCOLOR1
520 PRNT,"THE SUM SCALE MAP!"
530 PRINT/PTAB(9);B$;" SUM SCALE MAP"
540 PCOLOR3:PRINT/PTAB(8);C$:PRINT/P
550 PRINT/P" BUSINESS SUM DO %";
560 PRINT/PTAB(22)"PROFIT SUM DO %"
570 PCOLOR1:B=0
580 MODE GR:MOVE 120,-100:HSET
590 CIRCLE 0,0,100,0,360,5:PHOME
600 FOR I=1 TO N
610 A=A1(I)/A1(M)*360:B=B+A:C=B-A8.5
620 X=100*COS(B*PAI(1/180))
630 Y=100*SIN(B*PAI(1/180))
640 LINE 0,0,X,Y
650 X1=60*COS(C*PAI(1/180))
660 Y1=60*SIN(C*PAI(1/180))
670 MOVE 0,0,X1,Y1:GPRINT[W,0],I$(I)
680 PHOME:NEXT
690 PCOLOR3:B=0:V=B1(M)
700 FOR I=1 TO N:IF B1(I)<0 THEN V=V-B1(I):NEXT
710 MOVE 240,0:HSET
720 CIRCLE 0,0,100,0,360,5:PHOME
730 FOR I=1 TO N
740 IF B2(I)<0 THEN 820
750 A=B2(I)/V*360:B=B+A:C=B-A*.5
760 X=100*COS(B*PAI(1/180))
770 Y=100*SIN(B*PAI(1/180))
780 LINE 0,0,X,Y
790 X1=60*COS(C*PAI(1/180))
800 Y1=60*SIN(C*PAI(1/180))
810 MOVE 0,0,X1,Y1:GPRINT [W,0],I$(I)
820 PHOME:NEXT
830 MOVE -360,-130:MODE TS:PCOLOR2
840 FOR I=1 TO N:A=A2(I)/A2(M)*100
850 B=B2(I)/V*100:IF B<0 THEN B=0
860 PRINT/PTAB(10)"[";I$(I)"]=  ";A$(I);
870 PRINT/PUSING"  ###。##%";A;
880 PRINT/PTAB(50)"[";I$(I)"]  ";A$(I);
890 PRINT/PUSING"  ###。##%";B:NEXT
900 GOSUB1300
910 PCOLOR0:MODE TN:PRINT/P:GOTO180
920 ' ---[4]---
930 PRINT"  ":PRINT/P:PCOLOR1
940 PRINTTAB(6)"THE SAME PERIOD COMPARE MAP!"
950 PRINT/PTAB(7)B$;" SAME PERIOD COMPARE"
960 PCOLOR2:PRINT/PTAB(8)C$
970 PCOLOR3:MODE TS:PRINT/PTAB(30)"SAME PERIOD
 DO  100%"
980 MODE GR:MOVE0,-80:HSET:L=240
990 LINE %5,480,0:PHOME:PCOLOR1
1000 MOVE 0,-100:HSET
1010 FOR I=1 TO M
1020 X=A4(I):Y=L/M-Z:Y1=Y*.5
1030 LINE %1,0,X,Y,X,Y,0
1040 MOVE Y1,0:GPRINT [W,0],I$(I)
1050 MOVE Z+Y,0:HSET:NEXT
1060 PCOLOR3:AXIS0,10,10
1070 GPRINT [0,0],"100%":PHOME
1080 PCOLOR2:MOVE Z,0:HSET
1090 FOR I=1 TO M:S=B4(I)
1100 IF S<0 THEN S=0
1110 IF S>200 THEN S=200
1120 X=S:Y=L/M-Z:Y1=Y*.5
1130 LINE 0,X,Y,X,Y,0:MOVE Y1,0
```

(下转 17 页)

机器语言程序计算阶乘

北京大学物理系新生　易鹤龄

最近我在某电脑杂志上看到一个用 BASIC 编写的求阶乘精确值的程序，运行后觉得运算速度太慢。算 100 的阶乘需 36 秒，500 的阶乘用了 19 多分钟。如果算 1000 的阶乘，不等上一、二个钟头是不会有结果的。于是我用机器语言编了一个程序，计算 100 的阶乘约 2 秒，500 的阶乘不过 1 分钟，算 1000 的阶乘也只花了 5 分 8 秒，速度之快是 BASIC 程序望尘莫及的。

此程序输入内存后，在 BASIC 状态下键入 CALL7696 或在监控状态输入 1E10G 后，会出现提示光标。这时输入一个小于 10000 的数，程序就会算出该数的阶乘。如果输入大于 10000 或在输入中加带了字母，程序会自动排错。程序计算完一次后，会重新出现提示光标，准备下一次输入。如果想退出此程序，只需在出现提示光标时直接按回车键。

程序中 1FA3～1FA4，1FBB～1FC2 是为打印输出整齐设置的。如需打印，请先键入 POKE1657，80 把打印机设置成 80 列，如果只需屏幕显示，将上述地址内容均改为 EA 即可。

```
1E10— A9 01 C9 8D D0 0B A5 FC
1E18— D0 04 A5 FD F0 03 20 FF
1E20— 1E A9 00 85 FC 85 FD 20
1E28— 6F FD AD 00 02 C9 8D D0
1E30— 01 60 A2 00 BD 00 02 C9
1E38— BA B0 D7 C9 B0 90 D3 E9
1E40— B0 0A 0A 0A 0A A0 04 0A
1E48— 26 FC 26 FD 88 D0 F8 E8
1E50— 4C 34 1E A5 FC 29 F0 F8
1E58— F0 3B 4A 4A 4A 4A AA A9
1E60— 00 85 15 85 14 18 A5 16
1E68— 65 14 85 14 90 06 A9 00
1E70— 65 15 85 15 CA D0 EF A2
1E78— 04 06 14 26 15 CA D0 F9
1E80— 18 A5 10 65 14 85 10 A5
1E88— 11 65 15 85 11 90 06 A9
1E90— 00 65 12 85 11 90 06 A9
1E98— 0F F0 19 AA 18 A5 16 65
1EA0— 10 85 10 90 0C A9 00 65
1EA8— 11 85 11 A9 00 65 12 85
1EB0— 12 CA D0 E9 D8 60 A0 00
1EB8— 84 12 84 11 84 10 B1 06
1EC0— 85 16 A9 FC 8D 54 1E 8D
1EC8— 96 1E 20 53 1E A5 10 91
1ED0— 06 A5 11 85 10 A5 12 85
1ED8— 11 A9 00 85 12 A5 FD F0
1EE0— 0B A9 FD 8D 54 1E 8D 96
1EE8— 1E 20 53 1E E6 06 D0 02
1EF0— E6 07 A5 09 C5 07 D0 C6
1EF8— A5 08 C5 06 D0 C0 60 A5
1F00— FD 8D 09 1E A5 FC 8D 08
1F08— 1E AD 39 1F 85 07 A0 00
1F10— 84 06 A9 00 91 06 E6 06
1F18— D0 FA E6 07 A6 07 E0 70
1F20— D0 F2 AD 39 1F 85 07 85
1F28— 09 85 FF A9 00 85 06 85
1F30— FE A9 03 85 08 A9 01 8D
1F38— 00 20 A5 FE 85 06 A5 FF
1F40— 85 07 20 B6 1E C6 09 A0
1F48— FE B1 08 F0 09 E6 08 D0
1F50— F8 E6 09 4C 49 1F E6 09
1F58— A0 00 B1 FE D0 06 E6 FE
1F60— D0 02 E6 FF F8 18 A9 99
1F68— 65 FC 85 FC A9 99 65 FD
1F70— 85 FD D8 A5 FD D0 04 A5
1F78— FC F0 03 4C 3A 1F C6 09
1F80— A9 FD 18 65 08 85 08 A9
1F88— 00 65 09 85 09 AD 09 1E
1F90— 20 DA FD AD 08 1E 20 DA
1F98— FD A9 A1 20 ED FD A9 BD
1FA0— 20 ED FD A2 24 A0 00 B1
1FA8— 08 20 DA FD A5 08 D0 09
1FB0— A5 09 CD 39 1F F0 0F C6
1FB8— 09 C6 08 CA D0 E7 20 8E
1FC0— FD A2 27 4C A5 1F 20 8E
1FC8— FD 4C 8E FD
```

（上接 16 页）

```
1140 GPRINT [W,0],I$(I);MOVE Z+Y,0
1150 HSET:NEXT
1160 GOSUB1300:PCOLOR0:PRINT/P
1170 PRINT/PTAB(7)"[BUSINESS DO SAME PERIOD]";
1180 PRINT/PTAB(48)"PROFIT DO SAME PERIOD]":
PCOLOR1:PRINT/P
1190 FOR I=1 TO M
1200 PRINT/PTAB(10)"[";I$(I);"]= ";
1210 PRINT/PA$(I);"      ";
1220 PRINT/PUSING"###.##%";A4(I);
1230 PRINT/PTAB(50)"[";I$(I);"]=    ";
1240 PRINT/PA$(I);"      ";
1250 PRINT/PUSING"###.##%";B4(I):NEXT
1260 PRINT/P:GOSUB1300:MODE TN:PCOLOR0
1270 PRINT/P:PRINT/P:GOTO 180
1280 CLS:END
1290 ' ---[5]---
1300 PCOLOR3:MODE TS:PRINT/P
1310 FOR I=1 TO 80:PRINT/P"*";:NEXTI
1320 PRINT/P:RETURN
3000 ' ---90.10 DATA---
3010 DATA "1990.10",6
3020 DATA " HOTEL ", 280000, 275405, 300194, 25200,
23546,30902
3030 DATA "TAVERN ", 600000, 605489, 700456, 33000,
34005,56940
3040 DATA "F I R M", 550000, 556566, 490840, 16500,
15409,14538
3050 DATA"T O U R",120000,120454,64393,7200,9398,
6004
3060 DATA"T A X I",80000,74936,65373,6400,7096,5854
3070 DATA"B A R",100000,106539,84848,12000,13705,
9498
```

（上接 16 页）

ProDOS 磁盘操作系统入门(续)

廖　凯

第三章　ProDOS 命令

ProDOS 增加了 7 个新命令，改进了 DOS3.3 的 15 个命令，取消了 DOS3.3 的 6 个命令。

ProDOS 的大部分命令可在立即执行方式下使用，少数命令只能在程序方式下使用。在程序方式下使用 ProDOS 命令的方法与 DOS3.3 一样，其格式为：PRINT CHR $(4);"ProDOS 命令"。

如果你的计算机配置的是新监控 ROM，可用大小写字母输入所有 ProDOS 命令和 APPLESOFT 语句。

DOS3.3 所使用的槽口号及驱动器号，在 ProDOS 中已被卷名所代替。当你要访问某个磁盘时，只要给出此磁盘卷名，ProDOS 就会自动地在所有的驱动器上寻找此盘。槽口号及驱动器号在 ProDOS 上仍可使用。

在下面的命令格式中，带方括号的参数在使用时可要可不要，根据自己的需要而定。在格式中的参数可以用十进制或十六进制数表示，十六进制数必须以"$"表示。在下面的命令中，除特别说明外，均可在立即执行方式和程序调用方式下使用。

一、ProDOS 新的命令

1. CAT 和 CATALOG

格式：CAT 路径名[,S#][,D#]
　　CAT/卷名/子目录/子目录
　　CATALOG 路径名[,S#][,D#]
　　CATALOG/卷名/子目录/子目录

功能：CAT 以 40 列显示方式简单地显示磁盘目录，CATALOG 是以 80 列显示方式详细地显示磁盘目录。所显示的各项目的含义如下：

/××××：卷名(即磁盘名)。

NAME：文件名．在文件名前面的星号表示已上锁。

TYPE：文件类型．ProDOS 主要有以下几种类型：

文件类型	类型符号
APPLESOFT 程序	BAS
APPLESOFT 变量文件	VAR
二进制文件	BIN
目录文件	DIR
ProDOS 系统文件	SYS
可移动代码文件	REL
文本文件	TXT
用户自定义文件	$F#

其它文件类型也可能存在，有些是针对 APPLEIII 而言的，有些是由 INTEGER BASIC 或 INTEGER 变量转换而来的(ProDOS 不提供 INTEGER BASIC)，INTEGER BASIC 会有一个 INT 文件类型，INTEGER 变量会有一个 IVR 文件类型。

BLOCKS：表示一个文件在磁盘上占用的空间量，单位是块。ProDOS 以逻辑块代替磁道及扇区，一个逻辑块相当于 DOS3.3 的两个扇区，为 512 个字节，ProDOS 将磁盘划分为 280 个块。

MODIFIED 和 CREATED：这些项目表示修改文件的最后日期和时间，以及建立文件的日期和时间。用户可以通过计时卡设置时间或者用系统盘设置时间。

ENDFILE：ProDOS 分配给文件的最大字节数。

SUBTYPE：表示一个文本文件的长度或二进制文件的起始地址。

A：表示二进制文件的起始地址。

R：表示文本文件所占用的记录长度(用字节表示)。如果 R＝0，那么这个文件是一个随机存取文件。

BLOCKS FREE：显示磁盘上未用的块数。

BLOCKS USED：显示磁盘上已使用的块数。

TOTAL BLOCKS：显示磁盘上的总块数。

在列子目录时，用户可以先设置部首(PREFIX)再 CAT 或 CATALOG。在列目录时，可用 CTRL－S 键暂停列目录。

2. CREATE

格式：CREATE 路径名[,T 类型][,S#][,D#]
　　CREATE/卷名/子目录
　　CREATE/卷名/子目录/文件名,T 类型

功能：此命令用于产生不同的文件类型。ProDOS 允许用户建立一个目录或子目录文件。参数"T 类型"表示要产生的文件类型，"类型"以 3 个字母符号表示。

3. FLUSH

格式：FLUSH [路径名]
　　FLUSH/卷名/子目录/文本文件名

功能：当资料要写入一个文本文件时，FLUSH 会使程序每次都将缓冲区内的资料写入磁盘。ProDOS 在资料写入磁盘之前，先将资料存储在 512 字节的缓冲区内。如果程序在未写好这些资料前令系统失去控制，这些资料便会失去。使用 FLUSH 命令可确保你的程序的安全。虽然 WRITE 命令也可以将资料写入磁盘，但这命令会使磁盘存取程序的速度减慢。在使用 FLUSH 而未给文件名时，将使所有打开着的文件被 FLUSH。

4. PREFIX

格式：PREFIX

PREFIX,D#

PREFIX 路径名[,S#][,D#]

PREFIX/卷名/子目录/

功能:此命令是用于定义一个路径名。路径名可以指定为目录或子目录,使它较方便地与一个目录内的不同文件工作。当单独使用 PREFIX(无参数)时,ProDOS 会送回当前的路径名。如果只有 S# 和 D# 参数,PREFIX 会设置成此驱动器内磁盘的名字。如果用户在定义路径名而在结尾没加上斜线时,ProDOS 会自动加上,使系统保持完整性。

5. STORE 和 RESTORE

格式: STORE 路径名[,S#][,D#]

STORE/卷名/子目录/BASIC 文件名

RESTORE 路径名[,S#][,D#]

RESTORE/卷名/子目录/BASIC 文件名

功能:STORE 命令是将目前的程序所产生的变量值(要产生变量值,必须先执行程序)存入磁盘。STORE 命令将每个变量的值和名称保存在一个 VAR 型文件内。对于存储少量的变量值,STORE 命令执行的速度会比在程序调用下的 WRITE 命令快。而存储大量的变量值,则需要相当长的时间。

RESTORE 命令是将 VAR 型文件内的变量值去代替目前的变量的值。

6. —(短横线)

格式:—路径名[,S#][,D#]

—/卷名/子目录/文件名

功能:"—"命令可以将一个 BASIC 程序或机器语言程序或可执行的文本文件装入内存并执行。它可代替 RUN,BRUN 和 EXEC 命令。它可运行的文件类型有 BAS、BIN、SYS 和 TXT。

二、ProDOS 改进的命令

所有改进的命令可以被用在与 DOS3.3 相同的环境中,在转换 DOS 程序到 ProDOS 时,这些命令不需要改变。

1. APPEND

格式:APPEND 路径名[,L#][,S#][,D#]

APPEND/卷名/子目录/文本文件名

功能:此命令用于增添资料到文本文件的末尾。APPEND 先打开一个文件,将指针定位到文件的末尾,然后写入资料。如果用于随机存取文件,那么记录将被加到最后逻辑记录的后面。此命令只能用在程序方式下。

L# 参数表示在文件内的一个记录的长度,并被用在 APPEND 一个随机文件时。如果记录长度(L#)与文件被建立时一样,那么新文件将连接在旧文件的末尾。当 L# 是一个不同的长度时,ProDOS 会根据记录长度来划分 ENDFILE,剩余部分是当前记录的偏移量。ProDOS 使用如下公式计算 ENDFILE:

ENDFILE=偏移量+记录长度+1

2. BLOAD

格式:

BLOAD 路径名[,A#][,B#][,L#/,E#][,T 类型][,S#][,D#]

BLOAD/卷名/子目录/文件名,A#,T 类型

功能:此命令可以用"T 类型"参数从磁盘装入任何类型文件到内存的任意位置。如果没有"T 类型"参数,那么 ProDOS 将装入一个 BIN 类型文件。BLOAD 允许用户装入高分辨率图形到内存的图形页。高分辨率图形第一页的首地址为 $2000,第二页为 $4000。

各参数含义如下:

A#:指定文件的首起字节装入内存的起始起址。

B#:是指要从文件内第几个字节开始装入。如果没用这参数,那么就是从文件的首起字节开始装入。

L#/E#:是用于装入全部或部分的文件到内存。L# 指定文件要被装入到内存的字节长度,E# 指定文件要被装入到内存的最后地址。

T 类型:指定文件类型,以 3 个字母符号表示。

3. BRUN

格式:

BRUN 路径名[,A#][,B#][,L#/E#][,S#][,D#]

BRUN/卷名/子目录/文件名,A#

功能:此命令是先装入一个类型为 BIN 的二进制文件,然后运行这程序。ProDOS 无法区别图形文件与二进制数据,如果一个图形文件被运行,那么会改变 ProDOS 在内存的位置,如果发生这种情况,就需重新启动系统。

其参数含义与 BLOAD 相同,只不过 BRUN 除了要装入内存外还要运行。如果没使用 A# 参数,那么程序将被装入内存的地址与 BSAVE 时的起始起址一致。

4. BSAVE

格式:

BSAVE 路径名,A#,L#/,E#[,B#][,T 类型][,S#][,D#]

BSAVE/卷名/子目录/文件名,A#,L#/,E#

功能:此命令是用于将内存中的二进制数据存放到磁盘上。用户可以将内存任何位置的数据存放到任何类型的文件上,并可从文件内的任何位置开始存放。用户也可以保存高分辨率图形文件,供以后调用。

A#:此参数指明了要存储的数据的首起字节在内存中的起始地址。使用 BSAVE 时,必须使用此参数。

L#/E#:L# 指定要存放到磁盘的字节长度。E# 指定将被存储的数据在内存的最后地址。在使用 BSAVE 时,必须使用两个参数的其中一个。

B#:指出将被存储的数据从文件内的第几个字节开始存放。如果没用此参数,那么数据是从文件内的首起字节开始存放。

〈未完待续〉

用堆栈思想处理任意N个元素的排列组合问题

吉林市化学工业公司　第一中学　　索素文

引题：有3个人，从事三种不同工作。问：如何分配可得到最大效益？

这是一个固定N个元素的全排列问题，可用三重循环进行模拟，其中循环变量I,J,K分别代表三种不同工作；数组元素A(I),B(J),C(K)分别代表三个人各自从事每种工作时的效益。程序如下：

```
10  FOR  I=1  TO 3:READ A(I),B(I),C(I):NEXT I
20  FOR I=1 TO 3:FOR J=1 TO 3
40  IF I=J THEN 100
50  FOR K=1 TO 3
60  IF K=I OR K=J THEN 90
70  T=A(I)+B(J)+C(K)
80  IF T>T1 THEN I1=I:J1=J:K1=K:T1=T
90  NEXT K
100  NEXT J,I
110  PRINT "A--";I1,"B--";J1,"C--";K1
120  PRINT "T-----";T1
140  DATA 9个给定数据
```

思考：若将题中的3改为N(N不固定)，由于无法预知是几个元素，就无法用多重循环模拟。下题的引入可以帮助我们找到新方法。

引题2：对任意自然数N，打印出这前N个自然数的所有不同数字的排列来。

这是一个元素由自然数1～N构成的，特殊的N个不同元素的全排列问题。我们重新回顾一下循环的意义：多重循环的思想就是计算机中的堆栈思想：最先进入最外层循环，最后进入最内层循环；而在退出时，却是最先退出最内层循环，最后退出最外层循环。对于此题，可直接利用设置栈指针的方法。程序如下：

```
10  INPUT N:DIM A(N):T=1
30  IF T=0 THEN END
40  A(T)=A(T)+1:IF A(T))N THEN A(T)=0:T=T-1:GOTO 30
50  FOR I=0 TO T-1:IF A(I)=A(T) THEN 40
60  NEXT I
70  IF T<N THEN T=T+1:GOTO 40
80  FOR I=1 TO T:PRINT A(I);" ";:NEXT:PRINT:GOTO 40
```

程序说明：T指针模拟各循环层的走向；A(T)指针模拟每一循环层循环变量的取值(T,A(T)=1,2,……,M)。

20句，第一次进入最外层循环。

30句，判断是否已退出最外层循环。

40句，不断改变当前循环层循环变量的取值，即执行某一层循环体。

50句，相当于完成题1中各循环层间的IF…THEN…语句功能。

70句，从外循环依次进入内循环

80句，输出。

由解法我们看到：利用T和A(T)两个指针可以完成N重循环。这不仅解决了由于N不固定给使用多重循环带来的困难，也克服了不同机器对循环嵌套层数的约束。这样，引题1后面提出的问题即可有解。

例1：N个人从事N种不同工作，如何分配可得到最大效益？

分析：因N不固定，不能象引题1用三个一维数组表示效益。规定T——第T个人，A(T)——第A(T)项工作，引入二维数组B(N,N)，其中B(T,A(T))表示第T个人从事第A(T)种工作时的效益。程序：

```
10  INPUT N:DIM A(N),B(N,N),A1(N)
30  FOR I=1 TO N:FOR J=1 TO N:READ B(I,J):NEXT J,I
40  T=1
50  IF T=0 THEN 140
60  A(T)=A(T)+1:IF A(T)>N THEN A(T)=0:T=T-1:GOTO 50
70  FOR I=0 TO T-1:FI A(I)=A(T) THEN 60
80  NEXT I
90  IF T<N THEN T=T+1:GOTO 60
95  S=0
100  FOR I=1 TO N:S=S+B(I,A(I)):NEXT I
110  IF S<=S1 THEN 130
120 FOR I=1 TO N:A1(I)=A(I):NEXT I
125  S1=S
130  GOTO 60
140  FOR I=1 TO N
150  PRINT I;"--";A1(I)
160  NEXT I
170  PRINT S1
190  DATA N*N个数据
```

说明：40～100句产生一种工作分配方案，其中：T——第T个人，A(T)——第T个人从事第A(T)项工作；110～125句比较保留当前效益和最大时的方案；130句产生新分配方案；140～170句输出最佳方案。

例2：N皇后问题推广：在一个M×L格的棋盘上，摆N个皇后，使它们不能两两相吃，找出所有解来，(皇后的吃法是可吃同行、同列、同对角线上的子)。

分析：这是一个平面问题。所谓一个皇后在棋盘上的某一位置，是指它位于棋盘上第I行，第J列。我们可以分两步作这个题，第一步：在M行中任选出N行

不动；第二步在L列中任选N列与第一步所选的N行作各种不同的组合(即对所选的N列作排列)。之后，按题目要求将满足条件的各点打印出来。这里要搞清两点：1、选排例与全排列的区别；2、组合问题和排列问题的区别。对于1，区别在于选出元素的个数N≤总计元素个数M。修改引题2：

```
10  INPUT M,N
…  ……
40  A(T)=A(T)+1:IF A(T)>M THEN……
…  ……
70  IF T<N THEN T=T+1:GOTO 40
```

对于2，区别在于组合无序而排列有序，即同一组元素，不同顺序表示不同排列，而不同顺序与组合无关。根据这一特点修改引题2就可变为组合问题：

```
50  FOR I=0 TO T-1:IF A(T)≤A(I) THEN 40
```

由上分析给出皇后程序如下：

```
10  INPUT M,L,N:DIM A(M),B(L)
15  IF N>M OR N>L THEN 10
20  T=1
30  IF T=0 THEN END
40  A(T)=A(T)+1:IF A(T))M THEN A(T)=0:T=T
-1:GOTO 30
50  FOR I=0 TO T-1
55  IF A(T)<=A(I) THEN 40
60  NEXT I
70  IF T<N THEN T=T+1:GOTO 40
80  K=1
90  IF K=0 THEN 40
100  B(K)=B(K)+1:IF B(K)>L THEN B(K)=0:K=
K-1:GOTO 90
110  FOR I=0 TO K-1
120  IF I=0 THEN 150
130  IF B(I)=B(K) OR ABS(A(I)-A(K))=ABS(B(I)
-B(K)) THEN 100
140  GOTO 160
150  IF B(K)=B(I) THEN 100
160  NEXT I
170  IF K<N THEN K=K+1:GOTO 100
180  P=P+1:PRINT"N0:";P
190  FOR I=1 TO N
200  PRINT A(I);"--";B(I);"  ";
210  NEXT I:PRINT:GOTO 100
```

程序说明：A(I)——行元素，B(I)——列元素。15句，保证在棋盘内任选N点；20～70句在M行中任取N行；80～170句在L列中任取N列进行排列，其中146句保证皇后在不同列且不同对角线；120句和150句保证皇后在边界时处于不同列；180～210句输出符合条件的摆法。

由此可见，正确、深刻理解引题2对于处理任意N个元素的排列、组合问题是非常重要的，同时对于任意重循环问题，也可以利用此法解决。

有兴趣的读者可试解下列两题。

1、有一个表达式abcdef/ghi=1988，其中a～i这9个字母代表1～9中的数字，且这些数字各不相同。请编程序打印出这9个字母分别代表的数字。

2、用1与2两种数字写N位数，打印出其中所有任意相邻的两位不全是1的N位数。

简单有趣的解密办法

唐山白玉瓷厂　　张亭

BASIC程序防止被LIST有多种方法，有一种在REM或DATA语句中埋设CTRL－D的手段是十分有效的，它可以使LIST产生错误信息而被中断，也可以连上PR＃6命令重新热启动。甚至能连上INIT命令暗伏杀机。迄今尚未见到对这一手段的破解办法。

其实，这种保护手段的弱点是十分明显的。它必须依靠DOS才能实现，只要输出不经过DOS就会失效。在标准DOS3.3中，只要执行一次CALL40672就能达到这个目的。这样做使系统脱离了DOS，LIST已不会有任何后顾之忧。我们可用CTRL－RESET重接DOS。

这全部操作过程未免繁琐，而且对于不同版本的DOS，如存放在16K扩展RAM中的DOS就不通用。下面介绍一个更简便，并且十分有趣的办法：

```
]:PR#0:LIST↙
```

或者列在打印机上：

```
]:PR#1:LIST↙(假设打印卡插入1#槽)
```

PR＃命令前面加上冒号，是为了区别于DOS的PR＃命令而作为BASIC命令来执行。这样一来输出过程就"瞒过"过了DOS，而LIST执行完之后，控制权又自动归还了DOS，BAISC对打印机的接通与否DOS也概不认帐，因为那都是"背着它干的。"

(上接第24页)

```
            printf("mean=%d",mean(s));
        }
mean(a)
int a[];
{int i;
long s=0l;
for(i=0,i<MAX;i++)
{s+=a[i];
}
s/=MAX;
return((int)s);
}
A>EXP7-10
mean=16371
```

程序中出现的库函数srand()和rand()源型包含在stdio.h中，用法和功能分别是：

void srand(unsigned seed) 初始化随机发生器。

int rand(void)产生随机数，返回值为0～$2^{15}-1$，返回周期为2^{32}。　　(待续)

C 语言初阶讲座

第七讲　结构数据(一)

李文兵

二讲介绍了基本数据类型，这里介绍结构数据。这包括列举型、数组型、结构型、共体型和指针型。

1. **列举**(enum)

列举型也叫枚举型，其数据结构定义形式如下：

enum　列举类型名　{列举表}　变量表

列举类型名和变量表是任选项。例子如下所示：

```
enum   color   {red,green,blue};
enum   color   tv;
```

这里 color 是列举类型名，tv 为该列举类型的一个变量。对于列举变量，可以进行如下赋值和比较运算。

```
tv=red;
if(tv==green) printf("tv color's green\n");
```

这时，若打印变量 tv 的值，即：

```
printf("%d %d %d",red,green,blue);
```

则输出为 0、1、2。可见，列举型数据为整型常量。其列举表中各个标识符不仅能按 0、1、2…取值，而且能设定为任何整型常量。对于上例，若按如下初始化：

```
enum color {red=100,green,blue};
```

则 red、green、blue 就分别对应 100、101、102。

列举型变量可用于控制结构中，如在 for 结构中：

```
main()
{enum color tv;
  for(tv=red;tv<=blue;tv++)
{ ⋮
}
}
```

又如用于 switch 结构中：

```
switch(tv)
{   case red:语句 1;break;
  case green:语句 2;break;
  case blue:语句 3;break;
  default:语句 4;
}
```

巧妙地运用 enum 型结构数据，可使程序易读易懂，如练习 7.1 所示。

```
A>TYPE EXP7-1.C
main()
  {enum color{red,blue,green} c_char;
  /* enum type test 1 */
  printf("%d %d %d\n",red,blue,green);
  /* enum type test 2 */
  c_char=red;
  if(c_char==red) printf("RED\n");
  /* enum type test 3 */
  for(c_char=red;c_char<=green;c_char++)
    printf("%d  ",c_char);
  /* enum type test 4 */
  for(c_char=red;c_char<=green;c_char++)
    {switch(c_char)
      {case red:
        printf("\nred\n");
        break;
      case blue:
        printf("blue\n");
        break;
      case green:
        printf("green\n");
        break;
      }
    }
    printf("program end.\n");
  }

A>EXP7-1
0  1  2
RED
0  1  2
red
blue
green
program end.
```

2　**数组**

所谓数组在 C 语言中是指同一类型数据的集合。Turbo C 能定义任何维数的数组。

(1)一维数组　一维数组的定义格式为：

数据类型　数组名[尺寸]；

例如：char name[16]；

就定义了一个一维数组，其每个元素都是字符型数据，一共有 16 个这样的元素，变量名为 name。注意，该数组的 16 个元素是从 name[0]开始，到 name[15]为止的。实例见[练习 7.2]

```
A>TYPE EXP7-2. C
  main()
    { char ch[10];
        int i;
        for (i=0;i<10;i++)
          {ch[i]='A'+i;
            printf("%c ",ch[i]);
          }
    }
A>EXP7-2
ABCDEFGHIJ
```

使用 scanf()函数给字符型数组输入一个字符串的程序，如练习 7.3 所示。

```
A>TYPE EXP7—3. C
  main()
          {char name[30];
          printf("What is your name:");
          scanf("%s",name);
          printf("Hello,%s\n",name);
         }
A>EXP7—3
What is your name:WANG
Hello,WANG
```

注意，由于 name 为数组名，其值是数组本身的地址，因此，当作 scanf()函数的参数时，其前面不需要取址运算符 &。

(2)二维数组　其定义格式为：

数据类型　数组名[尺寸 1][尺寸 2]；

例如：int mat[5][5]；

就定义了一个 2 维整型数组；其行数为 5，列数亦为 5，共有 25 个元素；每个元素都是整型。如图 7.1。

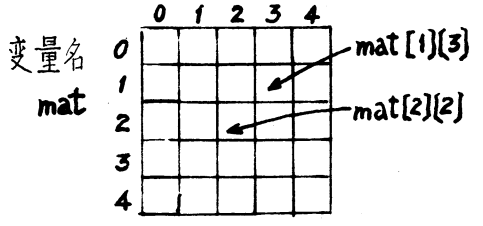


图 7.1　2 维数组的元素

二维数组赋值和输出的程序，如练习 7.4 所示。

```
A>TYPE EXP7—4.C
  main()
        { int row=5,column=5,i,j;
        int mat[5][5];
        for(i=0;i<row;i++)
            {for(j=0;j<column;j++)
                {mat[i][j]=i*row+j;
                  printf("%2d  ",mat[i][j];
                  }
                  printf("\n");
            }
        }
C>EXP7—4
   0    1    2    3    4
   5    6    7    8    9
  10   11   12   13   14
  15   16   17   18   19
  20   21   22   23   24
```

若对练习 7.4 进行修改，使程序执行结果为：

```
 0  1  2  3  4        sum=10       accum=10
 5  6  7  8  9        sum=35       accum=45
10 11 12 13 14        sum=60       accum=105
15 16 17 18 19        sum=85       accum=190
20 21 22 23 24        sum=110      accum=300
```

则程序如练习 7.5 所示。

```
A>TYPE EXP7—5.C
  main()
    {int mat[5][5];
      int row=5,column=5, i,j;
      for(i=0;i<row;i++)
          {for(j=0,j<column;j++)
              {mat[i][j]=i*row+j;
                printf("%2d  ",mat[i][j]);
                }
                  sum(mat[i]);
          }
      }
      sum(v)
      int v[];
      {int i,ss=0,column=5;
        static int sss=0;
        for(i=0;i<column;i++)
            ss+=v[i];
            sss+=ss;
            printf("sum=%3d accum=%3d\n",ss,sss);
      }
```

该程序中，语句 sum(mat[i])是函数调用，在函数 sum()中引进实参 mat[i]，以计算 i 行元素之和和截止到 i 行的累加和。可见，二维数组的每一行可作为一维数组使用。

(3)三维数组　其定义格式为：

数据类型　数组名[尺寸 1][尺寸 2][尺寸 3]；

例如：int arry[3][4][5]；

就定义了一个 3 维数组，它是由 3×4×5=60 个元素组成的数组，如图 7·2 所示：

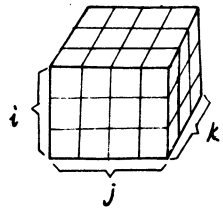


图 7.2　3 维数组[i][j][k]的元素

3 维数组的程序如练习 7.6 所示。

```
A>TYPE EXP7—6.C
  main()
      { int m=5,i,j,k;
        int arry[5][5][5];
        for(i=0;i<m;i++)
          {for(j=0;j<m;j++)
            {for(k=0;k<m;k++)
              {arry[i][j][k]=i*m*m+j*m+k;
              printf("%5d  ",arry[i][j][k]);
              }
              printf("\n");
            }
          }
```

```
    }
A>EXP7-6
   0    1    2    3    4
   5    6    7    8    9
  10   11   12   13   14
  15   16   17   18   19
  20   21   22   23   24
  25   26   27   28   29
  30   31   32   33   34
  35   36   37   38   39
  40   41   42   43   44
  45   46   47   48   49
  50   51   52   53   54
  55   56   57   58   59
  60   61   62   63   64
  65   66   67   68   69
  70   71   72   73   74
  75   76   77   78   79
  80   81   82   83   84
  85   86   87   88   89
  90   91   92   93   94
  95   96   97   98   99
 100  101  102  103  104
 105  106  107  108  109
 110  111  112  113  114
 115  116  117  118  119
 120  121  122  123  124
```

(4)数组的初始化　前面我们分别介绍了1～3维数组的定义格式，以此类推，可以定义任何维数的数组。这里，我们介绍一下数组的初始化方法。

①1维数组可按如下进行初始化：

```
int arry[2]={0,1};
```

②2维数组可按如下进行初始化：

```
int arry[2][2]={{1,0},
                {0,1}};
```

练习7.7所示的初始化方法也是允许的。

```
A>TYPE EXP7-7.C
  int max=4;
  int arry[4][4]={{1},
                  {1,1},
                  {1,1,1},
                  {1,1,1,1}
                  };
   main()
   {  int i,j;
      for(i=0;i<max;i++)
         {for(j=0;j<max;j++)
           printf("%2d ",arry[i][j]);
         printf("\n");
         }
   }
A>EXP7-7.C
 1  0  0  0
 1  1  0  0
 1  1  1  0
 1  1  1  1
```

在2维数组的定义中，行数是可以缺省的，如练习7.8所示。

```
A>TYPE EXP7-8.C
  main()
    { int max=5;
      int arry[][5]={{1},
                     {1,1},
                     {1,1,1},
                     {1,1,1,1},
                     {1,1,1,1,1}};
                   int i,j;
                   for (i=0;i<max;i++)
                       {for(j=0;j<max;j++)
                           printf("%2d ",arry[i][j]);
                       printf("\n");
                       }
        }
A<EXP7-8
  1  0  0  0  0
  1  1  0  0  0
  1  1  1  0  0
  1  1  1  1  0
  1  1  1  1  1
```

可以看出，2维数组arry是按5行5列处理的。

字符型数组可用双引号引起来的字串进行初始化，如练习7.9所示。

```
A>TYPE EXP7-9.C
main()
    { char a1[]="array of character";
      char a2[5]="char";
      printf("%s\n %s",a1,a2);
    }
A>EXP7-9
  array of character
  char
```

程序中，数组a1可初始化为任意长的字符串；而数组a2包括字符串结尾符\0，只限5个字符。

(5)数组参数　C语言中，数组可以做函数的参数，可使函数调用多个同类型的数据，如练习7.10。

```
A>TYPE EXP7-10.C
  #include"time.h"
  #include <stdio.h>
  #include "stdlib.h"
  #define MAX 10000
  main()
     { int i,s[MAX];
       long n;
       srand(time(&n));
       for(i=0;i<MAX;i++)
          s[i]=rand();
```

(下转21页)

学用单片机

单片机汉字打印驱动程序的设计思想

苏州信息技术研究所　　尹　征

本文以苏州信息技术研究所开发的 DHX－I 单片机汉字系统为例，介绍了以 8031 单片机为核心(CPU)的系统的打印机接口和打印驱动程序的设计思想。该方法主要解决了汉字在 9 针打印机上(如：CP－80 打印机)的打印输出，并且实现了汉字与 ASCII 码的同时打印。

一、DHX－I 单片机汉字系统

DHX－I 单片机汉字系统是一个只需由用户开发应用软件既可成为用户专用系统的系统，有以下几个部分组成：

1. CUP：INTEL 公司 51 系列 8031 单片机；

2. 存储器：64KB EPROM 含 16KB 操作系统；
64KB RAM 并可切换成打印字库；

3. 显示器：5.5 英寸 320 线，文本为 24＊8 汉字；

4. CRT 接口板：8255 并行接口芯片，2048 个显示字库；

5. I/O 接口：打印机接口，四个并行接口，RS232 串行通信接口；

6. 键盘：共 20 个按键(包括 12 个数字键，7 个功能键，1 个回车键)；

7. 软件：监控程序、打印驱动程序、屏幕编辑程序、数据处理程序、汉字造字程序等；

二 CP－80 打印机

CP－80 打印机是一种经济实惠的 9 针打印机，每秒可打印 80 个字符。每个字符由 9＊9 点阵组成。当以标准字符尺寸输出时，每行可打印 80 字符(或 132 个紧凑字符)。当以“宽体”字符尺寸输出时，每行可打印 40 字符(或 66 个紧凑字符)。CP－80 打印机还具有图形打印功能，通过软件送出一定的控制字符，可以实现图形和汉字的打印。

不同的打印机都有不同的输出字符集。CP－80 打印机可以打印出 96 个标准的 ASCII 字符和 128 个图形字符，另外它还可以处理和执行若干个控制字符。所谓控制字符是指那些能被打印机识别并使打印机执行某种特定功能的字符，一般它并不被打印出来，所以我们也称之为“不可打印字符”(有关控制字符的作用，在设计打印程序时是非常有用的，因篇幅所限，不再刊出，请参阅有关 CP－80 操作手册)。

三、打印机接口的设计

DHX－I 单片机汉字系统打印机接口采用的是标准的 Centronic 并行接口，它可与各种 9 针打印机联接。如 CP－80、CP－135、SP－1000、MX－80、MX－100、FX－100、FX－80 等打印机。接口信号共有 36 条，其中主要使用的信号如下表所示：

编号	信号名	方向	功　能　说　明	
1	$\overline{\text{STROBE}}$	IN	以 $\overline{\text{STROBE}}$ 脉冲读入数据，在接收端的脉冲宽度必须＞0.5μs	(选通)
2	DATA1	IN	这些信号分别表示第 1 位到第 8 位的并行数据信息。且各信号在数据为逻辑为“1”时为高电平，逻辑“0”时为低电平。	(数据)
3	DATA2	IN		
4	DATA3	IN		
5	DATA4	IN		
6	DATA5	IN		
7	DATA6	IN		
8	DATA7	IN		
9	DATA8	IN		
10	$\overline{\text{ACK}}$	OUT	约为 7.6μs 的脉冲，低电平表示数据已接收，准备接收新数。	(回答)
11	BUSY	OUT	高电平表示打印机不能接收数据，低电平时才能接收数据。	(“忙”)
12	PE	OUT	高电平信号表示打印机缺纸。	(缺纸)

从上表可以看出接口信号包括数据选通信号 $\overline{\text{STROBE}}$、打印机回答信号 $\overline{\text{ACK}}$、打印机“忙”信号 BUSY 和 8 根据信号 DATA1～DATA8 等。其时序图如图 1 所示：

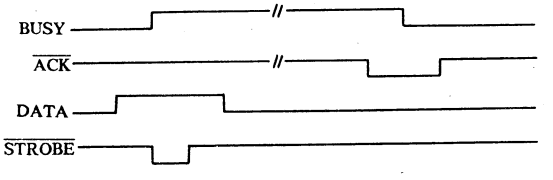


图 1

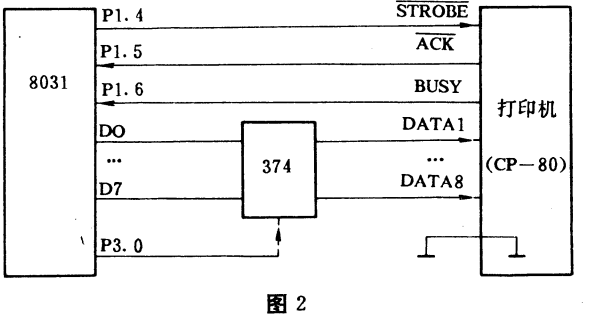


图 2

打印机接口的硬件使用了 8031 的四个口线 P1.4、P1.5、P1.6、P3.0，由软件执行口操作来完成与打印机的同步及数据传送。打印机接口硬件如图 2 所示。

四、打印机驱动程序的设计

DHX－I 单片机汉字系统打印驱动程序的主要功能是完成字符和汉字的打印输出。这是整个系统中除了汉字显示之外关于汉字处理的又一个重要环节。实现汉字打印输出的过程，实际上就是把由汉字分区、代码(二字节)构成的一组汉字机内码的集合转换为打印纸上的汉字图形的过程。各种 9 针打印机的控制字符可能有些不同，但这个过程总是一致的，其主要有以下五个步骤：

第一，从打印数据缓冲区 0(1600H－17FFH)中逐个取出预先存放在该区域内的需要打印输出的汉字文本的机内码即分区、代码。

第二，根据取出的机内码逐个计算出横向汉字点阵的首址并取出其点阵。

第三，把横向汉字点阵转换为可供打印的纵向汉字点阵并送入打印数据缓冲区 1 和 2。

第四，把打印数据缓冲区 1(1800H－1BFFH)的数据送打印机输出，即打印上半个汉字，然后 7/72"回车换行，以便能够打印下半个汉字。

第五，把打印数据缓冲区 2(1C00H－1FFFH)的数据送打印机输出，即打印下半个汉字，然后回车，以便接着能够打印 ASCII 码。

由于 9 针打印机一次只能打 9 个点，所以对 16＊16 汉字点阵的打印是分二次来完成的。汉字打印输出的流程图如图 3。

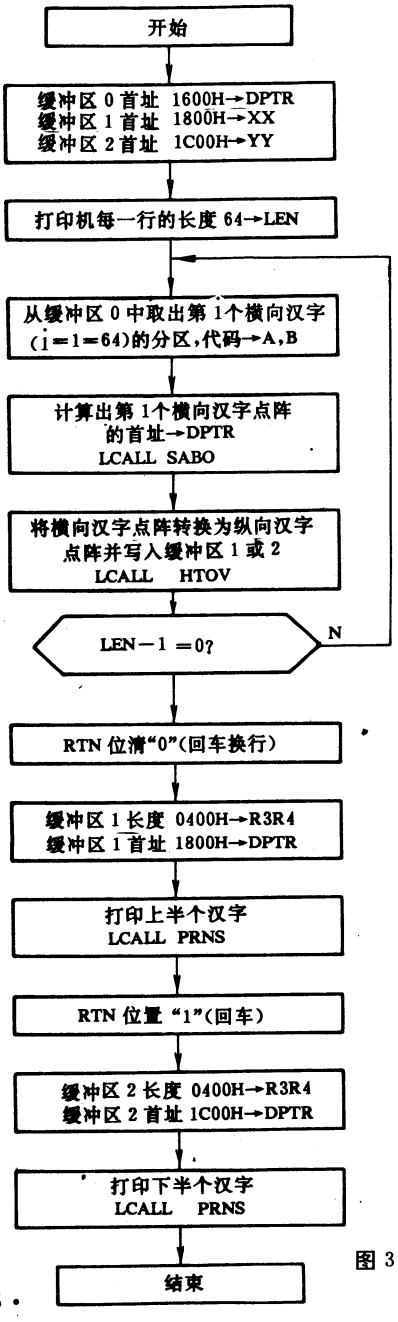


图 3

从以上的流程图中我们已经发现有许多模块都是通过调用不同的子程序来实现的，这些子程序是整个打印驱动程序的核心。下面，就围绕这些子程序对打印驱动程序作进一步的阐述。

1. 输出字符子程序 PIOC

本程序首先打开打印数据端口，把输出字符(数据)送到打印机的数据端口上，这里的输出字符可以是 ASCII 码(如：A、1、%……)，也可以是控制字符(如：回车、换行、换码……)。然后循环测试 BUSY("忙")信号，如果打印机忙，则继续测试；如果打印机闲，则发出 $\overline{\text{STROBE}}$(选通)信号，选通打印机，此时数据端口上的数据被打印机接收。循环测试 BUSY("忙")信号的时间应根据系统的不同而定。程序如下：

```
PIOC:SETB P3.0;  打开打印机数据端口。
     MOV    DPTR,#7FFFH;打印机地址 7FFFH→DPTR。
     MOVX   @DPTR,A;输出字符→打印机。
     MOV    R7,#7;设置超时计数器。
     MOV    R6,#0
     MOV    R5,#0
PIO1:JNB P1.6,PIO2;读 P1.6 状态，判打印机忙?
     DJNZ R5,PIO1;打印机忙，继续。
     DJNZ R6,PIO1
     DJNZ R7,PIO1
     SETB PERR;PERR 位置"1"。
     RET
PIO2:CLR P1.4;打印机闲，选通打印机。
     SETB P1.4
     RET
```

2. 回车换行与换行子程序 CRLF

本程序先向打印机输出一个回车(0DH)控制字符，打印机一旦接收到该控制字符即输出打印缓冲器中已有的内容，打印后回车。然后把换行(0AH)控制字

符送到打印机的数据端口上，循环测试 BUSY（“忙”）信号，如果打印机忙，则继续测试；如果打印机闲，则发出$\overline{\text{STROBE}}$（选通）信号，选通打印机，此时打印机换行。程序如下：

```
CRLF:MOV   A,#0DH;0DH(回车)→打印机
     ACALL  PIOC
LF:  MOV   A,#0AH;0AH(换行)→打印机
     MOVX  @DPTR,A
     MOV   R7,#7FH;设置超时计数器。
     MOV   R6,#00H
     MOV   R5,#00H
CLF1:JNB   P1.6,CLF2;读 P1.6 状态,判打印机忙?
     DJNZ    R5,CLF1;打印机忙,继续。
     DJNZ    R6,CLF1
     DJNZ    R7,CLF1
     SETB    PERR;PERR 位置“1”。
     RET
CLF2:CLR   P1.4;打印机闲,选通打印机。
     SETB    P1.4
     RET
```

3. 7/72”回车换行子程序 LF8Z

本程序完成打印输出上(下)半个汉字时的回车换行。当打印输出上半个汉字时，先输出 ESC(1BH)1(31H)控制字符(二个)，打印机接收到后即打印输出上半个汉字，打印结束后实现 7/72”的回车，然后调用换行子程序；当打印输出下半个汉字时，仅向打印机输出一个回车(0DH)控制字符，打印机接收到后即打印输出下半个汉字，然后回车。程序如下：

```
LF8Z:JB   RTN,NLF8Z;RTN  位=1? yes.转 NLF8Z
     MOV   A,#1BH;NO.ESC  1
     ACALL  PIOC
     MOV   A,#31H
     ACALL  PIOC
     ACALL  LF;0AH(换行)→打印机
     RET
NLF8Z:MOV  A,#0DH;0DH9 回车)→打印机
      ACALL  PIOC
      RET
```

4. 选择图象模式子程序 ESCS

本程序先输出 ESC(1BH)*(2AH)控制字符(二个)，其次输出 m(图象模式)，最后输出 n1n2(映象信息即汉字点阵的长度)。此时，打印机即可按照设定的图象模式进行图形和汉字的打印输出。程序如下：

```
ESCS:  MOV    A,#1BH;        ESC
                             *
       ACALL  PIOC
       MOV    A,#2AH
       ACALL  PIOC
       MOV    A,B;           m
       ACALL  PIOC
       MOV    A,R3;          n1
       ACALL  PIOC
       MOV    A,R4;          n2
       ACALL  PIOC
       RET
```

5. 设定横向列表位置子程序 ESCD

本程序先输出 ESC(1BH)D(44H)控制字符(二个)，其次输出 n(横向列表的位置)，最后输出 NUL(00H)。此时，打印机就从设定的位置(列)处开始打印输出。程序如下：

```
ESCD:  MOV    A,#1BH        ;ESC  D
       ACALL  PIOC
       MOV    A,#44H
       ACALL  PIOC
       MOV    A,R4          ;n
       ACALL  PIOC
       MOV    A,#0          ;NUL
       ACALL  PIOC
       RET
```

6. 设定多种打印模式子程序 ESCJ

本程序先输出 ESC(1BH)！(21H)控制字符(二个)。其次输出 n(打印模式)。此时，打印机即可按照设定的打印模式进行 ASCII 码的打印输出。程序如下：

```
ESCJ:  MOV    A,#1BH;       ESC !
       ACALL  PIOC
       MOV    A,#21H
       ACALL  PIOC
       MOV    A,B           ;n
       ACALL  PIOC
       RET
```

7. 打印上(下)半个汉字子程序 PRNS

本程序首先输出 ESC(1BH)*(2AH)2(02H)控制字符(三个)即选择 ESC Y 图象模式 9 高速双密度)，然后连续输出缓冲区 1(2)映象信息即汉字点阵到打印机，等到缓冲区 1(2)的内容全部输完之后立即进行 7/72”的回车换行。程序如下：

```
PRNS:  PUSH   DPH;保存 DPTR。
       PUSH   DPL
       MOV    B,#2;选择 ESC  Y 模式。
       LCALL  ESCS
       POP    DPL;恢复 DPTR。
       POP    DPH
       INC    R4
PRN2:  PUSH   DPH;保存 DPTR。
       PUSH   DPL
       MOVX   A,@DPTR;映象信息即汉字点
              阵→打印机。
       LCALL  PIOC
       POP    DPL;恢复 DPTR。
       POP    DPH
       INC    DPTR
       DJNZ   R3,PRN2;映象信息即汉字点阵
              取完吗? N0。转 PRN2
       DJNZ   R4,PRN2
       ACALL  LF8Z;yes.7/72”回车换行。
       RET
```

8. 计算横向汉字点阵的首址子程序 SAB0

本程序根据汉字文本中汉字的机内码即分区、代

码计算出一个横向汉字点阵的首址。

横向汉字点阵首址的计算公式为：

横向汉字点阵首址＝2000H×分区＋20H X 代码

9. 横向汉字点阵转换为纵向汉字点阵并写入缓冲区(1,2)子程序 HTOV

本程序把一个横向汉字点阵的四块点阵(左上角、左下角、右上角、右下角)分别旋转 90°为可供打印输出的纵向汉字点阵，同时写入缓冲区(1,2)。纵向汉字点阵的排列图如下所示：

第0字节	第1字节	第2字节	……	第13字节	第14字节	第15字节	上半汉字点阵
第16字节	第17字节	第18字节	……	第29字节	第30字节	第31字节	下半汉字点阵

10. 1/4 个汉字点阵旋转 90°子程序 HTVA

本程序把一个横向汉字点阵中的第 k 块(K＝1-4)即 1/4 个汉字点阵的 8 个字节的相同的位(由高到低：b7－b0)组成一个纵向的 8 个字节，即实现了 1/4 个汉字点阵旋转 90°功能。

11. 写入缓冲区 1(20)子程序 HTVC(HTVD)

本程序把纵向汉字点阵的第 i 个字节(i＝0－15)写入缓冲区 1，把纵向汉字点阵的第 j 字节(j＝16－31)写入缓冲区 2。

12. 数据指针加 8 子程序 DPAD

本程序实现数据指针 DPTR 加 8 的功能。

(接 36 页)

```
70 A=-32767 : B=-32762 : C=-32758 : D=-32753
   80 FOR I=0 TO 35
   90 F=130,813 * 1,0594634 # ∧ I : NUMBER=
4000000/16/F
   100 A(I) = INT(NUMBER/256) : B(1) = INT
(NUMBER-A(I) * 256+.5)
   110 NEXT
   120 POKE A,248:POKE B,254 : POKE C,251 :
       POKE D,15 : X=USR(0)
   125 POKE A,240 : POKE C,244
   130 READ X,Y : IF X=-1 THEN POKE A,248:
POKE B,255:X=USR(0) : END
   140 X1=11
   150 IF X>9 THEN X=X/10:X1=23
   160 IF X<1 THEN X=X * 10:XI=1
   170 I=INT(X * 1.8-1.3)+X1
   180 POKE B,A(I) : POKE D,B(I) : X=USR(0)
   190 FOR T=1 TO 100 * Y * SPEED:NEXT
   200 GOTO 130
   210 DATA……
   1000 DATA -1,-1
   OK
```


北京安华科技市场西城经营部

主要经营：电子计算机、电子元器件

各种微机：

· 中华学习机：CEC－1

¥900.0　小蜜蜂－I ¥920.00

· 家庭 PC 机：¥3150.00

主频：4.77/12 兆　RAM：640KB　软驱：360KB×2，101 键盘

14″双频单显

Super286：¥6950.00

主频：10/21 兆　RAM：1MB 可扩 5MB5¼软驱：1.2MB.360KB 各 1

硬盘 3.5″40MB. 101 键盘 14″双频单显 数显机箱

选配：14″CGA 彩显加 1500 元 14″1024×768 VGA 彩显加 3800 元

各种：APPLE Ⅱ、CEC—I、PC 机接口板；九针、24 针打印机

还承接各种微机的修理业务

对教育部门、个人购机实行优惠

地址：北京西四北大街 120 号 电话：65.6342 邮码：100034

联系人：孙义强　宋函华　BB 机：126 呼 45198　42616

北京华源技贸商行

经营：中华学习机、苹果机系列产品

中华学习机：750～1150 元(保修三年)

天坛学习机：995 元(保修一年)

软盘驱动器：520～620 元

中华学习机各种软件 800 余种

软盘：(学生盘、ACT、DATACE、3M、Maxell janus)

2.5—7.5 元

单包显示器　520—550 元

中华学习机扩充卡　打印卡　Z80 打印卡　驱动卡　双驱动卡等

游戏棒(开关量，模拟量)　39～50

中华学习机各种资料、书籍　30 余种

凡在本部购买学习机者，将得到保修、免费培训的优惠，欢迎来函索取目录，本部代修学习机、驱动器、并可为学校配置中华学习机网络。批发面议。

地址：北京东城沙滩五四大街 52 号　邮政编码：100009

电话：512.7308　开户行：工商行崇外分理处中都信用社

帐户：华源技贸商行

帐号：中都 03040

联系人：于民　张小键

学装微电脑

学装 μp—8

微电脑控制跨接线弯制机

易齐干

组装电子部件时多数印刷线路板上使用图 1 所示的跨接线。跨接线一般是∅0.6 毫米镀锡线，弯制成“⌐”形。

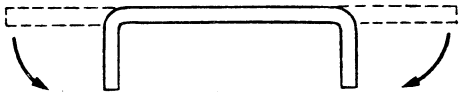


图 1

1. 机械结构

跨接线弯制机大致分为：模具定位、镀锡线送进、切断和弯曲等四部份。

(1)模具定位。

模具指弯曲镀锡线时使用的上模具、下模具。如图 2 所示。固定块和移动块组成上模具，圆柱凸轮控制它左右移动，决定镀锡线弯曲长度。下模具是正六角柱体。如图 3 所示。下模具固定在轴上，与轴一起旋转，在圆柱凸轮的控制下，上、下模具的宽度相适应，即可选择跨接线弯曲长度（六种选择：8；10；12；15；18；20mm）。小形 DC 电机通过 3 级减速后，驱动下模具轴低速回转。DC 电机轴上安装孔盘，由光传感器探测穿过的光，检测电机转速，将产生的脉冲输入给微电脑。如图 4 所示，由编码器控制模具移动。

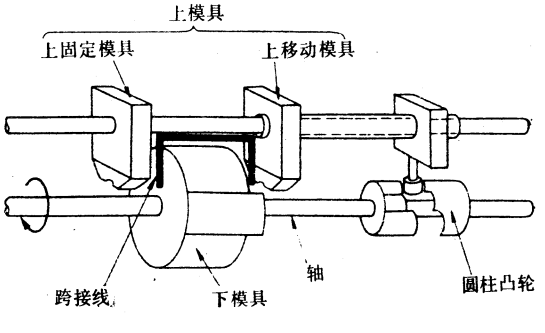


图 2

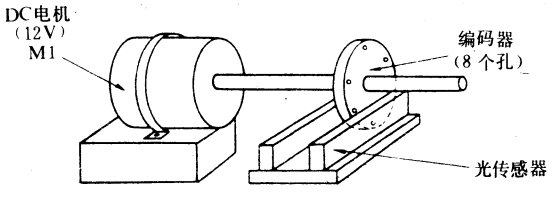


图 4

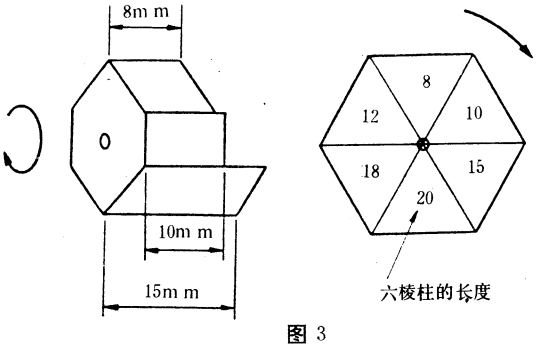


图 3

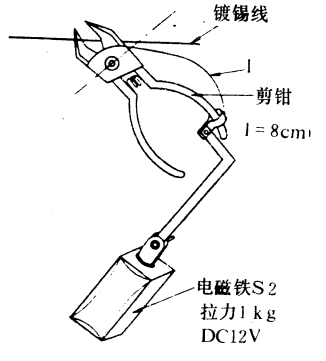


图 6

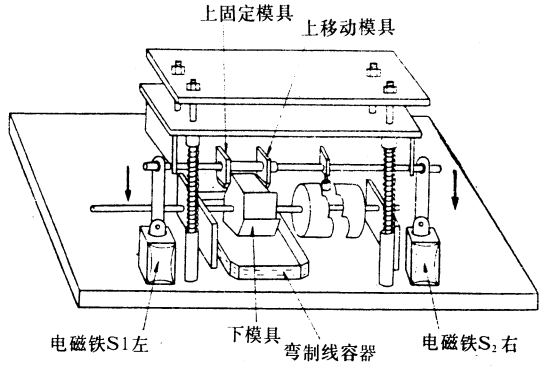


图 7

(2)镀锡线的送进

∅0.6mm 的镀锡线绕在线轴上，进入下模具后，由于卷绕过而成圆环状，不能正确地弯曲。因此，如图 5 所示，通过两对矫直滚将卷绕变形的线矫正过来。使用 DC 电机与减速齿轮驱动送进滚子回转，镀锡线从两个送进滚子之间通过，完成镀锡线的送进。

光传感器检测光穿过编码器的孔数，产生脉冲输入给微电脑，决定镀锡线的进给量。

(3)镀锡线切断。

如图 6 所示，按所需要的尺寸，切断镀锡线，使用小形剪钳，切断需要 3kg/cm 力矩，则电磁铁以 1kg 的拉力下拉剪把的端部(柄长 8cm)。

(4)镀锡线弯曲

镀锡线的弯曲如图 7 所示。在模具的左右两侧装有电磁铁，将上模具拉向下模具，镀锡线受压弯曲。

2. 接口制作：

微电脑教育部件 μp－80 发出的信息变换为适合操纵机械的信息。接口电路图如图 8 所示。μp－80 的输入输出部件端口 C 高位④输出“通”信息，导通晶体管 2SC1815，继电器 R4 为“通”，驱动模具电机 M4 回转。编码器检测到模具应该处于停止位置时，端口 C 高位④输出“断”信息，模具电机 M_4 停止回转。接着，从端口 C 高位⑤输出“通”信息，通过晶体管以及继电器 R_3，驱动镀锡线送进电机 M_3，传动如图 5 送进滚子回转，完成镀锡线的送进。同时，编码器(图 5)检测送进长度，达到既定长度后，M_3 电机自动停止转动。然后，由端口 C 高位⑥输出“通”信息，驱动电磁铁 S_2(图 6)拉动剪钳剪断镀锡线。最后，端口 C 高位⑦输出“通”信息，电磁铁 S_1(图 7)动作，上、下模具合垄折弯镀锡线。1 秒钟后，上模具返回原处。

这样，结束一个循环。在折弯镀锡线时，一旦 M_4 电机旋转，下模具转 1 转，如与基准起始位置一致，附着在下模具上的跨接线会落到盛放跨接线的盒内，如图 7 所示。

μp－80 与接口连接，如图 9 所示。

当控制信息为 5V 时，晶体管 2SC1815 导通，继电器 R_1～R_4 为“通”，步进电机与电磁铁工作。若控制信息为数 0V，晶体管 2SC1815 不导通，步进电机与电磁铁不工作。

3. 软件设计。

(1)主流程图

表 1 给出输入输出分配表，跨接线弯制机主流程图如表 2 所示。它将显示、基准、模具位置设置、镀锡线送进、切断、弯曲、根数的计数等各功能子程序化。讨

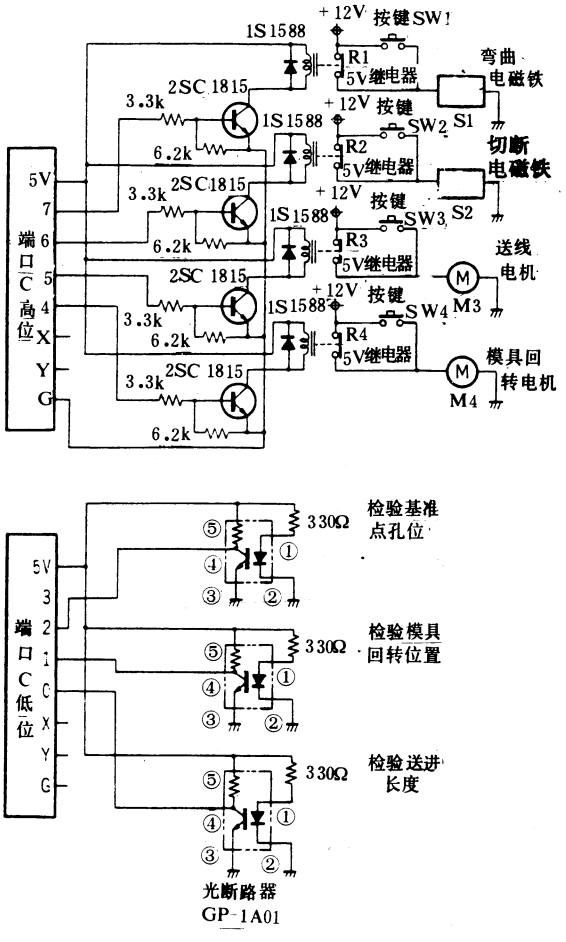


图 8

论如下。

(2)显示

使用 4 位数字显示部件，右边两个 7 段 LED 显示加工根数，左边两个 7 段 LED 显示弯曲长度。长度与根数的置位数据分配如表 3 所示。显示子程序流程图与程序如表 4 所示。

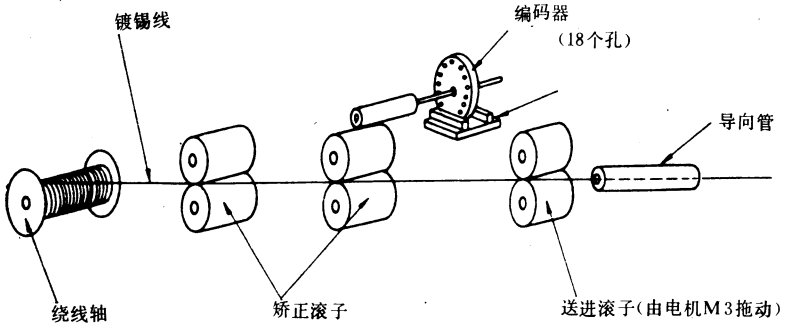


图 5

表 1 输入出分配表

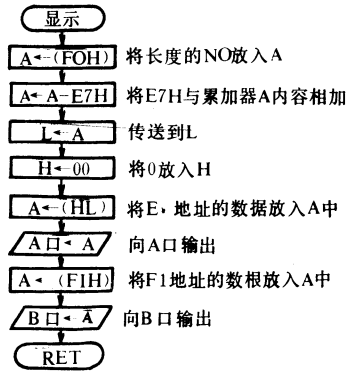
端口名	输入输出	出脚名	任　　务
端口A	输出	0～7	显示弯曲宽度
端口B	输出	0～7	显示加工根数(10进制数)
端口C	输出	7	弯曲电磁铁
	输出	6	切断电磁铁
	输出	5	送进电机
	输出	4	模具回转电机
	输入	3	—
	输入	2	编码器(线的送进)
	输入	1	编码器(模具的位置)
	输入	0	编码器(基准点)

表 3 长度与根数置位数据分配

地址	内容		
F0H	长度	01H	8 mm
		02H	10mm
		03H	15mm
		04H	20mm
		05H	18mm
		06H	12mm
F1H	根　数	以 10 进制表示，最大为99。	

表 2，主流程图与程序

表 4 显示子程序流程图与程序



标号	助记符	地址	机器语言	注释
HYOJI	LD A,F0H	23	3AF0 00	将长度数据NO放入A累加器
	ADDA,0E7H	26	C6 E7	将长度数据传送到HL
	LDL,A	28	6F	
	LD H,00H	29	26 00	
	LD A(HL)	2B	7E	长度数据输入给A
	OUT(00H),A	2C	D3 00	在端口A输出
	LDA(F1H)	2E	3AF1 00	向端口B输出数量
	OUT(01H),A	31	D3 01	
	RET	33	C9	

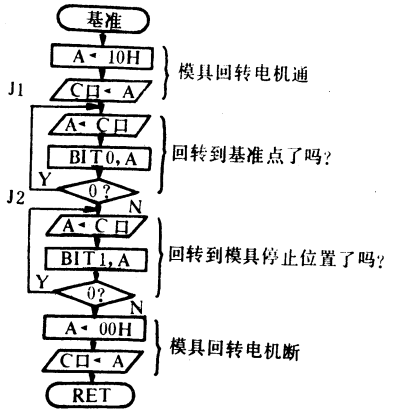
标号	助记符	地址	机器语言	注释
START	LDA,81H	00	3E 81	设制控制码
	OUT(03H),A	02	D3 03	
	LD SP,0100H	04	31 00 01	设置堆栈
MAIN	CALL HYOJI	07	CD 23 00	显示长度、数量
	CALL KIJUN	0A	CD 34 00	模具回转到基准点
	CALL TEIICH	0D	CD4B 00	模具转到所要求的位置
	CALL OKURI	10	CD66 00	供给跨接线
	CALL SETSUD	13	CD8D 00	切断
	CALL MAGE	16	CD9C 00	弯曲
	CALLGENSAN	19	CDAB 00	根数减法运算
	JP NZMAIN	1C	C207 00	加工完了吗?
	CALL HYOJI	1F	CD23 00	显示0
	HALT	22	76	停机

标号	助记符	地址	机器语言	注释
	DB 08H	E8	08	用于显示的数据
	DB 10H	E9	10	
	DB 15H	EA	15	
	DB 20H	EB	20	
	DB 18H	EC	18	
	DB 12H	ED	12	

(3)确定基准点和模具位置。

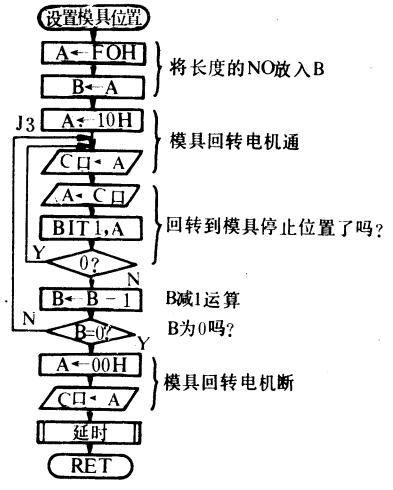
模具起始基准点的流程图与程序如表 5 所示，弯曲 8mm 跨接线的模具回转时基准点在正上方。为得到不同长度的跨接线，决定上、下模具位置的流程图与程序如表 6 所示。模具位置设置原理如图 10 所示。步进电机输入两支编码器产生的脉冲，以不同位置的孔决定电机停止转动。

表 5



标号	助记符	地址	机器语言	注释
KIJUN	LD A,10H	34	3E 10	模具回转电机通
	OUT(02H),A	36	D3 02	
J1	IN A,(02H)	38	DB 02	回转到基准点了吗?
	BIT 0,A	3A	CB 47	
	JP Z, J1	3C	CA 38 00	
J2	IN A,(02H)	3F	DB 02	回转到模具停止位置了吗?
	BIT 1, A	41	CB 4F	
	JP Z, J2	43	CA 3F 00	
	LD A ,00H	46	3E 00	模具回转电机断
	OUT(02H),A	48	D3 02	
	RET	4A	C9	

表 6



标号	助记符	地址	机器语言	注释
TEIICH	LDA(FOH)	4B	3A F0 00	将长度的NO放入B中
	LDB, A	4E	47	
	LDA, 10H	4F	3E 10	模具回转电机通
	OUT(02H), A	51	D3 02	
J3	INA ,(02H)	53	DB 02	能转到模具停止位置吗?
	BIT 1, A	55	CB 4F	
	JPZ, J3	57	CA 53 00	
	DEC B	5A	05	
	JPNZ, J3	5B	C2 53 00	
	LDA, 00H	5E	3E 00	模具回转电机断
	OUT(02H), A	60	D3 02	
	CALL TIMER	62	CDB500	1秒延时
	RET	65	C9	

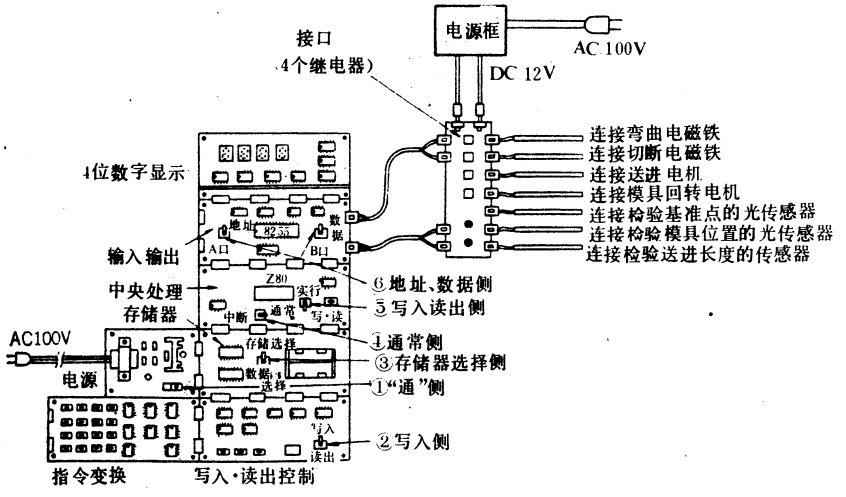


图 9

(4)镀锡线的送进

编码器孔数的不同,改变光传感器脉冲的产生,再通过脉冲计数器决定所需长度的镀锡线的送进。编码器孔数为 18 时,脉冲数与镀锡线长度的比例如表 7 所示。

1 个脉冲线的送进长度计算公式(由表 7)。

(例)镀锡线弯曲长度 8mm 时,

镀锡线的送进=28mm

每 1 个脉冲送进长度=1.12mm

这样,所需送进脉冲=28/1.12=25

25 为 10 进制数,如用 16 进制数表示为 19H(19H=1×16+9=25)。

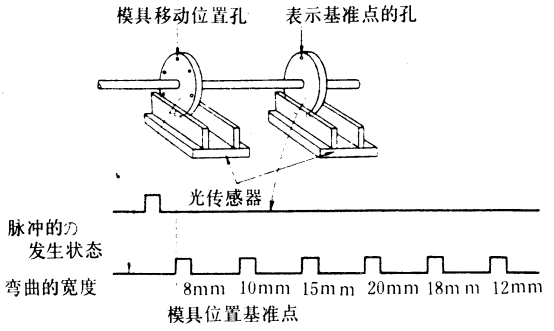


图 10

表 7

镀锡线弯曲宽度	8 mm	10mm	15mm	20mm	18mm	12mm
脉冲数	19H	1BH	20H	24H	22H	10H
镀锡线送进长度	28mm	30mm	35mm	40mm	38mm	32mm

镀锡线送进流程图与程序如表 8 所示。

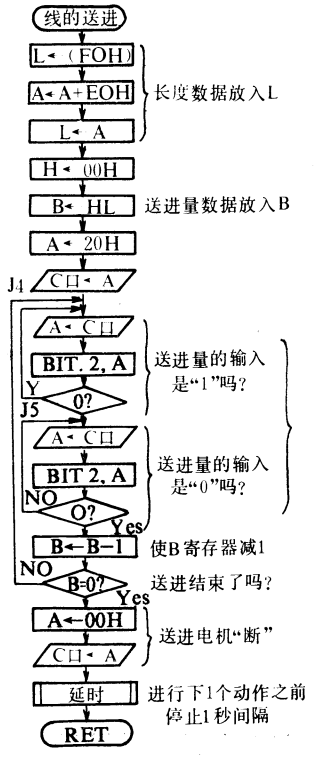
(5)镀锡线的切断

电磁铁 S_2 与剪钳连接，电磁铁 S_2 通电，它拉动剪钳切断镀锡线。流程图与程序如表 9 所示。电磁铁 S_2 通电时间延时 1 秒。延时流程图与程序如表 10 所示。

(6)镀锡线弯曲

设置在模具左右两侧的电磁铁 S_1 通电，弯曲镀锡线。流程图与程序如表 11 所示。电磁铁通电时间延时 1 秒，也使用前述的延时程序。

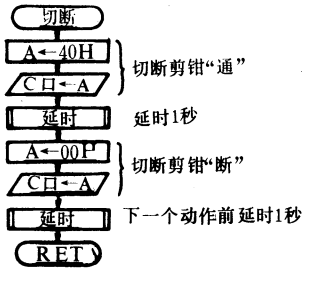
表 8



标号	助记符	地址	机器语言	注释
OKURI	LDA,(FOH)	66	3A F0 00	长度数据装入L
	ADD A,EOH	69	C6 E0	
	LD L,A	6B	6F	
	LD H,00H	6C	26 00	
	LD B(HL)	6E	46	送进量数据放入B
	LD A,20H	6F	3E 20	送进电机"通"
	OUT(02H),A	71	D3 02	
J4	IN A,(02H)	73	DB 02	送进量输入是"1"吗?
	BIT 2,A	75	CB 57	
	JPZ,J4	77	CA 73 00	
J5	IN A,(02H)	7A	DB 02	
	BIT 2,A	7C	CB 57	送进量输入是"0"吗?
	JPNZ,J5	7E	C27A 00	
	DEC B	81	05	使B减1
	JPNZ,J4	82	C2 73 00	送进结束了吗?
	LDA,00H	85	3E 00	送进电机"断"
	OUT(02H),A	87	D3 02	
	CALL TIMER	89	CDB500	延时1秒
	RET	8C	C9	

标号	助记符	地址	机器语言	注释
	DB 19	E1	19	送进量8cm
	DB 1B	E2	1B	送进量10cm
	DB 20	E3	20	送进量15cm
	DB 24	E4	24	送进量20cm
	DB 22	E5	22	送进量18cm
	DB 1D	E6	1D	送进量12cm

表 9

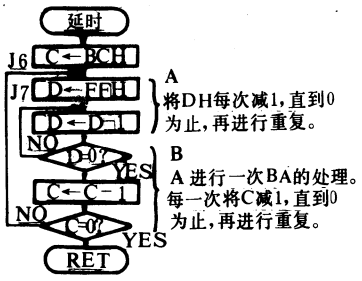


标号	助记符	地址	机器语言	注释
SETSUD	LD A,40H	8D	3E 40	切断剪钳"通"
	OUT(02H),A	8F	D3 02	
	CALL TIMER	91	CDB500	延时1秒
	LD A,00H	94	3E 00	切断剪钳"断"
	OUT(02H),A	96	D3 02	
	CALL TIMER	98	CDB500	延时1秒
	RET	9B	C9	

(7)根数减法运算

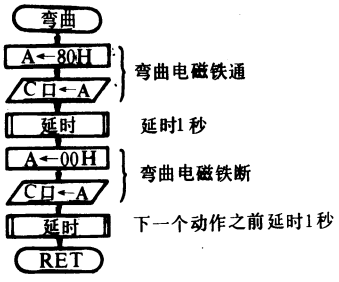
运算只能在 A 累加器内进行，F1H 地址内容为加工根数，要传送到 A 累加器。每加工 1 根跨接线要进行减 1 运算，并变为 10 进制数。μp－804 位数字显示部件 7 段 LED 右两位显示余数。根数减法运算流程图与程序如表 12 所示。

表 10



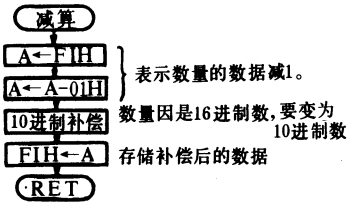
标号	助记符	地址	机器语言	注释
TIMER	LDC ,BCH	B5	0E BC	延时1秒
J6	LDD ,FFH	B7	16 FF	
J7	DEC D	B9	15	
	JP NZ,J7	BA	C2 B9 00	
	DEC C	BD	0D	
	JP NZ J6	BE	C2 B7 00	
	RET	C1	C9	

表 11



标号	助记符	地址	机器语言	注释
MAGE	LDA, 80H	9C	3E 80	弯曲电磁铁“通”
	OUT(02H),A	9E	D3 02	
	CALL TIMER	A0	CD B5 00	延时1秒
	LDA,00H	A3	3E 00	弯曲电磁铁“断”
	OUT(02H),A	A5	D3 02	
	CALL TIMER	A7	CD B5 00	延时1秒
	RET	AA	C9	

表 12

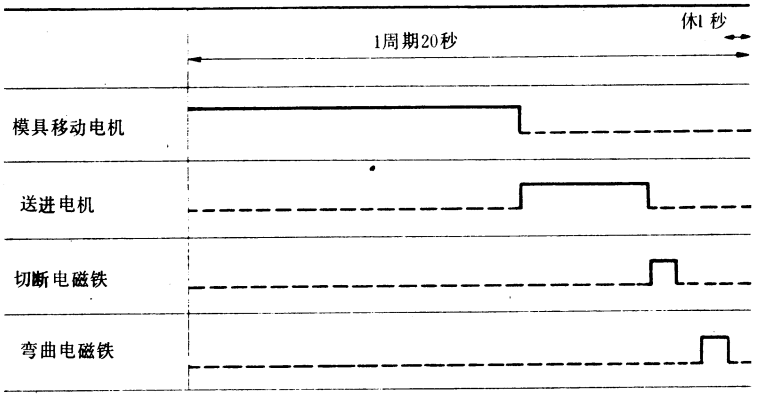


标号	助记符	地址	机器语言	注释
GENSAN	LDA,(F1H)	AB	3A F1 00	根数数据减1
	SUB 01H	AE	D6 01	
	DAA	B0	27	10进制补偿
	LD(F1H),A	B1	32 F1 00	存储补偿后的数据
	RET	B4	C9	

表 13

开关No	开关方向
①	“通”侧
②	读出侧
③	选择侧
④	通常侧
⑤	实行
⑥	端口A、端口B侧

表 14



(8)程序写入与检验

照图 9 连接 μp－80，各开关的方向如图中所示。由 00H 地址到 EDH 地址顺序写入程序。写入完毕后，再由 00H 地址开始，顺序调出检验程序写入是否正确。查出错误之处，要进行修改并再写入。FOH 地址内容为长度，F1H 地址内容为加工根数。

(9)运转

①首先按动接口印刷线路板上的按键开关 SW_3，切断镀锡线剪钳运动到最前位置。

②图 9 的①～⑥开关方向如表 13 所示。

③驱动源电机、电磁铁时电源开关为“通”状态。

④按动中央处理部件的复位开关，时间为 1 秒，开关⑤在实行侧，跨接线弯制机则开始运转。加工时序表如表 14 所示。

⑤加工完所需要的数量，4 位数字显示部件 7 段 LED 右边两位显示 00，弯制机运转停止。

(10)软件改进

本文软件举例中，每次输入 1 种跨接线的弯曲长度、数量等信息。改进软件程序，输入不同长度的多种跨接线的信息，也能进行多品种、大批量生产。1 个品种生产结束后，采用蜂鸣器、放大器等方法，通告操作者，这样，将会提高生产效率。本文举例中，精简基本程序，由多个子程序很方便地组成完整的程序。今后，在程序中为添加补充多种功能，采用补充子程序方法也较为方便，容易得到较为完整的程序。

如何用EPROM读写器对EEPROM编程

闫怀兵　陈立山

EEPROM电擦除存贮器用在控制电路中，可以方便的更新用户的一些控制参数，随写随擦，停电后保存信息，加电后又可改写数据，这种芯片可进行10000次以上的擦写工作，但是对于EEPROM在初始化的编程与写入时，它与传统上应用的EPROM(只读存贮器)写入与读出的方法不相同，因而需要专门的EEPROM写入器才能实现，笔者在使用电擦除的EEPROM时，仅将使用的紫外线擦除的EPROM写入器略加改动，就可以很方便的对电擦除的EEPROM进行编程写入，其方法如下：

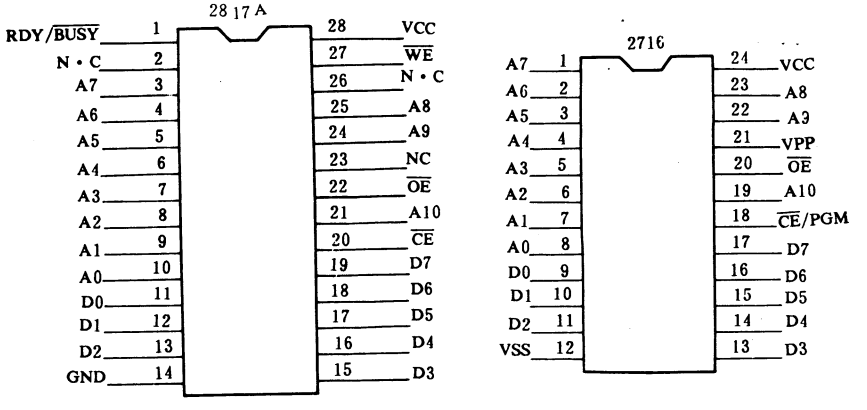


图1

用一台普通的PC机EPROM写入器，按照欲写入的EEPROM芯片，选择管脚与之相似的27××系列EPROM芯片，在电路上略加改动便可用EPROM写入器写入，如：电擦除芯片EEPROM 2817A，它是28脚芯片，而2716是24脚芯片，(参看图1)，仅和2817A的上四脚，即1,2,28,27四脚不相同。2817A的28脚是Vcc，2716的24脚是Vcc，而2817A的26脚正好是N.C(空脚)，从上所知，若要对2817A进行读出时，只要将其插入2716的24脚管座，然后从第24脚引一条线接在2817A的第28脚上，此时便可以按照读2716的方式进行读出，无须改动原2716的驱动程序便可。

当进行写入时，从图中得知，与2716第21脚V_{PP}对应的2817A的23脚为N.C。第1脚是RDY/$\overline{BUSY}$，这是2817A的专用写入查寻信号端，当正在写入时RDY/$\overline{BUSY}$输出低电平，表示忙，当写入结束时RDY/$\overline{BUSY}$输出高电平，表示写入结束，通知CPU可以写入下一个字节，据此，可利用PC机中的CPU READY端，当它为低电平时，停止数据传送，处于等待状态，当其变为高电平时，将完成数据传送，由此可见它的逻辑功能正好与2817ARDY/BUSY所要的逻辑状态完全相同，而等待信号可由PC机62芯I/O扩展槽上的A_{10}I/O CHRDY端控制产生，所以，便可从A_{10}上引一条线到2817A的第1脚，作为查寻写入是否完毕的控制信号。此外，2817A的第27脚$\overline{WE}$是写入控制信号，当$\overline{WE}$为低时，写入有效，因此我们完全可以用PC机62总线I/O插槽上的MEMW信号控制它，将B_{11}连接到2817A的第27脚，最后运行原PC机中的程序一个字节一个字节的自动写入到2817A中采用这种方法即可省略一套写入器，又不必改动写入驱动软件，也无须去计算写入的时间控制参数，经我们多次写入，从未失误，简单可靠。

LASER—310 机加接数字音响发生器

王建峰

SOUND-AY-3-8914 是数字音响发生器，可以与 8 位机配合工作，创造各种音色，丰富微电脑的使用范围。关于它的详细原理可以参见《无线电》86 年第三期，本文只向大家介绍它在 LASER 机上的实际应用方法，有兴趣的读者可以一试（同功能的芯片还有 AY--3-8910 和 AY-3-8912）。

连接线路非常简单，只使用一个非门，为了降低产生的声音频率，还可以在时钟输入端加接分频器。由于所有端口均与 TTL 电平兼容且为三态线，所以使用方便，工作可靠。如图 1 所示，实质上是把该芯片作为一个只写外设。通过实验，得出以下有用数据：

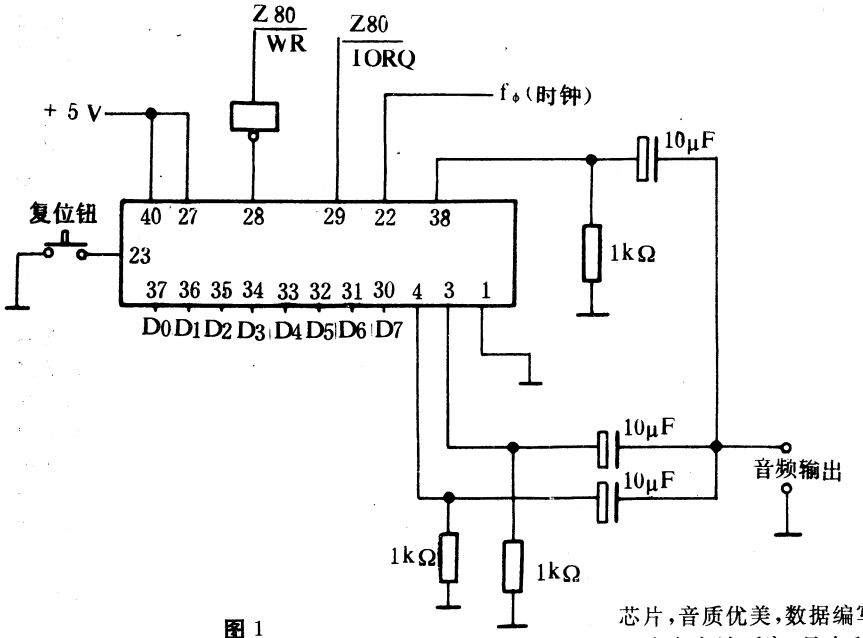


图 1

表一

通道	幅度	频率	
		高位	低位
A	251	240	244
B	252	241	245
C	253	242	246

（表一：数据为寄存器地址）

表二

音质	/	/	N	N	N	M	M	M
通道	/	/	C	B	A	C	B	A
二进制码位	B7	B6	B5	B4	B3	B2	B1	B0

（表二：M—music 乐音，N—noise 噪音）

该芯片共有三个音响通道，它们可以分别控制幅度、音乐频率、噪音频率（或不输出噪音）和采用包络音形，见表一。

幅度寄存器可以存入 0～15（即有 16 档音量）当放入 255 时即为包络发生器控制。频率寄存器高位可放入 0～15，低位 0～255，产生的频率为 f＝fφ/16/（高位＊256＋低位）。

另外有用的寄存器还有包络周期寄存器，地址为 243（高位）和 247，均可存入 0～255 的数据；包络形状寄存器，地址 250，可存 0～15；噪音频率地址 249，可存 0～31；还有一控制寄存器，功能见表二，当某一位为 0 时，可以打开相应通道（反之关闭该通道）。

向这些寄存器写数据采用以下程序，读者可通过实验来了解该芯片的使用。

LDA，x——→选取寄存器

LD（FFFFH），A——→锁存地址

LDA，y——→需写入之数据

OUT（FDH），A——→发出 $\overline{IORQ}$ 信号，写入数据

RET

把它作为 BASIC 程序的子程序，使产生各种声音讯号。以下提供一简单程序，用来演奏单音乐曲。使用该芯片，音质优美，数据编写简易，读者可以分析改编，用于演奏音域更宽，具有和声、节奏和一定动态的乐曲，同样可制造枪炮、海浪等音色，为电子游戏配声。

注：DATA 的编写：用此程序可以演奏 1～7 的 C 大调乐曲。DATA 的形式为一组二个数，前者为声调，可用 1～7 代表音乐中的 1～7，用 0.1～0.7 代表 1～7，10～70 代表 1～7，1.5、2.5 等表示 #1、#2，其它类推；后者为节拍时值，乐谱中时值为多少就取多大。（如一拍写 1，半拍写 0.5……）

```
]List
10 INPUT "SPEED(1-100)";SPEED
20 FOR J=-1 TO 7
30 READ X : POKE -32767+J,X : POKE -32758+J,X
40 NEXT
50 DATA 62,0,50,255,255,62,0,211,253
60 POKE -32750,201:POKE 30862,0 : POKE 30863,128
```

（下转 31）

维修经验谈

任天堂系列游戏机检修实例(续)

林华春

例十、故障现象:一台红白 LT－900D 游戏机,I 号操纵手柄操作时无规律出现暂停,将 Ⅱ 号手柄拔下,I 号手柄才能正常使用。

分析与检修:首先怀疑 Ⅱ 号手柄中有断线或碰线,经检测后没有发现断、碰线,根据暂停现象时有时无,确认故障还在 Ⅱ 号手柄操纵盒中,测量 4021 各脚电位均正常,换一片新 4021 后故障仍存在,经仔细检查后发现 4021 的⑭脚与电源 5V 的联接印刷电路板有开路现象,焊接后故障排除。此现象说明 CMOS 电路不使用的空脚一定不能悬空。

例十一、故障现象:一台任天堂 616 游戏机,有图像,有伴音,但图像上下跳动。

分析与检修:该游戏机在输出插口处有一兼容变换开关,当遇到不兼容的游戏卡时,可拨动此开关,游戏可正常进行。出现上述故障时先把兼容开关拨向另一边,但图像无变化,然后用逻辑笔按 4011 四与非门的真值表,检测其输出端的逻辑变化关系,发现有一路输出无变化,换一片 4011 后故障排除。

例十二、故障现象:一台红白游戏机,在同是 PAL－D 制的彩电上使用,有些彩电显示彩色图象,而有些彩电只显示黑白图像,此时伴音正常。

分析与检修:出现上述故障的原因有两方面,一是彩电中彩色副载波捕捉频率窄,二是游戏机中晶振频率偏移。对于不同型号的同制式彩电,虽然其彩色副载波的频率同为 4.43MHE,但捕捉频率范围宽的彩电,其彩色还原好,也就是能够显示出彩色图象的彩电,若捕捉频率范围窄的彩电,游戏机晶振频率稍有偏移,即不能还原彩电图像。因此我们从游戏机方面来排除上述故障,方法是改变与晶体 26.601712MHE 串联的电容,使电容量适当增加或减少,彩色即可还原,经上面方法处理后故障排除。

例十三、故障现象:一台红白 737 游戏机,图象正常,无伴音。

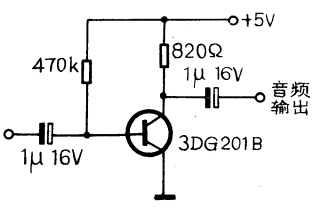
分析与检修:从电视荧光屏图象看,画面清晰,因此可以判断故障是由游戏机伴音电路损坏引起。用 2200Ω 高阻抗耳机,听 CPU 的①、②脚,有伴音信号;听 74HC368 的⑬脚也有伴音信号;听 74HC368 的⑭脚,无伴音信号,这说明 74HC368 损坏。由于游戏机其它功能正常,可确定 74HC368 只有伴音一路损坏,所以采用外接放大电路解决,电路如附图,接好后开机,伴音恢复正常。

例十四、故障现象:一台红白无型号游戏机,图象正常,伴音失真。

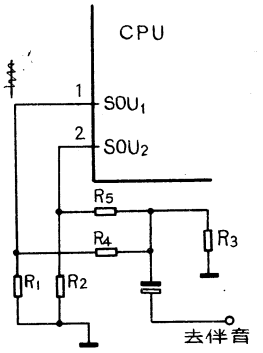
分析与检修:首先用小螺丝刀微调 RE 射频调制盒中的伴音振荡线圈,失真现象无变化,因此确定故障不是 RF 射频调制盒有故障。沿伴音电路用高阻抗耳机听伴音信号,听 CPU 的①、②脚有伴音信号;听输出电阻 R_5 另一端(见附图)却发现缺一路信号,经仔细辩别后认定是 CPU②脚的伴音信号没有送出。用万用表测试 R_5 阻值,该电阻已开路,用新电阻替代后,伴音失真现象消除。

例十五、故障现象:一台红白游戏机,图象正常,伴音极小。

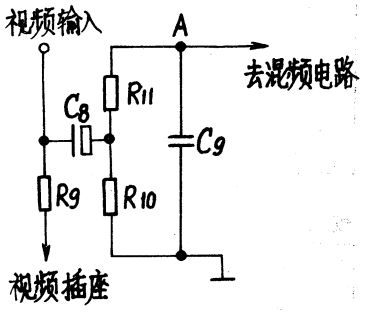
分析与检修:首先按例十四的方法听伴音信号,在



例十三



例十四



例十七

RF 射频调制盒以前电路均正常，微调伴音振荡线圈后，伴音信号增大，但图像却出现扭曲，再调图像载波振荡线圈，伴音信号又消失，用万用表检查射频调制电路后发现伴音振荡线圈次级开路，重绕后开机，经过常规调整伴音、图像均恢复正常。

例十六、故障现象：一台灰白 767 游戏机，无图像，无伴音。

分析与检修：按正常方法检修电脑板电路均正常，偶然碰一下射频盒，发现伴音线圈很热，启开盖后看到，这种射频调制盒电路采用了三只三极管，从印制板上的位置及电路原理分析，靠近伴音振荡线圈外侧的一只三极管为射频放大管，用万用表测量时，发现其已击穿，换新后开机，有图像，仍无伴音。拆下伴音振荡线圈，发现其内部断线，重绕后再开机，伴音、图像均正常。

例十七、故障现象：一台红白游戏机，有伴音，图像上有不同步的斜条。

分析与检修：用示波器测 CPU 的㉑脚，有正常的视频信号，在射频盒视频输入端测量，也有正常的视频信号，启开射频盒，见附图。测 C_8 两端均有视频信号，当测试到 A 点时无视频信号，断电后测量，发现 C_9 电容漏电严重，换新后伴音、图像恢复正常。

例十八、故障现象：一台红白任天堂 HVC－001 游戏机，开机后图像上有两条水平干扰带，伴音中有严重的交流声。

分析与检修：首先用万用表测量整流器输出的直流电压，为 12V 正常，然后测＋5V 输出电压发现不稳定，随即更换 7805 稳压 IC，开机后仍不正常，焊下电源滤波电容后发现漏电，换新后开机，干扰带去掉，但伴音中仍有交流声，用小螺丝刀调整 6.5MHZ 伴音振荡线圈后，伴音恢复正常。

“8088 系统板测试卡”对系统板总线部分的诊断测试

梁杰熙

“8088 系统板测试卡如果不能以连续方式测试时，说明故障在系统板的总线部分。此时，就要使用测试卡的单步诊断测试部分。单步诊断测试是通过测试卡上的状态来分析判断故障发生在系统总线的哪一点。所以需要对系统板总线的结构及工作状态，有一个简单的了解，才能正确的使用测试卡和判断其故障。

一、8088 微型计算机的总线结构：

1. 总线结构的划分：

以 8088 为 CPU 的微型计算机其总线可划分为地址总线，数据总线，控制总线。地址总线和数据总线是通过一级或多级缓冲器进行传输的。IBM PC/XT 及其兼容机和国产 0520 系列机中，地址总线和数据总线一般都使用的 74LS373、74LS244 和 74LS245 芯片。控制总线是 8088CPU 用于发送系统控制命令的，在 8088CPU 的微型计算机中由于 8088 是工作于最大模式，其控制命令不是由 8088 本身发送的，而是通过一

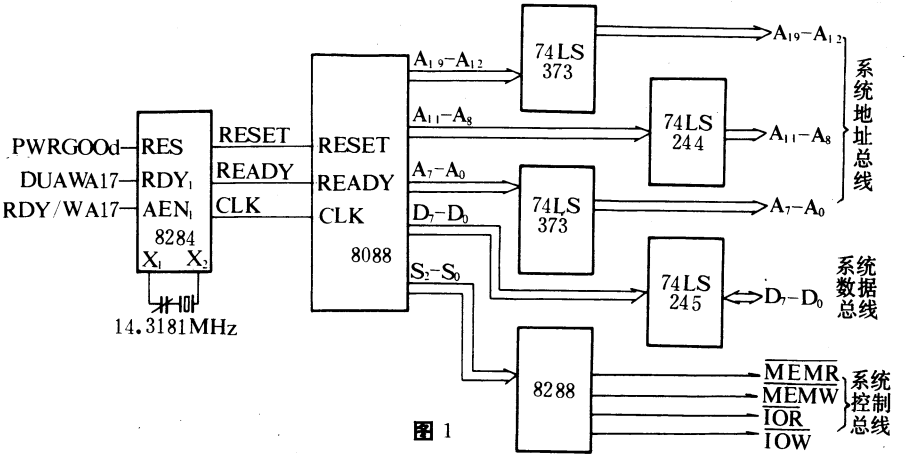


图 1

个总线控制器由 8288 提供发送。下面以 IBM PC/XT 机为例:用图 1 来描述其地址总线,数据总线,控制总线的结构:

2. 总线结构下的工作状态:

8088 在工作时主要有三个控制信号,RESET(系统复位),READY(外设准备好),CLK(CPU 时钟)。由图"1"看到 RESET,READY,CLK 都是 8284 时钟发生器提供给 8088 的,在系统加电时,这三个信号的状态如下:RESET 信号:微机电源箱的四种直流电压输出(+5V、+12V、-5V、-12V)正常后,送出 50μs 的低电平 PWRGood(电源好)信号,这个信号送到 8284 的 $\overline{RES}$ 端,经 8284 控制输出高电平的 RESET 信号,其需要保持 50μs 时间才能使 8088 达到稳定的工作状态。READY 信号:8284 的准备好控制输入端 RDY1,AEN1 分别接受来自外围电路的 $\overline{DMAWAIT}$,$\overline{RDY}$/WAIT 的信号状态,在 $\overline{DMAWAIT}$ 为高电平和 $\overline{RDY}$/WAIT 为低电平时,8284 的 READY 输出端为 8088 提供了有效的高电平信号。反之在 $\overline{DMAWAIT}$ 为低电平或 $\overline{RDY}$/WAIT 为高电平时 8284 的 READY 输出端为低电平,此时 8088 将插入等待状态,直到 READY 输出端恢复为高电平 8088 才脱离等待状态,进入 8088 工作周期。CLK 信号:8284 的 X_1,X_2 引脚连接晶体振荡器,在 8088CPU 的微型计算机中,一般晶振频率选择的都是 14.3181 MHE 的晶体振荡器,但现在有些 8088CPU 的兼容机在系统设计时为了提高其运行速度选择 24MHE,30MHE 等频率的晶体振荡器。但是其连接方式都是将晶体振荡器接至 8284 分频后提供给 8088 的 CLK 时钟端。以上的 RESET,READY,CLK 信号状态满足后,8088 就要通过地址缓冲器,数据缓冲器和总线控制器对微机系统进行,A_0～A_{19}的地址发送,D_0～D_7 的数据传输和 S_0－S_2 的总线控制状态的发送。地址总线和数据总线通过缓冲器其输入输出状态是相对应的,控制总线的输入输出状态是通过总线控制器由$\overline{S_0}$-$\overline{S_2}$的输入状态来决定 $\overline{MEMR}$,$\overline{MEMW}$,$\overline{IOR}$,$\overline{IOW}$ 等的输出状态,在 8088 进行正常工作时这些输入输出信号都应是在脉冲状态。

二、8088 系统板测试卡对总线测试的原理。

8088 微处理器的总线工作周期分成四个时钟周期,T_1 周期(总线地址发送周期),T_2 周期(总线数据传输周期),T_3 周期(8088 采样 READY 信号周期),T_4 周期(恢复周期准备下一个总线工作周期)。这里要说明的是 8088 采样 READY 信号周期,即 T_3 周期。在 8088CPU 的微机系统中 I/O 接口电路和一些慢速的存储器其数据存取的响应时间较慢,在整个系统总线工作周期频率速度下,不能获取正确的数据结果,此时需要在 8088 的总线工作周期内插入等待状态,这个等待状态取决于 T_3 周期开始时的信号下降沿 8088 采样 READY 信号,若 READY 信号为低电平,就在 T_3 周期结束后插入等待状态周期 Tw,在 Tw 周期开始时继续对 READY 信号进行采样,当 I/O 设备完成数据传输后 READY 信号变为高电平,8088 在采样到 READY 信号为高电平则此时总线工作周期进入 T_4 周期准备下一个总线工作周期。(见图"2"插入等待状态的总线工作周期时序图:

这种使总线工作周期插入等待状态让其总线工作状态一个周期一个周期的执行。因为系统总线在正常工作时在某一个周期中其信号是处在一定的有效状态下的只有在下一个时钟周期时其状态才发生变化,这样连续起来才呈现的是一个脉冲状态,我们把动态连续的总线状态通过插入等待状态的方法将其分成稳定单步(一个周期)的状态,再通过我们编制的诊断测试程序来测试是哪一点先发生了故障。此如说在插入等待状态一步一个周期的执行,在执行到某一个周期时按诊断测试程序给出的正确数码值某一个信号点的值应为高电平信号状态,此时呈现在测试卡上确为低电平状态,这就说明这个信号点是故障点。所以我们在对系统总线诊断测试时将其称为对系统总线的单步诊断测试。

三、使用"8088 系统板测试卡"如何诊断测试总线故障

1. 测试卡的单步诊断测试:

测试卡如果不能使用连续方式诊断测试系统板故障时,就要使用测试卡的单步诊断测试部分。测试卡将"L"处短接,改换成"D"处短接。见图"3""L"和"D"的短接点是测试卡的连续测试和单步测试的选择点。在单步诊断测试时,测试卡的右上角有一个标有 W_1 的按动开关,在未按下时,一直向 8088 发送 $\overline{READY}$ 低电平信号,让 8088 系统总线保持在第一个初始周期状态下。当按下按动开关 8088 测试卡就向 8088 发送一个高电平 READY 信号后马上恢复成低电平。使 8088 系统总线又保持在第二个周期状态下,当按动一下开关系统总线就向下执行一个周期并稳定在这个周期状态下。来达到走一步观查一次系统总线状态的目的。系统总线的状态在测试卡上是体现在 8088 系统板测试卡上标有 A_0～A_{19},D_0～D_7 的发光二级管上,其状态是用二进制的方法表示,亮时表示这位为"1",不亮时表示这位为"0"。在与测试卡上诊断测试程序给出的数码值(十六进制)要进行二进制与十六进制的转换后再比较其正确的对应关系,来分析判断其故障点。

2. 如何根据测试卡上的状态判断系统总线的故障点:

板测试卡在单步测试总线状态时,第一步呈现在测试卡上的地址总线的状态(A_0-A_{19})应是 FFFF0 数据总线的状态(D_0-D_7)应是 EA。为了说明问题我们列出测试卡上诊断测试程序给出的前 10 步的正确数码表:

第 1 步	FFFF0	EA	第 6 步	FFFF5	00
	FFFF1	77		FF877	FA
	FFFF2	F8		FF878	B8
	FFFF3	00		FF879	AA
第 5 步	FFFF4	F0	第 10 步	FF87A	AA

如果第一步状态出现多位不对则说明 8088 根本没有工作,这时需要检查的故障点是满足 8088 正常工

作时的三个信号的状态。即 RESET,READY,CLK,RESET 信号在开机加电时必须有一个由高向低的时间为 50μs 的跳变状态,READY 信号必须保持在高电平状态,CLK 的时钟频率必须在系统所要求的时钟频率范围内,比如 IBM PC/XT 机其频率为 4.77MHE。如果这三个信号都满足了说明 8088 芯片本身损坏,如果这三个信号有一个不满足则说明为 8088 提供这三个信号的 8284 时钟发生器损坏,或是晶体振荡器损坏。晶体振荡器其两极的对地电压平均值在 2.5V 左右说明晶体振荡器没有故障,而是 8284 损坏,如果晶体振荡器两极对地的平均电压达不到 2.5V 左右,则说明是晶体振荡器损坏,造成了 8284 无法向 8088 提供 CLK 时钟,至使 8088 不能工作。如果第一步呈现在 8088 系统板测试卡上的地址状态是正确的与数码表相符,而数据总线的状态是多位不对与数码表相差甚远,则说明故障出现在控制总线上。因为在 8088 进入正常工作时,将其地址发送出去后,接着在其进行数据传输的同时要发送出总线的控制命令,即现在的数据转输是进行存储器读操作($\overline{MEMR}$)还是进行存储器写操作($\overline{MEMW}$),或是进行 I/O 读操作(IOR)还是进行 I/O 写操作($\overline{IOW}$)。如果控制总线送不出这些总线控制命令,那么传输的数据就无法正确的呈现在测试卡上,所以必然是控制总线出现了故障。控制总线有故障也就是 8288 总线控制器有故障,因为 8284 是发送总线控制命令的控制芯片。在第一步的地址总线,状态和数据总线状态与数码表相符后,此时也就可以说 8088、8284,晶体振荡器,8288 这些芯片没有问题,这样就可以按动一下 W_1 按动开关一步一步的向下执行。在一步一步向下执行的过程中如果要发现呈现在 8088 系统测试卡上的地址总线状态或数据总线状态与数码表不相符,此时则说明故障出现在地址总线缓冲器上或是数据总线缓冲器上。如果呈现在 8088 测试卡上是地址总线的状态与数码表不符,则就要与数码表上的值找出每位的对应关系。比如在执行到第 6 步时出现了地址总线状态与数码表不符,正确的值应是 FFFF5,而呈现在 8088 系统板测试上的地址总线状态确是 F0FF5,此时将这个十六进制的状态数值按二进制展开,对应到地址总线的每一位上,即 F→A_{19}～A_{16},O→A_{15}～A_{12},F→A_{11}～A_8,F→A_7～A_4,5→A_3～A_0 则看到 A_{15}～A_{12}对应的那一位为“0”。也就是说在单步走到第 6 步时对应地址总线 A_{15}～A_{12}的那一位缓冲器芯片出现故障了,再由图纸或线路板上找到 A_{15}～A_{12}对应的那个缓冲器芯片,将其换下后故障一定排除。同样在单步执行到某一步时如果出现数据总线的状态与数码表不相符,也将其进行二进制展开来对应数据总线的每一位,找到其对应的缓冲器芯片将故障排除。总体来说单步测试的第一步是对 8088 8284 晶体振荡 8288 的诊断测试在第一步通过后的测试则是对地址总线缓冲器和数据总线缓冲器的测试,它把系统总线结构内的所有芯片都测试到了,判断出故障点所在位置。

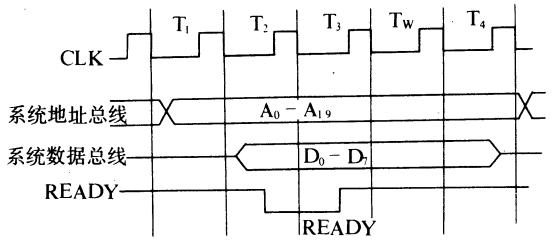


图 2

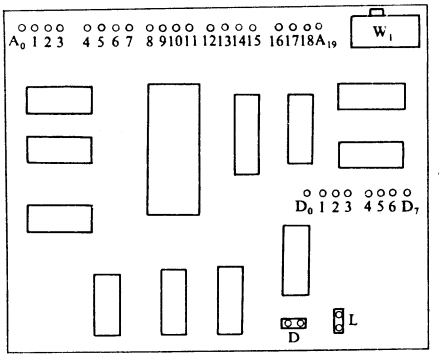


图 3

《电子与电脑》1991 年上半年合订本已经出版,每本 价 7.8 元,欲购者请附加 15%邮寄费。由于数量有限,需要者请从速。

邮购办法:邮局汇款 北京万寿路 173 信箱电子工业出版社《电子与电脑》编辑部

银行汇款 开户行北京市工商银行翠微路分理处帐号 891238—72

一九九一年计算机初级程序员有奖竞赛试题

主办单位：中国软件行业协会考试指导中心、中国高档微机协会、《电子与电脑》编辑部

一·从供选择的答案中选出正确答案，把相应的字母写在答卷纸的对应栏内。（每空1分，共40分）

1.二进制数1101110.01的十进制数表示是_____，十六进制数表示是_____；十六进制数50A.E的十进制数表示是_____，二进制数表示是_____，八进制数表示是_____。

供选择的答案：

A. 110.25　B. 108.625　C. 1290.85　D. 1280.85　E. 2412.7

F. 1414.7　G. 3D.25　H. 6E.4　I. 10100001010.111

J. 101001100.0111

2.设数X=0.1001　Y=-0.0101，则[X+Y]补=_____，[X-Y]补=_____。

供选择的答案：

A. 10.0010　B. 0.0100　C. 1.011　D. 0.1110

3.在计算机中表示0时_____码是唯一的，且全为0。

供选择的答案：

A.国标　B. 原　C. 补　D. 反

4.浮点数取值的大小由_____决定，而浮点数的精度由_____决定。

供选择的答案：

A.阶码的位数　B.阶码的数值　C.尾数的位数　D.尾数的数值

5.设A,B,C为逻辑变量,则逻辑表达式$(A+B)(A+\bar{B})$可化简为_____,$AB+A\bar{C}+\bar{B}C$可化简为_____。

供选择的答案：

A. AB+C　B. AC+B　C. AB　D. A

6.对二进制数1011001进行奇校验，其校验位的值为_____。

供选择的答案：

A. 4　B. 2　C. 0　D. 1

7.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、中央处理器　D.主机、输入/输出设备、文件、外存储器

8.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，具有全部内部命令、外部命令的处理程序

9.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；命令字及格式拼写是否正确

C.查看命令拼写是否正确；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

10.若要把 A 驱动器内DOS系统盘上的文件内容全部拷贝到空白的硬盘上，应采用如下方法_____。

供选择的答案：

A.使用 COPY A:*.* C: 命令　B.使用 DISKCOPY A: C: 命令

C.先使用 SYS C: 命令，再使用 DISKCOPY A: C: 命令　D.先使用 FORMAT C:/S 命令，再使用 COPY A:*.* C: 命令

11.COPY文件操作意味着_____

供选择的答案：

A.在同一个介质内产生一个同名同内容的文件

B.产生一个同内容的文件

C.传送一个文件，原文件消失

D.只能在另一设备介质上产生一个不同名而内容相同的文件

12. 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
13.在微机操作过程中磁盘驱动器指示灯亮时不能插取磁盘的原因是_____。
供选择的答案：
A.会损坏磁盘驱动器　　B.可能将磁盘上的数据破坏
C.影响计算机的使用寿命　　D.内存中的数据将丢失
14.文件的组织包括文件的逻辑结构和物理结构两方面。文件的逻辑结构是指_____；物理结构是指_____。
供选择的答案：
A.一个文件内部各部分之间的关系　B.文件由记录组成或为有序字符的集合
C.文件在外存上的存放方式　　D.采用连续结构或链接结构的文件
15.使用结构化程序设计的方法，使得软件的_____，_____加强。
供选择的答案：
A.可读性　B.可靠性　C.可移植性　D.可操作性
16.按照国家标准GB2312－80规定，一个汉字由_____个字节组成，该码表共分_____区，每个区有____位，共有_____个汉字，第一个汉字在_____区的第一位。
供选择的答案：
A.2　B.3　C.9　D.16　E.85　F.94
G.3755　H.3008　I.6763　J.7445　K.8836
17._____不是计算机汉字处理过程中采用的数据结构。
供选择的答案：
A.字形表示　B.内部码　C.外部输入码　D.打印输出码
18.BNF是一种描述_____的语言，最先被用于_____语言。
供选择的答案：
A.高级语言　B.汇编语言　C.语法　D.算法　E. ALGOL　F.PASCAL　G.FORTRAN
19.数据库管理系统的功能不包括_____。
供选择的答案：
A.数据库构造　B.输入／输出调度　C.缓冲区管理　D.程序的存储分配
20.数据库中常用的数据模型不包括_____
供选择的答案：
A.层次模型　B.网络模型　C.系统模型　D.关系模型
21.一个过程文件中允许最多包含的过程_____
供选择的答案：
A.256　B.128　C.32　D.28
22.数据操作语言一般不具有的功能是_____。
供选择的答案：
A.从数据库中检索数据　　B.数据收集器和集中操作
C.向数据库中添加数据　　D.用于并发访问控制的操作
23.用数据描述语言精确地定义数据模型的程序称为模式，对应于不同的模型也有三个相应的模式，其中不包括_____。
供选择的答案：
A.线模式　B.外模式　C.概念模式　D.内模式
24.以下说法_____不正确。
供选择的答案：
A.数据库减少了数据冗余　　B.数据库避免了一切数据重复
C.数据库避免数据的不一致　　D.数据库中的数据可以共享
25.关系数据库操作功能不包括_____。
供选择的答案：
A.投影　B.选择　C.连接　D.抽象

二、填空，将对应的答案填写在答卷纸的对应栏内。（每空1分，共30分）
1.CPU是指计算机的_____和_____部分。
2.冯·诺依曼是_____国人，他为现代计算机结构奠定了基础，于1946年他首先提出了_____程序的思想，由其思想制做的计算机的特点是程序和数据_____指令_____执行。

3.CPU中的控制寄存器有各自的功能，如：程序计数器（PC）常用于存放_____执行指令的地址，指令寄存器常用于存放_____执行指令的指令码。

4.在一个指令周期中，CPU要执行操作的依次步骤是读指令、_____、取操作数、执行指令和_____。

5.指令的地址段指出的是操作数本身称为_____寻址方式，指令的地址段指出的是操作数的地址称为_____寻址方式，指令的地址段指出的是操作数地址的地址称为_____寻址方式。

6.DBMS是数据库管理系统的一组软件，它一般包括数据库管理功能、_____功能、建立维护功能和_____功能。

7.设某软盘规格如下：
存储面：双面，磁道数/每面：77，每个磁道的扇区数：20，数据长：128字节/扇区，那么该软盘容量是_____KB；如果一个扇区能存储一张卡片的数据，那么这张软盘能存储____张卡片的数据。

8.打印机是计算机系统的最基本的_____设备，按印字方式可分为击打式和_____打印，按工作方式可分为并行式和_____打印。

9.计算机系统中使用的输出设备CRT的中文意思是_____。

10.设有一个空栈，栈顶指针是1FF0H,现有输入序列2、5、0、7、6，经过操作push，push,push,pop,push,pop,push后，栈中存储数据（按地址存放的顺序）是_____，栈顶指针是_____。

11.汉字编码方案有多种，常见的拼音码是_____码，区位码是_____码，五笔字型码是_____码；上述三种编码可在_____种汉字系统上使用。

12.信息处理是指对信息_____和传播的一系列活动的总和。

13.对于微型计算机系统用户，最常见的计算机病毒传染媒介是_____。

三·填空，将相应的答案填写在答卷纸的对应栏内。（每空2分，共12分）

请在 A 盘上建立如图所示的树型目录结构：

1.请写出显示 A 盘上根目录下所有扩展名为 .PRG 的文件目录的命令。

2.设当前目录为根目录，一级子目录BAS已经建立，请写出建立二级子目录 BA1 的命令。

3.设当前目录是 BAS，请写出在二级子目录 BA1 下建立文件 AA.TXT 的命令。

4.设当前目录为根目录，请写出把子目录 BA1下的文件 AA.TXT 与子目录 DB3下的文件 AA.BAK 联接成一个名为 ALL.BAK 文件，存放在子目录 BA2 下的命令。

5.设当前目录为BAS，请写出把子目录BA2下的文件ALL.BAK改名为ALL.TXT的命令。

6.设当前目录为 BAS，请写出删除子目录 BA1 下的文件 AA.TXT的命令。

四·设当前驱动器为 A，请在 B 驱动器内的盘上建立一个名为 PC.BAT 的批处理文件，它具有如下功能：（共8分）

1.清屏幕；

2.暂停，用中文显示考生姓名及工作单位；

3.复制 A 驱动器内系统盘上的 TREE.COM 文件到用户盘的根目录下；

4.在屏幕上显示用户盘的所有目录路径及子目录中的文件名；

5.在屏幕上显示本批处理文件的内容；

6.文件结束。

五·（共20分）

已知AAA.DBF数据库的结构和记录如下：

数据库结构：　数据库名：　C:AAA.DBF

数据库中的数据记录个数：　　8

数据库最后一次更新日期：01/01/80

序号	字段名	类　型	宽度	小数位
1	姓名	字 符 型	6	
2	性别	字 符 型	2	
3	出生日期	日 期 型	8	
4	民族	字 符 型	4	
5	数学	数 字 型	5	1
6	语文	数 字 型	5	1
7	平均分	数 字 型	6	2
** 总计 **			37	

数据库记录：

Record#	姓名	性别	出生日期	民族	数学	语文	平均分
1	王红	男	12/23/78	汉	98.0	78.0	0.00
2	乌冰	女	03/07/65	蒙	98.0	67.0	0.00
3	李夺	女	10/23/77	维	87.0	78.0	0.00
4	买买切	男	02/25/64	维	99.0	94.0	0.00
5	赵充	男	06/25/69	蒙	54.0	87.0	0.00
6	卓仁	女	07/26/45	藏	78.0	58.0	0.00
7	杨林	女	09/07/55	汉	97.5	87.0	0.00
8	李洋	男	06/17/64	维	78.0	63.0	0.00

1.假设数据库AAA.DBF已经打开，在答卷纸对应栏内写出完成下列要求的命令：（每个命令1分，共8分）

①填充数据库AAA.DBF中平均分字段。

②把记录指针移到第一个平均分大于90分的记录上。

③复制数据库AAA.DBF中全部姓王的同学的记录，新文件名为BBB.DBF。

④对数据库AAA.DBF 中的性别和平均分字段进行索引，新文件名为CCC.NDX。

⑤显示数据库AAA.DBF中少数民族名单，且不带记录号。

⑥将数据库AAA.DBF中姓名、性别、平均分字段的结构拷贝出来，新文件名为DDD.DBF。

⑦将 6 5 年底以前出生的人员情况显示出来。

⑧将数据库AAA.DBF 复制成标准格式数据文件，新文件名为EEE.TXT。

2.请在答卷纸对应栏内顺序写出执行下面各命令后记录指针指向的位置（共 4 分，每空 0 . 5 分）

```
.USE AAA
.2
.SKIP -1
.SKIP
.SKIP 2
.GO TOP
.LIST NEXT 8
.LIST
```

3.请在答卷纸对应栏内写出下面程序或命令执行后的运行结果（每空2分，共8分）

①程序如下：

```
SET TALK OFF
USE AAA
DO WHILE RECNO()<6
   IF .NOT. 民族='维'
      SKIP
      LOOP
   ENDIF
   DISPLAY 姓名
   SKIP
ENDDO
```

②程序如下：

```
SET TALK OFF
USE AAA
DO WHILE .NOT. EOF()
   IF RECNO()<6
      LIST 姓名 FOR 民族='维'
   ELSE
```

```
        EXIT
      ENDIF
    ENDDO
```

③命令序列：

```
.USE AAA
.COUNT FOR 性别='男' .AND. 民族='维' .OR. 民族='蒙' TO A2
.?A2
```

④命令序列：

```
.USE AAA
.COUNT FOR 性别='男' .AND. (民族='维' .OR. 民族='蒙') TO A3
.?A3
```

六．阅读下面的BASIC程序，将运行结果填写在答卷纸的对应栏内。(每小题3分，共12分)

①程序：

```
10 M=1:A=1
20 PRINT A;" ";
30 M=M+1:A=A+M
40 IF M>10 THEN 60
50 GOTO 20
60 END
```

②程序：

```
10 INPUT N                                  RUN
20 FOR I= 0 TO 2*N                          ? 2
30 PRINT TAB(20+ABS(N-I)-N);
40 FOR J=N TO ABS(N-I)STEP -1
50 PRINT RIGHT$(STR$(J),1);
60 NEXT J
70 IF J=N-1 THEN 110
80 FOR P=J+2 TO N
90 PRINT RIGHT$(STR$(P),1);
100 NEXT P
110 PRINT
120 NEXT I
130 END
```

③程序：

```
10 DATA -2
20 DEF FNA(X)=X*X-4
30 DEF FNB(X)=5
40 READ G
50 D=FNA(G)*SGN(G)+FNB(G)
60 DEF FNC(X)=FNA(X)-FNB(X)
70 PRINT FNA(D)
80 PRINT FNB(D)+FNC(D)
90 END
```

④程序：

```
10 READ N,P,X
20 GOSUB 50
30 PRINT P+X
40 END
50 IF N=0 THEN 80
60 N=N-2
70 GOSUB 50
80 P=P+X
90 PRINT P;
100 RETURN
110 DATA 6,2,3
```

七．下面是求300以内被6除余2，被7除余3的整数的个数及其累加和的BASIC程序，请填补空缺语句或表达式。（每空1.5分，共6分）

程序：

```
10 S=0:N=0
20 FOR I= 1 TO 300
30 IF (I-2)/6<>_______ THEN 70
40 IF (I-3)/7 <> _________THEN 70
50 PRINT I,
60 S=_____:N=_____
70 NEXT I
80 PRINT
90 PRINT "个数N=";N,"累加和S=";S
100 END
```

八．（每小题6分，共12分）

新华商店第一季度销售复印机情况如下：

品名	零售价	销售量（台）	销售额（元）
理光FT-5560	68000	15	
理光FT-4085	32800	6	
施乐1027	29850	7	
佳能NP-3825	49000	10	
东芝BD-5511	24800	8	
优美3200	32000	11	

在答卷纸上完成

1.补充完成下面的BASIC程序，使其具有统计第一季度总销售额及上交税额（税额＝销售额的20%），并按照销售额从高到低输出复印机销售情况表（如下）：

品　名	零售价	销售量	销售额
理光FT-5560	68000	15	1020000
佳能NP-3825	49000	10	490000
优美3200	32000	11	352000
施乐1027	29850	7	208950
东芝BD-5511	24800	8	198400
理光FT-4085	32800	6	196800

第一季度总销售额：2466150元

上交税额：493230元

2.画出相应的程序流程图。

BASIC程序：

```
10 CLS:PRINT:PRINT:PRINT
20 DIM A(6,4),X$(6)
30 FOR I=1 TO 6
40 READ X$(I)
50 FOR J= 2 TO 3
60 READ A(I,J)
70 NEXT J
80 NEXT I
90 FOR I= 1 TO 6
100 A(I,4)=___________
110 S=S+A(I,4)
120 NEXT I
130 T=S*.2
140 FOR I= 1 TO 5
150 FOR J= _____ TO 6
160 IF ___________ THEN 200
170 SWAP X$(I),X$(J)
180 FOR K=2 TO 4:SWAP A(I,K),A(_____):NEXT K
200 NEXT J
210 NEXT I
215 PRINT "品　名","零售价","销售量","销售额"
220 FOR I= 1 TO _____
230 PRINT X$(I),
240 FOR K=2 TO 4:PRINT A(I,K),:NEXT K
```

