

電子與電腦

一九九二年合訂本

●《电子与电脑》编辑部编



電子工業出版社

●《电子与电脑》一九九二年合订本

●计算机爱好者之友

●欢迎订阅 欢迎投稿



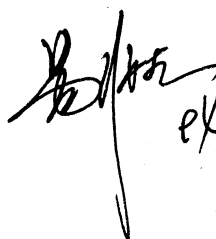
ISBN7-5053-1935-3/TP·468

定价:16.50元

电子与电脑

1992年合订本

《电子与电脑》编辑部编


PK-8.00.

电子工业出版社

(京)新登字 055 号

内 容 提 要

《电子与电脑》(月刊)以初、中级水平的电脑应用人员为读者,突出普及性、启发性和实用性。主要栏目有:综述、PC 用户、学习机之友、语言讲座、初级程序员级水平考试辅导、学用单片机、学装微电脑、电脑巧开发、电脑游戏机、新书与软件介绍、读者联谊等。本台订本补充了部份因月刊篇幅所限而未登出的程序清单。

《电子与电脑》1992 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:38 字数:1365 千字

1993 年 2 月第 1 版 1993 年 2 月第 1 次印刷

印数:20100 册 定价:16.50 元

ISBN7-5053-1935-3/TP·468

《电子与电脑》1992年总目录

·综述·

	期	页
猴年进步——本刊92年致读者	1	2
我国单片机应用技术发展趋势及展望	何立民	1 2
我国单片机应用技术发展趋势及展望(续)	何立民	2 2
软件汉化原理	郑茂松	3 2
引人注目的微型机	戴伟高	4 2
91年度 COMDEX 电脑博览会	吕问黎	5 2
美国当今图像处理软件	吕问黎	6 2
优越的专家系统语言 Prolog	王粤宁	7 2
计算机信息的安全保护	李群	8 2
量子芯片与人工智能计算机	姚立新	8 4
计算机安全性的几种实现方法	邱向群	9 2
改革开放结硕果	齐心	10 2
软件市场大有希望	温有良	10 2
新兴的教育技术—CAI	陈健	10 3
图象处理的一大革命——压缩		11 2
美国消费电子业	吕问黎	12 2
应用计算机网络辅助高中物理教学	肖美 王天谔 李燕萍	12 3

·PC 用户·

自动调平四舍五入误差简易快速软件	李燕妹	1 4
新颖的用户界面一下拉式菜单的实现	李晓峰	1 6
再谈硬盘无效时回收文件的方法	岳千钧	1 9
IBM 计算机几个重要硬件设备的软锁程序	温盛文	1 10
一个实用月历程序	沈玉江	1 11
用拉幕方式显示信息	刘善平	1 11
拉图游戏	刘晓峰	1 13
通用文件病毒防御系统	姜金友	1 14
彩色汉字通用下拉式窗口菜单的设计与实现	王宏	2 4
dBASE III 或 Foxbase 菜单式自由组合查询	毛维平	2 9
单显 PC 机的应用技巧	袁津生	2 10
BASIC 程序用 P 存盘的解密方法	鞠协贤	2 11
谈谈编译 BASIC 在长城机上内存不够的问题	卢祥江	2 11
也谈解决运行 FoxBASE 内存不够的问题	周立宇	2 12

编制能灵活打印二维报表的“报表生成系统”

..... 苏士俊	3 3	
在管理信息系统中实现计算模型的规则库方法	孙永芳 吴泉源	3 6
dBASE III 修改文件属性的方法	许再由	3 8
抗病毒软件		
——Turbl Anti-Virus V6.80A	唐银红	3 8
新颖的日历打印通用程序	何贤敏	3 8
用24针打印机打印 CAD 图形	卢耀志	4 4
编制菜单程序的辅助程序@.COM	张红	4 7
格式文件与 SET FILTER TO 命令的结合	麦红波	4 8
Azusa/2708病毒与介质描述	何悦荣	4 9
MS FORTRAN 程序的菜单设计技术	曹国钧 王健	4 10
如何制做病毒标本	梁宇翀	4 10
用 dBASE III MEMO 字段开发试题库系统	陈卫平	4 11
关于《dBASE III 过程文件加密的发现》质疑	张维仁	4 11
在单台驱动器上如何作软盘备份	杨栋林	4 12
一种不打印数值零的方法	曹和平	4 12
交互式的 DEL 命令	莫琳	4 12
获取 FOX 文件中密码的两种简单方法	周日初	5 3
对全屏编辑命令 BROWSE 做一点补充	宋开胜	5 4
感染 OMICROX 病毒文件的消毒方法	刘东祥	5 5
COLor400卡屏幕彩色图形的硬拷贝	张辉 李东升	5 6
利用格式化功能使磁盘“加锁”	张李文	5 9
一种有效软件解密方法	张彦洪	5 12
三维图形显示的遮掩技术及实现	季军杰	5 13
BASIC 超长随机文件的处理方法	朱建平	5 15
dBASE III 和 FoxBASE+ 环境下实用多		
窗口弹出式菜单的实现方法	张志远	6 3
CCDOS2.1特殊显示功能的妙用	李林枫	6 4
在 DOS 中使用功能菜单的方法	金林樵	6 5
PKZIP 压缩软件包的技术背景及使用	夏伟文	6 6
如何使用《IDOSV1.0》支持 AT101/102键盘	杨旭	6 9
恢复硬盘启动功能的方法	王晓红	6 9
结构文件与 FIELD0 函数的比较	严桂兰 刘甲耀	7 4

24小时服务系统的人员安排	陈君佐	7	6	PC 机软件加解密技术剖析(续)	李文亮	12	18
ARC 系列压档工具软件的应用	戴青松	7	7				
DOS 3.30若干新增命令的功能与应用	柳见成	7	9	·学习机之友·			
WPS 的彻底解密	瞿新国	7	10	苹果机 BASIC 语言容错性浅析	那履弘	1	16
处理系统不认硬盘时应注意的问题	陈 栋	7	11	Apple 内存校对发声程序	连 勤	1	19
dBASE III 管理系统中程序多级调用的跟踪	周 为	8	5	ProDOS 磁盘操作系统入门(续)	廖 凯	1	21
一种在 M1724机上打印图形的实用方法	李凯里	8	7	虚拟 DOS 的改进及其在通讯网络中的应用	石永琳	2	13
物理扇区与逻辑扇区	崔来堂	8	8	磁盘数据的压缩存储技巧	陈庆祥	2	14
DOS 命令在 IBM-PC 机通信中的妙用	林 立 林 娜	8	9	为 Apple DOS 创造三条实用命令	王志超	2	17
硬盘加锁小程序	刘志存	8	10	ProDOS 磁盘操作系统入门(续)	廖 凯	2	18
实用程序三则	任绥海	8	11	CEC-I 汉字系统子程序的应用	傅叔平	3	11
快而短的排序程序	许野平	8	12	CATALOG 命令的改进	陈治浩	3	14
1724打印机驱动程序行距的修改	谭人杰	8	13	中华学习机查找汉字区位码	胡瑞辉	3	14
加密 WPS 文件的解密	谷高平 孟建伟	8	13	监控程序的浮动	赵 旭	3	15
电脑病毒入侵报警程序	陶秋刚	8	14	中华学习机特殊使用技巧	王 冈	3	15
文本方式下汉字菜单的实现	曾跃忠 蔡以群	9	3	有趣的对比	丁志伟	3	16
巧解黑色星期五病毒	谭小敏	9	5	中华学习机汉化引导程序	张福森	3	17
通用的汉化 WordStar	曹国钧	9	6	如何在苹果机持续运行一程序	张 毅	3	18
反函数查找法的实现	王杰民	9	7	ProDOS 磁盘操作系统入门(续)	廖 凯	3	21
在 CCDS2.13F 支持下 dBASE III 画图技术	徐国茂 黄松德 刘少明	9	8	汽车大赛	马宇昊	3	21
磁盘的簇与簇管理分析	崔来堂	10	4	为 APPLE DOS 增加多目录功能	秦燕军	4	14
程序运行过程中汉字/西文输入方式的自动切换	傅 雷	10	6	验证四色猜想的 BASIC 程序	陈庆祥	4	15
C 语言在配平化学方程式中的应用	王 晰	10	7	磁盘不抹掉文件的35轨改40轨方法	钱仕宏	4	16
DISK MANGER 使用技巧	梁高军	10	9	解拆 C-WORDSTAR 五笔字型	陈晓乐	4	16
如何利用 dBASE III 的 F1 功能键	涂振宇	10	10	BASIC 中“宏代换”的实现	许 斌	4	17
CCBIOS 2.13 稿纸方式打印	宋 捷	11	11	改进的 UNNEW 程序	张 亭	4	18
微机屏幕的打印和放大	谭人杰	11	11	CEC-I 中华学习机磁带游戏软件的解密	胡发新	4	18
1992年全国青少年信息学(计算机)竞赛试题	10	13	13	也谈 CEC-I 全功能造字	覃 敏	4	19
PC 机软件加解密技术剖析	李文亮	11	13	ProDOS 磁盘操作系统入门(续)	廖 凯	4	20
字符处理函数在 FORTRAN 中的实现	任铁良 丁玲玲	11	5	打印机的定位和格式控制	栾傲发	5	16
CPAV 防病毒软件使用方法	吴桦	11	7	放大打印 APPLE II 高分辨率图形又一法	刘善平	5	18
带检索功能的通讯录录入程序	张春明	11	10	趣味速算练习器	赵 旭	5	19
PC 机系统时间显示	郑嘉琦	11	12	计算机解决数学问题	赵方明	5	19
最近出现的几种新病毒	苏民生	11	13	巧改系统	张 浩	5	20
电子扭计板	葛建华 吴立国	11	14	数字游戏	闫 浩	5	20
第四届国际信息学奥林匹克竞赛试题	11	15	15	ProDOS 磁盘操作系统入门(续)	廖 凯	5	21
也谈 FOXBASE+弹出式菜单的实现方法	王瑞华	12	5	无须驱动器的电子打字机程序	刘庆丰	6	11
C 语言如何读取 dBASE 的库文件	李晓华	12	6	TOOL-KIT 使用详解	姜 宏	6	13
如何实现题库系统的图文并貌	裴伟东	12	7	利用 BOOT 程序对出租软件进行计次使用	任晓芳	6	17
一个简单实用的磁盘加密程序	任绥海	12	9	中华学习机汉字双页显示	蒋建一	6	18
2.13HZ 汉字系统安装虚拟盘字库新探	廖 凯	12	10	高速排序程序的使用技巧	张世栋	6	19
怎样从死循环程序返回 DOS	方 晨	12	10	汉字编码打印程序	李 铁	6	21
BASIC 实现“卡拉—OK”	于龙滋	12	10	对《CEC-I 键控光标作图程序》的改进	李庆岱	6	22
实现1.2MB软磁盘之间直接拷贝简法	何崇乐	12	12	CEC-I 彩色 TV 字幕	包 敢	7	12
兼容机怎样运行 BASICA.COM	邓文超	12	13	BASIC 程序中变量名使用情况的列表印出程序	蔡 伟	7	14
香港病毒的消除和防范	孟 桥	12	14	磁头清洗及其辅助程序	唐汉雄	7	15
谈数据文件“死而复生”的技巧	李瑛彬	12	15	求法雷数列的新方法	廖汉雄	7	15
数据自动存盘程序	汪海波	12	16	TOOL-KIT 使用详解续	姜 宏	7	17
				对几种高精度数值计算方法的改进	陈宏祥	8	15
				CEC-I 自造字键盘辅助点阵生成程序	汤永进	8	16

二维函数曲线拟合	李 涛	8	17	
在 Applesoft 系统下实现 COMMON 功能	安赵根	8	18	
通用打印课表程序	程建伟	8	19	
CP/M 系统文件的恢复方法	李 齐	8	20	
英语单词快速记忆系统	张振堂	9	13	
用 CHR \$ 函数压缩存盘数据	梁才柱	9	16	
PB-700 微机解密技巧	孙 力	9	17	
双人百米赛跑	汪 波	9	17	
寻找莱蒙托夫游戏规律	钱雁群	9	17	
也谈“一题多解”	郑明达	9	18	
考试的统计分析程序	胡筱罡	9	19	
氯化氢制取的动态显示	刘 萍	9	20	
磁盘加密一法	黄晓晖	9	21	
BASIC 子程序的递归调用	陈继良	丘 文	10	15
APPLE 机的快速排序	张 亨	10	16	
百年公历—农历互查	刘安军	19	17	
内存数据代码搜索程序	苏 华	10	20	
微机模拟游标卡尺读数训练程序	王太文	11	17	
磁带虚拟磁盘的文件管理	胡发新	11	18	
APPLE—II 音乐功能扩充	陈建明	11	19	
POSITION 命令新用	周 进	11	20	
CEC—I 与 1724 打印机配接图形硬拷贝程序	张益贵	11	20	
游戏接口 PDL 的扩展	张建群	11	21	
SUPER DOS 简介	张 志	11	21	
推荐一个扩展 BASIC 语句——拆字串语句	王 凯	12	20	
也谈内存信息的打印	张 亨	12	22	
增强中华学习机的音响功能	杨云霄	12	23	
计算机各类债券储蓄收益利息 BASIC 程序	陶文庆	12	24	
哥德巴赫猜想 BASIC 程序的改进	卢良红	12	24	
数组的动态删除	冯端品	12	25	

•6520 机器语言讲座•

6520 机器语言程序设计讲座	朱国江	1	23
第二章 微处理器、存储器、寄存器简介	朱国江	2	20
第三章 6520 MPU 的寻址方式	朱国江	3	22
第四章 6520 MPU 的指令系统	朱国江	4	22
第五章 源程序的编辑、汇编和运行	朱国江	5	24
第六章 简单程序和分支程序设计	朱国江	6	23
第七章 循环程序设计	朱国江	7	20
第八章 子程序设计	朱国江	8	21
第九章 堆栈程序设计	朱国江	9	21
第十章 监控子程序的调用	朱国江	10	21
第十一章 监控子程序的调用(下)	朱国江	11	23

•FORTH 语言讲座•

第一讲 概述	丁志伟	12	26
--------------	-----	----	----

•初级程序员级水平考试辅导讲座•

计算机硬件基础知识	顾育麒	1	27
-----------------	-----	---	----

计算机硬件基础知识(续)	顾育麒	2	24
第三章 数据结构	宋丹颖	3	26
操作系统及 PC-DOS	宋丹颖	4	27
数据库基础	阚 路	5	29
BASIC 基础	李志刚	6	29
PC BASIC 自测试题解答与分析	李志刚	7	25
一九九二年计算机初级软件人员竞赛试题		8	26
试题解答与分析		9	25

•学用单片机•

8031 单片机最小系统	罗明宽	李金相	1	31
低功耗单片机最小系统	张培仁	刘振安	2	29
新一代超高集成度 Z80 微处理器系列	董伯明		2	31
单片机最小应用系统与液晶显示器(LCD)的接口	张培仁		3	29
实现单片机最小系统与 PC 机的通信	张培仁		4	30
提高单片机最小系统抗干扰能力和自恢复方法	张培仁	刘振安	5	32
TP801 单板机与微型机数据双向并行传送一例	刘浔和		5	33
MCS—51 汇编指令外延的应用	陈亿善		5	34
带 A/D 的单片机最小应用系统	张培仁	刘振安	6	31
BJS—51 单片机实验系统	盛焕鸣		7	29
BJS—51 的监控程序	盛焕鸣		8	31
单片机实验与 BJS—51 实验教程	李广弟		9	30
BJS—51 单片机实验系统(续)	李广弟		10	25
CYSCB—ZMCS—518098 单片单板机硬件设计原理	吴 微		10	26
BJS—51 单片机实验系统(续)	张俊谟		11	26
模/数、数/模转换实验(续)	张俊谟		12	29

•学装微电脑•

附带应急开关的顺序控制	易齐干	1	24
打印数字温度计	易齐干	2	32
电源 ON、OFF 自动运转装置	易齐干	3	32
红外线遥控室内温度	易齐干	4	34
制作能暂停显示的计数器	易齐干	5	35
7 段数码管读数装置	易齐干	6	34
步进电机的控制	易齐干	7	33
水平多关节型机器人	易齐干	8	34
自动输送装置	易齐干	9	32
微电脑控制微型钻床	易齐干	10	30
微电脑控制微型钻床	易齐干	11	29

•电脑巧开发•

多功能程序移植器——一种简易的硬件工具	朱立钢	1	37	
单板机 RAM 分段与程序保护的实现	王新明	2	37	
微机视频接口器的研制	韦江维	覃龙生	3	37
XMF 学习机(单驱扩展箱)改为 XMF 及 PC 两用机	杨青海	3	38	

Z80单板机与打印机巧相连	张慧成	4	37
降低 MP-1 电脑的功耗	邓文超	4	40
简易字幕灯的制作与控制	杨宪泽	5	38
DOS 系统下多种软件在一台微机上的共存方法	尤建忠	5	40
.....			
自带 EPROM 类单片机简易加密编程	庄凯	6	38
一种简单的键盘接口电路	罗许建	6	39
ASCII 码字符显示器设计	李德文	7	38
点阵字符液晶显示器与单片机接口	张培仁 杨建景	8	38
.....			
计算机语音输出功能的开发与应用(上)	陈竹林	9	36
怎样使 CCDOS2.10 的九针打印驱动程序适应	李修连	9	38
2.13H 汉字系统	李修连	9	38
8031 真的无 ROM 吗?	肖革文	9	38
计算机语音输出功能开发与应用(下)	陈竹林	10	35
数字集成电路简易测试器	王正英	11	35
中华学习机调制伴音电路	李永和	11	39
中华学习机巧测电容	邓本富	12	35
家庭如何配置中华学习机	聂铁轮	12	33

·电脑游戏机·

第二章 FBASIC 的基本语句	于春	1	39
第三章 FBASIC 的画面控制语句	于春	2	41
第三章 FBASIC 的画面控制语句	于春	3	39
第三章 FBASIC 的画面控制语句	于春	4	41
第四章 FBASIC 语言的深入理解	于春	5	41
第四章 FBASIC 语言的深入理解	于春	6	40
第四章 FBASIC 语言的深入理解	于春	7	41
FBASIC 语言的游戏程序编写技巧			
第一讲 游戏程序概念	于春	8	41
第二讲 故事情节和游戏结构	于春	9	41
第三讲 程序结构和程序框图	于春	10	38
第四讲 游戏程序的设计过程(上)	于春	11	40
第四讲 游戏程序的设计过程(中)	于春	12	35

·维修经验谈·

微机安装和使用中应注意的几个问题	胡野红	1	43
CEC-I 中华学习机修理一例	卢光怀	1	44
软盘驱动器综合故障维修一例	周凯歌	1	44
PC-1500 计算机的维修	何静	2	44
导电橡胶按钮被磨损的修理方法	梁绍建	2	44
IBM-PC 机故障维修一例	欧阳波	2	44
NP125 型复印机中一个易被忽视的故障	许鹰	3	42
PC-88 键盘分析与故障维修	王耀亭	3	42
APPLE II 电源常见故障维修方法	邓满园	3	44
IBM PC/XT 机的软磁盘驱动器磁头校准程序	范思尧	3	45
.....			
中华学习机电源故障维修经验谈	汪晖	4	44
袖珍计算机 PC 型 PB 型系统 RAM 扩展模块			
常见故障与维修	朱杰	4	46
TH3070 打印机常见故障分析与检修(上)			
.....			
刘立华	5	44	
维修打印头应注意的几个问题	蔡世清	6	44

TH3070 点阵式打印机常见故障分析与检修(下)			
.....			
刘立华	6	44	
Super AT 机显示电源原理及维修	范志盛	7	45
利用 PCTOOLS 校正软盘驱动器磁头	黄焕如	7	48
单色显示器故障维修一例	梁建华	8	43
恢复 CEC-I 学习机 DOS 的 I/O 控制	屈晓柳	8	43
再谈清洗盘的正确使用	刘德一	8	44
对磁盘局部缺损的处理	何管略	8	44
IBM-PC/XT 及其兼容机 RAM 故障的检修方法			
.....			
齐吉泰	9	39	
APPLE-II CEC-I 故障诊断仿真系统			
.....			
王志刚 张一建	10	43	
笔记本型电脑技术和市场	赵广恩	11	43
用软件对软驱进行简单读写检测			
.....			
李晓中 麻佳洛 人府静 张景生	11	44	
LQ-1600K 打印机接口专用集成电路			
M54610P 的简易修复方法	赵继文	12	38
GW286 计算机硬盘控制卡故障处理一例			
.....			
杨远成	12	40	

·新书与软件·

电子工业出版社软件部新出版软件介绍		1	45
DATA BASE IV 关系数据库	张冰毅	2	45
电子工业出版社软件部新出版软件介绍		8	45
软件介绍		9	44

·读者联谊·

普及型 PC 个人用户软件交流联谊活动问题解答(一)			
.....			
王路敬	1	46	
普及型 PC 个人用户软件交流联谊活动问题解答(二)			
.....			
王路敬	2	46	
显示器·显示卡·显示系统	孙梅英	2	48
普及型 PC 个人用户软件交流联谊活动问题解答(三)			
.....			
王路敬	3	46	
普及型 PC 个人用户软件交流联谊活动问题解答(四)			
.....			
王路敬	4	48	
普及型 PC 个人用户软件交流联谊活动问题解答(五)			
.....			
王路敬	5	47	
普及型 PC 个人用户软件交流联谊活动问题解答(六)			
.....			
王路敬	6	48	
普及型 PC 个人用户软件交流联谊活动问题解答(七)			
.....			
王路敬	7	48	
普及型 PC 个人用户软件交流联谊活动问题解答(八)			
.....			
王路敬	8	46	
普及型 PC 个人用户软件交流联谊活动问题解答(九)			
.....			
王路敬	9	46	
普及型 PC 个人用户软件交流联谊活动问题解答(十)			
.....			
王路敬	10	46	
普及型 PC 个人用户软件交流联谊活动问题解答(十一)			
.....			
王路敬	11	46	
普及型 PC 个人用户软件交流联谊活动问题解答(十二)			
.....			
王路敬	12	42	



猴年进步!

一九九二年

总期第82期

電子與電腦

目 录

• 综述 •

猴年进步——本刊92年致读者 本刊编辑部(2)
我国单片机应用技术发展趋势及展望 何立民(2)

• PC 用户 •

自动调平四舍五入误差的简易快速软件 ... 李燕姝(4)
新颖的用户界面一下拉式菜单的实现 李晓峰(6)
再谈硬盘无效时回收文件的方法 岳千钧(9)
IBM 计算机几个重要硬件设备的软锁程序
..... 温盛文(10)
一个实用月历程序 沈玉江(11)
用拉幕方式显示信息 刘善平(11)
拼图游戏 刘晓峰(13)
通用文件病毒防御系统 姜金友(14)

• 学习机之友 •

苹果 BASIC 语言容错性浅析 那履弘(16)
Apple 内存校对发声程序 连 勤(19)
ProDOS 磁盘操作系统入门(续) 廖 凯(21)

• 语言讲座 •

6502 机器语言程序设计讲座 朱国江(23)

• 初级程序员级水平考试辅导讲座 •

计算机硬件基础知识 顾育麒(27)

• 学用单片机 •

8031 单片机最小系统 罗明宽 车金相(31)

• 学装微电脑 •

附带应急开关的顺序控制 易齐千(34)

• 电脑巧开发 •

多功能程序移植器——一种简易的硬件工具
..... 朱立钢(37)

• 电脑游戏机 •

第二章 F BASIC 的基本语句 于 春(39)

• 维修经验谈 •

微机安装和使用中应注意的几个问题 胡野红(43)
CEC-I 中华学习机修理一例 卢光怀(44)
软盘驱动器综合故障维修一例 周凯歌(44)

• 新书与软件 •

电子工业出版社软件部新出版软件介绍 (45)

• 读者联谊 •

普及型 PC 个人用户软件交流联谊活动问题解答(一)
..... 王路敬(46)

封面 猴年进步

封二 新年献词

封三 8088、80286 系统板故障诊断

封四 智力明星——四达儿童教育电脑

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

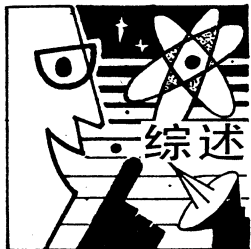
订购处:全国各地邮电局

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



猴 年 进 步

——本刊 92 年致读者

羊年过去,猴年到来。在新的一年里开始之际,我们《电子与电脑》全体工作人员向本刊国内外的新老读者、作译者、编委们、关心支持我们工作的领导和各界朋友们恭贺新春,祝大家猴年愉快,事业进步!

在过去一年里,本刊收到了许多热心朋友们的办刊建议、指导和希望。对于读者们参予办刊的热情,深受鼓舞,增加了我们办好刊物的信心和力量。我们一定努力克服主客观条件的困难,将刊物办的一年更比一年强,为我国电脑应用技术的普及,作出应有的贡献。

科技期刊作用的大小,取决于办刊的质量。在新的一年里,我们将从报道内容质量,编辑出版质量;为读者服务的质量等几大方面进一步改进工作。

一、提高办刊质量,增强刊物特色

1. 报道内容质量

本刊的读者层次多为初、中级水平的电脑应用人员。在此前提下,我们在刊物报道内容方面,不仅应注意政策性和科学性,更要突出普及性、启发性和实用性。

本刊读者群中,多数为自学电脑的爱好者,其学习的侧重点也不尽相同。为此,本刊在栏目设置上覆盖了电脑软件和硬件二方面。内容有一定的系统性,软硬件资料的透明度高,使初学者便于借鉴和移植,再辅之以必要的知识竞赛、联谊交流、函授教学和水平考试辅导、以及出版专题性的专辑,使读者通过刊物和社会活动结合起来,在实践中学习电脑应用技术。让我们共同努力把刊物办成普及电脑应用技术的开放式社会大学校。

2. 编辑出版质量

本刊主要栏目今年仍相对稳定,不拟作较大调整。编辑人员要积极组稿并认真审读来稿。不断提高文章加工水平,使文章层次结构清楚,逻辑性强,语言精炼顺畅,技术内容准确,减少文字和内容的差错率。

封面设计要富有特色,寓意和内涵确切,富有知识性和感染力。正文版面设计合理,图文协调,提高校对质量、印刷质量及出版准期率。

二、努力加强读者服务工作

科技期刊编辑部,一向把为读者提供信息、技术咨询、代购代邮等服务作为办刊的有机组成部分。去年,我们同中电总公司的北京振兴电子公司在京试办了《电子与电脑》读者服务部,开展了一些为读者邮购期刊和电脑零配件的工作,受到了读者的信赖和支持,由于人力和组织工作经验的不足,在邮购范围上,特别是技术咨询方面仍有较大差距,今后,我们将努力改进。

今年,为了搞好本刊的信息和技术咨询工作和为没有实践条件的读者开办一个具有技术实验条件并能将实验性设计转变为“产品”的《电子与电脑》技术开发服务部,正同有关单位协商具体事宜并办理申报手续。

希望有更多的读者通过刊物和我们的读者服务部,成长为具有一定应用水平的电脑人才,为我国电脑应用技术的普及和提高贡献力量。

《电子与电脑》

全体工作人员

1992. 元旦

我国单片机应用技术发展趋势及展望

北京航空航天大学 706 教研室(100083) 何立民

在我国,单片机的开发应用已有六年之余,形成了一支庞大的技术开发队伍,为我国单片机应用积累了丰富的经验。随着电子技术、计算机芯片技术、微电子技术的飞速发展,引起了单片机应用技术一日千里的变化。面对这种形势,不断地及时地了解、调查国内外现状、总结经验、分析技术发展趋势来指导我国单片机应用技术的健康发展具有重要意义。

由于我国微电子技术、计算机技术、半导体器件工业现状以及国民经济发展的模式,决定了我国在单片机应用系统的开发、研制方面走着与国外不尽相同的道路。为了指导我国单片机应用技术的发展,除了不断分析国内外市场形势,技术发展状况及其对我国开发环境的影响外,还应分析我国的技术状态、国情,提出中肯的意见,现提出一些不成熟的看法。

一、我国单片机主流芯片的发展趋势

我国单片机应用从起步到现在走的是一条自发地追随进口市场,沿国外 Intel 公司主干芯片更新变化的路径,从 MCS-48、MCS-51 到 MCS-96 及其新系列 8098、80C196。从 MCS-48 的淘汰和 8098、80C196 的兴起引起了对单片机主流机型的思考:我国单片机主流机型能否稳定下来,究竟能稳定在什么机型上。

单片机最确切反映其含义的名称是工业测控用微控制器,其应用对象决定了它的应用领域与应用形态。目前以及今后一个较长的时间内,8 位、4 位 CPU 已能满足 80% 测控对象的要求。因此,国外在单片机发展策略上十分重视 4 位、8 位机的研究,认识到单片机的位数只是单片机诸性能中的一个指标。对于工业测控对象来说,还有更重要的功能要求,如为提高系统可靠性的软件监视、电源监测、数据防改写及掉电保护;提高控制速度的高速 I/O 口;改善控制功能的可编程计数器阵列、多机通信的硬件识别以及 A/D 转换、PSW 电路、各种接口电路等。在不提高 CPU 数据总线宽度,甚至减小宽度至 4 位水平,把有限的集成度用来增加 I/O 口的数量与功能,甚至把一些模拟电路、功率电路集成到单片机中以提高其综合性能。

以上这些技术措施的基础是微电子技术的高度发展。我国现阶段盲目跟踪国外市售芯片的现象充分反映了我国微电子技术的落后状态。换言之,我国单片机技术的发展趋势在相当大的程度上决定于我国微电子技术和单片机国产化的进程。目前 MCS-48、MCS-51 系列单片机的国产化已有眉目。可以预计我国单片机的发展趋势。

1. 8 位机在相当长的时期内仍会是我国单片机的主流,根据用户不同要求会引进一些 8 位机的新产品系列。这些新型 8 位单片机完全具备或在某些方面已超过目前 16 位机的一些特殊功能。

2. 8 位机的国产化会使 8 位机在我国长期稳定,并开创我国 8 位单片机应用的崭新局面,如 8048、8051 的掩膜程序产品,典型系统的 ASIC 化,新型派生系列单片机的研制等。

3. 随着我国微电子技术的发展,ASIC 技术的成熟会导致 4 位机的应用热潮,因为 4 位机的应用只能走 ASIC 的道路。

4. 目前我国 8098、80C196 的供货及廉价开发环境为 16 位、准 16 位机的应用创造了极好的条件,但应用热中有一定的虚假现象。其中部分用户未必是非用 16 位 CPU 不可,而是看中其片内 A/D、高速 I/O 口、PSW 以及 80C196 的 CHMOS 工艺。实际上在 MCS-51 新型芯片中也具备了这些特性,甚至还有更特殊的功能模块,但苦于国内无供货渠道,只好选用国内供货方便,并有廉价开发环境的 8098 及 80C196。

二、单片机应用系统中的新器件

单片机应用系统中主要大规模集成电路依靠国外

市场的局面还会延续一个相当时期,但由于多年的努力,国内已形成较齐全、稳定的芯片供货市场,少量最新型器件也能快速引进,有利于及时跟踪国外先进技术。

1. 单片微型计算机

目前 8 位机仍是厂家重点发展的产品,包括早期的 MCS-48 及后来 MCS-51。其新型派生系列芯片沿两种模式发展:一是电子厂家用芯片,最典型的如飞利浦公司,将 MCS-48、MCS-51 为核心构成的 I²C 总线单片机,大量用于本公司的产品改造中;另一种是芯片厂家推出的新型市售单片机供用户使用。

16 位和准 16 位机也是芯片厂家注意发展的产品,以解决 8 位 CPU 无法解决的高速、复杂运算的应用场合。

国内在选用新型单片机时仍把目标盯在 Intel 系列上。MCS-51 系列新器件大多为 CHMOS 工艺,具有功耗低,抗干扰性能好、速度快、集成度高的特点。有了最早的 CHMOS 系列 80C51BH、80C31BH、87C51 以及继 8052、8352 后的 80C52、83C52。

1988 年以后 Intel 公司公布的新型单片机主要有带两个 DMA 的通用通信控制器 8XC152;带有可编程计数器阵列、可编程串行通道、增强掉电方式的 8XC51FA/FB;带 8 位 8 通道 A/D 的 8XC51GA/GB 以及带有大量 I/O 口的 8XC451。

16 位机、准 16 位机由于国内单片机开发公司的快速跟踪,无论在单片机芯片供货和开发环境都为用户创造了较好的条件,因此 8098、80C196 将会拥有不少新用户。

其它系列芯片,如 μ PD7810/11、Z86 系列单片机也是具有十分优异性能的单片机,限于国内开发环境的限制,受主流芯片的排斥,大面积推广有一定困难,但在一些地区仍有较好的市场。

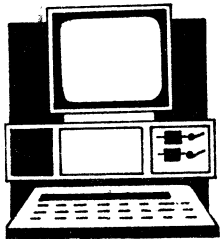
2. 存储器

存储器容量迅速增大,使 2764、27128、27256 成为当今单片机应用系统程序存储器的主流芯片,6264 成为数据存储器的主流芯片。

应用系统要求对 SRAM 的数据保护促进了形形色色带掉电保护 SRAM 的发展。目前先进的掉电保护集成芯片将可充电电池、充电电路、保护电路集成在一个芯片之中,如自保护存储器组件 9964,其外形和插脚与通用存储器类似,可直接替代 6116、6264、2716、2732、2764;DS1235YH 则是一种 32KB 集成掉电保护 SRAM 芯片,可直接替代 62256。

由于可靠的掉电保护、防改写电路的日趋成熟、焊接安装型可充电电池的出现,以及掉电保护集成 SRAM 芯片组件的出现,引起了人们对 E²PROM 前途的争议,可以预料,如果不提高 E²PROM 的写入速度就很难与掉电保护 SRAM 集成组件竞争。

(待续)



PC 用户

自动调平四舍五入误差的简易快速软件

天津电气传动设计研究所(300180) 李燕妹

一、问题的提出

在计划统计一类的报表处理中,一般基层表的计量单位:值用元、量用千克;当汇总后向上级机关报表时,值和量的单位却要转换为万元和吨。在变单位的过程中,由于要做四舍五入计算,其结果势必会产生基数和不等于合计数的误差。因此为使一个二维报表的横向和纵向的各个小小计、小计、合计和总计的合计关系保持平衡,必须对舍入误差进行调整。

二、自动调平舍入误差的设计

根据调整内容的不同,可将调整方式分为三种:横调、纵调和纵横调整。只对若干字段或若干记录进行变单位的报表,只需做横调或纵调即可。但对流向分析一类的报表,由于库文件的所有数据都需做单位换算,所以应进行纵横调整。

下面就调整舍入误差软件的设计思想和实现过程做一些说明,并将其程序清单附后。

首先由用户输入欲调整的 FOXBASE 数据库文件名、调整方式、合计数的字段号、记录号和基数的字段号(横调时指定)或记录号(纵调时指定)、以及调整位数(元变万元为 4、千克变吨为 3)。然后通过该软件运算几秒钟即可将所有数据调平。

横向调整时,由用户输入参加合计的各个基数的字段号,字段号间用逗号隔开,当字段号连续时,可只输入首尾两个字段号并用破折号隔开来简化操作。采用子串搜索函数 AT()和截子串函数 SUBSTR()将用户输入的字段号依次截取出来,并用字段名函数 FIELD()将字段号转变为字段名,便可生成以逗号为分隔符的字段名表 F,接着用拷贝命令 COPY TO ZJK FIELD &F 就可将所选各基数字段拷贝到中间库 ZJK 中。随后打开中间库,用散布命令 SCATTER TO T 将各基数字段的数据按顺序填入数组 T 的各个元素中。

纵向调整时,用类似的方法可把参加合计的各个基数的记录号截取出来,并依次将需调整的内容存入数组 T 的各个元素中,同时用删除命令 DELETE 给这些记录加上删除标志,由过滤命令 SET FILTER TO DELETE()把打上删除标志的记录筛选出来,通过拷贝命令 COPY TO ZJK FIELDS 行次,&Z1 就可将纵调选

择的基数字段拷贝到中间库 ZJK 中。

一旦需要调整的基数存入数组 T,该软件就要调用子程序 ZTG.PRG,将各项被调数据除以变单位的换算因子,并进行四舍五入运算。用 $CAS=ROUND(&Z1/10^WS,0)$ 把做了四舍五入的合计数存入变量 CAS 中,用 $U(J)=INT(T(J)/10^WS)$ 把各基数的整数部分存入数组 U 中,用 $V(J)=VAL(SUBS(STR(T(J)),11-WS,WS))$ 把各基数的小数部分存入数组 V 中,然后以合计数为准调整基数。此时做过舍入的合计数必然大于或等于只取整的各基数之和,所以如为等于,则无需调整,而如为大于,则应按差值的多少,看小数部分的大小,自大到小循环给相应基数的整数部分加 1,直到合计数与基数持平。这样就可将存放调平基数的数组 V 送入数组 T 中,以便下一步将它们反送回库文件的原始位置。

横调时仍用 AT()和 SUBS()函数顺次将字段名表 F 中的基数字段抽出来,通过替换命令 REPL 将数组 T 的各个元素逐一代入原记录中;而纵调时,需先将数组 T 的各个元素代入中间库中,然后通过更新命令 UPDATE 将中间库中的数据按行次更新被调字段的内容。

选择纵横调整时,采用先纵后横或先横后纵的方法,均是两种方式的综合,所以结果是一样的。本软件选用的是先横后纵的方法,即先以最大的合计数为准进行横调,再以该记录的各项分合计数为准进行纵调,最后将各记录的基数相加,反算出相应的横向合计数,即可自动地将各种合计关系调平。

纵横调整还可采用以记录号为内循环的控制条件,以字段号为外循环的控制条件,将所有要调整的数据存入二维数组中,随之在二维数组中折算单位四舍五入,然后视纵横双向的合计关系将数调平,最后仍以字段号和记录号为控制条件,将此二维数更新库文件的内容。由于篇幅的限制,在此就不再赘述了。

本软件不仅简易实用、快速准确,而且通用性好,实用性强,可为各种变单位的报表自动调平舍入误差使用。

程序清单

```
ZDTZ.PRG
set talk off
set safe off
publ z1,ws,gs,tzjl,bj
```

```
dime t(70),u(31),v(31),r(30)
stor 0 to hj,tzjl,c,gs,z2
stor spac(30) to jsh,jlh
stor "" to f
acce "请输入库名" to km
```

```
input "请输入变大单位的调整位数" to ws
clea
@ 2,24 say "选择调整方式"
@ 3,22 to 7,40 doub
```

```

@ 4,27 prom "1. 横向调整"
@ 5,27 prom "2. 纵向调整"
@ 6,27 prom "3. 纵横调整"
menu to x
sele 1
use &.km
if x=1. or. x=3
@ 10,23 say'请输入横向调整的;
合计字段号' get hj pict "99"
@ 11,0 say'请输入横向调整的基
数字段号,号间用逗号隔开,连续用破折
号连首尾'
@ 12,0 get jsh
@ 13,0 say'请输入纵向调整的合
计记录号';
get tzjl pict"99"
read
p=1
sele 1
jsh=trim(jsh)+' ','
l=1
do while .t.
b=at(',' ,',jsh)
if b<4
b1=subs(jsh,1,b-1)
gs=gs+1
else
c=at('- ',jsh)
c1=subs(jsh,1,c-1)
c2=subs(jsh,c+1,b-1-c)
do while val(c1)<=val(c2)
b1=c1
gs=gs+1
if f=" "
f=field(val(b1))
else
f=f+' ','+field(val(b1))
endif
f=trim(f)
c1=str(val(c1)+1)
l=l+1
endd
endif
b3=stuff(jsh,1,b,"")
if len(ltrim(b3)) # 0
b2=subs(jsh,b+1)
endif
if c=0
if f=" "
f=field(val(b1))
else
f=f+' ','+field(val(b1))
endif
f=trim(f)
l=l+1

```

```

endif
c=0
if len(ltrim(b3))=0
exit
endif
jsh=b2
endd
copy to zjk fields &.f
sele 2
use zjk
scat to t
do ztg
use
f=f+' ','
sele 1
use &.km
go tzjl
i=1
do while i<=1
b=at(',' ,',f)
y1=subs(f,1,b-1)
r(i)=y1
repl &.y1 with t(i)
b3=stuff(f,1,b,"")
if len(ltrim(b3)) # 0
b2=subs(f,b+1)
else
exit
endif
f=b2
i=i+1
endd
n=gs
endif
if x=2. or. x=3
if x=2
@ 12,23 say'请输入纵向调整的字
段号';
get hj pict "99"
@ 13,23 say'请输入纵向调整的合
计记录号' get tzjl pict"99"
read
n=1
endif
@ 14,0 say'请输入纵向调整的基
数记录号,号间用逗号隔开,连续
用破折号连首尾'
@ 15,0 get jlh
read
sele 1
copy stru exte to jg
use
sele 3
use jg
loca for field.name='行次'

```

```

if .not. found()
appe blan
repl field.name with'行次'
repl field.type with'c'
repl field.len with 2
endif
use
sele 4
crea tzk1 from jg
appe from &.km
repl all 行次 with str(recn(),2)
use
sele 1
use &.km
zap
appe from tzk1
jlh=trim(jlh)+' ','
q=1
do while q<=n
sele 1
p=2
gs=0
go top
do while .t.
b=at(',' ,',jlh)
if b<4
b1=subs(jlh,1,b-1)
loca for b1 $ 行次
gs=gs+1
if x=2
z1=field(hj)
else
z1=r(q)
endif
t(gs)=&.z1
dele
else
c=at('- ',jlh)
c1=subs(jlh,1,c-1)
c2=subs(jlh,c+1,b-1-c)
do while val(c1)<=val(c2)
loca for c1 $ 行次
gs=gs+1
if x=2
z1=field(hj)
else
z1=r(q)
endif
t(gs)=&.z1
dele
c1=ltrim(str(val(c1)+1))
endd
endif
b3=stuff(jlh,1,b,"")
if len(ltrim(b3)) # 0

```

```

b2=subs(jlh,b+1)
else
exit
endi
jlh=b2
endd
sele 1
set filt to dele()
go top
copy to zjk fields 行次,&z1
set filt to
sele 2
use zjk
reca all
do ztg
i=1
do while i<=gs
go i
z3=field(2)
repl &z3 with t(i)
z2=0
i=i+1
endd
sele 1
index on 行次 to tzsy
use &km index tzsy
upda on 行次 from b repl;
&z1 with b->&z3 random
use

```

```

sele 2
use
if x=3
sele 1
use &km
z2=field(hj)
if q=1
repl all &z2 with 0 for recn() # tzjl
endi
repl all &z2 with &z2+&z1;
for recn() # tzjl
endi
q=q+1
endd
endi
close all
dele file zjk.dbf
retu
ZTG. PRG
sele 1
go tzjl
if x=1. or. x=2. or. x=3. and. p=1
z1=field(hj)
cas=round(&z1/10^ws,0)
endi
if x=3. and. p=2
z1=r(q)
cas=&z1
endi

```

```

repl &z1 with cas
sele 2
stor 0 to u,v
j=1
do while j<=gs
u(j)=int(t(j)/10^ws)
v(j)=val(subs(str(t(j)),11-ws,ws))
cas=cas-u(j)
j=j+1
endd
do while cas>0
stor 0 to sz,wz
j=1
do while j<=gs
wz=iif(sz>=v(j),wz,j)
sz=iif(sz>=v(j),sz,v(j))
j=j+1
endd
u(wz)=u(wz)+1
v(wz)=0
cas=cas-1
endd
j=1
do while j<=gs
t(j)=u(j)
j=j+1
endd
retu

```

新颖的用户界面——下拉式菜单的实现

中国人民银行沈阳市分行(110014) 李晓峰

一、概述

下拉式菜单、弹出式菜单是目前流行的编程技术之一。它操作方便、快捷、形式新颖,扩大了屏幕的有效使用面积,为系统提供了一个良好的用户界面。

所谓下拉式菜单是指:一个运行的系统在屏幕上常驻一简单提示或一菜单,当用户打某一“热键”或在菜单上选择某一选项时,在屏幕某一位置上瞬间弹出相应的下一级菜单,当结束某一级菜单上的操作,退出到上一级菜单时,该菜单立即消逝,原屏幕内容恢复。

弹出式菜单是指深度只有一层的下拉式菜单。目前,很多软件,包括系统软件都采用了这种技术,例如: Turbo C 的集成调试环境 TC 等。下面介绍以 C 语言为工具,实现下拉式菜单的方法。

二、实现方法

描述一个菜单,需要这样一些属性:

- 菜单上的显示项目。
- 菜单左上角坐标(相对于整个屏幕坐标原点)。

- 当前光标所在的选项号。
- 菜单保存区的内存首地址。
- 菜单的前景、背景色彩。

... ..

当程序中菜单较多时,最好构造一个结构数组,以便于菜单的处理。例如:

```

typedef struct struc_menu {
char ** menu;
int x1,y1,x2,y2;
int cup;
char *p;
int count;
int foratt;
int bakatt;
... ..
}MMENU;

```

数组大小由菜单多少而定,每个菜单对应一个数组元素,这样就可用下标来调用一个菜单。

为了支持下拉菜单,相应于一个菜单的弹出—操作—消失,需要构造一系列函数:

•菜单生成函数:

该函数以一个菜单的部分属性为输入数据,计算菜单面积,并申请内存区做为菜单所占区域的保存区,生成全部属性后,将其存入一个结构数组元素中。

该函数在系统初始化时调用,每调用一次生成一个菜单。

•屏幕保存函数:

在一个菜单弹出前,把菜单将要覆盖的那部分屏幕的当前内容保存起来。

我们知道,屏幕的每个显示位置,在显示缓冲区中占两个字节空间:一个是显示代码,一个是显示属性。故对一个 m 行×n 列的菜单,实际应申请 2×m×n 个字节的空

有三种方法可实现屏幕保存:

1. 利用 BIOS 系统调用:

```
getch(char *zc, char *zs)
{
    _AH=8;
    _BH=0;
    geninterrupt(0x10);
    *zc=_AL;
    *zs=_AH;
}
```

在一个两层循环结构中,扫描矩形区域的每个显示位置,并调用上例函数,实现屏幕保存。

2. 直接读取显示缓冲区中的数据:这种方法比前一种速度快,但由于各种机器所配的显示卡不同,其显示缓冲区的地址也不相同,所以程序的可移植性差。

3. 如果用 Turbo C 编程,可使用 gettext 函数,既快又简便,即使保存整个屏幕,视觉上也没有停留感。

•菜单显示函数:

用菜单背景色清菜单区域,在菜单边界画框,然后,用前景色显示每一选择项。

•菜单操作函数:

在菜单的第一选项上,反转显示一光条,并支持下

列按键操作:
①光标键↑、↓、←、→和“热键”(选项首字母)用于移动反转光条。

②Esc 或其它特殊键用于结束操作。

③回车选中一个选项,并返回该选项序号。

•屏幕恢复函数:

在结束菜单操作后,把原屏幕内容恢复。相应于屏幕保存方法,也有三种方法恢复屏幕:

1. 利用 BIOS 系统调用。
2. 直接读写显示缓冲区。
3. 用 Turbo C 的 puttext 函数。

三、菜单的汉化显示

英文版 Turbo C 2.0 的集成环境下,不支持汉字,但是,我们可以在其它汉字编辑环境下,完成汉字部分的编辑。

Turbo C 的 printf、puts 函数可以完成汉字输出,但它们不支持色彩,而控制台的 cprintf、cputs 函数虽然支持色彩,却不支持汉字输出。因此,若想以丰富的色彩显示汉字,需另写一些输出函数,如:

```
void cdis_ch(char c,int fore,int bak,int n)
{
    union REGS r;
    r.h.ah=0x9;
    r.h.al=c;
    r.x.cx=n;
    r.h.bh=0;
    r.h.bl=(fore|(bak<<4))&0x7f; /* note: not
    flash!! */
    int86(0x10,&r,&r);
}
```

以 fore 为前景,bak 为背景,显示 n 个字符 c。

四、对汉字屏幕的保存与恢复

上述关于保存、恢复屏幕的方法以及目前其它一些专业刊物上登载的有关保存与恢复屏幕的技术,在英文系统下实现是没有问题的,但当我们在汉字系统下实现时,必然会遇到一个问题:即当菜单(或窗口)弹在汉字显示区域且边界跨在一个汉字的中间时,屏幕不能被有效恢复!!

原因是:若菜单左边界跨在某行的一个汉字中间时,由于保存的原该行显示数据是以一个汉字的第2个内码为首的一串显示内码及属性,所以在恢复屏幕重写该行时,这个汉字的第2字节内码与后续的内码(可能是汉字内码,也可能是 ASCII 码)重新组合显示,结果是一些杂乱的汉字或图形符。

若菜单右边界跨在某行的汉字中间,则恢复时,该行原内容可恢复,但边界上的汉字不能恢复。原因与上述类同。

下面介绍两个解决这个问题的方法:

1. 对每个菜单都申请一块足以保存整个屏幕的内存区,弹出前,扫描整个屏幕的每个显示位置,保存其显示代码和属性。然后弹出菜单。当要消掉菜单时,再用保存的数据重写整个屏幕。

这种方法实现简单,缺点是如果程序菜单的叠加程度很深时,需要较多的内存空间,并且由于扫描的面积大,若使用 BIOS 系统调用实现,视觉上有停留感。

2. 对一个 m 行、n 列的菜单,申请 2×m×n+m×4 个字节空间。当保存每一行最右列时,若是一个汉字的第2字节,则向左再读一个显示位置;当保存最右列时,若是汉字的第1字节,则向右再读一个显示位置。恢复屏幕时,处理逻辑类同(程序见清单)。

我用新写的 cgettext、cputtext 及其它函数,成功地用英文版 Turbo C 2.0 在 286 机上实现了一个具有全屏编辑功能的、汉化的、配有下拉菜单和弹出窗口的汇

```
表中:p=(unsigned char *)malloc(2*(x2-x1+1)*
(y2-y1+1))
pl=(unsigned char *)malloc(2*(x2-x1+1))
```

```

pr = (unsigned char *)malloc(2 * (x2 - x1 + 1))
void cgettext(int y1, int x1, int y2, int x2, char * p, char * pl,
char * pr)
{
int i, j, n;
char zc, zs;
char * zpl, * zpr;
zpl = pl;
zpr = pr;
for (i = x1; i <= x2; i++) {
for (j = y1; j <= y2; j++) {
gotoxy(j, i);
(void)GetTCE(&zc, &zs);
* p++ = zc;
* p++ = zs;
}
}
for (i = x1; i <= x2; i++) {
* zpl++ = '\1';
* zpl++ = '\1';
* zpr++ = '\1';
* zpr++ = '\1';
}
if (y1 != 1) {
for (i = x1; i <= x2; i++) {
gotoxy(y1, i);
n = GetCE(&zc, &zs);
if (n == 2) {
gotoxy(y1 - 1, i);
(void)GetCE(&zc, &zs);
* pl++ = zc;
* pl++ = zs;
}
else {
pl = pl + 2;
}
}
}
if (y2 != 80) {
for (i = x1; i <= x2; i++) {
gotoxy(y2, i);
n = GetCE(&zc, &zs);
if (n == 1) {
gotoxy(y2 + 1, i);
(void)GetCE(&zc, &zs);
* pr++ = zc;
* pr++ = zs;
}
else {
pr = pr + 2;
}
}
}
}
/* end 'cgettext' */

```

```

void cputtext(int y1, int x1, int y2, int x2, char * p, char *
pl, char * pr)
{
int i, j, n;
char zc, zs;
for (i = x1; i <= x2; i++) {
if (* pl != '\1') {
gotoxy(y1 - 1, i);
zc = * pl;
pl++;
zs = * pl;
pl++;
PutCE(&zc, &zs);
}
else {
pl = pl + 2;
}
for (j = y1; j <= y2; j++) {
gotoxy(j, i);
zc = * p;
p++;
zs = * p;
p++;
PutCE(&zc, &zs);
}
}
if (* pr != '\1') {
gotoxy(y2 + 1, i);
zc = * pr;
pr++;
zs = * pr;
pr++;
PutCE(&zc, &zs);
}
else {
pr = pr + 2;
}
}
}
/* end 'cputtext' */
GetCE(char * ch, char * att)
{
union REGS r;
r.h.bh = 0; /* 0 page */
r.h.ah = 0x48;
int86(0x10, &r, &r);
* ch = r.h.ah;
* att = r.h.ah;
return(r.h.bl); /* 0:ascii, 1:first, 2:second */
}
/* end 'GetCE' */
void PutCE(char * ch, char * att)
{
union REGS r;
r.h.bh = 0;
r.h.ah = 9;

```



```

r. x. cx = 1;
r. h. al = * ch;
r. h. bl = * att;
int86(0x10, &r, &r);
}

```

```
/* end 'PutCE' */
```

再谈硬盘无效时 回收文件的方法

长沙水利电力师范学院(410077) 岳千钧

贵刊1990年第12期《硬盘无效时回收文件的方法》一文,所叙述的恢复硬盘文件的方法较为繁琐,且不可靠,具体说来,有以下几点不足:

用 LOWFORM 对硬盘作低级格式化,当硬盘指示灯亮了以后再中断运行时,实际上已经有相当多的扇区被格式化,象引导扇区、文件分配表、根目录区及文件数据区起始部分的文件都很可能被破坏。文件本身被破坏以后就不可能恢复了,而文件分配表被破坏以后,硬盘上的文件是很难得到修复的,对于引导扇区和根目录区的修复,也是一件十分困难的事情,因此在用 PCTOOLS 恢复硬盘上的文件时,将会遇到很大的困难,恢复工作也将变得十分繁琐和复杂,有时甚至不能恢复。

其次,文中所述用 CTRL+C(03H)来中断 LOWFORM 的运行实际上是不安全的,CTRL+C 要中断可执行文件的运行需要满足一定的条件,即运行的应用程序有一个字符 I/O 操作的过程,也就是说,运行程序要监视键盘的输入,一旦检测到 CTRL+C(03H)时,则转入执行 INT 23H 中断处理程序,用 LOWFORM 对硬盘进行低级格式化时,它采用的是块设备驱动程序,因而没有字符 I/O 操作,这样的话,当硬盘正在作低级格式化时,按 CTRL+C 是中断不了的。

第三点,这种方法还有很大的局限性,它是通过对子目录的一个目录项“.”(2E,2E)的搜索来确定字目录数据区的起始位置,从而恢复子目录名,再用 PCTOOLS 恢复文件,这种方法因为无法确定根目录下的文件的起始位置,因此无法恢复根目录下的文件,这使它的使用受到很大的限制。

此外,《方法》一文中还有一个很明显的错误,文章中用 PCTOOLS 中的 Undelete 功能来恢复被删文件时,只恢复了文件的原名称,而没有恢复 FAT 表,因而硬盘文件不能运行只能拷贝,这种说法是不正确的,用 PCTOOLS 中的 Undelete 功能来恢复文件时,如果文件是可恢复的,则不仅恢复了文件的名称,同时也恢复了文件在 FAT 表中的簇链,否则,如果文件的长度大于一个簇,文件后面的内容就会丢失,这样拷贝过去的文件也是残缺不全的,实际上如果 FAT 表不能恢复,则

任何文件都不能通过 PCTOOLS 来恢复了。

事实上,微机运行中硬盘不能自举的故障一般都是由于以下三种原因引起的,第一是主引导记录损坏,它放在硬盘0柱面0磁头1扇区的位置,第二是 DOS 分区引导记录损坏,它放在硬盘的逻辑1扇区。第三是系统文件损坏,即 IBMBIO.COM、IBMDOS.COM 和 COMMAND.COM。象《方法》一文中所描述的故障一般都是由于主引导记录损坏引起的,其故障现象一般为:开机后硬盘不能自举,屏幕上显示 Missing operating system, Error loading operating system, Invalid partition table 等等,用软盘启动后不能进入硬盘,屏幕出现如下提示: Invalid drive specification。这种故障可以用如下的简单方法修复:

先从其它类型相同、DOS 版本也一致的机器上拷贝一个正常的主引导块,然后将这个引导块拷入坏的硬盘,方法如下:

```

A>DEBUG
-a100
xxxx:0100 mov dx,80
xxxx:0103 mov cx,1
xxxx:0106 mov bx,200
xxxx:0109 mov ax,0201
xxxx:010C int 13
xxxx:010E int 3
xxxx:010F <CR>
-g=100
-rcx
CX 0001
:200
-na:part
-w200
Writing 0200 byte
-q

```

这样我们就在 A 盘得到了一个正常的主引导块 PART,把软盘取出后再插入待修机器的 A 驱动器中,运行如下的程序:

```

A>DEBUG
-na:part
-1200
-a100
xxxx:0100 mov dx,80
xxxx:0103 mov cx,1
xxxx:0106 mov bx,200
xxxx:0109 mov ax,0301
xxxx:010C int 13
xxxx:010E int 3
xxxx:010F <CR>
-g=100
-q

```

这样就把一个正常的引导块拷入了坏硬盘的0磁头0柱面1扇区,再重新启动后,就可以进入硬盘了,硬盘上的文件也全部恢复了,笔者在实践中曾多次使用这种方法来恢复硬盘上的文件,取得了良好的效果。

IBM 计算机几个重要 硬件设备的软锁程序

广东省水电安装公司(511340) 温盛文

本人担任过几期 IBM PC 计算机培训班教师,对于初学者上机,有些设备是不允许动用的。比如打印机、硬盘甚至软盘驱动器都要限制使用。本人经分析内存中特殊存储单元的意义,分别制作了软盘、硬盘驱动器和打印机的软加锁程序。设备上锁后,如不重新启动,直到关机都有效,在一定程度上起到阻止使用这些设备的作用。各个加锁程序在 IBM PC/XT 机和 LQ-2500 打印机上调试通过。

以下介绍各个设备软锁程序的制作。

1. 驱动器 A: 软锁程序 LOCK-A.COM 的制作

```
C>DEBUG
-A 100
46FB:0100 MOV AX,0040
46FB:0103 MOV DS,AX
46FB:0105 MOV WORD PTR [0790],0000
46FB:010B INT 20
46FB:010D
-RCX
CX 0000
:D
-N LOCK.A.COM
-W
Writing 000D bytes
-Q
```

运行 LOCK-A.COM 后,A:驱动器便不会被读写,连指示灯也不会亮。这样,可避免初学者使用有病毒的软盘,也可防止没插入软盘时使用读写命令造成驱动器空转,从而保护磁头。如对 A:进行读写操作,将出现:

```
Sector not found error reading drive A
Abort,Retry,Fail?
```

但驱动器不转,指示灯不亮。

2. 驱动器 B: 软锁程序 LOCK-B.COM 的制作

```
C>DEBUG
-A 100
46FB:0100 MOV AX,0040
46FB:0103 MOV DS,AX
46FB:0105 MOV WORD PTR [07E2],0000
46FB:010B INT 20
46FB:010D
-N LOCK-B.COM
-RCX
```

```
CX 0000
:000D
-W
Writing 000D bytes
-Q
```

运行 LOCK-B.COM 后,B:驱动器便不能被读写,现象同运行 LOCK-A.COM 后相同。

3. 硬盘的软锁程序 LOCK-C.COM 的制作

```
C>DEBUG
-A 100
46FB:0100 MOV AX,0040
46FB:0103 MOV DS,AX
46FB:0105 MOV BYTE PTR [0075],00
46FB:010A INT 20
46FB:010C
-RCX
CX 0000
:000C
-N LOCK.C.COM
-W
Writing 000C bytes
-Q
```

运行 LOCK-C.COM 后,打入对硬盘的读写命令,硬盘没有反应,指示灯不亮,且显示:

```
General Failure error reading drive C
Abort,Retry,Fail?
```

这可防止对硬盘数据和程序的破坏。

4. 打印机软锁程序 LOCKPRN.COM 的制作

```
C>DEBUG
-A 100
46FB:0100 MOV AX,0040
46FB:0103 MOV DS,AX
46FB:0105 MOV BYTE PTR [0008],00
46FB:010A INT 20
46FB:010C
-N LOCKPRN.COM
-R CX
CX 0000
:000C
-W
Writing 000C bytes
-Q
```

运行 LOCKPRN.COM 后,打印机将不能打印。无论是否 ON LINE,是否有纸,进行打印操作都显示:

```
No paper error writing device PRN
Abort,Retry,Fail?
```

这可暂时避免使用打印机。

设备加锁后,如要解锁,可重新启动。

一个实用月历程序

沈玉江

月历程序已为广大的计算机爱好者所熟悉,但笔者迄今为止所看到的月历程序都是不带农历的。为了弥补这个缺陷,笔者试着编写并在PC机上调试成功了此程序。运行此程序(以1991年日历为例)后,能同时显示农历。

一、程序功能

只要将公元的年数、每月天数和农历每月天数输入后,即可获得如下效果:

- 1、每季度首末都打印星期标志。其中周一到周六(M1—M2)为黄色,星期天(SU)为红色;
- 2、周一至周六为公历都用蓝色打印,紧随其后的农历(除每月第一天外)都用青色打印;
- 3、星期日的公历和农历(除每月第一天外)都用亮洋红打印;
- 4、农历每月的第一天都用洋红打印,并且用“Y+该农历月份数”表示,以便识别;
- 5、在每月的左侧打印出该公历月名称(用数字表示);

二、程序说明

(一)、变量说明:

YN——公元年(本例为1991);TN 星期(初始值表示元旦是星期几);NL——农历(初始值表示元旦的农历);D 数组——公历每月天数;NAM\$ 数组——星期标志;X—农历每月天数;NX——农历月份;S——每月打印行数指针。

(二)、部分语句说明:

- 1、160和170语句是为了处理公历和农历月份不一致而设置的,每年的情况均不同;
- 2、100和250语句是为了让公历月名称打印在该月内容的第三行的位置上;
- 3、320——330语句是为显示控制语句,可改成其它形式或者不要;
- 4、390——440语句为打印星期标志子程序;
- 5、370——380语句分别存放公历和农历每月天数,本程序380语句中的前两个数据(30、30)为农历90年11月和12月的天数;
- 6、改动部分语句后可将星期日显示在每周的最前面;将程序中的色彩显示语句去掉后,可在黑白显示器上显示;稍作改动,即可在彩色打印机上打印出彩色日历。

```
10 SCREEN 0:WIDTH 80:COLOR 1,6:CLS
20 YN=1991
30 TN=2:NL=16:DIM D(12)
```

```
40 FOR K=1 TO 7:READ NAM$(K):NEXT K
50 FOR K=1 TO 12:READ D(K):NEXT K
60 PRINT TAB(32);“***”;:COLOR 4:PRINT YN;:COL
  OR 1:PRINT“***”:PRINT
70 READ X:NX=1
80 FOR I=1 TO 12
90 IF ((I-1)/3)=INT((I-1)/3) THEN GOSUB 390
100 S=1
110 FOR K=1 TO D(I)
120 IF TN=7 THEN COLOR 13
130 PRINT TAB(TN*10);USING“\”;STR$(K);
140 IF NL(<)1 THEN COLOR 3:GOTO 190
150 COLOR 5
160 IF NX=2 THEN PRINT USING “-(Y##-)”;12;:
  GOTO 180
170 PRINT USING “-(Y##-)”;NX-2
180 COLOR 1:GOTO 210
190 IF TN=7 THEN COLOR 13
200 PRINT USING“-(##-)”;:COLOR 1
210 IF NL(<)X THEN NL=NL+1:GOTO 230
220 READ X:NL=1:NX=NX+1
230 IF TN(<)7 THEN TN=TN+1:GOTO 270
240 TN=1:S=S+1:PRINT
250 IF S(<)3 THEN 270
260 COLOR 4:PRINT“*”;I;“*”;:COLOR 1
270 NEXT K
280 IF TN=1 THEN 300
290 PRINT
300 IF I/3(<)INT(I/3) THEN PRINT :GOTO 340
310 GOSUB 390:PRINT :PRINT
320 A$=INKEY$:IF A$="" THEN 320
330 IF A$(<)CHR$(13) THEN BEEP:GOTO 320
340 NEXT I
350 END
360 DATA M1,T2,W3,T4,F5,S6,SU
370 DATA 31,28,31,30,31,30,31,31,30,31,30,31
380 DATA 30,30,29,30,29,29,30,29,29,30,29,30,30
390 COLOR 14:FOR J=1 TO 6
400 PRINT TAB (3+J*10);NAM$(J);
410 NTXT J
420 COLOR 4:PRINT TAB (73);NAM$(7)
430 COLOR 1
440 RETURN
```

用拉幕方式显示信息

烟台师范学院(264000) 刘善平

各种信息的显示都需要有个好的显示效果,尤其是应用程序的封面和菜单的显示。目前,各种程序显示信息多采用滚屏式或下拉式。这里,提供一种更美观、更新颖的显示方式:拉幕式。

设计思想:将需要显示的信息以文件的形式存入

磁盘,显示时从盘中取出数据装入4K长度的显示缓冲区,而后按相应顺序把缓冲区的数据显示出来。显示前,先分别从左右两边向中间显示反相的空串列,这样不论原先屏幕上有什么内容都能有一种拉闭屏幕的效果。显示时,再从中间分别向左向右依次显示被存储屏幕的每一列的信息。这样屏幕就又被拉开了,所需的画面也就呈现在眼前。

具体实现:根据上述设计思想,笔者编写了汇编语言源程序 SAVE. ASM 和 LOAD. ASM。读者可用编辑程序输入 SAVE. ASM 和 LOAD. ASM,进行编译和连接以后,再用 EXE2BIN 分别生成 SAVE. COM 和 LOAD. COM。SAVE. COM 的功能是:把要显示的信息以文件的形式存入磁盘。LOAD. COM 的功能是:将需显示的内容从磁盘上取出,再以拉幕的方式显示出来。比如:在 dBASE III 环境下,可以编程序用 TEXT—ENDTEXT 或 SAY 语句将画面制作好,用 RUN SAVE. COM 把画面存入磁盘。在需要显示时再用 RUN LOAD. COM 把画面以拉幕的方式显示出来。

SAVE. ASM 清单:

```

PRG1 SEGMENT PARA PUBLIC 'CODE'
    ORG 100H
    ASSUME CS:PRG1,DS:PRG1,ES:PRG1,SS:PRG1
PRG2 PROC NEAR
    JMP NM1
DATA DB 4000 DUP(?) ;行数×列数×2
FNAM DB 'C:SCR',0 ;路径及文件名
FILE DW 0
NM1:MOV AH,3CH
    XOR CX,CX
    MOV DX,OFFSET FNAM
    INT 21H
    MOV AH,3DH
    MOV AL,1
    MOV DX,OFFSET FNAM
    INT 21H
    MOV FILE,AX
    MOV SI,0
    MOV DH,0
NM2:MOV DL,0
NM3:MOV AH,2
    MOV BH,0
    INT 10H
    MOV AH,8
    INT 10H
    MOV [DATA][SI],AL
    INC SI
    MOV [DATA][SI],AH
    INC SI
    INC DL
    CMP DL,80
    JNE NM3
    INC DH
    CMP DH,25 ;行数
    JNE NM2

```

```

    MOV AH,40H
    MOV BX,FILE
    MOV CX,4000 ;行数×列数×2
    MOV DX,OFFSET DATA
    INT 21H
    MOV AH,3EH
    MOV BX,FILE
    INT 21H
    MOV AX,4C00H
    INT 21H
PRG2 ENDP
PRG1 ENDS
    END PRG2
LOAD. ASM 清单:
PRG1 SEGMENT PARA PUBLIC 'CODE'
    ORG 100H
    ASSUME CS:PRG1,DS:PRG1,ES:PRG1,SS:PRG1
PRG2 PROC NEAR
    JMP NM1
DATA DB 4000 DUP(?) ;列数×行数×2
FNAM DB 'C:SCR',0 ;路径及文件名
FILE DW 0
NN DB 0
NM1:MOV NN,80
    MOV DL,80
NM2:SUB DL,NN
    MOV DH,0
NM3:MOV AH,2
    MOV BH,0
    INT 10H
    MOV AL,' '
    MOV BL,112
    MOV BH,0
    MOV AH,9
    MOV CX,1
    INT 10H
    INC DH
    CMP DH,25 ;行数
    JNE NM3
    DEC NN
    ADD DL,NN
    MOV DH,0
NM4:MOV AH,2
    MOV BH,0
    INT 10H
    MOV AL,' '
    MOV AH,9
    MOV BH,0
    MOV BL,112
    MOV CX,1
    INT 10H
    INC DH
    CMP DH,25 ;行数
    JNE NM4
    DEC NN

```

```

CMP NN,0
JNE NM2
MOV AH,3DH
MOV AL,0
MOV DX,OFFSET FNAM
INT 21H
MOV FILE,AX
MOV AH,3FH
MOV BX,FILE
MOV CX,4000 ;列数×行数×2
MOV DX,OFFSET DATA
INT 21H
MOV NN,0
MOV DL,39
NM5;SUB DL,NN
MOV DH,0
NM6;MOV AL,160
MUL DH
MOV SI,AX
MOV AH,2
MOV BH,0
INT 10H
MOV AL,2
MUL DL
ADD SI,AX
MOV AL,[DATA][SI]
MOV BL,[DATA][SI+1]
MOV BH,0
MOV AH,9
MOV CX,1
INT 10H
INC DH
CMP DH,25 ;行数
JNE NM6
INC NN
ADD DL,NN
MOV DH,0
NM7;MOV AL,160
MUL DH
MOV SI,AX
MOV AH,2
MOV BH,0
INT 10H
MOV AL,2
MUL DL
ADD SI,AX
MOV AL,[DATA][SI]
MOV AH,9
MOV BH,0
MOV BL,[DATA][SI+1]
MOV CX,1
INT 10H
INC DH
CMP DH,25 ;行数
JNE NM7

```

```

INC NN
CMP NN,80
JNE NM5
MOV AH,3FH
MOV BX,FILE
MOV CX,4000 ;列数×行数×2
MOV DX,OFFSET DATA
INT 21H
MOV AH,3EH
MOV BX,FILE
INT 21H
MOV AX,4C00H
INT 21H
PRG2 ENDP
PRG1 ENDS
END PRG2

```

拼图游戏

浙江杭州市米山路60号(310022) 刘晓峰

目前,市场上有一种拼图板,由八块小方块组成,各个小方块上分别是图案的一部分,通过一定的规则移动它们,使它们拼成一幅完整的图案。

我在XZ-PC机上,利用GW BASIC的图形语句,用数字代替图案,实现这一游戏过程的模仿。游戏者利用I:↑、K:↓、J:←、L:→、四键来控制小方块的移动。使它们最终成为规则的顺序排列。

本程序主要是利用了PUT和GET语句,把每个小方块看成是一个独立的部分,用一个数组Y(3,3)与之对应,使屏幕上小方块的排列与二维数组Y对应。由改变Y(3,3)的值来实现小方块的动态移动。

程序20—60构造小方块。80—127产生小方块的一种随机组合。500—2600通过键盘控制小方块,并由程序判断有否成功。

```

5 DIM A1(35),A2(35),A3(35),A4(35),A5(35),A6(35),
A7(35),A8(35),A9(35)
10 SCREEN 1,CLS,KEY OFF
20 LINE (7,0)-(30,20),2,B
30 FOR I=0 TO 8:T=I+1
40 LOCATE 2,3;PRINT CHR$(49+I);
43 IF T=1 THEN GET (7,0)-(30,20),A1;GOTO 60
45 IF T=2 THEN GET(7,0)-(30,20),A2;GOTO 60
47 IF T=3 THEN GET (7,0)-(30,20),A3;GOTO 60
49 IF T=4 THEN GET (7,0)-(30,20),A4;GOTO 60
51 IF T=5 THEN GET (7,0)-(30,20),A5;GOTO 60
53 IF T=6 THEN GET (7,0)-(30,20),A6;GOTO 60
55 IF T=7 THEN GET (7,0)-(30,20),A7;GOTO 60
57 IF T=8 THEN GET (7,0)-(30,20),A8;GOTO 60

```

```

59 IF T=9 THEN GET (41,2)-(62,20),A9
60 NEXT;CLS;CS=0
80 LINE(0,0)-(190,300),2,B
100 LINE(40,40)-(119,110),2,BF
105 DIM B(9),Y(3,3):FOR I=1 TO 9:B(I)=0:NEXT
106 IF INKEY$=" " THEN B(0)=RND;GOTO 106
107 FOR I=1 TO 9
112 IF I<=3 THEN T=I;U=1
113 IF I>3 AND I<=6 THEN T=I-3;U=2
114 IF I>6 THEN T=I-6;U=3
115 B(0)=INT(1+RND(1)*9)
120 IF B(B(0))=1 THEN 115
125 Y(T,U)=B(0);GOSUB 130
127 NEXT;GOTO 500
130 IF Y(T,U)=1 THEN PUT (17+T*23+4,20+U*20
+2),A1,AND;B(1)=1;GOTO 300
150 IF Y(T,U)=2 THEN PUT (17+T*23+4,20+U*20
+2),A2,AND;B(2)=1;GOTO 300
170 IF Y(T,U)=3 THEN PUT (17+T*23+4,20+U*20
+2),A3,AND;B(3)=1;GOTO 300
190 IF Y(T,U)=4 THEN PUT (17+T*23+4,20+U*20
+2),A4,AND;B(4)=1;GOTO 300
210 IF Y(T,U)=5 THEN PUT (17+T*23+4,20+U*20
+2),A5,AND;B(5)=1;GOTO 300
230 IF Y(T,U)=6 THEN PUT (17+T*23+4,20+U*20
+2),A6,AND;B(6)=1;GOTO 300
250 IF Y(T,U)=7 THEN PUT (17+T*23+4,20+U*20
+2),A7,AND;B(7)=1;GOTO 300
270 IF Y(T,U)=8 THEN PUT (17+T*23+4,20+U*20
+2),A8,AND;B(8)=1;GOTO 300
280 IF Y(T,U)=9 THEN PUT (17+T*23+5,20+U*20
+3),A9,AND;B(9)=1;JX=T;JY=U
300 RETURN
500 LOCATE 17,3;PRINT "i."CHR$(24);" " "k."CHR
$(25);" "; "l."CHR$(26);" "; "j."CHR$(27);
LOCATE 19,8;PRINT "e,end"
510 LOCATE 21,5;PRINT "?";I$=INKEY$;LOCATE 2,2;
PRINT CS,TIME$
545 IF I$="e" OR I$="E" THEN 2600
550 IF I$="i" OR I$="I" THEN Y(JX,JY)=Y(JX,JY+
1);Y(JX,JY+1)=9;GOSUB 1300
570 IF I$="k" OR I$="K" THEN Y(JX,JY)=Y(JX,JY
-1);Y(JX,JY-1)=9;GOSUB 1300
590 IF I$="l" OR I$="L" THEN Y(JX,JY)=Y(JX-1,
JY);Y(JX-1,JY)=9;GOSUB 1300
610 IF I$="j" OR I$="J" THEN Y(JX,JY)=Y(JX+1,
JY);Y(JX+1,JY)=9;GOSUB 1300
620 GOTO 510
1300 LINE(40,40)-(119,110),2,BF;CS=CS+1
1500 FOR T=1 TO 3;FOR U=1 TO 3
1640 W=W+1;GOSUB 130
1800 NEXT U;NEXT T;GOTO 2000
1900 RETURN
2000 YSZ=0
2050 FOR W=1 TO 3;FOR V=1 TO 3

```

```

2150 YSZ=YSZ+1
2200 IF P(V,W)<>YSZ THEN 1900
2300 NEXT V;NEXT W
2400 LOCATE 21,4;PRINT "Continue?(y/n)";INPUT A$
2500 IF A$="y" THEN 70
2600 SCREEN 0;END

```

通用文件病毒防御系统

合肥矿山机器厂研究所(230011) 姜金友

文件病毒是一种行为恶劣的病毒,它的载体是扩展名为.COM和.EXE的程序。现已发现的文件病毒有:黑色星期五、扬基、维也纳、音乐、1590(1575)等,这类病毒不象引导型病毒那样容易预防(只要用无病毒的系统盘引导即可),并且破坏性大,隐蔽性强,发作时常毁坏文件。现有的免疫系统都只能预防某种具体的文件病毒。有没有一种方法能够预防各种各样文件病毒的发作和传播呢?答案是肯定的。

笔者通过分析现今出现的各种文件病毒,发现它们有一个共同之处,即都是通过修改INT21H的入口,使其指向病毒体而获得控制权。由此可见,只要禁止修改INT21H的入口地址,则病毒即使驻留内存,也无法传染给其他文件,更谈不上破坏了。

如何禁止修改21H中断向量?可以在系统未染上文件病毒之前,修改INT21H使之指向免疫程序。该免疫程序常驻内存,它首先获取21H中断向量的入口地址,再同正确的相比较,若INT21H入口被修改,则再次修改INT21H入口,使之重新指向免疫程序,然后再执行INT21H所要求的功能调用。以下是根据该原理开发的免疫软件源程序清单。

C>TYPE PFV.ASM

```

pushr      macro
push ax
push bx
push cx
push dx
push ds
push es
endm
popr       macro
pop es
pop ds
pop dx
pop cx
pop bx
pop ax

```

```

        endm
call_old_int macro
    pushf
    cli
    call dword ptr cs:old_int21
    sti
    endm
code segment
    assume cs:code
    org 100h
start:   jmp install
old_int21 dw?,?
int21    dw?,?
msg1     db "Prevent V1. 0 by J. Jinyou, 6/6/1991",24h
msg2     db 0ah,0dh,"Your computer now may avoid files virus!!!",24h
msg3     db 0ah,0dh,"Prevent already installed!!!",24h
flag     dw?
new_int21 proc far
    pushr
    push cs
    pop ds
    mov ax,3521h
    call_old_int
    cmp int21,bx
    jnz recover
    mov ax,es
    cmp int21+2,ax
    jz ok
recover: mov ax,2521h
    lea dx,new_int21
    call_old_int
ok:      popr
    cli
    jmp dword ptr cs:old_int21
new_int21 endp

install: assume ds:code
    push cs
    pop ds
    mov ah,09
    lea dx,msg1
    int 21h
    mov ax,0
    mov ds,ax
    mov bx,182h
    cmp word ptr [bx],4a59h
    jz exit

    mov ax,4a59h
    mov [bx],ax
    push cs
    pop ds
    mov ah,09
    lea dx,msg2
    int 21h
    mov ax,3521h
    int 21h
    mov old_int21,bx
    mov old_int21+2,es
    mov ax,2521h
    lea dx,new_int21
    call_old_int
    mov ax,3521h
    call_old_int
    mov int21,bx
    mov int21+2,es
    lea dx,install
    mov cl,4
    shr dx,cl
    inc dx
    mov ax,3100h
    int 21h
exit:   push cs
    pop ds
    mov ah,09
    lea dx,msg3
    int 21h
    ret
code ends
end start

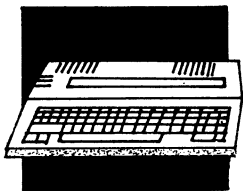
```

编译并连接上程序后,用 EXE2BIN.EXE 将之转化为.COM 文件。

笔者经过多次验证发现,一旦在内存中加载了 PFV.COM 程序,就可以预防迄今为止所发现的所有文件病毒,当然要在内存中尚无文件病毒时运行该程序(最好修改 AUTOEXEC.BAT 文件使之—开机就执行 PFV.COM)。

《电子电脑》1991年下半年合订本已经出版,每本定价7.8元,欲购者请附加15%邮费。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



学习机之友

苹果 BASIC 语言容错性浅析

沈阳黄金学院(110015) 那履弘

苹果机浮点 BASIC (APPLESOFT) 是解释性语言, 修改原程序较为方便, 使用这一语言的人多为青少年和初学者, 因此, 对于大多数的使用者来说, 往往在编制程序时不太注意程序语法, 而希望计算机运行解释程序时能甄别出自己程序中的错误的性质和出错的行号, 这就是通常所说的“动态侦错”。

在解释性语言中, 动态侦错的完善性是容错性的一个很重要的方面。容错性具有容留错误的含义, 但是它所容留的错误不应导致程序的失误, 因此它与侦错是不可分割的。一般来说, 作为一种高级语言, 应该具有充分完善的容错性, 应能识别出用户编制的程序中的任何导致失误的错误。但是, 解释程序也是某个或是某些人编制的, 由于程序规模和思考问题等各方面的原因, 难免存在着一些错误的情况超出了预先设计的范围, 因而造成某些错误不能侦出, 有些出错处理不当的情况。

由于人们对于容错性缺乏深入的了解, 往往把通过运行的程序认为是正确的程序。这个结论不完全正确。实际上, 除了数据、公式和程序流程等方面的错误之外, 仍有一些语法上的错误能混过解释程序的侦错检查。本文重点指出一些程序运行中 APPLESOFT 不能发现的语法错误和一些侦错后处理不当的例子。在这些例子中, 有些可能导致错误的计算结果、错误的程序流向, 有些可能造成“死机”的情况。

为使读者对容错性有进一步的认识, 本文还针对一些实例, 对容错性进行粗浅的分析, 希望能对读者, 特别是编制小规模商业化软件的读者有所帮助。

为叙述方便, 文中把以行号为标志的语句称为语句行或行, 把以“:”分隔的语句称为语句或句, 并将语句中可以替换的关键字用“()”括起来。

一、行号侦错

BASIC 程序的每一条叙述称为语句, 语句的标志是行号。浮点 BASIC 语言的行号由纯数字构成, 取值范围是 0—63999, 除了写在行首作为语句行的标志之外, 还有若干个命令以行号作为参数或可选参数。

这些命令是

GOTO; GOSUB;
ON (开关变量表达式) GOTO;
ON (开关变量表达式) GOSUB;
ON ERROR GOTO;
IF(表达式) THEN(GOTO)(或 GOSUB);
RUN; DEL; LIST.

APPLESOFT 有一段编辑解释程序, 每当使用者键

入一段程序并按下 RETURN 键, 这段解释程序便将键盘缓冲区的程序移入程序区。写在行首的行号在这个过程中侦错, 任何在取值范围内的, 以数字开始的连续的数码被解释成合法的行首行号。行首行号是不容错的, 如果插进了非数字代码, 那么解释程序仅将前面的数码作为行号, 非数字代码和后面的代码将作为语句行的内容, 并在程序运行时予以解释。但是上面列举的 GOTO 等命令的行号参数不在编辑时侦错, 这是解释性语言逐句解释的特点所决定的。这些非行首的行号参数随着它所依附的命令的不同, 也有着不同的容错性, 下面分别列举几种情况加以分析。

1. GOTO 语句

语法书中介绍 GOTO 语句是由 GOTO 命令和数字行号两部分构成的; 行号不能省略。请看下面例子。

```
0 PRINT " * ";  
10 GOTO
```

运行这段程序, 屏幕上会布满“*”号, 实际上进入了死循环状态。

分析解释程序可知, 程序的行号无论写在哪一个命令之后都首先由一段共同的子程序来解释。这个子程序首先将存放行号的单元清零, 再去找数字, 将数字的值存入行号单元, 一旦发现非数字代码便返回主程序。这段子程序只能发现行号超出取值范围的错误, 其他的错误不能侦出。在这个例子里, GOTO 命令后面没有跟随行号参数, 因此解释程序便认为 0 是要转移的行号。很多程序不常有 0 行, 省略行号便会出错, 因此初学者认为省略行号是非法的。一些计算机语言专家认为, 类似行号问题, 解释系统不应提供缺省值, 以防止引起误解, 这个观点是很有道理的。

由于行号的解释程序遇到非数字代码便认为行号结束, 实际上引用了“非”逻辑。非数字代码不是唯一的, 因此行号参数便有了容错性。通过分析可知:

```
GOTO 10.5       等于   GOTO 10  
GOTO 20+A      等于   GOTO 20  
GOTO B         等于   GOTO 0
```

看起来上述语法错误是不容易出现的, 但是一些初学者可能将数字零打成字母 O; 在键入数字时, 可能夹入不显示出来的控制字符, 这些错误初学者不易发现。由此可以看出引用“非”逻辑后侦错是不够强有力的。GOTO 语句是绝对转移语句, 除了 DATA 和 REM 语句之外, 任何其他的命令写在 GOTO 语句行的后面均没有实际的意义。解释程序考虑到 GOTO 语句后面没有可执行语句, 便不继续侦错是有一定道理的, 但是类似

夹入控制符之类的错误很难由屏幕清单上反映出来,因此也具有不利因素。作为一个系统软件也不应该以含混的解释形成所谓的技巧。实际上在 GOTO 语句中再对非数字代码进行验证,看其是否为“:”号或零,那么就不能出现上述的问题,引用“是”逻辑只占用很少的字节,代价不大。

2. GOSUB 语句

GOSUB 语句的格式与 GOTO 语句的格式有相似之处,也需要有行号作为它的参数,实际上由于对行号的解释用同一段子程序,因而也会出现 GOTO 语句中出现的问题。由于 GOSUB 语句要与 RETURN 语句成对使用,而 RETURN 语句又难以写在 0 行,因此 GOSUB 语句省略行号是难以实现的。GOSUB 语句也有容错性和侦错方面的问题,请看下例。

```
10 A=20:GOSUB 60 B=A:A=0
20 PRINT B,A:END
60 RETURN
```

在第 10 语句中,显然是丢掉了 GOSUB 语句后面的“:”号。但是运行这段程序,并没有出现错误中断或出错提示;打印出的 B,A 两值均为零。通过这个现象可以看出,赋值语句 B=A 没有执行,因而 B=0。分析解释程序可知,GOSUB 语句要经 RETURN 语句返回到本行,因此要保护现场,也就是要将 GOSUB 语句当前的地址、行号和 GOSUB 命令的标志符入栈保护。为了简化解释程序;为了使行号具有容错性,解释程序在解释 GOSUB 语句时,找到行号之后,它不是把下一字符的地址入栈,而是要找到“:”后,再将地址入栈。所有夹在行号和“:”之间的其他代码统统作为废料处理。在这个原则下,就有下面的关系:

GOSUB 60.5 等于 GOSUB 60

不出错,这样容错性似乎广泛了,但是也出现了可能丢失语句的问题。如果错将 GOSUB 语句后面的“:”号打成“;”号,也会发生丢失语句的情况,这种错误是解释程序不能侦出的,人的检查也容易忽视两个符号之间的差别。全面衡量利弊,该语句的侦错功能似应加强,对这一语句的容错性应予以重新认识。

3. ON...GOTO(GOSUB) 语句

ON...GOTO(GOSUB) 可有多个行号作为参数,其他有两个以上行号参数的语句还有 LIST,DEL 等。

解释程序在解释以多个行号为参数的语句时,侦错要严格一些,解释程序不仅检查行号是否在取值范围,而且还要核对行号之间的分隔符代码是否符合规定。请看下例。

```
10 ON A GOTO 50,60.5,,70,80.8
   当 A 值等于 1 时,相当于 GOTO 50;
   当 A 值等于 2 时,相当于 GOTO 60;
   当 A 值大于 2 时,程序将侦出错误,即行号 60
```

后面的分隔符不是所要求的逗号。如果将行号 60 后面的“.5”去掉,那么:

当 A 值等于 3 时,相当于 GOTO 0,在这里,解释程序又一次引用了缺省值。实际上,我们在 ON...GOTO

语句中用连续的逗号,往往是出于节省程序字节的考虑,并非有意转至 0 行,而对应的开关变量 A 值也是非预期的。如果不用缺省值的方法,那么对于侦出非预期值也是有帮助的。当开关变量指向最后一个行号时,ON...GOTO 语句会出现与 GOTO 语句一样的问题。

从上面的几个语句中可以看出,语法书中往往只列举正确的语法格式,而对解释系统的容错性介绍不够,在语法和解释程序之间的对应关系呈单向性;也就是说,动态侦错不能侦出所有的语法错误,通过运行的程序不一定没有错误。

二、表达式侦错

在学习浮点 BASIC 语言时,经常要用到表达式(EXPRESSION)的概念。算法语言书籍中没有为表达式作出严格的定义,一般仅称表达式是:“最终可以得出一个值的运算式”。这样的定义是较为含糊的。APPLESOFT 中规定了 3 种数据类型,即整数、实数和字符串,而没有逻辑型或布尔型量,那么最终得出的值是一个什么类型的量,初学者是很难掌握的。一些表达式的特殊用法难以在语法书籍中找出与之对应的举例。

表达式中经常出现常数、变量、函数;可以进行数值运算、关系运算和布尔运算。表达式的分析和侦错较为复杂,随着内涵的不同,表达式中可能出现的错误种类也很多,有些语法上的错误是不能依靠动态侦错的方法来发现的。下面就表达式中关系运算和布尔运算中不能由动态侦错发现的错误分述如下。

1. 关系运算

浮点 BASIC 语言没有为布尔值定义专门的数据类型,一个表达式的值究竟是整型量、实型量,还是布尔型量,要靠算式本身来确定。如果在表达式中写进了关系运算符,所得出的表达式的值就可能是布尔型量。实际上由于 APPLESOFT 没有明确的布尔型量,那么,布尔量也可以参加数值计算。从这个意义上说,有关系运算符的表达式也可以得出其他类型的量。

在表达式中有三种常用的关系运算符,分别为等量、大于和小于;与之对应的表达符号分别为 =、> 和 <。按照语法定义,这三种关系运算符可以单独使用;或者是不同的关系运算符两两组合使用。除此之外的任何组合形式在算法语言中是没有定义的,皆应视为非法的组合,也就是出错。所有的合法组合如下:

符号组合	定义
=	等于
>	大于
<	小于
>=或=>	大于或等于
<=或=<	小于或等于
<>或><	不等于

在解释程序中,专门在零页中设置了一个单元用来记载表达式中是否存在关系运算符,以及所出现的关系运算符属于哪一种组合。该单元的最低位用来记载表达式中是否出现关系运算符 > 号;(有此关系运

算符时,该位为1,否则为0,以下类似)这一单元的第一位和第二位分别用来记录表达式中是否出现关系运算符=号或<号。在甄别关系运算符的组合是否合法时,解释程序用两次异或的运算来侦错和记录。这种判别较为简捷,可以侦出同一个关系运算符在一次关系运算中出现两次和两次以上的错误。例如:

A>>B 或 C<=<D 等。

但是异或运算不能判别三个不同的关系运算符连写的错误,例如:

A>=<B 或 C=<>D 等。

尽管在编辑程序的过程中不容易出现这样的错误,但是就其容错性来说,它已经丧失了同语言、语法所设计的正确性之间的对应关系。作为一个容错性较好的软件,不应以错误出现的概率高低而做为是否进行侦错的依据。对于三个关系运算符同时连用的情况,要么予以定义,要么按出错处理,这样才更加完善。一般来说,如果在语句中出现了三个不同的关系运算符连接在一起使用,由于解释程序不能侦出错误,程序会继续执行,而将该运算的结果视为真,也就是1。如果这一表达式写在IF...THEN语句中,那么程序流向将是忽视判别条件而执行THEN子句。如果关系运算的各个常量和变量的值均为零,那么类似表达式的值会因得不到正确的比较而取值为0。这些复杂的关系是初学者难以记忆的,因此要注意此类错误。解释关系运算符时,解释程序要利用栈区,这种错误不仅可能导致程序的错误流向和错误的计算结果,而且在特殊的情况下,还可能出现死机的严重情况。

2. 布尔量的数值运算

在苹果机浮点 BASIC 语言中,布尔量可以通过关系运算或布尔运算得到,同时布尔量也可以参与其他的数值运算。例如:

```
10 A=5:B=5:C=(A=B)*10:PRINT C
```

运行该句,可知变量 C 取值为10,这句程序是符合苹果机浮点 BASIC 语言的语法的。

在一般情况下,布尔量经常用来判别条件式的真伪,进而控制程序的流向。但是语法书中并没有详细讨论布尔量的数值运算问题;也没有研究布尔量进行四则运算和其他运算的规则。从语言系统应具备的严格性上来讲,要么容许布尔量参加非布尔运算,使布尔量完全等同于数量0或1;要么不容许布尔量参加非布尔运算,一但在程序中出现此种情况,就按照出错处理。APPLESOFT 采用了前种灵活的方式,但是它在这些问题的解释上还缺乏慎重的思考,在处理稍微复杂的关系运算时,可能出现混乱的情况。下面我们一段程序来说明这个问题。

```
10 A=10:B=5:C$="B":D$="A"  
20 E=(A AND B)+(C$>D$)  
30 PRINT E:END
```

这段程序是符合语法的。不难分析,式中 A AND B 和 C\$>D\$ 两种运算皆为真,两布尔量相加后,为变量 E 赋值为2。

如果去掉20语句中的括号,该句程序变为:

```
20 E=A AND B+C$>D$
```

依照该句中运算符的优先级进行分析,可知三个运算符的优先顺序为+,>,AND。按这一顺序,在执行这句程序时,首先要进行加法运算,不难看出,变量 B 与字符串 C\$ 分别属于两种不同类型的数据,它们之间不能运算,这句程序有语法上的错误。

运行改后的程序,APPLESOFT 没有提示出错,而将变量 E 赋值为1。

严格来说,这种情况已经超出了容错性的范畴,是解释程序的失误。由于 APPLESOFT 对各种数据类型没有明确的区分,许多语法书中又介绍了类似的技巧,使得一些初学者愿意利用上面的方法编程。但是这种灵活性必须有侦错来保证,否则是容易出错的。

在上面的例子中,如果将第20语句作如下改动

```
20 E=C$>D$+A AND B
```

再运行这段程序便会得到数据类型不匹配的错误提示。为什么前者的错误没有被侦出,仅仅交换了顺序便能侦出错误?这些问题是较为复杂的,要借助汇编语言和汇编语言跟踪才便于解释清楚。由于篇幅和主题关系,本文对此不加详述。仅此可以看出 APPLESOFT 在表达式的解释上,既存在着比较灵活的方面;也存在着容易发生混乱的方面。很多人喜欢追求程序的技巧和简捷,经常把各种数据类型混合在同一个表达式中,这是要十分谨慎的。实际上,程序语言严谨是更好的编程指导思想,这一点对初学者尤为重要。

三、命令保留字的容错问题

苹果机浮点 BASIC 语言有99个保留字,它们已经作为64个命令、25个函数和运算符的专用字。语法书中已经强调,这些保留字不应作为变量名称和夹入变量名中使用。由于内存容量的限制,解释程序不能甄别此类错误,因此解释程序的侦错与语法书中定义的正误是不对应的,这种情况更不能用动态侦错。

上述错误的识别只能靠人来进行,而初学者又很难将近百个保留字一下子全记住。因此系统解释程序仍然在一些危险的命令中加入简单的安全措施,如:

```
200 NEW A=50
```

当程序执行到这一句时,解释程序会因 NEW 命令后没有“:”或或语句行结束的标志而不执行 NEW 命令。这种作法实际上是将 NEW 后面的语句分隔符与命令合在一起解释的,在核对命令时必须命令本身和命令之后的分隔符同时正确,那么此保留字才作为命令执行。事先对保留字代码之后的分隔符侦错的命令有7条,它们是:

```
CLEAR,      CONT,  
END,        NEW,  
POP,        RETURN,  
STOP
```

这些命令或者写在语句的最后,或者有“:”跟随其后才能被执行。

另一些无参数命令的危险性要小一些,例如:

10 HOME A=20

我们可以将这一个语句行看成是丢掉了分隔符“:”号,或是编程者误将 HOME A 当作了变量名称,计算机在运行这一语句时,一旦查到 HOME 命令之后,便先将字符屏幕清除,再往下执行时,才会发现语法错误。这样的无参数命令有16个。如果详细分析起来,16个命令破坏程序的危险性也不是相同的。例如 GR, HGR, HGR2等命令,在程序较长或数组较大时,如果误用这类命令,在显示区内的程序和数组会被消掉。因此,对这样的保留字也必须正确使用。

四、结束语

为一种语言设计良好的容错环境是很重要的工作,同时也是很复杂的工作。在不影响程序正确运行的前提下,应尽可能地扩大容错范围,但是由于采用了容错技术,侦错的范围会有所扩大,一些意想不到的错误会影响程序的正确性。容错技术的采用也不应该增加附加的语法,或使语法变得难于理解。从这些意义上讲,容错和侦错是对立和统一的两个方面,不能单纯强调一个方面而忽视另一个方面。

本文虽然列举了一些苹果机浮点 BASIC 语言中存在的一些问题,即容错性是否适当和侦错功能是否应加强的例子,但是作者仍然给苹果机的解释程序以很高的评价。这一系统软件为我们编制小规模的系统软件提供了经验。算法语言是本世纪形成的一门知识,这种知识目前仍在不断地深化和完善过程中,为了提高计算机知识的认识水平,深入地剖析一些系统软件是很重要的工作。尽管苹果机的内存容量较小,但是作为一个系统它是很健全的,小的系统便于分析,而从中得到的结论可以推广到大的系统中去。

对于广大的非专业人员来说,发现解释程序等系统软件中的不完善之处也许是很困难的,但是非专业人员适当地了解一些系统软件的设计思想,对于提高计算机的应用水平是有很大大好处的。

Apple 内存校对发声程序

浙江绍兴稽山中学(312000) 连勤

机器语言程序由于占用内存小,运行速度快,深受计算机爱好者的青睐。但由于十六进制数的16个字符在键盘上分布较分散,输入时很容易出错。常用的检查方法是在监控中用地址1.地址2 \checkmark 的方式显示内存值,然后看一眼屏幕,再看一眼原稿,这样检查,眼睛很吃力,也容易看错。我根据 Apple 讲话的原理,编了一个发音程序,使得计算机在显示内存值的同时,读出内存值,这样在核对机器语言时,只要眼看原稿,耳听计算

机就行了,即省力又准确。

初次使用本程序时,按下列步骤操作:

(1)取一台音质较好的录音机和一盒磁带,录下0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F、这16个音。注意:每个音之间有1~2秒的间歇,0音之前和F音之后也要留几秒钟的间隙。然后取一根电缆,一端插录音机的耳机(EAR)插孔或喇叭(EXT)插孔,另一端插 Apple 的录音输入插孔。

(2)分别在 BASIC 和监控状态下,正确无误地输入程序一和程序二。

(3)把磁带倒至开头,有音调旋钮的话,应开大高音,关小低音,把音量开关放在适中的位置。

(4)运行程序一,打开录音机,按一下计算机的任意键,即开始把声音信号输入计算机,同时计算机的喇叭也发出相应的音,如果计算机声音不正常,可调节录音机的音量开关。发完 F 音后,再等几秒钟,即可关掉录音机。

(5)等待大约4分钟后,计算机就依次发出0~F的每一个音,供你审听,如果不满意,按 N 键可退出程序一。在审听时,如果在某个音的位置连着发几个音,说明录音时,这几个音间的间歇太短,如果发的音和屏幕显示的数不符,说明环境噪音太大。出现这些情况,均应按步骤一重新操作。

(6)如果审听满意,则再等3分钟后,屏幕提示,让你插入磁盘,计算机会自动把程序二和刚生成的声音数据连接成一个以 TALK 为文件名的二进制文件存盘。因为文件较长,磁盘应有足够的空间。

经过以上步骤,以后需校对内存时,只要运行 TALK 文件就行了。TALK 程序从 \$1000 开始存放,故使用时,应先把待校对的机器语言程序用监控 M 命令搬到 TALK 以外的地方,如 \$1000 以前,然后 BRUN TALK \checkmark ,按屏幕提示输入起始地址、结束地址(4位16进制数表示),计算机就开始一边显示、一边读出内存值。如果需要修改内存单元的话,可按任意键退出 TALK 程序,待修改完以后,再用 1000G 运行之即可。

程序二的 \$FB2~\$FFF 是录音程序,用于把录音机放出的模拟信号变成数字信号存入内存,数据存放的起始地址低位由 \$FB3、高位由 \$FB8 的值决定,结束地址高位存于 \$FF9 内。\$1000~\$1076 的功能是输入并显示待校对内存的起始地址,结束地址,起始地址高位存 \$3D,低位存 \$3C,结束地址高位存 \$3F,低位存 \$3E。\$1077~\$1094 是主控程序,用于显示内存地址、内存值,并转发音程序,读出内存的值。\$10AD~\$10C8 的功能是把累加器 A 中的值分离成二个十六进制数,通过查表找到某个音的发音数据存放的起始地址和结束地址,\$10E5~\$111A 是发音子程序,\$10E8、\$10E7 分别存发音数据起始地址的高位和低位,\$110E、\$1116 分别存结束地址的高位和低位。

为了帮助使用者迅速正确地找到每个音的有关发声数据在内存中的存放地址,我又编了程序一,它的功

能是自动找出发每个音的起始、结束地址，填入 \$111B~\$115A 内，并把程序二和声音数据合并后存盘，形成一个可直接运行的 TALK 文件。

```

7 ONERR GOTO 80
10 HOME
15 PRINT "PLEASE SWITCH ON TAPE RECORDER THEN
    HIT ANY KEY TO BEGIN"
25 IF PEEK (49152) < 128 THEN 25
30 CALL 4018
35 PRINT "RECORDING DONE, PLEASE WAIT FOR 4 MIN-
    UTES"
40 S=36864:F1=260
45 DIM A(17),B(17)
50 FOR I=8194 TO S
55 IF PEEK(I)=0 THEN GOSUB 250:GOTO 75
60 FOR J=I+1 TO S
65 IF PEEK(J)< > 0 THEN F1=F1+1: NEXT
70 I=J-1:W=I-F1:F0 = 0
75 NEXT I
80 A $ = "0123456789ABCDEF"
85 FOR I=2 TO 17
90 PRINT "THE SOUND IS", MID $ (A $, I-1, 1)
95 C = A(I): GOSUB 200: POKE 4328, A: POKE 4327, B
100 C = B (I): GOSUB 200: POKE 4366, A : POKE 4374, B
105 CALL 4325
110 INPUT "IS IT RIGHT?"; B $ : IF B $ = "N" THEN
    END
115 NEXT
117 PRINT "PLEASE WAIT FOR 3 MINUTES"
120 DB = 4443: B(1) = DB - 1
125 FOR I = 2 TO 17
130 FOR J = A(I) TO B(I)
135 POKE DB, PEEK (J): DB = DB + 1
140 NEXT
145 C = A(I): A(I) = B(I-1) + 1: B(I) = A(I) + B(I)
    - C
150 NEXT
155 DD = 4443 - 64
160 FOR I = 2 TO 17
165 C = A(I): GOSUB 200: POKE DD, A: DD = DD + 1:
    POKE DD, B: DD = DD + 1
170 C = B(I): GOSUB 200: POKE DD, A: DD = DD + 1:
    POKE DD, B: DD = DD + 1
175 NEXT
178 X = B(17) - 4096 + 1
180 S $ = "BSAVE TALK, A $ 1000,"
185 PRINT S $ ; "L"; X
190 PRINT CHR $ (4); S $ ; "L"; X
195 END
200 A = INT (C / 256): B = C - 256 * A: RETURN
250 FOR J = I + 1 TO S
255 IF PEEK (J) = 0 THEN F0 = F0 + 1: NEXT
260 I = J - 1
265 IF F0 < 64 OR F1 < 260 THEN F1 = F1 + F0 + 2:

```

```

RETURN
270 N = N + 1: A(N) = W
275 M = M + 1: B(M) = I - 1 - F0
280 F1 = 0: RETURN

0FB2-   A2  00  8E  C4  0F  A9
0FB8-   20  8D  C5  0F  AD  60  C0  29
0FC0-   80  85  FF  EE  E6  34  D0  03
0FC8-   20  DC  0F  A5  FF  C5  FE  F0
0FD0-   EB  85  FE  8D  30  C0  20  DC
0FD8-   0F  4C  BC  0F  EE  C4  0F  D0
0FE0-   08  EE  C5  0F  A0  02  88  D0
0FE8-   FD  AD  C4  0F  8D  F6  0F  AD
0FF0-   C5  0F  8D  F7  0F  8E  E6  34
0FF8-   C9  90  F0  01  60  68  68  60
1000-   20  58  FC  A2  00  BD  95  10
1008-   20  F0  FD  E8  E0  0C  90  F5
1010-   A2  02  20  0C  FD  20  F0  FD
1018-   20  51  10  0A  0A  0A  0A  85
1020-   F9  20  0C  FD  20  F0  FD  20
1028-   51  10  29  0F  05  F9  85  3D
1030-   CE  2F  10  CA  D0  DC  A9  A1
1038-   CD  06  10  F0  2C  8D  06  10
1040-   20  8E  FD  EE  2F  10  EE  2F
1048-   10  EE  2F  10  EE  2F  10  D0
1050-   B2  49  B0  C9  0A  90  11  69
1058-   88  C9  FA  B0  0B  BA  E8  E8
1060-   9A  08  20  8E  FD  28  90  01
1068-   60  A9  95  8D  06  10  A9  3D
1070-   8D  2F  10  90  8E  D0  06  A5
1078-   3C  29  07  D0  03  20  92  FD
1080-   A9  A0  20  ED  FD  B1  3C  48
1088-   20  DA  FD  68  4C  AD  10  20
1090-   BA  FC  90  E3  60  D3  F4  E1
1098-   F2  F4  A0  C1  E4  E4  BA  A0
10A0-   A4  C5  EE  E4  A0  A0  A0  C1
10A8-   E4  E4  BA  A0  A4  48  29  F0
10B0-   4A  4A  20  C9  10  68  29  0F
10B8-   0A  0A  20  C9  10  AD  00  C0
10C0-   10  04  AD  10  C0  60  4C  8F
10C8-   10  A8  BE  1B  11  8E  E8  10
10D0-   C8  BE  1B  11  8E  E7  10  C8
10D8-   BE  1B  11  8E  0E  11  C8  BE
10E0-   1B  11  8E  16  11  18  AE  4D
10E8-   00  86  FA  A0  04  88  D0  FD
10F0-   EA  CA  D0  F7  A6  FA  F0  08
10F8-   8D  30  C0  A0  06  88  D0  FD
1100-   EE  E7  10  B0  0D  D0  DF  EE
1108-   E8  10  AD  E8  10  C9  00  4C
1110-   E6  10  AD  E7  10  C9  00  38
1118-   D0  CC  60

```

ProDOS 磁盘操作 系统入门(续)

廖 凯

4. 建立一个主菜单

下面的程序可以按照你需要的格式在屏幕上显示文字,它在屏幕四周产生一个边,这在显示菜单时很有用,你可以修改此程序以获得所期望的结果。

```
1 REM Screen output for a main menu
5 HOME
10 LL$="press ESC to Back Out"
80 GOSUB 900
89 REM MAIN MENU
90 L$="COMPUTE ASSET DEPRECIATION":V%
  =4:GOSUB 1130
100 L$="1. Straight Line Depreciation":
  H%=10:V%=10:GOSUB 1140
110 L$="2. Double Declining Balance":V%
  =12:GOSUB 1140
120 L$="3. Accelerated Method":V%=14:
  GOSUB 1140
130 GOSUB 1200
140 IF C%>48 AND C%<52 THEN 160
150 GOTO 140
160 GOSUB 910
200 REM DESCRIPTION OF PRODUCT
210 L$="PRODUCT DESCRIPTION":V%=4:GOSUB 1130
220 L$="Product Description":H%=10:V%=10:GOSUB
  1140
230 L$="Product cost":V%=12:GOSUB 1140
240 L$="Salvage value $":V%=16:GOSUB 1140
250 V%=22:L$=LL$:GOSUB 1140
260 GOSUB 1200:IF C%=27 THEN GOSUB 910:GOTO 90
890 END
899 REM BORDER ROUTINE
900 INVERSE:HOME:NORMAL
910 POKE 32,2:POKE 33,76:POKE 34,1:POKE 35,23:
  HOME:TEXT:RETURN
999 REM POSITION ROUTINE
1000 POKE 1403,H%:VTAB V%
1010 RETURN
1130 H%=INT(40-(LEN(L$))/2)
1140 GOSUB 1000:PRINT L$:RETURN
1200 REM GET A KEYBOARD COMMAND
1210 C%=PEEK(-16384):IF C%<128 THEN 1210
1220 C%=C%-128:POKE 49168,0
1230 IF PEEK(-18287)>127 THEN C%=C%+128
1240 IF PEEK(-16286)>127 THEN C%=C%+128
1250 RETURN
```

行号80将程序转到行号900的建边子程序,在屏幕四周建立一个反白的边,行号90至120用行号1130和行号1140的子程序,在屏幕上定位文字,建立主菜单,行号130转到行号1200键盘输入子程序等候用户输入。行号140检查用户是否按下1,2或3,如果符合条件,程序执行行号910,清除屏幕准备建立子菜单,在适当的子程序被执行后,子菜单被显示并等候用户输入。按 Esc 键即返回主菜单。

5. 清除屏幕

这是一个很短的子程序,它使用 POKE 屏幕指令清除屏幕而得到一种移离的效果,你可以结合这子程序来清除所有或部分屏幕。

```
10 INVERSE:HOME
20 FOR I=20 TO 0 STEP -2:POKE 32,I:POKE 33,80-I*
  2:POKE 34,I/2:POKE 35,24-I/2:NORMAL:HOME:
  NEXT
30 END
```

行号20经 FOR 循环设置 STEP-2为递减移动,这将使移动宽度变为两个字符宽度。POKE 33, 80设置屏幕宽度为整个80列,它可以用算式-I*2来改变。

四、连接打印机

有效利用打印机是程序设计的一个重要部分。连接打印机的接口有两种:并行接口和串行接口。一个并行接口通过8根线一次发送8位资料。一个串行接口经1根线一次发送1位资料。串行卡还可以用于与其它设备通信。

1. 有关指令

ON GOSUB 根据表达式的值,选择行号,使计算机转向该行号执行

PR# 发送输出到一个槽口或程序

2. 使用打印机

要发送输出到一个打印机,请用 PR# 命令,数字是用于指出打印机接口卡所在的槽口。

```
5 D$=CHR$(4)
10 PRINT CHR$(21)
20 PRINT D$;"PR#1"
30 PRINT "This is a test to send output to the printer"
40 PRINT D$;"PR#3"
50 END
```

行号10关闭80列卡。打印机可用 PR# 命令来选定或消除。在打印机与80列卡一起使用时一定要特别小心。建议在选择其它槽口之前先关闭80列卡。

3. 字符输出

大部分打印机可以打印多种格式和字符。以下程序是为 APPLE DOT MATRIX 和 IMAGE WRITER 打印机而设计的。APPLE DOT MATRIX 打印机使用一个并行接口。IMAGE WRITER 打印机使用 Super Serial Card。

```
100 REM Demo of APPLE Printer family
110 D$=CHR$(4)
120 V%=6
130 PRINT CHR$(21)
```

```

140 PRINT D$;"PR #1"
150 PRINT CHR$(9)"80N":REM send 80 columns to the
    printer
160 PRINT CHR$(27);CHR$(60):REM bidirectional
170 FOR I=1 TO 10
180 ON I GOSUB 1000,1010,1020,1030,1040,1050,1060,
    1070,1080,1090
190 VTAB(V%+2)
200 ON I GOSUB 1100,1110,1120,1130,1140,1150,1160,
    1160,1170,1180,1190
210 NEXT I
220 PRINT D$;"PR #3"
230 END
1000 L$="Here is a Demonstration":RETURN
1010 L$="of the":RETURN
1020 L$="APPLE DOT MATRIX":RETURN
1030 L$="and":RETURN
1040 L$="IMAGEWRITER PRINTERS":RETURN
1050 L$="as you can see":RETURN
1060 L$="we can control":RETURN
1070 L$="the size and":RETURN
1080 L$="overall makeup of":RETURN
1090 L$="the output":RETURN
1100 GOSUB 1280:PRINT SPC(78);L$:RETURN
1110 GOSUB 1270:PRINT SPC(46);L$:RETURN
1120 GOSUB 1220;GOSUB 1200;GOSUB 1310;
    PRINT TAB(12);L$:RETURN
1130 GOSUB 1230;GOSUB 1240:PRINT SPC(68);L
    $;RETURN
1140 GOSUB 1220;GOSUB 1310:PRINT TAB(10);L
    $;RETURN
1150 GOSUB 1230;GOSUB 1210;GOSUB 1300;
    PRINT SPC(75);L$:RETURN
1160 GOSUB 1240:PRINT SPC(64);L$:RETURN
1170 GOSUB 1310:PRINT SPC(31);L$:RETURN
1180 GOSUB 1260:PRINT SPC(48);L$:RETURN
1190 GOSUB 1290:PRINT SPC(37);L$:RETURN
1200 PRINT CHR$(27);CHR$(33):RETURN;REM
    start BOLD
1210 PRINT CHR$(27);CHR$(34):RETURN;
    REM BOLD off
1220 PRINT CHR$(14):RETURN;REM HEADLINE on
1230 PRINT CHR$(15):RETURN;REM HEADLINE off
1240 PRINT CHR$(27);CHR$(81):RETURN;
    REM ULTRA
1250 PRINT CHR$(27);CHR$(113):RETURN
    REM SEMI
1260 PRINT CHR$(27);CHR$(101):RETURN
    REM CONDENSED
1270 PRINT CHR$(27);CHR$(69):RETURN
    REM ELITE
1280 PRINT CHR$(27);CHR$(80):RETURN
    REM PROP ELITE
1290 PRINT CHR$(27);CHR$(78):RETURN
    REM PICA

```

```

1300 PRINT CHR$(27);CHR$(112):RETURN
    REM PROP PICA
1310 PRINT CHR$(27);CHR$(110):RETURN;
    REM EXTENDED

```

此程序内的大部分语句主要与打印机的格式有关。此程序运行打印的结果是10行不同字体的文字。

行号130关闭80列卡。行号140发送输出到槽口1的打印机。当发送输出到外设时，首先检查80列卡是否被启动。在80列卡启动时，发出输出到打印机会发生无法预料的结果。行号150设置打印机为80列输出，行号160设置双向打印，行号1200至1300是设置不同的字体。实际打印发生在行号1100至1190，在那儿设置间距，打印L\$，我们也可以使用POKE 36,X(X是分隔符位置)代替TAB和SPC语句，作为在页面上定位字符的另一种方法。行号220脱离打印机，设置屏幕为80列显示。

如果用户使用的是其它型号的打印机，就请参考打印机使用手册。

<未完待续>

(上接第33页)

显示2005

键入20

显示2006

键入00

显示2007

键入完毕后按下O键，便退出H命令。

②L命令

作用：以16进制数的形式显示存储单元之中的内容。

格式：L××××× ××××为四位地址，如列出从2000H开始的存储器单元之中的内容

键入L2000

显示2000 ×× ××为两位十六进制的数。

继续显示只要按下键，若不想再显示，则按下“O”键即可退出。

③E命令

作用：运行键入的程序

格式：E×××××

例如：运行刚键入的由2000开始的程序

操作：E2000

想停止执行程序按下复位键即可。

四、操作步骤：

1. 开机，数码管显示“8031—A”，若显示的不是该字型可按下复位键。

2. 按下“W”键以后数码管不显示任何数字，此时表示监控程序已作好接收各种命令的准备。

3. 根据需要即可键入以上命令。

有需要本开发系统程序清单的可寄复制邮资费3元至本文作者罗明宽、车金相，地址：北京宣武区南横街西街宣武青少年科技馆 邮编：100052。

·语言讲座·

6502机器语言程序设计讲座

南京大学大气科学系(210008) 朱国江

苹果机和中华学习机的三大系统软件,即监控程序、浮点 BASIC 解释程序 Applesoft 和磁盘操作系统 Apple DOS,都是用6502机器语言编写的。要想知道这些系统程序怎样工作,如何有效地利用它们提供的丰富的功能以提高编程水平,就得学好6502机器语言。

在16位机和32位机日趋普及的今天,如何让相当大一部分现有的8位机资源不致闲置,引起人们的关注。除了在中小学电脑教育方面暂时还将保持一席之地外,许多有识之士呼吁将8位机改制成各种专用机,使它们在社会和家庭的各方面继续发挥作用。要做好这一点,掌握6502机器语言也是一个重要条件。

我们根据许多读者来信表达的愿望,特邀请南京大学大气科学系朱国江副教授开设本讲座。朱老师在中华学习机的编程及应用方面有多种专著问世,深受读者欢迎。我们期望这个讲座对于想深入一层学习计算机的广大初学者有所帮助。随着讲座的开展,我们还将组织系统程序剖析和6502编辑、汇编程序使用介绍的稿件,欢迎广大读者、作者投稿支持。 编者

第一章 机器语言程序设计绪论

一、指令、指令系统

众所周知,不论计算机使用何种语言编写的程序,其本质都是按程序存储器中的内容依次动作的。这就是说,不论何种语言写成的程序,它最终都要转化为计算机程序区中的一个一个的8位二进制代码,计算机只能按这种转换好的代码动作。我们称这种最基本的代码为指令,而这种代码的集合称为该种型号计算机的指令系统。

不同型号的计算机具有不同的指令系统。中央处理单元 CPU (central processing unit) 中具有指令译码器,显然送入某种计算机的指令必须是该计算机 CPU 中的指令译码器所约定的。所以一台计算机使用什么指令系统,取决于它采用何种型号的 CPU 芯片。APPLE— I、紫金— I 及中华学习机是采用以6502微处理器为 CPU 的计算机,所以它们的指令系统是6502的。

例如,在指令提取运算时,当6502微处理器接到一个8位二进制代码“11101000”输入,意指:“将 X 暂存器内的值加1”。同样地,代码“10101001”则是:“将下一个内存单元所存之值移至累加器 A 中”。

微处理机仅“认识”二进制代码的指令或资料,它“看不懂”字组、八进制、十进制或是十六进制等数字。

二、程序及其执行过程

程序是由一系列的指令所组成,以使计算机执行特定的任务。计算机的程序不仅只包含指令,它也含有用以完成某任务的指令所需要用到的资料及存储器地址等。例如,微处理机要执行一个加法,则一定要有两个待相加的数,且有一地方存放相加后的结果。所以,这个计算机程序除了具有执行运算的功能外,还应决定此两个相加数的来源,并且决定加完后结果存放在何处。

例如,将内存单元 \$ 60 与 \$ 61 内的值相加,结果存于 \$ 62 中,其6502机器语言程序为:

```
10100101
01100000
01100101
01100001
10000101
01100010
```

如果将上述程序,送入一个以6502为 CPU 的微电脑中,则此微电脑就能直接执行它。

程序执行的过程,就是微机工作的过程。由于程序是由指令序列组成,因此,执行程序的过程就是执行指令序列的过程,也就是不断地取指令、执行指令的过程。这个过程可简单地分为两个阶段。开始时,微处理机进入取指阶段。在取指阶段,从存储器取出指令,送指令寄存器,经译码器译码后,进入第二阶段,即执行指令阶段。在执行指令阶段,微处理机执行指令所规定的操作,指令不同,执行的操作也不一样。而在取指阶段,指令不同,取指令的操作却是相同的。总之,微机工作的过程,就是周而复始地取指令、执行指令的过程。

三、机器语言

用二进制代码编写的程序,称为机器语言程序。在机器语言中,指令是用二进制数字表示的。

显然,以二进制代码书写的程序(又称目的程序),阅读和理解都很困难,诸如:

- 二进制数目看起来都差不多相似,尤其在大量阅读数据以后,令人眼花缭乱,难以区别。

- 二进制码必须一个接一个地输入,稍不留心就会出错,并且输入速度十分缓慢。

- 无法了解计算机所要完成何种任务,执行的是何种操作。

- 程序冗长,编写麻烦,乏味,容易混淆,很难发现错误,纠正也极不方便。

- 很难记忆,因为人们无法像电脑处理二进制数那样容易、直接识别。

为了克服用二进制代码编制和书写程序的困难,人们采用十六进制,一方面因其简洁;另一方面则是它与二进制数有简单的换算关系。

例如,上面所举的两数相加的6502程序变为:A5

60 65 61 85 62

可以看出,一位十六进制数用来表示四位二进制数,如A代表1010,6代表0110,0代表0000等。这样,程序简明多了,不像二进制数那样麻烦,也容易找出错误。下表是几个基本的二进制、十进制和十六进制表的关系:

二进制	十进制	十六进制
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F
1111111111111111	65535	FFFF

前面谈到,机器语言是计算机能够直接识别的唯一语言,微处理器也只了解二进制指令代码,那么,现在改用十六进制代码,是否可行?唯一的办法是将十六进制数转换成二进制数。但这种转换是一种令人厌烦的重复劳动,人们极易疲劳。幸好,计算机却非常适合做这项工作,它绝对不会疲劳、厌烦、也不会造成愚蠢的错误。将十六进制数转换成二进制数,是许多微电脑特有的本领。

然而,用十六进制代码代替二进制代码后,并没有解决所有机器语言设计上的问题。因为用十六进制代码表示的机器语言程序,人们仍然难以阅读和了解,例如:我们区别不出何者是指令、资料或是地址;也无法知道程序在做什么,到那里去;A5是代表什么?强记这些码,岂不是倒胃口吗?再说,不同类型的微处理机,这些码的含意也可能完全不同。因此,上述用十六进制代码表示的机器语言程序,虽比用二进制代码表示的机器语言程序有所改善,尚需要进一步改进。

四、汇编语言、汇编程序

如果把每一个指令码指定一个名称,那么机器语言程序设计就可以获得明显的改进。指令码的名称叫助记符,这是人们在不断实践过程中的一个创造。

助记符可以反映指令的功能和主要特征,用以为指令的动作做某些方面的陈述、注释。用助记符表示的

计算机语言叫做符号汇编语言,简称汇编语言(Assembly Language)。它是一种与计算机指令密切相关的计算机语言,直接用计算机指令的汇编符号来编写程序。

例如,前面的加法程序,用助记符表示则为:

```
LDA $60
ADC $61
STA $62
```

这个程序虽然还不够明显,但至少其中某些部分已变得较易了解了。用“ADC”比“65”有了相当的改进。如第一条指令 A5的助记符是 LDA,它是英文 Load A from memory(装入累加器 A 中的内容与存储器一致)的缩写。A 是 6502 中的一个寄存器。指令 LDA \$60 的意思是取 \$60 存储单元中的数据放入寄存器 A。在 6502 的指令系统中,采用 \$ 号来表示后面的数字是十六进制数编址的存储单元。第二条指令 65 的助记符是 ADC,它是英文 Add memory to A with carry(取存储器中的内容连同进位标志位与寄存器 A 中的内容做加法,结果送入 A 中保存)。指令 ADC \$61 的意思是取 \$61 中的数据与进位标志 C 及累加器 A 中数据三者相加,并存入 A 中。第三条指令 85 的助记符是 STA,它是 store A in memory(把 A 中的内容存入存储器中)的缩写。指令 STA \$62 意即把寄存器 A 的数据存入存储器 \$62 单元中。

由此可知,上述简单程序中,我们可以知道那个部份是指令,那个部份是数据,又那个部份是地址了。

汇编语言比机器语言前进了一大步,它的优点在于:

- 用汇编符号表示指令,比较直观,容易记忆。
- 用标号代替地址,写程序时可不考虑转移地址的具体数值。
- 程序修改较为方便,不会因修改一条指令而牵动整体。
- 允许对源程序进行注释,便于阅读、理解。
- 允许操作数为简单的表达式或标号,编写程序方便。

前面已提到,计算机只能理解和执行二进制数代码的机器语言程序,因此,需要有专门的汇编程序(Assembler),将汇编语言编写的程序翻译成二进制的程序代码。常用的 6502 的汇编程序有 EDITOR/ASSEMBLER (EDASM) 和 LISA 等。

五、汇编语言的语句格式

下面以 EDASM 采用的符号来说明汇编语言的语句格式。

汇编语言编写的程序由一系列语句组成,每个语句单独占一行。程序由两类语句构成:一类是指令语句(即由指令构成的语句);另一类是伪指令语句(在程序执行过程中不产生操作)它们都必须按语句格式的规定来书写。

一个语句由四个字段组成,或者说分四个区:标号区、操作码区、操作数区和注释区。语句的一般格式为:

```
标号 操作码 操作数 注释
```


Label Opcode Operand Comment
 字段之间用空格作为分隔符,同一字段内不使用空格,而操作数与注释之间用分号隔开。

例如,有一段汇编语言程序:

标号	操作码	操作数	注释
START	LDA	\$ 8000	;将8000单元的内容送入累加器 A
	STA	\$ 8100	;将 A 内容送入 8100 单元

•标号

标号在语句中代表以字符串方式表达存储单元的地址。这种地址不是用十六进制数表示,而是用符号、文字表示的,称为符号地址。当某行指令之前冠有标号时,这个标号就代表该指令存储在程序存储区中的地址值。因此在转换成目标程序时,标号就被具体的存储单元地址所取代了,程序员不必算出程序中存储单元的具体地址值,这就使得程序单元容易查找和修改,也给编制和调试程序带来方便。如果给每个用标号的地址加一个常数,则能将整个程序段浮动或插入其它程序中,也可用于内存搬家,即用移动标号来修改程序。使用标号时应注意几点:

1. 标号必须从一行开头处开始,它必须是唯一的,即在一个源程序中一个标号只能定义一次。即使一个源程序包含多个文件,各文件间也必须遵守这一规定。
2. 标号必须以英文字母开头,以8个字符为限,以空格符作为结束。
3. 不能使用操作码助记符作为标号,也不得使用 A、X、Y、P、S 这几个寄存器的名字作标号。
4. 标号不是每个语句都必需的,可根据程序的需要选用。一般对于程序入口地址、程序转移地址、计数器单元、常数单元等都可使用标号。

例如:一个求和程序

	LDA	# 0	;和=0(初始化),#号表示后面是数,不是地址。
	TAX		;计数器(这里是 X 寄存器)置 0
SUM	CLC		;清进位标志
	ADC	\$ 6000, X	;和=和+数据
	INX		;计数器加 1
	CPX	# 10	;比较计数值是否等于 10,即 10 个数是否加完
	BNE	SUM	;若不等,则未加完,继续求和
	STA	\$ 0340	;加完,送结果
	RTS		;结束

本程序是对从 \$ 6000 单元开始的十个数求累加和,最后结果存入 \$ 0340 单元,并假设和数不大于一个 8 位二进制数,这样可以不考虑进位。程序中 SUM 就是一个标号,当程序执行到 BNE SUM 时,若转移条件满足(上条指令执行后结果不为零, Z 标志=0),就转到标号为 SUM 相对应的存储单元,即执行 CLC 指令。

•操作码

6502 微处理器有 56 种基本指令,一条指令包括一个字节(由八位二进制代码组成)至三个字节,不论单

字节指令或多字节指令,都包含两个部分;操作码部分和操作数部分。

操作码指明操作性质,它规定某条指令进行什么样的操作。不论单字节指令或是多字节指令,操作码都是放在指令的第一个字节中。操作码由指令助记符或伪指令构成,前者一律三个字母表示,后者由二个以上字母表示,例如:

ORG	\$ 1000	;程序起始地址是 \$ 1000
LDA	# \$ 40	;取立即数 40 到 A
STA	\$ 08	;再存到 \$ 08 单元中
BRK		;中断

上述小程序中,ORG、LDA、STA、BRK 都是操作码,其中 ORG 是伪指令,其余均为指令助记符。显然,操作码是每一个语句的必备字段。

•操作数

操作数字段中放的是操作码所要操作的数据,或者是被操作的数字所存储的单元的地址。操作数和操作码一起确定指令要执行的内容。

应该注意理解操作数的含意,它不仅仅是一个“数字”的概念,而是表示更为丰富的内含。例如:

LDA	# \$ 40;	将数字 64 装入累加器 A 中。
CMP	\$ 03F4;	比较 A 中的数与 \$ 03F4 单元中存储的数哪个大。
LDA	\$ 3F, X;	X 寄存器中的内容加上 \$ 3F 再把该值指示的地址中的内容送入 A。

这里第一条指令的操作数 # \$ 40,是所谓“立即数”,是一个数字(十进制数的 64),不是地址;第二条指令中的操作数是一个地址,真正的操作数存放在 \$ 03F4 单元中;第三条指令中的操作数也是一个地址,不过这个地址必须通过 X 变址寄存器中的内容来变化,真正的操作数是在 (3F+X) 中。由此可见,计算机的指令中,表达操作数的方式是多样的,也很灵活,这种表达不同操作数的方式,就是我们将要介绍的寻址方式。

在汇编语言中,操作数可以使用标号、常数、表达式、字符或字符串,现分别说明如下:

1. 操作数是指定的标号

如前面介绍标号字段时所举的求累加和程序中, BNE SUM 这条指令中的操作数就是以标号 SUM 的形式出现的。又如下列指令:

LOOP1	LDY	# \$ 64
	:	
	JMP	LOOP1

在 JMP LOOP1 这条指令中,操作数是以标号 LOOP1 的形式出现的,它表示程序将转移到标号为 LOOP1 的地址去继续执行指令 LDY # \$ 64。

从这两个简单例子看出,用已有定义的标号作为操作数,在汇编语言中经常使用。此外,使用标号作为操作数,程序员如果使用带有汇编语言编译程序的计算机,就不必算出程序中存储单元的具体地址值。但如果使用手工编译,仍要人工计算存储单元地址。

2. 操作数是常数

所谓常数包括十进制常数、十六进制常数、八进制常数和字符串常数等四种。例如,前面的10个数据累加和程序中,LDA #0,CPX #10中的0和10都是十进制常数,在十进制数前面不加字符;STA \$0340中的0340是十六进制常数,它的前面必须加\$字符;字符串常数用跟在单引导之后的字符表示ASCII码字符,例如 LDA #'A',表示的是把字母A所对应的ASCII值C1,送入累加器A中。当所表示的ASCII码字符不止一个时,仅对第一个字符有效,而在最后一个字符之后应再缀有单引号。如 LDA 'ABC'和 LDA 'A'是等效的。

3. 操作数以表达式出现

这时表达式由运算符+、-、*、/及>、<构成。例如:

```
LDA SECD-1 ;表示(SECD-1)→A
'DA # <10000 ;将10000的高字节→A
STA TICH ;再送入 TICH 单元
LDA # >10000 ;将10000的低字节→A
STA TICL ;再送入 TICL 单元
```

在上面的程序中,十进制数10000转换成十六进制数的2701,高、低字节分别为27和01。

对表达式进行运算的规则,是按自左至右的顺序进行计算。如5+COUNT-2,是先将5加上COUNT对应值的高字节数,然后再减去2。

·注释

注释字段是一个可选字段,并不是每个语句都必须有的。注释只是为了阅读程序及编写文件方便,对指令或程序段所作的功能说明,用以了解程序当前工作或者转向等。注释必须以分号开始。其内容不属程序本身,因而不影响程序执行。

前面曾经提到汇编语句有两类,一类是指令语句,另一类是伪指令语句。这里选择常用伪指令的介绍如下。

伪指令是对汇编程序的一种命令,它在程序中执行一些特殊的操作。例如,它命令汇编程序在汇编时(即翻译汇编语言时),执行一些动作;为程序分配一定的存储区域;给符号赋值;将给定数据放入存储单元;留出存储数据用的存储区等等。伪指令构成的语句同样可以分为标号、操作码、操作数、注释四个字段。把用伪指令编写的程序输入到配有汇编程序的计算机中,汇编程序(编辑/汇编程序)与执行6502指令语言一样,同样执行伪指令所规定的各种操作。

·定义程序起始地址的伪指令 ORG

ORG,即ORIGIN(起点)的缩写,用于规定一段程序或数据的起始地址,它的操作数字段表示的是存储单元地址值。例如:

```
ORG $0300
START PHA
      LSR A
      : :
```

由于使用了ORG的伪指令,上述PHA指令的代码将安排在\$0300单元,同时START标号也与\$0300等量。若在上述程序中加一句ORG *+30,则将当前地址值加30。

·定义符号的伪指令 EQU

EQU是EQUATE(等值)的缩写。此语句用来给标号指定一个值。这个语句中必须有标号,并且不允许此标号再作其它语句的标号使用。例如:

```
DISPLAY EQU $FDF0
: :
JSR DISPLAY
: :
```

这个例子经过汇编则DISPLAY被赋值为\$FDF0这个地址值,而“JSR DISPLAY”便被汇编为“JSR \$FDF0”。

注意在写程序时,EQU伪指令应放在程序的开始处。

·定义字节的伪命令 DFB

DFB是DEFINE BYTE(定义字节)的缩写。DFB的功能是把字节或字节串存入从标号开始的连续单元中。例如:

```
ORG $300
MNEML DFB $00,$11,$22,$33,
        $44,$55,'A','B'
```

伪指令ORG \$300规定了标号MNEML的地址值是\$300,伪指令DFB指定字节串00、11、22、33、44、55以及字符A、B的ASCII码等八个字节,应该顺序地存放在标号MNEML开始(即从\$300开始)的连续单元中。其中第七、八两个地址中放的是字母A、B的ASCII码值。

·定义字的伪指令 DW

DW是DEFINE WORD(定义字)的缩写。此指令定义一个16位二进制数,即两个字节长度的数。其低8位在第一字节,高8位在第二字节。例如:

```
WORD DW $FB1C
```

此时,在WORD单元存入\$1C,在WORD+1单元存入\$FB。

伪指令还有不少,这里不再介绍。

六、汇编语言的特点

计算机的程序设计,从使用二进制数字表示的机器语言,发展到使用符号表示的汇编语言,之后,又发展到了使用接近人类语言的高级语言,是经历了很长时间的。程序设计语言的迅速发展和更新换代,极大地推动了计算机技术的普及和应用。高级语言受到广大青少年和科技工作者的欢迎,是因为对计算机本身的结构不必作详细的了解,而且可以减少程序设计时所下的功夫,同时它面向问题,面向使用者,简单易懂,易于学习和掌握,所有这些对于了解和掌握计算机的程序设计无疑是很有有效的。在学习了高级语言(例如BA

(下转第42页)

初级程序员级水平考试辅导讲座

1992年计算机软件专业初级程序员级

技术资格考试复习自测题

计算机硬件基础知识

北京619信箱(100083) 顾育麒

自测题

从供选择的答案中选出正确答案,把相应的编号写在自我测试题的对应横线上。

1. 将十进制数125.8125转换为二进制数是_____,八进制数是_____,十六进制数是_____。

供选择的答案:

- (1)1011111.1101 (5)175.61 (9)7D.B
(2)1111101.1011 (6)175.64 (10)7D.13
(3)1111101.1101 (7)175.46 (11)D7.D
(4)111101.1101 (8)571.64 (12)7D.D

2. 将二进制数10110010.0011转换为十进制数是_____,八进制数是_____,十六进制数是_____。

供选择的答案:

- (1)178.0625 (5)134.06 (9)B2.3
(2)178.1875 (6)544.14 (10)59.B
(3)176.125 (7)262.14 (11)B2.B
(4)178.375 (8)134.06 (12)32.3

3. 已知某微型计算机的字长是8位,则二进制数-1111111的原码表示是_____,反码表示是_____,补码表示是_____。

供选择的答案:

- (1)10000001 (2)11111111
(3)01111111 (4)10000000

4. 用补码表示的二进制数01110010的真值是_____,用补码表示的二进制数10110101的真值是_____.(真值用十进制数表示)

供选择的答案:

- (1)14 (2)114 (3)-114 (4)-14
(5)-53 (6)-75 (7)75 (8)53

5. 对于二进制数的代码10101101,若把它理解为无符号整数时,其对应的十进制数为_____;若把它理解为补码表示的有符号整数时,其对应的十进制数为_____。

供选择的答案:

- (1)-45 (2)173 (3)-83 (4)45

6. 某计算机采用定点整数格式,字长8位(包含一位符号位),当X采用原码表示时, $[X]_{原}$ 的最大正数值

是_____,最小负数值是_____,当X采用补码表示时, $[X]_{补}$ 的最大正数值是_____,最小负数值是_____.用十进制真值形式填写。

供选择的答案:

- (1)-1 (2)256 (3)255 (4)-127
(5)-128 (6)127 (7)-255 (8)128

7. 已知一逻辑表达式为 $F=A\bar{B}C+A\bar{B}\bar{C}+ABC+A\bar{B}$,化简后可得到表达式_____。

供选择的答案:

- (1)AB (2)B (3)A (4)AC

8. 化简逻辑表达式: $AB+\bar{A}C+BC=$ _____。

供选择的答案:

- (1) $\bar{A}C+BC$ (2) $AB+\bar{A}C$
(3) $AB+BC$ (4)AB

9. 在CPU中,指令寄存器的作用是_____,程序计数器的作用是_____。

供选择的答案:

- (1)用来存放后继指令地址。
(2)保存当前正在执行的一条指令
(3)保存将被存储的下一数据字节的地址。
(4)保存当前CPU所访问的主存单元的地址。

10. 指令系统中采用多种不同寻址方式的主要目的是_____。

供选择的答案:

(1)实现存储程序和程序控制。
(2)缩短指令长度,扩大寻址空间,提高编程的灵活性。

(3)可以直接访问外存储器。

(4)提供扩展操作码的可能性,降低指令译码的难度。

11. 变址寻址方式中,操作数的有效地址等于_____。

供选择的答案:

- (1)内存地址寄存器内容加上形式地址。
(2)数据寄存器内容加上形式地址。
(3)变址寄存器内容加上形式地址。
(4)指令寄存器内容加上形式地址。

12. 在间接寻址方式中,操作数处在_____。

供选择的答案:

- (1)通用寄存器 (2)主存单元
(3)程序计数器 (4)堆栈

13. 为了使微型计算机能正常工作,延长其使用寿命,机房的相对湿度应控制在_____范围内。

供选择的答案:

- (1)大于80%
(2)20%~80%
(3)小于20%

14. 计算机的技术性能指标 MIPS 的意义是_____。

供选择的答案:

- (1)每英寸扫描的线数。
(2)每秒能执行的百万条指令数。
(3)每秒能打印的字符数。

15. 计算机的技术性能指标 MTBF 的意义是_____。

供选择的答案:

- (1)磁带记录密度。
(2)平均无故障工作时间。
(3)机器翻译时间。

16. 半导体存储器在关机或停电后,_____芯片所存储的信息不会丢失,接通电源即可读出使用;而_____芯片,一旦断电,存储的信息全部丢失,接通电源后,其中已没有有用的信息。

供选择的答案:

- (1)RAM (2)ROM

17. 某微型计算机配备320KB/360KB 双面5.25英

寸软盘驱动器,能在这种软盘驱动器上读/写的软盘片有:_____和_____。

供选择的答案:

- (1)720KB 双面3.5英寸软盘片。
(2)160KB/180KB 单面,倍密度5.25英寸软盘片。
(3)320KB/360KB 双面,倍密度5.25英寸软盘片。
(4)1.2MB 双面,高密度5.25英寸软盘片。

18. 某磁带机,读写磁带时的走带速度为45IPS,磁带的记录密度为800BPI,则该磁带机的数据传送速度为_____。

供选择的答案:

- (1)32000BPS (2)36000BPS

19. 下列三种打印输出设备中,_____是击打式打印机;_____、_____是非击打式打印机。

供选择的答案:

- (1)激光印字机。
(2)喷墨印字机。
(3)点阵针式打印机。

20. 计算机科技文章中,英文缩写 CAD 代表_____。

供选择的答案:

- (1)计算机辅助制造
(2)计算机辅助管理
(3)计算机辅助设计
(4)计算机辅助教学

自测题答案

题号	答案	题号	答案
1	(3)(6)(12)	11	(3)
2	(2)(7)(9)	12	(2)
3	(2)(4)(1)	13	(2)
4	(2)(6)	14	(2)
5	(2)(3)	15	(2)
6	(6)(4)(6)(5)	16	(2)(1)
7	(3)	17	(2)(3)
8	(2)	18	(2)
9	(2)(1)	19	(3)(1)(2)
10	(2)	20	(3)

自测题的解答与分析

[第1题]这道题是数制及其转换的自测题。数制间的转换,实质上是基数间的转换。如果两个

有理数相等。则两数的整数部分和小数部分一定分别相等。因此,进行各数制间的转换时,把整数部分和小

数部分分别按它的转换规律进行转换。

十进制整数转换为二进制整数,采用“除2取余”法。它的转换规律是:用2不断地去除要转换的十进制数,若余数为1,则相应位为1;若余数为0,则相应位为0,从低位到高位依次求得相应位的数字符号,直到商为0时为止。最后一次除法得到的余数,相应于最高位的数字,然后按从高位到低位的顺序写出这个转换后的二进制数。

类似地把十进制整转换成八进制整数、十六进制整数,只需要用“除8取余”法,“除16取余”法。

十进制小数转换为二进制小数,采用“乘2取整”法。它的转换规律是:用2反复去乘十进制纯小数的小数部分,每次乘上2之后,所得新数的整数部分为1,则相应位为1;整数部分为0,则相应位为0。从高位向低位依次进行,直到满足精度要求为止。最后一次乘积的整数部分为最低位的数字,然后按从高位到低位的顺序写出转换后的二进制小数。

类似地把十进制小数转换成八进制小数、十六进制小数,只需要用“乘8取整”法、“乘16取整”法。

这道题要求将十进制数125.8125转换为二进制数。

整数部分“除2取余”:

$125 \div 2 = 62$	余数	1	(低位)
$62 \div 2 = 31$	余数	0	
$31 \div 2 = 15$	余数	1	
$15 \div 2 = 7$	余数	1	
$7 \div 2 = 3$	余数	1	
$3 \div 2 = 1$	余数	1	
$1 \div 2 = 0$	余数	1	(高位)

$$(125)_{10} = (1111101)_2$$

小数部分“乘2取整”:

乘2	纯小数部分	整数部分
$0.8125 \times 2 = 1.6250$	0.625	1 (高位)
$0.625 \times 2 = 1.250$	0.25	1
$0.25 \times 2 = 0.5$	0.5	0
$0.5 \times 2 = 1.0$	0.0	1 (低位)

$$(0.8125)_{10} = (0.1101)_2$$

整数部分与小数部分用小数点连起来,得到:

$$(125.8125)_{10} = (1111101.1101)_2$$

在选择填空时,应选择答案(3)

在数制转换的过程中容易发生的错误:

1. 把高位到低位的数字顺序写错。供选择的答案(1),把整数部分高位与低位的数字顺序写反;供选择的答案(2),把小数部分高位与低位的数字顺序写反。这两个答案都是错误的。

2. “除2取余”还没有进行到商为0,就急于结束数制转换过程。供选择的答案(4)比其它三个答案少一个二进制数的高位数字,产生的原因是:计算到商为1时就错误地认为“除2取余”的过程已经结束,因此,在整数部分的高位少一个二进制数字。

这道题的第2个空格是要求将十进制数转换为八

进制数。

整数部分	“除8取余”:	
$125 \div 8 = 15$	余数	5 (低位)
$15 \div 8 = 1$	余数	7
$1 \div 8 = 0$	余数	1 (高位)

小数部分“乘8取整”:

$0.8125 \times 8 = 6.5$	整数部分	6 (高位)
$0.5 \times 8 = 4.0$	整数部分	4 (低位)

$$(125.8125)_{10} = (175.64)_8$$

这道题的第3个空格要求将十进制数转换为十六进制数。

整数部分“除16取余”:

$125 \div 16 = 7$	余数	13 (写成 D)	(低位)
$7 \div 16 = 0$	余数	7	(高位)

小数部分“乘16取整”:

$$0.8125 \times 16 = 13.0 \quad \text{整数部分} 13 \text{ (写成 D)}$$

$$(125.8125)_{10} = (7D.D)_{16}$$

当读者熟练地掌握了数制及其转换的方法以后,可以发现还有更加简单的方法。

十进制数125.8125转换成二进制数1111101.1101以后,可以利用二进制数直接转换成八进制数、十六进制数。

$2^3 = 8$,三位二进制数与一位八进制数相对应。二进制数转换为八进制数的方法:以小数点为界,小数点左边的整数部分自右至左每三位二进制数分为一组,不足三位时左边补0;小数点右边的小数部分自左至右每三位二进制数分为一组,不足三位时右边补0,然后把每一组二进制数用相应的八进制数表示。

二进制数	001	111	101	110	100
	↓	↓	↓	↓	↓
八进制数	1	7	5	6	4

$2^4 = 16$,四位二进制数与一位十六进制数相对应。

二进制数转换为十六进制数的方法:以小数点为界,小数点左边的整数部分自右至左每四位二进制数分为一组,不足四位时左边补0;小数点右边的小数部分自左至右每四位二进制数分为一组,不足四位时右边补0,然后把每一组二进制数用相应的十六进制数表示。

二进制数	0111	1101	1101
	↓	↓	↓
十六进制数	7	D	D

从前面的计算中,还可以看到:十进制数转换成八进制数的计算过程短,转换成十六进制数的计算过程更短。因此可以先将十进制数转换成八进制数,再利用八进制数与二进制数的对应关系转换成二进制数,这样更节约时间。读者可以根据自己的熟练程度,灵活运用以上的几种转换方法。

[第2题]二进制数转换为十进制数的方法有两种:

1. 按权相加法:把每一位的权(2的某次幂)与数位值(0或1)的乘积项相加,所得到的和,就是相应的十进制数。

$$(10110010.0011)_2 = 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times$$

$$\begin{aligned}
& 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + \\
& 0 \times 2^0 + 0 \times 2^{-1} + 0 \times 2^{-2} + 1 \\
& \times 2^{-3} + 1 \times 2^{-4} \\
& = 128 + 32 + 16 + 2 + 0.125 + \\
& 0.0625 \\
& = (178.1875)_{10}
\end{aligned}$$

2. 对应关系法: 二进制数每一位的权与十进制数有一定的对应关系。例如小数点左边第2位的权, 对应的十进制数为2; 第4位的权, 对应的十进制数为8。将二进制数中数字为1的位的权, 所对应的十进制数值相加, 其和就是相应的十进制数。

$$(10110010.0011)_2 = 128 + 32 + 16 + 2 + 0.125 + 0.0625 = (178.1875)_{10}$$

二进制数转换为八进制数、十六进制数的过程如下:

二进制数	010	110	010	001	100
	↓	↓	↓	↓	↓
八进制数	2	6	2	1	4
二进制数	1011	0010	0011		
	↓	↓	↓		
十六进制数	B	2	3		

[第3题] 前面提到的二进制数, 没有提到符号问题, 是一种无符号的二进制数。但是在计算机中运算的数, 是有正负号的。人们规定用“0”表示正数符号, 用“1”表示负数符号, 而且规定一个数的二进制编码的最高位为符号位。这样, 连同符号位在内的若干位二进制

数作为一个数, 称为机器数。它是数在机器中的表示形式。它所代表的数值称为机器数的真值。

为了运算方便, 在机器中负数有三种表示法——原码、反码和补码。采用补码以后, 可以使正、负数的加、减运算简化为单纯的加法运算。

原码表示法:

正数的符号位用“0”表示, 负数的符号位用“1”表示, 数值位保持不变。

$$[-1111111]_{\text{原}} = 11111111$$

反码表示法:

正数的反码与原码相同, 最高位为符号位, 用“0”表示正, 其余位为数值位。

负数的反码表示, 即为它对应的正数按位取反(连同符号位)而形成的。

二进制数

-1111111

↓

↓

对应的正数

01111111

↓

↓

按位取反

10000000

$$[-1111111]_{\text{反}} = 10000000$$

补码表示法:

正数的补码与原码相同, 最高位为符号位, 用“0”表示正, 其余位为数值位。

负数的补码表示, 即为在它的反码的末位加1(简称“求反加1”)

$$[-1111111]_{\text{补}} = 10000001$$

《全国中小学计算机教育资料汇编》征订启事

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、中小学校和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13.5元, 已于1991年8月出版。

(注: 此书邮购需另加15%的邮费)

地址: 北京市万寿路173信箱 电子工业出版社软件部

邮 码: 100036 电 话: 815342

联系人: 谈众安 王 旭



8031单片机最小系统

北京宣武区青少年科技馆(100052) 罗明宽 车金相

我们利用 APPLE I 和 LASER310 研制出一套开发装置—EPROM 仿真器。利用这个仿真器开发出了 8031 单片机的最小系统。

一、EPROM 仿真器的工作原理

下面我们对 EPROM 仿真器的电路(见图1)做一下简单说明:该仿真器有两个系统,其中一个系统是 APPLE 机,该系统可以通过 IC₂、IC₃、IC₄、IC₅、IC₆、IC₇ 等芯片对 2K RAM 6116 进行读写操作,另一个为被开发系统,即 8031 单片机系统,这个系统通过 EPROM 插座 IC₂、IC₃、IC₄、IC₅、IC₆ 等芯片也能对 6116 进行读操作,可见 2K RAM 6116 为两个系统公共存储器。IC₂、IC₃、IC₄、IC₅ 为 74LS157,为二选一芯片,它的作用是对两个系统的地址线进行切换,由 1 脚进行控制,IC₇、IC₈ 为双向数据缓冲器,利用各自的 19 脚的低电平进行数据传送和封堵对方系统的数据传送,各自的 1 脚控制数据传送的方向,IC₆ 为 74LS138 译码器,它的作用是对 2K RAM

6116 提供片选信号和为 IC₇、IC₈、IC₂~IC₅ 提供控制信号。

当在 APPLE 机上对 RAM 6116 进行读写操作时,IC₆ 地址译码器 74LS138 的输出端 12 脚(Y₃) 为 6116 提供片选信号,通过计算,6116 的地址范围是 C800~CFFF 2K 存储区,当我们在监控系统下,对 C800~CFFF 这段存储区进行读写操作时,IC₆ 的输出端 12 脚输出“L”电平,这个信号被接到:

1. IC₇ 的 19 脚上,该片处于工作状态,即三态门的闸门被打开。
2. IC_{9b} 的一个输入端上,另一输入端不论何种状态,其输出端为高电平,这个高电平又接到 IC₈ 的 19 脚上,此芯片停止工作,这就保证在 APPLE 机对 2K RAM 6116 进行读写操作时,数据不能流入被开发系统,被开发系统的数据也不能流入 APPLE 机,起到两个系统之间数据的封堵,防止系统之间的混乱。

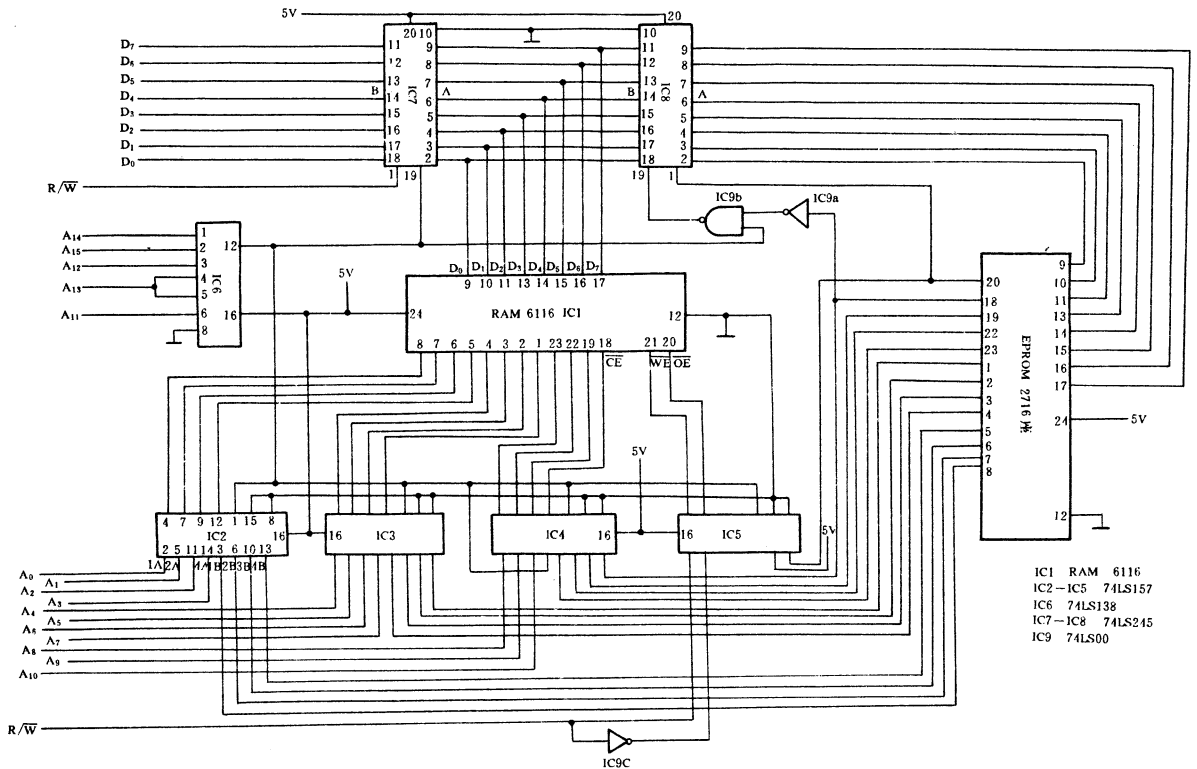
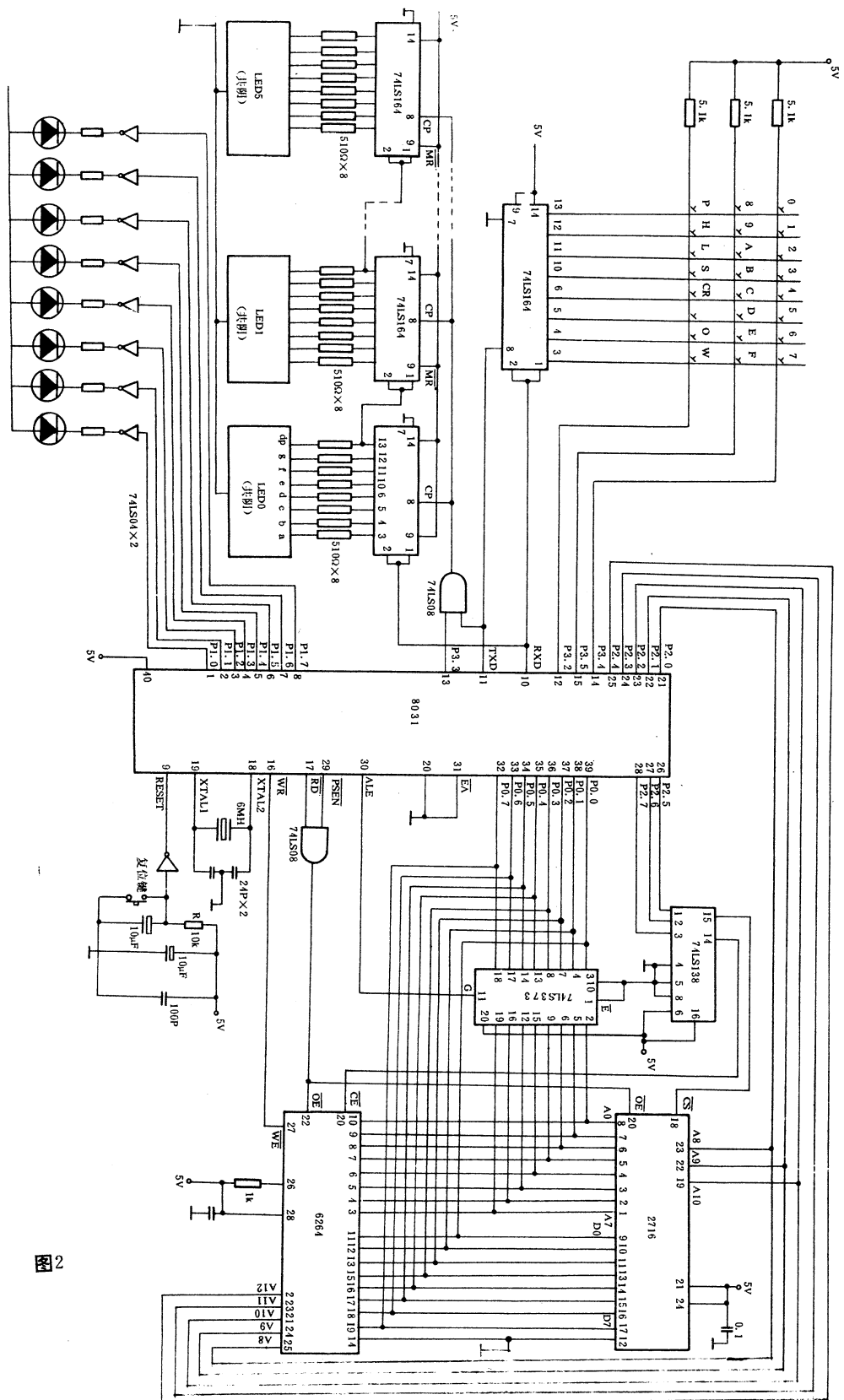


图1



2

3. IC₂~IC₅四个芯片所有的1脚上,则输出端按 A 口的输入状态进行输出,(A 口被接到 APPLE 机上)B 口状态不能被输出,(B 口被接到被开发系统上)这就达到被开发系统的地址线和控制线被封堵。

4. IC₄的14脚上,其对应输出端4y,将这个低电平信号传输到 RAM 6116的18脚的片选端上,只要对 C800~CFFF 这2K 存储区进行读写操作时,该片被选中。

当 CPU 读时,R/ \overline{W} =H(高电平)这个信号和这个信号通过反相器分别接到 IC₅的1A 和2A 上,其对应输出端1y,2y 又分别与 RAM 6116的21脚的 \overline{WR} (写控制端)和20脚 \overline{OE} (读控制端)相接,此时 \overline{WR} =H、 \overline{OE} =L,则 RAM 6116芯片处于被读状态,同时,R/ \overline{W} =H 这个电平又传输到 IC₇的1脚上,此时1脚为高电平,根据 74LS245这个芯片的特性,数据的流向由 A 到 B,即 CPU 将 RAM 中单元里的数据读回 CPU。

当 CPU 进行写入操作时,R/ \overline{W} =0,即读写线为“L”电平。而这个电平和这个电平通过反相器又分别通过 IC₅传输到6116的21脚和20脚上,此时这两个脚的电平分别为“L”和“H”。此时该芯片为写入状态,同时 R/ \overline{W} =“L”的信号又接到 IC₇的1脚上,则数据的流向为 B 到 A,即 CPU 将数据写入 RAM 6116存储器的单元之中。

当被开发系统(单片机)通过 EPROM 2716插座对 RAM 6116进行读操作时,IC₆的地址译码器的输出端12脚为 H 电平,这个电平被接到:

①IC₇的19脚上,此芯片停止工作,即内部闸门被关闭,不能传输数据,同样起到两个系统之间的数据被封堵,防止两个系统之间的混乱。

②IC₉的一个输入端上,而被开发系统的 EPROM 的片选信号,通过扁平电缆线传输到仿真器2716插座的18脚上,这个片选信号(L 电平)一路接到非门(IC_{9a})的输入端上,其输出端接到 IC₉的另一个输入端上,IC₉的两个输入端均为高电平则输出端为“L”电平,使 IC₈的19脚也变为 L 电平,该片处于工作状态,即内部的三态门被打开,另一路被接到 IC₄的4B 端口上,其对应输出端4y 接到6116的18脚的片选上,保证被开发系统读时该片被选中。

③IC₂~IC₅的四个芯片所有1脚上,均为高电平,则选择 B 口输出,A 口停止,APPLE 机的地址线和控制线被封堵。

当被开发系统“读”操作时,读信号传输到 EPROM 2716插座上(20脚),一路输入到 IC₈的1脚上,此时为“L”电平,则数据由 B 流向 A,即6116中的单元的数据通过数据总线读回被开发系统 CPU 中,另一路通过 IC₅的2B 的输入端,使其输出端2y 接到 RAM 6116的20脚读控制端上,而 IC₅的另一输入端1B 接到+5V 上,其输出端1y 接到6116的21脚写控制端上,此时 \overline{OE} =0, \overline{WR} =1,则该芯片处于读的状态。这样被开发系统将 RAM 6116单元中的数据读回 CPU。

二、硬件电路几点说明

1. 该系统利用 8031内部的振荡电路,在 XTAL1、XTAL2引脚上外接定时元件,内部振荡电路便产生自激振荡,形成单片机的时钟电路。晶振与电容的参数如图2所示。

2. 由上电复位和按键复位结合电路为8031提供复位信号(RESET),CPU 便从0000H 地址开始执行程序。

3. 地址总线由 P0口提供低八位(A₀~A₇)地址线,P2口提供高八位(A₈~A₁₅)地址线,由于 P0口还要作数据总线口,因此只能分时用作地址线,故 P0口输出低八位地址时必须用锁存器锁存在该系统中用 74LS373八 D 锁存器为 EPROM 2716和 RAM 6264提供低八位地址线,锁存信号由8031的 ALE 管脚提供。

4. 74LS138为地址译码器,为 EPROM 2716和 RAM 6264提供片选信号,其地址分别为 0000H~07FF 和 2000H~3FFF,其中2716存放着管理程序,6264为用户程序区和数据区。

5. 将外部程序存储器的选通信号线 \overline{PSEN} 和外部数据存储器选能信号线 \overline{RD} 相与,其输出端与6264 \overline{OE} 端相连,这样6264既可做程序存储器和数据存储器用,为用户的使用带来方便。

6. 本系统我们选用了 MCS-51串行口(设定串行口工作在移位寄存器方式0状态下)外接74LS164构成了键盘和显示器的控制电路,其中外接6片74LS164作为6位静态显示口,另外接一片74LS164作为键盘列线扫描输出口,通过实验静态显示亮度大,键输入稳定。主程序可不必扫描显示器,从而有更多的时间处理其它事情。

三、监控程序的几点说明

1. 本监控程序只编写了最基本的三大功能,其它功能留用户在学习本系统后根据需要自行开发,本系统显示缓冲区占68H~6DH 单元,键盘缓冲区占用58H~5EH 单元,用户不能占用。

2. 三大功能介绍如下:

①H 命令

作用:将程序写入存储单元。

格式:H×××× \swarrow H 后是存储器的四位地址,“ \swarrow ”为回车。数码管显示××××四位地址,然后键入数据再回车。

例如:从2000开始输入一段点亮 P1口的小灯程序。

操作:键入 H2000 \swarrow

显示2000

键入 74 \swarrow

显示2001

键入 FF \swarrow

显示2002

键入 F5 \swarrow

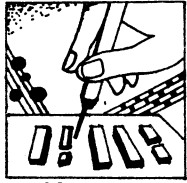
显示2003

键入 90 \swarrow

显示2004

键入 02 \swarrow

(转第22页)



学装微电脑

附带应急开关的顺序控制

易齐干

顺序控制过程中,为避免异常状态的发生,要不断地监视工作过程。如果发生应急事态,就要暂时中断正常工作,进行应急处理。本文讨论应急处理时硬件安排与软件的设计。

对应急处理的要求:应急开关为 ON,LED 灯熄;蜂鸣器蜂鸣。应急开关为 OFF,LED 灯恢复灯熄前的状态,蜂鸣器停止蜂鸣。

针对上述要求,我们可将它编制为子程序,在执行程序过程中,应急开关为 ON,或为 OFF,均执行预先安排的子程序。这在程序设计上非常方便。满足这种要求的功能称为中断。

μ p-80套件中 CPU Z80的16[#]出脚由 H 变为 L 电平,CPU 则产生中断,中断正执行的主程序,转而去执行 0038H 地址为开始的中断子程序。使用中断必须注意下述三点。

1. 为使 Z80 CPU 能接收中断程序,在程序开始时,必须添加“开中断”(EI 指令)和“1型中断”(IM1 指

令)语句。

2. 0038H 为中断程序开始地址,主程序的地址要避免中断程序地址。

3. CPU 执行完中断程序,要返回原主程序,则自动关闭中断。为能实现多次中断,应该在中断程序的最后,添入“开中断”(EI 指令)语句。

能正确满足上述要求的硬件安排如图1所示。输入输出部件的 A 口高位连接打码开关和 H/L 电平验证部件, S_0 打码开关为应急开关, S_0 为 ON, A 口第4位由 1 \rightarrow 0,同时 CPU 谋求中断。中断之后, CPU 查询 S_0 打码开关的 ON、OFF 状态,如果为 ON,LED 灯熄;蜂鸣器蜂鸣。如果为 OFF,LED 灯返回到熄灯以前的状态,蜂鸣器停止蜂鸣。这就是符合要求的应急开关处理。流程图与程序清单如图2所示。

应急开关的处理是在顺序控制过程中进行。顺序控制的流程图如图3所示。

0038H 地址开始的中断程序中,用 PUSH 指令

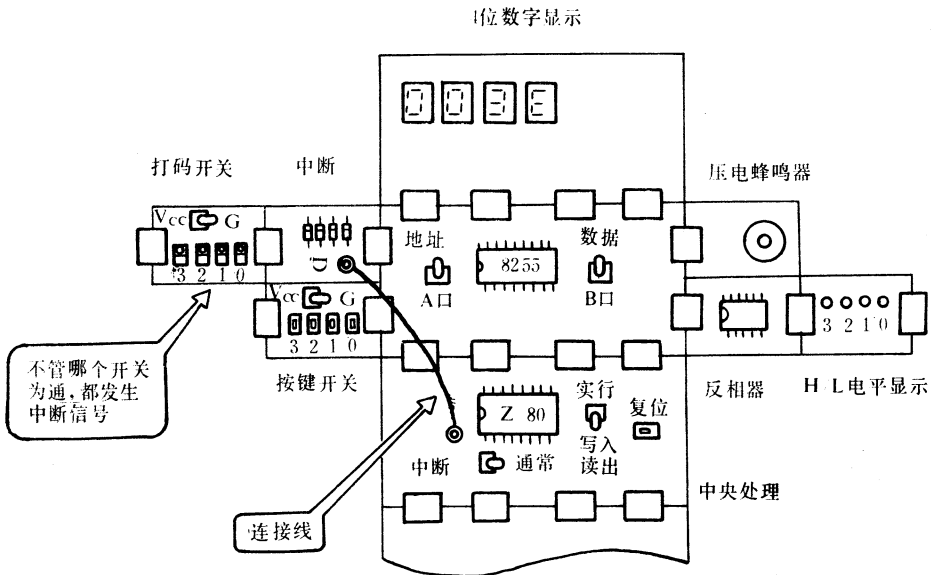


图1

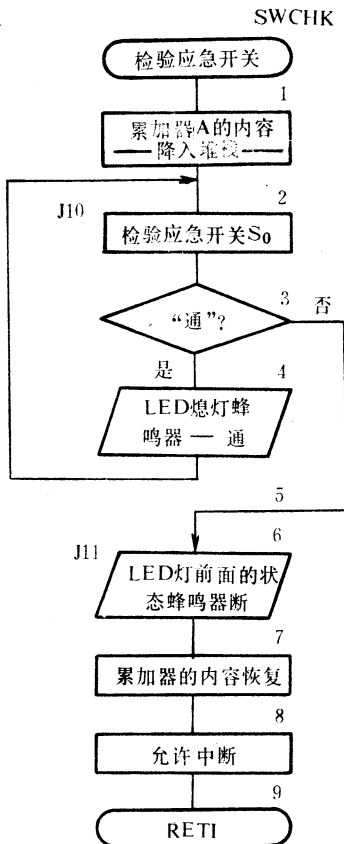
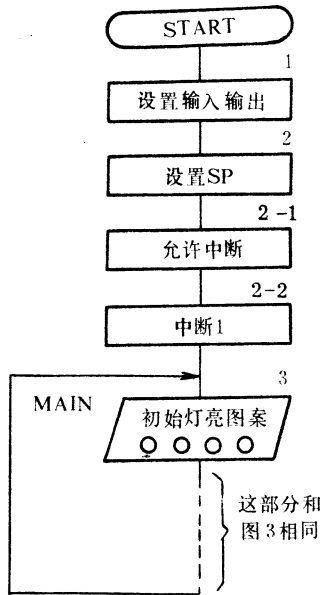
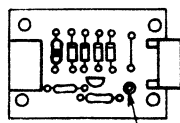


图2

标号		助记符	地址	机器语言
START	1	LD A,90H	0000	3E 90
		OUT(03H),A	0002	D3 03
	2	LD SP,0100H	0004	31 00 01
		JP MAIN	0007	C3 60 00
MAIN	3	LD HL,00F0H	0060	21 F0 00
		LD (HL),00H	0063	36 00
		LD A,(HL)	0065	7E
		OUT(02H),A	0066	D3 02
J1	4	IN A,(00H)	0068	DB 00
		BIT 0,A	006A	CB 47
		JP NZ,J1	006C	C2 68 00
	5	LD HL,00F0H	006F	21 F0 00
	6	LD (HL),06H	0072	36 06
		LD A,(HL)	0074	7E
		OUT(02H),A	0075	D3 02
J2	7	IN A,(00H)	0077	DB 00
		BIT 1,A	0079	CB 4F
		JP NZ,J2	007B	C2 77 00
	8	LD HL,00F0H	007E	21 F0 00
	9	LD (HL),09H	0081	36 09
		LD A,(HL)	0083	7E
		OUT(02H),A	0084	D3 02
J3	10	IN A,(00H)	0086	DB 00
		BIT 2,A	0088	CB 57
		JP NZ,J3	008A	C2 86 00
	11	LD HL,00F0H	008D	21 F0 00
	12	LD (HL),05H	0090	36 05
		LD A,(HL)	0092	7E
		OUT(02H),A	0093	D3 02
	13	IN A,(00H)	0095	DB 00
		BIT 3,A	0097	CB 5F
		JP NZ,J4	0099	C2 95 00
	14	JP NZ,J4	0099	C2 95 00
	15	JP MAIN	009C	C3 60 00



输出引线

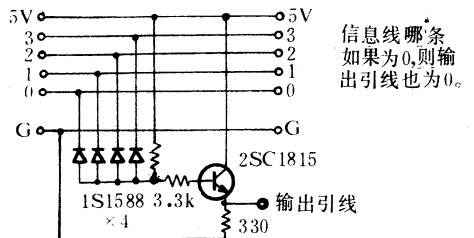


图4

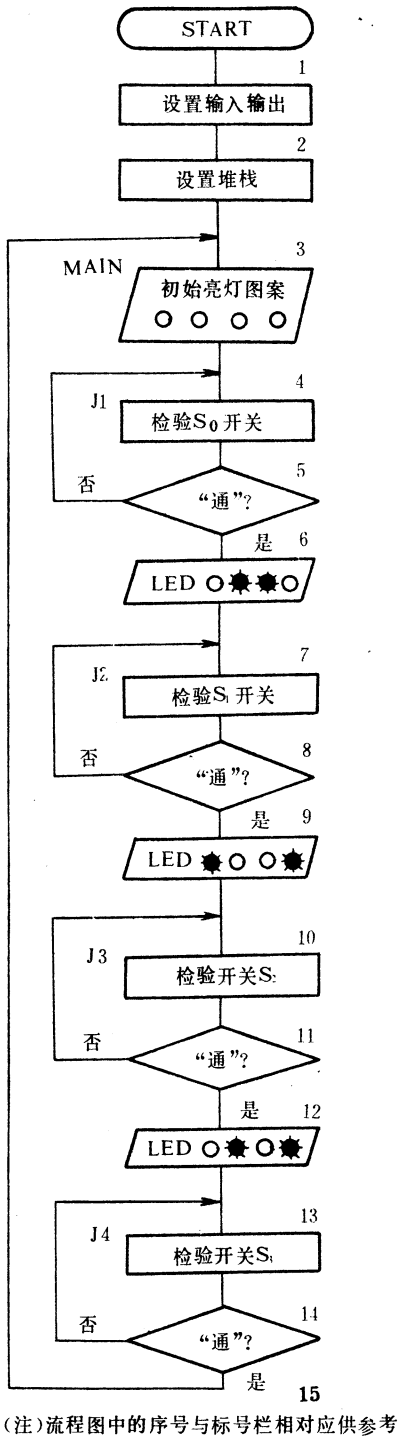


图3

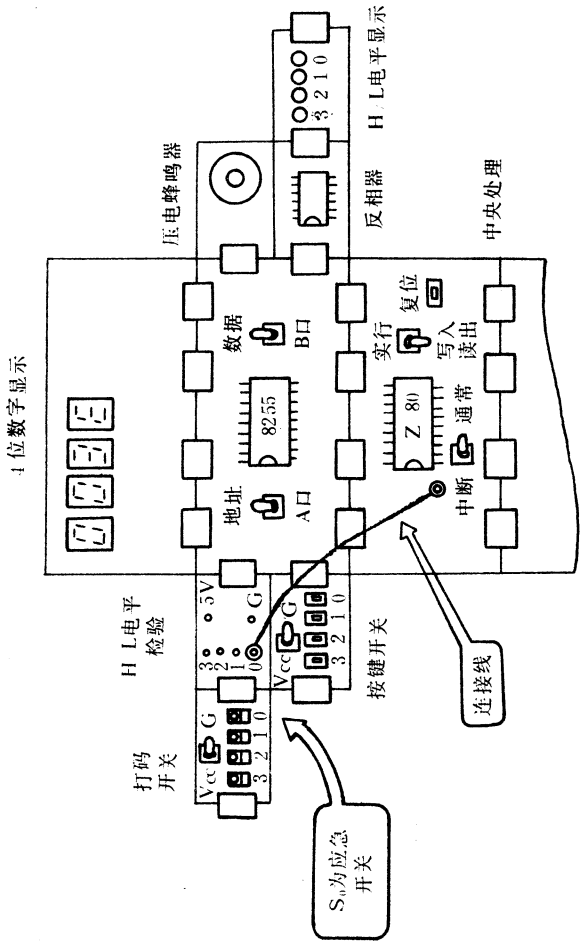


图5

将累加器 A 的内容推入堆栈,最后,用 POP 指令再将内容退回累加器 A。这样应急开关不会破坏主程序中开关输入的 ON,OFF 状态。

PUSH AF 语句中,A 为累加器 A,F 为标志 Z 或 C_y。

执行上述程序,既能对按钮开关 S₀~S₃进行顺序控制,又能查询应急开关 S₀的状态。

如果安排两个以上应急开关,可考虑如图4所示的部件。0*~3*任意一条信息线由 ON→OFF 时,均产生中断信号。按照图5接排 up-80 套件,则具备4种应急开关处理功能。任何一个打码开关为 ON,CPU 均执行 0038H 地址开始的中断程序。

本实验从另一个侧面告诉我们,硬件设计得好,可以减少软件负担。读者应该努力掌握硬件、软件两方面的知识,以达到平衡。



多功能程序移植器

——一种简易的硬件工具

北京西城区青少年科技站 朱立钢

1. 电路功能

我们经常希望把电子游戏卡或其它一些 ROM 片中的程序,复制到自己的可编程 ROM 片或 RAM 片中,进行研究和利用。而在单片机应用开发过程中,要把自己在 RAM 中编好的应用程序及简单监控程序写入 ROM,这样再投入实际应用,才不至于每次断电程序都会丢失,也可以省去软盘驱动器,实现廉价和无需专业人员操作的完全自动化。虽然我们对存储芯片已不再陌生,ROM、RAM 天天都在利用,然而这些存储芯片都在计算机的内部,对其读取都是在系统软件控制下,并通过 CPU 进行,而 ROM 的写入,目前通用的方法用专门的计算机仿真系统 MDS(如 ICE-48/49),或制做一些特殊接口配合计算机,接上存储器再编制专用程序完成内存复制(如国内的一些简易系统)。这些方法的价格都在千元以上,而且要求使用者有一定的软硬件知识,使一般的爱好者难以完成,使廉价的单片机开发受到限制。下面介绍的电路可以实现:把源存储器中任意始末地址的内容写入目标存储器。这个电路由于无需使用 CPU 和键盘,可以对存储器独立进行复

制,因而成本很低而且不受机型的限制,应用广泛灵活,设计使用 74LS 系列通用元件,成本只有十几元。即使无很多硬件和专门软件知识的人,也能操作,按下启动按钮,复制自动完成,简便迅速,应用范围广,地址总线数可变,可根据你的需要确定,所以很适合业余爱好者自制。

2. 电路原理

图 1 为电路设计思路: D11 与 D12 组成一个标准 RS 触发器,控制电路动态静态的切换。D13, D14, D21 组成可控振荡器,输出时钟信号。D23, D24 组成单脉冲发生器产生写入脉冲,并使此脉冲在两次地址加 1 之间的时刻产生,控制程序的写入与末地址的判断。IC1, IC2(四位二进制同步计数器)组成地址发生器。D42, D41, D32, D33, Kn-K0 与 Kn'-K0' 组成末地址检测器,当到达末地址时,产生一个停机信号到自停电路。

工作时首先把 K 拨到预置位,拨动 Kn'~K0' 与 Kn~K0 预置始末地址,然后按一下启动按钮, A 点置成高电平,可控振荡器 D13 门打开,脉冲可以通过, C 点立即输出时钟信号,动态指示灯亮。C 点的时钟信号进入各计数器,此时 B 点为低,各计数器置位,初地址被置入。再看 D22, 它在 CP 信号与单脉冲之间,设计它的目的有两个:一是在置位时 D22 的 4 脚为 0, CP 信号无法通过,不发出写入脉冲。这可以防止在置位地址的重复写入,因为 ROM 的写入次数是有限的;二是它使写入脉冲正好在两次地址加一之间,否则地址变化时的瞬时杂信号会影响工作。K 拨至存入,再看 C 点, CP 上升沿时计数器触发地址加 1,在 CP 下降沿时经 D22 变为上升沿触发单脉冲发生器产生一个大于 100 微秒的写入脉冲,原存储器的一位数据自动写入目标存储器,然后 CP 上升沿又使地址加 1,下降沿再写入一位。两部分如此交替工作直到地址输出与预置的末地址相同时, E 点输出低脉冲, RS 触发器翻转, A 点变为低电平,振荡器停振,动态指示灯熄灭,这时复制就完成了。

3. 使用举例

这里举一个把一片 ROM 中的程序写入一片 RAM 的例子来具体说明程序移植器的使用。第一步,打开电源,动态指示灯可能是亮着的,这时要等几秒钟它会自动熄灭。这时进行第二步, K 拨至置入,置入要写入程

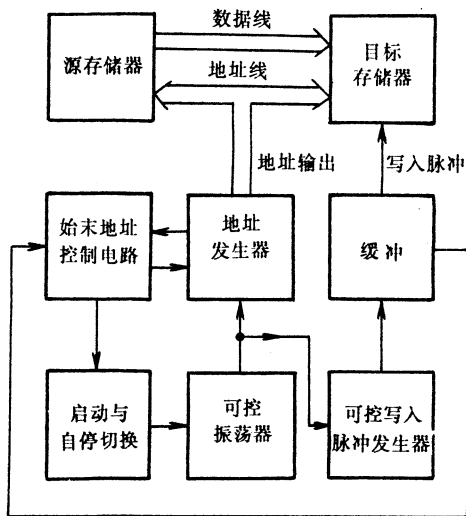


图 1

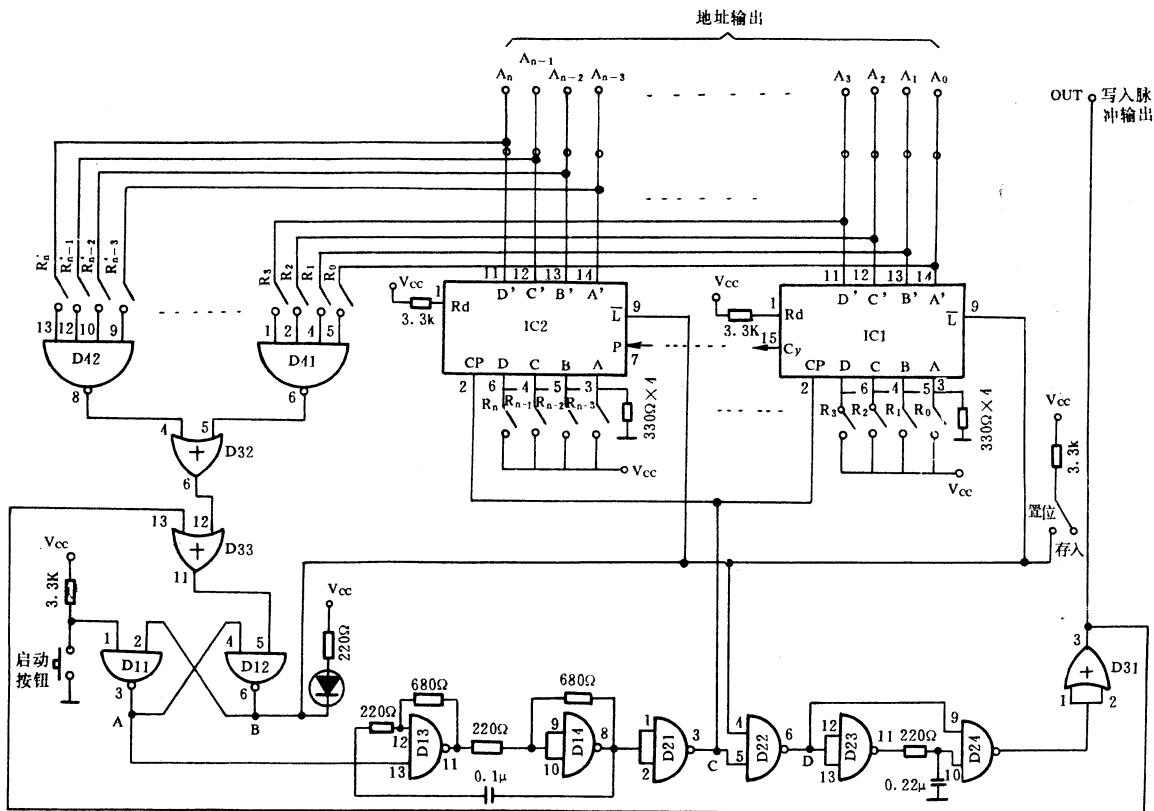


图2

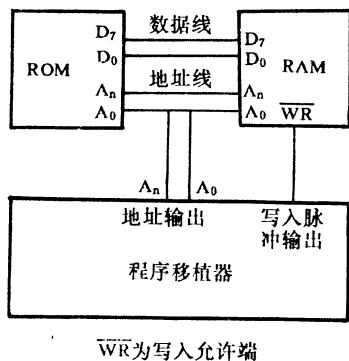


图3

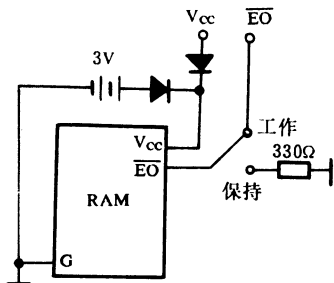


图4

序的始、末地址。第三步，连接源存储器与目标存储器如图3。第四步，按一下启动开关，动态指示灯亮，K拨至写入，经过几秒钟后，复制完成，动态指示灯熄灭。

4. 制作和使用注意事项

1. 试验电路工作可在输出口逐位检验，并用手动给定C点信号，检查地址变化。
2. 如果把此电路接入你的计算机时，要注意地址线由移植器占用时其它地址线上的输出单元都要断开（CPU要中断），即高阻输出，数据线也是这样，不能同时占用总线。
3. 如果只需全部地址的移植，可以省去n位拨码开关 $K_n \sim K_0$ ， $K_n' \sim K_0'$ 用导线代替，但置位按钮不可省。
4. 为实验方便，介绍一个RAM作ROM使用的方法，见图4，开机工作时电池不耗电，关机前把开关拨至保持，这时电流很小，如我使用的5101型CMOS RAM只有20微安（工作时40毫安）两节5号电池可使用半年。

5. 元件选择

74LS00 × 2; 74LS32 × 1
74LS161(或139) × 2; 74LS20 × 1;
其它元件无特殊要求。



电脑游戏机

第二章 F BASIC 的基本语句

山东苍山机械电子化学工业局(277700) 于 春

三、数值函数语句。

1、绝对值函数

ABS 简写 AB.

该函数的功能是将数值换成绝对值。如

P. AB. (-41) 回车后,显示41

2、符号判别函数

SGN 简写 SG.

SGN 用来判别数的性质,当 X 为正数时 SG. (X) = 1; X 为负数时 SG. (X) = -1; X 为零时 SG. (X) = 0。

3、随机数函数

RND 简写 RN.

RND 语句可使计算机产生随机数,其格式为 RN. (X), X 的取值范围为1~32767,用它可产生小于 X 的随机数。

为指导小学生进行数学练习,介绍两个实用程序,读者可以举一反三。

例14 一位数乘法练习程序

10 A=RN. (10):B=RN. (10)

20 P. A " * " B " = " ;

30 I. C

40 IF C=A * B T. P. "V":G. 10

50 P. "X":G. 20

RUN 后,显示一位数乘法算式,输入得数后,若正确显示"V",给出另一算式;若错则显示"X",重复打印算式。

例15 两位数加法练习程序

10 A=RN. (100):B=RN. (100)

20 P. A " + " B " = " ;

30 I. C

40 IF C=A+B T. P. "V":G. 10

50 P. "X":G. 20

读者可以发现,两个程序格式相同,不同的是更换了随机数上限和运算符号。因此,读者可照此编写其它位数的乘、加、减练习程序。

四、数据语句

数语句包括读数语句(READ)、置数语句(DATA)、恢复数据区语句(RESTORE)、存数语句(POKE)和取数语句(PEEK)。

1、READ 语句和 DATA 语句

READ 简写 REA.

DATA 简写 D.

当有许多变量需要赋值,用 LET 语句就显得很繁琐,这时可采用 READ 和 DATA 语句向变量提供数据:

10 REA. A,B,C,D,E,F,G

20 D. 3,4,5,6,7,8,9

当执行 READ 语句时,变量的值依次从 DATA 语句中读出,即 A 读3,B 读4……

可以发现,当变量赋值较多时,用 READ、DATA 语句比较简单。

在使用 READ、DATA 语句时要注意两点:

(1)由于 DATA 语句是非执行语句,所以它可以放在程序中的任何位置。但若有多个 DATA 语句,它们之间必须有先后顺序。另外 DATA 提供数据的个数,不能少于 READ 中的变量数。如:

10 D. 3,4,5,6

20 REA. A,B,C

30 P. A,B,C

与

10 REA. A,B,C

20 P. A,B,C

30 D. 3,4

40 D. 5,6

两程序效果是一样的。

(2)DATA 语句中的数据可以是数值、字符串,但不能是变量、函数或表达式。DATA 语句中的数据类型应严格与 READ 语句中的变量类型一一对应。如:

10 REA. A\$, AB, CDE

20 D. "Zhong Guo", 1991, 8, "Computer"

READ 中有三个变量。第一个是串变量,后两个是实变量。DATA 中有四个数据,一、四为字符串,二、三为实数。两语句中一、二、三分别对应,所以程序正确。至于 DATA 中多了一个串变量"Computer"则没有关系,不影响程序的运行。

我们学习了读数、置数语句后,例13分类统计学生成绩的程序中可不使用 INPUT 语句,因每输入一个数要回车一次,也嫌麻烦。可修改程序如下,改20、120两

行:

```
20 REA. X
120 D. (学生成绩单),110
```

读者可自己练习。

2、恢复数据区语句

RESTORE 简写 RES.

如果要3~9七个自然数分别送给变量 A~G、H~N、O~U,一般程序为:

```
10 REA. A,B,C,D,E,F,G
20 REA. H,I,J,K,L,M,N
30 REA. O,P,Q,R,S,T,U
:
100 D. 3,4,5,6,7,8,9
110 D. 3,4,5,6,7,8,9
120 D. 3,4,5,6,7,8,9
```

大家可以看到,三行 DATA 语句是重复的。为节省内存,简化程序,可使用 RESTORE 语句。当程序执行到 RESTORE 时,便使下一个 READ 语句需要的数据又从最小行号的 DATA 语句的第一个数据读起。于是上面的程序可改写为:

```
10 REA. A,B,C,D,E,F,G
20 RES. :REA. H,I,J,K,L,M,N
30 RES. :REA. O,P,Q,R,S,T,U
:
100 D. 3,4,5,6,7,8,9
```

3、三种提供数据语句的比较

到此为止,我们已经介绍了 LET、INPUT、READ/DATA 三种语句。这三种语句都可以给变量赋值,同一个问题可分别用三种不同的方法编写程序。现举一例说明。

例16 鸡兔同笼,已知鸡兔总头数为 T(Tou),总脚数为 J(Jiao),求鸡兔各有多少只?

先用代数求出鸡兔个数的数学表达式。设鸡 X 只,兔 Y 只,则有

$$X = (4 * T - J) / 2$$
$$Y = (J - 2 * T) / 2$$

今设 T=16, J=40,编程如下:

程序 I:用 LET 语句

```
10 T=16:J=40
20 X=(4*T-J)/2:Y=(J-2*T)/2
30 P. "JI="X, "TU="Y
```

程序 II:用 INPUT 语句

```
10 I. "T,J",T,J
20 X=(4*T-J)/2:Y=(J-2*T)/2
30 P. "JI="X, "TU="Y
```

程序 III:用 READ/DATA 语句

```
10 REA. T,J
20 X=(4*T-J)/2:Y=(J-2*T)/2
30 P. "JI="X, "TU="Y
40 D. 16,40
```

由上例可见,三种语句的相同之处是都能提供原

始数据,但使用特点各有不同。LET 语句适合于赋值变量少,且编程时数据已确定的场合。另外赋值语句可用来运算。这一功能,其它两种无法取代。如果数据很多,且编程时已确定,使用 READ/DATA 语句比较方便;如参数是变化的,要随时输入变化了的数据,则宜于采用 INPUT 语句,如商业柜台,银行付息的计算等。三种语句掌握熟练后,定会给编程带来极大的方便,希望读者多加练习。

4、POKE 和 PEEK 语句

POKE 简写 PO.

PEEK 简写 PE.

POKE 和 PEEK 是两个互为逆操作的语句,POKE 的作用是把数存到指定的内存单元;PEEK 的作用是把指定内存单元的数取出来。这两个语句,随机手册中已有举例,本文不再列举。由于这两个语句在 F BASIC 的程序设计中使用不多,读者仅仅了解就行了。

五、数组及数组说明语句(DIM)

在第一章中我们介绍了简单变量。现在介绍第二种变量——下标变量。

1、下标变量

顾名思义,下标变量就是带有下标的变量。如 A(K)、B(I)等。使用下标变量的目的,是为了使一些性质相近的变量便于归类。如就 A(K)而言,K=1时,A(K)就是 A(1);K=2时,A(K)就是 A(2)……。

同一个下标变量名可以包含有两个下标,其形式为 A(I,J)。一般称一个下标的为单下标变量;两个下标的为双下标变量。

2、数组:

由同一个变量名组成的下标变量的集合叫数组。在实际应用中,人们习惯把一些具有相同性质的数据放在一个数组内,用不同的下标去区分。

由单下标变量组成的数组称一维数组;由双下标变量组成的称二维数组。当数组进入电脑后,每个下标变量占用若干连续的内存单元。因此,一个数组要占用内存中连续的一片单元。

3、数组说明语句(DIM 语句)

用一个数组说明语句可以定义一个数组或几个数组,它规定数组的名称及变量个数。如

```
10 DIM A(8)
```

此语句定义了一个实型数组,变量名为 A,下标变量的个数有9个(下标为0~9),最大下标为9。对于 A(m,n)数组,变量个数为 m+1 个,对于 A(m,n)数组,变量个数为 (m+1) × (n+1) 个。DIM 语句是非执行语句,一般放在程序的开头。

例17 从 DATA 语句中读出10个数据存于 A 数组中,然后将数据次序颠倒后再存于 B 数组中。程序如下:

```
10 DATA 3,4,5,6,7,8,9,10,11,12
20 DIM A(9),B(9)
30 F. I=0 TO 9:REA. A(I):N.
40 RES.
```


50 F. L=9 TO 0 ST. -1;REA. B(L):N.

60 F. M=0 TO 9:P. A(M),B(M):N.

例18 有10个杂乱无章的数,要求编一程序把这些数按大小顺序排列出来。

```
10 DIM A(9)
20 D. 38,99,26,78,32,100,27,66,95,46
30 F. K=0 TO 9:REA. A(K):N.
40 F. L=0 TO 8
50 F. N=0 TO 8-L
60 IF A(N)<=A(N+1) T. 80
70 SWAP A(N),A(N+1)
80 N. :N.
90 F. I=0 TO 9:P. A(I):N.
```

程序运行后显示从小到大排列的顺序。若欲从大到小排列,只须把90行改为:

```
90 F. I=9 TO 0 ST. -1:P. A(I):N.
```

即可。

程序中使用了变量交换语句 SWAP。它的格式为 SWAP n_1, n_2 , 其功用是使变量 n_1, n_2 的内容进行交换。这一语句是 F BASIC 所特有的, 若用一般 BASIC 语句进行两变量内容的交换, 需如下编程:

```
70 T=A(N):A(N)=A(N+1):A(N+1)=T
```

可以看出, 使用 DIM 语句, 可以对学生成绩排列名次。读者可以试编一程序, 并打印出名次和成绩。

六、CLEAR 语句

CLEAR 简写 CLE.

CLEAR 语句有两个功能:

1、规定使用内存的范围。

为了防止因用户程序太长而与 F BASIC 系统内存空间相冲突, 用 CLEAR 语句来规定用户的内存空间范围。

如10 CLE. &H7600

规定程序不能使用十六进制数7600以前的内存空间。

注意——该功能只在对话进入 BASIC 状态时有效, 而在直接进入 BASIC 状态时无效。

2、清除内存中所有变量的值。

当程序执行到 CLEAR 时, 程序中所有变量的值被清除。即: 数字变量的值变为0; 字符变量变为空。

该功能在两种 BASIC 状态中都有效。

七、程序运行控制语句(STOP、CONT、PAUSE)

1、在程序运行中, 输入 STOP 指令可中断程序运行, 以便于测试程序或检查程序的执行情况。然后, 输入 CONT 指令, 可令程序在原中断点继续往下运行。

2、暂停语句(PAUSE)

PAUSE 语句可令程序运行暂停一定时间后再继续运行, 暂停时间的控制量为0~32767。示例见随机手册。

八、子程序调用(GOSUB)和返回语句(RETURN)

在一个程序中, 某些部分往往要反复执行, 为避免重复编写实现同一目的程序而引入了子程序。子程序为一具有某种功能的程序, 它是独立的, 可以放在主程

序之外。当主程序需要这一功能时, 就可以调用这一子程序。

1、子程序的调用

主程序调用子程序的方法很简单, 即编一个转子语句, 就可以转去执行子程序, 转子语句的格式为:

```
GOSUB<n> 简写 GOS. <n>
```

式中 n 为子程序第一语句的行号。

子程序的最后一条语句必须是返回语句 RETURN, 简写为 RE., 当程序执行到 RETURN 时, 自动返回到 GOSUB 下面的语句继续执行主程序。

例19 设 $A=4, B=5, C=6$, 试编一个程序求 $S=A!+B!+C!$?

```
10 GOS. 100
20 A=P
30 GOS. 100
40 B=P
50 GOS. 100
60 C=P
70 P. "=";A+B+C:E.
80 D. 4,5,6
100 P=1:REA. M
110 F. J=1 TO M:P=P * J:N.
120 P. "+"M"!";
130 RE.
RUN +4!+5!+6!=864
```

2、ON-GOSUB 语句

这是开关转移语句的第二种形式。格式为

```
O. <表达式>GOS. <子程序入口行号>
```

在这里有几个子程序, 就有几个入口行号, 根据 ON 后面表达式的值决定转向哪一个子程序。如:

```
10 ON A GOS. 100,700,1000,.....
A 若取值为1,则转向100行;
A 若取值为3,则转向1000行.....
```

执行完那个子程序后, 返回到10语句的后一语句。

3、子程序的嵌套

子程序也可以再调用其它子程序, 这叫子程序的嵌套。但不允许子程序调用主程序。关于子程序的嵌套, 上例中120行以后程序可以改写如下:

```
120 GOS. 200
130 RE.
200 P. "+" ;
210 GOS. 300
220 RE.
300 P. M ;
310 GOS. 400
320 RE.
400 P. "! " ;
410 RE.
```

当然, 在实际编程中, 这样转来转去没有多大意义, 示例仅在于使读者了解怎样使用子程序的嵌套。

九、有关程序调试的指令

重编号指令 (RENUM)、删除行号指令 (DELETE) 和查找指令 (FIND) 随机手册中没有介绍, FIND 指令还没有发表。由于它们在程序的调试中经常用到, 现予以分别介绍。

1. 重编号指令 (RENUM)

RENUM 简写 REM.

一个较复杂的程序编写完毕, 一般都要在运行中调试。由于调试程序将使行号变得杂乱, 不便于阅读。这时可键入 RENUM 指令, 使行号重新排列为 10, 20, 30, …… 的整齐形式。在重编号中, 语句中的 GOTO、THEN、GOSUB 等语句的行号也随之改变, 所以不要担心打乱了原有程序。如

```
10 F. A=1 TO 55
20 B=A * A
22 IF B>35 G. 33
25 P. A,
31 N.
33 P. "B>35"
```

键入 RENUM 回车后, 键入 LIST, 列表为

```
10 FOR A=1 TO 55
20 B=A * A
30 IF B>35 GOTO 60
40 PRINT A,
50 NEXT
60 PRINT "B>35"
```

2. 删除行号指令 (DELETE)

在调试程序中, 往往要删去部分行号, 可使用 DELETE 指令。

DELETE 简写 DEL.

若键入 DEL. 100 则删除 100 行。
若键入 DEL. 100-200 则删除 100-200 行。
若键入 DEL. 100- 则删除 100 及以下所有行。
若键入 DEL. -100 则删除 100 及以前所有行。

3. 查找指令 (FIND)

在调试程序中, 如欲查找某一数字、某一变量、某一表达式或字符串在程序中的位置, 可使用 FIND 指令, 尤其在阅读别人编制的程序, 掌握某一变量随程序运行的变值等, FIND 指令尤为重要。

FIND 简写 FI.

FIND 指令的使用格式为:

FI. "变量或数值或表达式"

回车后, 立即打印查找内容所在程序的所有行。

注意——FIND 指令中, 查找的内容必须用双引号引起来, 否则, 给出 TM ERROR 出错提示。

作为本章的结束, 再举一例:

例 20 民间趣题——韩信点兵

有一队兵, 若三个一数则余 A, 若五个一数则余 B, 若七个一数则余 C, 求最少有多少兵?

民间解法有一首诗以加强记忆:

三人行走七十稀,
五树梅花廿一枝。
夫妻团圆半月正, (15)
减百零五便得知。

意思是说, 被三除的余数乘以 70, 被五除的余数乘以 21, 被七除的余数乘以 15, 三个积加起来减 105 便是得数。即

$$70A + 21B + 15C - 105 = X$$

民间解法存在一个问题, 即有时三个余数之和大于 105 的两倍, 如 $A=2, B=4, C=5$, 按上述公式求出的 $X=194$, 其实 X 的最小解是 89, 所以必须减两次 105。用计算机求解, 便可避免这一问题的出现。

设某数为 N , 被 A 除余 X , 被 B 除余 Y , 被 C 除余 Z , 程序如下:

```
10 I. A, B, C
20 I. X, Y, Z
30 D=A * B * C
40 F. N=1 TO D
```

```
50 IF N MOD A=X T. IF N MOD B=Y T. IF N  
MOD C=Z T. 70
```

```
60 N.  
70 P. N:G. 20  
RUN
```

先输入三个除数, 如 3, 5, 7, 回车后, 再输入三个余数 X, Y, Z , 则可打印出得数 N 。这时除数一定, 每输入一组余数, 可得一个 N 。如欲改变除数, 如改为 5, 9, 13, 键入 STOP 中断运行, 再键入 RUN 重新运行, 依次输入 5, 9, 13 后便可进行变除数计算。

本程序对原题意进行了扩展, 因此, 游戏趣味更浓厚。

到此为止, 我们介绍了 F BASIC 33 条最基本的语句和指令。运用这些语句, 读者可以较容易地编制出解决一般复杂问题的程序。还有一部分语句, 将作为 F BASIC 的提高和加深另文介绍。下一章重点介绍 F BASIC 所特有的画面、卡通控制语句。

(上接 26 页)

SIC 语言) 之后, 为什么还要学习汇编语言呢? 这除了我们已经在前面介绍过的优点外, 汇编语言还有以下几个特点:

- 节省内存空间和 CPU 资源。
- 执行速度快。

- 能准确掌握程序执行时间。
- 更适用于实时控制和数据采集。
- 有利于对高级语言的评估, 更清晰理解它们在机器中的执行细节。
- 方便应用软件和系统软件的开发。



微机安装和使用中应注意的几个问题

中国人民大学信息中心系统研究室(100872) 胡野红

维修经验谈

微机在我国有众多用户,但总有些用户在安装和使用中忽略其供电要求,以致造成不必要的损失和麻烦。轻者是机器在运行时容易出现随机读写错、死机等故障。重者则损坏主机部件,甚至发生人身安全事故。据我所知且经过数年工作实践,只要注意以下几个问题,就可避免上述损失和麻烦,给工作带来方便。

一、按左零右火连接电源线

按常规,用电设备交流输入插座是按左零(零线)右火(火线)安装,微机随机所带电源线大都有明显标记。火线英文标记为 L(Live),零线标记为 N(Neutral)。其原理是按以上要求连接电源线时,能使火线中的电流经过机内电源的保险管。当供电线路或机器本身发生过流时,保险管则迅速熔断,使负载免受其害。其示意图如图1所示。

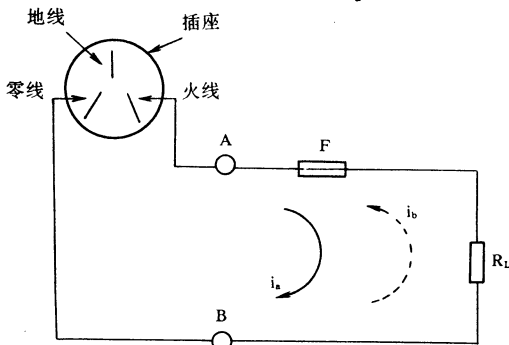


图1

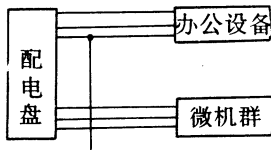


图2

正常工作时,A端是火线,电流 i_L 沿实线方向先流经保险管F再供给负载 R_L (主机)形成回路。当发生过流时, i_L 迅速超过额定值数倍或数十倍,F快速熔断切断供电回路,保护了负载 R_L 。若火线和零线接反,即B端接火线时,电流 i_B 沿虚线方向先经过负载 R_L 后再过

保险F,当发生过流时,迅速增大且大大超过额定值的 i_B 必先烧毁负载 R_L 保险管起不到任何保险作用。所以一定要按设备要求连接电源线。

当一台微机安装完毕,加电前应仔细检查电源线连接是否正确?具体做法是:拔下电源线连接主机的一端后,合上供电电源开关,用验电笔检查该线连接主机电源插座左边一端的插孔是否是火线?若是,则再用万用表测量供电电压是否和微机标牌要求一致?否则不可给主机加电。

二、接可靠的地线

接地线的目的是当微机本身发生故障或其它原因引起微机外壳带电时电流经地线流入大地,从而保护了微机 and 操作人员的安全。其次,因主机稳压电源采用振荡频率为20kHz的开关线路,虽然在线路上采取了一定措施,但仍在其屏蔽外壳上产生数十伏的感应电压。由于该壳和主机板地线同地,若不接地线,感应电压将被加到主机板上,足以损坏仅需5伏电压供电的集成电路,或者该电压干扰主机工作,使主机出现莫名其妙的故障,如“死机”、丢失数据等。以致机器“坏了”修,修了“坏”,同一故障现象多次发生而查不到故障的真正原因,造成人力、物力、机时的浪费。所以安装微机一定要接可靠的地线。

三、忌地线、零线绞接在一起

我国大多数工、民建筑均采用三相四线制供电线路,由电工原理可知,当三相负载不平衡时,零线中将有电流通过。若微机接入这类供电线路且接地线和零线绞结在一起时,零线中有相当能量的不平衡电流将通过地线流经微机外壳,使微机外壳带电,给机器本身和操作人员的安全带来威胁。当地线和零线分开连接时,零线中的不平衡电流将流经机内电源处理。所以,安装微机千万不要把零线和地线绞接在一起。

四、微机不要和其它办公设备同用一供电线路

大多数办公设备如复印机、空调机、吸尘器、电扇、日光灯均为感性负载,当这些设备启停时,会产生数倍于(一般为10倍)电网电压的过电压,如果把微机接入有这类负载的供电线路中,该过电压足以损坏微机。所以,微机供电线路一定要和办公设备分开连接。正确的连接如图2所示。

CEC-I 中华学习机修理一例

广西来宾铁路中学(546138) 卢光怀

我使用的 CEC-I 中华学习机,半年多来出现下列故障现象:主机与电视机连接,开机启动,操作正常运行。约过20分钟左右,电视机屏幕上图象出现上下翻滚,且跳出杂乱无规则的字符,主机发出笛声报警,然后出现死机。关机过一会儿再开机启动,屏幕显示正常,键入 CTRL-RESET-TEST 机器进入自检,其显示结果也正常。操作一段时间又出现上述故障现象。

修理:开盖首先检查电源,各路电压均正常,待机器出现故障时,电源电压也未发生变化,由此否定是由电源引起的故障,洗手或抚摸金属泄放手上电荷,仔细检查主机板上印刷线路有无断裂,各分立元件有无虚焊松动,各集成块插件有无松动,均无发现异常。重新开启主机,趁未出现故障时,触摸分立元件或集成块插件,当轻微敲击视频输出盒时、故障出现。于是关机,小

心焊下视频输出盒,打开盒盖仔细检查,发现视频线插口中心焊脚稍有松动,焊脚与印刷线路板焊接处有裂纹。由此可认为是引起故障的可疑之处。用烙铁加焊锡焊牢,尔后装机。经数日长时间开机操作使用,均未再出现故障。

大家知道,启动电脑都要求先开外设电源,最后再开主机,以免外设干扰冲击主机。同样,视频输出插座接触不良,造成时断时通,干扰脉冲必然影响主机正常工作。同理,视频输出线内的接触不良,电视机或监视器输入接口接触不良,都会造成这类故障。外部设备连线的插头,经常插入和拔出主机的插座,很容易损坏接口。这提醒我们,作为使用者,在插入或拔出插头时,要格外小心,位置应正确、用力要适度;作为修理者,首先要注意接口部位的检查。

软盘驱动器综合故障维修一例

建设银行黄冈地区中支电脑科 周凯歌

我单位有一台 AST 286微机上的1.2M 软盘驱动器(NEC FD1157C),在使用过程中突然出现三种不正常现象:1.工作时发出很大噪声。2.不能正常读写软盘。3.严重划伤软盘。

根据上述故障现象,初步分析为:故障现象1与灰尘和润滑条件有关;故障现象3可能是磁头上有灰尘或1面与0面磁头不平行造成的;故障现象2可能产生的原因有很多,有待深入细致的检查。在作了以上初略分析之后,我是这样处理的:首先打开主机箱,取下1.2M 软盘驱动器,发现里面堆积的灰尘的确很多,这与初步判断相吻合。用毛刷小心地将软盘驱动器上下表面的灰尘清扫干净后重新将其装到主机上,再开机观察,发现噪音依然存在,仔细辨听,发现噪音是从电机转动轴心位置传出的,而且噪音象是机械噪声。关机后,卸下软盘驱动器上的浮动夹紧装置,发现其压紧轮表面有一层灰尘,而且下面主轴电机轴心的杯形结构上也有很多灰尘,故先用毛刷清扫之,再用软布蘸上磁头清洁剂将其表面擦洗干净,然后装上浮动夹紧装置,重新开机后,噪音消除了。下一步再来解决划盘问题。用小镊子夹住软布蘸上磁头清洁剂小心擦洗上下两个磁头,然后再插入软盘,开机检查仍有划盘现象,关机后再认真观察上下两个磁头是否平行,当然凭直观很难看出,

我采取的办法是,不管平行与否,先作一次平行校正。具体作法是:将一块磁头清洗盘插入该软盘驱动器中,然后反复开关驱动器的小门,并在关上小门后用手指均匀下压1面磁头(注意,这些步骤是在没开机的情况下进行的),作了上述处理之后,划盘问题也得以解决。剩下的就是集中精力解决不能读写的问题。如前所述,造成不能读写的可能性很多,比如无索引信号,读写电路故障,读写磁头断线,磁头定位不准等等,都会造成读写不正常。使用一台 CQC-A 型磁盘驱动器测试仪和一台 V-1065型双踪示波器,接上该软盘驱动器,先用一张空白软盘插入该驱动器,关上小门后,发现有索引脉冲,“00”道信号和 RDY 信号,再检查其寻道功能也正常,然后进行自读自写,从示波器显示的读写波形来看是正常的。以上检查说明该驱动器的电路部分和两个磁头都是好的,可以断定是机械定位问题。把空白盘取出,换上 CE 盘,利用示波器和驱动器测试仪,反复调校“猫眼”信号,方位角信号,“00”道信号和索引信号,直到上述四种信号的波形都合乎精度要求为止。这时取下该驱动器,将其安装到主机上,先测试其读写和格式化功能都正常,再检查其互换性也很好,至此,该软盘驱动器修复。

电子工业出版社软件部新出版软件介绍

一、PC 及其兼容机软件

1. 汉字复杂报表自动生成系统 软盘:2片

定价:150元

本软件采用菜单选择方式,能自动生成多层表头汉字报表,表格修改方便,还能被其它管理系统调用,使已用 dBASE 进行事务管理的系统得到扩充。它在 DOS 系统下直接运行,操作简单,兼容性好。

2. 中西文辞书编纂系统 软盘:2片

定价:180元

本软件用 Turbo BASIC 编成。它能对用 EDLIN 或 Wordstar 输入的无顺序中西文词条,按设计文件格式,进行自动排序。排序方法有四种,即汉字按“纯笔顺”、“音序十笔顺”和“偏旁十笔顺”排序,英文按“字母”顺序排序。本软件可在 DOS 下直接运行,适合辞书作者和编辑以及其它需要进行字、词、人名与地名排序的工作人员使用。

3. 多方案联想式长城系列机 DOS 软盘:3片

定价:200元

本系统是在长城系列机及其兼容机的硬件基础上研制的新一代汉字磁盘操作系统。它沿用了原 GW BIOS 的扩充 ROM-BIOS 10H 外部显示管理模块的功能,新设计了 16H 中断管理程序,使之可支持 O14、CEGA 和 CMGA 等显示系统;屏幕提示区扩大,提出格式优化;它支持多种汉字输入法且具有联想功能;它有字词两种编码,可重复输入,能自动进行大小写转换,自动显示内码、区位码和外码,提供窗口式帮助。

4. 彩色屏幕界面程序自动生成工具 软盘:1片

定价:260元

本软件通过选择菜单,利用上下左右键即可确定其在屏幕上的位置,自动生成用户设计的、可在 dBASE 或 FoxBASE 环境下运行的程序,该程序能并入用户所开发的系统。

5. 通用工资管理系统 软盘:1片 定价:300元

本软件的功能是建立工资数据库,修改库结构和内容,任意设定和修改工资计算表达式,对工资数据进行查询和打印,完成任意单位的工资核算处理。它通用性强,结构灵活,操作简便,适用于 IBM PC/XT, AT 及其兼容机,要求 640K 以上内存,显示 25 行以上的汉字。软件环境:CCDOS,UCDOS,NCDS 和北大 DOS 及西山 DOS 等。

6. 清华中西文多语种操作系统 软盘:12片

定价:600元

本软件主要是汉字操作系统,也能处理其它多国文字,它不仅本身自成系统,也可挂靠其它汉字系统。汉字有繁简两种字库,输入方法有:双拼字词联想码、混含拼音码、形声码、方言代码、短语码、五笔字形码、区位码、电报码、英语码和德语码等。多国外文输入语种包括:日语、俄语、希腊语、德语、法语、意大利语、西班牙语和葡萄牙语等,而且还可以混合输入这些外文。

7. 手写汉字输入系统(HGD-9200-1)

软盘:3片(另含书写版、写字笔和专用卡)

定价:2000元

该系统通过接口卡和标准 RS-232C 串行接口与主机连接并启动后,即可用写字笔在书写板上手写汉字、绘图和制表等,实时地将汉字和图形信息输入计算机。凡是会写汉字的人,无需记忆任何编码,均可按手写汉字的习惯进行信息输入。

二、中华学习机及其兼容机软件

1. 电路实物连接图练习 软盘:1片 定价:15元

本软件用来练习连接中学物理课中常见的 56 种电路(简单电路、串联电路和并联电路),练习者可自选器材和电路接法。当练习者遇到困难时,机器能予以指导并绘出正确的连接图。本软件适合课堂教学演示和课外自学练习。

2. 美术字幕生成编辑系统 软盘:1片

定价:35元

本系统在超级汉字文章编辑系统 V3.0 支持下工作。它可将中华学习机硬汉字加工处理成空心、粗体、细体和立体美术字,字型可放大成 13 种不同的尺寸。经处理的美术字幕能生成独立的显示模块,存盘后可在其它系统下调用,还提供有打印字幕功能。

3. CEC-I 联想汉字输入系统 软盘:1片

定价:50元

本系统与拼音方式配合,具有多级联想方式,能自动转换。它有良好的扩展功能,用户可建立与系统字词库合并使用的专用联想词库,该词库在使用中享有优先权。

4. CEC-I 通用字词输入文章编辑系统

软盘:1片 定价:72元

该系统是中华学习机扩充汉字输入法的良好环境和编辑工具。用户可利用该系统定义五笔、双拼、声形等输入法,可建立自己的字、词输入法和联想字库,它配有五笔字形字、词库;用户可利用该系统象编制 BASIC 程序一样的方法来编写文章。

5. ML BASIC-3.0(程序设计语言) 软盘:2片

定价:75元

ML BASIC 语言是一种扩展 BASIC 语言,它比 CEC BASIC、APPLESOFT 多 78 个新语句,是软件设计的优选新语种。它可用于数值计算、符号、音乐与图形处理和程序自扩张方面的程序设计。它的图形功能不仅兼有 LOGO 语言的作图优点,甚至在某些方面还超过 LOGO 语言。

欲购软件,请汇款到:

北京万寿路 173 信箱(邮编:100036)

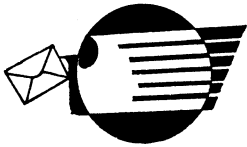
电子工业出版社软件部

联系人:谈众安 王旭

开户银行:北京工商银行翠微路分理处

户名:电子工业出版社杂志编辑部

帐号:8913333-59



读者联谊

普及型 PC 个人用户软件交流联谊活动

问题解答(一)

北京中国农科院计算中心(100081) 王路敬

自从九一年一月份《电子与电脑》杂志组织“普及型 PC 机个人用户软件交流联谊活动”以来,得到社会各界人士,尤其是《电子与电脑》读者的大力支持和充分肯定,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与广大用户直接联系,直接交流应用体会的机会,能为 PC 机个人用户获得自己需要的软件,提高用户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,我们先后收到来自全国各行各业 PC 机个人用户要求参加该项活动的已近 200 人,尤其《电子与电脑》今年第 7 期刊登第一批“常用交流软件清单”以来,收到大量用户来信。大家积极主动的提供了自己的软件清单;反映了希望通过联谊活动得到自己所缺乏的软件的愿望和要求;提出了在使用中碰到的困难和问题;对搞好联谊活动提出了殷切的希望。针对大家的来信、来函、来电,以及直接到我们这里联系的,我们已组织人员将有关材料进行了计算机管理。归纳大家的普遍要求:一是由于机器硬件资料缺乏,对其性能了解甚少,进行基本操作尚有一定困难。希望提供有关硬件方面的资料信息;二是由于缺乏软件,机器的功能不能很好的发挥,利用率低,希望交流交换软件;三是有了软件由于缺乏必要的使用性说明材料,致使软件不能得到利用,所以希望提供有关软件的使用说明或有关资料的来源;四是使用中碰到一些困难和问题,不能及时得到解决,借此活动交流使用经验、体会和实践中遇到的问题解决方法与途径。针对上述问题和大家的普遍要求,从今年第 1 期起在《电子与电脑》杂志上,以问答的形式,对普及型 PC 机从硬件系统,常用交流软件以及在使用中碰到的困难和问题,经过归纳、整理,奉献给参加 PC 机软件交流联谊会的用户以及 PC 机用户,以起抛砖引玉的作用,把我们这一“联谊活动”搞的更好,达到不断提高应用水平之目的。

PC 机硬件系统

1. 国内目前普及型 PC 机的代表机型有哪些?如何考虑 PC 机的配置问题?

继 IBM-PC 系列机、长城系列机以及中华学习系列机应用推广以来,在计算机知识日益普及,计算机已逐渐被家庭、被社会接受的时候,近年来在我国微机市场上又先后推出了大众化普及微型计算机。其代表产品如北京海华公司,航天 CAD 公司,北工大电子厂开发的海华—航天个人电脑 HH-PC,北京北方电脑公

司开发的 BFPC-BOY,四川的新潮 XC-PC,上海长江集团公司推出的东海 0520SD 小博士电脑,天津通博电子技术联合公司推出的 TH-MINI-PC,陕西省推出的长安 SUPER-PC 等都颇受用户的欢迎。这些机型与 IBM-PC 相比较运算速度更快,内存容量更大,系统扩充能力更强。可根据用户的不同需要进行组合配置,除少量集成电路外,大部分元器件均可在国内解决,维修方便。由于与 IBM-PC 系列和长城等系列兼容,所以系统软件,应用软件来源多而广,为发挥系统的功能提供了非常便利的条件。

这类微机适宜于各类学校、家庭、个人作为教学、学习和家庭事务处理使用,还可以作为企事业单位一般文字处理,信息管理等小型事务的微机处理以及计算机网络的结点机或多用户系统的智能终端机使用。

一台 PC 机系统组成应包括二大部分,硬件和软件。所谓硬件是指组成一台 PC 机所有固定装置的总称。主机,显示器,键盘打印机等都是系统的硬设备,统称 PC 机系统的硬件。主机包括微处理器和内存储器,微处理器是 PC 机的核心,它要完成数据的处理,加工和控制功能,区分 PC 机档次高低首先看 PC 机的微处理器性能。内存储器主要存储程序和数据,它们是通过键盘输入到系统的内存储器,然后再执行。显示器,打印机是 PC 机系统的输出设备,显示器显示 PC 机处理的信息,打印机在打印纸上打印处理的结果,包括字符,汉字,图表等。所谓 PC 机的软件是指提供 PC 机能够工作的各种程序的集合。PC 机系统的软件包括系统软件和应用软件两大类。系统软件是指使用管理 PC 机的软件。这一类软件包括磁盘操作系统软件,各种高级语言例如 BASIC 语言,dBASE I ,dBASE II 等编译或解释程序,各种服务性的程序,例如自检程序,诊断程序及工具软件 PC-TOOLS 等。应用软件是指根据 PC 机硬件资源、系统软件资源编制的解决具体问题的程序。使用 PC 机首先要了解和掌握硬件系统的配置有哪些,系统软件配置有哪些,在这种硬件和系统软件的环境下,能够运行的应用软件又有哪些,怎样进行系统扩展可以运行更多的应用软件等等。其次是掌握正确的操作使用方法。

以 HH-PC 和 BFPC-BOY 机为例,说明如何考虑 PC 机的配置问题。

HH-PC 型微机与 IBM-PC/XT 完全兼容,系统硬件基本配置:

CPU: Intel 8088-2

时钟主频: 4.77/10.7MHz

内存存储器: 256KB

软盘驱动器: 360KB×1

显示器: 12英寸高分辨率单色显示器, 分辨率720×350

键盘: 101键

主机接口: 显示器接口、电视机接口、软盘驱动器接口、光笔接口、并行打印机接口、RS-232异步通信接口、协处理器8087插座、PC总线扩展槽一个

硬件选配件:

黑白双频14英寸直角平面单色显示器, 分辨率720×350, 640×200, 高分辨率彩色显示器, 分辨率为640×350, 800×600(EGA), 1024×768(VGA)
内存扩展卡, 由256KB可扩展到640KB

5. 25英寸20M硬盘

3. 5英寸20M硬盘

5. 25英寸360KB软盘驱动器

打印机, 各类24针、9针打印机

CAD设备, 鼠标器, 光笔, 扫描仪, 数字化仪, 绘图

仪

16×16点阵汉字库

网络卡, 3+网卡, AT&T网卡, NOVELL网卡

软件基本配置:

PC-DOS(支持不同版本的DOS系统)

BFPC-BOY机

系统软件基本配置:

CPU: NEC V20与8086/8088兼容

8087插座: 配8087协处理器

主频: 4.77/10MHz

内存存储器: 256KB

软盘驱动器: 360KB×1

硬盘驱动器: 可选

软/硬卡: 软卡

单色显示器: 720×350/640×200

显示卡: MGP/CGA

接口: 并行口

键盘: 101键

主机板插槽: 标准的PC/XT插槽

电源: 功率100W

软件基本配置:

PC-DOS(支持不同版本的DOS系统)

MKF(图形转换软件)

HH-PC或BFPC-BOY主机采用Intel 8088(或V20)CPU, 这是一个16位的微处理器, 地址总线20条, 最大可寻址可达1MB字节, 主板内存可由256KB扩至512KB或640KB。

所配置的键盘为标准的PC/XT键盘, 其功能和使用方法与IBM-PC/XT完全相同, 其几个主要的组合控制键的功能如下:

Ctrl+Alt+Del: 重新启动系统(热启动)

Ctrl+Break: 终止当前执行中命令

Ctrl+NumLock: 暂停屏幕滚动显示

Ctrl+PrtSc: 将屏幕显示同时输出到打印机

Num Lock: 光标移动状态/数字状态转换

Caps Lock: 字符大小写转换

所配置的绿色显示器, 分辨率为720×350时, 文本状态下80×25行显示, 可进行西文信息处理。若玩游戏时, 首先要使用“图形转换软件”MDF或MKF将显示器的方式模拟成为图形转换方式。方法是先将“图形转换软件”盘插入软盘驱动器内, 然后执行MKF或MKB即可。即执行:

A>MKF

或 A>MKB

若使用CGA卡和640×200绿显时则可不用“图形转换软件”。

所配置的软盘驱动器及软磁盘与标准的IBM-PC机上使用的完全相同, 即5.25英寸双面双密度360K软盘。每张软盘分为40个磁道, 两面共80个磁道。每个磁道又分成9个扇区, 每个扇区内存放512个字节, 故每张软盘的密度为512×9×80=360640字节, 为360KB。

系统的安装与启动可以这样进行: 将主机放置在水平桌面上, 将键盘联于主机的键盘插座上, 显示器的信号线联于主机的显示卡端口上, 然后把主机及显示器的电源线分别联在交流电源插座上。

启动时将随机系统盘即DOS操作系统盘插入软盘驱动器中, 然后开机。开机时先开显示器电源(如有打印机, 再开打印机), 然后开主机电源。开机加电的过程称之为“冷启动”。关机时则顺序相反, 先关主机电源, 然后关显示器电源(如有打印机, 再关打印机电源)。当有非法键盘操作时, 有时会出现机器“死锁”, 这时可同时按Ctrl+Alt+Del键, 或按前面板上的“RESET”钮重新启动系统, 这一过程称之为“热启动”。

在HH-PC或BFPC-BOY等个人电脑上使用的操作系统DOS, 可以用低版本如DOS 2.0或2.10, 也可使用DOS 3.0以上的高版本, 但根据系统的硬件资源, 建议使用DOS 2.0或2.10就可以了, 不要追求高版本, 否则有些应用软件的使用会带来一些困难。

DOS为用户提供了一种工作环境, 它主要负责文件管理, 内存管理, 外设备管理和CPU管理, 而用户使用时则是透明的。

DOS的构造是复杂的, 主要层次为IBMBIO.COM(基本输入/输出系统), DOS核心IBMDOS.COM和命令处理程序COMMAND.COM。在DOS系统盘中我们只能见到COMMAND.COM, 而DOS的前两个层次IBMBIO.COM的IBMDOS.COM的接口是COMMAND.COM, 它接收用户从键盘上发出的命令并解释执行之。

DOS能够接收的用户命令分为三类:

(1)内部命令——驻留在内存中的如DIR、TYPE、COPY、ERASE、RENAME、CLS、MD、CD、RD等。

(2)外部命令——不驻留在内存中或称暂驻程序,

如 FORMAT, DISKCOPY, DISKCOMP, CHKDSK 等。

(3) 批处理命令——是由内部命令, 外部命令和批命令组成的文本文件, 是一类可执行文件, 其文件的扩展名为 .BAT。

当加电启动后, 机器的 ROMBIOS 开始执行自检和引导装入程序。ROM 引导装入程序从 DOS 系统盘上的第一扇区读入内存, 并由磁盘引导装入程序将两隐含的文件 IBMBIO.COM 和 IBMDOS.COM 读入内存, 然后分配磁盘缓冲区和文件控制块的地址空间, 最后装入 COMMAND.COM 的驻留部分, 便显示系统提示符 A>, 此时 DOS 进入正常工作状态, 等待用户的作业。

在系统内存容量基本配置 256KB, 360K 软盘驱动器的情况下作为家庭、学校进行智力开发, 语言启迪, 游戏等是完全胜任的, 若要要进行家庭事务处理、企事业单位一般的文字处理、信息管理等工作要增加汉字处理的功能。HH-PC 或 BFPC-BOY 或兼容机实现汉字处理功能有以下几种途径:

(1) 扩充专用汉卡。这类机器系统主板上都有一个或几个扩展插槽, 将汉卡直接插入扩展槽即可, 专用汉卡不占主机内存。这样相当于扩充了内存的容量。例如陕西省计算机公司推出的长安 SUPER 汉卡, 该汉卡采用了 7 个 4MBIT 芯片制作 48 点阵宋体、仿宋、黑体、楷体精密字库和 10 种 128 点阵西文字库。它支持 24 针打印机, 支持多种的汉字输入方法, 编辑打印集于一体, 兼容 WS 和 MS2401 功能, 全屏多窗口, 弹出选择, 系统帮助, 一看就懂, 一学就会。汉字字体美观, 文字可任意大小, 而且具有在屏幕上模拟打印输出功能, 使用起来很方便。航天汉卡, 王码高速字符型汉卡都具有类似的功能。

(2) 将主机内存由 256KB 扩充到 512KB 或 640KB 后即可使用市面上流行的中文操作系统 CC-DOS。例如 CC-DOS 2.0/2.1, 或 CC-DOS 4.0。在中文操作系统支持下可运行 IBM-PC/XT, 长城 0520A 型机的汉字软件。例如 WS, dBASE II, dBASE III, 五笔字型以及高级语言 BASIC、FORTRAN、COBOL 等软件。

(3) 利用压缩字库 CC-DOS 可以在基本配置即 256KB 内存的 HH-PC 或 BFPC-BOY 或兼容机上实现汉字处理功能。但是由于基本内存太小, 仅能做一般简单的文字处理, 象 dBASE III, 五笔字型等软件仍不能运行。

若将内存扩充至 512KB 或 640KB, 扩充一个 20MB 的硬盘, 其功能相当于 IBM-PC/XT, 所有在 IBM-PC/XT 机上运行的软件均可在个人电脑上运行。

无论采用上述何种方法实现汉字功能, 其汉字输入方法和操作都是相同的或类似的, 区别仅在于实现汉字的形式和途径不同。

个人电脑上启动 CC-DOS 进入中文状态后功能键的定义如下:

ALT+F1 选择区位码输入方式
ALT+F2 选择首尾码输入方式

ALT+F3 选择拼音输入方式
ALT+F4 选择快速输入方式
ALT+F6 选择 ASCII 码输入方式即英文输入方式
CTRL+F7 纯西文/纯中文方式转换
CTRL+F8 建立自动光标/取消自动光标转换
CTRL+F9 建立纯中文方式/取消纯中文方式转换
CTRL+F10 选择打印机字型和行宽方式

对个人电脑连有打印机的使用者, 在输出汉字时可在 CC-DOS 状态下挂打印机驱动程序。不同类型的打印机使用其相应的打印驱动程序。运行驱动程序的方式有两种: 一种在 CC-DOS 启动后, 当提示符出现, 在提示符后从键盘上打入相应打印机驱动程序的文件名并回车即可。另一种方法就是驱动程序的自动引导。也就是说打印驱动程序放在 AUTOEXEC.BAT 自动执行批处理文件中, 这样在 CC-DOS 启动的时候同时自动执行批处理文件的命令, 从而达到驱动程序的自动引导。但是有一点需要注意, 打印驱动程序要在 CC-DOS 的系统盘上。建立自动执行批处理文件方法可以利用 COPY CON: 命令, 也可用 EDLIN 行编辑程序或其他编辑软件。例如在 CC-DOS 2.0 系统盘上建立 AUTOEXEC.BAT 文件, 所用打印机是 3070, 其操作如下:

(1) 把 CC-DOS 系统盘插入软盘驱动器 A 中, 并启动;

(2) 待系统提示符 A> 出现后从键盘打入如下信息:

```
A>COPY CON:AUTOEXEC.BAT
ECHO OFF
CLS
FILE1
CCCC
ALL24P
^Z
```

这样 AUTOEXEC.BAT 就建好了, 再重新启动系统即可达到打印驱动程序的自动引导。

在 CC-DOS 状态下使用 CTRL+F10 可以选择打印字型和行宽。能变换多少种字型取决于驱动程序的功能。可以使用 SHIFT+PrtSc 拷贝屏上的内容, 在打印机上输出要打印的内容。如果要打印一个文本文件的内容, 可以使用 TYPE 命令将显示的信息送到打印机, 其操作如下:

```
A>TYPE 文件名 > PRN
```

普及型 PC 个人电脑是性能价格比较好的微机系统, 可以根据工作的需要, 在基本配置的基础上适当的组合配置, 以满足不同层次的需要, 运行各种流行中西文软件, 或开发自己满意的软件产品。

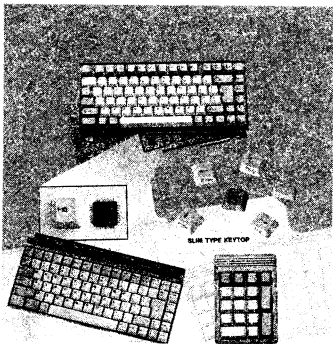
(待续)



ISSN 1000-1071

電子與電腦

2



一九九二年

总期第83期

電子與電腦

* ELECTRONICS AND COMPUTERS *

目 录

· 综述 ·

我国单片机应用技术发展趋势及展望(续)
..... 何立民(2)

· PC 用户 ·

彩色汉字通用下拉式窗口菜单的设计与实现
..... 王宏(4)

dBASE III 或 Foxbase 菜单式自由组合查询
..... 毛维平(9)

单显 PC 机的应用技巧 袁津生(10)

BASIC 程序用 P 存盘的解密方法 禡协贤(11)

谈谈编译 BASIC 在长城机上的运用 卢祥江(11)

也谈解决运行 FoxBASE 内存不够的问题
..... 周立宇(12)

· 学习机之友 ·

虚拟 DOS 的改进及其在通讯网中的应用
..... 石水琳(13)

磁盘数据的压缩存储技巧 陈庆祥(14)

为 Apple DOS 创造三条实用命令 王志超(17)

也谈 CEC-1 多功能造字 覃 敏(18)

ProDOS 磁盘操作系统入门(续) 廖 凯(18)

· 6502 机器语言讲座 ·

第二章 微处理器、存储器、寄存器简介
..... 朱国江(20)

· 初级程序员级软件水平考试辅导 ·

计算机硬件基础知识(续) 顾育麒(24)

· 学用单片机 ·

低功耗单片机最小系统 张培仁 刘振安(29)

新一代超高集成度 Z80 微处理器系列 董伯明(31)

· 学装微电脑 ·

打印数字温度计 易齐干(32)

· 电脑巧开发 ·

单板机 RAM 分段与程序保护的实现 王新明(37)

· 电脑游戏机 ·

第三章 F BASIC 的画面控制语句 于 春(41)

· 维修经验谈 ·

PC-1500 计算机的维修 何 静(44)

导电橡胶按钮被磨损的修理方法 梁绍建(44)

IBM-PC 机故障维修一例 欧阳波(44)

· 新书与软件 ·

DATA BASE IV 关系数据库 张冰毅(45)

· 读者联谊 ·

普及型 PC 个人用户软件交流联谊活动问题解答(二)
..... 王路敬(46)

显示器·显示卡·显示系统 孙梅英(48)

封面 键盘

封二 普及型 PC 机用户联谊会常用交流软件清单

封三 Z80 系列微处理器性能表

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

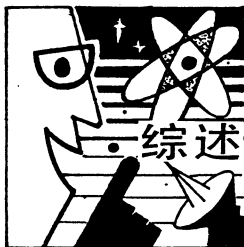
订购处:全国各地邮电局

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



我国单片机应用技术 发展趋势及展望(续)

北京航空航天大学 706 教研室(100083) 何立民

3. 外围器件

单片机应用系统的应用开发难度会逐渐转向外部接口,因此对外部接口的新型器件的新发展、市场供货状况要密切注视。

在前向通道中,小信号、高精度、长距离传送一直是通道接口设计的难点。模拟器件价格居高不下,致使一些非快速系统转而求助于纯数字系统,相应地推动了数字化传感器、V/F 转换接口的发展。在不得不使用模拟电路的前向通道接口芯片供货状态已大为改善,市面上已有较多的小信号放大、隔离放大、可编程增益放大、电压电流转换芯片可供选择。但从根本上解决模拟小信号采集时接口通道的高精度,抗干扰只能求助于集成调理器模块。例如可用作高精度、自带激励电压源电流源的小信号放大模块 2B30/2B31。

人机通道接口中语音接口芯片引人注目,其器件价格及供货状态已能供大面积推广使用。典型的语音芯片有两类:一类兼有语音分析与语音合成功能的 UM5100/5101、T6668;另一类只有语音合成功能,但能以很低比特率合成高质量语音的 TMS5220。这些语音芯片的价格都在百元以下。

在 RS232C 接口中用 MAX232(或 ICL232)芯片替代 1488/1489 或简化接口电路,而且可省去±12V 供电电源,MAX232 芯片中有两对 RS232C 收/发电平转换接口。

反向通道中,国产带控制端的 8 路驱动芯片 BIC87 系列正在大面积推广;高压大电流 TWH8751 已成功地构成了步进电机驱动电路;国产交直流固态继电器已进入实用阶段,由于它和 TTL、HTL、CMOS 电路的兼容性,在单片机应用系统中广泛采用;现代电力电子器件如 VMOS 功率器件的引入为后向通道直接驱动大功率电路开辟了良好的前景。

为了抑制电源投切、瞬时短路、压降以及电网串入干扰脉冲造成单片机系统的失灵,在一些系统中使用了电源监视用芯片 TL7705/7700。7705 为 5V 电压监视,在电源上电时能产生可靠的复位信号,能在 500ns 内检测出电源电压降低到 4.5~4.6V 的故障现象,TL7700 的监视电压可由用户设定,故还可对温度、湿度、风量、压力等参量电压进行监测。

在某些应用系统中需要高性能实时时钟时可采用 MSM5832RS(5V 供电)或 ICM7170(3V 电池供电)。

三、单片机应用的开发环境

上海复旦大学推出了 SICE 系列开发装置已形成

我国单片机开发装置的主流机型,对我国单片机开发应用起了重要的推动作用。但制造厂家过多,质量差异较大;用户无法了解监控程序文本,造成使用中的不便。随着用户应用水平的提高,要求介入监控程序的呼声越来越高,在经过了六个年头开发历史的今天,用户对开发装置监控程序仍然一无所知,必然会影响应用开发水平的提高,也不利于监控程序的完善与提高。因此,一些有识之士已开始独立研制开发了一些单片单板机、仿真器产品,其中部分产品的软件文本、硬件电路向用户公布。较早的有武汉大学无线电信息工程系研制的 SCB-1 型单片单板机,中国科技大学计算中心研制的 KDC-Ⅲ 型开发应用系统,均在《电子与电脑》杂志上公布了监控程序文本和硬件电路。

新器件的出现有利于改善开发装置的硬件电路,如大容量存储器的使用使开发装置少占用户资源,换体技术易实现零地址编程;可编程逻辑器件的使用可简化电路并实现一些特殊逻辑功能;E²PROM 及 SRAM 的可靠掉电保护的使用大大地方便了单键盘调试。为了提高开发模板与系统主机的联机通信的可靠性,开始废除用分立元件构成的准 RS323C 接口,代之以 1488、1489 的标准 RS232 接口,甚至采用单 5V 供电的标准 RS232C 接口芯片 MAX232(或 ICL232)。

在软件配置上,目前国内廉价的开发装置普遍都配备有主机用的组合软件,一些较高档的开发装置已配备有 C51/96、PLM51/96 开发软件。

四、单片机应用系统设计的硬件技术

单片机应用系统硬件设计时应充分调查新器件新技术的进展,把一些行之有效的电路技术和器件及时反映到系统中。

在系统的存储器部分由于大容量芯片出现而普遍趋向单片系统,数据存储器采用可靠的掉电保护,早期用夹持电池方法已淘汰而改用焊接腿可充电电池,并加有防改写电路,较先进的结构形式则采用掉电保护 SRAM 集成模块。

抗干扰设计成为系统设计中不可缺少的部分,如电源滤波、电源监测,可靠地系统复位设置、Watchdog 电路、使用 CMOS 芯片等。使用 CMOS 单片机不仅本身具有较好的抗干扰能力,而且它的低功耗运行方式为软件中利用“睡眠抗干扰”创造了条件。

语音接口将会在单片机应用系统中广泛使用,具有语音合成、语音分析功能的语音芯片要求存储容量大,并有语言录入部分,故接口电路较复杂,如 T6668

语音接口。但是在许多智能仪表、监测系统、机电一体化系统中,语音词条有限,不必要求接口有语音数据库功能,这时采用仅有语音合成功能的芯片如 TMS5220 较好,该芯片的语音数据压缩技术可以大量节约内存容量,可做到一个语音音节只占用 100 个字节左右(约为 T6688 接口的十分之一),在有限词条语音接口中可不单独设存储器,而采用加大单片机数据存储器容量来放置语音数据的办法。这种语音接口十分简单,但要求有现成的语音数据。目前,四川大学计算中心已有供 TMS5220 使用的语音数据库。

硬件电路的标准化、系列化、模块化是人们关心的议题。但由于新器件的发展、应用对象的复杂性,这些工作尚未提到日程上来。可以预计,其发展过程是首先优化单片机系统电路,如存储器配置、掉电保护、系统时钟、复位等,随后优化外围接口并不断筛选、定型、逐渐将难度向外扩展,先提供一部分成熟电路,不必进行试验即可一步制成印刷电路板,部分电路的 ASIC 化有利于系统电路的标准化进程。

五、单片机应用程序设计技术

目前我国单片机应用系统研究人员大多数是非计算机专业人员,对软件设计比硬件系统设计更为陌生,对应用程序设计的要求、内容、作用认识较为粗浅。因此我国单片机应用程序设计尚处于初级阶段,表现在

以功能设计为主要内容,以指令系统为基础的随意性逻辑设计方法。虽然不少应用程序设计中引入了抗干扰设计内容,也未有系统的论述。因此,认真研究单片机应用程序设计方法,正确拟定应用程序设计内容,对提高单片机应用技术水平有极其重要的作用。

应用程序设计所选择的语言与系统的应用对象、开发环境、软件包的容量有关。单片机的具体应用对象及应用环境决定了汇编语言仍是单片机应用程序设计的基本语言;对程序量较大的监控程序可采用 C51/96、PLM51/96 以提高效率,减少编程错误。

要不断总结应用程序设计方法。单片机应用系统的多样性不可能攀求一种全通用的方法;但目前可以逐步推行一些行之有效的设计方法。如实时多任务操作系统设计方法、智能仪表中监控程序的状态分析法和图解设计法。

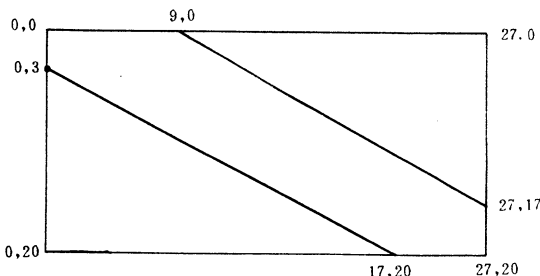
要正确拟定应用程序设计的内容与要求,改变以功能实现为唯一目的的设计思想,明确应用程序设计中功能实现、容错设计、抗干扰设计是三位一体不可缺少的内容。要保证应用系统可靠地运行,应用程序完成功能实现只是开始,只有具备抗干扰能力、容错能力的应用程序才有可能在实际环境中使用。

(编者按:本文摘自“1991 年全国单片机学术交流会议论文集”,本刊略有增删。)

(上接 41 页)

```
30 F. X=0 TO 27 ST. 4
40 Y=Y+1
50 DE. SP. Y, (0, 1, 0, 0, 0) = CH. (X+28) + CH.
    (X+29) + CH. (X+30) + CH. (X+31)
60 SP. Y, Y * 25 + 40, Y * 20 + 40
70 N.
80 LOC. 0, 22
```

RUN 后,返回 BG 画面。这时可按 BG GRAPHIC 方法选定一种或几种背景图形画一方框,本例以最外边



缘画框,并在框内加两条斜线,画面如下:

然后键入 ESC 和 STOP 返回 BASIC 后,把 10 行中的 BGTOOL 改为 VIEW, RUN 后,则显示丽莎在方框斜线包围的六边形中。

3. BG、SP 画面的配合

读者已经发现, BG 画面的 X 取值为 0~27, Y 取值为 0~20, 而 SP 画面的 X 取值一般为 0~240, Y 取值为 0~220。两个画面中的值域对应关系见随机手册附图。由于取值区域的差异,两个画面的座标必须通过换算才能正确配合。其座标换算方程。

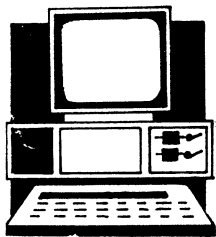
$$\begin{cases} x = X * 8 + 16 \\ y = Y * 8 + 24 \end{cases}$$

式中大写 X, Y 为 BG 画面座标,在 BG 画面左下角显示。小写字母 x, y 为 SP 画面座标。使用方程(1)可在设计 SP 显示座标时进行计算。有时已知 SP 的座标,要根据 SP 的座标设计背景图案,则可使用方程(2)进行换算。

$$\begin{cases} X = x / 8 - 2 \\ Y = y / 8 - 3 \end{cases} \quad (2) \quad (\text{待续})$$

(上接第 12 页)

8683520	bytes available on disk	21309440	bytes total disk space
655360	bytes total memory	55296	bytes in 2 hidden files
525456	bytes free	43008	bytes in 10 directories
C: \>cd\fox		12527616	bytes in 994 user files
C: \fox>mfoxplus		8683520	bytes available on disk
• ! \DOS\CHKDSK C:		655360	bytes total memory
		150704	bytes free



PC 用户

彩色汉字通用下拉式窗口菜单的设计与实现

邮电部成都电缆厂科研所(610042) 王宏

下拉式窗口菜单是时下流行的一种用户接口方式。许多西文软件都采用了多彩色的窗口菜单技术,提供集成化用户环境,使用户操作非常方便。它们具有新的风格,如选择菜单功能时采用光棒定位,以回车键或数字(或菜单内容首字母)选择相结合的方式。这种方法使用户感到直观、明快、不易产生误操作。多层次菜单的进入和退出也由原来逐层清屏、显示下层菜单的方法,变为保留主菜单画面,子菜单采用下拉式结构,层次分明,用户不致于因进层太多而如入迷宫。

本文以汉化 Clipper Summer'87 为工具,在 25 行汉字系统下如何实现彩色汉字下拉式菜单窗口问题加以阐述并提供完整的源程序。它不但可以通用于任何管理系统,而且兼管整个管理系统的所有菜单分支及分支选择处理。具有良好的通用性和可维护性。

1. 窗口菜单设计的基本原理

窗口一般是在屏幕上开辟一个矩形,它可以根据命令打开和关闭。当一个窗口打开时,它隐藏了将被覆盖的屏幕内容,包括与其它窗口重叠的部分,正文或图画输出只对窗口内部有效,对窗口外的任何输出都是无效的。窗口关闭时,则要清除它在屏幕上的信息。窗口菜单是窗口用于菜单的一种具体形式。实现窗口菜单流程如图 1 所示。它反映了窗口菜单建立的基本思想和主要步骤。

2. 彩色汉字下拉式窗口菜单的实现

① 窗口菜单状态数据的数据结构

一般的菜单都是多层次的。最先出现的菜单往往最后关闭,而且每一时刻,处于重叠式窗口最上面的最新开辟的窗口为活跃窗,菜单选择只能在活跃窗内进行。为了实现这样一种先进后出的存储信息要求,本文采用了菜单库存放有关窗口位置、色彩等状态数据,使程序中不存有菜单内容,而把所有的菜单内容连同功能模块等参数事先存入一个反映各功能模块间关系的菜单库中。菜单库描述信息见图 2。

这样,建立系统窗口菜单状态等参数工作,实际上变成了让用户向菜单库中输入菜单内容等参数(可用程序自动实现),然后按一定规律将这些菜单内容连接起来,显示在屏幕相应位置即可。这样菜单的显示及选择处理过程就是遍历树的过程,其程序框图见图 3。

② 窗口菜单的覆盖部分屏幕信息的保存与恢复

Clipper Summer'87 提供了两个函数,可将屏幕信息保存到内存变量或从内存变量中恢复,这样使窗口

菜单的覆盖部分的屏幕区域信息的保存与恢复变得轻而易举了。

储存屏幕函数为 Savescreen(),其格式为:

```
Savescreen(<expN1>,<expN2>,<expN3>,<expN4>)
```

其中:<expN1...expN4>是所要储存屏幕区域的坐标位置。系统返回一个字符串用于保存所指定的屏幕区域,最多保存 4000 个字节。

恢复屏幕区域函数是 Restscreen(),其格式:

```
Restscreen (<expN1>,<expN2>,<expN3>,<expN4>,<expC>)
```

它用于将事先用 Savescreen()函数保存屏幕区域的信息从内存变量中恢复到指定屏幕区域。恢复的屏幕位置可与原先储存的原始位置相同或不同。若是指定一个新的屏幕位置恢复时,应确保此位置与原储存位置有相同的大小,否则将会得到不明确的结果。而且不要使用 Restore screen 命令恢复用 Savescreen()储存的屏幕区域,否则也会得到不明确的结果。另外要注意,Savescreen()与 Restscreen()均不支持使用 ANSI·OBJ 终端机支援的电脑或应用程序。也就是说,在 DOS 的 CONFIG. SYS 文件中不应有语句:

```
DEVICE=ANSI·SYS
```

③ 窗口菜单的显示及选择

在进行用户界面设计时,一般都是将主菜单放在屏幕最上行或最下行,这样使整个屏幕画面空间集中。主菜单采用固定方式显示在屏幕上,用左右键头移动光棒定位,按回车键或数字键(或首字母)选择。子菜单的选择则依赖菜单库中描述参数,根据主菜单选择的某一功能,从菜单库中提取相应参数,显示出下一级子菜单。菜单布局采用下拉式显示(或重叠子菜单,依菜单库中描述参数而定),见图 4,图中位于主菜单块一的下面有一矩形框,它是当前第一个菜单内容被选中后出现的下一级菜单,当功能完成返回本菜单时,这时再按 Esc 键返回上一级菜单,同时该子菜单窗口消失。每选中一个菜单(若没到叶结点),就在其下方出现下一级菜单的窗口,它就是所谓“下拉式菜单”。

由于子菜单窗口是随主菜单中各功能的位置不同而出现在屏幕的不同地方,且各自的下一级菜单内容条目都不相同,而控制光棒移动却与行列起始位置及范围密切相关,故对每一个不同位置的下拉式菜单,均要编写不同的程序来实现。本文采用了统一的显示位置,即不管主菜单选择某功能,下拉菜单均从屏幕最左

边显示开始,并根据菜单库中描述参数,采用程序递归设计技术解决上述因位置和条目变化引起的困难。这是个非常方便实用的菜单选择程序。

为了使菜单显得醒目,光棒所在位置采用了色彩显示,光棒通过↑、↓键移动,且可实现自动翻转功能(本文介绍的程序未做,用户可根据需要用 Clipper 提供的函数 Scroll()实现),按 Esc 键则返回。由于 Clipper 不支持类似 dBASE III plus 中的 Return to Master 指令,本程序采用了将键值填入键盘缓冲区的方法来模拟 Return to Master,使用户也可从最下级菜单直接返回主菜单画面。菜单选择后,其状态保留,便于操作人员记忆操作流程。整个菜单选择给人一种不疲劳之感。窗口宽度随菜单项中最大长度项而变,使得菜单窗口实际上是个可自动伸缩的窗口。

以上详细介绍了利用汉化 Clipper 实现的彩色窗口菜单方法。该程序具有良好的通用性和可维护性。对

任意层次结构的应用管理系统,只要填好菜单库中描述参数,主菜单程序中提示作下修改,系统菜单内容,功能模块扩充、删除、修改等均不需对程序作任何改动,只改动菜单库内容等参数即可。在系统维护模块中,也可加上对菜单库的修改功能。笔者用本文方法实现的彩色窗口下拉式菜单程序已经在“全国职业卫生信息管理系统”中运用,经在配有 EGA 卡的 AST386 上使用效果良好。本程序稍作改动也可在 dBASE III plus、FoxBASE+ 上实现。本程序用 Clipper 提供的编译、连接程序生成了可执行文件,在 DOS 下即可执行。

数据库结构: C:\CLIPPER\MENU.DBF
 数据记录数: 589
 最新更改日期: 07/31/91

字段	字段名	类型	宽度	注释
1	MENU_NAME	C	20	菜单项
2	MENU_MARK	C	8	菜单标记
3	MENU_SUB	C	8	下级菜单标记或功能程序名
4	NUMBER	C	2	同层菜单序号
5	UP_MENU	C	8	上级菜单标记
6	XROW	N	2	下拉菜单行坐标
7	YCOL	N	2	下拉菜单列坐标
8	COLOR	C	1	下拉菜单色彩
9	NOTE	C	40	下拉菜单条目注释

图 2

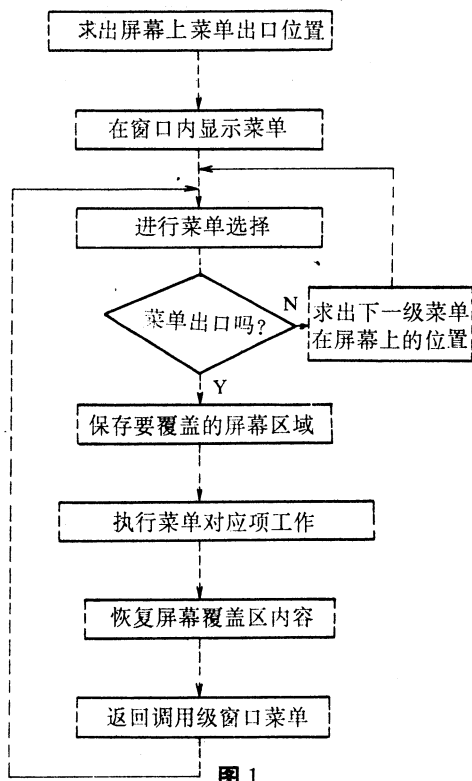


图 1

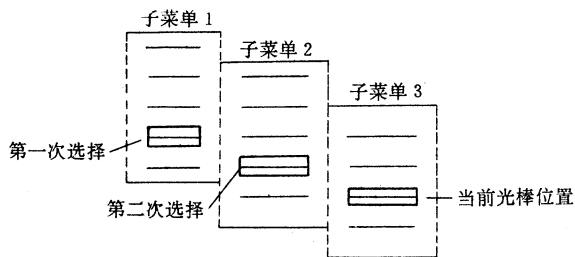


图 4

示例库

记录号	MENU_NAME	MENU_MARK	MENU_SUB	NUMBER	UP_MENU	XROW	YCOL	COLOR	NOTE
1	图形查询与录入	29		1	1	5	1	2	图形查询与录入
2	图形生成	29		2					图形生成
3	简史及一般情况	29		3					厂(矿)简史及一般情况查询
4	毒害种类及构成	29		4					厂(矿)毒害种类及构成查询
5	构成及分布	29		5					厂(矿)部门构成及毒害分布查询
6	个人档案	29	35	6					职工职业卫生个人档案查询
7	测定情况	29		7					厂(矿)毒害测定情况查询
8	体检情况	29		8					厂(矿)职工体检情况查询
9	历史资料	29		9					职工职业卫生历史资料查询

10	一般情况及各种史	35	96	1	29	6	18	3	职工一般情况及各种史查询
11	事故及各种疾病	35		2					事故及各种疾病查询
12	姓名	36	MODEL1	1	96	8	64	6	按姓名查询
13	年龄	36	MODEL2	2					按年龄查询
14	性别	36	MODEL3	3					按性别查询
15	民族	36	MODEL4	4					按民族查询
16	技术职称	36	MODEL5	5					按技术职称查询
17	任意条件	36	MODEL6	6					按任意条件查询
18	浏览	36	MODEL7	7					浏览全部
19	一般情况及嗜好史	96	36	1	9	7	46	5	一般情况及个人嗜好史查询
20	既往病史	96		2					既往病史查询
21	配偶史	96		3					配偶史查询
22	家族疾病史	96		4					家族疾病史查询
23	月经生育史	96		5					月经生育史查询
24	职业史	96		6					职业史查询

* MAIN_MENU.PRG
 * 功能:主菜单程序
 * 调用文件:SUB_MENU.PRG

CLEAR ALL

SET KEY-1 TO K_BOARD && 设定按 F2 键后执行的程序

SET SAFE OFF

SET DELI TO ' '

SET DELE ON

SET DECI TO 4

SET DEVI TO SCRE

SET SCOR OFF

SET STAT OFF

SET WRAP ON

SET CURSOR OFF

CLOSE DATA

SET CONS ON

PUBLIC LEVEL,V_N,W_RETU

SET COLO TO G+/N

CLEAR

V_FU1='1. 录入'

V_FU2='2. 查询'

V_FU3='3. 统计汇总'

V_FU4='4. 图表输出'

V_FU5='5. 维护通讯'

V_FU6='6. 综合评价'

V_FU7='7. 辅助功能'

DIS1='系统数据库的数据输入'

DIS2='查询系统数据库中数据'

DIS3='系统报表的预处理'

DIS4='报表打印及屏幕上图示显示'

DIS5='系统维护及与其它系统的接口'

DIS6='厂(矿)工业卫生的综合评价'

DIS7='做系统辅助工作'

TS1='再见! 欢迎下次使用!'

TS2='注意:'

TS3='关机顺序:先关主机,后关外设'

TS4='开机顺序同关机相反.'

TS5='→和←键:移动光带,ESC 键:退出系统,ENTER 键或数字:选中,其它键无效'

TS6='↑或↓键:移动光带,ENTER 键:选中,HOME 或 END 键:第一个或最后一个光带,按 F2 键回主菜单'

SET COLO TO W+/N

CLEAR

SELE1

* 打开菜单库

USE MENU

* 保存屏幕区域信息

WINBUFF=SAVESCREEN(0,0,24,79)

DO WHIL . T.

V_N=1 && 设定菜单层数

W_RETU=. T. && 判定用户按 F2 键否

SET COLO TO GR+/B

@ 0,0,2,79 BOX '+-+|+-+|'

@ 2,0,4,79 BOX '+-+|+-+|'

@ 4,0,21,79 BOX '+-+|+-+|'

@ 21,0,23,79 BOX '+-+|+-+|'

@ 5,1 CLEA TO 20,78

SET COLO TO G+/B,N+/R,BG

@ 1,1 CLEA TO 1,78

@ 1,CENTER(TS5) SAY TS5

SET COLO TO R+/B

@ 22,5 SAY '工业卫生微机管理系统'

@ 22,50 SAY '研制:中华人民共和国卫生部'

SET COLO TO G+/B

@ 22,29 SAY '|| 版权所有 ||'

SET COLO TO GR+/N

@ 3,1 CLEA TO 3,78

&& 菜单条目注释显示在第 24 行且居屏幕中央

SET MESS TO 24 CENTER

@ 3,1 PROM V_FU1 MESS DIS1

@ 3,10 PROM V_FU2 MESS DIS2

@ 3,20 PROM V_FU3 MESS DIS3

@ 3,33 PROM V_FU4 MESS DIS4

@ 3,45 PROM V_FU5 MESS DIS5

```

@ 3,57 PROM V_FU6 MESS DIS6
@ 3,69 PROM V_FU7 MESS DIS7
&& 设定画面执行目前 PROMPT 的光棒,将选择结果
存入 MENUCHOICE
MENU TO MENUCHOICE
DO CASE
CASE MENUCHOICE=1
LEVEL='2'
CASE MENUCHOICE=2
LEVEL='29'
CASE MENUCHOICE=3
LEVEL='13'
CASE MENUCHOICE=4
LEVEL='78'
CASE MENUCHOICE=5
LEVEL='83'
CASE MENUCHOICE=6
LEVEL='123'
CASE MENUCHOICE=7
LEVEL='124'
CASE MENUCHOICE=0
SET COLO TO BG+/G
@ 7,30 CLEA TO 9,50
V_RE='N'
@ 8,35 SAY '退出否? (Y/N)'GET V_RE PICT'
X'
READ
IF UPPE(V_RE)='Y'
CLEAR
SET COLO TO
SET COLO TO GR+/G
@ 5,CENTER(TS1) SAY TS1
@ 8,CENTER(TS2) SAY TS2
@ 11,CENTER(TS3) SAY TS3
@ 14,CENTER(TS4) SAY TS4
CLOSE ALL
QUIT
ELSE && 恢复屏幕区域信息并循环
RESTSCREEN(0,0,24,79,WINBUFF)
LOOP
ENDI
OTHE
RESTSCREEN(0,0,24,79,WINBUFF)
LOOP
ENDC
SET MESS TO
@ 1,1 CLEA TO 1,78
IF LEN(TRIM(LEVEL))<()
DO SUB_MENU
ENDIF
RESTSCREEN(0,0,24,79,WINBUFF)
ENDDO
RETU

```

```

* 文件名:SUB_MENU.PRG
* 功能:子菜单程序
* 调用文件:SUB_MENU.PRG
PROC SUB_MENU
PRIV MEN_N
MEN_N=LEVEL
SET COLO TO GR+/B
DO WHIL .T.
GO TOP
LOCA ALL FOR MENU_MARK=MEN_N
IF EOF()
SELE 10
WINBUFF1=SAVESCREEN(5,2,20,78)
DO &&LEVEL && 调功能程序执行
@ 5,1 CLEAR TO 20,78
SELE 1
MEN_N=CA_NAME
RESTSCREEN(5,2,20,78,M-WINBUFF1)
V_N=V_N+1
V_M=STR(V_N,1)
K=K&V_M
LOOP
ELSE
V_ROW=XROW && 下拉菜单屏幕行坐标
V_COL=YCOL && 下拉菜单屏幕列坐标
V_TEMP=V_ROW
V_LEN=0 && 初始化同层菜单项最大项长度
MENU_NUM=1 && 初始化同层菜单数目
CO=COLOR && 下拉菜单色彩
UP_NAME=UP_MENU && 保存上层菜单标记
DO WHIL MENU_MARK=MEN_N
V_LEN=MAX(LEN(TRIM(MENU_NAME)),V_LEN)
IF MENU_NUM<0
M=STR(MENU_NUM,1)
ELSE
M=STR(MENU_NUM,2)
ENDIF
M_NAME&M=TRIM(MENU_NAME)
MENU_NUM=MENU_NUM+1
INFO&M=TRIM(NOTE)
SKIP
ENDDO
MENU_NUM=MENU_NUM-1
SET COLO TO GR+/&CO
@ V_ROW,V_COL CLEA TO V_ROW+MENU_NUM+1,V_COL+V_LEN+1
@ V_ROW,V_COL,V_ROW+MENU_NUM+

```

```

1,V_COL+V_LEN+1 BOX '+-+|+-+'
SET MESS TO 24
@ 24,0 CLEA TO 24,79
N=1
DO WHIL N<=MENU_NUM && 显示菜单
  IF N<10
    M=STR(N,1)
  ELSE
    M=STR(N,2)
  ENDIF
  @ V_ROW + 1, V_COL + 1 PROM M _
  NAME&M MESS INFO&M
  N=N+1
  V_ROW=V_ROW+1
ENDDO
@ 1,CENTER(TS6) SAY TS6
@ 24,63 SAY '[ESC]返回上级菜单'
MENU TO K
GO TOP
LOCA FOR MENU_MARK=MEN_N.AND.NUM-
BER=LTRIM(STR(K))
LEVEL=MENU_SUB && 下级菜单标记或功能
程序名
IF LEVEL+' '<>' '.AND.K<>0 && 判
断叶结

```

```

点。
CA_NAME=MENU_MARK && 保存本层菜
单标记
V_M=STR(V_N,1)
K&V_M=K && 保存本层菜单选择光棒位置
RELEASE K
V_N=V_N+1
DO SUB_MENU && 递归调用
ENDIF
MEN_N=UP_NAME
IF MEN_N='1 ' && 若第一级下拉子
菜单返主画面
  @ 24,0 CLEA TO 24,79
  RETU
ELSE && 返上级菜单
  V_N=V_N-1
  V_M=STR(V_N,1)
  K=K&V_M
  SET COLO TO G+/B,N+/R,BG
  @ V_TEMP,V_COL CLEA TO V_ROW+1,V
  _LEN+1
  IF W_RETU=.F..AND.V_N< >0
    KEYBOARD CHR(27)

```

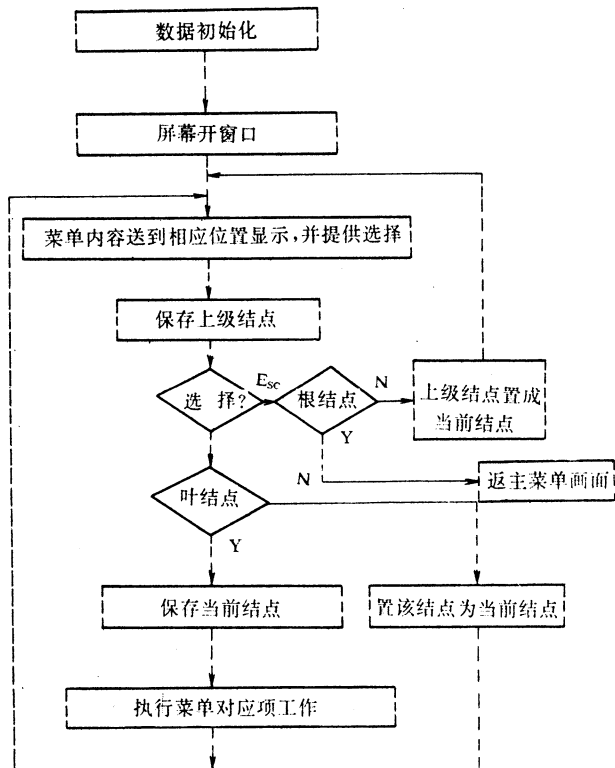


图 3

(下转第 15 页)

dBASE III 或 FoxBASE 菜单式自由组合查询

广东佛山绣品总厂(528000)毛维平

查询是所有管理系统不可缺少的程序之一。一般用 dBASE III 或 FoxBASE 编程时都是根据用户要求编写出各种条件的查询程序,但是当系统运行一段时间后,用户总有一些新的意想不到的查询要求提出,这样只得经常退出系统回到点状态开库查询。这对于非软件人员来说有一定的困难,同时整个系统又显得不完善,笔者为解决这个问题编了一个菜单式自由组合查询程序,实践证明效果良好。

大体做法是把某个库的可查询字段用菜单的形式显示在屏幕上,同时把一些运算符、逻辑符也显示在屏幕上,用户可根据提示随意进行字段、运算符的各种组合,编辑成各种查询条件,查出所需记录。

程序说明:数据库 ABC 及各字段仅为说明该方法而设,为方便阅读,字段各用汉字表示,考虑宏替换方便有规律的 a₁~a₁₂代替各字段名及运算符,其中 a₁~a₄ 为数据库可查询的字段名, a₅~a₁₂ 为运算符和逻辑字符;用 INKEY() 函数控制整个操作过程(该函数使按键不显示在屏幕上且又可得到其键值),程序中所用控制键的键值及作用如下:

用 INKEY() 得到的键值	对应键	程序中的作用
4	→	光标右移
19	←	光标左移
5	↑	光标上移
24	↓	光标下移
110	n	退出
121	y	选择确认
99	c	撤销组合
27	Esc	输入数据
13	↙(回车)	开始查询

程序中 x、y 为当前光标位置,用一小三角实心箭头(chr(17))指示;S 控制输入数据时的光标位置;hh 为所选择的组合查询字符串;P 是变量下标值从 1~12; i 为输入选择键各键值;考虑到不大熟悉逻辑关系与语法关系的使用者(语法错误主要是输入字符时忘了加引号)初用选择组合查询条件时难免出错,故做了一个简单的出错处理程序 ERROR.PRG,以保持程序的连贯性;程序的显示查询结果部分很简单,读者可根据自己的要求编写或调用某显示程序美化画面。

***** 查询 typ.prg *****

```
on error do error
clear
set escape off
set talk off
set heading off
```

```
a1='姓名'
a2='年龄'
a3='基本工资'
a4='职称'
a5='='
a6='>'
a7='<'
a8='>='
a9='and.'
a10='or.'
a11='not.'
a12='<='
hh=' '
s=12
s2=' '
x=6
y=22
p='1'
do while .t.
@ 2,20 say '*****自由组合查询*****'
@ 4,12 say '可查询的字段      算术逻辑运算符'
@ 6,12 say '姓  名[ ]      =[ ]      .and.[ ]'
@ 7,12 say '年  龄[ ]      >[ ]      .or. [ ]'
@ 8,12 say '基本工资[ ]    <[ ]      .not.[ ]'
@ 9,12 say '职  称[ ]      >=[ ]    <=[ ]'
@ 13,0 say '查询条件是:'
@ 17,1 say '['+chr(24)+chr(26)+chr(25)+chr(27)+']
+'键选择.[Y]'
@ 17,19 say '键确认.[esc]键输入.回车键查询![N]键退出!
[C]键撤销组合!'
@ 19,18 say '注意:字符型字段输入内容时要用引号括起'
@ x,y say chr(17)
i=0
do while i=0
i=inkey()
enddo
do case
case i=4 .and. y<=39
y=y+17
case i=19 .and. y>=39
y=y-17
case i=5 .or. i=24
@ x,y say ' '
p=val(p)
do case
case i=5 .and. x>=7
x=x-1
p=p-1
case i=24 .and. x<=8
x=x+1
p=p+1
endcase
p=trim(str(p))
@ x,y say chr(17)
case i=110
```

```

clear
use
cancel
case i=121
p=val(p)
do case
case y=39
p=p+4
case y=56
p=p+8
endcase
p=ltrim(trim(str(p)))
cx='a'+ '&cx'
x1=&cx
hh=hh+x1
s1=ltrim(trim(&cx))
@13,s say s1
s=s+len(s1)
x=6
p='1'
case i=27
@13, s get s2
read
@13,s say s2
s2=trim(s2)
s=s+len(s2)
hh=hh+s2
s2=' '
case i=99
s=12
x=6
y=22
hh=' '
p='1'
s2=' '
clear
loop
case i=13
use abc
locate all for &hh
if eof()
@18,20 say '无此条件的记录'
wait
else
clear
disp all for &hh
wait
hh=' '
s2=' '
x=6
y=22
p='1'
s=12
clear
endif

```

```

endcase
enddo

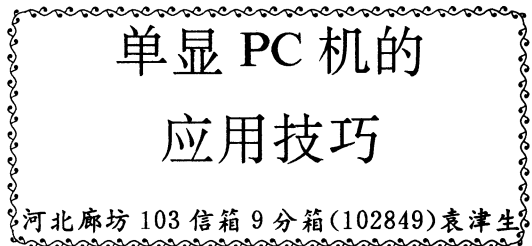
```

* 出错处理 filename is ERROR.PRG

```

clear
i=' '
@13,20 say '语法或逻辑组合或字符未加引号'
@15,28 say '按任意键重来' get i
clear typeahead
read
do typ

```



自引进 PC 机以来,配有单色图形卡的 PC 机及其兼容机在我国已为数不少。这种机器具有价格比高、字形优美及在单色汉字下显示行数多等优点,很快占领了市场,其中一部分已进入家庭。但有几点不足:

1. 为这种机器而写的游戏软件很少。
2. 在 BASIC.COM 软件中没有图形的功能。
3. 不能运用中分辨率的 CCDOS。
4. 在单色汉字下,不具有词组的功能。

基于上述原因使这种机器的应用受到限制。因此对于单位及家庭来说希望能更多地开发出这种机器的功能。下面介绍一些改进措施。

一、仿真 CGA 的功能

目前还没有一种方法可解决所有单色图形卡仿真 CGA 的图形功能,笔者在多次实验中发现了这一方法。在一些游戏软件中有一个用于单色图形显示的程序 MONO.COM,可将此程序稍加修改即可使单色图形卡具有仿真 CGA 的功能。改动如下:

```

A>DEBUG MONO.COM<CR>
-E 197
4E0D:0197 02.06<CR>
-W <CR>
Writing 026F bytes
-Q <CR>

```

其中<CR>表示回车。若将此程序加在自动批文件 AUTOEXEC.BAT 中,开机后即可完成对 CGA 的仿真。也可在 PC-DOS 操作系统下直接键入 MONO<CR>这样在仿真 CGA 下可运行游戏软件,可使 BASIC.COM 软件具有作图功能、可进入 CCDOS2.0 等中分辨率汉字操作系统,并具有图形美观、字形优美等特点。

二、给单色汉字系统加上词组功能

CCDOS 2.13H 以其优越的性能受到广大用户的

欢迎,在此系统中有一外部词组功能,可用字处理软件(WS 或 EDLIN)编辑外部词组,然后用 CZP. EXE 来建立词组文件,如 CZ2. COM。我们可将此文件拿来加入单色汉字启动的批命令中:

```
A>TYPE C. BAT
ECHO OFF
CLS
ECHO ccdos is loading.....
HGC FULL
FILE1
CCCC
CZ2
```

在启动单色汉字的批命令后,将词组调入内存。调用时先键入词组的编码,再按分号键即可。

BASIC 程序用 P 存盘的解密方法

广东恩平县供电局(529400)禩协贤

BASIC 程序编辑完成之后,如果存盘命令选用 SAVE“<文件名>”,P,程序将按加密二进制码格式保存到磁盘中去,这就是程序加密保护。经过加密的程序,用户将无法用 LIST 命令显示程序清单,也无法对其进行修改。本文介绍一种非常简单的 BASIC 程序的解密方法,本解密程序内容只有两个字节,编制、使用方便,任何可以上机操作的人都能轻易完成。

一、解密程序的编辑

解密程序是在 DOS 状态下,按二个步骤编辑。

首先建立一个名为“P. BAS”的空文件,操作如下:

```
A>COPY CON : P. BAS<CR>      建立文件名
-<CR>                          空一行
-^ Z 或[F6]<CR>              存盘
```

以上操作可以在任意当前工作盘上进行。到此,若用 DIR 命令查当前工作盘,可发现有一个名为 P. BAS 的文件,长度为 2 字节。

第二步用 DEBUG 命令修改上述文件,操作如下:

```
A>DEBUG P. BAS<CR>      装入 P. BAS 程序。
                          如 DEBUG 文件不在 A
                          盘上,DEBUG 命令前要
                          加上所在的盘符。
```

```
-D CS : 100 101<CR>    显示偏移地址 100~
                          101 的内容
```

```
239C : 0100 0D 0A      (内容为 0D 0A)
-E100<CR>             修改偏移地址为 100 的内容
239C : 0100 0D. FF<CR> 将 0D 改为 FF
-E101<CR>             修改偏移地址为 101 的内容
```

```
239C:0101 0A. 1A<CR>  将 0A 改为 1A
```

```
-W<CR>                写盘
```

```
Writing 0002 bytes
```

```
-Q<CR>                退出修改
```

解密程序已编辑完成,其内容就是:FF 1A.