```
245 PRINT
250 NEXT I
260 PRINT "第一季度总销售额:";S;"元 "
270 PRINT "上交税额:";_______;"元"
280 DATA "理光FT-5560",68000,15,"理光FT-4085",32800,6
290 DATA "施乐1027",29850,7,"佳能NP-3825",49000,10
300 DATA "东芝BD-5511",24800,8,"优美3200",32000,11
310 END
```

九·从供选择的答案中选出正确答案，把相应的字母写在答卷纸的对应栏内。（每空1分，共10分）

1.A total computer system includes both hardware and _____.
2.Key elements in a computer system include input, processing, and ___ devices.
3.One frequently used tool is the program _____.
4.BASIC is now by far the most popular high-level _____used in microcomputer system.
5.Softwaer is a set of _____that tell the computer what to do.
6.String _____too complex.
7.WARNING: All _____ on the disk will be lost.
8.The _____ capacity of this machine is equal to 10 MB.
9.UNIX is an _____ system developed at BeLi Laboratories.
10.Insert new diskette for drive B and _____ any key when ready.

供选择的答案：

A. Strike　B. memory　C. data　D. softword　E. language
F. operating　G. formula　H. output　I. flowchart　J. instructions

（考题答卷见 48 页）

给 TANGO 软件增加汉字元件库

黄健　陈军

TANGO 是原理图设计、连线网络表生成，印刷线路板自动布线等 CAD 及光绘、钻孔 CAE 于一体的多功能软件。其特点是“方便、易学、实用、快速”，自动布通率在 95%以上。但美中不足的是 TANGO 软件不能在中文状态下运行。笔者在充分研究了 TANGO 软件的元件库源文件(扩展名·SRC)后，编写了一个 BASIC 实用程序，该程序能将汉字图形库自动地转换成 TANGO 软件的元件库源文件，通过 TANGO 汇编后可直接供 TANGO 绘图时使用。从而使 TANGO 能编辑出各种带有汉字的图形，如程序框图等，附图是笔者在项目开发设计时通过 TAN—GO 绘制的面板图。

一、 程序的使用方法

1、进入 BASIC 状态，装入程序并运行。(如果程序能编译的话，则使用起来更方便)。

屏幕提示：filename?

这时应输入欲建立的汉字元件库的文件名。输入文件名后按回车，程序便进行自动转换。程序运行结束后便得到了元件库源文件。

2、运行 TANGO 的库文件汇编程序 COMPILE.EXE 对刚才建立的汉字元件库源文件进行汇编(注意：元件库源文件的扩展名必须是·SRC)，汇编完后便得到汉字元件库文件(扩展名·LIB)。

二、程序说明

20 句中的 C1 为欲转换的汉字库的起始区位码，程序中 C1＝0101 包含了汉字库中的符号图形，如果不要符号图形，仅转换汉字，可使 C1＝1601。

1204 句中的 8794 是汉字库中了最后一个汉字的区位码。值得注意的是 TANGO 的汇编程序一次只能汇编 2000 个元件，读者可将一个汉字库分几次转换。

60～140 句为一子程序，作用是根据区位码获得汉字的点阵信息。

(因篇幅有限，程序清单不再登出，需要者请寄 5 元复印及邮挂费来本刊索取。)

一九九一年计算机初级程序员有奖竞赛试题答卷

总分：_____

姓名：_____性别：_____年龄：_____文化程度：_____邮政编码：_____

工作单位：_____通信地址：_____

注意事项：

1. 答案必须在十月四日寄出，以邮戳为准(通信地址：北京学院路 31 号中间软件行业协会考试指导中心。邮政编码 100083)。
2. 填写答案时，必须将试题号与对应的答案栏对准，绝勿弄错，若某试题有多个答案时，请顺序给出(共 150 分)。
3. 解答时字迹务必清楚，字迹不清楚时不能得分。
4. 是否愿意参加复赛：是____，否____。

题号	小题号与答案										得分	
一	1		2		3		4		5		12	40
	6		7		8		9		10		5	
	11		12		13		14		15		8	
	16		17		18		19		20		10	
	21		22		23		24		25		5	
二	1		2		3						8	30
	4		5		6						7	
	7		8		9						6	
	10		11		12						9	
	13									1		
三	1											
	2											
	3											
	4											
	5											
	6											
四											8	
五	1	① ② ③ ④ ⑤ ⑥ ⑦ ⑧										20
	2										4	
	3	①		②		③		④			8	
六	①										3	12
	②		③		④						9	
七												6
八	①										6	12
	②	请另外附纸画出流程图									6	
九	1		2		3		4		5		5	10
	6		7		8		9		10		5	

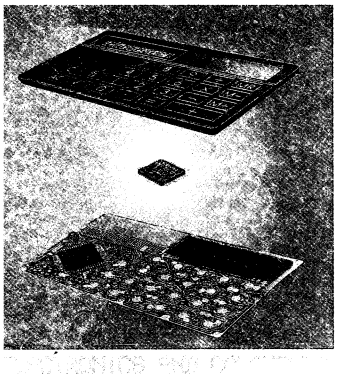
電子與電腦

ISSN 1000-1077

9

一九九一年

总期第78期



電子與電腦

目　录

·综述·

参加91'第三届国际信息学奥林匹克(IOI'91)竞赛的体会 …… 吴文虎　薛宏熙(2)

众说纷云:8位学习机是否已过时? 专家认定:君在教育界仍是主力军! …… (3)

·PC用户·

IBM PC/XT 定时中断服务程序设计 …… 张辉　李东升　李奎贤(5)

通用数据库修改程序 …… 周日初(7)

PC-DOS 命令 WHEREIS 的改正 …… 李　敏(8)

巧算1000! …… 李　帆(9)

又发现一种新病毒 …… 王家春(10)

杨基病毒快速解毒法 …… 李风明(10)

设备驱动程序 ANSI.SYS 的使用 …… 王立勇(11)

巧用虚拟盘 …… 李焕民(12)

用递归解九连环 …… 钟　武(12)

·学习机之友·

将 CEC-I 中华学习机改制成专用机 …… 姜良夫(14)

用栈结构优化程序 …… 陈合力(17)

寻找无理数 e 的约率和密率 …… 施为钢(17)

怎样节省 CEC-I 存储空间 …… 尹向阳　梁　曦(18)

ProDOS 磁盘操作系统入门(续) …… 廖　凯(19)

数制转换简法 …… 李　锋(20)

也谈扩展 COPY][PLUS 4.0 …… 关炳坤(21)

从 CP/M 转移文件到 APPLE DOS 下 …… 方福兴(21)

多功能 CEC-I 高速排序 …… 王庆顺(22)

小学生速算训练 …… 李永升(23)

·C 语言初阶讲座·

第七讲　结构数据(二) …… 李文兵(24)

·学用单片机·

MCS—51 单片机应用中二种 LED 显示方法 …… 傅　篱(27)

·学装微电脑·

相位控制部件的制作 …… 易齐干(29)

·电脑巧开发·

怎样用电脑检修录像机 …… 李玉全(32)

用娃娃机开发单片机程序 …… 张　涛(32)

·维修经验谈·

TH3070 打印机接口电路工作原理与故障分析 …… 李燕华(35)

长城 0520CH 机打印机适配器故障维修 … 邱　槟(37)

·初级程序员级水平考试辅导讲座·

汉字 dBASEIII 使用技巧(下) …… 王路敬(39)

91 年第三届国际信息学奥林匹克(IOI'91)试题 …… 吴文虎(46)

Z80 的"新"指令 …… 章永得(47)

封一:薄型计算器的内幕

封二封三:Z80 的"新"指令

封四:家庭电脑教学键盘

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

参加91'第三届国际信息学奥林匹克(IOI'91)竞赛的体会

领队　吴文虎　薛宏熙

今年5月19—25日在希腊的雅典举行了第三届国际奥林匹克信息学竞赛，有23个国家的74名选手参赛。由中国科协，国家教委，国家自然科学基金会委托中国计算机学会主办参赛选手选拔，集训和赴希腊参赛全过程。中国队的3名选手全部获奖，金牌两块，银牌1块，总积分名列第一。

一、普及是基础，组织是关键

学科奥林匹克的宗旨是：促进交流，建立友谊与联系，推动优秀人材的成长。中国队本着这一宗旨，力求反映中国青少年努力学习先进科学知识，奋发向上，全面发展的精神风貌，第三次出国参赛。在希腊的两轮竞赛中水平得到比较正常的发挥，取得了比较好的成绩。评委和各国领队们说：中国队是IOI中最强的队。在闭幕发奖会上组委会表场了中国队，说从解思路，编程能力到调试速度上，中国的三名选手都是非常出色的。

这次取得比较好的成绩不是偶然的，因为我们既有普及基础，又有良好的组织和训练。

从组织上讲，我们有得天独厚的优越条件，那就是党和国家高度重视，计算机的普及从娃娃抓起，有全国性的竞赛活动作为选拔高水平选手的依据。这次去希腊参赛的三名选手是经两次选拔产生的。第一次是全国统一命题的通讯赛。各省(自治区，直辖市)设考场，同日同时开始测试。每个省市的前三名的成绩和答卷(含有解题程序的软盘)报到学会，由学会统一测试，排序，选出参加复试的选手；第二次11名选手集中北京，由学会主持参加第三届国际信息学奥林匹克中国队选手的选拔赛(见中国计算机学会通讯(91)第2期)。通讯赛的题目和选拔赛题目都有相当难度。从这两次竞赛看91年全国青少年计算机竞赛中的高手们较之90年又有了比较大的进步，表现在掌握计算机语言，算法知识，数据结构知识，以及解题的能力上。91年的组队有了比90年更为广阔的基础，挑选人才的余地增大了。

二、训练中加强针对性，重在提高能力

在训练的策略思想上强调加强针对性，"对症下药"，有目的有目标地进行培训。一是我们分析了选手的情况；二是预测了赛事情况进行有针对性地准备。

选手的情况是：两名老选手杨澄(清华附中，高二)，杨云和(上海延吉中学，高二)曾参加IOI'90，分获银牌和铜牌，新选手是夏西远(南京师大附中，高二)。两名老选手可能有的优势是：经历过一次世界级别的大赛，见过世面，有了胆识，获得了如何备战的经验；但也可能会背上包袱，脑子里会形成固定的解题模式，临场放不开手脚。这两位选手在上述的全国通讯赛和组队选拔赛就有这方面的表现，比如选拔赛中的第一题，本来可以用"宏"的思想来做，这两名选手在90年的中国队培训中曾经学习过这种思路，但偏偏在选拔赛中不敢用这种思路解题，反映了一种套搬公式，求稳而不求新的思想，创造才能发挥不出来。我们首先加强心理素质的训练，让选手从零开始，以首次参赛的心态去迎接IOI'91。要求每一个选手要力争上游，瞄准世界第一的目标；顽强拼搏，发挥最佳的创造才能；勤奋学习，努力实践，掌握真本事，把必胜的信心建立在解题能力的稳步提高上。

为了提高解题能力，采取了四项措施：

1. 引导选手"温故知新"，精做通讯赛和选拔赛中的三道难题，深入钻研，总结更好的思路与方法；

2. 自学一些参考资料，学一些有关图论和组合数学的知识，将一些解题方法上升到理论高度，以起到举一反三的作用。

3. 扩充题型进行练习，以掌握广博的知识和算法；

4. 掌握最新的语言版本，提高编程与调试效率。

为了应付赛场上出现的一些特殊情况，训练与掌握所应采取的对策与思路，譬如在第二届IOI上，解第一题普遍出现由于算法的数据量大而溢出的问题，我们从两个方面引导选手：一是开数组时，选用大模式；二是当运行时间或搜索步数超过某个阈值时，转而采取不致出现溢出的"笨方法"。当然，这些方法要结合赛题的实际当场灵活运用。

通过短时间的赛前训练，从总体上看中国队的水平在以下四个方面有所提高：

1. 拼搏与竞争的意识增强了；
2. 解题的思路拓宽了；
3. 编程调试与自测能力提高了；
4. 应变的能力增强了。

三、试题难度加大，训练的视野尚须扩大

这届竞赛第一轮题目比较容易，这道题要求据三

个关系式往 5 * 5 的矩阵中填入数字，编程输出所有可能的方案。中国的三名选手采用深度优先的思路来解这道题，程序风格和屏幕效果都很漂亮，三个人都获满分。第二天组委会有意将第二轮赛题难度加大，经仲裁委员会表决后选出的一道题目是用递归方式定义 S 项，S 项是字符 S 和括号组成的序列，对 S 项的子项形式给出了一个缩简规则，要求写出生成，读，显示，缩减等五个过程，而且要求程序能演示每一个过程，最后将上述过程连成一个程序，将生成的 S 项以规格化的形式输出。这道题目涉及到编译原理中的某些内容，要编程的内容多，给的时间短，想不到用什么数据结构就做不出来。中国队的杨澄和夏西远选择二叉树的结构，杨云和选择表达式化简的方法，都做得比较好。杨澄编了 400 行 PASCAL 语句，杨云和编了 420 行 C 语言的语句。每小时平均 100 行。负责对中国队评分的希腊评委看了我们三位小将的程序，翘起了大拇指说：中国的孩子编出的程序，真像是专业人员编的。第二轮赛题的评分有的队进行得很快，中国队拖的时间很长，从下午 2:30 到 6:30。后来才知道谁做得多，自然被测试的项目就多，测得就细，一点一点扣，那真是“一丝不苟”，拿高分绝对不容易。

第二轮竞赛未被选中的题目有的难度可能比这道题还要大。其中的一道是涉及各国文字文本字频统计的，另一道是涉及医院门诊预约的。从题型看涉及的领域广，灵活多变，这从一个侧面反映了计算机的应用渗透到了社会生活的各个领域的发展状况，引导孩子们了解这方面的情况，开阔视野，对于启发思路是会有好处的。

四、仍然需要站到起跑线上

在希腊逗留十天，有幸到奥林匹克的发源地，踏一踏马拉松发源地的起跑线，感慨万千。想到普及计算机教育的事业也应该有锲而不舍的马拉松精神。从体育的奥林匹克到智力的奥林匹克，世界发生了多么大的变化！几届数，理，化，信息学奥林匹克又连连报捷，长了中华儿女的志气。众所周知，国际间的竞争说到底是综合国力的竞争，关键是科学技术的竞争。科学技术是第一生产力，科学技术需要人去掌握，去运用，去发展，人才又是至关重要的。科技进步的基础在教育，我们应该把开展学科奥林匹克和普及计算机教育，提高未来四化建设人才的素质紧密联系起来。要看到我们的不足，更要激发我们的勇气，作为教练和领队，我们从小将身上也学到不少东西。前三届国际信息奥林匹克取得的成绩已成为历史，第四届将于 92 年 7 月在波恩举行，我们想：中国队还应认真准备，从零开始，重新站到起跑线上，信心来源于勇气，来源于实力，来源于千百万孩子，教师与家长的支持，可以这样说：在写这篇总结时，我们是满怀信心的。

五、值得总结的一条经验

参加学科奥林匹克竞赛，参赛选手和领队在精神和体力上负担都很重，还要克服时差、环境等造成的困难。为了保证选手有充沛的精力投入紧张的两轮竞赛，中国科协多方设法联系机票和选定转机路线，使代表团能提前两天到达雅典，给选手以恢复体力的充分时间，更重要的是，使我们能及时了解到竞赛规则的变更细节，有助于我们及时调整参赛的策略，为取得良好成绩创造了有利条件。否则，将会对我们极为不利。因此，我们建议下一届竞赛，时间上仍要留有充裕余地。

众说纷纭：8 位学习机是否已过时？专家认定：君在教育界仍是主力军！

北京计算机五厂专家座谈会纪要

最近在社会上，特别是学校、家庭中，讨论以 6502 芯片为主体构成的 8 位学习机是否过时了，是否应该用 16 位的 PC 机取代这一问题成了热门话题。有些学校急需配置教学用计算机，有些学生家长也持币等购，但被众说纷纭的议论弄得主意难定。购现在流行的中华牌或天坛牌学飞机，又怕真的会很快被 PC 机淘汰；若购 16 位 PC 机又怕目前教育软件的匮乏而影响盼子成龙之大计不能成就，究竟买什么样的学习机真是左右为难。

为了帮助学校及家庭将有限的资金用于购置有用的设备，使之发挥最大效率，真正实现少花钱办实事的目的。六月下旬，由生产北京市优质产品天坛学习机的北京计算机五厂发起并邀请一部分计算机专家及教育界的教授、学者，就以上问题召开了一个专题讨论会。与会者各抒己见，经过热烈的讨论大家一致认为：

一、8位机并不过时，在中国如此，在教育界更是如此。

从改革开放至今，我国计算机引进有过三次高潮，其中8位机、准16位机占相当大的比重，我国市场上8位机的拥有量达50万台，因此分析、掌握、应用8位机是我们的基础工作。特别是在教育领域，目前中小学配置的大部分计算机是8位机，国产化的中华学习机和天坛学习机的生产已逾十万台，为中小学开展计算机教育提供了硬件环境。在谈到五年前教学用计算机选型时，国务院电子信息推广办公室副主任（高级工程师）陈正清同志指出："当年在确定机型时选择8位机，而不是16位机，就是从中国的国情出发，并已经考虑到满足教学的需要，否则为什么不选择4位机呢。当年教育界和工业界共同商定把8位机作为学习机生产和开发的应用机型，现在还没有到修改、更正的时候，政策上要有稳定性。"

大家一致认为：虽然计算机教育发展这么多年了，但是从全国看，中小学计算机教育总数并不高，全国有计算机的中小学还不到10%，这说明全国经济情况发展的不平衡。北京属于发展迅速普及较好的地区，但仅中学的配置还需翻两番才能达到标准。因此，从国情上看，我们是一个比较穷的国家，另外从我国目前中小学计算机教育内容来看，8位机还有相当大的潜力需要挖掘，以提高它的使用率。所以说在教育领域8位机是完全可以胜任的。

机电部微机处处长、高工于万源同志一席话，给人以很深的印象，发人深省，他认为："中国目前的教育水平不见得一定要用16位机最好。就象小学生不能一开始就用计算器一样，因为忘掉了数学本身的发展规律，仅仅变成使用"傻瓜"的畸形，孩子是不会脱颖而出的。只有知道了事物的发展过程，再使用先进的科学技术和工具才能有更大的创造。因此从这个意义上讲，用8位机甚至更低档的计算机对孩子进行启蒙教育是很有益的。"事实也确是如此，我们连续三届参加奥林匹克竞赛，参赛学生都是8位机培训的，只在出国前一个月才上PC机培训，但也同样取得了很好的成绩。因此与会者认为在教育领域8位机不但没有过时，而且是当之无愧的，今后的工作不是考虑换代的问题，而是应该考虑怎样更好的利用8位机为教育界服务，充分挖潜，在应用软件的开发上下功夫。

二、不能忽视8位机有丰富的软件资源、技术资源这一事实。

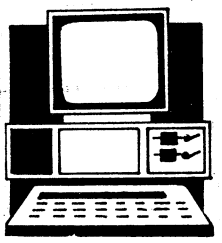
五年来我们在软件开发、应用方面做了很多工作，且90%的软件是在8位机上开发的，这些软件特别是教学软件已经比较符合当前中小学的教学环境，而且由于这些年计算机的普及，使部分中小学都是能够接触和使用计算机，并且在开展8位机的辅助教学方面有所建树。今明两年北京市将在四所中学、二所小学进行计算机辅助教学实验，实验内容范围很广泛，几乎涉及各个学科，而所有教学软件也都是在8位学习机上开发的。全国中小学计算机研究教育中心从87年开始由机电部投资30万元作为"七五"攻关项目，设立了软件评审制度。这些软件在中小学教学中发挥了很大的作用。所有这些工作为8位机在教育界推广应用提供了良好的软件环境并打下了基础。

正如陈正清同志所指出的："计算机技术发展到今天是以软件为核心的，这是一个趋势，也是现实。"如果硬件设备很好，而师资没有，与之配套服务的软件没有，那么计算机将是一事无成的。我们搞教学软件在选定教材方面的考虑不是一年两年了，而是相当多年积累起来的，是经过循环往复的实践、反复修改才成功的，这是一笔宝贵的财富，所以我们不应该轻易地抛弃自己的财富，要重视人类的一切资源，要珍惜自己的劳动所得；另一方面目前PC机上汉化的教育软件几乎没有，如果花很多的精力在PC机上去重新开发或移植教育软件不仅是要花许多人力、物力和时间，而8位学习机上的丰富软件以及硬件就会被丢弃，造成狗熊掰棒子的后果，延误计算机辅助教学及中小学计算机普及事业的发展。与会者一致认为对这种倾向，教育界要重视，工业界要重视，搞应用开发的也要重视，要把中小学计算机普及的工作扎扎实实地搞下去，充分用好现有8位学习机及软件资源，切忌见异思迁和不切实际的喜新厌旧。

三、向社会开放、全面发展。

在谈到向社会开放这一题目时，于万源同志深有感触地说："我们在软件开发方面还是做的不够，特别是在向社会开放，收集更多的软件等方面做的不够。尽管教委、科协、机电部做了大量工作，但光依靠专家、生产厂家是不够的，更重要的是要向社会开放，向用户开放。"我们知道苹果公司之所以拥有如此丰富的软件资源，很重要的一点就是借助社会力量发展自己的。在这一方面北京计算机五厂搞的很好，他们去年成立了"天坛机应用学组"，组织用户之间的技术、信息的交流，而且每年向社会开展学习机的应用成果有奖征集活动。通过这一活动首先鼓舞了社会上一大批人员，使他们认为国家需要，重视这一机型的发展问题，同时自己的劳动成果能够被肯定，且为社会作出贡献；从工厂的角度看，投资不多，收益很大，产生了异想不到的效果。

参加会议的有清华大学教授朱家维、北师大付教授张士英、北师院教授贺龙光、北京市电子振兴办高工华平澜、全国中小学计算机教育研究中心陈星火，一些新闻界同志也参加了讨论会。



PC用户

IBM PC/XT 定时中断服务程序设计

东北工学院机械二系　张辉　李东升　李奎贤

近几年来，PC 系列微机除用于办公室管理和科学计算外，还越来越多地用于自动控制、在线监测、故障诊断及数据采集处理、频谱分析等各个领域。这些应用在大多数场合下均要求应用程序有定时处理功能，如 A/D、D/A 转换板的定时采样和输出等。有些程序中还要求具有时钟功能。本文针对 IBM PC/XT 系列微机介绍定时中断服务程序的编写方法。

一、8253 定时器的编程

IBM PC/XT 机内装有一片 8253 定时/计数芯片。该芯片内有三个彼此独立的定时/计数器，每一定时器通道都已被系统指定为特殊用途。通道 0 的输出线连到系统板的 IRQ0 线上，使其能产生 0 级中断（INT 08H）；通道 1 用来发出数据请求信号，供存储器动态刷新用；通道 2 的输出接到机内扬声器，以产生声音效果。

定时器每一通道的时钟输入连到一个 1.19318MHz 的晶振上。因此，定时器每一时钟周期是 1/1.19318MHz＝840 毫微秒。通道 0 在计算机上电时被 BIOS 初始化，输出频率为 18.2Hz（定时常数为 65536），这个周期性中断被 BIOS 用来保持日历钟的走时。在定时器的三个通道中，只有通道 0 可供我们作定时中断用。如果我们在编程定时器中断时不需顾及日历钟，那么就可以重新对其进行编程。但通道 0 的中断还控制软磁盘的 I/O 操作，所以在对通道 0 编程后是不能进行磁盘的 I/O 操作的，完成中断服务任务后必须恢复通道 0 的原定时参数（18.2Hz）。

8253 定时器各寄存器的口地址为：

定时器通道 0 锁定寄存器存取口：40H；

定时器通道 1 锁定寄存器存取口：41H；

定时器通道 2 锁定寄存器存取口：42H；

定时器命令寄存器：43H。

修改定时器通道 0 参数的汇编语言程序如下：

```
MOV AL,36H          ;定时器命令：选通道 0，读写
                    ;LSB—MSB（低 8 位—高 8
                    ;位），二进制记数
OUT 43H,AL          ;命令发给 8253 命令寄存器
MOV BX,COUNT        ;定时器定时参数送入 BX
MOV AL,BL           ;先发送 LSB（低 8 位）值
OUT 40H,AL
MOV AL,BH           ;发送 MSB 值
OUT 40H,AL
```

定时器定时参数的计算公式为：

COUNT＝1193180/每秒中断次数（0≤COUNT≤65535）

二、中断处理及中断服务程序

掌握 8253 定时器编程方法后，还需知道中断处理和中断服务程序的编写规则。一个要准备中断的处理程序必须做到：

1. 若以前为开中断则应关闭中断，以防止修改中断向量；
2. 初始化要处理的中断向量，以指向自己的中断服务程序；
3. 若中断以前已被关闭，应保证其它中断向量指向一些有效的中断服务程序；
4. 再次开中断。

修改 8253 定时器通道 0 的中断向量就是要修改 INT 8H 的中断向量，通过调用 DOS 功能 INT 25H（置中断向量）和 INT 35H（取中断向量）来完成。若一个中断通过 PC 机内的 8259 中断控制器请求处理，则中断处理结束后，必须给 8259 的控制口发送一个叫做中断结束的特殊码 EOI（8259 控制口的地址为 20H，EOI 的值为 20H），通知 8259 中断结束。最后，中断服务程序执行特定的中断返回指令 IRET。中断服务程序的结构如下：

```
STI                 ;开中断
PUSH AX             ;保存中断服务中修改的寄存器
⋮
PUSH ES             ;内容
⋮                   ;完成中断服务工作的用户程序
MOV AL,20H          ;EOI 值；
MOV DX,20H          ;8259 控制口地址；
OUT DX,AL           ;将 EOI 送 8259
POP ES              ;恢复寄存器内容
⋮                   ;
POP AX              ;
IRET                ;中断返回
```

中断服务程序必须遵守以下简单而严格的步骤：

1. 保存系统内容（寄存器、标志等）；
2. 如果有些中断允许在该处理程序运行时发生，但这些中断会引起干扰，则应禁止这些中断；
3. 对该服务程序运行时允许的中断应开中断；
4. 确定中断原因；
5. 完成该中断要求的工作；
6. 恢复系统内容；
7. 对服务程序运行时封锁的各级中断再开中断；
8. 恢复运行被中断的过程。

在定时器中断服务完成并返回主程序后，还要将 INT 8H 原来的中断向量恢复，以便开始 BIOS 的时钟功能，对磁盘进行 I/O 操作。另外，在编写中断服务程

序时还需注意的是在中断服务中，硬件中断服务（如键盘、打印机、屏幕及通讯等）程序决不允许调用 DOS 功能，只能使用 BIOS 例行中断处理程序。

三、程序编写实例

为使读者能更快的掌握上述定时器中断服务程序的编写方法，笔者针对某一型号的 A/D 转换板编写了定时数据采集程序，供读者参考。其中数据定时采集程序用 8088 汇编语言编写，而数据处理及输出用 FORTRAN 编写。程序如下：

```
C       FORTRAN 语言采样主程序
        INTEGER * 2 CHN,AAR(1024),CCC(1024,100)
        INTEGER ACCOUNT
        CHARACTER * 20 OUTFIL,OUT1
        CHARACTER BI * 1,BC * 4,BC1 * 3
1       WRITE( * ,2)
2       FORMAT(1X,'请输入采样频率='\)
        READ( * ,'(I5)') IFREQ
        IF (IFREQ. LT. 19. OR. IFREQ. GT. 7100) GOTO 1
3       WRITE( * ,4)
4       FORMAT(1X,'请输入采样点数(1—1024)='\)
        READ( * ,'(I5)') NUM
        IF (NUM. GT. 1024. OR. NUM. LT. 1) GOTO 3
5       WRITE( * ,6)
6       FORMAT(1X,'请输入采样通道号(0—15)='\)
        READ( * ,'I5') CHN
        IF (CHN. GT. 15. OR. CHN. LT. 0) GOTO 5
7       WRITE * ,8)
8       FORMAT(1X,'请输入采样数据块数(1—20)='\)
        READ( * ,'(I5)') IBCK
        IF (IBCK. GT. 20. OR. IBCK. LT. 1) GOTO 7
        ACCOUNT =INT(1. 19318E6/IFREQ)
C   ACCOUNT 是 8253 定时器的定时常数
      WRITE( * , * )'请按任一键开始采样!'
      READ( * ,'(A1)') BI
      WRITE( * , * ) '请稍候.正在采样……'
      DO 10 II=1,IBCK
C   调用汇编语言采样程序
    CALL SAMPLE(AAR,ACCOUNT,NUM,CHN)
    DO 9 I=1,NUM
9   CCC(I,II)=AAR(I)
10  CONTINUE
    DO 11 II=1,IBCK
    DO 11 I=1,NUM
    IF (CCC(I,II). GT. 2047) CCC(I,II)=CCC(I,II)-4095
11  CONTINUE
    WRITE( * ,12)
12  FORMAT(1X,'采样结束,存储数据吗(Y/N)?'\)
    READ( * ,'(A1)') BI
    IF (BI. NE. 'Y'. AND. BI. NE. 'y') GOTO 18
    WRITE( * ,13)
13  FORMAT(1X,'请输入数据文件名='\)
    READ( * ,'(A20)') OUTFIL
    OPEN(6,FILE=OUTFIL,STATUS='NEW')
    DO 17 II=1,IBCK
    WRITE(6,14) II
14  FORMAT(1X,'BLOCK NO. ',I2)
    DO 16 I=1,NUM
    CCCC=2500./2047 * CCC(I,II)
    WRITE(6,15) CCCC
15  FORMAT(1X,F8.2)
16  CONTINUE
17  CONTINUE
18  CONTINUE
    STOP
    END
```

汇编语言采样子程序

```
        PUBLIC SAMPLE
FRAME   STRUC
SAVEBP  DW ?
SAVEDS  DW ?
RETADD  DD ?
CHN     DD ?
NUM     DD ?
ACCOUNT DD ?
AAR     DD ?
FRAME   ENDS
DATA    SEGMENT PUBLIC 'DATA'
CSS     DW ?
OFF     DW ?
DATA    ENDS
DEROUP  GROUP DATA
CODE    SEGMENT PARA 'CODE'
SAMPLE  PROC FAR
        ASSUME CS:CODE,DS:DATA
        PUSH BP
        PUSH DS
        MOV BP,SP
        PUSH DI
        PUSH SI
    LES DI,[BP]. CHN          ;取采样通道号,
    MOV AX,ES:[DI]            ;存入寄存器 SI
    MOV SI,AX
    LES DI,[BP]. ACCOUNT      ;取 8253 定时器时间常数,
    MOV BX,DS:[DI]            ;存入寄存器 BX
    LES DI,[BP]. NUM          ;取采样点数,
    MOV CX,DS:[DI]            ;存入寄存器 CX
    LES DI,[BP]. AAR          ;取数据存放的首地址
    PUSH DI
    PUSH CX
    PUSH BX
    MOV AX,DATA
    MOV DS,AX
    CLI                 ;清中断标志,禁止一切中断
    PUSH ES
    MOV AX,3508H        ;取原 INT 08H 的中断向量
    INT 21H
    MOV AX,ES           ;中断向量的段地址存入存储器
    MOV CSS,AX          ;CSS,偏移量存入存储器 OFF
    MOV OFF,BX
    PUSH DS
    MOV AX,SEG SAM      ;取采样程序的段地址及偏移量
    MOV DS,AX           ;利用 DOS 功能 INT25H 使 INT08H
    MOV AX,2508H        ;的中断向量指向采样程序
    MOV DX,OFFSET SAM
    INT 21H
    POP DS
    POP ES
    POP BX
    MOV AL,36H          ;编程 8253 定时器通道 0
    OUT 43H,AL          ;
```

```
        MOV AL,BL           ;修改 8253 定时器通道 0
        OUT 40H,AL          ;的定时常数,使之产
        MOV AL,BH           ;生所要求的每秒中断
        OUT 40H,AL          ;次数
        MOV AL,0FCH         ;编程 8259 中断控制器
        OUT 21H,AL          ;允许定时器和键盘中断
        POP CX
        POP BX
        STI                 ;开中断
CC1:    CMP CX,0            ;检查是否采完所有数据,
        JNZ CC1             ;未采完继续等待
        CLI                 ;清中断标志
        MOV AX,CSS          ;取出原 INT 08H 的中断向量
        MOV DX,OFF          ;恢复原中断服务程序
        MOV DS,AX
        MOV AX,2508H
        INT 21H
        MOV AL,36H          ;恢复 8253 定时器通道 0
        OUT 43H,AL          ;的定时常数
        MOV BX,65535
        MOV AL,BL
        OUT 40H,AL
        MOV AL,BH
        OUT 40H,AL
        MOV AL,0
        OUT 21H,AL
        STI                 ;开中断
        POP SI
        POP DI
        POP DS
        POP BP
        RET 16              ;返回 FORTRAN 主程序
SAMPLE ENDP
;采样程序—中断服务程序
SAM     PROC FAR
        MOV AX,SI
        MOV DX,2A0H
        OUT DX,AL
        INC DX
        XOR AX,AX               ;采样过程
        OUT DX,AL
        INC DX
S1:     IN AL,DX
        AND AL,1
        JNZ S1
        DEC DX
        IN AL,DX
        AND AL,15
        MOV AH,AL
        DEC DX
        IN AL,DX
        MOV ES:[BX],AX
        INC BX
        INC BX
        DEC CX
        MOV AL,20H              ;通知 8259 中断控制器
        OUT 20H,AL              ;中断服务完毕
        IRET                    ;从中断返回
SAM     ENDP
CODE    ENDS
        END
```

通用数据库修改程序

湖北天门市 121 号信箱　周日初

用 FOXBASE 编写较大软件时，为了屏幕的美观和软件设计的完整，采用窗口，以便把一些有用的信息始终保留在屏幕上，因此对数据库的操作并不用 EDIT、BROW 等全屏幕操作命令，而专门编写一些程序来取代它们。但象这样做，对不同的数据库或不同结构的数据库都要编一个修改程序，不仅浪费磁盘空间，还加大了编程的工作量。为了克服这些缺点，本人编了一个数据库的通用修改程序 TYXG.PRG，可以作为一个过程进行调用，调用时只需给出要修改的数据库名(后缀省略)、显示范围即左上角的位置(行、列数)和右下角的位置(行、列数)即可。程序清单附后，如要对数据库 SJK.DBF 进行修改，显示范围从第 2 行 20 列开始至 20 行 78 列时，调用格式为：

DO TYXG WITH "SJK",2,20,20,78

当数据库的字段数不超过 100，字段长不超过 80 时，可直接调该程序。当字段数大于 100 时，需对数组变量 VXX 重新定义，字段长超过 80 时，需另作处理。

该程序已在 AST 286/386 上运行通过。

```
PARA VNA,VM1,VN1,VM2,VN2
SET SCOR OFF
SET MENU OFF
SET TALK OFF
SET STAT OFF
DIME VXX(100)
IF VM1>VM2.OR.VN1>VN2
  RETU
ELSE
  IF FILE(TRIM(VNA)+'.DBF')
    USE &VNA
  ELSE
    RETU
  ENDI
ENDI
VI=1
DO WHIL LEN(TRIM(FIELD(VI)))<>0
  VXX(VI)=FIELD(VI)
  VI=VI+1
ENDD
VA=VI-1
IF VM2-VM1>VA
  VB=1
  VE=VA
ELSE
  VB=INT(VA/(VM2-VM1+1)+.9)
  VE=VM2-VM1+1
ENDI
```

```
DO WHIL .NOT. EOF()
  STOR 0 TO VC,VD
  DO WHIL VC<VB
    VD=0
    DO WHIL VD<VE. AND. VC * VE+VD+1<=VA
      STOR VXX(VC * VE+VD+1) TO VYY
      @VM1+VD,VN1 SAY VYY GET &VYY
      VD=VD+1
    ENDD
    READ
    VC=VC+1
  @ VM1,VN1 TO VM2+1,VN2 CLEA
  ENDD
  SKIP
  @VM2,VN1 SAY '继续修改吗(Y/N)?'
  WAIT '' TO VXG
  @VM2,VN1 SAY SPAC(VN2-VN1)
  IF UPPE(VXG)<>'Y'
    EXIT
  ENDI
ENDD
```

PC-DOS 命令 WHEREIS 的改正

天津电子计算机研究所　李　敏

DOS 命令 WHEREIS 用于搜索由用户定义的文件名，但这种搜索只能在硬盘中进行。例如，输入：

C>WHEREIS P*.*

WHEREIS 命令就会自动地搜索整个 C 盘，显示出所有以 P 字母开头的文件的位置(包括路径名，目录等)，这对用户查找文件是相当方便的。但是这儿也存在一个问题，即 WHEREIS 并没有显示文件的大小、日期和修改的时间。有时，同一个文件的不同版本分布在许多子目录下，如果没有时间显示，就很难判断哪一种版本是最新的。本文设计的 WHEREIN 命令可弥补这一点，而且功能还有所扩展，它能自动地搜索 A 盘和 B 盘(这可以在 WHEREIN. COM 生成过程中自动设置)。

WHEREIN 是在 WHEREIS 基础上开发的，该命令由一个 BASIC 程序 WHEREIN. BAS 生成，需要在 BASICA 解释程序下运行。如果你不想用 BASIC 程序，也可以在 DEBUG 环境下输入，具体步骤如下：

C>DEBUG WHEREIN. COM↙

(机器会显示"File not Found"信息，不必理会)。

然后输入：E 100↙，接下去，你可以利用文后的 BASIC 程序清单，输入 420 至 800 行 DATA 语句中的所有十六进制数，每一行 BASIC 数据对应一行 DEBUG 程序，BASIC 数据行中的逗号用空格代替。当你输入完毕就输入回车，程序输入就完成了。然后再输入 R CX，DEBUG 就会显示 CX 0000，最后依次输入：

```
:26B↙
-W↙
-Q↙
```

WHEREIN 命令就自动生成了。

```
110 DIM T(38):FOR A=1 TO 38:READ S:T(A)=S:G=G+S:NEXT
120 IF G=69210! THEN 140
130 PRINT "Error in lines 380-410":END
140 FOR A=1 TO 38:FOR B=1 TO 16
150 READ C$:C=VAL("&H"+C$)
160 TOTAL=TOTAL+C:NEXT
170 IF TOTAL=T(A) THEN 190
180 PRINT "Error in line ";A * 10+410:END
190 TOTAL=0:NEXT
200 RESTORE 420:CLS:KEY OFF:LOCATE 8,1,1:INS%=0
210 ADD%=4:DRIVE$="A":GOSUB 320
220 ADD%=2:DRIVE$="B":GOSUB 320
230 ADD%=1:DRIVE$="C":GOSUB 320
240 IF INS%=0 THEN INS%=1
250 LOCATE 16,30,1:PRINT "Writing Byte No. "
260 OPEN "WHEREIN. COM" AS #1 LEN=1:FIELD #1,1 AS BYTE$
270 FOR I=1 TO 618
280 LOCATE 16,50,0:PRINT USING "####";I
290 READ HEXTWO$:IF I=609 THEN HEXTWO$=MID$(STR$(INS%),2)
300 LSET BYTE$=CHR$(VAL("&H"+HEXTWO$)):PUT #1
310 NEXT:CLOSE #1:LOCATE 18,20:PRINT "Done":PRINT:END
320 PRINT TAB(20)"Include drive "DRIVE$" in search Y/N? ";
330 A$=INKEY$:IF A$="" THEN 330
340 IF A$="Y" OR A$="y" THEN INS%=INS%+ADD%:GOTO 370
350 IF A$="n" OR A$="N" THEN GOTO 370
360 BEEP:GOTO 330
370 PRINT A$:PRINT :RETURN
380 DATA 1711,1790,2094,1211,1123,1235,1254,1382,1556,2028
390 DATA 1589,1661,2042,2654,2099,1633,1993,1938,1430,1954
400 DATA 1826,1736,1970,1533,2312,1760,2439,2081,1989,2097
410 DATA 2258,1514,2478,1479,1695,1604,2101,1961
420 DATA B2,0A,E8,E3,01,BE,82,00,BF,B9,03,AC,3C,0D,74,03
430 DATA AA,EB,F8,30,C0,AA,A0,60,03,24,01,74,14,BF,65,03
440 DATA 30,C0,FC,B9,40,00,F2,AE,8B,DF,4B,BA,00,00,E8,52
```

```
450 DATA 00,A0,60,03,24,04,74,21,BF,65,03,C7,05,
    41,00,C7
460 DATA 45,01,3A,00,C7,45,02,5C,00,C7,45,03,00,
    00,8B,DF
470 DATA 83,C3,03,BA,00,00,E8,2A,00,A0,60,03,24,
    02,74,21
480 DATA BF,65,03,C7,05,42,00,C7,45,01,3A,00,C7,
    45,02,5C
490 DATA 00,C7,45,03,00,00,8B,DF,83,C3,03,BA,00,
    00,E8,02
500 DATA 00,CD,20,56,52,E8,65,01,E8,82,01,72,0D,
    E8,5F,00
510 DATA E8,98,01,72,05,E8,57,00,EB,F6,5A,52,E8,
    57,01,E8
520 DATA 6B,01,72,24,8B,F2,F6,44,15,10,75,0B,E8,
    7C,01,72
530 DATA 17,F6,44,15,10,74,F5,80,7C,1E,2E,74,EF,
    E8,0B,00
540 DATA 50,B4,1A,CD,21,58,EB,E4,5A,5E,C3,57,56,
    50,53,FC
550 DATA 8B,F2,83,C6,1E,8B,FB,AC,AA,08,C0,75,FA,
    8B,DF,FD
560 DATA AA,B0,5C,AA,E8,9C,FF,5B,C6,07,00,58,5E,
    5F,C3,50
570 DATA 52,56,53,51,83,C2,1A,8B,F2,8B,04,8B,54,
    02,BF,0A
580 DATA 00,E8,86,00,B2,20,E8,DF,00,E8,DC,00,83,
    EE,02,8B
590 DATA 04,25,E0,01,B1,05,D3,F8,E8,90,00,B2,2D,
    E8,C8,00
600 DATA 8B,04,25,1F,00,E8,A5,00,B2,2D,E8,BB,00,
    8B,04,25
610 DATA 00,FE,B1,09,D3,E8,05,50,00,E8,91,00,B2,
    20,E8,A7
620 DATA 00,E8,A4,00,83,EE,02,8B,04,25,00,F8,B1,
    0B,D3,E8
630 DATA E8,58,00,B2,3A,E8,90,00,8B,04,25,E0,07,
    B1,05,D3
640 DATA F8,E8,69,00,B2,20,E8,7F,00,E8,7C,00,59,
    5B,5E,BA
650 DATA 65,03,8A,07,C6,07,00,E8,CF,00,88,07,5A,
    52,83,C2
660 DATA 1E,E8,C5,00,E8,B3,00,5A,58,C3,31,C9,F7,
    F7,83,C2
670 DATA 30,52,31,D2,41,09,C0,75,F3,BA,20,00,52,
    41,83,F9
680 DATA 07,75,F9,B4,02,5A,CD,21,E2,FB,C3,31,D2,
    F7,F7,83
690 DATA C2,30,52,31,D2,F7,F7,09,D2,74,03,83,C2,
    10,83,C2
700 DATA 20,52,B9,02,00,B4,02,5A,CD,21,E2,FB,C3,
    31,D2,F7
710 DATA F7,83,C2,30,52,31,D2,F7,F7,83,C2,30,52,
    B9,02,00
720 DATA B4,02,5A,CD,21,E2,FB,C3,B4,02,CD,21,C3,
    56,BE,B9
730 DATA 03,E8,0B,00,5E,C3,56,BE,61,03,E8,02,00,
    5E,C3,50
740 DATA 57,8B,FB,FC,AC,AA,08,C0,75,FA,5F,58,C3,
    51,83,FA
750 DATA 00,77,03,BA,9B,03,83,C2,2B,B9,10,00,B4,
    1A,CD,21
760 DATA 52,BA,65,03,B4,4E,CD,21,5A,59,C3,51,52,
    BA,65,03
770 DATA B9,10,00,B4,4F,CD,21,5A,59,C3,50,52,B4,
    02,B2,0A
780 DATA CD,21,B2,0D,CD,21,5A,58,C3,50,52,56,FC,
    8B,F2,B4
790 DATA 02,AC,8A,D0,CD,21,AC,08,C0,75,F7,5E,5A,
    58,C3,00
800 DATA 05, 2A, 2E, 2A, 00, 43, 3A, 5C, 00, 00
```

巧算 1000!

东北电力学院　李　帆

本人采用一种特殊的技巧，求解 1000 以内的阶乘，不但程序量小，而且运算速度极快。现介绍原理如下：根据求 1000 阶乘的特点，是由一个不大于 1000 的数乘以一个位数极多的数，把这个大数的每一位都放到数组 a 中的每一个单元，而把不大于 1000 的这个数放入一个单元 i 中，用 i 中的数按顺序乘以数组中的每一个数，当数组中的数大于 9 时，留下个位，并记进位，在计算下一个数后，加上前一个数的进位，并留下个位，记进位，一直到计算完数组中的所有数。当 i 从 1～1000 变化，按以上规则计算，最后 a 中的数即为 1000!，当 i=1～n 变化时，就可以求出 n！。本程序用 C 语言编写，在 Micro VAX 机上计算 1000! 需用 50 秒，而在 286 微机上不到 30 秒。

```
#define max 5000
#include <stdio.h>
main (){
int i,j,x,c,n,a[max],o=0,d=1;
while (1==1) {o++;
for (i=0;i<max;i++) a[i]=0;a[0]=1;
printf("\n\n\n No. %d Input N=",o);
scanf("%d",&n);
printf("\n %d ! = \n",n);
if (n<0)break;
if (n==0)n=1;
for (i=1;i<=n;i++){c=0;
for (j=0;j<=d;j++){x=a[j]*i+c;c=x/10;a[j]=
    x%10;}
while (a[j]==0)j--;d=j+5;}
while (a[j]==0)j--;d=0;
for(;j>=0;j--){printf("%d",a[j]);if (d++%79==
0)putchar('\n');}
}
}
```

又发现一种新病毒

沈阳纺织工业非织造布技术开发中心　王家春

一、病毒的发现

开机后，计算机工作一切正常，输完程序后想打印，突然发现打印机不能联机。当初并没有认为有病毒，以为系统文件、打印模块或打印机等有问题，所以就逐个问题查找，过程是这样：

1. 退出汉字系统后仍不能联机，重新冷热启动计算机都不能联接打印机；排除了汉字系统有问题。

2. 更换主机检查，打印机和电缆都没问题。

3. 认为硬盘系统文件有问题，用软盘（贴写保护的系统盘）启动，打印机可以联机。传送系统文件到硬盘，但从硬盘启动仍不能联机。

从以上检查可以怀疑硬盘有病毒。但以前已知的病毒并没有不联打印机的。通过 PCTOOLS 发现，从硬盘启动时，系统内存被占用了 1KB。通过 DEBUG 发现，不联接打印机是由于 ROM BIOS 数据区的串、并行口地址被改变了。这时才断定，计算机染上了以前未发现的新病毒。由于此病毒更改系统串、并行口地址，这里暂称之为“IO”病毒。

“IO”病毒属操作系统型病毒，它藏在磁盘（磁盘指软盘和硬盘，以下同。）的第一物理扇区，即引导扇区。只要用带有“IO”病毒的磁盘启动计算机，“IO”病毒就常驻内存，而有传染“IO”病毒的能力，能感染任何没贴写保护的软盘。用带病毒的软盘启动，不论是否是系统盘，只要一次，就能感染硬盘。

“IO”病毒感染硬盘后，从硬盘启动 20H 次后，接在串、并行口上的外设都不能联机。

软盘感染后，某个文件可能有一个扇区内容被改变。如果系统盘感染“IO”病毒后又写某文件，可能因文件覆盖了引导扇区，此系统盘将不能启动系统。

“IO”病毒的特点是，硬盘感染后并没有什么症状。从带病毒硬盘启动系统 20H 次后，串、并行口外设才不联机。待到发现“IO”病毒时，已有很多软盘被感染。

“IO”病毒另一特点是，它把软盘的引导扇区写到 1 面 27H 道 8 扇区，覆盖了该扇区的数据，无可挽救的破坏了该区原数据。不象有些病毒仅破坏 FAT 表或文件目录，但文件体还存在盘上，还有挽救的机会。

对于 360KB 软盘，1 面 27H 道 8 扇区是在软盘的尾部，一般情况下问题不大。但对 1.2MB 的软盘，1 面 27H 道位于软盘前部，问题就经常出现了。

二、“IO”病毒的清除和免疫

1. 检查有否病毒

“IO”病毒藏在磁盘第一物理扇区，因为它修改引导扇区时，保留系统版本 OEM 标志和后部错误提示，初看是看不出病毒的，这也正是“IO”病毒的险恶之处。

读磁盘引导扇区到内存（硬盘为主引导扇区）。如头三个字节为 E9 8B 00，偏移 89H，8AH 两字节为 B1H，90H，即可断定磁盘已感染“IO”病毒。

2. 清除“IO”病毒

对于软盘很容易清除，可有多种方法。在清除“IO”病毒时一定要在系统无病毒情况下清除，方法是用一无病毒的系统盘启动系统。可用 DEBUG、PCTOOLS 等程序清除病毒，即把 1 面 27 道 8 扇区的内容写到 0 面 0 道 1 扇区。要注意的是：1 面 27H 道 8 扇区还是引导扇区否，如已被文件覆盖则不是原引导扇区，可用别的同类软盘引导扇区替代即可。

对于硬盘有点问题，但也容易消毒。因为“IO”病毒是直接改写硬盘主引导程序，没有保护原主引导扇区，所以要费点事。如果以前备份了硬盘主引导扇区，那么重新拷贝回去即可。如果未备份，可以把分区情况相同的其它硬盘的主引导程序备份出来，写到带病硬盘即可消毒。因为病毒程序仅修改引导程序的中间一部分，所以可以直接修改引导扇区，使之能引导活动分区就行了。最后也是最糟的方法是格式化硬盘，要低级格式化，高级格式化不能消除“IO”病毒。格式化前一定要备份硬盘上的所有文件。

3. 免疫

“IO”病毒是用引导扇区偏移 89H，8AH 两字节进行判别是否感染。如这两字节是 B1H、90H 则已感染，如不是则未感染。所以可以修改原引导扇区，把偏移 89H，8AH 修改为 B1H、90H 即能免疫。

扬基病毒快速解毒法

绵阳市职工社会保险事业管理处　李凤明

扬基病毒是一种文件型病毒，攻击所有.COM 和.EXE 程序。它采用了许多反跟踪措施，一旦发现跟踪，就将带毒程序还原。因此，可以利用它的这一特性来解毒，具体步骤如下（假设程序 A.EXE 感染了扬基病毒）：

一、运行有毒程序，使系统感染上扬基病毒；

C〉A

二、用 DEBUG 程序来加载染毒程序，再退出 DEBUG；

C〉DEBUG A.EXE

－Q

此时染毒程序已被病毒程序还原成无毒的了。然后，用同样方法对 DEBUG.COM 本身解毒，再重新启动。应特别注意检查 COMMAND.COM 程序，否则一启动又有毒了。

设备驱动程序 ANSI. SYS 的使用

青岛　山东纺织工学院　88 自动化二班　王立勇

MS-DOS 2. 0 及以后的版本，扩充了可安装设备驱动程序的功能，使计算机用户可以根据需要替换或编制自己的设备驱动程序，增强了灵活性及功能。

ANSI. SYS 是扩充键盘及屏幕显示的一个设备驱动程序，它载入内存后，替换了常驻内存的 CON 设备的驱动程序。ANSI. SYS 不仅支持原驱动程序所有的功能，而且还做了较大的扩充，提供了光标定位、移动光标、屏幕模式设置及键盘宏定义等功能。

下面就介绍一下 ANSI. SYS 的命令及使用方法。

一、命令格式：

①光标定位：

ESC[＃；＃　把光标移至指定＃行、＃列处，默认值为屏幕原点。

②光标上移：

ESC[＃A　光标上移＃行，默认为一行。若光标已在顶行则不动。

③光标下移：

ESC[＃B　光标下移＃行，默认为 1 行，如已在最后一行，则光标不动。

④清屏：

ESC[2J　清屏，光标回到原点。

⑤清行

ESC[k　从光标位置清至行末。

⑥设置屏幕模式：

ESC[＝＃h　其中＃的含义为：

0	40×25　黑白方式
1	40×25　彩色方式
2	80×25　黑白方式
3	80×25　彩色方式
4	320×200　黑白图形
5	320×200　彩色图形
6	640×200　黑白图形
7	行结束时换行。

⑦键的重定义：

ESC[＃；＃；……；p 或 ESC[＃；＃；'string'；＃p

其中＃为键值，string 为定义的字符串。

注：①＃为一数字参数，没指明时，使用默认值。

②ESC 为换码符，即 ASCII 的 27，而不是 ESC 三个字符。

③ANSI. SYS 的命令还有许多，详见 DOS 手册。

④使用 ANSI. SYS 的命令，必须在磁盘根目录中有 ANSI. SYS 文件并在设备配置文件 CONFIG. SYS 中加有命令 DEVICE＝ANSI. SYS。

二、使用方法：

ANSI. SYS 的命令字符串可以由系统功能调用 INT 21H 的 1、2、6、9 号子功能发出，因而我们可以在汇编语言或高级语言中使用 ANSI. SYS 命令。在高级语言中的应用可以替代调用汇编语言子程序，完成移动光标，清屏等高级功能。

值得一提的是，载入 ANSI. SYS 后，可以解决在 CCDOS 中使用 CLS 命令造成屏幕全黑的问题。

在汇编语言中使用 ANSI. SYS 命令十分简单，只要在数据段中加上以'$'结尾的命令字符串，用 INT21H 的功能 9 显示该字串即可，我们就不多说了。

下面我们举例说明在高级语言中的应用：

①屏幕模式的设置：

命令串为：ESC[＝＃h

i 在 FORTRAN 语言中可以用以下子程序实现：

```
SUBROUTINE SCREEN (mode)
INTEGER mode
WRITE (*,'(1X,A1,2H[=,I1,1Hh)') CHAR(27),
                mode
RETURN
END
```

ii 在 C 语言中表示为：

```
void screen (mode);
unsigned char mode;
{
  printf("\o33[=%dh",mode);
}
```

iii 在 PASCAL 中的应用：

```
procedure screen (mode:integer);
BEGIN
writeln(char(27),'[=',mode,'h');
END;
```

在主程序中只要正确设置 mode 的数值，再调用 screen 就能设置屏幕模式。

②清屏：

命令码为：ESC〔Z〕Z

i FORTRAN 语言的应用：

```
SUBROUTINE CLS
WRITE(*,'(1X,A1,3H[2J)') CHAR(27)
RETURN
END
```

ii C 语言中的应用：

```
void cls()
{  printf("\o33[2J");
}
```

iii 在 PASCAL 中的应用：

```
procedure cls;
BEGIN
writeln(char(27),'[2J');
END;
```

三、键盘重定义：

键的重定义是 ANSI. SYS 提供的较为重要的功能，也是较常用的功能。我们可以将一些键设置为编程语言的保留字，而另一些键则设置为相应的编译命令或 DOS 系统命令，从而加快了软件开发的速度。也可以将键定义为汉字的词组或句子，从而加快了汉字录入的速度，而且还不用调入词组。当然我们也可定义一些制表符，方便我们使用。

键重定义的 ESC 换码序列使用编程语言发出较为烦琐，使用不便，下面介绍两种较为方便的方法。

方法一：用 DOS 的内部命令 prompt。该命令是设置 DOS 系统提示符的，其格式为：promp[text]。其中 text 是将作为系统提示符的文本字符串，在此字串中可以加上以'$'打头的特殊元字符。

对于特殊元字符的定义可以参阅 DOS 手册，其中元字符 e 可以引导 ESC 的换码序列，用于定义键。

如我们设置 ALT—F10 为 BASICA↙（↙表示回车），可以用 prompt $e[0;113;'BASICA';13p。

使用这种方法会取消提示符，因而在定义完键后，要用 prompt ng 恢复原来的提示符，当然也可用元字符设计自己的提示符，方便使用。

《电子与电脑》91 年第 4 期，刊登何管略的文章中有一处错误，用 prompt 进行键的重定义，并不会被下一条 prompt 命令冲掉，除非下一条 prompt 又重新定义了这一键。因而我们可以在一个批文件中定义所有的键，而不必为每一条 prompt 写一个批文件，这样不但使用方便，而且还可以节约大量的盘空间（DOS 为每一文件至少分配一簇盘空间，对于不同的盘将可能是 0.5K 或 1K 或 2K 或 4K 空间）。

方法二：这是一种新的方法，即将 ESC 换码序列直接编辑成一个文本文件，需定义键时，用 type 命令显示该文本文件即可。

在编辑时，ESC 码的输入可以采用以下方法

①先键入 CTRL-V，再键入[，此时屏幕显示 ^ V [然后再输入以"["引导的字符串。

②按下 ALT 键，用数字小键盘键入 27，再放开 ALT 键，此时屏幕显示"←"。

③键入 CTRL+[此时屏幕显示"^ ["。

输入了 ESC 码后，再输入以"["引导的字符串即可。如我们定义 ALT—F10 为 BASICA↙可输入 ^ [[0;113;'BASICA';13p，当要恢复键的原定义时可键入键的扫描码，如 ^ [[0;113;0;113p 能使 ALT—F10 恢复 DOS 的原定义。

求　购

1802 程序设计手册、COMX PCI 机软硬件资料。

联系地址：西安市东效东方机械厂子第小学　　杨兴岐

邮政编码：710043

巧用虚拟盘

北京工业大学 901031 班　　李焕民

目前市场上的普及型 PC 比较多，但是许多用户只配了一台驱动器，这在进行文件拷贝的时候非常不方便，要多次抽、插盘。利用虚拟盘就能成解决这个问题，首先在主盘上建立 CONFIG. SYS 文件，使它包含下列命令：

DEVICE=VDISK. SYS 363 512 64

然后用主盘启动，机器会自动建好虚拟盘，并告诉用户分配给这个驱动器的字母是什么。注意，虚拟盘不能进行 FORMAT。本人使用的是 BF-PC Boy，内存为 512K，如果您的机子内存较小，可把盘容量 363K 改为适当的数。

递归解九连环

辽宁大学　计 3#　　钟　武

九连环是一种古老的智力玩具，相传是公元前 234 年前后我国的一位武士发明的，两千多年来在民间广为流传。在十六世纪中叶传入欧洲，并被意大利数学家载入典籍，誉之为"中国套环难题"。

九连环构思精巧、结构奇特，它是由九个关连着的环、一根套柄和一些附件构成。玩法分套上九连环和解开九连环两个过程。九连环从完全脱离的初始状态一直"走"到套柄将九个环全部套上的状态的过程，称为套上九连环；其逆过程称为解开九连环。

完成解开九连环的基本思想是：首先，将九连环的前七个环解开，接着将第九个环解开，再将前七个环套上，最后将八个环解开。这就完成这一过程。

套上九连环的基本思想是：首先，将前八个环套上套柄，再解开前七个环，接着将第九个环套上，最后将前七个环套上。

按此类推，九连环问题归结为一个环和二个环的套上和解开问题。显然，这两种情况是可直接解决的。

从以上分析可知：九连环问题是一个递归问题。递归有很强的能力，也给程序设计带来很大方便。递归表述程序的必要且充分的工具是过程或子程序，因为它允许语句被赋以一个名字并按其名字进行调用。设过程 unload(9)表示解开九连环，过程 load(9)表示套上九连环。

递归有直接递归和间接递归两种。若过程明显包含着对自己的引用，称为直接递归。若过程 A 包含着对另一过程 B 的调用，而 B 又直接或间接地调用 A，则称 A 是间接递归的。从分析可知，九连环问题的解决

既含有直接递归又含有间接递归。

下面描述 n 连环问题的一般算法：(n>2)首先描述过程 unload(n)：

(1)unload(n－2)

(2)解下第 n 个环

(3)load(n－2)

(4)unload(n－1)；

再有过程 load(n)：

(1)load(n－1)

(2)unload(n－2)

(3)套上第 n 个环

(4)load(n－2)。

下面，分别用 C 语言和 PASCAL 语言编写解开九连环的程序。若编写套上九连环的程序，只要在主程序中稍加改动，直接调用套上九连环过程即可。

C 语言源程序：

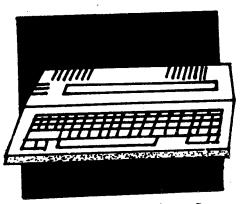
```
#include <stdio.h>
#define NCIRCLE 9
int step=0;
void unload__all(int)
void load__all(int)
main()
{
  unload__all(NCIRCLE);
}
void unload__all(int n)
{
  if (n>2) {
    unload__all(n-2);
    printf("%16d\t%d---> down\n",++step,n);
    load__all(n-2);
    unload__all(n-1);
        }
  else
    printf("%16d\t%s---> down\n",++step,(n==
2)? "1 & 2":"1");
}
void load__all(int n)
{
    if (n>2){
      load__all(n-1);
      unload__all(n-2);
      printf("%16d\t%d---> up\n",++step,n);
    load__all(n-2);
  }
  else
    printf("%16d\t%d---> up\n",++step,(n==1)?
"1":"1 & 2");
```

PASCAL 语言源程序：

```
PROGRAM ninecircle (output);
CONST
        ncircle = 9;
PROCEDURE load (n:integer);
        forward;
PROCEDURE unload (n:integer);
  BEGIN
    CASE n OF
      1: writeln('1 ---> down':20);
      2: writeln('1 & 2 ---> down':24);
    OTHERWISE
      BEGIN
        unload(n-2);
        writeln(n:10,' ---> down');
        load(n-2);
        unload(n-1)
      END
  END
END;
PROCEDURE load;
  BEGIN
    CASE n OF
      1: writeln('1 ---> up':18);
      2: writeln('1 & 2 ---> up':22);
    OTHERWISE
      BEGIN
        load(n-1);
        unload(n-2);
        writeln(n:10,' ---> up');
        load(n-2)
      END
  END
END;
BEGIN
  unload(ncircle)
END.
```

以上两个程序都在 VAX－8350 机上运行成功。当环数取 5 时，即 ncircle=5 时，运行结果为：

```
1     1 ---> down
2     3 ---> down
3     1 ---> up
4     1 & 2 ---> down
5     5 ---> down
6     1 & 2 ---> up
7     1 ---> down
8     3 ---> up
9     1 ---> up
10    1 & 2 ---> down
11    4 ---> down
12    1 & 2 ---> up
13    1 ---> down
14    3 ---> down
15    1 ---> up
16    1 & 2 ---> down
```



学习机之友

将CEC-I中华学习机改制成专用机

陕西电子技术研究所　姜良夫

CEC-I中华学习机已作为家庭教育、娱乐、事务用机进入千家万户，它还以简单易学，操作方便而适于改作各种专用机。对于改制专用机，人们希望尽量发掘机内潜力而不重新投资，一来可以降低成本，二来避免增加技术上的复杂程度。这就首先要求机内有提供存放专用程序的内存空间。本文介绍如何充分利用CEC-I机内存空间，置放自己的的专用程序，将其改制成使用方便的专用机。

一、CEC-I内存分配

CEC-I机的存储器系统要比APPLE Ⅱ＋机复杂、容量也大许多。CEC—I机内存分主存和辅存，主存储器有16KROM和64KRAM。利用主存储器和辅存储器之间的切换软开关

RAMRD；RAMWRT；ALTZP

还可访问辅存空间，其地址为$0000～$BFFF以及$D000～$FFFF。CEC—I机中辅存空间分为四块，三块为ROM区，一块为RAM区。

辅存中，$0000～$3FFF是RAM区，并映射到主RAM所对应的存储器上，可以说是主存和辅存的公共区。其余空间皆为ROM区，存放汉字系统软件和LOGO语言，它们分别存储于U_7和U_{35}两片27256芯片中。汉字库则存于两片1M位的U_{33}、U_{34}芯片中。

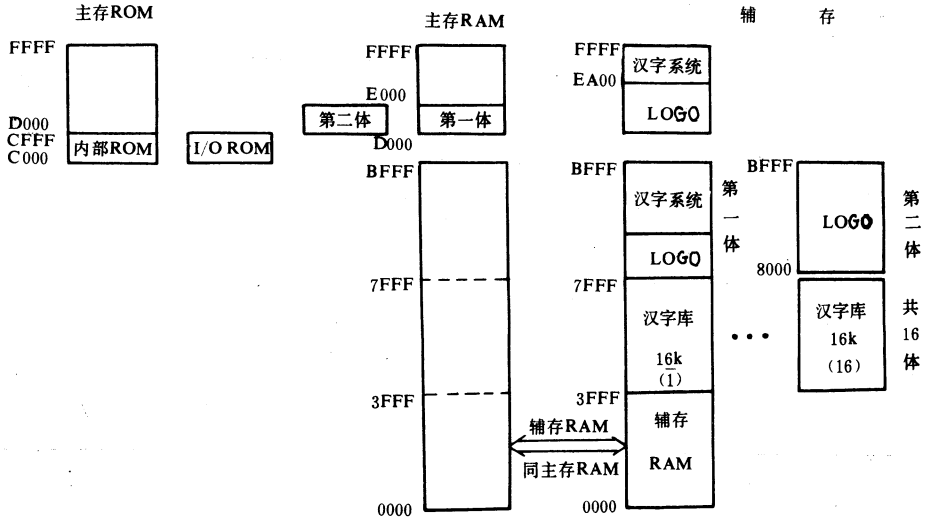
芯片U_7中，物理地址$5000～$7FFF的空间，固化着占系统逻辑地址$D000～$FFFF的监控程序、BASIC解释程序等，这是主ROM区。物理地址$4100～4FFF的空间是内部ROM区。物理地址$0000～$3FFF的是逻辑地址$8000～$BFFF第二体的辅存ROM，固化有LOGO语言。芯片U_{35}中，在32K的存储空间内，固化有汉字系统及LOGO语言等。占物理地址$0000～$3FFF是辅存$8000～$FFFF的第一体，固化有汉字拼音码表和LOGO语言，$5000～$7FFF的地址空间是辅存$D000～$FFFF-ROM区，固化有汉字管理程序和LOGO语言程序。$C300，$C600的I/O ROM也存放在U_{35}芯片内。而辅存$4000～$7FFF的空间通过体切换，扩充达16体，总共计16K×16＝256K，由两个128K×8的芯片组成，内装有国标一、二级汉字点阵字库。

二、LOGO语言程序的存放及调用

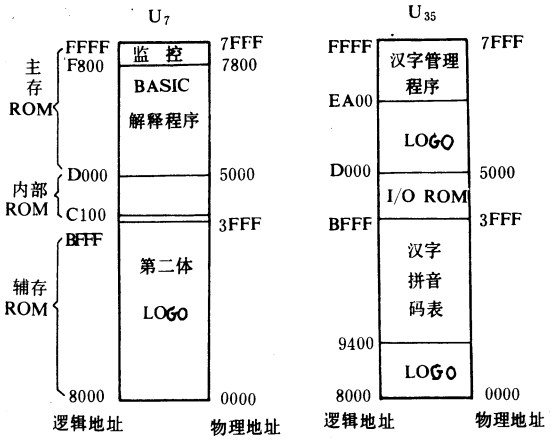
通过对内存配置的分析，不难想到LOGO占用的空间可以存放我们的专用程序。从图2可知，LOGO所占空间约16K＋5K＋6.5K＝27.5K，这对于存放一般的应用程序是可以满足的。

那么LOGO是怎样启动和工作的？在键盘上敲击LG命令，程序就转LG的入口地址$D10C，程序如下：

```
D10C-A9 10      LDA # $10
D10E-8D B0 C0   STA #C0B0 ;选第一体
D111-8D 07 C0   STA $C007 ;切换到内部ROM
D114-20 00 C6   JSR $C600 ;跳到转存子程序
D117-8D 06 C0   STA $C006 ;切换到I/O ROM
D11A-4C D1 D2   JMP $D2D1
```



图一　CEC-I系统地址分配图

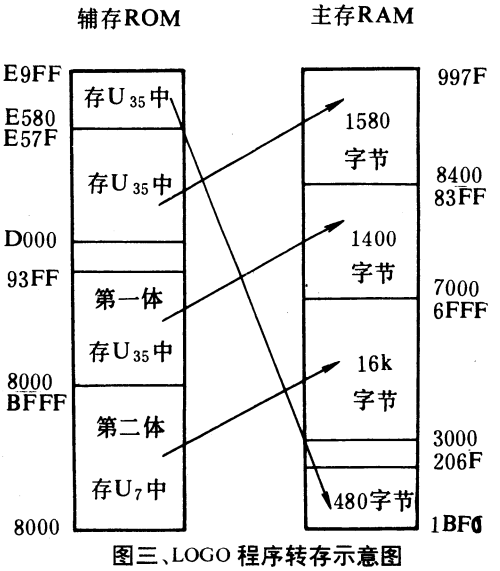


图二　U_7、U_{35}内存程序配置图

这段程序主要作用是跳转到内部 ROM 区从 $C600 开始的子程序。读者可以在监控状态下，对软开关 $C007 写操作，即可切换到内部 ROM，如：

```
*C007:1↙
*C600L↙
C600-8D 03 C0     STA  $C003
C603-8D 05 C0     STA  $C005
⋮
```

这个子程序分四部份，第一部份把辅存 ROM $8000～$BFFF 第二体中的程序转存到主存 RAM $3000～$6FFF，第二部份将辅存 ROM $8000～$93FF 第一体中的程序转存到主存 RAM $7000～$83FF，第三部份将辅存 ROM $D000～$E57F 中的程序转存到主存 RAM $8400～$997F，第四部份将辅存 ROM $E580～$E9FF 中的程序转存到主存 RAM $1BF0～$206F。转换对应关系如下图所示：



图三、LOGO 程序转存示意图

执行完转存子程序后，跳转到 $D2D1 入口地址。对转存到 RAM 中的程序作了校核，无误后，即转到 LOGO 语言的入口地址 $1BF6。这样 LOGO 程序从辅存调出，转存入主存 RAM，并进入正常运行。

三、BASIC 程序在内存中如何存放。

我们用 LIST 命令可以看到内存中的 BASIC 程序，可进行编辑、修改。但是它们在内存中是如何存放的呢？原来，BASIC 程序在零页使用了一些标志单元，指示这个 BASIC 程序的起始位置、结束位置、变量、数组、字符串存储起始地址等，如：

$67～$68：指向 BASIC 程序的起始地址

$69～$6A：指向程序中简单变量的起点。BASIC 解释程序冷启动时，指向 $803；执行 NEW 后，指向 $804，以后随着程序的输入而自动调整，总是指向程序末地址加 1。

$6B～$6C：数组变量起始指针，未执行程序前和 $69～$6A 一样，执行后会根据 $69～$6A 的指针自动重新设定新值，故修改 $69～$6A 会影响此指针。

$6D～$6E：指向数组变量的结束地址加 1。

$6F～$70：指向字符串使用区的起始点，无 DOS 时为 $C000，在中文状态时为 $9200。

$73～$74：BASIC 用户可使用的最高地址加 1。无 DOS 时最高指向 $C000。

$AF～$B0：BASIC 程序的终点，并随程序的修改而自动调整。

BASIC 程序一般从 $801 开始存放。每一行语句的结束用 00 来表示，程序最后以三个 00 作结束。

程序在内存中存放地址可由 $67～$B0 中有关参数确定，这就为我们从内存中取出 BASIC 程序或者把 BASIC 程序放入内存提供了依据。

四、应用程序的存放

搞清了 LOGO 语言及 BASIC 程序在内存中的存放情况，就可以着手应用程序的固化工作。

我们用中文 BASIC 编制了一个专用的计算程序，它在内存中的起始地址在 $801，末地址在 $21DC，程序全长 $19DC，约 6K 左右，因而可将其固化在芯片 U_7 中，放置在物理地址 $0000～$3FFF 的 16K 空间内，这是辅存 ROM $8000～$BFFF 的第二体空间。程序实际只占用逻辑地址 $8000～$99DC 的这部份空间，由于程序容量小，就不需改动 U_{35}芯片。另外还需修改、编写从 $C600 起的转存程序；从 $D2D1 起的予置、校核程序。对于 $C600 起的转存程序改写并不困难，可以参照原 LOGO 语言使用的 $C600 子程序的编写方法。需要注意的是：从主存切换到辅存和从辅存切换到主存，以及辅存中选择第一体与第二体时，软开关使用不当会产生意料不到的情况，当然也就不能正确地执行程序的转存。关于软开关的详细定义，参见“中华学习机 CEC-I 技术手册(硬件)”144～146 页有关内容。

$D2D1 起始的程序，在原 LOGO 中主要是校验从辅存转存到主存 RAM 中的程序正确与否，正确，则执

行LOGO语言，不正确则返回到BASIC状态。对于我们的专用程序，则应对零页中有关的标志单元予置参数。即＄67、＄68放入BASIC程序首址＄801，在＄69、＄6B、＄6D、＄AF中放入程序末地址低字节，在＄6A、＄6C、＄6E、＄B0中放入程序末地址高字节。然后可校验RAM中任意地址的内容是否为应存入的内容，若是，转到＄D566，若不是，再转到＄D10C，重新装入BASIC程序。

我们为上述专用程序编制的转存程序是：

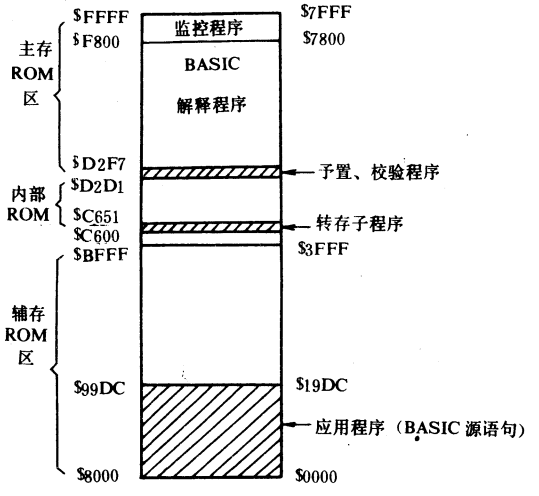
```
4600-  8D 03 C0   STA   $C003
4603-  8D 05 C0   STA   $C005
4606-  A9 00      LDA   # $00
4608-  A8         TAY
4609-  85 FC      STA   $FC
460B-  85 FE      STA   $FE
460D-  A9 80      LDA   # $80
460F-  85 FD      STA   $FD
4611-  A9 08      LDA   # $08
4613-  85 FF      STA   $FF
4615-  8D 03 C0   STA   $C003
4618-  8D 05 C0   STA   $C005
461B   A9 20      LDA   # $20
461D-  8D B0 C0   STA   $C0B0
4620-  B1 FC      LDA   ($FC),Y
4622-  8D 02 C0   STA   $C002
4625-  8D 04 C0   STA   $C004
4628-  91 FE      STA   ($FE),Y
462A-  C8         INY
462B-  D0 E8      BNE   $4615
462D-  E6 FD      INC   $FD
462F-  E6 FF      INC   $FF
4631-  A9 99      LDA   # $99
4633-  C5 FD      CMP   $FD
4635-  D0 DE      BNE   $4615
4637-  8D 03 C0   STA   $C003
463A-  8D 05 C0   STA   $C005
463D-  A9 20      LDA   # $20
463F-  8D B0 C0   STA   $C0B0
4642-  B1 FC      LDA   ($FC),Y
4644-  8D 02 C0   STA   $C002
4647-  8D 04 C0   STA   $C004
464A-  91 FE      STA   ($FE),Y
464C-  C8         INY
464D-  C0 D0      CPY   # $D0
464F-  D0 E6      BNE   $4637
4651-  60         RTS
```

校验、予置程序是：

```
52D1-  A9 01      LDA   # $01
52D3-  85 67      STA   $67
52D5-  A9 08      LDA   # $08
52D7-  85 68      STA   $68
52D9-  A9 DC      LDA   # $DC
52DB-  85 69      STA   $69
52DD-  85 6B      STA   $6B
52DF-  85 6D      STA   $6D
52E1-  85 AF      STA   $AF
52E3-  A9 21      LDA   # $21
52E5-  85 6A      STA   $6A
52E7-  85 6C      STA   $6C
52E9-  85 6E      STA   $6E
52EB-  85 B0      STA   $B0
52ED-  AD 02 08   LDA   $0802
52F0-  C9 07      CMP   # $08
52F2-  D0 03      BNE   $52F7
52F4-  4C 66 D5   JMP   $D566
52F7-  4C 3C D4   JMP   $D43C
```

注：以上地址是程序在U_7中的存放地址。

BASIC应用程序、转存程序、予置、校验程序在U_7芯片中的存放位置如图所示：



图四　新U_7中程序分配示意图

五、程序的固化及编程技巧：

下一步便是将这三部份程序写入U_7(27256)芯片内。可采用的办法很多，主要取决于读者手中现有的编程工具。我们采用的办法是：首先利用苹果机写入卡（只能写2764以下的芯片），将三部份程序写入2764中，然后分别将U_7中的程序和2764中程序读入PC机内存，经编辑后，使其存放在各自规定的地址上，形成新的U_7内容，用PC机编程卡写入27256芯片中即可。

如果是用机器语言编写的应用程序，存放和调出的基本方法同上，但不用对零页标志单元进行置数，校核后，直接转到程序入口地址即可。

怎样启动和运行应用程序呢？如果想保留CEC-I机的通用计算机功能，那么只要按下“LG”键，即可启动，运行专用程序。要退出或中途想停止执行，只要按下CTRL-Reset，即返回BASIC状态。但若想使其真正成为一台专用计算机，尽量减少不必要的操作，且一开机就自动进入专用程序，那就需对U_7中的程序内容作以下修改。

＄F1CB　3C→0C

＄F1CD　D4→D1

同时将 $D2D1 起始的予置程序中最后一句 $D2FF-4C 3C D4 改成：$D2FF-4C 0C D1，以上应在写 27256 前就修改好，一并写入 U_7 中。这样，开机后，就能自动进入专用程序的执行。

CEC-I 机冷启动或进入汉字状态后，会在屏幕上出现提示信息，启动中文专用程序后，由于使用了高分辩第二页，屏幕上出现杂乱无章的横道、竖道，会使人感到很不舒服，若欲去掉，可在 $D23C 程序中加一段清 $4000～$5FFF 的程序，改为：

$D23C-20 58 FC A9 40 85 FD A9 00

$D245-A8 85 FC 91 FC C8 D0 FB E6

$D24E-FD A2 60 F4 FD D0 F3 60

同时在 $D300 子程序中，将 $47B 置成 3，或在专用程序中，程序头上加一条 POKE 1147,3 语句，这样开机后，就会“干净”地进入专用程序。

“LG”是系统的保留字，存放在内存 $C498 和 $C499 中。若想改成自己所希望的保留字，如“JS”，则只需将 $4A，$D3 放入 $C498，$C499 中。这以后，按下“JS”键，就可以启动专用程序了。

用栈结构优化程序

浙江绍兴一中　　陈合力

《计算机教与学》90 年第 6 期上刊登《用微机模拟 +－*/^ ()E 运算》是采用双向链表结构。笔者用栈结构优化了程序。改进后的程序不但比原来的短，而且运算速度要快得多。

程序说明：

5 句，输入待求的算术表达式，如 1+2*(3－4/(7－8*3/2)－5^ 7)/8－6。为保证完成全部运算，在算术表达式后面加上一个＋号。定义二个数组，数组 A 存放运算符号，数组 B 存放算式表达式中的数码。

10 句，为保证运算不误，将 A(0)赋值－1。变量 M 记括号的重数。算术表达式中的数码暂存于 T$ 中。

20 句，把运算符用一个特定的数值表示。约定＋－*/^ 分别用 01234 表示，在一重括号内相应的数值增加 6，二重括号内增加 12，依次类推。运算次序是根据运算符对应的数值大小，大的优先运算。同一级运算如(＋－)应从左到后，故采用除以 2 取整再比较的办法，并把数码存于 T 中。

30 句，确定运算顺序，出栈。

40 句，进栈。

60～100 句，具体执行运算并返回。

```
5 INPUT A$:PRINT A$"=";:A$=A$+"+":DIM A(100),B(100
10 A(0)=-1:FOR I=1 TO LEN(A$):K$=MID$(A$,I,1):M=M+(K$="(")-(K$=")"):IF K$="."OR K$="E"OR K$)="0" AND K$(="9" THEN T$=T$+K$
20 Z=(K$="+")+2*(K$="-")+3*(K$="*")+4*(K$="/")+5*(K$="^"):ON Z=0 GOTO 50:Z=Z+M*6-1:T=VAL(T$):T$=""
30 IF INT(Z/2)(=INT(A(P)/2) THEN ON A(P)-INT(A(P)/6)*6+1 GOSUB 60,70,80,90,100:P=P-1:GOTO 30
40 P=P+1:A(P)=Z:B(P)=T
50 NEXT:PRINT T:END
60 T=B(P)+T:RETURN
70 T=B(P)-T:RETURN
80 T=B(P)*T:RETURN
90 T=B(P)/T:RETURN
100 T=B(P)^T:RETURN
```

寻找无理数 e 的约率和密率

上海崇明县城中学　　施为钢

大家知道无理数 π 的约率为 $\frac{22}{7}$，密率为 $\frac{355}{113}$，那么是否也能寻找到无理数 e 的约率和密率呢？

我们知道，数列 $\{(1+\frac{1}{n})^n\}$ 的极限为 e 但是，这个数列的分子分母递增太快而且向 e 逼近的速度很慢，如它的第七项为 $\frac{2097152}{823543}=2.546\cdots\cdots$，由此看出，我们不能借助于这个数列。

要想找到分子分母不很大而且又非常接近于 e 的分数，我们不妨作如下的思考：

设 ε 是一个很小的正数，如果存在分数 $\frac{p}{q}$ 非常接近于 e(其中 p,q 是自然数)，那么应该满足不等式

$$|e-\frac{p}{q}|<\varepsilon \text{ 即 } |eq-p|<q\varepsilon\cdots\cdots(*)$$

由于 qε 很小，所以满足(*)式的自然数 p 实际上是最接近于 eq 的自然数，这可以分两种情况讨论：

当 eq＞p 时 p=INT(eq)代入(*)式得

$$eq-INT(eq)<q\varepsilon\cdots\cdots(1)$$

当 eq＜p 时 p=INT(eq)＋1 代入(*)式得

$$INT(eq)+1-eq<q\varepsilon\cdots\cdots(2)$$

注：以上 INT(eq)表示不超过 eq 的最大整数。

经过以上分析，我们把寻找接近 e 的分数问题转化成寻找满足(1)式或(2)式的自然数 q 的问题，可以用 BASIC 语言编写一段程序，让计算机去寻找答案。

程序说明：10 语句行用来键入分母 q 的上界 M；以及 EP(即 ε)的值；210 语句行的作用是，每当找到一个接近 e 的分数后，马上修改 EP 的值，从而保证了以后找到的分数更近于 e。

由程序运行的结果看出，接近 e 的分数很多，我们不妨把其中的 $\frac{19}{7}$ 作为约率，把 $\frac{49171}{18089}$ 作为密率。

如果对程序的 15 语句稍作修改，就可以用来找出

与其它无理数相接近的分数。

```
10 INPUT M,EP
15 E= EXP (1)
20 FOR Q=1 TO M
30 X =E*Q:D= EP*Q:P=INT(X)
40 IF X-P(D THEN GOSUB 200:GOTO 60
50 IF P+1-X(D THEN P=P+1:GOSUB 200
60 NEXT Q
77 END
200 PRINT P;"/";Q,"=";P/Q
210 EP=ABS (E-P/Q):RETURN
]RUN
? 20000,0.005
19/7                =2.71428572
49/18               =2.72222222
68/25               =2.72
87/32               =2.71875
106/39              =2.71794872
173/71              =2.71830986
685/252             =2.71825397
878/323             =2.71826625
1071/394            =2.71827411
1264/465            =2.71827757
1457/536            =2.71828358
2721/1001           =2.71828172
12341/4540          =2.71828194
15062/5541          =2.7182817
17783/6542          =2.71828187
20504/7543          =2.71828185
23225/8544          =2.71828184
25946/7545          =2.71828182
49171/18087         =2.71828183
```

怎样节省 CEC-I 存储空间

杭州自动化研究所　尹向阳　梁　曦

采用 CEC-I BASIC 语言开发数据量大的管理软件常常感到内存和软盘容量不足，影响了这些软件的实用性。笔者在实际工作中，针对具体情况，采用了多种节省存储空间的有效方法。本文介绍其中三种效果明显并具一定普遍性的方法。

一、用整型数组代替实型数组

CEC-I BASIC 的实型数组的每个元素占五个字节，数值表示范围达 $\pm 9.999\times 10^{37}$，整型数组的每个元素占二个字节，数值表示范围为 ± 32767。如果把本来要用实型数组存储的数据，经一定处理能用整型数组存储，则可节省五分之三的存储空间。例如工资额、学生成绩这类数据，一般情况下要用实型数表示，但经某种变换（如乘 100）就变成整数，而且数值大小在整型数的表示范围内，这类数据就可用整型数组存储。参加运算前，从整型数组中读出，再经逆变换（如除 100）即可恢复原值。

有些数据（如工资总额）经上述处理后，会超出整型数的表示范围，但仍是小于 65536 的正整数，这时可以把整型数的表示范围调整到[0,65536]，即把大于 32767 的整数用负整数表示，方法如下：

设 N 是整数，且 $0\leqslant N\leqslant 65536$

把 N 存入整型数组前，执行语句

10 N=N-65536*(N>32767)

从整型数组中读出，执行语句

10 N=N+65536*(N<0)

二、以 B 类文件代替顺序或随机文件

CEC-BASIC 的顺序文件和随机文件都是以 ASCII 码形式存储在软盘中的，占用空间大，存取速度慢。在处理诸如职工档案、学生成绩等管理对象比较固定的管理软件中，可以用 B 类文件代替顺序文件或随机文件。我们知道，B 类文件是以文件名、内存地址和文件长度三要素进行存取的，而对数组的操作完全不涉及这些信息。怎样建立 B 类文件和数组之间的联系？现在假定有这样一个程序：

```
10 DIM A(200),B%(50,12),C(40)
20 A=PEEK(107)+256*PEEK(108)
30 B=A+(200+1)*5+7
40 C=B+(50+1)*(12+1)*2+9
```

A 数组定义为实型数组，共有 201 个元素，占用 1005 个字节；B 数组定义为二维整型数组，共有 51*13 个元素，占有 1326 个字节。

每个数组都有一个表头，用以描述该数组的特征，表头的长度由数组的维数决定。这里 A 数组的表头占 7 个字节，B 数组的表头占 9 个字节。因此，A 数组数共占 1012 个字节，B 数组共占 1335 个字节。

零页的 $6B，$6C 二个单元是指向数组区域开始地址的指针，语句 20 给 A 赋的值就是数组区域的首地址，也就是 A 数组在内存中分配的首地址。B、C 分别是 B 数组和 C 数组在内存中分配的首地址。

知道了数组在内存中的首地址和字节数，再命名一个文件，就可以用 BLOAD、BSAVE 命令存取数组。

三、压缩汉字引荐符

CEC-BASIC 语言处理的汉字是用三字节引荐符方式表示的，其汉字内码格式为 7F+区码+位码。这里的区位码和国标区位码略有区别。

对于人名、项目名称等用纯汉字表达的字符串，用 INPUT 语句接受汉字字符串后，用 ASC,MID $ 函数把区码和位码转换为整型数存入数组。在显示或打印前自动拼接上引荐符 7F，这样既节省了存贮空间，也不影响显示和打印速度。

转　让

中华学习机，日本原装驱动器，打印机，打印卡等。学习机已用半年，其余皆暂新。原价 2376 元，愿以 1800 元转让，并送所有资料与软件。也可单项转让，欲者从速。

地址：北京西单东斜街 53 号杨建鸿　邮编：100032　电话：602.2238

ProDOS 磁盘操作系统入门(续)

廖　凯

5. CHAIN

格式：CHAIN 路径名[,@行号][,S#][,D#]

CHAIN/卷名/子目录/BASIC 文件名

功能：此命令可使程序沿用上一个程序所产生的变量值。在一个 BASIC 程序太大而内存不够时，我们将程序分成一些易处理的程序段，用 CHAIN 命令运行每个程序段。当 CHAIN 时，第一个程序段所有变量包括名称及其值被保存在内存内，而后清除程序，接着执行第二个程序，使用前面程序所留下的变量值。你可以再次使用 CHAIN 命令，返回到第一个程序段。注意不能重复定义 CHAIN 程序的数组。你可以使用"@行号"参数来 CHAIN 一个程序的一部分。

@行号：使 ProDOS 在指定的行号执行。例如，设为",@95"，则程序会在行号 95 开始，如果没有此参数，那么 ProDOS 将从最小的行号开始。

在立即方式下，CHAIN 命令用在程序除错时才有实际意义。通常，CHAIN 是用于一个程序内的。

6. CLOSE

格式：CLOSE [路径名]

CLOSE/卷名/子目录/文本文件名

功能：此命令用于程序结束 READ 或 WRITE 操作之后，关闭打开的文件。如果使用此命令而没给出路径名时，所有打开着的文件都将被关闭。在文件被关闭之前，1K 字节的字符缓冲区将被清除。自 ProDOS 在 512 字节的区域内读写以后，CLOSE 命令将清除少于 512 字节的数据。注意在文本文件使用之后，一定要关闭文件，否则数据会丢失。

7. FRE

格式：FRE

功能：此命令是用于快速收集无用存储单元。此命令只能在程序调用方式下使用。在立即执行方式下，FRE 是一个 APPLESOFT 命令，并必须放在 PRINT 语句的后面，结果是一个在屏幕上所显示的数字。具有 64K 的计算机这数字是－29188，这数字表示供给用户可使用空闲 RAM 的字节数，－29188 相应的正数是 36348(加 65536 所得的结果)，约为 35.5K 字节。

8. IN#和 PR#

格式：IN#槽口号[,A#]

PR#槽口号[,A#]

功能：这两个命令除了用来处理输入和输出的接口控制外，还可与机器语言沟通，重新规划 I/O 口。

如果你正使用一个 80 列卡，在改变输出到另一个外围槽口之前，一定要关闭 80 列卡，其命令为：ESC CTRL __ Q

下面举例说明：

如果在内存地址 768($300)有一子程序，你想发送输出到该子程序，使用的命令为：

PR#A768 或 PR#A$300

输出子程序可以被指定给一个槽口，访问该槽口将被改变方向到该子程序。

PR#4,A768　　恢复为 PR#4,A$C400

内存分配给每个扩展槽的起始地址是 $C#00，#是槽口号，你也可以将所有送到一个槽口的输出均被转到另一个槽口之内，要转换槽口，就使用：

PR#槽口号,A$C#00 (#是槽口号)

子程序结束工作时，通过用 PR#0 来恢复控制权到控制台。用 INT#命令来使用相同情况的设备以接收输入信息。

9. OPEN

格式：OPEN 路径名[,S#][,D#]

OPEN/卷名/子目录/文本文件名

功能：此命令用于在 READ 或 WRITE 一个文件之前，打开一个文本文件。此命令只能用在程序调用方式下。ProDOS 允许有户同时打开八个文件。当每个文件被打开时，ProDOS 自动分配 1K 字节的缓冲区空间(1024 字节)。在首次 OPEN 一个文件时，ProDOS 在指定的目录内产生一个类型为 TXT 的文件。

10. POSITION

格式：POSITION 路径号,F#/,R#

POSITION/卷名/子目录/文本文件名,R#

功能：允许用户存取一个文件内的任何字段(数据域)或字节。在使用此命令之前，文件必须是打开的。此命令只能用在程序调用方式下，在文件内通过读取 RETURN 代码，将从当前的位置向前移动指针，这指针被定位在第#RETURN 后的第一个字节上。

F#/R#：是先读取而后放弃并被#所指定的数个字段(数据域)。为了与 DOS3.3 兼容，F#(字段)与 R#(记录)是相等的。

11. READ

格式：READ 路径名[,F#][,B#]

READ/卷名/子目录/文本文件名

功能：通知 ProDOS 读取一个文本文件的数据。此命令只能用在程序调用方式下。READ 命令允许用户使用 B#或 F#参数，以向前定位若干字节或字段(数据域)。要被读取的起始位置，可以由用户来指定。在 READ 文件前，必须先将此文件打开。在使用 READ 以后，将一直保持到给出另一个 ProDOS 命令为止。

F#：是将要被读取而后放弃的字段(数据域)的

数量。回车代码(CR)是用于指示被读取的字段(数据域)的号码。

B#是将要被读取而后弃的字节或字符。

12. RUN

格式：RUN 路径名[,L 行号][,S#][,D#]

RUN/卷名/子目录/BASIC 文件名

功能：此命令用于装入并运行一个 BAS 类型的 BASIC 文件。用户可以经指定行号有选择地运行部分程序。

L#：是用于指定要被执行的程序内的行号。

13. WRITE

格式：WRITE 路径名[,F#][,B#]

WRITE/卷名/子目录/文本文件名

功能：是通知 ProDOS 将数据写入磁盘(使用一个后续的 PRINT 语句)。在 WRITE 文件之前，必须先将文件打开。使用此命令后，将一直保持到给出另一个 ProDOS 命令为止，此命令只能用在程序调用方式下。

B#和 F#参数能用于向前定位若干字节或字段(数据域)于执行 WRITE 操作之前。B#和 F#参数的含义见 READ 命令的介绍。

三、ProDOS 取消的命令

1. FP 和 INT

ProDOS 不再支持 INTEGER BASIC，所以没有必要再在两个 BASIC 版本之间作转换了。

2. INIT

ProDOS 已将磁盘格式化子程序放置在 FILER 文件内。如果将它作为一个正规命令放入 ProDOS 的话，那么这格式化程序会占用很大的空间。

3. MAXFILES

ProDOS 会自动地分配缓冲区空间给八个打开的文件，没有必要再用 MAXFILES 分配缓冲区的空间。

4. MON 和 NOMON

为了使 ProDOS 尽可能地少占内存，所以将 MON 和 NOMON 从解释程序中删去了。

四、ProDOS 的其它命令

1. DELETE

格式：DELETE 路径名[,S#][,D#]

DELETE/卷名/子目录/文件名

功能：从磁盘上删除一个文件。此命令可以删除任何类型的文件。在删除一个目录文件时，此目录内必须不包含任何文件。

2. EXEC

格式：EXEC 路径名[,L#][,S#][,D#]

EXEC/卷名/子目录/文本文件名

功能：此命令用于从磁盘上执行一个顺序文本文件。

L#：可用此参数指定 EXEC 执行整个文件或文件的一部分。例如：L105，是从顺序文件中第 105 字段开始执行。

3. LOAD

格式：LOAD 路径名[,S#][,D#]

LOAD/卷名/子目录/BASIC 文件名

功能：此命令是将磁盘上的一个 BASIC 文件装入内存。

4. LOCK 和 UNLOCK

格式：LOCK 路径名[,S#][,D#]

LOCK/卷名/子目录/文件名

UNLOCK 路径名[,S#][,D#]

UNLOCK/卷名/子目录/文件名

功能：LOCK 命令用于给一个文件上锁，使其内容不能被改变。对于锁着的目录文件来说，只能保护空目录不被改变，而其包含的文件可以被改变。上锁标记是一个“*”(星号)，它放在文件名的左边。UNLOCK 命令用于给一个锁着的文件解锁。

5. RENAME

格式：RENAME 路径名，路径名[,S#][,D#]

RENAME/卷名/子目录/文件名，/卷名/子目录/文件名

功能：此命令用于更改文件名。注意“,”(逗号)必须放在两个路径名之间。在程序调用下使用此命令时，这逗号必须放在引号外边。

6. SAVE

格式：SAVE 路径名[,S#][,D#]

SAVE/卷名/子目录/BASIC 文件名

功能：此命令是将已建立在内存中的 BASIC 程序存储到磁盘上。在磁盘上的文件类型为 BAS。如果在存盘时显示一个 DISK FULL 信息，则表示磁盘已满，只有程序的最初部分留在磁盘上。

(未完待续)

数制转换简法

福建　南平师范　　李锋

有时临时需将 10 进制数转换为 16 进制数，编个程序总感到麻烦。下面介绍一种直接转化的方法(以十进制数 768 转为 16 进制数为例)

1. 在 BASIC 状态下键入 NEW↙

2. 输入 768? ↙

3. 进入监控，查看 803、804 单元的内容为 00 03，由程序在内存的存放规则可知$(768)_{10}=(300)_{16}$

需要强调的是，需转化的数据一定要做为行号，而不能放在语句体中。

逆用以上思路，可实现 16 进制到十进制的转换。

求　援

我部为修理福建产 LQ－1500K 电源，急需 3CT2KAT 可控硅元件及其代用器件。

浙江杭州延安北路　浙江省机电设备公司办公自动化经营部　刘琳　邮编：310006

也谈扩展 COPY Ⅱ PLUS 4.0

广西柳州市第二中学　关炳坤

贵刊 1990 年第 8 期刊登了牛保军同志的文章《使 COPY Ⅱ PLUS 4.3 工作到 41 磁道》。我参照文中介绍的方法，对《COPY Ⅱ PLUS 4.0》进行了修改，结果未能如愿。原因是《COPY Ⅱ PLUS 4.0》是以文件的形式存在于磁盘上的，由于每个用户得到软件的途径不同，故存放在磁盘上的位置也不同，按文中的方法修改自然不能成功。

我对该软件进行了分析，发现只要修改程序中的一些参数，就可使它的工作范围扩大到 $29 磁道。为了使大家能方便地分析源程序，特把《COPY Ⅱ PLUS 4.0》各功能的入口地址及相应单元的作用列于下表：

序号	功　能	入口地址	相关地址	原值	新值
1	CATALOG DISK	$1203			
2	COPY	$1415	$145A	$44	$52
3	BIT COPY	$18BE	$1F0C	$23	$29
4	DELETE	$1905			
5	LOCK/UNLOCK	$1A60			
6	RENAME FILES	$1A71			
7	FORMAT DISK	$1A97			
8	VERIFY	$1B99	$1C2E	$23	$29
9	TRACK/SECTOR MAP	$202C	$212D	$8C	$A8
10	VIEW FILE	$22AD			
11	FIX FILE SIZES	$251D			
12	CHANGE BOOT PROGRAM	$26A4			
13	UNDELETE FILES	$2743			
14	SECTOR EDITOR	$28A4	$2B16	$23	$29
15	NEW DISK INFO	$2BFB			
16	BOOT DISK	$2C6C			

修改源程序的方法如下：

1. 用 BLOAD COPY Ⅱ PLUS 4.0 把程序调入内存，按表中新值修改相应单元的值（序号为 3 的不改）；

2. 从 $4800 处键入下面程序：

```
4800:A9 2A 8D FE BE 8D EF B3 0A 0A 8D B5 AE
     20 18 11 60
```

3. 用 BSAVE COPY Ⅱ PLUS 4.0,A $7F8,L $4019 把程序存盘；

4. 用 BLOAD COPY Ⅱ PLUS BIT COPY 4.0 把程序调入内存，把 $1F0C 的值改为 $29；

5. 用 BSAVE COPY Ⅱ PLUS BIT COPY 4.0,A $900,L $2808 存盘。

完成了以上五步，就可使 COPY，BIT COPY，FORMAT DISK，VERIFY，TRACK/SECTOR MAP，SECTOR EDITOR 等六项功能工作到 $29 磁道。

从 CP/M 转移文件到 APPLE DOS 下

成都 201 信箱　方福兴

在 APPLE 机中，将文件从 APPLE DOS 转移到 CP/M 下可用 APDOS 方便地进行，而从 CP/M 中的文件转移至 APPLE DOS 下没有专门的软件，转移起来就不那么方便了，特别是 ASCII 码文件。有时要转移一个文件还得从头键入。这里介绍一种方法，可以方便地将文件从 CP/M 中转移至 APPLE DOS 下。

一、ASCII 码文件的转移

先准备一张 HELLO 程序为如下内容的 APPLE DOS 盘，然后将 CP/M 下要转移的文件装入内存，再放入有该程序的盘到主驱动器中，热启动，运行该程序，输入长度、文件名，运行完毕则转移结束。例：

```
10 D$=CHR$(4):INPUT"LEN:";I1:I1=4352+I1
20 INPUT "FILE NAME:";F$
30 PRINT D$;"OPEN";F$
40 PRINT D$;"WRITE";F$
50 FOR I=4352 TO I1:A=PEEK(I):IF A=10 THEN 58
52 PRINT CHR$ (A);
58 NEXT
60 PRINT CHR$(4);"CLOSE";F$
```

1. 放入要转移的文件到内存中（设文件名为 F1）。

```
A>DDT F1
DDT VERS 2.2
NEXT PC
0180 0100
```

2. 放入具有该 HELLO 程序的盘到主驱动器中，热启动，在 LEN: 后键入如上显示差值的十进制值（即 $(0180)_H-(0100)_H=(80)_H=128$），再键入要存入磁盘的文件名，运行完毕则转移结束。

3. 可从 RUN EDASM 装入已转移的文件，仅在文件的头和尾几个字节需修改一下。

二、二进制代码的转移。

先在 CP/M 下按上述方法装入内存，再放入具有较短 HELLO 程序的 APPLE DOS 盘到主驱动器中，热启动，直接用 BSAVE 文件名,A $1100,L $ 长度↙，该长度为上述方法计算的十六进制值。

用这种方法可方便地进行文件的转移。特别是目前单片机的普遍应用，在 APPLE DOS 下的仿真源文件与 CP/M 下的仿真源文件可以结合 APDOS 方便地进行相互转移，这对使用者将带来很大的方便。

多功能CEC-I高速排序

沈阳冶金机械专科学校 187# 王庆顺

关于排序问题，大多使用 BASIC 语言程序来处理。但由于 BASIC 语言运行速度的限制，占机时间都较长，且功能单一，适用性不强。本排序程序用机器语言编写的，特点如下：

1. 速度快，对 1000 个数据排序仅需一、二分钟。

2. 适用于对实型、整型和字符串变量排序。

3. 能对一、二维数组排序，对二维数组的某一列排序时，排序后同行的相关数据不变。

程序一是 BASIC 主程序，程序二是机器语言子程序。现以二维实型数组 A(M,N)为例介绍使用方法。

首先按程序二给出的机器码输入 $9000 开始的内存单元中，然后用“BSAVE PX.OBJ, A $9000, L $190”存盘，供程序一调用。

20 语句：定义数组大小，该语句必不可少。

40 语句：建立“&”跳转地址。

41 语句：为程序二设置参数，并由 45、46 和 47 语句置入指定内存单元。L2 和 L3 的计算公式是不变的。L1 为数据类型代码，取值为整型 2，实型 5，字符串 3。

42 语句：设置排序参数供 50 语句中使用。K2 表示按数组第 K2 列元素进行排序。K1 和 K3 表示两个方面的意义：1、第 K2 列中有(K3－K1)＋1 个元素参与排序；2、排序由 K1 行第 K2 列开始到 K3 行第 K2 列结束。调整 K1 和 K3 的值，可以方便地对数组第 K2 列的任意部分进行排序。各参数有如下要求：

$K1<K3$　　$0\leqslant K1\leqslant M$

$0\leqslant K3\leqslant M$　　$0\leqslant K2\leqslant N$

当 K1＝0、K3＝M 时，K2 列所有无素都参与排序。

50 语句：调用程序二进行排序。

对一维数组排序时，要把程序一中的二维数组改成一维数组，把 50 句改成如下形式：

&A(K1),A(K1),A(K3)

各参数取值与上述二维数组相同，但在 41 语句前应加上“N＝0”。改变排序数组维数或类型时，20、30、60 语句也要相应改变。

程序一是实型数组 A(M,N)，对第 1 列所有元素都参与排序的范例。对于其他情况，用户可以参照程序一编程。

程序一

```
5 HIMEM:36863
7 PRINT CHR$(4);"BLOAD PX.OBJ,A$9000"
10 INPUT "M=,N=";M,N
20 DIM A(M,N)
30 FOR J=0 TO M:FOR I=0 TO N:A(J,I)=100 * RND(1):PRINT A(J,I);" ";:NEXT:PRINT:NEXT
35 PRINT CHR$(7)
40 CALL 36864
41 L1=5:L2=(M+1) * L1:L3=N+1
42 K1=0:K2=1:K3=M
45 POKE 249,L1
46 POKE 29,INT(L2/256):POKE 28,L2-INT(L2/256) * 256
47 POKE 233,INT(L3/256):POKE 232,L3-INT(L3/256) * 256
50 & A(K1,0),A(K1,K2),A(K3,K2)
55 PRINT CHR$(7)
60 FOR J=0 TO M:FOR I=0 TO N:PRINT A(J,I);" ";:NEXT:PRINT:NEXT
70 END
```

程序二

```
9000-A9 0B 8D F6 03 A9 90 8D
9008-F7 03 60 20 E3 DF 84 1F
9010-85 1E 20 BE DE 20 E3 DF
9018-84 07 85 06 38 E5 1E 85
9020-1E 98 E5 1F 85 1F 20 BE
9028-DE 20 E3 DF 84 1B 85 1A
9030-A4 07 A5 06 84 09 85 08
9038-20 07 91 A5 F9 C9 05 F0
9040-0A C9 02 F0 0C 20 A7 90
9048-4C 54 90 20 98 90 4C 54
9050-90 20 EC 90 C9 01 D0 03
9058-20 07 91 A5 09 A4 08 20
9060-76 91 D0 0E A4 09 A5 08
9068-20 6B 91 84 09 85 08 4C
9070-3B 90 20 10 91 A5 07 A4
9078-06 20 76 91 D0 0E A4 07
9080-A5 06 20 6B 91 84 07 85
9088-06 4C 30 90 60 18 D8 65
9090-1C AA 98 65 1D A8 8A 60
9098-A4 FD A5 FC 20 F9 EA A4
90A0-09 A5 08 20 B2 EB 60 A0
90A8-00 B1 08 85 A8 D1 FC 90
90B0-0A F0 0C B1 FC 85 A8 A9
90B8-FF D0 06 A9 01 D0 02 A9
90C0-00 85 A7 C8 B1 FC 85 A5
90C8-B1 08 85 42 C8 B1 FC 85
90D0-A6 B1 08 85 43 A0 00 B1
90D8-42 D1 A5 F0 07 A9 FF B0
90E0-02 A9 01 60 C8 C4 A8 D0
90E8-EE A5 A7 60 A0 00 B1 08
90F0-D1 FC F0 02 D0 09 C8 A2
90F8-00 B1 08 D1 FC F0 06 A2
9100-FF B0 02 A2 01 8A 60 A5
9108-08 85 FC A5 09 85 FD 60
9110-A5 06 A4 07 20 88 91 86
9118-3C 85 3D A5 FC A4 FD 20
9120-88 91 86 3E 85 3F A0 00
9128-84 42 84 43 B1 3E 48 B1
9130-3C 91 3E 68 91 3C C8 C4
9138-F9 D0 2C E6 42 D0 02 E6
9140-43 A5 42 C5 E8 F0 02 D0
9148-06 A5 43 C5 E9 F0 1B A4
9150-3F A5 3E 20 8D 90 84 3F
9158-85 3E A4 3D A5 3C 20 8D
9160-90 84 3D 85 3C A0 00 4C
9168-2C 91 60 D8 18 65 F9 AA
9170-98 69 00 A8 8A 60 C5 1B
9178-90 0B F0 02 B0 03 98 C5
9180-1A A9 FF B0 02 A9 00 60
9188-38 E5 1E AA 98 E5 1F 60
```

小学生速算训练

武汉江岸车辆厂银行　李永升

本程序可由学生自己设定数值的大小，训练项目多，适应不同年级的小学生运用。

70～100 句为选择类型，做多少题，任意设定数值，K 计算出每题分值。

130 句　根据选择的类型转随机出题的子程序。

140～180 句 对答案进行判断，加分。

210 句 计算答案，打印被乘数，乘数。

280～310 句 出两数相减的随机数。

390～420 句 出随机平均数。

590～620 句 出随机若干个连续自然数。

程序在 CEC-I 中华学习机上通过。

```
10 PRINT CHR $ (4)"PR # 3":PRINT
20 T=0:HGR2:HOME
30 PRINT"1. 两数相加","2. 两数相减","3. 多位数相加","4. 求平均数","5. 两数相乘","6. 两数相除","7. 11－99 的平方","8. 连续整数相乘","9. 尾数 5 的平方",
40 PRINT"10. 尾数 5 的乘法","11. 100 以上相乘","12. 100 以下相乘","13. 求若干个连续整数之和"
50 PRINT"14. 个位数字之和为 10 的两数相乘"
60 PRINT"15. 个位相同其余数之和为 10 的数相乘"
70 INPUT X:HOME:INPUT"你要做多少道题？(3－100)";P:K=INT(100/P)
80 INPUT"请输入位数=";M
90 IF X=5 THEN INPUT"请输入乘数=";B
100 IF X=6 THEN INPUT"请输入除数=";B
110 FOR I=1 TO P:HOME:VTAB 5:HTAB 5
120 A=INT(10^ (M－1)+RND(1) * 9 * 10^ (M－1))
130 ON X GOSUB 260,280,320,390,210,380,430,450,460,530,550,570,590,470,500
140 INPUT C:IF C=Z THEN 170
150 PRINT:PRINT TAB(10)"计算错误!"
160 PRINT:PRINT TAB(8)"正确答案是:";Z:GOTO 180
170 VTAB 7:HTAB 10:PRINT"计算正确！加";K;"分":T=T+K
180 FOR J=1 TO 1000:NEXT J
190 NEXT I
200 HOME:GOTO 220
210 Z=A * B:PRINT A"×"B"=";:RETURN
220 VTAB 8:HTAB 10:PRINT"成绩=";T;"分"
230 PRINT:VTAB 9:HTAB 10:PRINT"是否还做？(Y/N)";
240 GET A $ :IF A $ ="Y"OR A $ ="y" THEN 20
250 END
260 B=INT(10^ (M－1)+RND(1) * 9 * 10^ (M－1))
270 Z=A+B:PRINT A"+"B"=";:RETURN
280 A=INT(10^ (M－1)+RND(1) * 9 * 10^ (M－1))
290 B=INT(10^ (M－1)+RND(1) * 9 * 10^ (M－1))
300 IF B〉=A GOTO 280
310 Z=A－B:PRINT A"－"B"=";:RETURN
320 B=INT(10^ (M－1)+RND(1) * 9 * 10^ (M－1))
330 D=INT(INT(1+5 * RND(1)) * 10^ M－A+.5)
340 E=INT(INT(1+5 * RND(1)) * 10^ M－B+.5)
350 F=INT(10^ (M－1)+RND(1) * 9 * 10^ (M－1))
360 Z=A+B+D+E+F
370 PRINT A"+"B"+"D"+"E"+"F;"=";:RETURN
380 Z=A/B:RPINT A"÷"B"=";:RETURN
390 NU(1)=101+INT(850 * RND(1)):CC=NU(1):VTAB(3):HTAB(4):PRINT NU(1)
400 FOR J=2 TO 10:NU(J)=NU(1)+INT(20 * RND(1)):CC=CC+NU(J):VTAB(3+2 * (J〉5)):HTAB(6 * (J－5 * (J〉5))－2):PRINT NU(J):NEXT
410 VTAB(10):HTAB(1):CALL 64578:HTAB(20)
420 Z=CC/10:PRINT"平均数=";:RETURN
430 A=11+INT(89 * RND(1))
440 Z=A * A:PRINT A;"的平方=";:RETURN
450 B=B+1:GOTO 210
460 A=10 * A+5:Z=A * A:PRINT A;"的平方=";:RETURN
470 A=INT(10^ (M－2)+RND(1) * 9 * 10^ (M－2))
480 B=1+INT(9 * RND(1))
490 A=A * 10+B:B=A+10－2 * B:GOTO 210
500 C=10+INT(9 * RND(1)) * 10
510 B=1+INT(9 * RND(1))
520 A=C+B:B=100－C+B:GOTO 210
530 A=10 * INT(40 * RND(1))+5
540 B=10 * INT(20 * RND(1))+5:GOTO 210
550 A=100+ INT(20 * RND(1))
560 B=100+ INT(20 * RND(1)):GOTO 210
570 A=100－ INT(20 * RND(1))
580 B=100－ INT(20 * RND(1)):GOTO 210
590 A=1+ INT(300 * RND(1))
600 B=5+ INT(100 * RND(1))
610 Z=(A+A+B－1) * B/2
620 PRINT A"+"A+1"+"A+2"+……+"A+B－1"=";:RETURN
```

更　正

本刊 1991 年第 4 期《在 dBASE III 中实现多维数组功能》一文，因对作者名字辨认不清，误将“石洪泰”看成“石供泰”，谨向石洪泰先生致歉。

又：同期《工资票面张数计算程序的简化》一文，作者陈承灿先生的名字被误排成“孙承灿”，校对时未能发现，特向陈承灿先生表示歉意。

C 语言初阶讲座

第七讲　结构数据(二)

李文兵

3. 结构

结构是结构数据的一种常用形式,它是用户自由设定,由基本数据类型构造出来的组合数据类型。

(1)结构类型的定义　结构数据类型的定义方法有两种。

〈第一种定义方法〉　其定义格式如下:

struct 结构数据类型名{组员表}

其中结构数据类型名也叫标记,是人为设定的;成员是根据需要设置的;成员的组合就是结构类型的实在模式。例如:

```
struct date{
int month;
int day;
int year;
};
```

就定义了由 3 个整型数据组成的一个结构数据类型,起名就叫 date。

〈第 2 种定义方法〉　使用数据类型定义关键字定义,如:

```
typedef struct{
    int month;
    int day;
    int year;
}date;
```

这里同样定义了包含有 3 个整型数据组成的一个结构数据类型 date。注意,typedef 可以给已经存在的数据类型起新的名字,但不能定义新的存储类别。

(2)结构变量的定义　结构变量有 2 种定义方式,比如,要定义具有 date 这种数据类型的变量,方法如下。

〈第一种定义方法〉　与结构类型分开定义,即:

```
struct date d1,d2[5];
```

其中 d1 和 d2 分别是 date 这种类型的一个变量和一个数组。不过,注意 d2[5]的每个元素均为 date 这样的结构数据,共有 5 个这样的元素。

〈第二种定义方法〉　与结构类型同时定义,如:

```
struct date{
    int month;
    int day;
    int year;
  }d1,d2[5];
```

(3)结构成员的初始化与调用　结构变量定义后,便可使用成员运算符及赋值运算符,对其成员赋初值,如:

```
d1.month=2;
d1.day=7;
d1.year=1991;
```

可见,使用结构成员运算符·,便可指定结构的成员,即

结构变量.成员名

使用 printf()和 scanf()这两个库函数,也可以对结构成员进行输入与输出,如练习 7.11 所示。

```
A>TYPE EXP7__11.C
struct date{
      int month;
      int day;
      int year;}
main()
     { struct date d;
       scanf("%d",&d.year);
       scanf("%d",&d.month);
       scanf("%d",&d.day);
       printf("%d ",d.year);
       printf("%d ",d.month);
       printf("%d ",d.day);
A>EXP7__11
1991 1 23
1991 1 23
```

对于结构数组,其赋值方法与一般变量相同。例如:对于结构数组 d2[5],要对其 3 号元素 d2[3]的各成员赋值,可按如下进行:

```
d2[3].month=5;
d2[3].day=27;
d2[3].year=1990;
```

(4)结构的嵌套使用　结构还可以做另一结构的成员,即结构可以嵌套使用。在 Turbo C 中,结构嵌套层次不限,但常用的为 2 重和 3 重,如:

```
struct date{
      int month;
      int day;
      int year;
      };
struct address{
      char name[20];
      char city[50];
      struct date zip;
      }p;
```

这里,在结构 address 中就嵌套了结构 date。这时,对变量 p 的赋值和调用,则如下所示:

```
p.zip.month=9;
printf("%d",p.zip.month);
```

结构嵌套的例子如练习 7.12 所示。

```
A)type exp7__12.c
struct date {
            int month;
            int day;
            int year;
            };
struct date a1,a2[5];
struct class {
             char name[10];
             struct date a3;
             int grade;
             };
typedef struct {
               char address[20];
               int tel;
               float num;
               struct date a4;
               struct class a5;
               }ddate;
main()
{     printf("size of date=%d\n",
               sizeof(struct date));
      printf("size of class=%d\n",
               sizeof(struct class));
      printf("size of ddate=%d\n",
               sizeof(ddate));
}
```

```
C)exp7__12
size of date=6
size of class=18
size of ddate=50
```

4. 字段结构

C 语言可以进行字单元内的位操作，因此，在象对待布尔(Boolean)变量那样，进行 1 位(true 或 false)操作的场合，可在一个字单元内放置多个 Boolean 变量，这样，可节省存储空间。此外，在控制外设时，也必须进行位操作。结构是适合位操作的数据类型。这样的结构就叫字段结构，也叫位域。

(1)字段结构的定义　其定义格式如下所示：

```
struct 字段结构类型名{
数据类型　字段 1 名:长度
数据类型　字段 2 名:长度
    :
}字段结构变量名;
```

这里，结构成员都是不满一个字的整型数据，故叫字段。字段名后(冒号后)的长度表示该字段的长度，即占几位，其值当然要小于 cpu 字长。注意，数据类型为 int，unsigned。例如练习 7.13 所示。

```
C)type exp7__13.c
main()
{ struct bit
  { unsigned f1:3;
    unsigned f2:5;
    int      f0:5;
    unsigned f3:1;
    unsigned f4:2;
    }bit__code;
    bit__code.f1=0x2;
    bit__code.f2=0x6;
    bit__code.f3=1;
    printf("%x %x %x\n",
    bit__code.f1,bit__code.f2,bit__code.f3);
    printf("%x\n",bit__code);
}
C)exp7__13
2 6 1
2032
```

(2)字段结构的用法及注意事项：

①字段不存在地址，故不能使用运算符 &，因而，也无指向字段的指针。

②在 PC 机上，字段在一个字上的分配方向是从右至左，如上例，其字段 f_1、f_2、f_0、f_3、f_4 的分布情况如图 7.3 所示。

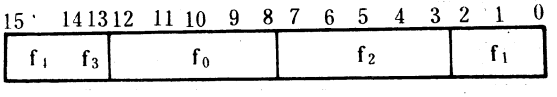


图 7.3　字段分配方向

③不能构造字段数组。

④如果字段未占满一个字存储单元，余下的位将保持原值不变。若余下的位已放不下某字段，则该字段开辟下一个存储单元使用。

⑤若想强制某字段放在一个字的开始部分，对 Turbo C 来说，只要在该字段的前面，放置一个长度为 1 的字段即可。

⑥无名字段(即无字段名)用来填空，所填的位数即该字段的长度。

上述用法如练习 7.14 所示。

```
C)type a:exp7__14.c
struct bitfield {
     unsigned d1:3;
     unsigned d2:4;
     unsigned d3:4;
     unsigned d4:16;
     unsigned d5:8;
     unsigned   :4;
     unsigned d6:5;
     unsigned d7:5;
     } bit__code;
main()
     { unsigned *p;
       p=&bit__code;
       bit__code.d1=5;
```

```
        bit __code.d2=10;
        bit __code.d3=9;
        bit __code.d5=0xaa;
        bit __code.d6=12;
        bit __code.d7=0x0a;
        printf("%x\n",*p);
        printf("%x\n",*(p+1));
        printf("%x\n",*(p+2));
        printf("%x\n",*(p+3));
    }
C)exp7 __14
4d5
0
c0aa
14
```

5. **共体**(union)

共体(union)是由使用同一存储空间的多个变量组成的结构数据。由于其各组成变量使用同一存储空间,故叫共体;有些专家直接翻译为联合。其定义例子如下所示:

```
union rec{
    int var1;
    int var2;
    char var3;
    char var4;
}
```

该定义确定了一个由 var1、var2、var3、var4 四个变量组成的共体类型 rec。四个变量的存储分配,如图 7.4 所示。

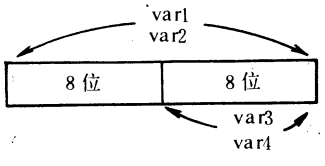


图 7.4　共体 rec 的存储分配

又如,下面的定义:

```
union rec{
    int vari;
    char var2[2];
}
```

使变量 var1 所占用的低位字节和高位字节同时用来存储 var2[0]和 var2[1],其程序如练习 7.15 所示。

```
A)TYPE EXP7 __15.C
union rec{
        unsigned int var1;
        char var2[2];
        }
main(){ union rec v;
         v.var1=0xaa66;
         printf("v.var1=%x\n",v.var1);
         printf("v.var2[0]=%x\n",v.var2[0]);
         printf("v.var2[1]=%x\n",v.var2[1]);
         }
A)EXP7 __15
v.var1=aa66
v.var2[0]=66
v.var2[1]=ffaa
```

可见,在共体中,同一存储空间可被多个变量使用。

模拟 8086 cpu 寄存器的结构设在标题文件 dos.h 中。该结构如下所示:

```
struct WORDREGS{
        unsigned int ax,bx,cx,dx,si,di,cflag,flags;};
struct BYTEREGS{
        unsigned char al,ah,bl,bh,cl,ch,dl,dh;};
union REGS{
        struct WORDREGS x;
        struct BYTEREGS h;
};
struct SREGS {
        unsigned int es;
        unsigned int cs;
        unsigned int ss;
        unsigned int ds;
};
struct REGPACK{
   unsigned r __ax,r __bx,r __cx,r __dx;
  unsigned r __bp,r __si,r __di,r __ds,r __es,r __flags;
};
```

其中,共体 REGS 是由两个结构组成的。AX 寄存器的高 8 位和低 8 位分别用 ah 和 al 表示。使用共体 REGS,表示系统时间的程序如练习 7.16 所示。

```
A>TYPE EXP7 __16.C
#include "dos.h"
main()
{union REGS in,out;
        in.h.ah=0x2c;
        intdos(&in,&out);
        printf("time:%2d:%2d:%2d",
        out.h.ch,out.h.cl,out.h.dh);
}
A>EXP7 __16
time:16:32:14
```

（天大分校蓝智斌同志参加了程序调试）

学用单片机

MCS-51 单片机应用中二种 LED 显示方式

傅 篱

在单片机应用中，常常要采用多位八段 LED 显示方式。较为常见的 LED 方式，其电原理如图 1。

软件流程图如图 2。

在这种 LED 显示方式，每位 LED 是依次刷新，每位显示时间约 1ms。这样，显示电流变化较大，频率较高，当开机无键入时，LED 显示器以毫秒级时间不断刷新显示。如果系统中接有喇叭，这种方法就不适用。喇叭中将产生明显的交流噪声。

为此，在 MCS－51 系列单片机应用中，可利用串行 I/O 口，外接 74LS164 移位寄存器，在方式 0 的工作方式下，将显示段码数据，串行发送出去，使 LED 显示器根据需要显示相应数据。其电原理图如图 3。

软件流程如图 4。

这种 LED 显示方式和上种不同之处，在于后者为

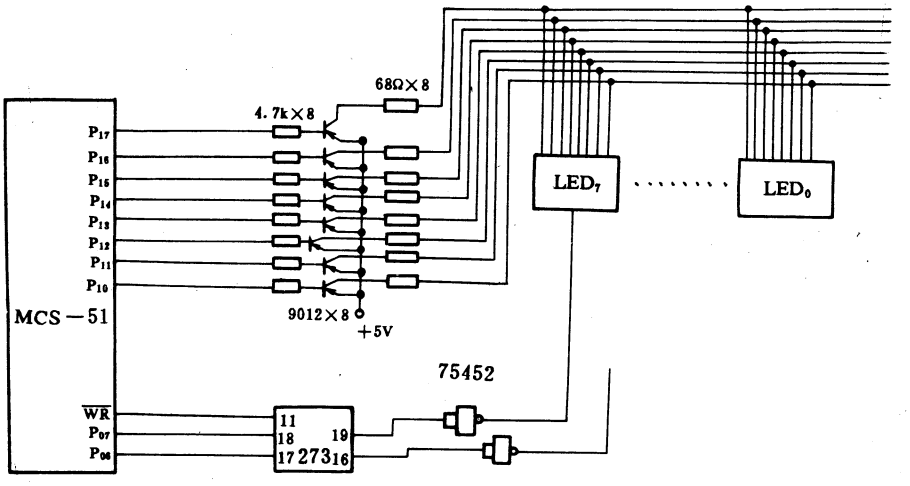


图 1

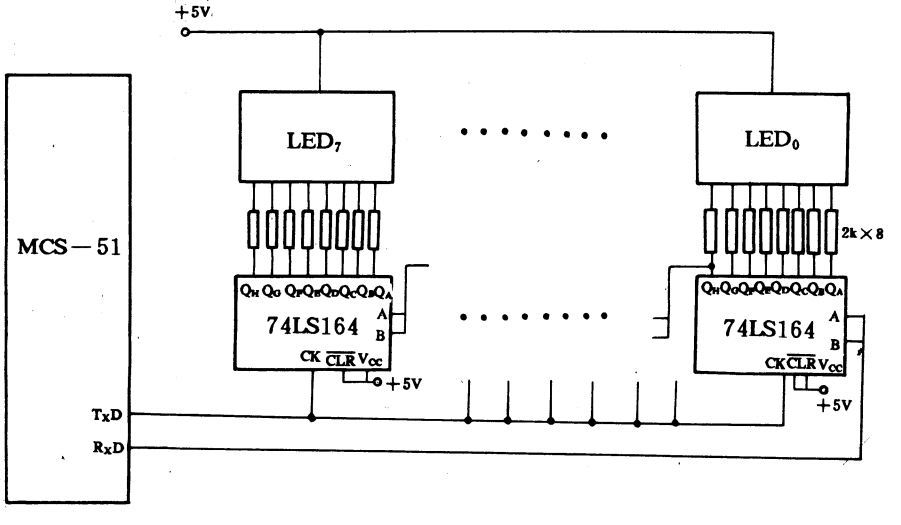


图 3

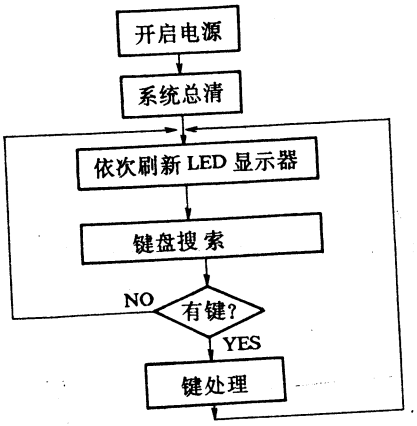


图 2

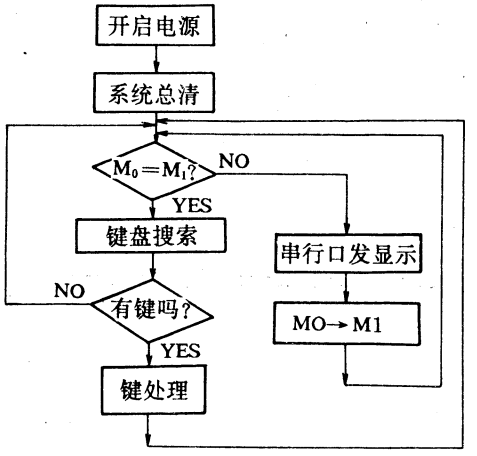


图 4

静态显示。如果 M_0 和 M_1 相同，则保持原显示数据；只有 M_0 不等于 M_1，显示器显示数据才需要改变。这样，LED 显示器的显示变化在秒级范围内变化，只要不按动键盘，显示器就不需要改变显示。这种显示方式虽克服了第一种显示方式的不足，但显示器的电流消耗却大了许多。所以，读者应根据自己设计要求，选择 LED 显示方式。

（上接 31 页）

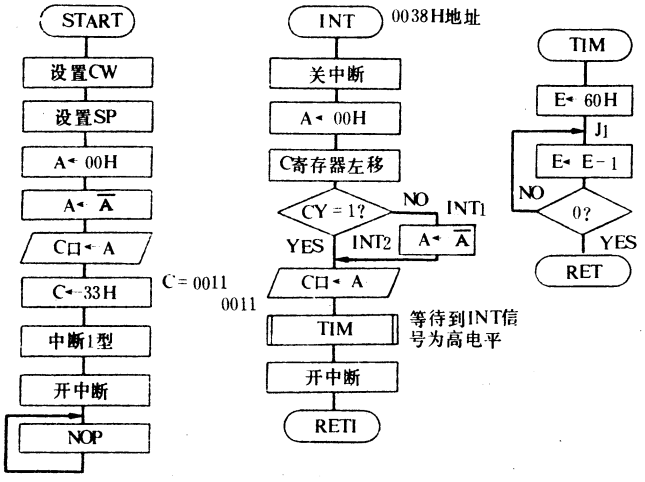


图 12

相位控制部件的应用不局限于上述事例。有兴趣的读者，不妨尝试一下。

表 3

标记	助记符	地址	机器语	注　　释
START	LD A,90H	0000	3E 90	设置输入输出
	OUT(03H),A	0002	D3 03	
	LD SP,0100H	0004	31 00 01	SP←0100H
	XOR A	0007	AF	A←FFH
	CPL	0008	2F	
	OUT(02H),A	0009	D3 02	C口为高电平
	LD C,33H	000B	0E 33	33H送入C寄存器
	IM1	000D	ED 56	中断1型
	EI	000F	FB	开中断
LOOP	NOP	0010	00	保存
	JP LOOP	0011	C3 10 00	
INT	DI	0038	F3	关中断
	XOR A	0039	AF	A←00H
	RLC C	003A	CB 01	C寄存器左移
	JP NC,INT1	003C	D2 47 00	
INT2	OUT(02H),A	003F	D3 02	C口输出低电平
	CALL TIM	0041	CD 60 00	延时
	EI	0044	FB	开中断
	RETI	0045	ED 4D	中断结束
INT1	CPL	0047	2F	A←FFH
	JP INT2	0048	C3 3F 00	
TIM	LD E(60H)	0060	1E 60	延时
J1	DEC E	0062	1D	延时常数送入E寄存器
	JP NZ,J1	0063	C2 62 00	
	RET	0066	C9	

学装 μp—80

相位控制部件的制作

易齐干

交流电为正弦波，如果使用可控硅遮断正弦波的一部份，以无级方式控制相位变化，如图 1 所示。再引进 μp—80 微电脑，则能以简单的电路实现对交流电进行准确控制。

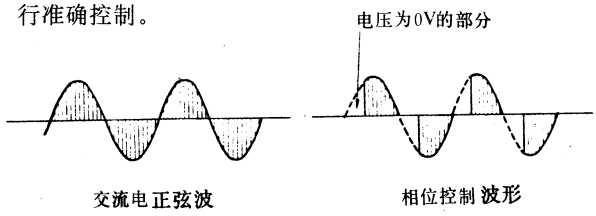


图 1. 交流电正弦波和相位控制波形

1. 相位控制部件

相位控制部件电路原理图如图 2 所示。

图中 S102DD2 和 PC827 是夏普公司制作的光可控硅耦合器和光敏晶体管耦合器。PC827 将交流电正弦波以 0V 时序输入给微电脑。S102DD2 是通过微电脑对交流电正弦波进行相位控制。

晶体管 2SC1815 接受微电脑信号，控制驱动 S102DD2。

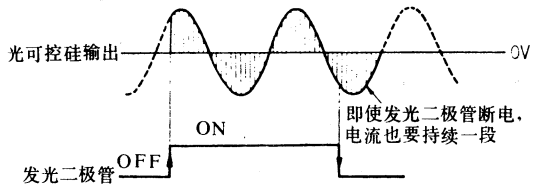


图 4　光可控硅的动作

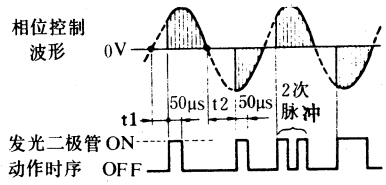


图 5　相位控制的方式

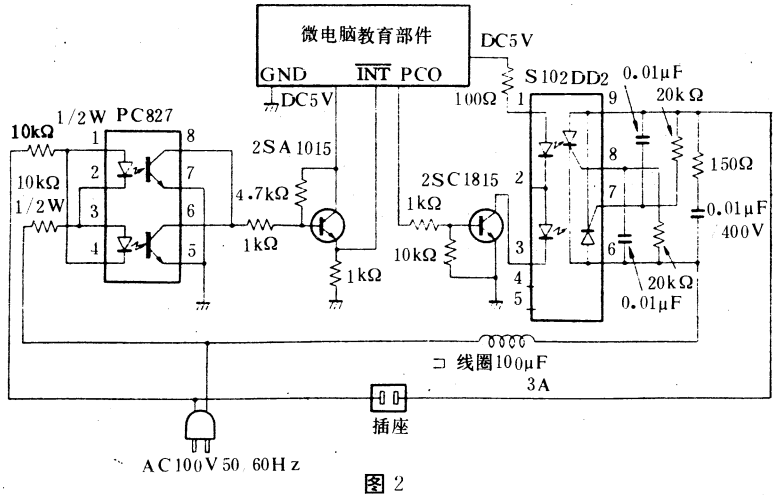


图 2

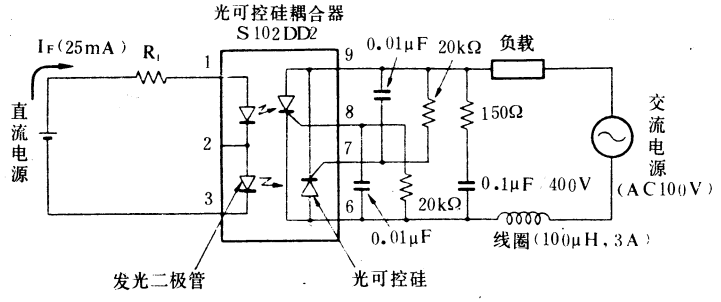


图 3

(1) 光可控硅耦合器—S102DD2

光可控硅耦合器(S102DD2)如图 3 所示。内部结构是两组发光二级管和光可控硅半导体装在一个盒内。如果有 25mA 的电流 IF 由引脚 1 流向引脚 3，则发光二级管发光，打开光可控硅，引脚 6 与引脚 9 之间有交流电流通过。

引脚 6、7、8、9、之间设置电阻电容，是防止由於引脚 6 与 9 之间电压急骤增加，正弦波形增大而产生误动作。交流正弦波在通一断变化时产生噪声，为防止噪声向外部释放，电路中设置 100μH 的线圈。

引脚 1 串联的电阻 R_1 起限流作用，使发光二极管的电流在 25mA 左右。

光可控硅的特点是无触点，寿命长，高速动作，输出延续。如图 4 所示，发光二级管即使熄灭，交流电变为 0V 之前，光可控硅输出仍然持续一段时间。

(2)相位控制

由光可控硅耦合器的动作很

容易理解：交流电的正弦波由 0V 开始的任意时间由微电脑驱动发光二极管产生触发脉冲实现相位控制，如图 5 所示。

触发脉冲宽度过窄，光可控硅可能没有应答。所需要的最低限度由负载种类和相位状态决定。必须通过实验验证。例如，触发脉冲宽度为 50μS 以上，均有应答。为提高可靠性，使脉冲连续触发两次更好。

(3)交流电 0V 检测

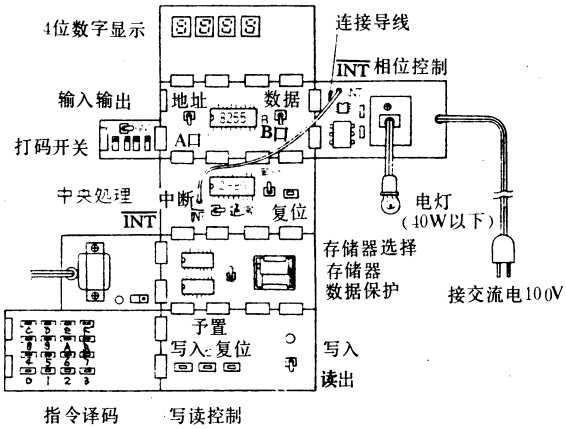
PC827 光晶体管耦合器也是一种半导体元件。结构也是一个盒内装有两组发光二级管和光敏晶体管。发光二极管一发光，光敏晶体管导通。

它检测交流电 0V 时序，输入给微电脑的电路原理图如表 1。

2. 应用例

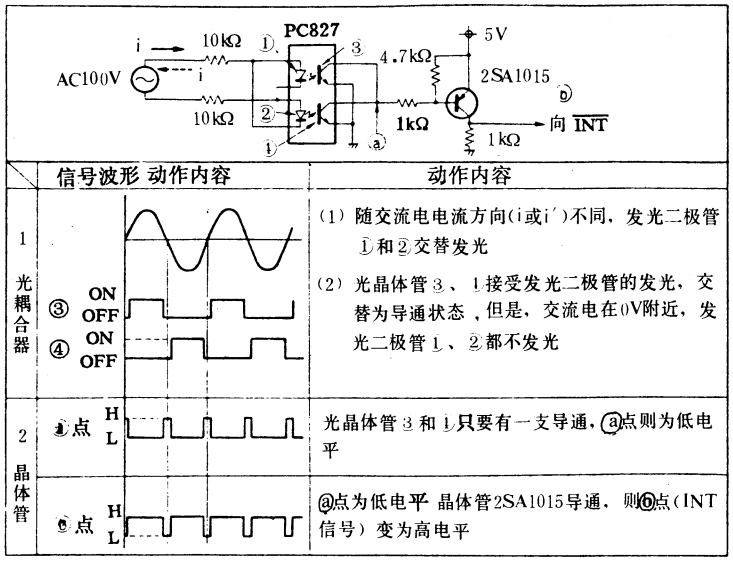
(1)照明灯亮度调节

照图 6 连接 up80 微电脑套件和相位控制部件，使



图

表 1



	信号波形 动作内容	动作内容
1 光耦合器	③ ON OFF ④ ON OFF	(1) 随交流电电流方向(i或i')不同，发光二极管①和②交替发光 (2) 光晶体管③、④接受发光二极管的发光，交替为导通状态，但是，交流电在0V附近，发光二极管①、②都不发光
2 晶体管	ⓐ点 H L	光晶体管③和④只要有一支导通，ⓐ点则为低电平
	⑤点 H L	ⓐ点为低电平 晶体管2SA1015导通，则⑤点(INT信号) 变为高电平

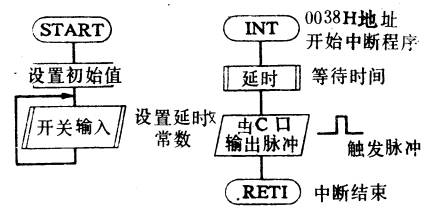
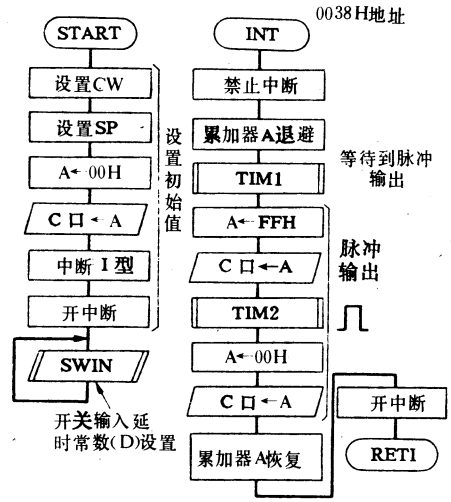


图 7



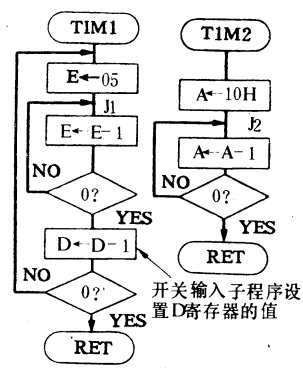


图 8(1)

用打码开关实现照明灯亮度 4 级调节。

完成这个实验很容易。由前进相位控制部件工作原理可知，交流电为 OV 的瞬间向微电脑申请中断。CPU 自动执行 0038H 地址开始的中断程序。中断程序的内容是光可控硅耦合器的触发脉冲产生程序，它出现於由打码开关控制所决定的既定时间之后。

程序流程图如图 7、图 8 所示，程序清单如表 2。

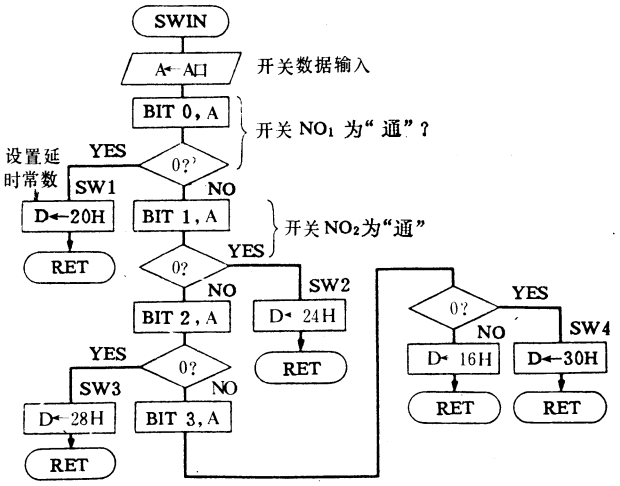


图 8(2)

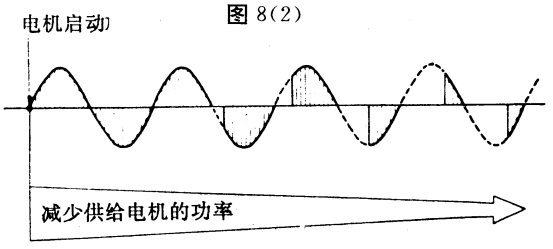


图 9

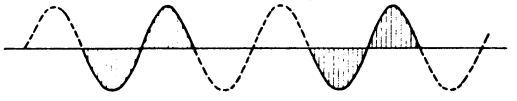


图 10

图 11

这种相位控制部件能够直接控制 40W 照明灯。

(2)交流电机转速控制

用交流电机代替照明灯,上述程序可以控制电机的转速。

如果适当变动程序,如图 9 所示,电机启动时正弦波不变,启动之后,进行相位控制,可以实现电机低速运转。

(3)零伏电压开关

除相位控制之外,如图 10 所示,交流电正弦波以 1 个周期为单位实现通、断控制,这就是零伏电压开关。与相位控制的波形不一样,正弦波无急骤变化,所以对外部噪声影响很小。

它主要用于加热器的温度控制,或用于对噪声干扰要求较严格的地方代替继电器。

程序流程图如图 11 所示。将 33H 放入寄存器 C,作为波形状态向外输出。在正弦波为 0V 时,每发生一次中断,寄存器 C 的数值向左移位,有进位输出时正弦波半周期为 OFF,没有进位输出时,正弦波的半周期为 ON。这样,实现交流电正弦波一个周期中为通一断状态。程序清单如表 3 所示。 (下转 28 页)

表 3

标记	助记符	地址	机器语	注　　释
START	LD A,90H	0000	3E 90	设置输入输出
	OUT(03H),A	0002	D3 03	
	LD SP,0100H	0004	31 00 01	SP←0100H
	XOR A	0007	AF	C口输出低电平
	OUT(02H),A	0008	D3 02	
	IM 1	000A	ED 56	中断1型
	EI	000C	FB	开中断
LOOP	CALL SWIN	000D	CD 80 00	
	JP LOOP	0010	C3 0D 00	
INT	DI	0038	F3　　0	
	PUSH AF	0039	F5	保存A累加器
	CALL TIM 1	003A	CD 60 00	延时
	LD A,FFH	003D	3E FF　0	C口输出高电平
	OUT(02H),A	003F	D3 02	
	CALL TIM 2	0041	CG 70 00	脉冲宽度
	XOR A	0044	AF　　0	C口输出低电平
	OUT(02H),A	0045	D3 02	
	POP AF	0047	F1	
	EI	0048	FB	
	RETI	0049	ED 4D	中断结束
TIM 1	LD E,05H	0060	1E 05	延时子程序
J1	DEC E	0062	1D	
	JP NZ,J 1	0063	C2 62 00	
	DEC D	0066	15　　0	
	JP NZ,TIM 1	0067	C2 60 00	
	RET	006A	C9	
TIM 2	LD A,10H	0070	3E 10	脉冲宽度延时
J 2	DEC A	0072	3D	
	JP NZ,J2	0073	C2 72 00	
	RET	0076		
SWIN	IN A,(00H)	0080	DB 00	A口输入
	BIT 0,A	0082	CB 47	
	JP Z,SW1	0084	CA 99 00	
	BIT 1,A	0087	CB 4F	
	JP Z,SW2	0089	CA 9C 00	
	BIT 2,A	008C	CB 57	
	JP Z,SW3	008E	CA 9F 00	
	BIT 3,A	0091	CB 5F	
	JP Z,SW4	0093	CA A2 00	
	LD D,16H	0096	16 16	设置延时常数
	RET	0098	C9	
SW1	LD D,20H	0099	16 20	设置延时常数
	RET	009B	C9	
SW2	LD D,24H	009C	16 24	设置延时常数
	RET	009E	C9	
SW3	1D D,28H	009F	16 28	设置延时常数
	RET	00A1	C9	
SW4	LD D,30H	00A2	16 30	设置延时常数
	RET	00A4	C9	

怎样用电脑检修录像机

李玉全

用电脑检修录像机有自动检测和辅助检测二种方式。

一、自动检测

实现电脑自动检测录像机，关键是先把录像机正常时的各有关测试点电压、波形、机内有关元件的各种相关值，按一定程序固化在微处理器中检修录像机时，把探头接入测试点，取样（取电压、波形）送给电脑与正常数据比较，找出可能发生故障的元件。也可把录像机工作正常时，各种参数（主要相关元件参数）编出程序，存入电脑中。使用时，通过模/数（接口电路）转换电路，把电压、波形信号变成数字信号送给电脑。通过电脑把检测到的信号与标准信号（机内存的程序）进行比较，显示器显示出故障所在。

录像机工作时，磁鼓的转速、主导轴转速、带盘的转速要保持一个恒定速度。当这些关键部件转速发生变化说明录像机出现故障。录像机的各种转动部分的检测也是维修中的一大环节。测转速时，用霍尔元件、磁敏元件或光敏元件作成检测探头。在检修时，探头检测到转动信号，通过转换电路把单位时间内电流的变化，变换成数字信号。把数字信号输入电脑，进行比较处理，判断故障所在，显示在显示屏上。

二、辅助检测

用电脑辅助检测录像机，已达到了实用阶段。这种检测方法大大地提高了故障判断时间，提高了故障判断的准确率（因录像机中设计了多种自动保护电路，有时无法分清是否出现故障）可实现人和机“对话”。把录像机各种故障现象都编制成程序，把程序存入电脑中，使用时将录像机的各种故障现象，送入（键入）电脑，在电脑的指导下找出故障所在。如“程序一”是NV－450录像机“电源自动保护检修过程”。

```
5   REM“程序一”
10  LPRINT“松下NV－450录像机”
20  LPRINT“电源自动保护检修过程”
25  LPRINT“请先生遵照下列步骤进行”
30  LPRINT“首先检查VEP03236H电路板上的电源”
35  LPRINT“电路板的Q1001基极是否为高电平?”
40  INPUT“V1$＝”;V1$
50  IF V1$＝“是”THEN 80
60  LPRINT“请检查VEP03236H电路板上系统控制”
61  LPRINT“电路IC6001(MN1542VG1)42脚是否为”
62  LPRINT“低电平?”
65  INPUT“V2$＝”;V2$
67  IF V2$＝“是” THEN 90
70  LPRINT“请测量IC6001的22脚是否为低电平?”
75  INPUT“V3$＝”;V3$
77  IF V3$＝“是” THEN 100
78  LPRINT“应检查电源开关(POWER SW)S6559”
79  GOTO 130
80  LPRINT“应检查Q1001(2SD1275)”
85  GOTO 130
90  LPRINT“应检查QR6005(DTC124A)”
95  GOTO 130
100 LPRINT“请测量IC6001的16脚在电源接通瞬”
101 LPRINT“间是否为低电平?”
110 INPUT“V4$＝”;V4$
112 IF V4$＝“是”THEN 120
115 LPRINT“应检查Q6002(2SD636)”
116 GOTO 130
120 LPRINT“请检查IC6001”
130 LPRINT“祝你成功！欢迎合作!!”
140 END
```

三、用电脑作信号发生器

用电脑可输出各种图形信号，如屏幕中间一条横线、一条竖线、方格信号、垂直条信号、水平条信号、圆形信号，有的还能输出彩条信号。用电脑给录像机输入一图形信号，记录后再重放，观察图象情况，并和记录时的图形进行比较，判断是记录还是重放故障。

这种方法和使用图形信号发生器相同，这样可提高电脑的利用率，给录像机维修带来方便。

用娃娃机开发单片机程序

河南省新闻出版局办公室　张　涛

目前，MCS－51单片机在电脑爱好者中日益普及，早期已经曾及的各种娃娃电脑却大有日趋闲置之势。但业余爱好者手中的单片机系统，程序只能手工汇编，费时费力。而LASER、COMX等各种娃娃机，大都

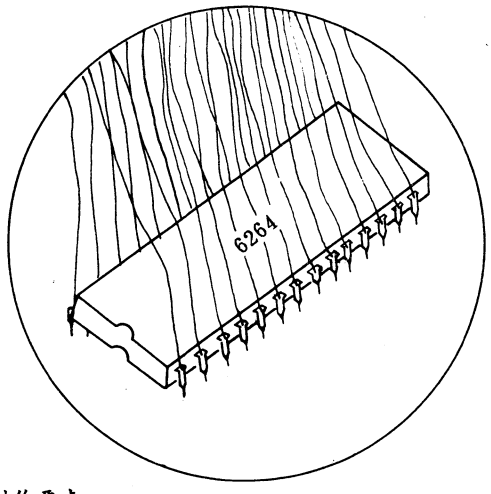
具有比较完善的屏幕显示和英文键盘，如能配上合适的汇编软件和接口电路来为单片机开发程序，必将给广大单片机爱好者带来莫大的方便。由于各种娃娃机的内存都比较小，如果汇编软件占内存量过大，则许多低档娃娃机就难以利用，因此，开发此类软件必须最大可能地减少其占内存量。

笔者经过潜心研究，在LASER机上开发成功了占内存仅1.5K的汇编程序，具有汇编、反汇编、源程序编辑、小汇编和内存编辑五大功能，配合笔者设计的超简易存储器仿真电路，使单片机的程序开发大为简便。下面介绍该程序的设计思想和设计中运用的主要技巧，希望有助于广大电脑爱好者在各种娃娃机上开发出更好的汇编软件，使单片机在业余爱好者中的应用水平得到进一步提高。

1、反汇编：在反汇编部分

扩展内存插口各引脚排列如下(面对插口时)：

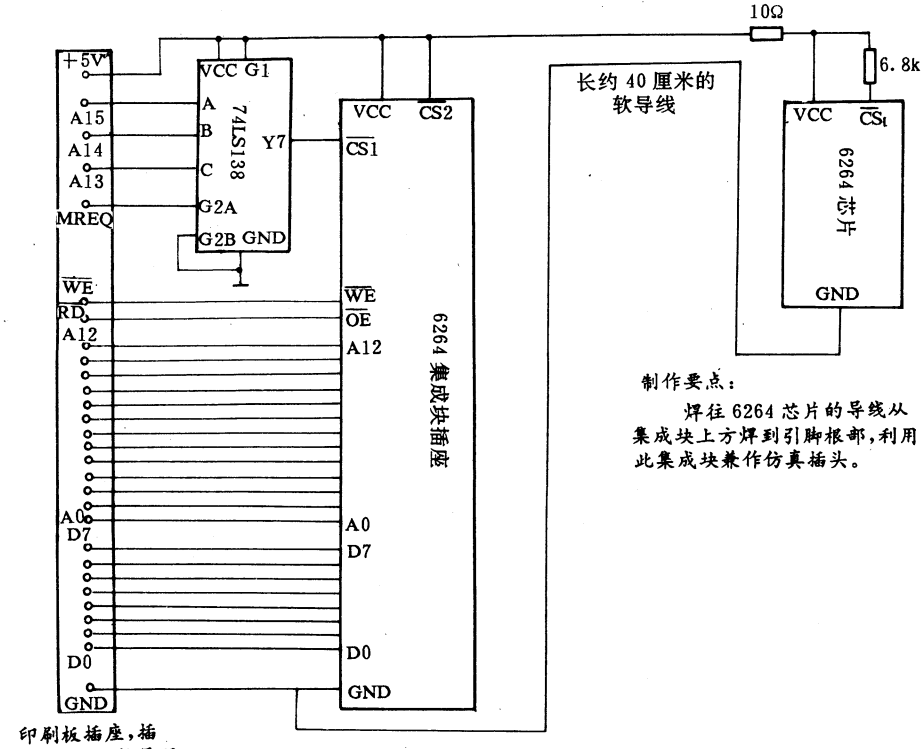
NC	+5V	IORQ	RD	NMI	WAIT	M1	RFSH	D7	D2	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	RESET	GND
22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23
NC	+9V	NC	WR	MREQ	HALT	INT	D1	D0	A0	NC	D6	D5	D3	D4	Φ	A15	A14	A13	A12	A11	GND



制作要点：

导线从集成块上方焊到各引脚根部，利用此集成块兼作仿真插头，注意焊点不要太靠下，以免影响集成块的插拔。

图2



印刷板插座，插LASER机主扩展器

制作要点：

焊往6264芯片的导线从集成块上方焊到引脚根部，利用此集成块兼作仿真插头。

图1

的设计中，笔者采用了“零件装配法”，大大节约了内存。其主要思想是：

①把一条助记符形式的指令看作是由几个“零件”组成，比如：MOV A，#DATA 可看作是由 MOV、A 和 #DATA 三个“零件”组成。在内存中设一个“零件表”，其内容是 MCS－51 指令集中用到的所有“零件”，共有约 80 个。

②由一个子程序求出待反汇编机器码对应助记符的各部分在零件表中的序号。MCS－51 指令矩阵有较强的规律性，在此子程序中要尽可能地利用，以缩短程序长度。

③由“装配”子程序按照序号从“零件表”中取出各个“零件”，在装配区中装配起来。

④由“配数”子程序把被反汇编操作码后所带的数据（如果有的话）取出来转换成十六进制 ASC 码写入装配好的指令中的适当地址。

⑤则装配区中的这条指令就可以显示输出了。

2、汇编一条指令：笔者在这部分的设计中利用反汇编功能来产生 MCS－51 指令集，避免了在程序中设置庞大的指令表。

①对数据 0 到 FFH 进行反汇编，逐条反汇编出 MCS－51 的每一条指令与待汇编的指令比较。

②若某一数据反汇编出的指令与待汇编指令完全一样，则这个数就是待汇编指令对应的操作码。

③把待汇编指令中的操作数读取出来（如果有的话）写到操作码后边。

④我们就得到了待汇编指令的完整机器码。

3、标号、伪指令：标号利用 BASIC 的变量系统来存储和查找，这样虽然限制了标号的长度只能是两个字符，却可省去大量的程序开销。

伪指令都用一个“&”作为前缀，后面跟一个伪命令字，如 &.0 代表 ORG；&.E 代表 EQU；&.F 代表 DEFB 等等，这样可以大大方便伪指令的识别与处理。

4、汇编起始地址的处理

各种单片机应用系统的地情况千变万化，而娃娃机中可用的 RAM 又都极其有限，笔者采用“双起始地址”的办法，解决了这一问题，其设计思想是：在源程序开头由用户设定汇编起始地址和虚拟起始地址（即实际运行地址），汇编时，将机器码生成在汇编起始地址开始的内存中，但对程序中的转移类指令，其目标地址都按实际运行地址来处理，这样汇编出的机器码，只要放入单片机中的实际运行地址处，就可正常运行。

5、源程序的处理：利用机器原有的 BASIC 编辑系统来编辑源程序，源程序为带有行号的 BASIC 形式，这样可以充分利用机器原有的编辑、存取磁带及打印等功能。

6、在程序中要尽可能地利用系统 ROM 中的子程序，比如显示输出、键盘输入等等常用功能在 ROM 中一般都有可用的子程序。

下面谈谈接口电路。笔者本着使制作尽可能容易的原则设计了这个程序存储器仿真电路如图 1。将此电路外插在 LASER 机上，将 6264 芯片插在插座中，可为 LASER 机扩展 8KRAM，地址在 E000－FFFF（用户也可通过改变三八译码器的输出端来将这 8KRAM 安排在其它地址上），用户可将程序汇编生成到这段内存中，然后将 6264 芯片从插座上拔下来，插到单片机系统中，就可以将这 8K 存有程序的存储器出借给单片机系统。本电路直接利用存储器芯片本身作为仿真插头，简化了制作工序，电路中所有零件可直接焊在印刷板插座和存储器芯片上，不必使用印刷电路板如图 2。此电路通用性极强，略加改动就可适用于各种常见娃娃机，用户还可以改用 6116 或 6132 等存储器芯片来作这个电路。

91 年全国少年儿童计算机应用竞赛在河北廊房举行

由全国妇联、团中央、中国科协、中国石油天然气总公司管道局、全国少年儿童计算机教育研究会主办的全国少年儿童计算机应用竞赛于 8 月 2 日至 4 日在河北省廊房市中国石油天然气总公司管道局少年宫举行。来自全国各地少年宫的 28 个代表队，140 名小选手分别进行了“五项全能”、“动画制作”和“智慧小车”三个单项竞赛。

五项全能竞赛获一等奖的是：陆靖（男，中福会少年宫一队）；慕巍（男，天津河西区少年宫队）；夏甜（男，管道局青少年宫一队）。

寓言动画竞赛获一等奖的是：张彬（男，中福会少年宫一队）；徐扬（男，大连少年宫队）。

智慧小车竞赛获一等奖的是：杨东林（男，中国儿童少年活动中心）；黄春江（男，佛山青少年宫）。

这次竞赛内容丰富，突出应用和动手能力。竞赛包括了汉字输入、数据处理、计算机硬件系统连接、利用工具软件进行动画的设计与制作，以及将计算机应用在实际的控制系统中等。

这次竞赛较全面地考查了选手们对计算机系统的了解和使用能力并在青少年计算机竞赛的竞赛形式上、计算机软件与硬件的结合上都做了有益的尝试。

（吕　迈）

维修经验谈

TH3070 打印机接口电路工作原理与故障分析

李燕华

一、故障现象

打开电源，通电后打印自检正常，但和联机后不打印，并且使计算机死机。

二、电路分析

由于打印机自检正常，故障可能在和主机相联接的接口电路上。以下首先对接口电路的工作原理做简要分析。

主机和打印机之间应具备数据信息、控制信息、这两部分信息的传输功能，打印机的接口电路正是为了完成这一功能而设置的。就“接口”而言，一般包括控制程序和硬件线路两部分。控制程序，存于计算机中，是主机对打印机的控制，通常有两种方式：查询方式和中断方式。在 IBM PC 机中，打印机的控制方式采用的是查询方式。硬件线路通常也有两种形式，一种是并行接口，一种是串行接口。在一般情况下，TH3070 采用的是 Centronics 标准并行接口。图一给出了 TH3070 并行接口电路图。

从图一可以看出，主机和打印机通过扁平电缆线将信号传输到打印机一侧的 36 线插座 PJ32 上，打印机采用 8 位并行方式接收主机输入的数据（DATA0～DATAF）。控制及相互通信的信号有：忙信号 BUSG、应答信号$\overline{ACK}$、数据选通信号 DATASTB、选中信号 SE-

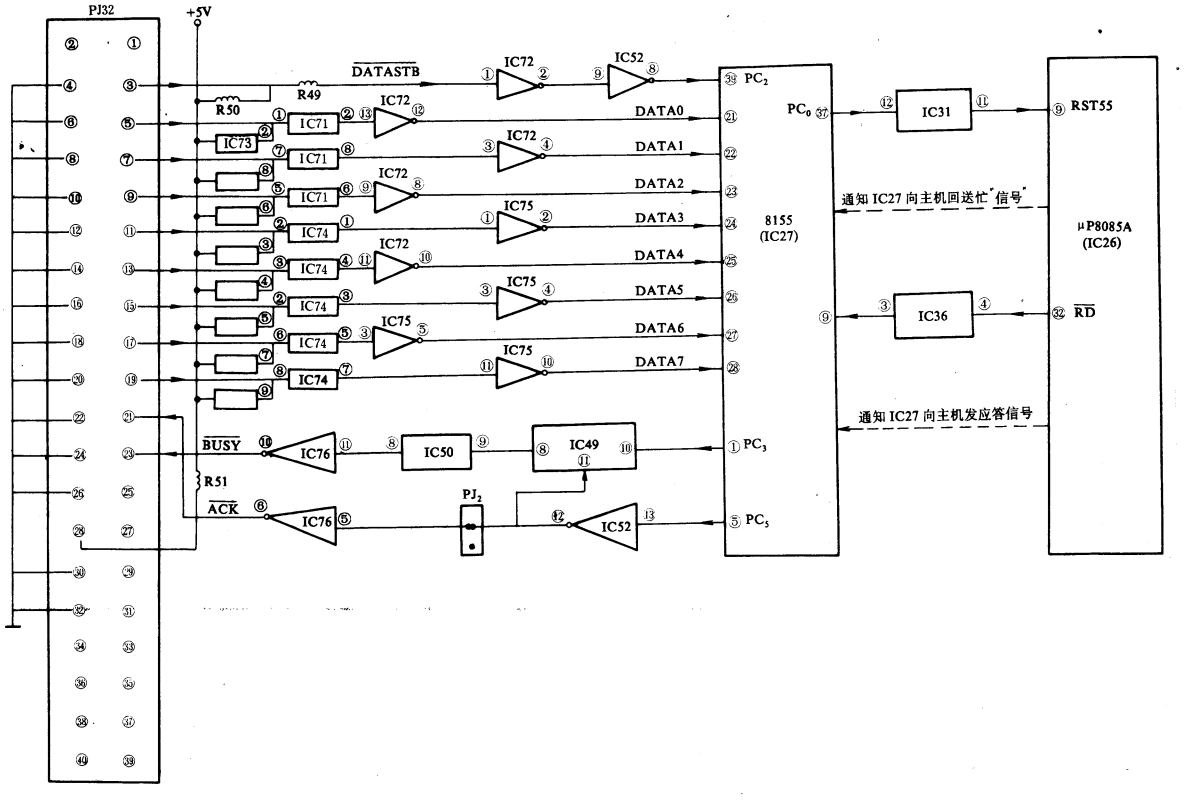


图 1 接口电路

图上表示了接口信号间的时序关系，接口电路的工作过程遵循信号时序的顺序。当打印机处于 BUSY＝即“非忙”状态时，主机将 8 位数据通过 PJ2 的第 5、7、9、11、13、15、17、19 脚送到打印机，经 IC72(74LS14)非门反相后送到接口 I/O 芯片 IC27(8155)的 A 口，接着主机发数据选通信号 DATASTB＝0，该信号 IC72

(74LS14)、IC52(74LS04)两极非门反相后送到 IC2F 的 PC_2。由于数据选通信号的作用，打印机将已到达 IC27A 口的数据打入 IC27，同时 IC27 的 PC_0 端发出 A 口数据选通中断信号，该信号经 IC31(或门)到达 CPU 8085A(IC26)，向 CPU 提出 RST5.5 中断。CPU 接受这一中断后，先屏蔽 RST5.5 中断，使得在此期间数据的处理不受外来信号的干扰。然后 8085A 判断该中断是接口提出的初始化中断，还是数据接收请求中断。由于此时是数据请求中断，因而 CPU8085A 通知 IC27(8155)向主机回送忙信号，此"忙"信号由 8155(IC27)的①脚 PC_3 发出，经 IC49 的⑩脚输入、置 IC49(74LS074)的 $\overline{Q}$ 端(⑧脚)为"0"，然后 IC50、IC76 后，"忙"信号送往主机。至此，CPU8085A 可以对接收的数据做相应的处理。当数据处理完毕，CPU8085A 通知 IC27(8155)向主机发应答$\overline{ACK}$信号，IC27(8155)的 PC_5 ⑤脚发出的应答信号，经 IC52(74LS04)反相后，经 PJ2，一路送往 IC76 再次反相，然后从 PJ32 的㉓脚送往主机；另一路经 PJ_2 送往 IC49(74LS074)的 CLR(时钟)端，将 IC49 的 $\overline{Q}$ 端置"1"，使 BUSY＝0 并送往主机，以便主机可以发送下一行数据。

三、故障的查找与排除

根据以上的故障现象和对电路的分析，我们考虑首先应从 I/O 接口电路的控制信号入手查找。参照图一的电路图和图二接口的时序关系，使用逻辑笔或示波器对"忙"信号(BUSY)、数据选通信号(DATASTB)、应答信号(ACK)等控制信号的通路逐点测量，经测量发现应答信号$\overline{ACK}$在 IC76(74LS06)的⑤脚处有输入，而⑥脚处无输出。IC76(74LS06)是一个六相反相器集成电路，六个输入端对地电阻应相本相同，六个输出端对地电阻应相本相同，用万用表电阻档测输入端、输出端各对地电阻，结果⑤脚对地电阻为＋∞，进一步断定 IC76(74LS06)集成片坏了，更换 IC76，打印机故障被排除，联机后打印一切正常。

对于有些经验的维修人员来说，也可以在使用逻辑笔和示波器之前，用万用表电阻档，对接口电路中传输控制信号的几个反相器集成片的输入对地、输出对地各电阻值进行测量。因为它们较其他电阻性、电容性元器件容易发生故障。这样做的目的是为了更快、更简便地寻到故障的原因。

北京天翔新技术开发公司新近推出

国产化、廉价普及型九针中西文打印机

随着计算机的日趋普及，作为与计算机配套的外围设备——打印机的需求量增加。在打印机领域中，24 针打印机固然有其优秀的特点，与九针打印机相比，其速度及字型等功能是九针机无法比拟的。但价格较高，对一些应用领域难以接受。而九针打印机却以其低廉的价格在中国市场上占据了一席之地。天翔公司为振兴我国的打印机产业，以优质廉价的打印机替代进口产品，在消化、吸收国外先进技术后推出 TX－800⁺、TX－800K 九针打印机。

TX－800⁺中西文打印机是 TX－800 九针中西文打印机的二次开发产品。它提高了打印速度，配置了 IBM 及俄文方式，增加了串行口选件等。作为 TX－800 的增强型，机器的可用性及可靠性均向前迈进了一步，其主要性能均已赶上甚至超过某些进口机型，如 KY－980 等。

TX－800K 是国内现行唯一的中西文九针打印机。选用 TX－800K 与计算机配成系统，可以大幅度提高计算机效率，降低其系统的成本。比如：使用西文打印机的图形方式打印汉字时，计算机必须备有硬汉卡或软字库，同时主机要有较大的内存，且配有相应的驱动程序。而 TX－800K 本身带有硬汉字库及其相应的控制软件，这就降低了计算机系统的成本。此外，每向其它打印机输出一个 16×16 点阵汉字，最少输出 32 个字节，而采用 TX－800K 仅需 2 个字节。提高了计算机的效率。它在软件上采用多种无级演变措施，丰富了可打印字型。

TX－800K 广泛适用于中小过程系统用户，程控交换机用户，小型文字处理系统及个人用户，不要求主机具有硬汉卡或软字库，就可进行汉字处理，这对我国计算机的普及应用，起到了一定的配套作用。目前，在邮政系统的电报、电话计费开据机、工业控制和饭店管理应用中，受到了广泛的欢迎。

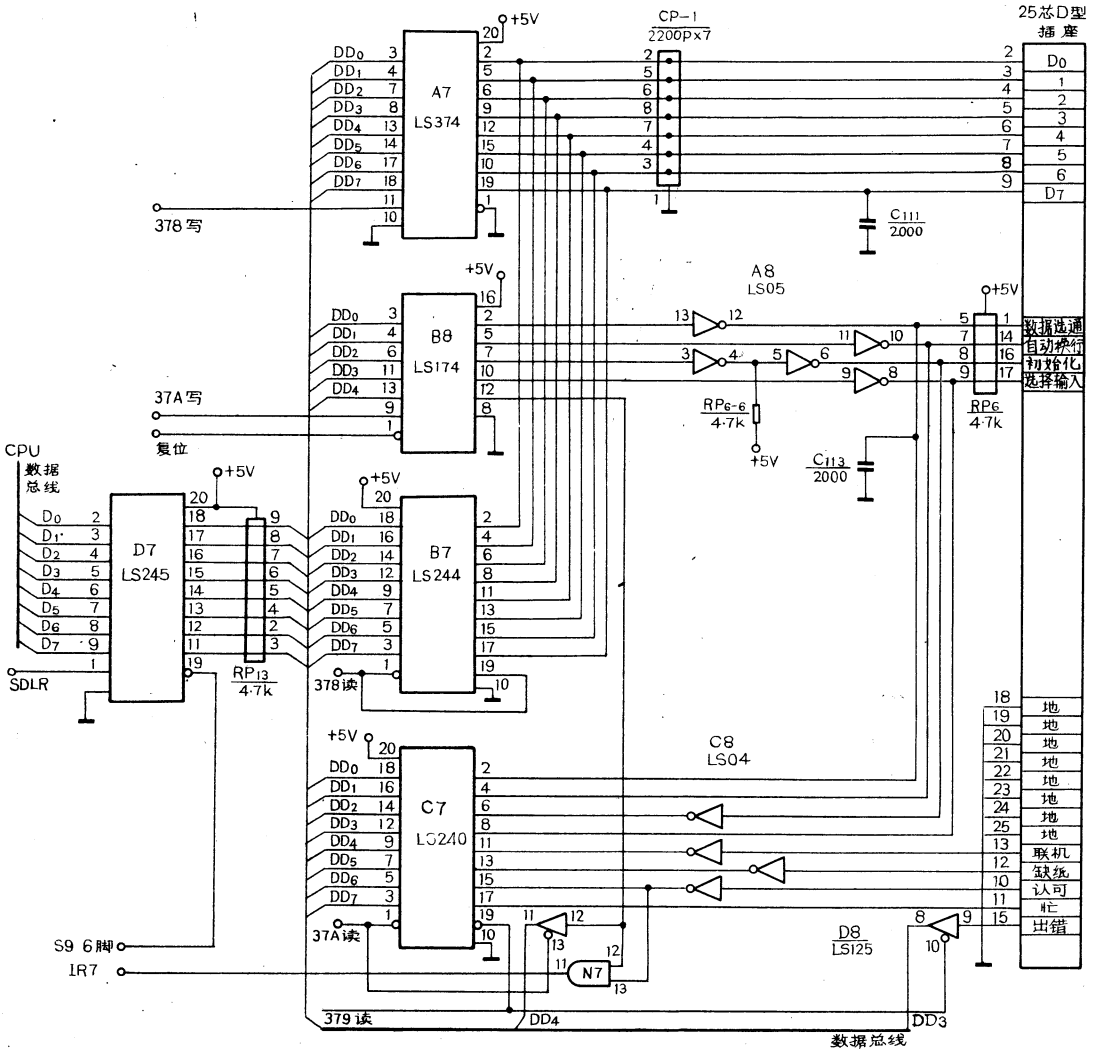
目前，TX－800、TX－800⁺和 TX－800K 的零售价约在 1200～1800 元左右。今后，随着机器产量的增加，售价将会逐渐下降。

长城 0520CH 机打印机适配器故障维修

邱　槟

一、长城 0520CH 机打印机适配器的简要工作原理及电路

为了能够准确地判断故障部位，同时，也考虑到长城 0520CH 机（以下简称 CH 机）硬件资料不多见，我们先来简单了解一下 CH 机打印机适配器的工作原理。其电路原理图如图一所示。它由命令译码器、总线缓冲器、数据锁存器、控制锁存器、控制驱动器和接收发器等组成。考虑到与 IBMPC 机兼容，CH 机打印机适配器的逻辑结构与端口地址与 IBMPC 机都相同，但 CH 机所选器件以及用作译码的地址线又与 IBM PC 机不尽相同。如图二所示。



图一　长城 0520CH 机打印机适配器原理图

CH 机命令译码由 N8 和 E7 两个 74LS138 三——八译码器完成(N8 表示该器件在系统板的第 N 行,第 8 列上,这是位置代码。其它同),分别译出写数据(地址为 378W)、读数据(378R)、读状态(379R)、写控制(37AW)、读控制(37AR)等命令。这些信号分别作用到各锁存器和缓冲器上。

数据收发器 D7/74LS245 的数据传送方向由来自 S9 的 6 脚信号及 C10 的 2 脚的 SDIR 信号控制。当前者信号为低电平时,SDIR 为高电平,则数据由 CPU 系统总线写入适配器,若 SDIR 为低电平,则数据由适配器读入系统总线上。

A7/74LS374 是输出数据锁存器,锁存主机向打印机送出的数据。B7/74LS244 是输入数据缓冲器,输入 CPU 的数据一般就是主机向打印机刚送出的数据,用于故障测试与诊断。

控制锁存器是 D8/74LS174,A8/74LS05 是控制驱动器。控制字的格式为:

7	6	5	4	3	2	1	0
×	×	×	允许中断(+)	选择输入(−)	初始化(+)	自动换行(−)	数据选通(−)

(+表示高电平有效,−表示低电平有效)它们接受主机对打印机的控制信号,并通过 25 芯 D 型接口送到打印机上。

控制/状态信号输入缓冲器由 C7/74LS240、N4/74LS125 及 C8/74LS04 组成。分别接受控制字和状态字的信息送到数据线上,通过 D7/74LS245 送给 CPU。读入的控制字节就是刚输出的控制字节,目的也是用于故障诊断。状态字格式为:

7	6	5	4	3	2	1	0
打印机忙(+)	认可(−)	缺纸(+)	联机(+)	打印机出错(+)	×	×	×

CH 机面板打印指示是由 379R 地址译码线提供的。

二、故障维修二例

下面我们来看两例打印机适配器的故障及维修。

• 故障现象一:打印机与 CH 机的打印接口接好后,进入系统,进行屏幕拷贝或文件打印时,打印机均没有反应,但 CH 机面板 I/O 口的打印机工作指示灯亮。当在总线槽上再插入一块打印卡时,此卡能够正常打印。

• 分析、维修:当进行打印时,虽然打印机无反应,但面板指示正常,这说明打印机适配器的译码正确,控制字及数据校验也正确。插入打印卡后又能正常打印,说明打印机是好的,状态字也能正常读入,那么就可以肯定故障就在 CH 板打印机适配器的数据发送器 A8/74LS05 或排电阻 RP6/47K 上。具体排放前,先用 DEBUG 的 O、I 命令向 378H 地址口送数和读数,用示波器(或三用表)测试 25 芯 D 型接口的第 1、14、16 和 17 脚的电平。经测试发现,无论送什么数,脱机时,25 芯 D 型接口,脚总是低电平,再测 72LS05 的 12 脚对+5V 电阻为 2.2K(正常为 4.7K),对地电阻为 140Ω(其它反向器输出端对地电阻为 560Ω),说明 74LS05 的一个门坏了,换上好的后,故障排除。这里应注意,排电阻一般不易损坏,应先检查 IC 片子,最后检查排电阻。另外,应急修理时,也可利用原来未用的 74LS05 的 1、2 脚的一个反向器,免得从系统板上将坏片子焊下。

• 故障现象二:打印机联机正常,进入系统后,屏幕拷贝和文件打印均不能进行,但打印机是好的,发出打印命令后,面板打印指示灯不亮。另插一个打印卡打印时,现象与原来一样。

• 分析、维修:不能正常打印,同时在打印时面板打印指示灯不亮,从电路上可以看出,故障可能出在译码器 N8 和 E7 的 74L138 上,也可能出在锁存器 74LS174、74LS374 或总线缓冲器 74LS244、74LS240 和 74LS245 上。尤其是再插入打印卡打印时,故障依旧,更说明了故障器件在命令译码控制的几个片子上。为了确定故障部位,还应进一步测试。进入 EDBUG 后,用 O 和 I 命令检查 378H 和 37AH 口地址输出和输入的数据是否一致。经查,脱机时,当向 378H 送 00 时,读入为 05,当送 FF 时,读入的为 FF;当向 37AH 口送 00 时,读入的为 E0,当送 FF 时,读入为 FF。由于 37AH 口送出和读入的是控制字,由其格式可知,其高三位未用,所以送 00,读入为 E0 是正确的。而 378H 口的数据就有问题了,从测试结果看,74LS374 的 DD_0 和 DD_2 锁存有问题,或者是 74LS244 的 DD_0 和 DD_2 读入有问题。将 25 芯 D 型接口的 2 脚和 4 脚接地,再向 378H 口送 00,读入为 00,这说明 74LS244 是好的,将 74LS374 换掉后,故障排除。值得注意的是,由于 CH 机的打印机适配器和总线座上的打印卡的端口地址一样,当出现故障时,将会互相影响。

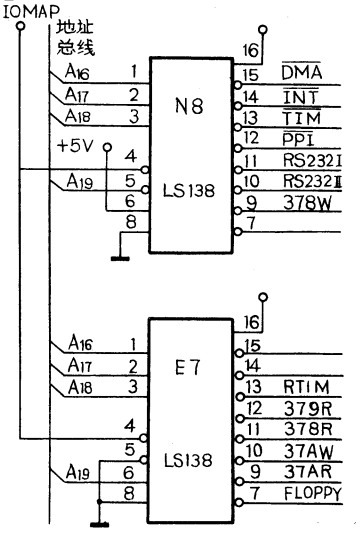


图 2 命令译码器

初级程序员级水平考试辅导
汉字 dBASE Ⅲ 使用技巧(下)

王路敬

54. dBASE Ⅲ 应用程序设计值得提倡的方法是什么?操作的基本步骤有哪些?

利用 dBASE Ⅲ 设计一个应用程序时,对于初学程序设计的人来说,感到困难的是缺乏程序设计的方法。而设计一个应用程序采用什么方法又往往是相当重要,如果方法不当,会导致事倍功半的结果。dBASE Ⅲ 提供了一套先进的结构式应用开发语言,它不但语句丰富、自成体系,具有灵活的人机会话功能,而且它的语言体现了结构式程序设计的特征。用 dBASE Ⅲ 语言编写的程序是一个结构式的程序。通常一个 dBASE Ⅲ 的实用程序由一个主控程序和若干个子程序组成的。一个稍微复杂的应用管理系统中有许多大小不同的程序,这里就存在从何处入手的问题。传统的方法是先从下层入手,也就是说先实现下层程序(模块),因为下层某一程序功能比较具体,可以实现一部分就调试一部分。但是这种方法包含着很大的危险性,因为它不是从整个系统的功能出发。使用这种从下往上的方法,很可能出现各个模块单独调试运行都可通过,但是系统联调试时却产生通不过的现象,其主要毛病产生于整体结构与各程序块间的接口处。

dBASE Ⅲ 实用程序的设计值得提倡的基本方法是"从上到下,先粗后细,化整为零,分而治之"。这种方法的基本思路是:

(1)先实现上层,后实现下层。具体来说就是首先调试整个系统的结构及各个程序之间的接口,在确保系统结构及各模块间接口正确无误之后,再实现下层程序的功能。

(2)在实现上层模块时,下层模块只保留其名字及输入参数和输出参数。而模块本身的处理先空着,或者只保留一两处象征性的语句或简单的显示信息。这样,上层模块可以调用下层模块,虽然其具体处理功能还未实现,可以验证调用是否成功,输入输出的传输是否正确等;这样就可以调试系统结构,保证其正确性。

(3)化整为零就是先部分后全部。即把整个系统分成若干"版本",先实现一个只反映系统骨架的版本,而后不断扩充,最后实现系统全部功能。

采用从上到下的方法既能有效地解决接口问题,又便于尽早发现系统结构整个上存在的问题,从而收到理想的效果。

实现上述方法的操作可分以下 7 步进行:

①明确管理系统应具有的功能,按这些功能划分程序模块,画出系统总体框图。

②要管理的信息由哪几种数据类型组成。字段多少,长度怎样,设计确定数据库的结构,并输入一定数量的数据。

③根据程序的功能画出程序的细框图。每个功能子程序的框图描述应尽量清晰详细,把每一步要做什么都在框图上标明。

④若需输出报表,则要设计报表格式。若屏幕输入数据,则设计屏幕格式文件。

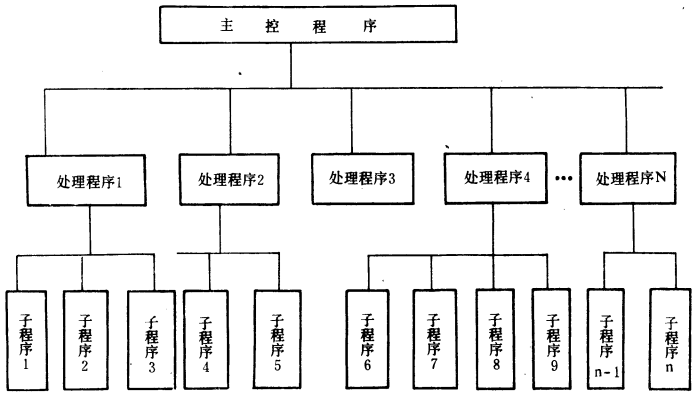
⑤检查程序框图没有错误,便可动手分模块编写程序。若几个人共同编程序,要按人分模块,完成功能比较完整的模块,不要将一个功能模块分给几个人来写,以免引起混乱。

⑥程序编写完以后,先分模块进行调试。调试的目的是检查该模块功能上是否达到了预定的要求,并排除错误。当各模块都通过分调之后,再联起来进行统试。

⑦统试通过后试运行,试运行无误后便可投入正常使用。

55. dBASE Ⅲ 菜单程序的结构特点是什么?

在进行 dBASE Ⅲ 实用程序设计时,菜单方式是经常使用的,因为实用程序主要是为了使用,而用户很多是不懂计算机的,不可能要求他们都会打入 dBASE Ⅲ 命令,如果利用菜单方式,即将程序的功能全部在屏幕上显示,并将每一功能对应一个数字或字母,使用户象点菜一样,按某一键即可完成某一功能。菜单分大菜单和小菜单,大菜单即为主模块(或主控模块);小菜单即子程序(子模块)。菜单程序的一般结构框图如图所示:



在一个比较完善的微机管理系统中，往往包含了数十个，甚至上百个菜单程序，因此，菜单程序如何设计，对系统的功能起着举足轻重的作用。众所周知，dBASE Ⅲ命令功能很强，往往一条命令就可以独立完成一类复杂的操作，例如数据处理中常用的分类、排序、检索、显示、打印报表等操作，都可以由一条命令完成，如果再加上范围、条件、函数等短语，其功能就更强了。因此，操作使用人员几乎可以不必考虑操作功能如何实现，而把主要精力放在菜单程序设计上，只要把一个个菜单设计好，整个程序设计的问题也就迎刃而解了。

dBASE Ⅲ的菜单程序通常有如下特点：

(1)多分支选择结构。“菜单”说到底是一个多分支程序，即程序执行到此先暂停，等待使用者决定下边执行哪个分支程序，而后继续执行。dBASE Ⅲ的条件转移命令有两个，IF 和 CASE。IF 命令要求 ENDIF 与 IF 成对使用，一般在分支较少时可以使用，而在分支较多时，使用 IF 远不如使用 CASE 编程简洁，在菜单程序中，大多使用 CASE 命令。

(2)单入口单出口结构。尽管菜单程序是多分支的，但其入口和出口是唯一的，它是由单支发散为多支，再由多支汇总为单支，程序在 DO CASE 后发散，在 ENDCASE 时汇总。通常把“一道菜单”程序编为一个独立的“子命令文件”，需要时调用，执行某分支后由唯一的 RETURN 返回主程序。

(3)采取了“容错”设施。实践表明，在微机系统运行中，绝大多数错误来自于操作失误。为了使微机系统在操作使用人员对“菜单”提问回答错误时仍能继续运行，并给操作人员以错误提示信息和改正错误的机会，而不至“死锁”或停止运行。在菜单程序中，一般都含有循环程序，在选择错误时，用 LOOP 命令返回程序开关，再显示该“菜单”，要求操作人员重新选择。

56. 利用 TEXT…END TEXT 设计菜单的优点是什么？

在 dBASE Ⅲ中提供了一组很简单的菜单设计命令 TEXT…END TEXT，该组命令的功能是显示命令文件的一个文本数据块，此命令用于屏幕菜单提示设计既方便又简单。使用时 TEXT 与 ENDTEXT 必须成对出现，例如主控命令文件中的菜单设计：

```
SET TALK OFF
CLEAR
TEXT
        1. 显示      2. 查找
           3. 增、删记录
        4. 打印报表   5、退出
ENDTEXT
⋮
⋮
RETURN
```

如果在 TEXT 之前加上 SET PRINT ON 语句，还可以输出菜单到打印机。

57. 主菜单程序应包括哪些基本内容

主菜单程序又称主控模块，它是管理系统的控制核心，它要调用和运行子程序，要为整个程序建立工作环境，在返回圆点提示符之前还要关闭各类文件。管理系统运行时，首先进入主菜单程序，主菜单程序应显示出系统所具备的各项功能名称，给出必要的提示信息。每一种功能子程序有其自己的编号和名称，操作者根据自己的需要键入相应的功能编号，使其控制转向功能号所对应的子程序，执行完后返回主菜单程序。

主菜单程序应该包括下面的基本内容：

(1)设置工作环境

首先要用 CLEAR 获得一个清晰的初始状态，再逐个设置所需要的 SET 状态。例如，控制台响铃，命令的对话状态，目录路径，小数位数及功能键的设置等。

(2)初始化内存变量

要定义整个程序中使用的内存变量，给出它们的初始值。

(3)设计主菜单

为了保证单入口单出口的程序结构，一般都采用循环来设计主控程序。在主控程序中通常必须有如下四种类型的语句：

①DO WHILE〈条件〉语句。一开始应确保条件为“真”，以保证执行主控程序时能进入主菜单，通常选用 DO WHILE ·T· 的形式。而在“退出”选择项时使用 EXIT 退出主控程序。

②?，TEXT…ENDTEXT 或@语句。用以显示主控菜单的各个选择项。

③人机交互性语句。例如，ACCEPT，INPUT，WAIT 等。用以接受用户键入的选择。若各选择项的编号为数字，则使用 INPUT 较为方便；若为字母，则使用 ACCEPT。

④DO CASE 或 IF 语句。用以根据不同的选择调用不同的子模块。在选择项较多时，使用 DO CASE 语句最为方便。

(4)程序结束前的内务处理

在返回圆点提示符之前，应恢复 SET 设置的标准默认状态，关闭各类文件。注意，RETURN 语句并不关闭文件。除此之外还应释放全局性的内存变量。

58. 主菜单程序编制的常用方法有哪几种？

编制菜单的方法有很多种，常用的一种方法是利用“?”和“CASE”语句相结合。用？在屏幕上显示各种功能及其对应选择键，然后用 CASE 语句来分别执行对应功能。

例如一个简单的工资管理程序的主菜单程序如下：

```
CLEAR
DO WHILE ·T·
? “××单位工资管理系统”
? “1……追加、插入、编辑”
? “2……查询”
? “3……删除”
? “4……显示”
? “5……打印”
? “0……退出”
? “置入您需要的选择”
WAIT TO SELECT
DO CASE
    CASE SELECT=“1”
    ……
    CASE SELECT=“2”
    ……
    ……
    CASE SELECT=“0”
    EXIT
ENDCASE
ENDDO
```

执行该程序时，用户按某一数字键 CASE 语句就执行某一数字所对应的功能，当按入“0”时即退出主菜单程序。

第二种是利用@命令、READ 命令与 CASE 命令配合，其优点是使屏幕上显示的格式美观。例如下面这个主菜单程序：

```
CLEAR
STORE "  " TO OP
DO WHILE OP<>"N" .AND. OP <> "n"
CLEAR
@0,27 SAY "(管理系统名称)"
@1,27 SAY "* * * * * * *"
@2,4 SAY "          "
@3,4 SAY "A……XXXXX"
@4,4 SAY "B……YYYYY"
……
@PROW()+1,4 SAY "N……结束"
@PROW()+1,4 "请按键进行选择" GET OP
READ
DO CASE
    CASE OP="N" .OR. OP="n"
    CLEAR
? "欢迎您再次使用本系统"
EXIT
    CASE OP="A"
……
CASE OP="B"
……
……
ENDCASE
ENDDO
```

第三种方法是利用 TEXT……ENDTEXT 与 DO CASE 语句相结合，用 TEXT……ENDTEXT 在屏幕上显示主菜单，用 DO CASE 语句选择某一功能。下面这个实用程序的主菜单就是这样设计的。

```
* 主菜单程序 CONTROL. PRG
SET DEVICE TO SCREEN
SET PRINT OFF
SET TALK OFF
DO A;FIRST
DO A;SECOND
DO WHILE .T.
CLEAR
? "      为  您  服  务      "
? "      ==========      "
WAIT "敲任一键开始"
TEXT
CLEAR
    * * * * * * * * * * * * * * * * *
    *                               *
    *        功能如下：             *
    *  1----人事档案材料的编辑      *
    *  2----人事档案材料的显示      *
    *  3----人事档案材料的查询      *
    *  4----人事档案材料的统计      *
    *  5----人事档案材料的报表输出  *
    *  0----退出                    *
    *                               *
    * * * * * * * * * * * * * * * * *
ENDTEXT
INPUT "请按 0—5 选择功能"TO TT
DO CASE
    CASE TT=1
        DO A;BJ
    CASE TT=2
        DO A;XS
    CASE TT=3
        DO A;CX
    CASE TT=4
        DO A;DJ
    CASE TT=5
        DO A;DY
    CASE TT=0
        DO A;LAST
ENDCASE
ENDDO
```

从上面几个实例我们不难看出在 dBASE Ⅲ 应用程序中通过菜单这种人机接口让用户选择执行各种功能，给使用带来了很大的方便，使得操作简单而又直观，同时给阅读程序和分析程序带来很多好处，是值得提倡的。

59. 使用分支语句进行分支结构程序设计时应注意哪些问题？

dBASE Ⅲ 提供了三种典型的分支语句：

简单分支语句：IF——ENDIF；

选择分支语句：IF——ELSE——ENDIF；

结构分支语句：DO CASE——ENDCASE。

使用这些分支语句进行结构程序设计时，以下几个问题不可忽视：

(1)分支语句是一种结构化程序，其入口语句例如：IF，DO CASE 与其相对应的出口语句，例如 ENDIF，ENDCASE 一定要配对出现。

(2)作为出口语句，例如 ENDIF 和 ENDCASE，在其语句后可增加一些文字性的注释内容，目的是提高应用程序的可读性。这些注释内容对程序的运行不会产生任何影响。

(3)DO CASE 与第一个 CASE 项之间不能有任何可执行的命令，系统永远不执行这个位置上的语句。

(4)三种分支结构，彼此都可以作为结构中待执行的语句行序列，组成嵌套形式，但嵌套的层次要清晰，不能交叉。否则造成逻辑混乱。

(5)为了防止仅有入口语句没有出口语句，或嵌套混乱情况发生，使程序的结构层次清晰，减少编写错误，增加程序的可读性，编写结构程序时，在书写上最好采用“锯齿”形的写法。即将结构的入口和出口语句写在同一纵座标的位置上，而嵌套在里面的其他结构语句的入口和出口向右移若干空格，写在另一同纵座标的位置上等等。这样一层一层的结构十分清晰，给查错也带来很大的方便。

60 在使用循环语句设计循环程序时应注意哪些问题？

在 dBASE Ⅲ 应用程序设计中常常需要重复执行某些相同的操作，即对一段程序进行循环操作。DO WHILE——ENDDO 循环语句保证了这一功能的实现。但在使用循环语句设计循环程序时需要注意以下四点：

(1)循环入口语句 DO WHILE 与出口语句 ENDDO 必须配对使用。

(2)在出口语句 ENDDO 之后可附加文字性的说明内容，

以便增加程序的可读性。其说明内容不影响程序的执行。

(3)循环程序中可以包含循环程序,即循环程序可以嵌套。循环嵌套层次不限,但内层循环必须完全嵌套在外层循环之中,不允许出交叉循环,否则,造成逻辑上的混乱,这是不允许的。

(4)循环语句和条件分支语句可以一起联用,同样允许嵌套,即在循环结构语句中可以包含条件分支语句,反之亦然。但是不允许交叉。为防止这类错误出现,在编写程序时亦可采用"锯齿"法书写(参看上个问答的有关部分)。

61. 为什么要使用过程文件?过程文件中的过程如何调用?

在一般调用中,过程是当作一个文件独立存放在磁盘上的,所以每调用一次过程,都要打开一个磁盘文件,而每和磁盘打一次交道都是需要一定时间的。这样一来,当频繁调用过程(子程序)时,信息管理系统的运行就会走走停停,影响系统的运行速度。从减少磁盘访问时间出发,可以把多个常用的过程合并成一个大的程序文件,将这个大的程序文件一次打开,之后可以多次调用其中包含的过程。在这个大程序文件中,每个过程仍然是独立的,这个大的公用程序被称之过程文件。一个过程文件可以容纳 32 个过程。过程文件被打开后,(dBASE Ⅲ 系统就能找到该文件所容纳的各个过程的所在位置,不需要频繁地进行磁盘操作,从而大大提高了调用过程的速度。同时一个过程文件由许多过程组成,既便于编写易于调试运行,也大大缓解了允许系统打开文件个数的要求。

在过程文件被打开前,其所包含的过程是不能被任何程序调用的。因此,在过程文件中进行过程的调用,在调用前,过程文件必须预先用打开过程文件的语句:

SET PROCEDURE TO〈过程文件名〉

打开。

过程文件被打开后,它所包括的过程便可为其他程序调用,调用时仍使用 DO 语句,只不过这时在 DO 语句中使用的是由 PROCEDURE 所指明的过程名。值得注意的是,过程文件名和过程名是两个不同的概念。过程文件仍是 .PRG 文件,过程名可由 1—8 个字符组成,但没有扩展名,每个过程以语句 PROCEDURE〈过程名〉开始,以 RETURN 语句终止。

62. 设计屏幕格式文件应注意些什么问题?

所谓屏幕格式文件实际上是编写若干条@—SAY—GET 语句所组成的一种文件。dBASE Ⅲ 执行时便在屏幕上显示用户自己满意的画面格式,供操作者输入和修改编辑数据。这种文件的扩展名为 .FMT,设计该类型的文件应注意以下几点:

(1)屏幕格式文件是文本文件,可以在 dBASE Ⅲ 的 MODIFY COMMAND 字处理方式下建立,也可以在其他编辑程序下建立,但文件的扩展名 .FMT 必须在命令格式中明确的给出,以标明建立的是屏幕格式文件。否则 dBASE Ⅲ 将自动地认为是命令文件,并赋给文件的扩展名为 .PRG。

(2)在调用屏幕格式文件之前,必须用 SET FORMAT TO〈格式文件名〉将格式文件打开。(可不带扩展名)

(3)屏幕格式文件只能由两种语句组成,其一是注释语句 *,用来对格式文件的功能及所使用的语句加以说明;其二是@—SAY—GET 语句,用来设计用户屏幕格式。

(4)调用屏幕格式文件只有下述五条命令:APPEND,INSERT,EDIT,CHANGE 和 READ 命令。但要注意,一个 READ 语句能关联的 GET 变量,在 dBASE Ⅲ 中最多为 128 个。

(5)屏幕格式文件用完之后,如果不再需要应该使用 CLOSE FORMAT 语句及时的关闭。

63. 在进行信息查询时,被查询的内容确定存在,但有时就是查不到,为什么?

出现这种情况往往是由于当前盘和系统定义的默认盘不一致而造成的。每当汉字 dBASE Ⅲ 被启动之后,汉字 dBASE Ⅲ 即自动地在其所在盘上寻找是否有 CONFIG.DB,若有这个文件,则首先执行 CONFIG.DB 所含的各条命令,执行完,进入汉字 dBASE Ⅲ 命令状态;如果没有,便立即进入汉字 dBASE Ⅲ 命令状态。若被查询的内容确实存在,但又查找不到,解决的方法是将当前盘和系统默认盘取得一致,或者被查询的文件放在默认盘上,就不会发生查找不到的现象。

64. 什么叫模糊查询,怎样实现?

模糊查询是相对明确查询来说的。一般说来,明确查询要求查询人员提出明确的查询条件,而且被查到的信息要和查询人员提出的查询条件一字不差地完全相同,而模糊查询则要求查询人员提出的查询条件可以是不完整的,零碎的,只字片语的,而被查到的信息也不要求和被查询的条件完全一致,只要包含符合查询条件的内容,满足用户要求就行了。

模糊查询主要是用字符串包含运算符 $ 实现的。

例一,用 LOCATE 进行模糊查询:

- USE〈库文件名〉↙
- LOCATE ALL FOR "高"$ 职称↙
- DISP↙
- COUNTINUE↙
- DISP↙

一个信息管理系统的设计,如何做到既能实现明确查询,也能进行模糊查询,这无疑是一个十分重要的问题。

例 2,用 LIST、DISPLAY 进行模糊查询

- USE〈库文件名〉↙
- LIST ALL FOR "王"$ 姓名

例 3,用 FIND 和 SEEK 进行快速模糊查询

这时需要建立多个按某个字段中的各个单字为索引关键字的索引文件。

例如,按姓名字段第二个汉字为"秀"的单字查询:

- INDEX ON SUBSTR (姓名,3,3) TO XMS↙
- FIND 秀↙
- DISP↙

65. 提高程序执行速度可采取哪些措施?

对于 dBASE Ⅲ 的大多数应用软件,影响其速度的主要是查询检索操作,而不是访问磁盘。提高程序执行速度的措施可以从以下几个方面着手。

(1)利用 dBASE Ⅲ 建索引文件和用程序段代替命令的方法。

一般来说,用单条命令处理时,在单位时间内所处理的记录数要比用程序所处理的多。但在实际工作中,大多数情况并不是对库中连续的记录进行一定的操作,而只是对库中满足条件的记录进行操作。此时,如果用单一的命令来完成一定的操作,则相当一部分时间花在对无关记录的判断上,而不是进行指定的操作,从而影响了整个程序执行的速度。如果充分利用 dBASE 索引文件的特性,同时用程序段来代替相应的命令,则可以大大地缩短完成一定的操作所需的时间。条件越复杂,其效果越明显。

众所周知,索引文件具有将关键字段相同的记录排在一个连续的区域中的特性,因此,根据不同的目的,对相应的关

键字段进行索引，就可将零乱的数据库按要求整理的很有规律。利用建索引文件，同时用具有特定功能的程序段代替具有相同功能的单一命令，就可实现在最短时间内完成某项具体的操作。

例如有一个 1000 个记录的人事档案数据库 JTGL，分为 20 个部门，要求按库 BM 的部门顺序分部门将每个人的基本情况打印输出。可采用两种方法：

其一，直接在基本上无规律排列的原始数据库

JTGL 上操作（见程序段 PROG1. PRG）。这种操作过程是对某一个部门的所有记录即将整个数据库的 1000 个记录从头到尾扫描一遍，符合条件的操作，不符合条件的跳过。对 20 个部门的所有记录进行扫描，就要将数据库从头到尾过 20 遍，因而大量的时间花在对无关记录的判断上。请看 PROG1. PRG：

```
CLEAR
SET TALK OFF
SET PRINT ON
SELECT A
USE JTGL
SELECT B
USE BM
DO WHILE · NOT · EOF()
    BUMEN=部门
    SELECT A
DISPLAY FOR 部门=BUMEN
SELECT B
SKIP
ENDDO
```

其二，首先对数据库在部门上进行索引，然后在索引数据库上进行操作，操作哪个部门，只对索引文件上相应区段内的记录过一遍，因此，对 20 个部门的所有记录按要求操作一遍，只相当于对整个数据库从头到尾过一遍（见程序段 PROG2. PRG）。

PROG2. PRG 清单如下：

```
CLEAR
SET TALK OFF
SET PRINT ON
USE JTGL
INDEX ON 部门 TO BUMEN
USE JTGX INDEX BUMEN
SELECT B
USE BM
DO WHILE · NOT · EOF<>
BUMEN=部门
SELECT A
FIND &. BUMEN
DO WHILE · NOT · EOF(  ) · AND · 部门=BUMEN
    DISPLAY
    SKIP
ENDDO
    SELECT B
    SKIP
ENDDO
```

这两种方法的功能完全相同，只是 PROG2. PRG 在利用索引文件的同时用了程序段：

```
FIND & BUMEN
DO WHILE · NOT · EOF(  ) · AND · 部门=BUMEN
DISPLAY
SKIP
ENDDO
```

代替了 PROG1. PRG 中的单一命令

```
DISPLAY FOR 部门=BUMEN
```

实际运行结果表明，完成同样的操作 PROG2. PRG 比 PROG1. PRG 的速度快约 6～7 倍。

(2)利用 COPY 命令生成临时库的方法。

利用 COPY TO〈文件名〉FOR〈表达式〉命令，把要查询或统计范围内的记录放到一个临时库文件名下，然后再对该文件进行查询或检索。若使临时库文件不占用磁盘空间，一旦完成指定操作后可用 ERASE 命令将其删除。操作时所用的程序段可以这样编制：

```
⋮
SELECT A
USE〈原始数据库名〉
COPY TO〈临时库名〉FOR〈表达式〉
SELECT B
USE〈临时库名〉
DISPLAY
ERASE〈临时库名〉· DBF
⋮
```

(3)巧用 PACK 命令。

在删除数据库的记录时一般分为二步：第一步先对要删除的记录作删除标记“*”，即执行 DELECT〔范围〕〔FOR〈条件〉〕命令；第二步执行 PACK 命令，对作删除标记的记录真正的从数据库中删除。若删除的记录不连续，不要逻辑删除一条或几条就执行 PCAK 命令，而是对要删除的所有记录都选进行逻辑删除，最后一次使用 PACK 命令将凡是作逻辑删除的记录都删除掉。这样要比逻辑删除一条，就执行一次 PACK 命令要节省时间。因为 PACK 命令又称打包命令，或紧缩命令，多次打包所费时间总是比一次打包要多得多，尤其是数据库文件中记录总数相当多的情况下就更明显。

如果要删除一个数据库的全部记录，不必用 DELECT ALL 命令，而最好用 COPY TO〈文件名〉STRU 命令，这样可大大提高删除的速度。

(4)控制打开文件的个数

在使用 dBASE Ⅲ 后，人们往往会发现，虽然它能够同时打开多个文件（最多 15 个），但当打开一定数量的数据文件后，程序运行的速度却降低了，特别是在子程序嵌套层次多的情况下尤为明显。解决的办法一是将数据缓冲区个数开辟到最低限度，也就是需要多少就开辟多少。具体做法是改写 CONFIG. DB 文件中的 BUFFERS 数值。因为，开辟大的缓冲区使得 dBASE Ⅲ 能保留更多的信息，因而运行速度快。但缓冲区比文件要占更多的空间。另一个办法是尽量减少子程序的嵌套次数。具体做法是编写命令文件尽量长一些，以利于减少文件个数，减少程序嵌套次数。但是也应当注意，程序也不能太大，太大了一次调入的时间增加，反过来也会影响运行的速度。

(5)消除存储碎片

存储碎片是由于文件分散在盘上不连续的块中而产生的。读写这些碎片中的数据要花更多的时间。消除碎片的方法是用 COPY 命令将其文件先拷贝到一块新盘中，然后再拷

回到原盘中。对于硬盘，可以使用 BACKUP 和 RESTORE 命令来达到连接文件物理位置的目的。

(6)灵活使用 RUN

RUN 命令用来运行外部命令或其他可执行文件。运行外部命令比执行 dBASE Ⅲ 命令快，可利用这一特点来提高执行速度。例如，在 dBASE Ⅲ 下把 ABC·DBF 复制到 XYZ·DBF 中，可以使用二种方法：

其一：·USE ABC

·COPY TO XYZ

其二：·RUN COPY ABC·DBF XYZ·DBF

这里的 COPY 是 DOS 文件复制命令，这种方便的速度远远高于其一所用的方法的速度。

(7)善于使用过程文件

应用系统往往由多个文件组成。dBASE Ⅲ 一次只调用一个命令文件。调用命令文件，系统必须到磁盘上寻找一次文件目录，再将所需的文件调入内存。随着访问磁盘次数增加，系统的运行速度就降低。使用 dBASE Ⅲ 过程文件可以减少访盘次数。程序在打开过程文件时就将该文件所包含的过程一同调入内存，在调用每一个过程时就不需进行磁盘操作，因此，可大大提高系统的运行速度。

过程文件的组成形式如下：

```
PROCEDURE 〈过程名〉
  ⋮
RETURN
PROCEDURE 〈过程名 2〉
  ⋮
RETURN
  ⋮
```

在用之前，用 PROCEDURE TO 〈过程文件名〉打开过程文件，调用结束后，用 CLOSE FROCEDURE 命令关闭过程文件。

一个过程文件，其中可以有多达 32 个过程。过程文件一旦被打开，各个子程序在盘上的位置就知道了，从而减少每次访盘再去查目录占用的时间。

(8)REPLACE 命令与 SUM 命令使用技巧

在实际应用中尽可能用一个 REPLACE 命令实现对多个字段的修改，比多次用 REPLACE 命令每次只对一个字段修改要快得多。例如，用

```
REPLACE〈字段 1〉WITH 〈表达式 1〉,〈字段 2〉
WITH〈表达式 2〉,…〈字段 n〉WITH 〈表达式 n〉
```

比用：

```
REPLACE 〈字段 1〉WITH 〈表达式 1〉
REPLACE 〈字段 2〉WITH 〈表达式 2〉
  ⋮
REPLACE 〈字段 n〉WITH 〈表达式 n〉
```

的速度快得多。

在统计求和时，用：

SUM 〈字段 1〉,〈字段 2〉,〈字段 3〉TO Y1,Y2,Y3 FOR 〈条件〉时，可用

```
Y1=0
Y2=0
Y3=0
DO WHILE 〈条件〉
Y1=Y1+字段 1
Y2=Y2+字段 2
Y3=Y3+字段 3
END DO
```

代替，其速度也能加快。

(9)利用屏幕格式文件

利用屏幕格式文件可以定义语句输入格式。所谓屏幕格式文件就是在一个以·FMT 为扩展名的文件中，利用格式化输入语句在屏幕的适当位置上接收数据。这样可以避免在用 APPEND·EDIT 命令输入、修改数据时，以 dBASE Ⅲ 提供的固定屏幕格式接收数据，这样输入数据更为方便，调用屏幕格式文件，可以提高速度，且可以提高应用程序的质量。

使用屏幕格式文件时，用 SET FORMAT TO〈格式文件名〉命令打开，用完后用 CLOSE FORMAT 命令关闭。

(10)循环语句中 LOOP 命令的使用

在 DO WHILE……ENDDO 循环结构语句中使用 LOOP，使程序短路的命令跳过循环体中 LOOP 之后的命令，比不用 LOOP 命令而用 IF 等命令跳过其后的命令，程序的执行速度快。这是因为在使用 IF 命令时，不管 IF(或 ELSE)后的命令是否要执行，都要调入内存解释一次，而 LOOP 命令就不需要这样。

66. 用 dBASE Ⅲ 编制报表需要解决哪几个问题？

(1)走纸问题

换行或走纸是 dBASE Ⅲ 制表中较头痛的问题之一。当前流行的各种 dBASE Ⅲ 版本中，用@命令制表时，都存在打印机不能连续打印超过 255 行的现象。若超过 255 行即会出现死机或打印机空走纸。虽然 dBASE Ⅲ 提供了一个换页命令 EJECT，但若使用该命令换页时，打印机走纸并不是一页而是两页。解决这一问题可以采用以下的方法：

在用格式输出命令制表时，打印机通过当前行函数 FROW()确认纵坐标。此函数的允许值是 0～255。一般此函数的值是连续累加的，当超过 255 时就会使打印机空走纸。可以通过以下的命令使该函数的值为零：

@〈行坐标，列坐标〉SAY CHR(27)+CHR(12)

利用上述命令换页，避开 EJECT 命令，可解决换页和空走纸的问题。参看下列小程序段：

```
SET DEVICE TO PRINT
SET PRINT ON
@PROW(),0 SAY CHR(27)+          CHR(12)
                                ;给当前行函数置零
DO WHILE ·T·                    ;假定打印若干内容
@PROW()+1,PCOL()+1 SAY          "打印内容"
IF PROW()>=66                   ;规定一页打印 66 行
@66,0 SAY CHR(27)+CHR(12)       ;将打印头移至次页
                                第一行并将当前行函
                                数 PROW()置零
ELSE
ENDIF
ENDDO
SET DEVICE TO SCREEN
```

(2)变换字型

通过 dBASE Ⅲ 提供的函数 CHR()向打印机发送控制码可以改变汉字字型。不同的打印机改变字型的控制码可能不同，即使相同的打印机使用不同的汉字驱动程序，控制码也可能不同。字型变换的种类决定于所使用的驱动程序。TH—3070 打印机在 3070C·EXE 驱动程序下，可以打印 A、B、C、D、E、F、G、H 和 P 九种字型，而使用 3. COM 驱动程序则只打

印 A～H 8 种字型，而在 CCDOS 4.0 支持下的 TH3070E.EXE 则可打印 A～N 和 a～n 共 28 种字型。所以在进行字型变换操作时，一定要注意汉字驱动程序要求：驱动程序要与打印机型号配套；驱动程序要与所要求的汉字库配套；驱动程序要与使用的汉字操作系统配套。在向打印机发送控制码前，要先接通打印机。

①TH—3070 打印机，3070C・EXE 驱动程序

改变字体的方法：

```
SET PRINT ON
@〈行，列〉 SAY CHR(27)+"I"+"字型号"
```

例如，选择 B 型字体：

```
? CHR(27)+"I"+"字型号"
或
SET DEVICE TO PRINT
SET PRINT ON
SET PRINT ON
? CHR(27)+"I"+"B"
或
SET DEVICE TO PRINT
SET PRINT ON
@〈行，列〉SAY CHR(27)+"I"+"B"
```

②M2024 打印机，P2024・COM 驱动程序

改变汉字字型的方法：

```
SET PRINT ON
? CHR(27)+"I"+"字型号"
SET PRINT OFF
```

例如，选择 D 型字体：

```
SET PRINT ON
? CHR(27)+"I"+"D"
SET PRINT OFF
```

(3)在一行中打印不同字型的问题

当需要在同一行中打印不同的字型时，可以采用格式命令输出，也可以采用非格式命令输出，还可以采用格式命令和非格式命令混合输出的方法。下面通过具体程序实例来说明在不同情况下的操作。

①采用非格式输出命令

```
SET CEVICE TO PRINT
SET PRINT ON
? CHR(27)+"I"+"A"
? "长城系列"
?? CHR(27)+"I"+"B"
?? "微型计算机"
?? CHR(27)+"I"+"A"
?? "使用技术培训班"
? "                "
SET PRINT OFF
```

②采用格式输出命令

```
SET DEVICE TO PRINT
SET PRINT ON
? CHR(27)+"I"+"A"
@PROW()+1,1 SAY "长城系列"
?? CHR(27)+"I"+"B"
@PROW(),PCOL() SAY "微型计算机"
?? CHR(27)+"I"+"A"
@PROW(),PCOL() SAY"使用技术培训班"
@PROW()+1,1 SAY "          "
SET PRINT OFF
SET DEVICE TO SCREEN
```

③格式命令和非格式命令混合方式

```
SET DEVICE TO PRINT
SET PRINT ON
? CHR(27)+"I"+"A"
@PROW()+1,1 SAY "长城系列"+CHR(27)+"IB";
+"微型计算"+CHR(27)+"IA"+"使用技术培训班"
SET PRINT OFF
```

(4)打印实践问题

要使打印机能打印出实线表格需解决以下 3 个问题：

①选择合适的制表符

制表符可以由打印机本身提供的制表符、ASCII 代码、中文制表符等几种方法得到。提供图形符号的机器，例如长城 0520CH，则可直接用图形符号做制表符。在制表时，横线制表符最好直接用西文符号"—"，仅占一个字符宽度，而汉字库中的制表符则占两个字符宽度，当表格中有些栏目宽度是奇数位时，若用专门制表符需给奇数位的栏目加一个空格使其变为偶数，打印出来的表格才能对齐。当表格栏目数较多或表格较宽时，若横线采用专门的图形制表符，就易造成一行打不下的现象。

②字间距压缩

即使采用中文制表符，如不压缩字间距仍不能打印出横向的实线。压缩字间距的命令形式：

? CHR(27)+"P"；P 为小写字母

③行间距压缩

必须将行间距压缩才能使竖线成为实线。行间距压缩的命令形式是：

? CHR(27)+"行压缩码"

不同的打印机，其行压缩代码不同。3070 打印机，其行压缩代码为：U08 或 U12。

下面给出一个打印实线表格的小程序段。

主机为长城 0520CH，打印机为 3070，打印驱动程序为 3070C・EXE，中文操作系统为 GWBIOS3.0。

```
SET DEVICE TO PRINT
SET PRINT ON
? CHR(27)+"P"
? CHR(27)+"U08"
@PROW()+1,1 SAY "┌──────┐";
──────────┐"
@PROW()+1,1 SAY "|学习班名称|";
课程设置|"
@PROW()+1,1 SAY "├──────┤";
──────────┤"
@PROW()+1,1 SAY"|人事财务管理|;操作系统、汉字 dBASE Ⅲ 等|"
@PROW()+1,1 SAY"└──────┘";
──────────┘"
@PROW()+16,1 SAY "      "
SET DEVICE TO SCREEN
SET PRINT OFF
```

91年第三届国际信息学奥林匹克(IOI'91)试题

吴文虎

第一轮试题

以下列方式向5×5矩阵中填入数字，设数字i(1≤i<25)已被置于座标位置(X,Y)，则数字i+1的座标位置应为(Z,W)。

(Z,W)可根据下列关系由(X,Y)算出：

(1)(Z,W)=(X±3,Y)

(2)(Z,W)=(X,Y±3)

(3)(Z,W)=(X±2,Y±2)

求解问题如下：

(A)编写一个程序，当数字1被指定于某个起始位置时，列举出其它24个数字应在的位置；列举出该条件下的所有可能方案。输出方式如图1所示。

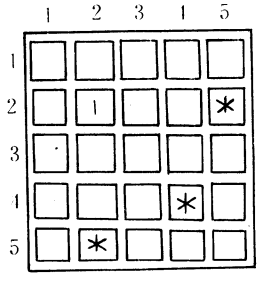


图1

(B)使数字1的起始位置分别处于矩阵的含主对角线的右上三角的每一位置，计算出每一种情况下的所有可能的方案数。

举例：如果数字1的起始位置座标被定为(2,2)则数字2的可能的位置座标应为(2,5)，(5,2)或(4,4)，上述位置在图1中用星号(*)标出。

注意：输出方式如与图1一致则较好。

评分标准：(A) 50分

(B) 25分

输出效果 15分

JURY 10分

(仲裁委员会)10分(程序风格，清晰程度，优美程度)

第二轮试题

一个S项(S－Term)是字符S和括号的序列，并以递归方式定义如下：

·字符S是一个S项。

·若M和N是S项，则(MN)也是一个S项。

S项的例子如下：

((((SS)(SS))S)(SS))

右括号没有提供任何新信息，所以可被删去。即可用(MN代替(MN)。因此上述例子可以变成：

((((SS(SSS(SS

1.写一个名为'gensterm'的过程以产生S项：该过程应产生n(n是字符S的数目，即长度)个正文文件。这些文件分别含有字符S的个数为1,…,n的所有的S项。S项由';'隔开，每个文件中的最后一个S项后面应跟有符号'·'。(文件的序号对应S字符的个数)写一个程序，接受整数n(n≤10)，使用上述过程并在屏幕上显示全部生成的S项。

考虑一种对S项的运算，唯一的运算规则(S规则)为：若S项的子项形式为：(((SA)B)C)

(其中A,B和C是S项)则该子项可被改写为：((AC)(BC))也就是说，

Context1(((SA)B)C)Context2——Context1((AC)(BC))Context2

对S项应用上述规则称为对S项的缩简。应用S规则来选择子项有不同的方法(策略)。

对S项规格化是指：对S项连续使用S规则直到不能再使用S规则为止。

一个缩减过程的实例如下：

((((SS)(SS))S)(SS))——(((SS)((SS)S))(SS))——

((S(SS))(((SS)S)(SS)))——((S(SS))((S(SS))(S(SS))))

2.选择适当的数据结构来表示S项(要便于应用S规则)写两个过程'read term'和'printterm'。'readterm'把由'gensterm'所产生的S项的表示转换为你的表示形式；'printterm'把你的表示形式转换为S项的形式。你的程序应能演示上述转换过程。

3.写一个名为'reduce'的过程。对一个指定的S项的子项(以你的表示形式给出)使用S规则，完成一个缩减步骤。你的程序应能演示此过程。

4.写一个名为'normalize'的过程。给定一个S项，该过程应能重复地使用S规则选择子项，直到不能再缩减为止；或者当缩减步骤超过某个最大值(现在给定为30)时为止，你的程序应能演示此过程。

5.最后，把上述程序合并成一个程序。应能：

a)请求用户输入一个长度n。

b)使用由过程'gensterm'产生的长度为n的S项。

c)把S项转换为你的表示形式。

d)若有可能，将其规格化。

e)输出经规格化的S项。

f)输出规格化过程中对每一S项进行缩减的步数；若在30步内不能成功地完成规格化，输出信息'not normalized'。

g)对给定长度为n的S项，输出：不能规格化的项的数目相对于总的S项个数的关系。即：

不能规格化项数；总的S项数；不能规格化项数/总的S项数。

评分标准：

序号	分数
1	20
2	25
3	15
4	20
5	10
Jury	10(清晰程度、优美程度、程序风格)

Z80的“新”指令

章永得

近些年来，人们在实践中探索到一批Z80的“新”指令。其中一部分可能是制造厂为了简化逻辑结构，从而删减的重复老功能的指令。也有一部分可能是厂家没有公开发表的“新”功能指令。这些指令对于开发Z80的软件是很有用的。

其中重复功能指令310条，大部分分布在ED带头的区域，也有一部分分布在DD和FD带头的区域，见表一。应当指出的是DISASSM88把ED4E、ED6E、ED66、ED76和ED7E等五条指令都反汇编为NOP指令是与实际情况不符的。经过本人验证，其中ED4E和ED6E本人把它叫做中断方式无定义指令。其功能为取消程序前面定义过的中断方式，直到后面再定义时止。它对可屏蔽中断是有影响的。所以应属于新功能指令。本人把它列于表四中。后三条指令应分别为IM 0、IM 1、IM 2。这些都是属于重复功能指令，本人把它们列于表一中。

新功能指令共有440条，主要分为三部分，一部分是单功能指令，主要是对变址寄存器IX和IY的高八位及低八位的传送、运算和比较指令共有92条，见表二。表中IXH、IYH分别表示为IX、IY的高八位而IXL、IYL是低八位。另一部分是所谓双功能指令，它是由位操作与传送合在一起的复合功能指令，共有336条，见表三。最后一部分是一组新的移位指令即逻辑左移指令，见表四。它与原来的一些移位指令组合在一起构成完整的八组移位指令，其功能见表五。

最后谈一下Z80CPU对所谓非法指令的响应。它们都分布在DD和FD带头的区域。当CPU遇到非法指令时，则去掉第一个字节看后面各字节，若还是非法的，则再去掉前面一字节，直到后面的是合法时，再执行该指令。如遇到DDFD1C，则应把DDFD都去掉，剩下1C是可执行指令。最后，三字节合在一起执行的结果为INC E。这一点可以由单步执行来验证。在单步执行中，整个三字节为一步，执行完后检查各寄存器的内容，可以发现E寄存器的数增加了1。就这样，Z80CPU对所有机器码都是能响应的。

本文指令表见本期封二、封三。

《电子质量》杂志

地址：广州沙河东莞庄路竹丝岗1号（广州1501信箱9分箱）

邮政编码：510610

全国各地邮局订阅，邮发代号：46－39。又月刊，年订价：7.50元

更　正

本刊90年8期《中华学习机通用EPROM读写卡》一文刊出后，发现在排印和原文稿两方面都有错误，特更正和修改如下：

1. 290行中“;A=A+2”为“:A=A+3”之误

2. 340行中“CETA$”为“GETA$”之误

3. 500行中“RESTURE”为“RESTORE”之误

4. 560行中“N$=MID(A$,K,1)”为“N$=MID$(A$,K,1)”之误

5. 580行中“RUTURN”为“RETURN”之误

6. 20行中“H=208”语句移到40行行首

7. 200行中“PRINTN”语句移到220行行首，并修改220行为：220PRINTN:A=N*16+128:PRINT”选择项正确否？(Y/N)”;:

GETA$:PRINTA$:IFA$=“N”ORA$=“n”THEN200

8. 360行中“IFB<=C…”应修改为“IFA>=B…”

9. 机器语言主程序自124地址开始修改3个数据：124:EA EA A9.00

电子工业出版社软件部征稿启事

为丰富中华机、苹果机的软件市场，为广大用户提供更多更好的软件产品，电子工业出版社软件部特向中华机、苹果机软件的开发个人或单位广泛征稿。

一、征稿范围：

1. 系统软件（语言系统、汉字系统、运行环境等）；
2. 工具软件（各种实用工具）；
3. 应用软件（管理软件、工程计算）；
4. 辅助教学软件；
5. 幼儿教育软件；
6. 游戏软件。

二、征稿要求：

1. 提供不加密的软盘（可另附拷贝加密盘）；
2. 提供通俗易懂的使用说明（用稿纸写，字迹要清楚）；
3. 软件不得一稿两投或已在社会上广泛流传；
4. 征稿软件不包括汉化的国外软件。

三、出版发行：

1. 投稿的软件经“软件评委会”评审通过后，本部寄出软件登录表及出版委托书由作者填报。我社对出版的软件给予版权号；向国内外公开发行；颁发软件出版证书；保护作者权益。

2. 对于出版的软件按软件的发行量确定稿酬。

3. 没有被采用的软件，本部在二个月内退回修改或退稿。

注：作者投稿，可先不寄磁盘而寄软件简介一份，以供我部预审。

联系地址：北京173信箱　电子工业出版社　软件部

联系人：吕　迈　电话：815342　邮编：100036

欢迎作(译)者和读者
参加91年本刊评刊活动

诸位作(译)者和读者：

时光过的很快，转眼间91年已过去大半，根据办刊特点，现在我们已着手安排92年的刊物工作了。为此，我们首先想到了热情为本刊撰稿的各位作(译)者们及喜爱本刊的广大读者，希望大家就一年来本刊各方面的工作，提出评审意见和建议，参加到我们办刊工作的队伍中来，运用大家的智慧，使我们自己的刊物越办越好，年年有新意，有进步！

“科学技术是第一生产力，是推动经济和社会发展的强大力量”。

在普及科学技术方面，《电子与电脑》愿以自己微薄之力，奉献给我国经济和技术的发展。

《电子与电脑》读者、作译者意见征询表

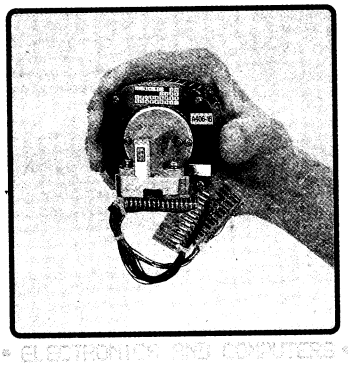
姓名		职业		地址及邮编	
您购有或使用何种微电脑	CEC-I□，天坛□，小密蜂□，APPle Ⅱ□，PC□，LASER□， PC－1500□，单片机□，单板机□，其它(机型)_____。				
您订阅本刊有：1987□，1988□，1989□，1990□，1991□。					
您对本刊现有栏目的看法及建议	综述□，学习机之友□，PC机园地□，电脑巧开发□， 学装微电脑□，维修经验谈□，学用单片机□，C语言初阶□， 读者联谊□，新书与软件□，初级程序员级水平考试辅导讲座□，信息与服务(广告)□。 建议增加_____，_____，_____，_____。				
您喜欢本刊已刊出文章中的哪几篇	软件方面				
	硬件方面				
您希望刊登哪方面的讲座					
您对本刊印制质量的看法	封面设计□，版式设计□，广告设计□，校对质量□， 正文印刷□，封面印刷□。 改进意见：_____				
您对本刊开展的读者服务工作有何看法及建议					
您认为本刊在提高知名度方面应做哪些工作					
您有何希望和要求					

表中□内，同意√，不同意×　　回函寄：100036　　北京173信箱《电子与电脑》编辑部

ISSN 1000-1077

電子與電腦

10



一九九一年

总期第79期

電子與電腦

目　录

·综述·

论电脑写作 …… 袁明光(2)

·PC用户·

BROWSE命令中FIELDS参数的菜单选择 …… 张述健(4)

给dBASE程序加入活用功能 …… 贺映文(5)

在dBASEⅢ中实现多维数组功能的另一方法 …… 程林(6)

集成字处理软件CCED的使用技巧 …… 许鹰(6)

如何自编Park? …… 葛芝宾(7)

需ROM BASIC支持的软盘改造 …… 曾礼伟(8)

最新病毒检测及清除软件VIRUSCAN Ver7.2v77 …… 梁伟杰(9)

Goldbach奇数猜想的BASIC程序 …… 陈君佐(10)

快速排序"快"在何处? …… 应宏(11)

两个高效率的排序程序 …… 李军杰(11)

·学习机之友·

CEC－I高级动画制作技术 …… 黄一禾(13)

警察抓小偷 …… 李明(15)

ProDOS磁盘操作系统入门(续) …… 廖凯(16)

高性能的CEC－I自定义组合键 …… 杨勇佳(18)

活泼的光标 …… 殷瑞冬(19)

CEC－I机汉字使用第十一行 …… 任义(19)

三页图形竖向并打 …… 陈钢(20)

电子打字机 …… 王蔚(20)

·C语言初阶讲座·

第八讲　指针与数组 …… 李文兵(21)

·学用单片机·

多功能电子钟 …… 张培仁　刘振安(27)

·电脑巧开发·

简易精确的APPLEⅡ时钟卡 …… 姜仕仁　施青松(30)

用EPPOM制作多路报警器 …… 王洪伟(32)

·游戏机电脑·

健盘电视游戏机的FBASIC语言及程序设计(上) …… 于春(33)

·维修经验谈·

AT(286)微机硬盘处理一例 …… 刘文峰(35)

从屏幕显示异常判断苹果主机板故障 …… 张艾美(36)

M2024型打印机故障检修一例 …… 王峰(37)

计算机设备故障检修表 …… 昝金省　宋靖涛(38)

IBM PC/XT微型计算机检修一例 …… 平凡(39)

MP－I微电脑快速检修法 …… 李红(40)

如何使九针打印机字迹更清晰 …… 陆秀忠(41)

YOKO747型教学语言学习机的修理 …… 郭长福(41)

微机软盘驱动器"卡盘"的简易修复 …… 乐冬青(42)

CEC－I软驱故障维修一例 …… 潘冬(42)

"坏"软盘的再生 …… 张智(43)

GW286BH系统板维修一例 …… 董纯坚(43)

怎样使用清洗盘 …… 瞿新国(43)

LASER机GA004电路的代换和修复 …… 张保田(44)

浪潮0530-AT死机故障恢复一例 …… 董国钦(45)

·专题·

一九九一年计算机初级程序员有奖竞赛试题参考答案 …… (46)

·读者联谊·

关于召开北京地区"PC"机个人用户软件交流联谊会的通知 …… (47)

·信息与服务·

天坛学习机教学网系统SEN－I型简易教学网络系统简介 …… (48)

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

论电脑写作

袁明光

一

目前，对电脑写作有各种不同的理解，有人以为，电脑写作就是电脑打字；也有人理解为，电脑写作是用电脑代替人脑进行写作构思，代替作者写成文章。这都是对电脑写作的性质不够明确。

大家知道，任何机器都是工具，不过，电子计算机是具有智能的工具，不仅可以进行数值计算，还可以进行自动控制，模拟人的感觉和思维过程等部分智能。正因如此，人们叫它做电脑是十分贴切的。虽然电脑已广泛应用于科学预测、事务管理、工程设计、医疗诊断、资料检索、文献翻译以及教学科研等各个领域。但是，这些功能都是辅助性的，其中有的被作为计算机技术的分支学科发展着，如计算机辅助测试(CAT)、计算机辅助设计(CAD)、计算机辅助教学(CAI)等。

电脑写作，不等于电脑打字，也不是用电脑代替人脑制作文章。电脑打字与电脑写作的关系，正如写字与写文章的关系一样，既有联系，又不能等而视之。电脑打字有“看打”、“听打”、“想打”之分别，着眼于单字；而电脑写作则重于“想打”，着眼于词语和语群输入。

用电脑代替人脑制作文章，这已大大超出了人工智能的范畴，也不是我们所探讨的电脑写作。因此，我认为，电脑写作是一种借助于电脑的人工智能，在电脑写作系统的支持下，运用写作软件进行的写作，简称为 CAW。

二

用电脑进行写作的可行性如何？实现电脑写作需要具备怎样的条件呢？

电脑写作的可行性，是以其电脑技术为基础的。电脑写作的实现，必须具有电脑设备(硬件)和写作软件以及会操作使用电脑的作者。这是三个必备条件，缺一不可。

电脑写作现已不是停留在理论上的探讨，而是被写作实践所证实了。国外用电脑写诗歌、小说、论文已屡见不鲜。七十年代末，美国一家刊物 Byte 就出版过电脑写作专辑，用电脑创作的小说及其插图早已见于 Omni 杂志。1982 年，加拿大作家伯克·坎贝尔用电脑花了 61.5 小时创作了一部共分 19 章的长篇小说《眼睛的法老》，经另一电脑校对后三个小时就从加拿大传输到美国的 STC 编辑部，并存入信息库中，供电脑读者索取。中文电脑写作起步稍晚，但海外华人作家陈若曦、聂华苓，台湾作家高阳、雷骧谦等，都已开始用中文电脑写作。北京作家韶华，1988 年底开始用电脑写作，深得效益，至 1989 年底陆续在电脑上写了 30 多万字的作品，1990 年 4－7 月，用“纯”工作时间 100 多小时写完 20 多万字的长篇小说《三角——红黄白》。这些事实，难道还不足以证明电脑写作的可行性吗？当然，他们所进行的电脑写作还处于初始阶段，属于低层次的，有待完善与提高，使之向较高层次发展，但这毕竟是开了个头。

三

为什么要采用电脑写作？因为电脑写作比传统写作有许多的优越性，能使写作实现自动化和现代化。电脑写作的优越性主要表现为快速性(Quickness)、灵活性(Adaptability)、规范性(Normality)等三大特性，合称为 QAN 特性。这是现代电子技术与写作结合的产物。

电脑写作的快速性(Q)是由电子计算机超速运算性能所提供的技术基础。而中文的输入，显示和打印速度等，则是决定电脑写作快速性的重要因素。经过短期训练的操作人员，采用形码输入，每分钟可输入 120 多个汉字；采用“自然码”输入，每分钟可输入 200 个汉字；要是采用语群化输入或智能化输入，其速度还会更快。CCDOS 的显示速度每秒钟 9600 位，一般的针式打印机中文打印速度每秒达 100 多字符。这些都是电脑写作快速性的技术保障。当然，快速性的实现，还要靠有好的写作软件和电脑写作者熟练地运用软件。

电脑写作的灵活性(A)表现在写作手段特别灵活。将文字信息输入电脑，这是电脑写作关键的一环。目前的输入手段有键盘编码输入、手稿扫描输入和语音合成输入等，而以键盘编码输入为最普遍。键盘编码输入方式种类繁多，已达 500 多种，但概括起来不外乎形码、音码、音形码、音义码和速写码等几类。这几类编码各有所长，具有互补性，电脑写作者可根据自己的喜好，自由选用一两种码法。将文字信息输入电脑后，则须进行编排，使之形成文章。诸如标题居中、缩格、对齐、断行、接行、调行，设定字距、行距、行宽，选用字号等，都十分方便，一按某键便成。这在传统写作中是不可思议的。

传统写作的修改相当麻烦，常常将稿纸画得琳琅满目，过后甚至作者自己都看不清。而在电脑上对文章进行增补、删除、插入、更改，无论是词语、行段还是章节，甚至自动查找全文中的某个词语并加以更改等都十分方便。正是因为有如此丰富和特别灵活的写作手段，才使得电脑写作奇妙无比。

电脑写作的规范性(N)是指用电脑写成的文章，在款式、文字、标点和文面等各方面都达到标准化、规范化。在电脑上写文章，最易做到文体格式的统一，写错字的情况不会出现(别字就难说)，繁简字相混杂的现象可消除，使字体标准，字迹清楚。而文面的整洁、清晰、美观，具有视觉美，这是传统写作无法相比的。

上述快速性(Q)、灵活性(A)和规范性(N)是电脑写作纵向方面的优越性，而横向方面的优越性则是写作编辑一体化(WEI)、写作阅读一体化(WRI)、写作存储一体化(WMI)、写作打印一体化(WPI)。总之，电脑写作的优越性是全方位的。电脑与写作结合，就给写作插上了现代化的翅膀，使写作理论与写作技能发生一次新的飞跃。

四

电脑写作的层次结构可分为：1. 字词处理层次(WHT)；2. 文体模式层次(LMT)；3. 智能辅助层次(IAT)。这三个层次是与电脑技术的发展进程相关联，与电脑技术的应用程度相适应的。字词处理层次(WHT)，是电脑写作的初级层次。它最显著的特点是输入信息以字词或词组为单位。现在的四通、科理、文豪、光明、翰林等各种型号的文字处理机，汉化 Wordstar 和 XE 之类的软件，集中地体现了第一层次的优越性。将

文字信息输入电脑，在屏幕上进行编辑，把文稿存储软盘，又将文稿打印成文，这确实比传统的执笔写作来得方便而快速。但从实现写作现代化的角度去要求，这种进步还是低层次的。

文体模式层次(LMT)，是电脑写作的中级层次。它虽然仍须文字信息输入，但它已与文体模式结合起来了，比字词处理更进了一步。写作各种文体都有一定的模式，诗与散文的模式不同，小说、散文与各类事务公文模式也不同。这种不同，是由其构成要素和构成方式相异所致。文体模式层次的特点，是将某种文体构成要素和构成方式输入电脑后，使其自动生成某种文体。国内这方面已有人作了有益的探索，上海学生梁建章，设计了一个诗歌写作程序，用来进行古典诗词和现代新诗创作。武汉的电脑写作研究专家研制了用于公文事务写作的ZL写作软件和新闻写作软件，使电脑写作在文体模式层次上迈出了可喜的一步。但总的说来，这第二层次的研究还相当薄弱，成果甚少，还不能象第一层次那样为写作者所广泛应用。之所以如此，除文体写作软件研制难度大、对此课题感兴趣的研制者少等原因外，而写作学界对各种文体模式的研究还很不够，未能达到科学化的程度也是个重要原因。

智能辅助层次(IAT)，是电脑写作的高级层次。它借助于人工智能，即电脑模拟人类的感觉、思维等能力，以辅助写作构思。在写作过程中，除材料的搜集准备外，其成败的关键在于构思。写作构思包括确立主题、优选材料、设定框架以及调动各种语言手段等各个环节。这些环节都必须运用形象思维、逻辑思维和灵感思维。除灵感思维难以得到电脑的辅助之外，我认为形象思维和逻辑思维是可以借电脑的辅助而展开联想、进行判断和推理的。比如，在众多的材料中筛选与主题相关的材料，将优选出的材料按时间或空间进行序列，循某一线索展开各式联想等等，这些都是电脑写作智能辅助层次所要解决的问题。

我国已经研制了文字识别系统和语音识别系统，在人工智能的征途上，迈出了可喜的一步。这些研究成果表明，电脑写作智能化是能够实现的。

五

最重要的是开发电脑写作系统(CWS)，电脑写作系统开发成功，乃是电脑写作进入实用阶段并走向成熟的标志。

电脑写作系统(CWS)是一个宏大的专家系统，它应由通用写作系统(GWS)和专业写作系统(SWS)构成。通用写作系统包括通用词语库、通用句型生成模块、通用词语联想模块、通用表达选择模块等。通用词语库是通用写作系统的基本构件，句型的生成、词语的联想、表达的选择，都须依靠它提供丰富的词语资料。词语库的建立，除了注意其丰富性和覆盖面之外，其排列及调用方式多样化也是不可忽视的。我主张，既要有音序、词形的排列，也要有词义的排列，既采用词首调用，也采用语尾调用，(如写诗要押韵，就需采用语尾调用)使其具有广泛的适应性。

通用句型生成模块，具有自动生成各种类型的句子功能，作者只要输入其句子的要素，便能自动生成所指定的那种句型。通用词语联想模块，具有通过词语帮助作者进行联想的功能，作者一旦输入想要表现的某一词语(事物)，便能展现与此词语(事物)有关联的各种词语(事物)，以扩展作者的思路与视野。通用表达选择模块，具有选择叙述、描写、说明、议论等各种表达方式的功能，作者只需输入其表达要素，便可形成各种表达方式供作者选择。句型生成、词语联想和表达选择等模块功能的实现，将使电脑写作在输入语群化、智能化的道路上向前迈进。

专业写作系统(SWS)是一个以适应各种专业写作需要而建立的大规模系统，它包涵事务写作系统(RWS)、科技写作系统(S－TWS)、新闻写作系统(NWS)、理论写作系统(TWS)、文学写作系统(LWS)等众多的子系统。每个子系统都建立有各自的专业词语库、素材储备库和文体生成模块。

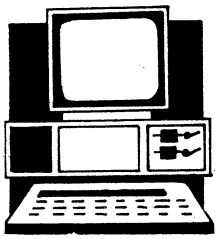
专业词语库中一部分是固定词语，一部分是自定词语，以适应各种不同语言风格的作者的需要。有人担心，用电脑写文章会千篇一律，失去个人风格。其实，这种担心是多余的。个人风格是由多种因素决定的，除了内容和思维方式的不同外，最主要的是语言风格的不同，自定词语，就是根据各自的独特的习惯用语和词语组合方式而建立的，它从词语方面保证了个人风格的实现。专业词语库是通用词语库的扩充。在进行专业写作时，可同时调用两个词语库的词语，也可调用通用写作系统中的句型生成、词语联想、表达选择模块。

素材储备库是一个内容独特的知识库，一般是由作者自己建立，或者是由写作系统研制者建立基本框架，然后再由作者自己扩充。它的特点是：既要容量大，又要便于存取。因此需要科学地划分好素材分类仓，给各种素材编上代号，随时将观察或阅读所获得的素材输入分类仓内。积累素材是写作的重要一环，无论传统写作还是电脑写作都是如此。传统写作将所获得的素材记入人脑或笔记本、卡片，而电脑写作是将素材输入电脑储存起来，以供调用。在写作实践中，许多作者都体验过，一旦写作灵感袭来，常常是来不及写下，所想好的东西就跑掉了，甚至事后很难追忆，感到十分遗憾。用电脑进行写作，在电脑中建立素材储备库，则可弥补这种遗憾。当灵感来临时，作者便可以迅速从素材储备库中找到与其相似的各种资料，很快形成文字，显示于屏幕上，然后再加以修改，使它更符合作者的构思。

专业写作系统中，有一些文体的模式已经定型，如事务写作系统中的通知、书信、总结、合同、起诉书、判决书，科技写作系统中的实验报告，新闻写作系统中的消息，理论写作系统中的短论，文学写作系统中的诗歌等，相对地说，实现文体自动生成容易一些，则可在系统中先实现。将某种文体的构成要素和构成方式输入电脑，使其按规定的模式自动生成某种文体，这是实现写作自动化、现代化关键的一环。

开发电脑写作系统有一个过程，在这个过程中，我认为进行双向交流与双向适应是十分重要的。双向交流，是指写作系统研制者与写作者之间所进行的正向和反向信息交流，一方面，写作系统研制者从写作者那儿取得写作动态信息，以促进系统研制；另方面，写作者从系统研制者那儿取得开发进展信息，以利于将系统用于写作。双向适应，是指写作系统研制者与写作者之间的相互适应，一方面，系统研制者需要适应写作者的各种不同构思方式、词语组合方式、表达方式，以满足写作的不同要求；另方面，写作者需适应系统研制者的新的构想与独特的技术处理，以调整自己的写作认识结构与技能结构。写作者的个人习性与风格并非一成不变的，当他采用电脑写作系统进行写作后，自己的认识结构和技能结构必然会起变化，最后将导致作品风格的改变。这种改变使得作品的风格变得多样化，并非坏事，写作者要有充分的思想准备，努力去适应它，从而充分发挥电脑写作系统的最大效益。

已经崛起的电脑写作将会按照它自己的规律向前发展，走向成熟。让我们迎接这一天的到来！



PC用户

BROWSE 命令中 FIELDS 参数的菜单选择

陕西扶风胜利机械厂子弟学校　张述健

在 dBASE Ⅲ 中，BROWSE 命令以窗口显示方式给用户的数据修改、浏览以及追加提供了很大的方便。但由于屏幕所能显示的字段有限，在数据库文件的字段比较多的情况下，人们一般要对字段进行选择，将需要的字段固定在屏幕上，使用户界面清晰、直观。然而，BROWSE 命令格式中，FIELDS 可选参数项的选择不仅输入量大（敲键次数多，尤其是字段使用汉字时），而且要求操作人员对具体的库文件结构也得非常熟悉（起码需要记住各个字段的名称），给操作人员造成很大的不便。笔者受“菜单”程序设计思想的启发，用 dBASE Ⅲ 语言实现了 BROWSE 命令中 FIELDS 可选参数项选择的“菜单”化，减轻了操作人员的负担，提高了数据修改的速度和准确性。

一、首先利用 dBASE Ⅲ 中的 COPY TO（文件名）STRU EXTE 命令将源库文件的结构拷贝到一个结构描述文件“XU_1. DBF”中，将库文件中的每一个字段转化为“XU_1. DBF”中的所有记录，然后程序访问结构描述文件“XU_1. DBF”，巧妙地应用记录号做为各字段选择的序号，将 FIELD_NAME 的内容（即源库文件中的各个字段名称）以“菜单”的形式直观地显示在屏幕上（可分屏显示），供操作人员选择。选择时，只需简单地敲入字段的序号即可，非常方便、灵活。若多项选择，序号之间用逗号“,”隔开；若直接回车则保留全部的字段，接收的序号序列送入变量 JLH 中。

二、根据变量 JLH 中所接收到的序号序列，程序分三种情况进行测试：①利用 LEN（字串）函数测试变量 JLH，结果送 L 中，若 L＝0，则表明用户将保留全部字段；②若 L≠0，再利用 AT（子串，主串）函数测试变量 JLH 中是否会有子串“,”，若没有，则表明用户仅选择了一个字段；③若有子串“,”，程序则进入一个双重循环结构，对整个序号序列（JLH 变量中的值）进行测试分析，做分离工作。内循环体将逐个确定每个序号在整个序号序列中的起始位置和字符数，分别送入变量 I、J 中；然后，外循环体根据 I、J 的值，使用 SUBS（）函数将它们逐个分离出来，还原成所需要的记录号，用 GO 命令在结构描述文件 XU_1. DBF 中进行定位，将所选字段名称（即 FIELD_NAME 的内容）送入变量 ZDM 中，利用变量 YY 排列组合所选择的所有字段名，形成用户所期望的可选参数序列。最后用 dBASE Ⅲ 中的宏替换命令 & 完成 BROWSE 中 FIELDS 可选参数的替换赋值，用户即可按照自己的意愿对所造各字段进行各种操作。

主要程序段如下：

“菜单”显示程序段

```
……
ACCE “库文件名?” TO KNAME
USE &KNAME
COPY TO XU_1 STRU EXTE
X＝1
Y＝6
DY＝17
YY＝INT(80/DY) * DY
USE XU_1
CLEA
@0,30 SAY “《需修改、追加及浏览的字段》”
DO WHILE . NOT . EOF()
IF X＞8
WAIT “请记住各字段名序号！按任一健继续……”
@1,0 CLEA
X＝1
Y＝6
DY＝17
YY＝INT(80/DY) * DY
ENDI
I＝STR(RECN(),3)
@X,Y SAY I＋“---”＋FIELD-NAME
Y＝Y＋DY
IF Y＞YY
X＝X＋1
Y＝6
ENDIF
SKIP
ENDDO
@ROW()＋1,5 SAY “多项选择时，序号间用逗号隔开；
                直接回车则保留全部字段。”
ACCE “请输入所选字段序号：” TO JLH

……
```

“测试、分离”程序段

```
……

L＝LEN(JLH)
YY＝SPAC(50)
I＝1
J＝0
IF L＝0
USE &KNAME
BROW
ELSE
IF AT(“,”,“&JLH”)＝0
X0＝SUBS(JLH,1,L)
GO VAL(“&X0”)
ZDM＝TRIM(FIELD_NAME)
YY＝YY＋ZDM
ELSE
DO WHILE . T.
DO WHILE I＜＝L
XX＝SUBS(JLH,I,1)
```

```
IF XX<>“,”
I=I+1
J=J+1
ELSE
EXIT
ENDIF
ENDDO
X0=SUBS(JLH,I-J,J)
GO VAL(“&X0”)
ZDM=TRIM(FIELD_NAME)
IF I<L
YY=YY+ZDM+“,”
FF=“LOOP”
ELSE
YY=YY+ZDM
FF=“EXIT”
ENDIF
J=0
I=I+1
&FF
ENDDO
ENDIF
YYY=TRIM(YY)
USE &KNAME
BROW FIELD &YYY
ENDIF
……
```

给 dBASE 程序加入活用功能

桂林市人民银行科技科　贺映文

我们在调试程序、运行管理系统时，经常会碰到一些特殊情况，不得不中止程序的运行，浪费了很多不必要的时间。例如，我们在编制会计电脑记帐程序时，稍不小心就碰到：

```
该文件不存在!
        ?
USE lszsj INDEX ndlszsj
调用处为－C:pxls.prg
……
……
是否中止执行命令文件? (Y/N)
```

出现这种情况的原因有两个：一个是 C 盘上没有 lszsj. dbf 数据库文件；另一种可能是没有产生 ndlszsj. ndx 索引文件。通常的解决办法是按 Y 键，中止执行命令文件，返回圆点提示符或返回 DOS 系统，转去执行特殊处理，将 lszsj. dbf 拷贝进去，并对其进行索引。但处理完毕后还要重新启动，就象上面的例子，还要先进主控菜单，再进子菜单，子菜单再进子菜单，才能执行打印排序流水帐的程序，做起来很麻烦。

于是，我就想：如果在碰到上面的情况时，能够不按 Y 键退回，而按 N 键在程序运行状态下进行相应处理后，再继续执行原程序就好了。经过一番努力，笔者利用 dBASE Ⅲ 设计了两个活用子程序（见 hy1. prg 及 hy2. prg），加到程序适当处，巧妙地解决了上述问题。在上述例子中，加入两个活用程序后，不必按 Y 键退回，而是按 N 键转去进行活用处理（即拷贝 lszsj. dbf 及产生索引文件 ndlszsj. ndx 等），处理完毕后，自动转到原来执行状态，继续执行原程序。

加入活用功能后，系统不仅能处理类似上述的问题，而且还可以在程序运行过程中执行各种 DOS 内部命令和外部命令，同时也可执行 dBASE 命令及 WordStar 等，活用处理后，继续执行原程序，大大地提高了程序的灵活性，优化了程序设计。

```
* * * 活用子程序 1   hy1.prg
* 在调用某文件的语句后插入本程序
if .not. file('文件名')
do hy2
* 在此重写调用某文件的语句
endif

* 活用子程序 2   hy2.prg
set talk on
set consoe   on
do while .t.
clear
acce '请进行活处理，处理好后，按回车继续执行原程
      序)：' to ml
if ml>' '
&ml
? ' '
wait '      按任意键继续!'
else
exit
endif
enddo
set console off
set talk off
use
return
```

（上接第 8 页）

即告成功。现在不需 ROM BASIC 支持即可运行。有些盘带有自启动，如不需要，可将 IBMBIO. COM、IBMDOS. COM、COMMAND. COM 三个文件删掉（在用步骤（4）时这三个文件已用 FILEnnnn. REC 的形式列出，步骤（6）更名），找出盘中的批处理文件名（扩展名为 BAT），以后在机器启动后，打入批处理文件名即可运行。

此法在 PC/XT 机上获得成功。

在 dBASE Ⅲ 中实现多维数组功能的另一方法

程　林

dBASE Ⅲ 没有数组语句。《电子与电脑》91 年第四期上介绍了一种方法，即巧妙应用 & 函数来实现多维数组功能。本人感到这种实现方法有其局限性，一是该方法所述的数组元素其实是内存变量，由于 dBASE Ⅲ 要求当前工作的内存变量总数小于 256 个，因此不能定义太大；二是其数组下标的实现要将循环变量由"N"型转换成"C"型，并且要删去前面的空格符再用 & 函数实现，故在调用数组元素时比较复杂。

这里介绍另一种方法——"取子串"方法。虽说实质还是采用内存变量代表其数组元素，故数组大小仍受限制，但实现方法比用 & 函数简便、直观，调用数组元素也较简单。对于各维元素小于 26 的数组，其实现程序如下(以实现一个 11×6 数组运算为例)：

```
SET TALK OFF
CLEAR
N="010203040506070809101112131415161718192021222
    232425"
I=1
DO WHILE I<=11
J=1
DO WHILE J<=6
V="A"+SUBSTR(N,I*2-1,2)+SUBSTR(N,J*2-1,2)
&V=I*J
?? "&V="+STR(&V,5)+SPACE(6)
J=J+1
ENDDO
?
I=I+1
ENDDO
```

运行结果：

A0101= 1	A0102= 2	A0103= 3	A0104= 4	A0105= 5	A0106= 6
A0201= 2	A0202= 4	A0203= 6	A0204= 8	A0205= 10	A0206= 12
A0301= 3	A0302= 6	A0303= 9	A0304= 12	A0305= 15	A0306= 18
A0401= 4	A0402= 8	A0403= 12	A0404= 16	A0405= 20	A0406= 24
A0501= 5	A0502= 10	A0503= 15	A0504= 20	A0505= 25	A0506= 30
A0601= 6	A0602= 12	A0603= 18	A0604= 24	A0605= 30	A0606= 36
A0701= 7	A0702= 14	A0703= 21	A0704= 28	A0705= 35	A0706= 42
A0801= 8	A0802= 16	A0803= 24	A0804= 32	A0805= 40	A0806= 48
A0901= 9	A0902= 18	A0903= 27	A0904= 36	A0905= 45	A0906= 54
A1001= 10	A1002= 20	A1003= 30	A1004= 40	A1005= 50	A1006= 60
A1101= 11	A1102= 22	A1103= 33	A1104= 44	A1105= 55	A1106= 66

集成字处理软件 CCED 的使用技巧

常州 102 医院　电脑室　许　鹰

CCED 是一种字处理、划线制表与数据加工融为一体的中文集成字处理软件。用户既可以方便地利用四个光标移动键任意画线，又可以利用程序本身的自动制表功能迅速地生成各种表格，对表中的每个格子都可以随时改变行宽列距。填表中的数据、文字时表线自动保护，不会破坏。对表中的数据可以方便地进行数据计算，还能自动打印 dBASE 的数据库表格。字处理时除可以进行各种字、词、块的拷贝、移动、复盖等多种功能外，还可做到中英文混排标点首尾禁则、首尾对齐。CCED 的使用环境可以适应各种显示器及显示行数。但是 CCED 有一个最大的不足，那就是不能进行子目录操作，虽然在它的按装说明中提出了解决的办法，但是在实际操作中往往困难较大，不够方便，现介绍一种十分简单而方便的方法，使 CCED 可以在任何路径下随意调用。

首先，将 CCED 的几个文件拷入 C 盘根目录，然后在根目录中调用一次，这时 C 盘根目录中就会自动产生一个 CCED. OV1 的复盖文件，大小只有 256 字节，只要将这个小文件拷入需要调用 CCED 的任何软盘或硬盘的子目录中，那么就可以在含有 CCED. OV1 的任何子目录中随时调用 C 盘根目录中 CCED 的各类文件了。如果在调用时还想使用在线帮助(按 Ctrl+J)，那么只要把 CCED. OVL 拷入相应的子目录就可以了，使用起来十分方便。

在各个不同的子目录中，可以通过 Shift+F7 设定互不相同的屏幕参数，即可以在不同的子目录中产生不同的前后景颜色，不同的画线线型，不同显示行数的工作屏幕，使用起来更加生动形象。但如果使用的是单色显示器，则在显示方式中应输入 18，否则无法正确显示。

如何自编 Park

盐城师专　葛芝宾

硬盘是微机系统的重要外部设备，它随主机电源的通断而运停。由于读写磁头与盘片的接触方式，在开关机的瞬间，磁头与盘面的磨损加大，常会使硬盘因划伤而出现故障或报废。减少硬盘被划伤的方法，是在关机断电前运行 park.com 程序，把磁头移至盘片中心附近的非数据存储区（具有断电自动 park 功能的硬盘系统不在此例）。然而，我们在实际工作中发现，不少 park 程序都受硬盘型号或机型的局限，而有些 park（PC DOS 系统盘提供或随机 Utility 软件盘提供）存在错误，不能正常运行。例如在 PC DOS 3.20 下运行该系统提供的 park，会显示“Unknown System…”而退出，用 AST 286 Premium Utility Software 盘上的 park.com 锁定该机的 40MB 硬盘，在显示“Disk error”后不能继续。笔者初步分析认为属于程序本身的问题，PC DOS 3.20 版的 park 在运行之初先检查硬盘 BIOS 的 INT 13H 入口段值的高字节是否为 C8H 或 FCH，而事实上 DOS 3.X 的内核初始化后，INT13H 及其它一些 ROM BIOS 的中断向量均被改写，难怪其 park 不认识系统了。据笔者对不同的 AST 286 机检测，硬盘 BIOS 所在段也不同，有的位于 CE00 段，而有些则在 F000 段。至于 AST 286 机的 park，主要问题在于对硬盘柱面数的计算错误。再如 GW 0520CH、紫金 AT 等机器虽有 park 功能的工具程序，但均为己用。在此情况下，笔者着手自编共用的 park.com 程序，现奉献给读者共享。

我们知道，不同型号的硬盘的参数、操作命令码、硬盘 BIOS（INT 13H）程序不尽相同，但有二点是基本相同的：（一）硬盘 INT 13H 向 DOS 内核提供的功能服务应是一致的；（二）硬盘的柱面编号及分布顺序是一致的，即柱面号总是由外圈向盘心递增的。park 程序的任务就是获取当前系统配置的硬盘数目、柱面数，然后依次将各硬盘的磁头移至最末一道的内侧。这里有二点需说明：（1）不论硬盘的磁头有多少（4、5 或 7 等），移动某一磁头时，所有磁头随之一起沿同一方面移动相同距离（磁道数）；（2）由于盘心区域很有限，磁头移出最末柱面不能多于 2－3 道。在介绍 park.com 编程之前，我们先将 INT 13H 的子功能列于表 1，对其各子功能的调用参数如下：

入口参数：AH 子功能号（见表 1）
　　AL 扇区数（读/写命令）或间隔数（格式化命令）
　　CH 柱面号低 8 位
　　CL 高 2 位为柱面号的高 2 位，低 6 位为扇区号
　　DH 磁头号
　　ES:BX 读/写的内存缓冲区始址
出口参数：成功 AH＝0 CF＝0
　　不成功 AH＝错误状态字节 CF＝1
　　对于返回驱动器参数子功能（AH＝8H），还返回
　　DH 磁头数－1
　　DL 微机系统的硬盘台数
　　CH 柱面数的低 8 位
　　CL 高 2 位为柱面数高 2 位，低 6 位为扇区数/道

表 1　硬盘 INT 13H 的子功能

AH＝00H	复位硬盘控制器，设置硬盘驱动器参数
AH＝01H	读上一次中断 13H 执行后的状态到 AL
AH＝02H	读
AH＝03H	写
AH＝04H	校核（读但不写到内存）
AH＝05H	格式化指定磁道
AH＝06H	格式化指定磁道并设置坏道标志
AH＝07H	从指定道格式化整个硬盘
AH＝08H	读驱动器参数
AH＝09H	设置驱动器参数到控制器
AH＝0AH	长读
AH＝0BH	长写
AH＝0CH	寻道
AH＝0DH	复位控制器并设置驱动器参数
AH＝0EH	读控制器缓冲区
AH＝0FH	写控制器缓冲区
AH＝10H	测试驱动器是否准备好
AH＝11H	重回 000 道（校准）
AH＝12H	控制器内存诊断
AH＝13H	驱动器诊断
AH＝14H	控制器内部诊断

从表 1 可见，可用来移动磁头的子功能有 02H、04H、0CH 等号子功能，但采用不需内存缓冲区的子功能较为方便。下面所附的即是在 DEBUG 下编写的简短而实用的 park.com，在此仅作二点说明：（1）由于软盘 INT13H 未提供 08H 号子功能，若系统未安装硬盘，可据 08 号子功能调用成功与否确定（不成功，CF＝1）。（2）在移动磁头前，未采用惯常的 11H 号子功能让磁头回 000 道，而用 10H 号子功能测试驱动器状态，这样可使磁头（若位于较内圈时）移动完成得快些。在驱动器无故障时，重校准（11H 号）不是必须的，而减少磁头来回移动则是有益的。

```
C>debug
-a100
cs:0100 jmp 180
cs:0103
-a10a
cs:010a db 'PARK Hard Disk Heads Utility.',0d,0a,24
cs:012a db 'No Drive Installed!',0d,0a,24
cs:0140 db 'Disk Error!',0d,0a,'Drive(s) Was Not
        Parked!',0d,0a,24
cs:0168 db 'Power System OFF',0d,0a,24
cs:017c
a180
cs:0180 mov dx,10a
cs:0183 mov ah,09
cs:0185 int 21
cs:0187 mov ah,08
```

```
cs:0189 call 1f0
cs:018c jnb 197
cs:018e mov dx,12a
cs:0191 mov ah,09
cs:0193 int 21
cs:0195 int 20
cs:0197 or dl,dl
cs:0199 jz 18e
cs:019b mov [0104],dl
cs:019f rol cl,1
cs:01a1 rol cl,1
cs:01a3 xchg ch,cl
cs:01a5 and cx,03ff
cs:01a9 mov ax,cx
cs:01ab inc ax
cs:01ac mov [108],ax
cs:01af mov [105],cx
cs:01b3 mov ah,10
cs:01b5 call 1f0
cs:01b8 jnb 1bf
cs:01ba mov dx,140
cs:01bd jmp 191
cs:01bf dec byte ptr [104]
cs:01c3 mov ax,0401
cs:01c6 call 1f0
cs:01c9 jb 1d8
cs:01cb inc word ptr [105]
cs:01cf mov ax,[105]
cs:01d2 cmp [108],ax
cs:01d6 jnb 1c3
cs:01d8 cmp byte ptr [104],00
cs:01dd jz 1e5
cs:01df inc byte ptr [107]
cs:01e3 jmp 1b3
cs:01e5 mov dx,168
cs:01e8 mov ah,09
cs:01ea int 21
cs:01ec int 20
cs:01ee mov dl,[107]
cs:01f2 or dl,80
cs:01f5 mov dh,00
cs:01f7 mov cx,[105]
cs:01fb xchg ch,c1
cs:01fd ror cl,1
cs:01ff ror cl,1
cs:0201 or cl,01
cs:0204 int 13
cs:0206 ret
cs:0207
-rcx
CX=0000
:0108
n park.com
-w
```

需 ROM BASIC 支持的软盘改造

四川涪陵地区林业局　曾礼伟

在 IBM－PC/XT 及其兼容机上运行的需 ROM BASIC 支持的图形软件，在不带 ROM BASIC 的机上往往不能运行或经过一定的加密处理而不能 COPY。本文据此提出一种改造这类软盘的方法，使之能在不带 ROM BASIC 的机上运行并且能 COPY 其中的各文件。

改造的基本思路是：一般这类图形或游戏软件运用的是 BASICA. COM，须在 ROM BASIC 的支持下才能运行，若能将 BASICA. COM 改换成 BASICA. EXE，问题即可解决。但往往这类软件经过了一些加密处理（如修改文件属性、改变簇号、隐含子目录等）使之不能 COPY。所以首先要解密。下面，依次给出改造的方法。

1. 用 DISKCOPY 命令备份软盘一张。

2. 用调试程序 DEBUG 的 D 命令将备份盘的文件目录项列出，并记下各文件名和各文件的 1A～1BH 字节和 1C～1FH 字节的内容（1A－1BH 字节为文件的起始簇号，有些加密盘将此改变；1C－1FH 字节为文件长度。记下这些数据的目的是为以后确定文件名用）。

3. 用检查磁盘状态命令 CHKDSK 〔d:〕/V 对备份盘进行查看，此时隐含文件和子目录都将显示出来。记下或打印所有文件名和子目录名。

4. 用恢复文件命令 RECOVER 〔d:〕对备份盘进行恢复。使用此命令时，备份盘上的文件目录将被认为是坏的而被重新设置在新的系统目录下，所有文件将以 FILE nnnn. REC 的形式恢复到指定的盘片上（子目录也以文件的形式出现）。

5. 用调试程序 DEBUG 的 D 命令将刚恢复的各文件目录项列出，并记下各文件的序号（即 nnnn）和各文件的 1A～1BH 字节和 1C～1FH 字节的内容。

6. 将步聚（5）与步骤（2）记下的内容进行比较（即比较各文件的簇号和长度）可确定出原文件的真实文件名。各文件名一一确定出后，便可用 REN 命令将恢复盘中的 FILEnnnn 文件更名为真实文件名。若原加密时用的是怪字符文件名，可更名为可辨认的文件名；若加密盘改变了原簇号，可根据文件长度和建立文件的顺序，找出真实文件名；在原加密盘中的子目录经步骤（4）后恢复为长度为 1024 的 FILEnnnn. REC 文件，经与步骤（3）记录的文件名和子目录名进行比较，可找出子目录名。更名完后，解密工作完成。

备份盘经上述步骤解密后，即可对用 DIR 列出的各文件进行 COPY 了。对于经步骤（4）恢复的各 FILE nnnn. REC 文件，其代表属性的 0BH 字节令为 00，使用者可根据对文件属性的要求进行改变。

用 DIR 命令可找到 A 盘中的 BASICA. COM 文件，然后用 BASICA. EXE 文件替换它（下转第 5 页）

最新病毒检测及清除软件 VIRUSCAN Ver7.2v77

梁伟杰

自从 1989 年美国 McAfee Associates 公司推出 VIRUSCAN 软件后，VIRUSCAN 就以其惊人的杀“病毒”能力而风摩全球电脑界。VIRUSCAN 更不断推陈出新，以对抗形形式式的新病毒，使得它们无所遁形。

今年 5 月份推出的 VIRUSCAN 最新版本 Version7.2v77（简称 SCAN77），更是强劲无比。它可以检测 243 种病毒及它们的若干变种，总共 503 种，并可清除其中的 98%共 492 种病毒。这些病毒占目前个人电脑上流行病毒的 96%。SCAN77 可在任何 PC 及兼容机上运行，只要求 256KB 内存及 DOS2.0 以上版本。

SCAN77 比其前身 SCAN76 可以多检测 4 种病毒，可以多清除 19 种病毒，如 Black Monday，Print Screen－2，3066 等病毒。时间上则比 SCAN76 快 5%。

SCAN77 中包括 SCAN.EXE，CLEAN.EXE 以及 VALIDATE.COM 三个主要文件。以下分别介绍其使用方法。

SCAN.EXE 是用于检测病毒的，其命令格式如下：

SCAN d1：… d10：/A/AV/CV/D/M/NOMEM/E.xxx/EXT d：filename/FR/MANY/NLI/NOBREAK/NOPAUSE/REPORT d：filename/RV

注：d1：…d10：为要检测的驱动器号，驱动器号后可带有路径名，文件名。“/”后的参数均可以缺省，缺省时不执行该项相应的操作。

各命令参数的意义如下：

/A：检测所有文件，包括数据文件（若不使用此操作，SCAN77 只检测.BIN，.COM，.EXE，.OV?，.PGM，.SYS，.XTP 以及用/E.xxx 参数指定的扩展名。

/E.xxx：操作如上所述，可以同时加入几个扩展名。

/AV：对不同文件加入不同的 CRC（循环冗余校验码）

/CV：检验文件的 CRC

/RV：删去文件的 CRC，不能与/AV 同时使用。

/FR：用法文代替英文显示。

/D：当检测到该文件有病毒时删除该文件。此参数一般只用于对付 CLEAN.EXE 不能清除的病毒。

/EXT d：filename：使用记有病毒特征串的外部文件来检测用户自己发现的病毒。该文件可用任何编辑器来写，如 PE，Turbo Basic 的编辑器等，记录格式如下：

#NEW virus in.EXE——此为说明

“5D FF 4E” NEW VIRUS 〔NVI〕

特征串 名称 代码

注意，每行结尾必须用＜CR/LF＞（回车/换行），特征串可带有“?”，“*”等万用符。

/M：对内存进行所有病毒的检测，注意：即使不用/M，SCAN77 也自动对内存检测 Disk killer，Taiwan－3，1554，Brain 等危害性大的隐蔽病毒，当检测到内存有这 28 种病毒之一存在时，它会建议你重新启动，因若在此时进行 SCAN，会带来大面积的破坏和感染。此点在使用时应多加注意。

/NOMEM：与上相反，不对内存进行检测。只有当时肯定内存没有病毒时才可使用此参数。

/MANY：对多个软盘进行检测。

/NLI：不检测用 LI.EXE 压缩的文件。

/NOBREAK：检测期间，不允许 Ctrl－C/Ctrl－Break 中断。

/NOPAUSE：检测期间，不允许屏幕的暂停上卷。

/REPORT d：filename：建立一个病毒感染报告文件，存放检测结果。

当检测完毕，会显示检测结果，并返回一代码。其意义如下：

0：未发现病毒。

1：发现一个或多个病毒存在。

2：程序有错误。

CLEAN.EXE 用于清除病毒，其格式如下：

CLEAN.EXE d1：…d10：〔Virus ID〕/A/E.xxx/FR/MANY/M/REPORT d：filename/ EXT d：filename/NOPAUSE

〔Virus ID〕为病毒代码，由 SCAN 时所得结果给出。注意：病毒代码必须严格按格式书写。其余参数的意义同 SCAN.EXE。

当用 CLEAN 清硬盘分区表中的石头病毒时，很可能会破坏分区表，此时最好将硬盘的数据先备份，再清病毒。

VALIDATE.COM 主要用于对文件的 CRC 校验。在 SCAN77 盘上所有文件均加有 CRC 码。如执行以下操作，便可得到：

```
A＞VALIDATE SCAN.EXE
    FILENAME：SCAN.EXE
    SIZE：59，123
    DATA：04－25－1991
FILE AUTHENTICATION
    Check Method 1：0DF0
    Check Method 2：0989
```

如果不是这样，那么你的 SCAN 程序已被修改，或已感染病毒了。

建议你把 SCAN77 做成一张可启动软盘，每次操作最好先启动，这样就绝对安全了。若能配合上专清“国产”病毒的公安部 SCAN2.0，就真的能防百毒了。

Goldbach 奇数猜想的 BASIC 程序

汕头四中　陈君佐

从 1991 年第 1 期的《数学学报》，得知著名数学家陈景润仍在攻哥德巴赫猜想的关，并进而追阅他和王天泽同志发表在 1989 年第 5 和第 6 期《数学学报》和 1989 年第 11 期《中国科学》上的论文，通过对 L 函数零点分布及算术数列中的素数分布两问题的研究，证明奇数 $N \geqslant e^{e^{11.503}}$ 都能表为三个奇素数之和的结论。

为配合他们的研究，我把电算奇数猜想的 BASIC 程序介绍于下，这程序能完整的，并没有重复的把奇数 N，一个不缺，也一个不多的，把满足：

$$N=P_1+P_2+P_3$$

的解数 T(N)展现在读者面前。

如 33 这个奇数，可表为

1　33=3+7+23
2　33=3+11+19
3　33=3+13+17
4　33=5+5+23
5　33=5+11+17
6　33=7+7+19
7　33=7+13+13
8　33=11+11+11

用潘承洞教授《哥德巴赫猜想》专著的符号，就是 T(33)=8。

今年是 1991 年，这是一个不大的奇数。LASER－310 微机，和 IBM PC/XT 微机都一致，并 5 次显示，顺序一个不变的给出 N=1991 的 3285 组解，即

T(1991)=3285。

考虑到偶数猜想 BASIC 程序（见 1991 年本刊第 3 期）还不是最佳的，故再度修改，配以 GOSUB 300 语句，使奇数猜想比偶数猜想程序更优化，节约上机时间。

```
10 REM 哥德巴赫奇数猜想程序
20 INPUT "请送入奇数=";N
30 FOR X=3 TO N/3 STEP 2
40 FOR I=2 TO SQR(X)
50 IF X/I=INT(X/I) THEN 100
60 NEXT I
70 A=0:B=B+1:GOSUB 300
100 NEXT X
120 END
300 R=N-X
310 FOR Y=X TO R/2 STEP 2
320 FOR I=2 TO SQR(Y)
330 IF Y/I=INT(Y/I) THEN 430
340 NEXT I
350 Z=R-Y
360 FOR I=2 TO SQR(Z)
370 IF Z/I=INT(Z/I) THEN 430
380 NEXT I
390 A=A+1:M=M+1
410 PRINT M;" ";B;" ";,;" ";N;"=";X;"+";Y;"+";Z
430 NEXT Y
440 RETURN
```

当用 RUN 运行程序后送入 33 时，IBM PC/XT 机很快给出：

```
1 1 1 33=3+7+23
2 1 2 33=3+11+19
3 1 3 33=3+13+17
4 2 1 33=5+5+23
5 2 2 33=5+11+17
6 3 1 33=7+7+19
7 3 2 33=7+13+13
8 4 1 33=11+11+11
```

我编这个程序，自己感到十分满意，因它不仅给出 33 个这奇数，可表为 8 组不同的三素数之和的解答，而且给出 X=3，有 3 组不同的解，X=5（第二个奇素数!!），有 2 组不同的 y 和 z 值的解，X=7，有 2 组不同的 y 和 z 值的解，X=11，只有一组满足 $N=P_1+P_2+P_3$ 的解，合共 T(33)=8。

由于没有打印机，故 T(1991)=3285 又得手抄。下面是部分结果：

```
250 6 19 1991= 17+233+1741
300 6 69 1991= 17+953+1021
350 8 16 1991= 23+269+1699
400 9 6 1991= 29+ 89+1873
450 10 2 1991= 31+ 53+1907
500 11 6 1991= 37+131+1823
550 12 24 1991= 41+293+1657
600 12 74 1991= 41+911+1039
650 14 15 1991= 47+281+1663
700 15 12 1991= 53+197+1741
750 16 1 1991= 59+ 59+1873
800 16 51 1991= 59+739+1193
850 18 4 1991= 67+191+1733
900 19 25 1991= 71+353+1567
950 20 3 1991= 73+197+1721
1000 21 25 1991= 79+929+ 983
1050 22 49 1991= 83+937+ 971
1100 24 1 1991= 97+107+1787
1150 25 25 1991=101+311+1579
1200 25 75 1991=101+859+1031
1250 27 21 1991=107+433+1451
1300 28 23 1991=109+773+1109
1350 29 45 1991=113+839+1039
1400 31 15 1991=131+277+1583
1450 31 65 1991=131+919+ 941
1500 33 7 1991=139+353+1499
1550 34 33 1991=149+691+1151
1600 36 7 1991=157+281+1553
```

1650 38 6 1991=167+241+1583
1700 39 10 1991=173+331+1487
1750 40 18 1991=179+389+1423
1800 41 26 1991=181+797+1013
1850 42 45 1991=191+739+1061
1900 44 18 1991=197+487+1307
1950 45 22 1991=199+761+1031
2000 47 17 1991=223+719+1049
2050 48 44 1991=227+853+ 911
2100 50 25 1991=233+641+1117
2150 52 4 1991=241+269+1481
2200 53 22 1991=251+461+1279
2250 54 18 1991=257+571+1163
2300 56 1 1991=269+269+1453
2350 57 11 1991=271+461+1259
2400 59 22 1991=281+601+1109
2450 61 8 1991=293+419+1279
2550 64 13 1991=313+659+1019
2600 66 15 1991=331+647+1013
2650 68 26 1991=347+787+ 857
2700 71 3 1991=359+401+1231
2750 73 9 1991=373+587+1031
2800 76 4 1991=389+421+1181
2850 78 16 1991=401+599+ 991
2900 80 22 1991=419+733+ 839
2950 83 2 1991=433+461+1097
3000 86 10 1991=449+613+ 929
3050 89 7 1991=463+617+ 911
3100 92 10 1991=487+683+ 821
3150 96 7 1991=509+643+ 839
3200 101 5 1991=557+613+ 821
3250 107 2 1991=593+641+ 757
3260 108 8 1991=599+691+ 701
3274 113 1 1991=619+653+ 719
3281 116 1 1991=643+647+ 701
3282 117 1 1991=647+653+ 691
3283 117 2 1991=647+661+ 683
3284 118 1 1991=653+661+ 677
3285 119 1 1991=659+659+ 673

总之，T(1991)=3285，是千真万确的，不多也不少。

其它结果有：Y(1993)=3182，T(1995)=2029，T(1997)=3309。含素数因子“3”的奇数，(1+1+1)的解数反而少。

华罗庚公式：

$$T(N)=\frac{N^2}{2L^3}C(N)+O(N^2L^{-4}LogL)$$

也是偏大了。

按华罗庚公式，N=1991 计算：

$$T=(1991)=10299.621$$

这 10299.621 是由主项算出的，如果连“O 项”计入，偏差就更大了。如用(经验公式)：

$$T(N)=\frac{[\Pi(N)]^2}{3L}$$

算

$$T(1991)=\frac{300^2}{3\times 7.5963923}$$

$$=3949.2431$$

会误差小些!!

快速排序“快”在何处?

四川万县师专　应　宏

排序是计算机处理非数值问题的一类重要运算。据有人统计，计算机整个运行时间的 25%以上是花在数据的排序上。显然，排序算法的效率高低是至关重要的，人们为设计高效率的排序算法作了许多努力。1962 年，由 Hoare 提出的快速排序(quick sort)算法，对批量的数据元素进行排序时，它比简单而直观的选择排序、插入排序、冒泡排序效率高、速度快。“快”的原因何在？要回答这个问题，我们先给出关于序列 $L=(a_1a_2a_3\cdots a_n)$ 逆序的概念。

序列 L 的一个逆序，是指存在 i,j($1\leqslant i,j\leqslant n$)使得 $i<j$ 有 $a_i>a_j$。无疑，序列 L 至多可以有 $n(n-1)/2$ 个逆序。

如下结论也是显然的：

假设有 $a_i,a_j\in L$，若(a_i,a_j)构成逆序，且在 a_i,a_j 之间有 k 个元素满足 $a_j<x<a_i$，有 r 个元素满足 $x=a_i$，有 s 个元素满足 $x=a_j$，其余的元素满足 $x<a_j$ 或 $x>a_i$，则将 a_i,a_j 对换后，序列 L 的逆序数减少 2k+r+s+1。

例如，有如下序列：

25 8 59 7 20 45 18 …… ①

按非递减顺序(后同)，序列①中共存在 11 个逆序，显然，(25,18)构成逆序。若将 25,18 两元素对换则得到序列

18 8 59 7 20 45 25 …… ②

②中还存在 8 个逆序，减少的逆序数正是前面结论所指出的数目。若序列 L 中相邻两数据元素构成逆序，则将此两元素对换后，序列 L 就减少一个逆序。

要对 N 个数据元素进行排序(不妨设 N 个元素存储于向量 A〔1…N〕中)，实质上就是要消除 A〔1〕，A〔2〕，…A〔N〕中的所有逆序(相对某种顺序而言)。试图提高排序的效率，就应着眼于如何提高消除逆序的效率。冒泡排序算法之所以不是高效率的排序算法，原因在于每次对换只在相邻元素之间进行。下面是对序列①施行冒泡排序算法第一轮的过程：

25 8 59 7 20 45 18

8 25 59 7 20 45 18
8 25 7 59 20 45 18
8 25 7 20 59 45 18
8 25 7 20 45 59 18
8 25 7 20 45 18 59 …… ③

经过第一轮的五次对换，得到的序列③中还有 6 个逆序，显然每对换一次仅消除一个逆序。根据前面的结论，如果允许对换不只是在相邻元素之间进行，那么每对换一次就有可能消除多个逆序。由 Hoare 提出的

快速排序算法就允许一个元素可以跳越几个元素的位置而移动，即不相邻的两个元素之间进行对换，从而可以消除多个逆序，达到提高排序效率的目的。

下面是对序列①施行快速排序算法第一轮的过程：

```
25  8  59  7  20  45  18   25送x,18<x
↑i                  ↑j   移动18,修改i
18  8  59  7  20  45  □    8<x,修改i
   ↑i               ↑j
18  8  59  7  20  45  □    59>x,移动59
      ↑i               ↑j  修改j
18  8  □   7  20  45  59   45>x,修改j
      ↑i       ↑j
18  8  □   7  20  45  59   20<x,移动20
      ↑i       ↑j          修改i
18  8  20  7  □   45  59     7<x,修改i
         ↑i ↑j
18  8  20  7  □     45  59  i=j,x送i位置
            i↑ ↑j
18  8  20    7  25  45  59 ……①
```

上面方框表示元素移动后的位置(空位)。经过第一轮的四次移动，得到的序列④中仅存在4个逆序。不难发现，算法中每次填补空位的方法实际上反复实现了两个非相邻元素(且构成逆序)之间的对换。从而使得一次对换消除了多个逆序。

除快速排序算法外，1959年Shell提出的缩小增量法(即Shell Sort)，1964年由Williams首先提出而经Floyd改进的堆排序(Heap Sort)算法，效率也都比冒泡排序高。仔细分析Shell排序算法，它不仅由于移动一个元素可以消除多个逆序，而且还由于每次采用新的增量排序，并不破坏老的增量已有的排序效果。至于堆排序，每次从堆顶删除时，空位的移动也是对构成逆序的非相邻元素进行对换，使得一次对换可能消除多个逆序。因此，在设计排序算法时，我们总是希望为消除逆序而进行对换的两个元素之间的距离能够拉开，以提高消除逆序的效率。快速排序、Shell排序和堆排序都运用了这一思想，只不过Shell排序算法在运用这一思想时表现得更典型、更直截了当。

两个高效率的排序程序

武汉水利电力学院教务处　李军杰

应用最广泛的快速排序法，是目前以比较为基础的内部排序中速度最快的方法，它的平均时间复杂度为$0(n\log_2 n)$。该算法可以写成相当短的递归形式(绝大多数书籍都这样处理，采用类PASCAL或PASCAL描述其算法)。若把它写成非递归算法，程序会变得较复杂，而且对于那些不熟悉如何把递归化为非递归以及用不支持递归的高级语言(如：BASIC、FORTRAN等)编程的用户，在工作中想要应用快速排序法就较困难。笔者在理解了快速排序法实质的基础上，经过上机实践和改进，编成了形式较为简洁的非递归BASIC排序程序(见程序1)。该程序以独立的子程序形式出现，用户可随处调用。在应用中，首先定义数组X的大小，不妨设为N，并为X(N)赋初值；二维数组S(栈)相应为S(2*LOG(N),2)。程序1中110句的L、U为数组X的下界、上界值，用户可加以修改。

堆排序法是一个既在平均情况下，又在最坏情况下，都是时间复杂度为$0(n\log_2 n)$的有效算法。此法的优点在于不占用额外的存储空间。程序2是经过优化处理的堆排序程序。数组B(N)为待排序数组，事先应赋值。语句90－150用于建堆，语句160－250用于排序和重建堆，直至B成为有序数组。

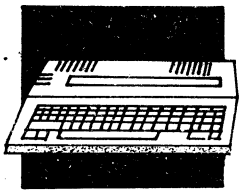
以上两个程序均在PC机上运行通过，对大量数据进行排序时，效果极佳。

程序1

```
100 'Quicksort for BASIC
110 L=1:U=100:P=0
120 I=L:J=U:F=-1
140 IF X(I)<=X(J) THEN 170
150 SWAP X(I),X(J):F=-F
170 IF F=1 THEN 200
180 J=J-1:GOTO 210
200 I=I+1
210 IF I<J THEN 140
220 IF I+1>=U THEN 250
230 P=P+1:S(P,1)=I+1:S(P,2)=U
250 U=I-1
260 IF L<U THEN 120
270 IF P=0 THEN 310
280 L=S(P,1):U=S(P,2):P=P-1:GOTO 120
310 RETURN
```

程序二