二、使用方法

系统进入 BASIC 状态后,第一步先用 LOAD“<文件名>”装入 P 加密程序到内存。(此时,如用 LIST 命令显示程序,会出错:“Illegal function call”)。

第二步再用 LOAD “P”命令,将解密程序调入内存,经过第二步聚后,原先装入内存中的被加密程序已被解开,用 LIST 命令即可以显示该程序的全部内容,并可以进行任意修改编辑。

编辑完后,若再用不带 P 参数的 SAVE 命令重新存盘,即可得到一个未加密的 BASIC 源程序文件。

谈谈编译 BASIC 在 长城机上的运用

江苏南京军区工程兵部(210016)卢祥江

用过的同志都知道,编译 BASIC 是一种非常受欢迎的语言。经过编译的 BASIC 程序有运行速度快、节省存储空间、保密性强、通用性好、其它语言调用方便等优点。但目前的编译 BASIC 版本也有其缺点,那就是在长城系列机上使用时(其它系列机上也),有的 BASIC 命令编译后无法执行,主要是画图语句,如 LINE、PAINT、CIRCLE 等。为了解决这个问题,我们用长城机上配套的 GRD. SYS 系统图形驱动文件中的同类命令来代替,取得了圆满成功。譬如用:

```
OPEN "GRP" FOR OUTPUT AS#1
WRITE #1,"L",C,X1,Y1,X2,Y2
WRITE #1,"B",C,A,X1,Y1,X2,Y2
WRITE #1,"C",C,X0,Y0,R
WRITE #1,"P",X1,Y1,C1,C2
CLOSE #1
分别代替 BASIC 语言中的:
LINE (X1,Y1)--(X2,Y2),C
LINE (X1,Y1)--(X2,Y2),C,B
CIRCLE (X0,Y0),R,C,,1
PAINT (X1,Y1),C1,C2
```

但在使用时要注意:

1. BASIC 程序中有的画图语句用的是相对坐标,用 GRD. SYS 时要换算成绝对坐标。

2. 使用 GRD. SYS 时,必须要用有 GRD. SYS 和 CONFIG. SYS 文件的硬盘或软盘启动微机,并且 CON-

FIG. SYS 中必须要有 DEVICE=GRD. SYS 这条命令。

3. 在程序中使用,每屏显示要先打开文件,即有:

```
OPEN "GRP" FOR OUTPUT AS #1
```

这样一条命令,每屏结束时有 CLOSE #1 关闭文件命令。请注意:每屏显示结束时一定要有 CLOSE #1 语句,否则将会出现最后的几句命令暂不执行的现象,即最后的几笔不画,清屏后再画出来。

4. 把 BASIC 源程序编译成 EXE 文件有两种方法,并各有其特点。

第一种是在编译时使用 BASCOM. LIB 库文件,编译成的 EXE 文件不要其它文件的支持就可独立运行,但编译后的 EXE 文件较长,单个文件占用磁盘空间较大。如:1K 左右的源程序编译成 EXE 文件约有 19K。

第二种是在编译时使用 BASRUN. LIB 库文件,编译后的 EXE 文件需要 BASRUN. EXE 的支持才能运行,即在运行编译成的 EXE 文件的当前盘中必须要有 BASRUN. EXE 文件。但这样编译成的 EXE 文件较短。如:1K 左右的源程序编译成 EXE 文件约有 2K。

如从运行方便和节约内存考虑可采用第一种方法;如从节约磁盘空间角度考虑可采用第二种方法。

5. 在用高分辨率显示时,屏幕设置语句解释 BASIC 与编译 BASIC 有所不同,用解释 BASIC 调试好的程序经过编译后功能会发生变化。如清屏功能,用解释 BASIC 清屏很正常的程序,经过编译后有可能清不掉。经过我们试用,一般开头要用 SCREEN 0,清屏或者结束退出时用 CLS;SCREEN 2;CLS;SCREEN 0。如不加 SCREEN 2;CLS 有可能会使字符清除了而画的图没清除。也可以利用该功能,退出系统,仍保留图形为其它语言提供使用。譬如:用该功能在屏幕上画出并保留表格线,而用其它语言(如 dBASE III 等)填写字符,就形成了表格线,其优点是表格线不占字符位置,且用 dBASE III 语言中的清屏命令也不会清除掉,可以反复使用,待用完后再用 BASIC 的清屏程序清除掉。该方法适合中国人的使用习惯,可以大大方便用户,可以用这种方法编制出实用性很强的软件。我们有一个软件用了这种方法,受到了用户的普遍欢迎和很高评价。

6. 请使用者注意,解释 BASIC 的源程序在要编译时,源程序必须要用 ASCII 码的形式存盘(即用:SAVE "文件名",A 命令存盘),否则,编译时会提示错误信息"0 SEVERE ERROR(S)"。

注释:1. #1 可以是 #2、#3……

2. X1, Y1 为起点坐标, X2, Y2 为终点坐标, X0, Y0 为圆心坐标, C, C1, C2 为颜色, R 为半径, A 为实心或空心。

也谈解决运行 FoxBASE 内存不够的问题

江苏徐州邮电局(221000)周立宇

本刊 91 年第二期发表的“如何解决运行 FoxBASE 内存不够的问题”,介绍了取消二级字库的办法来减少汉字驻留 RAM 空间,以解决 FoxBASE 运行空间不足的问题。本人经过实践,发现有更为简单有效的解决方法,其主导思想是:不把汉字装入 640K 内存,以达到节省空间的目的,而不必删除二级字库,以方便使用。

目前,市面上经常看到的 CCDOS4.0 和 CCDOS2.13(A~H)都提供了把汉字安装在硬盘上的处理方法。这不仅使得有足够的内存运行 FoxBASE,还可有足够的内存运行其它有用的软件,如 Watchman 等病毒检测程序和 PC-Cache 等辅助提高处理速度的软件,可谓一举多得。

以 CCDOS2.13H 为例处理步骤如下:

1. 把 CCDOS2.13H 的第七号盘装入 A 驱

```
A>ZK
```

按屏幕提示分别将 CCDOS2.13H 的内容装入系统。

2. 建立新的批处理 AUTOEXEC. BAT

```
C>\dos\edlin autoexe. bat
```

```
New file
```

```
* i
```

```
1: * Echo off
```

```
2: * cls
```

```
3: * cd \213
```

```
4: * file0a82
```

```
5: * cccc
```

```
6: * cc11(或 cc25)
```

```
7: * int10f
```

```
8: * yx1
```

```
9: * prta
```

```
10: * file16b
```

```
11: * file24a 1sfhk
```

```
12: * zf24 3
```

```
13: * path c:\; c:\213;
```

```
14: * key
```

```
15: * cd \
```

```
16: * prompt YpYg
```

```
17: * ^ Z
```

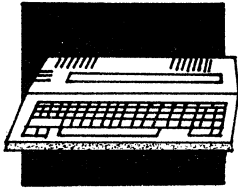
```
* e
```

3. 重新开机启动

```
C:\>\DOS\CHKDSK C:
```

```
21309440      bytes total disk space
   55296      bytes in 2 hidden files
   43008      bytes in 10 directories
12527616      bytes in 994 user files
```

(下转第 3 页)



学习机之友

虚拟 DOS 的改进及其在通信网中的应用

甘肃临洮农校(730500) 石永琳

本刊 1991 年 7 期刊登的《为无驱动器系统设置虚拟磁盘》一文,为中华学习机的无驱动器用户提供了一个相当于 DOS3.3 的运行环境,减轻了频繁操作录音机的麻烦。此外,虚拟磁盘操作系统还能解决在无驱动器配置的机器上,进行 DOS3.3 操作系统的教学和实习问题。鉴于该文提供的程序有较大的实用价值,笔者现就其中的不完善之处作些补充。

原文程序中有一处排版错误需要更正: \$BDC8 单元应为 \$BD。

一、虚拟盘数据存储、恢复方法的改进

原文给出的数据备份和恢复方法对“小蜜蜂—1(XMF—1)中华机是正确的,只是操作有点烦琐,备份数据时需要分两步完成:

(1) 键入 CALL 48557

(2) 用 → 键扫描屏幕显示的 BSAVE “XNP”, A \$3FFD, L \$4057 命令,回车执行。

恢复数据时执行 BRUN “XNP”命令从磁带读数据。

由于 CEC—1 等苹果兼容机 BASIC 语言无 BSAVE、BRUN 命令,只能在监控状态下存储和恢复数据,为这些机型修改的程序执行时将显示一条监控命令 3FFD.8054W,不进入监控就不能执行。恢复数据时也要经过 CALL—151、3FFD·8054R、3FFDG 等步骤。

针对原文存储、恢复数据操作烦琐的问题,笔者对原程序进行了分析,找到了各种机型通用的数据备份和恢复方法,并通过以下两方面实现。

一是重新编制了数据备份和恢复程序,安排在原程序中 \$BDF7~\$BE99 这一区域,程序附后。

二是修改了两条 DOS 命令,使之适宜于在虚拟盘系统中使用。具体修改过程是在监控状态下键入 A8EF;49┘CE┘4F┘55┘D4┘N 9D4A;19┘BE┘AC┘BD。现在,DOS 就有了两条新命令——OUT 和 IN。其中 OUT 命令用来备份数据,把“盘”中数据写入磁带,IN 命令从磁带读回数据,原 DOS 命令 FP、INT 取消。

改进后的程序,键入 OUT 命令,机器显示 Press any key to SAVE !! 按下录音机录音键,依照提示敲任一键即开始存储数据,约 2 分钟时间全“盘”数据存入磁带。存储开始和结束时机器响一声“嘟”。

键入 IN 命令时,机器显示 Loading... 字样,表示正在等待读入数据,按录音机放音键,读带完毕即可恢复“盘”中原来数据。

二、在简易通信网上运行虚拟磁盘操作系统

把中华学习机的录音机输入输出 EAR、MIC 用屏蔽线交叉连接,就构成能实现机器间通信的简易网络。下面介绍利用简易网运行虚拟磁盘操作系统的方法。这将使您的简易网实现比原来更高一级的资源共享。要求简易网中至少有一台机器配置驱动器。

1. 建立系统程序

在配有驱动器的机器上照原文及本文的改进办法建立系统程序。CALL—151 进入监控,键入 6CFD<9CFD.BFFF,把程序搬出 DOS 区,退出监控,用 PR#6 命令重新启动 DOS 3.3,键入 BSAVE RAMDOS,A \$6CFD,L \$2303。至此,磁盘上已建立了名为 RAMDOS 的系统程序。

2. 通过简易网为无驱动器机器设置虚拟盘

为了叙述方便起见,把配置驱动器的一台机器称为“主机”,网上其它机器为“分机”。用主机为分机设置虚拟盘的步骤如下:

(1) 主机:在 DOS 下键入 BLOAD RAMDOS,将系统程序从磁盘上调入主机。

(2) 分机:监控下键入 9CFD.BFFF R 读程序;

主机:监控下键入 6CFD.8FFFW 发送程序。

(3) 分机:从首址 \$9CFD 或 \$BD00 处执行程序,机器的 16K 卡即被设置为虚拟盘,可用 DOS3.3 命令对其进行管理了。

可见,这种方法要比从磁带读入程序设置虚拟盘方便可靠。

3. 虚拟盘与主机磁盘的通信

分机“盘”中数据在主机磁盘中建立备份:

(1) 主机:监控下执行 5000.8FFFR;

分机:键入 OUT 命令,之后按任一键。

(2) 主机:退出监控返回 DOS 系统。键入 BSAVE RAMDISK n,A \$5000,L \$4000,即将虚拟盘中的数据存入磁盘。n 是为了防止文件重名造成覆盖对文件的编号。

恢复虚拟盘中的数据:

(1) 主机:BLOAD RAMDISK n 读数据,注意文件编号与存入时一致。

(2) 分机:键入 IN 命令

主机:监控下执行 5000.8FFFW 传送数据至分机。

由于条件限制,中小学校计算机室大部分没有为每台机器配置磁盘驱动器。利用简易通信网设置虚拟盘的办法,不仅能满足学习 DOS 3.3 的需要,也将使每台机器都享受到磁盘驱动器的便利。

事实上,即使配置有驱动器,把 16K 卡设置为虚盘也是有意义的,这样可以减少对磁盘的访问,有利于保护机器节省时间。

BDF7— 2C
BDF8— 10 C0 2C 00 C0 10 FB 20
BE00— 3A FF 20 09 BE 20 CD FE
BE08— 60 A9 00 85 3C A9 40 85
BE10— 3D A9 FF 85 3E A9 7F 85
BE18— 3F 60 A2 00 BD 8E BE 20
BE20— F0 FD E8 E0 0B 90 F5 20
BE28— 09 BE 20 FD FE 8D 81 C0
BE30— 8D 81 C0 A9 00 85 42 85
BE38— 3C A9 D0 85 43 A9 40 85

BE40— 3D A9 FF 85 3E A9 6F 85
BE48— 3F A0 00 20 2C FE 8D 89
BE50— C0 8D 89 C0 A9 00 85 42
BE58— 85 3C A9 D0 85 43 A9 70
BE60— 85 3D A9 FF 85 3E A9 7F
BE68— 85 3F A0 00 20 2C FE 8D
BE70— 82 C0 60 00 00 8D D0 F2
BE78— E5 F3 F3 A0 E1 EE F9 A0
BE80— EB E5 F9 A0 F4 EF A0 D3
BE88— C1 D6 C5 A0 A1 A1 8D CC
BE90— EF E1 E4 E9 EE E7 AE AE
BE98— AE 00

磁盘数据的压缩存储技巧

四川南充一中(637000) 陈庆祥

众所周知,在利用磁盘存储数据信息时,应尽可能地将性质结构相同的记录存放在同一文件中,以便于进行各种处理。这一点对于只配备了单驱的系统显得更为重要。例如,现在要将某地初中升学考试的万名考生记录以随机文件的形式存盘(目前,百万人口的县市每年初中毕业生都在万名左右)。每个考生的姓名、性别、类别和 7 科考分共 10 个数据项组成一个占 45 个字节的记录(前三项共 6 个汉字,占 18 个字节,后 7 项占 15 个字节,加上 10 个分隔符占 10 个字节),那么,一张单面单密度软盘即使格式化到 40 磁道且不存入 DOS,至多也只能容纳 3000 左右的考生记录。万名考生记录就不得不存放在四张盘内,致使查询记录时必须频繁地换盘(特别是随机查询时),不仅增加许多麻烦,也明显地降低了处理效率。

那么,能否在现有的硬件条件下,用一张软盘来存放更多的数据信息呢?比如,将上述的万名考生记录存入一张盘内。经笔者的反复探索,证明是可行的,其出路就在于对数据进行巧妙的压缩处理。

根据考生记录各数据项的类型与特征,分三种情况进行压缩处理:对于姓名这几个汉字,只存入其内码中的区码与位码,而不必存入汉字的鉴别码 \$7F(中华机硬汉字以 \$7F 为鉴别码);对于考生性别与类别分别用代码表示(如 1—男,2—女),并将两个代码合为一个二位整数,再用 CHR \$ 函数转换为对应的一个 ASCII 字符;对于 7 门考分皆为 0~100 的整数,当然可用上面类似方法转换为 7 个 ASCII 字符,不过由于 CHR \$(4) 为 DOS 命令引导符,会影响数据的正常存取,故可将各考分加上 5 再转换。所以本来占用 45 个字节的考生记录可压缩为 15 个(由于各 ASCII 字符均表示一个相对独立的信息,相互间不必再用分隔符,仅在记录尾部以一个回车符用来区别不同记录),整个文件长度仅为原来的三分之一,一张磁盘就完全可以容

纳一万多名的考生记录。

很明显,数据压缩存储的结果不仅大大提高了磁盘的利用率,方便了大批量的数据处理,而且由于对同一批数据的磁盘存储量大为减少,文件存取时间也相应缩短(虽然数据的转换压缩要耗费时间,但由于 CPU 的处理比磁盘动作快千倍以上,故文件存取仍比原先快得多),真可谓一举数得。

以上数据压缩存储的方法适用于所有长度变化范围较小的数据(较长数据可分段处理),既适用于数值,也适用于字符(如整篇中文文章);既适用于顺序文件,也适用于随机文件,而且不限定任何一种机型。

本文末尾提供了考生记录压缩存储与随机查询的两个程序实例(在中华学习机上通过)。其中程序 1 在每次运行前打入不同的 DATA 语句(用 DATA 语句提供数据便于检查、修改),可将万名考生记录分任意次存入同一随机文件内,(文件长度不受限制,直至数据充满整张磁盘,程序后提供了两次运行的压缩存储例子),程序 2 能随机地查询由程序 1 建立的随机文件内任何一名考生的记录,并且用通常方式输出。

程序 1

```
10 REM 考生记录的压缩存储,以单个 ASC 字符代替  
1—3 位数  
20 D$ = CHR$(4):H$ =CHR$(127):S = 0  
30 INPUT "是该文件数据的首次输入吗?(Y/N)";SC$  
40 PRINT "请稍候!正在将数据进行转换及存入磁  
盘。";PRINT D$;"OPEN HK,L15"  
50 IF SC$ <> "Y" THEN PRINT D$;"READ HK,R0":  
INPUT N  
60 READ H: IF H = 0 THEN N=N + S: GOTO 150  
70 READ N$,XB,SL: FOR I = 1 TO 7: READ M: M  
$(I)=CHR$(M + 5): NEXT  
80 PRINT D$;"WRITE HK,R";H  
90 FOR I = 1 TO LEN(N$): P$ = MID$(N$,I,1)
```

```

100 IF P$ < > H$ THEN PRINT P$;
110 NEXT
120 PRINT CHR$(10 * XB + SL);
130 FOR I = 1 TO 7: PRINT M$(I);
140 NEXT: PRINT: S = S + 1: GOTO 60
150 PRINT D$; "WRITE HK,R0": PRINT N: PRINT
D$; "CLOSE"
160 PRINT "OK! 本次存入了"; S; "个考生记录, 文件内
共有"; N; "个记录, 下次从 200 行起重新打入数
据。": END
170 PRINT D$: GOTO 40
200 DATA 00001, 王建华, 2, 1, 85, 73, 70, 66, 74, 67, 82
210 DATA 00003, 周 敏, 2, 1, 70, 84, 65, 86, 90, 79, 70
220 DATA 00002, 刘 宏, 1, 1, 86, 90, 92, 77, 86, 79, 92
230 DATA 00004, 肖 军, 1, 2, 88, 94, 75, 82, 90, 84, 96
240 DATA 00000
]RUN

```

是该文件数据的首次输入吗? (Y/N) Y
 请稍候! 正在将数据进行转换及存入磁盘。
 OK! 本次存入了 4 个考生记录, 文件内共有 4 个记录, 下
 次从 200 行起重新打入数据。

```

]LIST 200—
200 DATA 00006, 何 锐, 1, 1, 88, 94, 85, 82, 90, 99, 94
210 DATA 00005, 张中明, 1, 2, 80, 64, 70, 56, 64, 69, 74
220 DATA 00000
]RUN

```

是该文件数据的首次输入吗? (Y/N) N
 请稍候! 正在将数据进行转换及存入磁盘。
 OK! 本次存入了 2 个考生记录, 文件内共有 6 个记录。下
 次从 200 行起重新打入数据

程序 2
 10 REM 随机查询文件中的考生记录
 20 H\$ = CHR\$(127): K\$ = CHR\$(32): XB

```

$ (1) = "男": XB$ (2) = "女": SL$ (1) = "应
届": SL$ (2) = "往届"
30 D$ = CHR$(4): PRINT D$; "OPEN HK,L15"
40 INPUT "请输入欲查询的考生考号: "; H: N$ = "
50 IF H = 0 THEN PRINT D$ "CLOSE": END
55 PRINT D$ "READ HK,R"H
60 FOR I = 1 TO 3: GET A$: GET B$
70 IF A$ = K$ AND B$ = K$ THEN N$ = N$ +
K$ + K$: GOTO 90
80 N$ = N$ + H$ + A$ + B$
90 NEXT: M = 0: GET XL$: XL = ASC(XL$)
100 XB = INT(XL / 10): SL = XL - 10 * XB
110 FOR I = 1 TO 7: GET M$: M(I) = ASC(M$)
- 5: M = M + M(I): NEXT
120 PRINT D$: PRINT "第"H"号考生记录如下:"
125 PRINT "考号 姓名 性别 类别 政 语 数
英 物 化 生 总分"
130 PRINT RIGHT$("0000" + STR$(H), 5) " ";
140 PRINT N$ " " XB$(XB) " " SL$(SL) " ";
150 FOR I = 1 TO 7: PRINT LEFT$(STR$(M(I)) +
" ", 3);
160 NEXT: PRINT M: PRINT
170 PRINT D$: GOTO 40
]RUN

```

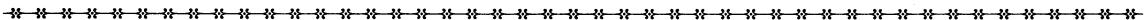
请输入欲查询的考生考号: 6
 第 6 号考生记录如下:

考号	姓名	性别	类别	政	语	数	英	物	化	生	总分
00006	何锐	男	应届	88	94	85	82	90	99	94	632

 请输入欲查询的考生考号: 1
 第 1 号考生记录如下:

考号	姓名	性别	类别	政	语	数	英	物	化	生	总分
00001	王建华	女	应届	85	73	70	66	74	67	82	517

 请输入欲查询的考生考号: 0



(上接第 8 页)

```

ENDIF
ENDIF
ENDIF
ENDDO
RETU
* K_BOARD.PRG
* 功能: 模拟 dBASE III Plus 中的 Return to Master

```

```

PARAMETERS P1,P2,P3
KEYBOARD CHR(27)
W_RETU=.F.
RETU
* 自定义函数 CENTER
FUNCTION CENTER
PARAMETERS STRING
RETURN INT(80-LEN(STRING))/2

```

为 Apple Dos 创造三条实用命令

湖南株洲硬质合金厂(412000)王志超

使用 DOS 3.3 的苹果机或中华学习机用户都会感到:①经常要为 10—16 进制的数据转换大伤脑筋;②复制文件不得不用专用的拷贝程序,颇为不便;③要了解软盘某个文件的大致内容而不破坏内存的现行程序也很不方便。那么,能否编制几条格式简单的 DOS 命令,来解决上述问题呢?笔者经过一番探索,终于达到了目的,现在推荐给大家,希望能为各位带来方便。

一. 数制转换命令—ZH

1. 本命令用于立即执行方式。格式:

① 10→16 进制转换:ZHAX (X 为 0—65535 的 10 进制数据)

② 16→10 进制转换:ZHAX (X 为 0—FFFF 的 16 进制数据)

以上的格式中,A 为键语,不可缺少。

2. 原理:本命令巧妙地利用了 DOS 本身的几段子程序。首先调用 \$A1A4 来检查命令中是否有“\$”(16 进制标志),接着调用 \$A1B9 来转换数据,转换好的数据放在 \$44、\$45 单元,这时再一次判断输入数据是否 16 进制,是则调用 \$DE24 将转换好的 10 进制数输出到屏幕,否则调用 \$F940 以 4 位 16 进制的形式输出。

二. 文件拷贝命令—COPY

1. 本命令在立即方式或延迟方式均有效。格式:
COPY 文件名 1,文件名 2

格式中的文件名 1 为源盘的文件名,文件 2 为目标盘的新文件名。

2. 本命令可拷贝 A、B、I 类型的文件。它综合利用了“BLOAD”和“BSAVE”的部分功能。命令执行时,首先读源盘,当文件 1 全部读出后,光标会停在命令的末尾,待用户换上目标盘并按任意键后,就以文件名 2 为名拷至目标盘。拷贝过程中的被拷文件暂存区为 \$3000-\$7FFF,可拷贝的文件最大长度为 20K。如果在中文状态下使用,立即方式下可先进入“西文”再执行命令,延迟方式下可在程序中增加以下语句:

```
<行号>HOME: TEXT: PRINT D$“COPY 文件  
名 1,文件名 2”: PRINT D$“PR#3”: PRINT
```

三. 查看文件命令—TYPE

1. 本命令在立即方式或延迟方式均有效。

2. 说明:类似的命令曾见于其它资料,但均存在以下弊病:①命令执行后再执行别的 DOS 命令时常出现“缓冲区不够”的错误;②在中文状态使用时很容易转入“西文”,且常进入反相显示。笔者编制的 TYPE 进行了改进,即:增加清屏功能;调用 \$A316 以释放缓冲区;调用 \$F273 恢复正相显示;屏蔽“西文”键等,这样,在中、西文状态下都能正常显示出指定文件的内

容。另外,CTRL—S 暂停、CTRL—C 中止命令的执行。

四. 命令修改步骤

1. 输入后附的机器码程序。其中,\$9500—\$9534 为 TYPE,\$9535—\$9598 为 COPY,\$9599—\$95BC 为 ZH。

2. 修改有关单元:

①改命令名。首先将原 \$A898—\$A8E3 的机器码顺序向前移一个字节,这只要执行 A897 < A898.A8E3M 命令即可。然后:

```
A893:54└59└50└C5
```

```
A8E3:43└4F└50└D9
```

```
A8EF:5A└C8
```

②改键语表。

```
A931:71└79└
```

```
A935:00└01
```

③改入口地址。

```
9D26:FF└94
```

```
9D46:34└95
```

```
9D4A:98└95
```

3. 调出 HELLO 程序,在最前面加一句:

```
HIMEM:38144:PRINT CHR$(4)“BLOAD COM-  
MAND”(如果是中文 HELLO,则可去掉 HIMEM:38144  
语句)
```

4. 格式化一个新盘,并将机器码程序以“COM-MAND”为名存盘。

5. 启动新盘。现在,你已经可以得心应手地使用这三条新的 DOS 命令了。

最后请注意:原 DOS 命令 CHAIN、IN#、FP 失效。

9500—20	7E	D2	20	A8	A2	20	8C
9508—A6	09	80	C9	91	F0	03	20
9510—ED	FD	AD	00	C0	2C	10	C0
9518—C9	93	D0	08	AD	00	C0	10
9520—FB	2C	10	C0	C9	83	F0	07
9528—AD	C5	B5	C9	05	D0	D7	20
9530—16	A3	4C	73	F2	A9	00	8D
9538—72	AA	A9	30	8D	73	AA	20
9540—A8	A2	AD	C2	B5	29	7F	85
9548—08	20	D5	A3	A5	08	C9	02
9550—F0	08	C9	01	F0	07	20	7A
9558—A4	85	06	84	07	20	82	A3
9560—20	0C	FD	A0	1D	B9	93	AA
9568—99	75	AA	88	10	F7	A5	08
9570—20	D5	A3	AD	60	AA	8D	6C
9578—AA	AD	61	AA	8D	6D	AA	A5
9580—08	C9	02	F0	0E	C9	01	F0
9588—0A	A4	06	A5	07	20	48	A3
9590—4C	51	A8	20	4B	A3	4C	51
9598—A8	20	8E	FD	A2	03	8E	5D
95A0—AA	20	A4	A1	48	CE	5D	AA
95A8—20	B9	A1	A8	68	C9	A4	F0
95B0—08	A9	A4	20	ED	FD	4C	40
95B8—F9	98	4C	24	ED			

ProDOS 磁盘操作系统入门(续)

廖 凯

五、屏幕编辑

ProDOS 允许光标既可在 40 列方式下又可在 80 列方式下的屏幕范围内移动,如果计算机是处在 80 列方式下,那么在光标内会出现一个小+(加号),以区别于 40 列方式,按下 ESC 键即进入屏幕编辑状态。它有以下编辑键:

ESC A	光标向右移一位,退出编辑状态
ESC B	光标向左移一位,退出编辑状态
ESC C	光标向下移一位,退出编辑状态
ESC D	光标向上移一位,退出编辑状态
ESC K	光标向右移一位,保持编辑状态
ESC J	光标向左移一位,保持编辑状态
ESC M	光标向下移一位,保持编辑状态
ESC I	光标向上移一位,保持编辑状态
ESC E	由光标处删至行尾
ESC F	由光标处删至页末
ESC @	同时按下 ESC 和@(SHIFT-2)键,功能同 HOME 指令
CTRL-X	删除当前打入的行
CTRL-K	光标上移一行
CTRL-J	光标下移一行
→键	光标右移一位,将光标下的字符输入内存
←键	光标左移一位,将光标下的字符从内存中删去
↑键	同 CTRL-K。只有 APPLE IIe、IIc 及中华学习机才有此键
↓键	同 CTRL-J。只有 APPLE IIe、IIc 及中华学习机才有此键
DELETE 键	删除行中最后一个字符并回送该字符。只有 APPLE IIe 和 IIc 才有此键

六、程序除错

ProDOS 有几种除错的方法。TRACE 命令被用来显示当前正在执行的行号。即使你从磁盘发送或接收信息,TRACE 也将同 ProDOS 一起正确地工作。如果在 TRACE 有效时而输出同时被显示在屏幕上,那么会有混乱的结果,终止 TRACE 的命令是 NOTRACE。

另一个除错的方法是使用 STOP 和 CONT 命令,从你的程序内部将 STOP 放置在重要的位置上,当程序遇到 STOP 时将停止,你可以由立即方式打印出变量的当前值,要继续执行输入 CONT 即可。这个过程是分析程序的一个最简单和有效的方法。

七、错误中断

错误中断是开发一个程序的一个很重要的部分。每当程序发生错误时,它会纠正错误并重新运行程序。在一个程序运行时,发生许多不同类型的错误是不罕

见的,在一个错误发生及中断它时,要懂得处理错误的技巧。

处理错误最常用的命令是 ONERR GOTO 及 RESUME。若错误发生,将出错标志单元 216 的最高位置 1,并将错误类型码放在 222 单元内。POKE 216,0 用于清除错误标记。当一个错误发生时,内部控制堆栈指针被破坏,控制堆栈记录着 GOSUB/RETURN 的位置,错误发生时,错误代码被加到堆栈上,因此在 GOSUB 子程序内发生的任何错误都将破坏 GOSUB/RETURN 指针。ONERR GOTO 语句是用于程序出错时转到指定的错误处理子程序,RESUME 是错误处理子程序的返回语句,它将自动地消除堆栈上无用的字符而保持控制堆栈的完整性。

控制堆栈问题可以通过下面的程序来减轻,此程序通过消除堆栈上无用的字符,来清除错误并保持控制堆栈的完整。

```
10 FOR I=768 TO 777
```

```
20 READ J
```

```
30 POKE I,J;NEXT
```

```
40 DATA 104,168,104,166,223,154,72,152,72,96
```

中断错误之后,用 CALL 768 来执行错误处理子程序,然后用一个 GOTO 语句返回到程序。

在中断错误时,认清错误发生的类型是很重要的,它给使用者一个信息,以指示使用者下一步做什么。例如使用者正发送输出到一个没被连接的打印机,使用者可用下面的方法截取和显示错误:

```
10 ONERR GOTO 1000
```

```
.....
```

```
1000 IF PEEK(222)=3 THEN PRINT"The printer is not connected or is not ON,Please check the condition and try again."
```

八、程序设计结构与优化

1. 程序结构

程序主要由三个部分组成:初始部分、主体和子程序,首先给程序置初值,设置变量和定义所有数组,然后开发主体,包括菜单和树形分枝,最后是子程序,主体将充分利用这些子程序。使用子程序将使你的程序更灵活、更易读和更容易除错。

2. 变量的规定

APPLESOFT 使用变量的三种类型是:整型、实型和串变量。整型变量用加在变量名后的%表示,其值在-32767至32767之间。整型变量变量名占用2字节,值占用2字节。实型变量是以最多9位十进制数字的精确度来表示其值。实型变量变量名占用2字节,值占用5字节。串变量表示一个字符串,串变量的最大长度为239个字符。串变量变量名占用2字节,长度占用1字节,指针占用2字节,字符串内每一字符占1字节。

3. 程序优化

一个好的程序应当具备以下一些条件:

- ① 执行效率高,节省运算时间,节省内存空间。
- ② 程序条理清楚,用比较简单易懂的方法表示。
- ③ 考虑到运行时可能发生的各种情况,应设立必要的错误处理程序。
- ④ 有适当的说明(用 REM 语句),使程序的使用者能了解程序中各部分的作用。

要想设计好的程序,建议按照以下几方面进行设计:

- ① 需选择好的计算方法,这是最重要的。
- ② 最好采用结构化的程序设计方法,尽量少用 GOTO 语句,而用 IF-THEN 形式,使复杂的程序也能一目了然。
- ③ 重复使用同一部分语句时尽量用子程序。
- ④ 节省使用数组,这是节约内存的关键,不要任意指定过大的数组,尽量重复使用前面已用过而不再用的数组或变量以节省内存。
- ⑤ 尽量在一行中写几个语句,这样可省内存。
- ⑥ 重复出现的表达式应事先计算好赋予一个变量保存,以减少重复计算。
- ⑦ 整型变量的计算比较准确且运算速度快。如内存不够用时,应尽量用整型变量和数组代替单精度变量和数组。整型变量占二个字节,而单精度变量占四个字节。
- ⑧ 内存不够时,可减少不必要的 REM 语句。
- ⑨ 在写程序时应考虑到调试和日后使用维护的需要。

下面是一些语句占用内存的情况:

用 DEF FN 语句定义一个函数时占用 6 字节,各保留字占用 1 字节,一个 FOR/NEXT 循环占用 16 字节,一个正被执行的子程序占用 6 字节,在表达式内的各对括号占用 4 字节,在表达式内各个计算的临时结果占用 12 字节。

九、文本文件

ProDOS 的最大特点之一是文件的快速存取,它比 DOS3.3 快 8 倍,在对大型文件进行操作时尤其明显。ProDOS 文件的大小在磁盘上不限于 140K 字节,在使用硬盘时,你可以建一个具有 16M 字节的文件。ProDOS 取消了 MAXFILES 命令,它可同时使用 8 个文件,对于每个打开的文件,ProDOS 将自动分配 1K 字节的缓冲区。

ProDOS 允许用“-”命令来运行文本文件,如果文件带有行号,那么 EXEC 命令不执行这些程序行,而是按行号次序将它们装入内存。如果文件不带有行号,那么 ProDOS 将执行它们。

ProDOS 的文本文件有两种类型:顺序存取文件和随机存取文件。

(一)顺序文件

顺序文件是由若干个连接在一起的数据域组成,数据域由若干字节的字符组成并以回车字符(CR)为

结束标志,一个数据域最多可包含 230 个字符。

1. 写顺序文件

WRITE 命令在 ProDOS 内被改进,在顺序文件内 WRITE 命令可以使用 F# 参数,它是在写操作之前把文件位置指针向前移动若干个数据域作为当前的数据域,这参数取代了 POSITION 命令,在顺序文件内使用 F# 参数的形式是:

```
10 PRINT D$;"WRITE/卷目录/文件名,F#"
```

下面的例子是向 EXAMPLE 文件写入一些数据。

```
10 REM Sequential Text File
100 HOME
110 D$ = CHR$(4)
120 DIM N$(100),C$(100),P$(100)
130 A = A + 1
140 INPUT "Name:";N$(A)
150 INPUT "Company:";C$(A)
160 INPUT "Phone:";P$(A)
170 INPUT "Enter another name?";Q$
180 IF Q$ = "Y" OR Q$ = "y" THEN HOME;GOTO 130
190 IF Q$ = "N" OR Q$ = "n" THEN 210
200 GOTO 170
210 PRINT D$;"OPEN/DATA.DISK/EXAMPLE";PRINT
D$;"CLOSE";PRINT D$;"DELETE/DATA.DISK/EXAMPLE";PRINT D$;"OPEN/DATA.DISK/EXAMPLE"
220 PRINT D$;"WRITE/DATA.DISK/EXAMPLE"
230 FOR I = 1 TO A
240 PRINT N$(I);PRINT C$(I);PRINT P$(I)
250 NEXT
260 PRINT D$;"CLOSE"
270 END
```

磁盘命令有通过路径来指定的,路径包含卷目录和文件名,文件的打开、关闭、删除和再打开是用以清除原有的同名文件,以防文件被重写。

注意 CLOSE 命令没有路径,当你只用 CLOSE 命令时,所有打开的文件都将被关闭,如果需要,你可以关闭一个指定的文件。

如前所述,ProDOS 是以 512 字节/块为单位来写盘的,并且 CLOSE 或 FLUSH 命令会强使所有剩余的字符被写入文件,FLUSH 命令不能将文件关闭,但会写盘并清除缓冲区。当你确信要立即将缓冲区内的数据写盘时,你就要用 FLUSH 命令。

注意每次打开文件时会占用很多的时间,故最好减少打开文件的次数。最有效和省时的方法是一次将所有数据输入完。

PRINT 语句的用法:在写盘时,你可以用 PRINT 命令来改变数据域的输出格式。例如:

① PRINT "ProDOS":将字符串 ProDOS 写入文件,随后紧跟一个回车字符。

② PRINT "Pro";PRINT "DOS":将字符串 Pro 写入文件并将下一个 PRINT 语句加到此数据域的末尾。

③ PRINT "Pro","DOS":将字符串 ProDOS 写入文件,这逗号无效。(未完待续)

第二章 微处理器、存储器、寄存器简介

南京大学大气科学系(210008) 朱国江

一、计算机的主要功能

计算机最主要的功能是处理信息。所谓信息,是指客观事物发出的消息、情报、指令、数据和信号等。这些信息可以是文字、图形、声音等等。这种功能可以简单概括成一句话:输入——处理——输出。这就是说,向计算机输入待处理的各种信息,计算机接收了这些信息,对它进行处理、计算、变换和加工,最后再由计算机向外部设备输出加工好的信息。因此,计算机就其本质来说,可以称为信息处理机。计算机在处理信息的过程中,可以不需要人的干预而能自动进行,并且处理速度非常快。自动化和高速度是计算机处理信息的两个最突出的特点。

二、计算机的硬件组成

一台微型计算机通常由四部份组成:存储器、运算器、控制器和输入输出设备。由于运算器和控制器是计算机进行信息处理的中心,通常将它们合在一起称为计算机的中央处理单元(Central Processing Unit,常简称CPU),输入计算机的信息都是在这里处理的。在微型计算机中,中央处理单元CPU是做在一块集成电路芯片内,因而叫做微处理器(Microprocessor Unit,简称MPU)。CPU与存储器、输入/输出接口(常简称为I/O口)一起组成计算机的主机,而把输入/输出设备统称为计算机的外部设备。在外部设备中键盘是输入设备,荧光屏显示器、打印机等是输出设备,而把磁盘、磁带等称为外部存储器。

(1)输入输出设备

输入设备的功能是输入数据和程序,它与计算机之间的连接部分称为输入接口。输出设备的作用是输出计算机处理的结果,它与计算机的连接部分称为输出接口。因此,接口就是计算机与外部设备之间的连接部分,或者说是计算机与外部设备之间交换信息的通道。一般情况下,外部设备和计算机之间信息流通的接口应包括三个部分:暂存输入/输出数据的数据口(数据缓冲寄存器);用以存放外设忙、闲状态信息的状态口。控制外设启动和停止的控制口等。

计算机访问外设时,首先由CPU向该外设对应的接口发出要访问外设的地址和控制信号,检查该外设接口中的外设是否准备好。若准备好,则CPU通过数据总线与接口中的数据口交换数据,接口再通过它与

外设之间的I/O数据线与外设交换数据,从而完成了计算机对外设的一次访问。若未准备好,则等准备好后,再完成上述交换。所以,计算机对外设的控制,实际上是对接口的控制,外设的地址也是赋予其接口的。

(2)存储器

存储器是计算机中的重要部件,它的用途是存储数据和程序,既能把信息保存起来,又可把存入的信息取出来而又不破坏原存储内容,还可以重新记录新的信息,把原存储内容消去。向存储器存入数据,叫做写入。从存储器取出数据,叫做读出,对存储器进行读或写操作,叫做访问存储器。

存储器分内存和外存。微型计算机的外存储器主要是磁盘或磁带,而内存存储器又分为ROM(只读存储器)和RAM(随机存取存储器)。ROM只能读出数据,不能向它写入数据,主要用来存放计算机的系统程序和常数;RAM既可以对它写入数据,又可以从中读出数据,所以又称读/写存储器,它用来存放用户程序、数据及结果等。

内存存储器ROM或RAM内部含有很多存储单元,每一个存储单元都有唯一确定的编号,叫做地址。存储器是按地址存取信息的。当计算机要把一个数据存入某个单元中,或从某个单元中取出数据时,都要首先给出该存储单元的地址,然后查找此存储单元,查到后再根据指令进行数据的存取。地址好比一个教室有一个唯一确定的教室号码,到大楼里找人,要按照教室的号码去寻找一样。所不同的是一个单元能存储八位二进制数,又称一个字节数(字长为8)。对16位机或32位机来说,存储单元的字长则是16位或32位。

微型计算机的内存存储器是用大规模集成电路来实现的。最基本的单元电路是触发器,它有“0或1”两个不同状态,代表一位二进制数,因而一个存储单元(存放八位二进制数的)就需要8个触发器。由此构成的存储器称为半导体静态存储器。此外还有利用集成电路电容上有无电荷分别代表“0和1”状态的。因此构成的是半导体动态存储器,它必须周期性地自动将存储的数据重写,以维持电容器上的电荷,这种操作称为存储器的刷新。例如CEC-1的RAM是使用的HM50464动态存储芯片,每片存储容量为64K×4位,两片并联用,组成64K×8位的读/写存储器。读写存储器由存储体、读/写控制部分、译码驱动器三部分组成,CPU通

过控制总线、地址总线、数据总线来控制对存储器的读和写。

(3) 运算器

运算器是对数据进行算术运算(加、减、乘、除)和逻辑运算(与、或、异或)的部件。复杂的运算则是这几种基本运算的组合。

算术逻辑运算单元(ALU)是运算器的核心部分。ALU的两个操作数一个来自累加寄存器(简称累加器A),另一个来自某一个暂存寄存器。运算结果一方面通过CPU内部总线送往累加器A,同时将运算结果的标志送往标志状态寄存器P。由此可以看出累加器A在运算中的地位十分重要,它既是数据的来源地,又是算出结果所要送到的目的地。

运算器的取数操作和运算操作,都是在控制器的控制下进行的。

(4) 控制器

控制器是用来控制整个计算机进行计算的核心部件,它用来指挥和协调计算机各部分的工作。控制器的主要工作有:发出允许输入命令将程序和数据通过输入设备送入存储器中保存起来;发出取数命令从存储器中取数送入运算器;发出运算命令通知运算器完成计算工作;发出送数命令把数据或计算结果送入存储器;从存储器中取出计算结果,并发出输出命令控制输出设备将结果打印在纸上或显示在荧光屏上。总之,计算机能够自动地处理信息,按照一定的顺序工作,按照指令规定的内容执行相应的操作,全都离不开控制器的指挥和协同动作。

控制器由指令寄存器(IR),指令译码器(ID)和操作控制部件三部分组成。运行程序时,CPU从存储器取出指令,首先送入指令寄存器,然后再送入指令译码器,经过译码分析指令的内容,再根据这些内容,送入操作控制部件,产生一系列的控制信号,并在节拍发生器控制下,按一定时间顺序向计算机各有关部件发出执行命令,控制计算机按一定时序有条不紊地进行工作。

三、计算机信息流动和总线结构

前面指出,以微处理器MPU为核心,加上大规模集成电路的存储器、输入/输出接口等组成的微型计算机,再配以所需要的外部设备就构成了微型计算机硬件系统,硬件系统加上必须的系统软件(操作系统、语言处理程序、数据库管理软件、服务性软件等)和应用软件(用户程序)就构成了一个微型计算机系统。

那么软件和硬件之间,各逻辑部件之间,信息是如何流动的?这通常都是采用一束公共的信息传输线——系统总线来进行的。这就是说CPU通过系统总线实现与存储器、输入/输出接口之间的相互连接来交换信息。根据系统总线上传输信息的性质,系统总线又分为地址总线、数据总线和控制总线。

微处理器通过地址总线指定它要访问的内存储单元的地址码,或者输入输出口的地址码;通过数据总线

与存储器、输入/输出接口交换数据;通过控制总线发出控制信息,控制存储器和输入/输出接口工作。

因此,在计算机内部有两股信息在流动;

一是数据流,它包括各种原始数据,程序,现场信息等。它们由输入设备输入到计算机并存储于存储器中,在运算处理过程中,数据又从存储器读入运算器进行运算,运算结果再存入存储器或由输出设备输出。

二是控制信息流,它包括各种命令(或程序)。这些控制信息以数据的形式输入到存储器中,运行时再由存储器读入控制器,由控制器经译码后变为各种控制信号,来控制运算器进行各种运算和处理,控制存储器的读和写,控制输入/输出设备的启动和停止等。

四、微处理器的内部结构

6502MPU的内部结构可简单分为两大部分:寄存器部分和控制器部分。寄存器部分包括数据总线缓冲器、数据输入锁存器、程序计数器(PCL,PCH)、累加器(A)、算术逻辑运算单元、堆栈指示器(S)、变址寄存器(X,Y)、地址总线锁存器、处理器状态标志寄存器(P)等。寄存器的主要功能是完成对数据的传送、运算和处理。控制器部分包括时钟发生器、指令寄存器、指令译码器、中断逻辑电路等。控制器的主要功能是对读入微处理器的指令进行译码,确定执行该指令的周期数,提供各寄存器工作过程的控制信号,指定寄存器执行该指令的操作,以及改变当前指令的执行次序等。

五、微处理器内部寄存器

由上面分析可知,微处理器的实际处理工作是由寄存器部分完成的,为了更好地学习6502指令系统,我们必须对微处理器内部寄存器有一个全面的了解,特别是对6502中可供编程的几个寄存器加以关注。此外,微处理机中状态标志寄存器的标志位,是实现条件转移的依据,在程序设计中有非常重要的地位,学习中更应仔细和小心。

1. 累加寄存器 A(Accumulator)

累加寄存器是一个8位寄存器(可存储8位二进制数),用大写字母A作标志。它是计算机执行指令过程中的重要逻辑部件。从外部数据总线输入的数据,一般都暂时存放于A中,而在运算时才从A中取出,送到算术逻辑单元参与运算,其运算的结果仍送回A中。

由于寄存器在CPU内,取用它们的信息比取用CPU外部存储器的信息方便,速度也快,所以现代计算机都在运算器内设置这样的寄存器,用来存放参加运算的原始数据及中间结果,访问它和访问存储器一样,并可在它们之间进行数据传送和各种操作,所以又简称累加器。

与累加器A有关的指令有:实现内存与寄存器之间的数据传送,如LDA,STA;实现寄存器与寄存器之间的数据传送,如TAX,TXA,TAY,TYA;进行算术和逻辑运算的有ADC,SBC,AND,ORA,EOR;进行比较运

算的有 CMP;进行堆栈操作的有 PHA,PLA 等等。这些指令以后还要详细介绍。

2. 程序计数器 PC(Program Counter)

程序计数器用 PC 表示,它由两个 8 位计数器组成(可存放 16 位二进制数),一个是高位地址程序计数器 PCH,用以表示地址的高位部分;另一个是低位地址程序计数器 PCL,用以表示地址的低位部分。它们是 6502 中唯一的一个 16 位寄存器。

PC 中存放的是内存地址,它将到内存的相应地址中取出指令执行,每取出一个指令后,为取下一个指令字节作好准备,PC 中的内容要发生改变,通常是指向下一个指令首字节的地址。这样,就可以使程序按顺序运行。

当程序需要转移到新的地址(如执行转移指令、调用子程序或中断)时,PC 中将被放进要转移的目标地址值。由此可知,PC 控制着程序的执行或转向,因而也可称为指令地址指示器。

例如,指令 JMP (\$ 4000),其指令代码为 6C 00 40,表示重新设置 PC 值。假设内存单元 \$ 4000 的内容是 00,\$ 4001 的内容是 60,则执行 JMP (\$ 4000)指令后,PC 的值为 \$ 6000。

3. 变址寄存器(Index Register)

变址寄存器共有两个,每个可表示 8 位二进制数,分别用 X 和 Y 作标志。它们都是跟地址操作有关的寄存器,主要用于变址寻址方式中存放地址偏移量,也常在编程中当做一个计数器来用,它们可以用指令控制而被置成一个常数,也可以方便地用于加 1、减 1、比较、暂存、恢复数据等简单操作。

与变址寄存器有关的指令:由内存单元取数至变址寄存器的有 LDX,LDY;将变址寄存器中的数存入内存的有 STX,STY;累加器中内容与变址寄存器相互传送的有 TAX,TAY,TXA,TYA;堆栈指示器的内容与变址寄存器互换的有 TSX,TXS;变址寄存器内容进行加 1、减 1 操作的有 INX,INY,DEX,DEY;变址寄存器与存储器内容比较的有 CPX,CPY 等等。

4. 堆栈指针寄存器 S(Stack Pointer)

堆栈指针寄存器,用 S 作标志,它是用来指示堆栈栈顶位置的寄存器,即用于存放栈顶地址,以便进行堆栈操作。由于 6502 规定堆栈设置在第 1 页存储器中,所以堆栈指针 S 也是 8 位寄存器,只用来指出堆栈地址的低 8 位地址。当数据进栈时,堆栈指针 S 自动减 1;而在数据出栈时,栈指针 S 自动加 1。

和堆栈指针 S 有关的指令有:

- TSX, TXS;堆栈指示器内容与变址寄存器内容互换。

- PHA, PLA;对累加器 A 进行进、出栈处理。
- PHP, PLP;对标志寄存器 P 进行进、出栈处理。

5. 标志位寄存器 P(Processor Status)

标志位寄存器,用 P 作标志,这也是一个 8 位寄存器,其中第 5 位规定不用。

在程序设计中,每条指令在执行以后,往往发生进

位、溢出、以及结果为全零,或者结果为正数、负数的情况。指令执行后常常需要保留这些情况,以作为条件分支的根据,标志寄存器 P 就是为了适应这些情况而设置的。

这七个标志位是:

- C—进位标志(Carry),表明指令执行后最高位的进位状态。

C=1,表示有进位或无借位;

C=0,表示无进位或有借位。

- Z—零标志(Zero),表明刚才指令的执行结果是否为零。

Z=1,结果为零;

Z=0,结果非零。

- I—中断屏蔽或中断禁止标志(Interrupt disable),表明是否允许中断。

I=1,禁止中断;

I=0,允许中断。

- D—十进制运算标志(Decimal mode),表明采用 10 进制运算还是二进制运算。

D=1,算术指令作十进制运算;

D=0,算术指令作二进制运算。

D 的状态只对后续的加法或减法指令有作用,它不表示运算的结果,只是在运算前对数作出的规定。

- B—软件中断指令标志(Break flag),表示中断状态。

B=1,中断;

B=0,未中断。

- V—溢出标志(Overflow),表明溢出状态。

V=1,算术运算结果发生溢出;

V=0,无溢出。

- N—负数标志(Negative flag),表明一个数是否为负(一个带符号的 8 位二进制数,最高位为 1 时为负,否则为正)。

N=1,为负数;

N=0,非负数。

对标志位的学习应注意几点:

- 各条指令对标志位的影响不尽相同。

- 寄存器 P 中各标志位的状态一般是指当前指令执行后的状态,所以,标志位常常用在执行条件转移时作为条件判断的依据。

- 6502MPU 还可用专门指令对某些标志位进行置位或复位的操作。如 CLC 指令,其作用是清进位位,即对 P 寄存器中的 C 标志位置 0,而不影响其它各位;SEC 指令则使 C 标志位置 1,对其余各标志位均无影响。

六、内存地址分配

6502MPU 有 16 条地址线可以形成,可访问 64K 字节内存空间。但对中华学习机 CEC—I 来说,由于采用了存储体切换技术(存储空间的映射技术),故实际可寻址 96K 字节,其中 ROM 占 32K 字节,RAM 占 64K 字节。

1. CEC—I 系统的存储空间分配

CEC—I 机的内存系统共有 64K 存储空间,其内存地址分配为三部分:读写存储器 RAM (\$ 0000—\$ BFFF, 共 48K)、外部设备码输入/输出空间 (\$ C000—\$ CFFF, 共 4K)、存储体切换空间 (\$ D000—\$ FFFF, 共 12K),此空间可映射到 RAM,也可映射到 ROM)。

2. RAM 空间分配

CEC—I 机具有 64K 字节的 RAM 空间, $64K=2^8 \times 2^8=256 \times 256$,故常将 64K 空间划分为 256 页(0—255),每页有 256 个单元。其中高八位为 0 的页称为零页(地址范围是 \$ 0000—\$ 00FF);高八位为 1 的页称为第 1 页(地址范围为 \$ 0100—\$ 01FF),其余类推。RAM 的空间分配情况如下:

- \$ 0000—\$ 00FF,零页,是系统 RAM 区,用于存放监控程序、BASIC 解释程序和 DOS 系统数据。

由于 6502 的指令系统中采用零页寻址方式的指令很多,监控程序、Applesoft、整数 BASIC 和 DOS 都有使用零页的情况,因而只有很少的 10 余个单元未被系统占用,因此,在您编制汇编程序时,若要使用到零页地址,必须特别小心,千万不要破坏当前使用的系统程序占用的单元。若要安全地使用零页地址,可先将零页中要用的单元的内容存放在别处去,在使用这些零页单元后,回转系统程序之前,再设法恢复这些单元原先的内容。

- \$ 0100—\$ 01FF,第 1 页,是系统堆栈区,MPU 通常只访问堆栈的栈顶单元。由于该区只有 256 个单元,所以堆栈内最多只能存放 256 个字节的内容。

- \$ 0200—\$ 02FF,第 2 页,是键盘的一个输入行的缓冲区,也是 256 个单元。所谓输入行的缓冲区,是指当你从键盘打入程序、命令等时,在未按回车键之前,这些输入的字符都以 ASCII 码的形式存放在这个缓冲区内。

- \$ 0300—\$ 03FF,第 3 页,在这一区中,监控程序和 DOS 都使用了少数高地址单元作为向量地址,BASIC 程序中有些目标子程序也使用了该区的部分低地址单元。如果用户的机器语言程序不太长,可以放在第 3 页中,但应注意不能存放在 \$ 3D0 以后的存储单元,这是因为 \$ 3D0 以后的单元已被系统程序占用。

- \$ 0400—\$ 07FF,共 1K 字节存储空间,它是文本显示和低分辨率图形方式第一页的显示缓冲区。这 1024 个字节不能存放用户的程序和数据。

- \$ 0800—\$ 0BFF,也是 1K 字节存储空间,它是文本和低分辨率图形方式第 2 页的显示缓冲区。但该空间通常是用户 BASIC 程序和数据的存放区。

- \$ 0C00—\$ 1FFF,用户区,存放用户程序和数据。

- \$ 2000—\$ 3FFF,共 8K 内存,是高分辨率图形第 1 页显示缓冲区。如果不用高分辨图形显示时,该空间可以供用户程序使用。

- \$ 4000—\$ 5FFF,共 8K 内存,高分辨率图形第 2 页显示缓冲区,同样,在不用高分辨率图形显示时,该空间也可存放程序和数据。

应该说明的是,无论低分辨率图形或是高分辨率图形都安排了两页显示缓冲区,这主要是为了方便用户,可以利用它们进行画面的分页显示和动画显示。

- \$ 6000—\$ BFFF,是用户区。但在调用磁盘时,这个区域还要存放磁盘操作系统。例如,使用 DOS3.3 操作系统时,存储单元 \$ 9600—\$ BFFF 区域就被用来存放操作系统以及作为文件输入/输出缓冲区。

- \$ C000—\$ CFFF,是输入/输出设备占用的地址码区域。输入/输出大致分为两种:一是内核输入输出,二是外围扩展输入输出。

- \$ D000—\$ FFFF,是存储体切换空间。在这个地址范围内的 ROM 区,用于存放 BASIC 解释程序和监控程序,在这个地址范围内的 RAM 区用于存放整数 BASIC 或 PASCAL 操作系统。

3. 存储体空间的切换

把不同的存储模块映射到同一地址空间称为存储体空间的切换。在 CEC—I 机中,把 12K 地址空间 \$ D000—\$ FFFF 进行两层切换。一层是在这 12K 空间进行 ROM 和 RAM 之间的切换;另一层是在 \$ D000—\$ DFFF 这 4K 空间又进行两个 4K RAM 之间的切换。

12KROM 用于存放监控程序和 BASIC 解释程序,16KRAM 用来存放 BASIC 或 PASCAL 操作系统的一部分。

存储体的切换是通过改变软开关的状态来实现的。这可以实现三种选择:一是将 \$ D000—\$ FFFF 这 12K 空间切换(映射)到 ROM 还是 RAM;二是允许写 RAM 还是禁止写 RAM;三是将 \$ D000—\$ DFFF 这 4K 空间切换到第一个 4KRAM 还是第 2 个 4KRAM。软开关的地址为 \$ C080—\$ C08F。如果软开关地址最低位为“1”,则为写 RAM;为“0”时禁止写 RAM。如果软开关地址的 A_3 位为“1”时,则选 RAM 第一存储体中的 4K;若为“0”,则选 RAM 第二存储体中的 4K 等等。

设置软开关时应注意:

- 在 \$ D000—\$ DFFF 空间的两个 RAM 体,不能选一个读,另一个写,只能选择读、写同一个 RAM 体。

- 对 \$ D000—\$ FFFF 空间,不能选择读其中一部分空间的 ROM,而读剩余部分的 RAM。

- CEC—I 在加电或复位时, \$ D000—\$ FFFF 的初始状态被设置为允许读 ROM,允许写 RAM,4KRAM 空间在第二体。

例如,读两次 \$ C083,可以接通 1#、3#、4# 3 个 4K 体,并置为可读/写状态(可用 LDX \$ C083);用两次 LDX \$ C08B 指令接通 2#、3#、4# 3 个 4K 体,并置这 12K 空间为可读/写状态。在选通上述 RAM 时,ROM 将不起作用。而在切断 RAM,恢复 ROM 时,可用 LDA \$ C081 指令处理。

初级程序员级水平考试辅导问答

计算机硬件基础知识(续)

电子部十五所(100083) 顾育麒

[第4题] 一个用补码表示的二进制数,最高位为符号位,当符号位为“0”(即正数)时,符号位后面的数值是真实的值;当符号位为“1”(即负数),符号位后面的数值不是真实值,而是补码值,必须对它按位取反,然后在最低位加1,才得到真实的值,而且一定是一个负数。

$$[X]_{\text{补}} = 01110010$$

$$X = +(1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^1)$$

$$= +(64 + 32 + 16 + 2) = +114$$

$$[Y]_{\text{补}} = 10110101$$

符号位	按位取反
-----	------

$$1001010 + 1 = 1001011$$

$$Y = -1001011 = -(1 \times 2^6 + 1 \times 2^3 + 1 \times 2^1 + 1 \times 2^0)$$

$$= -(64 + 8 + 2 + 1) = -75$$

[第5题] 对于二进制数的代码10101101,若把它理解为无符号整数时,八个二进制数字都是数值部分,利用二进制数转换成十进制数的方法,能得到其对应的十进制数:

$$10101101 = 1 \times 2^7 + 1 \times 2^5 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^0$$

$$= 128 + 32 + 8 + 4 + 1 = (173)_{10}$$

若把它理解为用补码表示的有符号整数时,最高位是符号位,这个二进制数的代码的最高位是“1”,所以它是一个负数。这个8位的二进制数代码是一个负数的补码表示法,现在要求出其对应的十进制数。

$$[X]_{\text{补}} = 10101101$$

符号位	按位取反
-----	------

$$1010010 + 1 = 1010011$$

$$X = -1010011 = -(1 \times 2^6 + 1 \times 2^4 + 1 \times 2^1 + 1 \times 2^0)$$

$$= -(64 + 16 + 2 + 1) = -(83)_{10}$$

[第6题] 当X采用原码表示时,8位定点二进制整数所能表示的最大正数、最小负数:

最大正数 $[X]_{\text{原}} = 01111111$

(符号位)

$$X = (1111111)_2 = (127)_{10}$$

最小负数 $[X]_{\text{原}} = 11111111$

(符号位)

$$X = (-1111111)_2 = (-127)_{10}$$

当X采用补码表示时,8位定点二进制整数所能表示的最大正数、最小负数:

最大正数 $[X]_{\text{补}} = 01111111$

(符号位)

$$X = (1111111)_2 = (127)_{10}$$

最小负数 $[X]_{\text{补}} = 10000000$

(符号位)	按位取反
-------	------

$$1111111 + 1 = 10000000$$

$$X = (-10000000)_2 = (-128)_{10}$$

在思考这个问题时,可能错误地认为用补码表示的最小负数是 $[X]_{\text{补}} = 11111111$,如果将补码转换成真值X

$$[X]_{\text{补}} = 11111111$$

(符号位)	按位取反
-------	------

$$0000000 + 1 = 0000001$$

$$X = (-0000001)_2 = (-1)_{10}$$

显然,在8位定点二进制整数范围内,还能表示比 $(-1)_{10}$ 更小的负数,这个数不是最小负数。

[第7题] 如果逻辑表达式复杂,则电路设计时使用的器件就多,电路复杂;如果逻辑表达式简单,则电路设计时使用的器件就少,电路简单,因此在电路设计时要将逻辑表达式化简。

逻辑表达式化简到什么程度,可以认为是“最简”的逻辑表达式呢?

首先,乘积项的个数应该是最少的;
其次,在满足乘积项个数最少的条件下,要求每一个乘积项中的变量最少。

常用的化简方法是公式化简法。

下面列出逻辑代数的基本公式:

1. 关于变量和常量关系的等式:

[公式1] $A + 0 = A$ [公式1'] $A \cdot 1 = A$

[公式2] $A + 1 = 1$ [公式2'] $A \cdot 0 = 0$

[公式3] $A + \bar{A} = 1$ [公式3'] $A \cdot \bar{A} = 0$

2. 满足交换律、结合律、分配律

[公式4] $A + B = B + A$ (交换律)

[公式4'] $A \cdot B = B \cdot A$ (交换律)

[公式5] $(A + B) + C = A + (B + C)$ (结合律)

[公式5'] $(A \cdot B) \cdot C = A \cdot (B \cdot C)$ (结合律)

[公式6] $A \cdot (B + C) = A \cdot B + A \cdot C$ (乘对加的分配律)

[公式6'] $A + B \cdot C = (A + B) \cdot (A + C)$ (加对乘的分配律)

3. 逻辑代数的一些特殊规律

[公式7] $A + A = A$ (重叠律)

[公式 7'] $A \cdot A = A$ (重叠律)

[公式 8] $A + \bar{A} = \bar{A} \cdot \bar{B}$ (反演律)

[公式 8'] $\bar{A} \cdot \bar{B} = \bar{A} + \bar{B}$ (反演律)

[公式 9] $\bar{\bar{A}} = A$

当我们初步掌握了这些基本概念和基本公式以后,就可以做这道题了。

$$\begin{aligned} F &= A \bar{B}C + A \bar{B}\bar{C} + ABC + A \bar{B} \\ &= AC(\bar{B} + B) + A(\bar{B} + \bar{C}) + A \bar{B} \\ &= AC + A \bar{B} + A \bar{C} + A \bar{B} \\ &= AC + A \bar{B} + A \bar{C} \\ &= A(C + \bar{C}) + A \bar{B} \\ &= A + A \bar{B} \\ &= A(1 + \bar{B}) \\ &= A \end{aligned}$$

在化简的第二步,应用了反演律。

$$\bar{A} \cdot \bar{B} = \bar{A} + \bar{B} \quad [\text{公式 8}]$$

$$\bar{A} + \bar{B} = \bar{A} \cdot \bar{B} \quad [\text{公式 8}']$$

使得逻辑表达式的第二项

$$A \bar{B}C = A(\bar{B} + \bar{C}) = A \bar{B} + A \bar{C}$$

为下一步的化简工作,提供了有利条件。

反演律是逻辑代数中的一组重要公式,又称为摩根定理,在电路设计、化简逻辑表达式时经常要用到这组公式。为了便于读者理解,并能灵活运用它,这里用列出真值表的方法,证明它的正确性,见表 1

表 1 真值表

A	B	$A \cdot B$	$\bar{A} \cdot \bar{B}$	\bar{A}	\bar{B}	$\bar{A} + \bar{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

从表 1 中可以看到 $\bar{A} \cdot \bar{B}$ 和 $\bar{A} + \bar{B}$ 在变量 A、B 的任何组合状态下都有相同的取值,这就证明了 [公式 8] $\bar{A} \cdot \bar{B} = \bar{A} + \bar{B}$ 的正确性。

用同样的方法列出表 2。

表 2 真值表

A	B	$A + B$	$\bar{A} \cdot \bar{B}$	\bar{A}	\bar{B}	$\bar{A} \cdot \bar{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

从表 2 中可以看到 $A + B$ 和 $\bar{A} \cdot \bar{B}$ 在变量 A、B 的任何组合状态下都有相同的取值,这就证明了 [公式 8'] $A + B = \bar{A} \cdot \bar{B}$ 的正确性。

反演律可以推广到多个变量的场合,例如:

$$\bar{A} \cdot \bar{B} \cdot \bar{C} \cdot \dots \cdot \bar{M} = \bar{A} + \bar{B} + \bar{C} + \dots + \bar{M}$$

$$\overline{A + B + C + \dots + M} = \bar{A} \cdot \bar{B} \cdot \bar{C} \cdot \dots \cdot \bar{M}$$

在 1989 年和 1990 年的全国初级程序员水平考试的试题中,都有化简逻辑表达式的试题,难度适中,因此,在复习时要注意掌握逻辑代数的基本公式和逻辑代数的一些特殊规律,同时要做一些练习题,以帮助理解基本公式的应用方法。

[第 8 题] 运用逻辑代数的基本公式进行化简:

$$AB + \bar{A}C + BC = AB + \bar{A}C + (A + \bar{A})BC$$

这里采用了公式化简法中的“配项法”,在第三项配以因子 $(A + \bar{A})$,由 [公式 3] $A + \bar{A} = 1$,所以逻辑表达式的含义不变。

$$\begin{aligned} AB + \bar{A}C + BC &= AB + \bar{A}C + (A + \bar{A})BC \\ &= AB + \bar{A}C + ABC + \bar{A}BC \\ &= (AB + ABC) + (\bar{A}C + \bar{A}BC) \\ &= AB(1 + C) + \bar{A}C(1 + B) \\ &= AB + \bar{A}C \end{aligned}$$

这道题的答案应选择 (2)

[第 9 题] 中央处理器 (CPU) 是计算机的重要部件,它由运算器、控制器组成,能执行规定的计算机基本操作指令。

控制器是计算机的指挥中心,它的基本功能是取指令,分析指令格式,按指令的要求产生相应的控制电平 and 脉冲,指挥协调计算机内各部件工作,完成指令规定的操作。

指令寄存器的作用是保存当前正在执行的一条指令。这条指令是新从存储器取出来的。

程序计数器的作用是用来存放后继指令地址。也就是说,它存放一个地址。所谓后继指令是指后面要继续执行的指令,按程序计数器提供的地址,从相应的存储单元中取出来的内容,就是要执行的指令和要操作的数据。一般指令是顺序存放在存储器中的,所以,程序计数器的内容自动增加,以指向下一条指令的地址。程序计数器也叫做指令地址计数器。

控制器根据人们预先编写好的程序,依次从存储器中取出指令,放在指令寄存器中,通过指令译码器进行译码分析,确定应该进行什么操作,通过微操作控制线路产生相应的微操作控制信号,在确定的时间,往确定的部件,发出确定的信号,使运算器和存储器等部件自动而协调地完成该指令所规定的操作。当这条指令完成以后,再按程序计数器中的地址从存储器中取出下一条指令,并照此同样地分析与执行该指令,如此重复,直到完成所有的指令为止。

这道题的第一个填空应选择答案 (2),第二个填空应选择答案 (1)。

[第 10 题] 指令是计算机用以控制各个部件协调动作的命令。每种机型都有自己的一套指令,称为该机型的指令系统。指令由两部分组成:操作码和地址码。操作码指明这条指令应执行的操作种类;地址码指明这条指令参与运算的数或数所在的单元地址。

查找指令操作数的方法就是寻址方式。最简单的寻址方式是:指令中的地址码就是操作数在主存储器

中存放的有效地址。允许访问主存储器的最大空间决定于指令中地址码的位数。例如地址码 16 位表示可直接访问的内存最大容量为 2^{16} , 即 64K。对于指令字较短的指令, 能够访问的操作数是有限的。为了扩大指令允许访问的内存空间而又不增加指令字的位数, 提供编制程序的灵活性, 在计算机指令系统中采用多种不同的访问内存存储器的方式, 对指令的地址码, 进行变换, 求得操作数的有效地址, 然后才去访问内存存储器。

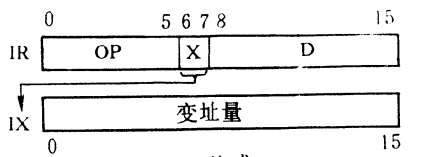
因此, 这道题的答案是(2), 指令系统中采用多种不同的寻址方式的主要目的是缩短指令长度, 扩大寻址空间, 提高编程的灵活性。

一个指令系统中有若干种寻址方法, 例如: 直接寻址、变址寻址、间址寻址等等。某一条指令使用哪一种寻址方式, 通常要在地址码字段中专门设置一个寻址方式字段。由该字段中的二进制数码指定具体的寻址方式。

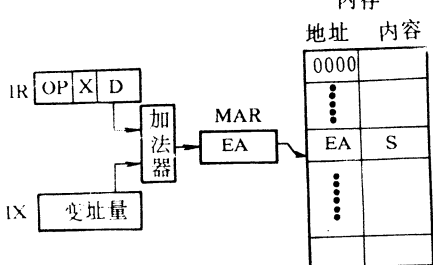
[第 11 题] 为了扩大可访问的主存空间, 希望增加有效地址的位数。硬件上又不允许随意增加指令字的长度和形式地址的位数。因此, 提出一种新的寻址方式—变址寻址。

这种寻址方式的实质是进行一次地址加法, 使形式地址(位数较少)与某个指定的寄存器内容相加, 求得有效地址。这个被指定的寄存器叫做变址寄存器, 变址寄存器的内容叫变址量。

变址寄存器的位数较多, 与形式地址相加所得到的有效地址位数也较多, 因此可访问的地址空间就扩大了。例如, 8 位形式地址与 16 位变址量相加, 得到 16 位的有效地址, 则指令可访问的主存空间为 $2^{16} = 64K$ 。



(a) 格式



(b) 结构

- IR-指令寄存器
- IX-变址寄存器
- MAR-内存地址寄存器
- EA-有效地址
- OP-操作码
- X-变址寻址特征
- D-形式地址
- S-操作数

图 1

变址寻址方式的示意图如图 1 所示。这道题的答案应选择(3)

[第 12 题] 当指令地址码指出的主存单元的内容不是操作数, 而是存放操作数的地址时, 则称由指令地址码所决定的地址为间接地址。

要寻找操作数的地址, 必须按照指令给出的间接地址去访问主存, 然后把这个主存单元的内容当作操作数的存放地址。这种寻址方式叫做间接寻址方式。

指令中必须给出间接寻址的标志, 以便与直接寻址相区别。

直接寻址时, 由指令地址码所决定的地址, 就是操作数的地址, 这个地址可以用直接寻址、变址寻址、相对寻址等方法来决定; 间接寻址时, 由指令地址码决定的只是某个操作数地址的地址。这个地址也可以用直接、变址等方法来确定。

间接寻址指令应包括如下几部分:

OP	I	X	D
----	---	---	---

- OP 操作码
- I 间接寻址特征
- X 变址寻址特征
- D 形式地址

当 $I=0$ 表示直接寻址; 当 $I=1$ 表示间接寻址。

间接地址 $IA = (IX) + D$

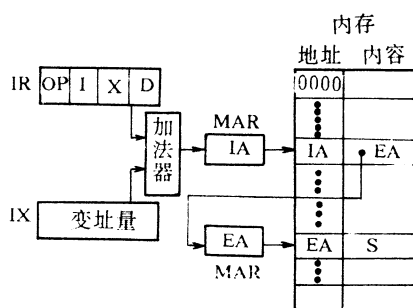
其中 (IX) 表示变址寄存器的内容。当不变址时, $(IX) = 0$

操作数的有效地址 $EA = (IA)$

间接寻址的过程如图 2 所示。

这道题的答案应选择(2), 在间接寻址方式中, 操作数在主存单元中。

[第 13 题] 对于微型计算机来说, 空气中的相对湿度不能超过 80%, 若超过这个值则会发生结露现象



- IR-指令寄存器
- IX-变址寄存器
- IA-间接地址
- MAR-内存地址寄存器
- S-操作数
- EA-有效地址

图 2

象,使元器件受潮变质,甚至发生短路,损坏机器;若空气中的相对湿度小于20%,则因过于干燥,使微机系统容易产生静电干扰,引起机器的错误动作。因此,机房的相对湿度应控制在20%~80%之间。这道题的答案应选择(2)。

[第14题] MIPS(Million Instructions Per Second)值是人们用于表示计算机性能的一种方法。若其值为1MIPS,表示计算机每秒能执行100万条指令。

不同的计算机制造厂,在设计计算机时,都有一定的针对性,有的侧重于科学计算;有的侧重于商业管理,有的侧重于图形处理。所以制造厂在选择测试MIPS值的基准程序时,各有不同。因此不同厂家制造的计算机,MIPS值的含义不同,不一定能说明性能的高低。因此,在选购机器时,不仅要看MIPS值,还要了解计算机的其它性能,综合比较,才能区别性能高低。

同一个厂家生产的机器,用相同的基准程序测试得到的MIPS值,具有可比性。MIPS值越大,说明机器的速度越快。

表3列出DEC公司的VAX系列计算机的MIPS值。

表3列出VAX 11/780计算机的MIPS值为1.06,它表示每秒能执行106万条指令。DEC公司的VAX系统均以VAX 11/780机的速度作为衡量该公司其它机器的性能单位,用VAX性能单位VUP(VAX Unit of Performance)表示。例如,VAX 9000计算机单个中央处理器的性能等于40倍VUP;VAX 9000—440型计算机为150倍VUP;VAX 6000型计算机系统性能范围可以从2.6倍VUP增加到36倍VUP。

表3 DEC公司计算机的MIPS值

系统性能	VAX 11/780	VAX 8300	VAX 8700	VAX 6000	VAX 9000
MIPS	1.06	1.70	6.36	3.0~38.2	42.4~159.0
VUP	1.0	1.6	6.0	2.8~36	40~150

这道题的答案应选择(2)

[第15题] MTBF(Mean Time Between Failure)是计算机的可靠性指标,称为平均无故障工作时间。有的资料上称为平均故障间隔时间,其实两种说法的含义是一样的。

可靠性是指在规定的时间内,规定的条件下完成规定功能的能力。这是一个定性的要求。能力究竟有多大?要有定量的指标来描述。

可靠度是指在规定的时间内、规定的条件下完成规定功能的成功概率,成功概率是个定量指标,它是一个统计量,可以度量 and 比较。对于计算机系统来说,可靠度定义中的“完成规定功能”是指计算机系统的技术性能,在生产实践中,用计算机系统的平均无故障工作

时间MTBF来表示。

因此,这道题的答案应选择(2)。

[第16题] 计算机的内存储器按性能和用途分类,可以分成两大类:只读存储器ROM和随机存取存储器RAM。

只读存储器ROM的特点是:它存放在其中的信息是固定的,即使由于关机或停电,切断电源后,它所存储的信息不会丢失。再次接通电源后,它所存储的信息立即可以读出使用。要求在通电后立即可以取用的指令和数据都预先存放在ROM中。例如计算机内存储器中常驻的监控程序、引导程序、加载程序、管理程序、常用的数据表格等。

随机存取存储器RAM的特点是:既可以“读出”其中存放的信息,又可以将新的信息“写入”其中。一旦断电,则所存储的信息全部丢失。再次接通电源后,其中已没有有用的信息。若开机后不断电,计算机处于正常工作状态,则“写入”的信息就能保持。

只读存储器ROM所存储的信息,是事先写入的,根据写入信息的方式不同,又可以分为下列几种:

1. ROM:不可编程的只读存储器。这种ROM所存储的信息,是在芯片生产的最后工序时由生产工厂“写入”的,芯片封装后,无法更换信息,这种ROM使用可靠,成本低。

2. PROM:可编程的只读存储器。它所存储的信息可以在芯片封装完成后,再编程写入。常用的方法是“烧入法”,用编程用的强电流脉冲,将芯片内需要烧断的熔丝烧断,熔断的位为逻辑0,未熔断的位为逻辑1。信息一旦被写入,熔丝已烧断,再想改写也不可能了,写入的信息就永久固定下来。

3. EPROM:可擦除改写的只读存储器。用户不但可以“写入”信息,还可以用紫外光通过封装上的玻璃窗口照射芯片,使各位全部恢复为1,亦即将“写入”的信息清除掉,以便重新写入信息。

4. EEROM或E²ROM:可电擦除只读存储器。

这道题的第一个填空应选择答案(2);第二个填空应选择答案(1)。如果题型改变,在供选择的答案中有PROM、EPROM等,那么也属于第一个填空可选择的答案。

[第17题] 目前在微型计算机上配置的软盘驱动器规格如表4所示。

表4 磁盘驱动器的品种和规格

直径(英寸)	说明	容量
5.25	单面	160KB/180KB
5.25	双面	320KB/360KB
5.25	高密度	1.2MB
3.5	双面	720KB
3.5	双面	1.44MB

微型计算机上使用的软盘片品种和规格如表5所示。

表 5 软盘片的品种和规格

直径 (英寸)	说 明	存储容量	磁道 数	磁头 数	每道 扇区数	每个扇 区字节 数
5.25	单面,倍密度	160KB/180KB	40	1	8/9	512
5.25	双面,倍密度	320KB/360KB	40	2	8/9	512
5.25	双面,高密度	1.2MB	80	2	15	512
3.5	双面	720KB	80	2	9	512
3.5	双面	1.44MB	80	2	18	512

用 DOS 命令到软盘片中去读写时,必须考虑软盘片和软盘驱动器的兼容性。具体规定如下:

1. 160KB/180KB 单面 5.25 英寸软盘驱动器。

可以在这种软盘驱动器上读写的软盘片,只有 160KB/180KB 单面,倍密度 5.25 英寸软盘片一种。

2. 320KB/360KB 双面 5.25 英寸软盘驱动器。

可以在这种软盘驱动器上读写的软盘片有: 160KB/180KB 单面,倍密度 5.25 英寸软盘片; 320KB/360KB 双面,倍密度 5.25 英寸软盘片。

3. 1.2MB 高密度 5.25 英寸软盘驱动器。

可以在这种软盘驱动器上读写的软盘片有: 160KB/180KB 单面,倍密度 5.25 英寸软盘片; 320KB/360KB 双面,倍密度 5.25 英寸软盘片;

1.2MB 高密度 5.25 英寸软盘片。

4. 720KB 双面 3.5 英寸软盘驱动器。

可以在这种软盘驱动器上读写的软盘片只有 720KB 双面 3.5 英寸软盘片一种。

5. 1.44MB 双面 3.5 英寸软盘驱动器。

可以在这种软盘驱动器上读写的软盘片有:

720KB 双面 3.5 英寸软盘片;

1.44MB 双面 3.5 英寸软盘片。

这道题的答案应选择(2)、(3)。

(上接第 29 页)

程序、显示、前置放大器、A/D 等器件的功耗,才能真正达到高性能、低功耗的效果。

这里主要是有两方面要注意:

(1)改变键盘扫描方式

一般智能仪器中 CPU 处理数据占的整个时间并不多,而扫描键盘、显示是经常不断进行的。这种用程序查询的方式扫描键盘和显示器很不经济。这时 CPU 永远处于忙的状态,不可能进入空闲状态。为了减少功耗,让 CPU 在平时处于空闲方式,并开放外部中断。一般将键盘设计成中断方式,利用外部中断 0 控制键盘,只要有键按下,向 CPU 请求中断,使 CPU 退出空闲状态,进入正常状态。CPU 响应中断后,为了防止重新按键,首先禁止外部中断 0,然后进行键盘扫描,根据按

[第 18 题] 磁带机的数据传送速度=读写磁带的走带速度×磁带记录密度。

磁带机的数据传送速度用每秒传送的比特数 BPS (Bits Per Second)表示;读写磁带的走带速度用每秒走带的英寸数 IPS 表示;磁带记录密度用每英寸存储的比特数 BPI(Bits Per Inch)表示。

这台磁带机的数据传送速度为:

$$45 \times 800 = 36000 \text{ BPS}$$

这道题的答案应选择(2)。

[第 19 题] 在输出设备中,打印输出设备的应用十分广泛,几乎所有的计算机系统都配置有打印机。

打印输出设备按其打印方法的不同,可分为击打式和非击打式两种。

击打式打印机是利用打印针击打色带和打印纸,而打印出由点阵构成的字符、汉字、图形。点阵针式打印机就是这种类型。它的打印头由若干根针组成。常用的有 9 针、24 针,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。

非击打式打印机是采用物理或化学方法印出字符。它具有高速、无噪声印字等优点。激光印字机、热敏印字机、喷黑印字机等都是这种类型。常用的激光印字机是将激光扫描技术和电子照相技术相结合的非击打式打印输出设备。

这道题的第一个填空应选择答案(3);第二、三个填空应选择答案(1)(2)。

[第 20 题] 这道题的答案应选择(3)。

“计算机辅助”的概念是指依靠计算机技术来得到比人工方法更好、更快、更精确的设计、教学、管理、制造过程。

CAD(Computer Aided Design)是计算机辅助设计的简称。它的含义是使用计算机系统来辅助一项设计的形成、修改、分析和优化。它的应用范围很广,例如:机械 CAD、建筑 CAD、电气设计 CAD 等。

下的相应键去处理相应的功能。处理完后返回,并开放中断 0 和置 PCON. 0=1,使 CPU 重新进入空闲方式。这样可以大大降低系统的功耗。硬件一般用多个二极管组成或门(对逻辑 0 来说)或者用多输入端的与门解决。

(2)使用液晶显示

用 LED 作为数字,地址的显示时,一般耗电 200MA—300MA。而要使整个系统降低功耗,一般使用液晶显示。液晶分静态和动态两种,两种驱动电路都比较复杂,具体电路将在另一节讲。

总之我们认真做好各个环节节电,使整个 80C31BH 应用系统总耗电在 10MA 以下是完全可行,也是不困难的。



低功耗单片机最小系统

合肥中国科技大学计算中心(230026) 张培仁 刘振安

CMOS 电路具有低功耗,低速度,高抗干扰能力。它一般比 TTL 电路功耗小 100 倍,速度慢 10 倍。近几年来出现既具有 CMOS 电路优点又兼有 TTL 的速度的 CHMOS 电路。相应出现了如 80C31BH,80C51BH 等单片机,也越来越受人们的重视。特别象野外,井下,无人值守监测站等非常需要低功耗的单片机系统。

对于片内无程序存储器的低功耗单片机 80C31BH 必须配上片外程序存储器。这时要求配置外围芯片也应该是低功耗芯片。这种单片机最小系统由三片组成即 80C31,27C64,74HC373。电路如图 1 所示。

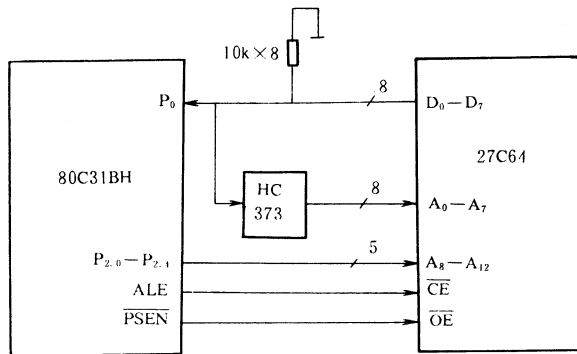


图 1

单片机在节电状态运行功能,工作方式如何设定,以及在低功耗系统中应注意什么,我们对这些问题作一个简单介绍。

1. 单片机节电运行功能

80C51BH,80C31BH 具有正常,空闲、掉电三种工作状态。它们速度快,功耗是 8051,8031 的 1/10。80C51BH,80C31BH 的 CPU 不是静态而是动态,但 80C51,80C31 片内的 RAM 是静态的。故时钟频率不能太低。

几种工作方式耗电量对比表如表一所示。

表一

时钟频率	工作方式	8031	80C31BH
12MHz	正常	125mA, +5V	24mA, +5V
12MHz	空闲	没有	3mA, +5V
	掉电	没有	50μA, 2V

空闲和掉电二种节电方式均可通过软件来选择运行,仅在需要正常工作时,才进入正常运行,其它时间均保持在空闲或掉电方式下。这样大大减少单片机的功耗。

空闲方式下,CPU 停止工作,而 RAM,定时器/计

算器,串行口和中断系统继续工作。

2. CHMOS 单片机的节电工作方式

CHMOS 单片机提供三种工作状态是由图 2 的硬件电路实现的。

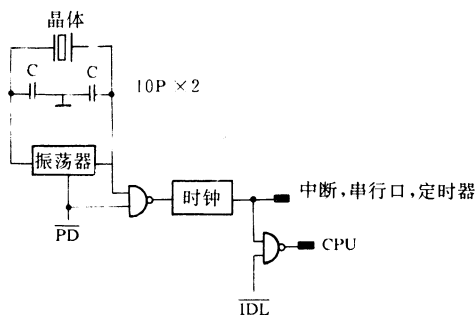


图 2

由图可见,当 $\overline{IDL}=0$ 时,时钟仍继续工作,中断系统,串行口,定时/计数器电路继续由时钟驱动,但此时时钟信号不送往 CPU,即 CPU 处等待状态,称之为空闲工作方式。当 $\overline{PD}=0$ 时振荡器停止工作,只有片内 RAM 和 SFR 的内容被保存,这就是掉电工作方式。这两种工作方式都是由 SFR 中的电源控制寄存器 PCON 设定的。

MSB

LSB

SMOD	-	-	-	GF1	GF0	PD	IDL
------	---	---	---	-----	-----	----	-----

PCON 是电源控制寄存器,地址为 87H。各位定义如下:

SMOD(PCON. 7):串行口控制用的波特率倍增位。

GF1,GF0:通用标志位。

PD(PCON. 1):掉电方式位。当 PD=1 时,进入掉电方式。

IDL(PCON. 0):空闲方式位。当 IDL=1 时,进入空闲工作方式。

若 PD 和 IDL 同时为 1,则先进入掉电方式。

复位时,各控制位全为 0。

PCON 不能位寻址,只能进行字节寻址。

空闲工作方式

空闲工作方式下,CPU 进入等待状态。此时仍继续驱动中断系统,定时/计数器,串行口。CPU 内部的状态,堆栈指针(SP),程序计数器 PC,程序状态字 (PSW),累加器,片内 RAM 的状态完整保留下来。这时也称之为冻结工作方式。

单片机处于空闲状态时, ALE, \overline{PSEN} 电平为“1”当我们外部取指时, 就象图 1 所示 HCMOS 的三片最小系统时, 进入空闲状态 P0 口呈高阻状态, P2 口状态是进入空闲前 PC 高八位地址的状态 (PCH), P1, P3 口是进入空闲状态之前的状态。CPU 从空闲状态回到正常工作方式的条件是用中断或硬件复位。

掉电方式

在掉电工作方式 ($\overline{PD}=0$) 下, 由于 $\overline{PD}=0$ 的作用, 片内振荡器停止工作, 因此单片机所有运行状态都停止, 只有片内 RAM 中的数据保存起来。

处于掉电时 ALE, \overline{PSEN} 均为低电平 (逻辑 0)。I/O 口引脚状态见表二。

退出掉电工作方式, 只能用硬件复位, 复位操作时将重新定义所有 SFR, 但不改变片内 RAM 的内容。

表二

引脚	空闲	掉电
ALE	1	0
\overline{PSEN}	1	0
P0	高阻	高阻
P1	SFR 数据	SFR 数据
P2	PCH	SFR 数据
P3	SFR 数据	SFR 数据

(有片外 EPROM, 是外部取指)

3. 低功耗单片机系统要注意的问题

80C51BH, 80C31BH 有很多优点。但在实际的应用中尚要注意以下几个问题:

(1) 可控硅导通现象

由于 CMOS, CHMOS 电路半导体工艺容易产生 pnpn 寄生结构。这种结构正好构成可控硅, 一旦产生闭锁, CMOS 的电流可大于 100mA。这种短路现象无法用软件恢复, 只有断电后才可恢复。但 80C31 由于工艺改进, 已经大大减少了这种现象。一般在电源中串联一个小于 50 Ω 的电阻, 以使由于工艺制作原因产生的可控硅触发电流达不到触发水平 (即达不到产生维持可控硅现象的触发电流), 从而避免了可控硅失控现象。

(2) 未使用管脚的处理。

P1, P2, P3 口因片内都有上拉电阻, 所以在未使用时可不做任何处理。但 P0 在片内无上拉电阻, 必须在片外接上拉或下拉电阻。这是因为当复位信号过后, 由于 P0 口用作数据/地址总线, 其引脚上的电平总是有定义的。然而当 80C31BH 处于空闲或掉电方式时, 由于它不执行取指令操作, 故当不外加上拉电阻或下拉电阻时, P0 口引脚悬空, 一旦恢复正常工作方式, CPU 要从 P0 口取指令可能会出错。

上拉和下拉电阻一般在 2K Ω ~50K Ω 之间, 一般可选 10K Ω 。电阻太小时, 对输出低电平影响较大, 功耗也大些; 电阻太大对 P0 口的高电平影响比较大。一般接下拉电阻比上拉电阻可减少功耗效果是一样的。

(3) P1, P2 和 P3 口的驱动能力

80C31BH 中 P1, P2, P3 口都接了三个上拉电阻 (片内接的), 电阻分强, 中, 弱。强、中都是由脉冲控制接通的, 弱是一直接上的 (大小一般为 20K Ω)。这三个电阻组成端口的驱动能力。驱动电流最大在 650 μ A 左右。

(4) 主振频率对功耗的影响

功耗随着主振频率的提高而增加。频率越高, CMOS 管处于放大区的时间占一个周期的整个时间越多, 功耗也随之增加。理论上在 500KHz 时功耗最小, 频率太低不行, 因为 80C31 中 CPU 是动态的, 数据会丢失。

(5) 在 80C31BH 的最小应用系统中对 27C64 的 \overline{CE} 片选信号处理。

在空闲状态时, 27C64 的片选信号 \overline{CE} 不接地, 而与 80C31BH 的 ALE 端相连, 以便在 80C31BH 进入等待工作方式时, ALE 变为高电平, 使 27C64 进入低功耗备用状态, 如图 1 所示。

在掉电方式时, 如系统的备用电源仅给 80C31BH 供电, 那么掉电时, ALE, \overline{PSEN} 应为低电平。但如果在电源掉电期间仍给 EPROM 供电, 则需禁止 27C64。在正常工作时 P2.7 输出为 0, 选中 27C64。掉电时 P2.7 = 1 不选中 27C64。如果既有掉电方式也有空闲方式时 27C64 的 \overline{CE} 端接法如图 3。

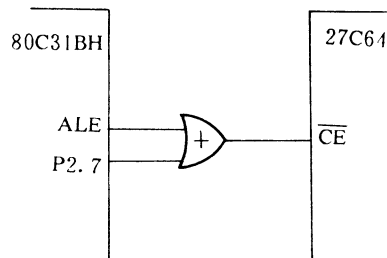


图 3

(6) 空闲前的准备

进入空闲状态前要把准备允许哪个中断源激活 CPU 确定下来, 空闲前要用软件置这些中断源为允许, 当然总中断源也一定开放。

空闲后要端口处于什么状态, 也要在启动空闲前写入端口。原则有两个: 一是维持整个系统正常工作, 二是尽可能降低功耗。例如 P1 口中有几位在应用系统中是控制某些继电器的, 则要置 1。进入空闲时, 这些还是要保持 1, 而其它一些空闲时可能没用, 可以置 0。

空闲时一般不选中 27C64, PC 值在空闲时一般不写入 0000H。

(7) 掉电前的准备和掉电后恢复时的复位时间。

掉电前的准备工作基本和空闲前准备工作相似。掉电后再复位, 信号必须有足够时间使主振荡器重新振荡并稳定下来, 一般需要 10ms 左右。

对于一个应用系统来讲, 只考虑以 80C31 的最小系统的功耗还是不够的, 还应考虑整个系统如键分析

(下转第 27 页)

新一代超高集成度 Z80 微处理器系列

ZILOG 中国代理
中国电子器材公司特区公司 董伯明

Z80 是世界上和我国流行最广的一种八位微处理器。我国广大的工程技术和科研人员几乎都学过和应用过该系列产品,时至今日,我国大专院校有关计算机的教材中,还是以 Z80 系列产品为例进行教学。许多工业控制、尖端电子产品、各种智能终端等等场合仍然大量应用 Z80 系列产品。

近年来,随着 VLSI 工艺水平的不断提高,Zilog 公司采用 Assps(特殊应用标准产品)的先进设计手段,推出了软件与原 Z80 完全兼容的超高集成度 Z80 处理器系列。该系列产品具有高度集成,高度灵活,高速,高性能等特点,又具有很高的性能/价格比,故而在世界上已广泛地应用在各种控制与通信的领域。目前,我国刚开始接触,为了加速这一系列的先进产品在我国推广应用,我们准备从今年开始,继续在“电子与电脑”的“学装微电脑”专栏上介绍这一系列产品的性能及应用技术。相信原有通用型 Z80 的广大用户很快就会步入与掌握新一代超高集成度 Z80 微处理系列。附表介绍新一代超高集成度 Z80 微处理器性能(见封三)。

该系列的产品共有九个品种,采用 NMOS 和 CMOS 二种工艺生产,凡产品型号中有 C 的均是 CMOS 工艺生产的,而没有 C 的是采用 NMOS 工艺。其中:

Z84C01 是该系列产品中最基本的一种产品,它由 Z80 CPU 与 CGC(时钟发生,控制器)组成。

Z84C50 是在 Z84C01 的基础上再加 1KB 的静态 RAM 和等待状态发生器二个模块。

Z84C90 是 Z80KIO(Killer I/O),称为通用的或专用型的多用途 Z80 I/O 电路,意味着只要有了这片电路后,可以取代各种 I/O 电路,它内含二个独立的同步或异步的串行通道,三个 8 位的并行端口,四个独立的定时器/计数器,一个并行接口适配器和片内振荡器,时钟驱动电路等。

Z84011/C11 是有 NMOS 和 CMOS 二种工艺生产的产品,它内含 Z80 CPU,时钟发生、控制器,四个独立的定时器/计数器,五个独立的 8 位并行端口,尚有 WDT(Watch Dog Timer)控制定时器模块,该模块启动后,对于由电源或其它硬件而造成的软件出错,具有缩短出错周期的作用,换言之具有抗干扰的功能。

Z84013/C13 亦采用 NMOS 与 CMOS 二种工艺生产,称之为智能化外设控制和增强型智能化外设控制器。该芯片内含 Z80 CPU,四个独立的计数器/定时器,二个独立的 SIO,一个时钟产生器,控制器,一个监视定时器。

Z84015/C15 亦是有 NMOS 和 CMOS 二种工艺生

产的智能外设控制器,但比 Z84013/C13 增加了二个独立的 8 位并行的 I/O 端口。

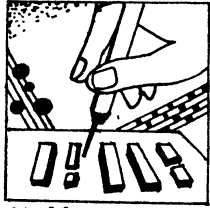
Z80180 是一块高性能的 Z80CPU 与外设接口的组合,称之为 MPU(微处理器单元)。该芯片内含一个增强型的 Z80 CPU 工作频率可高达 12.5MHz,有一个可达 1MB 的存储器管理单元 MMU,二个独立的直接存储器存取 DMA 通道,二个通用的异步发送器/接收器,二个 16 位的计数器/定时器,一个定时的串行 I/O 端口,片内振荡器,等待状态发生器,中断控制器等,实际上原来通用型 Z80 芯片组成的单板机的功能就由这一块芯片完成了,是用 CMOS 工艺制造的。

Z80280 是又一块高性能的微处理器单元 CPU,用 CMOS 工艺制造,它有三态的指令流水线,采用 16 位的 CPU 结构形式,机器码与通用型 Z80 微处理器完全兼容,它有页面存储器管理的单元 MMU,寻址能力可达 16MB,有 256B 的片内快速指令数据缓冲器 CACHE;有一个高性能的 16 位 Z-BUS 的总线接口或 8 位的与 Z80 CPU 兼容的总线接口;三个 16 位的计数器/定时器,四个直接存储器存取通道 DMA;一个全双工的通用型异步接收器/发送器;动态 RAM 的刷新控制器;片内振荡器控制器或外部时钟输入选择器;振荡时钟频率可高达 25MHz;亦有等待状态发生器模块,适用于各种智能外设、打印控制、智能终端等。

该系列的最后一块芯片是 Z80181,称为灵活随机控制器 SAC。该片内部有一个 Z80180 的大模块;另有两个 8 位通用型的端口,每位均可以软件来进行设定。一个串行的通信控制器;一个时钟发生器控制器,四个定时器/计数器;外接振荡器频率可高达 25MHz,用 CMOS 工艺制造,与通用型 Z80 的软件是 100% 的兼容,该芯片可以灵活地应用于各种智能型的外设、打印机控制器、传真机、调制/解调器、各种智能终端等等场合。

不难看出,上述各种产品使 Z80 进入一个崭新的时代,形成了一套超高集成度的新一代 Z80 系列,已在国内外竞争激烈的商品经济市场中大量地应用。近来也被我国很多有关工程技术人员和大专院校所关注,相信不久的将来 Z80 超高集成度微处理器一定会应用到原来 Z80 应用的各种个方面,并不断开拓新的应用领域。

(编者按,有关咨询等事宜,请与深圳市振华路电子大厦,中国电子器材公司特区公司联系,邮编:518031,电话:384214,电挂:8410)。



学装微电脑

打印数字温度计

易齐干

本文介绍用热敏电阻检测温度、 $\mu P80$ 进行数据处理、简易打印机打印温度三部份组成的打印数字温度计。

1. 打印数字温度计组成

打印数字温度计系统框图如图 1 所示。热敏电阻检测温度，微电脑 1 将温度变换为数据，以简易 Cen-

tronics 方式把数据送给微电脑 2，微电脑 2 把数据变换为驱动打印机的信息，打印机打印出数字。微电脑 2 与打印机一般为一体，统称为打印机。

2. 微电脑将室温数值化

$\mu P80$ 微电脑将室温数值化可称简单的数字温度计。它的构成与工作原理请参阅“初级微电脑图解教程”。原理图如图 2 所示。

3. 微电脑之间传送数据

微电脑之间传送信息常使用 Centronics 方式。图 3 所表示的微电脑 1 和微电脑 2 的连接是简易 Centronics 方式。

图中， $D_7 \sim D_0$ 是 8 位数据线。STROBE 是微电脑 1 向微电脑 2 传送数据的选通信号线。BUSY 是微电脑 2 通知微电脑 1 可以接收数据的信号线。微电脑 2 驱动打印机需要时间，打印机打印同时，不接收微电脑 1 传送的数据，如果微电脑 1 不分青红皂白地传送数据，微电脑 2 就发生混乱。BUSY 信号线的作用是避免出现混乱。

例如，微电脑 1 向微电脑 2 传送日文片假名“テ”和“・”的数据时序如图 4 所示。

(1) 首先，微电脑 1 检查微电脑 2 的 BUSY 的电平。如果为低电平，微电脑 2 才可以接收数据。

(2) 微电脑 1 用 $D_0 \sim D_7$ 数据线要向微电脑 2 传送数据。

(3) STROBE 为低

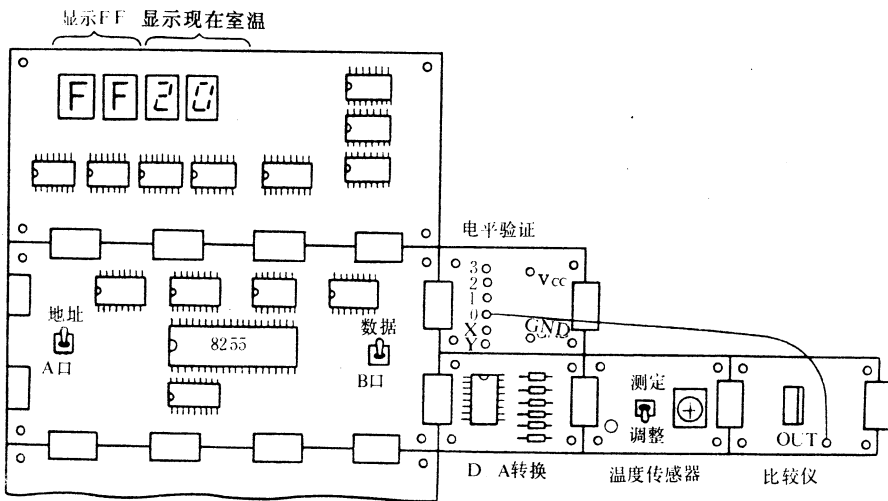


图 1

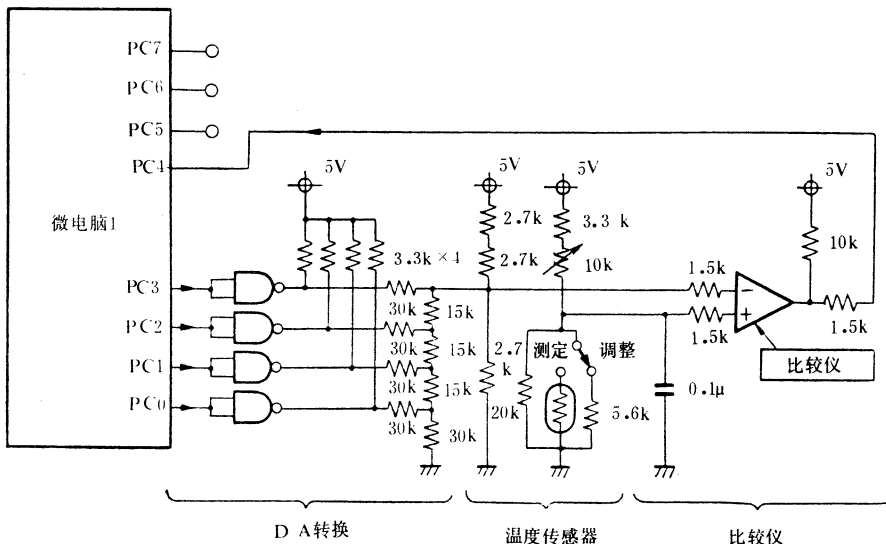


图 2

电平,微电脑 1 输出数据。

(4)STROBE 为低电平的同时,微电脑 2 向微电脑 1 发出 BUSY 为高电平信息,表示接收到微电脑 1 输出的数据,并正在内部进行处理。此时微电脑 1 不能向微电脑 2 输送下一个数据。如果微电脑 2 处理前个数据

完毕,则将 BUSY 变为低电平,通知微电脑 1 重复上述过程。

4. JIS 码

微电脑所认识的打印机能打印出的文字由 JIS 标准来决定。称为 JIS 码。如表 1 所示。

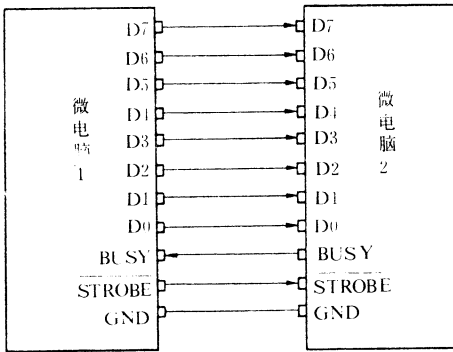


图 3

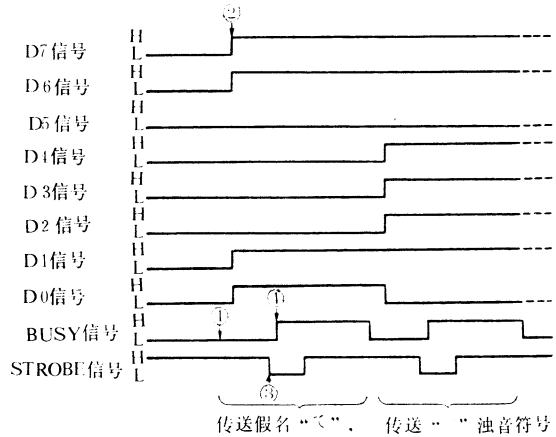


图 4

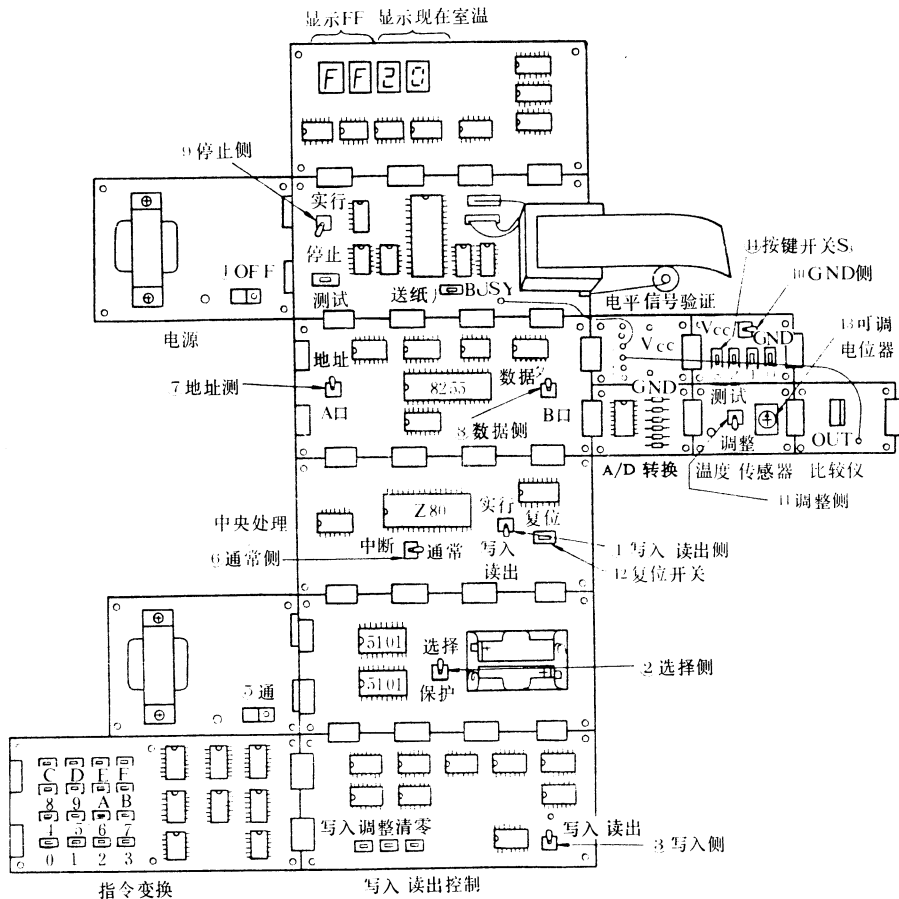


图 5

表 3

端口名	输入/输出	出脚名	任务
端口A	输出	出脚0	向打印机部件输出 STROBE 信号
端口B	输出	出脚0~7	向打印机部件输出 D ₀ ~D ₇ 信号
端口C	输出	出脚0~3	向D/A转换部件输出D/A转换数据信号
	输入	出脚4	输入比较部件的输出信号
	输入	出脚5	输入打印机部件的BUSY信号
	输入	出脚7	输入按键开关S ₁ 的输入信号

表 2

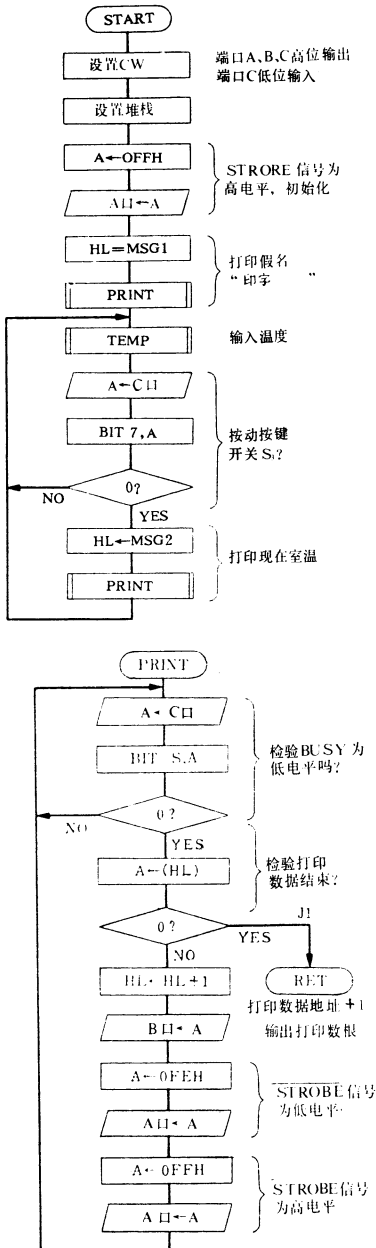
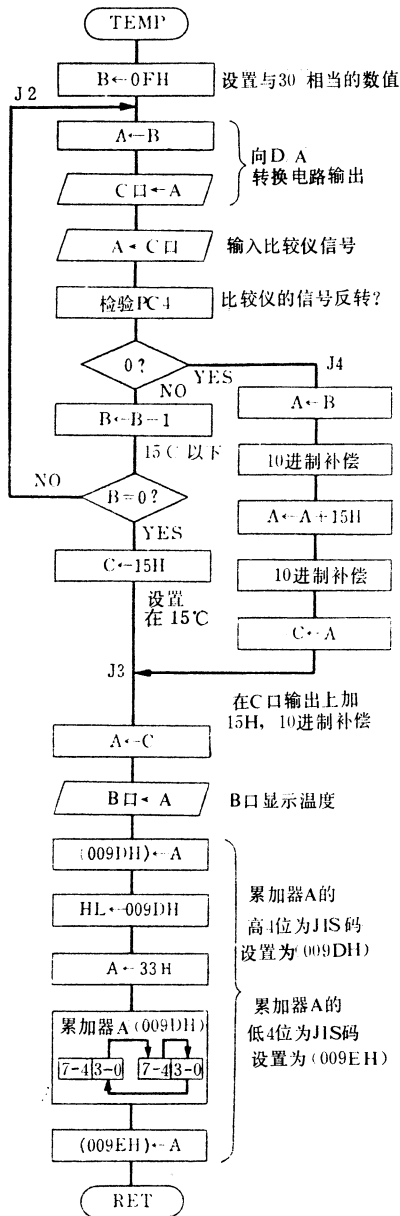


表 1

	0	1	2	3	4	5	6	7
0	0000	0001	0010	0011	0100	0101	0110	0111
1	0001	0010	0011	0100	0101	0110	0111	1000
2	0010	0011	0100	0101	0110	0111	1000	1001
3	0011	0100	0101	0110	0111	1000	1001	1010
4	0100	0101	0110	0111	1000	1001	1010	1011
5	0101	0110	0111	1000	1001	1010	1011	1100
6	0110	0111	1000	1001	1010	1011	1100	1101
7	0111	1000	1001	1010	1011	1100	1101	1110
8	1000	1001	1010	1011	1100	1101	1110	1111
9	1001	1010	1011	1100	1101	1110	1111	0000
A	1010	1011	1100	1101	1110	1111	0000	0001
B	1011	1100	1101	1110	1111	0000	0001	0010
C	1100	1101	1110	1111	0000	0001	0010	0011
D	1101	1110	1111	0000	0001	0010	0011	0100
E	1110	1111	0000	0001	0010	0011	0100	0101
F	1111	0000	0001	0010	0011	0100	0101	0110

标号	助记符	地址	机器语	注释
START	LD A,88H	00	3E 88	A、B、C口高位输出 C口低位输入
	OUT (03H),A	02	D3 03	
	LD SP,100H	04	31 00 01	设置SP堆栈
	LD A,0FFH	07	3E FF	A口初始化
	OUT (00H),A	09	D3 00	
	LD HL,MSG1	0B	21 80 00	设置标号MSG1的地址
	CALL PRINT	0E	CD 30 00	打印标题
MAIN	CALL TEMP	11	CD 50 00	输入温度
	IN A,(02H)	14	DB 02	C口输入
	BIT 7,A	16	CB 7F	若按键开关“通”
	JP NZ,MAIN	18	C2 11 00	
	LD HL,MSG2	1B	21 90 00	现在室温
	CALL PRINT	1E	CD 30 00	打印
	JF MAIN	21	C7 11 00	跳入标号MAIN
PRINT	IN A,(02H)	30	DB 02	检验BUSY是否
	BIT 5,A	32	CB 6F	为低电平
	JP NZ,PRINT	34	C2 30 00	
	LD A,(HL)	37	7E	检验打印数据
	AND A	38	A7	结束?
	JP Z,J1	39	CA 4A 00	
	INC HL	3C	23	打印数据地址 - 1
	OUT (01H),A	3D	D3 01	输出打印数据
	LD A,0FFH	3F	3E FE	输出STROBE 信号
	OUT (00H),A	41	D3 00	
	LD A,0FFH	43	3E FF	
	OUT (00H),A	45	D3 00	
	JP PRINT	47	C3 30 00	跳入标号PRINT
J1	RET	4A	C9	返回主程序

表 2



标号	助记符	地址	机器语	注释
TEMP	L.D B,0FH	50	06 0F	设置D A转换的初始值
J2	L.D A,B	52	78	输出 D A 转换
	OUT (02H),A	53	D3 02	
	IN A,(02H)	55	D3 02	比较仪反转输出?
	BIT 4,A	57	C3 67	
	JP Z,J4	59	CA 70 00	若反转跳入标号J4
	DEC B	5C	05	数据 -1
	JP NZ,J2	5D	C2 52 00	若 B ≠ 0, 跳入标号J2
	L.D C,15H	60	0F 15	温度定为 15°C
J3	L.D A,C	62	79	B1 输出温度
	OUT (01H),A	63	D3 01	
	L.D (ONDO),A	65	32 9D 00	
	L.D HL,(ONDO)	68	21 9D 00	
	L.D A,33H	6B	3E 33	
	RHD	6D	ED 67	温度数据变换为 JIS 码
	L.D (ONDOH+1),A	6F	32 9E 00	
	RET	72	C9	返回主程序
J4	L.D A,B	73	78	温度数据变为 10 进制数
	ADD A,00H	74	C6 00	
	DAA	76	27	
	ADD A,15H	77	C6 15	
	DAA	79	27	
	L.D C,A	7A	4F	加 1.15°C 变为 10 进制数
	JP J4	7B	C3 62 00	跳入标记 J3
MSG 1	DB C3H	80	C3	テ
	DB DEH	81	DE	ト
	DB BCH	82	BC	シ
	DB DEH	83	DE	フ
	DB C9H	84	C9	ヲ
	DB E7H	85	E7	ル
	DB B7H	86	B5	ナ
	DB D1H	87	DD	ン
	DB C4H	88	C4	ト
	DB DEH	89	DE	フ
	DB B9H	8A	B9	ケ
	DB B2H	8B	B2	イ
DB 01H	8C	0D	CR	
DB 00H	8D	00	エンドマーク	
MSG 2	DB C6H	90	C0	ヲ
	DB C0H	91	C0	ヲ
	DB DEH	92	DE	フ
	DB B2H	93	B2	イ
	DB C1H	94	C1	マ
	DB C9H	95	C9	ヲ
	DB 20H	96	20	フランク
	DB B5H	97	B5	ナ
	DB DDH	98	DD	ン
	DB C4H	99	C4	ト
	DB DEH	9A	DE	フ
	DB CAH	9B	CA	ハ
DB 20H	9C	20	フランク	
ONDO	DB 20H	9D	20	フランク
	DB 20H	9E	20	フランク
	DB DEH	9F	DE	フ
	DB 43H	A0	43	C
	DB 00H	A1	00	エンドマーク

例如,微电脑 1 向微电脑 2 输出的信息表示 26°C 时,按表 1 将它数码化,如下所示:

2=32H
6=36H

(下转第 38 页)



单板机 RAM 分段与程序保护的实现

福州市福建师范大学物理系微机室(350007) 王新明

在单板机上开发应用程序时,由于程序局部存在错误或者操作失误,甚至环境干扰,都极易使程序运行混乱,造成内存(RAM)程序和数据遭破坏。特别对较长的程序调试是一种潜在威胁。开发简单实用、旨在保护随机存储器(RAM)中程序的硬软件,对解决单板机程序保护、加快程序开发应用是很有意义的。

一、TP801 RAM 分段保护原理

TP801 单板机上配置了 2K 的 RAM,空间分布为 2000H—2FFFH。除高端几个单元被系统使用外,都是用户区。我们可以把 RAM 分成两段。一段为保护区;另一段为非保护区。

两段以 $2 \times \times \times H$ 为分段地址。如图 1(保护区与非保护区位置可以交换)。①把待调试的程序和只供程序读的数据放在保护区内,其他数据区、堆栈区等设置在非保护区内。在程序运行时,把保护区 RAM 转变成 ROM 性质的存储器,只提供 CPU 读取,一旦发现 CPU 向保护区作写操作时,强迫程序中斷来达到保护目的。②只把程序放在相应空间的保护区内,追踪 CPU 的取指地址,一旦发现 CPU 向非程序区取指就强迫中斷程序运行,来达到保护目的。

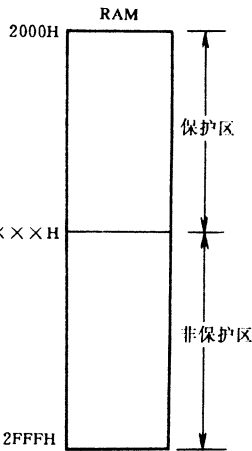


图 1. RAM 分段

设保护区在 RAM 低段,选分界地址为 2520H,则

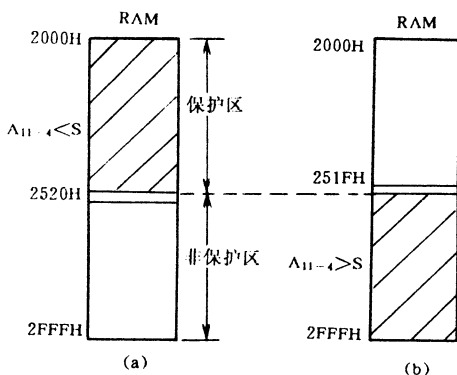


图 2 RAM 的两种分段
(a)S=52H; (b)S=51H

保护区地址 2000H—251FH;非保护区 2520H—2FFFH。由于 RAM 地址高四位总是 0010,当分界地址选在最低四位为 0000 处,同时分段判断用小于、则只要取 2520H 中间的 52H 即地址 $A_{11}-A_4$ 就能在 2000H—2FFFH 把 RAM 分成两段如图 2(a)。其中 $A_{11}-A_4$ 表示地址的 $A_{11}-A_4$ 位,分界地址(2520H)的 $A_{11}-A_4$ 用 S 表示不妨称为段地址。当分段判断用大于,则分界地址应选在低四位为 1111(0FH)处如图 2(b)。

二、硬件设计

电路如图 3。U₁、U₄ 为 74LS273 八位锁存器。1 脚是清零端,低电平有效;11 脚为锁存允许端,低电平有效。U₁ 用于锁存段地址 S,由 \overline{PS}_5 (端口地址 94H)和 \overline{IOW} 选通 U₁ 的 11 脚,在 TP801 单板机上由 94H+PORT(键)将 S 送入 U₁。S 作为数据,输出给 U₂ 比较器的 B 输入端,即把 S 作为比较器的一个数值输入。U₄ 用于在发生 CPU 向保护区作写操作时,锁存被改写单元地址的 $A_{11}-A_4$ (电路在一次 CPU 向保护区写后,转入非屏蔽中斷服务)以提供调试人员在中斷错误程序后,快速找到被破坏的那个单元。读出 U₄ 的内容,由 \overline{PS}_5 和 \overline{IOW} 选通 U₃ 八位缓冲器,将 U₄ 内容送数据总线 DB。U₁ 和 U₄ 使用共同的端口地址,所以由 94H 口读出(PORT)并在显示器上以两位十六进制数显示的是 U₄ 中的保护区被改写单元地址的 $A_{11}-A_4$ (显示成 $\times \times$)。这样我们只要在保护区 $2 \times \times 0H-2 \times \times FH$ 16 个单元查找,就能很快找到被改写的单元。U₂ 是用二片 7485 四数值比较器按图 4 级联成 8 位数值比较器。A 输入

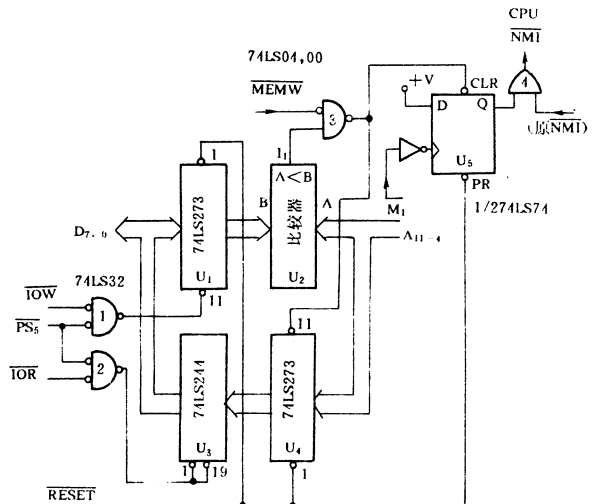


图 3 保护电路

端接地址线的 A_{11-4} 因此, 电路工作时, 比较器不断

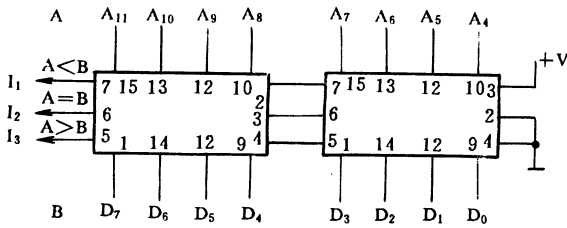


图 4 8 位数值比较器

比较段地址 S (B 端) 和当前地址的 A_{11-4} 。 I_1, I_2 和 I_3 是比较器 $A < B, A = B$ 和 $A > B$ 的结果输出端, 它们的关系见表 1。

电路用小于判断 (I_1) 分辨保护区, 所以分界地址选择第一种方法 (即选在最低四位为全 0000 处。如 2520H)。信号 (原 \overline{NMI}) 是单板机原系统非屏蔽中断申

表 1

A, B	I_1	I_2	I_3
$A < B$	1	0	0
$A = B$	0	1	0
$A > B$	0	0	1

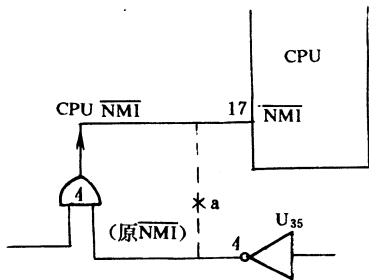


图 5

请线。如图 5 虚线。原电路在 a 处切断, U_{35} 脚接保护电路门 4 的输入端为 (原 \overline{NMI}) 信号, 门 4 输出接到 CPU 的 \overline{NMI} 输入端 (17 脚) 如实线所示。

U_5 为中断请求触发器。由信号 I_1 和 \overline{MEMW} 清零 ($Q=0$) 产生中断请求信号 ($\overline{NMI}=0$)。由 $\overline{M_1}$ 信号的下降沿置 1 来清除中断请求。 $I_1, \overline{MEMW}, \overline{CLR}$ 及 \overline{NMI} 关系见表 2。

表 2:

\overline{MEMW}	I_1	\overline{CLR}	\overline{NMI}	说明
0	0	1	1	非保护区写
0	1	0	0	保护区写, 中断请求
1	0	1	1	非保护区读或 I/O 操作
1	1	1	1	保护区读

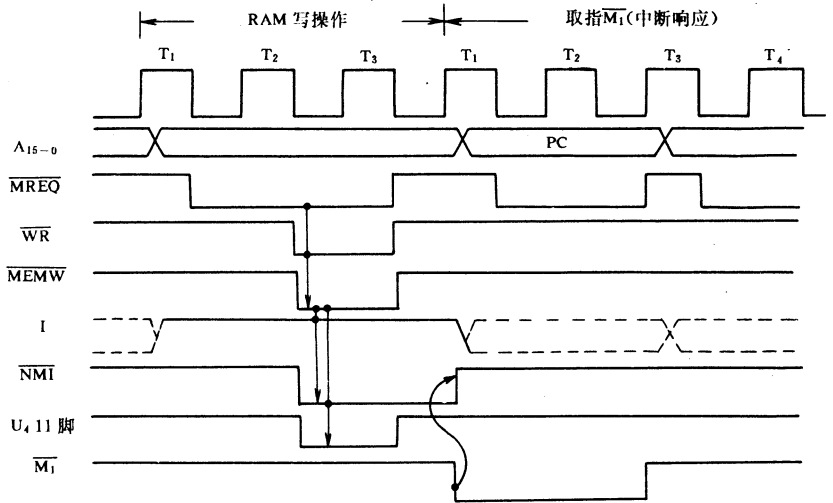


图 6

当 $\overline{MEMW}=0$ 且 $I_1=1 \rightarrow \overline{CLR}=0 \rightarrow Q=0 \rightarrow \overline{NMI}=0$, 导致电路向 CPU 发出中断请求, 这就是当 CPU 作 RAM 写 ($\overline{MEMW}=0$) 且 RAM 地址在保护区内 ($I_1=1$) 时, 电路清零中断请求触发器 U_5 , 发出中断请求。 Q 的低电平维持到取指周期到来, 由 $\overline{M_1}$ 下降沿自动清除中断请求状态。 U_4 11 脚接 \overline{CLR} , 所以, 当 CPU 向保护区一个存储单元作写操作的同时, 该单元地址的 A_{11-4} 被锁入 U_4 。CPU 作保护区写操作及中断响应周期中各有关信号时序见图 6。

\overline{RESET} 可利用原机的复位信号, 电路自带复位电路使用更方便。

电路复位 ($\overline{RESET}=0$) 使 U_1, U_4 复位, U_5 置位。电路处于初始状态: U_1 内 $S=00$, 等效分界地址为 2000H, 保护区长度为 0, RAM 全是非保护区。在这种状态下, 可以

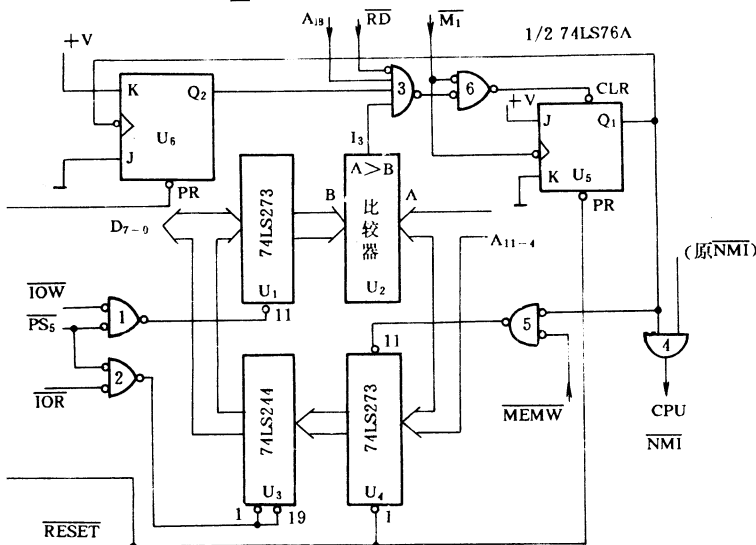


图 7

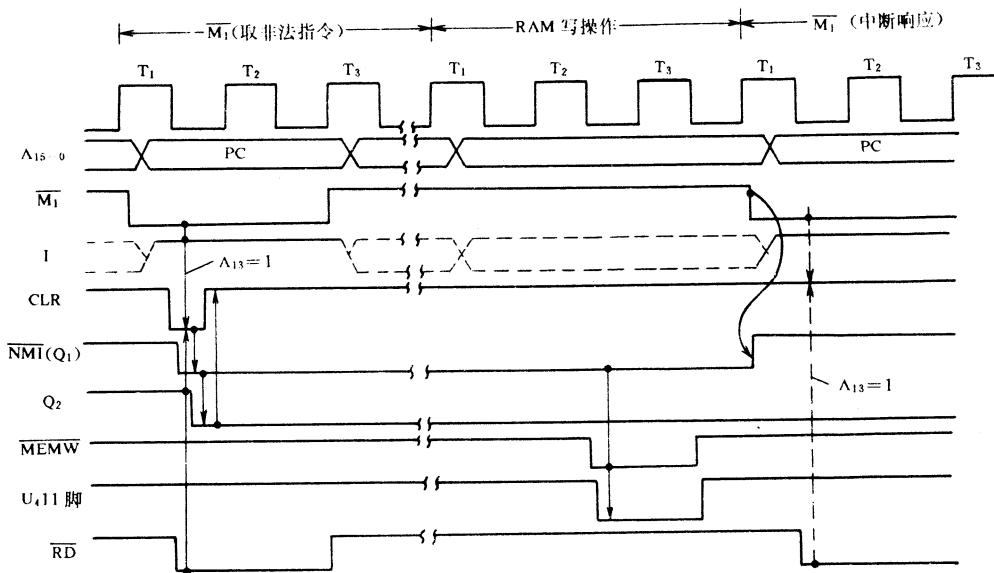


图 8

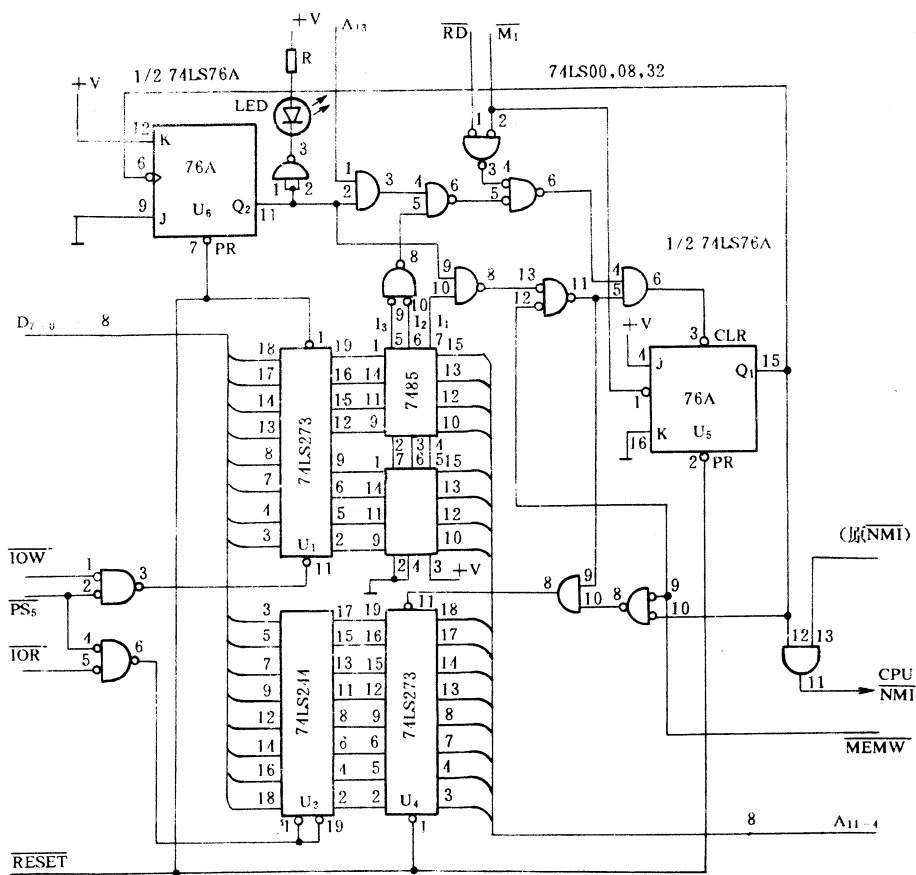


图 9

进行程序、数据输入及 RAM 内容修改等原机操作功能。电路在将段地址 S 置入 U₁ 后的程序运行中对所设置的保护区起保护作用。但对非保护区无任何作用。

3. 取指检测保护

这种保护取保护区长度稍大于或等于程序长度，区内多余的空单元，填上 76H(暂停指令码)，以防不必要错误产生。电路如图 7。分界判断用大于(I₃)，这样分界地址选择第二种方法，即选在低四位为 1111 处(如 251FH, 这时段地址 S=51H)。

U₁~U₅ 的作用同前。U₄ 为中断屏蔽触发器，由 $\overline{\text{RESET}}$ 置位呈允许中断状态(Q₂=1)。U₅ 由 U₅ 的 Q₁ 下降沿触发使 Q₂=0, 这时，由 U₆ 的作用，禁止电路产生新的中断请求信号。屏蔽状态维持到电路重新初始化($\overline{\text{RESET}}=0$)。

U₅ 的 CLR 端状态可由下式表示：

$$\text{CLR} = \overline{M_1} + \overline{I} + \overline{Q_2} + \overline{A_{13}} + \overline{RD}$$

电路初始化后，Q₂=1。因此，必须在其余四个信号同时为：A₁₃=1(地址线上的地址应为 RAM 地址 2××××H)；I₃=1(地址的 A₁₁₋₄>S)； $\overline{M_1}=0$ ， $\overline{RD}=0$ (在取指令周期内)时，CLR=0→Q₁=0→ $\overline{NM\overline{I}}=0$ ，产生中断请求。同时 Q₁ 下跳使 Q₂=0，从而禁止了电路可能的新的中断申请(图 8 中断响应时序中虚线所示)。中断请求信号维持到 $\overline{M_1}$ 到来时，由 $\overline{M_1}$ 下降沿使 U₅ 的 Q₁=1，自动清除申请中断状态。门 5 的作用是在中断请求期间(Q₁=0)，执行非法指令若作 RAM 写(MEM \overline{W} =0)，则由门 5 选通 U₄，将被写的单元地址的 A₁₁₋₄ 锁存下来，U₆ 的禁止中断状态也保证了 U₄ 锁存地址的唯一性。CPU 取非法指令及 RAM 写、中断响应的各信号时序如图 8。

电路在 CPU 向保护区以外的非程序区取指时起保护作用，执行非法指令至多破坏一、二个 RAM 单元；A₁₃ 的加入也防止 CPU 取监控区(0000H—07FFH)指令时引起误动作。电路允许向保护区作写操作，这对一些变量在保护区内的程序是有利的。但电路对取指错误发生在保护区内(如转移指令转移到指令操作数上及之后发生在保护区内的错误运行)没有保护作用。

4. 复合保护

电路如图 9，是上述两电路的合成。它在两种情况之一出现时起保护作用：①在 CPU 取指周期内，指令地址在保护区外；②在执行周期内，CPU 向保护区作写操作时。不仅对保护区有保护作用，而且在多数出错情况下对非保护区也有保护作用。这样，在出错而发生中断后，不仅保存了断点各寄存器状态，还保留了 RAM 数据区、堆栈等的当前值，保证 RAM 程序、变量的安全，有利在发生中断后，供调试人员参考。中断保护发生后，单板机转入非屏蔽中断处理，且显示程序首地址(监控 TPBUG-A 版本)。

电路中，I₁、I₂ 和 I₃ 的组合构成保护区以小于判断，所以分界地址以第一种方式。Q₂ 处的发光二极管用来指示当前电路工作状态(保护或非保护态)。另外，U₆ 的 PR 可由 $\overline{\text{RESET}}$ 和另加的开关来共同置位 U₆，这样，可在不影响 U₁ 的 S 情况下用开关单独使 U₆ 从屏蔽态中恢复。

电路可装在一片插板上，利用原机 S-100 总线插座，接插方便，原机硬件几乎不变。方便实用，特别适合于实验开发及学生实验用机。



(上接第 34 页)

$$\ominus = \text{DFH}$$

$$C = 43\text{H}$$

这样将数字或文字变换为 8 位数码。按图 4 所示时序形式，微电脑之间传递信息。

5. 微电脑与打印机联接

微电脑 2 与打印机装在一块印刷线路板上。通过 8 线接插件与图 2 所示的数字温度计相连。如图 5 所示。

6. 软件

热敏电阻检测室温经过数据转换，由打印机打印出来。程序清单如表 2 所示。

执行该程序首先打印出“数字温度计”的日文片假

名“デジタルオンドケイ”(程序标记 MSG1 处)。按键开关 S₃ 为“通”，打印出“现在室温××°C”的日文片假名“タダイマノオンドハ °C”(程序标记 MSG2 处)。测试现在室温用 TEMP 子程序。现在室温数字放在程序标记 ONDO 处，即 9DH 和 9EH 地址中，执行 MSG2 打印指令，可自动调入室温数据。

输入输出口的设置如表 3 所示。

7. 软件改进

上述程序与数字时钟程序相结合，可以实现隔一定时间自动记录温度。

也可以考虑应用于实验室的计测仪器中，或者是自制的测试设备中。有兴趣的读者可进行尝试。



第三章 F BASIC 的画面控制语句

山东苍山县机械电子化学工业局(277700) 于 春

F BASIC 语言有着独特的背景画面设计和卡通画面控制功能。它独特的画面控制语句,是任何 BASIC 语言难以比拟的。使用该部分语句,可以巧妙地设计背景画面(BG GRAPHIC)、灵活地控制卡通图案(SPRITE)的动作,从而设计出妙趣横生的游戏程序。

BG GRAPHIC 是英文 BACK GROUND GRAPHIC 的缩写,意思为利用背景绘图。SPRITE 在英文中是幽灵,小精灵的意思,在此表示卡通图案。为叙述方便,以后称背景图形面为 BG 面,卡通图案面为 SP 面。

由于 F BASIC 语言画面控制语句的独有性,所以,对新老程序人员来说都较陌生,尤其对初学者,接受更感困难。因此,本文以较多的示例,循序渐进,分层介绍该部分语句的功能和使用。并对随机手册中的错误予以更正,遗漏予以补充,并增添了几条新语句。读者只要按介绍的顺序,逐句领会,勤加练习,定能熟练掌握。

一、画面控制语句

1. 设定光标语句(LOCATE)

LOCATE 简写 LOC.

LOCATE 语句的功能是设定光标的显示位置,其格式为

LOC. X,Y

式中:X—水平方向座标。取值 0~27

Y—垂直方向座标。取值 0~23

LOCATE 语句设定光标显示位置的座标为(X,Y)。

如:10 LOC. 0,10

RUN 光标在座标为(0,10)处闪烁。

注意——使用 LOCATE 语句设定闪烁光标时,X 取值恒为 0,Y 取值可以为 0~23。若 X 取其它值,计算机执行时,仍将它当做 0 处理。这是因为闪烁光标为输入语句的起点,而每行的输入起点必然在 X=0 的位置。所以 LOC. 0,15 与 LOC. 10,15 效果是一样的。

例 1.

```
10 F. I=0 TO 20
20 LOC. I,I : P. "# "
30 N.
```

例 2.

```
10 F. A=0 TO 5
20 X=0 : Y=0
30 X=X+1 : Y=Y+1+A
40 IF X>27 OR Y>23 T. 60
50 LOC. X,Y : P. "*" : G. 30
60 N.
```

2. 清屏语句(CLS)

CLS 语句的功能是清除屏幕的文字、画面等、使屏幕变为空白。如例一、例二中最后加一句

PAU. 100 : CLS

RUN 后,打印的图象显示一段时间,然后清屏,仅在屏幕左上角显示 OK 和光标。

3. 测试光标语句(POS 和 CSRLIN)

在程序的运行中,有时要了解程序的执行情况和光标在目前画面上所在位置的座标,使用测试光标语句可以达到这一目的。

POS(0)给出光标在目前画面上的水平座标。

CSRLIN 给出光标在目前画面上的垂直座标。

CSRLIN 简写为 CSR.

例 3.

```
10 F. X=0 TO 25
20 LOC. X,0
30 P. POS(0) : PAU. 10
40 N.
```

RUN 后,打印光标的水平座标。

例 4.

```
10 F. I=0 TO 20
20 LOC. 0,I
30 P. CSR. : PAU. 20
40 N.
```

RUN 后,打印光标的垂直座标。

例 5.

```
10 X=Y=0
20 X=X+1 : Y=Y+5
30 IF X=20 T. E.
40 IF Y>23 T. Y=0
50 LOC. X,Y
60 P. POS(0);",";CSR. : PAU. 10 : CLS
70 G. 20
```

RUN 闪烁打印出光标的各组座标。

4. 字符—ASCII 码转换语句(ASC, CHR \$)

ASC 语句、CHR \$ 语句是一对相互逆操作语句。ASC 语句实现把字符转换为 ASCII 码。CHR \$ 语句(简写 CH.)实现把 ASCII 码转换为相应的字符。语句格式为:

ASC(字符串)

CH. (X) X 取值范围 0~255

例 6.

```
10 I. A $
20 P. A $ "="ASC(A $)
```

30 G. 10

RUN 后每输入一个字符,则显示该字符对应的 ASCII 码。

例 7.

10 I. A

20 P. A="CH. (A)

30 G. 10

RUN 后每输入一个数字(0~25)则显示该数字对应的字符或图案。

例 8.

10 F. I=0 TO 255

20 P. I="CH. (I)

30 N.

RUN 打印 0~255 所对应的字符和背景图案。

注一在随机手册中仅刊印了背景图形库的 104 种图案,但没有给出各图案所对应的 ASCII 码,这给调用背景图形带来了困难。下面给出所对应的 ASCII 码。读者可把数字标在背景图形库上以便使用。在背景图形库中

- 第一行的一~八列对应的 ASCII 码为 0~7
- 第二行的一~八列对应的 ASCII 码为 8~15
- 第三行的一~八列对应的 ASCII 码为 16~23
- 第四行的一~八列对应的 ASCII 码为 24~32
- 第五行的一~八列对应的 ASCII 码为 184~191
- 第六行的一~八列对应的 ASCII 码为 192~199
- 第七行的一~八列对应的 ASCII 码为 200~207
- 第八行的一~八列对应的 ASCII 码为 208~215
- 第九行的一~八列对应的 ASCII 码为 216~223
- 第十行的一~八列对应的 ASCII 码为 224~231
- 第十一行的一~八列对应的 ASCII 码为 232~239
- 第十二行的一~八列对应的 ASCII 码为 240~247
- 第十三行的一~八列对应的 ASCII 码为 248~255

说明:在第五行至十三行的 72 个图形可用 CHR \$ 语句直接调用,而第一行至第四行的 32 个图形则必须用以后将要介绍的 CGEN 及 SPRITE 语句才能调出来。

5. BG、SP 面符号配置语句(CGEN)

CGEN 语句的功能是决定背景(BG)面和卡通(SP)图案面上的符号配置。规定哪些内容显示于 BG 面,哪些内容显示于 SP 面。CGEN 语句的格式为

CGEN(n)

式中 n 的取值为 0~3,它的分配情况如下表

n	背景(BG)面	卡通(SP)面
0	显 BG	显 SP
1	显 BG	无 SP,显符号、字母
2	无 BG	显 SP
3	无 BG	无 SP,显符号、字母

由于 CGEN 语句较难懂,所以补充了上表,读者可配合随机手册中指令分配表,再结合后文例 11 的演

示,领会该语句的变化和作用。

二、卡通显示语句

该部分共有四个语句:DEF SPRITE、SPRITE、SPRITE ON、和 SPRITE OFF,主要用于卡通图案的显示或消除。随机手册中称这四条语句为立体分解画面用语句,我觉得称为卡通显示语句较贴切、也较易理解。

1. 定义 SP 语句(DEF SPRITE)

DEF SPRITE 简写 DE. SP.

该语句的功能是定义显示于 SP 面上的卡通图案。

在 F BASIC 系统中固化了十六类卡通图案 58 种姿态,每个姿态由一组 4 个数字进行定义,如玛丽哥哥跳为(12,13,14,15),丽莎小姐滑倒为(44,45,46,47)等。其语句格式为

DE. SP. n, (A,B,C,D,E)=字符表达式

式中(1)n 是 SP 的编号,0~7。F BASIC 允许在同一画面中显示八个卡通图案,每个卡通编一个号码,以便识别。

<2>A 为配色号码,0~3,以后介绍

<3>B 为显示图案的大小。取值 0~1,当 B=0,图案为一个文字大小(8×8点),取值为1时,显示图案有四个文字大小(16×16点)。一般取值为1。

<4>C 表示显示的方式。即 SP 是显示于背景前,还是背景后。取值 0~1。C=0,显示于前;C=1,显示于后。

<5>D、E 为显示图案反转指示、各取值为 0~1,为 0 时不翻转,为 1 时翻转。D 定义 X 轴方向,E 定义 Y 轴方向。

<6>字符表达式:一般由 CHR \$(X) 一组四个组成,也可由字符列或字符变量组成。

2. 显示 SP 语句和消除 SP 语句(SPRITE ON, SPRITE OFF)

显示 SP 语句 SPRITE ON 简写 SP. O. 功能是打开显示开关,显示 DEF SPRITE 语句定义的卡通图案。

消除 SP 语句 SPRITE OFF 简写 SP. OF. 功能是关闭显示开关,消除已显示的卡通图案。

在进行程序设计时,必须先使用显示语句 SPRITE ON,否则卡通图案不能显示。消除语句一般不用。

3. SP 显示位置语句(SPRITE)

SPRITE 语句的功能是定义卡通图案显示或消失的位置座标。

SPRITE 简写 SP.

语句格式为

SP. n,X,Y

式中 n 是 SPRITE 编号 0~7,F BASIC 允许在同一水平方向上同时显示 4 个卡通图案,若超过 4 个,以后的 SP 不能显示;在同一画面上允许显示 8 个卡通图案,若超过 8 个,以后的 SP 也不能显示。X 是 SP 显示的水平座标,Y 是垂直座标,取值均为 0~255。实际上由于显示器画面的有效范围限制。一般 X 取值 0~240,Y 取值 0~220。

另外当 SPRITE 语句后省略 X,Y 时,其功能为消除 DEF SPRITE 的作用。

例9. 在(100,100)处显示玛丽哥哥跳,在(150,150)处显示丽莎小姐跳。

```
10 SP. O.
20 DE. SP. 0,(0,1,0,0,0)=CH. (12)+CH. (13)
   +CH. (14)+CH. (15)
30 DE. SP. 1,(0,1,0,0,0)=CH. (40)+CH. (41)
   +CH. (42)+CH. (43)
40 SP. 0,100,100
50 SP. 1,150,150
```

例10试编一程序,显示玛丽的七种动作。

```
10 SP. O. :Y=-1
20 F. X=0 TO 27 ST. 4
30 Y=Y+1
40 DE. SP. Y,(0,1,0,0,0)=CH. (X)+CH. (X+1)
   +CH. (X+2)+CH. (X+3)
50 SP. Y,Y*25+50,Y*25+50
70 N.
```

RUN 显示玛丽的七种姿式。若欲显示其它卡通只须改50行中 CHR \$ 括号中的数字即可。也可以改30行的循环变量初、终值。

例11. 编一程序显示 CGEN 的四种状态

```
10 CLS:SP. O.
20 F. I=191 TO 255
30 P. CH. (1);:N.
40 DE. SP. 0,(0,1,0,0,0)=CH. (0)+CH. (1)+
   CH. (2)+CH. (3)
50 SP. 0,100,150
60 F. A=0 TO 3:PAU. 300
70 CGEN A.
80 LOC. 0,23:P. "CGEN";A:N.
```

可反复 RUN,仔细观察 CGEN 语句的变化和功能。

通过上例的演示,细心的读者可以发现,在 CGEN 等于1和3时,玛丽哥哥变为背景图形,它们正对应于背景图形库中第一行1~4列的四个图案。因此,可以编一程序调出图形库中前32个图形。

例12. 调出背景图形库前32个图形。每八个一组,依次显示。

```
10 CLS:SP. O. :CGEN 3
20 F. A=0 TO 7
30 DE. SP. A,(0,1,0,0,0)=CH. (A)
40 SP. A,50,A*30:N.
50 PAU. 100:SP. OF. :SP. O.
60 F. I=8 TO 15
70 DE. SP. I-8,(1,1,0,0,0)=CH. (I)
80 SP. I-8,100,(I-8)*30:N.
90 PAU. 100:CLS
100 F. I=16 TO 23
110 DE. SP. I-16,(2,1,0,0,0)=CH. (I)
```

```
120 SP. I-16,150,(I-16)*30:N.
130 PAU. 100:CLS
140 F. I=24 TO 31
150 DE. SP. I-24,(3,1,0,0,0)=CH. (I)
160 SP. I-24,200,(I-24)*30:N.
```

RUN 将依次显示32个背景图形,并每八个图形变换一种颜色。程序50行使用了 SPRITE OFF 语句,消除前一组图形,然后用 SPRITE ON 语句再打开显示开关。也可以使用 CLS 语句达到同一目的。如90行和130行。读者可以用 GOTO 语句反复演示。

上例程序重复部分较多,我们可以再加一重循环,以简化程序。双重循环程序如下:

```
10 CLS: SP. O. :CGEN 3:X=-1
20 F. A=0 TO 31 ST. 8
30 X=X+1
40 F. I=0 TO 7
50 DE. SP. I,(X,1,0,0,0)=CH. (A+I)
60 SP. I,50+50*X,I*30:N.
70 PAU. 100:CLS:N.
```

RUN 与上例程序效果相同。

以上程序若再加进 CGEN 语句使从0~3变化,形成三重循环,则可观察 CGEN 语句四种状态的变化,读者可自己练习。

三、BG、SP 画面结合语句(BGTOOL、VIEW)

1. 返回 BG 画面语句 BGTOOL
BGTOOL 简写 BG.

BGTOOL 语句可作为直接指令使用。其功能是在直接进入 BASIC 状态进行程序设计中,当需要进行背景画面设计时,返回 BG 画面。使用方法:不键行号,直接键入 BGTOOL,回车后,则进入 BG 画面。也可以将它编在程序语句中,在程序运行中返回 BG 画面。在按 BG GRAPHIC 方法设计背景图案完毕后,键入 ESC 和 STOP 可返回 BASIC 状态,继续程序设计或运行。

该语句随机手册中没有介绍,而介绍的从 BASIC 状态返回 BG 画面语句 SYSTEM 在直接进入 BASIC 状态中也无效。SYSTEM 语句只有在对话进入 BASIC 状态中才有效。由于对话进入 BASIC 可供用户使用的内存比直调 BASIC 要少一倍,一般使用直调 BASIC,所以 BGTOOL 语句比 SYSTEM 更有用。读者可在你的随机手册中添上,以利使用。

2. 调 BG 画面语句 VIEW

VIEW 简写 V.

语句功能是在 BASIC 程序执行中调入已设计好的背景画面。其格式为 VIEW。使用方法是在程序的适当地方添上该语句。当程序运行到 VIEW 语句,则调进 BG 画面。

例13. 画一方框,使丽莎的七种姿态在方框中依次显示。

```
10 BG. :SP. O.
20 Y=-1
```

(下转第3页)



PC-1500计算机的维修

四川雅安地区气象局(625000) 何 静

日本夏普公司生产的 PC-1500袖珍计算机以它功能强,体积小,运算精度高等独特优点,被广泛应用在我国气象水文、地质勘探、测量规化等部门的业务中。伴着它适应性强,普及面广,使用环境不限的长处,也随之带来了比较容易出故障的缺点。根据近几年的使用情况看,其故障大多出在外部设备上。CE-150四色绘图打印机的故障率为百分之四十;主机故障率为百分之二十;录音机的故障率为百分之二十;接口电路和其它故障率为百分之二十。因售后服务不太完善,即使一个细小的零件坏了,也不易配制。这给维修工作带来一些麻烦,也在一定程度上影响了 PC-1500计算机的使用效率。以下是笔者从事 PC-1500计算机维修的笔记整理。希望对读者在维修和使用 PC-1500计算机上有所帮助。

一、故障现象:开机就显示“CHECK:6”,加上充电器充电达5小时以上也是一样。做常规检查时,敲 TEST 后打印机不动作。

故障分析排除:机器显示“CHECK:6”,即是系统警告用户停机检查。其故障是由包括电源在内的多方面原因引起的。按照一般维修规则,首先应从电源电路上检查,因为是主机有显示,打印机不初始化,故针对本故障现象应从打印电源入手检查。打印电源如图 A。整个电路由充电回路,可充电镍镉电池组和电力检测电路组成。充电回路由 R_1 、 D_1 组成; $D_3 \sim D_5$ 对直流稳压电源 EA-150 输出的 9V 电压降压后送给打印机的各部分; D_2 和 D_6 是将电池组电压分别送给主机和打印机;整个电路的后半部分是电力检测电路,其作用是检查电力是否足够,将检测结果送 CPU 告诉用户。为了检查打印机电源的充电回路是否有故障,将电池组的正极引线如图 B 处断开,并接上 EA-150 充电器向打印机电源充电,然后用电压表测量印刷电路板上连接电池的正负极的两个焊点(如图 C、D)上的电压。此时量得电压为 0.03(正常时应为 9V 左右)。用电阻沿充电回路测试发现二极管 D_1 内部开路。

经过更换 D_1 后显示屏上再没出现“CHECK:6”提示。

二、故障现象:主机显示“ERROR 80”,而且屏上字符暗淡、模糊。打印机不初始化。

故障分析及排除:主机显示 80 类错误,即是系统提示电力不足。但经过用电压表检查电池组电压是充足的。说明电源的前半部分是正常的,要从电力检测电路(如图 A 后半部分)去查找故障原因。电力检测电路由稳压管 HZ4CLL,电位器 W_1 ,三极管 2SC2021,电阻 R_2

组成。当在前的电池组输出电压大于 6V 时,稳压

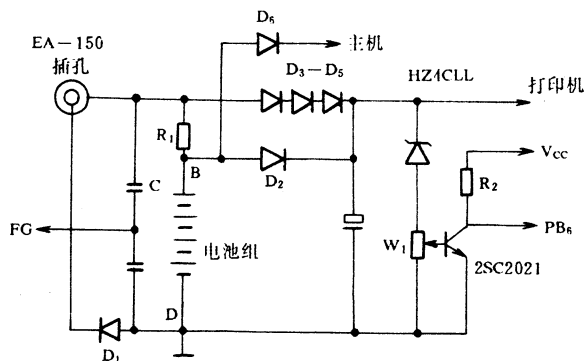


图1
CE-150四色绘图打印机电源电路

二极管反向击穿导通,电位器 W_1 上将有 2.3V 左右的压降,使三极管饱和导通。发射极与集电极间压降为 0,由集电极向 I/O 端口的 PB_6 输送低电平,CPU 给显示屏发出电力充足的信号。反之,当电池电压低于 6V 时,三极管不能饱和导通。处于放大或截止状态,三极管对地电位为 1.4V 左右,输送给 I/O 端口的 PB_6 为高电平。CPU 发出电力不足的显示。本故障是一直提示电力不足,说明 PB_6 端口未得到低电平,而一直处于高电平状态。用电压表测量三极管集电极电压为 1.41V。经逐步检查发现是由于稳压管开路使三极管一直未导通。因为手头无现成的稳压管替换,故只好采取应急措施,将三极管集电极接地,人为地使 PB_6 得到了低电平。这只能是暂时的,这是将 PB_6 的电位强行拉为永久的低电平,是对机器的一种“欺骗”。其后果将是:当电池电力真正不足时,系统也不会提示警告用户。

机器经过以上处理后,基本上能工作。但显示屏上字符还是模糊不清。显示暗淡,一般是主机电源电压不够或虽电源够但某个元件坏使电源电压降低引起。显示屏电源 V_A 、 V_B 、 V_M 、 V_{DISP} 都是由主机电源供给的。主机电源厚模电路 MA1066 的外接电位器 W_2 ,主要控制调整这四个电压。

经过反复调整电位器 W_2 ,屏上字符清晰可见,本机完全恢复正常工作状态。

三、故障现象:加电后按进纸键有时进纸,有时不进纸。初始化常规检查时,敲 TEST 后打印机未印出标准的四色方框。有的画成长方形,有的画成一条粗线。

故障分析及排除:根据有时进纸,有时不进纸的故

障现象,初步分析故障属于 CE—150四色绘图打印机的机械故障,并且故障在打印机 Y 方向(即垂直方向)的动力传动上。拆开打印机的外壳,外观检查 Y 方向动力传动部分,看不出什么可疑的地方。但当从打印机的正面向后顺时针轻轻转动 Y 方向的过轮时,手感觉偶尔有卡住现象。为此,取下 Y 方向的过轮,轻轻转动

Y 齿轮,手感转动轻松;再用手转动 Y 电机齿轮时,手感异常,再进一步借助放大镜观察,结果发现 Y 电机齿轮上有一条与轴向平行的裂缝。

经过更换 Y 电机齿轮后,CE—150四色绘图打印机的故障消除。

导电橡胶按钮被磨损的修理方法

梁绍建

在现代电子产品中,很多方面都应用导电橡胶按钮技术。例如:电子计算机键盘、电子手表、电视遥控器,计算器键盘等。用导电橡胶做按钮有很多优点,它防水防尘性能极佳,可以在恶劣的环境下工作。用导电橡胶做的键盘决无大头针等微小物品掉进去造成事故的麻烦。

导电橡胶按钮一般是在橡胶按钮上和印刷电路板接触的键面上涂复一层导电胶而成。在使用中由于键面和印刷电路板之间的磨擦而产生磨擦,导电键面磨损后导致按钮灵敏度下降,最后完全失效。这种现象一般发生在常用的一个或几个按钮上。如果拆下按钮,用万用表的电阻档测一下那个失效的按钮键面,正常值应为几百欧姆至一千欧姆左右。如果阻值很大,就可以

判定为导电橡胶失效,就需要更换修理。但在一般情况下很难购到配件,我们可以自己用下面的办法修复。拆下键盘,用细砂纸把键面进行粗化处理,轻轻磨去表面的亮光即可,然后用酒精棉擦净。再找一点石墨粉,或用细砂纸磨下一点铅笔芯的粉末代用也可,最好用软些的3B 铅笔。把石墨粉均匀摊在一块蜡纸上,(蜡纸最好用不干胶标签的衬纸)石墨粉的厚度大约在0.5mm左右,用502胶水在处理后的键面上薄薄地涂上一层。立即把键面按在石墨粉上不动,十几秒后把蜡纸移走。由于蜡光纸表面非常光滑,可以在键面上获得一个平滑的导电表面。最后用酒精棉将键面上的浮粉擦去。这个键面的即告修复。用同样的方法修复其它按钮,这种方法简便易行,效果极佳。

IBM—PC 机故障维修一例

武汉江岸车辆工厂(430012) 欧阳波

故障现象:平时硬盘启动正常。突然有一次,一开机,硬盘不断循环自检,不能系统自举。用软盘启动,但不能进入硬盘,如强行进入,则提示:INVALID DRIVE SPECIFICATION(非法驱动器标识符)如用非低级格式化来格式化硬盘则同样提示非法驱动器标识符。

用磁盘医生 NDD 软件对硬盘进行诊断,结果为:系统自举区已破坏,且提示是属硬件上的破坏,低级格

式化是唯一的办法,针对这种故障,把74LS244芯片用酒精擦洗并压紧,再用 NDD 诊断,则没有硬件上的故障,这样避免了破坏硬盘上已有的数据,同时排除了故障。这是由于芯片74LS244的管脚接触不良和表面灰尘影响所致。如有上述这种故障,擦洗和压紧74LS244芯片后,仍未排除故障,则换上一个好的74LS244芯片,故障即排除。这种故障多半是74LS244芯片所致。



DATA BASE IV 关系数据库

鸡西矿务局计算中心(158100) 张冰毅

新书与软件

DATA BASE IV (以下简称 DBIV) 关系数据库管理系统是1984年10月由 IKE 计算机有限公司推出的, 1985年初国内对它进行了汉化。DBIV 可以运行于 IBM-PC、PC/XT、PC/AT、IBM5550 和长城 0520 系列机等及其兼容机上, DBIV 对系统要求: 内存 256KB, 2.0 版以上 PC-DOS 和由 DOS 支持的一个 360KB 软盘或硬盘; 现有汉字 DOS 均支持 DB IV 软件系统。

时过七年, dBASE III 各版本和 FoxBASE 汉化版已广泛应用, DBASE IV 汉化版已推出的今天, 为什么又重提 DB IV 呢?

近两年中华学习机已逐步进入中、小学校和家庭, 与 IBM-PC 兼容的主要机种有 CEC-PC 和 BF PC-BOY, 主机内均配有 16×16 点阵汉字库, 基本配置均为内存 256KB, 一个 360KB 软盘, 提供与电视机的接口, 可将黑白或彩色电视机作为监视器用。在这种基本配置下, 无论 dBASE III, 还是 FoxBASE 和 DBASE IV 均无法运行, 而 DB IV 却可以, 并能够开发一些小型数据库信息管理系统, 这对只具有基本配置的用户不能不说是一件好事。

DBIV 最多可同时处理 10 个文件, 每个文件最多可容许记录数达 65535 条, 每条记录最多可容纳 4000 个字节, 每条记录最多可设 64 个属性; 观察(索引)命令中关键字最大长度为 80 个字符。数据有效位数对标准 BCD 码达 14 位, 对专用版本使用双精度可达 16 位, 而通常算术运算精度也可达到 6 位。

DBIV 可用两种操作方式工作, 一是直接执行 DB IV 命令的直接方式, 二是使用 DBIV 命令集的程序工作方式。DBIV 设有五种类型文件, 命令文件、观察文件、报告文件与文本文件, DBIV 对其记录采取分级处理方式。

DBIV 设置 44 种命令、四类共 35 种函数; 尤其是该系统设置了一种编辑器 (DB Editor), 用它可方便地建立新的文本文件, 编辑一个已有的文本文件, 或是开发应用程序等较强的编辑功能。此外, 它还设有一种可方便地建立输入/输出屏幕的 DB 屏软件。

使用过 PC 机的人基本上都熟悉 PC-BASIC 语言

和 dBASE III 关系数据库; 而 DBIV 有约四分之一的命令与 BASIC 命令名字相同, 功能接近, 有约二分之一的命令与 dBASE III 命令名字相同, 功能接近。例如: 显示磁盘文件目录用 FILES 命令, 打开一个数据文件用 OPEN 命令, 显示与数据文件有关的信息用 DISPLAY 命令, 对数据文件中数字属性求和用 SUM 命令。在 DB IV 中, 与 BASIC 语言相同的是内存变量的数目仅受硬件设备的限制; 字符串变量要在变量名后加上“\$”号, 数字值变量则不加, 在整型数字值变量后加上“%”号; STR\$ 函数是将数字值变为字符表达式, 而 VAL 函数则是将字符表达式变为数字值。DBIV 可以象 BASIC 语言那样用 GOSUB-RETURN 命令调用子程序, 这对提高编程速度和质量是很有用的。若内存够用, DB IV 可用 // 命令执行 DOS 命令及语言命令 (如 BASIC、Multiplan 等), 这对编制较复杂的应用软件提供了一个非常方便的条件。

DB IV 本身提供了把数据文件转换成文本文件的命令, 可以用高级语言对文本文件进行处理; 另外经过对 DB IV 的关系数据文件的结构和记录进行研究, 找出文件结构和记录之间的数学关系式, 给出了一种用高级语言直接处理 DBIV 关系数据文件的方法, 这就提供了 DBIV 与高级语言的直接接口, 采用这种直接接口方式编制应用软件, 既可以互相取长补短, 又不多占用外存空间, 具有较大的实用价值。

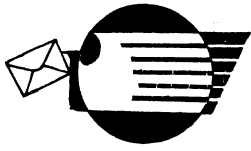
DBIV 主系统文件 DB4.EXE 只有 113KB, 且无覆盖文件, 这样在一张 360KB 软盘上给用户至少提供 200KB 外存空间; 若单独使用一张数据程序盘, 可先把 DBIV 主系统文件 DB4.EXE 调入内存后再抽出系统盘, 而后把数据程序盘插入 A 驱动器, 这样整个 360KB 软盘外存空间全部留给用户了。采用这种方法对建立小型数据库信息管理系统用一张软盘就够用了, 这就为 CEC-PC 等机型用户使用关系数据库提供了方便条件。

总之, DBIV 功能较强, 使用方便, 易学易懂, 对在一定范围内推广使用具有较大的意义。

(上接第 47 页)

普通显示器的扫描频率是固定的, 同一种显示器只能与一种对应的显示卡相匹配。如果要更换显示卡, 则必须同时更换显示器。

多同步显示器的扫描频率是可变的, 既可以配置 EGA 显示卡, 又可以配置 VGA 显示卡, 当需要提高显示分辨率时, 只要更换显示卡, 不需要更换显示器。



读者联谊

普及型 PC 个人用户软件交流联谊活动

问题解答(二)

北京中国农科院计算中心(100081) 王路敬

2. PC 机的系统板包括哪些主要功能部分?

PC 机的主机采用大板结构。系统板水平地固定于机箱底部。系统板的主要部分分成五个主要功能区。它们是微机处理器子系统和只读存储器(ROM)子系统,读写(R/W)存储器子系统,各种 I/O 适配器和 I/O 通道的支持部件。

系统板的核心是 Intel 8088 微处理器。它是一种 16 位微处理器,它的内部结构是 16 位的,而对外的数据总线是 8 位。它与 Intel 8086 在软件上是完全兼容的,其指令系统和汇编语言是相同的。因此,支持包括乘和除的 16 位操作。它的内部有 8 个 16 位的通用寄存器,可以存放操作数。可以实现寄存器间接寻址、基址寻址、变址寻址、以及基址加变址等多种寻址方式,使指令更加灵活,能适应简单变量、下标变量、矩阵等方面的运算。可实现 16 位算术运算和逻辑运算,实现 16 位数的移位和循环,且能指定任意的移位次数,可实现多种 16 位数的串操作。

在 Intel 8088 内部有 9 个标志位,可反映 CPU 操作的状态,实现各种条件转移和循环,重复控制。可实现 16 位数(或 8 位数)的输入输出,采用间接寻址方式,I/O 端口地址可以扩展到 64K 个。

Intel 8088 具有 20 条地址引线,可直接寻址 1M 字节内存空间。8088 可实现 256 个矢量中断,它具有软件中断,非屏蔽中断请求(NMI),屏蔽中断请求(INTR)和追踪等中断方式。利用软件中断可以很方便地调用操作系统中的大量子程序,大大简化了程序的编制。若把 8088 接成最大组态,可以很方便扩充浮点运算协处理器 Intel 8087。加上了 8087 可以使浮点运算的速度大大提高。

Intel 8088 可寻址的内存地址范围 0000—FFFFH。给定一个 20 位的地址,就可以从 1M 字节中取出所需要的指令或操作数。在 8088 系统中,存储器的访问,在不改变段寄存器值的情况下,寻址的最大范围是 64KB。所以,若有一个任务,它的程序长度,堆栈长度,以及数据区长度都不超过 64KB,则可在程序开始时,分别给 DS、SS、ES 置值,然后在程序中就可以不再考虑这些段寄存器,程序就可在各自的区域中正常地进行工作。若某一任务所需的总的存储长度包括程序长度,堆栈长度和数据长度等不超过 64KB,则可在程序开始时使 CS、SS、DS 相等,程序也能正常地工作。如果对于一个程序中要用的数据区超过 64KB,或要求从两个或多个不同区域中去存取操作数,也是十分方便的,只要在取操作数之前,用指令给数据寄存器重新赋值就可以了。

在系统主板上可以安装 8K—48K 的 ROM 或 EPROM。它是 PC 机系统中不可缺少的一部分。由于只读存储器的芯片具有永久存储信息和只准读出不准写入的功能,因此,常把 PC 机系统的磁盘操作系统的引导程序、系统自检测试程序、I/O 驱动程序、BASIC 解释程序等固化在 ROM 中。

PC 机的 I/O 通道是 8088 CPU 总线扩展,它对信号进行隔离、增强驱动能力并由附加中断和直接存储器存取(DMA)来提高功能。I/O 通道一般包括:一个 8 位双向数据总线、20 根地址线、6 级中断、存储器和 I/O 读写的控制线、时钟和计时信号线、DMA 控制器,其中三个通道用于 I/O 设备与存储器之间的高速数据传输,一个通道用于对动态存储器进行刷新。

为了在硬件上对 IBM-PC 系统进行扩展,在系统板上安放了几个 I/O 插槽。对于 PC 机作为基本系统来说,一个槽要用来插 5.25 英寸软盘驱动器适配器,用以带 1—2 个软盘驱动器。一个槽用来插 IBM 单色显示器和一个并行打印机。也可以用两个插槽,一个用来插彩色并行打印机适配器,以带彩色显示器;另一个用来插并行打印机适配器,以带并行打印机。剩下的插槽可以用来扩展系统的内存 RAM,可以用来扩展 I/O 接口,并行或串行接口;可以作网络接口板,以形成局部网络,也可以用来插 A/D 或 D/A 转换板等等。

3. PC 机主机板上开关如何设置?

各类 PC 主机板上一般有两个开关 S1 和 S2(PC/XT PC/AT 无 S2)。这两个开关的设置用来确定:PC 机有无 8087 协处理器,内存存储器的总容量,显示器类型以及软盘驱动器的数量等。随着微机技术的发展,有的 PC 机已改为一个开关,或用软件系统配置方法。开关设置正确,或系统配置参数正确,主机能正常工作,否则自检给出错误信息。

IBM-PC 主机板上 S1 和 S2 开关,对 PC 机系统进行配置,故称系统配置开关。开关 S1 设置及其对应的功能如下:

位 置	功 能
1—7—8	决定所装 5.25 寸软盘驱动器个数
ON ON ON	无驱动器
OFF ON ON	一个驱动器
OFF OFF ON	两个驱动器
2	为 8087 协处理器留,不用协处理器置 ON
OFF	带协处理器
ON	不带协处理器

3-4	设置系统板上的 RAM 存储容量
ON ON	16KB
OFF ON	32KB
ON OFF	48KB
OFF OFF	64KB
5-6	显示器类型
ON ON	无
OFF ON	40×25彩色
ON OFF	80×25彩色
OFF OFF	单色显示器

开关 S2 设置与功能如下:

位置	功能
1-2-3-4	决定存储器选件的容量
OFF ON ON ON	选96KB
ON OFF ON ON	选128KB
OFF OFF ON ON	选160KB
ON ON OFF ON	选192KB
OFF ON OFF ON	选224KB
ON OFF OFF ON	选256KB
OFF OFF OFF ON	选288KB
ON ON ON OFF	选320KB
OFF ON ON OFF	选352KB
ON OFF ON OFF	选384KB
OFF OFF ON OFF	选416KB
ON ON OFF OFF	选448KB
OFF ON OFF OFF	选480KB
ON OFF OFF OFF	选512KB
OFF OFF OFF OFF	选544KB
5-6-7-8	均置于 OFF 状态

PC 机用户不少是 IBM-PC/XT 机,其系统板上 S1 为系统配置开关,该开关的位置与其对应的功能如下:

位置	功能
1	常规操作为 OFF 状态,ON 时通道循环返回
2	使用协处理器时置成 OFF,否则置成 ON
3-4	系统板存储容量
OFF ON	128KB
ON ON	192KB
OFF OFF	256KB
5-6	正在使用的显示器适配器类型
ON ON	无
OFF ON	40×25彩色/图形
ON OFF	80×25彩色/图形
OFF OFF	单色显示器
7-8	5.25寸软盘驱动器个数
ON ON	带一个软盘驱动器
OFF ON	带二个软盘驱动器

4. PC 机系统主板的 I/O 插槽是如何工作的?

PC 机系统主板上装有的 I/O 插槽,有的系统板上装有5个,有的系统板上装有8个。所剩余的空插槽它们是为 PC 机配加选件方便而设置的,每一个 I/O 插槽有62根接线,这62根线可以把各种信号提供加到扩充设备上。62根线并列操作,因此任何扩充板都可以插入到 I/O 插槽。所有扩充板均采用一种公用的62根线连接结构,因此,PC 机的功能扩充是比较方便的。

I/O 插槽62根线的具体用法分成四类:

(1)8条线用来给扩充板提供各种不同电压的电源;

(2)8条线用来传送数据总线的8位数据。所有数据,其中包括输入存储器和输入/输出设备的数据都要经过这8条线。

(3)20条用于寻址。当数据要输入或输出到存储器或输入/输出设备时,需要指定一个地址,以标明该存储器地址单元或设备号。用于存储时,所有20根都用来指明1024K个存储器中是哪一个单元正在被寻址。当用于输入/输出设备时,用9根地址线对512种不同设备进行寻址。

(4)26根线用来传送控制信号。例如读存储器命令,写存储器命令,输入/输出设备的读命令和写命令。

连接到 I/O 插槽的每一件设备都在不断地注视着输入/输出通道上的信号。例如,要发出一个输入/输出命令,它将通过在输入/输出该命令线上出现一个信号表示出来。而当该命令发出以后,所有输入/输出设备都注意地址线,而所有存储器电路都不管这些,如果发出一个存储器的写命令,就会产生与上述情况相反的情况,所有输入/输出设备将不管地址线,而所有的存储器电路都注视着那个给出的地址线。由于输入/输出的命令是请示性的,所以每个输入/输出设备都要注视地址线。一旦某个设备的地址出现在它的面前,它就立即采取行动,否则,它就等候下一次出现它的地址线。

5. PC 机的端口是干什么用的?

PC 机的端口是指微处理器8088用来简化和统一与外界进行联系的一种机构,它是微处理器与存储器以外的设备传送数据的唯一通路。

PC 机的微处理器与外围设备取得联系的设备例如键盘、软盘驱动器、硬盘驱动器等,都配备一个端口供它使用,一个端口就是一条假想的数据传送通路,给它分配一个端口号,它能根据处理器的命令接受或发送数据。当8088需要把数据发到某个端口时,它利用 OUT 指令来指定端口号,以及要被发送的数据,该数据通常为一两个字节。当从端口接收数据时,它利用 IN 指令来指定端口号,以及要接收的数据。8088可以同任何一个端口联系,而不管它是否在工作。

PC 机微处理器8088端口是用16位二进制数来规定的,因此可最多达64K个端口可供使用,而在实际应用上只分配了很少几个端口,因此,有大量的端口可供扩充。

但是要注意 PC 机实际上是以端口作为地址单

元,来区分不同的外设。因为一个外围设备不仅有数据寄存器,还有状态寄存器和控制命令寄存器,它们各需要一个端口才能区分,故一个外围设备往往需要数个端口地址。

问题解答

显示器·显示卡·显示系统

机电部第15研究所(100083) 孙梅英

1. 显示器与显示卡的关系?显示卡在实际显示系统中起什么作用?

微型计算机的显示系统由显示器和显示卡组成。

显示器是一种显示设备。它的功能是接收视频信号,在屏幕上显示字符、汉字、图形和图象。

显示卡又称为显示控制接口板、显示适配器。它是主机与显示器之间的接口。它直接插在主机的总线插槽内。它的作用是:接收 CPU 和主存储器发送的信息,输出视频信号和同步信号到显示器。

由此可见,显示器与显示卡的关系是非常密切的,是微型计算机显示系统中不可分割的两个组成部分。

2. 单色显示器与彩色显示器的区别?

阴极射线管显示器简称为 CRT 显示器,有时直接称为 CRT。它分单色和彩色两种。

阴极射线管(Cathode Ray Tube 简称 CRT)由电子枪、偏转装置和荧光屏组成。电子枪是阴极射线管的主要组成部分,包括灯丝、阴极、栅极、加速阳极和聚焦极。

单色显示器的工作原理:电子枪发射的电子束(阴极射线)通过聚焦和偏转系统撞击荧光屏,所发射的电子数取决于施加在电子枪上的电压。在控制电压的作用下,聚焦和偏转系统产生的电磁场将电子束聚焦于荧光屏上的某一点。当电子束轰击荧光粉涂层时,受轰击的部位便发出光来。光点的亮度取决于电子束中的电子数目。若我们控制电子束使其射向荧光屏上不同的点,便可显示出图形。

彩色显示器的工作原理与单色显示器基本相同。但是彩色显示器要有三个电子枪,分别对应红、绿、蓝三基色的信号强弱。在荧光屏内壁涂有彩色荧光粉,按三基色迭加原理形成彩色图象。

3. 显示器与显示卡怎样有机组合才是最合理的?CGA、EGA、VGA 之间是否有联系?同一个显示器中能否配备多种显示卡?

显示器是显示设备;显示卡是显示控制接口,两者组合在一起则成为显示系统。在配置显示系统时,要根据显示的技术要求(颜色、分辨率、显示器的扫描频率……等),选择适当的显示器和显示卡,组合成合理的显示系统,这样才能获得最佳的显示效果。

这里介绍几种显示卡的显示标准和特点:

1. MDA

MDA 是单色字符显示系统的显示控制接口板。它的特点是字符显示质量高,采用 9×14 点阵的字符窗口(指每个字符在屏幕上所占的点数,它包括字符显示点阵和字符间隔),满屏幕可以显示 80×25 行字符,对应的分辨率为 720×350 个像素。MDA 不能兼容图形方式。

2. CGA

CGA 是彩色图形/字符显示系统的显示控制接口板。它的特点是可以兼容字符与图形两种显示方式。在字符方式下的字符窗口为 8×8 点阵,因而字符显示质量不如 MDA,但是字符和背景可以选择颜色。在图形方式下,可以显示分辨率为 640×200 ,2种颜色的彩色图形或分辨率为 320×200 ,4种颜色的彩色图形。

3. EGA

EGA 集中了 MDA 和 CGA 两种显示标准的优点,功能进一步增强。字符显示窗口为 8×14 点阵,字符显示的质量优于 CGA 而接近于 MDA。图形方式的分辨率为 640×350 ,16种颜色,显示彩色图形的性能明显优于 CGA,而且兼容原 CGA 和 MDA 的各种显示方式。改进型 EGA(EGA+)或超级 EGA(Super EGA)的图形分辨率可以达到 640×480 或者 800×600 个像素。

4. VGA

VGA 是 IBM PS/2 系统的显示标准,在字符方式下的字符窗口为 9×16 点阵。在图形方式下,可以显示分辨率为 640×480 ,16种颜色的彩色图形或分辨率为 320×200 ,256种颜色的彩色图形。按 VGA 标准设计的显示控制接口板已经用于 IBM PC/AT 和 386 超级微型计算机系统。改进型 VGA(VGA+)或超级 VGA(Super VGA)显示控制接口板的图形分辨率可以达到 800×600 , 960×720 和 1024×768 个像素。

决定显示系统性能的另一个因素是显示器。各种不同型号的显示器,主要区别是扫描频率不同。显示卡的分辨率越高,输出视频信号的频率也越高,与它配套使用的显示器,扫描频率也应该越高。

对于固定频率的显示器来说,扫描频率是固定不变的。显示器的扫描频率必须与显示卡输出的视频信号频率相同,这样,才能得到显示系统设计时所要求的显示效果。

下面列出几种常用的显示卡所要求的显示器扫描频率。

显示卡	MDA	CGA	EGA	VGA
显示器水平扫描频率(KHZ)	18.432	15.75	21.85	31.75

随着显示技术的发展,1985年研制出一种新的显示器,它的扫描频率是可变的。它利用自动跟踪技术,使显示器自动与显示卡输出的视频信号相同步,同步的频率范围在 $15\text{KHz} \sim 35\text{KHz}$ 之间。这种显示器称为多同步显示器。

(下转第46页)

中软总公司(电脑大厦)

向您提供下列库存积压产品:

序号	名称	单位	销售价
1	XIDEX 5" 软盘	片	2.00
2	5" 单/双燕牌	片	2.00
3	5" 单/双3M	片	2.00
4	5" GW	片	2.00
5	JANUS 5" 加密盘	片	5.00
6	8" 单/双	片	7.00
7	8" 单/双3M	片	7.00
8	8" 双/双3M	片	13.00
9	8" 磁盘盒	个	10.00
10	8" 空盘盒50466	个	10.00
11	5" 清洗盘 JANUS	盘	15.00
12	5" 清洗盘3M	盘	20.00
13	8" 清洗盘	盘	10.00
14	框带:3070	只	20.00
15	框带:9400	只	20.00
16	电脑电话	部	100.00
17	起拔器	个	2.00
18	涂抹器	个	150.00
19	鼠标控制器	个	400.00
20	游戏控制板	块	100.00
21	LQ 彩打接口板	个	200.00
22	扩展板5550	块	300.00

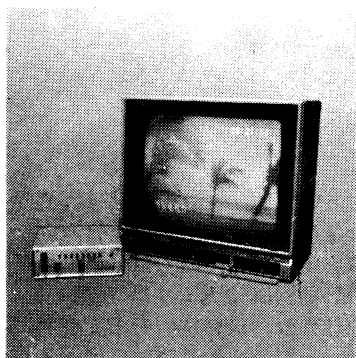
序号	名称	单位	销售价
23	ROM 板 16 * 16	块	300.00
24	内存扩充板 17-1	块	350.00
25	时钟板	块	50.00
26	串行口板 RS-232	块	100.00
27	多功能板 - 17 0520CE	块	350.00
28	HED 汉卡 AT	块	700.00
29	学习机打印卡	块	60.00
30	五笔字型卡	块	180.00
31	显示卡	块	180.00
32	数字化仪 WT - 4000	台	3500.00
33	绘图仪 SPL-400	台	3500.00
34	绘图仪 LP-3700	台	20000.00
35	绘图仪 MP-1000	台	1500.00
36	打印机9250A	台	3500.00
37	TX-104清洗剂	瓶	20.00
38	TX-248清洗剂	套	20.00
39	TX-300清洗剂	套	20.00
40	TX-308/309A 清洗 剂	瓶	20.00
41	TX-800清洗剂	包	20.00
42	扁平电缆压接工具 MT-PSSEOIB	台	500.00
43	扁平电缆压接工具 MT-PS-DIP	台	500.00
44	74、75等系列各种元 器件		

我公司一次性的处理库存积压产品共有3000多种,包括各种 PC 机、286、386各档次不同机型,价格特优,欢迎选购。

地址:北京海淀区学院南路55号电脑大厦,邮编:100081,乘16路汽车电脑大厦下车。

联系人:温友良(经营计划处副处长) 开户行:中国工商银行海淀分理处,帐号:461187-79

电话:831.7722转1302



一九九二年

总期第84期

電子與電腦

• ELECTRONICS AND COMPUTERS •

目 录

- 综述 •
 - 软件汉化原理 郑茂松(2)
- PC 用户 •
 - 编制能灵活打印二维报表的“报表生成系统” 苏士俊(3)
 - 在管理信息系统中实现计算模型的规则库方法。 孙永芳 吴泉源(6)
 - dBASE III 修改文件属性的方法 许再由(8)
 - 抗病毒软件
 - Turbo Anti-Virus V6.80A 简单 唐银红(8)
 - 新颖的日历打印通用程序 何贤敏(10)
- 学习机之友 •
 - CEC-I 汉字系统子程序的应用 傅叔平(11)
 - CATALOG 命令的改进 陈治浩(14)
 - 中华学习机查找汉字区位码 胡瑞辉(14)
 - 监控程序的浮动 赵旭(15)
 - 中华学习机特殊使用技巧 王冈(15)
 - 有趣的对比 丁志伟(16)
 - 中华学习机汉化引导程序 张福森(17)
 - 如何在苹果机持续运行一程序 张弢(18)
 - ProDOS 磁盘操作系统入门(续) 廖凯(19)
 - 汽车大赛 马宇昊(21)
- 语言讲座 •
 - 6502 机器语言程序设计
 - 第三章 6502 MPU 的寻址方式 朱国江(22)
- 初级程序员级软件水平考试辅导 •
 - 第三章 数据结构 宋丹颖(26)
- 学用单片机 •
 - 单片机最小应用系统与液晶显示器(LCD)的接口 张培仁(29)
- 学装微电脑 •
 - 电源 ON、OFF 自动运转装置 易齐干(32)
- 电脑巧开发 •
 - 微机视频接口器的研制 韦江维 覃龙生(37)
 - XMF 学习机(单驱扩展箱)改装为 XMF 及 PC 两用机 杨青海(38)
- 电脑游戏机 •
 - 第三章 F BASIC 的画面控制语句 于春(39)
- 维修经验谈 •
 - NP125 型复印机中一个易被忽视的故障 许鹰(42)
 - PC-88 键盘分析与故障维修 王耀亭(42)
 - APPLE II 电源常见故障维修方法 邓满园(44)
 - IBM PC/XT 机的软磁盘驱动器磁头校准程序 范思尧(45)
- 读者联谊 •
 - 普及型 PC 个人用户软件交流联谊活动问题解答(三) 王路敬(46)

封一:封面说明见 21 页

封二:SP1504 电源图

封三:CEC-I 中华学习机主控电路原理图

封四:PAL 制式转换卡

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

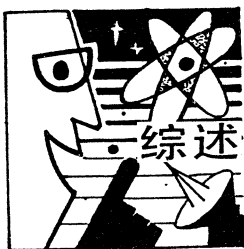
责任编辑:施玉新

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



软件汉化原理

中国科学院计算技术研究所(100080) 郑茂松

随着计算机在中国的推广使用,对中文软件的需求日益增多。鉴于软件汉化是中文软件开发的主要途径,提高软件汉化工作的效率和质量迫在眉睫,软件汉化研究势在必行。

中文软件的开发有两条途径:一条途径是重新设计和开发完全中文化的中文软件。这种中文软件的开发方法同世界上迅速发展的软件技术不相适应,不能充分利用国际上通用的丰富的软件资源,其应用也有很大的局限性。因此,这条途径看来很难走通。另一条途径是对西文软件实行汉化,这是目前广泛采用的一种切实可行的中文软件开发方法。

一、软件汉化的原理

软件汉化实质上是将被汉化的西文软件看成是一个缺少汉字处理功能的错误程序。软件汉化就是要测试和调试这个程序,直至它具有汉字处理功能为止。

根据测试和调试的原理和方法,我们把软件汉化的过程分为三个阶段:发现“错误”,查找“错误”,纠正“错误”。这里的“错误”是指“先天不足”、“水土不服”、“言语不通”三方面问题。前一个阶段是测试程序的过程,后两个阶段是调试程序的过程。通过这三个阶段的反复使用,反复迭代,反复试验,来完成软件汉化的全过程。

1. 发现“错误”

发现“错误”阶段的任务是:根据西文软件的功能和用户的要求,对被汉化软件拟定汉字处理功能描述,并据此设计测试用例,通过运行软件,检验测试用例是否符合预先拟定的汉字处理功能描述。若符合,则说明被汉化软件本来就具有这种汉字处理功能,在这一点上无需汉化;否则,说明被汉化软件不具有这种汉字处理功能,因而把不符合汉字处理功能的测试用例用于软件汉化的全过程。至于哪些测试用例符合预先拟定的汉字处理功能描述,取决于软件汉化的层次以及被汉化软件本身。

对于软件汉化这种特殊的软件测试,它所测试的软件一般都是目标代码。软件汉化人员往往只有软件的使用说明书,很少具备软件的技术说明书,面对可读性极差的目标代码,很难了解程序的内部结构和逻辑。因此,软件汉化设计测试用例一般无法采用白箱法,而只能采用黑箱法,而且主要采用黑箱法的等价分类法和边界值分析法。

在发现“错误”过程中,无论是拟定汉字处理功能描述,还是设计测试用例,或是运行软件试用各种汉字处理功能,都需要充分了解被汉化的西文软件的功能及其所处的环境,尤其是与增加汉字处理功能有关的原有功能。

2. 查找“错误”

查找“错误”阶段的任务是:根据从发现“错误”阶段得到的出错症状诊断出错误位置和出错原因。

软件汉化把对被汉化软件执行路径的分析看作是一个由“黑箱”到“灰箱”的测试和调试过程。所谓灰箱,是指通过分析对整个软件的功能和结构有个大概的了解。这种方法强调针对问题的蛛丝马迹有目的地寻找和定位与测试用例等价类相对应的目标代码段。这种方法既避免了繁重的全局分析工作,又可有效地确定出错位置。查找“错误”的具体方法如下:

(1) 猜测试探查找法

根据从发现“错误”阶段得到的迹象和线索着手,提出对出错原因的种种猜想,列出发生错误的种种可能原因,并用测试用例去试探可能会引起这些原因的程序代码,通过调试,逐个证实或排除有关出错原因的猜想,最后把出错范围缩小到最低限度,再经过认真分析和试验,最终找到程序出错位置。

猜测试探查找法没有确定的步骤,在很大程度上是凭经验和直觉来猜测出错原因和试探程序出错位置。这些经验和直觉取决于软件汉化人员对被汉化软件的了解程度、是否熟悉编译原理、程序设计经验多少、推理和归纳能力强弱等。尽管它似乎是一种“瞎凑”、“蛮干”和“凭运气”的方法,但对于具有丰富的程序设计经验和软件知识的软件汉化人员来说,却是一种常用的行之有效的办法。

(2) 逻辑分析查找法

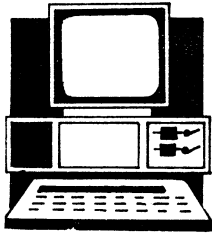
逻辑分析查找法从分析程序的内部逻辑着手查找“错误”。

对于全局逻辑分析,要分析被汉化软件的文件结构及内存映像结构,分析各个文件及其所含子程序的功能和相互调用关系。通过全局逻辑分析,可大致了解整个软件的总体逻辑结构,为进一步进行局部逻辑分析奠定基础。

当我们通过全局逻辑分析确定或基本上确定错误所在文件或子程序时,就可以对其中的各个疑点所在局部程序逐一进行局部逻辑分析,进一步确定出错原因和出错位置。局部逻辑分析又可分为静态分析和动态分析两种。静态分析就是把与错误有关的局部程序打印出来,仔细分析。动态分析就是通过调试工具分析程序的执行情况。静态分析可以帮助我们掌握局部程序的静态结构,动态分析可以帮助我们了解局部程序的动态执行过程。在局部逻辑分析过程中,常常把静态分析和动态分析结合起来使用。

(3) 启发式查找法

(下转第5页)



PC 用户

编制能灵活打印二维报表的“报表生成系统”

北京计算机学院(100044) 苏士俊

在一个具体的应用系统中,会有许多数据库文件,每个数据库又含多个字段,所以在打印报表时,不可能把全部字段都打印出来。常规的报表都有固定的格式和确定好的数据字段,但它们有时满足不了实际工作的需要。比如,经常遇到的随机查询,查询后要把满足条件的记录打印出来,但要求打印的数据项却常常与常规报表不同,或者只需打印一个仅含几个字段的简表。对于这样的系统,可以按下述方法为它设计一个相应的“报表生成子系统”,用来支持该种报表的表头和内容的打印。

由于数据库中的字段名常常用西文或汉语拼音,所以把数据库中的全部字段名和对应的汉字都放于一个“字段名数据库”内,查询记录的指针定位后,在某工作区打开该库,将字段名以菜单形式用汉字显示,请用户任选其中的某些字段。这时再用字符串相加的运算,将相应的中文字段名临时拼凑出一个汉字表头(表头=表头+中文字段名)。

假如用户选择了 NAME SEX, AGE..., 那么用 LISTNAME, SEX, AGE... 命令可以显示这些字段的值,但程序中难以预先写出这个随机的、含逗号的字段名表,普通用户也常常不会使用单条命令对数据库进行操作。为显示选中的字段内容,可使用下面 2 条命令:

```
STRING=NAME+" "+SEX+" "+STR(AGE,2,0)
? &STRING
```

如果把“? &STRING”语句放于 DO WHILE 循环中且移动指针,就可以代替 LIST 命令完成打印功能:

```
SELECT <区号>
USE <数据库文件>
LOCATE ALL FOR <条件>
DO WHILE .NOT. EOF()
? &STRING
CONTINUE
ENDDO
```

问题的关键是如何获得这个具有随机性质的 STRING,现简述如下:

我们从这个内存变量 STRING 后面的表达式结构可以得到启示,它与中文表头的结构相似,只是字段名是西文的,所以在拼接中文表头时,从存有中、西文字段名的“字段名数据库”中同时取出相应的西文字段名,也设法拼接在一起。

STRING 拼接时,适当地加入某些报表表格符号,然后把它放于语句(? &STRING)中,即可将报表符号连同字段的值形成一个完整的数据行,供打印使用。不断地移动满足条件的记录指针,不断地输出拼接好的

记录,就可以完成整个二维报表的灵活打印功能。

若字符串拼接时再增加串长的测试和判断,还可保证报表内容的总长不超过纸宽或屏幕宽度。例如,有一职工档案库(DA1.DBF),其库结构和数据如下:

.LIST STRUCTURE

```
库文件结构——文件名:C:DA1.dbf
库文件中的记录数量: 3
库文件的最后更新日期:01/01/80
```

字段	字段名	类型	宽度	小数
1	XM	字符型	6	
2	XB	字符型	4	
3	MZ	字符型	4	
4	ZC	字符型	6	
** 总计 **				21

.LIST

```
记录号: XM XB MZ ZC
1 李 森 男 汉 讲师
2 张丽萍 女 回 助教
3 王 立 男 汉 教授
```

注:为简化示范程序,该库中暂未设工资(GZ)和工作日期(GZRQ)等非字符型字段。

为编制“报表生成子系统”而设立的“字段名提示库”(ZD.TSK.DBF),其库结构和数据如下:

.LIST STRUCTURE

```
库文件结构—— 文件名:C:ZD-TSK.dbf
库文件中的记录数量: 4
库文件的最后更新日期: 01/01/80
```

字段	字段名	类型	宽度	小数
1	西文字段名	字符型	6	
2	汉字字段名	字符型	6	
3	字段长度	数值型	2	
** 总计 **				15

.LIST

```
记录号: 西文字段名 汉字字段名 字段长度
1 XM 姓名 6
2 XB 性别 4
2 MZ 民族 4
3 ZC 职称 6
```

下面是完成具有灵活性打印功能的简化程序:

* 字段名提示部分:

```
SET TALK OFF
USE ZD-TSK
A=0
```

```

M=0
CLEAR
@6,18 SAY " * * * * 相关项提示 * * * * (括
号内的数字为字段宽度)"
DO WHILE .NOT. EOF()
M=M+1
@7+A,20 SAY STR(M,2,0)+" : "+TRIM(汉
字字段名)
@7+A,32 SAY "("+STR(字段长度,2,0)+
")"
SKIP
A=A+1
ENDDO
* 以下拼汉字表头和 STRING:
总长=80
DO WHILE .T.
汉字表头="|"
STRING="'|'"
线="-"
剩余宽度=总长-2
@ 0,2 SAY "表头:"
DO WHILE .T.
CCD=0
@5,20 SAY SPACE(60)
@4,0 SAY SPACE(80)
@4,5 SAY "用户自定义项",
@4,21 SAY "请选择显示项的编码:" GET CCD
PICTURE "9" RANGE 1,M
READ
@ 4,50 SAY "稍等....."
GO CCD
OLD=剩余宽度
剩余宽度=剩余宽度-字段长度-1
IF 剩余宽度>=0
汉字表头=汉字表头+TRIM(汉字字段名)+
|"
STRING=STRING+"+"+西文字段名+"+"
+"'|'"
A=1
DO WHILE A<=字段长度+1
线=线+"-"
A=A+1
ENDDO
@ 4,50 SAY SPACE(15)
@ 1,0 SAY 线
@ 2,0 SAY 汉字表头
@ 3,0 SAY 线
CON="Y"
@ 5,0 SAY SPACE(80)
@ 5,5 SAY "屏幕剩余宽度:"+STR(剩余宽
度,3,0)
@ 5,20 SAY "继续输入吗(Y/N)?" GET
CON PICTURE "!"
READ
@ 5,20 SAY SPACE(60)

```

```

IF CON="N"
RK="Y"
@4,0 SAY SPACE(80)
@4,5 SAY " 认可吗(Y/N)?"GET
RK PICTURE"! "
READ
IF RK="Y"
EXIT
ELSE
@5,0 SAY SPACE(80)
@5,5 SAY" 不认可,请重新编辑表
头内容,按任一键..."
WAIT " "
@ 5,0 SAY SPACE(80)
EXIT
ENDIF
ENDIF
ELSE
?"屏幕超宽!此时的处理已省略。"
RK="Y"
EXIT
ENDIF
ENDDO
* 以下程序显示表头和报表内容:
IF (RK="Y").AND. (LEN(STRING)>1)
USE
SELECT 2
USE DA1
?线
?汉字表头
?线
DO WHILE .NOT. EOF()
?&STRING
?线
SKIP
ENDDO
WAIT
USE
RETURN
ELSE
@1,0 SAY SPACE(80)
@2,0 SAY SPACE(80)
@3,0 SAY SPACE(80)
LOOP
ENDIF
ENDDO
RETURN
注:①下面是操作时的屏幕显示情况:
表头:


|    |    |    |
|----|----|----|
| 姓名 | 职称 | 民族 |
|----|----|----|


用户自定义显示项,请选择显示项的编码:
屏幕剩余宽:59
* * * * 相关项提示 * * * * (括号
内的数字为字段宽度)
1: 姓 名 (6)

```

2: 性别 (4)
3: 民族 (4)
4: 职称 (6)

②拼接表头时,若增加表头的超宽处理,可直接用下面的程序段代替 RK="Y"和 EXIT 2条语句:

```

剩余宽度=OLD
CN=0
@ 4,0 SAY SPACE(80)
@ 4,0 SAY "超出80列!你可以:"
@ 5,0SAY " 1.反复改最后一栏 2.重新编辑表头
0.不再增加
@ 5,50 SAY "请选择(0-2):" GET CN PIC-
TURE "9" RANGE 0,2
READ
CN=STR(CN,1,0)
DO CASE
CASE CN="0"
RK="Y"
EXIT
CASE CN="1"
@ 5,0 SAY SPACE(80)
@ 5,5 SAY "屏幕剩余宽:"+STR(剩余宽度,
3,0)

```

(上接第2页)

有时,我们可以借助一些启发信息来分析出错原因和确定出错位置,这样做有时确能找到解决问题的捷径。启发信息来源很多,例如,通过观察工作环境,查看历史、暂时去掉程序的某些部分等手段,可以从中发现一些启发信息。启发式查找法的关键在于善于发现和善于利用启发信息。

(4)跟踪查找法

在无法采用上述方法找到出错位置的情况下,可采用跟踪程序的方法来查找“错误”。

跟踪查找法一般借助于调试工具采用线路跟踪方式,把单步方式与断点方式结合起来使用,即用断点执行方式以较快的速度通过确认无错的程序段,而出现错误迹象后再采用单步方式逐步跟踪,直至找到出错位置为止。当执行程序出错时,亦可采用回溯跟踪方式,从发现错误征兆的地方开始往回追溯程序代码,把出错前的一段程序的有关线路和赋值情况输出,以供软件汉化人员分析出错原因用。

3. 纠正“错误”

纠正“错误”阶段的任务:根据出错原因和出错位置,修改被汉化软件的程序,并用测试用例反复执行程序,检验是否符合汉字处理功能描述,直到它具有汉字处理功能并且排除因修改程序而引入的新错误为止。

在修改程序时,要斟酌修改后的程序代码,最好修改后的程序代码长度比修改前的程序代码短或一样长,以便覆盖;否则,就需要采用“打补丁”的办法把多余的部分接到空白位置,有时还需要增加文件的驻留长度。

软件汉化常常会产生副作用,这是难免的。产生副

```

CASE CN="2"
RK="N"
@ 5,0 SAY SPACE(80)
EXIT
ENDCASE

```

③本程序只处理了字符型字段的拼接。由于STRING中只准拼接字符型的内容,为处理其它类型字段,可在ZD.TSK.DBF中改设“变形字段名”字段,其内直接存放XM、XB、ZC、STR(GZ,6,2),DTC(GZRQ)...这时:

```

STRING=STRING+" "+TRIM(变形字段名)+" "+
+"'|'".

```

④下面是运行简化程序形成的STRING:

```

.?STRING
屏幕显示:'|'+XM +'|'+ZC +'|'+MZ
+'|'

```

⑤相应的报表内容如下:

姓名	职称	民族
李森	讲师	汉
张丽萍	助教	回
王立	教授	汉

作用的原因是多种多样的,往往是由于考虑不周或不完全弄清被修改程序的含义而引起的。例如,在被汉化软件中,一处的数据可能被多处程序所引用,因而当修改了这一处数据后,虽使一处程序纠正了错误,但却影响到另一处程序。在修改程序的过程中,应尽量避免产生副作用;当出现副作用时,要设法消除副作用。

软件汉化对程序的修改是否正确,是否存在副作用,除了回归测试外,在汉化软件投入使用后,还需要在使用过程中逐渐发现。要及时反馈用户发现的问题,不断维护和完善汉化软件。

二、软件汉化的工具

由于软件汉化是依照测试和调试程序的原理和方法进行的,因此需要借用测试工具和调试工具当作软件汉化工具。目前,测试工具尚很少用于软件汉化,而调试工具则是软件汉化的主要借用工具。

为了提高软件汉化的效率和质量,有必要总结软件汉化的规律,研制适合软件汉化的专用工具。软件汉化的专用工具一方面要汲取软件汉化借用工具的优点,另一方面要力求实现软件汉化的自动化。由于软件汉化是一个复杂的推理、分析、判断、归纳和思维过程,因而软件汉化的自动化很难实现。目前只能研制一些软件汉化辅助工具,例如,提示信息汉化辅助工具。

提示信息汉化辅助工具大多采用并行的提示信息汉化方法,自动地寻找西文提示信息,通过全屏幕编辑修改提示信息,使提示信息汉化工作并行化、自动化、简单化,从而提高软件汉化的生产力和可靠性。

(编者按:郑茂松同志编著的“中文软件和软件汉化”一书,即将由电子工业出版社出版发行;本文作了适当删节)。

在管理信息系统中 实现计算模型的规则库方法

孙永芳 吴泉源

高级管理信息系统的一个重要特征是系统含有大量的辅助决策模型,并且,模型的求解往往采用先进的人工智能技术和软件重用技术实现,因而系统一般都具有鲜明的智能化和工具化的特征。本文以我们在 IBM PC 机上实现的一个机电设备高级管理信息系统为背景,用基于规则的计算工具,描述这种智能管理信息系统的设计方法和实现流程。

一、嵌套型分段函数计算模型

在我们的管理信息系统中,为有效地评价和管理各种大型机电设备的性能完好状况,经高度抽象,最终以一种通用的嵌套型分段函数建立了相应的辅助决策模型,它的一般形式是:

$$Y = \begin{cases} f_1(x_{11}, \dots, x_{1m_1}) & \text{条件1} \\ f_2(x_{21}, \dots, x_{2m_2}) & \text{条件2} \\ \dots\dots\dots \\ f_n(x_{n1}, \dots, x_{nm_n}) & \text{条件n} \\ f_{n+1}(x_{n+1,1}, \dots, x_{n+1,m_{n+1}}) & \text{其它} \end{cases}$$

这里,变量 x_{ij} ($i=1, 2, \dots, n+1; j=1, 2, \dots, m_i$) 允许是另一分段函数。若某个 f_i 不含变量,或者说某个 $m_i=0$,这个 f_i 即表示一常量。

这类模型的含义是:若条件1成立,则用 f_1 计算 Y 的值;否则,若条件2成立,则用 f_2 计算 Y 的值;……;否则,若条件 n 成立,则用 f_n 计算 Y 的值;否则,用 f_{n+1} 计算 Y 的值。

按照传统的实现技术,这类函数不难直接编写到程序中。但是,这样做不利于模型的修改和扩充。我们知道,管理信息系统的生命力就在于它的易修改性和易扩充性。例如,在我们的系统中,许多设备的性能函数属经验函数,许多计算公式的结构和有关数据需在实际应用中不断完善;在该系统研制期间,不少设备的计算公式甚至还在讨论之中。以后,对于需要加入该系统管理的新设备,其性能函数更要经过多方研究平衡才能确定,并扩充到系统中。凡此种种,那种把大量计算公式编死在程序中的传统做法,既不可取,也行不通。为此,我们运用知识工程的方法,把计算模型依规则形式组织到规则库中,并编制一个规则解释程序,由该程序对规则库中的所有规则进行统一的解释与计算。这样,模型与程序是分离的,对模型的增删改就是对规则库的增删改,正象对数据库的增删改一样,不用改动和新编任何程序。事实上,数据库系统的基本功能

就在于用户可以随意进行系统所规定的各种库操作,而不会对系统程序的正常运行造成丝毫影响。

二、计算规则和计算规则库

所谓计算规则就是表示嵌套分段函数的规则,它的一般形式是:

(规则名, 条件1, 计算公式1,
条件2, 计算公式2,
……
条件 n, 计算公式 n,
TRUE, 计算公式 n+1)

其中,规则名对应于函数名 Y,条件部分为逻辑表达式,计算公式部分为算术表达式。“TRUE”表示恒真,当条件1至 n 均不成立时起作用,相当于分段函数的最后一个条件“其它”。

把各个分段函数对应的计算规则集中在一起,可以组成一个或多个数据库文件。每个这样的数据库文件就是本系统的一个计算规则库,它们的每一个记录由五个字段组成如下:

规则名	条件	计算公式	变量说明	注释
C(10)	C(50)	C(50)	C(50)	C(40)

每一记录共200个字符,对应于一个分段函数的一个分支,显然,整个分段函数的计算规则应当由 n+1 个记录表示,且它们的规则名均对应于同一个函数名。“变量说明”用以说明“条件”和“计算公式”中出现的变量;若说明的形式为

[变量名, 数据库名, 检索条件]

则相应变量的值由检索指定数据库文件按指定检索条件获得;若说明的形式为

{变量名, 规则库名}

则相应变量的值由调用指定规则库的指定规则获得,这个规则的规则名就是说明中的变量名。

每一个记录的注释部分用以增强规则的可读性,可以填写计算规则的各种注释,与规则的计算无关。

三、规则解释程序

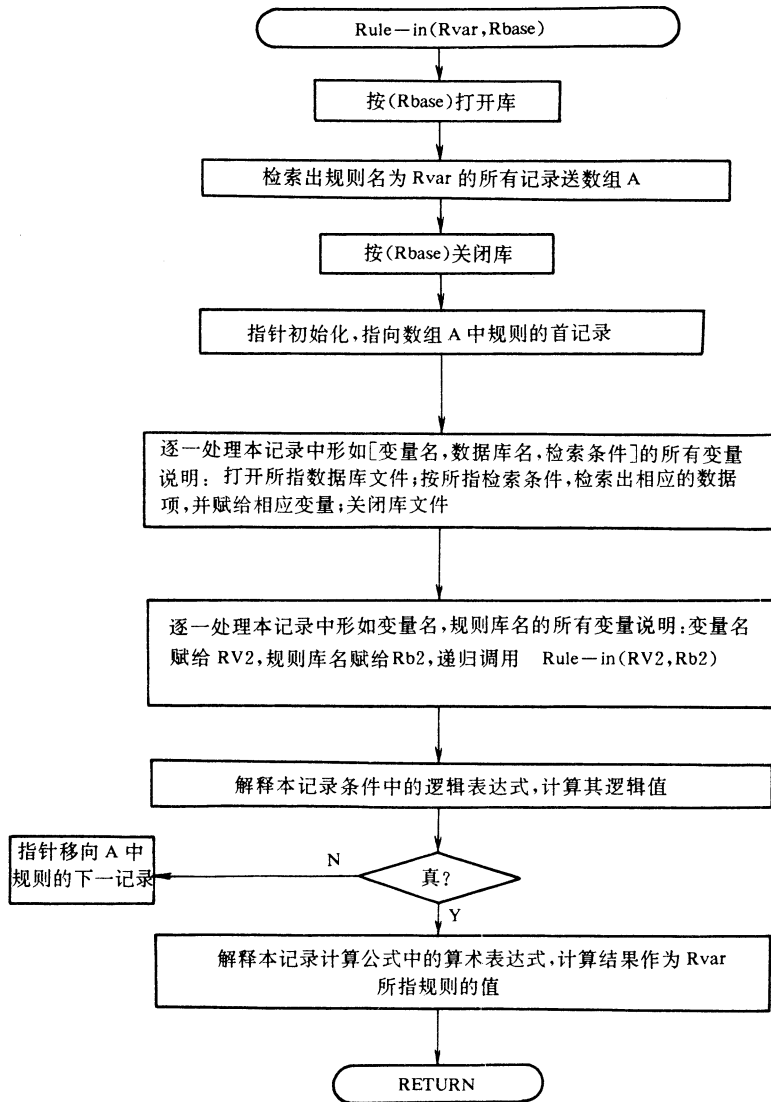
孤立地看,规则库与数据库没有什么区别,只有规则库加上规则解释程序才能完成动态的计算任务。本系统的规则解释程序的任务是,从指定的规则库中对指定的规则进行计算。其主要过程是 Rule.in (Rvar,

Rbase),调用时,变量 Rvar 已赋以指定的规则名,Rbase 已赋以指定的规则库名,要求计算结果回送作为变量 Rvar 所指规则的值。

规则解释过程 Rule.in 的程序流程详见流程图。注意,由于分段函数允许嵌套,该过程成了一个递归过程,即内部出现了递归调用。如果你的机器配有 Oracle

数据库系统,则该过程容易用高级语言 Pascal 嵌入 Oracle 实现。

在计算规则库中,如果把变量说明划分为两部分,每一部分分别对应于规则的条件部分和计算公式部分的变量说明,那么 Rule.in 的程序流程稍加改进可提高它的执行效率。



规则解释过程 Rule.in 的流程图

dBASE III 修改文件属性的方法

湖北江汉油田物探处(433100) 许再由

在 dBASE III 下,倘若把文件改为隐含属性,就不能被 DOS 系统的 DIR、COPY、ERASE 等命令显示、拷贝、删除,从而起到加密文件的作用。本文介绍用 dBASE III 生成小汇编程序的方法,通过 DOS 软中断 INT 21H 的 43H 功能调用,可以修改文件属性。dBASE III 生成汇编程序的关键是如何形成汇编语句的 16 进制机内码。为此,我们先把汇编语句的机内码换算成十进制数,利用 CHR(n) 函数将十进制数变成字符,存入数据库,再用 COPY TO<文件名>DELI WITH BLAN 命令把字符拷贝到后缀为 TXT 的文件中,该文件里的机内码就是所要求的汇编语句,最后换名后缀是 .COM 的文件,用 RUN 命令运行这个文件即可修改文件属性。但由于 dBASE III 不能产生 16 进制 00H,而汇编程序里的 43H 功能调用,规定 00H 为文件名的结束标志。所以我们建一个仅含 00H 机内码的文件如下:

```
C)DEBUG
-R CX
CX 0000
:1
-N BB.COM
-W
-Q
```

本文产生的汇编程序名为 JM.COM,用 COPY JM.COM+BB.COM 命令就能在 JM.COM 中补充一个 00H。源程序用 dBASE III 编写,也适用于 FOXBASE+ 系统。该程序还可修改子目录的属性,具有隐含属性的目录,不能被 DIR、TREE 命令发现。

程序名:JM.PRG

```
CLEA
SET TALK OFF
SET SAFE OFF
P=SPAC(12)
P1=SPAC(1)
@ 2,25 SAY "输入文件名:" GET P
@ 3,25 SAY "文件属性(普通0,只读1,隐含2,系统4)"GET
```

```
P1
READ
USE JM1
ZAP
APPE BLAN
P1=VAL(P1)+32
K = CHR(184)+CHR(01)+CHR(67)+CHR(177)+CHR
(P1)+CHR(186)+CHR(16)
K1=CHR(01)+CHR(205)+CHR(33)+CHR(205)+SPAC
(5)+TRIM(P)
REPL A1 WITH K+K1+CHR(26)
COPY TO JM DELI WITH BLAN
COPY FILE JM. TXT TO JM.COM
RUN COPY JM.COM+BB.COM
RUN JM.COM
RETU
```

数据库结构—数据库 :C:JM1.dbf
数据库中的数据记录个数: 1
数据库的最后更新日期 :12/03/90

字段	字段名	类型	宽度	小数
1	A1	字符型	28	
总计			29	

由 JM.PRG 产生的 JM.COM 汇编程序如下:

```
C)DEBUG JM.COM
9CAF:0100 B80143 MOV AX,4301
9CAF:0103 B120 MOV CL,20
9CAF:0105 BA1001 MOV DX,0110
9CAF:0108 CD21 INT 21
9CAF:010A CD20 INT 20
9CAF:010C 2020 AND [BX+SI],AH
9CAF:010E 2020 AND [BX+SI],AH
9CAF:0110 7177 JNO 0189
9CAF:0112 2E CS;
9CAF:0113 7072 JO 0187
9CAF:0115 67 DB 67
9CAF:0116 001A ADD [BP+SI],BL
```

抗病毒软件 Turbo Anti—Virus v6.80A 简介

湖北远安县财政局 唐银红

Turbo Anti—Virus v6.80A 是 CARMEL 软件工程公司于一九九〇年六月推出的抗病毒软件包,由于其诊断的病毒种类多达 154 种,诊断和消毒能力强,采用下拉式菜单,用户接口界面良好,易于使用,越用越觉

得方便。特别值得推崇的是,当 Turbo Anti—Virus v6.80A 驻留内存后,它就象一名忠诚的卫士,时刻监视着机器的工作,对各种操作,首先进行病毒检查,如发现病毒,立刻用文字和声音报警,提醒操作员及时消除

病毒。但当它驻留内存后,对机器的运行速度稍有影响。

一、Turbo Anti-Virus v6.80A 的构成

Turbo Anti-Virus v6.80A 全部存放在一张软盘上,共8个文件,因其主文件名称为 TNTVIRUS.EXE,故整个抗病毒软件包常被称为 TNT,在检测和清除病毒的过程中,其全部参数均可用下拉式菜单指定,也可用命令项中的高亮度字母来选择。BOOTS SAFE.EXE 用于检查命令行中参数指定的驱动器的引导记录,如果命令行上无参数,则检查缺省的驱动器。TSAFE.COM 使病毒检查程序驻留内存,用于监测计算机的工作,并可随时用热键激活。INSTALL.EXE 向指定盘和路径上安装整个抗病毒软件包,并在指定盘根目录上产生自启动文件 AUTOEXEC.BAT,使机器在启动时首先检查内存和分区表,并将病毒监测程序驻留内存。README.DOC 文件是软件包的使用说明,包括上述四个文件的使用方法和参数,本软件包能检查的154种病毒的名称和特点。

二、Turbo Anti Virus v6.80A 的使用

1. TNTVIRUS.EXE 检查和清除病毒软件,如果要检查和清除病毒,只需键入 TNTVIRUS<ENTER>。

执行该程序后,首先检查自身是否被病毒感染,如果检查到 TNTVIRUS.EXE 本身已被病毒感染,则向用户提出警告,告诫用户 TNTVIRUS 本身具有危险性。如程序本身完好,则对内存进行检查,通过后出现主控窗口。

主控窗口可以分成四部分:最上面一行是主命令,可用 ALT 键使光标移到主控窗口的命令行上,用光标键移动光标至所需的命令上,回车键使命令生效。中间又分为两部分,左边显示当前或指定盘上的树型目录结构,检查时则显示被检查子目录中的被检查的文件名称。右边显示有关系统信息,诸如工作盘的类型,子目录的个数,当前目录中还没有检查的文件个数,系统时间、日期,发现的病毒的名称,系统所提供的消息等。最下面的一行是对当前命令的说明,在任何时候,要想退出 TNTVIRUS,按 ALT-X 键返回 DOS。

2. BOOTS SAFE.EXE 检查引导区程序

BOOTS SAFE 检查指定盘的引导扇区和分区表,使用时对内存也进行检查。一旦发现引导扇区或分区表与映象图不同,则警告窗被打开,用声光报警,并显示以下信息:

Boot/Partition sector was modified

Restore Update Continue Stop

移动光标键可选择恢复(Restore)、更新映象图(Update)、继续(Continue)、终止(Stop)。

BOOTS SAFE 命令行上可带以下参数:它们是

D:指定检查的目标盘

M:对被检查盘的引导扇区产生新的映象图

R:从 A 盘读取映象图,并拷贝到指定盘

例如:BOOTS SAFE D:/M 的意义是:检查 D 盘的引导扇区和分区表,并将 D 盘的引导扇区映象图拷贝到

软盘中。

3. TSAFE.COM 驻留内存程序

本程序运行后,使保护程序驻留内存。它提供八个方面的保护功能,其控制窗口用热键 ALT-T 激活,用数字键设定是否保护。这八个功能是:软盘格式化保护、硬盘低级格式化保护、驻留保护、模拟写保护、检查所有文件、检查引导扇区、引导扇区保护、执行文件保护。TSAFE 命令行上的参数可改变热键,如要使 ALT-S 为热键,只需打入

TSAFE/AS

4. INSTALL.EXE 安装程序

当执行 INSTALL 程序后,系统自动将整个 Turbo Anti-Virus 软件包安装到指定盘上。当盘符被指定后,自动产生 TNTVIRUS 的子目录,并将主要文件拷贝到子目录中,同时生成 AUTOEXEC.BAT 文件,文件内容如下:

C:\TNTVIRUS\TSAFE

C:\TNTVIRUS\BOOTS SAFE C:

三、与 VIRUSCAN 2.0 V54 的比较

VIRUSCAN 2.0 V54 是美国 McAfee Associates 于九〇年元月注册发行的,因美国 AST 公司购买其版权并免费向其286用户提供,加上公安部的推荐,在我国拥有众多的用户,因此有必要作一比较。

1. Turbo Anti-Virus 采用下拉式菜单,整个运行过程中,检查的文件和有关运行参数都在主控窗口上显示,人机界面十分友好。

2. Turbo Anti-Virus 的运行参数可在使用过程中指定,参数(如盘符、路径、检查方式等)容易修改。而 Viruscan 的运行参数必须在命令行中给定,如果参数设置有错,则不能进入检查程序。

3. Turbo Anti-Virus 集病毒检查、消毒于一体,如果没有设置清除病毒的功能,程序在运行过程中一旦发现病毒,除用声光报警外,还询问用户是否消毒,而 Viruscan 中的 SCAN 程序只能检查是否染毒,消毒必须用其他软件,即使是新版 Viruscan v 61,查出病毒后,须另外调用 CLEAN-UP 清除文件病毒,如果病毒隐藏在分区表或引导扇区,则须调用 MDISK 程序消毒。

4. Turbo Anti-Virus 能检查154种病毒,而 Viruscan 仅能检查60余种。

5. Turbo Anti-Virus 能发现被检查盘中 BOOT/PARTITION(引导扇区)的任何改变,并能恢复到未修改的状态,也能根据用户的要求,将被检查盘的引导/分区扇区备份至软盘上。

6. 当 Turbo Anti-Virus 的 TSAFE 程序驻留内存后,能对系统资源进行保护,特别是当病毒企图对引导/分区扇区进行写(修改)操作时,它能产生一个假想的写保护,使病毒放弃传染的企图。

7. Turbo Anti-Virus 的消毒能力强,无论病毒是在内存,或是在文件及分区表内,均能有效地杀灭病毒,消毒后不须关机或热启动。

新颖的年历打印通用程序

浙江台州卫生防疫站(317000) 何贤敏

以往的年历打印程序,只能打印阳历,在很大程度上不能满足日常需要,本文提供的程序能同时打印阳历、阴历,打印格式合理,经过修改某些数据,又能起到通用的打印其他年份的作用,效果比较满意。

在程序设计过程中,采用对阳历、阴历分别计数、比较,阳历用普通数数字,阴历号数用国际汉字库中的半角数字,月号用罗马数字,阴、阳历同时打印的方法实现。

程序10~130为设置初值;140~200打印年历头;210~260及430~450是为保证上、下月衔接正确所作的数值计算及判断;270~410为打印年历正文。

若要进行其他年份的打印,只需对20、40、50、70、80、90、150行中的某些变量初值或常数作相应的修改即可。有关变量的含义已在程序中说明。

提供的程序在CCBIOS 2.13H、BASICA 3.30下调试运行通过。

```

10 WIDTH "LPT1:",150;DIM N1(15),N2(16),N$(15)
20 DATA 31,"一月",29,"二月",31,"三月",30,"四月",31,"五月",30,"六月"
30 DATA 31,"七月",31,"八月",30,"九月",31,"十月",30,"十一月",31,"十二月"
35 REM 上二个语句行是阳历各月总天数及月名
40 DATA 30,30;REM 阴历上一年延续至阳历今年的各月总天数
50 DATA 29,30,30,29,29,30,29,29,30,29,30,30;REM 阴历今年1-12月各月总天数
60 DATA 30,30
70 KK=3;REM KK为阳历今年第一天星期数(0-6)
80 J2=27;REM J2为阴历在阳历今年第一天的号数
90 A=11;REM A为阴历在阳历今年第一天的月份
100 R=12-A
110 FOR I=1 TO 12;READ N1(I),N$(I);NEXT
120 FOR I=1 TO 16;READ N2(I);NEXT
130 I=1;II=I+1;J=1;JJ2=J2;D$="日一二三四五六"
140 LPRINT "@P";
150 LPRINT TAB(17);"-----一九九二-----"
160 IF J > 12 THEN GOTO 470
170 LPRINT "@I"
180 LPRINT TAB(19);"~~~~~";N$(J);"~~~~~";
    TAB(69);"~~~~~";N$(J+1);"~~~~~";
190 LPRINT "@A";
200 LPRINT TAB(5);D$;TAB(55);D$
210 K=KK;IF K > 6 THEN K=0

```

```

220 KK=(N1(J)+K)-7*INT((N1(J)+K)/7)
230 J2=JJ2;JJ2=N1(J)-N2(I)+JJ2
240 IF N2(I)-J2+1>=N1(J) THEN II=I ELSE IF N2(I)-J2+N2(I+1)+1>=N1(J) THEN II=I+1 ELSE II=I+2
250 M=1;IF N2(I)-J2+1=N1(J) OR N2(I)-J2+N2(I+1)+1=N1(J) THEN M=3
260 IF II>12 THEN II=II-12
270 IF J1<N1(J) THEN J1=J1+1 ELSE GOTO 340
280 IF J1>9 THEN P=6*K+4 ELSE P=6*K+5
290 IF J2<=N2(I) THEN P$=CHR$(172)+CHR$(160+J2);J2=J2+1;GOTO 320
300 II=I-R;IF II<=0 THEN II=II+12
310 P$=CHR$(162)+CHR$(240+II);J2=2;IF MM=3 THEN I=I+1;MM=MM-1
320 LPRINT TAB(P);J1;P$;
330 IF K>=6 THEN K=0 ELSE K=K+1;GOTO 270
340 IF JJ1<N1(J+1) THEN JJ1=JJ1+1 ELSE GOTO 410
350 IF JJ1>9 THEN PP=6*KK+54 ELSE PP=6*KK+55
360 IF JJ2<=N2(II) THEN PP$=CHR$(172)+CHR$(160+JJ2);JJ2=JJ2+1;GOTO 390
370 III=II-R;IF III<=0 THEN III=III+12
380 PP$=CHR$(162)+CHR$(240+III);JJ2=2;IF M=3 THEN II=II+1;M=M-1
390 LPRINT TAB(PP);JJ1;PP$;
400 IF KK>=6 THEN KK=0 ELSE KK=KK+1;GOTO 340
410 IF J1<N1(J) OR JJ1<N1(J+1) THEN GOTO 270
420 J1=0;JJ1=0;J=J+2
430 IF N2(II)-JJ2+1>=N1(J+1) THEN I=II ELSE IF N2(II)-JJ2+N2(II+1)+1>=N1(J+1) THEN I=II+1 ELSE I=II+2
440 MM=1;IF N2(II)-JJ2+1=N1(J+1) OR N2(II)-JJ2+N2(II+1)+1=N1(J+1) THEN MM=3
450 IF I>12 THEN I=I-12
460 GOTO 160
470 LPRINT;LPRINT;END

```

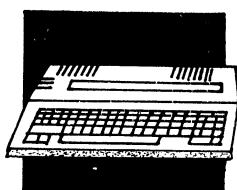
一九九二

一月							二月						
日	一	二	三	四	五	六	日	一	二	三	四	五	六
5	6	7	8	9	10	11	2	3	4	5	6	7	8
12	13	14	15	16	17	18	9	10	11	12	13	14	15
19	20	21	22	23	24	25	16	17	18	19	20	21	22
26	27	28	29	30	31		23	24	25	26	27	28	29

三月							四月						
日	一	二	三	四	五	六	日	一	二	三	四	五	六
1	2	3	4	5	6	7	5	6	7	8	9	10	11
8	9	10	11	12	13	14	12	13	14	15	16	17	18
15	16	17	18	19	20	21	19	20	21	22	23	24	25
22	23	24	25	26	27	28	26	27	28	29	30		

五月							六月						
日	一	二	三	四	五	六	日	一	二	三	四	五	六
2	3	4	5	6	7	8	1	2	3	4	5	6	
9	10	11	12	13	14	15	7	8	9	10	11	12	13
16	17	18	19	20	21	22	14	15	16	17	18	19	20
23	24	25	26	27	28	29	21	22	23	24	25	26	27

(由于地方有限年历适当删去)



学习机之友

CEC—I 汉字系统子程序的应用

成都市成人教育学院(610017) 傅叔平

中华学习机 CEC—I 上固化的汉字系统是一个丰富的资源,其中有很多有用的子程序。用户程序直接调用它们往往事半功倍,有时还可获得一般情况下难以得到的效果。直接使用系统子程序还可大大缩短程序长度,对提高编程效率,节约内存空间及增强程序的可读性等方面均有积极的意义。

汉字系统的管理程序主要分为接口部分(\$C300~\$C3FF)和主体部分(辅存2的\$EC00~\$FFFF)。对于接口部分的子程序,用户程序可用通常的方式直接调用,而对主体部分的子程序则应以如下方式调用: JSR \$C3AB;进入辅存2

JSR <子程序入口>

JSR \$C3B9;回到主存2

下面介绍部分子程序应用的例子。

1. 代码转换

CEC—I 一共用了四种汉字内码:区位码、国标码、异形国标码和内码,常涉及到这些代码之间的转换。前三种代码之间有非常简单的数学关系,转换很容易。内码与这些代码之间的转换要复杂一些,转换时可直接调用下面的两个子程序:

\$C389——功能:将异形国标码转换为内码,但转换后最高位为1,入口、出口参数均在累加器 A 中。

\$C392——功能:将内码转换为国标码,入口、出口参数均在累加器 A 中。

程序1是将\$300单元中的区位码转换成内码。程序中前三行是先将区位码转换成异形国标码。

【程序1】

```

0310— AD 00 03      LDA      $ 0300
0313— 18           CLC
0314— 69 A0        ADC      # $ 0A0
0316— 20 89 C3     JSR      # C389
0319— 29 7F        AND      # $ 7F
031B— 8D 00 03     STA      $ 0300
031E— 60           RTS

```

2. 自动切换汉字输入方式

汉字输入方式的切换一般是用按功能键 F1~F5 的方法。但有的程序要求在某些时刻自动切换输入方式,这时,调用如下的子程序可以达到目的:

\$EC18—功能:置汉字输入方式;入口参数:累加器 A 中是 F1~F5 的键码,分别为 \$ 01、\$ 0C、\$ 17、\$ 14、\$ 06)。

程序2可将当前状态转入区位输入方式,第5~6行是恢复执行前面指令时清掉的状态提示字。

【程序2】

```

0310— A9 17      LDA      # $ 17
0312— 20 AB C3   JSR      $ C3AB
0315— 20 18 EC   JSR      $ EC18
0318— 20 B9 C3   JSR      $ C3B9
031B— A9 12      LDA      # $ 12
031D— 20 2B C3   JSR      $ C32B
0320— 60         RTS

```

3. 包括状态行的清屏

\$EC54——功能:全屏幕清屏

HOME 或 PRINT CHR \$(12)在汉字状态下清不了状态行。若想连状态行一起清除,可用如下指令:

JSR \$C3AB

JSR \$EC54

JSR \$C3B9

RTS

此子程序清屏后光标仍在当前行,而不是在屏首。它也可用于清高分辨率图形第二页。

4. 打印机失控和 F4/F5键引起死机的解除

\$EC94——功能:初始化打印机工作单元和功能键扩充口。

不少 CEC—I 的用户曾遇到这样两种情况:一是在汉字状态下工作以后,再用 PR #6 热启动 DOS,以后就会出现打印机失控的现象;二是使用某些要用 F4 或 F5 键的程序后又热启动 DOS,以后一按 F4 或 F5 键就死机。这两种现象都是因为汉字系统用的第三页单元中的值在 DOS 启动的过程中被破坏所致。这时,用上述初始化程序即可解决问题。可用下列指令:

JSR \$C3AB

JSR \$EC94

JSR \$C3B9

RTS

它们相当于如下 BASIC 程序(程序3)的功能,

【程序3】

```

10 POKE 1659,0:POKE 1787,1:POKE 1915,1:POKE
2043,40
20 POKE 913,96:POKE 911,96
30 POKE 915,76:POKE 916,27:POKE 917,65:POKE
918,51:POKE 919,1:POKE 920,24

```

也可用下列键盘操作完成同样的初始化工作:

①按复位键(CTRL)←(RESET)进入西文 BASIC 状态;

②POKE 1147,0

③按<中文>键进入汉字状态。

5. 不显 CEC 汉字封面

有的软件希望在第一次进入汉字状态时,不显示

CEC 原来的汉字封面而直接显示自己设计的封面,这时也可利用上述初始化子程序 \$EC94(程序4):

【程序4】

```
0300— A9 03 LDA # $ 03
0302— 8D 7B 04 STA $ 047B
0305— 20 AB C3 JSR $ C3AB
0308— 20 94 EC JSR $ EC94
030B— 20 B9 C3 JSR $ C3B9
030E— 60 RTS
```

注意不能只用程序4的前两条指令,否则由于跳过初始化打印机控制单元的步骤,虽然可以不显示 CEC 汉字封面而进入汉字状态,但一旦打印就会紊乱。

6. 获取汉字点阵

\$ECCD——功能:查找汉字点阵;入口参数:国标码高字节置入 \$D6、国标码低字节置入 \$D7,若是 ASCII 字符,则 \$D6置0、ASCII 码放入 \$D7。出口参数:查找成功则状态标志寄存器的进位位 C 为1,否则为0。调用后,若查找成功,则16×16点阵的32个字节存放在 \$94D0开始的32个单元。

该子程序在汉字及西文状态下均可使用。

由于汉字在屏幕上的点与对应内存映射区中字节的各位的对应关系是倒序的,CEC-I 为了提高效率,从汉字库中读出的字节的各位也是倒序的。所以若要得到正确的点阵数据,在调用本子程序后还要将 \$94D0开始的32个字节的各位颠倒过来。

还要指出的是,汉字屏幕上所有要显示的汉字或字符,均要用 \$94D0开始的32个字节作为暂存点阵字模的缓冲区。故调用本子程序获得点阵数据后,应立即将其从 \$94D0~\$94EF 移到安全区域,否则当显示新字或程序运行完后显示系统提示符时,均会冲掉 \$94D0~\$94EF 中有用的点阵数据。另一个办法是在西文状态下调用本子程序,就不会发生上述问题了。

程序5是以16个四位16进制数的形式按行向量输出一个汉字或 ASCII 字符的16×16点阵数据,该程序可作为一个子程序被调用,它有两个入口,入口1为 \$6000,调用前须将汉字的内码高字节放 \$300、低字节放 \$301。入口2为 \$6010,调用前将汉字的国标码高字节放入 \$D6,低字节放入 \$D7。对字符,\$D6置0、\$D7放入 ASCII 码。调用后汉字点阵在 \$310开始的32个字节中,程序5既可在中文状态下运行也可在英文状态下运行。其中地址 \$6029~\$603F 部分是调用监控子程序以四位16进制形式输出点阵数据。

程序5-1是调用程序5的 BASIC 程序,此程序可以高速地获得国标一、二级汉字中的任何汉字或符号的点阵数据。

【程序5】

```
6000— AD 00 03 20 92 C3 85 D6
6008— AD 01 03 20 92 C3 85 D7
6010— 20 AB C3 20 CD EC 20 B9
6018— C3 90 25 A0 1F B9 D0 94
6020— 20 41 60 99 10 03 88 10
6028— F4 A0 00 B9 10 03 C8 BE
```

```
6030— 10 03 20 41 F9 20 48 F9
6038— C8 C0 20 90 EE 20 8E FD
6040— 60 A2 08 0A 6E 02 03 CA
6048— D0 F9 AD 02 03 60
```

【程序5-1】

```
20 PRINT ;PRINT CHR $(4);“BLOAD PRO5”
25 CH =768;CL=769;FC=24576
30 INPUT “汉字?”;A$;IF A$ =“”THEN 90
40 POKE CH,ASC(MID$(A$,2,1));POKE CL,ASC
(RIGHT$(A$,1))
50 CALL FC
60 GOTO 30
90 DEL 20,20
```

]RUN

汉字?啊

```
0004 2F7E F904 A904 AA14 AA7C AC54 AA54
AA54 A954 E974 AD54 0A04 0804 0814 080C
汉字?
```

7. 在第11行显示汉字

\$EE0D——功能:显示字符或汉字;入口参数:A 中放字符的 ASCII 码或汉字码。这里的汉字码为三字节,第一个字节为汉字引导符 \$7F,第二、三两个字节为汉字的异形国标码的高字节和低字节,显示一个汉字须分三次调用。

与 CSWA 和 GB·CSWA 不同,此子程序能在屏幕的任何地方显示汉字,包括第11行(提示行),而前者在第11行显示时,和 BASIC 中一样,超过20列时显示的汉字会出现残缺不全的现象。

程序6是在屏幕第11行第33列处显示“A 啊”。

【程序6】

```
0300—A5 24 48 A5 25 48 A9 0A
0308—85 25 A9 20 85 24 20 AB
0310—C3 A9 41 20 0DEE A9 7F
0318—20 0DEE A9 B0 20 0D EE
0320—A9 A1 20 0D EE 20 B9 C3
0328—68 85 25 68 85 24 60
```

8. 造字

\$F0B5——功能:将 \$94D0开始的32个字节表示的16×16点阵显示出来。

利用该子程序可以造任意16×16点阵的汉字或其它符号,这只需在调用前将点阵数据放入上述单元即可,但要注意,点阵数据字节的各位位置为左右相反。另外,该子程序只能用于显示。

程序7是供显示用的造字程序,使用该程序前应将由用户造的字库置于 \$7000开始的区域,每32个字节为一个自造字的16×16点阵,各个字依次排队,并将 \$3F5~\$3F7三个单元置为 \$4C、\$00、\$60,调用时用命令“&n”,表示显示用户造的第 n 个字,该程序中已有先颠倒点阵字节各位的功能,故自造点阵中的各字节可直接按正序给出。

其实,不只是造汉字,只要是16×16点阵的任何符号或图形均同样可造,用户只要自己设计所需的点阵即可。还可以利用造图形的方法获得动画效果。程序

7-1是一个简单的例子,它在屏幕上显示一个不断张口、闭口的小精灵。方法是造两个相似的小精灵图形,一个张口,一个闭口,在同一位置交替显示,并适当控制显示时间。程序7-1中第三行是将磁盘上的点阵数据装入内存,在这个例子中用的点阵数据是:

```
7000— 0F F0 3F FC 7F FE FF FF
7008— C3 C3 9B 9B BF BF DF DF
7010— EF FB 63 E3 70 06 30 06
7018— 38 0E 1C 1C 0F 78 07 E0
7020— 0F F0 3F FC 7F FE FF FF
7028— C3 C3 9B 9B BF BF DF DF
7030— FF FF 7F FB 77 F6 3B EE
7038— 3C 1E 1F FC 0F F8 07 E0
```

【程序7】

```
6000— A0 00 B1 B8 C9 30 90 18
6008— C9 3A B0 14 20 7B DD 20
6010— FB E6 8A 18 69 20 85 D7
6018— A9 2A 85 D6 20 23 60 60
6020— 4C C9 DE 20 AB C3 20 30
6028— 60 20 B5 F0 20 B9 C3 60
6030— A5 D6 C9 2A F0 06 20 CD
6038— EC 4C 6A 60 A9 00 85 06
6040— A9 70 85 07 A5 D7 38 E9
6048— 20 AA CA F0 10 A5 06 18
6050— 69 20 85 06 A5 07 69 00
6058— 85 07 4C 4A 60 A0 1F B1
6060— 06 20 6B 60 99 D0 94 88
6068— 10 F5 60 A2 08 0A 6E 78
6070— 60 CA D0 F9 AD 78 60 60
```

【程序7-1】

```
1 B$ =CHR$(13)+CHR$(4)+"BLOAD"
2 PRINT B$;"PRO7"
3 PRINT B$;"PRO7-1.LIB,A$7000"
4 POKE 1013,76;POKE 1014,0;POKE 1015,96
5 PRINT CHR$(12)
6 FOR J=1 TO 10
7 GOSUB 100
10 VTAB 4;HTAB 10;& 1
15 GOSUB 100
20 VTAB 4;HTAB 10;& 2
50 NEXT
90 DEL 1,3;END
100 FOR I=1 TO 300;NEXT
120 RETURN
```

9. 清屏幕任一行

\$F388——功能:清任一行(包括提示行);入口参数;\$25单元中置要清的行的行坐标(0~10)。

程序8的作用是清除提示行,若要清其它行,只要将\$307单元的值改为相应的行坐标即可。

【程序8】

```
0300— A5 24 48 A5 25 48 A9 0A
0308— 85 25 20 AB C3 20 88 F3
0310— 20 B9 C3 68 85 25 68 85
```

0318— 24 60

10. 在汉字状态中打印文本字符

\$F656——功能:打印文本字符;入口参数:累加器A中是打印字符的ASCII码。该子程序可在各种状态(中文、西文、高分辨率图形、低分辨率图形)中打印出文本ASCII字符(包括控制字符),而且它不需要连通和断开打印机的指令。

程序9运行时打印出“ABCDE”五个字符,妙就妙在它在中文状态下打印出7×5点阵的文本字符。使用程序9时将要打印字符的ASCII码置于\$314开始的单元,最后跟回车码\$0D和结束标志\$00。

【程序9】

```
0300—A2 00 20 AB C3 BD 14 03
0308—F0 06 20 56 F6 E8 D0 F5
0310—20 B9 C3 60 41 42 43 44
0318—45 0D 00
```

11. 打印机换页

若利用上述子程序\$F656向打印机输出一个换页码\$0C,则可在汉字状态(也可在其它任何状态)下使打印机换页,即用如下指令:

```
LDA # $0C
JSR $C3AB
JSR $F656
JSR $C3B9
RTS
```

汉字系统中可供用户程序直接调用的子程序还有很多,由于篇幅所限就不一一列举了。

新书征订

- | | |
|----------------------------|--------|
| ① 电脑编程技巧 200例 | 9.80元 |
| ② 电脑维修经验 200例 | 6.30元 |
| ③ 微机可编程序控制器原理使用及应用实例 | 5.50元 |
| ④ 实用家电维修与制作集锦 第一集 | 13元 |
| ⑤ 电冰箱、冷藏柜、空调器、电动机维修技术和修理经验 | 8.40元 |
| ⑥ 国内外黑白电视机修理经验300例(续集一) | 15元 |
| ⑦ 业余电子制作详解(初级篇) | 8.30元 |
| ⑧ 业余电子制作电路技术手册 | 8.60元 |
| ⑨ 音箱制作精选 | 16.10元 |
| ⑩ 实用录像机维修技术 | 14元 |
| ⑪ 微型计算机系统原理分析与维修(上、中、下) | 22元 |
| ⑫ IBM-PC和长城系列微型机操作应用实践问答 | 14.30元 |

购书者请汇款至北京农展馆南路十二号(通广大厦)中国电子学会北京爱谊电子服务部 邮编100026。

本服务部还为读者提供电子类书刊100余种,购书者可直接来函索取书目(自付邮资0.30元)。

CATALOG 命令的改进

广西南宁第二十四中学(530001)陈治浩

本文提出了一种对 APPLE DOS 中 CATALOG 命令的改进方案(见程序一和程序二)。改进后的命令能在列完目录后显示出磁盘上的文件数目和自由空间的扇区数,并且使 CATALOG 命令适应于中文状态。

一、磁盘中文件数和自由扇区数的显示

及时地了解磁盘的使用情况,无疑是很意义的。而利用执行 CATALOG 命令时顺便显示磁盘中自由扇区数的方法也许是最为实用的了,具体实现的方法本刊以前曾有过介绍。但是,将自由扇区数显示在原命令显示磁盘编号的地方,笔者认为是一种缺憾。因为了解磁盘编号对于管理磁盘也是很有益处的,特别是在使用的磁盘比较多时更是如此。本文介绍的方法保留了原命令中显示磁盘编号的功能,当 CATALOG 命令列完目录以后才显示出磁盘中的文件数和自由扇区数,其显示格式与 PC-DOS 中的 DIR 命令一样。如果读者使用的 DOS 已经过修改,CATALOG 命令具有中断列目录功能的话,那么显示出的文件数仅是在中断以前已经被列出的文件数目。

二、CATALOG 命令在中文状态下的使用

APPLE DOS 设定文件名由30个字符(ASCII 码)组成,即使我们的文件名只有一个字符,DOS 也要用空格将其补足30个字符。这种设定不适用于 CEC-I 等微机在中文状态下的使用。CEC-I 等微机在中文状态下每行只能显示34个字符,一个汉字在显示时占两个字符的位置。列目录时,每个文件名前面要用7个字符来显示文件类别、长度等信息,这样就要两行才能列完一个文件名。而实际上一般文件名的长度并没有(也不需要)30个字符,后面全是空格,故列目录时看起来就是空一行列一个文件名了。如果我们在文件名中加入四个以上的汉字,列目录时“空行”就不存在了。这是因为每个汉字在内存中用三个字节表示,而显示时只占两个字符的位置,含四个以上汉字的文件名在显示时的长度不会超过26个字符,故在一行里可以显示完毕,“空行”也就自然消失了。

我们只要在 DOS 中修改一个字节(程序二中的60语句),就可使 CATALOG 命令只显示文件名前面的26个字符。这样,不管文件名中是否有汉字,都不会在列目录时产生空行了。这时,如果某一文件名的长度超过26个字符的话,在列目录时其超出的部分将被略去,我们就看不到完整的文件名了,但在磁盘中的文件名仍是完整的。

程序二中的50语句使 CATALOG 命令每页只列八

个文件,若在西文状态下使用可将其删去。

若用 INIT 命令格式化一张新磁盘,载入新磁盘的将是修改后的 DOS,程序一、二均没有必要保留在磁盘上了。

程序一

```
BC56- 20 2F
BC58- AE A9 00 85 06 85 08 85
BC60- 09 60 20 ED FD E6 06 60
BC68- 20 70 FC 20 48 F9 A6 06
BC70- A9 00 20 24 ED A2 0B BD
BC78- B7 BC 20 ED FD CA D0 F7
BC80- A0 A0 B9 F2 B3 F0 0D 0A
BC88- 90 FB 48 E6 08 D0 02 E6
BC90- 09 68 D0 F1 88 D0 EB A6
BC98- 08 A5 09 20 24 ED A2 0D
BCA0- BD AB BC 20 ED FD CA D0
BCA8- F7 4C 7F B3 E5 E5 F2 E6
BCB0- A0 F3 F2 EF F4 E3 E5 F3
BCB8- A0 A0 A0 A9 F3 A8 E5 EC
BCC0- E9 C6 A0
```

]BSAVE DIR.1,A\$BC56,L\$6D

程序二

```
10 POKE 44487,86:POKE 44488,188
20 POKE 44518,98:POKE 44519,188
30 POKE 44589,104:POKE 44590,188
40 PRINT CHR$(4);"BLOAD DIR.1"
50 POKE 44452,12:POKE 44605,8
60 POKE 44567,25:NEW
```

中华学习机(CEC-I) 查找汉字区位码

福建 永定一中(364100)胡瑞辉

随着高考的不断改革,许多省市的高考招生用计算机统计分数和录取志愿。由于计算机处理的需要,考生在报名时填写考生信息卡,要求考生将自己的姓名用国标区位码来填写。通常用人工对照区位码表来查找汉字区位码是比较麻烦的,而且容易查错。本文的程序可以帮你解决困难。

程序运行后首先进入提示是否要打印输出,按下功能键 F2 后即可用汉语拼音输入所要查找的汉字姓名。40-120 语句和子程序 500-560 语句是实现将所输入的汉字的 ASCII 码转换成区位码,其中字符串数组 AS(1)的每一个元素存放一个汉字的区位码。130-160 语句是输出所查找的汉字区位码。

```
10 PR# 3: PRINT: HOME
15 VTAB 5: HTAB 8: INPUT "是否要打印?(Y/N)";C$
20 HOME: VTAB 3: HTAB 2: PRINT "按 F2 键后,请输入汉语拼音查找姓名"
```

```

25 IF C$ = "Y" THEN POKE 1659,3
30 VTAB 5: HTAB 10: INPUT "姓名:";A$
40 A = LEN(A$):N = INT(A/3)
50 FOR I = 1 TO N:FOR J = 0 TO 1
70 X$ = MID$(A$,3*I - 1 + J,1)
80 X = ASC(X$)
90 GOSUB 500
100 A$(I) = A$(I) + Y$
110 NEXT J:NEXT I
130 VTAB 6: HTAB 10: PRINT "区位:";
140 FOR I = 1 TO N
150 PRINT A$(I)," ";
155 A$(I) = ...
160 NEXT I: PRINT : PRINT: POKE 1659,0
170 VTAB 8: HTAB 5: PRINT "按任意键查找另一姓名区位"
180 GET B$: GOTO 20
500 IF X < 34 THEN X = X - 28: GOTO 540
510 IF X < 44 THEN X = X - 29: GOTO 540
520 IF X < 58 THEN X = X - 30: GOTO 540
530 X = X - 31
540 Y$ = STR$(X)
550 IF X < 10 THEN Y$ = "0" + Y$
560 RETURN
]RUN

```

姓名:张小彬
 区位:5337,4801,1782,
 姓名:李虹
 区位:3278,2671,

监控程序的浮动

江苏、苏州、景范中学(215005) 赵旭

中华学习机的监控程序驻在 \$F800~\$FFFF 空间。利用监控命令可以方便地对内存中的信息进行检查、修改、比较、传送等工作。但中华学习机的汉字管理程序及16K RAM卡所占空间都与监控所占空间有重叠。当计算机切换到汉字管理程序或16K RAM卡时,监控程序就不能运行。也就不能直接利用监控命令来剖析汉字管理程序或已驻在16K RAM卡中的程序。更不能利用监控命令将机器程序直接写入16K RAM卡。

针对以上情况,笔者采用将监控程序先移到 \$2800~\$2FFF 空间,修改程序中有关绝对转移地址,再调用浮动后的监控程序的方法,解决了以上问题。

以上工作是由一个工具软件完成。具体操作方法:先输入程序一和程序二。再运行程序一,即刻计算机发出“嘟”声,并进入浮动后的监控程序。以后如再要进入

浮动后的监控程序,只要在 BASIC 状态键入 CALL24576(或监控状态键入6000G)。

应用举例:1. 在监控状态键入 C3AB G,中华机将内存切换到汉字管理程序。此时即可用监控中 L 命令直接阅读汉字管理程序了,也可用 M 命令将汉字管理程序移到其他空间。2. 键入 C3B9 G 退出汉字管理程序后,再键入 C08B C08B,中华机将内存切换到16K RAM卡。如果 RAM卡中已有程序,则可用 L 命令直接阅读。当然也可直接用监控命令修改 RAM卡中的程序,或直接将机器程序写入 RAM卡中。

程序一

```

10 FOR I=512 TO 527
15 READ A:POKE I,A
20 NEXT I
25 POKE 1017,16:POKE 1018,96
30 CALL -144
50 DATA 178,184,176,176,188,198,184,176,176,
174,198,198,198,198,153,141

```

程序二

```

6000- A9 2D 8D 54 AA 8D 56 AA
6008- A9 2E 8D BF 2F 4C 65 2F
6010- 20 75 FE 20 2C FE A2 00
6018- 20 8C F8 A5 2F C9 02 D0
6020- 45 A0 01 38 B1 3A E5 40
6028- CB B1 3A E5 41 90 37 A0
6030- 01 A5 3E F1 3A A5 3F C8
6038- F1 3A 90 2A 38 A0 01 A5
6040- 40 E5 44 85 FE A5 41 E5
6048- 45 85 FF 38 A5 3A E5 FE
6050- 85 00 A5 3B E5 FF 85 01
6058- B1 3A 38 E5 FE 91 00 C8
6060- B1 3A E5 FF 91 00 20 53
6068- F9 85 3A 84 3B 18 A5 3A
6070- E5 3E A5 3B E5 3F D0 9E
6078- 4C 00 60

```

中华学习机特殊使用技巧

北京二十二中(100007) 王冈

1. REM 加密

这是一种较难被识破的 BASIC 程序加密方法,用这种方法加密的程序只能运行,而不能列出清单。这种加密方法的关键在于借用了 DOS 命令的引导字符 CTRL-D,所以不使用磁盘操作系统的机器不能使用此方法加密。

下面介绍具体操作步骤:

(1) 在需要加密的程序前加入一行 REM 语句。

例如:0 REM ABCDEFG

(2)CALL-151 进入监控状态

(3)键入以下数据:

806;0D 04 50 52 23 36 0D

(4)3D0G 返回 BASIC 状态

这个程序即加密完毕。它可以运行,也可以存盘,但如果要执行列表命令 LIST,机器不但不能列出程序,反而会去引导 DOS,原来的程序自然也被清除了。

2. 磁带 BASIC 程序的自动运行

使用过磁盘操作系统的用户都知道使用 RUN 命令可以从磁盘中调入一个 BASIC 程序并立即运行,而使用磁带存取程序却没有这种功能。但是,只要按以下方法操作,就能做出可自动运行的磁带 BASIC 程序。

(1)键入一个 BASIC 程序(或从磁盘、磁带上装入)。

(2)POKE 82,213

(3)SAVE 将程序写入磁带

等到写带完毕,一个可自动运行的程序便做好了。以后只要用 LOAD 命令把程序从磁带装入内存,它就会自动运行。

3. 特殊命令 GAME

中华学习机,有一条未公布的命令—GAME(有些厂家生产的中华学习机没有此命令),这条命令的作用是使主机的喇叭中发出一声短促的鸣叫,所以可将它作为一条发声语句写入程序中。但是,由于此命令触动了一些软开关,所以不能在中文状态下使用。

4. LOGO 语言的截取

中华学习机固化了 LOGO 语言子集,使用户用 LOGO 语言编写过程十分方便。但是,由于一些 APPLE 机没有与 CEC—LOGO 兼容的 LOGO 系统盘,所以使在 APPLE 机上调试 CEC—LOGO 语言过程的愿望成为泡影。但是,可以通过截取中华学习上的 LOGO 语言系统程序的方法来达到这个目的。具体作法为:

(1) MAXFILES1

(2) CALL-151

(3) C007;00 (将 I/O 空间切换到内部 ROM 空间)

(4) C600G (将 ROM 中的 LOGO 解释程序装入 RAM)

(5) 3D0G

(6) BSAVE LOGO,A\$1BF6,L\$7D8A

(7) 输入一段引导程序:

10 D\$=CHR\$(4)

20 PRINT D\$;"MAXFILES1"

30 PRINT D\$;"BRUN LOGO"

用 SAVE 命令将此程序存盘。

以后,在 APPLE 机上只要运行这张盘上的引导程序,便可以将 CEC—LOGO 解释程序装入内存,输入、调试或运行在中华学习机上编的 LOGO 过程(使用 CEC—LOGO 的 APPLE 机应有 64KB 的内存)。

有趣的对比

北京西北三条10号(100034)

中国 FORTH 应用研究会 丁志伟

91年第8期和第9期分别刊登过两个计算1000的阶乘的程序。一个是用6502机器语言编的,运行时间是5分8秒;一个是C语言编的,在286机上运行时间是15秒。这里再介绍两段阶乘程序,都是在中华学习机 CEC—I 上分别用 FORTH 语言和 BASIC 语言编写,以资比较。

FORTH 程序

```
0 VARIABLE KS 0 VARIABLE ZZH
CODE *1W/MOD (略) ENDCODE
CODE SUAN1W (略) ENDCODE
DEC (DECIMAL)
: XIAN DEC 20480 ZZH @
DO I @ DUP 1000 < IF 0. THEN
    DUP 100 < IF 0. THEN
        DUP 10 < IF 0. THEN
            -2 +LOOP ;
: N! 20480 DUP DUP ZZH ! KS ! ON
1001 I
DO 0 ZZH @ 1+ KS @
DO I @ J *1W/MOD ROT SUAN1W I !+
2 +LOOP ?DUP
IF 2 ZZH +! ZZH @ ! THEN
KS @ @ 0= IF 2 KS +! THEN
LOOP XIAN ;
```

计算1000!运行时间是13分58秒。其中 SUAN1W 和 *1W/MOD 是用汇编语言方法定义的 FORTH 动词(相当于语句),前者意思是把两数相加再除以10000,取余数和商,后者是把两数相乘再除以10000,取余数和商。

有些人对 FORTH 语言不太了解,其实若有一本合适的教科书做参考,读懂以上程序并不难。以下的 BASIC 程序是仿照上边程序的算法编写的。

```
10 DIM A(3000);A(0)=1;PRINT
20 W=1000000;
30 INPUT N;FOR I=1 TO N
40 FOR J=M TO K;A=I*A(J)+B;B=INT(A/W).
NEXT
50 IF B THEN K=K+1;A(K)=B;B=0
60 IF NOT A(M) THEN M=M+1
70 NEXT
80 PRINT A(K);;IF K=0 THEN RUN
100 FOR I=K-1 TO 0 STEP -1
110 IF A(I)<1000 THEN PRINT 0;
120 IF A(I)<100 THEN PRINT 0;
130 IF A(I)<10 THEN PRINT 0;
```


140 PRINT A(1);,NEXT,RUN

计算到400!耗时8分6秒,推算1000!要用1小时略多。若经过编译,速度加快,但也需时29分多。

下面对 BASIC、FORTH 和汇编做个比较,这都是 APPLE 机(或兼容机)运行的。

速度。汇编语言最快,BASIC 最慢。上边两段程序中 BASIC 循环次数只用 FORTH 循环次数的2/3,应该加快,再经编译也能提高速度,但仍不如 FORTH。FORTH 是一种非常重视效率的语言。

编程效率。汇编语言编写费时、调试麻烦已是人所共知。FORTH 和 BASIC 都是高级语言,编写、调试都方便得多。前边这段 BASIC 程序是在 FORTH 之后编的,编程比 FORTH 快,不过 BASIC 由于执行速度太慢,测试起来相当费时间。据笔者经验,程序规模越大,用 FORTH 编程效率越高。

改进情况。一般说来,一段程序被编出来之后,总有改进余地。汇编语言程序较长,改进起来也会相当麻烦。BASIC 语言编的这段程序是反复改进的结果,看来已无法再完善以提高速度了。FORTH 程序也经过几次改进,每次都使得程序更简单,执行速度更快。其中一种改进方法是把某些高级成分用汇编重新定义,能有效地提高速度。这段程序还有改进余地,由于没有实际用途,就没继续改进。

我们前后发表计算阶乘的文章共三篇,主要是为了比较各种计算机语言的性能(当然不是严格和全面的),也是想借此向读者介绍 FORTH。自91年第8期“谈谈 FORTH”刊出后,引起了广泛兴趣。有关 FORTH 的学习资料及软件咨询,请直接与丁志伟同志联系。本刊将陆续组织 FORTH 学习和应用方面的材料,欢迎投稿。编者

中华学习机汉化引导程序

北京中国农业电影制片厂(100081)张福森

CEC—I 中华学习机具有汉字系统,用中文作文件名,对于没有英文基础的使用者,将会是很方便的。

仿照 APPLE 机西文自动引导程序中的一种,参考有关资料,我编写了一个用中文作文件名的汉化自动引导程序。

把西文状态的自动引导程序汉化,关键在于屏幕字符的查询。本程序中20行的一段机器语言子程序,经机内入口地址为 C389 的 USR.DCOD 子程序,把从 \$07 单元读入的异形国标码转换为学习机内码后,送入 \$08 单元。300行至330行的 BASIC 子程序调用该段机器语言子程序,完成屏幕字符查询。其中300行处理 ASCII 码,310行、320行处理汉字。

针对中文方式下屏幕显示行数少的特点,30行至40行的一段机器语言修改了 DOS 中 CATALOG 命令的显示格式。令其显示适当行数后暂停,按回车键中止列目录,进入自动引导方便使用。程序中设置了两页屏幕显示说明文字。

本程序对 BASIC 文件和一般二进制文件操作效果良好。对于存储和显示与中文方式有冲突的二进制图像文件,应用 FID 程序中单文件拷贝功能将其拷贝到用本程序格式化的磁盘上,再把名字改为中文。切不可在中文方式下直接存盘。否则不能正常运行。此时应把程序中280行作如下改动,即可正常运行。

```
280 PRINT CHR$(14);PRINT CHR$(17);PRINT D$;B$
```

注意每个文件名应使用四个以上汉字,否则在列目录时会出现空行,致使自动定位寻找屏幕字符功能出错。以 CEC—I 中文目录为文件名,用本程序格式化

磁盘,在以后的使用中,你一定会感到它的方便实用。

```
10 HOME:DATA 165,7,32,137,195,133,8,96,32,12,
    253,201,14,1,208,3,76,44,174,76,60,174
20 FOR I=768 TO 775:READ A;POKE I,A;NEXT
30 FOR I=43453 TO 43465:READ A;POKE I,A;NEXT
40 POKE 44452,6;POKE 44601,76;POKE 44602,189;
    POKE 44603,169;POKE 44605,5
50 D$=CHR$(4);C$="CATALOG";PRINT D$;"PR
    #3";PRINT CHR$(18);PRINT CHR$(12)
60 HTAB 5;PRINT " * * * 本磁盘用中文作文件名 *
    * "
70 HTAB 5;PRINT "列目录时,第一幕列出七个文件名暂
    停。按【CR】键即可退出列目录,进入自动引导。按其它
    键继续列目录"
80 HTAB 5;PRINT "按文件名前的代表字符即可运行该文
    件。按数字键后,再按文件名前的代表字符,即可进行以
    下注明的功能操作。"
90 HTAB 22;PRINT "按任一键继续";PRINT CHR$(19);
    PRINT CHR$(12)
100 PRINT "1—LOAD(装载文件)";
    PRINT "2—LOCK(文件加锁)";
    PRINT "3—UNLOCK(文件解锁)";
110 PRINT "4—DELETE(删除文件)";
    PRINT "5—TEXT(退出本程序)"
120 VTAB 8;HTAB 17;PRINT "按任一键开始列目录";
    PRINT CHR$(19);PRINT CHR$(12)
130 PRINT D$;C$;PRINT CHR$(13)
140 T=0;H=4;FOR V=1 TO 7;GOSUB 300;IF F<=64
    AND F<>32 THEN VTAB V;HAB H;PRINT CHR
    $(219)CHR$(T+193)CHR$(221);T=T+1;S=V
```

如何在苹果机持续运行一程序

辽宁大学611#(110036) 张 毅

有时需要在计算机上连续不间断运行一程序,例如计算 π 的高精度解,常常需要几天甚至几个月的连续运算。然而操作人员和机器不能长时间不间断地工作,为了解决这个问题,我们可利用外存来帮助。

我们知道,APPLE I 及其兼容机上运行 BASIC 时,解释程序将 BASIC 产生的数据存放在 BASIC 程序后的存储器内,通过第零页的几个指针查找。在 Applesoft 解释程序工作时,第零页起到至关重要的作用。根据以上两点,我们只需将第0页及数据区储存到外存即起到保护 BASIC 工作环境的作用。具体作法如下:

1. 运行 BASIC 程序(在 DOS 环境下)。
2. 连续运算几小时后中断(用 Ctrl-C)。
3. CALL -151 进入监控,查看变量区首址指针(\$69, \$6A),自由区首址指针(\$6D, \$6E),字符串区首址指针(\$6F, \$70),HIMEM 值指针(\$73, \$74);记下这些指针数值,并计算各区长度(注意:当程序无新串产生时,不必查看字符串指针及 HIMEM 值)。

4. 将零页,数据区分别存入外存,关机。

这样下次开机引导 DOS 后,将程序、数据区、零页所对应的文件分别调入内存后,打入 CONT 便继续运行了。

例如有以下测试程序:

```
10 A$="This is ";B$="a test program!";J=10
20 C$=A$+B$
30 FOR I=1 TO 1000;FOR J=1 TO 1000;NEXT J
150 NEXT V
160 B$="RUN";K=PEEK(-16384);IF K<128 THEN
    FOR K=1 TO 5;NEXT K;K=FRE(0);GOTO 160
170 POKE -16368,0;K=K-176;IF K<1 OR K>5 THEN
    250
180 IF K=5 THEN SPEED=255;END
190 IF K=1 THEN B$="LOAD"
200 IF K=2 THEN B$="LOCK"
210 IF K=3 THEN B$="UNLOCK"
220 IF K=4 THEN B$="DELETE";PRINT CHR$(15)
230 A$="哪一个文件?";PRINT B$;A$
240 PRINT CHR$(7);PRINT CHR$(14);GET K$;K=
    ASC(K$)-48
250 IF K<17 OR K>T+16 THEN 160
260 H=2;V=S-T+K-16;GOSUB 300;IF F=66 AND
```

```
40 PRINT I,C$
50 NEXT I
)RUN
1 This is a test program!
.....
^C 中断,屏幕最后显示为:
22 This is a test program!
)CALL-151
* 69.74 假设这 n 个单元分别为:
变量区首址: $ 830;自由区首址: $ 845
字符串区首址: $ 95E0;HIMEM 值: $ 9600
变量区长度 = $ 845 - $ 830 = $ 15
字符串区长度 = $ 9600 - $ 95E0 = $ 20
(^C 返回 BASIC 状态)
)SAVE TEST PROGRAM
)BSAVE ZERO PAGE,A$0,L$FB
)BSAVE DATA,A$830,L$16
)BSAVE CDATA,A$95E0,L$20
此时关机,下次开机引导 DOS 后:
)LOAD TEST PROGRAM
)BLOAD DATA
)BLOAD CDATA
)BLOAD ZERO PAGE
)CONT
屏幕出现:
23 This is a test program!
:
(B$="RUN" OR B$="LOAD" THEN B$="B")+B$
270 N=0;FOR H=7 TO 34;GOSUB 300;N$=N$+C$;
    NEXT H;B$=B$+N$;PRINT CHR$(15);PRINT
    LEFT$(B$,34)
280 PRINT CHR$(14);PRINT D$;B$
290 CLEAR;GOTO 50
300 D=37376+2*(H-1)+68*(V-1);E=PEEK(D);F=
    PEEK(D+1);IF E<>127 THEN C$=CHR$(F);
    GOTO 330
310 N=N+1;F=F+128;POKE 7,F;CALL 768;W=PEEK
    (8);IF N=1 THEN C$=CHR$(127)+CHR$(W);
    GOGO 330
320 C$=CHR$(W);N=0
330 RETURN
```

ProDOS 磁盘操作系统入门(续)

廖 凯

2. 增加数据

APPEND 命令可以在一个顺序文件的末尾增加数据而不会破坏前面的数据。APPEND 命令首先打开文件,并将指针指到文件的末尾,然后发出 WRITE 命令。若增加资料到顺序文本文件(见上例)内,就将行号210和220合并为:

```
220 PRINT D$;"APPEND/DATA.DISK/EXAMPLE"
再次运行这程序并输入其它的数据,你在运行下面的程序(Read Sequential File)时,可以看到结果。
```

3. 读顺序文件

READ 命令可以使用 F# 参数,其一般形式是:

```
10 PRINT D$;"READ/卷目录/文件名,F#"
下面举例说明 READ 命令的用法,并将结果打印到屏幕上。
```

```
10 REM Read Sequential File
50 DIM N$(100),C$(100),P$(100)
100 D$=CHR$(4);PRINT D$;"PR#3".
120 PRINT D$;"OPEN/DATA.DISK/EXAMPLE"
130 PRINT D$;"READ/DATA.DISK/EXAMPLE"
140 ONERR GOTO 260
150 PRINT "NAME";TAB(25);"COMPANY";TAB(25);
    "PHONE";PRINT
160 I=I+1
170 INPUT N$(I);INPUT C$(I);INPUT P$(I)
180 X1=LEN(N$(I));X2=LEN(C$(I));X3=LEN(P$(I))
190 IF X1>26 THEN N$(I)=MID$(N$(I),1,25);X1=25
200 IF X2>26 THEN C$(I)=MID$(C$(I),1,25);X2=25
210 IF X3>21 THEN P$(I)=MID$(P$(I),1,20);X3=20
220 PRINT N$(I);TAB(29-X1);C$(I);TAB(32-X2);
    P$(I)
230 GOTO 160
240 PRINT D$;"CLOSE";END
260 POKE 216,0
270 GOTO 240
```

此程序首先打开文件 EXAMPLE 并准备读取,行号150在屏幕上显示三个标题。行号170的 INPUT 语句将从 EXAMPLE 中读取数据作为输入数据。行号180将各字串的长度存入相应的 X 变量内,行号190至210使各类数据在屏幕上以竖列(栏)的形式显示,并使各列向左对齐。行号140建立错误中断,转至行号260清除错

误并关闭文件。

① INPUT 与 GET 的区别:INPUT 和 GET 语句都可以从文件读取数据,但有所区别。INPUT 一次可以从文件读取若干个数据域。若一串字符中包含逗号或冒号,INPUT 语句将以此符号作为结束符,并放弃剩余的字符,而 GET 一次只从文件中读一个字符。GET 语句将不影响读取逗号和冒号。

② 读取任何文件:ProDOS 不支持 MON 和 NOMON 命令。因此,磁盘 I/O 信息不能反映到屏幕上,我们不知道写入文件内的是什么。我们可用下面的程序来解决这个问题。这程序既可在顺序文件内又可在随机文件内读取和列印每个字符。

```
100 REM RETRIEVE TEXT WITHOUT MON
110 D$=CHR$(4);PRINT D$;"PR#3"
120 INPUT "FILE NAME TO READ?";N$
130 PRINT D$;"OPEN"N$
140 PRINT D$;"READ"N$
150 ONERR GOTO 210
160 GET A$;PRINT A$;
180 GOTO 160
190 PRINT D$;"CLOSE";END
210 POKE 216,0
220 PRINT;PRINT"End of File"
230 GOTO 190
```

4. 读取其它文件类型

ProDOS 允许读取任何文件类型。一个有用的例子是读取一个卷目录或各种子目录。在一个程序内,用户会有机会看到各种文件。下面的程序将列印卷目录,并在屏幕上按四列格式列印文件名。另外,如果屏幕空间是有限的,那么你可以增加一个 IF/THEN 语句来终止显示。

```
90 REM Horizontal Catalog
100 V=5;I=0;D$=CHR$(4)
110 PRINT D$;"PR#3"
115 PRINT D$;"PREFIX"
120 INPUT "";P$
125 PRINT D$;"OPEN"P$,"TDIR"
130 PRINT D$;"READ"P$
135 ONERR GOTO 210
140 INPUT A$
145 IF A$="" THEN 140
150 IF MID$(A$,18,1)="D" THEN INVERSE
155 IF MID$(A$,1,12)="BLOCKS FREE;" THEN A$=""
160 IF MID$(A$,1,5)="NAME" THEN A$="" :GOTO
```

```

165 IF LEFT$(A$,1)="/" THEN B$=MID$(A$,1,
    15);GOTO 175
170 B$=MID$(A$,2,16)
175 H=(1*18);GOSUB 215
120 PRINT B$;NORMAL
185 I=I+1;IF I=4 THEN I=0;V=V+1
190 IF LEFT$(A$,1)="/" THEN V=V+1;I=0
195 GOTO 140
200 PRINT D$;"CLOSE";END
210 POKE 216,0;GOTO 200
215 POKE 1403,H;VTAB V;RETURN

```

行号115和120读取和设置当前磁盘的部首,将其值存入P\$内。行号125打开目录(P\$)。由于P\$不是文本文件,故必须指定文件类型(,TDIR)。行号140读取首个数据域或第一个文件名。每个文件名在起始位置保留一个空格,若文件上锁则留给上锁标记(*星号);若使用目录,则为/(斜线)所留。行号150至165检查文件的不同属性。若找到目录文件将以反相显示。行号170至190读取文件名并打印,设置显示窗口,记录要被打印到屏幕上的各个文件。

(二)、随机文件

随机文件的建立和检索方法与顺序文件相似,不同的是随机文件必须指明记录长度和记录号。

在使用随机文件时,要查明当前的部首,以便将数据写到正确的目录内。在文件被打开时,ProDOS查看文件名是否已存在。若文件名不存在,则建立一个,指定缓冲区空间将文件指针指到起始位置;若卷目录不存在,则发生错误并中断程序。

随机文件是以记录为单位进行数据读写。每个记录可以包含若干数据域。一个数据域最多可包含230个字符。

1. 记录长度

当一个随机文件首次被打开时,必须定义记录长度。在程序内用L参数指定一个记录的长度。如果没指定长度,那么ProDOS将预定长度为1。记录长度的取值范围在1~65535之间。其格式为:

```
10 PRINT D$;"OPEN/卷目录/文件名,L#"
```

不同的随机文件有不同的记录长度,但在同一个随机文件内,记录的长度都是一样的。

2. 写随机文件

在写随机文件时,必须指定记录号:

```
10 PRINT D$;"WRITE/卷目录/文件名,R#"
```

WRITE命令可以使用B#参数或F#参数。B#参数表示从该记录的第几个字节开始执行写操作,而F#参数则表示从当前位置向前移动若干个数据域,从此数据域开始执行写操作。

3. 读随机文件

在使用READ命令时必须指明记录号,象顺序文件一样,随机文件使用INPUT或GET语句从文件中读取数据。其一般格式为:

```
10 PRINT D$;"READ/卷目录/文件名,R#"
```

READ命令也可以使用F#参数或B#参数。当用B#参数时,你必须从0开始计数。

(三)建立一个16M字节的文件

ProDOS可以存取如16M字节一样大的文件。事实上你可以在一个143K字节的磁盘上建立一个16M字节文件。

```

10 D$=CHR$(4)
20 PRINT D$;"OPEN BIG,L256";PRINT D$;"WRITE
    BIG,R0"
25 PRINT 0
30 PRINT D$;"WRITE BIG,R65535"
35 PRINT "A"
40 PRINT D$;"WRITE BIG,R0"
50 PRINT 1
60 PRINT D$;"CLOSE"

```

这程序将在5英寸软磁盘上建立一个名为BIG的16M字节随机存取文件。如果你CATALOG此磁盘,则会看到如下显示:

```

/PRODOS
NAME TYPE BLOCKS ENDFILE SUBTYPE
BIG TXT 5 16776962 R=256

```

尽管当前只有两个记录(R0和R65535),ProDOS却分配了5块磁盘空间给此文件。只有一块是用于存储数据,其它块全为零,如果你要打印其数据,ProDOS则读取此文件并打印零,直到读取R65535为止。

(四)STORE和RESTORE

STORE和RESTORE命令是新增加的ProDOS命令。STORE命令允许你将存储器内的所有变量保存起来。相反,RESTORE命令是将被STORE保存的所有变量装入存储器内。使用STORE命令的不利因素是所有变量和数组都被保存而不管重要与否。

ProDOS以压缩格式放置变量于存储它们之前。用STORE存储数百个变量要比用顺序或随机文件快得多,但要占用较多的磁盘空间。STORE将变量放入类型为VAR的文件内。在用RESTORE时,所有在内存的变量被清除,并将新的变量装入内存。

STORE和RESTORE可以在程序开始时读取数据,而在程序结束时保存数据,大致与顺序文件一样。你还可以用STORE和RESTORE增加数据。这过程需要程序将数据存入一个数组内。

```

5 REM Store/Restore
10 D$=CHR$(4)
20 DIM N$(100),C$(100),P$(100)
25 HOME
50 PRINE "1. Store, Write to Disk"
60 PRINE "2. Restore, Read from Disk"
65 PRINE "3. END"
70 GET A$;B=VAL(A$);IF B>0 AND B<4 THEN 80
75 GOTO 70
80 ON B GOTO 130,230,300
130 A=A+1
140 INPUT "Name:";N$(A)
150 INPUT "Company";C$(A)

```

汽车大赛

浙江湖州市新风小学 马宇昊

我在暑假里编了一个电脑游戏,取名“汽车大赛”。运行后,出现跑道,你驾驶的汽车在跑道下端。你必须用 J、L 和 K 键控制汽车左行、右行和停下,以逃避迎面奔来的汽车。随着道路变得越来越窄,难度也越来越高。当汽车被撞或驶出跑道,则游戏结束。屏障下端的数字是你的分数,如超过十万分,则说明你是高水平了!

```
10 C=17;D=35
20 GR; COLOR=1
30 W=10; B=20
40 COLOR=3;VLIN 0,39 AT 15
50 COLOR=2
60 VLIN 0,39 AT W; VLIN 0,39 AT B
70 COLOR=10; PLOT C,D
80 E=INT(RND(1)*19+1)
90 IF E=W OR E<W OR E=15 OR E=B OR E>B THEN
80
```

```
100 FOR F=0 TO 39
110 COLOR=15;PLOT E,F
120 P=PEEK(-16384)
130 IF P=204 THEN COLOR=0;PLOT C,D;C=C+1
140 IF C=15 THEN C=16
150 IF C=W THEN 300
160 IF C=B THEN 300
170 IF P=202 THEN COLOR=0;PLOT C,D;C=C-1
180 IF C=15 THEN C=14
190 IF C=W THEN 300
200 COLOR=0;PLOT E,F
210 COLOR=10;PLOT C,D;IF P=202 OR P=204 THEN
COLOR=0;PLOT C,D
220 V=V+100;VTAB 24;HTAB 14;PRINT V
230 IF F=35 THEN 260
240 IF F=39 THEN 280
250 NEXT F
260 IF C=E THEN 300
270 NEXT F
280 IF W=13 OR B=17 THEN GOTO 50
290 COLOR=0;VLIN 0,39 AT W;VLIN 0,39 AT B;W=W+
1;B=B-1;GOTO 50
300 TEXT;HOME;VTAB 11;HTAB 19;PRINT "End"
```

电视音像系统的佳音 ——CZZ 彩色台标时码发生器

超大规模集成电路的采用使得本彩色台标时码发生器具有:

1. 多种调节功能;
2. 简便的操作方法;
3. 清晰稳定的彩色字符;
4. 成倍压缩的体积和重量;
5. 极高的性能价格比。

它成为电视台、闭路电视系统、音像出版系统、公用天线系统不可缺少的设备之一。它为用户随机的提供各种彩色台标和时码,从而保护您的节目源不受侵犯。

由于本机采用超大规模集成电路及单片机控制,故整机故障率几乎为零。本机售后一年内如发生质量故障,一律采取整机以坏换新的办法,以保障信誉。外地用户来京取货免费赠送3盘录像带。

(上接20页)

```
160 INPUT "Phone";P$(A)
170 INPUT "Enter another Name?";A$
180 IF A$="Y" OR A$="y" THEN HOME;GOTO 130
190 IF A$="N" OR A$="n" THEN 210
200 GOTO 170
210 PRINT D$;"STORE/DATA. DISK/EXAMPLE"
220 GOTO 50
230 PRINT D$;"RESTORE/DATA. DISK/EXAMPLE"
```

本机与国内同类产品比较(外型见封面)

	CZZ-1	国内同类产品
显示功能	正反两种显示	只能正显示
台标时码	位置满屏任意移动	不可移动
彩色锁相	移定	爬行严重
时码误差	每日<1秒	<10秒
集成度	大规模集成电路单片机控制	分立芯片组合

本机每台售价4800/元

还可设计几种台标图案供换用,每增加一种图案,收费500—800元,视图案复杂程度而定。

园恩寺电子实验厂

邮编100009

地址:北京市东城区后园恩寺甲20号

联系人:刘宜秋、王丽燕

电话:4016359 6853592

开户银行:北京工商银行王府井分理处

帐号:402052—60

北京站换乘104、108、113路车交道口下车

```
240 FOR I=1 TO A
250 PRINT N$(I);PRINT C$(I);PRINT P$(I)
260 NEXT
270 PRINT;INPUT"PRESS RETURN TO CONTINUE";A$
280 GOTO 25
300 END
```

(未完待续)

本刊不慎,遗失1121991003号记者证,声明作废

第三章 6502 MPU 的寻址方式

南京大学大气科学系(210008) 朱国江

一、什么是寻址方式

在第一章中,我们曾指出控制计算机各部件协调动作的命令称为指令。每条指令均由两部分组成:一部分称为操作码,它指明该条指令的操作性质,执行什么样的操作(即指令功能);另一部分是操作数,实际上是操作的对象,它指明参与操作或运算的操作数据本身,或者操作数据所存放的单元地址,或者操作数据所在地址的地址。

数据和指令在存储器中存放的位置,称为地址。地址又分为两种:指令地址,即为存放指令的地址;操作数地址,为存放数据的地址,简称为操作地址。

简单地说,寻址方式就是寻找指令地址或操作数地址的方式。

二、6502 指令系统的特点

6502 微处理器共有 56 条基本指令,13 种寻址方式,其主要特点有以下几点:

- 虽然 6502 内部可供访问的寄存器较少,但它的指令系统包含零页寻址方式,可访问 \$0000 - \$00FF256 个地址,零页寻址方式的指令占用字节少,执行时间短,对节约内存、缩短程序执行时间都是有利的。

- 6502 的指令长度为 1 至 3 个字节,第一字节一律是操作码,它决定 6502 完成一种运算或操作,而操作码后面跟着的是操作数或操作数地址,占用一个或二个字节,字节多少视寻址方式而定。

- 6502 和外设交换数据或信息采用存储器映象方式,凡是访问存储器的指令,都可以用于外设,这样访问外设的指令多,给使用带来极大的方便。

三、13 种寻址方式

1. 立即寻址(Immediate Addressing)

立即寻址方式,指的是操作对象部分直接给出了操作数本身,该操作数称为立即数。

例如,指令代码 A9 5A,是一条两字节指令,用汇编语言写出就是 LDA # \$5A,其中 A9 是操作码,表示取一个数装入累加器 A 中;累加器 A 中的值,就是 A9 后面的 5A,这是一个立即数而不是地址码。这条指令的功能是将立即数 \$5A 送累加器 A,即 \$5A→A。

说明:

- 立即寻址方式的指令,常用来设置程序所需用的各种固定初值。

- 立即寻址方式指令,长度均为两个字节。

- # 符号表示后面跟着的是立即数,\$ 是 16 进制数的表示符号。

- 操作数部分不是地址,而是操作数本身。

- 立即数的形式是 # \$nn,其中 nn 是两位 16 进制数,所以立即数不超过 8 位二进制数范围(\$FF)。

- 立即寻址方式指令中,用于取数至累加器 A 的有 LDA # \$nn;取数至变址寄存器的有 LDX # \$nn, LDY # \$nn;累加器 A 与立即数带进位加、减的有 ADC # \$nn, SBC # \$nn;与累加器 A 的值进行逻辑运算的有 AND # \$nn, EOR # \$nn, ORA # \$nn;与寄存器比较的有 CMP # \$nn, CPX # \$nn, CPY # \$nn 等共 11 条指令。

2. 绝对寻址(Absolute Addressing)

绝对寻址方式中,操作数部分是操作数在存储器中的实际地址(称为绝对地址,又称直接地址)。

例如,采用绝对寻址方式的取数指令 LDA \$4000,是把内存中地址为 \$4000 单元的内容装入累加器 A 中,即(\$4000)→A。这里 \$4000 表示绝对地址,(\$4000)表示 \$4000 单元存储的内容(以下表示法同)。

又如,指令:20 DA FD (JSR \$FDFA)通知 CPU 转入 \$FDFA 地址单元取指令执行。

注意 LDA # \$5A 和 LDA \$4000 两条指令的异同点:相同点是它们都是向累加器 A 中置数,所以它们的汇编符号都是 LDA(LOAD A 的缩写);但它们的操作码不同,前者是 A9,后者是 AD。操作码不同表示寻址方式不同,前者是立即数寻址,后者是绝对寻址。这个简单例子可以加深对寻址方式这一概念的理解。

说明:

- 绝对寻址方式的指令,其长度都是三个字节。

- 绝对寻址方式用两个字节存放操作数的地址,它们的寻址范围可以是整个 64K 存储器的任何一个地址,使用方便,缺点是每条指令占据三个字节。

- 凡用两个字节表示一个地址时,总是先给出低位字节,再给出高位字节,即高位在后,低位在前。绝对寻址方式指令表示成机器码时,就属这种情况。

- 绝对寻址指令有:内存单元取数至寄存器的 LDA \$nHnL, LDX \$nHnL, LDY \$nHnL;寄存器中的数存入内存的 STA \$nHnL, STX \$nHnL, STY \$nHnL;无条件转移 JMP \$nHnL;转移到子程序 JSR \$nHnL;存储器内容加、减 1 的 INC \$nHnL, DEC \$nHnL;累加器与存储器带进位加、减的 ADC \$nHnL, SBC \$nHnL;逻辑运算的 AND \$nHnL, EOR \$nHnL, ORA \$nHnL;检测存储器的 BIT \$nHnL;算术左、右移位的 ASL \$nHnL, LSR \$nHnL;循环左、右移位的 ROL \$nHnL, ROR \$nHnL;寄存器与存储器比较的 CMP \$nHnL, CPX \$nHnL, CPY \$nHnL 等共 23

条。

3. 零页寻址(Zero-page Addressing)

零页寻址方式表达的操作数,实际上是一个操作数地址,并仅限于存储器的零页范围 0000—00FF 内。

例如,取数指令 LDA \$FB,它的意思是吧零页地址 FB 单元中的操作数送到累加器 A。这里的操作数是地址,而且只有一个字节 FB,表示它是零页地址 00FB。其指令的机器码为 A5 FB,功能为(\$FB)→A。

指令 A5 FB (LDA \$FB)和指令 AD FB 00(LDA \$00FB)在功能上是等同的。前者系零页寻址,后者则为绝对寻址,但前者比后者少用一个字节。

零页寻址和绝对寻址的操作对象部分都是操作数所在地址,不同的是前者只能在零页的 256 个单元中,而后者可以是整个 64K 存储单元中的任一个地址。

说明:

- 零页寻址均为二字节指令,写成机器码时第一个字节是操作码,第二个字节为操作数的零页地址。

- 零页地址中操作数地址仅限于零页范围(0000—00FF),故在表示操作数地址码时,不必给出高 8 位地址(约定为 \$00),因而指令长度可以减少一个字节。

- 零页寻址方式中的操作数表达式是 \$nn,其中 nn 代表两位十六进制数。

- 零页寻址指令占用字节少,执行速度快,在节省内存和缩短时间上,为程序编制提供了方便。

- 零页寻址方式的指令有 20 条,它们是:内存单元取数至寄存器 LDA \$nn,LDX \$nn,LDY \$nn;寄存器中数存入内存 STA \$nn,STX \$nn,STY \$nn;算术运算指令 ADC \$nn,SBC \$nn;增减指令 DEC \$nn,INC \$nn;逻辑运算指令 AND \$nn,BIT \$nn,EOR \$nn;ORA \$nn;移位指令 ASL \$nn,LSR \$nn,ROL \$nn,ROR \$nn;比较指令 CMP \$nn,CPX \$nn,CPY \$nn。

4. 隐含寻址(Implied Addressing)

所谓隐含寻址,指的是地址已为操作码隐含,不需再指出,即操作数所在地址隐含于操作码之中。

例如,指令 CLC,其机器码为 18,功能是置标志寄存器中的 C 标志位为 0(清进位标志)。清除进位位显然是对 CPU 中的状态寄存器进位位 C 进行操作,因为操作码(\$18)本身已包含了操作数的内容,所以,完全没有必要在指令中再用一个字节来表示操作数。

又如,DEX 指令,它执行变址寄存器 X 中的内容减 1 操作,然后再存入 X 中,即 X-1→X,操作码为 \$CA。INX 指令则执行变址寄存器 X 加 1 操作,即 X+1→X,操作码为 \$E8。这里 \$CA 和 \$E8 的操作码中,分别隐含了对变址寄存器 X 进行操作(减或加)的信息,同样不需要选用另外字节来存储操作数。

说明:

- 隐含寻址方式的指令,均为单字节指令,因操作数地址隐含在操作码中。

- 隐含寻址的操作数是在寄存器 X、Y、S 及 P 中的各标志位。

- 隐含寻址的指令有:进出栈操作 PLA,PHA,PLP,PHP;各寄存器之间数据传送 TAX,TXA,TAY,TYA,TSX,TXS;中断返回 RTI;子程序返回 RTS;设置进位标志 SEC;设置十进制运算方式 SED;设置中断禁止状态 SEL;清除进位标志 CLC;清除十进制运算标志 CLD;清除中断禁止状态 CLI;清除溢出标志 CLV;强迫中断 BRK;空操作 NOP;变址寄存器加、减操作 DEX,INX,DEY,INY 等共 25 条。

5. 累加器寻址(Accumulator Addressing)

累加器寻址方式指的是:操作数在累加器 A 中。

例如,循环左移指令 ROL A,其功能为:将 A 中的操作数的各位连同进位 C 在内循环左移一位,最高位移到标志位 C 中。机器码为 2A,助记符为 ROL。

说明:

- 累加器寻址皆为单字节指令。

- 由于操作仅涉及累加器 A,因此,操作数地址不需要单独表示,只要隐含在操作码字节中即可。从这个角度上说,累加器寻址也是一种隐含寻址。

- 累加器寻址主要用于移位,指令只有 4 条,即:ASL A,LSR A,ROL A,ROR A。

6. 绝对 X 变址(Absolute X-Indexed Addressing)

这是使用 X 寄存器进行变址的一种寻址方式。操作数地址是由绝对地址加上 X 寄存器中的值构成。

例如,指令:BD 00 40 (LDA \$4000,X)是一条取数指令,其操作码为 BD,操作数存放在一个内存单元,其地址由指令的第二字节和第三字节指示的地址(本例是 \$4000)加上变址寄存器 X 的内容得到。

我们将指令给出的 16 位的绝对地址称为基地址,而寄存器 X 中的内容称为偏移量,则:基地址+偏移量(X)=操作数实际存放的有效地址。

因此,上述指令的功能就是把 4000+X 单元中的值取出来送到累加器 A 中。如果 X 变址寄存器中的值是 \$20,那么这条指令就是把 \$4020 中的数送入累加器 A 中。

又如取数指令 STA \$0330,X,其功能是把累加器 A 中的内容存放到存储器 \$0330+X 单元中去,即 A→(\$0330+X)。

说明:

- 绝对 X 变址方式的指令皆为三字节指令。

- 同一条绝对 X 变址指令,因 X 中的值不同,而改变了操作数的地址。这就是把 X 寄存器称为变址器的原因。由于基地址加 X 偏移量才是操作数的有效地址,所以,有效地址是经过上述“变址”过程得来的。

- 绝对 X 变址指令共有 15 条,其中取数指令两条:LDA \$nHnL,X,LDY \$nHnL,X;存数指令一条:STA \$nHnL,X;算术逻辑运算指令 12 条:ADC \$nHnL,X,DEC \$nHnL,X,INC \$nHnL,X,SBC \$nHnL,X,AND \$nHnL,X,EOR \$nHnL,X,ORA \$nHnL,X,ASL \$nHnL,X,LSR \$nHnL,X,ROL \$nHnL,X,ROR \$nHnL,X,CMP \$nHnL,X。

7. 绝对 Y 变址(Absolute Y-Indexed Addressing)

它是使用 Y 寄存器进行变址的一种寻址方式。这种寻址方式把指令给出的 16 位地址作为基地址与偏移量(Y 变址寄存器中的值)相加后所得的地址,作为操作数实际存放的有效地址。

例如,传送指令 LDX \$FE30,Y(设 Y 寄存器中的值为 \$20),其功能是把 \$FE30+Y,即 \$FE50 单元中的内容取出来送入寄存器 X 中,即(\$FE30+Y)→X。

说明:

- 绝对 Y 变址方式的指令皆为三个字节。
- 绝对 Y 变址和绝对 X 变址十分相似,其区别前者使用的是 Y 变址器,后者使用的 X 变址器。
- 绝对变址寻址(X 或 Y)与零页变址寻址(X 或 Y)很类似,其区别仅在于零页变址中只给出一个 8 位地址(低 8 位),其高 8 位约定为 \$00。

• 绝对 Y 变址的指令有:LDA \$nHnL, Y、LDX \$nHnL, Y、STA \$nHnL, Y、ADC \$nHnL, Y、SBC \$nHnL, Y、AND \$nHnL, Y、EOR \$nHnL, Y、ORA \$nHnL, Y、CMP \$nHnL, Y 等共 9 条。

8. 零页 X 变址(Zero—page X—Indexed Addressing)

使用 X 变址寄存器的零页变址,简称零页 X 变址。将零页地址作为基地址与 X 寄存器的内容相加,即得操作数存放的实际零页地址。

例如,取数指令 LDA \$A3,X,其功能是把存储器 \$00A3+X 单元中的内容取出,送累加器 A。

又如,LDA \$3F,X 与 LDA \$003F,X,这两条指令表达的含意相同,功能一样。但使用上有点区别:前者操作码为 B5,后者操作码为 BD;前者为零页 X 变址,后者为绝对 X 变址;前者为两个字节,后者为 3 个字节。

说明:

- 零页 X 变址指令皆为两个字节长度。
- 绝对变址寻址(X 或 Y)和零页变址寻址(X 或 Y)含义相近,但寻址方式不同,前者寻址范围广,可以是任意地址;后者仅限于零页地址。

• 零页 X 变址只能产生零页地址,例如,指令 LDA \$A3,X,若 X=\$62,则操作数的有效地址不是 \$105,而是 \$05。这一点在使用时应特别注意。

• 零页 X 变址寻址指令有 16 条:LDA \$nn,X、LDY \$nn,X、STA \$nn,X、STY \$nn,X、ADC \$nn,X、DEC \$nn,X、INC \$nn,X、SBC \$nn,X、AND \$nn,X、EOR \$nn,X、ORA \$nn,X、ASL \$nn,X、LSR \$nn,X、ROL \$nn,X、ROR \$nn,X、CMP \$nn,X。

9. 零页 Y 变址(Zero—page Y—Indexed Addressing)

使用 Y 变址寄存器的零页变址,简称零页 Y 变址。它与零页 X 变址类似,只是变址器取自 Y 的当前值。

例如,指令 STX \$5F,Y,表示将一个数送至变址寄存器 X,而这个数存放在变址器 Y 的值加上 \$5F 之后所指示的零页存储单元中。

说明:

- 零页 Y 变址指令是二字节的。

• 采用零页 Y 变址的指令只有两条:LDX \$nn,Y 和 STX \$nn,Y。

• 零页 Y 变址的其它意义同零页 X 变址有关说明。

10. 间接寻址(Indirect Addressing)

间接寻址指令的操作对象部分给出的是操作数所在地址的地址(称为间接地址)。这意味着,在操作码后面的两个字节并不是有效地址,而是存放有效地址低 8 位的存储单元(间接地址)。亦即操作数的有效地址必须通过两次寻址才能找到。

例如,无条件转移指令 JMP(\$2000),首先从指令指出的间接地址 \$2000 单元取出操作数有效地址的低 8 位(设为 \$5F),从间接地址加 1 单元(即 \$2001 单元)取出有效地址的高 8 位(设为 \$4D),则 \$4D5F 即为操作数的有效地址;然后将 \$4D5F 送程序计数器 PC,CPU 随即转向 PC 所指示的单元(\$4D5F)去执行指令。

注意 JMP \$2000 和 JMP(\$2000)这两条指令的差别:两者均为无条件转移指令,但操作码不同,前者是 4C,后者是 6C,因而寻址方式各异。前者是绝对寻址,有点像 BASIC 中的 GOTO 2000,后者是间接寻址,像 BASIC 中的 GOTO 2000,而 2000 行语句是 GOTO 4D5F。当然这是形象化的比喻,GOTO 语句后面的行号,相当于我们举例中的内存地址。

间接寻址都是 3 字节指令,而且仅有 1 条:JMP(\$nHnL)。

11. 相对寻址(Relative Addressing)

相对寻址的指令格式是:首字节为操作码,次字节为相对转移地址,又称相对位移量,或偏移量 D,它实际上是条件转移指令的跳转步长。D 值可正可负,符号由最高位指明(1 为负,0 为正)。操作数的实际地址是由取出本指令后,下一条指令的操作码所在的地址加偏移量 D 来得到。即取出本指令后的 PC 值加 D。

为了弄清相对寻址的概念,我们不妨先看一个机器语言程序实例:

```
0300—AD 00 20   LDA  $2000
0303—D0 05     BNE  $030A
0305—A9 20     LDA  # $20
0307—8D 00 40   STA  $4000
030A—00       BRK
```

第一条指令采用绝对寻址方式,把 \$2000 单元的内容送累加器 A 中。第三条指令采用立即寻址,把立即数 \$20 送累加器 A。它们都是取数指令。第四条指令采用绝对寻址方式,把累加器 A 的内容送入 \$4000 单元,是存数指令。最后一条指令是单字节指令;显然是隐含寻址方式,产生中断,它把标志位寄存器 P 中的 B 标志位(第 4 位)置 1。第二条指令 BNE,是一条条件转移指令,条件是非 0,其条件成立与否取决于上一条指令执行的结果。当第一条指令从 \$2000 单元取出的数为 0 时(Z=1),条件不满足,不产生转移动作,程序接着执行第三条指令;而当从 \$2000 取出一个非 0 值

(Z=0)时,条件满足,程序转移到第五条指令。

如何判别这种转移?转移地址又是如何计算出来?这就要分析指令计数器 PC 的作用。

上述程序执行时,PC 中首先置入的是首地址 \$0300,于是计算机依 PC 的值执行 \$0300 中的指令,同时将该指令的长度(字节数 3)与 PC 的值相加(\$0300+\$03),再送入 PC,此时 PC 值为 \$0303,正好是第二条指令的首地址。第一条指令执行完后,依 PC 中的值去执行 \$0303 单元开始的第二条指令,同时把第二条指令长度(二字节)加入 PC 中,使 PC 变成 \$0305。第二条指令要求检查第一条指令执行后的 Z 标志位状态(\$2000 中的数为 0 时,Z=1;\$2000 中的数非 0 时,Z=0)。当 Z=1 时,程序不产生转移,按正常顺序执行下一条指令(即 \$0305 开始的第三条指令);当 Z=0 时,非 0 条件满足,程序产生转移,于是将 PC 中的值(\$0305)加上偏移量 D(本例是 \$05)置入 PC 中,变成 \$030A,因此,执行 \$0303 的第二条指令后,直接转向 \$030A 执行第五条指令。

由此可知,条件转移指令 BNE \$030A 的意思是根据上一条指令执行后零标志位 Z 的状态,决定程序是否分支,若 Z=0,则分支转向 \$030A 去执行;反之不产生分支,执行 \$0305 开始的指令。

对本例来说,若产生分支转移,则偏移量 D(即跳转步长)应从 \$0305(本指令 BNE 取出后的 PC 值)算起,到 \$030A 为止,即 $D = \$030A - (PC + 2) = \$030A - (\$0303 + \$02) = \$05$ 。或者由 $\$0303 + \$02 + D = \$030A$ 中算出 D。后一种方法物理意义明确且容易记忆,即:

转移指令的首地址+2(转移指令都是两字节指令)+D(偏移量)=目标地址(操作数的实际地址)

偏移量 D 也可取负值,请看下列实例:

0300—CE	00	20	DEC	\$2000
0303—AD	00	20	LDA	\$2000
0306—D0	F8		BNE	\$0300
0308—A9	20		LDA	# \$20
030A—8D	00	40	STA	\$4000
030D—60			RTS	

本程序,条件转移指令 BNE 的首字节(操作码 D0)存放在 \$0306 单元,偏移量 D 存放在 \$0307 单元。该指令的意思是若 Z=1(非 0 条件不满足),则执行 \$0308 单元处的指令;若 Z=0(非 0 条件满足)跳转至 \$0300 处执行。显然这里偏移量 D 是负值,由上述计算方法有:

$$\$0306 + \$02 + D = \$0300$$

$$\therefore D = \$0300 - \$0308 = -\$08$$

D 为负值,应用补码表示,即 F8。(注:负数的补码可以通过对原码求反加 1 得到)。

说明:

- 相对寻址指令皆为两字节。
- 相对寻址只适用于条件转移指令。
- 操作数部分是条件转移指令的跳转步长。
- 操作数的实际地址是取出本指令后 PC 值(本

指令首地址+2)与偏移量之和。

- 偏移量为负时,要用补码表示。
- 为减少实际编程时的麻烦,书写相对寻址指令的助记符时,用转移后的目标地址代替偏移量。
- 相对寻址指令有: BCC、BCS、BEQ、BMI、BNE、BPL、BVC、BVS 等 8 条。

12. 先变址(X)间接寻址((IND,X))

这种寻址方式实际上是零页 X 变址和间接寻址两种方式的结合。首先,它对零页基地址 IND 进行 X 变址,求得零页地址 IND+X;再以 IND+X 作为间接地址,经两次间接寻址找到操作数的有效地址;第一次对 IND+X 间址得到操作数有效地址的低 8 位;第二次对 IND+X+1 间址得到操作数有效地址的高 8 位。

例如,取数指令 LDA(\$06,X),其机器码为 A1 06。若已知变址寄存器 X 中的值为 02,存储单元(08)=65,(09)=87,(8765)=FF,零页基地址 IND=06,则根据上述定义,先计算(IND+X)=(06+02)=(08)=65,这是该指令操作数第一次间址,得到其有效地址的低 8 位;第二次间址(IND+X+1)=(06+02+1)=(09)=87,得到有效地址的高 8 位,所以操作数实际地址是 \$8765。而本指令的功能则是把 \$8765 地址单元中的数 FF 取出来送到累加器 A 中。本例可以归结为下述程式:

$$\text{LDA}(\$06,X) \Rightarrow (06+X) = (08) = 65 \rightarrow \text{PC}_L \Rightarrow (06+X+1) = (09) = 87 \rightarrow \text{PC}_H \rightarrow \text{PC} \rightarrow 8765 \Rightarrow (8765) = \text{FF} \rightarrow \text{A}$$

说明:

- 先变址(X)间接寻址指令皆为两字节指令。
- 变址寄存器只能用 X。
- 先做 X 零页变址,再两次间接寻址,才能得到操作数的有效地址。
- 指令有:取送数 LDA(\$nn,X)、STA(\$nn,X)、算术运算 ADC(\$nn,X)、SBC(\$nn,X)、逻辑运算 AND(\$nn,X)、EOR(\$nn,X)、ORA(\$nn,X)及比较运算 CMP(\$nn,X)等 8 条。

13. 后变址(Y)间接寻址((IND),Y)

后变址(Y)间接寻址,是先间接寻址,后 Y 变址。它是间接寻址和绝对 Y 变址相结合的一种寻址方式。其步骤是先在零页间接寻址,取得一个 16 位基地址,然后再进行绝对 Y 变址寻址,即以基地址加 Y 变址寄存器的内容得到操作数实际存放的有效地址。所以,这种寻址方式应称为先间接寻址后绝对 Y 变址。

例如,取数指令 LDA(\$06),Y,机器码为 B1 06,设(06)=65,(07)=87,(Y)=02,(8765)=EE。第一步间接寻址,因(06)=65,(07)=87,所以间接寻址后的地址为 \$8765;第二步绝对 Y 变址,因(Y)=02,那么 Y 变址后为 \$8767;第三步将内存单元 \$8767 中的数 EE 取出来送累加器 A。

说明:

- 后变址(Y)间接寻址为两字节指令。
- 变址寄存器只限于 Y 寄存器。

(下转第 31 页)

数据结构

北京电脑天地学校(100051) 宋丹颖

数据结构自测题

一、从供选择的答案中选出同下列叙述关系最密切的字句,把编号写在题目的对应题号旁。

- A. 说明数据元素之间的顺序关系,抽象地反映数据元素的结构。
- B. 各数据元素之间存在着线性关系,具有均匀性和有序性。
- C. 在进行插入或删除操作时,只需改变结点的指针,不必移动表中别的元素。
- D. 经常用于表达式的转换和求值、处理子程序调用等的数据结构。
- E. 通常采用顺序存储结构,每个数据元素由一个值和一组下标组成。
- F. 由一个或多个数据项组成,各数据项的数据类型可不同。
- G. 求出字符串中含有字符个数的操作。
- H. 求出字符串中从指定位置开始且长度为一定值的串。

供选择的答案:

- ①数据结构 ②数据的存储结构 ③数据的逻辑结构
- ④栈 ⑤队列 ⑥线性表
- ⑦链表 ⑧记录 ⑨数据类型
- ⑩数据对象 ⑪数组 ⑫联接
- ⑬求长度 ⑭定位 ⑮求子串

二、从供选择的答案中选出正确答案,将其编号填写在叙述中的横线上。

1. 栈和队列是两种基本的数据结构,栈是具有_____特点的特殊线性表,而队列是具有_____特点的特殊线性表。

供选择的答案:

- A. 只进不出 B. 先进先出
- C. 后进先出 D. 随机进出

2. 使用双向链表存储数据,其优点是可以_____。

供选择的答案:

- A. 提高检索速度 B. 很方便地插入和删除数据
- C. 节约存储空间 D. 很快回收存储空间

3. 设有一个栈,元素进栈的次序为1,2,3,4,5。能得到下列出栈序列中的_____。

供选择的答案:

- A. 1,2,3,4,5 B. 2,3,4,5,1
- C. 5,1,2,3,4 D. 5,4,2,3,1

4. 排序是数据处理中重要的运算,它的功能是将一个数据元素的无序序列按某种指定的顺序调整成为一个

有序序列。各种排序方法可按不同的原则进行分类。冒泡排序和快速排序属于_____。

供选择的答案:

- A. 插入排序 B. 选择排序
- C. 交换排序 D. 归并排序

三、从供选择的答案中选出应填入下列叙述中_____内的正确答案。

1. 在作进栈运算时,应先判别栈是否[A];在作退栈运算时,应先判别栈是否[B]。当栈中元素为n个,作进栈运算时发生上溢,则说明该栈的最大容量为[C]。

为了增加内存空间利用率和减少溢出的可能性,由两个栈共享一片连续的内存空间时,应将两栈的[D]分别设在这片内存空间的两端,这样,当[E]时,才产生上溢。

供选择的答案:

- A, B: ①空 ②满 ③上溢 ④下溢
- C: ①n-1 ②n ③n+1 ④n/2
- D: ①长度 ②深度 ③栈顶 ④栈底

- E: ①两个栈的栈顶同时到达栈空间的中心点
- ②其中一个栈的栈顶到达栈空间的中心点
- ③两个栈的栈顶在栈空间的某一位置相遇
- ④两个栈均不空,且一个栈的栈顶到达另一个栈的栈底

2. 线性表中的[A]的主要优点是从表中任一结点出发都能访问到所有结点。使用[B],可根据需要在前后两个方向上方便地移动,而使用[C]则只能顺链循一个方向移动。若有几个线性表同时并存,且在处理过程中各表的长度动态地变化,元素的总数也动态地改变,这种情况下应选用[D]存储结构。供选择的答案:

- A, B, C: ①单向链表 ②可利用空间表 ③双向链表
- ④循环链表
- D: ①顺序 ②链接

3. 对大小为n的队列,随着元素的进队和出队,其首、尾指针将随之发生变化。当队列的队首指针值等于队尾指针值时,则表示队列处于[A]状态;当队尾指针=n时,表示队列处于[B]状态。队列是限定在[C]进行操作的具有[D]性质的特殊线性表。

供选择的答案:

- A, B: ①进队 ②出队 ③队满 ④队空
- C, D: ①一端 ②两端 ③先进先出 ④随机进出

4. 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]。

供选择的答案:

- 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: ① n ② $n/2$ ③ n^2

5. 有一个二维数组 A,行下标的范围是 1 到 9,列下标的范围是 1 到 5,每个数组元素用相邻的 4 个字节存储。存储器按字节编址。假设存储数组元素 A[1,1]的第一个字节的地址是 0。存储数组 A 的最后一个元素的第一个字节的地址是[A]。若按行存储,则 A[4,5]和 A[6,3]的第一个字节的地址分别是[B]和[C]。若按列存储,则 A[8,1]和 A[3,4]的第一个字节的地址分别是[D]和[E]。

供选择的答案:

- A—E: ①28 ②44 ③76 ④92 ⑤108
⑥116 ⑦132 ⑧176 ⑨184 ⑩188

四、从下列叙述中选出正确的叙述,把编号写在横线内。

1. 顺序存储方式只能用于存储线性结构。
2. 顺序存储方式的优点是存储密度大,且插入、删除运算效率高。
3. 链表的每个结点中都恰好包含一个指针。
4. 栈和队列的存储方式既可以是顺序方式,也可以是链接方式。
5. 数组和记录都是只能存储相同类型数据元素序列的数据结构。
6. 栈和队列都是一端固定,只能在另一端进行存取操作的特殊的线性表。
7. 由于数组中的数据元素的存储位置是下标的线性函数,所以可对其中的元素进行随机访问。
8. 采用紧缩格式存储字符串,可以节省存储空间,但在进行串的运算时需要多花费时间去分离同一个字中的字符。
9. 对以顺序方式存储的栈和队列,必须预先分配存储空间。而对于以链表形式存储的栈和队列,则无须预先分配存储空间。
10. 串在存储结构与具体计算机的编址方式没有关系。