```
10 'Heapsort for BASIC
50 DEFINT F,I,J,K,N
90 FOR I=1 TO N
100 J=1
110 WHILE J>1
120 F=INT(J/2):IF B(F)>=B(J) THEN 150
130 SWAP B(F),B(J):J=F
140 WEND
150 NEXT I
160 I=N
170 WHILE I>1
180 SWAP B(I),B(1):I=I-1
190 J=1:K=J+J
200 WHILE K<=I
210 IF (K<I AND B(K)<B(K+1)) THEN K=K+1
220 IF B(J)>=B(K) THEN 250
230 SWAP B(J),B(K):J=K:K=J+J
240 WEND
250 WEND
```



学习机之友

CEC—I 高级动画制作技术

四川雅安教育学院微机室　黄一禾

在 CEC—I 或 APPLE Ⅱ 上设计生动逼真的动画，常感到有几个困难，一是绘图麻烦，二是运动困难，三是内存开销大。虽然制作动画的方法很多，但不少常见的方法都或多或少存在上述某方面的不足。下面将结合实例介绍较先进实用的动画制作技术，其主要点是利用工具软件绘制运动分解图，并运用预移图序法原理剪辑成动画分解图，最后利用汇编语言程序显示这些分解图，形成动画，这种动画制作技术的主要优点是绘图方便、生成的动画动感强、逼真、速度很快，而且占用内存较少。缺点是运动显示程序编写较麻烦。

一、预移图序法原理

我们知道，动画电影是由一幅幅分解动作的图画连续高速放映后生成的。如果我们在计算机上将动画活动过程的主要瞬间变化的图象预先画好，然后按顺序快速显示出来，同样可产生动画效果。如图一（文后）有七幅长 28 个点、宽 10 个点的不同瞬间状态的直升飞机分解图，在每幅图中，仅主螺旋桨和尾螺旋桨的位置不同，若在高分辨显示页的相同位置上反复依次快速将这七幅图形显示出来时，就会出现一驾悬停在空中，其螺旋桨不停旋转的直升飞机，显然这仅仅是动画的一部分，即物体自身在不停的变化。

现在主要讨论在物体自身不断变化的同时，还要作上下左右运动的动画。物体的上下运动比较简单。根据 CEC—I 高分辨显示页的结构，只须在原位置擦去图形，随即在上或下一点的新位置上又将其显示出来，物体就上下运动了。注意：在每一显示位置上都需要将七幅分解图依次显示一遍，以产生自然平滑的动感。

物体的左右运动比较麻烦，主要原因是 CEC—I 屏幕上任一行的第 7k 至第 7k＋6（k＝0，1，……39）的 7 个点，用内存中的一个字节来表示，即高分辨显示页中每个字节的第 0—6 位分别代表屏幕上某行中从左到右的 7 个点。如果直接将物体分解图的显示位置左移或右移一个字节，那么显示出的图形实际上向左移或向右移了 7 个点。这样虽然是动起来了，但有较强的跳跃感和闪烁感，明显失真。为了解决这一问题，可以模仿动画电影画面安排物体位置的方法，把物体安排在分解图中的不同位置上，（注意：分解图均是大小完全相同的矩形图，如上例中为 28×10 点），即在第一幅分解图中将物体最左边的点安排在矩形左起第二列，（这里留出一列空白点以利显示），在第二幅分解图中将物体最左边的点安排在矩形左起第三列，……，在第七幅分解图中将物体最左边的点安排在左起第八列，也就是将每一幅图形中的物体安排在相差一个点的位置上，见图 2 中的七幅直升飞机分解图。这样在相同位置上，从 1 号图依次显示到 7 号图时，虽然显示位置没变，但直升飞机却前进了七个点，刚好一个字节，接着将显示位置右移一个字节，又将这七个分解图依次显示一遍，……。于是就得到了一幅直升飞机在不断前进，其螺旋桨也在不停旋转的动画画面。这种动画没有闪烁感和跳跃感，平滑自然，非常逼真。同样，若从 7 号图形到 1 号图形显示一遍后，显示位置左移一个字节，再在新位置上反序显示一遍，……，就会得到直升机不断后退的动画画面。

概括起来，在同样大小的矩形内，绘制出形状基本相同的物体，并使这些物体的起始点依次相差一个点，形成若干动画分解图，调用时在同一位置上依次将分解图显示后，再在新位置上依次显示（即向左或向右移一个字节），不断重复下去，就会产生出物体本身在不断变化，同时又在不断向左或向右运动的动画效果，这种产生动画的方法一般称为预移图序法。

二、图形绘制和剪辑

图形绘制和剪辑历来比较麻烦。为了绘制出符合上述动画要求的分解图，一般是利用专用绘图软件，这使得绘图和剪辑变得十分方便。本文主要使用 A. C. E 软件的绘图系统（1 号盘）作为制作分解图形的工具。利用 A. C. E 软件绘图是十分方便的，这里不再加以介绍，下面主要谈谈有关剪辑和转换等问题。

将所需物体图形绘制完毕后，进入绘图系统的窗口剪裁，将物体图形剪裁下来，操作时实际上就是用一个矩形方框将图形框起来，得到的矩形图块就是前面所说的分解图了。在确定矩形分解图左右两条边界线时，应按预移图序法的思想设取，即第一个分解图左边界线应与物体最左边的点相隔一个空白点；右边界线应与物体最右边点相隔 7 个空白点。第二个分解图左边界线与物体最左边点相隔 2 个点，右边界线与物体最右边点相隔 6 个空白点；……第七个分解图左边界线与物体最左边点相隔 7 个点，右边界线与物体最右边点相隔 1 个点。矩形分解图的上下边界线始终不变，这样所截取的各分解图完全一样大，便于快速显示，图 2 中的七幅直升飞机分解图就是按上述方法截取的。

在剪裁时还应注意，当剪裁好一个分解图后，应立即存盘，并给分解图文件编上号，以示区别，然后再剪裁下一个。在 A. C. E 系统中，这类文件是以窗口图形文件存取的，文件类型为 G，文件名加上前缀 WI. 。这类图形文件还不能直接在 DOS 3. 3 系统下调用，还要进行转换，转换方法很简单，步骤如下：

①引导 DOS 3. 3 系统，键入 POKE 44272，4↙。

②插入文件盘，键入 LOCK WI. f↙（f 是窗口文件名）。此命令执行后，磁盘中的 G 类文件 WI. f 已转换为可在 DOS 3. 3 系统下调用的 B 类图形文件了。

每一个窗口图形文件经过转换后，其首地址和长度均未变，如果窗口图形的大小不同，则文件的首地址

和长度也将不同，但其末地址始终是一致的，均为$7FFF。应该指出，每个窗口文件的最末8个字节是文件的描述项，不是图形数据，其中：

$7FF8单元为图形纵向长度，以点为单位。

$7FF9单元为图形横向长度，以7个点为单位，例如若($7FF9)=4，则图形的实际横向长度为4×7=28个点。

$7FFB单元为图形数据首地址(低位)。

$7FFC单元为图形数据首地址(高位)。

$7FFD单元为图形文件长度(低位)。

$7FFE单元为图形文件长度(高位)。

为了调用方便，一般应将这些分解图形文件合并起来，形成一个完整的图形文件。本文后面所附数据中$6000～$6117就是图一的七幅分解图合并而成的，其中每幅分解图的首地址位于$6334～$6341。$6118～$622F就是图二的七幅分解图合并而成的，其中每幅分解图的首地址位于$6326～$6333。(低位在前，高位在后)。

至此，各分解图形的绘制、剪辑、转换和合并工作全部完成。当然，读者也可以而选用其它自己较熟悉的方法来得到物体的运动分解图，同样也可以不用移点剪裁分解图的方法得到预移分解图，而用程序将统一剪裁得到的数据编辑成预移数据。

三、动画的实现

有了动画预移分解图后，动画的实现就是将分解图依次快速地在屏幕上显示出来。由于客观物体的活动过程各不相同，因此，实现动画的程序也各不相同，本文仅给出一种较简单易行的实现动画程序。

1. $627A～$629F，直升机从左向右飞行程序。程序在初始位置依次显示图二中的七幅分解图后，再在新位置上重复显示。新位置的纵坐标保持不变，横坐标为旧位置加1个字节。

$627E单元为横向起始坐标值。

$6283单元为直升机横向飞行分解图形数据起始地址表地址低位(高位为$63)。

$6292单元为行数据个数，即每行有4个数据。

$6307此子程序将生成高分辨显示地址表并进入和清除高分辨第二页。

2. $62A0～$62B0，直升机悬停在空中程序。此程序将图1中七幅分解图反复显示。

$62A4单元为直升机悬停分解图形数据地址表地址低位(高位为$63)。

3. $62B1～$6306直升机向上飞行程序。

$62B5单元为横向起始坐标初值。

$62C7单元为纵向起始坐标初值。

$62D3单元为列数据个数，即每列有10个数据。

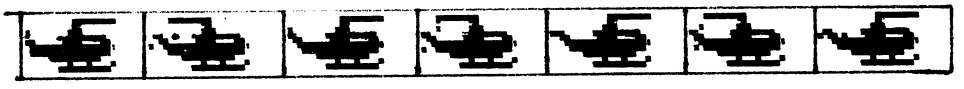
$62E7子程序将清除刚才显示出的一个直升机分解图，为下一步在新位置显示作好准备。

上述3个程序都直接调用了程序$6230，此程序是动画显示的核心程序。

4. $6230～$6279在相同位置依次显示图一或图二中的各分解图子程序。

此子程序由三层循环组成，最外层从$6234～$6278，共循环执行7次，每次显示一个分解图。第二层从$6249～$626B，共循环执行10次，每次将分解图的一行显示数据送显示地址显示。以层循环结束后，程序调用了延时子程序，以延长显示停留时间，使动画更清楚自然。第三层为$6255～$6263，共循环4次，每次将分解图的一个数据送显示地址显示。在这个子程序中用到的一些初值，大都由程序1,2,3所设定。

由于本例中的各分解图形数据量较小，仅10×4个。在程序4中采用了X寄存器变址取得图形数据的方式设计。因此，本例程序只适用于单个分解图形数据量在1页(256个数据)范围内的动画演示，但程序只要稍加修改，即可处理单个分解图形数据量大于256个的动画演示了。



图一

图二

```
6000— 00 60 7F 00 00 20 00 00
6008— 04 70 00 00 06 7C 1F 00
6010— 0C 78 30 00 78 7F 7F 00
6018— 78 7F 3F 00 00 7F 0F 00
6020— 00 10 42 00 00 7C 3F 00
6028— 70 3F 00 00 00 20 00 00
6030— 00 70 00 00 26 7C 1F 00
6038— 0C 78 30 00 79 7F 7F 00
6040— 70 7F 3F 00 00 7F 0F 00
6048— 00 10 42 00 00 7C 3F 00
6050— 00 60 7F 00 00 20 00 00
6058— 02 70 00 00 06 7C 1F 00
6060— 0C 78 30 00 78 7F 7F 0Q
6068— 70 7F 3F 00 00 7F 0F 00
6070— 00 10 42 00 00 7C 3F 00
6078— 70 3F 00 00 00 20 00 00
6080— 10 70 00 00 06 7C 1F 00
6088— 0C 78 30 00 78 7F 7F 00
6090— 72 7F 3F 00 00 7F 0F 00
6098— 00 10 42 00 00 7C 3F 00
60A0— 00 60 7F 00 00 20 00 00
60A8— 00 70 00 00 07 7C 1F 00
60B0— 0C 78 30 00 78 7F 7F 00
60B8— 70 7F 3F 00 00 7F 0F 00
```

```
60C0— 00 10 42 00 00 7C 3F 00
60C8— 70 3F 00 00 00 20 00 00
60D0— 08 70 00 00 06 7C 1F 00
60D8— 0C 78 30 00 78 7F 7F 00
60E0— 74 7F 3F 00 00 7F 0F 00
60E8— 00 10 42 00 00 7C 3F 00
60F0— 00 60 7F 00 00 20 00 00
60F8— 00 70 00 00 06 7C 1F 00
6100— 2D 78 30 00 78 7F 7F 00
6108— 70 7F 3F 00 00 7F 0F 00
6110— 00 10 42 00 00 7C 3F 00
6118— 00 60 7F 00 00 20 00 00
6120— 04 70 00 00 06 7C 1F 00
6128— 0C 78 30 00 78 7F 7F 00
6130— 78 7F 3F 00 00 7F 0F 00
6138— 00 10 42 00 00 7C 3F 00
6140— 60 7F 00 00 00 40 00 00
6148— 00 60 01 00 4C 78 3F 00
6150— 18 70 61 00 72 7F 7F 01
6158— 60 7F 7F 00 00 7E 1F 00
6160— 00 20 04 01 00 78 7F 00
6168— 00 00 7F 03 00 00 01 00
6170— 08 40 03 00 18 70 7F 00
6178— 30 60 43 01 60 7F 7F 03
6180— 40 7F 7F 01 00 7C 3F 00
6188— 00 40 08 02 00 70 7F 01
6190— 00 7F 03 00 00 00 02 00
6198— 00 01 07 00 30 60 7F 01
61A0— 60 40 07 03 40 7F 7F 07
61A8— 10 7F 7F 03 00 78 7F 00
61B0— 00 00 11 04 00 60 7F 03
61B8— 00 00 7C 0F 00 00 04 00
61C0— 00 00 0E 00 70 40 7F 03
61C8— 40 01 0F 06 00 7F 7F 0F
61D0— 00 7E 7F 07 00 70 7F 01
61D8— 00 00 22 08 00 40 7F 07
61E0— 00 7C 0F 00 00 00 08 00
61E8— 00 02 1C 00 41 01 7F 07
61F0— 00 03 1E 0C 00 7E 7F 1F
61F8— 00 7D 7F 0F 00 60 7F 03
6200— 00 00 44 10 00 00 7F 0F
6208— 00 00 70 3F 00 00 10 00
6210— 00 00 38 00 00 03 7E 0F
6218— 00 16 3C 18 00 7C 7F 3F
6220— 00 78 7F 1F 00 40 7F 07
6228— 00 00 08 21 00 00 7E 1F
6230— A9 00 85 03 0A A8 B9 34
6238— 63 8D 58 62 C8 B9 34 63
6240— 8D 59 62 A2 00 A9 00 85
6248— 02 A4 02 B9 08 64 85 05
6250— B9 48 63 85 04 A0 10 BD
6258— 78 60 EA EA 91 04 E8 C8
6260— C0 14 90 F3 E6 02 A5 02
6268— C9 0A 90 DD A9 40 20 A8
6270— FC E6 03 A5 03 C9 07 90
6278— BB 60 20 07 63 A9 FF 8D
6280— 56 62 A9 26 8D 37 62 8D
6288— 3E 62 EE 56 62 AD 56 62
6290— 18 69 04 8D 61 62 C9 28
6298— B0 09 20 30 62 4C 8A 62
62A0— 20 07 63 A9 34 8D 37 62
62A8— 8D 3E 62 20 30 62 4C AB
62B0— 62 20 07 63 A9 10 8D 56
62B8— 62 A9 14 8D 61 62 A9 34
62C0— 8D 37 62 8D 3E 62 A9 70
62C8— 8D 46 62 CE 46 62 AD 46
62D0— 62 18 69 0A 8D 69 62 C9
62D8— 0B 90 09 20 30 62 20 E7
62E0— 62 4C CB 62 4C A3 62 A9
62E8— 51 8D 5A 62 A9 04 8D 5B
62F0— 62 A9 01 8D 6D 62 20 43
62F8— 62 A9 EA 8D 5A 62 8D 5B
6300— 62 A9 40 8D 6D 62 60 A9
6308— 40 85 E6 A0 00 98 48 20
6310— 11 F4 68 A8 A5 26 99 48
6318— 63 A5 27 99 08 64 C8 C0
6320— C0 D0 EA 4C D8 F3 18 61
6328— 40 61 68 61 90 61 B8 61
6330— E0 61 08 62 00 60 28 60
6338— 50 60 78 60 A0 60 C8 60
6340— F0 60 00 00 00 00 00 00
```

警察抓小偷

宁夏　中卫县中学高一(2)班　李明

假如你是一名警察，现有一个小偷蹿上了一座六层楼，你必须在他跑到楼顶之前抓住他，不然他就会驾上准备好的飞机逃走。在楼道里，布满了障碍物，小偷碰到障碍物会一跃而过，而你遇到障碍物则必须绕道而行。若在小偷跑到楼顶之前你没有抓住他，那你就重新玩这一局；若你抓住了他，就会听到一阵动听的音乐，计算机也会对你作出一次评价，并询问你是否继续玩。若不玩了，则按"N"回车；若继续玩，只要回车就进入难度更大的下一局。玩过第三局后，计算机就会告诉你，你是一名合格的警察了。

游戏难度逐渐增大，可以照顾到各个层次的人，并伴有英文提示和逼真的音响，使人仿佛置身其中，极能锻炼人的反应能力，以及眼、手、脑的的协调能力。

说明：↑、↓、←、→分别控制警察上、下、左、右移动，颜色暗的为小偷，颜色亮的为警察。当警察超过了每层楼的最左端时，就会跑到上一层楼的最右端。

```
5 A$(1)="GOOD":A$(2)="BETTER":A$(3)="EXCELLENT":FOR T=1 T0 3
10 GR:COLOR=3:FOR I=2 T0 38 STEP 6:HLIN 0,39 AT I:HLIN 0,39 AT I+1:NEXT
20 FOR I=1 T0 85+T*25: PLOT RND(1)*39,RND(1)*37+2:NEXT
25 HOME:VTAB 21:PRINT"  LEVEL";T,"TIMES=";S:GET A$
30 A=39:B=37:M=5:N=37
40 COLOR=X:PLOT M,N:M=M-.4-T/10:IF M<0 THEN N=N-6:M=39:IF N<2 THEN S=S+1:GOTO 10
60 X=SCRN(M,N):COLOR=4:PLOT M,N
70 Q=A:W=B:P=PEEK(49152)-128:A=A+(P=21)-(P=8):B=B+(P=10)-(P=11):IF A<0 THEN A=39:B=B-6
80 IF SCRN(A,B)=4 THEN 110
90 IF SCRN(A,B)=3 THEN A=Q:B=W:GOTO 40
100 COLOR=7:PLOT A,B:COLOR=0:PLOT Q,W:MUSIC 1,2:GOTO 40
110 FOR I=1 TO 90:MUSIC I,10:NEXT:PRINT"YOU ARE";A$(T);"!":IF T=3 THEN PRINT "YOU ARE A QUALIFIED POLICEMAN"
120 INPUT "CONTINUE? (Y/N)";A$:IF A$<>"N" THEN NEXT T:RUN
```

ProDOS 磁盘操作系统入门(续)

廖 凯

第四章 ProDOS 实用程序

为 APPLE II 机系列设计的 ProDOS 实用程序有四套:ProDOS USERS. DISK、ProDOS COPY Ⅱ PLUS、APA 和 SYSTEM UTILITIES。前三套可在任何 APPLE 机上运行,而后者是为 APPLE Ⅱc 设计的,只能在 APPLE Ⅱe、Ⅱc 和 Ⅱ GS 上运行。

在 ProDOS USERS. DISK 上有两个很有用的实用程序:FILER 和 CONVERT。这些程序将允许用户对磁盘和文件进行管理。

一、FILER

在启动 USERS. DISK 进入主菜单后,按 F 便进入 FILER 菜单。FILER 主要分为三部分,其中 FILE 和 VOLUME 是最常用的部分。FILE 部分是对文件进行管理,VOLUME 部分是对磁盘进行管理。FILER 的每一部分都有一个简要的说明(即 TUTOR,按? 键)。

使用说明:在 FILER 中使用文件名时,必须以字母开头,最多 15 个字符(包括句号);路径名不能超过 64 个字符(包括斜线)。文件名必须用斜线分开,斜线需放在卷名后面,否则 FILER 将把文件名当作部分路径名。你可以用右和左箭头键对屏幕上的资料进行编辑。你并可随时按 Esc 键返回原位置或回到 FILER 主菜单。

1. VOLUME COMMANDS

这部分有七项功能。要想进入各项功能,只需按前面的字母即可。

FORMAT A VOLUME: 此功能是格式化一张 ProDOS 格式磁盘,用户只要依提示给出槽口号和驱动器号及卷名(磁盘名),或直接按回车接受系统设定的值。如果磁盘已格式化过,系统会问你是否再格式化,在格式化的过程中,磁盘上的数据将被清除。

COPY A VOLUME: 此功能是进行整盘拷贝。用户依提示输入源盘和目的盘的驱动器号和槽口号,也可直接按回车接受系统设定的值。系统会读取源盘的卷名作为目的盘的卷名,你可以按回车接受系统设定或输入自定的新卷名。如果目的盘已格式化过,系统会问你是否取消原卷名。在拷贝的过程中,如果磁盘有问题,系统将有提示。

LIST VOLUMES: 此功能将列印目前所有存在的磁盘的卷名。

RENAME A VOLUME: 此功能允许用户改变指定磁盘的卷名。在目前的卷名被显示后,你可输入/(斜线)和新的卷名,斜线必须放在卷名前面。

DETECT BAD BLOCKS: 此功能允许用户检查磁盘的情况。如果磁盘损坏或有软损伤,屏幕将显示坏块号,并将其删除。

BLOCK ALLOCATION: 此功能将显示在磁盘上块的使用情况。

COMPARE VOLUMES: 此功能允许用户对两个磁盘的资料进行比较,并显示不符的块号。

2. FILE COMMANDS

此项目允许用户在任何磁盘设备上对文件进行管理。在此项目中,可使用通配字符(WILDCARDS)即"="和"?"来代替文件名。"?"字符将提示每个文件,用户可以回答 Y 或 N 来选择文件,"="字符是表示所有的文件。在使用此项之前,用户应该对目录、路径名、文件名和部首有所了解。此部分有七项功能,要进入各功能,只要键入前面的字母即可。

LIST PRODOS DIRECTORY: 此功能将显示目录、文件名、文件类型、文件长度、修改日期和块分配的情况,相当于 CAT 命令。

COPY FILES: 此功能允许用户在任何磁盘设备之间拷贝文件。在拷贝过程中,如果在目的盘上有与源盘相同的文件名时,那么系统将提示用户是否需要重拷(复盖原文件)。另外,在拷贝单个文件时,用户可以更换文件名。

DELETE FILES: 此功能相当于 DELETE 命令。在删除一个完整的子目录时,使用通配字符是很方便的。在删除一个目录文件之前,必须先删除目录内的所有文件。

COMPARE FILES: 用于将两个文件的资料进行比较。在拷贝文件之后,可用此功能检验文件的准确性。

ALTER WRITE-PROTECTION: 此功能相当于 LOCK 和 UNLOCK 命令。用户可依提示对文件进行加锁或解锁操作。

RENAME FILES: 用于更换文件名和目录,相当于 RENAME 命令。

MAKE DIRECTORY: 用于在卷目录内建立副目录,相当于 CREATE 命令。注意在卷目录下只能存放 51 个文件。

SET PREFIX: 用于设置路径名。这只允许用户输入文件名或部分路径名以处理其它的文件命令。在文件被装入时,会自动地设置当前的部首。斜线必须放在卷名和目录的前面。如果要加一个副目录到现有的路径名时,可用右箭头键移动光标到行的尾端并开始输入。

3. CONFIGURATION DEFAULTS

此项目是用于设置磁盘驱动设备和选择信息输出的设备。

SELECT DEFAULTS: 此功能是设置当前你的系统。用户通过提示设置源盘和目的盘的驱动器号及槽口号以及显示器和打印机。系统设定打印机的槽口号为 1,

如果打印接口卡在其它的槽口上，那么就必须正确地设定槽口号。

RESTORE DEFAULTS： 此功能是恢复系统设定值。

二、CONVERT

这程序是用于将文件在 ProDOS 和 DOS 3.3 之间进行转换。具有保护的文件和随机存取文件不能被转换。由 ProDOS 主菜单按 C 进入此项目。屏幕前三行显示当前的状态：箭头指出转换的方向，日期可设定，部首(PREFIX)指出当前的路径名。

1. REVERSE DIRECTION OF TRANSFER

此功能是改变传输的方向，由 DOS 到 ProDOS 或由 ProDOS 到 DOS。

2. CHANGE DOS 3.3 SLOT AND DRIVE

允许用户改变 DOS 3.3 的槽口号的驱动器号。

3. SET PRODOS DATE

此功能是设置系统的日期。

4. SET PRODOS PREFIX

此功能是设置路径名。在任何文件被转换之前，必须设置正确的部首。你可通过路径名或驱动器号和槽口号来选择部首。

5. TRANSFER(OR LIST) FILES

此功能是转换或列印文件。经过前四项的设定之后，就可以进行转换或列印了。你可用左右箭头键来选择文件，按空格键即开始转换。你可用通配字符"?"来转换单个文件，或用"="转换所有文件。

在转换的过程中，DOS 3.3 文件名可能包含有控制字符、空格和其它标点符号，它们不能作为 ProDOS 文件名，ProDOS 将自动地在无效字符的位置上设置一个句号。某些 DOS 3.3 磁盘有反相显示的文件名，ProDOS 将不能转换这些文件。

三、ProDOS COPY Ⅱ PLUS

这是最新的多功能磁盘管理软件，它不仅可以管理 ProDOS 文件，而且也可以对 DOS 3.3 文件进行管理，并具有显示小写字母的功能。其版本有许多种，功能最多最强的版本是 ProDOS COPY Ⅱ PLUS 9.0。其用法与 DOS 3.3 的 COPY Ⅱ PLUS 最新版本相似，只不过将"SECTOR EDITOR"这一功能放在 BIT COPY 中了。用法这里不再赘述，请参考有关资料，如有需要者，请与本人联系。

四、APPLESOFT PROGRAMMER'S AID(APA)

这是用于 ProDOS 的程序设计辅助程序。它存放在 ProDOS EXAMPLES 磁盘上，它具有的功能是：自动产生行号、计算程序的长度、进行十进制与十六进制数的转换、删除 REM 语句、显示控制字符、列印变量、合并程序、重编行号、保护内存程序、中止自动行号、删除控制字符。APA 命令只能在立即执行方式下执行。

APA 命令集：

1. AUTO 和 MANUAL

格式：AUTO [开始行号],[增量]

MANUAL

功能：AUTO 命令是自动产生行号。行号范围为 0 至 63999。每当需要新行号时按空格即可。在输入程序时，如果想重新输入该行，只要按 CTRL－X，再按空格即可。MANUAL 命令是取消自动行号功能。

2. LENGTH

格式：LENGTH

功能：此命令是以字节数来显示内存程序的长度。长度以十进制和十六进制数表示。

3. CONVERT

格式：CONVERT [十进制或十六进制数]

功能：在执行该命令后跟一个十进制数后，将显示相对应的十六进制数，反之亦然。十六进制数用 $ 表示。

4. COMPRESS

格式：COMPRESS

功能：此命令用于删除程序内的所有 REM 语句，以加快执行速度和节省内存。

5. SHOW 和 NOSHOW

格式：SHOW

NOSHOW

功能：SHOW 命令可将程序内所有隐含的控制字符显示出来。输入 SHOW 并列印程序，控制字符以反相方式显示。NOSHOW 命令是取消 SHOW 命令。

6. HOLD 和 MERGE

格式：HOLD

MERGE

功能：使用这两个命令可合并程序。先将第一个程序装入内存，用 HOLD 命令将程序放在 HIMEM 内，以防重写，再将第二个程序装入内存，用 MERGE 命令将第一个程序释放，这样两个程序已合并在一起。如果存在相同的行号，系统会把重复行号列出，供你选择。

7. RENUMBER

格式：RENUMBER [始行号],[增量],[首行号]
,[末行号]

功能：此命令用于将整个程序或部分程序进行重编行号。

8. XREF

格式：XREF

功能：是用于将程序内所有的变量及所在的行号一起显示出来。此命令将分类显示每个变量，变量分为五种类型：

$	串变量	%	整型变量
$(	下标串变量	%(	下标整型变量
(	下标变量		

9. EXIT

格式：EXIT

功能：此命令是用于将 APA 从内存中删去。

四、SYSTEM UTILITIES

这是一个只适合于 APPLE IIe、IIc 和 IIGS 的系统实用程序，它具有很强的管理功能，但由于此软件使用的范围太窄，故在这里不予介绍，如有需要者，请与本人联系。

(未完待续)

高性能的 CEC-I 自定义组合键

无锡柴油机厂设计科　杨勇佳

CEC-I 缺乏自定义组合键功能，给操作者带来诸多不便。为了弥补这个不足，笔者设计出一种自定义组合键的方法，在几乎不影响原来键盘功能的情况下，一次最多可定义组合键 96 个，每个组合键可容纳 250 个字符，并能方便地更换组合键内容。

本方法首先修改了 DOS 的输入拦截指针（$AA55 和 $AA56 单元），当键盘上有键输入时，先转入程序一进行新扩展的键盘分析和外理，首先判断该键是否新设置的辅助键（Test 键），如不是，则把该键码交回原系统显示或操作。如确认是辅助键，则系统等待键盘上的再次键入，并根据这次的键值作“组合键”判读处理。如不是组合键就返回原系统处理，如是，则显示该组合键的内容，再返回原系统操作。

CEC- I 机系统决定了 F1～F5 键、Quit 键、光标键、SHIFT、Ctrl、Reset、ESC 诸键、中文键、西文键、Capslock 和 Return 这 18 个键不能用来定义组合键。除此以外，每次只按二次键即可获得组合键功能的键共有 48 个，而按三次键（其中一次为 SHIFT 或 Capslock 键）可获得功能的键亦可有 47 个键，再加上辅助键本身亦可被定义，故最多可定义 96 个组合键。

在 DOS 引导后，用 CALL－151 进入监控，无误地键入程序一，并用 BSAVE DYJ,A $6000,L $89 存盘，然后返回 BAS1C，再进入中文状态，键入程序二。从 1000 行开始用 DATA 语句成对输入要定义的键和组合键的内容。在全部被定义键和定义内容输入后，再增加一个字符串“END”然后用 SAVE DYJ 1.0 存盘。

运行程序二，回答计算机提问的组合键定义文件简名。一般随用户的习惯输入一个短小的文件名“××”，然后由程序自动生成文件名的全名。接着显示该文件所包括的被定义键名和定义内容，并按扩展名把组合键文件记盘。这时计算机也具备了组合键功能。

如果用 CATALOG 检查磁盘，将会发现盘上除了先前存入的程序外，又新增了一个名为 DYJ－××的 B 型文件。

在本方法中，每一个被定义的键及其定义内容是按下列数据表格式放入内存的：

定义键码，定义内容长，定义内容

由于前二项各占一个字节，而各定义键的数据表是衔接的，因此内存利用率较高。

当您需要设置不同的组合键定义时，只要修改程序二的 DATA 语句，然后运行程序，并重新起名。

每次开机后，只要 BRUN DYJ－××即可形成或更换组合键定义。也可以把最常用的定义文件用 PRINT CHR $(4);“BRUN DYJ-××”加入 HELLO 程序中，以后只要一引导 DOS，随即可以得到一组常用的组合键。

本方法的自定义组合键功能，可在中、西文 BASIC 状态下使用，也可在程序运行中使用。如果您想在监控和小汇编状态下使用，可输入 6000G↙或 $6000G↙。

当使用中进行了中、西文状态转换，或按过 Reset 键，组合键暂时失效，但只要有 &↙或 6000G↙命令输入，又重新激活组合键功能。

您可以根据使用需要灵活定义组合键内容。如定义“L”为 LIST，定义“C”为 CATALOG，定义“@”键为：10 HOME:D $ =CHR $ (4):? D $;“PR # 3”:?:HGR2 等等。也可把一些常用的汉字词组定义成组合键，这等于扩充了一个最多 96 键的汉字词组输入。

作为对本系统功能的扩展，您可以在程序一中再增加一次对暂且称之为“代回车键”键值的判断和适当的程序转移，并根据需要在定义命令键时，把这个“代回车键”码放在命令字的最后。如设“代回车键”为字符“]”，定义“L”为：“LIST]”，当以后使用组合键“L”时，即可进行 LIST 操作，而不必再按回车键。

如果有兴趣也可以进行不太大的改造，在定义功能键时，按 PC 机的 CCDOS 的汉字拼音码的拼音缩写规则进行定义，如定义“Y”为“ing”，则可以在 CEC- I 机上增加了仿 PC 机的汉字拼音输入功能。

最后指出一点：$6087 和 $6088 单元的值指示了定义码表的起存地址，因此在对程序一改造时必须把这两个单元的值移放在新的地址，并把这个地址在程序一的有关字节和程序二的某些变量反映出来，否则会严重影响本自定义组合键方法的实现。

程序一

```
6000- A9 0F 8D F6 03 A9 60 8D
6008- F7 03 A9 4C 8D F5 03 AD
6011- 55 AA 85 06 AD 56 AA 85
6018- 07 A9 27 8D 55 AA A9 60
6020- 8D 56 AA 60 6C 06 00 20
6028- 24 60 85 00 C9 96 D0 2F
6030- 20 24 60 85 00 AD 87 60
6038- 85 01 AD 88 60 85 02 A0
6040- 00 B1 01 F0 1A 85 03 C8
6048- B1 01 85 04 A5 03 C5 00
6050- F0 10 A5 04 20 7B 60 A9
6058- 02 20 7B 60 4C 3F 60 A5
6060- 00 60 C8 A5 04 F0 0E B1
6068- 01 9D 00 02 20 78 60 E8
6070- C6 04 4C 62 60 4C 0C FD
6078- 6C 53 AA 18 65 01 85 01
6080- A9 00 65 02 85 02 60 89
```

```
6088- 60
```

程序二

```
10 D$=CHR$(13)+CHR$(4):PRINTD$;"PR#
   3":PRINT:HGR2:HOME
20 PRINT D$;"BLOAD DYJ"
25 PRINT"请输入组合键定义文件名:";:INVERSE:
   PRINT"  ":NORMAL
30 VTAB 1:HTAB 24:INPUT NA$
35 NA$="DYJ-"+NA$:PRINT"全名:";NA$
40 N=0:E=0:L=PEEK (24711):H=PEEK(24712):
   BB=256 *  H+L:RESTORE
50 READ N$:IF N$="END" THEN POKE BB+E,0:
   PRINT:GOTO 140
55 PRINT"被定义的第";N+1;"键=";N$
60 PRINT"其内容是:";:READ A$:PRINT A$
70 B=BB+E:LL=LEN(A$)
80 POKE B,ASC(N$)+128:POKE B+1,LL
90 FOR I=1 TO LL
100 POKE B+I+1,ASC(MID$(A$,I,1))+128
110 NEXT I
130 E=E+LL+2:N=N+1:X=FRE(0):GOTO 50
140 PRINT D$;"BSAVE";NA$;",A$6000,L";BB+
    E+1-24576
150 CALL 24576:PRINT"……OK!":END
1000 DATA"1","LIST","2","RUN","3","CATALOG",
    "END"
```

活泼的光标

吉林永吉县教师进修学校　殷瑞冬

一幅生动的屏幕画面将使人赏心悦目，倍感亲切和向往。这也是程序设计所追求的目标之一。能否一改 APPLE Ⅱ 机呆板刺眼的方块光标使之具有一种活力，为屏幕画面增添生机？在这里介绍一段子程序，应用于程序中需要键盘输入的地方，将有一个形如往复似蹲下站起的问号光标，活泼可爱，你肯定会喜欢。

```
5000 REM 光标子程序
5002 FOR 1=0 TO 1:PRINT W$(I)CHR$(8);:FOR J
=0 TO P:X=PEEK(-16384):IF X)127 THEN POKE -16368,
0:X$=CHR$(X-128):PRINT"  "CHRS(8);:RETURN
5005 NEXT:I=-(I=1):NEXT
```

在引入子程序时，需要在程序一开始的初始化部分加入下面的 10 语句，其中变量 P 是控制问号蹦跳的快慢，你可以随意调整：

```
10 DIM W$(1):W$(0)="?":W$(1)=CHR$(95):
   P=14
```

光标子程序将把你按下的键赋值给 X$ 送回。每次"GOSUB 5000"相似于执行一次"GET X$"。也正因为如此，象

```
200 INPUT"";A$
210 ……
```

的改写就需要多几条语句：

```
200 A$=""
205 GOSUB 5000:IF X$()CHR$(13) THEN
    PRINT X$;:A$=A$+X$:GOTO 205
210……
```

CEC—I 机汉字使用第十一行

北京一〇一中学高一　任义

在中华学习机汉字系统里，第十一行是作为系统提示用的，用户没法使用。我有一次在 VTAB11 定位打印时发现只在第十一行显示了一个汉字，其余的又回到前十行里显示去了。这给我一个启示，如果把汉字字符串分成若干个子字符串，一个子字符串存放一个汉字，然后分别用 HTAB、VTAB 逐个打印，这样就可以实现在第十一行显示汉字。笔者为此编了一段小程序，稍加修改就可加在用户的程序里作为子程序调用。

```
1000 A$="中华学习机汉字使用第十一行":A
     =LEN(A$):H=5
1010 FOR I=1 TO A:B$=MID$(A$,I,1):B=
     ASC(B$):IF B=127 THEN 1030
1020 HTAB H:VTAB 11:PRINT B$;:H=H+1:
     NEXT:GOTO 1040
1030 C$=MID$(A$,I,3):HTAB H:VTAB 11:
     PRINT C$;:H=H+2:I=I+2:NEXT
1040 END
```

如果想清除第十一行的汉字时，可以用下列方法：CALL 62450（清除全屏幕）或 PRINT CHR $(18)：PRINT CHR $(18)（只清除第十一行）。这里值得一提的是：打印一个子字符串后最好在程序中加一个分号，因为 PRINT A$ 后不加标点符号，机器便自动回车（屏幕上卷一行），有时会使我们在前十行中打印的字符上卷，所以大家在使用时要注意。还有一个应注意的问题是在第十一行显示汉字时，从第二十一列往后显示的汉字中，有时上半部分的字符点阵乱七八糟的，所以在第十一行显示的汉字一不能太多，二最好靠前显示。

更　正

本刊 91 年第 7 期"为无驱动器系统设置虚拟盘"一文，第 15 页程序代码中的 BDC8 单元内容应为"BD"，误排为"DD"，特此更正。

三页图形竖向并打

北京大学物理系一年级　陈钢

苹果机上高分辨率图形的分辨率只有 280×192，因此有时需要将一张较精细的图形分割制作，然后在打印时拼接起来。普通打印机在正常密度下只能实现两页图形并打，若在倍密度状态下打印又会引起比例失调。针对上述问题，我编了一段长度不到 120 个字节的机器语言小程序，将图形旋转 90°打印，实现了三页并打，并有效地解决了多张图拼接问题。

将该程序输入后，把三页图形分别调至内存 $2000－$3FFF，$4000～$5FFF，$6000～$7FFF，然后打开打印机，在监控状态下键入 8000G 或在 BASIC 状态下 CALL32768 即可。需要注意的是，不必用诸如 PR＃1 的命令接通打印机，这样就可以做到拼接打印，亦即打印完三幅图后再调入三幅图，再次运行程序就可以紧接着前三张图打印出来，中间不留空隙，因此本程序可以完成任意 n×3 张图形的拼接打印。

此程序适用于象 PC－80，RX－80，PX－100 这些普通九针打印机，要求打印卡插在一号扩充槽上。

愿此程序帮助您打印出精美的图形！

```
8000- 20 6A 80 A9 41 20 6C 80
8008- A9 07 20 6C 80 A0 27 20
8010- 6A 80 A9 4B 20 6C 80 A9
8018- 41 20 6C 80 A9 02 20 6C
8020- 80 A9 20 85 08 A2 00 8A
8028- 0A 0A 29 1C 85 07 8A 6A
8030- 6A 6A 6A 29 03 05 07 05
8038- 08 85 07 8A 6A 29 E0 85
8040- 06 6A 6A 29 18 05 06 85
8048- 06 B1 06 29 7F 20 6C 80
8050- E8 E0 C0 D0 D2 A5 08 18
8058- 69 20 10 C7 A9 00 20 6C
8060- 80 A9 0A 20 6C 80 88 10
8068- A6 60 A9 1B 2C C1 C1 30
8070- FB 8D 90 C0 60
```

电子打字机

长沙交通学院 4＃616　王蔚

本文所介绍的程序，非常之短小，却非常实用。它可以使你的计算机变成一部电子打字机，把键入的字符显示在屏幕上。还有一个突出的优点，它的运行速度非常之快。读者要是手头有本打字训练的指导手册，很快便可以成为打字高手。程序清单如下：

```
300- 20 58 FC      JSR $FC58
303- 20 0C FD      JSR $FD0C
306- 20 ED FD      JSR $FDED
309- 4C 03 03      JMP $0303
```

读者可在监控或小汇编下键入程序。运行时可在监控状态下键入指令 300G 或 BASIC 下使用命令 CALL768，如要退出可按 Ctrl-Reset 键。

程序的第一行调用监控的清屏幕子程序，清除屏幕，把光标移至左上角；第二行调用监控的读键子程序，把键入的键码存放在 6502 微处理器的累加器 A 中；第三行调用监控的显示子程序，把 A 中的键码转换成相应的字符显示在屏幕上；第四行无条件转移指令转回到第二行，继续读键。由于调用了监控子程序，因而程序十分短小。如果作以下的修改：

```
309- D0 F8 BNE     $0303
30B- F0 F6 BEQ     $0303
```

用条件转移指令模拟无条件转移指令的动作，则可以用监控的 M 命令把程序移至内存合适的地方。

不熟悉机器语言的读者，可以运行以下的 BASIC 程序：

```
10 FOR I=768 TO 779
20 READ A:POKE I,A:NEXT
50 DATA 32,88,252,32,12,253,32,237,253,
   76,3,3
60 CALL 768
```

C 语言初阶讲座

第八讲　指针与数组

李文兵

指针在 C 中占有重要位置。只有掌握了指针，才算把 C 学到手。这里，介绍指针结构及其各种用法。

1. 一维数组的指针

下面的说明语句定义一维数组 s 和指针变量 p。

```
char s[16], *p;
```

通过赋值语句：

```
p=s;
```

将在指针里赋上数组 s 的首地址，即指针 p 指向了数组 s。而且，数组元素 s[i]可表为 *(p+i)，元素 s[5]就可用 *(p+5)表示。因此，数组元素的读写，完全可由指针实现，如练习 8.1 所示。

```
A>type exp8_1.c
#define MAX 10
main()
{ int arry[MAX], *ptr,i;
    ptr=arry;
    /* 指针 ptr 指向数组 arry */
    for(i=0;i<MAX;i++) arry[i]=i;
    /* 变化数组下标,给数组赋值 */
    for(i=0;i<MAX;i++) printf("%3d",arry[i]);
    for(i=0;i<MAX;i++) *ptr++=i*i;
    /* 用指针给数组赋值 */
    printf("\n");
    for(i=0;i<MAX;i++) printf("%3d",arry[i]);}

A>exp8-1
  0   1   2   3   4   5   6   7   8   9
  0   1   4   9  16  25  36  49  64  81
```

2. 数组的动态分配

程序中设定的数组，到程序被编译时，其空间便被确定。但有时希望需要空间时再从称为堆的自由空间分配。为此，要使用存储管理库函数；诸如：

```
malloc(nbyte)
free(s)
```

来确定。这些函数能自由地占有或释放堆的存储空间，malloc()的调用形式如下：

```
int *p;
unsigned int nbyte;
p=malloc(nbyte);
```

结果把分配的存储空间的首址赋给指针 p。练习 8.2 是确定能存 1000 个整型数据存储空间程序。

```
a>type exp8_2.c
#include <stdio.h>
#include "stdlib.h"
#define MAX 1000
main()
{ int *p,*s,i;
    s=p=malloc(MAX*sizeof(int));
    for (i=0;i<MAX;i++)
        *p++=1000+i;
    for(i=0;i<10;i++)
        print("%x",*(s+i));
    free(s);
}
A>exp8_2
3e8 3e9 3ea 3eb 3ec 3ed 3ee 3ef 3f0 3f1
```

执行该程序，函数 malloc()便确定了含有 1000 个整型元素的数组。当该数组不用时，该程序通过函数 free()，把所确定的存储空间返回给系统。

这样动态地分配堆空间，会更有效地利用存储空间，即该空间可多次用于不同目的，而且其数组的大小，可根据系统来设定，提高了程序的可移植性。这些函数，不仅可以动态分配数组，而且对于队列、链表、二叉树等，在动态分配存储空间时，也是必须的。

3. 指针和字符串

在字符串的操作中，使用指针是很方便的。关于字符串的长度计算、拷贝、连接等操作，如练习8.3所示。

```
A>type exp8_3.c
#define MAX 10
main()
{ char s[MAX],t[MAX], *p, *pp;
int len=0;
gets(s);
p=s;
/* 字符串长度 */
while( *p++) len++;
printf("length of strimg=%d\n",len);
/* 字符串拷贝 */
p=s;pp=t;
while( *pp++= *p++);
printf("string of t=%s\n",t);
/* 字符串连接 */
p=s;pp=t;
while( *pp++);
 *pp--;
while( *pp++= *p++);
printf("string of t=%s\n",t);
}
A>exp8_3
qwer
length of string=4
string of t=qwer
string of t=qwerqwer
```

程序中，对字符型数组 s 输入字符串，是使用函数 gets()进行的；若使用指针，可按如下进行。

```
char *s;
s="string";
```

也可把这两个语句合并为：

```
char *s="string";
```

4. 指针的运算性质

适于指针的运算有：

(1)指针加减一个整数　在 C 语言中，允许指针加减一个整数，其意义是当指针指向数组时，使指针相对原位置移动，以使其指向另一个数组元素。例如：

```
int a[10], *p;
p=a;
```

是使指针 p 指向了数组 a，即指向数组 a 的第一个(0 号)元素。这时，若执行：

```
p+=i;
```

便使 p 指向数组 a 的 i 号元素。如练习 8.4 所示。

```
A>type exp8_4.c
#define W 10
main()
{ int a[W]={0,1,2,3,4,5,6,7,8,9}, *p;
p=a;
printf("%d,%d,%d\n", *p+3, *(p+3), *++p);
printf("%d\n", *p++);
printf("%d,%d,%d\n", *p, *(p+3), *(p-2));
}
A>exp8_4
4,4,1
1
2,5,0
```

通过这个练习，应该对指针操作有如下了解。

① *(p+3)与 *p+3 不同：*(p+3)是取指针当前位置后面的第 3 个元素的值；而指针位置并无改变；*p+3 是先取 p 所指元素的值，再在该值上加 3。

② *p++与 *(p+1)不同：*p++是先取 p 所指元素的值，再把指针向后调整一个元素，且指针便固定在此位置上；而 *(p+1)则是取指针当前位置后面一个元素的值，指针位置不改变。

③ *++p 的结合性是 *(++p)。因此，是先把指针后调整一个元素，再取所指元素的值。

指针的这一运算性质，被广泛地用于堆栈的操作上。所谓堆栈是按照"先进后出"进行存取操作的存储空间。程序中 push()函数的功能是，把一个数据存入到指针所指定的堆栈空间内，且把指针向高地址移动一个位置。pop()函数则把指针位置向低地址移动一个位置，取出当前指针所指定的值。在堆栈操作中，使用指针变量是极为方便的，如练习 8.5 所示。

```
A>type exp8_5.c
#include <stdio.h>
#define MAX 10
int *p; /*指向自由区*/
int *t; /*指向栈顶*/
int *b; /*指向栈底*/
void push();
main()
{ int x,y,i,var;
char s[80];
p=(int *)malloc(MAX*sizeof(int)); /*建栈*/
if(!p)
{ printf("建栈失败");
    exit(1);
}
t=p;b=p+MAX-1
printf("输入 10 个数:");
for(i=0;i<10;i++)
{ scanf("%d",&var);
   push(var);
}
printf("弹出 5 个数:");
for(i=0;i<5;i++)
{ x=pop();
  printf("%d ",x);
}
printf("\n");
}
void push(i) /*压栈*/
int i;
{ if(p>b)
{ printf("栈满");
  return;
}
*p=i;
p++;
}
pop()/*退栈*/
{ p--;
if(p<t)
{ printf("栈空\n");
  return 0;
}
return *p;
}
A>exp8_5
输入 10 个数:1 2 3 4 5 6 7 8 9 1
弹出 5 个数:1 9 8 7 6
```

对于不同类型的指针，同样移动一个元素的位置，实际移动的距离(字节数)是完全不同的，如下所示。

指针的数据类型	移动一个元素的实际距离
char	1
int	2
float	4
double	8

为了使指针移动时，能恰好指定相应元素，C 编译系统对指针所加减的整数乘上一个比例因子。这比例因子正是各数据类型所占字节数，例见练习 8.6。

```
A>type exp8_6.c
```

```
main()
{ char c[4]={"1234"}, *pc;
  int i[4]={ 1,2,3,4,}, *pi;
  float f[4]={ 1,2,3,4}, *pf;
  double d[4]={1,2,3,4}, *pd;
  int n;
  pc=c;pi=i;pf=f;pd=d;
  for(n=0;n<4;n++)
    printf("c[%p]=%c\n",pc, *pc++);
    printf("\n");
  for(n=0;n<4;n++)
    printf("i[%p]=%i\n",pi, *pi++);
    printf("\n");
  for(n=0;n<4;n++)
    printf("f[%p]=%f\n",pf, *pf++);
    printf("\n");
  for(n=0;n<4;n++)
    printf("d[%p]=%f\n",pd, *pd++);
}
A>exp8_6
c[FF9F]=1
c[FFA0]=2
c[FFA1]=3
c[FFA2]=4

i[FFA4]=1
i[FFA6]=2
i[FFA8]=3
i[FFAA]=4

f[FFB0]=1.000000
f[FFB4]=2.000000
f[FFB8]=3.000000
f[FFBC]=4.000000

d[FFC6]=1.000000
d[FFCE]=2.000000
d[FFD6]=3.000000
d[FFDE]=4.000000
```

(2)指针赋地址　这类运算有：

p=&var;把变量 var 的地址赋给指针 p;

p=&a[i];把数组元素 arry(i)的地址赋给指针 p;

p=a;把数组的首地址赋给指针 p;

p=px;把同类指针 px 的值赋给指针 p。

关于这类运算，请看练习 8.7。

```
A>type exp8-7.c
main()
{int x=15, *p, *q, *r;
int a[10]={1,2,3,4,5,6,7,8,9,10};
q=a;p=q;
printf("%p,%p,%p\n",a,q,p);
p=&a[3];q=&a[5];r=&x;
printf("%p,%p,%p\n",p,q,r);
printf("%4d,%4d,%4d\n", *p, *q, *r);
}
A>exp8_7
FFCA,     FFCA,     FFCA
FFD0,     FFD4,     FFC8
   4,        6,       15
```

(3)两个指针比较运算　当两个指针指向同一数组时，可用关系运算符对它们进行比较。实际上，就是比较两个指针在该数组的相对位置。设 px 和 py 指向同一数组，那么，

px<py;当 px 所指位置在 py 的之前时为真；

px>py;当 px 所指位置在 py 的之后时为真；

px<=py;当 px 所指位置在 py 的之前或为同一位置时为真；

px>=py;当 px 所指位置在 py 的之后或为同一位置时为真；

px==py;当 px 所指位置与 py 的相同时为真；

px!=py;当 px 所指位置与 py 的不相同时为真。

〔练习 8.8〕

```
A>type exp8_8.c
#define TRUE 1
#define FALSE 0
main()
{if(isitpal("FOUR SCORE IN SEVEN YEARS")==
    TRUE)
printf("string 1 is a pal indrome\n");
if(isitpal("ABLE WAS I ERE I SAW ELBA")==TRUE)
printf("string 2 is a pal indrome\n");
}
isitpal(str)
char *str;
{char *strl=str;
while(*strl!='\0') ++strl;
--strl;
while((str<strl)
if(*str++!=*str--)
return(FALSE);
return(TRUE);
}
A>exp8_8
string 2 is a pal indrome
```

(4)两个指针相减　当两个指针指向同一数组时，其差值是两指针相对移动的元素个数。利用这一运算性质，可以计算字符串的长度，如练习 8.9 所示。

```
A>type exp8_9.c
main()
{char *s="abcdef ghijkl.\n";
char *p=s;
while(*p)
p++;
printf("length of string s=%d\n",p-s);
```

```
}
A>exp8_9
length of string s=15
```

5. 指针与多维数组

多维数组可用指针调用，这是因为多维数组在存储空间是按行存储的，用一般指针变量调用多维数组的程序，如练习 8.10 所示

```
A>type exp8_10.c
#include "time.h"
#include <stdio.h>
#include "stdlib.h"
#define ROW 4
#define col 4
main()
{int i,j,s[ROW][col];
      long n;
      srand(time(&n));
      for(i=0;i<ROW;i++)
      for (j=0;j<col;j++)
      s[i][j]=rand()/100;
      print_mat(" mat s[][]=",s);
      trans(s);
      print_mat(" trans mat=",s);
}
trans(a)
int a;
{ int i,j;
for(i=0;j<ROW;i++)
for(j=i+1;j<col;j++)
swap(a+i*ROW+j,a+j*ROW+i);
}
swap(a,b)
int *a,*b;
{int t;
t=*a;*a=*b;*b=t;
}
print_mat(s,mat)
char *s;
int mat[][col];
{int i,j;
printf(" ---%s---\n",s);
for(i=0;i<ROW;i++)
{for(j=0;j<col;j++)
printf("%4d ",mat[i][j]);
printf("\n");
  }
}
A>exp8_10
  ---mat s[][]=---
199     177     263     84
242     119     115     262
239     165     44      288
73      68      29      60
---trans mat=---
1991    29682   -20224  0
21510   256     30465   0
239     165     44  288
73      68      29  60
```

6. 指针数组

用户还可以定义指针数组，例如：

```
float *x[100];
```

就定义了一个指向浮点数的指针数组。可按如下所示，对该数组赋初值。

```
x[0]=a;
x[1]=b;
⋮
```

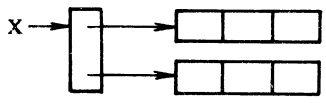
为了较清楚地理解指针数组的概念，这里，有必要再多说几句。假如，我们定义了一个指针数组：

```
float *x[2];
```

注意，变量 x 不是两维数组；而是由两个指向一维 float 数组的指针组成的数组。但对它适当地初始化，可用作两维数组。要想使它能表示 2×3 的 float 数组，只要按如下初始化即可

```
x[0]=(float *)malloc(3*sizeof(float));
x[1]=(float *)malloc(3*sizeof(float));
```

现在 x[0]和 x[1]都指向能保存三个 float 数据的存储空间，其图示如下：



其实例如练习 8.11 所示。

```
A>type exp8_11.c
char *err[]={"EZER0 0 /* ERROR 0 */",
             "EIVLFNC 1/* invalid function number
              */",
             "ENOFILE 2/* File not found */",
             "ENOPATH 3/* Path not found */",};
main()
{ int i;
  for(i=0;i<4;i++)
    printf("errno[%d]=%s\n",i,err[i]);
}
A>exp8_11
errno[0]=EZER0 0   /* ERROR 0 */
errno[1]=EIVLFNC 1   /* invalid function number */
errno[2]=ENOFILE 2   /* File not found */
errno[3]=ENOPATH 3   /* path not found */
```

引用命令行参数是使用指针数组的典型例子。即如下所示的命令行：

A>程序名　参数 1　参数 2　……

为引用命令行参数，主函数 main()应按如下设置：

```
main (argc,argv)
int argc;
```

```
char *argv[];
{          …          }
```

其中，argc 的值等于参数总数＋1；argv 是指针数组，其内容如下：

argv[0]的内容为程序名的地址

argv[1]的内容为参数 1 的地址

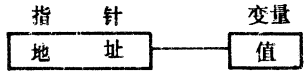
argv[2]的内容为参数 2 的地址

作为例子，请看练习 8.12。

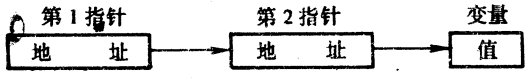
```
A)type exp8_12.c
main(argc,argv)
int argc;
char *argv[];
{ while(argc--)
  printf("argv[%d]=%s\n",argc,argv[argc]);
}

A)exp8_12 arg1 arg2 arg3 arg4
argv[4]=arg4
argv[3]=arg3
argv[2]=arg2
argv[1]=arg1
argv[0]=A:\EXP8_12.EXE
```

7. 指向指针的指针

应该指出的是，指针数组是指向指针的指针，即形成一个指针链。前面我们所介绍的指针其值是某变量的地址，如下所示：



而在指向指针的指针里存放的是第二个指针的地址，其实际值是由第二指针指出的，即



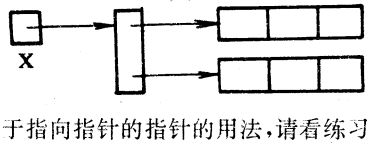
在 C 语言里，用 * *x，表示 x 为指向指针的指针，其作用相当于 *x[]。例如，在程序中，我们定义了：

```
float **x;
```

则意味着 *x 的类型是指向 float 的指针，而 **x 的类型是 float，即用 **x，可引用值。这样的结构按如下初始化后，也可作两维数组。

```
x=(float **)malloc(2*sizeof(float **));
x[0]=(float *)malloc(3*sizeof(float));
x[1]=(float *)malloc(3*sizeof(float));
```

其图示如下：



关于指向指针的指针的用法，请看练习 8.13。

```
A>type exp8_13.c
main(argc argv)
int argc;
char *argv[];
{ printf("argc=%d\n",argc);
  while(*argv++,argc--)
  printf("argv %0x %s\n",argv,*argv);
}
A)exp8_13 aa bb cc
argc=4
argv ffdc aa
argv ffde bb
argv ffe0 cc
argv ffe2 (null)
```

8. 数组指针

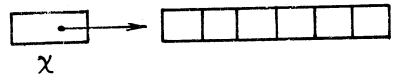
下面的说明语句便定义了一个指向三个 float 数据组成的数组指针。

```
float(*x)[3];
```

它的值最初是不确定的，必须使用 malloc()函数来确定。按如下所示初始化，便可得到 2*3 数组。

```
x=(float(*)[])malloc(2*3*sizeof(float));
```

这样，x 所指向的存储空间，不是保存一个三元素 float 数组，而是连续存放这样的两个数组。x 可图示为：



练习 8.14 是求某年某月某日是这一年第几天的程序，该程序就使用了数组指针。

```
A)type exp8_14.c
int dtab[][13]={
                {0,31,28,31,30,31,30,31,31,30,
                31,30,31},
                {0,31,29,31,30,31,30,31,31,30,
                31,30,31},
                };
main()
{ int y,m,d,yd;
  scanf("%d,%d,%d",&y,&m,&d);
  yd=dofy(dtab,y,m,d);
  printf("%d:%d:%d=%d\n",y,m,d,yd);
}
dofy(x,ty,tm,td)
int (*x)[13],ty,tm,td;
{ int i,j;
  j=ty%4==0&&ty%100!=0||ty%400==0;
  for(i=1;i<tm;i++)
  td+=*(*(x+j)+i);
  return(td);
}
A)exp8_14
1991,2,11
1991:2:11=42
```

我们把本节所介绍的几种指针结构，就二维数组的情况，小结一下，如表 8.2 所示，供参考。

表 8.2　指针结构用于二维数组小结表

指针结构	名称	行、列情况	左值	效率
float xa[2][3]	二维数组	静态数组(固定行列)	xa,xa[0]均不是左值	高
float *xb[2]	指针数组	固定行、动态列	xb 不是左值,xb[0]是左值	低,节省空间
float(*xc)[3]	数组指针	固定列、动态行	xc 是左值,xc[0]不是左值	低,节省空间
float **xd	指针的指针	行、列皆为动态	xd、xd[0]皆为左值	低,节省空间

邮购广告:

▲中外合资温州福灵达仪器仪表有限公司邮购部,长期优价供应进口组装 $3\frac{1}{2}$ 位数安万用表。批价从优。

型号	主要特点	邮购价(含运费)
53188	袖珍,可放置上衣口袋。多功能、全保护,可测于电池 9V、1.5V 好坏。最大电阻量限 200M。	170.00
M3800	显示器带放大镜、全保护功能,误测不致烧坏仪表。微电流测量分辨率为 0.01MA	180.00
DT8908	显示器带放大镜,量程全保护 5 档电容测量。不需调零	260.00
DT940C	显示器放大镜,量程全保护;测电容 5 档,不需调零。内置温度传感器,可直测室温。	390.00
5115	全量程保护,配有专用稳压管,改善电池降压物性和温度特性。	190.00
PF72	台式、便携两用。$3\frac{1}{2}$ 位绿色显示,适用车间、实验室、车、船、野外及光线暗弱环境或远距离观测	620.00

注:备有详细说明,函索即寄,款到一周发货。
电话:225222 直拨区号:0577 电挂:2752
帐号:中行温州分行营业部 12825069—3
地址:浙江温州黄府巷 40 号
邮编:325000
联系人:王文瑞

(上接第 40 页)

因而地址 LED 全亮。用万用表的一根表笔取下当短路线,一端接地,另一端分别触碰 6502 的 9～18 脚,则相应 LED 灭。如不受控制,则说明此线路有故障,故障原因可能有:U_{13}、U_{14}三态门的控制端开路,连接线开路或短路等。

此外,6502 的 2、4、6、40 脚如有断路,则 6502 也不执行程序。

五、接口电路故障的检修

在前面检查显示器时已检查了输入接口电路 U_{20},下面只介绍输出接口电路 U_{21}(74LS374)的检修方法。

用键盘发出地址 $201(或 $203、$205……等奇数地址),选中 U_{21},将 8 位数据开关全置 1,按住 $\overline{WR}$ 键,则 U_{21}的 8 个数据输出端应全为 1。然后将数据为全 0,数据输出端也应为全 0。如有异常,可检查相应连接线及 U_{21}各引脚的连接情况,即可排除故障。

学用单片机

多功能电子钟

(KDC-Ⅲ最小系统应用实例)

张培仁　刘振安

以单片机最小系统为核心，增加显示器和有关功能键，即可组成多功能电子钟。

电子钟用定时器 O 产生了定时中断，控制电子钟走时，电子钟的初值(时、分、秒)由键盘输入，而后运行。增减定时常数值，即可细调电子钟走时精度。

定时器 O 的定时溢出脉冲周期 $t=tc*T_0*10^{-6}$ 秒。其中 tc 当单片机主振频率是 6MHz 时，等于 2μs。t 为每次溢出中断的间隔时间，我们取0.1s，也就是每 0.1s 中断服务一次。这样求出 T_0 值＝50000 即等于 16 进制数是 3CBOH。因为每次中断都执行几条中断服务程序，所以 T_0 的修正值为 3CBFH。这样确定了 T_0 的时间常数值为：$TH_0=3CH$

$TL_0=BFH$。

显示器和键盘和 KDC－Ⅲ开发机的显示和键盘完全一样，地址口也一样，只是用 2 个或 4 个键就可以。键的用法和电子表对时使用方法相似。键分析和处理程序用了一个智能仪器中常用的多次复用算法，这里不再讲了。图一是电子钟程序框图。

电子种程序清单如下：

```
        MOV     DPTR,#4101H
        MOVX    @DPTR,A     送字形
        MOV     A,R2
        INC     DPTR
        INC     DPTR
        MOVX    @DPTR,A     开显示
DSUP2:  DJNZ    R3,DSUP2
        INC     RO          指向下一缓冲区
        CLR     C
        MOV     A,R2
        RRC     A           指向下一位显示
        MOV     R2,A
        JNZ     DSUP1       未完继续
        MOV     R0,#DS6     向显示缓冲区放数
        MOV     A,SBF
        ACALL   PTDS
        MOV     A,MBF
        ACALL   PTDS
        MOV     A,HBF
        ACALL   PTDS
        POP     DPL
        POP     DPH
        SJMP    DSUP
STOP:   CLR     TR0         停止定时器
        LJMP    2000H       关中断，返回
GTDS1:  MOV     A,@R0       从显示缓冲区取数字程序
        CJNE    A,#0AH,GTDS2
        JNC     STOP
GTDS2:  ANL     A,#OFH
        MOV     R1,A
        DEC     R0
        MOV     A,@R0
        SWAP    A
        ANL     A,#0F0H
        ORL     A,R1
        RET
PTDS:   MOV     R1,A 向显示缓冲区放数字程序
        ACALL   PTDS1
        MOV     A,R1
        SWAP    A
PTDS1:  ANL     A,#0FH
        MOV     @R0,A
        DEC     R0
        RET
CLOCK:  MOV     TL0,0B7H 中断服务程序
        MOV     TH0,#3CH
                ORG   2000H
                BUF   EQU 23H1/10 秒单元缓冲区
                SBF   EQU 22H 秒单元缓冲区
                MBF   EQU 21H 分单元缓冲区
                HBF   EQU 20H 时单元缓冲区
TIMER:  SJMP    GTDS
                ORG   200BH
        AJMP    CLOCK 转中断服务程序
                ORG   2030H
GTDS:   MOV     R0,#DS2
        ACALL   GTDS1 向显示缓冲区取数
        CJNE    A,#24H,CHK 测试时间初值合法性
        JNC     STOP
CHK:    MOV     R0,#DS4
        ACALL   GTDS1
        CJNE    A,#60H,CHK1
        JNC     STOP
CHK1:   MOV     R0,#DS6
        ACALL   GTDS1
        CJNE    A,#60H,CHK2
        JNC     STOP
CHK2:   ANL     TMOD,#0F0H 设定时器工作方式 1
        ORL     TMOD,#01H
        ORL     IE,#82H 允许中断
        ORL     TL0,#0B0H 送时间常数初值
        MOV     TH0,#30H
        MOV     BUF,#00H1/10 秒单元清 0
```

```
        SETB    TR0 开定时器
DSUP:   MOV     R0,#DS1 显示程序
        MOV     R2,#20H
        MOV     R3,#00H
        MOV     DPTR,#4100H 指向字形表
        MOV     A,#0DH
        MOVX    @DPTR,A
        MOV     DPTR,#SGTB
        PUSH    DPH
        PUSH    DPL
DSUP1:  MOV     A,#0FFH
        MOV     DPTR,#4101H
        MOVX    @DPTR,A 关显示
        MOV     A,@R0 取字形
        POP     DPL
        POP     DPH
        MOVC    A,@A+DPTR
        PUSH    DPH
        PUSH    DPL
        PUSH    PSW
        PUSH    ACC
        SETB    RS0
        INC     BUF
        MOV     A,BUF
        CJNE    A,#0AH,DONE
        MOV     BUF,#00H
        MOV     A,SBF
        INC     A 秒增 1
        INC     A
        DA      A
        MOV     SBF,A
        CJNE    A,#60H,DONE
        MOV     SBF,#00H
        MOV     A,MBF
        INC     A 分增 1
        DA      A
        MOV     MBF,A
        CJNE    A,#60H,DONE
        MOV     MBF,#00H
        MOV     A,HBF
        INC     A 时增 1
        DA      A
        MOV     HBF,A
        CJNE    A,#24H,DONE
        MOV     HBF,#00H
DONE:   POP     ACC 恢复
        POP     PSW
        RETI    中断返回
        DS1     DATA    5AH
        DS2     DATA    5BH
        DS3     DATA    5CH
        DS4     DATA    5DH
        DS5     DATA    5EH
        DS6     DATA    5FH
SGTB:   DB      0C0H, 0F9H, 0A4H, 0B0H, 099H,
                092H,082H,0F8H 字形代表码
        DB      080H, 090H, 088H, 083H, 0C6H,
                0A1H,086H,08CH,0FFH
        END
```

我们在这个电子钟基础上又增加了一些功能。增加了三十二年夏时制自动转换，闰年判断和处理，三十二年农历的年、月、日的计算和显示。

三十二年夏时制自动转换程序：

我国夏时制是每年的四月中旬的第一个星期天开始至9月中旬的第一个星期天结束。一周为七天，故星期天只能在四月或九月的10日－17日之间。所以程序中只判断四月或九月的10日－17日。再查夏时制转换日期表。夏时制的实现过程在相应的日期将时钟拨快或拨慢1小时。程序框图如图2：

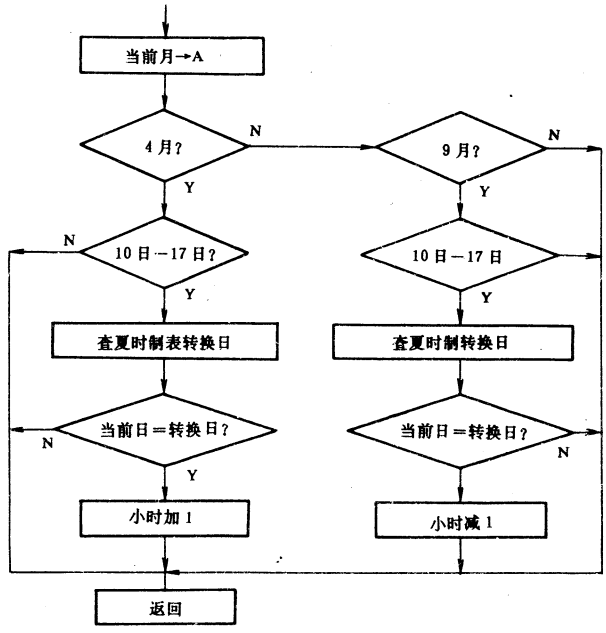


图2　夏时制处理程序框图

闰年判断程序中，用寄存器B来确定是否为闰年。闰年的判断是根据建立的闰年表来查询的。闰年表是根据天文常识编制的。一般来讲，公历年数能够被4整除时为闰年，否则为平年。但有特例除外，能被100整除，而不能被400整除的不是闰年。在1989年至2021年无此特例。当闰年时B置1，否则B为0。

农历时转换程序：

农历是具有我国民族特色的历法体制。它是根据月亮的运转周期确定的。由于它的天文计算过程很繁琐，并且农历日期还有个别特例地方很难处理。所以我们依旧使用建表的方法来实现日历变换。但表又不能太长，不可能把32年每天都建一个农历表。如每天建表需占用12K ROM。因此我们采用建表和计算相结合的方法。即要转换农历时，先查本月公历2日的农历日期，然后进行数据分离和计算，再得相应农历年、月、日。农历日表共占12×32×1字节＝384字节。这样大大节省了ROM的容量。

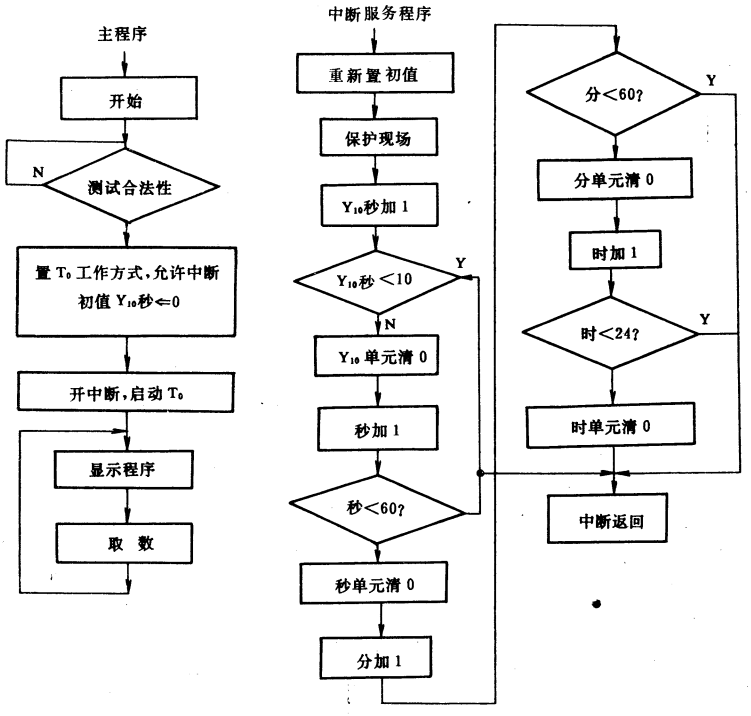


图 1 电子钟主程序和中断服务程序框图

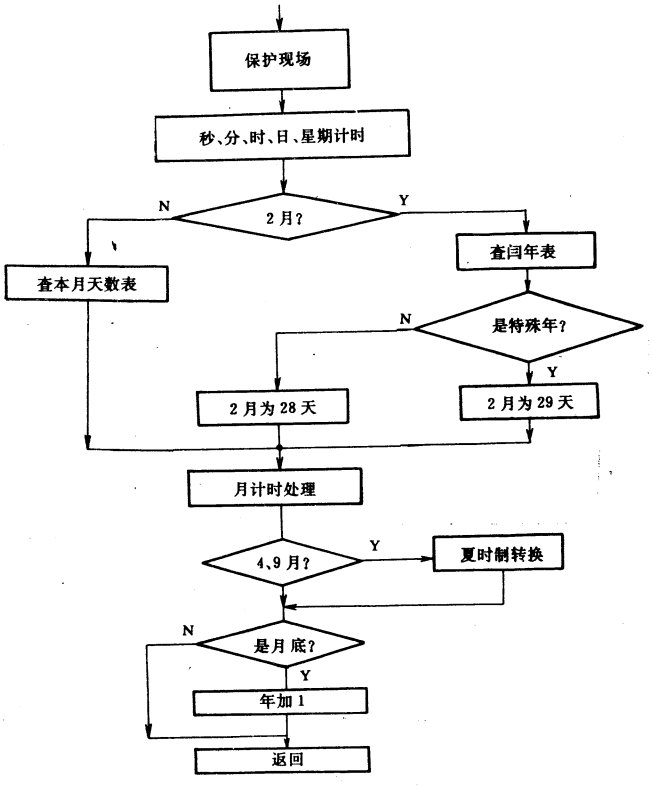


图 3 闰年处理程序框图

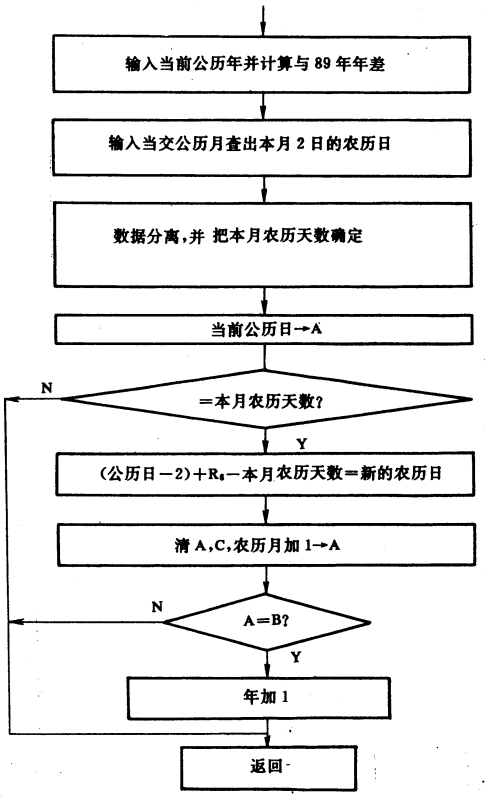


图 4 农历转换流程框图

农历日计算方法如下：

(公历日－2)＋公历 2 日的农历日期。若此数大于当月农历总天数，则减掉总天数，剩余数为农历日。同时农历月，年进行相应的变化。若当前公历日为 1 日时，则 2 日的农历日减 1。但在这三十二年中，只有一特例。只有 2000 年 6 月 1 日特殊。2000 年的 6 月 1 日农历日则应为 4 月 30 日(按以上算法计算出的)，但 2000 年农历 4 月为小月，只有 29 天，所以公历 6 月 1 日的农历日实际应为 4 月 29 日。因此要对 2000 年 6 月 1 日这一天进行特殊处理。

2000 年部分日期农历、公历对应表

农历	公历
4 月 26 日	5 月 29 日
4 月 27 日	5 月 30 日
4 月 28 日	5 月 31 日
4 月 29 日	6 月 1 日
5 月 1 日	6 月 2 日

农历转换程序对农历日表中数据进行数据分离。农历日表按顺序列了自 1989 年 1 月至 2020 年 12 月每月 2 日的农历日期。每一数据一个字节(8 位)，数据位定义如下：

(1)数据低五位是该农历日期的 16 进制数。

(2)数据的第六位表示闰月信息。置 1 时为农历月为闰月，0 为平月。(下转第 35 页)

电脑巧开发

简易精确的 APPLE Ⅱ 时钟卡

杭州大学　姜仕仁　施青松

APPLE Ⅱ 和中华学习机被广泛应用于过程控制和自动采样系统，但是当它们被用于与时间有关的控制场合和定时采样过程中时，用户往往为该机没有时间输出指令（如许多计算机都可通过 PRINT TIME 指令得到时间值）而烦恼。如笔者在做小型动物活动频率自动记录实验中，需要记录动物每次出入洞口的时间，在此，由于程序跳转等原因，很难用软件来准确记时。于是，我们利用电子钟机芯制作了一块计时精确的时钟卡。

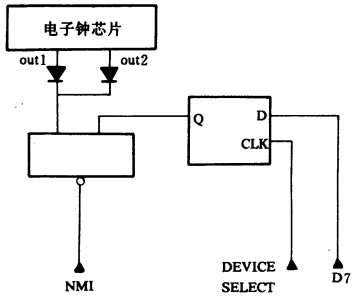


图 1

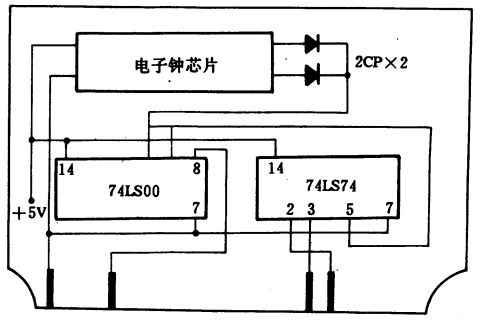


图 2

该卡是利用电子钟产生的秒脉冲。APPLE Ⅱ 扩展槽口上的第 29 脚为非屏蔽中断请求接脚（$\overline{NMI}$），当该脚出现高电平向低电平的负跳变时能被 6502 所识别，6502 在执行完正在执行的本条指令之后就暂停主程序而进入$\overline{NMI}$中断响应周期。故只要在 $FFFA 和 $FFFB 两单元指向的地址（$3FB）开始的三个单元中置一跳转指令，将非屏敝中断响应引导到一个计时子程序。这样，每当出现一个负脉冲，中断响应计时子程序便加 1，并且用户程序和计时子程序分时执行，互不影响。当用户程序中需要当前时间，可方便地从有关地址中读取。

本时钟卡的线路图如图 1 所示，图 2 为实际接线图。

电子钟机芯上有两个 0.5Hz 的脉冲输出端 out_1 和 out_2，两者交替推动秒针每秒走动一次。在 out_1 和 out_2 两输出端分别接上一个二极管，两个二极管构成或电路，在输出端产生 1Hz 的脉冲送入与或门电路中的一个输入端，另一个输入端受 D 触发器控制。当 D 触发器输出为 0 时，不管电子钟机芯的脉冲是 0 还是 1，与或门的输出不变。而当 D 触发器输出为 1 时，与或门输出为与输入反相的脉冲，从而引起非屏蔽中断响应。D 触发器受设备选择线DEVICE SELECT和 D_7 控制，当 D_7 为 0 时输出为 0，D_7 为 1 时输出为 1。

按图 2 的接线联结各元件，将它们安装在一块实验线路板上，检查无误后插入 1～7 任一槽口中，然后给 $3FB～$3FC 三个地址中置入跳转指令，可在监控状态下输入：

```
3FB:4C　　00　　03
```

然后再以 $300 开始输入 CLOCK.SUB 中断响应计时子程序，最后通过 POKE AD,X 打开时钟卡，（AD＝49280＋SLOT×16;X≥128）。由于 CLOCK.SUB 子程序是自动显示的，时钟卡一打开，先是显示出分别存放时、分、秒的 $7、$8、$9 三个地址中的随机数，用户可通过 POKE 指令来设置这三个地址的当前时间值，以后便开始自动记时。用户在程序的任何地方的任何时刻都可向 $7、$8、$9 三个单元得到精确的时间。程序 1 是用该时钟卡记录老鼠每次出入洞口时间的实例（洞口安装一光电传感器）。同样有了这块时钟卡，可方便地应用于任何定时采样及过程控制。

当数据或程序存备之前，必须关掉时钟卡，以免损坏盘中的程序。关时钟卡的指令与开时相仿，但 X 值＜128 即可。

程序 1

```
5 POKE 1019,76:PLKE 1020,0:POKE 1921,3
10 PRINT CHR$ (4);"BLOAD CLOCK.SUB"
20 INPUT "PRESENT TIME:(Hour,Minute,Second)";
   HO,MI,SE:HOME
25 IF HO>23 OR MI >59 OR SE >59 THEN 20
30 POKE 7,HO:POKE 8,MI:POKE 9,SE
35 POKE 49344,255
40 ST =4096
50 A=PDL(1)
60 IF A =255 THEN 80
70 IF PEEK (7) =HO-(MI=0) AND PEEK(8)=(MI-1)
```

```
    +60*(MI=0) THEN 100
75 N=0:GOTO 50
80 N=N+1:IF N=1 THEN POKE ST,PEEK(7):POKE
    ST+1,PEEK(8):POKE ST+2,PEEK(9)
85 N=2:ST=ST+3
90 GOTO 50
100 END
```

CLOCK.SUB 子程序

```
0300-  48          PHA
0301-  8A          TXA
0302-  48          PHA
0303-  98          TYA
0304-  48          PHA
0305-  E6 09       INC  $09
0307-  A5 09       LDA  $09
0309-  C9 3C       CMP  #$3C
030B-  30 1C       BMI  $0329
030D-  A9 00       LDA  #$00
030F-  85 09       STA  $09
0311-  E6 08       INC  $08
0313-  A5 08       LDA  $08
0315-  C9 3C       CMP  #$3C
0317-  D0 10       BNE  $0329
0319-  A9 00       LDA  #$00
031B-  85 08       STA  $08
031D-  E6 07       INC  $07
031F-  A5 07       LDA  $07
0321-  C9 18       CMP  #$18
0323-  D0 04       BNE  $0329
0325-  A9 00       LDA  #$00
0327-  85 07       STA  $07
0329-  A9 BA       LDA  #$BA
032B-  8D 12 04    STA  $0412
032E-  BD 15 04    STA  $0415
0331-  A9 01       LDA  #$01
0333-  85 22       STA  $22
0335-  A9 00       LDA  #$00
0337-  85 06       STA  $06
0339-  A2 00       LDX  #$00
033B-  A0 00       LDY  #$00
033D-  86 1A       STX  $1A
033F-  B5 07       LDA  $07,X
0341-  C9 0A       CMP  #$0A
0343-  30 08       BMI  $034D
0345-  38          SEC
0346-  C8          INY
0347-  E9 0A       SBC  #$0A
0349-  C9 0A       CMP  #$0A
034B-  10 F9       BPL  $0346
034D-  48          PHA
034E-  98          TYA
034F-  69 B0       ADC  #$B0
0351-  A6 06       LDX  $06
0353-  9D 10 04    STA  $0410,X
0356-  68          PLA
0357-  69 B0       ADC  #$B0
0359-  E8          INX
035A-  9D 10 04    STA  $0410,X
035D-  E6 06       INC  $06
035F-  E6 06       INC  $06
0361-  E6 06       INC  $06
0363-  A6 1A       LDX  $1A
0365-  E8          INX
0366-  E0 03       CPX  #$03
0368-  D0 D1       BNE  $033B
036A-  68          PLA
036B-  A8          TAY
036C-  68          PLA
036D-  AA          TAX
036E-  68          PLA
036F-  40          RTI
```


邮 购 服 务

电子工业出版社软件部向广大用户提供LASER310/200计算机磁带软件，质量可靠，欢迎选购，定价如下：

1. 天体运行 10 元
2. BASIC 入门①
3. BASIC 入门② 20 元
4. 简汉字库 10 元
5. 监　　控 15 元
6. 标准带 10 元
7. 机械波 10 元
8. 英语同义词 10 元
9. 键盘绘图 15 元
10. 一元二次方程 10 元
11. 二元一次方程 10 元
12. 测试图 5 元
13. 打字编辑 20 元
14. 电脑唱歌 8.50 元
15. 快速反应 8.50 元
16. 抓特务 8.50 元
17. 溶洞探险 8.50 元
18. 高炮射击 8.50 元
19. 超级大盗 8.50 元
20. 打飞机 8.50 元
21. 宝龄球 8.50 元
22. 吃豆子 8.50 元
23. 陷井 8.50 元
24. 冲出迷宫 8.50 元
25. 取宝 8.50 元

另加邮费 15%。

联系地址：北京 173 信箱软件部　邮编 100036

联系人：谈众安

用 EPROM 制作多路报警器

王洪伟

用 2K 的 EPROM2716 制作的 11 路报警器非常简单，如图 1 所示。

地址线 A_0～A_{10}作报警器输入端，数据线 D_0～D_6 用于驱动七段 LED 数码管显示，D_7 用作触发报警声电路。BG_0～BG_6 为驱动管。

正常情况下，通过前级的报警探头使 A_0～A_{10}维持低电平，D_0～D_7 输出 000H 单元数据，LED 显示为零，报警声不触发。当有报警输入使 A_0～A_{10}中某一位出现高电平时，不妨设 A_2 变为高电平，其余皆为低电平，D_0～D_7 输出 002H 单元数据，数码管显示为 2，同时 D_7 的高电平触发报警声电路推动喇叭发声，这种状态一直维持到报警解除，A_2 变为低电平为止。由于有可能产生某几路报警同时输入的情况，因此通过对 EPROM 编程，使 A_0～A_{10}进行优先排队，优先顺序是 A_{10}、A_9 ……A_0。

EPROM 的编程图见图②。可将一片已擦除干净的 2716 插入单板机 TP801A 的编程插座中，分段输入数据。例如：100H～1FFH 地址应输入 EF 数据。数码管用共阳的 LED 数码管。报警声电路可用 KD--9561 等音乐片。

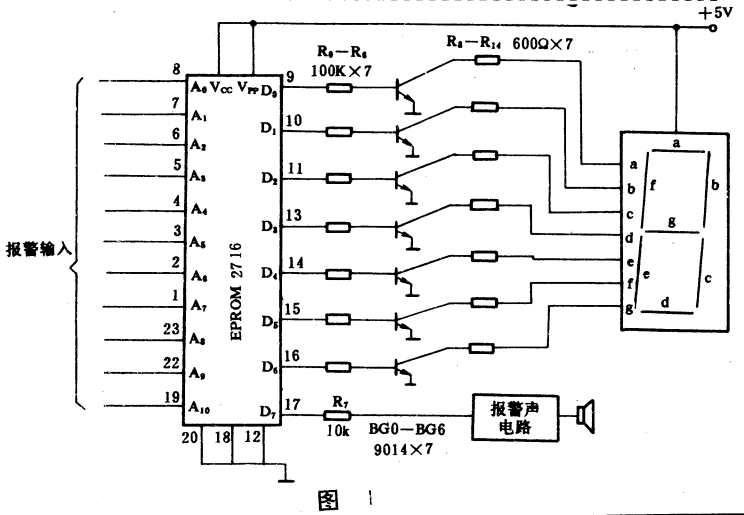


图 1

报警输入 A_{10} A_9 A_8 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0	对应 EPROM 首址	对应地址数据 二进制 D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	对应地址数据 十六进制	数码显示
0 0 0 0 0 0 0 0 0 0 0	000H	0 0 1 1 1 1 1 1	3F	0
0 0 0 0 0 0 0 0 0 0 0	001H	1 0 0 0 0 1 1 0	86	1
0 0 0 0 0 0 0 0 0 1 ×	002H	1 1 0 1 1 0 1 1	DB	2
0 0 0 0 0 0 0 0 1 × ×	004H	1 1 0 0 1 1 1 1	CF	3
0 0 0 0 0 0 0 1 × × ×	008H	1 1 1 0 0 1 1 0	E6	4
0 0 0 0 0 0 1 × × × ×	010H	1 1 1 0 1 1 0 1	ED	5
0 0 0 0 0 1 × × × × ×	020H	1 1 1 1 1 1 0 1	FD	6
0 0 0 0 1 × × × × × ×	040H	1 0 1 0 0 1 1 1	A7	7
0 0 0 1 × × × × × × ×	080H	1 1 1 1 1 1 1 1	FF	8
0 0 1 × × × × × × × ×	100H	1 1 1 0 1 1 1 1	EF	9
0 1 × × × × × × × × ×	200H	1 1 1 1 0 1 1 1	F7	8
1 × × × × × × × × × ×	400H	1 1 1 1 1 1 0 0	FC	6

（× 表示任意电平，即 0、1 皆可）

图 2

北京安华科技市场西城经营部

主要经营：电子计算机、电子元器件

各种微机：

• 中华学习机：CEC－I ¥900.0　小蜜蜂－I ¥920.00

CEC－E：驱动器 2 个 1150 元/台，扩展器 1 个、

• 家庭 PC 机：¥2950

主频：4.77/12 兆　RAM：640KB　软驱：360KB×2，101 键盘　14″双频单显

Super286：¥6300 元

主频：10/21 兆　RAM：1MB 可扩 5MB5 1/4 软驱：1.2MB. 360KB 各 1　硬盘 3.5″40MB. 101 键盘 14″双频单显　数显机箱　选配：14″CGA 彩显加 1500 元 14″1024×768VGA 彩显加 2300 元

各种：APPLE Ⅱ、CEC－Ⅰ、PC 机接口板；九针、24 针打印机，还承接各种微机的修理业务，对教育部门、个人购机实行优惠。

地址：北京西四北大街 120 号　电话：65.6342

邮编：100034　联系人：孙义强 宋涵华 BB 机：126 呼 63882

键盘电视游戏机的FBASIC语言及程序设计(上)

于 春

目前,市场上销售的家用电视游戏机(如小天才、任天堂等)都开始增加与游戏机配套的键盘,不但能进行电视游戏,还具有算术运算、谱曲演奏、英文打字、背景绘图和使用FBASIC语言进行各种运算程序、游戏程序设计等功能。使孩子们从游戏中学习电脑知识和基本编程技巧。将会有越来越多的"游戏机迷",带着游戏机画面是怎样画出来的疑惑进入电脑之门。

第一讲 FBASIC程序浅述

一、FBASIC语言的概念

电脑进行数值计算、数据处理、自动控制等工作,是通过执行一系列的命令而完成的。这些命令叫指令。一条指令进行一种操作,它告诉电脑进行什么样的运算,有哪些数参加运算,结果如何处理等。所以指令可以看做是人和电脑进行交谈的语言,称为电脑语言。目前电脑语言种类也很多,初学者一般都常用BASIC语言。

BASIC是BEGINNER'S ALL-PURPOSE SYMBOLIC INSTRUCTION CODE(初学者通用符号指令代码)一词的缩写。它的特点是语法结构简单、学习掌握容易,是一种会话式语言。它还可以不编程序,利用键盘象计算器那样直接进行运算。如欲求35与25两数之和,可用BASIC语言编一段程序进行运算:

```
10  A=35
20  B=25
30  C=A+B
40  PRINT C
```

也可以直接利用键盘进行立即运算:

```
PRINT 35+25
```

BASIC语言有许多版本。不同型号机器的BASIC语言也略有不同。键盘电视游戏机所用的语言是日本MSX BASIC语言的扩展,称为FAMILY BASIC语言,简称FBASIC语言。FBASIC语言除符合BASIC语言的指令、语法、逻辑运算、程序设计等一般的通用规则及标准外,其显著特点是在简化指令的前提下,利用最简单的程序,最少的容量完成绘图。

上例程序是进行两数求和运算,为加深认识,下面再列一个完整的FBASIC程序。在电子电路中计算流过并联电阻的总电流I,计算公式为 $I=U/R=U(\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{R_3})$ 若U、R_1、R_2、R_3已知,则求I的程序可写为

```
10  R1=10
20  R2=20
30  R3=30
40  U=12
50  I=(1/R1+1/R2+1/R3)*U
60  PRINT I
70  END
```

这个程序的头四行是提供原始数据,第五行是计算I,第六行是将计算结果显示打印出来,最后一行是说明程序到此结束(当然该程序还可以简化)。由此所见,程序是由一些英语词汇或缩写和一些数学表达式来表达的。我们学习电脑语言就是要学会这些表达方式,知道那些符号是合法的,数据如何书写,函数怎样命名,有些什么句型及句型的含义和书写格式,一个程序如何构成以及编程技巧等等。游戏机外接教学键盘,必须插上FBASIC语言卡。这些卡上有编制好的专用程序,如卡通图案,电子音乐,计算器等,有关这些专用程序的用法,可参阅有关厂家的说明书,本讲座只介绍FBASIC的常用知识及编程方法。

二、FBASIC语言的构成

1、FBASIC所用的字符

FBASIC语言中规定使用如下字符:

a. 26个英文字母,有大、小写之分,一般使用大写字母。

b. 0~9十个阿拉伯数字。

c. 算术运算符:+(加)、-(减)、*(乘)、/(除)、MOD(求余数)。

d. 关系运算符:=(等于)、>(大于)、<(小于)、>=(大于等于)、<=(小于等于)、<>(不等于)。

e. 逻辑运算符:AND(与)、OR(或)、NOT(非)、XOR(异或)。

f. 分隔符:,(逗号)、:(冒号)、"(双引号)、'(单引号)、;(分号)、—(连号)、.(点)。

g. 其它符号:$(字符串变量)、#(半音)、&(十六进制符号)、?(代替PRINT)。

2、数据

FBASIC使用的数据有两类,一类是数值数据;一类是字符串数据。

a. 数值数据为整型数。整型数可以是正整数或负整数。不含正负号的数认为是正整数。最小的整数是-32768,最大的数是+32767。FBASIC不识别实型数,即没有小数(在进行算术运算时可以有小数)。遇到小数,计算机自动舍去小数点后的数字,且不遵循四舍五

入法则。

b. 字符串数据。用双引号括起来的一串字符叫字符串。它也可以作为数据参与运算和处理。字符串中的字符可以是字符集里除引号外的所有字符。(由于引号要作为字符串的标志，所以字符串中不能再包含引号。)字符串最长不超过 31 个，如下一些是字符串数据：

"FBASIC YU YAN HE CHENG XU"

"FCS-90 IS AMICROCOMPUTER"

"X,Y,Z="

"(50X-3A)/6A="

3、变量

变量有简单变量、下标变量两种。

简单变量又分整型变量和字符串变量两种。字符串变量也称串型变量。

简单变量的名称可以用一个、二个、三个字符表示，表示规则为：

×××

最后一个字符表示变量类型。$ 表示串型变量，没有这个字符表示整型变量。

可以是字母或数字，也可以没有。

第一个字符必须是字母。

除了变量类型符号 $ 之外，变量名也可以超过两个字符，变量名称最长可以到 255 个字符(但只能是字母和数字)。计算机会按照你命名的变量名称存贮和打印，但运算时它只识别头两个字符，因而 ABC1 和 ABC2 代表同一个变量 AB，EOOP 和 EOCK 是同一个变量，当然 EOOP 与 EOOP$ 不同。

在 FBASIC 语句中使用了一些特定的词，如 GO-TO、FOR、THEN 等，这些词在 FBASIC 中都有确定的语法意义，它们叫保留字，我们设定的变量名不能是保留字，也不能包含保留字，否则计算机会把它当做保留字对待，并不认为是变量。

变量的类型应与数据类型一致。整型变量只能取整型数，字符串变量的值只能是字符串常数，即某个具体的字符串。以后还会看到，下标变量所能取的值也是数值或字符串。下标变量留待第二章介绍。

4、数学运算的运算符，运算规则与表达式

a. 运算符：

FBASIC 中的数学运算符与数学中的运算符基本相同。

加号＋，减号－，乘号 *，除号/，求余数 MOD。

注意：表达式中"*"号不能省略，也不能用"·"代替。如 5*Y 不能写成 5Y 或 5·Y，A*B 不能写成 AB 或 A·B。

b. 运算规则与普通代数中运算规则一样。如果表达式中有括号，则先算括号的内容，然后是函数，乘除、求余数、加减、关系运算，逻辑运算。它们的优先顺序是( )→函数→*，/→MOD→＋，－→＝、<>、>、<、>＝、<＝→NOT→AND→OR→XOR FBASIC 中的括号无大、中、小之分，一律用园括号( )，可以在( )再套( )。

c. FBASIC 表达式

数学中，用运算符、括号把常数、变量连接起来的式子叫代数式。FBASIC 语言中用符合 FBASIC 规定的运算符和括号把常数、变量连接起来的式子叫 FBASIC 表达式。下面列表举例说明表达式的写法：

代数式	FBASIC 表达式	错误表达式
$\frac{A}{B}\div\frac{C}{D}$	(A/B)/(C/D)	A/B/C/D
a[x+b(x+c)]	(A*(X+B*(X+C))	A(X+B(X+C))

下期刊登三、FBASIC 语句和程序。

《全国中小学计算机教育资料汇编》简　介

为总结我国"七五"期间中小学计算机教育方面的经验，向广大教师、中小学校及有关教育管理部门提供计算机教育方面的信息，以促进计算机教育事业的发展，"全国中小学计算机教研中心"和电子工业出版社编辑出版了《全国中小学计算机教育资料汇编》，内容包括：

1. 有关部委及省、市领导部门关于计算机教育方面的文件；

2. 中央领导同志的题词与讲话；

3. 国内外计算机教育工作会议纪要及学术活动；

4. 教师的优秀论文及研究报告；

5. 中学计算机课教学大纲及有关规定；

6. 全国中学计算机配置及教师队伍情况的调查；

7. 全国及各省市青少年计算机竞赛与竞赛题，国际信息学奥林匹克竞赛题；

8. 全国教育软件评审机构、评审标准、优秀软件简介；

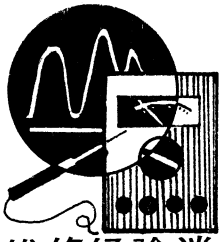
9. 开展计算机教学的中小学名单，科技馆、青少年宫名单；

10. 有关计算机普及教育的管理部门及学会、协会等机构通讯录；

11. 全国出版的青少年计算机类图书、报刊目录；

本《汇编》是我国计算机教育方面的一本较全面的资料，可供中学师生、计算机教研工作人员、中小学校领导及各级教育管理部门参考。

本《汇编》16 开本，75 万字，定价 13.50 元，1991 年 8 月出版。



维修经验谈

AT(286)微机硬盘处理一例

刘文锋

AT DIAGNOSTICS 是专门用于 PC—AT(286)机，而无法用于 PC—XT 机的诊断测试软件，常用于帮助 AT(286)机使用者判断计算机的系统板、键盘、磁盘及驱动器等装置是否在正常操作状态和故障的过程。还有对硬盘进行低级格式化处理等功能，由于计算机内装有大量的数据信息，这一功能不敢偿试使用，其使用方法也不得知。为了同行在使用 AT 机的过程中，遇到硬盘出现问题，或者被染上病毒，需要对硬盘进行处理时，不至于走弯路，现将操作步骤介绍于后。由于篇幅限制，仅给出有关的屏幕显示，供参考。

1. 装置 AT DIAGNTICS 的盘片入 A 驱动器，进行冷启动或热启动，屏幕出现菜单，有 6 种选择项：

0—SYSTEM CHECKOUT （系统诊断）
1—FORMAT DISKETTE （格式化软盘）
2—COPY DISKETT （复制软盘）
3—PREPARE SYSTEM FOR MOVING （锁系统）
4—SETUP （参数设置）
9—END DIAGNOSTICS （结束）

2. 选择系统诊断“0”并回车，等待约一分钟，给出配置情况。各机器的配置不一样，显示也不同！

1—SYSTEM BOARD （系统板）
6—2 DISKETTE DRIVE (S) AND ADAPTER （2 个软盘驱动器及接口）
17—1 FIXED DISK DRIVE (S) AND ADAPTER （1 个硬盘驱动器和接口）
IS THE LIST CORRECT (Y/N)? （正确否?）

3. 如显示与实际配置不相符，可按“N”并回车，选择补充正确后再继续。一般是相符的，选择“Y”并回车，进入系统诊断菜单。

0—RUN TESTS ONE TIME （诊断一次）
1—RUN TESTS MULTIPLE TIMES （诊断多次）
9—END SYSTEM CHECKOUT （结束）
SELECT THE ACTION DESIRED? （选择操作号）

4. 选择测试一次“0”并回车；其显示的选择项与步骤“2”显示的相同，其要求是：选择要操作的号或按回车全部诊断。

5. 选择硬盘驱动器“17”并回车，对硬盘进行操作。进入

FIXED DISK DIAGNOSTIC MENU （硬盘诊断菜单）。
1—WRITE,READ,COMPARE(ON TEST CYLINDER) 读写，比较(测试柱面))
7—FORMAT MENU （格式化菜单）
9—RETURN TO CONTROL PROGRAM （返回主控程序）
FOR OTHER OPTIONS （其它操作，键入操作数，驱动器标识(1,C)并回车）
TYPE THE OPTION NUMBER,DRIVE ID (1,C),
AND PRESS“ENTER”

6. 选择格式化菜单“7”并回车，对硬盘进行格式化处理。进入

FORMAT SELECTION MENU （格式化选择单）。
1—CONDITIONAL FORMAT （条件格式化）
2—UNCONDITIONAL FORMAT （非条件格式化）
9—RETURN TO FIXED DISK MENU （返回硬盘菜单）
FOR OTHER OPTIONS （其它操作
TYPE THE OPTION NUMBER, （键入操作数，驱动
DRIVE ID(1,C),AND PRESS“ENTER” 动器标识(1,C)并回车）

7. 键入“1,C”并回车，即条件格式化。“1,C”为硬盘的标识。

8. 询问继续否，回答“Y”，开始对硬盘进行低级格式化，需要时间约十分钟。

9. 格式化完后，根据提示进行选择，退出诊断系统。

10. 用 AT(286)机的系统盘从新启动机器后，用 FDISK 建立分区，大于30兆的硬盘，还要建立扩展 DOS 分区，并建立逻辑 D 盘。活动分区只能为硬盘的主 DOS 分区 C 盘，逻辑 D 盘不能作为活动的 DOS 系统盘。

11. 用 FORMAT C:/S/V 和 FORMAT D:高级格式化后，用 COPY 命令或 RESTORE 命令装入 DOS 文件及实用程序与数据。

(上接第 29 页)

(3)数据第七位表示农历月比公历月滞后信息。置 1 时农历月份比公历月滞后 2 个月数，置 0 为滞后 1 个月数。

(4)数据的第八位表示大小月信息。置 1 时该农历月为大月，置 0 农历月为小月。

例如查表得 89 年 6 月 2 日的数据为 0DDH。即 2 进制数 11011101。分离开来即 89 年 6 月 2 日时，农历为 89 年 4 月 29 日。4 月为大月有 30 天。农历转换流程框图如图 4 所示。

多功能钟还可在此基础扩展有功能秒表、节日的显示、农历 24 节气的显示转换、临时加闹、数字脉博计、数字温度计等。从而使单片机更多发挥本身功能多功能时钟整个占 2K 字节 ROM。

从屏幕显示异常判断苹果主机板故障

杨州工学院计算中心　张艾美

苹果微机及其兼容机在我国推广使用已有多年，目前不少机器正处于故障率较高时期。由于苹果微机及其兼容机的原理和结构相对于其它功能更强的微机来说要简单些，尤其是对那些具有两台以上同类型号苹果机的用户来说，在维修过程中使用替代法或比较法，即使没有多少维修经验，也能将一般的故障排除。问题是对仅有一台苹果机或兼容机的用户来说，怎样在经济代价不大的原则下，使用普通的万用表和高频示波器，自己动手解决检修问题。笔者根据多年微机维修实践，在此介绍一些从苹果机及其兼容机屏幕所显示的异常现象入手，分析解决检修问题的方法，供同行参考。

故障现象	故障可能出现的位置及需检测或更换的芯片
通电后电源指示灯亮，屏幕上只有光栅显示	①时钟产生区，检查时钟发生器的 A_2、B_1、B_2、C_1、C_2。 ②视频地址发生器，检查 D_{11}、D_{12}、D_{13}。 ③影响产生区，检查多路数据选择电路部件 A_9、A_{10}。
通电后屏幕上只出现绿黑色边框，没有任何图形或文字	检查存储器 RAM 的外围电路即或门 H_1，与非门 E_2，多路选择器 E_{12}、E_{13}、J_1，地址缓冲器 H_3、H_4、H_5，地址译码器 F_2 和数据寄存器 B_5、B_6、B_7、B_8 等。

故障现象	故障可能出现的位置及需检测或更换的芯片
通电后屏幕上出现无规则的图形（线条、字母、符号或方块等）	①时钟产生区，检查与非门 A_2，数据选择器 C_1 及计数器 D_{11}。 ②影响产生区，检查反相器 C_{11}，或门 C_{14}，高分辨率数据选择器 A_8，视频发生器部分的 A_{12}、B_{11} 以及脉冲控制器 B_{13}。 ③程序控制区，首先检查 CPU 的外围电路 H_4 地址缓冲器及 H_{10}、H_{11} 数据缓冲器，再检查 ROM 部分的 3—8 译码器 F_{12}，I/O 部分的 3—8 译码器 H_{12}，RAM 部分的选择电路 E_{14}，数据选择器 C_{12}，以及与非门 D_2 等。

由于苹果机主机板上的芯片都是插在插上的，更换芯片非常方便，关键是找准故障的部位或器件。一台故障机通电后，必然在 CRT 屏幕上显示异常现象，几种常见故障及其对应部位和需检测或更换的芯片，列于下表中：

根据上述屏幕显示异常，进行分析、检修的若干实例：

例 1、开机后屏幕上出现绿黑色边框，框内无图形或文字。

屏幕上的图形或文字来源于 RAM 中被选中的地址单元信息的输出，现在没有输出信息，应先检查 RAM 部分数据寄存器的芯片 B_5、B_8，如果其中有一个芯片损坏，则信息送不到视频发生器。经检查 B_8（74LS174）六 D 触发器的 6 脚、7 脚逻辑关系不对，6 脚输入脉冲器，但 7 脚却输出低电平，即 B_8 芯片损坏，换片后正常。

例 2、同例 1。

首先按例 1 查 B_5、B_8 均正常。再检查地址缓冲器 H_3、H_4、H_5（74LS367），发现 H_4 中第 2 脚输入端有波形，而对应的第 3 脚输出端无波形，即 H_4 芯片损坏，换片后工作正常。

例 3、开机后屏幕上只有光栅，而无文字或图形。

查时基脉冲区的有关芯片，发现 B_2（四异或门 74LS86）的输入端 10 脚、9 脚与输出端 8 脚逻辑关系不正确，即 8 脚无脉冲信号输出，致使视频发生器部分接收不到 14MHZ 的脉冲信号，而使影象产生区没有同步信号，无法正常工作，换片后正常。

例 4、同例 3。

查影象产生区 A_{10} 芯片（74LS194 移位寄存器）的 3、4、5、6 脚有输入波形，而 12、13、14、15 输出端无电平，即 A_{10} 损坏，换片后，屏幕显示正常。

例 5、开机后屏幕上出现规则宽边框和中间有无规则线条图案。

用示波器按顺序先检查时钟产生器的输入、输出逻辑关系都正常，再在影象区查 C_{14} 芯片（二输入四或门），发现 9 脚上有 R/W 脉冲波形，10 脚上有 $\varnothing_1$ 脉冲波形，但 8 脚输出不符合二输入或门的逻辑关系，即该组或门损坏了，换芯片后，屏幕画面正常。

例 6、开机后屏幕上呈现中间垂直深黑宽线条，左右两旁呈对称的淡黑色横线条。

首先检查时钟产生区，发现 C_1 芯片（74LS153）的 11 脚、13 脚、14 脚有脉冲波形，而输出端 9 脚无脉冲波形，致使时序电路同步脉冲输出不正常，整机同步信号不正确，因此视频电路输出不正常而造成屏幕画面混乱，即 C_1 损坏，换片后工作正常。

例 7、开机后屏幕呈现无规则的粗细不同的横线

条，且线条之间有稀散的字符。

检查时钟产生区和影象区均无损坏芯片，检查程序控制区，在RAM地址部分，E_{11}芯片(74LS153)2脚无脉冲选择信号，而芯片插座对应点却有脉冲信号，可见是芯片与插座接触不良造成，切断电源，用无水酒精清洗该插座及芯片腿脚后，再将芯片插入，开机后画面正常。

例8、开机后，屏幕出现规则或不规则排列的小黑方块图案。

检查时钟产生区及影象区未发现异常现象，再检查程序控制区，RAM中的C_5损坏或C_7损坏都会使屏幕呈现此现象，换片后工作正常。

例9、开机后屏幕呈现不规则的黑绿色边框及无规则的竖线条图案。

经检查发现在程序控制区中C_{11}的13、14脚一组反相器坏，致使H_3、H_4、H_5三态门不能正常工作，使视频部分输出波形混乱，换片后问题解决。

例10、开机后系统不能进入BASIC工作状态，屏幕上显示：

```
                3---- PEAKS
F12A-   A=C0  X=00  Y=03  P=35  S=F1
*▌
```

或

```
                3---- PEAKS
F150-   A=C0  X=00  Y=00  P=37  S=EB
*▌
```

且喇叭发出嗒嗒响声。

经检查并非主机板芯片损坏，而是某些芯片接触不良所造成，只要将主机板上每个芯片用手指往下按，使之接触良好，机器就恢复正常，若主机板积尘较多，可用无水酒精清洗后，再将芯片重新插入，机器即可正常工作。

M2024型打印机故障检修一例

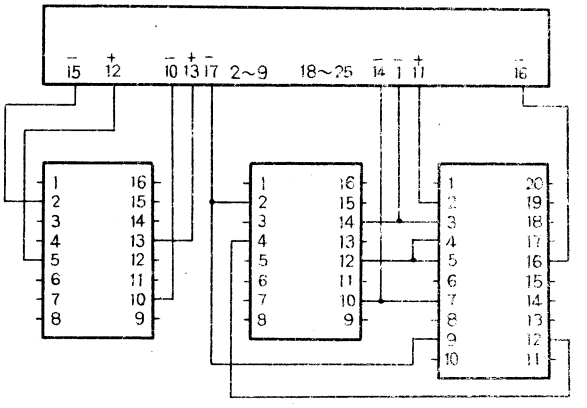
烟台师范学院　王　峰

有一台打印机(M2024型)用不到半年出现下列故障：连通电源打开电源开关，机器自校正常，但不能联机，并且打出不规则的变形码。

故障分析与检修：我们看该机控制信号部分的接口电路，如图所示：

Centronics并行接口：

1:$\overline{\text{strobe}}$，2～9:DATA。	10:$\overline{\text{ACK}}$
11:BUSY；	12 PE(PAPER ERROR)
13:SELECT；	14:$\overline{\text{AUTOFEED}}$
15:$\overline{\text{FAUIT}}$；	16:$\overline{\text{INIT}}$
17:$\overline{\text{SELECTIN}}$；	18～25:GND。



信号说明：

(1)$\overline{\text{DATAS}}$ $\overline{\text{TROBE}}$(CPU→PR)，读数据选通脉冲。脉冲宽度最低需要1μs，正常状态下为高，变低时读出。

(2)$\overline{\text{ACKNLG}}$(CPU←PR)，这是4±1μs的应答脉冲，连文字输入完毕或者打印动作结束时产生，但是输入DC3信号后变成BUSY状态，不能输出ACKNLG信号。

(3)BUSY(CPR←PR)表示打印机能否动作的DC电平信号；

(4)DATA(CPR→PR)数据信号，脉冲宽度最低需3μs。

(5)PE(CPU←PR)打印纸不正常时为高电平。

(6)SELECT(CPU←PR)打印机选择状态时为高电平。

经我们检查发现是U_{23}(74LS240)烧坏。更换后仍不能打印。进一步检查打印机控制接口电路，发现上述信号均能正常传送，但7406无取反作用，更换7406后，故障排除。之所以导致此类故障是因为电网电压波动太大所致。

计算机设备故障检修表

咎金省　宋靖涛

笔者从事计算机维修工作，就近年来所接触的部分故障作以下小结，供同仁们参考和借鉴，如遇相同类型的设备发生故障，不妨测试如下部位，一般尚可解决。

故障设备	现象	部位	可代元件
614型交流稳压器	2A交流保险烧断	控制整流晶体二极管2ZD14M击穿	
	电压不稳	大功率偏置电阻烧断或电子管坏	
王安电源	无电压输出，保险烧断	压敏电阻RV250短路，开关管2N6675烧坏短路。	C1942或BUX48
王安12寸显示器	无显示	行输出管BU407坏或行输出变压器高压包局部短路	
王安硬盘控制卡	起动失败	查专用芯片L4WD1100-05 L23 WD1100—01	
王安512K内存扩充卡	I/O系统错	内存芯片4164坏，具体位置可用王安诊断测试代码。	4864
M2024打印机	自检正常，联机后死机	接口板7414和74LS06易坏	74LS14
	无+36伏电源电压	调整管TR22 D1026坏	
AR3240打印机	无交流输出	变压器坏（内部保护电阻烧断，急用可短接此电阻）	
AR2463打印机	字车步进电机不能被驱动	B825，3A保险电阻坏	B601
GW100显示器	无光	行输出管V403 2SD1365	BU407
	水平一条亮线	TDA1170坏	
GW300显示器	无光	行输出管D1426坏	
GW286主机	实时时钟CMOS信息丢失	换锂电池（6伏）	
IBM PC/XT	PARITY CHECK1	奇偶校验RAM芯片4164或奇偶校验芯片74LS280损坏	
GW286EX	机器加电自检RAM时死机，无错码显示。	查数据收发器U6　(74F646) U12 (74F245) U9　(74F245)	
360K软盘驱动器	读错	磁头脏，用清洗盘清洗；或磁头小车卡死，上润滑油。	
PULSE 2000R型UPS	逆变器不工作	末级驱动晶体管QM50H—H坏	
苹果机电源MEQ—063P	无输出电压	调整管C2502坏 TL494CN坏（无锯齿波形）	BU508 BU407

IBM PC/XT 微型计算机检修一例

平　凡

故障现象：

开机时，出现一长一短“嘟”声后，系统不自检，不启动。

故障分析与检修：

IBM PC/XT 机一旦系统板上 8088CPU 出了故障，或系统时钟工作不正常，再就是数据总线有损坏元件，都将导致以上故障现象的出现。

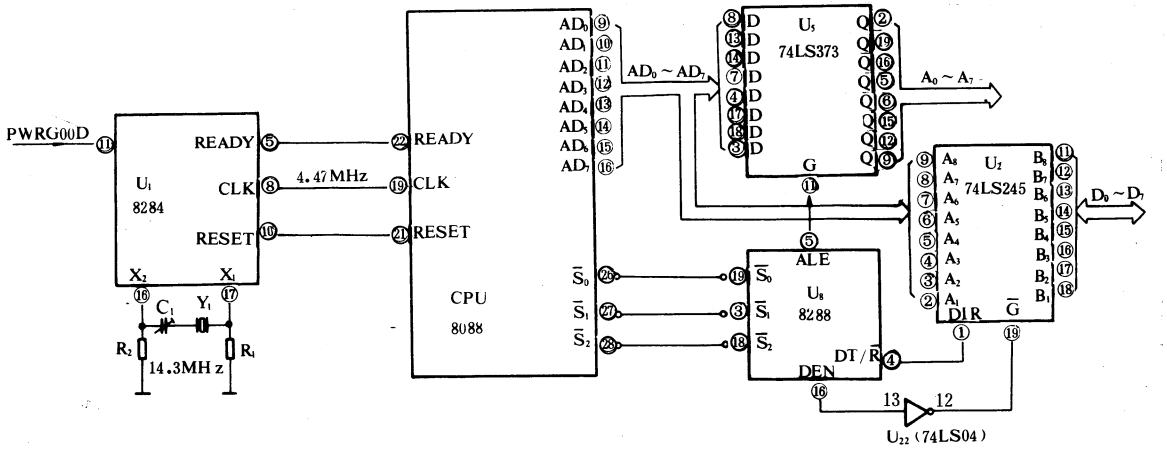
首先，开机用示波器测如图所示 8284 时钟发生器引脚⑧(CLK 时钟)，有脉冲波形信号，再测 CPU8088 引脚⑲(CLK 时钟)，也有脉冲波形信号，且波形很正常。再次开机时，测 8088 引脚㉑(RESET 复位)有一个明显的正脉冲(该脉冲是先从低电平变为高电平，进而再恢复为低电平)。因此，基本说明系统板上的 CPU8088，时钟发生器 8284 电路没问题，通过测 8088 引脚㉒(READY 准备好)为高电平，推测故障可能出在数据总线电路上。

这里，先了解一下数据总线的工作过程，如图为系统板数据传送情况。

在时钟发生器 8284 引脚⑪内部，由斯密特触发输入电路和同步触发器组成的复位逻辑电路，与 8284 引脚⑪外部 RC 网络构成加电复位电路。在开机时，加电复位电路便产生 PWRGOOD(电源好)脉冲信号，使复位逻辑电路进入工作状态，从 8284 的引脚⑩输出一个先从低电平升为高电平，进而再恢复为低电平的正脉冲复位信号。该复位信号进入 8088 引脚㉑使 8088CPU 复位，复位后的 CPU 便从 $FFFF0 单元开始执行指令。如图可见，数据总线是通过 V_2(74LS245)数据缓冲器与 8088CPU 的数据线连接的。由于 8088CPU 的数据线引脚(AD_0～AD_7)又兼地址低 8 位(A_0～A_7)的引脚，所以，8088CPU 引脚⑨～⑯采用分时传送方式将地址与数据信息分别传送，即在不同时间里，既传送数据信息，又传送地址信息。8088CPU 的总线控制器 V_8(8288)为 20 脚双极型芯片，它能根据 8088 执行指令时提供的状态信号$\overline{S_0}$、$\overline{S_1}$、$\overline{S_2}$建立控制时序，输出相应的地址锁存允许信号 ALE 和读写控制命令。图中的 CPU 地址锁存器 V_5(74LS373)用来锁存地址信息。当 8288 引脚⑤输出的地址锁存允许信号 ALE 为高电平时，地址锁存器 V_5 的锁存控制端 G(引脚⑪)接受 8288 输出的 ALE 信号开始工作，把 CPU8088AD_0～AD_7 送来的地址信息锁存选通到 V_5 中，然后送往地址总线。另一方面，8288 总线控制器根据 8088 的状态信号$\overline{S_0}$、$\overline{S_1}$、$\overline{S_2}$，从 8288⑯脚发出的数据总线允许信号 DEN，经过 V_{22}(LSO4)反相器反相后，至双向数据收发器 V_2(74LS245)引脚⑲，控制 V_2 工作，使 8088 的数据总线和系统数据总线相接。同时，8288 引脚④发出数据发送/接收信号 DT/$\overline{R}$ 至 V_2 引脚①，以控制数据传输方向。当 DT/$\overline{R}$=1 时，数据从 8088CPU 经 V_2 传送到数据总线；当 DT/$\overline{R}$ = 0 时，数据从总线经 V_2 传送到 8088CPU。

如果机器正常的话，其数据总线上的数据信息将处在不断变化之中，用示波器观察将显示出变化无常的断续波形。测试结果，数据总线上无波形信号。于是，测试双向数据收发器 V_2 的输入和输出端，均为低电平，进而再测试 V_2 引脚⑲收/发控制端，无论开机瞬间或开机之后，总在高电平上。在这种状况下，数据收发器 V_2 是不能正常的。造成 V_2 引脚⑲总为高电平的原因有三个：一是 V_2 本身出现故障；二是反向器 V_{22}不正常；三是总线控制器 V_8 损坏当用示波器测反相器 V_{22} 输入引脚⑬时，示波器显示有波形，而 V_{22}输出引脚⑫无波形，断定故障出在反相器 V_{22}芯片上。因为 V_{22}损坏，使 V_2 数据信息的收发功能失控，导致 8088CPU 指令和数据信息道中断，引起机器不能启动。

更换 V_{22}(LS04)芯片后，微机恢复正常工作



MP—I 微电脑快速检修法

青岛大学　李　红

本方法只用万用表就可在20分钟内检查出MP—I机90%以上的常见故障。

检查前，先用万用表测量一下印刷电路板上各芯片的V_{CC}和GND两脚，$V_{CC}=5V$，GND=0V。如若不是，说明电源引线开路或断路。

一、显示器故障的检修

1、显示电路的检查

将U_7(74LS00)从电路板上卸下，则DMA开关停止工作。由于键盘上的地址三态门和数据三态门都处于高阻态，因而键盘与地址总线、数据总线断开。又因DMA=1，MPU也与总线断开。

再将U_{18}(74LS02)卸下，则存储器2114的片选端$\overline{CS}=1$，因此存储器也与总线断开。

由地址译码电路可知，此时输入接口电路U_{20}(三态门74LS244)被选中，但其8个数据输入端悬空(高电平)，因而输出端为1。输出接口U_{21}(8D触发器74LS374)未被选中，对总线无影响。

此时，10根地址线悬空(高电平)，8根数据线为1，因此，18个LED应全亮，哪个灭灯，说明故障在哪条线路上。

然后，取下万用表上的一根表笔用作短路线，一端接地，另一端分别触碰10根地址线(电路板上左上角处的10根地址线端口(沉铜孔))和8根数据线(板上右上角8根地址线端口)。碰到哪根，相应LED就应灭。否则，说明这条线路上存在故障。

2、数据总线的检查

将8位数码开关拔下，改插到输入接口插座，此时，由于输入接口导通，因此8个开关可以直接控制数据总线。

①令8位数据开关分别置0，相应的LED灭，这说明数据线$D_7\sim D_0$之间彼此无短路现象。

②令8位开关全部置0，用万用表分别检查U_{12}(6502)，U_{16}(2114)，U_{17}(2114)、U_6(74LS244)各片上的数据端引脚也应为0；再将数据开关全置1，各相应引脚也应为1。否则说明某数据线上存在故障。

显示器故障的原因有：LED损坏或两个引脚焊反，U_1、U_2、U_3(反相器74LS04)芯片个别引脚断路，其插座焊点开路或断路，线路板上引线断路或短路，沉铜孔开路等。

显示器检修完毕，将U_7，U_{18}插回原处。

二、键盘故障的检修

1、DMA键或OFF键不正常

先查DMA开关电路。按过DMA键后，U_7的6脚(即DMA信号线)应为1，3脚($\overline{DMA}$信号线)为0。按过OFF键，则6脚、3脚电平翻转。DMA开关电路出现故障的可能原因：按键本身短路或开路，U_7引脚开路或短路、外围阻容元件开路或损坏等。如果DMA开关电路一切正常，则继续检查U_5的8、9、10脚处的三态门是否正常。

2、A_0、A_1、A_n键不正常

A_0键抬起时，U_9、U_{10}、U_{11}三片的2、3脚都为0；按下A_0键，2、3脚为1。

A_1键抬起时，U_8的1、2脚为1，3脚为0；按下时，1脚为0，3脚为1，而2脚仍为1。而3脚输出的信号应能送到U_9计数器的时钟端$\overline{CP_0}$(14脚)。而U_9的11脚输出进位信号应能送到U_{10}的$\overline{CP_0}$端，同理，U_{10}11脚应送到的$U_{11}$$\overline{CP_0}$端、

A_n键抬起时，U_8的2脚为1；按下时，由于接通了振荡电压(约为1.2～2.7V)，因而U_8的2脚电压为1.2～2.7V，这时，地址应能连续加1。

上述三键出现故障的可能原因有：按键开路或断路，U_7～U_{11}引脚接触不良，外围阻容元件开路或损坏等。

三、存储器的故障检修

1.检查存储器片选端$\overline{CS}$

由地址译码电路可知，当地址$A_9A_8=00$、01、11时，选中存储器$00、$01、$11页，此时两片2114的$\overline{CS}$端为0，$U_{20}$的选通端(1脚及19脚)应为1，$U_{21}$的CP端为1。当$A_9A_8=10$时，选中I/O接口，这时，两片2114的$\overline{CS}$端为1，$U_{20}$的1脚及19脚当$A_0=0$时为0，$A_0=1$时为1，$U_{21}$的CP端为1。

如果出现异常，说明地址译码电路有故障，可检查U_{20}、U_{21}各引脚与总线连接情况。

2.检查存储器与读写线R/$\overline{W}$的连接情况。

按一下DMA键后，则U_3的8脚($\overline{DMA}$信号)为0，此时$\overline{WR}$键抬起，两片2114的R/$\overline{W}$端应为1；按下，则2114的R/$\overline{W}$端为0，如果异常，说明2114与读写线的连接有故障。

3.检查存储器与地址总线$A_0\sim A_9$的连接情况。

将8位数据开关全置0，利用A_n键及$\overline{RW}$键，将存储器$00、$01 $11页全清零($01页为I/O接口地址)。

然后在$000单元写入$FF。如果存储器与地址总线连接良好，则用键盘重新读数据，除$000单元为$FF，其余均为$00。如果存储器与某一根地址总线断路，例如2114的7脚(A_2)与地址线A_2断路，则此脚电平将处于不稳定状态：可能为0，也可能为1。若为1，则当用键盘给$000单元写进数据$FF时，存储器的实际地址将为$A_9A_8A_7A_6A_5A_4A_3A_2A_1A_0$=00 0000 010 0，因而$FF实际写入$003单元。这样，读数据时，会在$000和$003两个单元都读出$FF。

当2114的7脚处于不稳定状态0时，用同样方法可以查出错误。

4、检查存储器与数据总线$D_0\sim D_7$的连接情况，这在前面一、2中已进行。

四、微处理器故障的检修

按一下OFF键，地址LED全亮，按住RES键，MPU将发出一个随机地址，地址LED显示这个随机数。当RES键抬起，LED又全亮。如有异常，用万用表检查6502的40脚($\overline{RES}$)的电平变化：按下RES键为0，抬起为1。如果不是这样，应检查RES键与U_8的4、5、6脚以及$\overline{RES}$信号总线之间的连接情况是否良好，以及U_8右边阻容元件是否断路或损坏，。其次应检查U_{15}构成的时钟发生器是否振荡，振荡时各引脚的电压大约为1.2～2.7V(12脚为0)，停振时各脚电平为0或1。如停振，可在U_{15}的11、13脚之间并联一个1KΩ的电阻。如仍不起振，说明U_{15}引脚断路、或外围阻容元件断路、损坏。

经以上检修后，如果MPU仍不执行程序，则说明6502与地址总线的连接存在问题。可将6502从电路板上卸下，按一下OFF键，此时地址总线受6502地址端9～18脚控制，由于6502已卸下，使U_{13}、U_{14}的输入端悬空，(下转26页)

如何使九针打印机字迹更清晰

陆秀忠

目前，中华学习机，APLLE 机配备的打印机基本上是九针打印机。其打印针一般为片状针。在这些打印机使用日久后，打印字的笔划明显变粗。当打印小号等中文字时，字迹便不清晰。这对于使用打印纸打字时，还问题不大。若是使用打印蜡纸，则油印出来后，笔划多的字就变得模糊一团了。当然，此时可调换打印针。可是，一来打印针并非每个地方都能买得到，二来还有一个经费上的问题。如果刚好你的打印机正是如此，请别心焦。按下面的方法，便可使你的打印机针头恢复正常或者变得更细一些。下面以 CPA－80 为例。

一、工具准备：中、小号十字头螺丝刀各一把，300 号以上水砂纸一片或者粒度相近的油石一块，软毛刷一个，放大镜一个，小镊子一把，可架在打印机上的平板一块，其长、宽为 15×5cm。

二、拆卸打印头：不要把机器从机箱拆出来。仅用螺丝刀把打印头和电缆的固定螺丝拆下就行了，打印头便可拿出到打印机外。此时注意，由于有电缆连结，不能用力拉扯，以免损坏电缆，接着，把可见到一列 9 支打印针头的那一面盖板从打印头上拆下，便可见 9 支片状打印针上下整齐迭成一叠，建议拆到此为止。若把每一支打印针都拆下来，要打磨得一致是很不容易的。

三、打磨针头：这一部分工作最要细致。每支打印针的前半部如图。粗一些的针头大小约为 0.4×0.4mm 的矩形。现要将其磨成 0.2×0.2mm 的矩形。打磨共分三步。

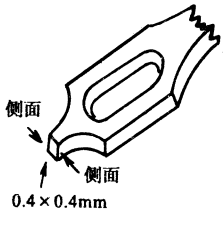
第一步，打磨九支并排针头的侧面。把平板架在打印机上，用镊子夹紧九支打印针，把针头部分压在平板上，使九支针头整齐一致。然后用油石或者一小片水砂纸包在钢锯片的外面打磨两侧，每侧磨去 0.1mm。

第二步，打磨每支针头的上下面，也是每面磨去 0.1mm。此时要分支打磨。把要打磨的那支针用镊子夹住，稍向一边移出。然后用水砂纸包在钢锯片处仔细打磨。建议磨完每支针头同一个面后，把打印头翻转，再磨每支针头的另一面。这样以免记不清那一支磨过。然后检查各针头磨得是否一致，进行一些修整。

第三步，用水砂纸轻擦每支针头的边、角，使之不致太锐利。

打磨注忌事项：(1)为防止磨过头，要经常用放大镜检查针头的形状。(2)在下方用纸垫好，以免铁屑掉进打印机。(3)要经常用软毛刷把针头上的铁屑轻轻刷去，注忌刷时针头朝下，以免铁屑掉进打印头内。(4)对针头不可用力扳动，特别是磨打印针头上下两面时更要注忌，以免打印针变形。

四、试机：这时要特别小心，把打印头重新装好，便可开机试打，由于针头变小，其冲击力便增大，再加上新磨的针头边角锐利，所以很容易打破色带。建议先不装色带，用一张金相砂纸作为打印纸，其砂面要面向打印头，利用自检，把砂纸打几遍，以磨钝针头的边角。并可从纸上字迹的深浅来了解打印力度的大小，以便调整打印头的位置使力度恰到好处。万一力度调不小，可把在打印头上用来固定打印头的螺丝孔靠胶轮那边用圆锉稍稍锉去一些，使打印头的位置可退后一点；打印力度便会小一些。



打印针头图

下面，给出打磨前与打磨后两种字迹的比较如下。有兴趣的朋友不妨动手一试。

YOKO747 型教学语言学习机的修理

郭长福

该教学语言学习机使用灵便，实用性强，特别是放音卡的走带速度可以根据需要而连续可调，很适合学校外语教学的需要。但在实际使用中，也发现该型机存在不少问题，在修理 40 多台该型机中，总结几点共性问题如下：

1、该型机中的一些插接件，特别是 2 卡的磁头插座极易发生接触不良现象，具体表现为用 2 卡放音时无声，扬声器发出“嘟嘟”的叫声。解决的办法是把该插接件去掉，引线直接焊在印刷电路板上即可。

2、该机如果使用小型稳压电源供电时要求所用电源的输出电流必须足够大，否则，当 1 卡和 2 卡同时工作时，稳压输出电压降低过多，带速变慢，声音产生变调现象。笔者所修理多台此型机中，都是因为稳压电源带负载能力差，放音变调，而一些使用者用提高输出电压的方法来加以弥补（该机额定直流电压为 6V，有的使用者把电压调到 9V）。造成的后果是烧坏该机的功放集成电路 TBA820M。分析其原因，系稳压电源带负载能力差，当电机工作时消耗电流较大，电源输出电压降低，在电机工作时，把稳压直流输出调到 6V 左右，当电机停止工作时，电压马上回升到 9V 左右，这样，电压瞬间的升高，很容易将功放块烧毁。多台该型机功放集成电路的损坏，都是该原因所造成的。

3、为使修理人员对故障的集成功放块的检测有据可查，笔者在维修实践中总结了功放集成电路的电压电阻数据：

功放集成块 TBA820M 正常电压：

单位：伏

引　脚	1	2	3	4	5	6	7	8
IC_1	0.7	0.55	0	0	3	6	5.6	4.5
IC_2	0.66	0.6	0	0	3.3	6	5.6	4.5

功放集成块 TBA820M 正常电阻：

单位：KΩ

引　脚	1	2	3	4	5	6	7	8
IC1	1.1	1	3	0	0.85	0.8	0.9	5
IC2	1.1	1	0	0	0.85	0.8	0.9	5

MF10 型万用表，RX100 档黑表笔接地时测得。

微机软盘驱动器“卡盘”的简易修复

乐冬青

“卡盘”(盘片抽不出)是软盘驱动器常见的故障之一。关于它的修复方法一般都是采用松开软盘驱动器浮动压紧轮螺丝,调整压紧轮架的位置,或是拆下压紧轮架,用小刀将浮动压紧轮外圈刮去一层塑料。但用这两种方法修复的软驱,使用效果都不甚好。因为“卡盘”故障不是因压紧轮架位置不正确造成的,而是因浮动压紧轮变形造成的;由于浮动压紧轮是伞片状塑料轮,很不容易刮得均匀,一不小心,就会使压紧轮失圆,造成软磁盘定位不准。笔者对“卡盘”故障进行了较细的分析,发现“卡盘”故障从现象上看,好象是浮动压紧轮变形造成,实际上是浮动压紧轮与内压紧轮卡位不能复位造成。其工作过程如下:当软盘驱动器小门关上时(读、写磁盘信息,浮动压紧轮支架被压杆压下,内压紧轮又压下浮动压紧轮,这时伞片状浮动压紧轮被撑开,同时将软盘片卡压在软驱主轴凹轮内。软驱操作结束,驱动器小门开关打开,浮动压紧轮支架压杆抬起,内压紧轮借助复位弹簧,(内压紧轮与浮动压紧轮之内的弹簧)弹起,被撑开的浮动压紧轮恢复原状。这时如内压紧轮不能弹起,浮动轮就一直处于撑开状态,于是就出现了“卡盘”,对此,我们采用了一种简便的修复方法。

首先打开机壳,用老虎钳拆下浮动压紧轮轴上的卡片,取出浮动压紧轮,将轴上的宝塔弹簧拿下,再用仪表小起子拆下内压紧轮上的卡片,取出内压紧轮,用锉刀将内压紧轮的外圈锉去一层,然后装回轴上,再将宝塔弹簧、浮动压紧轮装回轴上,卡好卡片就行了。装好后,可检查一下。用手指将内压紧轮压到底,然后松开手,如内压紧轮顺利弹起,说明复位良好,否则,将内压紧轮再锉一层,直至内压紧轮复位自如。

CEC-I 软驱故障维修一例

潘　冬

CEC-I 型中华学习机,由于价格适宜、功能较强,已逐步进入家庭。但是由于它的驱动器接口不够完善,如使用不慎易使驱动器出现故障。下面谈谈一例维修经验

一、故障原因及现象

由于安装驱动器(建南 JN—510A 型)时不慎,只插了上面一排插针,造成了驱动器故障。驱动器读软盘操作时正常,但不能进行写操作及格式化磁盘。驱动器在工作时指示灯不亮,当停止工作时指示灯反而发亮。

二、故障的排除

检修时,首先让 CEC—I 主机自检,确定其完好。因了解到驱动器损坏是由于插错位造成的,所以把检修重点放在驱动器接口周围的电路。打开驱动器的外壳,此电路板设在驱动器的底盘上,板上有几块集成电路和其它元件。驱动器的工作指示灯由 74LS05 控制,用万用表检查 74LS02 各脚的逻辑关系,确认 74LS05 损坏,更换后驱动器工作指示灯故障消除。但仍不能进行写操作,认为是驱动器写保护电路的毛病。检查红外线光电耦合器是否失效。用有写保护的磁盘和无写保护的磁盘分别插入驱动器,测得 74LS05 的输入端 11 脚有变化,而输出端 13 脚无变化,确认 74LS02 损坏。更换后用驱动器测试盘检查一切正常,故障消除。

三、总结

上述故障是由于接口与电缆线的插头错了位,只插了上面一排插针造成的。为此,同行们务必引以为戒。同时在更换集成块时,最好使用集成电路插座,以防不测。

求　援

美国产 TMS9900A 型 16 位家用计算机,音频信号和视频信号经 RF 调制器后,电视机(全制式,要求用 NTSC 制式)能收到载波,但无图象无声音。希能提供 TMS 产品的有关资料或线索。

通讯地址:长春市东北师大电教系

联系人:刘杰

邮编:130024

中华学习机小蜜蜂 1(XMF-1)内一片门列阵坏,型号:HKE　610　605　8820 何处有售。

联系地址:河北保定建设路化工部第一胶片厂设计室

孙长令　邮编:071054

浙江温州市精密电子仪器经营部

地址:温州市飞霞南路龟湖路

邮编:325004　　电挂:5072　　电话:336072

开户:工商五马办　　帐号:1567892

供应(单位)EPROM2716/32,12 元;2764/128,10 元;27256,15 元;RAM6116,15 元;6264,25 元;6117,10 元;8031/8085/8155,15 元;80C31,24 元;8748,35 元;8255,10.5 元;8035/8039,11.5 元;D780(Z80CPU),7.2 元;Z84C00(低功耗型 Z80CPU),10.8 元;8243,12.5 元;82C43,16.8 元;8251,16.5 元;82C51,23 元;8212/8253,14 元;8257,18 元;8279,22.5 元;光隔 TLP521-4,7.5元;TLP521-1/541-1,2 元;TLP541-2,3.5 元;晶振 3.579M,2 元;4M,3.2 元;6M/8M/10M,3.6 元;集成插座 40P,0.65 元;以上邮资 2 元/次。进口开关电源成品板,输入 AC220V/110V 转换;输出+5V/7A,−5V/0.11A,+12V/1.7A,−12V/0.4A,+115V/0.4A,外形尺寸(毫米)232×178×35,价 188 元,含邮资。另有系列 74LS,CD4000 电路及数百种其它集成电路,晶体管,继电器等均为进口器件,函索价目表者,请用贴上肆角邮票信封,写清回邮地址寄我部。

“坏”软盘的再生

张　智

有的新软盘，由于厂家装盒不当，造成盘套变型无法使用，只好弃之，但这类软盘可通过以下方法再生。

检查：

将中指与食指插入软盘的驱动圆中转动一周，同时通过读写槽观查盘片有无明显的痕迹，若有痕迹，此盘就无法再生。

复原：

取打印纸两张，把软盘放于上下两纸之中，平整地放在光滑的桌面上，找一平底瓷盅（底面积要大于软盘面积），内装沸水，将瓷盅表面的水擦净，压在纸上面，两分钟后将软盘翻面，再将瓷盅压在纸上，两分钟后取出软盘，马上放置于16开的厚书之中，二十分钟后取出，此盘已经复原。

将该盘进行格式化，若格式化不成功，可找一张同容量的软盘格式化后，用DISKCOPY命令（格式化的好盘作源盘，再生盘作目标盘）进行盘备份后，再将再生盘格式化，通过以上工作，使手中积累多年的“坏”盘得到了再生。

GW286BH系统板维修一例

董纯坚

1、故障现象：自检正常，且能进硬盘，但不认软盘，硬盘转软盘时显示“驱动器读错”。

2、分析与排除：该故障属于引导和自举软盘操作系统时失败，这与DMA电路、软盘适配器锁相式数据分离器有关，也与软盘驱动器方面的数据读出电路，磁头定位偏差及磁盘片的好坏有关。首先取已确认是好的磁盘片，插入软盘驱动器中，故障未消除，然后依次检查并替换了软盘驱动器、信号电缆线、软盘适配器，故障仍未消除，看来故璋肯定出在系统板的DMA电路上，因为DMA电路有故障，就会导致引导块装入到不正确的RAM区域，引导块程序就不能正确执行，DOS不能自举，致使系统无法进入DOS系统。用示波器检查DMA电路的有关信号，发现CHIPS门阵芯片82C206的66脚ADSTB信号（地址选通信号，高电平有效）虽为高电平，但它所连接的八D锁存器74F573的输入端却为低电平，且复位时，仍无变化，这说明82C206的66脚与74F573的输入端有开路现象，卸下82C206发现该芯片第66脚已断，连接两点，故障消除。

怎样使用清洗盘

瞿新国

清洗盘是用来清洗磁盘驱动器磁头的专用工具。当磁盘驱动器工作时间长了之后或用了质量低下的软盘后，磁头上就会附着磁粉，影响软盘信息的正确存取，甚至根本不能读取。这时，可使用清洗盘清洗，但使用不当反而会缩短磁盘机的寿命。

怎样正确使用清洗盘呢？将清洗液涂在清洗盘上，要尽量均匀，将清洗盘放入驱动器，关上仓门，此时对那些关上仓门磁盘就转动的机器（如一些PC系列机）让它们转30秒即可，对关上仓门磁盘不转的机器（如APPLE Ⅱ系列机），用启动或调盘命令使驱动器转30秒。一般情况下，磁头业已清洗干净。

注意：不要用不涂清洗液的清洗盘清洗磁头。因为清洗盘较粗糙，不湿润时对磁头磨损较大；另外，磁头上的磁粉附着力较强，被清洗液浸润后容易脱落，也易被清洗盘吸附。还有，清洗时间不要过大，否则，同样会磨损磁头。

《计算机辅助设计与图形学学报》正式公开发行

由中国计算机学会主办的《计算机辅助设计与图形学学报》已被国家科委批准，将于1991年第四季度开始正式向国内外公开发行。该刊已获得国内统一刊号CN11－2925/TP，国际标准刊号ISSN 1003－9767。这是我国CAD和计算机图形学领域第一个公开出版的学术性刊物。该刊面向全国，刊登CAD与图形学领域里的学术论文，CAD与图形学的科研、开发与应用的最新成果和学术动态，及时反映该领域学科的发展水平与发展方向。该刊于1989年8月创刊，暂为季刊，已内部出版了9期。此次被批准公开发行，将会进一步促进CAD和计算机图形学的发展，更好地为我国国民经济建设及计算机事业服务。该刊编辑部设在中国科学院计算技术研究所。

LASER 机 GA004 电路的代换和修复

张保田

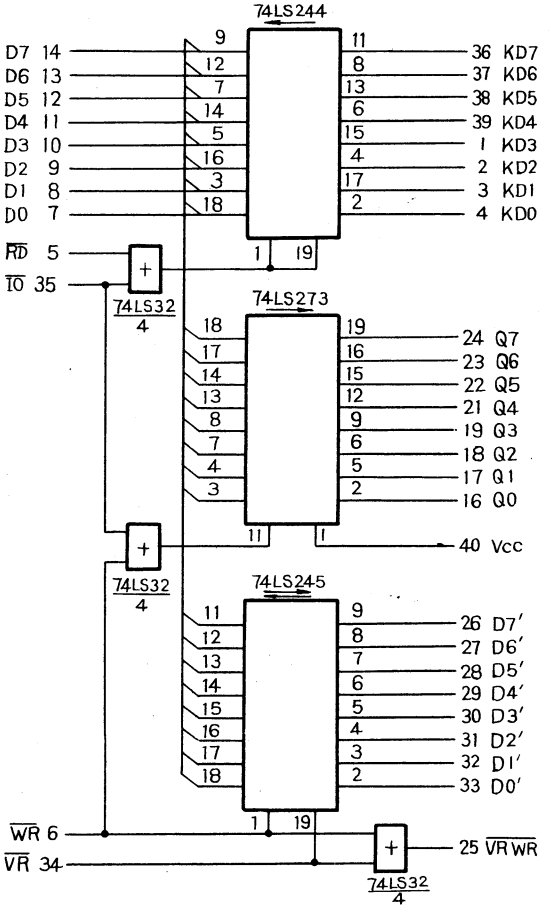
近年来我们修复了一些 LASER 机，其中大规模集成电路 GA004 损坏率较高，约占总故障率的一半。但这种芯片是原整机生产厂家专为 LASER 定做的，目前市场上根本无货供应，所以在维修时采用了整块电路代换和局部修复两种方式。

一、用 LS 系列代换 GA004

GA004 是 40 线封装的大规模集成电路，其引线标号及在主机中的连接见本刊九一年一期封三。引线定义和功能如下：

$\overline{RD}$ 由系统总线来的读信号，低电平有效表示 CPU 从数据总线上读 I/O 口或存储器数据。

$\overline{WR}$ 由系统总线来的写信号，低电平有效表示 CPU 向 I/O 口或存储器写数据。



图一　GA004 代换电路

$\overline{IO}$ 实际是一个存储器空间译码信号，当 CPU 读写 6800H 到 6FFFH 空间时为低，此空间在 LASER 上作 IO 用。

$\overline{VR}$ CPU 读写显示存储器时的译码信号，低电平有效，地址是 7000H 到 77FFH。

$\overline{VRWR}$ 其准确逻辑式为 $\overline{VR}+\overline{WR}$，简写为 $\overline{VRWR}$ 为低时 CPU 写 6116，$\overline{VR}$低而$\overline{WR}$高时 CPU 读 6116。

D0 到 D7　来自系统总线，三态双向数据线。

KD0 到 KD7　接键盘矩阵，单向数据线，当$\overline{RD}$和$\overline{IO}$同时为低时 CPU 读键盘矩阵，检测用户按键情况。

D′ 0 到 D7　与 6116 相连的双向数据线，方向见 VRWR 的说明。

Q0 到 Q7　GA004 内部 8D 锁存器的输出端，当 $\overline{WR}$ 和 $\overline{IO}$ 同时为低时 CPU 写 8D 锁存器，各端口作用：

Q0 和 Q5 接蜂鸣器，由软件控制 Q0 和 Q5 总是反相，在两端电平连续翻转时蜂鸣器发声。若使 Q0 和 Q5 同相蜂鸣器就不会发声了。

Q2 磁带记录电平输出端。

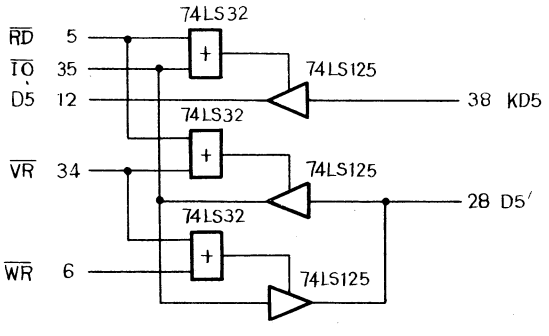
Q4 屏幕底色控制端，为低时绿色屏幕，为高时橙色屏幕。

Q3 屏幕显示状态控制，为低是文本状态，为高时图形状态。

还有三个锁存端主机未用，所以参考电路中没有画出，它们对扩展功能是有用的：

Q1、17 脚主机硬件电路未接，Q2 系统正是利用它锁定高显状态。

Q6，23 脚和 Q7，24 脚，如果扩展系统仅为 305 和 310 机用，此两端可以利用；但如同时要在 200 上用则不宜用 Q6 和 Q7，因为大部分 200 机内部只有 6D 锁存器。



图二　GA004 局部修复电路

根据以上分析可以用四片 74LS 系列中小规模芯片组成 GA004 的等效代换电路，电路见图一，图中大矩形框内是 LS 电路本身的引线编号；而框外是与 GA004 对应的 40 线引线编号。四片 LS 电路装在一块小印刷板上，由印板焊出 40 条引线插入原 GA004 位置使用。

二、局部修复

有时 GA004 只是局部损坏，此时可设法采用较少的器件有针对性地只修复损坏部分。维修中曾遇到一实例：数据线 D5－D'5 和 D5－KD5 都损坏不通，但锁存器 D5 位可写入。这种情况下如用图一代换电路则成本高功耗大且自制印板并不容易。所以用了 74LS32 和 74LS125 各一片修好 D5－D'5 和 D5－KD5 通道。LS 芯片用万能胶粘在 GA004 上省去制印板的麻烦，电路如图二。74LS125 是四路缓冲驱动器，注意由于其中有两路是输入输出端相互并接的，所以电路一定要反复检查确认连线无误后再开机，否则容易烧毁电路芯片。据用户反映按图二修复的微机已正常工作了半年。

浪潮 0530—AT 死机故障恢复一例

董国钦

故障现象：一台 LC0530 机，在用机过程中，突然掉电，事后开机可自检，但进入记忆联想汉字显示后死机，热启动无效（软盘可启动，使用硬盘）。

故障分析：低于 40MB 的硬盘微机在系统突然掉电后，磁头不能自动退至主保护道内，这时如果搬动机器或磁头正在读写磁盘，磁头很容易划伤磁盘，而造成硬盘的操作系统不能使用，甚至于不能进入、使用硬盘，初步估计为磁盘文件或磁盘损坏，

故障排除：刚开始笔者尝试着利用以下方法来恢复故障，但均无效：

1. 利用 DOS 盘 A＞SYS C：传送系统，然后重新装入联想记忆汉字系统，启动无效。

2. 利用 FDISK 软件，删除 DOS 分区，然后重新建立 DOS 分区（注意：如果此时建立的 DOS 分区与原来的 DOS 分区不一致，将导致硬盘信息的丢失）重新启动，故障仍未排除，进行操作系统格式化 A＞FORMAT C：/S 后，硬盘可进入使用，但当调用记忆联想汉字系统时，故障又出现。

确定为硬磁盘磁道损坏，需要对硬盘进行低级格式化，利用浪潮机随机测试盘：

A＞LC0530 显示菜单：

0—SYSTEM CHECKOUT

1—FORMAT DISKETTE

2—COPY DISKETTE

3—PREPARE SYSTEM FOR MOVING

4—SETUP

9—END DIAGNOSTICS

选择 0 进行系统测试，接下按 Y，再按 0 进行单次测试运行，显示菜单：

1—SYSTEM BOARD

2—1024KB MEMORY

3—KEYBOARD

6—2DISKETTE DRIVE(S) AND ADAPTER

……

17—1FIXED DISK DRIVE(S) AND ADAPTER

……

选择 17 进行硬盘处理，显示菜单：

FIXED DISK DIAGNOSTIC MENU

……

1 — WRITE, READ, COPARE (ON TEST CYLINDER)

……

7—FORMAT MENU

9—RETURN TO CONTROL PROGRAM

选择 7，C 进行硬盘格式化处理，显示菜单：

FORMAT SELECTION MENU

……

1—CONDITIONAL FORMAT

2—UNCONDITIONAL FORMAT

……

选择 2，C 对硬盘进行无条件低级格式化，最后利用 FDISK 建立 DOS 分区，进行操作系统高级格式化，至此故障排除。

微机硬盘系统出故障时，不要急于对硬盘进行低级格式化，而要从其它途径想办法恢复，如果确诊是硬盘磁道或扇区损坏，且高级格式化失败，则要进行低级格式化，剔除盘面损坏的块，我们建议用户在搬动机器或关机前，进行磁头复位工作。

一九九一年计算机初级程序员有奖竞赛试题参考答案

（共 150 分）

题号	小题号与答案										得分	
一	1	AHCIE	2	BD	3	C	4	AC	5	DA	12	40
	6	D	7	B	8	A	9	C	10	D	5	
	11	B	12	CB	13	B	14	AC	15	AC	8	
	16	AFFID	17	D	18	CE	19	D	20	C	10	
	21	C	22	B	23	A	24	B	25	D	5	
二	1	运算器　控制器	2	匈亚利　存储 存储　串行	3	下一条　当前					8	30
	4	指令译码存放结果	5	立即　直接　间接	6	定义　通讯					7	
	7	385　3080	8	输出　非击打式 串行式	9	阴极射线管显示器					6	
	10	256　1FF3H	11	纯音 顺序 纯形 多	12	收集　存储　加工					9	
	13	软磁盘									1	
三	1	DIR A:*.PRG									2	12
	2	MD A:\BAS\BA1									2	
	3	COPY CON A:BA1\AA.TXT									2	
	4	COPY A:\BAS\BA1\AA.TXT+A:\DB3\AA.BAK A:BAS\BA2\ALL.BAK									2	
	5	REN A:BA2\ALL.BAK BA2\ALL.TXT									2	
	6	DEL A:BA1\AA.TXT 或 ERASE A:BA1\AA.TXT									2	
四		COPY CON B:PC.BAT CLS PAUSE 姓名　单位 COPY A:TREE.COM B:				TREE B:/F TYPE B:PC.BAT TYPE B:PC.BAT ^Z						8
五	1	① REPLACE ALL 平均分 WITH(语文+数学)/2									1	20
		② LOCATE FOR 平均分>90									1	
		③ COPY TO BBB FOR SUBSTR(姓名,1,2)='王'									1	
		④INDEX ON 性别+STR(平均分,6,2) TO CCC									1	
		⑤ LIST FOR 民族<>'汉'FIELDS 姓名 OFF									1	
		⑥ COPY STRUCTURE TO DDD FIELDS 姓名,性别,平均分									1	
		⑦ LIST FOR 出生日期 ≤CTOD('12/31/65')									1	
		⑧COPY TO EEE SDF									1	
	2	1 2 1 2 4 1 8 9									4	
	3	①Record#　姓名 3　李夺 Record#　姓名 4　买买切		②Record#　姓名 3　李夺 4　买买切 8　李洋		③	4	④	3		8	
六	①	1 3 6 10 15 21 28 36 45 55									3	12
	②	2 212 21012 212 2		③	21 21		④	5 8 11 14 17			9	
七		INT　((I−2)/6)　INT((I−3)/7)　S+I　N+1										6
八	①	A(I,2)*A(I,3)　I+1　A(I,4))A(J,4)　J,K　6　T									6	12
	②	流程图(略)									6	
九	1	D	2	H	3	I	4	E	5	J	5	10
	6	G	7	C	8	B	9	F	10	A	5	

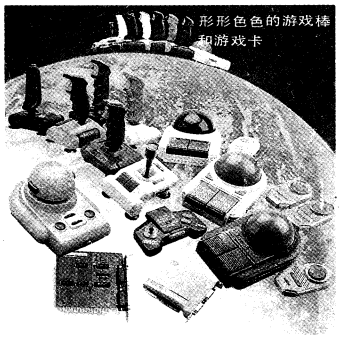
ISSN 1000-1077

電子與電腦

11

一九九一年

总期第80期



電子與電腦

ELECTRONICS AND COMPUTER

目　录

·综述·

在中华机上扩充中文输入码的探索 …… 胡志群、胡志广、胡沛云(2)

·PC 用户·

利用 PASCAL 扩充 dBASE Ⅲ 的函数功能 …… 王　越　张秀邦(4)

如何动态生成有记忆功能的菜单 …… 苏士俊(6)

在 dBASE Ⅲ 下实现菜单的光带键选 …… 阮自力(7)

双列分页打印程序 …… 陶　锋(8)

广州大学一号病毒解除与免疫 …… 张文锋(10)

谈谈 2.13 汉字系统在软盘上的使用 …… 廖红旭(11)

PC/XT 机病毒报警程序 …… 王伟智　刘秉瀚(12)

考生姓名编码自动生成程序 …… 任绥海(12)

·学习机之友·

把 APPLE DOS 移入 RAM 卡 …… 傅　剑(14)

将程序 CHAIN 升级为 APPLE DOS 命令 …… 赵江龙　孙杰亮(15)

向天坛机移植中华机汉字软件的一些体会与经验 …… 张　斌(17)

FORTH 问答：堆栈和逆波兰表示法 …… 丁志伟(18)

中华学习机图书管理程序 …… 王二珠(19)

ProDOS 磁盘操作系统入门(续) …… 廖　凯(21)

·C 语言初阶讲座·

第九讲　指针与函数、结构 …… 李文兵(23)

·学用单片机·

汽车转弯信号灯控制 …… 张培仁　刘振安(26)

·学装微电脑·

Z80 CPU 与单片微处理器 8031 的交换 … 高殿斌(28)

·电脑巧开发·

在 APPLE 机接口卡中固化 BASIC 应用程序 …… 解武杰(30)

设计微电脑控制电路的小常识 …… 王　林(31)

型号不同的绘图仪绘制 TANGO 软件图形的方法 …… 黄　健　陈　军(32)

·游戏机电脑·

键盘电视游戏机的 FBASIC 语言及程序设计(下) …… 于　春(33)

·维修经验谈·

驱动器修理一例 …… 张红平(35)

维修小经验 …… 张中弦(35)

硬磁盘驱动器专用接口电路 501262 的测试 …… 卜建辉(36)

IBM－PC/XT 一种系统错误的分析 …… 姜金友(41)

用非格式化方法修复硬盘一例 …… 周建辉(42)

检修 IBM AS/400B35 故障二例 …… 王志远(42)

自己学修计算机 …… 陈　勇(43)

TEC-I 机磁盘驱动器 I/O 接口板故障检修一例 …… 崔时珍　玄勇活(45)

3070 打印机断针定位方法及 8088 汇编程序 …… 龙兵生(46)

打印机故障维修——以 STAR 或 GX-15 打印机为例 … 王　威　王　侯(47)

普及型 PC 机上可用的印刷电路板辅助设计软件 smARTWORK …… 张保田(48)

显示器故障检修一例 …… 张　智(48)

封面　形形色色的游戏棒和游戏卡

封二、封三　微型计算机电源原理图

封底　电子工业出版社计算机新书介绍

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：张　丽

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

在中华机上扩充中文输入码的探索

胡志群　胡志广　胡沛云

"杞码"是一种普及型的中文输入码，是广西民族学院李冠盛同志研制的，已经通过了技术鉴定。它具有编码规则简单、重码较少、易学、易记、易用等优点，很适合中、小学生应用。而中华学习机是我国普及中、小学电脑教育的机型，能把"杞码"开发应用于中华机是很有意义的。即将出版的《速查学生字典》及《速查实用成语词典》两书也把"杞码"作为正文的编排查检码。这就使得人们在学会"杞码"查字法的同时，也为学"杞码"汉字输入法打下基础。

由于中华学习机的内存有限，直接把"杞码"调入内存 RAM 中是不可能的，所以首先要把"杞码"的信息压缩，让它转换成存储码，再固化到 EPROM 中。本文就是说明按什么规律进行压缩；并说明在扩充程序中如何实现把输入码转换成存储码，再和已固化的码表比较，找出对应的汉字；同时讨论扩充程序与原机内中文管理程序的衔接及调用其子程序等问题。

一、如何打出原机内中文管理程序

为使我们的扩充程序与原有的中文管理程序能相互衔接，并尽可能的调用管理程序中有关的子程序，使扩充程序更简练，故必须打出机内管理程序进行分析。

中华机所用的微处理器 6502 可寻址 64K，其逻辑地址与物理地址不是一一对应的，物理地址分主存和辅存，整个中文系统是在辅存中，所以不能直接打出其管理程序，必须编制一定程序利用软开关的切换把它调入主存后才能由打印机打出。

程序中设四个单元，分别存放管理程序新、旧的首地址，然后令其逐一转移，在开始取数前，先执行子程序 $C3AB，其目的是选择辅存 2。转移完毕执行子程序 $C3B2，让其转主存 1 再返回。

二、码表的转换

一、二两级汉字共 6763 个，每个汉字有五位码。如果直接把杞码存入，则需要 33K 多的内存储单元，这对内存有限的中华机来说是不现实的。为此我们按一定的规律把杞码转换成存储码。这样每个汉字只要两个存储单元就可以了，节省了 3/5 的内存，只占用 13K 多一点就够了。

在杞码表中，应用 15 个英文字母作部首，再加上六个笔形码，在杞码中笔形码是用数字 1—6 表示，但 1—6 的数字键跟选择汉字提示的指令键冲突，因而改用部首没有用到的几个英文字母代替，Q、E、U、I、O、P 分别表示 1—6，现在反过来连同部首的 21 个英文字母分别用 1—21 的数字表示，其对应关系为：

Q	E	U	I	O	P	B	C	G	H	J	K	M	N	R	S	T	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

这样我们把杞码所对应的五位数字码分别以 X_4、X_3、X_2、X_1、X_0 表示，则

存储码 $=X_4 \cdot 7^4+X_3 \cdot 7^3+X_2 \cdot 7^2+X_1 \cdot 7+X_0$

结果是小于或等于 16 进制的四位数，只用两个单元存储即可。

这个转换工作是让计算机完成的。我们只要把杞码以"T"文件的方式打入计算机，并转存在磁盘上，由于两级汉字的数据太多，故分成四个文件，通过一定程序自动的进行变换，把十进制转换成十六进制，并把转换后的数据自动的存贮在磁盘中。最后把这四个文件合并，就成为杞码的存储码了。

三、码表的固化

虽然把杞码转换成存储码，节省了 3/5 的信息量，但仍有 13K 多的字节，如果把它存在磁盘中，每次使用时再转入内存，这样余下给用户的内存 RAM 空间就太少了，因此有必要把这码表固化在 EPROM 中。我们将它固化在 U_7 的低地址部分，工作时由计算机直接到 U_7 查此表即可。

我们的码表只占用了 U_7 低地址的一部分，因而在固化前必须把我们的码表与 U_7 的余下数据合并后再写入新片中。以后要再写则直接利用磁盘的经整理后的数据即可。

四、扩充程序的编制

1. 编制换指针程序以解决程序间的衔接

中华机以"F_4"作为用户自定义的功能键。现在我们把它作为扩充输入码的状态转换控制键。换指针程序的任务是修改 $391、$392 的内容，使它从原来的 RTS 改为显示杞码状态字的入口地址，并把状态标志单元 $03AE 置为"$F_4$"的键码（# $94），把扩充的处理字符程序的入口地址送入 $38F 与 $390 单元。这样只要用户按下"$F_4$"键，就能显示"杞码："这状态提示字。此后再键入的可显示字符（大于或等于 $A0）或退格键（$88）都转到扩充的输入处理程序入口地址去执行。

2. 编制退格键处理程序

退格键码 $88 是通过累加器转到扩充程序的，可仿效区位及拼音输入时退格键的处理方式，并直接调用其子程序处理，不必考虑软开关的转换。

3. 编制杞码字符的处理程序

首先解决杞码字符的转换与显示。如前所述我们键入杞码时是用 Q、E、U、I、O、P 键代替杞码的数字码 1—6 的，但为了直观还必须通过一定的程序，使在字符区显示的仍然是原杞码中的数字。

杞码一般为 5 位码，不足 5 位的用"]"键结束。遇

到这样的结束符，由程序控制其在键入区自动补零，补足5位为止，无须通过键盘一一键入，因此可提高键入速度。

程序的一部分用于把输入的5位码转换成存储码。然后与码表进行比较。码表所表示的汉字是按区位码的顺序排列的，查找时设一个与区位码对应的指针。当输入码经转换后与码表比较，结果相同时可由指针找到它所对应的汉字，这样把对应于输入码的所有重码字都找出来，放在一定的区域中，并在汉字区显示头六个汉字。还可根据换页指令“>”键显示下6个汉字提示。或根据“<”指令键显示前一幕提示汉字。用户可在每一页中用1—6的数字键选提示字送系统。这种功能是调用拼音输入法所对应的子程序完成的。

以上是扩充程序的主要部分，还有一些细节，如出错时自动响铃并自动返回等。

五、编制使杞码运行的程序

我们是采用软硬件结合的方法，扩充程序没有固化，这样可以随时修改，以增加它的功能。因此在开始使用杞码输入前，首先要运行换指针程序，还要把字符处理程序由磁盘送入内存。这些任务是由BASIC程序实现的。此后凡按“F_1”键即可使用。

本项目在研制过程中得到华明公司袁晓岚先生、清华大学卓小越老师以及我区计委计算站及计算中心等热情的支持与帮助，特此致谢。

附录：

“杞码”——又称笔型码的编码和检字方法

汉字的笔型分六种（见表一）：横（包括提）、竖、撇（包括左向点）、捺（包括右向点）、左向折、右向折，分别用代码1、2、3、4、5、6表示，记作“杞”字笔型代码。

表一　汉字“杞”字笔型代码表

代　码	笔形名称	笔　形	同类笔形
1	横、提	→	一 ㇀
2	竖、垂	↓	丨
3	撇、左向点	↙	丿 ㇒ ㇓
4	捺、右向点	↘	乀 乛
5	左折	↲	𠃌 ㇆ ㇇ ㇋
6	右折	↳	𠃍 ㇈ ㇉ ㇊

取码方法：

1. 字的起笔（即书写时的第一笔）不在表二所列常见部首内时，按写字的顺序依表一取五笔（前三笔加末二笔）“杞”字笔形代码，不足五笔的全取。

例：部　41452　　干　112

　　乏　3454　　盛　13521

2. 字的起笔在表二所列常见部首内时，先取相应部首代码，后按写字的顺序依表一取四笔（前二笔加末二笔）“杞”字笔形代码，不足四笔的全取。

例：萍　C4412　　逛　G1154

　　马　M　　伐　R1634

表二　汉字部首代码表

代　码	部　　首
B	贝　疒　宀
C	艹　虫
G	广　犭
H	火　禾
J	钅　金
K	口　口
M	门　马　木　目
N	女
R	饣　礻　日　曰
S	氵　纟　石
T	土　扌
W	王
X	讠
Y	讠　衤（衣）　鱼　月
Z	足（足）　⺮（竹）

（上接第32页）