自测题分析及参考答案

一、分析

A. 本叙述是指数据的逻辑结构。

B. 线性表中各数据元素之间存在着线性关系。具有均匀性:各元素具有同样的数据类型;有序性:各元素是有序的,不可交换次序。

C. 线性表有两种存储结构:采用顺序存储结构的线性表叫“顺序表”。采用链接存储结构的线性表叫“链表”。这种存储方式不必为其分配一块连续的存储空间。每个元素的存储单元叫结点,每个结点由两部分组成:数据项和指向下一个数据项的指针,数据元素之间的关系是用指针来体现的。无论插入还是删除操作,只要改变结点的指针即可,无需移动别的元素。

D. 这种数据结构是栈。它的重要特性是“后进先出”。因此,只要处理数据的顺序与这些数据的输入顺序不同时,栈都是很有用的。子程序调用时,由于返回的次序和调用的次序相反,所以,一般是设置一个返回地址栈。高级语言编译中的表达式转换和求值也要用到栈。设操作数栈和运算符栈两个栈,编译程序自左至右扫描表达式,凡遇操作数一律进栈,若遇运算符则判其优先数是否大于运算符栈顶元素的优先数。若大,则进栈;反之,则栈顶运算符退栈,并形成一条只有一次运算的机器指令,指令中的运算对象为操作数栈顶上的两个元素。

E. 数组是线性表的推广,作为一种数据结构,在几乎所有的程序设计语言中都允许使用。它通常采用顺序存储结构。一维数组中的每个数据元素由一个值和一个下标组成;二维数组中的每个数据元素由一个值和一对下标组成; n 维数组中的每个数据元素由一个值和一组下标组成。

F. 数组是相同类型数据元素的序列;而记录是各种不同类型成分组合起来构成的数据结构。记录是可作为一个个单位来处理的一组相连的数据,每个记录由一个或多个数据项组成,各数据项的数据类型可不同。

G. 字符串是一种字符型的数据结构,有一系列串的计算。求出字符串中含有字符个数的操作称为求长度,是串的基本运算之一。很多程序设计语言都提供了求串长度的函数。

H. 简单地说,字符串就是一个字符序列。而串中任意个连续字符组成的字符子序列称为子串。求串中从指定位置开始且长度为一定值的子串的操作称为求子串运算。

- 答案: A③ B⑥ C⑦ D④
E① F⑧ G⑬ H⑮

二、分析

1. 栈是一种所有的插入或删除只能在表的一端进行的特殊的线性表。由于只能在一端出入,所以先进栈的数据“压在下层”,只能后出来,而最后进栈的数据元素最先出栈,它具有“后进先出”的特点。队列是在表的两端进行操作的特殊线性表,专门用于插入元素的一端叫队尾,用于删除元素的一端叫队首。它类似于日常生活中的购物排队。先来的人排在前面,后到的人顺次排在队尾。先来的人买到货物先离开队伍,是一种“先进先出”的结构。

2. 链表可分为单向链表和双向链表。单向链表是用一个指针循一个方向连接的链表,当给定一个表元素时,只能顺链的方向移动查找,影响存取速度。而双向链表有左、右两个链指针,一个接前趋元素,一个接后继元素。可以在前后两个方向上移动,提高检索速度。

3. 已知元素的进栈次序为 1, 2, 3, 4, 5, 根据“后进先出”的特性,它只能得到出栈序列中的 B 组答案。操作步骤是:

- | | |
|-----------|-----------|
| (1)push 1 | (6)push 4 |
| (2)push 2 | (7)pop 4 |
| (3)pop 2 | (8)push 5 |
| (4)push 3 | (9)pop 5 |
| (5)pop 3 | (10)pop 1 |

出栈序列为 2, 3, 4, 5, 1。

4. 排序方法可按不同的原则进行分类。整个排序过程都在内存进行的排序称为内部排序,常用的内排序法有:插入排序、交换排序、选择排序、归并排序和基数排序等。冒泡排序的基本思想是:每次将待排序序列中的数据项两两比较,凡是逆序则进行交换。将序列照此方法从头至尾处理一遍称为一次冒泡,关键字值小的记录好比水中气泡逐次往上飘浮,而关键字值大的记录好比石块沉入水底,如此反复比较交换,直到排好为止。快速排序是对冒泡排序的改进。它的基本思想是通过一次排序将文件分成两部分,然后分别对这两部分进行排序以达到最后整个文件有序。这两种排序方法都是基于比较交换,因此属于交换排序。

答案:1. C, B 2. A 3. B 4. C

三、分析

1. 栈是一种“后进先出”的数据结构。栈底固定而栈顶浮动,要设一个栈指针 top 指出栈的栈顶元素位置。栈有进栈、退栈、判满、判空四种基本运算。在作退栈运算时,应先判别栈是否为空,若已没有数据元素,再作退栈运算就要发生“下溢”。在作进栈运算时,应先判别栈是否已满,当 top=Maxsize 时,表示分配给栈的空间已满,这时再有元素进栈就要发生“上溢”。若栈中元素个数为 n 个,此时再作进栈运算则发生上溢,说明该栈的最大容量就是 n。

对于以顺序方式存储的栈,常采用一块连续的存储区域。在有多个栈的情况下,会遇到两个问题:一是每个栈所需空间大小很难估计;二是每个栈的容量在使用中都是动态变化的。在设置栈时要注意解决其中一个栈发生上溢,而其余各栈尚留有很多空间的问题。当有两个栈共享一片连续的内存空间时,应将两个栈的栈底分别设在这片内存空间的两端,然后,各自向中间延伸。这样,只有当两个栈的栈顶在栈空间的某一位置相遇时才产生上溢。两个栈互补余缺,是一种经常被采用的、比较合理的方案。

2. 线性表中的循环链表,是使单向链表最后一个元素的指针又指向第一个元素构成的。因为是循环结构,所以从表中任一结点出发都能访问到所有结点。使用双向链表,可在前后两个方向上移动。单向链表则只能顺

链沿一个方向移动。顺序存储结构不便插入、删除;对于经常要动态变化的表,应采用链接存储结构。

3. 队列设有队首、队尾两个指针。对大小为 n 的队列,当队首指针值等于队尾指针值时,表明队列为空;当队尾指针=n 时,表明队列已满。队列是一种限定在表的两端进行操作的具有“先进先出”性质的特殊线性表。

4. 查找就是根据给定的某个值,在表中确定一个关键字等于给定值的记录或数据元素。顺序查找的方法是:用待查的关键字值与线性表中各数据元素逐个进行比较,若找出相等值的数据元素则查找成功,若找遍整个线性表都没有值相等的数据元素,则查找失败。顺序查找的优点是对线性表中数据元素的次序无任何要求,不必按关键字值排序;对线性表的存储结构无特殊要求,顺序存储、链接存储均可。若表中有 n 个元素,则比较次数最少一次,最多 n 次,平均比较次数为 n/2 次。二分法查找又称为折半查找,是一种效率较高的线性表查找方法,它不仅基于关键字值查找,而且要求线性表中数据元素按关键字值排序,并且以顺序方式存储该线性表。

5. 数组通常采用顺序存储结构,它是在内存中连续存放的一组数据。这样就有一个按什么次序把数组元素排在线性序列里的问题。常用的排列次序有行优先顺序和列优先顺序。二维数组行主序下元素排列顺序为:

$a_{11}, a_{12}, \dots, a_{1n}, a_{21}, a_{22}, \dots, a_{2n}, \dots, a_{m1}, a_{m2}, \dots, a_{mn}$
列主序下元素的排列次序为: $a_{11}, a_{21}, \dots, a_{m1}, a_{12}, a_{22}, \dots, a_{m2}, \dots, a_{1n}, a_{2n}, \dots, a_{mn}$

二维数组元素的地址计算公式为:

行主序: $LOC(a_{ij}) = LOC(a_{11}) + ((i-1) * n + (j-1)) * L$

列主序: $LOC(a_{ij}) = LOC(a_{11}) + ((j-1) * m + (i-1)) * L$

其中 a_{11} 为第一个元素存放的起始地址, n 和 m 分别为数组每行和每列的元素个数, L 为每个元素占用的存储单元个数。

根据条件可知:二维数组 A 行下标范围是 1 到 9, 列下标范围是 1 到 5, 即 A 数组共有 9 行 5 列 45 个元素, 每个元素用 4 个字节存储, 因此可计算出存储数组 A 共需要 180 个字节, 最后一个元素的第一个字节的地址是 176。

行主序下, 元素 A[4, 5] 的第一个字节的地址计算方法是:

$$\begin{aligned} LOC(A[4, 5]) &= LOC(A[1, 1]) + ((4-1) * 5 + (5-1)) * 4 \\ &= 0 + (3 * 5 + 4) * 4 = 76 \end{aligned}$$

同理, 可计算得 A[6, 3] 的第一个字节的地址是 108。列主序下, A[8, 1] 的第一个字节的地址是:

$$\begin{aligned} LOC(A[8, 1]) &= LOC(A[1, 1]) + ((1-1) * 9 + (8-1)) * 4 \\ &= 0 + (0 + 7) * 4 = 28 \end{aligned}$$

A[2, 4] 的第一个字节的地址为 116。

(下转第 31 页)



单片机最小应用系统与液晶显示器(LCD)的接口

合肥中国科技大学计算中心(230026) 张培仁

液晶显示器是一种极低功耗显示器件,在袖珍式仪表式低功耗应用系统中广泛使用。

液晶显示器是一种被动式显示器件。它本身不发光而只是调制环境光。它与其他显示器相比有如下特点:

(1)低电压(3V—6V),微功耗(0.3 μ w—100 μ w)。它适于与CMOS电路直接相配,可用于各种自动化检测设备中,尤其适合于电池供电的移动式仪器、仪表、手表、计算器中。

(2)不怕亮光,可在室内外明亮环境下使用,显示清晰度不会随环境亮度增加而减弱,在太阳光下也能清晰显示。

(3)体积小,外形薄为平板型显示。手表液晶显示器厚度1.2~1.6mm,计算器显示器厚度1.6~2.2mm,仪器厚度为2.2~5mm,引出线用导电橡胶引出,使用方便。

(4)显示面积、字形、大小在一定范围内不受限制。

1. LCD 基本的工作原理:

LCD是由两层玻璃之间封入一些液晶材料,当光线射入时,产生偏振光,这些光被液晶材料旋转90°,光线被反射回去,呈透明状态;当上下电极加一定电压,液晶分子转成垂直排列,失去旋光性,入射光无法通过下偏振片返回,因而呈黑色。

2. LCD 的驱动方式:

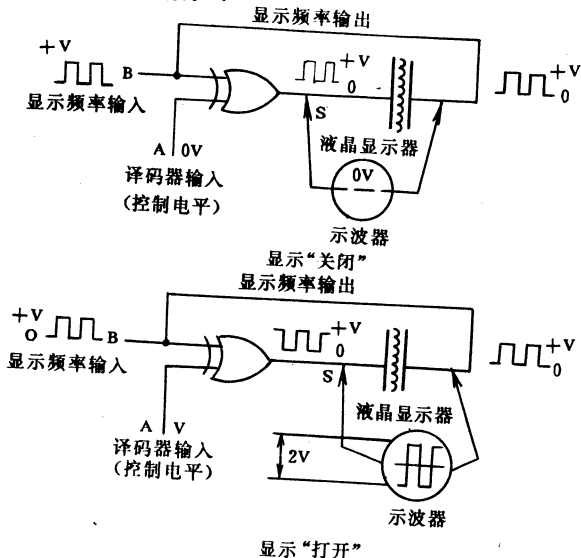


图 1

液晶显示器的驱动方式由电极引线的选择方式确定。当液晶显示器选定后,驱动方式也就确定下来。一般有静态驱动和动态驱动两种。不论哪一种方式驱动都是用方波(交流)来驱动。动态驱动只是分时驱动,也就是每一段片分时地被驱动。静态驱动,广泛用单一寻址由异或门构成的移相电路构成,见图1。

显示“关闭”,A端为“0”电平,S端与B端同相位,这时显示器两端相对电平差“0”V,液晶的片段不显示。显示“打开”,A端的为“1”电平,S端与B端反相位,这时显示器两端构成交流电压,片段显示。这里要说明的是每一段都要有一个异或门来驱动。

液晶显示器交流驱动波形如图2。

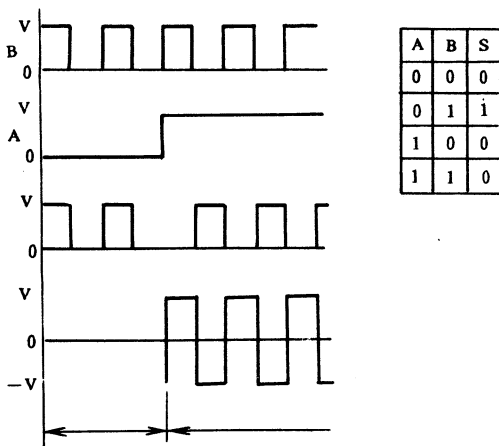


图 2

由于每段都要有一个异或门来驱动,单片机最小应用系统要驱动6个八段液晶显示器,光用8031的P1,P3口是不够,必须扩展2片8255芯片才能胜任。

这样显然用了较多的器件。为了减少器件,也可以用6片4095(小规模芯片)用串行口输出要显示的数据,稳定一段时间后如0.1秒,再从串行口输出上次输出的数据的反码,这样循环下去,同时可以得到稳定的显示,并且接口非常简单,所用芯片比前者大大地减少。因为这时节约2片8255和48个异或门。

3. 单片机和液晶显示器接口,驱动时要注意的问题

首先要保证加到液晶显示器两端交流电压平均值等于零,要注意直流分量越小越好,至少要小于100mV。否则,过大的直流电压使液晶材料迅速分解,大大缩短了显示器的工作寿命(一般直流寿命只有几

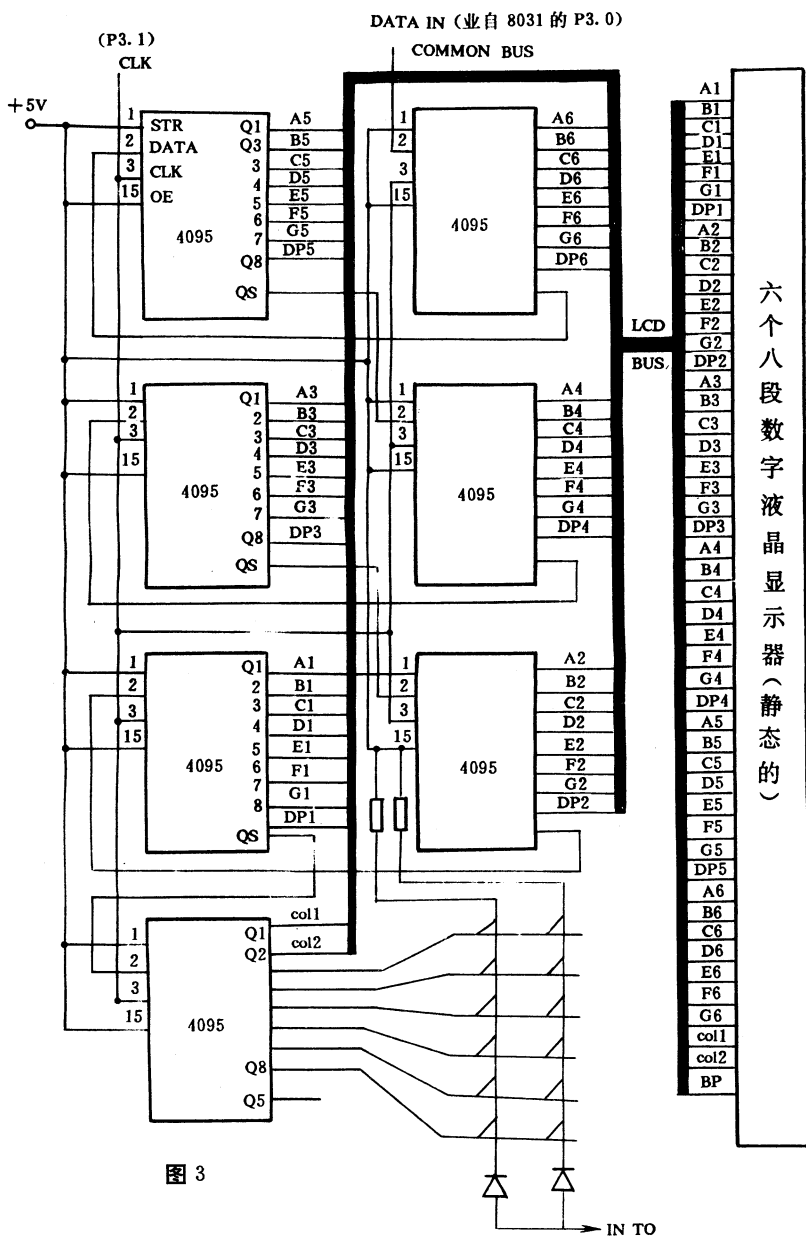


图 3

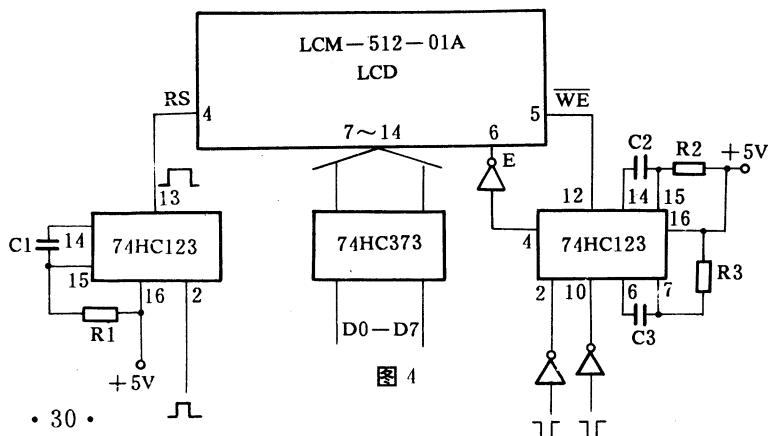


图 4

十小时)。这样就要求显示器的公共电极绝对不能接数字信号地。要加一个显示频率方波，它的相位与段电极（也称前电极）相位相反，两者严格对称，从而保证显示器平均电压等于零，如图 1。在图 3 中从单片机最小系统的 8031 串行口 P3.0, P3.1 分别输出要显示的数据和同步时钟。LCD 的公共电极也接在 4095 一个端上，这样公共极和前极同步求反，保证平均直流电压为 0。图 3 是 6 个八段静态液晶显示器驱动和接口电路。

这个接口所用程序是很简单的。即把串行口设为移位寄存器，每次输出 7 个字节数据（其中包括公共电极的）。经过 0.1 秒原来数据求反再输出，这样重复循环下去。

对于动态液晶显示器和单片机接口电路是比较复杂的。因为动态液晶要分时驱动不同的片段。一般一个 LCD（动态）有 3 个到 4 个 COM 端。如果有 32 根 SEG 线（即段驱动端，公共端有 3 个）这样可以显示 $32 \times 3 = 96$ 个显示段。每段是否显示由每个 SEG 与 COM 的电压来决定。这样使线路复杂化。但是在国外一般已把动态驱动集成到一个芯片上，因此它与单片接口也就不复杂了。

我们在便携式数据采集器采用日本三洋产的点阵式液晶显示组件 LCM-512-01A 作显示器。一次可显示一行 16 个 ASCII 码字符。这种显示屏由于内部有控制电路，字符发生器和显示数据存储，其驱动方式简单，可直接与 8031 的数据总线相连，进行写命令和数据。显示器和 8031 还有两条控制线 E 和 WE，一条命令/数据选择线 RS。其中 E 为选通 LCD 的使能信号，高电平有效。WE 为 LCD 的读/写控制线。WE 为高电平，则 8031 对 LCD 进行读操作，WE 为低电平时 8031 向 LCD 写命令或数据。由于

LCD 是一种低速 I/O 设备,它接受一个命令大约要 1—2 μ s 以上,而执行一个命令则需要 40 μ s—1.6ms 之间,因此 8031 向 LCD 写时要保持 1—2 μ s。对于 8031 的 12MHz 晶体,一般一条指令小于 1 μ s。所以我们为了保证正确的控制,传送用了 74HC123 单稳电路,从而保证速度的匹配,使数据的可靠传送。同时用 74HC373 锁存 8031 的数据线的的数据。图 5 中定时电容 C1—C3

(上接第 25 页)

· 寻址方法是先在零页中作间接寻址,后再进行绝对 Y 变址寻址。

· 指令有: LDA(\$nn), Y, STA(\$nn), Y, ADC(\$nn), Y, SBC(\$nn), Y, AND(\$nn), Y, EOR(\$nn), Y, ORA(\$nn), Y, CMP(\$nn), Y 等 8 条。

四、寻址方式小结

1. 寻址方式共 13 种,但不是每一类指令都具有,那一类指令具有哪些寻址方式,可以查 6502 指令表。

2. 学习不同的寻址方法,应注意它们之间的异同点,特别应该掌握它们各自的功能。

3. 编程时可根据具体要求,灵活选用某条指令允许的某一种寻址方式,以达到程序占用内存少,执行速度快的目的。

4. 同一道题目,可以有不同的编程方法。一般来说,同一道题目可以用不同的寻址方式指令处理。

5. 累加器寻址、隐含寻址方式的操作数在 CPU 的

为 150P,电阻 R₁—R₃ 为 33K,可满足大于 2 μ s 脉宽。为了简化硬件我们没有采用访问 LCD 的忙标志。这样避免数据双向锁存。

总之液晶显示器与单片机(特别 CMOS)相结合在很多领域很有用途。特别在携带式移动的仪器和设备更是如此。

内部寄存器中,不必访问存储器,执行速度快。

6. 零页寻址占用内存少,执行速度快,有利于提高编程的效率。

7. 变址寻址方式,常用于处理数据块、表格等问题,方便灵活。

8. 立即数寻址常用于设置初始数据,程序初始化或当计数器选用。

9. 相对寻址专用于条件转移指令,偏移量指出跳转的目标地址。

10. 间接寻址常用来存放专用程序的入口地址,从而使专用程序方便地在存储器中浮动。

11. 先变址(X)间接寻址适用于多个数据块传送,后变址(Y)间接寻址则常用于动态数据处理。

12. 零页 X(Y)变址寻址和先变址(X)间接寻址,仅限于在零页中进行变址,而绝对 X(Y)变址,后变址(Y)间接寻址,可访问 64K 空间的任一地址。

(上接第 28 页)

答案:1. A② B① C② D④ E③

2. A④ B③ C① D②

3. A④ B③ C② D③

4. A⑤ B③ C②

5. A⑧ B③ C⑤ D① E⑥

四、分析

1. 正确。

2. 错误。顺序存储方式的优点是存储密度大,但由于插入、删除时要引起大量数据元素的移动,所以运算效率低。

3. 错误。双向链表的结点中就包含左、右两个链指针。

4. 正确。

5. 错误。本叙述对数组来说是正确的,而记录可以包含

一个或多个数据项,各数据项可以具有不同的数据类型。

6. 错误。队列的两端都不固定,一端专门进行存(插入)操作,另一端专门进行取(删除)操作,两端都在动态变化。

7. 正确。

8. 正确。

9. 正确。

10. 错误。串的顺序存储方式分为紧缩格式和非紧缩格式。采用紧缩格式时,一个存储单元中存放几个字符与机器字的长度有关,所以与计算机的编址方式是有关的。

答案:1. \checkmark 2. \times 3. \times 4. \checkmark 5. \times

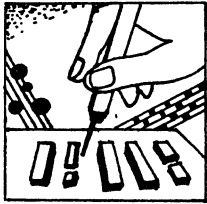
6. \times 7. \checkmark 8. \checkmark 9. \checkmark 10. \times

(上接第 44 页)

四、保护电路和次级整流滤波

如果启动电源时,发出“嗡嗡”或“滴答”声时,首先肯定有一组输出电压短路。测量 R₂₁ 电阻值可能比原值增大 7—12 Ω ,使在它的压降足以触发可控硅 Q₅ 控制端,使可控硅导通,致使输出短路,电路停止振荡,起到保护电路作用。如果测量 R₁₂ 值与原值差别不大,就要检查相应次级整流滤波电路上的有关元件。

另外发出“嗡嗡”或“滴答”声有另一个原因。测量 +12V 时,电压值超过 +15V。稳压管 CR₂₀ 导通,使可控硅 Q₅ 导通,也造成 +12V 短路接地而这种过压又不能通过 R₁₆ 来调正。出现这个问题的关键是振荡环路上 R₁₁ 电阻值变小使其振荡反馈信号增大, Q₃ 集电极电压增大,使次级端产生电压过大,适当选择 R₁₁ 使次级整流 +12V 电压不能超过 +15V 电压,方可使电源正常工作。



学装微电脑

电源 ON、OFF 自动运转装置

易齐干

内藏微电脑的智能仪器的可靠性试验,是对电源进行反复地 ON、OFF 实验。以一定的时序,电源 ON、OFF,检验仪器工作是否正常。如果设计或装配存在问题,电源 ON 的瞬间,微电脑必然出现混乱,仪器不能正常工作。

传统的方法是用延时器、计数器和继电器组合成电源 ON、OFF 装置。存在的缺点是时间的设置受到限制。现在使用微电脑控制,通过编制各种各样的延时子程序,可以获得与使用要求相一致的时间。同时,根据需要能很方便地设置 ON、OFF 次数。

该装置的框图如图 1 所示。按压启动开关,显示所设置的 ON、OFF 次数。以设置 ON 的时间,启动继电器,继电器工作到所设置的 OFF 的时间,计数器显示 -1。之后,按所设置的 ON、OFF 次数重复动作。如果计数器显示 0,即使继电器为 ON,但装置已停止工作。如果装置在运行中,想让其之暂停,要按动停止开关,微电脑执行中断,即使继电器为 ON,装置 ON、OFF 运转暂停。

该装置所设置的 ON、OFF 最小时间间隔为 1 秒,最大为 15 秒,ON、OFF 次数最多为 15 次。改变程序很容易调整时间与次数。

电路原理图如图 2 所示。端口 A 和端口 C 低位相连开关为 ON,经过 IC7407 反相器,将低电平变为高电平输入给微电脑。

端口 B 输出经过 M54405 驱动 7 段数码管。

端口 C 第 4 位连接继电器部件,输出高电平,继电器为 ON。

程序简略流程图如图 3 所示。

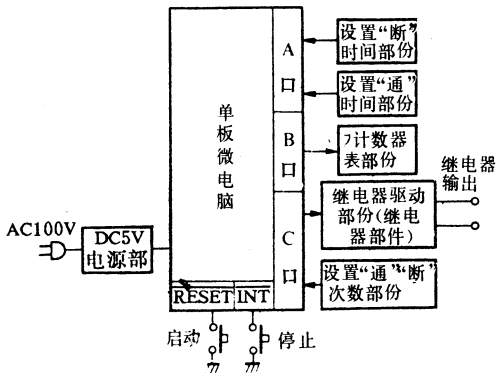


图 1

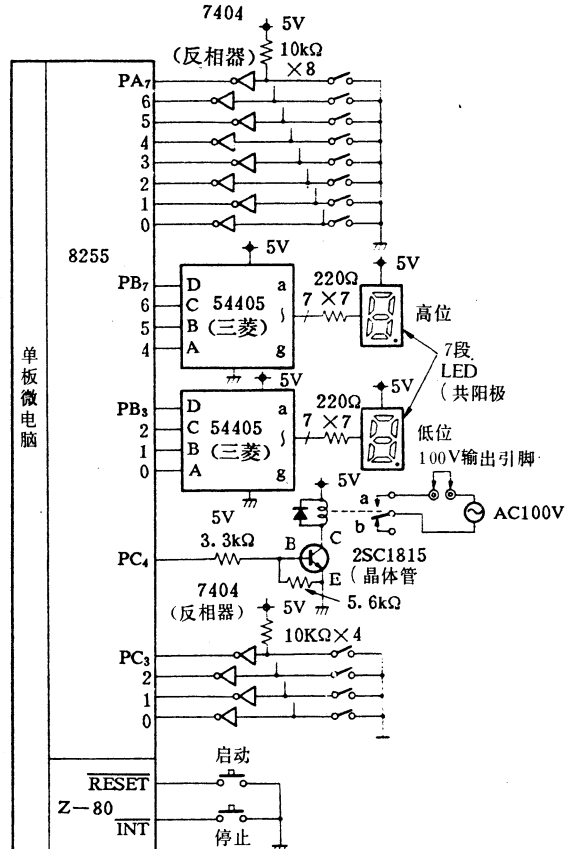
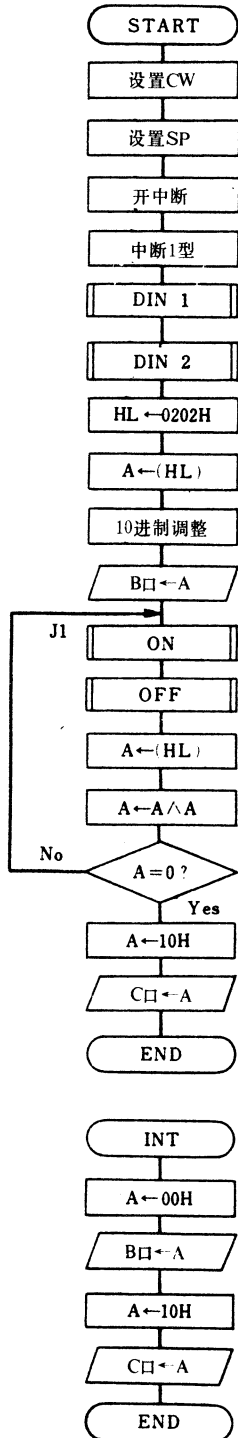


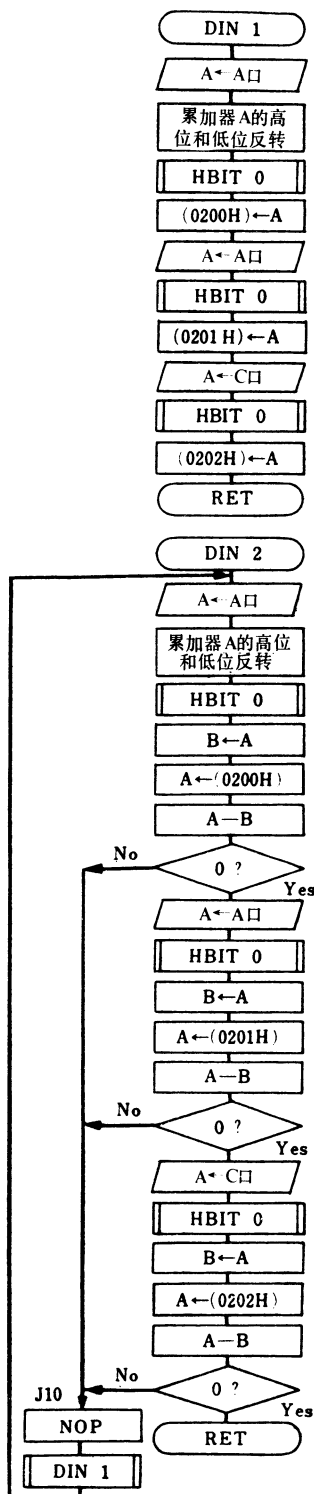
图 2

表 2



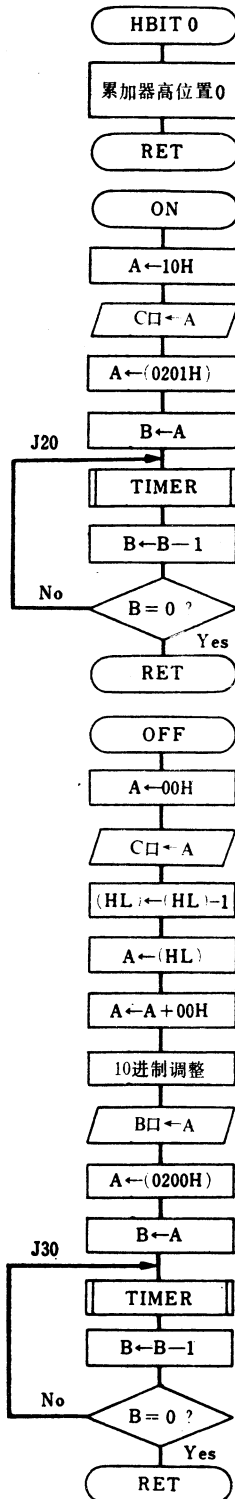
标记	助记符	地址	机械语	注 释
START	LDA,91H	0000	3E 91	A口、C口低位输入 B口、C口高位输出
	OUT(03H),A	0002	D3 03	
	LD SP,0700H	0004	31 00 07	设置堆栈
	EI	0007	FB	开中断
	IM 1	0008	ED 56	中断1型
	CALL DIN1	000A	CD 50 00	读入数据
	CALL DIN2	000D	CD 80 00	再次读入数据
	LD HL,0202H	0010	21 02 02	“通”—“断”次数 数据进行10进制调整, 向B口输出
	LD A,(HL)	0013	7E	
	ADD A,00H	0014	C6 00	
	DAA	0016	27	
	OUT(01H),A	0017	D3 01	
J1	CALL ON	0019	CD D0 00	ON
	CALL OFF	001C	CD E0 00	OFF
	LD A,(HL)	001F	7E	设置的“通”、“断”次 数终止了吗?
	AND A	0020	A7	
	JP NZ,J1	0021	C2 19 00	
	LD A,10H	0024	3E 10	如果设置的“通”、 “断”次数终止, 继电 器为“通”之后结束。
	OUT(02H),A	0026	D3 02	
END	HALT	0028	76	
INT	LD A,00H	0038	3E 00	如果发生中断, 计数 器显示为0, 继电 器为“通”之后, 结束
	OUT(01H),A	003A	D3 01	
	LD A,10H	003C	3E 10	
	OUT(02H),A	003E	D3 02	
END	HALT	0040	76	

表 2



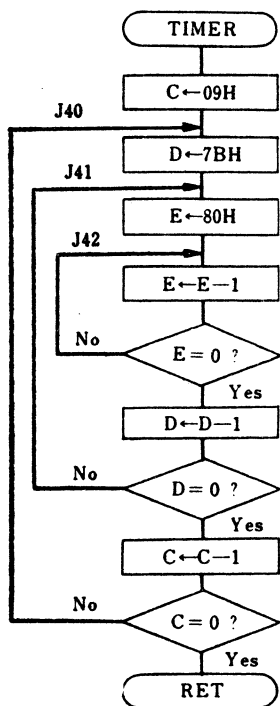
标记	助记符	地址	机械语	注释
DIN1	IN A,(00H)	0050	DB 00	读入“断”时间数据
	RRC A	0052	CB 0F	累加器A的高位和低位反转
	RRC A	0054	CB 0F	
	RRC A	0056	CB 0F	
	RRC A	0058	CB 0F	
	CALL HBIT0	005A	CD C0 00	“断”时间数据放入0200H地址
	LD(0200H),A	005D	32 00 02	
	IN A,(00H)	0060	DB 00	读入“通”时间数据
	CALL HBIT0	0062	CD C0 00	“通”的时间数据放入0201H地址
	LD(0201H),A	0065	32 01 02	
IN A,(02H)	0068	DB 02	读入“通”-“断”时间设定值	
CALL HBIT0	006A	CD C0 00	“通”-“断”次数数据放入0202H地址	
LD(0202H),A	006D	32 02 02		
RETURN	RET	0070	C9	
DIN2	IN A,(00H)	0080	DB 00	读入“断”时间数据
	RRC A	0082	CB 0F	反转累加器A的高位和低位
	RRC A	0084	CB 0F	
	RRC A	0086	CB 0F	
	RRC A	0088	CB 0F	
	CALL HBIT0	008A	CD C0 00	“断”时间数据放入B寄存器
	LD B,A	008D	47	
	LD A,(0200H)	008E	3A 00 02	比较两次读入的数据不相等时跳入J10
	CP B	0091	B8	
	JP NZ,J10	0092	C2 B0 00	
	IN A,(00H)	0095	DB 00	读入“通”时间数据
	CALL HBIT0	0097	CD C0 00	“通”时间数据放入B寄存器
	LD B,A	009A	47	
	LD A,(0201H)	009B	3A 01 02	比较两次读入的数据,不相等时,跳入J10
CP B	009E	B8		
JP NZ,J10	009F	C2 B0 00		
IN A,(02H)	00A2	DB 02	读入“通”-“断”时间设定值	
CALL HBIT0	00A4	CD C0 00	“通”-“断”次数数据放入寄存器B	
LD B,A	00A7	47		
LD A,(0202H)	00A8	3A 02 02	比较两次读入的数据不相等时,跳入J10	
CP B	00AB	B8		
JP NZ,J10	00AC	C2 B0 00		
RETURN	RET	00AF	C9	
J10	NCP	00B0	00	比较结果不相等时,再次读入数据,返回D1N2
	CALL DIN1	00B1	CD 50 00	
	JP DIN2	00B4	C3 80 00	

表 2



标记	助记符	地址	机械语	注 释
HBIT 0	RES 7,A	00C0	CB BF	累加器的高位置 0
	RES 6,A	00C2	CB B7	
	RES 5,A	00C4	CB AF	
	RES 4,A	00C6	CB A7	
RETURN	RET	00C8	C9	
ON	LD A,10H	00D0	3E 10	继电器为“通”
	OUT(02H),A	00D2	D3 02	
J20	LD A,(0201H)	00D4	3A 01 02	仅在设定时间继电器为“通”
	LD B,A	00D7	47	
	CALL TIMER	00D8	CD D0 00	
	DEC B	00DB	05	
	JP NZ,J20	00DC	C2 88 00	
RETURN	RET	00DF	C9	
OFF	LD A,00H	00E0	3E 00	继电器“断”
	OUT(02H),A	00E2	D3 02	
J30	DEC(HL)	00E4	35	端口B显示的“通”→“断”设定次数 - 1
	LD A,(HL)	00E5	7E	
	ADD A,00H	00E6	C6 00	
	DAA	00E8	27	
	OUT(01H),A	00E9	D3 01	
J30	LD A,(0200H)	00EB	3A 00 02	仅在设定时间继电器“断”
	LD B,A	00EE	47	
	CALL TIMER	00EF	CD D0 00	
	DEC B	00F2	05	
	JP NZ,J30	00F3	C2 BF 00	
RETURN	RET	00F6	C9	

表 2



标记	助记符	地址	机械语	注 释
TIMER	LD C,09H	0100	0E 09	延时约1秒钟
J40	LD D,7BH	0102	16 7B	
J41	LD E,80H	0104	1E 80	
J42	DEC E	0106	1D	
	JP NZ,J42	0107	C2 D6 00	
	DEC D	010A	15	
	JP NZ,J41	010B	C2 D4 00	
	DEC C	010E	0D	
	JP NZ,J40	010F	C2 D2 00	
RETURN	RET	0112	C9	

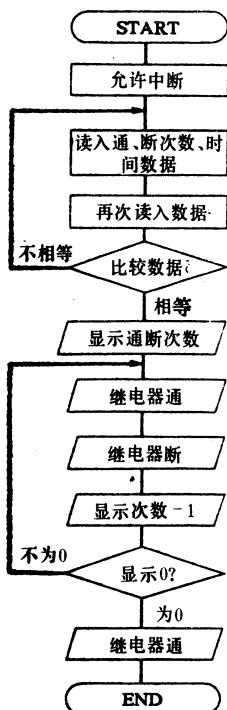


图 3

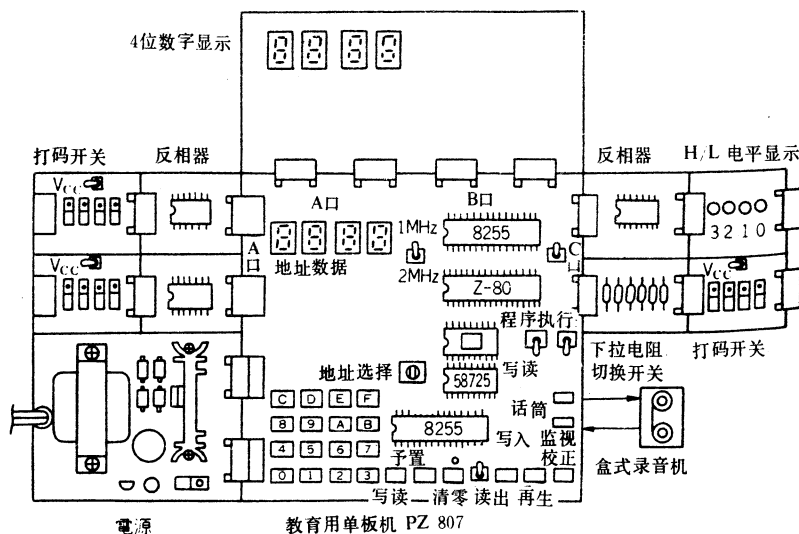


图 4

执行仿真程序, 4 位数显部件显示预先设置的 ON、OFF 次数, 端口 C 高位电平显示部件的第 0 位 LED 灯亮表示所设置的时间, 同时, 4 位数显部件出现减 1 显示。

如果仿真程序无误, 即可将仿真程序改为工作程序, 根据需要考虑是否写入 EPROM 存储器。装入工作用微电脑, 对仪器完成电源 ON—OFF 自动运转控制。



微机视频接口器的研制

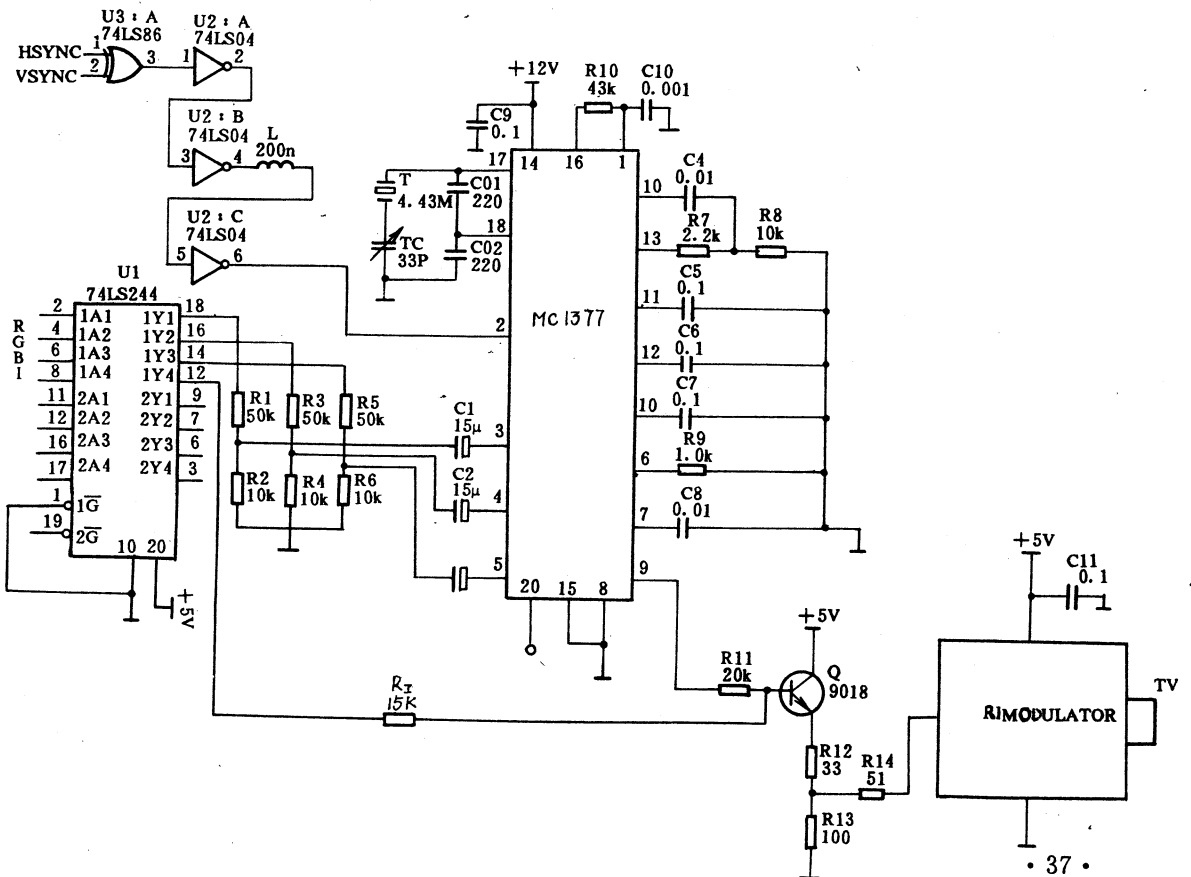
广西民族学院物理系(530006) 韦江维 覃龙生

目前,很多用户尤其是众多经费不足而又必须开设微机实验的大、中专院校、基础科研及家庭教育、娱乐,都需要一台既便宜又能完成一定任务的电脑。低档PC机已基本能满足这一需要。如能考虑充分利用现有资料又能增强PC机功能,那将给低档PC机增添色彩。为此,我们成功地研制出微机视频接口器,接上家用电视机就可完全取代原来的彩色显示器,使用户根据自己的需要,花不多的钱,就能配置成真正的个人电脑。

在我国,彩电的制式都采用PAL制,而目前几乎所有的微机显示系统都采用NTSC制。另外,在电视发射与接收系统中,为抑制外界的干扰,全电视信号的图像信号采用负极性调制,而由于微机显示系统受外界干扰较小,因此采用正极性调制,其行场同步信号是高电平有效。本视频接口器成功地完成上述的信号转换,即把微机的显示信号(包括R、G、B、I、VSYNC、HSYNC)在新副载波基础上重新组合,形成符合我国电视技术标准的视频信号,原理如图1所示。

本接口器的核心部件采用了MOTOROLA公司生产的MC1377彩色电视信号处理器,又称为PAL/NTSC转换器。它的主要功能就是从基础波段取出红、绿、蓝和同步信号,按PAL(或NTSC)制直接产生复合视频信号。其内部电路主要有:副载波形成电路、同步电路、压控相位调整、解编码器、输出放大调整电路。使用MC1377芯片,使得接口器电路十分简单,易于调制,工作可靠。

电路中,74LS244是视频接口器与彩色显示适配器的接口; $R_1 \sim R_6$ 是分压网络,因为MC1377的R、G、B输入信号电平要求为 $1.0V_{P-P}$ (输入的彩色信号电平为 $5.0V_{P-P}$); C_1, C_2, C_3 为MC1377的输入电容;同步信号 H_{SYNC}, V_{SYNC} 经异或门(74LS86)后形成复合同步信号,再经反相器(74LS04)反相后送至MC1377的复合同步输入端(2脚),另外的两级反相和电感L在一定程度上起到延时同步信号的作用;17、18脚为时钟输入,晶振采用4.43MHz晶体振荡器;20脚为PAL/NTSC转换开关,悬空为PAL制;19脚为(R-Y)与(B



—Y)信号相位调整;16脚为8.2V参考电压输出;1脚为同步信号调整;10、13脚为色度信号调整;6、8脚为延时调整;9脚为复合视频信号输出,其输出幅度为 $2.5V_{P-P}$ 。晶体管Q主要起放大、阻抗匹配作用。RF调制器电路包括本机振荡器和混频器,它所输出的高频振荡与视频信号混频、经混频之后可将此信号直接与电视机天线连接。调节电视机频道,电视机就可当彩色显示器使用了。

显然,本接口器输出的只是复合视频信号,要形成全电视信号还得有音频信号。在这里我们就如何才能获得全电视信号作简单讨论;在介绍接口器电路时,曾经提到MC1377 16脚为8.2V参考电压输出,其主要作用是提供MC1374直流工作电压,MC1374是音、视频复合放大输出器,其内部电路主要有:音频调节器、声音载波振荡器、RF振荡器、RF调整器,只要将MC1377输出的复合视频信号和音频信号送至MC1374的相应引脚,就可获得全电视信号。当然,使用MC1374时在技术上有它自己的要求,另外还得需要一些外围电路支持(如4.5MHz陷波电路等),这里就不一一介绍了。

本视频接口器可连接黑白或彩色电视机,值得一提的是,用家用电视机充当显示器,由于分辨率有限,较理想的选择是选取CGA显示模式。

XMF学习机(单驱扩展箱) 改装为XMF及PC两用机

南京903信箱33分箱 杨青海

中华学习机有丰富的学习软件,是中小学教学的主要机种,但在实际工作中所使用的绝大多数为PC系列机,应用软件丰富,使学习机相形见绌。目前PC机价格虽然逐年降低,但皆在3000元左右,对一般家庭而言,这是一笔不小的开支。最近我对XMF机进行改装,投资千元,使XMF机成为XMF及PC两用机,大大提高了性能价格比,较好的解决了学与用的矛盾,效果良好,有兴趣的同志不妨一试。改装原理见图1。

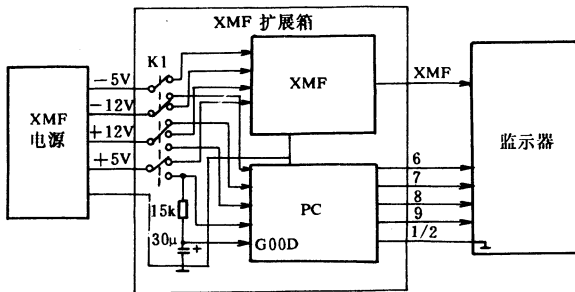
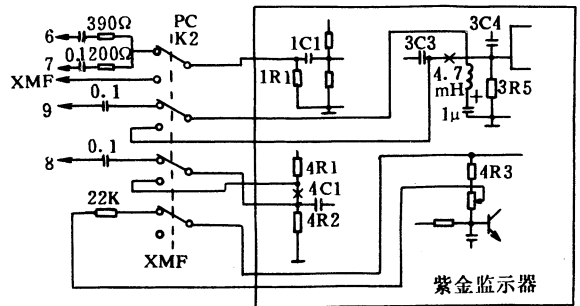


图1

一、电源及其改装:XMF机电源设计容量较大,该机使用手册提供数据为:+5V 4A,+12V 2A。机内电源组件上标注为:+5V 7A,+12V 3A,而PC机在基本配置(单驱、显示卡)状态下,实测+5V电流不超过3A, $\pm 12V$ 、-5V电源容量更没问题,这样就为PC机解决了关键的供电问题。PC机所需的电源GOOD信号可通过简单的RC电路解决(见图1)。电源改装比较简单,打开电源组件外壳,把印刷板抽出一小部分,即可看到引出线焊点,用新的引出线(长度约30cm,最好与原线颜色、粗细一致),逐根替换原引线(原电源线及插头留给XMF机用),本人选用小型瓷质8D2W波段开关,其中3刀并联给+5V用,2刀并联给+12V用,-12V及-5V各用一刀,电源地线与XMF、PC机主板地线接在一起不需控制。



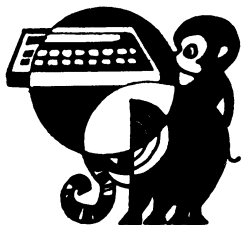
注:1、2、6、7、8、9为显示卡9针插头针号

图2

二、显示器改装:如果选配具有直接使用电视机及学习机显示器接口的CGA卡则不需对学习机显示器进行改动。如使用单显或双频卡,把学习机显示器改为XMF-PC两用显示器(CG/MDA),改装方法可参看《软件报》91年第23期。本人使用的显示器为紫金CZX-12型,改装方法略有不同,利用 4×2 拨动开关分别切换数据输入、行频、帧频调整,线路见图2。帧同步输入端与地之间增加 $4.7mH$ 、 $1\mu F$ 只元件组成的串联电路,是为了减少光栅顶部的卷边现象。为求简单,行扫描电路只把行频调整进行改动,所以 720×350 方式工作时,西文方式只能显示75列字符,但影响不大,中文方式下显示正常。

三、结构安装:XMF机单驱扩展箱中空余一台驱动器的位置,正好可以固定一只5.25英寸半高驱动器。电源组件移开,空出PC主板安装位置。在安装时应根据键盘接口及扩展槽的位置进行调整,以便于安装显示卡和驱动器卡。应注意:XMF机扩展箱内有效高度只有11cm,所以主板与底板之间在保证绝缘的前提下间隙应尽量小,否则显示卡将无法直立安装,有条件时使用90度转接口更为方便。电源组件可固定在机内空余处,但为了提高可靠性,利于散热,建议把电源固定在机箱后部。

改装后,在PC机状态下连续工作4小时以上运行正常。



电脑游戏机

第三章 F BASIC 的画面控制语句

山东苍山县机械电子化学工业局(277700) 于 春

四、SP 动作控制语句

1. 定义 SP 动作语句(DEF MOVE)

DEF MOVE 简写 DE. M.

语句功能是定义卡通图案的种类、运动方向、运动速度、移动量等。其格式为

DE. M. (n)=SP. (F,G,H,I,C,A)

式中(1)n—动作编号 0~7。

(2)F—SP 的种类编号 0~15

(3)G—运动的方向 0~8

(4)H—运动速度 1~255,H=1 最快,为 1 秒移动 60 点;H=255 最慢,255 秒移动 60 点。

(5)I—移动量 1~255,每一单位移动量为 2 点。如 I=200,则移动 400 点。

(6)C—显示的优先度 0~1,C=0 显示于 BG 面前,C=1 显示于 BG 后面。

(7)A—配色号码 0~3。

2. 启动 SP 语句(MOVE)

MOVE 简写 M.

MOVE 语句的功能是启动 DEF MOVE 语句所定义的 SP 开始运动,直到运动完 DEF MOVE 语句规定的移动量才停止。MOVE 语句的格式为

M. n₀,n₁,...n₇ n₀~n₇ 任选

式中 n 为动作编号。MOVE 语句可同时启动 1~8 个卡通图案开始运动。

例 14 使乌龟向左移动 200 点。

```
10 CLS : SP. O.
20 DE. M. (0)=SP. (13,7,1,100,0,0)
30 M. 0
```

RUN 乌龟从屏幕中心(120,120)处开始向左移动,完成移动量后停止。

说明:在 DEF MOVE 语句中,运动方向 G、显示优先度 C 和配色号码 A 可以省略,省略 A 时,计算机自动设定 A=0,并使 SP 从(0,0)处开始运动。省略 G 时,计算机自动选定运动方向为 4。在省略时要注意 A、C、G 可以省略,但分隔符“,”要保留。

上例欲使乌龟反复动作,可加—40 句

```
40 PAU. 300 : G. 10
```

RUN 后,乌龟反复移动,可以发现,乌龟每次移动的起点都在屏幕中心。能否人为设定乌龟的运动起点呢?答案是肯定的。POSITION 语句可以达到这一目的。

3. 规定 SP 动作坐标语句(POSITION)

POSITION 简写 POS.

语句格式为

POS. n,X,Y

式中 n:动作编号 0~7。X,Y 为动作前的座标(起点座标)。X,Y 取值,一般 X 取 0~240,Y 取 0~220。

例 15 定义太空船向右上方运动,起点座标为(50,50)。

```
10 CLS : SP. O.
20 DE. M. (0)=SP. (9,2,1,100,0,0)
30 POS. 0,50,50
40 M. 0
```

RUN 太空船则从(50,50)起向右上方运动,若使其反复运动,则需加入 50 句

```
50 PAU. 300 : G. 10
```

注意:在例 14、15 所加的反复运动语句中均加了 PAUSE 300,即程序执行暂停 300,引入暂停的作用是给卡通图案留出一段时间完成所定义的运动量。若不加这一语句,则卡通图案只在原地抖动而不能移动。

例 16 试编程序使太空船的运动起点和运动方向随机变化。

```
10 CLS : SP. O.
20 I=RND(9) : IF I=0 T. 20
30 X=RND(240) : Y=RND(220)
40 DE. M. (0)=SP. (9,I,1,100,0,0)
50 POS. 0,X,Y : M. 0
60 PAU. 300 : G. 10
```

例 17 编一程序使画面上显示 8 个卡通图案分别向 8 个方向运动,起点座标随机变化。动作反复进行。

```
10 CLS:SP. O.
20 FOR I=0 TO 7
30 DE. M. (I)=SP. (I,I+1,I+1,100,0,0)
40 X=RND(240) : Y=RND(220)
50 POS. I,X,Y
60 M. I : N.
70 PAU. 500 : G. 10
```

RUN 屏幕上显示 0~7 八种卡通图案从不同的起点,以不同的速度、向八个方向反复运动。

若将上例 30 行中配色号码由 0 换为 I/2,可以使八个卡通图案显示四种色彩。

若要 8~15 八种卡通图案显示、运动。可将 30 行中的 F 由 I 改为 8+I 即可。

从以上几例的程序运行中,读者可以发现,尽管 SP 的运动起点可以设定,但在反复运行时,第二次运行的起点不是第一次运行的终点,即运行路线不连贯。为使运行路线连贯,再介绍一个语句。

4. 测量 SP 的座标语句(XPOS, YPOS)

XPOS 简写 XP.
YPOS 简写 YP.

使用这两个语句,可以测量卡通图案动作前的座标或动作结束时的座标,也可以连续测试卡通图案在运动中的座标。其格式为

XP. (n) : YP. (n)

式中 n 为卡通图案的动作编号,两语句中的 n 必须相同。有了这两个语句,可在卡通图案动作前,测量出所在座标(X, Y),然后把座标值赋给 POSITION 语句,就可以使卡通图案的运行路线连贯。

例 18

```
10 CLS : SP. 0. : X=50 : Y=50
20 F. I=1 TO 8
30 DE. M. (0)=SP. (0, I, 1, 100, 0, 0)
40 POS. 0, X, Y
50 M. 0 : PAU. 300
60 X=XP. (0) : Y=YP. (0) : N.
```

RUN 显示玛丽由(50, 50)开始连续向八个方向运动。

5. 判断动作完成语句[MOVE(n)]

以上举例中,都要引入 PAUSE 语句,留出一定的时间,等待卡通图案完成运动量后再运行以后程序。这种方法虽然有效,但不精确。尤其在连续运动时,不是时间不够,没有完成移动量就改为下一运动,就是时间多余,上一移动量完成后停一段时间才进行下一运动。引入 MOVE(n)语句可以弥补这一缺陷。

MOVE(n) 简写 M. (n)

MOVE(n)语句用于判断 n 号卡通图案动作是否完成。当动作正在进行中 MOVE(n)=-1,当动作完成时 MOVE(n)=0。

例 19 编一程序,令玛丽哥哥走八边形

```
10 CLS : PAU. 100
20 LOC. 9, 9 : P. "Ma Li Ge Ge";
   LOC. 6, 10 : P. "Zou Ba Bian Xing"
30 SP. 0. : X=35 : Y=130
40 F. I=0 TO 7
50 DE. M. (0)=SP. (0, I+1, 1, 30, 0, 0)
60 POS. 0, X, Y
70 M. 0
80 IF M. (0)=-1 T. 80
90 X=XP. (0) : Y=YP. (0)
100 N.
```

6. 令 SP 停止、消失语句(CUT, ERA, CAN)

(1) 令卡通图案停止动作语句(CUT)

CUT 语句可使 MOVE 语句启动的卡通图案在移动中停止,等待下一个 MOVE 语句的启动。再启动后,继续运行上一次未运行完的移动量,直至运行终了才停止。CUT 语句可同时定义八个卡通图案停止运动。其语句格式为:

CUT n₀, n₁, ……n₇ n 为动作编号

(2) 令卡通图案消失语句(ERA)

ERA 可使 MOVE 语句启动的卡通图案在移动中消失,直到下一次 MOVE 语句启动后,再显示,继续运行上次未运行完的移动量,直至运行终了才停止。ERA 语句可同时定义八个卡通图案消失。语句格式为

ERA n₀, n₁, ……n₇ n 为动作编号

(3) 令卡通图案永远消失语句(CAN)

CAN 语句可使 MOVE 语句启动的卡通图案在移动中永远消失。即使再输入 MOVE 语句也不能启动显示,直到程序运行终了。再重新运行或者再重新用 EDF MOVE 语句定义,才能显示。该语句主要用于游戏程序设计中,当卡通图案超过八个时,使程序在运行中消除部分卡通图案,另定义启动其它的卡通图案。(注:虽然 ERA 语句也能令卡通消失,但它仍占用八个卡通之中的位置,这时,若定义其它卡通,仍不能显示。)

在游戏程序的设计中, CAN 语句是一条较有用的语句,随机手册中没有介绍,读者自己添上。

CAN 语句可同时定义八个卡通图案永远消失。其语句格式为

CAN n₀, n₁, ……n₇

(4) 三个语句的不同点

三个语句的不同点在于 CUT 语句只令卡通图案运行停止,但不消失; ERA 语句令卡通图案停止并消失; CAN 语句则令卡通图案永远消失。 CUT、ERA 语句作用后,可用 MOVE 语句再启动; CAN 语句作用后,用 MOVE 语句不能启动。实际上, CAN 语句的作用就是解除 DEF MOVE 语句的定义。

例 20, 编一程序, 演示 CUT、ERA、CAN 语句的功能区别。

```
10 CLS : SP. 0.
20 DE. M. (0)=SP. (0, 3, 1, 100, 0, 0)
30 DE. M. (1)=SP. (1, 3, 1, 100, 0, 0)
40 DE. M. (2)=SP. (2, 3, 1, 100, 0, 0)
50 POS. 0, 10, 70,
60 POS. 1, 10, 120
70 POS. 2, 10, 170
80 M. 0, 1, 2 : PAU. 100
90 CUT 0 : P. "CUT0" : PAU. 100
100 ERA 1 : P. "ERA1" : PAU. 100
110 CAN 2 : P. "CAN2" : PAU. 100
120 M. 0, 1, 2; P. "MOVE 0, 1, 2"
```

RUN 显示 3 个卡通图案, 0 号玛丽哥哥, 1 号丽莎, 2 号飞鸟自屏幕左边向右运动。运动大约 2 秒钟, 玛丽停止, 屏幕左上角显示 CUT0, 随后大约 2 秒钟, 丽莎消失, 左上角显示 ERA1, 又过大约 2 秒钟, 飞鸟消失, 左上角显示 CAN2, 两秒钟后, 丽莎出现与玛丽同时运动, 左上角显示 MOVE 0, 1, 2, 这时, 飞鸟不再出现。

通过上例, 读者可以直观体会到这三个语句的不同和各自的特点。

上例运行可反复按 RUN 多观察几次, 也可以补入

使程序自动反复运行。

7. 判断卡通相遇语句(CRASH)

CRASH 简写 CR.

在游戏程序的设计中,经常要判断一卡通图案是否与其它卡通相遇,若相遇,相应作某种处理。如采蘑菇游戏中,当玛丽与红蘑菇相遇时,身体要比以前增大一倍。这就要首先判断是否相遇,若相遇转去执行身体长大子程序后再返回,若不相遇,则继续往下运行。在F BASIC语言中判断是否相遇使用的是CRASH语句,其格式为

CRASH(n) 简写 CR.(n)

式中n为被测试卡通图案的动作编号,在卡通图案的运行中,被测卡通与几号卡通相遇,CRASH(n)的值就等于相遇卡通的编号,未相遇CRASH(n)的值则等于-1。

例21,令0~7类八个卡通图案随机运动,分别测试8个卡通是否与其它卡通相遇,打印出各自的CRASH值。

```
10 CLS : SP. O.
20 F. I=0 TO 7
30 DE. M. (I)=SP. (I,I+1,I+1,200,0,0)
40 X=RND(170) : Y=RND(150)
50 POS. I,X+I*10,Y+I*10
60 M. I
70 A=CR. (I) : P. I"#=";A
80 N.
90 PAU. 300 : G. 10
```

RUN 屏幕左上角依次打印出0#~7#八个卡通图案的各自的CRASH值。

上述程序只能测试一次相遇。若要测试某一个卡通图案在全部运动过程中与其相遇的所有卡通,可将上述程序改动如下:

```
(1)删去70行
(2)加85行
85 A=CR. (7);P. "7#CRASH="A;
IF M. (7)=-1 T. 85
```

RUN 在屏幕左边垂直打印7#CRASH=与其相遇的卡通编号或-1。读者可清楚地看到,7#卡通太空堡垒与其它卡通相遇时7#CRASH=后由-1变为相遇的卡通编号。

五、操纵器控制语句(STICK、STRIG)

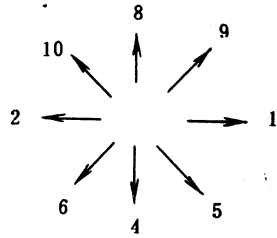
随机手册中仅介绍了两条操纵器语句,没有介绍如何使用操纵器控制卡通图案运动,而且只介绍了STICK语句的四个方向控制,遗漏了另外四个斜方向的控制。因此,使大多数使用者对操纵器如何控制卡通束手无策,不知从哪里下手编制程序,本文通过示例介绍使用方法。

1. 操纵器方向控制语句(STICK)

STICK 简写 STL.

STICK语句可使用操纵器对卡通图案进行八个运

动方向的控制。在八个方向的取值为



2. 操纵器功能控制语句(STRIG)

STRIG 简写 STRL.

随机手册中对STRIG的简写STR.是错误的,请读者更正。

有关这两个语句的说明,请参阅随机手册,现通过示例,介绍如何使用操纵器控制卡通图案运动。

例22 用操纵器控制玛丽哥哥运动。

```
10 CLS;SP. O. : X=35 : Y=130 : I=0
20 K=STL. (0) : IF K=0 T. 20
30 IF K=1 T. I=3
40 IF K=2 T. I=7
50 IF K=4 T. I=5
60 IF K=8 T. I=1
70 IF K=9 T. I=2
80 IF K=5 T. I=4
90 IF K=6 T. I=6
100 IF K=10 T. I=8
110 DE. M. (0)=SP. (0,I,1,2,0,0)
120 POS. 0,X,Y
130 M. 0
140 IF M. (0)=-1 T. 140
150 X=XP. (0) : Y=YP. (0) : G. 20
```

RUN 用1#操纵器可控制玛丽哥哥向各个方向运动。(待续)

邮 购 信 息

电子工业出版社于一九九二年元月十四日在北京人民大会堂举办首发式,隆重推出全国首次家电维修技术精华大奖赛的丰硕成果——《家电维修技术精华丛书》(1-10)。欢迎广大读者选购。

书名及邮购价(含邮挂费)

- ①收音机 8.30元②收录机、组合音响 8.70元③黑白电视机 14.40元④彩色电视机(上、下册)30.00元
- ⑤录像机、摄像机 8.50元⑥卫星电视接收机、共用天线 6.20元⑦电子表、电度表、万用表 6.20元⑧游戏机、家用微电脑、计算器 6.00元⑨家用制冷、空调设备 17.80元⑩小家电 8.70元

邮购办法:1. 邮局汇款请寄北京万寿路173信箱电子工业出版社发行部邮购科(汇款单附言栏内请注明购书清单)。2. 银行汇款:单位名称: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 电话:813693
注:购全套书邮购价110元。



维修经验谈

NP125 型复印机中一个易被忽视的故障

常州 102 医院(213003) 许 虞

NP125 型是广东湛江复印机厂生产的一种性能稳定,质量较好的复印机,其使用寿命长,操作维护简单,在社会上有相当的一批用户。但是,在使用了 7—8 万张以后,常会出现“E4”故障,其现象为:复印机一启动,操作台就出现“E4”故障字样,同时左边依次出现“卡纸”、“缺少墨粉”、“缺纸”等图形显示,机器死锁,无法运行。当把电源插头断开再重新插上或将前侧盖板拉开再合上,机器偶能运行一下,或中途再次停止,重复上述过程,仍然如此。

对此故障,开始感到十分棘手,因为这常常是复印机的交流电路板或直流电路板出了故障。为此,笔者花了许多时间对这两块电路板进行了仔细的检查,发现各电路工作点均正常,集成电路也没什么问题,反复检

查了好几遍,结果均为正常,而故障仍然存在。最后又查看各动态信号,发现复印机的时钟脉冲信号 CPS 幅度很小,有时干脆没有,对此信号顺藤摸瓜,最后终于发现在复印机后侧面中部靠近硒鼓处,有一小块与交流电路板用电线相连的转轮式光电时钟脉冲发生器,由于长时间的使用,光电发射与接收装置上已积满了许多灰尘,光电脉冲无法正常产生,经用干净毛笔掸去灰尘,装好试用,故障立即排除,恢复正常,真是踏破铁鞋无觅处,得来全不费功夫。因此当 NP125 复印机出现“E4”故障时,不妨先检查一下光电时钟脉冲发生器,如果不是该问题,再检查交、直流电路板或送修理部门修理,这样可以少走弯路。

PC—88 键盘分析与故障维修

西安空军工程学院六系计算中心(710038)

王耀亭

编者按:

PC—88 兼容机是香港两个公司分别组装的,习惯上称美国 PC—88 和日本 PC—88。因为各主要部件的来源不同,所以二者不仅在主机、终端、键盘的外形上不同,而且在实现功能的逻辑线路上也不同。它们的相同之处是:都是 16 位个人计算机,并且与 IBM PC 完全兼容。凡是在 PC—88 上运行的软件,均可以在 IBM PC 机上运行。基本配置如下:

128K 内存

打印机并行口

RS—232 串行口

两个 $5\frac{1}{4}$ 软盘驱动器

ROM BASIC(日本 PC—88 机在不用 MS—DOS 启动时,将自动进入 BASIC 状态。)

有单色和彩色适配器两种

键盘用 83 键

PC—88 机在操作使用上和 IBM PC 机完全相同。

键盘是人—机通信的通道,它通过键盘接口直接和系统交换数据信息。目前微机种类繁多,键盘接口电路也各有不同。IBM—PC/XT 的键盘接口采用 Intel8048 单片微处理器,IBM—PC/AT 的键盘接口采用一片 8 位单片微处理器 Intel8042/8742,PC—88 兼容

机中键盘接口是用一片 8748 单片微处理器来控制的。由于采用的微处理器和其它控制芯片不同,实现功能的逻辑线路有很大的差异,给维修带来很大的困难。我们有一批 84 年配置的 PC—88 兼容机,经过几年的使用,不仅主机、软驱、终端不断出现故障,而且键盘也时有故障出现。下面就 PC—88 兼容机键盘的工作原理和维修分析如下:

1. PC—88 键盘的逻辑线路

该机无随机逻辑线路,因维修需要,我们根据实物描绘线路如图 1 所示,大致可以表明逻辑关系。

2. 工作原理

键盘接口电路采用 8748 微处理器,矩阵译码电路由两块 MSM4051 和一块 MC74HC42 组成。MSM4051 为 8 选 1 模拟开关电路。它同 C511 和 C541 芯片,也称单八通道多路传输器。它有三个输入端 A、B、C,选中输出八种状态中的一种。无论那一路输出,③脚 COMMON 端同时有输出,接到 LM311 的③脚。两块 MSM4051 译码产生 16 条行扫描。MC74HC42 为 BCD 十进制译码器,用来产生 8 条列扫描。检测电路采用单电压比较器 LM311,②、③脚输入比较信号,通过放大,⑦脚输出的波形是②脚波形的检测放大,送到 8748 进行检查。8748 根据检测器送来的信号,确定有无键被按下,和同时按下的键数以及按键的位置。另外两块芯片是 7405 和 7408,它们起驱动控制作用。

3. 故障维修

[例 1] 按任何键主机不响应,也就是说,键盘没有输出信号到主机。

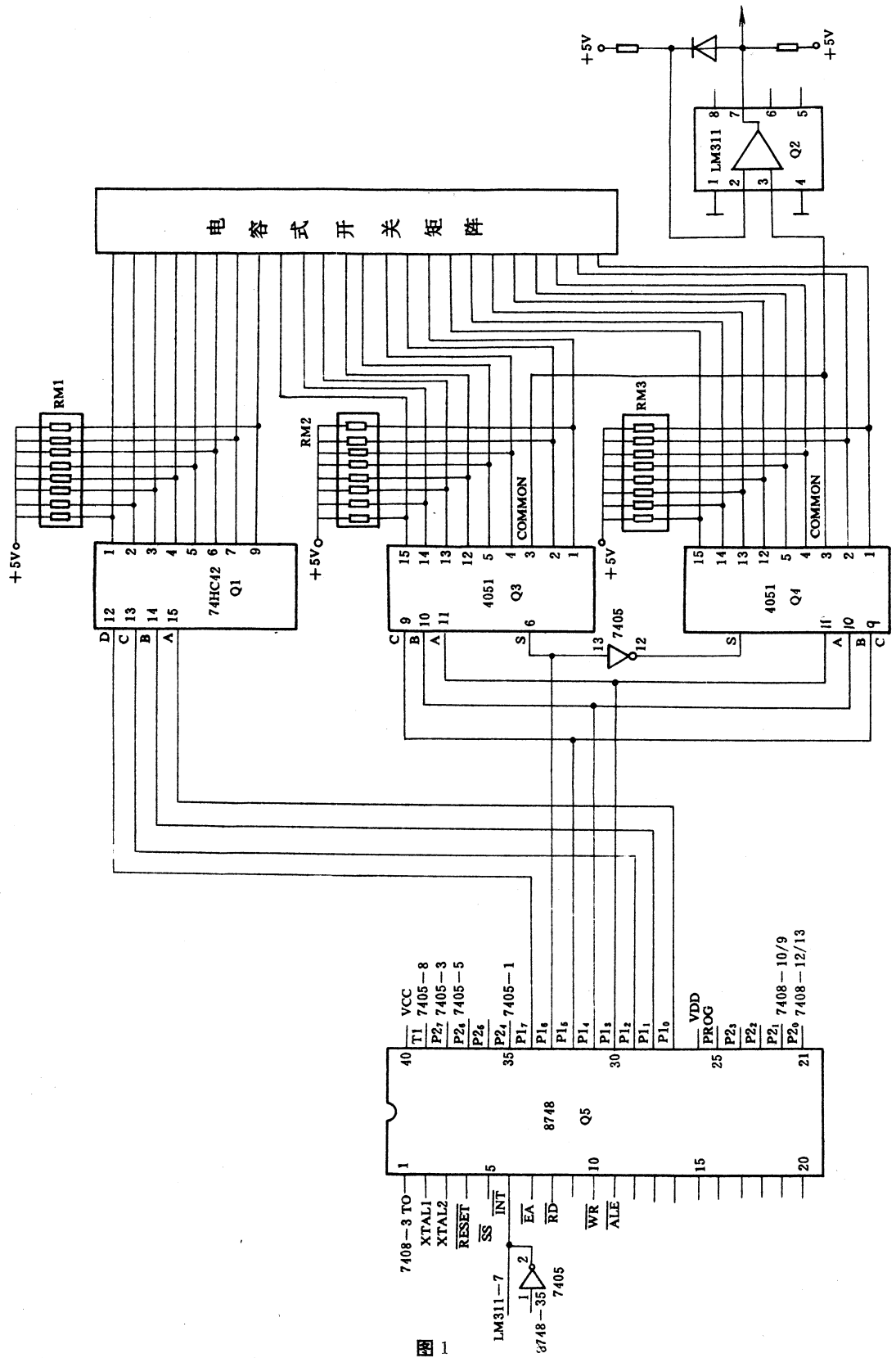


图 1

打开键盘,首先用示波器观察键盘接口,看有无+5V、KBDDATA、KBDCLK及KBDRESET信号。只要有开机或复位动作,KBDRESET就应该有由高电位—低电位—高电位的瞬变。KBDDATA和KBDCLK平时为高电位,只要有按键动作,就可以看到负脉冲串。若发现那个信号不正常,则可跟踪查找。

8748的②、③脚输入振荡脉冲,时钟频率为6MC,④脚为复位输入,低电平有效。用来对8748初始化。①脚T₀为编程方式控制端,从接口2通过7408的③脚送来。⑨脚T₁为测试端,从接口1通过7405的⑧脚送来。⑳、㉑、㉒、㉓脚接74HC42的12、13、14、15脚。⑳、㉑、㉒脚接两个4051的11、10、9脚。静态测量74HC42的各输出端均有3.76V,两个4051的输出端均为2.23V电平。按键时,74HC42输出端有负脉冲串。

根据各信号之间的关系,用示波器跟踪检查波形,

就会发现一些异常现象,特别是两块4051的输入输出波形、电平应相似。通过检查,发现静态时两块4051的电平基本相同,但在动态时一块4051输出电路上却叠加了负脉冲,影响了译码电路和检测电路。换了之后,电平和波形正常,键盘工作也正常了。

〔例2〕只有部分按键起作用,部分按键无反应。按Caps Lock和Num Lock键不起作用。

用示波器观察到,LM311的②脚电压为2V,是正常的,但③脚却在2V上叠加有正脉冲,不正常。7408的⑧脚平时高电平,但按Num Lock键后低不下来,灯不亮。断开Q₃的④脚,只有8个字符(8、9、I、O、K、L、M)不显示,其它正常。测量④与⑩脚之间的阻值只有110K左右,④脚有+5V电压,说明两脚间有短路。换了Q₃后,键盘工作正常了。

APPLE I 电源常见故障维修方法

天津肿瘤医院 邓满国

APPLE I 微机电源是一种高频转换的开关电源。一旦微机不能正常工作或出现信息误差,最大的可能是由于电源故障引起的,现将维修电源中排除故障的方法供读者参考。

见电路原理图。从图中可知CR₁、C₁、C₂、C₃是前级整流滤波环节。初级绕组L₁、R₁、C₄、Q₃,绕组L₂、R₁₁,CR₆组成振荡环节。Q₁、Q₂、CR₇、Q₄、AR₁是反馈控制环节。CR₂₀、R₂₇、R₂₁、Q₅是保护电路。以下就每一个电路上可能出现的问题加以讨论。

一、前级整流与滤波环节

若无电压输出,检查保险管已断,这多半是Q₃或CR₃已坏。如果经检查两管无问题,而换上新保险管开

机后保险管再断,那么应该检查CR₁中各P-N结电阻,若其正反向电阻相等或趋于零,则该管已击穿,若CR₁完好,则故障出现在前级整流滤波电路。

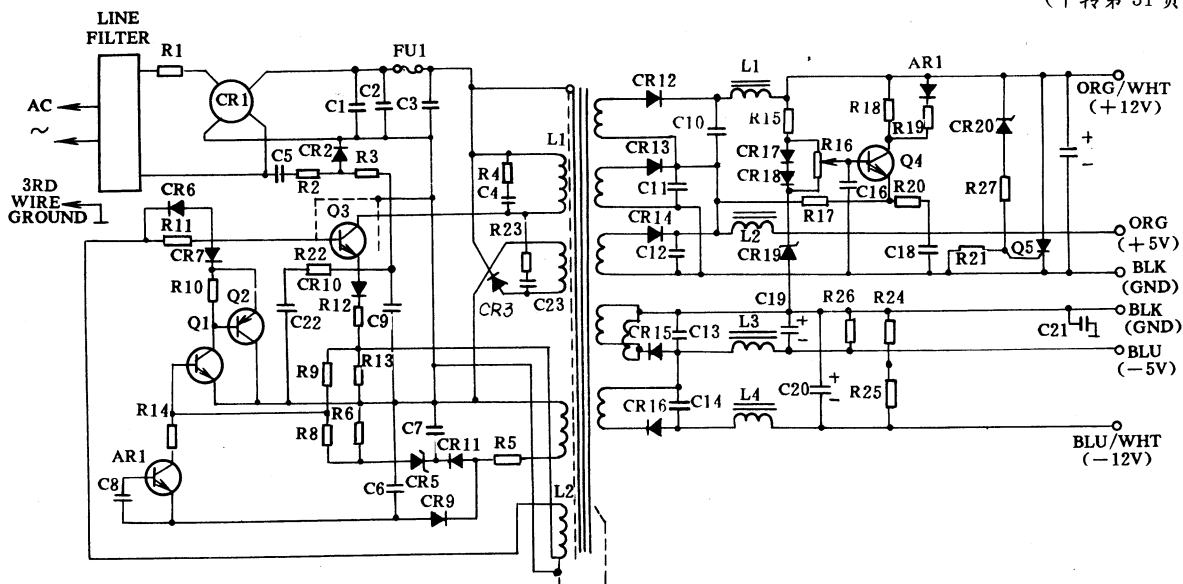
二、振荡环节

电路正常振荡,能在Q₃集电极上用示波器测出有600V大约20KHz交流电。若只有300V直流电压,说明振荡环节有问题。检查Q₃管各P-N结是否烧断。若检查是好的,再查一查R₁₂有可能呈高阻或开路状态,而R₁₂只有十几个欧姆,若更换R₁₂后,电路还不能起振,就要进一步检查振荡回路L₁和CR₆是否同时出现问题。以上这些情况是电路不能起振的原因。

三、反馈控制环节

±12、±5四组电压不准时,首先调正R₁₆。如果调正无效,就可能是光电耦合器件AR₁出现问题。它有问题就不能使电压过高或低的信号反馈给Q₁、Q₂来调整振荡管Q₃的输出电压幅度,而振荡环路上其它元件的损坏也可能使调整电压的R₁₆不起作用。

(下转第31页)



IBMPC/XT 机的软磁盘驱动器磁头校准程序

宁波市农行电脑室(315040) 范思尧

软磁盘驱动器是一个机电结合的精密产品。驱动器的磁头在长期使用中容易产生偏移,使磁头不能正常读写。校准磁头偏移一般应用校准盘和示波器进行测试校准;如没有该设备也可自编一个校准程序进行校准。本人在实际工作中用 DEBUG 的汇编功能自编了一个校准程序,并在 0520 系列机上多次成功地校准了驱动器的磁头。使用时只要对该程序中的有关寄存器值作适当设置(CH=磁道号 0-39,DH=磁头号 0 或 1,DL=驱动器号 0-1)就可以对不同驱动器,不同磁头,不同磁道分别进行校准,为了使用方便可以预先

把校准不同驱动器、磁头、磁道的程序分别命名,以便调用方便。校准时只要把一个完好的软盘放入要校准的驱动器,再调用相应的程序,当磁头偏移时,则屏幕显示 ERROR,当校准后则显示 PASS;调整磁头的一般顺序是先调整 0 面定磁头 39 道和 0 道,再调整 1 面动磁头 39 道和 0 道;当 39 道和 0 道均通过后,其它磁道一般均能通过,不一定再需校准;具体操作时需根据磁头定位机构的不同,采取不同的调整方法,以保证磁头物理位置的准确,以下为程序清单。

(本程序在 0520A,0520DH 等机上通过)

c>debug diskc.com

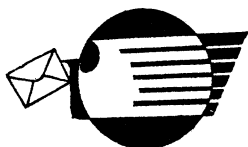
-d0100 01ff

```
5622:0100 52 45 53 45 54 2E 2E 2E-2E 2E 2E 2E 2E 2E 2E RESET.....
5622:0110 2E 2E 2E 52 45 41 44 2E-2E 2E 2E 2E 2E 2E 2E ... READ....
5622:0120 2E 2E 2E 2E 57 52 49 54-45 2E 2E 2E 2E 2E 2E .... WRITE...
5622:0130 2E 2E 2E 2E 2E 2E 56 45-52 46 2E 2E 2E 2E 2E 2E ..... VERF..
5622:0140 2E 2E 2E 2E 2E 2E 2E 2E-2E 2E 2E 2E 2E 2E 2E .....
5622:0150 2E 2E 2E 2E 2E 2E 2E 2E-2E 2E 2E 2E 2E 2E 2E .....
5622:0160 2E 2E 2E 2E 2E 2E 2E 50-41 53 53 2E 2E 2E 2E ..... PASS.
5622:0170 2E 45 52 52 4F 52 2E 2E-2E 2E 2E 2E 90 90 90 90 .ERROR.....
5622:0180 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
5622:0190 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
5622:01A0 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
5622:01B0 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
5622:01C0 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
5622:01D0 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
5622:01E0 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
5622:01F0 90 90 90 90 90 90 90 90-90 90 90 90 90 90 90 .....
```

-u0200 02cb

```
5622:0200 BB0001      MOV     BX,0100      5622:0227 B80030      MOV     AX,3000
5622:0203 E8B900      CALL    02BF        5622:022A 8EC0       MOV     ES,AX
5622:0206 B201       MOV     DL,01       5622:022C BB0000      MOV     BX,0000
5622:0208 B400       MOV     AH,00       5622:022F B402       MOV     AH,02
5622:020A CD13       INT     13          5622:0231 B008       MOV     AL,08
5622:020C F6C4FF     TEST    AH,FF       5622:0233 B527       MOV     CH,27
5622:020F 7409       JZ      021A        5622:0235 B101       MOV     CL,01
5622:0211 BB7101     MOV     BX,0171     5622:0237 B600       MOV     DH,00
5622:0214 E8A800     CALL    02BF        5622:0239 B201       MOV     DL,01
5622:0217 EB07       JMP     0220        5622:023B CD13       INT     13
5622:0219 90        NOP                    5622:023D 07        POP     ES
5622:021A BB6701     MOV     BX,0167     5622:023E F6C4FF     TEST    AH,FF
5622:021D E89F00     CALL    02BF        5622:0241 7409       JZ      024C
5622:0220 BB1301     MOV     BX,0113     5622:0243 BB7101     MOV     BX,0171
5622:0223 E89900     CALL    02BF        5622:0246 E87600     CALL    02BF
5622:0226 06        PUSH   ES           5622:0249 90        NOP
```

5622:024A	EB06	JMP	0252	5622:0293	B404	MOV	AH,04
5622:024C	BB6701	MOV	BX,0167	5622:0295	B008	MOV	AL,08
5622:024F	E86D00	CALL	02BF	5622:0297	B527	MOV	CH,27
5622:0252	BB2401	MOV	BX,0124	5622:0299	B101	MOV	CL,01
5622:0255	E86700	CALL	02BF	5622:029B	B600	MOV	DH,00
5622:0258	06	PUSH	ES	5622:029D	B201	MOV	DL,01
5622:0259	B80040	MOV	AX,4000	5622:029F	CD13	INT	13
5622:025C	8EC0	MOV	ES,AX	5622:02A1	07	POP	ES
5622:025E	BB0000	MOV	BX,0000	5622:02A2	F6C4FF	TEST	AH,FF
5622:0261	B403	MOV	AH,03	5622:02A5	750C	JNZ	02B3
5622:0263	B008	MOV	AL,08	5622:02A7	90	NOP	
5622:0265	B527	MOV	CH,27	5622:02A8	90	NOP	
5622:0267	B101	MOV	CL,01	5622:02A9	90	NOP	
5622:0269	B600	MOV	DH,00	5622:02AA	BB6701	MOV	BX,0167
5622:026B	B201	MOV	DL,01	5622:02AD	E80F00	CALL	02BF
5622:026D	CD13	INT	13	5622:02B0	EBD2	JMP	0284
5622:026F	07	POP	ES	5622:02B2	90	NOP	
5622:0270	F6C4FF	TEST	AH,FF	5622:02B3	BB7101	MOV	BX,0171
5622:0273	7409	JZ	027E	5622:02B6	E80600	CALL	02BF
5622:0275	BB7101	MOV	BX,0171	5622:02B9	EBC9	JMP	0284
5622:0278	E84400	CALL	02BF	5622:02BB	90	NOP	
5622:027B	EB07	JMP	0284	5622:02BC	90	NOP	
5622:027D	90	NOP		5622:02BD	90	NOP	
5622:027E	BB6701	MOV	BX,0167	5622:02BE	90	NOP	
5622:0281	E83B00	CALL	02BF	5622:02BF	B90800	MOV	CX,0008
5622:0284	BB3601	MOV	BX,0136	5622:02C2	8B17	MOV	DX,(BX)
5622:0287	E83500	CALL	02BF	5622:02C4	B402	MOV	AH,02
5622:028A	06	PUSH	ES	5622:02C6	CD21	INT	21
5622:028B	B80020	MOV	AX,2000	5622:02C8	43	INC	BX
5622:028E	8EC0	MOV	ES,AX	5622:02C9	E0F7	LOOPNZ	02C2
5622:0290	BB0000	MOV	BX,0000	5622:02CB	C3	RET	



读者联谊

普及型 PC 个人用户软件交流联谊 活动问题解答(三)

北京中国农科院计算中心(100081) 王路敬

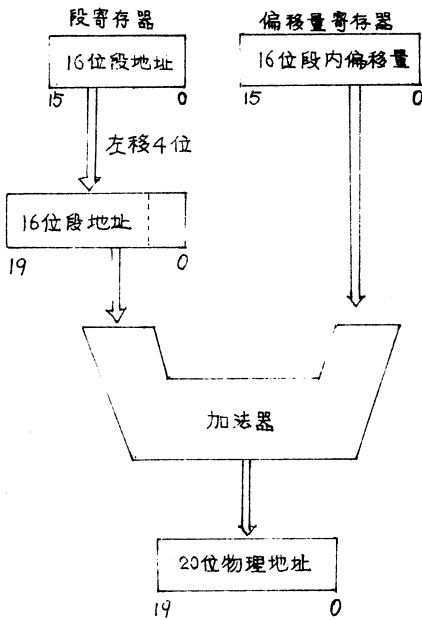
6. PC 机程序中编排的地址和信息在存储器中实际存放地址有何不同?

PC 机的 CPU 8088 具有 1 兆字节的存储寻址空间,需要 20 位地址,而其寄存器都是 16 位的,为此引入了分段的概念。CPU 对内存进行访问时,把内存分成若干段。段可以看成是一个连续的存储区,每一段最多 64K,必须从一个能被 16 整除的绝对地址开始,这 16 整除的地址称之为节,亦就是说,段地址必须以节为边界。

在具有地址变换机构的微机中,允许程序中编排的地址和信息在存储器中实际存放的地址不同,在 8088 系统中每个存储单元认为有两类地址:程序中所涉及到的地址是逻辑地址,一个逻辑地址由两部分组成:段地址和段内偏移量,对于给定逻辑地址的任何一

个存储单元来说,段地址决定了该段的第一个字节的位置,段内偏移量则是这个存储单元相对于该段首字节的距离。逻辑地址表示形式是:××××:××××,前面 4 位十六进制数为段地址,后面 16 位二进制数为段内偏移量。

8088 与存储器之间的所有信息交换却要使用物理地址。由于 8088 具有 20 根地址线,存储器的物理地址是一个 20 位二进制数值(00000H—FFFFFH),所以它与 1M 字节存储空间中和每个存储单元是一一对应的。20 位物理地址描述信息在存储器中实际存放的地址,也叫绝对地址。它的形成是首先将段寄存器内容左移 4 个二进制位,右边补上 4 个“0”形成 20 位段起始地址,再与 16 位段内偏移量相加,从而形成 20 位的存储器物理地址如下图所示:



图

7. PC 机系统中插入 8087 有什么好处? 如何处置?

在 PC 机系统中,为了增强其性能,可插入 8087 协处理器与 8088 一起工作,以提高数值运算的速度。

8087 协处理器有如下五个主要特点:

(1)8087 与 8086/8088 微处理器一起工作控制它们之间的数据传送。8087 专门执行数值运算,对同样的数值,它的执行速度比 8086/8088 快约 10—100 倍,甚至更多。

(2)8087 有 8 种数据类型:8 位、16 位、32 位、64 位字长的整数;32 位、64 位、80 位字长的浮点数;18 位十进制数,它的浮点数符合 IEEE 浮点标准。

(3)具有 6 类数据传送、算术运算、超越函数、常数、比较、处理器控制共 68 条指令。

(4)具有可寻址的 8 个 80 位专用寄存器堆栈。这 8 个 80 位寄存器,用于保存内部数据。

(5)具有 6 种内部异常功能处理。8087 捕捉各种各样的运算错误,当出现错误时,8087 产生一个异常条件,这些异常条件是:无效操作、除数为 0、下溢出、未规格化操作数,上溢出,结果不精确。8087 捕捉到异常时,产生中断请求,要求用户程序进行处理。

PC 机运行时,8087 监视 8088 的工作,每当执行 8087 的数值运算指令时,8087 立即接过控制,执行指令规定的操作。操作结束后,交出控制,让 8088 继续运行。

在主机系统板上有一个 8 位系统配置开关 DIP,用来选择和设置系统配置的情况。只要将 DIP 开关的第 2 位开关拨到 OFF 位即可插入 8087 工作。(对长城 0520 把 8087 协处理芯片按缺口方向插入,再将第 1 号开关拨到 OFF 位置即可。)

8. PC 机系统的适配器是指一种什么设备?

PC 机通过各种外部设备与外界通信或交换数据,这种过程称为输入/输出。PC 机的外部设备主要有:键盘、显示器、打印机、软盘驱动器、硬盘驱动器、终端设备及其他专用设备。这些设备有机械的、电子的、机电的,它们传送的信号可能是数字信号,也可能是连续变化的电流或电压信号,而且信号电平有高低,信号传送速度也各不相同,当把这些外围设备接到 PC 机时,就需要相应的接口电路,亦称为适配器。

适配器的任务是把外围设备送往 8088 的数据转换成 8088 适用的格式,或把 8088 送往外围设备的数据转换成与外围设备相容的格式;向 8088 提供外围设备的状态信息,如设备“准备好”、忙或闲、缓冲器“满”或“空”等等。协调 8088 与外围设备处理数据速度上的差异。一般说来,每种外部设备都有其特点,必须用专门适配器实现 8088 与外围设备间信号的转换与匹配,并提供适当的时序与控制信号,进行数据缓冲、同步协调,传递外围设备的状态信号以及可编程设置工作方式等功能。

9. PC 机 8088 与外围设备间数据传送有哪几种方式?

一般有三种方式:程序控制的数据传送方式;中断传送方式;DMA 方式传送数据。

程序控制下传送数据时,在执行数据传送操作前,8088 先要检测外部设备的状态,如果外部设备正在工作(“忙”),则程序继续检测状态,8088 进行等待。如果设备准备好下一次传送,8088 就执行一次传送操作,如此循环下去,直接把数据传送完。这种传送的缺点是 8088 要不断查询外部设备是否“准备好”;当外部设备准备好时,8088 等待,无法执行其他操作,这就浪费了时间,对于低速外部设备,如键盘,其传送速度很慢,因而降低了 8088 的效率。为了提高计算机的效率,8088 与外部设备间传送数据通常采用中断方式或 DMA 方式。所谓中断是指计算机在正常执行程序的过程中,由于种种原因,使 CPU 暂时停止当前程序的执行,而转去处理临时发生的事件,处理完后,再返回去继续执行被暂停的程序。换言之,在程序执行过程中,插入另外一段程序运行,这就是中断。使用中断技术,使得外部设备与 CPU 不再是串行工作,而是分时操作,从而大大提高了计算机的效率,随着计算机的发展,中断不断赋予新的功能。中断系统已成为现代计算机不可缺少的组成部分。高效率的中断系统,可以最少的响应时间和内部操作处理所有外部设备的服务请求。PC 机利用中断功能,可使 8088 与外部设备并行工作,消除 8088 的等待时间,提高 8088 的利用率,同时 8088 可以同时管理多个外部设备的工作,提高输入输出数据的吞吐量。除此之外,还可利用中断技术进行实时处理,故障处理等。

PC 机 8088 具有简单而多用途的中断系统,每个中断都规定一个中断类型号(代码 0—255),8088 根据中断类型号从中断向量表中取中断服务子程序入口地址,转去执行中断处理子程序。

中断技术解决了高速 8088 与低速外部设备之间的矛盾,但对于高速外部设备,它就无能为力了。为了快速传输大量数据,PC 系统中采用存储器直接访问技术,简称 DMA。用 DMA 方式传送数据时,在存储器和外部设备之间,直接开辟高速传输数据的通道,不需要 CPU 8088 的干预,因而使数据传送速度达到最快。利用 DMA 方式传送数据时,数据的传输过程完全由硬件控制,这种电路叫 DMA 控制器。该控制器向 8088 申请 DMA 传送,在 8088 允许 DMA 工作时,处理总线控制权的转交避免因进入 DMA 工作而影响 8088 的正常活动或引起总线竞争,在 DMA 期间,管理系统总线,控制数据传送,确定数据传送的起始地址和数据长度,修正传送过程中的数据地址,在数据块传送结束时,给出表示 DMA 操作完成的信号。

10. IBM-PC/XT 及其兼容机系统内存存储器空间是如何分配的

IBM-PC/XT 及其兼容机系统具有 1MB 存储空间。用十六进制表示的地址范围是 00000H—FFFFFH。这 1MB 存储空间划分为三个区域:RAM 区;保留区和 ROM 区。

存储器空间的低区,0—640K 的存储区域是读写存储器 RAM,对于 640KB 的系统 RAM,一般在系统板上安装 256KB,剩下的部分安装在 I/O 插槽中的存储器扩展板上。系统板上安装的 RAM 容量由系统配置中第 3 位和第 4 位状态来决定。运行 DOS 2.0, RAM 容量至少要有 128KB。接下来的 128KB 存储空间是系统保留用作字符/图形显示缓冲区的区域。对于不同的显示适配器,实际使用的存储器区域各不相同。IBM 单色字符显示适配器使用的缓冲区容量是 4KB 地址,为 B0000H—B0FFFH。IBM-PC/XT 中的彩色字符图形显示适配器使用的缓冲区容量是 16KB,地址为 B8000H—BBFFFH。如果使用高分辨率的显示适配器则显示缓冲区要使用更大的存储区域。

存储空间的最末 256KB 是系统的 ROM 区,这个区里安装的存储器都是只读存储器,其中前 192KB 的区域放系统中的控制 ROM 或 ROM 扩展板。一般高分辨率显示适配器的控制 ROM 安放在 C800H—C7FFFH 的区域内;硬盘驱动器适配器的控制 ROM 安放在 C800 开始的区域内。如果要安装用户固化在 ROM 中的程序,可以使用 ROM 区中尚未使用的地址。系统最后 64K 的存储器区域是基本系统 ROM 区。IBM-PC/XT 系统板上安装了 40KB 的基本 ROM,其中包括基本输入/输出系统 BIOS (8KB) 和 ROM BASIC (32KB), GW0520A 安装了 8KB 的基本 ROM。

计算机中的读写存储器 RAM 中存放的信息在停电关机后就消失了,为了使计算机初始化和进入正常工作状态,在计算机中必须安装不会因停电而丢失的初始化程序。初始化程序放在只读存储器中,加电后自动执行,对计算机进行初始化,再引导各种软件工作。

对于 IBM-PC/XT 系统板上安装的基本 ROM 40KB 占用地址是 F6000H—FFFFFH。其中固化 BASIC

占用地址 F6000H—F6FFFH;基本输入输出系统 BIOS 占用地址 FE000H—FFFFFH。BIOS 的功能对 PC/XT 系统进行初始化,同时也是高层软件和硬件之间的接口,它具有如下的功能:

- (1) 系统冷启动和热启动;
- (2) 系统自测试;
- (3) 基本外部设备的输入输出驱动程序: CRT 显示器(单色和彩色)、键盘、打印机、软磁盘和异步通信接口等,这些驱动程序都要通过中断调用;
- (4) 硬件中断管理程序;
- (5) 系统配置分析程序;
- (6) 字符图形发生器;
- (7) 一天的时钟管理程序;
- (8) DOS 引导程序。

系统板的 40KB 基本 ROM 信息装在两块 ROM 芯片中。一块是 8KB 的芯片,内装固化 BASIC 的前 8KB;另一块是 32KB 的芯片,内装固化 BASIC 的后 24KB 和 BIOS 8KB。

(待续)

中国微计算机单片微机 学会单片微机函授班 (第二期)招生简章

一、宗旨

为适应“八五”计划要求,以高科技促国民经济的发展,加速单片微机向社会生产和生活各个领域的渗透。根据学会理事会决议,在总结第一期函授的成功基础上,特在全国范围内举办第二期单片微机并用,费用少、见效快的单片微机函授班。

二、教学形式

以 MCS-51 单片微机为学习内容,通过自学函授教材,阅读指定的辅导材料,要求人手一机,理论联系实际,定期完成指定的作业,习题和实验。学习中学员可随时反馈疑难问题,并定期作书面答疑。

三、函授对象

凡知识更新各类工程技术人员,科教人员,管理干部,解放军军地两用人才,向高科技进军的技术工人,职业学校的师生,科技兴厂的乡镇企业技术骨干,自谋职业的知青等均中报名参加。

四、招生事项

报名日期:自简章公布之日起即可报名
报名办法:邮寄 2 元报名费,索取报名表、章程等。
报名时务请写明通讯地址、邮编号码及联系方式。

开学日期:1992 年 5 月

报名地点:南京市 东南大学计算机系

联系人:孙育才,邮编:210018

五、办学单位

主办:中国微计算机单片微机学会

机电电子工业出版社《电子与电脑》编辑部

承办:南京电脑应用协会单片微机专委会



一九九二年

总期第85期

電子與電腦

• ELECTRONICS AND COMPUTERS •

目 录

- 综述 •
 - 引人注目的微型机 戴伟高(2)
- PC 用户 •
 - 用 24 针打印机打印 CAD 图形 卢耀志(4)
 - 编制菜单程序的辅助程序@. COM 张 红(7)
 - 格式文件与 SET FILTER TO 命令的结合 麦红波(8)
 - Azusa/2708 病毒与介质描述 何悦荣(9)
 - MS FORTRAN 程序的菜单设计技术 曹国钧 王 健(10)
 - 如何制作病毒标本 梁宇舫(10)
 - 用 dBASE III MEMO 字段开发试题库系统 陈卫平(11)
 - 关于《dBASE III 过程文件加密的发现》质疑 张维仁(11)
 - 在单台驱动器上如何作软盘备份 杨栋林(12)
 - 一种不打印数值零的方法 曹和平(12)
 - 交互式的 DEL 命令 莫 琳(12)
- 学习机之友 •
 - 为 APPLE DOS 增加多目录功能 秦燕军(14)
 - 验证四色猜想的 BASIC 程序 陈庆祥(15)
 - 磁盘不抹掉文件的 35 轨改 40 轨方法 钱仕宏(16)
 - 解拆 C—WORDSTAR 五笔字型 陈晓乐(16)
 - BASIC 中“宏代换”的实现 许 斌(17)
 - 改进的 UNNEW 程序 张 亨(18)
 - CEC—I 中华学习机磁带游戏软件的解密 胡发新(18)
 - 也谈 CEC—I 全功能造字 覃 敏(19)
 - ProDos 磁盘操作系统入门(续) 廖 凯(20)
- 6050 机器语言讲座 •
 - 第四章 6502MPU 的指令系统 朱国江(22)
- 初级程序员级软件水平考试辅导 •
 - 操作系统及 PC-DOS 宋丹颖(27)
- 学用单片机 •
 - 实现单片机最小系统与 PC 机的通信 张培仁(30)
- 学装微电脑 •
 - 红外线遥控室内温度 易齐干(34)
- 电脑巧开发 •
 - Z80 单板机与打印机巧相连 张慧成(37)
 - 降低 MP-1 电脑的功耗 邓文超(40)
- 电脑游戏机 •
 - 第三章 F BASIC 的画面控制语句 于 春(41)
- 维修经验谈 •
 - 中华学习机电源故障维修经验谈 汪 晖(44)
 - 袖珍计算机 PC 型 PB 型系统 RAM 扩展模块
常见故障与维修 朱 杰(46)
- 读者联谊 •
 - 普及型 PC 机个人用户软件交流联谊活动问题解
答(四) 王路敬(48)

封一:绘图仪
封二:病毒大事年表
封三:病毒大事年表(续)
封四:电子工业出版社图书

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部
(北京 173 信箱 邮政编码:100036)

印刷:北京三二〇九厂

国内总发行:北京报刊发行局

国内统一刊号:CN11-2199

邮发代号:2-888

国外代号:M924

出版日期:每月 23 日

主编:王惠民 副主编:王昌铭

责任编辑:施玉新

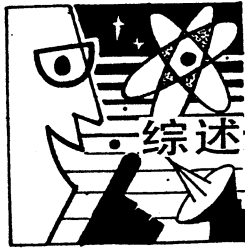
订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:0.95 元



引人注目的微型机

武汉 70005 信箱(430070) 戴伟高

一、引言

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

二、产品的主要特征

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

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

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

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

三、第三方产业支援

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

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

四、各种各样的 I/O 模块

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

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

五、结束

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

简 讯

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

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

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

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

光盘图书通向未来

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

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

薄一波同志到会并为此项高新技术题词:

“光盘图书,通向未来。”

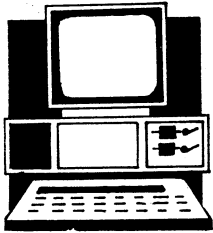
最新 FORTH 版本 PC35

中国 FORTH 应用研究会迄今已成立五年,一贯致力于推广第四代高级计算机语言 FORTH,拥有包括最新版本 PC/FOTH35 在内的多种 PC 及 APPLE 版本的软件以及各种 FORTH 资料,并经常举办函授及各种培训班。

欢迎有志之士及爱好者与我们联系。

地址:北京 西北北三条 10 号

邮编:100034 电话:655661



PC 用户

用 24 针打印机打印 CAD 图形

长春 东北勘测设计院物探计算室(130062)卢耀志

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

一、方法原理

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

1. 获得 CAD 图形数据的方法

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

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

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

2. 生成图形点阵数据

上述读取工作得到的是一对对坐标值,也就是一

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

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

(1)把从 PLT 中读出的坐标数据化成以点为单位的坐标值。

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

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

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

$$X_i = X + DX;$$

$$Y_i = Y + DY;$$

如此计算 N-1 次便将此线分解完毕。

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

```
TYPE POI = ^ DIM
DIM = ARRAY[0..1999] OF BYTE;
VAR P; ARRAY[0..209] OF POI;
```

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

```
P[Y DIV 8]^ [X]; = P [Y DIV 8]^ [X] OR
ER[Y MOD 8];
```

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

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

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

```
.....
```

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

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

3. 打印

点阵数据生成后,就可顺次地读出送给打印机打

印了。见 PLOT 子程序。

4. 裁剪与拼接

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

二、操作步骤

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

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

(2)在 CAD 子目录下键入

ACAD TU P8 回车

运行完后再键入

LQ1600 文件名 回车

假设文件名指定为 ABC,则运行完后准备好打印机,再键入

COPY ABC?.PIC PRN 回车

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

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

最后几点说明:

(1)本程序在 IBM/PC XT AT 及其兼容机上通过运行。

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

(3)CAD 应将绘图机配置成 DMP52MP。

附 1: P8.SCR 命令文件清单

```
3 w 0.00,0 23.70,28 y y * 1 * 0 * 16 x y m 0,0
max n 0.16 n n 10=1 p1
3 w 23.70,0 47.40,28 n p2
3 w 47.40,0 71.10,28 n p3
```

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

附 2: LQ1600.PAS 源程序清单

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

```

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

```

```

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

```

(下转13页)

编制菜单程序的辅助程序@.COM

湖北省公路局科研所(430030) 张红

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

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

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

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

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

参考文献

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

```
色,  
;3:黄色,4:兰色,5:洋红,6:青色,7:白色  
CR EQU 0DH  
SPACE EQU 20H  
SLASH EQU 2FH  
DOLLAR EQU 24H  
DISP MACRO MESSAGE  
MOV DX,OFFSET MESSAGE  
MOV AH,09H  
INT 21H  
ENDM  
CODE SEGMENT PARA PUBLIC 'CODE'  
ASSUME CS:CODE,DS:CODE,SS:CODE,ES:CODE  
ORG 100H  
START PROC NEAR  
JMP MASTER  
FLAG DB 0  
CURSORLOC DB 1BH, '['  
DB 8 DUP(?)  
COLOR0 DB 1BH, "[0m", "$"  
COLOR DB 1BH, "[3;4 m", "$"  
MESS DB 40 DUP(?)  
MASTER: CLD  
MOV SI, 80H  
XOR AX, AX  
LODSB  
CMP AX, 0  
JE EXIT  
MOV CX, AX  
LABEL1: LODSB  
CMP AL, SPACE  
JNE LABEL2  
LOOP LABEL1  
LABEL2: JCXZ EXIT  
LEA DI, CURSORLOC+2  
MOV CL, 0  
LABEL3: CMP AL, '0'  
JB EXIT  
CMP AL, '9'  
JG EXIT  
STOSB  
INC CL  
LODSB  
CMP FLAG, 1  
JE LABEL4  
CMP AL, ','  
JNE LABEL3  
CMP CL, 2  
JC EXIT  
MOV AL, ';'   
STOSB  
MOV CL, 0  
MOV FLAG, 1  
LODSB  
JMP LABEL3
```

```

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

```

```

CODE      ENDS
END START
C>EDLIN XTGL0.BAT
End of input file
* L
1: * ECHO OFF
2:CLS
3:@3,27 SAY"总系统目录"IN 2/3
4:@6,28 SAY "1...计划管理系统" IN 3/2
5:@08,28 SAY "2...财务管理系统" IN 2/5
6:@10,28 SAY"3...人事管理系统" IN 6/3
7:@12,28 SAY "4...工资管理系统" IN 4/2
8:@14,28 SAY "5...结束系统操作" IN 6/5
9:ECHO ON

```

格式文件与 SET FILTER TO 命令的结合

云南昆明凉亭轧钢厂(650215)麦红波

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

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

格式文件是用来生成一个屏幕格式,它只能用 APPEND, INSERT, CHANGE, EDIT 和 READ 命令来激活,当格式文件被激活后,所生成的屏幕格式具备了全屏幕编辑功能,尤其是可用 <PgUp> 键读出上一个记录,用 <PgUp> 键读出下一个记录,<Esc> 键结束输入操作,这比用命令语句来实现上述功能要方便,简化了程序。注意示例中有一条 SET CLEAR OFF 命令,它的作用是在敲键读记录时,屏幕不闪动,即显示新记录时,系统不清屏重显屏幕格式。(注意 dBASE III 中没有 SET CLEAR OFF 命令)

示例

1. 水电费数据库结构(SDF.BDF)

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

2. 格式文件(SDF.FMT)

@6,30,say '水电费输入表'

(下转 13 页)

Azusa/2708 病毒与介质描述

华南师范大学计算机系(510631) 何悦荣

一、引言

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

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

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

二、Azusa 病毒的特征

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

三、Azusa 病毒干扰软盘操作的原因

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

```
Not ready error reading drive A
```

```
Abort, Retry, Fail?
```

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

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

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

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

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

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

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

四、Azusa 病毒的解毒与免疫方法

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

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

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

注:本文提及的软盘为 5.25 英寸 360K 磁盘。

MS FORTRAN 程序的 菜单设计技术

重庆医药设计院 曹国钧
渝州大学图书馆 王 健

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

一、屏幕的光标定位及清屏

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

1. 清屏语句:

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

其中 CHAR(27)就是控制码 ESC 的 ASCII 代码。

2. 光标定位语句:

```
WRITE(*,'(1X,A1,1H[,I2,1H;,I2,1HH)')  
CHAR(27),10,20
```

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

```
SUBROUTINE CLSLOCA(L,L1,L2)  
IF(L.EQ.0) WRITE(*,'(1X,A1,3H[2J]')  
CHAR(27)  
IF(L1.LT.10.AND.L2.LT.10) THEN  
WRITE(*,'(1X,A1,1H[,I1,1H;,I1,1HH)')  
CHAR(27),L1,L2  
ELSE IF(L1.GE.10.AND.L2.GE.10) THEN  
WRITE(*,'(1X,A1,1H[,I2,1H;,I2,1HH)')  
CHAR(27),L1,L2
```

ELSE

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

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

ENDIF

RETURN

END

二、汉字大小显示程度:

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

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

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

```
WRITE(*,'(1X,A1,5H+["@L,A,2H"]')  
CHAR(14,'重庆医药设计院工程经济室')
```

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

```
C>MODE BW80
```

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

最后要说明的是:

1. 对于彩色图形显示器,还考虑汉字的颜色及屏幕颜色;

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

如何制做病毒标本

华南师大计算机 90 级(510631) 梁宇翀

目前全世界约发现近三百种计算病毒。计算机病毒成为社会的新“公害”。

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

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

病毒。

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

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

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

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

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

用 dBASE III MEMO 字段开发试题库系统

长沙中建五局中专 陈卫平

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

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

至于图形处理，我们也可采取同样手段，在 BASIC

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

本方法在 LX-286 机和 LQ-1600 打印机下运行成功。

关于《dBASE III 过程文件 加密的发现》质疑

浙江嘉兴民丰造纸厂(314000) 张维仁

《电子与电脑》91 年第 9 期刘人伟同志《dBASE III 过程文件加密的发现》一文存在着多处问题。①文中提到的 CMIS V1.0(88.10) 计量管理软件名称应叫作“工业企业通用计量管理信息系统”，（简称 CMMS）是由国家技术监督局组织开发和发行的，而 CCDOS V4.0 才是由机电部六所开发的，文中将两者弄混了；② CCDOS V4.0 与 2.13、UCDOS 及 XSDOS 等中文操作系统一样，并不能提高 dBASE III 程序的运行速度；③文中提及的 DB05.PRG 等其实并非过程文件，而是一般的命令文件。CMMS 的作者为防止内存配置较小的计算机装载程序时内存不够，在编制程序时除 V2.0 的 XZ36.PRG 外，均未采用过程文件这一形式；④ CMMS V1.0 是一个完全开放的系统，无一程序进行过加密处理。进入时需将 #1 盘插入 A 驱动器是所采用的 dBASE III 所致，并非程序编制时进行的加密处理。程序并没有加密过，用户完全可以根据需要自行修改，但从 dBASE III 换到 FOXPLUS 环境下执行确是存在文中所提及的问题，到底是为什么呢？原因是当初作者编写 CMMS V1.0 时是使用 WS 的格式文件（或者叫文本文件，进入 WS 时选 D）编写的。WS 的格式文件每满一页均自动插入一个 ASCII 码 8AH(138) 作为分页符，这就是刘文用 DEBUG 所见的 0D8AH。dBASE III 环境下命令文件中这一分页符在执行时是予以忽略的，并不会引起出错，用 WS 或在圆点提示符下用 modi comm<命令文件名>查看时不能显示出 8AH 这一 ASCII 码，但可看到文件的全部内容，说明文件并未加密。在 FOXPLUS 环境下则将这一字符视为出错，这就是 dBASE III 与 FOXPLUS 的一个不同之处。明白了 8AH 产生的机制，要除去它就相当简单了，不必费脑筋去编制程序，只要把这些命令文件用 WS 的编制非格式（文本）文件（选 N）调入后马上重新存一次盘，即可将这些每隔 55 行出现一次的 8AH 去掉了。④文中提及 CMMS V1.0 有些命令文件执行速度慢并可能出错，是由于 V1.0 编制仓促不够完善所致。因此换用 FOXPLUS 固然可

（下转 13 页）

在单台驱动器上如何作软盘备份

太原太行仪表厂研究所(630006) 杨株林

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

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

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

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

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

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

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

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

或者用全盘拷贝命令作整盘的复制

```
DISKCOPY A: D:
```

同样,DOS 会在需要时提示用户更换盘片。



交互式的 DEL 命令

国防科技大学计算机系 90 级(410073) 莫琳

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

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

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

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

一种不打印数值零的方法

蚌埠石油站(233040) 曹和平

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

```
IIF(<逻辑表达式>,<表达式 1>,<表达式 2>)
```

利用此函数可代替 IF...ENDIF 语句,方便实现。

例如:

∴

```
P6=STR(独子女费,6,2)
```

```
P6=IIF(Val(P6)=0,“————”,P6)
```

∴

```
@&H,B Say“.....|独子女费|.....”
```

```
@&H+1,B Say“.....|&P6.|.....”
```

在打印时,当数值为 0 时,即以空格取代。

```

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

```

```

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

```

~~~~~  
(上接 6 页)

```

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

```

```

end else PLOT;
close(lst);
if ord(ch)=27 then goto 2
end;
2: if name<>'prn' then begin
   gotoxy(4,15);
   writeln('Printer plot command;copy',
           name,'?.pic prn');
   end;
end.

```

~~~~~  
(上接 8 页)

```

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

```

```

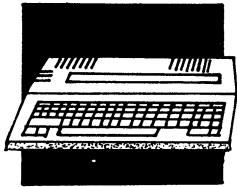
3. 主程序(SDF.PRG)
CLEAR
USE SDF
SET FILTER TO 部门='车队'
GO TOP
SET FORMAT TO SDF.FMT
EDIT
CLOSE DATA
SET FORMAT TO

```

~~~~~  
(上接 11 页)

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

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



## 学习机之友

# 为 APPLE DOS 增加多目录功能

上海番禺中学(200052) 秦燕军

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

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

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

### 一、使 CATALOG 命令允许携带 L 参数

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

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

```
CATALOG D2,L4
```

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

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

### 二、在 CATALOG 命令处理程序中修改 VTOC 表缓冲区

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

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

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

### 三、VTOC 表的恢复与磁盘使用状况的更新

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

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

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

### 四、使用 DOS 在磁盘的空区作为新的文件目录区

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

### 五、多目录功能 DOS 的实现

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

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

```
* B00E;4C D9 B6 ✓
* B6D9;C9 01 F0 0F 48 A9 11 8D
  BC B3 A9 0F 8D BD B3 68
  4C 52 B0 20 52 B0 AD D7
  B6 8D BD B3 AD D8 B6 8D
  BC B3 60 ✓
* ^ C ✓      用 CTRL-C 退出监控系统
```

```
]INIT HELLO ✓
```

至此,就产生了一个多目录功能 DOS 的磁盘

## 六、多目录功能 DOS 的应用

多目录功能 DOS 的用途很多,现略举一、二:

### 1. 文件加密

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

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

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

### 2. 隐藏 DOS 的引导文件

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

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

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

### 3. 提供相关文件的储存

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

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

## 验证四色猜想的 BASIC 程序

四川南充一中(637000) 陈庆祥

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

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

### 程序说明:

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

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

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

### 程序清单及运行实例:

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

```

70 FOR K=1 TO N
80 IF A(J,K)=1 AND C=B(K) THEN 60
90 NEXT K;B(J)=C;NEXT J
100 FOR K=1 TO N;PRINT LEFT$( " " +STR$(B
(K))+ " ",4);;B(K)=0

```

```

110 NEXT K;PRINT;T=T+1
120 NEXT I
130 PRINT;PRINT "TOTAL=";T;END

```

```

200 DATA 8
201 DATA 0,1,1,0,0,0,0,1
202 DATA 1,0,1,0,0,0,0,1
203 DATA 1,1,0,1,1,0,0,1
204 DATA 0,0,1,0,0,0,0,1
205 DATA 0,0,1,0,0,1,1,1
206 DATA 0,0,0,0,1,0,1,1
207 DATA 0,0,0,0,1,1,0,1
208 DATA 1,1,1,1,1,1,1,0

```

]RUN

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

TOTAL=3

修改从 200 语句开始的数据便可处理第二个图的着色问题:

```

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

```

]RUN

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

TOTAL=2

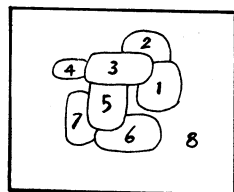


图 1

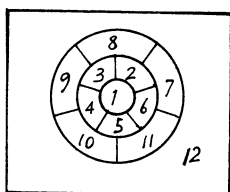


图 2

## 磁盘不抹掉文件的 35 轨改 40 轨方法

浙江杭州(310015) 钱仕宏

在 APPLE I, CEC-1 上, 标准 DOS3.3 FORMAT 过的磁盘都是 35 轨的, 怎样能把 35 轨的磁盘不抹掉上面的文件, 而改成 40 轨或 41 轨呢?

下面介绍一下改成 40 轨的方法, 需准备以下工具:

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

2) COPY 工具盘一张(最好 40 轨)

3) 空盘一张

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

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

3) 用 CPS 上的 SECTOR EDITOR 从原盘读入 \$11 轨 \$00 扇区, 将地址 \$34 处的 23 改为 28(41 轨则为 29), 再从地址 \$C4 处开始依次向后填 FF FF00 00 FF FF 00 00 FF……填到 \$D7(41 轨则为 \$DB)为止。最后将这一扇区写回原盘。

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

## 解拆 C—WORDSTAR 五笔字型

广东惠东职业中学

高一(5)班(516300) 陈晓乐

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

<一>启动 CEC—WORDSTAR 1.0, 进入主菜单后按“X”退出。

<二>在西文状态下进入监控, 键入:

BCB2:0C 0F

BCB6:00 40

BC66:4C 59 FF

BC56G

<三>光标出现后键入:



3D0G

然后存盘: BSAVE WB, A \$ 8A00, L \$ 550

BSAVE WB. DATA, A \$ 4000, L \$ 4000

(四)在 BASIC 程序中调用可编下面的程序:

10 D \$ =CHR \$ (4);PRINT D \$ "BLOOD WB";  
PRINT D \$ "BLOOD WB. DATA";CALL 353  
28;CALL 976

## BASIC 中“宏代换”的实现

安徽马鞍山钢铁公司红星中学(243000)

许斌

由于软硬件条件的限制,目前 CEC 上所运行的中文事务管理软件基本上都采用 BASIC 程序+数据文件的工作模式,虽然也能够实现对数据记录的各种处理,但使用总不如 dBASE 得心应手。这里一个很重要的原因是 dBASE 具有灵活的库记录筛选功能,在操作命令中可同时指定操作对象,使得对个别或部分数据记录的处理非常方便。尽管 BASIC 程序也能用 IF...THEN 结构来进行按某种特定条件的筛选,但不能根据需要动态地修改筛选条件,难以满足管理工作全方位的要求。从另一个方面讲,这也造成了 BASIC 语言编写的管理软件通用性较差,往往一个软件只适合某个方面的管理工作。

要是让 BASIC 程序具有类似 dBASE 的宏代换功能,根据用户要求,随时改换筛选条件,势必大大提高其数据处理能力。笔者日前在开发一个通用事务管理软件时,对这个问题进行了一些研究,找到一种较为理想的方法,现推荐给大家。

基本思路是这样的:我们可以用人工方式修改 IF...THEN 结构中的条件表达式,从而改变筛选条件。但在程序运行中,是不能这样做的,不过可以把这个工作给程序自行去完成,通过修改其自身的内存映像来改变筛选条件。具体步骤如下:

首先构造一个判断子程序(程序 1),条件表达式的位置暂用一特定的字符串填满,字符串长度视以后筛选条件的可能长度而定(本文中为 40)。程序输入后,进入监控,查看内存映像,根据字符串的代码,能很快找到其在内存中的位置(图 1), \$AD 是保留字 IF 的代码,记下这个代码的存放地址(本文中为 \$DB4 = 3508)。建议程序 1 最好放在主程序的前部,既便于查找,又可避免以后修改主程序会影响其在内存中的位置。

程序 1

1539 FG=0  
1536 IF 0000000000000000000000000000000000000000  
THEN FG=1  
1540 RETURN

0DA0- 36 38 2C 30 3A B1 00 B0

0DA8- 0D FA 05 46 47 D0 30 00  
0DB0- E3 0D 00 06 AD 30 30 30  
0DB8- 30 30 30 30 30 30 30 30  
0DC0- 30 30 30 30 30 30 30 30  
0DC8- 30 30 30 30 30 30 30 30  
0DD0- 30 30 30 30 30 30 30 30  
0DD8- 30 30 30 30 30 C4 46 47  
0DE0- D0 31 00 E9 0D 04 06 B1  
0DE8- 00 F2 0D OE 06 97 3A A2

图 1

程序 2 是筛选条件的动态修改子程序。其功能是接受用户输入的条件,转换成 BASIC 的内存映像形式,POKE 到 IF...THEN 结构中,长度不足的部分用空格填满。

程序 2

1550 HOME;VTAB 3  
1555 PRINT"请输入条件或回车"  
1560 INPUT"-";A \$;BT \$ =A \$  
1562 IF A \$ =" "THEN RETURN  
1565 I=1;L=LEN(A \$);C \$ =" ";B=I  
1570 A=ASC(MID \$ (A \$,I,1));IF A=127 THEN I=I  
+3;GOTO 1570  
1575 IF A<60 OR A>62 THEN I=I+1;GOTO 1570  
1580 B \$ =MID \$ (A \$,B,I-B);J=1  
1585 IF N \$ (J)=B \$ THEN B \$ =STR \$ (J);GOTO  
1600  
1586 J=J+1;IF J<=XM THEN 1585  
1590 PRINT CHR \$ (7);"ERR!";GOTO 1550  
1600 IF LX \$ (J)="N"THEN C \$ =C \$ + "Q (" +B \$  
+)"  
1605 IF LX \$ (J)="C"THEN C \$ =C \$ + "Q (" +B \$  
+)"  
1610 C \$ =C \$ +CHR \$ (269-A)  
1620 I=I+1;B \$ =MID \$ (A \$,I,1);A=ASC(B \$);IF  
A>59 AND A<63 THEN 1610  
1625 IF B \$ <> "\*" AND B \$ <> "+" THEN 1640  
1630 C \$ =C \$ +CHR \$ (163+A);I=I+1;B=I;GOTO  
1570  
1640 C \$ =C \$ +B \$;IF I<L THEN 1620  
1650 FOR I=1 TO LEN(C \$);POKE 3508+I,ASC(MID  
\$ (C \$,I,1));NEXT  
1655 FOR J=I TO 40;POKE 3508+J,32;NEXT  
1660 RETURN

输入条件的格式为:<字段名><关系算符><数值或字符串>AND/OR<字段名><关系算符><数值或字符串>.....,其中,为方便输入处理,AND 用 "\*"表示,OR 用 "+"表示。

例如,在一个工资库中,第三字段是工龄,第五字段是职称,第十七字段是工资,则要找职称为工程师或工龄超过 20 年,工资不少于 250 元的人,输入条件为:

职称="工程师"+工龄>20\*工资>=250

运行后,程序 1 中的条件表达式相应地变为:

程序 4

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

程序 3 是应用实例。说明如下:

3710:调用程序 2,设置筛选条件。

3730:打开工作文件。

3750:读出一个记录中的各字段。这里的数组变量名必须和程序 2 一致。

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

程序 3

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

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

程序中所用变量说明:

H:数据文件中的记录个数。

XM:每个记录中所含字段数目。

N\$(I):第 I 字段的字段名。

LX\$(I):第 I 字段的数据类型。

"N"表示数值型,"C"表示字符串型。

## 改进的 UNNEW 程序

唐山白玉瓷厂 张亭

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

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

```
]CALL-151↵
```

```
* 2E7:A001↵F3↵67↵88↵A2↵3↵B1↵69↵
E6↵69↵D0↵2↵E6↵6A↵C9↵0↵D0↵F2↵CA↵
D0↵F1↵4C↵F2↵D4↵
```

```
* 2E7G↵
```

```
]
```

至此,BASIC 程序已完全恢复。

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

比起原 UNNEW 程序来,本文程序在循环部分更精炼,同时使用了一个 6502 怪指令码 F3,与后面的 67 实现"INC(\$67),Y"的功能。这在监控的反汇编"L"命令中是看不到的。65C02 的用户不能用这个怪指令,只可根据道理稍加修改,也只需增加 1 个字节的长度。

## CEC-I 中华学习机 磁带游戏软件的解密

中国农科院蜜蜂研究所(100093)胡发新

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

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

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

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

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

## 也谈 CEC—I 全功能造字

广西巴马一中(547500) 覃敏

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

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

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

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

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

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

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

### 程序 1

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

### 用户造字字库

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

# ProDOS 磁盘操作系统入门(续)

廖 凯

## (五)、EXEC 命令

EXEC 命令将控制权从键盘转移到文本文件,而使计算机执行文件内的命令,就如同你从键盘打入一样。EXEC 不能被 CTRL-C 或 RESET 中断,要想中断只能重新启动系统。在执行 EXEC 命令时,不消除内存中的程序,而将 EXEC 文件内带行号的程序段加到内存中的程序上。EXEC 命令一次只能用于一个程序上。如果正在执行的命令中又有一个 EXEC 命令,则关闭第一个 EXEC 文件,执行第二个 EXEC 命令。EXEC 命令不被 NEW 或 CLOSE 命令所影响。监控命令在 EXEC 文件内不能执行。

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

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

EXEC 路径名[,F#][,S#][,D#]

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

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

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

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

### 1. 实际应用

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

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

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

②EXEC MAKE.TEXT 文件到你的程序内。

③列印行号 2 并改变 MAKE.TEXT 为你所需的文件名。

④打入 RUN,程序将被转换到行号 2 所指定的文

本文件内。

将文本文件加到程序内：

①装入你的 BASIC 程序。

②打入 EXEC 文件名,F6。这 F6 参数会跃过文本文件的首五行。

#### (六)CHAIN 命令

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

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

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

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

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

#### (七)、RAM 磁盘

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

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

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

#### (八)、系统启动

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

#### ProDOS 启动磁盘

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

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

```
PRODOS BASIC 1.1
```

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

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

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

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

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

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

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

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

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

#### (九)、绘图

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

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

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

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

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

(未完待续)

## 第四章 6502 MPU 的指令系统

南京大学大气科学系(210008) 朱国江

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

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

### 一、数据传送类指令

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

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

#### 1. 取数指令

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

##### ①LDA——由存储器取数送入累加器, M→A。

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

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

为是正数,所以负数标志 N 被置为 0。

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

##### ②LDX——由存储器取数至寄存器 X, M→X。

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

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

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

#### 2. 送存指令

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

##### ①STA——累加器 A 中的数存入内存, A→M。

STA 指令执行后累加器内容不变,内存内容改变。例如,(A)=63,(003A)=40,则执行 STA \$3A 指令后,零页地址 0034 中的内容变为 63。STA 指令有 7 种寻址方式,或者说内存单元的地址可分别由七种寻址方式表达。

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

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

#### 3. 传送指令

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

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

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

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

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

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

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

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

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

#### 4. 堆栈指令

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

计算机中的栈,是存放数据的,它是在随机存储器 RAM 中开辟的一个特殊存储区,即从一个固定地址开始的一片连续单元。在这个存储区中,数据的存(进)、取(出)必须遵从“先进后出”或者说“后进先出”的规则,这好像货栈一样,最先放入的货物,总是放在最低下,再放入货物时,往上堆;而取货时,总是把最上面的货物先取走(后进先出),最后才能取最底下的货物(先进后出)。这种特殊结构的存储区叫做堆栈。6502 规定堆栈设置在第 1 页中,即只能把 \$0100—\$01FF 这 256 个单元作为堆栈使用。在堆栈中,数据的存放是从

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

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

#### ①PHA——累加器进栈指令

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

#### ②PHP——标志寄存器 P 进栈指令。

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

#### ③PLA——累加器出栈指令

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

#### ④PLP——标志寄存器 P 出栈指令

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

进出栈指令的几点说明:

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

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

- PHA 指令要与 PLA 指令配对使用。

- PHP 指令要与 PLP 指令配对使用。

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

#### 5. 移位指令

移位指令是将累加器或存储单元中的各位依次左、右移动或循环左、右移动。

#### ①ASL——算术左移指令

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

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

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

### ②LSR——逻辑右移指令

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

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

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

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

### ③ROL——循环左移指令

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

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

### ④ROR——循环右移指令

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

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

至此,我们将数据传送类指令介绍完毕。

## 二、算术逻辑运算类指令

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

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

### 1. 加法指令

ADC——累加器与存储器带进位相加,结果送入

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

例如,立即寻址方式指令 ADC # \$ 33,设相加前(A) = 55,C=1,则执行 ADC # \$ 33 指令后,累加器 A 的内容变为 89。而 C 变为 0(因相加结果最高位上无进位)。其它标志位情况变动如下:结果不为 0,因而 Z=0;结果最高位是 1,则使负数标志 N=1;又由于相加过程中第 6 位和第 7 位中的一个发生进位(此例是第 6 位进位而第 7 位无进位),V 标志=1。即如果把两数视为带符号的正数,相加结果超出了一个字节所能表示的最大正数 7F,产生溢出。

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

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

### 2. 减法指令

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

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

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

### 3. 比较指令

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

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

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

本指令最常出现在条件分支指令的前面,影响 N、



Z、C 标志位的状态,作为实现条件跳转的依据。

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

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

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

#### 4. 加 1 指令

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

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

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

加 1 指令同将要介绍的减 1 指令,可以在多种应用上做为计数器用。

#### 5. 减 1 指令

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

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

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

#### 6. 标志位指令

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

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

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

③CLC——将进位标志 C 置 0:  $0 \rightarrow C$

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

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

#### 7. 逻辑运算指令

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

①AND——累加器与存储器内容逻辑“与”

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

AND 指令常用于屏蔽某些位或者取出某些位。

例如,已知  $(\$040) = 3D$ ,要求将其最低 4 位存放

至  $\$0041$  单元中,而最高 4 位清 0,则用程序:

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

又如,已知  $(\$4000) = AD$ ,要求将其  $d_6$  位存于 A 中,则可用下列指令实现:

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

②ORA——累加器与存储器内容逻辑“或”

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

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

③EOR——累加器内容与存储器内容“异或”

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

EOR 指令可用于求反码,6502 没有专门求反码指令。

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

#### 8. 位测试指令

BIT——用累加器检测存储器各位

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

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

至此,以上对 20 条算术逻辑运算指令作了介绍。

### 三、程序控制类指令

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

### 1. 无条件转移指令

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

如,对于绝对寻址方式指令 JUMP \$ 6000,CPU 执行:00→PC<sub>L</sub>,60→PC<sub>H</sub> 操作,而转向 \$ 6000 单元执行指令。对于间址寻址方式指令 JUMP(\$ 6000),CPU 执行:( \$ 6000)→PC<sub>L</sub>,(\$ 6000+1)→PC<sub>H</sub> 操作,而转向 PC<sub>H</sub> PC<sub>L</sub> 所指向的单元执行指令。

### 2. 条件转移指令

条件转移指令有四个特点:

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

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

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

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

①BEQ——结果为 0(Z=1)时转移,否则继续。

②BNE——结果非零(Z=0)时转移,否则继续。

③BCC——进位标志 C 为 0 时转移,否则继续。

④BCS——进位标志 C 为 1 时转移,否则继续。

⑤BPL——结果为正数(N=0)时转移,否则继续。

⑥BMI——结果为负数(N=1)时转移,否则继续。

⑦BVS——有溢出(V=1)时转移,否则继续。

⑧BVC——无溢出(V=0)时转移,否则继续。

### 3. 子程序指令

#### ①JSR——转子指令

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

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

#### ②RTS——返主指令

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

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

### 4. 中断指令

中断指令共有 4 条:即 CLI,SEI,BRK,RTI。它们均

采用隐含寻址方式。

#### ①CLI——开中断指令

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

#### ②SEI——关中断指令

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

#### ③RTI——中断返回指令

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

#### ④BRK——软件中断指令

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

### 5. 空操作指令

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

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

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

## 电 脑 报

●集实用性、知识性、资料性、趣味性于一体,可读性强,可长期保存、查阅。

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

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

●全国邮局均可订阅,月价:0.68 元,年价:8.16 元

●社址:重庆市双钢路 3 号,邮编:630013

# 初级程序员级软件水平考试辅导

## 操作系统及 PC—DOS

北京电脑天地学校(100051) 宋丹颖

### 自测题

一、判断下列叙述是否正确,请在相应的题号前面√(正确)或×(错误)。

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

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

3. 接收和完成用户对数据库的存取请求的系统称为数据库系统。

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

5. 兼容机之间指令系统基本上是相同的,但硬件实现方法可以不同。

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

7. 在多级目录结构中,不允许两个不同的文件具有相同的名字。

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

9. 内部命令是在系统启动时由装入程序读入内存并常驻内存的命令。

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

二、从供选择的答案中选出正确答案的编号填写在横线上。

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

供选择的答案:

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

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

供选择的答案:

A. 负责接受和解释用户输入的命令及出错处理,具有全部内部命令的处理程序

B. 负责出错处理,具有全部内部命令、外部命令的处理程序

C. 负责接受和解释用户输入的命令,具有全部内部命令、外部命令的处理程序

D. 负责出错处理,具有全部外部命令的处理程序

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

供选择的答案:

A. 磁盘驱动器是否关好小门;磁盘是否贴有写保护

B. 磁盘是否贴有写保护;命令字拼写及格式是否正确

C. 查看命令及格式拼写是否正确;查看磁盘上是否有该文件

D. 内存中是否有该文件,是否为可执行文件

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

供选择的答案:

A. 建立或编辑文件的一系列的子命令 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的命令 D. 合并单独产生的目标模块,控制被调试程序运行的命令

三、从供选择的答案中选出与下列叙述关系最密切的字句。

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

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

C, D: ①设备管理 ②文件管理 ③驱动器管理 ④作业管理

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

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

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

供选择的答案:

- A: ①可直接使用,不必进行格式化
- ②只有硬盘才必须先进行格式化

- ③只有软盘才必须先进行格式化
- ④都必须先进行格式化
- B: ①“不同操作系统下格式化的软盘是不可通用的”
- ②“写保护装置起作用的磁盘无法被格式化”
- ③“格式化一个磁盘将破坏磁盘上的所有信息”
- ④“在 IBM-PC 机 DOS 下被格式化过的磁盘不能再在其它种类的操作系统下被格式化”
- C: ①有文件 ②有子目录 ③有多级子目录 ④空
- D: ①可读 ②可写 ③写保护 ④读保护
- E: ①可读 ②可写 ③写保护 ④覆盖

四、设当前系统提示符为 C>, 请建立批处理文件, 它具有的功能如下:

1. 自动在用户盘(放在 B 驱动器内)上建立一个名为 A1 的子目录;
  2. 把系统盘(放在 A 驱动器内)上的 WB1. TXT 文件复制到 A1 子目录下;
  3. 在用户盘上再建立一个名为 A2 的子目录;
  4. 把系统盘上的 WB2. TXT 文件复制到 A2 子目录下;
  5. 把两个子目录下的. TXT 文件连接为一个 BIG. TXT 文件放在 A1 子目录下, 要求 WB1. TXT 在前, WB2. TXT 在后;
  6. 在屏幕上显示本批处理文件自身的内容;
- 五. 现有 A 盘目录结构如下:

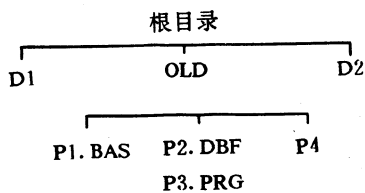


图 1

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

1. 在子目录 D1 下建立一个名为 BAS 的子目录, 把 OLD 下的 P1. BAS 文件复制到新建的子目录下;
2. 在子目录 D1 下建立一个新的子目录 DB, 把 OLD 下的 P2. DBF 和 P3. PRG 文件复制到新建的子目录下;
3. 在子目录 D2 下建立一个名为 WS 的子目录, 把 OLD 下的 P4 文件复制到新建的子目录下;
4. 删除 OLD 子目录下的全部文件;
5. 删除子目录 OLD;
6. 画出更改后的树形目录结构图。

## 分析及答案

### 一、分析

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

一方面, 因此操作系统不仅仅是键盘命令的集合。  
2. 正确。  
3. 正确。  
4. 错误。本叙述中描述的特点及应用都适用于实时系统而不是分时系统。实时系统的特点是时间性强, 响应快, 可靠性高。

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

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

答案: 1. × 2. ✓ 3. ✓ 4. × 5. ✓  
6. ✓ 7. × 8. ✓ 9. ✓ 10. ×

### 二、分析

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

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

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

响从盘上读出文件。

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

答案: 1. B 2. A 3. C 4. C, B

### 三、分析

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

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

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

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

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

答案:

1. A② B① C① D②  
2. A④ B④ C④ D① E②

### 四、分析

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

解:C>COPY CON B;P1. BAT

MD B;\A1

COPY A;WB1. TXT B;\A1

MD B;\A2

COPY A;WB2. TXT B;\A2

COPY B;\A1\WB1. TXT + B;\A2\

WB2. TXT B;\A1\BIG. TXT

TYPE B;P1. BAT

^ Z

C>TYPE B;P1. BAT

### 五、解:

A>MD \D1\BAS

A>COPY \OLD\P1. BAS \D1\BAS

A>MD \D1\DB

A>COPY \OLD\P2. DBF \D1\DB

A>COPY \OLD\P3. PRG \D1\DB

A>MD \D2\WS

A>COPY \OLD\P4 \D2\WS

A>CD OLD

A>DEL \* . \*

A>CD. .

A>RD \OLD

更改后的树形目录结构图:

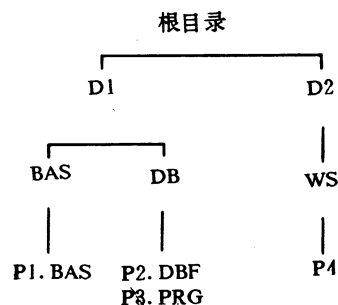


图 2



# 实现单片机最小系统与 PC 机的通信

合肥中国科技大学计算中心(230026) 张培仁

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

单片机通信模块的设计:

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

单片机 MCS-51 串行口是全双工串行通信口。

串行口控制寄存器是 SCON。

SCON

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

SM0,SM1:工作方式选择位,如下表。

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

表中 UART 是异步串行通信。

SM2:允许方式 2,3 多机通信。

REN:允许串行接收。

TB8:在方式 2 和方式 3 时是发送的数据的第 9 位。

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

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

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

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

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

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

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

波特率的设定:

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

波特率=(定时器 1 的溢出率) \* 1/32(或 \* 1/16) 1/32,1/16 选择由波特率选择寄存器 PCON 的第七位确定。PCON.7=0 是乘 1/32,PCON.7=1 乘 1/16。

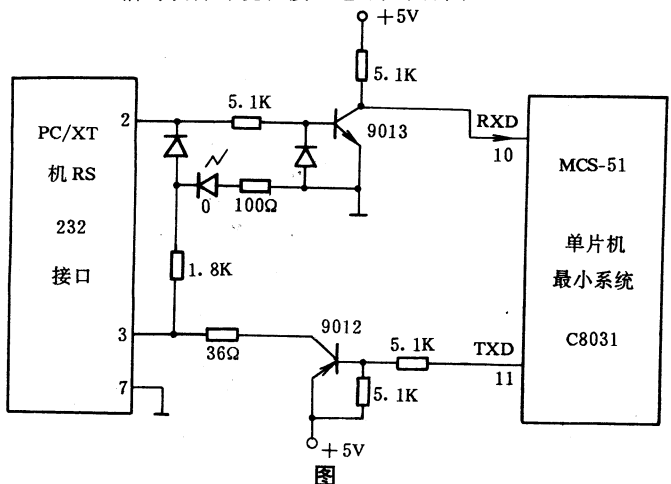
定时器 1 的溢出率=计数速率/(256-一定时器计数常数)

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

当波特率是 1200 时,这时 TH1=TL1=0F3H

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

通信的硬件环境和接口电路如图所示。



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

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

单片机通信模块包含三个子模块。

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

发送模块:将累加器 ACC 中的内容发送到串行口。

接收模块:接收串行口上的数据,送到累加器 ACC 中。

PC/XT 机通信模块简介如下:

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

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

PC 机通信模块也分 3 个子程序模块。

初始化模块:定义波特率,定义串行口数据格式。

接收模块:将数据接收缓冲器的内容写入指定文件中。

发送模块:将指定的文件内容送给数据发送保持器。

PC 机和单片机通信的运行步骤如下:

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

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

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

把 ABC. ASM 进行汇编产生 ABC. OBJ 文件。

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

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

通信程序,其他步骤同上,即可完成相应通信功能。

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

DP. ASM, ZHANKAN. PAS 两个通信程序全部清单如下:

```
FILE NAME DP. ASM
org 2000h
anl tmod, # 0f0h
orl tmod, # 020h ;定时器 1 置方式 1
mov th1, # 0f3h
mov t11, # 0f3h ;置时间常数
clr pcon. 7 ;清波特率倍增位
setb tr1 ;启动定时器 1
clr es
clr etl ;禁止中断
mov scon, # 052h ;置串行口工作方式
pd: lcall load ;判断接收
jz frompc ;结束
cjne a, # 01h, pd; ;发送
ajmp topc
load: jnb ri, load ;如果 n=0 则等待
clr ri ;清接收中断请求位
mov a, sbuf ;接收数据送 a
ret
rd: lcall load ;将起始地址送地址寄存器
mov dph, a
lcall load ;
mov dpl, a ;
lcall load
subb a, dph ;用末址减首址得总字节数
mov r0, a ;送 r0, r1
inc r0 ;
lcall load
subb a, dpl
mov r1, a
ret
frompc: lcall rd
jnc f1 ;无进位转 f1 否则 r0-1
dec r0 ;
f1: lcall load ;
movx @dptr, a ;接收的数据送外存
inc dptr ;指向下一个地址
djnz r1, f1 ;
djnz r0, f1 ;文件没有接收完,
ajmp pd ;继续接收
topc: lcall rd
jnz t1 ;进位位=0 转 t1 否则
dec r0 ;r0-1=>r0
t1: movx a, @dptr ;发送
lcall send
inc dptr
djnz r1, t1
djnz r0, t1
```

```

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

```

FILE NAME ZHANKAN.PAS

```

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

```

```

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

```



```

file name:');
gotoxy(48,10);
hh(c1);
writeln;
gotoxy(46,12);
hh(c2);
gotoxy(60,14);
readln(st);
gotoxy(20,22);
write('Waiting...');
assign(file1,st);
rewrite(file1);
ww(hi(c1));
ww(lo(c1));
ww(hi(c2));
ww(lo(c2));
send(hi(c1));delay(50);
send(lo(c1));delay(50);
send(hi(c2));delay(50);
send(lo(c2));
for i:=1 to c2-c1 do
begin
cc(c3);
ww(c3);
end;
gotoxy(40,23);
writeln('O.K. ');
close(file1);
end;
BEGIN
10: clrscr;           ;主程序
gotoxy(22,2);       ;菜单选择
write(' * * * * * ');
gotoxy(22,3);
write(' * * * * * ');
gotoxy(22,4);
write(' * * * * * ');
gotoxy(25,4);
writeln('1. Move date from pc to 8031');
gotoxy(22,5);
write(' * * * * * ');
gotoxy(22,6);
write(' * * * * * ');
gotoxy(25,6);
writeln('2. Move date from 8031 to PC');
gotoxy(22,7);
write(' * * * * * ');
gotoxy(22,8);
write(' * * * * * ');
gotoxy(25,8);
writeln('3. Exit');
gotoxy(22,9);
write(' * * * * * ');
gotoxy(22,10);
write(' * * * * * ');
gotoxy(25,14);
write('Please select(1..3);');
ch:=readkey;
if ch<>'1' then if ch<>'2' then
if ch<>'3' then goto 10;
ws;
case ch of
'1':begin
send($00);
pctomcs;
goto 10;
end;
'2':begin
send($01);
mcstopc;
goto 10;
end;
'3':begin
send($11);
halt;
end;
end;
END.

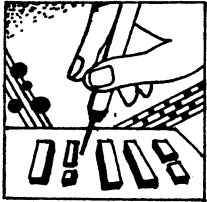
```

(上接43页)

上运动。可在同一个画面上出现16个卡通图案，八个静止的，八个运动的。

- (2)解除 DEF MOVE 语句的作用使用 CAN 语句。解除 DEF SPRITE 语句的作用使用 SPRITE 语句，因为 SPRITE OFF 语句虽能令卡通图案消失，但它没有选择消失的功能，所以必须使用 SPRITE n 语句。
- (3) 设定卡通图案的显示座标，即显示用 DEF SPRITE 定义的卡通图案用 SPRITE。设定卡通图案运动座标用 POSITION。

- (4)我们学习了三个设定座标语句 LOCATE、POSITION、SPRITE，必须分清三个语句的适用对象和取值范围。LOCATE 语句设定打印座标，取值范围 X:0~27,Y:0~23;POSITION 设定运动卡通图案的显示座标，取值范围 X:0~240,Y:0~220;SPRITE 语句定义静止卡通图案的显示座标，取值范围同 POSITION 语句。
- (5)用 DEF SPRITE 语句定义的卡通图案只有 SPRITE、SPRITE ON、SPRITE OFF 对它起作用，其它语句不能控制。



# 学装微电脑

## 红外线遥控室内温度

易齐干

自动控制室内温度的过程中,室内任何一角落欲获得舒适温度时,必须随时改变室温设定值。如果室内多处设置温度传感器,通过红外线遥控将传感器测得的温度信息,传输给空调机,这样可获得舒适的室温。

本实验使用日本太平洋工业(株)制作的 LSI PX-1000。它适用简易低速多路传输。用于发送时,将 20 路并行数据自动地变换为串行数据;用于接收时,所接收的串行数据自动地复原为 20 路并行数据。发送或接收取决于 PX-1000 的切换出脚的电位。出脚名称如图 1 所示,功能如表 1 所示。

发送电路原理图如图 2 所示。PX-1000 输出串行

数据经过晶体管和红外发光二极管转换为光信息。串行输出 H 电平,红外发光二极管发光。红外二极管为 NEC 的 SE303A,它定向角广,线性性好。

接收电路如图 3 所示。光敏二极管 PH302 接收来自发送部份的光信息,转换为电平信号,经过运算放大器 TL321 放大,输入给 PX-1000 的串行输入脚。

如果使用  $\mu\text{P}-80$  微电脑教育套件由红外线遥控进行温度显示,只能在  $15^{\circ}\text{C}\sim 30^{\circ}\text{C}$  温度范围内进行。首先,照图 4 所示连接硬件。端口 C 低位连接 D/A 变换部件、温度传感器部件和比较器部件\*。端口 C 高位连接 H/L 电平验证部件。电路原理图如图 5 所示。

表 1

| 出脚的符号                              | 区分   | 功 能                         |
|------------------------------------|------|-----------------------------|
| D19~D0                             | 输入输出 | 输入、输出 并行数据                  |
| T $\checkmark$ R                   | 输入   | H: 发送; L: 接收                |
| X1、X2                              | 输入   | 输入 1~6 MHz 的振荡频率            |
| RESET                              | 输入   | L 电平为 5 个周期以上时复位            |
| EC                                 | 输入   | H: 发送或接受; L: 停止             |
| DC                                 | 输入   | L: 两次检验接受数据                 |
| 20 $\checkmark$ 8                  | 输入   | H: 发送或接受 20 路; L: 发送或接受 8 路 |
| RD                                 | 输入   | 输入 串行数据                     |
| TD/ERR                             | 输出   | 发送时输出 串行数据, 接受时输出 接受误差      |
| P $\checkmark$ N                   | 输入   | H: 正逻辑输入输出, L: 负逻辑输入输出      |
| V <sub>cc</sub> (V <sub>cc</sub> ) | 电源   | 5V 接线端                      |
| GND (GND)                          | 电源   | 地线接线端                       |

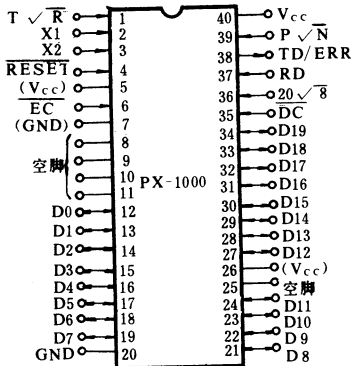
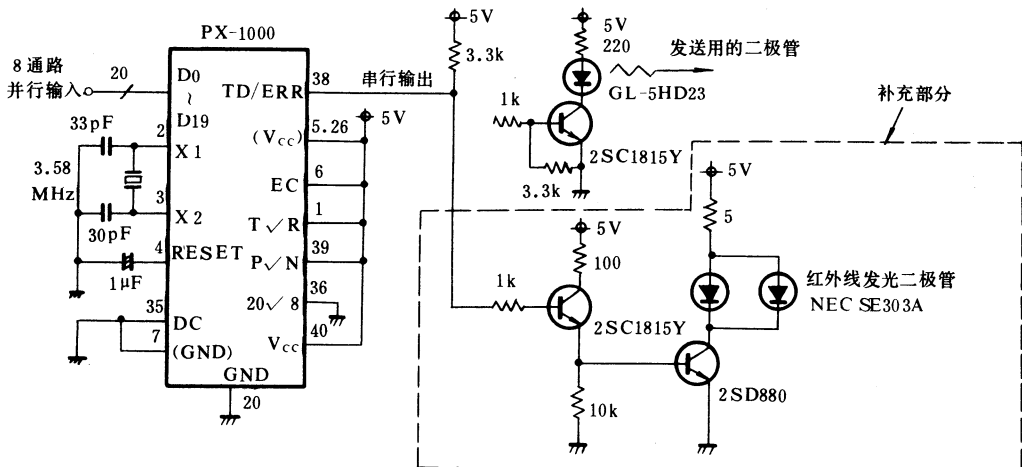
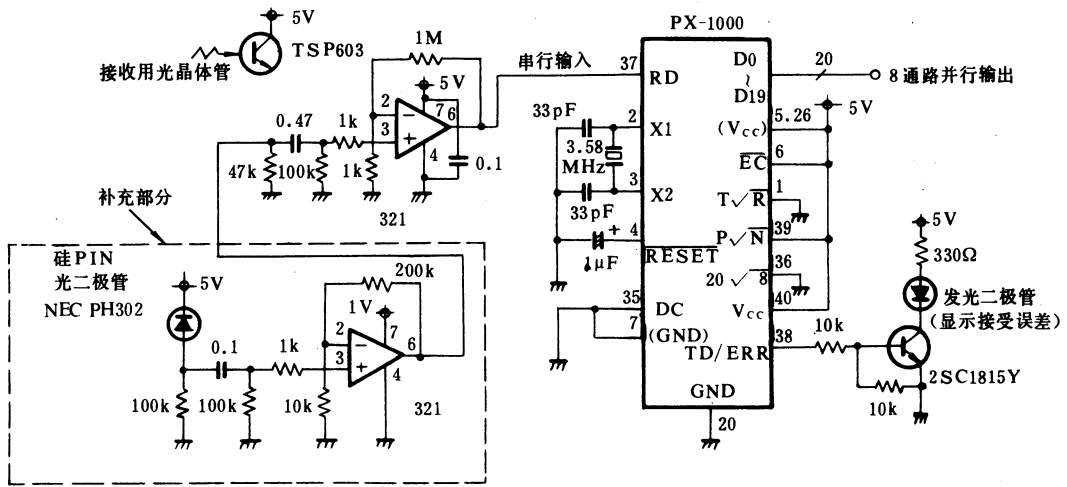


图 1 PX-1000 出脚接线图



注 (1) 8 通路发送, 正逻辑输入。(2) 使用 3.58MHz 陶瓷振荡时的传送速度为 31ms/8 路

表 2 流程图示例



注：8 通路发送，双重检验，正逻辑输入。

图 3 接收部分的电路图

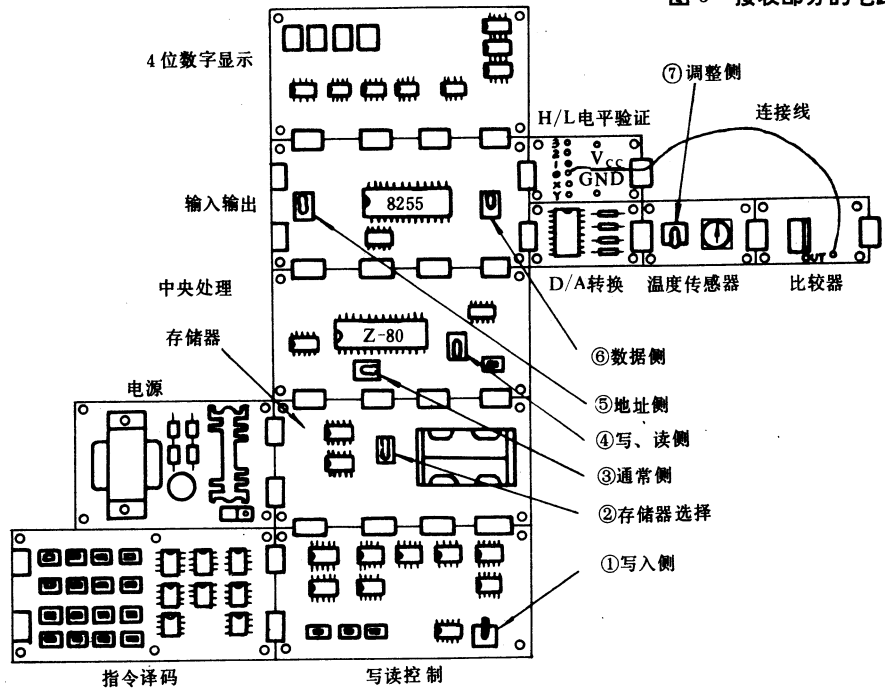


图 4 太平洋微电脑教育部件速接图

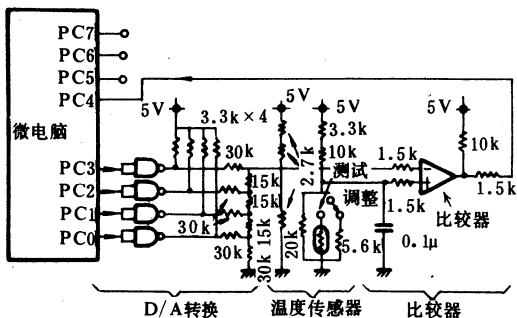


图 5 输入输出部件端口 C 的电路图

编制程序的要点：

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

寄存器 B 暂时存放 D/A 转换数据，寄存器 C 暂时存放现在室温。

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

程序流程图和程序清单如表 2、表 3 所示。

表 2

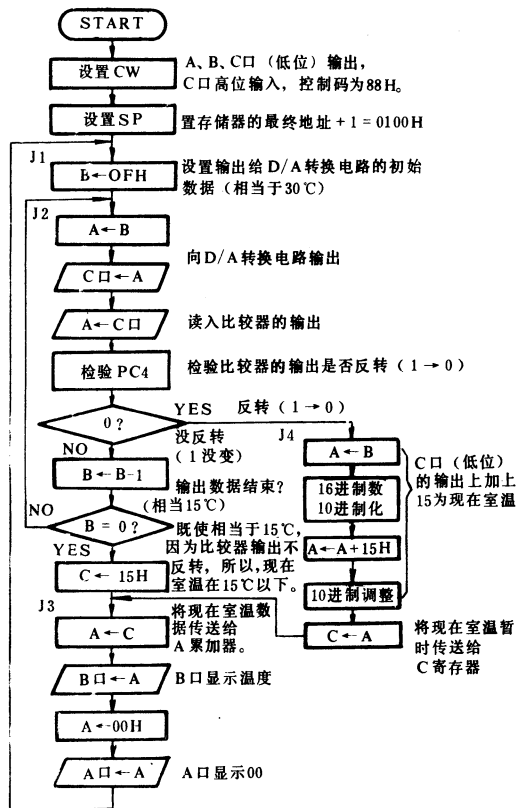


表 3

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

将程序写入微电脑, 检验写入是否正确。执行程序。

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

用手指触摸温度传感器部件的热敏电阻, 温度显示发生变化, 说明微电脑工作正常。

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

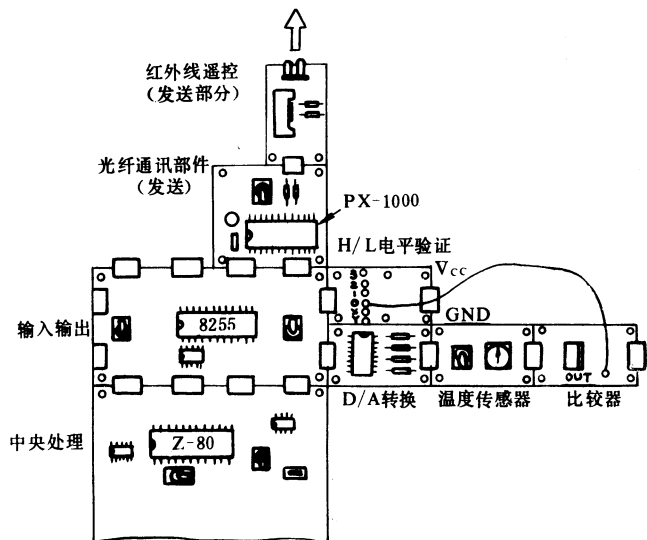


图 6 发送部分连接图

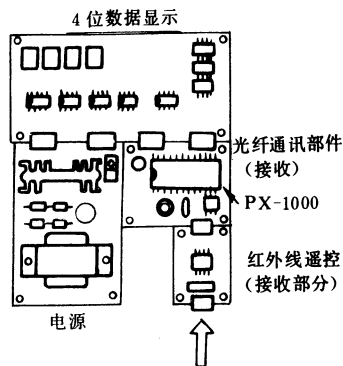


图 7 接收部分连接图

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

再用手指触摸热敏电阻, 4 位数字显示发生变化, 说明红外线遥测发送实验成功。

发送距离随周围条件而变化, 日光灯照明条件下, 能发送 4 米距离。

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



# Z80 单板机与打印机巧相连

北京信息工程学院(450002) 张慧成

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

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

由原理图 1 可以看出:单板机来的数据总线通过 74LS244 驱动与打印机数据线相连。其中 D<sub>0</sub> 线有双重作用:写操作时,D<sub>0</sub> 为数据信息,读操作时,D<sub>0</sub> 为打印机的状态。读打印机状态时的读信号由  $\overline{IOR} + \overline{PS}_7$  组成,  $\overline{IOW} + \overline{PS}_7$  作为选通打印机的写信号。当单板机向打印机送打印信息时,数据总线上出现的是写向打印机的待打印信息。当向打印机接口板发读命令时,若读得结果的 D<sub>0</sub> 位为“1”,则打印机处于“忙”状态,否则处于空闲状态。据此可以编写出打印字符程序如图 2:

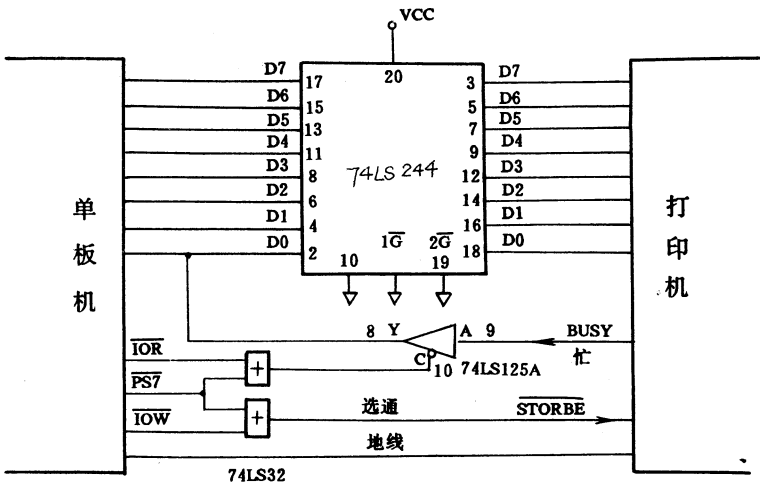


图 1. Z80 单板布线区上外接的扩展打印机接口板

```

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

图 2 打印字符子程序

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

```

入口 A-----待打印的数据
PHEX: PUSH AF      ;保存数据
      AND 0F0H     ;取高 4 位
      RLCA
      RLCA        ;移到字节的低 4 位
      RLCA
      ADD A, 30H   ;变为 ASCII 码
      CP 3AH      ;判是小于 9 的数字吗?
      JR C, PH1   ;是,转 PH1
      ADD A, 07H  ;;数字为 10—15,
                  ;变为 A—F 的 ASCII 码
PH1:  CALL PCHR   ;打印输出
      POP AF     ;恢复数据
      AND 0FH    ;取低 4 位
      ADD A, 30H ;变为 ASCII 码
      CP 3AH    ;判是小于 9 的数字吗?;
      JR C, PH2 ;是,则转 PH2
      ADD A, 07H ;否,变为 A—F 的 ASCII
                  ;码
PH2:  CALL PCHR   ;打印输出
      RET

入口: (HL)-----信息首地址指针,数据
                  ;信息
                  ;以控制字符 03H 表示
                  ;结尾
PMSG:  PUSH AF
      PUSH HL    ;保护现场
PMSG1: LD A, (HL) ;数据信息送 A
      CALL PCHR ;打印输出
      INC HL    ;指向下一个
      LD A, (HL) ;取信息到 A
    
```

```

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

```

图 4 打印信息子程序

```

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

```

```

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

```

图 5 打印 Z80—CPU 寄存器内容子程序

(上接 39 页)

```

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

```

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

```

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

```

除了上述 6 行有不同之外,这四个源文件的其他部分是完全相同的。

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

# PAL 卡使用中的几个问题

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

该卡在使用中可能出现一些问题,解决办法如下:

1. 场不同步

画面上下滚动或为两三个静止画面。

此时,应调整电视机的垂直同步旋钮或电位器(V—HOLD)。

2. 行不同步

荧光屏上是许多向左或向右倾斜的条子。

此时,应调整电视机的水平同步旋钮或电位器(H—HOLD)。

3. 画面左右偏移

可调整 PAL 卡上的电位器 W2。

4. 无彩色

微调 PAL 卡上的可变电容器 C1 和电位器 W1。

5. 色调失真

画面颜色与实际颜色不符。

微调 PAL 卡上的电位器 W1。

6. 场不同步且电视机无垂直同步旋钮或电位器可调。此时可运行下列程序:

(1)AT40.COM

用于 40X25 字符方式。

(2)AT80.COM

用于 80X25 字符方式。

(3)AG40.COM

用于 320X200 图形方式。

(3)AG80.COM

用于 640X200 图形方式。

上述四个程序运行后,荧光屏上将显示由“.”组成的网格。

按 ↑ ↓ ← → 四个键可使行场同步且使画面上下左右偏移。

按 ESC 键退出该程序。

AT40.ASM 源文件如下:

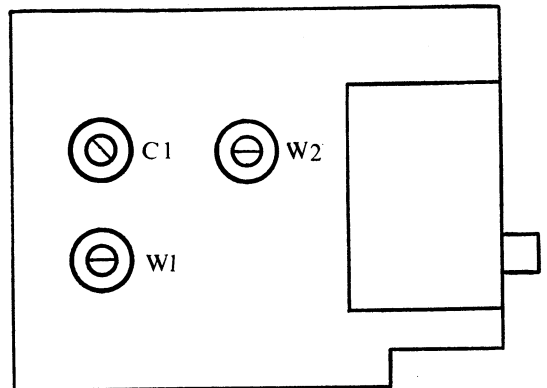
```

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

```

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

(下转 38 页)



图

# 降低 MP-1 电脑的功耗

广西苍梧县佛子冲铅锌矿机动科(543100) 邓文超

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

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

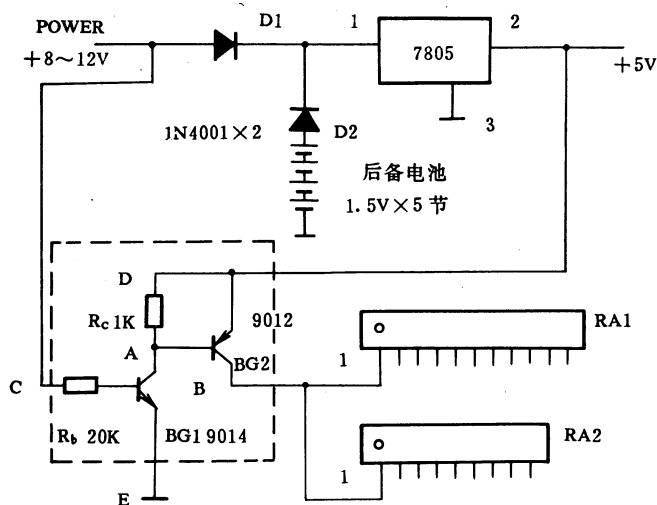


图 1

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

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

## 书讯

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

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

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

改装步骤:

1. 将电阻排 RA1、RA2 焊下(若没有专用拆焊集成电路的工具可借助于 12 号注射针头, 逐个脚拆焊), 并将它们的第 1 脚摺弯起来。
2. 将 RA1、RA2 的各脚除第 1 脚之外焊回原处。
3. 用细导线将 RA1、RA2 的第 1 脚焊接连通。
4. 用刀片将 POWER 与 7805 稳压集成块的第 1 脚之间连接的印刷线路切断(插元件面和焊点面均须割断), 并在断口处跨焊一只 4001 二极管(即图 1 中的 D1)。D2 设置在 7.5 伏电池箱中。

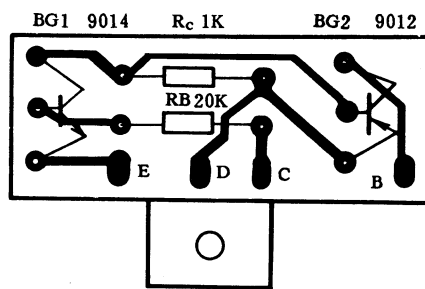


图 2

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

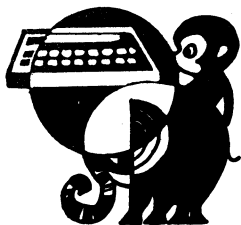
调试:

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

## 敬告作者

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





## 电脑游戏机

# 第三章 F BASIC 的画面控制语句

山东 于春

### 六、配色语句(COLOR、CGSET、PALET)

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

#### 1. 置色语句(COLOR)

COLOR 简写 COL.

COLOR 语句的功能是在屏幕的某位置显示某种颜色。语句格式为

COL. X, Y, n

式中 X: 屏幕水平方向的位置座标0~27.

Y: 屏幕垂直方向的位置座标0~23

n: 配色号码0~3

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

#### 例23.

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

RUN 屏幕先连续打印一兰色底面,然后产生各种色彩块的千变万化。

#### 2. 规定 BG、SP 面颜色组合语句(CGSET)

CGSET 简写 CG.

CGSET 语句的功能是规定背景及卡通画面的颜色组合。语句格式为:

CG. m, n

式中 m—背景用的色块代码0~1

n—卡通用的色块代码0~2

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

#### A. 卡通图案的配色

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

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

#### B. 背景图案的配色

(1) 通过 CGSET 语句中的 m 选定 BG 的面板代码0~1。

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

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

#### 例24.

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

RUN 可以看到背景、色块、飞鸟的各种颜色变化。其变化顺序为

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

⋮  
⋮

#### 3. 重新定义配色号码语句(PALET)

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

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

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

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

80 G.10

RUN 显示玛丽,丽莎两个卡通图案三种色彩的反复变化。

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

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

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

为便于阅读和理解,示例有关部分加进中文注释。为减少篇幅,删去了音乐程序。

例26. 游戏程序片断

```

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

```

160 GOS.440 (转SP相遇处理子程序)

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

(测试动作结束的座标。)

```

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

```

(180~220定义玛丽向右跳上去。)

```

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

```

(230~270为玛丽向右跳下。)

180~270为玛丽向右跳一次。)

```

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

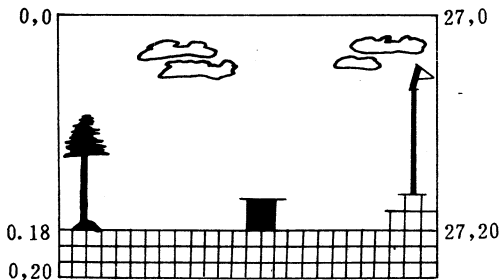
```

```

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

```

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



背景图案设计图:

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

最后, 列举一个完整的、简单的游戏程序示例。

#### 例27. 打飞鸟

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

```

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

```

```

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

```

#### 240 END

程序中使用了发声语句 PLAY, 以前没有介绍, 读者可参阅随机手册。

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

最后说明几点:

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

(下转 33 页)



# 中华学习机电源故障维修经验谈

浙江嘉兴市化工厂企管办(314021) 汪晖

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

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

## 一、电路工作原理

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

### 1. 主电路

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

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

### 2. 控制电路

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

### 3. 保护电路

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

## 二、电路的优缺点

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

缺点:电路较复杂,相互牵扯多,维修相对较困难,对各元件要求较高。

## 三、举例谈维修

下面结合两例维修实例,谈修理技法。

例 1. 开机屏幕上无任何显示,键盘右上方指示灯不亮。

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

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

#### 例 2. 故障现象同上

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

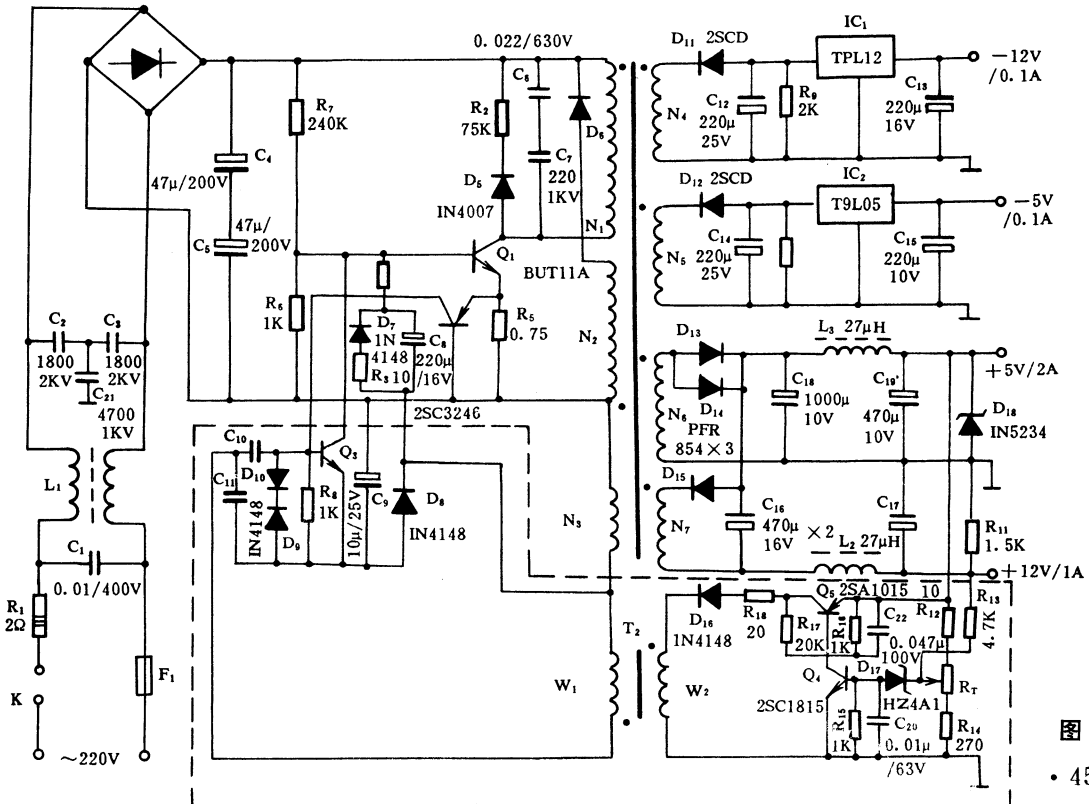
#### 四、小结:

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

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

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

| 编号    | 型号      | 电压值               |      |           |      |           |      |
|-------|---------|-------------------|------|-----------|------|-----------|------|
|       |         | $V_c$ (伏)         |      | $V_b$ (伏) |      | $V_e$ (伏) |      |
|       |         | 静态                | 动态   | 静态        | 动态   | 静态        | 动态   |
| $Q_1$ | BUT11A  | 270               | 260  | -3.3      | -3.1 | 0         | 0    |
| $Q_2$ | 2SA1015 | -7.4              | -8.7 | 0         | 0    | 0         | 0    |
| $Q_3$ | 2SC3246 | -3.3              | -3.1 | -7.4      | -8.7 | -4.6      | -5.8 |
| $Q_4$ | 2SC1815 | 4.4               | 4.4  | 0.7       | 0.6  | 0         | 0    |
| $Q_5$ | 2SA1015 | 1.4               | 2.6  | 4.4       | 4.4  | 5.1       | 5.1  |
| 备注    |         | 以上数据为 MF64 型万用表实测 |      |           |      |           |      |



图

表二

MF64 型 R×100

| 管脚             | Q <sub>1</sub> |      | Q <sub>2</sub> |    | Q <sub>3</sub> |      | Q <sub>4</sub> |       | Q <sub>5</sub> |      |
|----------------|----------------|------|----------------|----|----------------|------|----------------|-------|----------------|------|
|                | 红地黑测           | 黑地红测 | 红地             | 黑地 | 红地             | 黑地   | 红地             | 黑地    | 红地             | 黑地   |
| R <sub>c</sub> | 20K            | 1K   | 1.7K           | ∞  | 1K             | ∞    | 1.5K           | 1.5K  | 1.1K           | 1.5K |
| R <sub>e</sub> | 0              | 0    | 0              | 0  | 1.5K           | ∞    | 200Ω           | 200Ω  | 0              | 0    |
| R <sub>b</sub> | 700Ω           | 700Ω | 0              | 0  | 1.7K           | 900Ω | 1.1K           | 1.15K | 950Ω           | 850Ω |

表三

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

## 袖珍计算机 PC 型 PB 型系统 RAM 扩展 模块常见故障与维修

湖北沙市饲料公司微机室(271000) 朱杰

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

常见原因:

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

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

(3) V<sub>CC</sub> 电源隔离二极管 1S598 开路, 片选电路没有加上工作电源。对此可启开模块, 重新焊好即可。

(4) RAM 存储器电路损坏。可选用片状扁平封装的 HM6116 替换。

2. 显示内容与预存内容不符或丢失, 主机显示紊乱, 出现无规则显示点。

常见原因:

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

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

(3) 模块内电源滤波电容漏电, 使 RAM 供电不稳定。

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

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

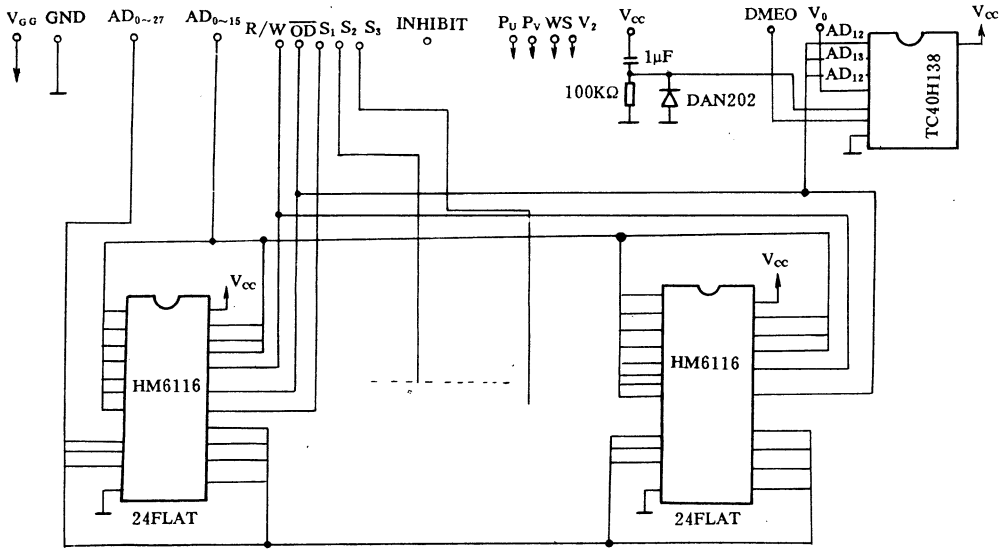
常见原因:

(1) CE-155 模块由 4 片 RAM 芯片 MH6116 组成, 其中有三块的片选由主机 PC-1500 机的三线八选一译码分配电路 40H138 控制而另一片则由模块内的片选电路 40H138 控制片选, 该芯片由主机 DME0, Y<sub>0</sub> 控制。当 40H138 损坏, Y<sub>7</sub> 输出低电平, 受控的 MH6116 CS 端电平为低, 使该片容量挂不上内存。可用相同的芯片代换。

(2) 主机模块仓或模块插头接触不良, 可用清洁剂擦洗插头部分。

(3)主机的 DME0 Y<sub>0</sub> 信号没有送到模块的片选电路,如因铜箔线被腐蚀,隔离元器件损坏。

(4)RAM 电路损坏,可用片状扁平封装的 MH6116 替换。



图

(上接 48 页)

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

- C800:0000 55;第 0 字节内容
- C800:0001 AA;第 1 字节内容
- C800:0002 XX;第 2 字节为 ROM 程序长度
- C800:0003 YY;第 3 字节从该字节开始为硬盘控制板上 ROM 程序及参数

对硬盘进行低级格式化的具体操作如下:  
在 DEBUG 状态下:

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

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

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

—G=C:800××××

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

例如长城 0520CH,20MB 的硬盘,利用 ROM 实现

硬盘低级格式化的过程是:

```
A)DEBUG
-D C800:0000 000F 55 AA EB E9 52 08-5B
  42 58 44 30 36 28
-U C800:0003 000F
  C800:0003 EBIJ      JMP 0023
  C800:0005 E95208   JMP 085A
  C800:0008 5B      POP BX
  C800:0009 42      INC DX
  C800:000A 58      POP AX
  C800:000B 44      INC SP
  C800:000C 30362028 XOR 12820],DH
```

-G=C800:0005

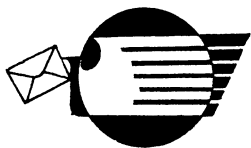
Hard Disk Formatting Utility v3.0

This program will erase all data on the specified hard disk

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

此时只要正确回答硬盘的数目然后打回车键即开始对硬盘进行低级格式化操作。

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



# 普及型 PC 机个人用户软件交流联谊活动

## 问题解答(四)

### 读者联谊

中国农科院计算中心(100081) 王路敬

#### 11. PC 机系统板上的 ROM BIOS 从结构上有哪几部分组成?

根据 ROM BIOS 所提供的功能,它有以下七部分组成:

##### (1)程序数据区

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

##### (2)自测试程序

包括如下内容:

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

##### (3)自举装入程序 INT 19H

(4)主要输入/输出设备的驱动程序

这些驱动程序主要有:

RS 232 I/O——INT 14H

键盘 I/O——INT 16H

软盘 I/O——INT 13H

打印机 I/O——INT 17H

显示器 I/O——INT 10H

##### (5)系统配置分析程序

它包括:

确定 ROM 的容量——INT 12H

确定系统配置的设备——INT 12H

##### (6)其他中断服务程序

这些服务程序包括:

不可屏蔽中断处理程序;

系统的时钟——INT 1AH;

屏幕打印——INT 05H;

系统使用的临时中断服务程序。

##### (7)图形方式的字符发生器图形提供 128 个字符 8 \* 8 点阵图形。

#### 12. 硬盘 ROM BIOS 由哪几部分组成?

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

##### (1)硬盘 I/O 参数的定义

其中包括:

错误代码参数

设备地址端口

硬盘控制命令字

##### (2)硬盘数据区

它包括:

在 0 段设置磁盘中断向量号以及 40 段设置磁盘状态字。

##### (3)硬盘 I/O 的初始化程序

该程序的作用是使硬盘输入/输出驱动程序进入系统。

##### (4)自举装入程序——INT 19H

##### (5)硬盘 I/O 驱动程序。

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

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

2. 一个有效的 ROM BIOS 程序应具有如下的规定:

(1)第一字节内容为 55H

(2)第二字节内容为 AAH

(3)第三字节内容是长度指示器。ROM 的大小是以 512 字节块为单位。

(4)第四字节:EBH 一定是一条 JMP 跳转指令。

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

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

(下转 47 页)



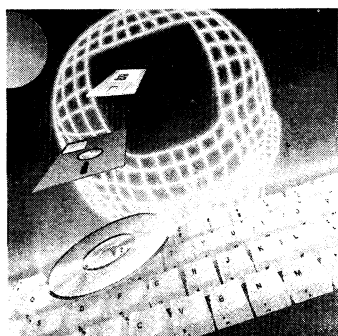


ISSN 1003-1077  
**電子**  
**與**  
**電腦**

# 5

一九九二年

总期第86期



# 電子與電腦

ELECTRONICS AND COMPUTERS

## 目 录

### · 综述 ·

91 年度 COMDEX 电脑博览会 ..... 吕问黎(2)

### · PC 用户 ·

获取 FOX 文件中密码的两种简单方法 ..... 周日初(3)  
 对全屏编辑命令 BROWSE 做一点补充 ..... 宋开胜(4)  
 感染 OMICROX 病毒文件的消毒方法 ..... 刘东祥(5)  
 COLOR400 卡屏幕彩色图形的硬拷贝 ..... 张辉 李东升(6)  
 利用格式化功能使磁盘“加锁” ..... 张李文(9)  
 一种有效的软件解密方法 ..... 张彦洪(12)  
 三维图形显示的遮掩技术及实现 ..... 季军杰(13)  
 BASIC 超长随机文件的处理方法 ..... 朱建平(15)

### · 学习机之友 ·

打印机的定位和格式控制 ..... 栾傲发(16)  
 放大打印 APPLE II 高分辨率图形又一法 ..... 刘善平(18)  
 趣味速算练习器 ..... 赵 旭(19)  
 计算机解决数学问题 ..... 赵方明(19)  
 巧改系统 ..... 张 浩(20)  
 数字游戏 ..... 闫 浩(20)  
 ProDos 磁盘操作系统入门(续) ..... 廖 凯(21)

### · 6502 机器语言讲座 ·

第五章 源程序的编辑、汇编和运行 ..... 朱国江(24)

### · 初级程序员级软件水平考试辅导 ·

数据库基础 ..... 阙 璐(29)

### · 学用单片机 ·

提高单片机最小系统抗干扰能力和自恢复方法

..... 张培仁、刘振安(32)

TP801 单板机与微型机数据双向并行传送一例

..... 刘浔和(33)

MCS-51 汇编指令外延的应用 ..... 陈亿善(34)

### · 学装微电脑 ·

制作能暂停显示的计数器 ..... 易齐干(35)

### · 电脑巧开发 ·

简易字幕灯的制作与控制 ..... 杨宪泽(38)

DOS 系统下多种软件在一台微机上的共存方法

..... 尤建忠(40)

### · 电脑游戏机 ·

第四章 F BASIC 语言的深入理解 ..... 于 春(41)

### · 维修经验谈 ·

TH3070 打印机常见故障分析与检修(上) ..... 刘立华(44)

### · 读者联谊 ·

普及型 PC 个人用户软件交流联谊活动问题解答(五)

..... 王路敬(47)

封一：电脑键盘

封二：让中华机发挥更大作用的中华机教学网(CENET)

封三：向读者提供“新一代 Z80”资料

封四：《LDQ 光笔》简介

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



# 91 年度 COMDEX 电脑博览会

吕问黎

美国拉斯维加斯每年举行的电脑界盛会——COMDEX 博览会是北美规模最大的个人计算机产品展销会。1991 年的博览会共有来自 100 多个国家的 18000 名参加者，数百家公司和一些国家如美国、印度、比利时等的代表。

微型软件公司 (Microsoft) 的著名软件设计师比尔·盖茨首先作了基调性发言《指尖的信息》，指明了个人电脑今后发展的四个实用方向：1. 办公室系统中的声音、视觉与传真组成的综合信息传输网；2. 为农业工人服务的数据和信息处理方面的电子装置；3. 家用计算机进行室内规划和设计厨房设施；4. 在学校辅助教学。盖茨的发言不仅是微型软件公司的战略目标，也是美国电脑业今后的发展方向。

在“软件出版者协会”主办的一个研讨会上，与会者们着重探讨了“手写输入计算机” (Pen-based Computer) 的发展前景，代表们注意到了首先进入市场的 Linus 手写系统。此外 Go 和 Slate 两家大公司在这方面的发展也引人注目，他们都在开发这种电脑的应用软件，最初的应用是帮助用户填写表格，然后就要研制便于携带的笔记本式电脑 (Note book Computer)。

本届博览会上，各种尖新产品层出不穷，厂家之间竞争激烈。

**便携式计算机** 便携式电脑的市场这几年增长神速，带来激烈竞争，包括 Intel 公司在内的 40 家企业正在研制 386 膝上式电脑和笔记本式电脑，其产品颇为类似，内存都是 1—2 兆字节，硬盘都是 20~50 兆，都有鼠标和 VGA 显示屏。APPLE 和 Intel 两家大公司也在研制高度综合化的便携式集成电路，这将使明天的电脑更小、更轻、更便宜。

**附加硬卡** 近年来，靠外围设备来辅助主机以提高效率的方法颇为盛行，普拉斯发展公司的 IIXL 硬卡代表了这种趋势。这种硬卡可以提供大量存储空间；并且读写速度非常快，高过以前常用的硬盘，表 1 显示了一个实例；它的安装也极为方便，操作人员当场表演了五分钟装上主机的技术，令在场的观众叹为观止。

**税单软件** 杰普软件公司推出了“特普税单”软件。这套软件在 APPLE 公司的 Mackintosh 系列机上运行，主要帮助用户处理纳税方面的事务。它包括多达 60 种纳税方式，为一些州的特种税也专门做了考虑，并且速度极快。对一般纳税人一年纳税情况，它只需一秒即可复查一遍。

**轻型板状传真机** 佛蒙特信息公司推出的这种轻型传真机为 9600 波特 (每秒传输 9600 比特)。这是电脑在硬件方面的出色运用。这种小板似的传真机可以存储一个电话簿，可以发送、接收、定向传递文件，而且

接收到的文件既可以显示在屏幕上，也可打印在纸上。产品的主要特点是小型化，而且价格低廉，仅为 195 美元。

**柯达的 PHOTO-CD 系统** 柯达公司的这项新产品是它在图象数字化方面的最新进展。PHOTO-CD 系统可以将 35mm 的胶片或幻灯片上的彩色图像直接经过数字化以后输入电脑内存，或显示在屏幕上，或打印在纸上，或存入激光磁盘。这项产品成功地实现了自然图象与电脑的数字图像之间的方便的直接的转换，而这一直就是人们的梦想，因为这样可以在电脑屏幕上看到电视里那种活生生的图像了。此外，柯达在这个系统中运用的数字化方式非常先进，以致一张磁盘可容纳彩色图像多达一百幅左右。

今天的美国电脑业竞争非常激烈。在新上市的 486 型计算机上，AMP 和 Intel 两家大公司展开了激烈角逐。由于 AMD 在上一代产品 386 型机上有相当优势，1990 年它还占有美国 286 型市场的 52%，所以估计 Intel 将吸取上次的教训，下大力推销产品。在操作系统这个广阔领域里，传统 DOS、扩展 DOS、UNIX 系统和 OS/2 系统展开了一场较量。迄今为止，最大的赢家是扩展 DOS，90 年秋其市场占有率达到了 40%；而 OS/2 则由 88 年的 32% 下降到 12%，以上情况都是在博览会上调查而得。

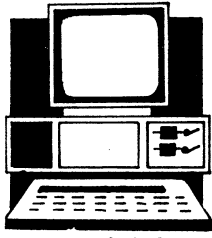
美国电脑业不仅内部竞争不断升级，还面临着外国公司的强大挑战。亚洲的南朝鲜以高士达为首的一批公司正决心把自己从 OEM 状态 (原始产品生产者) 中解脱出来，大力扩展海外销售，而且其产品极具竞争力。这次博览会上高士达公司的 386 型 16 位电脑及高分辨显示器都明显优于美国公司的产品。

COMDEX 电脑博览会是当今美国电脑业的一个缩影和窗口，它完整地显示了美国电脑业的特点，那就是：产品更替速度快、新颖独到的构思和你死我活的竞争。

表 1 IIXL 硬卡与 AST 硬盘读写速度比较

| 系统内存缓冲区 (KB) | 0      | 128   | 256   |
|--------------|--------|-------|-------|
| AST 硬盘       | 80.9 秒 | 77.1  | 53.9  |
| IIXL 硬卡      | 62.8   | 52.1  | 42.2  |
| 提高速度百分比      | 22.3%  | 32.5% | 21.6% |

在内存缓冲区的三种不同情况下，在读写同一个程序的速度上，IIXL 硬卡都明显占有优势。



# 获取 FOX 文件中密码的两种简单方法

石油天然气总公司三机厂研究所(431734) 周日初

## PC用户

多数用户乐于使用 FoxBASE 来替代 dBASE 的原因之一就在于 FoxBASE 具有一定程度的保密性,可将 PRG 文件经 Foxpcomp. exe 编译后生成后缀为 FOX 的文件,它用系统命令 TYPE 是看不懂其含义的。因此在编写密码程序时,往往将其编译生 FOX 文件,为了防止别人从源程序中获得密码就将原 PRG 文件删除掉。但如果忘了密码或想破译别人的 FOX 文件中的密码怎么办呢? 下面我介绍两种简单易行的办法:

### 一、增加一中断处理程序

增加中断处理程序的目的就是改变 SET 命令的设置,将原来的 OFF 状态改为 ON,以便看到文件的内容及其执行过程,具体步骤如下:

1. 进入 FoxBASE,建一中断处理程序 ESCERR. PRG,内容见程序 1。

2. 敲入 ON ESCA DO ESCERR 命令

3. 装好打印纸,接通打印机

4. 执行密码程序,并按<ESC>键。

5. 分析打印纸上打出的程序,判断其密码。

举例如下:

设有一密码程序,其源程序见程序 2,将其中的第四句改为 SET ESCA ON(因为这种方法只适用于将 ESCAPE 设为 ON 的情况)。

按以上五步执行后,打印纸上便记下密码程序运行时的每一条指令,当读到 IF A#“1”时便知道 1 为密码,然后重新执行密码程序,当提示输出密码时,敲入 1,程序便正常结束,证明密码正确。密码错时会接连响三次铃,这时要继续分析打印结果,有的需要经过一定的转换才能获得真正的密码。这种增加中断处理程序获得密码的方法只适合于将 ESCAPE 置为 ON 的情况,当为 OFF 时则要采取第二种方法。

### 二、借助 DEBUG 修改 FOX 文件

要获得密码,就要了解源程序。这里所说的获得源程序的方法就是将 FOX 文件中有关 SET 命令置为 OFF 的地方改为 ON 状态,以便能看到执行的每一步,FOX 文件中这些 SET 命令是看不见的,但我们可以用 DEBUG 得到它们编译后的目标码,详见表 1。例如: SET TALK OFF 编译后产生的目标码为 04 47 6A 51 FE, SET TALK ON 的目标码为:04 47 6A 52 FE,若要将 SET TALK OFF 改为 SET TALK ON,则用 DEBUG 的 S 命令先搜索一下代码 04 47 6A 51 FE,搜到后将对应的 51 改为 52 即可,再用 W 命令写回原文件,然后在 FoxBASE 状态下执行。现举例如下:

密码程序见程序 2,经编译后产生的 FOX 文件在 DEBUG 状态下显示的结果见表 2,查看表 1 可知:只要将 120~13F 间的 4 个 51 用 E 命令改为 52,便将程

表 1

| 设置项   | 设置为 ON/OFF 时的目标码 |                |
|-------|------------------|----------------|
|       | ON               | OFF            |
| ALTE  | 04 47 16 52 FE   | 04 47 16 51 FE |
| CARR  | 04 47 1F 52 FE   | 04 47 1F 51 FE |
| CLEA  | 04 47 79 52 FE   | 04 47 79 51 FE |
| CONF  | 04 47 27 52 FE   | 04 47 27 51 FE |
| DEBU  | 04 47 2C 52 FE   | 04 47 2C 51 FE |
| DELI  | 04 47 2F 52 FE   | 04 47 2F 51 FE |
| ECHO  | 04 47 33 52 FE   | 04 47 33 51 FE |
| EXAC  | 04 47 37 52 FE   | 04 47 37 51 FE |
| FIEL  | 04 47 3B 52 FE   | 04 47 3B 51 FE |
| HEAD  | 04 47 43 52 FE   | 04 47 43 51 FE |
| HIST  | 04 47 BD 52 FE   | 04 47 BD 51 FE |
| MENU  | 04 47 9B 52 FE   | 04 47 88 51 FE |
| SAFE  | 04 47 8B 52 FE   | 04 47 8B 51 FE |
| STAT  | 04 47 65 52 FE   | 04 47 65 51 FE |
| TALK  | 04 47 6A 52 FE   | 04 47 6A 51 FE |
| BELL  | 04 47 1B 52 FE   | 04 47 1B 51 FE |
| CENT  | 04 47 90 52 FE   | 04 47 90 51 FE |
| COLO  | 04 47 25 52 FE   | 04 47 25 51 FE |
| CONS  | 04 47 28 52 FE   | 04 47 28 51 FE |
| DELE  | 04 47 2E 52 FE   | 04 47 2E 51 FE |
| DOHI  | 04 47 C4 52 FE   | 04 47 C4 51 FE |
| ESCA  | 04 47 36 52 FE   | 04 47 36 51 FE |
| EXCL  | 04 47 CE 52 FE   | 04 47 CE 51 FE |
| FIXE  | 04 47 86 52 FE   | 04 47 86 51 FE |
| HELP  | 04 47 97 52 FE   | 04 47 87 51 FE |
| INTE  | 04 47 46 52 FE   | 04 47 46 51 FE |
| PRINT | 04 47 57 52 FE   | 04 47 57 51 FE |
| SCOR  | 04 47 8C 52 FE   | 04 47 8C 51 FE |
| STEP  | 04 47 66 52 FE   | 04 47 66 51 FE |
| UNIQ  | 04 47 8D 52 FE   | 04 47 8D 51 FE |

注:以上数值均为 16 进制数

序 2 中最前面的四条 SET 命令置为 ON 状态,然后用 DEBUG 的 W 命令写回到 FOX 文件中,重新执行,仿照第一种方法,把执行结果打在纸上(先执行 SET DEVI TO PRIN 和 SET PRIN ON),再分析密码。

用这两种方法,不仅可以方便地找到密码,而且还可以获得 FOX 文件对应的 PRG 程序。

表 2

```

-r
AX=0000 BS=0000 CX=00D9 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=3000 ES=3000 SS=3000 IP=0100      MV UP EI PL NZ NA PO NC
3000:0100 FB      STI
-d100 1d9
3000:0100 FB 2B 02 00 00 00 00 00-00 00 00 00 00 00 00 00 .+.
3000:0110 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
3000:0120 00 00 9F 00 04 47 6A 51-FE 04 47 65 51 FE 04 47 ...GJQ..GEQ..G
3000:0130 33 51 FE 04 47 36 51 FE-0B 54 F7 00 00 10 FC F8 3Q..G61..T....
3000:0140 01 01 FD FE 1B 04 FC F8-02 0A FD 07 FC F8 02 14 .....
3000:0150 FD 5F FC FB 09 CA E4 C8-EB C3 DC C2 EB 3A FD FE .-.....
3000:0160 10 18 73 FC F7 00 00 F8-01 04 0D FD F9 05 9A 00 ..s.....
3000:0170 FE 04 47 28 51 FE 06 05-6C F7 01 00 FE 04 47 28 ..G(Q..1....G(
3000:0180 52 FE 0F 25 FC F7 01 00-FB 01 31 0F FD F9 05 7D R..%. ....1....}
3000:0190 00 FE 0E 02 FC 78 78 F8-01 07 22 F8 01 03 5C FD ....XX..."..\.
3000:01A0 FE 06 1B F9 05 87 00 FE-02 21 FE 02 1E FE 0F 54 .....!. ....T
3000:01B0 F7 00 00 10 FC F7 00 00-F8 01 01 06 FD FE 02 1D .....
3000:01C0 FE 01 55 02 00 4B 4B 00-00 00 00 00 00 00 41 ..U..KK.....A
3000:01D0 00 00 00 00 00 00 00 00-00 C4 .....

```

程序 1

```

* ESCERR. PRG
SET TALK ON
SET DEBUG ON
SET STAT ON
SET CONS ON
SET ECHO ON
SET PRIN ON
SET DEVI TO ON
程序 2
SET TALK OFF
SET STAT OFF
SET CEHO OFF
SET ESCA OFF
KK=1

```

```

@ 10,20 SAY! 输入密码:
DO WHIL KK<4
SET CONS OFF
ACCE TO A
SET CONS ON
IF A(<)'1'
? REPL(CHR(7),3)
ELSE
EXIT
ENDI
KK=KK+1
ENDD

```

对全屏编辑命令 BROWSE 做一点补充

广州中山二路 48 号之二 1702 室 宋开胜

现已出版了多种 dBASE III 书籍,但纵观各种书籍,对全屏编辑命令 BROWSE 的介绍都是一种模式 BROWSE[FIELDS<字段名>]。在使用中按照这一模式进行全屏输入、修改时经常会遇到一些操作失误,尤其是那些不懂计算机操作的人,对于多个字段宽度超过 80 个字符的数据库,进行全屏输入、修改操作时,往往不知道利用 CTRL+HOME 来锁定字段,再利用 CTRL+(←或→)调出屏幕之外的字段。对于某一个字段的输入、修改操作,如不利用 CTRL+HOME 来选择字段,操作失误也时有发生。

本人在实际应用中发现它有多种模式,在以往各种书籍中都没有介绍过,现补充如下:

- ①BROWSE[FIELDS<字段名>]
- ②BROWSE [FIELDS<字段名>][LOCK #] # :是数字(1,2 ...)
- ③BROWSE [FIELDS<字段名>][FREEZE<字段名>]



④BROWSE [FIELDS<字段名>][LOCK #FREEZE<字段名>]]

利用以上几种模式,可以代替 CTRL+HOME 来锁定字段和字段选择。

功能:①是通用型书籍中已有说明。

②从某一字段开始锁定(#:1,2...)用 CTRL+←(或→)左右控制移动字段。

③固定对其中某个字段输入、修改并能保护其他字段误操作。

④同时使用②、③的功能。

附上使用范例

数据库:DATA.DBF

| 字段 | 字段名 | 类型        | 宽度 | 小数 |
|----|-----|-----------|----|----|
| 1  | A1  | Character | 8  |    |
| 2  | A2  | Numeric   | 6  | 2  |
| 3  | A3  | Numeric   | 6  | 2  |
| 4  | A4  | Numeric   | 6  | 2  |
| 5  | A5  | Numeric   | 6  | 2  |
| 6  | A6  | Numeric   | 6  | 2  |

USE DATA

BROWSE FIELDS A1,A2,A3,A4,A5,A6 LOCK 2

字段 A1 和 A2 被锁住,用 Ctrl+←(或→)可以控制 A3,A4,A5,A6,左右移动

BROWSE FIELDS A1,A2,A3,A4,A5,A6FREEZE

A3

固定对字段 A3 输入、修改,保护其他字段误操作  
BROWSE FIELDS A1, A2, A3, A4, A5, A6 LOCK 2  
FREEZE A5

锁住字段 A1 和 A2,用 Ctrl+←(或→)可以控制 A3,  
A4,A5,A6 左右移动,只对字段 A5 输入、修改,保护其  
他字段误操作

## 感染 OMICRON 病毒文件的消毒方法

上海同济大学 8736(200092) 刘东祥

计算机病毒作为一类具有隐蔽性、传染性和破坏性的特殊程序,日益严重地干扰着用户的正常工作。以往的病毒按其寄生的部位,可分为引导区和执行文件病毒两大类。现在市场上流行的 ANTI、SCAN 或 TNTVIRUS 都可以消除这两类病毒的绝大多数。但是最近我国电脑软件中又发现了一种病毒,它区别于以往的单宿主病毒,可以同时寄生在硬盘主引导区和执行文件中,所以传染速度极为疯狂。经破译后,发现该病毒的显示信息有一段文字很奇特,为 OMICRON by PsychoBlast (OMICRON 为希腊语中的第十五个字母),所以常称之为 OMICRON 病毒。

感染这种病毒之后的文件将增加 2153 个字节。用 ANTI、SCAN 或 TNTVIRUS 都无法将它消除。分析其病毒程序,可见作者的编程技巧相当强,最大特点是它利用时钟产生的随机数作为密码钥,所以表面上看,不同时间感染所追加的病毒程序也不相同。

OMICRON 病毒发病时间是在每月 2 日上午 10 点,屏幕上的内容将上下左右相互对换。该病毒并没有多么深重恶劣的破坏功能,属良性病毒一类。

下面就详细介绍对 OMICRON 病毒的解毒方法。

首先用一张干净的系统盘启动机器,盘上要有 DEBUG.COM 文件。

COM 文件的解毒方法很简单,运行 DEBUG 调入染毒 COM 文件,程序的第一个语句为 JMP 语句,跳转地址记为 Addr,找出 Addr+9 处的一个字节的值,记为 KEY,KEY 即密码钥。原正常文件的长度为 Addr-821-100(HEX),然后再查出 Addr-821+BF(HEX)起始连续三个字节,记为 B1、B2、B3 将 B1、B2、B3。分别加上 KEY 之后依次放到 COM 文件开头,修改 CX、BX 为原正常 COM 文件的长度,记盘退出。

EXE 文件的消毒工作稍为麻烦一些,可分为以下四步:

(一)运行 DEBUG,调入 EXE 文件,用 G 功能运行到病毒程序中第一个 LOOP 语句后的 JMP 语句,再用 T 功能运行完随后的 CALL 语句、POP SI、SUB SI,0113 后止,查看 SI,用笔记录 CS:SI+C2 始的六个字节。注意,SI+C2、SI+C3 保存原 IP,SI+C4、SI+C5 与原 CS 有关,而 SI+C6、SI+C7 与原 SS 有关,退出 DEBUG。

(二)改 EXE 文件为以 BIN 为后缀的文件。

(三)再进入 DEBUG 调改名后的文件,将病毒文件长度减去 2153(DEC)个字节后,修改 CX、BX 为相应的值,同时修改 CS:102—CS:105 四个字节,如病毒文件长度为 27461,减去 2153 后为 25308,25308 个字节将占 50(HEX:32)个扇区,但最后的一个扇区只用了 220(HEX:DC)个字节,所以在 CS:102 处写 DC,CS:103 处写 00,CS:104 处写 32,CS:105 处写上 00,另外在 CS:10E 上填 [SI+C6]-10(HEX),CS:10F 上填 [SI+C7],CS:114 上写 [SI+C2],CS:115 上填 [SI+C3],CS:116 上写 [SI+C4]-10(HEX),CS:117 上填写 [SI+C5],之后记盘退出。

(四)将 BIN 文件恢复为 EXE 可执行文件。

值得注意的是,病毒文件可能是经过多次感染的,所以要经检查无毒后方可运行,若依然有毒,要做多次消毒工作。

如果机器带有硬盘,很有必要检查硬盘主引导区是否已经感染。如果在此之前运行过带毒程序,则主引导区肯定被感染。只要用硬盘主引导区的备份将之覆盖即可。

具体过程如下:

A:\>DEBUG C.INI (C.INI 为硬盘主引导区备份)

```
-A400
3212:0400 PUSH CS
3212:0401 POP ES
3212:0402 MOV AX,0301
3212:0405 MOV BX,0100
3212:0408 MOV DX,0080
3212:040B MOV CX,0001
3212:040E INT 13
3212:0410 INT 20
3212:0412
-RIP
IP 0100
:400
-G
-Q
```

我们发现 OMICRON 病毒可以破坏硬盘分区表,因此不能视为良性病毒。美国 CENTRAL POIN 公司的抗病毒软件 CPAV 可以消除 OMICRON 病毒,关于 CPAV 的编者使用本刊将在近期另文介绍。

# Color 400卡屏幕彩色图形的硬拷贝

东北工学院319信箱(110006) 张 辉 李东升

Color400彩色图形卡(又称 CGE400)是为 IBM PC 显示适配器的软硬件最大程度相兼容而设计的。它可提供640×400象素的具有16种色彩的高分辨率彩色图形,目前已在我国广泛应用。可令人遗憾的是,用Color400卡显示的漂亮图形,只能在屏幕上显示,而不能从打印机上打印出来。为了解决这一问题,我们以国内广泛使用的 M2024和 M1724打印机为例,介绍了Color400卡彩色图形的16种灰度硬拷贝的编程方法。

## 一、屏幕象素色彩的灰度转换

Color400卡的640×400高分辨率彩色图形模式可以显示16种颜色的象素。象素彩色值的获取可通过BIOS的INT10H的13号功能来实现。获取象素彩色值的8088汇编语言程序如下:

```
MOV AH,13
MOV DX,ROW
MOV CX,COL
INT 10H
```

程序中 ROW 和 COL 分别为象素的行和列,象素的彩色值在 INT 10H 返回时存放在 AL 寄存器中,其值为0~15。

由于打印机只能打印黑白两种颜色,所以获得象素的彩色值后,还需将彩色值转换为相应灰度值的打印机输出数据。由于 M2024/M1724打印机的最大横向打印列数为2716,所以我们设计了3×4点阵中点的排列,可获得16种灰度。为了数据的处理方便,我们用3个字节数据来表示一个屏幕象素的灰度值输出数据。如深灰象素的灰度图和输出数据如图1所示。象素彩色与打印机灰度输出数据的对照见表1。

表1 象素彩色与打印机灰度输出数据

|          |      |                              |    |    |                              |    |    |                              |    |    |   |    |    |    |    |    |    |
|----------|------|------------------------------|----|----|------------------------------|----|----|------------------------------|----|----|---|----|----|----|----|----|----|
| · · ·    | 彩色   | 黑                            | 兰  | 绿  | 青                            | 红  | 洋  | 棕                            | 淡  | 深  | 浅 | 浅  | 浅  | 淡  | 品  | 黄  | 白  |
| · · ·    | 色值   | 0                            | 1  | 2  | 3                            | 4  | 5  | 6                            | 7  | 8  | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| · · ·    | 打印数据 | 0F0604000E0606020F0E020D0604 | 00 | 00 | 0F0D01040S0F09090F0F0F0B0F01 | 04 | 00 | 0F0604020B0606060707060D0704 | 00 | 00 |   |    |    |    |    |    |    |
| 0F 0F 07 | 数据   |                              |    |    |                              |    |    |                              |    |    |   |    |    |    |    |    |    |

图1 灰度打印数据

## 二、打印机的设置

M2024/M1724打印机为24针点阵式打印机,故每一纵列打印数据按3个字节传送。当送出的图形打印数据中的某一位为“1”时,对应的打印针动作,为“0”时打印针不动作。为了输出图形必须首先设置好相应的行距,然后将打印机置成图形模式。打印图形时可选择24针同时打印或只用其中的8针打印,选择24针

打印时,由于易使打印头产生高温影响打印速度。所以我们选用其中的8针进行打印,即送一个字节的数据后,紧接着送2个“0”字节。

确定了使用的打印针数后,就可以设置打印行距。使用8针时,打印机的行距设置为6/120英寸比较合适。这样设置后打印机打印出的图形既不断开也不重迭。其十六进制控制码为:1B 4A 06。24针打印机在图形模式下每行输出的一列有24个点,即每3个字节的图形打印数据对应某打印行的一列。图形打印数据与打印针的对应关系及控制码如图2所示。

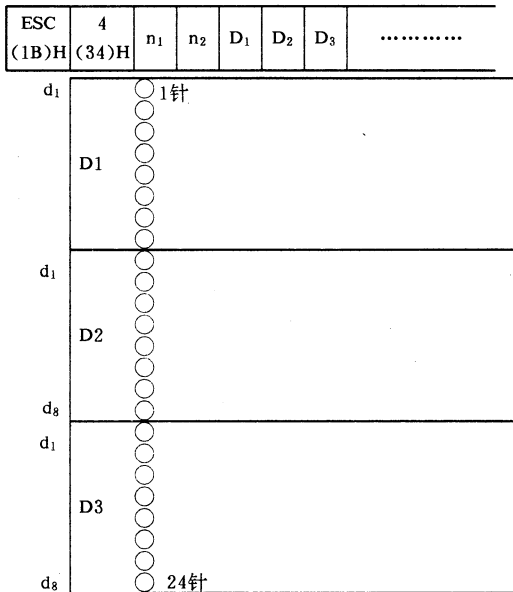


图2 图形打印数据与打印针对应关系

对于打印640×400的屏幕图形,如果用3×4点阵对应于屏幕上的一个象素,则打印机打印的列数为3×640=1920,对应的打印机控制码为:

1B 34 07 80 D1 D2.....D5760,其中 D1~D5760为打印图形数据。第二个代码34表示双向打印,若改为47则为单向打印。打印机收到以上5764个数据后仍然处于等待状态,还必须再向打印机送回车(0DH)和换行(0AH)的控制码,打印机才打印缓冲区中的数据。

## 三、程序实例

根据以上方法,我们用8088汇编语言编制了Color400卡彩色屏幕图形拷贝程序 PRTSCR.COM,其源程序清单如附录所示。

在绘制图形前首先执行 PRTSCR.COM, 则该程序将常驻内存, 直至计算机重新启动。屏幕图形绘制完毕后, 按下拷屏幕键 (prtsc 键) 即可在打印机上拷贝出具有 16 种灰度的精美图形。拷贝的图形例子如图 3 所示。本方法也适于其它彩显卡 (如 EGA, VGA) 的彩色图形拷贝。



图3 640×400屏幕1/4图形拷贝

附录: 屏幕拷贝程序 PRTSCR.COM 源程序清单

CODE SEGMENT PARA 'CODE'

ORG 100H

START PROC FAR

ASSUME CS, CODE

ASSUME DS, CODE, ES, CODE

JMP CCCC

; 拷贝屏幕中断服务程序

PRTS PROC NEAR

STI

PUSH DS

PUSH ES

PUSH DI

PUSH SI

PUSH BX

PUSH CX

PUSH DX

PUSH AX

MOV AX, CS

MOV DS, AX

MOV ES, AX

; 查找打印机号

XOR DX, DX

Z: MOV AH, 2

INT 17H

AND AH, 10H

CMP AH, 10H

JZ ZZ

INC DX

JMP Z

; 确定屏幕状态, 如不是640x400图形模式则退出

ZZ: MOV SI, DX

MOV AH, 15

INT 10H

CMP AL, 42H

JNZ EXIT

; 置打印机行距

MOV AL, 0AH

CALL PRI

MOV AL, 0DH

CALL PRI

MOV AL, 1BH

CALL PRI

MOV AL, 4AH

CALL, PRI

MOV AL, 6

CALL PRI

XOR DX, DX

XOR AX, AX

; 每次打印屏幕上的二行图形

P42: CALL PRCON

CALL PR640

CMP AX, 200

JZ P41

ADD DX, 2

INC AL

JMP P42

; 完成屏幕拷贝中断服务返回

P41: MOV AL, 18H

CALL PRI

CALL CRLF

MOV AL, 0FH

CALL PRI

EXIT: MOV AL, 20H

MOV DX, 20H

OUT DX, AL

POP AX

POP DX

POP CX

POP BX

POP SI

POP DI

POP ES

POP DS

IRET

PRTS ENDP

; 输出两行屏幕像素至打印机子程序

PR640 PROC NEAR

PUSH CX

PUSH AX

XOR CX, CX

P2: CALL READ2

CMP CX, 639

```

        JZ P1
        INC CX
        JMP P2
P1:    CALL CRLF
        POP AX
        POP CX
        RET
PR640 ENDP
; 输出回车和换行命令子程序
CRLF  PROC NEAR
        PUSH AX
        MOV AL,0AH
        CALL PRI
        MOV AL,0DH
        CALL PRI
        POP AX
        RET
CRLF  ENDP
; 置打印机图形状态子程序
PRCON PROC NEAR
        PUSH AX
        MOV AL,1BH
        CALL PRI
        MOV AL,47H
        CALL PRI
        MOV AL,7
        CALL PRI
        MOV AL,80H
        CALL PRI
        POP AX
        RET
PRCON ENDP
; 向打印机输出 AL 中数据子程序
PRI   PROC NEAR
        PUSH AX
        PUSH DX
        MOV AH,0
        MOV DX,SI
        INT 17H
        POP DX
        POP AX
        RET
PRI   ENDP
; 象素彩色值灰度转换子程序
READ1 PROC NEAR
        PUSH CX
        PUSH DX
        PUSH SI
        PUSH BX
        MOV AH,13
        INT 10H
        MOV AH,0
        PUSH BX
        MOV BX,AX
        SHL BX,1

```

```

        ADD AX,BX
        POP BX
        LEA SI,GRAY
        ADD SI,AX
        MOV CX,3
        REP MOVSB
        POP BX
        POP SI
        POP DX
        POP CX
        RET
READ1 ENDP
; 灰度数据
GRAY  DB 0FH,0FH,0FH
        DB 06H,0DH,06H
        DB 04H,01H,04H
        DB 00H,04H,02H
        DB 0EH,05H,0BH
        DB 06H,0FH,06H
        DB 06H,09H,06H
        DB 02H,09H,06H
        DB 0FH,0FH,07H
        DB 0EH,0FH,07H
        DB 02H,0FH,06H
        DB 0DH,0BH,0DH
        DB 06H,0FH,07H
        DB 04H,01H,04H
        DB 00H,04H,00H
        DB 00H,00H,00H
READ2 PROC NEAR
        PUSH DX
        PUSH CX
        MOV BX,2
        LEA DI,BUF
OOP1: CALL READ1
        INC DX
        DEC BX
        JNZ OOP1
        MOV CX,3
        MOV BX,OFFSET BUF
OOP2: PUSH CX
        PUSH BX
        MOV AL,[BX]
        ADD BX,3
        MOV AH,[BX]
        SHL AH,1
        SHL AH,1
        SHL AH,1
        SHL AH,1
        PUSH DX
        MOV DL,AH
        MOV DH,0
        MOV AH,0
        ADD AX,DX
        CALL PRI

```



```

POP DX
MOV AL,0
CALL PRI
MOV AL,0
CALL PRI
POP BX
POP CX
INC BX
DEC CX
JNZ OOP2
POP CX
POP DX
RET
READ2  ENDP

```

```

BUF  DB 18 DUP(0)
; 修改 INT 5 中断向量并使以上程序常驻内存
CCCC: PUSH DS
      PUSH CS
      POP DS
      MOV AX,2505H
      MOV DX,OFFSET PRTS
      INT 21H
      MOV DX,OFFSET CCCC
      INT 27H
      POP DS
START ENDP
CODE  ENDS
      END START

```

## 利用格式化功能使磁盘“加锁”

上海科学教育出版社(200233) 张李文

合法用户为了存档或机械的原因而复制软件是正常的。然而正因为复制软件十分简单,以至于许多用户为朋友或出售而进行额外非法复制。这样的非法复制使软件商难以获利,同时也极大地打击了程序员的积极性。为此许多软件公司为保护软件不被非法复制,大力研究软件保护技术。在五花八门的保护技术中,磁盘“加锁”课题一直以简便易行、成本低和效果好而受到人们的青睐。

### 一、磁盘结构

磁盘有两个面,每面由一条条同心圆的磁道构成,每磁道又被划分成一段段扇区。360K 软盘有两面,每面有40磁道,每磁道有9扇区,每扇区有512字节;1.2M 软盘有两面,每面有80磁道,每磁道有15扇区,每扇区有512字节。每张磁盘上还有一个索引孔。磁道的起点是在索引孔附近,终点也在索引孔附近(绕过一圈),这是硬件规定的(以下提供的数据均对360K 软盘而言)。

逻辑0扇区被规定用户不能使用。里面存放着磁盘的重要信息:一段小巧精悍的引导程序(存放着磁盘格式字、规格表和磁盘自举程序)。其它扇区还有文件分配表(存放着每个文件的簇链接表)、文件目录表(登记磁盘文件目录)。利用这几个重要文件也能成功地对磁盘加锁,如:修改文件目录表某一项的0BH 字节,使之产生隐含、系统等不在 DIR 中列出来的文件,或修改文件目录表的1AH 和1BH 字节,使文件丢失起始簇而无法使用。这些方法简便而颇具效果。

磁道的起始处含有50H 个4EH、0CH 个00H、3H 个C2H、1H 个FCH、32H 个4EH 的代码用来表示磁道开头。然后是前置部、扇区部和后置部三个部分。前置部紧接在索引孔后,作为扇区部数据的缓冲。扇区部是用户真正可以填写数据的地方,由两部分组成:1D 部分与数据部分。1D 部分是在磁道格式化时已确定不再改变的,用来标示扇区位置。包含有磁道号、磁头号、扇区号、扇区数据长度和 CRC 校验码。数据部分则是实际存放数据的地方,包含有 SYNC、DAM (DDAM)、数据和 CRC。后置部位于磁道的末尾与索引孔之间,全部为间隙,由4EH 代码填充,其字节数与扇区数据的多少及磁盘转速有关。

### 二、保护技术

防止复制的思想是制造一些有“缺陷”或“不合规则”的磁盘,使复制被迫中止或复制后的信息仍无法使用。下面我们介绍一种利用格式化功能来加锁的方法,它简便易行,效果不错。

我们知道磁盘读/写操作和格式化依赖于 BIOS 的 INT 13H。参数如下:

- ①入口参数:AH=02H(读磁盘扇区到内存)
- AL=传送扇区数(1~9H)
- ES;BX=用户盘 I/O 缓冲区段地址:偏移量
- CH=磁道号(0~27H)
- CL=扇区号(1~9H)
- DH=磁头号(0~1H)

DL=驱动器号(0~3H)

出口参数:成功:AH=0

AL=实际传送的磁盘扇区数

进位标志清0

失败:AH=状态字节

进位标志置1

②入口参数:AH=03H(从内存写盘)

AL=传送扇区数(1~9H)

ES:BX=用户盘 I/O 缓冲区段地址:偏移量

CH=磁道号(0~27H)

CL=扇区号(1~9H)

DH=磁头号(0~1H)

DL=驱动器号(0~3H)

出口参数:成功:AH=0

AL=实际传送的磁盘扇区数

进位标志清0

失败:AH=状态字节

进位标志置1

③入口参数AH=05H(格式化磁道)

AL=格式化扇区数(1~9H)

ES:BX=4字节参数段地址:偏移量

CH=磁道号(0~27H)

CL=扇区号(1~9H)

DH=磁头号(0~1H)

DL=驱动器号(0~3H)

出口参数:成功:AH=0

AL=实际格式化磁盘扇区数

进位标志清0

失败:AH=状态字节

进位标志置1

\* 4字节参数地址区域含义如下:

字节1:磁道柱面号 字节2:磁头号

字节3:扇区号 字节4:每扇区字节数

( 0=128字节/扇区 1=256字节/扇区 )

( 2=512字节/扇区 3=1024字节/扇区 )

\* 错误状态字节内容如下:(保存在寄存器内)

00H 未出错。

01H 非法功能调用命令。

02H 地址标记损坏,扇区标识(ID)无效或未找到。

03H 企图对有写保护的磁盘执行写操作。

04H 可能由于扇区号太大,所找的扇区没找到。

05H 复位操作失败。

06H 无介质。

07H 初始化错误,驱动器参数无效。

08H DMA 故障

09H DMA 边界错误。

0AH 标测出错误的扇区标志。

10H CRC 奇偶校验错且 ECC 码不能纠正。

11H 读磁盘时奇偶校检错,ECC 已纠正错误。

20H 控制器故障。

40H 查找操作无效。

80H 超时错误,驱动器不响应。

AAH 驱动器未准备好。

BBH 未定义错误。

CCH 被选驱动器出现写故障。

FFH 读出操作失败(只对 XT)。

### 三、范例

这里我们利用上述的功能做一张钥匙盘,这张盘是利用格式化功能制做的不能被拷贝的加锁盘。每次启动硬盘时用这张钥匙盘来打开 COMMAND.COM 程序,使硬盘正常工作。

具体步骤如下:(准备好一张带 DEBUG.COM 程序的 DOS 盘 V3.20)

步骤 A(格式化一张钥匙盘):

①格式化一张盘片,然后放入 B 驱动器;

②将带有 DEBUG.COM 程序的 DOS 盘放入 A 驱动器;

③输入如下命令:

```

A>DEBUG✓
-A 100✓
  ××××:0100 MOV AX,0524✓
  ××××:0103 MOV BX,0200✓
  ××××:0106 MOV CX,0901✓
  ××××:0109 MOV DX,0001✓
  ××××:010C INT 13✓
  ××××:010E MOV AX,0524✓
  ××××:0111 MOV BX,0300✓
  ××××:0114 MOV CX,1601✓
  ××××:0117 MOV DX,0101✓
  ××××:011A INT 13✓
  ××××:011C MOV AX,0524✓
  ××××:011F MOV BX,0400✓
  ××××:0122 MOV CX,1701✓
  ××××:0125 MOV DX,0101✓
  ××××:0128 INT 13✓
  ××××:012A INT 20✓
  ××××:012C✓

```

-R CX✓

CX 0000

:0050✓

-N ABC.COM✓

-W✓

Writing 0050 bytes

-Q✓

(A 盘中将生成 ABC.COM 文件,它是专门用来制做钥匙盘的,它的功能是毁坏0面9道、1面16道、1面17道,使之在读盘时产生返回码为04H 错误)。

A>ABC✓

等红灯全灭之后,B驱动器内的这张盘已经被格式化好了,给它贴上写保护(不要将它揭下,因为写保护也是一条判断条件)。它是一张不能被拷贝的钥匙盘,我们将用它来打开 COMMAND.COM 程序。

步骤 B:(修改 COMMAND.COM 程序)

```
①A>DEBUG C:COMMAND.COM
-A 100
XXXX:0100 JMP 5100
XXXX:0103
-A5100
XXXX:5100 MOV AX,0201
XXXX:5103 MOV BX,0200
XXXX:5106 MOV CX,0901
XXXX:5109 MOV DX,0001
XXXX:510C INT 13
XXXX:510E CMP AH,04
XXXX:5111 JNZ 5100
XXXX:5113 MOV AX,0201
XXXX:5116 MOV BX,0200
XXXX:5119 MOV CX,1601
XXXX:511C MOV DX,0101
XXXX:511F INT 13
XXXX:5121 CMP AH,04
XXXX:5124 JNZ 5100
XXXX:5126 MOV AX,0201
XXXX:5129 MOV BX,0200
```

```
XXXX:512C MOV CX,1701
XXXX:512F MOV DX,0101
XXXX:5132 INT 13
XXXX:5134 CMP AH,04
XXXX:5137 JNZ 5100
XXXX:5126 MOV AX,0301
XXXX:5129 MOV BX,0200
XXXX:512C MOV CX,1701
XXXX:512F MOV DX,0101
XXXX:5132 INT 13
XXXX:5134 CMP AH,03
XXXX:5137 JNZ 5100
XXXX:513C JMP 0CE0
-W
-Q
```

硬盘内 COMMAND.COM 程序已被修改好。每次启动硬盘时将钥匙盘插入 B 驱动器,启动将顺利通过。如果别人利用其它磁盘假冒钥匙盘将导致死循环。

限于版面,这里介绍的仅仅是一个利用格式化功能的小例子,其实它有着很强大的功能。我们可以利用它做一些其它的加锁,如:一条磁道中具有几种不同长度的扇区,每扇区都有自己正确的 CRC。这样的磁道是不可能用  $\mu$ PD765A 来拷贝的。相信读者在阅读之后肯定能想出一些更复杂、效果更好的方法来。

\*\*\*\*\*

(上接第20页)

```
60 VTAB I * 4;HTAB J * 5 + 1;PRINT A(I,J);C=C+1;
NEXT J,I
70 VTAB 3;HTAB 2;PRINT E;VTAB 14;HTAB 26;PRINT
"Please Remember"
80 FOR I=1 TO 250;T=INT(RND(1) * 4) + 1;ON T GOSUB
140,150,160,170;NEXT I;VTAB 14;HTAB 26;PRINT"
";T=0
90 FOR I=1 TO 5;FOR J=1 TO 4;VTAB I * 4;HTAB J * 5 +
1;PRINT A(I,J);NEXT J,I;VTAB X * 4;HTAB Y * 5 +
1;FLASH;PRINT E;NORMAL
100 B=B+1;PRINT CHR$(7);VTAB 20;PRINT B;IF B=
150 THEN 190
105 GET C$;IF C$="I" THEN GOSUB 140;GOTO 130
110 IF C$="J" THEN GOSUB 150;GOTO 130
115 IF C$="K" THEN GOSUB 160;GOTO 130
120 IF C$="M" THEN GOSUB 170;GOTO 130
125 GOTO 105
130 C=10;FOR I=1 TO 5;FOR J=1 TO 4;IF A(I,J)/C=1
THEN C=C+1;NEXT J,I;GOTO 200
135 GOTO 100
140 IF X>1 THEN A(X,Y)=A(X-1,Y);A(X-1,Y)=E;
```

```
X=X-1;IF T=0 THEN X1=X+1;Y1=Y;GOSUB 180
145 RETURN
150 IF Y>1 THEN A(X,Y)=A(X,Y-1);A(X,Y-1)=E;
Y=Y-1;IF T=0 THEN X1=X;Y1=Y+1;GOSUB 180
155 RETURN
160 IF Y<4 THEN A(X,Y)=A(X,Y+1);A(X,Y+1)=E;
Y=Y+1;IF T=0 THEN X1=X;Y1=Y-1;GOSUB 180
165 RETURN
170 IF X<5 THEN A(X,Y)=A(X+1,Y);A(X+1,Y)=E;
X=X+1;IF T=0 THEN X1=X-1;Y1=Y;GOSUB 180
175 RETURN
180 VTAB X1 * 4;HTAB Y1 * 5 + 1;PRINT A(X1,Y1);VTAB
X * 4;HTAB Y * 5 + 1;FLASH;PRINT E;NORMAL;RE-
TURN
190 VTAB 14;HTAB 28;PRINT"Try Again!";FOR I=1 TO
1000;NEXT I;GOTO 10
200 VTAB 14;HTAB 28;PRINT"Game Over";IF B/A < 1
THEN A=B;A$=B$;VTAB 6;HTAB 28;PRINT
"HI.";A;" ";A$
210 VTAB 17;HTAB 29;PRINT"Hit 'S'!";GET C$;IF C$
="S" THEN 10
220 GOTO 210
```

# 一种有效的软件解密方法

扬子石化公司信息处 (210048) 张彦洪

软件开发为了保护其软件的版权和经济利益,防止使用者方便地复制其软件,因而将大多数出售给用户的商品化软件进行了众多不同方式的加密。这种措施虽然有效地维护了软件开发者的利益,但同时也给许多合法的软件用户带来诸多的不便,例如:用户购买一份软件后无法为其备份;一个单位如有多机使用则必须购买多份;对引进的软件无法进行解剖分析和消化吸收等等。为了解决这样的问题,许多用户都努力寻找能有效地对加密软件进行解密的方法。笔者在在实践中经过长期摸索和研究,得出了一种切实可行且有效的软件解密方法,本文将对这种方法作一详细介绍。

## 一、解密的原理

凡是具有一定高级程序设计经验的软件人员都知道,目前市场出售的软件的加密方法有许多,如:激光加密法,密钥加密法,等等,种类五花八门,加之繁衍变种不计其数,常常令解密者费尽心机。虽然如此,对加密软件进行解密的有效方法还是存在的,因为众所周知,不管软件用何种方法进行过加密处理,只要它进入计算机内存运行,就必须进行自脱密处理,还其原代码形式,以便计算机识别并执行其指令。而且许多软件加载后是常驻内存的,即便是非常驻内存,但只要该软件没有使用动态复盖技术,那么它退出后在内存中依然会有其影子存在。本文所要介绍的解密方法就是根据这一基本原理实现的,下面以经过激光加密处理的压缩汉字库 CCBIOS 软件为例进行详细说明:

压缩汉字库 CCBIOS 系统实际上就是一个 CCDOS 系统,既然如此,那么压缩汉字库 CCBIOS 就必然有与 CCDOS 类似的汉字实现处理结构,也就是说它一定有以下几个处理模块:

- a. 汉字的文件管理
- b. 汉字的设备管理
  - \* 汉字输出
  - \* 汉字显示
  - \* 汉字打印
  - \* 汉字字模的传送

具体说来,也就是压缩汉字库 CCBIOS 系统必须包含有:

- \* INT 10H 扩充的汉字显示模块
- \* INT 16H 扩充的汉字输入模块
- \* INT 17H 扩充的汉字打印模块
- \* INT 1DH 适合于汉字显示的视频号数

以上4个软中断是为了使系统能进行汉字处理而代替 BIOS 改写的。

另外,要完成汉字的显示和打印,系统还需有一个能对所要处理的汉字从字库中取出其相应的字模点阵的模块和一个相应的字库。不难理解,上述各模块一定是常驻内存的,加载后内存中存在的就是已经自行脱密过的可执行的代码了。只要从内存中找到以上几个模块的位置,将其复制出来,问题也就迎刃而解了。

## 二、解密的步骤

以下还是以压缩汉字库 CCBIOS 为例具体说明。拿一张压缩汉字库 CCBIOS 系统盘,用 DOS 的 DIR 命令查看,不难发现,它包含以下几个文件。

- \* STEP1. EXE
  - \* STEP2. EXE
  - \* STEP3. EXE
  - \* CCCC. EXE
  - \* 打印驱动程序
- 其中经过激光加密的文件只有 STEP2. EXE (管理和传送字模库模块)  
CCCC. EXE (CCBIOS 主体)

### 1. 解密 CCCC. EXE

加载压缩汉字库 CCBIOS,进入调试程序 DEBUG 利用 DOS 系统的第 35H 号系统功能调用找出 INT 10H,INT 16H,INT 1DH 的入口,可以发现这三个模块在同一段内,这便是 CCCC. EXE 所有组成,只要将其复制出来即可获得解密的 CCCC. EXE。

一般说来要从内存中复制出一个文件,有两个信息是必须知道的。

- a. 该文件在内存中的起始地址
- b. 该文件的长度

第一个信息上面已经得到,余下要找的是第二个信息。由经验可知,要使程序能常驻内存则必须利用 DOS 的 INT 27H 才能实现,故在 CCCC. EXE 中一定有以下两条指令。

```
MOV DX,Length  
INT 27H
```

其中 Length 为程序需要保留的长度

通过 DEBUG 中的 S 命令设法找到正确的 INT 27H 中断指令,即可得到 CCCC. EXE 在内存中的位置。到这时就可以把 CCCC. EXE 从内存中复制出来了。复制的方法为:

```
C>DEBUG  
—RCX  
CX: * * * *  
: * * * * (要保存的长度)  
—RBX
```

BX: \* \* \* \*

:0000此例因程序没有跨段,所以为0000H

—NCCCC.COM

—W \* \* \* \* 此处的 \* \* \* \* 为起始地址

不过,这时复制出来的 CCCC.COM 还不能使用,还必须做一些改动才行。主要有以下两点:

a. 原 CCCC.EXE 中的重定向语句

```
MOV DS, * * * *
```

这里可以改为:

```
PUSH CS
```

```
POP DS
```

b. 因 CCCC.COM 文件将是 100H 处加载而原 CCCC.EXE 是从 0H 处开始加载的,故两个文件中的 DS 值相差 10H,所以在 CCCC.COM 中的设置 INT 10H, 10H, 17H, 1DH 入口指令中的段值必须作加 10H 的修改。具体修改的步骤,这里就不赘述了。

完成了以上工作后,就算是得到了可以自由使用和拷贝的 CCCC.COM 了,从而也就完成了对 CCCC.EXE 的解密工作。

### 2. 解密 STEP2.EXE

STEP2.EXE 实际上就是一个管理和传送汉字字模库的模块。要对 STEP2.EXE 进行解密,首先必须知道压缩汉字库 CCBIOS 系统是如何进行汉字字模传送的,这一点可以在 INT 10H 的第 18 号功能块中找到,请看 INT 10H 中第 18 号功能块的一段程序。

```
AND DX, 7F7F
```

```
PUSH AX
```

```
MOV AX, BX
```

```
INT 6EH
```

```
MOV DX, AX
```

```
POP AX
```

RET

由此可知,取字模的工作是通过 INT 6EH 来完成的,这与 CCCC.EXE 同理也就算找到了其解密的突破口了。

进入 DEBUG,利用 DOS 的 35H 号系统功能调用找到 INT 6EH 的入口,然后仿照解密 CCCC.EXE 的方法将 STEP2.EXE 复制出来并取名为 STEP2.COM 即完成了其解密工作。

### 三、关于这种解密方法的两点说明

#### 1. 适用范围

了解了本解密方法就不难知道,只要经过加密处理的软件在它加载后是常驻内存的,一般都可很好地用这种方法解密,只是寻找其在内存中的驻留位置和长度需要解密者经过分析后获得。这里介绍的只是众多寻找方法中的一种,不同的软件因其功能不同,用这种方法并不一定能找得到,这时就需要解密者用别的方法去寻找,这里笔者就不一一介绍了。对于那些不是常驻内存的软件,只要它没有使用动态复盖技术的话,这种方法也同样适用,因为任何软件在它执行完后且没有在同一位置加载过别的软件时,它在内存中总是会有其影子存在的,可设法找到这个影子,再用上面介绍的方法将其完整地复制出来。

#### 2. 本解密方法的特点

本解密方法具有以下特点:

----不用了解软件是用何种方式进行加密的。

----不用象其它解密方法那样需费许多时间和精力去寻找加密的密钥,并跟踪其加密过程并逐渐脱密。

----其原理和实现方法简单且易掌握。

----安全、可靠,不会破坏原软件。

## 三维图形显示的遮掩技术及实现

武汉水利电力学院(430072) 季军杰

二元函数  $Z=f(x,y)$  所呈现的图形一般是曲面形状,有的在图形上表现出突出的几何特征。这些曲面在生活和生产实践中,在数学、物理和工程技术中都是常见的,熟悉它们的图形很重要。

为了获得较为真实的  $Z=f(x,y)$  图象,我们须避免描绘那些被隐藏了的曲面段,这就要用到计算机图形学中的三维图形显示遮掩技术。

#### • 遮掩技术:

遮掩技术的实现基于以下两个前提:(1)前景线条

(Z轴正向)在背景线条之前描绘,(2)若一线条或其一部分落入先前已描绘线条所在的区域内,它就被遮掩(或称为消隐)。消隐可通过生成一个可视最大函数和一个可视最小函数获得,这样,凡落入以上界限的线条部分即被遮掩。下面的隐藏线消隐程序可实现此技术。

#### • 编程要点:

本程序以描绘函数  $Z=7e^{-0.1(x^2+y^2)}$  为例,产生一线状结构曲线映象以表示该曲面(源程序附后)。

程序开头的初始化工作是建立视区和屏幕参数以

及待描绘的函数。定义两个数组 YMAX(I)和 YMIN(I)用于对每个水平屏幕坐标进行确认:(1)所遇到的最大垂直屏幕坐标存入 YMAX(I),(2)所遇到的最小垂直屏幕坐标存入 YMIN(I)。

最初,每个 YMAX(I)的值为0,每个 YMIN(I)的值为199(屏幕最大扫描线),一旦屏幕坐标点 $(x_s, y_s)$ 已描绘过,YMAX $(x_s)$ 值就重置,一旦一个 $(x_s, y_s)$ 被考虑描绘时, $y_s$ 就与 YMAX $(x_s)$ 和 YMIN $(x_s)$ 比较,若 $y_s$ 在 YMAX $(x_s)$ 和 YMIN $(x_s)$ 之间,点 $(x_s, y_s)$ 就被消隐,否则就描出。

运行本程序,隐藏线消除后的三维图象更加美观。

该程序用 BASIC A 写成,下面对程序中所用参数进行说明。

D:视点 P 到物体所在坐标系原点 O 之距离。

(THETA, PHI):视者观察物体的方位。其中 PHI 为 OP 与 Z 轴正向之夹角,THETA 为 X 轴正向与 OP 在 XY 平面上的投影的夹角。

VD:视点和显示屏(投影平面)间的距离。

S:屏幕中心大小值,图象在 $2S \times 2S$ 正方形区域内显示。

SCF:屏幕比例调整因子,以反映图象真实纵横比。

TX, TY:把视口坐标变换为屏幕坐标,并调整图象显示位置。

以上参数均可按照实际应用情况进行调节。

参数间关系如下:

①改变 THETA 和 PHI 的值可从不同角度观察物体。

②改变 D 值可从较远处或较近处观察。

③当视点(D, THETA, PHI)固定时,VD 的变化将影响显示屏上物体映象的大小。若显示屏与观察者较近(VD 减小),则投影映象将减小。

④D 和 VD 同时控制着屏幕映象大小,增大 D 值可减小透视效果,但映象变小,放大映象可增大 VD,而不再引入透视效果。

⑤VD/S 比率的大小类似于照相调焦时从取景器中看到的图象效果。若比率小,类似于用短焦照相,产生广角镜头映象画面;若比率大,类似于长焦效果,生成如“望远摄影”的一幅有限图象。

⑥当  $D=VD$  且  $D \rightarrow \infty$  时,将产生正投影映象。

```
110 CLS;KEY OFF
120 SCREEN 2
130 DIM YMAX(640),YMIN(640)
140 GOSUB 190
150 GOSUB 250
160 GOSUB 280
170 END
190 TX=120;TY=100;PI=3.141593;SCF=2.4
200 D=30;VD=420;THETA=30;PHI=80;S=2
210 THETA=THETA*PI/180;PHI=PHI*PI/180
220 SN1=SIN(THETA);SN2=SIN(PHI);CN1=
COS(THETA);CN2=COS(PHI)
```

```
230 RETURN
250 FOR I=1 TO 640;YMAX(I)=0;YMIN(I)=199;NEXT
I
260 RETURN
280 DEF FNZ(X,Y)=7*EXP(-.1*(X*X+Y*Y))
290 FOR X=8 TO -8 STEP -1
300 F=0
310 FOR Y=-8 TO 8 STEP 5
320 Z=FNZ(X,Y)
330 GOSUB 400
340 GOSUB 470
350 GOSUB 530
360 NEXT Y
370 NEXT X
380 RETURN
400 XE=-X*SN1+Y*CN1
410 YE=-C*XN1*CN2-Y*SN1*CN2+Z*SN2
420 ZE=-X*SN2*CN1-Y*SN2*SN1-Z*CN2+D
430 RETURN
470 XS=(VD/S)*(XE/ZE)
480 YS=(VD/S)*(YE/ZE)
490 XS=XS+TX;XS=SCF*XS
500 YS=-YS+TY
510 RETURN
530 IF F=0 THEN 540 ELSE 570
540 F=1;F2=0
550 OX=XS;OY=YS
560 RETURN
570 DX=OX-XS;IF DX=0 THEN DX=1
580 DY=OY-YS
590 SLOPE=DY/DX
600 T1Y=OY
610 NX=INT(OX)+1
620 FOR T1X=NX TO XS
630 F1=1
640 T1Y=T1Y+SLOPE
650 IF T1X<0 OR T1X>639 THEN F1=0;F2=0;GOTO 700
660 IF T1Y<0 OR T1Y>199 THEN F1=0;F2=0
670 IF T1Y<=YMIN(T1X) THEN 730
680 IF T1Y>=YMAX(T1X) THEN 780
690 F2=0
700 NEXT T1X
710 OX=XS;OY=YS
720 RETURN
730 YMIN(T1X)=T1Y
740 IF F1=0 THEN 770
750 IF F2=0 THEN PSET(T1X,T1Y);F2=1
760 LINE-(T1X,T1Y)
770 IF T1Y<YMAX(T1X) THEN 700
780 YMAX(T1X)=T1Y
790 IF F1=0 THEN 700
800 IF F2=0 THEN PSET(T1X,T1Y);F2=1
810 LINE-(T1X,T1Y)
820 GOTO 700
```

# BASIC 超长随机文件的处理方法

上海前进实业公司中心电脑室(202179) 朱建平

在 BASIC 程序设计中,磁盘数据文件的输入和输出分为二类:顺序文件和随机文件。由于随机文件具有占用磁盘空间少,数据可以随机地存取等特点而被广泛采用。一般而言,随机文件的使用分为下面几步:

1. 以随机存取方式打开文件,并指定记录长度,缺省值为128个字节。
2. 对于要存取的随机文件的变量用 FIELD 语句在随机缓冲区中为其分配空间。
3. 用 LSET 和 RSET 将数据送入随机缓冲区。
4. 用 PUT、GET 语句将数据从缓冲区写入磁盘或从磁盘送入缓冲区。

由此可见,随机文件的建立与使用需要设定记录长度,并且要对记录缓冲区进行定义。现在的问题是,当一个随机文件的记录长度超过128个字节,且记录中的数据项较多,用一个 FIELD 语句定义不完时,如何用变通的方法实现随机文件的正确存取。笔者在软件开发中,就曾碰到过这个问题,一个随机文件的记录长度为230个字节。每10个字节分配一个区段,需要用二个 FIELD 语句对记录中的每个数据项进行定义。对这样一个超长记录的随机文件如何来进行正确的存取呢?经查阅有关资料,笔者找到了解决这个问题的简单方法。

## 一、随机文件超长记录的长度设定

我们知道,IBM PC BASIC 命令的完整格式为:

```
BASIC[A][filespec][ /F:files ][ /S:bsize ][ C:combuffer ][ /M:max workspace ]
```

其中:可选项/S: bsize 参数是设置随机文件的记录长度,OPEN 语句中表示记录长度的参数不得超过该值,其缺省值为128个字节,可指定的记录长度的最大值为32767。

由此可见,对于一个随机文件的长度超过128个字节时,可在 BASIC 启动时,设置可选项参数/S: bsize 来完成超长记录的定义。笔者碰到的问题就是采用下面的命令来解决的。

```
A>BASICA 文件名[.BAS] /S:230
```

## 二、随机文件的续行方法

BASIC 语言规定,一个语句行的最大长度为256个

字节。对于一个超长记录的随机文件,可能会出现用一个 FIELD 语句定义不完的现象,如何对每个数据项进行正确定义,即 FIELD 语句的续行定义呢?笔者发现:当一个 FIELD 语句定义不完时,可在后一个 FIELD 语句的开头设置一个虚拟变量,其长度为前一个 FIELD 语句定义项的总长度,继后再定义余下的数据项。采用这种方法,即可完成对 FIELD 语句的续行定义。假如在后一个 FIELD 语句中不设置虚拟变量,则在随机文件的存取中不保证数据的正确性。以笔者的例子为例,其程序段为:

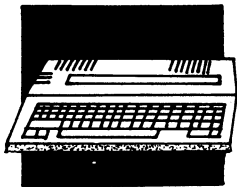
```
100 OPEN "JBADATA.DTA" AS #1 LEN = 230
110 FIELD #1, 10 AS CH$(1), 10 AS CH$(2), ..., 10 AS CH$(13)
120 FIELD #1, 130 AS CH$(1), 10 AS CH$(14), ..., 10 AS CH$(23)
```

程序中,120行语句中的130AS CH\$(1)是一个虚拟变量,其长度为上一个 FIELD 语句定义的数据项总长度。虚拟变量,在这里仅起到一个虚拟前一条 FIELD 语句定义的数据长度的作用,亦即仅起到几个 FIELD 语句之间的衔接作用。在这个例子中,笔者采用的虚拟变量名 CH\$(1)与上一个 FIELD 语句的缓冲数据项名同名,这并不影响随机文件的正确存取。当然,虚拟变量名亦可用其他变量名,如:A\$,B\$等,但是,虚拟变量的长度则必须等于上一条 FIELD 语句所定义的数据项的总长度。上面的例子中,假如后一个 FIELD 语句中不设置虚拟变量的话,则记录在随机文件中的数据为后一个 FIELD 语句所定义的数据项。

为了程序设计方便,也可采用循环语句来简化上述程序,以达到同样的目的。简化后的程序为:

```
100 OPEN "JBADATA.DTA" AS #1 LEN = 230
110 FOR I=1 TO 23
120 FIELD #1, 10 * (I-1) AS CH$(0), 10 AS CH$(I)
130 NEXT I
```

上述方法在 IBM PC/XT 机上通过。



# 打印机的定位和格式控制

西安石油仪器研究所(710054) 栾傲发

## 学习机之友

在数据输出时,我们往往要求在打印机上的格式整齐、美观。APPLE-I 和中华学习机的 BASIC 语言中没有提供很方便的格式输出。常规在40列范围内用 TAB 函数作为定位控制。超过40列则用 SPC 函数作为定位控制,但事先必须估计输出数据的域宽。尽管如此仍给编程时的数据输出带来了许多不便。下面介绍一种既方便、又灵活的打印机行宽和格式控制编程方法。

在 APPLE 和中华学习机中有一个可供用户调用的汇编子程序。调用地址为49568(如果打印机驱动卡插在1号槽)。它能够像 FORTRAN 语言中的 I、F、E 输出格式那样来使用,但缺点是不能作为数据输入。在调用这个子程序之前先定义打印机的宽度。若输出宽度在1—79列内则用 POKE1657,80语句定义,打印机输出的字符为79列标准字符。若输出宽度在1—135列内则用 POKE1657,135语句定义,打印机输出的字符为135列缩小字符,再按如下介绍的两种调用形式来编程就可实现打印机的定位和格式输出。

### 一、一般形式

CALL SUB;M1,V1;F1,M2 V2;F2,……Mn,Vn;  
Fn,CHR\$(表达式);

其中:

SUB——变量名或直接数49568

M1~Mn——字符串,字符串变量,字符串数组元素。

V1~Vn——整型变量、浮点变量

F1~Fn——格式控制字,有如下几种形式

1. In 整型输出格式,n 为域宽。
2. Fw.d 浮点输出格式,w 为域宽,d 为小数位数。
3. En 指数输出格式,n 为小数位数。

CHR\$(表达式)——打印机控制函数,其中:

10 换行

11 垂直定位

12 跳页

13 回车

\* 其它控制函数请参阅有关打印机说明书。

### 二、特殊形式

CALL SUB;CHR\$(表达式1);[CHR\$(表达式2)]  
……[CHR\$(表达式n)];

其中

SUB——同上

CHR\$(表达式1)……CHR\$(表达式n)——参阅

表一

这种调用形式定义打印机的各种功能。它往往与

一般形式共同使用以达到专门的输出格式。

### 一般形式调用实例

```

10 A%=254;B=3.141592;C=69.435
20 POKE 1657,80
30 WR=49568
40 PR#1
50 CALL WR;CHR$(13);
60 CALL WR;"A=",A%;I4,CHR$(13);
70 CALL WR;"B=",B;F8.6,CHR$(13);
80 CALL WR;"C=",C;E4,CHR$(13);
90 END

```

]RUN

```

A=    254
B=    3.141592
C=    6.943E+01

```

### 特殊形式调用实例

调用内部水平制表符

```

10 POKE 1657,80
20 WR=49568
30 PR#1
40 PRINT"0123456789012345678901234567890"
50 FOR I=0 TO 2
60 CALL WR;CHR$(9);"TAB";
70 NEXT
80 END
]RUN
]0123456789012345678901234567890

```

TAB TAB B TAB

调用垂直制表符

```

10 POKE 1657,80
20 WR=49568
30 PR#1
40 FOR I=1 TO 4;READ X%(I);NEXT
50 CALL WR;CHR$(27);"B";CHR$(1);CHR$(3);CHR
$(6);CHR$(10);CHR$(0);
60 CALL WR;CHR$(11);"TAB";X%(1);I6,"stline";
70 CALL WR;CHR$(11);"TAB";X%(2);I6,"rdline";
80 CALL WR;CHR$(11);"TAB";X%(3);I6,"thline";
90 CALL WR;CHR$(11);"TAB";X%(4);I6,"thline";
100 END
110 DATA 1,3,6,10
]RUN
TAB      1stline
TAB      3rdline
TAB      6thline
TAB      10thline

```



建立右边列的起始位置(Right Margin)

```
10 POKE 1657,80
20 WR=49568
30 PR#1
40 A$="0123456789012345678"
50 CALL WR;CHR$(13);
60 CALL WR;CHR$(27);"Q";CHR$(12),A$,CHR
$(13);
70 CALL WR;CHR$(27);"@";
80 END
]RUN
012345678901
2345678
```

建立左边列的起始位置(Left Margin)

```
10 POKE 1657,80
20 WR=49568
30 PR#1
40 A$="0123456789012345678"
50 CALL WR;CHR$(13);
60 CALL WR:A$,CHR$(13),CHR$(27);"1";CHR$(8);
"left margin";
80 END
]RUN
0123456789012345678
```

left margin

建立7/72英寸的打印间隔

```
10 POKE 1657,80
20 WR=49568
30 PR#1
40 A$="7/72 inch line spacing"
50 CALL WR;CHR$(13);CHR$(27);"1";
60 FOR I=0 TO 6
70 CALL WR:A$,CHR$(13);
80 NEXT
90 CALL WR;CHR$(27);"@";
100 END
```

建立字符的下划线

```
10 POKE 1657,80
20 WR=49568
30 PR#1
40 A=3.1415926
50 CALL WR;CHR$(13);CHR$(27);"—";CHR$(1);"A
="";
60 CALL WR;CHR$(27);"—";CHR$(0),A;F9.7;
70 END
]RUN
A=3.1415926
```

一般形式与特殊形式使用实例

```
4=A$="0123456789012345678901234567890123456
789012345678901234567890123456789012345
678901234567890123456789012345678901234
567890123456789012345"
```

表一

| 功 能      | 规 则                                    |
|----------|----------------------------------------|
| 初始化      | CHR\$(27);"@";@可用CHR\$(64)替代           |
| 1/8英寸间隔  | CHR\$(27);"0";0可用CHR\$(0)替代            |
| 7/72英寸间隔 | CHR\$(27);"1";1可用CHR\$(1)替代            |
| 1/6英寸间隔  | CHR\$(27);"2";2可用CHR\$(2)替代            |
| 清除缓冲区    | CHR\$(18);                             |
| 加重打印     | CHR\$(27);"E";E可用CHR\$(69)替代           |
| 取消加重打印   | CHR\$(27);"F";F可用CHR\$(70)替代           |
| 压缩打印     | CHR\$(15);                             |
| 放大打印     | CHR\$(14);                             |
| 下划线      | CHR\$(27);"—";CHR\$(1);一可用CHR\$(45)替代  |
| 取消下划线    | CHR\$(27);"—";CHR\$(0);一可用CHR\$(45)替代  |
| 建立上标打印   | CHR\$(27);"S";CHR\$(0);S可用CHR\$(83)替代  |
| 建立下标打印   | CHR\$(27);"S";CHR\$(1);                |
| 取消上、下标打印 | CHR\$(27);"T";T可用CHR\$(84)替代           |
| 建立左列起始位置 | CHR\$(27);"1";CHR\$(n);1可用CHR\$(108)替代 |
| 建立右列起始位置 | CHR\$(27);"Q";CHR\$(n);Q可用CHR\$(81)替代  |
| 回退一个字符   | CHR\$(8);                              |
| 响铃       | CHR\$(7);                              |
|          | 上表只列出一般常用的几种功能,其它功能请参阅有关打印机说明书         |

注:本文中所列举的程序例子均将打印机接口卡插在1号槽口中。若打印卡插在其它槽口中,程序应按如下修改。

```
10 POKE 1656+N,80
20 WR=49312+256*N
30 PR#N
```

其中N为打印机接口卡所在的槽口。例如打印机接口卡插在3号槽口中,N为3。

```
5 B$="--"
10 POKE 1657,136
```

```

20 WR = 49568;PR#1
30 CALL WR;CHR$(13),CHR$(15);A$,CHR$(13):
40 GOSUB 200;GOSUB 300: GOSUB 200
50 FOR I=1.0 TO 1.5 STEP 0.1
60 CALL WR;"!",I;F2.1,"!":
70 FOR J=0 TO 0.1 STEP 0.01
80 S=SQR(I+J)
90 CALL WR;S;F7.6,"!":
100 NEXT :CALL WR;CHR$(13):
110 NEXT :GOSUB 200;PRINT CHR$(14)
120 END
200 FOR I=0 TO 126;CALL WR,B$:
210 NEXT :CALL WR;CHR$(13):
220 RETURN
300 SP$="1"
310 FOR I=0 TO 9
320 CALL WR;SP$,I;I:
330 NEXT :CALL WR;CHR$(13):
340 RETURN
350 RETURN

```

|    |          |          |          |          |          |          |          |          |          |          |
|----|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| 0: | 1.000000 | 1.004987 | 1.009950 | 1.014899 | 1.019803 | 1.024695 | 1.029563 | 1.034408 | 1.039230 | 1.044030 |
| 1: | 1.048808 | 1.053565 | 1.058300 | 1.063014 | 1.067707 | 1.072380 | 1.077032 | 1.081663 | 1.086278 | 1.090871 |
| 2: | 1.095445 | 1.100000 | 1.104526 | 1.109023 | 1.113502 | 1.117963 | 1.122407 | 1.126834 | 1.131370 | 1.135781 |
| 3: | 1.140175 | 1.144552 | 1.148872 | 1.153256 | 1.157583 | 1.161895 | 1.166190 | 1.170469 | 1.174734 | 1.178982 |
| 4: | 1.183235 | 1.187434 | 1.191637 | 1.195826 | 1.200000 | 1.204159 | 1.208304 | 1.212435 | 1.216552 | 1.220655 |
| 5: |          |          |          |          |          |          |          |          |          |          |
| 6: |          |          |          |          |          |          |          |          |          |          |
| 7: |          |          |          |          |          |          |          |          |          |          |
| 8: |          |          |          |          |          |          |          |          |          |          |
| 9: |          |          |          |          |          |          |          |          |          |          |

# 放大打印 APPLE II 高分辨率图形又一法

烟台师院数学系(264000) 刘善平

方法:利用不同的字符代替图形上相应颜色的点,输出到打印机.可进行矩形裁剪,正相、反相、放大及加重打印输出。

步骤:1. 运行程序 A,对高分辨率图形进行预处理.对第一页操作,预处理后的图形存放在第二页;对第二页操作,预处理后的图形存放在第一页.这样可以保留原图形.在运行程序 A 之前,需要设置4个参数(表1).程序 A 的首地址是 \$ 6100。

2. 运行程序 B,按照要求对预处理后的图形打印输出.程序 B 将裁剪后图形的相应存储单元的内容按顺序调入 \$ 00FC 单元.用指令 LSR 顺序移位,每移一次判断一次标志位 C 的值及 \$ 00FB 单元的内容,进行不同的打印输出.程序 B 的首址是 \$ 6200,在运行前需要设置6个参数(表2)。

举例:把程序 A 和 B 装入内存以后,运行程序 C,取 P=1、V=1 可将高分辨率图形第一页反相打印输出。

至于横向、纵向的放大、压缩以及加重等打印功能,只需编一块 BASIC 子程序,用 PRINT CHR\$(n) 语句将打印机预置好.有关的几个 n 值见表3。

表1

| 十进制地址 | 十六进制地址  | 参数内容及意义        |
|-------|---------|----------------|
| 6     | \$ 0006 | 被操作页首址的低位      |
| 7     | \$ 0007 | 被操作页首址的高位      |
| 252   | \$ 00FC | 预处理后图形存放页首址的低位 |
| 253   | \$ 00FD | 预处理后图形存放页首址的高位 |

表2

| 十进制地址 | 十六进制地址  | 参数内容及意义    |
|-------|---------|------------|
| 6     | \$ 0006 | 矩形上界的首址的低位 |
| 7     | \$ 0007 | 矩形上界的首址的高位 |
| 8     | \$ 0008 | 矩形左界/7     |
| 9     | \$ 0009 | 矩形右界/7     |
| 250   | \$ 00FA | 下界—上界+1    |
| 251   | \$ 00FB | 0-正相;1-反相  |

(下转第23页)

# 趣味速算练习器

苏州景范中学(215005) 赵 旭

对孩子进行速算练习,除提高运算能力外,还能开发智力。只要输入以下程序,就可将中华学习机变成一台趣味速算练习器。它能帮助家长出色地完成辅导孩子速算练习的工作。练习器除自动出题和评分外,还能根据孩子的实际程度自动调节练习题的难度。孩子的速算能力随时都可通过练习器得到了解。

练习器具体使用方法是:运行程序后,屏幕上出现五个级别供选择。选定后,练习器自动出题,题目下面有四个答案供选择。同时,屏幕上出现一条不断伸长的光带,并限定在光带伸到某一长度的时间内答出此题。如果回答正确,屏幕上增添五颗星并成绩加五分。连续答对二十题得一百分并升一级,练习题难度也跟着提高。如果成绩不佳,练习器能自动将难度降低,直到孩子能适应为止。

```
1 A$(1)="特";A$(2)="高";A$(3)="中"
2 A$(4)="初";A$(5)="低";A$(6)="级"
3 PRINT CHR$(4);"PR #3";PRINT ; HGR2 ; POKE
  49168,0
4 L=1;FOR K=1 TO 9 STEP 2;VTAB K;HTAB 7;POKE 50,
  63
5 PRINT L;POKE 50,255;PRINT " ";A$(L);A$(6);L=L
  +1;NEXT
6 POKE 50,127;VTAB 5;HTAB 15
7 PRINT"请您选择(1-5)";POKE 50,225
8 A=PEEK(49152);IF A<127 THEN 8
9 P=A-176;N=10;FOR K=1 TO P;N=N*2;NEXT;
  HGR2
10 TEXT;HOME;T=0;VTAB 8;HTAB 30;PRINT"TIME";
  FOR I=1 TO 19 STEP 2;FOR J=1 TO 11 STEP 10
20 A=INT(99*RND(1))+1;B=INT(99*RND(1))+1;IF
  A+B>99 OR A+B<10 THEN 20
30 C=A+B;VTAB 21;HTAB 16;PRINT " ";VTAB 21;
  HTAB
  16;PRINT A;"+";B;"="
40 FOR K=1 TO 4;X=INT(90*RND(1))+10;IF X=
  C THEN 40
45 B(K)=X;NEXT
50 D=INT(4*RND(1))+1;B(D)=C
60 FOR K=1 TO 4;VTAB 23;HTAB 2+K*6;POKE 50,63;
  PRINT K;POKE 50,255;PRINT " ";B(K);NEXT K
70 POKE 49168,0;FOR L=40 TO 1 STEP -1;POKE 50,63;
  HTAB L;VTAB 10;PRINT"<";POKE 50,255;FOR M=1
  TO N;NEXT
75 B=PEEK(49152);A=B-176
80 IF A=D THEN FOR Q=J TO J+9 STEP 2;VTAB Q;HTAB
```

```
I;PRINT"*";NEXT;T=T+5;VTAB 13;HTAB 30;
PRINT"DE.";T;FOR M=I+1 TO 40;HTAB M;VTAB 10;
PRINT";NEXT M,J,I
85 IF A<>D AND B>127 THEN 100
90 IF B<127 THEN NEXT L
100 PRINT CHR$(4);"PR #3";PRINT;HGR2
105 VTAB 4;HTAB 8;PRINT"您的成绩是:";POKE 50,63;
  PRINT A$(P);A$(6);POKE 50,255;PRINT " ";T;
  "分";VTAB 7;HTAB 6
110 IF T<60 THEN PRINT"别灰心,好好练习,一定成功!";
  N=N*2;P=P+1;IF P>5 THEN P=5
120 IF T>=60 AND T<80 THEN PRINT"初步成功,继续努
  力!"
130 IF T>=80 AND T<100 THEN PRINT"成绩优秀,争取
  满分!"
140 IF T=100 THEN PRINT"祝贺您获得成功!";N=N/2;P
  =P-1;IF P=0 THEN P=1
150 GET A$;GOTO 10
```

## 计算机解决数学问题

天津财经学院 216 信箱(300222) 赵方明

计算机无疑是计算能手,它的计算速度和计算范围都是令人惊叹的。但对于计算一些具体的数学问题,编写巧妙的程序是十分重要的,下面是我在平时为解决数学问题而编写的几则程序。

### 一、分解质因数程序

```
5 M=0;DIMB(100)
10 INPUT N
20 X=N
30 FOR I=2 TO SQR(X)
40 IF N/I <=1 THEN 80
50 IF N/I <> INT(N/I) THEN 70
60 M=M+1;B(M)=1;N=N/I;GOTO 40
70 NEXT I
80 PRINT X;"=";
90 IF M=0 THEN 130
100 FOR J=1 TO M
110 PRINT B(J);" * ";
120 NEXT J
130 PRINT N
```

140 END

本程序利用数组存放各个因数,由 80-130 语句打出。而 30-70 语句则是巧妙利用判断语句求出各因数,此求解过程完全模仿人为计算方法,而这对计算机来讲便迅速多了,于是计算机的特长发挥出来了。

### 二、求两数最大公约数与最小公倍数程序

```

10 INPUT M,N
20 A=(M+N-ABS(M-N))/2
30 FOR I= A TO 2 STEP -1
40 IF INT(M/I) <> M/I THEN 60
50 IF INT(N/I)=N/I THEN 80
60 NEXT I
70 PRINT I,M*N;GOTO 10
80 PRINT I,M*N/I,GOTO 10

```

本程序通过计算机搜索最小公倍数 I,也就得到了最大公约数  $M * N / I$ 。20 语句实际是求得 M 与 N 中较小的一个。当然用短除法也可以编程解决此题,与程序一相仿,不如现程序简明。

### 三、求圆周率近似值程序

在一次计算机竞赛中出现过这样一道题:请编程计算出  $\pi$  的近似值,而且须用 355/113 求得,并保留 20 位有效数字。有的同学想方设法扩大计算机运算范围,以达到保留位数,但一无所成。其实灵活一点,很容易获得答案:

```

5 A=355;B=113;N=0
10 PRINT"355/113=";
20 N=N+1;PRINT INT(A/B);
30 IF N=1 THEN PRINT". ";
40 A=(A-INT(A/B)*B)*10
50 IF N<20 THEN 20
60 END

```

程序中 40 语句相当于笔算除法中的落零,50 语句判断是否达 20 位。这样程序一运行,计算机就象人一样算起除法,很快地完成任务。

以上三个程序看来并不高深,但却结构清晰。它们的共同特点都是考虑到我们自己的处理方法,结合具体过程,用灵巧的语句编制而成,这样就使计算机出色地完成了工作。对于数学问题和其他任何问题,我们都应通过合理简洁的程序来解决它。

## 巧改系统

北京三里河群智巷(100051) 张浩

是否能用自己或朋友的名字作为系统命令呢?或把关键的命令更改,使别人不能使用?在 LASER310 机上 BASIC 命令是固化的,但通过对编译和执行阶段的拦截可以改造系统。我用 Z-80 机器语言编了一个小程序,用它可以把系统命令改成任何字符串。本程序以

改命令 CLS 为例,要改其它命令则要做相应改动。

使用方法:键入并 RUN 本程序后,出现问号,要求输入改后的命令字符串,然后回车即可。这样命令就被改了。如果,再键入 CLS 命令则会出现语法错误。本程序 RUN 后可 NEW 掉,不占 BASIC 区。程序附后。

```

10 INPUT B$;T=LEN(B$)
20 FOR I=1 TO T;B(I)=ASC(MID$(B$,I,1));NEXT
30 FOR I=31058 TO 31112;READ A;POKE I,A;NEXT
40 FOR I=31113 TO 31112+T;POKE I,B(I-31112);NEXT
50 POKE 31088,T;POKE 30862,82;POKE 30863,121;A=USR(0)
60 DATA 33,89,121,34,4,120,201,217,33,91,29,209,183,237,82
70 DATA 213,217,194,120,29,205,120,29,254,132,202,151,25
80 DATA 229,6,5,17,137,121,26,190,32,14,205,120,29,19,16,246
90 DATA 209,209,205,201,1,195,30,29,225,126,201

```

## 数字游戏

河北机电学院自动化电气 91、1 班 闫浩

这是一个简单但很有趣的智力游戏。程序运行后,计算机先询问一下:“Player's Name:”,当你键入三个字如(代表你的姓名)后,游戏开始:屏幕上画出一个数盘,数盘里面的数字按一定的顺序排列,这时你应抓紧时间记住这个顺序和左上角显示的那个数字的位置。然后数盘被打乱,原先显示在左上角的那个数字变为闪烁字符,你能用控制键(I,M,J,K)来控制它的移动,分别使其和上,下,左,右的数字换位。如果你能在规定的步数(150 步)内将数盘复原,则算你胜,游戏结束(再玩,则按“S”键),若没能复原,则算失败,计算机再让你试一次。

注:20 句中和 200 句中引号内均为 2 个空格,80 句中引号内为 14 个空格。

```

5 DIM A(5,4);A=150
10 HOME;B=-1;C=10;B$="";E=INT(RND(1)*20)+10
20 VTAB 2,HTAB 29;PRINT"1. Number";VTAB 6,HTAB 28;PRINT"HL.";A;" ";A$;VTAB 9,HTAB 28;PRINT"Player:";
30 FOR I=1 TO 3:GET C$;B$=B$+C$;PRINT C$;NEXT I
40 FOR I=0 TO 5;VTAB I*4+2,HTAB 4;FOR J=1 TO 21;PRINT"-";NEXT J,I;FOR I=0 TO 4;VTAB 2;FOR J=1 TO 21;HTAB I*5+4;PRINT"1";NEXT J,I
50 FOR I=1 TO 5;FOR J=1 TO 4;A(I,J)=C;IF C/E=1 THEN X=I;Y=J

```

(下转第 11 页)

# ProDOS 磁盘操作系统入门(续)

北京铁路局中心卫生防疫站(邮编 100038) 廖 凯

## 第六章 外围设备

本章将介绍 ProDOS 如何用不同的外围设备及附件工作。涉及到的硬件有扩展 80 列卡,扩展 80 列 Text/AppleColor 转换卡,ROM Upgrade Kit,Apple 鼠标器,ProFile 5M 硬盘,Thunderclock 和 Proclock 时钟/计时卡。

### 一、APPLE 扩展 80 列卡

扩展 80 列卡将在 ProDOS 下作为一个 RAM 磁盘,这允许用户在 RAM 磁盘内快速地存取程序或文件。APPLE II c 具有 128K RAM,64K 主存储器,64K 辅助存储器,和 APPLE II e 一样,在 ProDOS 下这辅助存储器作为一个 RAM 磁盘。

APPLE II c 和 II e 具有双高分辨率图形功能。注意,APPLE II e REV A 不具备双高分辨率图形。

### 二、扩展 80 列 Text/AppleColor 转换卡

Color 卡是 APPLE 最初具有 RGB 输出方式的 80 列卡,有扩充的 64K 容量,可代替扩展 80 列卡。此卡用一个调制器连接 APPLE RGB 彩色显示器,或经由标准视频输出连接复合彩色显示器或单色显示器。可以同时用 RGB 和单色两种方式显示。对于单显和复合彩显所使用的视频 I/O 是标准的连接器,在主机后面可找到,RGB 彩显是用扩展/RGB 卡连接的。

此卡支持所有的标准 APPLE II 视频方式。另外,扩展/RGB 卡支持文字颜色的转换和新的双高分辨率图形。借助于此卡可有 6 种图形方式:

- 方式 1 560×192 黑色和白色
- 方式 2 140×192 全 16 种颜色
- 方式 3 混合方式 方式 1 与方式 4 的组合
- 方式 4 160×192 全 16 种颜色
- 方式 5 280×192 16 种颜色
- 方式 6 混合方式和低分辨率一样,80 列

此卡包含一套实用软件包。为了启动卡的图形方式,APPLE 提供了一个很有用的接口模块。B 型文件 Hires 程序将给用户某些惊人的功能。Hires 模块是由 19 个图形输出子程序组成,这些程序执行一串 & 指令,使用很简单。指令如下:

& GR # 设置指定的图形子程序。在执行此指令时,当前笔的位置被设为 0,0(屏幕的左上角)。笔的颜色系统设定为白色,背景为黑色,若选择 0,则关闭图形方式并清除存储器空间。

& TEXT 此指令转换图形方式为文本显示方式。

& COL = # 设置被 PLOT, DOT, DRAW 和 GPRNT 指令所使用的笔的颜色。此指令使用与低分辨率图形

相同的 16 种颜色。方式 1 只能用黑色和白色绘图。

& BCOL = # 设置背景颜色或经由 CLEAR 和 VFILL 指令来涂色,使用与低分辨率图形相同的 16 种颜色。

& CLEAR 此指令用 BCOL 指令指定的颜色来清屏,若未用 BCOL 指定颜色,则系统设定为黑色。

& VFILL(X1, X2, Y1, Y2) 此指令在屏幕上设置四个座标点并在此范围内用 BCOL 指令最后设置的颜色进行涂色。X1 和 X2 是从左到右的水平座标, Y1 和 Y2 是从上到下的垂直座标。

& MOVE(X, Y) 将笔移到 X, Y 位置而不在屏幕上绘图。X 表示从左到右的水平座标, Y 表示从顶到下的垂直座标。

& DOT 用当前前景颜色(COL)放置一个点作为当前笔的位置。此指令不需要参数。

& SCRNC) 送回当前笔位置的颜色(COL)值。

& SAVE/卷名/文件名 此指令是将高分辨率图形页 1 图形作为一个二进制文件存盘。

& LOAD/卷名/文件名 从磁盘上将上述保存的图形装到内存的高分辨率图形页 1。

& GPRNT 允许文字显示在图形屏幕上。在此指令有效时仍可以绘图,所有随后的输出经一个 PRINT 语句使文字显示到图形屏幕上。使用字形与 APPLE II 一样。字符被显示在当前笔的位置的右边,在字符之间有 7 个象素。除了回车字符(ASCII 13)和换行字符(ASCII 10),控制字符将被忽略。

& CPRNT # 此指令在图形打印子程序中选择控制码。若 # 等于 1,将选择控制码并打印;若 # 等于 0,则不选择控制码。例如,这功能可接受附加的自定义图形字符。

& TPRNT 取消 & GPRNT 指令的设置。

& NCHARS/卷名/文件名 此指令用于改变在图形显示下的标准字形为用户自定义字形。自定义字形必须与 & DRAW 指令设置的参数一致。

& SCHARS 此指令恢复标准的系统字形。

& DRAW(M, N, N1, R, W, H) 在屏幕上绘制一个预先定义的造型。此造型被放置在当前笔位置的右边。M 参数是造型首字节的存储器地址; N 参数指定造型每行的字节数或长度; N1 参数指定每行之间的位数; R 参数指定在转移造型之前所跳过的行数; W 参数指定造型的宽度; H 参数指定被转移的造型的高度。

注意,上述指令中,括号内的参数必须与指令一起使用。

你可以用 PRINT 语句在任何图形方式中显示文字。下面的短程序将使用这方法在方式 1 和双高分辨率图形中打印一行文字。

```

10 PRINT CHR $(4);“-HIRES”
20 & GR1
25 & CLEAR
30 & PLOT (0,191);& PLOT (559,191);& PLOT (559,0); & PLOT (0,0)
40 L$ = “You can output text to a GRAPHICS screen using PRINT”
50 Y=60;GOSUB 100
60 L$ = “ESC to back out”
65 Y=135;GOSUB 100
70 GET A$ ; IF A$ =CHR $(27) THEN TEXT;GO TO 99
80 GOTO 70
99 & TEXT;END
100 X = 280-(LEN(L$)*3.5);&MOVE (X,Y)
110 & GPRNT;PRINT L$ ;& TPRNT;RETURN

```

程序先装入 HIRES 程序并设置图形方式为 1。行号 30 用四个 & PLOT 指令从当前图形座标(0,0)沿屏幕四周画一边框。& PLOT 指令将从当前位置到绘图位置画一条线。行号 50 设置 Y 座标并转到行号 100 执行。行号 100 将文字放在屏幕中央并移动指针到新 X, Y 位置。& GPRNT 指令将使文字显示在图形屏幕上。随后的打印语句将输出指向图形屏幕,& TPRNT 指令恢复屏幕为图形方式。

HIRES 模块也可用于扩屏 80 列卡,有三种双高分辨率图形方式。它们是:

- 560×192 黑色和白色
- 140×192 16 种颜色
- 混合方式 上面两种方式的组合

HIRES 模块在主机内占用极少空间,模块较大部分保留在辅助存储器内,其余部分在主存储器内。

### 三、APPLE 鼠标器

鼠标器经 9 芯连接器连接在 APPLE 机上。鼠标器可以取代箭头键在屏幕上移动光标。鼠标器必须用 PR # 命令来启动;要从鼠标器上接收信息就必须使用 IN # 命令,随后跟一个 INPUT 语句以存储资料到变量内。鼠标器的当前位置和按钮的状态随时都可以知道。

鼠标器用一个 BASIC 程序来沟通。下面的程序将演示用户怎样用鼠标器绘制一个高分辨率图形。

```

10 D$ =CHR $(4);PRINT CHR $(21)
20 HGR2
30 COLOR=15;SCALE=1;ROT=0
40 HPLOT 0,0 TO 0,191 TO 279,191 TO 279,0,TO 0,0
60 PRINT D$ ; “PR # 4”;PRINT CHR $(1)
70 PRINT D$ ; “PR # 0”
80 PRINT D$ ; “IN # 4”
90 INPUT “”;X,Y,Z
100 INPUT “”X1,Y1,Z1
110 IF Z=1 AND Z1=1 THEN 130
115 IF Z=4 THEN 90

```

```

120 IF Z<0 THEN 190
130 X=INT(X/3.66)
140 Y=INT(Y/5.33)
150 X1=INT(X1/3.66)
160 Y1=INT(Y1/5.33)
170 HPLOT X,Y TO X1,Y1
180 GOTO 90
190 PRINT D$ ; “PR # 4”;PRINT CHR $(0)
200 TEXT ;PRINT D$ ; “PR # 3”
210 END

```

行号 60 接通鼠标器;行号 70 设置屏幕输出。用 IN # 命令读取鼠标器的状态,随后跟两个 INPUT 语句用于从 A 点到 B 点绘图。鼠标器 X 和 Y 的取值范围在 0 和 1023 之间。当鼠标器向右移时,X 值将增加;当鼠标器向自己移动时,Y 值将增加。X 和 X1 不能超过 279,Y 和 Y1 不能超过 191。如果按住按钮并很快地移动鼠标器,那么将会引起跳行和线不相连。

行号 110 和 120 测试按钮的状态。如果 Z 是负数,则表示已从键盘上按过键。如果连续地按着按钮,将没有行被绘制。按钮可能存在的状态如下:

| Z | 当前读取 | 最后读取 |
|---|------|------|
| 1 | 按下   | 按下   |
| 2 | 按下   | 释放   |
| 3 | 释放   | 按下   |
| 4 | 释放   | 释放   |

如果键盘上的键被按过,行号 190 将关闭鼠标器,并返回文本显示状态。

### 四、ROM Upgrade Kit

ROM Upgrade Kit 可替换计算机内的三块 ROM。MouseText 字符发生器 ROM 替换 Video ROM 342-0133,此 ROM 将产生 32 个新字符并替换反相大写字符集。用两块监控 ROM 替换 ROM C66 和 C67,这些 ROM 与 MouseText 一起工作并提供一些新的功能。

MouseText 是为使用 APPLE 鼠标器而开发的,它可产生 32 个 ICONS 图形。标准字符集不受 Upgrade Kit 影响。如果你打算用 BASIC POKE 语句在屏幕上显示反相的大写字符,那么将会发生问题,因为新的字符集已被放在 ASCII 码 64 到 95 的位置。反相大写字符集已被放在 ASCII 码 0 到 31 的位置。

如果你使用的是 I c,这些 ROM 已被安装在主机上。其它机型可自行购买安装。

两块监控 ROM 在 80 列显示方式时会有些改进:

1. APPLESOFT 指令和 ProDOS 命令均可用小写字母输入。所有键盘指令名都已做在键盘上,与 shift 合用,不包含 DATA 指令、REM 和串指令。在 ProDOS 下所有指令在任何状态下都可用小写字母输入。
2. HTAB、TAB、SPC 和 PRINT 语名中的逗号均可在 80 列范围内使用。
3. BASIC 在 80 列状态下输出字符的速度将提高 30%。
4. 粗糙的屏幕卷印已被取消。
5. 增加了两个 ESC 指令:

\* ESC CTRL-D 中止所有打印控制码,回车、换行、退格和警铃除外。

\* ESC CTRL-E 恢复打印控制码为系统设定码。

6. 在列印程序时,整个程序都向右移动一列,这样在编辑程序时不必再向左移动光标扫描行号了。

下面的程序将显示 ROM Upgrade Kit 的 32 个新字符。用此程序可检查计算机是否装有新 ROM。

```
5 HOME
10 PRINT CHR $(4);"PR #3"
20 PRINT
30 INVERSE
40 FOR I=64 TO 95:PRINT CHR $(27);CHR $(I)CHR
  $(24);
50 NEXT
60 NORMAL:END
```

### 五、ProFile 硬盘

APPLE II e 现在可以使用名为 ProFile 的 5M 或 10M 硬盘,此硬盘是苹果公司为 APPLE III 和 Lisa 计算机生产的。APPLE I 和 APPLE II 使用相同的磁盘格式,故两种计算机都可使用 ProFile 硬盘。

ProFile 是 5 英寸 Winchester 硬盘系统。ProFile 存取资料比 DISK I 快三倍。ProDOS 实用程序用 ProFile 操作并允许用户在不同的系统之间使用和拷贝文件。硬盘具有 5M 的容量,你可以用 ProFile 存取大量的资料,它相当于 35 个软磁盘。ProDOS 的分级文件结构使它很容易在列印的大量文件中找到一个文件。

有一个为硬盘而设计的名为 Catalyst 的程序,允许你从一个主菜单选择并运行硬盘里的程序。此程序可以在具有 64K 的 APPLE 上运行。这程序对 ProFile 的用户来说是必要的。

### 六、时钟/计时卡

#### 1. Thunderclock

Thunderclock 是一个与 ProDOS 系统兼容的时钟/

计时卡。ProDOS 可以自动识别此计时卡,并可在文件上记录文件的建立及修改日期和时间。在 MLI 内有一个支持 Thunderclock 的中断驱动程序。每当存取磁盘时,ProDOS 将当前的时间和日期存入内存地址 49040 至 49043 之间。此地址 BASIC 程序不会占用。

你可以用以下的语句显示日期和时间:

```
DAY=PEEK(49040)-INT(PEEK(49040)/32)*32
YEAR=INT(PEEK(49041)/2)
MONTH=(PEEK(49041)-YEAR*2)*8+INT(PEEK
  (49040)/33)
HOUR=PEEK(49042)
MINUTE=PEEK(49043)
```

这些指令只显示时间和日期。当前时间不能修改直到磁盘被存取为止。在 Users. Disk 的主菜单上选择 Display/Set Time 功能,你可以观察时间并过一会儿再观察时间,你会发现它没有变化。你在 CATALOG 磁盘后再观察时间,你会发现时间已改变。不足之处是你不能用 BASIC 写一个中断程序。

Thunderware 公司开发了一个名为 TUT(Time Utility)的实用程序以在 ProDOS 下存取此卡。此程序支持所有用于此卡的指令,它占用内存 4K 字节。注意, HIMEM 将下移 4K 字节以保持正常工作。

#### 2. Proclock

Proclock 是另一个具有存取 ProDOS 能力的兼容时钟/计时卡。象 Thunderclock 一样,此卡将在文件建立或修改时放置时间和日期,其用法与 Thunderclock 一样。 (全文完)

ProDOS 磁盘操作系统入门的介绍暂告一段落。关于 ProDOS 系统功能的调用,需要经由机器语言接口 MLI。这一部分资料我们将在准备就绪后陆续刊出,请读者留意,有关技术咨询请与廖凯同志接洽。 编者

(上接第 18 页)

表 3

| n 值     | 功能           |
|---------|--------------|
| 27;64   | 初始化设置        |
| 14      | 放大方式         |
| 15      | 压缩方式         |
| 27;65;n | 行间距为 n/72 英寸 |
| 27;69   | 加重方式         |

#### 程序 A:

```
6100- A5 06 85 08 85 FA A5 07
6108- 85 09 85 FB A9 08 8D 07
6110- 60 8D 08 60 A9 03 8D 06
6118- 60 A9 28 8D 00 60 A9 80
6120- 8D 02 60 A9 04 8D 05 60
6128- A9 00 8D 01 60 8D 03 60
```

```
6130- 8D 04 60 A2 00 A0 00 B1
6138- FA 91 FC C8 C0 28 90 F7
6140- D8 18 A9 28 65 FC 85 FC
6148- A9 00 65 FD 85 FD 18 AD
6150- 04 60 65 FA 85 FA AD 05
6158- 60 65 FB 85 FB CE 08 60
6160- D0 D3 A9 08 8D 08 60 18
6168- AD 02 60 65 08 85 08 85
6170- FA AD 03 60 65 09 85 09
6178- 85 FB CE 07 60 D0 B6 A9
6180- 08 8D 07 60 18 AD 00 60
6188- 65 06 85 06 85 08 85 FA
6190- AD 01 60 65 07 85 07 85
6198- 09 85 FB CE 06 60 D0 95
61A0-60
```

#### 程序 B:

```
6200- A4 08 A2 07 B1 06 85 FC
6208- 46 FC 90 09 A9 00 C5 FB
6210- 90 09 4C 20 62 A9 00 C5
```

(下转第 31 页)

## 第五章 源程序的编辑、汇编和运行

南京大学 大气科学系(210008) 朱国江

用汇编符号语言编写的汇编语言源程序,必须翻译成用机器码表示的机器语言程序(即目标程序),计算机才能识别和执行。完成这种翻译工作的过程称为汇编。这可以用不同的方法来实现,最常用的是用系统提供的软件,如编辑汇编软件 EDASM、LISA 等;也可以用小汇编程序 Mini Assembler。在调试和执行机器语言时,也常用监控程序 MONITOR 中提供的监控命令。下面我们分别介绍这些软件和使用。

### 一、编辑汇编程序(EDASM)的使用

用汇编语言写好的源程序必须首先用编辑软件 EDITOR 将源程序送入计算机,形成源程序的 ASCII 码文件(源文件),存放在磁盘上,这个过程称为编辑。然后调用汇编程序 ASSEMBLER,对磁盘上的源文件进行汇编,形成目标文件送入磁盘,然后才能运行该目标程序。编辑程序和汇编程序已经合并成一个文件,称为 EDASM 文件(编辑汇编),下面介绍它的用法。

#### 1. 编辑汇编程序的启动

启动 EDASM 有两种方法;它们是:

(1)如果系统已经引导了 DOS3.3,则应将 DOS3.3 软盘取出,换以 TOOL KIT(EDITOR 和 ASSEMBLER 均在此盘中)软盘插入驱动器,然后打入命令:

```
]BRUN EDASM. OBJ
```

即从磁盘调入并执行 EDASM 的目标码文件,使系统处于它的管理之下,屏幕上出现提示符:及游标。

(2)如果系统接通电源后,没有在驱动器中插入 DOS,而是插入 TOOL KIT 盘片,则可打入命令:

```
]RUN EDASM
```

从软盘调入并运行 EDASM 的 BASIC 文件,屏幕上出现:

```
APPLE I EDITOR-ASSEMBLER
CURRENT ASSEMBLER ID STAMP IS:
20 JAN 89 # 000001
```

字样,并有闪动游标,此时按一下 RETURN 键,待屏幕上出现 EDASM 提示符:和游标时,系统进入 EDASM 的编辑状态,等待用户输入命令,即可使用。

#### 2. 源程序的编辑

首先用 EDASM 的编辑命令将源程序送入计算机,然后再使用有关命令对源程序进行必要的编辑和修改。

##### (1)A 命令(ADD)——键盘输入源程序命令

使用 A 命令可通过键盘输入汇编语言源程序。例如,将 16 个数据的数据块从内存单元 DATA~DATA

+15 移到 0 页的 \$F0~\$FF 中。源程序输入过程如下:

```
:A
1  ORG $7000
2  BUFFER EQU $F0
3  LDX #0
4  START LDA DATA,X
5  STA BUFFER,X
6  INX
7  CPX #16
8  BNE START
9  RTS
10 DATA DS 16
11 CTRL-D (按住 CTRL 键再按下 D)
```

上述程序输入过程中,每行的行号是机器本身自动产生的。输完一行后应输入回车符(用↵代表)。若某句中有标号,则紧跟行号后打入标号,行号和标号之间不留空格;若某句无标号,则应在行号后敲一空格键。全部汇编语言源程序输入完毕后,用 CTRL-D 敲入表示结束,并退出 A 命令返回“:”状态。

##### (2)L 命令(LIST)——列表显示源程序命令

此命令用于显示已送入内存的文本,其操作为:L 行号↵。

如:L 1-3↵ 显示已输入文本的 1-3 行;:L ↵则显示文本的所有行。

##### (3)P 命令(PRINT)——打印源程序命令

该命令作用同 L 命令,但不显示行号,其操作为:P 行号↵。

##### (4)C 命令(CHANGE)——修改源程序命令

此命令可修改已送入内存中文本的某行,格式为:

```
:C 行号.旧字符串.新字符串
```

```
或:C 行号$旧字符串$新字符串
```

符号.或\$是分界符,不能省略,上述格式的含义是以新字符串代替旧字符串。并回答系统的询问:

```
ALL OR SOME? (A/S)? (全部或部分)
```

若敲 A 键回答,则表示该行中 C 命令指出所有旧字符串都要用新字符串代替;若敲 S 键回答,则表示仅修改部分旧字符串。对要修改的旧字符串,需要再敲一次 S 键,系统才能执行将新字符串替换该旧字符串;对于要保留的旧字符串,则敲 ESC 键跳过。例如,将前例中第 1 行的字符 7 改为字符 8,操作如下:

```
:L 1↵ ;显示第 1 行 1 ORG $7000,以便下面
```



### 观察修改情况

:C 1.7.8 ✓ ;将第 1 行的 7 改为 8

ALL OR SOME? (A/S)?