```
1040 IF c1=44 THEN 1100
1041 IF c5=1 THEN 1070
1042 x(j)=c1
1043 j=j+1
1060 GOTO 1020
1070 y(k)=c1:k=k+1
1080 GOTO 1020
1100 c2=c2+1
1110 IF c2=2 THEN 1130
1111 c5=1
1120 GOTO 1020
1130 LPRINT CHR$(13)
1131 IF x(6)=200 THEN 1135
1132 x(4)=(X(1)-48)*100:X(5)=
     (x(5)-48)*10:x(6)=x(6)-18
1133 x1=x(1)+x(5)+x(6)
1134 GOTO 1138
1135 IF x(5)=200 THEN 1137
1136 x(4)=(x(4)-48)*10:x(5)=x(5)
     -48:x1=x(4)+x(5):GOTO 1138
1137 x(4)=x(4)-48:x1=x(4)
1138 IF y(6)=200 THEN 1141
1139 y(4)=(y(4)-48)*100:y(5)=(y(5)-48)*10:
     y(6)=y(6)-48
1140 y1=y(4)+y(5)+y(6):GOTO 1158
1141 IF y(5)=200 THEN 1143
1142 y(4)=(y(4)-48)*10:y(5)=y(5)-48:y1=y
     (4)+y(5):GOTO 1158
1143 y(4)=y(4)-48:y1=y(4)
1158 GET #1,g:g=g+8
1159 c3=ASC(a$)
1160 c3=c3-48
1161 x=c3*COS(0):y=c3*SIN(0)
1162 x2=x1+x:y2=y1+y
1163 LPRINT "M";x2;",";y2
1164 LPRINT CHR$(13)
```

（下转第29页）

PC用户

利用PASCAL扩充dBASE Ⅲ的函数功能

沈阳工业学院计算机系　王越　沈阳市工程咨询公司　张秀邦

许多用户都感到DBASE Ⅲ的计算能力较差，特别是函数功能欠缺，给系统的进一步开发带来困难。为此，我们分析了dBASE Ⅲ与外部接口的数据交换方式，利用具有丰富函数功能、运行速度快的TURBO PASCAL语言完成了dBASE Ⅲ的函数功能扩充。

一、基本思想

dBASE Ⅲ与高级语言间进行数据传送的方法，传统上一般以ASCII码文件(即.TXT文件)进行数据交换。这种方式在进行大批数据的交换时比较理想，而实际应用中往往需要把某几个参数直接传送出去，进行计算后返回。

dBASE Ⅲ提供了二种接口与外界交换数据：一是通过CALL命令，将参数直接传送给被调用的汇编程序，然后由汇编程序传送给高级语言程序。新版本的dBASE Ⅲ可以用CALL命令传送给高级语言形成的.BIN文件，但这种方法较为复杂，对一般基层用户不太实用。另一种方法是利用RUN命令与宏代换函数&将参数传送给高级语言。RUN命令的格式为：

RUN＜命令文件名＞WITH＜参数表＞

这里，＜命令文件名＞可为任意的后缀为.COM的可执行文件；＜参数表＞即为执行该文件需要的参数。dBASE Ⅲ规定，该参数表须为一系列值参。例如，我们可以写出下述调用格式：

```
RUN CALL_F WITH 'Y','=','COS',3.14159
```

这里的自变量是常量：如果它是变量的话，设为X，则下述调用

```
RUN CALL_F WITH 'Y','=','COS',X
```

dBASE Ⅲ认为是非法的。怎么解决这个问题呢？dBASE Ⅲ还提供了一个很有用的宏代换函数&，该函数的操作变量只能是字符变量，因此直接写成&X是不行的，只能先将X变换成字符串，然后代入该参数表。那么上述调用即为：

```
X1=STR(X,10,2)
RUN CALL_F WITH 'Y','=','COS',&X1
```

便可达到计算Y=COS(X)的目的。

根据这一原理，我们可以用高级语言编写一个命令文件，该程序对由RUN命令传送过来的参数进行接收、分析，然后计算并生成一dBASE Ⅲ命令文件，将结果写入该文件，在dBASE Ⅲ中执行该文件后，便完成了计算结果的回送。

二、扩充软件实现

利用上述方法，我们编制了一个软件，实现了对dBASE Ⅲ函数的扩充。程序组成如下：

- CALL_F.PAS：是一个用TURBO PASCAL编制的程序，经联接生成后，生成命令文件CALL_F.COM，其目的就是接收由dBASE Ⅲ程序提供的参数，对其识别后，根据计算公式计算出函数结果，回填到一个TRANSFER.PRG文件中。它的调用格式为：

RUN CALL_F WITH＜结果变量＞,＜=＞,＜函数名＞,＜自变量＞

- PASCAL_F.PRG：dBASE Ⅲ命令文件，将接收的参数进行类型转换等预处理后，调用CALL.F.COM计算，并执行TRANSFER.PRG文件，实现函数结果的回送。该程序调用时要求提供四个参数，从左到右依次为：＜结果变量＞,＜=＞,＜函数名＞,＜自变量＞。
- TRANSFER.PRG：回送结果文件，程序结构为：

```
PUBLIC ＜结果变量＞
＜结果变量＞=＜计算结果＞
```

正确调用CALL_F.COM后，执行它便得到计算结果。

在该软件中，CALL_F.COM对用户透明，用户可不去管怎样正确调用它，而只要提供PASCAL_F.PRG要求的参数形式即可。

下面是一个将数据库TEMPT.DBF中ANGLE取出，计算COS值后，将结果写到另一字段COS_VALUE中的应用程序。数据库结构如下：

Field	Field name	Type	Width	Dec
1	ANGLE	Numeric	10	
2	COS_VALUE	Numeric	10	5

程序为：

```
set talk off
use tempt
set devi to print
@ prow()+1,0 say '计算前数据库中数据显示:'
@ prow()+1,0 say 'ANGLE(角度)'
@ prow(),35 say 'COS_VALUE(余弦值)'
do while .not.eof()
    @ prow()+1,0 say ANGLE pict'9999999999'
    @ prow(),35 say cos_value pict'9999.999999'
    skip
enddo
go top
* 数据库测试
do while .not.eof()
  * 弧度换算
  TEMP_MV=ANGLE*3.14159/180
  * 计算函数值
  do pascal_f with 'sd','=','cos',TEMP_MV
  replace cos_value with sd
skip
enddo
```

```
go top
@ prow()+3,0 say '计算后数据库中数据显示:'
@ prow()+1,0 say 'ANGLE(角度)'
@ prow(),35 say 'COS_VALUE(余弦值)'
do while .not.eof()
    @ prow()+1,0 say ANGLE pict'9999999999'
    @ prow(),35 say cos_value pict'9999.999999'
    skip
enddo
use
*变量测试
abc=3.14159/3
do pascal_f with 'y','=','cos',abc
@ prow()+2,0 say'r='
@ prow(),5 say y pict'99999.9999'
@ prow(),10 say ' '
set devi to screen
return
```

运行结果:

计算前数据库中数据显示:

ANGLE(角度)	COS_VALUE(余弦值)
30	0.00000
60	0.00000
90	0.00000
120	0.00000

计算后数据库中数据显示:

ANGLE(角度)	COS_VALUE(余弦值)
30	0.86602
60	0.50000
90	0.00000
120	-0.50000

Y=0.5000

细心的读者会发现,上述程序对数据库字段操作时,先将 ANGLE 的值赋给一临时内存变量 TEMP_MV,这是因为 dBASE Ⅲ 的 RUN 命令只承认内存变量而不承认数据库字段变量的缘故。

有关程序清单附后,本软件只写了 COS 函数,读者可根据需要扩充,其实现大同小异,十分容易。

```
*CALL_F.PAS
program Call_functions_of_PASCAL;
Type str80=string[80];
var
    i:integer;
    operator:str80;
    operand,result:real;
    filevar:text;
procedure error(ss:str80);
begin
writeln(chr(7),ss);
assign(filevar,'transfer.prg');
rewrite(filevar);
close(filevar);
exit;
end;
function check_str(ss:str80):boolean;
var
    i:byte;
begin
if length(ss)<=0 then error('call function error!');
check_str:=true;
for i:=1 to length(ss) do
    if not (ss[i] in ['0'..'9','a'..'z','A'..'
Z','_','-'])then check_str:=false;
end;
function upcase_str(ss:str80):str80;
var
    i:byte;
begin
if length(ss)>0 then
    for i:=1 to length(ss) do ss[i]:=upcase(ss[i]);
upcase_str:=ss;
end;
function change_type(ss:str80):real;
var
    p:integer;
    value:real;
begin
val(ss,value,p);
change_type:=value;
end;
procedure write_result(ss:str80;result:real);
begin
assign(filevar,'transfer.prg');
rewrite(filevar);
writeln(filevar,'Public ',ss);
writeln(filevar,ss '=',result:20:5);
writeln(filevar,' ');
close(filevar);
end;
begin
for i:=1 to paramcount do writeln(paramstr(i));
if (paramcount<5)
    or(upcase_str(paramstr(1))<>'WITH')
    or(not check_str(paramstr(2)))
    or(paramstr(3)<>'=')
    then error('Parameters Format Error!');
operator:=upcase_str(paramstr(4));
operand:=change type(paramstr(5));
if operator='COS' then begin
    result:cos(operand);
    write_result(paramstr(2),result);
    end;
end.
*Name of the procedure:PASCAL_F.prg
Parameter LetVar,Equal,FName,SelfVar
str_data=str(selfvar,15,4)
run callf with &LetVar &Equal &FName &str_data
do transfer
return
```

如何动态生成有记忆功能的菜单？

北京计算机学院　　苏士俊

菜单生成技术，也常用来减少汉字输入。但简单的、常规的菜单却有时满足不了用户的功能要求。比如在录入学生或职工档案时，把民族的汉字名称和编号写成菜单，则用户只需输入相应的编号。但菜单中写出全部民族，不仅增加了选择的困难，也影响屏幕的美观，实际上有的民族可能根本用不上。因此可用动态菜单生成技术弥补其不足。

技术问题是这样解决的：在系统中专门设置一个民族编码提示库，用它来支持系统运行。库内只存放民族名称和对应的编码（用记录号代替编码也是可取的）。在用户输入民族字段的数据时，打开民族编码提示库（这一工作由程序中的语句完成），将其内容按一定格式显示在屏幕上，并询问有无需要的民族（Y/N）？这时用户即可选择相应的编码，代替汉字输入。当菜单中没有需要的民族时，只需键入 N，即可转去向民族提示库中增加一条新记录（汉字的民族名称和对应的编码）。该民族的名称和编码一旦写入数据库，下次再显示菜单时，新的民族和编码会永远出现在菜单中。这种有记忆办法的动态菜单会极大地方便用户。

下面是一个生成动态菜单的示范程序，凡是用户使用过的民族都可以自动出现在民族提示菜单中，其菜单内容会随系统运行而动态变化，用户不妨试一试。

存放民族名称的 MZTSK·DBF 库已有 7 条记录，其库结构和数据如下：

```
. LIST STRUCTURE
  库文件结构--      文件名:C:MZTSK.dbf
  库文件中的记录数量:          7
  库文件的最后更新日期:    01/23/91
    字段  字段名      类型      宽度     小数
      1   民族名称    字符型      4
  * *总计* *                    5
. LIST
  记录号:  民族名称
         1  汉族
         2  回族
         3  苗族
         4  壮族
         5  满族
         6  白族
         7  傣族
```

动态生成菜单的程序如下：

```
*民族菜单显示:
  SET TALK OFF
  RK=" "
  Y12=" "
  DO WHILE .T.
     CLEAR
  @3,8 SAY "* *民族提示* *"
  USE MZTSK
  SET HEADING OFF
  LIST
  SET HEADING ON
  IF RK="Y"
    EXIT
  ENDIF
  @1,8 SAY "如果没有需要的民族,"
  @1,28 SAY "请按 N 键,否则按其它键!"GET Y12 PICTURE "!"
  *增加新民族:
  IF Y12<>"N"
  @1,0 SAY SPACE(80)
  EXIT
  ELSE
  APPEND BLANK
  RK="N"
  DO WHILE RK <>"Y"
  @2,10 SAY "新民族:"GET 民族
  READ
  @2,30 SAY "认可吗?"(Y/N)?"GET RK PICTURE"!"
          READ
          @2,0 SAY SPACE(80)
        ENDDO
      ENDIF
    ENDDO
  *选择民族:
    N=0
    @1,8 SAY"请选择民族编码:"GET N PICTURE
      "99" RANGE 1,RECNO()-1
    READ
    @1,40 SAY "以下操作略…"
    USE
    SET TALK ON
    RETURN
```

①下面是用户操作时显示的一个屏幕画面：

```
如果没有需要的民族,请按 N 键,否则按其它键!
    * *民族提示* *
        1  汉族
        2  回族
        3  苗族
        4  壮族
        5  满族
        6  白族
        7  傣族
```

②为了同时显示几十个民族并使屏幕更加美观，可直接用下面的程序段代替上述程序中的 LIST 语句，这时屏幕菜单每行显示 5 个民族：

```
@3,8 SAY"* * * * * * * * * * * * * * * * 民族提示 * * * * * * * * * * * * * * * * *"
DO WHILE .NOT. EOF()
    K=1
    STRING=""
    DO WHILE .NOT. EOF() .AND. (K<=5)
        STRING=STRING+STR(RECNO(),3,0)+":"+民族+" "
        K=K+1
        SKIP
    ENDDO
    @ROW()+1,8 SAY STRING
ENDDO
```

③下面是程序自动生成的一个"动态菜单",最初含7个民族,现在含8个民族。

```
* * * * * * * * * * * * * * * * 民族提示 * * * * * * * * * * * * * * * *
 1:汉族  2:回族  3:苗族  4:壮族  5:满族
 6:白族  7:傣族  8:哈萨克
```

在dBASE Ⅲ下实现菜单的光带键选

泉州五中电脑室　阮自力

屏幕的菜单显示状况及选择方式如何,直接影响着用户的心理。本人编写的这一程序在IBM PC及其兼容机已运行通过,效果很好,这一程序具有一定的技巧性,使用起来方便灵活。为方便阅读作几点说明:①本程序既可通过→、←、↑、↓及回车键选择菜单项外,还可直接键入项编码进行选择。②程序使用的函数PEEK(1023+PEEK(1050)能指示出用户的按键内容,这里使用的72、75、77、80对应4个箭头键,28对应回车键。③程序安全性能良好,不会因为操作失误而可能影响系统的数据。④如果选择菜单项时,不慎按到指定键外的其它键,程序能响铃警告,并让你重新选择。

```
SET TALK OFF
A1="1.输　　入"
A2="2.修　　改"
A3="3.查　　询"
A4="4.统　　计"
A5="5.报表输出"
A6="6.使用说明"
A7="7.退　　出"
XZ=""
DO WHIL .T.
L=1
CLEA
@0,28 SAY "泉州五中学籍管理系统"
@1,24 SAY "……"
DO WHIL L<8
KL="A"+STR(L,1)
@L+1,28 SAY &KL
L=L+1
ENDDO
L=1
DO WHIL .T.
KK="A"+STR(L,1)
@L+1,28 GET &KK
CLEA GET
SET CONS OFF
WAIT " " TO XZ
H=PEEK(1023+PEEK(1050))
DO CASE
  CASE H=28
    XZ=STR(L,1)
  CASE H=77 .OR. H=80
    @L+1,28 SAY &KK
      L=L+1
      IF L=8
      L=1
      ENDIF
      LOOP
    CASE H=75 .OR. H=72
      @L+1,28 SAY &KK
      L=L-1
      IF L=0
      L=7
      ENDIF
      LOOP
ENDCASE
SET CONS ON
DO CASE
    CASE XZ="1"
      DO SURU. PRG
    CASE XZ="2"
      DO XIUGAI. PRG
    CASE XZ="3"
      DO CAXUN. PRG
    CASE XZ="4"
      DO TONGJI. PRG
    CASE XZ="5"
    DO BAOBIAO. PRG
    CASE XZ="6"
      DO SHIYONG. PRG
    CASE XZ="7"
      CLEA ALL
      RETURN
    OTHE
    ? CHR(7)
    LOOP
ENDCASE
EXIT
ENDDO
ENDDO
```

双列分页打印程序

中国科学院化学所　陶锋

在编写程序时，常常需要把源程序打印出来，以便检查。我在用C语言、汇编语言等的编程中，感觉到，每个语句所占用的字节并不多，但在打印时，它要占据一行，不但浪费纸张，更重要的是打印的程序所占的纸张太长不便于检查程序。能否使被打印的程序，在每页纸上打印出两列的程序清单呢？这样可以使打印出来的程序，在纸张的长度上基本达到缩短1倍的目的，便于程序清单的检查和保存。

现在打印的程序不少，但没有现成的程序可利用。本人动手用Turbo C语言的1.5版本，编写了一个适合于IBM—PC机编程打印的程序，希望能为有兴趣者所用。程序特点：

1. 此程序可在命令行上接收多个要打印的文件名，而不产生混乱。一个程序打印完后，从另一页开始打印另一个程序。命令格式为：

dcdpp filename1 filename2…↙。

2. 在要打印程序清单时，屏幕提示“Will the filename be printed with line number (y/n)?”，此时键入y，则在被打印的程序清单中加入行号。

3. 本程序可在行宽80个字符的纸上，每页打印双列程序清单。一页打印完后，等待键入任意键，打印另一页。

程序说明：本程序可在行宽80个字符的纸上输出二列，每列由54行组成，每行最多为34个字符。这些参数均为宏定义，可按具体要求进行调整。程序首先定义了一个二维数组，将待打印的程序2×54行，读入这个数组中。若源程序行超过34个字符，自动截取前34个字符，将余下的放入下一行中，如此下去。数组第一维是行号，第二维中存放这一行的内容。

函数dprint()将待打印的程序按双列形式读入二维数组中。函数nextline()每读入一个34个字符的字符串时，行号自动加1，每行内容少于34个字符时，则用空格补足。当其读入了2×54行时，调用printout()函数将其在打印机上输出。函数printout()将读入二维数组中的内容按2列的形式在打印机上输出(源程序清单就是用本程序打印的)。函数ys()将每行开头的空格或TAB键压缩掉。

```
/* DOUBLE COLUMN PAGING PRINTOUT */
#include"stdio.h"
#include"fcntl.h"
#include"conio.h"
#include"io.h"
#include"dos.h"
#include"bios.h"
#include"graphics.h"
#include"mem.h"
#include"stdlib.h"
#define LL 96
#define COL 2
#define CSIZE LL/COL-14
#define PL 108
#define DL 54
#define MARGIN 2
#define MAX 200

char buff[PL][CSIZE];
char *tt;
int ln[PL];
int col,l,p,page,key;

main(argc,argv)
int argc;
char *argv[];
{
int p,qq;
if (argc<2){
fprintf(stdout,"\nUsage: dcdpp filename[.ext]\n");
abort();
}
setmode(fileno(stdprn),O_TEXT);
for(p=1;p<argc;p++){
ys(argv[p]);
tt=argv[p];
pp=wherey();
do {
gotoxy(18,qq);
printf ("\nWill the %s be printed with line number(y/
n)?",tt);
key=getch();
}while (key! ='y' && key ! ='n');
clrscr();
dprint();
unlink("tmpll");}
}

/* Delete space or tab at beginning of lines */
ys(fname1)
char *fname1;
{
char buf[MAX];
int i,j,flag;
FILE *fp1, *fp2, *fopen();
if ((fp1=fopen(fname1,"r"))==NULL)
{
printf("\n Can't read file %s!",fname1);
```

```
abort();}
if ((fp2=fopen("tmpll","w"))==NULL)
{
printf("\n Can't write file %s!","tmpll");
abort();}
flag=1;
while(flag){
memset(buf,MAX,'\0');
if ((fgets(buf,MAX,fp1))==NULL)
flag=0;
if(flag){
j=0;
while (((buf[j])==' ')||(buf[j]==''))&&(j<
MAX))
j+=1;
for(i=j;buf[i]!="\0";i+=1)
fputc (buf[i],fp2);
}}
fclose(fp1);
fclose(fp2);
}

/* READ FILE AT D---D MODE */
dprint()
{
FILE *fp; *fopen();
int line,c;
extern int page;
if ((fp=fopen("tmpll","r"))==NULL)
{
fprintf(stdout;"\n Can't open file %s! \n","tmpll");
abort();}
line=0;p=0;l=0;
page=1;c=getc(fp);
while(c! =EOF){
ln(1)=++line;
while(c! ='\n'&&c! =EOF){
if(p>=CSIZE){
nextline();
ln[l]=++line;}
buff[l][p++]=c;
c=getc(fp);
}
nextline();
if(c! =EOF)
c=getc(fp);
}
while(l! =0){
ln[l]=0;
nextline();
}
fclose(fp);
}

/* READ THE NEXT LINE */
nextline()
{ int i;
while(p<CSIZE)
buff[l][p++]=' ';
if(++l>=PL){
printout();
for (i=0;i<(61-DL);i++)
fprintf(stdprn,"\n");
l=0;
}
p=0;
}

/* PRINT THE FILE OUT */
printout()
{ int l,i,lpos,d,p,col;
extern int page;
char line[LL];
fflush(stdprn);
gotoxy(18,5);
printf("Replace paper, then hit any key to continue!");
getch();
gotoxy(18,5);
printf("Printing now,please wait! ......");
if (key=='n'){
fprintf(stdprn,"          %10s

%5d PAGE",tt,page++);
fprintf(stdprn,"\n");
fprintf(stdprn,"----------
----------");
}
else{
fprintf(stdprn,"          %10s

        %5d PAGE",tt,page++);
fprintf(stdprn,"\n");
fprintf(stdprn,"----------
----------");
}
fprintf(stdprn,"\n");
fprintf(stdprn,"\n");
for(l=0;l<DL;l++){
for(i=0;i<LL;i++)
line[i]=' ';lpos=0;
for (col=0;col<COL;col++){
if (key=='n')
p=lpos;
else{
d=ln[l+DL*col];
p=lpos+2;
while(d>0){
line[p--]=d%10+'0';
d=d/10;}
p=lpos+4;}
```

```
for(i=0;i<CSIZE;i++)
line[p++]=buff[1+DL*col][i];
if(key ==='n')
lpos+=CSIZE+3;
else
lpos+=CSIZE+6;
}
p=LL;
while((p>=0)&&(line[p-1]==' '))
p--;
for(i=0;i<p;i++)
putc(line[i],stdprn);
putc('\n',stdprn);
}
for(l=0;l<MARGIN;l++)
fprintf(stdprn,"\n");
}
```

广州大学一号病毒解除与免疫

华南师范大学计算机系　张文锋

一、病毒特点：

1. 感染性十分强，由于它属于引导区病毒，修改了中断 13 的向量，所以只要有关于 INT 13H 的操作就首先执行病源程序从而传染磁盘，即使用无系统的数据盘启动时，只要磁盘中有病毒，也会感染硬盘。

2. 采用 stone（大麻）标志（××××：0000JMP 07CD：0005）并且采用小球病毒的标志(5713)。不但如此，它还具有自己标志（地址[01BA]开始四个字节容为：A1　0D　0A　00），所以它对 stone，小球免疫。

3. 由于标志多，往往导致一些解毒工具（如 scan 等）的误操作，执行解大麻病毒的程序，结果越解越糟。

4. 占据主引导区，转移原引导区。

a. 软盘（360K）：把原引导区调到 719 扇区（即最后），所以影响不大。

b. 硬盘：把主引导区调到 1 道 1 面 F 扇区。对于不同的硬盘只要分区的第一个柱不同，那么对盘破坏就不同，这是因为分区引导区的位置前后移到的位置不同，倒置目录区和数据区的首扇区有所变化，有时会占据目录区，有时会破坏 IBMBIO. COM 文件。

c. 对 1.2 M 软盘：与 360K 一样引导区调到 719 扇区。

二、解毒与免疫

1. 对软盘解毒

```
A >DEBUG↙
  -L 100 0 0 1↙
  -D 29D↙
  ××××××××:××××××××病毒一号
           :          :      广州大学黄永钊
                             制造!
  -L 100 0 719 1↙
  -W 100 0 0 1↙
  -Q
```

2. 对软盘免疫（对象为无病毒的盘）

```
A >DEBUG↙
  -L 100 0 0 1↙
  -E 2BA↙（连续加四字节标志）
     2BA. A1  2BB. 0D  2BC. 0A  2BD. 00↙
  -W 100 0 0 1↙
  -Q↙
```

3. 对硬盘解毒

```
A>DEBUG↙
-A 100↙
××××:0100 XOR AX,AX
           INT 13
           MOV AX,0201
           MOV BX,0200;读正确引导区
           MOV CX,010F
           MOV DX,0180
           INT 13
           MOV AX,0301
           MOV BX,0200
           MOV CX,0001;写回主引导区
           MOV DX,0080
           INT 13
           INT 3
-G=CS:100↙
-Q↙
A>SYS C:（如果不能启动时才恢复 DOS 系统程序）
```

4. 对硬盘免疫

```
A>DEBUG↙
-A 100↙
××××:100 MOV AX,0201
          MOV BX,0200
          MOV CX,0001
          MOV DX,0080
          INT 13
          INT 3
-G=CS:100↙
-E 3BA↙
3BA. A1  3BB. 0D  3BD. 0A  3BD. 00↙
-G=CS:100↙
-Q↙
```

（本人对许多软盘和硬盘按上步骤解毒都通过）

谈谈 2.13 汉字系统在软盘上的使用

韶关大学计算机系　廖红旭

2.13 系列汉字系统(以下简称 213)以其卓越的汉字处理功能赢得国内广大用户,同时,随着硬件性能价格比的提高,基本配置的 PC 机如 Boy 机和长城 0520—EM 机等,开始普及到中小学和家庭。

现在学校和家庭一般使用的汉字系统是电子工业部六所的 CCDOS4.0。它提供了驻留字库的功能,看上去很方便,实际上由于字库驻留不完全,很多字因为顾此失彼而找不到,给用户带来不便。如今基本配置的 PC 机内存都有 640K。驻留国标(GB)一、二级字库后仍从容运行巨大的 dBASE Ⅲ,而 213 就具有驻留国标一、二级字库的功能,因此把该功能移植到软盘上是很有意义的。通常 213 是安装到硬盘上使用的(213 系统盘共有 29 片,加载高点阵字库和繁体字库时硬盘容量不得低于 8M)。下面就以 2.13F 为例,给出较完整的移植步骤,望能与同行们共同讨论。

一、系统盘的操作步骤:

```
1. A>FORMAT B:/S
2. A>CDEBUG HZK16
    -RCX
    CX EF40
    :6900
    -W
    -NCCCC.COM
    -L
    -A10E
    1BCB:010E INT 20
    1BCB:0110
    --G=100
    -A100
    1BCB:0100 JMP 010E
    1BCB:0102 NOP
    1BCB:0103
    -A10E
    1BCB:010E CALL 0257
    1BCB:0111
    -A13B
    1BCB:013B JMP 013F
    1BCB:013D
    -E1820 55 AA
    -W
3. A>COPY CONFIG.SYS B:(拷贝 213 系统文件)
    A>COPY ANSI.SYS B:
    A>COPY HZK16 B:
    A>COPY FILE2.COM B:
    A>COPY CCCC.COM B:
    A>COPY CH21.COM B:(不同 CRT 选不同显示模块)
    A>COPY CON B:AUTOEXEC.BAT
    ECHO OFF
    CLS
    ECHO          PLEASE WAIT!!
    FILE2
    CCCC
    CH21
    ECHO ON
    ^Z
```

不显示版权可作下步:

```
4. A>CDEBUG CH21.COM
    -A123
    1BCB:0133 JMP 019C
    1BCB:0135
    -W
```

经过上面几步,用户就能使用 213 的驻留汉字库功能了。若想使用 213 别的小巧功能如预选字表、键盘重定义等,就必须把 213 的 YX1.COM 文件、KEY.COM 文件等拷贝到用户的工作软盘内运行之。

二、两个重要的向量入口。

向量名	入口地址	注　意
INT 16	CS:9A80	原 INT 16 拷贝为 INT 7E
INT 10	CS:184A	原 INT 10 拷贝为 INT 78

了解这两个向量的入口,分析和修改它们都是方便的。

三、进一步修改(兼谈键盘扫描码)

使用过长城机的同行都知道,它有方式 1～4,全/半角转换和英文这 6 个键。而标准的 101 键盘没有。在汉字录入的响应和方便程度方面,前者确比后者方便。能否利用标准的 101 键盘仿真长城机键盘呢?这就要先了解键盘 I/O 的基本原理。

我们知道,键盘的每一个非变换键被按下,硬件电路就会调用监控程序 INT 16(键盘 I/O 中断)产生相应的 ASCII 码和扩展 ASCII 码(又称扫描码)。键盘上除 SHIFT、CTRL、ALT、SCROLL、CAPSLOCK 等 5 个键外,每一个键都有一个对应的扫描码,但不一定有对应的 ASCII 码。组合键的 ASCII 码等于零值。通过判断 ASCII 码或扫描码可以知道哪一个键(或组合键)被按下过。这在 PCTOOLS 等工具软件及 CCBIOS 中的键盘管理模块中广泛使用。(附录中给了常用的组合键的扫描码)。

知道上述这些原理,修改 2.13F 的键盘管理模块 CCCC.COM,使标准 101 键盘仿真长城机键盘是容易的。当然,被重定义的功能键原功能将丢失。本文旨在提出思想方法。步骤如下:

```
    A>CDEBUG B:CCCC.COM
    -E9BC1 1F
    -E9BD9 52
    -E9BF4 47
```

-E9C00 49
-E9C0C 53
-E9C18 51

这样，101 标准键盘仿真如下：

INSERT(区位)	HOME(首尾)	PGUP(拼音)
DELETE(快速)	END(全/半角)	PGDN(英文)

附录：常用组合键扫描码表

扫描码	字符组合
54H～5DH	SHIFT+F1～F10
5DH～67H	CTRL+F1～F10
68H～71H	ALT+F1～F10
72H	CTRL+PRTSC
75H	CTRL+END
76H	CTRL+PGDN

PC/XT 机病毒报警程序

福州大学　王伟智　刘秉瀚

许多病毒常采用修改内存总量，使病毒程序常驻内存，这样，用户软件就不能复盖病毒程序。

PC 机系统内存总量参数存放在 0:0413H 单元中，并以 K 为单位，例如，640K 内存的配置，0:0413H 单元内容为 280H。对一般 XT 系统，内存配置为 640K、512K 和 256K。如果系统启动后，内存总量不等于这几种参数，系统就有可能传染上病毒。用户如能知道这个情况，就可及时采取措施，避免许多修改内存总量的病毒的传播。

根据这个思想，我们设计了一个病毒报警程序，把本程序的调用命令存放在 AUTOEXEC. BAT 文件中，系统引导结束，自动执行病毒报警程序。如发现内存总量减少，立即报警，提醒用户注意病毒将传染；如无减少，则转去执行正常操作。本程序可发现“小球”病毒、“大麻”病毒、“巴基斯坦”病毒、“磁盘杀手”和“毛虫”病毒等。实验表明，本程序对于操作系统型病毒和传染了 COMMAND. COM 的外壳病毒，并且利用修改内存总量使病毒程序隐蔽起来的情况效果很好。

本程序仅能发现病毒在活动，但不能确定病毒类型，也不能除去病毒。对于修改 INT 21H 的病毒，程序中显示报警提示的中断应改为 INT 10H。病毒对 INT 10H 服务程序的修改较少出现，这样，可避免报警时病毒进行传播的可能性。修改方法可参见 DOS 手册。

```
CODE    SEGMENT
        ASSUME CS:CODE,DS:CODE,ES:CODE
STACK   SEGMENT PARA STACK 'STACK'
        DB 20 DUP(0)
STACK   ENDS
        ORG 100H
BEGIN:  JMP START
WARN    DB 'WARNING: TOTAL MEMORY REDUCED!',0AH,0DH
        DB 'BEWARE OF VIRUS INFECTION!!!','$'
START:  PUSH DS
        XOR AX,AX
        MOV DS,AX
        MOV BX,0413H        ;取 413H 单元内容
        MOV AX,[BX]         ;到 AX
        POP DS
        CMP AX,0280H        ;内存是 640K 吗?
        JZ RET              ;正常则退出
        CMP AX,0200H        ;内存是 512K 吗?
        JZ RET              ;正常则退出
        CMP AX,0180H        ;内存是 384K 吗?
        JZ RET              ;正常则退出
        CMP AX,0100H        ;内存是 256K 吗?
        JZ RET              ;正常则退出
        LEA DX,WARN         ;不正常则报警
        MOV AH,09H
        INT 21H
RET:    INT 20H
CODE    ENDS
        END BEGIN
```

考生姓名编码自动生成程序

甘肃庆阳长庆一中　任绥海

随着标准化考试的日益深化和完善，考生的姓名等汉字信息也可以由机器来阅读处理了。如《考生机读报名卡》就要求考生将汉字信息以编码的形式涂写出来。要涂写编码，首先当然应该知道编码。国家招生委员会连续两年编印出版了《汉字编码简明对照表》，供考生自己查找。遗憾的是它们都只收录了两千五百多个常见姓名用字，有相当一部分学生根本就找不到自己的姓名编码，不得不向上一级主管部门伸手求援；即使是“榜上有名”者，一个学校，几百考生，要想从那几张对照表上一个一个查出自己的姓名编码，也绝非易事。

实际上，凡了解计算机汉字操作系统的人都能看出，所谓姓名编码原不过是汉字区位码，是根据国家有关标准编制的一种没有重码现象的汉字输入码。它是各种编码中最容易掌握的、与机内码联系最紧密的一种编码，而且可以用很简单的方法获得。下面这个 BASIC 程序，就可以利用各考点现有考生姓名数据库，一次生成姓名编码对照表，并以文件的形式存放在磁盘上，供编辑软件随意编排成各种格式，多次打印，既准确无误，又省时省力。

使用本程序前，只需将数据库文件中的姓名字段

转换成标准数据格式(SDF)即可。

以下是程序及打印实例,供您参考。

```
10 KEY OFF:CLS:LOCATE 6
20 SOUND 440,8:INPUT"考生姓名文件名?",WJM$
30 OPEN WJM$ FOR INPUT AS #1
40 WJM1$=WJM$+".DAT"
50 OPEN WJM1$ FOR OUTPUT AS #2
60 PRINT STRING$(30,"-")
70 PRINT #2,STRING$(30,"-")
80 IF EOF(1) GOTO 280
90 INPUT #1,A$:A$=RIGHT$(A$,6):PRINT A$;
100 PRINT #2,A$;
110 FOR Q=1 TO 6
120 A$(Q)=MID$(A$,Q,1)
130 A(Q)=ASC(A$(Q))-160
140 A1$(Q)=STR$(A(Q)):IF A(Q)<0 THEN A1$(Q)=" "
150 A1$(Q)=RIGHT$(A1$(Q),2)
160 IF LEFT$(A1$(Q),1)=" "AND RIGHT$(A1$(Q),1)<>" "THEN A1$(Q)="0"+RIGHT$(A1$(Q),1)
170 NEXT
180 PRINT STRING$(4,32);
190 PRINT #2,STRING$(4,32);
200 FOR Q=1 TO 6
210 PRINT A1$(Q);
220 PRINT #2,A1$(Q);
230 IF Q=2 OR Q=4 THEN PRINT" ";:PRINT #2," ";
240 NEXT
250 PRINT:PRINT STRING$(26,"-")
260 PRINT #2,"";PRINT #2,STRING$(26,,"-")
270 GOTO 80
280 CLOSE:CLS:SOUND 440,8:LOCATE 10,18:COLOR 7,2
290 PRINT"姓名编码文件已经以";WJM1$;"的名字存于盘上
300 LOCATE 12,37:PRINT"再见!":COLOR 7,0 END
```

李　蕊	3278		4079	唐莉梅	4438	3282	3523
王建伟	4585	2908	4616	金志宏	2980	5430	2674
王志梅	4585	5430	3523	尹　虎	5092		2702
符永宏	2391	5132	2674	邓　燕	2143		4964

(上接第 25 页)

```
    scanf("%s",new->name);
    scanf("%s",new->city);
    scanf("%s",new->zip);
    new->next=last;
    last=new;
    } while((i++)<3);
    do {printf("%s\n",last->name);
       last=last->next;
       } while(last);
    }
```

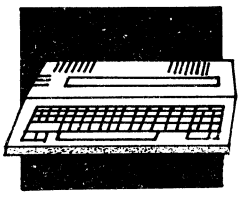
```
A>exp9_9
王宜滨      天津      300071
李文华      湖北      500065
卢元来      北京      600073
林俞锡      东北      154600
林俞锡
卢元来
李文华
王宜滨
```

可以改变连接方向的程序,如练习 9.10 所示。

```
A>type exp9_10.c
#include <stdio.h>
#include"stdlib.h"
#include"alloc.h"
struct address {char name[9];
                char city[7];
                char zip[7];
                struct address *next;
                };
main()
  {struct address *new, *last=0; *top=0;
   int i=0;
   do {if((new=malloc(sizeof(struct address)))==0)
       {printf("out of memory\n");
       exit(0);
       }
   if(top==NULL)
          new->next=NULL;
   scanf("%s",new->name);
   scanf("%s",new->city);
   scanf("%s",new->zip);
   last->next=new;
   last=new;
   } while((i++)<3);
   do {printf("%s\n",top->name);
      top=top->next;
      } while(top);
  }
```

```
A>exp9_10
李一民      天津市    3000074
鲁晓春      北京市    1000009
王里黎      河北省    5000002
林太态      山东省    4000003
(null)
李一民
鲁晓春
王里黎
林太态
```

(南开大学王玉华同志参加了程序调试)



学习机之友

把 APPLE DOS 移入 RAM 卡

舟山市电力公司变电工区　傅　剑

一般的 APPLE II PLUS 或中华学习机都配置有 16K RAM 卡(即 $D000～$FFFF RAM 辅存),在 DOS 3.3 中把该卡作为存放 INTBASIC 解释语言的地址,因此也把该卡称作为语言卡。从大量的应用程序中可以看出,我们很少使用 INTBASIC 编制程序,因此语言卡经常被闲置,笔者为充分发挥现有设备的效益,特编制一程序,把 DOS 移入语言卡中。从而使 CEC BASIC 的最大编程容量高达 42KB 之多,由于妥善地处理了 DOS 与监控、CEC BASIC 解释程序的接口问题,因此当 DOS 移入语言卡后,所有的 DOS 命令和功能与未移入之前完全一样(除不能再使用语言卡)。该程序用 6502 汇编语言编写,因为需处理多种接口问题,所以内容比较长(约占 7 个扇区),在此以机器码的方式列出,有兴趣的读者可用反汇编命令将其列出分析。

程序的使用方法很简单,只要输入下列程序并存盘后,用 BRUN"程序名"命令即可。

```
4000-08 78 A0 04 B9 00 00 48
4008-88 10 F9 A9 00 8D 65 43
4010-AD D2 03 C9 9D F0 19 EE
4018-65 43 C9 AF B0 09 A9 8C
4020-85 00 A9 43 4C 4A 43 A9
4028-FC 85 00 A9 43 4C 4A 43
4030-AD 83 C0 AD 83 C0 AD 00
4038-E0 49 A5 8D 00 E0 CD 00
4040-E0 8D 81 C0 8D 81 C0 F0
4048-0C EE 65 43 A9 C8 85 00
4050-A9 43 4C 4A 43 A9 4C 8D
4058-68 A5 A9 04 8D B3 AE A9
4060-FB 8D CD AE A9 AF 8D CE
4068-AE A9 0B 8D F9 AE 20 D1
4070-9E A9 00 8D B3 AA A9 88
4078-85 00 A9 41 85 01 A2 00
4080-20 A1 40 E6 02 D0 02 E6
4088-03 18 A1 02 69 40 81 02
4090-E6 02 D0 02 E6 03 88 88
4098-D0 E9 20 A1 40 D0 E4 F0
40A0-1C A1 00 85 02 20 B6 40
40A8-A1 00 85 03 20 B6 40 A1
40B0-00 20 B6 40 A8 60 E6 00
40B8-D0 02 E6 01 60 A2 00 86
40C0-04 BD 9A 41 85 02 BD 9B
40C8-41 85 03 A2 00 A1 02 20
40D0-8E F8 A4 2F C0 02 D0 60
40D8-B1 02 C9 9D 90 5A C9 C0
40E0-90 22 C9 D0 90 52 A2 0B
40E8-B1 02 DD B6 41 D0 0E 88
40F0-B1 02 DD C1 41 D0 06 BD
40F8-CC 41 4C 28 41 A0 02 CA
4100-10 E6 30 34 88 C9 B7 D0
4108-0F B1 02 C9 E8 90 20 C9
4110-F9 B0 1C 38 E9 13 D0 10
4118-C9 B5 D0 13 B1 02 C9 A4
4120-90 0D C9 D1 B0 09 69 2B
4128-91 02 C8 A9 BF D0 07 A0
4130-02 B1 02 18 69 40 91 02
4138-38 A5 2F 65 02 85 02 A9
4140-00 65 03 85 03 A6 04 DD
4148-9D 41 90 07 A5 02 DD 9C
4150-41 B0 03 4C CB 40 8A 18
4158-69 04 AA E0 1C B0 03 4C
4160-BF 40 AD 83 C0 AD 83 C0
4168-A2 23 A9 BF 85 03 A9 FF
4170-85 01 A0 00 84 02 84 00
4178-B1 02 91 00 C8 D0 F9 CA
4180-F0 55 C6 03 C6 01 D0 F0
4188-00 9D 56 64 9D 02 6C 9D
4190-04 C3 AA 3A EE B7 02 00
4198-00 00 78 9D 84 A8 FD AA
41A0-97 B3 00 B6 E0 B7 00 B8
41A8-11 BA 69 BA 96 BA 56 BC
41B0-A8 BF C8 BF 00 C0 D6 D7
41B8-E5 ED F9 FD FD FD FD FE
41C0-FE 65 D2 1B 24 48 0C 8E
41C8-DA ED 8B 95 27 33 39 3F
41D0-45 4B 51 57 5D 63 69 A2
41D8-AF BD BF 43 9D 00 BE E8
41E0-D0 F7 BD BF 44 9D 00 BF
41E8-E8 D0 F7 A2 11 BD E8 F7
41F0-9D D5 BF BD BB F5 9D E6
41F8-BF CA 10 F1 A9 D5 8D C1
4200-EA 8D E4 F7 A9 BF 8D C2
4208-EA 8D E5 F7 A9 E6 8D 0E
4210-DD A9 BF 8D 0F DD A9 6F
4218-8D 02 DD A9 BF 8D 03 DD
4220-A9 78 8D 04 DD A9 BF 8D
4228-05 DD A9 87 8D BB DE A9
4230-BF 8D BC DE A9 81 8D C6
4238-DF A9 BF 8D C7 DF A9 4C
4240-8D BA DE 8D C5 DF A9 C4
4248-8D A4 DD 8D B7 DD A9 BF
4250-8D A5 DD 8D B8 DD A2 0B
4258-BD 20 44 9D 56 DD CA 10
4260-F7 A2 12 BD ED FF 9D B2
4268-E5 CA 10 F7 A9 B2 8D 78
4270-F3 A9 E5 8D 79 F3 A9 94
4278-85 00 A9 DD 85 01 20 C7
4280-42 A9 CD 85 00 A9 DD 85
4288-01 20 C7 42 A9 7C 85 00
4290-A9 E5 85 01 20 C7 42 A9
4298-A0 85 00 A9 E5 85 01 20
42A0-C7 42 A9 E6 85 00 A9 F6
42A8-85 01 20 BB 42 A9 7D 85
42B0-00 A9 F7 85 01 20 BB 42
42B8-4C D8 42 A0 02 B1 00 D9
42C0-D2 42 D0 0D 88 10 F6 A0
42C8-02 B9 D5 42 91 00 88 10
42D0-F8 60 AD 00 E0 20 E2 BE
42D8-A2 1E BD 32 44 9D 51 DE
42E0-CA 10 F7 A2 05 BD 2C 44
42E8-9D FA FF CA 10 F7 A2 0C
42F0-BD 51 44 9D BB F5 CA 10
42F8-F7 A9 4C 8D C6 E0 A9 BB
4300-8D C7 E0 A9 F5 8D C8 E0
4308-A2 0F BD 5E 44 9D E6 F7
4310-CA 10 F7 A9 4C 8D 36 E8
4318-8D 94 E3 A9 E6 8D 37 E8
4320-A9 F7 8D 38 E8 A9 EE 8D
4328-95 E3 A9 BE 8D 96 E3 A9
4330-00 8D 5F EA A9 06 8D 42
4338-DE A0 1D B9 6E 43 99 75
4340-EA 88 10 F7 A9 FC 85 00
4348-A9 43 85 01 AD 82 C0 A0
4350-00 B1 00 F0 06 20 F0 FD
4358-C8 D0 F6 A0 04 68 99 00
4360-00 88 10 F9 A9 00 F0 02
4368-28 60 28 4C 06 BF C8 C5
4370-CC CC CF A0 A0 A0 A0 A0
4378-A0 A0 A0 A0 A0 A0 A0 A0
4380-A0 A0 A0 A0 A0 A0 A0 A0
4388-A0 A0 A0 A0 8D 8D C3 C1
4390-CE A7 D4 A0 C5 D8 C5 C3
4398-D5 D4 C5 A0 A2 C4 CF D3
43A0-AD D5 D0 A2 AE 8D 87 C4
43A8-CF D3 A0 CE CF D4 A0 C1
43B0-D4 A0 CE CF D2 CD C1 CC
43B8-A0 B4 B8 CB A0 CC CF C3
43C0-C1 D4 C9 CF CE AE 8D 00
43C8-8D 8D C3 C1 CE A7 D4 A0
43D0-C5 D8 C5 C3 D5 D4 C5 A0
```

43D8-A2 C4 CF D3 AD D5 D0 AE
43E0-A2 8D 87 CE CF A0 CC C1
43E8-CE C7 D5 C1 C7 C5 A0 C3
43F0-C1 D2 C4 A0 C6 CF D5 CE
43F8-C4 AE 8D 00 8D 8D A0 A0
4400-A0 A0 A0 A0 A0 A0 A0 C4
4408-CF D3 A0 CE CF D7 A0 C9
4410-CE A0 C8 C9 C7 C8 A0 CD
4418-C5 CD CF D2 D9 AE 8D 00
4420-9F BF AA BF AD BF B3 BF
4428-B9 BF BF BF 8D BF 93 BF
4430-99 BF 4C 00 BF 4C 06 BF
4438-4C 0C BF 4C 15 BF AD D4
4440-BF AC D3 BF 60 AD D2 BF
4448-AC D1 BF 60 4C 1E BF C5
4450-BF 8A F0 07 C9 06 B0 03
4458-4C D1 E0 4C C9 E0 68 68
4460-AD FE BF 48 AD FF BF 48
4468-AD B6 EA 4C 39 E8 08 2C
4470-83 C0 2C 83 C0 28 60 8D
4478-FA BF 08 68 8D FB BF 68
4480-8D FC BF 68 8D FD BF A9
4488-BE 48 A9 AE 48 AD FD BF
4490-48 AD FC BF 48 AD FB BF
4498-48 FD FA BF 28 8D 82 C0
44A0-60 20 B8 BE CD 00 E0 08
44A8-AD 00 E0 28 60 AD 72 EA
44B0-8D FE BE AD 73 EA 8D FF
44B8-BE 20 B8 BE 4C 00 00 20
44C0-AF BE 4C BF DD 20 AF BE
44C8-4C 84 DD 20 AF BE 20 FD
44D0-EA 4C DE BE 20 AF BE 20
44D8-B5 F7 4C DE BE 20 AF BE
44E0-20 51 E8 4C DE BE 8D 82
44E8-C0 20 65 D6 20 AF BE 4C
44F0-FF E4 8D 82 C0 4C D2 D7
44F8-20 B8 BE 4C 1B E5 20 B8
4500-BE 4C 24 ED 20 B8 BE 4C
4508-48 F9 20 B8 BE 4C 0C FD
4510-20 B8 BE 4C 8E FD 20 B8
4518-BE 4C DA FD 20 B8 BE 4C
4520-ED FD 20 B8 BE 4C 8B FE
4528-20 B8 BE 4C 95 FE 20 AF
4530-BE 20 81 DE 4C DE BE 20
4538-AF BE 20 BD DE 4C DE BE
4540-20 B8 BE 6C 36 00 20 B8
4548-BE 6C 38 00 20 B8 BE 6C
4550-FA FF 8D 82 C0 6C FC FF
4558-20 B8 BE 6C FE FF AD B6
4560-EA D0 03 8D 82 C0 6C C5
4568-BF 6C C7 BF 8D 82 C0 6C
4570-C9 BF 8D 82 C0 6C CB BF
4578-8D 82 C0 6C CD BF 8D 82
4580-C0 6C CF BF 00 00 00 00
4588-00 00 00 00 00 00 00 00
4590-D5 BF E6 BF 01 00 00 00
4598-00 00 00 00 00 00 00 00
45A0-00 00 00 00 00 00 00 00
45A8-00 00 00 00 00 00 00 00
45B0-00 00 00 00 00 00 00 00
45B8-00 00 00 00 00 AF BE FF

将程序 CHAIN 升级为 APPLE DOS 命令

保定地区商业学校　赵江龙　孙杰亮

CHAIN 用来实现 BASIC 程序的链接与数据的传递，弥补了 APPLESOFT 的一大不足。但当初 APPLE 公司在发表它时，并未将其开发成 DOS 命令，而只作为一个 B 类文件保留在系统主盘上，供用户在需要时调用。目前，对 CHAIN 的研究性文章已为数不少，许多作者均从各自的角度提出了不同的改进方案，但他们大都着眼于对 CHAIN 功能的拓展上。这种仅将该功能程序当作一般的应用程序来调用的处理方式，影响了 CHAIN，包括改进后的 CHAIN 的实用价值。

笔者在对系统进行了认真分析之后，成功地将该程序升级为新的 DOS 命令。此后，APPLESOFT BASIC 程序的链接将和其它的 DOS 功能一样，用标准的命令格式，以预定命令激活的方式来实现。这对于在应用程序开发过程中需要频繁传递数据的用户，具有实际价值；对于一般的用户，由于采取了一定的技术处理，用户可用的空间并不见减少，而且还保证了最大限度的兼容。为了同用于整数 BASIC 的 CHAIN 命令相区别，该命令取名为“SCHAIN”。

为了对 DOS 进行有效的扩充而又不影响 DOS 原有的全部功能，我们模仿 DOS 对命令的处理方式，在 DOS 和 BASIC 之间增加一个新的拦截判断层次，专门截获处理 SCHAIN。由于本问题把该命令约定为 DOS 命令，SCHAIN 一定属于 DOS 的拦截判断之列，因而仅对 DOS 传给 BASIC 的非 DOS 命令进行拦截即可。

考虑到 DOS 3.3 的特殊性，在程序方式下，SCHAIN 将不为 DOS 所接受。原因是在该执行方式下，系统规定以“CTRL－D”字符打头的输出内容只应是 DOS 命令，不属于 DOS 的则认为是错误命令。非 DOS 命令时就直接由 DOS 提示出：? SYNTAX ERROR IN XX”的错误提示信息。因此，加入的拦截处理程序将接收不到该命令。本文对此进行了调整，将 DOS 的这种错误判定功能全部交给了 BASIC，该调整见步骤 6。这样，对于任何一个命令，属于 DOS 就转去执行 DOS 命令，是 SCHAIN 则进行程序链接，是错误命令的话，将由 BASIC 解释程序最终判定为错。

在运行空间的选择上，SCHAIN 放在 $BA69～$BC55 的 DOS 区域。原 CHAIN 占用的是 $208～$3B5，由于第二页同作键缓区，CHAIN 极易受到干扰，以致在链接过程中即遭到破坏；而 DOS 的两个资料缓冲区却被固定在 DOS 内部，当作系统的一部享受保护。系统的这种安排是不合理的。我们参考了姚政曙“从 DOS 中获取用户空间”一文(《苹果园》87 年第 3 期)提出的方法，将资料缓冲区连同读翻译表一起，与 CHAIN 原占有的空间进行了对调。具体的操作方法在本文后面的部分列出。

升级步骤如下：

1. 引导标准 DOS3.3，CALL－151 进入监控；

2. 在监控状态下键入：

B811:2　NB880:2　NB890:2　NB8CB:2　NB921:2　NBEC1:2　NBEC7:2　NB80A:3　NB80E:3　NB820:3　N

B825:3 NB83A:3 NB868:3 N B8CE:3 NB8D2:3
NB910:3 NB86B:2 NB90A:C0 2 NB91B:C0 2
NB92B:C0 2 NB64A:4C B3 08

3. 键入：

B6B3:A2 0 BD 6C 3 9D 56 3 E8 E0 6A
D0 F5 A6 2B 6C FD 8 0

4. 执行

356<BA96.BAFFM〈CR〉

键入程序 1；

5、键入：

A008:4C 69 BA〈CR〉

将 DOS 的转移指针指向 SCHAIN；

6、去掉 DOS 的错误识别功能，键入：

A000:EA EA EA EA EA EA EA EA〈CR〉

7、插入新软盘，执行 INIT 命令。

程序一：

```
BA69- A0 FF A2 00 AD 00 02
BA70- C9 84 D0 01 E8 C8 BD 00
BA78- 02 E8 C9 A0 F0 F8 D9 8B
BA80- BA F0 F2 B9 8B BA F0 0A
BA88- 4C A4 9F D3 C3 C8 C1 C9
BA90- CE 00 20 12 BC 20 31 BC
BA98- 20 84 E4 A9 07 85 8F A5
BAA0- 69 A6 6A 85 9D 86 9E E4
BAA8- 6C D0 04 C5 6B F0 05 20
BAB0- 0B BB F0 F3 85 9F 86 A0
BAB8- A9 03 85 8F A5 9F A6 A0
BAC0- E4 6E D0 07 C5 6D D0 03
BAC8- 4C 6D BB 85 9D 86 9E A0
BAD0- 00 B1 9D AA C8 B1 9D 08
BAD8- C8 B1 9D 65 9F 85 9F C8
BAE0- B1 9D 65 A0 85 A0 28 10
BAE8- D3 8A 30 D0 C8 B1 9D A0
BAF0- 00 0A 69 05 65 9D 85 9D
BAF8- 90 02 E6 9E A6 9E E4 A0
BB00- D0 04 C5 9F F0 BA 20 15
BB08- BB F0 F3 B1 9D 30 46 C8
BB10- B1 9D 10 41 C8 B1 9D F0
BB18- 3C C8 B1 9D AA C8 B1 9D
BB20- 85 9C 86 9B C5 B0 F0 02
BB28- B0 2B 88 88 B1 9D 48 38
BB30- A5 6F 85 94 F1 9D C8 91
BB38- 9D 85 6F C8 A5 70 85 95
BB40- E9 00 91 9D 85 70 68 18
BB48- 65 9B 85 96 A5 9C 69 00
BB50- 85 97 20 9A D3 A5 8F 18
BB58- 65 9D 85 9D 90 02 E6 9E
BB60- A5 9D A6 9E A0 00 60 C4
BB68- C1 CF CC 84 8D A9 00 85
BB70- 95 85 9C 85 97 A9 69 85
BB78- 9B A9 71 85 96 A9 EF 85
BB80- 94 20 9A D3 A2 01 B5 69
BB88- 95 9B B5 6D 95 96 B5 6F
BB90- 95 94 CA F0 F1 20 9A D3
BB98- A5 94 85 A1 A6 95 E8 86
BBA0- A2 A0 06 B9 66 BB 20 5C
BBA8- DB 88 D0 F7 C8 B1 B8 F0
BBB0- 0A C9 8D F0 06 20 5C DB
BBB8- C8 D0 F2 A9 C9 85 36 A9
BBC0- BB 85 37 20 51 A8 4C FB
BBC8- DA 20 93 FE 20 51 A8 20
BBD0- 65 D6 38 A5 69 E5 E7 85
BBD8- 9D A5 6A E5 E8 85 9E A2
BBE0- E9 20 03 BC E8 20 03 BC
BBE8- A2 01 B5 ED 95 6F 95 3E
BBF0- B5 A1 95 3C B5 69 95 42
BBF8- CA F0 EF A0 00 20 2C FE
BC00- 4C D2 D7 18 B5 00 65 9D
BC08- 95 82 E8 B5 00 65 9E 95
BC10- 82 60 CA CA 86 B8 A9 02
BC18- 85 B9 AD 59 AA 8D 38 BC
BC20- BA 8E 59 AA A9 95 48 A9
BC28- BA 48 20 5B A7 20 83 9F
BC30- 60 AD 38 BC 8D 59 AA 60
```

（上接第 22 页）

BASIC（BAS）、二进制（BIN）、文本（TXT）和变量（VAR）等文件分别经 SAVE、BSAVE、OPEN 或 STORE 命令自动地产生。行号 110 至行号 190 建立一个菜单，将文件类型显示给用户。行号 200 至行号 290 用户从 1～8 之间选择文件类型。行号 300 和行号 305 是要求用户输入路径名或文件名。行号 320 用 CREATE 命令产生文件名和文件类型。CREATE 命令必须用“,T 类型”参数产生所需要的文件类型。只有目录文件的产生不需要 T 参数。

如果用户选择了用户定义文件类型，那么程序由行号 310 转至行 350，允许用户选择所需的文件号（从 1 至 8）。行号 380 将两个串变量相加，而后返回到行号 320 产生文件及文件类型。

ProDOS 系统程序（SYS）、浮动代码（REL）和用户定义（$F#）类型文件可以用 BASIC 产生。

〈未完待续〉

北京安华科技市场西城经营部
——主要经营：电子计算机、电子元器件

各种微机：

• 中华学习机： CEC—I ￥900.00 小蜜蜂—Ⅰ ￥920.00

• 家庭 PC 机：￥3150.00

主频：4.77/12 兆 RAM：640KB 软驱：360KB×2
101 键盘
14″双频单显
Smper286：￥6950.00
主频：10/21 兆 RAM：1MB 可扩 5MB 5 1/4 软驱：1.2MB 360KB 各 1
硬盘 3.5″40MB. 101 键盘 14″双频单显 数显机箱
选配：14″CGA 彩显加 1500 元 14″1024×768 VGA 彩显加 3800 元

各种：APPLE Ⅱ、CEC—Ⅰ、PC 机接口板；九针、24 针打印机

还承接各种微机的修理业务。

对教育部门、个人购机实行优惠。

地址：北京西四北大街 120 号 电话：65.6342
邮码：100034
联系人：孙义强 宁函华 BB 机：126 呼 45198 42616

向天坛机移植中华机汉字软件的一些体会与经验

张　斌

天坛机及其加强型问市以来，便以与中华机汉字90%以上的兼容度而受到好评。但是，也有一部分中华机的汉字软件在天坛机上不能直接运行。我曾经试着移植了一批汉字软件，包括'人事档案'、'个人办公系统'、'家庭账目管理'、'备忘录'等，现在，我就把在移植过程中出现的问题和几点经验做一简单的说明。其中，出现的问题大致可分为以下几点。

(1)**汉字状态下图象的移植**。

这是中华机汉字软件移植中的一个关键性问题。中华机的汉字屏幕是显示在高分辨率第二页上的，即内存地址4000H～5FFFH。而天坛机的汉字屏幕占用的是高分第四页，即内存地址8000H～9FFFH。故中华机上图文并茂的软件到天坛上便只见文不见图了。如果图形是以34扇区B类文件的形式存在盘上的，那么在读文件时只需把起始地址从4000H改到8000H，图形就会显示出来了。但有一些较高级软件中的图形是以压缩形式存放的(如通讯录)，在它运行时先调入一个解压缩文件和压缩的图形文件，然后再把图形还原到高分第二页上去。对此解决的方法有两种，一是先把还原好的图形存盘，然后修改主程序，直接把图形调入高分第四页。这种方法的优点是简单，省事，缺点是占用磁盘空间较大，调盘时速度较慢。另一种方法是修改解压缩文件，使它把压缩的图形直接还原到高分第四页。这就要求移植者对机器的内部结构、6502语言以及一些常用的系统子程序有个大概的了解才行。这种方法的优点是使用时速度绞快，对主程序的改动不大，缺点是移植的周期较长。

还有一些程序，如'个人办公系统'是先把图形调入高分辨率第一页做些处理，再通过调用机器码子程序把图形移入高分第二页。对于此类程序的修改主要是针对机器码子程序的修改，即把图形移动的目的地址从高分第二页改到高分第四页就成了。

(2)**汉字状态下特殊字符的移植**

在中华机的硬汉字库中包含了一些特殊的图形字符，如'①''②''Σ'等等。而在天坛机中所对应的不是空字符就是"怪"字符。比如中华机中的'①'对应于天坛机的就是大写的'X'；':'对应是大写的'B'等等。这就是为什么有些中华机的汉字软件在天坛机上可以运行，但在中文中间总是加杂有"怪"字符的原因。对于这类问题解决方法一般有两种：一是硬替换，一种是软替换。硬替换法就是直接用天坛机中的字符替换掉。比如用':口'两个字符替换掉'B'；用两个减号替换掉'!'等等。这种方法的特点是快速、简便，适用于画面要求不高，特殊字符不太多的场合。缺点是显示效果不如软替换法。而软替换法就是把所有的中华机图形字符制成造型表，然后用造型表替换掉那些"怪"字符。我在"小学数学"软件移植过程中，考虑到原来的显示比较生动活泼，适合于小学生使用，如使用硬替换法就会影响画面的质量，因而采用了软替换方法。这种方法的优点是显示灵活，方便，画面质量较高，效果好。自己甚至可以造出中华机没有的字符。缺点是大量显示造型表时会影响速度，移植难度大，周期长。

(3)**汉字状态下与DOS的配合问题**

在中华机上，汉字系统并不侵占DOS区，因而一些经过改进的快速DOS可以在汉字系统下正常运行。而在天坛机中的汉字系统侵占了9600H～9FFFH这一部分的DOS区作为汉字显示页，这样使得一些快速DOS无法正常运行。经我实验在汉字状态下真正能正常运行的DOS只有标准的DOS3.3。所以，如果你的汉字软件不能正确的打开和读写文件时，请换一个标准DOS3.3试试看。

(4)**汉字状态下英文字符大小写问题**

在我移植的"小学英语"软件中，最突出的问题就是在汉字状态下英文大小写字符的问题。在英文中，一般在句子开头的第一个字母必须大写，在人名、地名、缩写和其它一些专有名词的第一个字母也必须大写。但在天坛机的汉字状态下，英文字符没有小写状态。如果把试题和答案全部改成大写字符，不仅工作量成倍增加，而且会使一部分考题失去意义。为了解决这一矛盾，我采用了特殊字符移植问题中的软替换法。即把二十六个小写字符全部都做成造型表。然后在应显示小写字符的位置上用造型表替换。

关于移植过程中的问题就介绍这些，其中难免有失误之处，欢迎大家批评指正。下面，我将在移植中积累的一些小经验介绍一下。

(1)去掉原程序中所有CALL 43089语句。

(2)在PR#3语句后面必须紧跟一条PRINT语句。

(3)在MUSIC语句后必须跟一个PR#3及PRINT语句。

(4)有些程序中DOS命令不能正确执行，就在这条命令前加一条空DOS命令，即PRINT CHR$(4)

(5)在汉字状态下GET语句不能读进控制字符，用下面子程序模拟：

```
9998 POKE $C010,0:A=PEEK($C000):A$=CHR$(A):IF A<128 THEN 9998
9999 RETURN
```

(6)汉字状态下有时要知道光标所处位置，下面语句把横座标放入H，纵座标放入V：

```
H=PEEK(214):V=PEEK(215)
```

FORTH 问答：堆栈和逆波兰表示法

北京机工印刷厂动力设备科　丁志伟

甲：你上回曾说 FORTH 是面向堆栈的语言，就是汇编语言常用的后进先出栈吗？

乙：正是。不过对于 FORTH，堆栈的作用相当重要，使用起来内容也更丰富。

甲：逆波兰是怎么回事？和堆栈是什么关系？

乙：逆波兰是一种运算表示法，在算术逻辑运算时与堆栈互为表里。解释起来有些麻烦，这样吧，我问你，用 BASIC 做 3+5，再显示出来，你怎么做呢？

甲：可以用 A=3+5:PRINT A。FORTH 是怎样的呢？

乙：用 3　5+.↙，显示 8。（↙表示回车）

甲：看起来有些新鲜，怎么不是 3+5？变量是怎么表示的？数又存到哪里？还有……

乙：你看，逆波兰是有不少特点，对这么一个简单的算式，能让你提出这么多问题。我一个个回答。这种表示法，一般不用变量，而把数据存在堆栈里。

甲：在这个例子里，怎么用到堆栈的？

乙：FORTH 每输入一个数都要放到栈顶，取出时也从栈顶取。这里先输入的是 3，存入栈顶，见图 1a。再输入 5，栈顶就变成 5，3 并不消失，而在次栈顶，见图 1b。

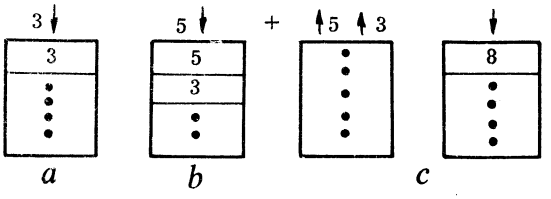


图　1

甲：遇到“+”呢？也入栈吗？

乙：不，栈中只能存放数据，“+”则是从栈里取出两个数，相加后把两数之和再送入堆栈。此时栈顶是 8，见图 1c。3 和 5 都已不复存在了。

甲：噢，这就是逆波兰。这个名字有点奇怪，你能告诉我它是什么意思吗？

乙：这种表示法是波兰逻辑学家 J. 卡西维兹发明的，因而得名。实际上有两种，一是波兰表示法，一是逆波兰法。我们平常所用的运算表示法，比如 A、B 两数相加，都是把运算符“+”放在中间，写做 A+B，称为中缀表示法，而波兰和逆波兰则是分别把运算符放在前边和后边，表达式写出来就是+AB 和 AB+，又称为前缀法和后缀法。FORTH 用的是后缀法，就象你看到的 3 5+，请注意 3 和 5 之间要有空格，否则有可能误认为 35。其他高级语言的编译系统常把它做为中间语言表示法。

甲：把运算符放在后边有什么好处吗？

乙：FORTH 中，数据存放于堆栈，后缀法是与之相应的表示法。它不用括号，运算的先后顺序由表达式决定，这样系统处理起来简单，因此就能提高速度。

甲：不用括号？比方(a−b)＊c 用逆波兰怎么表示呢？

乙：a b−c＊。先做 a 减 b，其结果乘以 c。首先，a 和 b 先入栈，见图 2a，遇到“−”时，得出 a−b，见图 2b。随后 c 再入栈，见图 2c，遇到“＊”时，(a−b)与 c 相乘，得出(a−b)＊c，见图 2d。

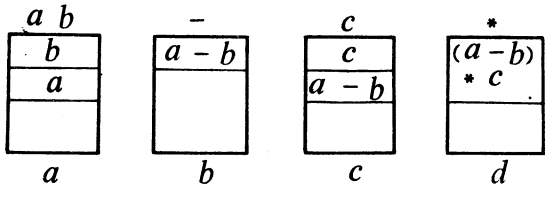


图　2

甲：(a−b)与 c 相乘，这里的汉语表达法，正好可以看做后缀表达法。

乙：你真细心。这样理解起来就方便多了。

甲：虽说如此，还是不太习惯。

乙：刚开始这是难免的，时间一长就能适应了。

甲：堆栈内数据变化也不易弄清，有好办法吗？

乙：这一般是用在括号内加注释的办法来解决。a b−c＊就可以这样注释：

a(--a)　b(a--ab)　　−(ab--a−b)

c(a−b--a−bc)　＊(a−bc--a−b＊c)

括号在 FORTH 中是注释用的，不会被执行。括号内，虚线左边是执行前栈状态，右边是执行后栈状态。将来熟练了，就不必注释得这么细了。

甲：这下清楚多了。前边的 3　5+.中的“.”相当于 BASIC 中的 PRINT 吗？

乙：差不多，它从栈中取出一个数据，显示在屏幕上，栈状态是(n--)。这里 n 代表 16 位有符号数，这是 FORTH 中常用的数据类型。

甲：这样执行“.”后栈中数就消失了，如果还想用又怎么办呢？

乙：那就先用 DUP，意思是复制栈顶数据，栈状态是(n--nn)。把这个式子改成 3　5+DUP.，输出后栈中还留有一个 8。由于堆栈的重要性，FORTH 安排了一组对堆栈进行操作的动词，DUP 就是其中之一。其余的列表如下：

SWAP(n_1n_2--n_2n_1)　　栈顶和次栈顶两数对换

DROP(n--)　　丢掉栈顶一个数

OVER(n_1n_2--$n_1n_2n_1$)　复制次栈顶

ROT($n_1n_2n_3$--$n_2n_3n_1$)旋转，把栈顶第三项旋转到栈顶

甲：能举几个例子，说明它们的用法吗？

乙：行。你输入 1　2　3↙，再...↙，该显什么？

甲：堆栈是后进行先出，该显 3　2　1. 想按原来的顺序显出 1　2　3，看样子上边的几个动词该派上用场了。

乙：此时用 ROT. SWAP. .↙或 SWAP ROT…↙都能显出 1　2　3。我给你出个题，现在栈顶状态是(--abc)，想要做(a+b)/c，用逆波兰该怎么做呢？

甲：我看看，先把 a 和 b 移到最上边，该用 ROT ROT+，栈顶就是(c　a+b)。现在该做除法了吧。

乙：还不行。除法要求被除数放在次栈顶。

甲：那就先用 SWAP。总起来就是 ROT ROT+SWAP /。

乙：完全正确，不防再给它加上注释，继续熟悉一下。

甲：DROP 有用吗？

乙：很有用。为了说明，我先介绍两个动词：

/MOD(n_1n_2--n_3n_4)　　n_1 除以 n_2，n_3：余数，n_4：商

MOD　(n_1n_2--n_3)　n_1 除以 n_2，n_3：余数

我所用的版本里，没有 MOD，在只需要余数时，就用得着 DROP 了。比方 10 3 /MOD DROP. ↙显 1。

甲：用/MOD DROP 代替 MOD，是否有些麻烦？

乙：如果嫌麻烦，可以定义一个 MOD。方法是，

:MOD /MOD DROP;

甲：你最好仔细讲解一下定义是怎么回事。

乙：因为篇幅关系，这回不行了，下次再说吧。

中华学习机图书管理程序

甘肃西和县西路乡王磨村　王二珠

不少人手头都有几十本书籍，对它们的管理是件麻烦事情。但若用本文程序管理，就方便多了。

特点：采用模块结构，使用方便。而且在检索时，除了采用多种检索方法外，不论用那种方法检索，都会列出书的开本，这就对寻找这本书带来了方便。最后还显示出所检索的书有多少本和它们的总价值。

使用时应注意：

(1)所用盘上应有 CHAIN 程序。可用 FID 拷贝。

(2)首次使用应先建立书库，以后再不用建立了。

(3)图书库文件名已取了 TSK. L100，也可改为用输入语句输入。

(4)如把主控程序 TSGL 作为"哈罗"程序存在盘上，那使用就更方便了。

(5)除了 TSGL 程序外，其它程序都不能单独起动。

主控程序(TSGL)

```
10  D$=CHR$(4):PRINT  D$;"MAXFILES 1"
20  PRINT D$"PR#3":PRINT:HOME
30  TT$="*********************":
PRINT TT$:PRINT TAB(1);"*";TAB(14);"图书管理";
TAB(33);"*":PRINT TT$
40  PRINT"      功能表:":PRINT"1. 建立书库 2. 图书检索
3. 图书扩充 4. 修改图书 5. 删除图书 6. 退出系统"
50  INPUT"你用哪一个(1-6)?";Y$:A=VAL(Y$):IF A
<=0 OR A>6 THEN 50
60  IF A=6 THEN HOME:VTAB6:PRINT TAB(15);"再
见!";CHR$(19):NEW
70  O$=D$+"OPEN TSK. L100,L100":C$=D$+
"CLOSE":R$=D$+"READ TSK. L100,R":W$=D$+
"WRITE TSK. L100,R":IF A=1 THEN 110
80  PRINT O$:PRINT R$;0:INPUT N,F$,Z$,L$,B
$,T,L,S:PRINT C$:P=L:PRINT"共有记录";P;"个"。
90  PRINT D$;"BLOAD CHAIN,A520"
100  PRINT:ON A-1 GOTO 150,160,170,170
110  INPUT"是第一次使用吗？(Y/N)";Y$:IFY$="Y"
THEN 130
120  PRINT"请选用'扩充图书'项!";CHR$(19):GOTO 20
130  INPUT"请输入现在日期(年.月):";T:E=0:N$="书
名":Z$="作者":L$="类别":B$="出版社":L%=0:S
=0
140  PRINT O$:PRINT W$;0:PRINT E;",";N$;",";Z
$;",";L$;",";B$;",";T;",";L%;",";S:PRINT C$:
PRINT"完成!";CHR$(19):GOTO 20
150  CALL 520"图书检索"
160  CALL 520"增加图书"
170  CALL 520"修删图书"
```

图书检索：

```
10  PRINT D$;"PR#3":PRINT:ONERR GOTO 190
20  HOME:INPUT"是否检索?";Y$:IF Y$="N"THEN
    190
30  DIM N$(P),V%(P),E(P),L%(P):HOME:PRINT TT
$:PRINT TAB(1);"*";TAB(14);"图书检索";TAB(33);
"*":PRINT TT$
40  PRINT"      功能表:":PRINT"1. 全部 2. 编号 3. 类
别 4. 出版社":PRINT"5. 书名 6. 作者 7. 单价 8. 时间 9. 退出"
50  INPUT"      你选哪一个?(1-9)";Y$:ZJ=0:C1=
0:A=VAL(Y$):IF A<=0 OR A>9 THEN 50
60  ON A GOSUB 80,90,100,160,140,150,170,180,190
70  GOTO 20
80  N1=1:N2=P:GOSUB 200:RETURN
90  INPUT"起始号:";N1:INPUT"结束号:";N2:GOSUB
    200:RETURN
100  HOME:FOR I=1 TO14:READ H$:PRINT I;".";H$;
"  ";:NEXT:PRINT:RESTORE
110  DATA 语文,数学,物理,地理,化学,生物,历史,英语,
政治,科普,技术,电子,棋术,计算机
120  INPUT"      你选哪一个?(1-14)";Y:IF Y<1 OR Y
>14 THEN 120
130  FOR I=1 TO Y:READ H$:NEXT:RESTORE:E$=
"":I$="":G$="":GOSUB 230:RETURN
```

```
140 INPUT"请输入书名:";E$:G$=""":I$=""":H$=
    "":GOSUB 230:RETURN
150 INPUT"请输入作者名:";G$:E$=""":I$=""":H$
    =""":GOSUB 230:RETURN
160 INPUT"请输入出版社名:";I$:E$=""":H$=""":G
    $=""":GOSUB 230:RETURN
170 INPUT"请输入起始单价:";S1:INPUT"请输入结束单
    价:";S2:T1=0:T2=0:GOSUB260:RETURN
180 INPUT"请输入起始时间:";T1:INPUT"请输结束时
    间:";T2:S1=0:S2=0:GOSUB 260:RETURN
190 PRINT D$;"RUN TSGL"
200 D=N2-N1+1:K=0:PRINT O$
210 FOR I=1TO D:B=N1+I-1:GOSUB 390:BJ=0:GO-
    SUB 380:IF T=0 THEN C1=C1+1
220 NEXT:PRINT C$:G$=""":I$=""":H$=""":E$=
    "":GOSUB 310:RETURN
230 K=0:PRINT O$
240 FOR I=1 TO P:B=I:GOSUB 390:IF L$=H$ OR F$
=E$ OR Z$=$G$ OR B$=I$ THEN BJ=0:GOSUB
380
250 NEXT:PRINT C$:GOSUB 310:RETURN
260 K=0:PRINT O$:IF S1=0 AND S=0 THEN 290
270 FOR I=1 TO P:B=I:GOSUB 390:IF S>S1 AND S<S2
    THEN BJ=S:GOSUB 380
280 GOTO 300
290 FOR I=1 TO P:B=I:GOSUB 390:IF T>T1 AND T<
    T2 THEN BJ=T:GOSUB 380
300 NEXT:PRINT C$:GOSUB 310:RETURN
310 HOME:U=0
320 FOR I=1 TO K
330 IF E(I)=0 THEN 350
340 PRINT V%(I);"  ";N$(I);TAB(25);E(I);TAB
    (32);L%(I);:GOTO 360
350 PRINT V%(I);"  ";N$(I);TAB(24);I$;H$;G$;
    TAB(32);L%(I)
360 U=U+1:IF U=INT(U/8)*8 THEN GET Q$
370 NEXT:GET Q$:PRINT"总数=";U-C1;"本";TAB
    (20);"价值=";ZJ;"元";CHR$(19):RETURN
380 K=K+1:V%(K)=X%:N$(K)=F$:E(K)=BJ:
    L%(K)=L%:ZJ=ZJ+S:RETURN
390 PRINT R$;B:INPUT X%,F$,Z$,L$,B$,T,L%,
    S:RETURN
```

增加图书

```
10 PRINT D$;"PR#3":PRINT:HOME:ONERR GOTO 130
20 INPUT"你想增加图书吗? (Y/N)";Y$:IF Y$="N"
   THEN 130
30 HOME:PRINT TT$:PRINT TAB(1);"*";TAB(14);
"增加图书";TAB(33);"*":PRINT TT$:Q$="请输入":
DIM N$(7)
40 DATA 书名,作者姓名,类别,出版社名,出版日期,此书
开本,此书单价
50 FOR I=1 TO 7:READ H$
60 PRINT Q$;H$;:INPUT N$(I):IF I>4 AND VAL(N
$(I))<=0 THEN PRINT"请重新输入!":GOTO 60
70 NEXT:IF P=0 THEN P=1:B=P:GOTO 100
80 PRINT O$:FOR I=1 TO P:PRINT R$;I:INPUT X,F
$,Z$,L$,B$,T,L%,S:IF T=0 THEN B=P:P=X:
PRINT C$:GOTO 100
90 NEXT:P=P+1:B=P:PRINT C$
100 PRINT"请稍等,正在写入!":PRINT O$:PRINT W$;
P:PRINT P;",";N$(1);",";N$(2);",";N$(3);",";N
$(4);",";N$(5);",";N$(6);",";N$(7):PRINT R$;
0:INPUT X,F$,Z$,L$,B$,T,L%,S
110 PRINT W$;0:PRINT X;",";F$;",";Z$;",";L$;
",";B$;",";T;",";B;",";S:PRINT C$:PRINT"完成!"
120 INPUT"你还要增加吗? (Y/N)";Y$:IF Y$="Y"
    THEN RESTORE:GOTO 50
130 PRINT D$;"RUN TSGL"
```

修删图书

```
10 PRINT D$;"PR#3":PRINT:HOME:ONERR GOTO 150
20 IF A=4 THEN H$="修改":GOTO 40
30 H$="删除"
40 PRINT"你想";H$;"图书吗? (Y/N)";:INPUT Y$:IF
   Y$="N" THEN 150
50 HOME:PRINT TT$:PRINT TAB(1);"*";TAB(14);
"图书";H$;TAB(33);"*":PRINT TT$:DIM P$(7)
60 PRINT"请输入要";H$;"记录的书名或编号";:INPUT
   J$:IF VAL(J$)<>0 THEN 90
70 PRINT O$:FOR I=1 TO P:PRINT R$;I:INPUT E,P
$(1),P$(2),P$(3),P$(4),P$(5),P$(6),P$(7):IF
P$(1)=J$ THEN PRINT C$:GOTO 110
80 NEXT:PRINT C$:PRINT"无此书名!":GOTO 140
90 X=VAL(T$):IF X>P OR X<=0 THEN 60
100 PRINT O$:PRINT R$;X:INPUT E,P$(1),P$(2),
    P$(3),P$(4),P$(5),P$(6),P$(7):PRINT C$
110 U$="  ":PRINT"本记录如下:":PRINT E;U$;:
    FOR I=1 TO 7:PRINT P$(I);U$;:NEXT:PRINT
120 PRINT"是否";H$;"? (Y/N)";:INPUT Y$:IF Y$=
    "N" THEN 140
130 ON A-3 GOSUB 160,220
140 PRINT"还要";H$;"吗? (Y/N)";:INPUT Y$:IF Y$
    ="Y" THEN 60
150 PRINT D$;"RUN TSGL"
160 FOR I=1 TO 7:GOSUB 200:IF Y$="Y"THEN GOSUB
    210
170 IF I>4 AND VAL(P$(I))<=0 THEN PRINT"请重新
    输入!":GOSUB 210:IF VAL(P$(I))<=0 THEN 170
180 NEXT:RESTORE:GOSUB 230:RETURN
190 DATA 书名,作者,类别,出版社,日期,开本,单价
200 READ K$:PRINT K$;"修改? (Y/N)";:INPUT Y$:
    RETURN
210 PRINT K$"改为";:INPUT P$(I):RETURN
220 FOR I=1 TO 4:P$(I)="{      }":NEXT:P$(5)=
"0":P$(6)="0":P$(7)="0":GOSUB 230:RETURN
230 PRINT"请稍等,正在";H$;"!":PRINT O$:PRINT W
$;E:PRINT E;",";P$(1);",";P$(2);",";P$(3);",";
P$(4);",";P$(5);",";P$(6);",";P$(7):PRINT C$:
PRINT"完成!":RETURN
```

ProDOS 磁盘操作系统入门(续)

廖 凯

第五章 程序设计

这章将结合 APPLESOFT 命令和 ProDOS 命令，介绍如何在 ProDOS 系统下进行程序设计。由于 DOS3.3 与 ProDOS 最大区别之一是在数据存储方面，所以将着重介绍文本文件在 ProDOS 系统下的使用。在这章中的大部分程序是在 80 列显示方式下运行的。如果要在 40 列方式下运行，只要略微改动一下即可。

一、APPLESOFT BASIC 基础

这部分只对 APPLESOFT BASIC 作简单的介绍，不熟悉 BASIC 语言的初学者，请参阅有关方面的书籍。

1. 基本指令

END	使程序停止运行
FOR/NEXT	循环执行 FOR 与 NEXT 之间的程序行
GOTO	使程序转到指定的行号去执行
HOME	清除屏幕并将光标移到左上角
IF/THEN	条件语句，如果 IF 的表达式值为真，就执行 THEN 子句；如果为假，就执行下一个语句
INPUT	由当前输入设备接收数据给计算机
NEW	清除当前内存中的程序及所有变量
PRINT	输出字符符号到屏幕或指定的输出设备

2. 开始工作

将 ProDOS User's Disk 磁盘插入驱动器 1 中并启动计算机，过一会儿屏幕将显示主菜单，按 B 键跳出主菜单，进入 BASIC 状态。打入 HOME 指令，清除屏幕，光标移至左上角，再打入 NEW 指令，清除内存程序。

BASIC 执行方式有两种：立即方式和间接方式。立即方式是指直接打入语句或指令，按回车键后立即执行。如刚才的 HOME 和 NEW 指令。间接方式是指每个语句都有一个行号，程序需 RUN 指令来执行。大多数 APPLESOFT BASIC 指令均可在两种方式下执行。

下面的程序是在屏幕上打印 1 到 10 的数字：

```
10  FOR I= TO 10
20  PRINT I
30  NEXT I
40  END
```

运行后，我们可以看到每当 FOR/NEXT 循环的变量 I 递增 1 时，打印语句就将变量 I 的值打印出来，如果将行号 10 改为：

```
10  FOR I=1 TO 10 STEP 2
```

运行结果则是印出 1 至 10 之间的奇数，这是因为变量 I 每次递增 2。如果我们再将行号 20 改为：

```
20  PRINT I;或 20 PRINT I,
```

运行程序后，你会看到它们的变化是在屏幕输出格式上。分号不允许输出字符之间有空格，而逗号则可以，它是按制表域输出字符。

下面的程序是计算汽车每加仑汽油跑多少英里：

```
10  HOME
20  INPUT"NUMBER OF GALLONS IN A FULL TANK?";FT
30  INPUT"ENTER THE TOTAL MILES DRIVEN?";TM
40  MPG=TM/FT
50  PRINT"MILES PER GALLON IS";MPG
70  END
```

运行后程序要你输入加仑数和英里数给计算机，计算机会自动计算并将结果打印出来。在上面的程序中再加入以下语句：

```
35  IF TM=0 THEN 70
60  GOTO 20
```

这样就可以连续计算了，直到你输入的英里数为零，程序才停止运行。

二、数组和串变量

这部分讨论 ProDOS 如何处理串变量的输出。

1. 有关指令

ASC	返回串表达式值第一个字符的 ASCII 码
CALL	使程序转向指定的内存单元开始的机器语言子程序
CHR $	返回一个与表达式值相对应的 ASCII 字符
DIM	定义数组，在内存中分配变量空间
GET	等待用户由键盘输入单个字符，无需按回车键
GOSUB	使程序转向子程序，当遇到 RETURN 时，程序立刻转回到 GOSUB 语句的下一句
INVERSE	使字符反相输出到屏幕上
LEFT $	返回字符串中最左边的若干字符
LEN	计算字符串的长度
MID $	返回字符串中指定的若干字符
NORMAL	使字符正常输出到屏幕上
ONERR GOTO	当错误发生时，转到指定的语句
PEEK	返回指定内存地址单元中的值
READ/DATA	当有 READ 指令时，计算机寻找 DATA 指令，将 DATA 语句中的数据赋给 READ 语句中的变量
REM	用于书写程序的注解，运行时忽略不计
RETURN	使程序回到上一个 GOSUB 后面的语句
RIGHT $	返回字符串中最右边的若干字符
STOP	终止程序执行并显示行号
VAL	将字符型数据转换成算术型

2. 数组

数组用于识别一组同一名称的变量，数组的每个元素用赋给它的编号来识别。数组可用于串变量、整数或实数。下面的程序将说明如何组成数组 G。

```
10  DIM G(4)
20  FOR I=0 TO 4
30  READ G(I)
40  NEXT I
50  DATA 22.68,1002,601,-.033,98
```

```
60  REM Print the numbers with another loop
70  FOR H=0 TO 4
80  PRINT G(H)
90  NEXT H
100  END
```

这程序首先定义一个数组 G，它有五个元素 G(0)至 G(4)，然后经循环语句和 READ/DATA 语句将数据赋给数组的五个元素，最后由 FOR/NEXT 语句和 PRINT 语句将数组五个元素的值顺序打印出来。

3. 串变量

串变量表示一个字符串。串变量将 $ 号放在变量的最右边以区别于数值变量。在给串变量赋值时，字符串必须用引号括起来。如：

A$ ="a string 12345"

A$ ="good morning"

下面的程序是询问用户是否使用 80 列卡，即屏幕显示是否是 80 列。程序会根据用户的回答来决定是用 40 列还是用 80 列方式显示磁盘目录。由于 ProDOS 使用分级文件结构，所以程序设计者必须注意路径。部首(PREFIX)被设为要 CATALOG 的磁盘。

```
1   REM Program to CATALOG or CAT different ProDOS disks.
4   HOME
5   D$ =CHR$ (4)
10  INPUT"Is the screen in 80-column mode?";A$
20  IF LEFT$ (A$ ,3)="YES"OR LEFT$ (A$ ,3)="yes"
    THEN 50
30  IF LEFT$ (A$ ,2)="NO"OR LEFT$ (A$ ,2)="no"
    THEN W=1:GOTO 50
40  HOME:GOTO 10
50  INPUT"Enter prefix to be CATALOGed:";P$
55  ONERR GOTO 170
60  IF LEFT$ (P$ ,1)<>"/"THEN P$ ="/"+P$
70  PRINT D$ ;"PREFIX";P$
80  IF W=1 THEN 120
90  PRINT D$ ;"CATALOG"
100  GOTO 150
120  PRINT D$ ;"CAT"
150  END
170  POKE 216,0
180  GOTO 50
```

行号 5 是将 ASCII 码为 4 的字符(即 CTRL-D)赋给变量 D$。行号 10 等待用户输入大写或小写的 YES 或 NO，行号 20 和行号 30 分析 INPUT 语句并进行比较。若符合条件就转至行号 50；若不符就执行行号 40。行号 50 是等待用户输入要被列目录的部首，行号 60 是分析用户输入的部首，若卷目录前面没有斜线，那么将自动加上一个。行号 55 是错误处理子程序，如果输入了错误的部首，那么程序将转向行号 170 处理。行号 70 至行号 150 是程序根据分析结果，以 80 列或 40 列显示方式列印目录。

4. 进一步介绍 PREFIX

当你初次启动系统时，系统没有指定磁盘的部首。在没有使用 PREFIX 命令的情况下，你可以将驱动器号与命令一起使用，如同 DOS3.3 一样，对磁盘进行操作，而不会有路径错误发生。

在立即方式下执行 PREFIX 命令时，当前的部首将被显示出来。此命令也可在程序方式下设置部首：

```
5  D$ =CHR$ (4)
10  INPUT"NEW PREFIX:";P$
20  PRINT D$ ;"PREFIX";P$
```

你也可以将部首存储在一个串变量中：

```
5  D$ =CHR$ (4)
10  PRINT D$ ;"PREFIX";INPUT"";P$
```

5. CREATE 命令

ProDOS 具有产生任何文件类型的能力。下面是一个用 CREATE 命令来产生不同文件类型的程序。CREATE 命令最常用于产生目录文件。

```
100  HOME:D$ =CHR$ (4)
104  PRINT D$ ;"PR#3"
110  PRINT"Create A File Type"
120  PRINT"1. Applesoft Program    BAS"
130  PRINT"2. Applesoft Variable        VAR"
140  PRINT"3. Binary              BIN"
150  PRINT"4. Directory           DIR"
160  PRINT"5. ProDOS System Program    SYS"
170  PRINT"6. Relocatable Code                REL"
180  PRINT"7. Text            TXT"
190  PRINT"8. User defined              $F#"
200  PRINT:GET N$ :N=VAL(N$ )
210  IF ASC(N$ )<48 OR ASC(N$ )>56 THEN 200
220  IF N=1 THEN A$ ="BAS"
230  IF N=2 THEN A$ ="VAR"
240  IF N=3 THEN A$ ="BIN"
250  IF N=4 THEN A$ ="DIR"
260  IF N=5 THEN A$ ="SYS"
270  IF N=6 THEN A$ ="REL"
280  IF N=7 THEN A$ ="TXT"
290  IF N=8 THEN A$ ="$F#"
300  PRINT: PRINT "Give the pathname or appropriate file
     name to CREATE a ";A$ " type file":PRINT
305  INPUT"";F$
310  IF N=8 THEN 350
320  PRINT D$ ;"CREATE";F$ ",T";A$
330  GOTO 400
350  PRINT:PRINT"Enter the User Defined number between 1
     and 8 inclusive":PRINT
360  INPUT"";NO$
370  IF VAL(NO$ )<1 OR VAL(NO$ )>8 THEN 350
380  A$ ="$F"+NO$
390  GOTO 320
400  END
```

注：行号 104 是启动插在 3 号槽口的 80 列卡，后同。对于中华学习机可将此句删去或将槽口号改为 80 列卡所插的槽口号。 (下转第 16 页)

C 语言初阶讲座

第九讲　指针与函数、结构

李文兵

1. 函数指针

指针不仅可以指向一般变量、数组，而且还可以指向函数。我们把指向函数的指针叫做函数指针，其一般定义形式如下：

数据类型　(*标识符)();

例如：　　int (*p)();

表示 p 是一个指向函数的指针，且说明 p 所指向的函数的返回值是整型数据。

函数指针的用法，如练习 9.1 所示。

```
A>type exp9_1.c
main()
{ int (*p1)(),print1();
  int (*p2)(),print2();
  p1=print1;
  p2=print2;
  (*p1)(123);
  (*p2)(456);
}
print1(x)
int x;
{ printf("%d\n",x);}
print2(y)
int y;
{ printf("%d\n",y);}
```

```
A>exp9_1
123
456
```

在该程序的函数体中：

1、2 行为说明语句，说明 p1、p2 为函数指针，print1()和 print2()为函数；

3、4 行为函数指针赋值语句，用来使 p1 和 p2 分别指向函数 print1()和 print2()；

5、6 行为函数的函数语句方式调用，即利用函数指针 p1 和 p2 分别调用函数 print1()和 print2()，其中 123 和 456 为代替形参的实参。

再看一个例子，如练习 9.2 所示。

```
A>TYPE EXP9—2.C
#include "time.h"
#inciude <stdio.h>
#include "stdlib.h"
static int (*cmpf)();
#difine MAX 10
main()
    { int quick(int *,int,int(*)()),cmp(int,int);
      int i,arry[MAX];
      long now;
      srand((unsigned)time(&now));
      for(i=0;i<MAX;i++)
      printf("%d ",arry[i]=rand());
      quick(arry,MAX,cmp);
      printf("\n\n");
      for(i=0;i<MAX;i++)
      printf("%d ",arry[i]);
      printf(("\n\n");
}
quick(item,count,cmp)
int item[],count,(*cmp)();
{ cmpf=cmp;
  qs(0,count-1,item);
}
qs(l,u,item)
int l,u,item[];
{ int i,j,x,y;
  i=l;j=u;
  x=item[(l+u)/2];
  do { while((item[i]>x)&&(i<u))i++;
       while((item[j]<x)&&(j>l)j--;
       if(i<=j)
       { y=item[i];
         item[i]=item[j];
         item[j]=y;
         i++;j--;
         }
  } while(i<=j);
    if(l<j)qs(l,j,item);
    if(i<u)qs(i,u,item);
  }
  int cmp(x,y)
  int x,y;
  { if(x>y) return(-1);
    else if(x==y) return(0);
    return(1);
  }
```

```
A>exp9_2
15250 7082 21072 29365 24173 1478 22943 7166 10947 23544
29365 24173 23544 22943 21072 15250 10947 7116 7082 1478
```

该练习是快速排序程序。快速排序是靠函数 qs()实现的，qs()为递归函数，2 次递归调用其本身。快速排序原理是以数组的中间元素为基础值，把比中间元素值大的放在中间元素的一边，小的放在另一边。把小的放在中间元素的左边，是升序；而把大的放在左边，

是降序。按升序，还是按降序，qs()中是靠连结 item()与 x 的关系运算符决定的。这里，qs()中是按降序处理的。如图 9.1 所示，第 1 次选的中间元素其值为 5，以它为基准进行比较，则把比 5 大的换到 5 的左边，把小的放在 5 的右边。

原数组：item〔0〕……　　item〔17〕

1	4	2	6	3	7	4	9	5	1	6	4	7	6	8	2	9	0

x＝item〔(0＋17)/12〕的结果为：x＝5

第 1 次排序后：

9	8	6	6	7	7	6	9	5	1	4	4	3	2	4	2	1	0

图 9.1 qs()降序第 1 次排序后情况

改变 item〔i〕与 x 间、item〔j〕与 x 间不等号的方向，即可改为升序排序。

指向函数的指针用 static 说明为全局变量，即：

```
static int ( * cmpf)();
```

其中 cmpf 为指针名，用它指向比较函数 cmp()，即：

```
cmpf＝cmp;
```

2. 返回指针的函数

说明语句：

```
int * p();
```

表示 p 是一个函数，一个返回整型指针的函数。返回指针的函数的用法，如练习 9.3 所示。

```
A>type exp9_3.c
#include <stdio.h>
main()
{char *cpy(char *,char *);
 char s[]="abcd";
 char d[]="efgh";
 printf("%s",cpy(s,d));
}
char *cpy(s,d)
char *s,*d;
{int i;
 char *p;
 p=d;
 if(*s==NULL) exit(-1)
 do {*d++=*s++;
 }while(*s! =NULL);
return (p);
}
A>exp9_3
abcd
```

该程序是从 s〔〕中把字符串拷贝到 d〔〕中。这时函数 cpy〔〕的返回值为指针 d。

3. 指针参数与函数参数值调用

在 C 语言里，函数的参数值有两种调用方法：

(1)值调用(call by value)　　值调用的例子如练习 9.4 所示。

```
A>type exp9_4.c
main()
{int i,j;
 char a;
 i=110;j=220;a='c';
 func(i,j,a);
}
func(i,j,k)
int i,j;
char k;
{ printf("arg1=%d\narg2=%d\narg3=%c\n",i,j,k);
}

A>exp9_4
arg1=110
arg2=220
arg3=c
```

该程序调用函数 func()时，引用 3 个参数。

(2)参考调用(call by reference)　　这是使用指针的值调用，其例如练习 9.5 所示。

```
A>type exp9_5.c
main()
{int i=111,j=222;
 char a='a';
 func(&i,&j,&a);
 printf("i=%d\nj=%d\na=%c\n",i,j,a);
}
func (i,j,k)
int *i,*j;
char *k;
{ printf(" arg1=%d\narg2=%d\narg3=%c\n",*i,*j,*
  k);
  (*i)++;
  *j++;
  *k++;
}

A>exp9_5
arg1=111
arg2=222
arg3=a
i=112
j=222
a=a
```

从程序中，我们可以看到，函数 func()的三个参数 i、j、k 都是指针，也就是说，i、j、k 为指针参数。这样，在调用它时，必须使其参数与之对应，即也必须是指针类型，因此，要在一般变量 i、j、a 前面冠以取址运算符 &，即使其三个参数为：

&i,&j,&a

不返回值的函数，可用关键字 void 来说明，如练习 9.6 所示。

```
A>type exp9_6.c
#include <stdio.h>
main()
{void pnt(char *);
 char s[]="abcd";
 pnt(s);
```

```
}
void pnt(s)
char *s;
{ do {printf("%c", *s++);
    }while (*s! =NULL);
}
```

```
A>exp9_6
abcd
```

从练习 9.6 可以看出，在 Turbo C 中可用 void 类型来显式说明一个无返回值的函数，除此之外，void 还有如下一些有用法：

①用来说明一个空参数表　如练习 9.7 所示。

```
A>type exp9_7.c
void putmsg(void)
{ printf("Hello,world\n")
}
main()
{ putmsg();}
```

```
A>exp9_7
Hello,world
```

②用来作一个特殊结构，如：

```
(void) getch();
```

其功能是暂停执行，直到按任何键为止。

③说明一个 void 指针。注意，这不是空指针，而是建立一个数据目标为任何类型的指针，对它们的类型无需知道。可把任何指针赋值为 void 指针，反之亦然，而无需事先安排。

void 指针不能使用 * 。

4、指针与结构

(1)指针型结构成员　指针变量同样可以作结构成员。下面的结构定义中，就把指针变量 pname 当作了结构的成员。

```
struct person{
        int number;
        char *pname;
}
```

这样的结构的用法，如练习 9.8 所示。

```
A>type exp9_8.c
#include<stdio.h>
struct person {
        int number;
        char *pname;};
main()
{int i;
 struct person qp[10];
 for(i=0;i<10;i++)
     qp[i].pname=NULL;
 qp[1].number=100;
 qp[1].pname="aoi";
 qp[2].number=200;
 qp[2].pname="akai";
 /*print out*/
 i=1
 do { printf("%d %s\n",
      qp[i].number,qp[i].pname);
      i++;
 } while (qp[i].pname!=NULL);
}
```

```
A>exp9_8
100 aoi
200 akai
```

(2)结构指针　指针不仅可以指向变量、数组、指针、函数，也可以指向结构。我们把指向结构的指针，叫做结构指针。结构指针在链表、二叉树等数据结构中是很有用的。设一链表的结构为：

```
struct address{
    char name[20];
    char city[40];
    char zip[5];
    struct address *next;
};
```

如果定义如下指针 top：

```
struct address *top
```

则指针 top 就是指向该结构的一个结构指针。这时，就可使用->运算符，引用该结构的各个成员，如下所示：

```
top ->name
top ->city
top ->zip
top ->next
```

连接结构数据可使用引用自身的结构，如练习9.9中的结构 address。练习9.9可连接4个新的结构，并能读出全部第一成员姓名。

```
A>type exp9_9.c
#include <stdio.h>
#include"stdlib.h"
#include"alloc.h"
struct adderss {char name[9];
                char city[7];
                char zip[7];
                strudt address *next;
               };
main()
  {struct address *new, *last=0;
   int i=0;
   do {if((new=malloc(sizeof(struct address)))==0)
       {printf("out of memory\n");
       exit(0);
   }
   if(last==NULL)
          new->next=NULL;
```

(下转第13页)

学用单片机

汽车转弯信号灯控制

（KDC—Ⅲ最小系统应用实例）

张培仁　刘振安

汽车控制面板上有一个控制杆。此控制杆有三个位置：

中间位置时，汽车不转弯；向上时，汽车左转；向下时，汽车右转。汽车转弯时，要求左右尾灯，左右头灯和仪表板上的2个指示灯发出闪烁的信号。当应急开关合上时，所有6个信号灯都应闪烁。汽车刹车时，2个尾灯发出不同闪烁信号。如刹车时正在转变，则相应的转弯闪烁信号不受影响。

汽车转弯或应急状态下，外部信号灯和仪表板指示灯的闪烁频率为1Hz，称低频信号。当停靠开关合上时，外部信号灯以高频（约30Hz）频率闪烁，以适应低亮度背景的使用场合，增加亮度。

硬件电路原理如图1所示：

图中用1.1KΩ电阻与发光二极管串联模拟车灯，两旁并上1.5KΩ电阻消除了故障检测电路引起的发光二极管余光现象。故障指示灯可以检测所有信号灯是否有断路现象，从P3.4的电平来判断。

对汽车转弯灯控制系统功能可以用数字电路来实现，但灵活性不够。

从表一中列出转弯信号灯的真值表，我们用单片机最小系统来完成逻辑电路要完成的功能。

系统软件：

主要二部分，一部分是主程序，另一部分是中断服务程序。

主程序开始是输入，输出口线说明和变量定义。以后就是初始化，置定时器0，总允许中断，启动定时器0，等待等几个过程。定时器0和一个软件计数器SOB——DIV产生一秒的定时信号，以实现1Hz闪烁功能。

中断服务程序：

主要先断是否1秒到。1秒到重置

表1转弯信号工作的真值表

输入信号				输出信号			
刹车开关	应急开关	左转开关	右转开关	左头灯和仪表板灯	右头灯和仪表板灯	左尾灯	右尾灯
0	0	0	0	断	断	断	断
0	0	0	1	断	闪烁	断	闪烁
0	0	1	0	闪烁	断	闪烁	断
0	1	0	0	闪烁	闪烁	闪烁	闪烁
0	1	0	1	闪烁	闪烁	闪烁	闪烁
0	1	1	0	闪烁	闪烁	闪烁	闪烁
1	0	0	0	断	断	通	通
1	0	0	1	断	闪烁	通	闪烁
1	0	1	0	闪烁	断	闪烁	通
1	1	0	0	闪烁	闪烁	通	通
1	1	0	1	闪烁	闪烁	通	闪烁
1	1	1	0	闪烁	闪烁	闪烁	通

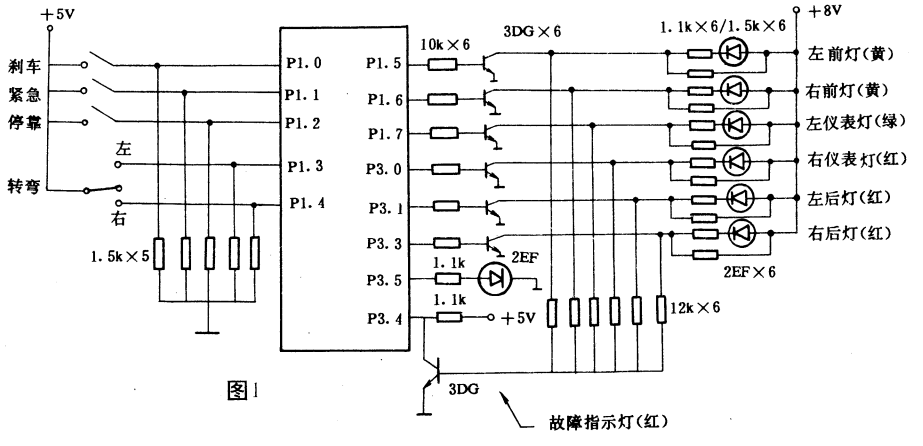


图1

T_0时间常数和 SOB-DIV 的值，并检查信号灯是否有故障。如果1秒未到，产生了30Hz，占空比是62.5%的信号，输出相应仪表左右灯，车前，车后灯，最后返回。

因为 MCS-51是一个功能很强布尔处理机，这方面指令很多，使运算速度快，程序可读性好。这是单片机的很大优点。单片机是功能很强易于进行数字逻辑运算的处理机。程序清单如下：

```
        BRAKE EQU P1.0          ;刹车
        EMERG EQU P1.1          ;应急
        PARK EQU P1.2           ;停靠
        L—TURN EQU P1.3         ;左转
        R-TURN EQU P1.4         ;右转
        L-FRNT EQU P1.5         ;左前灯
        R-FRNT EQU P1.6         ;右前灯
        L-DASH EQU P1.7         ;左仪表灯
        R-DASH EQU P3.0         ;右仪表灯
        L-REAR EQU P3.1         ;左后灯
        R-REAR EQU P3.3         ;右后灯
        TO     EQU  P3.4        ;故障检测输入
        S-FALL EQU P3.5         ;报警
        SUB-DIV EQU 20H         ;软件计数器
        SUB-DIV.0 EQU 00H
        SUB-DIV,1 EQU 01H
        SUB-DIV.2 EQU 02H
        SUB-DIV.7 EQU 07H
        HI-FREQ EQU SUB-DIV.0
        LO-FREQ EQU SUB-DIV.7
        DIM EQU PSW.1 ;
        ORG 1000H
        LJMP INIT
        ORG 100BH
        MOV TH0,#0F8H           ;重置定时器0
        PUSH 0D0H
        AJMP UPDATE
        ORG 1040H
INIT:   MOV TL0,#00H            ;定时器0置初值
        MOV TH0,#0F8H
        MOV TMOD,11110001B
        MOV SUB-DIV,244
        MOV IE,#10000010B
        SETB TR0
        ASD:SJMP ASD
UPDATE: DJNZ SUB-DIV,TOSERV
        MOV SUB-DIV,#224
        SETB L-FRNT             ;对线路检测故障
        SETB R-FRNT
        SETB L-DASH
        SETB R-DASH
        SETB L-REAR
        SETB R-REAR
        CLR L-FRNT
        JB TO,FAULT
        SETB L-FRNT
        CLR L-DASH
        JB TO,FAULT
        SETB L-DASH
        CLR L-REAR
        JB TO,FAULT
        SETB L-REAR
        CLR R-FRNT
        JB TO,FAULT
        SETB R-FRNT
        CLR R-DASH
        JB TO,FAULT
        SETB R-DASH
        CLR R-REAR
        JB TO,FAULT
        SETB R-REAR
        JB TO,TOSERV
FAULT:  CPL S-FALL              ;使报警闪一次
TOSERV: MOV C,SUB-DIV.1         ;30Hz 信号
        ANL C,SUB-DIV.0
        ORL C,SUB-DIV.2
        ANL C,PARK
        MOV DIM,C
        MOV C,L-TURN            ;控制左仪表灯
        ORL C,EMERG
        ANL C,LO-FREQ
        MOV L-DASH,C            ;控制左前灯
        MOV F0,C
        ORL C,DIM
        MOV L-ERNT,C
        MOV C,L-TURN            ;控制左后灯
        CPL C
        ANL C,BRAKE
        ORL C,F0
        ORL C,DIM
        MOV L—RFAR,C
        MOV C,R-TURN            ;控制右边灯
        ORL C,EMERG
        ANL C,LO-FREQ
        MOV R-DASH,C
        ORL C,DIM
        MOV R-FRNT,C
        MOV C,R-TURN
        CPL C
        ANL C,BRAKE
        ORL C,F0
        ORL C,DIM
        MOV R-REAR,C
        POP 0D0H
        RETI                    ;返回
        END
```

学装微电脑

学装 μp-80

Z80 CPU 与单片微处理器8031的交换

高殿斌

Z80 CPU 微电脑系统必须有一系列的外围 LSI 芯片支持。例如，需要 EP—ROM 存储器、RAM 存储器、I/O接口等，这样，使控制系统较复杂。

单片微处理器将 CPU、ROM、RAM 与 I/O 集成在一块 LSI 芯片中，体积缩小，集成度提高，为微电脑应用开发带来了很大方便，目前，较为流行的8位单片微处理器是 MCS51系列单片微处理器。

MCS51系列的特点

8051系列单片微处理器的主要特点

1）8位 CPU 为中央处理器。

2）能够进行位操作。

3）I/O 口有32条双向引脚，也能实现逐个引脚存取。

4）内藏128字节的数据 RAM。

5）有两个可编程的16位定时器/记数器。

6）内藏双重异步接收发送装置。

7）有五个中断源，是具有两级优先权的中断系统。

8）64K 字节的数据存储器和程序存储器严格分开。

单片微处理器8031

MCS-51系列单片微处理器概要如表1所示。

8051片内设有程序存贮器 ROM，适用于大批量的产品中引用。

8751片内设有程序存贮器 EP—ROM。可以根据需要对 EP—ROM 进行程序写入或程序擦除。必须具备专用的 EP—ROM 写入器。

8031片内没有程序存贮器。因此8031所有的取数均来自外部存贮器，很容易搞到的2716或者2732 EP—ROM 存贮器。

使用8031的中央处理部件

μP80微电脑套件是按功能划分的部件，通过接插件的连接，能够完成各种各样的实验。

“实践篇”的六块部件：键盘部件、写读部件、存储器部件，中央处理器部件、输入输出部件和4位数字显示部件。

本文讨论用8031替代中央处理部件的 Z80 CPU。这样，六块部件连接起来就组成一套简易 MCS—51系列开发系统，8031价格便宜，使用方便。如果再配备 EP—ROM 写入器，将为 MCS—51系列单片微处理器的应用开发提供极大方便。

8031中央处理部件的电路图如图1所示。

〈1〉数据总线 D_0～D_7

数据总线 D_0～D_7使用 P0.0～P0.7引脚。

和外部进行8位数据的双向传输。在实验过程中，下方连接存储器部件，可以看作外部程序存储器，上方连接输入输出部件，看作外部数据存储器。与 ALE 信号同步，输出8位数据。

选择外部程序存储器由 $\overline{PSEN}$ 信号控制。

P_0端口与外部数据存储器进行8位数据的输入、输出由 $\overline{RD}$、$\overline{WR}$ 信号控制。

〈2〉地址总线 A_0～A_7

在与 ALE 信号同步同时，随8位数据的输出输入低8位地址 A_0～A_7由74LS373输出。

表1　MCS—51系列单片微处理器一览表

ROM 型	无 ROM 型	EPROM 型	ROM 容量	RAM 容量	16位定时器	电路类型
8051	8031	(8751)	4KB	128B	2	HMOS
8051AH	8031AH	8751H	4KB	128B	2	HMOS
8052AH	8032AH	8752BH	8KB	256B	3	HMOS
80C51BH	80C31BH	87C51	4KB	128B	2	CHMOS
83C152	80C152	——	8KB	256B	2	CHMOS
83C51FA	80C51FA	87C51FA	8KB	256B	4	CHMOS
83C51FB	80C51FB	87C51FA	16KB	256B	4	CHMOS
83C51GA	80C51GA	87C51GA	4KB	128B	2	CHMOS

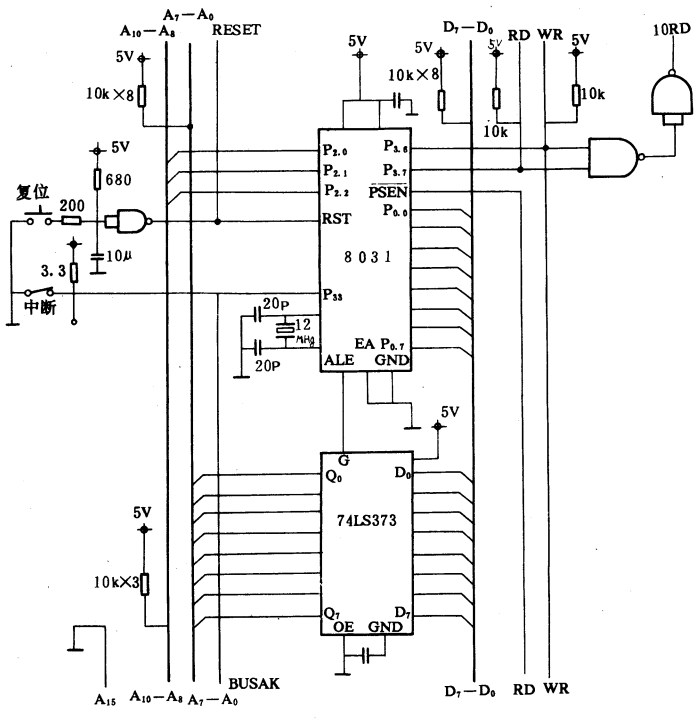


图1

〈3〉端口 P2.0～P2.2

为高3位地址线 A_8～A_{10}，所以，可以存取2K 地址以下的外部程序存贮器。

〈4〉只读信号 $\overline{RD}$

只读信号 $\overline{RD}$ 引脚是 P3.7引脚。

单片8031读外部数据存贮器的数据。

〈5〉写入信号 $\overline{WR}$

写入信号 $\overline{WR}$ 引脚是 P3.6引脚。

单片8031向外部数据存储器写入数据。

〈6〉地址锁存选通 ALE

为外部存贮器锁存地址使用 ALE 信号。

8031每一个指令周期由12个 T 组成。程序存贮器总线周期由6个 T 组成，所以一个指令周期一定要存取两个字节。

在 ALE 信号的下降沿，锁存总线上的地址信号。我们使用了74LS373存锁器。

〈7〉程序存贮器选通输出信号 $\overline{PSEN}$

单片8031输出 $\overline{PSEN}$ 信号，选择外部程序存贮器，将其数据再送入单片8031。

由图1可知，$\overline{PSEN}$ 引脚与存贮器部件的 $\overline{RD}$ 引脚相连。

〈8〉程序存贮器选择信号 $\overline{EA}$。

单片8031的 $\overline{EA}$ 引脚如果为“L”电平，程序存取过程全部为外部程序存贮器合成。

〈9〉主电源 V_{CC}。

与5V 电源连接。

〈10〉接地引脚 GND

与0V 电源连接。

〈11〉石英晶体振荡输入端 $XTAL_1$、$XTAL_2$。

为使在内部产生时钟脉冲，$XTAL_1$、$XTAL_2$引脚连接12MHz 的石英晶体和20P 的电容。

〈12〉复位引脚 RST

是单片8031初始化的引脚。

在引脚 RST 上保持24个 T 时间的“H”电平，CPU 停止执行指令，总线持续为休止状态。引脚 RST 的信号由“H”向“L”电平变化，经过12个 T 时间，CPU 又从0000H 地址开始执行程序。

〈13〉外部中断引脚 $\overline{INT_1}$

单片8031的 P3.3引脚为外部中断引脚 $\overline{INT_1}$。

由外部向 $\overline{INT_1}$ 引脚施加“L”电平、则向 CPU(8031)提出中断要求，程序存贮器从0003H 地址后执行指令。

邮购：

印刷线路板(包括8线插脚)20元/块

全套散件： 60元/套

开户行：工商行天津河东区唐口分理处 天津纺织工学院，订购8031单片套件。

联系人：天津纺织工学院机械系机电一体化室高殿斌

邮政编码 300160 电话：41.2833

(上接第42页)

未经定影，很容易被涂抹，所以产生字体拉丝的现象。

这种故障很少发生，日本制造厂家决定改进设计。

(二)修复 IBM9347磁带机故障

故障现象：磁带机读带很不稳定，有时能通过一盘，有时磁带全绕在卷带轴上，有时刚引带就读错。

分析与解决：脱机作诊断测试214，这是检验磁带机工作正确与否的测试项目。结果发生错误而停机，部件参考码为3004，65%的原因是张力臂移动错误。测试张力臂124手工移动张力臂位置，在诊断灯上显示出数值为95～255，超出正常范围，调整张力臂动片，使符合要求。把磁带装上去，仍不能引带。得到部件参考码为0013，90%的原因是 BOT 传感器故障。脱机作诊断测试122，在诊断灯上显示的数值均为零，换 BOT/EOT 部件后工作正常。

(上接第3页)

```
1170 FOR j=1 TO 36
1180 c4=j*10:c4=c4*3,14159/180
1190 x=c3*COS(c4)
1200 y=c3*SIN(c4)
1201 x2=x1+x:y2=y1+y
1210 LPRINT "D";x2;",";y2
1220 LPRINT CHR $(13)
1230 NEXT j
1240 RETURN
```

在 APPLE 机接口卡中固化 BASIC 应用程序

解式杰

如果 APPLE 机在特殊场合单一使用，把应用程序固化在接口卡中，既便于携带又增加了系统的可靠性。为此，下面给出一种比较简单的电路，其原理如图1：

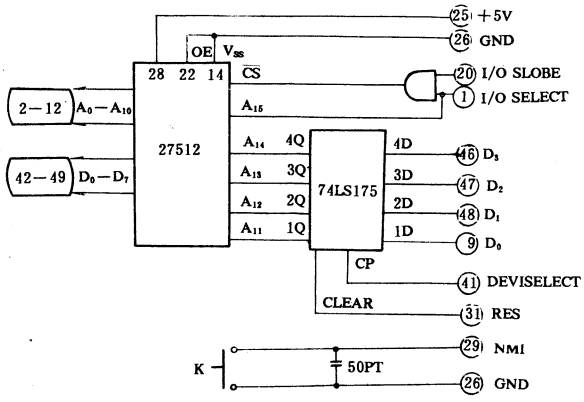


图1

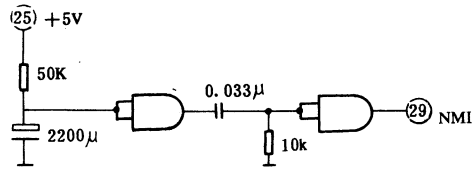


图2

用一块27512EPROM 存贮器存放管理程序和用户的 BASIC 应用程序，应用程序可达32K。74LS175四 D 触发器产生 EPROM 片选信号。非屏敝中断信号 $\overline{NMI}$ 通过按钮开关 K 控制。这样，主机启动后首先进入 BASIC 状态。若要进入用户程序，用开关 K 触发即可产生非屏敝中断信号。机器原有功能未受影响。电容 C 用作抗干扰，防止误触发。

设接口卡占用2号插槽，应把原主机监控程序 $FFFA 和 $FFFB 单元内容分别改为00和 C2，同时用短接帽短接2号插槽。这样，当用开关 K 触发 $\overline{NMI}$ 即可进入卡上管理程序，。运行管理程序的结果是把用户程序以2K 字节为单位调入内存，再给0页有关单元置初值，然后执行用户程序。

管理程序清单：

```
C200-   20 FB DA    JSR $DAFB
C203-   A9 08       LDA # $08
C205-   85 FF       STA $FF
C207-   A2 00       LDX # $00
C209-   8A          TXA
C20A-   8D A0 C0    STA $C0A0
C20D-   A0 00       LDY # $00
C20F-   98          TYA
C210-   85 3C       STA $3C
C212-   A9 01       LDA # $01
C214-   85 42       STA $42
C216-   A9 C8       LDA # $C8
C218-   85 3D       STA $3D
C21A-   A5 FF       LDA $FF
C21C-   85 43       STA $43
C21E-   A9 CF       LDA # $CF
C220-   85 3F       STA $3F
C222-   A9 FF       LDA # $FF
C224-   85 3E       STA $3E
C226-   20 2C FE    JSR $FE2C
C229-   EA          NOP
C22A-   A5 FF       LDA $FF
C22C-   18          CLC
C22D-   66 08       ADC # $08
C22F-   85 FF       STA $FF
C231-   E8          INX
C232-   C9 0F       CMP # $06
C234-   D0 D5       BNE $D5
C236-   A9 08       LDA # $08
C238-   85 68       STA $68
C23A-   A9 01       LDA # $01
C23C-   85 67       STA $67
C23E-   A9 33       LDA # $33
C240-   85 6A       STA $6A
C242-   85 6E       STA $6E
C244-   85 6C       STA $6C
C246-   A9 BC       LDA # $BC
C248-   85 69       STA $69
C24A-   85 6B       STA $6B
C24C-   85 6D       STA $6D
C24E-   85 AF       STA $AF
C250-   A9 01       LDA # $01
C252-   85 AE       STA $AE
C254-   20 66 D5    JSR $D566
C257-   60          RTS
C258-   00          BRK
```

```
C259-    00    BRK
C25A-    00    BRK
```

该管理程序的 CMP# $nn 语句中 nn 应根据用户 BASIC 程序长度在01H 至0FH 之间选择。$6A，$6E，$6C 中存放用户程序未地址高字节。$69，$6B，$6D，$AF 中存放用户程序未地址低字节，其赋值也应根据 BASIC 程序长度而定。

管理程序在 EPROM 中固化时从0200地址开始，用户程序从8000地址开始。

如果想在开机后直接进入用户程序，用图2电路可产生 $\overline{NMI}$ 信号。

设计微电脑控制电路的小常识

王　林

微型计算机在过程控制的开发应用中要进行不同程度的接口扩充、接口芯片与微型计算机总线的挂接、其 CPU 承受负载的能力都是有限的，设计者如不很好注意，就会带来许多困难，下面就 Z80—CPU 为例来说明其带负载的能力。

在计算机总线上，传送的均为“1”“0”信号，挂在总线上的各个部件对“1”“0”信号进行识别，从而完成各种操作。

以正逻辑为例，在微计算机中，“1”“0”所代表的电压范围，当逻辑1时，必须输出大于2.4V 的电压，而要输出逻辑0，则必须输出小于0.4V 的电压。在电路中，器件输出的电压除与本身的输出能力有关外，还与负载有关。Z80—CPU 输出口在输出逻辑1时，要满足逻辑1的电压值，负载电流最大不应超过250μA，或 RL≥2.4V/250μA＝9.6KΩ；而输出逻辑0时，灌入电流不能大于1.8mA，方能使输出电压不大于0.4V。

在微计算机中，与总线接口的器件有 MOS 器件，也有 TTL 器件。MOS 器件的功耗很小，在小系统中一般不会对 CPU 的负载能造成威胁，而 TTL 器件一般较 MOS 器件功耗大，但由于 TTL 器件速度快，驱动能力大，不易静电损坏，因此，在接口电路中大量应用，在总线上挂的 TTL 器件较多时，要满足负载条件，一般要加总线驱动器。

表一列出74系列中六个反相器作为输入负载和输出驱动情况下的技术条件，从而不难看出，Z80—CPU 输出口在以反相器输入端为负载时最多可带几个反相器。

CPU 输出“1”时250μA/20μA＝12.5个门

CPU 输出“0”时1.8mA/0.36mA＝5个门

因此，Z80—CPU 在以74LS04的输入端为负载时，最多可带5个门；否则就必须加总线驱动才能保证正确的逻辑输出。

			74LS04	7404	74S04
输出驱动	1	输出电压降到2.4V时能供给的最小电流	400μA	400μA	1000μA
	0	输出电压升到0.4V 时能吸入的最小电流	8mA	16mA	20mA
输入负载	1	输入入电压升到2.4V 时的最大电流	20μA	40μA	50μA
	0	输入电压降到0.4V 时的最大电流	0.36mA	1.6mA	2.0mA

对其它电路在使用中也存在以上问题，读者可查阅有关手册，以了解其使用参数。

微机通用系统板测试卡

适合8088兼容机、286兼容机和386兼容机系统板的维修测试设备，国家“七·五”重点攻关项目，获北京第2届国际博览会金奖。

·通用性强，适用于所有采用 PC/AT 总线的各种机型。

·使用方便，测试卡的使用与普通控制卡完全相同。

·无特殊的设备要求，测试时使用原微机的 CGA 或 EGV 或兼容的显示系统。

·显示的结果直观明确，可将故障测试定位到芯片/模块一级。

·资料齐全，备有详细的使用说明和维修方法介绍，并以集中方式免费培训。

·负责终生保修，并作维修方面的技术后援。

微机通用测试卡　全国统一零售价3800.00元，有资料备索。

《电子与电脑》编辑部办理邮购经销业务。

邮局汇款：北京173信箱《电子与电脑》编辑部　邮编100036

包装邮寄费另加10元。

型号不同的绘图仪绘制 TANGO 软件图形的方法

湖北黄石机械自动化研究所　黄健　陈军

TANGO 的原理图输出程度 Schematic－PLOT 可以在多种打印机、绘图仪上输出各种格式的原理图，其图形的大小可任意设置，甚至 x、y 方向的比例因子均可独立调整。本文以 SHARP 的 CE－515P 绘图仪为例，介绍与 TANGO 不兼容的绘图仪绘制 TANGO 原理图的简便方法。

该法分两步进行。

（一）首先通过 Schematic－PLOT 将欲输出的原理图文件转化为绘图指令序列文件。

（二）根据您的绘图仪编一个转换程序，将第（一）步中得到的绘图指令序列文件中的绘图指令转换成您的绘图仪的相应指令，使绘图仪绘出图形。

第（一）步

1. 运行 PLOT. EXE 文件，屏幕显示主菜单：

```
Tango—Schematic PLOT TER MAIN MENU
Actire Directory:D:\TANGO
Sheet Filename:
Sheet Size:B
Free memory:416624 Byees
Orientation:Normal Border Plot:ON
Plotter Pen:1 Pin Numbers:OFF
Communications Menu Scale Menu
Dir Load Plot
X offset:1.00 Y offset:0.50
Pen Speed:9
Quit
〉
```

2. 按 L(Load)键，装入欲转换的原理图文件。

按 O（Orientation）键，选择原理图按横向(Normal)还是按纵向(Rotated)打印绘图。

按 C(Communicotion Menu)键，进入通讯协议子菜单。

按 L(Plotten Language)键，选择到 Roland DXY－800。

按 P(Plotter Output Device)键，直到出现 Filename。

按 F(Filename)键，输入欲得到的绘图指令序列文件名。

按 Q(Quit)键，回到主菜单。

按 S(Scale Menu)键，进入比例选择子菜单。

按 S(Plotter Scale)键，指定输出图形的比例。对于 CE－515P 绘图仪我们设置为0.400:1.000。

根据菜单提示还可选择 x 方向比例因子、y 方向比例因子等，这里不再详述。

按 Q(Quit)键，回到主菜单。

按 P(Plot)键，转换开始。

当转换结束后，便得到了绘图指令序列文件。

第（二）步

编制转换程序。下列程序是针对 CE－515P 绘图仪编写的，供大家参考。

```
1 LPRINT CHR$(13)
2 LPRINT CHR$(27);"a"
3 LPRINT CHR$(13)
4 LPRINT CHR$(27);"b"
5 LPRINT CHR$(13)
6 DIM x(10),y(10)
7 x=0:y=-999
19 INPUT "File Name:";f$
20 OPEN "r",#1,f$,1:FIELD
   #1,1 AS a$
30 b=LOF(1)
31 LPRINT "M";x;",";y
32 LPRINT CHR$(13)
33 LPRINT "I"
40 g=5
50 FOR i= TO b
60 GET #1,g:g=g+1
61 c=ASC(a$)
70 IF c=77 THEN 100
80 IF c=68 THEN 150
81 IF c=13 THEN 300
82 IF c=67 THEN 400
90 LPRINT a$;
95 NEXT i
96 CLOSE:END
100 LPRINT "M";
110 GOTO 95
150 LPRINT "D";
160 GOTO 95
200 GOTO 95
300 LPRINT CHR$(13)
310 GOTO 95
400 GOSUB 1000
410 GOTO 60
1000 c2=0
1001 j=4:k=4:c5=0
1002 x(5)=200:x(6)=200:
     y(5)=200:y(6)=200
1020 GET #1,q:q=q+1:i=i+1
1030 c1=ASC(a$)
```

（下转第3页）

键盘电视游戏机的FBASIC语言及程序设计(下)

于 春

三、FBASIC语句和程序

先举一个简单的FBASIC程序的例子,看如何运用FBASIC语言解题。

1、语句与程序示例

设全班某课程的成绩,A等(100分)的10人,B等(90分)18人,C等(80分)21人,D等(70分)5人,求全班的平均成绩。

解此题可用FBASIC语言编出如下程序:

```
10 LET A=10
20 LET B=18
30 LET C=21
40 LET D=5
50 LET U=A+B+C+D
60 LET G=100*A+90*B+80*C+70*D
70 LET F=G/U
80 PRINT F
90 END
```

这是一个完整的FBASIC程序,可以看出,它是由若干行组成。一般一行写一个语句,上面9行共有9个语句。也可以一行内写多个语句,不过同一行内语句间要用冒开":"隔开。

每一个语句让计算机完成一定的操作,若干语句的有序集合就构成一个程序。

一个语句一般可分成三部分:

a、标号 每个语句前,都有一个数字(本例中10~80),这个数字叫行号或语句标号。行号是无符号整数,一般情况下,计算机按行号的大小由小到大执行各语句。行号范围0~65534,不要求连续,一般上下行间都留出一定间隔以便修改补插新的语句。各语句送入计算机不一定按行号顺序输入,送入后,计算机会自动按行号大小顺序整理排列好。

b、语句定义符 它规定计算机执行一定的操作,如上例中的LET、PRINT、END。

c、语句体 是跟在语句定义符后面的,需要执行的具体内容,如上例中的A=10等。每个语句不一定都有这三部分,但行号一定要有。一行有几句,则这几句共用一个行号。每个程序一般以END语句结束。只要后面没有别的语句,允许省略END。

2、赋值语句(LET语句)

上例中10~70行具有共同的形式:

〈行号〉LET〈变量〉=〈表达式〉

这类语句叫赋值语句。它的含义是:将等号右边的表达式先算出来,然后将其值赋给左边的变量。执行此语句之后,左边的变量就获得了一个新的确定的值。上例中10句执行后,变量A得到一个确定的值10;60句执行后,变量G得到一个确定的值4650。

说明几点:

a、语句说明符LET可以省略。前面说过,一行可以写几个语句,因此,上述例题可以简写成如下形式:

```
10 A=10:B=18:C=21:D=5
20 U=A+B+C+D:G=100*A+90*B+80*C
   +70*D
30 PRINT G/U
40 END
```

b、赋值语句中的等号与数学上的等号不同,在数学上X+Y=Z表示等号两边完全相等,也可以写成Z=X+Y。而赋值语句的Z=X+Y是指把X、Y的当时值相加后送到Z中去而不管当时Z已经是多少,此语句执行后Z得到一个新值。又如

```
100  X=X+1
```

是经常用的一个赋值语句,它表示把X的当前值加1之后送给X作为X的新值。这在代数学中显然是不成立的。因此要特别注意这一区别。

另外,赋值语句的左边只能是一个变量,不能是多个变量或表达式。如只能写Z=X+Y,决不能写成X+Y=Z。同样4*Y=20也是错误的。

c、也可以对字符串变量赋值。如

```
10 LET A$="FCS-90 COMPUTER"
```

此句是将右边字符串送入左边串变量中,此时A$获得新值。A$的新值就是一个确定的字符串。注意字符串都要带上双引号。且字符串只能赋给串变量,数值只能赋给数值变量,不能交叉赋值。

d、赋值语句是用得最多的一种语句,不但可以用

它给变量赋值，更主要的是可以用它完成各种运算。因为赋值语句等号右边的表达式可以表达各种复杂的算术运算和字符串运算，在执行此语句时，先将右边表达式的值计算出来，再赋给左边的变量，所以任何复杂的算 术运算和字符串运算都可用赋值语句来完成。如

$$X=\frac{3a}{2c+\frac{b}{5+c}}$$

只要给定了 a、b、c、值，用一个赋值语句就可以完成 X 的计算：

```
10    X=3*A/(2*C+b/(5+c))
```

3、程序的执行(RUN 指令)

上述关于程序和语句的例子，如果没有给以运行命令，电脑不执行，只是把输入的程序存在内存里。电脑内只有一个计算计划，不会产生任何结果。

使电脑执行运算的命令是 RUN 指令。当 RUN 输入后，计算才按程序的要求一步一步执行，直到结束。执行一次 RUN 后，程序并没有被破坏，可以多次执行。

4、打印语句(PRINT 语句)

打印语句的作用是在显示器上显示或在打印机上打印出变量或表达式的值或字符串。

a. 打印语句的格式

〈行号〉PRINT〈输出项〉

输出项可以包含多个变量，表达式及串变量，各量之间用逗号或分号隔开。如

```
10 A=100:B=200
20 A$="HOW ARE YOU"
30 PRINT A$,A,(A+B)/2
40 END
```

输入 RUN 语句后，结果如下：

```
HOW ARE YOU      100      150
```

b. 打印语句的输出格式

〈1〉标准格式：(分区格式)

PRINT 语句中的输出项之间如用逗号分开则输出的格式叫标准格式，或叫分区格式。

显示屏一行28个字符。根据输出项字符的多少，自动分成四个区，三个区、两个区。如果最后一区打不下，多的部分会自动移到下一行第一区。

〈2〉紧凑格式

输出项之间用分号隔开，则输出为紧凑格式。例如：

```
10 A$="GOOD"
20 B$="MORNING"
30 C$="*"
40 PRINT A$;C$;B$
50 END
RUN     屏幕显示
GOOD*MORNING
```

输出项之间不留空格，紧凑打印。若上例30句改为

```
30    C$=" "
RUN     屏幕显示
GOOD MORNING
```

这是 C$ 为空格，空格也是字符。

〈3〉单列打印

在循环语句中，若输出项后不打标点，则单列打印。如

```
10 FOR A=0 TO 5
20 PRINT A
30 NEXT
40 END
RUN 屏幕显示
0
1
2
3
4
5
```

〈4〉隔行打印

若在打印语句的上一行或下一行加入 PRINT 空语句，则输出隔一行打印。例

```
10 FOR A=0     TO     3
20 PRINT
30 PRINT A
40 NEXT
50 END
RUN
0
1
2
3
```

5、列表与清内存语句(LIST、NEW)

一个程序编写完毕或运行后，欲察看程序或进行修改可使用 LIST 语句列出程序表。

LIST 语句是一个直接指令，可不编行号，直接输入电脑执行。LIST 指令可以随意、部分或全部列出行号。其格式为

LIST〔〔m〕〔-n〕〕

中括内为任选项。如：

LIST m——列出行号为 m 的那一行程序。

LISTm-——列出 m 行后所有程序。

LIST-m——列出起始行至 m 行程序。

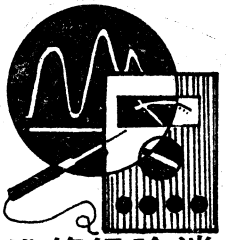
LISTm-n——列出 m 行至 n 行程序。

LIST——列出全部程序。

一个程序运行完毕，要进行新的程序设计时，必须清除原有程序(以免干扰新编程序)，即须进行清内存。清内存有多种方法，如关机后重新开机。常用的是使用 NEW 指令。

NEW 指令也是直接指令，执行后，内存中所有数据全部清除。

两条指令的使用很简单，不再举例，读者可以自己练习。



驱动器修理一例

张红平

机型：APLLE Ⅱ型计算机。

故障现象：主机、打印机、显示器均工作正常只是驱动器无法取出磁盘信号，驱动器上的指示灯亮一两秒钟就熄灭了。

故障分析：驱动器正常情况，指示灯一般要亮十几秒到几十秒（由磁盘内容决定时间长短）后熄灭，显示器显示磁盘内容并提示问题或者让用户选择。根据故障现象，驱动器亮，说明主机电源已送出，只是没有接通控制信号和读写文号，计算机误认为磁盘没有放进去，因此显示器上没有任何磁盘信息。据本人修理多台驱动器所知，大多是人为损坏，很可能是驱动器与主机的20线杆头插反造成主机电源对驱动器信号电路的损伤，驱动器接口插头各脚作用如下：上面一排从1、3、5、7脚为接地线，19、17、15、13脚为正12伏电源线，下面一排为步进电机A、B、C、D相，10、14、16、18、20脚分别为写令，驱动器启动、读数据、写数据、写保护、和负12伏电源，据此插头的分布情况，上面一排是电源，而下面一排主要为控制信号和读写信号，如果插头插反正好是上面一排电源对驱动器的下面控制电路，就会造成驱动器控制电路损坏。

检修：据以上分析，打开驱动器拆下电路板，仔细观查发现主要控制信号都由集成块74LS125进行转换，因此认为此集成块损坏的可能性是较大的，小心取下此集成块，测试发现区别较大，最后花两元钱换掉此集成块，试机，一切正常，此机修复。

注：驱动器内部电路大多为双面电路，拆卸时最好用两个烙铁两人同时进行，望同行们在遇到类似故障时多加分析故障产生的原因，少走弯路。

维修小经验

张中弦

钟控收音机的时钟集成块失控故障的检修

有一钟控收音机其时钟集成块是MM5387AA/N，故障是不能定时收音、响闹，当设在睡眠功能时，按住SLEEP（睡眠）键才能收音，一放手就立即自动关机。检测集成块各脚电压，发现各控制输入脚均接近0V，唯有打盹输入脚（第24脚）接近电源正极电压9V，检查打盹按键（SNOOZE）并无短路。原来故障出自集成块内部软击穿，导致打盹输入常通呈高电平，这相当于一直按住打盹按键，使定时收音、闹响及睡眠功能失灵。后经试验用一只150Ω的电阻将打盹输入脚与电源负极相连，使其电平拉低，电路即可恢复正常工作。时钟集成块一般若损坏设置输入部分，可按此法排除，只要适当选取电阻的阻值就可，理论上说阻值越小越好，但阻值过小当按下按键时会使电源电压下降太大。若时钟集成电路块是完全击穿的话则只有更换集成块了。

PC8300微电脑读取程序特殊故障的排除

一台PC8300微电脑正常使用时发现键入“LOAD”及程序名，按回车键后，显示器一片空白，插放程序磁带也没有显示及不能读取程序，按SHIFT－BREAK不能暂停读取程序过程，只能按ENTER-RESET键或断电重新启动。检查外围元件并无损坏，经测量发现接口芯片C4005的TPEIN端（第2脚）工作时的电平偏高，估计为C4005内部软击穿导致电平升高所致。后经试验将一只2.2KΩ的电阻接在C4005的第2脚与电源负极之间，故障即消除，使得一块价值五十多元的C4005接口芯片恢复正常工作。

硬磁盘驱动器专用接口电路 501262 的测试

卜建辉

501262 适用于 ST506/412 标准工业接口，使用在美国 MICROSCIENCE 公司的产品 HH-725，HH-825，HH-830，HH-1050 及 HH－4050 等系列硬磁盘驱动器中，由于美国 MICROSCIENCE 公司的硬磁盘驱动器在国内有较多的用户，通过该篇文章，能给用户及硬磁盘驱动器维修人员提供帮助。

该集成电路产生索引，在索引脉冲的同步下能产生伺服区和数据区，同时产生准备好信号寻道完成信号，读写控制，时钟分频，步进脉冲缓冲作用。

1. 引脚功能及定义（见表 1）

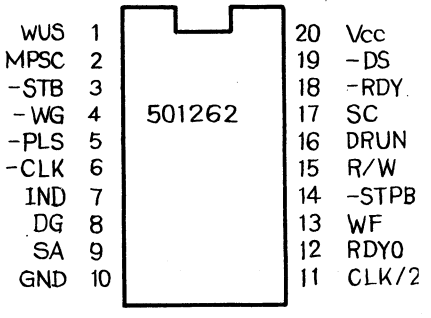


图 1

表 1

WUS	写不安全信号	输入
MPSC	CPU 控制寻道完成	输入
－STB	步进脉冲	输入
-WG	写门	输入
-PLS	索引脉冲	输入
-CLK	时钟	输入
IND	索引	输出
DG	数据区同步信号	输出
SA	伺服区同步信号	输出
GND	地	
CLK/2	时钟二分频	输出
RDYO	准备好信号输出	输出
WF	写故障	输出
-STPB	步进脉冲缓冲输出	输出
R/W	读/写	输出
DRUN	数据运行	输出
SC	导道完成	输出
-RDY	准备号	输出
-DS	驱动器选中	输入
Vcc	+5V	输入

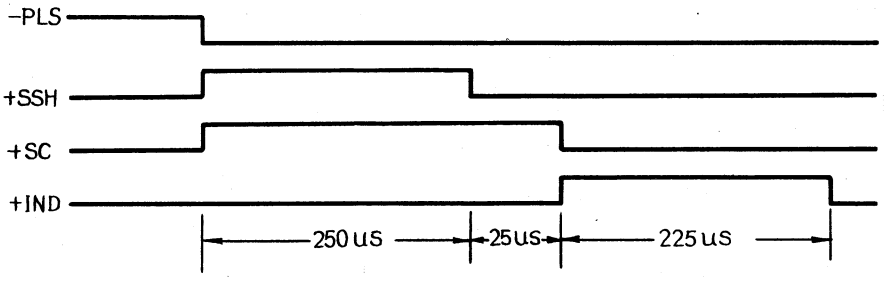


图 2

2. 逻辑关系

(1)＋RDYO＝(＋DS)＊(＋RDY)＊(－WF)

(2)－STPB＝(－STP)脉冲下降沿的一个时钟脉冲

(3)＋SC＝〔＋MPSC)＊(STPNG)＊DS

在这里(＋STP)＊(＋RDS)＝1 时　　STPNG＝1

MPSC＝1 时　　STPNG＝0

(4)＋R/-W＝-((＋WG)＊(＋DS)＊(－WFW)＊(-DG)＊RDY＊(-STPNG)＊(-MPSC))

(5)＋SA＝SSH＋(－RDY)＋STPNG

在这里 SSH　见图 2

(6)＋DG　见图 2

(7)＋IND　见图 2

(8)-CLK/2　时钟二分频

3. 该集成电路的测试及方法

(1)该专用集成电路的测试原理图，见图 3。

(2)测试过程

a. 将专用集成电路插入测试仪中的 IC 插座

b. 开电源

c. 按 START 开关，开始测试.

e. 测试完后，指示灯亮及显示相应的代码，绿灯表示通过，红灯表示失败.

(3)测试程序

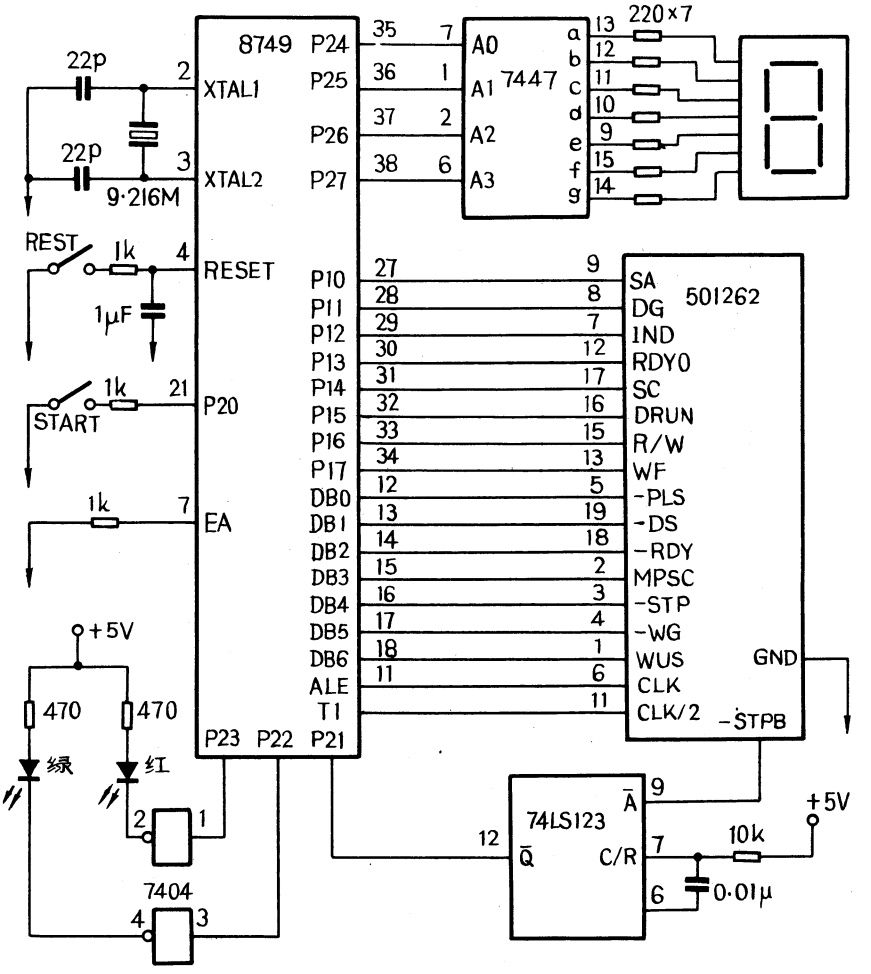


图 3

```
        ORG 00H
        JMP RESET
        ORG 0AH
RESET:  CLR A
        OUTL BUS,A
        CPL A
        OUTL P1,A
        MOV A,03H
        OUTL P2,A
        DIS TCNTI
        DIS I
        SEL RB0
        SEL MB0
        ORL P2,F0H
START:  IN A,P2
        ANL A,01H
        JNZ START
        MOV R2,20H
        CALL DELAY2
        IN A,P2
        ANL A,01H
        JNZ START
        ANL P2,F3H
        MOV R6,00H
        MOV R5,10H
PULSE:  MOV A,FBH
        OUTL BUS,A
        MOV R2,20H
        CALL DELAY2
        ANL BUS,FEH
        MOV R3,00H
        MOV R4,00H
        IN A,P1
        ANL A,07H
        XCH A,R3
        MOV R1,2DH
        CALL DELAY1
        IN A,P1
        IN A,67H
        XCH A,R4
        NOP
IND:    IN A,P1
```

3.4 程序流程框图

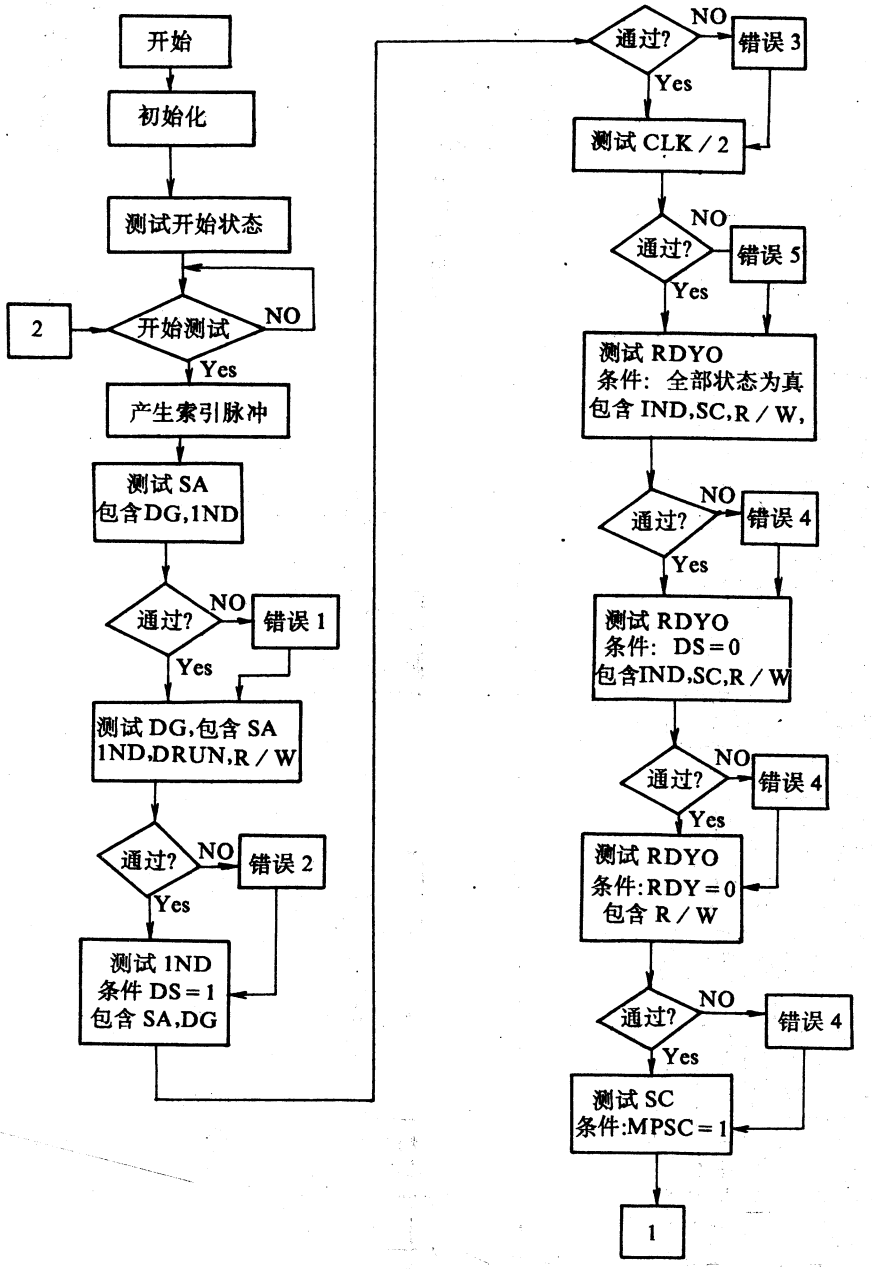


图 4

```
        ANL A,07H
        JZ IND1
        CALL ERROR
        ANL P2,3FH
IND1:   ANL BUS,FDH
        IN A,P1
        ANL A,07H
        ANL A,04H
        JZ IND2
        CALL ERROR
        ANL P2,3FH
IND2:   MOV R1,23H
        CALL DELAY1
        IN A,P1
        ANL A,07H
        JZ TST1
        CALL ERROR
        ANL P2,3FH
TST1:   XCH A,R3
        XRL A,03H
        JZ TST2
        CALL ERROR
        ANL P2,1FH
TST2:   XCH A,R4
        XRL A,42H
        JZ PCNT1
        CALL ERROR
```

```
        ORL P2,F0H
        ANL P2,9FH
PCNT1:  MOV A,FBH
        OUTL BUS,A
        DJNZ R5,PULSE
CLK2:   MOV A,D0H
        MOV T,A
        STRT CNT
        MOV R1,5DH
        CALL DELAY1
        STOP TCNT
        JTF RDYO
        CALL ERROR
        ANL P2,5FH
RDYO:   MOV A,79H
        OUTL BUS,A
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,08H
        JNZ RD1
        CALL ERROR
        ANL P2,4FH
RD1:    MOV A,7BH
        OUTL BUS,A
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,5CH
        XRL A,40H
        JNZ RD2
        CALL ERROR
        ANL P2,4FH
RD2:    MOV A,7DH
        OUTL BUS,A
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,48H
        XRL A,40H
        JZ SCT
        CALL ERROR
        ANL P2,4FH
SCT:    MOVA,79H
        OUTL BUS,A
```

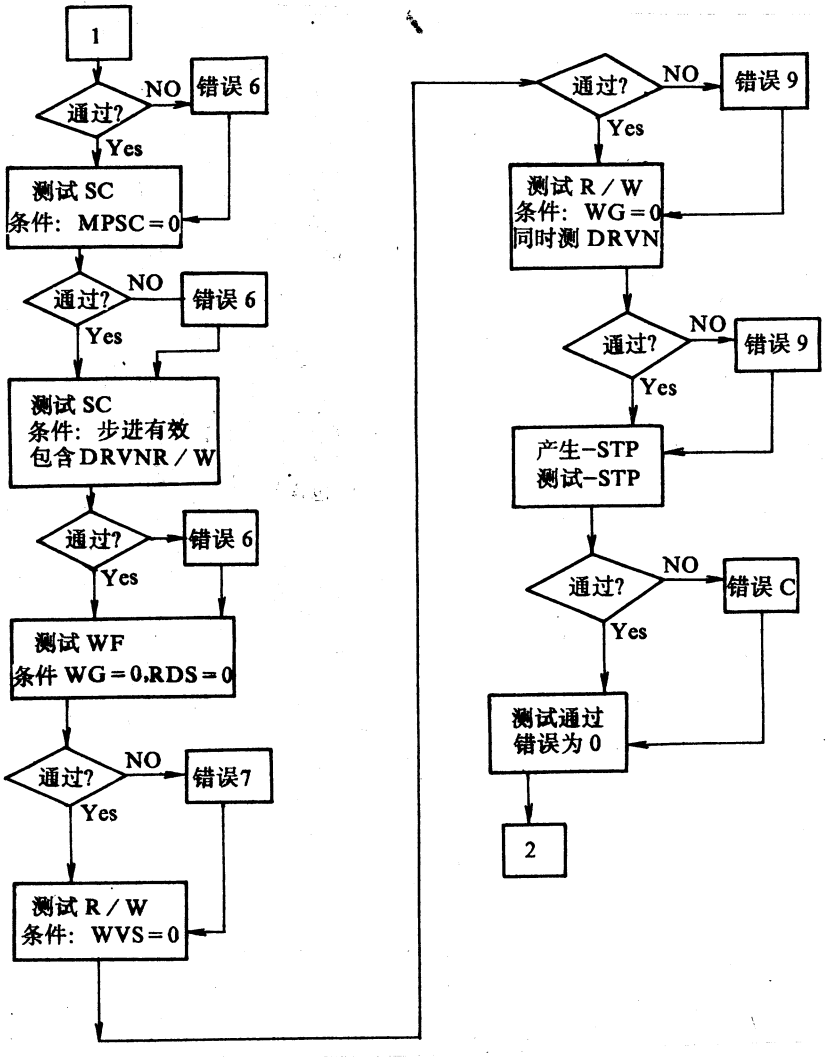


图 5

```
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,10H
        JZ SC1
        CALL ERROR
        ANL P2,6FH
SC1:    MOV R2,60H
        CALL DELAY2
        MOV A,71H
        OUTL BUS,A
        IN A,P1
        ANL A,10H
        JNZ SC2
        CALL ERROR
        ANL P2,6FH
SC2:    MOV A,73H
        OUTL BUS,A
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,30H
        JZ SC3
        CALL ERROR
        ANL P2,6FH
SC3:    ANL BUS,65H
        MOV R1,50H
        CALL DELAY1
        MOV A,71H
        OUTL BUS,A
        IN A,P1
        ANL 70H
        XRL A,40H
        JZ WFT
        CALL ERROR
        ANL P2,6FH
WFT:    ANL BUS,5FH
        IN A,P1
        ANL A,80H
        JNZ SC4
        CALL ERROR
        ANL P2,7FH
SC4:    MOV A,79H
        OUTL BUS,A
        MOV R1,50H
        CALL DELAY1
        ANL BUS,77H
        IN A,P1
        ANL A,10H
        JNX WRT
        CALL ERROR
        ANL P2,6FH
WRT:    ANL BUS,BFH
        IN A,P1
        ANL A,40H
        JNZ WR1
        CALL ERROR
        ANL P2,9FH
WR1:    ANL BUS,DFH
        IN A,P1
        ANL A,60H
        JZ TSTP
        CALL ERROR
        ANL P2,9FH
TSTP:   MOV R5.10H
STP1:   MOV A,F1H
        OUTL BUS,A
        MOV R2,90H
        CALL DELAY2
        IN A,P2
        ANL A,02H
        JNZ STP2
        CALL ERROR
        ANL P2,AFH
STP2:   ANL BUS E1H
        IN A,P2
        ANL A,02H
        JZ STP3
        CALL ERROR
        ANL P2,AFH
STP3:   DJNZ R5,STP1
GOOD:   MOV A,R6
        XRL A,AAH
        JZ AIN
        ANL P2 0FH
        ORL P2,04H
AIN:    JMP START
```

延迟子程序 1

```
DELAY1:DJNZ R1,DELAY1
RET
```

延迟子程序 2

```
DELAY2:MOV R1,FFH
DLY:    DJNZ R1,DLY
        NOP
        DJNZ R2,DELAY2
        RET
```

错误子程序

```
ERROR:  ORL P2,08H
        ORL P2,F0H
        MOV R6,AAH
        RET
```

电子工业出版社计算机类新书介绍

代号：	书　　名	邮购价
B1580	Microsoft Windows V3.0 用户手册	16.10
B1581	Microsoft Windows V3.0 程序设计指南和工具	24.20
B1582	Microsoft Windows V3.0 程序员参考手册	30.50
B1530	计算机环境的可移植操作系统界面 POSLX.1	23.00
B0890	DEC NET 计算机网络的结构与实践	46.00
B1671	现代计算机技术博览(一)	28.80
B1570	UNIX 使用大全	14.00
B1660	UNIX 系统 V 实用指南	12.70

邮局汇款：北京万寿路 173 信箱电子工业出版社发行部邮购科。

银行汇款：

户　　名：电子工业出版社销售服务部。

开 户 行：北京市翠微路分理处。

帐　　号：661036—40　　邮政编码　100036

汇款同时请用正楷字写明书代号、册数。

IBM-PC/XT 一种系统错误的分析

姜金友

在 IBM-PC/XT 及其兼容机上，即使系统有两个 5 英寸软盘驱动器，DIR B:时偶尔也会出现以下信息：Insert diskette for drive B: and press (or strike) any key when ready，其意是在 B 驱动器内插入软盘后再任敲一键，按键后发现 DIR B:执行的就是 DIR A:，以后每次 DIR B:都是对 A 驱动器进行操作。若再执行 DIR A:，又会出现以上提示，只是 B:变成了 A:，让你在 A 驱动器内插入软盘。一旦出现上述情况，以后所有对软盘的文件操作都将会有类似现象发生。这些表现同只有一个 5 英寸软盘驱动器的情况相同。也就是说系统此时认为你的微机只有一个 5 英寸软盘驱动器。发生上述情况后，只要重新起动，一般就不会再出现以上现象。

究其原因，起初还以为是病毒所至，经检查后发现系统并没有染毒。再一分析，发现用 INT 13H 对软盘扇区进行访问，一切正常。因为 INT 13H 是调用 BIOS 对软盘进行访问的，故此可以肯定以上现象只与 DOS 系统有关。

下面再看看 DOS 系统是如何引导的？

开机后，系统首先执行 0 道 0 面第一扇区的引导模块，该模块将 IO. SYS(或 IBMBIO. COM)读入内存，IO. SYS 由以下三个部分组成：

1. 初始化程序；2、BIOS 接口程序；3、其他服务程序。

其中初始化程序有一项就是确定软盘驱动器数，该数目是由系统配置开关 DIP 所决定的。主机通过 8255A 可编程外围接口芯片读取 DIP 开关的设置，该开关的第七、八两位表示系统连接的 5 英寸软盘驱动器数，其组合如下：

SW-8	SW-7	软盘驱动器数
ON	ON	1
ON	OFF	2
OFF	ON	3
OFF	OFF	4

在 IBM PC/XT 中，8255A 工作于无联络信号的基本输入输出方式(方式 0)，其 24 根外围数据线分成三组：PA0-PA7、PB0-PB7、PC0-PC7，BIOS 中使用的 I/O 地址分别为：60H、61H、62H。PC0－PC7 工作于输入状态，用来读取 DIP 开关的设置。从 62H 口读入数据的第二、三位分别表示 SW－7 与 SW－8 的状态。IO. SYS 就是通过从口 62H 读数据而获得软盘驱动器数的，并将其结果存入内存。经多次查找分析发现，结果存放在内存 0 段某一单元内，该单元的偏移量随 DOS 版本不同而不同。下面是几种不同版本的偏移量：

DOS 版本	偏移量
2.00	0acfH
2.10	0ae0H
3.10	0b2cH
3.20	1abfH
3.30	0bd8H
4.01	0becH

对于 DOS3. 20 以下版本，如果偏移量处数值为 01H，则表示系统只有一个 5 英寸软盘驱动器；如果为 00H，则表示系统有二个 5 英寸软盘驱动器；DOS3. 20 (含 3. 20)以上版本则正好与上面相反，01H 表示系统有二个 5 英寸软盘驱动器，00H 表示系统只有一个 5 英寸软盘驱动器。因此只要改变以上偏移量处的值，系统所识别的驱动器数目就要发生变化，例如对于 DOS4. 01 系统，只要完成以下操作(键入划线部分即可)，系统就认为只有一个软盘驱动器。此时 DIR A:与 DIR B:结果完全相同。

```
C>DEBUG
-e0:bec 00
-q
```

导致本文开头所说情况的原因就是因为基本输入输出模块读取 DIP 开关错误所至，因此如果你的计算机是 PC/XT 型，且带有双软盘驱动器，只要在 DOS 系统引导后，检查该版本所对应偏移量处的值是否为双驱动器所对应的值，若不是就改过，这样即可避免再发生上述错误。

对于 PC/AT 等系列微机，尚未发现上述现象，但若将以上偏移量处的值改为单驱动器所对应的值，也会有上述现象发生。

用非格式化方法修复硬盘一例

西安石油学院　周建辉

我室一台 IBM－PC/XT 微机开机出现死锁现象，用软盘启动也不能联上硬盘，判定为硬盘主引导记录(主 BOOT)坏。因机上有大量科研成果资料，我们决定用非格式化方法来修复此硬盘。

通过分析 XT 机、AT 机和 386 机的主引导记录，发现它只起到一个 ROM－BIOS 和引导 DOS(BOOT)之间过渡作用，主要功能是找到合法硬盘的活动分区并把控制权交给该分区的 BOOT 以引导 DOS。所以主 BOOT 都大同小异，特别是同一容量硬盘在常规使用下更是相似甚致相同。因此我们的做法是利用 DEBUG 将好盘主 BOOT 代替同容量坏盘主 BOOT，具体操作如下：

第一步：拷贝好的主 BOOT 到软盘，在 DEBUG 下输入下述小程序将好的主 BOOT 读入内存××××：0400；再用命令"—W400　|　12　|"将好主 BOOT 写入 B 驱动盘上的软盘。如图一所示。

```
A)debug
—a100
21A2:0100 mov dx,0080
21A2:0103 mov cx,0001
21A2:0106 mov bx,0400
21A2:0109 mov ax,0209
21A2:010C int 13
21A2:010E
—g=100 10e
AX=0000 BX=0400 CX=0001 DX=0080 SP=
FFEE BP=0000 SI=0000 DI=0000 DS=21A2 ES=21A2
SS=21A2 CS=21A2 IP=010E NV UP EI PL NZ NA PO
NC 21A2:010E 47          INC     DI
-
-
-w400 1 12 1
-
```

图一　拷贝好的主 BOOT 到软盘

第二步：将好的主 BOOT 写入坏盘。用软盘启动已坏的机器，在 DEBUG 下用命令"——L400　1　12　1"将 B：驱动器上软盘中的好主 BOOT 读入内存××××：0400 位置；再用下述小程序将好的主 BOOT 写入坏盘，即完成修复工作。如图二所示。

我们用上述办法恢复了一个 20M 硬盘，其上的所有数据完好。事实证明，只要对 PC 机引导机制及主 BOOT、BOOT 工作过程进行分析，就可以许多情况下无须通过格式化就可修复硬盘。

```
A)debug
-
-1400 1 12 1
-
-
-a100
21A2:0100 mov dx,0080
21A2:0103 mov cx,0001
21A2:0106 mov bx,0400
21A2:0109 mov ax,0301
21A2:010C int 13
21A2:010E
-g=100 10e

AX=0000 BX=0400 CX=0001 DX=0080 SP=
FFEE BP=0000 SI=0000 DI=0000 DS=21A2 ES=21A2
SS=21A2 CS=21A2 IP=010E NV UP EI PL NZ NA PO
NC 21A2:010E 47          INC     DI
-
```

图二　将好的主 BOOT 写入坏硬盘

上面操作是在双软驱情况下，但此方法也可在单软驱情况下进行。另外分区表信息可以 DOS 版本和硬盘容量进行重构；但分区表信息还与硬盘使用情况(如是否分区，怎样分区)有关，因而作一个主 BOOT 备份是有益的。

检修 IBM AS/400 B35 故障二例

王志远

(一)IBM 5337 激光打印机故障检修

故障：打印出来的字体拉丝

原因：在感光鼓上残留有纸片，未被卡纸传感器检出。

分析：IBM 5337 激光打印机是一种电子照相式打印机。它的工作原理是：用充电器使 OPC 鼓表面带电，被调制的激光束打在鼓表面，使鼓表面电位发生变化，产生不可见的潜象。显象部件的墨粉被吸入鼓表面被激光照射的部分，从而在鼓上产生映象。转写器将鼓上的墨粉吸到打印纸上，经喷光定影，将映象加热并熔化于纸上。当鼓上残留有纸片，而未被卡纸传感器检出，打印机不产生错误信息。而残留的纸片象一把刷子碰触在转写后的打印纸上，因为此时打印出来的字体还

(下转第 29 页)

自己学修计算机

陈　勇

随着近几年来微型计算机的不断普及，微型计算机的维修工作日趋变得重要起来。微机一旦出现故障，往往会耽误工作，如果请人维修还会花费大量金钱和时间，如果我们微机使用者本人能够学习一些微机简单故障的修理知识，那么不但能够丰富自己的计算机知识而且还能节约大量请人修机所需的时间和金钱。本文仅以 IBM PC 机为例作一介绍，并愿与计算机同行们共同切磋探讨。

一、IBM PC 配备的两种故障辅助诊断功能

每一台 IBM PC 机都配有两种故障辅助诊断功能，以帮助维修人员确定引起故障的可能原因，第一种辅助诊断功能是开机后的自检过程，这一过程叫作开机自检(POST)。第二种辅助诊断功能是一种诊断软件，一般与随机配置的操作手册说明书放在一起，较之第一种辅助诊断功能，这一诊断软件能对计算机的各种系统、电路板和外设进行更广泛的测试。

两种辅助诊断功能的不同之处就在于：两者的测试范围不同。后者比前者测试的范围更广一些。IBM 公司没有为用户提供充分的有关如何使用这两种辅助诊断功能的资料，也没有提供一些简单的错误信息，即，没有像“你的随机存/取存储器坏了”这样简单明了的信息，而只有用错误代码表示的信息，比如，“03400201”，这看起来很不直观，但是，一旦我们查清代码的含义，那么我们就不但能够知道出的是什么故障，还能知道故障的确切位置。另外，由于开机自检所产生的错误代码与诊断软件所产生的错误代码是一样的，所以只要我们查清一种代码含义，就可以知道另一种的含义。

利用这两种检验手段，在配合以其他一些检验手段和识别故障的方法，我们就可以检验出所有微机故障的百分之九十五。

·利用开机自检功能(POST)维修计算机

每次开机后，微机的开机自检功能首先对内部的主要系统进行一次快速检查，所需时间依 RAM 容量的多少从几秒钟至三分钟不等，在接下来的期间内，POST 对系统主板、RAM、电源、键盘、外部机架和一些适配器插卡进行检验。在此期间，如果 POST 没有发现故障，那么微机屏幕上将不会有任何显示，同时计算机也不会有任何反映，如果，POST 发现了错误，它将用故障信息代码以及可看到的信息向你发出信号。

每当我们打开计算机，总会听到微机喇叭发出几声“嘟嘟”声，以及软硬磁盘动作时发出的声音，这些声音都是在开机后装入自检程序时发出的，而且，这些声音是有规律的。但往往我们并不在意这些声音，岂不知这些声音对帮助我们发现问题是很有用的。我们都可能有过这样的经历：打开计算机后，开始我们的工作之前，我们发现微机发出的声音与往常有所不同，但由于计算机仍然能正常引导启动，所以，这些异常的声音并未引起我们足够的重视，于是我们就继续下一步的工作了，然而，当过了一段时间，微机出现死锁或我们编写的程序不能正常存盘时，我们才发现微机出现了故障，并认识到开机时那些声音的重要性。对于 IBM PC 机来说，如果开机后一切工作正常，那么我们会看到微机有以下一系列的反映：光标闪烁几秒钟，至到 POST 结束为止。接下来，会看到 A 盘驱动器的指示灯亮起来，并可听到一声短促的“嘟”声，随之 A 盘驱动器的指示灯一闪之后就又熄灭了。当 A 盘驱动器的指示灯再次亮起来之后，微机开始装入引导程序。如果，有任何不同于上述情况的现象发生，则说明微机发生了故障。

最初，IBM PC 本想为微机配备盒式磁带机构，用于装入和存储程序，虽又立即又放弃了这种打算，不过微机内仍保留了磁带机接口，由一种长驻内存的 ROM BASIC 来管理磁带机。有时当你试图装入一个程序时，你会发现你所面对的确是 BASIC 的画面，这说明 ROM 已经抢先把磁带 BASIC 引导入内存。

当我们所要装入的程序能正常装入时，我们会听到软盘驱动器发出声音，否则，微机会自动引导磁盘 BASIC。

表 1 列出了 POST 所有的错误代码和错误提示，其中 X 代表一位数字，比如，屏幕上并不会出现“XXXX201”这样的内容，而只会出现象“0340201”这样的信息，“0340201”这一代码的后三位数字“201”代表 RAM 有故障，而这一代码的前四位数字则代表前排右起第二个 RAM 模块出现了错误。

表 1　POST 的故障信号和代码表

信号和代码	故障所在位置
无任何现象(nothing happens)	电源或其他
连续的“嘟嘟”声	电源
重复发出短“嘟”声	电源
一声长“嘟”，一声短“嘟”	系统板
一声长“嘟”，二声短“嘟”	监视器
无显示	监视器
磁带 BASIC 显示	磁盘驱动(常常是 A)
101,131	系统板
201	RAM
XXXX201 和同位核对 X	RAM
同位核对 X	电源
301,XX301	键盘
601	软盘驱动器
1701	硬盘驱动器
1801	扩展槽

值得一提的是，一些提示严格地讲并不是 POST 的一部分，比如，“nothing happens”这一既不是错误代

码，也不是磁盘 BASIC 的有关显示，但它的出现表明微机有故障存在。有些故障信息是以声音的形式表示的，有时伴随声音的出现也会有错误代码。

当微机提示出错误信息或代码后，就要用诊断盘进一步检查，在微机尚未配备诊断程序的情况下，我们可以对微机进行一些简单的检查，比如，当我们接通计算机电源的时候，计算机没有任何反映，很可能是计算机并未加电，造成这种故障的原因可能在以下几个地方：墙上的电源插座坏了；导线可能断了；微机内部电源可能出了故障；微机内某个地方短路引起电源断电。

·利用诊断盘检修计算机

我们往往只在微机出现故障时才使用诊断盘，这是不对的，我们应当经常使用诊断盘对微机进行检查，以便极早消除一些尚未造成大错的隐患。诊断盘能对微机进行更全面的检验，它还能进行多功能测试，这一点是非常重要的，因为 POST 或单项检验无法对一些模块工作时好时坏这种故障进行检验，而使用诊断盘的多功能测试，便会迎刃而解。

要使用诊断盘，首先把诊断盘插入 A 驱动器，然后从新启动微机(同时按下 Alt，Ctrl，Del 三个键)；或关闭计算机等候 5 秒钟后从新打开计算机，随后便是前面讲过的微机正常起动所发出的有规律的声音和喇叭发出的“嘟嘟”声。在 POST 运行完毕后，驱动器 A 的指示灯会亮起来，扬声器也会发出一声短“嘟”声，随后是驱动器装入程序读磁盘时发出的声音。

在使用诊断盘时，我们可以把故障存入磁盘，当然这是以你的微机和软盘驱动器能正常工作为前提条件的。

一般我们是把错误信息在打印机上输出。

当执行诊断程序时，诊断盘首先检验软盘驱动器，当要执行诊断程序时，微机会停下来并且会发出一声长“嘟”和一声短“嘟”，此时，你应当用一张存有无用信息的软盘代替诊断盘，这是因为执行诊断程序时会破坏盘上的信息。

表 2 列出了诊断盘所产生的初始错误代码、信号、提示信息。与诊断盘所产生的同类内容相比，诊断盘能产生某一部分系统或电路工作正常的信息，而 POST 确不能，这些表示工作正常的代码都是以 0 结束(比如系统板工作正常的代码是 100，软盘驱动器工作正常的代码是 600)

当你的微机监视器出现故障时，你可以通过诊断盘产生的代表不同错误信息的声音对微机进行检验，在有关操作手册上给有测试和屏幕顺序的信息(见表 3)，但没有与各种声音相对应的错误信息，所以你应当先运行诊断程序，并记录下相关信号的出现和意义。

二、找出功能故障

找出功能故障并不象你所想象的那么难，只需比消除一个故障多知道一些信息。引起某一功能故障的原因常常有多种可能。

消除一个故障往往需由简单到复杂一步一步地进行检查。比如，你的微机完全不能工作了，你就要对微机内外进行检查，首先你要检查墙上的电源插座的电源是否正常，再检查微机电源是否与插座接触良好。然

表 2　诊断程序代码

代码	故障位置
02X	电源
1XX	系统板
20X	RAM
XXXX	RAM
XX20X	RAM
30X	键盘
XX30X	键盘
4XX	单色显示器
5XX	彩色显示器
6XX	磁盘驱动器
7XX	8087 算术协处理器
9XX	打印机适配器
11X	同步通信
12X	同步通信
13XX	游戏适配器
14XX	打印机
15XX	同步数据链路控制通信
17XX	硬盘驱动器
18XX	扩展槽
20XX	二进制同步通信适配器
21XX	二进制同步通信适配器

表 3　诊断盘在以声音形式进行提示情况下的运行过程

1. 微机加电；在 POST 期间光标闪烁；短的“嘟”声；装入诊断程序

2. “对选择项进行选择”屏幕(0——运行诊断程序；1——格式化磁盘；2——拷贝磁盘；9——退回到系统盘)；键入 0 运行诊断程序

3. “配制设备”屏幕；“嘟”声；键入“Y”

4. “诊断功能选择”屏幕(0——运行一次诊断程序；1——运行诊断程序多次；2——使用记录功能；9——退出诊断程序)；键入“0”或“1”；程序开始运行

5. “键盘”(键盘测试)屏幕；按回车键；短“嘟”声

6. “显示属性”屏幕；键入“Y”；短“嘟”声

7. “设置字符显示方式”屏幕；键入“Y”；短“嘟”声

8. “80×25”显示；键入“Y”；短“嘟”声

9. “40×25”显示；键入“Y”；短“嘟”声

10. “320×200 图形显示方式”；键入“Y”；短“嘟”声

11. “640×200 图形显示方式”；键入“Y”；按任意键 8 次，短“嘟”声；键入“Y”

12. 一声长“嘟”，一声短“嘟”之后将要测试磁盘驱动器，把一存有无用信息的软盘插入 A 驱动器；按回车键；驱动器 A 的指示灯亮，驱动器也发出声音；两声“嘟”声；键入“Y”

13. 对驱动器 B 重复 12 中的过程；不要在意 RAM 驱动的结果

14. 稍等一会儿，一声短“嘟”表示诊断程序的结束

后，把微机外的设备都去掉，比如象电源接线板之类，而把微机直接与墙上的电源插座相连，这样可以检验故障是否出在微机以外设备上，(微机内部的开关大部分是自检的，风扇能转动，说明微机已经加上了电，否则，可能是电源或开关出了故障)，接下来，对微机内部电源和电路进行检验，你可以使用电压表对微机电源进行测试，也可以按下列部骤进行：先把与微机电源相连的所有器件和电路都断开，看此时，电源是否定有电压，如果有，则说明是某一与电源相连的器件出了故障，我们再每次一个地把断开的器件与电源连接起来，当电源与某一器件连接后，出现故障，则说明此器件出了故障。需要注意的是，每次断开器件或连接器件时，都要把电源关闭。

上述检验故障的思想方法是：先对故障原因做出两种可能的推测，然后消除其中之一，则证明是另一种可能原因造成的。这种检验故障的思想方法也适用于其他一些故障检验，比如，某一程序不能正常装入，引起这种故障的可能有两种：或是磁盘驱动器出了故障，或是程序的故障；若是磁盘驱动器的故障，则引起磁盘驱动器故障的原因又有两种：或是驱动器本身，或是微机内部的电路板。把装有程序的软盘放在另一个软盘驱动器 B 上，如果，装入成功了，就说明是 A 驱动器出了问题，否则，说明与驱动器 A 相连的插卡或系统电路板出了故障。

三、引起计算机故障的其他一些原因

有时对计算机不正确的操作以及对所用软件使用不当，也会造成计算机功能故障。

比如，某位操作员把一块多功能电路板插入计算机插槽中，当他试图使用它时，计算机出现死锁现象，而当他把此插卡拿掉时，计算机仍然不能恢复正常工作。这是为什么呢？原来，当他插入新的插卡时，实际的通信接口变为三个，而计算机只能管理两个通信接口，这就使计算机发生混乱。另外，由于他插入新的插卡改变了系统板上的开关设置，即使去掉了插卡，设置仍指向去掉的插卡上的 RAM。所以微机仍然不能运行。

总之，只要一步一步仔细检查，对于大部分计算机故障我们自己是完全可以修理的。

TEC－I 机磁盘驱动器 I/O 接口板故障检修一例

崔时珍　玄勇活

笔者最近检修了一台 TEC－I 机(苹果机)的磁盘驱动器的 I/O 接口板，收集了一点经验，现介绍如下，愿与同行交流。

故障现象：

本机在不连接磁盘驱动器的 I/O 接口板时，能进入固化 BASIC 状态。但是，连接之后，屏幕上显示长方形大白块，不能启动磁盘驱动器。

故障分析及故障排除：

磁盘驱动器的 I/O 接口板是由并串行转换电路，多路器，马达控制电路，磁盘驱动器选择电路等组成。并串行转换电路，一方面，负责把磁盘驱动器中读出的串行数据转换成并行数据，再送入主机；或者，把主机送来的并行数据转换成串行数据，再写入磁盘驱动器。另一方面，根据从磁盘驱动器中读出的状态信息或从主机中送来的控制信息，向多路器送工作控制信息。多路器依据工作状态，把主机送来的启动马达和选择磁盘驱动器的信息，再送到马达控制电路，启动马达，同时对磁盘驱动器进行选择。

为此，笔者首先检查了磁盘驱动器选择电路的两个输出端 ENBL1 和 ENBL2(74LS132 又输入四与非斯密脱触发器的 6 脚和 8 脚)，发现两个输出端都为高电平，所以，两台驱动器都不能启动。正常时，6 脚首先为低电平，接通 1 号驱动器。经过对该集成电路逻辑分析之后，确定未坏。为此，顺着此端向上找，发现多路(74LS259 带锁存器的三位译码/选择器)的输出端 9 脚为低电平，工作控制输入端 14 脚为高电平，都不正常。正常读写时，14 脚应有图①所示的脉冲信号，输出端 9 脚应有高电平。

这时，有几种可能性，或者多路器损坏，或者前级某元件损坏，或者包括多路器，前级元件都损坏。为先排除第一种可能性，拨掉了多路器之后，检查了 74LS323(八位双向通用移位寄存器)控制输出 2 脚，发现仍然是高电平。再检查其它管脚，发现 6 脚输入/输出端波形也失常，脉冲幅度只有 1V 左右，再拨掉该集成电路之后，测 6 脚管座的脉冲是正常的。由此可以确定 74LS323 已损坏。更换之后再开机，故障消除了。

特别注意，绝对不允许开机的情况下，插拨接口板或集成电路，否则，会损坏元件。

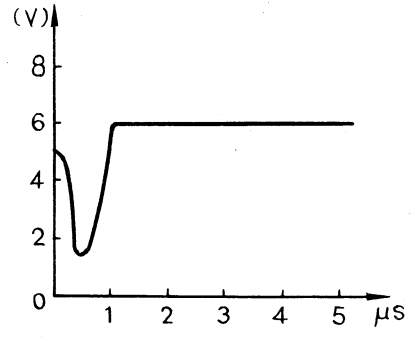


图 1

3070打印机断针定位方法及8088汇编程序

龙兵生

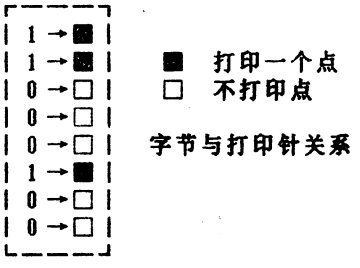
本文针对紫金3070 24针汉字打印机断针问题，介绍一种用来测试断针位置的简易方法，并给出一个使用8088汇编语言编制的测试软件。该软件只要修改一个字节，即可用来测试其它型号的打印机断针定位问题，故具有较强的通用性。

计算机外部设备之一的打印机，由于打印针具有一定的使用寿命，用户使用一段时间以后，或多或少都会遇到打印机发生断针现象（从打印输出的图文信息即可确认）影响打印输出字符的美观。为迅速进行断针的更换，需确定断针的准确位置之所在，这虽然可通过人的肉眼对打印信息和打印头部位进行观察得出答案，但准确性较差且需反复校验，费时费力。为此，笔者根据自己在工作中碰到的这类问题及解决方法，以紫金3070 24针汉字打印机断针为例，介绍一个简易的测试方法和程序，断针测试的准确率达100%且一次完成。其它类型的打印机断针定位问题，只需将本文提供的程序作一字节改动，即可用来进行断针测试。

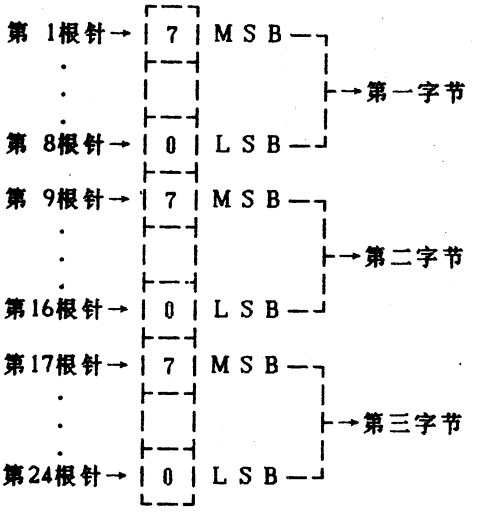
测试打印机的断针之位置，基本原理就是设法直接控制打印机的某一根针来打印一条直线，据此观察所测试的针是否可以打印出直线来。若能够打印出直线，那么可以确认该针完好无损，打印不出直线来，那么该针已经打断。记下该断针的编号，然后用一根好针去替换（打印针的编号顺序我们规定为：最上面的一根针为一号针，其次为二号针、三号针……最下面的一根针为二十四号针），恢复该位置针的打印功能。为了实现直接控制打印机一次只出一根针打印信息，需要了解打印机24根打印针与打印信息字节的对应关系。3070打印机的打印针与打印字节的相互关系是这样对应的（如图一所示）：

图象数据为“1”时，对应的针打印一个点，为“0”时对应的针不打印。与打印头对应的数据分三字节传送（如图二所示），第一字节的最高位对应第一根针，最低位对应第八根针；第二字节的最高位对应第九根针，最低位对应第十六根针；第三字节的最高位对应第十七根针，最低位对应第二十四根针。图象数据的传送列数用n1，n2表示，如图三所示，n1和n1不能同时为零。打印点列数N与n1和n2的关系如下：N＝256Xn1＋n2。

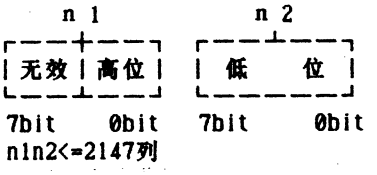
了解了打印信息与打印机24根针的对应关系以后，我们可以向打印机发送控制码序列ESC＋49H＋n1＋n2，将打印机设置成图象打印方式，从程序中对打印信息字节的内容，根据所选定的针号进行有目的的控制，一次只出一根针来打印768个点组成的直线（在程序中我们取n1＝3，n2＝0，得到打印列数为N＝256Xn1＋n2＝3X256＋0＝768）。例如：将3字节数128，0，0传送给打印机后，第一根针出针打印一个点，其余23根针不动作，连续打印768次，即可打印出由第一根针打印出来的一条768点组成的直线。直线的点阵长度用户可以取其它的值，以打印出来的直线便于观察为准。



图一



图二



图三

用户将本文提供的8088汇编语言源程序DAJC.ASM用字处理软件键入计算机，经编译（MASM DAJC;〔回车〕），连接（LINK DAJC;〔回车〕）以后，生成可执行文件DAJC.EXE。此后，用户需要检测3070打印机断针位置时，只要在操作系统提示符下键入：

A＞DAJC〔回车〕

程序运行后，屏幕提示“检测第几号针（1→24），按其余键程序结束）?”用户可输入1－24中的任一数字

来检测相应的打印针打印一条由768个点组成的实线条，并在直线打印完后自动打印出第XX根针打印字样，以便于用户记下断针位置。若用户键入1—24以处的其它键，则程序自动识别用户意图而结束测试，返回DOS。

打印机故障维修——以STAR或GX—15打印机为例

王威　王侯

一、可以自检，但不能与主机连：

I_{c7}(74LS05)，I_{c20}(74LS32)，I_{c9}(74LS374)

I_{c18}(74LS14)

二、有自检现象，但不走纸

1. 走纸马达坏，检测方法：按LF，FF看其是否走纸，如走则是好的，否则是坏的。

2. 如走纸马达是好的，则走纸马达控制电路有问题，查一下TA2，I_{c22}(74LS05)

三、开机能自检，但打印乱字符：

1. 看一下打印机连线是否插好或损坏。

2. 用橡皮打并行接口擦一下，再查一下I_{c19}(74LS374)

四、打印机有自检功能，但乱打空字符

1. 打印头控制电路不好，则查一下以下片子：I_{c16}，I_{c18}，I_{c8}，I_{c21}，I_{c22}，I_{c17}，I_{c20}，TR_3—TR_{11}

2. 如以上没问题，查一下打印头，看针是否有损坏的

针控端　　97531　+5V　2468

各针控制端与+5V端为7Ω，或电流为600mA。

3. 如以上都没问题，则2764ROM坏

五、开机有自检动作，但打印不成形的字符(缺点少划)

1. 打印头本身坏，或有的针被堵塞。

2. 2764ROM字符库坏

六、开机有自检动作，但打印头不向右移。

1. 行打印马达坏：检查方法：关机，把头向右移到头，开机，如打印头回到左侧则好的，否则马达坏。

2. 行打驱动电路坏：查TA1，I_{c23}。

中国计算机学会 电子工业出版社联合举办“学装微电脑”函授班

第四期招生简章

近几年来，我国的计算机普及教育活动有了很大的发展，许多人学习了计算机语言，甚至具备了编制程序的能力。但是，如何进一步利用计算机的接口电路作一些开发应用，以及计算机出了故障敢于动手修理，许多人仍缺乏硬件方面的知识。我们办函授班的宗旨是：在广大计算机爱好者中培养具有一定计算机硬件基础知识和动手能力的人才。

一、招生对象

具有中等文化程度的微电脑爱好者、中学师生、各行业的技术人员、计算机维修人员，也为解放军培养两用人才服务。

二、教学计划与学习要求

每期函授班为四个半月(每周6学时，分为三个单元授课)。

要求学员按时完成每个单元的学习，将作业和实验报告寄回，提出学习中遇到的疑难问题。教师批改作业，评定成绩，解答问题，复信给学员。

学业结束后，根据学员的作业总成绩，经考试合格者，由中国计算机学会发给“全国学装微电脑函授学习结业证书”。

三、招生日期

截止至1992年1月30日

四、教材及学费

全部学费275元，包括：MP—1型学习机全套元器件(以6502CPU为中心的IC电路21块，元器件102个，电源一只，印刷电路板一块，实验板一块)，实验指导书和补充讲义等教材(已购到MP—1学习机者交60元)。

五、报名办法

报名者可写信到中国计算机学会办公室(地址：北京2704号信箱，邮码100080，联系人：宁伟成)索取“学员登记表”。学员填好后连同学费一并交齐，函授班将发出“录取通知书”，由电子工业出版社寄发套件、全套教材。

普及型 PC 机上可用的印刷电路板辅助设计软件 smARTWORK

张保田

八十年代以来出现了微机上使用的印刷电路板辅助设计软件(CAD),smARTWORK 是常用的一种。该软件对微机的要求低,便于掌握使用,可绘制双面印板,具有自动布线功能,所绘制的底图可直接照相制板。

硬件要求和运行环境:

主机:IBM—PC、PC/XT,国产 05 系列,各种 286、386 等,要求 256KB 内存。

显示系统:SmATWORK 工作在 CGA 方式,所以要有彩色图形适配器和彩显,可同时显示印板的两面。在单色适配器配单显情况下,如 SUPER—PC 和 HH—PC,可先运行 MKF. EXE 文件仿真 CGA 模式,再使用 SMATWORK 软件。

输出设备:可选用 FX—100 针式打印机或 DMP 系列绘图仪。可打印绘制 1∶1 图纸用于校对或 2∶1 图纸用于照相制板。

软件要求 PC—DOS 2.0 以上版本支持及 smATWORK 盘两张。

软盘上的 EDIT. EXE 文件是"自动布线编辑软件",运行后有两种工作方式,即布线和命令方式。布线的主要控制键是:

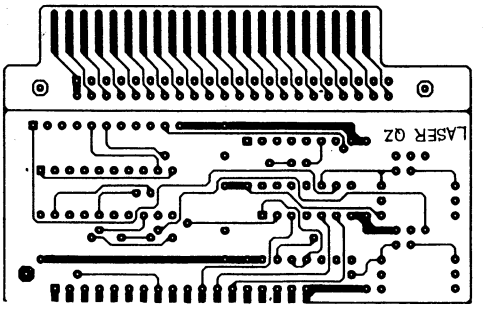
箭头键控制光标上下左右运动,光标每移动一个步距相当于实际尺寸 1.27mm(0.05 英寸),使用数字键可快速移动光标。

F3—置焊盘;

F4—清除焊盘;

F1—自动布线,光标在起点时按一下 F1,再把光标移到终点按下 F1,就在起、终点之间自动布线,以最短路径为标准进行选择。

F2—清除连线;



F5—F7　粗细线转换,粗线为 1.27mm(0.05 英寸),细线为 0.3mm(0.012 英寸)

F8—重复布线,在布一组相同走向尺寸的连线时可重复使用 F8 键。

ALT/F1——单显方式,只显示印板的一面。

ALT/F7——显示全图;

ALT/F9——显示网格;

其他 ALT 和 F 组合键都用于彩显选择色彩。

击回车键转入命令方式,是常用的几个命令是:

HELP——帮助,屏幕上显示 smATWORK 的功能菜单,共三页。

SAVE 文件名.扩展名——把当前板图存入磁盘。

LOAD 文件名.扩展名——调入板图。

smATWORK 的功能很多,本文介绍的只是最主要的部分。附图是用该软件绘制的 LASER 打印卡的印刷板底图,现已缩小为 1∶1,可以看出是相当漂亮的。

近年来出现了功能更强的 CAD 软件包,如 TANGO、E. ESYSTEM(电子工程),但它们对机器要求较高,有兴趣的读者可参阅有关专业资料。

显示器故障检修一例

张　智

故障现象:

一台长城 0520 计算机,打开显示器光栅正常,打开计算机后屏幕全黑,无任何显示,几分钟后,屏幕隐约可见字符,半小时后屏幕显示正常。

故障检查与分析:

将显示器与同型号显示器交换后,故障排除,说明故障出在显示器上,据电器修理经验,怀疑该故障为显象管座受潮,导致聚焦极对地有放电现象,使聚焦电压远低于正常电压,所以,在开机时屏幕上无任何显示,随开机时间的增长,机内温度升高,显象管座内的湿气逐渐蒸发,聚焦极对地之间的绝缘电阻增高,使屏幕显示正常。

故障排除:

拔掉电源,打开显示器机壳,小心拔下显象管座,向管座封装间隙内注满乙醇,一分钟后倒掉乙醇,用电吹风吹干后,插上管座,开机故障仍存在,再次拔下管座,从电路板上焊下显象管座,用小刀和小平口镙丝刀,将管座的密封间隙塑料盒敲开,发现密封盒内的聚焦极受潮严重氧化,产生绿色氧化物,盒内四壁表面吸附有杂质,用小刀仔细刮除氧化物及杂质,用乙醇清洗后,在白炽灯上烘干,再在盒内表面涂上一层硅脂,装上密封间隙塑料盒,用 502 胶水密封,装好显象管座,故障排除。

通过以上故障的检修,充分说明,维护好机房洁净,经常除尘除湿,是一个不可疏忽的问题。

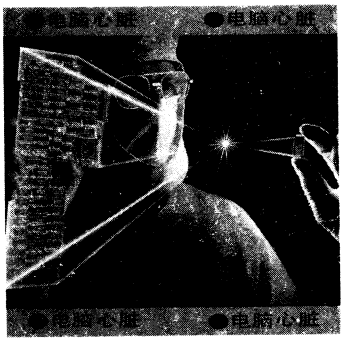
ISSN 1000-1077

電子與電腦

12

一九九一年

总期第81期



電子與電腦

目　录

·综述·

新型信息录入技术 一条码技术及其应用 …… 王枞等(2)

·PC 用户·

dBASE Ⅲ 实现 Foxbase 中 prompt 的菜单功能 …… 毛维平(4)

信息系统中模糊查找的设计 …… 戚靖山　朱玉环(5)

高密软盘的单驱动器拷贝 …… 李　毅(6)

用汇编程序显示硬盘主引导记录 …… 李日林(7)

PC DOS 内存管理 …… 陈　宏(8)

CC BIOS 2.13F 的小补钉 …… 严仲武(9)

中文弹出式窗口设计 …… 姜咏江(10)

一种实用新颖的排序方法 …… 朱国江(12)

·学习机之友·

多层次全能阴影菜单 …… 黄文贵(13)

虚线 …… 刘　裕(14)

技巧两则 …… 宋正荣(15)

改进 MUSIC 的发音 …… 熊　波(15)

中华学习机 RAM 监控及其应用 …… 朴一兵(16)

CEC-I 键控光标作图程序 …… 陈国良(17)

ProDOS 磁盘操作系统入门(续) …… 廖　凯(18)

·C 语言初阶讲座·

第十讲　文件管理 …… 李文兵(20)

·学用单片机·

SCB-I 型单板单片机的存储器改进 …… 江　琪(26)

MCS-51 单片机端口结构图示中的一个普遍错误 …… 王修才(27)

·学装微电脑·

串行 I/O 部件的制作与实验 …… 易齐干(28)

"微电脑控制技术"函授学习班总结 …… 易齐干(30)

·电脑巧开发·

将 MP－1 电脑的时基信号改为晶体振荡 …… 邓文超(31)

·电脑游戏机·

第二章　FBASIC 的基本语句 …… 于　春(32)

·维修经验谈·

计算机电源修理 …… 黄培鑫(36)

四通 MS－2401 打字机使用维修经验 …… 蔡长安(40)

修理 CEC-I 型中华学习机软盘驱动器一例 …… 赵　升(41)

谈谈尼龙打印色带的使用技巧 …… 林　斌(42)

·初级程序员级水平考试辅导讲座·

计算机病毒防范问答 …… 常晓冀(44)

SCB－1 单片单板机简介 …… (48)

封面　电脑心脏

封二　电源原理图

封三　中软总公司市场开发部提供的产品

封底　SCB－1 单片单板机

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:张　丽

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

新型信息录入技术
——条码技术及其应用

机械科学研究院　王枞　高群　董丽珠　赵文　过海春

随着现代科学技术的高度发展，计算机已经步入越来越多的领域，逐步从科研进入民用，进入到人们的生活中。人们已经发现电脑在民航售票处、图书馆、铁路托运站、银行中应用。不远的将来，人们还会发现：在医院挂号、商店买东西，甚至上班报到时，自己会直接地跟电脑打交道。原来由人做的事，现在由电脑做了。

本文将为您介绍条码技术及其应用，希望它能帮助您扩展知识、了解身边发生的事情，从而跟上时代的发展。

一、条码的种类及条码系统

条码(BarCode)是一种由一组宽窄不同、黑白相间的线条的排列组合、以代表一定字符的图形编码。这个编码可以是一组数字，或一串英文字母，或数字和字母的混合。数字和字母本身并不重要，重要的是它们所代表的内容，而计算机就是通过编码识别、处理它要完成的工作。

一般计算机工作是通过键盘输入，而条码技术是通过条码译码器和扫描器输入，扫描器的作用象镜子，它把"看到"的条码通过译码器输入给计算机，这样计算机就得到了和键盘输入同样的信息。所以，条形技术作为一种图形识别技术，具有方法简单、可靠性高、信息采集速度快、采集信息量大、信息的对应性强等特点，不仅大大降低了操作人员的工作强度，也减少了人为错误的发生。条码技术的开发，是人们对信息的采集和输入技术的突破。目前，采用自动识别技术，实现信息采集和输入的自动化已成为一种趋势。

由于条码技术的广泛应用，其种类很多，目前国际上使用的条码主要有六种：

1. 通用产品码(Universal Product Code)。该码 1973 年由美国统一代码委员会制定，简称 UPC 码，它主要用于美国、加拿大商品流通领域。

2. 国际物品码(International Article Number)。该码 1977 年由国际物品编码协会制定，简称 EAN 码。它主要用于世界各国和地区(美国、加拿大除外)的商品流通领域，该码与 UPC 码兼容。

3. 2/5 码(2 of 5 Bar Code)。它是一种仅能表示数字信息的高密度条码，它又分矩阵二五码和交叉二五码两种。

4. 39 码(Code 39)。该码由于可以表示英文字母，因而广泛用于工厂自动化和办公自动化领域。

5. 库德巴码(Codabar Code)。该码主要用于邮电、图书管理等领域，在美国该码还用于医疗卫生业。

6. 128 码(Code128)。它是一种与 ASCII 码完全对应的条码。

条码系统(Barcode System)是指由条码符号、条码符号的生成(印刷、打印等方式)、条码符号的扫描以及译码等部分组成的自动识别系统。该系统是以大规模集成电路技术、新型光电传感技术、精密机械加工技术、印刷、打印技术以及计算机应用技术作为支撑。因此条码系统是一种综合性的高技术系统。

二、条码技术的特点

从上面介绍，我们知道条码技术实际上是计算机信息输入的一种方式。在计算机信息输入所采用的多种自动识别技术中，条码技术与其它识别技术相比，其特点表现为下面七个方面：

1. 方法简单、成本低

条码作为一种图形信息技术与其它技术(如文字识别技术)相比，有图形简单，易识别的特点。

2. 信息采集速度快

信息采集速度是衡量信息采集水平的一个重要的指标。由于条码自身的黑白相间的线条正好对应于信号的高低电平，即逻辑信号的"1"和"0"，因而使用条码技术采集信息的速度快，一般译码时间不超过几毫秒，甚至可以更短。如日本东研公司已开发出能读出转动着的电风扇叶片上条码的解读器；

3. 采集信息量大

条码自身具有很大的灵活性，它可进行多种编码，许多条码方法对其信息长度(一个条码所代表字符的位数)从理论上讲不受限制(实现上受限于译码器的能力，如光电扫描的幅度等)。由于条码图形简单这一特点，因而可以相对提高线条的密度；

4. 可靠性高

利用普通键盘输入信息时，其字符的置换率为 800 分之一，且信息的可靠性受人为因素的影响甚大。其它一些自动识别技术，例如，文字识别技术，由于文字图形的不规范和复杂性，对其可靠性有所影响。再如，声音识别技术，在声音识别中，由于人为因素可使其可靠性下降。使用条码技术，其字符的置换率可达百万分之一以上，并且在计算机软件的支持下，其可靠性可进一步提高；

5. 信息的对应性强

在生产、生活活动中，大量的信息是针对某一确定物体的，离开物体，信息本身则失去意义。由于条码可以印(贴)在物体或物体的包装物上，因而使物流信息和物品的对应性得以保证。这是条码的一大特点，因此有人把条码技术称之为物流信息技术；

6. 大大降低劳动强度

手工信息录入是一项劳动强度很大的工作。然而条码技术是一种自动识别技术，在使用固定式条码译码器时，不需要人力，机器自动完成信息采集任务，在使用台式或便携式译码器时，操作人员仅做一些非常简单的工作，如扳动开关

等。因此使用条码技术能大大降低劳动强度；

7. 应用范围大

条码技术自问世十余年来，其发展迅速是人们不曾预料的。在经济发达国家，十余年来的发展，使得条码的有关产品系列化，以适应各方面的需要。如适用自动化流水线的固定式条码译码器，应用于办公自动化及图书管理等的台式译码器以及不受工作现场限制的便携式译码器已被人们广泛使用。目前在国际上，条码技术被广泛用于商业、工业、邮政、生产管理、图书管理、人事管理等各个领域。

三、条码技术的应用

70年代美国最早开始应用条码，由于条码技术所具有的诸多优点，因此，自它问世以来，无论是其技术本身还是应用领域都以惊人的速度向前发展和扩展。近几年来，条码技术在许多国家广泛用于各流通领域和生产部门，它所带来的经济效益和社会效益是人们无法估量的。

1. 国外条码技术应用概况

(1)商业

商业是条码技术应用最早的领域，也是当代条码技术应用最广的领域之一。七十年代中期，美国超级市场商曾预言：没有条码的商品将不能摆在超级市场的货架上。今天美国超级市场场商出口商标上的印码率已达99%。在日本，现也已普遍采用条码的实时售货系统——POS(Point of Sale)系统。

(2)工业

继条码技术在商业上广泛应用之后，条码技术又开始向工业领域渗透。在美国汽车工业中，协作件的流通规定采用条码技术，它涉及了美国25000家公司。这一措施大大加快了协作件的流通速度，提高了效益。此外，汽车生产装配线也采用了条码技术进行实时控制和工作过程的跟踪。

(3)仓库管理

现代化仓库管理如同超级市场一样，是条码技术的应用领域。在此值得一提的是美国国防部在军用品管理上采用条码技术。美国国防部1982年下达了1189号文件(MIL－STD－1189)，要求供应军用品的5万家公司必须在他们的产品上附有规定的条码。现在美国国防领域已成为美国条码技术产品的最大市场，其市场占有率平均为30%。

(4)办公自动化

条码技术在办公自动化方面的应用也发展很快，现在已广泛用于人员管理(如考勤系统等)、图书馆管理、传票处理、档案管理、海关的出入境管理以及机票管理等，在办公自动化领域条码起着逐渐替代键盘的作用。

2. 我国条码技术应用概况

由于条码技术作为一种新技术引入我国时间不长，因此无论是在条码技术的研究、开发方面，还是在条码技术的应用方面，尚处于起步阶段。

四、我院的开发成果

在国家“八五”计划中，我院承担了便携式译码器和台式译码器的研制开发。现在，我院已开发出键盘仿真型条码译码器及有关的应用系统，如条码出入管理(考勤)系统和条码档案管理系统等。

1. 键盘仿真型条码译码器

键盘仿真型条码译码器与微机键盘并联，其输出与键盘输出信号相同，通过键盘接口直接送入计算机，不需要任何硬件扩充及编写专用支持软件，使用起来和键盘一样方便。该译码器可读取ITF码、CODE39码、CODABAR码和JAN/EAN码，可用于IBM-PC、PS/55、PS/2系列的微机，可使用光笔、CCD或激光枪扫描器。

2. 条码出入管理(考勤)系统

条码出入管理系统用于需要进行人员出入情况记录和统计的任何部门和单位。采用条码技术出入管理系统，融合了条码录入技术和计算机管理，使人员的出入管理工作具有手续简单、速度快、实时性强等优点，已成功地用于职工出勤情况管理。

条码考勤系统由一台IBM-PC、一台键盘仿真型译码器、一支光笔(或卡槽)及考勤软件组成。

条码考勤软件由四个子系统组成：

- 实时考勤：随时记录人员的出勤、请假、缺勤等情况的时间和原因。
- 查询：使部门领导随时掌握某部门人员的出勤情况。
- 修改：是某些特定情况下，人工干预时使用的。
- 考勤卡生成：根据不同的需要用激光打印机或针式打印机生成不同格式、特征的胸卡。

条码考勤系统操作简便、使用方便、还在我院试点运行半年以上，深受用户欢迎。

3. 条码在档案管理系统中的应用

条码技术与计算机相结合应用于档案管理，不仅可以提高档案管理的可靠性，促进档案管理现代化、制度化，而且可以大大提高工作效率，同时实现档案借阅、需求情况的实时登录、管理。

档案的收集、整理、鉴定、保管工作都是为了向使用者提供全面、丰富的资料。可以说，档案的使用工作是档案管理的目的。因此，档案馆中的管理主要是指档案使用情况的管理。

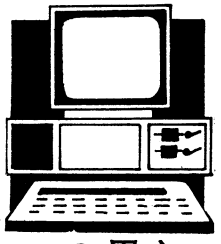
在接待室、出纳台和库房各配一台微机，每台微机配有与键盘并连的条码解读器作为信息输入手段，微机间则用通讯方式进行数据交换。

在这样的系统配置下，各部门完成的主要工作如下：

- 接待室：接受读者实时档案目录查询，对索求的在库档案产生出库提单，对索求的在外档案按需要和可能进行预约登记。
- 库房：通过该处微机上所连的译码器读取出入库案卷信息，修改库存数据库。
- 出纳台：办理借、还手续。

此情况下，三处均可实时响应，通过三处中任一处的微机可实时查阅，统计档案出借情况和使用情况。

条码技术作为一种先进的信息采集和输入技术，引入我国的时间虽然不长，但已有了长足的进步。现在专门负责条码登记管理的国家物品编码中心已经成立，最近我国已参加了国际物品编码组织，并计划于“八五”期间使我国6%以上商品使用标有条码的商标；同时，国家也将条码设备作为“八五”重点项目投资开发，为条码技术在我国的发展创造了的条件。随着我国经济建设和科学技术的发展，随着人们对信息采集和输入技术需求的增强，条码技术定将广泛应用于我国国民经济建设的各个领域。



PC用户

dBASE Ⅲ实现 Foxbase 中 Prompt 的菜单功能

广东佛山绣口总厂　毛维平

Foxbase 中的…@prompt '…'message '…'和 menu to〈变量〉语句给程序设计者们提供了一种简洁的菜单编程方法，特别给用户带来了一种方便新颖的感觉。不少 dBASE 用户也想得到这种享受，然而 dBASE Ⅲ没有提供直接实现这种功能的语句。为了解决这个问题，笔者在 PC 机上用 dBASE Ⅲ编了一个菜单程序，从操作者界面看与 Foxbase 完全一样。

程序用 INKEY()函数接收键盘输入的"↑"、"↓"、"↙"键，整个过程只接受这三个键；a1～a5 为显示各模块的字符串，即相当于 Foxbase 中 prompt 语句后面的内容，b1～b5 为各相应模块的说明或下层菜单说明，也就是相当于 Foxbase 中 message 语句后面的内容；P 为 1～5，作为变量的下标，为宏替换提供方便；X 的值决定了所选择的模块；语句 set color to n/w 、set color to w/n 设置显示反相和正常显示，当按动"↑"(或↓)时，上(或下)一个模块(字符串)显示为反相，原来模块恢复正常显示，从而实现了光条滚动的效果。若按下"↙"(回车键)，表示选中某模块。

```
set exact on
set talk off
set heading off
do while .t.
clear
@ 5,25 say '* * * * * 菜　　单 * * * * * *'
a1='1. 输 入 资 料'
a2='2. 修 改 资 料'
a3='3. 查 询 资 料'
a4='4. 打 印 报 表'
a5='5. 退　　出'
b1='输入出库单.入库单.调拨单'
b2='修改各种单据及调整产品库存'
b3='查各种单据.产品库存.客户购货情况'
b4='打印各种库存报表.销售报表'
b5='退出成品仓库管理系统'
@ 7,26 say a1
@ 9,26 say a2
@ 11,26 say a3
@ 13,26 say a4
@ 15,26 say a5
@ 20,15 say '按 '+chr(24)+'.'+chr(25)+'键上、下移
  动光标，按回车键执行选择!'
aao=.t.
do while aao
x=7
p='1'
do while .t.
cm='a'+'&p'
bm='b'+'&p'
set color to n/w
@ x,26 say &cm
@ 18,21 say &bm
i=0
do while i=0
i=inkey()
enddo
set color to w/n
@ x,26 say &cm
@ 24,21 say &bm
if i=5
    if x=7
        x=15
        p=5
        p=str(p,1,0)
    else
        x=x-2
        p=val(p)-1
        p=str(p,1,0)
    endif
endif
if i=24
    if x>13
        exit
    else
        x=x+2
        p=val(p)+1
        p=str(p,1,0)
    endif
endif
if i=13
        aao=.f.
        exit
endif
enddo
enddo
clear
do case
        case x=7
          do 输入模块
        case x=9
          do 修改模块
        case x=11
          do 查询模块
        case x=13
          do 打印模块
        case x=15
        cancel
endcase
enddo
```

信息系统中模糊查找的设计

厦门集美航海学院计算机教研室　戚靖山　朱玉环

在信息系统的研制中经常会碰到通过某一项的已知条件找出相关的其它信息。当这一已知条件是精确值时，即正好是记录中的某一项的值(严格匹配)时，是很容易实现的，但一旦中间插入空格，或者是输入简称(如“上海师范大学”输入“上海师大”)时，则在记录中找不到相应的值。所以，在研制信息系统时，往往要考虑设计具有模糊查找功能的模块。

目前流行的模糊查找的方法，通常是单纯用 AT() 函数来判别输入的已知条件(汉字串)是否是记录中的被检测项值的子串，若是，则显示该项值(汉字串)，提问是否是该字符串；若不是，则继续找下一条记录；如果不是被检测项值的子串，则说明不是这条记录，继续找下一条记录。这样查找的局限性很大，使用灵活性也不高。譬如说，严格要求数据库里被测试项的值与输入的已知条件(汉字串)前后及中间不能空格，如要找记录中的单位这一项中“上海师范大学”，输入的已知条件为“上海 师”则不行；又譬如，要查找“上海师范大学”，输入“上海师”可以，输入“上海师大”则不行，严格要求输入的已知条件(汉字串)是被检测项中查找的汉字串中的连续汉字组成的子串，这样显然不便于用户的习惯使用，这样设计出来的查找功能只能属于初级阶段的模糊查找。

为此，我们设计了一种功能较强的模糊查找，其特点是：

1. 输入的已知条件(汉字串)和被查找的汉字串允许前后有空格，汉字串中间允许有若干个空格。

如：输入的已知条件(汉字串)为：

“上　　海　　”

被查找项中的值(汉字串)为：

“　　上海　师　范大学　　”

2. 输入的已知条件(汉字串)可以是要求找到的汉字串中间隔若干个汉字组成的汉字串。

如：要求查到的汉字串为“上海师范大学”

输入的已知条件(汉字串)为：“上海师大”

3. 上述两种情况的混合

如：要求查到的汉字串为：“上海师范大学”输入的汉字串为：“　上　　海师　大”

又如：要求查到的汉字串为：

“上海　　师　范大学　”

输入的汉字串为：

“　上　海师　　大”

上述三种情况照样能查找到。

这种模糊查找真正体现出模糊的功能，给用户提供了极大的方便。

该模糊查找的设计思想为：

取出第一条记录的被检测项的值，然后先把被检测项的值和已知条件(汉字串)都滤去前后的空格，将已知条件的第一个汉字取出，判断是否被包含在检测项中，若不是则退出找下一个记录，若是则截取该汉字后(不包括该汉字)的汉字串做为新的已知条件，截取被检测值该汉字以后(不包括该汉字)的字串作为新的被检测值。同样再滤去各自的前后空格，重复上述过程，直至原来已知条件中的所有汉字都被测试且全部匹配过为止。这时该记录中的测试值就是要查找的汉字串。

步骤：

1. 打开数据库(含被检测项)，指针指到起始位置。

2. 输入已知单位(汉字串)到 CXNR 变量中，并复制一备份在变量 CXNQ 中。

3. 假若输入的已知单位(汉字串)是空串，则返回。

4. 把第一条记录的被测试项的值送入 WX 变量中。

5. 滤去已知条件(汉字串)中的前、后空格，送入变量 ZXNR 中，并把汉字串的长度送入 X 中，当 X 为0时则返回。

6. 取出 ZXNR 中最左边的一个汉字，送入变量 E 中。

7. 滤去 WX 中的汉字串的前、后空格并送入 CX 中。

8. 判别 E 中的汉字是 CX 中的子串吗?若是，则把 ZXNR 的长度减去2，并判断是否为0。若为0则表示已知条件(汉字串)中所有的汉字都检测完了，包含在被检测项的值中，那么就显示该检测项的值是否要找的值，是则退出，否则恢复已知条件的初值，重新从下一条记录找起。当 ZXNR 的长度不为0时，则截取 CX 中的汉字串从 E 汉字以后到末尾的汉字串送入 WX，ZXNR 中的汉字串也同样去掉第一个汉字以后送入 ZXNR 中，返回到〈5〉以后继续执行。

当 E 中的汉字不是 CX 中的子串时，则退出后恢复 CXNR 的初值(即已知条件的汉字串)，找下一条记录的被检测项的值，送入 WX 变量以后，从第五步开始做起。

假如数据库所有记录中被检测项的值都不满足要求，则显示没有该单位，退回。

附上在“高教论文信息管理系统”中设计的模糊查找 MFCZ. PRG 模块。

```
CLEAR
SET TALK OFF
USE YBYXX                    (YBYXX 为含有被查项的数据库)
GO TOP
CXNR=SPACE(14)               (下标为被查项长度)
CXNQ=CXNR
```

```
WX=CXNR
CLEAR
@ 5,30 SAY "输入单位" GET CXNR
READ
CXNQ=CXNR
IF CXNR="                    "
RETURN
ENDIF
DO WHILE .NOT. EOF()
WX=NAME
DO WHILE .T.
ZXNR=SPACE(LEN(LTRIM(RTRIM(CXNR))))
ZXNR=LTRIM(RTRIM(CXNR))
E="  "
X=LEN(ZXNR)
IF X=0
RETURN
ENDIF
E=LEFT(ZXNR,2)
CX=SPACE(LEN(LTRIM(RTRIM(WX))))
CX=LTRIIM(RTRIM(WX))
IF AT(E,CX)<>0
X=X-2
IF X=0
CLEAR
@5,30 SAY "工作单位:"
@5,40 SAY NAME
T="  "
WAIT "是否(Y/N)?" to T
IF T="Y"
RETURN
ELSE
CLEAR
EXIT
ENDIF
ENDI
IF LEN(CX)=AT(E,CX)+1
CLEAR
EXIT
ENDI
WX=SUBSTR(CX,AT(E,CX)+2)
CXNR=SUBSTR(ZXNR,3)
LOOP
ELSE
EXIT
ENDIF
ENDDO
SKIP
CXNR=CXNQ
LOOP
ENDDO
CLEAR
@ 5,30 SAY "没有该单位!"
WAIT
RETURN
```


高密软盘的单驱动器拷贝

北京师大二附中高三(2)班　李毅

随着286/386微机的普及,容量为1.2MB的高密驱动器也越来越成为常见的配置了。一般的中档286/386微机上都只配一台高密驱动器,而且这些机器上的内存数量一般也只有一兆左右。这给高密软盘的拷贝带来了极大的不便。我在使用这类机器的时候,也遇到过这样的麻烦。因此,我编制了一个利用硬盘做为缓冲区的高密盘拷贝程序,把它推荐给大家,以便在某种程度上减少麻烦。

程序是用 Turbo C++ 1.0编写的。但由于没有使用与C++有关的特性,这个程序也可以用 Turbo C 2.0编译,效果完全一致。由于磁盘的格式化操作要求较高,需要较多的程序才能完成,这里给出的程序就不包括磁盘格式化部分了,也就是说,在使用这个程序之前,请事先将磁盘格式化好(如果磁盘上有内容,即过去曾经使用过,就没有必要再格式化了)。

李毅同学:请函告新地址,以便奉寄稿酬。编者。

程序清单:

```
#include <fcntl.h>
#include <dos.h>
#include <io.h>
#include <process.h>
#include <alloc.h>
#include <conio.h>
#include <ctype.h>
#define buffer_size 7680
#define MAX_SECTOR 2399
int testyn(void);

main()
{
char *buffer;
int secno, nsecno=15;
int handle, done=0;
_fmode=O_BINARY;
buffer=malloc(buffer_size);
puts("High Density Floppy Disk Copy Utility Version 1.01");
puts ("Public Domain 1991, by Shadow Information Lab.");
puts("Beijing Normal University, 100875, BJ, CHINA.");
while(!done)
{
puts("\nInsert SOURCE Diskette in drive A:");
puts("Press any key when ready.");
while (!kbhit());
getch();
if ((handle=_creat("C:\\TEMP.BUF",FA_ARCH))==-
1)
{
```

```
puts("Sorry! Unable to access buffer on hard disk.");
exit(1);
}
for (secno=0; secno<= MAX_SECTOR; secno+= nsecno)
{
if (absread(0, nsecno, secno, buffer)==-1)
{
puts("Error Access SOURCE Diskette.");
exit(2);
}
write(handle, buffer, buffer_size);
}
close(handle);
puts("\nInsert TARGET Diskette in drive A:");
puts("Press any key when ready.");
while (!kbhit());
getch();
if ((handle=_open("c:\\TEMP.BUF", O_RDWR|O_BINARY))==-1)
{
puts("Sorry! Unable to access buffer on hard disk.");
exit(1);
}
for (secno=0;secno<= MAX_SECTOR;secno+=nsecno)
{
read(handle, buffer, buffer_size);
if (abswrite(0,nsecno, secno, buffer)==-1)
{
puts("Error Access TARGET Diskette.");
exit(2);
}
}
close(handle);
printf("\nDo you want another copy(Y/N)?");
done=tstyn();
}
system("DEL C:\\TEMP.BUF");
return 0;
}
int testyn(void)
{
char ch;
for(;;)
{
while (!kbhit());
ch=getchar();
switch(toupper (ch))
{
case 'Y': return 0;
case 'N': return 1;
}
}
}
```

用汇编程序显示硬盘主引导记录

李日林

硬盘主引导记录是极为重要的系统程序，是使用硬盘必不可少的系统信息区。它包括引导与分区代码、分区表和结束标志"55AA"。但是，由于硬盘主引导记录位于DOS分区之外，在0磁头、0柱面的第1扇区，所以不能直接用调试程序DEBUG访问。为了查看其内容，通常利用BIOS提供的13H号软中断来读取，再用DEBUG中的命令显示，虽然该方法不失为一种可行的手段，但操作、使用不方便，尤其对不熟悉其结构的用户来说，更显得麻烦。

笔者用8088汇编语言编制了一段源程序，经汇编、链接后，可以方便地读取硬盘主引导记录，并且直接在屏幕上显示其内容，无须利用DEBUG命令，操作、使用都很简单。因注释相当清楚，对程序设计的细节就不再详述。

程序实例 DISPBOOT.ASM

```
output  macro a              ;显示一个字符的宏指令定义
        mov dl, a
        mov ah,2
        int 21h
        endm
;
stack segment para stack 'stack' ;堆栈段
      db  256 dup(?)
stack ends
;
data  segment para public 'data' ;数据段
buffer db  512 dup(?) ;定义存放一个扇区的缓冲区
data  ends
;
code  segment para 'code' ;代码段
start  proc far; 汇编程序标准序
       assume cs:code, ds:data, es:data,ss:stack
       push ds
       xor ax,ax
       push ax
       mov ax,data
       mov ds,ax
       mov es, ax
;
       mov ax,0201h   ;读硬盘主引导记录
       mov bx, offset buffer ;存放于定义的缓冲区
       mov cx,0001h
       mov dx,0080h
       int 13h
;
       mov dh, 0
       mov cx,512
```

```
        mov bx,0
lp:     push cx   ;以下程序段为取一个字节
        mov dl,[bx] ;将其转换为 ASCII 码用于显示
        push dx
        mov cl,4
        shr dl,cl
        call ascii
        pop dx
        shl dl, cl
        shr dl,cl
        call ascii
        cmp dh, 10h  ;每行显示的字节数为10H
        jne np
        output 0ah
        output 0dh
        xor dh,dh
        jmp np1
np:     output 20h
        inc dh
np1:    inc bx
        pop cx
        loop lp  ;循环,直至显示完一个扇区
        output 0ah
        output 0dh
        mov ax, 4c00h  ;返回 DOS
        int 21h
start   endp
ascii   proc near  ;子程序,将字符转换为 ASCII 码
        cmp dl,9 ;并显示
        jb number
        add dl, 37h
        jmp output1
number:   add dl, 30h
output1:  push ax
          mov ah,2
          int 21h
          pop ax
          ret
ascii     endp
;
code    ends
        end start
```

PC DOS 内存管理

海南省商业学校计算机室　陈宏

PC DOS 为了管理内存空间，采用了链式结构，在内存中建立一条由若干内存控制块组成的内存控制块链。该链掌握了整个内存的使用和分配情况，DOS 根据此链就能完成整个内存的管理。内存控制块(MCB)是在 DOS 分配内存时，在所分配的内存块的头十六个字节设置用于描述内存块的特性、段号和容量。MCB 结构如下：

0	1	2 3	4 5	16
标志	分配段号	分配块节数	保	留

1)标志：＝4DH 表示是 MCB 链中的某一块，＝5AH 则是最后的一块。

2)分配段号：说明该内存块已分配的还是空闲的

0000＝空闲的；　XXXX＝程序的 PSP 段址

3)分配块节数：说明该内存块的容量的大小，单位是节，1节＝16字节。

MCB 链中的第一块在 DOS 引导过程中随同第一个 PSP 建立而建立，相应内存块含有系统开工时的文件缓冲区(FILES)、磁盘缓冲区(BUFFERS)和可安装设备驱动程序(DEVICE)，因此该 MCB 容量大小取决于 CONFIG. SYS 的上述三个命令。第二块分给 COMMAND. COM 文件，内容除 PSP 后是 COMMAND. COM 常驻部分的的代码和数据。DOS 载入之后，在用户的可用内存低端总存在一块标志为5AH 的结束 MCB。当 DOS 要载入外部命令或用户应用程序时，EXEC 子功能为程序申请两块内存块，一块放程序的环境块，一块放程序的 PSP 和程序本身。无论是 EXE 或 COM 文件，载入后 DS、ES 总指向 PSP 段址，如将该指针减1就得到含有 PSP 和用户代码和数据内存块的 MCB 段址，文件病毒为在内存中潜伏下来又不能让 DOS 发觉，就直接去修改 MCB，这点与引导扇区病毒很不相同。

各内存块之间的关系是：控制指针 i＋1＝控制指针＋(位移3)i＋1，而 DOS 将第一块 MCB 的指针保留在 DOS 程序段的特定位移内，按此公式便能计算出任何一块 MCB 的段址，通过改写控制块的有关信息达到控制相应内存块的目的，如释放内存(49H)只是将相应的 MCB 位移1改成00，标志为可用块表明是空闲块即可。修改内存4AH 稍复杂些，要修改相应的 MCB 的大小和重新设立剩余 MCB。Microsoft 公司并没有公布 MCB 结构的细节，直接修改 MCB 要冒一定的风险，当然最好是调用 DOS 为用户提供的关于内存管理的系统调用功能，如申请内存48H、释放内存49H 和修改内存4AH 等，否则 MCB 链一旦断链，整个系统将陷入瘫痪状态并停机(System halt)

运行笔者开发的汇编语言程序的可执行代码便得到当前 DOS 内存的 MCB 情况(如图1)。图中 NO 项是 MCB 的序号，SEGMENT、SIZE(单位：节)分别是 MCB 的1－2、3－4字节的内容，FILE 是掌握该内存块的程序名；PEM 指明内存块的用途，PSP 存放程序 PSP 和程序代码数据；ENV 是环境块；MEM 是既非空闲块也不是 PSP、ENV 的驻留内存块，图中 NO. 6是1701病毒

占用的 MCB。

```
    NO. SEGMENT   PSP   SIZE   PEM   FILE
-------------------------------------------------
1   09F9    0008    817
2   0D2B    0D2C    211   PSP   \COMMAND. COM
                              0024(0D2C:0556)
                              002E(0D2C:0281)
-------------------------------------------------
3   0DFF    1197      3   ENV   MCB. EXE
4   0E03    0D2C     10   ENV   \COMMAND. COM
5   0E0E    0E14      4   ENV   C:\PASCAL\DEBUG. COM
6   0E13    0D2C    123   MEM\COMMAND. COM
                              0021(0E14:031C)
-------------------------------------------------
7   0E8F    0E95      4   ENV   C:\PASCAL\DEBUG. COM
8   0E94    0E95    769   PSP   C:\PASCAL\DEBUG. COM
                              \COMMAND. COM
                              0001(0E95:08ED)
                              0003(0E95:08E6)
                              0022(0E95:0242)
                              0023(0E95:08E1)
-------------------------------------------------
9   1196    1197  36,329  PSP   MCB. EXE
                                  C:\PASCAL\DEBUG. COM
                                  \COMMAND. COM
  Total free memory:36,329 tytes
  Program terminated normally
```

图1.

MCB. ASM 主要是利用了内存控制块、程序前缀控制块 PSP 和环境块三者之间的关系而设计的，其中在 PSP＋16H 处是父进程的 PSP 段址，子进程可以通过此指针返回父进程，最后会回到命令处理文件 COMMAND. COM，COMMAND. COM 的 PSP＋16H，则指向了自己本身，如下图

```
MCB
-D 1196:0 4F
(原 MCB. EXE 的 MCB 和 PSP 内容，程序结束后该内存块被释放，参阅图1)
1196:0000  5A 00 00 E9 8D A2 02 01-53 B4 30 CD 21 5B 3C 02
PSP (1197:0)
1196:0010  CD 20 80 9F 00 9A F0 FE-1D F0 42 02 995 0E 70 02
1196:0020  95 0E 56 05 2C 0D 95 0E-FF FF FF FF FF FF FF FF
1196:0030  FF FF FF FF FF FF FF FF-FF FF FF FF 00 0E E8 FF
1196:0040  A7 11 14 00 18 00 97 11-FF FF FF FF 00 00 00 00

-D E94:0 4F   DEBUG. COM PSP 段址 0E95＝E94＋1
0E94:0000  4D 95 0E 01 03 00 00 00-00 00 00 00 00 00 00 00
0E94:0010  CD 20 80 9F 00 9A F0 FE-1D F0 2F 01 2C 0D 3C 01
0E94:0020  2C 0D 56 05 2C 0D 2C 0D-03 04 01 00 02 FF FF FF
0E94:0030  FF FF FF FF FF FF FF FF-FF FF FF FF 90 0E C0 2A
0E94:0040  95 0E 14 00 18 00 95 0E-FF FF FF FF 00 00 00 00

-D D2B:0 4F   COMMAND. COM 段址 0D2C＝D2B＋1
0D2B:0000  4D 2C 0D D3 00 59 72 EC-16 1F 80 3E 15 05 FF 72
0D2B:0010  CD 20 80 9F 00 9A F0 FE-1D F0 B2 02 2C 0D 3C 01
0D2B:0020  2C 0D 56 05 2C 0D 2C 0D-03 04 01 00 02 FF FF FF
0D2B:0030  FF FF FF FF FF FF FF FF-FF FF FF FF 04 0E 25 08
0D2B:0040  2C 0D 14 00 18 00 2C 0D-FF FF FF FF 00 00 00 00
```

在 PSP＋2CH 处有环境块的段址，环境块反映的是 COMMAND. COM 和程序所处周围的情况，由若干以0为结束标志的字符串组成，一般至少有 COMSPEC、PROMPT、PATH 三个字符串组成，在整组环境块结束时以另一个0为结束，在此0字节之后是一组初始化变量，含有指明当前执行程序的驱动器、路径、文件名[.扩展名]的 ASSCII 串，如下图：(DOS 3.3)：

```
D E0E:0 4F
0000 4D 14 0E 04 00 00 00 00-00 00 00 00 00 00 00 00  M……
0010 50 41 54 48 3D 00 43 4F-4D 53 50 45 43 3D 43 3A  PATH=. COMSPEC=C
0020  5C 43 4F 4D 4D 41 4E 44-2E 43 4F 4D 00 00 01 00 \COMMAND. COM…
0030  43 3A 5C 50 4E 53 43 41-4C 5C 44 45 42 55 47 2EC:\PASCAL\DEBUG.
0040  43 4F 4D 00 04 D3 EA 8C-C8 03 C2 A3 58 16 A1 16OM. . SJ. H. B#X. ¦.
```

MCB. ASM 由此而获得当前进程名。另外，如在 PSP＋2C 处为0则表示环境完全复父进程

MCB. ASM 目的是能及时地了解当前内存的具体使用情况，便于发现文件病毒入侵的可疑之处，源程序中 ES:DI 一般指向程序的显示区，很容易加上其它功能模块，清单中已有较详细的注释，有兴趣的读者可在此基础上加以改进，实现如删去无用的内存空间，驻留程序、占用大量的内存的汉字库等。

（有关 MCB. ASM 的程序清单，有兴趣的读者可向编辑部函索，收复印及邮挂费2.00元）

CCBIOS 2.13F 的小补钉

福建省第二电力建设公司　严仲武

CCBIOS 2.13F 是中国计算机系统工程公司于89年9月推出的适用于 IBM－PC 各档微机的一个较成功的汉字操作系统，它以快速、方便和大量特殊的视屏、打印机扩展功能为用户提供方便。按 CCBIOS2.13F 汉字系统《用户手册》介绍，在汉字或词组输入时，用 Alt＋Esc 键重选第一个字。但实际使用中发现这时 Esc 键等同于 Alt＋Esc 键，使 Esc 失去原作用，给使用带来麻烦，影响用户软件的运行和坚固性。经分析 CCCC. COM 程序，发现处理这个功能的程序段如下：

1. 字输入时：

```
CS:9E9E  CMP   AH,01      ;是 Esc 或 Alt＋Esc 键吗?
   9EA1  JNZ   9EC0       ;不是,转9EC0
   9EA3  XOR   AL,AL      ;是,置 AL＝0
   9EA5  JMP   9D0C       ;9D0C 处理
```

2. 词组输入时：

```
CS:A757  CMP   AH,01      ;是 Esc 或 Alt＋Esc 键吗?
   A75A  JNZ   A767       ;不是,转 A767
   A75C  XOR   AL,AL      ;是,置 AL＝0
   A75E  JMP   A749       ;转 A749处理
```

从这两段程序可见，程序对 Esc 和 Alt＋Esc 键作同样处理。但我们希望 Alt＋Esc 键用于重选，而对 Esc 键不作特殊处理，保持原功能。为此，把这两段程序改为：

1. 字输入时：

```
CS:9E9E  CMP   AH,01      ;是 Esc 或 Alt＋Esc 键吗?
```

```
9EA1  JNZ  9EC0      ;不是,转9EC0
9EA3  JMP  AA28      ;是,转 AA28处理
```

2. 词组输入时:

```
CS:A757  CMP  AH,01  ;是 Esc 或 Alt+Esc 键吗?
A75A  JNZ  A767      ;不是,转 A767
A75C  JMP  AA32      ;是,转 AA32处理
```

为了区分 Esc 和 Alt+Esc 键,需要增补一小段程序,把这一小段程序接在 CCCC. COM 的最后(即分别从 AA28和 AA32开始),该段程序为:

```
CS:AA28  OR   AL,AL  ;是 Alt+Esc 键吗?
AA2A  JZ   AA2F      ;是,转 AA2F
AA2C  JMP  9EC0      ;不是(即是 Esc 键),转9EC0
AA2F  JMP  9D0C;     转9D0C 作重选处理
AA32  OR   AL,AL     ;是 Alt+Esc 键吗?
AA34  JZ   AA39      ;是,转 AA39
AA36  JMP  A767      ;Esc 键,转 A7667
AA39  JMP  A749      ;转 A749作重选处理
```

Esc 和 Alt+Esc 键虽然扫描码都是1(即 AH=01),但 Esc 键的 ASCII 码是1B(即 AL=1B),而 Alt+Esc 键的 ASCII 码是0(即 AL=0),AA28和 AA32就是据此区别它们的。

增补后把文件长度 CX 从 A928改为 A93C,然后存盘即可。虽然我们增加了文件长度,但 CCCC. COM 原驻留长度有裕度(为 AB07),因此,增补的程序能安全地驻留内存而不会被覆盖。

中文弹出式窗口设计

中国金融学院信息系　姜咏红

弹出式菜单、弹出式窗口,都是使人感觉良好的屏幕设计,但这些技术需要了解较多的显示原理和数据在机器内的存放方式,许多软件技术人员还不善于运用。特别是中文弹出式菜单和窗口,其边界可能要从某些汉字中间穿过,这又涉及到保留半个汉字和将来恢复全字的问题,弄不好就会驴唇不对马嘴。

本文给出弹出式窗口设计的最基本程序,它主要包括窗口区域原内容存储和恢复两个子程序,弹出的窗口是蓝颜色的,可以向其中输入字符和汉字。为了避免程序过长,没有作边界检查。作者相信,只要掌握了本文给出的基本方法,那么通向实用的弹出式窗口及菜单的道路是平坦的。

中文弹出式窗口设计的要点在哪里呢?为了说清这一问题,我们必须介绍一下汉字的显示问题。

采用长城高彩显卡或 CEGA 卡的微机一般都是在字符方式下显示汉字,将汉字作为全角字符处理。假设屏幕可以显示25行,80列,那么屏幕上的内容是内存的某些区域内容通过硬件线路的反映。这些区域称为显示缓冲区。显示缓冲区分为基本显示缓冲区和扩展显示缓冲区两部分。基本显示缓冲区地址规定为0B8000H,而扩展显示缓冲区的地址随显示卡而不同。CEGA 卡规定的地址是0B0000H 而长城高彩显卡规定的地址却是0C0000H。本文的程序是按后者设计的,因为使用后者的长城0520－CH 机的速度较慢,弹出式窗口或菜单的效果更加明显。

不论是半角还是全角字符都有它的显示码。显示码通常由 BIOS 自动生成,它由两个字节组成,低字节在基本显示缓冲区的偶地址,而高字节在扩展显示缓冲区的相同偏移地址处。高字节的第七位即位6是全角字符的标志,此位为1则显示的是全角字符,否则将显示一个半角字符。如果原来显示的是一个汉字,将位6改成0,则将显示这个汉字的左半部分(程序的61行就是完成这一功能的)。字符的属性在两个区各占一字节,规定它们必须紧接在显示码字节之后。读者不难看出在中西文显示方式下,一个半角字符在基本显示缓冲区占二个字节,而在扩展显示缓冲区也占两个字节,全角字符就应该占8个字节。由于全角字符的显示码也由两个字节组成,故显示汉字要四个字节就够了。对于全角字符,硬件规定后四个字节的内容可以任意。由于占位的需要,这四个字节是不能不要的。为了统一的记录作为全角字符的后四个字节,汉字 BIOS 规定在扩展显示区的显示码位写上1FH。这样只要查到这一位置是1FH,就知道包括前面两区内的四个字节都被某一全角字符使用。利用这一点我们可以查找汉字并对左边界上的汉字进行处理(程序的55－59行)。

窗口技术的最重要问题是对显示缓冲区窗口区域的内容保存和恢复,而左边界上的汉字处理正是弹出式汉字菜单和窗口的难点。左边界上的汉字在生成窗口时变成了只显示左半边,由于底层屏幕的内容是任意的,故这种半角汉字出现的位置完全是随机的。本文程序中使用了行位标记法,即每一行给一个标记字节(程序第2行定义的 S_LABEL),如果本行有左边界上的汉字,则送0FFH 给相应的字节,不然这个字节是0(程序的60行)。有了行位标记,在区域恢复时,就可以先查本行的标记(程序的99行),然后决定是否恢复全角字符(程序的101行),如此就能作到准确无误。

中文窗口技术可以不考虑右边界的汉字,由显示缓冲区的汉字存储方式,读者不难想出这是为什么。

本文给出的源程序是完整的,程序运行时敲“>”产生坐标为(X1,Y1),(X2,Y2)的矩形窗口;敲“<”则使原屏幕恢复,敲回车键结束。41－82行和83－120行的两子程序是通用性子程序,可以用到任何窗口设计程序中,如果用到 CEGA 卡上,只需将扩展显示缓冲区的地址改成0B0000H。两子程序开始一段都是根据屏幕位置计算内存的地址,了解显示与内存关系的读者一看便知。

```
1;DATA      SEGMENT
2;S-LABEL   DB  25  DUP(0)
3;X1        DB  10
4;Y1        DB  10
5;X2        DB  20
6;Y2        DB  70
7;BUFFER    DB  4000  DUP(?)
8;DATA      ENDS
9;STACK     SEGMENT  STACK  'STACK'
10;         DB  100H  DUP(?)
11;STACK    ENDS
12;CODE     SEGMENT
13;         ASSUME  CS;CODE,DS;DATA,ES;DATA
14;START    PROC  FAR
15;         PUSH  DS
16;         MOV  AX,0
17;         PUSH  AX
18;         MOV  AX,DATA
19;         MOV  DS,AX
20;         MOV  ES,AX
21;C00;     MOV  AH,1
22;         INT  21H
23;         CMP  AL,'>'
24;         JNZ  C01
25;         CALL  SAVE
26;         MOV  AH,2
27;         MOV  DH,X1
28;         ADD  DH,1
29;         MOV  DL,Y1
30;         INC  DL
31;         INT  10H
32;         JMP  C00
33;C01;     CMP  AL,'<'
34;         JNZ  C02
35;         CALL  RESTORE
36;         JMP  C00
37;C02;     CMP  AL,0DH
38;         JNZ  C00
39;         RET
40;START    ENDP
41;SAVE     PROC  NEAR
42;         MOV  DI,OFFSET  BUFFER
43;         PUSH  ES
44;         MOV  AX,0B800H
45;         MOV  ES,AX
46;         MOV  BL,X1
47;         MOV  DL,Y1
48;C-1      MOV  AL,160
49;         MUL  BL
50;         XOR  BH,BH
51;         XOR  DH,DH
52;         ADD  AX,DX
53;         ADD  AX,DX
54;         MOV  SI,AX
55;         MOV  AL,ES;[SI+8000H]
56;         CMP  AL,1FH
57;         JNZ  C-10
58;         CMP  DL,Y1
59;         JNZ  C-10
60;         MOV  S-LABEL [BX],0FFH
61;         AND  BYTE  PTR  ES;[SI+8000H-2],0BFH
62;C-10;     MOV  AX,ES;[SI]
63;         MOV  [DI],AX
64;         INC  DI
65;         INC  DI
66;         MOV  AX,ES;[SI+8000H]
67;         MOV  [DI],AX
68;         INC  DI
69;         INC  DI
70;         MOV  BYTE  PTR  ES;[SI],20H
71;         MOV  BYTE  PTR  ES;[SI+1],1FH
72;         MOV  BYTE  PTR  ES;[SI+8000H],00H
73;         INC  DL
74;         CMP  DL,Y2
75;         JBE  C-1
76;         MOV  DL,Y1
77;         INC  BL
78;         CMP  BL,X2
79;         JBE  C-1
80;         POP  ES
81;         RET
82;SAVE     ENDP
83;RESTORE  PROC  NEAR
84;         MOV  SI,OFFSET  BUFFER
85;         PUSH  ES
86;         MOV  AX,0B800H
87;         MOV  ES,AX
88;         MOV  BL,X1
89;         MOV  DL,Y1
90;C-01     MOV  AL,160
91;         MUL  BL
92;         XOR  BH,BH
93;         XOR  DH,DH
94;         ADD  AX,DX
95;         ADD  AX,DX
96;         MOV  DI,AX
97;         CMP  DL,Y1
98;         JNZ  C-010
99;         CMP  S-LABEL [BX],0FFH
100;        JNE  C-010
101;        OR  BYTE  PTR  ES;[DI+8000H-2],40H
102;        MOV  S-LABEL [BX],0
103;C-010   MOV  AX,[SI]
104;        MOV  ES;[DI],AX
105;        INC  SI
106;        INC  SI
107;        MOV  AX,[SI]
108;        MOV  ES;[DI+8000H],AX
109;        INC  SI
110;        INC  SI
111;        INC  DL
112;        CMP  DL,Y2
113;        JBE  C-01
114;        MOV  DL,Y1
115;        INC  BL
116;        CMP  BL,X2
117;        JBE  C-01
118;        POP  ES
119;        RET
120;        RESTORE  ENDP
121;CODE    ENDS
122;        END  START
```

一种实用新颖的排序方法

南京大学　朱国江

几乎所有的BASIC语言教程，在介绍名目繁多的排序方法时，其设计思想均采用设置数组、反复比较、不断交换的雷同模式。虽然它们各具特色，但都存在占用内存大、速度慢，无法区分并列名次等问题。

本文介绍的排序程序跳出了传统的框框，因而可以克服不同排序方法的缺点。以10个学生单科成绩排序为例，程序及运行结果如后。几点说明：

一、N为考生人数，M为满分。N、M可视实际情况灵活选用，但应和DATA语句中存储记录一致。

二、X记录不同考生分数，兼作数组A(X)，A$(X)的下标。其中A(X)=A(X)+1，统计考分为X的学生人数，是数值相加；A$(X)=A$(X)+X$，是存储相同成绩X、不同学号的组合，是字符串连接。例如本例中考99分的有二个人，学号分别为8807，8809，则经过10－60句处理后，A(X)＝A(99)＝2，A$(99)＝88078809。

三，设置不同条件语句，分别判断名次。70－140句的循环，打印排序后的学号、名次和成绩。根据条件语句满足与否，分别执行三个程序段。

·循环变量I控制分数，从满分120开始，只要A(I)＝0成立，就不要。这有两层意思：若存储单元A(I)中没有相应成绩的人数，则循环上去，否则执行J循环；若没有80句条件语句安排，则J循环(100－130句)无法执行。

·若相同成绩X的人数A(I)等于或大于2时，转向处理并列名次(160－210句)。

·当A(I)仅为1时，执行100－130句，其中C＝C＋1作为名次计数器用；而120句采用MID$函数，是将A$(I)中的字符分成4个一组的学号依次取出，由于此时A(I)＝1，因而J＝1，所以MID$(A$(I)，4＊(J－1)＋1，4)中，是从A$(I)的第一个字符开始，取四个字符出来，这正好是成绩为X的、仅为一个人的学号。采用MID$函数，省去交换学号的方法，能保证对应成绩的学号正确打印。

四，安排加、减计数器，巧妙处理并列名次。这由160－210句程序段解决。由于此时A(I)≠1，即相同成绩的人数不止1人，因而需要处理同名次问题，方法是在打印一个相同成绩的学号和成绩后，名次必须减1；而在完成所有相同成绩的同名次人员情况后，计数器应加1，并返回140句。

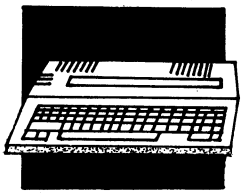
五、不受人员限制，占用内存少。本程序最大特点是占用内存少，且不受人员限制，这是因为DIM语句中仅开辟两个数组A(M)，A$(M)(下标值仅由M决定和人数无关。这就是说，当M一定时(如M＝120)，只占用内存240个，无论是300人，还是3000人，内存并未增加；而用冒泡法排序时，300人内存为720，3000人时内存为7200。

六、占用时机少，执行速度快。传统的排序程序，采用反复比较、不断交换的方法。比较、判断、交换是需要时间的，当信息量足够大时，这种矛盾尤为突出。本程序排序没有采用比较、交换的方法，因而速度特别快。如对一个3000人的排序，用冒泡法需时10多个小时，而本法不到4分钟。

七、修改循环变量初终值，速度还可以提高。例如，单科成绩最高分为100，则M取100；最低成绩为40分，则改70句为：FOR I＝100 TO 40 STEP－1，这样，由于循环次数大大减少，执行速度可进一步提高。

八、兼有成绩分档、录取划线和检索功能。这主要修改70句的循环语句来实现。例如，FOR I＝120 TO 100 STEP－1，可以将100－120分的成绩归挡；FOR I＝80 TO 80，可以检索80分的人员情况；FOR I＝120 TO 70，决定70分以上的考生数，作为录取划线的标准等等。而所有这些工作，必须在执行本程序一次后，直接RUN 70↙完成。