若按 S 键回答,则表示只修改“7”字符,以后按 ESC 键,则不再作修改。此时用 L 命令

:L 1 ✓

1 ORG \$ 8000

(5)D 命令(DELETE)——删除命令

此命令用来删除已用 A 命令送入机内的文本中的某行(或某连续段),如:

:D 1 ✓ ;删除第 1 行

:D 1-5 ✓ ;删除第 1 到第 5 行

执行 D 命令后,可用 L 命令检查被删除的情况。

(6)I 命令(INSERT)——插入命令

该命令可在文本的规定的行处插入一行语句。如在前面的例子中第 8 行插入 NOP,则操作如下:

:I 8 ✓ ;机器立即空出第 8 行,等待插入

8 NOP ✓ ;在第 8 行插入 NOP

9- ;还可继续在第 9 行插入

如果不需再插入,则按 CTRL-D 退出 I 命令,用 L 命令检查插入后的结果,将发行后续行号已重新排列。

(7)R 命令(REPLACE)——替换命令

此命令对机内的文本中某行(或连续行)进行替换。例如,将上述程序第 10 行的 RTS 改为 SBC,则

:R 10 ✓ ;修改第 10 行

10 BRK ✓

11- ;还可继续修改 11 行

如不需要继续修改,则按 CTRL-D 退出 R 命令,再用 L 命令检查修改后的情况。

(8)E 命令(EDIT)——编辑命令

此命令用来对已用 A 命令送入机器的文本中的某一行作局部编辑,执行:E 行号 ✓,在显示出指示行号内容后,配合使用下述控制键对源程序进行修改:

→ 光标向右移动一个字符位置。

← 光标向左移动一个字符位置。

✓ 编辑结束并有效,退出 E 命令,回到 EDASM 状态。

^ D 删去光标指出位置上的字符。

^ I 在光标指出位置插入字符。

^ T 删去光标指示位置以后的全部字符。

^ R 改写光标指定位置的字符。

^ F 在此命令后跟着打入要在该行寻找的字符,则光标会自动移动到所寻的字符位置上。

^ X 取消正在进行的操作,回到 EDASM 状态。其中 ^ 代表 CTRL 键。

(9)F 命令(FIND)——查找字符命令

此命令用来对已用 A 命令送入机内的文本中找出某行,或某连续段,或某指定字符串所在行。如:

:F 1 ✓ 查找上例中的第 1 行

:F. STA ✓ ;查找上例中含字符 STA 的所有行

4 START LDA DATA,X;第 4 行中含有 STA

5 STA BUFFER ;第 5 行中含有 STA

8 BNE START ;第 8 行中含有 STA

(10)SAVE——存盘命令

SAVE 文件名,可将用编辑命令编辑好的一个用户源文件存入磁盘。例如,我们将上面的汇编源程序起一个名字,如 EXE1,则

:SAVE EXE1 ✓

存盘后可用 CATALOG 命令检查是否已存进盘中。

:CATALOG ✓

这时屏幕上显示的磁盘目录中已有一个名为 EXE1.EDASM 的文件存在了。

(11)LOAD——文件载入命令

LOAD 文件名将指定的文件从磁盘中调入内存。如:LOAD EXE1 ✓,则将文件 EXE1 调入内存中。

以上介绍了如何将一个汇编语言源程序送入计算机,并用上述有关命令进行必要的修改和编辑,汇编语言源程序编辑完成后,可以存盘备用。

### 3. 源程序的汇编

汇编语言源程序经过编辑并存盘以后,就可以执行汇编命令 ASM 源文件名。此命令的作用是对磁盘上指定的源程序文件进行汇编,翻译成目标程序。

(1)如何汇编源程序

例如,有一汇编语言源程序,它将 \$ 2000 和 \$ 2001 两个单元的内容进行十进制相加后,存入 \$ 2002 单元:

ORG \$ 0300

START SED

CLC

LDA \$ 2000

ADC \$ 2001

STA \$ 2002

BRK

则采用下述前步对该程序进行编辑和汇编:

• 启动编辑汇编程序 BRUN EDASM · OBJ ✓

• 将汇编语言源程序送入计算机,并进行编辑

• 将编辑完成后的源程序取名存盘,如 SAVE

SHUX ✓

• 在编辑汇编状态“:”下对 SHUX 进行汇编,即:ASM SHUX ✓。此时,系统首先从 EDASM 文件中调出汇编程序,并在屏幕上显示如下提示:

PRESS ANY KEY TO CONTINUE

这时敲入任一键作回答,机器就开始对盘上的 SHUX 文件进行汇编,并在屏幕上显示源文件名 SHUX、汇编后的目标文件 SHUX.OBJ0 以及汇编后的程序清单:

SOURCE FILE:SHUX

--- NEXT OBJECT FILE NAME IS SHUX.OBJ0

0300 1 ORG \$ 0300

0300:F8 2 START SED

```

0301:18      3      CLC
0302:AD 00 20 4      LDA $ 2000
0305:6D 01 20 5      ADC $ 2001
0308:8D 02 20 6      STA $ 2002
030B:00      7      BRK

```

\*\*\* SUCCESSFUL ASSEMBLY,NO ERRORS

? 0300 START ? 0300 START

汇编完成后敲任意键,系统退出汇编,回到 EDASM 状态,屏幕上出现冒号提示符。至此汇编完毕,若用 CATALOG 命令,可以看到盘上已有一个 SHUX.OBJ0 的目标码文件。

#### (2)如何打印汇编清单

要在汇编过程中打印清单,可按如下步骤操作:

- 打开打印机电源
  - 键入以下命令
- ```

:PR # 1,LXX PXX✓
:ASM 源文件名✓

```

其中 LXX 表示一页中要打印的行数,PXX 表示一页中的总行数。如 L 18 P20,表示这一页共 20 行,打印 18 行后空四行。如 LXX 中填入的数大于 PXX 中填的数,则打印将连续下去,不分页。

#### (3)汇编过程的控制

在汇编过程中,可用下面三个命令,对汇编过程进行控制:

- 停止汇编,回到编辑状态(CTRL-C)
- 暂停汇编(空格键),按任意键继续汇编
- 关闭显示(CTRL-N),CTRL-O 继续显示

#### 4. 目标程序的运行

仍以 SHUX 为例,若要运行它的目标文件 SHUX.OBJ0,其操作如下:

```

:END✓;执行此命令,系统进入 BASIC 状态
]BRUN SHUX.OBJ✓;在 BASIC 状态下运行目标程序
* ;运行完毕,进入监控

```

若要在监控下运行此程序时,可用 0300G✓,这时不用再调盘,而可直接运行已送入内存的目标程序。这里的 0300 是 SHUX.OBJ0 的首地址。

当遇到程序汇编后产生的目标程序不止一个时,例如某程序由三个 ORG 伪指令分成了三段,因而汇编后产生三个目标文件,运行时,首先将三个目标文件分别调入内存,如:

```

]BLOAD NUMBER.OBJ0✓
]BLOAD NUMBER.OBJ1✓
]BLOAD NUMBER.OBJ2✓

```

然后键入命令:

```

]CALL-151✓ ;进入监控状态
* 300G✓ ;从目标程序首地址开始运行

```

## 二、小汇编程序(Mini Assembler)的使用

固化在 CEC-I 中华学习机内的小汇编程序,是一

个简单实用的汇编工具软件,它可以将输入的汇编源程序的符号指令,逐条翻译成机器指令,存放在指定的内存单元中并显示出来,这在我们编写好汇编语言程序后常常采用,它会省去我们查指令表翻译机器码的麻烦。但要注意的是,小汇编程序不认识伪指令,所以使用小汇编时不能用符号地址和标号等,因而只能用来汇编一些简单的源程序。此外,小汇编程序在 APPLE I 机中是放在 DOS 系统主盘的 INTBASIC 文件中,它的启动与中华学习机 CEC-I 有些差别。

#### 1. 进入和退出小汇编状态

(1)对中华学习机 CEC-I 进入小汇编

```

]CALL-151✓;在 BASIC 状态下进入小汇编
* D350G✓

```

! ;! 是进入小汇编的标志

在监控状态下进入小汇编可直接执行上述第二条命令。

(2)对 APPLE I 进入小汇编

• 插入带 INTBASIC 文件的系统主盘,键入 PR # 6✓,待出现了提示符后键入:

```

]INT✓
> ;表示进入整数 BASIC 控制之下
>CALL-151✓
* ;表示进入 RAM 卡的监控程序控制之下
* F666G✓
!

```

(3)对 CEC-I 退出小汇编

```

! $ D360G✓ ;返回监控状态
或! $ FF69G✓ ;返回监控状态
或! $ 35D0G✓ ;返回 BASIC 状态

```

(4)对 APPLE I 退出小汇编

```

! $ FF69G✓ ;返回监控状态
或! $ 3D0G✓ ;返回整数 BASIC 状态
或! INT✓ ;返回整数 BASIC“>”状态
或! FP✓;返回 Applesoft BASIC“]”状态

```

#### 2. 在规定内存地址送入符号指令

在小汇编提示符下,按照地址:符号指令格式操作,就可以将输入的一条符号指令翻译成机器码,并显示在指定的地址中,若输入多字节指令,则将机器码存放在以该地址为首地址的几个连续存储单元中。例如,敲入

```

! 1000;LDA $ 4000✓

```

则输入一条符号指令,汇编一条指令,并显示结果

```

1000-AD 00 40 LDA $ 4000

```

这里 LDA 的操作码是 AD,它放在 \$ 1000 单元中,操作数 00 40 存放在 \$ 1001 和 \$ 1002 单元中。

若想继续输入下一条指令助记符,则不用再输入地址,可在输入“地址”的位置先敲入一个空格,紧接着输入符号指令即可,这时小汇编会自动计算下一个地址。这样,只要设定初始地址后,就可以连续输入一系列的汇编语句。例如,有一完整的汇编源程序,按下述

操作送入：

```
! 300;SED✓  
0300—F8 SED  
!  
! CLC✓  
0301-18 CLC  
!  
! LDA # $ 40 ✓  
0302-A9 40 LDA # $ 40  
!  
! ADC # ¥10✓  
0304-69 10  
!  
! BRK✓  
0306-00 BRK  
!
```

当上述汇编语言输入结束后，汇编工作也就完成，为了显示上述工作的结果，可打入！\$ 300L✓，这样，小汇编程序就把自\$ 0300单元开始存放的目标程序进行反汇编，并显示如下（一次显示 20 行）：

```
! $ 300L✓  
0300—F8 SED  
0301—18 CLC  
0302—A9 40 LDA # $ 40  
0304—69 10 ADC # $ 10  
0306—00 BRK  
0307—00 BRK  
:  
:  
0313-00 BRK
```

### 3. 在小汇编状态下使用监控命令

在小汇编状态下，可以使用中华学习机或 APPLE II 机的各种监控命令，这是非常方便的。使用时应注意在监控命令前加上符号“\$”，这样，监控命令就变成了小汇编状态下的命令。例如：

! \$ 首地址 G✓，运行指定首地址的目标程序。

! \$ 首地址 L✓，从规定的首地址开始列出源程序及对应的机器码。

! \$ 0300✓，检查地址为 \$ 0300 单元的内容。

! \$ 1000:00 00✓，修改从 \$ 1000 单元开始的连续两个单元的内容，使皆为 00。

### 4. 使用小汇编的注意点：

• 小汇编不能识别伪指令。因而在使用小汇编时，不能输入伪指令，也不能用标号和符号地址，只能给出具体数据和绝对地址。

• 符号指令中的操作数或操作数地址，只能使用 16 进制数，不能用其它任一进制数，因为小汇编程序只识别 16 进制数。而且 16 进制表示符号 \$ 可以不写。

• 在使用转移指令时，其地址码部分，小汇编规定只能用转移目标的绝对地址，而不允许用符号地址，也禁止用转移步长。

### 三、监控(MONITOR)命令的使用

监控系统是与计算机结构密切联系的管理计算机的重要程序，它固化在只读存储器 ROM 中，是人们使用计算机的有用工具。监控命令提供对系统的基本操

作，比如机器语言程序的输入、修改、读出、检查、传送等。还为机器语言的使用、运行目标程序、跟踪程序执行、比较寄存器和内存内容，对程序进行反汇编提供了良好的手段。监控系统中许多结构严谨、短小精悍的子程序，便于用户直接调用，大大减少了编制其它程序的工作量。

#### (1) 进入和退出监控状态

• 在 BASIC 状态下进入监控状态，方法很简单，即 CALL-151✓，出现的 \* 号就是进入监控的标志符。

• 在监控状态下返回 BASIC 状态，可用 CTRL-B 或 CTRL-C 键，但前者将使 BASIC 程序及变量全部消失，而后者能保存原来的 BASIC 程序及变量。此外，用 \* 3D0G✓也可返回 BASIC。并保存内存中的 BASIC 程序。

#### (2) 显示存储器的内容

在调试程序时，有时要检查存储单元中的内容以及变化情况，就可以使用监控程序提供的显示存储器内容的功能。它有四种格式：

格式 1：\* 地址✓，这个命令只能显示一个存储单元的内容，如 \* 1000✓，显示 1000—30，这里 30 就是 \$ 1000 单元中的内容，它是一个 16 进制数（下同）。

格式 2：\* 首地址. 尾地址✓，它显示首地址到尾地址的所有存储单元的内容。

格式 3：\* . 地址 2✓，显示从当前地址开始到地址 2 的各存储单元内容。

格式 4：\* ✓，只敲回车键，显示从当前地址开始的八个单元内容。

#### (3) 修改(或输入)内存单元的内容

当需要直接修改某一存储单元的内容，或者在某些内存单元设置一些参数时，可以使用监控提供的这种功能。它有两种格式：

• 格式 1：\* 地址:数据 数据 …✓，这时敲入需要修改的内存单元地址和一个冒号“:”，然后送入要向该单元传送的数据，最后敲回车。若要连续存放一组数据，可在每个数据后敲一空格即可。

• 格式 2：\* :数据 数据 …✓，这是接前面的地址将数据继续送入内存。

如 \* 1000:00 01 02 03✓，这是从 \$ 1000 单元开始连续修改 4 个单元的内容，用这 4 个数据代替原来的数据。又如 \* :04 05 06✓执行后将 \$ 04，\$ 05，\$ 06 放入了 \$ 1004、\$ 1005、\$ 1006 单元。

#### (4) 传送一段内存区域的内容

有时需要将存储器中某一段程序或数据，从某一段内存区域搬到另一段内存区域，可用这个命令。其格式是：\* 目的地址<源地址 1. 源地址 2M✓，例如，将 \$ 2000 到 \$ 3FFF 单元中的内容传送到 \$ 4000 到 \$ 5FFF 中，则可用：

\* 4000<2000.3FFF M✓

又如，将 \$ 1000 到 \$ 4000 单元都填上 00，则可

用:

\* 1000:00  
\* 1001<1000.3FFF M

(5)比较两段内存区域的内容

对内存中两个长度相同的数据块内容进行比较,考察是否一致。当我们想比较一下两个相似程序是否完全一样时,就可以使用这个命令,其格式是:

\* 地址 1<地址 2.地址 3V

命令执行的结果是将地址 2 到地址 3 之间的数据,与从地址 1 开始的长度相等的地址范围内的进行比较。如果一致,则退出该命令,并出现提示符\*;否则就显示相应地址和数据。

(6)检查和修改寄存器暂存区内容

其格式是:\* CTRL-E,命令执行后屏幕上顺次显示 A,X,Y,P,S 五个寄存器在零页的暂存单元(\$0045~\$0049)的内容。如:

\* CTRL-E  
A=43 X=27 Y=00 P=A0 S=F6

这时修改当前各寄存器暂存单元的内容:

\* ;A1 A2 A3 A4 A5

\* CTRL-E  
A=A1 X=A2 Y=A3 P=A4 S=A5

用 G 命令时,监控程序首先将这五个暂存单元的内容分别送入相应各寄存器,然后才进入 G 命令的首地址运行程序。

(7)反汇编命令

汇编语言程序是用指令助记符编写的程序,经过汇编后被转换成二进制的机器语言程序代码,使计算机可以认识和执行。但是,这种二进制代码不易记住,即使对指令代码相当熟悉的人,一一翻译它们也有困难。在调试这种二进制代码的机器语言程序时,人们常常希望看到它们的指令助记符,以便更好地了解程序在做什么。反汇编命令就提供了这种功能,它把内存中的二进制代码按照地址顺序一一列出来,而且还在旁边标出相应的助记符命令,这样一看就明白。例如,已知一段机器语言程序,它存在 \$300 到 \$30C 一段存储区域:

0300-A9 C1 20 ED FD 18 69 01

0308-C9 DB D0 F6 F0

执行 \* 300L 命令后,屏幕上显示如下:

```
0300 A9 C1 LDA # $C1
0302 20 ED FD JSR $FDED
0305 18 CLC
0306 69 01 ADC # $01
0308 C9 DB CMP # $DB
030A D0 F6 BNE $0302
030C 60 RTS
030D 00 BRK
:
0313-00 BRK
```

这个程序就十分明确了,首先将 A 的代码 C1 放进累加器,调用输出一个字符子程序,其入口地址为 \$FDED,然后清进位位,累加器内容加 1,判断是否为 \$DB(Z 的代码是 \$DA),不是跳转 \$0302 继续操作,是则返回调用程序。由此可知,本程序可以在屏幕上显示 26 个英文字母。

所以,\* 地址 L 的操作是把指定地址开始的 20 条机器语言指令翻译(反汇编)成符号指令,并显示在屏幕上。而 \* L,则是将从当前地址开始的 20 条机器指令反汇编成符号指令并显示出来。

(8)运行机器语言程序命令

用汇编语言编好了程序,经汇编后将目标程序装入内存,你就可以用下述命令来执行程序。如:

\* 1000 G 即从 \$1000 单元开始执行程序。

\* 1000 S,这是单步命令,执行从 \$1000 单元开始的一条指令,并显示寄存器值和指令助记符,例:

```
1000 A9 43 LDA # $43
A=43 X=CE Y=D8 P=30 S=EC
```

\* 1000T,这是一种跟踪命令,它把多个单步操作连起来做,直到执行完一条 BRK 指令为止。

此外,还有屏幕显示方式命令,选择输入、输出设备命令等,如:

\* I,置屏幕为反相显示方式。

\* N,置屏幕为正常显示方式。

\* 1^P,启动打印机,显示打印结果。

\* 0^P,停止打印,回到屏幕显示。

\* 6^P,转入磁盘驱动器

\* 3^P,进入中华学习机中文状态。

\* 0^K,进入以键盘为输入设备的状态。

\* 地址 1.地址 2 R,将磁带上的数据读入指定的内存单元。

\* 地址 1.地址 2W,将内存地址 1 到地址 2 中的信息写到磁带上。

\* 300L 300G,是多重命令,反汇编并执行从 \$300 开始的机器语言程序。

#### 四、编辑和汇编程序 LISA 简介

编辑和汇编程序 LISA,是 6502 汇编程序中比较适合初学者的一种工具软件,其特点是:能即时提示输入语法错误;汇编过程在内存中进行,不读写磁盘,速度快;结合 16K RAM 卡或语言卡,一次可汇编 4K 字节的目标代码。其不足之处是编辑功能没有 TOOLKIT 的 EDASM 强,目标码存盘时要用户算出实际代码长度,用 DOS 命令写入磁盘。LISA 有 15 条操作命令,33 条伪指令,操作码助记符除与 6502 指令一致外,还扩展了 5 条;BTR(相当于 BEQ),BFL(BNE),BGE(BCS),BLT(BCC)和 XOR(EOR),操作命令有 I(插入),D(删除),M(修改),L(列程序),LO(读入源程序),SA(源程序存盘),A(汇编),N(清源程序),BRK(进入监控),F(查找标号)等等。

## 初级程序员级软件水平考试辅导

# 数据库基础

北京电脑天地学校(100051) 阚璐

### 习 题

一、从供选择的答案中,选出应填入下面 \_\_\_\_\_ 内的正确答案。

1. 内模式用 \_\_\_\_\_ 语言定义。  
A. 设备介质 B. 模型介质 C. 结构介质 D. 外型介质
2. 概念模式用 \_\_\_\_\_ 语言定义。  
A. 集合定义 B. 查询定义 C. 运算定义 D. 模式定义
3. 外模式用 \_\_\_\_\_ 语言定义。  
A. 关系模式定义 B. 层次 C. 子模式定义 D. 操纵
4. 下面 \_\_\_\_\_ 不属于数据模式。  
A. 查询模式 B. 外模式 C. 概念模式 D. 内模式
5. 用树型结构来表示实体之间联系的模型称为 \_\_\_\_\_。  
A. 关系模型 B. 层次模型 C. 网络模型 D. 运算模型
6. 用表格数据来表示实体之间联系的模型称为 \_\_\_\_\_。  
A. 关系模型 B. 层次模型 C. 网络模型 D. 运算模型
7. 关系数据库的任何检索操作的实现都是由三种基本检索运算组合而成的,这三种基本运算不包括 \_\_\_\_\_。  
A. 选择 B. 投影 C. 关闭 D. 联结
8. DML 是 \_\_\_\_\_ 语言,又是 \_\_\_\_\_ 语言;DDL 是 \_\_\_\_\_ 语言,又是 \_\_\_\_\_ 语言。  
A. 非过程化 B. 过程化 C. 系列化 D. 非系列化  
E. 数据操纵 F. 数据选择 G. 数据运算 H. 数据定义
9. 关系模型的数据语言是 \_\_\_\_\_ 的语言,其核心部分为查询,因此又称为查询语言。  
A. 非过程化 B. 过程化 C. 系列化 D. 非系列化
10. 在数据库 dBASE III 中,一个过程文件中允许最多包含的过程为 \_\_\_\_\_ 个。  
A. 256 B. 28 C. 32 D. 128
11. 数据操作语言一般不具有的功能 \_\_\_\_\_。  
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
12. 以下说法哪个不正确 \_\_\_\_\_。  
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

13. 数据库管理系统和操作系统之间的关系是 \_\_\_\_\_ A \_\_\_\_\_。为实现数据保护,数据库管理系统通常提供了保证数据 \_\_\_\_\_ B \_\_\_\_\_、\_\_\_\_\_ C \_\_\_\_\_ 及并发控制等方面的机制。数据库系统采用的数据模型有 \_\_\_\_\_ D \_\_\_\_\_ 为使用数据库方便,常常将数据库管理系统提供的数据库操作语言嵌入到某一高级语言中,称此高级语言为 \_\_\_\_\_ E \_\_\_\_\_。

供选择的答案

- A ①数据库管理系统调用操作系统  
②操作系统调用数据库管理系统  
③两者互相调用
- B、C:①随机性 ②有效性 ③完整性 ④安全性  
⑤相容性 ⑥顺序性
- D:①网状模型,链状模型和层次模型  
②层次模型,分散模型和环状模型  
③层次模型,网状模型和关系模型
- E:①虚拟 ②会话 ③宿主

二、从下列叙述中选出 5 条正确的叙述。

1. 用数据描述语言精确地定义数据模型的程序称为模式。
2. 数据库系统是一组软件的组合,可应用于各种程序。
3. 在 DBF 文件中存在记录的情况下,使用 GO BOTTOM 命令肯定能使记录指针指向记录号最大的记录。
4. 在数据库中,对数据(结构)的描述称为数据模式。
5. 用户使用的数据视图叫外模型。
6. 全局的逻辑数据视图叫概念模型,又叫数据模型。
7. 数据库管理系统可以用于各种应用程序。
8. 数据库系统即数据库管理系统,是由硬件、软件、用户等组成的。
9. 物理数据存储的模型叫内模型。
10. 在数据库系统中,数据一致性系指数据库中的数据类型一致。

三、现有一个数据库 ABC.DBF,库中有 10 条记录,请写出顺序执行下面各命令后库指针的位置。

- .USE ABC
- .GO 5
- .SKIP-3
- .DISPLAY

```
. LIST NEXT 4
```

```
. GO BOTTOM
```

#### 四、在下述程序中

```
IF a<b
```

```
LOOP
```

```
SKIP
```

```
ENDIF
```

语句 SKIP 是永远得不到执行的(是否正确)。

#### 五、有一个已经打开的 DBF 文件,文件名为 BH-

SL. DBF,用 LIST 命令所显示的结果为:

| RECORD # | 编号  | 数量 |
|----------|-----|----|
| 1        | A1  | 10 |
| 2        | A0  | 85 |
| 3        | A2  | 67 |
| 4        | A10 | 50 |
| 5        | A12 | 65 |

(注:“编号”字段为 C 型,“数量”字段为 N 型)阅读下列程序段:

```
SET TALK OFF
```

```
CLEAR
```

```
USE BHSL
```

```
a=数量
```

```
DO WHILE .NOT. EOF()
```

```
IF a<数量
```

```
    a=数量
```

```
ENDIF
```

```
SKIP
```

```
ENDDO
```

```
? 编号,a
```

此程序执行后,屏幕中所显示的内容为( )

供选择的答案:

(1)A0 85 (2)A1 10 (3)85 (4)10

## 习 题 分 析

一

1. 内模式(或存储模式)又称物理模式,是对数据库的存储结构等方面内容的描述,用物理模式定义语言来进行描述,有些系统,把描述存储模式的语言称为设备介质控制语言,或数据存储描述语言。

2. 概念模式简称为模式,是对数据库的全局逻辑结构的描述,用模式定义语言来进行描述。

3. 外模式又叫子模式,是对数据库的局部逻辑结构的描述,具有相同数据视图的用户共用一个子模式,用子模式定义语言来进行描述。

4. 在数据库中,对数据(结构)的描述称为数据模式。数据视图被划分为三个层次,那么对数据结构进行描述的数据模式也分为三个层次。与外部层,概念层和内部层相对应,数据模式分为外模式、概念模式和内模式。这是 ANSI(美国国家标准化协会)模型的划分方法。

5. 数据库的特征之一是数据库中的数据是有结构

的,即体现了数据之间的联系,因此数据库系统必须研究记录间的联系。所谓数据模型就是指具有这种联系的数据结构形式。

在格式化模型中,用树的数据结构形式来表示实体及其联系即为层次模型,这种树由结点和连线组成。结点表示实体集,而连线表示相连两个实体之间的关系。

6. 用表格数据来表示实体间联系的模型叫关系模型。关系模型是数学化的。它把数据看成二维表中的元素,而这个表就是关系。

7. 关系型数据库管理系统所必须具有的三种操作是:选择、投影和联结。所谓“选择”就是从 n 维空间内的所有点中选择出满足一定条件的那些点,而除出其它点。例如工资高于 100 元的职员表。通常包括指定范围选择和指定条件选择两种。“投影”操作是指只对记录类型中的某些项进行数据操作。而“联结”操作指的是将两个库文件按一定的条件联结成一个新的库文件。

8. DML(Data Manipulation Language)是数据操纵语言,它是数据库管理系统提供给应用程序员用以存储、检索、修改、删除数据库中数据的工具。DML 是过程化语言。

DDL(Date Description Language)是数据定义语言,它是数据库中用来描述数据及其关系的语言。是数据库管理软件的一部分。具体描述数据库管理系统中的数据结构、数据库的逻辑特征。描述的对象包括初等项、组项、记录等。DDL 是非过程化语言。

9. 关系数据库使用的语言一般具有定义、查询、更新和控制一体化的特点。它即可以嵌入到宿主语言,又可作为独立的交互语言使用;关系模型的数据语言是非过程化的语言,其核心部分为查询,因此又称为查询语言。

10. 在 dBASE III 中,一个命令文件可以包含过程也可以不包含过程,但过程总是包含在某个命令文件中的,包含过程的命令文件又称为过程文件,一般允许一个过程文件中最多含 32 个过程。

11. 数据操作语言适应于 DBMS 的数据模型,对于不同的数据模型,数据操作语言是不同的。数据操作语言一般应具有如下功能:

(1)从数据库中检索数据,这是最重要、最常用的一类操作。

(2)向数据库中添加数据。

(3)删除数据库中某些已经过时,没有保留价值的原有数据。

(4)修改某些属性发生了变化的数据项的值,使之能确切反映变化后的情况。

(5)用于并发访问控制的操作。

12. 数据库管理系统技术的特点是:

(1)减少数据的冗余;

(2)避免数据的不一致;

(3)数据可以共享;

- (4)有数据安全和完整性保障及并发控制;
- (5)多用户操作的并行调度;
- (6)易于使用;
- (7)便于扩展

13. 操作系统是对计算机硬件的第一级扩充,它是系统软件,是其它软件的运行基础,因此数据库管理系统和操作系统之间的关系是操作系统调用数据库管理系统。根据 12 题解释的数据库管理系统技术的特点可以说明,为实现数据保护,数据库管理系统通常提供了保证数据完整性、安全性及并发控制等方面的机制。

数据库系统的一个核心问题就是研究如何表示和处理实体间的联系。人们把表示实体和实体之间联系的模型叫数据模型。

常用的数据模型有三种,层次模型、网络模型和关系模型。也就是通常人们所说的层次数据库、网状数据库和关系数据库。

为了克服自含系统在数据处理能力方面的不足,可以将数据库管理系统提供的数据库操作语言(DML)嵌入到某一个高级语言中去。这个高级语言称为宿主语言。整个系统就称为宿主系统,宿主型的数据库操作语言(DML)是过程化语言。

答案:1. A 2. D 3. C 4. A 5. B 6. A 7. C  
8. E, B, H, A 9. A 10. C 11. B 12. B 13. A② B③ C④ D③ E③

二

1. 正确

2. 错误。数据库系统是指计算机系统中引进数据库后的系统构成。构成数据库系统至少有四个成份,即数据、硬件、软件 and 用户。

3. 错误。此题没有明确指出用户打开了什么类型的文件,只是说数据库文件有记录,如果题目明确指出打开的是数据库文件而没有打开其它文件,可以说是正确的;但是如果此题还打开了索引文件,则题目说法是不对的,因为对数据库中不同的字段进行索引,记录号排列的顺序是不一样的。因此说本题目的答案是错误的。

4. 正确。

5. 正确。

6. 正确。

7. 错误。数据库系统主要不是用来作科学计算,它主要用来处理各种报表,是管理现代化和办公自动化的得力工具之一。

8. 错误。数据库系统和数据库管理系统之间的关系是数据库系统包含有数据库管理系统。

9. 正确。

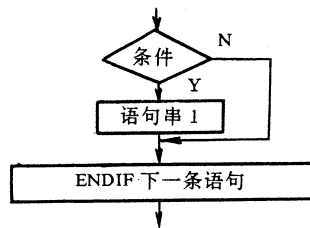
10. 错误。数据库系统中数据的一致性系指在数据库系统管理数据(修改、删除或增加数据)时,数据的一致。

三、题目明确指出打开的是数据库文件,因此题目的答案为:1、5、2、2、5、10。

在做指针类题目时,应注意数据库文件记录指针、

排序后生成的数据库文件记录指针和索引文件记录指针指向的区别。数据库文件记录的排列顺序是物理排列,排序后生成的库文件记录的排列顺序是顺序排列,而索引文件记录的排列顺序是逻辑排列,因此它们的首记录和尾记录的记录号可能相同也有可能不同,这要看索引文件是按照什么关键字进行索引的。

四、本题是条件语句结构,执行条件语句的框图为:



当条件成立( $a < b$ )时,执行语句串 1,也就是本题目中的 LOOP 语句,而 LOOP 语句是回到循环头的命令;那么当条件不成立( $a >= b$ )时,应当执行 ENDIF 的下一条语句。因此 SKIP 语句是永远得不到执行的。

五、程序首先关闭无用信息,清屏,开库,把数量送给 a 内存变量,此时 a 中存放的数字应为 10,进入循环结构,条件为在没有遇到库文件尾时循环,条件语句判断,将最大的数量送给内存变量 a 也就是挑选最大数,最后输出编号和 a。当库文件指针函数为真( $EOF() = T.$ ),也就是退出循环时,指针指向最后一条记录之外,因此编号中没有数据。因此题的答案应为——(3) 85。

(上接第 23 页)

6218- FB 90 05 A9 A0 4C 22 62  
6220- A9 CD 20 ED FD CA D0 E0  
6228- C8 C4 09 90 D5 20 8E FD  
6230- D8 18 A5 06 69 28 85 06  
6238- A5 07 69 00 85 07 C6 FA  
6240- D0 BE 60

程序 C:

```
10 INPUT "P=" ; P
15 INPUT "V=" ; V
20 POKE 6,0;POKE 7,32 * P
25 POKE 252,0;POKE 253,32 * (3-P)
30 CALL 24832
35 INPUT "X1,Y1=" ; X1,Y1
40 INPUT "X2,Y2=" ; X2,Y2
45 N1 = (3-P) * 8192 + Y1 * 40
50 N3 = INT(N1/256); N2 = N1 - N3 * 256
60 POKE 6, N2; POKE 7, N3
65 POKE 8, X1/7; POKE 9, X2/7
70 POKE 250, Y2 - Y1 + 1; POKE 251, V
75 PR # 1
80 CALL 25088
85 PR # 0 ; END
```



# 提高单片机最小系统抗干扰能力和自恢复方法

合肥中国科技大学计算中心[230026] 张培仁 刘振安

单片机也象其他计算机一样遵循程序存储和程序控制的原理。最小系统的程序存储在 EPROM 中,单片机接通电源后从 0000H 地址开始执行一条一条指令。这时程序计数器(PC)不断指出目前程序执行到何处。一旦系统受到干扰后,CPU 取指令取的是随机错误拼写起来的指令,从而造成不是按原来程序进行工作,造成程序“跑飞”以至系统死锁。这时往往可冲乱 RAM (包括片内和片外)的数据。最终造成系统失控。这种现象在工业控制系统中特别突出。工业控制系统中,在强电、强磁场环境下工作的单片机系统易受干扰。那么我们如何以较低的成本,用硬软相结合的方法提高系统本身抗干扰能力呢?

在回答这个问题之前,首先要了解一般单片机系统的干扰源主要组成。干扰源由三大部分组成:信号线引入的干扰(电场、磁场、电磁波);电源引入的干扰;地线干扰。

要想提高抗干扰的能力,首先要尽可能防止干扰的产生,尽可能隔离单片机的输入输出信号和干扰信号的联系。可采用光电隔离,一般用 4N25,4N36 等。见图 1。

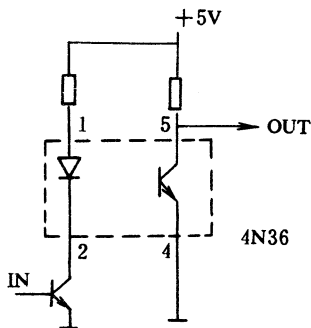


图 1

对于输入是模拟信号,主要采用屏蔽和浮地技术。严格区分模拟地、数字地和系统地,防止数字信号和模拟信号相互干扰。对 A/D 变换器来说,当干扰很大时,应适当增加信号的积分。当采样速度要求在 0.1 秒数量级时,可采用积分式 A/D 变换器。这样抗干扰能力明显提高。对于大功率开关、按键,电磁阀为了减少对单片机系统的干扰,可在强电回路中增加电阻,电容吸收回路,以便减少干扰。

以上这些都是系统注意防止干扰进入用户系统。一旦发生干扰又如何尽快排除干扰而恢复系统正常运行呢?一般是强迫系统复位或进入处理干扰的程序。但是如何知道已经有干扰呢?这里存在着一个如何测试、

分析、判断干扰已经发生的问题。只有判断的正确,才能通过处理,回到相应正常程序去执行。

系统受干扰后发生两种跑飞的现象。一种在用户程序内部转来转去,另一种现象是跳出用户程序而进入系统未使用的程序空间。对于这两种情况都应尽量减少 CPU 执行错误指令的时间。对于后一种情况,判断干扰已经发生是比较容易。例如 KDC-III 开发机中,138 译码器输出  $Y_0$ — $Y_7$ ,而系统只用  $Y_0$ ~ $Y_2$ , $Y_3$  至  $Y_7$  都未使用,可用硬件接成如图 2。

一旦 PC(程序计数器)脱轨而进入系统未使用区,就会产生中断请求,CPU 响应后进入中断处理,这时 CPU 先找出栈顶,将栈顶存储单元的下一个单元内容清零,此时将 0000H 装入堆栈,中断返回后,0000H 弹回 PC,系统复位。

如果跑飞的程序在用户程序内部跳转,这时将如何识别,判断呢?程序设计时常又是模块化设计,按照程序的要求一个模块、一个模块执行。在执行本模块时,为下一模块设立一个标志码。进入下一模块后再检测该标志码,如果正确就正常执行,如果不正确认为是非正常进入本模块的。这样就象接力棒一样,一级传一级,各模块有自己的标志码。这样程序很容易判断出是否是“跑飞”来的跳转,还是正常跳转过来。关键是跳转时是否带有本模块的标志码。一旦发现非正常程序在执行,即转到错误处理程序中。

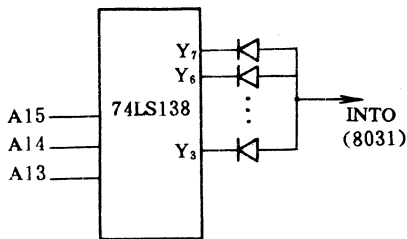


图 2

对这种“跑飞”程序还有另外一个办法解决。就是在用户程序中设置一些陷阱。在正常程序中不执行这些程序,也就是不会进入陷阱。而一旦“跑飞”就有可能遇到这些陷阱,只要遇到一个,马上进行错误处理程序,做出相应的处理。这些陷阱的指令组如下:

```
NOP
NOP
NOP
NOP
AJMP SLG
```

这样的指令组要有多组散放在用户程序各模块之间空余的单元里。这组指令前面是四个空操作,目的使



PC 跑飞后的指令首地址纳入正轨,以保证最后执行跳转指令。这个方法很有效,陷阱的多少一般依用户程序大小而定,一般每 1K 字节有几个陷阱就可以了。

还有一个方法可以很方便提高抗干扰的能力。一般用户 EPROM 常常是用不完的。可以把余下单元全部写成 LJMP 0000H 指令。一旦“跑飞”很快回到初始化程序,不会死锁。

不论哪一种“跑飞”,不论是在用户区还是在未使用空间区跳转时都有可能把 8031 片内和片外的 RAM

破坏掉。这时将如何尽快判断 RAM 已被破坏,并尽快恢复原来的数据呢?这里主要的方法是:在系统初始化,就在一些用户不用 RAM 区域的几个单元中(最好散列在整个 RAM 空间)放入特定标志数,当 RAM 受破坏时,这些标志将发生变化,读这些标志数与原来设定值比较,即可判断 RAM 是否被破坏。一般在 1K 的 RAM 有几个字节就可以了。

以上介绍几种简单容易实现抗干扰的方法,这些方法将系统可靠性大大提高了。

## TP801 单板机与微型机数据双向并行传送一例

江西九江国棉二厂(332006) 刘浔和

利用 TP801 单板机和微型机(笔者使用 BCM-3 型机)中的 PIO 芯片可以组成简单的双向并行传送装置,按图联接 9 根导线即可。

程序 1

```
org 2000h      ;
ld hl,800h    ;数据起始地址
ld c,06h     ;数据长
ld b,00      ;
ld a,4fh    ;置 PIO 为非中断输入
out (83h),a ;B 口
ld a,07
out (83h),a
tina,(81h)
cp 13h      ;控制数
jr nz,t
tl in a,(81h)
cp 13h
jr z,t1
ld (hl),a
inc hl
djnz t
dec c
jr nz,t
```

程序 2

```
org 2000h
ld hl,800h
ld c,06
ld b,0
ld a,0fh
out (246),a ;置 PIO 为非中断输出
ld a,07
out (246),a ;A 口
tld a,13h ;控制数
out (244),a
ld d,10 ;延时
tl dec d
```

```
out (244),a
jr nz,t1
ld a,(hl)
out (244),a
ld d,06 ;延时
t2dec d
out (244),a
jr nz,t2
inc hl
djnz t
dec c
jr nz,t
jp 0001 ;返回监控
```

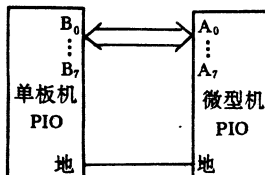


图 1

当微型机向单板机送数时,单板机先执行程序 1,微机再执行程序 2。在程序中,13H 为控制数,应是传送数据中没有出现的数,否则,在传送中会丢失 13H 这个数。另由于单板机与微机的频率不一样,且 PIO 输电平也要达到稳定。因此增加了延时部分,延时量由实际情况来定。

当单板机向微机送数时,应将两个程序中的输入、输出口交换,然后微机先执行程序 1,单板机再执行程序 2。

该方法简化了硬件部分,对于其它的单板机和微机,单板机和单板机及微机和微机,只要装有并行输入、输出芯片,都可按此法传送数据,但要改动程序中的输入、输出口和延时量。

# MCS—51 汇编指令外延的应用

湖南怀化地区电子研究所(418000) 陈亿善

在使用 MCS—51 单片机汇编语言编写程序时,经常会遇到工作寄存器 R<sub>1</sub>~R<sub>7</sub> 之间进行数据传递。而在 MCS—51 的指令系统中没有工作寄存器 R 之间传递数据的指令。例如要把 R<sub>3</sub> 的数据送到 R<sub>5</sub> 中,如果写成:

```
MOV R5,R3
```

在用 MCS—51 汇编软件汇编时,就会指出这条指令是错的,因为 MCS—51 的指令系统中找不到与这条指令相对应的机器码。在一些介绍 MCS—51 指令和编程技巧的书中,都是采用累加器 A 或其他寄存器中转的办法来实现 R 之间的数据传递,如用 A 中转,上例便写成:

```
MOV A,R3
MOV R5,A
```

上面两条指令虽然可以达到传送数据的目的,但在编写时要多用一条指令,还动用了累加器 A,如果 A 中原来的数据紧接着在下面程序中用到的话,还得先把 A 中的数据保护起来,中转完后又得调回 A 中,这样还需增加两条指令。笔者在编写程序中发现,MCS—51 指令有些还有很深的外延,如实现 R 之间的数据传递,应用 MCS—51 指令的外延,用一条指令完全可以达到目的,而且无需动用其他寄存器。编写时只要把指令中的两个 R 中的一个改写成该 R 的直接地址即可。如上例(假设 PSW 选择的 R 区在 0 区)可以写成:

```
MOVR5,03H
```

或:

```
MOV 05H,R3
```

这两条指令都可以达到把 R<sub>3</sub> 中的数据送入 R<sub>5</sub> 中的目的。这种编写方法在各种版本的 MCS—51 汇编软件下汇编,都能生成正确的机器码。编程实践表明,除了使用 R<sub>0</sub> 或 R<sub>1</sub> 进行间接寻址时,@R<sub>0</sub> 或 @R<sub>1</sub> 中的 R 不能用其直接地址代替外,其他任何场合 R<sub>0</sub>~R<sub>7</sub> 都可以用其直接地址来代替。比如在压栈和弹栈指令中,没有把 R 压入堆栈和弹出堆栈的指令,即 PUSH R<sub>n</sub> 和 POP R<sub>n</sub>(n 为 0~7)指令在 MCS—51 指令系统中是找不到的,但象 PUSH 01H(等同于 PUSH R<sub>1</sub>),POP 05H(等同于 POP R<sub>5</sub>)之写法在指令系统中是可以找到相对应的机器码的。R 在不同工作区的直接地址见下表:

| RS <sub>1</sub> | RS <sub>0</sub> | R 工作区位 | R <sub>0</sub> | R <sub>1</sub> | R <sub>2</sub> | R <sub>3</sub> | R <sub>4</sub> | R <sub>5</sub> | R <sub>6</sub> | R <sub>7</sub> |
|-----------------|-----------------|--------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 0               | 0               | 0      | 00H            | 01H            | 02H            | 03H            | 04H            | 05H            | 06H            | 07H            |
| 0               | 1               | 1      | 08H            | 09H            | 0AH            | 0BH            | 0CH            | 0DH            | 0EH            | 0FH            |
| 1               | 0               | 2      | 10H            | 11H            | 12H            | 13H            | 14H            | 15H            | 16H            | 17H            |
| 1               | 1               | 3      | 18H            | 19H            | 1AH            | 1BH            | 1CH            | 1DH            | 1EH            | 1FH            |

(上接 43 页)

STR \$ 语句的格式为:

```
STR $(X) X 取值—32768~32767
```

例 10.

```
10 I. "A";A
20 I. "B";B
30 P. A+"B"="A+B
40 P. "STR$( "A")+STR$( "B")="";
STR$(A)+STR$(B)
50 G. 10
```

Run 显示

```
A? 键入 125 回车
```

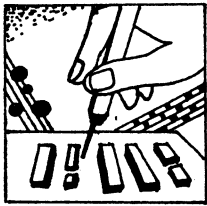
```
B? 键入 135 回车
```

```
125+135=260
```

```
STR$(125)+STR$(135)=125135
```

⋮

读者不难发现,通过 STR \$ 语句的处理,原来的十进制数 125、135 虽然外形没有改变,但已发生质的变化,这时已不再是数字而变为字符串。因此,当它们相加时,就按照字符串的运算规则相加了。



## 学装微电脑

# 制作能暂停显示的计数器

易齐干

计数器多数是计量输入的脉冲数,连续不断地显示。也有随用途不同,连续计数过程中,有要求暂停显示的情况。例如,检验一定时间的生产量时,计数器的显示有暂停功能,则会带来很大方便。这就是本文讨论的计数器。

### 1. 硬件安排

显示有暂停功能的计数器必须具备三部份:脉冲输入端;控制显示更新、暂停的开关;脉冲计数结果显示部份。照图1连接 $\mu P-80$ 教育套件。电源部件提供50Hz脉冲送入输入输出部件C口第0位;4位数字显示部件显示微电脑计数结果;C口高位相连的打码开关 $S_0$ 开关(C口第4位)“通”暂停显示;“断”恢复显示。这就是本文讨论的计数器。

### 2. 软件开发方法

可按如下步骤进行:

- 1)按功能展开课题。
- 2)绘制简单流程图。
- 3)分配RAM区域。
- 4)绘制详细流程图。

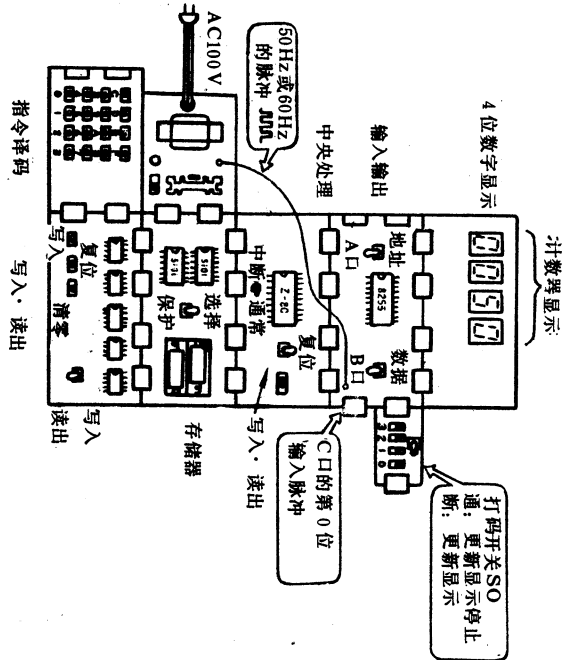


图1. 带有显示暂停功能的计数器硬件。

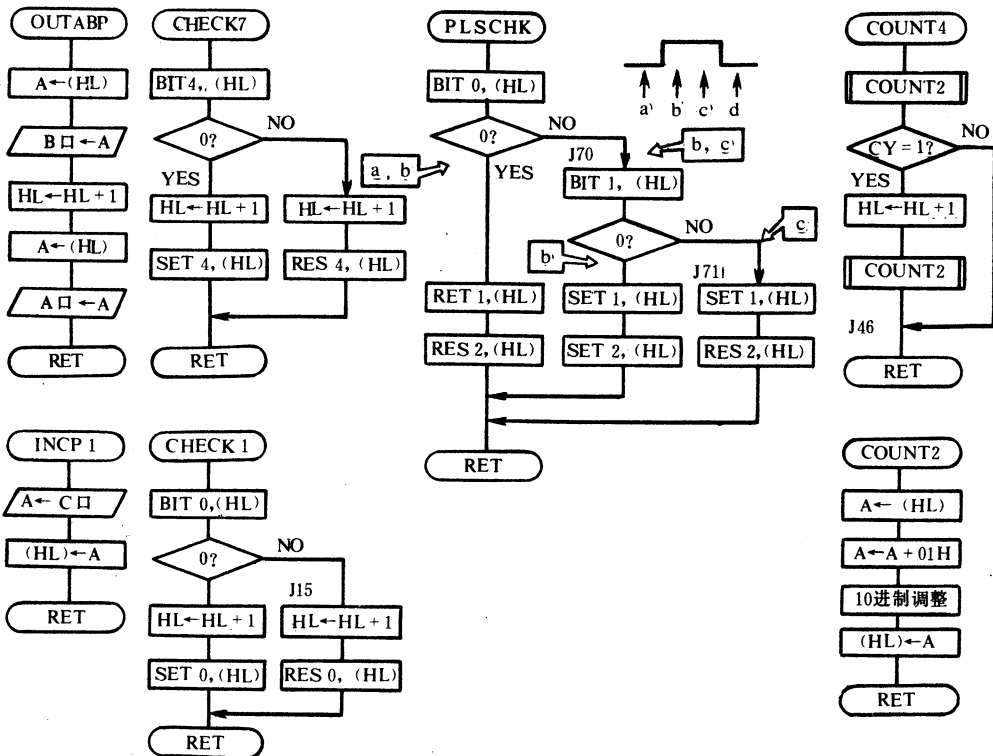


图2(b)详细流程图

表 1 RAM 区域。

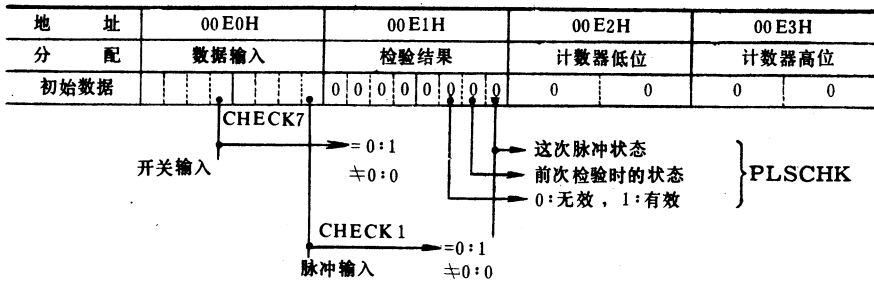


表 2. (a)程序清单(1)

表 2(b)程序清单(2)

| 标号          | 助记符          | 地址           | 机械语言     | 注 释              |
|-------------|--------------|--------------|----------|------------------|
| START       | LD A, 89H    | 0000         | 3E 89    | ) 设置输入输出<br>设置 S |
|             | OUT(03H), A  | 0002         | D3 03    |                  |
|             | LD SP, 0100H | 0004         | 31 0001  |                  |
| INIT        | LD B, 03H    | 0007         | 06 03    | 设置初始值            |
| J1          | LD HL, 00E1H | 0009         | 21 E1 00 |                  |
|             | LD(HL), 00H  | 000C         | 36 00    |                  |
|             | INC HL       | 000E         | 23       |                  |
|             | DEC B        | 000F         | 05       |                  |
|             | JP NZ, J1    | 0010         | C2 0C 00 |                  |
| OUTPUT      | LD HL, 00E1H | 0013         | 21 E1 00 | 输出               |
|             | BIT 4, (HL)  | 0016         | CB 66    |                  |
|             | JP NZ, INPUT | 0018         | C2 1F 00 |                  |
|             | INC HL       | 001B         | 23       |                  |
|             | CALL OUTABP  | 001C         | CD 48 00 |                  |
| INPUT       | LD HL, 00E0H | 001F         | 21 E0 00 | 输入               |
|             | CALL INCP1   | 0022         | CD 50 00 |                  |
| PROCESS     | LD HL, 00E0H | 0025         | 21 E0 00 | 处理               |
|             | CALL CHECK7  | 0028         | CD 54 00 |                  |
|             | LD HL, 00E0H | 002B         | 21 E0 00 |                  |
|             | CALL CHECK1  | 002E         | CD 61 00 |                  |
|             | LD HL, 00E1H | 0031         | 21 E1 00 |                  |
|             | CALL PLSCHK  | 0034         | CD 6E 00 |                  |
|             | LD HL, 00E1H | 0037         | 21 E1 00 |                  |
|             | BIT 2, (HL)  | 003A         | CB 56    |                  |
|             | JP Z, OUTPUT | 003C         | CA 13 00 |                  |
|             | LD HL, 00E2H | 003F         | 21 E2 00 |                  |
|             | CALL COUNT8  | 0042         | CD 87 00 |                  |
|             | JP OUTPUT    | 0045         | C3 13 00 |                  |
|             | OUTABP       | LDA, (HL)    | 0048     |                  |
| OUT(01H), A |              | 0049         | D3 01    |                  |
| INC HL      |              | 004B         | 23       |                  |
| LD A, (HL)  |              | 004C         | 7E       |                  |
| OUT(00H), A |              | 004D         | D3 00    |                  |
| RET         |              | 004F         | C9       |                  |
| INCP1       |              | LD HL, 00E0H | 0050     | DB 02            |
| LD(HL), A   | 0052         | 77           |          |                  |
| RET         | 0053         | C9           |          |                  |

| 标号          | 助记符         | 地址       | 机器语      | 注 释     |                 |
|-------------|-------------|----------|----------|---------|-----------------|
| CHECK7      | BIT 4, (HL) | 0054     | CB 66    | 开关输入的检验 |                 |
|             | JP NZ, J2   | 0056     | C2 5D 00 |         |                 |
|             | INC HL      | 0059     | 23       |         |                 |
|             | SET 4, (HL) | 005A     | CB E6    |         |                 |
|             | RET         | 005C     | C9       |         |                 |
| J2          | INC HL      | 005D     | 23       |         |                 |
|             | RES 4, (HL) | 005E     | CBA 6    |         |                 |
|             | RET         | 0060     | C9       |         |                 |
|             | BIT 0, (HL) | 0061     | CB 46    |         | 脉冲输入的检验         |
|             | JP NZ, J15  | 0063     | C2 6A 00 |         |                 |
| INC HL      | 0066        | 23       |          |         |                 |
| SET 0, (HL) | 0067        | CBC 6    |          |         |                 |
| RET         | 0069        | C9       |          |         |                 |
| J15         | INC HL      | 006A     | 23       |         |                 |
|             | RES 0, (HL) | 006B     | CB 86    |         |                 |
|             | RET         | 006D     | C9       |         |                 |
|             | BIT 0, (HL) | 006E     | CB 46    |         | 检测脉冲的一个峰        |
|             | JP NZ, J70  | 0070     | C2 78 00 |         |                 |
| RES 1, (HL) | 0073        | CB 8E    |          |         |                 |
| RES 2, (HL) | 0075        | CB 96    |          |         |                 |
| RET         | 0077        | C9       |          |         |                 |
| J70         | BIT 1, (HL) | 0078     | CB 4E    |         |                 |
|             | JP NZ, J71  | 007A     | C2 82 00 |         |                 |
|             | SET 1, (HL) | 007D     | CB CE    |         |                 |
|             | SET 2, (HL) | 007F     | CBD 6    |         |                 |
|             | RET         | 0081     | C9       |         |                 |
| J71         | SET 1, (HL) | 0082     | CB CE    |         |                 |
|             | RES 2, (HL) | 0084     | CB 96    |         |                 |
|             | RET         | 0086     | C9       |         |                 |
|             | CALL COUNT2 | 0087     | CD 92 00 |         | 10进制 4 位<br>计数加 |
|             | JP NC, J46  | 008A     | D2 91 00 |         |                 |
| INCHL       | 008D        | 23       |          |         |                 |
| CALL COUNT2 | 008E        | CD 92 00 |          |         |                 |
| RET         | 0091        | C9       |          |         |                 |
| J46         | LD A, (HL)  | 0092     | 7E       |         |                 |
|             | ADD A, 01H  | 0093     | C6 01    |         |                 |
|             | DAA         | 0095     | 27       |         |                 |
|             | LD(HL), A   | 0096     | 77       |         |                 |
|             | RET         | 0097     | C9       |         |                 |

5) 编制程序清单。

以下逐项进行讨论:

1) 本课题归纳为两大功能: 脉冲计数; 显示计数结果。

脉冲计数包括: 脉冲输入; 脉冲检验; 脉冲计数。

显示计数结果包括: 打码开关 ON, OFF 输入; ON

为暂停显示; OFF 为恢复显示。

2) 简单流程图的顺序依次如下: 设置输入输出; 设置堆栈; 设置初始值; 输出; 输入; 处理; 再返回到输出。输出程序在前目的要显示初始值。

3) 由功能展开看出, 本课题需要多个子程序, 子程序之间数据交换使用 HL 寄存器对通过指定的地址来

进行。所以必须分配 RAM 区域。如表 1 所示。数据区域分配在 00E0H~00E3H 地址。

4)详细流程图如图 2(a)、(b)所示。请注意调用各子程序之前,必须用 HL 寄存器对指定的 RAM 地址进行数据交换。

5)程序清单如表 2(a)、(b)所示。照表 2 程序,写

入  $\mu\text{P}-80$  教育套件,检验之后,执行输入输出部件的切换开关搬向 A 口、B 口,4 位数字显示部件显示计数加。此时打码开关  $S_0$  为 ON,显示立即停止;打码开关  $S_0$  为 OFF,显示重新开始,已不是停止显示时的数值,说明停止显示时,内部计数器仍然在进行计数加。

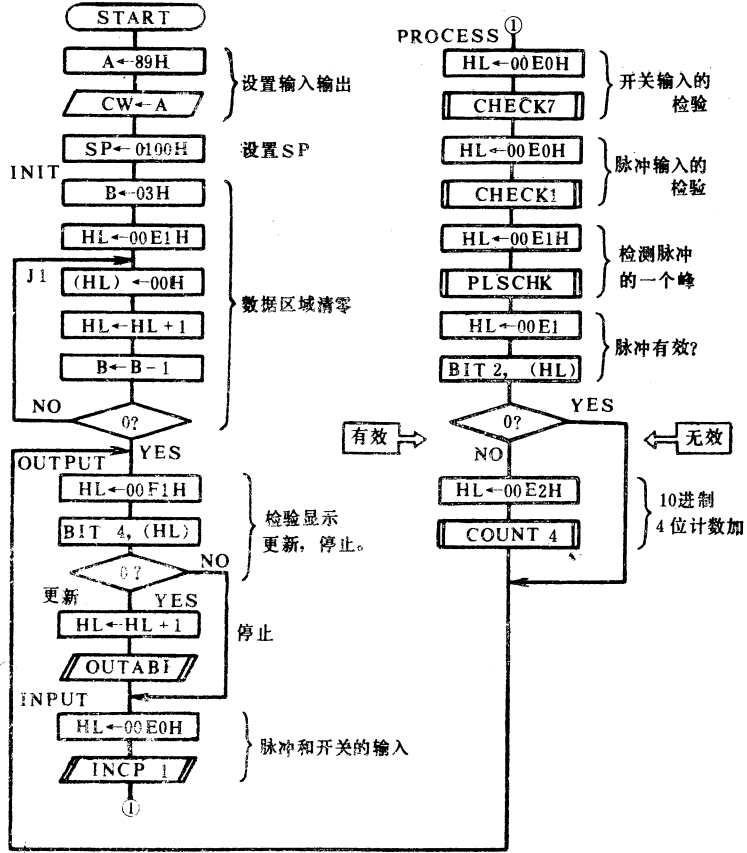


图 2(a)详细流程图

## 注意电视游戏机的干扰

南京市金陵职业大学计算中心 钱雁群

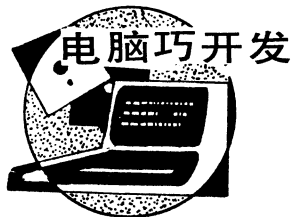
电视游戏机是电脑的一种,它也有中央处理器、运算器和存储器。当前、电视游戏机以越来越好的性能价格比冲破重重阻力大步走向家庭。对计算机工作者来说,游戏软件中的菜单技术,色彩动画音响制作技术都是值得代借鉴的。游戏机配上电脑键盘就成了学习机。这下可真是老少皆宜。

前几天一开电视,噢!怎么会有游戏画面?这免费

观看别人玩游戏也挺有趣。但其它频道的节目多少也受到干扰。后来才知道是邻居的孩子在玩游戏机。算一下直线距离,穿墙过物少说也在一、二十米。可见发射功率不算小。

市场游戏机十有八九都可以无线发射,我家的游戏机也行。但我总是用电缆和电视机连。这样虽然麻烦些,但声音图像都好一些。邻居也不会有什么意见。他人的干扰只好调整天线以减轻。

我们这些无线电爱好者,游戏机使用者和计算机工作者,很想知道电视游戏机的发射频率范围,频道宽度,发射功率及对哪些频道的电视节目干扰最大。记得美国苹果电脑公司曾选出一种很好的计算机,只是用起来有些干扰,于是不许生产。当前游戏机发射的电波是否无线管理法则。所以我们希望有关专家将这些知识,能介绍给读者。



# 简易字幕灯的制作与控制

成都西南民族学院(610041) 杨宪泽

机场、车站、城市中心及体育馆的自控字幕，造价以数十万人民币计。本文介绍 TP801A 单板机控制的简易字幕灯，其方法若加以扩充，适宜于一般单位作为较固定的广告和小型娱乐场所使用。

## 一、灯点阵与控制电路

图 1 是普通白炽灯组成的点阵，涂黑部分代表灯亮时出现的字型“庆元旦”。

图 1 的点阵为  $7 \times 7$ ，可组成一些简单字。阵列中的每一点译码(一个灯点亮)采用矩阵坐标方式，由二极管  $D_1, D_2$  构成与门，功率管 VT1 和微型直流继电器组成，电路如图 2 所示(交叉点线路连接一个灯)。

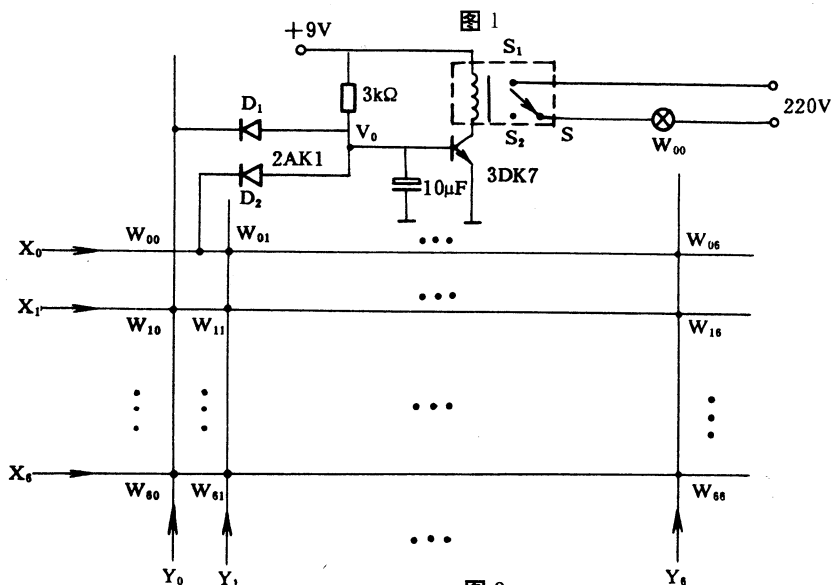
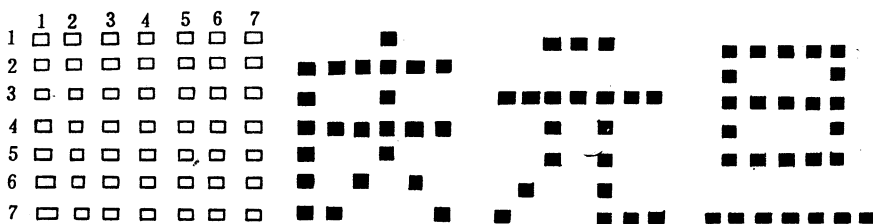


图 2

当需要点亮阵列中  $W_{00}$  灯时， $X_0$  和  $Y_0$  均为高电平约 5V，即“1”， $V_0$  为高电压使 VT1 导通，继电器线圈获足够的电流使触点 S 闭合在  $S_1$  处，从而灯  $W_{00}$  亮；反之，只要  $X_0$  和  $Y_0$  任一或全为低电平，即“0”， $V_0$  为低电压，VT1 截止，直流继电器 S 回到  $S_2$  位置，灯  $W_{00}$  熄灭。类似的所有点阵上灯由同样电路组成，原理也相

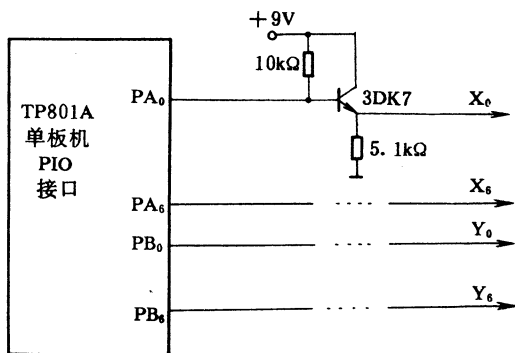


图 3

同。

## 二、微机控制与程序

灯点阵由 TP801A 单板机控制，采用单板机上现成的 PIO 接口的 PA 口和 PB 口实施，如图 3 所示。

图 3 中，为了增强 PA 口和 PB 口引脚驱动能力，每一脚加了一级射极跟随器，然后作为  $X_0 \dots X_6, Y_0 \dots$

Y<sub>6</sub> 信号。

下面给出字显示程序,以“庆”字为例,其它编程参照这种方法:

```

DISPLY
    ORG 2000H
    LD A,00H
    OUT (82H),A ;PA 口为输出方式
    OUT (83H),A ;PB 口为输出方式
    LD C,0FFH
    LD A,01H
    OUT (80H),A
    LD A,08H
    OUT (81H),A } 显示“庆”字第一排
    CALL DELAY ;延时约 1ms
    LD A,02H
    OUT (80H),A
    LD A,3FH
    OUT (81H),A } 显示“庆”字第二排
    CALL DELAY
    LD A,04H
    OUT (80H),A
    LD A,09H
    OUT (81H),A } 显示“庆”字第三排
    CALL DELAY
    LD A,08H
    OUT (80H),A
    LD A,3FH
    OUT (81H),A } 显示“庆”字第四排
    CALL DELAY
    LD A,10H
    OUT (80H),A
    LD A,09H
    OUT (81H),A } 显示“庆”字第五排
  
```

```

    CALL DELAY
    LD A,20H
    OUT (80H),A
    LD A,15H
    OUT (81H),A } 显示“庆”字第六排
    CALL DELAY
    LD A,40H
    OUT (80H),A
    LD A,23H
    OUT (81H),A } 显示“庆”字第七排
    CALL DELAY
    DEC C ;显示时间为 2 秒,未完继续
    JR NZ,DISPLY
    JP (显示“元”程序段)
    DELAY LD B,0FFH
    J1: DEC B
        JR N2,J1
    RET
  
```

上述显示程序每一字停留时间约 2 秒,利用分时原理,每一次接通点阵中一排相关点(灯)。由于每隔 7ms 程序再次返回,加之电容 C 的作用,继电器来不及断开电源,人们看到的是整个字 2 秒时间的显示。

字除了这种每两秒变换一个的显示方式外,还可以左、右、上、下移动显示。限于篇幅,本文仅介绍“庆”字向左移动显示的部分程序,其余的部分和其它方式读者不难补充和自行编程(见图 4)

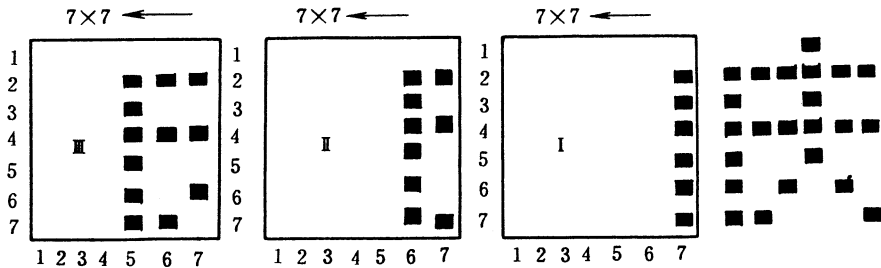


图 4

```

DISPLY1,
    ORG 2000H
    LD A,00H
    OUT (82H),A
    OUT (83H),A
    LD C,0FFH
    LD A,40H
    OUT (81H),A } 显示图 4 I 部分
    LD A,7EH
    OUT(80H),A
    CALL DELAY
    CALL DELAY
    DEC C
    JR NZ,DISPLY1 } 共显示约 1 秒钟
  
```

```

    LDC,0FFH
    LD A,20H
    OUT (81H),A
    LD A,7EH
    OUT (80H),A } 显示图 4 II 部分
    CALL DELAY
    LD A,40H
    OUT (81H),A
    LD A,4AH
    OUT (80H),A
    CALL DELAY
    DEC C
    JR NZ,DISPLY2
  
```

```

LD C, 54H
LD A, 10H
OUT (81H), A
LD A, 7EH
OUT (80H), A
CALL DELAY
LD A, 20H
OUT (81H), A
DISPLY3, LD A, 4AH
OUT (80H), A
CALL DELAY
LD A, 40H
OUT (81H), A
LD A, 2AH
OUT (80H), A

```

显示图 4 部分

```

CALL DELAY
DEC C
JR NZ, DISPLY3
:

```

这一自制 7×7 阵列字幕灯成本低于 300 元人民币(单板机除外)。若需显示任意汉字,可扩充字幕成 16×16 阵列,同时在单板机上扩充一片 PIO 芯片。作为广告使用的移动字幕,一般需两个或两个以上方阵更好,当然这要增加成本(一般 5 元人民币/点)。此外,选择灯泡瓦数注意继电器触点能承受的电流大小。

## DOS 系统下多种软件在一台微机上的共存方法

杭州浙江大学出版社(310027) 尤建志

随着计算机的应用普及和计算机的不同事务(作业)骤增,计算机所需配置的软件种数也不断增多,在同一台微机上装有二个或更多的大系统已不是什么新鲜事。从计算机的使用角度来讲,一机多用有利于提高计算机的使用效率。但如何在一台机器上有效地配置二种或更多种软件系统,则一是取决于微机硬盘的容量,二是取决于软件系统的安装方法。

举例,我社有一台 ALR Business VEISA 386/33—110 微机,硬盘内存达 106MB,其中硬盘 C、硬盘 D 和硬盘 E 均为 33MB。由于工作需要,在这台微机上要同时配置北大华光 IV 型排版系统和 中国印研所的《科印》排版系统,外接 HP I 型激光打印机。对于这么大的两个软件系统(同时需装 64×64 点阵汉字字库),如全装在硬盘 C 上,则因 DOS 系统的限制,硬盘 C 无法装入,即硬盘 C 的容量不够大,如将两个系统分装在硬盘 C 与硬盘 D 上,则在硬盘 D 上的软件系统无法运行。

现用 DOS 命令 JOIN 来解决这个问题。首先,将华光 IV 型排版系统安装在硬盘 C 的子目录 C:\HG 中,而将《科印》排版系统安装在硬盘 D 的根目录中(硬盘 D 的子目录也可)。然后,在硬盘 C 中做一个空目录 C:\KY(必须是空的,即除“.”和“..”文件外,无任何其他文件)。这时,在硬盘 C 的根目录中做一个批命令文件:

```
C>COPY CON,KY.BAT<CR>
```

```

JOIN D: C:\KY
CD KY
ctrl—Z

```

即可。以后每次只要执行 KY.BAT 将硬盘 D 挂靠到硬盘 C 的子目录 C:\KY 上,就可运行《科印》排版系统,并且所有科印排版操作都是在硬盘 D 上进行,结果文件也存在硬盘 D 上。只是一执行过 KY.BAT 后,硬盘 D 就变为硬盘 C 上的一个子目录(C:\KY)。此时若访问硬盘 D,就会得到“非法驱动器号”。而重新启动系统,则会发现硬盘 D 已与硬盘 C 相分离,C:\KY 仍为空目录。

仿效这个例子,我们可编制以下批命令文件:

```

C>COPY CON;XXXX.BAT<CR>
JOIN E:C:\XX
CD XX
ctrl—Z
C>COPY CON;YYYY.BAT<CR>
JOIN F: C:\YY
CD YY
ctrl—Z
.....

```

即可将硬盘 E 上的系统和硬盘 F 上的系统……在 C:\XX,C:\YY……子目录上运行,从而实现多种软件在一台微机上的共同存在与使用。





## 第四章 F BASIC 语言的深入理解

山东苍山机械电子化学工业局(277700) 于 春

前三章介绍了 61 条 F BASIC 的基本语句,其中有 8 条是随机手册中没有的。使用这些语句可以编制一些较复杂的运算程序和简单的游戏程序。但是,当进行较复杂的程序设计,尤其是游戏程序时,仅使用这 61 条语句往往不能达到预期目的。如,使卡通图案在规定的图形中运动,加配音乐使游戏更加生动逼真等都难以实现。可以这样说,前三章为 F BASIC 的入门篇——基础部分;第四章为 F BASIC 的深入篇——提高部分。

本章将重点介绍 F BASIC 的字符处理、出错处理和演奏语句,补充介绍完 F BASIC 的其它语句。在弄通本章语句后,你已经基本掌握了 F BASIC 语言的真谛。那么,再编制程序时,就会随心所欲,水到渠成了。

### 一、一般语句

#### 1. 收录机读写语句:

收录机读写语句共有三条,分别为

写入语句:SAVE      简写      SA.  
核对语句:LOAD?    简写      LO. ?  
读出语句:LOAD      简写      LO.

具体操作过程如下:

#### (1)写入过程:

按说明书联接好信号线和完成录音机准备后,给欲写入的程序起一个名字,如:GAME1”,然后键入 SAVE “GAME1”

启动录音机后,按回车键,这时闪烁光标消失,出现 WRITING GAME1

意思为正在写 GAME1。程序写完后,显示 OK 和光标。

#### (2)核对过程

程序写完后,一般要核对一次,检查程序是否准确地写入磁带。先将磁带倒回到开始处,键入核对语句 LOAD? “GAME1”

回车后,按下录音机 PLAY 键,光标消失,若显示 LOADING GAME1

表示正在核对 GAME1。核对完毕,出现 OK 及光标,表示程序写入正确。这时可 NEW 去 GAME1,再输入欲写入的第二个程序,重复上述操作。若出现 ? TP ERROR 表示无法从磁带中读出程序。这时,要重写一次,再核对,直到正确写入为止。

#### (3)读出过程:

若欲读出的程序名为 GAME2,而磁带中顺序写入了 GAME1、GAME2……,首先按说明书联接好信号线,键入

LOAD “GAME2”

回车后,按下录音机的放音键 PLAY,若读到程序,屏

幕显示:

SKIP GAME1

LOADING GAME2

当显示 OK 和出现光标时,表示读出完毕,可列出程序,检查读出的程序是否正确。若显示

? TP ERROR

表示无法从磁带中读出程序,必须再读,直到正确地读出程序。

#### 2. 键盘输入语句(LINPUT)

第二章曾介绍了键盘输入语句 INPUT,它只能一次输入一个数据,而 LINPUT 语句可以输入一整行的数据,其中包括各种符号。

LINPUT    简写    LIN.

LINPUT 语句的功能是从键盘上接收整行字符的输入,并把它们当做一个字符串赋给一个串变量。语句格式为

LIN. [提示符];字符串变量

例 1

```
10 LIN. “A $ =”;A $  
20 P. “Shu Ru De” A $  
30 G. 10
```

Run 显示

A \$ = 键入 ABC[]123FGE 回车后,显示:

Shu Ru De A \$ =ABC[]123FGE

注意——输入的字符个数,包括提示符不能超过 31 个。

#### 3. 返回 BG 画面语句(SYSTEM)

SYSTEM    简写    S.

SYSTEM 语句的功能是在对话进入 BASIC 状态(BS 状态)时,返回到 BG GRAPHIC 画面以进行背景图案的设计。

注意——在直调 BASIC 状态时该语句的功能是打开驱动器系统。由于一般不配备磁盘驱动器,所以该语句在直调 BASIC 时无效。

#### 4. 注释语句(REM)

REM 语句的功能是对程序进行说明或注释,以使程序容易理解,便于阅读。REM 语句可以编在程序中,注释的内容可以是任何文字或字符,甚至可以是 BASIC 的语句。程序运行时,并不执行 REM 语句。因此 REM 语句为非执行语句。

注意——REM 语句后的注释或叙述不能超过 255 个字符。

#### 5. 功能键控制语句(KEY,KEYLIST)

(1)设定功能键语句(KEY)

F BASIC 系统提供了 8 个功能键 F1~F8,这 8 个功能键可由用户任意设定以代表某种意义。如在编制程序前,将本程序常用的语句定义给功能键,在以后的编程当中,就可以直接使用,以提高程序的输入速度。

KEY 简写 K.

语句格式为

KEY N,“字符串”

式中 N 的功能键 F1~F8 的号码。

注意——每个功能键所代表的字符数不能超过 15 个。

(2)显示功能键定义语句(KEYLIST)

KEYLIST 简写 K.L.

其功能是将功能键 F1~F8 的定义显示于屏幕上。

例 2.

10 P. “Yuan Ding Yi:”

20 K. L.

30 K. 1, “DEF SPRITE”

40 K. 2, “POSITION”

50 K. 3, “CLS;SPRITE ON”

60 K. 4, “DEF MOVE C”

70 P. “Xin Ding Yi:”

80 K. L.

90 E.

Run

Yuan Ding Yi;

F1 GAME 0 (M)

F2 GAME 1 (M)

F3 GAME 2,1(M)

F4 GAME 3(M)

F5 SPRITE

F6 PRINT

F7 LIST

F8 RUN(M)

Xing Ding Yi

F1 DEF SPRITE

F2 POSITION

F3 CLS;SPRITE ON

F4 DEF MOVE C

F5 SPRITE

F6 RPINT

F7 LIST

F8 RUN(M)

由于 F BASIC 系统已将 F1~F4 四个功能键定义为调用 GAME0~3 四个游戏程序,因而,在直调 BASIC 状态,可直接按 F1~F4 任一个功能键调入游戏程序并运行。但在进行程序设计时,这四个功能键毫无用处。因此,在编程前将 F1~F4 重新定义为编程中经常用到的语句或字符,以提高编程输入速度。

6. 测试空余单元语句(FRE)

FRE 语句的功能是显示内存单元的空余数量,以

防止输入的程序较长时,内存不能完全容下而丢失了数据。尤其在从磁带中读出程序时,若不了解内存空间。空余数量而盲目读出,就容易造成数据丢失。

语句格式为

PRINT FRE

7. 键盘扫描语句(INKEY \$)

INKEY \$ 简写 INK.

语句功能是对键盘进行扫描,等待读取键入的字符。如果在该语句执行的瞬间,用户按下某一键,则该键的字符就会赋给 INKEY \$。这样,程序的运行就可根据 INKEY \$ 的取值决定转向。语句格式为

INK. (0)

例 3.

10 A \$ =INK. (0)

20 P. A \$

30 IF A \$ = “Z” T. P. “OK”

40 IF A \$ = “A” T. P. “NOT”

50 IF A \$ = “X” T. E.

60 G. 10

Run

按 Z 键则打印 OK;按 A 键则打印 NOT;按 X 键则结束。

8. 进制转换语句

(1)十进制数转换十六进制语句(HEX \$)

HEX \$ 简写 H.

HEX \$ 语句的功能是将十进制数转换为十六进制数。语句格式为

H. (X) X 取值: -32768~32767

例 4.

10 F. I=100 TO 200

20 P. I. “D=&H” H. (I),

30 N.

(2)十六进制数转换十进制

十六进制数转换为十进制,随机手册中给出了 VAL 语句。实际上,使用 PRINT 语句就可以把输入的十六进制数转换为十进制数输出。(有关 VAL 语句的功能,下文介绍)

如:P.&HFF 回车后打印 255

P.&H7FFF 打印 32767

例 5.

10 F. A=&HAF TO &HFF

20 P. A,

30 N.

Run 则打印 175~255

二、画面控制语句

1. 测试卡通运动方向语句(VCT)

在程序运行中,VCT 语句可以测试出卡通图案的运动方向,根据卡通的运动方向决定程序运行的转向。VCT 语句在随机手册中没有介绍,请读者自己添上。

语句格式为

VCT(n) n 为被测试卡通的编号

### 例 6.

```
10 SP. O. : CLS
20 DE. M. (0)=SP. (0,RND(8),1,250,0,0)
30 POS. 0,120,120;M. 0
40 LOC. 0,0;P. VCT(0);PAU. 100
50 G. 20
```

Run

卡通图案的运动方向随机产生,在座标为(0,0)处不断打印出 SP 的运动方向 0~8。

### 2. 测试 BG 画面的数值语句(SCR \$)

SCR \$ 语句的功能是求出 BG GRAPHIC 画面中所显示的文字或图案的数值。语句格式为:

SCR \$(X,Y,n)

式中 X,Y 为测试画面的座标。n 为该处的配色号码 0 或 1,也可以省略。

注意——在随机手册中给出了 SCR \$ 的简写为 SC., 这只有在对话 BASIC 状态才有效,而在直调 BASIC 中,SC. 是 SCREEN 的简写。所以,在直调 BASIC 中,SCR \$ 没有简写,请读者更正,以免输入程序出错。

SCR \$ 语句可以求出指定点的 ASCII 码。有了这条语句,就可以判断卡通图案在任一位置时的背景图案是什么,以便于转向控制。

### 例 7.

```
10 F. I=0 TO 100
20 P. CH. (200); CH. (208);
   CH. (215);CH. (221);CH. (232);
30 N.
40 F. K=0 TO 20 ST. 2
50 P. ASC(SCR $(K,K))
60 N.
```

Run

打印 11 个点的图案的 ASCII 码。

### 三、跟踪语句(TRON、TROFF)

TRON 简写 TRO.

TRON 语句的功能是对程序的运行位置进行跟踪。在跟踪状态下,程序在执行中会不断显示正在执行着的语句的行号。对程序的运行进行跟踪的目的是为了对程序进行调试和查错或了解程序执行的中间结果。

TROFF 简写 TROFF.

TROFF 语句的功能是关闭跟踪。

TRON、TROFF 两个语句配合使用,可任意插在程序的任意位置,打开或关闭跟踪,以了解程序的运行情况。

### 例 8.

```
10 REM“TRON, TROFF”
20 TRON
30 P. “TRON”;
40 F. I=0 TO 100
50 TROFF;P. “TROFF”
```

60 P. I“\*”I“=”I\*I,

70 N.

80 TRON

90 P. “Yan Shi Jie Shu”

100 E.

Run 显示

# 30 TRON # 40 # 50 TROFF

0 \* 0 = 0 TROFF

1 \* 1 = 1 TROFF

∴ ∴ ∴  
99 \* 99 = 9801

100 \* 100 = 10000 # 90

Yan Shi Jie Shu

# 100

### 四、字符串处理语句

#### 1. VAL

VAL 语句的主要功能是将字符串型的数据转换成十进制数。即 VAL 具有去掉字符、引号、\$ 符号的功能。经 VAL 语句处理后的数据可以参加数值运算。随机手册介绍 VAL 语句的功能是“将十六进制数转换成十进制数”,语句功能介绍不完全。本章在进制转换语句的介绍中已经讲了可用打印语句 PRINT 把十六进制数转换为十进制数。因此,VAL 语句的主要功能是将串型数据转换为十进制数。

语句格式为

VAL(字符串)

#### 例 9.

```
10 REM “VAL”
20 I. “Shu Ru Zi Fu Chuan”;A $
30 A = VAL(A $)
40 P. “A $ =”A $, “VAL(”A $ “)=”A,
50 G. 20
```

Run 显示

Shu Ru Zi Fu Chuan? 键入 34ABC

A \$ = 34ABC VAL(34ABC) = 34

Shu Ru Zi Fu Chuan? 键入 7AB38

A \$ = 7AB38 VAL(7AB38) = 7

…… 键入 GOOD

A \$ = GOOD VAL(GOOD) = 0

…… 键入 AB3214

A \$ = AB3214 VAL(AB3214) = 0

∴

通过演示,读者可以发现,VAL 语句就是保留字符串开头的十进制数,后面的字母等一概略去。若第一个字符不是数字则其值为 0。

#### 2. STR \$

STR \$ 语句的功能是将十进制数转换成相应的字符串。随机手册中介绍 STR \$ 语句的功能是“将十进制数转换成十六进制数的字符串”的叙述是错误的;在该语句的说明中,说该指令用法与 VAL 相反”也是错误的。请读者更正。(下转 34 页)



# TH3070 打印机常见故障分析与检修(上)

国务院办公厅秘书局技术处(100017) 刘立华

TH3070 点阵式打印机是一种 24 针可打印汉字的打印机,目前被广泛采用,是一种优选机型,现针对此类型机的常见故障进行分析与实例检修。

表 1 功能代码表

| 代 码 符                                                            | 十进制表 示       | 十六进制表示       | 功 能          |
|------------------------------------------------------------------|--------------|--------------|--------------|
| HT                                                               | 09           | 09           | 横向制表命令       |
| LF                                                               | 10           | 0A           | 新行(换行+回车)    |
| VT                                                               | 11           | 0B           | 纵向制表命令       |
| FF                                                               | 12           | 0C           | 换页命令         |
| CR                                                               | 13           | 0D           | 回车命令         |
| SO                                                               | 14           | 0E           | 横向扩展一倍命令     |
| SI                                                               | 15           | 0F           | 解除横向扩展       |
| DC1                                                              | 17           | 11           | 选择使打印机联机     |
| DC3                                                              | 19           | 13           | 解除选择         |
| CAN                                                              | 24           | 18           | 清除当前行缓冲器中的数据 |
| ESC1                                                             | 2749         | 1B31         | 水平表位设定       |
| ESC2                                                             | 2750         | 1B32         | 水平表位清除       |
| ESC5n <sub>1</sub> n <sub>2</sub> n <sub>3</sub>                 | 2753<br>...  | 1B35<br>...  | 垂直表位设定       |
| ESC6                                                             | 2754         | 1B36         | 垂直表位清除       |
| ESC A                                                            | 27 65        | 1B 41        | 链式走纸模式设定     |
| ESC C                                                            | 27 67        | 1B 43        | 滚式走纸模式设定     |
| ESC E                                                            | 27 69        | 1B 45        | ELITE 字型设定   |
| ESC Hn <sub>1</sub> n <sub>2</sub> n <sub>3</sub>                | 27 72<br>... | 1B 48<br>... | 页长设定         |
| ESC In <sub>1</sub> n <sub>2</sub>                               | 27 73<br>... | 1B 49<br>... | 图象打印         |
| ESC N                                                            | 27 78        | 1B 4E        | PICK 字型设定    |
| ESC Un <sub>1</sub> n <sub>2</sub>                               | 27 85<br>... | 1B 55<br>... | 换行单位设定       |
| ESC Vn <sub>1</sub> n <sub>2</sub> n <sub>3</sub> n <sub>4</sub> | 27 86<br>... | 1B 56<br>... | 反向走纸         |
| ESC Wn <sub>1</sub> n <sub>2</sub> n <sub>3</sub> n <sub>4</sub> | 27 87<br>... | 1B 57<br>... | 正向走纸         |
| ESC Zn <sub>1</sub> n <sub>2</sub>                               | 27 90<br>... | 1B5A<br>...  | 空列设定         |
| ESC <                                                            | 27 60        | 1B3C         | 双向打印         |
| ESC >                                                            | 27 62        | 1B3E         | 单向打印         |

## 一、功能代码表

TH3070 打印机所能执行的控制功能简表如表 1 所示。

## 二、TH3070 点阵式打印机常见故障分析与检修

不论是计算机还是其外部设备,在机械结构方面发生故障的概率是很小的,这种故障即便发生亦能直观地看出,在条件允许的情况下,也比较好排除。而电路部分的故障,就没有机械方面那样简单直观,由于这种故障隐蔽在整个电路的某一部分,要想找到这方面的故障并排除掉它不仅要求维修人员具备一定的电路知识水平,而且还要求维修人员对其所维护的设备有所了解,如该设备的工作原理,所使用的器件及各器件的功能等。

下面,我们就 TH3070 的一些常见故障,做一些简单说明,并给出故障检查的流程图。

### 【故障实例一】走纸异常故障

在打印正常的情况下,打印机不走纸,致使打印字迹成一条黑印,这种故障,除了使用者在拆装机器时,把打印辊装反,以及在安装时未能正确地插入 PJ12 插座外,一般都是出在电路部分的器件损坏上,这种损坏所导致的结果就是走纸电机缺相,走纸电机和字车电机一样,是四相步进电机。四相步进电机缺一相或几相,必然会使“步进”(换相)失去作用,从宏观上看,就是打印机不走纸。图 1 为这一故障的检修逻辑图。

在检修流程图中,检查某一器件是否正常,可以用示波器也可以用逻辑笔检查该器件的输入与输出状态,看是否与逻辑关系符合,即可判定其好坏。在走纸故障中,一般最容易坏的部分,除了机电本身外,就是 74LS06(IC44)。另外,在更换元器件时,如无吸锡工具,最好用剪刀先将该芯片的所有引脚剪断,再一个脚一个脚的焊下,否则容易损坏电路板。

### 【故障实例二】回车撞“墙”故障

这种现象表现为当打印机加电后字车回到左界,仍然不停止、继续向左墙板撞击,引起电机发出怪叫声,检查这种故障的流程图如图 2 所示:

在流程图中“检查 2764、8155”这一框,是要先检查 2764,因为打印机的字库型和普通型所用的控制程序是不一样的,二者不能通用,在有两台以上机器的单位,可通过交换 2764 芯片来检查 2764 的好坏,如果确实证明了 2764 芯片无问题,而 8155(IC27)的 33 脚有信号变化则必然是 8155 芯片损坏,换成好的,即可排除故障。不过这种故障多是发生在开关位置的左移动上,把左界开关适当换成好的,向右移动一下,这种故障即可排除掉了。

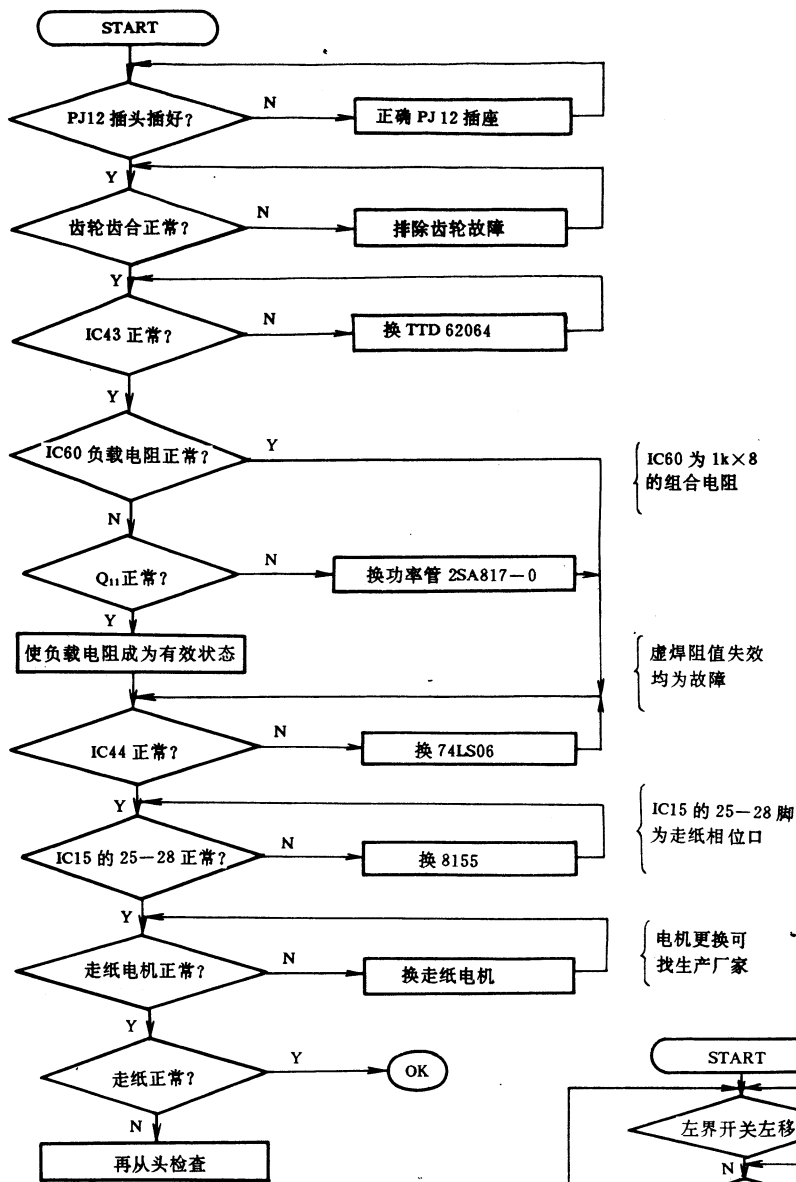


图 1

【故障实例三】回车后不自检故障

这种故障的表现是打印机通电后字车回到左界，但在按下“测试”(TEST)键时，打印机无打印动作，且在“联机”键按上时变无联机灯亮，这种故障，多是在打印机控制程序 2764 装错或 2764 本身出了故障，若是后者可由厂家对三片 2764 重新写一次即可，这种故障的检修逻辑图如图 3 所示。

【故障实例四】印字漏点故障

打印漏点，是打印机最常发生的故障，其表现形式为打印的字形中缺点(一点或多点，它虽然不影响打印机的使用，但却使打印出的字形失真，影响美观，这种

故障，或者是由于电路内部的器件失效而造成，或者是由于打印头上线圈失效，针断、针短造成，或是由这两方面的原因造成的，下面是这种故障的检修逻辑图，如图 4 所示。

在图中，虚框中“确定不出针的位置”，实际上可由打印时漏点的位置确定下来，也可拆下打印头从针头上数出来。打印头上的打印针是这样排列的；打开打印头的后盖，从中线分两半，右边一半自上而下分别为第 2 针、第 4 针……第 24 针，左边一半分别为第 1 针至第 23 针，如图 5 所示。

打印针直径 0.2mm，  
点中心距：纵 = 横 = 0.141mm  
点覆盖率：20%  
两列针跨距：8 列 = 1/20 寸

图 5 左图中的数字，代表的是衔铁排列的顺序(衔铁下面是线

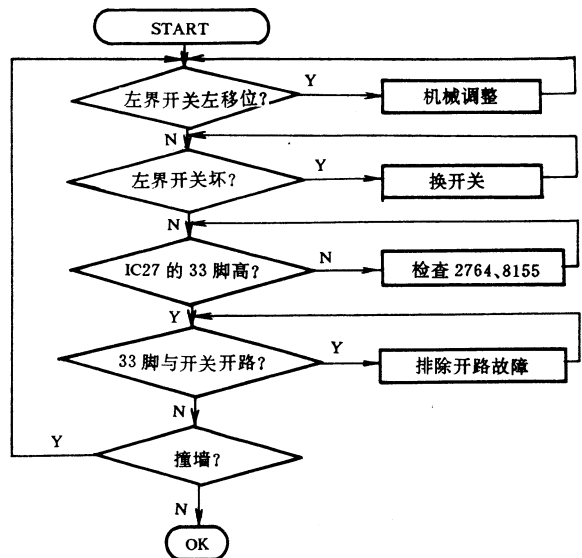


图 2

圈),它与右图的打印针的数字是一一对应的(打印针与衔铁线实际上是焊接在一起的),由此,也可以看出,打印机在打印时,偶数针先打,奇数针再到位补打,使字形成为完整的一列。

在确定打印头无问题,进行电路检查时,实际上就是检查 TD62064 芯片和 74LS374 芯片,如第 13 坏,检查 IC56 和 IC64 即可找出。这二个芯片必有一个已坏了。

若是打印机针变短或者已断,则需要换针,

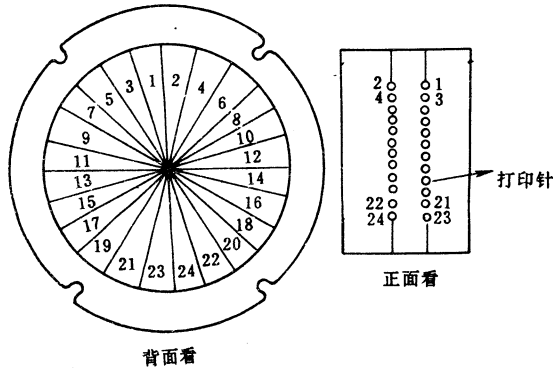


图 5

换针是一项有难度的工作,除了具备一定的技巧和熟练程度外,还需要有耐心,初换者往往由于开始换不好而失去耐心,乱捅硬压,从而使新换的针还未装进去就被再次弄断,这是换针时一定要避免的。

(下转 48 页)

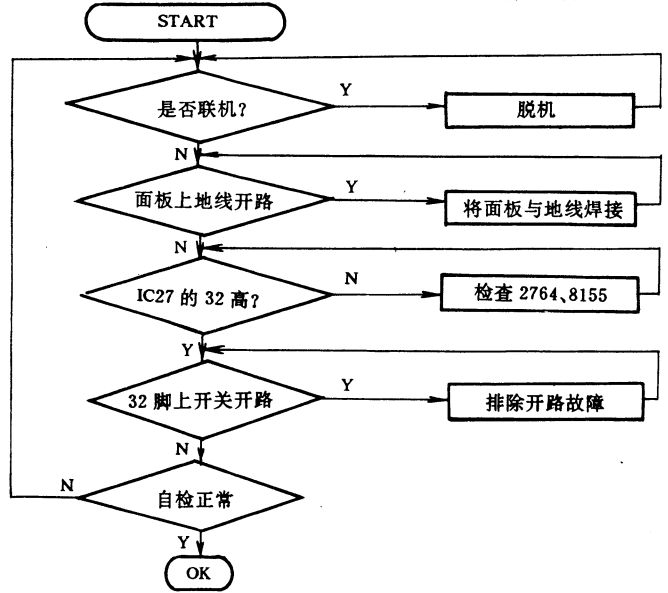


图 3

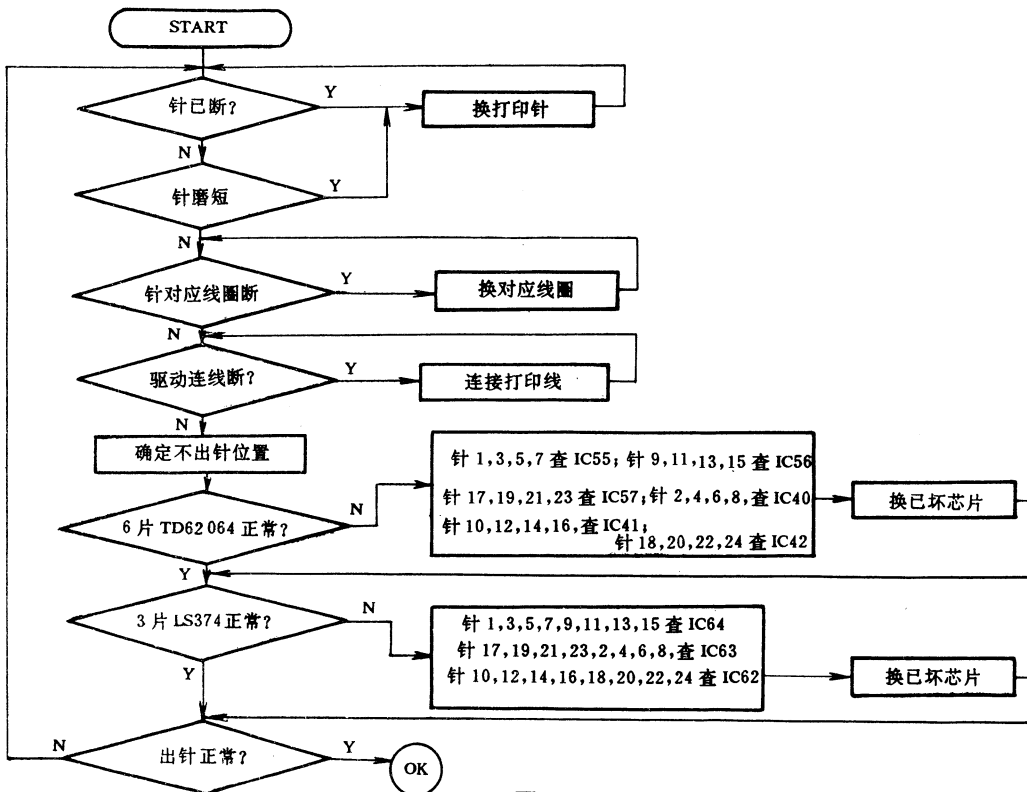
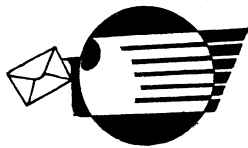


图 4



## 读者联谊

# 普及型 PC 个人用户软件交流 联谊活动问题解答(五)

北京中国农科院计算中心(100081) 王路敬

### 13. 怎样判别 PC 机键盘的好坏?

键盘是向微机提供指令和信息的必备工具之一,它是微机的一个重要输入部件。在 IBM PC 机上,键盘由一条永久性连接电缆,连到系统部件后部的 DIN 插头上,它是一条带屏蔽的四线电缆包括 +5 Vdc、地和两条双向信号线。

PC 机的标准键盘是 83 个键,现在不少 PC 兼容机又增加了一些键,如有 101 个键的等等,但它们的工作原理是一样的,操作方法也基本没有多大区别。键盘作为微机的输入设备,它与其他的外部设备如显示器、打印机、磁盘驱动器等一样,都需要硬件接口和软件接口的支持才能实现真正的功能操作。

PC 机的键盘可以分为三个区:功能键区、打字机键盘区、数字键盘区。键盘上键的功能和作用是由软件定义的,所以在不同的工作环境下,各键的作用就可能不同。键盘上的键好坏或工作正常与否可以利用随机的诊断程序进行检测。只要运行一次诊断程序,就可以对键盘上的每一个键进行检测,帮助查看键盘有没有问题,这种诊断比较方便、简单、易掌握。

使用键盘操作时,切记不要用力过猛,轻轻击打即可。保持键盘清洁、干净。防止灰尘落入键盘。

### 14. 微机的显示系统包括哪两部分?显示功能有哪两大类?

显示器是微机信息输出的重要设备,也是实现人机对话的主要工具,它既可以显示键盘输入的命令和数据,又可将结果数据变成字符或图形显示出来。它和键盘结合在一起可以方便地进行人机对话。

微机的显示系统包括 CRT 显示器和 CRT 显示适配器两部分。CRT 显示器的屏幕显示原理采用光栅扫描方式,即是在水平偏转和垂直偏转的同步信号控制下,电子束在屏面上从左到右,从上到下有规律地扫描,在屏面扫描期间,电子束强度受视频信号的影响而变化,在水平回扫和垂直回扫期间,由消隐电路抑制电子束,使得回扫线不会在屏幕上显示出来。CRT 显示适配器又称显示卡,是系统和显示器的接口,它的主要功能是控制光栅或显示器。

从显示功能上看可分为两大类:一类是字符显示,另一类是图形显示。虽然字符显示是用点图的方法来显示的,但在结构上是不同的。对字符显示方式,字符

点图来自于字符发生器;对图形显示方式,显示缓存的数据位对应着显示屏幕的相应象点,象点的颜色取决于数据位的状态。因此,如果要求显示适配器能支持图形方式,就必须有足够大的显示缓存,以至能容纳屏幕上所有的显示象点。IBM PC 单色显示适配器,显示缓存容量小,只能支持字符显示方式,而 IBM 彩色显示适配器显示缓存达 16K,它不但支持 80×25 字符方式显示,还能支持 320×200 和 640×200 的图形方式。

### 15. 单色显示器主要性能指标应了解哪些?显示卡的主要部件有哪些?

从使用角度对单色显示器主要性能指标应了解:

- (1) 满屏可显示 25 行×80 列字符。
- (2) 每个字符块大小为 9×14 点阵,字符大小由 7×9 点阵组成。
- (3) 可显示 256 种 ASCII 编码的字符点阵。
- (4) 每个输出字符均有各自的显示属性,如加亮,反视频(白底黑字),下横线等。

以 IBM 单色显示卡为例,主要部件包括以下几部分:

- (1) MC 6854 CRT 控制器芯片,它是整个控制逻辑的核心。
- (2) 字符缓存及属性缓存各 2K 字节。
- (3) 字符发生器。它是一个容量为 8KB 的只读存储器,保存 256 个不同的字符点阵模型。
- (4) 移位寄存器。
- (5) 时钟发生器。包括两个基本时钟,一个象点时钟 DOTCLK,另一个是字符时钟 CCLK。
- (6) CRT 控制器。
- (7) CRT 状态器,反映当前显示器的状态。
- (8) 视频信号合成电路。它是适配器中视频信号处理逻辑部件,也是与 CRT 显示器的接口界面。
- (9) 地址译码电路。

显示控制器的工作过程大致如下:CPU 8088 在显示器水平和垂直回扫期间,把欲显示的字符及其属性送入字符缓冲器和属性存储器;MC 6845 CRT 控制芯片以每秒 50 帧的频率,一面产生行同步、帧同步等信号送到视频信号处理逻辑,一面产生地址去读出缓存器中的字符码及其属性;以缓冲器中读出的字符代码和 MC 6845 CRT 控制芯片提供的扫描线序号为地

址,读出字符发生器中的字型信息,字符的属性则进行一定的译码处理;字型信息读出后送入移位寄存器,然后逐位移出到视频控制器逻辑中,与有关的属性光标等组合处理后,形成显示器所需要的视频输出信号和亮度信号,水平扫描信号直接驱动单色显示器。

#### 16. PC 机系统的彩色图形显示器有哪两种基本工作模式?

PC 机的彩色图形显示器在系统中有两种使用方式,它既可以代替单色显示器作为控制台显示器使用,也可以与单色显示器同时使用。它具有两种基本工作模式:一种是字母数字模式也称 A/N 模式。在这种工作模式下,高分辨率:每帧 25 行,每行 80 个字符;低分辨率:每帧 25 行,每行 40 个字符。

上述两种分辨率既可支持单色字符显示方式,又可支持彩色字符显示方式,当选择单色(黑白)显示方式时,显示字符具有闪烁、加亮、反视频的属性,而选择彩色显示方式时,每个字符可以选用 16 种背景(底色)和前景(显示色)色彩,而且仍可以使用闪烁属性。字符方式下的工作过程:CPU 在显示器水平和垂直回扫期间,把欲显示的字符码及其属性码分别送入显示缓存的偶地址单元和奇地址单元;MC 6845 CRTIC 片在字符字节时钟同步下,一面产生 15.75kHz 行同步和 50Hz 的帧同步等信号送到彩色显示器,一面产生刷新地址码去读显示缓存中的字符和属性;以显示缓存中读出的字符代码和 MC 6845 提供的扫描线序号为地址,读出字符发生器中的字符点阵信息,字符属性则送颜色编码器,字符点阵信息读出后送入字符像素并/串转换器,然后逐位移出到颜色编码器中,与有关的属性、光标等配合,形成显示器所需要的视频三元色输出

信号。

另一种是图形显示模式,也称 APA 模式。在这种工作模式下,显示器屏幕上的每个象点均可由程序控制其亮度或颜色,因而能显示出质量较好的图形,APA 模式又可分为三种不同的分辨率:高分辨率模式:每帧 200 线,每线 640 点,每点只能取黑白两种颜色;中分辨率模式:每帧 200 线,每线 320 点,每点可以有 4 种不同的颜色;低分辨率模式:每帧 100 线,每线 160 点,而每点则有 16 种不同的颜色。目前在 BIOS 基本输入/输出系统中的驱动程序不能支持低分辨率显示模式,需要时应由应用软件人员自行开发。

彩色图形显示控制器逻辑结构与单色显示控制器类似,它也采用 6845 CRTIC 芯片作为控制器的核心,一方面产生显示缓冲器的地址码以读出其中的信息,另一方面同步地给出水平和垂直扫描信号去控制 CRT 显示器。图形方式下的工作过程:CPU 在显示器水平和垂直回扫期间,把欲显示的图形象素送入显示缓存,其中偶数扫描线象素放在显示缓存的前 8KB,奇数扫描线象素放在后 8KB;MC 6845 CRTIC 芯片在象素字节时钟 CCLK 同步下,一面产生 15.75kHz 的行同步,50Hz 的帧同步送彩色显示器。另一方面产生刷新地址码,每次先后读出显示缓存中偶数和奇数地址单元的两个象素字节,送入图形锁存器;这两个象素字节同时送入并/串移位器,然后每次移出二位,送到颜色编码器中,若是  $320 \times 200$  彩色图形方式,移出的二位与颜色选择寄存器的第 4、5 位一起经颜色编码器产生视频彩色信号,若是  $640 \times 200$  图形方式,颜色编码器输出的视频三元色信号仅是黑色或白色。



(上接 46 页)

换针时,首先要准备好工具,主要是十字槽螺丝刀、镊子及小锉刀,换针开始,先拆下后盖,把打印头朝下,尾部朝上,用镊子取出所换的针,然后把要换的新针在对应位插下,打印针自尾部到探出头,共要经过四道孔槽,上面的两个孔,起着打印针的定位作用,接着,针顺势而下,穿过导向块,导向块是一个尼龙片,在它上面有两个弧形孔,奇数针(12)个和偶数针(12)个分别穿过其中的一个孔,针在穿过这个孔时,由于孔不象前两个孔是一一对应的,故可用镊子先把针捋几下,使该针基本上处于该处的位置,针进入最后一个孔,既从打印头冒出来,是最难的一步,可用镊子在第三、第四孔之间再捋几下,或者是边捋边插,一直到针穿出。

当针换过后,新针总是要比其它针长一点,可用锉刀把长出的部分锉掉,使该针和其它针平齐,然后再装上后盖,这样,打印针换针的工作就算是完成了。

在换针时,还有几点要注意的,一是切忌把拆盖后

的打印头朝上放置,以免其它针掉下来,若要头朝上时,一定用拇指把所有衔接铁按住。再就是若在换针时,其它的针已经掉下来,则可先将掉下来的针恢复上去,因为这样的针,再插上去比取下来的针撤上去要容易些,另外,如果需要把所有的针都拿下来,一定不要搞乱了它们的顺序。因为打印头上靠近上下两端的针比位置在中间的针要稍长些。把针的位置搞乱,这个打印头实际上也就没有什么用了。

打印针更换完毕再装上打印机时,也要注意一点,打印头和打印辊之间的距离是有规定的,在无屏蔽罩(MASK)的情况下是 0.35mm,在有屏蔽罩的情况下,距离是 0.25mm。在有屏蔽罩的情况下上头时,必须先要松开屏蔽罩的螺丝,使其紧贴打印辊上,然后装上头,用 0.5mm 的塞尺(厚薄规)塞在头与屏蔽罩之间,过紧或过松都不行,手感觉得适度后,把打印头上紧,再把屏蔽罩上好。头与辊之间的距离是否准确,关系着打印头的寿命。

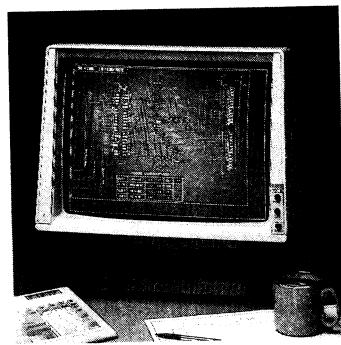




電子

ISSN 1000-1017

與電腦



ELECTRONICS AND COMPUTERS

一九九二年

总期第87期

# 電子與電腦

## 目 录

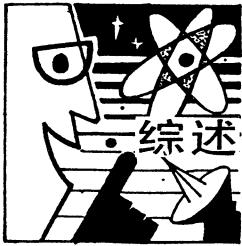
- 综述 •
  - 美国当今图像处理软件…………… 吕问黎(2)
- PC 用户 •
  - dBASE III 和 FoxBASE+ 环境下实用多窗口弹出式菜单的实现方法…………… 张志远(3)
  - CCDOS 2.1 特殊显示功能的妙用…………… 李林枫(4)
  - 在 DOS 中使用功能菜单的方法 …… 金林樵(5)
  - PKZIP 压缩软件包的技术背景及使用…………… 夏伟文(6)
  - 如何使《UDOSV1.0》支持 AT101/102 键盘…………… 杨 旭(9)
  - 恢复硬盘启动功能的方法…………… 王晓红(9)
- 学习机之友 •
  - 无须驱动器的电子打字机程序 …… 刘庆丰(11)
  - TOOL-KIT 使用详解…………… 姜 宏(13)
  - 利用 BOOT 程序对出租软件进行计次使用…………… 任晓方(17)
  - 中华学习机汉字双页显示…………… 蒋建一(18)
  - 高速排序程序的使用技巧…………… 张世栋(19)
  - 汉字编码打印程序…………… 李 铁(21)
  - 对《CEC-I 键控光标作图程序》的改进…………… 李庆岱(22)
- 语言讲座 •

- 第六讲 简单程序和分支程序设计…………… 朱国江(23)
- 初级程序员级软件水平考试辅导 •
  - BASIC 基础…………… 李志刚(29)
- 学用单片机 •
  - 带 A/D 的单片机最小应用系统…………… 张培仁 刘振安(31)
- 学装微电脑 •
  - 7 段数码管读数装置…………… 易齐干(34)
- 电脑巧开发 •
  - 自带 EPROM 类单片机简易加密编程…………… 庄 凯(38)
  - 一种简单的键盘接口电路…………… 罗许建(39)
- 电脑游戏机 •
  - 第四章 F BASIC 语言的深入理解…………… 于 春(40)
- 维修经验谈 •
  - 维修打印头应注意的几个问题 …… 蔡世清(44)
  - TH3070 点阵式打印机常见故障分析与检修(下)…………… 刘立华(44)
- 读者联谊 •
  - 普极型 PC 个人用户软件交流联谊活动问题解答(六)…………… 王路敬(48)

封一:监视器 封二:资料 封三:MP-1 原理图 封四:广告

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



# 美国当今图像处理软件

吕问黎

在美国丰富的软件种类之中,图像软件是一个相当引人瞩目的领域。这类软件主要有两大用途:图像显示和数据分析。前者主要是把书面的字句转化为视觉信号,经常在展销会、研讨会一类场所大显身手。后者则主要参与决策。在决策者和分析的信息之间建立起高速的信息交流,这一点正是决策者们从书面印刷材料中难以得到的;这类软件对于经常改变但主要输出形式又基本不变的各种数据(如股票市场信息)的处理尤为得心应手。此外,还有一部分图像软件专注于图像本身,不涉及其他信息,比如动画制作、宣传画绘制软件等。

现在美国市场上的图像软件的第一大特点是用户使用非常方便。用户自己无需有很高的绘画才能就可以制作出精美图像来。软件通过各种途径减少用户的麻烦,常设有大容量的图形库,提供现成的图像,以免用户自制之苦。比如 Computer Support Corporation 生产的 Arts Graphics Composer 软件存有 3700 个小型图像并有 15 种变化形式,而它的扩展版本的图像数上升到 5000 个。T/Maker Company 生产的 Clickart Images 是一个专门的图库软件,内有 1250 个如食品、电脑、旗一类的日常用品,140 多个动物、卡通形象和 240 万个万圣节、复活节、情人节、舞会等场面,用户使用起来选择余地很大,绘图非常方便。

软件设计者对软件的操作程序也费尽心机使其尽量简化,如 Solution 国际公司出产的 Curator 软件是一个图像管理软件,用它寻找画面的方法就有五种,首先可以通过名称直接调入;如用户记不清全名而只输入名称的一部分也可调入相应的画面;还可以将所有画面名称列于屏幕上供选择;假如用户对画的名称一点儿也不记得了,软件会在屏幕上绘出多幅图像略图供用户选择;若这样还不行,用户也可逐一浏览图像。这个软件帮助用户整理、排列、打印、复制图像,它本身即是应方便使用这一要求而诞生的。

美国当今图像软件的另一大特点是作品质量上精益求精,技术上不惜花费大功夫。Autodesk 公司生产的 Autodesk Animator 是一个动画制作软件。它在动画显示时不仅运用了传统的翻页显示方法,而且采用了多种光学技术提高图形的三维立体性,还采用 Polymorphic(多形体)技术来直接使两个完全不同的图像相连接,而无需作者绘制中间的过渡画面。它还对字符的颜色等作了处理,还使各种字符也可以在画面上活动,产

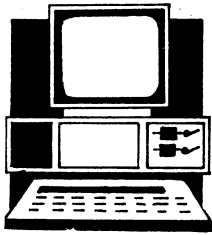
生极佳的动画效果。软件还提供 20 多种工具来描绘作品细节,用户可以改变图像色调和色散程度,使图像变得不透明、柔和、有光泽或模糊,产生各种特殊显像效果。

MicroMind 公司出产的 MicroMind Director 是一个制作文字、图像、动画和小影片的软件,功能非常强大,制作成品的效果也令人赞叹不已。软件分两大部分,前一部分(Overview)控制着图像放映的先后次序和时间(最高达每秒 30 幅);后一部分(Studio)则是一个大工作室,有一套复杂的画面插入系统、动画操作系统和控制音响的程序。在细节上更是体现了软件的质量和制成品的质量,用户不仅可以对文字、音响效果等做微妙的控制,而且在画面与画面交替时,还可以采用电影中镜头交替的各种形式,如下一个镜头淡入,或者新画面逐渐推入将旧画面遮蔽,或者整个画面同时转化等。这些特殊的细节处理使人看了软件成品后觉得简直就是职业摄像师的作品一样。

图像软件的第三大特点是对硬件的开发达到了极高的程度。前述的 Autodesk Animator 软件可以容许同时有 4000 个物体参与动画过程,这对硬件的速度可谓利用到了极点。Deneba 软件公司的产品 Canvas 可以显示 1/1000 英寸的色点,可以有 1670 万种颜色,图像放大率可以在 3% 到 3200% 之间,图形旋转度可以小到百分之一度。

第四个特点是软件的通用性日增。前述的 MicroMind Director 软件制作的动画一旦完成后就可以单独拷贝,也就是说其他用户拷贝动画时无需将整个软件全部拷贝,拷贝后的动画作品可以在无原软件支持的条件照常运行。通用性的另一特点是软件互相嵌套,也就是在使用一个软件时可以直接调用其它软件,而无需退出一个再调另一个。比如 Software Publishing 公司的 Harvard Graphics 软件就有专门模拟 DOS 的程序段,因而在这个程序运行时,DOS 系统支持的七个重要工具软件都可以照常调入运行。

当今美国图像软件中还有很多独创的技巧和新颖的构思,比如一般图像软件都内含一个拼写检查库,对用户的拼写错误予以自动纠正;有的软件可让几台微机同时使用,并分享重要昂贵的硬件设备(如高级打印机);还有的软件在必要的硬件支持下可以与摄像机、录像机相联通,从而使电脑图像产品可以录制在录像带上,录像带上的画面也可以进入电脑屏幕。



## PC 用户

# dBASE III 十和 FoxBASE 十环境下 实用多窗口弹出式菜单的实现方法

山东省计算机服务公司临沂分公司 张志远

目前开发和使用的某些新软件,正在流行着一种新的菜单界面方式:多窗口弹出式。这种方式的优点是:窗口位置、大小可以任意,光棒定位,清晰美观,选择不易出错,逐级弹出,逐级消失,并恢复被其复盖的原来屏幕画面,等等。这种菜单虽然好,但大都是用 C 语言等编写,大部分用户不熟悉。使用最多的是数据库管理系统,如 dBASE III 十等,又不具备这种功能,编制应用程序时,仍使用数字或字母键选择,易出错、不美观。那么,能否利用编制一段子程序,使用时给出某些参数进行调用,以便实现上述功能,本人通过多次实践,在 dBASE III 十和 FoxBASE 十下,实现了这种多窗口弹出式菜单,现介绍给大家供参考。

该程序的实现过程是:

1. 将每行欲显示的菜单内容付给内存变量。
2. 在给定的位置开窗口,显示菜单内容。
3. 调用子程序,利用 Inkey() 函数,返回 ↑、↓、Enter 和 Esc 的键值,控制菜单光棒移动、选择和返回。
4. 弹出下一级菜单或执行该级被选项对应的功能模块。
5. 功能模块执行完后,再现当前级菜单(也可恢复所有前级菜单画面)。
6. 按 Esc 键,当前级菜单窗口消失,返回上一级窗口菜单。

其菜单实例见程序 CD1. PRG(一级)、CD23. PRG(二级)、CD32. PRG(三级)。

\* CD1. PRG 一级窗口菜单

```
SET TALK OFF
SET COLOR TO /W
@0,0 CLEAR TO 24,79
PUBLIC Z
G1="系统管理"
G2="计划管理"
G3="固定资产"
G4="备件管理"
G5="人事管理"
G6="统计管理"
DO WHILE . T.
SET COLO TO W/B
@5,7 SAY G1
@6,7 SAY G2
@7,7 SAY G3
@8,7 SAY G4
```

```
@9,7 SAY G5
@10,7 SAY G6
DO IC WITH 5,7,1,6,"G"
IF Z=" "
RETU
ENDI
DO CD2&Z
ENDD
```

\* IC. PRG 开窗口子程序

```
PARA A,B,C,D,E
* A:起始行,B:起始列,C:行距,D:行数,E:内容
SET COLO TO R/G
@24,20 SAY "提示:上移↑下移↓选择 Enter 退出 Esc"
L=A
LE=LEN(G1)
SET COLO TO W/B
@A-1,B-1 TO A+D+(D-1)*(C-1),B+LE DOUBLE
DO WHILE . T.
SET COLO TO W/G
X=STR((L+C-A)/C,1)
@L,B SAY &E&X && 显示被选菜单
X1=X
W=INKEY(0) && 返回击入的键值
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=" "
SET COLO TO /W
@A-1,B-1 CLEA TO A+D+(D-1)*(C-1),B+LE
EXIT
ENDD
SET COLO TO W/B
```

```
@(&X1-1)*C+A,B SAY &E&X1 && 恢复  
上次选单
```

```
ENDD
```

```
RETU
```

```
* CD23. PRG 二级窗口菜单
```

```
H1="台帐打印"
```

```
H2="台帐管理"
```

```
H3="报表输出"
```

```
DO WHILE . T.
```

```
SET COLO TO W/B
```

```
@8,19 SAY H1
```

```
@9,19 SAY H2
```

```
@10,19 SAY H3
```

```
DO IC WITH 8,19,1,3,"H"
```

```
IF Z=" "
```

```
RETU
```

```
ENDI
```

```
DO CD3&Z
```

```
ENDD
```

```
* CD32. PRG 三级窗口菜单
```

```
I1="台帐查询"
```

```
I2="台帐修改"
```

```
I3="台帐增加"
```

```
I4="台帐删除"
```

```
DO WHILE . T.
```

```
SET COLO TO W/B
```

```
@6,24 SAY I1
```

```
@7,24 SAY I2
```

```
@8,24 SAY I3
```

```
@9,24 SAY I4
```

```
DO IC WITH 6,24,1,4,"I"
```

```
IF Z=" "
```

```
RETU
```

```
ENDI
```

```
DO CD4&Z
```

```
ENDD
```

对于 FoxBASE+, 还可以使用下面更简便的方法实现, 这里仅给出一级窗口菜单程序 MU1. PRG, 其余下级弹出菜单依此类推。

```
* MU1. PRG 一级窗口菜单
```

```
SET TALK OFF
```

```
SET COLO TO /W
```

```
@0,0 CLEA TO 24,79
```

```
G1="系统管理"
```

```
G2="计划管理"
```

```
G3="固定资产"
```

```
G4="备件管理"
```

```
G5="系统退出"
```

```
DO WHILE . T.
```

```
SET COLO TO W/B
```

```
@6,7 TO 12,16 && 开窗口
```

```
SET COLO TO W/G
```

```
@7,8 PROM G1
```

```
@8,8 PROM G2
```

```
@9,8 PROM G3
```

```
@10,8 PROM G4
```

```
@11,8 PROM G5
```

```
N=1
```

```
MENU TO N
```

```
M=STR(N,1)
```

```
IF N=5
```

```
SET COLOR TO /W
```

```
@6,7 CLEA TO 12,16
```

```
EXIT
```

```
ENDIF
```

```
DO MU2&M
```

```
ENDD
```

## CCDOS 2.13 特殊显示功能的妙用

安阳市建设银行电算科(455000) 李林枫

目前, CCDOS 2.13 汉字操作系统应用十分广泛, 利用其特殊打印功能打印文本文件和各类报表已极为普遍, 然而其特殊显示功能的使用者却为数不多。笔者发现, 在每屏仅能显示 10 行的中分辨率显示器上编制各类应用软件时, 由于客观条件限制, 人机界面很难处理, 计算机快速准确的提示功能得不到发挥, 因而编程工作倍受限制。如果把 CCDOS 2.13 的特殊显示功能用于应用软件的设计中, 能够有效地缓解这一矛盾, 较好

地进行人机界面的处理。

下面, 我们针对配置有中分辨率显示器的微机, 以 CCDOS 2.13F 为例, 说明在 FoxBASE 编程中特殊显示功能的具体应用。

数据库的内容常需查看, 利用特殊显示功能中的画矩形和横、竖线的命令, 可像手工制作表格一样把屏幕划分成几个窗口。由于构成窗口的矩形框和横竖线仅占用字符之间的空隙, 能充分利用有限的屏幕空间,

而且排列井然有序,紧凑美观。

利用 CCDOS 2.13F 的上滚当前页和下滚当前页命令,可实现显示内容的上下平移,效果甚佳。

CCDOS 2.13F 的特殊显示功能还可取消光标、设置颜色和清提示行,使显示屏幕清晰整洁,对于无图形功能的 FoxBASE 无疑是一种有效的功能补偿。

命令举例:

1. [在 0 列,1 行]至[20 列,1 行]之间做一横线。

@0,0 SAY CHR(14)+“H5,0,18,160”

2. 在[20 列,0 行]至[20 列,5 行]之间做一竖线。

@0,0 SAY CHR(14)+“S5,160,0,90”

3. 以[0 列,0 行]和[60 列,8 行]为对角线做一矩形框。

@0,0 SAY CHR(14)+“B5,0,0,480,144”

4. 取消光标

@0,0 SAY CHR(14)+“119,0”

5. 建立光标

@0,0 SAY CHR(14)+“119,1”

6. 清提示行

@0,0 SAY CHR(14)+“116,0”

7. 使屏幕上[1 列,1 行]和[70 列,8 行]为对角线的矩形窗口内的字符以行为单位上下平移。

@0,0 SAY CHR(14)+“16,1,,,1,1,8,70”上滚一行

@0,0 SAY CHR(14)+“17,1,,,1,1,8,70”下滚一行

使用 CC-DOS 2.13 特殊显示功能应注意以下三点:

1. 画点、画线和矩形框命令中的(X Y)座标均是用分辨率的点来描述的,而上滚/下滚当前页命令中的座标则是用行列值表示的。

2. CCDOS 2.13 的特殊显示命令因其版本不同而略有差异,使用时应查看相应的用户手册。

3. 特殊显示命令中的字母大小写均可。程序执行方式下,命令行末尾必须加],否则死机。

## 在 DOS 中使用功能菜单的方法

浙江杭州松木场河西 94 号(310007) 金林樵

菜单,是广大程序设计者广泛使用的选择分支方法。在 dBASE、FoxBASE 及 Turbo 系列软件中,菜单的设计给软件增添了勃勃生机。屏幕显示直观,选项移动灵活方便,给用户的使用提供了极大的方便。在 DOS 系统下,没有直接提供菜单功能。给用户在系统级上使用不同的系统设置了障碍。

我们知道,为了实现菜单功能,必须要有一个选择输入的过程,且能根据输入自动进行判断处理分支。但在 DOS 下,仅有一个 PAUSE 命令能实现任一键的输入,但它却不能将该输入值传送给 DOS 以加以判别。因此,只能设计一个过程。实现选项输入,并将该值传送给 DOS,随后用 DOS 提供的 IF、Errorlevel(n)Goto(标号)判断命令在一个批命令文件中依次实现上述菜单分支功能。

首先,用下附程序 MENU.COM 显示功能菜单,并等待用户的选择输入。最后将输入值放入 AL 中后带回 DOS。

然后根据用户输入用 IF、Errorlevel(n)Goto(标号)命令来实现分支转移。具体实现方法见下附程序 AU-TOEXEC.BAT。

该批命令文件执行时,首先显示功能菜单,然后等待用户根据以上选项进行具体选择。输入后,其后的 IF Errorlevel 语句就根据用户的具体输入,作出相应的判断处理并实现分支。

以上程序已在 PC/XT 及其兼容机上调试通过。

MENU.COM 程序清单

C>DEBUG

-A

8EC9:0100 MOV DX,0110

8EC9:0103 MOV AH,09

8EC9:0105 INT 21

8EC9:0107 MOV AX,0C01

8EC9:010A INT 21

8EC9:010C MOV AH,4C

8EC9:010E INT 21

8EC9:0110 DB '0 CCDOS 4,0',A,D

8EC9:0120 DB '1 CCBIOS 2,13H',A,D

8EC9:0133 DB '2 UCDS',A,D

8EC9:013F DB '3 BASICA',A,D

8EC9:014C DB '4 PCTOOLS',A,D

8EC9:015A DB '5 TURBO PASCAL,A,D

8EC9:016D DB '6 TURBO C 2.0',A,D

8EC9:017F DB '7 TURBO DEBUGGER 2.0',  
A,D

8EC9:0198 DB '8 MFOXBASE 2.10',A,D

8EC9:01AC DB '9 dBASE 1,10',A,D,A,D

8EC9:01BF DB ' SELECT(0-9)? \$'

-R CX

CX 0000

:D8

-N MENU.COM

-W

-Q

AUTOEXEC.BAT 程序清单

echo off

cls

menu

if errorlevel 57 goto J

if errorlevel 56 goto I

if errorlevel 55 goto H

if errorlevel 54 goto G

if errorlevel 53 goto F

if errorlevel 52 goto E

if errorlevel 51 goto D

if errorlevel 50 goto C

if errorlevel 49 goto B

if errorlevel 48 goto A

:A

cd\ccdms

..... ;CCDOS 4.0 各引导程序

goto K

:B

cd\213

..... ;2,13H 各引导程序

goto K

:C

cd\ucdos

..... ;UCDOS 各引导程序

goto K

:D

cd\BASIC

BASICA ;进入 BASICA

goto K

:E

cd\PCTOOLS

PCTOOLS ;进入 PCTools

goto K

:F

cd\TP

TURBO ;Turbo PASCAL

goto K

:G

cd\TC

TC ;Turbo C

goto K

:H

cd\TD

TD ;Turbo debugger

goto K

:I

cd\MFOXBASE

MFOXPLUS

goto K

:J

cd\DBASE

DBASE

:K

cd\

## PKZIP 压缩软件包的技术背景及使用

崇文税务分局计算机室(100062) 夏伟文

“文件压缩存档(PKZIP)软件包 1.02 版”是美国“软件专业共享协会(ASP)”推荐,PKWARE 公司研制的产品。这个系列软件的前身是 PKARC,都属于使用“文档文件”来实现对原始文件的加密、压缩存储。所谓“文档文件”是指一种具有自己独特的内部结构标准的文件,当原始文件被储存时,按统一的标准压缩后存入“文档文件”中,用户可以方便地进行文件存取。国际上流行的许多软件都使用了该方法来分类存储各种文件,既可节省磁盘空间,又在一定程度上达到了加密的效果。对于用户来说,PKZIP 压缩软件又是自己加强文件管理的好助手。

PKZIP 与 PKARC 相比,有显著的优点。一是使用了新的 Imploding 方法,压缩的程度更高,虽然压缩时

慢一些,但是提取时速度相当快。二是原来 PKARC 压缩存储可执行的二进制文件时有毛病,PKZIP 则没有这方面的限制。三是可以由用户指定口令字进行加密压缩,并支持 LAN 等局域网工作和其它并行工作方式,其安全性、适应性有了很大的提高。四是可以使用 -rP 参数自动搜索文件的路径名,并将其压缩在文档中,提取时加 -d 参数自动生成所需的子目录。五是增加了“自提取型文档文件(.EXE)”,既可以直接运行实现文档提取,也可以用 PKZIP 或 PKUNZIP 处理(不包括 -V 查看功能),适合于不同水平用户的要求。六是可以根据 CPU 的型号,自己决定是否使用 32 位指令寻址方式和调用 80386 CPU 的加速器。

PKZIP 软件所管理的“文档文件”是后缀名为 ZIP

的文件,这种文件的标准结构如下:

局部文件头部信息+文件内容,局部文件头部信息+文件内容,……,核心目录,核心目录结束信息

局部文件头部信息和文件内容构成一个文件项,一般是每一个文件项存放一个原始文件。

“局部文件头部信息”包括以下字段:

1. 局部文件头标志:4字节(0x04034b50)

2. 能够提取文件的(软件)版本号:2字节

3. 通用的标志位:2字节

第0位:如果置1则为加密文件

如果压缩类型为6,则第1位置1生成8K浮动数据字典,置0生成4K浮动数据字典;第2位置1说明用了3棵Shanno-Fano树来完成数据的输出;第2位置0说明用了2棵Shanno-Fano树来完成数据的输出

4. 压缩方式:2字节。有以下几种:

0——文件不压缩直接存储

1——文件以Shrink方式压缩

i——用第i-1个压缩因子进行Reduce压缩(i=2,3,4,5)

6——以Imploded方式进行压缩

5. 最后的文件(更新)时间:2字节(按标准的MS-DOS格式来编码)

6. 最后的文件(更新)日期:2字节(同上)

7. 32位CRC校验码:4字节

8. 压缩以后文件长度:4字节

9. 压缩以前文件长度:4字节

10. 文件名的长度:2字节

11. 保留字段的长度:2字节

12. 文件名(可变量)

13. 保留字段(可变量)

第12.和13.两项与下面提到的“文档注释”的长度之和不得超过64K

“核心目录”的结构如下:

[文件头],[文件头]……[核心目录结束信息]

[文件头]除了包含“局部文件头部信息”的所有字段(不包括第一个)外,还包括以下字段:

1. 核心目录头部标志:4字节(0x02014b50)

2. 适用的版本号:2字节

高位字节代表操作系统,低位字节代表生成文档的软件版本号

3. 文档注释的长度:2字节

4. 开始的磁盘(扇区)号:2字节

5. 内部文件属性:2字节

如果最低位置1则说明原始文件是ASCII或文本文件,否则为二进制文件。

6. 外部文件属性:4字节 原始文件的属性(其类型依赖于OS环境)

7. 局部文件头部信息的相对偏移量:4字节

据此可以找到“局部文件头部信息”的存储位置

[核心目录结束信息]包含的数据项有:

1. 核心目录结束标志:4字节(0x06054b50)

2. 本磁盘核心目录数目:2字节

3. 核心目录开始的磁盘(扇区)号:2字节

4. 本磁盘核心目录中登记项数目:2字节

5. 核心目录中登记项数目:2字节

6. 核心目录的长度:4字节

7. 核心目录相对于文件开始磁盘扇区号的偏移量:2字节

8. ZIP文档注释的长度:2字节

9. ZIP文档注释的内容(可变量)

压缩的方式主要有以下几个:

1. Shrink;这是一种改进过的动态Ziv-Lempel-Welch压缩算法。初始的码长是9位,最大的码长是13位。它改进的功能有:(1)码长由压缩软件控制,不会使码长没有必要地自动增长,避免编码时的数据阻塞。(2)当码表填满时,并不清除全部结点,只释放Ziv-Lempel树的叶结点,以供使用。

2. Reduced;这种算法实际上是两种不同算法的混合使用。第一种算法是压缩重复的字节序列,第二种算法根据第一种算法压缩得到的数据流,再应用一种概率压缩方法进行处理。

3. Imploding;该算法也是由两种不同算法组合而成的。第一种算法是用浮动数据字典来压缩重复的字节序列。第二种算法是用2~3棵Shannon-Fano树来实现对浮动数据字典编码输出的压缩。如果使用3棵Shannon-Fano树,第一棵树用来描述ASCII码字符集。后两棵树分别用来描述浮动数据字典中的数据项对——长度和距离信息。如果只使用2棵Shannon-Fano树,则描述ASCII字符的树被忽略,只存在后两棵树。

4. 加密压缩。每一个加密的文档都定义一个12字节长的加密文件头存放在文件内容区域的开始部分。每一个加密文件头初始时被设置成随机数,接着用3个32位密钥来加密,密钥由用户指定的口令字生成。等到原始文件的每一个字节都被加密后,密钥将用伪随机代码生成技术与文件的32位CRC校验码结合在一起,放在文件的某个地方。

解密的步骤如下:①32位密钥被用户指定的口令字初始化。②读入12个字节的加密文件头,解密后用以更新32位密钥。③用这32位密钥对整个文件进行解密。

该软件可以在IBM PC及其兼容机上运行,要求DOS版本在2.0以上或者是OS/2。PKZIP需要85K的内存、PKUNZIP需要70K的内存才能运行。该软件一次最多可以将3900个文件压缩入一个文档文件。为了使文档文件在不同的OS上统一,路径名中用“/”代替MSDOS中的“\”。

“文件压缩存档(PKZIP)软件包1.02版”包括下列应用程序及文件:

WHATSNW.102 PKWARE 1.02版新增加的功能。

OMBUDSMN. ASP“软件专业共享协会(ASP)”有  
关于本软件包的通知。

DEDICATE. DOC 关于开始使用. ZIP 作为“压缩  
(Compressed)型数据文件”的括展名(后缀)的通知。

LICENSE. DOC 关于机关,单位(公司)取得本软件  
包使用许可证的手续。

MANUAL. DOC 本软件包的使用说明书。

ORDER. DOC 本软件包的使用登记与定购单。

README. DOC 本软件包简介。

REZIP. DOC 旧版(PKZ091,PKZ092),ZIP 文件的  
兼容与升级。

APPNOTE. TXT 本软件包的技术背景资料。

BIOSFIX. COM 停用 80386 CPU 加速器(32-bit  
accumu lator;EAx)功能。

PKSFX. PRG 自动配合 ZIP2EXE. EXE 文件使用。

MAKESFX. COM 当 PK102. EXE 在当前目录时,  
执行 MAKESFX. COM 则生成 BIOSFIX. PRG 文件。

PKUNZIP. EXE 对已压缩的文档文件(\*. ZIP)的  
提取程序。

PKZIP. EXE 对文件进行压缩存档的程序。

PKZIPFIX. EXE 用于修复已局部损坏的文档(\*.  
ZIP)文件。

READ\_1ST. EXE 本软件包的来源说明。

REZIP. EXE 用于将原用 PKZ090. EXE 或 PKZ092  
压缩存档的文档文件改为 PKZ101. EXE 的格式存档。

ZIP2EXE. EXE 用于生成自提取(Self-extracting)  
文档文件。

PKZ102. EXE 本软件包内的全部文件均已存入这  
个自提取文档。

注:当我国软件保护法生效后,请您自觉按规定向  
版权法人交费。

下面简要介绍一下 PKZIP 软件包的使用。

(1)PKZIP. EXE 的使用格式:

PKZIP[-b[路径]] [可选参数] [-s(password)]  
[ZIP 文档文件] [@文件表] [文件...]

可选参数包括:

-d=删除文件 -l=显示版权信息

-f=更新文件(只更新以前已经存储的文件)

-u=更新文件(将已经存储的文件用最新版本更  
新,将没有存储的文件加入)

-b=在指定的驱动器上建立临时的 ZIP 文件

-c=加入/编辑文件注释

-C=只对新文件加入注释

-i=只将已经修改过的文件加入文档

-m[u,f]=将原始文件移动到文档内,原文件将  
被删除

-r=自动寻找当前目录下的子目录中的文件

-es=使用快速压缩 -a=加入文件

-ex=使用最大限度的压缩(默认)

-k=保持原有的 ZIP 日期

-z=增加/编辑文档文件的总注释

-o=将 ZIP 文件日期设置成最近的原始文件的  
日期

-P=将路径存入 ZIP 文档内(应与-r 参数合  
用,注意不要把大小写弄错)

-q=允许 ANSI 的注释

-x<通配符>=不包括指定的文件

-s<口令字>=用口令字给文件加密

-w/W<H,S>=包括/(或不包括)隐含/系统文件

-j/J<H,S,R>=对文件压缩存储时屏蔽/(或不屏  
蔽)隐含/系统/只读文件的属性

ZIP 文件=ZIP 文档文件名(包括路径名),默认  
的扩展名是. ZIP

文件 =将要压缩的原始文件名(包括路径名)可  
使用通配符如\*,?。默认的是所有的文件

@文件表=包含了将要观察或压缩存储的文件名  
字的登记表

(2)PKUNZIP. EXE 的使用格式:

PKUNZIP [可选参数] ZIP 文档文件 [盘符:输出  
路径] [要提取的文件.....]

可选参数为:

-c[m]=把原始文件提取出来并送到屏幕[加 m  
参数可逐屏或逐行显示]

-t=测试 ZIP 文件的完整性

-l=显示软件的版权信息

-d=建立输出路径中的目录用以存储提取出的  
原始文件

-o=覆盖已经存在的原始文件

-q=允许 ANSI 的注释

-s<口令字>=用口令字将文件解密提取

-n=只提取出较盘上已有文件更新的文件

-p[a,b,c][1,2,3]=将提取出来的文件送到打  
印机

[按 ASCII 码/按二进制码/送往 COM 通信接口]  
[通信口编号]

-j/J<H,S,R>=在提取原始文件时屏蔽掉/(或不  
屏蔽)隐含/系统/只读等文件属性

-v[b,c,d,e,n,p,s,r]=观察 ZIP 文件[简要地列  
表/按 CRC 码/日期/括展名/名字/压缩百分比/长度  
排序/按逆序排列]

-e,x=分离出原始文件(默认)

-v 参数是 PKZIP 和 PKUNZIP 都可以选用的,其  
含义为:

-v[b,c,d,e,n,p,s,r,t]=观察 ZIP 文件[简要地  
列表/显示注释/按日期/括展名/名字/压缩百分比/长  
度排序/按逆序排列/详细列出技术细节]

(3)PKZIPFIX. EXE 使用格式如下:

PKZIPFIX ZIP 文件

PKZIPFIX 将会重建已经局部损坏的 ZIP 文件,并  
将从该文件中可能恢复的原始文件存到 PKFIXED.  
ZIP 文档中。

(4)自提取文档生成程序(ZIP2EXE. EXE)的使用



格式如下:

### ZIP2EXE ZIP 文档文件

它可以把已生成的. ZIP 文档用本程序转变成同名的自提取型(. EXE)文件,使用这个文件时必需把PKSFX. PRG 文件与它拷在一个目录内或建立路径后使用。自提取型文件仍能用 PKZIP. EXE 或 PKUNZIP 进行处理。执行自提取型文件后将根据指定的参数提取出文件。执行的格式如下:

自提取文档名[可选参数][盘符:输出路径][文件.....]

可选参数包括:

-c[m]/-e/-d/-l/-o/-t/-n/-p[a,b,c]

[1,2,3]其含义与 PKUNZIP 的参数相同。

对于 1.1 版的 PKZIP 软件包而言,还可以加上一 s(psw)参数(其含义同上)。

自提取型文档一旦生成以后,可以用 -d 参数删空存放其内部的无用文件,将它作为一个基础文档保留起来。需要自提取型文档时,只要拷出一个备份,就可以用 PKZIP 直接将原始文件压缩到该自提取型文档中即可,无需再用 ZIP2EXE 进行转换。

## 如何使《UCDOS V1.0》 支持 AT 101/102 键盘

辽宁省计量测试技术研究所(110006) 杨旭

中国科学院希望高级电脑技术公司开发的《超级组合式汉字系统》UCDOS V1.0,具有许多独到的优点,深受广大用户的欢迎。遗憾的是,该软件不支持 AT 101/102 键盘中部的两组小键盘(扫描码为 E0 xx)。不但不能发挥 AT 101/102 键盘的优势,反而造成许多不便,影响输入速度。因此,本人通过对其键盘程序的分析,加入一小段程序,恢复了这两组小键盘的正常功能。使 UCDOS V1.0 在 AT 及其兼容机上应用起来更加方便。用 DEBUG 修改其键盘程序的方法如下:(修改后的程序在 286 及 386 机上运行通过)

```
C:\>CD\UCDOS
C:\UCDOS>REN KB. EXE KB
C:\UCDOS>DEBUG KB
-A CS:9039
CS:9039 JMP A1C0
CS:903C
-A CS:A1C0
CS:A1C0 JNZ A1CD
CS:A1C2 OR AL,AL
CS:A1C4 JZ A1DD
CS:A1C6 CMP AL,E0
CS:A1C8 JZ A1D8
CS:A1CA JMP 900E
CS:A1CD OR AL,AL
CS:A1CF JZ A1DD
CS:A1D1 CMP AL,E0
CS:A1D3 JZ A1D8
CS:A1D5 JMP 926E
CS:A1D8 MOV AL,00
CS:A1DA MOV [0038],AX
CS:A1DD JMP 9048
CS:A1E0
```

```
-R CX
:A1B3
A1E0
-W
-Q
C:\UCDOS>REN KB KB. EXE
```

## 恢复硬盘启动 功能的方法

福建泉州华侨大学机械系(362011) 王晓红

操作系统型病毒感染或其他原因,破坏硬盘主引导记录的数据,将使硬盘不能启动。由于硬盘主引导扇区是非 DOS 部分,DOS 不能直接访问,因而常见的解决办法是低级格式化硬盘,重新分区,格式化。这样做,硬盘原来的数据全部丢失,造成巨大的损失。针对这一问题,本文介绍一种利用 ROM BIOS 的 INT 13H 来“救活”硬盘的方法,具体步骤如下:

1)准备一张带有 DEBUG.COM 的用户盘,并用行编辑命令建立 RDBOOT 和 WTBOOT 两个文件;

2)在一台带有同型号健康无毒的硬盘的机上键入:

```
A>TYPE RDBOOT | DEBUG
```

不同类型的硬盘的结构差别较大,分区表各不相同,所以最好在机器正常的时候做上述两步,并存好磁盘待用。若有几种型号不同的计算机,只需改变上述两文件中的 N BOOTXT 句,如在 286 机上,则改为 N BOOT286。

3)当机器出现主引导扇区坏而不能启动时,用软盘启动后,插入上述软盘,然后键入:

```
A>TYPE WTBOOT | DEBUG
```

重新启动后机器工作正常。

```
RDBOOT 文件
A>type rdboot
a 100
mov ax,0201
```

```
WTBOOT 文件
A>type wtboot
n bootxt
1 200
```

```

mov bx,0200      a 100
mov cx,0001      mov ax,0301
mov dx,0080      mov bx,0200
int 13           mov cx,0001
int 20           mov dx,0080
                int 13
                int 20
g=100           int 20
r cx
0200           g=100
n bootxt       q
w 200
q

```

(上接 48 页)

有错,这往往是硬件故障造成。例如一台长城 0520CH 的彩色显示器,显示信息杂乱,经检查结果,显示卡故障。现在又发现,如机器感染上象圆点等一类病毒,一旦病毒发作,屏幕显示就出现“圆点”跳动干扰屏幕显示。这种问题的出现只要用解病毒软件清理一下即可恢复。

#### 19. 打印机分哪两大类? 有哪几种打印方式?

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,复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 8 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。一般使用的大多数是击打式打印机,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,而字符的形状是以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,也可能打印汉字字形或任意形态的图形。主机所输出的西文字符的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,也可以存储于随机存储器 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。

#### 20. 打印机控制器与打印驱动程序有什么关系? 使用汉字打印驱动程序应注意什么问题?

打印机控制器或称适配器可以是一块独立的选件板,也可以是“显示控制器/打印控制器”选件板的一个组成部分。打印控制器由命令译码器数据锁存器、总线缓冲器、控制锁存器、控制驱动器以及接收发送器等逻辑部件组成。命令译码器负责识别从 CPU 送来的具有系统分配给打印控制器所规定的端口地址的输入输出指令,并且译出“数据传送方向”,“读数据”,“写数据”,“写控制”,“读状态”,“读控制信号”等六个命令。数据锁存器用来暂存 CPU 送出的打印数据字节,然后一方面通过 D 形插头座送到打印机去打印,另一方面又回送到总线缓冲器,以便 CPU 必要时可以读入比较,进行故障测试与诊断。控制信号锁存器及集电极开路驱动器用来暂存并驱动四个控制信号:初始化、选通、自动换纸以及选择输入信号。这些控制信号在输出的同时也回送到另一个总线缓冲器,同样也是为诊断时用。打印机的五个状态信息:联机、认可、忙、缺纸、出错均为稳态电平信号,它们经由总线缓冲器被“读状态”命令选通后,即送往数据总线,由 CPU 的输入指令取回送入某寄存器中,供软件作分析打印机状态时使用。

有了打印控制器,还要有打印驱动程序的支持。通常打印输出控制选用查询方式或者中断方式来实现。在 IBM PC 的 ROM 常驻 BIOS 中,打印输出控制采用的是查询方式。需要指明的是,系统里提供的打印驱动程序仅支持英文字符的打印,即在 PC DOS 各种版本控制下的系统,只能输出字符信息,对于汉字信息处理时,就需要另配汉字打印驱动程序,这些程序要根据不同类型的打印机分别进行设计。

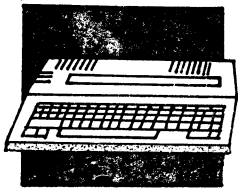
使用汉字打印驱动程序应注意下列问题:

(1) 打印驱动程序应与打印机的型号配套。换言之,什么型号的打印机,要选择与该型号打印机配套的打印驱动程序。要注意,同一型号的打印机可有功能强弱不同的打印驱动程序。所以当你购买一型号打印机时,千万不要忘记让公司或厂家配给相应的打印汉字驱动程序。

(2) 打印驱动程序应与汉字库文件配套。因为有的功能较简单的汉字打印驱动程序,仅要求 16×16 点阵 CCLIB 汉字库文件,功能稍强一些的打印驱动程序,不但要求 16×16 点阵 CCLIB 汉字库文件在内存,同时往往由于变换汉字字型的需要,要求 24×24 点阵 CLIB24 汉字库文件在硬盘上。

(3) 打印驱动程序要与使用的汉字系统版本配套。汉字打印驱动程序运行的环境在汉字系统下,不同汉字系统下,同一打印机,有功能不同的打印驱动程序。

工作的实践经验告诉我们如果打印机打不出汉字来,或者打印汉字时死机,或打印的不是汉字就很有可能是上述三个问题没有解决好。



## 学习机之友

# 无须驱动器的电子打字机程序

重庆出版社科技编辑室(630050) 刘庆丰

各类型号的中华系列学习机均有较强的汉字功能。如小蜜蜂(XMF-1)机不但具有五种汉字输入法(其中的部首方式连选择键在内最多四键即可输入一汉字),而且固化了七千多单词的联想式词组,汉字输入速度已不逊于各类昂贵的电子打字机。但要使各种类型的学习机具备电子打字和汉字处理功能,必须要依赖于驱动器,对广大未配驱动器的学习机用户,要用电脑打字则必须将其内容逐一填入 PRINT 语句之中,且每行字数还不能超过八十汉字,相当繁琐。

若能使输入的汉字或西文字符自动变为 PRINT 语句中的内容,则可使学习机具备电子打字机的大部分功能,有相当的实用价值,本程序即是根据这一构想设计的。

### 使用方法

键入本程序并存带。每次使用时只需将其输入内存,运行后首先让你选择标题及正文的字号及字距等,接着即按日常书写习惯在句首空出两个汉字空间(也可用退格键删去)等候输入。对输入的所有汉字及西文,它均将其变为 PRINT 语句的内容,对双引号则以 CHR\$(34)的形式生成程序。当每行超过五十字符时自动加入末尾的双引号和分号,并另起一行继续书写 PRINT 语句,所以各自然段的字数可无任何限制,直至按下回车键时才自然分段。写作过程中若发现输入有误,可按退格键,程序将自动删去最后的一个汉字或西文或双引号,继续按可继续删除,若本语句行字符已全部删完,则删去上一行的结尾字符,同时行号自动减十。若按 CTRL-X 键,则删去上次回车后的本自然段全部内容。若按右行键,就又把光标划过的字符逐一添入到程序后,若误按回车使内容换行,可按 CTRL-B 键删去回车。

为提高人机界面的友好度,系统还设置了 CTRL-A 和 CTRL-C 两个功能键。后者用于查看已写字数和已生成的行行数,前者是一个用途较广的键,现将功能简述如下:

1. LIST 任何时候均可按下,以观察已完成的内容。列表结束后将自动返回等待输入状态。
2. RUN 按下后显示键入的全部文字内容,显毕结束程序运行。若要继续,可键入 GOTO 50。在对程序进行手工改动后欲继续写作,则应键入 S%=行号: GOTO 50。
3. 选字号 当键入选好的字号后,程序即已将该号字体下每行的打印字数一并设好(按普通稿纸宽度计算),以 POKE 语句的形式生成一个单独的语句行。

它还能按使用者要求自行设置前置空格以供标题制作之用。同时,程序还提供了两种用于正文的标准字体,均按每行 20 字,每页 20 行设置;标 1 为标准二号字,标 2 为小号长方字,但每字间自动加入半汉字空间,用于文章引述,诗歌等需另置字体的场合。

4. 序号 这是为文内作段落标题和句首标号设置的,可免去写作时查索标号数字的麻烦。

文章全部写完后,若需作大的编辑加工,可用任何编辑 BASIC 程序的手段进行处理。若要存带,可先删去 0—500 行,也可连同这部分写作系统一并存储。后者在存带前最好加上一句 0 GOTO 1000 以免使用时发生混乱。

### 程序分析

本写作系统主要分为自动生成程序的机器语言核心程序和汉字输入与提供人机对话接口的 BASIC 程序两大部分。

由于电脑打字时输入的并不全是欲打印的内容,还有很多是控制命令等,所以必须对输入的内容逐一进行识别,选择。这用 GET 语句是很理想的。但各学习机的 GET 都不能用于汉字状态,如 CEC 用 GET 采入汉字时得到的却是当时键入的拼音码或区位码,小蜜蜂机虽可键完一个汉字才采集一次,却只能获得以该汉字三内码最后一码开始的一串无用字符。

通过对内存的查询,找到了小蜜蜂在汉字 GET 时的一个特殊储存单元: \$B714。该单元在输入汉字时储存的是汉字内码的第二码,当输入 ASCII 码和控制符时内容为 35 或 33。据此就可以编出汉字 GET 的程序了。因为第一字节的引导码是可以自己加进去的,繁体汉字为 127,简体为 126。

作为人机界面的 BASIC 程序设计了较多的功能与提示,这部分读者可根据自己的情况进行增删。

位于 \$7800 至 \$79C9 间的机器语言程序是系统的核心,它又是由若干子程序组成,有较强的容错能力,对各种误操作均能响铃示警,并考虑了与各种机型的兼容,凡是汉字引导码为 127 或 126,且码长为三字节的各种苹果系列软硬汉字系统均可运用。为方便读者灵活运用,现将一些主要功能段及其入口地址简介如下:

\$780E:查变量。调用前将单字节的字符串变量或整数型变量的 ASCII 码存入 \$FE 单元,调用后变量地址在 \$4C, \$4D 二单元中。

\$78C0:根据 S% 之值将新编行号写至程序尾部。

\$78EC:将 A\$ 内容变为程序内容。灵活调用孩子

程序,可生成任何程序。但要注意的是程序中的保留字只能用 CHR\$(保留字内码)的形式生成。

\$ 7922:删去程序尾部的一个字符。

\$ 7987:删去上次回车后输入的所有内容

### CEC 机使用本系统的方法

本程序是在 XMF 机上设置通过的。CEC 机若要用本系统,需对程序作如下改动:

1. 程序首的 PR#5 应改为 PR#3。

2. 改动汉字输入语句。CEC 的汉字输入入口不同于 XMF 机,为此需编一段机器语言程序来作为汉字 GET 语句。具体作法是:

(1)加入以下两行程序

```
18 FOR X=768 TO 774:READ A,POKE X,A;
NEXT:DATA 32,129,158,141,16,3,96
```

```
55 CALL 768:A$=CHR$(PEEK(784)):IF
PEEK(784)=127 THEN CALL 768:A$=A$+CHR
$(PEEK(784)):CALL 768:A$=A$+CHR$(PEEK
(784))
```

(2)删去 95,100 两行全部内容。

3. 本程序中 240—250 三行为设置打印机的控制字符(字号,行宽,行距等),CEC 机用户可对此作自己的任意安排。方法是参照此程序将本机的控制码写入 A\$ 内(程序中的 CHR\$(185)即 POKE 的保留字代码)即可。

4. 由于 CEC 机不能任意使用汉字屏幕第十一行(但汉字状态下可有反白显示),故可将提示改为反相显示。即将本程序中所有用到 VTAB 11 的地方都改为 INVERSE,并删去 140 行的子程序。

本程序在有驱动器的情况下亦可照常运行

```
2 PR# 5:PRINT
```

```
3 VTAB 3
```

```
5 PRINT"          汉字文稿写作系统":VTAB 8
```

```
6 PRINT SPC(9)"编制人:刘庆丰":PRINT SPC(13)"1991.
1."
```

```
8 FOR X=30720 TO 31177:READ A,POKE X,A:NEXT
```

```
9 DATA 56,165,175,229,255,133,74,165,176,233,0,
133,75,96,56,165,107,229,105,133,23,160,0,196,
23,144,3,76,221,251,177,105,197,254,24,240,7,
152,105,7,168,76,23,120,152,101,105,133,76,8
```

```
10 DATA 200,177,105,201,128,240,4,136,76,37,120,
40,165,106,105,0,133,77,96,56,165,74,229,255,
133,74,176,2,198,75,96,24,165,74,101,255,133,
74,144,2,230,75,96,160,0,152,145,74,200,145
```

```
11 DATA 74,200,145,74,160,0,24,165,74,105,1,145,
238,200,165,75,105,0,145,238,24,165,74,105,3,
133,175,165,75,105,0,133,176,96,160,0,152,133,
252,169,1,133,255,230,252,32,69,120,177,74
```

```
12 DATA 201,186,208,245,56,165,74,233,4,133,238,
165,75,233,0,133,239,96,169,211,133,254,32,14,
120,160,3,56,177,76,233,10,145,76,136,177,76,
233,0,145,76,96,169,2,133,255,32,0,120,165
```

```
13 DATA 74,133,238,165,75,133,239,169,211,133,
254,32,14,120,160,2,177,76,200,145,74,177,76,
```

```
136,145,74,169,4,133,255,32,81,120,76,93,120,
169,65,133,254,32,14,120,160,2,177,76,133,25,
200
```

```
14 DATA 177,76,133,23,200,177,76,133,24,169,3,
133,255,32,0,120,160,0,196,25,176,8,177,23,145,
74,200,76,12,121,165,25,133,255,32,81,120,76,
93,120,169,6,133,255,32,0,120,160,0,177
```

```
15 DATA 74,133,227,201,127,240,19,201,126,240,15,
169,2,133,255,32,81,120,177,74,133,227,201,34,
240,3,76,93,120,169,6,133,255,32,69,120,177,74,
201,34,208,3,76,93,120,169,1,133,255,32
```

```
16 DATA 69,120,177,74,201,59,240,8,32,221,251,
169,160,108,54,0,32,134,120,24,165,238,101,252,
133,74,165,239,101,253,133,75,32,168,120,76,41,
121,32,168,120,32,134,120,169,6,133,255,32,69
```

```
17 DATA 120,160,0,177,74,201,59,240,235,169,8,
133,255,32,81,120,76,93,120,32,134,120,169,6,
133,255,32,69,120,160,0,177,74,133,227,201,34,
240,3,76,221,251,32,134,120,165,252,133,255,32,
81,120,32,93,120,76,168,120
```

```
20 LOMEM:28672:HIMEM:30720:S%=1000:VTAB 11
```

```
22 PRINT"          行号自 1000 行开始?":GET A$
```

```
23 IF A$="N" THEN INPUT "多少?":S%
```

```
24 CALL 30912:A$=CHR$(186)+CHR$(34)
```

```
"          ":CALL 30956
```

```
26 GOSUB 180:S%=S%+10:GOSUB 140:IF A=1 THEN
29
```

```
28 INPUT"          1. 设空间,字距 0. 不设          ":A$:IF A$
="1" THEN 260
```

```
29 V=10:GOSUB 140
```

```
30 CALL 30912:PRINT"          "
```

```
35 A$=CHR$(186)+CHR$(34):CALL 30956
```

```
40 A$="          ":CALL 30956
```

```
45 IF FRE(0)<1000 THEN PRINT "FRE(0)/3="FRE
(0)/3
```

```
50 FOR I=1 TO 50
```

```
55 GET A$:A$=LEFT$(A$,1):PRINT A$;
```

```
60 IF A$=CHR$(34) THEN A$=CHR$(34)+CHR
$(231)+"(34)" +CHR$(34):GOTO 105
```

```
65 IF A$=CHR$(13) THEN A$=CHR$(34):CALL
30956:S%=S%+10:ZZ=ZZ+Z:Z=0:GOTO 30
```

```
70 IF A$=CHR$(8) THEN GOSUB 130:PRINT CHR$(8
*(PEEK(227)=126 OR PEEK(227)=127)):Z=Z-1:
GOTO 110
```

```
75 IF A$=CHR$(24) THEN CALL 31111:PRINT "\":
PRINT"          ":Z=0:GOTO 40
```

```
80 IF A$=CHR$(1) THEN 150
```

```
85 IF A$=CHR$(2) THEN CALL 31139:V=8*(PEEK
(227)=34):PRINT CHR$(V)CHR$(V)CHR$(V)
CHR$(V):GOTO 110
```

```
90 IF A$=CHR$(3) THEN 350
```

```
95 IF PEEK(46868)=35 OR PEEK(46868)=33 THEN 105
```

```
100 A$=CHR$(126)+CHR$(PEEK(46868))+A$
```

```
105 A$=A$+K$:CALL 30956:Z=Z+1
```

```
110 NEXT
```

```

115 A$=CHR$(34)+“;”;CALL 30956          $(185)+“40962,20”;CALL 30956;K$=CHR
120 S%=S%+10;CALL 30912                  $(160);A=1;ZS=40
125 A$=CHR$(186)+CHR$(34);CALL 30956;GOTO 255 RETURN
      45
130 CALL 31010;IF PEEK(227)=160 THEN 130 260 GOSUB 140;INPUT“本行共有几字?”;A
135 RETURN                                265 GOSUB 140;INPUT“字间留多少空格?”;B;QK=ZS-A
140 VTAB 11;HTAB 1;PRINT“              -(B*(A-1)/2);KG=2*QK
      ”;VTAB V;RETURN                    268 INPUT“1.前 1/3 2.后 1/3 0.正中”;A$;IF A$=
150 VTAB 11;HTAB 1                      “1” THEN QK=KG/4
152 INPUT“1.LIST 2.RUN 3.选字号 4.序号”;A$;V=270 IF A$=“2” THEN QK=KG*3/4
      10;GOSUB 140;ON VAL(A$) GOTO 155,500,26,273 REM
      300
155 LIST 1000,;GOTO 110                 274 IF QK<0 THEN PRINT“超宽!重选”;GOTO 260
180 V=11;GOSUB 140                      275 K$=“”;IF B<>0 THEN FOR I=1 TO B;K$=K$
181 REM                                  +CHR$(160);NEXT
182 INPUT“几号字? 1—8;字号 9;标 1 0;标 2”;A$;IF A$ 280 V=10;GOSUB 140;CALL 30912;A$=CHR$(186)+
      $(<“0”OR A$)>“9” THEN 180          CHR$(34);CALL 30956;A$=“”;FOR I=1 TO QK;
185 IF A$=“1”OR A$=“7” THEN ZS=44      A$=A$+“”;NEXT;PRINT A$;;CALL 30956;GO-
195 IF A$=“2”OR A$=“5”OR A$=“8” THEN ZS=TO 50
      22
205 IF A$=“3” THEN ZS=30                300 V=11
215 IF A$=“4” THEN ZS=15                 302 VTAB V;PRINT“1.设序号 0.写序号”;GET A$;IF A
225 IF A$=“6” THEN ZS=11                 $()“1” THEN 315
230 IF A$=“9” THEN 245                   303 HTAB 1
235 IF A$=“0” THEN 250                   305 XH=1;PRINT“若序号不从头开始则按(1)”;GET A
240 CALL 30854;CALL 30813;A$=CHR$(185)+$
      “40963,”+A$+“;”+CHR$(185)+“40961,”+308 IF A$=“1” THEN INPUT“从几开始?”;XH
      STR$(ZS);CALL 30956;K$=“”;A=0;RETURN 310 GOSUB 140;INPUT“1.正中 2.句首”;B$;GOTO 320
245 CALL 30854;CALL 30813;A$=CHR$(185)+315 XH=XH+1
      “40963,2;”+CHR$(185)+“40961,20;”+CHR320 V=10;GOSUB 140;IF B$=“2” THEN 330
      $(185)+“40962,20”;CALL 30956;K$=“”;A=1;325 A$=“”;FOR I=1 TO ZS-7;A$=A$+“ ”;NEXT;
      ZS=20;RETURN                        A$=A$+STR$(XH)+“.”;CALL 30956;PRINT A
250 CALL 30854;CALL 30813;A$=CHR$(185)+$;S%=S%+10;GOTO 30
      “40963,1;”+CHR$(185)+“40961,30;”+CHR330 A$=STR$(XH)+“.”;CALL 30956;PRINT A$;;
      $;V=10;GOSUB 140;GOTO 50          GOTO 50
350 VTAB 11;PRINT“行号=”S%“字数=”ZZ+Z;GET A
      $;V=10;GOSUB 140;GOTO 50

```

## TOOL—KIT 使用详解

北大附中 姜 宏

### (一)系统简介:

TOOL—KIT 软件是苹果公司推出的工具软件,之所以叫工具箱,是由于它里面包含着几个很有用的工具程序,这些程序是用户更好利用苹果机的基础。本文就是为介绍它们而写的。

这张磁盘里,有以下几类程序:

(1)工具类:这类文件可帮助用户设计 CAI 软件,比较典型的是 HRCG 程序。

(2)游戏类:这类程序大多是工具程序的示范,剖析这些程序,你会有意想不到的收获。

(3)字符集:可帮助用户编写游戏程序。

### (二)使用详解:

#### 1. 再定位载入程序——RBOOT、RLOAD

TOOL-KIT 这张软盘里,有一类很奇怪的文件,即 R 型文件。R 是英文 Relocatable 的简写,中文意思是再定位,在 DOS3.3 中,没有存取这类文件的命令。这使 R 类文件的功能无法全部发挥,苹果公司为此写了这两个程序,在介绍它们之前,我们先介绍一些基本知识。

根据苹果公司的资料,DOS 共可处理 8 类文件,它

们在目录道的代码及中文含意为:

- \$ 00—TEXT 数据文件(T 型)
- \$ 01—INTEGER 整数 BASIC(I 型)
- \$ 02—FP—BASIC(APPLE SOFT)(A 型)
- \$ 04—BINARY 二进制代码文件(B 型)
- \$ 08—SYSTEM 系统文件(S 型)
- \$ 10—RELOCATABLE 可再定位文件(R 型)
- \$ 20—新 APPLESOFT 文件(?)
- \$ 40—由 LISA 2.5 生成的 6502 汇编文件源程序(L 型)

但由于种种限制,DOS3.3 并没有加上对 S、R、? 和 L 四类文件的处理程序,LISA 2.5 在中国风行后,L 类文件被人注意了,一些汉字系统推出后,字库文件符号采用 S、?,那两类文件也被人注意了,而 R 类文件一直到 TOOL—KIT 推出之后才有人注意,那么,它的原理是什么呢?

我们先来看看下面两个简单的 6502 汇编程序:

```
程序 1:0800-A9 00    LDA # $ 00
           0802-4C 00 08    JMP $ 0800
程序 2:0900-A9 00    LDA # $ 00
           0902-4C 00 09    JMP $ 0900
```

这两个程序执行效果相同,只有一个代码不同,即: \$ 804 单元中的 \$ 08 与 \$ 904 中的 \$ 09,由于内存地址的问题,地址的高 8 位应根据地址值高位予以适当的修改。即当我们想把输入好的程序 1,拷贝至 \$ 900 作程序 2 时,应做如下工作:

\* 900 < 800. 804M

(把 \$ 800—\$ 804 单元的内容拷贝至 \$ 900 开始的内存)

\* 904;9

(把 JMP 的高 8 位地址改为 \$ 09,即 \$ 904)

象上面这样,一个 B 型文件要在不同地址的内存中运行,就要做较大的修改,由于这类问题,R 型文件应运而生,我们先看看 LOADAPA 程序的调用段:

```
140 PRINT CHR $ (4);“BLOAD RBOOT”
150 CALL 520
160 ADRS USR(0),“APA”
180 CALL ADRS
```

下面逐一分析各行程序:

140 行:调入 RBOOT 程序,该程序首地址为 \$ 208 (520)

150 行:启动 RBOOT,设定 USR 地址,调入 RLOAD 文件。

160 行:用 USR 命令调入 APA,这里对名为任意的 filename 文件,都可采用下法调用:

行号 ADRS=USR(0),“filename”

变量 ADRS 是程序的首地址,RLOAD 把要调入的程序转移至末尾地址为 HIMEM—1 的地址中,并按绝对地址修改好 APA 程序,这样,就可保证程序可在任意内存运行,如不想要 APA 的标题显示,在 180 行中用 CALL ADRS+3 即可。

180 行,调入 APA,设定 & 命令转移地址。

这里应介绍一个基本知识,即 B 与 R 型文件是不同的,B 型文件(如果是程序的话)一般只能在某一固定内存运行,而经 RBOOT 处理后的 R 型文件,则可以在任意内存中运行。

R 型的文件用处有很多,由于不想牵扯太远,故只介绍到此。

最后,简单介绍苹果公司编写的 3 个 R 型文件:

1)APA,是一个实用的程序设计辅助工具,收录在 TOOL—KIT 中。

2)HRCG,是一个可在高分辨页显示 ASCII 字符集的文件,也收在 TOOL—KIT 中。

3)ASSEMBLER,是一个可在 APPLE 机运行的汇编文件,使用方便,功能齐全,收在 ASSEMBLER 软件中,该软件的简化版本 EDASM 亦已收在 TOOL—KIT 中。

## 2. 程序设计辅助工具——APA

APA 程序全名为 Applesoft Programmer's Assistant,取其第一个英文字母,名为 APA。由于它是一个 R 型文件,因此只能用前文所述方法或执行 LOADAPA 调用。步骤如下:

1)插入 TOOL—KIT 盘,开机,按 E 键

2)打入 RUN LOADAPA,回车

3)过约 30 秒后,APA 调入内存。

下面的字幕显示出来:

```
APPLESOFT PROGRAMMER'S ASSISTANT
VERSION 1.0
(C) COPYRIGHT 1979, APPLE COMPUTER INC.
& RENUMBER (START), (INC), (FIRST), (LAST)
& HOLD
& MERGE
& COMPRESS
& SHOW
& NOSHOW
& AUTO (START), (INC)
& MANUAL
& XREF
& KEYS
```

屏幕上半部是版权说明,下半部是命令列示;下面,我们逐一介绍它们:

1)& RENUMBER 重排后第一行行号,重排后行号增量,欲重排程序始行号,欲排号程序末行号:

它是用来重编一个程序的行号的,在 DOS3.3 里,有一个 RENUMBER 程序,这个程序里有个致命的弱点,请看下面的程序(在调入 DOS3.3 的 RENUMBER 环境下运行):

```
10 A=A*10
20 PRINT A
30 END
]& F100
]& LIST
100 A=A*100
110 PRINT A
```

```
120 END
```

A=A \* 10 莫名其妙地变成了 A=A \* 100,这是由于 RENUMBER 设计的失误,错把“\*”号当成转移命令了。而在 APA 中的 &RENUMBER 就没有这个问题。请看程序 1、2。

```
程序 1: ]LIST
          10 A=A * 10
          20 PRINT A
          30 END
```

```
程序 2: ]& RENUMBER 100,10,10,30
          ]LIST
          100 A=A * 10
          110 PRINT A
          120 END
```

如省略其中一个参量,只需在两逗号之间什么字符也不打就可以了。如下例:

```
& RENUMBER 100,10,,
```

则与 & RENUMBER 100,10,0,63999 相同,省略值如下:

第一个参数:100      第二个参数:10  
第三个参数:0        第四个参数:63999

本命令只修改 GOTO, GOSUB, ON.....GOTO, ON.....GOSUB, ONERR.....GOTO 和 THEN 后面的行号, REM, LET, PRINT 后面的数字不予修改,另外如果有输入错误,或在程序中执行本命令,如下例:

```
]& RENUMBER 100,100,110,1000
```

这里,我们把新旧两个程序的行号交叉了,系统显示:

```
INTERLEAVED OR DUPLICATE LINE NUMBER
```

系统不会修改这个程序中的任何行号。

2)& AUTO 开始行号,行号增量

此命令将模拟 INT BASIC 的 AUTO 命令,自动产生行号,键入命令后,按下回车键,如果你紧接回车键按空格键,那么系统将显示开始行号,输完内容,按回车键,再紧接空格,系统又显示下一行号.....如果回车之后不紧接空格,那么系统还将按你的命令执行。如下例:

```
]& AUTO 10,10
] 10 HGR
] 20 HCOLOR=3
] 25 SCALE=1
] 30 DRAW 1 AT 140,80
] 35 FOR I=1 TO 1000:NEXT
] 40 XDRAW 1 AT 140,80
] 50 END
] LIST
10 HGR
20 HCOLOR=3
25 SCALE=1
30 DRAW 1 AT 140,80
35 FOR I=1 TO 1000:NEXT
40 XDRAW 1 AT 140,80
50 END
```

在 & AUTO 命令里,如省略第二个参数,系统按 10 计算,& AUTO 有一个奇怪的现象,当你键入 &ABC 后,也会与 &AUTO 产生同样的效果。这里,25、35 句前没有空格,因此,产生的不是整十的行号,而是用户自己输入的行号。

如果想取消 & AUTO 的功能,键入 & MAUAL 即可。

3)& MANUAL

执行这个命令后,系统退回原来的输入方式。

4)& XREF

列出这个程序中共用了哪些变量名及所在的行号。实型量直接用字母表示,整形量加“%”表示,字符串量加“\$”表示,数组加“(”(左括号)表示。如:

```
10 A=10
20 B%=20
30 C$="MY NAME IS C$"
40 ABC(3,4)=5
50 DEE$(9,5)="MY NAME IS DE$"
60 ZZZ%(2,3)=100
```

```
]& XREF
```

```
A 10
AB( 40
B% 20
C$ 30
DE$( 50
ZZ$( 60
```

这里,变量名按英文字母的次序排列显示时只取前两个字符;行号在变量名的右边。

5)& KEYS

早期的苹果机,键盘上没有下横线“-”,反斜杠“\”及左方括号“[”,结果,连自己的商标“APPLE”都产生不了。有了 & KEYS,我们可以用下面的键代替它们:

CTRL-O 产生“-”字符

CTRL-K 产生“[”字符

CTRL-L 产生“\”字符

如果想取消 & KEYS 的功能,键入 & MANUAL 即可。

6)& SHOW

该命令可显示控制字符,有些磁盘用控制字符加密文件名,也可用它解密,CTRL-字符的列印结果用反白显示。

7)& NOSHOW

恢复原来的显示模式,即取消 & SHOW 的效果。

8)& COMPRESS

有时,一个程序有许多 REM 语句,很占内存,为了节省内存,可用 & COMPRESS 命令清除 REM 语句。

9)& LENGTH

打印 BASIC 程序的长度。如:

```
]& LENGTH
PROGRAM LENGTH IS 131( $ 0083) BYTES
```

10)& HOLD

将程序移到可用最高地址 HIMEM 的正下方,修

改 HIMEM, 将其保护起来。

11) &. MERGE

恢复掩盖的程序。该命令可将两个程序连接起来:

]LIST

10 A=5

20 B=3

]&. HOLD

]LIST

]LOAD A-2

]LIST

30 C=4

40 D=7

50 PRINT A,B,C,D

]&. MERGE

]LIST

10 A=5

20 B=3

30 C=4

40 D=7

50 PRINT A,B,C,D

这些功能是怎么实现的呢?下面逐一分析它们,以便大家更好的利用。

(1) &. AUTO

本指令将 \$ 38, \$ 39 单元的值改至一个小程序的首地址。该子程序判断是否键盘上输入的是“回车+空格”,如是,显示第一个参数,同时把第一个参数值加上第二个参数值,等待用户输入,如此循环,便实现了上面所述的功能。

(2) &. MANUAL

本指令将 \$ 38, \$ 39 中的值换回原值(\$ FD1B),不做任何检查。所以,它有切掉 &. AUTO 和 &. KEYS 功能。

(3) &. RENUMBER

本指令从 \$ 803 单元开始读每一字节,首先,判断其值是否为 \$ AB(GOTO 代码), \$ B0(GOSUB), \$ A5(ONERR), \$ C4(THEN)之一,如是,参照其后的参数把这些地址后面两字节修正,如不是,地址加 \$ 1,继续反复,如此循环,改正行号的工作就做好了。

(4) &. HOLD

此指令用 MOVE 程序把 BASIC 程序从程序区搬到(HIMEM—程序长度)的区域,执行这个指令后,FP、MAXFILES、HIMEM:等指令将不能执行,否则,程序有可能丢失。

(5) &. MERGE

此指令将藏起的程序和内存程序区的程序结合起来,同时根据行号大小将程序重新安排,如遇到重复的行号,就提出警告。

(6) &. COMPRESS

本指令从程序区开始处一直找到 PRGEND (\$ AF、\$ B0 单元指示的地址),如遇到 \$ B2(REM),就从这个单元开始删,遇到 \$ 00(一行的程序结束的标志),或遇到 \$ 3A(“:”代码)为止。如已找完,把空白的

单元删除,然后将消去的总单元数目打印出来。

(7) &. LENGTH

打印 \$ AF、\$ B0 的值减去 \$ 67、\$ 68 的值。

(8) &. SHOW

类似 &. AUTO, &. SHOW 改动的是 CSW (\$ 36, \$ 37 单元)的值,改动后的子程序检查输入、输出的是否是控制字符(\$ 81~\$ 9F),如是,则执行 AND # \$ 3F,即显示相应键符的反相字。

(9) &. NOSHOW

把 CSW 的值改为 \$ FDF0。

(10) &. KEYS

本指令修改 KSW 单元(\$ 38, \$ 39)的值,检查键符并作如下的变换:

CTRL- 0: \$ 8F → \$ DF (“-”)

CTRL- K: \$ 8B → \$ DB (“[”)

CTRL- L: \$ 8C → \$ DC (“\”)

如执行 &. AUTO, 则 &. KEYS 失效,如 &. KEYS 先执行,则 &. AUTO 亦失效。也就是说, &. AUTO 与 &. KEYS 不能兼容。

(11) &. XREF

本程序从程序区开始扫描,遇到变量之形式的内存单元,就将其挪出来。许多读者对变量存放格式很感兴趣,在这里,顺便介绍一下:

A 实数变量 例: A=40000

| 单元组 | 内容含义      | 内容    |
|-----|-----------|-------|
| 0   | 变量名的第一个字元 | \$ 41 |
| 1   | 变量名的第二个字元 | \$ 00 |
| 2   | 指数部分      | \$ 90 |
| 3   | 小数部分      | \$ 1  |
| 4   |           | \$ 40 |
| 5   |           | \$ 00 |
| 6   |           | \$ 00 |

B. 整型变量 例: B%=257

| 单元组 | 内容含义       | 内容    |
|-----|------------|-------|
| 0   | 变量名第一、二个字元 | \$ C2 |
| 1   |            | \$ 80 |
| 2   | 变量值高位字节    | \$ 01 |
| 3   | 变量值低位字节    | \$ 01 |
| 4   | 未用         | \$ 00 |
| 5   |            | \$ 00 |
| 6   |            | \$ 00 |

C. 字符变量 例: CD\$ = "ABC"

| 单元组 | 内容含义               | 内容    |
|-----|--------------------|-------|
| 0   | 变量名第一字元(负 ASCII 码) | \$ 43 |
| 1   | 变量名第二字元(正 ASCII 码) | \$ C4 |
| 2   | 长度                 | \$ 03 |
| 3   | 字符串首地址低位字节         |       |
| 4   | 字符串首地址高位字节         |       |
| 5   | 未用                 | \$ 00 |
| 6   |                    | \$ 00 |

D. 整型数组变量 例 JK%(2,3)

| 单元组 | 内容含义              | 内容    |
|-----|-------------------|-------|
| 0   | 名称第一字元(负 ASCII 码) | \$ CA |
| 1   | 名称第二字元(负 ASCII 码) | \$ CB |



|   |         |       |   |         |        |
|---|---------|-------|---|---------|--------|
| 2 | 长度低位    | \$ 21 | 6 | 后一维长度低位 | \$ 04  |
| 3 | 长度高位    | \$ 00 | 7 | 前一维长度高位 | \$ 00  |
| 4 | 数组维数    | \$ 02 | 8 | 前一维长度低位 | \$ 03  |
| 5 | 后一维长度高位 | \$ 00 |   |         | (未完待续) |

## 利用 BOOT 程序对出租软件进行计次使用

无锡西潭压力容器厂技术科(214171) 任晓方

随着软件交流形式的多样化,出租软件也将成为一种使用形式。为了保护出租软件的权益,通常采用限时限次的方法来加以限制,这就产生了一个问题,即怎样才能确定使用次数。较为简单的方法是建立一个限时使用的计数文件,当其记录的使用次数达到额定的使用次数后,就停止其使用,或者是清除内存,或者干脆 INIT 磁盘,将其软件破坏掉,但是这种计次方法的计数文件容易被破解,也比较繁琐。

我通过分析 DOS 的引导过程,设计出了一种新的计次方法,即在引导 DOS 时就显示出该软件允许使用的次数,每次引导 DOS 时该使用次数自动递减,当使用完规定的次数后,DOS 就不能正常引导,而直接进入机内的 ROM BASIC 冷起动程序。

基本原理:我们知道 DOS 的引导过程是分三步进行的,第一步是执行软盘驱动器接口卡上的 ROM 引导程序,将软盘上的 0 道 0 扇区内容(即 BOOT1)读入内存,这一过程我们称做 BOOT0;然后执行 BOOT1 将软盘上的 0 道的 1-9 扇区内容(即 BOOT2)读入内存,再执行 BOOT2 将整个 DOS 读入内存,转入 DOS 的冷起动程序执行,至此 DOS 引导完毕。

在 DOS 中 \$BB00-\$BC55 为数据缓冲区,用来存放读/写磁盘的数据转换之用,位于磁盘上 0 道第 5 扇区和第 6 扇区的一部分,这一部分扇区的内容在引导 DOS BOOT1 阶段时不起任何作用,但仍被读入内存,因此我们可以在这一部分扇区内设计一段软件计次使用程序,在 BOOT1 完后执行该段程序,将使用次数单元值自动减 1,并利用已读入的 RWTS 程序修改磁道上的计次单元值。若计数单元值不等于 0,则转入 BOOT2 继续引导 DOS,否则转入 BASIC 冷起动程序执行,DOS 失效。

具体改造和使用方法如下:首先引导 DOS 格式化一张从盘,再用 COPY ][ 工具软件或 LOCKSMITH 等其他能改写磁盘扇区的工具软件,将 0 道 0 扇区和第 5 扇区按下表划线部分修改后仍存回原扇区,全部修改完毕后这张盘就有了上述计数功能了,第 5 扇区的 \$DF 字节是设定的计次数,本例中为 \$12,使用次数为 \$DF 单元的值减 1,如 (\$DF)=\$12,则可以使用 17 次。当其值为 1 时,则不能引导 DOS,转入机内的

BASIC 冷起动程序执行,从而使得出租软件得到保护。在程序中我另外在 \$1B00-\$1C55 开辟了一段数据缓冲区,以保护位于 \$BB00-\$BBFF 中的程序不被破坏,DOS 引导完后位于 \$BB00-\$BBFF 中的程序自动失效,该区域仍为 DOS 的数据缓冲区,用此盘引导的 DOS 格式化从盘则无上述功能,同正常 DOS 一样。

由于本程序只占用数据缓冲区所在扇区,所以对其他修改后的 DOS 从盘仍适用。另外,数据缓冲区共有 342 个字节,有较大的空间,可以设计出其他利用 DOS 的方法,如还可以设计一段程序进一步将使用次数已到的磁盘进行格式化,从而彻底保护出租软件,此加密方法同其他加密方法结合起来使用,能起到绝妙的效果。有兴趣的同志不妨试一试。

```

TRACK=0 SECTOR=0
00: 01A5 27C9 09D0 18A5 2B4A 4A4A 4A09 C085
10: 3FA9 5C85 3E18 ADFE 086D FF08 8DFE 08AE
20: FF08 3015 BD4D 0885 3DCE FF08 ADFE 0885
30: 27CE FE08 A62B 6C3E 00EE FE08 EEFE 0820
40: 89FE 2093 FE20 2FFB A62B 4C00 BB00 0D0B
50: 0907 0503 010E 0C0A 806B 0402 0F00 2064
60: A7B0 08A9 00A8 8D5D B691 40AD C5B5 4CD2
70: A6AD 5DB6 F008 EEBD B5D0 03EE BEB5 A900
80: 8D5D B64C 46A5 8DBC B520 A8A6 20EA A24C
90: 7DA2 A013 B142 D014 C8C0 17D0 F7A0 19B1
A0: 4299 A4B5 C8C0 1DD0 F64C BCA6 A2FF 8E5D
B0: B6D0 F600 0000 0000 0000 0000 0000 0000
C0: 0000 0000 0000 0000 0000 0000 0600 0000
D0: 2058 FCA9 C220 EDFD A901 20DA FDA9 AD20
E0: EDFD A900 20DA FD60 0000 0000 0000 0000
F0: 0000 0000 0000 0000 0000 0000 0000 B609
TRACK=0 SECTOR=5
00: 2058 FCA9 0820 5BFB A900 8524 CEDF BBD0
10: 1020 B8BB A225 20D0 BBA2 60BD 88C0 4C00
20: E020 B8BB A91C A01B 20E0 BBA9 B7A0 E884
30: 4885 49A9 00A0 0491 48A9 05C8 9148 A902
40: A00C 9148 A900 A008 9148 C8A9 BB91 4820
50: 04BD A9BC A0BB 20E0 BBA2 60A9 6C8D 4AB6
60: A9FD 8D4B B6A9 088D 4CB6 4C00 B760 0000
70: D4E8 E9F3 A0F0 F2EF E7F2 E1ED A0E9 F3A0

```

|                                                                                                     |                                                                                                     |
|-----------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| 80: <u>E1EC</u> <u>ECEF</u> <u>F7E5</u> <u>E4A0</u> <u>F5F3</u> <u>E5A0</u> <u>00A0</u> <u>F4E9</u> | C0: <u>DFBB</u> <u>2024</u> <u>EDA2</u> <u>1D20</u> <u>D0BB</u> <u>6000</u> <u>0000</u> <u>0000</u> |
| 90: <u>EDE5</u> <u>F3A1</u> <u>008D</u> <u>8D8D</u> <u>8787</u> <u>8787</u> <u>A0A0</u> <u>A0A0</u> | D0: <u>BD70</u> <u>BBF0</u> <u>0620</u> <u>F0FD</u> <u>E8D0</u> <u>F560</u> <u>0000</u> <u>0012</u> |
| A0: <u>A0A0</u> <u>C9EC</u> <u>ECE5</u> <u>E7E9</u> <u>F4E9</u> <u>EDE1</u> <u>F4E5</u> <u>A0F5</u> | E0: <u>8D0A</u> <u>B88D</u> <u>0EB8</u> <u>8D20</u> <u>B88D</u> <u>25B8</u> <u>8D3A</u> <u>B88D</u> |
| B0: <u>F3E5</u> <u>A100</u> <u>0000</u> <u>0000</u> <u>A200</u> <u>20D0</u> <u>BBA9</u> <u>00AE</u> | F0: <u>68B8</u> <u>8C11</u> <u>B88C</u> <u>6BB8</u> <u>8C80</u> <u>B88C</u> <u>90B8</u> <u>6000</u> |

## 中华学习机汉字双页显示

安徽蚌埠医学院(233003) 蒋建一

中华学习机是具有汉字处理功能的计算机。它提供了汉字的输入、显示及打印等多项功能,为用户编制具有汉字处理功能的程序带来了方便,但它只能在高分辨率图形第二页显示汉字,而不能随意选用第一页或第二页,使得其汉字的显示有一定的局限性。

通过应用和分析学习机汉字系统程序,我编制了一个可实现汉字双页显示随意选择的汇编语言程序。程序使用说明:首先按所附程序清单键入全部机器码程序并存盘保存。以后只要BRUN该程序,便可自动进入汉字系统,并选用高分辨率第一页作为汉字显示页。

1C00- A9 00 8D 7B 04 20 00 C3  
 1C08- A9 00 85 06 A9 EC 85 07  
 1C10- A0 00 20 AB C3 B1 06 20  
 1C18- B9 C3 48 AD 83 C0 AD 83  
 1C20- C0 68 91 06 8D 82 C0 8D  
 1C28- 82 C0 C8 D0 E5 E6 07 D0  
 1C30- DF AD 81 C0 AD 81 C0 A9  
 1C38- 05 8D F4 F1 A9 E6 8D F5  
 1C40- F1 A2 6F CA F0 13 BD FF  
 1C48- 1C 85 06 CA BD FF 1C 85  
 1C50- 07 A0 00 A9 1F 91 06 D0  
 1C58- EA A9 1D 8D 88 F2 2C 82  
 1C60- C0 2C 82 C0 A9 20 85 E6  
 1C68- 20 F2 F3 A9 91 85 3C A9  
 1C70- 1C 85 3D A9 FF 85 3E A9  
 1C78- 1C 85 3F A9 00 85 42 8D

1C80- F6 03 A9 03 85 43 8D F7  
 1C88- 03 A0 00 20 2C FE 4C 14  
 1C90- 03 C9 31 F0 10 C9 32 F0  
 1C98- 4B 4C 76 DD 60 A9 20 85  
 1CA0- E6 2C 54 C0 60 20 0C 03  
 1CA8- 4C 5D 03 2C 52 C0 2C 57  
 1CB0- C0 A9 96 85 74 A9 33 85  
 1CB8- 36 A9 43 85 38 A9 03 85  
 1CC0- 37 85 39 60 8D 80 C0 8D  
 1CC8- 80 C0 20 CE EF 8D 82 C0  
 1CD0- 8D 82 C0 60 8D 80 C0 8D  
 1CD8- 80 C0 20 A3 F7 8D 82 C0  
 1CE0- 8D 82 C0 60 A9 40 85 E6  
 1CE8- 2C 55 C0 4C 17 03 2C 50  
 1CF0- C0 68 68 A9 E0 48 A9 02  
 1CF8- 48 4C 1A 03 00 00 00 00

1D00- ED 7A F0 11 F0 1B F0 26  
 1D08- F0 D4 F1 88 F3 64 F3 67  
 1D10- F3 E0 F3 E5 F4 1D F4 33  
 1D18- F4 6C F4 71 F4 76 F4 80  
 1D20- F4 85 F4 8F F4 98 F4 9F  
 1D28- F4 AA F4 AF F4 B2 F4 B7  
 1D30- F4 BE F4 C8 F4 D2 F4 E5  
 1D38- F4 EC F4 F3 F4 F8 F5 03  
 1D40- F5 08 F5 0B G5 0E G5 13  
 1D48- F5 16 F5 1B F5 2B F5 35  
 1D50- F5 4F F5 5A F5 98 F5 9F  
 1D58- F6 01 F6 61 F6 66 F6 AA  
 1D60- F6 B8 F6 DE F7 02 F7 10  
 1D68- F7 36 F7 63 F7 7E 00 00

若想换到第二页显示,只要键入&2/便会立即转为第二页显示。(注:汉字显示页转换命令为&1选择第一页;&2选择第二页)。无论何时,只要键入&1或者&2就能方便地实现汉字显示页的转换。本程序只要运行一次后,其内存中所占空间便可供再使用,不需保留。另外,本程序将汉字显示映射区改在\$1D00~\$1FFF。当选用汉字第一页显示时,用户空间为\$800~\$1CFF,和\$4000~\$9600。

### 敬告《电子与电脑》读者

今后凡读者来函、来款均请按下列地址及收件人邮寄:邮编100036 北京173信箱 杂志收,并注明读者当地详细地址及邮编,字迹要清楚。

# 高速排序程序的使用技巧

山东淄博五中 33 级 1 班(255028) 张世栋

在诸多排序方法中,笔者认为使用汇编语言编写的高速排序程序(以下简称高速排序程序)优点最大:

1. 使用简便。在你编写的 BASIC 程序中,只要有一句调用高速排序程序的命令,就可对指定的数组排序。2. 排序速度快。例如这篇文章中引用的高速排序,对具有 1000 个元素的一维实型数组排序,仅需 5 秒钟!

以上两条优点就大大方便了对数据处理程序的编写及调试,下面就以《中华学习机》杂志 1990 年第 4 期刊登的汇编程序在考分统计中的应用为例,说明高速排序程序的使用技巧。

先简要介绍一下高速排序程序的使用方法:首先运行高速排序程序,然后在你所编的程序中加入:& 数组变量名(a),b。这样就可对指定的一维数组中,第 a 至 a+b 项,共 b+1 个数据进行排序。(注:高速排序程序及示范程序见附录一,其中示范程序的第 10 句及第 15 句是运行已存在磁盘中的高速排序程序,其文件名为:PMC'HART NEW)。

当您把附录中的高速排序程序输入计算机,存入磁盘就可用下面介绍的两种方法,得心应手的使用它了。

## 方法之一:二次排序法

这种方法的设计思想是,把所要排序的数据,也就是各科考试的分数放入 2 个数组(这两个数组所放数据完全一样),再对其中一个数组进行快速排序,然后根据已排好次序的数组中各个数据在原数组中的位置,找出每个数据所对应的其它相关数据,如学号、姓名、性别等等。程序一就是根据这个设计思想编写的。它的功能就是对 N 名学生,Q 个学科的总成绩排序,并打印出名次表。

程序如下:

### 程序一:

```
10 INPUT N,Q:N=N-1:D$=CHR$(4)
20 DIM A(N),S(N),Y%(N)
30 FOR I=0 TO N:READ Y%(I)
40 FOR J=1 TO Q:READ A:A(I)=A(I)+A: NEXT J
50 S(I)=A(I): NEXT I
60 PRINT D$;"BRUN PMC'HART 2.0";& A(0),N
80 FOR I=0 TO N;V=0:PRINT I+1;:FOR J=0 TO Q
90 IF A(I) < >S(J) THEN I10
100 V=V+1:PRINT TAB(7);Y%(J);TAB(14);INT(S(J)/Q*100+.5)/100;TAB(23);A(I)
110 NEXT J:IF V>1 THEN I=I+(V-1)
```

```
120 NEXT I
200 DATA 1,89,99,78,2,78,65,67,3,88,99,100,4,57,86,97,5,78,65,67
```

]RUN

? 5,3

|   |   |       |     |
|---|---|-------|-----|
| 1 | 3 | 95.67 | 287 |
| 2 | 1 | 88.67 | 266 |
| 3 | 4 | 80    | 240 |
| 4 | 2 | 70    | 210 |
|   | 5 | 70    | 210 |

程序一的几点说明:

1. 10~20 句是输入人数(N)和科目数(Q),并定义数组大小。

2. 30~50 句是读数据,把学号放入 Y%(N)中,每个学生的总分放入 A(N),S(N)。

3. 60 句运行排序程序对指定的 A(N)数组排序。

4. 80~120 句是第二次排序(对学号等其它相关数据)并打印名次表。变量 V 是用来避免名次并列时的重复打印。这种形式决定了名次并列时,学号小的在学号大的前面(见运行结果中的第 4 名)。

5. 200 句的数据存放格式为:学号在前,各科成绩依次在后。

6. 运行结果的 4 个纵行数据自左向右分别为:名次、学号、平均分、总分。

这种方法的优点在于程序简单,明白。但美中不足的是当人数一多,第二次排序时间必然会增长,如果您有一台高速打印机,就会出现打印机等主机的不正常现象。使用方法二,这种现象将根本消除。

## 方法之二:编码排序法

这种方法的设计思想是把所要排序的数(在此是考试总分)放大一定倍数,使它成为一个整数,把代表它在数组中的位置的数缩小一定倍数加在整数后面,成为整数的小数部分。再对这个“新”的数进行排序,排序完毕后再根据这个“新”数的整数与小数部分,求出实际考试总分,及对应的其它相关数据的位置(即这个新数在数组中未排序之前的位置),从而打印出名次表。这种设计思想是依据高速排序程序在排序时,总是从高位开始,高位数字大的必然大这一特点设计的。这样虽然各个数据在数组中位置不一样,但是把它们的位置作为低位数字加在各个数据后,对排序结果是毫无影响的。但必须注意的是:原来的数一定要扩大成为整数,代表位置的数一定要缩小为小数(即零点几)。这样相加所组成的“新”数才具有意义。这个过程很象编码,解码,因此我们就不妨称之为“编码排序”。程序

二就是根据这个设计思想编写的。它能对全年级学生排名次,并据各班前 50 名的总分,作为班级成绩,进行排序。(注:这个“前 50 名”是可以任意改变的),程序如下:

**程序二:**

```

5  D$ = CHR$(4);INPUT "How many students? ";N;
   N = N-1; INPUT "How many classes?";Q;DIM
   A(N),Y%(N),S(Q);Y1 = 10;Y2 = 10000
10  FOR I = 0 TO N;READ Y%(I),A(I);IF Y%(I)<
   1000 OR A(I) > 1000 THEN PRINT I + 1,Y%(I
   + 1),A(I+1);STOP
20  X = VAL(RIGHT$(STR$(Y%(I)),2));Y =
   VAL(MID$(STR$(Y%(I)),2,1));IF X<
   = 50 THEN S(Y) = S(Y) + A(I);R(Y) = R
   (Y) + 1
30  A(I) = A(I) * Y1 + I / Y2;NEXT ;FOR I=0 TO
   Q-1;S(I) = S(I + 1) * 100 + I;NEXT ;PRINT
   ; PRINT D$ "BRUN PMC'HART 2. 0";& A(0),N;
   & S(0),Q
35  INPUT "Hard copy? (Y/N) ";A$; IF A$ = "Y"
   THEN PRINT ; PRINT D$;"PR #1"; PRINT
40  PRINT " ";; FOR I = 0 TO N;J = INT(A(I) *
   Y2 - INT(A(I)) * Y2 + .5);A(I) = INT(A
   (I))/Y1; IF I > 0 THEN IF A(I) < > A(I-1)
   THEN PRINT " ";I + 1;
50  PRINT TAB(8)Y%(J) TAB(17)A(I); NEXT ;
   PRINT ;PRINT
55  PRINT ;PRINT " ";; FOR I = 0 TO Q-1;J =
   VAL(RIGHT$(STR$(S(I)),1));S(I) = (S
   (I)-J)/100; IF I > 0 THEN IF S(I) < > S(I-
   1) THEN PRINT " ";I + 1;
60  PRINT TAB(8);J + 1 TAB(14)S(I); TAB(27)
   INT(S(I) / R(J + 1) * 100 + .5) / 100; TAB(
   37)R(J + 1); NEXT ;PRINT ;PRINT D$;"PR #
   0"
200 DATA 1102,56,1102,89,1103,98,1201,98,
   1202,98,1203,90
] RUN
How many students ? 6
How many classes ? 2
Hard copy
? (Y/N) Y
1      1202      98
        1201      98
        1103      98
4      1203      90
5      1102      89
6      1101      56

1      2          286          95.33      3
2      1          243          81          3

```

程序二的几点说明:

1. 200句的数据存放格式为学号在前总分在后。学号的构成为:ABXX。A 代表年级,B 代表班级,XX 代表班级名次。例如,高一年级一班第 8 名,其学号就是

1108。

2. 第 5 句是输入年级总人数 N,班级数 Q,A(N)放总分,Y%(N)放学号,S(Q)放各班前 50 名总分。变量 Y1、Y2 是在编码,解码时用。

3. 第 10 句中的条件语句,是判断所读数据有无错误,如有错误则立即打印出错误位置及总分、学号,并暂停。

4. 变量 X、Y 是根据学号构成特点,从学号中提取的名次及班级。

5. 第 30 句是对 A(N),S(Q)编码,并排序。

6. 第 40~50 句是对 A(N)解码,同时打印年级名次表。这种排序方式决定了名次并列时,学号大的在学号小的前面。

7. 第 55~60 句是对 S(Q)解码,同时打印班级名次表。

8. 运行结果中三个纵行自左向右分别为名次、学号、成绩。下面的五个纵行自左向右分别为名次、班级、总分、平均分、人数。

应注意的是解码过程中,由于 APPLE 机的运算误差,因此解码不能是编码运算式的简单逆运算过程,请读者着重看一下程序二中的第 40 句、第 50 句的解码运算式。

当您掌握以上两种方法的原理,并应用到自己所编的程序上,您会发现:原来需要运行 1 个多小时,甚至几个小时的程序,现在只需十几分钟,并且还可以在一定程度缓解内存紧张问题。

如果有的读者想把这两种排序方法运用到中华学习机上,在程序中加入汉字,那么您可以用附录二中的汇编程序。(摘自《学生计算机世界》)。这是因为附录一中的汇编程序的结尾一部分,占用了中华学习机的汉字管理程序的工作单元,在汉字状态下运行程序会出现“死机”现象。

程序选自《学生计算机世界》

**附录一:**

```

0300- A9 0B 8D F6 03 A9 03 8D
0308- F7 03 60 20 E3 DF 84 3D
0310- 85 3C 20 BE DE 20 67 DD
0318- 20 52 E7 A5 50 85 3E 85
0320- 40 A5 51 85 3F 85 41 A7
0328- 41 87 3A 87 3B A5 40 46
0330- 41 6A A8 05 41 F0 D3 98
0338- 85 40 0A 0A 86 FB 26 FB
0340- 65 40 85 FA A5 FB 65 41
0348- 85 FB A5 3E 38 E5 40 85
0350- 42 A5 3F E5 41 85 43 A5
0358- 3B A4 3A 85 4E 84 4F 85
0360- FF 98 0A 26 FF 0A 26 FF
0368- 65 4F AA A5 FF 65 4E A8
0370- 8A 65 3C AA 98 65 3D A8
0378- 8A 20 F9 EA 18 A5 5E 65
0380- FA 85 60 A5 5F 65 FB 85
0388- 61 20 B6 EB B0 12 E6 3A

```

```

0390- D0 02 E6 3B A5 42 C5 3A
0398- A5 43 E5 3B 90 89 B0 B7
03A0- A0 04 B3 5E B1 60 91 5E
03A8- 8A 91 60 88 10 F4 A5 4F
03B0- E5 40 A8 A5 4E E5 41 90
03B8- D5 D0 A0 A5 4F C5 40 90
03C0- CD A9 00 B0 96
10 PRINT ;D$ = CHR$ (4); PRINT
15 PRINT D$;"BRUN PMC'HART NEW"
20 INPUT N;DIM A(N); FOR I = 1 TO N;A(I) =
  RND (1) * 1000;PRINT A(I);NEXT
25 &.A(0),N
27 PRINT ; PRINT
30 FOR I = 0 TO N-1; PRINT A(I); NEXT
] RUN
?5
360. 889731
628. 262312
581. 121379
768. 268873
723. 68035

```

```

768. 268873
723. 68035
628. 262312
581. 121379
360. 889731

```

### 附录二

```

8FF0- A9 4C 8D F5 03 A9 00 8D
8FF8- F6 03 A9 90 8D F7 03 60
9000- 20 E3 DF 85 EC 85 EE 84
9008- ED 84 EF 38 E9 02 85 85
9010- B0 01 88 84 86 A0 01 B1
9018- 85 85 FE 88 B1 85 85 FF
9020- A9 01 8D B0 90 A9 00 8D
9028- B1 90 F0 11 18 A5 EC 69
9030- 05 85 EC 85 EE A5 ED 69
9038- 00 85 ED 85 EF A5 EC A4
9040- ED 20 F9 EA AE B0 90 AD
9048- B1 90 8D B2 90 20 7E 90
9050- E8 D0 03 EE B2 90 E4 FE
9058- D0 F3 AD B2 90 C5 EF D0
9060- EC 18 AD B0 90 69 01 8D
9068- B0 90 D0 03 EE B1 90 AD
9070- B0 90 C5 FE D0 B6 AD B1
9078- 90 C5 FF D0 AF 60 18 A5
9080- EE 69 05 85 EE 90 02 E6
9088- EF A5 EE A4 EF 86 FA 20
9090- B2 EB A6 FA C9 FF F0 01
9098- 60 A0 04 B1 EC 48 B1 EE
90A0- 91 EC 68 91 EE 88 10 F3
90A8- A5 EC A4 ED 4C F9 EA
10 INPUT N; DIM A(N)
20 FOR I = 0 TO N;A(I) = RND (1) * 1000;PRINT

```

```

A(I); NEXT ; PRINT
50 CALL 198;&.A(0); CALL 198
60 FOR I = 1 TO N;PRINT A(I); NEXT
] RUN
?7
594. 331536
130. 726507
834. 264561
567. 285335
920. 549918
464. 319447
829. 422036
571. 668958
834. 264561
829. 422036
594. 331536
571. 668958
567. 285335
464. 319447
130. 726507

```

## 汉字编码打印程序

宝鸡市金陵路21号(721001)李 铁

中华学习机扩展操作系统 V2.11 以上版本可以使用五笔,表形,苍吉,首尾,电报等十余种编码输入汉字。但是,在以前发行的软盘中,没有专门的编码打印程序,对于初学汉字编码输入汉字的用户很不方便。为此,特编了下面这个汉字编码打印程序,使用如下:

启动 CEC-I SC V2.5 系统,调入需要打印的编码,然后执行本程序,便可为你打印一份码表。程序中,U 为每行字数,VV \* K 为每页 600 字。ADD 为码表参数地址,V2.11 版改为 \$ACA7。

```

10 REM PRINT CHINESE CODE
20 IO#2;PRINT ; HOME ; PRINT
30 U = 10;VV = 6;K = 100
40 ADD = 9 * 4096 + 15 * 256 + 10 * 16 + 4
50 DEF FN A(X)=X+28+(X>5)+(X>14)+(X>
  27)
60 D= PEEK(ADD+10);Z=64;S=160
70 L= PEEK(ADD+9)
80 I= PEEK(ADD+14)-159+(PEEK(ADD+15)-
  160)*100
90 T = PEEK(ADD+12)+PEEK(ADD+13)*256+
  D
100 BL=2^PEEK(ADD+11)
110 M$="<<";FOR E=0 TO 5;S IF PEEK(ADD+E)
  <128 THEN M$=M$+CHR$(PEEK(ADD+
  E));ELSE;M$=M$+CHR$(FN A(PEEK

```

```

(ADD+E)-160))
120 NEXT E;M$=M$+"码>>"
130 POKE 1659,1;PRINT
140 FOR A=1 TO I+4096
150 X=INT(A/K);E=FN A(X)

160 X=A-X*K;F=FN A(X)
170 IF X<>95 THEN 230
180 PRINT ;A=A+5;S1=S1+1; IF S1<>VV
    THEN 380
190 S1=0;S2=S2+1
200 GOSUB 400
210 POKE 1659,0;INPUT A$;POKE 1659,1
220 GOTO 380
230 X=(A-1)*D
240 IF X/U=INT(X/U)THEN PRINT;
    PRINT I+X/D;"-";
250 PRINT CHR$(127);CHR$(E);CHR$(F);
260 X=X+T;F=0
270 FOR E=D-1 TO 0 STEP -1
280 F=F*256+PEEK(X+E)
290 NEXT E
300 A$=" ";FOR E=1 TO L
310 X= INT(F/BL^(L-E))
320 S=PEEK(ADD+17+X)
330 IF S=0 THEN A$=A$+" "
340 PRINT CHR$(S);
350 F=F-BL^(L-E)*X
360 NEXT E
370 PRINT A$;
380 NEXT A
390 GOSUB 400;POKE 1659,0;END
400 PRINT ;PRINT ;PRINT
410 PRINT SPC(20);M$;SPC(10);"区位:";A-VV
    *100;"-";A-1;SPC(5);"第";S2;"页"
420 RETURN

```

## 对《CEC—I 键控光 标作图程序》的改进

北京永外建筑磨石二车间(100075)李庆岱

1991年第12期《电子与电脑》“学习机之友”专栏发表的陈国良同志的《CEC—I 键控光标作图程序》，很有创意，为计算机绘图爱好者提供了一个非常好的工具。本人在学习该程序中，发现了几点可改进之处。

首先，原程序没有控制键用途说明，用时颇觉不便。为免去时时查书之苦，我试编了一个说明模块，见所附程序700—745语句。

其次，大多数计算机绘图爱好者，都有保存满意图形，供以后调用，欣赏的愿望。原程序无此功能，颇觉遗憾。我试加了一个存盘模块，见400—440语句。

再次，由于使用方向不同，我们往往调用盘中现有图形，经改进后加以利用。如能在同程序中调用已有的同系统下的图形，将是非常便利的。我尝试给原程序加了一个调盘模块，见500—520语句。它可以在调出图形后显示出来，并进入绘图状态，以利修改、欣赏。

最后，原程序无结束运行功能，只能按“CTRL—RESET”强行中断，似觉不完整。我为它增加了600句，能在结束时，问候使用者，亲切而方便。

几个模块的转向控制，利用了原程序语句行号的空置间隔，略增了几句。

为节约篇幅，原程序不再给出，详细增补程序如下：

```

161 IF P=211 THEN 400
162 IF P=193 THEN 500
163 IF P=197 THEN 600
164 IF P=212 THEN 700
.....
400 TEXT;HOME;VTAB 12;HTAB 3;PRINT"GIVE A
    NAME FOR THE PICTURE TO SAVE;"
405 INPUT M$;FLASH
410 HOME;VTAB 15;HTAB 20;PRINT"SAVING..."
415 NORMAL
420 PRINT CHR$(4)"BSAVE";M$;" ,A$ 4000,L
    $ 2000"
425 VTAB 20;HTAB 20;PRINT"GOOD!"
430 FOR I=1 TO 1500;NEXT
435 POKE -16304,0;POKE -16299,0;POKE -16297,0
440 GOTO 60
500 TEXT;HOME;VTAB 22;HTAB 10;PRINT"GIVE
    A NAME FOR LOADING;"
505 INPUT N$
510 POKE -16304,0;POKE -16299,0;POKE -16297,0
515 PRINT CHR$(4)"BLOAD";N$
520 GOTO 60
600 TEXT;HOME;VTAB 22;HTAB 15;PRINT
    "GOOD-BYE!!";
    END
700 TEXT;HOME;PRINT
705 PRINT "U=LEFT UP; I=UP; 0=RIGHT UP"
710 PRINT;PRINT"J=LEFT; K=RIGHT;"
715 PRINT;PRINT"N=LEFT DOWN; M=DOWN; ,
    =RIGHT DOWN;"
720 PRINT;PRINT"Q=DRAW; W=XDRAW; ESC=
    CLEAR;"
725 PRINT;PRINT"0→7=CHANGE COLOR;"
730 PRINT;PRINT"A=LOAD; S=SAVE; E=QUIT;"
735 PRINT;PRINT"《HIT 'SPACE-BAR' TO
    RETURN》"
740 PRINT
750 GET E$;IF E$=" " THEN 435

```

原程序增加以上语句后，将多占用3个扇区的磁盘空间，但与增加的功能比较，还是划算的。

## 第六讲 简单程序和分支程序设计

南京大学大气科学(210008) 朱国江

从本讲开始介绍机器语言程序设计的方法和技巧,包括简单程序设计、分支程序设计、循环程序设计、子程序设计、堆栈程序设计及调用监控子程序等六个方面。

虽然我们分节介绍这六个方面内容,但实际上,一个实用程序常常是六个方面的有机结合。

简单程序是复杂程序的基础,也是初学者程序设计入门的必经之路。分支程序是循环程序的基础,用以解决各种各样的分支转移问题。循环程序完成大量的在处理形式上完全相同的重复性计算工作,是机器语言程序设计的核心和精华。子程序用以处理在功能上具有一定独立性,能够完成相同计算和操作,并被经常调用的程序段,用以简化设计并使结构清晰。堆栈是计算机信息处理中的重要概念,是保护和恢复现场的主要手段。监控子程序简短独立、功能完整、结构合理、简炼灵活,在程序设计中,合理调用能起到事半功倍的效果。

### § 1. 简单程序设计

简单程序一般是指整个程序自始至终按指令顺序逐条执行,而在执行过程中不会出现分支或转移,因而又称直线程序。

简单程序设计方法虽然比较单一,但包含的内容还是比较丰富的,即包括数据输入、进行计算、输出结果、结束停机等方面内容。也就是说它包含了任何一个程序设计中不可缺少的每一部份,反映了程序设计的结构和全貌。

简单程序仅能完成一些简单操作,一般不单独使用。而且,任何一个实际程序绝非单纯由简单程序组成,但初学编程从简单程序入手实为必要,因为它是构成复杂程序的基础,又是初学者入门的必经之道。

现在,举一些简单程序的设计实例。

#### 〔例1〕数据块转移

将起始地址 \$ 2000—\$ 2004 的5个字节内容转移到 \$ 4000—\$ 4004 单元中去。

由于转移的数据较少,问题十分简单,直接写出程序6.1。

#### 程序6.1:

```
1000- AD 00 20   LDA   $ 2000
1003- 8D 00 40   STA   $ 4000
1006- AD 01 20   LDA   $ 2001
1009- 8D 01 40   STA   $ 4001
100C- AD 02 20   LDA   $ 2002
100F- 8D 02 40   STA   $ 4002
```

```
1012- AD 03 20   LDA   $ 2003
1015- 8D 03 40   STA   $ 4003
1018- AD 04 20   LDA   $ 2004
101B- 8D 04 40   STA   $ 4004
101E- 60         RTS
运行前:2000-0A 0B 0C 0D 0E
        4000-00 00 00 00 00
运行后: * 1000G✓
        * 4000-4004✓
        4000-0A 0B 0C 0D 0E
```

程序6.1用的是取数和送数两条指令,反复进行同样的操作共5次。显然,若要传送250个字节的数据,则程序将要写上500个语句,这不仅没有必要,而且几乎也是不可能的,因此,对于大量数据的传送,直接程序有很大的局限性。对于少量数据的传送,程序6.1既简单又实用,复杂程序中的数据传送常采用类似的结构。

#### 〔例2〕数据交换

将 \$ 06 单元和 \$ 08 单元中的内容互换。设原来 (06) = AA, (08) = BB。交换后应有 (06) = BB, (08) = AA。

对这类简单问题设计方法很多,这里借助于一个暂存单元 \$ 07,做中间寄存器来互换。见程序6.2。

#### 程序6.2

```
1000- A5 06     LDA   $ 06
1002- 85 07     STA   $ 07
1004- A5 08     LDA   $ 08
1006- 85 06     STA   $ 06
1008- A5 07     LDA   $ 07
100A- 85 08     STA   $ 08
100C- 60       RTS
运行前: * 6:AA✓
        * 8:BB✓
运行后: * 1000G✓
        * 6✓
        0006-BB
        * 8✓
        0008-AA
```

分析程序6.2时,应紧紧抓住这样一个事实:数据从源地址传送到目的地址,目的地址的内容被源地址内容更新,而源地址内容不变。这也是分析一切传送指令的要领。

#### 〔例3〕求反码

设原码为 \$ 6A,设计一个程序求出它的反码。结果为 \$ 95。

6502指令系统中没有求反码的指令,但可以将一

个8位二进制数与 \$FF 进行异或处理,异或结果即为该8位二进制数的反。

即:  $6A = 01101010$   
 $+ FF = 11111111$   
 $\hline 95 \quad 10010101$

故有程序6.3。

程序6.3:

```
1000- A9 6A   LDA   # $6A
1002- 49 FF   EOR   # $FF
1004- 20 DA FD JSR   $FDDA
1007- 60      RTS
```

运行: \* 1000G ✓

95

〔例4〕求补码

设原码为 \$09,设计一个程序求出它的补码。结果为 \$F7。

6502指令系统中也没有求补码的指令,但可根据原码取反加1就是补码的定义,容易写出程序6.4。

程序6.4:

```
1000- A9 09   LDA   # $09
1002- 49 FF   EOR   # $FF
1004- D8      CLD
1005- 18      CLC
1006- 69 01   ADC   # $01
1008- 20 DA FD JSR   $FDDA
100B- 60      RTS
```

运行: \* 1000 G ✓

\* F7

〔例5〕拆字

拆字又叫字的分离或字的分解。

设原字为 A3(10100011),将其分成两段,每段四位,高4位放入 \$6001单元中低4位位置上,低4位放入 \$6000单元低4位位置上,并使 \$6001和 \$6000两个单元的高4位全部清零。

即:  $(6000) = 00000011 = 03$

$(6001) = 00001010 = 0A$

解本题的思路可分为二步:第一步将原字 A3取出来,屏蔽高四位(AND # \$0F),留下低4位(0011),存入 \$6000单元中;第二步将原字 A3再一次调出来,用4次逻辑右移一位移令 LSR,这样原高四位的值 A(1010)全部移入低4位,而高4位变0,再存入 \$6001单元。故有程序6.5。

程序6.5:

```
1000- A9 A3   LDA   # $A3
1002- 29 0F   AND   # $0F
1004- 8D 00 60 STA   $6000
1007- A9 A3   LDA   # $A3
1009- 4A      LSR
100A- 4A      LSR
100B- 4A      LSR
100C- 4A      LSR
100D- 8D 01 60 STA   $6001
1010- 60      RTS
```

运行: \* 1000G ✓

\* 6000 ✓  
 \* 6000-03  
 \* 6001 ✓  
 \* 6001-0A

〔例6〕组字

组字又称语句组合,字组合。例如,将 \$01和 \$02单元中的低4位组成一个字,放在 \$03单元中。其中 \$01单元的低4位放入 \$03单元高4位; \$02单元的低4位放入 \$03单元低4位。

即:  $(01) = 6A$

$(02) = 43$

$(03) = A3$

设计本题程序的要领是先拆字后组字,见程序6.6。

程序6.6:

```
0300-A9 6A   LDA   # $6A
0302-85 01   STA   $01 } (01)=6A
0304-A9 43   LDA   # $43
0306-85 02   STA   $02 } (02)=43
0308-A5 02   LDA   $02取43
030A-29 0F   AND   # $0F屏蔽高四位
030C-85 03   STA   $03存数(03)=03
030E-A5 01   LDA   $01取6A
0310-0A      ASL
0311-0A      ASL }左移四位
0312-0A      ASL
0313-0A      ASL
0314-85 04   STA   $04 存数(04)=A0
0316-18      CLC   清进位,0→C
0317-A5 03   LDA   $03 取03
0319-65 04   ADC   $04 加A0
031B-85 03   STA   $03 存结果A3
031D-60      RTS   结束
```

非常有趣的是,程序6.6中的 ADC \$04指令,可以换成下面三条指令中的两条:

```
AND $04
ORA $04
EOR $04
```

您能指出是哪两条指令呢?为什么?

当您换上逻辑运算指令后,CLC指令是否还需要?为什么?

值得指出的是,程序6.6还可以进一步化简,完全可以不用 \$04单元暂存,因为在执行第七条指令 STA \$03后, \$03单元已存好03的值;而在执行四条左移一位指令 ASL后, \$01单元的值为A0。因此,可以将 \$01单元的内容和 \$03单元中的内容进行组合。方法是修改 \$0314单元开始的内容。

```
如:0314-A5 03 LDA $03
    0316-05 01 ORA $01
    0318-85 03 STA $03
    031A-60 RTS
或 0314-A5 03 LDA $03
    0316-45 01 EOR $01
```



```
0318-85 03 STA $03
```

```
031A-60 RTS
```

程序6.6还可以进一步简化,可以省去\$01-\$04的4个零页地址的选用,同样完成字的组合任务,见程序6.7。

程序6.7:

```
1000- A9 43 LDA # $43
1002- 29 0F AND # $0F
1004- 85 06 STA $06
1006- A9 6A LDA # $6A
1008- 0A ASL
1009- 0A ASL
100A- 0A ASL
100B- 0A ASL
100C- 18 CLC
100D- 65 06 ADC $06
100F-20 DA FD JSR $FDDA
1012- 60 RTS
```

运行: \*1000G✓

A3

总之,一道题目的解法是多种多样的,学习用不同方法编程,可以训练思维,扩大思路,学会方法,熟习指令,提高编制程序的能力和技巧。

〔例7〕求平均值

为简单起见,仅求两个数\$3A和\$22的平均值。结果应为\$2E。见程序6.8。

程序6.8:

```
1000- 20 58 FC JSR $FC58
1003- A9 3A LDA # $3A
1005- 18 CLC
1006- 69 22 ADC # $22
1008- 4A LSR
1009- 20 DA FD JSR $FDDA
100C- 60 RTS
```

程序6.8在使用上有很多不足之处:

- 不能进行N个数的累加和求平均
- 和数只能限于一个8位数,不能有进位
- 只适用于无符号数或正数,LSR相当于 $\div 2$

因此,本例和本节中的其它几个例子,处理的都是一些最简单问题。而对一些比较复杂的问题,如:N个数据的累加和,大量数据的传送,多个数据的交换,判断奇偶数,确定正负数,输入输出码的转换,字符串的比较,多个数据块的连续搬移等等诸如此类的问题,简单程序设计是不能胜任的。故有必要介绍其它一些程序设计方法,以补不足。但不管怎样,简单程序设计方法,仍是一切初学者程序设计入门的必然阶梯,因为它处理的问题简单、直接,用的指令少,没有分支、转向和循环,直观、形象,易于初学者接受。通过上述实例分析,我们初步勾画了程序设计的部份面貌,也为程序设计方法和技巧抛出了引子,希望起到开阔视野、启迪思维的作用。

## §2. 分支程序设计

在程序设计中,除极简单的问题外,一般都比较复

杂,解决问题的程序常常带有分支,即根据不同条件,使程序转向不同的地方执行。因此,我们十分需要一种具有判断能力的控制转移手段,以便对程序进行有效的控制。而计算机则按给定的条件进行分析、比较、判断后决定程序的走向,这就是分支程序设计。

6502指令系统中为我们提供了众多的转移指令,这种指令使程序在执行过程中出现分支,其重要性在于它们使计算机具有某种判断功能,可以根据具体条件满足与否,来决定是进行分支转移还是继续执行下一条指令。

我们可以根据问题的性质,选择不同的转移指令,控制程序的流向,以解决各种各样的分支问题。必须着重指出,灵活而准确地使用转移指令,特别是条件转移指令,是高质量汇编语言程序设计不可缺少的部份,而且使用的技巧性很强。

现举若干实例,作为分支程序设计入门的引导。

〔例1〕数的比较

一数放在\$06单元,一数放在\$07单元,比较两者是否相同。相同给出00标志,不同给出FF标志。

对于数或字符串的比较是否相同的问题,最好用异或指令EOR,它的功能是将存储器同累加器内容异或,并将结果送累加器,见程序6.9。

程序6.9:

```
1000- A5 06 LDA $06
1002- 45 07 EOR $07
1004- D0 06 BNE $100C
1006- 85 0B STA $08
1008- 20 DA FD JSR $FDDA
100B- 60 RTS
100C- A9 FF LDA # $FF
100E- 85 09 STA $09
1010- 20 DA FD JSR $FDDA
1013- 60 RTS
```

运行前: \*6:AA✓✓

\*7:FF✓

运行后: \*1000G✓

FF

运行前: \*6:AA✓

\*7:AA✓

运行后: \*1000G✓

00

程序6.9中,在执行异或指令EOR后,有两种情况:两数相同,结果为0;两数不同,结果非0。因此,标志寄存器P中的零标志Z保留了执行EOR指令后结果的情况,前者Z=1,后者Z=0,所以,选用BNE指令也是顺理成章的,它是根据Z标志是0还是1,发生转移或者继续执行下一条指令。两数相同,结果为0,Z=1,执行下一条指令,0→(08),显示(A)=0的结果并停机;两数相异,结果非0,Z=0,转移到\$100C地址,取不同标志FF→(A),FF→(09),显示(A)中内容FF并结束。

〔例2〕判断奇偶数

在 \$06 单元中存放一个数 B, 若 B 是奇数, 给出 01 标志; 若 B 是偶数, 给出 00 标志。

任何一个数(正数, 负数或零), 将其展开成一个 8 位二进制数时, 则第 0 位不是“1”就是“0”。若原数是偶数或零时, 则第 0 位一定是“0”; 若原数是奇数时, 则第 0 位一定是“1”。这个事实告诉我们, 为了判断数的奇偶性, 可以改为判断第 0 位是“1”还是“0”。

如果我们将一个数先右移一位, 再左移一位, 则第 0 位一定是“0”。

这样, 我们就可以根据运算前后的值相同与否, 判定数的奇偶性, 结果相同为偶数, 结果相异为奇数。故有程序 6.10。

程序 6.10:

```
1000- A5 06    LDA  $06
1002- 4A      LSR
1003- 0A      ASL
1004- C5 06    CMP  $06
1006- D0 05    BNE  $100D
1008- A9 00    LDA  # $00
100A- 4C 0F 10 JMP  $100F
100D- A9 01    LDA  # $01
100F- 20 DA FD JSR  $FD0A
1012- 60      RTS
```

运行前: \* 6; C7 ✓ 输入一个奇数

运行后: \* 1000G ✓

01 给出奇数标志

运行前: \* 6; 34 ✓ 输入一个偶数

运行后: \* 1000G ✓

00 给出偶数标志

〔例 3〕找两数中较大者

两数分别存入 \$6000 和 \$6001 单元中, 将大数找出并放入 \$6002 单元中。

取第一个数, 与第二个数比较, 若第一个数大, 则存入 \$6002 单元, 反之, 第二个数大, 将第二个数调入累加器 A, 再用 STA \$6002 指令, 存入 \$6002 单元。见程序 6.11。

程序 6.11:

```
1000- AD 00 60  LDA  $6000
1003- CD 01 60  CMP  $6001
1006- B0 03     BCS  $100B
1008- AD 01 60  LDA  $6001
100B- 8D 02 60  STA  $6002
100E- 60      RTS
```

运行前: \* 6000; 4F 3F

运行后: \* 1000G ✓

\* 6002 ✓

\* 6002-4F

运行前: \* 6000; 6D 80

运行后: \* 1000G ✓

\* 6002 ✓

\* 6002-80

这段程序的思想是用 CMP 指令来比较两个数, 然后利用 BCS 指令进行程序分支。在比较指令之后配合条件转移指令, 是分支程序设计中经常使用的一种方

法。

应该指出的是, 在选用 CMP 指令时, 要注意几点:

• CMP 指令是累加器 A 的内容同存储器 M 的内容比较, 比较实际上是做一次减法操作, 即  $A-M$

• CMP 指令同减法指令 SBC 区别之一是, 借位标志  $\bar{C}$  (表示 C 标志取反) 不参加减法运算, 前者  $A-M$ , 后者  $A-M-\bar{C}$ 。

• CMP 指令同减法指令 SBC 的另一个区别是, 减法运算结果不送入累加器 A, 即 CMP 执行后不改变 A 的内容, 前者  $A-M$ , 后者  $A-M-\bar{C} \rightarrow A$ 。

• 执行 CMP 指令后, 影响 P 寄存器中的标志位 C、Z、N, 尤其是影响标志位 C。若  $A \geq M$ , 表示够减无借位, C 置 1; 若  $A < M$ , 表示不够减有借, C 置 0。因此, 可由 C 的状态判断 A 同 M 中两数谁大谁小。

因此, 本例在选用 CMP 指令后, 自然配合使用 BCS 指令, 因为 BCS 指令是根据 P 寄存器中 C 的状态决定分支转移或继续执行下一条指令, 即  $C=1$  时转移,  $C=0$  时继续。

〔例 4〕确定正负数

一个数(正数或负数)存放在 \$06 单元中, 设计一个程序能判断该数是正数还是负数, 并给出正数标志 00、负数标志 01。

由于本例中只判断正数或负数性质, 而且仅限于对一个数的判别, 因此, 选用 BPL 指令是合适的, 因为它以 N 作为判别标准,  $N=0$ , 表示正数, 执行分支动作;  $N=1$ , 表示负数, 则执行下一条指令。N 标志的状态可由执行 LDA 指令后, P 标志寄存器中的 N 标志位取得, 因为 LDA 指令对符号位 N 及零标志位 Z 皆有影响。故有程序 6.12。

程序: 6.12:

```
1000- A5 06    LDA  $06
1002- 10 05    BPL  $1009
1004- A9 01    LDA  # $01
1006- 4C 0B 10 JMP  $100B
1009- A9 00    LDA  # $00
100B- 85 07    STA  $07
100D- 60      RTS
```

运行前: \* 6; 7D 输入一个正数

运行后: \* 1000G ✓

\* 7 ✓

\* 0007-00 给出正数标志

运行前: \* 6; 96 输入一个负数

运行后: \* 1000G ✓

\* 7 ✓

\* 0007-01 给出负数标志

程序 6.12 中安排了一条无条件转移指令 JMP, 它是编程人员事先安排好的, 它控制程序没有任何条件地实现转移, 即将程序由通常的顺序执行状态转向另一个指定的目标地址, 这和条件转移指令的执行情况不同, 后者则是在满足某种条件的情况下, 才使程序转移, 即根据 P 寄存器中各标志位状况来确定程序是否转移。

〔例5〕输入码转换

如从键盘上按下一位10进制数,将其转换成16进制数。

10进制数的代码是0-9,程序的设计必须保证小于0的和大于9的键码都不要,而且还应保证按任何字符和运算符都不应出错,能自动返回等待正确输入;同时,也应安排一个控制键,例如回车键,让其程序自动停机。因此,本程序的设计是包含多个条件语句的分支程序设计。见程序6.13。

程序6.13:

```

1000- 20 1B FD   JSR   $FD1B
1003- C9 8D     CMP   # $8D
1005- F0 11     BEQ   $1018
1007- C9 B0     CMP   # $B0
1009- 90 F5     BCC   $1000
100B- C9 BA     CMP   # $BA
100D- B0 F1     BCS   $1000
100F- 20 DA FD   JSR   $FD1B
1012- 20 8E FD   JSR   $FD8E
1015- 4C 00 10   JMP   $1000
1018- 60                RTS
    
```

程序6.13运行后,等待操作者从键盘键入输入的数码,若在0至9之间,则显示B0-B9(0-9对应的ASCII码),否则按任何字符、符号都无效,转向\$1000开始重新接受按键输入,若要停机,则按RETURN(它的ASCII码是8D)键。程序中用了三个监控子程序,\$FD1B:等待键盘输入,\$FD1A:输出一个字符的代码子程序,\$FD8E:换行。

〔例6〕16进制数转换成ASCII码

计算机键盘与屏幕显示器都采用ASCII码表示字符。一个16进制数字可以用代码0-9或者用字母A-F表示。这两段内容在正常方式的屏幕显示时,对应的ASCII码是B0-B9及C1-C6。一个需要显示的数字必须先换成对应的ASCII码,然后再送入显示器。

要完成16进制数向ASCII码值转换,其方法是:如果属于0-9数字,则只要将它们加上B0;如果属于A-F字母,则需将它们加上B7。所以,16进制数字转换成ASCII码的程序,实际上是一个分支转移程序:即判断输入码属于0-9还是A-F,是前者加B0,是后者加B7。故有程序6.14。

程序6.14:

```

1000-A5 06     LDA   $06
1002-C9 0A     CMP   # $0A
1004-90 03     BCC   $1009
1006-18                CLC
1007-69 07     ADC   # $07
1009-69 B0     ADC   # $B0
100B-85 07     STA   $07
100D-20 DA FD   JSR   $FD1A
1010-60                RTS
    
```

运行前: \* 6;F  
运行后: \* 1000G  
\* C6

或: \* 1000G

\* 7  
\* 0007-C6

运行前: \* 6;05  
运行后: \* 1000G

\* B5  
或: \* 1000G

\* 7  
\* 0007-B5

本题也可用另法求解,其思路是先把数字0的ASCII码B0加到所有待转换的16进制数上去,然后与BA比较,以判断是否属于0-9,再决定是否加07。故有程序6.15。

程序6.15:

```

1000- A5 06     LDA   $06
1002- 29 0F     AND   # $0F
1004- 09 B0     ORA   # $B0
1006- C9 BA     CMP   # $BA
1008- 90 02     BCC   $100C
100A- 69 06     ADC   # $06
100C- 85 07     STA   $07
100E- 20 DA FD   JSR   $FD1A
1011- 60                RTS
    
```

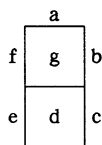
程序6.15的运行方法同程序6.14。

〔例7〕10进制数转换成7段代码

7段代码是七段发光二极管显示器显示字符所采用的代码,对共阴极7段显示器来说,所用7段代码显示10进制数字的对应关系为:

| 数字 | 代码 | 数字 | 代码 |
|----|----|----|----|
| 0  | 3F | 5  | 6D |
| 1  | 06 | 6  | 7D |
| 2  | 5B | 7  | 07 |
| 3  | 4F | 8  | 7F |
| 4  | 66 | 9  | 6F |

7段代码的安排是:



64 32 16 8 4 2 1

|  |   |   |   |   |   |   |   |
|--|---|---|---|---|---|---|---|
|  | g | f | e | d | c | b | a |
|--|---|---|---|---|---|---|---|

2<sup>6</sup> 2<sup>5</sup> 2<sup>4</sup> 2<sup>3</sup> 2<sup>2</sup> 2<sup>1</sup> 2<sup>0</sup>

在一个字节中构成7段代码的方法是:最高位总是0,其余7位由低位向高位顺序为a、b、c、d、e、f、g各段代码(1亮,0不亮)。

例如,当a、b、c、d、g均为1时,则显示10进制数3,即:

$$\begin{aligned}
 &2^0 \times 1 + 2^1 \times 1 + 2^2 \times 1 + 2^3 \times 1 + 2^4 \times 0 + 2^5 \times 0 + 2^6 \times 1 \\
 &= 1 + 2 + 4 + 8 + 0 + 0 + 64 \\
 &= (79)_{10}
 \end{aligned}$$

= (4F)<sub>16</sub>

所以,要求编一个程序,输入0—9这10个10进制数中的任一代码,显示出相应的16进制数。

程序可以这样安排,在\$6000中存放一个10进制数,比较是否在0—9之间,是的,取对应的16进制代码;不是的,显示0。故有程序6.16。

程序6.16:

```

1000- A9 00   LDA   # $00
1002- AE 00 60 LDX   $ 6000
1005- E0 0A   CPX   # $0A
1007- B0 03   BC9   $ 100C
1009- BD 00 70 LDA   $ 7000,X
100C- 8D 01 60 STA   $ 6001
100F- 20 DA FD JSR   $ FDDA
1012- 60     RTS
    
```

程序6.16运行前应先设置:

```

7000-3F 06 5B 4F 66 6D 7D 07
7008-7F 6F
* 6000:5 ✓
* 1000G ✓
* 6D
    
```

本题可以进一步简化,只要在\$06单元中放置0—9当中的任一个10进制数码,运行程序6.17后,立即得到结果。

程序6.17:

```

1000-A4 06   LDY   $ 06
1002-B9 00 70 LDA   $ 7000,Y
    
```

```

1005-20 DA FD JSR   $ FDDA
    
```

```

1008-60     RTS
    
```

程序6.17非常简捷,运行前应设置\$06单元和\$7000—\$7009单元中的值。

如(06)=03

```

7000-3F 06 5B 4F 66 6D 7D 07 7F 6F
    
```

\* 1000G ✓

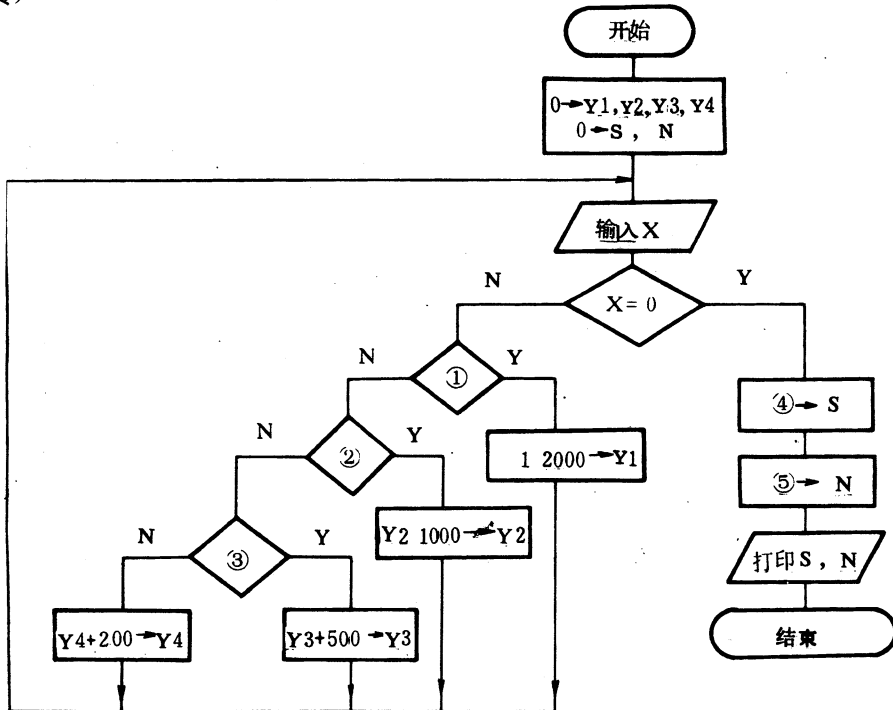
显示:4F

以上我们通过几个实例,介绍了分支程序设计的一些问题。可以看出,分支程序解决和处理的问题,要比简单程序广泛和多样,在设计方法和应用技巧上,又比简单程序丰富和灵活。

应该着重指出的是,学习和掌握分支程序设计有一定的难度,其困难所在是如何选用条件转移指令,以及什么条件转移和转向到那里去,这需要经常编程和大量调试。但有一点是至关重要的,必须牢牢掌握标志寄存器P的各个有关标志位是0还是1的状况。因为标志寄存器P中保留了每条指令执行后的情况,如执行后结果为全零或非0;是正数或负数;有进位或无进位;有溢出或无溢出等,这些情况即为判断条件分支的重要依据,也是学习本节内容的要领和程序设计的重点。

还有一点要说的,仅仅掌握了简单程序、分支程序的设计方法,还不能解决名目繁多形式各异的实际问题,特别是不能有效地解决大量的在处理形式上又完全相同的重复性计算问题。因此,循环程序的设计方法和技巧,应该作为重点来介绍,请看下讲内容。

(上接30页)



答案:①

②

③

④

⑤

# 初级技术员级软件水平考试辅导

## PC BASICA 试题

北京市计算中心(100005) 李志刚

### 一、填空

①  $(2\sin 45^\circ + \cos 30^\circ) / (3|X| \ln(Z-5))$  的 BASIC 表达式是 \_\_\_\_\_

② 等效于 SWAP X, Y 的语句组是 \_\_\_\_\_

③ 表达式  $\text{SQR}(\text{VAL}(\text{RIGHT} \$ ("1245", 2))) + 4$  的值是 \_\_\_\_\_

④ 将内存中的 BASIC 程序以 PROG 为文件名, 用 ASCII 码方式存入 A 盘的命令是 \_\_\_\_\_

⑤ 在使用 FOR/NEXT 循环语句时, 要求内循环的 NEXT 语句必须出现在外循环的 NEXT 语句之前, 是由堆栈的 \_\_\_\_\_ 性质决定的。

六、阅读下列说明, 从选择答案中找出正确答案, 把对应的字母填入括号内。

① 标准程序流程图是指国家标准( )

供选择的答案:

- A. GB2312—80                      B. GB1526—89  
C. ISO 5807—85

② 程序的三种基本控制结构是( ), 它们的共同特点是( )。

供选择的答案:

- A. 过程、子程序、函数      B. 顺序、条件、循环  
C. 递归、队列、堆栈      D. 调用、返回、转移  
E. 不能嵌套使用              F. 只能写简单程序

G. 只有一个入口和一个出口    I. 已经用硬件实现, 只需调用

③ 结构化程序设计的一种基本方法是( )。

供选择的答案:

- A. 筛选法                      B. 枚举法  
C. 归纳法                      D. 逐步求精法

④ 结构化程序设计语言中( ) GOTO 语句。

供选择的答案:

- A. 必需有    B. 没有    C. 可有可无    D. 限制

⑤ 在子程序头, 一般要写出一些变量, 这些变量被称为( )。它们应与调用该子程序的语句中的( ) 在个数、类型、顺序等方面相一致。

供选择的答案:

- A. 条件参数    B. 形式参数    C. 入口参数  
D. 实际参数    E. 出口参数    F. 局部参数  
G. 全局参数

三、读下列程序, 从选择答案中找出正确答案, 把对应的字母填入括号内”

① 程序:

```
10 READ A, B, C
20 RESTORE
30 DATA 1, 2, 3, 4
40 READ G, F, E, D, C, B
50 DATA 4, 5, 6, 7
60 PRINT A; B; C; D; E; F; G
70 END
```

运行结果是: ( )

供选择的答案:

- A. 1 2 3 4 5 6 7  
B. 1 4 5 4 3 2 1  
C. 1 5 4 4 3 2 1  
D. 7 6 5 4 3 2 1

② 程序:

```
10 INPUT "n="; N
20 M=N+1
30 F=1
40 GOSUB 60
50 PRINT F; END
60 M=M-1
70 IF M<>1 THEN GOSUB 60
80 F=F*M
90 M=M+1
100 RETURN
```

运行结果是: ( )

RUN

5 ↓

- A. 24  
B. 48  
C. 96  
D. 120

③ 此程序运行后打印的结果是( )

程序:(设真为-1, 假为0)

```
10 FOR A=-1 TO 0
20 FOR B=-1 TO 0
30 PRINT (A AND B)+(NOT A)
40 NEXT B
50 NEXT A
60 END
```

供选择的答案:

- A. 0 0 -1 -1  
B. 0 1 -1 -1  
C. -1 0 -1 -1  
D. 1 0 -1 -1

④ 此程序运行的结果是( )

程序:

```
10 Y=14
20 DEF FNA(X)=X^2+2*X
30 Y=Y+FNA(FNA(3))
40 PRINT "Y1="; Y
50 Y=(Y-FNA(2))/2
60 PRINT "Y2="; Y
70 END
```

供选择的答案:

- A. 269  
B. 269 130.5  
C. Y1=269 Y2=130.5  
D. Y1=269  
Y2=130.5

⑤ 运行此程序, 打印结果是( )。

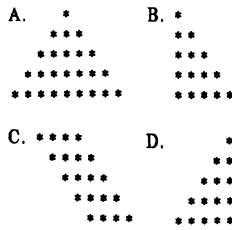
程序:

供选择的答案

```

10 N=120 FOR X=1 TO N
20 FOR X=1 TO N
30 PRINT "*";
40 NEXT X
50 PRINT
60 N=N+1
70 IF N>5 THEN 90
80 GOTO 20
90 END

```



```

60 NEXT I
70 IF A(1)<A(2) THEN SWAP A(1),A(2)
80 FOR I=3 TO 10
90 IF _____ THEN SWAP A(1),A(I);
    GOTO 110
100 IF _____ THEN SWAP A(2),A(I)
110 NEXT I
120 PRINT"最大数";A(1),"最小数";A(2)
130 A$ = _____
140 B$ = _____
150 PRINT"中奖号码"A$+B$
160 END

```

四、读程序,从供选择答案中将程序执行屏幕显示的  
错误信息对应的字母填入括号内。

①( )

```

10 FOR I=2 TO 3
20 DIM A(I)
30 A(I)=I
40 PRINT A(I),
50 NEXT I
60 END

```

②( )

```

10 DIM A(5)
20 FOR I=1 TO 10
30 READ A(I)
40 PRINT I;A(I)
50 NEXT I
60 DATA 0,9,8,7,6,5,4,3,2,1
70 END

```

③( )

```

10 FOR I=1 TO 5
20 PRINT TAB(1);I*I;PRINT SIN(I)
30 NEXT I
40 END

```

④( )

```

10 READ A$
20 IF A$=EWS THEN PRINT P$;END
30 P$=P$+A$
40 GOTO 10
50 DATA THE,IS,A,PEN,YES

```

- 供选择的答案:A. Duplicate Definition  
B. Syntax error  
C. Subscript out of range  
D. Type mismatch

五、下面是模拟儿童画竞赛的鼓励奖抽奖程序,请补充  
填空。

抽奖原则:随机产生10个四位整数,取其中最大  
数的十位数字与最小数的个位数字组合成中奖号码  
的两位尾数。

```

程序:10 RANDOMIZE
    20 DIM A(10)
    30 FOR I=1 TO 10
    40 A(I)= .....
    50 PRINT A(I),

```

六、阅读下列程序说明和 BASIC 程序,请补充填空。  
[程序说明]

本程序是将一正整数序列 {K1,K2,……,Kn} 重  
排列成一个新的序列。新序列中,比 K1 小的数都在 K1  
的前面(左面),比 K1 大的数都在 K1 后面(右面)。

例如:当 n=9 时,序列 {6,8,9,1,2,5,4,7,3}  
经重新排列后成为: {3,4,5,2,1,6,8,9,7}

[程序]

```

10 INPUT "N=";N
20 IF N<2 THEN 10
30 _____
40 FOR I=1 TO N
50 INPUT A(I);IF _____ THEN 50
60 NEXT I
70 FOR I=1 TO N:PRINT A(I),:NEXT I;
    PRINT
80 I=1
90 FOR I=2 TO N
100 IF _____ THEN 170
110 X=A(I)
120 FOR J=I TO 2 STEP -1
130 _____
140 NEXT J
150 A(1)=X
160 _____
170 NEXT I
180 PRINT
190 FOR I=1 TO N:PRINT A(I),NEXT I
200 END

```

七、“合理化建议”奖金发放以年经济效益为依据,具  
体发奖标准是:

| 等级 | 年经济效益(万元) | 奖金(元) |
|----|-----------|-------|
| 一  | 100以上     | 2000  |
| 二  | 10~99     | 1000  |
| 三  | 1~9       | 500   |
| 四  | 不足1       | 200   |

请填写流程图中编号处的内容,使之能根据读入的年  
经济效益 X,按发奖标准兑奖,并统计发奖总数 S(单  
位万元)和合理化建议总数 N。(下转28页)



# 带 A/D 的单片机最小应用系统

合肥中国科技大学计算中心(230026) 张培仁 刘振安

A/D 变换广泛应用于工业控制、自动化、数据处理、智能仪器中。和单片机接口的 A/D 芯片主要有三类：双积分式 A/D 变换器，逐次比较式 A/D 变换器和 V/F 式 A/D 变换器。下面分别介绍它们和单片机的接口和硬软件结构。

智能热电偶温度测试仪中是以 8031 最小系统为核心，使用高精度  $4\frac{1}{2}$  位积分式 A/D 变换器作为采样。该仪器的系统框图如下：

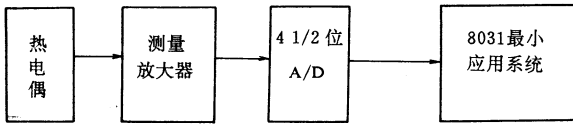


图1

在实验室和工业系统中，通过热电偶测温度经常遇到的问题是由于热电偶测出是毫伏数。最后要查阅热电偶手册，查出相应的温度，而使用单片机可以控制  $4\frac{1}{2}$  位 7135A/D 片不断进行采样，再经过单片机的数据处理后显示出相应温度值。

用测量放大器把热电偶信号从毫伏放大到几伏。测量放大器由四个运放组成的直流放大器。共模抑制比可达 90dB，基本上满足了三种热电偶在工业测量中的要求。放大器必须加以屏蔽，以尽量减少干扰和噪声。放大器模拟地与单片机数字地严格相互分开，最后再联在一点，使地线上压差达最小。

双积分 A/D(7135)与单片机连接电气原理图如图2所示。

A/D7135的特性如下：

- (1) 总计数达  $\pm 20000$  个计数，精度为  $\pm 1$  个数。
- (2) 输入阻抗大于  $10^9\Omega$ 。
- (3) 自校零，保证零电压输入时读数为 0。
- (4) 采用 BCD 码扫描输出。
- (5) 设有 6 个 I/O 辅助信号可方便地与单片机接口与控制。

(6) 输出与 TTL 电路兼容。

7135 由  $\pm 5V$  供电， $V^+$  接  $+5V$ ， $V^-$  接  $-5V$ ，DGND 为数字地，即接  $\pm 5V$  电源地。 $U_R$  为基准电压的高电位输入端，AGND 为模拟地，它是模拟信号的地，又是基准电压的低电位端。INT 为积分器输出，接积分电容。BUF 为缓冲放大器的输出，接积分电阻。AE 为自动调零输入端，接自动调零电容。 $CR^+$  为基准电容的高电位端， $CR^-$  为基准电容低电位端。INHI 为模拟信号输入的正端。INLO 为模拟信号输入的负端。R/H 为运行或保持操作控制端。D1, D2, D3, D4, D5 为位扫描输出端，其中 D1 为个位，D5 为万位。B1, B2, B3, B4, B8 为 BCD 码数据，B1 为低位，B8 为高位。POL 为信号极性输出，信号为正时，POL 为 1，OR 为超量程状态输出，UR 为欠量程输出。BUSY 为忙信号输出，它指示 A/D 是正在进行转换还是已经转换完毕。 $\overline{STB}$  为数字选通

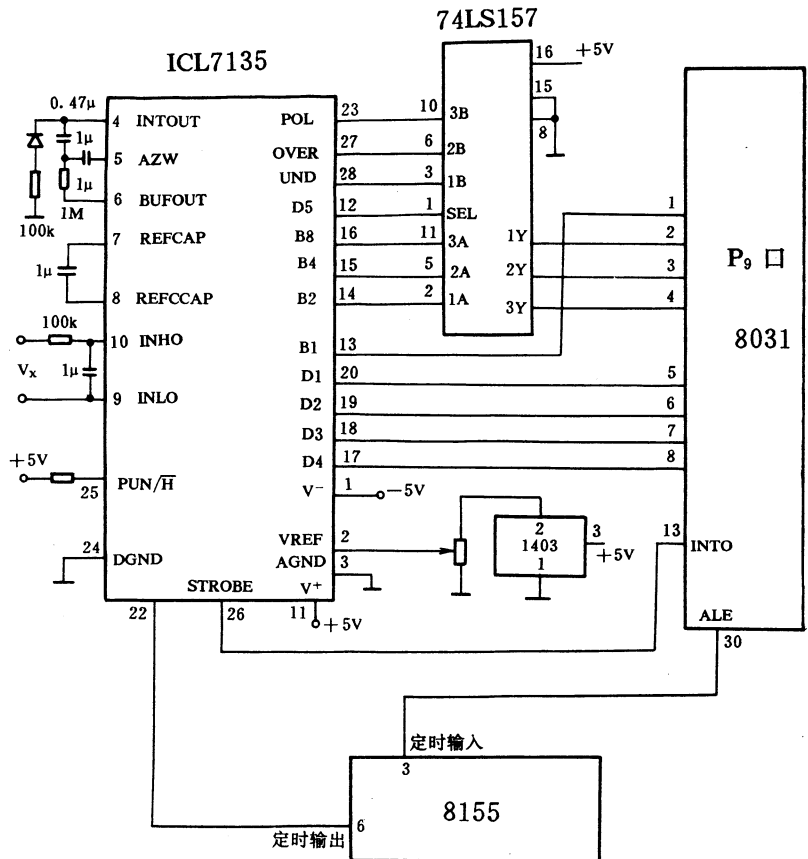
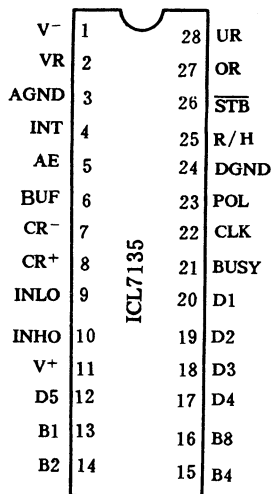


图2



ICL7135 管脚图

图3

程信号也一同被读入。在每一位数据产生后,都会产生中断请求,在中断服务中设置一个中断次数计数器。单片机根据第几次中断来判断是第几位产生的中断。准确将数据每一位存入所对应的缓冲区内。

7135输出定时波形如图4:

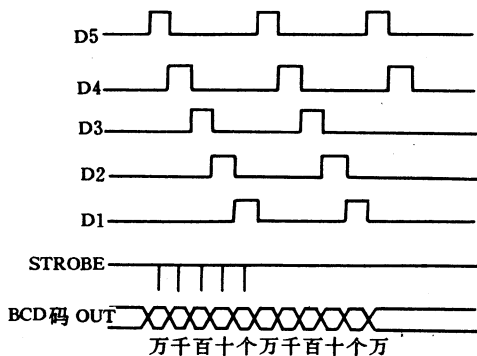


图4

单片机将数据读入内存以后,经过处理,将温度信号由8155的PA与PC口送显示。A口是段码,C口是位码,即选中六个数码管的那一个数码管。

数据输入在INT00中断服务子程序中完成。该子程序中用计数器计中断次数,决定是哪一位的中断,从而存入相应内存中,第一次中断将万位,极性,欠过量程一起读入8031中的RAM中,第二次中断将千位,第三次将百位,第四次将十位,第五次将个位读入,这样不断循环。

8031中RAM的数据内容和格式说明(与7135A/D变换有关的)如下:

40H:A/D输入极性。

41H:过量程标志。

输出,用来通知单片机准备读取7135的输出数据。

1403精密稳压电路经电阻分压产生7135的基准电压。

7135的转换时钟,通过8031的ALE的信号进行分频取得(分频由8155完成),7135和74LS157联用,数据通过8031的P1口读入,用7135的STBC(转换好)做8031的外中断源,使8031的中断在STB的上升沿得到响应,读取数据,7135数据输出顺序D1,D2……D5,在万位的数据有效时,极性,欠过量

42H:欠量程标志。

43H:A/D BCD码十位,个位。

44H:A/D BCD码千位,百位。

45H:A/D BCD码万位(低三位)

程序框图(中断服务程序即A/D控制部分)

程序清单:A/D部分

```

;
int00:      mov ie, # 00h
            push a
            mov a, r0
            push a
            mov a, 47h
            inc a,
            mov 47h, a
            cjne a, # 01, i1
            ajmp ino
i1:         cjne a, # 02, i2
            ajmp in8
i2:         cjne a, # 3, i3
            ajmp in9
i3:         cjne a, # 4, i4
            ajmp ina
i4:         cjne a, # 5, i5
            ajmp inb
i5:         ajmp ine
ino:        mov a, p1
            jnb acc. 3, in1
            mov 40h, # 0
            ajmp in2
in1:       mov 40h, # 0ffh
in2:       jb acc. 2, in3
            mov 41h, # 0
            ajmp in4
in3:       mov 41h, # 0ffh
in4:       jb acc. 1, in5
            mov 42h, # 0
            ajmp in6
in5:       mov 42h, # 0ffh
in6:       jb acc. 0, in7
            mov 45h, # 0
            ajmp ine
in7:       mov 45h, # 1
            ajmp ine
in8:       mov a, p1
            anl a, # 0fh
            swap a
            mov 44h, a
            ajmp ine
in9:       mov a, p1
            anl a, # 0fh
            mov r0, # 44h

```



```

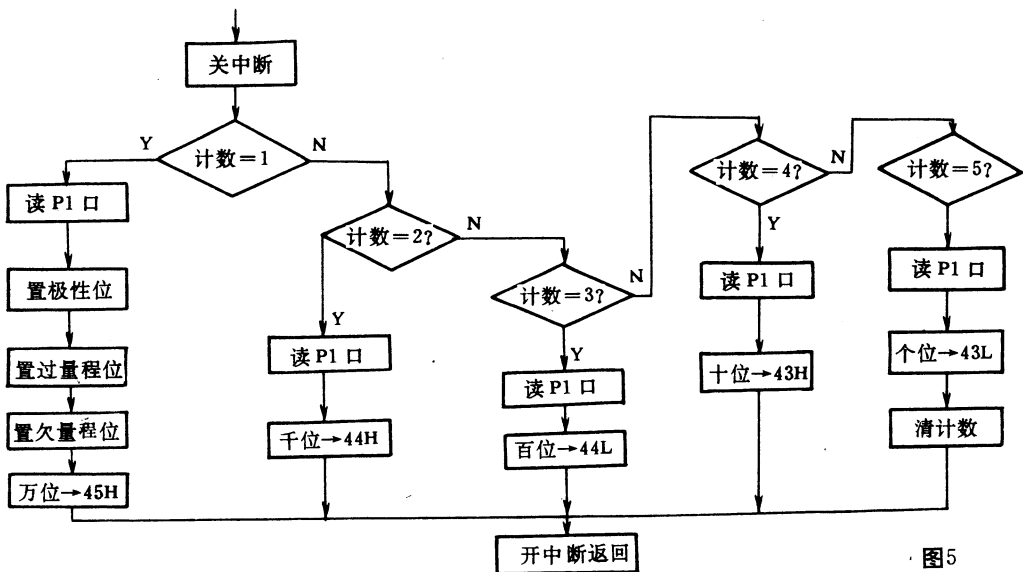
xchd a, @r0
ajmp ine
ina:   mov a, p1
       anl a, # 0fh
       swap a
       mov 43h, a
       ajmp ine
inb:   mov a, p1
       anl a, # 0fh
       mov r0, # 43h

```

```

xchd a, @r0
clr a
mov 47h, a
pop a
mov r0, a
pop a
mov p1, # 0ffh
mov ie, # 84h
reti

```



· 图5

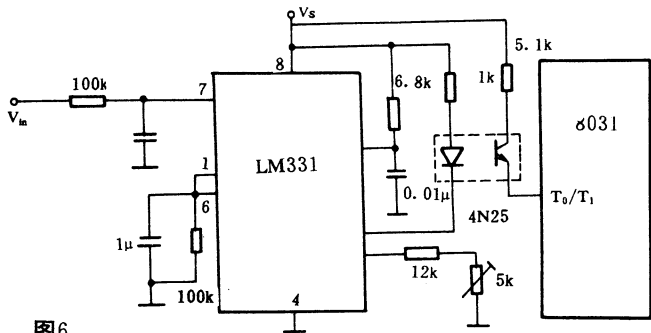


图6

使用 V/F 转换器作 A/D 时具有独特的优点。当单片机采用 V/F 型 A/D 时有如下一些优点：

- (1) 接口简单, 占 8031 资源少。
- (2) 频率信号输入灵活。一般从 T0 或 T1 进入单片机。
- (3) 抗干扰性能好, 容易进行光电隔离。
- (4) 便于远距离传输。

主要用于非快速过程 A/D 变换。  
用 LM331 和 8031 做接口图如下：

图7是 LM331 原理图, 它的原理简介如下：

输入比较器将输入电压  $V_{IN}$  与  $V_x$  比较, 当  $V_{IN} > V_x$  时, 启动单脉冲定时器并导通频率输出晶体管和开关电流源, 定时器的定时周期  $t = 1.1R_1 \cdot C_1$ , 在这个周期中电流  $i$  向电容  $C_L$  充电, 使  $V_x$  上升, 当  $V_x$  上升至  $V_x > V_{IN}$ , 电流  $i$  关断, 定时器自动复位, 同时,  $C_1$  逐渐通过  $R_1$  放电直到  $V_x < V_{IN}$  为止, 然后比较器再次启动定时器, 开始下一个循环。

V/F 的变换程序是很简单的。单片机初始化时把 T0 清 0, 以后启动 T0, 定时器计输入脉冲。输入脉冲计数多少是和输入电压成线性关系。

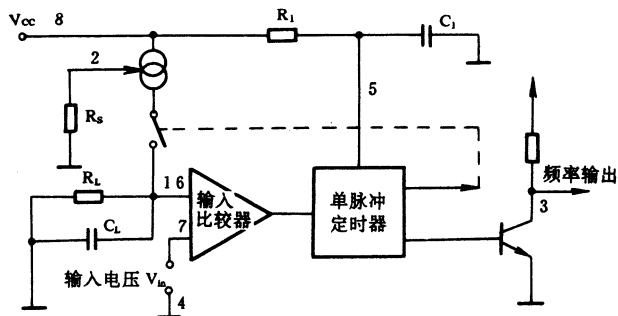
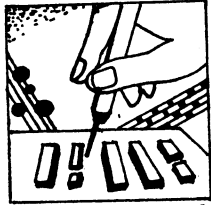


图7



## 学装微电脑

# 7段数码管读数装置

易齐干

本文介绍使用光导纤维检测7段数码管每段发出的光的信息;以及光—电信息转换和微电脑组成的读数装置。

同一距离检测有无光信号的状况

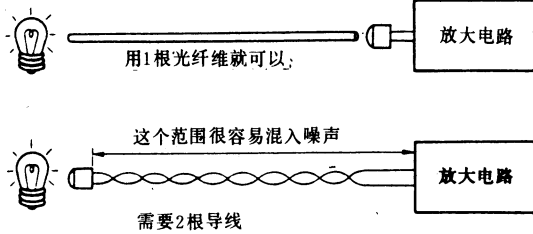


图1

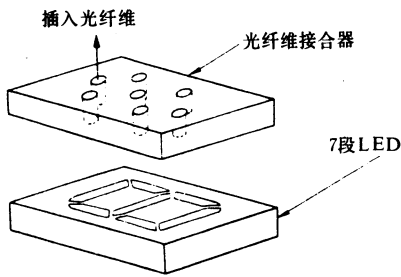


表1

图2

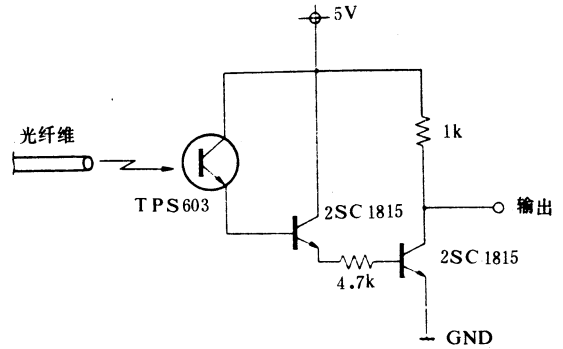


图3

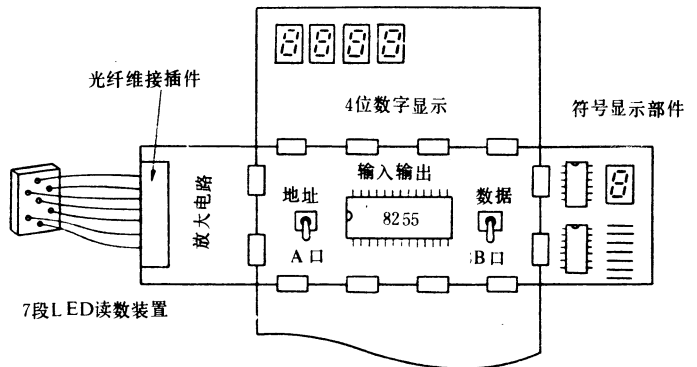


图4

| 标号    | 助记符         | 地址 | 机器语言     | 注释          |
|-------|-------------|----|----------|-------------|
| START | LD A,90H    | 00 | 3E 90    | 设置CW        |
|       | OUT (03H),A | 02 | D3 03    |             |
|       | LD SP,0100H | 04 | 31 00 01 | 设置SP        |
| J1    | LD HL,00D0H | 07 | 21 D0 00 | 设置最前面数据的地址  |
| J2    | IN A,(00H)  | 0A | DB 00    | 输入数据        |
|       | LD B,HL     | 0C | 46       |             |
|       | CP B        | 0D | B8       | 与数字数据进行比较   |
|       | JP Z,J3     | 0E | CA 1E 00 |             |
|       | INC HL      | 11 | 23       |             |
|       | LD A,(HL)   | 12 | 7E       |             |
|       | AND A       | 13 | A7       | 字数据结束?      |
| J3    | JP NZ,J2    | 14 | C2 DA 00 |             |
|       | LD A,9EH    | 17 | 3E 9E    | 输出E1        |
|       | OUT (02H),A | 19 | D3 02    |             |
|       | JP J1       | 1B | C3 07 00 |             |
| J3    | LD A,L      | 1E | 7D       | 数字数据地址 + 16 |
|       | ADD A,10H   | 1F | C6 10    |             |
|       | LD L,A      | 21 | 6F       |             |
|       | LD A,(HL)   | 22 | 7E       |             |
|       | CPL         | 23 | 2F       |             |
|       | OUT (02H),A | 24 | D3 02    |             |
|       | JP J1       | 26 | C3 07 00 |             |

| 标号     | 助记符    | 地址 | 机器语言 | 注释     |
|--------|--------|----|------|--------|
| DATA M | DB 03H | D0 | 03   | 0 数字数据 |
|        | DB 9FH | D1 | 9F   | 1      |
|        | DB 25H | D2 | 25   | 2      |
|        | DB 0DH | D3 | 0D   | 3      |
|        | DB 99H | D4 | 99   | 4      |
|        | DB 49H | D5 | 49   | 5      |
|        | DB 41H | D6 | 41   | 6      |
|        | DB 1BH | D7 | 1B   | 7      |
|        | DB 01H | D8 | 01   | 8      |
| DB 09H | D9     | 09 | 9    |        |
| DB 00H | DA     | 00 | 结束   |        |
| DATA F | DB 03H | E0 | 03   | 0 显示数据 |
|        | DB 9FH | E1 | 9F   | 1      |
|        | DB 25H | E2 | 25   | 2      |
|        | DB 0DH | E3 | 0D   | 3      |
|        | DB 99H | E4 | 99   | 4      |
|        | DB 49H | E5 | 49   | 5      |
|        | DB 41H | E6 | 41   | 6      |
|        | DB 1BH | E7 | 1B   | 7      |
|        | DB 01H | E8 | 01   | 8      |
| DB 09H | E9     | 09 | 9    |        |

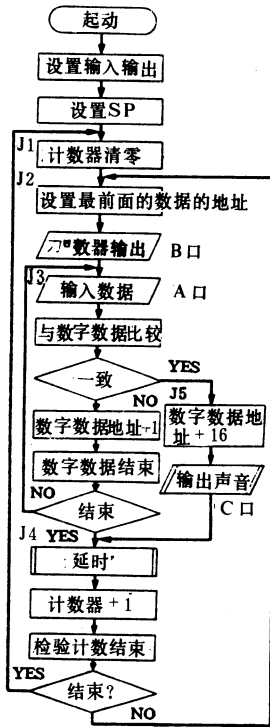


图5

光的信息转换为电的信息过程中,光导纤维的传输工具。与使用光敏晶体管直接转换,电缆根数可以减少一半。

另外,光电转换元件与放大电路距离愈近,噪声干扰愈少,效果愈好。

这就是使用光导纤维的优点。如图1所示。

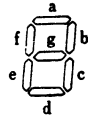
光导纤维的接合器由5mm厚丙烯酸塑料加工而成。要求光导纤维的中心尽量接近每段LED的中心部位。如图2所示。

光电信息转换部份电路如图3所示。光敏晶体管(东芝TPS603)将光信息转换为电信息。

光导纤维传输来的光照射到光敏晶体管的基极,

表2

| 7段LED<br>的显示 | A口 |    |    |    |    |    |    |    | 16进制<br>表示 |
|--------------|----|----|----|----|----|----|----|----|------------|
|              | a  | b  | c  | d  | e  | f  | g  | —  |            |
|              | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |            |
| 0            | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 1  | 03H        |
| 1            | 1  | 0  | 0  | 1  | 1  | 1  | 1  | 1  | 9FH        |
| 2            | 0  | 0  | 1  | 0  | 0  | 1  | 0  | 1  | 25H        |
| 3            | 0  | 0  | 0  | 0  | 1  | 1  | 0  | 1  | 0DH        |
| 4            | 1  | 0  | 0  | 1  | 1  | 1  | 0  | 1  | 99H        |
| 5            | 0  | 1  | 0  | 0  | 1  | 0  | 0  | 1  | 49H        |
| 6            | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 1  | 41H        |
| 7            | 0  | 0  | 0  | 1  | 1  | 0  | 1  | 1  | 1BH        |
| 8            | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 01H        |
| 9            | 0  | 0  | 0  | 0  | 1  | 0  | 0  | 1  | 09H        |



发射极与集电极则导通。再经过两个晶体管(2SC1815)逐级放大,输出端或是L电平、或是H电平。

要读出7段管和小数点时,必须有8个如图3所示电路。

7段数码管读数装置与μP-80教育套件端口A相连,符号显示部件与端口C相连。如图4。程序要求输入7段数码管读数装置的LED灯亮信息,由符号显示部件显示。如果读到0~9数字以外的字符时,显示“E”。

编程的基本思想是将0~9数据预先写入存储器,与端口A输入数据进行比较,两者一致,端口C输出该数据;如果不一致,输出“E”。

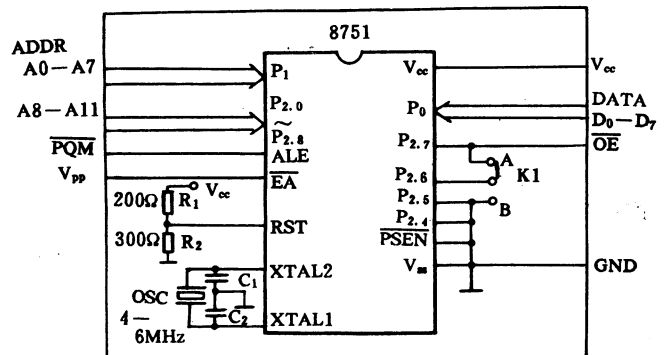
流程图如图5所示。程序清单如表1所示。7段数码管显示编码表如表2。

类似这种程序,随着读数装置不同的输入,仅改变微电脑输出数据,能够完成各种控制工作,例如,灯亮的启动;继电器驱动等等。有兴趣者不妨一试。

(上接38页)

|      |                  |   |    |                 |   |   |   |   |
|------|------------------|---|----|-----------------|---|---|---|---|
| 校验   | V <sub>IHL</sub> | 0 | 1  | 1               | 0 | 0 | × | × |
| 加密编程 | V <sub>IHL</sub> | 0 | 0* | V <sub>PP</sub> | 1 | 1 | × | × |

注：“1”是逻辑高电平  
“0”是逻辑低电平  
“×”是任意状态  
“V<sub>PP</sub>”是+21V±0.5V  
“0\*”是50ms低电平脉冲  
“V<sub>IHL</sub>”是RST输入高电平, 2.5V ≤ V<sub>IHL</sub> ≤ V<sub>CC</sub> + 0.5V



# 向用户推荐——PIC16C5X 系列和 17C42 单片机

中国电子器材深圳公司供稿

美国 Microchip(微晶片)科技公司,是以生产 RISC 微控制器(单片机)和 EPROM 存储器为主的半导体厂商。其产品在国际市场上有相当强的竞争能力,该公司近年来着力开发亚太市场,对我国单片机市场很感兴趣,该公司生产的 PIC16C5X 系列和 PIC17C42 单片机已开始向国内供货。本文将简要介绍其性能,供用户参考。

一、PIC16C5X 系列单片机(采用 CMOS 工艺,8 位,内含 EPROM)

## 1. 引脚及说明

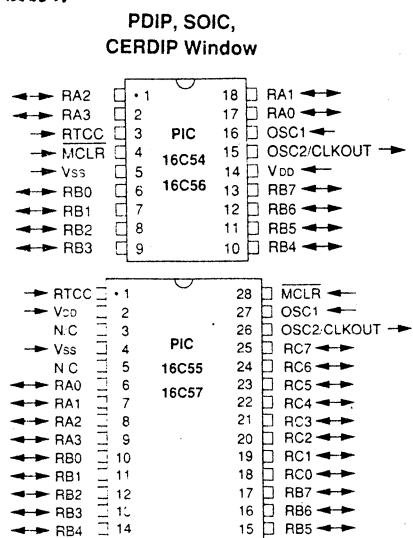


图 1

图中 PDIP 为塑料双列直插式封装;SOIC 为小引线封装;CERDIP,Window 为陶瓷双列直插式封装,开窗式。

现将芯片列表如下(电源电压:4.0~5.5V;工作频率:DC~20MHz;每种均有 PDIP,SOIC,CERDIP Window 三种封装方式)。

| 型号       | EPROM  | RAM  | I/O 口 | 引脚数   |
|----------|--------|------|-------|-------|
| PIC16C54 | 512×12 | 32×8 | 13    | 18PIN |
| PIC16C55 | 512×12 | 32×8 | 21    | 28PIN |
| PIC16C56 | 1K×12  | 32×8 | 13    | 18PIN |
| PIC16C57 | 2K×12  | 80×8 | 21    | 28PIN |

从图 1 中可以看出 PIC 系列单片机由于采用 RISC 技术,内部集成度很高,使得引脚十分简单,便于外电路的设计与使用。

(1)OSC1(输入),OSC2(输出) OSC1 可外接 RC,晶体振荡器或其他时钟电路。OSC2 在选用 RC 振荡器时,可用作 CLK 输出,其输出频率为 OSC1 输入频率的 1/4。

(2)RTCC(real-time clock counter,实时时钟计数器,输入) RTCC 是外加可编程控制时序。内部有 RTCC 暂存器,该暂存器可用程序设置和读取。用户可根据不同的外接时钟频率,利用程序设置 RTCC 和其他内部信号组成有关控制信号。

(3)MCLR 系统 RESET 信号。

(4)RA0~RA3,RB0~RB7,RC0~RC7(输入/输出) 用户编程的双向 I/O 口(RC0~RC7 只有 28PIN 的 PIC165X 才有)。

(5)VDD 和 VSS +5V 电源和地。

## 2. 内部结构及特点:

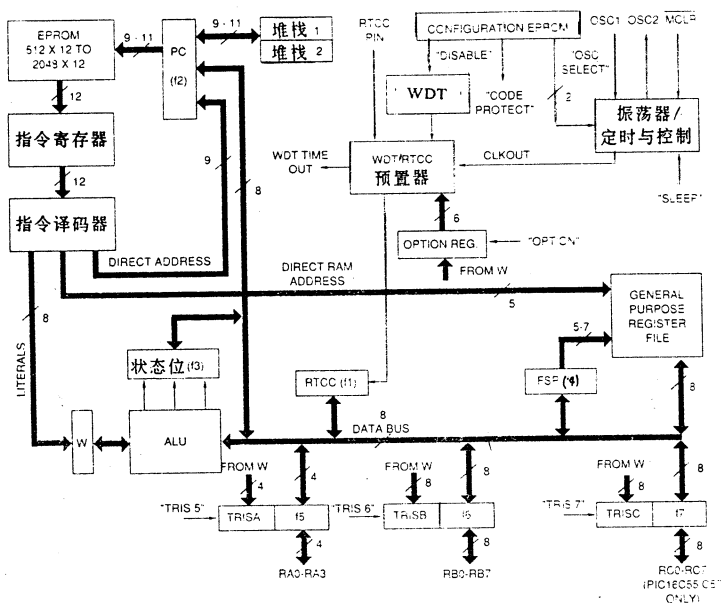


图 2

(1)从图 2 可以看出,PIC16C5X 单片机内部除含有一般单片机的通用结构如 ALU,PC,指令寄存器,指令译码器,状态寄存器等之外,还有一些特殊的结构。如通用寄存器阵列,实际上是一个可寻址的  $32 \times 8$ (或  $80 \times 8$ )的 RAM。该 32 个寄存器分别以 F0~F31 表示。均可用指令直接寻址,其中 F0~F7 是单片机本身专用的寄存器。如 F2 是起 PC+1 作用的,F5 是设定 RA0~RA3 口性质的寄存器等等。F4~F31 可用作 RAM。

(2)二级堆栈。

(3)有监视计时器功能(Watchdog timer;WDT)它是芯片内部用来巡视工作状况的“看门狗”。其作用是当该计时器计数到终点时,会自动使单片机 RESET,并重新启动。其作用是可防止在不正常的情况下“死机”或进入无限循环,提高芯片的抗干扰能力。也可以不使用此保护功能。

(4)有 33 条指令;指令字长 12 位;单周期,指令周期为 DC~200ns。可进行位操作;有直接,间接和相对寻址方式。

### 3. EPROM 的保密

PIC16C5X 内含 EPROM。EPROM 的编程需用专用的 PIC-PAK-I, PIC-ICE 或 PICMASTER 等仿真开发系统。

EPROM 中有四个专用位,其中一个为保密位,当程序写入后,将此位烧录为“0”,这样,任何人用任何器材都不能拷贝或读出 EPROM 中的程序。另一位是 WDT 的使能位,将此位烧录为“0”,则无 WDT 功能。其他二位是用来选择振荡方式的。

EPROM 中还有 16 位留给用户专用,只能在 PIC 仿真开发系统中读写这些位,与程序的运行无关。

PIC16C5X 系列中采用无窗口 PDIP,SOIC 封装的,其中的 ROM 用户可以用仿真器自己烧录,亦可加保密位,外人不能读出和擦除。因而它们适合定型的产品选用。

### 二、PIC17C4 高性能 8 位 CMOS EPROM 单片机

PIC17C42 是新推出的一种性能高于 PIC16C5X 系列的单片机。在增强的功能方面主要有:

- (1)55 条指令,多为单周期指令,指令字长 16 位,指令执行周期 250ns。有直接、间接、相对和立即等寻址方式。
- (2)寻址范围  $64K \times 16$  片上 2K。(可外接 2K~64K)。
- (3)片内 EPROM  $2K \times 16$ 。
- (4)通用寄存器  $232 \times 8$ ,其中有 48 个专用寄存器,其余可用 RAM。
- (5)16 级堆栈。
- (6)11 级中断。
- (7)33 个 I/O 口。
- (8)二个高速 PWM 输出(10 bit,15.6KHz)。
- (9)带有波特率发生器的 UART 口。
- (10)有 EPROM 保密位。
- (11)44PIN PLCC(塑料无引线芯片载体)封装。和 P44PIN PQFP(塑料四边引线扁平封装)。图 3 为 40PIN 双列直插封装。

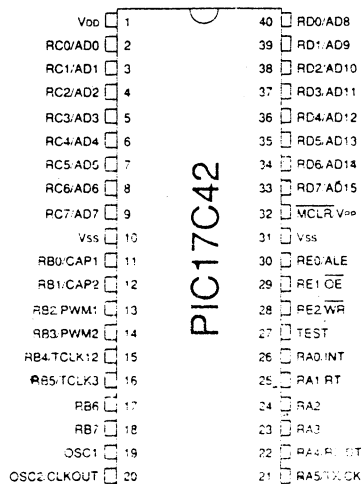


图 3

### 三、专用仿真器

Microchip 公司为开发 PIC16C5X 系列,PIC17C42 单片机推出多种仿真器。

1. PICPAK-I——PIC16C5X 普及型微控制器开发系统,该系统主要有:

- (1)EPROM 编程器(仅适用于 PIC16CX 系列),可以同 PC 机连用。
- (2)PICALC 宏汇编器,可实现源码到目的码的转换。
- (3)PICSIM 模拟器,可对 PIC16C5X 进行指令级模拟。支持 PC 机的 MSDOS 系统。

2. PIC-ICE——PIC16C5X 在线仿真开发系统

(1)EPROM 编程器,可独立也可同 PC 机或 286,386 及 486 连用。仅适用于 PIC16C5X 系列。

(2)宏汇编器。

(3)仿真器系统,用于 PIC16C54,16C55 和 16C57 的在线仿真。

3. PICMASTER-16——通用在线仿真器

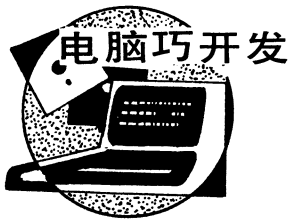
(1)EPROM 编程器,使用于 PIC16C5X 系列和 PIC17C42。

(2)宏汇编器。

(3)仿真器,可用于对 PIC16C5X 和 PIC17C42 的在线仿真。

适用在 Microsoft Windows 3.0 环境下的 286,386 和 486 机连用。

通过以上介绍,相信有兴趣的用户,对 Microchip 公司的 PIC16C5X,PIC17C42 单片机及其开发仿真系统会有一个概貌性的了解,相信会有越来越多的用户选用 Microchip 公司的单片机来开发自己的产品。



# 自带 EPROM 类单片机 简易加密编程方法的探索

河南周口地区水科所(466001) 庄 凯

计算机软件设计人员都为软件的加密工作搅尽了脑汁。但是,任何软件加密的方法,都不能绝对保护被加密软件不被非法拷贝或破译。在各种单片机应用系统中,最简单、可靠的加密方法是采用具有保密位的、自带 EPROM 程序存储器的单片机芯片。这类芯片,一旦将保密位编程(即加密编程),系统将禁止采用任何外部方式访问片内程序存储器。然而,对这类单片机芯片加密编程,往往要具有专用编程器或万用编程器,这对于只有普通开发设备的地方开发此类芯片是有困难的;另一方面,各类单片机技术手册、开发机手册以及许多单片机应用资料,只说明编程方式的要求,并不详细说明编程与加密的关系、以及实现的方法步骤,这使得许多人在开发过程中容易以失败而告终。因此,将会大大影响这类单片机的普及应用。本文拟以 MCS-51 系列 8751H 单片机为例,探索采用普通开发机对其加密编程的简易方法步骤。

## 一、创造一个可编程、加密的硬件环境

一般普通开发机大多数不具备对 8751 芯片编程的条件,往往只能对普通 EPROM 芯片编程,因此,我们要先创造一个硬件环境。

首先,根据 8751H 编程方式表(见附表)、时序图以及其他条件(参阅单片机技术手册),在具有 2764 芯片编程器的基础上(可以是任何芯片编程器,这里仅以 2764 为例),制造一个适宜于 8751 各种编程方式要求的简易硬件环境来。具体方法是比较 8751H 编程方式的要求条件和 2764 编程的一般条件:

1. 先找出共同点直接代用。例如:8751H 的 ALE 引脚在编程时,需要输入 50ms 负编程脉冲,而 2764 的 CE/PGM 引脚也需要输入 50ms 负编程脉冲,因此,可以用 2764 的引脚信号 CE/PGM 代替 8751 的 ALE 引脚的输入讯号。

2. 再找出不同点,使用简单电路满足其要求。例如:附表中 RST 引脚的编程要求必须满足  $2.5V \leq V_{IH} \leq V_{CC} + 0.5V$ , 否则不能编程,这在 2764 各引脚中是没有的,因此必须设法满足它,这里可以采用附加电阻分压的方法解决。

附图就是使用上述方法为 8751H 设计的简易编程板电路。图中,方框内是 8751 编程板,电路很简单,可制成一块小印刷电路板(甚至可以不用印刷板);方框外是 2764 的引脚名;将框内电路通过电缆和仿真插头插入任何编程器的 2764 插座上,即可对 8751H 编程。

另外还有一类开发机,如江苏省启东计算机厂生

产的 DVCC-51-ED 型开发机,自备对 8751 编程和校验的功能,但不能加密编程,这只要将原机中编程区里 8751 插座的 P2.6 引脚从接地处断开,象本文附图一样接入一个单刀双掷开关 K1,就可以方便的实现编程、加密编程的功能和转换。这类机器最好还要核查一下 8751 的 RST 引脚电压是否能可靠满足编程要求。

## 二、编程步骤及其之间的关系

将制成的 8751 编程板插入开发机编程器 2764 插座中,并把 8751H 芯片插入编程板,然后分别按照下面的步骤操作:

1. 把编程板上的开关 K1 拨向 A 端,启动开发机和编程板进行 8751 查空,在保证全空状态后,按照所用机型对 2764 编程的方法开始编程,直到正常结束编程。切记要保证  $V_{PP}$  应符合 8751 的要求。

2. 开关 K1 拨向 A 端,按照对 2764 的查空、转移、校验等方法可对 8751 进行相同的操作。例如编程后的校验等。

3. 在编程、校验合格后,将开关 K1 拨向 B 端,按照对 2764 编程的方法再次对 8751 编程,即保密位编程或加密编程。这个过程一瞬间即可完成。应该说明,不同的开发机编程软件其反应是不同的:对于边编程边校验的编程方式,第一个单元编程后立即呈现校验错误,编程停止,这时,加密编程已经完成;对于只编程不校验的编程方式,机器将一直编程到地址终端,但只有第一单元编程是有效的,其余时间是无效运行,所以当显示器显示过第一单元地址后,可立即停止编程,这时加密编程也已完成。应当注意,一旦完成这个步骤,便不能再经校验等方法验证,只能通过应用系统的运行来证明。

上述每个步骤的功能都是独立的,第一步只编程不加密,第二步可对 8751 进行编程以外的各种操作,第三步只对 8751 加密,因此,第三步只有在前两步先完成的基础上才是有效的,否则,仅进行第三步操作是无用的。

该方法同样适用于其他此类单片机简易编程板的设计。

8751H 编程方式表

| 方式 | RST      | PSEN | ALE | EA       | P2.7 | P2.6 | P2.5 | P2.4 |
|----|----------|------|-----|----------|------|------|------|------|
| 编程 | $V_{IH}$ | 0    | 0*  | $V_{PP}$ | 1    | 0    | ×    | ×    |
| 禁止 | $V_{IH}$ | 0    | 1   | ×        | 1    | 0    | ×    | ×    |

(下转 35 页)

# 一种简单的键盘接口电路

南京市1206信箱数控所(210012) 罗许建

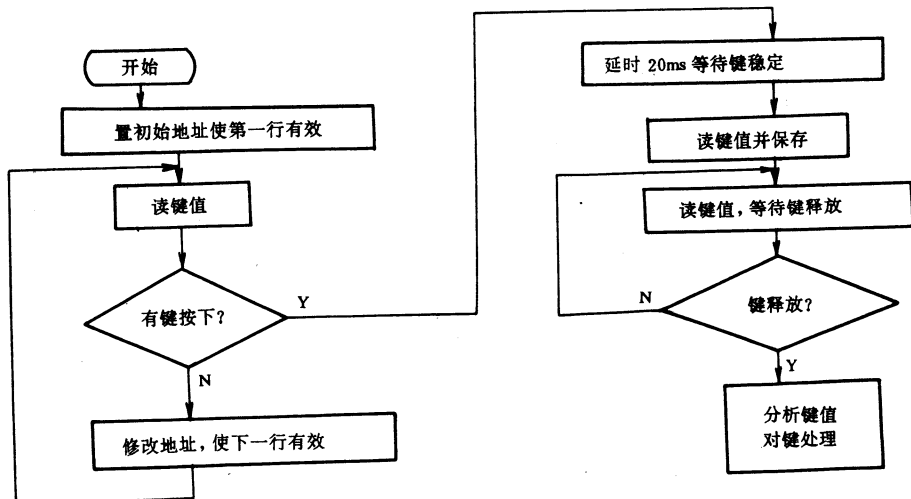
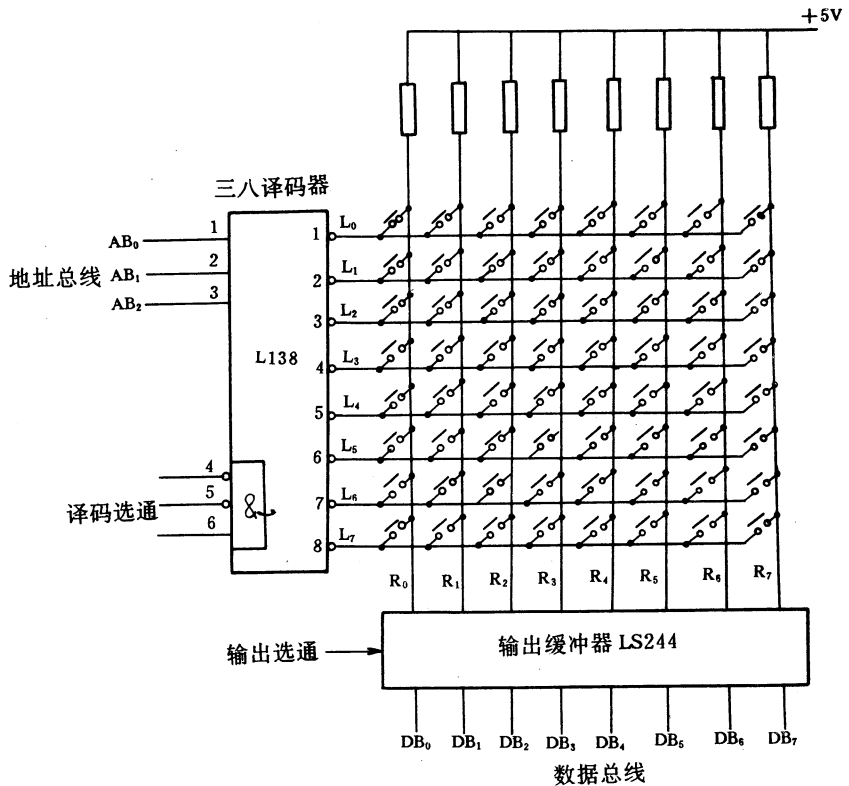
下面介绍一种简单的键盘接口电路,它由一个三八译码器和一个LS244缓冲器组成,构成一个由64个键组成的 $8 \times 8$ 矩阵。

如图,当三八译码器工作使得某行线有效时(低电平),如果这根线上有键被按下,则相应的列线为低电平,其它的列线为高电平。键盘扫描过程就是执行一系列的读操作,使每根行线依次为低,读列线内容。若读出数据中某位为“0”,则对应于该行、列的键被按下。这里的行线,列线都可以任意增加(增加一个三八译码器或输出缓冲器),因而扩展十分方便。

由于按键是机械结构,因而在键闭合的过程中会产生抖动。这样每按一次键,会产生不止一次状态变化,抖动时间一般在十几毫秒之内。一般用软件延时来等待按键的稳定,然后才读入键信息。

按键时,如果同时按下几个键,则读出的数据中有几个位为“0”。用户可根据实际情况对重键进行处理。

下面是键盘功能识别流程图:





## 电脑游戏机

# 第四章 FBASIC 语言的深入理解

山东苍山机械电子化学工业局(277700) 于 春

### 3. LEN

LEN 语句的功能是计算字符串的长度,即测量字符串共有多少个字符,其中包括空格、标点(除逗号)、运算符等。

语句格式为

LEN(字符串)

#### 例11.

```

10 I. "PLEASE INPUT";A $
20 A=LEN(A $)
30 P. "LEN("A $")="A
40 G. 10

```

RUN 显示

PLEASE INPUT? 键入 ABCD+3214

LEN(ABCD+1234)=9

..... ? 键入 A #B/"

LEN(A #B/" )=5

..... ? 键入 XYZABCDEFG

LEN(XYZABCDEFG)=10

...

### 4. LEFT \$ 和 RIGHT \$

LEFT \$ 语句的功能为从字符串的左边取出字符形成子字符串。其简写为 LEF.

RIGHT \$ 语句的功能为从字符串的右边取出字符形成子字符串。其简写为 RI.

语句格式为

LEF.(字符串,n1)

n1表示从字符串左边取 n1个字符。

RI.(字符串,n2)

n2表示从字符串右边取 n2个字符。

#### 例12.

```

10 A $ ="ABCDEFGF"
20 B $ ="1234567"
30 F. I=2 TO 6
40 P. LEF.(A $,I),
50 P. RI.(B $,I)
60 N.

```

RUN 打印

```

AB          67
ABC         567
ABCD       4567
ABCDE     34567
ABCDEF    234567

```

### 5. MID \$

MID \$ 语句在随机手册中关于语句功能的介绍为“取出某个字符的字和数。”有点令人费解,建议改为“取出某个字符串中指定位置和个数的字符组成子字

符串”。

MID \$ 简写 MI.

语句格式为:

MI.(字符串,取出位置,取出个数)式中:字符串最多可以有31个字符。取出位置均从左边数。

#### 例13.

```

10 A $ ="FCS-90 F BASIC"
20 F. I=1 TO LEN(A $) ST. 2
30 P. MID $ (A $,I,3)
40 N.

```

RUN 打印

```

FCS  CS-  S-9  -90
90F  0FB  FBA  BAS
ASI  SIC

```

从示例可以看出,MID \$ 语句兼备了 LEFT \$、RIGHT \$ 两个语句的功能,且比这两个语句使用更灵活、更方便。因此,一般使用 MID \$ 语句较多,而 LEFT \$、RIGHT \$ 两个语句使用的较少。

### 6. INSTR

INSTR 语句的功能是检索某一子字符串在另一字符串中的位置。

INSTR 简写 INS.

语句格式为

INSTR(被查串,子串)

该语句随机手册中没有介绍,请读者添上。

#### 例14

```

10 A $ ="ABCDEABC"
20 B $ ="BCDEA"
30 C $ ="DE"
40 B=INS.(A $,B $):
   C1=INS.(A $,C $):
   C2=INS.(B $,C $)
50 P. "B $ ="B,"C $ ="C1"- "C2
RUN 显示
B $ =2      C $ =4-3

```

#### 例15. 设计一个绘制心脏图形的程序。

```

10 CLS
20 A $ ="AIIIIIIIIIGIGGGGH"
30 B $ ="HHBHBEBEBEE"
40 K $ ="ABCDEFGHI"
50 I $ ="ABCGHIDEF"
60 X=13:Y=22:I=0
70 GOS. 100:SWAP A $,B $:GOS. 100
80 IF I=1 T. LOC. 0,0:END
90 IF I=0 T. SWAP K $,I $:I=1:G. 70
100 L=LEN(A $):F. P=1 TO L

```



```

110 K=P
120 IF I=0 T,140
130 K=L+1-P
140 C=INSTR(K$,MID$(A$,K,1))-1
150 V=C/3:H=C MOD 3
160 X=X+H+3*(H=2):Y=Y+V+3*(V=2)
170 LOC.X,Y:P.CHR$(207):N.:RE.
RUN

```

(打印出一心的图形。)

由于 F BASIC 语言没有画曲线的语句和函数(如正弦、余弦、求平方根等),因此,仅使用前三章介绍的语句画一心形将很困难。本例巧妙地使用了字符处理语句 INSTR、MID\$、LEN 就画出了心形。

为便于理解,特对程序说明如下:

本程序共分四部分:

20~60行为变量赋值。

100~170行为绘图子程序。

80 行为结束判断处理语句。

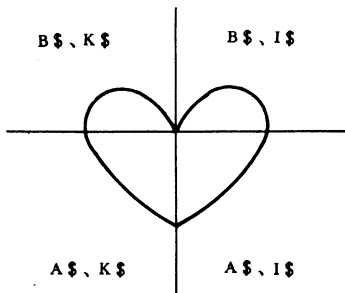
70、90两行为绘图控制语句,这两行是本程序的关键行。

程序的执行过程如下:

- (1) 变量赋值后,通过70行第一个 GOSUB 语句转绘图子程序,画出心形的左下部分。这时 A\$、K\$参与绘图。返回70行。
  - (2) A\$、B\$交换后,由第二个 GOSUB 语句转绘图子程序,画出心形的左上部分。这时 B\$、K\$参与绘图。
- 返回70行,执行90行。(由于 I=0 跳过80行)
- (3) K\$、I\$交换。令 I=1,返回70行。由第一个 GOSUB 语句转绘图子程序,画出心形的右上部分。这时 B\$、I\$参与绘图。返回70行。
  - (4) B\$、A\$交换后,由第二个 GOSUB 语句再转绘图子程序,画出心形的右下部分。这时 A\$、I\$有效。

返回70行,由于 I=1,执行80行程序运行结束。

若把心形分成四部分,各字符串在绘图中的有效区域如下图。



关于画图子程序的分析,读者可以列一张表,详细列出 P、K、C、V、H、X、Y、的变化,即可看出用字符串语句绘图的巧妙性和方便之处。

另外,上例中160行使用了特殊判断式(H=2)、(V=2)。这种判断式的值如下:(以 H=2为例)

当 H<2 (H=2)=0

当 H=2 (H=2)=-1

当 H>2 (H=2)=0

同样,若判断 H>2,则有

当 H<2 (H>2)=0

当 H=2 (H>2)=0

当 H>2 (H>2)=-1

即判断式以括号内为形,以判断内容为体。当判断内容成立时,判断式的值为-1;当判断内容不成立时,判断式的值恒为0。请读者记住这种判断式及使用方法,游戏程序设计中经常用到。

在随机手册、说明书及一般书籍中仅介绍了如何使用 F BASIC 语言进行计算程序设计和用卡通图案进行游戏程序设计,而忽略了绘图程序设计的介绍。这使不少使用者产生了误解,认为 F BASIC 语言不能设计绘图程序。其实,F BASIC 语言也有着很强的绘图功能。例15就是一个证明。尽管它看起来不如 BG GRAPHIC 方便,但还有胜过 BG GRAPHIC 的方面。特别是 BG GRAPHIC 在 Y 轴方向上有下面三行不能绘图,从而使绘图有效面积缩小了13%,这就限制了它的使用。而用程序绘图则纵横自如,不受限制,充分利用了有效画面。作为 F BASIC 语言的提高篇,再介绍几个设计程序绘制图案的实例,以开阔思路,使读者掌握设计绘图程序的方法和技巧。

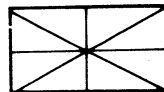
例16. 编制程序绘制米字形图案,用方框围起来。

```

10 CLS
20 F.X=0 TO 25
30 LOC.X,0:P."*"
40 LOC.X,21:P."*"
50 LOC.X,10:P."*"
60 LOC.X,(21-21*X/25):P."*"
70 LOC.X,21*X/25:P."*"
80 N.
90 F.Y=0 TO 12
100 LOC.13,Y:P."*"
110 LOC.0,Y:P."*"
120 LOC.25,Y:P."*"
130 N.

```

RUN  
打印



例17. 编制绘出长方体图案的程序

```

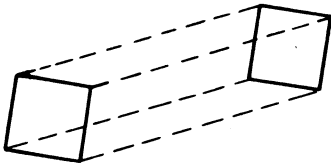
5 CLS
10 F.X=0 TO 9
20 LOC.0,X:P."*"
30 LOC.X,0:P."*"
40 LOC.9,X:P."*"
50 LOC.X,9:P."*"N.
60 F.X=17 TO 26
70 LOC.X,9:P."@"
80 LOC.X,18:P."@"N.

```

```

90 F. Y=9 TO 17
100 LOC. 17, Y:P. "@"
110 LOC. 26, Y:P. "@" :N.
120 F. X=1 TO 16
130 LOC. X, 9 * X/17:P. CH. (205)
140 LOC. X, 9 * X/17+9:P. CH. (205):N.
150 F. X=10 TO 25
160 LOC. X, 9 * X/17-4:P. CH. (205)
170 LOC. X, 9 * X/17+5:
    P. CH. (205):N.
180 E.
RUN    打印(如图2)

```



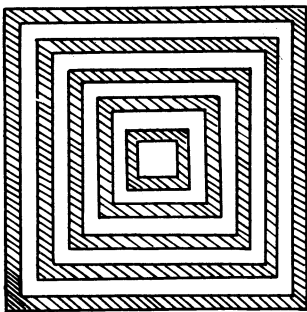
为便于理解,上两例均使用了单循环。也可以采用多重循环以简化程序。读者可以自己练习。

**例18.** 编程绘制五个正方形套在一起的图案。

```

10 CLS: X=13: Y=11: A$=CH. (253)
20 F. N=1 TO 10 ST. 2
30 F. L=-N TO N
40 LOC. X+L, Y-N:P. A$
50 LOC. X+N, Y+L: P. A$
60 LOC. X-L, Y+N: P. A$
70 LOC. X-N, Y-L:P. A$
80 N. : N.
90 E.
RUN    打印(如图3)

```



以上程序可以再加一段程序,使方框从里圈陆续显示,显示出全部图像后,再从外圈开始陆续消失,全部消失后再陆续显示。如此,周而复始,产生较强的动感。读者可自己练习。

还可以编制绘制五角星、八角形、园等图形的程序。由于篇幅所限,不一一列举,读者可自己练习。

### 五、出错处理语句

F BASIC 语言共有程序出错处理语句五条,随机手册中均未介绍,请读者自己添上。

在一个较复杂的游戏中,不可能把各个方面都考虑周全而编制相应的判断处理程序(那样程序将变得很庞大)。但在游戏过程中,情况千变万化,可能出现如出界、溢出等错误。出现错误时,计算机将停止运行,给出出错提示,从而使游戏中断。往往刚玩出一个好的积分,因游戏中断再从头开始,很令人扫兴。若使用出错处理语句,就可以消除这一弊病。使用出错处理语句后,当程序运行出错时,计算机自动转出错处理子程序,处理后,再返回继续下面的程序运行。这样可以把错误处理在程序的运行中,即自动处理。因此,引入出错处理语句,方便了程序的运行。

#### 1. 出错转移语句(ON ERROR GOTO)

ON ERROR GOTO 简写 O. ERR. G.

ON ERROR GOTO 语句的功能是设置错误转移入口,以便对错误进行拦截处理。语句格式为:

ON ERROR GOTO 行号

式中行号为出错处理子程序的首行号。

该语句的意思是,当程序出错时转去执行 X X 行。

#### 2. 错误处理返回语句(RESUME)

RESUME 简写 RESU.

RESUME 语句的功能是通知 BASIC 从出错处理子程序返回,并恢复执行原程序。

语句格式为

RESUME [NEXT][行号]

当程序中设计了 ON ERROR GOTO 语句后,在程序出错时,则令 BASIC 转去执行出错处理子程序。处理完后要返回到原程序,使用 RESUME 语句可以实现返回。返回语句有三种情况:一种是 RESUME 语句后跟 NEXT,返回时,程序就会在出错语句的下一行恢复运行;第二种是在 RESUME 语句后跟行号,返回时,程序就会在规定的行号恢复运行;第三种情况是省略 NEXT 和行号,程序就会返回到原出错语句,由于原出错语句有错,又转出错处理子程序,从而进入死循环状态。所以,使用时要牢记不要省略 NEXT 和行号。

#### 例19.

```

10 CLS
20 O. ERR. G. 70
30 I. "Please Input"; X
40 A=10000/X
50 P. "10000/"X"="A
60 G. 30
70 P. "X="X;
80 P. "0 Bu Neng Zuo Chu Shu"
90 RESU. 30
RUN

```

Please Input? 键入 30

10000/30=333

Please Input? 键入 0

X=0 0 Bu Neng Zuo Chu Shu

Please Input? .....

从上例运行可以看出,当输入0后即打印零不能

做除数,可继续输入。虽然出错,运行并不停止。这就是出错处理语句的作用。

### 3. 测试出错行号语句(ERL)

ERL 语句的功能是给出最近一次出错语句所在的行号。这是为测试出错行号而设置的语句。语句格式为:

ERL

### 4. 测试出错代码语句(ERR)

ERR 语句的功能是给出最近一次出错语句的出错代码。语句格式为

ERR

每一类错误都有一个确定的代码。错误的类型不同,出错代码也不同。当程序没有错误时,ERR=0(注:

表一 常用错误代码表

| 错误类型代码 | 错误表示简写 | 错误说明                  |
|--------|--------|-----------------------|
| 0      | NF     | 无 FOR 却有 NEXT         |
| 1      | SN     | 语法错误                  |
| 2      | RG     | 无 GOSUB 却有 RETURN     |
| 3      | OD     | DATA 语句中的数据不足         |
| 4      | IL     | 语句的叙述或称呼错误            |
| 5      | OV     | 演算结果超限(溢出)            |
| 6      | OM     | 内存不足。                 |
| 7      | UL     | GOTO、GOSUB、IF 的行号未定义。 |
| 8      | SO     | 排列变数的添字在规定外。          |
| 9      | DD     | 排列设双重定义。              |
| 10     | DE     | 用零做除数。                |
| 11     | TM     | 变数的类型不一致。             |
| 12     | ST     | 字符串超过31个。             |
| 13     | FT     | 式子过于复杂                |
| 14     | CC     | CONT 无法继续执行程序。        |
| 15     | UF     |                       |
| 16     | MO     | 参数所需要的指令未指定。          |
| 17     | TP     | 无法从磁带读出资料。            |
| 18     | NR     |                       |
| 19     | RE     |                       |
| 20     | NB     |                       |
| 21     | UP     |                       |

0同时也是一类错误代码)。如某一类错误发生时,ERR便记录下这个错误的类型代码。当又有新的错误发生时,ERR 又会变成新的错误类型代码。即 ERR 中所存的总是最近一次发生的错误的代码。F BASIC 共有22个错误类型代码,下表分别列出各错误类型代码所代表的错误信息简写和部分说明。

掌握了每种错误代码所代表的错误类型,就可以设计相应的错误处理子程序。

### 5. 定义出错代码语句(ERROR)

ERROR 简写 ERR.

ERROR 语句的功能是定义用户自己的出错代码。

在程序设计中,有时要根据出错类型做相应的转移处理。而往往有些人设定的错误在 BASIC 系统中并未设定,如 X 不能大于1000或者 X 不能等于500等。这些实际上都不是错误,不过是在程序设计中一些特定场合的需要而人为设定的。这些错误没有设定的错误代码,因此必须由用户自己设定。

ERROR 语句的格式为

ERROR n 式中:n取值0~21。

例20. 设定出错代码的演示。

```

10 CLS:ON ERROR GOTO 100
20 I. "Please Input";X
30 IF X>1000 T. ERR. 10
40 IF X=500 T. ERR. 15
50 IF X<100 T. ERR. 20
60 P. "OKOK":G. 20
100 IF ERR=10 T. P. "X>1000"
110 IF ERR=15 T. P. "X=500"
120 IF ERR=20 T. P. "X<100"
130 P. "ERL="ERL,"ERR="ERR
140 RESU. 20
RUN
    Please Input? 键入 1234
    X>1000
    ERL=30      ERR=10
    Please Input? 键入 500
    X=500
    ERL=40      ERR=15
    Please Input? 键入 200
    OKOK
    Please Input? 键入 50
    X<100
    ERL=50      ERR=20
    Please Input?
    :
    ~~~~~

```

## 消 息

电子工业出版社《电子与电脑》杂志编辑部将于8月31日在北京举办为期四周的微型机二级维修技术学习班。

培训内容:PC/XT/AT(8088、80286)主板、软硬盘及适配器、显示器及适配器、打印机及适配器、键盘、主机电源、UPS 的工作原理、维修方法,设有典型故障演示课以提高实际维修能力与技巧。

培训费:390元 资料费:70元(实报实销)

联系人:中国人民大学信息中心(100872)胡野红  
电 话:255. 5431-2518(或2514)



## 维修打印头应注意的几个问题

西安532信箱(710061) 蔡世清

打印头是打印机的关键部件之一,也是一个精密的机电组合件。打印机使用久了,断针是打印头中常见的故障。根据我的体会,修复断针时应注意以下几个问题。

1. 对于有断针的打印头,不宜仅补穿上好针完事,而应该拆除全部打印针,将空头仔细进行清洗(用酒精棉球),彻底清除其中的油污。对拆下的打印针,也要逐根用酒精棉球将其上面的油污擦净。擦洗时要格外细心,以免把针折断。这样装出来的打印头如同新的一样。

2. 穿针前先拿一根新针做标准,把拆下的旧针一一与之对照,比较长度,并去掉长度不够的旧针。做此项工作时,一定要仔细、准确。同时要注意,新针往往比

未磨损的旧针长,头脑中要留出余量,以免把合格的旧针也剔除掉。

3. 穿针时宜将增补的新针穿在同一边附近位置。有的打印针(如 TH3070R2)其新旧长度之差约2mm,穿好后露出部分过长,如用油石或细砂布打磨,则易引起折断。对此,宜在穿针前用手术剪轻轻剪掉1mm左右,余下部分待穿好后再用油石或细砂布打磨校正。

4. 用油石或细砂布打磨打印头时,开头用力要很轻。宜将细砂布或油石平放于玻璃板上,打印头一定要和细砂布或油石面垂直,切勿歪斜。用力要逐渐加大,不能用猛力。当全部打印针与陶瓷定位片在同一平面上时,这个打印头就算装好了。

## TH3070点阵式打印机常见故障分析与检修(下)

国务院办公厅秘书局技术处(100017) 刘立华

### 【故障实例五】印字连点故障

这个故障现象和“印字漏点”刚好相反,它的表现形式是在不该打印的地方,仍打印了,某一针(或几根针)连续出针在打印的字形中出现了一条连续的横线,这种故障完全出在电路上,检查和排除比较简单。

和“印字漏点”故障的检索一样,首先要确定是哪一针(或几针)连续,然后参照“印字漏点”故障中的针与各芯片的对应关系,检查六片 TD62064和三片 LS374。若是确实判定这几个芯片没有问题和话,再检查另外一个 LS374(IC46)。这个故障的范围,就在这十个芯片之内,而且这种故障一般都是出在 LS374芯片上。

### 【故障实例六】打印不出字故障

这种故障的表现形式为在自检或联机打印时,字车正常走动。走纸也正常,唯独打印无字形,即打印头不出针,检查这种故障的流程图如图6所示。

在上边流程图中,实际上包含着三个方面的检查内容。依次是7.5中断故障,出针控制信号故障及出针驱动信号的故障,图中“打印头插头插好”与“打印头线

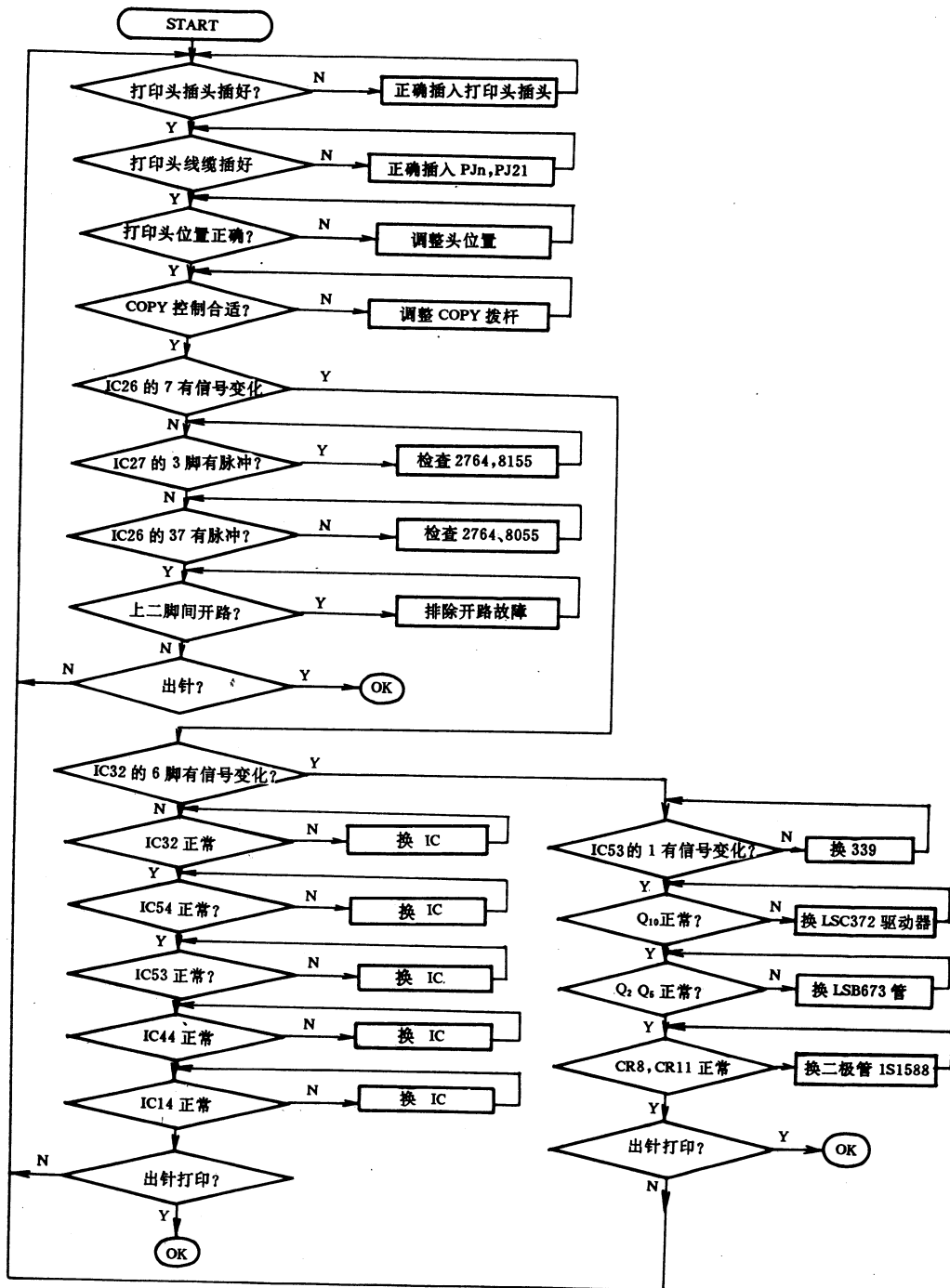
缆插好”两步,是很重要的一个检查内容,如果插得不对,最严重的后果将会是烧坏打印头的线圈。

电路检查中,IC54的型号为“555”,被称作“时基电路”,主要作为时间控制用,检查这个芯片,主要是检查一下它的第三脚有无信号变化就可以了,其它脚的输入与输出都与第三脚是密切相关的,也就是说,如果 IC53(339)的第二脚和第十四脚有信号变化,而“555”(IC53)的第三脚无信号变化,则说明“555”芯片已经坏了。

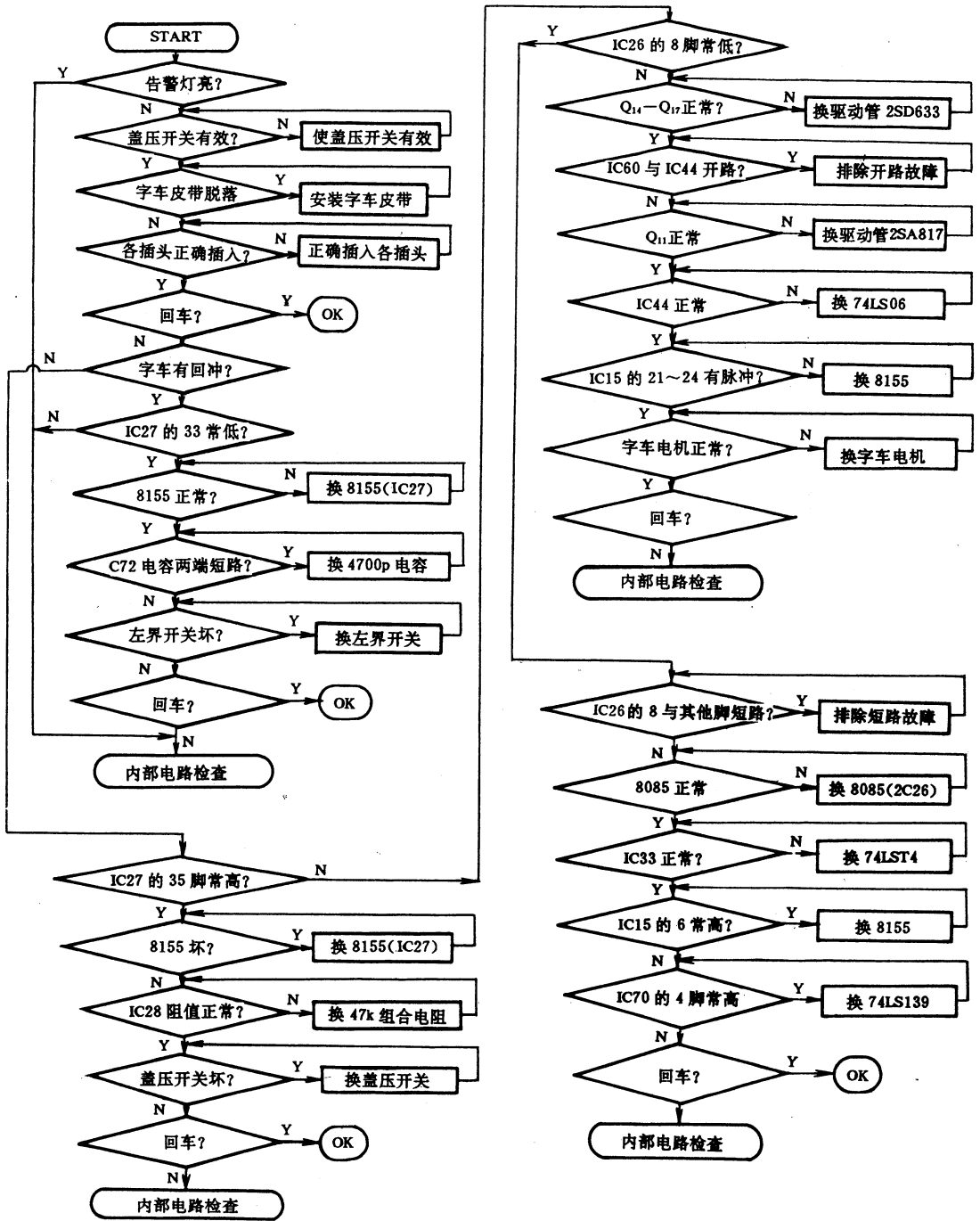
### 【故障实例七】加电后不回车故障

打印机加上电而打印字车不动或不回到“左界”是打印机最常发生的故障,这种故障由于其牵涉面比较广,因而隐蔽性也比较强,所以检查起来也比较棘手。

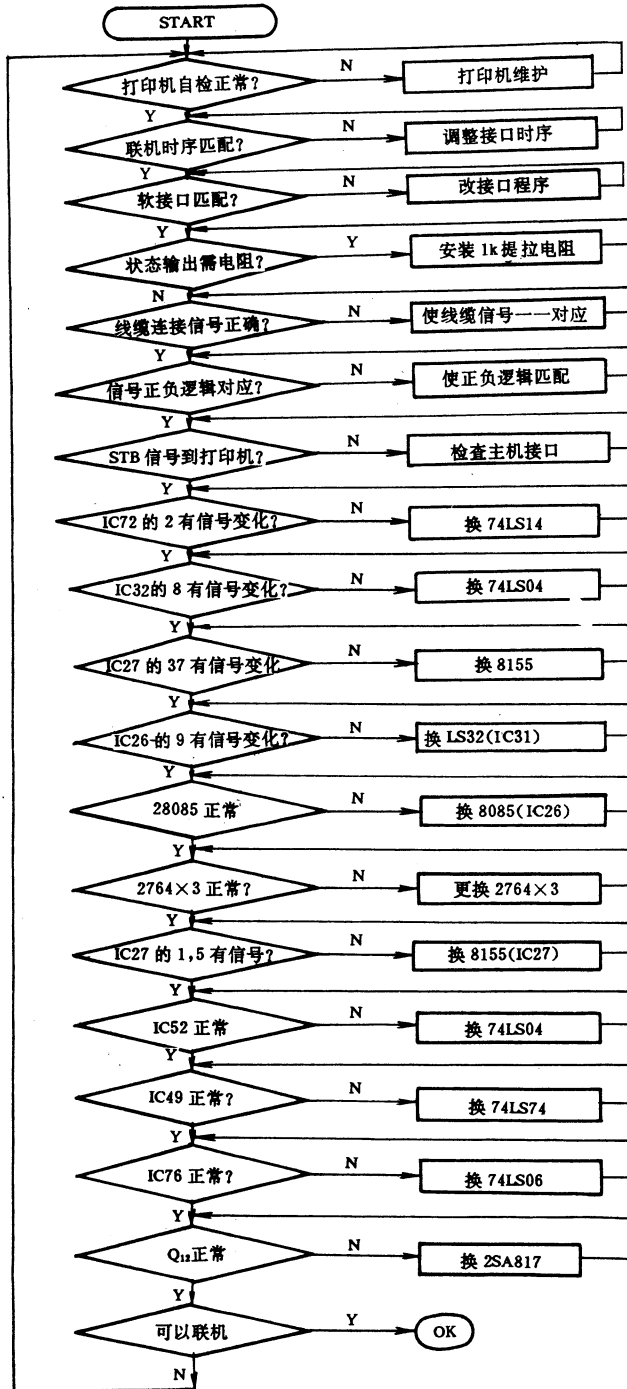
在检查这种故障时,首先要检查一下,看看保险丝是否已熔断。若换好保险丝仍然熔断,则说明电源有毛病。若电源没问题,那么,还需检查一下打印机控制板上的插头是否插错、插反或未插,在确认以上检查无误后,即可进入电路部分的检查,加电不回车的检查流程图如图7、图8所示。



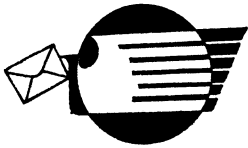
图(6)“打印不出字故障”检修逻辑图



图(7) “加电后不回车故障”检修逻辑图



图(8)“并行接口不能联机故障”检修逻辑图



# 普及型 PC 个人用户软件交流联谊活动 问题解答(六)

读者联谊

北京中国农科院计算中心(100081) 王路敬

## 17. CGA、MCGA、EGA、CEGA、CMGA、VGA、TVGA 各表示什么含义?

它们分别代表微机系统显示器视屏标准。

CGA 是 Color Graphics Adapter (彩色图形适配器) 的缩写。这是 IBM PC 系列机一种显示标准, 该标准有 7 种工作方式, 其中 4 种字符方式, 3 种图形工作方式, 它们是:

- 40×25 字符方式 2种颜色
- 40×25 字符方式 4种颜色
- 80×25 字符方式 2种颜色
- 80×25 字符方式 4种颜色
- 320×200 图形方式 4种颜色
- 320×200 图形方式 2种颜色
- 640×200 图形方式 2种颜色

IBM PC/XT、GW0520A 等机即为这种显示标准

MCGA 是 Multi—Color Graphics Adapter (多色图形适配器) 的缩写。它是低档 IBM PS/2 系列机的视屏标准。它除能提供 CGA 所有工作方式外, 增加了下述两种图形工作方式:

- 640×480 图形方式, 2种颜色
- 320×200 图形方式, 256种颜色

EGA 是 Enhanced Graphics Adapter (增强型图形适配器) 的缩写。EGA 能仿真 CGA 的所有功能, 支持 CGA 的所有工作方式, 并且新增加了下列几种工作方式:

- 80×25 字符方式, 2种颜色(单色)
- 320×200 图形方式, 16种颜色
- 640×350 图形方式, 16种颜色
- 640×350 图形方式, 4种灰度(单显)
- 640×350 图形方式, 16种颜色

CEGA 是 Chinese Enhanced Graphics Adapter (中文增强型彩色显示系统) 的缩写。该显示标准与 CGA 全兼容。同时增加了与 EGA 全兼容方式, 640×350 图形方式颜色由 8 种提高到 64 种, 并做到了与 VGA 的 640×480 高分辨率图形方式兼容。

CMGA 是 Chinese Monochrome Graphics Adapter (中文增强型单色多灰度显示系统) 的缩写。这种显示标准与 Hercules 单色方式 (720×350) 全兼容, 与 CGA 彩色显示方式全兼容, 具有 16 级灰度, 可自动把彩色转换成灰度, 增加了 640×480 单色高分辨率图形方式。

VGA 是 Video Graphics Array (视屏图形阵列) 的缩写。这是 IBM 公司为其 PS/2 系列中高档机设计的一种高性能视屏标准。VGA 与 EGA 高度兼容, 它能支持

EGA 提供的所有工作方式, 并且增加了下列工作方式:

- 640×480 图形方式, 两种颜色
- 640×480 图形方式, 16种颜色
- 320×200 图形方式, 256种颜色

自从 IBM 新的图形标准 VGA 卡推出以后, 各种高性能的高分辨率彩卡不断问世, TVGA 卡就是其中的一种。TVGA 卡不仅支持 CGA, EGA, VGA 的图形标准, 而且提供了比 VGA (640×480) 标准更高的视频分辨率 (1024×768) 和图形功能。采用 1024×768 高分辨显示模块进行 CAD 一类的图形处理, 无论是图形质量还是工作效率都有较大的提高。一般 TVGA 卡配置到处理速度较快的 386 机上。由于 TVGA 卡性能高, 价格便宜, 因此, 在新购置的各类微机中, 都可按照用户的要求配置这种彩卡。

## 18. 使用显示器应注意哪些问题? 常见故障有哪些?

首先要注意的是显示器的电源开关, 有些显示器没有电源开关, 它是和主机电源开关在一起, 开启主机电源时, 显示器电源也随之被打开。不过大多数显示器电源不和主机在一起, 它有自己的电源开关。在计算机系统启动时, 一般都要先打开显示器开关, 然后再开主机开关, 以防止瞬间电流脉冲影响主机, 关闭时则要求先关主机, 然后再关显示器。

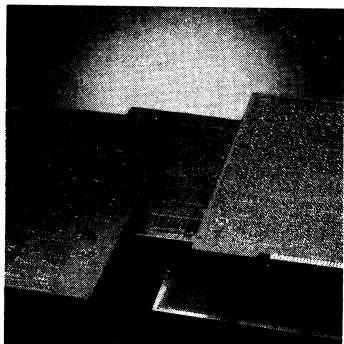
第二, 有些是专门的显示器可供不同型号或种类的微机使用, 因此, 它们背面可能有选择开关, 使用时要把系统配置选择开关拨到相应的位置, 否则屏幕上出现杂乱无章的信息。

第三, 显示器一般都要有亮度、对比度、色彩等调节选配, 我们可以根据自己的爱好, 调节选择一定亮度、对比度和色彩等。

在使用中常见故障: 屏幕一片黑, 无任何显示内容。碰到这类故障首先要区别是系统本身的故障还是显示适配器或是显示器本身的问题, 可根据故障表现出的其他现象逐个判别。例如 PC 机单色显示器若加电后屏幕上没有任何显示, 随即听到一长二短的鸣叫声, 这说明可能是系统板上的 5、6 两位开关设置不对或显示板问题。屏幕滚动也是常见故障之一, 这种现象一般是显示器的帧频不同步, 调整显示器的垂直同步旋钮即可恢复; 如果调整无效, 就需要检查适配器的输出波形是否正常了。屏幕有时出现杂乱无章的字符或图形。出现这种故障现象的实质就是数据在传送过程中

(下转 10 页)





一九九二年  
 总期第88期

# 電子與電腦

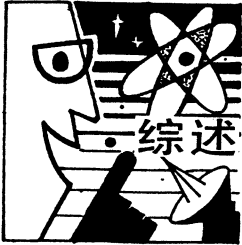
• ELECTRONICS AND COMPUTERS •

## 目 录

|                                    |                                    |
|------------------------------------|------------------------------------|
| • 综述 •                             | PC BASIC 自试测题解答与分析 ..... 李志刚(25)   |
| 优越的专家系统语言 Prolog ..... 王粤宁(2)      | • 学用单片机 •                          |
| • PC 用户 •                          | BJS-51 单片机实验系统 ..... 盛焕鸣(29)       |
| 结构文件与 FIELD()函数的比较                 | • 学装微电脑 •                          |
| ..... 严桂兰 刘甲耀(4)                   | 步进电机的控制 ..... 易齐干(33)              |
| 24 小时服务系统的人员安排 ..... 陈君佐(6)        | • 电脑巧开发 •                          |
| ARC 系列压档工具软件的应用 ..... 戴青松(7)       | ASCII 码字符显示器设计 ..... 李德文(38)       |
| DOS 3.30 若干新增命令的功能与应用 ..... 柳见成(9) | • 电脑游戏机 •                          |
| WPS 的彻底解密 ..... 瞿新国(10)            | 第三章 F BASIC 的画面控制语句 ..... 于 春(41)  |
| 处理系统不认硬盘时应注意的问题 ..... 陈 栋(11)      | • 维修经验谈 •                          |
| • 学习机之友 •                          | Super AT 机显示电源原理及维修                |
| CEC-I 彩色 TV 字幕 ..... 包 敢(12)       | ..... 范志盛(45)                      |
| BASIC 程序中变量名使用情况的列表印出程序            | 利用 PCTOOLS 校正软盘驱动器磁头 ..... 黄焕如(46) |
| ..... 蔡 伟(14)                      | • 读者联谊 •                           |
| 磁头清洗及其辅助程序 ..... 唐汉雄(15)           | 普及型 PC 个人用户软件交流联谊活动问题解答(七)         |
| 求法雷数列的新方法 ..... 廖庆平(16)            | ..... 王路敬(48)                      |
| TOOL-KIT 使用详解续 ..... 姜 宏(17)       | 封一: 线路板                            |
| • 语言讲座 •                           | 封二: MOTOROL 公司单片微机一览表              |
| 6502 机器语言程序设计                      | 封三: 同上                             |
| 第七章 循环程序设计 ..... 朱国江(20)           | 封四: BJS 系列单片机教学实验系统                |
| • 初级程序员级软件水平考试辅导 •                 |                                    |

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:杨逢仪  
 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



# 优越的专家系统语言 Prolog

空军航空工程部(410124)王粤宁

专家系统是人工智能中最活跃的组成部分之一。专家系统是具备相当丰富和权威性的知识(知识库)、采取一定的推理策略,为解决具有专家级的适当规模的实际问题的计算机系统。专家系统还具有学习机制,可对知识库进行改进,以增进解决问题能力。知识库和推理机制是构成一个专家系统的最核心部分。由于 Prolog 语言具有强大的推理功能,已经作为构建专家系统最自然的工具。用 Prolog 语言编写的专家系统与一般程序设计语言和数据库比较,在许多方面体现出明显的优越性。

与一般程序设计语言的比较

专家系统可以表示为:“知识+推理=系统”,较之“数据+算法=程序”的一般应用程序有观念性改变。以下通过对程序一与程序二进行的比较,可以看到专家系统语言 Prolog 的优越性。程序一是用 Prolog 语言编写的一个安排某班九岁同学乒乓球赛的微型专家系统。它由事实、规则两大部分构成。程序二是实现同一功能用 BASIC 语言编写的程序。

程序一:predicates

```
pupil(symbol,integer)
clauses
pupil(Peter,9)
pupil(Chales,9)
pupil(Paul,10)
pupil(Susan,9)
match(P1,P2):-pupil(P1,9),pupil(P2,9),
P1<>P2,write(P1,P2),fail.
```

程序二:10 INPUT N

```
20 DIM P$(N)
30 J=0
40 FOR I=1 TO N
50 READ X$,Y
60 IF Y<>9 THEN 30
70 J=J+1
80 P$(J)=X$
90 NEXT I
100 FOR K=1 TO J
110 FOR L=K+1 TO J
120 PRINT P$(K);P$(L)
130 NEXT L
140 NEXT K
150 DATA Peter,9,Charles,9,
Paul,10,Susan,9
160 END
```

程序二的框图见后页。

1. 专家系统语言 Prolog 与一般程序设计语言解决

问题的方式有着根本的差异。一般程序设计语言必须为待解决的问题设计算法,才能由系统实现。在用 BASIC 语言编写的程序二中,为了解决确定参赛选手配对问题,预先设计了算法:挑出有资格参赛选手,将选手两两配对。但 Prolog 语言不需要预先设计解决问题的算法,只要在程序中给出系统目标,利用系统内部的模式匹配,回溯搜索等功能,便可自动寻求问题的解答。比如在用 Prolog 语言编写的程序一中,只要给出目标:将有资格参赛的不同两选手配对,系统便可给出答案。Prolog 语言的这一特点根本改变了一般程序设计语言进行程序设计的思考方式,使得程序设计简单易行。

2. 专家系统语言 Prolog 对问题的求解是基于事实、规则的逻辑推理过程,用其设计的程序具有简洁性。如程序一中只用了一条规则:

```
match(P1,P2):-pupil(P1,9),pupil(P2,9),P1<>P2,write(P1,P2),fail.
```

便可自动实现系统要求的两两选手配对功能。这是因为 Prolog 语言作为一种专家系统语言,是以面向自然语言的谓词逻辑作为语言基础的,其本身就是规则的描述,使程序设计更接近于人的思维方式,结构清晰易懂、易读。而 BASIC 是过程型语言,使用时要告诉计算机如何一步一步地解决问题,明确表达出完成一种计算所必须的控制流程的一切细节,因此写出的程序显得冗长繁琐。实现同样的功能,程序二中使用了三个循环十六条语句才实现,编程的耗费大大超过了程序一。

3. 专家系统具有学习机制,容易对知识库进行修改。而一般程序设计语言针对某一具体问题的程序,一旦编写调试完毕,其功能就确定下来,不易更改,如若要在程序一中增加一个系统目标,只需要增加一个命题(或一条规则)即可。而程序二中却要做很大改动,“牵一发动全身”,有时甚至整个程序要重写,重新构造算法。因而一般应用程序的应变能力和扩展能力远远不如 Prolog 语言。知识的更新,要求一个好的系统应具有较强的灵活性,这一点上 Prolog 语言明显优于一般程序设计语言。

与数据库比较

Prolog 语言所能处理的问题不仅包括了数据库所能处理的范围,而且大大超过了它。程序三是用 dBASE III 编写的程序,具有同程序一相同的功能,将程序三与程序一比较,可以看出 Prolog 语言的优越性。

程序三:use class.dbf

```
accept "Enter Pupils' Number" to N
I=1
```

```

do while I <= N
if Age = 9
replace MAT with 1
endif
skip
store I + 1 to I
enddo
store I to L
count for MAT1 = 1 to M
go top
locate for MAT = 1
do while M > 0
do while MAT1 = 1
continue
enddo
replace MAT1 with 1
store NAME to X
continue
do while .not. EOF
@L,5 say X
@L,15 say NAME
store L + 1 to L
continue
enddo
go top
locate for MAT = 1
M = M - 1
enddo
return

```

程序三在执行过程中对以下 DBF 文件进行操作：

class. dbf:

|   | NAME    | AGE | MAT | MAT1 |
|---|---------|-----|-----|------|
| 1 | Peter   | 9   | 0   | 0    |
| 2 | Charles | 9   | 0   | 0    |
| 3 | Paul    | 10  | 0   | 0    |
| 4 | Susan   | 9   | 0   | 0    |

文件中字段 MAT 是存放具有比赛资格的同学标记“1”(无资格参赛者标记为“0”),MAT1 的数字“1”或“0”分别表示已参加过“配对”处理的记录。

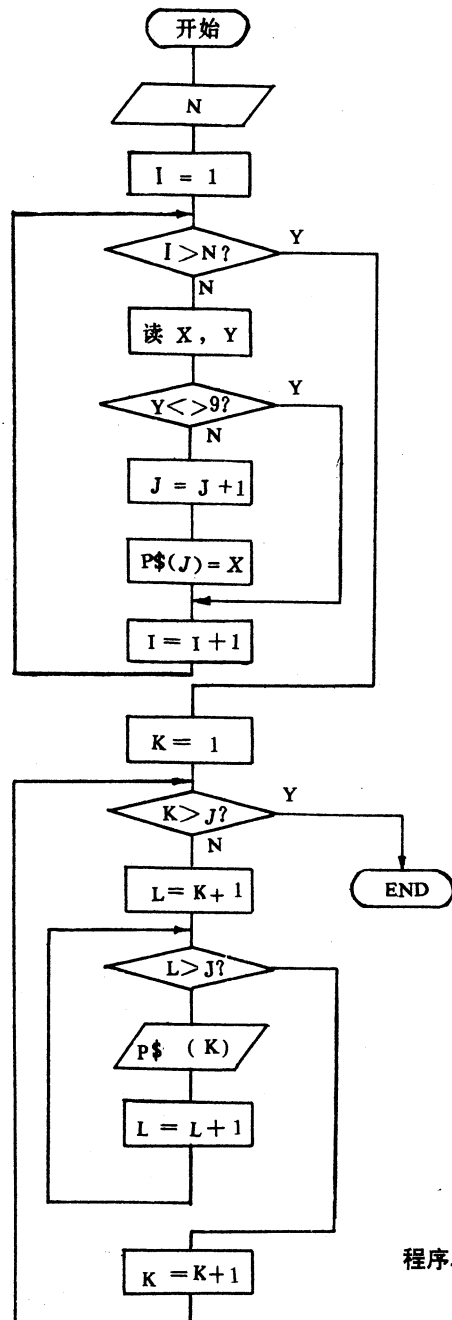
1. 数据库管理系统的目标是对大批量数据进行管理,主要对数据进行存取、增删、统计等操作。而 Prolog 语言除了具有上述功能外,还建立知识库,并与推理网络联接起来,由系统自行处理那些有关知识,最后得到问题的结论,具有推理功能,这一点是数据库系统无法比拟的。因而,专家系统语言 Prolog 被称为“智能数据库”。

2. 数据库语言 dBASE III 仍属于过程型语言,编程复杂、冗长、功能的修改不方便。如程序三中用了三十一条语句来实现程序一中一条规则的功能,其表达能力远远低于 Prolog 语言。

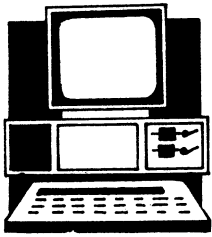
3. 用 Prolog 语言编写的专家系统,由于其程序设计高效简洁,使得其运行速度快,而 dBASE III 数据库系

统的处理时间长,工作效率低。

由于 Prolog 语言的程序设计思想,处理问题的手段,所处理的对象完全不同于常规的事务处理程序设计语言,它以更接近于人们思维方式的设计思想来处理数据,以巧妙的搜索技术(Prolog 中的目标直接深度优先搜索——回溯)及模式匹配执行程序等处理问题的手段,以专家的领域专门知识(而不是固定不变的数据)作为处理对象,使用 Prolog 语言设计的程序具有常规程序设计语言和数据库系统无可比拟的优越性。



程序二框图



PC 用户

# 结构文件与 field() 函数的比较

福建华侨大学(362011)严桂兰 刘甲耀

当使用数据库管理系统来实现各种事务管理时,最大的优势就是数据与程序的相互独立,多个功能模块在设计时丝毫不考虑数据库的结构与具体数据,而数据库的建立与修改除考虑实际事务情况外,对程序并无要求与影响,因而,各功能模块的独立性与适应性是十分明显的,使该模块在实现其功能时,对任一数据库都可通用。为达到此目的,在程序设计中,不论是 dBASE III 还是 FoxBASE 系统,都要用到宏替换函数 &。除此之外,对于中文管理信息,由于各系统对数据库字段名长度都有限制,若直接使用中文字段名,往往会遇到麻烦。为此,多采用代码字段,这就要求各代码有相应的中文显示,但又不失原字段代码的数据处理,在此前提下,对 dBASE III 来说,关键手段是使用结构文件,而对 FoxBASE 则采用 field() 函数,使程序设计的模块不受具体数据库的限制。

## 一、建立字典库,以查找代码字段的对应中文信息

为了显示各用户数据库代码字段的中文含意,可采用字典库的办法,字典库中以库名(KM)作为联接关键字,对照字段(ZD)数据则按代码库中字段顺序加入中文信息,字典库中可以复盖形式装入当前代码库的字段中文,也可装入多个代码库的字段中文,例如 DZ 字典库

```
.use dz
.list
RECORD # KM ZD
1 RU 姓名
2 RU 性别
3 RU 年龄
4 CK 品名
5 CK 规格
6 CK 数量
7 CK 厂家
:
```

当使用某一数据库时,由于数据库的库名已知,从而在 DZ 库中可滤出对应数据库的中文字段,对应不同的系统软件,其实现的手段也各有所异。

## 二、dBASE III 中的结构文件

在 dBASE III 管理系统中,结构文件的产生是依附于任一已知数据库文件,其结构由系统提供,它由 FLELD-NAME、FLELD-TYPE、FIELD-LEN 和 FLELD-DEC 四个固定字段组成,如 CUST.DBF 结构文件所示:

```
.use cust
.list stru
字段 字段名 类型 宽度 小数
```

```
1 FIELD-NAME 字符型 10
2 FIELD-TYPE 字符型 1
3 FIELD-LEN 数字型 3
4 FIELD-DEC 数字型 3
```

\*\*总计\*\* 18

其记录则是已知数据库的结构,不同的数据库,CUST 的结构永远不改变,仅是其记录随之改变,不同记录就可得到不同的事务管理的内容,从而达到同一程序对不同的事务而可完成同样的功能,现以输入功能模块为例说明其实现过程:

(1)产生结构文件,取出各代码字段存入宏替换数组之中。

```
:
acce"请输入数据库名:"to wj
sele 1
use &.wj
copy to cust stru exte
sele 2
use cust
m=1
do while .not. eof()
k1=field-name
if m<10
u="x"+str(m,1)
else
v="x"+str(m,2)
endif
&.v=k1
skip
m=m+1
endd
:
```

不同的数据库输入数据库名后,立即产生结构文件 CUST,然后通过宏替换函数 & 形成的数组取出 CUST.DBF 中各记录所含的字段名。

(2)取出字典库中各代码字段的中文信息。

```
:
sele 3
use dz
loca for km="&.wj"
m=1
do while .not. eof()
k2=dz
if m<1
u="y"+str(m,1)
else
u="y"+str(m,2)
```

```

endi
&u=k2
m=m+1
cont
endd
:

```

由于已知数据库名,就可以在字典库中找到对应的中文字段,用函数 & 形成另一数组加以保存。

### (3) 输入记录

在执行上述程序段之前,已建成用户各自的管理数据库,例如人事数据库(RU.DBF),仓库数据库(CK.DBF),订单数据库(DD.DBF)等。

```

.use ru
.list stru
字段 字段名 类型 宽度 小数
1 A1 字符型 6
2 A2 字符型 2
3 B3 数字型 2
总计 11

```

则执行上述程序段时,首先在提示“输入数据库名:”处,按下 ru 或 ck 或 dd..., 为下列输入程序段作了必要的准备,使下列程序段执行时,既有对应中文字段显示,又在用户所指数据库中输入了一条新的记录。

```

:
sele1
appe blan
m1=1
clea
do while m1<m
if m1<10
v="x"+str(m1,1)
u="y"+str(m1,1)
else
v="x"+str(m1,2)
u="y"+str(m1,2)
endi
k1=&.v
k2=&.u
@ m1,1 say k2+":" get &k1
m1=m1+1
endd
read

```

例如下列事务管理:

|      |      |       |
|------|------|-------|
| 人事   | 仓库   | 订单    |
| 姓名:- | 品名:- | 订单号:- |
| 性别:  | 规格:  | 客户:   |
| 年龄:  | 数量:  | 型号:   |
| 籍贯:  | 单价:  | 交货期:  |
| :    | :    | :     |

“-”为光标等待位置。

不同的中文字段提示,就会输入相应数据。上述各程序段并不会因数据库结构不同或者数据库结构的改变而改变,不同的数据库表现在结构文件的记录在发生变化,各数组元素对应的内容相应变化,仅此而已。反过

来,程序的变化,也不会影响各数据库的变化。

### 三、FoxBASE 中的 field() 函数

在 dBASE III 中,虽然用结构文件能很好地解决程序模块的通用性问题,但因不同的数据库都要通过一个结构文件来取得相应的字段名变量,加上 dBASE III 中没有数组语句,而是借用 & 函数形成数组,使用 & 函数虽很方便,但其替换时间影响程序执行的速度,而在 FoxBASE 系统中,由于有 field(m) 函数,它直接给出当前数据库中 m 值指定位置的字段名变量,因此,用 FoxBASE 实现同样输入功能时,就省去了产生结构文件的过程,这在时空上都胜过 dBASE III 一筹;另一方面, FoxBASE 可用 dimension 语句直接建立数组变量,省去了 & 函数替换时间。对于不同的数据库,其字段名个数不等,其数组长度也要随之改变,为此,程序中要有自动测定不同字段个数的办法,用数组元素存取代码字段的中文含意就显得十分方便了。

```

:
acce "请输入数据库名:" to wj
sele 3
use dz
set filt to km="&.wj"
n=reco()
set filt to
dime a(n)
loca for km="&.wj"
m=1
do while .not. eof()
a(m)=dz
m=m+1
cont
endd
sele 1
use &.wj
appe blan
m=1
L=1
clea
do while L<>0
k1=field(m)
@m,1 say a(m)+":" get &k1
m=m+1
L=len(field(m))
endd
read
:

```

### 四、结构文件与 field() 函数的比较

结构文件中的 field name 字段与 field() 函数的作用十分相似,由它们都能得到用户数据库的字段名,并借助此法来实现程序模块的通用性。除上述比较外,在作用时还有一些差异。例如:

```

use cust
@m,1 say field.name get field.name
read

```

执行时不会出错,它能够通过结构文件来修改用户数据库中的字段名。而

```

:
use ru
@1,1 say field(1) get field(1)
read

```

此程序段形式上与上程序段相似,但执行时会出错,这是因为 get 语句后要求变量而不是表达式,因此,field(1)函数不能直接用在 get 语句之后,如若改

为:

```

use ru
s=field(1)
@1,1 say field(1) get &s
read
:

```

执行时,会修改用户数据库当前记录第一个字段数据,因此,即使 field.name 与 field(1)有些相似,但在使用时,有本质上的差别。

## 24 小时服务系统的人员安排

汕头四中(515031) 陈君佐

宾馆,酒楼,工厂,医院等 24 小时连续工作系统,都想用较少的员工,做较多的事,达到提高经济效益的目的。

例 1 某饭店日夜营业,每个员工连续工作 8 小时,经测定,0~4 点需员工 4 人;4~8 点需员工 8 人;8~12 点需员工 10 人;12~16 点需员工 14 人;16~20 点需员工 12 人;20~0 点需员工 7 人。试求所需工作人员的最少人数,并给出排班表。

下面是我在 IBM-PC 微机上编写的“工作安排”程序:

```

10 REM "WORKAP. BAS"
20 INPUT A1,A2,A3,A4,A5,A6
30 X=A2+A4+A6
40 Y=A1+A3+A5
50 H=ABS(X-Y)
60 PRINT
70 PRINT X,Y,H
80 PRINT
90 FOR X1=0 TO A1
100 X2=A2-X1
110 IF X>Y THEN X3=A3-X2+H
 ELSE X3=A3-X2
120 IF X<Y THEN X4=A4-X3+H
 ELSE X4=A4-X3
130 X5=A5-X4
140 X6=A6-X5
150 IF X1<0 OR X2<0 OR X3<0
 OR X4<0 OR X5<0 OR X6<0
 THEN 200
160 M=M+1
170 PRINT M,X1,X2,X3,X4,X5,X6
180 PRINT
190 S=X1+X2+X3+X4+X5+X6
200 NEXT X1
210 PRINT "SUM=";S,X,Y,H
220 END

```

当 RUN 运行程序,送入 6 个时间段所需员工数 4,8,10,14,12,7

电脑立即显示,需员工 29 人,其中 3 人是富余人员,安排在上午 8 点来上班,在工作最忙时洗洗菜,为客人送饭菜,洗盘碗。

同时,给出下列 5 种人员上班表:

| 该<br>上<br>班<br>排<br>班 | 时<br>段    |     |      |       |       |      |
|-----------------------|-----------|-----|------|-------|-------|------|
|                       | 0~4       | 4~8 | 8~12 | 12~16 | 16~20 | 20~0 |
|                       | 来 上 班 人 数 |     |      |       |       |      |
| 1                     | 0         | 8   | 5    | 9     | 3     | 4    |
| 2                     | 1         | 7   | 6    | 8     | 4     | 3    |
| 3                     | 2         | 6   | 7    | 7     | 5     | 2    |
| 4                     | 3         | 5   | 8    | 6     | 6     | 1    |
| 5                     | 4         | 4   | 9    | 5     | 7     | 0    |

例 2 某工厂,0~4 点需员工 5 人;4~8 点需员工 8 人;8~12 点需 200 人;12~16 点需 200 人;16~20 点需 50 人;20~0 点需 48 人;问该厂最少需员工多少人? 怎样合理排班?

同上可知,需员工 256 人,有 6 种排班表:

|   |   |   |     |   |    |   |
|---|---|---|-----|---|----|---|
| 1 | 0 | 8 | 193 | 7 | 43 | 5 |
| 2 | 1 | 7 | 194 | 6 | 44 | 4 |
| 3 | 2 | 6 | 195 | 5 | 45 | 3 |
| 4 | 3 | 5 | 196 | 4 | 46 | 2 |
| 5 | 4 | 4 | 197 | 3 | 47 | 1 |
| 6 | 5 | 3 | 198 | 2 | 48 | 0 |

例 3 某厂,6 个时间段,需要人员为 5,8,25,14,12,17.这是较难安排上班的单位,用电脑,也能给出下列 1 种合理安排。知员工需 42 人,富余人员 3 人,上班表为:

1 0 8 17 0 12 5 再没比此更合理的安排了。

本程序也适用于非连续工作的系统!

# ARC 系列压档工具软件的应用

陕西西安电子科技大学 125 号信箱(710071) 戴青松

用过 Turbo Pascal 5.5 的用户,列表该软件包的目录,不难发现其中有不少扩展名为 .ARC 的文件,它们是一些文件的集合,称作“压缩文档文件”(ARC 文件)。

压缩文档文件的出现不是偶然的。当前,计算机硬件价格急剧下降,使得计算机的配置愈来愈齐全,普通出售的计算机内存容量都在 640KB 以上。内存的扩大,减少了编写软件的束缚,因而现今推出的软件性能愈趋完善,而所占的磁盘介质也越来越多,动辄几张、十几张磁盘,不但占用空间大,而且携带安装不方便。因此,不少软件公司在推出软件时,就已将其中一部分文件压缩存储,待安装时再展开,使用携带十分方便。如 Borland 公司近期推出的 Turbo 系列软件,MS 公司的 WINDOWS,Novell 公司的 Netware 等,其中的一些文件均进行了压缩。

对于普通计算机用户来说,利用压缩文档文件管理自己的文件系统更为可取。这是因为用户手中一般均收集有相当数量的优秀软件,但经常使用的不过几种,大多数软件均存储在磁盘上束之高阁,占用了大量磁盘空间。如果事先对这些软件进行压缩归档,就可以节省近一半的磁盘容量,经济效益十分明显。并且由于文件减少,用户更便于管理。另外,大多数压缩文档软件支持口令字数据加密,保证了数据安全。

目前,国际国内流行的压缩文档软件工具有多种,如 ARC; ZOO; PAK; PKARC; PKZIP; PCSECURE; LHARC 等。这些压缩软件由于推出时间的不同以及压缩算法的差异,性能差别很大。表一为实际使用这些软件进行文档压缩的结果,所用机型为 8MHz AT 机。根据笔者实际使用的经验,以及表一的结果,我们向大家推荐三种高性能的文档压缩工具: LHARC、PKZIP 及 LZEXE。

LHARC 是日本 89~91 年产出的高性能压档软件,至今已发展到 2.1 版(91 年 3 月 3 日出品)该软件短小精悍,一个仅 30 多 K 的程序就完成了文件归档、展开、列表、删除、检验、生成自展开文件等全部功能。最吸引人的是,由于其采用的 LZHUF 算法先进,在诸多压档软件中, LHARC 的压缩比最高。

直接运行 LHARC 不带参数, LHARC 就自动显示其 HELP,这个规律亦可应用在其他压档软件上。(事实上,尽管各压档软件性能不同,参数也不一样,但基本命令 A、E、D、V 等均是兼容的)

LHARC 的命令格式为:

```
[d1:][path] LHARC <COMMAND> <Archive. [LZH]>
[d2:][path][FileNames]
```

其中: Archive. [LZH] 指压缩后生成的文档名,扩

展名省略则默认为 .LZH。

FileNames 指被压缩的源文件名,必须写齐扩展名,但可用通配符 \* 或 ? 代替。

LHARC 的 COMMAND(命令)很多,但常用的只有几种,下面一一分别介绍。

## 1. 文件归档命令 A

例: A>C:\LHARC A C \*.C

该命令执行后,将首先从 C 盘上读入 LHARC(设 LHARC 在 C 盘根目录下),而后把 A 盘上所有扩展名为 .C 的文件压缩归档,存储在 A 盘上名为 C.LZH 的文件包中。若 A 盘上原先不存在 C.LZH 文件,则 LHARC 自动在 A 盘生成该文件;若 A 盘上已有同名文件,则 LHARC 把 \*.C 文件压缩后追加加入 C.LZH 文件中。

此命令执行后源文件 \*.C 仍存在于 A 盘上。

## 2. 文件归档并删除源文件命令 M

该命令的执行与 A 命令相同,但在生成归档文件包 Archive.LZH 的同时将源文件 FileNames 全部删除。

## 3. 文档列表命令 L 或 V

例: A>C:\LHARC L C.[LZH]

两个命令 L、V 均为归档列表命令,执行后将在屏幕上列出有关 C.LZH 包中所有归档文件的信息(源文件名;文件原始大小;压缩后大小;压缩比;文件生成日期;文件属性;CRC 校验码等),两命令的差别只在显示格式的不同。

## 4. 文件复原(展开)命令 E 或 X

例: A>C:\LHARC X C.[LZH]

执行该命令后,将把 C.LZH 中所有文件展开恢复为源文件 FileNames。若源文件 FileNames 已存在,则 LHARC 在展开该文件时自动跳过(在老的 1.X 版本中将询问是否覆盖),展开后的文件与相应原文件完全相同。并且该命令还能这样使用:

A)C:\LHARC E C BAUP.C

当 C.LZH 包中确实存在 BAUP.C 文件时,执行该命令则 LHARC 只展开 C.LZH 中 BAUP.C 一个文件。

## 5. 归档文件校验命令 T

源文件归档压缩后, LHARC 在压缩文件包中加入 CRC 检验码。为了确保文件内容不损坏丢失,在每次归档压缩后,用户可利用 T 命令对 Archive.LZH 文件的 CRC 码进行检验。

例: A>C:\LHARC T C

事实上,压档文件出现 CRC 错误的概率是很小的。

## 6. 文件自展开命令 S

例: A>C:\LHARC S C

用户在使用归档文件包 C.LZH 时,必须将其展

开,如每次均用 LHARC 很感不便,因此 LHARC 提供了一个自展开命令 S,执行后归档包 C.LZH 将自动转换成一个 C.COM 或 C.EXE 文件,将它拷入 C 盘直接执行,即无需 LHARC,C.EXE 将自动展开原 C.LZH 包中的源文件,使用十分方便。而生成的 C.EXE 只比 C.LZH 大 1K 左右。

LHARC 极为出色,可惜仍有不足:一是它不支持口令字数据加密,而这在数据安全中是相当重要的;二是它的工作速度较慢,特别在低速 PC/XT 机上感觉尤为明显。幸好,另一份压档软件——PKZIP 恰能弥补 LHARC 的这两个缺点。

PKZIP 是美国 PKWARC 公司 90 年推出的压档软件。它的最大特点是工作速度十分快。相对 LHARC 来说可节约 1/2~1/3 的时间,而其效率也仅比 LHARC 低 5%。因此在低速的 PC/XT 机上,推荐大家使用 PKZIP;而在高速的 286、386 机上,由于速度已不重要,因而最好使用 LHARC。

PKZIP 软件包内含数个执行程序。其中 PKZIP.EXE 主要用于压缩,PKUNZIP.EXE 主要用于展开,ZIP2EXE.EXE 用于形成自展开文件,PKZIPFIX.EXE 用于对压缩归档文件维护。PKZIP 的基本命令与 LHARC 相似,这里就不详细介绍了。需注意的是,PKZIP 命令参数前必须加一短杠“-”以便于 PKZIP 识别命令参数。

PKZIP 支持加密压缩方式,使用方法为:

例:A>C:\PKZIP[-A] -gCD45H C[.ZIP] \*.C

执行该命令后,PKZIP 把 A 盘上所有扩展名.C 的文件压缩归档,并送入密码字“CD45H”后存储在 A 盘 C.ZIP 文件中。此后当需要展开扩展名.C 的源文件时,也需要送入正确的密码字“CD45H”,否则 PKZIP 将拒绝展开。

例:A>C:\PKUNZIP-gCD45H C BAUP.C

由于送入了正确密码,PKUNZIP 执行展开操作,展开 C.ZIP 中的一个源文件 BAUP.C。如果省略源文件名,则 PKUNZIP 将把 C.ZIP 中所有文件展开。

表一 注:以下数据在 3.33MHZ AT 机上所得

|             | 原文件长度<br>(byte) | 压缩后文件长度 (byte) |         |         |            |           |           |            |        |
|-------------|-----------------|----------------|---------|---------|------------|-----------|-----------|------------|--------|
|             |                 | ARC5.2         | ZOOV1.0 | PAKV1.6 | PKARCV3.61 | PKZIPV1.1 | LHARCV2.1 | LZEVEV0.91 |        |
| 源<br>文<br>件 | FOX.BAT         | 19             | 19      | 19      | 19         | 19        | 19        | 19         | --     |
|             | FOXBIND.EXE     | 24512          | 17010   | 17520   | 16165      | 17205     | 13351     | 13040      | 13837  |
|             | FOXPCOMP.EXE    | 72480          | 49797   | 49421   | 45610      | 49184     | 38023     | 30200      | 37043  |
|             | MFOXPLUS.EXE    | 247808         | 173044  | 169436  | 154250     | 166839    | 128198    | 119754     | 105156 |
|             | PC430.EXE       | 171088         | 142631  | 137359  | 122356     | 136761    | 103580    | 99444      | 106667 |
|             | MFOXPLUS.OVL    | 138032         | 100994  | 69285   | 89768      | 98348     | 72321     | 69285      | -      |
|             | FOXPHHELP.HLP   | 149909         | 98477   | 75839   | 65055      | 73383     | 56295     | 54604      | -      |
| 压缩比         | 100%            | 28%            | 32%     | 39%     | 33%        | 49%       | 51%       | 49%        |        |
| 统共压缩时间      | —               | 5分02秒          | 1分31秒   | 2分20秒   | 0分33秒      | 1分47秒     | 2分33秒     | —          |        |

利用 LHARC 和 PKZIP 压缩不常用的软件是相当出色且实用的。但若利用它对常用软件压缩归档,则用户将感到不便。因为通过 LHARC 或 PKZIP 使用归档文件,首先必须在软盘或硬盘上展开,而后才能使用,这个过程是繁琐而无趣的。如果一份软件压缩后无需在磁盘介质中展开即能直接运行,那将是多么方便啊! ComputerLink Magazine 推出的 LZEXE 实用软件可实现这项功能。

LZEXE 使用格式:

C>LZEXE [ ] Filename[.EXE]

LZEXE 软件只对 EXE 可执行文件进行压缩,生成一个新的压缩过的 EXE 文件。这个新文件与源文件 Filename.EXE 同名,而源文件 Filename.EXE 则改名为 Filename.OLD 两者可同时存在。在新生成的 EXE 文件中,不但包含了源文件的全部内容(压缩),而且附加了一小段程序。当直接运行新 EXE 文件时,附加的程序段首先工作,在机器内存中展开源文件的内容,并试图在 DOS 控制下执行它。由于全部操作均在内存中进行,因而速度极快。据笔者经验,在较高速的 AT、286 等机型上,当从软驱运行文件时,压缩后的文件执行速度将比直接运行源文件更快!(这是因为压缩文件体积相对较小,减少了磁盘读取量,因而速度得以提高)而在普通带 Turbo 加速键的 PC 机上,两者执行速度相当。

LZEXE 只可对 EXE 文件压缩,应用范围较小。幸而该软件包还提供了一套 COMTOEXE,用于将 COM 文件转换为 EXE 文件,这样经过转换后 COM 文件也能压缩了。另外由于一般 EXE 文件均已经 Microsoft 公司的 EXEPACK 程序对文件头压缩过,因而 LZEXE 包还提供一套 UPACK 程序,利用它事先对 EXE 文件反压缩后再动用 LZEXE,压缩效果更好。

LHARC、PKZIP 及 LZEXE 三套工具软件性能极佳,且压缩文件内容完全忠实于源文件,用户可放心使用不必担心意外。当用户能熟练地操作它们时,将会赞叹其卓越的性能。



# DOS3.30 若干新增命令的功能与应用

长沙水电师范学院计算中心(410077)柳见成

自从 1981 年 IBM PC 系统问世以来,近十年时间内,所配置的 DOS 操作系统版本不断改进和更新,其中以 1983 年推出的 DOS 2.0 版最为著名。DOS 2.0 版的主要特点是采用树结构目录以适应硬盘配置,以及吸取了 UNIX 系统的 PIPE 机制,支持管道和 I/O 重定向操作等功能。DOS 2.0 的影响非常广泛,可以说,我国广大的 IBM PC 和 0520 系列微机用户都是在 2.0 的基础上掌握 DOS 操作系统使用的。1984 年公布适应 PC/AT 大容量软盘驱动器的 3.00 版,1985 年由于局部网络和 3.5 英寸软盘驱动器的需要相继推出了 3.10 和 3.20 版本。1987 年问世的 3.30 在兼容以前各版本的基础上又增加了一些新的功能。对单用户单任务情况而言,3.30 已是一个比较成熟的版本,因而目前在 286 和 386 系列微机上被广泛地应用。本文结合工作中的实践经验,介绍 3.30 版中对一般用户很有意义的若干新增命令(相对于 2.0 版而言)的功能和应用实例。

## 1. 安装虚拟磁盘

3.30 提供的 VDISK.SYS 是一个设备驱动程序,用于将内存的一部分作为存储介质来仿真磁盘驱动器。由于虚拟盘以内存速度运行,因此远快于访问硬盘的存取操作。安装 VDISK.SYS 的语句应包含在 CONFIG.SYS 中,并在重新启动 DOS 后才起作用。

格式:DEVICE=[d:][path]VDISK.SYS ddd sss mmm/e

格式中三个选择参数分别为虚盘容量、扇区大小和目录数。选/e 可在超过 1M 的扩充内存中安装虚盘。

在目前基于 8086/8088 CPU 和基于 80286 CPU 的微机中,尽管寻址空间可达 1M 和 16M,但 DOS 管理的自由空间最多为 640K。汉字系统工作时如装入字库,则难以运行大型程序。例如标准 FoxBASE+ 需要至少 375K 内存,最大程序执行时可达 560K。在 CCDOS 2.13F 下即使只装入一级字库(130K 左右),一般命令和程序也无法运行。虽然可将全部字库驻留硬盘来保证 FoxBASE+ 的内存要求,但调用汉字时将大大降低运行速度。此种情况下如果主存储器扩展到了 1M 或 1M 以上,就可利用 DOS 3.30 的设置虚盘功能来解决这一矛盾。方法如下:

用行编辑或字处理程序在配置文件 CONFIG.SYS 中加一条命令:

```
DEVICE=VDISK 384 128 64/E
```

重新启动 CCDOS 2.13F 后,键入屏幕上的选择 3,

即可将两级字库装入虚盘 D,能正常运行 FoxBASE+ 程序且保证快速调入汉字。

## 2. LASTDRIVE 命令

格式:LASTDRIVE=X

此命令用于设置用户能访问到的最大驱动器号,从而使用户可设置多于物理驱动器数量的逻辑驱动器。X 代表 DOS 可接受的最后有效驱动器字母,取值范围为 A~Z。此命令和 VDISK.SYS 一样应置于 CONFIG.SYS 文件中,在启动 DOS 时进行配置。

## 3. SUBST 命令

格式:SUBST d: d:path

此命令可使用不同的驱动器标识符来代替其它驱动器或路径。其中 d: 指定用于代替的驱动器字母,d:path 指定被代替的驱动器和路径。

在使用 WordStar 字处理软件时,由于 WS 4.0 以下版本不支持 DOS 的树结构路径,即使利用后面将讨论的 APPEND 命令可解决不同路径下共享字处理软件的问题,也不能直接建立当前目录以外的文件。解决办法是利用 SUBST 命令替换。例如,欲在 C 盘的子目录 PATH2 下建立文件 FILE2,设用 F: 代替路径 PATH2,则操作步骤如下:

```
C>SUBST F:\PATH2
```

然后进入字处理软件所在路径,运行该软件并在起始菜单中选择 D,在文件名后面键入 F:PATH2。此标识符能为 WordStar 所识别,经过 SUBST 命令转换后即可按指定路径存盘,因此可支持任意路径,在树结构层次较多的情况下更显得方便。事实上,对一切不能识别路径名的应用软件,均可应用这一方法。使用时应注意两点:①替换命令中指定驱动器字母不得大于 CONFIG.SYS 中 LASTDRIVE 的值,且不可与当前驱动器号相同。②指定路径时必须从根目录开始。

## 4. APPEND 命令

格式:APPEND[d:][path[;[d:][path...]]

此命令用于查找和装入在当前目录之外,且扩展名不为 .COM、.EXE 和 .BAT 的数据文件。以前的 DOS 版本只提供了 PATH 命令,仅能搜索可执行文件和批文件。对于某些包含覆盖文件的软件系统,如 WordStar 和 FoxBASE+ 等就无法作到让不同用户在不同子目录下共享,只能将这些软件分别拷贝到自己的子目录中,造成磁盘空间的大量浪费。利用 DOS 3.30 提供的 APPEND 命令可完全解决这一问题。

设硬盘上装有 CCDOS 2.1F 汉字系统、WordStar

和 FoxBASE+, 分别在子目录 \213、\WS、\FOX 下, 则可在 AUTOEXEC. BAT 中设置如下命令:

```
PATH C:\213;C:\WS;C:\FOX
APPEND C:\213;C:\WS;C:\FOX
```

系统启动后, 即可在任意路径下直接调用字处理和 FoxBASE+ 软件, 从而实现这些应用软件的真实共享。

还有一点应予说明, APPEND 查找数据文件的顺序与 PATH 相同。先在当前路径找, 未找到再按 APPEND 命令给出的搜索路径查找。因此当使用某些文本编辑程序(如 EDLIN)建立一个新文件时, 有可能调入一个位于搜索路径上的其它子目录下同名文件, 且对它的编辑和修改将被存入当前目录, 而对 COPY、RENAME、DEL 和 DIR 等命令则不起作用。

#### 5. ATTRIB 命令

格式: ATTRIB [+R/-R][+A/-A] <文件标识符>/S

用于修改指定目录中选择的文件或该目录层下全部文件的文件属性。ATTRIB 命令是 DOS 3.0 增加的, 但 DOS 3.30 进一步增强了修改文件存档属性的功能。[+R/-R] 为设置或取消指定文件的只读属性, 在一定程度上可保护文件不被任意修改和删除。[+A/-A] 为设置或恢复指定文件的存档属性。DOS 在文件目录表中为每个磁盘文件分配了一个 32 字节的“目录登记项”, 其中第 11 个字节为属性域, 通常其值为 20H, 表示该文件可被写入或修改。恢复该属性可影响 BACKUP/M、XCOPY/M 等操作。当设置存档属性时,

文件可被复制, 否则就不被复制。应用此命令为用户有选择地备份文件提供了一个有效的手段。实例见后。

#### 6. XCOPY 命令

格式: XCOPY <源标识符> <目的标识符> [/A][/D][/E][/M][/P][/S][/W][/V]

应用此命令可有选择地复制一组文件, 文件中可包含较低层次的子目录。格式中的源可以是一个驱动器, 一个路径, 一个文件名或三者的组合, 各可选参数的说明此处从略。

XCOPY 命令为 DOS 3.30 以前所没有。与 COPY 命令相比, 其特点在于:

① 可根据文件属性有选择地复制一组文件, 也可加上参数 [/P] 一个一个选择复制, 而不像 COPY 命令那样受通配符的限制。

② 可以复制树结构文件, 包括在复制的同时建立子目录。

例: 欲将 A 盘上除 .COM 和 .EXE 文件以外的其它文件全部复制到 C 盘上尚未建立的子目录 \AAA 下, 操作步骤如下:

```
C>ATTRIB -A A:*.COM /S
C>ATTRIB -A A:*.EXE /S
C>XCOPY A:*. *.* C:\AAA /S /M
```

在屏幕提示 (F=file, D=directory)? 后面回答 D 即可完成操作。

以上各命令的完整格式及详细说明均请参看 DOS 3.30 使用手册。

## WPS 的彻底解密

江苏如东县中学(226400) 瞿新国

香港金山公司推出的高级文字处理系统 WPS 2.0 因其功能强大、使用方便, 在国内使用很广。作者为了保护其成果, 采取了加密措施。运行时, 一旦机器上的日期超过 1991 年 2 月 4 日, 将破坏系统(删除 WPS EXE 及两个覆盖文件), 给使用者带来诸多不便。虽说可以在使用前将日期改成 1991 年 2 月 4 日前的日子, 但其本身的“当前日期复写到文章中”的功能失去了作用。很多软件需要机器上的正确日期。人们也不习惯于错误的日期; 在使用前一旦忘记修改机器上的日期, 就删除了系统, 又得重新恢复。

笔者现已找到一种将其彻底解密的方法。考虑到此软件通过判断日期决定是否删除文件, 程序中肯定要取机器日期, 取机器上的日期一般通过 2AH 号功能调用实现, 汇编指令为

```
MOV AH, 2A
INT 21
```

这两指令的代码是: B4 2A CD 21。我们只要用 DEBUG 工具的 S 命令找出软件中取机器日期的程序段加以分析, 就不难找出其做手脚之处。具体操作是:

```
REN WPS EXE WPS
DEBUG WPS
```

用 S 命令查找上述两指令代码的首地址: S100 LFF00 B4 2A CD 21, 发现软件有四个取时间的程序段, 再用 U 反汇编命令分别汇编分析, 发现地址为 XXXX:DB42 和 XXXX:E1A4 的两段程序与加密有关, 其中地址为 XXXX:E1A4 的一段用于显示警告信息, 地址为: XXXX:DB42 的一段用于删除文件。我们只需用 A 命令将两段程序中的 CMP CX, 07C7 (地址分别为 XXXX:DB46 和 XXXX:E1AB) 改成 RET 便使这两段程序失去应有的作用, 也就完成了解密工作。WPS 的其它功能均不受影响。

# 处理系统不认硬盘时应注意的问题

华南师大电教系(510631) 陈栋

日前笔者的一台 XT 机无法从硬盘启动,改用软盘启动后无法进入硬盘,出现“Invalid drive specification”的出错提示。笔者分析是硬盘的主引导记录受到病毒侵害所致,通过调用调试工具 DEBUG,用 INT13H 读出位于硬盘磁头 0 磁道 1 扇区的主引导记录,发现果然如此。于是从另一台完好的 XT 机的硬盘取出正确的主引导记录写入该机的引导扇区,关机启动,硬盘工作正常。

但当笔者运行硬盘上的部分文件时,却出现“File Allocation Table Bad”或“Sector Not Found Error”的出错提示,于是怀疑硬盘的文件分配表亦受到破坏。再次运行 DEBUG,将文件分配表副本拷入文件分配表中,退出 DEBUG 后运行硬盘上的那部分文件,依然出现相同的出错提示信息。

后经分析发现,问题仍然出在主引导记录所在的 0 磁头 0 磁道 1 扇区。这个扇区包含主引导记录块、硬盘分区信息表和自举有效标志字等内容。不能从硬盘启动的那台 XT 机的 20MB 硬盘只有一个分区,而完好的那台 XT 机的 20MB 硬盘却分成两个分区,因而 C 盘的容量就变小了。将好机的引导扇区的内容拷入坏机的引导扇区后,坏机的 C 盘容量也随之“变小”,导致硬盘上的部分文件被“拦腰切断”或“消失”,也就出现了文件分配表坏或没找到文件区的错误。根据这个道理,笔者将硬盘分区信息表中的终止地址和实用扇区数的值恢复正常后,那部分文件也都可以正常运行了。

系统不认硬盘的主要原因是主引导记录所在的扇区内容受到破坏,修复工作也要从这里着手。最方便的方法是从一台完好的微机硬盘上将该扇区的内容复制过来,但要注意由于两个硬盘的分区不同而产生的后遗症。笔者认为用户最好将引导扇区的内容保存在软盘上,必要时再将其复制回硬盘;也可将分区信息表的数值抄录保存,当需要从一台完好的微机复制主引导记录时,硬盘分区信息表的值也就可以正确填写了,否则用人工计算起始扇区、终止扇区的值是一件麻烦的事情。

现将读写主引导记录扇区和显示分区信息表的方法介绍如下:

1. 读取硬盘 0 头 0 道 1 扇区的内容,并以文件名 BOOT 存储在软盘上:

```
A>DEBUG↵
-A 1000↵
```

```
XXXX:1000 MOV DL,80 ;硬盘驱动器号
 MOV DH,0 ;0 磁头
 MOV CH,0 ;0 柱面
 MOV CL,1 ;1 扇区
 MOV BX,0100 ;内存首址
 MOV AL,1 ;扇区数
 MOV AH,2 ;读扇区
 INT 13 ;调用 13H 中断
```

```
-G=1000↵
-N A:BOOT↵ ;命名为 BOOT
-R CX↵
:0200↵ ;一个扇区的长度(字节)
-W 0100↵ ;从内存 0100H 处开始写
```

## 2. 显示硬盘分区信息表:

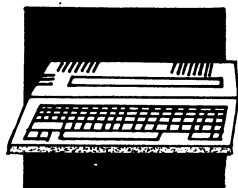
```
-D 01BE 01FF↵
其中:01BE~01CD :第一分区信息表
 01CE~01DD :第二分区信息表
 01DE~01ED :第三分区信息表
 01EE~01FD :第四分区信息表
 01FE~01FF :自举有效标志 55 AA
```

## 3. 将存储在软盘上名为 BOOT 的引导扇区内容写入硬盘 0 头 0 道 1 扇区:

```
A>DEBUG BOOT↵
-A 1000↵
XXXX:1000 MOV DL,80
 MOV DH,0
 MOV CH,0
 MOV CL,1
 MOV BX,0100
 MOV AL,1
 MOV AH,3
 INT 13
 INT 3
-G=1000↵
```

## 更正

本刊 91 年 11 期《将 APPLE DOS 移入 RAM 卡》一文,程序代码中 \$ 4499 单元的内容误排为 FD,正确值为 AD,特此更正。



## 学习机之友

# CEC-I 彩色 TV 字幕

南京中共江苏省委党校图书馆(210004) 包敏

微机制作字幕是令人向往的。笔者参考有关资料后编写了《CEC-I 彩色 TV 字幕》程序,它具有下列功能:

1. 字幕制作:可以对 CEC 汉字库内的所有汉字/字符进行彩色放大,字体可变形,背景可为彩色,放大后的字串可自定坐标显示。
2. 字幕存盘:制作好的字幕可以存盘。
3. 字幕打印:字幕可以打印成硬拷贝。
4. 单页字幕显示:已存盘的字幕单页再显示。
5. 连续字幕显示:已存盘的字幕多页连续显示。

程序为模块结构,扩展后可产生更多的字型。经试用,可满足一般的字幕效果,例如,在公用天线系统(CATV)内能起到一般字幕机的作用,只需一台 CEC 主机加一台软驱即可制作、播放字幕。

### 一、程序简要说明:

120-160 行:主菜单。

200-700 行:字幕制作模块,其中 205-340 行为用户输入参数,经自动处理后变为实际字幕参数(330 行)。在这段程序中,调用了 2000-2090、2100-2160、2200-2250 三个子程序,它们的功能分别为:放大后字串的行宽和字高、放大后字串行首 X/Y 坐标的预置、放大字颜色的限制。由于这些子程序的功能使繁琐的计算由机器自动完成,这样用户操作就变得十分简便,并且避免了屏幕色彩互相干涉,保证色彩清晰,355 行实现背景色,360-700 行是程序的核心部分。本程序放大原理是:按一定顺序扫描汉字/字符点阵,当遇亮点时即在相应位置放大。具体为,由 380 行指定高分辨第二页为工作页面(不显示),存放待放大的字串(390 行),经 520 行扫描判断为亮点时,转至 1000-1500 行子程序,按预置数据进行放大处理,此时的工作页面已由 1000 行的 POKE230,32 指向高分辨第一页,因 350 行已指定显示页面为第一页,故屏幕显示页面始终是放大后的页面(第一页)。“&”命令是经扩充的扫描功能,由 3110-3130 行预置。本程序可用线描方式放大(1200 行),也可用图形表方式放大(1100 行),图形表由 3160-3170 行预置为四个点的正方形,当 XZ=0 时以正方形为放大点,当 XZ=8 时以正方形旋转 45°后为放大点。1300 行是立体处理。掌握以上原理,只要在放大子程序(1000-1500 行)内进行扩充就可以创造出更多的字体。

5000-5040 行:打印字幕模块。

5100-5170 行:字幕存盘/单页字幕显示模块。

5200-5350 行:连续字幕显示模块。此模块采用了“黑幕技术”,即当屏幕显示某一页(5310 行)的同时,

读入另一页(5270 行第一页,5280 行第二页),当按一下 Esc 键(5300 行)后自动重复以上换页操作,这样以不断按动 Esc 键实现连续显示字幕。

### 二、操作说明

1. 字幕制作:按中文提示输入即可。(背景)指屏幕背景色。(输入字符串)即需放大的汉字/字符串(应小于等于 17 个汉字)。本行之前的提示为:本行输入的行数/本行屏幕 Y 坐标的起始点/屏幕 Y 轴还可多少点。(放大倍数)可参考以上提示设置,每个汉字/字符的基本点阵为 16×16/16×8,倍数与字串宽度及字形有关,当放大后串宽大于 279 时机器自动减 0.5 倍,直至等于小于 279。因此用户在输入放大倍数时无需精密计算,只要字串小于等于 17 个汉字即可。(立体)选择是否为立体字(只支持线描放大)。(正斜)选择是正体、还是斜体,斜体字向右倾斜 45°。(色彩)指放大后的字串颜色,按括号内提示选择。(放大描绘)指按线描方式(0)放大,还是按图形表方式放大,1 为正方形、2 为正方形旋转 45°。(X 坐标)指定本行在屏幕偏左(L)、居中(M)或偏右(R)位置。(Y 坐标)指定本行在屏幕首行(T)、中行(M)或末行(B)的位置,当为连续行时输入 M 即可,行行之间相距 7 个点。

当为了硬拷贝(打印)而制作时,建议背景色置黑、字色置白,这样打印后的字迹清晰。

输入字符串后直接按回车键,机器即按输入参数进入自动制作,完成制作后,按空格键取消本屏,重新进入制作;按 P 键进入打印;按 S 键进入存盘;按 L 键取消本屏,进入读盘;按回车键返回主菜单。

2. 字幕存盘:由“主菜单”或“制作”均可进入存盘,存盘时需按系统规定输入文件名,如果用户是为了连续显示而制作,请采用以下形式:

FS(F:为字母组成的文件名;S:为序号 1,2,3……n)

例:PIC1(第 1 屏字幕)

PIC2(第 2 屏字幕)

:

PICn(第 n 屏字幕)

3. 字幕打印:由制作进入,即直接打印;由主菜单进入,需输入文件名再进入打印,打印后按回车键返回各自出口处(鸣笛表示已返回“制作”模块)。打印之前莫忘开打印机。

4. 单页字幕显示:由主菜单进入,输入文件名即可。显示后按回车键返回主菜单。

5. 连续字幕显示:由主菜单进入,输入文件名(即存盘时的主文件名:F)、起始号(即存盘时的序号:1)、

终止号(即存盘时的序号;n),当驱动器红灯熄灭时按一下 Esc 键,即显示一屏字幕,连续这个操作直至显示第 n 屏字幕;此后,若重复一遍则按 R 键;返回主菜单则按回车键。

6. 当操作失误时,程序转入主菜单,如果运行中程序被挂起,可强制中断(Ctrl-Reset),此时进入监控,再键入 Ctrl-C 即返回 BASIC,再重新运行程序(RUN)。

```

100 ONERR GOTO 120
110 GOSUB 3110
120 PRINT CHR$(4);"PR#3";PRINT,HGR2
130 VTAB 1;PRINT TAB(9);"CEC-I 彩色 TV 字幕";
 VTAB 3;HTAB 11;PRINT "①字幕制作";HTAB 11;
 PRINT "②字幕存盘"
135 HTAB 11;PRINT"③字幕打印";HTAB 11;PRINT"④单
 页字幕显示";HTAB 11;PRINT"⑤连续字幕显示";
 HTAB 11;PRINT "⑥结束"
150 VTAB 9;HTAB 21;INPUT"选择:";XX$:XX=VAL
 (XX$);IF XX<1 OR XX>6 THEN CALL -1052;
 GOTO 150
152 IF XX=2 THEN HH=211
154 IF XX=3 THEN PR=1;HH=204
156 IF XX=4 THEN HH=204
160 ON XX GOTO 200,5100,5100,5100,5200,190
190 END
200 CLEAR;GOSUB 3200;HGR2;VTAB 1
205 INPUT"背景(0黑;1绿;2紫;3白;5橙;6蓝)?";BJ;
 GOSUB 2200
210 HOME ;A=A+1;PRINT"第";A;"行/";PRINT"本行
 Y坐标起始为";LJ;"/" ;PRINT"还可用";192-LJ
220 INPUT"输入字符串:";ZC$;IF LEN(ZC$)=0
 THEN 350
230 INPUT"放大倍数:";BS
240 INPUT"立体(是/1;非/0)?";LT
250 INPUT"正斜(正/0;斜/1)?";ZX
260 W$=ZC$;GOSUB 2000;I=W
265 PRINT"(背景为:";S$(BJ);"色)"
270 PRINT"色彩";BJ$;INPUT SC
275 INPUT"放大描绘"(线/0;图/1;图/2)";MH
276 IF MH=0 THEN 280
277 IF MH=1 THEN TX=1;XZ=0
278 IF MH=2 THEN TX=1;XZ=8
280 INPUT"X坐标(左/L;中/M;右/R)";HW$
290 INPUT"Y坐标(首/T;中/M;底/B)";VW$
295 IF VW$="T" THEN LJ=0
300 LJ=LJ+SG+7
305 IF VW$="B" THEN 320
310 IF LJ-7>191 THEN PRINT"本行超出屏底!";CALL
 -1052;A=A-1;LJ=LJ-SG-7;GOTO 340
320 GOSUB 2100
330 SG(A)=SG;W$(A)=W$;I(A)=I;LT(A)=LT;ZX
 (A)=ZX;BS(A)=BS;SC(A)=SC;X(A)=X;Y(A)=
 Y;MH(A)=MH;TX(A)=TX;XZ(A)=XZ;GOTO 210
340 INPUT"重新输入本行数据?(Y/N)";CX$;IF CX$=
 "Y" THEN 210
350 HGR;POKE-16302,0

```

```

355 HCOLOR=BJ;HPLOT 0,0;CALL 62454
360 FOR R=1 TO A
370 W$=W$(R);BS=BS(R);SC=SC(R);LT=LT(R);
 ZX=ZX(R);I=I(R);X=X(R);Y=Y(R);MH=MH
 (R);TX=TX(R);XZ=XZ(R)
380 POKE 230,64
390 HOME ;PRINT W$
400 & 1,1,A%;M=A%;N=M;IF M=(INT(N/2))*2
 THEN M=M+1
500 CC=0;FOR H=0 TO I;FOR V=0 TO 16
510 IF ZX THEN CC=16-V
520 & H,V,A%;IF M=A% THEN GOSUB 1000
540 NEXT V,H
700 NEXT R
710 HH=PEEK(-16384);POKE-16368,0;IF HH=160
 THEN 200
720 IF HH=208 THEN GOSUB 5000
730 IF HH=211 OR HH=204 THEN 5100
740 IF HH=141 THEN 120
790 GOTO 710
1000 POKE 230,32;POKE 49328,16;HCOLOR=SC
1100 IF MH=1 OR MH=2 THEN F=X+CC+H*BS;G=
 Y+V*BS;ROT=XZ;SCALE=BS;DRAW TX AT F,
 G
1200 IF MH=0 THEN F=X+CC+H*BS;G=Y+V*BS;
 FOR Z=G TO G+BS-1;HPLOT F,Z TO F+BS-1,
 Z;NEXT
1300 IF LT THEN HCOLOR=6;HPLOT F+BS,Z TO F+BS
 +3,Z+4;HCOLOR=SC
1500 POKE 230,64;RETURN
2000 W=LEN(ZC$);HZ=0;ZF=0
2010 FOR I=1 TO W
2020 IF ASC(MID$(ZC$,I,1))=127 THEN HZ=HZ+1
2030 NEXT I
2040 ZF=W-HZ*3
2050 W=HZ*16+ZF*8
2060 SK=W*BS+(LT*3)+(ZX*16);SG=16*BS+
 (LT*4)
2070 IF SK>279 THEN BS=BS 0.5;GOTO 2060
2080 IF BS<1 THEN PRINT"本行太宽,请重新输入!";
 CALL-1052;FOR T=0 TO 2000;NEXT ;A=A-1;
 POP;GOTO 210
2090 RETURN
2100 IF HW$="L" THEN X=0
2110 IF HW$="R" THEN X=279-SK
2120 IF HW$="M" THEN X=INT((279-SK)/2)
2130 IF VW$="T" THEN Y=0
2135 IF VW$="T" AND MH<>0 THEN Y=1*BS
2140 IF VW$="B" THEN Y=191-SG
2150 IF VW$="M" THEN Y=LJ-SG
2160 RETURN
2200 IF BJ=0 OR BJ=3 THEN BJ$="(0"+S$(0)+"
 ;1"+S$(1)+" ;2"+S$(2)+" ;3"+S$(3)+" ;5"
 +S$(5)+" ;6"+S$(6)+")"
2210 IF BJ=1 THEN BJ$="(0"+S$(0)+" ;1"+S

```

```

$ (1)+“;3”+S$ (3)+“”
2220 IF BJ = 2 THEN BJ $ =“(0”+S$ (0)+“;2”+S
$ (2)+“;3”+S$ (3)+“”
2230 IF BJ = 5 THEN BJ $ =“(4”+S$ (4)+“;5”+S
$ (5)+“;7”+S$ (7)+“”
2240 IF BJ = 6 THEN BJ $ =“(4”+S$ (4)+“;6”+S
$ (6)+“;7”+S$ (7)+“”
2250 RETURN
3110 POKE 1013,76;POKE 1014,0;POKE 1015,3
3130 FOR I=768 TO 817;READ J;POKE I,J;NEXT
3140 DATA 32,185,246,32,17,244,164,229,177,38,41,
127,133,255,165,48,41,127,37,255,9,128,197,
48,208,4,169,1,208,2,169,0,72,32,190,222,32,
227,223,160,0,145,131,200,104,145,131,96,0,0
3150 DATA 1,0,4,0,44,62,0
3160 DZ=818;FOR I=DZ TO DZ+7-1;READ J;POKE I,
J;NEXT
3170 POKE 232,50;POKE 233,3
3180 RETURN
3200 S$ (0) =“黑”;S$ (1) =“绿”;S$ (2) =“紫”;S$ (3)
=“白”;S$ (4) =S$ (0);S$ (5) =“橙”;S$ (6) =
“蓝”;S$ (7) =S$ (3)
3210 RETURN
5000 POKE 49236,0;PRINT CHR $ (4);“PR # 1”;POKE
1913,1;PRINT CHR $ (17);PRINT CHR $ (4);“PR
0”
5010 T = PEEK (-16384);POKE -16368,0;IF T = 141
THEN 5030
5020 GOTO 5010
5030 IF HH=208 THEN CALL -1052;RETURN
5040 GOTO 120
5100 PRINT CHR $ (4);“PR # 3”;PRINT;HOME

```

```

5110 INPUT“请输入文件名:”;F$
5120 P1= PEEK(49236);POKE 49328,16
5130 IF HH=211 THEN PRINT CHR $ (4);“BSAVE”;F$;
“,A $ 2000,L $ 2000”;GOTO 5170
5140 IF HH= 204 THEN POKE -3086,0
5150 PRINT CHR $ (4);“BLOAD”;F$
5155 IF PR=1 THEN 5000
5160 T= PEEK(-16384);POKE -16368,0; IF T=141
THEN 120
5170 GOTO 5160
5200 PRINT CHR $ (4);“NOMON,C,I,O”;PRINT CHR
$ (4);“PR # 3”;PRINT;HOME
5210 INPUT“输入文件名:”;FF$
5220 INPUT“画面起始号:”;S
5230 INPUT“画面终止号:”;E
5240 POKE 49328,16
5250 FOR I=S TO E
5260 F$ = FF$ +STR$ (I)
5270 IF 1/2<> INT (1/2) THEN PK = 49236;POKE 230,
32;PRINT CHR $ (4);“BLOAD”;F$;“,A8192”
5280 IF 1/2=INT (1/2) THEN PK = 49237;POKE 230,64;
PRINT CHR $ (4);“BLOAD”;F$;“,A16384”
5300 T = PEEK (-16384);POKE -16368,0;IF T<>155
THEN 5300
5310 POKE PK,0
5320 NEXT I
5330 T = PEEK (-16384);POKE -16368,0;IF T = 210
THEN 5250
5340 IF T=141 THEN 120
5350 GOTO 5330
6000 REM By:BAO GAN 1990.7.10

```

## BASIC 程序中变量名使用情况的列表程序

成都四川大学材料系(61006) 蔡伟

BASIC 程序中使用了大量的变量名,如果能对一个程序中的全部变量名(简单变量,数组变量,字符串变量)按字母顺序列表出来,每个变量名被使用的程序行号也列印出来,对于分析程序或修改程序都将极其有用。

下面介绍的 VLIST 程序能迅速完成这一项工作。程序是由 6502 机器语言编成,适用于苹果机、中华学习机及其兼容机等。程序置于内存 \$7D00~\$7E64(相当于十进制地址 32000)。其中 \$7D00~\$7D4E 是 VLIST 的主程序段。\$7D50~\$7DE3 是根据变量名的命名规则在 BASIC 程序中去识别各种类型的变量名。\$7DE4~\$7E31 是将搜寻到的变量名排序。\$7E32~\$7E57 是针对每个变量名查出所出现的行号并列

印出。

VLIST 程序搜查变量名及行号非常迅速,不到一秒钟就能完成搜查列印工作。每印出一个变量名,在冒号后面印出它所在的全部行号。变量名是按照首字母的英文字母顺序排列的,首字母相同的变量名族内,按其第一次在程序中出现的先后来排序,因而便于清查。使用办法:

VLIST 程序见附录,假定你磁盘上已有 VLIST 程序,使用如下:

```
LOAD ××××(×××指要列印变量名的 BASIC 程序文件名)
```

```
BLOAD VLIST
```

```
CALL 32000
```

屏幕上将连续列印变量名及行号,可用 Control-S 键暂停。

开启打印机,执行 PR #1 命令后,再执行 CALL 32000,可在打印纸上获得列表输出。

```
7D00— A9 41 85 5C 8D 00 02 A9
7D08— 00 85 0C 20 58 7E A8 91
7D10— 0A 20 50 7D C6 0C 20 58
7D18— 7E A4 5D A2 00 B1 0A F0
7D20— 23 20 8E FD B1 0A 9D 00
7D28— 02 C9 0D F0 09 09 80 20
7D30— ED FD E8 C8 D0 EE 84 5D
7D38— A9 BA 20 ED FD 20 50 7D
7D40— E6 5D D0 D5 E6 5C A5 5C
7D48— C9 5B 90 B8 4C 3C D4 00
7D50— A6 67 A5 68 86 B8 85 B9
7D58— A0 00 B1 B8 D0 06 C8 B1
7D60— B8 D0 01 60 A0 04 B1 B8
7D68— D0 0A A0 00 B1 B8 AA C8
7D70— B1 B8 D0 E0 10 0F C9 83
7D78— F0 04 C9 B2 D0 14 C8 B1
7D80— B8 D0 FB F0 E5 C9 22 D0
7D88— 0C C8 B1 B8 F0 DC C9 22
7D90— D0 F7 C8 D0 D1 C9 41 90
7D98— F9 C9 5B B0 F5 D0 00 02
7DA0— F0 06 A9 00 85 5A F0 02
```

```
7DA8— 84 5A C8 B1 B8 C9 30 90
7DB0— 0C C9 3A 90 F5 C9 41 90
7DB8— 18 C9 5B 90 ED C9 21 F0
7DC0— 08 C9 24 F0 04 C9 25 D0
7DC8— 03 C8 B1 B8 C9 28 D0 01
7DD0— C8 84 5B A2 00 A4 5A F0
7DD8— 31 B1 B8 9D 80 02 E8 C8
7DE0— C4 5B 90 F5 A5 0C D0 4A
7DE8— A9 00 9D 80 02 20 58 7E
7DF0— A4 5D A2 00 B1 0A F0 20
7DF8— DD 80 02 D0 04 E8 C8 D0
7E00— F3 C9 0D D0 09 BD 80 02
7E08— D0 0B A4 5B D0 85 C8 B1
7E10— 0A C9 0D D0 F9 C8 D0 DA
7E18— A2 00 BD 80 02 F0 06 91
7E20— 0A E8 C8 D0 F5 A9 0D 91
7E28— 0A A9 00 C8 91 0A A4 5B
7E30— D0 DA A9 0D 9D 80 02 A2
7E38— 00 BD 80 02 DD 00 02 D0
7E40— 13 E8 C9 0D D0 F3 A0 02
7E48— B1 B8 AA C8 B1 B8 20 24
7E50— ED 20 48 F9 A4 5B D0 B4
7E58— A5 B0 85 0B A9 00 85 5D
7E60— 85 0A E6 0B 60
```

## 磁头清洗及其辅助程序

广西师范大学(541001) 唐汉雄

Apple 机的驱动器使用时间长了,或使用了质量低劣的磁盘,磁头就会附着一些磁粉。影响信息的正确存取,严重时会出现读写错误,更甚时会划伤磁盘。因此磁头清洗是计算机日常维护不可缺少的一项内容。本文将介绍使用清洗盘进行磁头清洗的一般方法并给出一个磁头清洗辅助程序。

清洗盘是用来清洗驱动器磁头的专用工具。其使用的一般方法是:先将清洗剂均匀涂在清洗盘的窗口上,然后将清洗盘插入第一驱动器并关上仓门,再打开计算机电源,待驱动器转动约 30 秒钟后,按复位键即可完成第一驱动器的清洗工作。

对第二驱动器的清洗最简便的方法是:在 BASIC 状态下键入 POKE49387,0,POKE49385,0 命令即可启动第二驱动器,当到达预定清洗时间后再按复位键即可。

上述方法最大的优点是不需要载入 DOS 或其它工具软件就可进行清洗工作,尤其是第一驱动器不能正常读入 DOS 时特别有用。当计算机能正常载入 DOS 或工具软件时,也可使用 DOS 命令或工具软件的调试操作进行清洗。

然而这些方法也有其不足,就是在清洗时磁头多

半停在 \$00 或 \$11 轨道上,磁头不会沿盘的直径方向上来回移动,磁粉也只能被吸附在清洗盘特定的轨道上,无疑会缩短清洗盘的使用寿命。同时清洗时间也需人为控制,启动命令需逐个输入,常感不便。要是计算机开机后能自动执行能使驱动器转动,其磁头又能在径向快速来回移动,同时还能自动控制清洗时间的程序,那么操作将会更加简便,清洗效果也将会更好。为此笔者编写了一个能实现上述功能的磁头清洗辅助程序。现将该程序的执行过程介绍如下:

首先自动寻找驱动卡所在的槽号,若找不到会给出“NO DRIVE CARD”(无驱动卡)的提示。若找到驱动卡则显示“DRIVE=1”,这时插入清洗盘再直接回车或按“1”键即可启动第一驱动器。清洗盘转动的同时磁头也沿盘的径向快速来回移动,约 20 秒钟后自动停止,随后又显示“DRIVE=2”,此时可改插清洗盘再回车或按“2”键就可清洗第二驱动器。

每次清洗完毕后驱动器号码可自动转换并给出提示,既可直接回车选取默认值,也可敲入 1 或 2 另行选择,还可按“Esc”键随时中断或退出清洗状态。

本程序最大的特点是自动化程度高,能自动寻卡,

驱动器号码自动转换,自动定时,且操作特别简单,插好清洗盘后只需按两次回车就可完成两台驱动器的清洗工作。因此本程序特别适用于大批量驱动器磁头的定期清洗。

本程序设定的清洗时间约为 20 秒,如需改变这一时间可增减程序 \$ 9079 单元的值。虽然本程序在有无 DOS 时运行效果一样,但若将此程序作为 HELLO 程序或由 HELLO 程序直接调用并制成专用磁盘,启动磁盘后可自动执行本程序,会使清洗更为简便快速。顺便说明,因本程序可控制磁头快速来回运动,若手头没有专用的清洗盘,用一张干净的磁盘代替清洗盘也同样有清洗磁头的效果。

附磁头清洗辅助程序如下:

```

9000— 20 58 FC 84 06 A9 C8 85
9008— 07 A0 07 C6 07 A5 07 C9
9010— C0 F0 3F B1 06 D9 01 FB
9018— D0 EF 88 88 10 F5 A5 07
9020— 0A 0A 0A 0A 85 2B A9 10
9028— 85 24 20 5B FB A9 EA A0
9030— 90 20 3A DB C6 24 20 0C
9038— FD C9 8D F0 11 C9 9B F0
9040— 1B C9 B1 F0 1A C9 B2 F0
9048— 21 20 3A FF F0 D8 A9 B1
9050— D0 EF A9 F2 A0 90 20 3A
9058— DB 20 3A FF 4C D0 03 20
9060— E1 90 BD 8A C0 20 75 90
9068— D0 BC 20 E1 90 BD 8B C0
9070— 20 75 90 D0 B1 BD 89 C0
9078— A0 06 98 48 20 A8 90 A9
9080— 80 20 A8 FC 20 C2 90 68
9088— A8 AD 00 C0 10 05 8D 10
9090— C0 D0 03 88 D0 E4 A6 2B
9098— BD 88 C0 38 A9 63 ED 4F
90A0— 90 8D 4F 90 8D F0 90 60
90A8— A6 2B A0 50 BD 80 C0 98
90B0— 29 03 0A 05 2B AA BD 81
90B8— C0 A9 56 20 A8 FC 88 10
90C0— EB 60 A6 2B BD 80 C0 A0
90C8— 01 98 29 07 05 2B AA BD
90D0— 80 C0 A9 56 20 A8 FC BD
90D8— 7F C0 C8 C8 C0 A3 D0 E9
90E0— 60 8D 4F 90 20 ED FD A6
90E8— 2B 60 C4 D2 C9 D6 C5 BD
90F0— B1 00 CE CF A0 C4 D2 C9
90F8— D6 C5 A0 C3 C1 D2 C4 00

```

## 求法雷数列的新方法

湖北省煤炭工业学校(610064) 廖庆平

在数学中,一个有名的数列叫法雷数列。它出自著名数学家法雷之手,它的构成为:对于任何自然数 N,

求出所有分母为小于或等于 N 的真分数,把这些真分数按顺序排列起来,并且在第一个分数的前面加上分数 0/1,在最后面加上分数 1/1,于是我们把得到的这一组数,称之为 N 级法雷数列。

例:  $F_5 = \{0/1, 1/5, 1/4, 2/5, 1/2, 3/5, 2/3, 3/4, 4/5, 1/1\}$

$F_8 = \{0/1, 1/8, 1/7, 1/6, 1/5, 1/4, 2/7, 1/3, 3/8, 2/5, 3/7, 1/2, 4/7, 3/5, 5/8, 2/3, 5/7, 3/4, 4/5, 5/6, 6/7, 7/8, 1/1\}$

它在数学中有着许多重要的应用,现在我们只来研究它的构成规律。从上面可以看出,我们不能写出它们的通项,通过观察,发现它们之间仍服从一个简单的规律。设  $FA(I)$  为数列中的分子,  $FB(I)$  为分母,则有:

$$FA(I)/FB(I) = (FA(I-1) + FA(I+1)) / (FB(I-1) + FB(I+1))$$

考虑到数列中的每一项是通过约分后的真分数,把左边分子、分母乘以一个整数 K,则:

$$FA(I+1) = K * FA(I) - FA(I-1)$$

$$FB(I+1) = K * FB(I) - FB(I-1)$$

$$K = 0, 1, 2, 3, 4, \dots$$

由于对于任何前两项均为 0/1 和 1/N, 这样我们可以按上规律推出后面诸项。

现给出计算法雷数列的 BASIC 程序如下:

```

10 INPUT N; DIM FA(4 * N), FB(4 * N), A(4 * N, N), B(4 * N, N)
15 PR# 1; PRINT
20 FA(1)=0; FB(1)=1; FA(2)=1; FB(2)=N; I=3
30 FOR K=1 TO N
40 A(I,K)=K * FA(I-1) - FA(I-2)
50 B(I,K)=K * FB(I-1) - FB(I-2)
60 IF A(I,K) <= 0 OR B(I,K) <= 0 THEN 100
70 AA=A(I,K); BB=B(I,K); GOSUB 200
80 A(I,K)=AA/BB; B(I,K)=B(I,K)/BB
90 IF B(I,K) > N THEN 110
100 NEXT K
110 FA(I)=A(I,K-1); FB(I)=B(I,K-1)
120 IF FA(I)/FB(I)=1 THEN GOTO 140
130 I=I+1; GOTO 30
140 FOR K=1 TO I; PRINT FA(K); "/"; FB(K); " ";
145 PP=PP+1; IF PP=13 THEN PP=0; PRINT
150 NEXT K; END
200 R=AA-BB * INT(AA/BB)
210 IF R=1 THEN BB=1; RETURN
220 IF R=0 THEN RETURN
230 AA=BB; BB=R; GOTO 200
]RUN
? 5
0/1 1/5 1/4 1/3 2/5 1/2 3/5 2/3 3/4 4/5
1/1
]RUN
78
0/1 1/8 1/7 1/6 1/5 1/4 2/7 1/3 3/8 2/5
3/7 * 1/2 4/7 3/5 5/8 2/3 5/7 3/4 4/5 5/6
6/7 7/8 1/1

```



# TOOL-KIT 使用详解(续)

北大附中 姜宏

APA 就介绍到这里。关于如何扩展自己的 APA, 以及如何利用 & 命令扩充系统功能, 请参阅《电子与电脑》前几年的有关文章。顺便提一句, APA 中除了 &MANUAL 外, 其他命令都可用一个“&”字符和命令头一个字母代替, 如 &RENUMBER 用 &R; &XREF 用 &X, 但 &MANUAL 命令必须用 &MA 的简写方式。

### 3. 高分辨显示字符程序 HRCG

HRCG 是一个 R 型文件, 通过 LOADHRCG 程序运行。

```
开机键入 RUN LOADHRCG 后, 屏幕显示:
HOW MANY ALTERNATE CHARACTER SETS
WOULD YOU LIKE(0~9)?
```

这时, 你应键入你想好的字符集文件名(后缀为“.SET”的), 如果不准备用扩充字符集, 则请你键入 0, 回车。如果你输入了一个大于 0 或小于 9 的数, 那么屏幕还会出现:

```
NAME OF CHARACTER SET1?
```

这时, 我们比如想输入希腊文(Greek)字母, 就键入 GREEK.SET。按顺序把所需的字符集依次调入内存后, 屏幕被转到高分辨率第一页, 显示:

```
HI-RES CHAR GEN VERSION 1.0
```

下面, 我们依次来说明 HRCG 系统的各个命令, 说明中用 ^ A 表示同时按下 CTRL 和 A 键, 余者亦然。

#### (1) 字符集之间的切换

在启动系统时, 我们调入了几套字符集, 但屏幕上只能显示一套, 这就存在一个字符集之间切换的问题了, 怎么办呢?

^ An 命令可以帮我们解决这个问题, 比如我们调入了三套字符集, ^ A1 就可以把屏幕上字符显示改为第一套字符集的显示, 如果想返回标准字体, ^ A0 即可。

您如果想在程序中切换字型, 用 PRINT CHR \$(1);n, 表示选择第几套安符集。

#### (2) 正相与反相显示的切换

为了使一些字符串引起用户的注意, 我们可以使用反相显示的功能, 键入:

```
^ I 进入反相模式
```

```
^ N 返回正相模式
```

^ I 命令只不过在显示每一个字符时, 将它的每一个代码作逻辑运算 EOR # \$FF。

如果在程序中切换正反相显示字型, 用 PRINT CHR \$(9)代表 ^ I, PRINT CHR \$(14)来代替 ^ N。

#### (3) 大写与小写的切换

大写、小写的切换有以下命令:

```
^ K 进入大写模式
```

```
^ L 进入小写模式
```

```
^ S 以后键入的第一个字母大写, 余者小写。
```

```
^ Z 切换至最初状态, 也就是 0 号字型, 大写模式, 正相显示, 全屏幕显示。
```

在程序中, 用 CHR \$(19)代替 ^ S, CHR \$(11)代替 ^ K, CHR \$(12)代替 ^ L, CHR \$(26)代替 ^ Z。下列是一个综合运用几条命令的例子:

```
10 PRINT CHR $(1);2
20 PRINT CHR $(14)
30 PRINT CHR $(19)“I LOVE COMPUTER.”; PRINT
 CHR $(1);0
40 PRINT CHR $(11)“MY MOTHERLAND IS CHINA.”
50 PRINT CHR $(26);END
```

#### 4. 屏幕的清除

有关屏幕清除的命令也有几条, 说明如下:

```
^ P 清除窗口内的内容, 并将光标送回(1,1)位置。
```

```
^ E 清除光标以右的所有字符。
```

```
^ Q 将光标送回(1,1)位置, 不清屏。
```

```
^ F 清除光标以后(包括该行右边和以下各行)的所有字符。
```

下面列一张表, 是 HRCG 和文本显示时的命令对照表:

| HRCG 的命令 | 文本型的指令                   | 用途       |
|----------|--------------------------|----------|
| ^ P      | HOME 或 ESC-@<br>CALL-936 | 清窗口字符    |
| ^ E      | ESC-E 或 CALL-868         | 清余行      |
| ^ F      | ESC-F 或 CALL-958         | 清余幅      |
| ^ Z      | TEXT                     | 一切恢复正常显示 |

#### 5. 窗口的设计

我们在显示文字时, 可以要求它只在一定范围内显示, 如同 TEXT 的 \$20-\$23 单元的作用一样, 所涉及到的命令及说明如下:

```
^ Y 将窗口设为全屏幕。
```

```
^ V 光标目前位置的右下方为窗口(定义窗口左上角)
```

```
^ W 光标目前位置的左上方为窗口(定义窗口右下角)
```

#### 6. 文字方阵的显示

如果我们想在一个特定的位置印一个 n×m 的文字方阵, 用普通方法相当繁, 如:

```
10 VTAB 10;HTAB 10;PRINT“ABC”
```

```
20 HTAB 10;PRINT“DEF”
```

```
30 HTAB 10;PRINT"GHI"
```

如果我们想只用一个 PRINT 解决文字方阵的问题,就要用文字方阵显示命令:

- ^ B 进入方阵显示模式
- ^ D 退出方阵显示模式
- ^ C 显示中的换行

还以上面的程序为例,用文字方阵显示命令的程序就简单得多了:

```
10 VTAB 10;HTAB(10);PRINT CHR $(2);"ABC";CHR $(3);"DEF";CHR $(3);"GHI";CHR $(4);CHR $(13)
```

两个程序的运行结果,都是在屏幕(10,10)处显示:

```
ABC
DEF
GHI
```

### 7. 滚动显示及循环显示的切换:

屏幕上关于这部分的命令如下:

- ^ O+^ S 卷动方式
- ^ O+^ W 循环方式

HRCG 初次启动时,屏幕显示方式为卷动方式。

### 8. 显示页的选择

苹果机共有两个高分辨图形页,可用下面的命令来切换 HRCG 将字符印到哪一页上:

- ^ O+^ A 操作第一页,显示页不变。
- ^ O+^ B 操作第二页,显示页不变。
- ^ O+^ D 显示操作页。

### 9. 显示模式的选择

HRCG 可进行 NOT,OR,EOR,覆盖重写(正常)四种方式的显示:

- ^ O+^ P 覆盖重写
- ^ O+^ C EOR 方式
- ^ O+^ O OR 方式

至于 NOT 方式,就是常说的反显方式,用 ^ I 实现。

比如,在 China 底下划一横线,变为 China,程序如下:

```
10 A $="China";PRINT CHR $(16)
20 B $="_____"
30 PRINT CHR $(15);CHR $(15)
40 HTAB 1;VTAB 15;PRINT A $
50 HTAB 1;VTAB 15;PRINT B $
] RUN
```

China

### 10. 两页显示运算方式的选择

命令如下:

- ^ O+^ R 上下两页 EOR 在上页
- ^ O+^ T 上下两页 OR 在上页
- ^ O+^ P 正常印法

### 11. 使用自己的机器语言子程序

如果你在 HRCG 状态下想使用自己写的机器语言程序时,有如下两条命令可用:

- ^ O+^ Y 调用 A 子程序
- ^ O+^ E 调用 B 子程序

A 子程序 首地址的低 8 位 R+10

A 子程序 首地址的高 8 位 R+11

B 子程序 首地址的低 8 位 R+13

B 子程序 首地址的高 8 位 R+14

R 的位置在 48K 或 64K 机器中,通常为 \$ 8DFF (十进制的 36351)。

### 12. 一切恢复至起始状态

用 ^ Z 可完成,它做的工作有:

- ①将字符集选择在 0 号,即 CHR \$(1);0
- ②进入大写模式 CHR \$(11)
- ③进入正相模式 CHR \$(14)
- ④全屏幕显示 CHR \$(25)
- ⑤操作页和显示页都为第一页 CHR \$(15);CHR \$(1);CHR \$(15);CHR \$(14)
- ⑥正常印字方法 CHR \$(15);CHR \$(16)
- ⑦进入卷动方式 CHR \$(15);CHR \$(19)
- ⑧A 子程序,B 子程序均为 RTS
- ⑨取消方阵显示 CHR \$(4)

### 13. 如何在自己的程序中调入 HRCG

HRCG 既然有如此多的好处,那么,你可以用下面的子程序在自己的程序中调入它:

```
行号 ADRS=0
行号 PRINT CHR $(4);"BLOAD RBOOT";CALL 520;
 ADRS=USR(0),"HRCG"
行号 CALL ADRS
行号 RETURN
```

至此,HRCG 亦已介绍完。大家如能利用它做一些 CAI 的软件,那将是很不错的。

### 4. 高分辨字符集制造及编辑程序——ANIMATRIX 的使用与改进

Animatrix 是扩充 HRCG 工具的一个有效的工具程序,这个程序相当长,占用了 26 个扇区,其运行环境内存要求为 48K 或 64K,一台软驱。

在 "]" 提示符下键入 RUN ANIMATRIX 并回车后,屏幕提问:

Character set to edit?

要求你键入要编辑的字符集名。

这时,我们可键入一个字符集文件,比如说希腊字符集 GREEK.SET,键入后按回车键,现在,屏幕显示一个由 7×5 个 7×8 的小方阵组成的大方阵,右边有两个方框,上面一个是用于用户输入标准字型,下面一个是供编辑的字型。下面有一行提示:

Press control-I to display Instructions

意思是:按 CTRL-I 可以阅读提示。接上例,我们输入了 GREEK.SET,现在,我们按下小写的 A、B、C 三个字母(关于如何键入小写字符的问题已在第三部分 HRCG 的说明中谈过了)这时,上面的框显示图 1,下面的框显示图 2。

ABC

ABΓ

图 1

图 2

当我们编辑 "["、“\”、“-”三个字符时,用 CTRL-K 代替 "[" ,用 CTRL-L 代替 "\",用 CTRL-O 代替 "-"。这

在改进后的 APPLE 及 CEC-I 上,用键盘上已有的键就可输入它们。

①光标的移动

有几个功能键:

→将光标右移一格

←将光标左移一格

RETURN 换行

上移用←把光标移至行首后再按←键即可。

②字型的创作及修改

这部分可用游戏杆。

向上方推:光点向下移。

向下方推:光点向上移。

向左方推:光点向右移。

向右方推:光点向左移。

按钮 0:消除亮点。

按钮 1:加入亮点。

键盘上的功能键有:

CTRL-T 该行所有的点均移半点,这是水平方向 560 点分辨技巧的应用。

CTRL-A 大写。

CTRL-D 显示所有字型。

CTRL-I 显示提示

CTRL-S 字型复制:把光标所在的字符复制成键盘键入的字符。

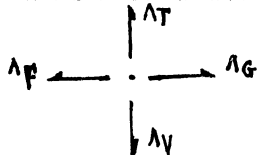
③字型的存储

编辑好 ASCII 字符集后,只要键入 ESC, 屏幕提问:

Name to save? (问你文件名取什么。)

当您输入文件名后,此字符集便被存盘。如果您误按了 ESC 键,则出现“Name to save?”后按回车再按 N, 回车即可回到编辑状态。

由于游戏杆操作十分不便,为此,可将 ANIMATRIX 作了修改,使它能用键盘操作,功能键如下:



擦光点: ^ W  
写光点: ^ E

由于 ^ T 复盖了原 ^ T 的功能,因此,改过的 ANIMATRIX 用 ^ Y 代替原 ^ T。

仅将改过的部分列在下面:

```

]LIST 400
400 IF C=25 THEN GOSUB 630
]LIST 490,560
490 REM MOVE PX & PY
495 IF C=6 THEN PX=PX-1;PL=-1;IF PX<0
 THEN PX=48
497 IF C=7 THEN PX=PX+1;PL=-1;IF PX>48
 THEN PX=0
500 IF C=20 THEN PY=PY-1;PL=-1;IF PY<0

```

```

THEN PY=39
503 IF C=22 THEN PY=PY+1;PL=-1 IF PY>39
 THEN PY=0
505 P0=PX*4+2*INT(PX/7)+2;P1=PY*4+INT
 (PY/8)+1
508 XDRAW 2 AT P0,P1;FOR I=1 TO 30;NEXT;
 XDRAW 2 AT P0,P1
510 IF C=5 AND PL<>0 THEN PL=0;GOSUB 570
520 IF C=23 AND PL<>1 THEN PL=1;GOSUB 570
560 GOTO 290

```

键入程序后,再输入下面的命令:

```
]DEL 1394,1398
```

```
]SAVE ANIMATRIX VERSION KEYBOARD 即可。
```

改进的程序无论在速度上还是在使用上,都比原程序好得多。

TOOL-KIT 到这里已经全部介绍完了。希望您看了以后对您的工作有一些帮助。在国内,已经有了许多仿 TOOL-KIT 的工具软件,电子工业出版社出的《苹果机、中华学习机 CEC-I 工具箱》就是其中的一个很好的示范。笔者亦已完成了对 TOOL-KIT 的修改及扩充,试用后觉得效果不错。关于 TOOL-KIT 的扩充是很容易的,不少国内书刊都介绍了给 APPLE 加功能的程序,把它们依次输入,存盘后即是一个“软件”。

## 传播科技知识 推动科技进步

### 九月初举办“广州科技图书交易会”

由中国版协科技出版工作委员会、广州市科委、市科协、广州市新华书店主办的“广州科技图书交易会”将于今年九月三日至十三日在中国出口商品交易会九号馆举行。

本次科技图书交易会是全国历次科技图书展销规模最大、图书知识门类最齐、品种最丰富的一次,参展的科学技术图书有来自中央部属的、高等院校的和各省(区)市的近百家科技出版社,参展还有来自香港的万里机构出版有限公司,荟萃了国内版和港台版科技图书近二万个品种。

为了方便单位选购图书,在交易会同时展出几千种文史哲类图书供参加单位选购。

在展销期间,还将由广州市科委、市科协组织举办有关科学技术专题讲座和咨询活动,并设立为外地单位读者代办托运等服务项目和进行书店与出版社之间的订货业务。

联系地址:广州市北京路 336 号广州科技书店  
邮政编码:510030 电话号码:3345723

## 第七章 循环程序设计

南京大学大气科学系(210008) 朱国江

## § 3 循环程序设计

在科学技术和生活领域里,常常遇到一些需要大量重复计算的课题,人们所关心的是如何减少一些不必要的重复性工作,以节省存储单元和提高计算效率。

完成大量的在处理形式上完全相同的重复性工作,这就是循环。在程序中,对某一段内容反复执行多次,这就是循环程序。而在一个循环程序中,包含另一个或几个循环,则称为循环的嵌套或多重循环。

循环程序的结构包括以下几个部份:

- 初始化部份,它建立计数器,地址寄存器(指针)和有关变量的起始值。如累加器清零、源数据块首地址、数据块长度、内外循环计数器、循环次数等量初值。

- 处理部份,在这部份进行实际的数据处理,这是循环的工作部份,它完成重复执行程序段的内容。

- 循环控制部份,它检查每循环一次是否结束,并为下一轮循环做好修正计数器和指针的准备。

- 结束部份,它分析、存放或显示结果,结束。

循环程序对于处理数据块的传送问题比较方便,而采用变址寻址方式处理动态数据块和多个数据块则较为理想,因为变址寻址方式允许用改变变址寄存器内容的方法来改变存储器的实际地址。为此,程序必须在每一轮处理后将变址寄存器内容加 1,使它指向数据块的下一个元素,这样就使下一轮对下一个存储单元的数据执行相同的操作,因而可用同一个程序段来处理 256 个字节长度的数据块(因变址寄存器是 8 位字长)。

另外,我们将从下面所举的实例中看到,在循环程序中大量的包含了程序分支内容,因为每循环执行一次后是继续重新循环呢?还是不再循环而往下执行新的程序段呢?这里就存在着控制循环的程序分支,以根据某种条件满足与否决定程序的走向,控制语句的执行路径。因此,我们可以看到分支和循环的概念是不同的,对于一些单纯的分支程序,其中每条指令可能只执行一次而不进行循环,而循环程序的处理部分则在程序执行过程中将被执行许多次。同时,我们也看到,单纯分支不包括循环,循环却一定包含分支,它们既有区别又有联系,而 6502 指令系统中的比较指令和条件转移指令,为设计分支程序和循环程序提供了方便。

[例 1]求数据的累加和

计算一串二进制的和,这串数的长度放在 \$06 单元,而这串数从 \$6001 单元开始存放。将和数放在 \$07 单元,设和数是一个 8 二进制位数,这样可以不考虑进位。见程序 7.1。

程序 7.1:

```

1000- 20 58 FC JSR $FC58 ;清屏
1003- A9 00 LDA # $00 ;和置初值
1005- AA TAX ;计数器置初值
1006- D8 CLD ;二进制求和,清十
 ;进制标志位
1007- 18 CLC ;清进位位 C
1008- 7D 01 60 ADC $6001,X ;和=和+数据
100B- E8 INX ;计数器加 1
100C- E4 06 CPX $06 ;次数够了吗?
100E- D0 F7 BNE $1007 ;不够,继续循环
1010- 85 07 STA $07 ;够了,送结果
1012- A5 07 LDA $07 ;取结果→A
1014- 20 DA FD JSR $FDDA ;显示结果
1017- 60 RTS ;结束

```

程序 7.1 设计要点:

- 初始化:和数放累加器 A 中,初始值为 0,寄存器 X 当计数器,未加前为 0,清 10 进制标志和清进位位标志。

- 循环:取数相加,变址计数器加 1 指向下一个地址的数据,判断数据是否加完,未完继续循环求和,是则送结果并显示

- 结束:从 \$07 单元取结果,调显示字符(16 进制)子程序,送现行输出设备上,结束。

运行前: \* 6.05 设串长度为 5

\* 6001:12 34 1D 56 20      5 个数据

运行后: \* 1000G✓

\* D9

\* 7✓

\* 0007-D9

关于程序 7.1 的几点说明:

- 本程序原则上可以完成 N 个数的重复相加,数据个数放 \$06 单元,并可灵活更改。

- N 个数的累加和不能超过 \$FF(255),否则结果不正确。

- 是一个单循环,数据的累加是从前向后进行的。

- 循环中包括分支,主要由执行比较指令 CPX \$06 后,P 寄存器中的零标志 Z 值决定 BNE 指令是否转移。而循环的次数由指令 INX,CPX,BNE 共同决定。

对 N 个数据的累加和,也可以如下编制,见程序 7.2。

程序 7.2:

```

1000- 20 58 FC JSR $FC58
1003- A9 00 LDA # $00
1005- AE 00 60 LDX $6000

```

```

1008- D8 CLD
1009- 18 CLC
100A- 7D 00 60 ADC $ 6000,X
100D- CA DEX
100E- D0 FA BNE $ 100A
1010- 8D 30 10 STA $ 1030
1013- AD 30 10 LDA $ 1030
1016- 20 DA FD JSR $ FDDA
1019- 60 RTS

```

运行前: \* 6000;05✓  
\* 6001;12 34 1D 56 20✓

运行后: \* 1000G✓  
\* D9  
\* 1030✓  
\* 1030-D9

程序 7.1 和 7.2 的异同点:

- 共同点,都是用单一循环编程,完成和数不超过 8 位二进制数的 N 个数累加。

- 主要区别,设计思想不同,程序 7.1 求和时是从前面往后面加的,而程序 7.2 求和时则是从后面向前面加。

此外,我们还看到在选用 ADC 指令时,其操作数部份的基地址不同,程序 7.1 中用的是 \$ 6001,程序 7.2 中用的是 \$ 6000,这是两种程序不同算法和字串长度选用的不同单元决定的。这是编写程序时应该注意的重要细节。

最后,我们还应指出,这两个程序在编制结构上虽然具有一定的灵活性(可以进行 N 个数的累加和),但通用性不够(和数不能超过一个 8 位二进制数)。解决的思路是设法保存进位的值或采用多字节加法程序。

#### [例 2]找最大值

在一段数据中寻找最大的元素。设此段数据个数放在 \$ 1A 单元, \$ 1B- \$ 1E 单元中存放数据,要求结果放在 \$ 1F 单元。并设各数据皆为无符号的 8 位二进制数。见程序 7.3。

```

1000- A6 1A LDX $ 1A ;计数器初值为数的个数
1002- A9 00 LDA # $ 00 ;设最大值=0
1004- D5 1A CMP $ 1A,X ;下一个数>最大值吗?
1006- B0 02 BCS $ 100A ;否,维持 A 中最大值不变
1008- B5 1A LDA $ 1A,X ;是用该数代替最大值
100A- CA DEX ;X-1
100B- D0 F7 BNE $ 1004 ;所有数都比较完了? 否,循环
100D- 85 1F STA $ 1F ;是,存最大值
100F- 60 RTS ;结束

```

运行前: \* 1A;04 05 FF 15 E3

运行后: \* 1000G✓  
\* 1F✓  
\* 001F-FF

程序 7.3 虽然较短,但因用了比较指令和两条条件转移指令,使程序阅读有些困难,下面作一些简单解释。

开始, X 寄存器作为计数器用置初值(1A)→X, 即 X=04,接着在累加器 A 中放了一个立即数 00,表示找数开始前假想的一个最大值为 0,执行第三条指令 CMP \$ 1A,X 时,表示 (A)-(1A+X)=(A)-(1E)=0-E3,显然不够减,P 寄存器中的 C 标志置 0,执行第四条指令 BCS 时,由于执行前一条指令后 C=0,所以不发生转移而执行下一条指令 LDA \$ 1A,X,即将 (1A+X)→A,(1E)→A,(A)=E3,可见这时已用 E3 代替原来假想的最大值 0。执行第六条指令 X-1→X,只要 X 不为 0,则执行 BNE 指令后循环上去转至 \$ 1004,由于此时 X=03,故再次执行 CMP 指令,(A)-(1A+X)=(A)-(1D)=E3-15,够减,C=1,又执行 BCS 指令,因此时 C=1,故发生转移到 \$ 100A,注意此时 A 中仍为 E3,维持最大值不变。X 又减 1,为 02,不为 0,Z=0,又循环上去。执行 CMP 指令,(A)-(1A+X)=(A)-(1C)=E3-FF,不够减,C=0,执行 BCS 指令时由于 C=0 执行下一条指令 LDA \$ 1A,X,即将 (1A+X)→A,(1A+02)→A,(1C)=A,(A)=FF,可见这时已用更大值 FF 代替次大值 E3。……,下面的过程重复执行上述类似的步骤,每一次循环后都将更大的数放入 A 中,不断循环,直至比较完所有的数,最后将最大值(真正的)放入 \$ 1F 中,结束。

综上所述,分析程序时应注意:

- 什么时候循环,循环多少次,以及何时退出循环,都由实现分支的条件语句控制着,循环中包含着分支。

- 比较指令 CMP 表示 A-M,够减 C=1,不够减 C=0;BCS 指令在 C=1 时转移,C=0 时执行下一条指令;BNE 指令表示结果不为 0(Z=0)时转移,而结果为 0 时(Z=1)继续执行下一条指令。记住这些对看懂程序大有好处。

#### [例 3]确定负数的个数

已知数据块的长度放在 \$ 6000 单元中,而数据块本身从存储单元 \$ 6001 开始存放,要求将其中负数的个数放在 \$ 07 单元。

如:6000;06 68 F2 87 30 59 2A

因 \$ 6002 和 \$ 6003 两个单元的值均为负数,所以程序设计运行后的结果应有:(07)=02。

分析:

- 确定一个数据块中负数的个数,就是查找数据块中数的最高位为 1 的个数。

- 由于本例中只有正数和负数两种,而没有零值,因此,选用 BPL 指令将是合适的,因为它以 N 标志作为判别标准,若 N=0,表示正数,执行分支动作;若 N=1,表示负数,则执行下一条指令。

- 由于本例中有 6 个数(为简单起见只选了 6 个数,原则上可以有 N 个数,N<256),必须逐一比较、检查才能确定那些是负数,那些是正数,而这种重复性的比较显然可用循环的方法处理。

- 在选用 BPL 指令之前,选用取数指令 LDA \$ 6001,X 也是顺理成章,一是因为只有取出一个数

来,才好判别是否负数;二是 LDA 指令执行后对符号位 N 及零标志 Z 皆有影响;三是采用绝对 X 变址寻址方式的 LDA 指令,只要改变 X 值,即可改变指向下一个数据的地址。

• 为了控制循环的次数,必须设置计数器,例如选用 X 寄存器作为计数器。为了统计负数的个数,也应设置一个负数计数器,例如选用 Y 寄存器。

根据上述分析,容易写出源程序 7.4。

程序 7.4

```

1000- A2 00 LDX # $ 00 ;计数器置初值
1002- A0 00 LDY # $ 00 ;负数计数器置初值
1004- BD 01 60 LDA $ 6001,X ;取一个数
1007- 10 01 BPL $ 100A ;是负数吗? 否,转
1009- C8 INY ;是,负数个数加 1
100A- E8 INX ;计数器加 1
100B- EC 00 60 CPX $ 6000 ;所有数都检查完了吗?
100E- DO F4 BNE $ 1004 ;否,继续取数检查
1010- 84 07 STY $ 07 ;是,存负数个数
1012- 98 TYA ;Y→A
1013- 20 DA FD JSR $ FDDA ;显示
1016- 60 RTS ;结束

```

运行前: \* 6000;06 68 F2 87 30 59 2A

运行后: \* 1000G

\* 02

\* 07

\* 0007-02

讨论:若要确定正数的个数,如何编程。

这个问题同确定负数的个数相类似,可以仿照程序 7.4 的模式编程。但最为简捷的方法是只要改动一个指令,您能指出来吗?

在您思考之前,最好先不要看下面的结果。

事实上,只要将程序 7.4 中的 BPL 指令,改为 BMI 指令。因为,这两条指令虽然都以 N 标志作为判别的依据,但两者转移的条件是不同的。BMI 指令是当 N=1 时,表示负数转移;而当 N=0 时,表示正数继续执行下一条指令。

实际改动时,只要改动程序 7.4 中 \$ 1007 单元的值,即将 10 改为 30,这是因为两条指令的操作码不同,代表的意义相异。

当然,经改动后,Y 寄存器中存放的是正数的个数,显示出来的也是正数的个数。

[例 4]确定正数、负数和 0 的个数

设数据块长度为 06,要求这 6 个数中逐一确定正数、负数和零的个数。

分析:

• 这个问题相对于仅仅判定一段数据中负数的个数或正数的个数来说,要复杂一些。因为,当读取一个数据时,要做几种判断。

• 可以选用 BNE 指令,先判断是否为 0,若是 0,则记录 0 的个数;若不是 0,再用 BPL 指令,以判别正、负数。在用 BPL 指令判别正、负数时,是负数立即存其

个数;不是负数跳转并存放正数个数。

• 为了统计数据块中正数、负数和零的个数,必须设置三个暂存单元,如负数、零、正数的个数分别存放在 \$ 10, \$ 11 和 \$ 12 单元中,在未统计各自个数前,应使这三个单元为 0,这就是程序的初始化条件。

• 在每一个数的性质判定以后,要做两件事,一是存每一个数的个数,这可以通过存储单元内容加 1 来解决,如用 INC \$ 10, INC \$ 11 或 INC \$ 12;二是让计数器加 1,并判所有的数是否处理完,这可以用 INX 和 CPX \$ 03 两条指令完成,其中 \$ 03 单元为存放数据块的长度,X 寄存器为计数器,当所有数都判别完以后结束,而只要有一个数未判完,即应循环下去再判。初始化时也应设置 X=0。

• 每判定一个数的性质以后,都要比较是否已是最后一个数,这可以用一个共同的子程序来实现,这样做可以减少相同操作的编程,使程序简化。关于子程序的设计我们将在下一节介绍。数据放在 \$ 04— \$ 09 单元中。

由上分析,不难写出源程序 7.5。

程序 7.5:

```

1000- A9 00 LDA # $ 00 初始化,累加器,
1002- 85 10 STA $ 10 正、负、零的
1004- 85 11 STA $ 11 个数存贮单元,
1006- 85 12 STA $ 12 计数器置 0。
1008- A2 00 LDX # $ 00
100A- B5 04 LDA $ 04,X ;取一个数
100C- D0 06 BNE $ 1014 ;非 0,转
100E- E6 11 INC $ 11 ;是 0,0 的个数加 1
1010- 20 1E 10 JSR $ 101E ;转子程序
1013- 60 RTS ;返回
1014- 10 06 BPL $ 101C ;不是负数,转
1016- E6 10 INC $ 10 ;是负数,负数个数加 1
1018- 20 1E 10 JSR $ 101E ;转子程序
101B- 60 RTS ;返回
101C- E6 12 INC $ 12 ;正数个数加 1
101E- E8 INX ;计数器加 1
101F- E4 03 CPX $ 03 ;所有数比较完吗?
1021- D0 E7 BNE $ 100A ;否,循环
1023- 60 RTS ;是,结束

```

运行前: \* 03;06 68 F2 87 00 59 2A

运行后: \* 1000G

\* 10.12

\* 0010-02 01 03

程序 7.5 编制中有几点技巧:

• 一是仅用一条指令 LDA # \$ 00,给三个存储单元 \$ 10, \$ 11, \$ 12 赋初值 0,而不是用三条 LDA # \$ 00 指令,这样做程序清晰,又节省内存。

• 二是调用 \$ 101E— \$ 1023 单元的子程序并正确返回。因为在判断一个数的性质以后,计数器指针加 1,以便正确指向下一个数据;同时,就程序的整体而言,还必须判别是否比较完全体数据,因此,编制一段具有共同功能的子程序实为必要,目的已如前所述。

·注意了主程序和子程序之间的正确衔接。调用子程序后必须注意正确返回,这不仅要要在\$1023单元放置一条由子程序返回到主程序断点处的RTS指令;而且对本程序来说,\$1013和\$101B两个单元也必须放置RTS指令,否则程序不能正确运行。

[例5]数据插入

假定存储单元\$6000中的内容6B,尚未出现在序列中,则把该单元内容增加到此序列中。已知序列的长度存放在\$6001单元,而序列本身从\$6002单元开始存放。

示范题:(6000)=6B  
 (6001)=04  
 (6002)=37  
 (6003)=61  
 (6004)=38  
 (6005)=1D

结果:(6001)=05  
 (6006)=6B

该单元(6B)被附加到序列中,因原序列中没有此单元,这个序列的长度增加1。见程序7.6。

程序7.6:

```

1000- AD 00 60 LDA $ 6000 ;取欲插入的数
1003- AC 01 60 LDY $ 6001 ;取序列长度
1006- D9 01 60 CMP $ 6001,Y ;比较,相同吗?
1009- F0 0F BEQ $ 101A ;是的,转$101A
100B- 88 DEY ;否,Y-1→Y
100C- D0 F8 BNE $ 1006 ;所有数都比较完
 ;吗?否,转
100E- EE 01 60 INC $ 6001 ;是,序列长度加1
1011- AC 01 60 LDY $ 6001 ;放增加后的长度
1014- 99 01 60 STA $ 6001,Y ;取增加的数
1017- 20 DA FD JSR $ FDDA ;显示
101A- 60 RTS ;结束

```

运行前:\*6000:6B 04 37 61 38 1D

运行后:\*1000G✓

\*6B

\*6000.6006✓

\*6000-6B 05 37 61 38 1D 6B

程序7.6编制要点:

·取欲插入的数,与序列中的其它数相比较,如果序列中原来就有和插入的数相同的数则结束。

·取欲插入的数,与序列中的数逐一比较,只要有一个数不同,就将长度减1,只要未比较完全部序列的数,就循环再比,直到比较完全部数据,原数据序列长度加1,并显示插入的结果。

说明,程序7.6也可采用绝对X变址寻址方式安排CMP指令和STA指令,同时改LDY\$6001指令为LDX\$6001指令,即程序中所有有Y的地方,改成X,并改正相应指令的操作码,也能正确运行。

[例6]单个数据块的传送

将存放在\$6000-\$60FF地址单元内的256个数据顺序传送到\$7000-\$70FF单元中去。

分析:本题虽然数据量较大(256个),但设计思想比较简单,取一个数,送一个数,计数器加1,指向下一个数据地址,再取,再送,只要没有取完送完,就继续循环,反之结束。故有程序7.7。

程序7.7:

```

1000- A2 00 LDX # $ 00 ;计数器置初值
1002- BD 00 60 LDA $ 6000,X ;取数
1005- 9D 00 70 STA $ 7000,X ;送数
1008- E8 INX ;计数器加1
1009- D0 F7 BNE $ 1002 ;送完了吗?否,转
100B- 60 RTS ;是的,结束

```

如果我们将题目改一下,将\$6000-\$60FF单元中的数据块按相反顺序传送到\$7000-\$70FF单元中去,程序又是如何编制呢?

这个题目的解题方法很多,今摘其一如程序7.8。

程序7.8:

```

1000- A2 00 LDX # $ 00 ;源数据计数器初
 ;值
1002- A0 FF LDX # $ FF ;目的数据计数器
 ;初值
1004- BD 00 60 LDA $ 6000,X ;取源数据
1007- 99 00 70 STA $ 7000,Y ;送入目的地址
100A- E8 INX ;源计数器加1
100B- 88 DEY ;目的计数器减1
100C- D0 F6 BNE $ 1004 ;次数不够,循环
100E- 60 RTS ;次数够了,结束

```

程序中用了三条取数指令和一条存数指令,并使用了两个变址寄存器X和Y,作为源数据块计数器和目的数据块计数器用。应该指出,变址寄存器X(或Y)都是八位寄存器,在编程中常被当作一个计数器来用。它们可以由指令控制而被置成一个常数,并能方便地用加1(INX,INY)、减1(DEX,DEY)、比较(CPX,CPY)操作来修改和测试其内容,从而使得程序能够方便灵活地处理数据块、修改地址指针、控制循环次数等。而在程序7.8中由于要处理数据块按逆序传送,因而一个变址寄存器就显得不够用,所以既用了变址寄存器X,又用了变址寄存器Y。

[例7]多个数据块的传送

将\$6000-\$600F单元的第一个数据块送到\$7000-\$700F单元中,将\$6100-\$610F单元的第二个数据块传送到\$7100-\$710F单元中,将\$6200-\$620F单元中的第三个数据块传送到\$7200-\$720F单元中去。

分析:这是三个数据块的传送问题,由于是顺序搬迁,而且数据块长度相同(都是16个),使得问题求解不至于过分复杂。数据块可以一个一个送,送一个数据块就是一个循环,当送完一个数据块后只要改变一些地址指针,就可以再送另一个数据块,直到全部送完。因此,一个循环程序是处理不了的,所以本题应是一个多重循环问题。

我们先给出源程序清单,见程序7.9,然后看一下运行结果,最后再对程序编制思想作一说明。

程序 7.9:

```

1000- A2 00 LDX # $00 ;变址计数器 X 置 0
1002- A0 10 LDY # $10 ;数据块长度送 Y 寄存器
1004- A1 1A LDA ($1A,X) ;传送一个源数据到目的地址第一次将(6000)→7000
1006- 81 FA STA ($FA,X) ;修改源地址,使之增 1
1008- F6 1A INC $1A,X ;修改目的地地址,使之增 1
100A- F6 FA INC $FA,X ;一个数据块传送完了吗?
100C- 88 DEY ;未送回,循环
100D- D0 F5 BNE $1004 ;三个数据块传送完了吗?
100F- E0 04 CPX # $04 ;是,结束
1011- F0 05 BEQ $1018 ;否,将 X 增 2,为取下一个数据块作准备
1013- E8 INX ;
1014- E8 INX ;转入下一个数据块
1015- 4C 02 10 JMP $1002
1018- 60 RTS ;结束

```

运行前: \* 1A:00 60 00 61 00 62

将三个源数据块首地址送入 \$ 1A- \$ 1F 单元

\* FA:00 70 00 71 00 72

将三个目的块首地址送入 \$ FA- \$ FF 单元

\* 6000.600F

6000- 01 01 01 01 01 01 01 01

6008- 01 01 01 01 01 01 01 01

\* 6100.610F

6100- 02 02 02 02 02 02 02 02

6108- 02 02 02 02 02 02 02 02

\* 6200.620F

6200- 03 03 03 03 03 03 03 03

6208- 03 03 03 03 03 03 03 03

将第 1,2,3 个源数据块的数据分别送入 \$ 6000—\$ 600F, \$ 6100—\$ 610F, \$ 6200—\$ 620F 单元中。

运行后: \* 1000G

\* 7000.700F

7000- 01 01 01 01 01 01 01 01

7008- 01 01 01 01 01 01 01 01

\* 7100.710F

7100- 02 02 02 02 02 02 02 02

7108- 02 02 02 02 02 02 02 02

\* 7200.720F

7200- 03 03 03 03 03 03 03 03

7208- 03 03 03 03 03 03 03 03

完成了正确传送。

程序 7.9 编制思想说明:

• 源数据块的首地址放在 \$ 1A—\$ 1F 单元,目的数据块的首地址放在 \$ FA—\$ FF 单元,这些单元都是采用零页地址单元。这样处理比用多组 LDA 和 STA 指令设置地址单元要简捷得多,同时也使初始化部份省去了不少语句。

• 程序中的第 1,2 两条指令,是安排了两个计数器,其中 X 变址寄存器作为修改源地址和目的地址的指针计数器,为读取下一个数据作准备,同时又兼作三个数据块是否全部送完的一个标志,配合 CPX # \$ 04 来处理。Y 变址寄存器则作为传送数据块长度的计数器,和 DEY, BNE 指令配合,以使决定一个数据块的所有数据是否全部送完。

• 程序中有两个循环,第一个是从 \$ 1002—\$ 1007 的外循环,第二个是从 \$ 1004—\$ 100E 的内循环。它们完成的任务是不同的,内循环主要完成一个数据块(16 个数据)的传送工作,而外循环则是完成三个数据块的传送工作。

• 循环程序中包含了分支转移,在内循环中,控制循环次数,判断一个数据块的所有数据是否全部送完,主要由条件转移指令 BNE 决定的(是否分支转移)。而在外循环中判断三个数据块是否全部送完则主要由条件转移指令 BEQ 决定的(送完结束,未送完修改地址指针再循环)。

• 程序中数据块的传送,主要选用了 LDA(\$ 1A, X)和 STA(\$ FA, X)这两条指令,它们都是采用先变址(X)间接寻址的寻址方式,其优点是,使得多个数据块的处理变得简单、容易。一般来说,处理各个数据块的传送问题,用先变址间接寻址方式的指令比较理想。当然也不是绝对的,本章的下面几节内容还将介绍多个数据块的传送方法,那时,我们还可以看到不少的编程方法和技巧。

关于循环程序的设计,我们暂介绍到这里。现在,对本节的内容作一简单小结。

• 循环程序的任务,在于完成大量的在处理形式上完全相同的重复性的计算工作。

• 循环程序的结构,包括初始化、处理、循环控制、结束四个部份。各部份有机结合,协同动作,缺一不可。

• 循环程序的应用,它几乎能解决名目繁多,形式各异的各种课题,是机器语言程序设计的核心和精华。

• 循环包括分支,分支控制循环,两者互相联系又相互制约。灵活而准确地使用比较指令,特别是条件转移指令,是高质量设计循环程序所必不可少的。

• 循环程序的嵌套,是指循环内还可以嵌套循环,一个循环内可以包含几个循环,但应注意内外循环的层次要清楚,内循环允许并列几个小循环,但不允许交叉。



## 初级程序员级软件考试辅导

# PC BASIC 自测试题解答与分析

北京市计算中心(100005) 李志刚

〔第一题〕这是一组填空题,共有五个小题。

(1)要求将给出的代数表达式改写成 BASIC 算术表达式。在改写时要将表达式中的代数运算符和函数转换成相应的 BASIC 的算术运算符和函数,应特别注意:①将三角函数中出现的度数利用公式“度数 \* (3.1416/180)”化成弧度。如  $\sin 45^\circ$  改写成 BASIC 表达式为  $\sin(45 * 3.1416/180)$ 。②在书写时两个因数间的乘号 \* 不能省略或用 · 表示。完整的 BASIC 表达式为  $(2 * \sin(45 * 3.1416/180) + \cos(30 * 3.1416/180)) / (3 * \text{ABS}(X) * \text{LOG}(Z-5))$ 。

(2)SWAP X, Y 语句的作用是交换数据存储变量 X 和 Y 的值。这是一条常用语句,在某些 BASIC 版本中没有这条语句,为了实现同样的功能,就要用一组赋值语句来实现。由于变量被赋值后,原有内容会丢失,因此,交换两个变量的内容(值)须借助一个临时变量。交换过程是这样的:先将第一个变量的内容赋值给临时变量保存,再把第二个变量赋值给第一个变量,然后再将临时变量的内容赋值给第二个变量,由此完成了两个变量内容的交换。本题语句组可以是:

Z=X;X=Y;Y=Z

(3)这是一个三层嵌套的函数表达式。按照 BASIC 表达式中算符的优先次序,首先计算最里层的函数值,得到字符串“1245”最右边长度为 2 的子串“45”,再计算  $\text{VAL}(\text{“45”})$  得数字值 45,然后求开方函数  $\text{SQR}(45+4)$  得结果为 7。

(4)把当前驻留在内存中的 BASIC 程序写入磁盘有以下几种方式:

- ①SAVE “文件名”
- ②SAVE “文件名”,A
- ③SAVE “文件名”,P

其中,文件名包含了盘符和完整路径。方式①使程序以压缩二进制格式存储。这种方式节省存储空间,程序可运行,可在 BASIC 下列表及编辑,不能在 DOS 下显示或打印。方式②是最常用方式,参数 A 使程序做为一系列 ASCII 码字符存储,在任何环境下均可编辑、列表和打印,也可以做为数据文件读入。方式③较少使用,P 参数把程序以编码二进制格式存储。这种方式保存的程序不能修改、显示和打印,可以对源程序在某种程度上起到保护作用。按题目要求,命令为 SAVE “A:PROG.BAS”,A 也可以写成 SAVE “A:PROG”,A,如省去扩展名“.BAS”,系统在这种场合下会自动加上。

(5)栈是一种所有插入和删除均限定只能在一端进行的特殊类型的线性表,它遵从后进先出的操作规

则。在设计 BASIC 的计数循环语句 FOR/NEXT 时采用了堆栈技术,因此本题答案应填写“后进先出”。

〔第二题〕本题有五道选择填空题

(1)流程图是用规定的图形、连线和文字说明表示算法的一种图形。由于流程图直观、清晰、易读易交流,所以目前使用较为广泛。标准流程图是指由我国国家标准局批准的国家标准 GB1526—89,即按《信息处理—数据流程图、程序流程图、系统流程图、程序网络图和系统资源图的文件编制符号及约定》而画出的流程图。GB2312—80 是汉字信息交换用的国家标准;ISO 5807—85 是指国际标准化组织公布的标准《ISO5807—85 Information processing—documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts》。本题答案是 B。

(2)不论用何种计算机语言编制程序,一般都离不开顺序、分支(条件)和循环这三种基本的控制结构。采用自顶向下,逐步求精的结构化程序设计技术,并运用三种基本控制结构,可以设计满足各种复杂功能需求的程序结构。这三种基本控制结构的共同点是只有一个入口和一个出口。采用基本结构按照结构化程序设计方法编制的程序,不仅可读性强,也易于维护和扩充。本题的答案是 B、G。

(3)结构化程序设计方法已被人们广泛采用,它的基本思想是自顶向下,逐步求精。例如,把一个较复杂的问题逐步分解成若干个较简单或功能单一的问题,对每一个这样的问题,可以用程序的基本结构实现算法,这种方法就称为逐步求精法。本题答案是 D。

(4)按照结构化方法编制程序,根据模块(结构)只有一个入口、一个出口的原则,模块间的调用关系一般不用 GOTO 语句实现,这样程序不仅可读性强,还避免了因过多或不适当使用 GOTO 语句而产生的控制逻辑错误。在模块内部在不使用 GOTO 语句将产生大量程序冗余的情况下,也可适当地采用 GOTO 语句。本题答案是 D。

(5)在主程序和外部子程序之间的信息传递一般是通过一组变量来实现。在主程序中调用子程序语句内的变量表,一般称为形参;相应地在子程序头也必须与调用语句变量表在个数、顺序和类型上相匹配的一组变量,称为实际参数。本题的答案是 B、D。

〔第三题〕本题有五段简单的 BASIC 程序,先阅读程序,然后指出程序运行后生成的结果。

(1)阅读这段程序,要掌握 READ, DATA 和

RESTORE 语句组的实现功能和特点。在 READ 语句中变量的类型与 DATA 语句中数据的类型要一一对应，DATA 中的数据个数不得少于 READ 读取的变量次数。DATA 是不可执行的静态语句，它可出现在程序的任何部位，数据项的顺序只与 DATA 语句排列的先后顺序有关，可以理解为在程序执行前，系统已把全部数据项按各 DATA 语出现的先后和 DATA 中数据的自然顺序放在 DATA 数据区中，同时将读取指针指向第一个数据，如下图

读取指针  $\rightarrow$

| DATA 数据区       | 1   | 2      | 3   | 4      | 5   | 6      | 7   |
|----------------|-----|--------|-----|--------|-----|--------|-----|
| 10 N=5         | 60  | M=5    | 60  | M=4    | 60  | M=3    | 60  |
| 20 M=6         | 70  | M<>1   | 70  | M<>1   | 70  | M<>1   | 80  |
| 30 F=1         |     |        |     |        |     |        | 90  |
| 40 GOSUB 60    | 80  | F=5!   | 80  | F=4!   | 80  | F=3!   | 100 |
|                | 90  | M=6    | 90  | M=5    | 90  | M=4    |     |
| 50 PRINT F;END | 100 | RETURN | 100 | RETURN | 100 | RETURN |     |

(3)逻辑运算符把一些真值或假值连接起来，运算的结果也是具有真值或假值的逻辑值。正如关系运算符通常用来决定程序的流向一样，逻辑运算符也用来连接两个或多个逻辑变量，多在 IF 语句中使用，由逻辑表达式的值来确定程序的流向。在本程序中，真值为 -1，假为 0。可利用逻辑运算的性质，由程序执行的次序，列出下列真值表，由表很容易得到答案是 C。

| 循环次数 | A  | B  | A AND B | NOT A | (A AND B) + (NOT A) |
|------|----|----|---------|-------|---------------------|
| 1    | -1 | -1 | -1      | 0     | -1                  |
| 2    | -1 | 0  | 0       | 0     | 0                   |
| 3    | 0  | -1 | 0       | -1    | -1                  |
| 4    | 0  | 0  | 0       | -1    | -1                  |

(4)BASIC 中的函数主要有标准函数与自定义函数两类。本题主要是考虑对自定义函数的理解和使用。自定义函数是用户为解决自己的特定问题而定义的非标准函数，用 DEF FN 语句来完成定义。一经定义就可以象标准函数一样在表达式中调用。本题 20 语句定义了一个 A 函数，然后由 30 和 50 语句分别调用。在读程序写结果时，不仅输出数据要正确，输出格式也要符合要求才行。40 和 60 打印语句格式，在答案中，只有 D 满足要求，再由自定义函数分别计算出 30 和 50 赋值语句，结果为 269 和 130.5。答案是 D。

(5)这是一段图形打印程序，它的主要特征在打印控制部分。研究这部分程序就能很快确定正确答案。20 语句到 50 语句是打印部分，利用 20 到 40 语句的计数循环结构印出一行“\*”字符，50 语句空打印产生回车换行。每个打印行的起始列是第一列，第 i 行 ( $1 \leq i \leq 5$ ) 印出 i 个“\*”字符。由 10, 60, 70 和 80 语句中 IF 和 GOTO 组成了一个直到型循环，控制图形印完五行后程序结束。答案是 B。

[第四题]提供了四个具有语法错误的程序段，执行后系统将显示不同的出错信息，阅读出错信息并弄清含意后再研究各程序段，找出程序出错点并与各出错信息比较，就可确定正确答案。如果熟悉出错信息，

语句 10 将 1, 2, 3 分别读入 A, B, C 中；指针指向 4，语句 20 的 RESTORE 恢复指针指向 1；30 语句不执行，40 语句分别将 1, 2, 3, 4, 4, 5 读入 G, F, E, D, C, B 中；50 语句不执行；60 语句打印变量 A~G 内的值，结果是 1, 5, 4, 4, 3, 2, 1。答案是 C。

(2)这是一段用递归调用方法计算 N! 的程序，主要利用了主—子程序间调用和返回的关系。为便于了解执行过程，将程序的执行序列图示如下，答案是 C, F 值为  $5! = 120$ 。

|     |        |     |        |     |        |
|-----|--------|-----|--------|-----|--------|
| 60  | M=3    | 60  | M=2    | 60  | M=1    |
| 70  | M<>1   | 70  | M<>1   | 80  | M=1*1  |
|     |        |     |        | 90  | M=2    |
| 80  | F=3!   | 80  | F=2!   | 100 | RETURN |
| 90  | M=4    | 90  | M=3    |     |        |
| 100 | RETURN | 100 | RETURN |     |        |

又有 BASIC 的基本知识，这类题目较易完成。

(1)这是执行两次的一层计数循环结构，在循环体内的 20 语句是对数组 A 进行维数定义。在两次循环中，数组 A 被定义了两次。BASIC 系统不允许对同一数组进行一次以上的定义，再参照答案组中的 A 是“重复定义”的出错信息，刚好与本题吻合。由此也确定了正确答案。在实际应用中，如需要对某数组再次定义维数，可先使用 ERASE 语句取消这个数组，然后再定义。本程序可以改为：

```
10 FOR I=2 TO 3
20 DIM A(I)
30 A(I)=I
40 PRINT A(I),
45 ERASE A
50 NEXT I
60 END
```

就不会出现语法错误了。

(2)语句 10 定义了一维数组 A，它的下标上界是 5。20 到 50 语句是计数循环结构，执行 10 次循环。循环体内的 30 和 40 语句引用了数组 A 的元素，当循环五次后，控制变量 I 值为 6；进入第六次循环的 30 语句时，因下标  $I > 5$ ，产生数组下标越界错。答案为 C。

(3)仔细阅读这段程序，容易看出 20 行有两条 PRINT 语句，按语法要求，同一行号下有多条语句时，各语句之间用冒号“:”分隔，程序中丢掉了冒号，将产生语法错误。答案是 B。

(4)从逻辑上看这段程序是用 READ 语句依次读入 DATA 数据区的四项字符常量，连接后在屏幕上印出。即便不考虑它的逻辑关系，仅从字面上看也很容易发现 20 语句中存在的语法错误。等号左边是字符型变量 A\$, 右边的 YES 按 BASIC 的规定，它是一个数值型的变量名，比较运算符两边的数据类型不一致将产生“数据类型不匹配”错误。答案是 D。对 20 行的 YES 加上一对双引号后，程序就能正确运行了。

[第五题]程序实现三种主要功能：①随机产生 10

个 1000~9999 间的整数;②从这 10 个数中找出最大数和最小数;③取最大数的 10 位上的数字和最小数个位上的数字重新组成一个两位数。由三种功能将程序划分为三部分,10~60 行语句产生 10 个四位随机整数并存入数组 A 中,30~60 语句是一层计数循环,执行 10 次 40 赋值语句,每次把某个数赋值给数组 A 的一个元素。40 行缺少的部分是产生随机数的表达式。产生随机整数的公式为:INT(RND \* (上界-下界+1))+下界,产生四位随机整数的表达式应为

INT(RND \* (9999-1000+1))+1000

可简化成 INT(RND \* 9000) + 1000 或 INT(RND \* 9000 + 1000)。70~120 语句实现从 10 个四位整数中选出最大和最小数。由 120 语句可知,结果的最大、最小数存放在 A(1)和 A(2)中。开始的 70 语句也是经过比较后,大数存 A(1),小数存 A(2)。80~110 语句是执行 8 次的计数循环,将遍历 A 中第三个到第十个元素 90 和 100 语句分别涉及到这 8 个整数 A(I)与 A(1)或 A(2)可能进行的交换。由 120、70、90 和 100 语句可以推断,在整个选大小的比较和交换的过程中,A(1)和 A(2)始终分别存放最大数和最小数,算法是:当 A(I) > A(1)时,A(I)与 A(1)交换;否则当 A(I) < A(2)时,A(I)与 A(2)交换;因此在 90 行应填入 A(1) < A(I);100 行填入 A(2) > A(I)。130~150 语句是将最大数 10 位上的数字与最小数个位上的数字组成一两位数并印出。由程序的 150 语句可知,130 的 A\$ 存放最大数的 10 位上的数字,140 行的 B\$ 存放最小数的个位数字。我们知道,取数值型数字中的某位的方法之一是先将数转串,再用取子串函数实现。取 A(2)的个位数字用右子串函数较方便,140 行中填入 RIGHT\$(STR\$(A(2)),1)。在 PC BASIC 中数转串后,符号占最左边一位,因此用取子串函数取 A(1)中 10 位上数字时,应从第四位起取 1 位。130 行填入 MID\$(STR\$(A(1)),4,.)。

〔第六题〕由程序说明,这是一段按特定要求对一正数序列实现重新排列的分类问题。以某一正数序列的第一项 K1 为标准,比 K1 小的,放在 K1 的左边;比 K1 大的放在 K1 的右边。从实例可以看出,对重新排列后的新序列,K1 左边和右边的两个子序列中的数分别小于或大于 K1,而并不要求它们自身有序。应首先阅读程序,根据程序提供的结构和已有语句来推断使用的排序方法,在此基础上,补足未完成的程序语句。语句 10 由 INPUT 语句通过键盘为变量 N 提供一个数值,那么变量 N 在程序中充当何种角色可从已有程序语句中识别。从 40 到 60 语句是一循环结构,在 N 次循环中,通过键盘向数组 A 供 N 个数,由此可知,这个序列中正整数个数为 N,数组 A 的下标上界为 N。根据数组定义在先,使用在后的性质,30 语句应填入 DIM A(N)。为了保证序列中的成员为非负整数,在 50 行空处应填入条件表达式 A(I) <= 0 OR A(I) <> INT(A(I))。从 10 行到 70 行都是准备阶段,80 行开始是本题的难点——排序部分。仔细研究实例,可以发现,在排

序后的序列中,凡是比 K1 大的数,除了向右移动外,它们原来的顺序没有改变;而小于 K1 的数全部插在 K1 的前边,排列次序与原来相反,由此可以推测,在比较之后,如果 A(I) 小于 K1,则将 A(I) 插入在序列第一项之前。插入的过程是这样的:先将 A(I) 送入某一临时变量 X 保存,然后把 A(2)到 A(I-1)的全部元素向右平移一位,移动的次序为:A(I-1)→A(I),A(I-2)→A(I-1),...,A(1)→A(2);X→A(1)。实现上述动作的程序是 110~150 语句,100 语句对 A(I) 进行识别,如果 A(I) > K1 那么对 A(I) 不做处理,控制转移到外层循环的出口。100 行的空应填入 A(I) > A(T),A(T) 就是 K1,变量 T 在循环开始前值为 1,此时 K1 位于序列的第一项,如果 A(I) > A(T),A(I) 和 A(T) 的位置都不改变,当 A(I) < A(T) 时,要求将 A(I) 插入到序列第一项之前,在插入之前,子序列 A(1),...,A(T),...,A(I-1) 向右平移一位(1 ≤ T ≤ I-1),因此在插入操作完成后,指示 K1 项的下标变量 T 应增 1,160 行空处填入 T=T+1。120~140 语句是一层计数循环,功能是实现平移,130 处填入 A(J)=A(J-1),2 ≤ J ≤ I。在缺少的语句补全之后,最好使用说明中的实例代入程序中走一遍,以检查结果是否正确。

〔第七题〕首先要掌握流程图中各种图符的含义,在正确理解题意的基础上,将图中各编号处的内容补齐。这是一个分支嵌套结构的程序流程图,其中变量 X 读入年经济效益数,S 累加发奖金总数,N 累加合理化建议个数。由流程图可以直接看出变量 Y1,Y2,Y3,Y4 是分别为奖金 2000,1000,500,200 设置的累加器。程序首先对变量 S,N,Y1~Y4 初始化,然后进入当循环结构,循环条件是 X ≠ 0,X 保存输入的年经济效益数,当 X ≠ 0 时,应按给定的四个等级标准将 X 值所对应的等级奖金数累加到 Yi 中(i=1,2,3,4)。空①所在的判定符下,如果条件为真,则将 2000 累加进 Y1,①处填入条件 X ≥ 100;同理②处填入条件 100 > X ≥ 10;③处填入条件 10 > X ≥ 1。题中变量 X 的值,由题意假定取值范围为 0 和正数。当输入结束时,X 值为 0,程序在打印 S 和 N 值之前应进行计算。由于 Y1~Y4 中分别存放着四个等级的效益总数,因此④处应填入 Y1+Y2+Y3+Y4,⑤处应填入合理化建议总个数 Y1/2000+Y2/1000+Y3/500+Y4/200。

答案:

一①(2 \* SIN(45 \* 3.1416/180) + COS(30 \* 3.1416/180)) / (3 \* ABS(X) \* LOG(Z-5))

②Z=X;X=Y;Y=Z ③7

④SAVE "A:PROG.BAS",A ⑤后进先出

二①B ②B G③D ④D ⑤B D

三①C ②D ③C ④D ⑤B

四①A ②C ③B ④D

五 INT(RND \* 9000 + 1000)

A(1) < A(I)

A(2) > A(I)

MID\$(STR\$(A(1)),4,1)

RIGHT\$(STR\$(A(2)),1)

## 第三届全国计算机软件人员竞赛(1992)将在京举行

### 主办单位

由中国软件行业协会考试指导中心、北京市科学技术协会、电子工业出版社、《计算机世界》等单位联合举办

### 参赛对象

竞赛选手产生自以下两途径:

1. 初级程序员级:按规定时间完成刊登于《电子与电脑》杂志今年第八期上刊出的有关初级程序员级软件水平考试辅导综合自测题,成绩优秀者。

程序员级:参加第四期全国计算机软件水平考试函授辅导班结业考试,成绩优秀者。(以上由中国软件行业协会考试指导中心推荐)

2. 各省市软件水平考试实施办推荐各1-2名。

3. 由台北县电脑同业公会负责推荐。

### 参赛条件及内容:

(1)初级程序员级 未获得计算机专业本科学历或助理工程师职称者。计算机知识、DOS、中西文录入、BASICA、DBASE III、IBM PC 及兼容机机种。

(2)程序员级 未获得计算机研究生学历或工程师职称者。计算机基础、程序设计(CASL 必答、FORTRAN、PASCAL、C、COBOL 后四种中任选其一)。

(3)年龄均在35周岁以下。

### 竞赛日期、地点

1992年10月下旬,北京清华大学。

两个级别各设:一等奖1名,二等奖2名,三等奖5名。鼓励奖若干名。将分别发给获奖证书、奖杯及其它纪念品。选手差旅及食宿一律自理,大会代为安排食宿。

### 联系地址:

1. 北京学院路29号 中国软件行业协会考试指导中心(邮编100083)

电话:2012233-322,传真:2024674

电报:4614,联系人:马素琴

2. 北京清华大学计算机系(邮编100084)

电话:256144-2373,联系人:金慧芬

### 六 DIM A(N)

A(I)<=0 OR A(I)<>INT(A(I))

A(I)>=A(T)

A(J)=A(J-1)

T=T+1

七①X>=100 ②10<=X<100 ③1<=X<10

④Y1+Y2+Y3+Y4 ⑤Y1/2000+Y2/1000+Y3/500+Y4/200

(上接32页)

2. 状态修改功能——可以对内部RAM、特殊功能寄存器、外部RAM空间、寄存器组按字节进行修改。此外还可以对单片机系统的位控区进行逐位修改。

3. 运行控制功能——可以控制单片机进行单步运行、多步跟踪运行、断点运行、连续全速运行。除全速运行外,其他各种运行方式都能在运行结束后,向用户提供运行后单片机系统的状态信息。

4. 数据传送功能——可以在外部RAM空间中进行SRAM(静态随机存储器)与SRAM与EEPROM(电可改写的只读存储器)之间进行数据传送。可以在外部RAM空间与EPROM空间(由EPROM固化板开辟的存储空间)之间进行数据传送。此外还能在单片机外部RAM空间与磁盘文件之间进行数据传送。

5. 其他功能:

1)反汇编功能:可以对单片机外部RAM空间中的机器码进行反汇编,以列表方式进行显示、存盘。

2)字符串搜索功能:可以在指定外部RAM空间中对指定字符串(字符串最多可由十个字符组成)进行搜索,并在屏幕上显示出指定字符串的存储地址(出现字符串中首字符的地址)。

3)数据块比较功能:可以对指定的二个数据块进行比较,若二个数据块中内容不同,则显示所有不相同单元的地址及内容。

4)EPROM编程功能:包括编程、读出、检查、检验等功能(必须与EPROM编程板配合)。

## 书 讯

《PC机用户实用维修技法》是一本面向广大微机用户的实用教程。该书的编著者集多年的教学与维修经验,根据用户的需要,重点总结了用户能够自行排除的故障,特别适用于非专业从事维修PC/XT及其兼容机的广大用户,亦可作为维修培训班教材。该书且适当兼顾到PC/XT和286微机的维修技术。全书约35万字,现已正式出版,压膜包装,定价7.2元/本(邮购请另加邮费10%)。需购者请直接汇款至安徽蚌埠86181部队计算机教研室涂从润收,信汇或批量购买可与其通信联系(邮编233000)。



# BJS—51 单片机实验系统

北京航空航天大学 盛焕鸣

编者按:单片机由于其功能强、价格低廉、抗干扰能力强、易于掌握,稍加一些外围设备就能方便地构成一个应用系统等特点,近年来在数据采集、工业机器人、仪器仪表等领域中已逐渐取代 Z-80 等八位机成为这些领域智能化的主力。各高等院校也逐步把单片机教学列入教学计划,研制一种专为单片机教学服务的教学实验系统也已经提到日程上。

为了满足这一需要,北京市单片机应用技术协会组织部分高校教师及有关技术人员研制开发了以 MCS—51 系列单片机为核心的 BJS—51 和以 MCS—98 系列单片机为核心的 BJS—98 两套单片机实验系统。本刊从本期起将连载 BJS—51 单片机实验系统的介绍,BJS—98 单片机实验系统将在以后介绍,以飨读者。

一个好的教学实验系统应该由程序编写与调试用的开发手段、模拟实际应用的输入输出手段、指导教学实验的实验大纲、指导书、程序清单等三部份组成。前者为实验系统的硬件部份,后者为实验系统的软件部份。

以往的教学实验大多采用市场上出售的单片机仿真器或带自开发功能的单片单板机作为调试开发手段,用外接面板来搭接实际应用的输入输出装置。由于面板接触不可靠,影响了实验的效果。

BJS—51 是一种专为教学实验而设计的单片机系统,在设计一开始就把开发手段、实验器、实验指导书三者紧密地结合起来,尤其着重于把调试手段与模拟输入输出手段的紧密结合。使学生既能方便地完成教学实验,又能从中学到接口设计与编程的有关知识。

BJS—51 教学实验系统采用模块式结构,以仿真器常用的单片机引脚排列方式来定义接口总线,从而使各种模块具有最大的兼容性,它们既能与 BJS—51 教学实验装置构成不同的配置,又能与各种 51 系列仿真器连接。

BJS—51 可以根据不同类型的教学实验建成不同的配置。

一、最小配置——最小配置用于除 A/D、D/A 外的绝大部份大学本科生的教学实验,以及初学者对单片机的学习。最小配置为一块 BJS—51 系统板。BJS—51 的结构见图 1。

BJS—51 系统板由二部份组成。右边为带有自开发功能的单片机扩展系统,左边为实验器。单片机扩展系统的原理电路图见图 2。

图中存储器系统配有四个 28 脚插座。它们的地址空间分别为 0000H~1FFFH ( $U_3$  及  $U_4$ )、4000H~5FFFH ( $U_5$ ),及 8000H~FFFFH ( $U_6$ )。其中  $U_3$  由  $\overline{PSEN}$  控制为 8031 的 ROM 空间, $U_4$  由  $\overline{RD}$  控制为 8031 的 RAM 空间。把 0000~1FFFH 分为二个空间是为了使该系统既能进行 51 系列单片机的汇编语言教学实验,又能在配上 BASIC—52 解释程序后进行 BASIC—52 教学实验。

$U_6$  可以通过边上的短路块配接为不同的容量。当将短路块插入左边的插针,在  $U_6$  座上配上 6264 则为 8KB 空间,若将短路块插入右边插针,在  $U_6$  座上配上 62256 则为 32KB 空间。

除  $U_3$  及  $U_4$  外, $U_5$  及  $U_6$  均为 8031 的混合空间,它们都既能作为用户的程序区,也能作为用户的数据区。 $U_5$  为 8155,其地址空间为 6000H~67FFH。

8155 的内部 RAM 用以存放监控程序使用的各种标志、数据以及保存用户程序的运行现场。8155 的口线 (PA 口, PB 口, PC 口) 连同 8031 的部份口线通过 40 线扁平电缆线 CZ2 引出。CZ2 可以与键盘显示板连接,以使用键盘 LED 方式进行调试。

另一插座 CZ3 是 BJS—51 的系统扩展插座,BJS—51 系统的所有扩展板均与 CZ3 相接。CZ3 的引脚排列基本上与 8031 的引脚排列的次序相同,仅用二条片选线 (7400H~77FFH, 7800H~7BFFH) 代替 8031 的时钟线,以提高信号的利用率。BJS—51 采用这种总线方式是为了最大限度地与其他仿真器兼容,因为大多数仿真器及自开发型单片单板机均带有这种总线插座,因此 BJS—51 的所有扩展板均能连接在这些装置上运行。

8031 的 RXD 与 TXD 通过晶体管 9012、9014 及其附加电路组成简易 RS232 接口。它可与 PC 机的标准 RS232C 连接,在通信程序支持下进行系统的编程与调试。由于 PC 机能够直接对单片机程序进行汇编以及对数据进行存盘等功能,随着 PC 机价格的下降,在 PC 机支持下开发单片机系统已成为单片机开发的主要形式,也是在最小配置下 BJS—51 的程序调试方式。

系统板的左部为实验器,它们包括:

1. 二个按键开关。用来模拟各种开关量的输入装置。因为开关量都是 0 与 1 二个状态,因而这二个按键开关可以模拟实际应用中的各种开关,诸如机械式开关、光电开关、自动线上计数器、行程开关、编码器等。

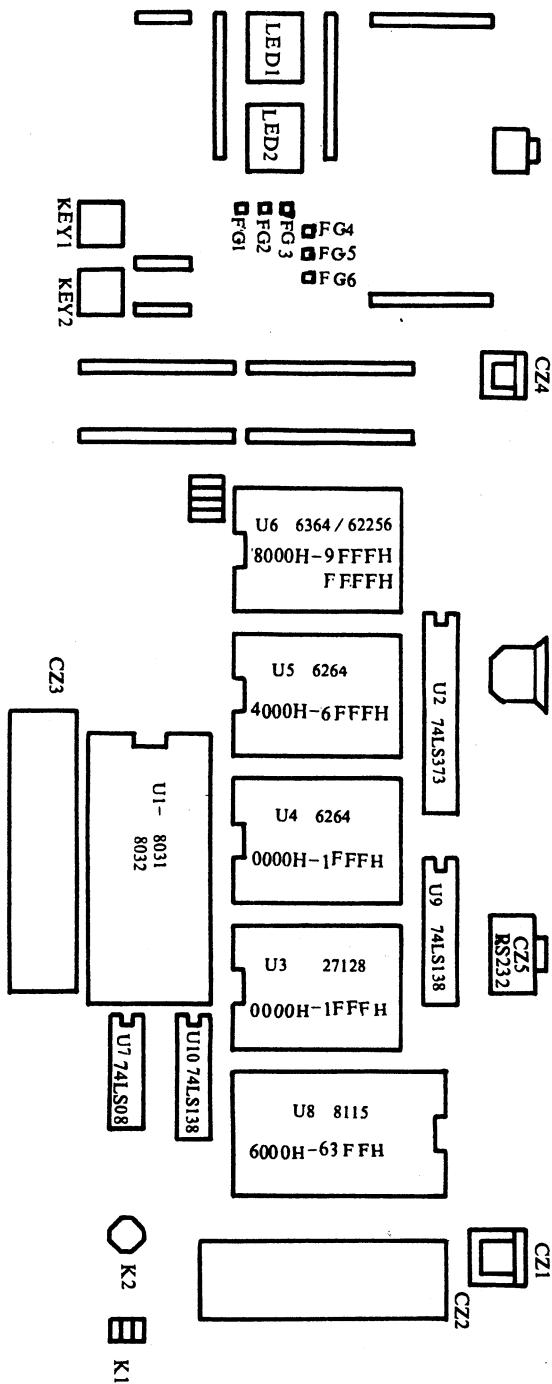


图1 BJS-51 系统板结构

2. 二个 LED 数码管。用来显示、输出二位数字,作为数码输出装置。诸如显示时钟、计数值以及进行静态或动态扫描方式的数字、字符输出实验。

3. 六个发光二极管。发光二极管为输出装置,可以模拟各种类型开关量的输出,例如开关的接通(亮)或断开(灭),数字输出的 0 与 1,阀的开与关等。此外六个发光二极管排列成二组“红绿灯”可以进行传统的交通灯实验。

4. 蜂鸣器。蜂鸣器也是输出装置,可以用于各种发声实验,输出音乐及警报等。

5. 二组晶体管驱动电路。用以对单片机的输出信号进行功率放大、驱动各种输出装置。

此外系统板上还配有二芯、三芯插座,以便从外部接入其他实验器(例如小电机……等)

所有实验器的引脚以及 8031 的引脚都被引到插针上,在实验时通过绕接器把需要连接的二点用导线接上便可进行实验。BJS-1 通过绕接方式进行实验时的接线,与使用面包板相比,不仅缩小了实验装置的体积,降低了成本,而且大大提高了接线的可靠性,从而提高了实验的质量。

系统板上配置了实验器,大大简化了教师的实验准备工作。

BJS-51 系统设计时,为了突出单片机的特点即“一片集成电路就是一台计算机”,所以在系统板及实验设计上都强调尽量使用单片机本身的资源来完成各种实验。

使用最小配置,可以熟悉在 PC 机支持下 51 系列单片机的调试方法,进行 51 系列单片机的程序编写汇编练习,以及各种简单的输入输出装置的驱动程序练习,直至进行复杂的交通灯试验。

二、基本配置——基本配置是在最小配置(一块系统板)的基础上再加接一块 LED 显示键盘板及 EPROM 写入读板。

LED 显示键盘板为带有 32 键及 6 个 LED 显示器的扩展板,同时还扩有打印接口。使用时,显示键盘板接在系统板的 CZ2 插座上。其电原理图见图 3。LED 显示器上八条段控线由系统板上 8155 的 PB 口通过驱动电路(QD<sub>2</sub>)提供。8155 的 PA 口通过驱动电路(QD<sub>1</sub>)来提供 LED 显示器的位控信号,同时也驱动键盘。8155 的 PC<sub>0</sub>~PC<sub>3</sub> 用于拾取按键状态。

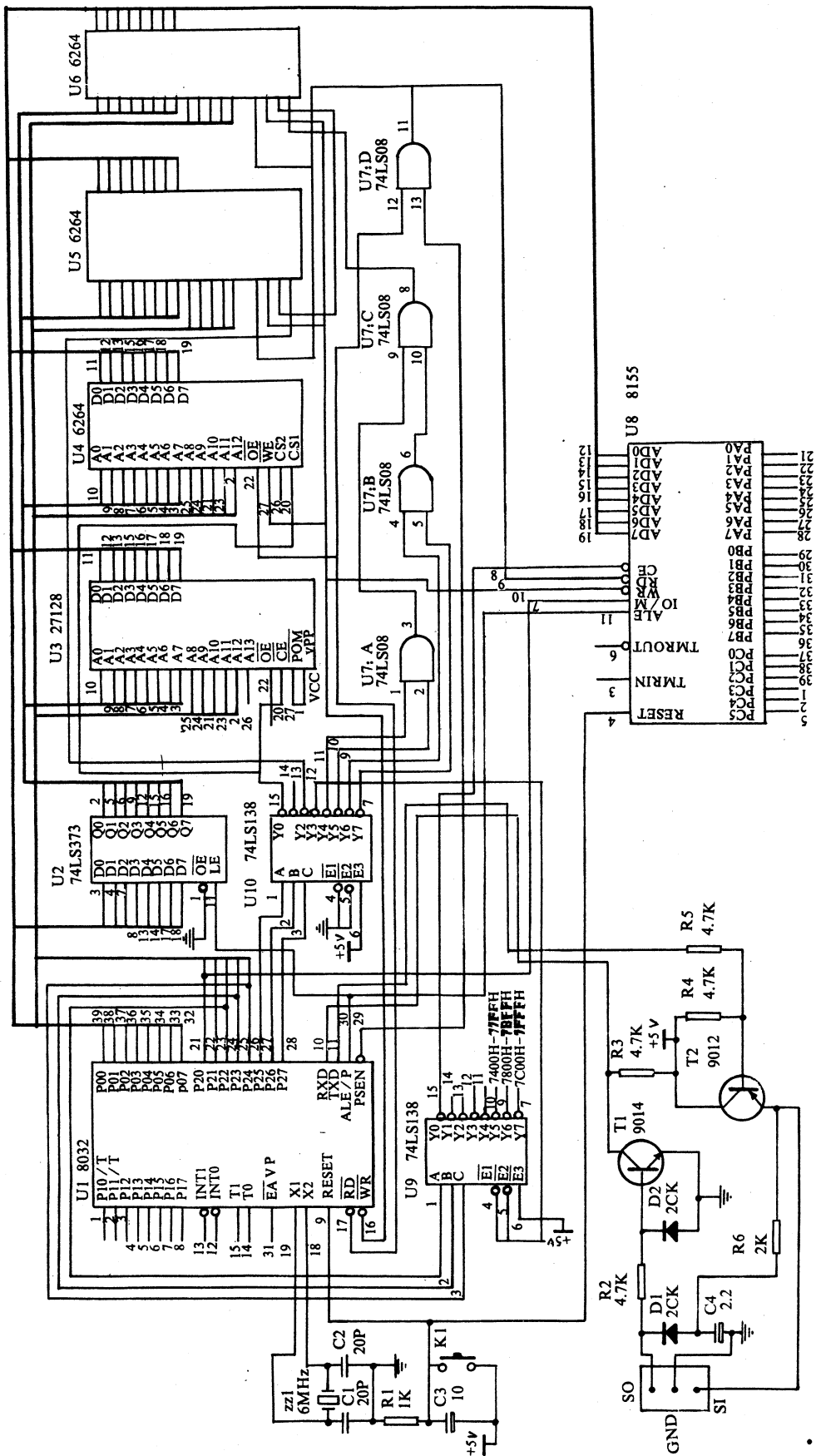


图 2 单片机扩展系统电原理图

打印接口用 PB<sub>0</sub>~PB<sub>7</sub> 作为数据线 D<sub>0</sub>~D<sub>7</sub>。P1.0 为打印机的启动信号 STB, PC<sub>5</sub> 用来读入打印机的 BUSY 信号。

EPROM 读写板在监控程序支持下对 2716、2732、2764、27128、27256、27512 各种型号 EPROM 进行写入、读出、比较、检查。

EPROM 操作时所有地址、数据信号均由 8031 数据总线通过 8255 锁存后提供,而操作 EPROM 所有的控制线由 8031 P0 口通过 74LS377 锁存后提供。由于 EPROM 采用 5.4V 供电以及配合监控程序中的智能编程方法,在正常情况下写入一片 2764 约为 20 秒左右,大大提高了 EPROM 写入速度。

在基本配置下,可以脱离 PC 机,在 LED 显示器及键盘支持下进行各项实验,此外还可以进行电子钟、显示器的动态扫描、键盘的扫描、译码、打印机驱动程序、EPROM 操作等实验。

使用显示键盘板进行调试的监控程序已固化在系统板上,其功能类似于 BJ-51 仿真器。

### 三、扩展配置

扩展配置是一种无穷尽的配置,在 BJS-51 系统设计开始时就确定为一种开放式的结构。即所有资源都向用户开放,以使用户也参与到 BJS-51 系统开发中来。

扩展板由硬件与软件二部分组成。硬件部分为以

某种接口芯片(例如 12 位 A/D 芯片 AD574)为核心,配以完善的单片机接口电路组成的电路板。该扩展板与系统板连接后可以完成该接口芯片的各种实验。软件部分包括该接口芯片的有关资料例如引脚排列、使用方法、应用电路、驱动程序等。用户可以通过对扩展板的实验,学会一种新的芯片的使用方法。

目前已设计有各种 A/D 芯片、显示键盘芯片、电机控制系统等扩展板。随着大量专用芯片(语言芯片、实时钟芯片等)的出现,扩展板的设计将具有广阔的前景。同时,教育实验系统也不仅用于大学本科生的单片机学习,还将成为广大工程技术人员进行学习研究的好帮手。

BJS-51 的软件包括监控程序 BJS-51-MON 及实验程序库,前者固化在一片 27128 上用于进行系统的编程与调试;后者存放在软磁盘上以便今后不断地丰富扩充。

BJS-51-MON 由二个独立的监控程序组成,它们各占 8KB。前 8KB 用于在 PC 机支持下进行编程与调试,后 8KB 用于在 LED 显示器及键盘支持下进行编程及调试。它们具有下列功能:

1. 状态显示功能——可以显示单片机内部 128B (或 256B)字节 RAM 的内容,以及所有特殊功能寄存器及二个 64K 的外部存储器空间的内容。

(下转 28 页)

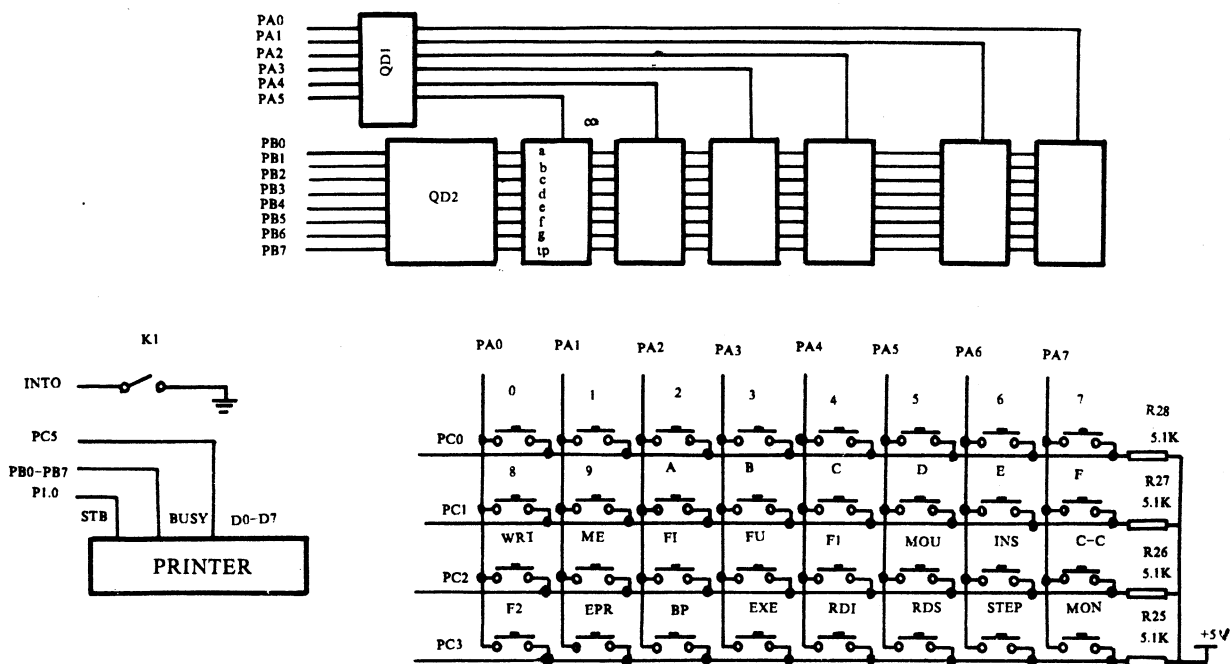
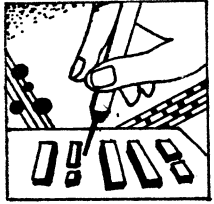


图 3 显示键盘板原理电路图





# 学装微电脑

## 步进电机的控制

易齐干

步进电机有输入脉冲与电机轴转角成比例的特征。微电脑控制步进电机最适宜。

本实验使用日本 PXB43—01A 型步进电机，步距角为 1.8°。规格如表 1 所示。

将外部负载转矩施加于电机轴上，对电机以额定电压激励，电机轴开始启动时的扭矩称为激励最大静转矩。

电机的外形尺寸与内部原理如图 1 所示。

表 1 PXB43—01A 规格

| 电压    | 电流      | 激励最大静转矩之后 | 线圈电阻   |
|-------|---------|-----------|--------|
| DC12V | 0.16A 相 | 600g·cm   | 770Ω/相 |

外线圈

电机内部接线图

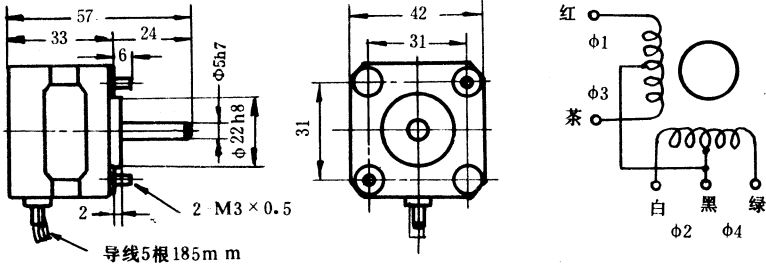


图 1 PXB43—01A 外形尺寸与内部原理

表 2.2 相激励通电次序

| 顺序 | 相 | φ1 | φ2 | φ3 | φ4 |
|----|---|----|----|----|----|
| 0  |   | 1  | 1  | 0  | 0  |
| 1  |   | 0  | 1  | 1  | 0  |
| 2  |   | 0  | 0  | 1  | 1  |
| 3  |   | 1  | 0  | 0  | 1  |

1: 附加 DC12V

0: 电波

反时针方向回转

顺时针方向回转

电机线圈由 4 相组成。即: φ1(红—黑)、φ2(白—黑)、φ3(棕—黑)、φ4(绿—黑)。驱动方式为 2 相激励方式。各线圈通电顺序如表 2。

表中首先向 φ1 线圈—φ2 线圈输入 DC12V 电压，接着 φ2—φ3、φ3—φ4、φ4—φ1，又返回到 φ1—φ2，按这种顺序切换，电机轴沿顺时针方向旋转。

步进电机的驱动电路如图 2 所示，对照表 2，微电脑向步进电机输入端传送 1 或 0 信息，则可实现上述动作。

为简化驱动电路，考虑步进电机线圈的电流为 0.16A，可选用三菱电机的 M54532P 驱动器 IC，再配备整流二极管。或者选用晶体管搭接成电路也可组成驱动电路，这种事例读者很容易查阅到。

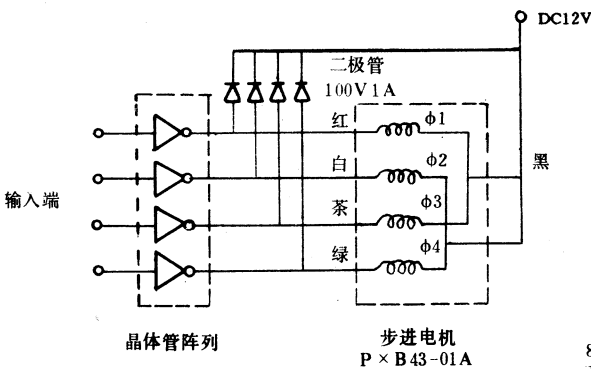


图 2 步进电机驱动电路

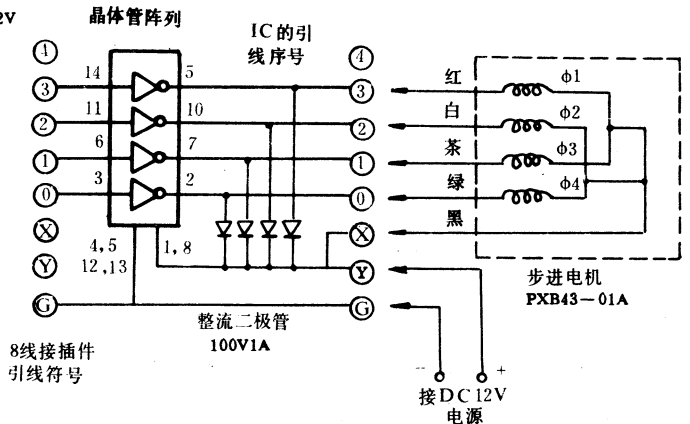


图 3 步进电机驱动电路

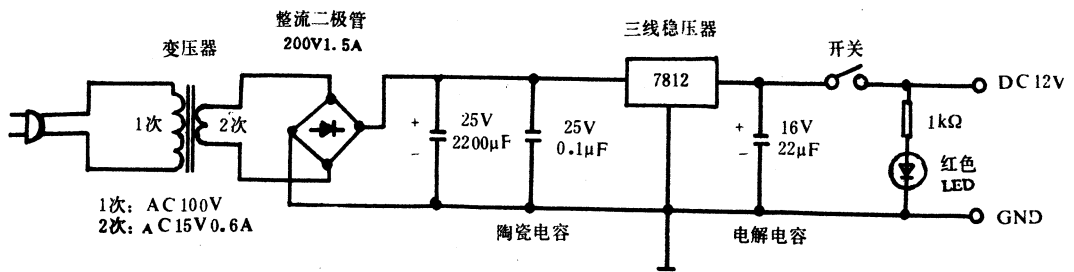


图 4 DC12V0.6A 电源原理图

为方便与  $\mu\text{P}-80$  微电脑连接,驱动电路部件原理图如图 3 所示。考虑各线圈切换电流时,要产生反向电压,为保护晶体管,电路中设置二极管。

步进电机额定电压为 DC12V,流过每相线圈的电流为 0.16A。2 相激励时,电流容量必须为  $0.16 \times 2 = 0.32\text{A}$ 。考虑安全系数,需要制作 DC12V、0.6A 的驱动电源,电源原理图与元件布置分别如图 4、图 5 所示。

步进电机、驱动电路和 DC12V 电源与  $\mu\text{P}-80$  套件件的连接如图 6 所示。输入输出部件端口 C 低位与驱动电路之间介入反相器部件,避免步进电机出现过热现象。端口 C 高位连接按键开关,开关  $S_0$  为 ON,步进电机沿时钟方向旋转;开关  $S_1$  为 ON,步进电机停止旋转。

给步进电机的脉冲数由 4 位数显示部件显示。

步进电机 2 相激励驱动,向各相施加脉冲照表 2 所示。应该将初始数据 33H(00110011)存入用 HL 寄存器对寻址的指定地址。按正转或反转的要求,以某一定时将 33H 执行 RRC(HL)或 RLC(HL)指令,同时,

向端口 C 低位输出即可满足上述要求。数据 33H 的移动变化如图 7 所示。

程序流程图和 RAM 区域分配如图 8、图 9 所示。

任何一项工作不管多么复杂,流程图的格式均可如图 8 所示。但是它可以包含若干个子程序,下面逐个讨论子程序。

设置初始数据(INIT)子程序如图 10 所示。

程序内容包括清除 00F1H 地址的开关标志;驱动数据 33H 放入 00F2H 地址;脉冲计数器清零。

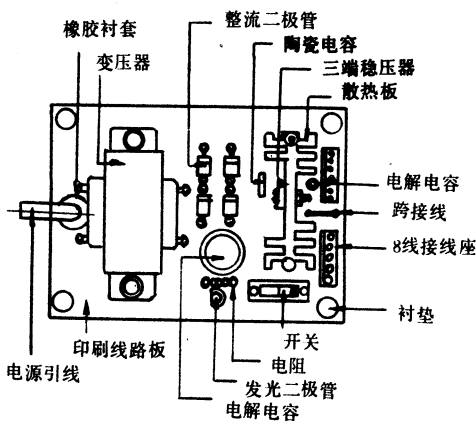


图 5 DC12V 电源元件布置图

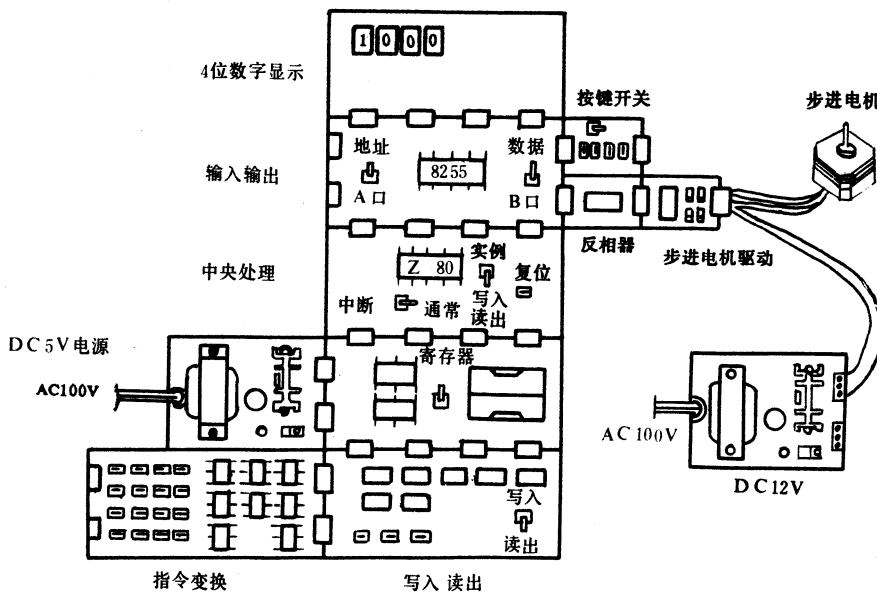


图 6 步进电机和  $\mu\text{P}-80$  套件连接

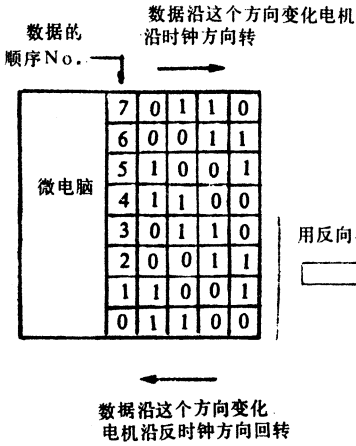


图7 微电脑输出数据与电机附加数据

数据的顺序No.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 1 | 0 | 0 | 1 |
| 2 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 |

... 向电机的φ1附加  
... 向电机的φ2附加  
... 向电机的φ3附加  
... 向电机的φ4附加

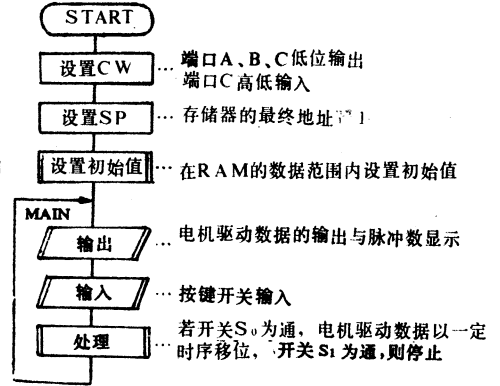


图8 简单流程图

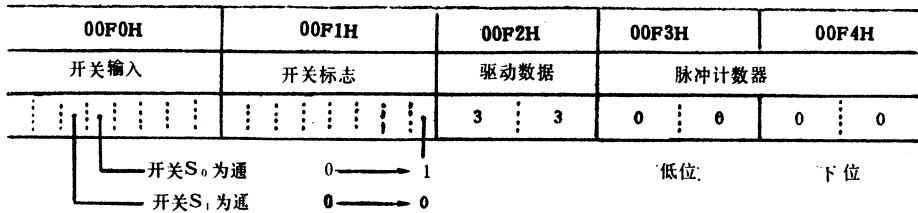


图9 RAM区域

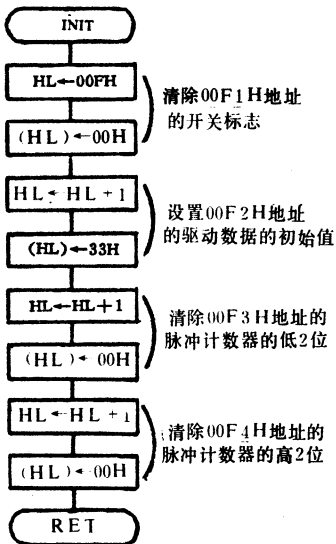


图10 设置初始数据子程序

输出(OUTPUT)子程序如图11所示。向端口C低位输出00F2H地址的驱动数据;向端口B、端口A输出00F3H和00F4H地址的脉冲计数器数值。

输入(INPUT)子程序如图12所示。按键开关通过端口C高位将开关状态放入00F0H地址。

处理(PROCES)子程序如图13所示。它由三个子程序组成。类似这种较大的处理内容再划分为多个小的处理内容,编程过程中思路清楚,减少失误,程序准确。

第一个处理开关标志(SWFLAG)子程序如图14所示。程序中相对于按键开关S<sub>0</sub>、S<sub>1</sub>为ON状态,00F1H地址第0位为1或为0。查询00F1H地址第0位,就可以判断是否按动过按键开关S<sub>0</sub>。

第二个处理驱动数据(DRDATA)子程序如图15所示。首先查询按键开关S<sub>0</sub>,如果按动过S<sub>0</sub>,以TIMER延时程序获得的时间处理驱动数据。

第三个处理脉冲计数器(COUNT)子程序如图16所示。COUNT1子程序是10进制计数加程序。

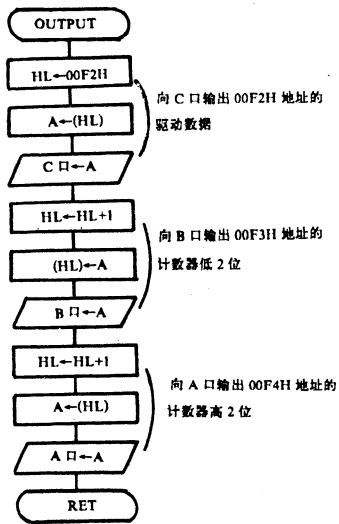


图 11 输出子程序

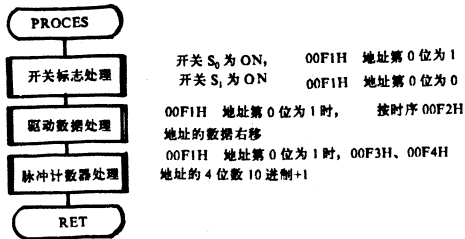


图 13 处理子程序

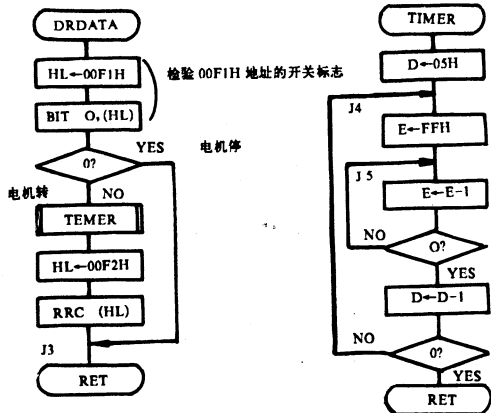


图 15 驱动数据处理子程序

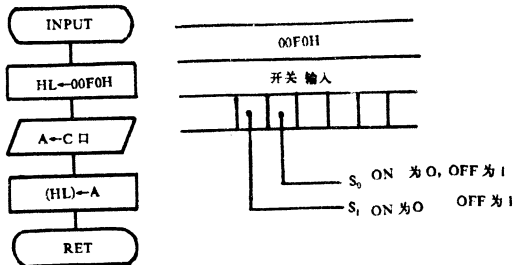


图 12 输入子程序

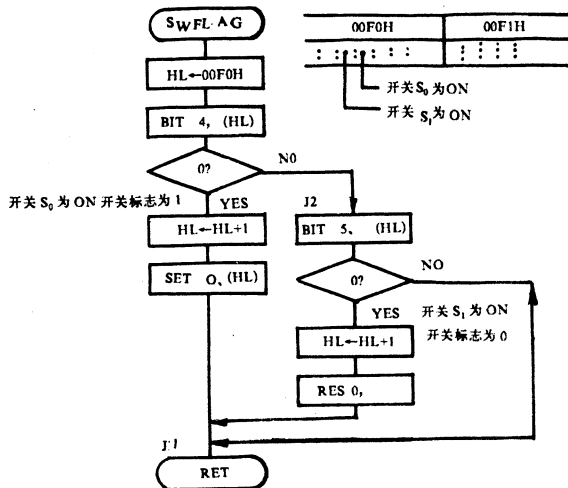


图 14 开关标志处理子程序

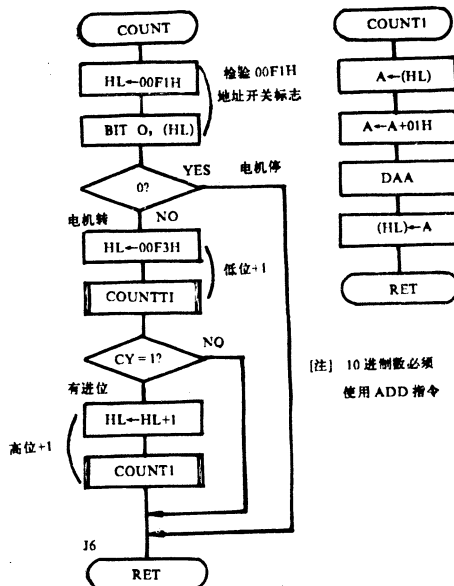


图 16 脉冲计数子程序

表3 程序清单(对照流程图编制出程序清单,如表3所示。)

| 标记     | 助记符         | 地址   | 机械语      |
|--------|-------------|------|----------|
| START  | LD A88H     | 0000 | 3E 88    |
|        | OUT (03H)A  | 0002 | D3 03    |
|        | LD SP,0100H | 0004 | 31 00 01 |
|        | CALL INIT   | 0007 | CD 20 00 |
| MAIN   | CALL OUTPUT | 000A | CD 30 00 |
|        | CALL INPUT  | 000D | CD 40 00 |
|        | CALL PROCES | 0010 | CD 50 00 |
|        | JP MAIN     | 0013 | C3 0A 00 |
|        |             |      |          |
| INIT   | LD HL,00F1H | 0020 | 21 F1 00 |
|        | LD HL,00H   | 0023 | 36 00    |
|        | INC HL      | 0025 | 23       |
|        | LD HL,33H   | 0026 | 36 33    |
|        | INC HL      | 0028 | 23       |
|        | LD HL,00H   | 0029 | 36 00    |
|        | INC HL      | 002B | 23       |
|        | LD HL,00H   | 002C | 36 00    |
| RET    | 002E        | C9   |          |
| OUTPUT | LD HL,00F2H | 0030 | 21 F2 00 |
|        | LD A,HL     | 0033 | 7E       |
|        | OUT (02H)A  | 0034 | D3 02    |
|        | INC HL      | 0036 | 23       |
|        | LD A,(HL)   | 0037 | 7E       |
|        | OUT (01H),A | 0038 | D3 01    |
|        | INC HL      | 003A | 23       |
|        | LD A,(HL)   | 003B | 7E       |
|        | OUT (00H),A | 003C | D3 00    |
|        | RET         | 003E | C9       |
| INPUT  | LD HL,00F0H | 0040 | 21 F0 00 |
|        | IN A,(02H)  | 0043 | DB 02    |
|        | LD (HL),A   | 0045 | 77       |
|        | RET         | 0046 | C9       |
| PROCES | CALL SWFLAC | 0050 | CD 60 00 |
|        | CALL DRDATA | 0053 | CD 80 00 |
|        | CALL COUNT  | 0056 | CD A0 00 |
|        | RET         | 0059 | C9       |
| SWFLAG | LD HL,00F0H | 0060 | 21 F0 00 |
|        | BIT 4,(HL)  | 0063 | CB 66    |

|        |             |      |          |
|--------|-------------|------|----------|
| J2     | JP NZ,12    | 0065 | C2 6C 00 |
|        | INC HL      | 0068 | 23       |
|        | SET 0,(HL)  | 0069 | CB C6    |
|        | RET         | 006B | C9       |
|        | BIT 5,(HL)  | 006C | CB 6E    |
|        | JP NZ,J1    | 006E | C2 6B 00 |
|        | INC HL      | 0071 | 23       |
| J3     | RES 0,(HL)  | 0072 | CB 86    |
|        | JP J1       | 0074 | C3 6B 00 |
| DRDATA | LD HL,00F1H | 0080 | 21 F1 00 |
|        | BIT 0,(HL)  | 0083 | CB 46    |
|        | JP Z,J3     | 0085 | CA 90 00 |
|        | CALL TIMER  | 0088 | CD 91 00 |
|        | LD HL,00F2H | 008B | 21 F2 00 |
|        | RRC (HL)    | 008E | CB 0E    |
|        | RET         | 0090 | C9       |
|        | LD D,05H    | 0091 | 16 05    |
|        | LD E,FFH    | 0093 | 1E FF    |
|        | DEC E       | 0095 | 1D       |
| TIMER  | JP NZ,J5    | 0096 | C2 95 00 |
|        | DEC D       | 0099 | 15       |
|        | JP NZ,J4    | 009A | C2 93 00 |
|        | RET         | 009D | C9       |
| COUNT  | LD HL,00F1H | 00A0 | 21 F1 00 |
|        | BIT 0,(HL)  | 00A3 | CB 46    |
|        | JP Z,J6     | 00A5 | CA B5 00 |
|        | LD HL,00F3H | 00A8 | 21 F3 00 |
|        | CALL COUNTI | 00AB | CD B6 00 |
|        | JP NC,J6    | 00AE | D2 B5 00 |
|        | INC HL      | 00B1 | 23       |
|        | CALL COUNTI | 00B2 | CD B6 00 |
|        | RET         | 00B5 | C9       |
|        | LD A,(HL)   | 00B6 | 7E       |
|        | ADD A,01H   | 00B7 | C6 01    |
|        | DAA         | 00B9 | 27       |
|        | LD (HL),A   | 00BA | 77       |
|        | RET         | 00BB | C9       |
|        | COOUNTI     |      |          |
|        |             |      |          |

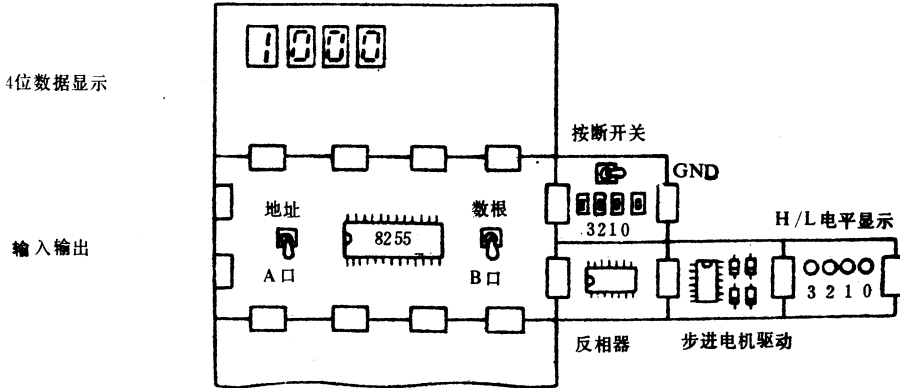


图17 检验程序的硬件

μP-80 套件执行程序前,应该用 H/L 电平显示部件代替步进电机。避免因写入错误而损坏步进电机。执行程序时,输入输出部件的切换开关搬至 A 口、B 口侧,4 位数显部件应该显示 0000。硬件连接图如图 17 所示。按动按键开关 S<sub>0</sub>,H/L 电平显示部件的 4 支 LED 应该反复闪亮,说明程序动作正确。

确认动作正确之后,将驱动部件、步进电机和 DC12V 电源代替 H/L 电平显示部件,重新执行程序。按键开关 S<sub>0</sub> 为 ON,步进电机沿时钟方向运转,4 位数显部件显示计数加。如果按动按键开关 S<sub>1</sub>,则步进电机停止旋转。



# ASCII 码字符显示器的设计

贵州风华电冰箱厂 李德文

数字显示器(又称为数码管)常用的有 7 段数字显示器和 8 段数字显示器两种。由于使用 7 段笔划便能拼成十个数字和少量英文字母,在电子手表、时钟、计算器及数字仪表中广为采用。

能否设计出一种由若干段笔划组成的字符显示器,用它可以显示 10 个数字和 26 个英文字母,乃至显示所有的 ASCII 码字符呢?

回答是肯定的。

为了演示这一结果,本文中附上了一个在 AST 上 GWBASIC 和 BASICA 下运行通过的演示程序。

标准 ASCII 字符集共有 128 个字符,其中 94 个为可显示字符。通过分析和抽象,笔者发现,使用图 1 所示的 20 段笔划,便能组成除“#”之外的 93 个 ASCII 码字符。

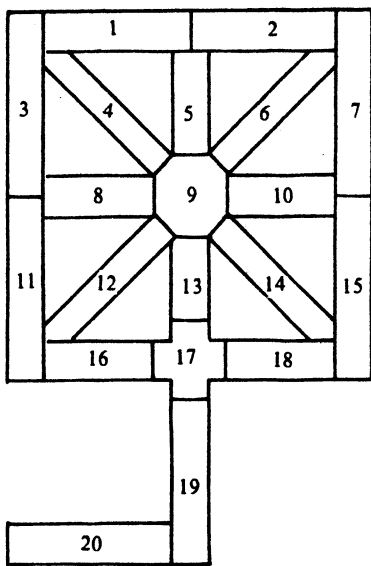


图 1 20 段字符显示器

如果把此 20 段笔划做成发光二极管封装于透明的环氧树脂中,如图 2 所示,便可以形成 20 段字符显示器。

然而,为了控制 20 段发光二极管的发光,至少需要 20 条引脚,这对生产厂家和用户都不一定方便。解决此问题,可以如图 3 所示,设计一个“ASCII 码-20 段字符显示码转换电路”接于字符显示器之前,并与它集成在一起。这样,控制 20 段发光二极管的发光,只需有 7 条引脚就足够了。

至于转换电路的设计,稍懂数字电路的人都容易实现。

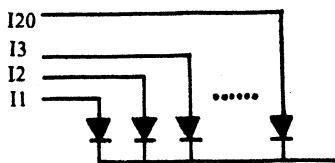


图 2 发光二极管 20 段字符显示器

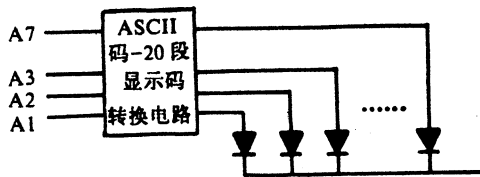


图 3 带转换电路的发光二极管 20 段字符显示器

根据图 1 中的编号,当为了显示一字符而点亮某段时,令此段编码为 1,否则,编码为 0。这样,每个字符的编码共有 20 位。按四位一组分开成五组,每组的四位二进制数组成一位十六进制数,按编号从小到大的次序排列,得到五位十六进制数,作为 20 段字符显示器显示编码。

表 1 为所有 ASCII 码可显示字符与编码对照表。其中,无法显示字符“#”的编码令为 FFFFF。

表 1 字符 ASCII 码 20 段数码显示器编码对照表

|                   |              |              |              |              |              |              |              |              |              |              |              |              |
|-------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 字符<br>ASCII<br>编码 | !            | "            | #            | \$           | %            | &            | .            | (            | )            | *            | +            | ,            |
|                   | 033<br>08808 | 034<br>0A000 | 035<br>FFFFF | 036<br>E9CBE | 037<br>35D60 | 038<br>988E4 | 039<br>04000 | 040<br>04840 | 041<br>10900 | 042<br>15D40 | 043<br>09C80 | 044<br>0000B |
| 字符<br>ASCII<br>编码 | -            | .            | /            | 0            | 1            | 2            | 3            | 4            | 5            | 6            | 7            | 8            |
|                   | 045<br>01C00 | 046<br>00800 | 047<br>04900 | 048<br>E2A3C | 049<br>08888 | 050<br>C2D1C | 051<br>C243C | 052<br>29C80 | 053<br>48C3C | 054<br>E1E3C | 055<br>C4880 | 056<br>D495C |
| 字符<br>ASCII<br>编码 | 9            | :            | ;            | <            | =            | >            | ?            | @            | A            | B            | C            | D            |
|                   | 057<br>E3C3C | 058<br>00808 | 059<br>0080B | 060<br>0491C | 061<br>01C1C | 062<br>1085C | 063<br>C2C88 | 064<br>C3ABC | 065<br>06D20 | 066<br>E5A5C | 067<br>E021C | 068<br>CA8BC |
| 字符<br>ASCII<br>编码 | E            | F            | G            | H            | I            | J            | K            | L            | M            | N            | O            | P            |
|                   | 069<br>E121C | 070<br>E1A00 | 071<br>E063C | 072<br>23E20 | 073<br>C889C | 074<br>C8890 | 075<br>0C8C0 | 076<br>2021C | 077<br>36A20 | 078<br>32A60 | 079<br>E223C | 080<br>E3E00 |
| 字符<br>ASCII<br>编码 | Q            | R            | S            | T            | U            | V            | W            | X            | Y            | Z            | [            | \            |
|                   | 081<br>E227C | 082<br>E3E40 | 083<br>E1C3C | 084<br>C8880 | 085<br>2223C | 086<br>24B00 | 087<br>22B60 | 088<br>14940 | 089<br>14880 | 090<br>C491C | 091<br>48884 | 092<br>10840 |
| 字符<br>ASCII<br>编码 | ]            | ^            | _            | `            | a            | b            | c            | d            | e            | f            | g            | h            |
|                   | 093<br>88890 | 094<br>00940 | 095<br>0001C | 096<br>10000 | 097<br>01A9C | 098<br>21298 | 099<br>01210 | 100<br>09A98 | 101<br>01310 | 102<br>49C8B | 103<br>01A9B | 104<br>21288 |
| 字符<br>ASCII<br>编码 | i            | j            | k            | l            | m            | n            | o            | p            | q            | r            | s            | t            |
|                   | 105<br>00888 | 106<br>00898 | 107<br>08CC0 | 108<br>0888C | 109<br>01EA0 | 110<br>01A88 | 111<br>01290 | 112<br>00CAE | 113<br>01A9A | 114<br>00C80 | 115<br>00444 | 116<br>09C8C |
| 字符<br>ASCII<br>编码 | u            | v            | w            | x            | y            | z            | {            |              | }            | ~            |              |              |
|                   | 117<br>00A98 | 118<br>00B00 | 119<br>00B60 | 120<br>01984 | 121<br>0028B | 122<br>01918 | 123<br>49884 | 124<br>08080 | 125<br>88C90 | 126<br>30860 |              |              |

当然,表 1 给出的编码仅仅是参考码而已,如果投入生产,生产厂家完全有理由从美观、工艺等角度对之进行修改。

以下是 20 段字符显示器显示 ASCII 码字符的演示程序:

LIST

10 SCREEN 2,1;CLS;GOTO 420

20 '子程序 1(画框架)

30 LINE(110,10)-(115,110),,B;LINE(110,60)-(115,60)

40 LINE(115,10)-(195,15),,B;LINE(155,10)-(155,15)

50 LINE(195,10)-(200,110),,B;LINE(195,60)-(200,60)

60 LINE(115,18)-(150,57);LINE-(152,55);LINE-(118,15)

70 LINE(192,15)-(157,55);LINE-(160,57);LINE-(195,18)

80 LINE(115,57)-(150,63),,B;LINE(160,57)-(195,63),,B

90 LINE(152,15)-(157,55),,B;LINE(152,65)-(157,103),,B

100 LINE(152,113)-(157,160),,B;LINE(110,155)-(152,160),,B

110 LINE(115,102)-(150,63);LINE-(152,65);LINE-(118,105)

120 LINE(192,105)-(157,65);LINE-(160,63);LINE-(195,102)

130 LINE(115,105)-(150,110),,B;LINE(160,105)-

(195,110),,B

140 LINE(150,105)-(152,105);LINE-(152,103)

150 LINE(157,103)-(157,105);LINE-(160,105)

160 LINE(160,110)-(157,110);LINE-(157,113)

170 LINE(150,110)-(152,110);LINE-(152,113)

180 RETURN

190 '子程序 2(点亮)

200 ON I GOTO 210,220,230,240,250,260,270,280,290,300,310,320,330,340,350,360,370,380,390,400

210 PAINT(120,13),,2;GOTO 410

220 PAINT(180,13),,2;GOTO 410

230 PAINT(113,20),,2;GOTO 410

240 PAINT(117,17),,2;GOTO 410

250 PAINT(155,20),,2;GOTO 410

260 PAINT(193,17),,2;GOTO 410

270 PAINT(196,20),,2;GOTO 410

280 PAINT(120,60),,2;GOTO 410

290 PAINT(155,60),,2;GOTO 410

300 PAINT(180,60),,2;GOTO 410

310 PAINT(113,80),,2;GOTO 410

320 PAINT(117,104),,2;GOTO 410

330 PAINT(155,80),,2;GOTO 410

340 PAINT(193,104),,2;GOTO 410

350 PAINT(196,80),,2;GOTO 410

360 PAINT(120,108),,2;GOTO 410

370 PAINT(155,108),,2;GOTO 410

380 PAINT(180,108),,2;GOTO 410

390 PAINT(155,120),,2;GOTO 410

400 PAINT(120,158),,2;GOTO 410

410 RETURN

```

420 '主程序(显示字符)
430 LOCATE 3,50:INPUT"请输入一个 ASCII 码字符:";
 CH$:CLS
440 IF LEN(CH$)<>1 THEN END
450 LOCATE 5,50:PRINT"字符:";CH$
460 LOCATE 7,50:PRINT"ASCII 码:";ASC(CH$)
470 READ CH1$;IF CH1$<>CH$ GOTO 470
480 GOSUB 20
490 READ CH2$;CH5$=""
500 FOR I=1 TO 5
510 CH3$=MID$(CH2$,I,1)
520 IF CH3$="0" THEN CH4$="0000":GOTO 690
530 IF CH3$="A" THEN CH4$="1010":GOTO 690
540 IF CH3$="B" THEN CH4$="1011":GOTO 690
550 IF CH3$="C" THEN CH4$="1100":GOTO 690
560 IF CH3$="D" THEN CH4$="1101":GOTO 690
570 IF CH3$="E" THEN CH4$="1110":GOTO 690
580 IF CH3$="F" THEN CH4$="1111":GOTO 690
590 ON VAL(CH3$) GOTO 600,610,620,630,640,
 650,660,670,680
600 CH4$="0001":GOTO 690
610 CH4$="0010":GOTO 690
620 CH4$="0011":GOTO 690
630 CH4$="0100":GOTO 690
640 CH4$="0101":GOTO 690
650 CH4$="0110":GOTO 690
660 CH4$="0111":GOTO 690
670 CH4$="1000":GOTO 690
680 CH4$="1001"
690 CH5$=CH5$+CH4$
700 NEXT I
710 FOR J=1 TO 20
720 I$=MID$(CH5$,J,1)
730 IF I$="1" THEN I=J:GOSUB 190
740 NEXT J
750 RESTORE:GOTO 430
760 '主程序(编码)
770 DATA !, 08808, " ", 0A000, #, FFFFF, $,
E9CBE, %, 35D60, &, 988E4, " ", 04000, (, 04840
780 DATA), 10900, *, 15D40, +, 09C80, ", ", 0000B,
-, 01C00, ., 00800, /, 04900, 0, E2A3C
790 DATA 1, 08888, 2, C2D1C, 3, C243C, 4, 29C80, 5,
48C3C, 6, E1E3C, 7, C4880, 8, D495C
800 DATA 9, E3C3C, ":", 00808, ;, 0080B, <, 0491C,
=, 01C1C, >, 1085C, "?", C2C88
810 DATA @, C3ABC, A, 06D20, B, E5A5C, C, E021C, D,
CA8BC, E, E121C, F, E1A00, G, E063C
820 DATA H, 23E20, I, C889C, J, C8890, K, 0C8C0, L,
2021C, M, 36A20, N, 32A60, O, E223C
830 DATA P, E3E00, Q, E227C, R, E3E40, S, E1C3C, T,
C8880, U, 2223C, V, 24B00, W, 22B60
840 DATA X, 14940, Y, 14880, Z, C491C, [, 48884, \,
10840,], 88890, ^, 00940, _ , 0001C
850 DATA ', 10000, a, 01A9C, b, 21298, c, 01210, d,
09A98, e, 01310, f, 49C8B, g, 01A9B

```

```

860 DATA h, 21288, i, 00888, j, 00898, k, 08CC0, l,
0888C, m, 01EA0, n, 01A88, o, 01290

```

```

870 DATA p, 00CAE, q, 01A9A, r, 00C80, s, 00444, t,
09C8C, u, 00A98, v, 00B00, w, 00B60

```

```

880 DATA x, 01984, y, 0029B, z, 01918, {, 49884, |,
08080, }, 88C90, ~, 30860

```

20~180号语句为第一个子程序,用于画图1所示的图案。

190~410号语句为第二子程序,用于根据编码使相应的笔划变亮。

420~750号语句为主程序的控制部分,用于提示用户输入待显示的字符,并调用子程序一和子程序二演示20段字符显示器的显示结果。

760~880号语句为主程序的数据部分,根据表1给出所有ASCII码可显示字符及其在20段字符显示器上的编码。

本程序在运行时,请注意:

(1)由于BASIC语言的原因,“”字符不能显示,但读者不难根据编码想象出其形状;“,”为BASIC分隔符,在欲输入“,”时需输入“,”。

(2)若不输或输入两个以上字符后回车,程序便告退出。

根据表2所示程序的演示结果得知:编号为19和20的两段笔划,只有在显示,.,:;f.g.p.q.y等字符时才会用到;编号为9和17的笔划单独使用,也只有用来显示!.\.:;:;|等字符。如果对这些字符的兴趣不大,例如,只关心大写英文字母等。可以把20段字符显示器简化成图4所示的16段字符显示器。读者可以参照本文前面的内容给出16段字符显示器上ASCII码可显示字符的编码,并编写相应的演示程序。

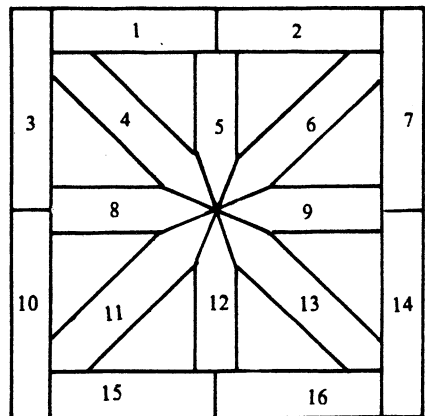


图4 16段字符显示器

诚然,有的字符显示不够美观,但分辨起来还是较容易的。

使用一排足够长的20段字符显示器,可以组成任意的英文句子,而且还能随时变换,在车站、码头、街头广告牌等处有着广泛的应用前景。在工业控制专用机显示屏等电子显示设备上,也能发挥其特殊效用。





## 电脑游戏机

# 第四章 F BASIC 语言的深入理解

山东苍山县机械电子化学工业局(277700) 于 春

### 六、音乐控制语句

科特 FCS-90 电脑学习机可以在 BS 状态通过人机对话进入 MUSIC BOARD(谱曲与演奏)状态,进行谱曲练习和演奏。由于 MUSIC BOARD 操作简单、输入方便、音色优美,深受广大用户的喜爱。但是,谱曲演奏只有一页的版面供使用。而且这一页仅有四行,一次最多可演奏 96 个音符(重奏除外);只有一种节拍,音长的允许变化范围也太窄(一般两个音长,要使用三个或四个音长,那么输入的音符要减少 3~4 倍)。这对于气势较大,有节拍变化的乐曲简直不能演奏。更有甚者每次只能演奏一小段乐曲,一首乐曲要断续几次甚至十几次才能演奏完毕,因而破坏了乐曲的完整性,影响了欣赏效果。可以说, MUSIC BOARD 仅是儿童游戏。欲真正地欣赏电脑演奏乐曲,还必须使用音乐控制语句,编制音乐演奏程序才能实现。

#### 1. 发音语句(BEEP)

BEEP 简写 B.

BEEP 语句的功能是使扬声器发出 1000Hz 的蜂鸣声。

例 21

```
10 F. I=0 TO 10
20 BEEP;PAU. 20
30 N.
RUN
```

扬声器发出断续的蜂鸣声。

#### 2. 演奏语句 (PLAY)

PLAY 简写 PL.

PLAY 语句的功能是谱曲和演奏乐曲。语句格式为

PL. “弦乐资料”

式中的弦乐资料是指按 F BASIC 的约定,编制的乐曲演奏字符串。

随机手册中对 PLAY 语句中的弦乐资料介绍的较笼统,读者虽能仿效演奏几句乐谱,但很难设计出一个完整的演奏程序。只有了解弦乐资料的语法结构,掌握编程的方法步骤,明确各项编程的约定后,才能编制出合法的程序,得心应手地谱写乐曲。

#### 〈一〉弦乐资料的约定

##### A. 节拍:

节拍以 T1~T8 指定

T1(快)↔T8(慢)

##### B. 音色:

音色以 Y0~Y3 指定

Y0: 12.5%

Y1: 25%

Y2: 50%

Y3: 75%

#### C. 主音:

主音由 M0~M1 指定。

M0 表示音量。以 V0~V15 指定音量的大小。V0(最小)——V15(最大)。

M1 表示音长。以 V0~V15 指定发音的长短。V0(最短)——V15(最长)。

#### D. 8 度音

8 度音以字母 O 指定, O0~O5。

以简谱中 1 音为例,对照如下:

O0 O1 O2 O3 O4 O5

1 1 1 1 . :

若乐曲中有 1̇ 音,可将 8 度音升一级 O5 为 1̇,其余类推。

若乐曲中有 1̇ 音,可将 8 度音升一级 O5 为 1̇,其余类推。

#### E. 音名和音符

音名分别用 C、D、E、F、G、A、B 表示。

音符分别用 1, 2, 3, 4, 5, 6, 7 表示。

音名和音符的对应关系如下:

|    |   |   |   |   |   |   |   |
|----|---|---|---|---|---|---|---|
| 全音 | C | D | E | F | G | A | B |
|    | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

|    |                |                |                |                |                |
|----|----------------|----------------|----------------|----------------|----------------|
| 半音 | #C             | #D             | #F             | #G             | #A             |
|    | 1 <sup>#</sup> | 2 <sup>#</sup> | 4 <sup>#</sup> | 5 <sup>#</sup> | 6 <sup>#</sup> |

1<sup>#</sup> 2<sup>#</sup> 4<sup>#</sup> 5<sup>#</sup> 6<sup>#</sup>

读者要记熟每个音符所对应的音名。

每个音名对应一个固有的频率。为便于查找和使用,下面列表给出各音名所对应的频率近似值供参考。

表 2 音名频率对照表

| 音名   | 频率  | 音名   | 频率  | 音名   | 频率   | 音名   | 频率   |
|------|-----|------|-----|------|------|------|------|
| O0 C | 56  | O1 G | 172 | O3 D | 500  | A    | 1500 |
| D    | 64  | A    | 190 | E    | 563  | B    | 1700 |
| E    | 72  | B    | 215 | F    | 600  | O5 C | 1800 |
| F    | 75  | O2 C | 220 | G    | 640  | D    | 2050 |
| G    | 84  | D    | 247 | A    | 760  | E    | 2500 |
| A    | 95  | E    | 280 | B    | 860  | F    | 2400 |
| B    | 107 | F    | 300 | O4 C | 920  | G    | 2700 |
| O1 C | 110 | G    | 340 | D    | 990  | A    | 3100 |
| D    | 120 | A    | 380 | E    | 1120 | B    | 3450 |
| E    | 140 | B    | 430 | F    | 1180 |      |      |
| F    | 150 | O3 C | 445 | G    | 1340 |      |      |

### F. 音符的时值

在音名后面附上 0~9 的整数用来指定音名的时值。时值与整数的对应关系见下表。

表 3 音名时值对照表

| 数字 | 音符     | 拍节      | 示例                     |
|----|--------|---------|------------------------|
| 9  | 全音符    | 4 拍     | 1---                   |
| 8  | 付二分音符  | 3 拍     | 1--                    |
| 7  | 二分音符   | 2 拍     | 1-                     |
| 6  | 付四分音符  | 1 1/2 拍 | 1.                     |
| 5  | 四分音符   | 1 拍     | 1                      |
| 4  | 付八分音符  | 3/4 拍   | <u>1</u> .             |
| 3  | 八分音符   | 1/2 拍   | <u>1</u>               |
| 2  | 付十六分音符 | 3/8 拍   | <u><u>1</u></u> .      |
| 1  | 十六分音符  | 1/4 拍   | <u><u>1</u></u>        |
| 0  | 三十二分音符 | 1/8 拍   | <u><u><u>1</u></u></u> |

### G. 休止符

休止以 R 表示。用 R0~R9 表示休止的长短。具体时间见 F 的约定。

### H. 重音

演奏重音时可在字符间加冒号实现。可演奏 2 重音、3 重音。最多为三重奏。演奏重音的格式为：

PL. “系统 A:系统 B”

PL. “系统 A:系统 B:系统 C”

### (二) 语法结构和编程约定

- 首先指定主音 M、音长 V、节拍 T 和音色 Y。一经指定后，各参数值会一直保持到再次指定为止。未指定时的初值为 M0V15T4O3，且音符的时值为 5。
- 每行以 PLAY 语句引导，字符串用双引号括起，每行引号内的字符不能超过 31 个。在二、三重奏时，冒号也计算在内。
- 在定义二、三重奏时，必须先定义节拍 T，几个系统中的节拍必须一致，否则，演奏不能同步。
- 音符的时值一经指定后，若后一个音符的时值与前一个一样，可省略时值，直到改变时值时再指定，即使换行也不影响。
- 每句中的 8 度音只要后一个与前一个相同，可不必再指定，直到重新指定 8 度音为止。即使换行也不影响。
- 在三重奏时，系统 C 的音色是固定的，因此系统 C 可省略音色的指定。

### (三) 编程方法：

- 把乐谱以拍分段，分段距离稍大些。一般十六开纸竖用，每行分八段。若是重奏，也对应分好。
- 在每个乐谱下面填上相应的音名，再根据每个

音名的时值对照时值表填入相应的整数。若与前一音名的时值相同，可以省略。

- 在音名前填上 8 度音 On，若与前一音名相同可以省略。当编程熟练后，b、c 两步可以一次填满。
- 将字符按每行少于或等于 31 个的规则填入 PLAY 语句。注意分行时，一拍内的音名不能分开。
- 输入计算机，进行演奏，修改错误。满意后，可写入录音磁带，以备调用。

例 21 编制《东方红》乐曲演奏程序。

首先分段写出乐谱，填上音名等。

填好核对无误后，填入 PLAY 语句。

本例设定 M1V15Y2T4。程序如下：

```
10 PL. "M1V15Y2T4"
20 PL. "O3G5G3AD7C5C3O2AO3D7"
30 PL. "G5GA3O4CO3AGC5C3O2AO3D7"
40 PL. "G5DCO2B3AG5O3GDE3DC5C3O2A"
50 PL. "O3DEDCDCO2BAG7"
60 E.
```

RUN 即可演奏出东方红乐曲。

下面再举一个三重奏的示例。

例 22 三重奏乐曲《铃儿响叮当》

《铃儿响叮当三重奏曲》是我根据彼尔彭特的《铃儿响叮当》原曲改编的。

首先分段列出三重奏曲谱，然后依次填入音名、时值、8 度音等。

《铃儿响叮当三重奏曲》

(略)

填完核对无误后，便可填入 PLAY 语句中。

由于是三重奏，必须保证每行 PLAY 语句中三个系统的节拍一致。为了保证每行不超过 31 个字符，可按节拍逐拍计算三个系统的字符数，以小于或等于 29 个字符为一句。但要注意，系统 A 中有 11 处 2 分音符或付 4 分音符，这就要求必须把该音符所在的节拍及后一节拍填在一行，否则将出现三个系统节拍紊乱。可先将字符分段，然后分别填入语句中。程序如下：

```
10 REM "LingErXiangDingDangSanChongZou"
20 PL. "M1V12Y0T3;M0V9M1V9Y2T3;M1T3"
30 PL. "O4E3EE5E3E;O3C5EG;O2C5G3O1GO2C5"
40 PL. "E5E3GCDE7;ECEGE;G3O1GO2C5GC3CDE"
50 PL. "F3FFFFEEEDDC;CFGEDA;F5ACGCG"
60 PL. "DG6E3EE5E3E;GBCEG;DGCG3O1GO2C5"
70 PL. "E5E3GCDE7;ECEGE;G3O1GO2C5GC3CDE"
80 PL. "F3FFFFEEEG;CFGED;F5ACGD3F"
90 PL. "FDC6O3G3;FCC;O1GO2CO1GABO2C"
100 PL. "GO4EDCO3G6G1G;C5EGE;C5GC3O1GGG"
110 PL. "G3O4EDCO3A6A3;CEFD;O2C5GF3O1AAA"
120 PL. "AO4FED;DA;O2F5A"
130 PL. "O3A6A3;FD;F3O1AO2AO1A"
140 PL. "O4GGFD;ED;O2GO1GAB"
150 PL. "E6O3G3GO4E;ECC;O2CO1GGO2CC5"
160 PL. "DCO3G6G3GO4E;EGEC;GC3O1GGGO2C5"
```

170 PL. "DC03A6A3,EFD,GF301AAA"  
 180 PL. "AO4FEDGGGG,AFBG,O2F5AGB"  
 190 PL. "AGFDC7,FDC7,G301GABO2CO1GAB"  
 200 END[或 GOTO20]

核对无误后,输入计算机,就可以欣赏这首世界名曲了。

读者可以发现,若编程熟练后可直接对照乐谱输入程序。那么输入速度并不比 MUSIC BOARD 慢。

### 七、F BASIC 语言的深入探讨

到此为止,本文共介绍了 90 条语句,其中有 17 条随机手册中没有介绍,是由我补充的。但是 F BASIC 语言并不仅有这 90 条语句。目前,我已发现了 26 条语句,除了以上我已弄清功能的 17 条外,还有 9 条目前尚不能确定。在此介绍给大家,愿与有兴趣的读者共同探讨。兹介绍如下:

#### 1. SCREEN

SCREEN 简写 SC.

SCREEN 语句的功能是设置屏幕的显示模式。语句格式为

SC. n1,n2

式中 n1,n2 取值为 0,1。

#### 2. FILTER

FILTER 简写 FIL.

语句格式为

FIL. n 式中 n 取值 0~7

#### 3. BGGET

BGGET 简写 BGG.

语句格式

BGG.

#### 4. BGPUP

BGPUP 简写 BGP.

语句格式

BGP.

BGET 和 BGPUP 好像是成对语句。BGGET 在前, BGPUP 在后。

#### 5. BACKUP

BACKUP 简写 BA.

语句格式为

BA.

回车后,屏幕上出现 SYSTEM ON,功能同 SYSTEM 语句在直调 BASIC 中差不多。

#### 6. CLICK

CLICK 简写 CLI.

#### 7. TAB

TAB 简写 TA.

TAB 语句在一般 BASIC 中功能是按列打印输出项,但在 F BASIC 中功能不同。

#### 8. CALL

CALL 简写 CA.

语句格式

CALL-n

式中 n 取值分别为 151,198,868 时,清屏,显示内存为 8182 BYTES FREE;当 n 取值为 958 时,清屏,显示内存为 32246 BYTES FREE;当 n 取值为 802 时,自锁,重新直调 BASIC 后显示 36342 BYTES FREE;当 n 取值为 1913 时,自锁,重新直调 BASIC 后显示 48630 BYTES FREE。好象使用 CALL 语句可以扩展内存。

#### 9. SPC

SPC 语句的功能与一般 BASIC 也不相同。

BASIC 语言是一个大家族,如 IBMPC 机的 BASIC 语句就有 170 多条。F BASIC 语言是 BASIC 家族中的一个新成员,大家还不太熟悉,因此它有待我们去深入发掘,以发挥 F BASIC 语言应有的作用,使它在微机普及领域中开出灿烂的花朵。

为便于读者研究探讨,文末列出 F BASIC 语言的保留字表,共计 99 个。使用这些保留字可以组成各种不同功能的语句。如用 TR 可以组成跟踪语句 TRON、TROFF;用 DEF 可以组成 DEF MOVE、DEF SPRITE;用 ON 可组成 ON GOTO、SPRITE ON、ON ERROR GOTO 等。

我愿同广大键盘游戏机用户结为挚友,以互相学习交流、共同提高。

附 F BASIC 语言 99 个保留字表:

F BASIC 语言保留字

|          |          |         |        |          |
|----------|----------|---------|--------|----------|
| GOTO     | GOSUB    | RUN     | RETURN | RESTORE  |
| THEN     | LIST     | SYSTEM  | TO     | STEP     |
| SPRITE   | PRINT    | FOR     | NEXT   | PAUSE    |
| INPUT    | LINPUT   | DATA    | IF     | READ     |
| DIM      | REM      | STOP    | CONT   | CLS      |
| CLEAR    | ON       | OFF     | CUT    | NEW      |
| POKE     | CGSET    | VIEW    | MOVE   | END      |
| PLAY     | BEEP     | LOAD    | SAVE   | POSITION |
| KEY      | COLOR    | DEF     | CGEN   | SWAP     |
| CALL     | LOCATE   | PALET   | ERA    | TR       |
| FIND     | GAME     | BGTOOL  | AUTO   | DELETE   |
| RENUM    | FILTER   | CLICK   | SCREEN | BACKUP   |
| ERROR    | RESUME   | BGPUP   | BGGET  | CAN      |
| XOR      | OR       | AND     | NOT    | MOD      |
| ABS      | ASC      | STR \$  | FRE    | LEN      |
| PEEK     | RND      | SGN     | SPC    | TAB      |
| MID \$   | STICK    | STRIG   | XPOS   | YPOS     |
| VAL      | POS      | CSRLIN  | CHR \$ | HEX \$   |
| INKEY \$ | RIGHT \$ | LEFT \$ | SCR \$ | INSTR    |
| CRASH    | ERR      | ERL     | VCT    |          |

常用错误代码表

| 错误代码 | 错误表示 | 错误信息                   | 说明                           |
|------|------|------------------------|------------------------------|
| 0    | NF   | NEXT without FOR       | 无 FOR 却有 NEXT                |
| 1    | SN   | syntax error           | 语法错误                         |
| 2    | RG   | RETURN without GOSUB   | 无 GOSUB 却有 RETURN            |
| 3    | OD   | Out of DATA            | READ 读的数据未备于 DATA 中          |
| 4    | IL   | Illegal function call  | 功能调用不合法                      |
| 5    | OV   | Overflow               | 演算结果超限(溢出)                   |
| 6    | OM   | Out of memory          | 内存不足                         |
| 7    | UL   | Undefined line Number  | GOTO、GOSUB、IF …… THEN 的行号未定义 |
| 8    | SO   | Subscript out of range | 数组变量的下标在规定的范围之外              |
| 9    | DD   | Duplicate Definition   | 数组变量定义重复                     |
| 10   | DZ   | Division by Zero       | 0 做除数                        |
| 11   | TM   | Type Mismatch          | 变量的类型不一致                     |
| 12   | ST   | String too long        | 字符串超过 31 个                   |
| 13   | FT   | Formula Too complex    | 式子过于复杂                       |
| 14   | CC   | Can't Continue         | CONT 无法继续执行程序                |
| 15   | UF   |                        |                              |
| 16   | MO   | Missing Operand        | 缺少操作数                        |
| 17   | TP   | Tape Read Error        | 无法从卡带正确地读资料                  |
| 18   | NR   | No RESUME              | 错误处理程序无 RESUME               |
| 19   | RE   | RESUME Without ERROR   | 无错误处理程序有 RESUME              |
| 20   | NB   | No BGGET               | 无 BGGET 却有 BGPUT             |
| 21   | UP   | Unprintable Error      | 无法显示打印的错误代码                  |

(上接 48 页)

720KB 双面盘片, 1.44MB 双面盘片。

### 23. 怎样检测维护硬盘驱动器?

在 PC 机系统中, 硬盘存储设备主要包括硬盘控制器和驱动器两部分。硬盘控制器用以完成硬盘驱动器和主机之间信息传输; 硬盘驱动器主要包括机械结构和电子线路两部分。驱动器机械结构比较复杂, 它安装在主机箱中的一个密封腔体中, 对于用户绝对不能去打开它, 即使出现硬故障, 也只能送到专门生产厂家去维修。

这里介绍三种检测硬盘驱动器的方法:

#### (1) 利用系统自检

PC 机对硬盘驱动器具有较强的自测试能力。它是整个微机系统诊断的一部分, 这一自诊断程序被固化在主机板的一片 ROM 中, 当主机加电时, 这个诊断程序从 ROM 中调出, 对微机系统的各个部分进行自动检测, 当对硬盘进行检测时, 一旦发现故障即在屏幕上

显示“1701”故障代码, 告诉用户硬盘驱动器自检未通过, 应停机进行检查。在诊断硬盘过程中, 若出现下列故障之一:

- 控制器复位 37 次都不成功
- 控制器内部诊断有故障
- 控制器 RAM 诊断有故障
- 驱动器在 25 秒内未准备好
- 驱动器重新校准不成功
- 设置驱动器参数不成功
- 写磁盘扇区有故障

都会在屏幕上出现“1701”出错代码

#### (2) 利用文件检查

可以通过有关的系统文件, 直接对硬盘驱动器读/写文件的能力, 存储数据的容量以及格式化的能力进行检测。这种检测要破坏盘中原有数据和文件, 为此, 在操作之前应把硬盘上的文件进行备份, 以免丢失必要的文件。

#### (3) 利用高级诊断程序检测

PC 机随机资料中都提供了诊断盘, 诊断程序以菜单形式, 便于用户操作。

选择“0”便进入诊断过程。在这过程中主要完成: 执行读测试, 写测试, 零磁道测试, 表面扫描测试以及硬盘磁头定位操作。在上述的各项诊断中, 如一项不通过, 都将显示出故障代码, 可以通过故障代码便知道故障原因所在。

但需说明的是用诊断盘对硬盘驱动器进行检查后, 只是在某些质量和指标上得到确认, 而对硬盘驱动器是否能在实际工作中运行, 以及盘上具体有多少空间能够写进数据, 多少扇区已被破坏等等, 这些参数只能在操作系统下用文件检查进一步确定。

硬盘驱动器在使用中应从以下几方面注意维护:

(1) 安装时要认真阅读有关用户安装使用手册一类的随机资料。仔细正确地连接好信号电缆线。

(2) 万不得已需从机器上取下硬盘驱动器时, 也不要拆开盘体外壳螺钉, 即不要打开硬盘, 否则可能因此而报废。

(3) 硬盘驱动器一般在主机箱中, 主机箱应尽量放置平稳。否则当机器进行读/写操作时, 一旦振动, 容易出现磁头损坏盘片数据区, 造成盘内某些文件读不出来。40MB 以下容量的硬盘, 当工作完毕后, 磁头不能自动退到盘区“复位区”, 必须用 DOS 专用的固定磁头的命令 PARK.COM 或 SHIPDISK.COM, 或用诊断程序中固定硬盘磁头的功能选择。所以作为用户, 在关机之前要固定硬盘磁头。

(4) 要保持所在环境的清洁, 因为硬盘驱动器内有一个重要的机构, 包括一个位于盘体外壳上的起呼吸作用的小窗口, 如果环境空气中的灰尘太多, 就有可能使灰尘堵死这个小窗口, 那么空气过滤系统将无法正常工作, 从而导致硬盘驱动器的损坏。

(5) 要避免环境的高温和潮湿, 也要防强磁场的干扰。



# Super AT 机显示器电源原理及维修

镇江市科技情报所(212001) 范志盛

## 一、工作原理

Super AT 机显示器采用的是开关电源,因此它具有效率高、输出稳定可靠、负载变化范围宽等优点,也有一般开关电源结构复杂、非线性的缺点。图1是该电源的线路图。图及器件参数是作者实际测得。

从图1可见,输入的交流信号经过滤波器滤波后,分成两路,一路经过二极管 D1 半波整流后,提供给 TDA4601 作为工作电压,另一路经桥堆 RS205L 全波整流后作为开关管 C1942 工作时产生交变信号用。TDA4601 是该显示器电源电路的控制器,其输出控制

着开关管的不断导通、截止,从而在开关管的集电极上产生方波信号。

TDA4601 的外部定义见图2。电路开始工作时,在1端建立一个大约4V的基准电压。TDA4601的内部控制逻辑由稳定电路启动,内部各部分均受控制逻辑的控制。4端通过一个RC电路,经变压器与开关管的集电极相联。它采集的信号是对集电极电流的模拟,可作为控制的依据。根据4端的电压值和内部的基准电压值,过载检测机构确定内部调节放大器的工作范围。4端的输入电压由于通过了一个RC电路而变成锯齿

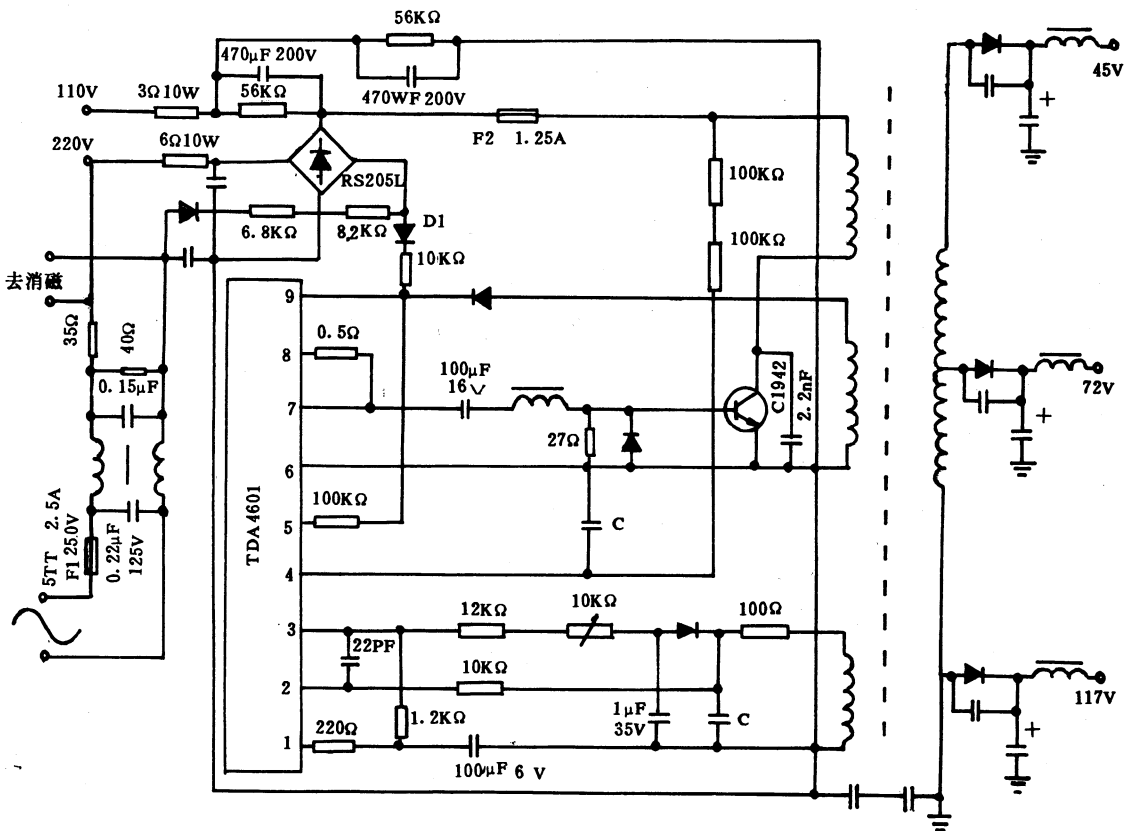


图 1

波,其变化范围为 2V~4V。3 端为反馈端,用作输入调节和过载保护。在变压器反馈线圈上采集的信号经过滤波后,送到 3 端。2 端用作输出的零检测。2、3、4 端均与内部控制逻辑相联。5 端与内部附加的控制电路相联。当 5 端的电压降到 2.2V 以下时,8 端的输出就被阻塞。控制逻辑驱动开关管基极电流放大器(其输出为 8 端)的开闭。8 端的输出电流正比于 4 端的锯齿电压。通过这种关系,TDA4601 能够控制开关管电流满足  $I_c = \beta I_b$  而退出饱和区,8 端与开关管之间的电路提供电流反馈,其它电路参数可决定基极驱动电流的大小。如果控制电路停止提供基极电流,则 7 端的电压被强制降到 1.6V,使开关管截止。若 9 端的电压低于 7.4V,或 5 端电压低于 2.2V,TDA4601 也会使开关管截止。当变压器次级出现短路,TDA4601 会自动进行保护。与开关管基射极相联的二极管的作用是保护开关管。R 用来调整次级输出电压。C 的作用是在开关管输出功率超过允许值时,限制集电极电流。

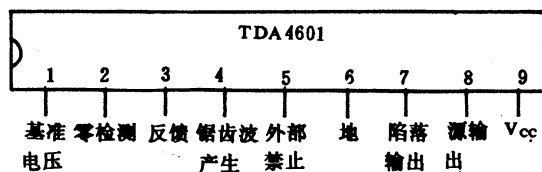


图 2

## 二、维修方法

当显示器出现故障时,应首先判断是否是电源板的故障。可用万用表测量电源板的三个输出是否是 45V、72V、117V。如果输出没有,多数情况下故障在电

源板上。由于开关电源原理复杂,负载端出现故障也可能引起电源板不工作。因此也要检查一下负载端是否出现故障。如判定是电源板故障,应观察两个保险管是否烧坏。

1. 如保险管 F1 烧坏,应检查桥堆 RS205L 是否损坏。按电路原理,桥堆坏的可能性较大。若桥堆完好,可更换保险管 F1 试一下。

2. 如保险管 F2 烧坏,应检查开关管 C1942 是否损坏。如损坏找不到 C1942,则可用 C1875、C1413A、C1325A、C1308 代换。

3. 如 F1、F2 都完好,TDA4601 及外围电路器件损坏的可能性较大。应首先检查外围电路的二极管、电阻、电容是否损坏,阻值、容值是否与标称值一致,是否有虚焊。首先将外围电路的故障排除。如果外围电路器件完好,就要考虑是否是集成电路块 TDA4601 损坏了。通常对集成块的判断要慎重,只有排除外围电路故障可能性后,再考虑是集成块的故障。

最后注意:因为开关电源结构复杂、非线性,同时该显示器电源中的开关管是由 TDA4601 直接驱动的,没有变压器隔离,故往往一旦出现故障,就可能有几个器件同时损坏。为方便大家维修,现将该电源正常工作时的主要参数给出如下:

桥堆正常工作时直流输出电压为 300V。开关管正常工作时  $V_{ce} = 690V(\text{交流}) = 300V(\text{直流})$ 。TDA4601 的各端电压(万用表直流电压档测得)为:  $V_9 = 13V$ ,  $V_8 = 2V$ ,  $V_7 = 2V$ ,  $V_4 = 2.2V$ ,  $V_3 = 2.2V$ ,  $V_2 = 0.16V$ ,  $V_1 = 4.2V$ 。为方便判断 TDA4601 的好坏,将各端对地(6 端)电阻列出如下:  $R_{96}(\text{表示红表笔接 9 端,黑表笔接 6 端——下同}) = R_{69} = 29.3K\Omega$ ,  $R_{86} = 13.75M\Omega$ ,  $R_{68} = 17.3M\Omega$ ,  $R_{76} = \infty$ ,  $R_{67} = 11M\Omega$ ,  $R_{46} = \infty$ ,  $R_{64} = 13.80M\Omega$ 。

# 利用 PCTOOLS 校正软盘驱动器磁头

江西拖拉机发动机厂(330044) 黄焕如

微型计算机的软盘驱动器是故障率较高的部件之一,主要表现在:软盘驱动器使用一段时间后,磁头位置产生偏差,读写数据出错;读了有霉点或者已损伤的软盘,磁头上沾染了不易清洗的介质,造成读写磁道偏位。在上述两种情况下,往往还会出现本驱动器格式化的软盘不能在其他软盘驱动器上使用、双面盘只能被格式化单面盘、启动系统时不能自举、字节数小的文件读写正常,字节数较大的文件无法读写、磁盘 0 面存入的文件能正常读写,而 1 面存入的文件读不出来、只能列目录或部分目录,不能读写文件等故障。

专业修理软盘驱动器需要校正盘(或称猫眼盘)和示波器等工具来调整磁头方位角,一般用户不但缺乏以上工具,而且也难掌握其使用方法,因此不少人利用

BIOS 中断调用 INT13H 的 0H 号(复位磁盘)和 4H 号(检测扇区)功能编制程序,来校验驱动器的磁头位置偏差。

PCTOOLS 是常用的工具软件,一般的用户很容易得到,笔者在实践中发现,利用 PCTOOLS 的拷盘功能能够很方便地校正软盘驱动器磁头位置,该方法安全可靠、简单易学、实用方便。

360K 软盘的存储格局一般为:每个软盘分两面,面 0 和面 1;每个面分 40 个磁道,0~39 道;每个磁道分 9 个扇区;每个扇区 512 个字节。PCTOOLS 在执行拷盘操作时,首先读源盘,读磁道的顺序是:0 面 0 道、1 面 0 道、0 面 1 道、1 面 1 道……0 面 39 道、1 面 39 道,这种读盘方式给软盘驱动器磁头的校正提供了方

便。下面详细介绍利用 PCTOOLS 校正软盘驱动器磁头的具体方法。

如果该驱动器使用的磁盘互换性不好,首先应该在正常的软盘驱动器上重新格式化一块软盘,将软盘插入需要校正的驱动器内(假定为 A 驱动器),然后在另一驱动器或者在硬盘上执行 PCTOOLS 文件,并选择拷盘功能。

C>PCTOOLS\ /F3(磁盘)\ /C(拷贝)\ /A(A 驱动器)\ /A(A 驱动器)\

如果读完该盘后出现图一的画面,则表示该驱动器磁头位置正常,可进一步检验其写盘功能,说明该驱动器磁头位置基本正确。

```

PC Tools Deluxe R4.30
-----Disk Copy Service-----
Insert TARGET diskette in drive B
Press any key to continue

 1 2 3 3
Track 01234567890123456789012345678901234567890
side 0 RRR
side 1 RRR
Press ESC to Exit

```

图一

如果读完该盘后出现图二的画面,则表示该驱动器上磁头(1 面)严重偏移,在该驱动器下格式化的双面软盘变成了单面软盘,或者无法读写 1 面的文件。实际上只要一一开始发现不读 1 面磁道,就可以立即按 ESC 键强行中断,然后依次按 C、A、A 键,等待下一次试读。松开驱动器固定上磁头的螺钉,将上磁头清洗干净,微微移动其位置,然后拧紧螺钉,再按任一键重新读盘,直到出现图一所示的情况为止。

```

 1 2 3 3
Track 01234567890123456789012345678901234567890
side 0 RRR

```

图二

如果读完该盘后出现图三的画面,则表示该驱动器上磁头(1 面)偏移情况比图二所示的情况好一些,1 面的文件仍无法读写,目录却可列出。如上所述,在 1 面连续出现几个 E 时就可强行中断,以节省调试时间。

```

 1 2 3 3
Track 01234567890123456789012345678901234567890
side 0 RRR
side 1 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE

```

图三

如果读完该盘后出现图四的画面,则表示该驱动器上磁头(1 面)偏移情况比图三所示的情况更好一些,存于磁盘前面磁道的文件可能可以读取,后面磁道

上的文件不能读写。这时仅仅需要微微松开螺钉,向某一方向移动上磁头位置,重新读盘后如果在 1 面出现的 E 更多,说明移动的方向和正确的方向相反;如果在 1 面出现的 E 比原来的少,说明移动的方向和正确的方向相同,反复调试直到出现图一般所示的情况为止。

```

 1 2 3 3
Track 01234567890123456789012345678901234567890
side 0 RRR
side 1 RRR

```

图四

如果读完该盘后出现图五的画面,则表示该驱动器上磁头(1 面)位置基本正确。1 面中间出现的少量的几个 E,可能是被测试的软盘本身有坏的扇区,这时可另换一张好的盘片测试。根据经验,只要在最前面和最后面大部分磁道都是 R,中间出现少量几个 E 无关紧要,不需要重新调试磁头位置,否则会越调越乱。

```

 1 2 3 3
Track 01234567890123456789012345678901234567890
side 0 RRR
side 1 RRRRRRRREERRRRRRRRRRRRRRRRRRRRREERRRRRR

```

图五

必须指出,以上调整方法都是假设没有驱动器的读写电路或 DMA 控制电路故障,并且驱动器的磁头已经清洗干净的条件下进行的,否则无法调整正确。由于驱动器下磁头(0 面)基本固定,同时下磁头位于下方,容易清洗和作其他处理,出现位置偏移的情况较少,以上说明的是驱动器上磁头的校调方法。

如果驱动器的上磁头偏移太严重,可能会出现 0 面也不能读的情况,在这种情况下还是要先调试上磁头。如果确系驱动器的下磁头偏位,以上介绍的方法仍然适用,不同的是应该松开固定步进电机的螺钉调试。

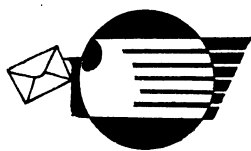
笔者利用 PCTOOLS 及上面介绍的方法,修复校调了十几部有各种读写故障的驱动器,感兴趣的读者不妨试一试。

### 新书邮购消息

|                 |             |
|-----------------|-------------|
| 电子测量仪器选购指南(91年) | 40元(含邮费)    |
| 中国机械电子工业年鉴(91年) | 46元(含邮费)    |
| 微型计算机维修实例 999   | 15.60元(含邮费) |
| 汉字 dBASE IV 基础  | 14.40元(含邮费) |
| 现代计算机技术博览       | 28.80元(含邮费) |
| PC 机通用软件操作手册(一) | 9.70元(含邮费)  |
| PC 机通用软件操作手册(二) | 11元(含邮费)    |
| 91年《电子与电脑》合订本   | 17.80元(含邮费) |

注:邮局汇款:北京万寿路电子工业出版社发行部邮购科,在汇单附言栏内注明书名、册数

银行汇款开户行:北京市工商银行翠微路分理处  
 帐号:661036-40(请随信汇单一起寄书名、册数)  
 本部电话:813693 邮编:100036



读者联谊

## 普及型 PC 个人用户软件交流联谊活动 问题解答(七)

北京中国农科院计算中心(100081) 王路敬

21. 软盘控制器的主要任务是什么?使用软盘驱动器时应注意什么问题?

软盘控制器又称软卡,其主要任务是将 CPU 的并行数据流转换成软盘所需的串行数据流,以及将软磁盘的串行数据流转换成 CPU 所需的数据流。CPU 一旦向软盘控制器发出命令,软盘控制器就要进行如下的工作:选择当前软盘驱动器;控制磁头寻道和定位的机械动作;寻找所需的磁道和扇区并进行读/写操作;在主存储器 and 软盘之间进行数据传送;最后进行较验和出错检测。

主机通过软盘控制器对软盘驱动器进行控制。使用软盘驱动器需要注意下列问题:

(1)工作环境要干净通风。

(2)分清 A 驱动器和 B 驱动器。因为对无硬盘的 PC 机来说只能从 A 驱动器引导系统。

(3)插软盘片时轻轻地插入,而且必须插到位,然后慢慢地关好门。否则,关门时会压伤盘片定位孔,严

重时会使盘片损坏。

(4)盘片的取代,一般情况下双面盘可以代替单面盘,双密度软盘可代替单密度盘,反之不然。

(5)当软盘驱动器指示灯亮时,不能打开驱动器的门从驱动器中抽出盘片,因为指示灯亮,表示软盘驱动器正在运行,若这时硬要开门取盘片,很容易破坏盘中的有用数据,或者损坏磁头,只有在驱动器指示灯熄灭后,再开门取出盘片。

(6)定期的对驱动器磁头进行清洗。清洗时可以用清洗盘清洗,但注意时间一般掌握在 1—2 分钟,否则时间长了会损坏磁头。清洗盘市面上有售。也可以用擦照相机的镜头纸或用棉球沾异丙醇轻擦磁头 1—2 遍即可。

22. 软盘驱动器与软盘的兼容性是如何规定的?

目前在 PC 机上装置的软盘驱动器规格大都是 5.25 英寸或 3.5 英寸,如下表所示:

| 尺寸      | 说明    | 容量          |
|---------|-------|-------------|
| 5.25 英寸 | 单面    | 160KB/180KB |
| 5.25 英寸 | 双面    | 320KB/360KB |
| 5.25 英寸 | 双面高容量 | 1.2MB       |
| 3.5 英寸  | 双面    | 720KB       |
| 3.5 英寸  | 双面    | 1.44MB      |

与各类软盘驱动器相兼容的软盘片从尺寸上也有 5.25 英寸和 3.5 英寸之分。如下表所示:

| 尺寸(英寸) | 说明    | 容量          | 磁道数 | 磁头数 | 扇区/道 | 字节/扇区 |
|--------|-------|-------------|-----|-----|------|-------|
| 5.25   | 单面    | 160KB/180KB | 40  | 1   | 8/9  | 512   |
| 5.25   | 双面    | 320KB/360KB | 40  | 2   | 8/9  | 512   |
| 5.25   | 双面高容量 | 1.2MB       | 80  | 2   | 15   | 512   |
| 3.5    | 双面    | 720KB       | 80  | 2   | 9    | 512   |
| 3.5    | 双面    | 1.44MB      | 80  | 2   | 18   | 512   |

当用 DOS 命令到软盘片中去读写时,必须考虑到软盘片与软盘驱动器的兼容性,具体规定如下:

(1)160KB/180 KB 单面 5.25 英寸驱动器,可以在这种驱动器上读/写的盘片只有:160KB/180KB 单面倍密度盘片。

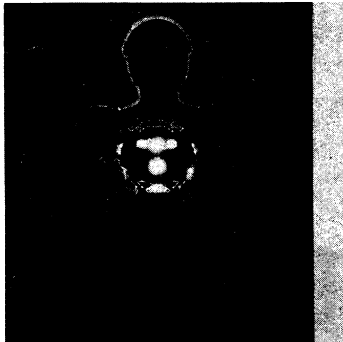
(2)320KB/360KB 双面 5.25 英寸驱动器,可以在这种驱动器上读/写的盘片有:160KB/180KB 单面,倍密度;320KB/360KB 双面倍密度。

(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,倍密度盘片;320KB/360KB 双面,倍密度盘片;1.2MB 高密度盘片。

(4)720KB 双面 3.5 英寸驱动器。可以在这种驱动器上读/写盘片只有,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/写的盘片, (下转第 44 页)





• ELECTRONICS AND COMPUTERS •

一九九二年

总期第89期

# 電子與電腦

## 目 录

• 综述 •

- 计算机信息的安全保护 ..... 李 群(2)
- 量子芯片与人工智能计算机 ..... 姚立新(4)

• PC 用户 •

- dBASE III 管理系统中程序多级调用的跟踪 ..... 周 为(5)
- 一种在 M1724 机上打印图形的实用方法 ..... 李凯里(7)
- 物理扇区与逻辑扇区 ..... 崔来堂(8)
- DOS 命令在 IBM-PC 机通信中的妙用 ..... 林立 林娜(9)
- 硬盘加锁小程序 ..... 刘志存(10)
- 实用程序三则 ..... 任绥海(11)
- 快而短的排序程序 ..... 许野平(12)
- 1724 打印机驱动程序行距的修改 ..... 谭人杰(12)
- 加密 WPS 文件的解密 ..... 谷高平 孟建伟(13)
- 电脑病毒入侵报警程序 ..... 陶秋刚(14)

• 学习机之友 •

- 对几种高精度数值计算方法的改进 ..... 陈庆祥(15)
- CEC-1 自造字键盘辅助点阵生成程序 ..... 汤永进(16)
- 二维函数曲线拟合 ..... 李 涛(17)
- 在 Applesoft 系统下实现 COMMON 功能 ..... 安赵根(18)
- 通用打印课表程序 ..... 程建伟(19)
- CP/M 系统文件的恢复方法 ..... 李 齐(20)

• 6502 机器语言讲座 •

- 第八讲 子程序设计 ..... 朱国江(21)

• 初级程序员级软件水平考试辅导 •

- 一九九二年计算机初级软件人员竞赛试题 ..... (26)

• 学用单片机 •

- BJS-51 的监控程序 ..... 盛焕鸣(31)

• 学装微电脑 •

- 水平多关节型机器人 ..... 易齐干(34)

• 电脑巧开发 •

- 点阵字符液晶显示器与单片机接口 ..... 张培仁 杨建景(38)

• 电脑游戏机 •

- F BASIC 语言的游戏程序编写技巧  
第一讲 游戏程序的概念 ..... 于 春(41)

• 维修经验谈 •

- 单色显示器故障维修一例 ..... 梁建华(43)
- 恢复 CEC-1 学习机 DOS 的 I/O 控制 ..... 屈晓柳(43)
- 再谈清洗盘的正确使用 ..... 刘德一(44)
- 对磁盘局部缺损的处理 ..... 何管略(44)

• 新书与软件 •

- 电子工业出版社软件部新出版软件介绍 ..... (45)

• 读者联谊 •

- 普及型 PC 个人用户软件交流联谊活动问题解答(八)  
..... 王路敬(46)

封一:机器人

封二:NS 公司单片微机一览表(资料)

封三:NS 公司单片微机一览表(资料)

封四:广告

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



# 计算机信息的安全保护

辽宁省财政厅信息中心(110002) 李群

随着工业化社会向信息化社会过渡的潮流,以计算机技术为主导的高科技成果日益进入社会的各个职能部分,计算机网络迅速发展。据统计,我国现有各种微机四十多万台,中、小型机一万台,计算机已经广泛应用于一系列国民经济的重要部门,在提供决策、制定政策等方面起着越来越重要的作用。然而,由于计算机信息系统和网络本身的脆弱性,以及各种人为因素的影响,计算机信息常常受到干扰、滥用、丢失、甚至被泄漏、窃取、篡改、冒充和破坏;计算机可能受到“病毒”感染而工作瘫痪。利用计算机犯罪也成为高技术犯罪的衍化。加之各种技术性事故、无意疏忽而带来的一系列信息的不安全性,都要求从法律上、组织上及技术上加强计算机信息安全的保护。

## 一、计算机信息安全保护的范畴

信息是一种资源,它是依据某种目的组织起来、经过加工处理和具有一定结构的数据总括。计算机信息可分为应用信息和系统信息两种。对这些信息的威胁大致分为三类:

1. 对计算机应用信息机密性的威胁;
2. 对计算机应用信息真实性和完整性的威胁;
3. 对计算机系统信息的破坏性威胁。

信息保护的主要目的,一是保护信息的机密性,防止未经授权的泄漏;二是保护信息的真实性和完整性,防止未经认可的修改。信息保护有二个重要环节:一是保护通信线路上传输的信息,二是保护存储于计算机系统内的信息。

经计算机通信线路传输的信息,常常受到两种威胁。一是被动窃取,即信息机密性遭到破坏,通常指截取报文而不被察觉出来。保护报文内容不被识破,是依靠对传输信息进行加密变换。二是主动窃取,即信息真实性或完整性遭到破坏,这是指蓄意修改报文流。对这种威胁,加密保护只能提供发现作案,而不能阻止作案。由于我国现在通信网络还没有完全发展起来,针对此类信息的安全的威胁还比较少见。

存储于计算机系统的信息,易于受到和通信线路同样的威胁。威胁到信息机密性的手段有:从主存储器或外存储器查找信息;用合法存取信息的进程为掩护,把信息传给未经认可的用户;或根据公布的一批综合性统计数据,推断出有价值的机密数据。

存储于计算机系统的信息所受到的真实性和完整性威胁,是大量的计算机犯罪的一种结果,这主要指蓄意篡改和破坏计算机的信息(如把某人存款非法提高等);冒充系统获得对其他用户的访问权,而非法

修改合法用户的信息。

另一种对计算机系统信息造成较大破坏性威胁的,就是大量的计算机病毒。这种威胁也是广大计算机用户最常见的。由于我国在计算机立法方面起步较晚,软件产品非法复制、拷贝,加之前几年正处我国掀起普及微机的热潮,为计算机病毒的侵入提供了必要环境,而计算机软、硬件产品的脆弱性又为病毒的侵入提供了技术原因。计算机病毒的传染一般通过三种途径:通过软盘,指不经过严格的检查而轻易使用外界有可能被感染的软盘;通过带有病毒的硬盘,将干净的软盘传染并再扩散;通过网络,这种传染扩散极快,能在很短时间内传遍网络上的机器。不管是良性病毒还是恶性病毒,轻则占用 CPU 时间和存储空间,影响机器使用效率,重则破坏用户或系统数据,甚至完全毁灭系统。病毒对我国的广大微机用户造成了巨大的威胁,尤应引起高度重视。

## 二、计算机信息安全保护的措施

为了有效地对各种计算机信息进行安全保护,必须从法律上,行政上及技术采取相应措施。根据计算机安全专家的思想,保护计算机系统的软件(数据文件和程序)和硬件(设备)有四层安全措施:第一层是法律和社会管理,它通过国家法律和地方法律及社会上允许的行为方式提供保护;第二层是用户本单位的行政规章,规定有关计算机系统使用方法和手段,以及人事结构和人事政策提供保护;第三层是物理保护,指使用防盗和防火技术、软盘保护、限制实际接触计算机系统;第四层是电子的和程序的,是指由硬件器件和在操作系统、数据文件和应用程序中,用软件方法提供的保护。

在国际上,许多国家已经正式颁布了信息保护法。美国于 1974 年正式颁布信息保护条例,1980 年修订了著作权法,正式认定计算机软件是独特的智力产品,是著作权保护的客体。随后,日、英、法等西方国家也为保护软件修改和实施了著作权法。同时,以美国为首的西方国家还竭力以法律手段为软件提供保护,并使之具有国际效力。为了适应改革和对外开放的需要,加速引进国外智力投资,衡量和保证我国引进技术的质量,维护软件开发者合法利益,促进软件业的发展和竞争,防止重复投资,避免低水平重复劳动,我国已制定并颁布了著作权法和软件保护条例,明确规定了软件产品保护的客体、范围、保护期限及侵权的法律责任,从而为智力产权提供了法律上的保护,为各级司法机关审理侵权行为提供了法律准则。为了维

护软件开发者的利益,迫切需要在全民中普及法制观念,当发生侵权行为时,运用法律手段解决问题。

操作保护措施是指管理上采用的政策和过程,它们保证计算机系统及其信息的安全,虽然管理政策和过程的若干方面已经由外部规定,但是,多数管理政策和过程的实施细则,仍需由内部规定。与信息安全有关的主要管理问题是认可权,即认可谁,根据什么,如何确定。这种保护可以从操作上,组织上提供。主要采取以下方法:

1. 对使用计算机人员的数量加以限制。操作人员应分为一般用户,系统维护人员及特殊用户(如对一个公司的计算机系统,经理则为此类人员,授权级别最高,可以浏览全部信息),对各种人员根据其实际需要,授与相应权限,严格禁止越权操作。

2. 系统地提出并得到上级支持和群众积极参与的安全管理条例及其实行细则,使单位人员认识到信息安全的紧迫性和重要性,自觉地遵守规则,并运用行政手段对违反人员进行惩罚。

物理保护也称实体保护,是对其环境,设施,设备和人员等采取安全措施,使它不致受自然或人为因素的损害。它主要可分为:计算机场地环境和设施保护、实体和信息载体戒备措施。

选择计算机房场地时,应考虑以下问题:

1. 机房不应选择在下列地区:山脚、地基不牢处,靠水或低洼潮湿处,地下有输油输气管道、地震活动频繁及邻近有高层建筑结构处。

2. 应尽量远离有害化学气体、腐蚀性物品和易燃易爆物品存放处。

3. 应远离高压线,避开雷达站、无线电发射台和微波中继站。

4. 应尽量远离强振动源和噪声源。

5. 机房不应设在建筑物高层,因为高层易受外来电磁波干扰。

6. 应有紧急照明供电设施和各种报警装置。

7. 机房应有防火及防水设施。

8. 提供有不同持续工作时间的不间断电源。

9. 保证必要的温度、湿度和防尘指标。

实体和信息载体戒备,可以将处理中心分为二个区:主机房和数据处理区。根据人员实际需要规定进入的区域;在建筑上应防止电磁波破坏,磁带和磁盘应在离开现场的安全处,对载有重要信息的磁带和磁盘,应建库由专人保管,严禁非法的访问与外借。

对信息实体保护除以上措施外,还应当制定应急计划,防备一旦计算机实体发生重大事故并难以迅速恢复时,能够尽可能减少损失。这类应急计划应尽可能详尽地考虑到各种情况下有可能造成的信息丢失或泄漏时的补救方案,以备灾后恢复,如同一现场有两套计算机系统同时工作,互相备用,此时安全度随两机距离增加而加大;对重要信息进行备份等。

对计算机硬件的保护主要包括:中央处理机、存储器、输入输出通道和外围设备的保护,主要由硬件

人员负责定期检查和维修,使计算机系统处于良好的状态,避免由于硬件故障而造成信息破坏。

对计算机信息安全造成威胁最大,破坏程度最深的是大量的针对计算机软件方面的威胁,安全保护的主要工作也在这个方面。由于无论数据还是程序,应用软件还是系统软件,在计算机中都是以文件方式存放的,因而,现在在技术上采用的方法主要有:操作系统保护和文件保护。

对操作系统进行保护,在每次交互式或批量处理会晤期的开始和结束,登录过程必不可少。通过登录,操作系统激活标识和鉴定机构,准确无误地辨识出合法用户。同时,由操作系统负责控制一些安全软件(有些已经固化),一旦发生用户企图修改或无意识地修改系统数据或未被授权的数据,则加以制止。这种方法可以有效地防止一些计算机病毒的侵害。

对文件的保护通常利用口令字、存取权控制、加密和其他一些保护措施,保证文件安全。在用户登录、文件存取或数据项存取期间,设置口令字可对用户身份作出鉴定。一个具体系统选用的鉴定技术,有赖于风险和代价。对可能的威胁、威胁出现的概率及构成损失的估价,与采用保护对策付出的代价存在一种关系。与其他鉴定技术比较,口令字技术付出代价小,口令字手段对制止浏览(即用户利用对系统中一部分合法存取权存取其他未经认可的文件)特别有效。在各级目录和文件上设置口令字,可以对文件存取控制起到保护作用。一个用户申请存取文件,就必须提出正确口令字。这种口令字保护的优点是简便,所需存储空间少。缺点是系统管理人员能取得全部口令字。因为该保护信息存储于系统内。另一种保护措施是把全部文件作加密编码。所有用户都可以存取加密码的文件,但只有合法用户才知道解码的途径。任何时候,用户可以根据需要变更密钥。对文件解码和编码工作,均由操作系统实现。这种方法可以杜绝利用系统内存储信息的非法存取,但以编码和解码时的开销为代价。还有一种保护措施即存取权的设置和利用。存取权(许可权)可分为共用许可权和专用许可权。共用许可权的文件或目录,对所有系统确认的用户都是可用的;而专用许可权的文件或目录,则只对那些事先授权的用户是可用的。一个文件或目录只有一种许可权,当同时拥有共用许可权和专用许可权时,则共用许可权取代专用许可权。许可权大致包括下列几种:可读、可写、可执行、恢复、消除、建立、修改和上锁。除以上措施外,还可以在文件正常管理中,统计和追踪异常情况下文件的封锁和非法存取文件的次数,记录文件的跟踪记录(系统日记)。为减少系统开销,可由操作员用命令选择全跟踪、跟踪几项、不跟踪。根据情况,可以把上述几种方法综合利用。

### 三、计算机安全保护的前景及面临的问题

计算机信息安全保护的很广、涉及面大,必须尽可能防止各种不安全因素,特别是随着科技的发展,各国之间的竞争将更趋激烈,以电子技术为先导

的竞争对计算机信息安全提出了更高的要求,即保证计算机信息永远处于安全状态。信息尤其是重要信息更不能丢失或毁坏。这就要求安全专家们从方法学角度,研究计算机系统和通信网络内信息保护方法,使计算机信息保护理论更为系统和完整。可以说,一个好的计算机系统应该在为用户提供灵活方便的功能的同时,也为用户提供良好的安全保护措施。

任何事物都不是绝对的,计算机信息也不可能有百分之百的安全。信息共享与保护、破坏与安全、扩散与保密是互相对立、互为制约的。安全性的提高是以对其他方面的限制为代价的。同时,计算机信息保护环境的复杂多变性,使实现一个有成效的、经济可行和易于使用的信息保护系统十分困难,这是因为:

1. 信息保护系统是否有成效,不仅取决于保护措施是否完善,也取决于袭击者的手段是否高超,取决

于系统脆弱性是否易于识破,取决于系统一旦遭到破坏所造成的直接和间接损失。事实上,要估量袭击者采用的手段和系统潜在威胁是极其困难的。

2. 内部和知情人员的破坏,更危险和难以防止。

3. 技术事故和误操作造成的损失和破坏次数较之蓄谋作案大得多。

4. 各类信息保护系统差异很大,研制通用的信息保护系统很困难。

5. 降低信息保护软件、硬件的成本和管理费用,减少多余开销,始终是一个重要因素,尤其在商用系统中。同时,还应当把保护与易于使用结合起来。

总之,计算机使用得越广泛深入,对信息共享与保护的需求就越突出。因此,满足用户的要求,具有合理的安全价格比的系统就是一个良好的计算机系统。

## 量子芯片与人工智能计算机

山西临汾地区计委经济信息中心(041000) 姚立新

电子计算机从1946年间问世以来的40年多里,经历了电子管、晶体管、集成电路、大规模集成电路四代更新,目前正向着采用第五代元器件构成的智能计算机(也叫第五代计算机)方向发展。那么,人工智能计算机与传统的计算机有什么区别呢?与计算机发展密切相关的元器件——芯片发展的现状又是什么样呢?