```
10  INPUT"n,m=";N,M
20  DIM A$(M),A(M)
30  FOR I=1 TO N:READ X$,X
50  A(X)=A(X)+1:A$(X)=A$(X)+X$
60  NEXT I
70  FOR I=M TO 0 STEP -1
80  IF A(I)=0 THEN 140
90  IF A(I)>=2 THEN 160
100  FOR J=1 TO A(I):C=C+1
120  PRINT MID$(A$(I),4*(J-1)+1,4),C,I
130  NEXT J
140  NEXT I
150  END
160  FOR J=1 TO A(I):C=C+1
180  PRINT MID$(A$(I),4*(J-1)+1,4),C,I
190  C=C-1:NEXT J
210  C=C+1:GOTO 140
230  DATA 8801,96,8802,98,8803,97,8804,98
240  DATA 8805,120,8806,98,8807,99
250  DATA 8808,98,8809,99,8810,95
RUN
n,m=? 10,120
8805          1          120
8807          2          99
8809          2          99
8802          3          98
8804          3          98
8806          3          98
8808          3          98
8803          4          97
8801          5          96
8810          6          95
```



学习机之友

多层次全能阴影菜单

四川攀枝花钢铁公司冶金工业学校　黄文贵

我们自编各种各样的程序时，通常都需要菜单，以选择所需的功能。根据程序的内容不同，菜单的项目、每项项目的字符个数也会随之不同；在同一程序中，有时会使用两个以上不同的菜单。阴影菜单是软件中常采用的一种方法，因调试工作较费时间，人们只好不用阴影菜单，而用1～9数字键来选择菜单项目。笔者编制了一菜单程序，基本上不做任何调试，就能适用于各种各样的模块化程序。能实现菜单的“双列显示”、“单列显示”、“多层次菜单”三大功能。现分别介绍如下：

一、双列显示

读者只需在11语句中输入“菜单标题”，根据自己所编程序的功能在15～21语句中输入菜单的各项名称。最后修改23语句，就能连接好自己所编的程序。去掉25语句，就能进入正常使用。

二、单列显示

对于7个以下的菜单项目，如果需要单列显示时，只需把10语句中的“双列显示”字样改为“单列显示”字样就行了。其它步骤与“一、双列显示”相同，请见500～515语句。

三、共同问题

(1)屏幕上显示的实际项目，由13语句中的Q值决定，故用户随时都可在2～14个项目之间任意选择，最多为14个项目。如果使用不完，也不必删去，保留下来，为以后增添用，请见503语句以及508～511语句中的Q$(8)～Q$(14)。

(2)为使菜单能显示在屏幕的中间位置，读者需修改12语句中的H值。这也是本程序中唯一需调试的一步，除此之外，读者不需要做任何调试，故此程序的通用性广、灵活性大。现提供一个“双列显示”的参考数字，每项目的字符数分别为4、6、8、10、12(每个汉字为2个字符，即为2～6个汉字)时，可分别取H＝8、7、6、5、4。“单列显示”的参考数字，H＝17－字符个数/2(即为H＝17－汉字个数)。

(3)菜单项目较少时，可在部分项目中输入空格，使项目之间有间隔，以达到屏幕显示上的美观。

四、多层次菜单

在同一个程序中，当需要两个以上不同的菜单时，可反复使用10～24语句，请见500～514语句。用23语句和513语句就能分别连接好第一部分程序和第二部分程序。25语句和515语句用作演示，正常使用时，去掉这两语句。读者在演示时，分别选择菜单一和菜单二中的“结束”项目，就能了解到多层菜单的特点。

五、共用程序

1000～1020语句是“多层次全能阴影菜单”主要部分，即两个以上不同菜单的共用程序。这一部分已调试好，不必修改。只需完成以上步骤，就能正常使用。

六、使用方法

运行本程序后，在菜单状态下，用户只需按动中华学习机键盘右侧的上、下、左、右四个方向键，移动阴影选择自己所需的功能，按回车键就能进入到相应的程序中。

```
5  PRINT CHR$(4);"PR#3":PRINT:DIM Q$(15)
10  Q$(0)="双列显示":REM Q$(0)="单列显示"
11  Q$(15)="菜单一":REM 菜单标题
12  H=6:REM 每个项目4－12个字符分别取H=8－4
13  Q=14:REM 实际菜单数目，最多为14个。
14  REM 14－20语句菜单项目，只修改内容不删除。
15  Q$(1)="查　　看":Q$(2)="修　　改"
16  Q$(3)="添　　加":Q$(4)="插　　入"
17  Q$(5)="删　　除":Q$(6)="建　　立"
18  Q$(7)="打　　印":Q$(8)="目　　录"
19  Q$(9)="存　　盘":Q$(10)="调　　盘"
20  Q$(11)="加　　锁":Q$(12)="解　　锁"
21  Q$(13)="说　　明":Q$(14)="结　　束"
22  GOSUB 1000
23  ON Q1 GOSUB 25,25,25,25,25,25,25,25,25,25,25,
    25,25,500
24  GOTO 22
25  HOME:VTAB 4:HTAB 3:PRINT"第一部分程序：";Q
    $(Q1);:GET K$:PRINT:RETURN
500  Q$(0)="单列显示"
501  Q$(15)="菜单二"
502  H=14
503  Q=7
505  Q$(1)="继续画":Q$(2)="重新画"
506  Q$(3)="图形表":Q$(4)="加汉字"
507  Q$(5)="存磁盘":Q$(6)="结　束"
508  Q$(7)="返　回":Q$(8)="目　录"
509  Q$(9)="存　盘":Q$(10)="调　盘"
510  Q$(11)="加　锁":Q$(12)="解　锁"
511  Q$(13)="说　明":Q$(14)="结　束"
512  GOSUB 1000
513  ON Q1 GOSUB 515,515,515,515,515,515,10
514  GOTO 512
515  HOME:VTAB 4:HTAB 3:PRINT "第二部分程序：";Q
    $(Q1):GET K$:PRINT:RETURN
1000  IF Q$(0)="单列显示" THEN Q2=Q:GOTO 1003
1001  Q2=INT(Q/2)
1002  IF Q/2<>Q2 THEN Q=Q+1:Q2=Q2+1:Q$(Q)=
      "　　　　":REM 空8格
1003  HGR2:VTAB 1:PRINT TAB(17－LEN(Q$(15))/3);
    Q$(15)
1004  FOR I=1 TO Q
1005  V=I－(I>Q2)*Q2:VTAB V+2
1006  HTAB H+(I>Q2)*14:PRINT Q$(I)
1007  NEXT I:VTAB 3:HTAB H
```

```
1008  PRINT CHR $ (15);Q $ (1);CHR $ (14)
1009  Q1=1:I=1
1010  K=PEEK(-16384)-128:IF K<0 THEN 1010
1011  V=I-(I>Q2) * Q2:VTAB V+2
1012  HTAB H+(I>Q2) * 14:PRINT Q $ (I)
1013  Q1=Q1+(K=10)-(K=11)+(K=21 AND Q1<Q)
      * Q2-(K=8 AND Q1>1) * Q2+(K=21 AND Q1>
      Q2)-(K=8 AND Q1<Q2+1)
1014  Q1=Q1+(Q1<1) * Q-(Q1>Q) * Q
1015  V=Q1-(Q1>Q2) * Q2:VTAB V+2
1016  HTAB H+(Q1>Q2) * 14
1017  PRINT CHR $ (15);Q $ (Q1);CHR $ (14)
1018  IF K=10 OR K=11 OR K=8 OR K=21 THEN I=Q1:
      GAME
1019  POKE -16368,0:IF K<>13 THEN 1010
1020  RETURN
```

虚　线

郑州煤田地质学院　刘　裕

虚线是机械制图中的常用线型。根据国家标准规定，虚线在起始与终端应是线段，不能是间隔；线段与间隔有一定比例关系且应各自大致相等。笔者认为，作为计算机绘图，线段与间隔应各自严格相等，而不能满足于手工绘图的大致相等。现行一些绘虚线程序及绘图机命令所绘虚线总不尽如意，有的会出现终端不是线段的情况，绘图机中的画虚线命令也有这种情况，如日产 MP－1000 型小平板绘图机就是如此；有的虽总能使终端是线段，但最后一线段不是稍长，就是稍短一些，很难保持各自相等；有的虽能按比例变化线段与间隔的长度，但最后一段总是不变，也很难保持各自相等。这种误差虽然很微小，但都不能算是完美的虚线，现存一些弊端给用户带来诸多不便。经过对虚线绘制过程的分析，笔者找到了一个关系式：

$$P_1^* = L\frac{P_1}{(N+1)P_1+NP_2}$$

$$P_2^* = L\frac{P_2}{(N+1)P_1+NP_2}$$

依此式编程绘虚线，可以圆满解决上述弊端，使绘虚线工作达到完美境界。

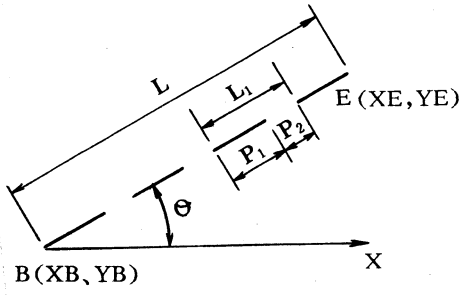


图　1

1. 程序设计

设　由始点 B 到终点 E 的虚线长为 L，与 X 轴正方向的倾角为 θ，线段部分 P_1 可称为实部，间隔 P_2 为虚部，那么 $L_1=P_1+P_2$ 可称为一个虚节，如图 1 所示。

则　$N=INT(L/L_1)$，N 为 L 中所含虚节 L_1 的整倍数。然后由(1)式求得的 P_1^* 与 P_2^* 就是绘图时所用的实部与虚部。再配合抬笔、落笔命令驱动画笔，即可绘出满意的虚线。其 BASIC 程序如下：

```
1400  INPUT "XB,YB,XE,YE=?";XB,YB,XE,YE
1403  P1=12:P2=8
1404  HGR :HCOLOR=3
1405  LX=XE-XB:LY=YE-YB
1410  L=INT(SQR (LX * LX+LY * LY)+0.5)
1415  SI=LY/L:CO=LX/L
1420  L1=P1+P2
1425  N= INT(L/L1)
1430  IF L <(2 * P1+P2) THEN PRINT "NO": GOTO 1495
1435  P1=L * P1/((N+1) * P1+N * P2)
1440   P2=L * P2/((N+1) * P1+N * P2)
1445  X1=P1 * CO:Y1=P1 * SI
1450  X2=P2 * CO:Y2=P2 * SI
1455  HPLOT XB,YB
1460  FOR I=1 TO N
1465  XB=INT (XB+X1+0.5):YB= INT (YB+Y1+0.5)
1470  HPLOT TO XB,YB
1475  XB= INT(XB+X2+0.5):YB= INT (YB+Y2+0.5)
1480  HPLOT XB,YB
1485  NEXT I
1490  HPLOT  TO XE,YE
1495  END
```

2. 程序说明：

(1)　1405～1415 行　由始点 B(XB,YB)与终点 E(XE,YE)的坐标，求出虚线长度 L 以及倾角 θ 的正弦和余弦值，使之能绘任意倾角的虚线；

(2)　1430 行　当 L 长内不够二个实部和一个虚部时，显示"NO"并结束，否则执行绘虚线工作；

(3)　1435～1440 行　计算出 P_1^* 与 P_2^*；

(4)　1445～1485 行　绘制虚线；

(5)　1490 行　绘制最后一实部；

(6)　该程序可直接在 APPLE Ⅱ 机上运行显示，在作为子程序时须删改程序的首部与尾部；

(7)　实用时须适当选取参数 P_1 与 P_2 的值；

(8)　该程序容易移植到其它机型上，也容易改写成适用于绘图机的程序。

3. 程序变型

机械制图中还有另一种虚线如图 2 所示，当虚线在实线的延长线上时，其起始与终端应是间隔，不能是线段。以往的程序虽也能绘出这种虚线，但要符合标准则是更加困难。本文所给程序稍加修改就能绘制这种两头虚的虚线，将相应语句行做如下改动即可：

```
1403  P1=8:P2=12
1470  HPLOT XB,YB
1480  HPLOT TO XB,YB
```

```
1490  HPLOT XE,YE
```

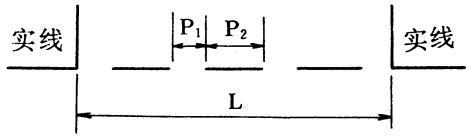


图 2

技巧两则

安徽铜陵一中　宋正荣

一、跟踪解密初探

有一些软件的程序加密很严格，不论用 CTRL—C 还是 CTRL—RESET 都不能使软件终止运行。CTRL—RESET 还可以导致冷启动、死机等现象。是不是没有办法呢？跟踪启动是一个很好方法，具体步骤如下：

1　先将驱动卡中的 BOOT0 移至内存中的 RAM 区。例 * 8600<C600、C6FFM↙

2　将磁盘放入驱动器中，并在监控状态输入 * 86F8：A9 00 8D 4A 08 4C 01 08↙ C600G↙

3　待光标出现输入 B747：4C BF 9D↙ B700G↙

现在已进入 DOS 状态，并且可以列印软件目录，调用程序等工作。

二、35 轨与 40 轨磁盘间的通信

《电子与电脑》杂志刊登过如何格式化 40 轨的磁盘。这样虽然容量变大了，但由于按常规方法将 35 轨盘上的文件转移到 40 轨的盘上较繁。使得格式化 40 轨盘好处并不太多。为此我编了个程序。使得这一转化过程十分简单。首先在监控状态下键入程序。用 BSAVE CHANGE，A $300，L $2E。然后用 BRUN CHANGE，A $××××即可。程序是浮动的。可以在任一 RAM 区内运行。

命令 &35↙是对 35 轨磁盘操作。&40↙是对 40 轨磁盘操作。

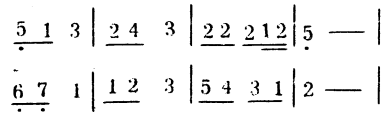
```
0300-  AD  73  AA  LDA  $AA73
0303-  8D  F7  03  STA  $03F7
0306-  A9  4C      LDA  #$4C
0308-  8D  F5  03  STA  $03F5
030B-  AD  72  AA  LDA  $AA72
030E-  18          CLC
030F-  69  15      ADC  #$15
0311-  8D  F6  03  STA  $03F6
0314-  60          RTS
0315-  20  F8  E6  JSR  $E6F8
0318-  A9  A0      LDA  #$A0
031A-  E0  28      CPX  #$28
031C-  F0  06      BEQ  $0324
031E-  E0  23      CPX  #$23
0320-  DB  0D      BNE  $032D
0322-  A9  8C      LDA  #$8C
0324-  8D  B5  AE  STA  $AEB5
0327-  8E  EF  B3  STX  $B3EF
032A-  8E  FE  BE  STX  $BEFE
032D-  60          RTS
```

改进 MUSIC 的发音

九江国棉二厂子弟学校初三班　熊波

在 CEC－Ⅰ型计算机上有一条音乐语句“MUSIC”，后跟两个参数 X，Y。但要给计算机编一段音乐程序，还需查表对照，得花很长一段时间，使用非常不便。我针对上述问题把 X，Y 两参数简化，X 参数的取值范围是 1－7，与音乐简谱相同。当遇上高音或低音，则在其后跟上一个 H 或 L。这样就形成了音阶参数。音长取值范围是 1－5，1 表示 1 拍，2 表示 2 拍，3 表示 1/4 拍，4 表示 4 拍，5 表示 1/2 拍，如此编起程序来非常方便直观。

例如



程序清单如下：

```
10  DIM D(15),E(5):FOR I=1 TO 15:READ D(I):NEXT I:DATA 192,171,152,140,128,114,102,95,84,75,68,62,255,228,205
14  FOR I=1 TO 5:READ E(I):NEXT I:DATA 110,160,30,255,70
15  READ G$,Q
17  IF G$="-1" THEN END
20  A$=RIGHT$(G$,1):B=VAL(G$):A=(A$="H")*(B+7)+(A$="L")*(B+8)+(A$<>"H" AND A$<>"L")*B
30  MUSIC D(A),E(Q):GOTO 15
40  DATA 5L,5,1,5,3,1,2,5,4,5,3,1,2,5,2,5,2,5,1,3,2,3,5L,2
50  DATA 6L,5,7L,5,1,1,1,5,2,5,3,1,5,5,4,5,3,5,1,5,2,2,-1,-1
```

中华学习机 RAM 监控及其应用

沈阳于洪区吴家荒小学　朴一兵

CEC－Ⅰ型中华学习机的 ROM 中固化了与 APPLE Ⅱ机兼容的监控程序，并有所增强，增加了跟踪命令 T 和单步执行命令 S，功能较强。但在一些场合就没那么方便了，如不能用监控命令解读 CEC－Ⅰ的汉字管理程序和 16K RAM 语言卡中的内容，其主要原因是 ROM 中固化的监控程序和上述程序在地址上的冲突。为此，本人将 ROM 中的监控移到主存 RAM 中，并进行了相应的修改，使其不但能显示内存单元的 16 进制内容，还能将其对应的字符显示出来，使用起来非常方便。

程序一　将 ROM 中的监控程序移到 $1800 开始的主存 RAM 中，并进行重新设定工作。使用时，将程序一输入完后，300G 即可进入 RAM 监控，为了和 ROM 监控区别，提示符改成%。

程序二　对主存 RAM 监控进行修改，使其将内存中对应的字符显示出来。程序输入完后，1700G 即可完成对监控程序的修改工作，然后 BSAVE RAM MON，A $1700，L $900 存盘，以后使用时只要 BRUN RAM MON 即可。

下面简单介绍一下 RAM 监控的应用。

1. 剖视 CEC－Ⅰ汉字管理程序

在 RAM 监控下，C3ABG 切换到辅存 ROM 中，然后就可以用监控命令反汇编汉字管理程序了。注意，因内存上的冲突，反汇编的程序不能打印。若要打印，可用拷贝屏幕的方法打印。

2. 解拆文件

LOCKSMITH V6.0 中有一很有用程序，即 BOOT TRACE，这个程序对调试及解密 6502 机器语言是一强大的工具，但每次使用都要启动 LOCKSMITH V6.0，很不方便，因 BOOT TRACE 是存在 16K RAM 卡上。卡上无监控程序，不能直接输入 C080 后读出来，把它解拆成单个文件还是比较麻烦的，但有了这一 RAM 监控后，问题就简单多了，方法如下：

BOOT 起 LOCKSMITH V6.0。

按 A 进入 BOOT TRACE。

输入 RAM 卡位置为 0，

CTRL－Y 进入监控程序。

6 CTRL－P 启动标准 DOS 3.3，

BRUN RAM MON。

C080↙ 3000＜D000.FFFFM↙，输入程序三，BSAVE BOOT TRACE，A $2FD0，L $3040 即可，以后使用时只要 BRUN BOOT TRACE。

3. 剖示 64K DOS

许多用户喜欢用 64K DOS 磁盘操作系统，如 DAVIE DOS、SUPER DOS V1.6 等，它把用户内存扩充了 10K 多，这对于内存小的八位机来说是一大突破。但现在提供的操作系统多少都有改进的余地，不少用户也在进行这方面的工作，但 64K DOS 是存在 16K RAM 卡上的，不能直接用系统提供的监控命令修改。利用本文介绍的 RAM 监控就非常方便了。方法很简单，BOOT 64K DOS，再 BRUN RAM MON，再 36:F0 1D 1B 1D↙，然后 C0 80↙ 就可以随心所意地用监控命令进行修改工作了。

程序一

```
0300- A9 00 85 3C 85 42 85 06
0308- A8 A9 F8 85 3D A9 FF 85
0310- 3E 85 3F 85 08 A9 18 85
0318- 43 85 07 20 2C FE A9 1F
0320- 85 09 A0 00 B1 06 C9 B9
0328- D0 31 A0 02 B1 06 C9 F8
0330- 90 04 E9 E0 91 06 A5 06
0338- C5 08 A5 07 E5 09 E6 06
0340- D0 02 E6 07 90 DC A9 1D
0348- 8D 54 AA 8D 56 AA A9 1E
0350- 8D BF 1F A9 A5 8D 6A 1F
0358- 4C 69 1F C9 BD F0 CB C9
0360- D9 F0 C7 C9 20 F0 C3 C9
0368- 4C D0 CB 4C 2A 03
```

程序二

```
1700- A9 4C 8D C3 1D A9 1A 8D
1708- C4 1D A9 17 8D C5 1D A9
1710- 1D 8D 54 AA 8D 56 AA 4C
1718- 69 1F 90 03 4C 28 17 A5
1720- 3C 29 07 F0 03 4C B6 1D
1728- 20 56 17 A9 1F 85 24 4C
1730- 3B 17 A5 3C 29 07 D0 03
1738- 4C B3 1D B1 3C 20 46 17
1740- 20 BA 1C 90 ED 60 C9 A0
1748- 90 02 25 32 84 35 48 20
1750- F0 1B 68 A4 35 60 A5 3C
1758- 29 07 D0 0D 18 A5 3C E9
1760- 07 85 3C 90 01 60 C6 3D
1768- 60 A5 3C 29 07 85 08 A5
1770- 3C E5 08 85 3C 60 4C 43
1778- 17
```

程序三

```
2FD0- 2C 81 C0 2C 81 C0 A9 00
2FD8- 85 3C 85 3E 85 42 A9 30
2FE0- 85 3D A9 60 85 3F A9 D0
2FE8- 85 43 A0 00 20 2C FE AD
2FF0- 80 C0 4C 00 E4
```

CEC— Ⅰ 键控光标作图程序

安徽蚌埠平板玻璃厂　陈国良

本程序具有方便的键控作图功能：八个方向的留轨迹及不留轨迹的光标运动作图；任意角度的直线、各种矩形、园及椭园、任意园心角的园弧及各种正多边形；作图过程中随时可更换选择颜色，且速度较快。

操作使用介绍：

(1)　U　I　O　　↖ ↑ ↗

　　J　K　　← →

　　N　M　，　　↙ ↓ ↘

分别控制光标移动的八个方向。

(2)按 0～7 八个数字键可在作图过程中随时改选色笔颜色。(由于高分辨彩色设置的限制，实际使用时可能会有部分颜色变异)。

(3)按 Q 键后：落笔，按 W 键后：抬笔。即可通过控制抬落笔选择来实现留轨迹及不留轨迹作图。

(4)画直线　在光标移到直线的起点时按 W 键抬笔，然后移动光标至直线终点后再按一下 F_1 键，即可画出一条直线。

(5)矩形　光标移到欲画矩形的任一角时按 W 键抬笔，按一下 F_2 键，然后移动光标至矩形的对角点再按一下 F_2 键即可画出一个矩形。

(6)园或椭园　光标移到园心处后按 W 键抬笔，移动光标后按一次 F_3 键定义 X 轴方向的半径，继续移动光标后再按一次 F_3 键则定义 Y 轴方向半径，此时即可画出椭园。如两次定义的半径相等即画出一个园。

(7)园弧　在作园过程中，按除 F_3 键以外的任意键，光标只移动而不留轨迹，再按 F_3 键则可控制留轨迹画弧。

(8)园(或椭园)内接正多边形　在定义园心坐标后按 F4 键，此时闪烁光标消失，可按 3～9 的数字键定义欲画多边形的边数(过多边数的多边形已近似园形，故本程序未予考虑)，接下来可按照画园(或椭园)的操作步骤进行即可，不过此时定义的是所画多边形外接园(或椭园)的半径。

(9)在整个作图过程中，可按 ESC 键退出本次作图回到原始状态以便从头开始作图。

```
10  FOR I=768 TO 778:READ J:POKE I,J:NEXT:SCALE=1:ROT=0:POKE 232,0:POKE 233,3
20  DATA 1,0,4,0,36,244,111,41,159,54,0
30  HGR2: K=-16384:L=-16368
40  X=139:Y=95:XX=X:YY=Y:D=1:C1=COS(.1):S1=SIN(.1):N=64:C=3
50  HCOLOR=C: HPLOT 0,0 TO 279,0 TO 279,191 TO 0,191 TO 0,0
60  P=PEEK (K):POKE L, 0:A = X:B=Y
70  X=X+(P=203 OR P=207 OR P=172)-(P=202 OR P=213 OR P=206):Y=Y+(P=205 OR P=206 OR P=172)-(P=201 OR P=213 OR P=207)
80  IF X<  >A OR Y<  >B THEN 170
90  IF P>=176 AND P<=183 THEN C = P-176:GOTO 60
100  IF P = 209 THEN D=1:GOTO 60
110  IF P=215 THEN D=0:XX=X: YY=Y:H=0: GOTO 60
120  IF P=129 THEN 220
130  IF P=151 THEN 250
140  IF P=140 THEN 200
150  IF P=148 THEN 230
160  IF P=155 THEN 30
170 X =X+(X<1)-(X>278):Y=Y+(Y<1)-(Y>190)
180  IF D=1 THEN HCOLOR=C:HPLOT X,Y
190  XDRAW 1 AT X,Y:FOR I=0 TO10:NEXT : XDRAW 1 AT X,Y: GOTO 60
200  R(H)=X:H=H+1:R(H)=Y:IF H<2 THEN H=H+1:GOTO 60
210  H=0:HCOLOR=C:HPLOT R(0),R(1) TO R(0),R(3) TO R(2),R(3) TO R(2),R(1) TO R(0) ,R(1):GOTO 60
220  HCOLOR=C:HPLOT XX,YY TO X,Y:XX=X:YY=Y:GOTO 60
230  GET A$:P=VAL (A$):IF P<3 THEN 230
240  C1=COS (6.2832/P):S1=SIN (6.2832/P):N=P:GOTO 60
250  R(H)=SQR ((X-XX)*(X-XX)+(Y-YY)*(Y-YY)):IF H =0 THEN H=1:GOTO 60
260  H=0:F=0:X1=1:Y1=0:HCOLOR=C
270  FOR I=0 TO N:T =X1*C1-Y1*S1:Y1=X1*S1+Y1*C1:X1=T
280  X=XX-.96 *Y1*R(0):Y=YY-X1*R(1)
290  IF X<1 OR X>278 OR Y<1 OR Y>190 THEN F=0:GOTO 350
300  P=PEEK (K): POKE L,0
310  IF P<  >151 AND P<  >23 THEN F=0:GOTO 340
320  IF F =0 THEN HPLOT X,Y:F =1
330  HPLOT TO X,Y: GOTO 350
340  XDRAW 1 AT X,Y: FOR J=0 TO 50:NEXT : XDRAW 1 AT X,Y
350  NEXT
360  HCOLOR=0:HPLOT XX,YY:HCOLOR=C:X = XX:Y = YY:C1 =COS(.1):S1=SIN (.1):N=64:GOTO 190
```

ProDOS 磁盘操作系统入门(续)

廖 凯

6. 灵活使用 INPUT

在程序内大量灵活地使用 GET 或 INPUT 语句是很困难的。用 INPUT 或 GET 从键盘得到的许多字符将不能正确操作。例如一个回车键能够破坏一个菜单的结构，它会清除右边的任何字符。下面的程序包含一组能单独或共同使用的输入子程序，它能够阻止输入到程序内的任何字符或字符串显示在屏幕上。经设置 H%和 V%的值，可以将一个字符定位在屏幕的任何位置。L 变量用于调节一个字符串的长度。利用这些子程序可以在任何条件下有效地监视所有键盘命令。

第 1 部分设置参数作为用户对"Enter"的反应。第 2 部分在屏幕上定位字符。第 3 部分在屏幕上显示命令。第 4 部分先打印一个反相字符串，然后将字符打印到屏幕。可以使用退回键删除字符。此部分中允许删除键在任何情况下正确操作。第 5 部分最重要，所有键盘输入都来自这部分，可以认为是改进的 GET 语句。

在下面的例子中，任何类型的串可以用大写或小写字母输入，数字值也可以用这些子程序输入。

第 1 部分

```
100  D$=CHR$(4)
110  PRINT D$;"PR#3"
120  B=1
130  H%=20:V%=10:GOSUB 1000:PRINT "Enter"
140  H%=30:L=10:GOSUB 1000:GOSUB 1250
```

第 2 部分

```
999  REM POSITION
1000  POKE 1403,H%:VTAB V%:RETURN
```

第 3 部分

```
1150  REM SCREEN A COMMAND
1155  C%=0:GOSUB 1500:C$=CHR$(C%)
1160  IF C%> 64 AND C%<91 THEN RETURN
1165  IF C%>96 AND C%<123 THEN C%=C%-32:RE-
      TURN
1170  IF C%=27 THEN RETURN
1175  IF C%>41 AND C%<58 THEN RETURN
1180  PRINT CHR$(7):GOTO 1150
```

第 4 部分

```
1250  REM INPUT A CHARACTER & USE DELETE
1260  INVERSE:FOR I=1 TO L:PRINT" ";:NEXT:B$=
      "":L=L+1:NORMAL
1300  FOR I=B TO L
1305   INVERSE: GOSUB 1500: H1%= H%: C $ = CHR
       $(C%):IF C$=CHR$(27) THEN RETURN
1310  IF C$=CHR$(13) THEN NORMAL:RETURN
1315  IF C$=CHR$(127) THEN 1355
1325   IF C$>=CHR$(8) AND C$ <=CHR$(21)
       THEN PRINT CHR$(7):GOTO 1305
1330  IF I=L THEN 1345
1335  GOSUB 1000:PRINT C$;:H%=H%+1:B$=B$
      +C$
1340  NEXT
1345  IF RIGHT$(B$,1)<>CHR$(13) THEN PRINT
      CHR$(7):GOTO 1305
1350  RETURN
1355  IF LEN(B$)=0 THEN I=1:PRINT CHR$(7):GO-
      TO 1300
1360  IF LEN(B$)=1 THEN B$="":C$="":I=I-1:
      H%=H%-1:GOSUB 1000:PRINT " ";:H%=H%
      -1:GOSUB 1000:B=1:GOTO 1300
1370  I=I-1:H%=H%-1:GOSUB 1000:PRINT " ";:
      GOSUB 1000:H%=H1%-1:B$=LEFT$(B$,I-
      1)
1380  GOTO 1305
```

第五部分

```
1500  REM GET A KEY
1510  C%=PEEK(-16384):IF C%<128 THEN 1500
1520  C%=C%-128:POKE 49168,0
1530  IF PEEK(-16287)>127 THEN C%=C%+128
1540  IF PEEK(-16286)>127 THEN C%=C%+128
1550  RETURN
```

GET A KEY

这个例子中最重要的是第 5 部分。行号 1510 读取由键盘打入的最后字符，如果已按过一个键，那么 PEEK(-16384)得到一个比 127 大的数；如果没有键按过，重复执行此行。行号 1520 确定哪个键被按过。例如 A 是由键盘最后输入的，PEEK(-16384)将等于 193，由 C% 减去 128 得到 65 或 A 的 ASCII 值，并将其值存到 C%内，而后用 POKE 49168,0 清除键盘。

POSITION

第 2 部分用于在屏幕上定位字符。POKE1403，H%是用于在 80 列显示方式下水平定位字符。HTAB 只能用在 40 列方式下。H%和 V%变量是用于设置屏幕位置。注意这部分是用于整个程序定位。

PROGRAM SET-UP

第 1 部分只说明了必须使用什么变量和什么格式。在行号 130 内，为通过用行号 1000 内的定位子程序而设置 H%和 V%。行号 140 安置用户的设置，并用 L 变量设置输入字符的长度，程序然后转到行号 1250 的字符输入子程序。

SCREEN A COMMAND

第 3 部分在这个例子中是不使用的，但对于在屏幕上显示最后按的键是很有用的。用户可以 GOSUB 这部分。行号 1155 设置 C%为零，从内存清除以前的键，程序并转到 GET A KEY 程序，计算机等候用户输入单键字符。在按过键后，其 ASCII 值被转为一个串变量，

作为以后使用。行号 1160 比较从 A 到 Z 的 ASCII 值，如果符合条件，子程序返回。另外，行号 1165 比较小写字母 A 到 Z 的 ASCII 值，如果符合条件，则从 C%减去 32 以获得大写字母的 ASCII 值。用这方法用户不必再检查大写和小写。行号 1170 检查 ESC 键，这是取消一个命令或返回主菜单的一个好方法。行号 1175 将检查数字键是否被按过。如果上面的条件有一个不符，那么行号 1180 将发出警告并重新执行这部分。

如果你想要使用箭头键，就需增加如下行号：

```
1176  IF C%=8 OR C%=10 OR C%=11 OR C%=21
      THEN RETURN
```

在子程序返回后，你必须用 POSITION 子程序重新定位。

DELETE AND PRINT

第 4 部分被分成五个部分：反相显示并打印一个空串，打印一个字符，允许正确的数个字符被打印，删除一个字符和检测回车键。

行号 1260 打印一个反相字符串。行号 1300 设置字符串的长度。行号 1330 将阻止打印任何额外的字符。行号 1335 根据每个 H%值打印 C$。行号 1315 检测 DELETE 键并执行删除操作。行号 1355 检查 B$ 内的字符是否是当前的。如果不是，删除则不成功，重新执行循环。行号 1360 检测 B$ 的长度，若符合条件就打印一个空格以有效地删除字符。所有其它字符的删除是在行号 1370。记住要按以下步骤删除一个字符：先退回，而后打一个空格，再退回。行号 1345 测试最右边的字符是否是回车。

7. 数组清除器

在你重新定义一个数组时，必须清除数组以获得更多的空间。这可以用下面的子程序来实现。

```
1   REM Array Eraser
10  DIM A(100),B(100),C(100)
100  FOR L=768 TO 823
110  READ N: POKE L,N
130  NEXT L
140  CALL 768,A:CALL 768,B
150  END
160  DATA 32,177,0,32,217,247,24,160,2,165,155,
     133,66,113,155
170  DATA 133,60,200,165,156,133,67,113,155,133,
     61,136,56,165
180  DATA 109,133,62,241,155,133,109,200,165,110,
     133,63,241
190  DATA 155,133,110,160,0,32,44,254,32,163,217,
     76,152,217
```

此程序是调用内存地址 768 开始的机器语言子程序来删除 A 和 B 数组。

三、屏幕输出

这部分将讨论可在屏幕上显示的不同输出类型。

1. 有关指令

HTAB　控制光标的水平位置(1 到 40)
VTAB　控制光标的垂直位置(1 到 24)
TAB(X)　将光标从右向左移到到 X 位置
POS　送回光标当前所在的水平位置
SPC(X)　将光标从上一次光标位置向右移动 X 空格
SPEED　设置字符输出的速度(0 到 255)
INT(X)　送回一个近似 X 又不大于 X 的最大整数

在屏幕上显示输出有许多方法。下表列出了控制文本显示窗口的指令。HTAB 只能用于 40 列显示，当用 80 列显示时，就要用 POKE 1430,H(H 是水平位置)代替 HTAB。对于 APPLE IIc 和具有 ROM Upgrade Kit 的 APPLE IIe，HTAB 已被扩展，可以用于 80 列显示。

指　令	控　制	范　围
POKE 32,L	L 是左界	在 0～39 之间
POKE 33,W	W 是宽度	在 1～40 之间
POKE 34,T	T 是上界	在 0～23 之间
POKE 35,B	B 是高度	在 1～24 之间
POKE 1403,H	H 是宽度	在 1～80 之间

2. 文字居中

下面的程序是将文字显示在 80 列屏幕的中央。

```
5   HOME
10  L$="An example of text centering"
20  GOSUB 1250
999  END
1000  POKE 1403,H:VTAB 12
1100  RETURN
1250  H=INT (40-(LEN(L$)/2))
1260  GOSUB 1000:PRINT L$
1270  RETURN
```

如果没有 80 列卡，就将行号 1250 改为：

```
1250  H=INT(20-(LEN(L$)/2))
```

3. 列印文字

下面的程序用 READ/DATA 语句将文字输出到屏幕上。

```
10  HOME:VTAB 24
20  FOR I=1 TO 3
30  READ A$:PRINT A$
40  POKE 69,255:CALL-856
50  IF I=2 THEN PRINT CHR$(13)
55  NEXT I
60  FOR I=1 TO 23:CALL-912
70  IF I=21 THEN FOR Z=1 TO 500:NEXT Z
80  FOR J=1 TO 100:NEXT J
90  NEXT I
99  END
100  DATA HELLO,This demonstration will show you a simple
     scrolling technique
110  DATA GOOD BYE
```

行号 10 将光标放在屏幕的底部。行号 40 产生一个延迟，POKE 69,X 设置延迟的长度，从 0 到 255，CALL-856 执行延迟。行号 50 在 I=2 时加入一个回车字符。行号 60 开始从底部向上列印，CALL-912 将屏幕上的所有文字向上移动一行。行号 80 只作为一个延迟，减缓列印速度。　(未完待续)

C 语言初阶讲座

第十讲　文件管理

李文兵

本讲介绍文件管理概念、标准输入输出、文件输入/输出、低级文件 I/O 函数、随机文件的存取。

1. 文件的概念

(1)文件系统及其功能　文件是指具有名字(文件名)的一组信息序列，系统和用户都可以将具有一定功能的程序模块、一组数据或文字命名为一个文件。一般的操作系统中都有专门负责文件管理的文件管理系统(filing system)。其基本功能有：

①能够建立新文件或删除旧文件；

②能对文件进行读写操作；

③能为文件自动分配物理空间，并建立逻辑结构与物理位置的映象关系；

④能对文件实施维护。

(2)文件结构　文件结构分逻辑结构和物理结构。

①文件的逻辑结构　它有两种形式：一是记录式文件，亦称标准文件；另一种是非结构的流式文件。

记录式文件是一种结构文件。例如，DOS 系统的文件结构是由文件、记录块、记录三要素组成的。其结构如图 10.1 所示：

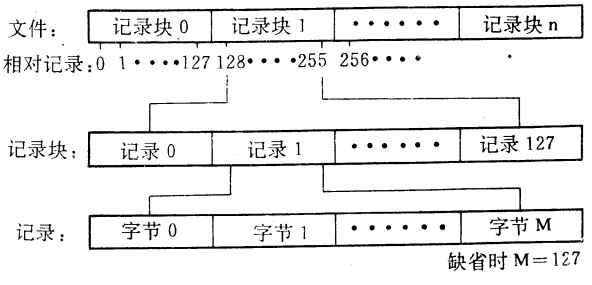


图 10.1　DOS 文件的逻辑结构

由该图可以看出：

• 一个 DOS 文件由若干个记录块组成。

• 一个记录块由 128 条记录组成。

• 一个记录由若干字节(缺省时为 128)组成。所有记录的长度都相等的文件，叫定长记录文件，其文件长度取决于记录数；记录长度不相等的文件，叫变长记录文件，其文件长度为各记录长度的累加和。

非结构流式文件是字符的有序集合。Turbo C 提供了这种文件逻辑结构形式。流式文件的长度是该文件所包含的字符数。这种文件，很便于操作系统进行管理；也适于用户进行文本处理，可不受任何约束地灵活组织其内部逻辑结构。

②文件的物理结构　它是指文件在外存上的存储结构，可分为连续、链接和索引三种组织方式。这种文件的读写方法有 2 种，即顺序读写和随机读写。

(3)C 语言的文件种类　C 语言中文件的分类如下：

文件 { 数据文件 { 流式文件(stream file)
　　　　　　　 { 标准文件(regular file)
　　 { 设备文件

在 C 语言中，I/O 设备可以当作文件来处理，这样的文件叫设备文件。设备文件既可按流式文件处理，又可按标准文件处理。常用的设备文件其名称、文件指针如表 10.1 所示。

表 10.1 常用设备文件

设　备	文件名	文件指针
键　盘	CON:/KYBD:	stdin
显示器	CON:/SCRN:	stdout/stderr
打印机	PRN:/LPT1:	stdout
串　行	AUX:/COM1:	

2. 标准 I/O

前面我们介绍了数据的按格式输入输出。这里以文件处理为中心，介绍 I/O 处理。

(1)标准 I/O　所谓标准输入输出是指在操作系统所认可的所谓标准 I/O 设备上进行的输入输出。把标准输入文件拷贝到标准输出上的程序，如练习 10.1 所示。该程序是利用标准输入函数 getchar()和标准输出函数 putchar()实现的。程序中的 EOF 是文件结尾符。

```
C>type exp10_1.c
#include <stdio.h>
main()
{ char c;
  while(( c=getchar())!=EOF)
          putchar(c);
}
C>exp10_1
学习 Turbo C,使用 Turbo C.
学习 Turbo C,使用 Turbo C.
^Z
C>
```

C 语言除了支持标准输入、标准输出外，还支持输入输出重定向，以及文件追加、管道功能。

(2)输入输出重定向　输入重定向的符号为<，其用法如下：

outf　<inf

功能是 outf 文件使用函数 getchar()、scanf()、gets(char *s)、cgets(char *s),从文件 inf 中读取数据,且实现 outf 的功能。

输出重定向的符号为>,其用法如下:

```
inf  >outf
```

功能是文件 inf 使用函数 putchar()、printf()、puts(char *s)、cputs(char *s)、把数据输出到文件 outf 中。

此外,< 和 >还可以同时使用,如:

```
prog <newin >newout
```

功能是文件 prog 从文件 newin 中读取数据,把文件 prog 的处理结果输出到文件 newout 中。

因为文件 exp10_1.C 中含有 putchar( )函数,故执行 exp10_1 >a.txt 输出重定向命令,便可把从标准输入设备(键盘)上输入的数据,通过程序中的函数 putchar(),输出到 a.txt 文件中;即改变了 putchar()函数的输出方向。如练习 10.2 所示。

```
C>exp10_1  >a.txt
学习 Turbo C,
使用 Turbo C.
C> type a.txt
学习 Turbo C,
使用 Turbo C.
^Z
```

同样,命令:

```
A>exp10_1 <a.txt >b.txt
```

改变了函数 getchar()的输入方向,从文件 a.txt 输入数据;同时亦改变了 putchar()的输出方向,把 exp10_1 要输出的数据输出到文件 b.txt 中。如练习 10.3 所示。

```
C>exp10_1<a.txt>b.txt
C>type b.txt
学习 Turbo C,
使用 Turbo C.
```

文件追加 其符号为 >>,用法如下:

```
fild >>output
```

功能是把文件 fild 的内容附加到文件 output 的原有内容后。例如:

```
A>exp10_1 <a.txt >>b.txt
```

执行结果是把文件 a.txt 中的数据,追加到文件 b.txt 的原有内容后,如练习 10.4 所示。

```
C>exp10_1  <a.txt  >>b.txt
c>type b.txt
学习 Turbo C,
使用 Turbo C.
学习 Turbo C.
使用 Turbo C.
```

(4)管道 其符号为|,用法如下:

```
inpr |outpr
```

功能是执行文件 inpr 和 outpr,且使 inpr 的标准输出作为 outpr 的标准输入,等效于:

```
inpr  > temp
outpr  < temp
del temp
```

文件 exp10_5.c 是对字符串计数的程序,执行命令:

```
C>exp10_1 <exp10_1.c |exp10_5
```

可以实现对文件 exp10_1.c 计数,如练习 10.5 所示。

```
C>type exp10_5.c
#include <stdio.h>
main()
  {  int i=0;
     char c;
     while ((c=getchar()) ! =EOF)
        if(c ! = '\n') i++;
     printf("total character=%d\n",i);
}
C>exp10_1 <exp10_1.c |exp10-5
total character=87
```

C 语言之所以能够实现 I/O 重定向、文件追加、管道等功能,就在于其输入输出函数能够重新分配标准输入输出设备,如练习 10.6 所示。

```
C>type exp10_6.c
#include <stdio.h>
#include "ctype.h"
main()
    { char c;
      while ((ch=getchar()) ! =EOF)
      {  if(ch==  '&')
           putc(ch,stderr);
         else
           putchar(ch)
      }
    }
C>exp10_6 >x.txt
abcdef
a&b&c&d
&&&
&&&
&&&
a b c d
^Z
c>type x.txt
abcdef
abcd
a b c d
```

由文件 exp10_6.c 和执行 type 命令的结果来看,标准输出(stdout)分配给文件 x.txt,标准诊断输出分配给显示终端,可见,根据程序的功能,可以把从键盘输入的一部分字符(这里为 &)分离出来,输出到显示终端上(兼作标准诊断输出)。注意,当把标准输入(stdin)分配给其他文件时,便不能从键盘进行输入操作;要想从键盘上输入,就必须把标准输入重新分配给键盘。

3. 缓冲型文件的输入输出

经常用到的使用缓冲区的文件输入输出函数如表10.2所示。

表10.2 使用缓冲区的文件I/O函数

函数名	功　能
fopen()	stream 的打开
fclose()	stream 的关闭
putc()	向 stream 输出一个字符
getc()	从 stream 输入一个字符
fseek()	查找 stream 的指定字节
fprintf()	输出到 stream
fscanf()	从 stream 输入
feof()	文件结尾(EOF)时为真
ferror()	发生错误时返回真
rewind()	把文件位置定位在文件的开头
remove()	删除文件

用如下命令行：

exp10_7 file1 file2

实现程序拷贝的程序如练习10.7所示。

```
C>type exp10_7.c
#include <stdio.h>
main(argc,argv)
int argc;
char *argv[];
  { FILE *from, *to;
    char ch;
    if(argc != 3)
    {  printf ("usage:exp10_7 from_file to_file\n");
       exit(1);
    }
if((from=fopen(argv[1],"rb"))==NULL)
       { printf ("can't open from_file\n");
         exit(1);
       }
if (( to=fopen(argv[2],"wb"))==NULL)
    { printf("can't open to_file\n");
      exit(1);
    }
while(! feof(from))
    { ch=getc(from);
      putc(ch,to);
    }
fclose(from);
fclose(to);
}
C>type a.txt
学习 Turbo C,
使用 Turbo C.
C>exp10_7  a.txt  exp.txt
C>type exp.txt
学习 Turbo C,
使用 Turbo C.
```

该程序的功能是把文件file1的内容拷贝到文件file2中。变量from、to是指向结构FILE的指针。因此，命令行第1、第2参数是文件名，分别用fopen()函数打开。若能打开，则返回指向结构的指针值；若打开失败，返回NULL的值。若两个命令行参数所指定的文件都打开成功，则用feof()函数检查EOF，便可进行文件之间内容的拷贝了。

函数getw()和putw()可用于整型数的输入输出，即输出用putw(n,file)，功能是把变量n的值输出到文件指针file所指定的文件中。输入用getw(file)，功能是从文件指针file所指出的文件中读取下一个数据。利用这两个函数进行整数输入输出的程序，如练习10.8所示。

```
C>type exp10_8.c
#include <stdio.h>
#define NMAX 20
main()
{ FILE *from, *to;
  char *fname="fint";
  int i;
  if((to=fopen(fname,"wb"))==NULL
     { printf("can't open to_file\n");
       exit(1);
     }
  for(i=0; i<NMAX; i++)
        putw(i,to);
  fclose(to);
  if((from=fopen(fname,"rb"))==NULL)
     { printf("can't open from_file\n");
       exit(1);
     }
  for(i=0; i<NAMX; i++)
        printf("%d ",getw(from));
  printf("\n");
  }
C>exp10_8
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

要随机存储文件，可使用能指定文件内任意位置的函数fseek()。该函数的用法如下所示：

```
int fseek(FILE *fp,long int n_byte,int org)
```

其中，fp为文件指针，打开文件时使用。n_byte为org所设定场所的字节数。org可使用如下宏：

SEEK_SET	文件的开头位置
SEEK_CUR	文件的现在位置
SEEK_END	文件的最末位置

使用函数fseek()进行文件随机存储的程序如练习10.9所示。设文件名从命令行引进。读文件使用函数fread()，其用法如下：

```
int fread(void *buf, int num_bytes,int count,FILE *fp)
```

功能是从文件指针fp所指定的文件中读取每字段为num_bytes个字节，一共count个字段的数据到指针buf所指定的缓冲区中。

```
C>TYPE EXP10_9.C
/*example for fseek(),fread()*/
#include <stdio.h>
#include "ctype.h"
#define SIZE 128
unsigned char buf[SIZE];
```

```
main(argc,argv)
int argc;
char *argv[];
{ FILE *fp;
    int sect,read_num;
    if(argc!=2)
    { printf("usage:exp10_9 filename\n");
        exit(1);
    }
    if((fp=fopen(argv[1],"rb"))==NULL)
    { printf("can't open file\n");
      exit(1);
    }
    do { printf("enter secter:  ");
        scanf("%ld",&sect);
        if(fseek(fp,sect*SIZE,SEEK_SET))
        { printf("fseek error\n");
          exit(1);
        }
        if ((read_num=fread(buf,1,SIZE,fp))!=SIZE)
        { printf("EOF reached\n");
          exit(1);
        }
    display(read_num);
    } while(sect>=0);
}
display(n)
int n;
{ int i,j;
  for(i=0;i<(n/16);i++)
     {for(j=0;j<16;j++)
     printf("%3x",buf[i*16+j]);
     printf("  ")
     for (j=0;j<16;j++)
       { if(isprint (buf[i*16+j]))
       printf("%c",buf[i*16+j]);
     else printf(".");
     }
     printf("\n");
     }
     }
```

```
A>exp10_9 exp10_9.obj
enter sector: 0
 80   b   0   9 45 58 50 31 30 5f 39 2e 43 15 88 1d  ....EXP10_9.C...
  0   0   0 19 54 43 38 36 20 42 6f 72 6c 61 6e 64  ....TC86 Borland
 20 54 75 72 62 6f 20 43 20 32 2e 30 20 fc 88 11   Turbo C 2.0 ...
  0   0 e9 fc 31 18 17   9 45 58 50 31 30 5f 39 2e  ....1...EXP10_9.
 43 c2 88   f   0   0 e9   1 52 dc 16   7 53 54 44 49 C.......R...STDI
 4f 2e 48 3b 88 10   0   0 e9 a0   8 8a   f   8 53 54 O.H;..........ST
 44 41 52 47 2e 48 fb 88   f   0   0 e9 a0   8 8a   f DARG.H..........
  7 43 54 59 50 45 2e 48 3d 88   6   0   0 e5   1   0 .CTYPE.H=.......
enter sector: 1
   0 8c 88   a   0   0 e3 19   0   2   0 15   8   4 4f 88  ..............O.
   a   0   0 e3 18   0   2   0 15 19   4 3f 88 15   0   0  ..........?....
  e6   4 61 72 67 76 18   a   6   0   4 61 72 67 67 63   4  ..argv.....argc.
   a   4   0 ee 88   6   0   0 e5   1   6   0 86 88 48   0  .............H.
   0 e2   0   5 6c 65 76 65 6c   4   0   5 66 6c 61 67  ....level...flag
  73   a   0   2 66 64   8   0   4 68 6f 6c 64   8   0   5 s...fd...hold...
 -62 73 69 7a 65   4   0   6 62 75 66 66 65 72 19   0 bsize...buffer..
   4 63 75 72 70 19   0   6 69 73 74 65 6d 70   a 80  .curp...istemp..
enter sector: ^C
```

在使用文件找开函数 fopen()时，用参数“rb”指定为只读打开二进制文件模式。

删除文件，使用如下函数：

```
int remove(char *filename);
```

其应用程序如练习10.10所示。

```
C>dir *.bak
EXP10_1 BAK     97     1-01-80      12:05a
EXP6_19 BAK     549    1-01-80      12:02a
EXP10_6 BAK     182    1-01-80      12:15a
      3 FILE(S)  12830720 bytes free
C> type exp10_10.c
    #include <stdio.h>
    #include "dos.h"
    main()
    { char fp[80];
    printf("enter remove filename:");
    gets(fp);
    remove(fp);
    }
C>exp10_10
enter remove filename:exp10_6.bak
C>dir *.bak
EXP10_1 BAK     97     1-01-80      12:05a
EXP6-19 BAK     549    1-01-80      12:02a
      2 File(s) 12832768 bytes free
```

4. 非缓冲型文件的输入输出

3中所介绍的文件输入输出函数，使用系统自动设

定的缓冲区。这里介绍向用户指定的缓冲区直接进行输入输出的函数。这样的输入输出叫无缓冲区的文件输入输出。这些函数叫低级输入输出函数，也叫操作系统连接的文件子程序。这些函数有：creat()、close()、open()、read()、write()、setmode()、rename()、unlink()。关于这些函数的功能、用法和返回值，请参考《IBM PC C 语言例题　习题　库函数》。

这些低级输入输出函数，使用如下标题文件：

```
io.h
fcntl.h
\sys\stat.h
```

文件打开的一般形式如下：

```
int open (char *filename,int access,[int permiss]);
```

其中 filename 为文件名；access 指定文件存取模式，取文件 fcntl.h 中所定义的任何一种，如图10.2所示：

```
/* The first three can only be set by open */
#define O_RDONLY      1
#define O_WRONLY      2
#define O_RDWR        4
/* Flag values for open only */
#define O_CREAT   0x0100 /* create and open file */
#define O_TRUNC   0x0200 /* open with truncation */
#define O_EXCL    0x0400 /* exclusive open */
/* a file in append mode may be written to only at its end。*/
#define O_APPEND      0x0800 /* to end of file */
/* MSDOS special bits *  /

#define O_CHANGED   0x1000 /* user may read these bits,
                              but */
#define O_DEVICE    0x2000 /* only RTL\io functions may
                              touch */
#define O_TEXT      0x4000 /* CR-LF translation */
#define O_BINARY    0x8000 /* no translation */
/* DOS 3.x options */
#define O_NOINHERIT     0x80
#define O_DENYALL       0x10
#define O_DENYWRITE     0x20
#define O_DENYREAD      0x30
#define O_DENYNONE      0x40
```

图10.2　文件存取模式

另外，当指定为 O_CREAT 时，permiss 可指定为如下存取模式当中一个，如图10.3所示。

```
#define S_IREAD     0x0100/* owner may read */
#define S_IWRITE    0x0080/* owner may write */
```

图10.3　文件的存取模式

或者指定为：

```
S-IREAD|IWRITE
```

open()函数的返回值为文件句柄，读、写或关闭时使用。关闭文件时，使用 close()函数：

```
int close(int fd);
```

文件读写时，分别使用 read()和 write()：

```
int read(int fd,void *buf,int size);
int write(int fd,void *buf,int size);
```

低级输入输出函数的例子如练习10.11所示。该程序的流程为：文件的生成、数据的写进、文件的关闭、文件以读方式打开、数据的读出、文件的关闭。

```
C>type exp10_11.c
#include <stdio.h>
#include "io.h"
#include "fcntl.h"
#include "sys\stat.h"
#include "string.h"
#define SIZE 80
#define ERR -1
main()
{ int i,fp;
char *fname="test.f";
char buff[SIZE];
if ((fp=creat(fname,S_IWRITE|S_IREAD))==ERR)
{ printf("can't create %s",fname);
  exit(1);
}
/* write to file */
printf("enter string-->\n");
do{ gets(buff);
    for(i=strlen(buff);i<SIZE;i++)
     buff[i]='\0';
if(write(fp,buff,SIZE)!=SIZE)
{ printf ("error on write ()\n");
  exit(1);
}
}while (strcmp(buff,"end"));
close (fp);
/* read from file */
if ((fp=open(fname,O_RDONLY))==ERR)
{ printf("error on read()\n");
  exit(1);
}
while(1)
{ if (read(fp buff,SIZE)==0) break;
    printf("%s\n",buff);
}
close(fp);
}
C>exp10_11
enter string-->
111111111111
11111111
111111
111
11
1
end
111111111111
11111111
111111
111
11
```

```
1
end
```

5、文件的随机存取

要想实现文件的随机存取，就得使用能使文件指针定位在任何位置的函数 lseek()，其用法如下：

```
long lseek(int handle,long offset,int fromwhere);
```

该函数把对应句柄值的文件指针从 fromwhere 位置移动 offset 个字节。其中，fromwhere 按如下取值：

SEEK_SET(0)　　文件的开头
SEEK_CUR(1)　　当前文件指针位置
SEEK_END(2)　　文件的末尾

函数的返回值为文件指针的新位置距文件开头的偏移量。要知道文件指针的当前位置，使用函数 tell()：

```
long tell (int handle);
```

使用这些函数的程序，如练习10.12所示。该程序根据输入的所要打开文件的扇区号，便可显示其内容。若 lseek 的返回值为－11，则意味着发生错误。该程序既能读出文本文件，也可以读出二进制文件。

```
C>type exp10_12.c
#include "io.h"
#include "fcntl.h"
#define SIZE 80
#define ERR -1
main ()
{ int i,fh,n;
  char buf[SIZE],s[80];
  long offset;
  printf("Enter filename:");
  gets(s);
  if ((fh =open(s,O_RDONLY))==ERR)
{ printf("can't open %s\n",s);
  exit(1);
}
while(1)
{ printf("Enter the sector number:");
  offset=(long)(atoi(gets(s)) * SIZE);
  if (lseek(fh,offset,SEEK_SET)==-11)
    { printf ("lseek() error\n");
      exit(1);
    }
  if((n =read(fh,buf,SIZE)) ==ERR)
  printf("read error\n");
  printf("current file pointer=%ld\n",tell(fh));
  for (i=offset;i<offset+SIZE;i++)
  { printf(%3x",buf[i]);
    if ((i%10)==9)
    printf ("\n");
  }
  if (eof(fh)==1) break;
  }
  close(fh);
}
```

```
C >exp10-12
Enter filename:exp10-12.obj
Enter the sector number:0

current file pointer=2
  a  a  0  0  0  0  0  0  0  0
ff86  0  0  0  0  0  0  0  0  0
  0  0  0  0  0 0ff87  0  0  0
  0  0  0  0  0  0  0  0  0  0
  0 0ff88  0  0  0  0  0  0  0
  0  0  0  0  0  0  0 0ff89  0
  0  0  0  0  0  0  0  0  0  0
  0  0  0 0ff8a  0  0  0  0  0
Enter the sector number:1

current file pointer=82
 31  0 70 31 30 5f 31 32 2e 6f
 62 6a  0  0  0  0  0  0  0  0
  0  0  0  0  0 0ff8c  0  0  0
  0  0  0  0  0  0  0  0  0  0
  0 0ff8d  0  0  0  0  0  0  0
  0  0  0  0  0  0  0 0ff8e  0
  0  0  0 0ffd4ffff 6d  5 10  0
  0  0  0 0ffdeffff 36  6 10  0
Enter the sector number:
```

从执行结果来看，文件指针的位置为80的整数倍。但是，实际输出的字节数，0扇区和2扇区是不同的，并不都是80个字节。原因是0AH 前面的0DH 给删除掉了。若把 open()说明为：

```
if(fh=open(s,o_RDONLY|O_BINARY))==ERR)
{ printf("can't open %s\n",s);
  exit(1);
}
```

则每个扇区的目标码便是80个字节，且上例中出现的0AH，也变成了0DH、0AH 了。

到本期止，C 语言初阶讲座已按计划刊出十讲。自讲座发表后，编辑部及作者都陆续收到许多读者的良好反映，为满足学习 C 语言的需要，本刊将于1992年内出版专辑，并请李文兵老师作重要补充，请有兴趣的读者留意。

本编辑部

学用单片机

SCB—1型单板单片机的存储器改进

江　琪

SCB—1型单板单片机以它的实用、简便、经济等优点受到广大用户的欢迎，更好地完善它，又是广大用户的共同心愿。笔者在分析了SCB—1型单板单片机的硬件以后，改进了它的存储器方面的设计，使之能够更加灵活地为用户服务。

一、问题的提出

我们知道，SCB—1型单板单片机具有8K EPROM和16K RAM，而对于有些用户来说，可能嫌8K ROM小，而16K RAM大。实际上，只要在加工该单板单片机时，稍加改动，就可解决这一问题，变8K EPROM、16K RAM为16K EPROM和8K RAM。这样一方面解决了上面所提出的问题，另一方面也使得用户可以在不改动原来监控程序的基础上，自己把执行程序固化在另一块EPROM上运行。

二、改进原理

下面把与改进有关的原SCB—1型单片单板机原理图摘抄如图1所示：

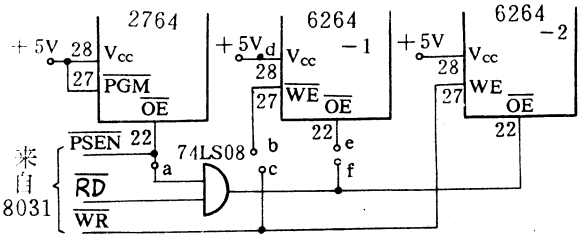


图1

由图1可知，2764、6264与8031的连接有以下不同：

1、2764的第22引脚 $\overline{OE}$ 为数据输出选通脚直接引至8031的 $\overline{PSEN}$ 引脚；而6264的第22引脚 $\overline{OE}$ 为读允许引脚是由8031的 $\overline{PSEN}$ 引脚和 $\overline{RD}$ 引脚经与门后引至的，这样做的目的是使6264既可以视作外部程序存储器，亦可视为外部数据存储器。

2、2764的第27引脚 $\overline{PGM}$ 需接到+5V，而6264的第27引脚为 $\overline{WE}$ 需接至8031的 $\overline{WR}$ 引脚。

显然，欲改变6264_{-1}、为2764只要将原第27引脚接至+5V，第22引脚接至8031的 $\overline{PSEN}$ 即可。这样，SCB—1型单板单片机的外部存储器容量即为16K，外部数据存储器容量为8K了。

三、改进方法

根据上面的分析可知，只需制板时在适当的位置留出几个插针，适当的插接插头就可以解决问题。如图2所示，可设置六个插针a、b、c、d、e和f。当系统为8K EPROM和16K RAM时，b、c短接，e、f短接。当系统为16K EPROM和8K RAM时，b、d短接，e、a短接。在线路

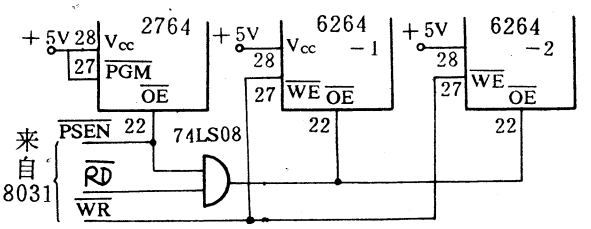
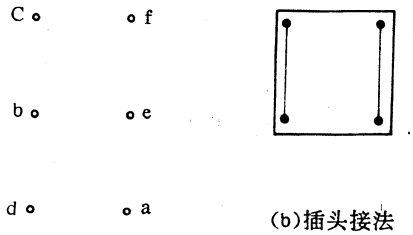


图2

板上可如图3所示按排插针和焊好插头。这样，当插头



(a)插针位置　　(b)插头接法

图3

插在上面位置时，系统为8K EPROM、16KRAM。如拆去6264_{-1}，插上2764，然后将插头插在下面位置时，系统就为16K EPROM和8K RAM了。

全国软件资格水平考试函授辅导班
（第四期）招生

为配合人事部，国务院电子办每年进行的全国计算机软件资格和水平考试，由中国软件行业协会考试指导中心主办，清华大学计算机系，北京大学计算机系等校专家，教授主讲。

主要通过《全国软件人员水平考试丛书》、《软件函授通讯》等多种教材、资料，指导学员围绕考试大纲进行自学，优惠供应配套的教学录相带。在学员集中省市设三十余个面授辅导站。9月初结业模拟考试，推荐优秀者参加全国软件人员（初级程序员级和程序员级）竞赛。

报名费5元，学费初级程序员级110元、程序员级120元、高级程序员级150元、系统分析员级160元（暑期集中一月面授另交费140元）。现在开始报名。简章函索。

邮汇：北京学院路31号，中国软件行业协会考试指导中心　邮编：100083　传真：2024674　电话：2012233—322

银行汇款：北京工商银行海淀区东升分理处62学信，账号：07187—28

MCS-51 单片机端口结构图示中的一个普遍错误

上海师大计算机科学系　王修才

(编者按:本文根据 P3 口的逻辑图的分析指出其逻辑图上的错误,希望有使用经验的读者就此发表一些看法。)

MCS-51 单片微机的应用适合於我国国情、对双增双节、产品的升级换代、机电一体化具有重大意义。但是,在我国已出版的有关 MCS-51 单片微机的教科书、资料中,在对 P3 端口的结构图示中有一个隐蔽性错误一直未被发现和纠正。

图 1 是国内出版的有关 MCS-51 单片微机资料对 P3 端口结构普遍采用的示意图。

按照 MCS-51 单片微机的规定,P3 口除了具有与 P1 口、P2 口同样的准双向功能(即作一般 I/O 口用)接口之外,还具有特殊的变异功能(有人称其为第二功能),P3 口各位的变异功能如下表所示。

按照 MCS-51 单片微机的规定,P3 口用作一般 I/O 口使用的条件是变异功能输出端自动置"1";P3 口用作变异功能使用的条件是位锁存器置"1"。但从图 1 所给出的端口结构来看,它只能实现用作一般 I/O 口使用,而无法实现全部变异功能的输入,况且要求变异功能的输入项(例如 RXD—P3.,$\overline{INT_0}$—P3.2,T_0—P3.4 等)从"读引脚"三态门的左边引入是错误的。按照常识,这些变异功能的输入项应该从它们各自的外引脚处输入;还有,图 1 中外引脚 P3.X 与"读引脚"三态门之间的一个门电路 G 在这里又有何用呢?为此,我查遍了国内出版的有关 MCS－51 单片微机的教科书和资料,均存在同样问题,如出一辙,最后追查到美国 Intel 公司出版的 MCS-51 单片微机用户手册——MCS－51 Family of Single Chip Micricomputer user's manual,它也是像图 1 所示那样画出的。我估计这是原文出错,因此就一错百错了。

P3 口变异功能表

I/O 引脚	变异功能
P3.0	RXD——半行数据接收端
P3.1	TXD——串行数据发送端
P3.2	$\overline{INT_0}$——外部中断 0 请求
P3.3	$\overline{INT_1}$——外部中断 1 请求
P3.4	T_0——定时器/计数器 0 的外部输入端
P3.5	T_1——定时器/计数器 1 的外部输入端
P3.6	$\overline{WR}$——外部数据存贮器写选通
P3.7	$\overline{RD}$——外部数据存贮器读选通

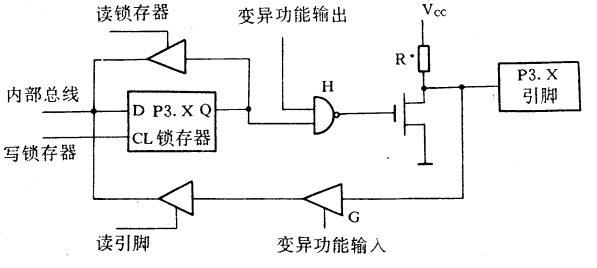


图 1

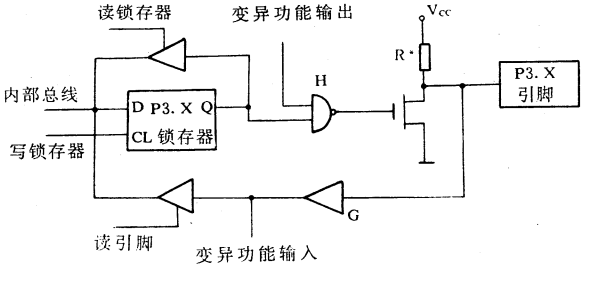


图 2

我认为 MCS－51 单片微机的 P3 端口的端口结构示图应当像图 2 所示那样才是正确的,能自圆其说的。

按照图 2,门电路 G 的作用也就清楚了。按照图 2,P3 口作为一般 I/O 口的使用条件是:变异功能的内部输入、输出端自动置"1",把与非门 H 和三态门 G 打开;P3 口用作变异功能使用条件是:相应位锁存器置"1",变异功能中的输出项从内部变异输出端送出,经与非门 H 和驱动器至外引脚处输出;变异功能项的输入项,在对三态门 G 控制端自动置"1"的条件下从相应外引脚处输入。

学装微电脑

串行 I/O 部件的制作与实验

易齐干

关于并行 I/O 8255A 的基本用法，在 μp80 套件中已经介绍过了。

这里介绍串行 I/O 芯片 8251A 的基本使用方法。8251A 也称 USART(Universal Synchronous/Asynchronus Receiver/Transmitter)通用同步/异步接收器/发送器。

串行传送的示意图如图 1 所示。即使用一条件数据线可将 8 位数据由 A 传到 B。

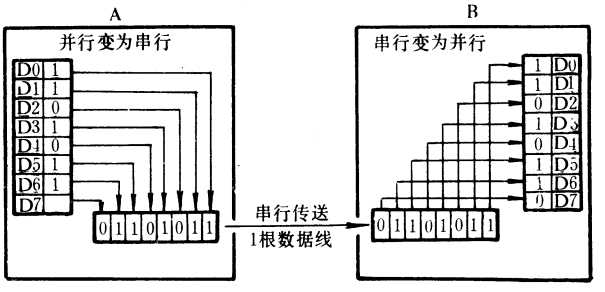


图 1　串行传送方式

下面结合 8251A 讨论如何实现非同步(异步)通讯的基本方法。图 2 给出 8251A 管脚说明。由於篇幅所限，各个管脚的作用不再叙述，读者可参阅有关书籍和资料。

1. 非同步(异步)通讯

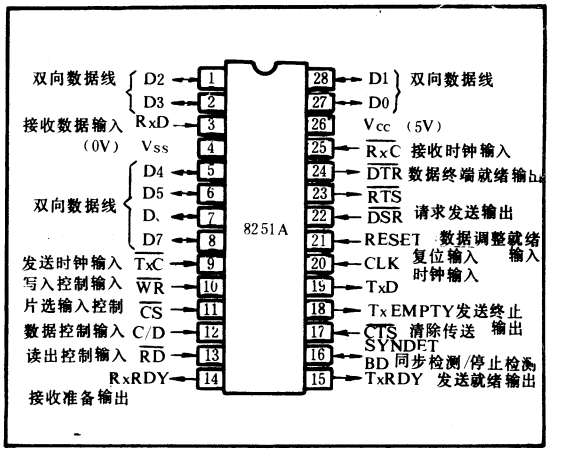


图 2　8251A 出脚

非同步(异步)通讯的数据有起始位、数据位、奇偶校验位和停止位所表示的字符所组成。是一个字符挨着一个字符地传送。有如下主要特征：

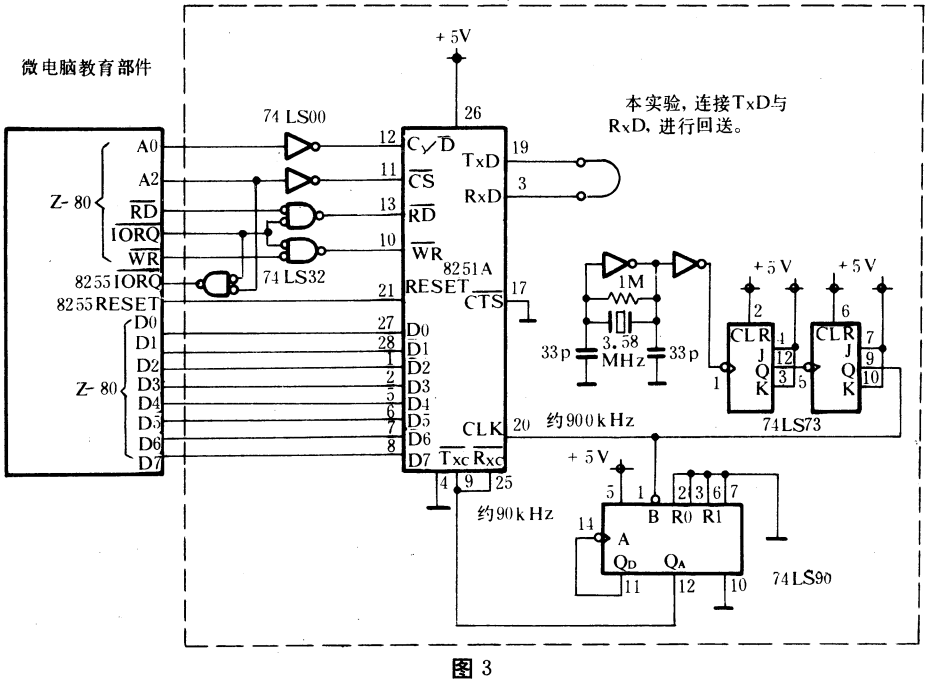


图 3

(1)根据方式指令码的设定传送 5～8 位数据。

(2)时钟速率为通讯波特率的 1、16、64 倍。波特率是数据传送速度。

(3)有 1、1.5 倍 2 个停止位。

(4)非同步方式波特率为 0～9.6K 波特。

2. 串行输入输出部件的制作。

串行输入输出部件的电路原理图如图 3 所示。时钟脉冲信号由 3.58MHE 石英振子和 CMOS4069 产生。经过 74LS73 的 1/4 分频变为 900KHz 输入给 8251A 的 CLK 出脚。再经过 74LS90 的 1/10 分频得到 90KHz 的脉冲输入给$\overline{R_{XC}}$、$\overline{T_{XC}}$出脚。

μp－80 套件的输入输出部件(8255A)A、B、C 各端口和 CW 寄存器的初始化地址分配为 00H－03H,则 8251A 的 C/$\overline{D}$(控制/数据)分配为 04H、05H。使用事例如下:

IN A,(04H)　表示 8251A 的工作状态的状态码送入 CPU 的累加器 A。

OUT(04H),A　表示 CPU 向 8251A 输出 8251A 的控制码。

IN A,(05H)　表示 8251A 接受的数据送入 CPU 的累加器 A。

OUT (05H),A　表示 CPU 的累加器 A 的数据由 8251A 向外部传送。

地址决定表如表 1 所示。

表 1　地址决定表

		地址 (2进制)								地址
		A7	A6	A5	A4	A3	A2	A1	A0	(16进制)
输入输出部件 (8255A)	A口	×	×	×	×	×	0	0	0	00H
	B口	×	×	×	×	×	0	0	1	01H
	C口	×	×	×	×	×	0	1	0	02H
	CW寄存器	×	×	×	×	×	0	1	1	03H
8251A	控制	×	×	×	×	×	1	×	0	04H
	数据	×	×	×	×	×	1	×	1	05H

3. 串行传送实验。

设计一个封闭反馈实验,开关的 ON/OFF 状态送入 μp－80 套件的输入输出部件 A 口,输入输出部件的 C 口连接 H/L 电平显示部件,8251A 串行传送、接收开关状态信号,如图 4 所示。

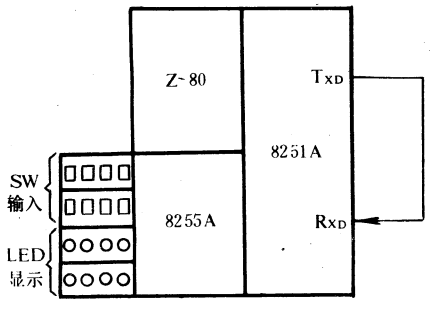
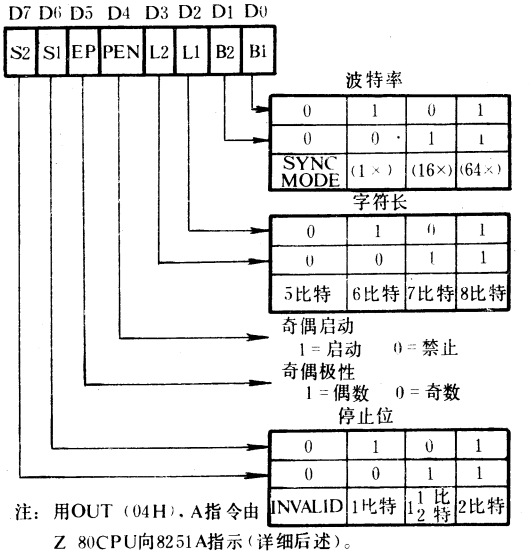


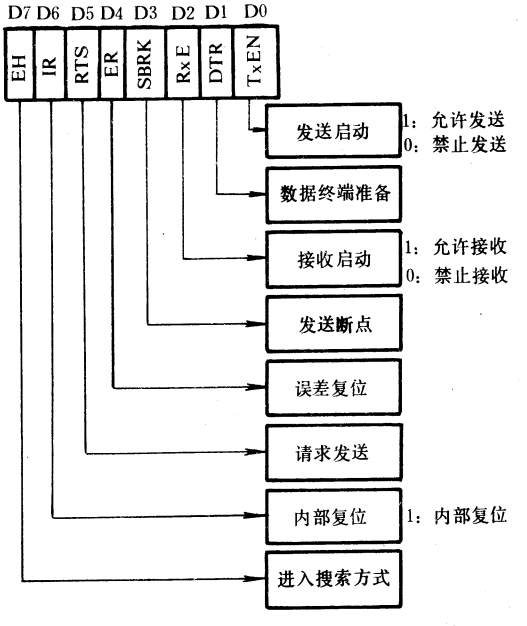
图 4　封闭反馈实验示意图

8251A 是可编程 LSI 集成电路。使用时首先要进行初始化,决定异步方式码和指令字格式。分别如图 5、图 6 所示。它们均为控制码。如前所述,应该送入 I/O 地址 04H 中。



注：用OUT (04H)，A指令由 Z 80CPU向8251A指示(详细后述)。

图 5　方式码



注:(1)本实验中,仅使用D6, D2, D0。其他详细说明请参照文献(4)－(6)。

(2) 用OUT (04H), A指令由Z 80CPU向8251A指示

图 6　指令字格式

图 6 中，取 $D_6=1$ 为内部复位，其它位均为 0，这样得到指令字格式为

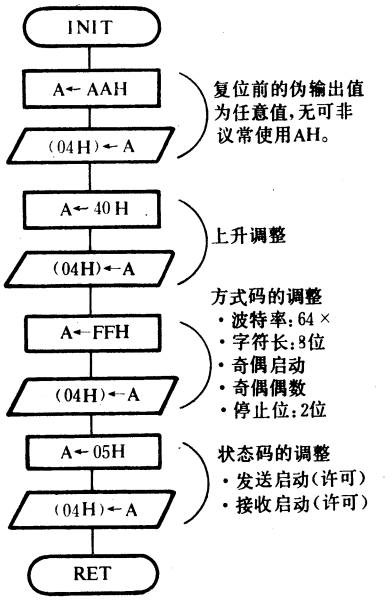
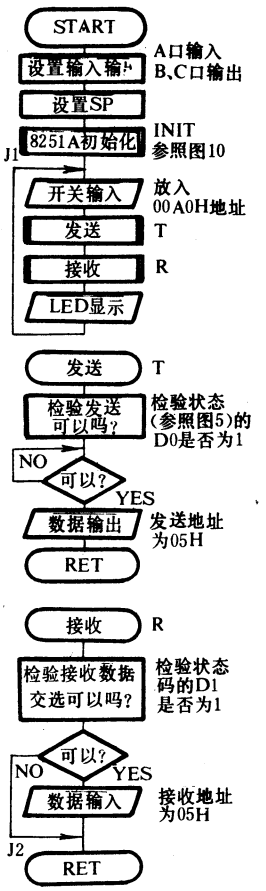


图 7　8251A 初始化流程图

40H。异步方式码为 FFH。为使 8251A 能传送和接收数据，指令字格式 $D_0=1$、$D_2=1$，其它位均为 0，指令字格式为 05H，应放入 I/O 的 04H 地址中。

封闭反馈实验流程图和程序清单如图 7、图 8 所示。它包括 8251A 的初始化、开关输入、传送数据、接收数据和 H/L 电平显示子程序。

执行这个程序，与开关 ON/OFF 状态相对应，H/L 电平显示中的发光二极管应该亮。



● RAM区域

00A0H	00A1H
输入开关的“通”“断”状态	存放接收数据

标记	助记符	地址	机械语	注　释
START	LDA, 90HH	0000	3E 90	设置输入输出
	OUT(03H), A	0002	D3 03	
	LD SP, 0100H	0004	31 00 01	设置SP
	CALL INIT	0007	CD 48 00	8251A初始化
J1	LD HL, 00A0H	000A	21 A0 00	
	IN A, (02H)	000D	DB 00	开关输入
	LD (HL), A	000F	77	
	CALL T	0010	CD 60 00	发送
	INC HL	0013	23	
	CALL R	0014	CD 70 00	接收
	LD A, (HL)	0017	7E	
	OUT(02H), A	0018	D3 02	LED显示
	JP J1	001A	C3 0A 00	
INIT	LD A, AAH	0048	3E AA	
	OUT(04H), A	004A	D3 04	
	LD A, (04H)	004C	3E 40	
	OUT(04H), A	004E	D3 04	
	LD A, FFH	0050	3E FF	
	OUT(04H), A	0052	D3 04	
	LD A, (05H)	0054	3E 05	
	OUT(04H), A	0056	D3 04	
	RET	0058	C9	
T	IN A, (04H)	0060	DB 04	
	RRC A	0062	CB 0F	
	JP NC, T	0064	D2 60 00	
	LD A, (HL)	0067	7E	
	OUT(05H), A	0068	D3 65	
	RET	006A	C9	
R	IN A, (04H)	0070	DB 04	
	RRC A	0072	CB 0F	
	RRC A	0074	CB 0F	
	JP NC, J2	0076	D2 7C 00	
	IN A, (05H)	0079	DB 05	
	LD (HL), A	007B	77	
J2	RET	007C	C9	

图 8　8251 封闭反馈实验程序

“微电脑控制技术”函授学习班总结

易齐干

本刊与天津纺织工学院共同开办结合 μp80 微电脑的“微电脑控制技术”函授学习班第一期圆满结束了。全国各地学员 81 人。其中多数学员结合本职工作带着实践中的课题参加学习。

这期以电子工业出版社发行的《微电脑图解教程》为实验教材。接着本刊 90 年 9 期刊登的教学大纲向学员发寄讲义，讲义密切配合“教程”一书所讲述的硬件，重点讲述汇编语言程序的编制技巧；常用的子程序；子程序的组合；特殊程序的编制以及应用实例。

学员按照计划，认真阅读教材，边实验，边思考，由浅入深循序渐进，将疑难问题及时与函授教员通信联系，均得到满意解答。

最后，进行了开卷考试。学员成绩均为优良，天津纺院成人教育处颁发了单科结业证。

这期函授班学员普遍反映效果很好。讲授内容通俗易懂，结合实际，它作为一座桥梁将微电脑爱好者由入门引进到应用阶段。

（转 35 页）

将 MP-1 电脑的时基信号改为晶体振荡

邓文超

MP－1 电脑简单易装，可广泛应用于各种自动控制场合之中。但若用于定时控制，则显得时间精度稍差一些，原因是 MP－1 电脑的时基信号是由简单的 RC 电路产生，(见图 1)其振荡频率随环境的温度、湿度而变化。笔者取 MP－1 的一位输出作为数字电子钟的 50Hz 时基信号，日误差大于 40 秒。

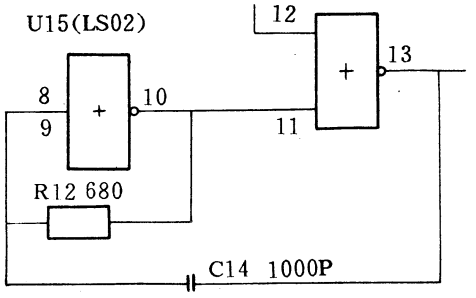


图 1

为了提高精度，笔者将 MP－1 改为晶体振荡，使 MP－1 的频率稳定性与普通计算机相仿。(见图 2)。

改装步骤如下：

1. 按照图 2，准备一只 1MHz 的石英晶体(MP－1 原设计工作频率为 500kHz，因 500KHz 的晶体价昂难觅，故改用 1MHz，查资料知 6502 CPU 的工作频率可为 1MHz)；0.01μ 涤纶电容一只；1kΩ 金属膜电阻二只。

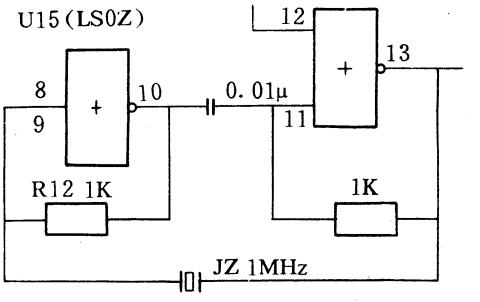


图 2

2. 用刀片割断连接 U15(LS02)第 10 脚与第 11 脚之间的印刷线路(在插元件面)。

3. 焊下 C_{14}，并在该位置焊上 1MHz 晶体。

4. 将 R_{12}的值换为 1k。

5. 在电路版的焊接面直接将一个 0.01μ 的电容跨焊在 U15(LS02)的第 10 脚与第 11 脚之间。

6. 以同样方式焊接一只 1k 电阻在第 11 脚与第 13 脚之间。

只要元件完好，焊接无误，完成以上六点工作之后，MP－1 就变成为工作频率非常之精确稳定的单板电脑了(无须调式)。

邮售中华学习机通用 EPROM 读写卡

南京华山电子技术服务部向读者提供本利 1990 年 8 期介绍的通用 EPROM 读写卡，该卡有以下特点：

• 功能齐全：可读、写或复写 27 系列 2K 至 32K 字节的各种型号 EPROM 芯片；扩展其它系列芯片也极方便。

• 操作简便：有详尽的汉字提示和过程显示；可不关主机更换操作芯片，便于连续读写大规模软件；可从任意地址开始读写任意字节数据，便于分段固化大容量芯片。大批量固化 EPROM 芯片只需一次建立工作参数后即可连续运行。

• 功能扩展：在上述基本型读写卡的基础上已可作三项扩展(详见下表)，读者可任选 π 项扩展功能。

邮售价格表：

序号	品种	主要功能说明	套件价(元)	成品价(元)
1	基本型	见 90 年 8 期文章介绍	110	120
2	扩展 I	增加自带监控系统，开机用 PR#n 命令联机运行。	35	
3	扩展 II	增加自备编程电源、减少外接电源的麻烦。	18	
4	扩展 III	增加专用 EPROM 读写插座(活洛带锁紧手柄)	65	

南京华山电子服务部　地址：南京市小营路 2 号　邮码：210016　电话：403872　开户行：南京市工行后宰门分理处　帐号：1686007　联系人：陈万伟

第二章　FBASIC的基本语句

于　春

上一章介绍了几个初步的FBASIC语句，从本章开始，陆续介绍FBASIC中用的较多的，最基本的语句。

一、循环语句

FOR～TO～STEP——NEXT

1. 循环语句的结构

循环语句由FOR、NEXT两句组成。FOR语句是循环的起点，是循环的说明语句，由它决定循环次数。NEXT语句是循环的终端语句。FOR与NEXT语句必须成对出现。

FOR语句的形式为

FOR i=L TO m STEP S

i：循环控制变量，可用任一简单变量表示。

L：循环控制变量初值。

m：循环控制变量终值。

S：循环变量每一次循环的增量，也叫做步长。步长为1，可以省略STEP S。

例1. 打印1～30整数的平方、立方表。

```
10 FOR A=1 TO 30 STEP 1
20 PRINT A,A*A,A*A*A
30 NEXT
```

上程序中10句也可写为FOR A=0 TO 30。

步长也可以为负值。当步长为负值时，终值应小于初值。

例2.

```
10 FOR X=20 TO 1 STEP -2
20 PRINT X,
30 NEXT
```

例3. 求5的阶乘5!

```
10 A=1
20 FOR B=1 TO 5
30 A=A*B
40 NEXT
50 PRINT A
```

RUN后 显示120

循环变量的初值、终值、步长必须控制在－32768～32767之间取值。否则，程序不能输入，且屏幕显示OV ERROR溢出提示。

F BASIC允许输入简写语句，简写输入后，计算机会自动翻译成原语句。如FOR—NEXT语句可简写为F. ～TO～ST. ——N. 。注意，简写中字母后的点不能丢失。为节约篇幅，以后介绍的语句和举例均用简写。读者练习时，在弄通语句的结构后，建议使用简写输入程序，这样，可以提高程序的编写、输入速度。同样LIST的简写为L. PRINT的为P. 或?；END的为E. 等。

例4. 求前n项和 $\sum_{n=1}^{100} n$。

```
10 C=0
20 F.  D=1 TO 100
30 C=C+D
40 N.
50 P. C
```

RUN后，显示5050

2. 多重循环

以上介绍的是单循环，F BASIC允许两重以上的多重循环嵌套。

例5. 打印九九乘法表。

```
10 F. A=1 TO 9
20 F. B=1 TO A
30 C=A*B
40 P. B"*"A"="C,
50 N.
60 N.
```

例5也可以写为

```
10 F. A=1 TO 9:F. B=1 TO A
20 P. B"*"A"="B*A
30 N. :N.
```

二、转移语句

计算机在执行程序时，一般是按行号由小到大依次进行，但由于解决问题的需要，往往要求改变执行顺序，转到其它行号执行，这一现象叫转移或叫转向。实现这一功能的语句叫转移语句。转移语句有三种：无条件转移语句(GOTO)；条件转移语句(IF——THEN)；开关转移语句(ON——GOTO)。下面分别介绍。

1. 无条件转移语句

GOTO 简写 G.

GOTO语句的格式为

G.＜行号＞

当程序执行到这一语句时，把程序的运行无条件的转移到GOTO语句中规定的行号去，并从这一行号起继续执行。

2. 条件转移语句

IF——THEN　　简写 IF——T.

条件转移语句也叫 IF 语句，使用该语句进行条件判断，决定某种功能。

IF 语句的格式为

IF＜关系式＞THEN＜行号＞

IF＜关系式＞GOTO＜行号＞

IF＜关系式＞THEN GOTO＜行号＞

IF＜关系式＞THEN＜语句＞

值得说明的是：关系式后可以用 THEN，也可以用 GOTO，也可以用 THEN GOTO，效果是一样的。头三种格式可以任用。

IF 语句的作用是：若关系式成立，就转到 THEN、GOTO 后的行号继续执行或执行 THEN 后面的语句；若关系式不成立，就跳过 IF 语句，继续执行后面的语句。

例 6. 用转移语句求前 n 项和 $\sum_{n=1}^{100} n$

```
10 A=B=0
20 A=A+1 : B=B+A
30 IF A<100 T. 20
40 P. "B=" B
50 E.
```

RUN 后，显示 5050

程序中 10 行对 A，B 赋以初值，20 行进行连加运算，30 行为条件判断。当执行到 30 句时，计算机进行判断 A 是否小于 100，若小于 100 则转到 20 行继续进行加法运算。若 A 等于 100，关系式 A＜100 不成立，则跳过 30 行，执行 40 行。30 行关系式后若换用 GOTO 20 或 THEN GOTO 20 效果一样，读者可自己验证。

例 6　还可以改为如下程序

```
10 A=B=0
20 A=A+1 : B=B+A
30 IF A=100 T. P. "B="B : E.
40 G. 20
```

30 行中的条件关系式改为 A=100，它表示若 A=100 时，则执行 THEN 后的语句；若 A≠100 则跳过 30 行，执行 40 行。

通过例 6 可以发现，程序的编制是很灵活的，解决同一个问题，可以编制不同的程序来解决。

由于循环语句、转移语句是程序设计中最常用，最有用的语句，也是最难接受，最易出错的语句，因此，本文通过较多的示例介绍这两语句的配合使用。

例 7. 有一算式，式中每一种动物代表一个阿拉伯数字，试求出各代表的数字。

算式如下：

马蛇×马马＋牛牛＝1991

设 A＝马、B＝蛇、C＝牛，程序如下：

```
10 F. A=0 TO 9 : F. B=0 TO 9 : F. C=0 TO 9
20 IF(A * 10+B) * (A * 11)+C * 11=1991
   T. P. "Ma="A,"She="B,"Niu="C : E.
30 N. : N. : N.
```

RUN 后显示

Ma＝4　she＝3　Niu＝9

THEN 后可以跟多个语句。程序执行完第一个语句再执行第二个语句……直到执行完所有语句再执行下一行。20 行中，THEN 后的第二语句是 END，所以程序执行到此就停机。

上程序只求出了例 7 的一组解，可能还有其它解，这只要将 20 行的 END 语句移到最后一行（添 40 行），则可求出全部解。结果如下：

Ma＝4　she＝3　Niu＝9

Ma＝4　she＝4　Niu＝5

Ma＝4　she＝5　Niu＝1

例 8. 民间趣题

用 100 匹马驮 100 块瓦，大马驮 3 块，中马驮两块，两个小马驮一块，求大、中、小马各多少？

这是一个多解问题，先列出求解方程

设大马为 X，中马为 Y，小马为 Z 匹，则可列出方程组：

$$\begin{cases} X+Y+Z=100 \\ 3X+2Y+\frac{1}{2}Z=100 \end{cases}$$

整理得

$$\begin{cases} X+Y+Z=100 \\ 6X+4Y+Z=200 \end{cases}$$

这道题用代数方法求解，需用试商法，比较麻烦。若用计算机求解，则比较简单。程序设计思路是：先设定单用一种马驮瓦最多用多少匹，决定循环变量终值，则大马最多用 34 匹；中马最多用 50 匹；小马用 100 匹（因出题限定 100 匹）。程序如下：

```
10 F. X=1 TO 33 : F. Y=1 TO 50 : F. Z=1 TO
   100
20 IF X+Y+Z=100 T. 40
30 G. 70
40 IF 6 * X+4 * Y+Z=200 T. 60
50 G. 70
60 P. "X="X,"Y="Y,"Z="Z
70 N. : N. : N.
80 E.
```

RUN

X＝2　Y＝30　Z＝68

X＝5　Y＝25　Z＝70

X＝8　Y＝20　Z＝72　……

本题共有 6 组解。

由于程序中有三重循环嵌套，程序执行速度较慢。若把 Z 的循环变量初、终值改为

F. Z=60 TO 80

速度要快一倍。（由于小马 60 匹才驮 30 块，下余 40 匹马，70 块瓦，若全部用中马驮，也仅用 35 匹，所以小马最少不能低于 60 匹。同理，80 匹'小马驮'40 驮 40 块瓦，下余 20 匹马 60 块瓦，全部用大马刚好驮完，所以

小马最多不超过 80 匹。)

例 9. 山上一群羊,山下一群羊,若山上拿一只羊给山下,上、下羊数相等;若山下拿一只羊给山上,则山上羊数是山下的两倍,山上山下各有多少只羊?

由于山上、山下羊数有一只波动,就产生倍级的量变,所以山上山下羊数各不会超过 10 只。设山上有 A 只,山下有 B 只。

```
10 F. A=1 TO 10 : F. B=1 TO 10
20 IF A-1=B+1 AND A+1=(B-1) * 2 T. P. "Shan Shang=" A, "Shan Xia=" B
30 N. : N.
40 E.
RUN
Shan Shang=7      Shan Xia=5
```

20 行使用了逻辑判断式 AND,它表示两关系式相与。即若两关系式都成立,则执行 THEN 后的语句;若有一个不成立,则跳过 20 行而执行 30 行。使用逻辑判断式,可使程序简捷。大家可以发现例 8 的程序可以改编如下:

```
10 F. X=1 TO 20 : F. Y=1 TO 30 : F. Z=60 TO 80
20 IF X+Y+Z=100 AND 6 * X+4 * Y+Z=200 T. P. "X=" X, "Y=" Y, "Z=" Z
30 N. : N. : N.
40 E.
```

例 10 《电子与电脑》1989 年 9 期曾刊登了美国物理学家爱因斯坦做过的数学趣题,有一阶梯(1)每步跨 2 阶,剩 1 阶;(2)每步跨 3 阶,剩 2 阶;(3)每步跨 5 阶,剩 4 阶;(4)每步跨 6 阶,剩 5 阶;(5)每步跨 7 阶,剩 0 阶;问阶梯至少应有多少阶?

原文程序用了 18 行,本文编程如下:

```
10 F. N=0 TO 500 ST. 7
20 IF N MOD 2=1 AND N MOD 3=2 AND N MOD 5=4 AND N MOD 6=5 T. P. N : E.
30 N.
RUN      119
```

程序中使用了 MOD 运算,即求余运算,关系式中只判断余数如何而不管商是多少。

例 11. 民间趣题

兄弟三人打鱼,老大把打的鱼分成三等分,最后剩了一条,由于不好分,老大把这条鱼扔进了河里,拿了一份回家。老二不知道老大已分了鱼,就把剩下的两份鱼合在一起,又三等分,结果也剩了一条,他也扔进了河里,拿走了一份。老三也不知道已分鱼,他又把剩下的两份合在一起又三等分,又剩了一条,又扔进了河里。问最少有多少条鱼才能满足以上分法。

该题比较复杂,得数必须满足除以 3 余 1 商乘以 2 之积除以 3 余 1……。该题虽有无数个解,但用代数求解也比较麻烦。设总数为 A,程序如下:(设定循环终值为 100)

```
10 F. A=1 TO 100
20 B=A/3 : C=B/3
30 IF A MOD3=1 AND 2 * B MOD 3=1 AND 2 * C MOD 3=1 T. P. "A=" A : E.
40 N.
RUN A=25
```

若求 100 内的解,只须把 30 行未的 END 删掉即可求得 25,52,79。若求 1000 内的解,只须把循环终值改为 1000。

在 IF 语句中,THEN 后面的语句也可以是 IF 语句,这称为 IF 语句的嵌套,因此上例 30 行也可以写为如下形式:

```
30 IF A MOD 3=1 T. IF 2 * B MOD3=1 T. IF 2 * C MOD 3=1 T. P. "A=" A,
```

两种形式,效果一样,读者自己验证。在使用 IF 语句嵌套时,IF、THEN 必须成对出现。

例 12. 民间趣题,留给读者自己编制程序。

龟、蟾、蟹共 19 只,100 条腿进行分配,求各有多少只?(注:龟四条腿,蟾四条腿,蟹十条腿。)

3. 开关转移语句

ON——GOTO　　简写　　O. ——G.

开关转移语句起分支转移的作用,它控制转向,随变量的值不同而分别执行不同的功能。开关语句的格式为

ON<表达式>GOTO<行号 1>、<行号 2>……

ON<表达式>GOSUB<行号 1>、<行号 2>……

ON<表达式>RETURN<行号 1>、<行号 2>……

ON<表达式>RESTORE<行号 1>、<行号 2>……

在一般程序设计中,开关,语句用的不多,本文简要介绍其应用的方便之处。

在举例前,先介绍一个常用的语句——键盘输入语句 INPUT 简简写 I.

使用 INPUT 语句,可以不把数据编在程序里,而在程序执行时,临时从键盘输入。当程序运行时,计算机执行到 INPUT 语句,自动停止运行,打出"?"等待用户输入数据。输入数据后,再继续运行程序。下面结合开关语句示例介绍 INPUT 语句的使用。

例 13　分段统计学生成绩,分不及格、60～69、70～79、80～89、90～99、100 六档。

使用以上所学语句,编程序如下:

```
A. 回车(自动提供行号)
10 A=B=C=0 : D=E=F=0
20 I. X
30 IF X>100  T. 100
40 IF X<60 T. A=A+1 : G. 20
50 IF X>=60 AND X<70 T. B=B+1 : G. 20
60 IF X>=70 AND X<80 T. C=C+1 : G. 20
70 IF X>=80 AND X<90 T. D=D+1 : G. 20
80 IF X>=90 AND X<100 T. E=E+1 : G. 20
```

```
90 IF X=100 T. F=F+1 : G. 20
100 P. "<60="A,"60～69="B,"70～79="C,
"80～89="D,"90～99="E,"100="F
110 E.
```

按 Ctrl+C 键(解除自动提供行号)

程序开始前 A. 为自动提供行号语句 AUTO 的简写。当程序较长时,每输入一行要键入一次行号很麻烦,为提高编程速度,先键入 AUTO 语句,回车后则显示 10 行号,该行编完回车后,自动另起一行,显示 20 行号……。程序编写完毕,同时按 ctrl+c 键可解除自编行号功能。

上程序中,20 行为键盘输入语句,程序运行到此自动停止运行,等待输入数据。30 行为键盘输入结束语句,若键入大于 100 的任何数,程序运行结束,打印结果。

上程序 RUN 后,显示?号后跟闪动光标,输入一个得分回车后,则另起一行出现? 等待再次输入,直到输入大于 100 的数,立即打印各档成绩人数。

若使用开关转移语句,例 13 的程序要简捷的多。

A.

```
10 A=B=C=0 : D=E=F=0
20 I. X
30 IF X>100 T. 110
40 O. X/10+1 G. 50,50,50,50,50,50,60,70,
   80,90,100
50 A=A+1 : G. 20
60 B=B+1 : G. 20
70 C=C+1 : G. 20
80 D=D+1 : G. 20
90 E=E+1 : G. 20
100 F=F+1 : G. 20
110 P. "<60="A,"60～69="B,"70～79="C,
    "80～89="D,"90～99="E,"100="F
120 E.
```

程序中 40 行为开关语句,X/10+1 是为了使得分从 0～59 变化时,关系式值为 1～6,均转 50 行;当得分为 60 时,关系式值为 7,则转 60 行,余者依此类推。

至于开关语句的其它形式,留待以后示例中附带介绍。

(未完待续)

(接 30 页)

现将函授过程中学员来信摘录如下:

广西柳州赵克明学员讲:"……μp80 已组装完毕,其原理也已搞清。而且,我还给它扩充了内存。学习之余有个强烈的念头——给它装备一个 CRT 控制器,让它在电视机或在显示器上输出信息。"

苏州某乡办企业赵斌学员讲"……我们乡办厂搞微电脑应用多为外面引进人材,而本地没有很好的继承人,作为本乡人,我想自己具有这方面特长会有远大前景的。通过函授学习,增加了一门才干,不仅对个人,而且对自己家乡的经济、科技都有益。……"

很多学员共同反映,在学习过程中,对照实验教材参照讲义,一块一块焊接 up80 微电脑套件;一条线一个焊点地检查;发现问题及时修复,μp80 微电脑不知不觉地工作起来时,那种高兴劲儿无法形容。

现在第二期函授学习班正在进行。我们也预祝第二期学员每人都能取得丰硕果实。将微电脑控制技术得心应手地用于本职工作上。在祖国的四化建设中大显身手。

读者询问有关"微电脑控制技术"函授学习与 μp80 微电脑套件的事宜请与天津纺织工学院机械系高殿斌同志联系。邮政编码 300160 电话 41.2833 转 279 电报挂号:天津 3251。

本刊"学装微电脑"栏目介绍 μp－80 微电脑教育套件及其应用。颇受广大非电专业工程技术人员、微电脑爱好者的欢迎。纷纷来信询问 μP-80 套件有关情况与教材。现给予答复:

教材为我社发行的"初级微电脑图解教程"售价:2.9 元/册,邮资 0.40 元。邮购者请与"北京 173 信箱"电子与电脑"编室联系。编码 100036。

μp－80 教育套件是"教程"一书配套实验器材。分为三大部份。

电源部件售价 35 元

基础篇部件:21 块,是学习数字电路的理想器材,售价 130 元。

实践篇部件:6 块,组装成 Z80 单板机,使用汇编语言结合基础篇部件可进行简单的应用研究。

应用篇部件:11 块,包括光、声、温度传感器部件和继电器、DC 电机、压电蜂鸣器、扬声器和 7 段数码管等驱动元件,与实践篇组合进行应用控制研究。售价 140 元。

上述套件可选购、可只购印刷电路板、8 线插接件等,方式灵活、服务周到。

邮购事宜请与天津纺织工学院机械系高殿斌同志联系,编码 300160。

维修经验谈

计算机电源修理

黄培鑫

计算机电源常采用效率高、稳定性好、重量轻、体积小的开关稳压电源，由于计算机不带图纸，常给维修带来极大不便。图1是根据某一种计算机开关电源实物整理出的电路图。图2是开关电源中排风扇的电路图。下面就图1图2的维修作一介绍，供读者参考。

第一部分　开关电源的修理

图1是一种隔离式并联型开关电源。其工作原理简单叙述如下：220V交流电经0.5A保险丝BX、保险电阻R_1、扼流圈T_1送到D_1桥整流块的交流输入端，D_1整流后输出直流经L_2、L_3扼流圈再经C_6、C_7滤波，产生约300V的高压。R_3对约300V的电压降压后提供给Q_2基极正偏，Q_2开始导通，集电极回路形成电流。随着Q_2导通程度的加深，IC_2不断增大。当达到一定值时，$T_2$②－①绕组中感应出脉冲电压使Q2基极反偏，Q_2由导通变成截止。从而使反馈绕组脉冲极性发生颠倒，Q_2重新获得R_3提供的正偏再次开始导通，重复节一过程，形成间歇振荡。在Q_2导通期间，次级绕组感应出的脉冲电压，经D6～10半波整流，电感$L_{5\sim7}$、电解$C^*_{19\sim21}$的滤波后，输出±12V、±5V四组低压直流电，作为计算机主机、接口电路。及驱动器的工作电源。＋5V和＋12V还通过$R_{22\sim26}$组成的取样电路，取出电压的变化量送到IC_1输入端。IC_1将输入的变化量进行放大，并产生一脉冲信号，经Q_3放大，通过隔离变压器T_3推动Q_1，来改变Q_{1E-C}之间的内阻，进而改变Q_2基极负偏压的大小，来改变Q_2导通、截止的周期。如负载减轻，次级电压升高，IC_1输入端取样电压升高，IC_1输出脉宽变窄，Q_3集电极电流增大，Q_1基极正偏时间增长，使Q_2导通时间缩短，T_2中储能减少，次级电压降低。一般开关电源的振荡频率为20KHZ。(可用示波器测量获得)。$R_{9\sim12}$、D_4组成过流保护电路。一旦Q_2集电极电流异常增大时，V_{e2}也增大，$R_{11,12}$分压值也增大，使Q_{2E-C}内阻迅速减小，从而迫使Q_1退出导通区，保护了开关调整管Q_2。D_{14}、R_{27}、CR_1组成次级低压保护电路，一旦＋5V由于故障升高至5、6V时，CR_1单向可控硅导通，＋12V对地短路，迫使Q2停振，保护了计算机的安全。

故障及修理

如发现彩色显示器不显示(显示器工作正常)和键盘指示灯不亮时，其大多是开关电源发生了故障，应立急关机。将计算机上盖打开，取出电源盒。然后将电源盒盖镙钉拧下，将电源盒盖拆下。先采用直观法和电阻测量法检查，然后运用电压测量法检查。修理仪表为一只500型万用表和一只示波器(没有条件可以省去)。

一、直观、电阻测量检查法

电源盖拆下后，首先检查BX(0.5A)保险丝管是否损坏、检查有无异常元器件。如BX损坏应更换同规格熔丝管。从而可以判断故障可能在高频振荡部分。用万用表X1K挡测量交流电源输入两端的正反向电阻应均为350K左右，说明D_1之前元件完好，无短路现象。再测量C_6正极C_7负极两端正向电阻为5K左右，反向电阻为500K左右，说明D_1整流、$C_{6,7}$滤波等元件正常。在检查中如发现异常元件，应拆下更换同规格元件，并要根据元件的位置，找出故障原因。证实高频振荡部分基本正常后，方可通电进行电压测量法的检查。

二、电压测量检查法

下面主要介绍三种故障现象的修理方法。

1)无电压

接上电源后，测量次级四组低压全无。用万用表直流500V挡，正极红棒接C_6正极，负极黑棒接C_7负极，应有300V左右高压，说明$C_{6,7}$滤波之前电路正常。黑棒位置不变，红棒改接Q_2集电极。测Q_{2-C}极如无300V左右电压则是R_2开路损坏而造成。如有300V左右电压，则测量Q_2基极和发射极电压。如均无电压，则是R_3损坏。断开200V交流电，用一导线同时接触C_6正极和C_7负极两端，将高压短路放电，以免修理时被高压击伤。用烙铁拆下R_3一端，万用表×10K挡测量R_3，R_3阻值为∞，从而造成Q_2无起始偏置，使电路无法工作。更换电阻，电路可恢复工作。当测量基极有0.9V左右电压，而发射极无电压。这种现象一般是Q_{2C-E}极开路损坏或R_{11}损坏或R_{12}、D_4同时开路损坏造成的。更换D_4如手头无同型号二极管，可选用$I_0=1A$，$V_R\geqslant700V$的高频二极管。Q_2可选用$B_{VCeO}\geqslant1500V$，$P_{CM}=50W$，$I_{CM}=5A$，$H_{FE}=5$($I_C=2.5A$时)，T_{O-3}金属封装管代替。造成Q损坏的原因一是Q_1损坏，二是C_8无容量而失效，使Q_2一直导通，I_C越来越大最后损坏。在更换Q_2后通电之前，应对Q_1和C_8进行检查，以免Q_2再次损坏。

在通电后，能听到“嗒嗒……声”，测量Q_{2-E}极电压，会发现电压随着“嗒嗒”声而来回摆动。这种现象通常是Q_3损坏或是Q_3发射极回路元件损坏，如L_8导。

2)通电几秒钟后次级无电压

用万用表测量四组低压。接通电流时电压正常，接着电压逐渐上升。＋5V升高到5、6V时，电压突然跌落。从现象看，是次级低压保护电路已工作所造成电压跌落。保护电路工作很正常，关键是要找出导致＋5V升高的原因。接通电源瞬间，测量I_{C1}①脚电压值正常。

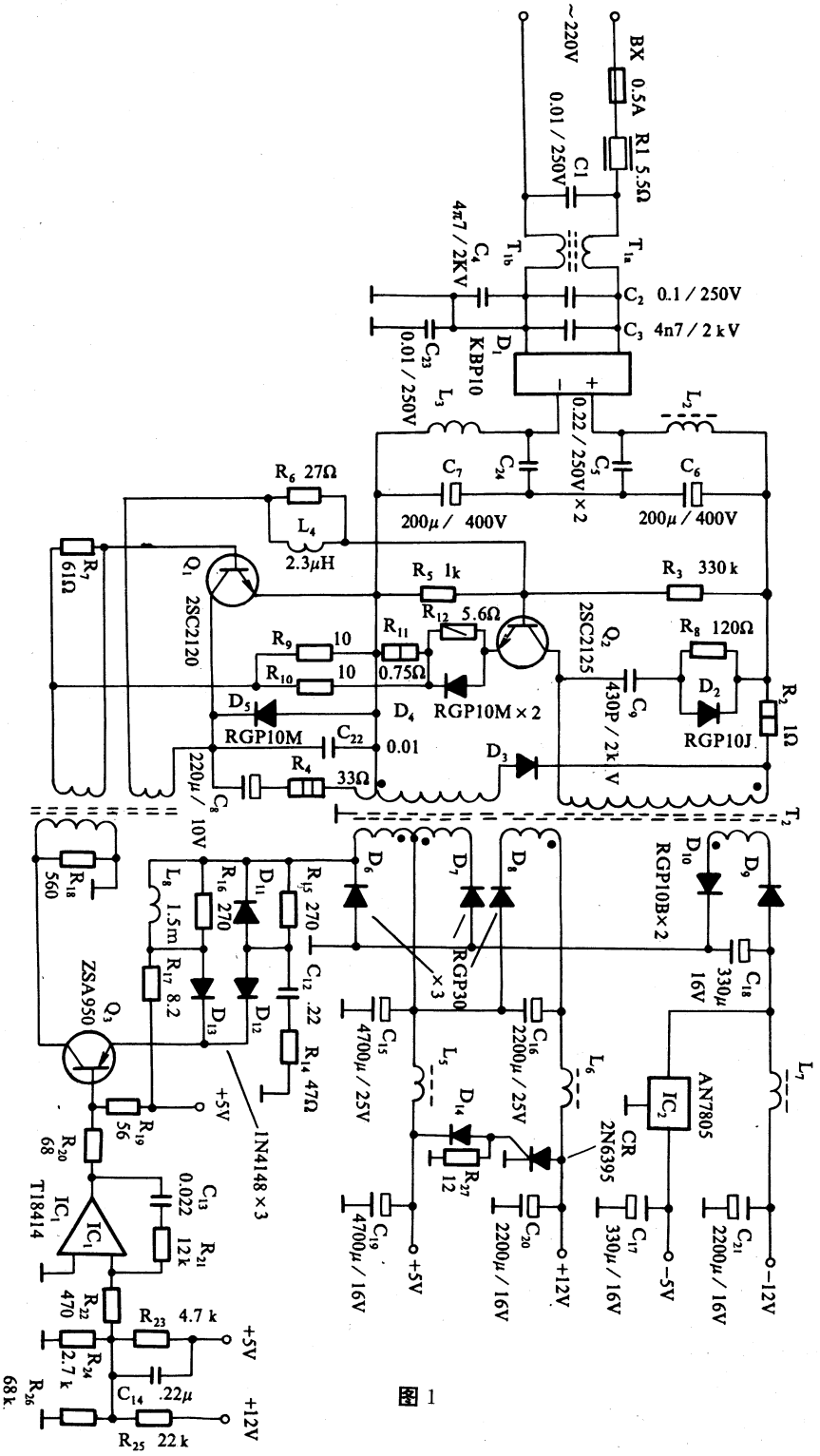

~220V
BX 0.5A
R1 5.5Ω
C1 0.01 / 250V
C4 4n7 / 2KV
T1b
T1a
C2 0.1 / 250V
C3 4n7 / 2 kV
C23 0.01 / 250V
D1 KBP10
L3
L2
0.22 / 250V × 2
C7
C24
C5
C6
200μ / 400V
200μ / 400V
R6 27Ω
L4 2.3μH
R7 61Ω
Q1 2SC2120
R5 1k
R3 330 k
R12 5.6Ω
R11 0.75Ω
R9 10
R10 10
D4 RGP10M × 2
Q2 2SC2125
R8 120Ω
C9 430P / 2k V
D2 RGP10J
R1 1Ω
D5 RGP10M
C22 0.01
D3
R4 33Ω
C8 220μ / 10V
T2
R18 560
L8 1.5m
R16 270
D11
R15 270
D6
D7
D8
D10 RGP10B × 2
D9
C18 330μ 16V
R17 8.2
D13
D12
C12 .22
R14 47Ω
Q3 ZSA950
×3 RGP30
C15 4700μ / 25V
C16 2200μ / 25V
L5
L6
L7
IC2 AN7805
CR 2N6395
+5V
1N4148 × 3
R19 56
R20 68
D14
R27 12
C19 4700μ / 16V
C20 2200μ / 16V
C17 330μ / 16V
C21 2200μ / 16V
+5V
+12V
-5V
-12V
IC1 T1841
C13 0.022
R21 12 k
R22 470
R23 4.7 k
R24 2.7 k
C14 .22μ
R25 22 k
R26 68 k
+5V
+12V

图 1

用同样方法测 I_{C1}③脚电压，发现电压比正常值 2.9V 升高了 1.3V，变成了 5V。从而造成 Q_3 基极反偏，Q_3 截止，高频振荡脉冲变宽，Q_2 导通时间变长，次级电压上升。当＋5V 升至 5、6V 时，CR 导通，＋12V 短路，迫使电路停止工作。分析判断是 I_{C1}有问题。拆下 I_{C1}，用 Ω×1K 挡测量其电流阻值应和表 1 相似。I_C 是一只多功能触发器，它能将输入电压的变化变成输出脉冲宽度的变化，来改变 Q2 的导通时间，起到使电压稳定的作用。在更换新 I_{C1}之前，也应对其进行测量。如买不到，笔者曾试用代替法，使用效果良好。选用两只优质 CS9014 三极管，β 应≥250，BVceo≥25V，BVces≤0.4V。将它们接成达林顿接法，见图 3。接入电路后，应对 R26 进行调整，使 Q3 基极电压值为 4V 左右。

在通电后瞬间，测 $IC_1$③脚电压正常。用同样方法测 Q_3－C 极为 0.006V，测 Q_1－B 极为 0.5V，测 Q_2－E 极为 0.55V。说明电源通电瞬间电路工作正常。再测量＋5V 也正常。在电路工作正常的情况下，造成次极电压跌落只能是保护电路误动作所致。关掉电源，拆下 D_{14}一端，再通电检查开关电源工作是否正常。如正常则是 D_{14}变值或损坏。如仍不正常则是 CR 损坏。用万用表×1K 挡测量 D_{14}正反向电阻值，如正确应在 JT－1 图示仪上再次对其测量。笔者多次发现 D_{14}稳压值变小的特殊情况。再换 CR 只要 IF≥3A 就可以使用。如读者无条件使用 JT－1 仪，可采用简单方法测量 D_{14}的稳压值，按图 4 电路连接，万用表指示值即为被测管的稳压值。对待换管也要进行测量，符合要求后才能上机使用，否则起不到保护作用。D_{14}送用功能耗为 1W，V_2 值为 5V 的玻封稳压二极管。

3）负压偏高，无正压

接通交流电，测量＋5V、＋12V、－5V、－12V 电压分别是 0V、0V、－5V、－12V。根据现象判断是整流二极管 D_6、D_7 损坏，造成＋12V 也无输出。由于无＋5V，使取样电压下跌，Q_2 导通时间增长。由于保护电路接在正电压网络，所以次级中的负电压就升高（－5V 已被 I_{C2}稳压，故不受影响）。图 1 电路中 D_6、D_7 采用两只高频二极管，读者可用国产型号 RGP－3C 高频整流二极管代替，其 I_0＝3A，V_R＝300V。PC 机电源中采用高频全波整流块，也可用 RGP－3C 管代替。如果手头只有小电流二极管，可以用两只或叁只并联使用，效果基本一样。

开关电源修复后不要急于上机使用，应对开关电源接上假负载进行老化试验。老化“中华学习”机电源时，用 2.5Ω、15W 线绕电阻接在＋5V 与地端，用 12Ω 15W 线绕电阻接在＋12V 与地端，－5V 和－12V 端各接 50Ω1W 和 120Ω2W 电阻。接通交流电，用万用表观察次级各组低压值的变化情况，有载时电压下降幅度和空载时相比应不大于 3%，重点是＋5V 和＋12V 两组电压。如四组电压均下降较多，应对 R24 或 R26 重新调整。（R_{24}为粗调、R_{26}为细调）。如＋5V、＋12V 两组电压下降较多，应对 R25 进行调整。以上老化时间不应少于 2 小时。老化 PC 机电源时，用 0.83Ω30W 线绕电阻接在＋5V 端，用 6Ω30W 线绕电阻接在＋12V 端，－5V、－12V 用负载电阻规格同上，调整也相同。由于假负载上呈受的功率很大，表面温度很高，通电老化时，不要用手接触负载电阻，以免被烫伤。

第二部分　排风扇的修理

开关电源中的排风扇由于长时间工作，很容易损坏，以致使开关调整管、整流二极管元件工作温度升高而损坏。目前，众多的微机电源中，排风扇结构基本相同，其工作过程是不断改变送给定子线圈电流，使定子铁芯产生磁场作用于转子（有许多 NS 极性磁体组成的磁环）的某异性磁场而形成旋转力矩。用来根据转子的磁性来判断送入某组定子线圈电流的任务是由定子供电电路担任。见图 2，其工作原理是这样的。＋12V 电源送入电源后，经 D_1 对电源进行极性识别后，首先加在开关管 Q_1 上。Q_1 基极偏置受 I_C 控制。I_C 是一只霍尔磁敏器件。当转子中 S 磁场靠近 I_C 时，I_C③脚输出一高电位，触发 Q_1 导通，发射极呈高电位，Q_3 工作。L_2 经过＋12V——→L_2——→$V_{ce}z$——→地形成电流回路，使 L2 的两个成对角同向绕组的铁芯上产生 N 磁场，推动与 N 极定子铁芯临近的转子中的 S 工作面，促使转子旋转。而 L1 的两个对角绕组无电流而不发生作用。待转子旋转 90°，转子工作区脱离了定子的磁力，同时 I_C 也脱离 S 区。由于转子旋转而来到的 N 区，使 I_C 关闭，③脚恢复低电位，Q_1 截止，集电极呈高电位。Q_{1-C}极高电位经隔离二极管 D_2 使 Q_3 由截止变成导通。L_1 经＋12V——→L_1——→Vcs3——→地形成电流回路，L_1 绕组中的两铁芯产生 N 磁场，作用于旋转到来的转子 S 极工作区上，形成第二次转矩。接着又重复第一工作过程，如此周而复始。转子越转越快，L_1、L_2 轮流工作的周期就越短，直至达到 I_C 最高开关频率时，转子保持稳定转速。

如计算机在使用中突然发现排风扇不转，应立急停机。拆下开关电源盒上盖，拧下固定排风扇的四只镙钉，取下风扇，并从电路板上焊下风扇电源线。用独立的 12V、关于 200mA 直流源作为临时电源。排风扇常见故障现象有两种。接上 12V 电源，手帮助旋转转子，转子旋转后一种是可贯性旋转，另一种是不可贯性旋转。下面就以上现象的修理介绍如下。

小心地取下风叶，拆下转子轴夹片，取下转子。先直观检查有无异常元件，并加以处理。然后按上 12V 电源。测量 Q_{1-C}极电压应为 1.4V，Q_{1-E}极应为 0V。因为这时 Q1 处于截止状态，而 Q_3 则工作在导通状态。测得该管 B 极为 0.6V，集电极应为 8V 左右。如果 Q_{3-c}极电压为 0V，则可能是 L_1 开路。关闭电源，用 Ω×10 挡测量 L_1 绕组直流阻值应为∞（正常值为 140Ω 左右）。说明 L_1 确实短路。如无法找出线头，则需重新绕制。在拆下定子之前，应先用尖头烙铁将剩余线头焊下，并做好定子位置标记。然后轻轻来回转动定子并同时向外拉。取下定子后，将 L_1 绕组从铁芯上轻轻绕出，并记住绕向和圈数，最后用同号线径的高强度漆包线绕制，有条件的话在线包上涂一层绝缘漆以助防潮。在

安装定子之前，还要对 $Q_{2,3}$进行检查。如果 L_1 是因通过电流过大而烧短，即 Q_3 早已先击穿损坏了。在更换 $Q_{2,3}$时，除采用同型号管子外，还可选用 BVceo≥25V，Pcm≥0.9W，Icm≥1A，β≥200，TO－92 塑封管代替。如 2SD471、2SD773 等。

当出现转子不可贯性旋转的现象，这说明 Q3 已工作，L1 有电流流过，产生了 N 磁场吸住了转子中的 N 极磁区。遇到这种情况先将直流 12V 降至 10V，防止 L_1、Q_3 长时间而损坏。取下转子，通电测量 Q_{3-c}极应为 6V 左右，再测量 Q_{2-c}极电压应等于电源电压，因 Q_1 截止，Q_2 也处于截止状态。用一只 1K 电阻和 R_3 并联，再次测量 Q_{2-c}极电压应为 6V，Q_{3-c}极应为 10V，说明开关管 Q_1 和放大管 Q_2 正常。去掉并联电阻，取一小磁铁将 S 极靠近 I_c，测量 $Q_{2,3-c}$极电压和刚才相同，说明 I_c 工作正常。如果 I_c 损坏最好选用同型号器件，如果买不到可选用其它型号，如 UGN3019、UGN3020 等。判断 I_c 的好坏可参考表 2 数据，用×1K 挡测量。在安装 IC 时可将原来卧式装置改成立式装置，这样可以增大磁敏作用面，提高霍尔器件的开关效果，见图 5。另外还应注意 I_c 的平面应和定子铁芯面垂直，防止 I_c 被转入磁铁磨擦而损坏。最后装上转子进行通电试验。要求转子通电后应能迅速起转，转速均匀平稳。正常后的排风扇就可以投入使用。

表 1

表棒 \ 阻值 \ 序号	①— ②	①—③	②— ③	1 2 3 ①—— 1N 端 ②—— COM ③—— OUT 端
红棒－黑棒	35 k	∞	∞	
黑棒－红棒	27 k	7.6k	5 k	

表 2

表棒 \ 阻值 \ 序号	①-②	①-③	②-③	123
红棒－黑棒	6.2 k	∞	∞	
黑棒－红棒	13 k	50 k	6.2k	

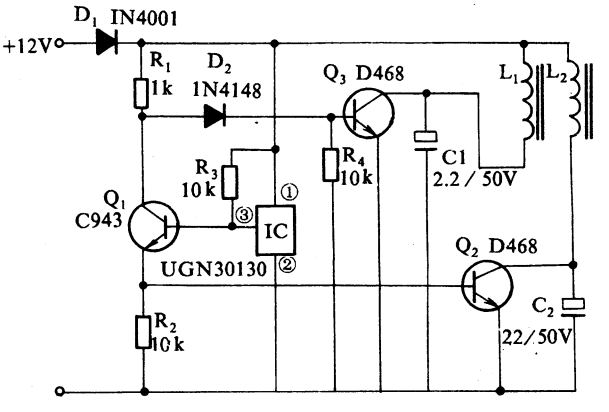


图 2

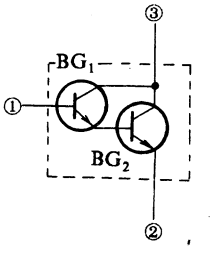


图 3

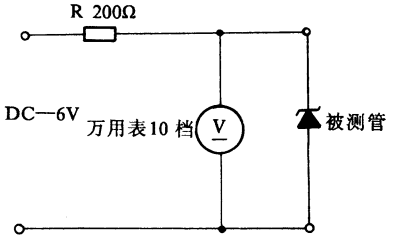


图 4

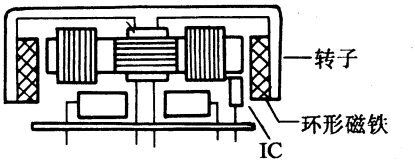


图 5

四通 MS-2401 打字机使用维修经验

江苏盐城师专　蔡长安

四通 MS－2401 中外文高级打字机是集显示、键盘、打印、主机于一体的专用于文字处理的计算机系统，它有体积小、重量轻、结构紧凑、打印的字体、字号多变等的特点，进入了各个办公室，以取代传统的文印设备。但如何使用维护好这类设备，除了按使用说明书上的规定要求进行操作外，我们还总结出了下面几点，以便使用者参考。

(1)蜡纸的打印方法。蜡纸的正确打印方法是：把档次调节杆调整到合适的位置，取下色带，将蜡纸下面的校对棉纸拿出放在蜡纸的上面。用这种方法打印蜡纸的好处是：既防止了蜡纸上的油墨等污物被打印针带入打印头中，阻塞打印针出口孔，使出针和回针阻力增大，打印针被色带拉弯或折断；又可防止了由于色带的原因(由于色带上有孔洞等)而刮弯打印针。

(2)打印色带的更换。打字机经过一段时间后，需要更换色带。色带实际上是一个墨比乌斯带，由于色带采用了这种形式，从而使色带的实际使用长度增加了一倍，提高了利用率。根据扭转方向的不同，墨比乌斯带分为左旋和右旋，左旋和右旋墨比乌斯带是不能互换的，这就由色带盒的结构所决定，四通 MS－2401 电脑打字机必须使用右旋的色带。更换色带时，只要打开盒盖，取出旧带装进右旋新色带即可。由于市售色带无标记，购买时必须认准旋转的方向。如果买回的色带是左旋的，可以用剪刀将色带剪开个斜口。把色带回转为右旋后，再把剪开处搭接，用结实的细线缝好即可。

(3)字形上下出现扭曲的处理。四通 MS－2401 打字机使用一段时间后，会出现上下扭曲。这是由于打印头经常向左边回转而碰撞造成的。处理这个现象的方法是：按住键盘上的“＜控制＞＋＜面＞”键，出现“请输入误差修正值(－4－3)”，可输入误差修正值“－2”，即可得到修正。

(4)“划痕”的处理。有时我们在打印蜡纸时，会在蜡纸上出现一条一条的“划痕”。这可能是档次调节杆调节的位置不适当造成的，这时我们只要把档次调节杆调到合适的位置即可消除“划痕”。如果调节档次调节杆后，还出现“划痕”，那么这可能是打印头的一根或几根打印针坏了，这时我们就要拆开打印头上的四颗小螺钉，更换新的打印针了。

(5)个别按键失灵的处理。由于打字时经常敲击键盘。会造成部分按键敲入时，屏幕上无显示的故障。现将修理方法简单介绍如下：首先确定是否是键盘故障。这可断开电源，旋下外壳上的 4 颗螺钉，取下外壳上盖，小心取下各接线卡，命出键盘。用万用表进行检测。按动键盘，测两焊点间的电阻，应为零值。如不为零值，说明两焊点间存在接触不良现象。

其次，取下键帽。方法是：去掉键体两脚上的焊锡。用烙铁进行加温，再用气球吹去焊锡。然后，在键盘表面上用两把中等型号的改刀放在键体两旁，适当用力向上挑，就能取下键体。拆开键体后，会看到两块铜片，一片像“π”，一片像“山”型。可把“山”型铜片的两边向外拨动，目的使两铜片接触良好。然后把两铜片装入到内部的两个小槽中，再用万用表测试是否已修理好。

对于上述方法不起作用的键，还可采用不常用的键体进行对换。

(接 41 页)

表 1

74LS123 功能表

	CLR	A	B	Q	Q
	L	X	X	L	H
	X	H	X	L	H
	X	X	L	L	H
	H	L	↑	⎍	⎎
	H	↓	H	⎍	⎎
	↑	L	H	⎍	⎎
脚号	3,11	1,9	2,10	5,13	4,12

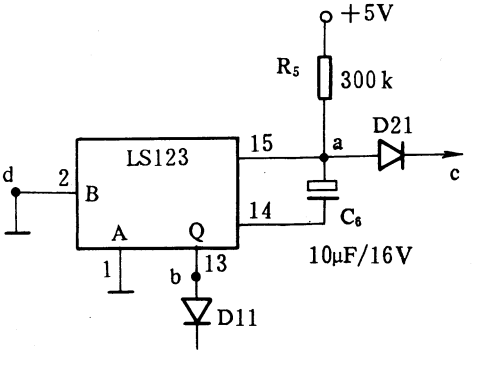


图 2

脉宽缩短到

$$tw = 0.45 \cdot R_5 \cdot C_6 = 0.45 \times 300k\Omega \times 10\mu F = 1.35S$$

经上述处理后，开机使用表明：前述的两个毛病均已治愈。读写操作也都正确无误。这样修理工作到此结束。修改后的局部电路图如图 2 所示。74LS123 的真值表如表 1 所示。

结论：

此毛病是因电路设计不周而引起的，并非使用不当，应引起生产厂的注意。

修理 CEC－I 型中华学习机软盘驱动器一例

赵　升

有 CEC－I 型中华学习机一套，其软盘驱动器有如下二个毛病：

1. 将软磁盘插入或取出时，驱动器中的驱动电机起动，运行约 30 秒钟后才停止。软驱上的 LED 灯不亮。

2. 对磁盘进行读写操作后（即软驱上的 LED 灯灭后）驱动电机仍要运行约 30 秒钟才停止。不影响对磁盘读写的正确性。

虽说以上两点只是“毛病”而非故障，也不影响对驱动器的使用，但毕竟对使用者来说是件很不舒服的事。正常情况应该是：插入、取出磁盘时驱动器电机不起动；对磁盘进行读写操作后（即软驱上 LED 灯灭后）约 1～2 秒，驱动电机即应停机。

分析：

1. 根据毛病 1 判断：电机的起动一定与驱动器中的防误写红外光电开关信号有关。

2. 根据毛病 2 及毛病 1 判断：驱动器内的驱动电机控制电路中有时间电路，并且这部分电路可能有问题。

检查：

此驱动器是杭州磁记录设备厂生产的 HS－550 型。因为无图纸，所以测绘其电路是十分必要的。但其控制电路较复杂，全面测绘即费时，也没有必要。基于以上分析，测绘电路时可以从以下三个方面入手进行。

a）驱动器上的防误写红灯开关。

b）从 CEC－I 主机来的磁驱控制信号。经分析应该是 J4－14 脚上的磁驱允许工作信号 ENBLI。

c）送向驱动电机的控制信号（驱动器内的 PIN1 插头）。

将与这三个方面都密切相关的那一部分电路先测绘出来，然后再测绘与此部分电路相关联的其它部分电路。直到能看清电路原理就可以了。有了这样一份局部电路图就可以对毛病进行定量分析了。测绘出的局部电路如图 1 所示。

这部分电路的核心是一可重触发单稳态电路，由 R_5、C_6、$\frac{1}{2}$片 74LS123 构成。查 74LS123 真值表可知：当 2# 脚上有一个个跳沿出现时，在 Q 端（13# 脚）输出一个正向单稳脉冲，其脉宽为：

$$tw = 0.45 \cdot R_5 \cdot C_6$$
$$= 0.45 \times 300k\Omega \times 220\mu F = 29.7S$$

与实际情况相符（见毛病 1）。

从主机来的允许工作信号 ENBLI 是一低电平有效信号。异或门的接法为同相驱动方式。故当 ENBLI 信号为低时，C_6 通过 D_{21}、74LS86 回路放电，使 a 点电位为“0”，Q 端输出“1”电平，驱动电机启动运行。当完成读写操作后，ENBLI 信号则上跳为“1”。此时电源经 R_5 向 C_6 充电，a 点电压逐步升高。经过约 30S 后，Q 端输出“0”电平，驱动电机停转（见毛病 2）。

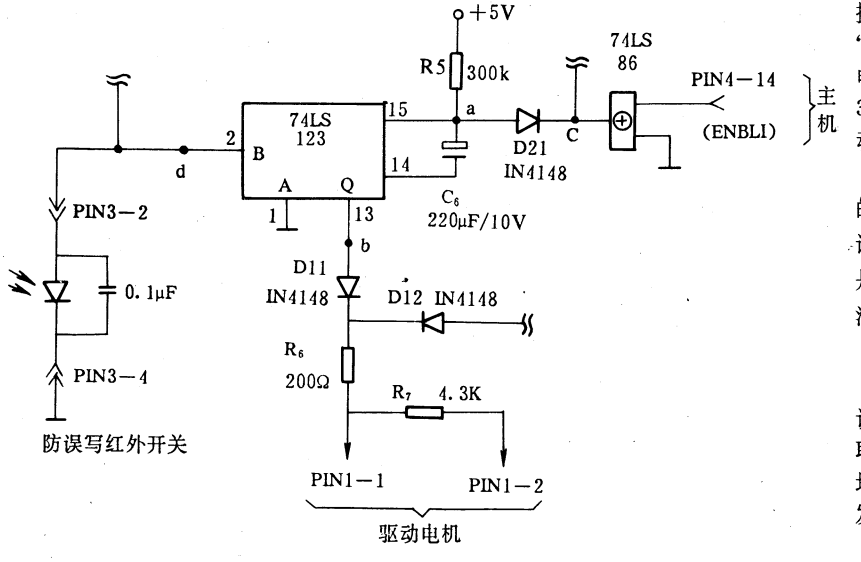


图 1

根据以上分析，对电路中的 a、b、c、d 各点电压进行了测试，测试结果证实了以上分析是正确的。这样解决问题的办法也就有了。

修理：

1. 在 d 点将印刷线断开，认切断红外开关与单稳电路的联系。并将 LS123 的 2# 脚接地，以防有干扰脉冲将其误触发。

2. 电容 C_6 用 10μF/16V 替换原 220μF/10V 规格，使单稳

（下转 40 页）

谈谈尼龙打印色带的使用技巧

福建省建阳一中　林斌

合式环形尼龙色带，由于具有不易折断，打印字符清晰，而且可以反复使用，比较经济等特点，所以深受广大用户的欢迎。下面介绍一些尼龙打印色带的使用技巧，供参考。

1. 为什么从市面上买来的尼龙打印色带，有的刚打印时颜色就很淡，而且字符不清楚？

答：这是因为各个厂家生产的尼龙打印色带，尽管出厂前已用油墨浸渍过了，但有的由于存放时间太久，油墨已干了，有的只是用很稀的油墨浸渍过，所以它只能直接放在装有墨盆的打印机上使用，如放在没有墨盆的打印机上使用时，则会出现上述现象。遇到这种情况时，一般只须将买来的尼龙打印色带先上好色带油后，再放入打印机中使用就行了。

2. 尼龙打印色带必须使用哪一种油墨？

答：必须是打印机色带专用的油墨，也就是通常有墨盆装置的打印机所采用的油墨，市面上称为高级再生打印色带油。这种油墨在一般计算机商店均有出售，只是价格较贵，但决不能用普通油印机中所用的油墨来代替。

3. 怎样给尼龙打印色带上油？

答：先在桌面上放一块塑料薄膜，并将色带盒内的色带全部拉出，然后将色带一段一段地铺平在薄膜上上油。上油时，可先用毛笔蘸少许色带油，然后在包带与打印头相背的那一面上的中间部分均匀地抹上一层薄薄的色带油，中间上油宽度约为带基的三分之一即可。但千万要注意，色带油不能抹得太多，而且色带朝向打印头的那一面不要上油，即使上油的那一面，也只能在中间部分，上下两边也不要上油。整条色带上油完毕后，最好不要立即卷入色带盒内，可以先露在外面让油墨凉干二、三天后再卷入色带盒内，这样以后就很好用了。

4. 尼龙打印色带使用时，打印字符油墨太浓，字迹模糊不清，且打印纸上有许多横条的“干扰线”，怎么办？

答：这种情况多是属于色带上油太多的缘故，必须采取以下几项措施：

(1)将色带全部拉出，用毛笔在色带上过油的部分再重新刷过一遍，将过多的油墨弄去；

(2)将上述处理过的色带先露在外面凉干几天后再卷入色带盒内使用；

(3)用棉花蘸上酒精将打印头前端部及色带护罩上粘有的油墨全部擦拭干净。因为色带上油太多，使用时必然会造成打印头前端部及色带护罩上粘有油墨，而影响了打印效果；

(4)检查一下打印头调整手柄的位置，看看打印头与打印辊之间的间隙是否太小，若间隙太小，也容易产生上述现象，特别是使用刚上过油的色带。这时可将打印头调整杆后移，将间隙调大即可。

5. 尼龙打印色带使用时产生“停带”现象，是什么原因？怎么解决？

答：这里所讲的“停带”，是指色带盒内的驱动轮照常转动，而色带却停滞不前，这样由于色带的打印位置不能变换，所打的字符就会越变越淡，甚至还会损坏色带。产生这种现象的原因可能有以下几种：

(1)色带盒内的驱动轮与被动轮上粘有较多的油墨。由于色带是夹在驱动轮和被动轮之间，靠摩擦传动而前进的，所以如果它们粘有较多的油墨时，就会使摩擦力减少，甚至产生打滑，而带不动色带。对此只好拆开色带盒的盖子，将两轮上的油墨全部擦拭干净。从这里进一步启示我们，在给色带上油时，驱动轮及被动轮与色带的接触部位(即色带的上下两边部分)千万不能上油，否则既浪费了色带油，又容易使驱动轮及被动轮粘上了油，而不能带动色带前进。

(2)色带盒内被动轮上的压紧弹簧脱落或漏装。由于该弹簧能使被动轮、色带、驱动轮紧密地接触在一起，以增大它们之间的摩擦力，而带动色带前进，所以如果该弹簧脱落或者漏装上去，就会造成摩擦力不足，而无法带动色带前进，这时应拆开色带盒的盖子，检查一下该弹簧是否装上或装好，弹力够不够等。

(3)色带在色带盒的出带端被挂住或出带阻力太大。遇到这种情况，可先用手或摄子将色带拉拉看有无被挂住或阻力太大等情况，通常多是尼龙色带接头处连接不好或色带某处有破损或出带端支架有裂缝等等现象都会造成色带被挂住而不能前进，此外色带出带路线安装不妥或出带压紧弹簧片没有装好也极易使色带绞合而拉不出。对此应仔细检查，并采取相应措施排除，甚至要更换新的色带。

(4)色带卷入色带盒内后，有时由于某种原因(如色带安装不好、久置不用等)，会使色带在带盒内的前后排列顺序紊乱，甚至有的与驱动轮或被动轮缠绕在一起，而造成色带不能前进，这种情况尽管很少发生，但也不能忽视，解决方法就是将色带全部拉出，再重新卷过一遍即可。

6. 尼龙打印色带使用时，产生“拖带”现象，是什么原因？如何解决？

答：这里所讲的“拖带”，是指色带在使用中，不能始终保持张紧状态，而被打印头拖出，又来不及卷入色带盒内，致使色带脱落下垂，这样轻者使打印机打不出字符来，重者会使色带与打印机架里的齿轮带或齿轮绞合在一起，使打印机不能正常工作，甚至会损坏打印机。产生这种故障的原因可能是：

(1)尼龙打印色带质量不好，如色带接头处连接不

圆滑，或色带使用久后某处有产生破损等，这样色带极易被打印头挂住，而产生“拖带”，对此只好更换一条新的色带。

(2)打印头及色带护罩在移动过程中与色带之间的滑动摩擦阻力太大。当色带受到的该阻力大于拉出端所受的阻力时，也会产生这种现象。对此可以从两方面来解决：一方面要尽量减少打印头及色带护罩与色带之间的摩擦力，即看看打印头与色带护罩上有无毛刺，它们之间的间隙是否太小；另一方面应想法增大色带拉出时阻力，即看看色带在出带端是否有被弹簧片压好，弹簧片的弹力足不足等。

7. 尼龙打印色带使用时经常碰到这种情况(特别是冬季)：在刚打印头几张时，字符不清楚，甚至还会发生“漏点”，只有当工作了一段时间后，字符才逐渐变为正常，这是什么原因？

答：这种现象开头往往被人误为色带上的油已用光，其实不然，它的原因多数是属于打印头上粘有较多油墨的缘故。

我们知道，上油的尼龙打印色带极易使打印头粘有油墨，而且使用久了，油墨还会慢慢渗入到打印针之间去，特别是充塞了打印针的中孔部分。这样在刚开机打印时，由于打印头的温度还较低(特别是冬季)，打印针之间被油污粘住了，致使打印针运动阻力增加，打印幅度减小，所打印的字符就不清楚。但工作一段时间，打印头的温度逐渐升高，油污熔解，阻力减小，打印针又可运动自如了，打出的字符就恢复正常清晰度了。

为此，使用尼龙打印色带的打印机，要定期清洗打印头，方法如下：将打印头从托架上拆下(注意电缆线不要拆)，先将打印头前端部的油污用酒精擦拭干净，再将打印针面的圆形导杆拆下，露出打印针的上半部分，然后用摄子夹一小团蘸有酒精或清洁剂的棉花对打印针进行清洗。清洗时应着重在针尖附近，特别是针尖中孔部分，并可将针尖左右分开以便逐根擦拭，但不要上下扳动针，以免针尖弯曲变形，也不要将打印针上下位置顺序弄错，同时要避免清洗后的油污流到打印头里面去，最好将打印头前端部朝下清洗，或事先用棉花塞在打印针下端以汲取清洗后的油污，洗完再取出。

8. 尼龙打印色带怎样应急使用？

答：尼龙打印色带使用一段时间后，打印出的字符就会逐渐变淡，这时应及时给色带上油才可继续使用，但给色带上油要花费一定时间，而且最好还要凉干几天后才好用，这样如果遇到要急需马上打印某些内容时，怎么办？下面我们介绍一种应急办法：即在色带盒底部前面两边的定位销与打印机架上的定位槽之间垫上2～3毫米厚的物体(例如硬纸片等)，将色带盒的前端部抬高，实则是为了改变色带的打印位置，这样打印时，由于色带的打印位置全部往下移一些，就会继续打出清晰的字符来。不过采取这种办法，不能将色带盒的前端部抬得过高，否则容易发生色带脱落现象。

为了方便使用，我们建议有用尼龙色带的打印机，最好都配备二条色带(包括色带盒)，以便交替上油，轮流使用，这样就不会出现上述问题了。

9. 尼龙打印色带使用到什么程度，需要更新？

答：尽管是可以反复上油使用，但这并不意味着它可以一直使用到坏为止。在一般情况下，尼龙打印色带在使用适当程度时，通常是色带表面开始起毛了，就不宜再继续使用了，这样既可减少打印针的磨损，又可避免发生刮断打印针现象，还能提高打印的质量和效率。

1991年度“AST”杯第二届全国软件人员竞赛在京揭晓

为了提高我国软件人员的素质，鼓励广大青年计算机软件人员为振兴中华而勤奋学习，以利于发现人才、培养人才，由中国软件行业协会、北京市科学技术协会、电子工业出版社联合举办，并由AST虹志(电脑)有限公司赞助的“1991年度《AST杯》全国计算机软件人员竞赛”已于11月20日在京揭晓，并在清华大学隆重举行了授奖大会。国内许多知名的计算机专家、教授出席了授奖大会有胡昆山(中国软件行业协会副秘书长)、吴文虎(清华大学计算机系教授，初级程序员级评奖委员会主任委员)、方裕(北京大学计算机系教授，程序员级评奖委员会主任委员)、张大力(北京市科学技术协会副主席)、胡贻志(中国软件行业协会副会长，物资部信息中心主任)、吴金生(电子工业出版社副社长)、郑人杰(清华大学软件开发中心副主任)、徐国平(中国软件行业协会高档微机协会副理事长、秘书长、中国软件行业协会考试指导中心副主任)、贾月珍(北京电脑天地学校副校长)。最后由胡贻志、张大力、吴金生及AST虹志(电脑)公司北京首席代表邓平贵等先生分别将奖品授予优胜选手。

程序员级获奖名单

等级	姓名	性别	年龄	学历	单位
一等奖	马　力	男	18	大学	吉林大学少年班学生
二等奖	郭东伟	男	19	大学	吉林大学学生
	王英超	男	19	大学	鞍山钢铁学院学生
三等奖	徐建东	男	20	大学	大连理工大学学生
	李志宏	男	23	大学	南昌市工商银行人民广场储蓄所
	夏伟文	男	23	大学	北京税务局崇文区分局
	王　革	男	22	大学	大连船用柴油机厂
	陈　晖	女	23	大学	北京市丰台区长辛店二七机车厂

初级程序员级获奖名单

等级	姓名	性别	年龄	学历	单位
一等奖	陈兆前	男	20	大学	杭　浙江大学学生
二等奖	潘　杰	男	17	高中	北京一六六中学生
	李万	男	19	大学	天津大学学生
三等奖	吴志军	男	23	大专	大连纺织品批发公司
	楼激扬	男	25	大学	浙江广播电视学校教师
	冯爱民	男	19	中专	南京工业学校教师
	李中政	男	19	中专	青岛碱厂
	陶　坚	男	21	高中	浙江省嘉兴市第一中学

初级程序员级水平考试辅导讲座
计算机病毒防范问答

常晓冀

从我国目前情况看，计算机病毒的防范已进入综合治理阶段。国内流行的七种常见病毒已经得到基本控制；但是，新的计算机病毒的数目仍在增加。今后随着我国计算应用技术的不断提高及工作站、计算机网络的普及，对计算机病毒的防范工作将变得更为迫切。

问题 1　何谓计算机病毒？

答：作为一种能够通过某种途径潜伏在计算机存储介质，如磁盘、磁带、光盘、网络中，条件成熟时，即被激活的具有对计算机资源进行破坏作用的“程序”或“指令集合”，称为计算机病毒。

问题 2　计算机病毒的实质及共性是什么？

答：计算机病毒是通过正常的操作运行的非授权入侵。所谓“非授权入侵”指的是病毒非法入侵系统，对其进行破坏的行径。

病毒的共性是：

(1)程序设计精巧；

(2)有较强的再生机制；

(3)潜伏深，难以发现；

(4)可激发性；

(5)危害性大。

问题 3　计算机病毒模块划分与工作流程情况如何？

答：病毒入侵计算机系统及在系统内传染循环过程如图一所示：

(2)计算机病毒模块划分如(图二)所示：

(3)病毒工作流程(以小球病毒为例)如图三所示

问题 4　计算机病毒的分类原则是什么？

答：(1)根据病毒入侵途径大体可分为以下四种

*．源码型病毒：病毒在源程序被编译之前插入到诸如 C、PascaL、COBOL、FORTRAN 等高级语言所编写的源程序中。该类病毒难以编制，较少见。

*．入侵型病毒：病毒侵入主程序中，不破坏主程序，难以清除病毒。

*．操作系统型病毒：嵌入或部分取代操作系统工作。病毒通常在操作系统启动时(开机时)自动装入。具有极强的传染力和破坏力。如常见的小球和大麻即为该类病毒。(包括它们的变种)

*．外壳型病毒：病毒将自身置于主程序的外围(或头或尾)，不修改原程序主体，以感染. COM 或. EXE 文件为多见。如常见的以色列(黑色星期五)、1575 病毒及其变种均属这类病毒。

(2)根据病毒的表现形式，可分为良性和恶性二种。后者是破坏系统文件、数据文件和造成死机的罪魁祸首。

(3)根据病毒攻击的对象可分为三类：

*．攻击微型机(我国目前以这种病毒所占比例为最大)

*．攻击小型机

*．攻击工作站、网络

其中以第三种病毒所造成的危害最大，它的存在，构成对重要信息系统和金融信息系统的潜在危胁！是今后计算机病毒防范的重点。

问题 5　如何理解计算机病毒的复杂性？

答：病毒的复杂性表现在以下几方面

(1)病毒种类多，国外发现的病毒已达上百种，国

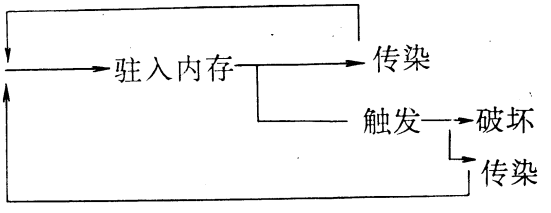


图 1

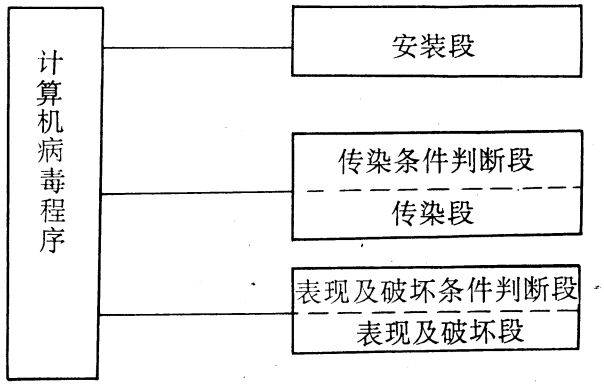


图 2

内也有数十种之多。

(2)病毒变种多,仅小球病毒变种就有十多种。

(3)来源复杂,国外传入,如小球、大麻、以色列、杨基、1701/1704、1575及其变种等。国产病毒数目也在增多,如"中国炸弹"、"6.4"等。

(4)病毒正在升级;正在由单一型向智能型方向演变。如"4096"病毒即是智能型病毒,该病毒已突破通用解毒软件的防护系统,通过篡改DOS系统内部数据,对DOS进行大量修改,而不改变其中断向量。主要感染.COM或.EXE文件,有些病毒甚至带上了某些政治色彩。

问题6　计算机病毒将对计算机系统造成什么样的危害?