### 人工智能计算机与传统计算机的区别

首先,这两种计算机的区别在于传统的计算机采用的是冯·诺依曼结构(也叫冯·诺依曼计算机),而人工智能计算机采用的是非冯·诺依曼结构。所谓冯·诺依曼结构,它的特点是把要处理的数字、文字、图象、声音全部变成1或0这样的数字进行运算和记忆,而且按指令的内容,一条一条地完成所规定的任务。人工智能计算机中引入了数据流的概念,这种计算机是由数据而不是由指令来驱动程序的执行,它是靠运算器中操作数的全部到达为标志来驱动操作执行的机器,因此,当多个操作数同时满足这一条件时,它们就可以并行地执行而不受程序的顺序限制,据科学家初步估计,数据流计算机的处理速率要比冯·诺依曼计算机至少提高2个数量级。

其次,这两种计算机的区别还在于人工智能计算机采用了并行结构,传统计算机的基本运行顺序是从中央存储器中取出数据,对它进行加工,再存回到中央存储器,而在人工智能计算机中采用的却是并行结构。比如:有一种平行计算机,它的一个处理机可以管十来个象素,那么就可能在时间上并行地取出多片数据来,

对它们同时施以一系列的运算,然后再送回到分布式存储器中,这样一来,就极大的提高了计算机的效率。

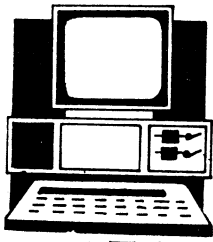
另外,在人工智能计算机中还采用了比传统计算机更容易设计的机器语言——PROLOG语言,这种语言是一种逻辑设计语言,它使得程序员能够用逻辑来规定机器的任务而不是象传统计算机那样用程序设计来规定机器在什么时候作什么工作。

### 芯片发展趋势——量子芯片

计算机离不开芯片的支持。对人工智能计算机来说,芯片的精度要求更高。在日本,科学家已研制出体积只有指甲大小但却具有64兆位的记忆芯片,在这种芯片上密密麻麻地分布满了1.4亿个电子元件,而连接它们的电路只有0.4微米宽,相当于人的头发丝直径的千分之四。但这还是不够的,为了提高芯片的运算速度,还必须缩小电子元件的尺寸。然而科学家认为,当芯片的电路宽度小于0.1微米时;电子就不会那么老实实在地在电路里运动了,它们很可能从电路里跳出来,将整个芯片搅得一片混乱,因为此时电路的宽度太窄已使得电子的波动性不可忽略了,此时,电子既具有粒子特性,又具有较强的波动特性,这样就引进了量子芯片。

所谓量子芯片就是利用电子的波动性来设计出的一种高密度芯片,该芯片的问世将使巨大的超级计算机体积变得与现在膝上电脑那么大,并且使得计算机的运算速度到万亿次成为可能。虽然量子芯片的研究还处在初级阶段(比如说,目前日本富士通研制出的实验型量子芯片必须在零下200C才能工作。),但是,通过科学家的努力,不久的将来这种芯片将进入实用阶段。

总之,人工智能计算机是一种集成度更强、运算速度更快、智能程度更高、更接近于人的思维行为的一种计算机系统,量子芯片的研制为人工智能计算机的设计提供了一种理想的硬件,使将来的计算机体积更小,运算速度更快。



PC用户

# dBASE III 管理系统中 程序多级调用的跟踪

甘肃平凉 103 信箱 302 分箱(744000) 周 为

编写或使用 dBASE III 管理系统程序的人员常常由于工作上的需要,希望能够迅速地了解管理系统调用各子程序的调用关系和能打印一份该系统完整的程序清单,以便于存档,用于今后该系统的维护、学习,交流和改进编程技巧。但由于每个管理系统中的每一功能还常常包含着多级的子程序调用关系,而且每级中又有着多个数量不确定的子程序,一般情况下想要全面地了解系统的程序调用关系和打印出一份完整的程序清单是不容易的。因为我们通常只能自上而下地对程序进行分析,逐步了解各程序之间的调用关系而打印出全部程序清单。显然这是一件相当麻烦的工作。

笔者针对以上存在的问题编写了一个包括“自动建立管理系统多级调用关系数据库”、“打印系统程序调用关系的结构示意图”和“自动打印系统程序的全部清单”三个功能的子程序。操作者只需通过选项 1 键入当前管理系统的主菜单名,就能自动建立起该系统中的调用关系数据库。随后再键入 2 或键入 3,就可以达到我们期望的目的。

| 库文件结构—— |       | 文件名:C;DB.dbf |    |    |
|---------|-------|--------------|----|----|
| 字段      | 字段名   | 类型           | 宽度 | 小数 |
| 1       | PRG 名 | 字符型          | 12 |    |
| 2       | JB    | 数值型          | 1  |    |

| 库文件结构—— |       | 文件名:C;DB1.dbf |     |    |
|---------|-------|---------------|-----|----|
| 字段      | 字段名   | 类型            | 宽度  | 小数 |
| 1       | PRG 名 | 字符型           | 100 |    |
| 2       | JB    | 数值型           | 1   |    |

```
* FILE. PRC
SET TALK OFF
SET SAFE OFF
DO WHILE . T.
CLEAR ALL
CLEAR
STORE 0 TO LL,YY
NAME1=" "
@5,24 SAY"*****"
@7,24 SAY"1. 自动建立子程序多级调用关系数据库"
@8,24 SAY"2. 打印子程序调用结构示意图"
@9,24 SAY"3. 自动打印系统程序全部清单"
@10,24 SAY"0. 退出"
@12,24 SAY"*****"
@14,34 SAY"请输入(0-3)"GET YY PICT"9"RANGE 0,3
READ
DO CASE
CASE YY=1
```

```
ACCE "主菜单文件名:" TO NAME1
NAME=NAME1+". PRG"
IF FILE('&NAME')
SELECT 1
USE DB
ZAP
APPE BLANK
REPL PRG 名 WITH NAME,JB WITH 1
KK=JB+1
POINT=1
DO FILE1
POINT=POINT+1
USE DB
APPEND FROM DB1
APPEND BLANK
REPL JB WITH 0
DO WHILE . T.
USE DB
GO POINT
IF JB=0
EXIT
ELSE
NAME=PRG 名
ENDI
KK=JB+1
POINT=POINT+1
GO POINT
COPY TO DB2 NEXT 100
DO FILE1
IF LL=0
LOOP
ENDI
USE DB1
APPEND FROM DB2
USE DB
GO POINT
DELE NEXT 100
PACK
APPEND FROM DB1
ENDD
USE
ELSE
? CHR(7)
? '没有此文件'
ENDI
CASE YY=2
USE DB
SET PRIN ON
```

? “子程序调用结构示意图”

```
DO WHILE .NOT. EOF()
DO CASE
CASE JB=1
? PRG 名
CASE JB=2
? SPACE(4)+“┆——”+PRG 名
CASE JB=3
? SPACE(14)+“┆——”+PRG 名
CASE JB=4
? SPACE(24)+“┆——”+PRG 名
CASE JB=5
? SPACE(34)+“┆——”+PRG 名
CASE JB=6
? SPACE(44)+“┆——”+PRG 名
CASE JB=7
? SPACE(54)+“┆——”+PRG 名
CASE JB=8
? SPACE(64)+“┆——”+PRG 名
CASE JB=0
EXIT
ENDCASE
SKIP
ENDD
?
SET PRIN OFF
CASE YY=3
USE DB
DO WHILE .NOT. EOF()
```

```
MM=PRG 名
TYPE &MM TO PRIN
SKIP
ENDD
CASE YY=0
RETU
ENDCASE
ENDD
* FILE1. PRG
USE DB1
ZAP
APPE FROM &NAME SDF
DELE ALL FOR AT(“DO”,PRG 名)=0
PACK
GO TOP
DO WHILE .NOT. EOF()
NAME=TRIM(SUBS(PRG 名,AT(‘DO’,PRG 名)+3,8))+
“_PRG”
IF FILE(‘&NAME’)
REPL PRG 名 WITH NAME,JB WITH KK
ELSE
DELE
ENDI
SKIP+1
ENDD
PACK
COUN TO LL
USE
RETU
```

(接第 42 页)

```
25 PL. “G4A1;F4F1;O3D4O2B1”
30 PL. “B5O4C7;F5E7;G5C7”
35 CLS;ERA 0,1;SP. O.;CG. 0,0;U=21;PAL. B 0,15,
48,48,U
40 A$=“AIIIIIIIIIGIGGGGH”;B$=“HHBHHBBE-
BEEE”;K$=“ABCDEFGH1”;I$=“ABCGHIDEF”
45 LOC. 9,10;P. “START!!!”;PAU. 80;CLS;X=13;
Y=22;I=0
50 GOS. 100;SW. A$,B$;GOS. 100
55 IF I=0 T. SW. K$,I$;I=1;G. 50
60 G. 150
100 L=LEN(A$);F. P=1 TO L;K=P;IF I=0 T. 110
105 K=L+1-P
110 C=INS. (K$,MI. (A$,K,1))-1;V=C/3;H=C
MOD 3
115 X=X+H+3*(H=2);Y=Y+V+3*(V=2);LOC.
X,Y;P. CH. (207);N.;RE.
150 W=52;F.N=0 TO 1;DE. M. (N)=SP. (N,3+4*N,
1,1,0,0);POS. 236*N,200;N.
155 M. 0,1;IF CRA. (0)=1 T. 165
160 G. 155
165 LOC. 10,10;P. “PERFECT”;F.N=0 TO 100;PAL. B 0,
6,48,48,U;SW. U,W;N.
170 PL. “M1V2Y1T2;M1V10Y1T2;T2”
```

```
175 PL. “O3E3G#DGDG#DG;O1#B5B#AB;R”
180 PL. “EGO4CO3E;Y2O2#B6E3;O1C5C”
185 PL. “#GAO4DC;F5#F;#C#C”
190 PL. “O3B#ABO4A;F3O3DFA;D6F3”
195 PL. “GO3AO4FO3B;E5DV15;GA#AB”
200 PL. “#B3GT3FGE1DV12E7;C9;#B5T3GC7”
210 LOC. 0,22;E.
```

程序说明:

1. 10~30 行为开始音乐。
2. 40~115 行打印心形图案,其原理已在 FB ASIC 语言及程序设计中作过介绍。
3. 150~160 行定义玛丽和丽莎运动,注意两卡通运动方向和运动横座标的技术处理。
4. 165 行的作用是使图案闪光,实际上是红、白两种颜色背景的交替显示。
5. 170~200 行是祝贺音乐。

分析上两个简单游戏程序,得出结论如下:游戏程序是具有某种游戏功能的特殊程序。它同其它任何程序一样,具有输入、输出、循环、判断、结束等处理语句,所不同的是,它比一般程序结构更复杂,内容更丰富。所以,我们说游戏程序是电脑实力的充分发挥,是电脑计算、判断、声音、图象、动画等各种功能的集合表现。

注:例 2 的程序清单的指令使用了简写形式,为压缩篇幅,以后各讲均使用简写形式,不再说明。

# 一种在 M1724 机上打印图形的实用方法

山东烟台大学(264005) 李凯里

许多只具备基本配置的 PC 机用户,为了便于技术信息交流,常常希望使用普通打印机得到较满意的图形。

这里向配有 M1724 打印机的 PC 用户介绍一种简便实用的图形打印方法。其主要特点一是能将屏幕图形完全按比例地拷贝下来,不会出现几何畸变(如当屏幕显示正方形时,打印输出图形仍为正方形)。其二是分有正常尺寸与扩大尺寸两种选择。后者恰为前者之两倍,约占一页打印纸的长度。图为按正常尺寸打印的图形例子(一平面振荡波形的局部立体图)。

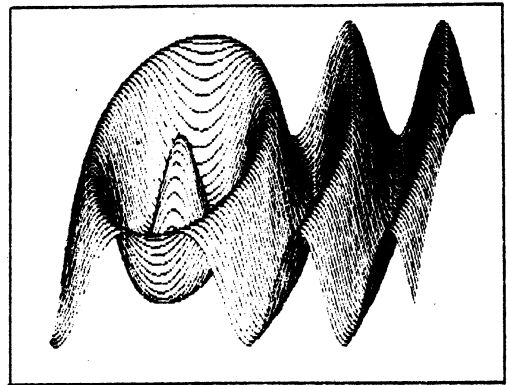
程序用 BASIC 语言写成,用户易于读懂。其中 BX(20)数组用于扩大方式下的打印字节转换。变量 WOB 产生反视效果(黑色底图)。

欲打印的图形事先已经生成,并以 \*.GDT 文件名存入硬盘。每次打印前可按文件名进行选择。

本程序中各循环参数视图形规模及内存格式而定。这里为 16K/帧,首地址在 B8000H 处。对其规模 and 存放格式的图形,需修改相应的参数。另外特别指

出,通过适当调整打印控制字,也可将该程序的设计原理应用于其他型号打印机的图形打印。

(注:本例 M1724 机的 DIP 开关第 7 位置于 ON 位置)



图

## 图形打印程序清单

```
100 '*** M1724 ***
110 CLEAR;CLS;SCREEN 1;KEY OFF;DIM BX(20)
120 BX(0)=&H0;BX(4)=&H30;BX(8)=&HC0;BX(12)
 =&HF0
130 BX(1)=&H3;BX(5)=&H33;BX(9)=&HC3;BX(13)
 =&HF3
140 BX(2)=&HC;BX(6)=&H3C;BX(10)=&HCC;BX(14)
 =&HFC
150 BX(3)=&HF;BX(7)=&H3F;BX(11)=&HCF;BX(15)
 =&HFF
160 FILES "*.GDT
170 PRINT;INPUT "Filename?";FILENAME$
180 CLS;PRINT "0--Normal 2--Enlarge
190 K$=INKEY$;IF K$="0" THEN SX=0 ELSE IF K$
 ="2" THEN SX=2 ELSE 190
200 PRINT;PRINT "1--Black on white 3--White on black
210 K$=INKEY$;IF K$="1" THEN WOB=0 ELSE IF K$
 ="3" THEN WOB=255 ELSE 210
220 DEF SEG=&HB800;BLOAD FILENAME$
230 E$=CHR$(27);LPRINT E$+"@";WIDTH
 "LPT1";,255
240 LPRINT E$+"J"+CHR$(19)
250 IF SX=0 THEN 680 ELSE 260
260 '*** Enlarge ***
270 FOR ROW=0 TO 80 STEP 3
280 LPRINT E$;"=";CHR$(1);CHR$(244);
290 FOR COL=99 TO 0 STEP -1
```

```
300 LOCA=COL*80+ROW
310 BYTE1=PEEK(LOCA+&H2000) XOR WOB
320 BYTE2=PEEK(LOCA+&H2001) XOR WOB
330 FOR K=1 TO 3
340 LPRINT CHR$(BX((BYTE1 AND &HF0)/16));
350 LPRINT CHR$(BX(BYTE1 AND &HF));
360 LPRINT CHR$(BX((BYTE2 AND &HF0)/16));
370 NEXT K
380 BYTE4=PEEK(LOCA+0) XOR WOB
390 BYTE5=PEEK(LOCA+1) XOR WOB
400 FOR K=1 TO 2
410 LPRINT CHR$(BX((BYTE4 AND &HF0)/16));
420 LPRINT CHR$(BX(BYTE4 AND &HF));
430 LPRINT CHR$(BX((BYTE5 AND &HF0)/16));
440 NEXT K
450 NEXT COL
460 LPRINT
470 LPRINT E$;"=";CHR$(1);CHR$(244);
480 FOR COL=99 TO 0 STEP -1
490 LOCA=COL*80+ROW
500 BYTE2=PEEK(LOCA+&H2001) XOR WOB
510 BYTE3=PEEK(LOCA+&H2002) XOR WOB
520 FOR K=1 TO 3
530 LPRINT CHR$(BX(BYTE2 AND &HF));
540 LPRINT CHR$(BX((BYTE3 AND &HF0)/16));
550 LPRINT CHR$(BX(BYTE3 AND &HF));
560 NEXT K
```

```

570 BYTE5=PEEK(LOCA+1) XOR WOB
580 BYTE6=PEEK(LOCA+2) XOR WOB
590 FOR K=1 TO 2
600 LPRINT CHR$(BX(BYTE5 AND &HF));
610 LPRINT CHR$(BX((BYTE6 AND &HF0)/16));
620 LPRINT CHR$(BX(BYTE6 AND &HF));
630 NEXT K
640 NEXT COL
650 LPRINT
660 NEXT ROW
670 END
680 '*** Normal ***
690 FOR ROW=0 TO 80 STEP 3
700 LPRINT E$,"G",CHR$(1);CHR$(244);
710 FOR COL=99 TO 0 STEP -1
720 LOCA=COL*80+ROW
730 BYTE1=PEEK(LOCA+&H2000) XOR WOB
740 BYTE2=PEEK(LOCA+&H2001) XOR WOB
750 BYTE3=PEEK(LOCA+&H2002) XOR WOB
760 BYTE4=PEEK(LOCA+0) XOR WOB
770 BYTE5=PEEK(LOCA+1) XOR WOB
780 BYTE6=PEEK(LOCA+2) XOR WOB
790 A$=CHR$(BYTE1)+CHR$(BYTE2)+CHR$(BYTE3)
800 B$=CHR$(BYTE4)+CHR$(BYTE5)+CHR$(BYTE6)
810 LPRINT A$;A$;A$;B$;B$;
820 NEXT COL
830 LPRINT
840 NEXT ROW
850 END

```

# 物理扇区与逻辑扇区

石家庄铁道学院(050043) 崔来堂

## 一、两种编址方式

扇区是磁盘操作的最小可寻址单位。PC-DOS对磁盘扇区的管理,分为物理扇区和逻辑扇区两种方式。物理扇区是按照磁道(硬盘为柱面,下同)、磁头、扇区的三维物理位置而进行编址的,逻辑扇区则是将磁盘扇区自始至终进行连续一维编址的结果。

逻辑扇区也称相对扇区,其编址规则是:以盘的0磁道0磁头1扇区为逻辑扇区0,由此开始,对0磁头各扇区依次编号,接着对1磁头各扇区编号,……直到各磁头对应的扇区编完,再编1磁道各磁头的扇区,……这样一直编到末磁道末磁头的末扇区。

## 二、有关的中断调用

有两种中断调用专用于磁盘的扇区操作。一是ROM BIOS的INT 13H,其入口参数用物理扇区:

- AH 子功能号(0:磁盘控制器复位,2:读盘,3:写盘,5:格式化)
- AL 扇区数
- CH 磁道号
- CL 起始扇区号
- DH 磁头号
- DL 驱动器代号(0:A,1:B,80:C,……)
- ES:BX 内存缓冲区指针

另一是DOS的INT 25H(读盘)和INT 26H(写盘),其入口参数用逻辑扇区:

- AL 驱动器代号(0:A,1:B,2:C,……)
- CX 扇区数
- DX 起始逻辑扇区号
- DS:BX 内存缓冲区指针

物理扇区直观,但参数较多,如果待操作的若干扇

区不在同一磁道同一磁头内,就必须分几次调用INT 13H。逻辑扇区参数单一,待读写的多个扇区只要编号连续,一次调用INT 25H(26H)即可,使用起来很方便。对于硬盘,物理扇区的概念适用于整个盘空间,逻辑扇区只适用于DOS分区,因此,对主引导扇区的读写不能使用INT 25H(26H),而必须使用INT 13H,所用程序小段如下(设由DEBUG编写,读主引导扇区):

```

-A100
MOV AX,0201
MOV CX,0001
MOV DX,0080
MOV BX,0200
INT 13
INT 20

```

## 三、换算方法

应用中,两种扇区之间有时需进行换算。

### 1. 由物理扇区换算为逻辑扇区

5英寸360K软盘的换算公式是:

逻辑扇区号=磁头号\*9+磁道号\*18+扇区号-1

5英寸1.2M高密软盘的换算公式是:

逻辑扇区号=磁头号\*15+磁道号\*30+扇区号-1

### 2. 由逻辑扇区换算为物理扇区

通用公式是:

扇区号=逻辑扇区号 MOD 每道扇数+1

磁头号=(逻辑扇区号/每道扇数) MOD 磁头数

磁道号=逻辑扇区号/每道扇数/磁头号

其中,MOD为除后取余数,“/”为除后取整数商。

DOS模块能够完成上述换算。实现逻辑扇区换算为物理扇区的子程序如下(其中,CODE为段名,BPB



为盘的 I/O 参数表, LOG-S 单元含有待换算的逻辑扇区号):

```

LS-PYA PROC NEAR
 PUSH CS
 POP DS
 ASSUME DS, CODE
 PUSH AX
 PUSH DX
 MOV AX, WORD PTR LOG-S
 XOR DX, DX
 DIV WORD PTR BPB+13; 除以每道扇数
 INC DL
 MOV PYS-S, DL; 送物理扇区号
 XOR DX, DX
 DIV WORD PTR BPB+15; 除以磁头数
 MOV PYS-H, DL; 送磁头号
 MOV PYS-C, AX; 送磁道号(PYS-C 按字定义)
 POP DX
 POP AX
 RET
LS-PYA ENDP

```

#### 四、基本应用

磁盘文件的读写过程清晰地说明了两种扇区的应用及它们之间的关系。今以读文件为例简述如下:

DOS 首先从盘的文件目录表 FDT 中查找待读文件名和扩展名, 若存在, 由对应目录项中查得该文件的起始簇号; 再由此访问盘的文件分配表 FAT, 依次查出文件分布在盘文件区的各个簇, 每查得一个簇号, 先按下述公式换算为逻辑扇区:

| 盘类型     | DOS 版本号 | 换算公式       |
|---------|---------|------------|
| 360K 软盘 | 2. X    | 簇号 * 2 + 8 |
| 1.2M 软盘 | 3. X    | 簇号 + 27    |

|        |      |              |
|--------|------|--------------|
| 20M 硬盘 | 2. X | 簇号 * 16 + 49 |
| 20M 硬盘 | 3. X | 簇号 * 4 + 107 |

然后利用前述子程序将逻辑扇区换算为物理扇区, 再调用 INT 13H 完成读盘。

DOS 模块中调用 INT 13H 的子程序如下(其中, DRIVE 单元含有驱动器代号, 其它标号的意义同上述子程序):

```

DISKIO PROC NEAR
 PUSH CS
 POP DS
 ASSUME DS, CODE
 PUSH DX
 PUSH CX
 MOV DX, WORD PTR PYS-C
 MOV CL, 6
 SHL DH, CL; 获取磁道号的高 2 位
 OR DH, PYS-S; 与扇区号组合
 MOV CX, DX
 XCHG CH, CL; 使 CH 为磁道号, CL 为扇区号
 MOV DH, PYS-H
 MOV DL, DRIVE
 INT 13H
 POP CX
 POP DX
 RET
DISKIO ENDP

```

需要说明的是: 硬盘的柱面数通常大于 256, 8 位的 CH 不够用, 故借用 CH 的高 2 位。这样, 程序中磁道(柱面)号共 10 位; 高 2 位送 CL 的高 2 位, 低 8 位送 CH; 扇区号为 6 位, 送 CL 低 6 位。

## DOS 命令在 IBM-PC 机通信中的妙用

中科院广州电子所(510070) 林立  
水利部珠江水保局 林娜

介绍 PC 机通信的书刊很多, 一般是利用 PC 机的异步通信适配器与 PC 机、VAX 机、单片机等进行通信, 编写的通信程序大多数采用 BASIC 语言、汇编语言和 C 语言。下面介绍一种用 DOS 命令直接实现 PC 机通信的方法。

首先, 通信双方设置异步适配器参数应一致。例如, 设置异步通信口 COM1: 的传输速率为 1200BPS, 无校验, 8 位数据位, 1 位停止位。在 PC 机中, 用 MODE 命令设置异步适配器参数。

```
MODE COM1:1200,N,8,1
```

然后, 利用 DOS 中的 COPY 命令把异步通信适配器 COM1: 作为 I/O 设备进行通信。例如,

```

COPY COM1: FILENAME 接收文件
COPY COM1: CON: 控制台接收字符
COPY/B FILENAME COM1: 发送二进制文件
COPY CON: COM1: 控制台发送字符

```

注意: 在异种机进行通信时, 只可以从 PC 机异步通信口发送二进制文件, 不可以接收二进制文件; 而在 PC 机间进行通信时则无此限制。

可见, 利用 DOS 命令实现 PC 机近程通信, 简单、方便、可靠, 可推广应用。由于 DOS 命令没有验错、数据流同步、通信接口状态判别等措施, 对于远程通信不太可靠。

# 硬盘加锁小程序

陕西师范大学物理系(710062) 刘志存

随着个人计算机的广泛使用,特别是配有硬盘的微机越来越多。很多应用软件和数据都存放在硬盘中,硬盘容量大,使用人员杂,任何用户都可以不加限制地读写硬盘,难以确保文件的安全和保密。因此,对于硬盘宜加以保护,以禁止非法用户的使用。

这样做有如下好处:1. 避免他人使用硬盘时,不慎破坏其中存储的有用信息;2. 防止外来软盘上的病毒感染给硬盘;3. 防止他人查看盘上的保密信息。

本人编制了一小程序,可对硬盘进行加锁,使用极方便。硬盘加锁后,未经解锁,他人只能使用软盘启动系统,而不能使用硬盘。

用 EDLIN、MASM、LINK 对该程序进行编辑、汇编、链接后,再用 EXE2BIN.COM 将程序转化成扩展名为 .COM 的可执行程序(起名为 LOCK.COM),存入软盘中备用。需对硬盘加锁时,运行软盘上的加锁程序 LOCK.COM,则屏幕上显示:

The hard disk is unlocked! Lock it? (y/n)

键入“y”则程序执行完毕,硬盘被加锁,重新开机后硬盘不能引导系统。

下次使用硬盘时,先用系统软盘引导系统,然后运行 LOCK.COM,屏幕上显示:

The hard disk is locked! Unlock it? (y/n)

键入“y”程序执行完毕,硬盘被解锁,此时需重新启动系统才可使用硬盘。

程序在 PC/XT、0520、286 等系列机上运行通过。

```
code segment
 assume cs:code,ds:code,es:code
 mov ax,0201h
 mov bx,0200h
 mov cx,0001h
 mov dx,0080h
 int 13h
 jb q2
 mov si,03feh
 mov bp,0aa55h
 cmp [si],bp
 jz q1
 mov [si],bp
 mov dx,offset d1+100h
 jmp q3
q1: mov [si],ax
 mov dx,offset d2+100h
q3: mov ah,09h
 int 21h
```

```
 mov ah,00h
 int 16h
 cmp al,79h
 jz q4
 cmp al,6eh
 jz q5
 jmp q3
q4: mov ax,0301h
 mov bx,0200h
 mov cx,0001h
 mov dx,0080h
 int 13h
 jb q2
 jmp q5
q2: mov dx,offset d3+100h
 mov ah,09h
 int 21h
q5: int 20h
d1 db "The hard disk is locked!"
 db "Unlock it? (y/n)",13,10,"$"
d2 db "The hard disk is unlocked!"
 db "Lock it? (y/n)",13,10,"$"
d3 db "Error! I cannot read or"
 db "write the hard disk!",13,10,"$"
code ends
end
```

~~~~~

## 大量供应配电脑用显示器及电源

我部现货供应进口彩色/单色(黑白)高分辨率图形显示器,规格有 19"、14"、12"、9"等。并有美国产配电脑用大功率不间断电源,以及各种规格“日立”电缆和装配线供应。以上产品批发价将相当优惠,欢迎来函、来电或来人洽谈。

广东四会县南方电子厂经营部

地址:广东省四会县城高观东路 71 号

电话:(07663)322686

电挂:1311

邮编:526200

# 实用程序三则

甘肃庆阳长庆一中(745100) 任绥海

## 一、方便实用的服务程序

本程序虽然短小,功能却多。它可以建立文件,也可以显示已有的文本文件,还能修改任意一行的内容,并把它打印出来。

```
1 SCREEN 2;KEY OFF
2 SOUND 440,8;PRINT "Create or Display? (c/d)
3 CD$=INPUT$(1);PRINT
4 SOUND 440,8;INPUT "File name?";FL$
5 PRINT;SOUND 440,8;PRINT "Maximum number of
 line?";;INPUT";LAST
6 DIM A$(LAST)
7 IF CD$="d" OR CD$="D" THEN 16
8 CLS
9 FOR X=1 TO LAST
10 PRINT USING "# # #";X;PRINT " ";
11 SOUND 440,8
12 LINE INPUT A$(X)
13 IF A$(X)="." THEN 15
14 NEXT
15 PRINT;PRINT;PRINT;GOTO 25
16 CLS;OPEN FL$ FOR INPUT AS #1;PRINT
17 FOR X=1 TO LAST;IF EOF(1) THEN LAST=X;GOTO
 22
18 LINE INPUT #1,A$(X)
19 PRINT USING "# # #";X;PRINT " ";A$(X);A=
 CSRLIN
20 IF A MOD 24=0 THEN A2$=INPUT$(1);CLS
21 NEXT
22 CLOSE
23 SOUND 440,4;PRINT "Edit the text?";Y$=INPUT
 $(1)
24 IF Y$<>"y" AND Y$<>"Y" THEN 44 ELSE GOTO
 27
25 SOUND 440,4;PRINT "Edit the text?";Y$=INPUT
 $(1)
26 IF Y$<>"Y" AND Y$<>"y" THEN 32
27 SOUND 440,4;INPUT "Which line to edit?";Y1
28 IF Y1=0 THEN 32
29 PRINT SPC(4);A$(Y1);A=CSRLIN-1
30 LOCATE A,5;SOUND 440,4;LINE INPUT A$(Y1)
31 GOTO 27
32 CLS
33 FOR X=1 TO LAST
34 PRINT USING "# # #";X;PRINT " ";A$(X);A=
 CSRLIN
35 IF A MOD 24=0 THEN A2$=INPUT$(1);CLS
36 NEXT
37 SOUND 440,4;PRINT "Go on or End? (g/e)";A$=
 INPUT$(1)
38 IF A$="g" OR A$="G" THEN 27
39 OPEN FL$ FOR OUTPUT AS #1
40 FOR X=1 TO LAST
41 PRINT #1,A$(X)
42 NEXT
43 CLOSE
44 SOUND 440,4;PRINT "Lprint the text?";Y$=INPUT
 $(1)
45 IF Y$<>"y" AND Y$<>"Y" THEN GOTO 48
46 FOR X=1 TO LAST
47 LPRINT A$(X);NEXT
48 SOUND 440,4;PRINT "Go on or End? (g/e)";A$=
 INPUT$(1)
49 IF A$="g" OR A$="G" THEN RUN
```

## 二、字符居中打印程序

我们在设计文件封面或标题时,总要求将字符打印在纸张中部,每打印一次,都要换算字符串长度、纸张宽度、不同字型时的字符宽度等等,往往需要调试多次才能成功。本人编了一个短短的 BASIC 程序,大大简化了这一过程。实际操作时,只需将纸张的中部对准打印机的中部即可。

```
10 KEY OFF;CLS;COLOR 7,0
20 LOCATE 2,27;PRINT "字符居中打印程序"
30 LOCATE 3,27;PRINT "#####"
40 LOCATE 10,22;PRINT "
"
50 LOCATE 6,25;PRINT "请输入要打印的内容";SOUND
 440,4
60 LOCATE 8,COLOR 7,2;PRINT STRING$(80,32);LO
 CATE 8;LINE INPUT";B$
70 IF B$="." OR B$="*" THEN SOUND 800,8;COL
 OR 7,0 END
80 IF B$=" " THEN LPRINT CHR$(10);COLOR 7,0;
 GOTO 40
90 SOUND 400,4,COLOR 7,0
100 LOCATE 6,25;PRINT "
"
110 LOCATE 10,22;PRINT "请选择字型(A,B,C,D)";C$
 =INPUT$(1)
120 IF C$="A" OR C$="a" THEN PR$=CHR$(27)+
 "IA";A1=1;GOTO 170
130 IF C$="b" OR C$="B" THEN PR$=CHR$(27)+
 "IB";A1=2;GOTO 170
140 IF C$="c" OR C$="c" THEN PR$=CHR$(27)+
 "IC";A1=1;GOTO 170
150 IF C$="D" OR C$="d" THEN PR$=CHR$(27)
```

```

+“ID”;A1=2;GOTO 170
160 SOUND 800,8;GOTO 110
170 A1=50-LEN(B$)*A1/2
180 F$=STRING$(A1,“ ”)
190 LPRINT CHR$(27)+“IA”+F$+PR$+B$
200 GOTO 40

```

### 三、屏幕拷贝妙法

我们经常需要将屏幕上的显示信息输出到打印机上。虽然操作系统中有屏幕硬拷贝功能,可以在键盘上方便地实现,但它并不很实用。因为它会把屏幕上所有有用的和无用的信息全部打印出来。而下面的这个小程序,可以随意拷贝屏幕上任何指定区域的信息(区域由 H1-H2,L1-L2 指定)。

```

500 FOR I=H1 TO H2;’行数
600 FOR J=L1 TO L2;’列数
700 C=SCREEN(I,J)
800 A$=A$+CHR$(C)
900 NEXT
1000 LPRINT A$
1100 A$=“”
1200 NEXT

```

## 快而短的排序程序

山东菏泽县一中(274035) 许野平

本文给出了三个简化的基数排序程序。由于采用了基数排序的基本原理,排序效率当然在众多的算法中要算最高了。它所花费的时间与排序数据成正比,不会象其它算法那样随数据量增大所需时间急剧增加。在程序设计方面,本文提供的三个程序经过了巧妙的构思,因此整个排序过程只用短短的一行。

程序首先产生 1000 个随机数,然后按三种不同方式排序。程序假设这 1000 个数为学生成绩,sort01,sort02 两程序按学员号顺序输出,但前者允许并列名次,后者取消了并列名次。stor03 程序可按名次顺序依次输出学生成绩单。

在 IBM PC/XT 上实测,sort01 用 5 秒,sort02 用 7 秒,sort03 用 8 秒。如果你仔细地分析一下基数排序的原理并研究一下这三个小程序在设计时耍的小花招,一定会相信这样设计的算法是最短也是最快的。

```

10 REM sort01
20 S=1000
30 DIM M(S),H(100)
40 FOR I=1 TO S;M(I)=INT(RND(1)*100);NEXT I
50 PRINT “sorting……”;;TIME$=“00:00:00”
60 FOR I=1 TO S;H(M(I))=1;NEXT I
70 FOR I=100 TO 0 STEP -1;T=T+H(I);H(I)=T;
NEXT I
80 PRINT “OK! Time;”;TIME$

```

```

90 PRINT “No”,“Scores”,“Order No”
100 FOR I=1 TO S;PRINT I,M(I),H(M(I));NEXT I

```

```

10 REM sort02
20 S=1000
30 DIM M(S),H(100)
40 FOR I=1 TO S;M(I)=INT(RND(1)*100);NEXT I
50 PRINT “sorting……”;;TIME$=“00:00:00”
60 FOR I=1 TO S;H(M(I))=1+H(M(I));NEXT I
70 FOR I=100 TO 0 STEP -1;T=T+H(I);H(I)=T;
NEXT I
80 PRINT “OK Time;”;TIME$
90 PRINT “No”,“Scores”,“Order No”
100 FOR I=1 TO S;PRINT I,M(I),H(M(I));NEXT I

```

```

10 REM sort03
20 S=1000
30 DIM M(S),H(100)
40 FOR I=1 TO S;M(I)=INT(RND(1)*100);NEXT I
50 PRINT “sorting……”;;TIME$=“00:00:00”
60 FOR I=1 TO S;K=M(I);M(I)=H(K);H(K)=I;NEXT
I
70 PRINT “OK Time;”;TIME$
80 PRINT “Order No”,“No”,“Scores”
90 FOR I=100 TO 0 STEP -1
100 P=H(I)
110 IF P=0 THEN 140
120 Q=Q+1;PRINT Q,P,I
130 P=M(P);GOTO 110
140 NEXT I
150 END

```

.....

## 1724 打印机驱动程序行距的修改

武汉铁路分局电子中心(430071) 谭人杰

我国许多 PC 机用户配置了 M-1724 打印机,并利用 CCBIOS.1 下的 1724P.EXE 汉字驱动程序打印中文报表。为了使报表表格完全封闭,必须对 1724P.EXE 进行修改。

驱动程序 1724P.EXE 利用 M-1724 打印机行宽控制命令“ESC C n”控制走纸,n=6 表示标准行宽(每英寸 6 行),n=8 表示压缩行宽(每英寸 8 行),只要将 1724P.EXE 中的 ESC C 6 改为 ESC C 8,就可以压缩行距。具体修改方法见程序清单。

对于其他打印机的汉字驱动程序,也可以利用这种方法修改行距,首先查询打印机使用手册中的行宽控制命令,然后参照程序清单中的方法,用 DEBUG 对驱动程序进行修改。

```

C>COPY 1724P.exe 1724P
C>debug 1724P
-S0,fffe B0 06
1F39;2788
1F39;2AAF
-u1f39;2aa0
1F39;2AA0 06 PUSH ES
1F39;2AA1 0900 OR [BX+SI],AX
1F39;2AA3 0000 ADD [BX+SI],AL
1F39;2AA5 B01B MOV AL,1B
1F39;2AA7 E8ADFB CALL 2657
1F39;2AAA B043 MOV AL,43
1F39;2AAC E8A8FB CALL 2657

```

```

1F39;2AAF B006 MOV AL,06
1F39;2AB1 E8A3FB CALL 2657
1F39;2AB4 B00A MOV AL,0A
1F39;2AB6 E89EFB CALL 2657
1F39;2AB9 B80000 MOV AX,0000
1F39;2ABC B90011 MOV CX,1100
1F39;2ABF BFE200 MOV DI,00E2
-e2ab0
1F39;2AB0 06.08
-w
Writing 2C00 bytes
-q
C>copy 1724p 1724p.exe

```

## 加密 WPS 文件的解密

河南郑州中原制药厂设备处(450066) 谷高平 孟建伟

目前,许多微机都配有 Super-WPS 文字处理系统,用该系统的 WPS.COM 软件进行文字处理相当方便,而且可对由其编辑的文件进行加密处理,使文件的安全性得到一定的保证。但很多操作人员在给自己的文件加密后却忘记了密码,在备份文件已被破坏的情况下,对再次打开文件束手无策。经过探索,笔者找到了破解加密 WPS 文件的方法,简述如下:

WPS 文件的文件头由 1024 个字节组成,占两个扇区。被加密的 WPS 文件,密码经 WPS.COM 软件转换后被置于该文件文件头第二相关扇区、偏移量为 221~228 这 8 个字节中,未被加密的 WPS 文件,这 8 个字节的内容为 00。如果读出这 8 个字节的内容,再按本文附表即可查出密码的原形。例如:一个名为 GGP.WPS 的加密文件的解密过程如下:

用 PCTools 软件在 File Functions 菜单下对 GGP.WPS 文件进行 Edit 操作,使当前屏幕置于下图状态。查看偏移量 221~228 八个字节可发现,其内容为:

8B 0B 8A 00 00 00 00

可见,密码为三位。查本文附表可知密码为: GOW。

File=GGP.WPS

Relative sector 0000001,Clust 02244,Disk Abs Sec 0009130

Disp.....Hex codes.....

000(00) 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00

016(10) 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00

.....

.....

208(D0) 00 00 00 00 00 00 00 00-00 1A 0C 00 00 8B 0B 8A

224(E0) 00 00 00 00 00 00 E0 01-D0 01 C0 01 B0 01 A0 01

240(F0) 90 01 80 01 70 01 00 CE-F7 C9 BD A3 D7 A3 D0 A3

附表:

	1	2	3	4	5	6	7	8
F1~F12	FF	EF	DF	CF	BF	AF	9F	8F
	7F	6F	5F	4F	3F	2F	1F	0F
	FE	EE	DE	CE	BE	AE	9E	8E
ESC	7E	6E	5E	4E	3E	2E	1E	0E
空! " # \$ % & ' ( ) * + , - . /	FD	ED	DD	CD	BD	AD	9D	8D
0 1 2 3 4 5 6 7	7D	6D	5D	4D	3D	2D	1D	0D
8 9 : ; ( = ) ?	FC	EC	DC	CC	BC	AC	9C	8C
@ A B C D E F G	7C	6C	5C	4C	3C	2C	1C	0C
H I J K L M N O	FB	EB	DB	CB	BB	AB	9B	8B
P Q R S T U V W	7B	6B	5B	4B	3B	2B	1B	0B
X Y Z [ \ ] ^ _	FA	EA	DA	CA	BA	AA	9A	8A
{   } ~	7A	6A	5A	4A	3A	2A	1A	0A
	F9	E9	D9	C9	B9	A9	99	89
	79	69	59	49	39	29	19	09
	F8	E8	D8	C8	B8	A8	98	88
	78	68	58	48	38	28	18	08

# 电脑病毒入侵报警程序

苏州光明丝织厂微机室(215001) 陶秋刚

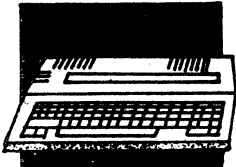
电脑病毒的防不胜防已使人深恶痛绝,在这里笔者向广大用户介绍一种实用的电脑病毒入侵报警程序,使用户能马上发现病毒的入侵并采取相应的措施。

纵观所有文件类电脑病毒,不难发现,这些病毒程序有一个共同特点即它们均是采用修改 DOS 的中断来实现的。于是我们可以这样设想,只要将原先标准的(未被病毒程序修改过的)中断向量表保存起来,并不时同当前正使用的中断向量表作比较,如发现不同便可认为已有病毒入侵了。PC 机的 08H 号中断是一个计时器脉冲中断,每秒产生 18.2 次中断,利用该中断每秒发生 18.2 次的特点不断对新旧向量表进行比较,就起到病毒入侵的发现作用。

笔者基于以上思路设计了一个汇编程序,并在 PC 机上调试通过。该程序能起到对所有文件类病毒入侵的报警功能,如发现病毒的入侵则马上会在屏幕左上角永久显示报警信息。报警信息的显示采用直接写显示缓冲区来实现。在该程序中报警信息为一空白方框。

```
CODE SEGMENT
 ASSUME CS, CODE, DS, CODE, SS, CODE
 ORG 100H
START: PUSH CS
 MOV AH, 09
 MOV DX, OFFSET X1
 INT 21H
 MOV AX, 3508H
 INT 21H ;取原 08H 中断向量
 PUSH ES
 POP DS
 MOV DX, BX
 MOV AX, 256DH
 INT 21H ;将原 08H 中断置为 6DH 中断
 PUSH CS
 POP DS
 MOV DX, OFFSET DATA2
 MOV AX, 2508H
 INT 21H ;置现 08H 中断向量
 MOV CX, 68H ;取标准中断向量表
 XOR AX, AX
 MOV SI, AX
 POP ES
 MOV DS, AX
 MOV DI, OFFSET DATA1
 REP MOVSB
 PUSH CS
```

```
 POP DS
 MOV DX, OFFSET DATAEND
 ADD DX, 300H
 INT 27H ;驻留退出
X1: DB '病毒报警程序!', 0AH, 0DH
 DB '苏州光明丝织厂微机室', 0AH, 0DH
 DB ' 1991.10.1', '$'
DATA1: DB 110 DUP(?), 标准中断向量表
DATA2: CLD ;新的 08H 中断入口
 PUSH DS
 PUSH ES
 PUSH AX
 PUSH BX
 PUSH CX
 PUSH DI
 PUSH SI
 PUSH CS
 POP DS
 XOR AX, AX
 MOV ES, AX
 MOV DI, AX
 MOV SI, OFFSET DATA1
 MOV CX, 64H
 REP CMPSB ;现中断向量表同标准中断向量表比较
 JZ NEXT1
 MOV AX, 0B800H ;与标准向量表不同,报警处理
 MOV ES, AX
 MOV DI, 0
 MOV SI, OFFSET DATA
 MOV CX, 20H
LOOP1: PUSH CX
 PUSH DI
 TEST CX, 01
 JZ NEXT2
 ADD DI, 2000H
NEXT2: MOV CX, 8H
 REP MOVSB
 POP DI
 POP CX
 TEST CX, 01
 JZ X2
 ADD DI, 80
X2: LOOP LOOP1
NEXT1: POP SI
 POP DI
 POP CX
 POP BX
 POP AX
 POP ES
 POP DS
 STI
 INT 6DH ;执行原 08H 中断
DATAEND: IRET
DATA: DB 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH
 DB 30 DUP(0C0H, 00H, 00H, 00H, 00H, 00H, 00H, 00H, 03H)
 DB 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH, 0FFH
CODE ENDS
END START
```



## 学习机之友

# 对几种高精度数值计算方法的改进

四川省南充一中(637040) 陈庆祥

我们知道,由于受字长和数据存储方式的限制,计算机输出数值的有效位数只能是固定的几位或十多位(因变量类型而异)。为了获得较长数字运算的准确结果,必须研究高精度的数值运算方法。目前,有关刊物上曾刊登过一些这方面的程序,但这些程序在算法上几乎全部模拟人工笔算,即利用数组来逐位存放多位数的每一位数字和逐位计算的方法。这样尽管也能达到目的。其缺点是处理效率偏低、占用内存空间大,能够处理的数值范围也比较小,可以说这种算法难以发挥计算机的潜力。例如电子与电脑 1989 年第 6 期《一个可以精确求解幂运算结果的程序》,1990 年 10 期《任意两整数的高精度乘法》及 1991 年 4 期《从重赏谈起》,笔者曾对各程序作过试验。第一篇文章中的程序计算  $14^{23}$  用 55 秒(在 Apple I 上运行,下同),计算  $50^{23}$  时便发生溢出错误,事实上,它仅能计算结果不超过 40 位的幂。第二篇文章的程序在计算 13 位数与 11 位数的乘积时用 12 秒,计算 30 位与 21 位数的乘积时用 57 秒,计算两个 100 位数的乘积要用近两小时。第三篇文章的程序在计算麦粒总数  $\sum_{i=0}^{63} 2^i$  时用了 1 分半钟,比笔算快不了多少。

其实,在处理多位数的乘法、幂与阶乘等运算时,我们完全可以采取将数字分段存储和运算的新方法,充分利用整型和实型变量的最多有效位数,这样不仅可以显著地提高处理效率,使计算的耗时量减少为逐位计算的几十分之一,还可以大大减少内存空间的占用,扩大数据处理的范围。

本文后面是笔者用上述思想设计的两个高精度数值运算程序,分别用来计算乘积与幂。容易看出,这两个程序比前面提到的三篇文章中的程序简练得多,但是其处理效率却高出几十倍。而占用内存却大为减少,数值处理范围也相应扩大。如程序 1 计算两个 20 以下的整数乘积仅用零点几秒,两个 30 位数的乘积用 2—3 秒,计算两个 100 位整数的积也不过用 30 秒。程序 2 求  $14^{23}$  时用 2 秒,计算  $9876^{23}$  时用 8 秒,求  $1991^{100}$  的 330 位数也仅用 90 秒。该程序在西文状态下可以计算近 7 万位的数字结果。

这两个程序的算法主要技巧是在数字分段处理上。程序 1 采用字符串补长分段的方法(即凑成 4 的倍数);程序 2 对各次中间计算结果的数字分段比较困难一些,因为其长度是变化的,这里采用了利用对数测出各次积的动态长度,然后再分段存储,计算的方法。用一个整型数组元素存放四位数字,每次具体计算的数

字便不会超过九位。保证了数据计算的准确性。

最后要提到的是算法上的另一种变通处理。上述第三篇文章中求  $\sum_{i=0}^{63} 2^i$  的问题,因涉及到高精度的幂运算与加法运算,故算法复杂,程序很长且运行费时,其实,用等比数列的求和公式容易推知: $\sum_{i=0}^n 2^i = 2^{n+1} - 1$ ,即只需求  $2^4 - 1$  即可。这样一来。只要将程序 2 的 70 语句略加改动,便可迅速求出  $\sum_{i=0}^{63} 2^i$  或  $\sum_{i=0}^{73} 2^i$ ,耗时不过 2 秒左右。如果再变通一下,求  $256^8 - 1$ ,因为  $2^4 - 1 = (2^2)^2 - 1 = 256^2 - 1$ ,输入 256 与 8 后立即可以打印出准确的结果。由此不难看出,灵活地处理算法对于提高程序质量有着非常重要的意义。

程序 1

```

10 REM 正整数的快速高精度乘法程序
20 INPUT "被乘数,乘数=";A$(1),A$(2);L(1)=INT
 ((LEN(A$(1))+3)/4);L(2)=INT((LEN(A$(2))
 +3)/4)
25 IF A$(1)="0" THEN END
27 CALL-198;PRINT"乘积=";
30 S=L(1)+L(2);W=(S+ABS(L(1)-L(2)))/2;DIM
 A%(2,W),B%(S);E=10000
40 FOR I=1 TO 2;A$(I)=RIGHT$("000"+A$(I),L
 (I)*4)
50 FOR J=1 TO L(1)
60 A%(I,J)=VAL(MID$(A$(I),4*(L(1)-J)+1,4))
70 NEXT;NEXT
80 FOR I=1 TO L(1);H=0
90 FOR J=1 TO L(2);K=I+J-1
100 B=B%(K)+A%(I,I)*A%(2,J)+H
110 H=INT(B/E);B%(K)=B-H*E
120 NEXT;B%(K+1)=H;NEXT
130 K=K-(B%(K+1)=0);PRINT B%(K+1);
140 FOR I=K TO 1 STEP -1;PRINT B%(I);NEXT
150 CALL-198;PRINT;RUN

```

被乘数,乘数=8521476391472,58735419635  
乘积=500512491762853455352720  
被乘数,乘数=369258147369258147147258369147,  
147258369963852741123  
乘积=543763528774697301272742665307220307963856  
1332081  
被乘数,乘数=0,0

程序 2

```

5. REM 快速求解幂运算精确结果的 BASIC 程序
10 INPUT "A,N=";A,N;IF A<>INT(A)OR A<=0

```

```

OR N<=0 THEN END
20 W= LOG(A)/LOG(10);DS%=N*W/4+1;DIM X%(
 DS%)
30 X%(1)=A;B=1E4;F=4;C=W+F;L$="000"
40 FOR Y=2 TO N;C=C+W
50 FOR I=1 TO C/F;D=A*X%(I)+E%
60 E%=D/B;X%(I)=D-B*E%;NEXT;NEXT
70 PRINT A;"^";N;"=";X%(I-1);
80 IF I>2 THEN FOR J=I-2 TO 1 STEP -1:PRINT RIGH
 T$(L$+STR$(X%(J)),4);:NEXT
90 CALL-198;PRINT;RUN
]RUN
A,N=14,32
14^23=229585692886981495482220544
A,N=9876,32

```

```

9876^32=6708023885067056779126280484374086768
00621039896345384011301752850609595990813326500520305
68130359850101425909428305449373925376
A,N=0,0
修改70语句后计算《从“重赏”谈起》中的麦粒数,用5
秒左右或0.5秒即求出。
]70 X%(I)=X%(I)-1;PRINT "S=";X%(I-1);
]RUN
A,N=2,64
S=18446744073709551615
A,N=2,73
S=9444732965739290427391
A,N=256,8
S=18446744073709551615
A,N=0,0

```

~~~~~

## CEC-I 自造字键盘辅助点阵生成程序

襄樊铁路运输技校(441001) 汤永进

笔者编写了一个工具程序,可象键盘绘图一样方便地生成字模点阵。该程序的主要变量和处理过程简介如下:

A0 为字库存放首址,可按所用调字程序的参数要求置为用户内存区中任一地址(这里配合91年6期博文置为\$7000);A 为当前所造字的点阵数据存放首址,每字数据占32个字节。L0 为字库总长度,可按需要在内存开销许可的范围内随意增减;INT(L0/32)为该字库最多可容字数。

A 数组存放 16×16 点阵的映象(0 或 1),其中左下标 1~16 为点阵行序,中下标 0~1 为每行左右组别(一行 16 个点要分左右两组各存 1 字节),右下标 0~7 为每组列序。N 数组存放 32 字节的点阵数据码(0~255),其中奇数下标元素顺次存左半点阵数据,偶数下标元素顺次存右半点阵数据。

程序 10 行读入已建立的字库(如无则下滑运行),20~45 行印出标题并在屏幕中间提供一有 16×16 格位的方框,50~70 行建立系统变量;80 行按提示输入所造字的代码后,90 行算出该字首址;100 行将对应内存中可能存在的原造旧字的点阵数据读入 N 数组,110~180 行将这些数据解拆分送 A 数组,并在方框中显映出字模。

190 行设定各绘图中间变量的初始值,200~290 行进入键盘辅助点阵描绘,将各点描绘结果随时存录于 A 数组对应元素。描绘时按四个箭头键在方框中移动光点,按空格键决定下笔还是擦除(同软汉字的造字操作相似)。造字过程中若欲放弃该字重新择码另造,可按 ESC 键;若只擦光底板重造该字则可按 TAB 键。字造好后按回车键,300~330 行便将 A 数组中数据归并到 N 数组中去,340 行将其写入内存,接着 350~

370 行决定是否继续造字,若结束则 380 行将内存字库全部存盘。

子程序 500~520 行行使“黑板擦”功能,负责清画框和清数组;600~620 行作为“笔尖”,负责绘点。

```

5 ONERR GOTO 20
10 PRINT CHR$(4)"BLOAD CEC-ZIKU"
20 PRINT CHR$(4)"PR#3";PRINT
30 HGR2;VTAB 2;HTAB 12;PRINT"CEC-I 造字";HTAB 9;
 PRINT"字模点阵生成系统"
40 HCOLOR=3;HPLOT 94,60 TO 160,60 TO 160,126 TO
 94,126 TO 94,60
45 HPLOT 127,61;HPLOT 95,93;HPLOT 159,93;HPLOT
 127,125
50 A0=28672;L0=4096;REM.$7000&.$1000
60 G$=CHR$(7)+CHR$(7);C$=CHR$(11)
70 GOSUB 500;PRINT G$;
80 VTAB 9;HTAB 8;PRINT C$"该字代码(1-";INT(L0/
 32);:INPUT"(";N
90 A=A0+(N-1)*32
100 FOR I=1 TO 32;N(I)=PEEK(A+I-1);NEXT
110 HCOLOR=3;Y=59
120 FOR I=1 TO 16;L=I*2-1
125 X=93;Y=Y+4
130 FOR J=0 TO 1;N%=N(L+J)
140 FOR K=0 TO 7;A%=N%/2
145 A(I,J,K)=N%-A%*2
150 IF A% THEN N%=A%;NEXT
160 FOR K=7 TO 0 STEP -1;X=X+4
170 ON A(I,J,K)=1 GOSUB 600;NEXT
180 NEXT J,I;PRINT G$;
190 X=97;Y=63;X0=97;Y0=63;H=1;V=1;C=0

```

(转第 20 页)



# 二维函数曲线拟合

上海交通大学 90111 班(200240) 李 涛

我们在有关数据采集和处理的过程中,经常遇到二维函数之间求解  $y=f(x)$  的问题,本文是根据样本数据求解一元高次方程:

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots + a_nx^n;$$

从而得出二维平面中任意曲线的函数关系式。本程序适合于大批量数据处理,这里只介绍程序使用方法。

程序 1 是统计计算程序。首先输入所需回归次数,然后键入样本数据文件的文件名。该数据文件由程序 2 生成。之后计算机计算出各项系数。按任一健后进入主菜单,对计算出来的函数式作进一步处理。共有 4 项选择:(1)数值计算。依照所得函数式对某一  $x$  值求解  $y$  值,这可用于对未知参数预测。(2)显示函数曲线。首先在提示下输入  $x, y$  坐标的作图范围,程序会依照该数据计算出屏幕上每一点所代表的  $x, y$  坐标宽度,并显示出来。此举目的在于进行定量观测。计算机在规定的范围内绘出曲线后,还会将该范围内的原始离散数据点显示在屏幕上,以供比较拟合程度。(3)调入另一批数据。(4)显示拟合出的各项系数。另外,在主菜单下按 H 键会进入高分辨率显示状态,以便随时观察已经绘好的曲线;按 T 键则回到文本状态。

程序 2 是数据文件生成程序。它可将输入的样本资料生成一个以 .DAT 为后缀的 T 型文件,供程序 1 调用。它还可对数据文件进行增加或删除等有关处理。该程序运行每一步均有提示。

程序 1

```
5 DIM A(20,20),B(40),C(20),X(20),Y(20)
6 D$ = CHR$(4)
10 TEXT;HOME;VTAB 10;INPUT"WQZ:";W;N=W+1;S=1;P=0
15 PRINT;INPUT"Input file name:";A$
17 PRINT D$;"OPEN";A$;".DAT"
18 PRINT D$;"READ";A$;".DAT"
19 INPUT X;INPUT Y;IF X=0 AND Y=0 THEN 42
20 FOR I=1 TO W*2;B(I)=B(I)+X^I;NEXT I
30 FOR J=0 TO W;C(J)=C(J)+Y*X^J;NEXT J
32 X(S)=X;Y(S)=Y
35 PRINT "x";S;"=";X;" y";S;"=";Y
36 S=S+1;GOTO 19
42 PRINT D$;"CLOSE";A$;".DAT"
43 GET A$;X(S)=0;Y(S)=0
45 HOME;VTAB 6;HTAB 4;PRINT"Calculating..."
50 FOR I=0 TO W
60 FOR J=0 TO W;A(I,J)=B(J+P);NEXT J
70 P=P+1;NEXT I
80 A(0,0)=S
90 FOR I=0 TO W;A(I,N)=C(I);NEXT I
200 FOR I=0 TO W;Q=I;P=1;E=A(I,1)
```

```
210 FOR J=0 TO W
220 IF J=1 OR A(J,Q)=0 THEN 280
240 R=A(J,Q)/A(I,Q)
250 FOR K=0 TO N;A(J,K)=A(J,K)-A(I,K)*R;
NEXT K
280 NEXT J;A(N,1)=Q
300 NEXT I
310 FOR I=0 TO W;Q=A(N,I);A(N,I)=A(I,N)/A(I,Q);PRINT "a";I;"=";A(N,I);NEXT
311 PRINT;PRINT"Hit any key to continue...";GET A$
312 HOME;VTAB 6
313 PRINT "(1) CALCULATE"
314 PRINT"(2) SHOW CURVE"
315 PRINT "(3) INPUT NEW DATA"
317 PRINT "(4) SHOW COEF FICIENTS"
320 GET A$;IF A$="2" THEN 400
321 IF A$="4" THEN 600
322 IF A$="1" THEN 345
324 IF A$="3" THEN RUN
325 IF A$="H" THEN POKE-16304,0;POKE-16299,0;POKE-16297,0
326 IF A$="T" THEN TEXT;GOTO 312
345 INPUT"x:";X;Y=0
350 GOSUB 500
360 PRINT "y=";Y
370 PRINT"Continue? (Y/N)";;GET A$
375 PRINT;PRINT
380 IF A$="N" THEN 312
390 GOTO 345
400 HOME;VTAB 4;PRINT "Plotting;"
410 PRINT;PRINT;PRINT "Range"
420 INPUT"Xmin:";XN;INPUT "max:";XM;XS=(XM-XN)/279
422 PRINT "X scale:";XS
424 INPUT"Ymin:";YN;INPUT "max:";YM;YS=(YM-YN)/191
426 PRINT"Y scale:";YS
428 GET A$;HGR2;HCOLOR=3
430 FOR I=XN TO XM STEP XS;Y=0;X=1;GOSUB 500
432 IF Y<YN OR Y>YM THEN 436
434 HPLOT (X-XN)/XS,191-(Y-YN)/YS
436 NEXT
438 S=1
439 IF X(S)=0 AND Y(S)=0 THEN 446
440 IF X(S)<XN OR X(S)>XM THEN 444
441 IF Y(S)<YN OR Y(S)>YM THEN 444
442 HPLOT (X(S)-XN)/XS,191-(Y(S)-YN)/YS
443 MUSIC 88,15
444 S=S+1;GOTO 439
```

```

446 MUSIC 128,45
450 GET A $;TEXT;HOME;GOTO 312
500 FOR K=0 TO W;Y=Y+A(N,K)*X^K;NEXT
510 RETURN
600 TEXT;VTAB 6;FOR K=0 TO W;PRINT "a";K="";A
(N,K);NEXT
610 GET A $;GOTO 312

```

程序 2

```

10 D $ =CHR $(4);DIM X(100),Y(100)
15 HOME
20 FOR I=1 TO 5;READ V,H,A $;VTAB V;HTAB H;
PRINT A $;NEXT
30 DATA 4,6,"DATA EDITOR VER 1.0"
40 DATA 6,4,"(1) INPUT NEW DATA"
50 DATA 8,4,"(2) APPEND DATA"
60 DATA 10,4,"(3) DELETE DATA"
70 DATA 12,4,"(4) QUIT"
100 GET A $;IF A $ ="1" THEN K=0;GOTO 200
110 IF A $ ="2" THEN K=1;GOTO 200
115 IF A $ ="3" THEN 400
120 IF A $ ="4" THEN END
130 GOTO 100
200 HOME;VTAB 6;INPUT "Input data file name:";A $

```

```

210 HOME;VTAB 3;HTAB 4;PRINT "Inputing data...";
PRINT
220 HTAB 4;PRINT "When over,hit<A>key..."
230 S=1;PRINT
232 PRINT"x";S;" ";:INPUT X(S)
234 PRINT"y";S;" ";:INPUT Y(S)
236 PRINT"x=";X(S);"y=";Y(S)
238 PRINT"Are you sure? (Y. N. A)"
239 GET B $
240 IF B $ ="N" THEN 232
242 IF B $ ="Y" THEN S=S+1;GOTO 232
244 IF B $ ="A" THEN 250
245 GOTO 239
250 HOME;VTAB 6;HTAB 6;PRINT"Saving to disk..."
260 PRINT D $;"OPEN";A $;". DAT"
262 IF K=1 THEN PRINT D $;"APPEND";A $;". DAT"
265 PRINT D $;"WRITE ";A $;". DAT"
270 FOR I=1 TO S;PRINT X(I);PRINT Y(I);NEXT
273 PRINT 0;PRINT 0
275 PRINT D $;"CLOSE";A $;". DAT"
280 RUN
400 HOME;VTAB 6;INPUT "Input file name :";A $
410 PRINT D $;"DELETE";A $;". DAT"
420 RUN

```

## 在 Applesoft 系统下实现 COMMON 功能

临汾山西师大实验中学(041000) 安赵根

在 CP/M 操作系统支持下的 MBASIC 语言有一个 COMMON 语句,在连接程序时与 CHAIN 程序连用,把指定数组的值传送给被连接的程序使用。而 Applesoft 系统却不具备这种功能(它只能把全部数组都传送下去),这对于被连接的程序来说是对内存空间的极大浪费。

为此笔者根据 SOFTBASIC 变量表的特点编写了一段汇编语言子程序,用于对不需传送的数组从变量表中清除,在 APPLESOFT 系统下实现了与 COMMON 类似的功能。程序采用了相对寻址方式,可调入内存任何位置以便于使用。对一个数组进行清除后,变量表中便没有了这个数组,它所占用的存储单元被后面的数组使用,变量表最后一部分单元被释放。

当某数组不需传送而进行清除时,可先用 POKE 语句将数组类型代码(实型为 0,整型为 % 的 ASC 码、字符型为 \$ 的 ASC 码)送入 253 单元;将数组名首字符的 ASC 码送入 254 单元,将数组名次字符的 ASC 码送入 255 单元(若名只有一个字母,则将 0 送入 255 单元),然后用 CALL 语句调用该汇编子程序即可。如,欲清除数 A \$(20),则可用以下语句实现(子程序首址为 \$9000)。

```

100. POKE 253,ASC("$");POKE 254,ASC("A");POKE
255,0;CALL 9×16^3
9000- A5 FD F0 13 38 E9 24 F0
9008- 07 18 A5 FE 69 80 85 FE
9010- 18 A5 FF 69 80 85 FF A0
9018- 02 A9 00 91 6D C8 91 6D
9020- A5 6B 85 F9 A5 6C 85 FA
9028- A0 00 A5 FE 38 F1 F9 D0
9030- 0A C8 A5 FF 38 F1 F9 D0
9038- 02 F0 18 A0 02 18 B1 F9
9040- 65 F9 85 FD C8 B1 F9 65
9048- FA 85 FA A5 FD 85 F9 A9
9050- 00 F0 D5 A0 02 18 B1 F9
9058- 65 F9 85 FB C8 B1 F9 65
9060- FA 85 FC A5 6D 85 FD A5
9068- 6E 85 FE A0 00 A9 00 91
9070- F9 C8 91 F9 A0 00 B1 FB
9078- 91 F9 E6 F9 D0 02 E6 FA
9080- A5 FB C5 FD A5 FC E5 FE
9088- E6 FB D0 02 E6 FC 90 E4
9090- A5 F9 85 6D A5 FA 85 6E
9098- A5 6D F0 03 C6 6D 60 A9
90A0- FF 85 6D C6 6E 60

```

# 通用打印课表程序

山东青岛九中微机室(266011) 程建伟

本程序是利用中华学习机的汉字与表格处理功能,打印出样式美观,形式多样的课表,且可大可小,使用方便。程序中的操作也非常简单,每一步都有善意的提示,等待您键入各课目的代码,如下所示:

A-政治      B-语文      C-代数  
D-几何      E-英语      F-物理  
G-化学      H-历史      I-地理  
J-体育      K-生理      L-空白  
M-数学      N-语音      O-生物

(用户欲增改课目,只需修改第20和340句)

输入课目代码时,每天占一行,六行输入完后,程序将自动存档,以备下次使用。程序输出可任选,如改变第130句1659单元的值,可改变其大小,将第520句中的变量I与J交换,可输出纵向排列的课表等等。

```
10 INPUT "请问用原记录还是输入新纪录(A/B)";A$
20 DIM B$(6),C$(6,6),N$(15)
30 D$=CHR$(13)+CHR$(4)
40 FOR I=1 TO 6:READ B$(I):NEXT;FOR I=1 TO
15:READ N$(I):NEXT
50 IF A$="A" THEN GOSUB 600
60 IF A$="B" THEN GOSUB 700
70 IF A$ <> "A" AND A$ <> "B" THEN RUN
80 INPUT "打印吗(Y/N)";R$
90 IF R$="Y" THEN 110
100 GOTO 170
110 PRINT "准备好了吗,按任意键开始"
120 GET K$;GET K$;PRINT "已经开始了,请等候"
130 POKE 1659,2;POKE 1687,0
150 POKE 1915,0;POKE 2043,40
170 PRINT SPC(23);"课程表";SPC(12);"92.9"
180 PRINT SPC(7);"
190 PRINT SPC(7);"
200 FOR I=1 TO 6
210 GOSUB 400
220 GOSUB 500
230 NEXT
240 PRINT SPC(7);"
250 PRINT SPC(18);"书山有路勤为径,学海无边苦作
舟"
260 PRINT;PRINT;PRINT;PRINT
270 POKE 1659,0
280 INPUT "还打印吗(Y/N)";R$
290 IF R$="Y" THEN 110
300 PRINT "谢谢合作,再见!"
310 END
320 DATA "一","二","三","四","五","六"
```

```
330 DATA "政治","语文","代数","几何","英语","物
理","化学","历史","地理","体育","生理","
340 DATA "数学","语音","生物"
400 PRINT SPC(7);"
410 RETURN
500 PRINT SPC(7);"|";B$(I);"|";
510 FOR J=1 TO 6
520 PRINT C$(I,J);"|";
530 NEXT;PRINT
540 RETURN
600 INPUT "请输入班级";AA
610 PRINT D$;"OPEN L.TAB-";AA
620 PRINT D$;"READ L.TAB-";AA
630 FOR I=1 TO 6;FOR J=1 TO 6
640 INPUT X$
650 C$(I,J)=X$
660 IF X$="" THEN C$(I,J)=" "
670 NEXT;NEXT
680 PRINT D$;"CLOSE"
690 RETURN
700 DIM L$(6)
710 PRINT "请输入课程代号"
720 FOR I=1 TO 6
730 PRINT "星期";B$(I);:INPUT L$(I)
740 NEXT
750 FOR I=1 TO 6;FOR J=1 TO 6
760 C$(I,J)=N$(ASC(MID$(L$(I),J,1))-64)
770 NEXT;NEXT
780 INPUT "请输入班级";AA
790 PRINT "请放入数据盘,按任意键"
800 GET K$;GET K$
810 PRINT "请稍候..."
820 PRINT D$;"OPEN L.TAB-";AA
830 PRINT D$;"WRITE L.TAB-";AA
840 FOR I=1 TO 6;FOR J=1 TO 6
850 PRINT C$(I,J)
860 NEXT;NEXT
870 PRINT D$;"CLOSE"
880 RETURN
```

```
]RUN
请问用原记录还是输入新纪录(A/B)B
请输入课程代号
星期一? BJHMEI
星期二? MEFAGI
星期三? BFMEBB
星期四? GJMNL
```

星期五? EBAIFM

星期六? FMGBHE ▾

请输入班级 44

请放入数据盘,按任意键

请稍候...

打印吗(Y/N)Y

准备好了吗,按任意键开始

已经开始了,请稍候

|   | 1  | 2  | 3  | 4  | 5  | 6  |
|---|----|----|----|----|----|----|
| 一 | 语文 | 体育 | 历史 | 数学 | 英语 | 地理 |
| 二 | 数学 | 英语 | 物理 | 政治 | 化学 | 地理 |
| 三 | 语文 | 物理 | 数学 | 英语 | 语文 | 语文 |
| 四 | 化学 | 体育 | 数学 | 语文 |    |    |
| 五 | 英语 | 语文 | 政治 | 地理 | 物理 | 数学 |
| 六 | 物理 | 数学 | 化学 | 语文 | 历史 | 英语 |

书山有路勤为径,学海无边苦作舟

## CP/M 系统文件的恢复方法

四川绵阳南山中学(621000) 李 齐

当你辛辛苦苦在 CP/M 操作系统下编制的实用程序或是生成的重要数据文件遭到了意外删除,一定十分痛心。不过只要你的文件删除后该盘没有存入新的文件,在读完本文后,你就等于具有了使文件起死回生的能力。

用 NIBBLES AWAY 等软件的扇区编辑功能将被删除文件所在盘的 03 道读出,可以清楚的看到 CP/M 的目录结构。

CP/M 的目录区依次安排在 03 道的 00、03、06、09、0C、0F 扇区,每个目录占连续的 32 个字节,00 字节为文件所在用户区号,当一个文件被删除后,其值被置为 E5H。

01—08 字节是文件主名,如不足 8 字节,未使用

部分则填以 20H(空格)

09—11 字节为文件扩充名。

13—14 字节未用,值为 00H。

15 字节是文件长度,以记录为单位。

16—32 字节表明该文件在磁盘中的位置,未使用的部分值为 00H。

由上可知,要恢复一个被删除的文件,只需将 03 道读出,找到该文件的索引,将其 00 字节的 E5H 改为该文件所在的用户区号,(一般可改为 00),再写回 03 道,即可大功告成。

这时回到 CP/M 操作系统。哈!,失掉的文件又回来了!

(接第 16 页)

```

200 HCOLOR=3:HPLLOT X,Y
210 P=PEEK(-16384);IF P<128 THEN 210
215 POKE-16368,0:P=P-128
220 HCOLOR=0:HPLLOT X0,Y0
230 ON P=9 GOSUB 500
240 ON P=13 GOTO 300;ON P=27 GOTO 70;ON P=32
 GOTO 280
250 XX=(P=21)*(H<16)-(P=8)*(H>1)
255 X=X0+XX*4;H=H+XX
260 YY=(P=10)*(V<16)-(P=11)*(V>1)
265 Y=Y0+YY*4;V=V+YY
270 X0=X;Y0=Y;GOTO 200
280 T=H>8;U=8+8*T-H
285 C=ABS(A(V,T,U)-1);A(V,T,U)=C
290 HCOLOR=C*3;GOSUB 600;GOTO 200
300 FOR I=1 TO 16:L=I*2-1
310 FOR J=0 TO 1:L=L+J;N(L)=0;E=0.5

```

```

320 FOR K=0 TO 7;E=E*2;N(L)=N(L)+A(I,J,K)*E
330 NEXT;NEXT;X=PEEK(49200);NEXT
340 FOR I=1 TO 32;POKE A+I-1,N(I);NEXT
350 VTAB 9;PRINT TAB(9)"继续造字?(Y/N)"CG;
360 GET X$;IF X$="Y" THEN 70
370 IF X$<>"N"THEN 360
380 PRINT X$;PRINT CHR$(4)"BSAVE CEC-ZIKU,A"
 A0;"",L"LO
400 END
500 HCOLOR=0;FOR I=96 TO 158:HPLLOT I,62 TO I,
 124;NEXT
510 POKE 109,PEEK(107);POKE 110,PEEK(108);DIM A
 (16,1,7),N(32)
520 RETURN
600 FOR II=X-1 TO X+1;FOR JJ=Y-1 TO Y+1
610 HPLLOT II,JJ;NEXT;NEXT
620 RETURN

```

## 第八讲 子程序设计

南京大学大气科学系(210008) 朱国江

## § 4. 子程序设计

在程序的编制过程中,常常遇到相同的计算或操作,或者,在一个程序中要多次处理重复计算的问题,如代码转换、数据传送、图形处理、函数运算等等。

可以把这些相同的部份编制成一个独立的程序段,当遇到相同的计算或操作时,就转入这个程序段,从而省去对这些相同部份重新编制程序的工作。

我们把在功能上具有一定独立性,能够完成一定计算或操作,并在程序的不同部位要被经常调用的程序段称作子程序。而把调用子程序的程序称为主程序。

大多数程序都是由一个主程序和几个子程序组成。在程序的适当位置安排调用点,就可以转向子程序,简称为转子。而当子程序执行之后再返回到主程序,简称为返主。转子指令 JSR 和返回指令 RTS,可以实现主程序对子程序的正确调用和子程序到主程序的自动返回。

无条件转移指令控制程序转出以后就不再返回了;而转子指令控制程序转向子程序以后,当子程序执行完毕时还要返回主程序被打断处(叫做断点),即返回到调用它的指令的下一条指令去执行程序。保证子程序正确返回的是返回指令 RTS。

为了准确地完成返回“断点”的任务,调用子程序的指令必须能保存断点;而子程序最后执行的必须是将断点送入 PC 寄存器(程序计数器)的指令。JSR 指令具有保护断点的能力,它将断点地址-1 后送入堆栈 S 保存再转向子程序,而 RTS 指令正好完成自堆栈 S 中取一个地址,加 1 后赋给 PC 寄存器的操作,从而使子程序返回到主程序原来的断点处继续执行。由此可见,JSR 和 RTS 的作用是转移程序的控制权,先转到子程序,再返回主程序。

上述概念较多,但很重要。否则就不能保证主程序对子程序的正确调用,也不能保证子程序到主程序的自动返回,特别是处理多个子程序的嵌套时,对于主程序和子程序的合理衔接问题,都将碰到麻烦。

让我们通过一些实例,来介绍子程序设计吧!读者也许会从这些实例中掌握子程序设计的方法和技巧。

## [例 1] 数据块转移

试用子程序改写简单程序设计中的[例 1]的数据块转移程序,即要求将从 \$2000 开始的 5 个连续字节的内容转移到 \$4000 开始的 5 个连续字节内。

分析:

• 整个程序设计可以分成一个主程序和一个子程序两个部份。

• 主程序结构十分简单,首先将转移的个数(可以根据题意要求任意定,本例是 5 个)安排在累加器 A 中,然后调用子程序,结束。一共只要安排三条指令。

• 子程序的任务是完成 N 个数据块的传送。取数、送数采用 Y 寄存器的绝对变址寻址方式指令,通过修改 Y 寄存器中的值(放 3 数据长度),达到修改地址指针的目的,并用 BPL 指令控制数据是否取、送完。为了完成 N 个数据的传送,采用循环的方法。特别注意子程序的结束处不应忘记放一条返回指令 RTS。

由上分析,容易写出程序 8.1:

```

程序 8.1
1000- A9 05 LDA # $05 } 主程序
1002- 20 06 10 JSR $1006 }
1005- 60 RTS
1006- A8 TAY
1007- 88 DEY
1008- B9 00 20 LDA $2000,Y } 子程序
100B- 99 00 40 STA $4000,Y }
100E- 88 DEY
100F- 10 F7 BPL $1008 }
1011- 60 RTS
运行前: * 2000,01 02 03 04 05
运行后: * 1000G✓
 * 4000.4004✓
 * 4000-01 02 03 04 05

```

比较程序 8.1 和简单程序设计中的[例 1]程序 6.1,可以看出:

• 程序 8.1 有更大的灵活性。因为只要改变数据块长度(即 LDA # \$nn 中的立即数),即可完成 N 个数(不超过 FF)的数据块传送而无需改动源程序。

• 程序 8.1 简捷和通用。程序 6.1 若要处理 250 个字节的数据块传送,至少要写上 500 条指令,而程序 8.1 除改变数据块长度外,源程序不加一条指令。

由此可知,程序编制中应注意简捷以节省内存,灵活以方便修改,通用以完成各种类似问题的处理。所有这些都是程序设计的质量要求。

## [例 2] 确定正数的个数

数据长度放在 \$6000 单元,数据从 \$6001 开始存放,要求统计出数据块中正数的个数。

这个问题在循环程序设计[例 3]中作过类似的介绍,那里是确定负数的个数,因而有关程序设计思想的分析,大同小异,不再重复。但我们这里采用子程序方法设计,请注意结构和指令选用上的差别。

程序 8.2

```

1000- AD 00 60 LDA $ 6000 }
1003- A0 00 LDY # $ 00 } 主程序
1005- 20 09 10 JSR $ 1009 }
1008- 60 RTS }
1009- AA TAX }
100A- BD 00 60 LDA $ 6000,X }
100D- 30 01 BMI $ 1010 }
100F- C8 INY }
1010- CA DEX } 子程序
1011- D0 F7 BNE $ 100A }
1013- 84 06 STY $ 06 }
1015- 98 TYA }
1016- 20 DA FD JSR $ FDDA }
1019- 60 RTS }

```

运行前: \* 6000;06 68 F2 87 30 59 2A

运行后: \* 1000G✓

\* 04  
或: \* 1000G✓  
\* 6✓  
\* 0006-04

### [例 3] 找最大值

找一个无符号数据块中的最大值,数据块的长度放在 \$ 6000 单元,数据块起始地址为 \$ 6001,最大值存于 \$ 08 单元。

求最大值问题,我们在循环程序的设计中(程序 7.3)作过介绍,程序 7.3 虽有简短易懂的特点,但通用性不好,数据量一大即不能正确运行。故这里再介绍一个找最大值的程序 8.3。

#### 程序 8.3

```

1000- A9 01 LDA # $ 01 }
1002- 85 06 STA $ 06 } 把数据块首地址
1004- A9 60 LDA # $ 60 } 送入 07 和 06 单元
1006- 85 07 STA $ 07 }
1008- AC 00 60 LDY $ 6000 ;数据块长度存 Y 寄存器
100B- 20 11 10 JSR $ 1011 ;转子程序
100E- 85 08 STA $ 08 ;存结果
1010- 60 RTS
1011- A9 00 LDA # $ 00 ;设最大值=0
1013- 88 DEY ;Y-1
1014- 08 PHP ;标志寄存器进栈,保护 Z 标志
1015- D1 06 CMP ($ 06),Y
1017- B0 02 BCS $ 101B ;下一个数>最大值,转
1019- B1 06 LDA($ 06),Y ;是换最大值→A
101B- 28 PLP ;标志寄存器出栈,
101C- D0 F5 BNE $ 1013 ;判 Y-1 是否为 0
101E- 60 RTS ;否,循环;是,返回主程序

```

运行前: \* 6000;09✓;数据块长度

\* 6001;75 44 A0 BF 54 C4 8F DD 20 ✓源数据

运行后: \* 1000G✓

\* 8✓

\* 0008-DD 找出最大值

现在,我们介绍程序 8.3 的设计思想:

• 主程序放在 \$ 1000—\$ 1010 单元,子程序放在 \$ 1011—\$ 101E 单元。在主程序中,数据块首地址放

在一个特定的地方,例如,放在 \$ 06、\$ 07 单元中,前者存数据块首地址低 8 位,后者放数据块首地址高 8 位,而将数据块长度放在 Y 寄存器中。然后调用子程序,存结果结束。子程序实际上是一个完成“找最大值”这个操作的独立程序段,任务就是找最大值。

• 在子程序中,把所得最大值结束放在 A 中,这是为了主程序、子程序的衔接所做的规定,有了这种规定,对于另一个起始地址,另一个长度的数据块,找最大值时只需仿照此例,把起始地址送入 \$ 07 和 \$ 06 单元,把长度送入 Y 寄存器,然后转入“找最大值”子程序,即可在累加器 A 中得到最大值,而无需对子程序作任何改动或修正。使这个子程序具有了通用性。

### [例 4] 搬家子程序

这里介绍一个实用的搬家子程序,它可以方便地实现数据块的搬移任务。无论对单个数据块或是多个数据块;无论对少量数据还是大量数据;无论对数据记录还是图形信息都可以方便地实现搬移。

#### 程序 8.4

```

1000- A0 00 LDY # $ 00
1002- A9 00 LDA # $ 00
1004- 85 3C STA $ 3C
1006- A9 60 LDA # $ 60
1008- 85 3D STA $ 3D
100A- A9 10 LDA # $ 10
100C- 85 3E STA # $ 3E
100E- A9 60 LDA # $ 60
1010- 85 3F STA $ 3F
1012- A9 00 LDA # $ 00
1014- 85 42 STA $ 42
1016- A9 70 LDA # $ 70
1018- 85 43 STA $ 43
101A- 20 2C FE JSR $ FE2C
101D- 60 RTS

```

将源数据块起始地址放在 \$ 3C 和 \$ 3D 单元,源数据块结束地址放在 \$ 3E 和 \$ 3F 单元,数据块的目的地址放在 \$ 42 和 \$ 43 单元,都是低位地址在前,高位地址在后,然后调用监控中的搬家子程序(入口地址为 \$ FE2C),即可完成单个数据块的搬移任务。但一定不要忘记应用本程序 8.4 时,注意将 # \$ 00 值放入 Y 寄存器中。

本例是将 \$ 6000—\$ 6010 单元的数据搬移至 \$ 7000—\$ 7010 单元中去。对于任意单个数据块的搬移,只需改变 \$ 3C—\$ 3F 及 \$ 42—\$ 43 单元中的内容。例如,将高分辨率第 1 页的图形搬至高分辨率第 2 页图形区,即将 \$ 2000—\$ 3FFF 单元内容搬至 \$ 4000—\$ 5FFF 单元中去,只要改动以下单元值:

1007-20  
100B-FF  
100F-3F  
1017-40

请注意搬移的信息量,它搬移了 8192 个存储单元的信息。而且其搬迁速度也很快,几乎瞬间完成。这是由于机器语言执行速度特别快的特点决定的。

对于多个数据块的搬移,仅仅用上述子程序 8.4 还不够,因为多个数据块的源地址有几处,搬移到的目的地址也有几个,从哪几个地方搬,结果又放在何处,必须随时调整地址指针。我们可以编一个主程序,设置好调整的地址指针,然后不断调用上述子程序 8.4,使之完成多个数据块的搬移任务。请看下面实例。

[例 5] 三个数据块的搬移

将 \$ 6000—\$ 600F, \$ 6100—\$ 610F, \$ 6200—\$ 620F 单元的内容分别顺序搬移到 \$ 7000—\$ 700F, \$ 7100—\$ 710F, \$ 7200—\$ 720F 的连续单元中去。

这个问题,我们在循环程序设计[例 7]中作过介绍(程序 7.9),现在用调用子程序 8.4 的方法处理。

先给出源程序 8.5,并看一下运行结果,然后对程序作一说明。

程序 8.5

```

1000- A9 60 LDA # $ 60
1002- 8D 2C 10 STA $ 102C
1005- 8D 34 10 STA $ 1034
1008- A9 70 LDA # $ 70
100A- 8D 3C 10 STA $ 103C
100D- 20 25 10 JSR $ 1025
1010- EE 2C 10 INC $ 102C
1013- EE 34 10 INC $ 1034
1016- EE 3C 10 INC $ 103C
1019- 20 25 10 JSR $ 1025
101C- EE 2C 10 INC $ 102C
101F- EE 34 10 INC $ 1034
1022- EE 3C 10 INC $ 103C
1025- A0 00 LDY # $ 00
1027- A9 00 LDA # $ 00
1029- 85 3C STA $ 3C
102B- A9 62 LDA # $ 62
102D- 85 3D STA $ 3D
102F- A9 10 LDA # $ 10
1031- 85 3E STA $ 3E
1033- A9 62 LDA # $ 62
1035- 85 3F STA $ 3F
1037- A9 00 LDA # $ 00
1039- 85 42 STA $ 42
103B- A9 72 LDA # $ 72
103D- 85 43 STA $ 43
103F- 20 2C FE JSR $ FE2C
1042- 60 RTS

```

运行前:

```

* 6000. 600F
6000- 01 02 03 04 05 06 07 08
6008- 09 0A 0B 0C 0D 0E 0F 10
* 6100. 610F
6100- 09 0A 0B 0C 0D 0E 0F 10
6108- 01 02 03 04 05 06 07 08
* 6200. 620F
6200- 2F 2E 2D 2C 2B 2A 29 28
6208- 27 26 25 24 23 22 21 03

```

运行后:

```

* 1000G
* 7000. 700F
7000- 01 02 03 04 05 06 07 08
7008- 09 0A 0B 0C 0D 0E 0F 10
* 7100. 710F
7100- 09 0A 0B 0C 0D 0E 0F 10
7108- 01 02 03 04 05 06 07 08
* 7200. 720F
7200- 2F 2E 2D 2C 2B 2A 29 28
7208- 27 26 25 24 23 22 21 03

```

程序 8.5 说明:

• \$ 1025—\$ 1042 就是本节[例 4]中介绍的搬家子程序(\$ 1000—\$ 101D),它们的指令完全一样,只是个别地址的内容不同,存储空间相异而已。其功能是一个独立的完成单个数据块搬家的子程序。

• \$ 1000—\$ 1024 是主程序,它完成第一个数据块首地址、末地址、目标地址的确定和搬迁任务(\$ 1000—\$ 100F),又完成第二、三两个数据块地址指针的调整和搬移任务(分别为 \$ 1010—\$ 101B, \$ 101C—\$ 1042)。

• 本程序执行过程共分三段:① \$ 1000—100F—\$ 1025—\$ 1042;② \$ 1010—\$ 101B—\$ 1025—\$ 1042;③ \$ 101C—\$ 1042。可以看出是一个主程序几次重复调用子程序的结构,最后一次省去了转子指令 JSR,而直接和子程序连成一片,从而完成了主程序和子程序的正确衔接,这是编制程序时应予注意的。

[例 6] 多个子程序的调用

在子程序的设计中,大多数由一个主程序和几个子程序组成,每一个子程序完成的功能不一样,它们之间又具有相互的独立性。以本节[例 5]为例,程序 8.5 中修改地址指针使内存单元加 1 的功能出现二处,可以改成一个独立的子程序,从而使程序 8.5 变成一个主程序调用二个子程序的结构,见程序 8.6。

程序 8.6:

```

1000- A9 60 LDA # $ 60
1002- 8D 20 10 STA $ 1020
1005- 8D 28 10 STA $ 1028
1008- A9 70 LDA # $ 70
100A- 8D 30 10 STA $ 1030
100D- 20 19 10 JSR $ 1019
1010- 20 37 10 JSR $ 1037
1013- 20 19 10 JSR $ 1019
1016- 20 37 10 JSR $ 1037
1019- A0 00 LDY # $ 00
101B- A9 00 LDA # $ 00
101D- 85 3C STA $ 3C
101F- A9 62 LDA # $ 62
1021- 85 3D STA $ 3D
1023- A9 10 LDA # $ 10
1025- 85 3E STA $ 3E
1027- A9 62 LDA # $ 62
1029- 85 3F STA $ 3F

```

```

102B- A9 00 LDA # $ 00
102D- 85 42 STA $ 42
102F- A9 72 LDA # $ 72
1031- 85 43 STA $ 43
1033- 20 2C FE JSR $ FE2C
1036- 60 RTS
1037- EE 20 10 INC $ 1020
103A- EE 28 10 INC $ 1028
103D- EE 30 10 INC $ 1030
1040- 60 RTS

```

程序 8.6 说明:

• 从程序看出,共有四次调用子程序(见 \$ 100D - \$ 1018),调用的是两个子程序(\$ 1019 - \$ 1036 和 \$ 1037 - \$ 1040)。

• 程序 8.6 中,有子程序嵌套的情况。例如,当主程序工作时,首先转入子程序 1(\$ 1019 - \$ 1036),而子程序 1 工作时又转入子程序 2(\$ 1033 - \$ 1036),当子程序 2(监控搬家子程序)工作完毕后返回子程序 1,子程序 1 执行完毕后再返回主程序。在这个过程中,子程序 1 相对于子程序 2 而言就处在主程序中的地位,这就叫做子程序嵌套。

[例 7] 输入码转换

在键盘上按下三位十进制数,转换成 16 进制数。

如:按键 254 (10 进制数) 显示 FE(16 进制数)

按键 128 (10 进制数) 显示 80(16 进制数)

关于从键盘上按下一位 10 进制数,将其转换成 16 进制数的问题,我们曾在分支程序设计一节中作过介绍,见程序 6.13。现在的问题是,要从键盘上按下三位 10 进制数,将其转换成 16 进制数,除了可以调用前面的程序 6.13(它的任务是把 10 进制数转换成 2 进制数)外,还应多次调用,以完成百位数、十位数、个位数均转换成 2 进制数的问题,同时,还要有另外一个子程序,它完成乘 10 加数的功能,并且不止一次调用。

程序 8.7

```

1000- 20 15 10 JSR $ 1015;取百位数转成 2 进制在 A
1003- 85 06 STA $ 06 ;存 06 单元作为中间结果
1005- 20 15 10 JSR $ 1015取拾位数转成 2 进制在 A
1008- 20 23 10 JSR $ 1023; 转乘 10 加数子程序
100B- 20 15 10 JSR $ 1015; 取个位数转 2 进制在 A
100E- 20 23 10 JSR $ 1023; 转乘 10 加数子程序
1011- 20 DA FD JSR $ FD0A;显示结束
1014- 60 RTS
1015- 20 1B FD JSR $ FD1B }
1018- C9 B0 CMP # $ B0 }
101A- 90 F9 BCC $ 1015 }
101C- C9 BA CMP # $ BA }
101E- B0 F5 BCS $ 1015 }
1020- 29 0F AND # $ 0F }
1022- 60 RTS }
1023- 85 07 STA $ 07 ;整存 10 位,个位数
1025- A5 06 LDA $ 06 ;调中间结果
1027- 0A ASL ;得 2A
1028- 0A ASL ;得 4A

```

10 进制数转 2 进制数子程序

```

1029- 65 06 ADC $ 06 ;得 5A
102B- 0A ASL ;得 10A
102C- 65 07 ADC $ 07 ;得 10 加数
102E- 85 06 STA $ 06 ;存结果到工作单元
1030- 60 RTS

```

运行: \* 1000G ✓  
\* 254 ✓  
\* FE  
\* 1000G ✓  
\* 128 ✓  
\* 80

• \$ 1015 - \$ 1022 是一个子程序,它完成 10 进制数转换成 2 进制数的工作,由于本程序要处理百位数、十位数、个位数转换成 2 进制的工作,所以三次调用了这个子程序。

• \$ 1023 - \$ 1030 是另一个子程序,它完成乘 10 加数的工作,共要调用两次。

• 本程序的两个子程序构成两重子程序结构。

[例 8] 比较两个字符串是否相同

比较两个 ASCII 字符串,看其是否相同。字符串长度在存储单元 6000 中,两个字符串的起始地址各为 6100 和 6200,如果这两个字符串相同,将 00 放入 6001 单元,反之不同放 FF 标志。示范题:

a) (6000)=03 字符串长度  
(6100)=43 "C"  
(6101)=41 "A"  
(6102)=54 "T"  
(6200)=44 "D"  
(6201)=41 "A"  
(6202)=54 "T"

结果: (6001)=FF

b) (6000)=03  
(6100)=43 "C"  
(6101)=41 "A"  
(6102)=54 "T"  
(6200)=43 "C"  
(6201)=41 "A"  
(6202)=54 "T"

结果: (6001)=00

程序 8.8

```

1000- AC 00 60 LDY $ 6000
1003- 20 0A 10 JSR $ 100A
1006- 8D 01 60 STA $ 6001
1009- 60 RTS
100A- A2 FF LDX # $ FF
100C- 88 DEY
100D- B1 06 LDA ($ 06),Y
100F- D1 08 CMP ($ 08),Y
1011- D0 05 BNE $ 1018
1013- 98 TYA
1014- D0 F6 BNE $ 100C
1016- A2 00 LDX # $ 00
1018- 8A TXA

```



1019-60 RTS

• 主程序见 \$1000—\$1009,字符串长度放 Y 寄存器,转字符串比较子程序,送结果,结束。

• 子程序见 \$100A—\$1019,首先置字符串不同标志 FF→X,Y-1,逐字比较两个字符串,若不相同返回;若相同,看每个字符是否都比较大完,是则取 00 标志→X,返回,不是再循环上去。

• 在运行本程序前,应将两字符串起始地存入

(接第 48 页)

统的提示符下,只要打入文件名 LOWFORM,然后根据屏幕提示信息打入“回车”键即可。

(3)运行随机提供的诊断程序

不管是 IBM-PC 系列机或是长城系列机及其兼容机,都随机提供给用户一个高级诊断程序。利用这个诊断程序也可进行硬盘的初始化操作,只是不同机型诊断程序的名称不同而已。例如新一代长城系列机提供给用户一个诊断程序盘,进行诊断程序时,显示如下: The GREAT Wersonal computer DIAGNOSTICS

```
Version 6.04(c)copyright ACI1958
SELECT AN OPTION
0-RUN DIAGNOSTIC ROUTINES
1-FORMAT DISKETTE
2-COPY DISKTEET
3-PREPARE FIXED DISK FOR RELOCATION
4-SETUP YOUR SYSTEM
9-EXIT TO SYSTEM DISKETTE
ENTER THE ACTION DESIRED
?
```

用户选择 0 项后,诊断程序就会清点系统的配置情况,并把这一配置情况显示出来。一种可能的显示情况如下:

```
THE INSTALLED DEVICES ARE
S SYSTEM UNIT 1
S 512KB MEMORY 2
S KEYBOARD 3
S 2 DISKETTE DRIVE(S)AND ADAPTER 6
S SERIAL/PARALLEL ADAPTER
-PARALLEL PORT 9
S SERIAL/PARALLEL ADAPTER
-SERIAL PORT 11
S ALTERNATE SERIAL/PARALLEL ADAPTER
-SERIAL PORT 12
S 1 FIXED DISK DRIVE(S)
AND ADAPTER 17
S TEST THE MATRIX PRINTER
INSTALLED 36
IS THE LIST CORRECT (Y/N)
?
```

根据屏幕上的提示信息,回答 Y 并回车,屏幕又显示如下信息:

```
SYSTEM CHECKOUT
0- RUN TESTS ONE TIME
```

06,07 单元和 08,09 单元,而在 6100 和 6200 开始存放字符串的 ASCII 码,然后 1000G↙,即可看到结果。

• 本程序比较两个字符串时,采用由后向前比。

从以上所举实例,可以看出在程序设计中,有一些程序需要反复使用,为了方便操作,简化设计,我们常采用子程序的方法,使那些在功能上一样,处理问题相同,并具有一定独立性的重复操作部份编在一起,只要注意正确调用和返回,主程序与子程序衔接完整即可。

```
1- RUN TESTS MULTIPLE TIMES
2- LOG UTILIES
9- END SYSTEM CHECKOUT
10-CHECK PASS COUNTER
SELECT THE ACTION DESIRED
?
```

选择 0 项后,诊断程序又显示系统配置情况,因为是对硬盘进行测试,所以应该选择 17 并按回车键,这时进入硬盘检测菜单:

```
REFER TO-TEST 1700
PRESS“ENTER”TO CONTINUE
?
```

按回车键后屏幕继续显示:

```
FIXED DISK DIAGNOSTIC MENU
1- WRITE,READ,COMPARE (ON TSET CYLINDER)
2- SEEK TEST
3- HEAD SELECT
4- ERROR DETECTION AND CORRECTION
5- RUN ALL TESTS
6- READ VERIFY
7- FORMAT MENU
9- RETURN TO CONTROL PROGRAM
FOR OPTION 9
TYPE“9”AND PRESS“ENTER”
FOR OTHER OPTION TYPE THE OPTION
DRIVEID(1,C),AND PRESS“ENTER”
?
```

此时回答 7,C 并回车就进入硬盘格式化菜单:

```
FORMAT SELECTION MENU
1- CONDITIONAL FORMAT
2- UNCONDITIONAL FORMAT
3- SURFACE ANALYSIS
4- CHANGE INTERLEAVE
9- RETURN TO FIXED DISK MENU
FOR OPTION 9
TYPE“9”AND PRESS“ENTER”
FOR OTHER OPTIONS
TYPE THE OPTION NUMBER
DRIVEID(1,C),AND PRESS“ENTER”
?
```

如果选择 1,C 则进行条件格式化硬盘;如果选择 2,C 则进行无条件格式化硬盘。不论选择上述哪种情况,屏幕均出现警告信息如下:

```
ALL DEFECTS WILL BE SHOWN ON THE DISPLAY
THEY CAN ALSO BE PRINTED ON LPT1 (转第 33 页)
```

# 一九九二年计算机初级软件人员竞赛试题

主办单位：中国软件行业协会考试指导中心

北京市科学技术协会

《电子与电脑》编辑部

《计算机世界》编辑部

说明：本试题以 IBM PC 及其兼容机为基本机型。

一、从供选择的答案中选出适合下列表格中的 a~h 的正确答案，把相应的编号写在答卷纸的对应栏内。（每空 1 分，共 8 分）

| 二进制数     | 八进制数 | 十进制数 | 十六进制数 |
|----------|------|------|-------|
| 01000010 | c    | 66   | g     |
| a        | d    | 127  | 7F    |
| 01010101 | 125  | e    | h     |
| b        | 143  | f    | 63    |

a,b 供选择的答案：

- (1) 01011110 (2) 01111111  
(3) 01111000 (4) 01100011

c,d 供选择的答案：

- (5) 135 (6) 120  
(7) 177 (8) 102

e,f,g,h 供选择的答案：

- (9) 92 (10) 85 (11) 52  
(12) 99 (13) 42 (14) 68  
(15) 73 (16) 55

二、从供选择的答案中选出正确答案，把相应的编号写在答卷纸的对应栏内。（每空 1 分，共 22 分）

1. 已知一逻辑表达式为  $F = \overline{A}B + \overline{B}C + \overline{C}A$ ，化简后可得到逻辑表达式\_\_\_\_\_。

供选择的答案：

- (1)  $\overline{A}B + \overline{B}C + \overline{C}A$  (2)  $\overline{A}B + \overline{B}C + \overline{A}C$   
(3)  $\overline{A}B + \overline{A}B + \overline{B}C$  (4)  $\overline{A}B + \overline{B}C + \overline{A}C$

2. DMA 控制器是控制外部输入输出设备与\_\_\_\_\_直接传送信息（数据）的逻辑电路。

供选择的答案：

- (1) 运算器 (2) 主存储器  
(3) 程序计数器 (4) 指令译码器

3. 计算机中的存储器分为主存储器和辅助存储器两大类。

主存储器用来存放当前运行所需要的程序和数据。它的存取速度\_\_\_\_\_、存储容量\_\_\_\_\_、价格\_\_\_\_\_。它所存储的信息\_\_\_\_\_永久性地保留。

辅助存储器用来存放当前不需要立即使用的程序和数据，一旦需要，再和主存储器成批地交换数据。它作为主存储器的后备和补充，是主机的外部设备，与主存储器相比较，它的存取速度\_\_\_\_\_，存储容量\_\_\_\_\_，价格\_\_\_\_\_。它所存储的信息\_\_\_\_\_永久性地脱机保留。

供选择的答案：

- (1) 慢 (2) 大 (3) 快 (4) 能够  
(5) 不能 (6) 小 (7) 低 (8) 高

4. 一个字节的 8 位二进制组成，它除了表示 26 个英语字母、0~9 个数字以外，还能表示\_\_\_\_\_个字符。

两个字节能表示的字符数量是一个字节能表示的字符数量的\_\_\_\_\_倍。

供选择的答案：

- (1) 2 (2) 8 (3) 128  
(4) 220 (5) 64 (6) 256

5. 显示设备的种类繁多，按它的功能分类，可分为普通显示器和显示终端两大类。显示器和显示终端是两个不同的概念。

显示器的功能简单，只能用于接收\_\_\_\_\_信号，它的控制逻辑和存储逻辑都在\_\_\_\_\_中，目前使用的个人计算机系统就是这种结构。

显示终端是由显示器和键盘组成的一套独立完整的输入/输出设备。它具有显示控制与存储、键盘管理、通信控制等功能，还可以完成简单的编辑操作。它可以通过\_\_\_\_\_接到远离主机的地方使用。

供选择的答案：

- (1) 音频 (2) 主机接口板  
(3) 标准通信接口 (4) 视频  
(5) 电缆线 (6) 控制台

6. 在显示技术中，图形和图象都是由称为象素的点组成的。显示设备所能表示的象素个数称为\_\_\_\_\_。象素越密，图象越\_\_\_\_\_。显示设备所显示象素点的亮暗差别、深浅变化称为\_\_\_\_\_。它在彩色显示器中则表现为\_\_\_\_\_不同。

供选择的答案：

- (1) 清晰 (2) 颜色 (3) 灰度级  
(4) 分辨率 (5) 模糊 (6) 精确度

7. 激光印字机是激光技术和\_\_\_\_\_相结合的产物。它的输出速度快，印字质量高，是一种\_\_\_\_\_硬拷贝输出设备。

供选择的答案：

- (1) 击打式 (2) 电子照相技术  
(3) 机械印字技术 (4) 非击打式

8. 微机安装时，应注意电源的供电电压是否和微机所要求的一致，同时必须按照\_\_\_\_\_的规则连接电源线。

供选择的答案：

- (1) 左零(零线)右火(火线) (2) 右零左火

三、从供选择的答案中选出正确答案，把相应的编号写在答卷纸的对应栏内。（每空 1 分，共 10 分）

1. 用\_\_\_\_\_编制的源程序输入计算机后，可直接运行得到结果。

供选择的答案：

- A. 汇编语言 B. 机器语言  
C. BASIC 语言 D. C 语言

2. \_\_\_\_\_记录的逻辑顺序与物理位置顺序不一致。

供选择的答案：

- A. 磁带文件 B. 打印机文件  
C. 索引文件 D. 顺序文件

3 在操作系统处理机管理中,一般把\_\_\_\_\_当做资源分配和调度的基本单位。

供选择的答案:

- A. 作业 B. 进程 C. 程序 D. 用户

4 在操作系统中,文件按其性质和用途可分为几类,其中库文件是指\_\_\_\_\_。

供选择的答案:

- A. 由用户建立起来的数据库文件  
B. 由标准子程序及常用的子程序组成的文件  
C. 允许文件主和授权的用户读写和使用的文件  
D. 有关操作系统及其系统程序信息组成的文件

5 定义一个数组A[a, b],设A[0, 0]的地址是L,这个数组按列存储,其元素的地址从L开始逐渐递增,那么 A[i, j] 的地址是\_\_\_\_\_。  
( $0 \leq i \leq a, 0 \leq j \leq b$ )。

供选择的答案:

- A.  $L+B*i+(j+1)$   
B.  $L+(b+1)*i+(j+1)$   
C.  $L+(a+1)*j+(i+1)$   
D.  $L+(a+1)*j+1$

6 \_\_\_\_\_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. 内码  
C. 外部输入码 D. 打印输出码

7 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

8 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

四. 判断下面的叙述是否正确,正确叙述画√,错误的叙述画×,填在答卷纸的对应栏内。(每空1分,共10分)

- DOS的外部命令以文件形式存在于磁盘上,在使用该命令时才调入内存。( )
- 堆栈是一种特殊的线性表,它采用先进先出的原则组织数据。( )
- 设计数据库的目的是使用户方便有效地存储和检索数据,并获得信息。( )
- 文件系统是在数据库基础上发展起来的。( )
- UNIX操作系统是一种通用、交互式的实时操作系统。( )
- 计算机病毒是由硬件故障造成的。( )
- dBASE是一种关系数据库管理系统。( )用表格数据来表示实体之间联系的模型称为关系模型。( )
- 反汇编是将机器语言翻译成高级语言。( )汇编程序是把用汇编语言编写的源程序翻译成机器语言程序。( )

言程序。( )

五. 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3分,共15分)

1. BASIC表达式 $INT(-50+50 * RND(0)) + FIX * (-50+50 * RND(1))$ 产生随机整数的区间是\_\_\_\_\_。

供选择的答案:

- A. (-99, -1) B. [-99, -1]  
C. [-100, -2] D. (-100, -2)

2. 已知BASIC语句 $A\$ = "A123\#-3456"$ ,  $L = LEN(STR$(-321)) + VAL(MTD$(A$, 6, 2))$ , 则 $L =$ \_\_\_\_\_。

供选择的答案:

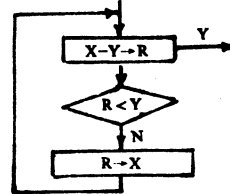
- A. -1 B. 0 C. 1 D. 2

3. BASIC表达式 $INT(SIN(290 * 3.14159 / 180)) + SGN(COS(190 * 3.14159 / 180))$ 的值是\_\_\_\_\_。

供选择的答案:

- A. -1 B. 0 C. 1 D. -2

4. 这个流程图对应的程序功能是\_\_\_\_\_。



供选择的答案:

- A. 求X对Y的模  
B. 求X与Y的差  
C. 求X与2Y的差  
D. 死循环

5. 下面关于循环的说法哪种正确\_\_\_\_\_。

供选择的答案:

- A. 计数循环, 当型循环, 直到型循环是不可转化的。  
B. 计数循环和当型循环可以相互转化, 而计数和直到型循环是不可以相互转化的。  
C. 当型循环和直到型循环可以互相转化而计数和后两种循环都不能相互转化。  
D. 三种循环之间是可以相互转换的。

六. 阅读下列BASIC程序, 将供选择答案中正确答案相应的字母填在答卷纸的对应栏内。(每空3分, 共15分)

1 程序

```

10 Y=1
20 DEF FNA(X)=X^2+2 * X+1
30 DEF FNB(X)=FNA(X)-1
40 Y=Y+FNB(FNA(1))
50 PRINT "Y="; Y
60 END

```

运行结果( )

供选择的答案:

- A. Y=4 B. Y=26 C. Y=25 D. Y=24

2 程序

```

10 A$="CBACBBAACBAC"
20 B$="BA"
30 V=INSTR(4,A$,B$)
40 Y$=MID$(A$,V,1)
50 FOR I=2 TO 1 STEP -1
60 Y$=LEFT$(A$,I)+Y$+RIGHT$(A$,I)
70 NEXT I

```

```
80 PRINT Y $
90 END
```

运行结果 ( )

供选择的答案:

- A. CCBBACC B. CBBAACC  
C. CBCBCAC D. CBABC

3 程序

```
10 N=0
20 FOR I=1 TO 10
30 I=I+4
40 N=N+I^2
50 NEXT I
60 I=I+4
70 N=N+I^2
80 PRINT N
90 END
```

运行结果 ( )

供选择的答案:

- A. 321 B. 350 C. 564 D. 599

4 程序

```
10 INPUT X
20 P=0
30 GOSUB 60
40 PRINT P
50 END
60 Y=X^2
70 IF Y>1 THEN 100
80 P=P+X^2
90 RETURN
100 P=P+X
110 RETURN
```

这个程序完成的功能是计算下列哪个函数的值 ( )

A.  $P = \begin{cases} X^2 & |X| \geq 1 \\ X & |X| < 1 \end{cases}$

B.  $P = \begin{cases} X^2 & |X| \leq 1 \\ X & |X| > 1 \end{cases}$

C.  $P = \begin{cases} X^2 + X & |X| \leq 1 \\ X & |X| > 1 \end{cases}$

D.  $P = \begin{cases} X^2 - X & |X| \leq 1 \\ X & |X| > 1 \end{cases}$

5 程序

```
10 FOR M=3 TO 20
20 FOR I=2 TO M-1
30 IF INT (M/I) = M/I THEN 60
40 NEXT I
50 PRINT M
60 NEXT M
70 END
```

运行结果 ( )

供选择的答案:

- A. 235711131719  
B. 235791113151719  
C. 35791113151719  
D. 35711131719

七. 补充完成下列程序, 使其能完成求任意两个数的最

大公约数, 将答案填在答卷纸对应栏内. (每空 3 分, 共 15 分)

```
10 INPUT X, Y
20 IF Y >= X THEN 60
30 A=X
40 B=Y
50 GOTO 80
60 A=Y
70 B=X
80 R=A-B
90 IF (1) THEN 120
100 A= (2)
110 GOTO 80
120 IF (3) THEN 160
130 A= (4)
140 B= (5)
150 GOTO 80
160 PRINT B
170 END
```

这个程序采用的算法是欧几里德辗转相除法. 其步骤如下:

- (1) 令二个数值放在 A、B 中, A 中为大数
- (2) 求用 B 除 A 的余数 R1
- (3) 然后求用 R1 除 B 的余数 R2
- (4) 再求用 R2 除 R1 的余数 R3
- (5) 依次类推, 用新求得的余数除前一个余数, 得到下一个余数
- (6) 经过若干次除法以后, 余数 Rn 便会为 0, 这时 Rn-1 就是 A、B 的最大公约数.

八 从供选择的答案中选出正确答案, 把相应的编号写在答卷纸的对应横线上. (每空 2 分, 共 14 分)

1. 在 dBASE III 中用户文件有 \_\_\_\_\_ 类, 其中索引文件的默认扩展名是 \_\_\_\_\_.

供选择的答案:

- A. 5 B. 7 C. 8 D. 9  
E. prg F. dbf G. ndx H. txt

2. CONFIG.SYS 文件的作用是 \_\_\_\_\_.

供选择的答案:

- A. 通过设置 CONFIG.SYS 改变 DOS 缓冲区数和允许同时打开的文件数的默认值  
B. 设置 DOS 缓冲区的大小  
C. 分配内存空间  
D. 设置路径

3. 汉字 dBASE III 有 \_\_\_\_\_ 个工作区, 在每个工作区中, 用户都可单独打 \_\_\_\_\_ 个库文件. 刚进入 dBASE III 时 \_\_\_\_\_ 号工作区被默认为当前工作区.

供选择的答案:

- A. 1 B. 2 C. 5 D. 10

4. 下面 \_\_\_\_\_ 不是 dBASE III 的字段类型:

供选择的答案:

- A. 数字型 B. 日期型 C. 逻辑型  
D. 字符型 E. 记忆型 F. 函数型

九. 在汉字 dBASE III 系统下, 用户已建立名为 GZ.DBF 的文件, 用 LIST 命令看显示的结果为: (每空 2 分, 共 26 分)

| RECORD# | 编号 | 姓名 | 工龄 | 职称  | 工资  |
|---------|----|----|----|-----|-----|
| 1       | 01 | 张一 | 10 | 技术员 | 100 |
| 2       | 04 | 王二 | 15 | 工程师 | 120 |
| 3       | 03 | 李三 | 20 | 高工  | 200 |
| 4       | 06 | 赵四 | 5  | 助工  | 110 |
| 5       | 02 | 马五 | 7  | 工人  | 80  |
| 6       | 05 | 严六 | 25 | 高工  | 250 |

下面命令将对哪些记录进行编辑或替换 ( 写出记录号即可 ) ?

- 1 .USE GZ  
.EDIT  
记录号( )  
供选择的答案:  
A. 1 B. 2 C. 3 D. 5
- 2 .USE GZ  
.GO 3  
.APPEND BLANK  
记录号( )  
供选择的答案:  
A. 1 B. 3 C. 5 D. 7
- 3 .USE GZ  
.REPLACE 工资 WITH 工资+20 FOR 工龄 > 15  
记录号( ),( )  
供选择的答案:  
A. 1 B. 2 C. 3 D. 4 E. 5 F. 6
- 4 .USE GZ  
.LOCATE FOR 职称="高工".AND.工龄 > 15  
.CONTINUE  
记录号( ),( )  
供选择的答案:  
A. 1 B. 2 C. 3 D. 4 D. 5 F. 6
- 5 .USE GZ  
.GOTO 3  
.COUNT NEXT 4 FOR 工龄 < 20 .OR. 姓名="严六"  
执行结果 ( )  
供选择的答案:  
A. 1 B. 2 C. 3 D. 4 E. 5 F. 6
- 6 .USE GZ  
.COPY TO GZ1 NEXT 5 FOR 职称="工程师"  
.OR. 工龄 > 10

- .USE GZ1  
.SUM 工资  
执行结果( )  
供选择的答案:  
A. 320 B. 570 C. 370 D. 420
- 7 .USE GZ  
.INDEX ON 编号 TO A:GZ.NDX  
执行后此文件的逻辑顺序为( )  
(注:按记录号排列)

- 供选择的答案:  
A. 2,3,4,5,1,6 B. 2,1,6,4,5,3  
C. 1,5,3,2,6,4 D. 1,2,3,4,5,6
- 8 SET TALK OFF  
USE GZ  
STORE 0 TO A,B,C,D  
DO WHILE .NOT. EOF( )  
IF 工资 > 150  
C=C+工资  
A=A+1  
ELSE  
D=D+工资  
B=B+1  
ENDIF  
SKIP  
ENDDO  
程序执行后变量  
A=( ), B=( ), C=( ), D=( )  
供选择的答案:  
A. 1 B. 2 C. 3 D. 4  
E. 400 F. 450 G. 410 H. 860

十. 请将下面的计算机英语术语译成中文。

- (1) screen
- (2) diagnostic routine
- (3) interrupt
- (4) debug
- (5) on-line
- (6) workstation
- (7) compile
- (8) temporary area
- (9) virtual address
- (10) default

十一. 将下面的计算机术语或屏幕信息译成中文:

- (1) Program too big to fit in memory
- (2) file allocation table bad
- (3) Internal stack overflow, system halted
- (4) Content of destination lost before copy

## 说明

本竞赛试题是初级程序员级软件水平考试辅导综合自测题,也是我考试指导中心函授班期末复习考试题,希望广大读者及学员踊跃参赛,以检验自己的学习成绩及计算机知识水平。

中国软件行业协会考试指导中心  
91.7.

## 培训消息

我中心常年举办文字处理办公自动化初级班,进行面授教学。培训内容: DOS, CCDOS, WS, 五笔字型录入, 计算机排版技术, dBASE III 数据库。每月一期, 月初开学, 每期三周, 讲授 80 学时, 上机 40 学时。培训费: 走读生 300 元 (包括培训费, 上机费, ), 资料费实收。住读生: 另加住宿费 180 元。结业经考试合格发给钢印证书。

联系地址: 北京学院路 29 号 (中国地质大学东门内), 中国软件行业协会考试指导中心, 邮政编码: 100083  
电话: 2012233-322

# 一九九二年计算机初级程序员有奖竞赛试题答卷

总分: \_\_\_\_\_ 学号: \_\_\_\_\_ (如是函授学员请注明学号)  
 姓名: \_\_\_\_\_ 性别: \_\_\_\_\_ 文化程度: \_\_\_\_\_ 邮政编码: \_\_\_\_\_  
 单位: \_\_\_\_\_ 通信地址: \_\_\_\_\_

**注意事项:**

1. 答案必须在九月二十三日前寄出, 以当地邮戳为准(通信地址: 北京学院路29号, 中国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  |   |
|----|--------|-----|-----|-----|------|----|---|---|----|--|--|----|-----|---|
|    | a      |     | b   |     | c    |    | d |   |    |  |  |    |     | 8 |
| 一  | e      |     | f   |     | g    |    | h |   |    |  |  |    | 22  |   |
|    | 1      |     | 2   |     | 3    |    | 4 |   |    |  |  |    |     |   |
| 二  | 5      |     | 6   |     | 7    |    | 8 |   |    |  |  |    | 10  |   |
|    | 1      | 2   | 3   | 4   | 5    | 6  | 7 | 8 |    |  |  |    |     |   |
| 三  | 1      | 2   | 3   | 4   | 5    | 6  | 7 | 8 | 10 |  |  |    |     |   |
| 四  | 1      | 2   | 3   | 4   | 5    | 6  | 7 | 8 | 10 |  |  |    |     |   |
| 五  | 1      | 2   | 3   | 4   | 5    | 15 |   |   |    |  |  |    |     |   |
| 六  | 1      | 2   | 3   | 4   | 5    | 15 |   |   |    |  |  |    |     |   |
| 七  | (1)    | (2) | (3) | (4) | (5)  | 15 |   |   |    |  |  |    |     |   |
| 八  | 1      | 2   | 3   | 4   | 14   |    |   |   |    |  |  |    |     |   |
| 九  | 1      | 2   | 3   | 4   | 26   |    |   |   |    |  |  |    |     |   |
|    | 5      | 6   | 7   | 8   |      |    |   |   |    |  |  |    |     |   |
| 十  | (1)    | (2) | (3) | (4) | (5)  | 10 |   |   |    |  |  |    |     |   |
|    | (6)    | (7) | (8) | (9) | (10) |    |   |   |    |  |  |    |     |   |
| 十一 | (1)    |     |     |     |      |    |   |   |    |  |  | 10 |     |   |
|    | (2)    |     |     |     |      |    |   |   |    |  |  |    |     |   |
|    | (3)    |     |     |     |      |    |   |   |    |  |  |    |     |   |
|    | (4)    |     |     |     |      |    |   |   |    |  |  |    |     |   |
| 总分 |        |     |     |     |      |    |   |   |    |  |  |    | 150 |   |



# BJS-51 的 监 控 程 序

北京航空航天大学(100083) 盛焕鸣

教育实验系统 BJS-51 的监控程序约 16K, 固化在一片 27128 中, 实际上它由两个独立的程序块组成。前 8K 为以键盘及 LED 为输入输出手段的管理程序 (BJS-51-LED), 后 8K 为以 PC 机(其他类型计算机或终端)为输入输出手段进行用户程序编写、运行、调试的管理程序(BJS-51-PC)。在运行时通过接在 27128 地址线 A13 的开关 K1(见图 1)来选择前 8K 或后 8K 投入运行, 因此用户可以根据自己的使用环境通过 K1 来选择。

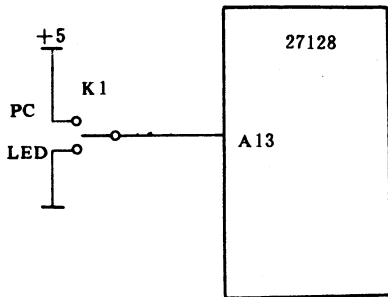


图 1

在单片机系统设计、调试过程中, 大多数人都经历过一个个敲机器码的阶段(而且很多同志由于习惯还在敲机器码)。但随着 PC 机的价格下降, PC 机已成为单片机应用系统开发过程中一种必不可少的工具, 从硬件设计阶段的电原理图设计、印制板设计到软件设计过程中的源文件编写、汇编、系统调试以至说明书的编写, PC 机都是最好的助手。今后从事单片机系统设计的工程技术人员为了适应市场竞争机制提高效率, 必须掌握 PC 机的使用, 并把它使用到单片机系统开发的各个环节中。因此, 设计教育实验系统时, 是以有 PC 机支持为最基本运行环境的, 这不仅是为了降低成本满足学校的使用环境, 更重要的是为了使学生会较先进的单片机开发调试技术。

作为补充, BJS-51 也提供在键盘、LED 显示器支持下进行用户程序调试的功能。由于使用键盘、LED 显示器进行程序调试具有携带方便设备简单等特点, 因此特别适合于在现场进行单片机系统故障检查、程序修改、EPROM 写入等工作。考虑到目前还有很多单位不具备 PC 机, 所以 BJS-51 上所配置的键盘-LED 监控程序具有一般仿真器的功能。下面分别对这两个监控程序进行介绍。

## 一、BJS-51-PC

BJS-51-PC 具有很强的调试功能, 可以代替仿真器进行 51 系列单片机用户程序的调试。这些功能包

括:

### 1. 状态显示功能

可以显示单片机内部 128(或 256)字节 RAM 的内容, 以及所有特殊功能寄存器及二个 64K 的外部存储器空间的内容。并可以 ASCII 字符方式显示外部 RAM 的内容。

### 2. 状态修改功能:

可以对内部 RAM、特殊功能寄存器、外部 RAM 空间、寄存器组按字节进行修改。此外还可以对单片机系统的位控区进行逐位修改。

### 3. 运行控制功能:

可以控制单片机进行单步运行、多步跟踪运行、断点运行、连续全速运行。除全速运行外, 其他各种运行方式都能在运行结束后, 向用户提供运行后单片机系统的状态信息。

### 4. 数据传送功能:

可以在外部 RAM 空间中 SRAM(静态随机存储器)之间以及 SRAM 与 EEPROM(电可改写的程序存储器)之间进行数据传送。可以在外部 RAM 空间与 EPROM 空间(由 EPROM 固化板开辟的存储空间)之间进行数据传送。此外还能在单片机外部 RAM 空间与磁盘文件之间进行数据传送。

### 5. 其他功能:

1) 反汇编功能。可以对单片机外部 RAM 空间中的机器码进行反汇编, 以列表方式进行显示、存盘。

2) 字符串搜索功能。可以在指定外部 RAM 空间中对指定字符串(字符串最多可由十个字符组成)进行搜索, 并在屏幕上显示出指定字符串的存储地址(出现字符串中首字符的地址)。

3) 数据块比较功能。可以对指定的二个数据块进行比较, 若二个数据块中内容不同, 则显示所有不相同单元的地址及内容。

4) EPROM 编程功能。包括编程、读出、检查、检验等功能(必须与 EPROM 编程板配合)。

BJS-51-PC 在系统复位后首先要与终端机进行异步通信的波特率同步。BJS-51-PC 采用测试终端机异步通信信号的位宽度, 通过计算确定定时器 T1 的时间常数并装入到 T1H, 来确定 BJS-51 的通信波特率。因此, 系统复位后 PC 机(或终端机)必须键入一个回车(送一个 ASCII 码 0DH)来提供终端机的波特率时间常数。串口初始化后在 PC 机的屏幕上将显示软件的版本号及提示符“\*”。这时用户可以根据需要键入各种监控命令。

BJS-51-PC 的监控命令格式为:

ZZ n1,n2,n3……〈回车〉

其中 ZZ 为命令字,用以确定哪一种操作。后面的空格是分隔符,表示命令字结束。

n1,n2,n3……为命令所需要的参数,例如数据块移动的源地址等,不同的命令需要的参数不同。

〈回车〉为整个命令的结束符。系统在接到回车后就开始执行键入的命令。

BJS-51-PC 的各类命令见表 1。

表、BJS-51-PC 监控命令表

| 命令类型 | 命令格式             | 执行功能                              |
|------|------------------|-----------------------------------|
| 状态显示 | DX n1,n2〈回车〉     | 外部数据存储器内容显示                       |
|      | DC n1,n2〈回车〉     | 外部程序存储器内容显示                       |
|      | D〈回车〉            | 片内数据存储器内容显示                       |
|      | R〈回车〉            | 显示局部参数 PC,ACC,SP,B,DPTR,RO,R1,PSW |
| 状态修改 | SX n1〈回车〉        | 外部数据存储器内容修改                       |
|      | SD n1〈回车〉        | 片内数据存储器内容修改                       |
|      | BIT n1〈回车〉       | 位空间内容修改                           |
|      | XX〈回车〉           | 特殊功能寄存器内容修改<br>XX 为特殊功能寄存器名       |
|      | REG n1〈回车〉       | 当前寄存器组内容修改                        |
| 数据传送 | M n1,n2,n3〈回车〉   | 在外部 RAM 空间进行数据块传送                 |
|      | ME n1,n2,n3〈回车〉  | 把外部 RAM 的内容送到 EEPROM 中去           |
|      | EPROM 操作         | MPJ n1,n2〈回车〉                     |
|      | MPR n1,n2,n3〈回车〉 | 把 EPROM 的内容送到 RAM 空间              |
|      | MPX n1,n2,n3〈回车〉 | 把 EPROM 的内容与 RAM 的内容比较            |
|      | MPK n1,n2,n3〈回车〉 | 把 RAM 空间的内容固化到 EPROM 中去           |
| 运行控制 | T n1〈回车〉         | 从 n1 开始执行一条指令,并显示当前状态             |
|      | T n1,n2〈回车〉      | 从 n1 开始多步执行至 n2                   |
|      | G n1,n2〈回车〉      | 从 n1 开始运行至断点 n2                   |

|      |                         |                      |
|------|-------------------------|----------------------|
|      | E n1〈回车〉                | 从 n1 开始连续运行用户程序      |
| 其他命令 | DI n1,n2,n3〈回车〉         | 反汇编 n1 到 n2,定位地址为 n3 |
|      | DA n1,n2〈回车〉            | ASCII 字符显示           |
|      | C-C n1,n2,n3〈回车〉        | 数据块比较,显示不相等单元        |
|      | SX n1,n2〈回车〉m1,m2……〈回车〉 | 字符串搜索                |

使用 BJS-51-PC 虽然占用一些单片机资源,但绝大多数情况并不影响用户程序的运行,因此 BJS-51-PC 也常被应用在用户自己的产品中,即在用户设计自己的产品时,把 BJS-51-PC 运行所需要的资源也考虑进去,然后在自己的产品上插上 BJS-51-PC,使之成为具有自开发功能的用户系统。

BJS-51-PC 使用时,需要使用单片机部份资源。因此,在编写用户程序时要把这些资源留给监控程序,以免死机。

BJS-51-PC 需要使用的资源如下:

1. 单片机内第三寄存器区(内部 RAM 18H-1FH)为监控工作区。

2. 位控单元 00H-0AH 为监控标志区。

3. 30H-41H 为监控程序用堆栈。

4. 串行口(RXD,TXD)用于与微机通信,传送命令及数据(在连续运行时用户也可用串行口)。

5. 定时器 T1,为串行通信的波特率发生器,因此有关的状态(TCON,TMOD,SCON,SMOD 中相应位)都不能修改。

6. 外部 ROM 空间中 0000H-1FFFH。用于放置监控程序。

7. 外部中断 0,用于单步运行控制。

8. 6000H-605FH 为监控程序的外部数据缓冲区,用以存放微机送来的命令、暂存数据以及用户的中断入口地址。

51 系列单片机的中断入口地址(0003H,000BH,0013H,001BH,0023H,002BH)均在监控程序地址空间中,用户要使用单片机的中断功能时就无法通过这些单元写入转移指令,转移到用户中断服务中去。为此,在监控程序中上述地址单元已分别写上长转移指令,把中断入口地址转移到用户能写入的 RAM 空间中。经监控程序处理后,在使用监控程序时,单片机的中断入口地址变为:

603BH 定时器中断入口地址

6043H 外部中断 1 中断入口地址

604BH 定时器 1 中断入口地址。

6053H 串行口中断入口地址。

605BH 定时器 2(8052 中)中断入口地址。

为此,用户在设计应用系统的硬件时,除考虑用户



本身的要求外,还需增加一些资源,它们是:

1. 在 0000H—1FFFH ROM 空间处配置一个 28 脚 IC 座,用于插入监控程序。

2. 在 6000H—7FFFH 空间处配置一片随机存储器 (6264),其读出应由 PSEN 及 RD 同时控制(即 PSEN 及 RD 通过与门后去控制)。

3. 把 8031 的串行口扩展为 RS232C 标准通信口。以便与微机的 RS232 口连通。

把 8031 的串行口扩展为 RS232C 标准口,可使用现成的集成芯片(MC1488,MC1489),但必需有 +12V 和 -12V 电源。因此更常用的方法是用二个晶体管组成的简易 RS232C 扩展电路(见图 2)。该电路借用微机 RS232C 中的 -12V 信号来建立负电源,因此该电路只需与 8031 共用一个 +5V 单电源即可工作。

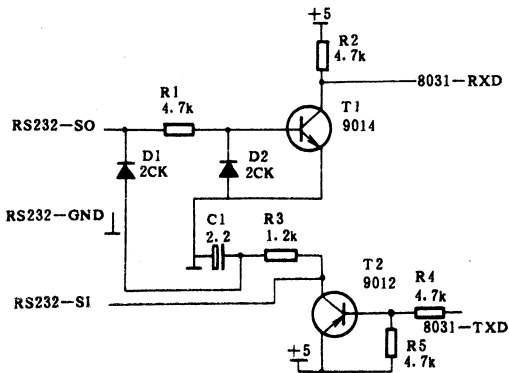


图 2 简易 RS232C 扩展电路

图中,SO 为微机串口发送端

SI 为微机串口接收端

GND 为微机串口地

在微机的 RS232 插座上分别是第 2、3、7 脚。

4. 若要使用监控程序中的打印驱动程序,打印反

汇编清单或其他调试信息,则还要配置一个打印机接口。打印机信号与 8031 的引脚连接如下:

|       |           |
|-------|-----------|
| 打印机   | 8031      |
| STB   | P3.4      |
| D0—D7 | P1.0—P1.7 |
| BUSY  | P3.3      |
| 地     | GND       |

5. 制作一块监控程序。读者可借助于仿真器把 BJS-51-PC 固化到一片 2764 中。只要固化无误,插入到用户系统后即可使用。需要 BJS-51-PC 的机器码清单可与作者联系。

## 二、BJS-51-LED

BJS-51-LED 是以 BJ-51 仿真器的监控程序为基础,根据 BJS-51 的特点修改而成。由于它以仿真器的监控程序为基础,因此它具有仿真器类似的功能。BJS-51-LED 的键功能见表 2。

表 2 BJS-51-LED 键功能表

| 键名  | 功能               |
|-----|------------------|
| WR1 | 写加 1             |
| ME  | 向 EEPROM 传送数据    |
| MOV | RAM 间传送数据        |
| F1  | 设置数据块首地址         |
| F2  | 设置数据块末地址         |
| RDI | 读加 1             |
| RDS | 读减 1             |
| EPR | EPROM 操作         |
| FI  | 数据块填充            |
| INS | 插入一条指令           |
| BP  | 断点运行             |
| FU  | 多功能键(反汇编,数据块搜索等) |
| C-C | 数据块比较            |
| EXE | 连续运行用户程序         |
| MON | 返回监控             |

(接第 45 页)

10. 发光二极管电子广告牌 12,000.00 元/m<sup>2</sup>  
 地址: 武汉武昌珞瑜路 100-1 号  
 开户银行: 武汉市洪山区科技信用社  
 帐号: 967-517

邮编: 430070 电话: (027)712761

传真: (027)701803 联系人: 姜艳艳

注:《CYSCB-2 MCS-51、8098 单片单板机硬件、软件设计原理》将在本刊第 10 期登出。

(接第 25 页)

IS A HARD COPY NEEDED?

ENTER (Y/N)

? Y (回车)

\*\*\*\*\* WARNING \*\*\*\*\*

ALL DATA ON THE FIXED DISK WILL BE DESTROYED  
 DO YOU WANT TO CONTINUE?

ENTER (Y/N)

? Y (回车)

ALL DATA ON DRICVEC WILL BE DESTROYED !

THIS YOUR LAST CHANGE TO CANCEL !

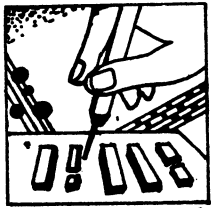
DO YOU WANT TO CONTINUE?

ENTER (Y/N)

?

在这时键入 Y,即进行硬盘的初始化。

该诊断程序既可用在长城系列机 CH、DH、286B、286BH、286EX、386 等机型,也可以用于兼容机。



## 学装微电脑

# 水平多关节型机器人

易齐干

本文介绍水平多关节型机器人， $\mu\text{P}-80$  微电脑套件控制它的两个步进电机和一个电磁铁，实现类似手臂关节动作，在手指部位装上钢笔，能控制其写字。

水平多关节型机器人的结构大体由肩部、肘部、上臂、前臂和手指组成。简图如图 1 所示。

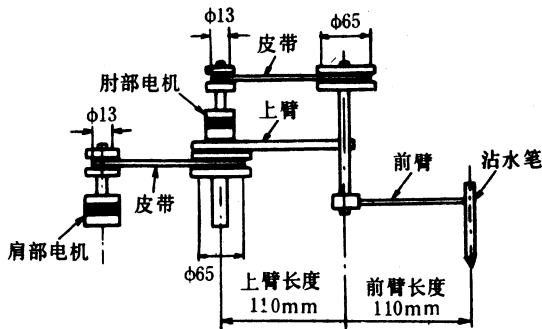


图 1

肩部和肘部驱动源均为 PXB43-01A 步进电机。上臂长度和前臂长度均为 110mm。分别用齿形皮带传递步进电机动力，带轮减速比为 1/5。手指由吸拉型直流电磁铁与弹簧控制。如写字时，手指夹持元珠笔，电磁铁断电，在弹簧作用下，元珠笔与纸接触，上臂和前臂摆动进行写字。如果直流电磁铁通电，将元珠笔拉起，则脱离与纸的接触。手指夹持元珠笔的移动范围如图 2 所示。

上臂和前臂位置夹角为  $60^\circ$  时，步进电机 1 个脉冲的移动量为最小移动量。如图 3 所示，计算方法：

$$110\text{mm} \times 1.8/360 \times 2\pi \div 5 = 0.69\text{mm}$$

式中：

1.8/360  $\times$  2 $\pi$ ：1 个脉冲的弧度

5：为减速比

最后，取最小移动量为 0.7mm

$\mu\text{P}-80$  微电脑套件控制步进电机与电磁铁的接口电路如图 4 所示。

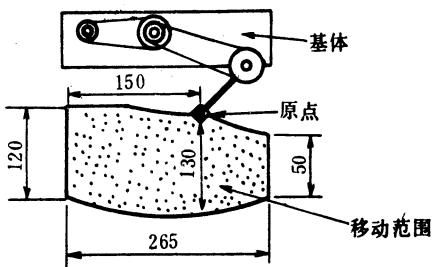


图 2

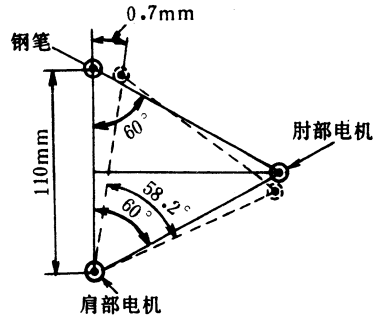


图 3

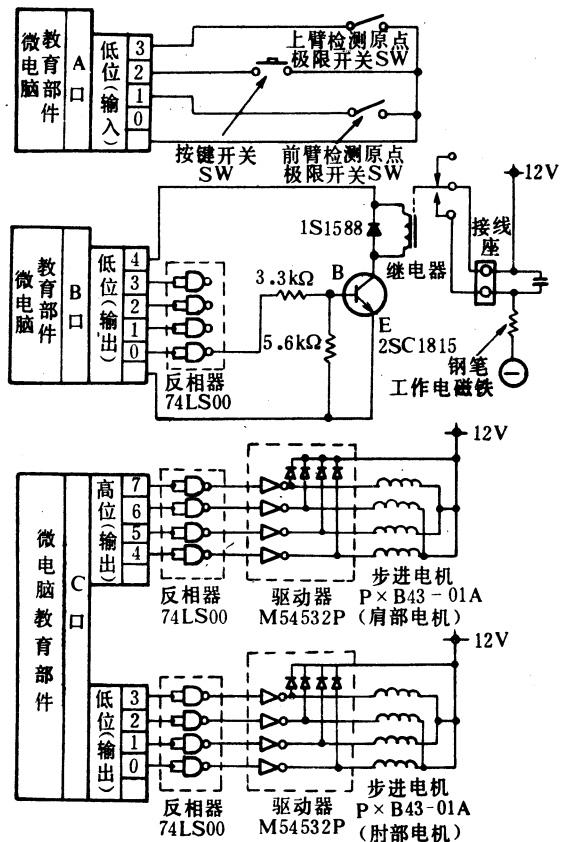


图 4

端口 C 高位控制肩部电机正、反转；端口 C 低位控制肘部电机正、反转。

端口 B 第 0 位输出 ON、OFF 信息，控制继电器部件 ON、OFF，驱动电磁铁，实现手指夹持元珠笔的上升和下降。

端口 A 的第 3 位、第 1 位分别接收上臂与前臂原点位置极限开关发出的信息。检测原点位置之后，按动启动开关，即开始写字。启动开关信息由端口 A 第 2 位接收。

操作过程是微电脑通过移位指令驱动肩部、肘部步进电机回转。端口 B 第 0 位输出 L 电平，经反相器反转为 H 电平驱动电磁铁工作。元珠笔移动到所要求的位置时，肩部电机与肘部电机暂停 0.5 秒。此时，直流电磁铁完成下降或上升动作。例如，画直线的时序是向肩部电机输入 1 个脉冲，向肘部电机输入两个脉冲。如图 5 所示。

介绍书写英文字母 P 的程序。在设置控制码、设置初始数据之后，有 4 个子程序，恢复原点、提笔、落笔、书写动作子程序。主程序流程图如图 6 所示。

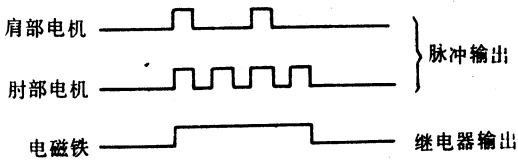


图 5

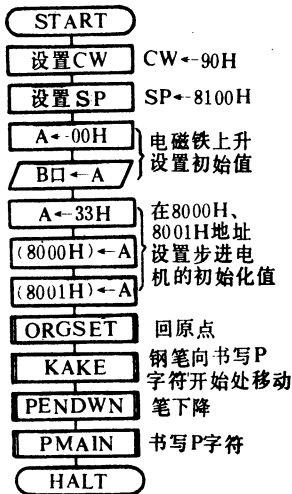
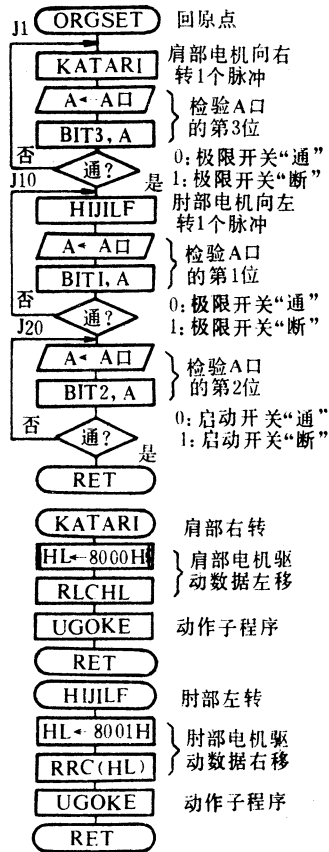


图 6

水平多关节型机器人动作起来，首先，必须决定原点位置。电源为 ON，肩部电机向右转，上臂移动到上臂原点极限开关为 ON 的位置。然后，肘部电机向左移、上臂移动到前臂原点极限开关为 ON 的位置。肘部电机继续左转，前臂移动到前臂原点极限开关为 ON 的位置。由极限开关决定的上臂和前臂的停止位置定为原点。

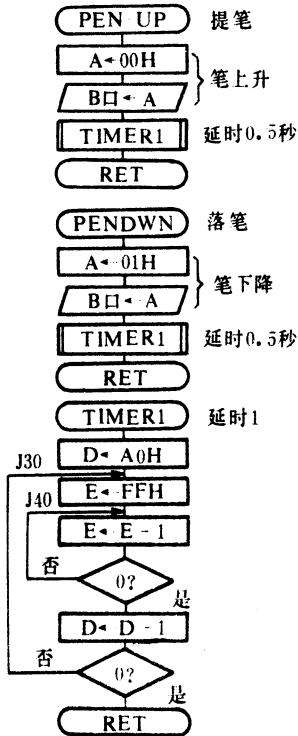


| 标记     | 助记符        | 地址     | 机器码    | 注释     |        |
|--------|------------|--------|--------|--------|--------|
| ORGSET | CALL       | KATARI | 20     | CD3C00 | 回原点    |
|        | INA. (00H) | 23     | DB60   |        | 检验原点   |
|        | BIT 3, A   | 25     | CB5F   |        | 极限开关   |
|        | FPXEJ1     | 27     | C22000 |        | 检验原点   |
| J10    | CALL       | HIJILF | 2A     | CD4500 |        |
|        | IXA. (00H) | 2D     | DB00   |        | 极限开关   |
|        | BIT 1, A   | 2F     | CB4F   |        |        |
|        | JPXEJ10    | 31     | C22A00 |        |        |
| J20    | IXA. (00H) | 34     | DB00   |        |        |
|        | BIT 2A     | 36     | CB57   |        | 检验启动   |
|        | JPNEJ20    | 38     | C23400 |        | 开关     |
|        | RET        | 3B     | C9     |        | 返回     |
| KATARI | LDHL       | 8000H  | 3C     | 210080 | 肩部电机   |
|        | RLC(HL)    | 3F     | CB06   |        | 驱动数据   |
|        | CALL       | LGOKE  | 41     | CD6B00 |        |
|        | RET        | 44     | C9     |        | 返回     |
| HIJILF | LDHLS      | 001H   | 45     | 210180 | 肘部电机   |
|        | RRC HL     | 48     | CBOE   |        | 驱动数据右移 |
|        | CALL       | LGOKE  | 4A     | CD6B00 |        |
|        | RET        | 4D     | C9     |        | 返回     |

图 7

检测到原点之后, 按压启动开关、书写开始。恢复原点程序流程图与程序如图 7 所示。

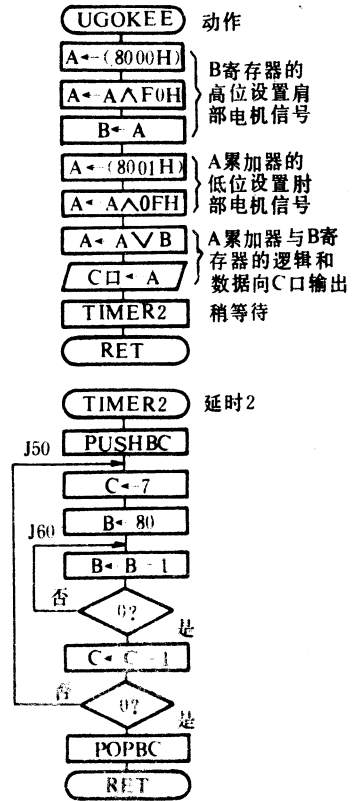
端口 B 向继电器输出 00H, 继电器为 ON, 输出 01H, 继电器为 OFF。电磁铁与继电器相对应, 产生 ON、OFF 变化, 完成提笔、落笔动作。笔划、文字要隔开, 在提笔、落笔之后有延时程序。提笔、落笔流程图如图 8 所示



| 标记     | 助记符          | 地址 | 机器码    | 注释   |
|--------|--------------|----|--------|------|
| PENUP  | LDA. 00H     | 4E | 3E00   | 提笔   |
|        | OUT (01H). A | 50 | D3 01  |      |
|        | CALL TIMER1  | 52 | CD5E00 | 暂时不变 |
|        | RET          | 55 | C9     | 返回   |
| PENDWN | LDA. 01H     | 56 | 3E01   | 落笔   |
|        | OUT (01H). A | 58 | D3 01  |      |
|        | CALL TIMER1  | 5A | CD5E00 | 暂时不变 |
|        | RET          | 5D | C9     | 返回   |
| TIMER1 | LDD. A0H     | 5E | 16A0   | 延时 1 |
| J30    | LDE. FFH     | 60 | 1EFF   |      |
| J40    | DECE         | 62 | ID     |      |
|        | JPNEJ40      | 63 | C26200 |      |
|        | DECD         | 66 | 15     |      |
|        | JPNEJ30      | 67 | C26000 |      |
|        | RET          | 6A | C9     | 返回   |

肩部电机、肘部电机动作程序上期讲座曾作介绍, 程序流程图如图 9 所示。

书写子程序中注意两个问题: 输入给肩部电机、肘

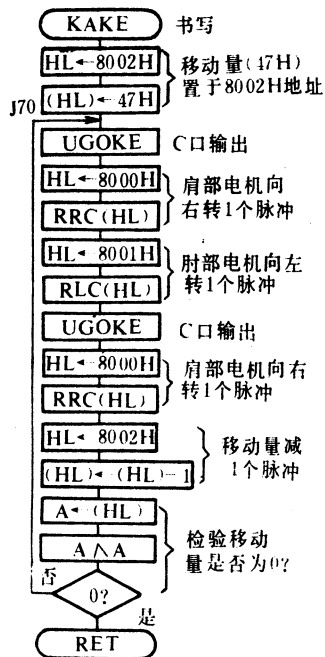


| 标记     | 助记符          | 地址 | 机器码     | 注释      |
|--------|--------------|----|---------|---------|
| UGOKEE | LDA, (8000H) | 6B | 3A00 80 | B 寄存器   |
|        | AND F0H      | 6E | E6F0    | 高位设置    |
|        | LDB, A       | 70 | 47      | 肩部电机信号  |
|        | LDA, (8001H) | 71 | 3A01 80 | B 寄存器低位 |
|        | AND 0FH      | 74 | E60F    | 设置肘部电机  |
|        | ORB          | 76 | B0      | 信号      |
|        | OUT (02H). A | 77 | D302    | 向 CD 输出 |
|        | CALL TIMER2  | 79 | CD7D00  | 稍等待     |
|        | RET          | 7C | C9      | 返回      |
| TIMERZ | PUSH BC      | 7D | C5      | 延时 2    |
|        | LDC, 7       | 7E | 0E 07   |         |
| J50    | LDB, 80H     | 80 | 06 80   |         |
| J60    | DEC B        | 82 | 05      |         |
|        | JPNE, J60    | 83 | C282 00 |         |
|        | DEC C        | 86 | 0D      |         |
|        | JPNE, J50    | 87 | C280 00 |         |
|        | POP BC       | 8A | C1      |         |
|        | RET          | 8B | C9      | 返回      |

部电机的脉冲数,决定上臂和前臂移动距离,即笔的书写长度。改变输入给两个步进电机的脉冲数比例,可改变笔划的倾斜角度。

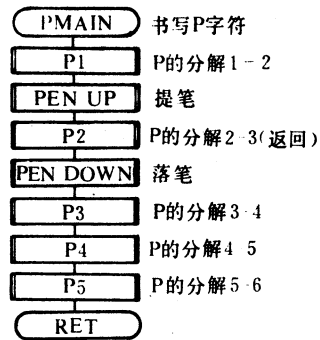
笔划方向由两个步进电机旋转方向的组合来决定。程序流程图如图 10 所示。书写英文字母 P, 首先将 P 字符分解。看出可以一笔写出, 分解过程与流程图如图 11 所示。

编出程序后, 使用反相器部件、H/L 电平显示部件

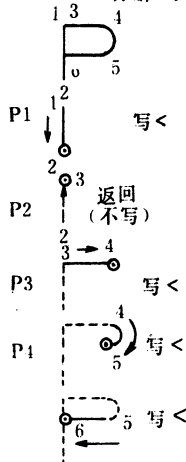


| 标记   | 助记符          | 地址 | 机器码      | 注释                   |
|------|--------------|----|----------|----------------------|
| KAKE | LDHL, 8002H  | 8C | 21 02 80 | 移动量为                 |
|      | LD(HL), 47A  | 8F | 36 47    | 47 脉冲                |
| J70  | CALL VGOKE   | 91 | CD 6B 00 |                      |
|      | LDHL, 8000H  | 94 | 21 00 80 | 肩部电机<br>向右移动<br>1 脉冲 |
|      | RRC(HL)      | 97 | CBOE     |                      |
|      | LDHL, 8001H  | 99 | 21 01 80 | 肘部电机<br>向左移动<br>1 脉冲 |
|      | RLC(HL)      | 9C | CB06     |                      |
|      | CALL VGOKE   | 9E | CD6B00   |                      |
|      | LDHL, 8000H  | A1 | 21 00 80 | 肩部电机<br>向右移动<br>脉冲   |
|      | RRC(HL)      | A4 | CB0E     |                      |
|      | LD HL, 8002H | A6 | 210280   | 移动量减<br>1 脉冲         |
|      | DEC(HL)      | A9 | 35       |                      |
|      | LDA, (HL)    | AA | 7E       | 检验移动<br>量是否为<br>0?   |
|      | ANDA         | AB | A7       |                      |
|      | JPNE, J70    | AC | C29100   |                      |
|      | RET          | AF | C9       | 返回                   |

图 10



\* 分解 P 字符, 为子程序化



| 标记    | 助记符          | 地址 | 机器码    | 注释   |
|-------|--------------|----|--------|------|
| PMAIN | CALL P1KAKE  | B0 | CD9001 | 写 P1 |
|       | CALL PENUP   | B3 | CD4E00 |      |
|       | CALL P2KAKE  | B6 | CDC501 | 返回   |
|       | CALL PENDOWN | B9 | CD5600 |      |
|       | CALL P3 KAKE | BC | CD2002 | 写 P3 |
|       | CALL P4 KAKE | BF | CD8002 | 写 P4 |
|       | CALL P5 KAKE | C2 | CDB002 | 写 P5 |
|       | RET          | C5 | C9     | 返回   |

图 11

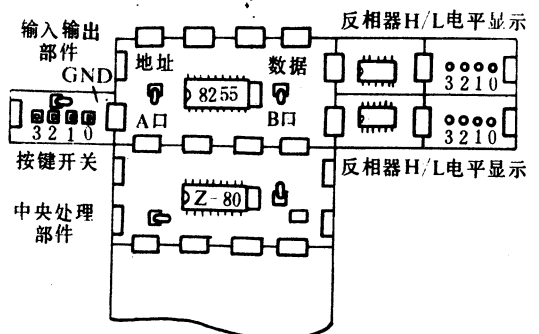


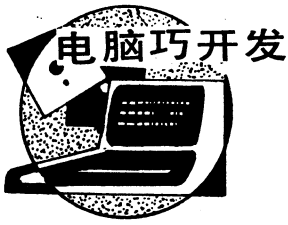
图 12

进行仿真。如图 12 所示。仿真实验时, 请将 TIMER2 延时子程序的常数 07H 改为 FFH, 仿真之后再改回来。

仿真结果可验证端口 C 高位相连的 H/L 电平显示部件的 LED 灯亮依次左移, 按键开关 S<sub>3</sub> 为 ON, 端口 C 低位相连的 H/L 电平显示部件的 LED 灯亮依次右移。

如果程序无误, 将水平多关节机器人接通 DC12V 电源与  $\mu\text{p}-80$  微电脑套件相连, 执行程序, 则在低上写出英文字母 P。

水平多关节型机器人也可用于电子元件组装、涂刷粘剂、粉刷油漆等工作。有兴趣的读者不妨可进行实验。订购该机器人请与天津纺织工学院机械系高殿斌同志联系, 邮码 300160。



# 点阵字符液晶显示器与单片机接口

合肥中国科技大学计算中心(230026) 张培仁 杨建景

目前,液晶显示器(LCD)以其低功耗、平板化、无X射线和电磁辐射等一系列显著优点,已取代了其它显示器件成为继CRT之后的第二大显示器件。

液晶显示器件广泛应用于各种仪器、仪表、电子显示装置、计算机显示终端、电子打字机等诸方面。LCD的图形、汉字显示功能亦有极大的推广应用价值。

本文将介绍点阵字符液晶显示器和单片机的接口技术。

## 一、基本结构

在液晶板上排列着两行 $5 \times 7$ 点阵的字符显示位,每行可显示12位。内存180多种字符包括英文大小写字母、数字和书写符号等,用户还可自定义8个 $5 \times 7$ 点阵的字符。PCB板上有14个引线端,其中八位数据线、三条控制线(见表1),可与微处理器相联(见图1),通过送入指令和数据可对显示内容、显示方式作出选择。

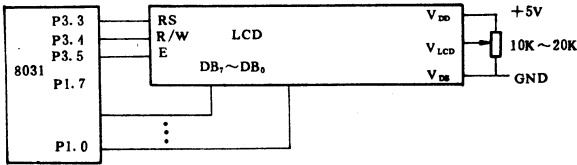


图1 与8位单片机联接示意图

表1 引线功能

| 引线号 | 符号               | 名称     | 功能                 |
|-----|------------------|--------|--------------------|
| 1   | V <sub>SS</sub>  | 地      | 0V                 |
| 2   | V <sub>DD</sub>  | 电路电源   | 5V±5%              |
| 3   | V <sub>LCD</sub> | 液晶驱动电压 | 见图1                |
| 4   | RS               | 寄存器选择  | H 数据寄存器<br>L 指令寄存器 |
| 5   | R/W              | 读/写    | H 读 L 写            |
| 6   | E                | 使能     | 下降沿触发              |
| 7   | DB0              | 8位数据线  | 数据传输               |
| 8   | ·                |        |                    |
| 9   | ·                |        |                    |
| 10  | ·                |        |                    |
| 11  | ·                |        |                    |
| 12  | ·                |        |                    |
| 13  | ·                |        |                    |
| 14  | DB7              |        |                    |

## 二、内部结构

DMC202(日本的HD4478D)是字符点阵式的LCD。它采用CMOS工艺的超大规模集成电路,它的内部结构框图示于图2。

## 三、指令描述

DMC202点阵式LCD,有11条指令,可以编程控制LCD的字符显示。

### 1. 清屏

指令码:0 0 0 0 0 0 0 0 0 0 1

对应位:RS R/W<sub>7</sub> D<sub>6</sub> D<sub>5</sub> D<sub>4</sub> D<sub>3</sub> D<sub>2</sub> D<sub>1</sub> D<sub>0</sub>(下同)

该指令的执行实际上是写空格码“20H”到所有的DDRAM中。DRAM地址计数器置0,如果光标移位,则回复到屏幕左上角。

### 2. 返回

指令码:00000001\*

\*表示无效位(下同),DDRAM地址计数器置0,DDRAM内容不变,光标回到第1行第1列。

### 3. 设置输入方式

指令码:00000001 I/D S

I/D: I/D=1,当将一字符写入DDRAM,地址增1,光标右移

I/D=0,当从DDRAM读出一字符码时,地址减1,光标左移

CGRAM的读写与DDRAM相同。

S: S=1,整个显示左移(I/D=1),右移(I/D=0)

### 4. 显示开/关控制

指令码:0000001D C B

D: D=1,显示开

D=0,显示关(显示数据保留在DDRAM中,当D=1时数据又立即显示出来。

C: C=1,显示光标

C=0,不显示光标

B: B=1,光标闪烁

B=0,光标不闪烁

### 5. 光标或显示移位

指令码:000001 S/C R/L \*\*

当不写数据时光标显示仍可移位,该功能用于改错或寻找显示的数据。

S/C R/L

0 0 光标左移(AC减1)

0 1 光标右移(AC增1)

1 0 所有显示左移,光标随显示移位

1 1 所有显示右移,光标随显示移位

### 6. 功能设置

指令码:00001 IF N 1 \*\*

IF: 设置接口数据长度

IF=1,8位数据接口

IF=0,4位数据接口,数据发送接收必须进行2

次(用 D<sub>7</sub>~D<sub>4</sub>)

N:设置显示行数

N=1,2 行显示

N=0,1 行显示

7. 设置 CGRAM 地址

指令码:0001 AAAAAA

用二进制数 AAAAAA 设置 CGRAM 地址,数据从 CPU 读或写入 CGRAM

8. 设置 DDRAM 地址

指令码:001 AAAAAA

用二进制数 AAAAAA 设置 DDRAM 地址

9. 读忙标志及地址

指令码:01BFAAAAAA

系统内部正在执行前面接受的指令(BF=1),不接受下一条指令。在下一字操作之前,检查 BF=0,读出地址计数器的数值是 AAAAAA,由前条指令决定地址是 DDRAM 或 CGRAM 的

10. 给 CGRAM 或 DDRAM 写数据

指令码:10DDDDDDDD

将二进制 8 位数据 DDDDDDDD 写入 CGRAM 或 DDRAM。

11. 从 CGRAM 或 DDRAM 读数据

指令码:11DDDDDDDD

从 CGRAM 或 DDRAM 读二进制 8 位数据 DDDDDDDD。

需要注意的是,上述 11 条指令中,第 1、2 条指令的执行周期为 1.64ms,而其余 9 条指令则需 40μs。

#### 四、读写时序

程序清单:

```
ORG 2000H
ANL IE, # 11110001B
ANL P3, # 11000111B
CPL P3.5
MOV P1, # 00000001B
CPL P3.5 ;清屏
CPL P3.5
```

```
MOV P1, # 00111000B
CPL P3.5 ;功能设置
CPL P3.5
MOV P1, # 00001110B
CPL P3.5 ;显示开关设置
CPL P3.5
MOV P1, # 00000110B
CPL P3.5 ;输入方式设置
CPL P3.5
MOV P1, # 01000000B
CPL P3.5 ;CGRAM 地址设置
CPL P3.5
SETB P3.3
MOV P1, # 00000100B ;自编字符开始
CPL P3.5
CPL P3.5
MOV P1, # 00100100B
CPL P3.5
CPL P3.5
MOV P1, # 01000100B
CPL P3.5
CPL P3.5
MOV P1, # 01100100B
CPL P3.5
CPL P3.5
MOV P1, # 10000100B
CPL P3.5
CPL P3.5
MOV P1, # 10100000B
CPL P3.5
CPL P3.5
MOV P1, # 11000100B
CPL P3.5
CPL P3.5
MOV P1, # 11100000B
CPL P3.5
CPL P3.5
CLR P3.3 ;自编字符结束
```

```
MOV P1, # 10000000B
CPL P3.5 ;DDRAM 地址设置
CPL P3.5
SETB P3.3
MOV P1, # 01010111B
CPL P3.5 ;显示“W”
CPL P3.5
MOV P1, # 01000101B
CPL P3.5 ;显示“E”
CPL P3.5
MOV P1, # 01001100B
CPL P3.5 ;显示“L”
CPL P3.5
MOV P1, # 01000011B
CPL P3.5 ;显示“C”
CPL P3.5
MOV P1, # 01001111B
```

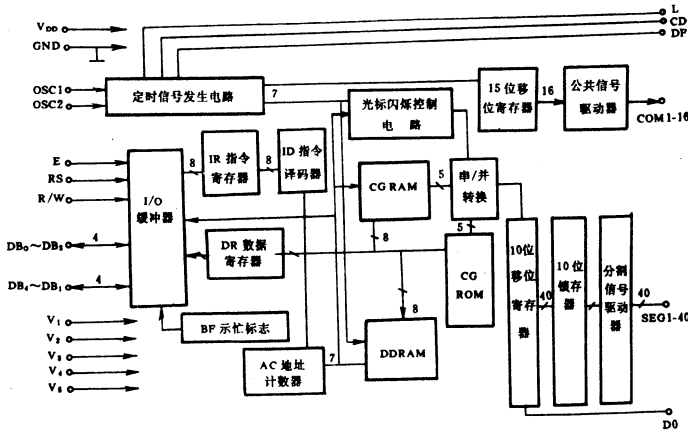


图 2 DMC202 LCD 结构框图

```

CPL P3.5 ;显示“O”
CPL P3.5
MOV P1,#01001101B
CPL P3.5 ;显示“M”
CPL P3.5
MOV P1,#01000101B
CPL P3.5 ;显示“E”
CPL P3.5
MOV P1,#00100000B
CPL P3.5 ;显示“空格”
CPL P3.5
MOV P1,#01010100B
CPL P3.5 ;显示“T”
CPL P3.5
MOV P1,#01001111B
CPL P3.5 ;显示“O”
CPL P3.5
CLR P3.3
MOV P1,#11000000B
CPL P3.5 ;DDRAM 地址设置,换行
CPL P3.5
SETB P3.3
MOV P1,#01010101B
CPL P3.5 ;显示“U”
CPL P3.5
MOV P1,#01010011B
CPL P3.5 ;显示“S”
CPL P3.5
MOV P1,#01010100B
CPL P3.5 ;显示“T”
CPL P3.5
MOV P1,#01000011B
CPL P3.5 ;显示“C”
CPL P3.5
MOV P1,#00100000B
CPL P3.5 ;显示“空格”
CPL P3.5
MOV P1,#01000011B
CPL P3.5 ;显示“C”
CPL P3.5
MOV P1,#01000011B
CPL P3.5 ;显示“C”
CPL P3.5

```

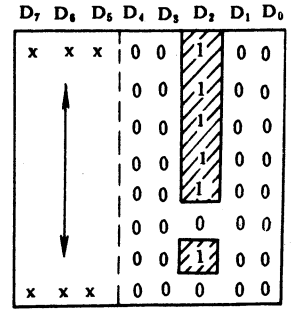
```

CLR P3.3
MOV P1,#00001011B
CPL P3.5 ;关闭整体显示开关,显示字符消失
CPL P3.5
MOV P1,#00001111B
CPL P3.5 ;打开整体显示开关,显示字符出现
CPL P3.5
MOV P1,#00000111B
CPL P3.5 ;设置成整体移动
CPL P3.5
SETB P3.3
MOV P1,#00000000B
CPL P3.5 ;显示自编字符“!”
CPL P3.5
CLR P3.3
MOV P1,#00011100B
CPL P3.5 ;整体向右移
CPL P3.5
MOV P1,#00011100B
CPL P3.5 ;整体向右移
CPL P3.5
MOV P1,#00000010B
CPL P3.5 ;返回
RET

```

程序说明:

1. 因为清屏、返回指令的执行需 1.64ms,其它 9 条指令的执行需 40μs,所以本程序最好采用单步执行,否则必须传递数据前后加延时程序。
2. 自编字符的 CGRAM Data,本程序自编字符“!”,其 CGRAM Data 为:



## 消 息

电子工业出版社《电子与电脑》杂志编辑部将于十月初在北京举办为期三周的“现代管理技术——计算机软件基础培训班”。

培训内容:微机 DOS 操作系统、字处理软件、FoxBASE 数据库。

培训费:330 元(含上机费。每个学员每天上机 2 小时)

资料费:50 元(实报实销)  
 联系人:中国人民大学信息中心(100872) 张红卫  
 电话:255.5431-2516(或 2514)

\*\*\*\*\*

## 敬告读者

本刊原有的程序清单已全部售完,请读者不要再汇款至我部。

编辑部





# F BASIC 语言的游戏程序编写技巧

电脑游戏机

山东苍山机械电子化学工业局(277700) 于春

许多玩过电脑游戏的朋友提出这样的疑问:丰富多彩的游戏画面是怎样形成的?游戏里的主人翁为什么会受操纵器的控制?一盒录像带最多放像2~3个小时,而很小的游戏卡为什么放起来没有完?读了“键盘电视游戏机的F BASIC语言及程序设计”一文后,基本清楚了游戏是通过编制程序实现的。因此,不仅跃跃欲试,想自己动手编一个游戏程序。但想时容易做时难,多次尝试,都不成功,实在令人沮丧。实际上,游戏程序有它独特的程序结构和编写技巧。不得要领,贸然编程,即使有经验的编程老手,也往往走弯路,何况我们初学者呢。

本文拟分游戏程序的概念、故事情节和游戏结构、程序结构和程序框图、游戏程序的设计过程、游戏程序的编制方法、游戏程序实例分析六讲,深入浅出,由简单到复杂,通过“成龙救金凤”游戏程序的编写,介绍用F BASIC语言编写游戏程序的方法和技巧。同时,每一讲还将穿插介绍部分趣味游戏程序实例,供大家练习。最后一讲将和读者一起分析几个完整游戏程序的结构和编程技巧。从而使读者逐步弄通游戏程序的结构,逐步掌握游戏程序的设计过程,功能模块的划分及程序编写、子程序的调用链接技术,最终自己编出满意的游戏程序。

## 第一讲 游戏程序的概念

我们先通过两个简单的游戏实例,认识游戏程序。

### 例一、猜数游戏

这是一个智能类游戏。本游戏由电脑产生一个随机数字,告诉你猜数的数值范围和最多允许试猜的次数,由游戏者使用键盘输入方式猜数。若在规定的次数猜中,则奏一段明亮的音乐向你祝贺,然后进入下一关;若猜不中则放一段低沉的音乐,并提示你所猜的数值与谜底数值的比较情况,若猜的数小则提示你猜大一点的数,否则猜小点的数。若在规定的次数没有猜中,则询问是否继续游戏。你共有三次机会,若三次都没有在规定的次数内猜中则结束游戏。本游戏共11关,猜数的范围一关比一关大,难度越来越高。你若能连破11关,则高奏欢快的乐曲祝贺你的成功。

运行程序,在一段欢快的音乐声中显示出标题画面:

```

* GUESS NUMBER *

```

Please check;

STAGE 1--11?

画面为蓝色,其中标题 GUESS NUMBER 为黄色、提示请你选择1~11关中的任一关。假如选3,回车后,则清屏,显示:

```

 STAGE 3

```

IN DATA<128 MAX TIME=7

TIME=1 IN DATA=?

电脑提示,现在是第三关,猜数范围小于128,最多允许猜7次,现在是第一次,请输入试猜数。假如电脑选的数是70,而我们猜的数是80,则演奏一段较低沉的音乐,在屏幕底行显示

80 is big! Give a smaller data

提醒“80大了,给一个小点的数”。这时输入提示行变为

TIME=2 IN DATA=?

如果我们猜中了70,则演奏一段明亮的音乐,清屏,显示红色的:

YOU ARE RIGHT!!!

PLEASE GO INTO STAGE 4

“你猜对了!请进入第4关”。然后清屏,显示第4关画面。

程序清单如下:

```
5 REM "NO. 1 GUESS NUMBER"
10 CLS;CLEAR;CGEN(2);A=32;B=5;G=0;A$="*****
 *****"
15 FOR I=8 TO 20;COLOR I,6,2;NEXT
20 LOCATE 5,4;PRINT A$;LOCATE 5,6;PRINT "└┐
 GUESS└NUMBER└ *";LOCATE 5,8;PRINT A$;
 LOCATE 7,12;PRINT "Please└check;";
25 LOCATE 7,16;PRINT "STAGE└└1└└11└└";GOSUB
 250
30 INPUT E;IF E=11 THEN A=32767;B=15;GOTO 70
40 IF E=1 THEN 70
50 FOR I=2 TO 10;A=A+A;B=B+1;IF I=E THEN 70
60 NEXT
70 X=RND(A);F=0
80 CLS;LOCATE 5,4;PRINT A$;LOCATE 9,6;PRINT
 "STAGE└└└└└B-4;LOCATE 5,8;PRINT A$;LO-
 CATE 0,12;PRINT "IN└DATA└<;"A;"└MAX└
 TIME="B
90 F=F+1;IF F>B THEN 170
95 LOCATE 0,16;PRINT "└└└└└└└"(28个空格)
100 LOCATE 0,16;PRINT "TIME=";F;"└└IN└DATA
 =";INPUT D
110 IF D<>X THEN 200
```

```

120 IF A=32767 THEN CLS;FOR I=5 TO 20,COLOR I,7,
1,1;NEXT ;LOCATE 5,7;PRINT"CONGRATULATION
!!!";GOSUB 250;FOR I=0 TO 10000;NEXT;GO-
TO 180
130 CLS;FOR I=5 TO 20,COLOR I,7,1;NEXT;LOCATE 5,
7;PRINT"YOU ARE RIGHT!!!";GOSUB 400
140 LOCATE 5,15;PRINT"PLEASE GO INTO STAGE
" B-3;FOR I=0 TO 10000;NEXT
150 IF A=16384 THEN A=32767;B=B+1;GOTO 70
160 A=A*2;B=B+1;GOTO 70
170 CLS;G=G+1;LOCATE 5,5;PRINT"TIME IS OVER
";B;!!!";LOCATE 4,10;PRINT"ON OR
END?(Y/N)";Y$=INKEY$(0);IF Y$<>"N"
AND G<4 THEN 70
180 CLS;LOCATE 8,8;PRINT"GOOD--BY!!!";
LOCATE 24,20;PRINT" ";END
200 LOCATE 0,20;PRINT"....."(38个空格);GOSUB
350;LOCATE 0,20
210 IF D>>X THEN PRINT D;"is big! Give a smaller
data";GOTO 90
220 PRINT D;"is small! Give a bigger data";GOTO 90
250 PLAY"MIV12 Y3T2;M0V9M1V9Y2T2;M1T2"
260 PLAY"O3E5GA;O2G5O3CC;O2C5RE"
270 PLAY"O4C6D3O3AO4C;O2GO3EC;GRE"
280 PLAY"O3G5E6G3;O2GGO3C;CRE"
290 PLAY"DCDEG9;O2GO3CCO2GGO3C;GRECGC"
300 PLAY"E5GAO4C6D3;O2GO3CCO2GO3E;CREGR"
310 PLAY"O3AO4CO3G5E6G3;CO2GGO3C;ECRE"
320 PLAY"DCDECE9;O2GO3CC9;GRECGE"
330 RETURN
350 PLAY"M1V12Y0T2;M0V9Y2T2;MIT2"
360 PLAY"O2B3ABAGA;O2C5EG;O1G5O2C3CC5"
370 PLAY"GAO3CO2AO3CR;EC7;O1G3GG7"
380 RETURN
400 PLAY"M1V7Y1T3;M0V7M1V6Y2T2;MIT2"
410 PLAY"O3G3O4CO3GO4C;O3C5E;O2G5C3C"
420 PLAY"O3AO4CO3AGEG;GEC;C5G3GG5"
430 PLAY"EDCEC;O2GAG;C3CC5G3G"
440 RETURN

```

程序说明:

1. 15、20 行打印标题画面。其中 15 行使标题“GUESS NUMBER”(猜数)显示黄色,以使标题醒目。
2. 25~60 行为游戏者选关处理程序。游戏者可在 1~11 关中任选。从 1 关至 11 关猜数范围依次是 32、64、128、256、512、1024、2048、4096、8192、16384、32767。
3. 70~100 行,游戏者在选关后,电脑根据所选关次自动生成随机数,要求游戏者输入试猜数。
4. 110 行是没有猜中的转移处理程序。
5. 120~160 行是猜中处理程序。其中 120 行是冲破 11 关的处理程序。这时,显示红色的贺词“CONGRATULATION!!!”意思是祝贺你的成功! 130~160 行是升级处理语句,即进入下一关。每关的猜数范围均是上一关的两倍(第 1、11 关除外)。在第 10 关升第 11

关时,由于  $16384 \times 2 = 32768$  已超出 F BASIC 取值上限,将因产生溢出而停机,因此设置了 150 句进行拦截处理。使第 10 关升 11 关时,直接令  $A=32767$ ,而不经过 160 句。

6. 170~180 行是结束处理程序。其中 170 行记录了游戏的次数。程序中共给游戏者三次接关的机会,若接关 3 次没有猜中,则结束。

7. 200~220 提示猜数的方向。

8. 250~330 行为标题、祝贺音乐子程序。

9. 350~380 行为猜错音乐子程序。

10. 400~440 行为猜对音乐子程序。

注——程序中所有提示均使用了英文。读者如感不便,可换成汉语拼音。“ ”表示空格。

你能连破 11 关吗? 请上机试一试。

结果怎么样? 你可能侥幸猜中一、两次,但大多数猜不中,很难升级到下一关。那么能否顺利地连破 11 关呢? 回答是肯定的。

我们以第一关为例说明实现的方法。假设电脑产生的随机数是 19。由于第一关的猜数范围是 32,所以第一次我们选 0~32 中间的数,猜 16,这时电脑提示“16 小了,请猜大一点的数。”这说明所猜的数在 16~32 之间,范围缩小了一倍。第二次我们猜 16~32 的中间数 24,则电脑提示“24 大了,请猜小一点的数。”这说明所猜的数在 16~24 之间,猜数范围又缩小了一倍。第三次我们仍猜中间数 20,电脑提示“20 大了,请猜小一点的数”。第四次我们猜 16~20 的中间数 18,电脑提示“18 小了,请给大一点的数”。第五次我们猜 18~20 之间的数 19,则必然猜中。

从上述猜数过程看,每猜一次,猜数范围缩小一次,第一关允许猜五次,正好把猜数范围缩小到 1 个数。所以不论电脑选中是 32 之内的哪个数,最多五次必然猜中。如果是第二关,猜数范围是 0~64,那么我们第一次猜 32 把猜数范围缩小一倍,又与第一关相同了。由于每升一关,允许猜数次数增加一次,所以,关关能破,必能顺利通过 11 关。

以上的猜数方法,在计算机术语中叫折半查找法,是计算机数据查找技术中常用的方法。其原理已超过本文范畴;有兴趣的读者可参阅有关书籍,不再赘述。

现在你已经掌握了猜数规律,验证一下,是否实用有效。

## 例二 爱的结合

这是一个由电脑自动完成的动画游戏,游戏开始在简短的音乐声中,屏幕上画出了鲜红的心形图案,随后从屏幕的左下角、右下角分别走出了玛丽哥哥和丽莎小姐,两人相向而行,相遇后,显示大红的背景,奏出欢快的音乐,打印出爱的结合的祝词,结合的爱心放射出耀眼的光芒。程序清单如下:

```

5 REM"NO. 2 PERFECT"
10 PL."M1V9Y2T3;M1V7Y1T3;M1T3"
15 PL."O3E5G;O3C5;O2G5G"
20 PL."O4CO3B4A1;CC;#F#F"

```

(转第 6 页)



## 单色显示器故障维修一例

广东珠海拱北海关技术处(519020) 梁建华

### 故障现象:

一台 OPCON 牌(台湾产)单色显示器,按下开关后,无光栅,且指示灯不亮。

### 维修:

该显示器是采用开关电源来为行、场电路提供电压的,由于按下电源开关后,指示灯不亮,而该指示灯的电压是经开关电源整流后提供的。经检查,电源开关及整流部分均正常,初步确认为开关电源部分的故障。

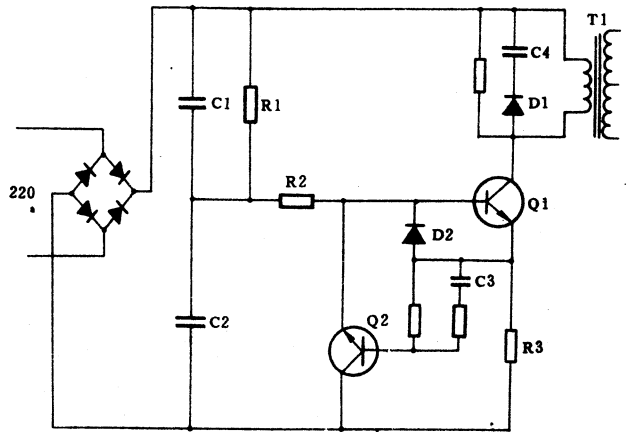
接通电源,用示波器检测开关管 Q1 的 C 极,没有锯齿波,开关管没有起振,说明是开关管及其控制部分的故障。拔掉电源,用万用表测量该部分电路,发现如图中的 Q2 管 CE 极短路,电阻 R3 开路,更换这两个元件后,再加电测试,发现有时开机正常,有时开机则无光栅,指示灯亦不亮,在出现故障的时候,关掉电源,用万用表测量时,发现当表笔碰到 Q1 的 C 极时,指示灯瞬间亮一下,说明电容所存的电荷还没有泄放完,开关管处于临界导通状态,用万用表测量开关管附近的元件,发现电阻 R2 的阻值变大,用原阻值的电阻替换后,每次开机均显示正常。

### 分析:

由于电压变化,电阻 R3 被烧开路,导致 Q2 管的 CE 极短路,而 R2 阻值变大,开关管 B 极电压降低,使

开关管有时不能导通,造成开机显示时有时无。值得注意的是,R2 阻值变大,往往很易忽略,由于显示时有时无,很容易错误地认为是电源开关的问题,这时,应仔细测量整流管后的电路是否有电压,开关管是否起振,如果确认故障是由开关管不起振造成的,应重点查一下与开关管导通、截止有关的元器件。

注:部分无关的线路没有画出。



图

## 恢复 CEC—I 学习机 DOS 的 I/O 控制

柳州钢铁厂技工学校(545002) 屈晓柳

把 DOS3.3 磁盘操作系统从磁盘装入 CEC—I 内存后,在 BASIC 程序中选择 I/O 输入输出设备时,须将“IN#S”和“PR#S”命令作为一条特殊的 BASIC 语句被调用,即命令的前面要加入 CTRL-D 或 CHR\$(4)。

语句格式:

[行号]PRINT “CTRL-D IN#S/PR#S”

或 [行号]PRINT CHR\$(4);“IN#S/PR#S”;

PRINT

说明:参数 S 表示外设所占用的槽口号,S=3 表示进入汉字系统;S=1 表示接通打印机。

如果没有把“IN#S/PR#S”命令当作特殊 BASIC 语

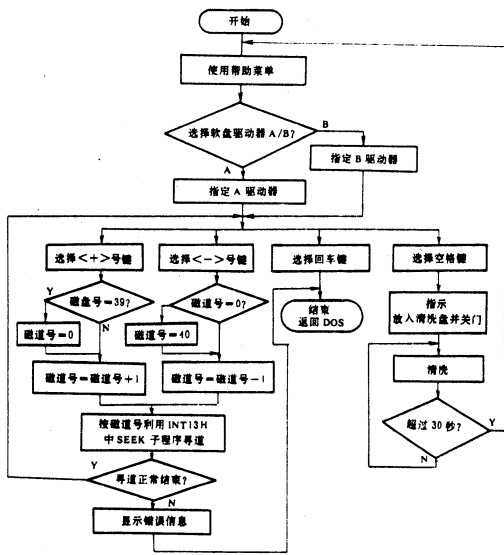
句处理,而运行含有“IN#S/PR#S”语句的程序,DOS 命令将失效,产生的原因,是监控程序的 I/O 控制寄存器 KSW(地址为 \$38,\$39 单元)和 CSW(地址为 \$36,\$37 单元)的内容指出的不是 DOS 操作系统下的 I/O 控制寄存器 KSW 和 CSW 的地址(\$9E81,\$9EBD)。

恢复 DOS 命令的功能,须改写 \$36~\$39 单元的内容,使监控程序的 I/O 控制寄存器 KSW 和 CSW 指向 DOS 操作系统下的 I/O 控制寄存器 KSW 和 CSW。即:在 \$36,\$37,\$38,\$39 单元中顺次送入如下数据 \$BD,\$9E,\$81,\$9E,更简单的办法是在监控状态下执行 3D0G↵,这样系统将恢复 DOS 命令的功能。

# 再谈清洗盘的正确使用

安徽蚌埠 86181 部队基础教研室(233000) 刘德一

本刊 1991 年 10 期中《怎样使用清洗盘》一文,简单地介绍了清洗盘的使用方法,但忽略了一个很重要的问题,即用什么命令来调盘进行清洗。目前大多数用户均采用 DIR 命令,但它只能使磁头在清洗盘的 00 道上清洗,多次使用就会造成“越擦越脏”的现象。



清洗盘的正确使用应遵循以下原则:

1. 仔细阅读使用说明,因为目前清洗盘有干式和湿式两种,其使用方法也不尽相同。
2. 清洗时间一般在 30~60 秒之间,时间太长会损坏磁头。
3. 不能反复使用清洗盘的某一磁道,最好第一次用 00 道,第二次用 01 道,依次类推,并记录在清洗盘标签上,以便下次使用时参考。
4. 清洗盘的使用寿命一般为 15 次。

为解决磁头定位到某一磁道问题,笔者用汇编语言编写了磁头清洗软件,由于篇幅有限,这里只能介绍该软件的粗框图及编程时应注意的问题。完整的源程序已收入本人编写的《PC 机用户实用维修技法》一书,另备有软件,有兴趣者可与本人联系。

在编程时应注意以下几点:

1. 应在驱动器门打开的情况下选择磁道,以免磁头在寻道时受到损伤。
2. 正向寻道时不能超过 39 道,反向寻道时不能小于 00 道,以免磁头“内撞”或“外撞”。
3. 磁道确定后,再提示放入清洗盘并关好驱动器门,然后开始清洗。
4. 清洗时间最好让操作者随意选择。

# 对磁盘局部缺损的处理

广东汕头金山中学(515036) 何管略

DOS 在格式化磁盘时,会将缺损的簇号用 FF7 作标志,自动登录在文件分配表 FAT 中,此后的读、写操作,都不理会这些坏簇。

但是,磁盘使用一段时间之后,难免出现新的缺损簇号。轻度缺损,只引起读盘出错;严重缺损,将导致每次的读、写操作,都报出错。DOS 对格式化以后新出现的缺损簇号,没有提供检查、登录的功能。Copy 命令中的 /V 开关,只能帮助写盘,不会帮助读盘。Chkdsk 命令也无能为力,因为它只检查 FAT 表中的各项标志,而不检查数据区中的缺损情况。

对于这类有局部缺损磁盘,丢弃不用,似乎可惜;继续使用,又会出错,干扰正常操作,实在令人左右两难。我们从实践中摸索出几点处理体会,简介于下:

- (1) 局部封存文件。如果只在个别的小型文件中,

读盘出错的话,可将该文件改名(最好将它置为不可见属性),此后对它不删、不读、不写,作坏簇处理。

- (2) 重新格式化。如果有多处或对大型文件读盘出错,宜先将其它完好的文件,转录到另外的软盘上,再将该软盘重新格式化,让 DOS 去自动登录坏簇。

- (3) 补录缺损簇号。对于硬盘来说,重新格式化前后的工作量,可能很大,令人生畏。最好调用工具软件 PCTools 中的 DiskVerify 功能,它会从已用的和未用的全部簇号中,将格式化以后新出现的所有坏簇,逐个检查出来,分别补记在 FAT 表中。工具软件的不同版本,功能基本相同;区别只在于版本较高(如 6.0),速度较快,菜单位置,稍有不同而已。此法功能强大,操作简便,软、硬盘都可适用,效果很好。

# 电子工业出版社软件部新出版软件介绍

## 一、PC 及其兼容机软件

### 1. C 语言科学与工程程序库 软盘:1片 定价:85元

本软件以 C 语言函数的形式,针对科学研究与工程应用领域中的数据分析与数据处理及工程图形等实际问题,编写了数理统计,线性规划、特殊函数,非线性方程求解、工程图形、三维图形及异步通信等十几类程序。程序经过严格测试,准确无误,具有很强的实用性。

本软件可运行于各类微型计算机。

\* 与本软件配套的“C 语言科学与工程程序库”一书由电子工业出版社同时出版。

### 2. 小学一、二年级数学语文练习 软盘:1片 定价:70元

本软件是按全国统一教材和教学大纲内容的要求编制的。数学部分可在 100 以内随机出题(加减法),每道练习题有解答和对与错的判断,每次可演算 10 道题。语文部分包括一年级的 630 个汉字和二年级的 817 个汉字拼读等练习。

## 二、中华学习机及其兼容机

### 1. 拼注练习 软盘 1片 定价:35元

本软件将学习寓于娱乐之中,集动画、音响于一体。它包括单音和多音汉字的拼音与注音、文件编辑(建立练习答案,查看和修改答案等)和打印答案。

该软件需配 STC2.0 汉字库和两台软驱动器。

### 2. 柴油机 软盘:1片 定价:30元

本软件是初中物理教学课件。它以直观图象模拟柴油机的工作过程,分柴油机构造、4 个冲程、连续工

作等部分,模拟速度可调。

### 3. 高能实验室 软盘:1片 定价:30元

本软件是高中物理教学课件。它演示“回旋加速器”、“质谱仪”、“威尔逊云室”的工作原理及其过程,形象生动。它操作简单,有汉字提示。

### 4. 大学英语快速阅读 软盘:2片 定价:30元

本软件可用于测试学生的英语阅读能力,它包括《大学英语快速阅读》4 册书的全部文章。学习者可选择“限时”和“自控时”两种方式阅读课文。读完每课后要回答 10 道与课文有关的选择題,然后计算机给出对本课的阅读理解率和阅读速度,据此即可了解学习者的阅读能力和程度,以便改进学习方式。

### 5. 长城加密系统 软盘:1片 定价:75元

它是用于整片软盘加密的工具软件,能有效地对所有在 DOS3.3 操作系统下开发的软件进行加密;加密性能可靠,加密后的软件用现在流行的解密拷贝均无法复制;加密过程简单,中途无需换盘,加密一次完成。

### 6. 应用软件集 ASOCS 软盘:1片 定价:60元

本软件集包括文本编辑、魔窗、图形处理、电子报表、记录文件、数学教师和电子琴七个部分。它内容丰富、实用性强。

文本编辑用于中英文文字处理;魔窗是英文字处理软件,它具有单词输入正确性的自检功能;图形处理可方便地使用图形移动、放大、倒置、对称和改色等;电子报表是一种通用财务软件;记录文件可作为一般性文件记录应用;数学教师是辅助数学教学(中学数学)软件;电子琴是一个多音色可变速的键盘演奏软件。

## 封底说明

武汉创意电子研究所系武汉东湖高技术小区的一家高技术企业。全所科技人员 21 名,其中具有硕士学位以上的有 9 名、其余皆具有大专以上学历,本所技术员吴微、马国敏、孔曙光、罗维国继 1990 年在《电子与电脑》杂志(笔名吴中国)里公布了 SCB-1 MCS-51 单片单板机以后,又于 1991 年在北航出版社出版的《单片单板机应用文选》中以 15 万字的篇幅公布了 SCB-2、MCS-51、8098 单片单板机,该机是一种性能优越、结构合理、成本低廉的两用性单板机。该机操作方便,功能完善,既便于学习,又适用应用开发,创意 CYSCB-2 MCS-51、8098 则是上面机型的改进型。下面是本所产品的报价单:

### 1. CYSCB-2 MCS-51、8098 单片单板机

(含主机板、键盘板、CYSCB-51 系统盘、CY8098 系统盘、通讯电缆) 980.00 元。

### 2. CYSCB-1 MCS-51 单片单板机

(含主板机、键盘板 CYSCB-51 系统盘、通信电

缆) 500.00 元

3. CYD-1 单片机 TV/CRT 显示接口板 400.00 元

4. CYD-1 全系列智能 EPROM 编程器 300.00 元

5. CYPRN-1 256×256×8 BITS 图像板 3500.00 元

6. CYEM-10 电镜图像分析系统(IBM386 系统一套,其中内存 2M、40M 硬盘,1.2M+360K 软驱、鼠标、TVGA 卡及显示器、扫描电路图像实时采集与输出接口、图像分析与处理软件) 2500.00 元

### 7. CYP-1 电脑打像机(两种配置)

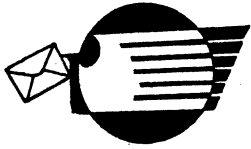
①(IBMPC 主机、CCD 摄影头、14 英寸黑白监视器、24 针打印机) 14800.00 元

②CCD 摄像头、14 英寸黑白监视器、图像处理板、打像软件与 控制机盒、打印机配套 8800.00 元

8. CYT-1 1 加电电脑电话分路器 2000.00 元

9. 固体(全数字化)录音机 500.00 元

(转第 33 页)



## 读者联谊

# 普及型 PC 个人用户软件交流联谊活动 问题解答(八)

北京中国农科院计算中心(100081) 王路敬

### 24. 磁盘的存储一般格式是怎样的?

一个磁盘,不是任何位置都可以储存数据信息的。磁盘在使用之前必须经过“格式化”处理,使之符合 DOS 的规定。磁盘格式化时把盘片划分成一个个同心圆的磁道,数据是存储在磁道里的。每一个磁道又划分成一个个扇区,扇区是磁盘的基本存储单位,每当读写磁盘数据时总是读写一个个完整的扇区。例如软盘,在 DOS 2.0 或 2.10 版本下,把软盘(双密度)划分为 40 个磁道,每磁道 9 个扇区,每个扇区可存储 512 字节的内容。磁盘的两个面都可以使用。磁盘的面、磁道、扇区都是固定编号的。面的编号是 0 或 1,磁道的编号是从 0 至 39。0 磁道在磁盘的最外圈,39 磁道在磁盘的最内圈。扇区的编号是 1 至 9,整个盘片有 720 个扇区。

DOS 管理磁盘时,把磁盘上的所有扇区有规则地按一定的次序连续排列,并且尽可能地按照这样的次序使用。排列后的每一扇区也有相应的编号,称为逻辑扇区号或称相对扇区号。格式化命令 FORMAT 对磁盘进行磁道和扇区划分的同时,还把扇区的每个字节的内容置为一个特定的值——F6H,用来表示扇区是空的。

DOS 把磁盘的扇区根据用途的不同划分为四个区域,它们是:

#### (1) 引导记录程序区

该区只占一个扇区,即 0 面 0 道 1 扇区,相对扇区号为 0。每一个 DOS 磁盘都有一个 DOS 引导记录扇区,它是磁盘格式化操作时所建立的。这一 DOS 引导记录是磁盘启动操作系统所必须的,它在 DOS 启动时即自动装入到内存特定的区域开始运行,它的主要作用是负责将 DOS 盘上的两个系统隐含文件 IBMBIO.COM、IBMDOS.COM 装入内存,如果引导失败或者非 DOS 系统盘,则给出错误信息“Disk boot failure”或“Nonsystem disk or disk error”。

DOS 引导记录扇区中除了包括引导程序外,还包括了两张磁盘参数表。第一张表称为磁盘输入/输出参数表,它位于引导记录扇区相对字节 0BH 至 1CH。另一张表称为磁盘基数表,位于相对字节 21H 至 2BH。表中的各项参数与引导扇区中的其他数据一道描绘了整个磁盘的各种状态的使用情况,磁盘的输入/输出参数表是供 DOS 管理文件使用的,磁盘基数表是供驱动程序直接控制硬件使用的。

#### (2) 磁盘文件分配表区

这是一个相当重要的区域,它用来记录每一个文件在磁盘空间的分配情况,磁盘空间分配以簇为单位,也就是说分配给文件的存储空间都是一个簇,而

不管该文件是否占满整个簇,一连串的“簇链,便指出了文件在磁盘数据存储区域中所占用空间对应的各个簇号。簇号与数据存储区域的扇区是依次对应的。在实际使用中,文件占用的磁盘扇区位置不一定是连续的,但通过“簇链”就能够搜索到文件的每一个连续的或不连续的扇区位置。一个文件的“簇链”从该文件目录登录项的“起始簇号”开始,它记载着分配给文件的第一个簇号,第一个簇号的 FAT 位置上的内容包含着第二个簇号,第二个簇号位置上的内容又包含第三个簇号,依次类推,直至分配到最后一个簇。

磁盘文件分配表简称 FAT 表,它包含以下四个方面的信息内容:

- (1) 磁盘的类型
- (2) 扇区使用情况
- (3) 指出一个文件的后继簇
- (4) 扇区封锁信息

在 FAT 表中,用 1.5 个字节即三位十六进制数来记录一个簇号,这 1.5 个字节称为一个登录项。也有一些磁盘是用二个字节来记录一个簇号的。FAT 表从第二个登录项即第 3 个字节开始记录簇号,所以有效簇号是从 002 开始的。

FAT 表的前三个字节有特殊用途。第 1—2 字节固定为 FFFFH,第 0 字节用来标识磁盘类型,规定如下:

- FFH——表示双面每道 8 扇区软盘;
- FEH——表示单面每道 8 扇区软盘;
- FDH——表示双面每道 9 扇区软盘;
- FCH——表示单面每道 9 扇区软盘;
- F8H——表示硬盘;

FAT 表每个登录项的内容包含三位十六进制数值,含义如下:

- 000H——表示一个未被使用但可以使用的簇;
- FFDH-FF6——表示保留的簇;
- FF7H——表示相应的簇为坏簇,不能使用;
- FF8-FFFH——表示该簇为文件中最后一个簇,即表示一个文件的结束。

其他在登录项中出现的任何十六进制数值都是指针,指向文件位置分配链中下一簇的簇号。而文件的第一个簇号是由文件目录的第 26—27 字节来指出。坏簇一般是在磁盘格式化时发现缺陷被检测出来而将其打上坏簇标记不再使用。但有一点例外,计算机病毒程序常常自动地设置一些坏簇来保护自己,这种坏簇是假坏,而不是物理上的损坏。

为了确保磁盘文件建立、修改、读写或删除操作的可靠性,在实际使用中,DOS 设置了两份内容完全

一致的 FAT 表,这样即使由于盘片磨损或软件故障引起一个 FAT 表读出错误,还可以通过访问另一个 FAT 表来完成相应的操作。如果两个 FAT 表中都出现一个非法的簇域,则在使用 CHKDSK 命令时系统将显示如下信息:Allocation error for file size adjusted

该信息指出在指定的文件簇链中发现一个非法簇域,将在文件有效末尾截断该文件。如果两个 FAT 表出现更为严重的故障,则 DOS 会提出警告:

File allocation table bad drive x

表示该磁盘根本不能使用,需要重新建立。

不管是建立一个新文件,还是对一个文件进行扩展,DOS 对一个文件分配簇号时,总是扫描 FAT 表,以便找到可用簇,而不论簇号是否连续。当一张盘多次使用文件修改、删除等操作时更是如此。

### (3) 磁盘文件目录表区

磁盘文件目录表具体描述文件名、子目录名、卷标及相应的信息。DOS 就是利用文件目录表掌握磁盘上每个文件所在的路径、文件属性、分配的位置、文件的长度以及建立或修改的日期和时间。当应用程序对某一磁盘文件提出读写请求时,DOS 首先在文件目录表上查找是否存在该文件。如果存在,则根据文件分配的位置并利用文件分配表及磁盘参数表,将对一个文件的请求转换成对某一相应逻辑扇区的请求。后一请求则由磁盘设备驱动程序变成实际的物理地址,最后,依赖于硬件 I/O 驱动程序来完成对磁盘的信息存取。

### (4) 数据区

磁盘的数据区又称用户区。值得注意的是不同 DOS 版本及不同磁盘类型,其磁盘空间划分亦不相同,如下面表中有关数据即说明这个问题。

| 磁盘类型    | FAT 起<br>始扇区 | 每个 FAT<br>长度 | 目录扇<br>区数 | 目录<br>项数 | 扇区<br>数/簇 | 数据区起<br>始扇区 |
|---------|--------------|--------------|-----------|----------|-----------|-------------|
| 单面 8 扇区 | 1,2          | 1            | 4         | 64       | 1         | 7           |
| 双面 8 扇区 | 1,2          | 1            | 7         | 112      | 1         | 0A          |
| 单面 9 扇区 | 1,3          | 2            | 4         | 64       | 1         | 9           |
| 双面 9 扇区 | 1,3          | 2            | 7         | 112      | 2         | 0C          |
| 10M 硬盘  | 1,9          | 8            | 20        | 512      | 8         | 31          |

### 25. 硬盘空间的划分有何特点?

硬盘的组织方式与软盘有其相似之处,软盘有 40 个磁道,编号为 0 到 39;10MB 硬盘相应应有 306 个柱面(与磁道等效),编号为 0—305。柱面 305 通常保留为诊断用,所以只有柱面 304 是实际可用的。10MB 硬盘有 2 个盘片,因而有 4 个面,从 0 到 3,每个盘面在每个圆柱面中为一个磁道,每个磁道有 17 个扇区,对应于软盘上的每个磁道有 8 个或 9 个扇区。扇区大小保持一致,都是 512 个字节。

一个硬盘系统可分为 4 个“分区”。划分这些分区的目的是支持 DOS 以外的操作系统。最多可以有四种磁盘操作系统,每一种操作系统在硬盘上建立自己专

用的分区操作时都由一个相应的实用程序来完成。例如 DOS 的分区实用程序名为 FDISK.COM,CP-M/86 的分区实用程序名为 HDMAINT。分区的具体数目(1—4)和 DOS 分区的大小(多少个柱面)可由 DOS 命令 FDISK 设定。需要注意的是,分区的修改将会擦掉磁盘上的数据,所以如果要改变分区而又要保留数据,唯一的办法是把数据先作备份,以后再重新装入。

硬盘空间的划分与软盘所不同的是每一硬盘上都有主引导记录和分区引导程序。硬盘的第 1 个扇区是主引导记录,它含有本硬盘的一个分区表。操作系统的启动工作由分区引导记录来做,而主引导记录程序的任务之一就是找到哪一个磁盘分区被标记为“活动分区”,然后把控制权交给该分区所对应的引导记录。至于哪个分区是实际可用的,即活动分区,可以通过调用 FDISK 命令的“改变活动分区”功能加以改变。

对于一个 DOS 分区,其文件分配表的大小、根目录大小以及每个簇的扇区数都与 DOS 的版本和分区大小有关,当 DOS 分区占据整个磁盘时,其文件分配表的大小、根目录大小与硬盘容量有密切关系。具体数据可从各类磁盘的基本输入/输出参数表查出。DOS 分区内的空间划分与软盘完全相同,也被划分为 DOS 引导区、文件分配表区、文件目录区和数据区。

### 26. 用户从硬盘分区信息表中可得到哪些信息?

为了实现多个微机操作系统共享硬盘资源,整个硬盘在逻辑上划分为 1 至 4 个分区。IBM-PC 及其兼容机用户可以通过 PC-DOS 随机提供的实用程序 FDISK 来指定各分区号的大小和起止的柱面号,磁道号和扇区号。分区信息表就是用来保持这些分区信息的。

硬盘分区信息表由 4 项组成,每区占一项。每一项由 16 个字节组成。整个分区信息表占 64 个字节。第一分区信息具体内容如下表所示:

| 字节    | 内容     |
|-------|--------|
| 0     | 引导标志   |
| 1—3   | 分区起始地址 |
| 4     | 系统标志   |
| 5—7   | 分区终止地址 |
| 8—11  | 起始扇区号  |
| 12—15 | 实用扇区数  |

第 0 字节:为引导标志。如果该字节值为 00H 表示系统加电或复位自举时不从这个分区引导操作系统。如果该字节值为 80H,则表示系统加电冷启动或热启动复位时从这个分区中引导操作系统。在系统冷启动或热启动引导过程中会检测分区信息表所有 4 项中的这个引导标志。PC-DOS 的 FDISK 分区命令可以置 PC-DOS 引导分区中这个字节值为 80H,而其余分区的这个字节为 00H。若指定从某一分区上引导操作系统,这个分区的引导程序必须在该分区的第一扇

区。

第 1 字节:为本分区起始磁头号

第 2 字节:为本分区起始扇区号。在这个字节中只有低 6 位是表示扇区号,而高 2 位则表示柱面号的最高两位。

第 3 字节:为本分区起始柱面号。

第 4 字节:为本分区操作系统代码。PC-DOS 的名称代码为 01H。

第 5 字节:为本分区结束磁头号。

第 6 字节:为本分区结束扇区号。

第 7 字节:为本分区结束柱面号。

第 8—11 字节:在本分区之前已经使用掉的扇区数。

第 12—15 字节:为本分区所使用的扇区数。

## 27. 如何正确的读出硬盘主引导程序和 PC-DOS 分区的引导程序?

了解这个问题首先要明确硬盘主引导程序和 PC-DOS 分区引导程序的区别和联系。硬盘的主引导程序在硬盘的第一扇区中,不属于任何一个分区。也就是说,它不属于任何一个操作系统,它是各操作系统的共同部分。它的作用就是查看分区信息表中的 4 个分区引导标志。当某一分区的引导标志为 80H 时,主引导程序就把这一分区的第 1 扇区读入内存 0000:7C00H 处,并从那儿开始执行,主引导程序是隐含扇区,不能用 DEBUG 程序读写它,只能用中断 13H 来读到它,可以用下面的汇编程序:

```
MOV DX,0080
MOV CX,0001
MOV AX,0201
MOV ES,SEG BUF
MOV BX,OFFSET BUF
```

INT 13

而 PC-DOS 分区引导程序则占据着分区的第 1 扇区,当系统是从硬盘启动引导时,硬盘主引导程序在执行过程中把它读入内存 0000:7C00H 处,并从该地址开始执行这个分区引导程序,此时该分区引导程序做如下三方面的工作:

(1)为 PC-DOS 分区引导程序的执行以及以后操作系统的运行提供一张表格,详细驱动器的参数。这些参数包括:每扇区的字节数,每簇的扇区数,保留扇区数,FAT 表的数目,总扇区数,介质说明字节,每个 FAT 表的扇区数,每道扇区数,磁头号,隐藏扇区数。

(2)根据上面这张表格提供的信息,把文件根目录区的第 1 扇区读入内存,查系统的两个主要隐含文件 IBMBIO.COM 和 IBMDOS.COM 是否存在。如果存在,则把 IBMBIO.COM 读进内存 0050:0000H 处并从此开始执行 IBMBIO.COM。

(3)提供一张有关软盘驱动器的电参数表。PC-DOS 分区引导程序不是隐含扇区,用 DEBUG 可以读写。

## 28. 软盘和硬盘 PC-DOS 分区引导程序有否差别?

软盘 PC-DOS 引导程序从组成主要包括磁盘驱动器的结构参数表、软盘驱动器的电参数表和引导程序三部分。对于各种各样的硬盘,其 PC-DOS 分区引导程序完全是一样的,差别仅在于磁盘结构参数有些项的内容不同。例如 IBM-PC/XT10MB 硬盘每簇扇区数为 8 个,而 IBM-PC/AT 则为 4 个;总扇区数 IBM-PC/XT 10MB 硬盘为 5103H,而 IBM-PC/AT 为 A307H 等等。

不管软盘还是硬盘 PC-DOS 分区引导程序,这一扇区的逻辑扇区号总是 0,该扇区的内容可以通过 DEBUG 程序装入命令 L 读入内存,然后通过显示内存单元内容的命令 D 在屏幕上显示出来,读入的时候指定起始扇区号为 0,装入的扇区数为 1,驱动号规定 A 为 0,B 为 1,C 为 2。例如读 C 盘上 PC-DOS 引导扇区内容,其操作:

```
A>DEBUG
-L 100 201
-D 100 2FF
```

须指出,PC-DOS 分区引导程序不一定就在硬盘的第 2 扇区上,因为 PC-DOS 分区不一定就在硬盘的第 1 个分区,另外即使 PC-DOS 分区是硬盘上的第 1 个分区,在硬盘的主引导程序和 PC-DOS 分区引导程序之间还可能有其分隐含扇区。例如 IBM-PC/XT 和 PC/AT 的隐含扇区数包括主引导程序分别为 1 和 17。

## 29. 在什么情况下硬盘需要进行初始化? 有哪些方法?

硬盘初始化就是对硬盘进行物理格式化。一般在下列情况之一出现时,均需对硬盘进行初始化操作。

- (1)用户更换新的硬盘驱动器;
- (2)移动机器前没有进行磁头复位的操作,而造成磁盘面损坏;
- (3)硬盘磁头的读写时断电,使磁头划伤磁盘面;
- (4)在硬盘上试图装入几种不同的操作系统。例如,若原来盘上装有多用户操作系统 XENIX 和单用户操作系统 DOS,这时,如果想装入另外一种多用户操作系统就需要对硬盘进行初始化。

对硬盘初始化可以通过以下三种方法进行:

(1)借助 ROM BIOS 程序

在控制板上的硬盘 ROM BIOS 程序中提供了硬盘初始化程序。如长城 0520 系列微机就可以在 DEBUG 下通过 G=C800:0005 命令来执行这个初始化程序。

请注意,初始化程序入口地址随控制器板的不同可能有所不同。

(2)运行 LOWFORM.EXE 程序

长城系列机或 IBM-PC 系列机及其兼容机有不少机型提供了初始化程序 LOWFORM.EXE。在操作系

(转第 25 页)

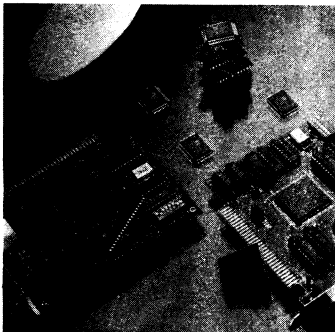




電子

ISSN 1000-1017

與電腦



ELECTRONICS AND COMPUTERS

一九九二年

总期第90期

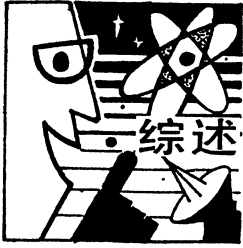
# 電子與電腦

## 目 录

- 综述 •
  - 计算机安全性的几种实现方法 ..... 邱向群(2)
- PC 用户 •
  - 文本方式下汉字菜单的实现 ..... 曾跃忠 蔡以群(3)
  - 巧解黑色星期五病毒 ..... 谭小敏(5)
  - 通用的汉化 WordStar ..... 曹国钧(6)
  - 反函数查找法的实现 ..... 王杰民(7)
  - 在 CCDOS2.13F 支持下 dBASE III 画图技术  
..... 徐国茂 黄松德 刘少明(8)
- 学习机之友 •
  - 英语单词快速记忆系统 ..... 张振堂(13)
  - 用 CHR\$ 函数压缩存盘数据 ..... 梁才柱(16)
  - PB-700 微机解密技巧 ..... 孙力(17)
  - 双人百米赛跑 ..... 汪波(17)
  - 寻找莱蒙托夫游戏规则 ..... 钱雁群(17)
  - 也谈“一题多解” ..... 郑明达(18)
  - 考试的统计分析程序 ..... 胡筱罡(19)
  - 氯化氢制取的动态显示 ..... 刘萍(20)
  - 磁盘加密一法 ..... 黄晓晖(21)
- 6502 机器语言讲座 •
  - 第九章 堆栈程序设计 ..... 朱国江(22)
- 初级程序员级软件水平考试辅导 •
  - 试题解答与分析 ..... (25)
- 学用单片机 •
  - 单片机实验与 BJS-51 实验教程 ..... 李广弟(30)
- 学装微电脑 •
  - 自动输送装置 ..... 易齐干(32)
- 电脑巧开发 •
  - 计算机语音输出功能的开发与应用(上) .....  
..... 陈竹林(36)
  - 怎样使 CCDOS2.10 的九针打印驱动程序适应  
2.13H 汉字系统 ..... 李修连(38)
  - 8031 真的无 ROM 吗? ..... 肖革文(38)
- 维修经验谈 •
  - IBM-PC/XT 及其兼容机 RAM 故障的检修方法  
..... 齐吉泰(39)
- 电脑游戏机 •
  - 第二讲 故事情节和游戏结构 ..... 于春(41)
- 新书与软件 •
  - 软件介绍 ..... (44)
- 读者联谊 •
  - 普及型 PC 个人用户软件交流联谊活动问题解答(九)  
..... 王路敬(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



# 计算机安全性的几种实现方法

西安交通大学(710049) 邱向群

由于近年来计算机病毒的泛滥,各式各样的数据窃取以及计算机犯罪,人们对计算机的安全性越来越重视。

目前普遍使用的保护方法主要是对合法用户的身份进行验证:例如阅读磁卡,指纹检验和密码口令校验等。

根据美国计算机安全委员会近来的一项调查,发现所有计算机损失事件中仅有3%是由外部非法用户造成的,65%的损失是由于雇员的差错而造成的,其余32%是那些不忠诚或为了泄私愤的雇员故意造成的。事实上很多人对计算机安全性的认识不足,不少人将软磁盘和写有密码的本子随意乱放,与别人共用磁盘,选择最易于被猜到的口令,例如生日,一星期内的日期或身份证等证件上的号码。一个单位竟有数十人用单位电话号码做密码。

下面给出一种美国普遍使用的RSA加密算法:这一算法是美国麻省理工学院的三位科研人员研制的。为了便于说明问题,下面的例子使用简单的质数,而在实际运用中,密码通常是一个很大很复杂的数,使非法用户几乎无法解密。

## 一、生成加密码:

1. 随机地取一个奇数:  $E=5$ ;
2. 取二个质数  $P$  和  $Q$ ,  $P=7, Q=17$ ; 使  $(P-1)(Q-1)-1$  可以被  $E$  整除;
3.  $P$  乘  $Q$  得  $N$ ;  $P \times Q = N = 7 \times 17 = 119$ ;
4. 将  $N$  和  $E$  连接起来即为加密码  $NE$ ;  $NE = 1195$ 。

## 二、生成解密码:

1.  $P, Q$  和  $E$  分别减 1 后相乘,再加 1:  
 $(P-1)(Q-1)(E-1)+1=6 \times 16 \times 4+1=385$ ;
2. 将结果除以  $E$ , 得到  $D$ ;  $D=385/5=77$ ;
3. 将  $N$  和  $D$  连接起来得到解密码  $ND$ ;  $ND=11977$ 。

## 三、用加密码对数据加密:

1. 先将数据转化成数字,例如用 19 代表  $S$ , 则明码 = 19。
2. 加密算法:
  - 1) 对明码进行  $E$  次幂运算:  $19^5 = 2476099$ ;
  - 2) 结果除以  $N$ :  $2476099/119 = 20807$ , 余数为 66;
  - 3) 余数即为密码: 密码 = 66。

## 四、用解密码对密码解密:

1. 解密算法:
  - 1) 对密码进行  $D$  次幂运算:  $66^{77} = 1.27 \times 10^{140}$ ;
  - 2) 结果除以  $N$ :  $1.27 \times 10^{140}/119 = 1.069 \times 10^{138}$ , 余数

为 19;

2. 余数即为明码: 明码 = 19。

有不少专家认为密码法对计算机并非万无一失,他们中的一些人提出并实现了一些改进方法,例如三次口令输入错误(一般合法用户是不会犯这样的错误的)就退出系统,并发出警报,或是再配合硬件一起使用——目前主要是磁卡,有人称此为动态口令,上面的密码是用户无法阅读或修改的。目前美国约有 70,000 ~ 100,000 计算机用户使用这种方法,用户主要是国防部门、航空航天、通信以及金融等要害机构。一般工业和民用用户对目前颇为昂贵的磁卡还不敢问津。

“生物法”是一种高科技的身份校验方法,它通过校验人体的生理特点或行为特性来防止非法用户的侵入。生理特点如指纹、视网膜上的血管布局和手的几何形状——这些特点因人而异,而对每个人来说则是固定不变的。行为特点如声音、动态签字方式以及敲击键

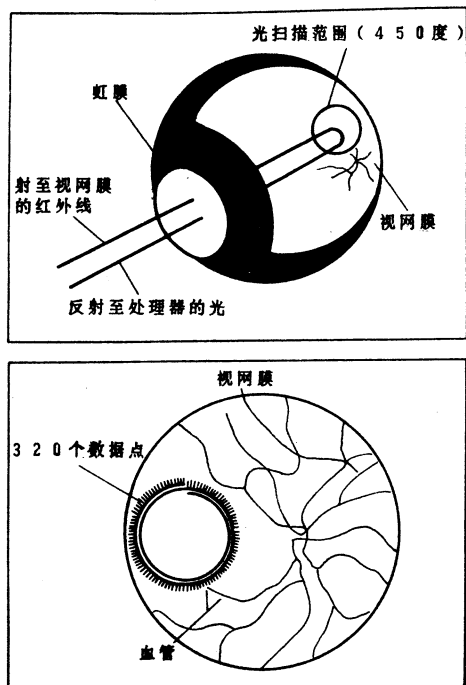
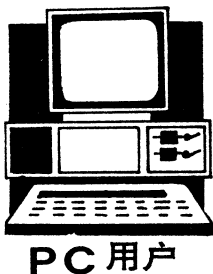


图 1

(转第 12 页)



# 文本方式下汉字菜单的实现

苏州三光电加工有限公司 曹跃忠 蔡以群

弹出式菜单通过屏幕把用户和系统联系起来,显示直观、选项快速方便,因而深得广大用户青睐,已被越来越多的系统所采用。但是,由于弹出式菜单需要把显示区内容保存在内存中,因此弹出式菜单一般都是在文本方式下以西文显示的方式实现的,这对许多不太熟悉西文的用户来说具有明显的缺陷。

在系统开发的过程中,我们对计算机的显示原理作了一些研究,实现了 EGA/VGA 显示器上文本方式下的汉字显示,从而开发成功了汉字显示的弹出式菜单,并在我们的实际系统中取得了令人满意的效果。

## 一、EGA/VGA 文本方式下的显示原理

EGA/VGA 显示卡上可以有不同数目的存储器,这些存储器一般分成四个独立的等容量的存储区段,即彩色页面。对于 CPU 来说,这些彩色页面具有相同的存储地址,CPU 执行读或写操作时,彩色页面的选通与否由相关寄存器的设置值来决定。

在文本模式下,彩色页面 0 和彩色页面 1 存放 ASCII 值及其文本属性,彩色页面 2 存放字符集。在彩色页面 1 中存放的文本属性是指相应的显示于屏幕上的 ASCII 字符的显示属性,它具有 8 个规定位,在标准彩色文本模式下,各位含义如表 1 所示:

|                |                |                |                |                |                |                |                |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| D <sub>7</sub> | D <sub>6</sub> | D <sub>5</sub> | D <sub>4</sub> | D <sub>3</sub> | D <sub>2</sub> | D <sub>1</sub> | D <sub>0</sub> |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|

- D<sub>2</sub>、D<sub>1</sub>、D<sub>0</sub>——背景彩色
- D<sub>3</sub>——前景增强或字符表选择
- D<sub>6</sub>、D<sub>5</sub>、D<sub>4</sub>——底色
- D<sub>7</sub>——底色增强或闪烁

表 1. 文本属性位

当一个 ASCII 字符要显示在屏幕上时,计算机首先从页面 2 的字符表中读取 ASCII 字符的相应的点阵信息,据此再向显示器输出显示信息。所谓字符表,就是指希望在屏幕上显示的符号的点阵信息的集合。

含有 256KRAM 的 EGA/VGA 显示卡的每个页面存储容量为 64K,每个字符表包含 256 个字符,每个字符占 32 个字节,每个字符表长为 8K,因此页面 2 可以包含多个字符表,表 2 是字符表的地址。

| offset |       | offset |       |
|--------|-------|--------|-------|
| E000H  | 未用    | E000H  | 字符表 8 |
| C000H  | 字符表 4 | C000H  | 字符表 4 |
| A000H  | 未用    | A000H  | 字符表 7 |
| 8000H  | 字符表 3 | 8000H  | 字符表 3 |

|       |       |       |       |
|-------|-------|-------|-------|
| 6000H | 未用    | 6000H | 字符表 6 |
| 4000H | 字符表 2 | 4000H | 字符表 2 |
| 2000H | 未用    | 2000H | 字符表 5 |
| 0000H | 字符表 1 | 0000H | 字符表 1 |

EGA

VGA

表 2. RAM 中字符表的驻留地址

由于 RAM 中可以驻留多个字符表,因此可以通过设置定序器中相应中的寄存器来选择 1 个或 2 个活化字符集,字符表选择寄存器各位含义如表 3 所示。

|                |                |                |                |                |                |                |                |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| D <sub>7</sub> | D <sub>6</sub> | D <sub>5</sub> | D <sub>4</sub> | D <sub>3</sub> | D <sub>2</sub> | D <sub>1</sub> | D <sub>0</sub> |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|

- D<sub>1</sub>、D<sub>0</sub>——字符表选择 B
- D<sub>3</sub>、D<sub>2</sub>——字符表选择 A
- D<sub>4</sub>——字符表选择 B(对 VGA)
- D<sub>5</sub>——字符表选择 A(对 VGA)
- D<sub>7</sub>、D<sub>6</sub>——保留

表 3. 字符表选择寄存器

当用户选择了两个字符表后,文本属性中的位 D<sub>3</sub>起选择字符表的作用,D<sub>3</sub>=0 表示选择字符表 B,D<sub>3</sub>=1 则表示选择字符表 A。

## 二、文本方式下汉字菜单的实现

从 EGA/VGA 文本方式下的显示原理可以知道,屏幕显示是由字符表中的点阵信息来决定的,因此当我们在字符表中装入汉字点阵信息时,汉字就会显示在屏幕上。

在一般的实际应用系统中,所需要显示的汉字不会很多,因此我们可以通过设置字符表选择寄存器来活化二个字符集,在系统中就可以活化二个字符表,一个字符表存放 ASCII 字符的点阵信息,而另一个字符表则存放汉字点阵信息,这样可以为系统提供一百多个汉字。如果在系统中变换汉字字符表,就可以为系统提供任意多个汉字。

为了装入汉字信息显示的字符表,必须对有关寄存器值进行修改,汉字字符表装入后,又必须对相关寄存器进行信息恢复,表 4 提供了几个我们用 Turbo C 编写的函数,其中 setvga() 完成装入汉字字符表前的寄存器值修改,resetvga() 在装入汉字字符表后对寄存器值进行恢复,而 sel.char(t<sub>1</sub>,t<sub>2</sub>) 则选择字符表,t<sub>1</sub> 表示选择字符表的个数,t<sub>1</sub>=1 表示选择 1 个字符表,t<sub>1</sub>≠1 则表示选择 2 个字符表,t<sub>2</sub> 则表示选择哪一个字符表。这些函数适用于 EGA 或 VGA。

setvga()

```

{
char seqparm[] = {0x00,0x02,0x04,0x00};
char seqparm1[] = {0x01,0x04,0x07,0x03};
char gcparm[] = {0x04,0x05,0x06};
char gcparm1[] = {0x02,0x00,0x00};
int i;
inportb(0x0da);
outportb(0x03c0,0x10);
outportb(0x03c0,0x00);
for(i=0;i<=3;i++)
{
outportb(0x03c4,seqparm[i]);
outportb(0x03c5,seqparm1[i]);
}
for(i=0;i<=2;i++)
{
outportb(0x03ce,gcparm[i]);
outportb(0x03cf,gcparm1[i]);
}
}
}

```

```

resetvga()
{
char seqparm[] = {0x00,0x02,0x04,0x00};
char seqparm1[] = {0x01,0x03,0x03,0x03};
char gcparm[] = {0x04,0x05,0x06};
char gcparm1[] = {0x00,0x10,0x0e};
int i;
for(i=0;i<=3;i++)
{
outportb(0x03c4,seqparm[i])
outportb(0x03c5,seqparm1[i])
}
for(i=0;i<=2;i++)
{
outportb(0x03ce,gcparm[i])
outportb(0x03cf,gcparm1[i])
}
}
}

```

```

sel __char(int t1,int t2)
{
union REGS regs;
if(t1==1)
{
regs.x.ax=0x1103;
regs.h.bl=t2;
int86(0x10,®s,®s);
}
else
{
regs.x.ax=0x1000;
regs.x.bx=0x0712;

```

```

int86(0x10,®s,®s);
regs.x.ax=0x1103;
regs.h.bl=t2;
int86(0x10,®s,®s);
}
}

```

表 4. 寄存器操作及字符表选择函数

表 5 是装入汉字字符表并选择字符表的流程图，限于篇幅，程序从略。

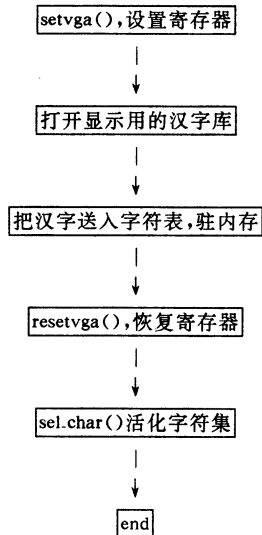


表 5

当我们在系统中装入并选择了汉字符表后，直接对显示区内存进行操作就可以将汉字显字在屏幕上。如果我们在系统中字符表 2 中装入了汉字字符表，字符表 1 中装入了 ASCII 字符集，那么表 6 所示程序将显示所有 ASCII 字符和字符表 2 中的所有汉字。

```

display()
{
int i;
for(i=0;i<256;i++)
poke(VSG,i*2,i+0x1700);/* VSG:video ram segment/
for(i=0;i<256;i++)
poke(VSG,i*2+640,i+0x1f00);
}

```

表 6. 字符表的显示

采用上述方法，我们用 Turbo C 在 AST386 SX/16 上实现了汉字显字的弹出式菜单。

在系统中装入汉字字符表来显示汉字的方法，可以在装 EGA 或 VGA 的 IBM-PC 系列及其兼容机上实现，并且可以把这种思想应用于装有其它类型显示器的计算机。使用这种方法显示汉字不占用系统内存，汉字显字快速方便，并且可以显示多种字体。这种方法简易方便，可以用汇编语言和多种高级语言来实现。

# 巧解黑色星期五病毒

长春市东北水利电力专科用电 891 班(130012) 谭小敏

黑色星期五病毒(Jerusalem)是一种流行较广且危害较大的恶性病毒。本文介绍一个利用 DEBUG 汇编的小程序,该程序可以比较方便地对计算机内存进行黑色星期五病毒的检查 and 清除。该程序本人在长城 0520 机上调试验证,效果不错。具体实现步骤如下:

```
A>debug
-A100
XXXXXXXX 0100 MOV SI,0173
XXXXXXXX 0103 CALL 0127
XXXXXXXX 0106 MOV AX,3521
XXXXXXXX 0109 INT 21
XXXXXXXX 010B CMP BX,025B
XXXXXXXX 010F JZ 011A
XXXXXXXX 0111 MOV SI,01B5
XXXXXXXX 0114 CALL 0127
XXXXXXXX 0117 RET
XXXXXXXX 0118 INT 20
XXXXXXXX 011A MOV AX,3508
XXXXXXXX 011D INT 21
XXXXXXXX 011F CMP BX,021E
XXXXXXXX 0123 JZ 0136
XXXXXXXX 0125 JMP 0111
XXXXXXXX 0127 CS:
XXXXXXXX 0128 LODSB
XXXXXXXX 0129 AND AL,7F
XXXXXXXX 012B JZ 0117
XXXXXXXX 012D MOV AH,0E
XXXXXXXX 012F MOV BX,0007
XXXXXXXX 0132 INT 10
XXXXXXXX 0134 JMP 0127
XXXXXXXX 0136 PUSH DS
XXXXXXXX 0137 ES:
XXXXXXXX 0138 MOV AX,[0015]
XXXXXXXX 013B MOV DS,AX
XXXXXXXX 013D ES:
XXXXXXXX 013E MOV DX,[0013]
XXXXXXXX 0142 MOV AX,2508
XXXXXXXX 0145 INT 21
XXXXXXXX 0147 MOV AX,3521
XXXXXXXX 014A INT 21
XXXXXXXX 014C ES:
XXXXXXXX 014D MOV AX,[0019]
```

```
XXXXXXXX 0150 MOV DS,AX
XXXXXXXX 0152 ES:
XXXXXXXX 0153 MOV DX,[0017]
XXXXXXXX 0157 MOV AX,2521
XXXXXXXX 015A INT 21
XXXXXXXX 015C MOV AL,90
XXXXXXXX 015E ES:
XXXXXXXX 015F MOV [0275],AL
XXXXXXXX 0162 ES:
XXXXXXXX 0163 MOV [0276],AL
XXXXXXXX 0166 ES:
XXXXXXXX 0167 MOV [0277],AL
XXXXXXXX 016A POP DS
XXXXXXXX 016B MOV SI,01D3
XXXXXXXX 016E CALL 0127
XXXXXXXX 0171 INT 20
XXXXXXXX 0173
```

```
-E173 0D 0A 54 48 45 20 50 72 6F 67 72 65 6D
-E180 20 6F 66 20 4B 69 6C 6C 20 4A 65 72 75 73 61 6C
-E190 65 6D 20 4D 61 64 65 20 42 79 20 54 61 6E 20 58
-E1A0 69 61 6F 20 4D 69 6E 21 31 30 2F 31 30 2F 31 39
-E1B0 39 31 0D 0A 00 0D 0A 4E 6F 20 4A 65 72 75 73 61
-E1C0 6C 65 6D 20 56 69 72 75 73 20 46 6F 75 6E 64 21
-E1D0 0D 0A 00 0D 0A 46 6F 75 6E 64 20 4A 65 72 75 73
-E1E0 61 6C 65 6D 20 56 69 72 75 73 21 21 4E 4F 57 20
-E1F0 69 73 20 4B 69 6C 6C 65 64 21 21 47 4F 4F 44 20
-E200 42 59 45 0D 0A 00 4D 73 44 6F 73
-N JD.COM
-R BX
BX XXXXX
:0000
-RCX
CX XXXXX
:010B
-w 100
-writing 10B Bytes
-Q
```

这样便在磁盘上生成了一个名为 JD.COM 的解毒程序,且在程序末尾加上了免疫标志,运行时只要在 DOS 状态下键入 JD 即可运行。

注意编制该程序时应应在无毒环境下进行。

# 通用的汉化 WordStar

四川重庆医药设计院(630042) 曹国钧

目前, IBM PC/XT, AT, 286, 386 及其各种兼容机上广泛使用汉字 WordStar, 它不仅可以在编辑公文、书信, 输入高级语言的源程序, 而且也具有极强的打印控制功能。现在, 在用户手上流行两种汉化版本, 一种是机电部六所研制的汉化 WordStar 显示 10 行汉字, 适用于 IBM PC/XT 机上彩色中分辨率模式; 另一种就是中国计算机技术服务公司研制开发的汉化 WS 显示 25 行汉字, 适用于各种类型 286, 386 型微机。当今, 广大用户拥有 Super-PC 微机, 它却只能显示 21 行汉字, 如果把上述两种 WS 汉化版本运行于这种微机上, 就会出现不断闪烁跳动的现象, 给用户带来诸多不便。因此, 笔者根据长期使用汉化 WordStar 的经验, 开发了一种通用的汉化 WordStar 版本。

由于汉化 WordStar 控制行数的模块在主引导文件 WS.COM 中, 我们只需修改此文件即可。为了使汉化 WordStar 通用化, 可以显示 10 行、16 行、21 行、25 行的汉字, 我们在 WS.COM 中增加一段新的汇编语言程序, 用它来控制不同显示行数。为此, 我们对 WS.COM 作了如下修改:

对 WS.COM 修改第一条指令(注: 原第一条指令用调试程序 debug.com 的 R 命令查看, 并记下, 下面有用), 使之跳转到我们新增加的行数控制程序段, 待此程序段运行完毕后, 转移到原 WS.COM 执行段处, 即原 WS.COM 中 `××××:0100 JMP 3AB1` 代码处, 然后运行 WS.COM 全部可执行代码。这样, 既控制了不同的显示行数, 又不破坏 WS.COM 文件。

下面就是具体修改 WS.COM 的步骤:

```
C> debug WS.COM
-A
××××:0100 JMP 5600 (先用 R 命令查看 CX 中内容)
××××:0103
-A 5600
××××: 5600 push ax
××××:5601 push dx
××××:5602 mov dx,5631
××××:5605 mov ah,09
××××:5607 int 21
××××:5609 mov ah,01
××××:560B int 21
××××:560D cmp al,31
××××:560F jz 5627
```

```
××××:5611 cmp al,32
××××:5613 jz 5620
××××:5615 cmp al,33
××××:5617 jnz 562C
××××:5619 mov byte ptr [0248],19 (25 行)
××××:561E jmp 562C
××××:5620 mov byte ptr [0248],14(21 行)
××××:5625 jmp 562C
××××:5627 mov byte ptr [0248],0F (16 行)
××××:562C pop dx (10 行)
××××:562D pop ax
××××:562E jmp 3AB1
××××:5631 DB ' 0:10 行屏幕显示',0D,0A
××××:5661 DB ' 1:16 行屏幕显示',0D,0A
××××:5682 DB ' 2:21 行屏幕显示',0D,0A
××××:56A3 DB ' 3:25 行屏幕显示',0D,0A
××××:56BC DB ' 本软件由重庆大学系统工程及
应用数学系曹国钧编改',0D,0A
××××:5707 DB ' 请选择屏幕显示行数<0>$'
××××:5722
-R CX
CX 5500
:5730
-W
Writing 5730 bytes
-Q
```

至此, WS.COM 修改完毕。为了使读者更好地使用 WordStar, 笔者编制了一个简单但优美的批处理文件 WS.BAT, 首先应将 WS.COM 换名为 CWS.COM, 其内容如下:

```
C> COPY CON WS.BAT
ECHO OFF
CLS
ECHO+(注: O 与十之间不应留有空格, 此命令是为了使光标换行)
ECHO+
ECHO+
ECHO * * * * * 通用的汉化 WordStar 1.00 Ver 选
择行数菜单 * * * * *
CWS
ECHO ON
^ Z(按下功能键 F6)
```

于是, 完整的通用汉化 WordStar 编改完成, 读者可以根据自己的微机显示行数使用 WordStar。

# 反函数查找法的实现

青海第二机床厂产品开发部(810021) 王杰民

查找在计算机程序设计中占有重要地位,对有序表(以下设定按从小到大的顺序),可以采用查找效率较高的折半查找法。折半查找的基本思想是:经过一次关键字比较,将有序表分割成两部分,尔后只在表的一部分(子表)中继续进行,直到找到或找不到某记录为止,具体过程是这样的:先取表的中间位置的记录关键字与给定值比较,若相等,则查找成功;否则,若给定值比该记录的关键字值大(小),就再在表的后(前)半部分找。在表的后(前)半部分中仍取中间位置的记录进行比较,又可舍去这部分中的一半,等等。依次反复进行,直到找到或区间长度小于零表明找不到为止。

由上述可知,折半查找是对有序表基于关键字比较的方法。显然,此表是有限个记录的有序集合,但在实用上,尤其是在工程设计中,有序表的查找并非都能简单套用折半查找法,以齿轮传动设计中常见的渐开线函数为例:渐开线函数  $y = tg(x) - x$ ,其中自变量  $x$  为传动啮合角。实际问题往往是已知某一渐开线函数值,求传动啮合角的值。这一类反函数计算在工程计算中是常见的,可以用迭代法求解,也可以用数据结构中的查找技术求解。

不难看出这类求反函数问题有以下特点:

1) 只知给定值,而未给出一系列待查记录及其有序表。换句话说,预先并不知道关键字的情况。这样一来,就无法直接采用关键字比较的方法,只能分别假设自变量,根据给定公式逐个求出函数值,再将其逐一与给定函数值比较。可见,需要花费大量时间来建造待查记录的有序表,而且这个表是动态变化的;

2) 给定区间内待查记录的个数是无限的,因此函数的个数也是无限的,其值往往是浮点数。由此,判断查找是否成功就不宜进行浮点数比较,而应给出允许误差,只要函数值与给定值两者绝对值之差小于或等于允许误差,就应视为查找成功。否则,区间长度永远不小于零,查找永远不会成功。

由以上两点可见,折半查找在求解反函数问题时,对动态表结构是不适宜的。为此,可对折半查找法加以改进。

反函数查找法是面向动态表的查找方法,它与折半查找法相似。该法要点是:为简化计算,初始查找区间只设上界,不设下界,自变量的估计采取变增量,每计算完一遍,令增量缩小一半,将每遍算出的函数值与给定值比较。如两者之差超过允许值,则将自变量增加或减少一个增量,再代入公式计算,直到满足精度要求为止。问题在于,折半查找法计算的是数组元素的下标,而反函数查找法计算的是元素值本身。由于受计算机字长的限制,在折半时对超长部分必须舍入。这样一

来,不仅增加了时间开销,而且会导致自变量某些区间复盖不到,以至引起查找失败。因此我们将自变量变换为 2 的若干次幂,这样,折半运算时就不存在余数问题了,这是反函数查找法的关键所在。

为了使读者把形式描述的算法改写成程序,这里我们给出算法基本思想的描述。

```
procedure search(y,x:real);
 {y,x 分别为函数给定值与所求的自变量}
var k,a1,b1,c1:integer;
 a12,yy,d,aa,aa1:real;
begin
 置允许的精度要求 d;
 k:=1; {自变量的指数}
 置自变量初值上界 a1;
 b1:=a1;
 repeat
 估算自变量 a12;
 估算函数值 yy;
 aa1:=abs(y-yy);
 if aa1<=d
 then x:=a12; {查找成功}
 else c1:=b1*a(-k); {增量减半}
 k:=k+1;
 case y=yy: q:=a12;return;
 y>yy: a1:=a1+c1;
 y<yy: a1:=a1-c1;
 end;
 until(查找成功)
end.
```

本算法的平均比较次数是自变量密度的函数,与表长无关。其适用范围为求反函数的场合。只要允许的精度合理,查找肯定会成功。

今举例说明。渐开线函数按下式计算

$$y = tg(x) - x$$

其中  $x$  为传动啮合角,一般小于 30 度,计算时以弧度值代入。

$$\text{已知 } y = 0.014904383$$

求  $x$ ,要求精确到秒。若以该  $x$  为自变量,得出的  $y$  值与给定值之差小于或等于 0.0000001,则视为查找成功。

解:

$$30 \text{ 度} = 108000 \text{ 秒}$$

$$\text{初值上界可取 } 2 \text{ 的 } 17 \text{ 次幂} = 131072 \text{ 秒}$$

$$\text{经 } 12 \text{ 次查找得 } x = 20 \text{ 度}$$

答:传动啮合角为 20 度。

下面给出一个由渐开线函数求 30 度角以内传动  
(转第 12 页)

# 在 CCDOS2.13F 支持下 dBASE III 画图技术

云浮硫铁矿企业集团公司信息中心 徐国茂 黄松德 刘少明

计算机辅助企业管理,较多的是为企业提供数据的存储、查询、统计和报表输出。为使问题更清楚、直观,要求反映问题的数据不但要以报表的形式输出,而且更需要以图形的方式来表示,即要求图文并茂。但现在普遍使用的 dBASE III 却没有提供绘图功能,笔者为满足公司所属的质检、销售、安全等用户单位的要求,通过不断摸索,找到一种在 dBASE III 状态下调用数据库数据自动在屏幕上绘制并通过硬拷贝输出企业管理中经常使用的折线图、直方图、扇形图的方法,在实际应用中效果良好。现将此方法编制成较通用的程序提供给读者参考。

## 1. 本文程序的运行环境

硬件:IBM-PC/XT 及其兼容机。软件:CCDOS2.13F 系统,dBASE III-PLUS,CEPSG.COM 硬拷贝程序(640\*200 分辨率用 SGP.COM)。

众所周知,各种图形都是由一些不同的线组成,而线又是由最基本的点组成。所以,画点是解决问题的关键。在 CCDOS2.13F 汉字系统中,提供了一组功能特殊的显示命令(在此按 dBASE III 格式列举与问题相关的部分命令):

(1)画点: ? CHR(14)+“D 色号,点 X,Y”

(2)画横线: ? CHR(14)+“H 色号,左端点 X,Y,长度”

(3)画竖线: ? CHR(14)+“S 色号,上端点 X,Y,长度”

(4)画矩形: ? CHR(14)+“B 色号,左上角点 X,Y,宽,高”或 ? CHR(14)+“E 色号,左下角点 X,Y,宽,高”

以上这些命令的参数都要求是字符型参数。

根据此组显示命令,且以用户单位提供的数据,作为相应的 X,Y 坐标画出点,组成线,构成图形。为了使图形更美观,我们在程序中使用 ? CHR(14)+“10,160”和 ? CHR(14)+“10,161”,CHR(14)+“119,0”和 ? CHR(14)+“119,1”来取消、建立光标和取消、建立提示行,使屏幕图形整洁、稳定。

## 2. 画折线、直方、扇形图的程序说明

程序用 dBASE III-PLUS 编写,如果手头上只有 dBASE III,则略加修改就可以使用(去掉 INKEY() 函数,用过程代替语句 DO(文件名)WITH(参数),并去掉该语句。)

程序使用的数据库是:GYK.DBF,内容如下:

| 编号<br>(BH) | 含硫量<br>(S) | 编号<br>(BH) | 含硫量<br>(S) | 编号<br>(BH) | 含硫量<br>(S) |
|------------|------------|------------|------------|------------|------------|
| 01         | 49.65      | 05         | 49.56      | 09         | 49.69      |
| 02         | 49.16      | 06         | 49.24      | 10         | 49.53      |
| 03         | 49.48      | 07         | 49.06      | 11         | 49.30      |
| 04         | 49.94      | 08         | 49.67      | 12         | 49.80      |

## (1)画折线图

①变量说明: X0,Y0 表示坐标轴原点;H:纵坐标高;L:横坐标长;M:折线的折点数;N:两折点间横坐标的距离;X1,Y1,X2,Y2 分别表示折线中某一段斜线的起始坐标和终止坐标;F:用来确定起点的位置。

## ②程序思想:

A:首先根据 GYK.DBF 求出第一段斜线的起点(X1,Y1)和终点(X2,Y2)

B:以 X 为自变量,步长为 0.5,根据直线方程  $Y = (Y_2 - Y_1) / (X_2 - X_1) * (X - X_1) + Y_1$  作(X1,Y1), (X2,Y2)两点间的斜线。

C:画完后再调用 ZD.PRg 在斜线的终点作一较大的点。

D:求下一段斜线两端点坐标;X1=X2;Y1=Y2; X2=X1+N;Y2=F-(S-49)\*180(Y2 的算式可根据实际需要自定)。

E:画完退出,否则转 B。

## ③程序清单:

```
* PROGRAM ZXT.PRg
SET TALK OFF
? CHR(14)+“1,160”
? CHR(14)+“119,0”
CLEA
USE GYK
MAX=S
MIN=S
DO WHILE NOT. EOF()
IF S>MAX
MAX=S
ENDIF
IF S>MIN
MIN=S
ENDIF
SKIP
```



```

ENDD
M=RECCOUNT()
GO TOP
N=640/M /* N 表示折线段数 */
COL0="2"
X0=10
Y0=315 /* X0,Y0 表示坐标原点 */
H=315
L=600
X1=X0
Y1=(MAX-S)*230/(MAX-MIN)+50
S1=S
C=2
E=0
@ 0,25 SAY "产品检验含硫量曲线图"
? CHR(14)+"S"+"5"+","+STR(X0)+","+STR
(Y0-H)+","+STR(H)+"}"
? CHR(14)+"H"+"5"+","+STR(X0)+","+STR
(Y0)+","+STR(L-5)+"}"
@ 0,1 SAY "品"
@ 1,1 SAY "位"
@ 2,0 SAY "(%)"
@ 22,(L+X0)/8 SAY "编号"
@ 0,0 SAY " "
I=1
X=X1
DO WHILE I<=M
? CHR(14)+"S5,"+STR(X)+","+STR(Y0-5)+
",5}"
@ 23,X/8 SAY BH
@ 0,0 SAY " "
I=I+1
X=X+N
SKIP
ENDD
GO TOP
SKIP
DO ZD WITH C,X1,Y1
DO WHILE E=0
X=X1
IF S>=S1
? CHR(14)+"[*"+STR(C)+"_"+STR(X1+5)+
"+|"+STR(Y1+5)+"@u"+STR(S1,5,2)+"}"
ELSE
? CHR(14)+"[*"+STR(C)+"_"+STR(X1+5)+
"+|"+STR(Y1-5)+"@u"+STR(S1,5,2)+"}"
ENDI
DO WHILE .NOT.EOF() .AND. E=0
X2=X1+N
Y2=(MAX-S)*230/(MAX-MIN)+50
DO WHILE X<=X2 .AND. E=0
@ 0,0 SAY " "
Y=(Y2-Y1)/(X2-X1)*(X-X1)+Y1
? CHR(14)+"D"+STR(C)+","+STR(X)+

```

```

","+STR(Y)+"}"
X=X+0.5
E=INKEY()
ENDD
DO ZD WITH C,X,Y
X1=X2
Y1=Y2
S1=S
SKIP
IF S>=S1
? CHR(14)+"[*"+STR(C)+"_"+STR(X1-
16)+"|"+STR(Y1+5)+"@u"+STR(S1,5,
2)+"}"
ELSE
? CHR(14)+"[*"+STR(C)+"_"+STR(X1-
16)+"|"+STR(Y1-20)+"@u"+STR(S1,5,
2)+"}"
ENDI
C=C+1
IF C=8
C=2
ENDI
ENDD
GO TOP
X1=X0
Y1=(MAX-S)*230/(MAX-MIN)+50
S1=S
SKIP
ENDD
USE
? CHR(14)+"I19,1}"
? CHR(14)+"I0,160}"
RETURN
* PROGRAM ZD. PRG
PARA C,X,Y
? CHR(14)+"D"+STR(C)+","+STR(X-1)+","
+STR(Y)+"}"
? CHR(14)+"D"+STR(C)+","+STR(X+1)+","
+STR(Y)+"}"
? CHR(14)+"D"+STR(C)+","+STR(X)+","+
STR(Y-1)+"}"
? CHR(14)+"D"+STR(C)+","+STR(X)+","+
STR(Y+1)+"}"
? CHR(14)+"D"+STR(C)+","+STR(X+1)+","
+STR(Y-1)+"}"
? CHR(14)+"D"+STR(C)+","+STR(X-1)+","
+STR(Y+1)+"}"
? CHR(14)+"D"+STR(C)+","+STR(X+1)+","
+STR(Y+1)+"}"
RETURN

```

(2)画直方图:

①变量说明:

X0,Y0:用来确定坐标原点,Y0还用来表示方块

的左下角纵坐标;H,L:分别表示纵横坐标长度;

SUM:表示方块图的方块数;K:表示方块宽度;G:表示方块高度(其算式可考虑字段S的值及显示器的分辨率自行确定,以免超出屏幕)。

### ②程序思想

A:确定第一个方块的左下角横坐标X及方块高度G。以(X1,Y0)为方块的左下角坐标,K为宽,G为高作第一个方块。

B:求下一个方块的左下角横坐标X,方块高G: $X=X+K$ ;以(X,Y0)为此方块的左下角坐标。

K为宽,G为高作方块。

C:画完退出,否则转B。

### ③程序清单:

```
* PROGRAM ZZT. PRG
SET TALK OFF
CLEA
? CHR(14)+"I19,0]"
? CHR(14)+"I0,160]"
USE GYK
MAX=S
MIN=S
DO WHILE .NOT. EOF()
IF MAX<S
MAX=S
ENDI
IF MIN>S
MIN=S
ENDI
SKIP
ENDO
COUNT TO SUM
D=0
C=5
X0=10
Y0=319
H= 309
L= 630
@ 0,20 SAY "产品检验含硫量直方图"
? CHR(14)+"S"+STR(C)+","+STR(X0)+","+
STR(Y0-H)+","+STR(H)+"]"
? CHR(14)+"H"+STR(C)+","+STR(X0)+","+
STR(Y0)+","+STR(L)+"]"
@ 0,2 SAY "(S%)"
@ 23,(L+X0)/8-4 SAY "编号"
DO WHILE D=0
@ 0,0 SAY " "
D=INKEY()
K=570/SUM
X=30
DO WHILE .NOT. EOF() .AND. D=0
@ 0,0 SAY " "
G=20+(S-MIN)*270/(MAX-MIN)
? CHR(14)+"E"+STR(C)+","+STR(X,10,2)+","
```

```
+STR(Y0)+","+STR(K,10,2)+","+STR(G,10,2)+
"]]"
```

```
? CHR(14)+"["+STR(X+5,10,2)+"|"+STR(Y0-
G-20)+"@u"+LTRIM(STR(S,6,2))+"]]"
```

```
@ 23,(X+K/2)/8 SAY BH
```

```
@ 0,0 SAY " "
```

```
L=X
```

```
DO WHILE L>=X .AND. L<X+K .AND. D=0 .
AND. X+K-L>=15
```

```
? CHR(14)+"S"+STR(C)+","+STR(L,10,2)+","+
STR(Y0-G,10,2)+","+STR(G,10,2)+"]"
```

```
@ 0,0 SAY " "
```

```
L=L+15
```

```
D=INKEY()
```

```
ENDD
```

```
C=C+1
```

```
IF C=8
```

```
C=2
```

```
ENDI
```

```
SKIP
```

```
X=X+K
```

```
ENDD
```

```
GO TOP
```

```
ENDD
```

```
USE
```

```
? CHR(14)+"I0,160]"
```

```
? CHR(14)+"I19,1]"
```

```
RETURN
```

### (3)画扇形图:

画扇形图时首先画出圆,然后再根据 GYK. DBF 的数据按比例分块,接逆时针方面旋转作扇形。

#### ①变量说明:

X0,Y0,R:表示圆心坐标和半径;NUM:为扇形数;SUM:为各块大小的总和;PIE:为 $\pi$ 的值;A:为当前扇形与第一个扇形起始边的高角;SINA:为角A的正弦值。

#### ②程序思想

A:以(X0,Y0)为圆心,R为半径作圆(分别以X,Y为自变量作两次)。

B:在点(X0,Y0)和点(X0+R,Y0)之间作横线,以此横线作为第一个扇形的起始边。

C:求出扇形角A。

D:根据 Maclaurin 展开式求出角A的正弦值SINA,根据SINA的值定出扇形终边与圆的交点(X1,Y1)

F:在点(X0,Y0)与(X1,Y1)之间作斜线。

G:画完退出,否则转C。

#### ③程序清单:

```
* FKT. PRG
```

```
SET TALK OFF
```

```
? CHR(14)+"I0,160]"
```

```
PIE=3.1415926
```

```
X0=320
```

```

Y0=170
R=130
C=5
Y=Y0-R
? CHR(14)+"119.0]"
CLEA
D=0
@ 0,30 SAY "产品含硫量扇形图"
DO WHILE D=0
DO WHILE (Y>=Y0-R) .AND. (Y<=Y0+R) .
AND. D=0
X1=X0+SQRT(R*R-(Y-Y0)*(Y-Y0))
X2=X0+SQRT(R*R-(Y-Y0)*(Y-Y0))
? CHR(14)+"D"+STR(C)+","+STR(X1)+","+STR(Y)+"]"
? CHR(14)+"D"+STR(C)+","+STR(X2)+","+STR(Y)+"]"
Y=Y+2.5
@ 0,0 SAY " "
D=INKEY()
ENDD
X=X0-R
DO WHILE (X>=X0-R) .AND. (X<=X0+R) .
AND. D=0
Y1=Y0+SQRT(R*R-(X-X0)*(X-X0))
Y2=Y0-SQRT(R*R-(X-X0)*(X-X0))
? CHR(14)+"D"+STR(C)+","+STR(X)+","+STR(Y1)+"]"
? CHR(14)+"D"+STR(C)+","+STR(X)+","+STR(Y2)+"]"
X=X+2.5
@ 0,0 SAY " "
D=INKEY()
ENDD
? CHR(14)+"H5,"+STR(X0)+","+STR(Y0)+","+STR(R)+"]"
USE GYK
COUNT TO NUM
SUM S TO SUMM
GO TOP
E=0
DO WHILE .NOT. EOF() .AND. D=0
A=S/SUMM*2*PIE+E
B=E+(S/SUMM*2*PIE)/2
M=1
SINA1=10
SINB=0
SINA=0
T=.T.
DO WHILE T.AND. D=0
D=INKEY()
I=1
SIN=1
SINBB=1
ZC=1

```

```

DO WHILE I<2*M .AND. D=0
D=INKEY()
ZC=I*ZC
I=I+1
ENDD
I=1
DO WHILE I<2*M .AND. D=0
D=INKEY()
SIN=SIN*A
SINBB=SINBB*B
I=I+1
ENDD
IF MOD(M,2)=0
NN=-1
ELSE
NN=1
ENDI
SINA=SINA+NN*SIN/ZC
SINB=SINB+NN*SINBB/ZC
M=M+1
IF ABS(ABS(SINA)-ABS(SINA1))<0.00001
T=.F.
ENDI
SINA1=SINA
ENDD
Y1=Y0-R*SINA
YB=Y0-R*SINB
IF (A>PIE/2).AND. (A<3*PIE/2).AND. D=0
X1=X0-SQRT(R*R-R*R*SINA*SINA)
ELSE
X1=X0+SQRT(R*R-R*R*SINA*SINA)
ENDI
IF (B>PIE/2) .AND. (A<3*PIE/2)
XB=X0-SQRT(R*R-R*R*SINB*SINB)
ELSE
XB=X0+SQRT(R*R-R*R*SINB*SINB)
ENDI
IF X1>X0
X=X0
MAX=X1
ELSE
X=X1
MAX=X0
ENDI
IF ABS(X1-X0)>1
DO WHILE X<=MAX .AND. D=0
@ 0,0 SAY " "
Y=(Y1-Y0)/(X1-X0)*(X-X0)+Y0
? CHR(14)+"D"+STR(C)+","+STR(X)+","+STR(Y)+"]"
X=X+0.9
D=INKEY()
ENDD
ELSE
IF A>PIE

```

```

? CHR(14)+"S"+STR(C)+","+STR(X0)+","+
STR(Y0)+","+STR(R)+""]
ELSE
? CHR(14)+"S"+STR(C)+","+STR(X0)+","+
STR(Y0-R)+","+STR(R)+""]
ENDI
ENDI
IF XB>=X0 .AND. YB<=Y0
? CHR(14)+ "["+STR(XB+15)+"|"+STR(YB-15)
+"@u"+LTRIM(STR(S,8,2))+""]
ELSE
IF XB<=X0 .AND. YB<=Y0
? CHR(14)+ "["+STR(XB-10 * LEN(LTRIM(STR(S,
8,2))))+"|"+STR(YB-15)+"@u"+LTRIM(STR(S,8,
2))+""]
ELSE
IF XB>=X0 .AND. YB>=Y0
? CHR(14)+ "["+STR(XB+5)+"|"+STR(YB+5)+
"@u"+LTRIM(STR(S,8,2))+""]
ELSE
IF XB<=X0 .AND. YB>=Y0

```

```

? CHR(14)+ "["+STR(XB-10 * LEN(LTRIM(STR(S,
8,2))))+"|"+STR(YB)+"@u"+LTRIM(STR(S,8,2))+"
"]"
ENDI
ENDI
ENDI
SKIP
E=A
C=C+1
IF C=8
C=2
ENDI
ENDD
ENDD
USE
? CHR(14)+"I19,1"
? CHR(14)+"I0,160"]
RETN

```

〔编者按：本文选自 1991.11 广东省计算机学会数  
据库技术交流会论文集〕

（接第 2 页）

盘的力量。总的来说生理特点校验比行为特点校验更为可靠，但所需投资也相应要高得多。图 1 是一种校验视网膜血管布局的示意图，光在视网膜上进行一次 450 度扫描，对其上的 320 个数据点进行验证——即使是极为相似的双胞胎也不会完全相同。现在美国市场上出售的生理校验设备有 8 种，另有 30 种正在研制中。但因其售价太高（600~7000 美元），目前计算机用户还无法普遍使用。表 1 是美国生理和行为校验设备销售情况，其中售出的设备仅有 16% 用于计算机安全，据报道近几年来这一百分比正在上升。

表 1

| 检验设备   | 售出数量   |        | 平均售价   |        | 最低价    |
|--------|--------|--------|--------|--------|--------|
|        | 1987 年 | 1988 年 | 1987 年 | 1988 年 | 1989 年 |
| 指纹检验机  | 260    | 475    | 7000   | 4200   | 1800   |
| 视网膜扫描仪 | 125    | 175    | 7504   | 7000   | 5000   |
| 手形检验机  | 60     | 130    | 4200   | 3300   | 3000   |
| 声音校验   | 600    | 625    | 1200   | 1200   | 1200   |
| 动态签字辨别 | 125    | 195    | 800    | 800    | 640    |
| 动态击键辨别 | 80     | 0      | 700    | 700    | 600    |
| 总和     | 1250   | 1600   |        |        |        |

注：售价为美元

（接第 7 页）

啮合角的实用的 BASIC 程序，读者据此很容易改写成求解具体问题的程序。

反函数查找程序

```

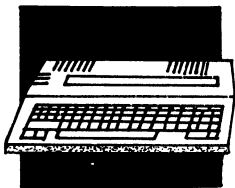
10 INPUT Y#
20 GOSUB 50
30 PRINT X# * 57.295779506#
40 END
50 'search(y,x)
60 D=.0000001
70 K=1
80 AL=131072#
90 BL=AL
100 AL2#=AL * 3.141592654# / 180 / 3600

```

```

110 YY#=TAN(AL2#)-AL2#
120 AQ#=Y#-YY#
130 AQ1=ABS(AQ#)-D
140 IF AQ1<=0 THEN 230
150 CL=BL * 2^(-K)
160 K=K+1
170 IF AQ#>=0 THEN 200
180 AL=AL-CL
190 GOTO 100
200 IF AQ#=0 THEN 230
210 AL=AL+CL
220 GOTO 100
230 X#=AL2#
240 RETURN

```



## 学习机之友

# 英语单词快速记忆系统

天津第七印刷厂设备科(300061) 张振堂

计算机教学的特点有利于因材施教。从教育心理学的角度讲,每个人有不同的情绪、智力,一位教师面对数十位学生,要清醒地掌握他们的智力情况,不是一件容易事。这里,计算机教学的优点就显露出来了。例如,计算机可出题供学习者练习,练习题可根据学生的回答内容动态生成,还能根据学生回答问题的“表现”,“诊断”出学生困难的根源,从而达到对学生逐个“精雕细刻”的目的。

“英语单词快速记忆系统”在这方面做了初步尝试,获得了较好的效果,现将它奉献给读者。

### 一、设计思想

本系统主要依据记忆规律,合理控制单词重复的时间间隔。由记忆特点得知:人们记忆时开始一段时间遗忘较快,随着时间的推移,遗忘逐渐放慢,最后记忆效果得到巩固。因此,欲使生词在遗忘前得到巩固,应确保其出现的频数,而熟词过多的出现不但浪费练习时间,也分散学生的精力,应受到控制。然而,这不应意味着对熟词的忽视。此外,某些人对某些特定的单词接受较快或较慢的现象也是有的。对此应加强系统对这些信息的反馈。

根据以上要求,本系统将附设的单词库分为生词库和熟词库,生词库中存放用户输入的生词。熟词存放已经学会的单词,根据学生对其熟悉程度又被分为九级。练习时,生词库中单词出现的概率最高,熟词库中单词出现的概率随单词级别的提高而递减,任何单词回答正确都可逐级上升到较高级别的词库,若答错了则不管原来级别都降回生词库。这种结构可保证充分练习生词而兼顾复习熟词,用最短的练习时间获取最佳的学习效果。

为了确保学习的连续性及有效性,系统词库内全部信息都可以脱机保存,以便任何时间开机都可接续上次的练习,并达到同样的学习效果。

### 二、主要结构及功能

本系统有多种功能,在一个汉字主菜单控制下进行选择操作,使用极为简单。根据菜单的提示,按一健即可转入相应的功能块。现分别叙述如下:

(1)查看单词库:此项功能可浏览指定级别词库内的单词及有关信息。学生可以由此找出自己的薄弱环节。教师也可以据此总结教学的规律。

(2)输入新单词:根据屏幕提示依次键入单词、词性及汉译,键入过程中屏幕将自动变换状态(字母、拼写)以方便操作。拼写单词时可大写也可小写;使用时应按照输入方式回答。拼写完毕,程序将对系统词库内

单词进行检索,只有词库内没有重复单词时,该单词才会被接纳进入生词库。

(3)删除旧单词:进入此项功能后,屏幕显示一个子菜单,可完成以下子功能:

i 删除指定级别单词:随着学生学习水平的提高,某些级别的单词已被记牢,我们可以将其删除,腾清词库空间,以便容纳新单词。

ii 删除指定单词:输入生词时,如果拼写出现错误,可通过此项功能进行删除,以便重新输入。

iii 删除全部单词:这里,词库内单词全部被删除,因此应慎用此功能。

(4)汉译英练习:由用户指定练习次数,然后开始练习,练习过程中将充分体现前文所叙各项特点。最后,练习结果将显示于屏幕。

(5)统计:词库内各级别单词的数量将被分别统计出来以供参考。我们可以据此了解学生对全部单词的掌握情况。

(6)退出:词库内全部单词及学习者对单词的掌握情况被编制成 DATA 语句,然后系统对使用磁盘机还是磁带机进行鉴别,若是磁盘机,这些数据将会自动存入磁盘,若是磁带,屏幕会提示用户操作录音机以便将数据录入磁带。

### 三、使用说明

这里对使用中可能出现的问题作几点说明:

如使用磁盘作外存,应使磁盘内含有以“ENGLISH”为名的本程序,否则由于寻不到该文件将会出现错误。如果欲改为其它文件名,用户可对 5 号语句的 U\$ 值进行修改即可。

由于本系统的特点在于针对每个学习者进行个别指导,所以不可数人共用一个程序,否则系统的优越性难以发挥。如有 A、B 二人学习,可以将程序分别命名为 ENGLISH-A 及 ENGLISH-B,用户同样应对程序中 U\$ 值进行修改。此后练习中,A、B 二人只可调用磁盘同名程序。

只有经常使用本系统进行练习,系统内才会积累有关你的大量资料,才能了解你,也才能有的放矢地帮助你。因此,只有经常和它交谈,不冷落它,它才会成为你的好助手。

本程序在 CEC-I 机上通过。

```
5 LOMEM:24576;E$=CHR$(18);D$=CHR$(4);F=0;
L=0;V$="请等待...";U$="ENGLISH"
10 REM DISK SYSTEM?
```

```

15 DK=PEEK(994)+PEEK(1001)-192
25 IF DK=0 THEN PRINT D$;"PR#3":GOTO 35
30 PR#3
35 PRINT:PRINT E$;HOME
40 VTAB 5;HTAB 3:PRINT " * * * 英语单词快速记忆系统
 * * * "
45 POKE 1013,76;POKE 1014,23;PRKE 1015,3;POKE 6,
 231;POKE 7,3
50 FOR I=0 TO 114:READ P;POKE 768+I,P;NEXT
55 READ N
60 DIM A%(9),A$(3,N)
65 IF N=0 THEN 80
70 FOR I=1 TO N;FOR J=0 TO 3:READ A$(J,I);NEXT
 J,I
80 HOME:P=FRE(0);HTAB 7:PRINT"(本词库内共含单
 词";N;"个)"
85 PRINT;HTAB 5:PRINT"请选择:";PRINT
90 HTAB 6:PRINT"[1]查看单词库"
95 HTAB 6:PRINT"[2]输入新单词"
100 HTAB 6:PRINT"[3]删除旧单词"
105 HTAB 6:PRINT"[4]汉译英练习"
110 HTAB 6:PRINT"[5]统计"
115 HTAB 6:PRINT"[6]退出";
120 GOSUB 625;IF P<1 OR P>6 THEN 120
125 IF N=0 AND(P=1 OR P=3 OR P=5)OR N<2 AND P
 =4 THEN 120
130 HOME;ON P GOSUB 145,200,275,395,520,575
135 GOTO 80
140 REM 查看
145 VTAB 4;HTAB 5:PRINT"请键入词库级别:(0~9)"
150 GOSUB 625;IF P<0 OR P>9 THEN 150
155 HOME:T=0;A$=STR$(P);Q$=CHR$(127)+
 CHR$(31)+CHR$(46+P)+"级词库"
160 FOR I=0 TO 3;VTAB 11;HTAB 8+I*2:PRINT MID
 $(Q$,3*I+1,3);NEXT
165 FOR I=1 TO N
170 IF A$(<>A$(3,I) THEN 185
175 T=T+1;VTAB T;HTAB 1:PRINT A$(0,I);TAB(16);
 A$(1,I);" ";A$(2,I)
180 IF T=10 THEN T=0;GOSUB 625;HOME
185 NEXT
190 GOSUB 625;HGR2;RETURN
195 REM 输入
200 PRINT E$;VTAB 2
205 INPUT"请键入英语单词: ";Q$;VTAB 2:PRINT" "
210 IF N=0 THEN 230
215 FOR I=1 TO N
220 IF A$(0,I)=Q$ THEN PRINT:PRINT"词库内已有此
 单词。";GOTO 250
225 NEXT
230 N=N+1;GOSUB 645:A$(0,N)=Q$:A$(3,N)="0"

```

```

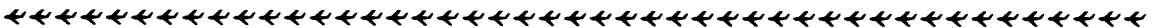
235 PRINT:INPUT"请键入单词词性:";A$(1,N);VTAB 4;
 PRINT" "
240 POKE 942,140;PRINT E$,E$
245 INPUT"请键入汉语译文:";A$(2,N);VTAB 6:PRINT
 " "
250 PRINT:PRINT"是否还输入新单词?"(Y/N)
255 POKE 942,129;PRINT E$
260 GOSUB 625;IF P=41 OR P=73 THEN HOME;GOTO 200
265 RETURN
270 REM 删除
275 HTAB 8:PRINT"请选择:";PRINT
280 PRINT;HTAB 6:PRINT"[1]删除指定分值单词"
285 PRINT;HTAB 6:PRINT"[2]删除指定单词"
290 PRINT;HTAB 6:PRINT"[3]删除全部单词"
295 GOSUB 625;IF P<1 OR P>3 THEN 295
300 HOME;ON P GOSUB 310,335,385
305 RETURN
310 PRINT;HTAB 5:PRINT"大于指定分值的单词将被删
 除。"
315 PRINT;HTAB 5:INPUT"指定分值=?";Q
320 T=0;FOR I=1 TO N
325 IF VAL(A$(3,I))<Q THEN T=T+1;FOR J=0 TO 3:
 A$(J,T)=A$(J,I);NEXT
330 NEXT;N=T;GOSUB 645;RETURN
335 PRINT;HTAB 5:PRINT"请键入欲删除单词:"
340 PRINT;HTAB 8:INPUT"→";Q$
345 FOR I=1 TO N
350 IF A$(0,I)=Q$ THEN 360
355 NEXT;GOTO 380
360 N=N-1;IF I=N+1 THEN 380
365 FOR K=I TO N;FOR J=0 TO 3
370 A$(J,K)=A$(J,K+1);NEXT J,K
380 GOSUB 645;RETURN
385 N=0;GOSUB 645;RETURN
390 REM 练习
395 PRINT" "
400 PRINT" "
405 PRINT" "
410 VTAB 5;HTAB 9:INPUT"单词个数=?";Q
415 VTAB 5:PRINT CHR$(26)
420 VTAB 1;HTAB 30:PRINT Q;R=0
425 FOR I=1 TO Q
430 K=0
435 K=K+1
440 H=INT(RND(1)*N)+1
445 A=VAL(A$(3,H));IF A>K/2 THEN 435
450 IF H=H1 THEN 440
455 H1=H;VTAB 3;HTAB 30:PRINT I
460 VTAB 2;HTAB (3+(21-LEN(A$(1,H))-2*LEN(A
 $(2,H)))/3)/2:PRINT A$(1,H);" ";A$(2,H)
465 VTAB 5;HTAB 2:INPUT"请译成英语→";Q$

```

```

470 IF Q$ = A$(0,H) THEN R=R+1; A$(3,H) = STR
 $(A+(A<9)); MUSIC 85,45; MUSIC 50,50; MUSIC
 80,50; GOTO 485
475 A$(3,H) = "0"; PRINT; HTAB 2; PRINT "正确答案是
 ---"; A$(0,H)
480 VTAB 10; HTAB 8; PRINT "按任何键继续进行"; GO-
 SUB 625
485 VTAB 2; HTAB 3; PRINT SPC(21)
490 VTAB 4; PRINT CHR$(11)
495 NEXT
500 HOME; VTAB 4; HTAB 6; PRINT "正确."; R
505 PRINT; HTAB 6; PRINT "错误."; Q-R
510 GOSUB 625; RETURN
515 REM 统计
520 VTAB 6; HTAB 12; PRINT V$
525 FOR I=0 TO 9; A%(I)=0; NEXT
530 FOR I=1 TO N
535 A=VAL(A$(3,I)); A%(A)=A%(A)+1
540 NEXT; HOME
550 FOR I=0 TO 9
555 VTAB I+1; HTAB 1
560 PRINT I; "级单词库....."; A%(I);
565 NEXT; GOSUB 625; RETURN
570 REM 退出
575 VTAB 6; HTAB 12; PRINT V$
580 CALL 786; N$ = STR$(N); &N$; IF N=0 THEN 590
585 FOR I=1 TO N; A$ = A$(0,I) + " " + A$(1,I) + " " +
 A$(2,I) + " " + A$(3,I); &A$; NEXT
590 P=PEEK(25)+256*PEEK(26)+2; POKE 176,INT(P/
 256); POKE 175,P-256*INT(P/256)
595 IF N=0 THEN POKE P-1,0; POKE P-2,0
600 IF DK=0 THEN PRINT D$; "DELETE"; U$; PRINT D
 $; "SAVE"; U$; TEXT; HOME; END
605 HOME; PRINT; PRINT "请将磁带装入带仓,使录音机处
 于录音状态,再按回车键。"
610 GOSUB 625; IF P<>-35 THEN 610
615 SAVE; TEXT; HOME; END
620 REM SUB-1
625 P=RND(1); P=PEEK(-16384)
630 IF P<128 THEN 625
635 POKE-16368,0; P=P-176; RETURN
640 REM SUB-2
645 F=27+PEEK(107)+256*PEEK(108); L=21+12*N
650 POKE F+2,L-256*INT(L/256); POKE F+3,INT(L/
 256)
655 POKE F+5,INT((N+1)/256); POKE F+6,N+1-256
 *INT((N+1)/256)
660 POKE 109,F+L-256*INT((F+L)/256); POKE 110,
 INT((F+L)/256)
665 RETURN
670 DATA 165,6,133,80,165,7,133,81,32,26,214,160,1,
 177,155,133,26,136,177,155,133,25,96,32,227,223,
 230,6,208,2,230,7,160,4,169,131,145,25,136,165,7,
 145,25,136,165,6,145,25,177,131,133,28,136,177,131,
 133,27,136,
675 DATA 177,131,133,31,24,169,5,101,25,133,29,152,
 170,101,26,133,30,177,27,145,29,200,196,31,208,247,
 138,145,29,200,145,29,200,145,29,136,24,152,101,29,
 160,0,145,25,72,200,138,101,30,145,25,133,26,104,
 133,25,96
999 REM 词库
1000 DATA 0

```



## 大量供应显示器及大功率开关电源

我部现货供应进口 19" 彩色、单色高分辨 (1280×1024) 图形显示器; 12" 终端, 并供应大功率开关电源, 规格有: 5V/18A, 5V/20A, +5V/50A, +5V/100A, 5V/120A, 5V/150A×2, 12V/15A。并有各种进口电脑装配电线、电缆 (单层屏蔽、多层屏蔽、单芯、多芯、镀银等) 品种繁多, 适合各类科研开发、生产、装配、工业自动化等单位使用。

以上产品均全部进口并通过美国 UL 安

全标准认证, 产品批发价相当优惠、欢迎来函、来电或来人洽谈。

广东省四会县南方电子厂经营部

地址: 广东省四会县城高观东路 71 号

邮编: 526200

电话: (07663) 322686

电挂: 1311

# 用 CHR \$ 函数压缩存盘数据

厦门教育学院(361003) 梁才柱

随着微机在我国的普及,“教育测量”的推广和实施已提到日程上来了。运用目前在中小学中持有量很大的中华学习机来实现“教育测量”所要求的统计与分析工作,实践证明是可行的,但需要解决好大量的统计数据与内存和磁盘容量不足这个突出问题。在编写“试题分析”软件的过程中,为了提高磁盘存储空间的利用率,本人使用过 CHR \$ 函数与 ASC 码函数来实现压缩记录成绩数据的存储空间的办法。结果表明,这种方法不但可以有效地节约磁盘的存储空间,而且使用简便,适当加强,还可以用于其他数据的压缩。为提供交流,兹将本方法介绍如下。

数据一般是使用文本文件存盘的。存盘时是以一个编码代表一个数字的方式写入。例如:“108”这个数目,在盘上是以“B1 B0 B8”三个编码存储,并且以回车(8D)作为数据间的分隔。这样,一个“108”就占用四个字节的存储空间。若能另选用一套有规律的代码,它只需用一个代码就能代表一个数据(例如 108)来存盘,并且取消分隔用的“8D”,那么一个数据将只占用一个字节,就可以节省 3/4 的空间。当然,就其整体来说;并不是所有的数据都是三位数,加上其他一些原因,实际节约的空间大约是 1/2 左右。但由于数据量很大,这仍将是一个很可观的数目。

选用什么代码?有没有现成的代码可供选用?回答是肯定的,那就是 CHR \$ 函数与 ASC 码函数。我们知道;CHR \$ 函数可以产生一个以给定的数值为其 ASCII 编码的字符,虽然其中有一部分是不可显示的字符,但代表该字符的编码是存在的,可以写入磁盘。而 ASC 码函数的作用恰好相反;它是产生一个用十进制数表示的字符的 ASCII 码。因此,我们就可以将数据(例如 108)转化为相应的 CHR \$ 给定值,以其编码存入磁盘中。当需要将数据读出时,便利用 ASC 码函数作为解码用,还原出相应的以十进制表示的数目。由于上述可供写入磁盘的编码共有 128 个,而目前的考试评分,最多只有 120 分。即使由于某些原因需要扣除编码中的几个,仍然是够用的。

为了具体说明上述方法的使用及其节约磁盘空间的效果,请运行以下两段小程序。其中:程序一是用普通方法存盘,程序二是用本文介绍的压缩法存盘,两者都是向磁盘写入 0—127 的 128 个数字。程序中的 60 句是两种存盘法差别之所在。

## 程序 1

```
10 INPUT "FILE NAME?";F$
20 D$=CHR$(4)
30 PRINT D$;"OPEN";F$
```

```
40 PRINT D$;"WRITE";F$
50 FOR I=0 TO 127
60 PRINT I
70 NEXT
80 PRINT D$;"CLOSE";F$
100 END
```

## 程序 2

```
10 INPUT "FILE NAME?";F$
20 D$=CHR$(4)
30 PRINT D$;"OPEN";F$
40 PRINT D$;"WRITE";F$
50 FOR I=0 TO 127
60 PRINT CHR$(I+5);
70 NEXT;PRINT
80 PRINT D$;"CLOSE";F$
100 END
```

现在如果用 COPY I PLUS 的 SECTOR EDITOR 功能来观察一下已写入磁盘的内容。可以看出:程序一使用了盘上的 1.5 个扇区,而程序二只用了约半个扇区,短短的 128 个数据,占用空间的差别是如此之大,压缩法的优越性是显而易见的。

最后,有几点需要说明一下:

1. 实际编程时,先用数组寄存每个输入的成绩,然后再通过上述 60 句转化为 CHR \$ 编码存入磁盘。

2. 为了取消数据间的分隔符(8D),CHR \$(I+5) 后的“;”一定要加上,否则将存入与数据量相等的“8D”,白白多占用了一倍的空间。

3. 因此,在读出时最好也用数组接受逐个读出的数据。而且要用 GET 语句而不是用 INPUT 语句,方能把数据逐个读出。

4. 由于磁盘操作时需要用到 CHR \$(4),为防止会影响正确的存取操作,应避免用它。所以要在成绩上多加 5,即“CHR \$(I+5)”句,使“0”这个数从“CHR \$(5)”开始而不是“CHR \$(0)”。为此,在读出时也应该减去 5,才能得到正确的读数。例如:

```
10 FOR I=0 TO 127
20 GET A$
30 E(I)=ASC(A$)-5
40 NEXT
```

5. 以上存取的数据,均为正整数。记录成绩一般不需考虑负数。若要求比较精确的成绩记录,例如有时需要出现半分的场合,可利用“121”(即“CHR \$(126)”,编码为 FE)的编码作为半分的代码而记录入盘,取出时相应还原即可。



# PB-700 微机解密技巧

山东兖州环保局(272000) 孙力

PB-700 微机是一种物美价廉的具有广泛用途的袖珍式计算机。该机所规定的 BASIC 语言中,有专门起保护程序作用的 PASS 指令。当程序加上了任意 8 个字节之内的 PASS 通行令后,程序就进入被保护状态,只能被使用而无被破坏之虞。

本文现介绍一种解开 PASS 通行令的方法,供大家在研究、修改和应用该计算机程序时参考。

首先将所需要解密的程序由 FA-10 磁带机调入主机 P0 区,再换一 P 区调入 CASIO 公司原装磁带 GRAPH SOFTWARE 中的绘图程序(A 面,从计数器 0 开始,屏幕显示为 GRAPH PF B,大约到 40 结束,调入时若打开监听则应听到 131、311 的乐音)。

调毕用 NEWALL 命令将程序全部清除,然后再调入事先任意编写的未加 PASS 通行令的程序于主机。注意:一定要将此编写好的程序在输入磁带时将其中途破坏,即在 SAVE 此程序时中途按停止键或 BRK 键,亦可将此程序输入磁带后再用别的程序在中间输入,使其破坏。当调入该未经 PASS 的程序到显示 RW error 时,原已加密的程序就解密恢复在主机里。若用 SAVEALL 命令输入的加密程序可反复在各区使用此方法解密。

## 双人百米赛跑

南京梅园中学(210001)汪波

运行程序后,先输入两位游戏者的姓名,接着,屏幕中央将出现一把发令枪,枪响后,两位游戏者就可分别通过按(Z,X)和(M,.)两键来控制两运动员的“脚”了。

此游戏最大的特点就是可两人同时玩,且互不干扰,不会出现一方按住键另一方走不了的情况。真正的体现了“竞争性”。

游戏结束时,计算机将打印出冠军,并为其奏乐,祝大家个个成为赛跑好手。

```

10 INPUT "NO. 1,NAME";A$:INPUT"NO. 2,NAME";B$:
MODE(1):COLOR2
20 FOR X=0 TO 127:SET(X,20):SET(X,31):SET(X,42):
NEXT
25 FOR Y=21 TO 41:SET(127,Y):NEXT
30 A=1:B=2:Y1=25:C=1:D=2:Y2=36:COLOR 4
32 SET(60,6):SET(67,6):FOR I=60 TO 66:SET(I,7):
NEXT:SET(64,8)
33 FOR I=8 TO 10:SET(65,I):NEXT:FOR I=8 TO 11:SET

```

```

(66,I):SET(67,I):NEXT
34 SOUND 10,2:FOR I=1 TO 400:NEXT:SOUND 15,2:FOR
I=1 TO 300:NEXT
35 POKE 30862,74:POKE30863,52:H=USR(0):SOUND3,
1:H=USR(0)
40 SET(A,Y1):SET(B,Y1+1):SET(C,Y2):SET(D,Y2+1)
41 IF A=127 OR C=127,100
45 P=PEEK(26875):K=PEEK(26863)
50 IF A>B AND K=223,RESET(B,Y1+1):B=B+2
60 IF A<B AND K=247,RESET(A,Y1):A=A+2
70 IF C>D AND P=239,RESET(D,Y2+1):D=D+2
80 IF C<D AND P=253,RESET(C,Y2):C=C+2
90 GOTO 40
100 SOUND 25.5;22,3;27,3;29,4;26,5;29,6
110 CLS:IF A=127,PRINT@230,A$;"WIN!";GOTO 130
120 IF C=127,PRINT @230,B$;"WIN!";PRINT
130 INPUT "PLAY(Y/N)";K$:IF K$="Y",MODE(1):
COLOR2:GOTO 20

```

## 寻找莱蒙托夫游戏规律

南京金陵职大计算中心(210001) 钱雁群

俄国著名诗人莱蒙托夫可以让人随便想出一个自然数,然后乘以 7,把所得的积的各位数字加起来,接着乘以 9,再把所得积的各位数字加起来,最后乘以 5,除以 3,加上 5,所得结果都是 20!