答:计算机病毒所造成的危害按其侵入系统的次数和它的病毒性质可分为以下几方面:

(1)单一型病毒的危害

*.一次侵入,以大麻为例,多次入侵系统导致计算机系统引导程序被病毒程序完全复盖造成死机,无法启动。

(2)两种以上病毒入侵计算机系统所造成的交叉感染,所导致程序和数据文件遭破坏及引导系统故障的严重程度将大大超过单一型病毒所造成的危害。以"6.4"与"大麻"病毒同时入侵计算机系统为例来说明。若以一般解毒软件去解会出现硬盘无法启动的严重后果。

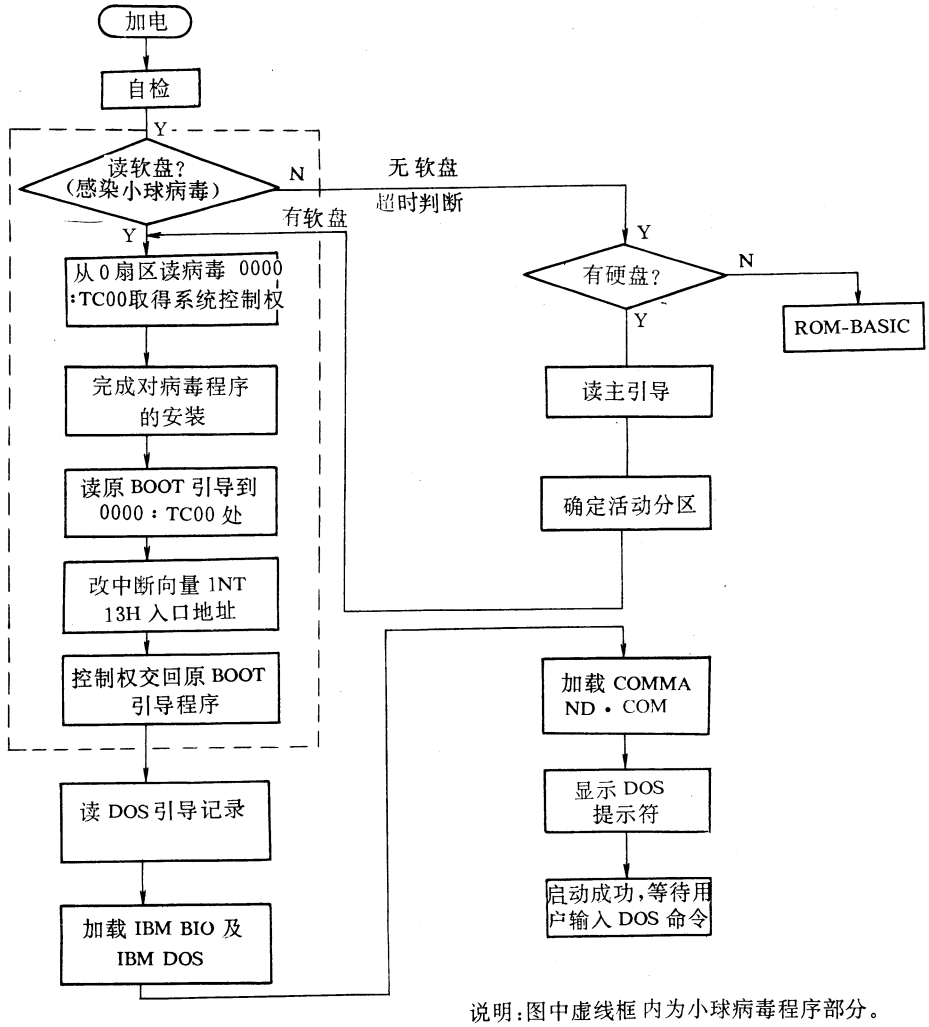
(3)智能型病毒的危害则难以预料。

问题7　如何实现对计算机病毒的检测?

答:检测计算机病毒主要通过以下工具和方法

(1)DEBUG调试程序,提供一个检测手段使用户能监督和控制欲调试的程序执行过程。

(2)PCTOOLS实用软件,完成对系统的检测及对系统数据的修改和恢复工作。根据用户需要可采用不



说明:图中虚线框内为小球病毒程序部分。

图3

同版本。目前,PCTOOLS 最高版本为 6.0V(英文版)

(3)重要系统信息的掌握

*.DOS 系统磁盘参数表

*.DOS 系统文件分配表

*.DOS 系统文件目录表

*.DOS 系统中断向量表

*.DOS 所使用的寄存器

(4)病毒检测软件

问题 8　如何对常见病毒进行处理?

答:对常见病毒可使用以下简易方法加以清除。

(1)清除小球病毒:系统复盖法,将未感染病毒的系统盘插入 A 驱动器,冷、热启动均可(该系统版本不得低于原硬盘操作系统版本)。

(2)清除大麻病毒:软盘清除法

A>DEBUG↙

−L　100 1 0b 1↙

−D　100 2FF↙

−W　100 101↙

免疫法:将未感染病毒的非系统盘引导区的前 4 个字节置成 EA,05,00,C0。

(3)以色列(黑色星期五)病毒症状与免疫

*.症状

(a).COM 或.EXE 文件字节长度增加 1.8K 字节。

(b)每运行一次感染病毒的文件,硬盘空间减少 1.8K,直至死机。

*.免疫方法:将病毒开始的第 5 个字节处标志"MSDOS"存入文件。具体作法是最后把 CX 的值置为文件长度加 10,再存盘。

(4)1575 病毒症状与免疫:

*.症状:

(a).COM 或.EXE 文件长度增加 1575～1591B 病毒文件日期为感染日期。

(b)每 DIR 后,磁盘空间减少 1575～1591B 字节。

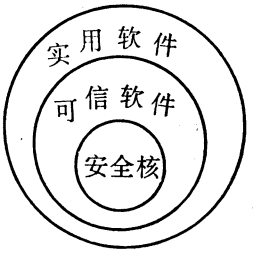
(c)屏幕出现一条"蜈蚣状虫子"。

*.免疫:在文件末尾加上 0A0C,并把 COM 或 EXE 文件属性改为只读。

问题 9　目前我国使用哪些办法对计算机病毒进行防范?

答:对计算机病毒的防范,可采用软件,硬件防范及对开放的 DOS 操作系统进行修改等方法。

1.软件防范



安全操作系统图示

注:图中"安全核"是由计算机硬件设备和软件构成,它的功能是完成对数据的存取控制。

图 4

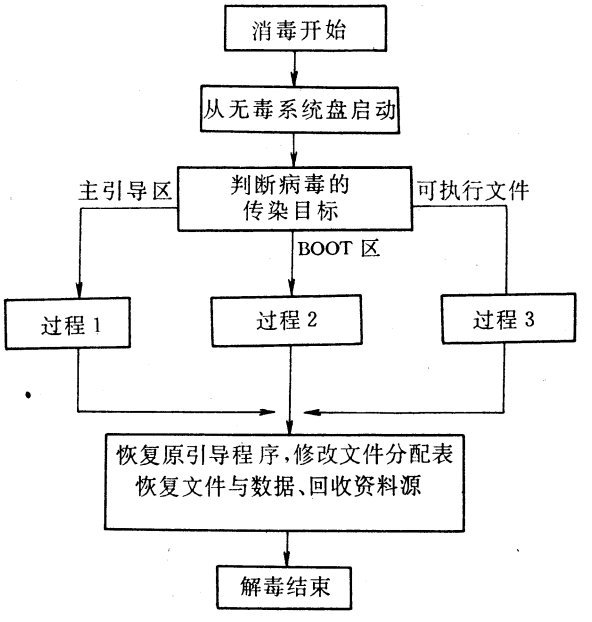


图 5(1)

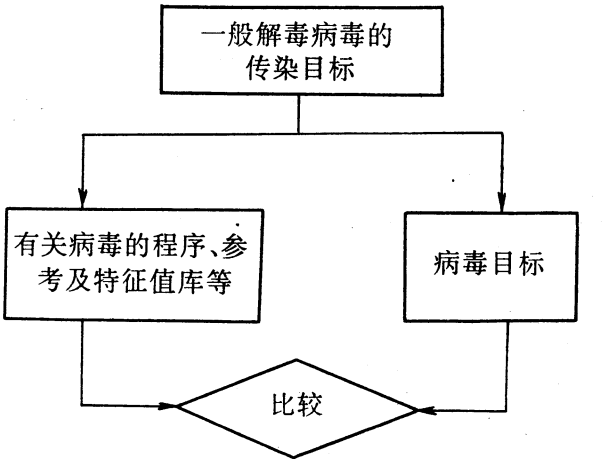


图 5(2)

(1)为用户设置数据保护区[ADM 软件]

(2)对常见病毒的免疫预防

(3)研制主动防御病毒程序,使其常驻内存密切注意系统内引导扇区及中断向量的变化,如有变化,立即报警,并中断其执行。

(4)解毒软件

*.一般解毒软件:

采用传统编程法编制的解毒软件,称为一般解毒软件,只能对付单一型病毒。

*.智能型解毒软件:

采用人工智能算法和知识库编制而成,可用以对付病毒变种和多种病毒交叉感染在同一计算机系统的复杂情况,是有发展前途的人工智能型软件。

2.硬件防护:

(1)病毒硬件防护卡,可直接插入系统扩充槽内。用以监督系统内中断向量等重要参数的变化。优点是速度快,缺点是造价高。因此,硬防护卡与解毒软件联合使用效果可能好些。

(2)硬盘保护电路,是对高档微机的硬盘,采用配置信息变换技术加以限制。开机时,配置为无硬盘驱动器,从而实现隐藏并保护硬盘的功能。

3.修改 DOS 操作系统等措施

(1)设计安全操作系统(如图 4 所示)

(2)固化 DOS"病毒防护卡

以固化 DOS 操作系统为基础,将引导扇区 DOS 系统文件写入芯片,用以避免引导区和外壳型病毒的入侵。

问题 10　一般解毒软件的工作原理是什么?

答:我们以图五为例,说明解毒软件的工作原理。

(图五)说明:图中"判断病毒传染目标",这一步骤是以该软件所建立的病毒特征值库等内容为依据完成对病毒目标的判别。

与病毒有关的程序文档、各类库内容如下:

(1)正常程序与 BOOT 区内容档案

(2)病毒特征值库

(3)内存中断向量表

(4)内存容量库

实践证明,病毒特征值库的建立,无论对病毒的检测还是消除都是必不可少的。

问题 11　如何正确使用解毒软件?

答:解毒软件的正确使用与否,直接关系到用户的切身利益,切不可等闲视之。以下从几方面来说明。

(1)解毒软件(尤其是一般解毒软件)是针对某些病毒(单一型病毒,并不包括其变种)。没有万能的解毒软件,如同没有万能的包治百病的药品一样。

(2)解毒软件一般不主张拷入硬盘运行,因它本身不具有对病毒的免疫能力。在装有 DOS 操作系统的硬盘环境下,只能助长病毒的进一步扩散。

(3)最好不使用不同解毒软件,去解同一系统内的病毒。其后果可能很糟。

*.其一,可能发生用户文件被删除,硬盘启动故障等严重问题。原因是不同的解毒软件,解毒原理不同,所使用的汇编语言、高级语言及算法均有差异。

*.其二,出现上述问题后,难以查明是哪个解毒软件的错误及错误所在?

(4)学会正确使用解毒软件的方法,严格按操作说明进行解毒。解毒后,一定要重新启动(热)机器。

问题 12　如何预防计算机病毒?

答:最根本的预防病毒措施,是加强对设备(微机、外设)的管理工作。作好对系统的后备和用户文件后备。养成良好的设备文档记录习惯尤其是对设备故障(包括病毒、硬件、软件等故障)的处理记录;此外,还要注意以下几点:

1.尽量避免往本系统内装入外来的、未经严格检测的应用软件。电子游戏盘是传染病毒的最有效的媒介之一。目前的检测软件,其检测范围有限,难以保证外来软件所携带的病毒不侵入本系统。

(2)对本系统中重要文件进行简单加密,来实现对文件和系统的保护作用。

(3)外单位人员上机,也是病毒入侵途径之一。

(4)机器故障,送出维修,有可能带回病毒。

(5)新购机器是传播病毒的又一途径。

(6)向外出租或外借机器,难以保证不传染上病毒。

为维护微机系统的安全运行,现提出以下建议:

1.微机系统中硬件、软件、外设故障与病毒感染交织情况发生,应尽快使用系统自身的检测软件、解毒软件分层次加以排除,避免发生因误判对系统所造成的不必要损失。

例如,硬盘系统不能正常启动。此时可采用以下步骤逐层进行排除:

(1)用系统检测软件对硬盘进行检测,用以判断硬盘故障。

(2)用解毒软件检测系统中存在的病毒。

(3)以上二个因素均排除后,再用其他软件检测工具(或其他工具)排除系统软件中存在的软故障,以求尽快恢复硬盘的正常启动功能。

2.病毒感染后的善后恢复工作很重要。有些系统反复被某种病毒感染。其结果造成文件分配表紊乱,严重影响系统正常工作。

(3)配备一名技术较强的系统管理员,负责微机系统的日常维护,对安全运行十分重要。

SCB—1 单片单板机简介

随着科学技术和生产的发展，单片机技术已广泛应用于工业、农业、国防、科研、教学及日常生活等各个领域。为进一步推广应用单片机技术，逐步取代日渐落后的以 Z80 作 CPU 的 TP801 单板机市场，我所 1989 年在国内首先研制成功一种性能优异的 SCB—1 型单片单板机，随即在《电子与电脑》杂志上发表，该机硬件原理及软件监控程序清单全部刊出。产品批量投放市场后，在全国引起广泛影响，曾几度供不应求。

SCB—1 何以有如此旺盛的生命力呢？

首先，SCB—1 的硬件配置全面、典型、规范。它以 8031 为 CPU，系统时钟最高可达 12MHz，扩有 8K 外部 EPROM，带掉电保护的 16K RAM，6 位 LED 数据显示，5×5 的键盘及专用于抗干扰的“看门狗”电路，另外还配有一般开发机上没有而又经常用的 8 路 8 位 A/D 和 1 路 D/A 电路，并配有一标准 RS232 口和由 8255 扩充的 3 个 8 位并行或位控的 I/O 口。另外还可以直接选配 E^2PROM 片子。

SCB—1 不仅硬件配置合理，而且还有愈 4K 的监控程序，它为用户提供了大量丰富实用的软件，并且都以子程序形式提供，它包括单步执行程序及现场保护与恢复程序、键扫描程序、显示程序、存贮器读写程序、存贮器内容数据移动与 E^2PROM 编程程序、串行通讯的接收程序及 IBM—PC 兼容机上的发送程序和 SCB—1 的发送和 IBM—PC 兼容机的接收程序，还有很实用的 A/D 转换子程序、D/A 程序、打印机驱动程序以及检空、比较、定时时钟等程序，在用户要求下还可以配上各种运算子程序库。

监控工作主程序可以使用户通过键盘操作进行搬家、检查、改写内外 RAM 并可以进行连续或单步执行程序，并可以直接与 PC 机进行通讯操作。除了监控中提供的软件外，SCB—1 系统还提供一个在 PC 机上运行的软盘文件，它有正反汇编程序，正反通讯程序及监控程序清单。

优良的软、硬件基础 SCB—1 具有广泛的应用价值。今年 9 月武汉大学出版社将介绍 SCB—1 单片机的《单片机原理及制作》一书列选为大中专教材，正式出版发行，该书不仅对 SCB—1 的软、硬件进行了详尽的介绍，还从教学实际出发，配以实验操作及应用实例等内容，使 SCB—1 系统在教学领域内大受欢迎。现已有几十所各类学校用此系统进行单片机教学。

SCB—1 的带掉电保护的 16K RAM 为用户提供了足够大的程序调试空间，当单步执行程序，可由用户选择观察某个寄存器的内容变化。利用软盘提供的汇编模块，直接用汇编语言编程后传送到 SCB—1 上去，使程序可以即调即改，省却了强记、手工输入机器码等繁重劳动，使开发周期大大缩短。因而 SCB—1 还被广泛用作开发工具。

SCB—1 外形有两种结构，即整体式和分体式。分体式结构特别适用于组配各种仪器仪表，配上面板即可成产品，又由于其有专门的抗干扰电路，即使在系统遇到强干扰死机后，它也可以自动地恢复并继续运行下去，再加上易于开发的特点及 SCB—1 优越的性能价格比，使其在工业控制及仪表配套领域里的应用日益广泛。

在全国同仁的大力支持之下，我所生产的 SCB—1 单板机销售量不断上升，生产成本不断下降，使其现在零售价仅为 495 元；又由于我所毫不保留地将软、硬件资料全面公开，以用户需要为第一原则，并有各种灵活的价格和销售方式努力做好销售服务工作，深受广大用户欢迎。为更好地开拓 SCB—1 的应用市场，本所将在全国各地征求经销代理单位，并可给予优惠的价格及技术支持。对于配产品用的大批量用户，本所除可以进行专门技术指导服务外，还可以按指定要求进行生产并另行定价。

本所另有详尽的 SCB—1 的销售及服务说明，函索即寄。

单　　位：武汉市尚吉电子研究所
联系地址：武汉武昌珞瑜路 37—3 号
邮政编码：430070
电　　话：716138
电报挂号：4561
开户银行：工商行洪山区办
帐号：2604—67—016072001

转　让

中华学习机 CEC—I 型主机一台，价格 750.00 元。负责邮寄费用并免费赠送若干软件及资料。

联系人：陈长健　地址：西藏拉萨市电信局长话科
邮编：850000

更　正

本刊 91 年第 6 期第 10 页《女子键美体型电脑咨询程序》一文的作者地址应为四川泸州医学院。

编辑部

一九九一年“AST”杯

全国计算机软件人员竞赛（第二届）

初级程序员笔试题（120分钟）

准考证号＿＿＿＿　　地区＿＿＿＿　　姓名＿＿＿＿

题号	一	二	三	四	五	六	七	八	总分
分数	11	15	4	6	9	25	15	15	100
得分									

一、从供选择的答案中选出正确答案的编号填写在横线上。（每空1分，共11分）

1.下列数中最小的数为＿＿＿＿＿＿。

供选择的答案：

A. $(101001)_2$　　B. $(52)_{10}$　　C. $(101001)_{BCD}$　　D. $(23)_{16}$

2.采用＿＿＿＿＿＿表示数字时，零的表示是唯一的。

供选择的答案：

A. 原码　　B. 补码　　C. 原码和补码　　D. 原码和反码

3.已知二进制数 X＝0.11011，Y＝−0.10010，则 X−Y 的补码形式的运算结果是＿＿＿＿＿＿。

供选择的答案：

A. 1.01101　　B. 1.10111　　C. 0.11001　　D. 0.01001

4.指令周期是指＿＿＿＿＿＿。

供选择的答案：

A. CPU 从主存取出一条指令和执行这条指令的时间

B. CPU 执行一条指令的时间

C. CPU 从主存取出一条指令的时间

D. 时钟周期时间

5.广泛用于 IBM PC 中的 Intel8088CPU 是＿＿＿＿＿＿。80386 是＿＿＿＿＿＿。

供选择的答案：

A. 8 位微处理器　　B. 16 位微处理器

C. 准 16 位微处理器　　D. 32 位微处理器

6.对逻辑表达式 $\overline{AB+\bar{A}C}$ 化简的结果是＿＿＿＿＿＿。

供选择的答案：

A. AB+BC　　B. $AB+\bar{A}C$　　C. $A\bar{B}+\bar{A}\bar{C}$　　D. $\bar{A}\bar{B}+\bar{A}C$

7.80286 采用了两种不同的操作方式，即＿＿＿＿＿＿方式和＿＿＿＿＿＿方式。在后一种方式下，提供分段式虚存系统所需的环境，是真正发挥硬件性能的操作方式。

供选择的答案:

A. 保护方式　　B. 批处理方式　　C. 假脱机方式

D. 虚拟方式　　E. 实方式　　F. 实时处理方式

8.为用户提供诸多 DOS 功能，具有 PCShell，DiskFix，PCBackup 和 DeskTop 功能模块的_____是高级 PC 工具软件之一。

供选择的答案:

A. Windows　　B. Pctools　　C. X-Window　　D. Open Desktop

9.在 IBM PC 机上使用的 CCDOS 是一个_____操作系统。

供选择的答案:

A. 中西文兼容　　B. 全中文　　C. 联想汉字　　D. 汉字排版印刷

二、从供选择的答案中选出与下列叙述关系最密切的字句，把编号写在题后答案对应的横线上。(每空 1 分，共 15 分)

1、在操作系统的存储中，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，用以合理地利用外部设备。对磁盘的管理属于 C；对磁盘中信息的管理属于 D。

供选择的答案:

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

答案: A_____　B_____　C_____　D_____

2.排序是数据处理中的重要运算，冒泡排序和快速排序属于 A；直接选择排序属于 B；二分法插入排序属于 C。

供选择的答案:

A，B，C. ①插入排序　②选择排序　③交换排序

④表插入排序　⑤归并排序　⑥希尔排序

答案: A_____　B_____　C_____

3.要进行顺序查找，则线性表 A。要进行二分法查找，则线性表 B。若表中元素个数为 n，则顺序查找的平均比较次数为 C。

A，B. ①必须以顺序方式存储

②必须以链接方式存储

③必须以顺序方式存储，且数据元素已按值递增或递减顺序排好

④必须以链接方式存储，且数据元素已按值递增或递减顺序排好

⑤既可以以顺序方式存储，也可以以链接方式存储

C. ①n　②n / 2　③n^2

答案: A_____　B_____　C_____

4.栈和队列都是 A。若进栈的序列为 1，2，3，4，则 B 不可能是一个出栈序列。若进队列的序列为 1，2，3，4，则 C 是一个出队列的序列。

供选择的答案:

A. ①顺序存储的线性结构　②链式存储的线性结构

③限制存取点的线性结构　④限制存取点的非线性结构

B. ①3, 2, 1, 4　②3, 2, 4, 1　③4, 2, 3, 1

C. ①4, 3, 2, 1　②1, 2, 3, 4　③1, 3, 2, 4

答案: A_____ B_____ C_____

5.奇偶校验是一种常用的代码校验方法. 这种校验方法能发现 A 个错误, B 纠正错误。

供选择的答案:

A. ①奇数　②偶数　③奇数或偶数　④奇数和偶数

B. ①不能　②能　③特定条件下能　④某些条件下不能

答案: A_____ B_____

三、判断下列叙述是否正确, 请在题前的括号内画✓(正确) 或×(错误). (共 4 分)

( ) 1.浮点运算指令对用于科学计算的计算机是必要的, 能提高机器的运算速度。

( ) 2.I/O接口电路是一种输入/输出设备。

( ) 3.一般的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。

( ) 4.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。

( ) 5.数据库系统最基本的特点是管理的数据量很大。

( ) 6.数据的完整性是指保护数据以防止不合法的使用。

( ) 7.计算机病毒的破坏能力主要取决于病毒程序的长短。

( ) 8.数据库管理系统是数据库中程序和数据的总称。

四、将下列英文的中文译文写在对应横线上. (每题 1 分, 共 6 分)

1.Program must be written in a language the computer will understand.

_____

2.Many microcomputer systems are single-user systems,only one person can use the systems at a time.

_____

3.The screen display tells you the program is ready to run.

_____

4.File allocation table bad,drive A Abort,Retry,Ignore?

_____

5.Warning! All data on non-removable disk drive C will be lost.

_____

6.Insufficient memory for system transfer.

_____

五、阅读下面的 BASIC 程序, 写出程序运行时显示的结果: (每题 3 分, 共 9 分)

①
```
10 DEF FNA(X)=1+2*X
20 X=VAL(CHR$(65))-2
25 PRINT "X=";X
30 Y=FNA(FNA(FNA(X)/2))
```

运行结果:

```
 40 PRINT "Y=";Y
 50 END
```

②

```
10 INPUT "输入";X
20 IF X<=0 THEN PRINT "x=";X:END
30 GOSUB 60
40 PRINT "X+X=";X+X
50 END
60 X=X*3
70 IF ABS(INT(X))=ABS(X) THEN 90
80 PRINT"x=";X
90 PRINT "&&&"
100 RETURN
```

输入 2 显示结果:

输入-4.6 显示结果:

输入 0 显示结果:

③

```
10 A$ = STR$(VAL(RIGHT$("123450",1)))
20 FOR X=5 TO 1 STEP -1
30    PRINT TAB(2*X+5);
40    FOR Y=1 TO 7-X
50       PRINT A$;
60    NEXT Y
70    PRINT
80 NEXT X
90 END
```

运行结果:

六、填空（共 25 分）

①填充下面的 BASIC 程序，使其功能与 dBASEⅢ的 FILE( )函数类似：如果磁盘上存在指定的文件，就显示-1；否则显示 0。(4 分)

提示：BASIC 程序运行时如果在磁盘上找不到指定的文件，返回的错误号是 53。

```
10 ON ERROR GOTO 100
20 CLS
30 X=-1
40 INPUT "Input a Filename:";F$
50 _____
60 _____
70 PRINT X
80 END
100 IF ERR=53 THEN X=0
110 RESUME 70
```

②设四个连续自然数中的最小数为 A，且四个数之积为 1680，填充下面的 BASIC 程序，使其能从小到大依次输出这四个数。(4 分)

（待 续）

附 录

KDC-Ⅲ开发机监控程序（在仿真器方面的）

```
FILE NAME P4.PRT
                                        ORG     0000H
0000  00                 NOP
0001  0130               AJMP    SS
;
                                        ORG     0003H
0003  D56908              DJNZ     69H,S1
0006  0569                INC      69H
0008  022003              LJMP    2003H
                                        ORG     000BH
000B  02200B              LJMP    200BH
000E  0203A2     S1:      LJMP    STEP1
                                        ORG     0013H
0013  022013              LJMP    2013H
0016  020465              LJMP    UNEXEC1
                                        ORG     001BH
001B  02201B              LJMP    201BH
                                        ORG     0023H
0023  022023              LJMP    2023H
                                        ORG     002BH
002B  02202B              LJMP    202BH
;
                                        ORG     0030H
0030  5389F0     SS:      ANL     TMOD,   #0F0H
0033  438920              ORL     TMOD,   #020H
0036  758BF3              MOV     TL1,    #0F3H
0039  758DF3              MOV     TH1,    #0F3H
003C  D28E                SETB    TCON.6
003E  C2AC                CLR     ES
0040  C2AB                CLR     ET1
0042  759852              MOV     SCON,   #052H
0045  75816A              MOV     SP,#6AH
0048  02005B              LJMP    START
004B  3099FD     SEND:    JNB     TI,     SEND
004E  C299                CLR     TI
0050  F599                MOV     SBUF,   A
0052  22                  RET
0053  3098FD     RECEIVE:JNB      RI,     RECEIVE
0056  C298                CLR     RI
0058  E599                MOV     A,      SBUF
005A  22                  RET
005B  120053     START:   LCALL   RECEIVE
005E  04                  INC     A
005F  12004B              LCALL   SEND
0062  B431F6              CJNE    A,#31H,START
0065  E4                  CLR     A ;MOV    SP,#6AH
0066  F570                MOV     70H     ,A      ;PC
0068  F571                MOV     71H     ,A
006A  F573                MOV     73H     ,A      ;PSW
006C  F574                MOV     74H     ,A      ;ACC
006E  F575                MOV     75H     ,A      ;B
0070  F576                MOV     76H     ,A      ;DPTR
0072  F577                MOV     77H     ,A
0074  F578                MOV     78H     ,A      ;R0
0076  F579                MOV     79H,A           ;R1
0078  F57A                MOV     7AH,A
007A  F57B                MOV     7BH,A
007C  F57C                MOV     7CH,A
007E  F57D                MOV     7DH,A
```

```
0080   F57E              MOV     7EH       ,A
0082   757230            MOV     72H,#30H        ;SP
0085   756901             MOV     69H,#01H
0088   120053    BEGIN:  LCALL   RECEIVE
008B   90009D            MOV     DPTR,     #TABLE1
008E   75F003            MOV     B,#03H
0091   A4                MUL     AB
0092   C0E0              PUSH    ACC
0094   E5F0              MOV     A,        B
0096   2583              ADD     A,        DPH
0098   F583              MOV     DPH,      A
009A   D0E0              POP     A
009C   73                JMP     @A+DPTR
009D   0200F6    TABLE1: LJMP    MAINSFR
00A0   0200D0            LJMP    LOAD
00A3   0203C4            LJMP    SSTEP
00A6   0203D4            LJMP    MULTSTEP
00A9   0203EC            LJMP    BREAK
00AC   02049D            LJMP    EXEC
00AF   0204C6            LJMP    TRACE
00B2   020185            LJMP    LRAM
00B5   02019C            LJMP    LSFR
00B8   020212            LJMP    LP
00BB   020228            LJMP    MRAM
00BE   02024C            LJMP    MSFR
00C1   02033B            LJMP    MP
00C4   02015D            LJMP    DISSA
00C7   020410            LJMP    UNEXEC
00CA   020030            LJMP    SS
00CD   020358            LJMP    ASSEM
;
00D0   120053    LOAD:   LCALL   RECEIVE
00D3   F583              MOV     DPH,      A
00D5   120053            LCALL   RECEIVE
00D8   F582              MOV     DPL,      A
00DA   120053            LCALL   RECEIVE
00DD   9583              SUBB    A,        DPH
00DF   F9                MOV     R1,       A
00E0   120053            LCALL   RECEIVE
00E3   9582              SUBB    A,        DPL
00E5   5001              JNC     LO1
00E7   19                DEC     R1
00E8   F8        LO1:    MOV     R0,A
00E9   09                INC     R1
00EA   120053    LO2:    LCALL   RECEIVE
00ED   F0                MOVX    @DPTR,    A
00EE   A3                INC     DPTR
00EF   D8F9              DJNZ    R0,       LO2
00F1   D9F7              DJNZ    R1,       LO2
00F3   020088    ERR1:   LJMP    BEGIN
;
00F6   E573      MAINSFR:MOV     A,73H             ;PSW
00F8   12004B            LCALL   SEND
00FB   E578              MOV     A,78H              ;R0
00FD   12004B            LCALL   SEND
0100   E579              MOV     A,79H              ;R1
0102   12004B            LCALL   SEND
0105   EA                MOV     A,R2
0106   12004B            LCALL   SEND
0109   EB                MOV     A,R3
010A   12004B            LCALL   SEND
010D   EC                MOV     A,R4
```

```
010E  12004B            LCALL   SEND
0111  ED                MOV     A,R5
0112  12004B            LCALL   SEND
0115  EE                MOV     A,R6
0116  12004B            LCALL   SEND
0119  EF                MOV     A,R7
011A  12004B            LCALL   SEND
011D  E574              MOV     A,74H           ;ACC
011F  12004B            LCALL   SEND
0122  E575              MOV     A,75H           ;B
0124  12004B            LCALL   SEND
0127  E572              MOV     A,72H           ;SP
0129  12004B            LCALL   SEND
012C  E58C              MOV     A,TH0
012E  12004B            LCALL   SEND
0131  E58A              MOV     A,TL0
0133  12004B            LCALL   SEND
0136  E58D              MOV     A,TH1
0138  12004B            LCALL   SEND
013B  E58B              MOV     A,TL1
013D  12004B            LCALL   SEND
0140  E570              MOV     A,70H           ;PCH
0142  12004B            LCALL   SEND
0145  E571              MOV     A,71H           ;PCL
0147  12004B            LCALL   SEND
014A  E576              MOV     A,76H           ;DPH
014C  12004B            LCALL   SEND
014F  E577              MOV     A,77H           ;DPL
0151  12004B            LCALL   SEND
0154  D56905              DJNZ    69H,MM1
0157  0569                INC     69H
0159  020088      MM:     LJMP    BEGIN
015C  22          MM1:    RET
;
015D  120053      DISSA:  LCALL   RECEIVE
0160  F583              MOV     DPH,A
0162  120053            LCALL   RECEIVE
0165  F582              MOV     DPL,A
0167  12016D            LCALL   DI1
016A  020088            LJMP    BEGIN
016D  120053      DI1:    LCALL   RECEIVE
0170  B4020F            CJNE    A,#2H,DI4
0173  E4          DI2:    CLR     A
0174  93                MOVC    A,@A+DPTR
0175  12004B            LCALL   SEND
0178  A3                INC     DPTR
0179  E4          DI3:    CLR     A
017A  93                MOVC    A,@A+DPTR
017B  12004B            LCALL   SEND
017E  A3                INC     DPTR
017F  02016D            LJMP    DI1
0182  40F5        DI4:    JC      DI3
0184  22          DI5:    RET
;
0185  E578        LRAM:   MOV     A,78H           ;R0
0187  12004B            LCALL   SEND
018A  E579              MOV     A,79H           ;R1
018C  12004B            LCALL   SEND
018F  7802              MOV     R0,     #2H
0191  E6          LR1:    MOV     A,      @R0
```

```
0192  12004B          LCALL   SEND
0195  08              INC     R0
0196  B880F8          CJNE    R0,#80H,LR1
0199  020088    LR2:  LJMP    BEGIN
.
019C  E580      LSFR: MOV     A,P0
019E  12004B          LCALL   SEND
01A1  E572            MOV     A,72H           ;SP
01A3  12004B          LCALL   SEND
01A6  E577            MOV     A,77H           ;DPL
01A8  12004B          LCALL   SEND
01AB  E576            MOV     A,76H           ;DPH
01AD  12004B          LCALL   SEND
01B0  E587            MOV     A,PCON
01B2  12004B          LCALL   SEND
01B5  E588            MOV     A,TCON
01B7  12004B          LCALL   SEND
01BA  E589            MOV     A,TMOD
01BC  12004B          LCALL   SEND
01BF  E58A            MOV     A,TL0
01C1  12004B          LCALL   SEND
01C4  E58B            MOV     A,TL1
01C6  12004B          LCALL   SEND
01C9  E58C            MOV     A,TH0
01CB  12004B          LCALL   SEND
01CE  E58D            MOV     A,TH1
01D0  12004B          LCALL   SEND
01D3  E590            MOV     A,P1
01D5  12004B          LCALL   SEND
01D8  E598            MOV     A,SCON
01DA  12004B          LCALL   SEND
01DD  E599            MOV     A,SBUF
01DF  12004B          LCALL   SEND
01E2  E5A0            MOV     A,P2
01E4  12004B          LCALL   SEND
01E7  E5A8            MOV     A,IE
01E9  12004B          LCALL   SEND
01EC  E5B0            MOV     A,P3
01EE  12004B          LCALL   SEND
01F1  E5B8            MOV     A,IP
01F3  12004B          LCALL   SEND
01F6  E573            MOV     A,73H           ;PSW
01F8  12004B          LCALL   SEND
01FB  E574            MOV     A,74H           ;ACC
01FD  12004B          LCALL   SEND
0200  E575            MOV     A,75H           ;B
0202  12004B          LCALL   SEND
0205  E571            MOV     A,71H           ;PCL
0207  12004B          LCALL   SEND
020A  E570            MOV     A,70H           ;PCH
020C  12004B          LCALL   SEND
020F  020088          LJMP    BEGIN
.
0212  120053    LP:   LCALL   RECEIVE
0215  F583            MOV     DPH,A
0217  120053          LCALL   RECEIVE
021A  F582            MOV     DPL,A
021C  7800            MOV     R0,#0H
021E  E0        LP1:  MOVX    A,@DPTR
021F  12004B          LCALL   SEND
```

```
0222  A3                    INC     DPTR
0223  D8F9                  DJNZ    R0,LP1
0225  020088                LJMP    BEGIN
;
0228  120053      MRAM:     LCALL   RECEIVE
022B  B4FF03                CJNE    A,#0FFH,MR1
022E  020088                LJMP    BEGIN
0231  B40007      MR1:      CJNE    A,#00H,MR2
0234  120053                LCALL   RECEIVE
0237  F578                  MOV     78H,A
0239  80ED                  SJMP    MRAM
023B  B40107      MR2:      CJNE    A,#1H,MR3
023E  120053                LCALL   RECEIVE       ;R0
0241  F579                  MOV     79H,A
0243  80E3                  SJMP    MRAM
0245  F8          MR3:      MOV     R0,A
0246  120053                LCALL   RECEIVE
0249  F6                    MOV     @R0,A
024A  80DC                  SJMP    MRAM
;
024C  120053      MSFR:     LCALL   RECEIVE
024F  B4FF03                CJNE    A,#0FFH,MS1
0252  020088                LJMP    BEGIN
0255  B48207      MS1:      CJNE    A,#82H,MS2
0258  120053                LCALL   RECEIVE       ;DPL
025B  F577                  MOV     77H,A
025D  80ED                  SJMP    MSFR
025F  B48307      MS2:      CJNE    A,#83H,MS3
0262  120053                LCALL   RECEIVE       ;DPH
0265  F576                  MOV     76H,A
0267  80E3                  SJMP    MSFR
0269  B4D007      MS3:      CJNE    A,#0D0H,MS4
026C  120053                LCALL   RECEIVE       ;PSW
026F  F573                  MOV     73H,A
0271  80D9                  SJMP    MSFR
0273  B4E007      MS4:      CJNE    A,#0E0H,MS5
0276  120053                LCALL   RECEIVE       ;ACC
0279  F574                  MOV     74H,A
027B  80CF                  SJMP    MSFR
027D  B4F007      MS5:      CJNE    A,#0F0H,MS6
0280  120053                LCALL   RECEIVE       ;B
0283  F575                  MOV     75H,A
0285  80C5                  SJMP    MSFR
0287  B4F107      MS6:      CJNE    A,#0F1H,MS7
028A  120053                LCALL   RECEIVE       ;PCL
028D  F571                  MOV     71H,A
028F  80BB                  SJMP    MSFR
0291  B4F207      MS7:      CJNE    A,#0F2H,MS8
0294  120053                LCALL   RECEIVE       ;PCH
0297  F570                  MOV     70H,A
0299  80B1                  SJMP    MSFR
029B  B48107      MS8:      CJNE    A,#81H,MS9    ;SP
029E  120053                LCALL   RECEIVE
02A1  F572                  MOV     72H,A
02A3  80A7                  SJMP    MSFR
02A5  B48007      MS9:      CJNE    A,#80H,MS10
02A8  120053                LCALL   RECEIVE
02AB  F580                  MOV     P0,A
02AD  809D                  SJMP    MSFR
02AF  B49007      MS10:     CJNE    A,#90H,MS11
```

```
02B2  120053          LCALL   RECEIVE
02B5  F590            MOV     P1,A
02B7  8093            SJMP    MSFR
02B9  B4A007  MS11:   CJNE    A,#0A0H,MS12
02BC  120053          LCALL   RECEIVE
02BF  F5A0            MOV     P2,A
02C1  8089            SJMP    MSFR
02C3  B4B007  MS12:   CJNE    A,#0B0H,MS13
02C6  120053          LCALL   RECEIVE
02C9  F5B0            MOV     P3,A
02CB  414C            AJMP    MSFR
02CD  B4B807  MS13:   CJNE    A,#0B8H,MS14
02D0  120053          LCALL   RECEIVE
02D3  F5B8            MOV     IP,A
02D5  414C            AJMP    MSFR
02D7  B4A807  MS14:   CJNE    A,#0A8H,MS15
02DA  120053          LCALL   RECEIVE
02DD  F5A8            MOV     IE,A
02DF  414C            AJMP    MSFR
02E1  B48907  MS15:   CJNE    A,#089H,MS16
02E4  120053          LCALL   RECEIVE
02E7  F589            MOV     TMOD,A
02E9  414C            AJMP    MSFR
02EB  B48807  MS16:   CJNE    A,#88H,MS17
02EE  120053          LCALL   RECEIVE
02F1  F588            MOV     TCON,A
02F3  414C            AJMP    MSFR
02F5  B48C07  MS17:   CJNE    A,#8CH,MS18
02F8  120053          LCALL   RECEIVE
02FB  F58C            MOV     TH0,A
02FD  414C            AJMP    MSFR
02FF  B48A07  MS18:   CJNE    A,#8AH,MS19
0302  120053          LCALL   RECEIVE
0305  F58A            MOV     TL0,A
0307  414C            AJMP    MSFR
0309  B48D07  MS19:   CJNE    A,#08DH,MS20
030C  120053          LCALL   RECEIVE
030F  F58D            MOV     TH1,A
0311  414C            AJMP    MSFR
0313  B48B07  MS20:   CJNE    A,#08BH,MS21
0316  120053          LCALL   RECEIVE
0319  F58B            MOV     TL1,A
031B  414C            AJMP    MSFR
031D  B49807  MS21:   CJNE    A,#098H,MS22
0320  120053          LCALL   RECEIVE
0323  F598            MOV     SCON,A
0325  414C            AJMP    MSFR
0327  B49907  MS22:   CJNE    A,#099H,MS23
032A  120053          LCALL   RECEIVE
032D  F599            MOV     SBUF,A
032F  414C            AJMP    MSFR
0331  B48705  MS23:   CJNE    A,#087H,MS24
0334  120053          LCALL   RECEIVE
0337  F587            MOV     PCON,A
0339  414C    MS24:   AJMP    MSFR
      .
033B  120053  MP:     LCALL   RECEIVE
033E  F583            MOV     DPH,A
0340  120053          LCALL   RECEIVE
0343  F582            MOV     DPL,A
0345  E0      MP1:    MOVX    A,@DPTR
0346  F5F0            MOV     B,A
0348  12004B          LCALL   SEND
```

```
034B  120053            LCALL   RECEIVE
034E  B5F003            CJNE    A,B,MP2
0351  020088            LJMP    BEGIN
0354  F0        MP2:    MOVX    @DPTR,A
0355  A3                INC     DPTR
0356  80ED              SJMP    MP1
;
0358  120053      ASSEM:  LCALL RECEIVE
035B  F583                MOV     DPH,A
035D  120053              LCALL   RECEIVE
0360  F582                MOV     DPL,A
0362  120053      AS1:    LCALL   RECEIVE
0365  B40403              CJNE    A,#4H,AS2
0368  020088              LJMP    BEGIN
036B  F8          AS2:    MOV     R0,A
036C  120053      AS3:    LCALL   RECEIVE
036F  F0                  MOVX    @DPTR,A
0370  A3                  INC     DPTR
0371  D8F9                DJNZ    R0,AS3
0373  80ED                SJMP    AS1
;
0375  5388FE      STEP:   ANL     TCON,#0FEH
0378  75B801               MOV     IP,#01H
037B  E572                 MOV    A,72H
037D  858172               MOV    72H,SP
0380  F581                 MOV    SP,A
0382  8573D0               MOV     PSW,73H
0385  8574E0               MOV     ACC,74H
0388  8575F0               MOV     B,75H
038B  857683               MOV     DPH,76H
038E  857782               MOV     DPL,77H
0391  A878                 MOV     R0,78H
0393  A979                 MOV     R1,79H
0395  C071                 PUSH    71H             ;PCL
0397  C070                 PUSH    70H             ;PCH
0399  756902                 MOV     69H,#02H
039C  43A881               ORL     IE,#81H
039F  C2B2                 CLR     P3.2
03A1  32                   RETI
03A2  53A87F      STEP1:  ANL     IE,#7FH
03A5  D070                 POP     70H             ;PCH
03A7  D071                 POP     71H             ;PCL
03A9  858277               MOV     77H,DPL
03AC  858376               MOV     76H,DPH
03AF  85F075               MOV     75H,B
03B2  85E074               MOV     74H,ACC
03B5  85D073               MOV     73H,PSW
03B8  8878                 MOV     78H,R0
03BA  8979                 MOV     79H,R1
03BC  E572                 MOV    A,72H
03BE  858172               MOV    72H,SP
03C1  F581                 MOV    SP,A
03C3  22          ST2:    RET
;
03C4  120053      SSTEP:  LCALL   RECEIVE
03C7  F570                MOV     70H,A
03C9  120053              LCALL   RECEIVE
03CC  F571                MOV     71H,A
03CE  120375              LCALL   STEP
03D1  0200F6              LJMP    MAINSFR
;
03D4  120053      MULTSTEP:LCALL  RECEIVE
03D7  F570                MOV     70H,A
```

```
03D9  120053          LCALL   RECEIVE
03DC  F571            MOV     71H,A
03DE  120053          LCALL   RECEIVE
03E1  F57F            MOV     7FH,A
03E3  120375    MU1:  LCALL   STEP
03E6  D57FFA          DJNZ    7FH,MU1
03E9  0200F6          LJMP    MAINSFR
;
03EC  120053    BREAK: LCALL  RECEIVE
03EF  F570            MOV     70H,A
03F1  120053          LCALL   RECEIVE
03F4  F571            MOV     71H,A
03F6  120053          LCALL   RECEIVE
03F9  F57D            MOV     7DH,A
03FB  120053          LCALL   RECEIVE
03FE  F57C            MOV     7CH,A
0400  120375    BR1:  LCALL   STEP
0403  E571            MOV     A,71H
0405  B57CF8          CJNE    A,7CH,BR1
0408  E570            MOV     A,70H
040A  B57DF3          CJNE    A,7DH,BR1
040D  0200F6          LJMP    MAINSFR
;
0410  120053    UNEXEC: LCALL RECEIVE
0413  F570            MOV     70H,A
0415  120053          LCALL   RECEIVE
0418  F571            MOV     71H,A
041A  120053          LCALL   RECEIVE
041D  F583            MOV     DPH,A
041F  120053          LCALL   RECEIVE
0422  F582            MOV     DPL,A
0424  E0              MOVX    A,@DPTR
0425  F57B            MOV     7BH,A
0427  7402            MOV     A,#02H
0429  F0              MOVX    @DPTR,A
042A  A3              INC     DPTR
042B  E0              MOVX    A,@DPTR
042C  F57A            MOV     7AH,A
042E  7400            MOV     A,#00H
0430  F0              MOVX    @DPTR,A
0431  A3              INC     DPTR
0432  E0              MOVX    A,@DPTR
0433  F57E            MOV     7EH,A
0435  7416            MOV     A,#16H
0437  F0              MOVX    @DPTR,A
0438  85837D          MOV     7DH,DPH
043B  85827C          MOV     7CH,DPL
043E  857683          MOV     DPH,76H
0441  857782          MOV     DPL,77H
0444  8573D0          MOV     PSW,73H
0447  8574E0          MOV     ACC,74H
044A  8575F0          MOV     B,75H
044D  A878            MOV     R0,78H
044F  A979            MOV     R1,79H
0451  E572            MOV     A,72H
0453  858172          MOV     72H,SP
0456  F581            MOV     SP,A
0458  C071            PUSH    71H
045A  C070            PUSH    70H
045C  22              RET
;
045D  1582      DDP:  DEC     DPL
045F  4001            JC      DP1
```

```
0461   22                RET
0462   1583       DP1:   DEC      DPH
0464   22                RET
;
0465   858376     UNEXEC1:MOV     76H,DPH
0468   858277            MOV      77H,DPL
046B   85F075             MOV      75H,B
046E   85E074             MOV      74H,ACC
0471   85D073             MOV      73H,PSW
0474   8878               MOV      78H,R0
0476   8979               MOV      79H,R1
0478   857D83             MOV      DPH,7DH
047B   857C82             MOV      DPL,7CH
047E   E57E               MOV      A,7EH
0480   F0                 MOVX     @DPTR,A
0481   12045D             LCALL    DDP
0484   E57A               MOV      A,7AH
0486   F0                 MOVX     @DPTR,A
0487   12045D             LCALL    DDP
048A   E57B               MOV      A,7BH
048C   F0                 MOVX     @DPTR,A
048D   858370             MOV      70H,DPH
0490   858271             MOV      71H,DPL
0493   E572               MOV     A,72H
0495   858172             MOV     72H,SP
0498   F581               MOV     SP,A
049A   0200F6             LJMP     MAINSFR
;
049D   120053     EXEC:   LCALL    RECEIVE
04A0   F570               MOV      70H,A
04A2   120053             LCALL    RECEIVE
04A5   F571               MOV      71H,A
04A7   857683             MOV      DPH,76H
04AA   857782             MOV      DPL,77H
04AD   8573D0             MOV      PSW,73H
04B0   8574E0             MOV      ACC,74H
04B3   8575F0             MOV      B,75H
04B6   A878               MOV      R0,78H
04B8   A979               MOV      R1,79H
04BA   E572               MOV     A,72H
04BC   858172             MOV     72H,SP
04BF   F581               MOV     SP,A
04C1   C071               PUSH     71H
04C3   C070               PUSH     70H
04C5   22                 RET
;
04C6   120053     TRACE : LCALL    RECEIVE
04C9   F570               MOV      70H,A
04CB   120053             LCALL    RECEIVE
04CE   F571               MOV      71H,A
04D0   120053     TRR1:    LCALL    RECEIVE
04D3   B4FF03             CJNE     A,#0FFH,TR2
04D6   020088             LJMP     BEGIN
04D9   120375     TR2:    LCALL    STEP
04DC   756902              MOV      69H,#02H
04DF   1200F6             LCALL    MAINSFR
04E2   857083             MOV      DPH,70H
04E5   857182             MOV      DPL,71H
04E8   756902              MOV      69H,#02H
04EB   12016D             LCALL    DI1
04EE   80E0               SJMP     TRR1
04F0   00         HF:     NOP
```

KDC-Ⅲ开发机 (PC机方面程序)

```
FILE NAME ASI.PAS
program abc(input,output);
  uses crt,dos,asm,public;
   const number:string[16]='0123456789ABCDEF';
         esc:char=chr(27);
    var r0,r1,r2,r3,r4,r5,r6,r7,r8,acc,b,sp,psw,th0,tl0,th1,
        tl1,pch,pcl,dph,dpl,lj:byte;
        try:word;
        dim:array [1..200] of byte;
        ram:array [0..$7f] of byte;
        error,break:boolean;
        ch:char;
        hex,ten:set of char;

  procedure  ks;
    begin
         port[$3fb]:=$83;
         port[$3f9]:=$0;
         port[$3f8]:=$60;
         port[$3fb]:=$03;
         port[$3fd]:=$40;
    end;

  procedure receive(var a:byte);
    var i:longint;
    begin
     i:=0;
     while ((port[$3fd] and $01)<>$01) and
           ((i<200000) or break) do inc(i);
     if (i<200000) or break then a:=port[$3f8]
     else begin
             clrscr;
             writeln;
             writeln('Communication Faile.');
             writeln;
             writeln('Please RESET 8031 and
                     restart ON-LINE EMULATOR');
             writeln;
             halt;
         end;

    end;

  procedure receiven(n:integer);
   var  i:integer;
        j:longint;
   begin
        for i:=1 to n do
            begin
                 j:=0;
                 while ((port[$3fd] and $01)<>$01) and
                       (j<200000) do inc(j);
                 if j<200000 then dim[i]:=port[$3f8]
                 else begin
                     clrscr;
                     writeln;
                     writeln('Communication Faile.');
                     writeln;
                     writeln('Please RESET 8031 and
                             restart ON-LINE EMULATOR');
```

```
        writeln;
        halt;
        end;
procedure err;
  begin
        gotoxy(48,25);
        write('Error Input.Try Again.');
        sound(800);
        delay(200);
        nosound;
        error:=true;
  end;

procedure nerr;
  begin
        gotoxy(48,25);
        write(' ':25);
        error:=false;
  end;

procedure  load;
  var file1:file of char;
      st:string[12];
      a,b,x:byte;
      m,n:char; ch:string[1];
  begin
        repeat
         upline(1,1,1,18,39);
         gotoxy(3,19);
         write('Input object file:');
         readln(st);
         {$i-}
         assign(file1,st);
         reset(file1);
         x:=ioresult;
         {$i+}
         upline(1,1,1,18,39);
         gotoxy(3,19);
         if x=0 then
           begin
             write('Waiting!');
             send($1);
             while not eof(file1) do
               begin
                    read(file1,m);
                    if not eof(file1) then
                       begin
                        read(file1,n);
                        change(m,a);change(n,b);
                        a:=a*$10+b;
                        send(a);
                       end;
               end;
            close(file1);
            upline(1,1,1,18,39);
            gotoxy(3,19);
            write('O.K.');
           end
        else write('File ',st,' con not be opened.');
        upline(1,1,1,18,39);
        gotoxy(3,19);
        write('Continue (Y/N)?');
        ch:=readkey;
   until (ch<>'y') and (ch<>'Y');
```

```
 end;

procedure save;
  var addr1,addr2,x,i:word;a,ad1:byte;
      ch:char;
      file1:file of char;
      st:string[12];
  procedure printf(a:byte);
      var   x,y:integer;
      begin
           x:=a div 16;
           y:=a mod 16;
           write(file1,number[x+1],number[y+1]);
      end;
  begin
    upline(1,1,1,18,39);
    gotoxy(3,19);writeln('Input object
                          file name (*.obj):');
    upline(1,1,1,18,39);gotoxy(3,19);
    readln(st);if st='' then exit;
    send($d);
    assign(file1,st);
    rewrite(file1);
    upline(1,1,1,18,39);
    gotoxy(3,19);write('Input begin address:');
    ch:=readkey;write(ch);
    addr1:=0;
    repeat
          change(ch,ad1);
          addr1:=addr1*$10+ad1;
          ch:=readkey;write(ch);
    until not (ch in ['0'..'9','a'..'f','A'..'F']);
    upline(1,1,1,18,39);
    gotoxy(3,19);write('Input end address:');
    ch:=readkey;write(ch);
    addr2:=0;
    repeat
          change(ch,ad1);
          addr2:=addr2*$10+ad1;
          ch:=readkey;write(ch);
    until not (ch in ['0'..'9','a'..'f','A'..'F']);
    upline(1,1,1,18,39);
    gotoxy(3,19);write('Waiting...');
    send(hi(addr1));delay(50);
    printf(hi(addr1));printf(lo(addr1));
    printf(hi(addr2));printf(lo(addr2));
    send(lo(addr1));  delay(50);
    if addr2<=addr1 then begin send($3);close(file1)
                             ;exit end;
    x:=addr2-addr1;
    for i:=1 to x do
        begin
             send($1);
             receive(a);
             printf(a);
       end;
    close(file1);
    send($3);
    upline(1,1,1,18,39);
    gotoxy(3,19);write('O.K.');
 end;

procedure mainsfr;
    begin
```

```
      receive(psw);
      receive(r0);receive(r1);receive(r2);receive(r3);
      receive(r4);receive(r5);receive(r6);receive(r7);
      receive(acc);receive(b);receive(sp);
      receive(th0);receive(tl0);receive(th1);
                   receive(tl1);
      receive(pch);receive(pcl);receive(dph);
                   receive(dpl);
      gotoxy(48,15);print(r0);
      gotoxy(56,15);print(r1);
      gotoxy(64,15);print(r2);
      gotoxy(72,15);print(r3);
      gotoxy(48,16);print(r4);
      gotoxy(56,16);print(r5);
      gotoxy(64,16);print(r6);
      gotoxy(72,16);print(r7);
      gotoxy(47,17);print(acc);
      gotoxy(56,17);print(b);
      gotoxy(64,17);print(sp);
      gotoxy(73,17);print(psw);
      gotoxy(48,18);print(th0);
      gotoxy(50,18);print(tl0);
      gotoxy(64,18);print(th1);
      gotoxy(66,18);print(tl1);
      gotoxy(48,19);print(pch);
      gotoxy(50,19);print(pcl);
      gotoxy(66,19);print(dph);
      gotoxy(68,19);print(dpl);
   end;
procedure page3;
  var s1,s2,s3,s4,s5,s6,s7,s8,s9:string[80];
      a,b,i:integer;
  begin
      clrscr;
      s5:='ZDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD?';
      s6:='3                                                3';
      s7:='@DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDY';
          a:=a+1;
       a:=1;b:=1;
      s2:='ZDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD?';
      gotoxy(b,a);
      write(s2+' '+s5);
      s3:='3                                                3';
      repeat
      a:=a+1;
      gotoxy(b,a);
      write(s3+' '+s6);until a=12;
      a:=a+1;
      gotoxy(b,a);write(s3+' '+s7);
      a:=a+1;gotoxy(b,a);write(s3+' '+s5);
      for i:=1 to 5 do begin
                           a:=a+1;
                           gotoxy(b,a);
                           write(s3+' '+s6);
                       end;
      s4:='@DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDY';
                                          a:=a+1;
      gotoxy(b,a);
      write(s4+' '+s7);
      gotoxy(45,15);
      s7:='R0=     R1=     R2=     R3=';write(s7);
      gotoxy(45,16);
      S8:='R4=     R5=     R6=     R7=';write(s8);
      gotoxy(45,17);
```

```
      s9:='A=       B=       SP=      PSW=';write(s9);
       gotoxy(45,18);
       s1:='T0=         H      T1=        H';write(s1);
       gotoxy(45,19);
       s2:='PC=                DPTR=';write(s2);
       textcolor(0);
       textbackground(7);
       gotoxy(1,22);write(' 0 Deassembler1 Trace
                           2 Multipstep ');
       gotoxy(1,23);write(' 3 List        4 mOdify
                           5 Singlestep ');
       gotoxy(1,24);write(' 6 Execute     7 Unexecbreak
                           8 execBreak  ');
       gotoxy(1,25);write(' 9 Assembler   A Fileoptions
                           B Quit       ');
       textcolor(7);
       textbackground(0);
       gotoxy(46,22);write(' Press Capitle letter to
                                        choose:');
       gotoxy(46,24);write(' >>');
end;
procedure dissa1(var i:integer;a,b:byte);
  var x,m,n:byte;
  begin
     send($1);if i=0 then delay(100);
     receive(x);print(x);
     i:=1;
     case x of
      $00:write('      NOP              ');
      $01,$21,$41,$61,$81,$A1,$C1,$E1:BEGIN
                        SEND($1);
                        RECEIVE(M);print(m);
                        write('   AJMP   ');
                        PRINT((A div 8)*8+x div $20);
                        PRINT(M);I:=2;
                    END;
      $02:BEGIN
             SEND($2);
             RECEIVE(M); print(m);
             RECEIVE(N); print(n);
             write(' LJMP   ');
             PRINT(M);
             PRINT(N);
             1:=3;
          END;
      $03:write('      RR      A       ');
      $04:write('      INC     A       ');
      $05:BEGIN
             SEND($1);
             RECEIVE(M);print(m);
             write('    INC     ');
             PRINT(M);
             I:=2;
          END;
      $06,$07:write('      INC     @R',X-$06,' ':6);
      $08,$09,$A,$B,$C,$D,$E,$F:
         write('      INC     R',X-$8,' ':6);
      $10:BEGIN
             SEND($2);
             RECEIVE(M);print(m);
             RECEIVE(N);print(n);
             write(' JBC     ');
             PRINT(M);
```

```
          WRITE(','); print(a);
          PRINT(N+B+3);
          I:=3;
     END;
$11,$31,$51,$71,$91,$B1,$D1,$F1:BEGIN
               SEND($1);
               RECEIVE(M);print(m);
               write('   ACALL  ');
               PRINT((A div 8)*8+(x div $20));
               PRINT(M);
               I:=2;
             END;
$12:BEGIN
          SEND($2);
          RECEIVE(M); print(m);
          RECEIVE(N); print(n);
          write(' LCALL  ');
          PRINT(M);
          PRINT(N);
          I:=3;
     END;
$013:write('     RRC    A     ');
$14:write('     DEC    A     ');
$15:BEGIN
          SEND($1);
          RECEIVE(M);print(m);
          write('   DEC    ');
          PRINT(M);
          I:=2;
     END;
$16,$17:write('     DEC    @R',X-$16,' ':6);
$18,$19,$1A,$1B,$1C,$1D,$1E,$1F:
     write('     DEC    R',X-$18,' ':6);
$20:BEGIN
          SEND($2);
          RECEIVE(M); print(m);
          RECEIVE(N); print(n);
          write(' JB     ');
          PRINT(M);
          WRITE(',');
          PRINT(A);
          PRINT(N+B+3);
          I:=3;
     END;
$22:write('     RET        ');
$23:write('     RL     A   ');
$24:BEGIN
          SEND($1);
          RECEIVE(M);print(m);
          write('   ADD    A,#');
          PRINT(M);                I:=2
     END;
$25:BEGIN
          SEND($1);
          RECEIVE(M); print(m);
          write('   ADD    A,');
          PRINT(M);                I:=2
     END;
$26,$27:write('     ADD    A,@R',X-$26,' ':4);
$28,$29,$2A,$2B,$2C,$2D,$2E,$2F:
     write('     ADD    A,R',X-$28,' ':4);
$30:BEGIN
          SEND($2);
```

```
            RECEIVE(M);print(m);
            RECEIVE(N);print(n);
            write(' JNB     ');
            PRINT(M);
            WRITE(',');
            PRINT(A);
            PRINT(N+B+3);                    I:=3
    END;
$32:write('      RETI          ');
$33:WRITE('      RLC     A     ');
$34:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    ADD     A,#');
            PRINT(M);              I:=2
    END;
$35:begin
            send($1);
            receive(m);print(m);
            write('    ADDC    A,');
            print(m);i:=2;
    end;
$36,$37:WRITE('      ADDC    @R',X-$36,' ':4);
$38,$39,$3A,$3B,$3C,$3D,$3E,$3F:
    WRITE('      ADDC    R',X-$38,' ':4);
$40:BEGIN
            SEND($1);
            RECEIVE(M); print(m);
            WRITE('    JC      ');
            PRINT(A);
            PRINT(M+B+2);          I:=2
    END;
$42:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    ORL     ');
            PRINT(M);
            WRITE(',A');I:=2
    END;
$43:BEGIN
            SEND($2);
            RECEIVE(M);print(m);
            RECEIVE(N);print(n);
            WRITE(' ORL     ');
            PRINT(M);
            WRITE(',#');
            PRINT(N);                        I:=3
    END;
$44:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    ORL     A,#');
            PRINT(M);                        I:=2
    END;
$45:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    ORL     A,');
            PRINT(M);                        I:=2
    END;
$46,$47:WRITE('      ORL     A,@R',X-$46,' ':4);
$48,$49,$4A,$4B,$4C,$4D,$4E,$4F:
    WRITE('      ORL     A,R',X-$48,' ':4);
```

```
$50:BEGIN
        SEND($01);
        RECEIVE(M);print(m);
        WRITE('   JNC    ');
        PRINT(A);
        PRINT(M+B+2);              I:=2
    END;
 $52:BEGIN
        SEND($1);
        RECEIVE(M);print(M);
        WRITE('   ANL    ');
        PRINT(M);
        WRITE(',A');               I:=2
    END;
 $53:BEGIN
        SEND($2);
        RECEIVE(M);print(m);
        RECEIVE(N);print(n);
        WRITE(' ANL    ');
        PRINT(M);
        WRITE(',#');
        PRINT(N);            I:=3
     END;
 $54:BEGIN
        SEND($1);
        RECEIVE(M);print(m);
        WRITE('   ANL    A,#');
        PRINT(M);                  I:=2
    END;
 $55:BEGIN
        SEND($1);
           RECEIVE(M);print(m);
           WRITE('   ANL    A,');
           PRINT(M);                       I:=2
    END;
 $56,$57:WRITE('     ANL    A,@R',X-$56,' ':4);
 $58,$59,$5A,$5B,$5C,$5D,$5E,$5F:
    WRITE('     ANL    A,R',X-$58,' ':4);
 $60:BEGIN
        SEND($1);
        RECEIVE(M);print(m);
        WRITE('   JZ     ');
        PRINT(A);
        PRINT(M+B+2);                      I:=2
    END;
 $62:BEGIN
        SEND($1);
        RECEIVE(M);print(m);
        WRITE('   XRL    ');
        PRINT(M);
        WRITE(',A');             I:=2
    END;
 $63:BEGIN
        SEND($2);
        RECEIVE(M);print(m);
        RECEIVE(N);print(n);
        WRITE(' XRL    ');
        PRINT(M);
        WRITE(',#');
        PRINT(N);            I:=3
     END;
 $64:BEGIN
        SEND($1);
```

```
            RECEIVE(M);print(m);
            WRITE('    XRL     A,#');
            PRINT(M);       I:=2
    END;
$65:BEGIN
            SEND($1);
               RECEIVE(M);print(m);
               WRITE('    ANL     A,');
               PRINT(M);       I:=2
    END;
$66,$67:WRITE('      XRL     A,@R',X-$56,' ':4);
$68,$69,$6A,$6B,$6C,$6D,$6E,$6F:
    WRITE('      XRL     A,R',X-$68,' ':4);
$70:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    JNZ     ');
            PRINT(A);
            PRINT(M+B+2);                   I:=2
    END;
$72:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    ORL     C,');
            PRINT(M);                       I:=2
    END;
$73:BEGIN
            WRITE('      JMP      @A+DPTR');
     END;
$74:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    MOV     A,#');
            PRINT(M);                   I:=2
    END;
$75:BEGIN
            SEND($2);
               RECEIVE(M);print(m);
               RECEIVE(N);print(n);
               WRITE(' MOV     ');
               PRINT(M);
               WRITE(',#');
               PRINT(N);        I:=3
    END;
$76,$77:BEGIN
                SEND($1);
                RECEIVE(M);print(m);
                WRITE('    MOV     @R',X-$66,',#');
                PRINT(M);               I:=2;
          END;
$78,$79,$7A,$7B,$7C,$7D,$7E,$7F:
         BEGIN
                SEND($1);
                RECEIVE(M);print(m);
                WRITE('    MOV     R',X-$78,',#');
                PRINT(M);I:=2
          END;
$80:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    SJMP    ');
            PRINT(A);
            PRINT(M+B+2);                   I:=2
```

```
    END;
$82:BEGIN
        SEND($1);
        RECEIVE(M);print(m);
        WRITE('    ANL     C,');
        PRINT(M);           I:=2;
    END;
$83:BEGIN
        WRITE('      MOVC     A,@A+PC');
     END;
$84:WRITE('      DIV     AB     ');
$85:BEGIN
        SEND($2);
           RECEIVE(M);print(m);
           RECEIVE(N);print(n);
           WRITE(' MOV     ');
           PRINT(n);
           WRITE(',');
           PRINT(m);                I:=3;
    END;
$86,$87:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    MOV     ');
            PRINT(M);
            WRITE(',@R',X-$86);   I:=2;
        END;
$88,$89,$8A,$8B,$8C,$8D,$8E,$8F:
        BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    MOV     ');
            PRINT(M);
            WRITE(',R',X-$88); I:=2;
        END;
$90:BEGIN
        SEND($2);
        RECEIVE(M);print(m);
        receive(n);print(n);
        WRITE(' MOV     DPTR,#');
        PRINT(M);print(n);
        I:=3;
    END;
$92:BEGIN
        SEND($1);
        RECEIVE(M);print(m);
        WRITE('    MOV     ');
        PRINT(M);
        WRITE(',C');
        I:=2;
    END;
$93:WRITE('      MOVC    A,@A+DPTR');
$94:BEGIN
        SEND($1);
        RECEIVE(M);print(m);
        WRITE('    SUBB    A,#');
        PRINT(M);
        I:=2;
    END;
$95:BEGIN
        SEND($1);
           RECEIVE(M);print(m);
           WRITE('    SUBB    A,');
```

```
                PRINT(M);
                I:=2;
    END;
$96,$97:WRITE('      SUBB   A,@R',X-$96,' ':4);
$98,$99,$9A,$9B,$9C,$9D,$9E,$9F:
    WRITE('      SUBB   A,R',X-$98,' ':4);
$A0:BEGIN
          SEND($1);
          RECEIVE(M);print(m);
          WRITE('   ORL    C,');
          PRINT(M);
          I:=2;
    END;
$A2:BEGIN
          SEND($1);
          RECEIVE(M);print(m);
          WRITE('   MOV    C,');
          PRINT(M);
          I:=2;
    END;
 $A3:WRITE('      INC    DPTR    ');
 $A4:WRITE('      MUL    AB      ');
 $A5:WRITE(' ');
 $A6,$A7:BEGIN
               SEND($1);
               RECEIVE(M);print(m);
               WRITE('   MOV    @R',X-$A6);
               PRINT(M);
               I:=2;
          END;
 $A8,$A9,$AA,$AB,$AC,$AD,$AE,$AF:
          BEGIN
               SEND($1);
               RECEIVE(M);print(m);
               WRITE('   MOV    R',X-$A8,',');
               PRINT(M);
               I:=2;
          END;
 $B0:BEGIN
           SEND($1);
           RECEIVE(M);print(m);
           WRITE('   ANL    C,');
           PRINT(M);
           I:=2;
     END;
 $B2:BEGIN
           SEND($1);
           RECEIVE(M);print(M);
           WRITE('   CPL    ');
           PRINT(M);
           I:=2;
     END;
 $B3:WRITE('      CPL    C    ');
 $B4:BEGIN
           SEND($2);
           RECEIVE(M);print(m);
           RECEIVE(N);print(N);
           WRITE(' CJNZ   A,#');
           PRINT(M);WRITE(',');
           PRINT(A);PRINT(N+B+3);
           I:=3;
      END;
      $B5:BEGIN
```

```
            SEND($2);
            RECEIVE(M);print(m);
            RECEIVE(N);print(n);
            WRITE(' CJNZ    A,');
            PRINT(M);
            WRITE(',');    PRINT(A);
            PRINT(N+B+3);
            I:=3;
    END;
$B6,$B7:BEGIN
              SEND($2);
              RECEIVE(M);print(m);
              RECEIVE(N);print(n);
              WRITE(' CJNZ    @R',X-$B6,',#');
              PRINT(M);
              WRITE(','); PRINT(A);
              PRINT(N+B+3);
              I:=3;
            END;
$B8,$B9,$BA,$BB,$BC,$BD,$BE,$BF:
            BEGIN
                 SEND($2);
                 RECEIVE(M);print(m);
                 RECEIVE(N);print(n);
                 WRITE(' CJNZ    R',X-$B8,',#');
                 PRINT(M);
                 WRITE(',');
                 PRINT(A);
                 PRINT(N+B+3);
                 I:=3;
             END;
 $C0:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    PUSH    ');
            PRINT(M);
            I:=2;
     END;
 $C2:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    CLR     ');
            PRINT(M);
            I:=2;
     END;
 $C3:WRITE('      CLR     C     ');
 $C4:WRITE('      SWAP    A     ');
 $C5:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('    XCH     A,');
            PRINT(M);
```

```
            I:=2;
        END;
$C6,$C7:WRITE('      XCH    A,@R',X-$C6,' ':4);
$C8,$C9,$CA,$CB,$CC,$CD,$CE,$CF:
            WRITE('      XCH    A,R',X-$C8,' ':4);
$D0:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('   POP    ');
            PRINT(M);
            I:=2;
        END;
$D2:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('   SETB   ');
            PRINT(M);
            I:=2;
        END;
$D3:WRITE('      SETB   C      ');
$D4:WRITE('      DA     A      ');
$D5:BEGIN
            SEND($2);
            RECEIVE(M);print(m);
            RECEIVE(N);print(n);
            WRITE(' DJNZ   ');
            PRINT(M);
            WRITE(',');
            PRINT(A);PRINT(n+B+3);
            I:=3;
        END;
$D6,$D7:WRITE('      XCHD   A,@R',X-$D6,' ':4);
$D8,$D9,$DA,$DB,$DC,$DD,$DE,$DF:
            BEGIN
                SEND($1);
                RECEIVE(M);print(m);
                WRITE('   DJNZ   R',X-$D8,',');
                PRINT(A);PRINT(M+B+2);
                I:=2;
            END;
$E0:WRITE('      MOVX   A,@DPTR');
$E2,$E3:WRITE('      MOVX   A,@R',X-$E2,' ':4);
$E4:WRITE('      CLR    A      ');
$E5:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('   MOV    A,');
            PRINT(M);
            I:=2;
        END;
$E6,$E7:WRITE('      MOV    A,@R',X-$E6,' ':4);
$E8,$E9,$EA,$EB,$EC,$ED,$EE,$EF:
            WRITE('      MOV    A,R',X-$E8,' ':4);
$F0:WRITE('      MOVX   @DPTR,A');
$F2,$F3:WRITE('      MOVX   @R',X-$F2,',A    ');
$F4:WRITE('      CPL    A      ');
$F5:BEGIN
            SEND($1);
            RECEIVE(M);print(m);
            WRITE('   MOV    ');
            PRINT(M);
            WRITE(',A');
            I:=2;
```

```
                END;
    $F6,$F7:WRITE('      MOV    @R',X-$F6,',A    ');
    $F8,$F9,$FA,$FB,$FC,$FD,$FE,$FF:
            WRITE('      MOV    R',X-$F8,',A    ');
    end{case};
  end;
procedure dissasember;
  var i,m:integer;ch:char;sd:string[4];
      a,b,c:byte;
  begin
    gotoxy(49,24);
    write('Deassembler');
    repeat
        upline(1,1,1,18,39);
        gotoxy(2,19);
        write('Input first addr:');
        readln(sd);
        if error then nerr;
        if sd='' then exit;
        if not (sd[1] in hex) or not (sd[2] in hex) or
           not (sd[3] in hex) or
           not (sd[4] in hex) then err;
    until not error;
    change(sd[1],a);
    change(sd[2],b);
    a:=a*$10+b;
    change(sd[3],b);
    change(sd[4],c);
    b:=b*$10+c;
    send($d);delay(100);
    send(a);delay(100);
    send(b);delay(100);
    repeat
     upline(1,1,1,18,39);
     m:=0;
     for i:=1 to 16 do
       begin
          gotoxy(2,19);write(' ');
          print(a);print(b);write('  ');
          dissal(m,a,b);
          upline(1,1,1,18,39);
          if (b+m)>$ff then a:=a+1;
          b :=b+m;
       end;
     gotoxy(2,19);
     write('Continue(Y/N)?');
     ch:=readkey;write(ch);
    until (ch<>'Y') and (ch<>'y');
    send($3);upline(1,1,1,18,39);
  end;

procedure step;
  var a,b:byte;i,m:integer;
  begin
      gotoxy(49,24);
      write('Singlestep');
      lj:=port[$3fb];
      send($02);delay(200);
      send(pch);delay(100);
      send(pcl);
      mainsfr;
      upline(1,1,44,10,77);
      upline(0,8,44,11,77);
```

```
          send($0d); delay(200);
          send(pch); delay(100);
          send(pcl); a:=pch;b:=pcl;
          gotoxy(46,8);write('*');
          for i:=1 to 5 do
             begin
                 if i<>1 then begin
                                    gotoxy(46,i+7);
                                    write(' ');
                              end;
                 gotoxy(47,i+7);
                 print(a);
                 print(b);
                 write(' ');
                 dissal(m,a,b);
                 if (b+m)>$ff then  a:=a+1;
                 b:=b+m;
             end;
          send($03);
       end;

    procedure multstep;
       var  a:byte;
       begin
            gotoxy(49,24);
            write('Mutipstep');
            repeat
              upline(1,1,1,18,39);
              gotoxy(3,18);
              write('Input number(1..256):');
              readln(a);
              if error then nerr;
              if a=0 then exit;
              if (a<1) or (a>256) then err;
            until not error;
            send($03); delay(200);
            send(pch); delay(100);
            send(pcl); delay(100);
            lj:=port[$3f8];
            send(a);
            mainsfr;
            upline(2,1,1,18,39);
            gotoxy(3,19);
            write('O.K.');
       end;
    procedure unexecbreak;
       var  st:string[4];
            i,j,k:byte;
       begin
            break:=true;
            repeat
              upline(1,1,1,18,39);
              gotoxy(3,19);
              write('Input break address: ');
              readln(st);
              if error then nerr;
              if not (st[1] in hex) or not (st[2] in hex) or
                 not (st[3] in hex) or
                 not (st[4] in hex) then err;
            until not error;
            send(pch);delay(100);
            send(pcl);delay(100);
            change(st[1],i);change(st[2],j);
```

```
        i:=i*$10+j;
        change(st[3],j); change(st[4],k);
        j:=j*$10+k;
        lj:=port[$3f8];
        send(i);delay(100);
        send(j);
        upline(1,1,1,18,39);
        gotoxy(3,19);
        write('Waiting!');
        mainsfr;
        upline(2,1,1,18,39);
        gotoxy(3,18);
        write('O.K.');
        sound(800);
        delay(200);
        nosound;
        break:=false;
    end;

procedure exec;
    begin
       gotoxy(49,24);
       write('Execute');
       send($05); delay(200);
       send(pch); delay(100);
       send(pcl);
       gotoxy(49,24);
       write(' ':9);
    end;

procedure trace;
    var  m:integer;
         a,b:byte;
    begin
         gotoxy(49,24);
         write('Trace');
         send($06);delay(200);
         send(pch);delay(100);
         send(pcl);delay(100);
         a:=memw[$40:$1c];
         b:=memw[$40:$1c];
         while a=b do
               begin
                    lj:=port[$3f8];
                    send($11);
                    mainsfr;
                    upline(1,1,44,11,77);
                    gotoxy(46,12);write('*');
                    print(pch);
                    print(pcl);
                    write(' ');
                    dissa1(m,pch,pcl);
                    send($03);delay(50);
                    b:=memw[$40:$1c];
               end;
         send($ff);
    end;

    procedure listram;
      var ch:char;
          i,j:integer;cc:string[3];
      begin
         lj:=port[$3f8];
```

```
      upline(0,1,1,18,39);
      gotoxy(3,2);write('8031 ram:');
      send($7);
      for i:=0 to $7f do
          receive(ram[i]);
      for i:=0 to 7  do
          begin
          · gotoxy(6,i*2+3);write(i);write('0H==>');
            for j:=0 to 7 do
              begin
               gotoxy(13+j*3,i*2+3);print(ram[i*16+j]);
              end;

            gotoxy(6,i*2+4);write(i);write('8H==>');
            for j:=0 to 7 do
             begin
              gotoxy(13+j*3,i*2+4);print(ram[i*16+8+j]);
             end;
          end;
  end;
procedure listsfr;
   const
        st:array [1..23] of
           string[12]=('01:P0[80H]  ','02:SP[81H]  ',
        '03:DPL[82H] ','04:DPH[83H] ',
        '05:PCON[87H]','06:TCON[88H]','07:TMOD[89H]',

        '08:TL0[8AH] ','09:TL1[8BH] ','10:TH0[8CH] ',
        '11:TH1[8DH] ','12:P1[90H]  ',
        '13:SCON[98H]','14:SBUF[99H]','15:P2[A0H]  '
        ,'16:IE[A8H]  ','17:P3[B0H]  ',
        '18:IP[B8H]  ','19:PSW[D0H] ','20:ACC[E0H] '
        ,'21:B[F0H]   ','22:PCL[F1H] ',
        '23:PCH[F2H] ');
   var  a,i:byte;
   begin
        send($8);
        a:=port[$3f8];
        receiven(23);
        upline(0,1,1,18,39);
        gotoxy(5,2);
        write('8031 sfr:');
        a:=3;
       →for i:=1 to 11 do
            begin
                 a:=a+1;
                 gotoxy(5,a);write(st[i*2-1],'==');
                 print(dim[i*2-1]);
                 gotoxy(23,a);
                 write(st[i*2],'==');
                 print(dim[i*2]);
            end;
        gotoxy(5,a+1);write(st[23],'==');
        print(dim[23]);
  end;
procedure listp;
   var a,b,i,j,k,l:byte;
   c:string[4];
   begin
       send($9);
       repeat
         upline(1,1,1,18,39);
         gotoxy(2,18);
```

```
        write('Input First Address:  ');
        readln(c);
        if error then nerr;
        if not (c[1] in hex) or not (c[2] in hex) or
           not (c[3] in hex) or not (c[4] in hex) then err;
      until not error;
      change(c[1],i);change(c[2],k);
      i:=i*$10+k;
      change(c[3],j); change(c[4],l);
      j:=j*$10+l;
      lj:=port[$3f8];
      send(i);delay(100);
      send(j);
      receiven(128);   upline(0,1,1,18,39);
      a:=3;b:=2;
      for k:= 1 to 16 do
        begin
             b:=b+1;
             gotoxy(a,b);
             print(i);
             print(j);
             write(' -->');
             for l:=1 to 8 do
                 begin
                      print(dim[k*8+l-8]);write(' ');
                 end;
            if (j+8)>$ff then i:=i+1;
             j:=j+8;

        end;
      lj:=port[$3f8];
      end;

 procedure list;
    var  cc:string[3];
    begin
      gotoxy(49,24);
      write('List');
      repeat
        upline(1,1,1,18,39);
        gotoxy(3,19);write('RAM or SFR or P -->');
        readln(cc);
        if error then nerr;
        if (cc='ram') or (cc='RAM') then listram
            else if (cc='sfr') or (cc='SFR') then listsfr
            else if (cc='p') or (cc='P') then listp
            else if (cc<>'q') and (cc<>'Q') then err;
     until not error;
  end;

procedure modify;
   var  cc:string[3];
   procedure modram;
      var  st:string[2];
      h,l,t:byte;
      begin
           listram;
           send($a);
           textcolor(0);
           textbackground(7);
           gotoxy(5,19);write('Address: ');
           gotoxy(20,19);write('result: ');
           textcolor(7);
```

```
        textbackground(0);                                    mipad
        repeat
          if error then begin
                                gotoxy(14,19);
                                write('  ':5);
                        end;
          gotoxy(14,19);readln(st);
          if error then nerr;
          if not (st[1] in hex+['q','Q']) or
                        not (st[2] in hex+[' '])
             then err;
    until not error;
    while (st<>'q') and (st<>'Q') and (st<>'') do
       begin
          change(st[1],h);change(st[2],l);
          h:=h*$10+l;
          send(h);
          textcolor(16);
          textbackground(7);
          gotoxy(13+(h mod 8)*3,(h div 8)+3);
          print(ram[h]);
          textcolor(7);
          textbackground(0);
          repeat
            if error then begin
                                gotoxy(28,19);
                                write('  ':5);
                          end;
            gotoxy(28,19);readln(st);
            if error then nerr;
            if not (st[1] in hex) or not (st[2] in hex)
               then err;
            if st='' then err;
          until not error;
          change(st[1],t);change(st[2],l);
          t:=t*$10+l;
          send(t);
          ram[h]:=t;
          gotoxy(12+(h mod 8)*3,(h div 8)+3);
          write(st:3);
          if h<8 then begin
                        gotoxy((h mod 4)*8+48,(h div 4)+15);
                        write(st);
                      end;
          repeat
            gotoxy(14,19);write('  ':5);
            gotoxy(28,19);write('  ':5);
            gotoxy(14,19);readln(st);
            if error then nerr;
            if not (st[1] in hex+['q','Q']) or
               not (st[2] in hex+[' '])
               then err;
            if st='' then err;
          until not error;
        end;
        send($ff);
    end;
procedure   modsfr;
    const   ad:array [1..23] of byte=($80,$81,$82,$83,$87,
               $88,$89,$8a,$8b,$8c,$8d,$90,$98,$99,$a0,$a8,
               $b0,$b8,$d0,$e0,$f0,$f1,$f2);
    var   st:string[2];
          i,j,h,l:byte;
```

```
begin
     listsfr;
     send($b);
     textcolor(0);
     textbackground(7);
     gotoxy(5,17);write('No:= ');
     gotoxy(20,17);write('result:= ');
     textcolor(7);
     textbackground(0);
     repeat
      if error then begin
                         gotoxy(11,17);
                         write(' ':5);
                    end;
      gotoxy(11,17);
      readln(st);
      if error then nerr;
      if not (st[1] in ten+['q','Q']) or
                      not (st[2] in ten)
         then err;
     until not error;
     while (st<>'q') and (st<>'Q') and (st<>'') do
         begin
              change(st[1],i);change(st[2],j);
              i:=i*10+j;
              if (i<1) or (i>23) then
                 begin
                      gotoxy(48,25);
                      write('Error Input. ');
                      sound(800);
                      delay(200);
                      nosound;
                      delay(1000);
                      gotoxy(48,25);
                      write(' ':20);
                      send($ff);
                      exit;
                 end;
              send(ad[i]);
              textcolor(16);
              textbackground(7);
              gotoxy(37-(i mod 2)*18,(i+1) div 2+3);
              print(dim[i]);
              textcolor(7);
              textbackground(0);
              repeat
                gotoxy(29,17);readln(st);
                if error then nerr;
                if (st='') or
                   not (st[1] in hex) or
                   not (st[2] in hex)
                   then begin
                                gotoxy(29,17);
                                write(' ':6);
                                err;
                        end;
              until not error;
              change(st[1],h);change(st[2],l);
              h:=h*$10+l;
              send(h);
              gotoxy(37-(i mod 2)*18,(i+1) div 2+3);
              write(st);
              case i of
```

```
                        02:gotoxy(64,17);
                        03:gotoxy(68,19);
                        04:gotoxy(66,19);
                        08:gotoxy(50,18);
                        09:gotoxy(66,18);
                        10:gotoxy(48,18);
                        11:gotoxy(64,18);
                        19:gotoxy(73,17);
                        20:gotoxy(47,17);
                        21:gotoxy(56,17);
                        22:begin
                                gotoxy(50,19);
                                pcl:=h;
                           end;
                        23:begin
                                gotoxy(48,19);
                                pch:=h;
                           end;
                        else gotoxy(37-(i mod 2)*18,
                                    (i+1) div 2+3);
                    end;
                    dim[i]:=h;
                    write(st);
                    repeat
                      gotoxy(11,17);write('  ':5);
                      gotoxy(29,17);write('  ':5);
                      gotoxy(11,17);readln(st);
                      if error then nerr;
                      if not (st[1] in hex+['q','Q']) or
                                      not (st[2] in hex)
                         then err;
                      if st='' then err;
                    until not error;
                end;
            send($ff);
    end;
procedure modp;
  label 11;
  var   st2:string[4];
        st1:string[2];
        h,l,j,k,m,n,i:byte;
  begin
        l:=port[$3f8];
        send($c);
        repeat
          upline(0,1,1,18,39);
          gotoxy(3,3);write('Input First Adress:');
          readln(st2);
          if error then nerr;
          if not (st2[1] in hex) or
                   not (st2[2] in hex) or
             not (st2[3] in hex) or
                   not (st2[4] in hex) then err;
        until not error;
        change(st2[1],h);change(st2[2],l);
        h:=h*$10+l;
        change(st2[3],l);change(st2[4],j);
        l:=l*$10+j;
        send(h);delay(100);
        send(l);
        k:=4;
        repeat
          receive(i);
```

```
                gotoxy(8,k);print(h);print(l);
                write(' ==> ');print(i);write('--> ');
                readln(st1);
                if (st1<>'q') and (st1<>'Q') and (st1<>'') then
                     begin
                          change(st1[1],m);change(st1[2],n);
                          m:=m*$10+n;
                          send(m);
                          if m=i then goto 11;
                          if l+1>$ff then h:=h+1;
                          l:=l+1;
                          if k<18 then k:=k+1
                                  else upline(1,1,1,18,39);
                     end
                else  send(i);
             until (st1='q') or (st1='Q') or (st1='');
11:     end;

    begin
         gotoxy(49,24);
         write('Modify');
        repeat
           upline(1,1,1,18,39);
           gotoxy(3,19);write('RAM or SFR or P-->');
           readln(cc);
           if error then nerr;
           if (cc='ram') or (cc='RAM') then modram
           else if (cc='sfr') or (cc='SFR') then modsfr
           else if (cc='p') or (cc='P') then modp
           else if (cc<>'q') and (cc<>'Q') then err;
        until not error;
    end;

  procedure fo;
    begin
         upline(0,1,1,18,39);
         gotoxy(2,2);write('1.MOVE object file to 8031');
         gotoxy(2,3);write('2.SAVE object file from 8031');
         gotoxy(2,4);write('3.QUIT');
         gotoxy(2,6);write('Please select 1..3 : ');
         repeat
            gotoxy(24,6);
            ch:=readkey;
            if error then nerr;
            case ch of
                 '1','m','M':load;
                 '2','s','S':save;
                 '3',chr(27):exit;
                else err;
            end;
        until not error;
    end;

  procedure debug;
    var r0,r1,r2,r3,r4,r5,r6,r7,r8,acc,b,sp,psw,th0,tl0,th1,
        tl1,pch,pcl,dph,dpl:byte;
        ch,c:char;          i:integer;
   begin
        page3;
        lj:=port[$3f8];
        send($0);mainsfr;
        upline(0,1,1,18,39);
        upline(0,1,44,10,77);
```

```
        gotoxy(5,17);write('O.K.');
        repeat
          gotoxy(49,24);
          ch:=readkey;
          if error then nerr;
          case ch of
               chr(0):begin
                    c:=readkey;
                    case c of
                         chr(64):begin
                              clrscr;
                              writeln;
                              writeln('Type EXIT to return to
                                                  USTC-51...');
                              swapvectors;
                              exec(getenv('COMSPEC'),' ');
                              swapvectors;
                             end;
                         else err;
                    end;
               end;}
             's','S':step;
             'm','M':multstep;
             'u','U':begin
                          gotoxy(49,24);
                          write('Unexecbreak');
                          send($4);delay(200);
                          unexecbreak;
                     end;
             'b','B':begin
                          gotoxy(49,24);
                          write('Execbreak');
                          send($e);delay(200);
                          unexecbreak;
                     end;
             'e','E':exec;
             't','T':trace;
             'l','L':list;
             'o','O':modify;
             'd','D':dissasember;
             'a','A':begin
                          gotoxy(49,24);
                          write('Assembler');
                          assembler;
                     end;
             'f','F':begin
                          gotoxy(49,24);
                          write('Fileoption');
                          fo;
                     end;
             chr(27),'q','Q':send($f);
             else err;
          end;
        gotoxy(49,24);
        write(' ':11);
       until (ch='q') or (ch='Q') or (ord(ch)=27);
    end;

begin {main}
    repeat
      clrscr;
      ks;
      for r2:=1 to 50 do
```

```
    begin
      send($40);
      error:=true;
      for try:=1 to 8000 do
         if (port[$3fd] and $01)=$01 then
                  begin
                       r0:=port[$3f8];
                       if r0=$41 then
                          error:=false;
                       try:=8000;
                  end
         else try:=try+1;
      if error then r3:=0;
      r3:=r3+1;
    end;
   if r3>20 then begin
                   send($30);
                   receive(r0);
                   error:=false;break:=false;
                   hex:=[' ','0'..'9','a'..'f','A'..'F'];
                   ten:=[' ','0'..'9'];
                   debug;
                   r7:=0;
                  end
   else begin
             writeln('Communication Faile.');
             write('Abort or Retry? ');
             ch:=readkey;writeln(ch);
             if (ch='r') or (ch='R') then r7:=1
                                    else r7:=0;
        end;
  until r7=0;
  clrscr;
  send($f);
  writeln('goodbye.');
end.
```

“电脑坦克”程序清单

```
                                ORG 0000H
0000  020050          LJMP   MAIN
                                ORG 000BH
000B  0201D0          LJMP TIME0
                                ORG 001BH
001B  020150          LJMP TIME1
                                ORG 0050H
0050  75900F    MAIN: MOV  P1,#0FH
0053  758130          MOV SP,#30H
0056  C200            CLR 00
0058  C201            CLR 01
005A  C202            CLR 02
005C  120200          LCALL MUSIC
005F  758911          MOV TMOD,#11H
0062  758DFF          MOV TH1,#0FFH
0065  758BC0          MOV TL1,#0C0H
0068  758C9F          MOV TH0,#09FH
006B  758A00          MOV TL0,#0H
006E  D2AF            SETB EA
0070  D2AB            SETB ET1
0072  D2A9            SETB ET0
0074  D28E            SETB TR1
0076  D28C            SETB TR0
0078  12007F    HERE: LCALL Perform
007B  80FB            SJMP HERE
007D  00              NOP
007E  00              NOP
007F  D2D3    Perform:SETB RS0
0081  120103          LCALL DELAY
0084  7FA4            MOV R7,#0A4H
0086  1200AC          LCALL Forward
0089  7F10            MOV R7,#10H
008B  1200C4          LCALL LTurn
008E  7FA4            MOV  R7,#0A4H
0090  1200AC          LCALL Forward
0093  7F10            MOV R7,#10H
0095  1200D9          LCALL RTurn
0098  7F64            MOV R7,#64H
009A  1200EE          LCALL Backward
009D  7F10            MOV R7,#10H
009F  1200D9          LCALL RTurn
00A2  7FA4            MOV R7,#0A4H
00A4  1200AC          LCALL Forward
00A7  C2D3            CLR RS0
00A9  22              RET
```

```
00AA  00              NOP
00AB  00              NOP
00AC  43900F    Forward:ORL P1,#0FH
00AF  C290            CLR  P1.0
00B1  C292            CLR  P1.2
00B3  120200          LCALL MUSIC
00B6  3000FD    WAIT0: JNB 00,WAIT0
00B9  C200            CLR 00
00BB  43900F          ORL P1,#0FH
00BE  120103          LCALL DELAY
00C1  22              RET
00C2  00              NOP
00C3  00              NOP
00C4  43900F    LTurn: ORL P1,#0FH
00C7  C291            CLR  P1.1
00C9  C292            CLR  P1.2
00CB  3000FD    WAIT1: JNB 00,WAIT1
00CE  C200            CLR 00
00D0  43900F          ORL P1,#0FH
00D3  120103          LCALL DELAY
00D6  22              RET
00D7  00              NOP
00D8  00              NOP
00D9  43900F    RTurn: ORL  P1,#0FH
00DC  C290            CLR  P1.0
00DE  C293            CLR  P1.3
00E0  3000FD    WAIT2: JNB 00,WAIT2
00E3  C200            CLR 00
00E5  43900F          ORL P1,#0FH
00E8  120103          LCALL DELAY
00EB  22              RET
00EC  00              NOP
00ED  00              NOP
00EE  43900F    Backward: ORL  P1,#0FH
00F1  C291            CLR  P1.1
00F3  C293            CLR  P1.3
00F5  3000FD    WAIT3: JNB 00,WAIT3
00F8  C200            CLR 00
00FA  43900F          ORL P1,#0FH
00FD  120103          LCALL DELAY
0100  22              RET
0101  00              NOP
0102  00              NOP
0103  D202      DELAY: SETB 02
0105  7E00            MOV R6,#00H
```

```
0107  BE1AFD            CJNE R6,#1AH,S
010A  7E00              MOV R6,#0H
010C  22                RET
010D  00                NOP
010E  00                NOP
                                ORG 0150H
0150  300109      TIME1: JNB 01,GUN
0153  B297               CPL P1.7
0155  8F8D               MOV TH1,R7
0157  8E8B               MOV TL1,R6
0159  D2AB               SETB ET1
015B  32                 RETI
015C  758DFF      GUN:   MOV TH1,#0FFH
015F  758BC0             MOV TL1,#0C0H
0162  300207             JNB 02,OVER
0165  DA05               DJNZ R2,OVER
0167  0C                 INC R4
0168  EC                 MOV A,R4
0169  FA                 MOV R2,A
016A  B297               CPL P1.7
016C  00          OVER:  NOP
016D  D2AB               SETB ET1
016F  32                 RETI
0170  00                 NOP
0171  00                 NOP
                                ORG 01D0H
01D0  758C9F      TIME0: MOV TH0,#09FH
01D3  758A00             MOV TL0,#0H
01D6  85D0F0             MOV B,PSW
01D9  C2D4               CLR RS1
01DB  0E                 INC R6
01DC  1F                 DEC R7
01DD  BF0002             CJNE R7,#0H,TCONT
01E0  D200               SETB 00
01E2  30020C      TCONT: JNB 02,CONT0        ;control gun's sound
01E5  7A02               MOV R2,#02H
01E7  7C01               MOV R4,#01H
01E9  0B                 INC R3
01EA  BB1004             CJNE R3,#10H,CONT0
01ED  7B00               MOV R3,#0H
01EF  C202               CLR 02
01F1  85F0D0      CONT0: MOV PSW,B
01F4  D2A9               SETB ET0
01F6  32                 RETI
```

```
                                ORG 0200H
0200  D201      MUSIC: SETB 01
0202  D2D4             SETB RS1
0204  0D               INC R5
0205  9002CC     START: MOV DPTR.#TAB
0208  ED               MOV A,R5
0209  14               DEC A
020A  23               RL A
020B  93               MOVC A,@A+DPTR
020C  C0E0             PUSH A
020E  ED               MOV A,R5
020F  14               DEC A
0210  23               RL A
0211  04               INC A
0212  93               MOVC A,@A+DPTR
0213  C0E0             PUSH A
0215  D082             POP DPL
0217  D083             POP DPH
0219  C297           SONG:CLR P1.7
021B  7400             MOV A,#00H
021D  93               MOVC A.@A+DPTR
021E  FA               MOV R2,A
021F  C4               SWAP A
0220  540F             ANL A,#0FH
0222  6045             JZ MMND
0224  75F005           MOV B,#05H
0227  84               DIV AB
0228  C5F0             XCH A,B
022A  7004             JNZ MOM
022C  7405             MOV A,#05H
022E  15F0             DEC B
0230  F8             MOM:MOV R0,A      ;YU SHU
0231  A9F0             MOV R1,B      :SAN
0233  EA               MOV A,R2
0234  540F             ANL A,#0FH
0236  6005             JZ  YANS
0238  FA               MOV R2,A
0239  740C             MOV A,#0CH
023B  A4               MUL AB
023C  2A               ADD A,R2
023D  C083           YANS:PUSH DPH
023F  C082             PUSH DPL
0241  6014             JZ TM
0243  D297             SETB P1.7
0245  14               DEC A
```

```
0246  23              RL A
0247  90027F          MOV DPTR,#FREQ
024A  F9              MOV R1,A
024B  93              MOVC A,@A+DPTR
024C  FF              MOV R7,A
024D  E9              MOV A,R1
024E  04              INC A
024F  93              MOVC A,@A+DPTR
0250  FE              MOV R6,A
0251  8F8D            MOV TH1,R7
0253  8E8B            MOV TL1,R6
0255  D28E            SETB TR1
0257  9002C7        TM:MOV DPTR,#LONG
025A  E8              MOV A,R0
025B  14              DEC A
025C  93              MOVC A,@A+DPTR
025D  F8              MOV R0,A
025E  5175          TMM:ACALL MDELAY
0260  D8FC            DJNZ R0,TMM
0262  D082            POP DPL
0264  D083            POP DPH
0266  A3              INC DPTR
0267  4119            AJMP SONG
0269  C297          MMND:CLR P1.7
026B  BD0602            CJNE R5,#06H,RETURN
026E  7D00              MOV R5,#0H
0270  C2D4        RETURN: CLR RS1
0272  C201              CLR 01
0274  22                RET
0275  7C40        MDELAY:MOV R4,#40H
0277  7B21         LOOP1:MOV R3,#21H
0279  00           LOOP2:NOP
027A  DBFD              DJNZ R3,LOOP2
027C  DCF9              DJNZ R4,LOOP1
027E  22                RET
```

```
FREQ:DB 0F8H,8DH,0F8H,0F8H,0F9H,5DH,0F9H,0BCH,0FAH,16H,0FAH,6BH
     DB 0FAH,0BCH,0FBH,07H,0FBH,4FH,0FBH,92H,0FBH,0D2H,0FCH,0EH
     DB 0FCH,46H,0FCH,7CH,0FCH,0AEH,0FCH,0DEH,0FDH,0BH,0FDH,35H
     DB 0FDH,5EH,0FDH,83H,0FDH,0A7H,0FDH,0C9H,0FDH,0E9H,0FEH,07H
     DB 0FEH,23H,0FEH,3EH,0FEH,57H,0FEH,6FH,0FEH,85H,0FEH,9AH
     DB 0FEH,0AFH,0FEH,0C1H,0FEH,0D3H,0FEH,0E4H,0FEH,0F4H,0FFH,03
LONG:DB 08H,10H,18H,20H,40H
 TAB:DB 04H,00H,04H,50H,04H,9DH,04H,0E4H,05H,2CH,00H,00H,00H,00H
     DB 00H,00H,00H,C0H,00H,00H,00H,00H,00H,00H
     ORG 0400H
  T1:DB 88H,70H,6AH,88H,70H,6AH,8CH,70H,0B3H,0D5H,70H,0B3H,0D8H
     DB 70H,0B7H,0DAH,70H,0B8H,0D7H,70H,0BAH,0D8H,70H,0B5H,0D3H
     DB 70H,0B3H,0D5H,70H,0B3H,0D8H,70H,0B5H,0D3H,70H,6CH,0AAH
     DB 88H,70H,6AH,88H,70H,6AH,8CH,70H,0B3H,0D5H,70H,0B3H,0D5H
     DB 70H,0B7H,0DAH,70H,0B8H,0D7H,70H,0BAH,0D8H,70H,0B5H,0D3H
     DB 70H,0B3H,0D8H,70H,68H,9AH,0E3H,0A8H,00H
     ORG 0450H
  T2:DB 0C4H,0C3H,0E3H,0C4H,0C3H,0E3H,0C4H,0C3H,0E3H,0EBH,0E0H
     DB 0CBH,0CAH,0E8H,0C8H,0C6H,0E4H,0C4H,0C3H,0E1H,0E1H,0E0H
     DB 0C3H,0C1H,0E1H,0C3H,0C1H,0E1H,0C3H,0C1H,0E1H,0EAH,0E0H
     DB 0CAH,0C8H,0E7H,0C7H,0C4H,0E3H,0C3H,0C1H,9BH,9BH,0E0H
     DB 0CBH,0CAH,0EAH,0E7H,0EAH,0E8H,0E3H,0E0H,0CBH,0CAH,0EAH
     DB 0E7H,0EAH,0E8H,0EBH,0CAH,0C8H,0C6H,0C4H,0E3H,97H,98H,9AH
     DB 9BH,0C1H,7BH,9AH,98H,0E3H,40H,50H,50H,00H
     ORG 0490H
  T3:DB 0B1H,0B6H,60H,0B1H,0B6H,60H,0B1H,0D6H,0C0H,0B1H
     DB 0B6H,0B8H,0B6H,0C5H,0B6H,0B8H,0E0H,0B1H,0B5H,0B0H
     DB 0B1H,0B5H,0B0H,0B1H,0D5H,0C0H,0B1H,0B5H,0B6H,0B5H
     DB 0C3H,0B5H,0B6H,0D0H,0BCH,0BAH,0B9H,0BAH,0B6H,0B5H
     DB 0B6H,0B1H,0C0H,6AH,6BH,0B1H,0B3H,0B5H,0B6H,0B8H
     DB 0BAH,0BBH,0B8H,0C0H,0BBH,0BAH,0BBH,0B8H,0B7H,0B8H
     DB 0B5H,0B4H,0B5H,0B1H,0C0H,0BCH,0BBH,0BAH,0B8H,0B6H,00H
     ORG 04E4H
  T4:DB 63H,60H,63H,60H,63H,60H,68H,90H,6CH
     DB 63H,60H,63H,60H,63H,60H,68H,90H,6CH
     DB 13H,10H,13H,10H,18H,30H,68H,10H,68H,10H
     DB 67H,60H,67H,60H,65H,60H,65H,60H,0A3H,20H
     DB 63H,60H,63H,60H,63H,60H,67H,90H,6AH,80H
     DB 63H,60H,63H,60H,63H,60H,67H,90H,6AH,90H
     DB 1AH,10H,1AH,10H,13H,30H,0C3H,0C5H,0C3H,0C1H
```

```
   ORG 052CH
T5:DB 73H,77H,7AH,7AH,70H,0BAH,10H,0BAH,30H,0B7H
   DB 10H,0B7H,30H,73H,73H,77H,7AH,7AH,20H,0BAH
   DB 10H,0BAH,30H,0B8H,10H,0B8H,30H,72H,72H,75H
   DB 7CH,7CH,20H,0BCH,10H,0BCH,30H,0B8H,10H,0B8H
   DB 30H,72H,72H,75H,7CH,7CH,20H,0BCH,10H,0BCH
   DB 30H,0B7H,10H,0B7H,30H,73H,73H,77H,7AH,0C3H
   DB 20H,10H,30H,0BAH,10H,0BAH,30H,73H,73H,77H
   DB 7AH,0C3H,20H,10H,30H,0BCH,10H,0BCH,30H,75H
   DB 75H,78H,7CH,10H,0ACH,10H,79H,7AH,0F7H,0C3H
   DB 77H,97H,75H,9CH,7AH,73H,10H,63H,73H,40H
   DB 6AH,10H,68H,30H,6AH,10H,68H,30H,7AH,0F7H
   DB 0C5H,6AH,10H,67H,30H,6AH,10H,67H,30H,7AH
   DB 0F5H,0C3H,6AH,10H,68H,30H,6AH,10H,68H,30H
   DB 7AH,0F7H,0C5H,7AH,0C3H,0C5H,0C7H,0EAH,0C8H,0B7H
   DB 0B7H,0C7H,0B5H,00H
```

```
INCLUDE IN HASHTAB:
    MAIN:    50
   TIME0:    1D0
   TIME1:    150
   MUSIC:    200
    HERE:    78
 PERFORM:    7F
   DELAY:    103
 FORWARD:    AC
   LTURN:    C4
   RTURN:    D9
BACKWARD:    EE
   WAIT0:    B6
   WAIT1:    CB
   WAIT2:    E0
   WAIT3:    F5
     GUN:    15C
    OVER:    16C
   TCONT:    1E2
   CONT0:    1F1
```

```
;Blue Da
;    by
;J.S.Str
```

```
  START:    205
    TAB:    2CC
   SONG:    219
   MMND:    269
    MOM:    230
   YANS:    23D
     TM:    257
   FREQ:    27F
   LONG:    2C7
    TMM:    25E
 MDELAY:    275
 RETURN:    270
  LOOP1:    277
  LOOP2:    279
     T1:    400
     T2:    450
     T3:    49D
     T4:    4E4
     T5:    52C
```

MSX-EPROM 读写/擦除器

```
 10:    D400            ORG 0D400H
 20:                    ;-----------------------------
 30:                    ;¦This program is write EPROM¦
 40:                    ;¦                           ¦
 50:                    ;¦EPROM ADDRES is BC         ¦
 60:                    ;¦DATA  START  is DE         ¦
 70:                    ;¦DATA  END    is HL         ¦
 80:                    ;¦___________________________¦
 90:    D400 23         INC HL
100:    D401 EB         EX DE,HL
110:    D402 3D         DEC A
120:    D403 2805       JR Z,WRITE
130:    D405 3D         DEC A
140:    D406 2816       JR Z,READ
150:    D408 1825       JR CPM
160:                    ;---- WRITE ----
170:    D40A            WRITE:
180:    D40A 3E80       LD A,80H
190:    D40C D303       OUT (3),A
200:    D40E CD57D4     WR:CALL ADDRES
210:    D411 7E         LD A,(HL)
220:    D412 D300       OUT (0),A
230:    D414 CD91D4     CALL VPP
240:    D417 CDACD4     CALL LDINC
250:    D41A 20F2       JR NZ,WR
260:    D41C 1829       JR LAST
270:                    ;---- READ ----
280:    D41E 3E90       READ:LD A,90H
290:    D420 D303       OUT (3),A
300:    D422 CD57D4     RD:CALL ADDRES
310:    D425 DB00       IN A,(0)
320:    D427 77         LD (HL),A
330:    D428 CDACD4     CALL LDINC
340:    D42B 20F5       JR NZ,RD
350:    D42D 1818       JR LAST
360:                    ;---- CP A,(HL) ----
370:    D42F 3E90       CPM:LD A,90H
380:    D431 D303       OUT (3),A
390:    D433 CD57D4     CM:CALL ADDRES
400:    D436 DB00       IN A,(0)
410:    D438 BE         CP (HL)
420:    D439 C460D4     CALL NZ,ERROR
430:    D43C CD6F04     CALL 46FH
```

```
440:    D43F DA12ED    JP C,0ED12H
450:    D442 CDACD4    CALL LDINC
460:    D445 20EC      JR NZ,CM
471:    D447           LAST:;-------
472:    D447 CD74F1    CALL RETURN
473:    D44A 2B        DEC HL
474:    D44B CD86F1    CALL BXASH2
475:    D44E 21B1D4    LD HL,BUF
476:    D451 CD7DF1    CALL PRINT
477:    D454 C312ED    JP 0ED12H
480:    D457           ADDRES:;--------
490:    D457 79        LD A,C
500:    D458 D301      OUT (1),A
510:    D45A 78        LD A,B
520:    D45B E67F      AND 7FH
530:    D45D D302      OUT (2),A
540:    D45F C9        RET
550:    D460           ERROR:;---------
560:    D460 E5        PUSH HL
565:    D461 F5        PUSH AF
570:    D462 CD74F1    CALL RETURN
590:    D465 C5        PUSH BC
600:    D466 E1        POP HL
610:    D467 CD86F1    CALL BXASH2
620:    D46A CD6AF1    CALL SPACS
630:    D46D F1        POP AF
640:    D46E CD8BF1    CALL BXASH1
650:    D471 CD6AF1    CALL SPACS
660:    D474 E1        POP HL
670:    D475 7E        LD A,(HL)
680:    D476 CD8BF1    CALL BXASH1
690:    D479 CD6AF1    CALL SPACS
700:    D47C CD86F1    CALL BXASH2
710:    D47F CD9C00    CALL 9CH
720:    D482 C8        RET Z
730:    D483 E5        PUSH HL
740:    D484 CD5601    CALL 156H
750:    D487 CD9C00    LOOP51:CALL 9CH
760:    D48A 28FB      JR Z,LOOP51
770:    D48C CD5601    CALL 156H
780:    D48F E1        POP HL
790:    D490 C9        RET
870:    F17D =         PRINT EQU 0F17DH
880:    F186 =         BXASH2 EQU 0F186H
890:    F18B =         BXASH1 EQU 0F18BH
900:    F16A =         SPACS EQU 0F16AH
910:    F174 =         RETURN EQU 0F174H
920:    D491           VPP:;---------
```

```
 925:    D491 F3         DI
 930:    D492 F5         PUSH AF
 940:    D493 DB02       IN A,(2)
 950:    D495 F680       OR 80H
 960:    D497 D302       OUT (2),A
 970:    D499 C5         PUSH BC
 980:    D49A 010002     LD BC,TIME
 990:    D49D 0B         WAIT:DEC BC
1000:    D49E 78         LD A,B
1010:    D49F B1         OR C
1020:    D4A0 20FB       JR NZ,WAIT
1030:    D4A2 C1         POP BC
1040:    D4A3 DB02       IN A,(2)
1050:    D4A5 E67F       AND 7FH
1060:    D4A7 D302       OUT (2),A
1070:    D4A9 F1         POP AF
1075:    D4AA FB         EI
1080:    D4AB C9         RET
1085:    D4AC            LDINC:;---------
1090:    D4AC 03         INC BC
1100:    D4AD 23         INC HL
1110:    D4AE C32000     JP 20H
1120:    D4B1 20454E44   BUF:DEFM ' END ADDRES'
1130:    D4BC            DEFS 1
1140:    0200 =          TIME EQU 200H
```

```
0457  ADDRES    D4B1  BUF      F18B  BXASH1    F186  BXASH2
0433  CM        D42F  CPM      D460  ERROR     D447  LAST
04AC  LDINC     D487  LOOP51   F17D  PRINT     D422  RD
041E  READ      F174  RETURN   F16A  SPACS     0200  TIME
0491  VPP       D49D  WAIT     D40E  WR        D40A  WRITE
```

给 TANGO 软件增加汉字元件库程序清单

```
10 OPTION BASE 1: DIM A%(16, 16), X(5), Y(5), Z(1066)
20 H=0:D=1:c1=0101
31 INPUT "Filename:"; F$
32 OPEN "r", #1, F$, 1: FIELD #1, 1 AS A$
40 OPEN "cclib"  AS #2 LEN = 32: FIELD #2, 32 AS WOR$
45 QM% = C1 \ 100: WM% = C1 - QM% * 100:H=0
50 GOSUB 60
52 C2 = C1  \ 1000: C3 = (C1 - C2 * 1000) \ 100
53 C4 = (C1 - C2 * 1000 - C3 * 100) \ 10
54 C5 = (C1 - C2 * 1000 - C3 * 100 - C4 * 10)
55 GOTO 150
60 REM
70 IF QM% >= 16 THEN 90
80 GOTO 110
90 RE% = (QM% - 9) * 94 + WM%: GET #2, RE%: WO$ = WOR$
100 GOTO 120
110 RE% = (QM% - 1) * 94 + WM%: GET #2, RE%: WO$ = WOR$
120 FOR K% = 32 TO 1 STEP -1: B% = ASC(MID$(WO$, K%, 1)):
    I% = (K% - 1) \ 2 + 1
130 FOR L% = 8 TO 1 STEP -1: J% = (((K% - 1) * 8 + L% -
    1) MOD 16) + 1
140 A%(I%, J%) = B% MOD 2: B% = B% \ 2: NEXT L%: NEXT K%:
    RETURN
150 LSET A$ = CHR$(39)
160 PUT #1, D: D = D + 1
170 LSET A$ = CHR$(C2 + 48)
180 PUT #1, D: D = D + 1
190 LSET A$ = CHR$(C3 + 48)
200 PUT #1, D: D = D + 1
201 LSET A$ = CHR$(C4 + 48)
202 PUT #1, D: D = D + 1
203 LSET A$ = CHR$(C5 + 48)
204 PUT #1, D: D = D + 1
210 LSET A$ = CHR$(39)
220 PUT #1, D: D = D + 1
230 GOSUB 1270
320 LSET A$ = CHR$(50)
330 PUT #1, D: D = D + 1
340 LSET A$ = CHR$(32)
350 PUT #1, D: D = D + 1
360 GOSUB 1440
370 GOSUB 1750
380 FOR I% = 1 TO 16: FOR J% = 1 TO 16
390 IF A%(I%, J%) <>0 THEN 480
400 LSET A$ = CHR$(46)
```

```
410 PUT #1, D: D = D + 1
420 NEXT J%
430 FOR K = 1 TO 5
440 LSET A$ = CHR$(46)
450 PUT #1, D: D = D + 1
460 NEXT K
470 GOTO 570
480 LSET A$ = CHR$(35)
490 PUT #1, D: D = D + 1
500 GOTO 420
570 LSET A$ = CHR$(13)
580 PUT #1, D: D = D + 1
590 LSET A$ = CHR$(10)
600 PUT #1, D: D = D + 1
610 H = H + 1
620 IF H = 1 THEN GOSUB 1810: GOTO 780
630 IF H = 2 THEN GOSUB 1850: GOTO 780
640 IF H = 3 THEN GOSUB 1890: GOTO 780
650 IF H = 4 THEN GOSUB 1930: GOTO 780
660 IF H = 5 THEN GOSUB 1970: GOTO 780
```

```
670 IF H = 6 THEN GOSUB 2010: GOTO 780
680 IF H = 7 THEN GOSUB 2050: GOTO 780
690 IF H = 8 THEN GOSUB 2090: GOTO 780
700 IF H = 9 THEN GOSUB 2130: GOTO 780
710 IF H = 10 THEN GOSUB 2170: GOTO 780
720 IF H = 11 THEN GOSUB 2190: GOTO 780
730 IF H = 12 THEN GOSUB 2210: GOTO 780
740 IF H = 13 THEN GOSUB 2230: GOTO 780
750 IF H = 14 THEN GOSUB 2250: GOTO 780
760 IF H = 15 THEN GOSUB 2270: GOTO 780
770 IF H = 16 THEN GOSUB 2290: GOTO 780
780 NEXT I%
781 GOSUB 810
790 GOSUB 2310
800 GOSUB 810
805 GOTO 890
810 FOR J = 1 TO 21
820 LSET A$ = CHR$(46)
830 PUT #1, D: D = D + 1
940 NEXT J
850 LSET A$ = CHR$(13)
860 PUT #1, D: D = D + 1
870 LSET A$ = CHR$(10).
880 PUT #1, D: D = D + 1:RETURN
```

```
890 GOSUB 2420
900 GOSUB 810
980 GOSUB 2530
990 GOSUB 810
1070 GOSUB 2640
1080 GOSUB 810
1160 LSET A$ = CHR$(13)
1170 PUT #1, D: D = D + 1
1180 LSET A$ = CHR$(10)
1190 PUT #1, D: D = D + 1
1200 C1=C1+1
1201 IF C1=795 THEN C1=1601
1202 C8=C1-(C1\100)*100
1203 IF C8>94 THEN C1=C1+6
1204 IF C1>8794 THEN CLOSE:CL
     :SYSTEM
1205 GOTO 45
1270 LSET A$ = CHR$(32)
1280 PUT #1, D: D = D + 1
1290 LSET A$ = CHR$(80)
1300 PUT #1, D: D = D + 1
1310 LSET A$ = CHR$(79)
1320 PUT #1, D: D = D + 1
1330 LSET A$ = CHR$(87)
1340 PUT #1, D: D = D + 1
1350 LSET A$ = CHR$(69)
1360 PUT #1, D: D = D + 1
1370 LSET A$ = CHR$(82)
1380 PUT #1, D: D = D + 1
1390 LSET A$ = CHR$(13)
1400 PUT #1, D: D = D + 1
1410 LSET A$ = CHR$(10)
1420 PUT #1, D: D = D + 1
1430 RETURN
1440 LSET A$ = CHR$(50)
1450 PUT #1, D: D = D + 1
1460 LSET A$ = CHR$(32)
1470 PUT #1, D: D = D + 1
1480 LSET A$ = CHR$(49)

1490 PUT #1, D: D = D + 1
1500 FOR I = 1 TO 11
1510 LSET A$ = CHR$(32)
1520 PUT #1, D: D = D + 1
1530 NEXT I
1540 LSET A$ = CHR$(13)
1550 PUT #1, D: D = D + 1
1560 LSET A$ = CHR$(10)
1570 PUT #1, D: D = D + 1
1580 LSET A$ = CHR$(66)
1590 PUT #1, D: D = D + 1
1600 LSET A$ = CHR$(73)
1610 PUT #1, D: D = D + 1
1620 LSET A$ = CHR$(84)
1630 PUT #1, D: D = D + 1
1640 LSET A$ = CHR$(77)
1650 PUT #1, D: D = D + 1
1660 LSET A$ = CHR$(65)
1670 PUT #1, D: D = D + 1
1680 LSET A$ = CHR$(80)
1690 PUT #1, D: D = D + 1
1700 LSET A$ = CHR$(13)
1710 PUT #1, D: D = D + 1
1720 LSET A$ = CHR$(10)
1730 PUT #1, D: D = D + 1
1740 RETURN
1750 GOSUB 2750
1760 LSET A$ = CHR$(48)
1770 PUT #1, D: D = D + 1
1780 LSET A$ = CHR$(125)
1790 PUT #1, D: D = D + 1
1800 RETURN
1810 GOSUB 2750
1820 LSET A$ = CHR$(49)
1830 PUT #1, D: D = D + 1
1840 GOTO 1780
1850 GOSUB 2750
1860 LSET A$ = CHR$(50)
1870 PUT #1, D: D = D + 1
1880 GOTO 1780
1890 GOSUB 2750
1900 LSET A$ = CHR$(51)
1910 PUT #1, D: D = D + 1
1920 GOTO 1780
1930 GOSUB 2750
1940 LSET A$ = CHR$(52)
1950 PUT #1, D: D = D + 1
1960 GOTO 1780
1970 GOSUB 2750
1980 LSET A$ = CHR$(53)
1990 PUT #1, D: D = D + 1
2000 GOTO 1780
2010 GOSUB 2750
```

```
2020 LSET A$ = CHR$(54)
2030 PUT #1, D: D = D + 1
2040 GOTO 1780
2050 GOSUB 2750
2060 LSET A$ = CHR$(55)
2070 PUT #1, D: D = D + 1
2080 GOTO 1780
2090 GOSUB 2750
2100 LSET A$ = CHR$(56)
2110 PUT #1, D: D = D + 1
2120 GOTO 1780
2130 GOSUB 2750
2140 LSET A$ = CHR$(57)
2150 PUT #1, D: D = D + 1
2160 GOTO 1780
2170 GOSUB 2810
2180 GOTO 1760
2190 GOSUB 2810
2200 GOTO 1820
2210 GOSUB 2810
2220 GOTO 1860
2230 GOSUB 2810
2240 GOTO 1900
2250 GOSUB 2810
2260 GOTO 1940
2270 GOSUB 2810
2280 GOTO 1980
2290 GOSUB 2810
2300 GOTO 2020
2310 LSET A$ = CHR$(123)
2320 PUT #1, D: D = D + 1
2330 LSET A$ = CHR$(32)
2340 PUT #1, D: D = D + 1
2350 LSET A$ = CHR$(49)
2360 PUT #1, D: D = D + 1
2370 LSET A$ = CHR$(55)
2380 PUT #1, D: D = D + 1
2390 LSET A$ = CHR$(125)
2400 PUT #1, D: D = D + 1
2410 RETURN
2420 LSET A$ = CHR$(123)
2430 PUT #1, D: D = D + 1
2440 LSET A$ = CHR$(32)
2450 PUT #1, D: D = D + 1
2460 LSET A$ = CHR$(49)
2470 PUT #1, D: D = D + 1
2480 LSET A$ = CHR$(56)
2490 PUT #1, D: D = D + 1
2500 LSET A$ = CHR$(125)
2510 PUT #1, D: D = D + 1
2520 RETURN
2530 LSET A$ = CHR$(123)
2540 PUT #1, D: D = D + 1
2550 LSET A$ = CHR$(32)
2560 PUT #1, D: D = D + 1
2570 LSET A$ = CHR$(49)
2580 PUT #1, D: D = D + 1
2590 LSET A$ = CHR$(57)
2600 PUT #1, D: D = D + 1
2610 LSET A$ = CHR$(125)
2620 PUT #1, D: D = D + 1
2630 RETURN
2640 LSET A$ = CHR$(123)
2650 PUT #1, D: D = D + 1
2660 LSET A$ = CHR$(32)
2670 PUT #1, D: D = D + 1
2680 LSET A$ = CHR$(50)
2690 PUT #1, D: D = D + 1
2700 LSET A$ = CHR$(48)
2710 PUT #1, D: D = D + 1
2720 LSET A$ = CHR$(125)
2730 PUT #1, D: D = D + 1
2740 RETURN
2750 LSET A$ = CHR$(123)
2760 PUT #1, D: D = D + 1
2770 LSET A$ = CHR$(32)
2780 PUT #1, D: D = D + 1
2790 PUT #1, D: D = D + 1
2800 RETURN
2810 LSET A$ = CHR$(123)
2820 PUT #1, D: D = D + 1
2830 LSET A$ = CHR$(32)
2840 PUT #1, D: D = D + 1
2850 LSET A$ = CHR$(49)
2860 PUT #1, D: D = D + 1
2870 RETURN
```