我们把他的算法剥去外壳,就可发现其中奥秘。他是把一个自然数横加、乘 9 以后再横加,结果必然是 9 (横加就是把一个自然数的个位、十位、百位……分别当作个位数横向求和)。这个游戏是利用九九乘法表中的一条规律:凡是表上乘数为 9 的积,经过一次横加之后,都会得 9。例如  $3 \times 9 = 27$ 。27 横加是 9。

莱蒙托夫方法是:先外套再横加,接着乘以 9,后来再横加一次。最后还有一个外套,当然是变化无穷,次次都准。

我有一个怪想法,反正要经过两次横加,不如外壳之后,先乘以 9,然后再来两次横加。结果会都等于 9 吗?手工验证几个数还行,成千上万个数的验证还得靠计算机。于是编了一个程序。

```

10 INPUT"起始数";W:INPUT"终止数";S
30 FOR A=W TO S
40 B=A*9:PRINT A;"*9=";B,
60 B$=STR$(B):C=LEN(B$):D=0
90 FOR I=1 TO C
100 D$=MID$(B$,I,1):D=D+VAL(D$)
120 NEXT I
130 PRINT D;" ";
140 IF D>9 THEN B$=STR$(D):GOTO 70
150 PRINT:NEXT A
RUN

```

```

起始数 20
终止数 29
20 * 9 = 180 9
21 * 9 = 189 18 9
22 * 9 = 198 18 9
23 * 9 = 207 9
24 * 9 = 216 9
25 * 9 = 225 9
26 * 9 = 234 9
27 * 9 = 243 9
28 * 9 = 252 9
29 * 9 = 261 9

```

RUN

```

起始数 99999998
终止数 100000002
99999998 * 9 = 899999982 72 9
99999999 * 9 = 899999991 72 9
100000000 * 9 = 900000000 9
100000001 * 9 = 900000009 18 9
100000002 * 9 = 900000018 18 9

```

本程序在中华学习机上通过，只要输入要验证的范围即起始数和终止数，计算机便可将这些数乘以 9 再来两次横加，结果应该是 9。

得到了验证，你就可以大胆出题了。我们可以让小朋友任想一个自然数，乘以 9 再横加两次，得到 9，为迷惑他人，在 9 的基础上，有意进行一些四则运算，然后报出结果，这个结果可以恰好是今天的日期或某个小朋友的年龄，朋友，你说这样好玩吗？

有一点请注意：小朋友想出的数一般都不太大，在 99999999 以内的数两次横加即可。如果和大人玩，可能要三次横加哟！

## 也谈“一题多解”

江苏海安县中学(226600) 郑明达

我想你肯定见过这样一道题目：从键盘接受三个整数给变量 A、B、C，然后求出它们中的最大者并输出。针对这一道题目，不少书上给出了这样的答案：

```

10 INPUT "A,B,C=";A,B,C
20 IF B>A THEN A=B
30 IF C>A THEN A=C
40 ? "MAX=";A
50 END

```

如果规定只能用一行程序完成，那该怎么办呢？

请分析一下上面这个程序，你可以发现它的基本思路是：首先，把 A 看成是三数中最大者，然后比较 A 与 B 的大小，如果此时 A 比 B 小，那么把 B 的值给 A，再用此时 A 的值与 C 比较，如果 A 的值比 C 小，那么把 C 的值给 A（即在比较过程中始终保持 A 为较大

者）。最后，输出 A 的值。

概括起来讲主要有两步：第一步找 A 与 B 中的较大者给 A。第二步找此时 A 与 C 的较大者给 A。其实质性步骤是：找两数中的较大者。针对此分析，我想用一行程序解决此问题至少有三种方法：

一、利用符号函数

大家知道式子  $(A+B+SGN(A-B)) * (A-B) / 2$  可以取出 A 与 B 中的较大者。利用这个式子要写出一行程序也就十分容易啦！

```

10 INPUT "A,B,C=";A,B,C;A=(A+B+SGN(A-B))
 (A-B))/2;A=(A+C+SGN(A-C))(A-C))/
 2;? "MAX=";A;END

```

二、利用逻辑运算

大家知道，如果  $A=3, B=2$ ，那么  $A>B$  的结果就是 1（逻辑真），反之则为 0。则式子  $(A>=B) * A + (B>A) * B$  不就可以取出 A 与 B 中的较大者吗？写出的程序如下：

```

10 INPUT "A,B,C=";A,B,C;A=(A>=B)*A+
 (B>A)*B;A=(A>=C)*A+(C>A)*C;?
 "MAX=";A;END

```

三、利用循环语句

大家都知道：当循环 FOR I=1 TO 10;NEXT 执行后 I 的取值为 11，即  $10+1$ ；循环 FOR I=10 TO 1;NEXT 执行后 I 的取值也是 11，即  $10+1$ 。

从上述所述，我想你很快会看懂下面的程序。

```

10 INPUT "A,B,C=";A,B,C;FOR I=A TO B:
 NEXT;A=I-1;FOR I=A TO C;NEXT;A=I-
 1;? "MAX=";A;END

```

最后，请大家思考：如果要输出的不只是数值，还要输出变量名，该怎么修改程序？

~~~~~  
(接第 21 页)

\* BEFE:28✓(以上设计 40 个磁道的磁盘)

\* B5FA:15✓

\* B4BC:15✓

\* B293:15✓

\* B3BC:15✓

\* AEDC:15✓

\* AE9F:15✓

\* AC01:15✓(十六进目录道，用户可自定)

\* AEC5:60✓(4 \* 目录道数)

\* AEC9:64✓(4 \* 目录道加 4)

(以上是设计目录道为 \$ 15，具有 40 个磁道功能的磁盘)

\* ADD:BD(封锁文件名)

(3)CTRL-C✓(退出监控系统)

POKE 40503,0✓(封锁 RESET 功能)

(4)键入 HELLO 内容

(5)INIT HELLO✓(格式化一张目录磁道为 \$ 15，40 个磁道，封锁文件名，封锁 RESET 功能的加密盘)

如在 BASIC 程序的前面加入：

```
0 ONERR GOTO 63999
```

```
63999 GOTO 63999
```

封锁 CTRL-C, (Quit) 功能后加密效果更好。

# 考试的统计分析程序

南京市人民中学(210005) 胡筱罡

对考试进行科学的统计分析,是评估教学质量、研究教学、揭示规律、提高管理水平的前提,为此笔者编写了《考试的统计分析程序》。程序自动绘出试卷分数频数分布统计表,可把某科目原始分数分为若干等距分数段,从高分到低分进行整理,并计算打印频数、频率、总均分、及格率、优(低)分率、学科标准差及相对标准差等,表格输出,清晰明了。而且在高分辨状态下作出分数分布直条统计图,并可自动进行图形硬拷贝。

程序还增强了容错能力,操作者不会因磁盘操作出错,机器中断退出程序而使辛苦输入的数据丢失。

```
0 ONERR GOTO 141
1 GOTO 3
2 POP;HOME;VTAB 6;HTAB 12;PRINT"谢谢使用!再见!";NEW
5 HOME;CLEAR;D$=CHR$(4);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.排序成绩报表";PRINT"6.单科成绩统计";PRINT"7.退出本系统";INPUT"请选择:";X;IF X<1 OR X>7 THEN 5
6 ON X GOSUB 7,8,20,23,25,26,2;HOME;PRINT"请稍候!"GOSUB 51;GOSUB 37;GOSUB 48;GOSUB 37;GOSUB 39;GOTO 5
8 GOSUB 36;GOSUB 45;DIM N$(N),A(N,10);GOSUB 46
20 GOSUB 36;GOSUB 45;DIM N$(N),A(N,10);GOSUB 46
25 POP;GOSUB 36;GOSUB 45;DIM N$(N),A(N,10);GOSUB 46;GOSUB 27;GOTO 5
26 POP;GOSUB 60;GOTO 5
60 GOSUB 36;GOSUB 45;DIM N$(N),A(N,10);GOSUB 46
61 HOME;FOR KM=1 TO 7;READ A$(KM);PRINT KM;".";A$(KM);NEXT KM;RESTORE;PRINT"0.返回"
62 INPUT"分析统计哪一科";J;J=J+1;IF J=1 THEN RETURN
63 INPUT"请输入本科的满分:";MF
64 PRINT"正在统计请稍候!";ZF=0;F1=0;F2=0;F3=0;F4=0;F5=0;F6=0;F7=0;F8=0;F9=0;F0=0
65 FOR I=1 TO N;ZF=ZF+A(I,J);NEXT I
66 S1=0;JF=INT(ZF*100/N+.5)/100
67 FOR I=1 TO N;S1=S1+(JF-A(I,J))*(JF-A(I,J))
68 IF A(I,J)=.9*MF THEN F1=F1+1;GOTO 78
69 IF A(I,J)=.8*MF THEN F2=F2+1;GOTO 78
70 IF A(I,J)=.7*MF THEN F3=F3+1;GOTO 78
71 IF A(I,J)=.6*MF THEN F4=F4+1;GOTO 78
72 IF A(I,J)=.5*MF THEN F5=F5+1;GOTO 78
73 IF A(I,J)=.4*MF THEN F6=F6+1;GOTO 78
74 IF A(I,J)=.3*MF THEN F7=F7+1;GOTO 78
75 IF A(I,J)=.2*MF THEN F8=F8+1;GOTO 78
76 IF A(I,J)=.1*MF THEN F9=F9+1;GOTO 78
77 F0=F0+1
78 NEXT I
79 F9=F0+F9
80 FA=INT(F1*1000/N+.5)/10;FB=INT(F2*1000/N+.5)/10;FC=INT(F3*1000/N+.5)/10;FD=INT(F4*1000/N+.5)/10;FE=INT(F5*1000/N+.5)/10;FF=INT(F6*1000/N+.5)/10
81 FG=INT(F7*1000/N+.5)/10;FH=INT(F8*1000/N+.5)/10;FI=INT(F9*1000/N+.5)/10
82 JG=FA+FB+FC+FD;YF=FA+FB;DF=FG+FH+FI
83 S=INT(SQR(S1/N)*1000+.5)/1000;CV=INT(S*10000/JF)/100
84 INPUT"计算完毕需要打印吗?";O$
85 IF O$="N" THEN 97
86 INPUT"哪一学年";Y
87 INPUT"哪一学期";Q1
88 INPUT"哪次考试?1.期中2.期末";Q2;IF Q2=1 THEN Q$="期中";GOTO 90
89 Q$="期末"
90 INPUT"是1.初中还是2.高中";N1;IF N1=1 THEN N$="初中";GOTO 92
91 N$="高中"
92 INPUT"哪个年级";N2
93 INPUT"哪个班级";BAN
94 POKE 1659,1;POKE 1915,4;POKE 2013,70
95 PRINT TAB(10);"考试质量分析表"
96 PRINT Y;"学年度第";Q1;"学期";Q$;"考试[";A$(J-1);"]学科";N$;N2;"年级";BAN;"班"
97 PRINT" 1.全卷分数分布频数表";PRINT
98 POKE 1915,0
99 PRINT"+-----+-----+-----+-----+-----+-----+
-----+"
100 PRINT"|试卷数|";N;SPC(4-LEN(STR$(N)));"|总均分|";JF;SPC(6-LEN(STR$(JF)));"|及格率|";JG;SPC(5-LEN(STR$(JG)));"%|"
101 PRINT"+-----+-----+-----+-----+-----+-----+
-----+"
102 PRINT"| | |";SPC(8-LEN(STR$(MF)));MF;"-";MF*.9;SPC(9-LEN(STR$(MF*.9)));"|";SPC(4-LEN(STR$(F1)));F1;"人|";SPC(5-LEN(STR$(FA)));FA;"%|"
103 PRINT"| | +-----+-----+-----+-----+-----+-----+
-----+"
104 PRINT"| | |";SPC(8-LEN(STR$(MF*.9-1)));MF*.9-1;"-";MF*.8;SPC(9-LEN(STR$(MF*.8)));"|";SPC(4-LEN(STR$(F2)));F2;"人|";SPC(5-LEN(STR$(FB)));FB;"%|"
105 PRINT"|分| +-----+-----+-----+-----+-----+-----+
-----+"

```

```

--+”
106 PRINT“ |各|”;SPC(8-LEN(STR$(MF*.8-1)));
MF*.8-1;“-”;MF*.7;SPC(9-LEN(STR$(MF*.
7)));“|”;SPC(4-LEN(STR$(F3)));F3;“人|”;SPC
(5-LEN(STR$(FC)));FC;“%|”
107 PRINT“|数| +-----+-----+
--+”
108 PRINT“ |个|”;SPC(8-LEN(STR$(MF*.7-1)));
MF*.7-1;“-”;MF*.6;SPC(9-LEN(STR$(MF*.
6)));“|”SPC(4-LEN(STR$(F4)));F4;“人|”;SPC(5
-LEN(STR$(FD)));FD;“%|”
109 PRINT“|分| +-----+-----+
--+”
110 PRINT“ |分|”;SPC(8-LEN(STR$(MF*.6-1)));
MF*.6-1;“-”;MF*.5;SPC(9-LEN(STR$(MF*.
5)));“|”;SPC(4-LEN(STR$(F5)));F5;“人|”;SPC
(5-LEN(STR$(FE)));FE;“%|”
111 PRINT“|布| +-----+-----+
--+”
112 PRINT“ |数|”;SPC(8-LEN(STR$(MF*.5-1)));
MF*.5-1;“-”;MF*.4;SPC(9-LEN(STR$(MF
*.4)));“|”;SPC(4-LEN(STR$(F6)));F6;“人|”;
SPC(5-LEN(STR$(FF)));FF;“%|”
113 PRINT“|频| +-----+-----+
--+”
114 PRINT“ |段|”;SPC(8-LEN(STR$(MF*.4-1)));
MF*.4-1;“-”;MF*.3;SPC(9-LEN(STR$(MF*.
3)));“|”;SPC(4-LEN(STR$(F7)));F7;“人|”;SPC
(5-LEN(STR$(FG)));FG;“%|”
115 PRINT“|数| +-----+-----+
--+”
116 PRINT“ | |”;SPC(8-LEN(STR$(MF*.3-1)));
MF*.3-1;“-”;MF*.2;SPC(9-LEN(STR$(MF*.
2)));“|”;SPC(4-LEN(STR$(F8)));F8;“人|”;SPC
(5-LEN(STR$(FH)));FH;“%|”
117 PRINT“ | | +-----+-----+
--+”
118 PRINT“ | | |”;SPC(6);MF*.2-1;“分以下”;SPC
(7-LEN(STR$(MF*.2-1)));“|”;SPC(4-LEN
(STR$(F9)));F9;“人|”;SPC(5-LEN(STR$(FI)));
FI;“%|”
119 PRINT“+-+-----+-----+
--+”
120 PRINT“|优分率|”;YF;“%”;SPC(13-LEN(STR
$(YF)));“|低分率|”;DF;“%”;SPC(11-LEN(STR
$(DF)));“|”
121 PRINT“+-----+-----+
--+”
122 PRINT“|学科标准差|”;S;SPC(8-LEN(STR$(S)));
“|相对标准差|”;CV;“%”;SPC(9-LEN(STR
$(CV)));“|”
123 PRINT“+-----+-----+
--+”
124 POKE 1915,4;PRINT
125 PRINT TAB(4);“备注:各率均为以全卷 100 分计算,60

```

```

分以上为及格。”
126 PRINT “80 分以上为优分,40 分以下为低分,各分数段
为满分”;PRINT“100 分计算逐级递减 10 分。”
127 PRINT;PRINT TAB(4);“2. 分数分布直方图”;PRINT
128 POKE 1659,0;HOME;HGR2;HCOLOR=3;HPLOT 20,
0 TO 20,151
129 HPLLOT18,6 TO 20,0 TO 22,6;HPLLOT 273,149 TO 279,
151 TO 273,153
130 FOR I=1 TO 10;HPLLOT 22*I+20,148 TO 22*I+20,
151;NEXT I
131 FOR I=1 TO 6;HPLLOT 20,151-20*I TO 23,151-20
*I;NEXT I
132 VTAB 1;HTAB 4;PRINT“人数%”;VTAB 3;FOR I=60
TO 10 STEP-10;PRINT I;NEXT I
133 VTAB 9;HTAB 33;PRINT“分 0 10 20 30 40 50 60 70 80
90 100”;;HPLLOT 20,152 TO 279,152
134 F(1)=FA;F(2)=FB;F(3)=FC;F(4)=FD;F(5)=FE;
F(6)=FF;F(7)=FG;F(8)=FH;F(9)=INT(F9*
1000/N+.5)/10;F(10)=INT(F0*1000/N+.5)/10
135 FOR I=10 TO 1 STEP-1;IF F(I)=0 THEN 137
136 HPLLOT 22*(10-I)+20,151 TO 22*(10-I)+20,151
-F(I)*2 TO 22*(11-I)+20,151-F(I)*2 TO 22*
(11-I)+20,151
137 NEXT I
138 PRINT CHR$(19)
139 IF O$="Y"THEN TEXT;PRINT D$;“PR #1”;POKE
1913,2;PRINT CHR$(17);PRINT D$;“PR #0”
140 PRINT D$;“PR #3”;GOTO 61
141 HOME;RR=PEEK(222)
142 IF RR>10 OR RR<5 OR RR=7 THEN 5
143 ON RR-4 GOTO 144,145,146,147,148,149
144 PRINT“磁盘数据不够! 按任意键返回!”;CHR$(19);
GOTO 5
145 PRINT“文件找不到! 按任意键返回!”;CHR$(19);GO-
TO 5
146 GOTO 142
147 PRINT“磁盘接口错误! 检查完毕后,按任意键再次存
盘!”;CHR$(19);GOSUB 37;GOSUB 48;GOSUB 37;
GOSUB 39;GOTO 5
148 GOSUB 48;PRINT“此盘已满! 换盘后,按任意键再次存
盘!”;CHR$(19);GOSUB 37;GOSUB 48;GOSUB 37;
GOSUB 39;GOTO 5
149 INPUT“文件加了锁! 需解锁后再次存盘吗?”;O$;IF O
$="N"THEN 5
150 PRINT D$;“UNLOCK”;T$;GOSUB 37;GOSUB 48;GO-
SUB 37;GOSUB 39;GOTO 5

```

## 氯化氢制取的动态显示

江西萍乡中学(337000) 刘萍

在高一化学氯化氢制取中,水吸收 HCL 气体这一动态现象,在实验室里很难实现,化学老师只能通过

讲解来描述这个过程,我编写一个程序来显示这个动态现象,经过讲述和图形显示两结合、在教学取得了良好的效果。

#### 操作过程

输入程序,运行程序,按任意键——加水,按任意键——加氯化氢气体,在显示屏上,可看到水吸收HCL气体的过程,若要结束实验只需按A键即可。

水吸收氯化氢气体的演示程序

```

10 HGR2;HCOLOR=3;J=140
30 H PLOT 80,90 TO 80,190 TO 179,190 TO 179,80 TO
 70,80 TO 80,90;H PLOT 80,140 TO 179,140
80 FOR I=80 TO 124;H PLOT I,J;
 J=J-1;NEXT I
115 J=96
120 FOR I=135 TO 179;H PLOT I,J;J=J+1;NEXT I
160 H PLOT 124,96 TO 124,40 TO 40,40 TO 40,29 TO 136,
 29 TO 136,96
194 GET A $
196 H PLOT 190,160 TO 190,180 TO 194,180 TO 194,160
197 H PLOT 187,163 TO 192,158 TO 198,163
198 H PLOT 220,160 TO 220,180 TO 217,175
199 H PLOT 210,163 TO 215,170 TO 208,178;H PLOT 230,
 163 TO 225,170 TO 232,178
202 FOR I=1 TO 50
203 H PLOT 80,190-I TO 179,190-I;NEXT I
204 GET B $
210 H PLOT 50,10 TO 40,10 TO 40,14 TO 50,14;H PLOT 47,
 7 TO 56,12 TO 47,17
220 H PLOT 60,5 TO 60,20;H PLOT 60,12 TO 70,12;H PLOT
 70,5 TO 70,20
230 H PLOT 85,7 TO 85,5 TO 77,5 TO 75,7 TO 75,17 TO
 77,20 TO 85,20 TO 85,18
240 H PLOT 91,5 TO 91,20;H PLOT 90,5 TO 92,5;H PLOT
 90,20 TO 92,20
290 C=0;D=3;G=2
300 GOSUB 704;GOSUB 805
370 A=82;B=177
380 FOR I=1 TO 30
385 HCOLOR=3
390 H PLOT A,140-I TO B,140-I
395 A=A+1;B=B-1
396 IF (I/2)=INT (I/2) THEN 400
397 HCOLOR=0
398 H PLOT 81,140+(I-1)/2 TO 178,140+(I-1)/2
400 NEXT I
410 C=1;D=4;G=4;GOSUB 704
610 A=111;B=148
615 FOR J=1 TO 30
620 HCOLOR=0
630 H PLOT A,109+J TO B,109+J
640 HCOLOR=6
650 FOR I=A TO B STEP 3;H PLOT I,109+J;NEXT I
660 A=A-1;B=B+1
664 IF (J/2)=INT (J/2) THEN 670
665 HCOLOR=3

```

```

666 H PLOT 81,140+(J-1)/2 TO 178,140+(J-1)/2
670 NEXT J
672 C=0;D=3;G=2
675 GOSUB 704
676 P=PEEK(-16384)
677 IF P=193 THEN TEXT:RUN
680 GOTO 370
704 IF C/2=INT (C/2) THEN E=6;GOTO 710
705 E=2
710 FOR I=39 TO 124
716 IF D=3 THEN F=4;GOTO 718
717 F=3
718 HCOLOR=0
720 FOR J=30 TO 40 STEP F
730 H PLOT I,J;NEXT J
735 HCOLOR=E
736 FOR J=30 TO 40 STEP D
737 H PLOT I,J;NEXT J,I
760 FOR J=29 TO 96 STEP 3
765 HCOLOR=0
766 FOR I=126 TO 134 STEP 2
767 H PLOT I,J;NEXT I
768 HCOLOR=E
769 FOR I=126 TO 134 STEP G
770 H PLOT I,J;NEXT I
780 NEXT J
803 RETURN
805 A=126;B=133
810 FOR J=96 TO 139
820 FOR I=A TO B STEP 3
830 H PLOT I,J
840 NEXT I
850 A=A-1;B=B+1
860 NEXT J
870 RETURN
880 END

```

## 磁盘加密一法

广东韶关市四中(512000) 黄晓晖

本法运用改变目录磁道号的方法来达到加密的目的(这里设目录道号为\$15),所以无法用目录道为\$11的标准盘起动。再加上一些其它加密法,那效果更好。用此加密法,只能用较高级具有全盘复制功能(40道)的拷贝盘才能拷贝,而且必须运行本盘,文件才能运行。方法如下:

- (1)运行 DOS 3.3
- (2)键入 CALL-151 (进入监控系统)
- \* AEB5;A0✓
- \* B3EF;28✓

(转第18页)

## 第九章 堆栈程序设计

南京大学大气科学系(210008) 朱国江

### § 5. 堆栈程序设计

堆栈是计算机信息处理中的一个重要概念。它实际上是在随机存储器 RAM 中开辟的某个特定的存储区域,用于对一些重要数据的暂存和保护断点地址。这个存储区域有这样的特点:数据可以连续存入,断点地址可以得到保护,不会发生冲掉已存入数据和断点地址的情况,并且严格按照先存入的数据后取出来的规则工作。这种存储结构就叫做堆栈(Stack)。

6502 规定堆栈设置在第 1 页位置上(第 1 页是指地址码高字节为 01 的存储区域),即只能把 \$ 0100—\$ 01FF 这 256 个单元用作堆栈。由于堆栈中的数据取存是按照“先进后出,后进先出”的顺序进行的,因而还必须有一个“指示器”,随时指出栈顶地址的变化,指出栈区中存入的数据已经堆到了哪里,或者说,其作用是指示堆栈中允许存取操作的当前地址。并且必须具有在数据出栈时有自动加 1 的功能,数据进栈时有自动减 1 的功能,这个指示器叫做堆栈指针寄存器,简称栈指示器,记作 S(堆栈指针 Stack Pointer)。它是一个 8 位寄存器,即 S 表示堆栈地址的低 8 位,而高 8 位固定为 \$ 01。

栈的操作只有两种,一是压入(PUSH),一是弹出(POP),每压入(或弹出)一个内容的字节,栈指针 S 就减 1(或加 1),微处理器就是根据 S 指出的位置(栈顶地址)取存数据,保证了按“先进后出”的原则去转子程序,保护断点,恢复断点,保护现场及恢复现场。

附带说一句,堆栈在使用过程中,栈指针 S 总是指向栈顶的一个空单元,而 S 的初值通常设在栈底位置 \$ 01FF 处。

关于堆栈的几条指令简述如下:

- 累加器进栈指令 PHA—— $A \rightarrow MS, S-1 \rightarrow S$
- 累加器出栈指令 PLA—— $S+1 \rightarrow S, MS \rightarrow A$
- 标志寄存器进栈指令 PHP—— $P \rightarrow MS, S-1 \rightarrow S$
- 标志寄存器出栈指令 PLP—— $S+1 \rightarrow S, MS \rightarrow P$

它们各自的操作码分别为:48,68,08,28。

〔例 1〕数据块的倒序传送

例如,将 \$ 6001—\$ 6010 单元的 16 个数据,按相反次序传送到 \$ 7001—\$ 7010 的连续单元中去。程序如 9.1 所示。

程序 9.1:

```
1000-A2 01 LDX # $ 01
1002-A0 10 LDY # $ 10
1004-BD 00 60 LDA $ 6000,X
1007-48 PHA
```

```
1008-E8 INX
1009-88 DEY
100A-D0 F8 BNE $ 1004
100C-A2 01 LDX # $ 01
100E-68 PLA
100F-9D 00 70 STA $ 7000,X
1012-E8 INX
1013-E0 11 CPX # $ 11
1015-D0 F7 BNE $ 100E
1017-60 RTS
```

运行前:

\* 6001.6010

6001-01 02 03 04 05 06 07

6008- 08 09 0A 0B 0C 0D 0E 0F

6010- 10

运行后:

\* 1000G

\* 7001.7010

7001-10 0F 0E 0D 0C 0B 0A

7008-09 08 07 06 05 04 03 02

7010-01

程序 9.1 的几点说明:

• 初始化条件:变址指针存于 X 寄存器,开始为 01→X,数据长度计数器 Y。

• 这是一个双循环结构的程序设计。第一个循环, \$ 1004—\$ 100B,第二个循环 \$ 100E—\$ 1016。前者取一个数,进栈保存, $X+1 \rightarrow X, Y-1 \rightarrow Y$ ,只要 16 个数未取完,就循环上去(\$ 1004),不断取数,不断进栈保存,当全部源数据取完,进栈保存好后,转入第二个循环。后者,把第一个循环在栈中保存的数,逐一取出来,放入目的地址(开始 \$ 7001),用 X 作计数器,并控制循环次数,一直到所有数(\$ 10=16)全部顺次出栈,并在目的地址一一保存后结束。

• 本例是学习堆栈概念存取原则的一个典型实例。若用中华学习机单步执行命令 S,就可以看到数据如何一个一个地进栈,又如何执行后进先出的原则一个一个地出栈的。

• 若改本题为 255 个数据传送,也是非常方便的,您能有最简便的方法吗?事实上,只要将 \$ 1003 单元改为 FF, \$ 1014 单元改为 00,即可。

〔例 2〕N 个数据的交换

例如,将 \$ 6000—600F 单元的内容和 \$ 7000—\$ 700F 单元的内容交换。见程序 9.2。

程序 9.2:

```

1000-A2 00 LDX # $00
1002-BD 00 60 LDA $ 6000,X
1005-48 PHA
1006-BD 00 70 LDA $ 7000,X
1009-9D 00 60 STA $ 6000,X
100C-68 PLA
100D-9D 00 70 STA $ 7000,X
1010-E8 INX
1011-E0 10 CPX # $ 10
1013-D0 ED BNE $ 1002
1015-60 RTS

```

运行前:

```

* 6000.600F
6000- 50 51 52 53 54 55 56 57
6008- 58 59 5A 5B 5C 5D 5E 5F
* 7000.700F
7000- 12 13 14 15 16 17 18 19
7008- 1A 1B 1C 1D 1E 1F 20 21

```

运行后:

```

* 1000G
* 6000.600F
6000- 12 13 14 15 16 17 18 19
6008- 1A 1B 1C 1D 1E 1F 20 21
* 7000.700F
7000- 50 51 52 53 54 55 56 57
7008- 58 59 5A 5B 5C 5D 5E 5F

```

几点说明:

• 程序 9.2 非常简短,并且只用了一重循环,就可以完成 16 个数据的互换。事实上本程序通用性也好,只要将 CPX # \$ 10 中的立即数改成任意数 N(N ≤ FF),即可完成 N 个数据的互换。不仅如此,源地址(本例是 6000)、目的地址(本例是 7000)也可以任意改变,从而可以完成从任何一个区域向另外一个区域的数据交换。

• 程序 9.2 中数据的交换,实际上是借助了一个中间存储区域作为数据交换的暂存空间,而这个暂存空间不是别的,正好是堆栈。其思路是,取一个数,进栈保存,调另一个数,存入原来数的地方(交换),数据出栈,再放数又一次交换,指针加 1,只要未将全部数据交换后,即循环上去,直到结束。

• 请注意,程序 9.2 只能用 PHA,PLA 指令,而不能 PHP,PLP 指令。为什么? 请读者自行分析。

### 〔例 3〕保护和恢复现场的方法

前面一节在谈到子程序设计时,曾指出当主程序转入子程序时,若主程序中已经用到的寄存器(指 X、Y、A、P)在子程序中也要用到,而且当子程序返回主程序后,主程序还要继续用这些寄存器。这时,为了防止寄存器内容被冲掉,就有一个保护现场和恢复现场的问题。即:在转子程序前应首先把这些寄存器的状态存入堆栈中保存起来(保护现场),然后执行子程序的主体部份,最后再返回;而当返回主程序之前,还要从堆栈中取出它们,以恢复寄存器原内容(恢复现场)。

下面的程序指出了保护现场和恢复现场的方法。

```

PHP ;P 进栈,保留 P
PHA ;A 进栈,保留 A
TXA ;X→A
PHA ;X 进栈,保留 X
TYA ;Y→A
PHA ;Y 进栈,保留 Y
: ...
PLA ;最后进栈的 Y 先出栈,送入 A
TAY ;A→Y,恢复 Y
PLA ;X 出栈,X→A
TAX ;A→X,恢复 X
PLA ;A 出栈,恢复 A
PLP ;P 出栈,恢复 P
: ...

```

由此程序可以看出:

• 保留现场时各寄存器进栈的顺序和恢复现场时各寄存器出栈的顺序恰好相反。这是因为各寄存器状态在栈中保存又从栈中出来,必须按先进后出的原则处理,这是堆栈的结构决定的。

• 堆栈指令中有 P 和 A 的进栈指令和出栈指令。因此,P 寄存器和 A 累加器需要进栈时直接使用 PHP 和 PHA 指令;而在出栈时也可直接用 PLP 和 PLA 指令。

• 6502 指令中没有 X 和 Y 寄存器的进出栈指令,因此它们的进栈出栈,需要经累加器 A 中转。即:X 和 Y 寄存器要进栈时,必须先用 TXA 或 TYA 指令,把进栈的内容先送到累加器 A 中,然后再使用 PHA 指令使 X 或 Y 的内容进栈;出栈时,则先用 PLA 指令,将 Y 或 X 的内容送入累加器 A 中(出栈),然后再用 TAX 指令恢复 X,或用 TXA 指令恢复 Y。

### 〔例 4〕标志寄存器 P 进出栈的一个实例

关于求最大值的问题,我们曾在循环程序设计(例 2,程序 7.3)和子程序设计(例 3,程序 8.3)中作过介绍,其中程序 8.3 中就是一个标志寄存器 P 进出栈的实例,不妨将它重新写出,并分析一下它保护的是什么现场,有什么用,恢复的是什么内容,应该在什么时候用。程序 8.3 的主程序:

```

1000-A9 01 LDA # $01
1002-85 06 STA $ 06
1004-A9 60 LDA # $60
1006-85 07 STA $ 07
1008-AC 00 60 LDY $ 6000
100B-20 11 10 JSR $ 1011
100E-85 08 STA $ 08
1010-60 RTS

```

前面 4 条指令,是把源数据块首地址送入 \$ 07 和 \$ 06 单元,第 5 条指令将数据块长度存入 Y 寄存器,以后是转子程序,待子程序执行完毕后存结果再结束。

程序 8.3 的子程序:

```

1011-A9 00 LDA # $00
1013-88 DEY
1014-08 PHP
1015-D1 06 CMP ($ 06),Y

```

```

1017-B0 02 BCS $101B
1019-B1 06 LDA ($06),Y
101B-28 PLP
101C-D0 F5 BNE $1013
101E-60 RTS

```

这段子程序的任务是找最大值。我们主要看 \$1013—\$101D 这段循环,而特别注意循环程序中标志寄存器 P 进栈指令 PHP 和出栈指令 PLP 的安排。循环前 A 中的内容为 00,这是一个假想的最大值(也可以取数据块中的第一个数据)。

执行 DEY 指令使 Y-1→Y,立即执行 PHP 指令,这样就使 P 进栈,从而保存了零标志 Z,如果执行 DEY 指令后结果为 0,则 Z 被置 1;反之 Z 被置为 0。这就为以后数据块长度是否被比较完留下了一个判断标志。当执行 CMP (\$06),Y 指令后,判断下一个数是否大于最大值,若是,换最大值放入 A,不是则保留最大值不变,但不管是与否,最后都转向执行 PLP 指令,这条指令是 P 出栈,给出 Z 标志,再用 BNE 判别是否 Z=0,是 0,返主程序(通过 \$101E 的 RTS 指令),非 0 继续转向 \$1013 处循环再比。然后 Y-1→Y,再使 P 进栈,每次均保留 P 的现场,一直到 A 是最大数,使 P 出栈,恢复现场,并当 P 中的 Z 标志位为 1 时,结果为 0(所有数据均比较完),结束子程序操作,返回主程序的断点地址 \$100E,存结果结束。

〔例 5〕输出码转换程序

将 6502 的 A 中 8 位二进制数转换成三位 10 进制数,并在显示器上显示(送显示器的字符必须用 ASCII 码)。

示范题:a) A=FF (16 进制数)

结果:显示 255 (10 进制数)

b) A=80 (16 进制数)

结果:显示 128 (10 进制数)

c) A=64 (16 进制数)

结果:显示 100 (10 进制数)

我们先给出完成本题的程序 9.3,然后再给予说明。

程序 9.3:

```

1000-A2 00 LDX # $00
1002-C9 64 CMP # $64
1004-90 06 BCC $100C
1006-E9 64 SBC # $64
1008-E8 INX
1009-4C 02 10 JMP $1002
100C-20 24 10 JSR $1024
100F-A2 00 LDX # $00
1011-C9 0A CMP # $0A
1013-90 06 BCC $101B
1015-E9 0A SBC # $0A
1017-E8 INX
1018-4C 11 10 JMP $1011
101B-20 24 10 JSR $1024
101E-09 B0 ORA # $B0
1020-20 F0 FD JSR $FDF0

```

```

1023-60 RTS
1024-48 PHA
1025-8A TXA
1026-09 B0 ORA # $B0
1028-20 F0 FD JSR $FDF0
102B-68 PLA
102C-60 RTS
102D-A9 FF LDA # $FF
102F-20 00 10 JSR $1000
1032-20 48 F9 JSR $F948
1035-A9 80 LDA # $80
1037-20 00 10 JSR $1000
103A-20 48 F9 JSR $F948
103D-A9 64 LDA # $64
103F-20 00 10 JSR $1000
1042-60 RTS

```

运行: \* 102DG

显示: 255 128 100

程序 9.3 说明:

• 本程序结构比较复杂,但可以分为三大段: \$1000—\$1023 为子程序 1, \$1024—\$102C 为子程序 2, \$102D—\$1042 为主程序。在主程序中,既有调子程序 1 的操作,又有调监控中子程序(入口地址 \$F948)的操作。而子程序 1 中又包含调子程序 2 的操作,这是一个二重子程序的嵌套结构。而从主程序看,则是一个主程序调三个子程序的复杂操作。

• 主程序中,首先将 A 赋值 FF(即最初输入的 16 进制数),然后转子程序 1,完成 16 进制数到 3 位 10 进制数的转换,结果为 255,再调 \$F948 的监控子程序,它的功能为空一格。然后输入第 2 个 16 进制数 80,转子程序 1,完成 \$80 向 3 位 10 进制的转换,结果为 128,再次调 \$F948 子程序空一格,最后输入第 3 个 16 进制数 64,转子程序 1,得 10 进制数 100,主程序结束。由此可见,主程序是为了不断送数到累加器 A 中,完成 3 个 16 进制数的赋值工作。

• 子程序 1 在执行过程中还要调用子程序 2,从这个角度上说,子程序 1 又相当于子程序 2 的主程序,这样的主从关系应该明确。这里,分析一下子程序 2,它是在数据换成 ASCII 码之前,首先把累加器 A 的内容(实际上是余数)进栈保护(保护现场),所以返主以后(此时应理解返回子程序 1,子程序 1 为主程序),主程序(即子程序 1)还要再接着使用这个余数,因此, A 中的内容不能由子程序(子程序 2)的执行而丢失。同时,子程序 2 也要用 A 来存放待显示的数,所以需要先把 A 保护进栈,然后再执行子程序 2 的内容。此外,还应注意,在子程序 2 内容执行完毕返主(子程序 1)之前,必须用出栈指令使恢复 A(余数出栈恢复)的内容(恢复现场),以返主后供主程序(子程序 1)继续使用。

• 作为一般情况,如果主程序和子程序使用的寄存器中,发生冲突的不只是 A,还有 X、Y、P 也发生冲

(转第 31 页)



# 一九九二年计算机初级软件人员竞赛 试题解答与分析

## 参考答案

题号	小题号与答案														得分		
一	a	(2)			b	(4)			c	(8)			d	(7)			8
	e	(10)			f	(12)			g	(13)			h	(16)			
二	1	(4)			2	(2)			3	(3)(6)(8)(5) (1)(2)(7)(4)			4	(4) (6)			22
	5	(4)(2)(3)			6	(4)(1)(3)(2)			7	(2) (4)			8	(1)			
三	1	B	2	C	3	B	4	B	5	D	6	D	7	A,C	8	A,C	10
四	1	✓	2	×	3	✓	4	×	5	×	6	×	7	✓,✓	8	×,✓	10
五	1	B		2	C		3	D		4	A		5	D		15	
六	1	C		2	A		3	B		4	B		5	D		15	
七	(1)	R < B		(2)	R		(3)	R = 0		(4)	B		(5)	R		10	
八	1	D,G			2	A			3	D,A,A			4	F			14
九	1	A			2	D			3	C,F			4	C,F			26
	5	C			6	A			7	C			8	B,D,F,G			
十	(1)	屏幕		(2)	诊断程序		(3)	中断		(4)	调试		(5)	联机		10	
	(6)	工作站		(7)	编译		(8)	临时区		(9)	虚拟地址		(10)	缺省			
十一	(1) 程序太大,内存容纳不下.																10
	(2) 文件分配表坏																
	(3) 内部栈溢出,系统被停止.																
	(4) 在拷贝之前,目标文件的内容已经丢失																
总分																	150

### 1991 年计算机软件水平考试试题及解答

本书为全国计算机软件水平考试的复习辅导材料,它是由中国软件行业协会考试指导中心组织清华大学、北京大学、中国科大研究生院、北京信息工程学院、南京东南大学、上海科技大学以及本考试指导中心、北京软件水平考试实施办等单位的有关专家对 1991 年度我国软件水平考试试题(程序员级、高级程序员级、系统分析员级)做了系统分析、研究,并做出了分析和参考答案,以供读者参考,亦可供各部门举办辅导班用作教材。定价 7 元。(邮购时另加 15% 邮费)。邮购地址:北京学院路 29 号,中国软件行业协会高档微机协会(100083)。

## 试题分析

**[第一题]** 这道题是数制及其转换的自测题。

表格的第 1 行:

$$(01000010)_2 = (102)_8 = (42)_{16}$$

因此  $c = 102$ 、 $g = 42$

表格中的第 2 行:

$$(127)_{10} = 128 - 1 = 2^7 - 1 = (10000000)_2 - 1 = (01111111)_2$$

$$(01111111)_2 = (177)_8 = (7F)_{16}$$

因此  $a = 01111111$ 、 $d = 177$

表格中的第 3 行:

$$(01010101)_2 = 2^6 + 2^4 + 2^2 + 2^0 = 64 + 16 + 4 + 1 = (85)_{10}$$

$$(01010101)_2 = (125)_8 = (55)_{16}$$

因此  $e = 85$ 、 $h = 55$

表格中的第 4 行:

$$(143)_8 = (01100011)_2 = (63)_{16}$$

$$(01100011)_2 = 2^6 + 2^5 + 2^1 + 2^0 = 64 + 32 + 2 + 1 = (99)_{10}$$

因此  $b = 01100011$ 、 $f = 99$

**[第二题]**

1. 同一个逻辑函数可以有繁简不同的表达式, 实现它的电路也不同, 采用什么样的表达式, 才能使电路所用的元器件最少、设备简单呢? 一般说来, 逻辑表达式越简单, 电路使用的元器件就越少。

常用的化简方法是代数化简法, 直接利用布尔代数的基本公式和规则进行化简。

这里介绍几种常用的化简方法:

(1). 合并项法: 利用公式  $AB + \overline{A}B = B$ , 将两项合并为一项。

$$\begin{aligned} \text{例如: } & A(BC + \overline{B}C) + A(\overline{B}C + \overline{B}C) \\ & = ABC + \overline{A}BC + \overline{A}BC + \overline{A}BC \\ & = AB + \overline{A}B \\ & = B \end{aligned}$$

(2). 吸收法: 利用公式  $A + \overline{A}B = A + B$  及  $AB + \overline{A}C + BC = AB + \overline{A}C$ , 消去多余项。

$$\begin{aligned} \text{例如: } & \overline{A}B + \overline{A}BCD(E + F) \\ & = \overline{A}B \end{aligned}$$

(3). 消去法: 利用公式  $A + \overline{A}B = A + B$ , 消去多余项。

$$\begin{aligned} \text{例如: } & \overline{A}B + \overline{A}C + \overline{B}C \\ & = \overline{A}B + (\overline{A} + \overline{B})C \\ & = \overline{A}B + \overline{A}C \\ & = \overline{A}B + C \end{aligned}$$

(4). 配项法: 将某一个乘积项乘以  $(1 = A + \overline{A})$ , 这样使一项拆为两项, 或者利用公式  $AB + \overline{A}C = AB + \overline{A}C + \overline{A}B$

$+ \overline{A}C = AB + \overline{A}C + \overline{A}B$ , 左逻辑表达式中增加  $\overline{A}B$  项, 再与其它乘积项合并化简, 达到求得最简单结果的目的。

此题可以用配项法化简。

$$\begin{aligned} & \overline{A}B + \overline{B}C + \overline{B}C + \overline{A}B \\ & = \overline{A}B + \overline{B}C + \overline{B}C(A + \overline{A}) + \overline{A}B(C + \overline{C}) \quad (\text{配项}) \\ & = \overline{A}B + \overline{B}C + \overline{A}BC + \overline{A}B\overline{C} + \overline{A}BC + \overline{A}B\overline{C} \\ & = \overline{A}B + \overline{B}C + \overline{A}C(\overline{B} + B) \quad (\text{合并}) \\ & = \overline{A}B + \overline{B}C + \overline{A}C \end{aligned}$$

这道题的答案应选择(4)

在每年的水平考试中, 都有关于数制及其转换方法、逻辑表达式化简的试题, 在复习时要多做练习题, 以便在考试时能快速、准确地进行计算和化简。

2. DMA 是“直接存储器存取”的缩写。DMA 控制器是控制外部输入输出设备与主存储器直接传送信息(数据)的逻辑电路。

在 CPU 与外部输入输出设备之间传送信息时, 由于 CPU 的速度比输入输出设备快得多, 为了不浪费宝贵的 CPU 时间, CPU 采取分时并行工作的方法。例如, 在磁盘存储器与主存储器之间传送数据期间, CPU 照常执行程序。当磁盘存储器准备好数据时, 向 CPU 发出一个称之为“DMA”的请求信息, 这时 CPU 让出总线, 让出时间为主存储器的一个存取周期, 完成数据写入主存储器的操作, 然后 CPU 继续执行程序。当磁盘存储器再次准备好数据时, 重复上述过程。

这道题的答案应选择(2)。

3. 计算机系统的主存储器用来存放需要立即使用的程序和数据, 它由 CPU 访问, 与 CPU 关系非常密切, 因此要求它存取速度快, 通常由半导体存储器构成。与辅助存储器相比较, 存储容量小、价格高、存储的信息不能永久性地保留。辅助存储器的特点是“存储容量大、价格低、可以永久性地脱机保存信息”。正因为同时使用了这两种类型的存储器, 解决了存取速度、价格成本、存储容量三者之间的矛盾, 提高了计算机系统的性能价格比。

辅助存储器主要有磁表面存储器和光存储器两大类。磁盘、磁带都是磁表面存储器, 它的工作原理是将磁性材料涂覆在盘片基体上形成记录介质, 利用磁头与记录介质的相对运动来存取信息。光盘是一种光存储器, 它的工作原理是利用激光束在具有感光特性的表面上存储信息。激光

可以聚焦成能量高度集中的极小光点，为高度存储信息提供了可能性。因此，它的存储容量比磁盘大。

4. 一个字节由 8 个二进制位组成。它能表示  $2^8 = 256$  个字符。由题目的叙述中可知，它已经表示了 26 个英语字母和 10 个数字，因此它还能表示  $256 - (26 + 10) = 220$  个字符。

二个字节有 16 个二进制位，能表示  $2^{16}$  个字符；一个字节有 8 个二进制位，能表示  $2^8$  个字符。前者能表示字符数量是后者的  $2^8 = 256$  倍。

$$2^{16} / 2^8 = 2^8 = 256$$

这道题的第一个填空应选择答案 (4)，第二个填空应选择答案 (6)。

5. 显示设备有若干种分类方法。

按所采用的显示器件分类，有阴极射线管 (CRT) 显示器、液晶显示器、等离子显示器等。CRT 显示器有黑白和彩色两种；以分辨率的不同，可分为低分辨率显示器和高分辨率显示器两种；以扫描方式的不同，分成光栅扫描显示器和随机扫描显示器两种。

按显示的信息内容分类，有字符显示器、图形显示器和图象显示器三大类。

按显示设备的功能分类，有普通显示器和显示终端两大类。前者的功能简单，只能用于接收视频信号，它的控制逻辑电路和存储逻辑电路都在主机接口板中。后者的功能较多，是一种独立完整的输入输出设备，可以通过标准通信接口接到远离主机的地方使用。

这道题的第一个填空应选择 (4)，第二个填空应选择 (2)，第三个填空应选择 (3)。

6. 分辨率指的是显示设备所能表示的象素个数。象素越密，分辨率越高，图象越清晰。显示器的分辨率取于显象管荧光粉的粒度、荧光屏的尺寸和 CRT 电子束的聚焦能力。同时刷新存储器要具有与显示象素数相对应的存储空间，用以存储每个象素的信息。

灰度级指的是所显示象素点的亮暗差别，在彩色显示器中则表现为颜色的不同。灰度级越多，图象层次越清楚逼真。灰度级取决于每个象素的刷新存储器单元的位数、CRT 本身的性能。如果用 4 位表示一个象素，则有 16 级灰度或颜色；用 8 位表示一个象素，则有  $2^8 = 256$  级灰度或颜色。字符显示器只用“0”、“1”两种灰度级，就可以表示字符的有或无，这种只有两级灰度的黑白显示器称为单色显示器；具有多种灰度级的黑白显示器称为多灰度级黑白显示器；具有多种颜色的显示器称为彩色显示器。图象显示器的灰度级

一般为 64 级或 256 级。

为什么要“刷新”？什么是“刷新存储器”？

CRT 器件的发光是由电子束打在荧光粉上引起的。电子束扫描过以后，其发光亮度只能维持短暂的一个瞬间（大约几十毫秒）便消失。为了使人眼能看到稳定的图象，就必须在图象消失之前，使电子束不断地重复扫描整个屏幕。这个过程叫做“刷新”。每秒刷新的次数称为“刷新频率”。结合人的视觉生理，刷新频率应大于 30 次/秒，人眼才不会感到闪烁。显示器通常选用电视的标准，每秒刷新 50 帧图象。

为了不断提供刷新图象的信号，必须把图象存储起来，存储图象的存储器称为“刷新存储器”，也可以称“帧存储器”或“视频存储器”。

刷新存储器的存储容量由图象分辨率和灰度能决定。图象的分辨率越高、灰度能越多，刷新存储器的存储容量应该越大。

刷新存储器的存取周期必须满足刷新频率的要求。

这道题的第一个填空应选择答案 (4)，第二个填空应选择答案 (1)，第三个填空应选择答案 (3)，第四个填空应选择答案 (2)。

7. 激光印字机是激光技术和电子照相技术相结合的产物。世界上第一台激光印字机是 1975 年推出的 IBM3800，我国在 1979 年研制成功国产激光印字机。

激光印字机突出速度快，印字质量高，印字分辨率能达到 300 DPI 或 400 DPI（每英寸 400 点）。它是一种击打式硬拷贝输出设备。

这道题的第一个填空应选择答案 (2)，第二个填空应选择答案 (4)。

8. 按规定，用电设备交流输入插座是按左零（零线）右火（火线）安装连接的。按这种“左零右火”的规则连接电源线时，能使火线中的电流经过机器内部电源的保险丝管。当供电线路或计算机本身发生过流时，保险丝迅速熔断，保护计算机免受其损害。

在计算机安装完毕后，加电前应用万用表测量供电电压是否和计算机使用的交流电电压数值一致。

计算机随机所带的电源线插头上一般都有明显的标记，火线的标记为 L (Live)，零线的标记为 N (Neutral)。

这道题的答案应选择 (1)。

【第三题】略

【第四题】

1. DOS 命令可分为两大类，即内部命令和

外部命令。

外部命令：指以程序文件的形式存放在磁盘上，使用时必须从存有该命令的磁盘上把文件读入内存后方可执行的命令。而内部命令是指驻留在 DOS 内部的程序，在启动 DOS 时已经调入内存，可以随时被调用执行。

2. 堆栈是一种特殊的线性表，它采用先进后出的原则组织数据。

3. 略

4. 数据库是在文件系统的基础上发展起来的。

5. UNIX 操作系统是一种通用、交互式的分时操作系统。

6. 计算机病毒是由软件引起的。

7. dBASE III 是一种关系型数据库管理系统，用表格数据来表示实体间联系的模型叫关系模型。关系模型是数学化的，它把数据看成二维表中的元素，而这个表就是关系。

8. 反汇编是将机器语言翻译成汇编语言。汇编程序又叫汇编系统，它的功能是把汇编语言编写的源程序翻译成机器语言程序。

#### [第五题]

1. RND 函数产生的值的范围是 (0, 1)，

INT 函数把给定的参数值取整，

如：INT (5.3) = 5

INT (-5.3) = -6

而 FIX 函数只是截取整数部分如：

FIX (5.3) = 5

FIX (-5.3) = -5

所以 INT (-50+50 \* RND (0)) 的取值范围是[-50, -1]。

FIX (-50+50 \* RND (1)) 的取值范围是[-49, 0]。

正确答案只能是[-99, -1]。

2. STR \$ 函数是把数字值转化为串，而 LEN 是求串的长度，

LEN (STR \$ (-321)) 的值是4

已知 A \$ = "A123#-3456"

而 MID \$ (A \$, 6, 2) 的意思是从 A \$ 串的第六个字符开始截取 2 个字符，VAL 函数是把数字字符串转换为数值。

所以 VAL (MID \$ (A \$, 6, 2)) 的值是-3，

所以 L = 1。

3. 290° 的正弦值是大于-1 小于 0，190° 的余弦值也是大于-1 小于 0，

所以 INT (SIN (290 \* 3.14159 / 180)) 的

值只能是-1。

而 SGN (COS (190 \* 3.14159 / 180)) 的值也是-1。

所以，正确答案只能是-2。

4. 此图功能是用 X 减去 Y，然后用结果 R 与 Y 比较，如果 R 大于 Y 则把 R 的值赋给 X，然后再减去 Y，这样一直进行下去直到 R 的值小于 Y，这时循环停止，所以这正与求 X 对 Y 的模相一致，所以正确答案为 A。

5. 这三种循环是可以相互转化的，现举一实例说明：

如：10 FOR I = 1 TO 5

20 PRINT I

30 NEXT I

40 END

用当型循环可表示如下：

10 I = 1

20 WHILE I <= 5

30 PRINT I

40 I = I + 1

50 WEND

60 END

用直到型循环可表示如下：

10 I = 1

20 PRINT I

30 I = I + 1

40 IF I <= 5 THEN 20

50 END

反过来，从当型循环和直到型循环向计数循环转化及当型和直到型循环相互转化也非常容易。

#### [第六题]

1. 根据定义的函数，第 40 句中的 FNB (FNA (1)) 相当于 FNA (FNA (1)) - 1，

而 FNA (FNA (1)) - 1 = FNA (1<sup>2</sup>+2 \* 1+1) - 1

= FNA (4) - 1

= 4<sup>2</sup>+2 \* 4+1-1

= 24

最后的 Y 值只能是 25。

2. 按照程序中对 A \$, B \$ 赋值，INSTR (4, A \$, B \$) 的值应该是 6，

所以第 40 句中 Y \$ 的值是字符串 B，而函数 LEFT \$ (A \$, I) 是取字符串 A \$ 最左边的 I 个字符，函数 RIGHT 串 \$ (A \$, I) 是取字符串 A \$ 最右边的 I 个字符，所以经过第一次循环，I = 2, Y \$ = CBBAC，经过第二次循环，

I=1, Y\$ = CCBACC, 最后的答案只能是 CCBACC.

3. 此题应该注意的问题是在 FOR 循环中如果循环体内对循环控制变量不改变, 只有执行 NEXT 语时才改变控制变量, 而在此题中, 在循环体内存在对控制变量的改变语句, 所以, 一定要注意 NEXT 语句对变量的自动改变.

如: I 的初始值是 1, 执行 I=I+4 语句后 I=5, 这时 N=25, 执行 NEXT I 后, I=6, 再执行 I=I+4 时, I=10, 这时 N=25+10<sup>2</sup>=125, 再执行 NEXT I 后, I=11, 这时 I 的值已超过 10, 退出循环执行 60 号语句, 执行结果是 I=15,

最后结果是 N=125+15<sup>2</sup>=350.

4. 对于输入的任意一个数 X, Y=X<sup>2</sup>, 如果 Y>1, 那么 P=P+X=X, 也就是说 P=X (当 |X|>1); 如果 Y<=1, 那么 P=P<sup>2</sup>=X<sup>2</sup>, 也就是说 P=X<sup>2</sup> (当 |X|<1),

所以, 这个程序完成的功能是计算下列函数的值

$$P = \begin{cases} X^2 & |X| < 1 \\ X & |X| > 1 \end{cases}$$

5. 这个程序的功能是求出 3 到 20 范围的素数.

它的思想是取一个数 M, 这个数不能用 2 到 M-1 范围的整数除, 那这个数就是素数, 只要 2 到 M-1 范围内的整数有一个能整除 M, 那么 M 就不是素数.

#### 【第七题】

1. 80-110 这段程序的功能是求余数.

2. 120-150 这段程序的功能是先判断素数 R 是否为 0, 若不为 0, 那么把新求得的余数赋给 B, 前一个余数赋给 A, 然后转到 80 语句, 再求下一个余数.

3. 如果新的余数为 0, 那么前一个余 B 就是所求的结果.

【第八题】dBASE III 中用户文件有 9 类, 即数据库文件(.DBF), 记忆文件(.DBT), 索引文件(.NDX), 屏幕格式文件(.FMT), 标签文件(.LBL), 报表格式文件(.FRM), 内存变量文件(.MEM), 命令文件(.PRG), 文本文件(.TXT).

2. Config.sys 是操作系统参数设置文件, 这个文件一般定为:

FILES=20

BUFFERS=24

它表示 dBASE III 运行时要求 CCDOS 能够同时打开 20 个文件, 缓冲区为 24 个. dBASE III 技

术指标规定, 能同时打开 15 个文件. 如果 CONFIG.SYS 文件中参数设置不当, 或者启动机器时根本就没有这个文件, 那么操作系统可以打开的文件数目为 8, 而扩充的输入输出操作本身将占用 5 个文件, 留给 dBASE III 用的文件数只有 3 个, 显然这是不够的, 所以必须通过设置 CONFIG.SYS 改变 DOS 缓冲区数和允许同时打开的文件数的默认值.

另外注意此文件必须放在操作系统盘的根目录上.

3. dBASE III 有 10 个工作区, 编号 A--J, (1--10), 在每个工作区中可打开一个 .DBF, 一个格式文件, 七个索引文件, 各个工作区互不影响, 刚进入时默认为一号.

4. dBASE III 的字段类型有五种, 分别是数值型, 日期型, 记忆型, 字符型, 逻辑型.

#### 【第九题】

(1). USE GZ 后文件指针指在第一号记录, 所以 EDIT 编辑 1 号记录.

(2) 打开 GZ 后, GOTO 3 指针定位在 3 号记录, 但最后执行 APPEND BLANK 后指针指到文件尾并且加 1, 为当前指针值, 即 7.

(3) USE GZ

REPLACE 工资 WITH 工资+20 FOR 工龄 > 15

满足条件工龄 > 15 的记录有 3 号, 6 号记录.

(4) 打开 GZ 后, 执行 LOCATE 语句, 找到满足条件: = "高工" 且工龄 > 15 的 3 号记录, 执行 CONTINUE 语句后指针指向下一个满足条件的记录, 即 6 号记录.

(5) 打开 GZ 后, GOTO 3 指针定位于 3 号记录, 执行 COUNT 语句, 它对 3 到 6 号记录中满足工龄 < 20 或姓名 = "严六" 的记录计数, 其中共有 4 号, 5 号, 6 号 3 个记录满足条件.

(6) 打开 GZ 后, 在 1 号到 5 号记录中选取职称 = "工程师" 或工龄大于 10 的人, 拷贝到新文件 GZ1 中, 共有 2 号、3 号两个记录, 马上打开 GZ1, 对工资项求和, 结果为 320.

(7) 逻辑顺序的含义为按编号后的记录顺序, 排序后编号顺序应为 01, 02, 03, 04, 05, 06, 其对应的记录顺序为 1, 5, 3, 2, 6, 4.

(8) 此程序的功能是对工资大于 150 的人计数, 结果放于 A 中并对其工资求和, 结果放于 C 中. 把工资小于 150 的人计数, 结果放于 B 中并对其工资求和, 结果于 D 中.



# 单片机实验与 BJS-51 实验教程

北京广播电视大学 李广弟

学习单片机是为了使用单片机,因此单片机这门课程的实践性极强,必须在教学活动中加强实验这一理论联系实际的重要环节。

通过实验可以使学生巩固和深化对课程内容的理解,从而把课堂学习引向深入具体;学到单片机系统电路设计及硬件连接的应用技术;掌握单片机汇编语言程序的编写、调试及运行方法;培养学生的独立分析和解决问题的能力以及严谨的科学作风;为今后单片机的应用开发打下基础。

为此,我们在研制 BJS—51 教学实验系统的同时,即着手实验教程的编写工作,以便于配套使用。

本实验教程共包括两部分。其中第一部分为基础实验,可在本系统的最小配置和基本配置上完成。第二部分为扩展实验,要与专用的扩展板配合实现。

基础实验部分共编写约一百个实验。其中包括: MCS—51 汇编语言程序设计实验、I/O 接口实验、LED 显示实验、时钟实验、音乐与声响实验、顺序控制实验、交通灯控制实验、串行通信实验等共八大类。这些实验复盖了单片机应用技术的主要内容,通过实验使学生在汇编语言程序设计、并串行数据传送、显示、中断、查询、控制等基本技术方面得到全面的训练。

本实验教程有如下特点:

1. 适用范围宽广。目前单片机教学活动分布范围很宽,包括从职业高中到研究生的多种层次;在普通教育之外,还有成人教育,以及在职技术干部的培训等。为了满足大范围多层次的教学实验需要,BJS—51 在硬件方面搞了结构的模块化,允许用户根据需要灵活配置系统;在软件方面则通过增加实验题目的数量,并把实验题目分类排档,使用户可根据不同层次的教学要求从中选作,以求扩大其适用范围。

2. 在实验内容上力求复盖单片机应用技术的各个方面。例如:汇编语言程序设计技术、系统扩展技术、并串行输入输出技术、显示技术、查询技术、中断技术、数据转换技术以及控制技术等。在实验方法和要求上,强调实用性,充分作到理论联系实际,以加强学生动手能力和独立解决问题能力的培养。

3. 采用绕接方法,确保实验电路的连接牢固可靠。经验表明,在传统的面包板使用的插接连线方式,时间长了簧片变松,接触不良,常引起实验电路工作不稳定,造成调试困难。这是实验者和指导者均感头痛的问题。为此,本教学实验系统在连接实验电路时,采用绕接连线方式。由于备有专用的绕线工具,因此使用起来方便快捷,而且连线牢固可靠。从根本上解决了实验中

连线接触不良的问题。

本实验教程可直接作为实验指导书使用,教程中对每一个实验都从题目、目的、电路连接、算法、程序流程、参考程序及思考题等几个方面进行说明。这样既可以减轻教师指导实验的负担,也可以减少学生在准备实验上所花的时间,保证实验得以顺利进行。

对于 40~50 学时的单片机课程,以安排 24 小时实验比较合适。每个实验三小时,共作八个实验。如学生的程度较低,可安排四次汇编语言程序设计实验,然后再安排 I/O 接口、LED 显示、时钟和交通灯控制实验各一次。

## 典型实验介绍

下面列出两个典型实验,供读者了解 BJS—51 实验教程的实验指导方法。

### 1. 数的排序实验

汇编语言程序设计是单片机实验的基本内容,因此这部分实验在教程中占较大份量。通过这些实验可以使使学生更加熟悉 MCS—51 的常用指令及寄存器的功能和使用,学会汇编语言程序设计的基本方法,掌握典型程序的算法和技巧,为今后的单片机应用开发打下基础。

下面以单字节无符号数的排序为例,说明有关汇编语言程序设计类实验的基本模式。

#### ① 实验题目

假定有八个单字节无符号数存放在 RAM 中以 20H 为首地址的连续单元中,请进行升序排序。

#### ② 实验步骤和内容

编写、输入、调试并运行程序。

检查程序运行结果,看开序排序是否正确实现。

#### ③ 算法说明

数据排序的算法很多,本实验以冒泡法实现。

冒泡法是一种相邻数互换的排序方法,因类似水中汽泡上浮,故称冒泡法。执行时从前向后进行相邻数比较,当发现数的次序与排序要求的递升顺序不符时,就将这两个数进行交换。通过这种相邻数互换方法,使小数向前移,大数向后移。如此从前向后进行一次,就会把最大数排在最后,第二次时,就会把次大数排在倒数第二位置……

例如原始数组为 50, 38, 7, 13, 59, 44, 78, 22 第一次冒泡过程是:

```
50, 38, 7, 13, 59, 44, 78, 22 (交换)
38, 50, 7, 13, 59, 44, 78, 22 (交换)
38, 7, 50, 13, 59, 44, 78, 22 (交换)
```

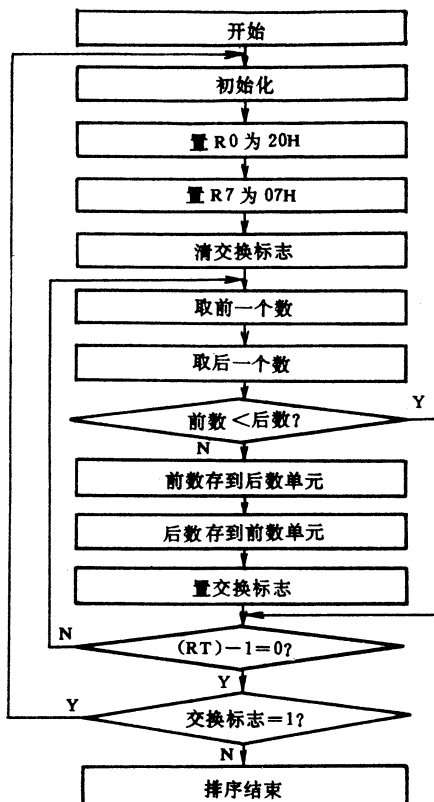
38,7,13,50,59,44,78,22 (不交换)  
 38,7,13,50,59,44,78,22 (交换)  
 38,7,13,50,44,59,78,22 (不交换)  
 38,7,13,50,44,59,22,78 (第一次冒泡结束)

如此进行,各次冒泡的结果是:

第一次 38,7,13,50,44,59,22,78  
 第二次 7,13,38,44,50,22,59,78  
 第三次 7,13,38,44,22,50,59,78  
 第四次 7,13,38,22,44,50,59,78  
 第五次 7,13,22,38,44,50,59,78  
 第六次 7,13,22,38,44,50,59,78

假定有  $n$  个数,从上述冒泡过程可以看出,第一次需比较  $(n-1)$  次以后各趟的比较次数应该是递减。但为了简化程序,各次一律比较  $(n-1)$  次。

对于  $n$  个数,冒泡排序最多需进行  $(n-1)$  次才能完成排序。但实际上,可能不到  $(n-1)$  次数即已排好序,如本例共八个数,按说应进行七次,但实际进行到第五次时排序就完成了。判定排序是否完成的最简单方法是看各次排序中是否有数交换发生,如果有数交换,就说明还没排好序,否则就表示排序已经完成。为此,控制排序结束可不使用计数方法而使用交换标志的判定方法。



#### ④参考程序流程

根据上述算法分析,本程序定义如下参数:

$R_0$  为数据区首地址,初始值为  $R_0=20H$

$R_7$  为各趟比较次数计数器,初始值为  $R_7=07H$

$TR_0$  为比较过程中的交换标志

$TR_0=0$  没交换

$TR_0=1$  有交换

#### ⑤实验参考程序

8000	7820	SORT	MOV R0,#20H	
8002	7F07		MOV R7,#07H	
8004	C28C		CLR TR0	清交换标志
8006	E6	LOOP:	MOV A,@R0	取前数
8007	F52B		MOV 2BH,A	存前数
8009	08		INC R0	
800A	862A		MOV 2AH,@R0	取后数
800C	C3		CLR C	
800D	96		SUBB A,@R0	
800E	4008		JC NEXT	前数小于后数则转移
8010	A62B		MOV @R0,2BH	大数放在后边
8012	18		DEC R0	
8013	A62A		MOV @R0,2AH	二个数交换位置
8015	08		INC R0	
8016	D28C		SETB TR0	
8018	DFEC	NEXT:	DJNZ R7,LOOP	
801A	208CE2		JB TR0,SORT	
801D	020000		AJMP [0000]	返回监控状态

#### ⑥实验参考数据

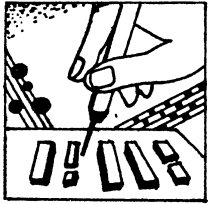
内存单元 排序前 排序后	(20H)	(21H)	(22H)	(23H)	(24H)	(25H)	(26H)	(27H)
	0A	AA	BB	7F	6E	47	64	30
	0A	30	47	64	6E	7F	AA	BB

(未完待续)

(接第 24 页)

突,那么在子程序中还要对 X、Y、P 的状态加以保护和恢复。因此,转入子程序时应当首先分析一下是否需要保护现场,若需要,那么应注意是哪几个寄存器需要保护,以及在什么时候退回保护。只有这样仔细的加以分析和妥善的处理,才能使程序得以正确执行。本题只有 A 的情况,没有 X、Y、P 的情况。

关于堆栈程序的设计,我们就介绍到这里。总之,要掌握堆栈的概念、结构、先进后出原则;掌握断点概念、保护和恢复现场的方法;掌握子程序和堆栈之间的关系;掌握 A、X、Y、P 进栈出栈的指令和方法。这是学习本节的要点。



# 学装微电脑

## 自动输送装置

易齐干

本文介绍冲压件的自动输送装置,讨论极限开关信息输入与控制。冲压件尺寸 $5 \times 15 \times 40\text{mm}$ ,材料为薄钢板。冲压机一次冲压成型,连续工作,冲压件经过送料器,数量1000个为单位装入成品货架。货架为六层,冲压件被放置在规定的位置。自动输送装置原理如图1所示。工作过程如图2所示。

由图1看出,要输入微电脑14个信息。其中有检测输送装置工作状态的13个极限开关,具体是LS1~LS13。另一个是向货架放置1000个冲压件的计数器。

接收微电脑输出信息的单元有10个。驱动送料器工作的继电器R1,驱动气缸工作的继电器R2~R10共9个。

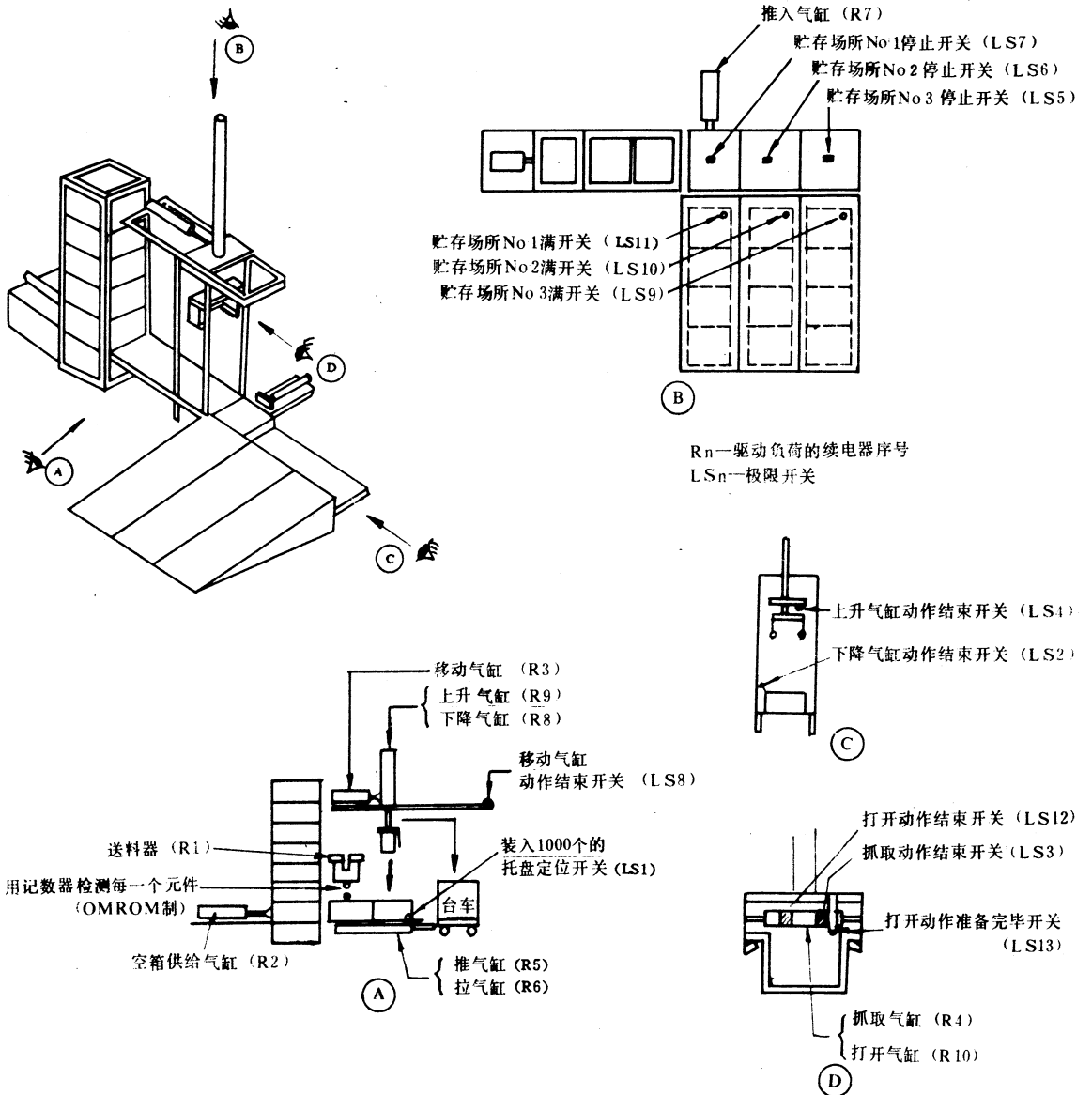


图 1



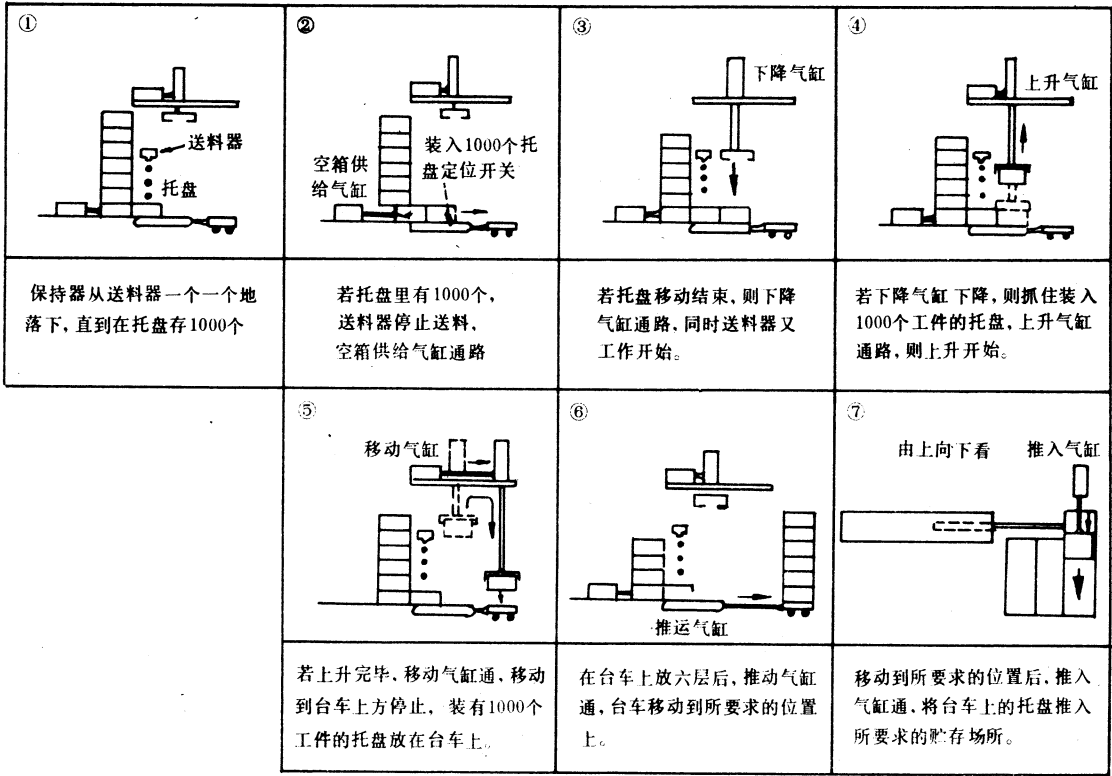


图 2

微电脑输入输出的划分如图 3 所示。

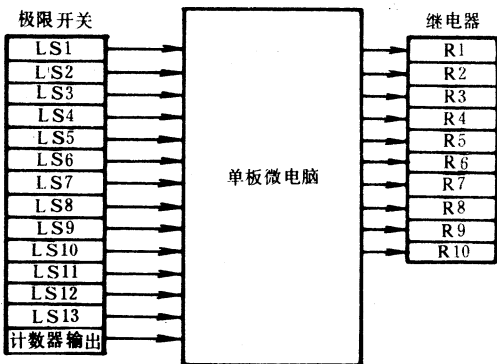


图 3

硬件原理图如图 4 所示。输入给微电脑的信息有前述的 14 个之外，再加上手动操作 8 个按键开关共计 22 个信息，采用开关动态扫描方式仅使用 8255A 的端口 C 低 4 位和端口 A (8 位) 合计 12 位即可接收 22 个信息。开关动态扫描方式如图 4，图 5 为开关动态扫描方式连接。

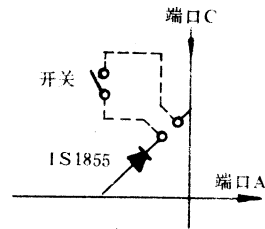


图 5

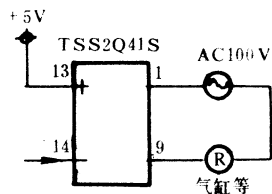


图 6

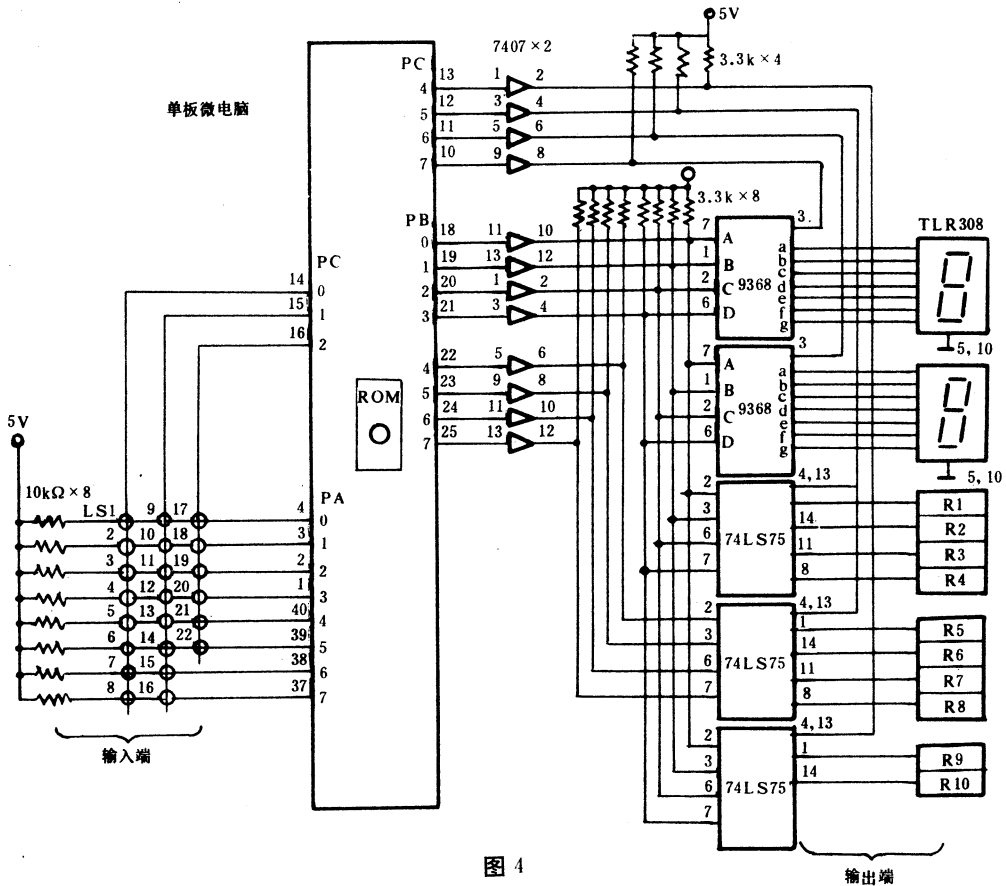
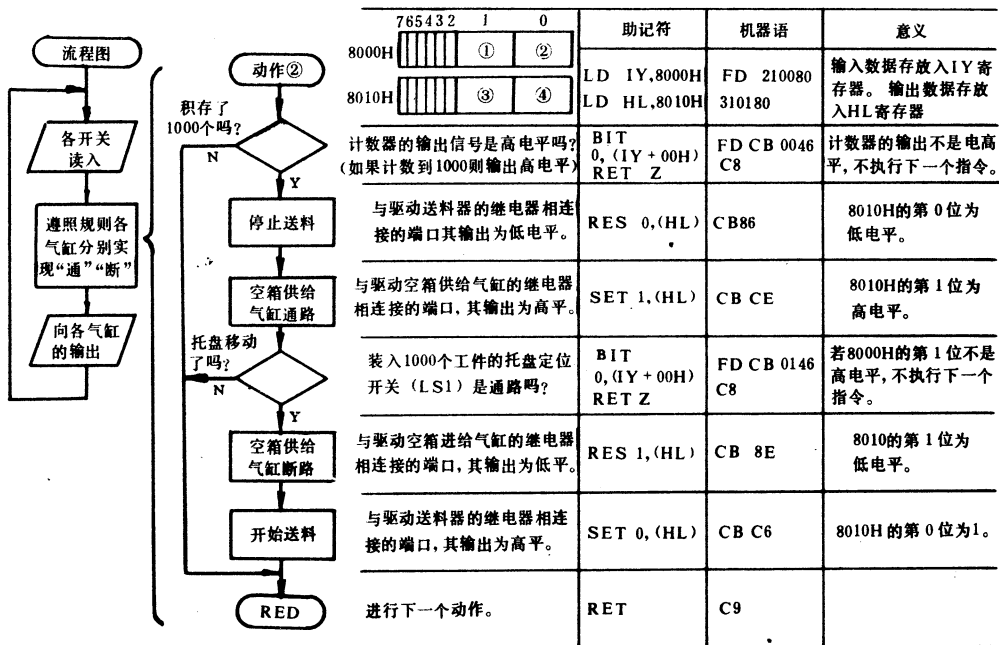


图 4



① 托盘定位开关 ② 来自计数器的输入 ③ 继电器驱动供给气缸 ④ 继电器驱动送料器

图 7



图 8 输入输出数据分配

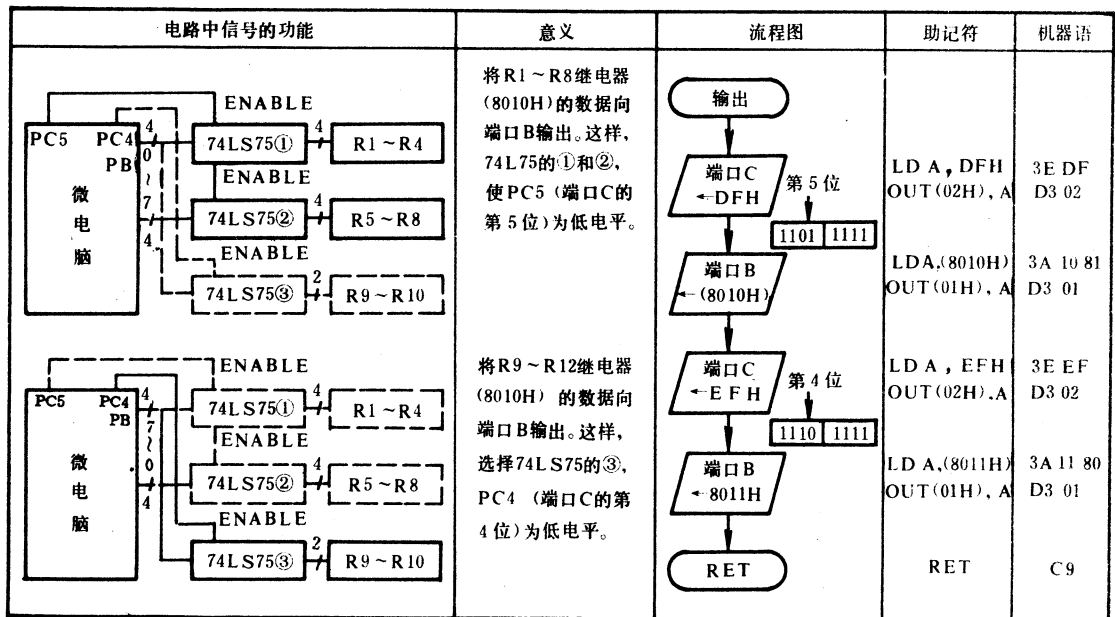


图 9 输出(8010H)、(8011H)的程序

开关动态扫描方式本刊91年曾讨论过,请读者查阅。

微电脑输出信息除前述的10个之外,再加上两个7段数码管。分别显示成品货架数目和放置层数。8255A的端口C高位和端口B共12位为输出端口,显然位数不够。使用74LS75双稳态锁存器扩展输出端口。R1~R10继电器可考虑采用半导体开关,避免产生噪声使微电脑误动作。连接方式如图6所示。9368IC是7段数码管驱动器。7407IC是反相器。

自动输送装置的软件工作方式是前述的极限开关LS和手动按键向微电脑输入ON或OFF信息,与储存在微电脑内的工作程序进行运算比较。输出与工作程序符合的指令,驱动R1~R10继电器工作。

例如,图2中动作②的程序如图7所示。首先,询问计数器“成品有1000个吗?”如果答案“是”,则发出“停止送料”、“供给空箱气缸ON”指令。接着询问“装入1000个工件的托盘定位开关LS1为ON吗?”如果答案“是”,则发出“供给缸OFF”,“送料器为ON”指令。

输入输出的信息预先分配在存储器的8000H~8011H地址中,如图8所示。端口B向继电器R1~R12输出8010H地址8011H地址内容的程序如图9

自动输送装置这个事例,对初学者学习微电脑应用控制很有借鉴,有兴趣的读者不妨将它应用于您的工作中去。



# 计算机语音输出功能的 开发与应用(上)

马钢南山铁矿技术科(243033) 陈竹林

一个优秀的计算机软件,首先应有一个良好的人机界面,这对方便操作、提高软件的使用效果和软件的生命力是至关重要的。然而,目前人们在人机界面上作的工作大都局限于菜单、图像与音乐的设计上,很少涉及语音的输出。其主要原因是,大多数计算机本身不支持语音输出的功能。由于语音输出在数据及文稿的校对、软件介绍、操作说明和告警等方面有其独特的作用,与显示配合可达到声形并茂的效果,因而越来越为人们所重视。

随着近几年语音处理技术的发展,一些大规模语音处理集成电路相继问世。如 UM5100/5101、T6668 及 TMS5220 等,有些可与计算机接口,对语音信号进行采集、分析、处理与合成,实现高质量的语音输出。采用语音处理芯片研制的语音输出卡也已上市,如中国科学院声学研究所用 TMS5220 芯片开发的语音输出卡,可直接插在 IBM 系列及其兼容计算机内,配以相应的驱动软件,完成标准的汉语语音普通话输出。

虽然语音输出卡能达到很好的语音输出效果,但由于必须增加硬件配置,使应用软件的通用性受到限制,很难在通用性较强的软件(如某些工具软件、管理软件等)上采用。可否在不增加硬件配置的前提下,用软件驱动计算机中的扬声器产生语音输出呢?回答应该是肯定的。问题的关键是如何获得语音数据和如何编制驱动软件。目前人们已经在 APPLE-Ⅱ 系列计算机上取得了初步的成功。由于这类计算机运行速度慢、内存与外存容量较小,只能完成简单的语音输出,实用价值并不大。本文将从语音信号的采集、语音数据的压缩处理和语音的合成等方面介绍在 IBM 系列及其兼容计算机上开发语音输出功能的方法,并介绍一个简单使用例子,旨在探讨一种简单易行、能被人们接受和采用,在使用广泛的微型计算机上普及语音输出功能的途径和方法。

## 一、语音采集方案的选择

语音数据是计算机完成语音输出的必要条件之一,通常是通过对人的说话声音进行转换、采集,并经过一定分析处理获得的。声音包含有音调、音色和响度三种特征。其中,音调反映声音的基本频率,在声音波形中,表现为单位时间内过零次数的多少;音色反映声音的高次谐波特性,决定了声音波形的复杂程度;响度反映声音的强弱变化,在声音波形中,表现为幅度(或包络)的变化。虽然三种特征都采集的方案能不失真的合成高质量的声音信号,但采集与合成较为复杂,一般需要增加语音输出的硬件配置,我们不易采用。根据声音信号的特点,忽略音色与响度,只采集

音调信息,合成的语音也能满足人们听懂的要求。采用这一方案可大大简化语音数据采集、数据处理及扬声器驱动软件复杂程度,采集装置也比较简单,使开发与使用的难度大为降低。

## 二、采集装置与原理

采集装置是一种能将人的话音转换成语音数据送往计算机的硬件设备。由于我们只采集语音信号的过零信息,这种装置十分简单,仅有一个话筒、三只晶体管 and 几个阻容元件组成(如图①所示),完全可以自制。我们选择 RS-232 串行接口作为语音数据进入计算机的通道,并利用两个引脚的高、低电平对外接采集装置供电。具体讲,用 RS-232 的载波检测信号线(CD 脚)作语音数据输入端,用数据终端就绪信号线(DTR 脚)和请求发送信号线(RTS 脚)分别给外接采集装置提供正负电源。采集装置与 RS-232 口的联接见图 1。

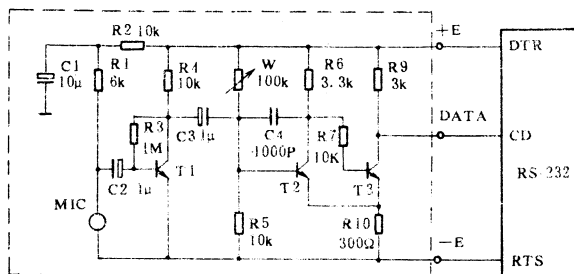


图 1 采集装置及其与计算机的联接

下面介绍其工作原理,用软件向 3FCH 端口(modem 控制寄存器)送 01H 使数据终端就绪端 DTR = 1, 请求发送端 RTS = 0, 此时采集装置的 +E 端和 -E 端分别得到约为 +9V 和 -9V 的电源,采集装置开始工作。在无声时,晶体管 T1 无信号输出, T2 与 T3 构成的施密特触发器电路输出低电平(T2 截止, T3 导通, DATA ≈ -7V)。当我们对着话筒讲话时,声音经过话筒 MIC 转换成电信号,经晶体管 T1 放大和电容 C3 耦合到施密特触发器的输入端,在其输出端 DATA 输出经过整形的语音数据。从图 2 中我们可以看出,采集装置输出的 DATA 只反映了语音信号的过零特征,没有音色变化,幅度也是固定不变的。最后只要用软件循环不断查询 3FEH 端口的 D7 位,并记录其变化情况,即可完成采集过程,得到相应的语音数据。查询的速率应大于最高语音频率的两倍,语音信号的频率一般在 300~3400Hz 之间,因此采样速率最好不要小于

8KHz。采集速率的控制可通过改变软件的延时来实现。

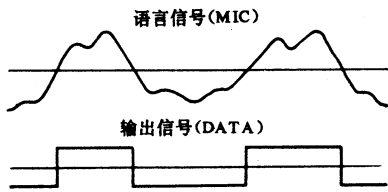


图2 采集装置的输出波形

### 三、扬声器的发声驱动

IBM 及其兼容计算机主机内部均有一个扬声器，一般用作告警和音乐输出。为了直接用这个扬声器输出语音，我们先以 IBM-PC/XT 计算机为例介绍一下扬声器接口电路和发声原理，如图 3。

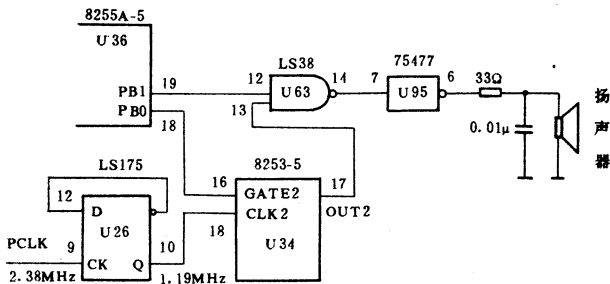


图3 IBM-PC/XT 扬声器接口电路原理图

来自计算机 2.38MHz 的时钟信号 PCLK 经 U26 两分频后，从 U26 的第 10 脚输出 1.19MHz 的计数脉冲，并送 U34 的时钟输入 CLK2 端。U34 是一个可编程计数/计时器，在此设定为方波发生器，其 17 脚 OUT2 的输出频率取决于 U34 内部寄存器的数值，数值越小频率越高，数值越大频率越低，该数值可随时由软件设置。改变这个数值就改变了扬声器的发声频率。U36 是一个通用可编程输入/输出控制器，其端口 B 的第 0~1 位用来控制扬声器的发声与否。其中，PB0 控制计数器 U34，当 PB0=1 时，U34 的 GATE2=1，计数器工作，OUT2 有音频输出。当 PB0=0 时，GATE2=0，计数器停止工作，OUT2 恒为 1。PB1 通过与非门 U63 控制发声的与否，当 PB1=0 时，U63 关闭，14 脚输出恒为 1，使 U95 脚恒为 0，扬声器断电不发声。当 PB1=1 时，U63 开通，如果此时 PB0=0，U63 的 12、13 脚均为 1，输出为 0，使 U95 输出恒为 1，扬声器处在通电状态，如果此时 PB0=1，扬声器将按照 U34 的 17 脚信号频率不断地通、断电，产生固定频率的音响。查手册可知，U36 端口 B 的口地址是 61H。

了解以上原理后，我们将端口 61H 的 D<sub>0</sub> 位置 0，D<sub>2</sub>~D<sub>7</sub> 位保持原状态，并用语音数据控制 D<sub>1</sub> 位状态的变化，从而使流过扬声器的电流按照语音数据的规律变化，就能产生相应的语音输出。

程序一是一个语音采集与发声试验程序，它将每次采集到的语音数据直接送往端口 61H，驱动扬声器发声。运行该程序，在我们对着话筒讲话的同时，就能实时地听到扬声器发出的声音，好象计算机就是一台扩音机。

#### 程序一：语音采集与发声试验程序

```

1: CLI
2: MOV DX,03FCH
3: MOV AL,01H
4: OUT DX,AL
5:CJ: MOV DX,03FEH
6: IN AL,DX
7: AND AL,80H
8: SUB AL,80H
9: JZ KS
10: IN AL,61H
11: AND AL,FDH
12: OUT 61H,AL
13: JMP CJ
14:KS: IN AL,61H
15: AND AL,FEH
16: OR AL,02H
17: OUT 61H,AL
18: JMP CJ

```

程序中第 2~4 句完成对外接采集装置的供电。第 5~6 句采集一次语音数据。第 7~8 句判断语音数据的状态。如果数据是 0，执行第 10~13 句切断扬声器的电流，然后返回第 5 句作下一次采集，如果数据为 1，跳到第 14~18 句执行，接通扬声器的电流，再作下一次采集。这个程序在调试采集装置时，是非常有用的。

### 四、声音数据的压缩存储

我们采集语音数据的目的是要建立一个语音数据库，以文件的形式存在磁盘上，供需要语音输出的软件调用，实现语音输出的功能。对语音数据进行适当的压缩处理，对缩短语音数据文件的长度，减少在磁盘上占用的空间是十分必要的。

语音频率有高低，通常在 300~3400Hz 之间。采集高频率语音时，语音数据中连续出现 0 或 1 的次数较少，而采集低频率语音时，语音数据中连续出现 0 或 1 的次数就比较多。基于这一特点，可对低频语音数据进行压缩处理。

在作语音采集的同时，对连续出现 0 或 1 的次数进行统计和判断，小于等于 7 次时，形成高频字节，大于 7 次时形成低频字节。高频字节的 D<sub>7</sub> 位恒为 0，D<sub>6</sub>~D<sub>0</sub> 位分别是连续 7 次的采样值。如“01011001”表示第 1,3,4,7 次采样值为 1，第 2,5,6 次采样值为 0。低频字节的 D<sub>7</sub> 位为 1，当 D<sub>6</sub> 位为 0 或 1 时，D<sub>5</sub>~D<sub>0</sub> 位分别表示连续 0 或 1 的个数。如“10001001”表示连续 9 次采样值为 0，“1100111”表示连续 15 次采样值为 1。

设采样频率为 F<sub>c</sub>，语音频率为 F，则可采语音的最高频率 F<sub>m</sub> 应小于 F<sub>c</sub>/2，高频字节对应的语音频率

范围 $\Delta F_h = F_c/16 \sim F_c/2$ , 单位时间占用的字节数  $N_h = F_c/7$ 。低频字节对应的语音频率范围 $\Delta F_l = F_c/128 \sim F_c/16$ , 单位时间占用的字节数  $N_l = 2F$ 。

例如, 当采样频率  $F_c = 8\text{KHz}$  时, 可采语音最高频率  $F_m = 4\text{KHz}$ 。高频字节对应语音频率范围 $\Delta N_h = 0.5 \sim 4\text{KHz}$ , 单位时间占用字节数,  $H_h = 1.14\text{KB/S}$ 。低频字节对应语音的频率范围是 $\Delta F_l = 62.5 \sim 500\text{Hz}$ , 单位时间占用字节数  $N_l = 125 \sim 1000\text{B/S}$ 。存储一分钟的

语音数据, 全部是高频字节时, 占用 68KB 的字节空间, 全部是低频字节时, 占用的字节数随语音频率的降低而减少, 最多时占用 60KB, 最少时占用 7.5KB。也就是说, 语音频率越低, 数据的压缩效果越明显。尤其在语音停顿期间, 采集装置的输出 DATA 恒为 0, 数据压缩比最高。

(待续)

## 怎样使 CCDOS2.10 的 9 针打印驱动程序适应 2.13H 汉字系统

湖南岳阳市 6906 工厂 (414007) 李修连

2.13H 有丰富的汉字处理功能, 但它没有提供 9 针 (包括 12 针) 打印驱动程序。而目前我国仍有一些用户使用 9 针打印机。还有些用户有可利用的 9 针打印机。为了能够把这些 9 针打印机利用起来, 我们修改 CCDOS2.10 的 9 针打印驱动程序 ALL9P.EXE 和 NEW9P.EXE, 使之适应 2.13H 汉字系统, 取得了很好的效果。

ALL9P.EXE 中, 从 1FD0H (在 DEBUG 下) 开始的三条指令取得汉字字模的段址, 入口 AX 中存放汉字机内码, 出口 DX 中存放字模的段址 (偏移量为 0)。

2.13H 汉字系统中由中断 7F 取得汉字字模的段址, 入口参数 DX 为汉字机内码, 出口参数 DX 为字模的段址。因此我们只要修改 ALL9P.EXE 中取汉字字模段址, 三条指令就可以使它适用于 2.13H 汉字系统修改过程如下:

```
A>REN ALL9P.EXE 9P
A>DEBUG 9P
-A1FD0
```

```
MOV DX,AX
INT 7F
NOP
NOP
NOP
NOP
NOP
-W
-Q
A>REN 9P ALL9P213.EXE
```

NEW9P.EXE 中, 从 1FE0H 开始的三条指令取汉字字模的段址, 其入口参数、出口参数同 ALL9P.EXE 一样。因此, 除地址不同外, 修改 NEW9P.EXE 的方法与修改 ALL9P.EXE 基本相同。

修改后的驱动程序可直接安装, 不需运行 FILE16B.COM。

注: 本稿是用修改后的 NEW9P.EXE 在 2.13H 下打印的。

## 8031 真的无 ROM 吗?

北京理工大学三系 90 研 (100081) 肖革文

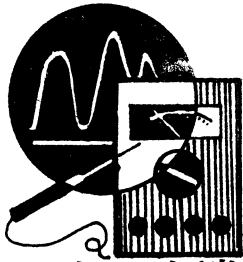
现在出版的许多单片机书籍中都说 MCS-51 系列中的 8031 内部无 ROM, 因而必须外接 EPROM, 实际上, 8031 内是有 ROM 的, 而且 ROM 中有内容。这一事实, 作者在江苏启东计算机厂生产的 DVCC-51 开发机上, 读出了 8031 的内容, 并在终端屏幕上显示程序机器码, 且可反汇编显示, 亦可用打印机把屏幕内容打印出来。

读出 8031 内容的方法:

在 DVCC-51-ED 开发机上, 将 8031 芯片插在

EPROM 固化区 8751 的位置上, 用键盘键入 0000 F1 1000 F20 EPMOV, 即把 8031 内容读到 000H 内存。然后用反汇编命令显示或打印, 并可通过 IBM-PC 及其兼容机存盘。

由此可见, 8031 内部是有 ROM 的, 只不过是用户不能使用而已, 其容量也为 4KB。由此推论: 8032, 8344, 8035, 8039, 8096 内也有 ROM, 且 ROM 中也有内容, 通过分析其中的内容, 我们可以学到国外编程的技巧, 也可以利用其中的一些子程序为我所用。



## 维修经验谈

# IBM-PC/XT 及其兼容机 RAM 故障的检修方法

黑龙江绥化师范专科学校(152061) 齐吉泰

IBM-PC/XT 及其兼容机的随机存储器 RAM 芯片损坏所出现的故障现象主要有下面三种:

- 1、自检过程中显示错误代码。
- 2、自检过程中不显示错误代码,也能够引导操作系统,但所显示的存储容量与实际不符。
- 3、开机后无法完成自检和系统初始化,不显示光标和任何信息。

对于上述由 RAM 芯片损坏而出现的故障,可根据故障现象分别采用下面的方法进行检修。

### 一、显示错误代码时 RAM 故障的检修

多数 PC/XT 及其兼容机,除第一排 RAM 外,其它各排中的任意一只 RAM 芯片损坏时,机器在自检过程中都会给出如下信息:

```

XXX KB OK
XXXXXX XX 201
ERROR (RESUME=F1 KEY)

```

其中第一行数字给出了通过自检的内存容量;第二行的第一位数字表示损坏的 RAM 芯片所在的 BANK,即某排 RAM;第六、七两位数字表示损坏的 RAM 芯片在 BANK 中的 Bit,即具体位置;201 表示故障在 RAM 电路中。

PC/XT 及其兼容机的随机存储器 RAM 由于内存容量的差异和采用不同的 RAM 芯片,使其在机器中的位置和排列次序也不尽相同。现以扩充到 640KB 的机型为例,如果内存全部采用 64K×1 位的 RAM 芯片,则系统板上安装的存储容量为 256KB,分为四个模块,如图 1 所示。剩下的 384KB 安装在内存扩充板上,分为六个模块,如图 2 所示。640KB 的存储容量共分为十个模块(BANK),每个模块共有九块 RAM 芯片,各模块所对应的错误代码见表一,每个模块中的各个 RAM 芯片所对应的错误代码见表二。明确了这个排位之后,根据屏幕上显示的错误代码,就可以很快地知道损坏的 RAM 芯片在哪个模块中的哪一位。例如自检

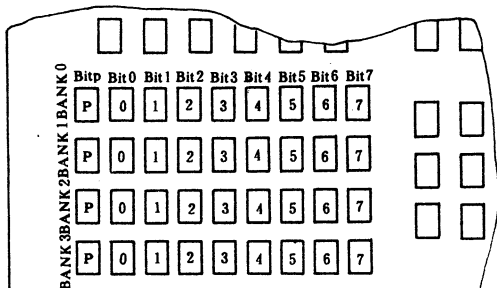


图 1 系统板

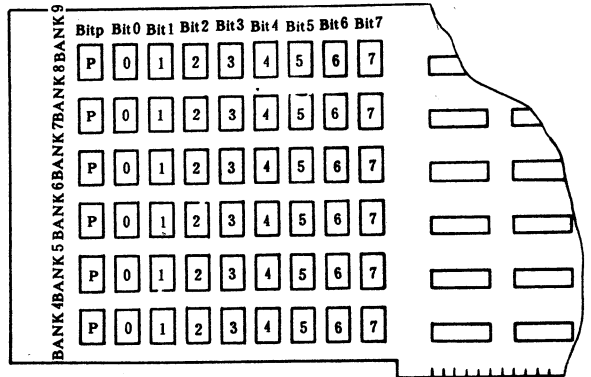


图 2 扩充板

表一

板名	模块	错误代码
系统板	BANK 0	00000
	BANK 1	10000
	BANK 2	20000
	BANK 3	30000
扩充板	BANK 4	40000
	BANK 5	50000
	BANK 6	60000
	BANK 7	70000
	BANK 8	80000
	BANK 9	90000

表二

位置	错误代码
Bit p	00
Bit 0	01
Bit 1	02
Bit 2	04
Bit 3	08
Bit 4	10
Bit 5	20
Bit 6	40
Bit 7	80

时显示的错误代码是 10000 00 201,则由 201 可以断定故障存 RAM 电路中,由 10000 可以断定故障存系统板上的 BANK1 中,由 00 可以断定故障在 BANK1 中的 BitP,即第二排 RAM 中的第一只 RAM 芯片。如果自检时显示的错误代码是 90000 80 201,则查表可知损坏的 RAM 芯片在扩充板上 BANK9 中的 Bit7。

如果内存采用的是 256×1 位和 64K×1 位两种 RAM 芯片,则 640KB 的存储容量全部安装存系统板上,也分为四个模块,排列次序和图 1 所示的相同,所不同的是前两排采用了 256×1 位的 RAM 芯片,使每排的存储容量增大到 256KB,后两排仍然使用 64K×1 位的 RAM 芯片,每排的存储容量还是 64KB。这种机型内存的各模块所对应的错误代码见表三。

表三

模 块	BANK0	BANK1	BANK2	BANK3
错误代码	00000	40000	80000	90000

每个模块中的各个 RAM 芯片对应的错误代码和表二所列的相同。如果自检时显示的错误代码为 40000 00 201,则损坏的 RAM 芯片的位置在 BANK1 中的 BitP,即第二排的第一位 RAM 芯片;如果出现错误代码为 90000 80 201,则损坏的 RAM 芯片的位置在 BANK3 中的 Bit7。

根据上述方法一般都能比较准确地确定出损坏的 RAM 芯片的位置。更换 RAM 芯片,故障即可排除。对于其它类型的微机。根据显示的代码和 RAM 芯片的容量及排列情况,参照上述方法也能够比较准确地确定出损坏的 RAM 芯片的位置。

**二、不显示错误代码时 RAM 故障的检修**

有一些 PC/XT 及其兼容机,当随机存储器中某一 RAM 芯片损坏时,只要损坏的芯片的位置不在 BANK0 或其它各模块的 P 位,则开机后能够自检和初始化,虽然自检时屏幕上显示的存储容量与实际安装的不符,但不显示错误代码。例如,一台内存为 640KB 的 PC/XT 微机,开机后自检到 512KB 时转去引导操作系统,内存“丢失”128KB,这种内存“丢失”的故障,由于不显示错误代码,一般无法直接确定损坏的 RAM 芯片所在的 BANK 和 Bit。对于这种类型的微机,如果有高级诊断盘,可以利用诊断程序确定损坏的 RAM 芯片的位置。利用高级诊断盘诊断 RAM 故障时,显示的错误代码和自检时出现的错误代码是相同的,因此确定损坏芯片位置的方法也相同。高级诊断盘的使用方法,有关资料已有详细的介绍,这里不再赘述。

在没有高级诊断盘的情况下,为了避免盲目地更换 RAM 芯片,首先应确定损坏的 RAM 芯片所在的 BANK。确定故障所在的 BANK,可以根据自检时屏幕上显示通过的内存容量和所使用 RAM 芯片的存储容量来确定。例如,通过自检的内存容量为 512KB,对于全部使用 64K×1 位 RAM 芯片的微机,因为每个模块的存储容量都是 64KB,自检通过了 512KB/64KB=8 个模块,故此可以断定故障在扩充板上第九个模块,即 BANK8。而对于使用 256K×1 位和 64K×1 位两种芯片的微机,因前两个模块中使用的是 256K×1 位的芯片,则自检通过了 512KB/256KB=2 个模块,故此可以断定故障在第三个模块中,即 BANK2。

由此可见,当通过自检的存储容量中含有 N 个模块时,故障在 N+1 个模块中。

在确定出损坏的 RAM 芯片所在的 BANK 后,逐个或全部更换该模块的 RAM 芯片,故障即可排除。这样虽然不能直接确定损坏芯片的准确位置,但可以缩小故障范围,避免逐行更换。

**三、第一排(BANK0)RAM 故障的检修**

开机后无法完成自检和系统初始化,不显示光标和任何信息,使整个系统“挂起”的故障现象多为第一排(BANK0)RAM 损坏而造成的,由于屏幕上不显示任

何信息,因此无法确定损坏的 RAM 芯片的位置。虽然可以将第一排 RAM 芯片个更换,但多数 PC/XT 及其兼容机的第一排 RAM 芯片是直接焊在印刷电路板上的,更换时要一只一只地烫下来试探着更换。为了找到一只损坏的芯片要烫下好几只,这样即费工又容易损坏印刷线路板。下面介绍一种既能判断上述故障是否在 RAM 又能给出第一排(BANK0)中损坏芯片位置的检修方法。

由于系统板上四排 RAM 的数据线和地址线都是公用“总线”,各排 RAM 所不同的只是  $\overline{RAS}$  和  $\overline{CAS}$  信号。第一排(BANK0)使用的是  $\overline{RAS0}$ 、 $\overline{CAS0}$  信号,第二排(BANK1)使用的是  $\overline{RAS1}$ 、 $\overline{CAS1}$  信号,其余类推。 $\overline{RAS}$  和  $\overline{CAS}$  信号是由系统板上的 U42、U56(不同的 PC/XT 机标号有所不同)两个三-八译码器译码后提供的,如果将 BANK0 和 BANK1 的  $\overline{RAS}$ 、 $\overline{CAS}$  信号对调,也就是将第二排 RAM 变成 BANK0,将第一排 RAM 变成 BANK1。如果第二排 RAM 完好,对调后系统能够完成初步自检并出现光标和显示错误代码,这不但说明了故障在第一排 RAM,而且损坏芯片的位置也由错误代码给出。根据前面介绍的利用错误代码确定损坏的 RAM 芯片的方法,找出损坏的 RAM 芯片的位置,将该片烫下来,换上同型号的新芯片,故障即可排除,这样就避免了一只只焊下来更换。

具体做法是:在系统板上找到 74LS138 译码器 U42、U56,把 U42 第 14、15 脚的引线用锋利的小刀割断后,用细导线互相对调连接;把 U56 的第 10、11 脚引线也割断并互相对调连接,见图 3。

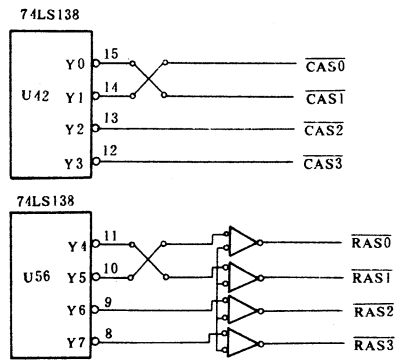


图 3 译码电路

焊好后要用万用表的电阻档测量一下各连线是否可靠,有无碰线、短路等,确定无误后方可开机,如果电路中仅是原来的第一排的某个 RAM 芯片损坏,则稍候即可显示光标和错误代码(此时屏幕上显示的错误代码是 BANK1 的信息,实际查找时应到 BANK0),损坏的芯片更换后,将 U42、U56 对调的引线复原。

在维修实践中,利用上述检修方法,对于由 RAM 芯片损坏而引起的故障现象,基本上都能够准确地诊断并加以排除,但其它因素如线路板断裂,扩充板开关位置不对等也会造成类似的故障现象,因此在维修过程中也必须加以考虑。





# 第二讲 故事情节和游戏结构

山东苍山县机械电子化学工业局(277700) 于 春

电脑游戏一般分为战争、冒险、管理、智能四大类。不管那一类游戏都有一个中心内容,能够完全细致地描述一个事件的发生、发展、结束全过程;有一定的知识性、启迪性和趣味性;有一定的难度和技巧,以激发游戏者的好胜心和竞争意识,促使他们思索、推理、判断、总结,从而达到游戏中增长知识,娱乐中提高才干的目的。

游戏程序是如何编制的呢?它一般要经过确定故事情节、规划游戏结构、确定程序结构、绘制程序框图、编写详细程序五个步骤。如果把游戏比作机械产品,故事情节就是原材料,游戏结构是加工图纸,程序结构是零件毛坯,程序框图是工艺文件,编制程序就是产品的最后加工。

众所周知,图纸是机械制造的依据。游戏结构则是编制游戏程序的关键,是故事情节的再创作、深加工。那么游戏结构的内容如何呢?欲说清这一问题,还须从故事情节说起。

我们知道许多战争、冒险类游戏都是从一个特定的故事情节开始的。如最为游戏者喜爱的双打游戏“魂斗罗”的故事情节是这样的:“公元 2631 年,世界大同,进入了和平时期。但是,地球海军陆战队司令部在一项绝密调查中发现了外星侵略者消灭人类的计划。海军陆战队所屬上等兵比尔·拉依乍和兰士·比恩受命歼灭外星侵略者。于是二人便向已变成侵略者要塞的嘎尔嘎戈岛进军。他们要突破原始森林、外围基地、瀑布天险、防御基地、雪山屏障、动力中心、飞机仓库七个关口,最后抵达外星人的栖息处,击毙外星人。摧毁敌基地后,乘飞机胜利返航。”

故事情节犹如文学创作的素材,有了故事情节,才能决定游戏结构。如魂斗罗的游戏结构是这样的:“这是一个战争游戏。因此首先给主人翁配备了光电子弹、火焰喷射器、火球、流星、辐射弹等各种武器。其次由于要经过各种复杂地带,主人翁要具有潜水、跳跃、飞腾的本领。第三,要设计各种不同的游戏画面、音乐以及与其对抗的敌人。第四,为增加游戏难度,把游戏分成了八大关,每关的关头都要与一个势力较强的敌人搏斗。若战术不力,很难过关……”等等。

归纳起来,所有游戏的游戏结构其本质上都是相同的。它一般包含以下几项内容:

### 1. 游戏的背景画面

游戏的背景、画面如同影剧中的布景,要根据游戏的故事情节,决定画面的种类、色彩、数量、变化方式和复杂程度。以增强游戏的艺术效果,使游戏更形象,更生动,更富有吸引力。

### 2. 游戏的音响效果

为使游戏逼真,一般游戏中都配有音响效果,如开枪、炸弹爆炸、敌人被击中,建筑物被摧毁等情节,都要有特定的音响,以烘托游戏的气氛。再如飞机、汽车的引擎声,人和动物行走的脚步声等。以增加游戏的真实感。精彩的游戏与丰富的音响是密切相关的。

### 3. 文字说明

文字说明,一般包括游戏的故事梗概、玩法说明、人物对话、积分、奖励、武器数量等方面的叙述和表达,以弥补画面显示的不足。

### 4. 游戏者的动作

游戏者指游戏中的主人翁。要规定主人翁的动作、本领,如跑、跳、潜水、腾空、射击、出拳、飞脚等。主人翁每一次动作后,都要计算其动作后的结果。若主人翁已取胜或失败,则进行结束处理(或者接关继续、或者重新开始)。否则,主人翁或另一位对手(或电脑)作另一次动作。动作的计算是游戏程序的核心。

### 5. 游戏的选项

游戏中可选项的类型、选择的方法和变量的修改,要根据选中的故事情节确定。如难度、关卡、武器、人数的选择等。而且一个选项可衍生出多层选项。如我们选中攻击,则继后就要有射击、出拳或踢脚等动作功能的选项。若选中射击,则又有各种武器的选项等等。选项一般与文字说明共存,也可以用画面选项。每个游戏选项的多少、选项的层次与游戏结构有关,同时还受电脑内存的约束。

### 6. 游戏的难度

一个有趣的游戏既要容易理解,又要有足够的难度;既要容易入门,又要在短时间内很难玩好。一般是设计许多关,开始几关较容易攻破,以后难度越来越大,要求技巧也越来越高。不动一番脑筋,不下一番功夫很难破关。只有这样,才能吸引游戏者不断思索,探讨,津津有味地玩下去。如多版游戏“淘金者”,I、II代各有 50 关。第 II 代的最后一关,难度最大,很多游戏者都过不去,必须精密策划,合理安排,快速行动才能过关。因此它是一个智能和技巧密切结合的典型游戏,受到广大游戏者的青睐。不难想象,一个很容易打穿的游戏,肯定令人乏味,难以引起兴趣。

游戏结构可粗略地划分以上六点。但并非所有的游戏结构都要具备这六点。如智能、管理游戏,只要有 1、2、3、5、6 几项就可以了。甚至可以更少,只有 3、6 两项也能构成游戏。可以断言,不具备以上几点内容的程序,肯定不是游戏程序,即使勉强凑成,也不会引起人们的兴趣。

另外,在设计游戏结构时,还必须兼顾电脑系统可供使用的内存裕量。诸如魂斗罗类的精彩游戏,结构相

当复杂,背景画面很多,程序极为庞大,一般都用机器语言编写,占用内存达 256K 之多。我们的学习机一般只有 4~8K 的内存供用户使用,所以,我们只能编一些简单的游戏程序。另外,由于用 BASIC 语言编程,程序的执行速度较慢,所以游戏中主人翁的动作是较慢的。虽然如此,游戏结构也是必不可少的。编程前,要对游戏程序精心策划,以使较少的内存发挥较大的作用,利用有限的内存空间编制出富有趣味的游戏程序。

现在,我们构思这样一个游戏,名字叫作“成龙救金凤”。故事情节如下:“成龙和金凤是一对热恋的情侣。山妖发现金凤美貌,就派山鹰叼走了金凤。欲打败山妖,救出金凤,必须找到制伏山妖的宝贝——降魔镖。但是,降魔镖是山神的镇山之宝,从不借人,只有冤机盗出。降魔镖秘藏在聚宝楼内,并由两名妖怪巡回看守,而聚宝楼是全封闭的楼房,每月只开一个时辰通风。成龙只有在通风时进入楼内,躲过妖怪,寻到宝贝,离开聚宝楼。否则,将葬身楼内。若被妖怪抓住则立即毙命。勇敢的成龙,机智地躲过妖怪,终于寻到宝贝打死山妖,救出爱侣。”根据故事情节,规化游戏结构如下:

### 1. 游戏画面

#### 配合游戏结构共设计四幅画面:

a. 开始画面:成龙和金凤正在郊外散步,突然降下山鹰,叼走了金凤;成龙狂奔追赶。

b. 主题画面:七层楼房;金凤显示在楼顶右上角;成龙在楼底左下角;两个妖怪随机出现在 3~7 层楼之间;九个大小、形状不同的宝箱散落在七层楼房的不同角落。

c. 结束画面两幅:一幅是成龙金凤二人欢呼相奔、拥抱。一幅是成龙和金凤手拉手高高兴兴回家。

### 2. 游戏的音响

a. 游戏开始有一段悠扬的乐曲,伴随着成龙金凤愉快的脚步。突然天空变暗,山鹰飞降,叼走金凤,乐曲紧促、强烈。

b. 成龙救出金凤,相向飞奔,热烈拥抱,有一段激动欢快的音乐。

c. 成龙与金凤手拉手回家时,又是一段悠扬的音乐。

d. 游戏进行中,穿插一些象声音响,如脚步声,上、下楼梯声,打开箱子声等等。

### 3. 文字说明

由于游戏较简单,不作说明,只打印题头、时间、得分等。

### 4. 动作

该游戏的主要情节是成龙依次打开箱子,寻找宝贝。因此,成龙的动作仅设计走路,上、下楼梯简单的动作。可用左、右键控制成龙左、右走;用上、下键控制成龙上、下楼;用 START 键控制游戏开始。

### 5. 游戏的难度

a. 从 27 个图形中随机选取 9 个作为存宝的箱子,箱子有一定的顺序(电脑约定,不标出),成龙只有按顺

序打开第一个箱子,然后再打开第二个、第三个……直到第九个。这犹如宝贝放在第九号箱子里,九号箱的钥匙放在八号箱,而八号箱的钥匙又放在七号箱……。所以必须从一号箱开始,依次打开,才能最终找到宝贝。

b. 两个妖怪在七层楼之间穿梭巡逻,并自动向成龙靠近。

c. 加进时间限制,超过规定时间,没找到宝贝,则游戏结束。

为压缩程序量,省去了选项。欲增加游戏难度,读者可调整时间参数,可把 T1 从 800 调到 600、400。

这就是成龙救金凤的游戏结构,规划好游戏结构后,就可进入下一步骤了。留待下一讲讨论。下面介绍一个简单的赌博游戏程序。

### 例三 赌博游戏——石头剪子纸

石头、剪子、纸的游戏(也叫剪子、包袱、锤)是小朋友最常玩的赌胜游戏。游戏中,双方在石头(ROCK)、剪刀(SCISSORS)、纸(TISSUE)三者之间任选其一。规则是,石头击败剪刀、剪刀击败纸、纸又击败石头。为了取胜,游戏者必须仔细研究对方的战术,随时调整自己的选择击败对方。游戏虽然简单却饶有趣味,以致成人也常用这种方法决定某些事件的输赢。这个游戏我们也可以与电脑一起玩。中华学习机已有成熟的程序,现改造移植到学习机中。

程序清单如下:

```

5 REM "GAMBLE GAME"
10 CLS:CLS:CG,0,0
20 LOC.7,3:P." * GAMBLE GAME *"
30 LOC.0,9:P."GUESS ROCK,SCISSORS,TISSUE":P.
40 P."BY PRESSING (R),(S),(T)":P.
50 P."TO END GAME PRESS (ESC)":P.:LOC.10,18:P.
 "1992.3."
60 GOS.360:CLS:GOS.310
70 PLA.B 0,RND(12)+1,13,29,45
75 LOC.0,0:P."YOUR TOTAL":T(2)
80 LOC.15,0:P."MY TOTAL ":T(1)
90 LOC.6,8:P." ":LOC.18,8:P."
 ":LOC.10,15:P." "
100 M=1+ABS(N(S,1)<=N(S,2))
110 M=M+ABS(N(S,M)<=N(S,3))* (3-M)
120 R=S : M=M+3*ABS(M=1)-1
130 LOC.0,8:P."YOUR ":G$=INK.(0)
140 IF G$="R" OR G$="S" OR G$="T" T.170
150 IF ASC(G$)=27 T.260
160 G.130
170 X=0:GOS.290:P.MI.(P$,X,9)
180 S=ASC(G$)-81:N(R,S)=N(R,S)+1
190 LOC.15,8:P."MY":G$=CH.(M+81)
200 X=0:GOS.290:P.MI.(P$,X,9)
230 IF S=M T.LOC.10,15:P."A DRAW":GOS.360:G.90
240 W=S-M+ABS(M>S)*3:T(W)=T(W)+1
250 LOC.10,15:P.N$(W):"WIN":GOS.360:G.70
260 CLS:P.:IF T(1)>=T(2) T.P"You loose!":G.280
270 P."You are very clever!"

```

```

280 LOC. 0,20;E.
290 F.I=1 TO 24 ST. 9;IF G$=MI.(P$,I,1) T.X=X+1
300 N.;RE.
320 P$="ROCK SCISSORS TISSUE "
330 DIM T(2),N(3,3),N$(2)
340 R=RND(3)+1;S=RND(3)+1
350 N(R,S)=1;N(S,R)=1;N$(1)="1":N$(2)="You";RE
360 LOC. 0,22;P. "Press space bar to continue"
370 A$=INK.(0);IF A$<>" " T. 370
380 LOC. 0,22;P. "
390 RE.

```

程序说明:

1. 10~70 行打印游戏标题和游戏说明。并随机改变背景的色彩,以使显示活泼。
2. 75~80 行打印游戏者和电脑的得分。
3. 90 行的作用是清除上次打印的游戏者、电脑所选内容及胜负结果。
4. 100~120 计算并确定电脑本次所选的内容。它是根据游戏者以前选择的规律算出的。这三行是本程序的关键,它决定了电脑的对策。
5. 130~170 行记录游戏者所选的内容,并打印在屏幕上。150 行为结束转移程序。
6. 180 行记录游戏者本次所选的内容。
7. 190~200 行打印电脑所选的内容。
8. 230~250 行判定胜负,累计得分。其中 230 行是对两者所选内容相同时的处理。240 行对得胜的一方加一分。250 行打印哪一方胜。
9. 260~280 行为结束处理程序。若电脑积分多,则打印“你输了?”;若游戏者得分多,则打印“你真聪明!”
10. 290~300 行是确定双方所选内容名字的全称的子程序。因在选择输入中,只输入石头、剪子、纸的字母 R、S、T,则由该子程序翻译出三种选择项的全称,打印出来。

11. 320~350 行是初始赋值子程序,也可以把它放入主程序的 60 行之后。

12. 360~390 行是场次转换子程序。每进行一场比赛,按空格键再进入下一场。

运行程序,清屏,显示

\* GAMBLE GAME \*

```

GUESS ROCK,SCISSORS,TISSUE
BY PRESSING (R),(S),(T)
TO END GAME PRESS (ESC)

```

1992. 3.

Press space bar to continue

标题内容是“赌博游戏。猜石头、剪刀、纸。依次按 R、S、T 键。结束游戏按 (ESC) 键。按空格键游戏开始”。

按空格键后,清屏,显示

```

YOUR TOTAL 0 MY TOTAL 0

```

```

YOUR ? MY

```

上一行显示:左边是游戏者(你的)得分,右边是电脑(我的)的得分。

第二行显示你选什么,后跟一光标,等待输入。比如,我们选石头,则键入“R”。这时,电脑若选剪刀,则打印

```

YOUR ROCK MY SCISSORS
YOU WIN

```

Press space bar to continue

意思是“你选的是石头,我选的是剪刀,你赢了。按空格键继续。”若按空格键,这时在“YOUR TOTAL 0”一栏中变为“YOUR TOTAL 1”。若第二次我们选石头“R”,电脑也选“R”,则打印“A DRAW”。意思是“不分胜负”。

若欲结束游戏,则按 (ESC) 键,电脑将根据你得分的多少,打印评语。

通过阅读程序和游戏实践可知,电脑能够记住每次游戏者的选择,并把它与游戏者前一次的选择联系起来,进行分析判断,做出击败游戏者的选择。电脑是怎样进行这一判断的呢?我们先看一张表

紧跟着的选择

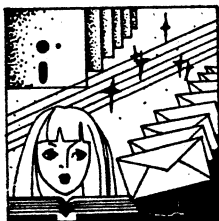
	石 头	剪 子	纸
前 一 次 选 择	石 头	0	8
	剪 刀	9	1
	纸	0	3
		3	9

游戏者选择记录表

上表是一个 3×3 阵列的目标表。表中的每一个单元对应了程序中 N(3,3) 数组的一个下标变量。电脑把游戏者每次选择记录在这张表中。轮到电脑选择时,它首先查看游戏者上次选的是什么,习惯于哪种规律的选择,从而选出取胜游戏者的玩法。

假定游戏者习惯于出石头后接着出剪刀,在上表的第一行中记录了游戏者的这种选择规律。即选过石头后再选石头为 0 次;选过石头后选剪刀为 8 次;选过石头后再选纸为 1 次。那么电脑知道游戏者在本次选择了石头,下一次大多数选剪刀,于是它自己就选石头以取胜(下面两行类推)。游戏中,电脑一直盯着这张表,以做出取胜的选择。当然,在实际游戏中,一旦游戏者意识到电脑已经掌握了他的规律,就会选用另外的策略。而当游戏者采用的策略多于一个时,电脑又会相应地选择对付游戏者的策略。不难想象,如果游戏者固定采用一个惯用策略,那么得胜一方将总是电脑。当然,绝大多数游戏者会意识到这一点,及时改变策略。

这一游戏结构比较简单,读者可自行分析,下一讲介绍游戏的程序结构和程序框图。



## 电子工业出版社软件部最新出版软件

### 新书与软件

#### 1. XF—通用管理系统 V1.0

本系统有三大特点:具有数据库的主要功能,但不需 CP/M 系统支持及 Z80 卡;响应速度快,全汉字操作;具有齐全、方便的记录和文件管理及统计打印功能。

本软件分为:1. 记录操作,包括查询、修改、删除、恢复、查库和输入共 6 个模块;2. 库操作,包括增字段、删字段、改字段顺序、改字段名、改字段类型、改字段长和库整理共 7 个模块;3. 统计打印,包括合计、统计、查询打印、顺序打印和查库共 5 个模块。此外还有磁盘及文件的基本操作。本软件配有详细的操作说明,可使用户操作一日通。

出版:1992 年 9 月 磁盘:1 张 定价:160 元

#### 2. 工资管理系统

本软件适用于中、小企事业单位进行工资管理,它具有 6 大特点:系统设计合理,功能全面实用,可管理 1400 人的工资档案,针对每人,可使用 24 个数据项和 3 个统计项;运行、存取速度快,数据修改方便;用户界面好,操作简单,提示明确;适用多种打印机型并可进行局部打印;系统还设有保密口令,以使数据资料安全可靠。

出版:1992 年 9 月 磁盘:1 张 定价:160 元

#### 3. CEC—I 英语单词记忆系统

本软件是新一代英语单词学习系统,它有 8 大特点:1. 单词词组随机出现;2. 单词库容量大,可存储 4050 单词;3. 自动时钟计时,它可计算学习时间和效率,使学习成绩量化;4. 有 8 种记忆方式,以适应不同的学习者;5. 8 种记忆方式可单选也可设置自动顺序学习,如设置 1、3、6、7 方式连续学习;6. 可查阅全部学习成绩值:学习时间,测验成绩,学习效率等;7. 可自行设定记忆牢固度值,在学习中系统根据测验的成绩和此值判定单词是否已掌握,对已掌握的单词系统不再提供记忆学习;8. 用户界面友好,操作简单可靠。

出版:1992 年 9 月 磁盘:1 张 定价 48 元

#### 4. 猜谜大奖赛

这是一个集益智、趣味和知识于一体的游戏软件,它配备了适合于儿童、少年、青年和成人的谜语一千余条。这些谜语构思精巧,妙趣横生。对猜中谜语者,软件还提供诗词、格言、音乐、对联、趣闻、笑话及游泳等作为奖品供猜谜者欣赏和娱乐。

本软件设计新颖,谜语全部采用美术字显示,并设有计时、记分和谜底显示功能。此外软件还可进一步增加新谜语。

出版:1992 年 9 月 磁盘:1 张 定价 30 元

(以上 4 种软件仅适用中华学习机)

## C-DBAG 中文数据库应用生成器系列新产品

### F/D \* AG V5.0

北京航空航天大学计算机系(100086) 李昭原

C-DBAG 中文数据库应用生成器是为数据库管理系统开发和使用而研制,其中 2.0 版曾获北京市科技新产品三等奖,4.0 版在全国首先推出网络多用户网络版,F/D \* V5.0 是在这些研究基础上,推出的 C-DBAG 系列的最新产品。

自动生成网络多用户 dBASE III plus 或 FoxBASE plus 源程序

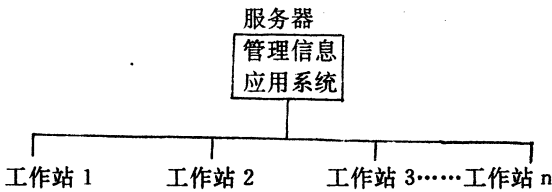
数据库应用生成器,用户在使用时只需按生成器中

中文提示,不需编程,即可生成(如财会、金融、统计、销售等)各种信息管理系统,并且在功能变动时,可任意修改,数据库应用生成器的使用,可大大提高编程人员的工作效率,减轻工作强度,并提高应用系统的可维护性。

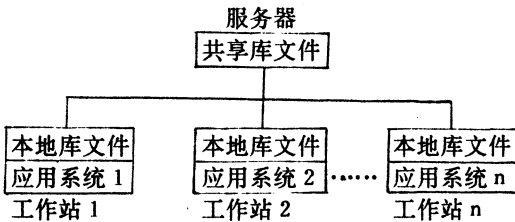
自 4.0 版以后,C-DBAG 产品根据市场需求特别强调,网络多用户应用系统的生成器,并根据不同环境为用户提供两种不同使用方式。

在用户有功能较强主机,管理信息联系紧密的应

用环境下,用户可生成一个应用系统,把其安置在服务器上各个工作站共享这个系统。



而在信息组成复杂,相互独立性较强的应用环境,用户可只将共享库文件安置于服务器中各个工作站可根据自己需要,生成管理共享库,与本地信息库的应用系统。



根据目前用户的需求,F/D \* AG5.0 共有以下 8 个功能生成器,用这些生成器即可生成单用户又可生成多用户系统。

### 下拉式/弹出式菜单生成器

菜单生成器可自动生成工作站独立的或多数子系统的系统菜单,当生成至每菜单树叶时,则自动要求生成共享库文件名或本地库文件名菜单,生成菜单理论值为 6 层,各级子系统个数最多达 20 个,菜单屏幕格式及画面生成可自编或系统定义。

### 数据库文件生成器

库文件结构生成器将建立共享或本地各数据库文件的数据结构,同时可建立代码数据库文件。

### 数据录入与维护模块生成器

生成对单库和多库进行添加、插入、修改、删除、查询、打印一体化操作程序,用户可选择录入屏幕格式,生成数据快速批量更新操作,生成浏览数据操作,生成从其它库添加操作生成代码库维护操作,生成修改器全操作,生成数据备份操作。

### 查询模块生成器

生成索引快速查询,匹配模糊查询,组合固定条件查询,组合任意条件查询,任意字段查询结果即可在屏

幕显示也可在打印机上打印。

### 统计与计算模块生成器

生成分类快速汇总,条件字段统计与计算,分类条件统计与计算,生成条件横向纵向统计与计算生成累计统计,上述各种统计与计算均可在单库或多库上进行。

### 统计图形生成器

用户可选择曲线图,直方图或圆饼图显示统计结果数据,可大屏幕地纵、横向显示,彩色图形的统计结果数据,并自动标记汉字坐标和统计结果值、统计图形打印程序支持图形打印的放大功能(放大倍数分别为 1、2、3、4、6、8 倍)。

### 任意格式报表生成器

可生成带横纵向统计与计算,带表头、表体、表层的任意格式报表,用户对字体、字号、报表格式、表头、表体、标题、尾注,左右角标均可自行定义,且可选择和控制打印字段,字段打印顺序,可实现分页打印,非零打印,折行打印,最大宽度可达 300 列。

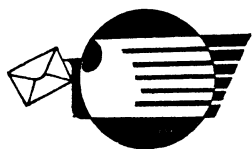
### 数据字典与文档生成器

由字典生成、文档生成、管理维护三个子系统集成,字典生成器通过扫描一个具体 MIS,可自动生成其库结构字典及程序结构字典。

### 特点:

1. F/D \* AG V5.0 各生成器可单独也可集成使用。
2. 生成器全部由 C 语言编程,具有很强的在线帮助及容错性能。
3. 软件生成全部采用对话方式,易学,易懂。
4. 可生成单用户系统,亦可生成网络环境下多用户信息管理系统。
5. 生成系统全部为 dBASE II plas 或 Fox BASE plas 源程序,用户可修改。
6. 支持“快速原型——增量”开发”方法:便于开发及修改。

C—DBAG 系列产品在现有生成 FoxBASE plas、dBASE II plas 信息管理系统的基礎上,将进一步开发其它数据库管理生成器,将要开发的有 ORACLE、SYBASE 等大型数据库管理系统的生成器、将支持 WINDOWS 界面,支持与高级语言的连接,从而大大提高信息管理系统水平,增大应用范围,为大中型厂矿企业的信息管理系统的开发提供快速工具。



读者联谊

# 普及型 PC 个人用户软件 交流联谊活动问题解答(九)

北京中国农科院计算中心(100081) 王路敬

### 30. 对硬盘进行格式化的目的是什么?

磁盘格式化的第一个目的是根据建立 DOS 分区 FDISK 记录在硬盘上的信息,对属于 PC-DOS 分区的一部分进行检查,不再对整个硬盘进行物理格式化;第二个目的是初始化 PC-DOS 的文件分配表和文件根目录区,初始化磁盘结构参数表,如果要从 PC-DOS 分区引导,则还要建立 PC-DOS 分区引导程序并记录在 PC-DOS 分区的第一扇区,同时建立 IBMBIO.COM, IBMDOS.COM(对长城系列机为 GWBIO.COM, GWDOS.COM, COMMAND.COM 三个文件。

格式化的工作是通过随机提供的 FORMAT.COM 文件来完成的,在操作系统提示符下打入:FORMAT C:/S/V<回车>再根据提示信息进行,用户将看到信息:

```
Format complete
system transferred
```

这个信息告诉用户硬盘格式化操作已经完成并已装入了 DOS 的副本,然后将出下列信息:

```
Volume label(11 characters)
```

用户可打入不超过 11 个字符的卷标名,以后执行 DIR 命令和 CHKDSK 命令时就会显示这个卷标名。如用户不希望给他的硬盘赋一个卷标名,则可直接按回车键。

### 31. 在 PC-DOS 不同版本中硬盘分区和格式化程序文件的长度有什么不同?

硬盘分区命令和格式化命令的功能随 PC-DOS 版本号增高而增强。一般在 PC-DOS 低版本下分区和格式化的硬盘,在高版本下可以对硬盘进行读写操作,但是在 PC-DOS 高版本下分区和格式化的硬盘,在低版本系统下不能对硬盘进行读写。硬盘分区命令 FDISK 和格式化命令 FORMAT 在 PC-DOS 不同版本中其文件长度有较明显的不同,如表所示:

表 PC-DOS 不同版本中 FDISK 和 FORMAT 文件长度比较

DOS 版本号	2.0	2.1	3.0	3.1	3.2	3.3
命令名称						
FDISK	6177	6369	8076	8173	8173	48216
FORMAT	6016	6912	9015	9398	11135	11616

### 32. PC-DOS 2.0/2.1 与 PC-DOS 3.0 以后版本的文件分配表有何不同? 对读写硬盘操作有什么关系?

DOS 通过文件分配表 FAT 来管理磁盘空间的分配,反映磁盘空间当前的使用情况。当文件需要空间时,系统将动态地分配一个簇。实际上,FAT 对每个文

件来说起着连接作用。它将磁盘文件实体所占的数据扇区连成一个文件整体。当对硬盘进行读写操作时,硬盘上最小可寻址的单位是一个扇区。通过 DOS 的文件系统功能来读写盘上的一个文件时,DOS 是以簇为单位进行的,簇是磁盘空间的分配单位,一个簇总是对应一个或多个逻辑扇区。盘上的每一簇,FAT 中必须有相应的一项来登记它。PC-DOS 2.0/2.1 中 FAT 表中的每一表项固定为 12 位(即 1.5 字节)长。PC-DOS 3.0 以后的版本里,FAT 的每一表项可以是 12 位,也可以是 16 位。12 位长的表项最多可以表达 4096 个簇,而 16 位长的表项最多可表达 65518 个簇。

当读硬盘上的文件时,FAT 与文件目录一起确定文件在硬盘上的位置;在写文件时,FAT 帮助确定盘上哪些簇是空的,可供分配的。对于一个 20MB 硬盘(GW0520CH 机),磁盘总扇区数 41751 个,采用 12 位长的表项,每簇为 16 扇区。同样是该硬盘,若采用 16 位长的表项,则每个簇分配不到一扇区,即小到 512 个字节。

如果每簇的字节数太小,文件在磁盘上分布过碎,读写文件的速度就会减慢,不宜采取。每簇所包含的扇区数与磁盘的容量有关,而在 PC-DOS 下每簇扇区数是由 FDISK 和 FORMAT 命令确定的,且以后不能改变,除非再一次运行 FDISK 和 FORMAT 命令。在 PC-DOS 3.0 以后版本里 FDISK 命令建立 DOS 分区时,究竟是使用 12 位还是 16 位表项来表示簇号,取决于 DOS 分区是大于还是等于 20740 簇,大于或等于则所建立的 FAT 的每一表项长为 16 位,否则为 12 位。

### 33. 新一代长城机基本配置硬盘的型号、主要参数及 CMOS 设置类型是什么?

掌握新一代长城机 GW0520DH、GW286B、G286BH、GW286EX、GW386/20 基本配置硬盘的型号,主要参数及 CMOS 设置类型是正确设置系统配置的一项重要内容,必须了解。如下表所示:

机器型号	硬盘型号	设置类型	容量(兆)	硬盘参数			
				最大磁道数 CYLS	磁头数 HEADS	每道扇区数 SECTORS	予补偿值 PRECOMP
GW0520DH	HH-825	—	20	615	4	17	300
GW286B	HH-830	3	30	615	6	17	300
	ST-238R	3	30	615	6	17	300
GW286BH	HH-1050	47	40	1024	5	17	NONE
GW286EX	ST251-1	40	40	820	6	17	NONE
GW386/20							

如是 GW286BH、GW286EX、GW386/20 等机器，当配置 ST251-1 时，其机器出厂流水号尾部有英文字母“S”，若无“S”的机器，如无特殊说明，则配 HH-1050 硬盘；所有配此硬盘的机器，如其 CMOS 硬盘类型 47 的各参数与表中参数不符，请选用类型 11。

### 34. 在什么情况下，从硬盘引导操作系统时，进入 ROM BASIC 解释程序？

当从硬盘上启动操作系统 PC-DOS 或 CC-DOS 时，有时启动失败，进入 ROM BASIC 解释程序，出现这种情况通常有下列原因：

(1) 当系统既没有硬盘又没有在软盘驱动器 A 中插入系统盘时，将使启动失败，而进入 ROM BASIC。因为 DOS 启动时，首先检查 A 驱动器有无 DOS 盘，若有，则从 A 驱动器将 DOS 装入内存，若无 DOS 盘，则判断 C 盘上有无 DOS，若有，则从 C 盘装入内存，A 驱动器没有 DOS 盘，C 盘也没有 DOS，则进入 ROM BASIC 解释程序。

(2) 既不能正确读出软盘第 1 扇区，又不能正确读出硬盘第 1 扇区。因为软盘的 0 面 0 道 1 扇区是引导扇区，在系统启动时用它来查找和装入 PC-DOS 的两个隐含文件 IBMBIO.COM 和 IBMDOS.COM，并把控制权交给 IBMBIO.COM，它的作用是首先检查驱动器 A 上是否插入了系统盘，若是，则引导 DOS 进入内存，否则给出出错信息并进入 ROM BASIC 解释程序。除此之外，还要检查上述两个隐含文件的顺序对不对，正确的顺序是 IBMBIO.COM 在先，IBMDOS.COM 在后。如果这个顺序不对也要给出出错信息。硬盘的第 1 扇区是主引导程序和分区信息表。硬盘分区表描述了硬盘分区状况，主引导程序供硬盘自举使用。该扇区的前 240 个字节是主引导记录，从 1BEH 位移开始的 64 个字节是硬盘分区表，最后两个字节是 AA55H，是自举记录的有效标志字。

(3) 没有在驱动器 A 插入 DOS 盘片或不能读出软盘的第 1 扇区，而硬盘上第 1 扇区最后一个字不是 AA55H，也进入 ROM BASIC 解释程序。

(4) 没有在驱动器 A 插入 DOS 盘片或不能读出软盘的第 1 扇区，而硬盘上又没有可引导的分区，也会进入 ROM BASIC 解释程序。

### 35. 在何情况下，从硬盘启动系统失败而进入死循环？

在下列条件之一发生时，系统引导失败，进入死循环：

(1) 没有在软盘驱动器 A 插入 DOS 盘片或不能读出软盘第 1 扇区，而硬盘有多个引导分区。一个硬盘系统可分为 4 个“分区”，划分这些分区的目的是支持 DOS 以外的操作系统。因而可在同一个硬盘上驻留 DOS 以及其他磁盘操作系统，并且都可工作，最多可容纳四种操作系统，每种操作系统在硬盘上建立自己专用的分区操作时都有一个相应的实用程序。分区的具

体数目和 DOS 分区的大小可由 DOS 外部命令 FDISK 设置。每一个分区都有一个对应的分区表，由 16 个字节组成，若第 0 字节值为 80H 时表示该分区可自举，其值为 00H 时，表示该分区不可自举。当硬盘上引导标志多于 1 个时即显示：“Invalid Partition Table”，进入死循环。

(2) 没有在驱动器 A 插入 DOS 盘片或不能读出软盘第 1 扇区，而硬盘 PC-DOS 分区引导程序又读不出来。通常在 DOS 分区的首扇区上存放 DOS 引导记录，当硬盘主引导记录搜索到 DOS 引导记录扇区时，便将控制权交给 DOS 引导记录，由它负责将 DOS 装入内存，这一过程和软盘启动相同。当硬盘 PC-DOS 分区引导程序读不出来时，当然使启动系统失败，此时会显示：“Error Loading Operating System”，使系统进入死循环。

(3) 没有在驱动器 A 插入盘片或不能读出软盘第 1 扇区，而硬盘 PC-DOS 分区引导程序的最后一个字不是 AA55H。一个硬盘 PC-DOS 分区引导程序正常的情况下，其最后一个字的目标码应当是 AA55H。若不是，则显示“Missing Operating System”。

(4) 不能把 IBMBIO.COM 文件读入内存。IBMBIO.COM 是 ROM 中 BIOS 的低级接口模块，在系统启动过程中，IBMBIO.COM 负责测定系统中设备的状态，初始化附加设备，使磁盘系统复位，设置低序号的中断向量，解释 CONFIG.SYS 文件并设置系统环境，加载可安装的设备驱动程序以及给引入内存的 IBMDOS.COM 重新定位。不能把 IBMBIO.COM 文件装入内存，将显示：“Disk Boot Failure”，从而进入死循环。

### 36. 开机后没有显示“1701”号错，但每次企图转 C 盘时屏幕只是显示：“无效驱动器参数”，是什么原因造成的？

出现这种情况一般有两种可能的原因造成。其一是硬盘控制器上的 ROM BIOS 程序被禁止了或有其他错误。硬盘 ROM BIOS 主要提供硬盘输入输出驱动程序，它的组成包括以下几个组成部分：

(1) 硬盘输入输出参数的定义，其中包括：错误代码参数，设备地址端口，硬盘控制器命令字。

(2) 硬盘数据区。它包括在 0 段设置磁盘中断向量号，在 40 段设置磁盘状态字。

(3) 硬盘输入输出的初始化程序。

(4) 自举装入程序——INT 19H。

(5) 硬盘输入输出驱动程序 INT 13H。

硬盘通过硬盘控制器的 ROM BIOS 进入系统的 ROM BIOS，从系统的 ROM 中取得对硬盘的输入输出请求的控制。

第二个原因：是硬盘上第 1 扇区的 DOS 分区信息或有关磁盘参数受到破坏。引起这种故障常常是由于主机电源容量不足，使得在长期工作时，主机突然掉电自动保护，接着又立即加电开机，这一断一通，对于正在读写的控制器和驱动器容易造成某些信息的丢失或

误写。为此作为主机的日常性维护,其中有一条要求开、关机之间时间间隔不要太短,即频繁的开关机是造成硬盘损坏或信息丢失一个重要原因。

### 37. 硬盘使用一段时间后读写文件时容易发生错误,是什么原因?

从使用的实践经验上来看,其原因既有介质方面的,也有磁头定位精度方面的,但使用方面的原因是一般用户尤其要注意的。对于 40MB 以下的大多数硬盘,在断电之后不会自动退回到非数据区磁道,关机之前磁头停在哪里,关机之后仍停在哪里,并和盘面接触。如果主机在这时搬运受到冲击,硬盘的磁头有可能擦伤硬盘盘面而影响读写。一旦出现这种情况需作格式

化的操作,才能重新建立硬盘系统。

### 38. PC-DOS 对硬盘容量是如何限制的?

由硬盘 PC-DOS 分区引导程序中的硬盘基本输入输出参数表分析,该表由 19 个字节组成,其中有二个字节表示硬盘的总扇区。二个字节为 16 个二进制数,可以表示最大数为 65536,因此,一个硬盘最大不能超过 65536 个扇区,大约 33MB。如果一个硬盘的容量大于 33MB,则需要把它划分为几个逻辑盘,而每个逻辑盘不能超过 33MB 最大限数。例如长城 286BH 配置硬盘为 40MB,可划分 C 盘和 D 盘两个逻辑盘,C 盘最大可为 33MB。

## 网络管理信息系统快速开发工具 支持“快速原型——增量”开发方法

### C-DBAG 中文数据库系列应用生成器新产品

## F/D \* AGV5.0

北京航空航天大学开发

使用本系统只需按照提示输入用户需求,不需编程,即可生成(如财会、金融、统计、销售等)各种管理软件,在功能变动时可任意改动,适用于网络及单用户环境。

**功能:**

- 生成 DBASE+/FoxBASE+ 源程序
- 生成任意格式报表
- 自动生成数据字典及文档
- 工程图形生成与管理

详细功能请看本期软件介绍栏目

**组成:**

- 下拉式/弹出式菜单生成器
- 数据库文件生成器
- 数据录入与维护模式生成器
- 查询模块生成器
- 统计与计算模块生成器
- 统计图形生成器

- 任意格式报表生成器
- 数据字典与文档生成器
- 92年10月27日在北航开办培训班

**经销:**中电华北公司电脑部 电话:81.1810

地址:北京万寿路西街五号

邮编:100036

联系人:石立军 魏国

北京景文科技开发公司

电话 832.2255 转 458

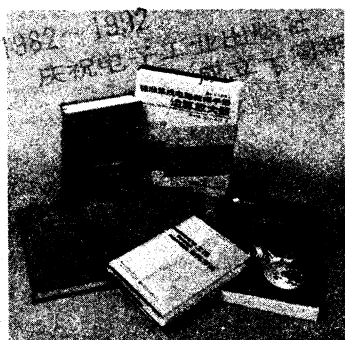
地址:北京西直门外首体主楼 305

邮编:100081

联系人:袁凯峰 李应知

本公司经营计算机及外设通讯设备,欢迎来电来函联系。





• ELECTRONICS AND COMPUTERS •

一九九二年

总期第91期

# 电子与电脑

## 目 录

### • 综述 •

- 改革开放结硕果 ..... 齐 心(2)
- 软件市场大有希望 ..... 温有良(2)
- 新兴的教育技术—CAI ..... 陈健(3)

### • PC 用户 •

- 磁盘的簇与簇管理分析 ..... 崔来堂(4)
- 程序运行过程中汉字/西文输入方式的自动切换  
..... 傅 雷(6)
- C 语言在配平化学方程式中的应用 ..... 王 晰(7)
- DISK MANGER 使用技巧 ..... 梁高军(9)
- 如何利用 dBASE III 的 F1 功能键 ..... 涂振宇(10)
- CCBIOS 2.13 稿纸方式打印 ..... 宋 捷(11)
- 微机屏幕的打印和放大 ..... 谭人杰(11)
- 1992 年全国青少年信息学(计算机)竞赛试题 ..... (13)

### • 学习机之友 •

- BASIC 子程序的递归调用 ..... 陈继良 丘 文(15)
- APPLE 机的快速排序 ..... 张 亭(16)
- 百年公历—农历互查 ..... 刘安军(17)
- 内存数据代码搜索程序 ..... 苏 华(20)

### • 语言讲座 •

- 6502 机器语言程序设计
- 第十章 监控子程序的调用 ..... 朱国江(21)

### • 学用单片机 •

- BJS—51 单片机实验系统(续) ..... 李广弟(25)
- CYSCB—ZMCS-518098 单片单板机硬件设计原理  
..... 吴 微(26)

### • 学装微电脑 •

- 微电脑控制微型钻床 ..... 易齐干(30)

### • 电脑巧开发 •

- 计算机语音输出功能的开发与应用(下)  
..... 陈竹林(35)

### • 电脑游戏机 •

- 第三讲 程序结构和程序框图 ..... 于 春(38)

### • 维修经验谈 •

- APPLE—II、CEC—I 故障诊断仿真系统  
..... 王志刚 张一建(43)

### • 读者联谊 •

- 普及型 PC 个人用户软件交流联谊活动问题解答(十)  
..... 王路敬(46)

### • 信息与服务 •

- C—DBAG 中文数据库系列应用生成器新产品 ... (28)
- 告读者作者和朋友 ..... 本刊编辑部(29)
- SJW—B 系列自动补偿式三相大功率电力稳压器  
..... (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:杨逢仪

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



## 改革开放结硕果

### 电子工业出版社建社十周年简介

我社自一九八二年成立,到今年十月二十二日将迎来十周年。在这十年里,由于党的改革开放方针的指引,上级领导机关的关怀和支持,全社同志克服困难,奋发努力,使我社以较快的速度健康发展,现已成为具有较强出版能力,在出版界具有较大影响的中央级科技出版社。

我社现有职工 160 多人,其中专业技术人员占 70% 以上,具有副编审以上高级职称的 25 名。年发稿能力 6000 多万字,年出书 300 多种,接近一日一书,达到人均两(种)书,出书码洋超过 3000 多万元,销售码洋近 3000 万元,实现销售收入约 2000 万元,人均创利超万元,据统计,十年共出版图书 2000 多种,总印数约 3500 万册,其中各类教材 400 种,专业学术专著 200 种,工程应用技术和工具书 800 种,其余为电子科普读物和直接为电子产品服务的实用读物。有数十种图书获得全国、省级科技成果奖,评为优秀畅销书。另外,还出版了《电子与电脑》杂志 90 多期,与图书配套的音像制品 30 多种,出版计算机软件上百种。这些书、刊和音像制品,门类齐全,适应不同专业,不同层次读者的需要,为我国电子工业的科研、生产、教育和电子科技成果的传播推广,以及电子科学知识的普及提供了良好的服务,受到广大读者和社会各界的好评。我社受新闻出版署委托,是高技术图书出版工作的牵头组织单位之一,电子工业出版社的知名度也随之不断提高。

目前,我社正认真贯彻邓小平同志南巡讲话精神和国务院关于加快发展第三产业的决定,决心抓住有利时机,加快改革开放的步伐,提高自我发展能力。我们的经营方针是:坚持一业为主,发展多种经营;巩固大本营,开发新基地,积极开展国内外不同形式的合作。图书出版工作无疑是我们的主业,必须以主要精力扎扎实实地大力去抓,这是我们不可推卸的责任,也是进一步发展和开拓的基础。我们要加强经营管理,在思想作风和业务建设等方面更上一层楼。与此同时,我们正积极探索加强国内外合作与交流的途径。一方面是出版业务的合作。如正和香港得实发展有限公司合作出版《UNIX 系统 V 第 4 版中文手册》丛书,共 49 本,2000 多万字,现已出版 9 本,其余力争年底出齐,这套丛书在我国计算机界产生了很大的影响;另外,已和美国万通科技有限公司达成协议,拟在北京合资创办“万国电子”刊物,得到了国家科委、机电部和新闻出版署的支持,现正在办理报批手续;和台湾“权岗出版公司”正在商谈加强合作的问题。另一方面是充分利用自身优势,大胆而又稳妥地在国内一些开放地区开辟新的基地,逐步拓宽业务渠道,如已与上海浦东、海南岛等地的单位商定在这些地区兴办合资企业。我们希望通过多方位的合作与交流,使电子工业出版社向国内外大市场发展。

我们深深体会到,电子工业出版社十年走过的道路是不平坦的;前面还有许多困难等着我们去克服。全社同志决心在新的形势下,发扬成绩,克服不足,同心协力,义无反顾,乘改革开放的东风,解放思想,挖掘潜力,迎接新的机遇和挑战。

本刊特约记者 齐 心

## 软件市场大有希望

中软总公司通用部副总经理 温有良

九十年代计算机市场发展重点是由硬件逐渐转向软件,一些先进发达国家以 25% 的增长率发展。1990 年,世界软件市场销售额 1000 亿美元,到 1991 年就超过了 1200 亿美元,其中微型机软件占 35% 以上。这些软件中最受欢迎的是文字处理软件、桌面排版软件、窗

口软件和 Novell 网络软件。

Novell Inc 宣布,该公司截至 1991 年 10 月 26 日,净收入 6.4 亿美元,与上一年同期 4.9 亿美元比,增加了 29%,由于 NetWare 3.0 网络服务操作系统软件营业额已超过了 3 亿美元,所以使得 91 年软件产品销售总额达 5.7 亿美元,增加幅度达 47%。

在我国,专家们认为软件市场潜力是很大的,可是目前启动很慢,从开发一个软件到商品化周期太长。造成这种现象的原因,一是我国软件价格便宜,开发软件

的劳动力更便宜,使得软件开发人员积极性不高,有的甚至到外国去进行劳务输出。二是国内软件开发缺乏统一组织,各单位以自我封闭作坊式进行开发,同一个产品几家都投入人力物力,有的产品市场上都有了,可是还有人再做重复性的工作。

自从软件保护条例公布以后,软件开发的积极性确实高起来了,在国内外引起了很大反响。国内一些计算机集团公司也清楚地看到了单一生产经营硬件路子会越来越窄,于是他们逐渐地转变为硬软结合,一手抓硬件产品更新,一手抓软件产品配套。这一举动也引起了一些外商注意,他们瞄准了中国软件市场,日本、美国、新加坡,还有台湾地区,他们积极地与国内一些大公司合资进行软件生产。

激烈的竞争引起国内有关部门的高度重视,决策部门重点地部署了软件生产基地,现在已确认的有南方软件生产基地和北方软件生产基地,这意味着,在中国要大力开发生产系列软件。中软总公司承担了北方软件生产基地的任务。

由于人们对硬件、软件认识观念转变了,使得计算机软件市场大门越开越大,推动了我国计算机的应用,提高了计算机应用率。广西壮族自治区在九一年底召开的全区第二届计算机应用年会上,区经委提出,检查企业达标最重要一条是,企业是否用上了计算机。还提出了计算机应用要求上新台阶,检查是否上了新台阶,最重要一条是:是否计算机联成了网。

随着软件产业的兴起,应用计算机的领域越来越

大了,可是,给计算机市场也带来了新的问题,就是软件应用人员短缺这个矛盾越来越突出了。在国外,一般先进发达国家硬件人员与软件人员是1:1关系,而我国目前据有关资料统计,硬件人员与软件人员比例悬殊,软件人员严重短缺。

据报导,日本到2000年,将缺乏100万各类软件人员。印度虽然软件出口高速增长,今年可达1.8亿美元,全国有700多家软件企业还在瞄准世界市场,可是他们面临的最大恐慌也是各类软件人员短缺。

我国91年软件经销单位有220家,销售软件400多万美元,92年软件经销单位270家,销售软件可能在800多万美元,这个数字并不高,为什么呢?其中一个原因是有些单位买了计算机不会用,想配软件人员又没指标,只好由硬件人员去硬顶,有些该买的软件,硬件人员也不敢买。看来人员结构比例不合理仍然是影响我国软件产业发展的一大障碍。这一严峻形势,引起了许多产业集团的关注。中软总公司与日本NEC、富士通等合资开发生产软件,国家又把重点攻关项目UNIX国产化任务交给了他们,可是该公司最大的困难,仍然是软件开发人员短缺。他们采取向社会招标、自办强化培训班等,都没有从根本上解决问题。

如今,软件市场竞争很激烈,表现在产品上,实质还是人员之间的竞争,如果国家从政策上能够保证合理比例的软件人员,再加上软件保护条例能够认真地贯彻;思想再解放一点,步子再大一点,那么,中国的软件市场是大有希望的。

## 新兴的教育技术——CAI

吉林四平师范学院 陈 健

CBE(Computer Based Education—计算机辅助教育)是计算机应用的一个重要领域;CAI(Computer Assisted Instruction—计算机辅助教学)是CBE的一个主要分支。CAI作为一门新兴的、现代化的教育技术,历史不过只有三十几年,但它所展现出的广阔前景正在吸引着越来越多的人,在世界很多国家得到越来越广泛的应用。在我国,CAI还较少为人所知。我国要实现现代化,作为其基础的教育应该首先现代化,CAI作为最新、最现代化的教育技术,应该为广大教育工作者了解和掌握,为现代化服务。

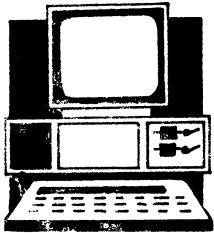
CAI起源于本世纪五十年代末。1958年,美国IBM公司设计了一个用于教授小学生二进制算术的教学系统,这是最早的教学软件,首次将计算机技术应用于教学,从此开始了CAI的历史,这一新生事物的出现,引起了很多有识之士的高度重视,一些发达国家的大学、研究院(所)和大公司纷纷投入资金和人力,开展对CAI的研究。

在美国,很多著名大学(如麻省理工学院、斯坦福

大学等)和计算机公司(如IBM公司、DEC教学设备公司)都对CAI进行了大规模的研究和实践,并推出很多计算机辅助教学系统,一些大学还制订出教学计算机化的计划。1967年,美国成立了“计算机教程公司”,专门研制和生产各种教学软件和教学管理软件,用于出售或出租,从而把教学软件推向了市场,成为商品。1984年,美国又创建了世界上第一所“电子大学”,通过远程教育网络向美国各地、加拿大和欧洲一些国家的学生教授课程,这一系统开设课程可达170门之多。加拿大的CAI研究开始于1986年,由十一所大学和一些研究院联合开发,取得了显著成果。英国在1972年制订出规划,投资200万英镑,计划开发29个辅助教育系统。日本的金泽工业大学已开发出近70门课程的CAI系统。为在发展中国家推广CAI技术,在联合国教科文组织的积极赞助下,国际信息处理协会曾多次在巴西、古巴、印度、尼日利亚等国举办有关CAI的研讨会,大大促进了CAI的发展进程。我国的CAI研究开始于六十年代,中间曾一度夭折,自1980年起,华东师大、西安交大、大连理工大学、华中工学院等数十所高等院校和一些基础好的中、小学都陆续开始了这方面的研究,取得了很好的成果。

CAI最主要的作用是教学,CAI应用于教学有两种形式:主体式(Primary CAI)和辅助式(Adjunct CAI)。

(下转29页)



## PC 用户

# 磁盘的簇与簇管理分析

石家庄铁道学院(050043) 崔来堂

### 一、盘的两端读写方式

PC DOS 磁盘的读写,从用户介面看,分为两种方式:一种是扇区读写,用户直接选择盘扇区,由中断调用 INT 13H、INT 25H(26H)或工具软件 DEBUG 等实现,扇区是这种方式的最小可寻址单位;另一种是文件读写,用户只需给出文件名和有关命令,由操作系统自动查找组成该文件的各个数据块,完成读写,这些数据块的分布及查找过程,对用户是透明的,DOS 对它们有一套严密的管理机制。

### 二、盘的分区结构与簇

磁盘空间(硬盘指 DOS 分区,下同)从功能上分为三个区:逻辑扇区 0 为保留区,即引导扇区;之后是内容相同的两份文件分配表 FAT,继之是文件目录表 FDT。FAT 与 FDT 合称控制区;再后直至末尾为文件区。文件区又均匀地划分为一系列的块,一块称为一个“簇”(cluster),文件分成若干簇存储在文件区。簇的大小为  $2^n$ ( $n=0,1,\dots,4$ )个连续逻辑扇区,它是磁盘进行文件读写的最小可寻址单位。

### 三、FAT 表与簇号链

为了充分利用盘空间,文件各簇在盘上的分布,是按“见缝插针”办法进行的,它们在物理上常常并不连续,经过多次文件删写的盘,更是如此。但是,为了使文件能正常读写,又必须保证这些簇在逻辑上的连续性,这是由 FAT 表来实现的。FAT 表均匀地分为若干簇域,并依次编号。000H 与 001H 号簇域合称保留簇域,其中第 1 字节为盘介质标志,定义如下:

磁盘种类	介质标志
5.25 英寸 360K 软盘	FD
5.25 英寸 1.2M 软盘	F9
3.5 英寸 1.44M 软盘	F0
硬盘	F8

保留簇域的其余字节内容均为 FFH;从 002H 号簇域开始,称使用簇域,用来记录文件在文件区分布的相应簇号。因此,文件区各簇,也相应地从 002H 开始依次编号。每个文件的起始簇号,记录在 FDT 表相应目录登记项的起始簇域,之后各簇号按下述规则记录在 FAT 表中:对应一个文件的各簇域,依次记录下一簇分配的簇号,直至最后簇,如此形成一个链表,称为文件簇号链。FAT 表包含着盘上所有文件的簇号链,因此也称簇号链库。

### 四、簇域长度的确定

FAT 表簇域的长度,按照磁盘类型、盘容量和所用 DOS 版本,分为两种:12 位和 16 位。各类软盘均为 12 位。硬盘:DOS 2. X 时 12 位;但因使用簇域从 002H

开始编号,又把 17 个簇域号留作专用,故 12 位簇域能够管理的最大簇数为  $2^{12}-1-17=4078$ ,因此,DOS 3. X 支持下的硬盘,当簇数不大于 4078 时,簇域长度为 12 位;大于时为 16 位。软盘的簇域长度,在 FORMAT 格式化时确定,硬盘在 FDISK 进行分区和 FORMAT 对 DOS 分区格式化时确定。

### 五、簇域值的观察和计算

FAT 的常见簇域值及含义如下表:

簇域值(12 位或 16 位)	含义
(0)000H	未用簇或可用簇
(F)FF7H	坏簇
(F)FFFH	文件的结束簇
(x)xxxH	文件下一簇簇号

簇域值可用 DEBUG 观察,方法是:

```
A>DEBUG
-L100 X Y Z (调 FAT1)
-D100 (显示)
```

其中,X 为驱动器代号(0=A,1=B,2=C,...);Y 为 FAT1 的起始逻辑扇区号(通常为 1);Z 为观测扇区数。

簇域长度为 16 位时,观察比较直观,只要将显示的高簇域内容的高低字节交换,即为簇域值。簇域长度为 12 位时,观察并不直观,必须按如下规则计算:待查簇域号乘 1.5,积取整后作为 FAT 表内的字节位移,由此取出 16 位的内容送某寄存器。若簇域号为偶数,取寄存器的低 12 位为该簇域值;否则,取高 12 位为簇域值。该规则可用下述子程序实现:

```
入口参数 AX=簇域号
DS:BX=FAT 的段:位移
```

```
返回参数 AX=所求簇域值
```

```
clust proc near
push bx
push cx
mov cx,ax
shl ax,1
add ax,cx
shr ax,1
add bx,ax
mov ax,[bx]
test cx,1
jnz odd
and ax,0fffh
jmp retn
odd: mov cx,4
shr ax,cl
retn: pop cx
pop bx
```

```

 ret
 clust endp
 为了实现 12 位簇域 FAT 表全部内容的自动直观
 显示,特编写了如下针对 360K 软盘的实用程序:
code segment
 org 100h
 assume cs:code,ds:code
start proc near
 mov al,1 ;检 B 盘
 mov cx,2
 mov dx,1
 mov bx,offset buf
 int 25h ;读 fat1
 popf
 mov si,0 ;si:簇域总数
rep1: mov di,0 ;di:每行簇域数
rep2: mov dx,[bx]
 mov cl,4
 shl dx,cl
 call disp ;显偶数簇域
 mov dx,[bx+1]
 call disp ;显奇数簇域
 add bx,3
 add di,2
 cmp di,16 ;一行显完否
 jb rep2
 mov ah,2
 mov dl,0dh
 int 21h
 mov dl,0ah
 int 21h
 add si,10h
 cmp si,355
 jb rep1
 mov ax,4c00h.
 int 21h
start endp
;显示子程序
disp proc near
 mov cx,3
rep3: push cx
 mov cl,4
 rol dx,cl
 push dx
 and dl,0fh;一位 16 进制
 cmp dl,9;转为 ASCII 码
 jbe aa
 add dl,7
aa: add dl,30h
 mov ah,2
 int 21h ;显一位
 pop dx
 pop cx
 loop rep3 ;显三位否?
 mov dl,20h

```

```

 int 21h ;插一空格
 ret
disp endp
buf db 400h dup(0)
code ends
 end start

```

该程序改动个别参数后,即可适用于 1.2M 或 1.44M 高密软盘,以及 DOS 2.×支持下的硬盘。

## 六、簇容量及其查询

文件区的一簇包含的扇区数,称簇容量。簇容量的大小,对磁盘的工作有较大关系。

由于文件是以簇为单位占据盘空间的,一个文件即使只有一个字节,也要占一个簇,因此,文件长度与它在盘上的分布长度是两个概念,后者通常比前者大。当短小文件(长度显著小于一簇的容量)较多时,会造成盘空间的较大浪费。为了提高盘利用率,簇容量应当小;但是,文件读写时,以簇为单位与内存进行信息交换,簇容量小,访问磁盘的次数必然增多,又因文件在盘上被分割得较碎,使磁头移动频繁,这些都会使操作效率明显降低。因此,确定簇容量时,考虑上述两方面的因素,选得适度小。

有关簇和扇区等盘参数,可由系统调用 INT 21H 的 1CH 子功能进行查询。调用的格式为:

入口参数 AH=1CH  
DL=驱动器代号(0=默认驱动器,1=A,  
2=B,3=C,……)

返回参数 AL=扇区数/簇,即簇容量  
CX=字节数/扇区  
DX=盘文件区总簇数  
DS;BX=FAT 表首字节的段;位移(该字节的内容即盘介质标志)。

由此查得的常见磁盘簇容量如下:5.25 英寸 360K 软盘,2 个扇区;5.25 英寸 1.2M 和 3.5 英寸 1.44M 软盘,1 个扇区;10M 硬盘,8 个扇区;20M 硬盘,使用 DOS 2.×时 16 个扇区,DOS 3.×时 4 个扇区;大于 20M 的硬盘(DOS 分区最大 32M,DOS 3.31 时除外),在 DOS 3.×支持下,簇容量均为 4 个扇区。

## 七、DOS 的簇管理过程

以建立文件和扩展文件为例,说明相应的簇管理过程。

每当 DOS 在盘上建立新文件时,总是从头开始搜索 FAT 表,跳过所有已分配的簇域,找到第一个可用簇域,把起始簇域号写到 FDT 表相应目录登记项中,并把文件从头分割出一块,写入文件区对应簇中;再继续查找第二个可用簇域,把簇域号写到第一个可用簇域,并向文件区对应簇写入文件的第二块,……依此类推,直至写完文件的最后一块,把结束簇标志(F)FFFH 写到 FAT 的对应簇域中,该文件的簇号链就形成了。

当对一个文件进行扩展时,DOS 也是从头搜索 FAT 表,找到第一个可用簇域,写入(F)FFFH,成为新的结束簇域,并将原结束簇域的值修改为指向新结束簇域,……如此下去,直至满足文件的扩展要求;文件

的扩展部分,则依次写入文件区对应簇。

### 八、结束语

簇管理策略是 PC DOS 的重要功能之一。用户深

入了解它的特点,对于提高编程能力,优化磁盘存储方案,改善文件读写效率,分析磁盘数据,解决病毒等问题,都是很有必要的。

## 程序运行过程中汉字、西文输入方式的自动转换

锦州铁路中心医院计算机室 (121001)傅 雷

在 2.13H 系统中,有一组模拟功能键的命令。格式为:CHR[\$](14)+“A 扩展 ASCII 码”。此命令将功能键的扩展 ASCII 码送入键盘管理模块,去执行系统定义的某种功能。例如,在 dBASE 环境下,由于需要输入汉字信息,可在程序中插入下面命令,将输入方式自动设置为拼音方式。

? CHR(14)+“A106” (相当于按 CTRL+F3 键)

如需转换其他输入方式,只要改变相应的功能键扩展 ASCII 码即可。常用的功能键扩展 ASCII 码见表 1。

(表 1)

功能键	内 容	扩展 ASCII 码
Alt+F1	区位码输入	104
Alt+F2	首尾码输入	105
Alt+F3	拼音码输入	106
Alt+F4	快速输入	107
Alt+F6	ASCII 码输入	109
Ctrl+F7	进/退纯西文	100
Ctrl+F8	建/取光标	101
Ctrl+F9	进/退纯中文	102

虽然此功能使用很方便,但是有一点很不理想,当在大写方式下的 ASCII 码方式转换到拼音方式后,仍为大写输入方式,这样必须转换为小写方式后,才能输入汉字。而且,在其他的汉字操作系统中,上述命令无效。下面给出在各种汉字系统中不同的输入方式转换方法,供大家参考。

大家知道,在 ROM BIOS 数据区中,从绝对地址 40H 开始的 256 个字节存放的是 ROM BIOS 的工作参数,其中,从 0017H(1047)到 001EH(1054)以及其后的 16 个字,是存放有关键盘的工作参数,详见表 2。

(表 2)

地址(10 进制)	内 容
1047(1 个字节)	换挡键和双态键状态
1050(2 个字节)	键盘缓冲区首指针
1052(2 个字节)	键盘缓冲区尾指针
1054(32 个字节)	环型键盘缓冲区

内存 1047 单元字节的每一位,表示了换挡键和双态键的状态。缓冲区首指针指的是第一个键入字符的位置,尾指针指的是键入字符的下一个位置。指针值只

在 30 到 60 之间变化,当两个指针相等时,缓冲区是空的。缓冲区可以存放 15 个键入字符,其中既有一个字节的 ASCII 码也有两个字节的扩展码。因此缓冲区必须在内存为每个字符提供两个字节。对于一个字节的 ASCII 码来讲,第一个字节是 ASCII 码,第二个字节是键扫描码。对于扩充码来讲,第一个字节是 ASCII0,第二个字节是扩展码。

根据上面所述,不难写出转换到不同输入方式的小程序,仍以 dBASE 为例:

\* PY. PRG 转换到拼音方式

```
SET ESCA OFF
POKE 1047,0 ;设置小写字母
POKE 1050,30 ;设置缓冲区首指针
POKE 1052,32 ;设置缓冲区尾指针
POKE 1054,0 ;设置 ASCII 码 0
POKE 1055,106 ;设置拼音方式的扩展码
```

\* XW. PRG 转换到 ASCII 码方式

```
SET ESCA OFF
POKE 1047,64 ;设置大写字母
POKE 1050,30
POKE 1052,32
POKE 1054,0
POKE 1055,109 ;设置 ASCII 码方式的扩展码
```

应用举例:

```
:
CLEA
@ 5,10 SAY '编号:' GET NO ;ASCII 码
DO XW
READ
@ 6,10 SAY '姓名:' GET NA ;汉字
DO PY
@ 7,10 SAY '学校:' GET SH ;汉字
READ
8,10 SAY '班级:' GET CL ;ASCII 码
DO XW
READ
:
```

采用上述方法,可以方便地在各种高级语言中实现各种不同输入方式的自动转换。

# C 语言在配平化学方程式中的应用

北京理工大学附中 王 晰

C 语言是一种目前很流行的语言,它既有很高的效率,又有清晰的程序结构和丰富的数据类型。它在实现结构化编程的同时又对编程者很少有严格的限制,使他们很容易表达自己的思想。这些优点使 C 语言很适合编制以逻辑判断为主的系统软件和人工智能程序。

配平化学方程式是中学化学中的一个重要问题。关于配平化学方程式有很多方法,但是都不适于用计算机来完成。然而,我们知道,每一个已经配平的化学方程式都遵守物质不灭定律,即每一个参加反应的原子,都必须包含在反应后生成的物质中。根据这一定律,通过比较反应前后的各类原子数目是否相等,就可以知道方程式是否已经配平,我们也就可以利用计算机速度快、适合循环重复的特点采用试凑法解决这个问题了。

本程序的优点是:程序执行中,根据屏幕提示直接输入各种未配平的化学方程式,程序执行后,也直接在屏幕上显示配平后的方程式,非常直观,而且符合人们的习惯。为了实现这一点,程序的第一部分是语法分析程序。它先从方程式中将各种不同物质的分子分离出来,再把每种物质分子分解为单个元素,然后存入结构 `lwz, rwz` 中。程序的第二部分是试凑部分。试凑程序对每一种可能的情况进行试验,如果反应前后各类原子数相等,则停止试凑,输出结果。试凑按系数由小到大进行,以保证输出最简结果。

程序使用方法十分简单。程序经 Turbo C 编译后即可运行。当屏幕上提示输入化学方程式的时候,就可以输入未配平的化学方程式。程序执行后便显示平衡的化学方程式。

例:

1. 输入  $\text{Cu} + \text{HNO}_3 = \text{CuN}_2\text{O}_6 + \text{NO} + \text{H}_2\text{O}$  ✓  
输出  $3\text{Cu} + 8\text{HNO}_3 = 3\text{CuN}_2\text{O}_6 + 2\text{NO} + 4\text{H}_2\text{O}$
2. 输入  $\text{NH}_3 + \text{O}_2 = \text{NO} + \text{H}_2\text{O}$  ✓  
输出  $4\text{NH}_3 + 5\text{O}_2 = 4\text{NO} + 6\text{H}_2\text{O}$
3. 输入  $\text{FeS}_2 + \text{O}_2 = \text{Fe}_2\text{O}_3 + \text{SO}_2$  ✓  
输出  $4\text{FeS}_2 + 11\text{O}_2 = 2\text{Fe}_2\text{O}_3 + 8\text{SO}_2$

注意:

① 大小写非常重要,当元素用两个字母表示时,后一个字母一定要小写,如  $\text{Cu}$ ,  $\text{Fe}$  等,否则程序将不能区分是一个元素还是两个元素。

② 本程序不处理括号,如  $\text{Cu}(\text{NO}_3)_2$  应写成  $\text{CuN}_2\text{O}_6$ 。

```
#include "stdio. h"
#include "string. h"
#include "stdlib. h"
#include "ctype. h"
```

```
#include "dos. h"
#define NL 3
#define YL 6
#define WL 20
#define FL 255
#define ZL 25
#define TRUE 1
#define FALSE 0
#define STEP 0
struct yuansu {
 char rname[NL];
 int xishu;
 int * zongji;
};
struct wuzhi {
 struct yuansu ys[YL];
 int ysshu;
}lwz[WL],rwz[WL];
char lzjname[ZL][NL],rzjname[ZL][NL];
int lzongji [ZL],rzongji[ZL];
int lwzshu,rwzshu;
int lzjs,rzjs;
int qs[WL * 2];

main()
{char fcs[FL];
setcbrk(1);
for(;;)
{printf("\nInput the chemical equation:");
gets(fcs);
yufa(fcs);
chushi();
shicou();
xwout();
}
}
yufa(char fcs[FL])
{char * p;
p = strpbrk(fcs, "=");
if(p == NULL) wrong("SYNTAX ERROR");
* p = NULL;
lwzshu = fenxi(fcs, &lwz);
rwzshu = fenxi(p+1, &rwz);
}
fenxi(char * fcs, struct wuzhi * wz)
{int i, ysshu = 0, wzshu = 0;
char * j = fcs;
fcs++;
for(i = 0; i < FL; i++, fcs++)
```

```

{if (isupper(* fcs) || * fcs == '+' || * fcs == NULL)
{char * n = fcs - 1;
if (! isdigit(* n))
{wz -> ys[ysshu]. xishu = 1;
n = fcs;
}
else{
for (; isdigit(* (n - 1)), n --)
wz -> ys[ysshu]. xishu = atoi(n);
}
strncpy(wz -> ys[ysshu]. name, j, n - j > NL -
1 ? NL - 1 : n - j);
if(fcs - j < NL)
{*(wz -> ys[ysshu]. name + (fcs - j)) = NULL;
}
ysshu ++;
if (ysshu > YL) wrong("YUAN SU TOO MANY");
j = fcs;
if(* fcs == '+' || * fcs == NULL)
{wz -> ysshu = ysshu;
ysshu = 0;
wz ++;
wzshu ++;
if(* fcs == NULL) break; /* END */
fcs ++;
j = fcs;
if(wzshu >= WL) wrong("WU ZHI TO MANY");
}
}
return(wzshu);
}
chushi()
{lzjs = csh(lwz, lwzshu, lzjname, lzungji, 0);
memcpy(rzjname, lzjname, sizeof(lzjname));
rzjs = csh(rwz, rwzshu, rzjname, rzongji, lzjs);
if (lzjs != rzjs) wrong("YUAN SU BU PI PEI");
}
csh(struct wuzhi * wz, int wzs, char zjn[][NL], int * zj, int
zjs)
{register i, j;
for(; wzs > 0; wzs --, wz ++)
{for(i = 0; i < wz -> ysshu; i ++)
{for(j = 0; j <= zjs && strcmp(zjn[j], wz -> ys[i].
name) != 0; j ++);
if(j > zjs)
{zjs ++;
j = zjs;
strcpy(zjn[j], wz -> ys[i]. name);
}
wz -> ys[i]. zongji = zj + j;
}
}
return zjs;
}
}

```

```

shicou()
{int jishu, i, zwzs = lwzshu + rwzshu;
register * pqs, * pqh = &qs[zwzs - 1];
for (i = 0; i < zwzs; i ++, qs[i] = 1)
for (jishu = 1; jishu < 20; jishu += STEP) /* jishu
为试凑次数 */
{testsc(jishu);
i = TRUE;
for(; i;)
{pqs = qs;
if(panduan()) return TRUE;
while (++ * pqs > jishu + STEP)
{ * pqs = 1;
pqs ++;
if (pqs > pqh)
{i = FALSE;
break;
}
}
}
}
panduan()
{memset(lzungji, 0, ZL);
memset(rzungji, 0, ZL);
tongji(lwz, qs, lwzshu);
tongji(rwz, qs + lwzshu, rwzshu);
return ! memcmp(lzungji, rzongji, sizeof(lzungji));
}
tongji(struct wuzhi * wz, int * qs, register wzs)
{register i;
for(; wzs > 0; wzs --, wz ++, qs ++)
{for(i = 0; i < wz -> ysshu; i ++)
* (wz -> ys[i]. zongji) += wz -> ys[i]. xishu * (* qs);
}
}
xwout()
{puts("\n");
xout(lwz, qs, lwzshu);
printf("=");
xout(rwz, qs + lwzshu, rwzshu);
puts("\n");
}
xout(struct wuzhi * wz, int * qs, int wzs)
{int i;
for(; wzs > 0; wzs --, wz ++, qs ++)
{if(* qs > 1)
printf("%d", * qs);
for(i = 0; i < wz -> ysshu; i ++)
{printf("%s", wz -> ys[i]. name);
if(wz -> ys[i]. xishu > 1)
printf("%d", wz -> ys[i]. xishu);
}
if (wzs != 1) printf("+");
}
}

```





# 如何利用 dBASE III 的 F1 功能键

江西教育学院微机室 (330029) 涂振宇

在 dBASE III 中 F2—F10 这些键可被用户动态定义,给我们带来很大的方便,唯独 F1 键被定义为 HELP,不能用于编程。本文介绍的方法可以使 F1 重新定义,可用于编程或在 MODI COMM 状态下输入定义字符串。

本方法的思想是修改 INT16H,当按下 F1 键时使其指向新的中断,而不被 dBASE 读取,这样在新的中断中,我们就可自由使用 F1 键了。

将下面程序汇编、连结、转换成 .COM 文件,在调 dBASE 前运行此程序,该程序即常驻内存。本例是把 F1 定义为 'MODI COMM',用户可自行改动,例如在程序中可借助本程序和 SET 命令将 F1—F10 定义为特殊字符,如半个汉字,这样配合 READ/GET 和 DO CASE 语句,我们可使用 F1 等功能键决定程序流向了。

```
codesg segment para 'code'
 org 100h
 assume cs:codesg,ds:codesg,es:codesg,ss:codesg
main proc near
begin: jmp start
endflag dw 0ffffh
old-int16 dd 0
f1 db 'modi comm',0
newint16: or ah,ah
 jz 11
 jmp dword ptr cs:old-int16
11: sti
 push bx
 push cx
 push dx
 push si
 push ds
 push cs
 pop ds
 cmp endflag,0ffffh
 jnz 13
 pushf
 call dword ptr cs:old-int16
 cmp al,0
 jnz exit
 cmp ah,3bh
 jnz exit
 mov si,offset f1
 mov endflag,si
13: mov si,endflag
```

```
 mov al,[si]
 mov ah,0
15: cmp al,0
 jz 16
 inc si
 mov endflag,si
 jmp exit
16: mov endflag,0ffffh
exit: pop ds
 pop si
 pop dx
 pop cx
 pop bx
 iret
start: mov ax,3516h
 int 21h
 mov word ptr old-int16,bx
 mov word ptr old-int16+2,es
 mov ax,2516h
 push cs
 pop ds
 lea dx,newint16
 int 21h
 mov ax,3100h
 mov dx,50h
 int 21h
main endp
codesg ends
end begin
```

## 一点补充:

今年第 8 期所刊“CEC—I 自造字点阵生成程序”生成的是与机内硬汉字库相同的正排点阵(即每行点阵按高位在左、低位在右的顺序排放),而有的调取点阵的程序要求提供的是倒排点阵(如本刊今年第 4 期《也谈全功能造字》一文的程序),否则会将左右两半字印反,若要生成倒排点阵,可将本造字程序 160 行中的“FOR K=7 TO 0 STEP -1”改为“FOR =0 TO 7”、将 280 行中的“U=8+8 \* T-H”改为“U=H-1-8 \* T”即如愿。

另外,所造的字亦可表以区位码,字码的具体对应完全由所使用的调字程序决定,如调字程序规定所调取自造字均在第 10 区,则本造字程序生成的第 1、2、……号字的区位码依次为 1001、1002、……。

该文作者:汤永进

# CCBIOS 2.13 稿纸方式打印

成都热电厂 (610051) 宋捷

我们经常希望文本文件能按稿纸方式打印, 现介绍一个在 CCBIOS 2.13 系统下, 将文本文件按稿纸方式打印的简单方法:

1. 在 FoxBASE 下建立数据库 GZ.DBF, 库结构见程序一, 该库仅有一个长为 40 的字符型字段 G;

2. 用 WS 建一个 FoxBASE 程序 GZ.PRG, 程序 GZ.PRG 结构整齐, 简单实用, 打印时自动分页, 并标页码, 由于程序所设的中间库 GZ.DBF 只设了一个宽度为 40 的字段, 且程序划框线时对于重复的框线使用了 REPL() 函数, 因此相应的简化了程序, 并提高了程序在打印过程中的运行速度。

3. 使用时应先利用 WS.XE 等字处理软件将文本的内容输入到一个文本文件中, 在输入时, 为防止在以后程序处理中字符的错位或者某些字符被截断, 应注意每行只能输入 20 个汉字, 并且在文本输入汉字或全角字符时, 汉字或全角字符的前半个字必须在奇数列, 后半半个字必须在偶数列, 标点符号最好用全角方式输入, 两个阿拉伯数字或英文字母应占一个汉字宽度;

4. 在运行程序 GZ.PRG 前, 应打开打印机, 装好 80 行打印纸, 程序运行时只要出现“请输入稿纸方式打印的文件名:”的提示, 便可输入后缀为 TXT 的文本文件名, 此时打印机就会在打印纸的正中按稿纸方式打印出相应的文本文件了。

程序在 AST486、HP386、GW386 机和 AR3240 打印机上实现, 操作系统为 DOS 3.3, 汉字系统为 CCBIOS 2.13H。

程序一:

GZ.DBF 数据库结构:

Field	Field Name	Type	Width	Dec
1	G	Character	40	
**	Total	**	41	

程序二:

```
C> TYPE GZ.PRG
*** 稿纸方式打印 ***
set talk off
set safe off
clear
accept '请输入稿纸方式打印的文件名:' to filename
use gz
zap
appe from &filename sdf
go top
n=1
do while .not.eof()
```

```
set devi to prin
line=1
@ prow(),1 say CHR(96)+CHR(64)+CHR(65)+CHR(38)+
"15"+CHR(91)+"8"+CHR(96)
do while line<=20
if .not.eof()
@ prow()+1,1 say "┌" + repl("─",19) + "┐"
@ prow()+1,1 say '|'
s=1
do while s<=20
@ prow(),pcol() say subs(g,(2*s-1),2)+"|"
s=s+1
enddo
@ prow()+1,1 say "└" + repl("─",19) + "┘"
skip
else
@ prow()+1,1 say "┌" + repl("─",19) + "┐"
@ prow()+1,1 say "|" + repl("─",19) + "|"
@ prow()+1,1 say "└" + repl("─",19) + "┘"
endif
line=line+1
enddo
@ prow()+3,37 SAY '第'+str(n,2)+'页'
n=n+1
@ prow()+29,1 say ' '
set devi to scre
enddo
return
```

## 微机屏幕的打印和放大

武汉铁路分局电子计算技术中心 谭人杰

本文介绍将屏幕缓冲区的显示数据转换成打印数据以及放大打印方法, 并提供已经实现的屏幕打印源程序。

### 一、直接存取屏幕显示缓冲区的原理和方法

屏幕图形由显示缓冲区中的 640×200 个二进制位表示, 在 CGA 中分辨率彩色图形方式下, 每个象素点用两个二进制位表示, 这两个二进制位取值 0 时表示背景色(无点), 取值为 1、2、3 时表示不同彩色(有点), 每个字节可表示四个象素点, 屏幕上每行扫描线用 80 字节表示 320 个象素点。

屏幕纵向共有 200 行扫描线,100 行偶数扫描线 (0,2,4,...,198) 占用以 B8000H 为起始地址的 8000 字节空间,100 行奇数扫描线 (1,3,5,...,199) 占用以 BA000H 为起始地址的 8000 字节空间。

要直接存取某一象素点,可按以下算法编程:

1. 计算该点所在扫描线的 80 字节的起址;
2. 计算在这 80 字节中该点所在的字节;
3. 计算在这一字节中该点所在的两位二进制数;
4. 对这两位二进制数设置彩色代码或赋象素值。

## 二、打印机以图形方式打印屏幕的原理和方法

24 针打印机以图形方式打印时,将 24 支针排成一列,每支针由一个二进制位控制,二进制位取值 0 时,针头不进针;取值 1 时,针头进针打点。24 支针由三个字节控制,对于 M-1724 打印机,每行可打印 2176 列的 24 针点,即  $2176 \times 3$  个字节表示的一行图形数据。

为了与图形打印方式对应,将屏幕以 24 行扫描线为单位转换成一行图形打印数据,重复八次将 200 线的显示数据全部转换成打印数据,整个屏幕图形就可以在打印机上打印出来。每次取 24 行扫描线时,需按奇偶行分别在奇数行扫描线缓冲区和偶数行扫描缓冲区取数。由于是两个二进制位表示一个象素点的彩色图形方式,所以每一打印列实际对应屏幕上两列二进制位,因此每次要在显示缓冲区取两列数据转换成一行打印数据后再发送到打印机;也就是说,要将显示缓冲区中表示一个彩色象素点的两位二进制数转换成打印缓冲区中表示黑白打印点的一位二进制数。转换方法是根据显示缓冲区中代表一个象素点的两位二进制数的取值来决定,如果取值 0,则打印针头对应的数据位赋值 0;如果取值 1、2、3,则打印针头对应的数据位赋值 1。

## 三、屏幕放大打印

知道了打印屏幕的原理,就可以按比例将屏幕图形放大,如果要求横向放大两倍,只需将屏幕上转换后的每列 (24 点) 数据连续送两次打印机,形成两列打印数据;如果要求纵向放大两倍,则需将屏幕上每个点转换成与打印针对应的纵向两个点;也就是说,将屏幕上纵向 12 个象素点转换成一行打印数据的 24 个点。以此类推,可以编写扩大四倍和扩大八倍的屏幕放大打印程序。

## 四、程序实例

由于篇幅所限,这里只提供按原幅打印屏幕的源程序,其思路是对屏幕上的 200 行扫描线,每次取 24 线转换成 24 针打印机对应的一行数据。并以图形方式打印。主程序首先将打印机置成图形打印方式,并将行宽置成 19/120 英寸,主程序调用过程 TRAN\_BYTEi,将显示缓冲区中纵向 8 行扫描线的一列象素转换成一字节打印数据,对屏幕上每 24 行扫描线的一列象素,主程序调用三次子程序,并将返回的三字节依次送打印机,形成控制 24 根针的一列打印数据。TRAN\_BYTEi 调用过程 TRAN\_BIT,将屏幕上指定位置的纵

向 8 个象素依次转换成一字节中对应位的图形打印数据。

打印机为 M-1724,编程语言是 TURBO PASCAL 3.01A。

以上介绍的原理和方法,适用于各种类型的微机和打印机,对不同的显示卡,只要知道显示缓冲区的地址以及扫描线的排列方式,就可以直接访问屏幕上的任意点。对不同的打印机,只需修改“置图形打印方式”命令和“置打印行宽”命令。

例如 EPSON 1000K 系列打印机的这两条命令为

```

ESC * 38 n1 n2
ESC 3 24
{SCRNDUMP.PAS---dump screen to M1724,line space 19/
120 inch}
PROGRAM SCRNDUMP (INPUT,OUTPUT);
VAR
 DATA:BYTE;
 LINE,ROW,COL:INTEGER;

PROCEDURE TRAN_BYTEi (I,COL:INTEGER;VAR DATA:
BYTE);
VAR
 START_ROW:INTEGER;
PROCEDURE TRAN_BIT (ROW,COL:INTEGER;VAR DA-
TA:BYTE);
CONST
 VRAM_BASE = $B800; {Segment address of video buffer}
TYPE
 BIT=INTEGER;
 BIT2=INTEGER;
VAR
 CF:BIT;
 CF2:BIT2;
 BASE,ROW_START,OFF_ROW,OFF_COL:INTEGER;
 OBJ_BYTE,OBJ_BIT,MARK:BYTE;
BEGIN
 {calculate start address of the 80 byte line}
 CF := ROW MOD 2;
 OFF_ROW := ROW SHR 1;
 IF CF = 0 THEN BASE := $0000
 ELSE BASE := $2000;
 ROW_START := BASE + OFF_ROW * 80
 CF2 := COL MOD 4; {ith two bit binary}
 OFF_COL := COL SHR 2; {column divided by 4 to get ith
byte in the 80 bytes}
 OBJ_BYTE := MEM[VRAM_BASE;ROW_START+
OFF_COL]; {take the ith byte}
 MARK := $C0 SHR (2 * CF2);
 OBJ_BIT := MARK AND OBJ_BYTE; {take the ith two
bit binary}
 IF OBJ_BIT <> 0 THEN BEGIN {set 1 to the bit of print
data}
 MARK := $80 SHR (ROW_START-ROW)
 DATA := DATA OR MARK;
 END;
```

```

END;
BEGIN
 DATA := $00;
 START_ROW := 8 * 1;
 FOR ROW := START_ROW TO START_ROW + 7 DO
 TRAN_BIT(ROW, COL, DATA);
END;
BEGIN
 WRITE(LST, CHR(24)); {Initialize printer}
 WRITE(LST, CHR(27), 'J', CHR(19)); {Set line space}
 FOR LINE := 0 TO 7 DO BEGIN
 WRITE(LST, CHR(27), '4', CHR(1), CHR(64));
 (Set bit graphics, 1 * 256 + 64 = 320 columns)
 FOR COL := 0 TO 319 DO BEGIN {column count}
 TRAN_BYTE(LINE * 3 + 0, COL, DATA);
 WRITE(LST, CHR(DATA));
 TRAN_BYTE(LINE * 3 + 1, COL, DATA);
 WRITE(LST, CHR(DATA));
 TRAN_BYTE(LINE * 3 + 2, COL, DATA);
 WRITE(LST, CHR(DATA));
 END;
 WRITE(LST, CHR(13), CHR(10));
 END;
END.
=====

```

## 1992 年全国青少年信息学(计算机)竞赛试题

### 第一轮 试题 1.

把一篇文章按要求排版。

文章的输入方式为:由键盘输入一个以回车符结束的文章(最大长度 2000 个字符)。

排版时以单词为基本单位。单词由不含空格的任意字符组成,是长度小于 20 个字符的串。空格符是分隔单词的唯一字符,在输入时连续的空格符在处理时应首先化简为单个空格符。

在排版前应先输入排版后每行的字符数 N,排版后将整理好的文章按行输出。输出时应保证不将一个完整的单词截断,并且要求输出的总行数最小。

将每个不足 N 个字符的行用空格符补足,填充空格符的方式有以下三种:

1. 将填充的空格符置于每行的末尾,并要求每行的起始为单词。
2. 将填充的空格符置于每行的起始,并要求每行的末尾为单词。
3. 将填充的空格符尽可能平均分配在每行中,并保证每行的起始和末尾均为单词。试编程对输入的一篇文章分别完成上述三个要求。

### 第一轮试题 2

由英文字母和符号  $\sim$ 、 $*$ 、 $+$ 、 $()$  组成逻辑表达式,英文字母表示变量,变量有两种可能的取值, False 或 True;  $\sim$ 、 $*$ 、 $+$  分别代表逻辑运算的非、与、或。运算的优先依次为  $()$ ,  $\sim$ ,  $*$ ,  $+$ 。括号  $()$  可改变表达式的运算次序,且可以嵌套。

逻辑“非”运算的公式如下表:

A	~A
True	False
False	True

逻辑“与”和逻辑“或”的运算公式如下表

A	B	A * B	A + B
False	False	False	False
False	True	False	True
True	False	False	True
True	True	True	True

两个逻辑表达式等价,并且仅当两个公式中相同名字的变量取任何一组值时两个公式的值都相同。如:

- $A * (B + C)$       与     $A * B + A * C$       等价
- $A * (\sim A + B)$     与     $A * B$             等价
- $(\sim A + A) * B + C$     与     $B + C$             等价
- $A * B + A * \sim B$     与     $A$                 等价

而:

- $A + B$                 与     $A * B$             不等价
- $A * B + \sim C$         与     $A * E + \sim F$     不等价

现要求你编程解决下列问题:

### 任务 1:

用键盘输入一个逻辑表达式,判断这个表达式的合法性;

### 任务 2:

将键盘输入的表达式化简,化简的表达式形式为  $a_1 * a_2 * \dots * a_n + b_1 * b_2 * \dots * b_m + \dots + x_1 * x_2 * \dots * x_l$

其中  $a_i, b_j, \dots, x_k (i = 1, 2, \dots, n; j = 1, 2, \dots, m; k = 1, 2, \dots, l)$  表示一个变量或一个变量的逻辑非。

### 任务 3:

将任务 2 中的化简的表达式优化为最简形式。所谓最简有如下两个条件:

- (1) 表达式中的“+”号最少;
- (2) 满足(1)的条件下“\*”号最少。

### 第二轮试题 1

无根树与通常所说的树(有根树)很相似,它包含

有节点和枝,但不含有根。无根树节点间只有相邻关系,而不存在父子节点的关系。如图 1 所示,是一棵有 7 个节点的无根树;以图 1 的 A 为根节点得到图 2 所示的有根树,以图 1 的 B 为根节点得到图 3 所示的有根树,但从无根树的角度看,图 1、图 2、图 3 是结构相同的无根树,同时无根树的结构与节点的名称无关。

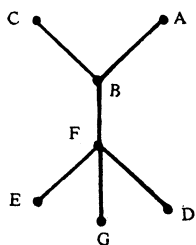


图 1

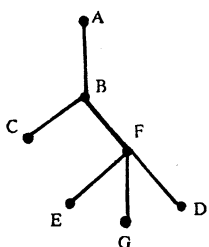


图 2

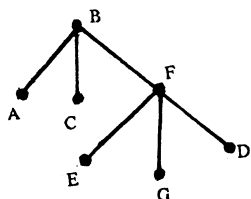


图 3

有根树可以以字符串的形式表示,其递归表示方法为:

根节点(子树 1 子树 2 子树 3……)

如图 2、图 3 的有根树可分别表示为 A(B(CF(EGD)))和 B(ACF(EGD))。需要注意的是,由于子树的表示顺序可以不同,所以一棵有根树可以有多种表示方法,如图 3 又可表示为 B(F(EGD)CA)或 B(ACF(DEG))等。

表示无根树时,可以以它的任一节点为根节点,将其看作有根树,从而可以利用有根树的字符串表示形式来表示无根树。

任务 1:

由键盘读入一个字符串表示的无根树,无根树的各节点的名称用互不相同的大写英文字母表示。由用户输入一个节点的名称,程序应能够输出一种以该节点为根节点的字符串形式。

程序输出无根树的字符串形式时,各个节点的名称无关紧要,所有节点都以 P 表示,以后的各种输出也采用这种方式。

例如,用户输入无根树的字符串形式:(A(B(CD(EF))))

指定的根节点为:D

程序应能输出:

P(P(P(P)P)) P(P(P(P)P)) P(PPP(P))中的任意一种即可。

任务 2:

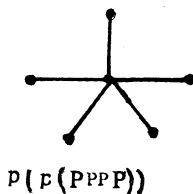
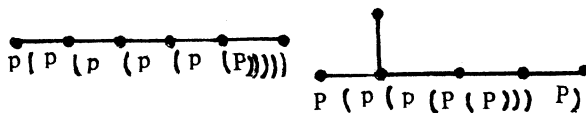
输入两个串表示的无根树,判断其结构是否一样。注意与节点名称无关,只考虑结构。

任务 3:

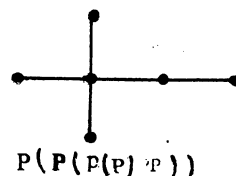
输入无根树的总枝数 N(1≤N≤11),输出所有枝数为 N 的互不相同的无根树,并记录总数。以字符串形式输出:

例如,N=5 时,共有 6 种不同结构的无根树,如下所示:

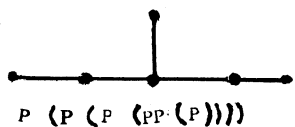
注意:各种树结构的字符串表达形式不唯一。



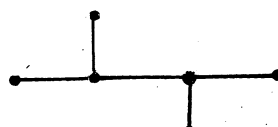
P(P(PPPP))



P(P(P(P)P))



P(P(P(P(P))))



P(P(PP(PP)))

### 第二轮试题 2

某机要部门安装了电子锁。M 个工作人员每人发一张磁卡,卡上有开锁的密码特征。为了确保安全,规定至少要有 N 个人同时使用各自的磁卡才能将锁打开。现在需要你计算一下,电子锁上至少要有多少种特征,每个人的磁卡上至少要有几个特征。如果特征的编号以小写英文字母表示,将每个人的磁卡的特征编号打印出来。要求输出的电子锁的总特征数最少。

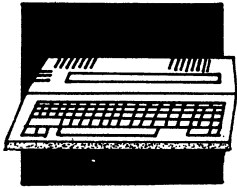
为了使问题简单,M 与 N 的上下限为

3≤M≤7, 1≤N≤4

M 与 N 由键盘输入,工作人员的编号用 1#, 2#, …等。

例如 M=3, N=2, 则电子锁上要有三种特征。每个人的磁卡上要有二种特征。

(此文由清华大学计算机系吴文虎教授提供)



## 学习机之友

# Applesoft BASIC 子程序的递归调用

韶关教育学院(512026) 陈继良 广东北江中学 丘文

一般认为,在 Applesoft BASIC 中,子程序是不能递归调用(即自己调用自己)的,如

[程序一]

```
20 GOSUB 20
]RUN
? Out of memory in 20
```

事实上并非如此,请看

[程序二]

```
10 I=0
20 I=I+1;PRINT I,
30 GOSUB 20
]RUN
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
```

? Out of memory in 20

可见,子程序是可以递归调用的,只是递归的次数一般不能超过 24,这与子程序嵌套的最大层数是一致的。通过剖析 Applesoft 可知,执行 GOSUB 语句时,计算机会把有关数据(GOSUB 后续语句的位置)进栈;遇 RETURN 语句时,栈顶数据退栈。子程序嵌套(包括递归调用)层数的限制是由于堆栈空间有限而引起的,子程序嵌套时,如果进栈的数据太多而又未能及时退栈,就会导致空间溢出而出错。对于子程序的非递归嵌套,一般极少超过 24 层,所以栈空间不足的矛盾并不突出;而在子程序的递归调用中(如程序一、二),就很容易造成栈空间溢出。因此,一般书刊都不提子程序的递归问题,甚至干脆说不能递归。

根据上面的分析,只要把次数控制在 24 以内,子程序完全是可以递归的。但在实际的递归问题中,递归的次数往往在 24 次以上。例如

[程序三]

```
10 INPUT "S,C=";S,C;X=140;Y=90
20 HGR2;HCOLOR=3;GOSUB 100
30 END
100 HPLLOT X,Y TO X,Y-S TO X+S,Y-S TO X+S,
 Y TO X,Y
110 S=S-C;IF S<=0 THEN RETURN
120 GOSUB 100
130 RETURN
]RUN
```

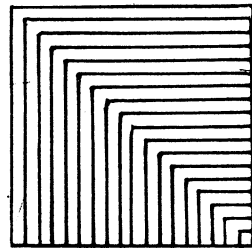
S,C=90,4

(结果见图一)

]RUN

S,C=90,3

? Out of memory error in 100



图一

S 为正方形的边长,C 为相邻正方形的边长之差,S=90,C=3 时,递归次数已超过 24 次,栈空间溢出。因此,要在非结构化的 Applesoft BASIC 中真正实现递归算法,必须先解决子程序的无限递归问题!

在子程序的递归调用中,进栈的数据中有很多是重复的。如果能设法把多余的数据及时弹出栈外,便可解决栈空间小的矛盾,从而实现子程序的无限递归。除了 RETURN 语句外,还有一个 POP 语句具有退栈功能,特别地,POP 语句退栈后并不返回,而是继续执行后续语句,只要我们设置适当的退栈条件,必要时通过执行 POP 语句退栈,便可实现子程序的无限递归。例如,在程序三中加入

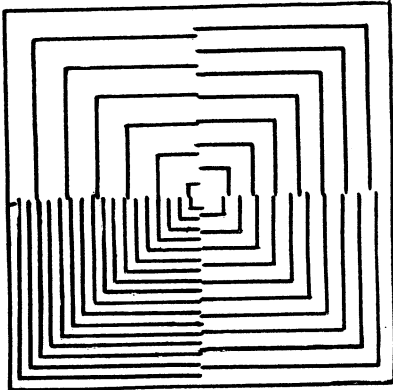
```
15 CHEN=1
105 IF CHEN>1 THEN POP
115 CHEN=CHEN+1
```

便可完成 S 和 C 为任意值时的递归过程,只有这样,程序三才具有一定的实用价值。其中的“CHEN>1”即为退栈条件,程序执行过程中栈的变化情况如下:执行 20 行的 GOSUB 100 时,数据一(即 20 行 GOSUB 100 后一语句的位置)进栈;因为 15 行设置 CHEN=1,所以 105 行不执行 POP,当 115 行 CHEN 加 1 后由 120 再 GOSUB 100,首先是数据二(即 120 行 GOSUB 100 下一语句的位置)进栈,然后转 100 行;由于这时 CHEN>1 即退栈条件满足,所以执行 POP 语句,数据二被弹出栈外而作废。如此反复,直到 110 行的递归边界“S<=0”被满足而执行 RETURN 退栈返回。由于栈中有效的数据始终只有数据一,所以程序这时将返回 30 行正常结束。这样,通过设置退栈条件及时把多余数据弹出栈外,解决了令人讨厌的栈空间溢出问题,原则上实现了子程序的无限递归。由以上分析可知,这时

130 行形同虚设(不会被执行);如果要保留并执行 130 行,则退栈条件可改为“CHEN>N”(N 为 2—22 之间的整数。可见,退栈条件可根据具体情况灵活设置,这是尾递归的情况,下面在程序三的基础上,给出通过子程序的非尾递归调用画出图二的程序

[程序四]

```
10 INPUT "S,C,D=";S,C,D;X=140;Y=90
15 CHEN=1;S0=S;X0=1;Y0=-1
20 HGR2;HCOLOR=3;GOSUB 100
30 END
100 HPLLOT X,Y TO X,Y+S*Y0 TO X+S*X0,Y+S*Y0 TO X+S*X0,Y TO X,Y
105 IF CHEN>2 THEN POP
110 S=S-C;IF S<=0 THEN CHEN=2;S=S0;C=C+D;RETURN
115 CHEN=CHEN+1
120 GOSUB 100
130 X0=-1;Y0=-1;GOSUB 100
140 X0=-1;Y0=1;GOSUB 100
150 X0=1;Y0=1;GOSUB 100
160 RETURN
]RUN
S,C,D=90,5,3
```



图二

## APPLE 的快速排序

北京航空航天大学机械厂(100083) 张亭

在 APPLE II 或兼容机中输入程序 1—1 和 1—2 并试运行,会令人感到一种速度的享受:排序 N=1000 的数组只须 2 秒,排序 N=3000 的数组只须 8 秒。这就是有名的快速排序算法。

通过对程序 1—1 反汇编可以看到,6502 机器语言程序中使用了递归调用技术,这就是快速排序算法天生具备的特性。程序 2—2 是用 BASIC 语言编制的快速排序程序,由于 BASIC 不允许递归调用,这里采用了一种可称为“层次计数”的编程技巧,程序中变量 C%就是层次计数器。程序 2—1 是变量交换子程序,

取自《APPLE II 彻底研究(二)》。但即便采用了这种办法,BASIC 的快速排序比汇编仍然要慢 80~90 倍。

那么既然有了风驰电掣的机器语言快速排序,为什么还要 BASIC 慢吞吞的程序呢?这是因为高级语言也有其优势,除易于阅读、理解外,还方便修改和移植。只要把“&”改为 SWAP,就可运行于 16 位以上的高档微机,只要把 50 行和 80 行改为关键字的比较,把 60 行和 90 行改为成组信息的交换,就实现了对多维变量或记录的排序。

笔者将其运行于 10MH 的 286 微机,当数组的 N=1000 时排序时间为 25 秒。可见,随着微机主频率的提高,BASIC 的排序仍然是有意义的。

不过,快速排序对已经有序或基本有序的情况来说效率却是很低的,而反序对快速排序来说则几乎是一个灾难。除了这类特殊情况以外,快速排序也不是稳定排序,即:不能保证关键字相等记录的最初次序。

程序 1—1

```
0249— 20 E3 DF 85 40 84 41
0250— 20 BE DE 20 67 DD 20 52
0258— E7 85 5F 98 0A 26 5F 0A
0260— 26 5F 65 50 85 5E A5 5F
0268— 65 51 85 5F A5 40 65 5E
0270— AA A5 41 65 5F D0 18 E4
0278— 40 D0 04 C5 41 F0 03 20
0280— 97 02 18 8A 69 05 85 40
0288— 90 02 E6 41 68 AA 68 E4
0290— 40 D0 04 C5 41 F0 30 86
0298— 5E 85 5F 48 8A 48 A5 41
02A0— 85 61 A5 40 85 60 38 A5
02A8— 5E E9 05 85 5E B0 02 C6
02B0— 5F C5 60 D0 07 AA A5 5F
02B8— C5 61 F0 BB 20 FD EA 20
02C0— B6 EB B0 E3 A0 04 B3 60
02C8— B1 5E 91 60 8A 91 5E 88
02D0— 10 F4 18 A5 60 69 05 85
02D8— 60 90 02 E6 61 C5 5E D0
02E0— 07 AA A5 61 C5 5F F0 8F
02E8— 20 FD EA 20 B6 EB B0 E2
02F0— A0 04 B3 60 B1 5E 91 60
02F8— 8A 91 5E 88 10 F4 30 A6
```

程序 1—2

```
10 HOME;INPUT"N=";N;C=1000
20 DIM A(N),B(N);FOR I=0 TO N:A(I)=RND(1)*C;B(I)=A(I);NEXT
40 P=0;Q=N+1
50 PRINT "PUSH ANY KEY";GET A$
60 CALL 585B(P),Q
70 PRINT CHR$(7);PRINT B(0),B(N/2),B(N)
80 GET A$;FOR I=0 TO N:PRINT A(I),B(I);NEXT;END
```

程序 2—1

```
02C6— 20 E3
02C8— DF 85 85 84 86 A5 81 48
02D0— A5 82 48 20 BE DE 20 E3
02D8— DF 68 45 82 30 1F 68 45
```



```
02E0— 81 30 1A A0 02 24 81 30
02E8— 10 24 82 30 02 A0 04 B1
02F0— 85 48 B1 83 91 85 68 91
02F8— 83 88 10 F3 60 4C 76 DD
```

```
03F5— 4C C6 02
```

```
程序 2—2
```

```
10 GOTO 135
```

```
30 Z%(C%)=DW%:C%=C%+1
```

```
35 DS%=S%
```

```
40 DW%=DW%-1,IF DW%=DS% THEN 100
```

```
50 IF A(DS%)>=A(DW%) THEN 40
```

```
60 & A(DS%),A(DW%)
```

```
70 DS%=DS%+1,IF DW%=DS% THEN 100
```

```
80 IF A(DS%)>=A(DW%) THEN 70
```

```
90 & A(DS%),A(DW%);GOTO 40
```

```
100 IF S% < DS% THEN 30
```

```
110 S%=S%+1;DW%=Z%(C%-1);IF S% < DW% THEN 35
```

```
120 C%=C%-1,IF C%>0 THEN 110
```

```
130 PRINT CHR$(7);GET A$;GOTO 150
```

```
135 HOME;INPUT "N=";N%;DW%=N%+1;C%=1000;DIM A(N%),Z%(20)
```

```
136 FOR I=0 TO N%;A(I)=RND(1)*C%;NEXT;C%=0;PRINT CHR$(7);GET A$;GOTO 30
```

```
150 FOR I=0 TO N%;PRINT A(I);NEXT;END
```

# 百年公历— 农历互查程序

陕西中药研究所 刘安军

由已知公历查相应农历、或由已知农历查相应的公历，采用计算的方法很难实现，一般的办法是查表，例如查日历卡或查万年历。在计算机上如果采用建表的方法，要查许多年范围内的日历，数据量相当大，非常困难。本程序采用数据压缩和计算相结合的办法，可以迅速得到百年内某月的公历—农历对照月历，实用、方便、准确无误。

## 一、基本原理

如果知道公历某月 1 日的农历月份、日期，又知道当月和下月农历的大、小，就可以推算出该月的对照月历。而计算下月公历 1 日的农历日则可以采用公式：

下月公历 1 日的农历日 = 本月公历 1 日的农历日 + 本月公历天数 - 本月农历天数

所以，只要得到每年公历 1 月 1 日对应的农历月份、日期，以及该年农历的大、小月和闰月的信息，就可以推算出该年任何一个月对照月历。但是，公历每年跨接的农历月有 13 或 14 个月，大、小月又无规律可循，加上闰月情况，公历 1 月 1 日的农历月份、日期等，数据量仍较大。

本程序采用数据压缩的办法，经过编码，每年采用一组八位十进制数，实现了以上信息的传递。表一列出了各位数的意义，并以 1992 年为例。表中“大”为大月 30 天，“小”为小月 29 天。从左起第一至第四位，每位用 1~8 八个数字表示三个月的大、小月信息，四位共 12 个月。编码情况见表二。第五位是闰月情况，当该位为“0”时，该年平；为“2”至“9”时，该年闰，数字是几就是闰几月；1984 年闰十月是特例，用“1”来表示。第六位用 1~4 四个数字编码，每个数字代码意义见表三。

表中，C 表示公历 1 月 1 日农历滞后几个月，S 表示农历第 13 个月的大、小月信息。第七、八位，表示农历对应公历 1 月 1 日的日期。

## 二、使用说明

程序运行后，如要查公历某年某月的农历，则输入公历年、月，屏上显示出该月公历—农历对照月历，农历每月的 1 日为该月的月份，如果是闰月则为“闰”字。如果要查某年农历某月某日的公历日，则输入该年该月或下个月的数字即可查到。打印时应设定行宽为 17 个汉字等有关参数。如要改变程序的查询年限范围，一种办法是直接更换压缩数据。另一种办法是扩大 T 数组，修改程序个别语句，可以直接查数百年甚至上千年的日历，只要不超出内存的变量允许最大范围。

## 三、程序说明

### (一) 变量

T 数组：一百年压缩数据，以变量 F 做数组下标；S 数组：农历该年每月天数（公历跨接共 13 个月），以变量 V 做下标；R\$ 数组：农历每月 1 日显示的月份名，变量 N 做中间变量，最后一次做 R\$ 数组的下标；SS\$ 数组：农历“干”名，以变量 BB 做下标；WW\$ 数组：农历“支”名，以变量 CC 做下标；A：待查年；B：待查月；C：滞后月数；L：闰月信息；Q：农历日；P：公历月的天数；E\$：该年压缩数据；S(14)：处理 1919、1938 年公历跨月历 14 个月特例；M、I、K、M\$、F\$：中间变量。

### (二) 部分语句说明

20~90 句，读数据；  
100~110 句，取该年压缩数据；  
120~130 句，计算农历干支数组下标；  
140~330 句，取解码后 13 个月每月天数、滞后月数、闰月信息和农历初始日；  
340~450 句，计算待查月份公历 1 日的农历日；  
480~640 句，输出公历和农历，每 10 天为一行，公历、农历每天上下对应，直至公历该月最后一日为止；  
690~730 句，输出农历为“1”日时的月份名，闰月时为“闰”。

表一

意义	第一位			第二位			第三位			第四位			第五位		第六位		第七八位
	1	2	3	4	5	6	7	8	9	10	11	12	闰月信息		C	S	农历日期
内容	大	大	小	大	大	小	小	大	小	小	大	小	该年平		2	大	公历1月1日
1992年编码	2			2			6			6			0		3		27

表二

编 码	月 份		
	1	2	3
	4	5	6
	7	8	9
	10	11	12
1	大	大	大
2	大	大	小
3	大	小	大
4	大	小	小
5	小	大	大
6	小	大	小
7	小	小	大
8	小	小	小

表三

	C	S
1	1	大
2	1	小
3	2	大
4	2	小

```

10 DIM T(100),S(20),R$(15),SS$(10),WW$(12)
20 F$=RIGHT$(M$,2):F=VAL(F$)
30 FOR I=0 TO 99:READ T(I):NEXT
40 FOR I=1 TO 15:READ R$(I):NEXT
50 FOR I=0 TO 9:READ SS$(I):NEXT
60 FOR I=0 TO 11:READ WW$(I):NEXT
70 PRINT CHR$(4)"PR#3":HOME:HGR2
80 INPUT"输入年月: ",A,B:M$=STR$(A)
90 IF A<1900 OR A>1999 OR B<1 OR B>12 GOTO 80
100 F$=RIGHT$(M$,2):F=VAL(F$)
110 E$=STR$(T(F))
120 BB=A-INT(A/10)*10
130 CC=A-INT(A/12)*12
140 I=1:V=1:N=1:S(14)=20
150 M$=MID$(E$,N,1):K=VAL(M$)
160 IF K=1 THEN S(I)=30:S(I+1)=30:S(I+2)=30:
 GOTO 240
170 IF K=2 THEN S(I)=30:S(I+1)=30:S(I+2)=29:
 GOTO 240

```

```

180 IF K=3 THEN S(I)=30:S(I+1)=29:S(I+2)=30:
 GOTO 240
190 IF K=4 THEN S(I)=30:S(I+1)=29:S(I+2)=29:
 GOTO 240
200 IF K=5 THEN S(I)=29:S(I+1)=30:S(I+2)=30:
 GOTO 240
210 IF K=6 THEN S(I)=29:S(I+1)=30:S(I+2)=29:
 GOTO 240
220 IF K=7 THEN S(I)=29:S(I+1)=29:S(I+2)=30:
 GOTO 240
230 IF K=8 THEN S(I)=29:S(I+1)=29:S(I+2)=29:
 GOTO 240
240 I=I+3:IF I<12 THEN N=N+1:GOTO 150
250 M$=MID$(E$,6,1):K=VAL(M$)
260 IF K=1 THEN S(13)=30:C=1:GOTO 300
270 IF K=2 THEN S(13)=29:C=1:GOTO 300
280 IF K=3 THEN S(13)=30:C=2:GOTO 300
290 IF K=4 THEN S(13)=29:C=2:GOTO 300
300 IF C=1 THEN N=2:GOTO 320
310 N=1
320 M$=MID$(E$,5,1):L=VAL(M$):IF L=1 THEN
 L=10
330 M$=MID$(E$,7,2):Q=VAL(M$)
340 FOR I=1 TO 12
350 IF I=4 OR I=6 OR I=9 OR I=11 THEN P=30:GOTO
 420
360 IF I<>2 THEN P=31:GOTO 420
370 IF A/100=INT(A/100) GOTO 400
380 IF A/4=INT(A/4) THEN P=29:GOTO 420
390 GOTO 410
400 IF A/400=INT(A/400) THEN P=29:GOTO 420
410 P=28
420 IF I=B THEN 460
430 Q=Q+P-S(V):V=V+1:N=N+1:IF L<>0
 THEN IF V-2=L THEN N=N-1
440 IF Q>S(V) THEN Q=Q-S(V):V=V+1:N=N+1
450 NEXT
460 PRINT A;"年";B;"月(农历)";SS$(BB);WW$(CC);
 "年";:IF L<>0 THEN PRINT"闰";L;"月)":GOTO
 480
470 PRINT "(")
480 M=0:K=0
490 PRINT"公历";
500 FOR I=1 TO P
510 IF I<10 THEN PRINT I;" ";:GOTO 530
520 PRINT I;" ";
530 IF I=10 OR I=20 OR I=30 GOTO 560

```

```

540 NEXT
550 PRINT
560 PRINT "农历";
570 K=K+1
580 IF Q=1 GOTO 700
590 IF Q>S(V) THEN 690
600 IF Q<10 THEN PRINT Q;" ";GOTO 620
610 PRINT Q;" "
620 Q=Q+1;M=M+1;IF M=>P GOTO 650
630 IF K=10 THEN K=0;HTAB1;PRINT"公历";NEXT
640 GOTO 570
650 PRINT;INPUT"还查吗? (Y/N)";M$
660 IF M$="Y" THEN HOME;GOTO 80
670 IF M$< >"N" GOTO 650
680 END
690 Q=1;V=V+1;N=N+1
700 IF L< >0 AND L-V+C=-1 THEN D=N-1;N=
 15
710 PRINT R$(N);" ";
720 IF L< >0 AND L-V+C=-1 THEN N=D
730 GOTO 620
740 DATA 37558201, 64350311, 36630322, 36655203,
 53770304, 55760326, 33634207, 63560317, 37350428,
 37612210
750 DATA 64350320, 64426101, 16650413, 53770422,
 16375206, 55630416, 37560326, 66222108, 37620319,
 24427330
760 DATA 24430311, 16660323, 22665104, 22370415,

```

```

62230425, 63334207, 66330327, 37630328, 37752109,
 24430321
770 DATA 34436102, 33660313, 32370424, 33365206,
 63350416, 66330326, 66553107, 37750319, 34437330,
 54430411
780 DATA 53660322, 55666104, 35630415, 63560325,
 63624106, 26610418, 57750428, 17762110, 16430421,
 22437203
790 DATA 62260313, 35630424, 36335205, 64220316,
 36620327, 36633108, 23760319, 23768101, 23430412,
 62360322
800 DATA 63366103, 36330415, 64330325, 64354106,
 36630317, 23770329, 53773110, 33430421, 35637202,
 63560313
810 DATA 37550424, 37615205, 64350415, 56650427,
 16654209, 53770419, 16378201, 55630412, 75560322,
 66226103
820 DATA 37620314, 24420326, 24434107, 16660318,
 15771429, 22370411, 62230421, 63336202, 76330312,
 37630324
830 DATA 37755105, 24430316, 22660327, 33663109,
 32370420, 33578210, 63350411, 76330322, 66555203,
 57750314
840 DATA [11],[12],[1],[2],[3],[4],[5],[6],[7],
 [8],[9],[10],[11],[12], 闰, 庚, 辛, 壬, 癸, 甲, 乙,
 丙, 丁, 戊, 己, 申, 酉, 戌, 亥, 子, 丑, 寅, 卯, 辰, 巳, 午,
 未

```



(上接第 23 页)

程序 10.18 设计方法和 10.17 大同小异,只是多调一个显示字符“-”(显示码为 BD),循环次数为 5 次,每次既显示一个“\*”号,又显示一个“-”号,当循环完成后,再调显一次“\*”,只有这样,才和题意要求符合,由于两个程序选用的计数器不同(前者 X,后者为 Y),因而操作码也有区别。

(8)连续显示全部 ASCII 码对应的字符

若从“!”开始显示到“-”结束,即按“!”# \$ ……?

@ABC……^ - ”显示,可用更为简单的程序,见程序 10.19:

```

0300- 20 58 FC JSR $FC58
0303- A9 A0 LDA # $A0
0305- 20 ED FD JSR $FDED
0308- 18 CLC
0309- 69 01 ADC # $01
030B- D9 E0 CMP # $E0
030D- D0 F6 BNE $0305
030F- 60 RTS

```

值得指出,程序 10.19 通用性好,事实上,由于反相显示和闪烁显示 ASCII 码(16 进制表示)按数值顺序递加,只要改动程序中的 \$0304 为 \$00, \$030C 为

\$40 即可顺序反相显示所有对应显示码的字符(见程序 10.20),改动程序中 \$0304 内容为 \$40, \$030C 内容为 \$80,则可顺序闪烁显示所有对应显示码的字符(见程序 10.21)。

程序 10.20:

```

0300- 20 58 FC JSR $FC58
0303- A9 00 LDA # $00
0305- 20 ED FD JSR $FDED
0308- 18 CLC
0309- 69 01 ADC # $01
030B- C9 40 CMP # $40
030D- D0 F6 BNE $0305
030F- 60 RTS

```

程序 10.21:

```

0300- 20 58 FC JSR $FC58
0303- A9 40 LDA # $40
0305- 20 ED FD JSR $FDED
0308- 18 CLC
0309- 69 01 ADC # $01
030B- C9 B0 CMP # $B0
030D- D0 F6 BNE $0305
030F- 60 RTS

```

# 搜索内存数据的程序

苏 华

搜索内存中机器语言程序的代码或数据,找出它们的所在地址,对于分析软件是十分有用的。本文所附的搜索程序有以下特点:

1. 在监控状态下以<首地址>、<末地址>^ Y (^ Y 代表 Ctrl-Y)方式输入搜索范围,再输入要搜索的代码。在改变地址或退出操作之前,该地址范围持续有效。

2. 由于调用了监控系统的行输入子程序,因此具有该子程序提供的功能,如允许一次输入多达 256 次键击、键入缓冲区将满的报警、ESC 键编辑功能等。

3. 搜索程序的装入地址是可浮动的。

键入搜索程序代码,核查无误后,以命令  
BSAVE CODESEEK, A \$ 300, L \$ BC

存入磁盘。使用时,执行

BRUN CODESEEK, A \$ <首地址>

<首地址>可为任一方便的内存页首址,如 9C00 等。程序运行设置有关向量后,仍返回监控状态。搜索时,执行

<首地址>、<末地址>^ Y

这时屏幕出现一个“>”提示符,等待输入要搜索的代码。例如,欲查询在 Applesoft 和监控系统中的哪些地方直接调用了字符输出子程序 COUT(\$FDED),可输入

\* D000.FFFF^ Y  
>20 ED FD (指令 JSR \$FDED 的代码)

显示:

DB64 F903 F91B F923 F94C FAE6 FAEC FAF1  
FD47 FD64 FD6C FDB8 FDD6 FE46 FE4B FE55  
FF2F FF34 FF37

可以看出,Applesoft 只在一处(\$DB64)直接调用了 COUT。

欲继续查询 Applesoft 和监控系统在哪些地方调用了显示累加器内容的 16 进制代码子程序 PRBYTE(\$FDDA),输入:

>20 DA FD (指令 JSR \$FDDA 的代码)

显示:

F8D6 F92D F941 FAF6 FDBD FE41 FE50

上面的显示告诉我们,Applesoft 没有调用 PRBYTE。

如果要接着查询 Apple DOS 在何处直接调用了 COUT,可以这样操作:

>9D00. BFFF 在>提示下改变地址不需输入 ^ Y。

>20 ED FD

显示:

ADB3 ADE5 ADF9 AD FE AE10 AE1B AE31 AE63 B6D4

要退出搜索程序,只需在>提示符下键入 X,即返

回监控状态。要从监控状态再次运行搜索程序,在给出首尾地址后仍需键入 ^ Y。

使用时应注意不要让地址 \$C000~\$C0FF 落入搜索范围,因为这一页地址内有许多软开关,触动某些软开关会造成屏幕进入图形模式或键盘死锁。这就是为什么在上面查询调用 COUT 时,Apple DOS 与 Applesoft 及监控要分开两次查询。

使用 GPIB 卡接收外部设备的测量数据时,在 BASIC 程序中要通过 INPUT 语句接收数据。Applesoft 的 INPUT 语句是不接受字符“:”(ASCII 码为 \$3A),因为它被作为语句分隔符处理。GPIB 卡把接收的每一数据字节拆开作为两个低位半字节,以 03 作为高位半字节,这是为了避免象 \$0D、\$0A 之类的数据被系统程序视为控制符而产生误动作。但这样一来就在转换的输入数据中有很多代码为 3A 的数据被 INPUT 语句所略去。用搜索程序查询发现,只要把 Applesoft 内部 \$DC45 处的一条指令 LDA # \$3A 改为 LDA # \$00,便可解决代码为 3A 的数据被遗漏的问题,具体做法是先把 Applesoft 及监控程序移入 RAM 卡,将 RAM 卡内的 \$DC46 单元改为 0,然后在 RAM 卡的 Applesoft 支持下运行 BASIC 程序,这时 INPUT 语句就可以接受代码为 3A 的数据了。

300- 20 58 FF BA BD 00 01 8D  
308- FA 03 8D FF 03 CA BD 00  
310- 01 18 69 1D 8D F9 03 69  
318- 0F 8D FE 03 4C 69 FF A2  
320- 01 B5 3C 95 FA 95 FC B5  
328- 3E 95 FE CA 10 F3 A2 00  
330- A9 BE 20 ED FD 20 0C FD  
338- C9 D8 F0 E0 20 78 FD A0  
340- 00 84 06 A9 02 85 07 20  
348- A7 FF C9 A7 D0 0B 85 31  
350- 20 A7 FF 20 C7 FF 6C F9  
358- 03 C9 C6 F0 04 49 99 D0  
360- 4A 48 A2 00 A5 3E 81 06  
368- E6 06 D0 02 E6 07 68 F0  
370- D6 A0 00 A5 06 85 08 38  
378- A5 FE E5 FC A5 FF E5 FD  
380- 30 2C B1 FC D9 00 02 D0  
388- 1E C8 C6 08 D0 F4 A5 FD  
390- A6 FC 20 41 F9 A9 A0 20  
398- ED FD A4 06 E6 FC D0 02  
3A0- E6 FD 88 D0 F7 F0 CC A0  
3A8- 01 D0 F1 20 3A FF 20 8E  
3B0- FD A5 FA 85 FC A5 FB 85  
3B8- FD 6C FE 03

## 第十讲 监控子程序的调用(一)

南京大学大气科学系(210008) 朱国江

中华学习机系统程序中,有很多可供用户调用的子程序,其中最有用的要算监控中的子程序,如果我们能充分利用这些子程序,这对于减少用户编程的工作量和节约内存空间,都有实际意义。不仅如此,这些子程序简短独立,功能完整,结构合理,简练灵活,如能合理巧用,往往起到事半功倍的作用,从而使程序清晰易懂,并能提高程序的执行速度。

本讲介绍常用监控子程序及其使用实例。

〔例 1〕将 26 个英文大写字母 A~Z 输出给打印机。

输出一个字符子程序的入口地址为 \$FDED,该子程序既可以将字符输出至屏幕,也可以将字符输出给打印机。若输出给打印机,则应先设备码 C100 存入 \$36 和 \$37 中。\$36 单元存放设备码低字节 \$00, \$37 单元存放设备码高字节 \$C1。然后调用入口为 \$FDED 的输出字符子程序,这样就可以把累加器 A 中存放的字符送往所指定的外部设备(本例是打印机)。当该子程序执行完毕后,又会回到刚才调用它的程序。源程序见 10.1。

程序 10.1:

```

0300- A9 00 LDA # $00
0302- 85 36 STA $36
0304- A9 C1 LDA # $C1
0306- 85 37 STA $37
0308- A9 C1 LDA # $C1
030A- 20 ED FD JSR $FDED
030D- 18 CLC
030E- 69 01 ADC # $01
0310- 89 DB CMP # $DB
0312- D0 F6 BNE $030A
0314- 60 RTS

```

设备送往打印机的设备码  
取字符“A”的 ASCII 码  
显示  
清进位  
指向下一个  
打完了吗?“Z”的 ASCII 码为 DA  
没有,再打  
打完结束

〔例 2〕在屏幕上连续显示 15 个星号“\*”

开机后,通常 \$36 和 \$37 的内容为 F0 和 FD。\$FDF0 就是一个设备码,它的功能是将输出字符送至显示器,因此,不需要重新设置设备码,而直接调用 \$FDF0 的子程序,即可将累加器 A 中的字符输出到屏幕。源程序见 10.2

程序 10.2:

```

0300- A2 0F LDX # $0F
0302- A9 AA LDA # $AA
0304- 20 F0 FD JSR $FDF0
0307- CA DEX
0308- D0 FB BNE $0302
030A- 60 RTS

```

置初值 15  
调“\*”的 ASCII 码  
显示  
X-1→X  
未减完,再循环显示  
减完,结束

〔例 3〕用反相显示方式在屏幕上第一列显示 16

个“+”号。

置反相显示方式的子程序入口地址为 \$FE80,该子程序置屏幕为白底黑字方式。

置正相显示方式的子程序入口地址为 \$FE84,该子程序可将原反相显示方式转为正相显示方式,即置屏幕为黑底白字方式。

程序 10.3

```

0300- 20 80 FE JSR $FE80
0303- A0 10 LDY # $10
0305- 20 8E FD JSR $FD8E
0308- A9 AB LDA # $AB
030A- 20 F0 FD JSR $FDF0
030D- 88 DEY
030E- D0 F5 BNE $0305
0310- 20 84 FE JSR $FE84
0313- 60 RTS

```

设置反相显示方式  
置初值 16  
回车换行  
“+”号的 ASCII 码  
显示  
次数减 1  
未完,继续  
恢复正常显示方式  
结束

程序 10.3 中用了调用 \$FD8E 的子程序,该子程序将回车换行的控制字符送往屏幕,以保证本题“+”号在一列上显示而不是一行上显示。

〔例 4〕将 \$03F0~\$03FF 单元中的两位十六进制数显示出来。

输出两位十六进制字符子程序的入口地址是 \$FDDA,该子程序的功能是将累加器 A 中的内容,按两位十六进数的形式输出到现行输出设备。

程序 10.4

```

0300- A9 F0 LDA # $F0
0302- 85 03 STA $03
0304- A9 03 LDA # $03
0306- 85 04 STA $04
0308- A0 00 LDY # $00
030A- B1 03 LDA ($03),Y
030C- 20 DA FD JSR $FDDA
030F- A9 A0 LDA # $A0
0311- 20 F0 FD JSR $FDF0
0314- C8 INY
0315- C0 10 CPY # $10
0317- D0 F1 BNE $030A
0319- 60 RTS

```

(04)(03)=03F0  
变址计数=0  
第一次调 \$03F0 内容→A  
显示两位十六进制数  
空格“ ”的 ASCII 码  
显示  
计数器+1  
16 个字节内容都送完吗?  
否,继续循环  
是,结束

程序 10.4 中取数指令 LDA(\$03),Y 是采用后的变址 Y 间接寻址方式,它常用于动态数据块处理,即数据块在存储器中存放的位置是可以变化的,只要把数据块首地址置入所选用的零页地址中即可。

〔例 5〕已知在 \$03F8~\$03FF 单元中,存放四个转移地址,现要求将它们打印出来。

输出四位十六进制数的子程序入口地址为 \$F941, 它的功能是将寄存器 X 和累加器 A 中的内容按四位十六进制数的形式输出到现行输出设备。

调用 \$F941 之前, 必须将高字节放入累加器 A 中, 低字节放在 X 寄存器中。

程序 10.5

```

0300- A0 02 LDY $03F9,Y 变址计数=2
0302- AD F9 03 LDA $03F8,Y } 设置源数据块首址
0305- AE F8 03 LDX $03F8 }
0308- 20 41 F9 JSR $F941 显示四位 16 进制数
030B- A9 A0 LDA # $A0 } 空一格
030D- 20 F0 FD JSR $PDF0 }
0310- B9 F9 03 LDA $03F9,Y 第一次取 $03FA~
0313- BE F8 03 LDX $03F8,Y $03FB 内容。
0316- 08 INY } 计数器加 2
0317 C8 INY }
0318- C0 0A CPY # $0A 数据块有无送完?
031A- D0 EB BNE $030B 没有, 继续
031C- 60 RTS 是的, 结束

```

LDA \$03F9, Y 和 LDX \$03F8, Y 均采用绝对 Y 变址寻址方式。

〔例 6〕把 \$03F0~\$03FF 中各单元内容按两位 16 进制数显示出来。要求每两个数之间空三个空格。

\$F948 是把三个空格字符送往现行输出设备的子程序入口地址。

程序 10.6

```

0300- A0 00 LDY # $00 初始化计数器
0302- B9 F0 03 LDA $03F0,Y 调($03F0+Y)的内容
0305- 20 DA FD JSR $FDDA 显示两位十六进制数
0308- 20 48 F9 JSR $F948 空三格
030B- C8 INY 计数器加 1
030C- C0 10 CPY # $10 送完吗?
030E- D0 F2 BNE $0302 否, 再送
0310- 60 RTS 是, 结束

```

在监控子程序中, 还有一个可以输出 X 个空格的子程序, 其入口地址为 \$F94A, 但调用该子程序前, 必须将空格数设置在寄存器 X 中。

〔例 7〕不断按键取字符送显示屏显示, 若按下空格键时则结束。

取一按键的值(取一个输入字符)的子程序入口地址为 \$FD0C, 该子程序可以接受从键盘输入的一个字符。

程序 10.7

```

0300- 20 0C FD JSR $FD0C 取一按键
0303- 20 F0 FD JSR $PDF0 显示
0306- C9 A0 CMP # $A0 是空格吗?
0308- D0 F6 BNE $0300 不是, 继续
030A- 60 RTS 是, 停机

```

读取键盘数据还有一个子程序, 其入口地址为 \$FD1B, 它能等待按键输入, 当有键按下时, 该子程序就将该键值送往屏幕光标当前位置和累加器 A 中。

〔例 8〕编制几个不同延时时间的延迟程序

在监控程序中, 有一个延时程序, 其入口地址为 \$FCA8, 可以直接调用。

程序 10.8、10.9、10.10、10.11、10.12 是用不同方法编制的几个延时程序。

程序 10.8

```

0300- A2 40 LDX # $40 计数器置初值
0302- A9 FF LDA # $FF 累加器置初值
0304- 20 A8 FC JSR $FDA8 调监控延迟子程序
0307- CA DEX X-1→X
0308- D0 F8 BNE $0302 只要 X 不为零, 循环
030A- 60 RTS X=0, 结束

```

程序 10.9

```

0300- A9 FF LDA # $FF 置延时常数
0302- 38 SEC 置进位标志 C=1
0303- E9 01 SBC # $01 A-01→C
0305- D0 FB BNE $0302 若不为 0, 循环
0307- 60 RTS 是 0, 结束

```

程序 10.10

```

031D- A0 10 LDY # $10 外循环计数器初值
031F- A2 FF LDX # $FF 内循环计数器初值
0321- CA DEX X-1→X, 为 0 否
0322- D0 FD BNE $0321 X 不为 0, 继续内循环
0324- C8 INY X=0, 外循环次数 Y+1
0325- D0 F8 BNE $031F Y≠0, 继续外循环
0327- 60 RTS Y=0, 结束

```

程序 10.11

```

031D- A2 10 LDX # $10 外循环计数器初值
031F- A0 FF LDY # $FF 内循环计数器初值
0321- 88 DEY Y-1→Y
0322- D0 FD BNE $0321 Y≠0, 继续内循环
0324- E8 INX Y=0, 外循环次数 X+1
0325- D0 F8 BNE $031F X≠1, 继续外循环
0327- 60 RTS X=0, 结束

```

程序 10.12

```

FCA8- 38 SEC 置进位标志 C=1
FCA9- 48 PHA 累加器 A 的内容进入堆栈
FCAA- E9 01 SBC # $01 (A)-1→A
FCAC- D0 FC BNE $FCAA (A)≠0 则循环
FCAE- 68 PLA A 的内容出栈
FCAF- E9 01 SBC # $01 (A)-1→A
FCB1- D0 F6 BNE $FCA9 循环至(A)=0
FCB3- 60 RTS (A)=0, 结束

```

几点说明:

• 程序 10.8 是一个延时程序, 但其中又调用了监控中的延时程序 \$FCA8。

• 程序 10.9 中用了 SBC 指令, 使用前必须用 SEC 指令将 C 标志位置为 1, 否则影响运算结果。

• 10.10、10.11 其程序设计思想完全一样, 都是用双重循环控制循环次数的办法达到延时的目的。但内、外循环计数器选用的不同, 因而寄存器 X(或 Y)内容增、减一的指令也不同, 使用时应注意搭配。

10.12 是监控延迟子程序,采用了堆栈处理方法,调用前先设置累加器 A 中的初值,然后调用,如:

```
0300-A9 20 LDA # $20
0302-20 A8 FC JSR $FCA8
0305-60 RTS
```

### 〔例九〕显示字符的各种方法

为了在屏幕上显示字符或汉字(一个或多个),可以有不同的方法。而字符的显示方式也可以多样(黑底白字的正常显示、白底黑字的反相显示、闪动显示)。字符可以在固定的位置上显示,也可以在屏幕上用“开窗口”的方法,设定一块显示区显示。

#### (1) 单个字符的正常显示

例如欲显示一个“A”字符,取正常方式,在文本状态第一页的起始位置上,可用程序 10.13:

#### 程序 10.13

```
0300- 20 58 FC JSR $FC58 清屏
0303- A9 C1 LDA # $C1 把 C1 送入累加器 A 中
0305- 8D 00 04 STA $0400 把 A 中的数存入地址 0400
0308- 60 RTS 返回
```

其中 C1 是字符“A”正常显示方式的显示码。\$0400 是屏幕文本第一页第一个位置的映像地址。在监控状态下,300G $\swarrow$ ,即可看到屏幕左上方显示一个“A”。

(2) 调用输出一个字符的监控子程序,正常显示“A”字符,见程序 10.14

```
0300- 20 58 FC JSR $FC58
0303- A9 C1 LDA # $C1
0305- 20 ED FD JSR $FDED
0308- 60 RTS
```

#### (3) 连续显示几个反相字符

例如,反相显示“BASIC”五个字符。

方法是反复调用 LDA 和 JSR 这两个指令,见程序

#### 10.15:

```
0300- 20 58 FD JSR $FC58
0303- A9 02 LDA # $02
0305- 20 ED FD JSR $FDED
0308- A9 01 LDA # $01
030A- 20 ED FD JSR $FDED
030D- A9 13 LDA # $13
030F- 20 ED FD JSR $FDED
0312- A9 09 LDA # $09
0314- 20 ED FD JSR $FDED
0317- A9 03 LDA # $03
0319- 20 ED FD JSR $FDED
0310- 60 RTS
```

300G $\swarrow$ 后,在屏幕左上方自左至右反相显示“BASIC”五个字符。# \$02, # \$01, # \$13, # \$09, # \$03 分别为字符 B, A, S, I, C 的反相显示代码

#### (4) 用循环的方法显示(闪烁)字符串

例如,闪动显示“6502 CPU”这七个字符。见程序

#### 10.16:

```
0300- 20 58 FC JSR $FC58 清屏
0303- A2 00 LDX # $00 寄存器 X 置初值 00
0305- BD 11 03 LDA $0311, 将(0311+X)的数送累加器
X
0308- 20 ED FD JSR $FDED 显示累加器 A 中的字符
030B- E8 INX X+1→X
030C- E0 07 CPX # $07 比较 X 的值是否与 07 相符
030E- D0 F5 BNE $0305 如果不等转 0305
0310- 60 RTS 相等则返回
0311- 76 75 70 72 43 50 55 $0311 开始存放 6502CPU 的闪
动方式的 ASCII 码
```

程序 10.16 的设计思想如下:

首先初始化 X 寄存器,置初值为 00,然后取 0311 + X = 0311 地址的值 76(字符 6 的闪烁显示码)至累加器 A,接着显示这个字符(调 \$FDED 输出一个字符),让寄存器 X+1,判断是否与 07 相符(6502CPU 共 7 个字符),不相等再循环上去(到 \$0305),重复上述过程;若相等则结束。

#### (5) 连续显示若干个相同字符

例如,连续显示 20 个“\*”,可用程序 10.17:

```
0300- 20 58 FC JSR $FC58
0303- A2 00 LDX # $00
0305- A9 AA LDA # $AA
0307- 20 ED FD JSR $FDED
030A- E8 INX
030B- E0 14 CPX # $14
030D- D0 F8 BNE $0307
030F- 60 RTS
```

程序 10.17 采用单重循环反复相减的方法,控制显示“\*”的次数。循环之前清屏、计数器置 0、取“\*”的显示码 AA,循环开始显示“\*”,计数器加 1,只要不是 20 次(\$14),继续循环显示,直到计满 20 次跳出循环结束运行。

#### (7) 间隔显示一串不同字符

例如,间隔显示: \* - \* - \* - \* - \* - \*, 参见

#### 程序 10.18:

```
0300- 20 58 FC JSR $FC58
0303- A0 00 LDY # $00
0305- A9 AA LDA # $AA
0307- 20 ED FD JSR $FDED
030A- A9 BD LDA # $BD
030C- 20 ED FD JSR $FDED
030F- C8 INY
0310- C0 05 CPY # $05
0312- D0 F1 BNE $0305
0314- A9 AA LDA # $AA
0316- 20 ED FD JSR $FDED
0319- 60 RTS
```

(下转 19 页)



## BJS—51 单片机实验系统(续)

北京广播电视大学 李广弟

### (7) 思考题

①如各轮比较按实际需要的次数进行,而不是统设的七次,应对本实验程序作哪些修改?

②同样是单字节无符号数,如进行降序排序,需对本实验程序作哪些修改?

### 2. 交通灯控制实验

除汇编语言程序设计之外,单片处机基本实验的另一方面内容是接口及应用方面的实验。现以交通灯控制为例进行说明。

交通灯控制是一个比较典型的控制类实验,使用实验器上的两组红黄绿发光二极管和两支数码管可以很形象地模拟实现。教程中按复杂程度不同,把这个题目分为四个既相互独立又互有联系的实验,供不同层次的学生选作。这四个实验分别是:

#### ①交通灯定时控制实验

要求两组交通灯定时按规则变化,即模拟普通路口的交通灯控制。本实验将使学生在编写计时程序和实现状态输出控制等应用技术方面得到实践机会。

#### ②带时间显示的交通灯定时控制实验

本实验是在前一个实验的基础上,加进时间显示,使交通灯状态转换与时间变化协调配合。这将使 LED 显示技术得到充分的应用。

#### ③主支线路口的交通灯控制实验

由于主线和支线交通繁忙程度相差悬殊,因此主支线路口的交通灯应根据车辆情况进行随机控制。为此在实验中需引入中断技术和查询技术。从而构成一个较大型的综合实验。

#### ④有救护车优先的交通灯控制实验。

本实验同样需引入中断控制技术,并且中断处理内容更加复杂。

下面以主支线路口交通灯控制实验为例对接口控制类实验的模式进行说明。

### 1. 实验题目

由主线和支线构成的路口,交通控制原则是尽可能保证主线车辆畅通,在有需要时才开通支线。为此,交通灯共有四种控制状态:

①在通常情况下,主线为绿灯,支线为红灯。

②如支线有车辆到达,则延迟 5 秒后使主线由绿灯变黄灯。再延迟 5 秒后,主线由黄灯变红灯,支线由红灯变绿灯。

③支线放行最长可持续 25 秒,然后变黄灯再经 5 秒后变为红灯。同时主线由红灯变绿灯。

④支线绿灯期间,若主线有 3 辆以上车辆到达,则

在第三辆车到达之时,使支线变为黄灯,延迟 5 秒后变红灯,同时主线由红灯变绿灯。

### 2. 实验目的

- 了解主支线路口交通灯控制的模拟实现方法。
- 学习计数中断的使用方法。
- 学习如何运用查询技术对外部状态变化作出反应。

### 3. 实验内容

- 以二组发光二极管模拟二组交通灯并进行电路连接。
- 编写主程序、时间延迟子程序及中断服务程序。
- 输入、调试并运行程序。
- 观察程序运行结果,看发光二极管是否按要求

规律变化:

键 1 代表支线有车辆到达,按一下后主线的黄色发光二极管亮,然后转红色亮,同时支线的绿色发光二极管亮。

键 2 代表主线车辆到达,在支线放行期间,按键 2 三次,代表主线已有三辆车到达。这时支线的黄色发光二极管亮,然后转红色亮,同时主线的绿色发光二极管亮。

### 4. 电路连接

本实验电路设计有如下要点:

①发光二极管的连接同“定时交通灯控制实验”完全一样。

②键 1 用来模拟支线车辆到达,按下后电平为 0。把键 1 与 P1.7 连线,这样,通过测试 P1.7 的电平状态,就可以了解支线是否有车辆到达。

③键 2 用来模拟主线车辆的到达。按下后电平为 0,通过 P3.5(定时器 1 的外部输入端)输入,因此键 2 与 P3.5 连线(电路连接图略)。

### 5. 中断技术的应用

当支线放行时,按键 2 三次,代表主线已有三辆车到达,这时应提前结束支线放行状态。对这种控制要求最好使用中断技术实现。具体说就是计数溢出中断。为此要使用 8031 的定时器/计数器进行按键次数的计数,因为计数值仅为 3,所以使用定时器/计数器 1。根据交通灯控制的需要,应采用模式 2 工作方式,以利用其溢出后计数值自动重装的特点。这时 TL1 构成一个可自动装载的 8 位计数器,计数溢出时,不但能发出中断请求,而且还能实现计数值的重新装入,实现控制的连续性。

#### ①定时器/计数器初始值设置



TH1 为预值寄存器(预置值为 FDH)

TL1 为计数器预置值也为 FDH

②模式控制寄存器(TMOD,地址 89H)的状态设置

GATE	C/T	M1	M0	GATE	C/T	M1	M0
0	1	1	0	0	0	0	0

定时器/计数器 1(TMOD)=60H

③控制寄存器(TCON,地址 88H)的状态设置

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
0	1	0	0	0	0	0	0

(TCON)=40H

④中断优先级控制器(IP,地址 B8H)

本系统只有一个中断请求,因此无需设定优先级,故取其值为 00H。

⑤允许中断寄存器(IE,地址 A8H)的状态设置。

EA	X	ET2	ES	ET1	EX1	ET0	EX0
1	0	0	0	1	0	0	0

(IE)=88H

⑥各寄存器状态设置汇总:

寄存器名称	IE	IP	TCON	TMOD	TH1	TL1	
地址	A8H	B8H	88H	89H	8DH	8BH	
内容	88H	00H	40H	60H	FDH	FDH	

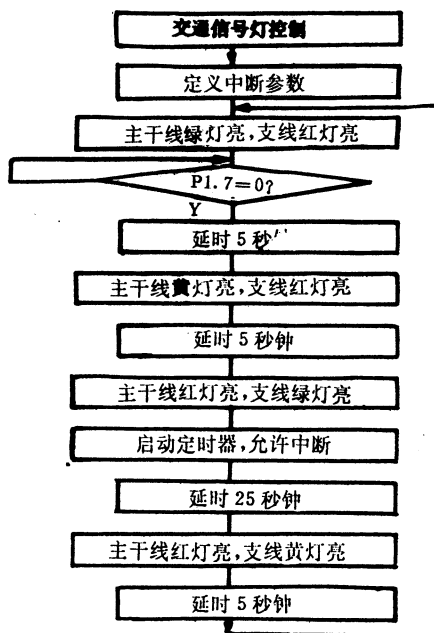
7. 参考程序清单

主程序

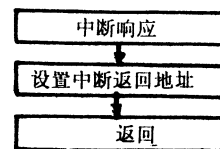
8000	0130		AJMP MAIN	
801B	0190		AJMP INT	
8030	758960	MAIN:	MOV TMOD, #60H	定义 T1 为工作方式 2
8033	758BFD		MOV TL1, #FDH	计数器初始值
8036	758DFD		MOV TH1, #FDH	计数器预置值
8039	7590F3	LOOP1:	MOV P1, #F3H	主线放行,支线禁止
803C	E590	LOOP2:	MOV A,P1	
803E	20E7FB		JB ACC.7,LOOP2	测试支线有车到达否
8041	7F05		MOV R7, #05H	支线有车
8043	1180	LOOP3:	ACALL DEL	5 秒延时
8045	DFFC		DJNZ R7,LOOP3	
8047	7590F5		MOV P1, #F5H	主线警告,支线禁止
804A	7F05		MOV R7, #05H	
804C	1180	LOOP4:	ACALL DEL	5 秒延时
804E	DFFC		DJNZ R7,LOOP4	

6. 参考程序流程:

主程序流程



中断服务程序流程



8050	7590DE		MOV P1, #DEH	主线禁止,支线放行
8053	758840		MOV TCON, #40H	启动计数器
8056	75A888		MOV IE, #88H	中断允许
8059	7F19		MOV R7, #19H	
805B	1180	LOOP5:	ACALL DEL	25秒延时
805D	DFFC		DJNZ R7, LOOP5	
805F	7590EE		MOV P1, #EEH	主线禁止,支线警告
8062	758800		MOV TCON, #00H	关闭计数器
8065	D7F05		MOV R7, #05H	
8067	1180	LOOP6:	ACALL DEL	5秒延时
8069	DFFC		DJNZ R7, LOOP6	
806B	0130		AJMP LOOP1	转主线放行,支线禁止

### 中断服务程序

8090	745F		MOV A, #5FH	
8092	C0E0		PUSH A	
8094	7480		MOV A, #80H	
8096	C0E0		PUSH A	
8098	32		RETI	

(1秒延时子程序略)

编者按:武汉创意电子研究所研究人员吴徽、马国敏、孔曙光、罗维国曾经在《电子与电脑》杂志上(1990年第2、3期)公布了“SCB-1 MCS-51 单片单板机”的有关论文(笔名吴中国)。去年他们又在北航出版社由何立民教授主编的《单片机应用文集》中以十五万字篇幅公布了“SCB-2 MCS-51、8098 单片单板机”的论文。CYSCB-2 MCS-51、8098 单片单板机则作了进一步的改进,硬件向用户公布,采用积木式结构,使用仅需+5V供电的TTL-RS232转换的MAX232、复位键置于主板上,有电视接口选件板、全系列高速EPROM编程板选配,提供总线输出等。新型机配有MCS-51、MCS-96二种本机板和二种CRT式监控,软件在文集中已公布,加之高位机交互式菜单构成的汇编、反汇编、编辑及通信,使得用户通过它既能学习、掌握MCS-51、8098二种单片机,又能开发以其为核心的新产品。新型机A/D、D/A、PIO配备齐全,能直接用于许多控制项目中。需要详细硬件原理及软件清单者请与武汉创意电子研究所联系。

### CYSCB-2 MCS-51 8098 单片单板机硬件设计原理

吴徽 罗维国 马国敏 孔曙光

#### 一、CYSCB-2 MCS-51、8098 单片单板机主要技术指

标:

1. 核心单元是MCS-51的8031芯片或准16位单片机8098。插上8031芯片则成为8031单片单板机,插上8098则成为8098单片单板机,两者不能同时插上。
2. 系统时钟选为8MHz和12MHz两种供用户选择。
3. 外部EPROM为27128一片,16K程序存储器,低8K为8031本机板监控,高8K为8098本机板监控。
4. 外部RAM为6264二片共16K,其中8K掉电保护。
5. 设有8255可编程I/O口,提供24位口线,可直接驱动打印机,开关量等。
6. 8031、8098的I/O(高速I/O)等口线通过插座输出。
7. 通过仅需+5V供电的MAX232实现TTL与RS232之间的转换,克服了以前需±12V供电的困难。
8. 通过DAC0832,1路8位D/A,可进行数模转换,模拟输出0~5V。8098则有一路PWM输出。
9. 通过ADC0809,8路8位A/D,可进行模数转换,模拟输入0~5V,8098片内有4路10位A/

D.

10. 可选配件有高速智能 EPROM 编程器,可以实现对 2716~27512 的高速编程,解决了低速编程器编程速度慢、无检验的问题。
11. 可选配件 TV 接口板实现主板与普通电视或 CRT 监视器的接口,方便现场操作。
12. 键盘显示板通过 20 芯电缆线与主机板相接,键盘显示板采用最简洁的三片结构,使用方便可靠,主板与键盘显示板之间的接口作了预留,方便用 8279 实现键盘显示,并可提供 8279 控制的键盘显示配件。
13. 总线通过 40 芯电缆线插头输出,方便用户扩展外围芯片。
14. 提供二套监控,本机板监控方便现场操作调试,主板 CRT 监控方便实验室调试扩展,特别是 CRT 监控仅使用 TXT、RXD 二根口线,其它所有资源全部提供给用户,为用户开发提供强有力的手段。
15. 具有外部抗干扰 Watch Dog 电路提高系统的可靠性,8098 则具有内部 Watch Dog 电路。

## 二、CYSCB-2 MCS-51、8098 单片单板机硬件原理:

理:

### 1. CYSCB-2 型单片单板机硬件概述:

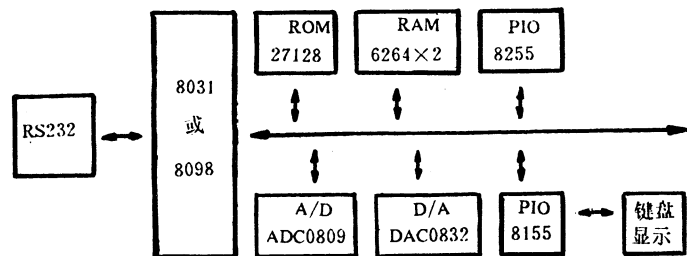


图1 CYSCB-2型单片单板机硬件结构框图

CPU: 8031 或 8098, 插上 8031 则成为 MCS-51 单片单板机, 插上 8098 则成为 8098 单片单板机。成品机上 40 脚插座插 8031, 48 脚插座插 8098, 但不能将两种单片机同时插上。

EPROM: ROM 选用一片 16K EPROM 27128, 其中前 8K 存放 MCS-51 监控程序, 后 8K 存放 8098 监控程序。

RAM: 外部随机存储器选用 2 片静态 8K RAM 6264, 这样机上外部 RAM 容量为 16K。外部高 8K RAM 配有掉电保护电路。在这二片 RAM 的插座上可以对 2864 进行编程。

PIO: 外部并行口选用了一片 8255, 利用它通过接一智能 EPROM 编程器, 能对全系列 EPROM 进行高速编程, 此外还可驱动打印机等。

A/D: 外部配的 A/D 是 8 路 8 位的 ADC0809, 若是 8098 单片机, 则片内有 4 路 10 位的高速 A/D。

D/A: 外部配有 D/A 是 1 路 8 位的 DAC0832, 若是 8098 则片内有 1 路脉宽调制输出。

键盘显示电路采用 8155 和 2 片 ULN2003 构成的最简单的三片结构, 可靠性大大提高。

键输入: 本机配有 26 个压电式小按键, 其中主板上有一个作复位键使用, 16 个做数值键, 9 个做命令键。

显示输出: 显示输出由六个共阴数码管组成, 左边 4 个用来显示地址或状态信息, 右边 2 个显示数据或代码。

RS232 接口: CYSCB-2 型单片单板机借助 8031 或 8098 片内 SIO 的, 通过一片 MAX232 实现简易 RS232 的接口, 通过该接口与高位机进行通信, 借助高位机 TTY 的模拟程序能对单片机实现更方便的调试。

### 2 CYSCB-2 型单片单板机硬件分析:

图 2 是 CYSCB-2 型单片单板机硬件电路原理图, 下面详细分析几个核心电路:

#### 1 8 位单片机 8031 和准 16 位单片机 8098

##### ① 8031 总线与 8098 总线

我们将 8031 和 8098 二种单机的数据/地址总线联一起, 其含义仅指二种单片机共用 EPROM、RAM、键盘、显示等。当 CYSCB-2 型单片单板机作 MCS-51 单片单板机时, 仅把 8031 插上, 不能插 8098; 当作为 8098 单片单板机时, 仅把 8098 插上, 不能插 8031, 否则单片单板机不但不能工作, 而且可能损坏二种单片机。当然, 若我们在硬件上加上 4 片 74LS244 和二片 74LS245, 通过开关控制平时仅有一种单片机上总线工作, 但硬件成本增加。

##### ② 统一编址:

大家知道 8031 程序存储器和数据存储器是分开编址, 8098 则是统一编址。为了兼顾到二者使用统一的 EPROM、RAM、键盘、显示等, 再者考虑到 CYSCB-2 型单片机供学习、调试程序需用 RAM 存储器器码, 我们将 8031 的 PSEN 与 RD 二根信号线相与形成统一读信号线 RD, 这样对 8031 而言外部 EPROM 和 RAM 也须统一编址。将 8031 RD 与 8098 单片机 RD 的信号线相与则形成 CYSCB-2 单片单板机的读信号线。

### 2 译码电路

上面谈了 CYSCB-2 型单片单板机采用统一编址, 而完成统一编址是译码电路。图 2 是译码电路的详细描述, 下面介绍挂在总线上的各部分电路的地址。

27128 地址: 0000H~3FFFH 或 4000H~7FFFH

6264 (M<sub>3</sub>) 地址: 4000H~5FFFH 或 0000H~1FFFH

6264 (M<sub>4</sub>) 地址: 6000H~7FFFH 或 2000H~3FFFH

27128 与 2 片 6264 地址切换取决于外接开关 CSW6。A/D、D/A、8155、键盘以及显示地址详见有关电路图。

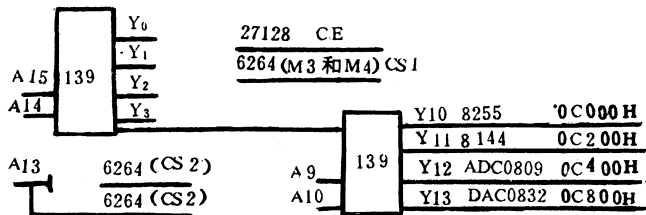


图 2 CYSCB-2 型单征单板机译码电路

3 掉电保护电路

CYSCB-2 型单片单板机采用了一比较简便的外部 RAM 掉电保护电路,当外部加电源时 2 片 6264 由外部供电,当外部电源断开时,其中一片 6264 由一个 4.5V 的纽扣电池供电。

4 对 8031 运用的外部 WATCHDOG 电路

8098 内部有 WATCHDOG 电路,对 8031 而言,只

有增加外部 WATCHDOG 电路才能增加系统的抗干扰性,通过 P1.0 在时间  $T=1/(2 * C_{12} * R_1)$  内用正沿触发 4098,则系统处于正常工作,若在此时间内没有正沿触发,则系统自复位。当读者不使用 WATCHDOG 电路时,则可将 4098 拨下。关于 CD4098,读者可参考 CMOS 手册。

5 总线输出

CYSCB-2 MCS-51、8098 单片单板通过 JP3 输出数据总线和地址总线,读者可以扩充相应的接口芯片,值得提醒读者的是若打算使用本机板上大多数芯片,同时还需扩充,则需在总线上通过 74LS244、74LS245 扩充总线的驱动能力。

6 不靠土 12V 供电的 TTL↔RS232 转换电路

MAX232 是国外最新推出的 TTL↔RS232 转换芯片,该芯片外接 4 个 10μ 的电容即能实现单+5V 供电的 TTL↔RS232 转换,详细请参考电路图及 MAX232 说明书。

网络管理信息系统快速开发工具 支持“快速原型——增量”开发方法

C-DBAG 中文数据库系列应用生成器新产品

F/D \* AGV5.0

北京航空航天大学开发

使用本系统只需按照提示输入用户需求,不需编程,即可生成(如财会、金融、统计、销售等)各种管理软件,在功能变动时可任意改动,适用于网络及单用户环境。

功能: • 生成 DBASE+/FoxBASE+ 源程序

- 生成任意格式报表
- 自动生成数据字典及文档
- 工程图形生成与管理

详细功能请看本期软件介绍栏目

组成: • 下拉式/弹出式菜单生成器

- 数据库文件生成器
- 数据录入与维护模式生成器
- 查询模块生成器
- 统计与计算模块生成器
- 统计图形生成器

- 任意格式报表生成器
- 数据字典与文档生成器
- 92 年 10 月 27 日在北航开办培训班

经销:中电华北公司电脑部 电话:81.1810

地址:北京万寿路西街五号

邮编:100036

联系人:石立军 魏 国

北京景文科技开发公司

电话 832.2255 转 458

地址:北京西直门外首体主楼 305

邮编:100081

联系人:袁凯峰 李应知

本公司经营计算机及外设通讯设备,欢迎来电来函联系。

大量供应显示器及大功率开关电源

我部现货供应进口 19" 彩色、单色高分辨(1280×1024)图形显示器;12" 终端,并供应大功率开关电源,规格有:5V/18A,5V/20A,+5V/50A,+5V/100A,5V/120A,5V/150A×2,12V/15A。并有各种进口电脑装配电线、电缆(单层屏蔽、多层屏蔽、单芯、多芯、镀银等)品种繁多,适合各类科研开发、生产、装配、工业自动化等单位使用。

以上产品均全部进品并通过美国 UL 安全标准认证,产品批发价相当优惠、欢迎来函、来电或来人洽谈。

广东省四会县南方电子厂经营部

地址:广东省四会县城高观东路 71 号

邮编:526200

电话:(07663)322686

电挂:1311

# 告读者、作者和各界朋友

## 本刊编辑部

随着改革开放的深入、科技事业的发展,随着人们对电脑认识的深化,今后将会有越来越多的家庭和初学者在“早日学电脑、终生都受益”的启示下,走入电脑爱好者的行列。

《电子与电脑》是电脑爱好者的朋友,希望大家充分利用这块园地,交流用机经验、编程技巧、开发实验的制作与体会。通过杂志的系列讲座和函授班较系统的学习软硬件知识和技术。值此 1992 年第四季度到来之际,我们向大家通告,93 年本刊编排和改进计划。本刊仍以中初级电脑用户为对象结合国内常用机型(PC 系列、中华机系列、单片与单板机系列),普及软硬件知识和用机经验。在保持现有栏目的情况下,93 年将增加二个新栏目,并不定期试办信息性栏目。为此,随着栏目的扩充,正文由每期 48 页增加到 56 页,定价调整到 1.60 元。

新增栏目为:

“IC 与应用”微电脑和微电子技术是一对孪生兄弟。随着科技发展的需要,大家迫切要知道国外 IC 的

发展及最新 IC 的应用资料,本刊将努力搜集这方面的资料,介绍给大家。诸如,开关电源 IC、图象处理 IC、模糊逻辑 IC、通讯 IC 等等。

“电脑与通信”随着国内企业集团化的发展,政府决策等需要,电脑联网通信的问题已势在必行。为了使本刊广大单机用户,开始尝试一下,点对点及网络通信、办公室自动化味道,我们将组织一些基础知识、模拟通信实验,模拟办公室自动化及网络实验方面的内容奉献给各位,使大家较快的进入网络之门。

信息、广告、经营和读者联谊等方面一直是本刊的弱点,在新的一年里我们将努力改进这方面的工作。使本刊不仅在技术业务方面给读者提供帮助,还要在信息咨询、开发产品,代购代销方面为本刊的单位(特别是生产单位)和个人订户提供合作和服务。

俗话说,“一个好汉三个帮”,要办好一个杂志,非常需要广大读者、作者和朋友们的帮助,各们对明年我们的改进意见有什么建议,望多多赐教。谢谢。

(上接第 3 页)

主体式 CAI 是由 CAI 系统全部取代教师,由系统传授知识,提供资料,还可进行检索、改编、测验等。

辅助式 CAI 是由 CAI 系统部分地代替教师,主要用于帮助练习、复习、解题和提供辅导测验等。

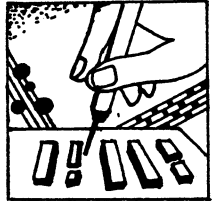
利用 CAI 系统传授知识和提供训练一般也有两种模式,一种是由计算机逐个显示问题,由学生通过键盘输入答案,然后计算机来判断答案是否正确。如果正确,则显示下一个问题;如果错了,则给予适当的提示并再给一次回答机会。这样,通过不断地提出问题和增加问题的难度来帮助学生巩固知识和增强能力。另一种是把教学内容适当地分成一些教学单元并排出顺序,当学习开始时,计算机首先向学生显示各教学单元的目录,由学生自己来选择学习内容。内容选定之后,计算机不断地显示教学内容并配以提问,帮助学生学习。如果学生感到学习有困难,计算机可提供适当的提示,学生也可重新选择适合于自己情况的教学单元学习。如果学生顺利通过本教学单元,则可跳到高一级的水平。

CAI 可以模拟物理、化学及自然界各种动态变化的现象,它以计算机屏幕作为直观的教具,显示的图像具有直观性、动态感强、速度快、色彩丰富并还可配有音响效果。使得某些以往只能用语言来间接描述的微观的、宏观的、瞬时的等现象得到准确、直观的表达。CAI 还可提供汽车、轮船、飞机的驾驶和各种兵器操纵

的训练,CAI 提供这些训练既安全、逼真,又大量节省资金,现已被很多国家广泛应用。

CAI 与传统教学模式相比较,最突出的特点是真正实现了以学生为中心的人格化教学方法,由学生按照自己的实际情况来选择学习内容,控制学习进度,及时了解自己的学习效果。同时,由于计算机在教学过程中表现出的无比“耐心”所提供的安祥和的学习气氛,使得学生消除了诸方面的畏惧心理,从而能使学习的成功性大大提高。

当然,CAI 作为一种新兴的教育技术,还有不完善之处。在我国,除教育指导思想,资金设备等问题外,在技术上还存在着自然语言和图象的输入输出,人工智能以及开发工具落后等突出问题。但是,CAI 所体现的全新的教育思想和方法的意义和价值,却是不可否认的。美国加州大学欧文分校的布克教授在《Personal Computers for Education》一书中这样写道:“再过二十五年,计算机将成为教育领域中占主导地位的传播媒介。在大多数学科中,越来越多的不同年龄人,从计算机上学到的东西将比从教科书上、课堂上或用其它方式学到的东西要多。”美国教育学家西摩·佩珀则预言:“到本世纪末,我们现在所熟知的教室将不复存在。”到那时,孩子们象现在使用铅笔和书本一样摆弄计算机。国际上最具权威性的信息技术专家们对 53 项计算机应用课题的发展前途进行名次评选时,将计算机教育排在第六位。



## 学装微电脑

# 微电脑控制微型钻床

(学习控制 X-Y 工作台方法)

易齐干

微型钻床大致分为: X 轴部件、Y 轴部件、工作台部件和钻头部件。

X 轴部件和 Y 轴部件的基体为有机玻璃。微型 DC 电机为驱动源, 经过齿轮减速带动丝杠, 拖动滑台移动。滑台运动随 DC 电机正、反转, 有正向、反向移动。

检测电机转速使用编码器, 电机转一周, 接收 4 个开闭信号, 发出 4 个脉冲。

被钻削的工件定位在工作台上。工作台部件完全由有机玻璃制成。

钻头部件基体为有机玻璃, 其上安装 DC 电磁铁和弹簧, 控制钻头上、下运动。钻头的旋转由微型 DC 电机驱动。

电路原理图如图 1 所示。图中 8255 为

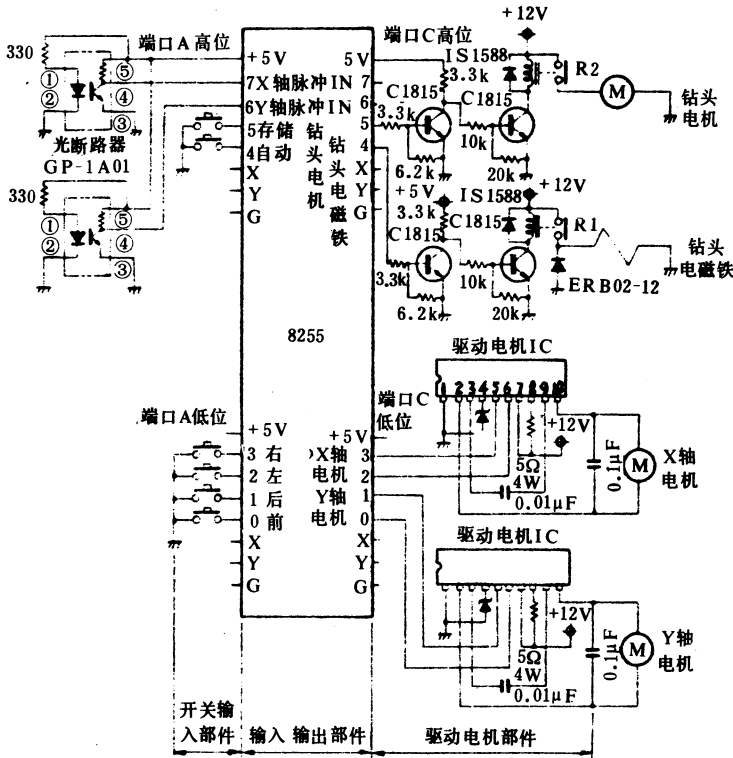


图 1

$\mu P-80$  套件的输入输出部件。

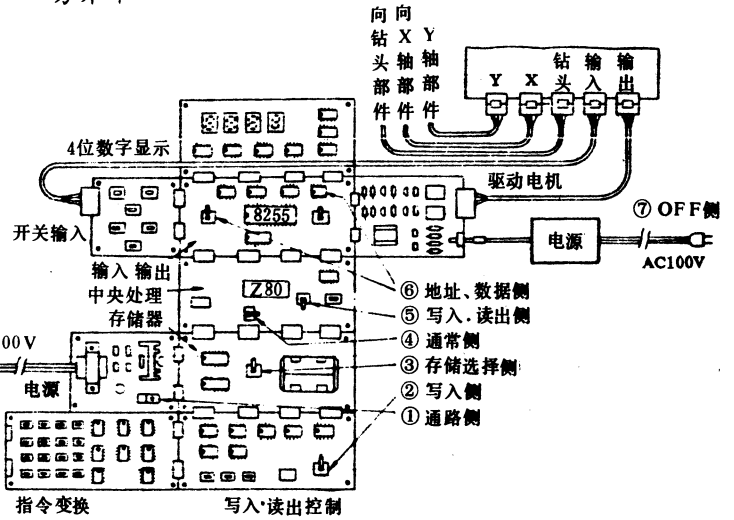


图 2  $\mu P-80$  套件与钻床连接

驱动电机 IC 型号为 BA6109, CPU 的控制信号

经过放大, 再分别拖动 X 轴、Y 轴电机。BA6109 能实现电机正、反转和刹车, 这样可大大简化结构和电路。

CPU 发出的控制钻头的信号, 经过晶体管放大, 通过继电器, 控制钻头电机和电磁铁。所需电压是 DC12V, 所以要再准备一个电源。

X 轴、Y 轴电机上的编码器检测出的脉冲信号输入给 8255 端口 A 的第 6 位、第 7 位。

Y 轴部件向前、向后运动, X 轴部件向左、向右运动; 自动钻孔开关均使用端口 A 的其余六位。

$\mu P-80$  微电脑套件与微型钻床的连接如图 2。

图 2 所示电路中, 端口 C 的  $PC_4$ 、 $PC_5$  分别输出 L 电平, 晶体管 (2SC1815) 导通, 继电器为通状态, 钻头电机和电磁铁工作。

$PC_4$ 、 $PC_5$  分别输出 H 电平, 晶体管、继电器均为断状态, 钻头电机和电磁铁也不工作。

驱动电机 IC (BA6109) 的输入输出引脚的电位与

电机转向的关系如表 1 所示。

输入侧		输出侧		电机的转向
5" 出脚	6" 出脚	2" 出脚	10" 出脚	
1	1	0	0	停止
0	1	0	1	正转
1	0	1	0	反转
0	0	0	0	停止

表 1 驱动电机 IC 引脚电平关系

为便于设计软件,一般首先画出输入输出分配简图。如图 3 所示,一目了然,很清楚地展示输入输出的分配。

软件流程图如图 4 所示。

CPU 内部寄存器的分配如表 2 所示。

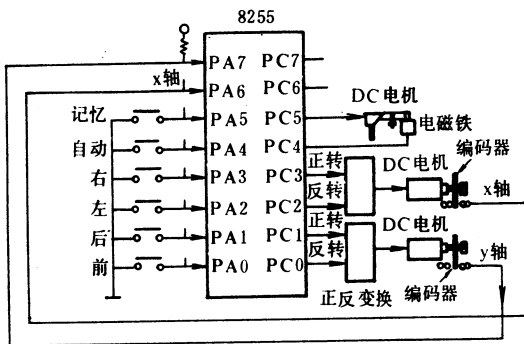


图 3 输入输出框图

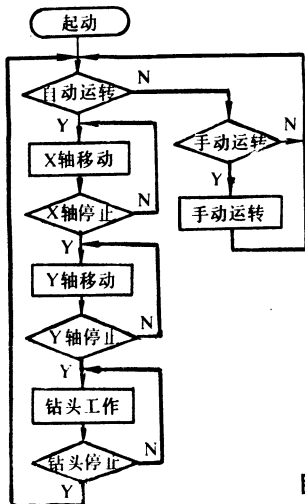


图 4 软件流程图

微型钻床 X 轴、Y 轴丝杠拖动工作台移动 1mm, 需要 100 脉冲(DC 电机一转有 4 个脉冲, 电机转 25 才移动 1mm)。

如果 A 点至 B 点为 10mm, 则需要 1000 个脉冲。变换为 16 进制数:

$$1000 \div 256 = 3 \text{ 余 } 232$$

$$232 \div 16 = 14 \text{ 余 } 8$$

则为 3E8H。

寄存器名	任务
A 累加器	与各端口进行数据交换
B 寄存器	暂时存储钻孔位置数据 0: 正转 1: 反转
C 寄存器	延时计数器 暂时寄存钻孔位置的数据
D 寄存器	暂时寄存延时数据 电机没有脉冲输入 电机有脉冲输入
E 寄存器	暂时寄存延时数据
H、L 寄存器	以寄存器对表示寄存器地址

表 2 寄存器任务分配

X 轴电机、Y 轴电机正转、反转的标志位由 B 寄存器的第 7 位决定。

钻孔结束数据为 FFH(参照表 3)

按上述规定进行软件设计的示例如表 4。

地址	数据								
	7	6	5	4	3	2	1	0	
00DC									X轴沿 方向移动 mm
00DE									
00DF									X轴沿 方向移动 mm
00E0									
00E1									X轴沿 方向移动 mm
00E2									
00E3									X轴沿 方向移动 mm
00F0									
00F1									Y轴沿 方向移动 mm
00F2									
00F3									Y轴沿 方向移动 mm
00F4									

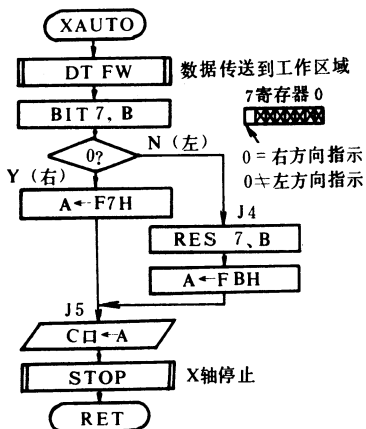
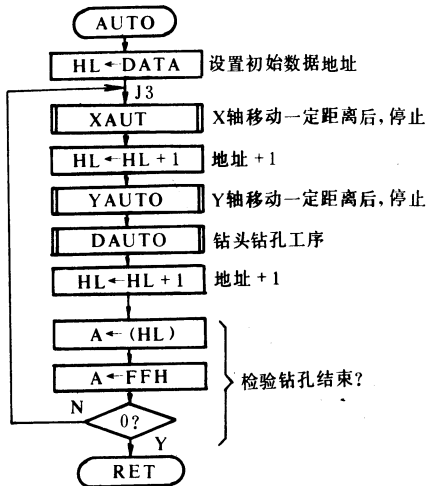
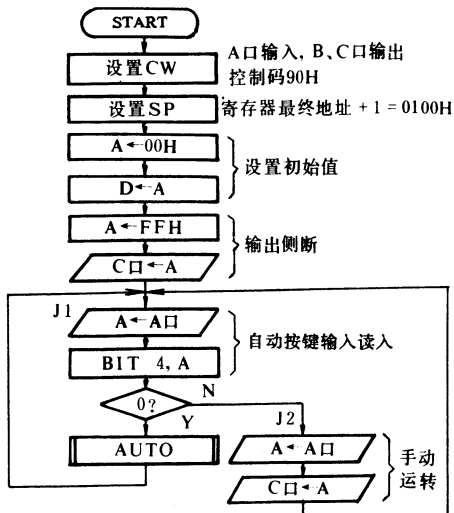
表 3 内存存储方法

表中从 00H 地址到 DBH 地址是基本程序。DCH 地址到 F3H 地址写入孔位数据。钻孔结束数据 FFH 放入 F4H 地址。F5H 地址以后为堆栈。

照图 2 连接硬件,将程序写入、检验无差错,则启动中央处理部件的复位开关,执行程序。按动标注有“前”、“后”、“左”、“右”字样的按键。手动控制调整工件与钻头相对位置进行钻孔。然后,再按动标有“自动”的按键,开始按预定程序自动进行钻孔。

微型钻床为理想的装置。有订购者请与天津纺织工学院机械系 高殿斌同志联系。邮码:300160 电话:412833 转 983

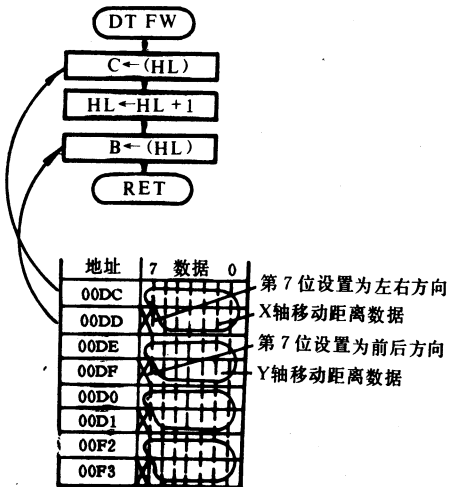
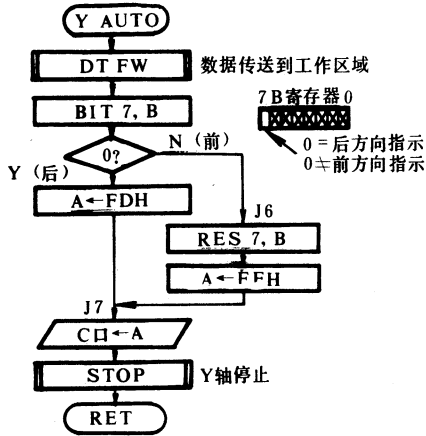
表 4 微型钻床程序示例(1,2,3)  
表 4(1)



标号	助记符	地址	机器语	注释
START	LD A,190H	0000	3E 90	
	OUT(03H),A	0002	D3 03	设置CW
	LD SP,0100H	0004	31 00 01	设置SP
	XOR A	0007	AF	
	LD D,A	0008	57	设置初始值
	LD A,FFH	0009	3E FF	
	OUT(02H),A	000B	D3 02	输出全部为断状态
J1	IN A,(00H)	000D	DB 00	读入自动按钮输入
	BIT 4,A	000F	CB 67	
	JP NZ,J2	0011	C2 1A 00	
	CALL AUTO	0014	CD 21 00	
	JP J1	0017	C3 0D 00	
J2	IN A,(00H)	001A	DB 00	手动运转
	OUT(02H),A	001C	D3 02	
	JP J1	001E	C3 0D 00	
AUTO	LD HL,00DCH	0021	21 DC 00	设置初始数据地址
J3	CALL XAUTO	0024	CD 36 00	驱动X轴
	INC HL	0027	23	地址 + 1
	CALL YAUTO	0028	CD 4D 00	驱动Y轴
	INC HL	002E	23	地址 + 1
	LD A,(HL)	002F	7E	
	CP FFH	0030	FE FF	检验钻孔结束
	JP NZ,J3	0032	C2 24 00	
	RET	0035	C9	
XAUTO	CALL DTFW	0036	CD 64 00	数据传送到工作区域
	BIT 7,B	0039	CB 78	检验正转、反转
	JP NZ,J4	003B	C2 46 00	
	LD A,F7H	003E	3E F7	指示X轴向右
J5	OUT(02H),A	0040	D3 02	
	CALL STOP	0042	CD 68 00	
	RET	0045	C9	
J4	RES 7,B	0046	CB B8	指示X轴向左
	LD A,FBH	0048	3E FB	
	JP J5	004A	C3 40 00	
YAUTO	CALL DTFW	004D	CD 64 00	数据传送到工作区域
	BIT 7,B	0050	CB 78	检验正转、反转
	JP NZ,J6	0052	C2 5D 00	
	LD A,FDH	0055	3E FD	指示Y轴向后
J7	OUT(02H),A	0057	D3 02	
	CALL STOP	0059	CD 68 00	
	RET	005C	C9	
J6	RES 7,B	005D	CB B8	指示Y轴向前
	LD A,FEH	005F	3E FE	
	JP J7	0061	C3 57 00	
DT FW	LD C,(HL)	0064	4E	传送数据
	INC HL	0065	23	地址 + 1
	LD B,(HL)	0066	46	传送数据
	RET	0067	C9	
STOP	LD E,A	0068	5F	
J8	IN A,(00H)	0069	DB 00	检验X轴、Y轴
	BIT 0,E	006B	CB 43	
	JP NZ,J9	006D	C2 7A 00	
J11	BIT 7,A	0070	CB 7F	Y轴脉冲输入有效
J12	JP NZ,J1	0072	C2 87 00	
	RES 0,D	0075	CB 82	
	JP J8	0077	C3 69 00	
J9	BIT 1,E	007A	CB 4B	
	JP NZ,J10	007C	C2 82 00	
	JP J11	007F	C3 70 00	
J10	BIT 6,A	0082	CB 77	X轴脉冲输入有效
	JP J12	0084	C3 72 00	
J13	BIT 0,D	0087	CB 42	
	JP NZ,J8	0089	C2 69 00	
	SET 0,D	008C	CB C2	

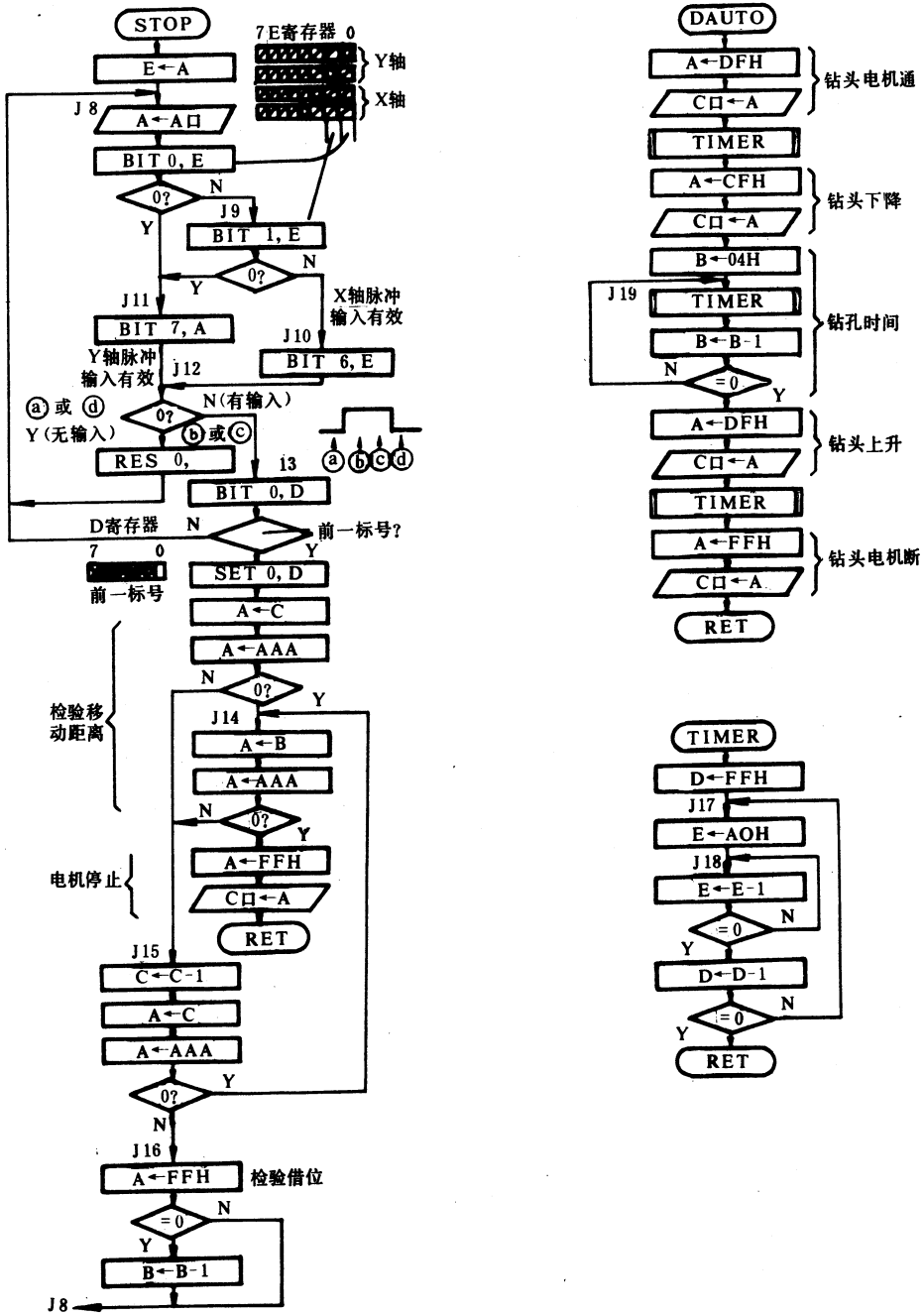


表 4(2)



标号	助记符	地址	机器语	注释
	LD A, C	008E	79	
	AND A	008F	A7	
	JP NZ, J15	0090	C2 9D 00	检验移动距离
J14	LD A, B	0093	78	
	AND A	0094	A7	
	JP NE, J15	0095	C2 9D 00	
	LD A, FFH	0098	3E EF	驱动电机停止
	OUT(02H), A	009A	D3 02	
	RET	009C	C9	
J15	DEC C	009D	0D	
	LD A, C	009E	79	
	AND A	009F	A7	
	JP NZ, J16	00A0	C2 A6 00	移动距离运算
	JP J14	00A3	C3 93 00	
J16	CP FFH	00A6	FE FF	检验借位
	JP NZ, J8	00A8	C2 69 00	
	DEC B	00AB	05	
	JP J8	00AC	C3 69 00	
DAUTO	LD A, DFH	00AF	3E DF	钻头电机接通
	OUT(02H), A	00B1	D3 02	
	CALL TIMER	00B3	CD CF 00	
	LD A, CFH	00B6	3E CF	钻头下降
	OUT(02H), A	00B8	D3 02	
	LD B, 04H	00BA	06 04	
J19	CALL TIMER	00BC	CD CF 00	钻孔时间
	DEC B	00BF	05	
	JP NZ, J19	00C0	C2 BC 00	
	LD A, DFH	00C3	3E DF	钻头上升
	OUT(02H), A	00C5	D3 02	
	CALL TIMER	00C7	CD CF 00	
	LD A, FFH	00CA	3E FF	钻头电机断
	OUT(02), A	00CC	D3 02	
	RET	00CE	C9	
TIMER	LD D, FFH	00CF	16 FF	
J17	LD E, AOH	00D1	1E A0	
J18	DEC E	00D3	1D	
	JP NZ, J18	00D4	C2 D3 00	
	DEC D	00D7	15	
	JP NZ, J17	00D8	C2 D1 00	
	RET	00DB	C9	
DATA		00DC	00	X1 的数据
		00DD	00	
		00DE	00	Y1 的数据
		00DF	00	
		00E0	00	X2 的数据
		00E1	00	
		00E2	90	Y2 的数据
		00E3	81	
		00E4	F4	X3 的数据
		00E5	81	
		00E6	00	Y3 的数据
		00E7	00	
		00E8	F4	X4 的数据
		00E9	81	
		00EA	00	Y4 的数据
		00EB	00	
		00EC	00	X5 的数据
		00ED	00	
		00EE	90	Y5 的数据
		00EF	01	
		00F0	F4	X6 的数据
		00F1	01	
		00F2	00	Y6 的数据
		00F3	00	
		00F4	FF	钻孔结束

表 4(3)





# 计算机语音输出功能的开发与应用(下)

马钢南山铁矿技术科(243033) 陈竹林

## 五、语音数据采集程序

由于语音数据的采集与处理速度很快(大于每秒 8000 次),采集程序必须用汇编语言编制。该程序的文件名是“YYCJ.BASM”,图 4 是该程序的工作流程图。实际应用时,可将其作为 BASIC 语言的一个内存映象文件,由 BASIC 程序调用执行,在运行过程中有任何击键动作都将停止采集返回主程序。它有三个入口参数,A%是延时值,可控制采样的速率;B%是数据区段地址,指示语音数据在内存中存放的起始位置;C%是已采语音数据的字节总数。其

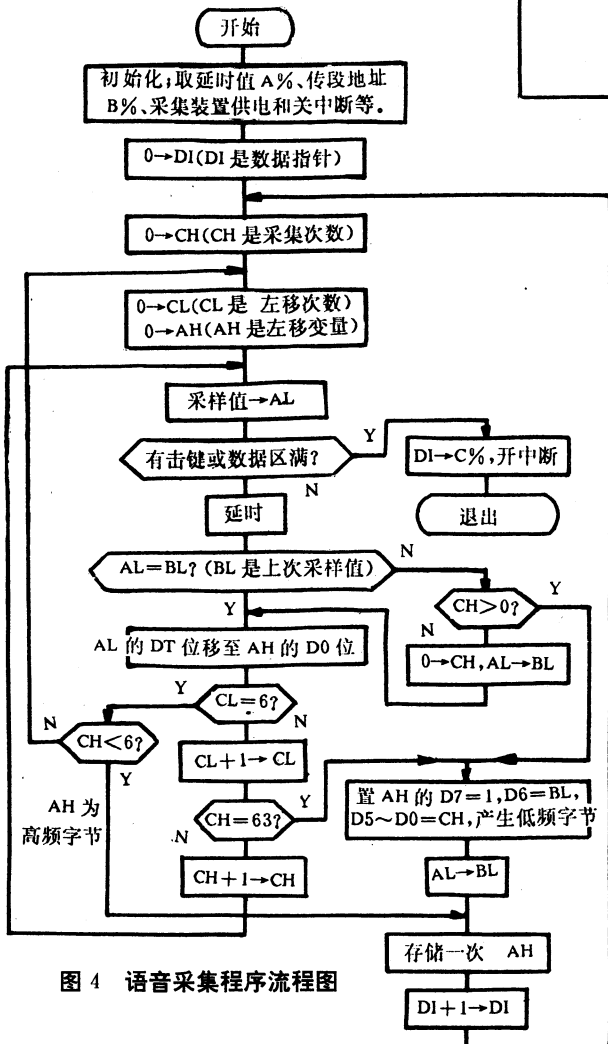


图 4 语音采集程序流程图

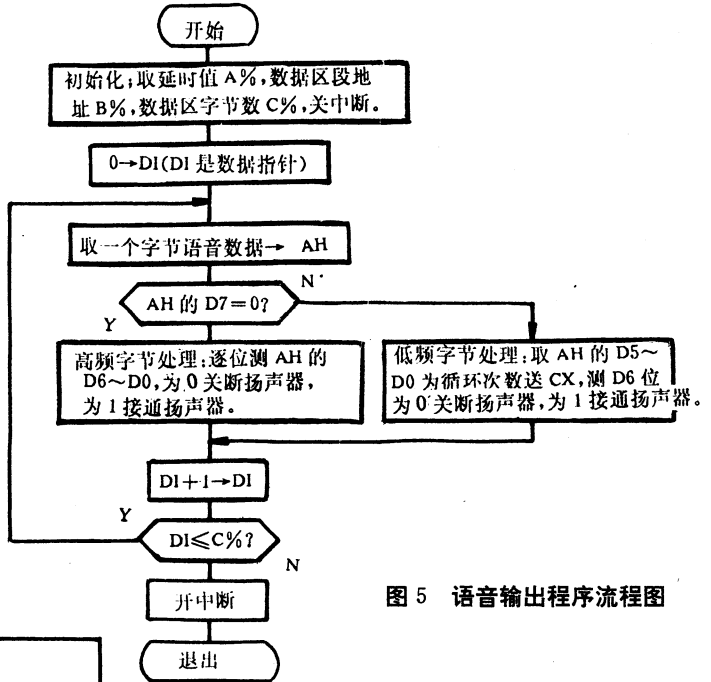


图 5 语音输出程序流程图

中 A%是由主程序传给采集程序的,而 B%和 C%是采集完成后由采集程序传给 BASIC 主程序的。

## 六、语音输出程序

语音输出程序的作用是,将内存中经压缩的语音数据还原后驱动扬声器,完成语音输出的功能。图 5 是语音输出程序的工作流程,其入口参数 A%、B%和 C%都是由 BASIC 主程序传送的。其中 A%是放音延时值,可控制放音速度的快慢;B%是语音数据区的段地址,可控制从内存中取语音数据的起始位置;C%是语音数据区的长度,可控制每次放音取语音数据的字节数。

## 七、语音数据库建立程序

在一个应用程序中需要语音输出单字或字串是有限的,将这些单字或字串的语音数据按一定的顺序存放在一个文件中,这个文件就是语音数据库。为了能快速找到每个单字或字串在语音数据库中的起始位置,还要建立一个语音数据索引文件。程序二是语音数据库及索引文件建立程序。其中语音数据库文件名是“YYSJ.DAT”,其索引文件名是“YYSJ.IND”、“YYCJ.BAM”和“YYSG.BAM”分别是语音采集与语音输出的 BASIC 内存映象汇编程序,“DZSJ.DAT”是单字或字串语音数据暂存文件。

程序二:

```

0 '语音数据库建立(JLYK.BAS)
10 '单字或字串输入与语音数据采集
20 CLS;DIM A$(100);I=1;L=1
25 PRINT"请逐一输入需要发声的单字或字串,输入
 END 结束"
30 PRINT STR$(I);";":H=CSRLIN;L=L+16;IF
 L>70 THEN L=1;H=H+1
40 INPUT";T$;IF T$="END" OR T$="end"
 THEN ZS=I-1;GOTO 60
50 LOCATE H,L;A$(I)=" "+T$;I=I+1;GOTO 30
60 DEF SEG=&H7000;BLOAD"YYCJ.BAM",0
70 CLS;PRINT"对着话筒朗读下列单字或字串,字与字
 之间停顿1秒钟左右!"
80 FOR I=1 TO ZS;PRINT STR$(I);";";MID$(A
 $(I),5),;NEXT I
85 PRINT;PRINT;INPUT"准备好按回车开始!按空格
 结束";T$
90 PRINT"开始朗读!";A%=5;CJ%=0;CALL
 CJ%(A%,B%,C%);T$=INPUT$(1)
100 '数据分隔处理
105 PRINT"现在作字分隔处理,请稍候"
110 DEF SEG=B%;J=0;FOR I=1 TO ZS
115 PRINT I;";";MID$(A$(I),5),
120 IF PEEK(J)=&HBF THEN K=0
130 K=K+1;J=J+1;IF K>10 THEN SW=J;MID$(
 (A$(I),1)=MKI$(SW) ELSE 120
140 IF PEEK(J)<>&HBF THEN K=0
150 K=K+1;J=J+1
160 IF K>5 THEN CD=J-SW;MID$(A$(I),3)=
 MKI$(CD) ELSE 140
170 NEXT I;PRINT
200 '语音核对
210 DEF SEG=&H7000;BLOAD"YYSC.BAM",0
215 A%=9;SC%=0;DZ%=B%
220 FOR I=1 TO ZS;T$=A$(I);PRINT
 STR$(I);";";MID$(T$,5),
230 SW=CVI(MID$(T$,1,2));CD=CVI
 (MID$(T$,3,2))
240 B%=DZ%+SW/16;C%=CD;CALL SC%
 (A%,B%,C%)
250 NEXT I;PRINT;PRINT"1:存盘 2:重读 3:再听"
260 INPUT T$;IF T$="1" THEN 300
270 IF T$="2" THEN 70
280 IF T$="3" THEN 220 ELSE BEEP;GOTO 260
300 '数据整理存盘
310 DW=0;FOR I=1 TO ZS;T$=A$(I)
320 SW=CVI(MID$(T$,1,2));CD=CVI
 (MID$(T$,3,2))
330 DEF SEG=DZ%;BSAVE"DZSJ.DAT",SW,CD
335 T%=INT(DW/16);IF T%<>DW/16
 THEN T%=T%+1
340 DEF SEG=DZ%+T%;BLOAD"DZSJ.DAT",0
350 MID$(A$(I),1)=MKI$(T%);DW=16*T%+
 CD
355 NEXT I;KILL"DZSJ.DAT"

```

```

360 DEF SEG=DZ%;BSAVE"YYSJ.DAT",0,DW
370 OPEN"YYSJ.IND" AS #1 LEN=12;FIELD
 #1,12 AS N$
380 FOR I=1 TO ZS;LSET N$=A$(I);PUT
 #1,I;NEXT I
390 CLOSE #1
990 END

```

程序 20~50 行完成单字或字串的输入,根据提示逐一从键盘输入待处理的单字或字串,并依次赋给 A\$( )数组,当输入的字串是"END"时结束输入。60~90 行完成语音采集,屏幕首先显示刚输入的全部单字或字串,提示使用者对着话筒逐一朗读,全部朗读完毕后击空格键结束采集。为便于程序对语音数据作分隔处理,要求字与字之间停顿 1 秒钟左右。语音数据在内存中的段地址在 B% 中,总字节数在 C% 中。第 100~170 行完成语音数据分隔,逐一找出每个单字或字串的起始地址和字节数,并记录在对应 A\$( )变量前面。第 200~280 行完成语音核对,逐一在屏幕上显示单字或字串,同时通过扬声器输出其读音,供使用者监听核对分隔的正确与否。第 300~390 行完成数据的紧缩与存盘,先去掉字间停顿期间采集到的大量 BFH 数据,然后将紧缩后的语音数据以"YYSJ.DAT"文件存盘,将记载单字或字串起始位置和字节长度信息的 A\$( )数组以"YYSJ.IND"文件存盘。到此为止,语音数据库及索引文件的建立就完成了。

#### 八、一个应用语音输出功能的例子

为了进一步说明语音输出功能的开发过程。现在举一个在 IBM-PC/XT 计算机上实现语音报时功能的实际应用例子。开发过程如下。

第一步:将语音采集装置(即话筒电路)联接在计算机的 RS-232C 接口上,启动计算机进入 CCDOS 汉字操作系统。

第二步:调入解释 BASIC 系统,并运行语音数据库建立程序"JLYK.BAS"。根据提示从键盘上依次输入 14 个单字"1、2、3、4、5、6、7、8、9、0、拾、点、整、分"和 1 个字串"现在是",输一个按一次回车键,最后键入"END"结束输入。在屏幕重新显示这些单字或字串并提示"准备好按回车开始!按空格结束"后,手握话筒,口正对着受话面,按一下回车键后大声朗读这些字。注意单字之间停顿 1 秒左右,而字串"现在是"字间不应停顿。全部朗读完毕,按空格键结束采集。

计算机经过一段时间的分隔处理后,在屏幕上逐一显示这些字,并驱动扬声器输出每个单字和字串的读音。此后,屏幕提示"1:存盘 2:重读 3:再听",如果字音关系正确、听音效果满意,键入"1",计算机对语音数据作紧缩处理后存盘,生成语音数据库"YYSJ.DAT"和索引文件"YYSJ.IND"。如果读音不正确或不满意,键入"2"回到前面重新朗读。键入"3"可再听一遍。

第三步:调入并运行计算机报时程序"JQBS.BAS",程序三是该程序的语句清单。

### 程序三:

```

10 '计算机报时(JQBS. BAS)
20 DIM A$(200),KEY OFF;ZS=15
25 DEF SEG=&H7000;BLOAD "YYSC. BAM",0
35 DZ%=&H7010;DEF SEG=DZ%;BLOAD "YYSJ. -
 DAT",0
40 OPEN "YYSJ. IND" AS #1 LEN=12;FIELD #1,12
 AS N$
45 FOR I=1 TO ZS;GET #1, I;A$(I)=N$;NEXT I
50 SCREEN 2;SCREEN 1;COLOR 9,0
55 LOCATE 2,10;PRINT "计算机语音报时演示软件"
60 LOCATE 6,15;PRINT TIME$
65 IF INKEY$=" " THEN 60
70 GOSUB 100;GOTO 60
100 '语音字符串形成
110 T$=TIME$;A$=MID$(T$,1,1);B$=MID
 $(T$,2,1)
120 C$=MID$(T$,4,1);D$=MID$(T$,5,1);
 Y$=""
130 IF A$="0" THEN Y$=B$+"点,"
135 IF A$="1" AND B$="0" THEN Y$="
 拾,点,"
140 IF A$="1" AND B$>"0" THEN Y$="
 拾,"+B$+"点,"
145 IF A$>"1" AND B$="0" THEN Y$="
 A$+"拾,点,"
150 IF A$>"1" AND B$>"0" THEN Y$="
 A$+"拾,"+B$+"点,"
155 IF C$="0" AND D$="0" THEN Y$="
 Y$+"整";GOTO 180
160 IF C$>"0" AND D$="0" THEN Y$="
 Y$+C$+"拾,分";GOTO 180
165 IF C$="0" AND D$>"0" THEN Y$="
 Y$+"0,"+D$+"分";GOTO 180
170 Y$=Y$+C$+"拾,"+D$+"分"
180 Y$="现在是,"+Y$;GOSUB 200
190 RETURN
200 '字符串语音输出
210 A$=Y$
220 V=INSTR(A$,".");IF V=0 THEN T$="
 A$";GOTO 230
225 T$=MID$(A$,1,V-1);A$=MID$(A$,V+
 1)
230 FOR I=1 TO ZS;E$=A$(I);F$=MID$(E$,
 5)
240 S=INSTR(F$,T$);IF S=0 THEN 270
250 T=CVI(MID$(E$,1,2));CD=CVI(MID$
 (E$,3,2))
260 A%=9;B%=DZ%+T;C%=CD;SC%=0;DEF
 SEG=&H7000;CALL SC%(A%,B%,C%)
270 NEXT I;IF V>0 THEN 220
290 RETURN

```

运行该程序,在屏幕中间显示当前的时间,任何时候敲任何键,计算机都用语音报告一次当前的时间。如“现在是八点二十五分”、“现在是九点整”等。

### 九、几点说明

1. 在早期的 XT 计算机上使用时,如感到扬声器的音量太小,可按图 6 在 RS-232 口上外接一个扬声器,使输出音量大大增加。图中变压器可选用普通收音机中的输出变压器。此时应将语音采集“YYCJ. BAM”和语音输出“YYSC. BAM”汇编程序中控制扬声器通、断电的程序行换成下列程序行:

```
通电:MOV DX,3FCH
```

```
MOV AL,01H
```

```
OUT DX,AL
```

```
断电:MOV DX,3FCH
```

```
MOV AL,02H
```

```
OUT DX,AL
```

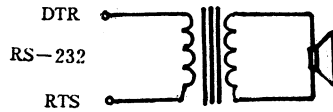


图 6 外接扬声器

2. 由于本方案在语音输出时只能再现语音的音调特征,而对声色和响度无法再现,故存在一定的失真。因此在语音采集过程中,朗读要尽可能清晰,最好请音调较高的女士朗读。

3. 由于不同档次的计算机运行速度不同,对延时值 A% 的选择也不尽相同。XT 计算机速度较低,A% 应小些(一般取 A%=1~5 较为合适),在 286、386 等速度较高的计算机上 A% 可选大一些。A% 越小采集或放音速度越快,音质越好,但语音数据量越大,占用内存或磁盘的空间也越大,要权衡考虑。

(接自 45 页)

```

529 PRINT "Φ1 故障! 转时钟测试模块";GOTO 585
520 A$="脚脉冲 P,高 H 或低 L 电平:"
523 PRINT"G1-6";A$;:INPUT B$
525 IF B$="L" THEN PRINT"G1 坏!";GOTO 585
530 PRINT "请测 F12 以下引脚:"
537 R=192
540 FOR I=15 TO 9 STEP-1
545 POKE D1,R
550 IF M=2 AND(I=12 OR I=10)THEN 560
555 PRINT I;A$;:INPUT B$
557 IF B$<>"P"THEN 580
560 R=R+8;NEXT I
565 IF M=2 THEN 585
570 POKE D1,R;PRINT 7;A$;:INPUT B$
575 IF B$="P"THEN 585
580 PRINT "F12 译码器坏!"
585 INPUT "请记录后回车返回!";A$
590 RETURN

```

本系统由于通过仿真接口卡代替故障机 CPU 进行读写控制,其最大的优点是能产生静态或动态的地址和数据,对于逻辑电路的测试极为方便。



# F BASIC 语言的游戏程序编程技巧

## 第三讲 程序结构和程序框图

山东苍山机械电子化学工业局(277700)于春

### 一、程序结构

程序结构一般分为三大类：

- 模块结构
- 顺序结构
- 混合结构

1. 模块结构也叫结构化程序。它根据总体需要和系统配置,把一个程序分成具有某些特定功能的程序段,每个程序段形成一个子程序,称为功能模块。每个模块相对独立地完成一定的任务。它可以是计算分析战斗场面;可以是游戏主人公的跳跃飞腾;也可以是一段音乐伴奏等等。然后再加一个调用子程序的主程序——称为管理程序。管理程序在需要时可调用所有模块。另外,各模块之间也可以互相调用。不难看出,一个完整的程序就是由管理程序和若干模块组成的。这种结构的优点是:(1)由于模块之间相对独立,所以在开发程序时,可以分头编写。程序的调试纠错较容易,因而编程周期短。(2)由于模块的相对独立性,能有效地防止错误在程序中的扩散与蔓延,因而提高了系统的可靠性和质量。(3)由于各模块功能明确、相对独立,通用性强,因此便于其它程序的借用和向其它语言移植。缺点是:编程时要加一定的限制;调用时要考虑有关变量的赋值;调用后要运算结果进行恰当的处理等。因此,结构化程序的程序量较大,一般在大型、复杂程序中使用。

2. 顺序结构也叫非结构化程序,它与结构化程序相反,采用顺序编写的方法。程序中的转向,一般使用 GOTO 语句实现,条件转移语句用得较多。这种结构的优点是:(1)程序顺序编出,变量之间不易混淆。(2)省去了程序的链接调用操作,从而程序量较小。(3)由于 GOTO、IF—THEN 语句使用较多,程序不易阅读,从而使程序的保密性加强。缺点是:只能一个人编写程序,编程周期长;程序的调试改错较困难,一处有错殃及整个程序不能运行;由于出现 GOTO 网络,不易阅读理解,程序的借用移植性能较差。这种结构一般在简单程序中使用。

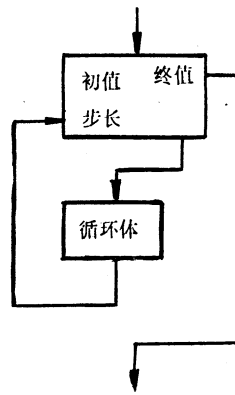
3. 混合结构则综合了以上两种结构的优点;它根据游戏的需要,对模块化结构程序进行了灵活处理,把那些整个游戏中只调用一次的模块则取消模块格式,直接放入主程序中。对调用最频繁的模块,则尽量安置在行号较小的位置,甚至移到主程序之前,以提高程序的执行速度。(由于电脑在执行 GOTO、GOSUB 语句时,总是从最小行号开始搜索,直到目标行号。)今后我们介绍的程序多是这种结构的程序。但是为使层次清楚,便于读者阅读理解,开始仍以模块化结构为主。

### 二、程序框图

什么叫程序框图呢?我们说,程序框图是一种流程图,它是用各种几何图形及文字说明来直观地描述电脑计算执行过程的有向图。程序框图常用的符号如下:

- (1) ○ 椭圆形框:表示程序的开始或结束。
- (2) □ 矩形框:表示赋值或完成运算操作。
- (3) ◇ 菱形框:判断某个关系式是否成立。若关系式成立则程序向 Y(Yes)方向进行;否则向 N(No)方向进行。
- (4) ▱ 斜框:表示输入输出。
- (5) → :箭头:表示流程的流向。
- (6) @ 小圆圈内一个字母或数字:表示流程图的接点。有些流程图很长或很复杂,往往一页纸画不下,需转到下一页时,则把一些地方断开,在断开的两头画圈,圈内标同样的符号,表示这两点是连在一起的。

(7) FOR 循环框图:循环变量初值占矩形框的左上角;终值写在右上角;步长在左下角;从右下角引出进入循环体;从终值处转出 FOR 循环。



程序框图的符号还有一些,最常用的是这七种,其它符号在以后用到时再介绍。

一般程序框图分两大类:

- 粗结构框图
- 细结构框图

1. 粗结构框图也叫功能模块结构图。它描述了整个软件系统由哪些模块组成,也说明了模块之间的调用关系。一般有图 1 的形式。图 1 中,方框表示模块,框内写有模块的名称及模块的主要功能;箭头表示上层模块对下层模块的调用;菱形符号表示有条件调用;

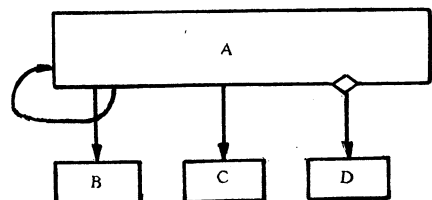


图 1

弧形箭头表示循环调用。图 1 说明 A 模块循环调用模块 B、直接调用模块 C、有条件地调用模块 D。

2. 细结构框图俗称流程图。它是对程序的精细描述。它给出模块功能的实现过程。它主要包括模块内部采用的计算方法、数据的存储结构和输入输出等方面的设计步骤。对于第二讲中介绍的例题三，我们可以画出功能模块结构图如图 2：

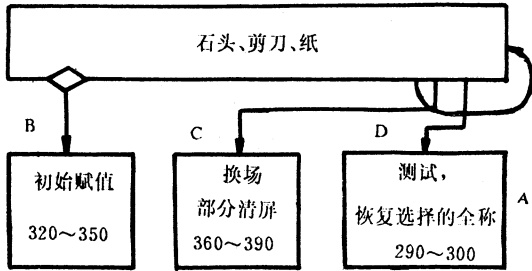


图 2、石头、剪刀、纸的模块结构图

例三是一个典型化的混合结构程序。A 为主控程序；B、D 为循环调用模块；C 为条件调用模块，仅在每次游戏开始时才调用。实际上初始赋值模块 C 也可以放到 A 程序的 55 行以后作为管理程序的一部分。

模块结构图与流程图有本质的区别。模块结构图反映了程序的层次特性，即某个模块要调用那些模块，哪些模块又调用什么模块。流程图则反映程序执行的过程特性，即先执行哪一部分，再执行哪一部分。两者是全局和局部的关系。

在一、二讲的三个示例中，介绍了游戏程序中加入音响的方法、游戏背景及卡通图案、色彩的变换技巧和键盘输入的处理。本讲再介绍如何使用操纵器控制卡通运动，卡通运动越界的判断处理及卡通随机自动连续运动的处理方法。

#### 例四“企鹅看外婆”游戏

故事情节：“企鹅太太带着五个孩子去看外婆。去外婆家要穿过一条马路。马路上车辆奔流不息。她只能瞅准车辆运行空隙，一次抱一个小企鹅到马路对过，往返五次，把孩子们都送过马路后，才能去外婆家。”

根据故事情节，我们可以这样规划游戏结构。游戏画面是一条很宽的横向马路。马路上有七辆汽车在不同的位置，以不同的速度运行。在马路的中段画出一段较窄的纵向人行道。人行道的北头（屏幕上方）有五个小企鹅。企鹅太太在人行道的南端（屏幕下方）。选一号操纵器控制企鹅太太前后左右运动。当运行到北端时，抱起一个小企鹅，北端还剩四只。企鹅返回到南端后，则南端多出一只小企鹅。直到五个小企鹅全部转移到人行道南端则游戏结束。游戏过程中加入企鹅太太

的脚步声、抱起企鹅和放下企鹅的声音、把企鹅安全转移到马路对过后的祝贺声和与汽车相撞的声音，共五种简单音响。为增加游戏难度，加入了时间限制。

根据游戏结构，首先划分功能模块。然后画出模块结构框图，见图 3。

各模块功能如下：

- 模块 A：管理程序。它完成游戏过程的控制。
- 模块 B：初始化模块。画出马路、人行横道。马路为横向，宽 12、长 27。人行道为纵向，宽 4、长 14。在人行道下端显示企鹅太太。上端显示五个小企鹅。七辆汽车从右向左行驶，车色、车速不同。画面如下：

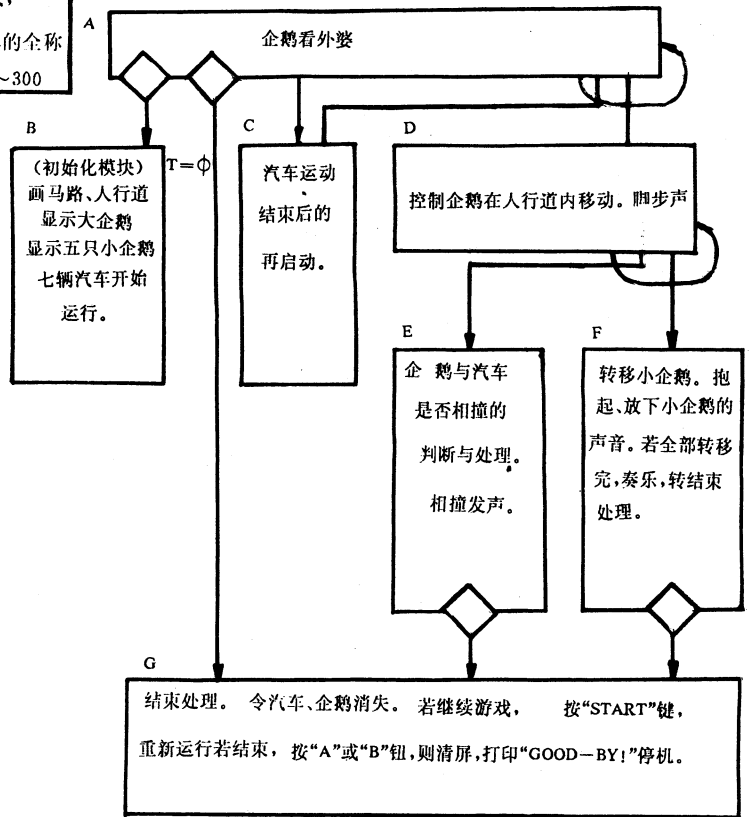


图 3、“企鹅看外婆”模块结构图

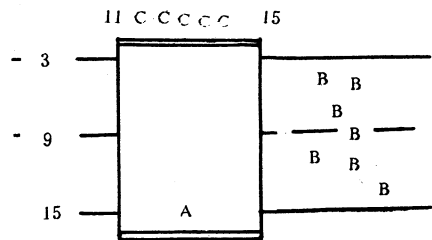


图 4、企鹅看外婆画面

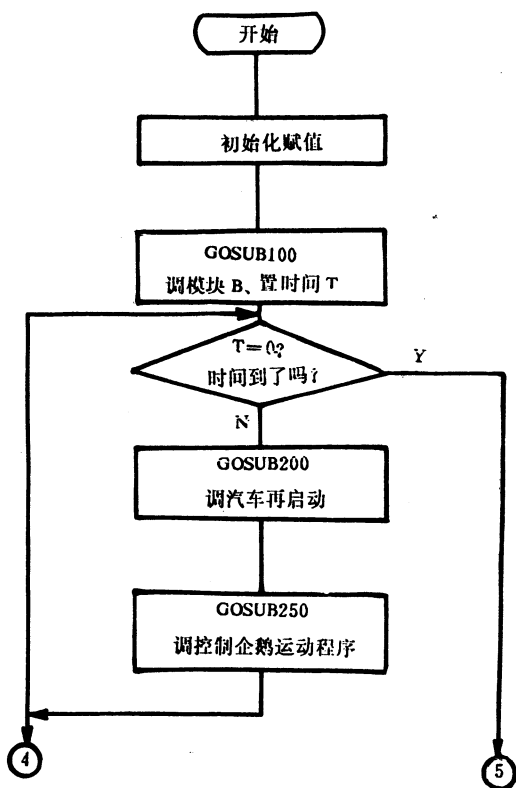


图 9

图 4 中, A 为企鹅太太; B 为汽车; C 为小企鹅。座标分别为: A (120, 144); B [250, RND(60) + 56]; C (104, 20)~(136~20)

- 模块 C: 汽车再启动。它随时检测七辆汽车的运行情况, 若有停止运行的汽车, 则立即启动, 令其运行。
- 模块 D: 用操纵器控制企鹅运动, 并随着企鹅的运动发出脚步声。它循环调用模块 E、F。
- 模块 E: 判断企鹅是否与汽车相撞。若相撞则发声, 转结束处理模块。
- 模块 F: 完成小企鹅的转移。抱起和放下小企鹅时伴有音响。若五个小企鹅全部转移则奏乐祝贺, 转结束处理。
- 模块 G: 结束处理模块。用“START”键控制继续游戏; 用 A、B 钮控制结束。结束时打印“GOOD-BY!”。停机。

对照功能模块, 可分别画出各模块的流程图(见图 5~图 9)。

根据程序框图可对应编写出每个模块的详细程序, 然后把各模块的程序安排适当的行号, 合并后, 就得出“企鹅看外婆”游戏的完整程序。该程序较简单, 直接给出结构化程序清单(为便于阅读、理解, 程序清单

中个别部分加入了中文说明)。

“企鹅看外婆”结构化程序

```

5 REM "NO. 4 PENGUINS VISIT GRANDMOTHER"
10 CLS; CLEAR; SP. O. ; CG. RND(2), 0; T=800
15 PL. "V15Y2T1; V15T1"
20 GOS. 100(画马路、显示汽车、企鹅)
25 T=T-1; LOC. 22, 21; P. T; "□□"; IF T=0 T. 60
30 GOS. 200(汽车的再启动)
40 GOS. 250(控制企鹅运动)
50 GOTO 25
60 ERA 0, 1, 2, 3, 4, 5, 6, 7; F. I=1 TO 5; SP. I; N.
65 CLS; LOC. 0, 20; P. "Press START to continue"
70 S=STRIG(0); IF S=0 G. 70
75 IF S=1 T. RUN
80 CLS; LOC. 8, 8; P. "GOOD-BY!"; LOC. 0, 20
85 E.
100 REM "GAME Picture"(游戏画面)
110 F. I=0 TO 27; IF I<10 OR I>16 T. LOC. I, 9; P. CH.
 (227)
120 P=3; Q=15; IF I>=10 AND I<=16 T. P=1; Q=17
130 LOC. I, P; P. CH. (197); LOC. I, Q; P. CH. (197)N.
140 F. I=2 TO 16; LOC. 10, I; P. CH. (226); LOC. 16, I; P.
 CH. (226); N.
150 F. I=1 TO 5; DE. SP. I, (RND(4), 0, 0, 0, 0)="defg";
 SP. I, (I-1)*8+104, 20; N.
160 DE. M. (7)=SP. (4, 0, 1, 2, 0, 2); POS. 7, 120, 144; M. 7
170 F. I=0 TO 6; DE. M. (I)=SP. (6, 7, I+1, 130, 0, RND
 (4))
180 POS. I, 250, RND(60)+56; M. I; N.
190 RE.
200 REM "Vehicles Restart"(汽车再启动)
210 F. I=0 TO 6
220 IF M. (I)=0 T. POS. I, 250, RND(60)+56; M. I
230 N.
240 RE.
250 REM "Penguin Moves"(企鹅运动)
260 S=0; SX=0; SY=0; K=STICK(0); IF K=0 T. 380
270 IF K=1 T. S=3; SX=1
280 IF K=2 T. S=7; SX=-1
290 IF K=4 T. S=5; SY=2
300 IF K=8 T. S=1; SY=1
310 PX=(XPOS(7)-16)/8; IF PX=10 T. PX=11
320 IF PX=15 T. PX=14
330 PY=(YPOS(7)-24)/8
340 S$=SCR$(PX+SX, PY+SY)
350 IF S=0 OR S$<>" "T. 380
360 DE. M. (7)=SP. (4, S, 2, 4, 0, 2); POS. 7, PX*8+16,
 PY*8+24; M. 7
370 PL. "O1#FORO0#F;R"
380 GOS. 400(与汽车相撞的判断处理)
385 GOS. 450(转移小企鹅的处理)
390 RE.
400 REM "Collides with Vehicle"(碰汽车的处理)
410 IF CR. (7)>=0 T. PL. "O1E0DCBAG; O0#A#G#FE
 #D#C"; GOTO 60(转结束处理)

```



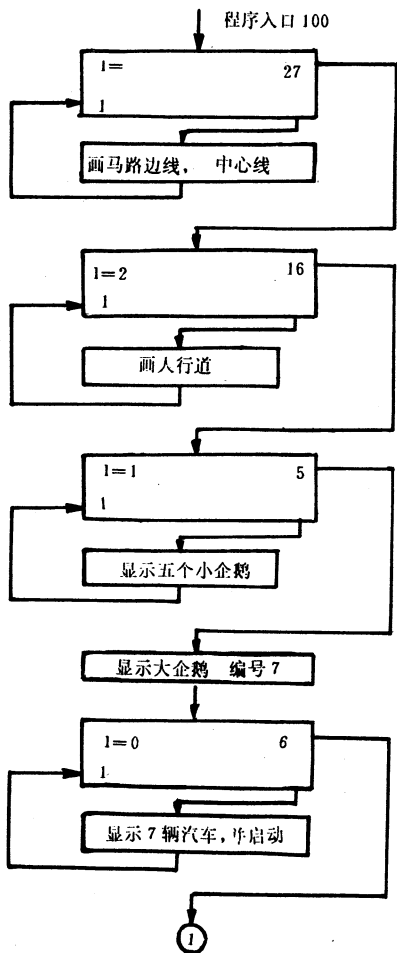


图 5

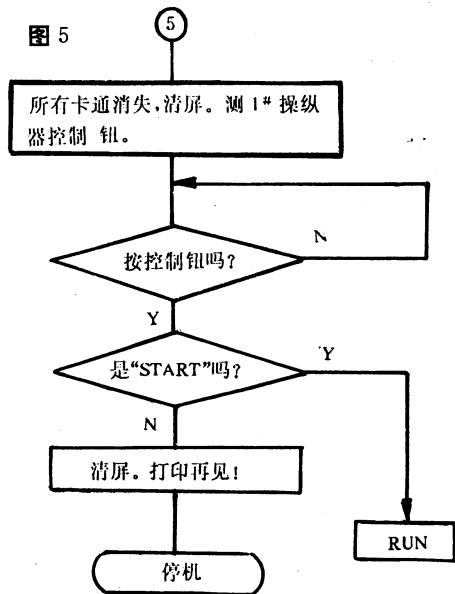


图 8

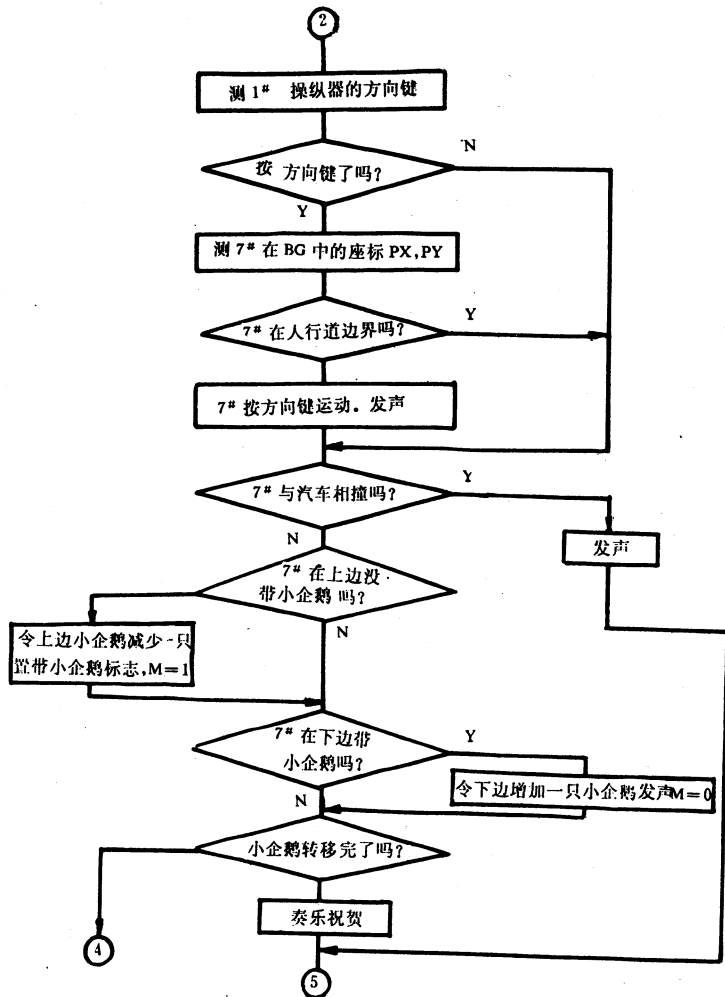


图 7

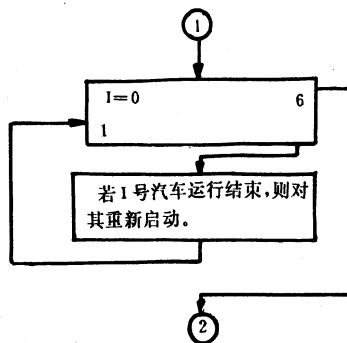


图 6

```

420 RE.
450 REM "Moves Small Penguins"(转移小企鹅)
460 IF PY=1 AND M=0 T. H=H+1;M=1;SP. H;N=1
 PL. "O5C0DEDE;O3C0DEF #G"
470 IF PY=15 AND N=1 T. M=0;SP. H,(H-1)* 8+
 104,152;N=0;PL. "O5 #G0FEDC;O3 #G0 #FE #D #
 C"
480 IF H=5 AND N=0 T. PL. "O2G3GO3CCEECC;O1
 C3EGEGCGC"; PL. "GGAGEDC5O2C3O3C1;CEGEGC
 C1CC";G. 60(转结束处理)
490 RE.
 程序 NO. 4 中某些模块如 D、E、G 是通用模块,可
 作为工具使用。为便于读者对照,下面再给出“企鹅看
 外婆”游戏顺序结构的程序清单。
 “企鹅看外婆”非结构化程序(顺序结构)
5 REM"NO. 5 PENGUINS VISIT GRANDMOTHER"
10 CLS;CLE. ;SP. O. ;T=800;CG. RND(2),0
20 PL. "V15T1;V15T1"
30 F. I=0 TO 27;P=3;Q=15
40 IF I<10 OR I>16 T. LOC. I,9;P. CH. (227)
50 IF I>=10 OR I<=16 T. P=1;Q=17
60 LOC. I,P;P. CH. (197);LOC. I,Q;P. CH(197);N.
70 F. I=2 TO 16;LOC. 10,I;P. CH. (226);LOC. 16,I;P.
 CH. (226);N.
80 F. I=1 TO5;DE. SP. I, (RND(4),0,0,0,0)="defg";SP.
 I, (I-1)* 8+104,20;N.
90 DE. M. (7)=SP. (4,0,1,2,0,2,);POS. 7,120,144;M. 7
100 F. I=0 TO 6;DE. M. (I)=SP. (6,7,I+1,130,0,RND
 (4))
110 POS. I,250,RND(60)+56;M. I;N.
120 T=T-1;LOC. 22,21;P. T;" "
130 IF T=0 T. 340
140 F. I=0 TO 6
150 IF M. (I)=0 T. POS. I,250,RND(60)+56;M. I
160 N.
170 S=0;SX=0;SY=0;K=STICK(80);IF K=0 T. 280
180 IF K=1 T. S=3;SX=1
190 IF K=2 T. S=7;SX=-1
200 IF K=4 T. S=5;SY=2
210 IF K=8 T. S=1;SY=1
220 PX=(XPOS(7)-16)/8;IF PX=10 T. PX=11
230 IF PX=15 T. PX=14
240 PY=(YPOS(7)-24)/8
250 S$=SCR$(PX+SX,PY+SY)
260 IF S=0 OR S$<>" "T. 280
270 DE. M. (7)=SP(4,S,2,4,0,2);POS. 7,PX* 8+16,PY
 * 8+24;M. 7;PL. "O1 #F0RO0 #F;R"
280 IF CR. (7)>=0 T. 330
290 IF PY=1 AND M=0 T. H=H+1;M=1;SP. H;N=1;
 PL. "O5C0DEDE;O3C0DEF #G"
300 IF PY=15 AND N=1 T. M=0;SP. H,(H-1)* 8+
 104,152;N=0;PL. "O5 #G0FEDC;O3 #G0 #FE #D #

```

```

C"
310 IF H=5 AND N=0 T. PL. "O2G3GO3CCEECC;O1
 C3EGEGCGC"; PL. "GGAGEDC5O2C3O3C1;CEGEGC
 C1CC";G. 340
320 G. 120
340 F. I=0 TO7;ERA I;SP. I;N. ;CLS
350 LOC. 0,20;P. "Press START continue"
360 S=STRIG(0);IF S=0 G. 350
370 IF S=1 T. RUN
380 CLS;LOC. 8,8;P. "GOOD-BY!";LOC. 0,20;E.

```

对比程序 NO. 4、NO. 5,显而易见,结构化程序层次分明,便于阅读理解,但程序量较大。顺序结构程序紧凑,阅读理解要困难些。另外 NO. 4 中模块 B 只调用一次,完全可以放入主程序中,从而变为混合结构。不难看出,调整后并不影响程序的层次和易读性。所以说混合结构是较优化的程序结构。

中国计算机学会 联合举办“学装微电脑”函授班  
电子工业出版社

## 第五期招生

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,计算机出了故障如何动手修理,许多人尚缺乏硬件方面的知识。为此,我们开办“学装微电脑”函授班。

1. 招工对象:具有中等文化程度的微电爱好者、中学师生、各行业的技术人员和维修人员。

2. 教学计划与学习要求:函授班为期四个半月(每周6学时,分3个单元)。

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,并提出疑难问题,教师批改作业,解答问题,评定成绩。

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”。

3. 招生日期:从即日起到92年12月31日止。

4. 开课日期:93年1月15日开课。

全部学费275元,包括MP-I型学习机全部元器件,实验板一块,教材(已购MP-I机者只交60元教材及教务费)。

报名办法:请写信到中国计算机学会办公室(地址:北京2704信箱,邮编:100080,联系人:宁伟成)索取“学员登记表”,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零件及教材。



# APPLE— II、CEC— I 故障诊断仿真系统

重庆西南师范大学计算机科学系(630715) 王志刚 张一建

## 维修经验谈

APPLE— II 和中华学习机在我国中、小学以及一些企事业单位仍有相当的数量,但因维修点分布不均,偏远地区的用户遇到微机故障时便感到束手无策,致使坏了的机器一放就是几个月甚至更久,影响正常使用,造成了很大的浪费。

本故障诊断系统是用一台完好的苹果机或中华学习机通过仿真接口卡,模拟故障机的 CPU 进行读写控制,通过人机对话,根据读写情况指示稍懂微机原理的人员作一些逻辑测试后送回状态,然后和正常状态进行比较与推理,并给出故障元器件的位置。

### 一、仿真接口卡原理

接口卡电路如图所示,U1 和 U2 两个锁存器用来生成被测机系统地址,U3 为故障机数据总线的输出寄存器,U4 为故障机数据读出寄存器,这四个寄存器对于宿主机来说都是输入输出端口,其端口地址来自 U5 对某一扩展槽口的 DEV SEL 空间译码,这一地址范围为 \$C0X0~\$C0XF,X=N+8,N 为扩展槽序号,因此 U1 和 U2 的端口地址分别为 \$C0X0 和 \$C0X1,数据输出端口的地址为 \$C0X2,输入端口为 \$C0X3,X 的值依仿真卡插入的槽号而定。由于两机时钟系统不同步,故送往故障机的读写控制信号由一个 D 触发器 U6 产生,该 D 触发器的时钟端连到端口地址译码器 U5 的 Y5 端,当主机复位时,该 D 触发器被置位,Q 输出端为高电平,形成读控制信号,如要进行写操作,只要对 \$C0X4 端口进行两次访问,这样在 Q 端产生一个低电平写控制信号,该信号送往故障机的 R/W 控制线,同时该信号允许 U3 锁存的写数据输出到故障机的数据总线。

故障机的  $\Phi_0$  时钟经反相后送 U4 的锁存触发端,这样当  $\Phi_0$  为 6502 周期的下降沿时,U4 时钟端将获得一个上跳沿的触发信号,将故障机数据总线的读出数据锁入锁存器中。因是异机工作, $\Phi_0$  不同步,为了保证诊断机读取输入寄存器时,故障机不发生锁存操作, $\Phi_0$  经或非门 U8 受控于 U6 的 Q2 端,当开机复位时,此端被置为“1”,允许故障机的  $\Phi_0$  反相通过,此时 U4 可以锁存故障机总线上的数据。当诊断机读输入寄存器时,先访问一次端口 C0X5,禁止锁存,读完后,再访问一次该端口,恢复锁存允许。

这些地址信号,数据信号,控制信号以及地 GND 与一个 40 脚的插头连接,其顺序和 6502 的引脚信号相同,此插头代替故障机的 CPU 芯片插入故障机的 CPU 插座。

### 二、诊断原理

诊断时由检测主机通过 CPU 仿真接口模拟 CPU,

进行地址发送,数据读写等操作。其主要思想是当机器发生故障时,我们无法了解 CPU 以及外部主要电路(包括 ROM、RAM 等)的逻辑状态,不能让 CPU 固定地发送某一地址或读写某一数据,也就不能静态地观察某部分电路的状态。苹果机的主要控制都是来自 CPU,而一旦控制失败,必然是某些正常状态被破坏。仿真接口的地址和数据可以保持不变直到你改变它,这样,我们可以针对某一操作固定地从总线给出产生这一操作的地址、数据和控制等信号,再依据完成这一操作的电路逻辑关系,便可逐元件逐脚地测试是否正常工作,最终找出故障元器件。例如:

对于 ROM 的检测,当故障机和诊断机型一致时,可以逐单元将 ROM 中的数据或指令码读出,并和诊断机中 ROM 相应单元进行比较,一旦不相同,可根据地址指示出故障 ROM 芯片的位置。当机型不一致时,从诊断盘中调出该故障机型 12K ROM 二进制文件到 RAM 后,便可依上述原理进行检测了。

检测 RAM 时,对故障机的随机存储器逐单元写一些数,如 \$00、\$FF 等,然后读出与原数进行比较,如果读出的数和原写进的不一致,通过判断可确定故障 RAM 的哪一位哪一片。也可用一些公认的 RAM 检测方法进行检测,如下雨法等。

板上 I/O 电路的测试是根据各部分电路的逻辑原理进行的。如扬声器电路的测试,通过对端口地址 \$C030 的交替访问,然后提示维修人员用逻辑笔或示波器测试 (APPLE— II) F12 的第 15 脚, F13 的第 12 脚, J13 的第 5 脚以及 Q4 的 C 脚,再将测得的电平信号或脉冲信号回答给诊断系统,正常时这几脚都有稳定脉冲信号,若不正常则依据逻辑关系可判断 F12、F13、J13 和 Q4 中某件或某几件坏了。

扩展接口电路一般不需仿真卡的帮助可直接插入测试机的扩展槽中进行测试与诊断。由于仿真接口卡能提供静态的或动态的地址与数据,这对于逻辑电路的测试有极大的帮助,本系统仍提供了可直接进行检测的软件。

### 三、程序设计方法及示例

诊断程序归结为对仿真卡输入输出的控制,根据不同部分的诊断要求,可以分为五类控制。

1. 观察静态的逻辑状态。观察地址总线、数据总线和一些译码电路等的好坏,可以分别将有关的地址数据写到 U1、U2 与 U3 中,和数据输出有关时,需将读写控制 R/W 置为低。

如将地址,数据置全“0”,BASIC 编程如下。(这里假设仿真卡插在 4 号槽口,下同):

```
POKE 49345,0 ;高八位地址置“0”
POKE 49344,0 ;低八位地址置“0”
POKE 49346,0 ;数据输出寄存器置“0”
DW=PEEK(49348) ;R/W 置低电平,允许数据输出。
```

2. 选通某电路或某一逻辑开关。例如置显示模式,只需将地址送往地址锁存器,以下命令置低分辨率图形显示第一页:

```
POKE 49345,192 ;置高八位地址 $C0
POKE 49344,80 ;置低八位地址 $50,图形方式
POKE 49344,85 ;置第二页模式低八位地址 $55
POKE 49344,86 ;置低分辨率方式
```

3. 读取 ROM 中的指令和数据。先将地址锁存,然后关闭数据输入锁存允许,读输入数据锁存器,允许锁存,如读 \$FFFF 单元内容:

```
POKE 49344,255
POKE 49345,255 ;置 $FFFF 到地址锁存器
RW=PEEK(49349) ;暂停数据输入锁存
DB=PEEK(49347) ;读数据输入端口到 DB
RW=PEEK(49349) ;允许锁存
```

4. 读写 RAM 存储器单元。如对 \$400 单元写 166,然后读回。

```
POKE 49345,4 ;置页地址
POKE 49344,0 ;页内 8 位地址
RW=PEEK(49348) ;R/W 置低电平
POKE 49346,166 ;写数到数据输出寄存器
RW=PEEK(49348) ;R/W 置高电平,恢复读状态
RW=PEEK(49349) ;暂停锁存
DB=PEEK(49347) ;读回
RW=PEEK(49349) ;允许锁存
```

5. 动态地给出地址,以便观察某些逻辑电路是否产生脉冲或稳定的波形。如让喇叭发声,选通地址后,在  $\Phi 1$  作用下, $F12-15$  等将有方波输出。

```
5000 POKE 49345,192
5010 POKE 49344,48 ;选通喇叭端口 $C030
5020 FOR I=1 TO 10:NEXT I ;延时可根据需要取舍
5030 POKE 49344,0 ;关闭
5040 GOTO 5010 ;循环
```

系统用 FPBASIC 和机器指令写成,运行环境为中华学习机带一个磁盘驱动器或 APPLE-II 等兼容机。可工作在自动诊断方式和菜单选择方式。主控程序先通过人机对话询问仿真接口卡插在那一扩展槽中,测试机和故障机分别为何种机型,然后询问是采用自动顺序检测还是选择对某一部分电路进行检测。如自动方式将按总线、ROM、RAM、板上 I/O 等顺序进行;若只检修与诊断某部分电路,可直接进入该部分相应的诊断模块执行。对于某些故障电路的诊断,为了加快诊断速度,提高诊断准确性,系统将询问故障现象。如显示电路部分,可将显示有无,高低分辨率绘图效果等告诉系统。

由于系统较大,这里以 APPLE II RFI 板为例给出检测 ROM、48KRAM、ROM 地址(包括 I/O 空间)译码电路测试程序:

主要变量及参数说明:

J1 \$, J2 \$ 分别代表测试机与故障机的机型,例如“RFI”、“CEC-1”等,由主程序通过询问检修人员后确定。

D1, D0 为仿真卡的两个地址锁存器端口地址, D1 的值为 \$C0X1, D0 为 \$C0X0, X=8+N, N 为仿真卡所插槽号, N 的值具体通过主程序对话获得,送往故障机的地址高位存(D1),低位存(D0)。

D2 为数据输出寄存器端口地址, D2 为 \$C0X2。

D3 为数据总线输入锁存器端口地址, D3 的值为 \$C0X3。

D4 为读写 R/W 信号控制端口地址, D4 的值为 \$C0X4。D5 为数据输入锁存器锁存允许端口, D5 的值为 \$C0X5。

A1, A0 为测试 ROM 时的起始地址, A1 装高位, A0 低位, 起始值分别为 208 和 0, 即十六进制的 \$D000, 每测一个单元, 低位地址加 1, 若大于 255 则 A0 清 0, A1 加 1, 直到 \$FFFF (ROM 最高地址)。

程序 1 检测诊断 ROM

```
1000 INPUT“ROM 为 2716 型(1)或 2732 型(2)”;M ;输入 1 或 2
1003 IF J1 $ = J2 $ THEN R0 = 53248 :GOTO 1020 ;机型一致,首地址为 $D000
1005 PRINT CHR $(4);“BLOAD”;J2 $;“,A $ 6000” ;不一致,调监控文件到 RAM
1010 R0 = 24576 ;置首地址为 $6000
1020 A1 = 208;A0 = 0 ;检测首地址 $D000
1025 FOR I=1 TO 6/M ;根据类型;确定循环次数
1030 B=0;FOR J=1 TO 2048*M
1035 POKE D1,A1;POKE D0,A0
1037 RW=PEEK(D5)
1040 IF POKE(D3)<>PEEK(R0) THEN B=R0
1043 RW=PEEK(D5)
1045 A0=A0+1;IF A0>255 THEN A1=A1+1;A0=0
1047 R0=R0+1
1050 NEXT J
1055 IF B=0 THEN 1070 ;无故障转测下一片
1065 PRINT“从右往左第”;I;“片 ROM 坏! 最后一个坏单元为:”;B
1070 NEXT I
1080 INPUT“请记载后按回车返回”;A $
1090 RETURN
```

程序 2 检测诊断 RAM

```
2000 PRINT CHR $(4);“BLOAD RAMT.COM,A $ 6002”;读机器子程序到内存
2003 POKE 24576,N*16 ;置槽号*16 到 $6000
2007 FOR R=0 TO 2 ;每 16K 一组
2008 POKE 24577,0 ;检测结果单元置“0”
2010 FOR I=R*64 TO (R+1)*64-1 ;16K 内页地址
2015 POKE D1,I ;置页地址到(D1)
2020 FOR J=0 TO 255
2025 POKE D0,J ;页内地址到(D0)
2030 CALL 24578 ;调子程序检测
```

```

2032 NEXT J;NEXT I
2035 W=PEEK(24577) ;读回 16K 检测结果
2040 IF W=0 THEN 2065 ;无故障转测下 16K
2045 FOR K=7 TO 0 STEP-1 ;循环判断
2050 IF W<2^K THEN 2060 ;第 K 位无错转测下一位
2055 PRINT"RAM 芯片";CHR$(67+R);K+3;"已坏!"
2057 W=W-2^K ;当前位置"0"
2060 NEXT K
2065 NEXT R ;循环
2070 INPUT "请记录后按回车返回!";A$
2080 RETURN

```

检测 RAM 机器指令子程序 RAMT.COM

```

6002- AE 00 60 LDX $6000 ;取槽号偏移量到 X。
6005- A9 FF LDA # $FF
6007- 9D 82 C0 STA $C082,X ;写全"1"到当前单元
600A- BD 84 C0 LDA $C084,X ;置 R/W 为低
600D- BD 84 C0 LDA $C084,X ;置 R/W 为高
6010- BD 85 C0 LDA $C085,X ;暂停锁存
6013- BD 83 C0 LDA $C083,X ;读回
6016- 49 FF EOR # $FF ;异或错误位变 1

```

```

6018- 0D 01 60 ORA $6001 ;保存到 $6001 单元
601B- 8D 01 60 STA $6001
601E- BD 85 C0 LDA $C085,X ;允许锁存
6021- A9 00 LDA # $00
6023- 9D 82 C0 STA $C082,X
6026- BD 84 C0 LDA $C084,X ;写"0"
6029- BD 84 C0 LDA $C084,X
602C- BD 85 C0 LDA $C085,X
602F- BD 83 C0 LDA $C083,X ;读回
6032- 0D 01 60 ORA $6001 ;保存
6035- 8D 01 60 STA $6001
6038- BD 85 C0 LDA $C085,X
603B- 60 RTS ;返回

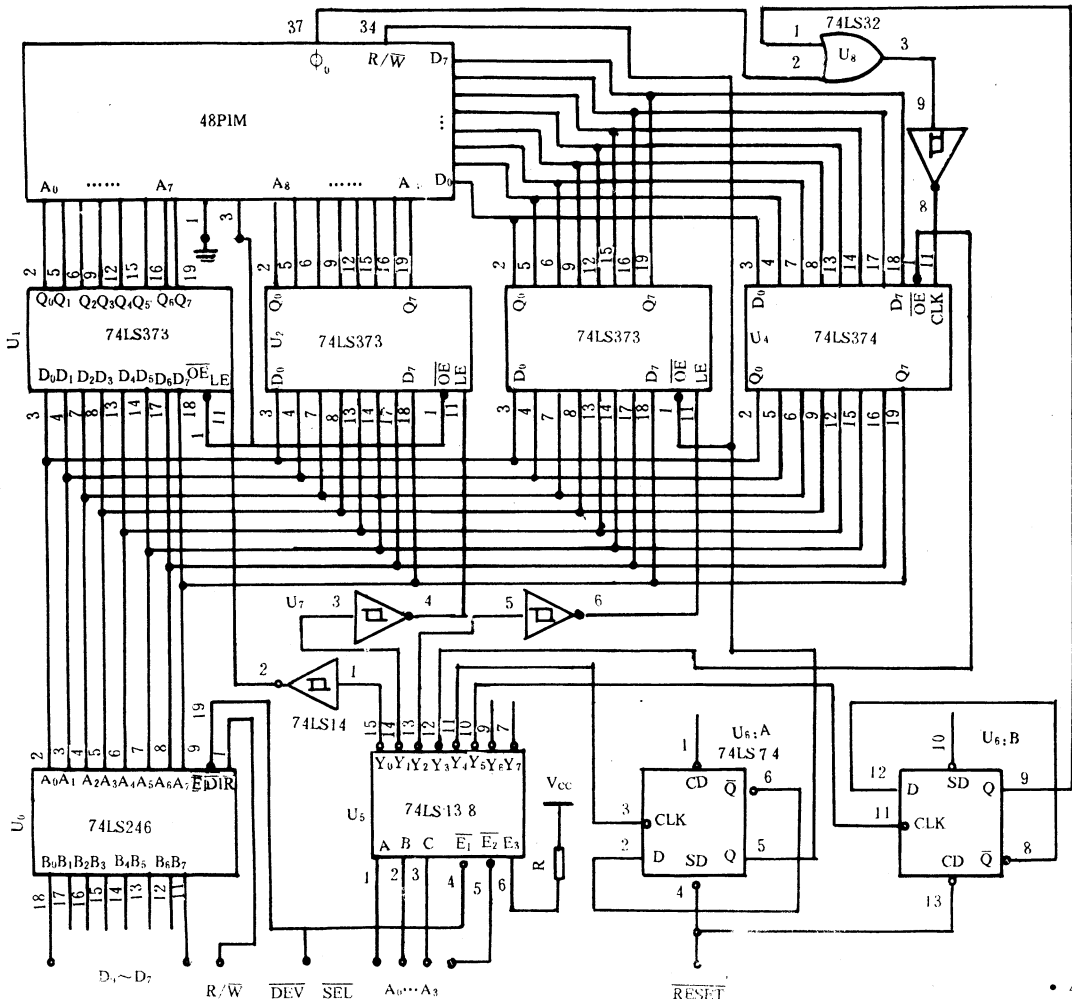
```

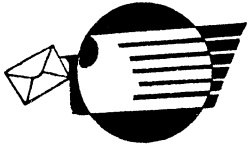
程序 3 检测 ROM 地址译码电路

```

500 INPUT"ROM 为 2716 型(1)或 2732 型(2)";M
505 POKE D1,192
510 INPUT "请测 F12-4 脚,脉冲 P 否则 L";B$
515 IF B$="P" THEN 520
517 T$="01" ;设测时钟标志
(下转 37 页)

```





# 普及型 PC 个人用户软件交流联谊 活动问题解答(十)

## 读者联谊

北京中国农科院计算中心(100081) 王路敬

### 39. 微型计算机病毒一般寄生在磁盘的哪些地方?

微型机系统在目前来说永久性存储介质是硬盘或软盘,微型机病毒是一种可直接或间接执行的文件,是依附于系统特点的没有文件名的秘密程序,它必须以现有的硬件资源而存在。从已经发现的微机病毒来看寄生在磁盘的如下几个区域:

(1)寄生在磁盘引导扇区中。

任何操作系统都有自举过程,例如 PC-DOS,首先由系统读入引导程序并执行它,将 DOS 读入内存。病毒程序就是利用了这一点,自身占据了引导扇区而将原来的引导扇区内容及其病毒程序的其他部分放在磁盘的其他空间,并给这些扇区标为坏簇。

(2)寄生在文件分配表中

作为软盘或者硬盘的 PC-DOS 分区都有一个文件分配表区,不同容量的硬盘或软盘 FAT 的大小有所不同。如表 1 所示:

表 1 各种 PC-DOS 版本各类磁盘 FAT 表

磁盘类型	FAT 表个数	每个 FAT 表占用扇区数	首扇区号	DOS 版本
160KB 软盘	2	1	1,3	V1.0
320KB 软盘	2	1	1,3	V1.1
180KB 软盘		2	1,3	V1.1
360KB 软盘	2	2	1,3	V2.0
1.2MB 硬盘	2	7	1,8	V3.0
10MB 硬盘	2	8	1,9	V2.0
增强 20MB 硬盘	2	8	1,9	V2.0
AT20MB 硬盘	2	8	1,9	V2.0
AT20MB 硬盘	2	41	1,42	V3.0
AT32MB 硬盘	2	64	1,65	V3.0

(3)寄生在硬盘的主引导扇区。

例如:Stone 病毒感染硬盘的主引导扇区,该区与 DOS 无关。

(4)寄生在磁盘内的可执行文件中。

有一类病毒寄生在可执行文件中,例如:.COM 文件,.EXE 文件。有些病毒寄生在文件头部,有些病毒寄生在文件的尾部,也有些病毒寄生在文件的中间,而成为文件的一部分,占据一定的磁盘空间。

### 40. 微机病毒对磁盘造成哪些危害?

从目前发现的微机病毒来看对磁盘造成的危害主要表现在:

(1)破坏软盘的引导区和硬盘的主引导扇区,使得磁盘操作系统不能正常启动或不能启动。

(2)破坏软盘或硬盘上文件分配表 FAT,使用户在磁盘上的信息丢失。

(3)删除软盘或硬盘上的可执行文件或数据文件。如果删除的文件是系统文件则会导致这片盘不能引导系统。

(4)改变或破坏文件中的数据;改变磁盘分配造成数据写入错误。

(5)在磁盘上产生坏的扇区,使磁盘可用的空间减小。

(6)改变或重写磁盘卷标,对整个磁盘或磁盘的特定磁道或扇区进行格式化。

软盘或硬盘,尤其是软盘,既是病毒寄生的介质,同时又是病毒侵害的重要对象,也是病毒传播的重要途径。

### 41. 为什么硬盘上的 Stoned 病毒 FORMAT 命令不能消除? 如何解决?

由于硬盘的大麻病毒寄生在硬盘的主引导扇区,系统传递或硬盘进行普通格式化都不能改变主引导记录,因为该区与 DOS 无关,故出现经 FORMAT 格式化的硬盘仍有 Stoned 病毒。

寄生在硬盘主引导扇区的病毒消除方法一是利用公安部颁发的消病毒软件,这是自动的方法。下面介绍另外三种方法:

方法一:

通过一个应用程序把被病毒移走的硬盘主引导扇区的内容复制到被病毒占用的第 0 头 0 柱 1 扇区完成。应用程序如下:

```

SSEG SEGMENT PARA' CODE'
 ASSUME CS,CSEG,DS,CSEG
 ASSUME ES,CSEG,SS,STACK
HDUS PROC FAR
 PUSH DS
 XOR AX,AX
 PUSH AX

 MOV AH,0
 MOV DL,80H
 INT 13

 MOV AX,0201H
 MOV DX,0080H
 MOV CX,0001H
 MOV BX,SEG STONE

```

```

MOV ES,BX
MOV BX,OFFS ET STONE
INT 13H
MOV AX,0301H
MOV DX,0080H
MOV CX,0001H
MOV BX,SEG STONE
MOV ES,BX
MOV BX,OFFSET STONE
INT 13H
RET
HDUS ENDP
STONE DB 512 DUP(0)
CSEG ENDS

```

```

STACK SEGMENT PARA STACK 'STACK'
DB 256 DUP(0)
STACK ENDS
END

```

#### 方法二：

对硬盘首先进行初始化操作，即进行低级格式化。如果系统带有低级格式化程序可在操作系统下直接执行，若没有低级格式化的文件，IBM PC/XT 及其兼容机可用 DEBUG 程序调用 ROM BIOS 中的低级格式化程序并执行之。

操作如下：

```

A>DEBUG
-G=C800:0005

```

该操作完成后，对硬盘进行 DOS 分区再执行 FORMAT C: /S 命令，即可消除硬盘上的大麻病毒。其他 CPU 为 286、386 微机运行相应的诊断程序中硬盘低级格式化程序即可。

#### 方法三：

如果硬盘中储存了有些没有备份的文件，重建硬盘主引导记录不能采取物理格式化硬盘以及重做硬盘分区操作，因为这样的操作将会造成硬盘的数据资料全部丢失。如果有另一台正常的相同类型的计算机，可以采用一种简单而又可靠的办法重建硬盘主引导记录。实现的方法：

从正常计算机硬盘中提取它的引导记录扇区的内容，然后把它写入已失效的硬盘中，用它来复盖掉已被破坏的硬盘主引导扇区。值得注意的是，用来提取主引导记录的硬盘一定要与失效的硬盘具有相同规格，相同系统和相同的分区格式，否则不可能达到要求。具体操作如下：

(1)在 DEBUG 状态下把正常系统硬盘的主引导扇区的内容以文件方式写在 A 软盘上。

```

-A 100
××××:0100 MOV AX,0201
××××:0103 MOV BX,0200
××××:0106 MOV CX,0001
××××:0109 MOV DX,0080

```

```

××××:010C INT 13
××××:010E INT 3
-G=100
-N A;HDBOOT
-R CX
××××
:400
-W
-Q

```

(2)把该软盘插入已感染大麻病毒系统的 A 驱动器中，然后在 DEBUG 状态下重写硬盘主引导扇区。

```

-N A;HDBOOT
-L
-A 100
××××:0100 MOV AX,0301
××××:0103 MOV BX,0200
××××:0106 MOV CX,0001
××××:0109 MOV DX,0080
××××:010C INT 13
××××:010E INT 3
-G=100
-Q

```

#### 42. 不同 PC-DOS 版本下硬盘分区表有什么差别？

不同的 DOS 版本对硬盘分区表的编排方式上有所不同，当整个硬盘空间只含一个 DOS 分区的情况下，DOS 2.0 或 2.10 等是把这一分区信息存放在偏移 1EEH 起始的位置上（即第 4 分区表位置），而 DOS 3.30 则把分区信息存放在偏移 1BEH 位置上（即第 1 分区表位置）。DOS3.30 可以从第 1 或者第 4 分区表位置上识别硬盘唯一的一个分区，而 DOS 2.0 或 2.10 等版本却不可以。因此，如果硬盘是 DOS 3.30 系统，从 A 盘用 DOS 2.0 或 2.10 作引导之后，会出现系统不认识硬盘的问题，试图对硬盘进行操作会显示：“Invalid drive specification”错误信息。这时并非硬盘有什么故障，而是 DOS 版本的差异所引起的。当然，如果是硬盘控制器板或者系统参数没有正确设置，也会出现同样的错误信息。除此之外，DOS 2.0 或 2.10 等版本与 PC-DOS 3.0、3.10、3.20、3.30 等版本在文件分配表 FAT 上也有一个重要的差别。这就是 DOS 2.0 或 2.10 等以下版本，FAT 表中的每一项固定为 12 位（二进制）。而 DOS 3.0 以上版本 FAT 表的每一项可以是 12 位也可以是 16 位，至于是 12 位还是 16 位是由 FDISK 程序在建立 DOS 分区时根据用户指定分区的大小来决定的，如果 12 位数能表示出 DOS 分区中所有的簇数就使用 12 位，否则使用 16 位。DOS3.0 以上可以产生和识别 12 位或 16 位两种形式的 FAT 表，而 DOS2.0 或 2.10 等版本只可以产生和识别 12 位的 FAT 表。这就是从 A 驱动器用 DOS2.0 或 2.10 作引导不能对 16 位 FAT 的 DOS3.0 以上系统硬盘进行操作的原因。

# SJW-B 系列自动补偿式 三相大功率电力稳压器

## 原理 · 特点 · 用途



上海精达电子仪器厂  
技术科

电压不稳不仅给工业生产、科学研究和日常生活增添了不少麻烦,有时甚至还会影响生产,损坏设备,造成事故。随着现代科技的迅猛发展,工矿、科研、邮电、医院、宾馆等部门,对电源电压稳定性的要求越来越高。尽管稳压电源种类很多,传统的有电子交流式、感应式与磁饱和式稳压器等,但这些稳压器容量较小,损耗较大,波形失真严重,且对负载性质适应性又差,均不能满足生产和科研日益增长的需求。

SJW-B 系列自动补偿式三相大功率电力稳压器是我厂引进、消化、吸收国外的先进技术,结合国情,精心研制、专业生产的节能型新产品。其性能之优、功率之大、价格之廉是传统的稳压器无法比拟的。我厂生产的 SJW-B 系列,其性能与技术指标可与国外同类产品媲美,而价格仅是国外同类产品的 1/6~1/7,深受广大用户的赞誉。

**1. 原理** 三相自动补偿式电力稳压器的工作原理如附图所示。它由调压器 T1、补偿变压器 T2、伺服电动机 MS 与电压采样比较控制装置等组成。现以 A 相为例,说明其稳压原理。从附图中可知:

$$U_{A0} = U_{A1} - \Delta U_A$$

式中:  $U_{A0}$ —A 相的输出电压;

$U_{A1}$ —A 相的输入电压;

$\Delta U_A$ —A 相的补偿电压。

当输入电压  $U_{A1}$  或负载变化引起输出电压  $U_{A0}$  变化时,电压采样比较控制装置从稳压器输出端采样,采样电压经整流滤波后与基准电压上限值和下限值比较。当采样电压大于基准电压上限值时,电压采样比较控制装置控制继电器动作,使伺服电动机 MS 作一定方向旋转,带动调压器 T1 上的电刷组作相对滑动,这时在 A 相调压器上出现一个  $U_{A2}$ ,在补偿变压器 T2 的 A 相一次侧相应产生  $U_{A3}$ ,则在二次侧产生补偿电压  $\Delta U_A$ ,它使输出电压  $U_{A0}$  下降,直至回复到稳压精度的允许范围内。此时,继电器释放,伺服电动机 MS 停止转动。反之,伺服电动机 MS 向反方向旋转,补偿电压  $\Delta U_A$  改变极性,使输出电压  $U_{A0}$  上升,直至回复到稳压精度的允许范围内,以达到输出电压稳定的目的。

### 2. 特点

(1) **稳压范围很宽** 输入电压容许在  $380V \pm 20\%$  范围内变化,即输入电压在 304~456V 范围内,均

能正常工作。

(2) **稳压精度很高** 稳压精度为  $\pm 1\%$  (输出电压为  $380V \pm 1\%$ ),且稳压精度在  $\pm (1 \sim 5)\%$  可调。

(3) **损耗很低** 电能转换效率  $> 98.5\%$ ,损耗很低,节能明显。

(4) **负载性质适应性很广** 能适应任何阻性、感性和容性负载。

(5) **输出功率很大** 额定输出功率有 20~1000kVA 等 12 种规格,其容量之大是传统稳压器望尘莫及的。

(6) **波形失真度很小** 输出电压的波形与输入的电网正弦波形几乎一致,其波形畸变  $< 0.1\%$ 。

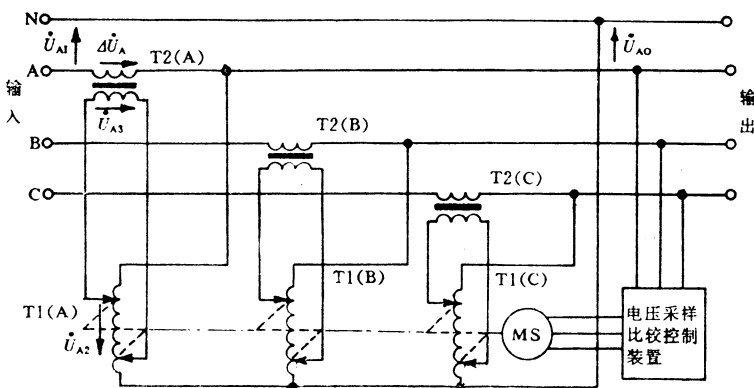
(7) **应变时间很短** 调压应变  $> 20V/s$ ; 过压保护应变  $< 0.5s$ 。

此外, SJW-B 系列电力稳压器还具备长期连续工作、能承受瞬时超负载、有可靠的过压保护和故障自停等优点。

**3. 应用** SJW-B 系列自动补偿式大容量三相电力稳压器能适用一切需要电压稳定的场合,尤其适用于电压波动大、负载变动大的用电场所。它是任何用电部门和引进设备稳压电源的理想产品,可广泛应用于工业、交通、邮电、军事、铁路、医院、宾馆和科研等领域。例如:它可作为工厂精密机床、仪器设备、自动联动生产线,大型医疗设备,广播和电视台等优质稳压电源。

本厂产品获国家电力电子产品合格证书。

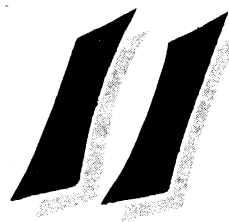
上海精达电子仪器厂  
地址 上海市新闻路 579 号  
电话 2563294  
2170089  
电挂 6910  
邮编 200041







ISSN 1000-1017  
**電子**  
**與**  
**電腦**



一九九二年

总期第92期



# 電子與電腦

## 目 录

### · 综述 ·

图象处理的一大革命——压缩 ..... (2)

### · PC 用户 ·

PC 机软件加解密技术剖析 ..... 李文亮(3)

字符处理函数在 FORTRAN 中的实现  
..... 任铁良 丁玲玲(5)

CPAV 防病毒软件使用简介 ..... 吴桦(7)

带检索功能的通讯录录入程序 ..... 张春明(10)

PC 机中系统时间的自动显示 ..... 郑嘉琦(12)

最近出现的几种新病毒 ..... 苏民生(13)

电子扭计板 ..... 葛建华 吴立国(14)

第四届国际信息学奥林匹克竞赛试题 ..... (15)

### · 学习机之友 ·

微机模拟光标卡尺读数训练程序 ..... 王太文(17)

磁带虚拟磁盘的文件管理 ..... 胡发新(18)

APPLE-Ⅱ 音乐功能扩充 ..... 陈建明(19)

POSITION 命令新用 ..... 周进(20)

CEC-与 1724 打印机配接图形硬拷贝程序  
..... 张益贵(20)

游戏接口 PDL 的扩展 ..... 张建群(21)

SUPER DOS 简介 ..... 张志(21)

### · 语言讲座 ·

第十讲 监控子程序的调用(下) ..... 朱国江(23)

### · 学用单片机 ·

BJS-51 单片机实验系统(续) ..... 张俊谟(26)

### · 学装微电脑 ·

微电脑控制微型钻床 ..... 易齐干(29)

### · 电脑巧开发 ·

数字集成电路简易测试器 ..... 王正英(35)

中华学习机调制伴音电路 ..... 李永和(39)

### · 电脑游戏机 ·

第四讲 游戏程序的设计过程(上) ..... 于春(40)

### · 维修经验谈 ·

笔记本型电脑技术和市场 ..... 赵广恩(43)

用软件对软驱进行简单读写检测  
..... 李晓中 麻佳洛 谢静 张景生(44)

### · 读者联谊 ·

普及型 PC 个人用户软件交流联谊活动问题解答(十一)

..... 王路敬(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:杨逢仪

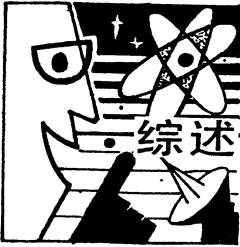
订购处:全国各地邮电局

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



# 图象处理的一大革命——压缩

中国电子器件深圳公司供稿

本期封面有两幅图象,其中原图信息量为 3.2MB,经压缩处理后为 155KB。人眼看不出二幅图象的色调有什么不同。不难想象,所谓压缩,就是将原图象信息量删去相当大的部分,而仍不改变原图本色的信息处理技术。这样一来,就可以利用压缩/还原技术,在 CD 唱盘上再存入 VHS 影象,一份消费,两份享受。将声音/图象压缩后存入 CD-ROM 中,只要在电脑里加上解压缩卡,就可以用电脑唱“卡拉 OK”了,为计算机的多媒体化提供方便。占 6MHZ 的电视频道,占 27MHZ 的卫星频道均可分割成四个频道,大大的增加电视的频道,为人们增添更多、更丰富的艺术享受。

美国 C-Cube Microsystems 公司首先推出 C-Cube CL550 图象压缩/还原处理器芯片及其开发系统,为用户提供了一个了解和应用压缩技术的环境。

本文根据 C-Cube 公司提供的资料,以问答形式对“压缩”作一简要的介绍。

问:何谓 C-Cube CL550 处理器?

答:C-Cube CL550 是一种采用 JPEG 标准的单片图象压缩/还原处理器。

问:图象压缩解决了图象处理的什么问题?

答:图象压缩技术解决了图象处理中的二个瓶颈问题:存储器花费大和总线速度的极限。如不采用压缩技术,要存储 10 秒钟的高分辨(720×480)的视频图象(30 帧/秒)就需要 311MB 的存储量,总线速度为 31MB/秒。而采用 JPEG 压缩标准处理之后,它被压到 13MB 的存储容量,总线速度降低为 1.3MB/秒。

问:图象压缩是一种新技术吗?

答:否,该技术在 high-end 系统中已经应用多年了。然而,作为图象压缩/还原芯片的出现,确是最近的事情。

问:什么是 JPEG 标准?

答:JPEG (Joint Photographic Experts Group) 是国际 CCITT 和 ISO 共同制订的连续色调压缩标准。C-Cube 公司是 JPEG 委员会的成员。

问:JPEG 的重要性何在?

答:JPEG 是一种国际开放系统标准,它使不同制造商所设计的系统均可互联通信和交换信息。JPEG 规则是在高压缩比下获得卓越的图象品质的保证。

问:在高压缩比下高品质图象意味着什么?

答:当用 JPEG 规则将一幅屏幕图象,以 10:1 进行压缩;一幅打印图象以 25:1 进行压缩时,凭人眼是观察不出它们有什么不同,对于人眼来说,似乎全部信息仍被完整保存。其实,压缩比 25:1 就意味着该图象有

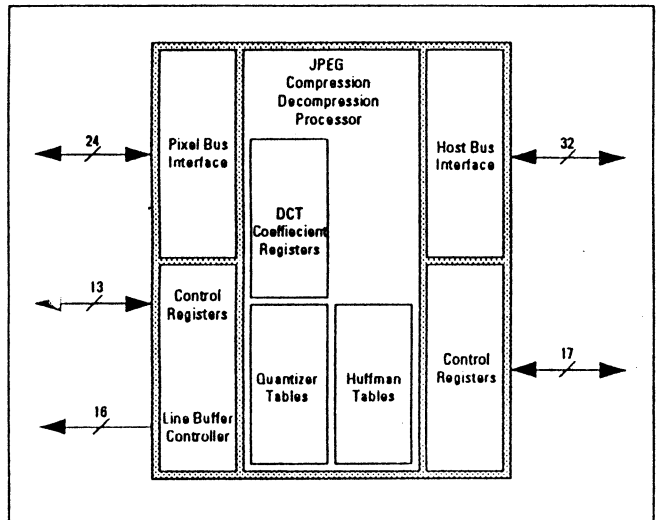
96% 的数字信息被删去了。

问:如何选择压缩比?

答:C-Cube CL550 处理器的压缩比是用编程的方法选择的。其压缩比最高为 100:1。

问:C-Cube CL550 处理器的结构如何?

答:CL550 的功能框图如下:



C-Cube CL550 Processor Functional Block Diagram

144-PIN 标准陶瓷座阵列封装的有:C-Cube CL550-35 和 CL550-30。

144-PIN 扁平封装的有 CL550-35 和 CL550-10。

采用 CMOS 工艺,在 35MHZ 时最小耗散功率为 3.5W。

问:C-Cube CL550 可处理的彩色范围为何?

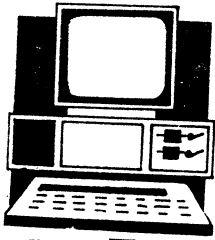
答:C-Cube CL550 处理器可处理 8 位灰度 24 位 RGB,CMYK 或 4:4:4:4 和 YUV(4:4:4 和 2:2:2)彩色空间。

问:C-Cube CL550 有几种处理速度?

答:C-Cube CL550 有三种处理速度:10MHZ、30MHZ 和 35MHZ。C-Cube CL550-10 (10MHZ)用于对静止图象的处理。C-Cube CL550-30 和 C-Cube CL550-35 能实时处理(压缩/还原)NTSC,PAL 和 CCIR601 视频图象。

多个 C-Cube CL550 处理器可对图象进

(下转第 11 页)



# PC 软件加解密技术剖析

华南理工大学计算机系软件专业 88 级 李文亮

## PC 用户

在计算机系统的开发过程中,软件费用所占的比例越来越大,软件的地位也越来越重要。一方面软件和数据非常有价值,另一方面任何人都很容易得到原版的复制品,因此,软件的非法拷贝现象也越演越烈。

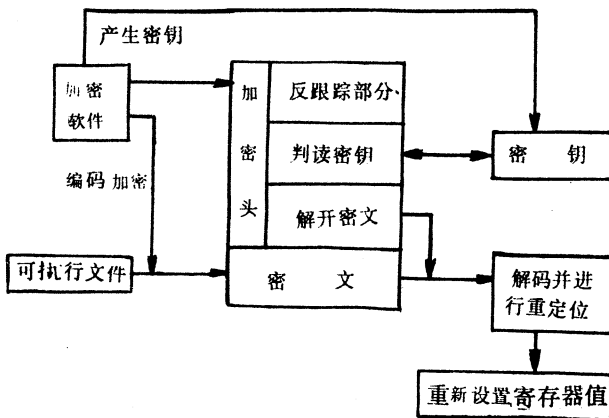
为了保护软件的潜在市场,人们开始研究加密技术,同时解密技术也伴随而来。加密和解密经过“魔高一尺,道高一丈”的矛盾斗争,从低级向高级发展。

PC 机加密技术最早在一些游戏上使用,后来因为其简单易行而迅速得到推广。并出现了象 PROLOK、SOFTGUARD 和 LOCK89 等一些专门的加密工具软件。我国近几年也对加解密技术做了深入的研究,并得到广泛的应用,生产了诸如 LOCK89、COPY02 等许多加解密工具。

### 一、加密技术基础

所谓的 PC 软件加密,就是使用特殊技术在磁盘上产生特殊的信息块,普通 PC 软盘驱动器可以把这些信息读出来,但是不能够产生同样的信息块。这样,当用户试图去复制该软盘时,那个特殊信息就会丢失。而我们可以把程序嵌入到软件中,使之去读取这个特殊信息,判断正确才能去执行。

一个典型的软件加密系统是由两部分组成:第一部分是在每个原版程序盘上都有的特殊信息块(即密钥);第二部分是密钥判读程序,一般嵌入在应用程序之中,形成加密文件。当启动执行加密软件时,该段程序将首先被执行。它先去判读密钥,正确的话才去执行应用程序。加密的原理图如下:



图一 软件加密原理图

由图可以看到,加密文件分为加密头和原执行文件的密文两部份。其优点在于加密软件的通用性强,而且无需修改原程序。但是缺点也是明显的,即破译者往往可以通过把加密头去掉而得到原可执行文件。

我们可以把程序体和加密头揉为一体,实行一体化的加密。这样,破译者需要对整个系统进行彻底剖析,才有可能解密。而对于一个比较庞大的系统,这基本上是不可能的。因此采用这种方法可以大大地提高保密性能。超想自然码就宣称使用了这种方法。但这种方法要求软件开发者和加密者必须是同一个主体,有很大的局限性。而且往往由于程序的嵌入,破坏了软件整体的严谨性。

由图一可以看到,密钥是整个加密系统的关键,如果密钥能随意被拷贝,那么整个系统就完全失去其意义了。至于密文的产生方法,并无特殊要求,只要编码和解码能正确进行即可。

在讨论加密技术之前,必须对软盘的物理结构;磁盘格式;磁盘参数表有所了解,由于篇幅所限,这里不作介绍,请参阅有关资料。

在了解了 PC 磁盘的格式,中断 13H 的调用方法和磁盘参数表后,就可以利用以上知识来产生密钥了。

最简单的就是通过修改软盘参数表,把软盘格式化为特殊格式来作为密钥。正常情况下 PC 机是无法读写这些信息的,当你试图用 DOS 来拷贝被保护的磁盘时,除了密钥之外的其它信息都被正确拷贝,运行程序时就会找不到密钥信息;而加密程序则可以通过修改软盘参数表来完成读取和判断。从硬件的角度讲,读和写是完全不同的,它们是两套分开的电路。同样道理,PC 机的磁盘驱动器可以识别原盘上的不确定格式,但不能在复制品上生成这一格式,防止拷贝的方法正是基于这一原理。然而,特殊格式的信息可以用特殊的软件拷贝出来,象 COPYWRIT 等一些高级拷贝工具,是以字节或位来拷贝的,并且具有一定的分析功能,往往可以成功地拷贝出一个新的“原盘”。

另一种方法就是利用特殊的设备,制造出一般驱动器根本无法生成的特殊格式。如在磁道上产生 256 个仅含有标志而不带数据的空扇区。这种方法能抵御大部份的攻击,但设备费用是比较昂贵的。LOTUS 1-2-3 就是在 0 道和 1 道的 08 扇区之后,制造了 10 个没有数据,只有 ID 部分的扇区。

后来经过研究,又产生了一种“指纹加密法”。

我们都知识,每个人的指纹都是其特有的,没有两个人的指纹是相同的。那么,软盘上是否存在“指纹”呢?正常情况下是不存在的,但可以人工生成。

一个磁道所能容纳的信息量是有限的,在双面倍密度的软盘上大概为 1850K 字节。正常情况下,所有的扇区都应容纳在一个磁道之内,但是如果扇区较大,数量较多的话,扇区就有可能从磁道尾越过磁道首尾

的接缝处,而跨回到磁道首。我们可以把磁道想象为一个环,当信息量大于环长时,就产生了一部分的重叠。此时,由于软盘驱动器的电磁性能指标(如电流、电压、磁场强度和磁盘转速等)的不稳定性,在磁道的头尾衔接处就会产生一些特殊的“指纹信息”。这些信息使用 COPYWRIT 等高级拷贝工具,拷贝卡或拷贝机,都是无法拷贝的,甚至对于同一部驱动器,每次产生的指纹也是不一样的。

作者根据这一原理,在 IBM 机上用汇编语言编写了两段制做和读取指纹的程序,用 TASM2.01 编译、链接和调试通过。具体过程和程序见附录。

另外,由于制作工艺上的差别,每个硬盘都有其自然“指纹”。即使是同一型号的硬盘,也有一定的差异,即使把一个硬盘 DISKCOPY 到另一个硬盘上去,两者也不会一模一样。这点是与软盘有很大区别的。加密系统可以在安装到硬盘上时搜索这些特殊信息,并用来作密钥。如果 2·13 汉字系统就是把标识放在硬盘的主引导扇上。

产生密钥的另一种方法是硬加密,其典型代表是激光加密。它利用激光在磁盘上打孔,制造出硬坏标志。一般有单孔和双孔两种。激光的方向性好,能量高,均匀,烧出来的孔很小,基本上不会破坏磁盘表面的光洁度,而且随机性很好,相同的概率为几十亿分之一。同时,由于激光孔是物理性标识,用软件方法是无法生成的,这就可以抵御各种拷贝工具的攻击,有极强的保密性。价格昂贵是其最大的局限性,同时,它也要求驱动器有较稳定的性能。国内也有用针直接在磁盘上扎孔的方法来模拟的,也有一定的效果,但会破坏磁盘表面的光洁度,损坏磁头。

还有一种方式就是使用加密卡。当加密程序运行的时候,需要寻找指定的芯片,并把一个数据送给它,经运算后返回一个密钥,判断正确后才继续运行。这种方式安全性好,但成本高,使用也不够方便。

最近,又提出了两种新型加密方法——掩模加密技术和电磁加密技术。前者采用半导体加工工艺中的镀膜方法,后者借助于专门的电磁机构,在软盘上生成一系列的密钥信息;前者与激光加密相似,但对磁盘无任何破坏;后者采用了与 PC 机软驱 MFM 或 FM 不同的编码格式,都有极强的反拷贝性,是很有发展前途的。

随着加密技术的发展,密钥往往已经不能被拷贝。因此破译者更多地把目光投向加密程序。现在更多是采用软件破译的方法,迫使加密者不得不考虑如何提高反跟踪性能。所谓的反跟踪,就是通过动态解码、破坏中断等方式,阻止破译者通过对加密头的跟踪分析,破解加密软件。这部分将在后面作详细介绍。

一般加密有如下的几种形式:

(1)使用口令密码,另一形式是用回答问题来替代。这种方法多为早期的游戏软件所采用,知道了密码便可随意拷贝,保护性最差。

(2)修改文件名、属性。通过修改文件属性为隐含,

在文件名中嵌入特殊字符等方式来加密。它无法阻止 DISKCOPY;在 PCTOOLS 等强劲的磁盘工具出现之后,更是不堪一击,仅在硬盘上还有一定的使用余地。

早期的王码五笔系统就是使用了这种方法。例如,用 PCTOOLS 把目录属性改为 17H,并把目录的最后一个字节改为 80H,那么,无论在 DOS 或 PCTOOLS 下都无法发现这个目录;但在 DOS 下可对此目录进行一切正常工作。

(3)判读密钥法。首先判断密钥是否存在和正确与否,才去执行程序,高级的还可利用密钥来对密文进行解密。这种方法保密性最强,应用也最广泛。

随着加密技术的发展,涌现了许多优秀的加密系统。美国加州 New BuryPark 市 VALUT 公司的 PRO-LOCK 就是激光加密的经典。我国的 CCDOS4.0 就是采用它来作保护工具的。Santa Clara 市 SOFT-GUARD 公司的 SOFTGUARD 是另一个著名的加密软件。我国的技术人员也开发了诸如 WL、SUPER-LOCK(虎符)和 LOCK89 等加密系统。LOCK89 采用了多层次硬加密、软加密和反动态跟踪技术,实现了多变和不可拷贝,是其中的佼佼者。

## 二、加解密技术分析

破解一个加密系统一般有下面两个途径:

(1)在目标盘上完成密钥的拷贝和制作,使目标盘和原盘达到一致。

COPYWRIT 就是使用了这种方法。严格地说,这还不是解密,因为拷贝后生成的新盘还是有保护的。它对付一些简单的加密系统还是比较有效的,但对指纹或激光加密等却是无能为力。

(2)通过对加密程序的跟踪分析,找到判读密钥部分,把它去掉或作一些修改,使之能脱离密钥而独立工作,实现破译。一般可分为静态分析——把程序反汇编打印出来进行分析和动态分析——利用调试工具进行动态跟踪分析两种方法,往往要把两者紧密结合起来,才能成功。

破译者必须熟悉整个 PC 机系统,特别是要熟练掌握 8088 汇编语言,中断调用和处理,DOS 功能调用以及一些特殊的端口操作命令。

下面介绍一些常见的解密工具。

这是一些市面上流行的拷贝工具: COPYIIPC、COPYWRIT、SUPERCOPY、LOCKSMITH、COPY-ANY 和 DISKEXPLORE 等。美国 Oregon 州 Portland 市的 Central Point 公司和加拿大安大略省多伦多市的 Quaid 公司专门出售破密软件,他们出售的软件也都不保护。他们的 COPYIIPC 和 COPYWRIT 就是拷贝者的两大皇牌。

一般拿到加密软件之后,可以先用手头上的拷贝工具试一试,不行的话才考虑破译。较新版的 COPYWRIT 甚至能成功拷贝早期的单孔激光加密盘,而 RCOPY02 对很多的加密系统都有效。

LOCKSMITH 具有读写磁盘扇区,分析、修改磁盘参数的功能,可以用来实现简单的加解密。

DEBUG 原是 DOS 的一个机器码调试程序,因为它有很强的动态跟踪功能,而被广泛应用于解密。SYMDEB 的功能与 DEBUG 相似,但它另外还有输入、输出改向的功能,可很方便地生成反汇编清单文件,非常适合于作解密分析工具。下面是 SYMDEB 的几个有关命令:

- A[ADDRESS]从地址 ADDRESS 开始输入汇编程序;
- BP[ADDRESS]在地址 ADDRESS 处设置断点;
- D[ADDRE1][ADDRE2]

显示从地址 ADDRE1 到 ADDRE2 间的数据信息,可以有 DB, DW, DD 等多种显示方式;

- T 单步跟踪;
- P 同 T,但遇到子程序或中断调用时一次执行完;
- U[ADDRE1][ADDRE2] 从地址 ADDRE1 开始反汇编到 ADDRE2;
- < IN-FILE 从文件 IN-FILE 读入命令;
- > OUT-FILE 把输出送到文件 OUT-FILE 中去;

默认的输入和输出文件都是 CON。

TURBO DEBUGGER 和 CODEVIEW 是高级的源代码调试程序,它们采用集成工作方式,所有的寄存器值和当前指令代码都显示在屏幕上,并用光条指示当前指令行,使用非常方便。可惜系统比较庞大,因此也较脆弱,很容易被反跟踪程序所破坏。

另外,市面上还有诸如 UNLOCK、UNGUARD 等一些解密软件,它们都是针对某种加密系统的,使用方便,效果也较好。江西科达通信电脑技术发展公司研制的 RCOPY02,实现了从内存中进行加密软件的拷贝。按照这一工作原理,各种工具所加密的程序,都可能被拷贝出一个工作副本。被认为是国内首屈一指的 LOCK89 加密盘的生产母盘,经 RCOPY02 处理后,也能产生可拷贝的工作副本、甚至工作副本还可以建立在硬盘上。另外,它还能对各种应用程序保留任一时刻的工作断点,有极强的现场保留功能。UNLOCK89 可以破解象意大利赛车等一些游戏软件,使之成为非加密版,解密能力也比较强。

由于软件保护的方法越来越多,难以使用单一的程序来战胜对手,软件拷贝公司用硬件巧妙地解决了这一问题。Central Point 和 Disk-Tech 公司通过增加 PC 机的拷贝卡,对磁盘实现了真正的磁介质拷贝。Central Point 公司的拷贝板与磁盘控制器和驱动器都有联线,拷贝卡直接控制驱动器的读写,它可以拷贝任意信息。正如 Central Point 公司总裁米开尔·布朗解释:“它只不过是把数据从一个磁盘拾起来,放在另一个磁盘上,并说来下一个。”

(未完,待续)

## 字符处理函数在 FORTRAN 中的实现

冶金工业部鞍山热能研究院 (114004)任铁良 丁玲玲

FORTRAN 有诸多的数据处理函数,但没有提供诸如 BASIC 和 dBASE 中那样丰富多采的字符处理函数,这使得它在字符处理方面比 BASIC 和 dBASE 逊色。特别是在用 FORTRAN 编程同时实现复杂的数值计算和繁多的字符处理时,字符处理问题便显得尤为突出。

FORTRAN77 V3.3 中提供的与字符处理有关的函数为:ASC 码转换成其对应字符的字符型函数 CHAR,字符串的第一个字符转换成对应 ASC 码的整型函数 ICHAR,及判断两字符串顺序的逻辑函数 LGE, LGT, LLE, LLT; FORTRAN 77 更高的版本还提供了求字符串长度的整型函数 LEN、求子字符串位置的整型函数 INDEX 和两字符串连接运算符“//”。

笔者根据 FORTRAN 77 V3.3 中的基本字符处理函数 CHAR 和 ICHAR,及取子字符串和为子字符串赋值操作,编制字符串处理外部函数程序,并且经编译生成目标程序,再把目标程序追加到 FORTRAN.LIB 文件中;当在 FORTRAN 中调用这些函数时,只要引用对应的函数名即可,这样就可 FORTRAN 中实现与 BASIC 和 dBASE 字符处理功能相同的一些函数,以弥补 FORTRAN 语言中字符处理功能的不足。

附后的 FORTRAN 程序实现了下列字符处理函数:

### (1)LEN 函数

函数功能:求出字符串的长度,字符串尾部空格不计。

### (2)INDEX 函数

函数功能:求出第二个字符串在第一个字符串中的起始位置(字符串尾部空格不计),若第一个字符串不含有第二个字符串,则函数值为 0

### (3)PLUS 函数

函数功能:连接两字符串(字符串尾部空格不计)。

### (4)REPLIC 函数

函数功能:调用该函数得到一字符串,该字符串由给定的第一个参数字符串(字符串尾部空格不计)重复组成,重复次数由第二个参数给出。

### (5)UPPER 函数

函数功能:字符串中的小写字符转换为大写字符(字符串尾部空格不计)。

### (6)LOWER 函数

函数功能:字符串中的大写字符转换为小写字符(字符串尾部空格不计)。

把附后的源程序 FUNC. FOR 用 FORTRAN 77V3.3 编译程序 FOR1. EXE 和 PAS2. EXE 编译产生目标程序,应用 DOS 中的库文件管理程序 LIB. EXE 把上述目标程序加到库文件 FORTRAN. LIB 中。

其方法是打入命令:

>LIB FORTRAN. LIB+FUNC. OBJ;

其中外部命令 LIB. EXE 文件应在当前目录中或在已用 PATH 命令指定的目录中。

用 FORTRAN 语言编制应用程序调用上述函数时,需要注意参数个数和类型及顺序,涉及到的字符变量或字符函数只要用字符型说明语句 CHARACTER \* 127 定义其长度即可,调用函数的方法与调用内部函数相同。

TYPE FUNC. FOR

```

C LEN 函数,求字符串的长度
C 函数类型:整型
C 参数个数:1
C 参数类型:字符型
 FUNCTION LEN(ST)
 CHARACTER * 127 ST
 LEN=0
 DO 10 LEN=127,1,-1
 IF(ST(LEN;LEN).NE.CHAR(32))RETURN
10 CONTINUE
 RETURN
 END
C INDEX 函数,子字符串搜索
C 函数类型:整型
C 参数个数:2
C 参数类型:字符型,字符型
 FUNCTION INDEX(ST1,ST2)
 CHARACTER * 127 ST1,ST2
 L1=LEN(ST1)
 L2=LEN(ST2)
 DO 10 INDEX=1,L1-L2+1
 IF(ST1(INDEX;L1+INDEX-1).EQ.ST2)
10 CONTINUE
 INDEX=0
 RETURN
 END
C PLUS 函数,字符串联结
C 函数类型:字符型
C 参数个数:2
C 参数类型:字符型,字符型
 CHARACTER * 127 FUNCTION PLUS(ST1,ST2)
 CHARACTER * 127 ST1,ST2

```

```

 L1=LEN(ST1)
 L2=LEN(ST2)
 PLUS(1;L1)=ST1
 PLUS(L1+1;L1+L2)=ST2
 DO 10 I=L1+L2+1,127
10 PLUS(I;I)=CHAR(32)
 RETURN
 END
C REPLIC 函数,字符串重复
C 函数类型:字符型
C 参数个数:2
C 参数类型:字符型,整型
 CHARACTER * 127 FUNCTION REPLIC(ST,N)
 CHARACTER * 127 ST,PLUS
 REPLIC=CHAR(32)
 DO 10 I=1,N
 REPLIC=PLUS(REPLIC,ST)
10 CONTINUE
 RETURN
 END
C UPPER 函数,字符串小写转变为大写
C 函数类型:字符型
C 参数个数:1
C 参数类型:字符型
 CHARACTER * 127 FUNCTION UPPER(ST)
 CHARACTER * 127 ST
 UPPER=ST
 DO 10 I=1,LEN(ST)
 IA=ICHAR(UPPER(I;I))
 IF(IA.GE.97.AND.IA.LE.122)UPPER(I;I)=
 CHAR(IA-32)
10 CONTINUE
 RETURN
 END
C LOWER 函数,字符串大写转变为小写
C 函数类型:字符型
C 参数个数:1
C 参数类型:字符型
 CHARACTER * 127 FUNCTION LOWER(ST)
 CHARACTER * 127 ST
 LOWER=ST
 DO 10 I=1,LEN(ST)
 IA=ICHAR(LOWER(I;I))
 IF(IA.GE.65.AND.IA.LE.90)LOWER(I;I)=
 CHAR(IA+32)
10 CONTINUE
 RETURN
 END

```

# CPAV 防病毒软件使用简介

交通部水运所(100088) 吴桦

美国 Central Point Software 公司 1991 年新推出的防病毒软件包 CPAV(Central Point Anti Virus)1.0 版,以其强大的功能和友好的用户界面而大受欢迎。该软件包可用于单用户和网络系统,并与 Windows 3 全兼容。本文对该软件包的三大核心软件:BOOTS SAFE.EXE,VSAFE.COM 和 CPAV.EXE 的用法做一简要介绍。

## 一 BOOTS SAFE.EXE 的用法

BOOTS SAFE.EXE 是一个以监视主引导区和文件分配表为手段的防病毒软件。在 DOS 提示符下,键入 Bootsaf e/? 并回车,可以得到怎样使用命令参数的英文帮助信息,其一般命令格式为:

Bootsaf e[驱动器 1][驱动器 2][驱动器 3].../选择项

常用的选择项有:

/M——建立一个新的引导区备份文件。

/R——用 A 驱动器中的引导区备份文件使引导区受损的磁盘引导区得以恢复

命令中若无驱动器号,则对当前驱动器操作。

Bootsaf e 首先检查内存有无病毒或被非法减小,若发现有其可清除的病毒,先将病毒除掉,若内存被非法减小而原因不明。则显示如下:

Memory has been suspiciously reduced, the cause may be an unknown virus. Press any key

提示指出:不知什么原因,内存减小了。可能是由某种不认识的病毒造成的,按任意键退出。

如果内存检查未发现可疑处并且盘内尚无叫 Boot. cps 的引导区备份文件,将弹出如下窗口提示:

Image file not found. Do you want to Create a new image file?  
[ ]Yes [ ]No

键入“N”即退出 Bootsaf e,返回 DOS。键入“Y”则在盘上建立一个名为 Boot. cps 的引导区备份文件。如果是对硬盘执行 Bootsaf e,还将弹出如下提示:

Save Image to Floppy. In Case of damage to the partition table, an image of it should be saved on to a diskette. Save partition table to drive A:?[Y]yes [N]No Insert back up disk in drive A:. Press any key

窗口内首先询问是否将该备份文件存到软盘上。键入“N”不把备份文件存入软盘,窗口消失。键入“Y”,则窗口内提示将软盘放入 A 驱动器后击任意键,按提示执行后,窗口内出现如下信息:

Partition table was backed up to file: A\\* BOOT. CPS

Press any key

其中 \* 为驱动器号,表示备份文件已经以 \* BOOT. CPS 名的形式存于软盘中。此时再击任意键,窗口下面出现如下提示:

Drive \* : Boot Images successfully created

表示已成功建立了引导区备份文件。这样,以后再运行 Bootsaf e 时,将把引导区内容与原备份文件的内容相

比较。若一致则给出如下信息:

Drive \* : Boot sector and Partition table are OK

表示主引导区和硬盘分区表完好。若引导区被破坏,可通过存于软盘的引导区备份文件恢复原来的引导区和硬盘分区表。例如:

A:Bootsaf e C:/R 表示恢复 C 盘引导区

Bootsaf e D:/R 表示恢复 D 盘引导区

首次进行备份时,程序是不管引导区是否有病毒的。所以首次运行 Bootsaf e 一定要保证主引导区和分区表是干净无毒的,否则,就会备份出一个带毒的引导区,那就失去使用 Bootsaf e 的意义了。

## 二 VSAFE.COM 的用法

VSAFE.COM 是一个以驻留内存形式动态监视病毒活动的软件。在 DOS 提示符下,键入 VSAFE [驱动器号]并回车后,VSAFE 将进驻内存,约占 24K 的内存空间。同时键入 ALT+V 键则激活控制窗口。窗口中的 8 个选择项依次是:

1. 硬盘低级格式化警告(HD Low level format)
2. 驻留内存警告(Resident)
3. 广泛的写保护(General write protect)
4. 检查执行文件(Check executable files)
5. 引导区型病毒警告(Boot sector viruses)
6. 保护硬盘引导区(Protect HD boot sector)
7. 保护软盘引导区(Protect FD boot sector)
8. 保护可执行文件(Protect executable files)

这 8 个选项均为开关项,即键入某选项的数字,如出现“√”表示将该选项打开,如“√”消失,将该选项关闭,根据需要选定配置后,按 Esc 键,则这种配置在系统重新启动前将一直保持,还可采用另一种方式即命令参数的方式来选定配置。

命令格式为:

VSAVE /1±/2±/3±/4±/5±/6±/7±/8±

对每项选“+”表示打开该选项,选“-”表示关闭该选项。可把这种命令格式放在一个批处理文件中,则只要运行该批文件,VSAFE 就驻留内存。如果不想使 VSAFE 驻留内存,只要在激活控制窗口后,同时键入 ALT+U 键,就可将 VSAFE 由内存中撤去。

## 三 CPAV.EXE 的用法

CPAV.EXE 是一个可查消 500 多种病毒,并采用对执行文件加免疫等手段来防止现有和将来可能产生的各种病毒侵袭的软件。键入 CPAV [驱动器号]并回车后,CPAV 首先查内存有无病毒或被非法减小。如未发现异常,则显示主屏幕。屏幕最上边一行为主菜单,有四个功能选项,自左至右为检查项(Scan)、选择项(Options)、配置项(Configure)和帮助项(Help),下面一行反映当前驱动器和路径。其下面的三个小窗口自左至右分别为文件信息,病毒信息和执行操作信息。

三个小窗口下面的两个大窗口中,左边为当前盘的目录树。右边为当前目录下的文件清单,最下一行是 10 个功能键选项。

主屏幕内各项的操作与 PCTools 6.0 版等软件的法大致相同。例如:击“Tab”键在目录树窗口与文件清单窗口之间进行切换。用↓、↑键选择子目录和文件。对选中的项目击回车键予以确认。在彩显中,选中的内容以蓝色显示;在单显中,选中的内容以高亮度显示。激活主菜单的方法有多种。常用的两种方法是:

(1)按 F10 功能键后,光条移到主菜单栏。用→、←键将光条移到欲选的功能项菜单上,击回车键或↓键调出下拉式菜单,然后用箭头键可在功能项菜单选项内和菜单之间进行选择。击回车键确认选择。

(2)对各选项直接击其异色或高亮度字母

下面对各主要选项予以说明

1. 选择功能菜单(Options)

共 11 个选项,均为开关项“√”表示打开选项,无“√”表示关闭选项。

(1)整体校验(Verify Integrity)

打开此项,查病毒时,将所查的执行文件与原来运行 CPAV 时建立的档案文件进行比较。发现有变化时,弹出如下窗口信息。

```
Verify Error
File:EDLIN.COM has been changed.
 From To
Attribute:
Time : 12 : 00 : 00 10 : 00 : 00
Date : 08/15/1989 07/24/1987
Size : 7508 7495
Checksum FD80 FD80
```

Update Delete Continue Stop

选“Update”则以新信息代替老信息。以后再查病毒,查到此文件时将不再报警;选“Delete”则删除这个已变化的文件;选“Continue”继续执行;选“Stop”停止执行,返回到上一级菜单,该选项默认为打开状态。

(2)建立新的文件档案(Create New Checksums)

打开此项,每查完一个子目录,将在该子目录中建立一个名为 CHKLIST.CPS 的档案文件。这个文件含有该子目录中所有的执行文件的大小、属性、日期、时间等基本信息。如果目录中已有 CHKLIST.CPS 文件,对任何新增加到该目录中的文件,再执行 CPAV 时,会将新文件的有关信息加到 CHKLIST.CPS 文件中。该选项默认为打开状态。

(3)在软盘中建立档案文件(Create Checksums on Floppy)

此项对软盘中每个子目录检查完后,也建立名为 CHKLIST.CPS 的档案文件。该选项默认为关闭状态。

(4)解除报警声响(Disable Alarm Sound)

默认为关闭状态,显示警告信息时无警告声响。

(5)建立备份(Create Backup)

打开此项,则任何被感染文件在被清除前都会被备份,备份文件扩展名为 VIR。打开这项意味着允许带毒文件仍存在盘上。只有当被感染文件是唯一而又是十分重要的时候,才可打开此项。

(6)建立报告(Create Report)

打开此项,则运行完 CPAV 后,在当前盘的根目录中产生一个名为 CPAV.RPT 的报告文件。通过查阅这个报告文件,可以了解在哪些文件中发现了哪些病毒。清除了多少病毒等信息。该选项默认为关闭状态。

(7)检查时随时反应(Prompt while Detect)

打开此项,一旦检查中发现病毒,立即显示一窗口,通过此对话框,可以修复(repair)被感染文件、不修复被感染文件继续执行 CPAV(continue)或停止执行 CPAV(Stop)。该选项默认为打开状态。

(8)快速检查(Fast Detection)

打开此项时,CPAV 只检查病毒通常传染的部分而不是检查整个文件,实现快速查毒,一般几十兆至一百兆的硬盘几分钟内就可查完。如果对快速检查或对某些文件还不放心,可关闭该选项,那么对选中的文件,CPAV 将进行全部已知病毒特征码的检查,这样自然要多花些查毒时间。该选项默认为打开状态。

(9)查全部文件(Check All Files)

打开此项,则对全部文件进行查毒,关闭此项,只对扩展名为 EXE、COM、OVL、OVR、SYS、BIN、APP、CMP 这样的执行文件查毒,该选项默认为打开状态。

(10)只查毒(Detection Only)

打开此项,则涉及清病毒、加免疫等选项将失效,该选项默认为关闭状态。

(11)允许入网(Allow Network Access)

打开此项,可在网络中使用 CPAV。默认为打开。

2. 配置功能菜单(Configure)共 6 个选项

(1)换至快速菜单(Change to Express Menu)

快速菜单是只包含五个命令选项的简化菜单,可完成 CPAV 的主要功能五个选择命令是:

- a, Detect 只查毒
- b, Detect & Clean 查并清毒
- c, Select new drive 改变驱动器
- d, Full Menu 回到主菜单栏
- e, Exit 退出 CPAV

击 F8 功能键也可进入快速菜单。

(2)非免疫文件表(Immunization Exceptions)

CPAV 可对大多数执行文件加免疫,但为了防止对一些不适于加免疫的文件加免疫。CPAV 在执行加免疫前先查非免疫文件表。对表中已有的文件不再加免疫。选此项后,选“Add”向非免疫文件表中加入新的非免疫文件。选“Remove”去除某些非免疫文件,选“OK”确认选择。

(3)改变警告信息(Change Alert Message)

选此项可改变查到病毒时给出的警告信息

(4)改变口令设置(Change Password)



#### 〈5〉配置存盘(Save Configuration)

此项可对已选的选项和配置项的更改内容存盘。

#### 〈6〉改变驱动器(Change Work Drive)

此项可改变当前驱动器。击 F<sub>2</sub> 键作用同此项。

### 3. 检查功能菜单(Scan)

#### 〈1〉检查(Detect)

该选项只查而不清病毒,当发现病毒或某文件被更改了,将发出警告。F<sub>1</sub> 功能键与此项作用相同。

#### 〈2〉查消病毒(Clean)

选此项时,在查到病毒的同时进行消毒。若发现文件被更改了,将显示一对话框,可选“Update”承认这种变化合法。选“repair”修复该文件,选“Delete”删除该文件,F<sub>3</sub> 功能键与此项作用相同。

#### 〈3〉加免疫(Immunize)

选此项时,将对选定的文件加免疫,加了免疫的文件增加不到 1K 的长度。但并不占据系统内存空间。运行加了免疫的文件时,文件首先进行自检。若发现有变化,将给出下面的信息:

```
Self Integrity Check Warning—File was changed!
```

```
Choose an option
```

```
[R]Self Reconstruction
```

```
[C]Continue execution
```

```
[E]Exit to DOS
```

提示指出:经自检发现文件已变化了,要求做出选择。键入“R”文件自行恢复原状态,键入“C”不理睬这种变化,继续执行,键入“E”退出程序返回 DOS。

对不适于加免疫的文件将给出有关的信息。可把这类文件加到非免疫文件表中。

有时,某些文件被加了免疫后无法正常使用。此时可用撤除免疫选项将这类文件的免疫撤去。F<sub>6</sub> 功能键的作用与加免疫选项相同。

#### 〈4〉撤除免疫(Remove Immunization)

将加了免疫的文件的免疫功能撤除。

#### 〈5〉删除 CHKLIST.CPS 文件(Delete Checklist Files...)

选此项将把选定子目录中的 CHKLIST.CPS 文件删除。

#### 〈6〉病毒清单(Virus List)

此项被选择后,可列出 CPAV 能查消的 500 多种病毒的有关信息,F<sub>9</sub> 功能键与此项作用相同

#### 〈7〉增补病毒特征码(Update Virus List)

若得到由 Central Point 公司提供的新资料,通过选此项输入病毒特征码,可以查出更多种类的病毒。

#### 〈8〉操作日记(Activity Log)

选此项可显示或打印出最近一段时间的检查操作记录。F<sub>7</sub> 功能键与此选项作用相同。

#### 〈9〉退出 CPAV(Exit)

选此项后将弹出如下窗口提示:

```
Do you really want to exit?
```

```
[] Save Configuration
```

```
Exit Cancel
```

问你是否真想退出 CPAV,不想退出选“Cancel”

项。若想退出,选“Exit”项。在选“Exit”项以前,若已更改过配置而在配置功能菜单中未进行配置存盘,此时还有机会存盘。先击“S”键,然后再选“Exit”项。则可将配置存盘后再退出 CPAV。

#### 4. 帮助功能菜单

此项给出一系列帮助信息。此外,无论 CPAV 运行至哪一步,击 F<sub>1</sub> 键均可得到对该步的帮助信息。

以上介绍了 CPAV 主屏幕中各菜单选项的用法及功能,在执行 CPAV 时,随时可击 F<sub>3</sub> 功能键或“Esc”键,则程序中断执行退回上一级菜单项直至最后退出 CPAV。

CPAV 也可采用命令参数的方式运行。如:

```
CPAV /S——查毒
```

```
CPAV /C——清毒
```

```
CPAV /I——查、清毒后加免疫
```

#### 四、使用中的一些技巧。

首先,在确保系统干净无毒的情况下,用 BOOT-SAFE 建立引导区备份文件并将此备份文件存入一个干净无毒的软盘中备用。然后将 CPAV 的软件集中在一个子目录中,子目录名可叫做 CPAV。运行 CPAV,对盘中的文件,至少是像 DOS、PCTools 等这样一些大多数人都要用到的子目录中的文件加免疫。建立起执行文件的内防护层。而后,运行 CPAV 使每个子目录中都建立 CHKLIST.CPS 文件从而建立起文件的外防护层。以后即使不运行 VSAFE,只要文件被病毒感染了。通过运行 CPAV 就可查出。加了免疫的文件还可被恢复。

为了更有效地防病毒。可将三大核心软件加到 AUTOEXEC.BAT 文件的开头部分。使每次开机后,防病毒软件自动进入工作状态。推荐在 AUTOEXEC.BAT 文件开头部分采用如下命令格式:

```
Path=C:\CPAV
```

```
Bootsafe [驱动器号]
```

```
Vsafe /1+/2-/3-/4+/5+/6+/7-/8+
```

```
CPAV /S
```

开机后程序首先检查引导区是否完好,然后 Vsafe 进驻内存实现对病毒行为的动态监视。最后,CPAV 将盘扫一遍,看看盘内文件是否有过任何变化。

对上面的 Vsafe 选项,由于第 1,4,5,6,8 项均为打开状态。所以,一旦出现硬盘将被低级格式化、执行文件带毒、发现引导区型病毒、硬盘引导区将被更改、执行文件将被更改等情况时,就会出现警告信息。出现警告信息窗口后,窗内有不同的选项供选择。以执行文件将被修改为例,三个选项是:“Stop”,选此项则模拟写保护,许多病毒在遇到写保护时则放弃侵扰,所以若选“S”后能通过,则可继续程序的执行。但也有些病毒,遇到写保护时将反复尝试写盘或出现其它一些不正常的情况,如果遇到这种情况,只好选“Boot”项重新启动系统先查消完病毒再运行文件了。还有一种情况,程序被修改并不是由于病毒作用而是由于某些正常原因,如使用 DEBUG,用 PCTools 压缩某文件等,此时

可选另一个选项“C”。表示这种更改属合法现象,要求继续执行,不过,如果你正用 PCTools 压缩某个子目录的许多文件时,则每执行一个文件压缩就报警一次,那就太麻烦了,此时,最好先关闭第 8 个选项。待文件被压缩完再将第 8 个选项打开。

第 2、3、7 项选关闭状态是因为:对第 2 项,如选打开则只要程序要驻留内存就会报警,因而在运行一些中文软件如“2.13”“WPS”等时,会出现误报。并且,由于某些中文软件与西文软件不完全兼容,还会出现不显示提示信息甚至死机现象。关闭此选项后,虽然病毒有可能被允许进驻内存,但只要其它几项,特别是第 8 项关口把牢,就不会产生什么危害。还有一点需要说明的是,由于中文系统采用图形方式显示,在中文系统下执行程序时,如遇到警告情况,报警信息常不能显示出来,但会有报警声响。比如在“2.13”下,运行 WS.COM,假如病毒感染了该文件,你会听到报警声而见

不到报警信息。对这类情况,可按 Ctrl+F7 键转至西文状态或退出中文系统直接在西文状态下再运行该文件,就可看到报警信息,从而判断是否有病毒出现。

VSAFE 中第 3 项选关闭是因为若将其选打开则任何文件拷贝时均要报警。第 7 项由于是保护软盘引导区,不属于保护硬盘范畴,故取关闭,否则当格式化软盘等操作时均会产生误报,这个选项可根据需要临时打开。例如,当您准备用软盘传递一些文件时,打开此项可避免病毒进入软盘引导区。

当 VSAFE 驻留内存时,对程序的执行速度可能会产生一定的影响。一般情况下,这种影响并不明显,如果是进行某些大型运算。可采用下面的方式:在 VSAFE 驻留内存的情况下,先运行一下执行文件。未见异常则中断程序运行。激活控制窗口后,按 ALT+U 键撤掉 VSAFE。然后再重新运行该执行文件,这样就可保证运算速度不受影响了。

## 带检索功能的通讯录录入程序

武汉市同济医科大学研究中心(430030)张春明

在数据库管理系统中,录入模块是必不可少的。当数据量很大时,难免录入重复的记录项,这样势必加大数据库的冗余度。

笔者编写了一个带检索功能的通讯录录入程序,可以有效地防止重复数据的录入,基本编程思想是:建立两个结构相同的数据库,一个作为主库(TXL.DBF),并以姓名为关键字建立索引文件;另外一个作为录入当前库(HYTX.DBF)。录入时首先输入姓名,如果在主库中检索到相同姓名的人员,则将该记录的全部内容用 REPL 命令赋值予当前库的各个字段,然后可在屏幕上修改各个字段的内容;反之则在两库中分别追加一个空记录,然后输入各个字段的内容。为了减少输入量,将前一记录中若干字段的值预先赋给当前记录相应的字段。尽管在当前库中可能出现相同姓名的人员(也可能不是同一人),但在主库中不会出现相同姓名的人员。

这段程序尤其适合于登记会议通讯录。因为在专业或行业会议中,许多人都是前次会议的参加者,一旦输入其姓名,相应的资料就调出,大大方便了输入操作。只要没有重新登记,当前库中的人员就是参加会议的人员。如需打印会议通讯录,只要在当前库中指定所需的字段进行打印即可。登记和打印结束后,可将当前库清空或删除,或者在下次运行程序中回答询问“重新开始登记吗?”时键入 Y 也可。

```
set talk off
set inte off
sele 2
use txl inde txl
```

```
sele 1
use hytx
set rela to 姓名 into b
go bott
clea
y='Y'
@ 4,5 say '重新开始登记吗? (Y/N)'get y
read
if y $ 'yY'
set safe off
zap
set safe on
endi
xb=性别
zc=职称
dw=单位
dz=地址
yb=邮编
dh=电话
appe blan
repl 性别 with xb,职称 with zc,单位 with dw
repl 地址 with dz,电话 with dh,邮编 with yb
do while. t.
clea
@1,9 say recn()
@2,13 say '姓名:'
@3,13 say '性别:'
@4,13 say '专业职称:'
@5,13 say '行政职务:'
@6,13 say '单位名称:'
@7,13 say '电话号码:'
```

```

@8,13 say '单位地址:'
@9,13 say '邮政编码:'
@10,13 say '备注:'
@2,23 get 姓名
read
xm=姓名
sele 2
seek xm
if eof()
 appe blan
 sele 1
else
 sele 1
 repl 性别 with b->性别,职称 with b->职称
 repl 职务 with b->职务,单位 with b->单位
 repl 地址 with b->地址,电话 with b->电话
 repl 邮编 with b->邮编
 repl 备注 with b->备注
endi
@3,23 get 性别
@4,23 get 职称
@5,23 get 职务
@6,23 get 单位
@7,23 get 电话
@8,23 get 地址
@9,23 get 邮编
@10,23 get 备注
read
if len(trim(姓名))=0
 xh=recn()
 dele for recn()=xh
 pack

```

```

endi
@12,5 say '是否登记下一位?(Y/N)'get y
read
sele 2
repl 姓名 with a->姓名,性别 with a->性别
repl 职称 with a->职称,职务 with a->职务
repl 单位 with a->单位,电话 with a->电话
repl 地址 with a->地址,备注 with a->备注
repl 邮编 with a->邮编
if len(trim(姓名))=0
 xh=recn()
 dele for recn()=xh
 pack
endi
@12,0 clea
if .not. y $ 'Yy'
 clos data
 clear
 exit
endi
sele 1
xb=性别
zc=职称
dw=单位
dz=地址
yb=邮编
dh=电话
appe blan
repl 性别 with xb,职称 with zc,单位 with dw
repl 地址 with dz,电话 with dh,邮编 with yb
endd

```

(上接第2页)

行平行处理。如：四个 C-Cube CL550-35 芯片能实时压缩或还原高清晰度 TV (HDTV) 视频路象。

问：如何将 C-Cube CL550 处理器安装在系统中？

答：C-Cube CL550 能嵌入任何处理装置内，如计算机外设（图象板、彩色打印机，彩色复印机和扫描仪），彩色传真机；数字式相机；数字 VCR 和桌面系统中。

问：C-Cube CL550 有哪些开发工具？

答：C-Cube CL550 开发系统板/M 是一种 OEM 开发卡，可插入 Mackintosh I 系列计算机的总线插座。C-Cube CL550 开发板/PC，也是一个 OEM 开发卡，采用 ISA 总线，用于 IBM PC/AT 及其兼容机。它们是了解 CL550 性能及开发 CL550 应用的有力工具。

问：C-Cube CL550 开发工具都包括些什么？

答：C-Cube CL550 开发工具/M 包括 Mackintosh I 兼容的 C-Cube CL550 开发板；软件驱动程序；demo 软件包等和一份压缩监控程序的拷贝件。

C-Cube CL550 开发工具/PC，包括用于 ISA 总线的开发板；软件驱动程序；demo 软件包和一份压缩监控程序的拷贝件。

问：目前国内何处可咨询 C-Cube CL550 方面的问题及订货？

答：请同中国电子器材深圳公司联系。

地址：广东省深圳市振华路 414 幢电子器材大厦

电话：354214、354345、350877 电挂：8410

传真：350876 邮编：518031

联系人：黄建新

# PC 机中系统时间的自动显示

北京六十一中(100007) 郑嘉琦

这里向大家介绍一个名为 DTIME.COM 的程序。可当 DOS 外部命令使用,只要执行一次之后,立即开始将系统时间的时、分、秒在屏幕右上角自动显示。在走时显示的同时,计算机完全可以照常使用。

在 IBM PC 机的中断系统中有一个 INT 1C,它被放置在8号硬件中断服务程序的结尾处。DOS 在初始化时,使其中断向量指向一个不执行任何任务的例程(它只包含一条 IRET 指令)。这是有意留给用户用来扩展8号中断的功能而专门按排的。由于8号中断固定每55毫秒发生一次,因而利用 INT 1C 可以实现一些实时操作功能。这里利用它进行计时并加以自动显示。具体做法是:编制一个自动显示时间的新例程,先让其常驻内存,然后修改 INT 1C 的中断向量使指向新例程。一般可以利用 INT 27或功能调用31H 让程序常驻内存。本文中自动显示时间的新例程较为短小,就用了将其放到中断向量表的空间部分的方法来实现常驻。具体存放地址为0:200H~0:294H。另外0:295H,296H,297H 三个单元作为计时器,以BCD 码的形式依次存放时、分、秒数;0:298H 作为计数器,用来计8号中断发生的次数,每计满18次作为一秒。由于8号中断实际每秒发生18.2次,因此例程中在每分钟和每小时的最后一秒进行一次校正调节。

DTIME.COM 用 DEBUG 的 A 命令直接生成最为简便。下面的清单除中文注释之外,既列出了程序内容,也表明了 DTIME.COM 生成的方法过程。例程是针对单色显示器编制的。如果用彩色显示器。只需将08F1:014D 处的代码改为 MOV AX,B800。

如果对中断向量表中的0:200H~0:298H 共153个单元有使用安排,需要将 DTIME.COM 做简单修改,改用 INT 27或功能调用31H 使新例程常驻内存,新例程本身无需修改。

A>DEBUG

-A100

```
08F1:0100 XOR AX,AX ;将自动显示时间的新例
08F1:0102 MOV ES,AX 程存放到中断向量表的
081F:0104 CLD 空闲部分常驻。
08F1:0105 MOV SI,14A
08F1:0108 MOV DI,0200
08F1:010B MOV CX,0095
08F1:010E REPZ
08F1:010F MOV SB
08F1:0110 MOV AH,2C ;取当前时间。
08F1:0112 INT 21
08F1:0114 MOV BX,0295; ;将当前时间的时、分、秒
08F1:0117 XOR AX,AX 作为初值赋给计时器
08F1:0119 MOV DS,AX 0:295,296,297
08F1:011B MOV AH,CH
```

```
08F1:011D CALL 013A
08F1:0120 MOV AH,CL
08F1:0122 CALL 013A
08F1:0125 MOV AH,DH
08F1:0127 CALL 013A
08F1:012A MOV AL,12 ;给计数器0:298赋初值。
08F1:012C MOV [0298],AL
08F1:012F MOV DX,0200 ;改变 INT 1C 的中断向量,
08F1:0132 MOV AL,1C 使其指向新例程。
08F1:0134 MOV AH,25
08F1:0136 INT 21
08F1:0138 INT 20; ;结束。
08F1:013A XOR AL,AL ;将 AH 中的数码转换
08F1:013C AND AH,AH 成 BCD 码并存入内存
08F1:013E JZ 146 的子程序。
08F1:0140 INC AX
08F1:0141 DAA
08F1:0142 DEC AH
08F1:0144 JNZ 0140
018F:0146 MOV [BX],AL
018F:0148 INC BX
08F1:0149 RET
08F1:014A PUSH BX ;从此至1DE 是自动显示
08F1:014B PUSH ES 时间的新例程。
08F1:014C PUSH DI
08F1:014D MOV AX,B000 ;将计时器中存放的时、
08F1:0150 MOV ES,AX 分、秒转换成 ASC II 码
08F1:0152 XOR AX,AX 后显示到屏幕右上角。
08F1:0154 MOV DS,AX
08F1:0156 MOV DI,008E
08F1:0159 MOV DH,70
08F1:015B MOV,BX,0294
08F1:015E CALL 01BE
08F1:0161 MOV DL,2E
08F1:0163 CALL 01D9
08F1:0166 CALL 01BE
08F1:0169 MOV DL,3A
08F1:016B CALL 01D9
08F1:016E CALL 01BE
08F1:0171 MOV AL,[0298] ;计数器298减一计数,
08F1:0174 DEC AL 每减到零为一秒并重
08F1:0176 MOV [0298],AL 新赋初值
08F1:0179 JNZ 01BA
08F1:017B MOV AL,12
08F1:017D MOV [0298],AL
08F1:0180 MOV AL,[0297] ;计时器计秒。
08F1:0183 INC AL
08F1:0185 DAA
08F1:0186 MOV [0297],AL
08F1:0189 SUB AL,60 ;满60秒吗?
```

08F1:018B JNZ 01BA  
 08F1:018D MOV [0297],AL ;满,计秒单元回零。  
 08F1:0190 MOV AL,1E;进行一次校正调节(粗调)\*。  
 08F1:0192 MOV [0298],AL  
 08F1:0195 MOV AL,[0296] ;计时器计分。  
 08F1:0198 INC AL  
 08F1:019A DAA  
 08F1:019B MOV [0296],AL  
 08F1:019E SUB AL,60 ;满60分吗?  
 08F1:01A0 JNZ 01BA  
 08F1:01A2 MOV [0296],AL ;满,计分单元回零。  
 08F1:01A5 MOV AL,18;进行一次校正调节(细调)\*。  
 08F1:01A7 MOV [0298],AL  
 08F1:01AA MOV AL,[0295] ;计时器计小时。  
 08F1:01AD INC AL  
 018F1:01AF DAA  
 08F1:01B0 MOV [0295],AL  
 08F1:01B3 SUB AL,24 ;满24小时吗?  
 08F1:01B5 JNZ 01BA  
 08F1:01B7 MOV [0295],AL ;满,计小时单元回零。  
 08F1:01BA POP DI  
 08F1:01BB POP ES  
 08F1:01BC POP BX  
 08F1:01BD IRET ;中断返回(到 INT8的结尾处)  
 08F1:01BE INC BX ;将 AL 中的 BCD 码转换  
 08F1:01BF MOV AL,[BX] 成 AX 中的两个 ASC I 码  
 08F1:01C1 MOV AH,AL 并显示到屏幕右上角的  
 08F1:01C3 AND AL,0F 子程序。

08F1:01C5 SHR AH,1  
 08F1:01C7 SHR AH,1  
 08F1:01C9 SHR AH,1  
 081:01CB SHR AH,1  
 081:01CD OR AX,3030  
 08F1:01D0 MOV DL,AH  
 08F1:01D2 ES;  
 08F1:01D3 MOV [DI],DX  
 08F1:01D5 INC DI  
 08F1:01D6 INC DI  
 081:01D7 MOV DL,AL  
 08F1:01D9 ES;  
 08F1:01DA MOV [DI],DX  
 08F1:01DC INC DI  
 08F1:01DD INC DI  
 08F1:01DE RET  
 08F1:01DF ↙

—N DITIME.COM

—RCX

CX 0000

:DF

—W

;将 DITME.COM 的长度  
放入 CX 寄存器

;将 DTIME.COM 存盘备  
用。

Writing 内 00DF bytes

\*注:走时的快慢可通过改变0:247H 和0:25CH 两个单元中的  
数码来调节。

## 最近出现的几种病毒

中国地质大学(北京)(100083) 苏民生

近期出现几种病毒,有些是现有检查与消除病毒软件(如 KILL V50.00)尚不能查出和消除的,已造成一些破坏,需引起注意。

### 一、GenB/GenP 病毒

系统型,改变软盘 BOOT 区及硬盘主引导记录(0面0道1区)。病毒部分占据第0—1BD 字节,其中第1B0—1BD 字节内容为字符串 RMBDRMCCQBD WRM,似为明显标志。硬盘感染后,仍能启动。感染软盘后,软盘不能启动,使用该软盘引导系统时,先执行占用了 BOOT 区的病毒程序,然后转到用硬盘引导。可导致失去硬盘。

KILL V50.00不能检出该病毒。版本8.4B89的 SCAN 可查出,定名为 GenP(对于硬盘)和 GenB(对于软盘)。同一版本的 CLEAN 可消除,但有时消除失败。

### 二、DIR—2(D2)病毒

文件型,感染 COM 文件和 EXE 文件。不改变文件大小和形成(最后修改)日期。驻留内存。用8.4B89版本的 SCAN 和 CLEAN 可检查与消除。破坏文件分

配表,使文件损坏或丢失。用 KILLV50.00检不出。用 CLEAN 消除之前必须关掉电源后用好的系统盘重新冷启动。感染病毒的文件往往无法复制(COPY)。

### 三、779/5病毒

文件型。感染 COM 文件和 EXE 文件。对于 COM 文件,改变后来文件前16字节,并将病毒部分779(30BH)或更多字节接在原来文件之后。对于 EXE 文件,仅在原来文件之后续接5字节或稍多字节内容。不改变原来文件内容。往往修改文件形成日期,有时又不修改。带病毒 COM 文件开头为 E9XXXX00 YYYYY 2219……,其中 XXXX 和 YYYYY 是和原来文件长度有关的数据。现有检查和消除病毒软件尚不能判断和消除该病毒。尚未发现破坏情况。

### 四、901/910病毒

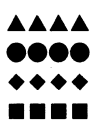
文件型。感染 EXE 文件,使其增大901至910字节。改变文件日期。除发现文件变大和改变日期之外,未发现其它破坏。现有检查和消除病毒软件尚不能查出。

# 电子扭计板

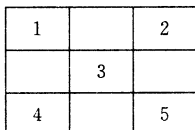
大庆市二十四中计算机室(163712) 葛建华 吴立国

扭计板是80年代风靡美国的智力玩具。其玩法类似于魔方,实际上是魔方的一种平面化方式,只是难度稍低。我们在PC机的2.13操作系统下用BASIC语言编写了一个电子扭计板的程序,操作方式与实际的扭计板一样,其游戏规则如下:

首先程序设定初始状态,如图(1),左上角和右上角分别为两个三角形和两个圆形方阵,中间为两个圆形和两个菱形方阵,左下角和右下角分别为两个菱形和两个正方形方阵。这五个方阵分别以1~5键定义,如图(2),每个方阵均可顺时针或逆时针方向旋转,每旋转一次为一次操作,并在屏幕上显示操作次数。由“K”键定义为顺逆时针旋转的转换键,并由“=>”或“<=”符号提示。由“R”键定义为返回初始状态。要求通过这五个方阵的旋转,将图形的初始状态旋转到目标状态,如图(3),图(4),图(5),图(6)等多种类型。



图(1)



图(2)



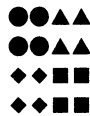
图(3)



图(4)



图(5)



图(6)

由不尽相异元素的排列公式可算出共有 $16! \div (4! \times 4! \times 4! \times 4!) = 63063000$ 种排列方式,趣味无穷,最后可由“Q”键退出。

```

15 CLS:SCREEN 1,0:P=1:DIM A(4,4)
22 C$(1)="▲";C$(2)="●";C$(3)="◆";C$(4)
 ="■"
28 REM 30~60行是初始状态的设定
30 FOR I=1 TO 4:FOR J=1 TO 4
50 A(I,J)=I
60 NEXT J,I
65 M1=1:N1=1:M2=4:N2=4
70 GOSUB 5000
100 A$=INPUT$(1)
102 IF A$="q" THEN CLS:END:REM 按“Q”键退出
104 IF A$="r" THEN Z=0:GOTO 15
108 REM "k"键是方阵顺时针和逆时针旋转的转换键

```

```

110 IF A$="k" THEN P=P*(-1):IF P=1 THEN
 LOCATE 4,32,0:PRINT "<="ELSE LOCATE 4,
 32,0:PRINT "=>"
118 REM 1~5的数码键分别是左上,右上,中间,左下,右
 下5个方阵的操作键
120 IF VAL(A$)>5 OR VAL(A$)<1 THEN 100
122 Z=Z+1
130 ON VAL(A$) GOSUB 200,300,400,500,600
135 M2=M1+1:N2=N1+1
140 GOSUB 1000:GOTO 100
200 X1=1:X2=2:Y1=1:Y2=2:M1=1:N1=1:RE-
 TURN
300 X1=3:X2=4:Y1=1:Y2=2:M1=1:N1=3:RE-
 TURN
400 X1=2:X2=3:Y1=2:Y2=3:M1=2:N1=2:RE-
 TURN
500 X1=1:X2=2:Y1=3:Y2=4:M1=3:N1=1:RE-
 TURN
600 X1=3:X2=4:Y1=3:Y2=4:M1=3:N1=3:RE-
 TURN
1000 T=0:FOR I=Y1 TO Y2
1002 FOR J=X1 TO X2
1004 T=T+1:B(T)=A(I,J)
1006 NEXT J,I
1010 IF P=1 THEN GOSUB 2000
1020 IF P=-1 THEN GOSUB 3000
1030 GOSUB 5000:RETURN
2000 T=0:REM 逆时针旋转子程序
2010 FOR J=X1 TO X2:FOR I=Y2 TO Y1 STEP-1
2030 T=T+1:A(I,J)=B(T)
2050 NEXT I,J
2100 RETURN
3000 T=0:REM 顺时针旋转子程序
3010 FOR J=X2 TO X1 STEP-1:FOR I=Y1 TO Y2
3030 T=T+1:A(I,J)=B(T)
3050 NEXT I,J:RETURN
5000 REM 打印子程序
5002 COLOR 9,0
5010 FOR I=M1 TO M2:FOR J=N1 TO N2
5030 LOCATE I*2,5*J,0:PRINT C$(A(I,J));
5040 NEXT J:PRINT:PRINT
5060 NEXT I
5070 LOCATE 10,10,0:PRINT "操作次数:":Z;
5080 IF P=1 THEN LOCATE 4,32,0:PRINT "<=":
 RETURN
5090 LOCATE 4,32,0:PRINT "=>":RETURN

```

# 第四届国际信息学奥林匹克竞赛试题

(第一轮竞赛试题)

## 海中的岛

用  $N \times N$  的网格表示海, 网格中的 \* 表示岛, 我们的任务是重构一张给定的地图。地图中岛的组成由垂直方向和水平方向的编码信息给出, 下例是一张地图的编码。

```
* * * 1 2
* * * * 3 1
* * * * * 1 1 1
* * * * * 5
* * * * * 2 1 1
 * 1
1 1 4 2 2 1
1 2 3 2
1
```

岛可以编组, 称为“岛组”, 上图每行右边的几个数字, 自左至右依次表示各“岛组”的大小, 例如第一行数字 1、2 表示该行有两个“岛组”, 第一个“岛组”有一个岛, 第二个“岛组”有两个岛。每个“岛组”的左右都是海, 海的宽度假定是任意的。类似地, 第一列下边的数字 1、1、1 表示本列自下而上有三个“岛组”, 每个“岛组”仅含一个岛。

任务:

试编一个程序, 重复以下步骤, 直到给定的输入文件 (ASCII 文件) 中的信息组全部读完为止。

1. 从输入文件中读入一个信息块, 并将其显示到屏幕上。每个信息块的组成如下: 网格大小  $N$ , 其后是  $N$  行“岛组”编码数据; 再后面是  $N$  列“岛组”编码数据, 数据中数字间应有空格用来分开, 最后用 0 作为每行或每列数据结束标志。

举例:

```
6 例1. 对应上面的地图, N=6
1 2 0 ←第一行数据
3 1 01 1 1 0
5 0
2 1 1 0
1 0
1 1 1 0 ←第一列数据
1 2 0
4 0
2 3 0
2 0
1 2 0
4 例2. 解为:
0 列
1 0 1 2 3 4
```

```
2 0 1
0
0 行 2 *
1 0 3 * *
2 0 4
0
```

例3. 注意: 据此数据, 构不成地图

```
2
0
0
2 0
2 0
```

例4. 注意: 据此数据, 可以构成两张不同的地图

```
2
1 0
1 0
1 0
1 0
```

2. 重构这张地图并将其显示到屏幕上, 若有多个解, 要构成每张图, 参看例4。

3. 将所构成的地图以 ASCII 文件形式输出, 每个岛用“\* 后跟空格”表示; 空白用连续两个空格表示, 若有多个解, 其间用空行隔开, 若无解 (即构不成地图), 则用一行文字“no map”表示。

由不同信息组所求得解用一行文字“Next Problem”隔开。

技术限制:

1.  $1 \leq N \leq 8$

2. 源程序名为“C:\IOI\DAY-1\413-PROG.PAS”

3. 用于读入的带有岛图的编码信息输入文件名为“C:\IOI\DAY-1\413-SEAS.IN”

4. 将地图写成 ASCII 文件的输出文件名为“C:\IOI\DAY-1\413-SEAS.OU”

文件举例:

为使你方便, 这里提供了举例文件, 其中含测试数据及正确的结果输出, 这些文件在

“C:\IOI\DAY-1\413-SEAS.IN”和

“C:\IOI\DAY-1\413-SEAS.OU”中。

警告: 倘若你的程序能够正确运行上述例子, 但这并不能保证你的程序是正确的。

评分标准:

• 由输入文件读一个信息块并加以显示 5分

- 一个接一个地处理所有的信息块,直至输入文件被读完为止。 10分
- 对每一个信息块,倘若有解就重构一张地图并加以显示 35分
- 将解出的地图写入输出文件 5分
- 倘若有不同的解,重构所有可能的地图并显示之。 20分
- 将全部解的图按要求分开写到输出文件 10分
- 标出无解的信息块 5分
- 遵守技术限制条件 10分

满分 100分

### 第二轮竞赛试题

#### 爬山

某爬山俱乐部有 P 个爬山队员,分别编号为 1、2、……、P。每个爬山队员的爬山速度都相同;且上山、下山的速度也相同。编号为 i 的爬山队员每天的物资消耗量为 C(i),其最大携带物资的能力为 S(i)。C(i)和 S(i)均为整数,计量单位相同。

假定一个爬山队员在有足够供应物资的情况下,爬到山顶所需时间为 N 天。山可能非常高,单个一人爬山没有能力带足所需的物资。因此,要一组爬山队员同时同地出发,某些队员在到达山顶之前提前下山,以便把他所不再需要的物资转交给其他队员,所有队员在爬山时不得停留。

本题的任务是给爬山俱乐部制定一个爬山计划,其要求是:至少有一个队员必须到达山顶;被选中的爬山队员(组成爬山小组)必须全部回到出发地点。

任务:

编写一个程序,完成如下任务:

1. 从键盘读入 N(从出发点到达山顶的天数)和 P(爬山俱乐部成员个数),以及 S(i)和 C(i),(i 从 1 到 P)。你可以假定所有输入数据均为整数。若输入的数据没有意义,拒绝之。

2. 制定一个爬山计划:选择可能的人组成爬山小组,该组成员为 a(1),a(2),……,a(j),……,a(k),出发时队员 a(j)携带的物资为 M(j),(j 从 1 到 k)。

请注意,对于给定的 N,S(i)和 C(i),也许并不存在能够完成任务的爬山计划。

3. 在屏幕上输出以下信息:

- a. 给出实际爬山小组中的队员数 k;
- b. 全组所需物资总量;
- c. 组中爬山队员的编号 a(1),a(2),……,a(k);
- d. 每一个爬山队员 a(j)在出发时所携带的物资数量 M(j),(j 从 1 到 k);
- e. 爬山队员 a(j)爬了几天后开始下山,(j 从 1 到 k)。(参考前页中的例子,用它的输出格式)

4. 使爬山计划尽可能接近最优。优化的含义是:

- a. 参加爬山的队员人数最少,而且
- b. 在满足条件 a 的情况下,总的物资消耗量最小。

技术限制条件:

限制条件1. 你的源程序文件名应为:  
“C:\IOI\DAY-2\422-PROG.PAS”

限制条件2

当 N 小于 1 或大于 100 时,程序必须拒绝之;  
P 的允许范围是:1 ≤ P ≤ 20。

举例:

以下对话方式是适当的,请采用之。

Days to arrive to top: 4

Number of club members: 5

Maximal supply for climber 1: 7

Daily consumption for climber 1: 1

Maximal supply for climber 2: 8

Daily consumption for climber 2: 2

Maximal supply for climber 3: 12

Daily consumption for climber 3: 2

Maximal supply for climber 4: 15

Daily consumption for climber 4: 3

Maximal supply for climber 5: 7

Daily consumption for climber 5: 1

2 climbers needed, total amount of supplies is 10.

Climber(s) 1, 5 will go.

Climber 1 carries 7 and descends after 4 day(s)

Climber 5 carries 3 and descends after 1 day(s)

Plan another party(Y/N) Y

Days to arrive to top: 2

Number of club members: 1

Maximal supply for climber 1: 3

Daily consumption for climber 1: 1

Climbing party impossible.

Plan another party(Y/N) N

Good bye

文件举例:

为使你方便,我们提供了举例文件,其中含测试数据及正确结果输出,请在目录

“C:\IOI\DAY-2”中查找这些文件。

警告:倘若你的程序能够正确运行上述的例子,但这并不能保证你的程序是正确的。

评分标准:

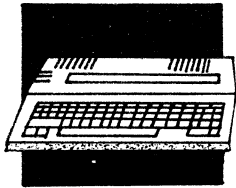
- 按上述例子所提供的方式和用户进行对话: 10分
- 在下述特殊情况下找到一个解,其条件是:  
所有 C(i)=1;所有 S(i)都相等。 20分
- 在一般情况下找到一个解。 20分
- 在一般情况下找到一个接近最优的解 30分
- 能够发现无解的情况 10分
- 遵守技术限制条件 10分

满分

100分

(此文由清华大学计算机系吴文虎教授提供)





# 微机模拟游标卡尺读数训练

学习机之友

宁夏中卫县宣和中学(751706) 王太文

利用游标卡尺进行长度测量是中学物理中所必须掌握的基本技能。由于实际的游标卡尺较小,刻度较密,在课堂上教学生学会使用是很困难的,特别是演示读数无法进行,为配合本节课的教学,我用 BASIC 语言设计了微机模拟游标卡尺读数训练的实用程序,在 APPLE II 机上运行通过,能显示出用十分度游标卡尺测量五种不同形状几何体长度的放大的图形,形象逼真,效果很好。

## 一、运行说明:

程序运行后,图形显示出游标卡尺使用时的初始(读数为零)状态,文本窗显示“PLEASE GIVE ‘FEN—DU’”,要求输入屏幕所显示的游标卡尺的分度(如0.1、0.05、0.02等)。分度给错了,文本窗显示“NO, TRY AGAIN.”告诉你“不对,再试一次”;分度给对了,文本窗显示“YES VERY GOOD!Z=10”,即“对了,很好!得10分”。接下来显示出用游标卡尺测不同形状几何体长度的图形,文本窗显示“L=?”,要求输入读数,读对了,加10分,变换画面(长度)继续进行;读错了,减10分,要求重新读数,直到读对为止。程序运行一次,满分为100分(共10个画面)。

## 二、程序说明:

程序中的 Z 表示分数, M 表示游标的零刻度线所在位置。30—90语句完成初始状态的图形显示及输入数据的处理。100—180语句完成测量过程的图形显示及读入数据的处理,其中110语句利用随机数使游标出现在屏幕上10—180以内的任意位置(以游标的零刻度线为准),120语句根据730子程序给出的 K 值调用800—1100五个子程序之一,画出被测量的一种几何图形。

500子程序画出主尺及其刻度、数字。

600子程序画出游标及其刻度、数字。

730子程序将测量范围分为五个区域,每个区域内测量一种几何体的长度(给开关提供五个数码)。

800子程序画出一个圆,测其直径。

900子程序画出一个“桶”,测其内径。

950子程序画一个直角梯形,测其高。

1000子程序画一个等腰三角形,测其高。

1100子程序画一个矩形,测其长。

该程序是笔者在宁夏教育学院继续教育微机培训班学习的结业程序,指导教师李更生。

```

30 Z=0:HGR:HCOLOR=3:GOSUB 500
40 M=10:HCOLOR=3:GOSUB 600
50 PRINT"PLEASE GIVE 'FEN—DU'":INPUT FD
60 IF FD=0.1 THEN 80
70 PRINT "NO, TRY AGAIN":CALL 64477:GOTO 50
80 Z=Z+10:PRINT"YES, VERY GOOD!","Z=";Z

```

```

90 HCOLOR=0:GOSUB 600:GOSUB 500
100 FOR T=1 TO 9
110 M=10+INT(170 * RND(1)):HCOLOR=3:GOSUB
500:GOSUB 600:GOSUB 730
120 ON K GOSUB 800,900,950,1000,1100
130 S=(M-10)/100
140 PRINT"L=?":INPUT L:IF L>S-0.005 AND L<S
+0.005 THEN 160
150 Z=Z-10:PRINT"NO, TRY AGAIN","Z=";Z:
CALL 64477:GOTO 140
160 Z=Z+10:PRINT"YES, VERY GOOD!","Z=";Z
170 HCOLOR=0:GOSUB 500:GOSUB 600
180 ON K GOSUB 800,900,950,1000,1100:NEXT T:
END
500 HPLOT 10,80 TO 279,80
510 HPLOT 279,65 TO 18,65 TO 18,55 TO 11,50 TO 11,
65 TO 1,65 TO 1,130 TO 9,140 TO 9,95 TO 279,95
520 FOR I=10 TO 279 STEP 10
530 IF (I-10)/50=INT((I-10)/50)THEN 550
540 HPLOT I,75 TO I,80:GOTO 560
550 HPLOT I,71 TO I,80
560 NEXT I
570 HPLOT 107,68 TO 107,73
580 HPLOT 206,68 TO 208,68 TO 208,71 TO 206,71 TO
206,73 TO 208,73
590 RETURN
600 HPLOT M,64 TO M,50 TO M-8,55 TO M-8,64
610 HPLOT M-5,81 TO M-5,97 TO M,97 TO M,140
TO M+8,130 TO M+8,97 TO M+95,97 TO M+
95,81
620 FOR J=M TO M+90 STEP 9
630 IF J=M OR J=M+45 OR J=M+90 THEN 660
640 HPLOT J,81 TO J,85:GOTO 670
660 HPL0T J,81 TO J,88
670 NEXT J
680 HPL0T M-1,89 TO M-1,94 TO M+1,94 TO M+
1,89 TO M-1,89
690 HPL0T M+46,89 TO M+44,89 TO M+44,91 TO
M+46,91 TO M+46,94 TO M+44,94
700 HPL0T M+89,89 TO M+89,94 TO M+91,94 TO
M+91,89 TO M+89,89
710 RETURN
730 IF M<=40 THEN K=1:GOTO 760
732 IF M<=70 THEN K=3:GOTO 760
733 IF M<=100 THEN K=2:GOTO 760
735 IF M<=130 THEN K=4:GOTO 760
736 K=5
760 RETURN
800 R=(M-10)/2
810 FOR O=0 TO 360 STEP 9

```

```

820 X=10+R+R * COS(O * 3.1416/180);Y=129+R *
 SIN(O * 3.1416/180)
830 H PLOT X,Y;NEXT O
840 RETURN
900 H PLOT 10,60 TO 10,5 TO M+1,5 TO M+1,60
910 RETURN
950 D=M-10
960 H PLOT 10,130 TO 10,130-D/4 TO M-1,130-D/4

```

```

 TO M-1,130 TO 10,130+D/4 TO 10,130
970 RETURN
1000 H PLOT 10,100 TO 10,160 TO M,130 TO 10,100
1050 RETURN
1100 H PLOT 10,120 TO 10,150 TO M,150 TO M,120
 TO 10,120
1150 RETURN

```

## 磁带虚拟磁盘的文件管理

中国农科院蜜蜂研究所(100093) 胡发新

中华学习机已有相当大的数量进入家庭,但购买磁盘驱动器的还不多,一般家庭还是用磁带机做外存、取信息。一盘磁带可存、取很多文件,时间一久,有的文件不易找到,很是费劲。作者为了解决这问题,编制了一种文件管理的方法,现介绍如下供参考:

首先根据《电子与电脑》91年第7期“为无驱动器系统设置虚拟磁盘”一文的程序,建立一个虚拟磁盘。然后在磁带的每一面开头一段磁带,以磁带机的计数器记数,空出000—060圈,建立虚拟磁盘文件总目录区。在060圈以后作为每个虚拟磁盘的文件存、取区(见程序1)。在每一个虚拟磁盘内最后几个扇区,作为本盘的文件目录扇区(见程序2),其他作为文件存、取扇区。

在建立每一个虚拟磁盘目录时,要从磁带的60圈开始,有计划地把每个文件键入或读入虚拟磁盘。例如要在A号盘装入以下文件,A1光子飞船,A2,保卫炮台,A3,电子琴……。先把文件键入或读入内存,再用SAVE A1↙,SAVE A2↙,SAVE A3↙……装入A号盘,一直装到接近60扇区时。根据装入的文件多少,最后留出2—4个扇区,把已装入盘内的所有文件用程序2,以文件目录形式装入最后几个扇区。文件目录的内容,第一项是文件名的代号,第二项是文件名,第三项是文件的简单说明或简单的标注。如A号盘的第二个文件,写成A2,保卫炮台(JIKL)。括号内是使用时的操作键符号。文件目录本身定名代号为本盘的盘号,如A号盘就是A,查阅A号盘文件目录时,键入LOAD A↙即可。

每个虚拟磁盘按以上方法装入文件,把所有的文件装入完毕后,再把磁带倒回000圈处,把每个虚拟磁盘的所有文件目录,用程序1,以文件目录形式装入000圈以后的磁带中,形成一个虚拟磁盘的文件总目录。其中内容是记录每个虚拟磁盘的盘号,占用磁带的圈数(磁带机上计数器的数值)和本盘的文件名代号及文件名。

在使用时,用LOAD提出虚拟磁盘的文件总目录,查找需要的文件,如需要保卫炮台这个文件,找到它在A号盘,记下A号盘所在磁带中的圈数。从起始圈数开始,把A号盘信息读入内存,用LOAD A2↙提出保卫炮台的程序。如忘记其操作方法或忘记了本盘

的其他文件名的代号,可键入LOAD A↙提出本号盘的文件目录查阅。使用起来很是方便。

另外在程序1中,还编入字符传递速度。如虚拟磁盘文件总目录显示速度太快,看不清楚时,可放慢字符显示速度。

无驱动器的学习机用户,如感兴趣,不妨采用本文件管理方法,试一试,定会带来不少方便。

程序1

```

10 HGR2
20 PRINT TAB(7)“虚拟磁盘文件总目录”
30 PRINT TAB(10)“(002—060圈)”
40 PRINT “A号盘文件目录□□060—XXX圈”
50 PRINT “文件名:A1,XXX□A2,XXX□A3,XXX
 ……A1A号盘文件目录”
60 PRINT:PRINT“B号盘文件目录□□XXX—XXX
 圈”
70 PRINT “文件名:B1,XXX□B2XXX□B3,XXX……
 B1B号盘文件目录”

```

(注:其他盘以此类推)

```

500 PRINT:PRINT“您选择字符传递速度吗?(Y/N);”
 GET S$
510 IF S$ = “N” THEN 530
520 PRINT:GOTO 550
530 INPUT “请键入(0—255)之间的数值!”;T
540 SPEED=T
550 PRINT:PRINT“1.请在计数器上找到虚拟磁盘起
 始圈数!”
560 PRINT:PRINT“2.请复位后,键入3FFD*8054RN
 3FFDG把程序装入虚拟磁盘!”
570 PRINT:PRINT“3.不需要本程序时,请用NEW清
 除”
580 END

```

注:3FFD\*8054RN 3FFDG命令是CEC—I中华学习机,把程序读入内存并装入虚拟磁盘用。其他机型需修改。

程序2

```

10 HGR2
20 PRINT TAB(9)“X号盘文件目录”
30 PRINT:PRINT“文件名:X1,XXX□X2,XXX□X3.
 XXX……”
40 PRINT “请您选择需要的文件!”
50 END

```

# Apple— II 机音乐功能的扩充

盐城师专 陈建明

《Apple— II 机 BASIC 语言程序设计》中介绍了九个音阶和五个音长代码,只能编写极其简单的音乐程序。为了扩充其功能,笔者重编了机器发音子程序,调试出七个音调的147种音阶代码,并通过精心设计 BASIC 程序,使乐谱输入简单,演奏时只要改变初值,就能改变音调和速度。

## 一、基本原理

### 1. 音长的控制和变速演奏

在 Apple— II 机中发音是通过通过对 \$C030 单元的操作来实现发音的。对这个单元操作的时间越长,发出的声音也越长。原发音子程序,音长由一个八位的二进制计数器来控制,且各音长值固定不变,因而,发出的最长音只有一拍半,演奏速度不能改变。如果将发音子程序中控制音长的计数器改为十六位,并且使音长代码与发音时间呈线性关系,则演奏时,只要给定一拍音符的代码值,其它音长的代码就可直接算出,最长音可达3分钟。这样,既简化了乐谱的输入,又可以方便地实现变速演奏。

### (2) 音阶控制与变调演奏

原发音子程序通过一个八位二进制计数器来控制对 \$C030 单元的触发频率,以达到音阶控制。八位二进制代码值已基本满足一般歌曲的演奏,只是原来给定的音阶代码值太少。笔者通过调试,得出了七个调门从1到7三个八度音程的147个音阶代码(也可以根据发音子程序的执行周期和各音阶所规定的频率,严格算出各音阶的控制代码)。演奏时,首先将音阶代码输入到一个两维数组,其中第一维下标为音调代码,第二维下标为乐谱的音高代码,通过改变第一维的下标值,就可以改变演奏的调门。

## 二程序及其说明

### 1程序1:

0303—	AD	30	C0	88	D0		
0308—	0D	CE	01	03	D0	08	CE 02
0310—	03	F0	0C	CE	01	03	CA D0
0318—	ED	AE	00	03	4C	03	03 60
0320—	FF	E2	CB	BF	AA	98	88 80
0328—	71	65	5E	54	4B	43	3F 38
0330—	32	2F	29	25	21	E2	CB B6
0338—	AA	98	88	79	71	65	5A 55
0340—	4B	43	3C	38	32	2D	2A 25
0348—	21	1D	CB	B6	A1	98	88 79
0350—	6C	65	5A	50	4B	43	3C 36
0358—	32	2D	27	25	21	1D	1A B6
0360—	A1	91	88	79	6C	60	5A 50
0368—	48	43	3C	36	30	2D	27 23
0370—	21	1D	1A	17	AA	98	88 80
0378—	71	65	5A	54	4B	43	3F 38

0380—	32	2C	29	25	21	1F	1C 19
0388—	16	98	88	79	71	65	5A 50
0390—	4B	43	3C	38	32	2C	27 25
0398—	21	1D	1B	18	15	13	88 78
03A0—	6B	65	5A	50	48	43	3C 36
03A8—	32	2D	27	23	21	1D	1A 19
03B0—	16	13	11				

### 程序2:

```

100 INPUT "F=C#---B$?";A$:INPUT"V=";V
110 L=800;N=27;F=ASC(A$)-66:IF F<1 THEN F=F+7
120 DIM A(7,27),X(3,N)
130 FOR K=1 TO 7:A(K,0)=1
140 FOR I=0 TO 2:FOR J=1 TO 7
150 A(K,I*10+J)=PEEK(L):L=L+1
160 NEXT J,I,K
210 FOR I=1 TO N
220 READ X(1,I),X(2,I)=V*X:X(3,I)=1
230 IF X<0 THEN X=-X:X(2,I)=(V*(1+X-INT(X)))/(2*INT(X)):GOTO 250
240 IF X(2,I)>255 THEN X(3,I)=1+INT(X(2,I)/255):X(2,I)=X(2,I)-255*(X(3,I)-1
250 NEXT
300 FOR I=1 TO N
310 POKE 768,A(F,X(1,I)):POKE 769,X(2,I):POKE 770,X(3,I)
320 CALL 771:NEXT
330 END
500 DATA 5,1,11,1,13,1.5,12,-1,12,-2,13,-2,12,-2,11,4,12,1,11,2,6,-1,5,-1,5,4,5,1,11,1,13,1,15,1,15,2,14,1,0,-1,14,1,6,1,14,1.5,14,-1,13,-1,12,-1,12,3

```

### 2程序说明

(1) 程序1中 \$303—\$3B0 为发音子程序, \$320—\$3B2 为7个音调的各音阶代码值。该程序不能浮动,并规定 \$300 单元存放音阶代码, \$301 和 302 单元分别存放音长代码的低位和高位,其中 \$302 的最小值为1。

(2) 程序2必须和程序1配合使用。其中100句为输入演奏的音调值和速度值,当速度值在120—180时为一般速度。110句中的 N 值为乐谱的音符数。140句—160句为读入各音阶代码。210句—250句为读入乐谱代码,并把其中的音高值转化为音长代码。

(3) 乐谱的输入规则。乐谱的每个音符包含音高和音长两个要素。本程序对乐谱的输入,要求按照表一、表二的规则进行转换。事实上,这些规则非常好记,完全可以看着乐谱直接输入。

程序2中500句的数据即为下面一段乐谱的代码。(略) (下转第22页)

# POSITION 命令新用

湖北随州市洪山医院设备科(441318) 周 进

在 Apple DOS 操作系统中,POSITION 命令是用来对顺序文件执行定位的操作,它可以把文件位置指针从当前位置移到下面若干个域的起始位置,在随机文件中一般没有用到 POSITION 命令.根据随机文件与顺序文件的结构特点,笔者通过实践将随机文件中的一条记录看成一个顺序文件,当读写这个随机文件的一条记录时,利用 POSITION 命令读写这条记录中的任一字段,结果获得成功.POSITION 命令的这种用法,能为我们组织随机文件的工作带来很大方便.

为使说明更清楚,请阅读下面两段程序,首先用程序 PROGRAM 1 建立一个随机文件,这里有五条记录,每条记录设立了八个字段,以 FILENAME 为文件名存盘.PROGRAM 2 是巧用 POSITION 命令,来检索出随机文件 FILENAME 中每条记录的第四个字段的程序,均已在 CEC-I 机上通过.

```
5 REM PROGRAM 1
```

```
10 D$ = CHR$(13) + CHR$(4)
```

```
15 PRINT D$;"OPEN FILENAME,L30"
20 FOR I=1 TO 5
25 FOR J=1 TO 8: READ A(J): NEXT J
30 PRINT D$;"WRITE FILENAME,R";I
35 FOR J = 1 TO 8
40 PRINT A(J);A$;
45 NEXT J
50 NEXT I
55 PRINT D$;"CLOSE FILENAME"
60 DATA 11,12,13,14,15,16,17,18
70 DATA 21,22,23,24,25,26,27,28
75 DATA 31,32,33,34,35,36,37,38
80 DATA 41,42,43,44,45,46,47,48
85 DATA 51,52,53,54,55,56,57,58
```

```
5 REM PROGRAM 2
```

```
10 D$ = CHR$(13) + CHR$(4)
```

```
15 PRINT D$;"OPEN FILENAME,L30"
```

```
20 FOR I = 1 TO 5
```

```
30 PRINT D$;"READ FILENAME,R";I
```

```
40 PRINT D$;"POSITION FILENAME,R3"
```

```
45 PRINT D$;"READ FILENAME"
```

```
46 INPUT A
```

```
47 PRINT A
```

```
50 NEXT I
```

```
55 PRINT D$;"CLOSE FILENAME"
```

# CEC-I 与1724打印机配接图形硬拷贝程序

马鞍山六中电教室 张益贵

中华学习机及其兼容机通常都配接9针打印机,因绝大多数图形软件只支持9针打印机,所以用24针打印机作外设时,需开发相应的驱动程序才能正常使用.如下所附的程序,可实现 CEC-I 与1724打印机配接时高分图形屏幕硬拷贝.为了便于无间隙的多幅连打,设计成将图形旋转90°打印.

将附后的机器语言程序键入,取名存盘备用.使用时,调入图形硬拷贝程序到内存 \$8000 首地址单元,将需打印的图形调入高分第一页 \$2000 首地址单元,在监控状态下:8000G $\checkmark$ 即可执行打印;在 BASIC 状态下:CALL 32768 $\checkmark$ 即可执行打印.若需连打,再调入另一幅图形,然后再执行打印.亦可自编一小程序,做到多幅连打,一气呵成.

修改程序的某些单元,可变通使用,举例如下:

1. 图形的正相与反相

在监控状态下,修改 \$804D 单元,可变更所打印图形的正反相:

正相图形 \$804D:EA EA $\checkmark$

反相图形 \$804D:49 7F $\checkmark$

2. 高分第几页的选择

需打印的图形可调入高分第一页,也可调入高分

第二页首地址 \$4000,或第三页(首地址 \$6000).选择拷贝第几页.只需在监控状态下修改 \$8022 单元即可.

拷贝第一页 8022: 20 $\checkmark$

拷贝第二页 8022: 40 $\checkmark$

拷贝第三页 8022: 60 $\checkmark$

3. 拷贝袖珍图形

将需打印的图形印成一寸照片大小,以便于制作图集保留.在监控状态下修改以下7个地址单元即可(前述正反相选择和高分页的选择仍有效).

8009: 05 N 8013:47 N 8052:A9 00 20 6F 80 $\checkmark$

4. 拷贝袖珍变形图

在袖珍图形的模式下,修改 \$8013 单元,可打印拉长一倍的变形图,使横构图的袖珍图形变为夸张的纵构图(正反相、选页仍有效).

\$8013: 3D $\checkmark$

未修改的当前程序,为正常、反相的图形,选第一页.读者可反汇编列出清单根据需要修改.亦可自编一 BASIC 图形处理程序,用 POKE 置数进行多项选择等,更为简明直观.

在图形打印前,1724打印机的配置开关(SW1)应

置⑦为 ON(拨向右)

8000-	20	6D	80	A9	4A	20	6F	80
8008-	A9	0B	20	6F	80	A0	27	20
8010-	6D	80	A9	3D	20	6F	80	A9
8018-	00	20	6F	80	A9	BF	20	6F
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	49	7F	20
8050-	6F	80	20	6F	80	A9	00	20
8058-	6F	80	E8	E0	BF	D0	C8	A9
8060-	00	20	6F	80	A9	0A	20	6F
8068-	80	88	10	A3	60	A9	1B	2C
8070-	C1	C1	30	FB	8D	90	C0	60

```
FB2B-D0 F8 BNE $FB25
FB2D-88 DEY
FB2E-60 RTS
```

这是一个单循环计数程序。运行时不断地查询内存 \$C064+X(X=0~3) 的内容。如果内存的高位为 1, 就增加一个记数; 如果内存高位为 0, 就退出循环。由于单重循环的最大记数为 255, 这就制约了 GAME 的模拟入口所能连接的最大电阻仅为 125kΩ 左右。对于大于 125kΩ 的电阻, 此单重循环就无法计数了。这给需用大于 125kΩ 电阻进行的实验操作带来不便。

解决的办法是改写这一段监控程序, 用双重循环程序代替单循环程序。改写的程序如下

```
0300- D8 A9 00 85 09 85 FA 85
0308- FB 85 FC 20 37 03 18 A5
0310- 07 65 FA 85 FA A5 08 65
0318- FB 85 FB A9 00 65 FC 85
0320- FC E6 09 A5 09 C9 20 D0
0328- E2 A2 00 46 FC 66 FB 66
0330- FA E8 E0 05 D0 F5 60 A6
0338- 06 AD 70 C0 A0 00 84 08
0340- BD 64 C0 10 09 C8 D0 F8
0348- E6 08 D0 F4 C6 08 84 07
0350- 60
```

因为双重循环的计数是单循环的 256 倍, 所以改写监控程序后 GAME 模拟入口能连接的电阻最大可达 30MΩ 左右。

在进行接口扩展操作时, 先将选择的模拟通道号存入 \$06 单元, 然后再调用上述双重循环程序, 就可以在 \$FB、\$FA 两单元中得到记数。

## 游戏接口 PDL 的扩展

武警长沙指挥学校训练处(410125) 张建群

苹果机的游戏接口的模拟输入口(PDL)监控程序如下:

```
FB1E-AD 70 C0 LDA $C070
FB21-A0 00 LDY # $00
FB23-EA NOP
FB24-EA NOP
FB25-BD 64 C0 LDA $C064,X
FB28-10 04 BPL $FB2E
FB2A-C8 INY
```

## SUPER DOS 简介

北京职工技术师范学院 张志

现在苹果机、中华机流行一种磁盘操作系统——SUPER DOS, 它具有许多优于 Apple DOS 的功能。并且该系统主盘上还提供了一个名为《BASIC》的文件, 用 & 指令扩展了几条 Applesoft 语句, 现介绍该系统的 1.6 版的功能如下。

### 一、磁盘操作系统

①本系统要求主机内存至少为 64K, 系统启动后, 自动将 DOS 搬入 16K RAM 卡, 把原来 Apple DOS 占据的 \$9600~\$BFFF 这段地址空间腾出来给用户程序使用。

②本系统具有用“\*”作为通配符功能, 各种含对文件名操作的命令都可利用此功能。详细用法见 Apple DOS 主盘 FID 程序的使用说明, 本系统只是将通配符改为“\*”了。

③本系统取消了原 Apple DOS 的 FP、INT 命令, 保留的 CHAIN 命令只可用于 A 类文件间链接。

④系统的 OPEN、APPEND、WRITE、READ、EXEC、CLOSE、DELETE、VERIFY、LOCK、UNLOCK、BRUN、BSAVE、SAVE、BLOAD、INIT、与 Apple DOS 用法相同。

⑤HELP 命令: 在文本状态下提示全部 SUPER DOS 命令。

⑥RUN、LOAD 命令: 扩展为对 A、B、T 类文件均可操作。

⑦TYPE 命令: 以 ASCII 码形式显示文件内容。格式: TYPE 文件名

⑧NTR 命令: 指定当前磁盘为普通格式磁盘。格式: NTR

⑨HTR 命令:指定当前磁盘为半轨格式磁盘。

格式:HTR

在此命令下,格式化一张磁盘,普通拷贝程序将无法拷贝这种格式的磁盘。只有启动 SUPER DOS 后,敲入 HTR 命令才能读写这种半轨格式磁盘,因而起到了加密的作用。

⑩CATALOG(或 DIR):为列磁盘目录命令,同时显示自由扇区数。

⑪ADR 和 LEN 指令:用来显示刚装入的二进制文件的首地址和长度

格式:ADR

LEN

⑫PRG 命令:显示磁盘的自由扇区数。

格式:PRG

⑬BASIC 命令:设定 BASIC 程序区始存地址

格式:BASIC An(n 为地址)

⑭MEG 命令:把当前内存中的 BASIC 程序与盘上的 BASIC 程序合并,行号均不变。

格式:MEG 文件名

⑮UNDEL 命令:恢复被误删的文件。

格式:UNDEL 文件名

⑯FLB 命令:设置缓冲区首地址,最高可设置为 \$ 95CA

格式:FLB 地址

⑰该系统还有自动执行文件功能。当键入一串既非 SUPER DOS 的命令保留字,也不是 Applesoft 命令的保留字的字符串时,系统会把该字符串当成文件名并运行它。

二、扩展 BASIC 命令

先运行系统盘上的《BASIC》文件,通过 & 命令提供13条扩展 BASIC 命令:

①&EDIT:编辑指定命令行。

格式:&EDIT 行号

②&UNNEW:追回刚被 NEW 命令清掉的 BA-

SIC 程序

③&INPUT:扩展的 INPUT 命令,可在输入的变量中加入表达式,其格式与 INPUT 命令相同。

例:10 &INPUT A

RUN

?SIN(5) \* 6 - 4 + TAN(3)

④&GOTO 和 &GOSUB:扩展 GOTO 和 GOSUB 命令,可用表达式表示行号,

格式:&GOTO 表达式

&GOSUB 表达式

⑤&RESTORE:将指定语句作为 DATA 语句的最初读取行

格式:&RESTORE 行号

⑥&REN:重整行号命令。用法与 TOOL KIT 中的 APA 程序所提供的 &RENUMBER 命令用法相同。

⑦&AUTO 指令:自动行号命令,用法与 APA 程序中的 &AUTO 命令相同。

⑧&ALIST 命令:

格式:&ALIST 行号(相当于 POKE 33,33:LIST 行号)

&ALIST(相当于 POKE 33,33:LIST)

⑨&BLIST:比 &ALIST 更紧凑的一种列程序方法。格式与 &ALIST 相同。

⑩&FIND:寻找并列含有指定字符内容的 BASIC 程序行。

格式:&FIND "字符串"

&FIND 字符串变量。

⑪&SHOW:显示控制字符

格式:&SHOW

⑫&R:取消 &SHOW 命令

格式:&R

⑬&POINT:指定要读取的 DATA 数据

格式:&POINT 表达式

(上接第19页)

表一

音符	1	2	...	6	7	1	2	...	6	7	1	2	...	6	7
音高值	1	2	...	6	7	11	12	...	16	17	21	22	...	26	27

表二

音符	X	=	X.	X	X.	X	X.	X-	X-
音长值	-2	-2.5	-1	-1.5	1	1.5	2	2.5	...

(上接第47页)

从软盘启动后,无法进入硬盘,即使使用低级格式化命令也无法执行;开机从硬盘启动进入 ROM-BASIC 状态等等硬盘故障,原因可能是多方面的,但系统设备配置信息错误是一条不可忽视的原因。CPU 为 286、386 等微机的设备配置情况一般由高级诊断程序的“SETUP”功能设置(长城 286BH 按 Ctrl + ALT + Esc 键设置)。存储在系统主板的 CMOS RAM 中,而 RAM 由系统主板上的后备电池供电,即关闭主机电源,该信息一般不会丢失。但当机器受到搬运等强烈震动,后备电池松动而接触不良;或者机器较长一段时间不使用,后备电池不能及时充电,能量不足,该 RAM 中的信息就会改变或丢失。这时开机系统配置情况与设备配置信息不符,BIOS 检测失效,就可能产生上述故障现象。

# 第十讲 监控子程序的调用(二)

南京大学大气科学系(210008)朱国江

⑨显示一个方框

如在屏幕上显示如下方框,要求充满整个屏幕。

```
* = * = * = * = * = *
=
*
=
*
=
*
=
*
=
* = * = * = * = *
```

这个问题看似简单,但实际编程还较复杂,先给出设计好的程序如下:

程序:10.24

```
0300- 20 58 FC 58 JSR $FC58
0303- A0 00 LDY # $00
0305- A9 AA LDA # $AA
0307- 20 ED FD JSR $FDED
030A- A9 BD LDA # $BD
030C- 20 ED FD JSR $FDED
030F- C8 INY
0310- C0 13 CPY # $13
0312- D0 F1 BNE $0305
0314- A9 AA LDA # $AA
0316- 20 ED FD JSR $FDED
0319- A0 00 LDY # $00
031B- 20 5C 03 JSR $035C
031E- C8 INY
031F- C0 0A CPY # $0A
0321- D0 F8 BNE $031B
0323- 20 62 FC JSR $FC62
0326- A9 BD LDA # $BD
0328- 20 ED FD JSR $FDED
032B- A2 25 LDX # $25
032D- 20 4A F9 JSR $F94A
0330- A9 BD LDA # $BD
0332- 20 ED FD JSR $FDED
0335- 20 62 FC JSR $FC62
0338- A0 00 LDY # $00
033A- A9 AA LDA # $AA
033C- 20 ED FD JSR $FDED
033F- A9 BD LDA # $BD
0341- 20 ED FD JSR $FDED
0344- C8 INY
0345- C0 13 CPY # $13
0347- D0 F1 BNE $033A
0349- A9 AA LDA # $AA
034B- 20 ED FD JSR $FDED
```

水平显示  
19对\* =,  
再补上一个\*

垂直显示10组  
\*,  
中间空37格

下称一行,显示一个\*,  
中间空37格,  
再示一个\*,  
再下移一行

同 \$300 ~ \$318

```
034E- A2 50 LDX # $50
0350- A9 FF LDA # $FF
0352- 20 A8 FC JSR $FCA8
0355- CA DEX
0356- D0 F8 BNE $0350
0358- 20 58 FC JSR $FC58
035B- 60 RTS
035C- 20 62 FC JSR $FC62
035F- A9 BD LDA # $BD
0361- 20 ED FD JSR $FDED
0364- A2 25 LDX # $25
0366- 20 4A F9 JSR $F94A
0369- A9 BD LDA # $BD
036B- 20 ED FD JSR $FDED
036E- 20 62 FC JSR $FC62
0371- A9 AA LDA # $AA
0373- 20 ED FD JSR $FDED
0376- A2 25 LDX # $25
0378- 20 4A F9 JSR $F94A
037B- A9 AA LDA # $AA
037D- 20 ED FD JSR $FDED
0380- 60 RTS
```

调用80次延时  
子程序延时  
清屏,结束

垂直显示一组  
\*,  
中间空37格

程序10.24共用了5个监控中机器语言子程序:

- \$FC58:清屏
- \$FDED:输出一个字符
- \$FC62:回车换行
- \$FCA8:延时
- \$F94A:屏幕上输出若干个空格,空格数在x寄存器中设定

程序10.24主要用了调用机器语言子程序的技巧,这包括两方面的内容,一是调用监控子程序,实现诸如清洗屏幕、输出字符、控制延时、光标下移、输出空格等功能;二是调用自编的机器语言子程序,如\$035C-\$0380单元中存放的子程序,它实现一组字符的显示,而用\$0319-\$0322程序段对其实实现次数控制。

(10)显示一个题头

在边框里放上一些说明性文字(或资料),就设计成了题头。边框设计可以用程序10.24,或者根据其设计思想,绘成其它图案。问题的关键是如何设计一个存有文字字符的数据表,并把它放在边框内的适当位置。

为了直接引用程序10.24设计的边框,我们可以设想再编一个独立的存放资料表的程序,为简单计,假设资料表就是一行,即PRESS RETURN TO CONTINUE,见程序10.25:

```
1000- A0 00 LDY # $00
```

```

1002- B9 0E 10 LDA $100E,Y
1005- 20 ED FD JSR $FDED
1008- C8 INY
1009- C0 26 CPY # $26
100B- D0 F5 BNE $1002
100D- 60 RTS
100E- A0 A0
1010- A0 A0 A0 A0 50 52 45 53
1018- 53 60 52 45 54 55 52 4E
1020- 60 54 4F 60 43 4F 4E 54
1028- 49 4E 55 45 60 61 A0 A0
1030- A0 A0 A0 A0

```

程序10.25的设计思想和结构安排均和程序10.16相类似,只是用的指令不尽相同,这里不再重述。从\$100E开始到\$1033止,存放的是PRESS RETURN TO CONTINUE!对应的闪烁显示方式的ASCII字符。运行时在监控下用1000G $\swarrow$ ,在BASIC状态下用CALL 4096 $\swarrow$ 。

有了边框和数据表,就可以设计一个题头,这实际上就是将程序10.24和程序10.25合起来,并修改适当的控制段和修改个别字节的内容,见程序10.26。

```

0300- 20 58 FC JSR $FC58
0303- A0 00 LDY # $00
0305- A9 AA LDA # $AA
0307- 20 DE FD JSR $FDED
030A- A9 BD LDA # $BD
030C- 20 ED FD JSR $FDED
030F- C8 INY
0310- C0 13 CPY # $13
0312- D0 F1 BNE $0305
0314- A9 AA LDA # $AA
0316- 20 ED FD JSR $FDED
0319- A0 00 LDY # $00
031B- 20 5C 03 JSR $035C
031E- C8 INY
031F- C0 0A CPY # $0A
0321- D0 F8 BNE $031B
0323- 20 62 FC JSR $FC62
0326- A9 BD LDA # $BD
0328- 20 ED FD JSR $FDED
032B- A2 25 LDX # $25
032D- 20 4A F9 JSR $F94A
0330- A9 BD LDA # $BD
0332- 20 ED FD JSR $FDED
0335- 20 62 FC JSR $FC62
0338- A0 00 LDY # $00
033A- A9 AA LDA # $AA
033C- 20 ED FD JSR $FDED
033F- A9 BD LDA # $BD
0341- 20 ED FD JSR $FDED
0344- C8 INY
0345- C0 13 CPY # $13
0347- D0 F1 BNE $033A
0349- A9 AA LDA # $AA
034B- 20 ED FD JSR $FDED

```

```

034E- A2 20 LDX # $20
0350- A9 FF LDA # $FF
0352- 20 A8 FC JSR $FCA8
0355- CA DEX
0356- D0 F8 BNE $0350
0358- 20 34 10 JSR $1034
035B- 60 RTS
035C- 20 62 FC JSR $FC62
035F- A9 BD LDA # $BD
0361- 20 ED FD JSR $FDED
0364- A2 25 LDX # $25
0366- 20 4A F9 JSR $F94A
0369- A9 BD LDA # $BD
036B- 20 ED FD JSR $FDED
036E- 20 62 FC JSR $FC62
0371- A9 AA LDA # $AA
0373- 20 ED FD JSR $FDED
0376- A2 25 LDX # $25
1000- A0 00 LDY # $00
1002- B9 0E 10 LDA $100E,Y
1005- 20 ED FD JSR $FDED
1008- C8 INY
1009- C0 26 CPY # $26
100B- D0 F5 BNE $1002
100D- 60 RTS
100E- AA A0
1010- A0 A0 A0 A0 50 52 45 53
1018- 53 60 52 45 54 55 52 4E
1020- 60 54 4F 60 43 4F 4E 54
1028- 49 4E 55 45 60 61 A0 A0
1030- A0 A0 A0 A0
1034- 20 F4 FB JSR $FBF4
1037- 20 ED FD JSR $FDED
103A- 20 F4 FB JSR $FBF4
103D- 20 1A FC JSR $FC1A
1040- 20 1A FC JSR $FC1A
1043- 20 1A FC JSR $FC1A
1046- 20 00 10 JSR $1000
1049- 20 66 FC JSR $FC66
104C- 60 RTS

```

\$0300—\$0380:为主要程序段,显示一个边框

\$1000—\$1033:显示文字资料,其中\$100E~\$1033为对应文字资料的ASCII码字符

\$1034—\$104C:为使文字资料放置在边框中的适当位置而设置的控制程序,其中\$FBF4是光标右移一格的子程序,\$FC1A使光标上移一行。

#### (11)汉字字符的显示

在中华学习机(Apple-I及其兼容机不具备)内,有一个重要的子程序CSWA,它的功能是完成汉字或ASCII字符的输出,其入口地址为\$C32B。

对CSWA子程序的调用,必须首先要求在累加器A中存放汉字或字符的显示码以及显示控制命令码。而对于汉字则一定要用三字节的中华学习机内码,同时需要分三次调用才能输出一个汉字。下面提供一个



实例,见程序10.27

```

1000-A9 0D LD # $D3
1002-20 2B C3 JSR $C32B
1005-A2 00 LDX # $00
1007-BD 18 10 LDA $1018,X
100A- 48 PHA
100B- 20 2B C3 JSR $C32B
100E- E8 INX
100F- 68 PLA
1010- 10 F5 BPL $1007
1012- A9 0D LDA # $0D
1014- 20 2B C3 JSR $C32B
1017- 60 RTS

```

```

1018-7F 55 4F 7F 39 27 7F 50
1020- 24 7F 4E 2E 7F 39 79 20
1028- 43 4F 4D 50 55 54 45 D2

```

程序10.27的运行方法比较特殊,因为程序中有显示汉字的安排,所以必须在中文状态下,并且处于BASIC状态才能运行,即]CALL 4096

屏幕显示:中华学习机 COMPUTER

(12)文本状态窗口控制显示

中华学习机在文本状态下可以显示各种ASCII字符,正常的显示屏幕被设定为24行40列。为了能在屏幕的任意位置上显示字符,可以采用“开窗口”的方法,所谓屏幕“开窗口”,就是在屏幕上任意设定一块显示区,而在这个区域以外的屏幕不发生变化。窗口的设置由用户向四个特殊零页单元置数确定:

\$20 单元存放窗口左极限列数(\$0~\$27)

\$21 单元存放窗口的宽度(\$1~\$28)

\$22 单元存放窗口顶行行数(\$0~\$18)

\$23 单元存放窗口底行行数(\$0~\$18)

正常情况下\$20~\$23四个单元中的值分别为\$0,\$28,\$0,\$18,在BASIC状态下可以用POKE命令设定参数,例如在\$20单元中放\$0A:

POKE 32,10

而用机器语言设定时,则可用:

A9 0A LDA # \$0A

85 20 STA \$20

例如要在屏幕中央反相显示“6502CPU”可用程序10.28:

```

0300- 20 58 FC JSR $FC58
0303- A2 00 LDX # #00
0305- BD 11 03 LDA $0311,X
0308- 20 ED FD JSR $FDED
030B- EB INX
030C- E0 07 CPX # #07
030E- D0 F5 BNE $0305
0310- 60 RTS
0311- 76 75 ROR $75,X
0313- 70 72 BVS $0387
0315- 43 ???
0316- 50 55 BVC $036D

```

```

0318- 20 58 FC JSR $FC58
031B- A9 13
031D- 85 20 STA $20
031F- A9 09 LDA # $09
0321- 85 22 STA $22
0323- 20 00 03 JSR $0300
0326- 60 RTS

```

程序10.28中\$0300~\$0317单元和程序10.16完全一样,反相显示“6502CPU”七个字符。\$031B~\$031E是设定窗口左极限,其值为\$13(取屏幕左、右边界的中间值), \$031F~0322是设定窗顶行数,其值为\$09(取屏幕上、下边界的中间值)。

(13)数据块搬家

将\$6000-\$60FF的内容搬至\$7000-\$70FF中去。

程序10.29

```

300- A0 00 LDY # $00
302- A9 00 LDA # $00
304- 85 3C STA $3C
306- A9 60 LDA # $60
308- 85 3D STA $3D
30A- A9 FF LDA # $FF
30C- 85 3E STA $3E
30E- A9 60 LDA # $60
310- 85 3F STA $3F
312- A9 00 LDA # $00
314- 85 42 STA $42
316- A9 70 LDA # $70
318- 85 43 STA $43
31A- 20 FC FE JSR $FE2C
31D- 60 RTS

```

这里利用了监控中搬家子程序,只需将源数据首地址和末地址放入\$3C-\$3F中,而把目的首地址放在\$42-\$43中,调用\$FE2C子程序一次搬完。在调用前必须设置LDY#00。值得提出的是,在监控状态下移动一段存储器单元的内容就是用的搬家程序\$FE2C,即所谓执行监控状态下的M命令。即:

\* <新地址><原首地址>·<原末地址>M

关于调用监控子程序的实例,我们就介绍到这里。必须指出的是监控子程序远远不只是上面的内容,还有许多子程序,我们没有列出,有兴趣的读者可查阅有关资料,另外,本讲座的其它章节也有一些调用监控子程序的程序,请注意对照学习。

(上接第27页)位,模拟信号为伪差分输入方式,电位器VR2用于调节INLO电位的。3. 控制电路:由74LS00,插座W1、W2、W3电阻R1、R2及电容C1等组成。

其中插座W1用于选择转换结束方式,可选择中断方式或者查询方式;插座W2用于转换器启动方式选择,可用MOVX指令启动,也可用口输出指令启动。除上述的控制转换之外,还可以选择连续转换方式。插座W3用于输出方式选择。

4. 负电源产生电路:由ICL7660及电容C8、C9、C10组成,产生一个-5伏的电源。



# BJS-51单片机实验系统(续) 模/数、数/模转换实验

北京工业大学 张俊谟

众所周知,在过程控制和智能仪器仪表中通常由微型机进行实时控制和实时数据处理。计算机加工的信息总是数字量,而被控制或被测对象的有关参量却往往又是一些连续变化的模拟量。因此,把被测参量送入微机之前必须进行模拟量到数字量的转换;反过来,微机处理后的数字量也必须经过数字量到模拟量的转换才能被执行机构等接收。

由此可见,A/D、D/A转换是微机与客观世界联系的桥梁,是微机接口的重要组成部分。A/D、D/A转换与单片机接口的复杂性,包含以下五个方面:

- 数据输入、输出与总片相应引脚的连接(包括数字接口和模拟接口)
- 转换启动方式的选择
- 转换结束信号(或状态信号)的处理
- 参考电压的获得及连接
- 时钟的产生和连接

A/D、D/A转换软件的编制也有其独特性,其中包括中断、查询、延时等等访问方式。

上述这些方面既是A/D、D/A接口电路设计必须解决的问题,又是学生学习A/D、D/A转换技术中的难点所在。因此在设计A/D、D/A转换实验时,必须让学生对这些问题有足够的了解和较为充分的训练。

为此,BJS-51单片机教学实验系统中,我们设计了有关A/D、D/A转换实验共5个。限于篇幅,只能着重介绍12位A/D转换实验和8位D/A转换实验。本期主要介绍双积分式12位A/D转换实验。

为了让设计的实验电路有充分的选择余地,我们选用廉价而高精度的双积分式A/D转换芯片——ICL7109。下面分别介绍A/D转换芯片、实验电路设计以及实验软件的编制等。

## 一、双积分式12位A/D转换芯片——ICL7109

### 1. 总片特点:

- 高精度(精确到1/4096)
- 低噪声(典型值为 $15\mu\text{Vpp}$ )
- 低漂移(小于 $1\mu\text{V}/^\circ\text{C}$ )
- 高输入阻抗(典型值为 $10^{12}\Omega$ ,典型输入电流 $1\text{pA}$ )

### • 与TTL兼容,三态控制输出

- 具有通用控制信号,可用来监视和控制转换时间
- 片内有振荡器,只须外接晶体或RC器件

### 2. 芯片的结构特点:

ICL7109的整体电路由模拟电路和数字电路两部分所组成,这里不详细叙述其结构。

### 3. 芯片的外部使用特点:

ICL7109芯片的外部引脚和外部连接状态如图1所示。

(1)电源供给:ICL7109为双电源 $\pm 5\text{V}$

(2)参考电压供给:ICL7109有一个良好的片内参考电压源,由 $\text{REF}_{\text{OUT}}$ (29脚)输出,可以用一个电位器分压获得一个合适的基准电压。

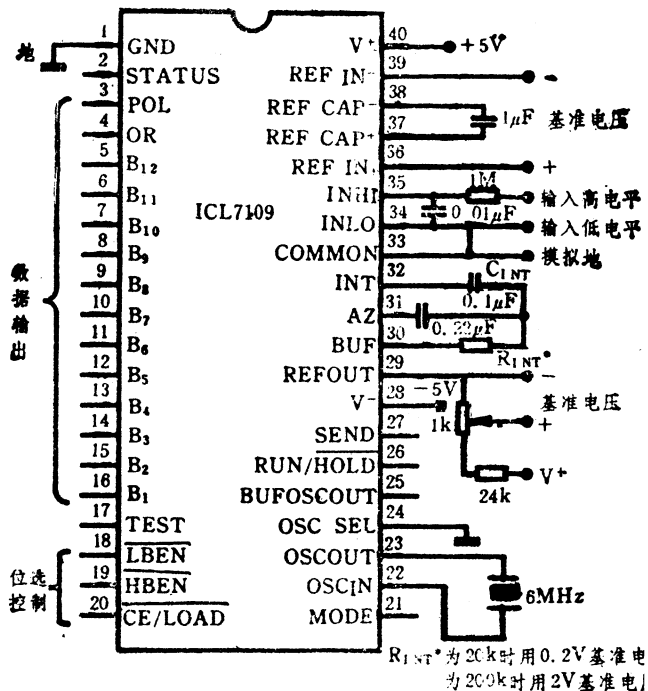


图1 ICL7109新脚及外部连接

参考电压的稳定与否,直接影响转换的精度。

参考电压输入为差分输入,分别由 $\text{REF}_{\text{IN}^+}$ (36脚)及 $\text{REF}_{\text{IN}^-}$ (39脚)引入。参考电压既可以由片内参考电压源供给,也可以由外部提供。

(3)模拟输入信号:模拟信号是差分输入,分别由差分输入高端 $\text{INHI}$ (35脚)和差分输入低端 $\text{INLO}$ (34脚)引入,模拟信号公共端为 $\text{COMMON}$ (33脚)。

(4)时钟电路:ICL7109有片内振荡器及时钟电路。片内提供的多功能时钟振荡器既可用于RC振荡器,也可用作晶体振荡器。

为了使电路具有抗50HZ串模干扰能力,积分时间(相当2048个时钟数)应等于50HZ的整数倍。

(5)工作状态控制:工作状态控制又分成转换状态控制和输出状态控制两部分:

转换状态控制:转换状态由 RUN/HOLD(26脚)输入控制。

若它是高电平时,每经8192个时钟脉冲完成一次转换。每次转换又都经历自动调零、信号积分和消除积分三个阶段。其中自动调零至少需要2048个时钟脉冲周期,信号积分阶段为2048个时钟脉冲周期,消除积分阶段最多为4096个时钟脉冲周期。在消除积分阶段,当输出电压返回零态,并进而过零之后,由模拟电压转换得到的数据将被锁存起来。

若它输入低电平时,转换器将立即结束消除积分阶段,并跳至自动调零阶段,从而缩短了消除积分阶段,提高了转换速度。

输出状态控制:由转换器输出状态标志 STATUS端(2脚)、输出方式选择 MODE端(21脚)、片选 CE/LOAD端(20脚)以及高、低字节输出选通 HBEN端(19脚)、LBEN端(18脚)控制。

输出状态标志(STATUS端):如图2所示,在

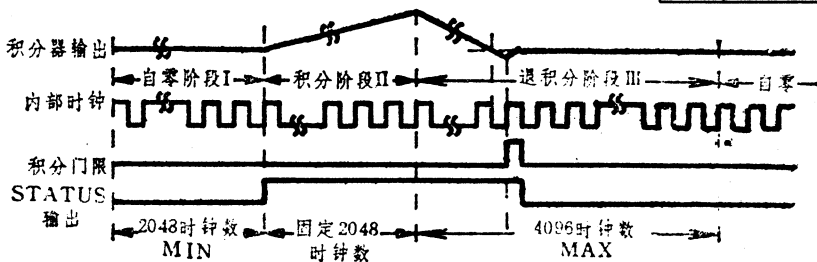


图2 ICL7109的转换时序

每一转换周期中,当自动调零阶段一结束,该状态由低电平跳至高电平,在消除积分阶段,当输出电压返回零态并进而过零,转换输出数据被锁存之后的半个时钟脉冲周期后,它由高电平跳回至低电平。在 STATUS端保持低电平状态时,输出的数据不变。

输出方式选择(MODE端):当 MODE端为低电平时,转换器为直接输出方式,这时数据在片选 CE/LOAD、HBEN及LBEN等信号的控制下直接输出。当 MODE为高电平时,转换器将在信号交换方式下,给每一转换周期的结尾输出数据。

片选 CE/LOAD:在 MODE为低电平时,片选 CE/LOAD为数据输出的主要选通信号。

当该信号为低电平时,为正常输出状态;当其为高电平时,所有数据线处于高阻悬浮状态。

字节选通 HBEN和 LBEN:当 MODE和片选 CE/LOAD均为低电平时, LBEN作为低字节(B1~B8)输出的选通信号; HBEN作为高字节(B9~B12)及 POL,OR输出的选通。其中 POL为输入模拟信号的极性标志, POL=1,表示信号为正;OR为过量程标志,高电平表示过量程。

现将直接输出方式下, MODE, CE/LOAD, HBEN及 LBEN的关系汇总表1中。

表1 直接输出方式下的输出控制

MODE	CE/LOAD	HBEN	LBEN	输出状态
0	1	×	×	所有数据线高阻态
0	0	0	1	高字节输出
0	0	1	0	低字节输出

## 二、实验电路的设计

根据对于 ICL7109芯片的转换过程的理解和对 A/D 转换实验电路的要求,设计12位 A/D 转换实验电路如图3所示。该电路包括转换器及其附属元件、转换控制电路、输入电路、参考电压及-5伏电源发生电路等几个部分。

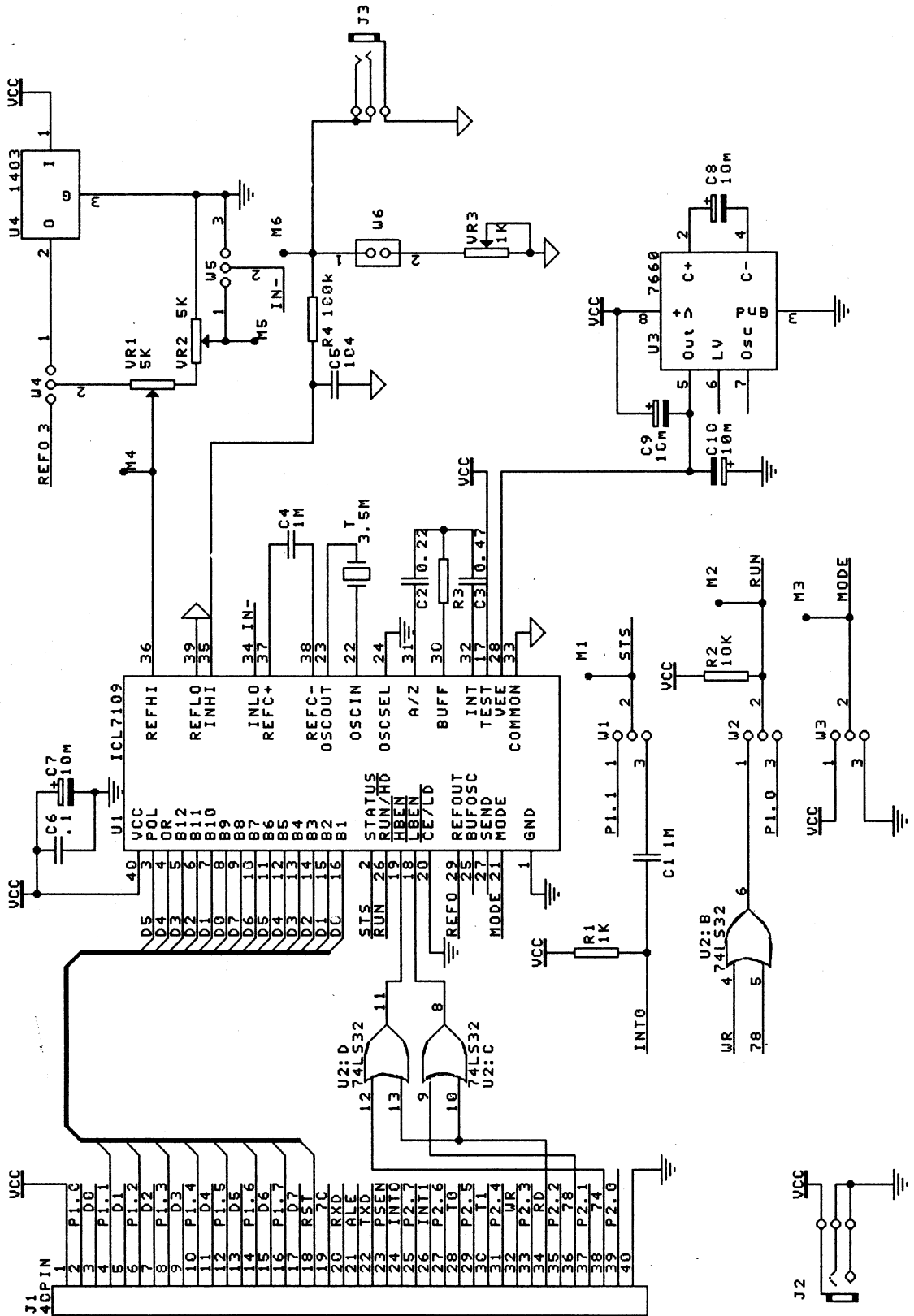
1. 输入电路:由插座 J3, W6, 精密电位器 VR3以及电阻 R4, 电容 C5所组成。R4, C5组成输入滤波网络, J3为模拟信号输入插座, W6用来选择电压信号还是电流信号的。当该插座上插入插块时,电路为电源信号输入。

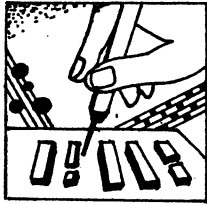
2. 参考电压电路:由稳压块 MC1403及插座 W4, W5, 精密电位器 VR1, VR2组成。插座 W4用来选择内部参考源还是外部参考源的, 电位器 VR1用于调节参考电压。插座 W5用于决定模拟差分输入低端 INLO 电位, 当2-3相联时, INLO 接模拟公共端, 模拟信号为单端输入方式; 当1-2相联时, INLO 端接有一定电

表2 插座功能表

(下转25页)

功能	引脚	插座	插块位置		测试点
			1-2	2-3	
状态标志	STATUS	W1	查询	中断	M1
转换启动方式	RUN/HOLD	W2	MOVX 指令启动	口输出指令启动	M2
输出方式	MODE	W3	信号交换方式	直接输出方式	M3
参考电压选择	REFHZ	W4	外部参考电压	内部参考电压	M4
模拟信号输入方式		W5	差分输入	单端输入	M5
模拟信号类型选择		W6	电流	电压	M6





## 学装微电脑

# 微电脑控制微型钻床

〈学习 X 轴、Y 轴同时驱动〉  
易齐干

双坐标数控机床工作时,由一个坐标点向下一个坐标点移动,多数采用同时驱动 X 轴 Y 轴电机的方法。实现这种方法大致有两种。例如,给 X 轴电机 3 个脉冲(X 轴移动量=分辨率×脉冲数=0.01×3=0.03mm),给 Y 轴电机 5 个脉冲(Y 轴移动量=分辨率×脉冲数=0.01×5=0.05mm)。

(1)两轴对脉冲同时计数,完成指定脉冲数的轴,电机先停止传动。如图 1 所示。

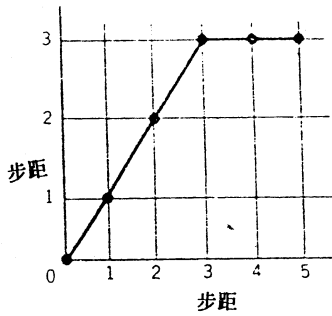


图 1 第 1 种脉冲输出方法

(2)一边对两轴的脉冲进行计数,一边驱动两个电机,对电机进行通/断控制。如图 2 所示。

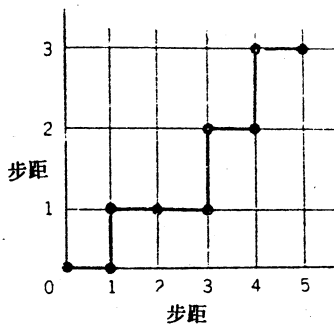


图 2 第 2 种脉冲输出方法

钻床按所要求的座标去钻孔,采用第 1 种方法简单。试分析如下:

编制在印刷线路板上按图形轨迹钻孔程序。

首先决定输入输出口、RAM 区域。分别如表 1、表 2 所示。

座标距离数字化上期文章已介绍。

以上述为基础,则可编制流程图与程序清单。

(1)主程序(MAIN)I/O 端口、RAM 初始化;判断自动运转还是手动运转;启动自动开关,执行 CALL 指令,跳到自动运转子程序;手动时,将工作台移动开关的状态送给电机驱动 IC,电机回转。

表 1 I/O 图

位口	7	6	5	4	3	2	1	0
A (输入)	X 轴编码器脉冲	Y 轴编码器脉冲	记忆开关	自动开关	右开关	左开关	后开关	前开关
C (输出)	未使用		电机	电磁铁	X 轴电机		Y 轴电机	
					正转	反转	正转	反转

表 2 RAM 图

位地址	7	6	5	4	3	2	1	0
8000H	电机驱动数据							
8001H	1: Y 轴 0: Y 轴 正转	1: X 轴 0: X 轴 正转			1: Y 轴 脉冲有 变化	1: X 轴 脉冲有 变化	1: Y 轴 数据 = 0	1: X 轴 数据 = 0
	记录 X 轴脉冲的前次数据							
8002H	记录 X 轴脉冲的前次数据							
8003H	记录 Y 轴脉冲的前次数据							

(2)自动运转程序(AUTO)

执行 CALL 指令,跳向数据传送子程序;电机驱动子程序;钻头驱动子程序;最后,查询数据是否结束。

(3)数据传送程序(DTCONV)

将 X 轴移动数据送到 BC 寄存器对,Y 轴移动数据送到 DE 寄存器对,设置回转方向标志。

标号	助记符	地址	机器码	注释
MAIN	LD A,90H	0000	3E 90	A 口输入,B 口、C
	OUT (03H),A	0002	D3 03	口输出
	LD SP,8100	0004	31 00 81	设置堆栈指针
	LD HL,8000H	0007	21 00 80	
	LD A,FFH	000A	3E FF	电机驱动数据为
	LF (HL),A	000C	77	FFH
	OUT (02H),A	000D	D3 02	
	XOR A	000F	AF	数据输出
	INC HL	0010	23	
	LD (HL),A	0011	77	标志位清零
	INC HL	0012	23	
	LD (HL),A	0013	77	X 轴脉冲数据清零
	INC HL	0014	23	
	LD (HL),A	0015	77	Y 轴脉冲数据清零
J1	IN A,(00H)	0016	DB 00	开关输入
	BIT 4,A	0018	CB 67	自动开关不通时
	JP NZ,J2	001A	C2 23 00	跳入 J2
	CALL AUTO	001D	CD 2A 00	
	JP NZ,J2	001D	CD 2A 00	
J2	IN A,(00H)	0023	DB 00	向 A 口输出开关
	OUT (02H),A	0025	D3 02	状态
	JP J1	0027	C3 16 00	

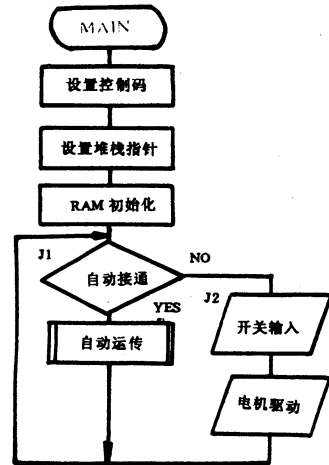
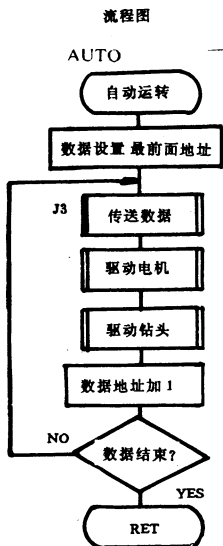


图3 主程序流程图与清单



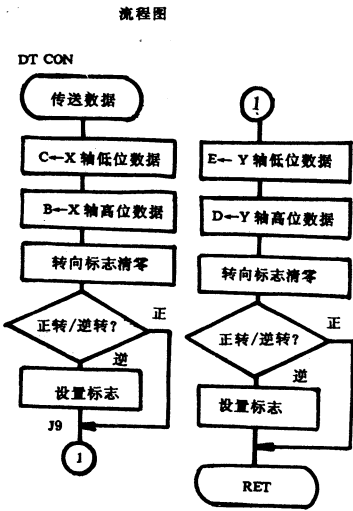
标号	助记符	地址	机器码	注释
AUTO	LD HL,0200H	002A	21 00 02	设置数据最前面地址
J3	CALL DTCONV	002D	CD 3E 00	
	CALL XYAUTO	0030	CD 64 00	
	CALL DAUTO	0033	CD 5A 01	
	INC HL	0036	23	
	LD A,(HL)	0037	7E	
	CP FFH	0038	FE FF	数据没结束时
	JP NZ,J3	003A	C2 2D 00	(FFH)跳向 J3
	RET	003D	C9	

图4 自动运转程序流程图与清单

(4) 电机驱动程序(XYAUTO)

如果脉冲有变化,电机驱动数据减1。驱动电机程序与转动方向的标志对应。CPU 处理速度比电机转速

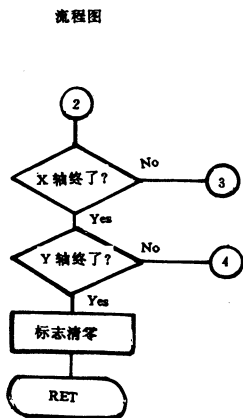
要快,通过对 X 轴和 Y 轴编码器脉冲进行检验,不会遗漏脉冲读数。



标号	助记符	地址	机器码	注释
DTCONV	LD C,(HL)	003E	4E	C←X轴低位数据
	INC HL	003F	23	数据地址+1
	LD B,(HL)	0040	46	B←X轴高位数据
	RES 7,B	0041	CB B8	转向标志清零
	BIT 7,(HL)	0043	CB 7E	X轴高位数据 第7位如果为1则 反转,设置标志
	JP Z,J9	0045	CA 50 00	
	LD A,(8001H)	0048	3A 01 80	
	SET 6,A	004B	CB F7	
	LD (8001H),A	004D	32 01 80	
J9	INC HL	0050	23	数据地址+1
	LD E,(HL)	0051	5E	E←Y轴低位数据
	INC HL	0052	23	数据地址+1
	LD D,(HL)	0053	56	D←Y轴高位数据
	RES 7,D	0054	CB BA	转向标志清零
	BIT 7,(HL)	0056	CB 7E	Y轴高位数据 的第7位如果为1, 则反转.设置标志
	JP Z,J10	0058	CA 63 00	
	LD A,(8001H)	005B	3A 01 80	
	SET 7,A	005E	CB FF	
	LD A,(8001H)	0060	32 01 80	
J10	RET	0063	C9	

(上接第32页)

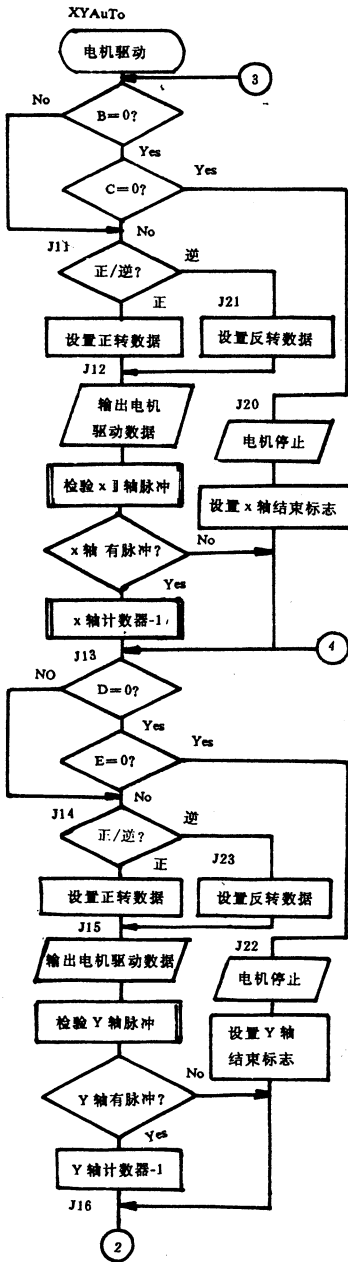
图5 数据传送程序流程图与清单



标名	助记符	地址	机器码	注释
J22	LD A,(8000H)	00E5	3A 00 80	设置Y轴电机 停止数据
	SET 1,A	00E8	CB CF	
	SET 0,A	00EA	CB C7	
	LD (8000H)	00F1	3A 01 80	输出电机驱动数据
	OUT (02H),A	00EF	D3 02	
	LD A,(8001H)	00F1	3A 01 80	设置X轴结束标志
	SET 1,A	00F4	CB CF	
	LD (8001H),A	00F6	32 01 80	
	JP J16	00F9	C3 B4 00	
J23	LD A,(8000H)	00FC	3A 00 80	将电机驱动数据 设置在Y轴反转上
	RES 0,A	00FF	CB 87	
	JP J15	0101	C3 A3 00	

图6 电机驱动程序流程图与清单

流程图

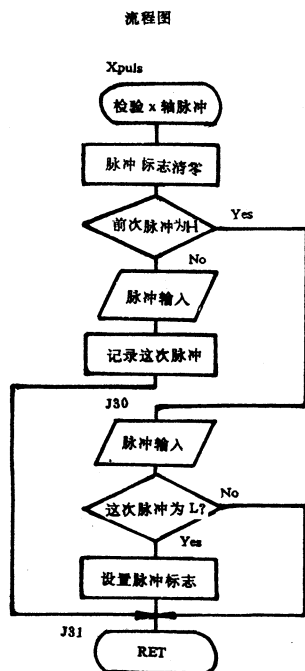


标名	助记符	地址	机器码	注释
XYAUTO	LD A,B	0064	78	X轴高位数据不为0时,向J11跳
	AND A	0065	A7	
	JP NZ,J11	0066	C2 6E 00	
	LD A,C	0069	79	X轴低位数据为0时,向J20跳
	AND A	006A	A7	
	JP Z,J20	006B	CA C6 00	
J11	LD A,(8001H)	006E	3A 01 80	反转标志成立时,向J21跳
	BIT 6,A	0071	CB 77	
	JP NZ,J21	0073	C2 DD 00	
	LD A,(8000H)	0076	3A 01 80	电机驱动数据设置在X轴正转
	RES 3,A	0079	CB 9F	
	J12	LD (8000H),A	007B	
	OUT (02H),A	007E	D3 02	X轴的脉冲由H→L变化时,计数器值减1
	CALL XPULSE	0080	CD 04 01	
	LD A,(8001H)	0083	3A 01 80	
	BIT 2,A	0086	CB 57	Y轴高位数据不为0跳向J14
	JP Z J13	0088	CA 8C 00	
	DEC BC	008B	0B	
J13	LD A,D	008C	7A	Y轴低位数据为0跳向J15
	AND A	008D	A7	
	JP NZ,J14	008E	C2 96 00	
	LD A,E	0091	7B	反转标志成立时跳向J23
	AND A	0092	A7	
	JP Z,J22	0093	CA E5 00	
J14	LD A,(8001H)	0096	3A 01 80	电机驱动数据设置在Y轴
	BIT 7,A	0099	CB 7F	
	JP NZ,J23	009B	C2 FC 00	
	LD A,(008H)	009E	3A 00 80	输出电机驱动数据
	RES 1,A	00A1	CB 8F	
	J15	LD (8000H),A	00A3	
	OUT (02H),A	00A6	D3 02	Y轴脉冲如果由H→L变化,计数器减1
	CALL YPULSE	00A8	CD 2F 01	
	LD A,(8001H)	00AB	3A 01 80	
	BIT 3,A	00AE	CB 5F	X轴数据为0?
	JP Z,J16	00B0	CA B4 00	
	DEC DE	00B3	1B	
J16	LD A,(8001H)	00B4	3A 01 80	Y轴数据为0?
	BIT 0,A	00B7	CB 47	
	JP Z,XYAUTO	00B9	CA 64 00	
	BIT 1,A	00BC	CB 4F	标志清零
	JP Z,J13	00BE	CA 8C 00	
	XOR A	00C1	AF	
	LD (8001H),A	00C2	32 01 80	设置X轴电机停止数据
	RET	00C5	C9	
	J20	LD (8001H),A	00C6	
	SET 3,A	00C9	CB DF	输出电机驱动数据
	SET 2,A	00CB	CD D7	
	LD (8001H),A	00CD	32 00 80	
	OUT (02H),A	00D0	D3 02	设置X轴结束标志
	LD A,(8001)	00D2	3A 01 80	
	SET 0,A	00D5	CB C7	
	LD (8001H),A	00D7	32 01 80	将电机驱动数据设置在X轴逆转上
	JP J13	00DA	C3 8C 00	
	J21	LD A,(8000H)	00DD	
	RES 2,A	00E0	CB 97	
	JP J12	00E2	C3 7B 00	

图6

(下转第31页)





标号	助记符	地址	机器码	注释
XPULSE	LD A,(8001H)	0104	3A 01 80	X 轴有脉冲,标志清零
	RES 2,A	0107	CB 97	
	LD (8001H),A	0109	32 01 80	
	LD A,(8002H)	010C	3A 02 80	前次脉冲为 L 时,记录这次脉冲
	BIT 6,A	010F	CB 77	
	JP NZ,J30	0111	C2 1C 01	
	IN A,(00H)	0114	DB 00	
	LD (8002H),A	0016	32 02 80	
	JP J31	0119	C3 2E 01	
	J30 IN A,(00H)	011C	DB 00	
	LD (8002H),A	011E	32 02 80	前次脉冲为 H,这次脉冲为 L 时,有脉冲,设置标志
	BIT 6,A	0121	CB 77	
	JP NZ,J31	0123	C2 2E 01	
	LD A,(8001H)	0126	3A 01 80	
	SET 2,A	0129	CB D7	
	LD (8001),A	012B	32 01 80	
J31	RET	012E	C9	

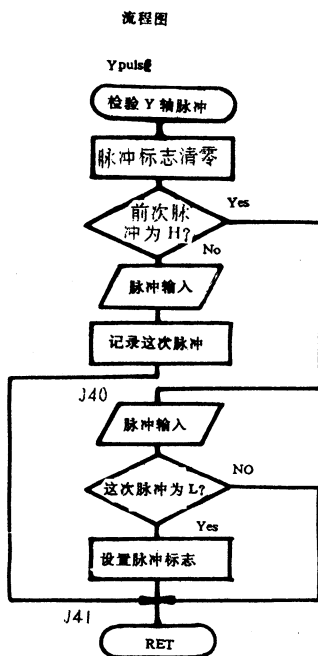
图7 X 轴脉冲检验程序流程图与清单

(5) X 轴、Y 轴脉冲检验程序(XPULS,YPULS)。

只限于各轴的脉冲从“H”电平到“L”电平变化时,设置标志,表示有1个脉冲输入。

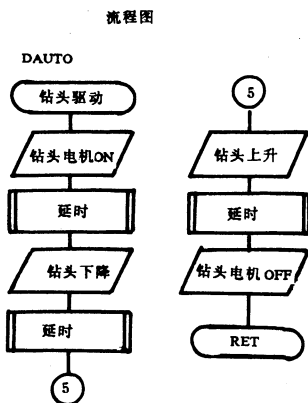
(6) 驱动钻头程序(DAUTO)

钻头工作循环:钻头旋转→钻头下降→钻孔→钻头上升→钻头旋转停止。



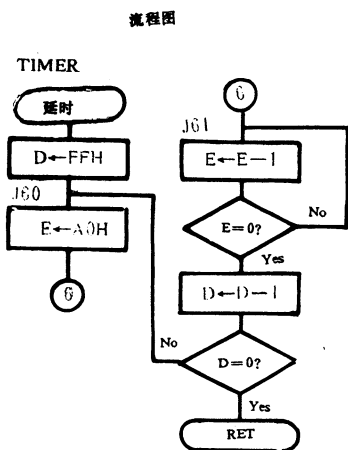
标号	助记符	地址	机器码	注释
YPULSE	LD A,(8001H)	012F	3A 01 80	Y 轴脉冲有标志清零
	RES 3,A	0132	CB 9F	
	LD (8001H),A	0134	32 01 80	
	LD A,(8003H)	0137	3A 03 80	
	BIT 7,A	013A	CB 7F	前次脉冲为 L 时,记录这次脉冲
	JP NZ,J40	013C	C2 47 01	
	IN A,(00H)	013F	DB 00	
	LD (8003H),A	0141	32 03 80	
	JP J41	0144	C3 59 01	
	J40 IN A,(00H)	0147	DB 00	
	LD (8003H),A	0149	32 03 80	前次脉冲为 H,这次脉冲为 L 时,有脉冲,设置标志。
	BIT 7,A	014C	CB 7F	
	JP NZ,J41	014E	C2 59 01	
	LD A,(8001H)	0151	3A 01 80	
	SET 3,A	0154	CB DF	
	LD (8001H),A	0156	32 01 80	
J41	RET	0159	C9	

图8 Y 轴脉冲检验程序流程图与清单



标记	助记符	地址	机器码	注释
DAUTO	LD A,DFH	015A	3E DF	
	OUT (02H),A	015C	D3 02	
	CALL TIMER	015E	CD 7A 01	
	LD A,CFH	0161	3E CF	
	OUT (02H),A	0163	D3 02	
	LD B,04H	0165	06 04	
J50	CALL TIMER	0167	CD 7A 01	延时程序执行 4 次(钻孔时间)
	DEC B	016A	05	
	JP NZ,J50	016B	C2 67 01	
	LD A,DFH	016E	3E DF	
	OUT (02H),A	0170	D3 02	
	CALL TIMER	0172	CD 7A 01	
	LD A,FFH	0175	3E FF	
	OUT (02H),A	0177	D3 02	
	RET	0179	C9	

图9 钻头驱动程序流程图与清单  
(7)延时程序(TIMER)  
满足钻孔所需要的时间



标号	助记符	地址	机器码
TIMER	LD D,FFH	017A	16 FF
J60	LD E,A0H	017C	1E A0
J61	DEC E	017E	1D
	JP NZ,J61	017F	C2 7E 01
	DEC D	0182	15
	JP NZ,J60	0183	C2 7C 01
	RET	0186	C9

图10 延时程序流程图与清单

(8)数据

由200H地址存放电机移动数据,结束标志为FFH。

标记	助记符	地址	机械语
DATA	DB	0200	00 00 00 00
	DB	0204	2C 01 28 80
	DB	0208	2C 01 78 80
	DB	020C	CA 00 A0 80
	DB	0210	96 00 F0 80
	DB	0214	32 00 F0 80
	DB	0218	32 80 F0 80
	DB	021C	96 80 F0 80
	DB	0220	CA 80 A0 80
	DB	0224	2C 81 78 80
	DB	0228	2C 81 28 80
	DB	022C	2C 81 28 00
	DB	0230	2C 81 78 00
	DB	0234	CA 80 A0 00
	DB	0238	96 80 F0 00
	DB	023C	32 80 F0 00
	DB	0240	32 00 F0 00
	DB	0244	96 00 F0 00
	DB	0248	CA 00 A0 00
	DB	024C	2C 01 78 00
	DB	0250	FF

数据

以上为微型钻床两轴同时移动的全部程序。

微型钻床为理想的教学装置。有订购者请与天津纺织工学院机械系 高殿斌同志联系。邮码:300160 电话:41.2833转983。



# 电脑巧开发 数字集成电路简易测试器

宁夏银川 王正英

## 一、概述:

数字集成电路在使用过程中经常需要检测好坏。比较实用有效的方法是测试其逻辑功能是否正确,本文介绍一个由 CEC-I 微机 and 可编程 I/O 器件 MC6821 构成的数字集成电路简易测试器,由于 MC6821 的两组 (PA<sub>0</sub>~PA<sub>7</sub>, PB<sub>0</sub>~PB<sub>7</sub>) 16 根双向数据

线每一根都可以通过写入控制字(编程)被确定为输入线或输出线,这样就可以利用 MC6821 在不改变接线的情况下对不同管脚排列的数字集成电路进行逻辑功能的测试。因而电路简单、制做容易,成本低廉。测试软件可由汇编语言编制、也可由 BASIC 语言编制,为了便于阅读本文的测试软件由 BASIC 语言编制。表(1)

型号	型号
74LS00	CD4002
02	4078
03	4012
04	4068
11	4072
20	4073
30	4069
32	4085
51	4043
54	4013
66	4027
74	40106
112	4518
85	4520
125	4017
138	4511
147	4028
151	4015
160	4066
183	4051
194	4512
248	4070
280	4008
390	CC14561
42	CC14585

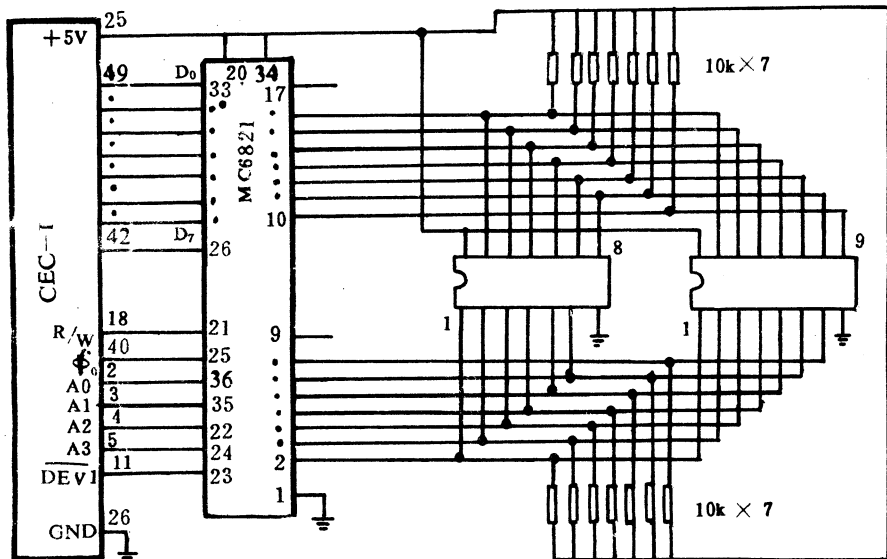


图1

(见附程序1)。限于篇幅本文仅给出50种型号数字集成电路的数据(见表1和表3),其中包括 TTL 电路 CMOS 电路、组合电路、时序电路。OC 门以及三态门。该测试器所用的全部元件只需一块 MC6821 集成电路,两个集成电路插座和14个  $\frac{1}{8}$  W 电阻,只要接线正确,不需调试即可成功。具体电路如图1所示,该测试器与 CEC-I 微机的1号扩充槽接口。

## 二、测试原理简介

首先将选定的各种数字集成电路正确的逻辑关系(各种输入输出组合)以一定的结构方式作为数据送入内存。当测试某芯片时首先把对应该芯片的控制字写入 MC6821 来确定 PA<sub>0</sub>~PA<sub>7</sub>, PB<sub>0</sub>~PB<sub>7</sub> 是输入线还是

输出线(有关 MC6821 的编程详见本刊1990年第十期),以便与被测芯片正确连接。然后利用 POKE 命令通过 MC6821 向被测芯片送入相应的各种逻辑电平,接着再用 PEEK 命令从被测芯片读得输出电平,并与已存入内存的正确的逻辑关系比较,若一致说明被测芯片是好的,否则是坏的,下面以 74LS04 (6反相器)为例加以说明:测试 74LS04 时该芯片各引脚与 MC6821 的 PA 口、PB 口的连接如图(2a)所示。其中 PA<sub>0</sub>、PA<sub>2</sub>、PA<sub>4</sub>、PB<sub>2</sub>、PB<sub>4</sub>、PB<sub>6</sub> 与 6 个反相器的输入端相连,对 MC6821 而言是输出端故这些线被定义为输出线(1)。PA<sub>1</sub>、PA<sub>3</sub>、PA<sub>5</sub>、PB<sub>1</sub>、PB<sub>3</sub>、PB<sub>5</sub> 与 6 个反相器的输出端相连,对 MC6821 而言是输入端,故这些线定义为输入线(0)。PA<sub>6</sub>、PA<sub>7</sub>、PB<sub>6</sub>、PB<sub>7</sub> 未使用,在这里定义为输出线(1)。这样对 74LS04 写入的控制字如图(2b)所示。化为十进制数为 PA 口:213, PB 口:213。在测试程序中利

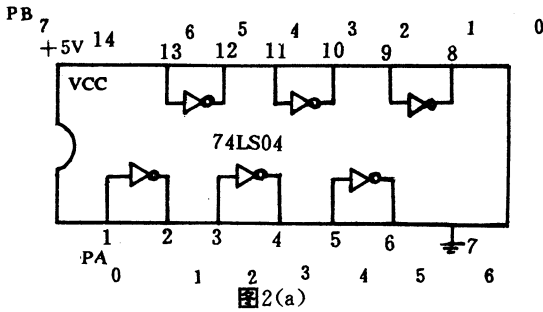


图2(a)

用 POKE 命令就可以把 74LS04 的控制字写入 MC6821, 从而完成了 MC6821 与 74LS04 的正确连接。74LS04 对应于 PA 口和 PB 口的真值表如表(2)所示。

由表(2)可知从 PA 口向 74LS04 送入的数据(十进制数)应为 42, 21。从 PB 口向 74LS04 送入的数据(十进制数)应为 42, 84。测试程序运行时以这组数据作为标准即可判断出 74LS04 的好坏, 从而完成了测试任务。

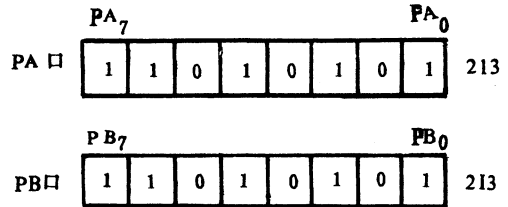


图2(b)

表2

	PA <sub>7</sub>	PA <sub>6</sub>	PA <sub>5</sub>	PA <sub>4</sub>	PA <sub>3</sub>	PA <sub>2</sub>	PA <sub>1</sub>	PA <sub>0</sub>	PB <sub>7</sub>	PB <sub>6</sub>	PB <sub>5</sub>	PB <sub>4</sub>	PB <sub>3</sub>	PB <sub>2</sub>	PB <sub>1</sub>	PB <sub>0</sub>	
	空	空	6	5	4	3	2	1	空	13	12	11	10	9	8	空	
42	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1	0	42
21	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	0	84

### 三、使用方法

1 测试前将表(3)的数据在监控状态下送入内存,

并以 SJ 为名存盘。

表(3)

4000- 00 00 DB ED 00 01 00 08  
 4008- 00 02 F6 B7 00 09 00 10  
 4010- 00 03 DB ED 00 11 00 18  
 4018- 00 04 D5 D5 00 19 00 1C  
 4020- 00 0B DF DD 00 1D 00 26  
 4028- 00 14 DF FD 00 27 00 32  
 4030- 00 1E FF FD 00 33 00 46  
 4038- 00 20 DB ED 00 47 00 4E  
 4040- 00 33 DF FD 00 4F 00 5C  
 4048- 00 36 DF FF 00 5D 00 6E  
 4050- 00 56 DB ED 00 6F 00 76  
 4058- 00 4A CF F9 00 77 00 8C  
 4060- 00 70 8F FE 00 8D 00 B4  
 4068- 00 55 8F FF 00 B5 00 D0  
 4070- 00 7D DB ED 00 D1 00 D4  
 4078- 00 8A BF 80 00 D5 00 EC  
 4080- 00 93 9F DE 00 ED 01 00  
 4088- 00 97 CF FF 01 01 01 22  
 4090- 00 A0 FF 83 01 23 01 40  
 4098- 00 B7 CF F5 01 41 01 50  
 40A0- 00 C2 FF 87 01 51 01 6E  
 40A8- 00 F8 FF 80 01 6F 01 94  
 40B0- 01 18 CF FF 01 95 01 A8  
 40B8- 01 86 8B E8 01 A9 01 C8  
 40C0- 00 2A 80 F8 01 C9 01 E8  
 40C8- 0F A2 FE BF 01 E9 01 F4  
 40D0- 0F EE FE BF 01 F5 02 08  
 40D8- 0F AC FE BF 02 09 02 14

40E0- 0F E4 FE BF 02 15 02 28  
 40E8- 0F E8 FE BF 02 29 02 34  
 40F0- 0F E9 FE BF 02 35 02 3E  
 40F8- 0F E5 D5 D5 02 3F 02 42  
 4100- 0F F5 F3 FF 02 43 02 50  
 4108- 0F CB FC FC 02 51 02 5E  
 4110- 0F AD FC 9F 02 5F 02 74  
 4118- 0F BB FC 9F 02 75 02 9C  
 4120- 9C AA D5 D5 02 9D 02 A0  
 4128- 11 A6 C3 C3 02 A1 02 D8  
 4130- 11 A8 C3 C3 02 D9 03 1E  
 4138- 0F B1 F0 80 03 1F 03 54  
 4140- 11 9F FF 80 03 55 03 80  
 4148- 0F BC 80 9E 03 81 03 A0  
 4150- 0F AF E1 E1 03 A1 03 B8  
 4158- 0F E2 DB F3 03 B9 03 C0  
 4160- 0F D3 BB FF 03 C1 03 E4  
 4168- 11 A0 FF C3 03 E5 04 0A  
 4170- 0F E6 F3 E7 04 0B 04 12  
 4178- 0F A8 FF C1 04 13 04 22  
 4180- 38 E1 E1 87 04 23 04 3E  
 4188- 38 F9 FB E7 04 3F 04 54  
 4190- 24 12 36 5A 2D 36 1B 6C  
 4198- 09 48 24 24 12 12 36 36  
 41A0- 24 12 36 5A 2D 36 1B 6C  
 41A8- 2A 2A 15 54 00 00 10 50  
 41B0- 0A 08 05 04 3F 7E 20 02  
 41B8- 30 42 28 22 22 0A 21 06  
 41C0- 1B 6C 00 02 00 22 00 12

41C8- 20 02 10 02 08 02 04 02  
 41D0- 02 02 01 02 3F 30 00 00  
 41D8- 36 5A 2D 36 3F 7E 20 02  
 41E0- 34 52 2A 2A 1F 06 19 7C  
 41E8- 06 82 07 60 20 00 32 50  
 41F0- 29 28 07 64 1F 7C 03 00  
 41F8- 1C 00 00 1C 00 60 00 00  
 4200- 36 5A 2D 36 1B 6C 2A 2A  
 4208- 29 4A 2D 5A 2B 6A 1F 7C  
 4210- 1B 6C 1F 7C 19 4C 2D 5A  
 4218- 11 44 1B 6C 69 12 69 72  
 4220- 11 71 19 73 18 63 1B 7B  
 4228- 6A 6A 6B 7A 6A 6A 69 72  
 4230- 68 62 6D 76 1C 67 1D 77  
 4238- 1C 67 1F 7F 6E 6E 6F 7E  
 4240- 1E 6F 6E 0E 10 40 41 00  
 4248- 10 10 40 20 10 08 40 04  
 4250- 10 02 40 01 18 00 42 00  
 4258- 20 00 20 00 0A 00 50 00  
 4260- 00 00 36 36 60 3F 61 5F  
 4268- 62 6F 63 77 64 7B 65 7D  
 4270- 66 7E 27 7F 40 7F 78 7F  
 4278- 50 7F 48 7F 7F 3F 7F 1C  
 4280- 6F 1F 17 3E 1B 3F 5D 3E  
 4288- 5E 3F 3F 2E 3F 37 7F 3A  
 4290- 68 00 18 00 2B 7C 12 02  
 4298- 2E 7E 10 41 2F 5D 10 13  
 42A0- 2F 77 27 78 14 04 2D 7A  
 42A8- 11 06 2F 39 10 25 2F 6B

42B0- 10 0F 1C 01 1D 00 1F 38  
 42B8- 41 3B 43 07 41 07 43 67  
 42C0- 41 67 43 03 41 03 43 23  
 42C8- 01 23 03 23 41 21 43 21  
 42D0- 00 00 21 42 24 22 15 68  
 42D8- 28 12 19 58 1C 38 3D 7A  
 42E0- 00 03 15 03 15 57 01 51  
 42E8- 01 2D 03 29 03 55 01 52  
 42F0- 01 26 41 22 41 4E 03 48  
 42F8- 03 4C 41 48 41 4C 1C 5F  
 4300- 5C 0C 1D 3B 5D 3E 1E 3C  
 4308- 5E 76 1F 77 5F 1C 3C 7F  
 4310- 7C 3F 3D 23 7D 26 3E 68  
 4318- 7E 72 3F 63 7F 00 77 00  
 4320- 04 00 7B 7F 10 00 28 00  
 4328- 1A 00 2B 00 1B 40 2B 60  
 4330- 1B 70 2B 78 1B 7C 2B 7E  
 4338- 0B 68 09 68 0C 18 0D 58  
 4340- 08 08 09 48 11 44 19 4C  
 4348- 21 42 29 4A 31 46 39 4E  
 4350- 41 41 49 49 01 40 09 48  
 4358- 7E 07 7D 47 7B 27 77 67  
 4360- 6F 17 5F 57 3F 37 7F 76  
 4368- 7F 0D 7F 4B 7F 2F 7F 6F  
 4370- 7F 1F 7F 5F 7F 3F 7F 7F  
 4378- 01 40 02 04 04 08 08 10  
 4380- 10 20 1E 3C 00 40 03 00  
 4388- 05 00 09 00 11 00 01 04  
 4390- 01 08 01 10 01 20 1F 3C  
 4398- 1E 3C 1D 78 1B 74 17 6C  
 43A0- 0F 5C 01 40 1F 3C 1C 7C  
 43A8- 1A 7C 16 7C 0E 7C 1E 78  
 43B0- 1E 74 1E 6C 1E 5C 00 62  
 43B8- 00 00 03 44 05 48 09 50  
 43C0- 11 60 1F 7C 00 00 0F 60

43C8- 15 52 1A 32 3F 7E 2A 2A  
 43D0- 15 54 0C 00 2E 44 1D 22  
 43D8- 33 66 33 00 00 66 00 18  
 43E0- 3B 2B 13 03 54 44 10 00  
 43E8- 3B 2B 47 47 2B 2B 21 42  
 43F0- 01 40 06 30 02 20 06 30  
 43F8- 12 24 15 54 11 44 15 54  
 4400- 24 2C 0B 6A 2A 2A 61 43  
 4408- 6B 6B 2A 2A 22 22 25 52  
 4410- 21 42 25 52 31 46 36 36  
 4418- 32 26 34 56 01 40 05 50  
 4420- 11 44 14 34 12 24 16 34  
 4428- 02 20 06 30 2A 2A 15 54  
 4430- 42 42 02 02 07 07 06 06  
 4438- 0B 0B 0A 0A 0F 0F 0E 0E  
 4440- 13 13 12 12 17 17 16 16  
 4448- 1B 1B 1A 1A 1F 1F 1E 1E  
 4450- 23 23 22 22 27 27 26 26  
 4458- 03 03 01 01 00 00 02 02  
 4460- 04 04 06 06 08 08 09 09  
 4468- 42 42 02 02 07 07 06 06  
 4470- 0B 0B 0A 0A 0F 0F 0E 0E  
 4478- 13 13 12 12 17 17 16 16  
 4480- 1B 1B 1A 1A 1F 1F 1E 1E  
 4488- 20 20 22 22 24 24 26 26  
 4490- 28 28 2A 2A 2C 2C 2E 2E  
 4498- 30 30 32 32 34 34 36 36  
 44A0- 38 38 3A 3A 3C 3C 3C 3C  
 44A8- 3D 3D 3F 3F 3D 3D 48 04  
 44B0- 08 04 28 02 08 02 28 08  
 44B8- 08 08 28 40 08 40 2A 00  
 44C0- 0A 00 20 01 00 01 20 10  
 44C8- 00 10 20 20 00 20 21 00  
 44D0- 01 00 24 00 04 00 28 04

44D8- 08 04 28 02 38 02 28 08  
 44E0- 08 08 28 40 08 7F 67 00  
 44E8- 0C 5F 4C 0C 0D 3B 4D 3E  
 44F0- 0E 6C 4E 6C 0F 67 4F 1C  
 44F8- 2C 7F 6C 7C 2D 00 6D 00  
 4500- 2E 00 6E 00 2F 00 6F 00  
 4508- 7F 00 6C 7C 7C 5C 7C  
 4510- 04 00 00 22 02 10 00 52  
 4518- 01 08 20 0A 40 18 08 1A  
 4520- 00 05 10 06 00 14 00 16  
 4528- 00 0C 00 0E 00 1C 00 1E  
 4530- 61 61 41 41 40 40 51 51  
 4538- 10 10 09 09 48 48 55 55  
 4540- 14 14 0B 0B 4A 4A 55 55  
 4548- 3F 7E 30 60 0F 1E 06 0C  
 4550- 07 50 03 54 07 52 03 56  
 4558- 07 51 03 55 07 53 03 57  
 4560- 18 28 1C 2C 18 2A 1C 2E  
 4568- 18 29 1C 2D 18 2B 1C 2F  
 4570- 3C 28 27 40 55 20 55 04  
 4578- 55 28 55 0C 55 30 55 14  
 4580- 55 38 55 1C 01 02 7F 7F  
 4588- 00 7C 2A 01 2A 25 2A 09  
 4590- 2A 2D 2A 11 2A 35 2A 19  
 4598- 2A 3D 00 00 2E 3A 1D 5C  
 45A0- 33 66 00 00 00 1F 2A 5E  
 45A8- 2A 61 55 1E 55 21 7F 60  
 45B0- 7F 7F 08 08 24 10 32 20  
 45B8- 10 48 18 40 14 38 1C 30  
 45C0- 12 28 1A 20 16 18 1E 10  
 45C8- 11 08 19 00 0F 04 08 50  
 45D0- 0A 10 48 10 08 12 08 10  
 45D8- 24 00 10 08 00 0C 00 09  
 45E0- 01 08 00 28 00

2 测试时将测试器与 CEC-I 微机的扩充槽接好,将被测芯片插在测试座上。

3 开机后运行程序(1),用户可根据屏幕上的提示键入被测芯片的型号,按下回车键后计算机将自动进行型号的查找,控制字的写入以及测试工作,测试完毕后屏幕上将显示出“GOOD!”或“BAD!”。当显示“BAD!”时还将同时显示出被测芯片各引脚的逻辑电平值(1或0),里圈是正确的逻辑值,外圈是错误的逻辑值。

4 测试完毕后按任意键,程序返回主菜单开始下一次测试。

#### 四、几点说明

1. 该测试器可测试14脚和16脚的数字集成电路。对 CMOS 数字集成电路要求  $V_{DD}=3\sim 18V$ 。

2. 图1电路中14个电阻是为了测试 OC 门和三态门设置的。测试其它电路时电阻 R 的引入将会在相应端头为低电平时增加输出门的灌电流,但由于 R 的阻值为  $10K\Omega$ ,增加的灌电流最大不超过  $0.5mA$ ,经反复试验证明 R 的引入对逻辑功能的测试没有影响。

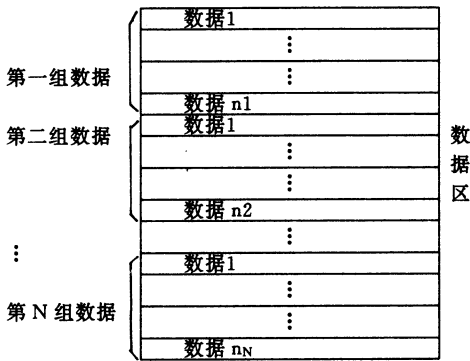
3. 测试时序电路的数据采用了模拟时钟变化的方法,即对应时钟端的数据反复地0,1变化,0,1变化时其它端头的数据保持不变。所以对时序芯片(触发器、计数器、移位寄存器)其数据的顺序不能随意变换。

4. 表3的测试数据采用了索引区+数据区的结构如表4、表5所示。增加数据时只需在相应的区域内按表4、表5的格式填写即可。由于数据的存放采用了相对地址,所以原则上数据可放在 CEC-I 内存的任意处,此时只要对测试程序稍加修改即可。

表(4)

索引1	型号1高位	索引区
	型号1低位	
	PA 口控制字	
	PB 口控制字	
	第一组数据首地址高位	
第一组数据首地址低位		
第一组数据末地址高位		
第一组数据末地址低位		
：	：	
：	：	
索引 N	型号 N 高位	
	：	
	第 N 组数据末地址低位	

表(5)



5 连续测试的过程中更换芯片是带电进行的。为了安全起见最好给被芯片的电源加一个开关。没有开关一般也不会损坏芯片。

6 该测试器也能在 APPLE-I 微机上使用。只要把程序 1 中的地址 49308 改为 49404, 49309 改为 49405, 49310 改为 49406, 49311 改为 49407 即可。此新地址对应 APPLE-I 的 7 号扩充槽。

程序(1)

```

10 D$=CHR$(4)
20 IF PEEK(6)=5 THEN 40
30 PRINT D$ "BLOAD SJ, A$4000"
40 POKE 6,5
50 C$="***TTL***";D$="***CMOS***"
60 V1=16384;V2=16583;V3=16584;V4=16783;A1=16783
70 HOME
80 PRINT"1---TTL";PRINT
90 PRINT"2---CMOS";PRINT
100 PRINT"3-END";PRINT;PRINT;PRINT;PRINT
110 INPUT"(1,2,3...?)";P
120 IF P<>1 AND P<>2 AND P<>3 THEN 70
130 IF P=3 THEN 510
140 IF P=1 THEN W1=V1;W2=V2;J$=C$;GOTO 160
150 IF P=2 THEN W1=V3;W2=V4;J$=D$
160 HOME;VTAB 8;HTAB 15;PRINT J$;PRINT;PRINT;PRINT;INPUT"---->IN";I
170 FOR J=W1 TO W2 STEP 3
180 XH=PEEK(J);XD=PEEK(J+1);Y2=XH;Y1=XD
190 GOSUB 530
200 X=Z
210 IF X<>I THEN 300
220 KA=PEEK(J+2);KB=PEEK(J+3)
230 SH=PEEK(J+4);SD=PEEK(J+5);Y2=SH;Y1=SD
240 GOSUB 530
250 SS=Z

```

```

260 MH=PEEK(J+6);MD=PEEK(J+7);Y2=MH;Y1=MD
270 GOSUB 530
280 SM=Z
290 GOTO 320
300 NEXT J
310 GOTO 70
320 POKE 49309,0;POKE 49311,0;POKE 49308,KA;POKE 49310,KB;POKE 49309,4;POKE 49311,4
330 FOR L=A1+SS TO A1+SM STEP 2
340 AX=PEEK(L);BX=PEEK(L+1)
350 POKE 49308,AX;POKE 49310,BX
360 AD=PEEK(49308);BD=PEEK(49310)
370 IF AD<>AX OR BD<>BX THEN 420
380 NEXT L
390 HOME;VTAB 8;HTAB 10;PRINT I;"---GOOD!"
400 MUSIC192,30;MUSIC171,30;MUSIC152,30;MUSIC140,30;MUSIC128,30;MUSIC114,30;MUSIC102,30;MUSIC95,110
410 GOTO 490
420 PRINT CHR$(7);HOME;VTAB 8;HTAB 13;PRINT I;"---BAD!";PRINT;PRINT
430 GOSUB 540
440 HTAB 10;PRINT D(7);" ";D(6);" ";D(5);" ";D(4);" ";D(3);" ";D(2);" ";D(1);" ";D(0)
450 HTAB 7;PRINT"16-";B(7);" ";B(6);" ";B(5);" ";B(4);" ";B(3);" ";B(2);" ";B(1);" ";B(0)
460 PRINT;PRINT;PRINT
470 HTAB 8;PRINT"1-";A(0);"-"A(1);" "A(2);" "A(3);" "A(4);" "A(5);" "A(6);" "A(7)
480 HTAB 10;PRINT C(0);" ";C(1);" ";C(2);" ";C(3);" ";C(4);" ";C(5);" ";C(6);" ";C(7)
490 POKE 49309,0;POKE 49311,0;POKE 49308,0;POKE 49310,0;POKE 49309,4;POKE 49311,4
500 GET A$;GOTO 70
510 HOME;VTAB 10;HTAB 15;PRINT "[END]"
520 END
530 Z=Y2*256+Y1;RETURN
540 FOR T=7 TO 0 STEP -1
550 A(T)=1;B(T)=1;C(T)=1;D(T)=1
560 IF AX-2^T<0 THEN A(T)=0;GOTO 580
570 AX=AX-2^T
580 IF BX-2^T<0 THEN B(T)=0;GOTO 600
590 BX=BX-2^T
600 IF AD-2^T<0 THEN C(T)=0;GOTO 620
610 AD=AD-2^T
620 IF BD-2^T<0 THEN D(T)=0;GOTO 640
630 BD=BD-2^T
640 NEXT T
650 RETURN

```

# 中华学习机调制伴音电路

武汉市解放大道200号(430030) 李永和

中华学习机的扬声器接口电路功率小,音量不可调。为此,设计制作了调制伴音电路,使声、像信号一并从电视机的天线插孔输入,调节电视机的音量电位器,可任意改变音量。

电路原理图如图1,该电路为改进型变压器反馈式振荡电路,反馈变压器B可采用任何型号电视机的6.5MHz伴音中周,变压器在电路中起正反馈作用,频带可似为足够宽。6.5MHz三端陶瓷滤波器作为选频网络,可等效为Q值极高的LC并联谐振回路,该器件为电视机伴音选频回路所采用的新型元件,故作为调制伴音的振荡选频网络具有免调试功能。反馈信号从三极管Q1的基极输入,信号幅度要求不大,起振容易。电容C2对6.5MHz振荡信号为短路,对声频信号为开路,二者经过Q1的非线性变频,幅度调制作用,从变

器B的次级输出为调频—调幅波,由于电视机伴音解调电路有限幅功能,该电路可视为调频伴音电路。电阻R4限制负载过重,可有效防止电路停振。

电路调试极为简单,将变压器B的次级L2短路调节偏值电阻R1使电路的总电流为2mA左右,去掉短路线后,总电流上升,则说明电路起振,(一般可上升到4mA左右),否则对调L2两端点,使其为正反馈。电路装入中华学习机中,不需要屏蔽与学习机的连接见图一,虚线框内为计算机相关的接口电路。在计算机进入中文拼音状态,按英文字母任意键,使之连续发出“嘟嘟”声,将电视机音量电位器置较小音量,调整磁芯,使电视机伴音的噪音最小。(磁芯调整如否,均有“嘟、嘟”声)。

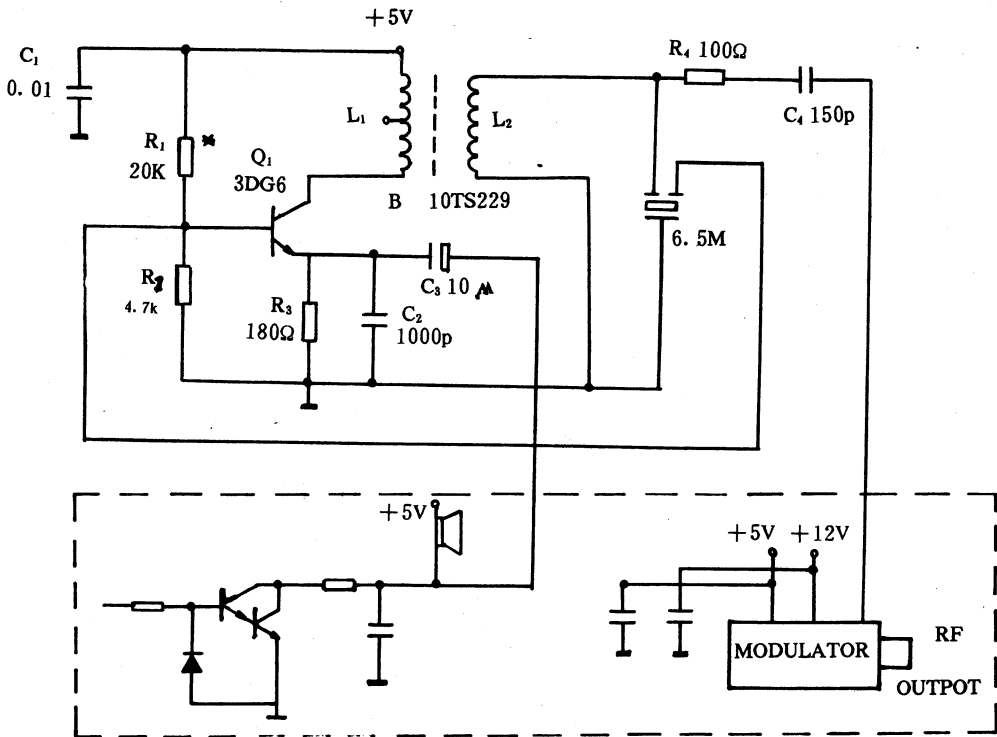


图1



# FBASIC 语言的游戏程序编程技巧

## 第四讲 游戏程序的设计过程(上)

山东苍山机械电子化学工业局(277700)于春

我们知道机械行业开发一种新产品,首先要进行市场需求调查,根据市场调查情况进行产品性能需求分析,以确定产品的功能要求;然后根据功能要求进行图纸、资料和各种技术细节的设计;在图纸资料完善的基础上再进行制造和装配;装配完成后还要进行调试、运行试验,只有产品完全合格后才能出厂。由此可知,在机械工业中最终的产品不仅仅是机器本身,它还包括一整套的技术资料(图纸、技术说明书、使用说明书、维护手册等)。归纳起来一个产品的开发共分需求分析、设计、制造、调试和运行五个工程阶段。

对于微机软件的开发,电脑专家们也提出了工程化管理方案,即也要经过需求分析、设计、编程、测试和运行五个阶段。一项软件产品要包括五个组成部分:系统说明书、模块说明书、程序清单、测试文件和使用维护手册等完备的技术资料。前四个阶段称为开发期,最后一个阶段称为运行期。可见前三讲我们仅讨论了游戏程序设计过程的第二、三两个阶段。当然大多数读者都是业余爱好者,设计游戏程序也仅是兴趣爱好,不是搞盈利性的软件开发,市场调查分析可不予考虑。但分析阶段形成的书面资料——系统说明书还是必需的。因为它是设计、编程、测试阶段的依据。为系统起见,下面我们仍以软件开发的五个阶段为基础,以“成龙救金凤”游戏为例,介绍游戏程序的设计过程。

### 一、分析阶段

首先必须认真地分析开发该软件系统的目的,要选择恰当的游戏类型,要确定游戏的具体功能目标;其次要规划好游戏的玩法,游戏的参加者可有几个对手,电脑是否充当对手;还要决定是否奖励优秀、处罚不好的游戏者等等;然后确定游戏的功能目标。“成龙救金凤”游戏的功能目标如下:

1. 游戏开始,出现金凤被山鹰叼走的画面。
2. 以后各场的开始画出七层楼房。九个藏宝箱散落在七层楼中。成龙位于楼底层左边入口处。金凤在七层楼顶右上角。妖怪在3~7层楼之间。
3. 游戏者使用1号操纵器的方向键控制成龙运动。运动中伴有脚步声。
4. 成龙必须按电脑约定的顺序依次打开九个箱子。每打开一个加10分,积分达到90分时游戏者胜。救人成功。奏乐。显示救出金凤,两人拥抱回家的画面。
5. 当游戏者被妖怪抓住或者时间已到而尚没有打开九个箱子则发声,游戏结束。转结束或继续选项画

面。

上述五项功能目标就是一份简单的系统说明书。

### 二、设计阶段

设计阶段要求设计出符合系统说明书要求的软件系统方案,并从几个方案中优选出较好的方案。方案确定后,先将系统划分成几个模块,并考虑处理好模块间传递的数据与类型,模块间的调用关系,选择出较好的模块结构图,并写出各模块的功能说明书。另外,在模块结构图中必须对所有模块编号,简单说明模块的功能,必要时还要标出每个模块的输入、输出内容。“成龙救金凤”游戏的模块结构图见图一。图一中A为管理程序;B为游戏开始时调用的模块,以后每场均不再调用;C、D、E为每场游戏开始时调用的模块;F为救人成功后调用的模块;B、C、D、E、F均为条件调用模块;G、H、I为循环调用模块,这三个模块构成游戏过程;K为结束处理模块,仅在时间到没有打开箱子、成龙被妖怪抓住和救人成功三种情况下才调用。由于各模块功能较简单,已在模块结构图中标出,不再分模块作功能说明。

### 三、编程阶段

本阶段的任务就是为每个模块编写程序,将模块说明书转换成用某种程序设计语言书写的源程序。为使读者能够编写逻辑上正确、结构合理、层次分明、思路清晰、便于阅读和理解的程序,我们介绍由顶向下、逐步求精的程序设计方法。其基本方法是:以模块说明书为基础,对每个模块的抽象算法由粗至精(即由抽象到具体),逐步精确化,逐步扩充细节,直到成为一个能够在机器上执行的程序。

#### 1. 模块的逐步求精

逐步求精要遵循以下四点:

- (1)以合理的程序结构为目标(程序结构清晰、易读、易理解、易修改)。
- (2)严格由粗到精的顺序。关于细节的决定尽可能往后推迟,以避免整体结构与局部细节的纠缠,使细节的修改不影响上一级的抽象结构。因而程序的易改性好。
- (3)程序结构精细化的同时进行数据结构的精细化。
- (4)一边编写程序,一边检查其正确性,以提高程序的可靠性,减少返工。



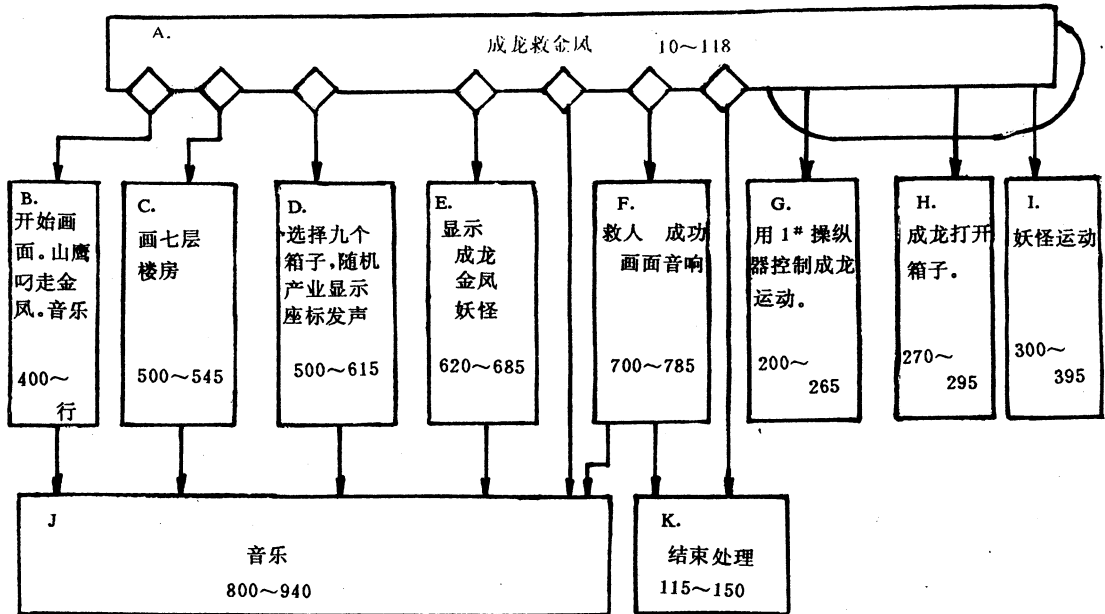


图1 模块结构图

## 2. 用子程序做公共模块

在程序的编写过程中,要注意把许多模块都调用的公用模块作为子程序来编写,以便于调用。对公用模块,必须高质量地编写好程序。这样既可使主程序的逻辑结构清晰,又可使程序的可靠性和效率提高。

## 3. “成龙救金凤”游戏的详细设计。

首先对照模块功能分别设计出各模块的流程图,然后根据流程图编写出详细程序。

### (1)模块 B:

画出开始画面。时间选在下午太阳刚刚落山,残霞满天。暗红色的背景。一条长长的路,路边有树,天空飘着白云。在悠扬的音乐声中,成龙与金凤手拉手从右向左悠闲散步,当走到白云下方时,从云中冲出埋伏的山鹰,叼走了金凤。天空变晴,成龙尾随狂奔,音乐急促激烈。流程图如下:

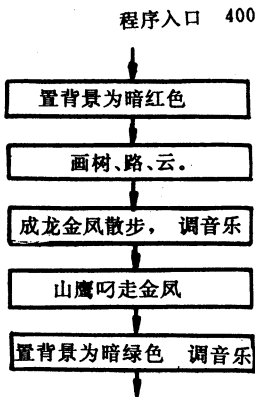


图2. 模块 B 流程图

对照图2写出详细程序。行号分配400

```

400 REM "BEGIN PICTURE"(开始画面)
405 CG. 0,1;SP. O. ;CLS:PAL. B 0,8,48,47,47
410 F. I=0 TO 27;LOC. I,12:P. CH. (197);N. (画路)
415 LOC. 10,2:P. CH. (216);CH. (218);LOC. 8,3:P. CH.
(216);CH. (217);CH. (218)(画白云)
420 LOC. 1,8:P. CH. (208);LOC. 0,9:P. CH. (249);CH.
(203);CH. (251);LOC. 1,10:P. CH. (238);LOC. 1,11;
P. CH. (212)(画树)
425 LOC. 2, 16; P. "CHENG _LONG _SAVES _JIN
 _FENG";LOC. 8,20:P. "BY _ _ _ YU _CHUN"
430 DE. M. (4)=SP. (0,7,8,90,0,3);POS. 4,220,100;M. 4
(成龙)
435 DE. M. (5)=SP. (1,7,8,80,0,3);POS. 5,230,104;M. 5
(金凤)
440 PL. "M0V12M1V12Y1T4;M0V9T4;M1V2T4":FF=1;
GOSUB800(散步音乐)
445 DE. M. (6)=SP. (15,5,1,23,1,0);POS. 6,88,48;M. 6
(鹰)
450 IF CR. (6)<>5 T. 450
455 DE. M. (5)=SP. (1,2,1,58,0,3);POS. 5,70,104;M. 5
460 DE. M. (6)=SP. (15,2,1,55,0,0);POS. 6,76,96;M. 6
(叼走金凤)
465 PAL. B 0,10,48,47,47(置暗景)
470 DE. M. (4)=SP. (0,3,1,94,0,3);POS. 4,60,100;M. 4
475 GOS. 900;ERA 4(急促音乐)
480 RE.

```

### (2)模块 C:

画七层楼房。用187号背景图案画地板,用210号画楼梯。使用变量 X、Y、B\$、J\$。流程图见图3。对应程序如下:

```

500 REM"BUILDING"(画楼房)
505 B$ =CH. (187);J$ =CH. (210);J$ =J$ + "□□□"
 +J$
510 F. X=1 TO 21 ST. 4
515 F. Y=2 TO 20 ST. 3
520 LOC. X,Y:P. B$;B$;B$;B$;IF Y>19T. 530
525 LOC. X,Y+1:P. J$;LOC. X,Y+2:P. J$
530 N.;N.
535 F. I=2 TO 20 ST. 3
540 LOC. 25,I:P. B$;N.
545 RE.

```

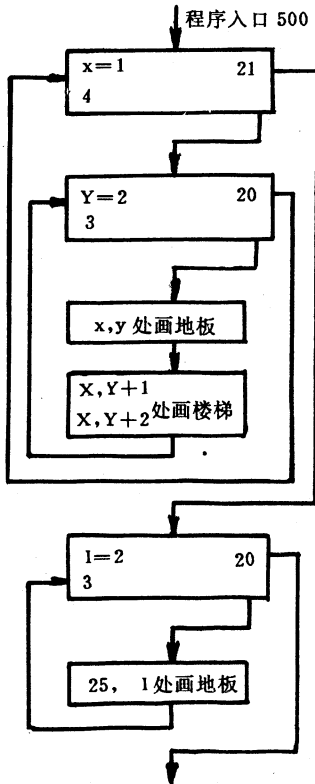


图3 模块 C 流程图

### (3)模块 D

选用190~197、212~219、244~251 27个背景图案为随机选用的藏宝箱。把每次选定的九个图形按顺序赋值给 PS\$。随机产生在楼板上的九个座标，打印九个图形。使用变量：

PS\$：装入九个藏宝箱的图形。

P1、P2：选定藏宝箱的图形。

PL：PS\$ 的字符长度。

SX、SY：藏宝箱的打印座标。

S\$：藏宝箱位置确定辅助变量，由它决定各藏宝箱打印在楼板上。

PS：藏宝箱的序号。

流程图如下：

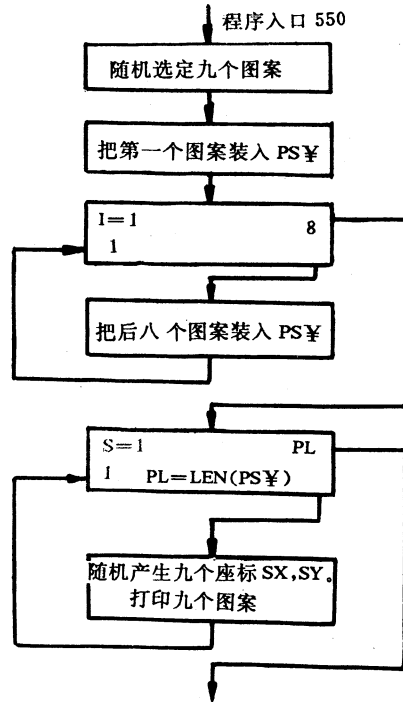


图4 模块 D 流程图

对应程序清单如下：

```

550 REM"BOX"
555 P1=RND(3)+5
560 C=80;IF P1=7 T. C=90
565 P2=P1 * 22+C;PS$ =CH. (P2)
570 F. I=1 TO 8
575 PS$ =PS$ +CH. (P2+I);N.
580 PL=LEN(PS$);PS=1
585 F. S=1 TO PL
590 SX=2+RND(24);SY=2+RND(18)
595 S$ =SCR$(SX,SY+1)
600 IF S$ ="□" OR INS. (PS$,S$)>0 OR (SX-1)
 MOD4=0 T. 590
605 LOC. SX,SY:P. MI. (PS$,S,1)
610 N.
615 RE.

```

### (4)模块 E：

在(200,16)处显示金凤。在(12,176)处显示成龙。在(92,32)~(220,104)之间显示妖怪。使用 E(KN-1)数组装妖怪的个数。欲增加游戏难度，可调整 KN 的数值。

该模块较简单，故略去流程图，直接写出详细程序。程序入口为620。

```

620 REM"SHOW SPRITE"(显示卡通)
625 LOC. 12,0:P. "□□□□□□□□□□"
630 DE. SP. 3,(2,1,0,0,0)="$%&'";SP. 3,200,16

```

(下转第48页)



# 笔记本型电脑技术和市场分析

哈尔滨建材工业学校(150036) 赵广恩

到目前为止,从美国日本以至于亚洲,欧洲等市场上,笔记本型电脑产品的种类已不胜枚举,随着计算机技术的发展,其竞争也将随之加剧,而竞争的结果将导致技术的更加完善和强化,性能价格比优异,从而也将会被更多的用户所接受。

从技术角度上讲,它的体积、主机、显示器,内存, I/O 接口及运行速度很多都可与 PC 机相媲美。也正是这一点,吸引了众多使用者的青睐。虽然目前多数的用户还只局限于房地产业、保险业、工程人员的文书处理类,但就其发展趋势而言,其市场是不可低估的。

目前我国市场上也相继出现了一些笔记本型电脑,主要来自美国、日本和港台地区。由于其价格偏高,销路尚不可乐观。

台湾销售额占世界市场22%之多,仅次于日本占第二位,尽管如此也还有许多隐忧。这其一是来自零部件价格及货源状况的影响。诸如 LCD、CPU 和硬磁盘机等主要组件依然主要由日、美厂商控制,从而在某种

程度上阻碍了生产能力,速度和技术的发展。其二是笔记本型电脑的生命周期相当短,人们常用所谓的“六、三、三”来比喻开发一个新机种要六个月,密集销售期三个月、之后便是三个月清仓出库。仅以去年的386sx-16为例,即使是配备了20MB 硬盘机也乏人光顾了。因为386SL,DX 和486的新机型问世,使其受到了较大的冲击、而今,掌上电脑又开始推出、其价格和使用范围使笔记本型电脑在某种意义上受到一定冲击。

在欧美市场许多消费能力较低的学生也开始关心起此类电脑,而笔记本型电脑已开始步入消费性产品的行列,人们对电脑知识也不断丰富,基于某种原因,人们对电脑性能价格比的要求也越来越高,从而使得笔记本型电脑的技术水平也不断提高。

预计全球笔记本型电脑的需求量到九五年有可能达到七百二十万台左右。其前景是乐观的,但发展中的困难不可忽视,需要各界不断努力,追求产品的质量和性能更优异和完善。(下表为几种笔记本型电脑指标)

公司名称	AST ASIA	AST	ABC COMPUTER CO. LTD.	NCR(H. K) LTD	MUL MI CORWARE USA LTD.
型号	AST PREMIU EXEC 386SX/20	PE 386SX-25C	ABC NYCOM N440i	NCR 3120	MITAC 3026E
中央处理器	20MHz 386SX	25Hz 80386SX	20MHz 386SX	20MHz 386SX	20MHz 386SX
协处理器	387 SX 插口		387 SX 插口	387 SX 插口	387 SX 插口
存储设备	FDD 3.5" 1.44MB×1	3.5" 1.44MB×1	外部扩展	3.5" 1.44MB×1	3.5" 1.44MB×1
	HDD 2.5" 20MB×1	60MB×1	2.5" 40MB×1	30/60MB×1	30/60MB×1
显示器	640×480 CCFT 32 灰度级 VGA, LCD	32 级灰度级 VGA, LCD	3 个双 640×480 带背景光 LCD	VGL LCD 16 级灰度级	640×480VGA LCD 32 级灰度级
I/O 接口	一串,一并,一监视器口,一个键盘口	一串,一并,一鼠标器口	一串,一并,一外部软盘机等	一串,一并,一个鼠标口,一个视频口	一串,一并,一 PS/2 键盘口,一 VGA 口
键盘	82/38 键		86 键全尺寸	85/86 键	81/82 键
体积(mm) <sup>3</sup>	280×289×57	280×220×55	210×280×30	210×296×44	221×294.6×58.4
重量	3.2(Kg)	71bs	2Kg	2.5Kg(包括电池)	3.4Kg
电池使用时间	3 小时		4 小时	2 小时	3 小时
随机存取存储器	2MB~8MB	4MB~8MB	2/4MB	1/5MB	1MB~2/5MB

# 用软件对软驱进行简单读写检测

北京海淀区 李晓中 麻佳洛 谢静 张景生

在 IBM-PC/XT 或 AT 使用中,有时在进行 DIR 目录显示时正常工作,但在进行读盘/写盘操作时出错;有时候读写正常,但有时候又连续出错。这时,用硬件检测或执行系统检测程序却发现驱动器工作正常。究竟是驱动器故障还是软磁盘损坏,一时难以判别。笔者在实践中编写了一段读盘检测小程序,以帮助维修人员了解目前系统盘驱动器的故障所在并认识故障的性质。

## 检测程序:

本程序采用汇编语言编写,使用系统的 INT13 中断调用,对驱动器进行逐面、逐道、逐扇的读/写,并在读/写过程中不断检测 INT13 调用的返回参数,以判断是否有错。若有错则继续分析是数据读/写错还是控制信号或其它类型错,本文只给出了数据读出和其它类型错,读者也可以扩充分析其它更多的类型错(利用 INT13 调用的返回参数 AH 的定义进行分析)。

使用此程序可以判别区分驱动器故障和软磁盘故障。也可以具体地检测到软盘的哪些磁道有问题。

比如,在实践中,发现一驱动器 0 面读/写全部正常,而 1 面 35 道以后的各道扇区读/写发生数据类型错,可以分析为两方面故障:①因软驱磁头在高磁道区的定位不准;②读出电路故障,对内道的高密度信息因信号幅度变化而出现的峰点飘移造成读出错。

可以从以上两个方面进行检查维修。

下面给出检测程序:

```
1:STACK SEGMENT PARA STACK 'STACK'
2: DB 128 DUP(0)
3:STACK ENDS
4:DATA SEGMENT PARA PUBLIC 'DATA'
5:DAT1 DB 0DH,0AH,20H,20H,20H,'
 欢迎使用磁盘诊断软件',20H
6: DB '空军电讯工程学院电子技术
 系',0DH,0AH
7: DB '软件编制:设备教研室
 1991年12月09日',0DH,0AH
8: DB 0DH,0AH,0DH,0AH,'请输入
 驱动器号 A/B/C'
9: DB 0DH,0AH,'$'
10:DAT2 DB '输入驱动器号错!请重新输
 入!',0AH,0DH,'$'
11:DAT3 DB '本驱动器操作完全正确!',
 0DH,0AH,'$'
12:DAT4 DB '驱动器操作有错!请检查维
 修!',0DH,0AH,'$'
13:DAT8 DB '读数据错!',0DH,0AH,'$'
```

```
14:DAT9 DB '其它类型错!',0DH,0AH,'
 $'
15:CLOR DB 1
16:ERRF DB 0 ;存错误码
17:SORT DB 01H ;扇区号
18:CRLF DB 0DH,0AH,'现在检测驱动器,
 请稍等!',0DH,0AH
19: DB '用 CTRL-C/CTRL-Break
 键可随时退出!',0DH,0AH,24H
20:KEY1 DB 00H ;驱动器号暂存
21:TRACK DB 00H ;磁道号计数器
22:HEAD DB 00H ;磁头(0/1)计数
 器
23:AAASMG DB 0DH,'正在检测',00,'面',
 00,00,'道',00,'扇区!',20H,
 20H,'S'
24:ERRLB DB 00 ;操作有错标志(非零错)
25:BUF DB 512 DUP(0)
26:DATA ENDS
27:CODE SEGMENT PARA PUBLIC 'CODE'
28:START PROC FAR
29: PUSH DS
30: MOV AX,0
31: PUSH AX
32: ASSUME CS:CODE,DS:DATA
33: ASSUME SS:STACK,ES:DATA
34: MOV AX,DATA
35: MOV DS,AX
36: MOV ES,AX
37: MOV AX,STACK
38: MOV SS,AX
39: MOV AX,0003H
40: INT 10H
41: MOV AX,0B00H
42: MOV BX,0001H
43: INT 10H
44: MOV AX,0600H
45: MOV BX,4F00H
46: MOV CX,0
47: MOV DX,184FH
48: INT 10H
49: MOV CX,OFFSET DAT1
50: CALL XS
51: KEY:MOV AH,01H
52: INT 21H ;接收键盘
53: AND AL,0DFH
54: MOV CRLF+16,AL
```

```

55: OR AL,20H
56: CMP AL,'a'
57: JZ DISKA
58: CMP AL,'b'
59: JZ DISKB
60: CMP AL,'c'
61: JZ DISKC
62: MOV CX,OFFSET DAT2
63: CALL XS
64: JMP KEY
65:DISKA: MOV KEY1,00H
66: JMP NEXT
67:DISKB: MOV KEY1,01H
68: JMP NEXT
69:DISKC: MOV KEY1,80H
70:NEXT: MOV CX,OFFSET CRLF
71: CALL XS
72:;-----

```

—功能程序如下

```

73: JMP NEXT1
74:NN1: INC AH
75: JMP NN0
76:NEXT1: MOV AL,HEAD
77: OR AL,30H
78: MOV AAASGMG+9,AL
79: MOV AL,TRACK
80: MOV AH,00H
81:NN0: SUB AL,10
82: JNC NN1
83: ADD AL,10
84: OR AL,30H
85: OR AH,30H
86: MOV AAASMG+12,AH
87: MOV AAASMG+13,AL
88: MOV AL,SORT
89: OR AL,30H
90: MOV AAASMG+16,AL
91: MOV,CX,OFFSET AAASMG
92: CALL XS
93: MOV AH,02H ;功能号(读盘)
94: MOV AL,01H ;传送扇区数
95: MOV BX,OFFSET BUF ;缓冲区首址
96: MOV CH,TRACK ;磁道号
97: MOV CL,SORT ;扇区号
98: MOV DH,HEAD ;磁头号
99: MOV DL,KEY1 ;驱动器号
100: INT 13H ;读盘操作
101: JC ERROR1
102:NNN: INC SORT
103: MOV AL,SORT
104: CMP AL,0AH
105: JC NEXT1
106: MOV SORT,01H
107: MOV CH,TRACK
108: CMP CH,39
109: JZ CEND1

```

```

110: INC TRACK
111: JMP NEXT1
112:CEND1: MOV AL,HEAD
113: CMP AL,1
114: JZ CEND
115: MOV TRACK,0
116: MOV HEAD,01H
117: JMP NEXT1
118:CEND: MOV AL,ERRLB
119: CMP AL,0
120: JNE ERROR
121: MOV CX,OFFSET DAT3
122: CALL XS
123: RET
124:ERROR1: AND AH,10H;检查 D4位=1否?
125: JNZ DERR
126: MOV CX,OFFSET DAT9;非数据错!
127: CALL XS
128: JMP NN00
129:DERR: MOV CX,OFFSET DAT8;数据类型错!
130: CALL XS
131:NN00: JNC ERRLB
132: JMP NNN
133: ERROR;MOV CX,OFFSET DAT4
134: CALL XS
135: RET
136: ;-----
137:XX PROC NEAR
138: MOV DX,CX
139: MOV AH,09H
140: INT 21H
141: RET
142:XS ENDP
143: START ENDP
144: CODE ENDS
145: * END START
* E

```

要了解最新的电脑信息  
 要准确地选购电脑  
 要掌握电脑入门的方法  
 要学会各种软件的使用  
 要快捷地修复电脑故障  
 要您的电脑产品迅速占领市场  
 要您的电脑产品艺术地再现

请订阅 **《电脑》** 杂志

请在 **《电脑》** 杂志刊登广告

《电脑》杂志为月刊刊号CN44—1188，邮发代号46—115，全国各地邮电局(所)均可办理订阅。全年12期共18元。

编辑部地址:

广州市华南师范大学 电脑杂志社

邮编:510631 电话:5516911—3273

## 普及型 PC 个人用户软件交流联谊 活动问题解答(十一)

北京中国农科院计算中心(100081)王路敬

### 43. 如果从硬盘启动 DOS 时出现“Error Loading Operating System”错误信息如何解决?

上述错误信息的含义是装 DOS 错。产生这类故障的原因常常是由于 DOS 分区柱面的损坏或者 DOS 分区表 FAT 被破坏,或硬件的读写电路坏等原因造成。

解决这类故障一般使用的方法是执行 FDISK 命令,重新建立 DOS 分区,或者使 DOS 分区起始位置避开 0 柱面,或者重做分区格式化,这是软件排除故障的方法,如果读写电路坏,需要更换硬盘控制器。但这些方法的缺点是往往会造成硬盘大量数据丢失。

如果对硬盘执行 DIR、SYS、CHKDSK 等命令时执行结果,系统发出“File Allocation Error”或“Seek error reading drive C”等出错信息,常常是文件分配表 FAT 损坏而引起的定位错误。DOS 分区的文件分配表有 2 份,其内容完全相同,FAT1 和 FAT2。一般文件分配表损坏往往是 FAT1, FAT2 并没有破坏,为此,可用 FAT2 表恢复 FAT1 表。以长城 0520CH 20MB 硬盘为例恢复 FAT1 表操作如下:

```
A>DEBUG
-L100 2 9 8 ;读 FAT2
-W100 2 1 8 ;写 FAT1
-Q
```

也可以采用下面一种方法,操作如下:

```
A>DEBUG
-A 100
××××:0100 MOV AX,0208;读 FAT2 8个扇区
××××:0103 MOV BX,0200;置缓冲区为 CS:0200
××××:0106 MOV CX,0009;FAT2起始扇区为逻辑9扇区
××××:0109 MOV DX,0080;硬盘0号磁头
××××:010C INT 13 ;磁盘输入/输出中断
××××:010E INT 20 ;程序终结
-G=0100 010E
```

```
××××:0110 MOV AX,0308;写 FAT1
××××:0113 MOV BX,0200
××××:0116 MOV CX,0001
××××:0119 MOV CX,0080
××××:011C INT 13
××××:011E INT 20
-G=0110 11E
-Q
```

还可以将 FAT2 写入一已格式化的空软盘,然后由软盘读出写入 FAT1 所占据硬盘扇区,操作过程与上述相似,读者可试试看。

### 44. 怎样利用 CHKDSK 命令增大磁盘空间?

不管软盘或者硬盘由于多次的进行拷贝或删除文件等操作,常常使得可用的磁盘空间减少。实践证明利用 DOS 的 CHKDSK/F 命令可以对出错的文件定位表进行修复,相对增大磁盘的空间。操作方法如下:

(1)用软盘启动系统,对 C 盘工作:

```
A>CHKDSK C:/F
```

此时无论屏幕上出现多少个(Y/N)选择,均回答 Y。

(2)待上述工作完成后,将凡是扩展名为.CHK 文件删除。

```
A>ERASE C:FILE *.CHK
```

例:长城 286BH 33M 硬盘,实现上述操作前磁盘剩余空间 12597248 字节,操作后磁盘剩余空间变为:12873748 字节。

在这里顺便提一下,CHKDSK 命令加“/V”参数可以将显示在硬盘包括根目录和所有子目录下的全部文件,不论加密与否均全部显示。屏幕显示格式为:

```
C:\子目录名\文件名
```

所有目录下的文件显示完后,屏幕提示:硬盘上有 X 个隐含文件,共有 Y 个目录,总计有 ZZZ 个文件。这样对于用 TREE、DIR 等命令列不出来的隐藏子目录和隐形文件,在 A>CHKDSK C:/V 命令下全部亮相。

### 45. 在实际应用中规定磁盘缓冲区个数的原则是什么?

因为各种应用的目的和实现手段不同,所以,没有一个规定的缓冲区数量对所有的应用都有同样的效果。最佳缓冲区数只能通过不同的值选择直到获得最佳的功能。如果用户用数据库系统,例如 dBASE III 常规定 24 个缓冲区,就可提供较好的效果。如果运行的程序有很多随机读写的记录,例如 BASIC 的随机文件,一般缓冲区数规定要多一些,这样通过加速访问时间提高程序运行的速度,增强应用功能。相反,对于顺序的读写应用,例如读写一个完整的文件,分配一个大的缓冲区没有什么优越性,反而增加内存的开销。所以,如果用户很少需要随机的读和写记录,系统的缺省值就足够了。

### 46. 什么是虚拟盘?有何特点?怎样设置?

PC DOS 3.0 及更高级版本的系统结构设置文件 CONFIG.SYS 中,为 DEVICE 命令增加了安装虚拟磁盘的功能。所谓虚拟磁盘是用计算机主存储器的一部分区域作为存储介质仿真磁盘驱动器,被仿真的磁盘叫虚拟磁盘。

虚拟磁盘有下列特点:

(1) 虚拟磁盘实际进行的只是对内存的操作,因此,虚拟磁盘操作速度快,但要占用一部分内存空间。

(2) 一个系统可配置多个虚拟磁盘。每个都可以用驱动器字母引用。例如,若微机系统已安装了 A、B 两个软盘驱动器和一个硬盘驱动器 C,那么,第一个定义的虚拟磁盘为 D,第二个为 E,等等。

(3) 对每一个虚拟磁盘,用户都可以规定要用的存储区域大小,扇区大小和它所包括的目录项。

(4) 每个虚拟磁盘除了要占指定或约定的内存容量外,还要增加 720 个字节的内存,用于 DOS 对它的管理。

(5) 在未指定的情况下,虚拟磁盘所占的内存区域是从 PC-DOS 的用户可用空间开始的连续区间,如果在定义语句中给定参数 /E,且在主机中装有大约 1MB 内存空间的情况下,虚拟磁盘将装在从 1MB 的边界开始,而不占用用户内存区。

(6) 存放在虚拟磁盘上的文件,当在系统重新启动或掉电时,其盘上的信息要丢失。

建立虚拟磁盘的方法是在系统结构设置文件 CONFIG. SYS 中设置语句 DEVICE = VDISK. SYS,且在启动时 VDISK. SYS 要在启动盘上。命令格式为:

```
DEVICE = [<盘符>] [<路径>] VDISK. SYS [<虚拟盘字节数>] [<扇区字节数>] [<目录数>] [/E]
```

其中 [ <虚拟盘字节数> ]:指虚拟磁盘的大小,以 K 字节为单位,用十进制数表示,缺省值为 64K 字节。其值范围从 1K 字节到微机可用内存的最大值。此值的定义要求:

(1) 如果在安装虚拟磁盘时内存空间已小于 64K, VDISK 发出错误信息,且不安装虚拟磁盘。

(2) 如果指定虚拟盘字节数小于 1K 字节,或小于微机系统可用内存空间,则系统自动按 64K 字节设置。

(3) 如果规定虚拟磁盘大小后,用户可用内存空间小于 64K,系统将自动向下调整虚拟磁盘的大小,使用用户可用内存空间保持 64K 字节。

<扇区字节数>:指可允许的扇区字节数为 128、256、或 512 字节。缺省值为 128 字节。

<目录数>:指虚拟磁盘可以包括的目录数。缺省值为 64。该值允许范围为 2—512。每个目录项所占字节数为 32。如果所定义的目录数乘以 32 恰是所定义的扇区字节数的整数倍,则系统对目录自动向上进行调整。使用时注意,所定义的目录数其中一个不能被用户使用,系统用它保持卷标。

</E> 该参数表示如果在定义虚拟磁盘的语句中给定它,则虚拟磁盘将安装在内存扩展部分中,且其地址是在 1M 以上的空间内。这时,用户可以用空间不受虚拟磁盘的影响,但必须有扩展内存的硬件支持,否则系统无法安装。

例如,装配一个 256 字节的扇区和 32 个目录项的 160K 字节的虚拟磁盘 D,其操作如下:

```
(1) 在启动盘上 CONFIG. SYS 文件中写入语句:
DEVICE = VDISK. SYS 160 256 32
```

(2) 保证在系统启动盘上有 VDISK. SYS 文件存在。

重新启动屏幕上便显示:

```
VDISK Version 1.0 Virtual disk D:
Buffer size: 160KB
sector size: 256
Directory entries: 32
```

用户使用时一定要注意,因为虚拟磁盘的建立是借助于系统内存存储器的一部分空间,虚拟磁盘所有存储的文件在切断微机的电源前(或者重新启动前)必须拷贝到物理磁盘 A 或 B 或 C 盘上,否则数据将全部丢失。

另外,如果在系统启动时屏幕显示错误信息:

```
Bad or missing VDISK
```

应从以下三个方面找原因:

(1) 在 CONFIG. SYS 文件没有找到以 DEVICE = VDISK. SYS 所指明的驱动程序名,所以这时可用 TYPE 命令查看该文件中是否包含了有关虚拟磁盘的命令,如果有,语句格式是否正确。

(2) 断点地址超过了机器范围。

(3) 装入该驱动程序出错。即 DOS 没有安装 VDISK. SYS 驱动程序。这时可进一步查看系统启动盘上有否该程序,如果没有可将 VDISK. SYS 复制到系统盘上。

47. 当从硬盘启动时屏幕出现“Invalid Partition Table”错误信息且死机,怎样用简单的方法排除?

当出现这种现象时不要急忙用低级格式化硬盘的方法解决,否则,要破坏硬盘原有的信息,可以采用下面一种简单的方法:

将 DOS 系统盘插入 A 驱动系统执行外部命令 FDISK。

```
A> FDISK
```

屏幕显示命令的功能菜单:

```
GW Personal Computer
Fixed Disk Setup Program Version 2.00
Great Wall 0520C Computer
FDISK options
Choose one of the following
1. Create DOS Partition
2. Change Active Partition
3. Delete DOS Partition
4. Display Partition Data
Enter choice: [1]
```

选择 3,删除 DOS 分区,然后按 <ESC> 键,返回主菜单后再选择 1,建立 DOS 分区。

待上述操作完成后,系统将提示用户返回 DOS,再从硬盘启动即故障已经排除。

有时候系统不能从硬盘启动,不显示任何信息转到软盘工作,但从软盘进入 DOS 后,可以转到硬盘工作,这种故障也可以采用上述操作解决之。

48. 无法进入硬盘与系统配置有否关系?

(下转第 22 页)

## 软件服务

### C—DBAG 中文数据库应用生成器

F/D \* AG V5.0

零售价:3900元

使用本系统只需按照提示输入用户需求,不需编程,各种管理软件,在功能变动时可任意改动,适用于网络及单用户环境。

功能:生成 dBASE+FoxBASE+源程序;生成任意格式报表;自动生成数据字典及文档;工程图形生成与管理。

组成:下拉/弹出式菜单生成器;查询模块生成器;数据录入与维护模块生成器;数据库文件结构生成器;彩色统计图形模块生成器;任意格式报表生成器;数据字典与文档生成器;

灵科汉卡

零售价:1200元

西文软件无需汉化即可直接使用中文,直接写屏,支持 1024×768,640×480,640×350 图形方式。支持中文排版功能。中文简体与繁体,中文与西文之间壹键转换。支持针式打印机、激光打印机及鼠标。精密宋、楷、黑、仿宋体平滑放大输出。能识别中西文窗口。显示字库不占用内存,且显示速度大大提高。

灵科加密器

零售价:180元

(上接第42页)

635 T=0;F=0;KN=2;KX=0;KY=0;KD=0;TI=800

(时间赋值为800)

640 LOC.0,22;P.“□□□SCORE□”;T(打印得分。)

645 DE.M.(7)=SP.(0,0,2,4,0,RND(4));POS.7,12,168;M.7

650 DIM E(KN-1)

655 F.F=0 TO KN-1(由 KN 决定妖怪数量)

660 DE.M.(F)=SP.(11,1,2,4,0,RND(4))

665 SX=10+RND(17);SY=1+RND(10)

670 IF SCR\$(SX,SY)、“□” OR SCR\$(SX,SY-1)、“□” G.665

675 POS.F,SX\*8+12,SY\*8+16;M.F

680 NEXT

685 RE.

调整635行的 KN 可以增加妖怪的数量。KN 最大值为7。

(5)模块 G:

用操纵器控制成龙上、下楼和左右移动,与例四中的模块 D 大同小异。直接写出程序。程序入口为200,使用变量:

K:1# 操纵器键值。

可加密任何 EXE 和 COM 文件,也可对任何源程序实施加密,如 dBASE,FOXBASE 等,加密文件个数不限。安装方便,只要外插于打印机接口上,且对打印机工作无丝毫影响。加密手段方便,保密性极高。具有强有力的反跟踪功能,使解密成为梦想。

灵科加密卡

零售价:450元

具有灵科加密器的一切功能,并且做到了真正的软硬件加密结合的功能。具有开机口令功能,并且可以方便地修改口令或关断开机口令功能。彻底避免了计算机的非授权使用,减少了计算机病毒的感染机会。本卡有2K字节可供编程人员进行二次开发。不用任何其它硬件设备,只需本公司提供的软件工具或函数即可完成向卡内读写加密数据或程序,写入次数以上五万次。本卡极适合保密性高且成本较高软件加密或大、中型 MIS 系统加密使用。

灵科 MIS 管理新工具

程序结构管理及菜单生成系统

零售价:500元

辅助生成结构图,程序文档;自动生成 WINDOWS、C 语言、dBASE 菜单源程序;支持鼠标操作,模拟显示。

程序流程图管理系统

零售价:500元

支持顺序判断(IF..ELSE)、循环(WHILE)、选择(SWITCH)逻辑结构图及相应文档;自动生成 C 语言逻辑调用程序;支持鼠标操作;支持6层嵌套。

经销:中国电子器材华北公司

地址:北京市海淀区万寿路西街五号

通信地址:北京144信箱

邮政编码:100036

联系人:石立军 魏国

电话:81.1810 81.0920

本公司经营计算机及外设通信设备,欢迎来电函联系。

SX,SY:BG 面的位移量。

PX,PY:成龙在 BG 面的座标

S.:成龙的运动方向。

S\$,S\$1:成龙周围的背景数值。

程序清单如下:

200 REM“CONTROL”(控制成龙运动)

205 S=0;SX=SY

210 K=STI.(0);IF K=0 G.265

215 IF K=1 T.S=3;SX=1

220 IF K=2 T.S=7;SX=-1

225 IF K=4 T.S=5 SY=1

230 IF K=8 T.S=1;SY=-1

235 PX=(XPOS(7)-12)/8;PY=(YPOS(7)-16)/8

240 S\$=SCR\$(PX+SX,PY+SY+1);S1\$=SCR\$(PX+SX,PY+SY)

245 IF S=0 OR S\$“□” G.265

250 DE.M.(7)=SP.(0,S,1,4,0,RND(4));POS.7,PX\*8+12,PY\*8+16;M.7

255 IF S=3 OR S=7 T.PL.“O0#F0RO1#F”;G.265(左、右运动的声音)

260 PL.“O1#F0RO0#FR”(上、下楼的声音)

265 RE.

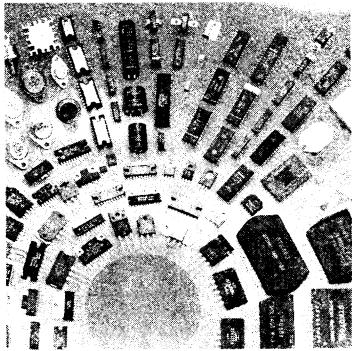
(未完待续)





ISSN 1000-1077  
**電子**  
**與**  
**電腦**

12



• ELECTRONICS AND COMPUTERS

一九九二年  
 总期第93期

# 電子與電腦

## 目 录

### • 综述 •

- 国外消费电子业 ..... 吕问黎(2)
- 应用计算机网络辅助高中物理教学  
..... 肖美 王天谔 李燕萍(3)

### • PC 用户 •

- 也谈 FoxBASE+弹出式菜单的实现方法  
..... 王瑞华(5)
- C语言如何读取 dBASE 的库文件 ..... 李晓华(6)
- 如何实现题库系统的图文并茂 ..... 裴伟东(7)
- 一个简单实用的磁盘加密程序 ..... 任绥海(9)
- 2.13HZ 汉字系统安装虚拟盘库新探 ..... 廖 凯(10)
- 怎样从死循环程序返回 DOS ..... 方 晨(10)
- BASIC 实现“卡拉—OK” ..... 于龙滋(10)
- 实现 1.2MB 软磁盘之间直接拷贝简法 ..... 何崇乐(12)
- 兼容机怎样运行 BASICA.COM ..... 邓文超(13)
- 香港病毒的消除和防范 ..... 孟 桥(14)
- 谈数据文件“死而复生”的技巧 ..... 李瑛彬(15)
- 数据自动存盘程序 ..... 汪海波(16)
- PC 机软件加解密技术剖析(续) ..... 李文亮(18)

### • 学习机之友 •

- 推荐一个扩展 BASIC 语句——拆字串语句  
..... 王 凯(20)
- 也谈内存信息的打印 ..... 张 亨(22)
- 增强中华学习机的音响功能 ..... 杨云霄(23)

- 计算各类债券储蓄收益利息 BASIC 程序  
..... 陶文庆(24)
- 哥德巴赫猜想 BASIC 程序的改进 ..... 卢良红(24)
- 数组的动态删除 ..... 冯端品(25)

### • FORTH 语言讲座 •

- 第一讲 概述 ..... 丁志伟(26)

### • 学用单片机 •

- 模/数、数/模转换实验(续) ..... 张俊谔(29)

### • 电脑巧开发 •

- 中华学习机巧测电容 ..... 邓本富(33)
- 家庭如何配置中华学习机 ..... 聂铁轮(34)

### • 电脑游戏机 •

- 第四讲 游戏程序的设计过程(中) ..... 于 春(35)

### • 维修经验谈 •

- LQ—1600K 打印机接口专用集成电路 M54610P  
的简易修复方法 ..... 赵继文(38)
- GW286 计算机硬盘控制卡故障处理一例  
..... 杨远成(40)
- TP801 单板机的常见故障 ..... 花 成(40)

### • 读者联谊 •

- 普及型 PC 个人用户软件交流联谊活动问题解答(十二)  
..... 王路敬(41)
- 培训消息 ..... (43)
- 1992 年总目录 ..... (45)

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

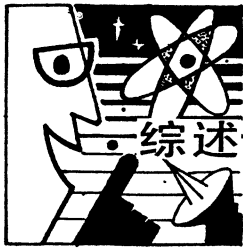
责任编辑:张 丽

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



# 国外消费电子业

吕问黎

1991年的美国消费电子业领域中有多项技术上或商业上的成功,其中视频压缩技术和位率压缩技术是最引人注目的两项。它们已经成为美国标准高清晰度电视传输系统的最佳的备用技术。有线广播的管理者们也已做好准备把这两项技术运用到传统的NTSC制式系统中去,以便在现有的电缆条件下增加新的电视频道。

数字化音响技术也得到了广泛的开发。这项技术可以消除微弱的杂音而保留较明显的声音信号。飞利浦公司就使用它的新产品“数字式压缩盒”系统去掉了人耳听不到的声音信号,以减少高保真声音存贮的位率。索尼公司在它的“微型唱片”系统中也这样做了。这项技术还将被运用到数字化广播和个人电脑电子通信领域中的多媒体应用产品方面。

## 高清晰度电视

几种先进的电视传输系统在技术上的进展终于在1991年7月弗吉尼亚州亚历山德拉的“先进电视测试中心”展示在人们面前。第一个被测试的“先进兼容电视系统”(ACTV)是由戴维·斯纳夫研究中心、NBC电视网、飞利浦电器公司和汤姆逊电器公司联合发起和研制的。这几家公司成立了一个支持“先进兼容电视系统”的国际财团:先进电视研究财团。这个财团还同时正在发展数字化的无线电与电视联播的高清晰度电视系统,这个系统也正在准备参加测试。其他已经测试过的系统还有Digi Cipher系统和日本广播公司的Narrow Muse。前者是通用仪器公司、马萨诸塞技术研究所等机构组成的美国电视联盟(HTVA)研制的。按照计划,今年还要参加测试的有Zenith电子公司和AT&T微电子公司联合开发的DSC高清晰度电视系统以及麻省理工学院与美国电视联盟生产的“ATVA高级系统”。

这些参加测试的系统必须能够完整地传播电视音响和画面,操作必须是实时的。必备的硬件包括电视台方面对电视信号进行编码和调制的全部部件,以及电视机方面的解码器和解调器。

每个系统在实验室内的测试时间约为7周,包括对共有频道测试(非常邻近的地区使用同一频道却装载不同的节目)、相邻频道测试(两个接近的频道装载不同的节目)以及对在飞机上电视图像的颤动的测试等。实验室测试主要集中在每个系统在传输信号过程中的缺点方面,如各种类型的电子噪声、环境噪音、由于信号传输路线的地面状况和建筑物的影响而产生的

多路传输中的重像现象等。系统还要在不同的界面和干扰条件下录制数字化的录像带,然后由非专业人士来放映,以做出主观的测试。

实验室测试完毕以后,联邦通信委员会的顾问委员会计划对占优势的系统进行实地测试。把这两项测试结果加起来,再考虑到法律、经济、公共政策等方面的因素,联邦通信委员会将在1993年提出一个新的作为标准的系统。

在欧洲和日本,高清晰度电视也正在蓬勃发展之中。欧洲委员会去年颁布了一个适用于卫星广播的指导性文件,鼓励使用D<sub>2</sub>-M<sub>cc</sub>标准(这是一个提高分辨率的过渡性的标准),作为走向高清晰度电视的一步。但是,据1991年9月的《经济时代》报道,虽然法国和荷兰政府支持这个系统,法国的汤姆逊、荷兰的飞利浦和芬兰的Nokia三大公司却持反对态度。

91年10月起,日本开始每天播放8小时的超过一般分辨率的电视节目;宽屏幕的NTSC制式电视机则作为日本提高分辨率努力的另一部分投放了市场。这些过滤性措施之后,也必然是数字化的高清晰度电视。

## 提高电缆容量

在美国,提高有线电视电缆的频道容量的倾向正在发展。例如在纽约昆斯区的白石(Whitestone),“时代华纳有线电视”已经把550兆赫75频道的有线电视网换成了一个1000兆赫150频道的系统;在纽约的Flushing地区的总台与白石分台间安装了一条光导纤维;还把以往在电缆网络的每一条支线上都有的放大器全部取消,改为在每个街区至多装4个1000兆赫放大器。由于频道增多,电视网节目更加丰富;由于放大器减少和光导纤维的使用,信号传输可靠性增加了,噪音和失真减少了。

这些改进只是提高电缆功能的开始,通过压缩技术还可以得到更多的频道。例如康涅狄格州美国电视与通信公司副总裁沃特在去年6月伊利诺斯州的一次国际会议上提出了一种方法,按照他的方法,可以将普通的1000兆赫系统的150个频道分为三部分:一部分为60个,传输普通的NTSC节目;另一部分30个,传输高清晰度电视节目;剩下的60个用5:1的压缩法产生300个频道,从而使这个1000兆赫系统共有390个频道。

去年秋天,有线电视实验室(Cable Labs)、远程通信公司、Viacom电视网和弗吉尼亚州亚历山德拉的公

共广播网联合发出一项声明,希望能够得到传送数字压缩节目的网络设备。数字压缩系统之所以有如此吸引力,还因为在今天的太空时代,如果有了优秀的压缩技术,只需一颗卫星就可以将众多节目传向四面八方。

### 数字式声音存储

虽然真正意义上的数字式音频磁带还没有上市,但已经有两种相互竞争的系统出现了。一种是飞利浦的“数字式压缩盒”;另一种是索尼的“微型唱片系统”。

飞利浦的系统可以录制新式的数字式磁带,播放这种磁带或传统的磁带。新式磁带与普通磁带大小相同,当它以最高速度运转时,录制声音信号时间为90分钟。

系统的编码技术称为“精密自适应分波段编码”,是整个产品的核心。它录制声音信号的密度可以比普通脉冲信号调制法大4倍。它的动态范围是108分贝(16位),杂音低于0.0025%,频率范围从5赫兹到22千赫。在录制时,声音频率范围被数字式地分为32个分波段,信号处理器首先根据不同的频率确定声音的最低录制音量起点(也就是只有这个音量以上的音才录,其余的全部舍弃。对人耳来说,对3.4千赫这种中等频率的声音最为敏感,所以处理器将中等频率的起点音量定得最低),然后处理器在录制时动态地适应着不同的音量起点。正由于大量低于起点的无用信号被丢弃,编码速度很快,磁带的容量也很大。这样编码得到的信息,伴随着纠错码,被多路传输到16个185微米宽的磁道上,并列的16个数据磁道加上旁边的辅助磁道都容纳在3.88毫米宽的磁带上。

索尼的“微型唱片系统”可以在64mm的光磁唱片上录制74分钟数字式音频信号,这比标准的音频压缩唱片要多一倍。它使用的压缩技术称为“自适应变换音频编码技术”。

这个系统是在44.1千赫频率用16位量化,对声音进行数字化。编码器首先将一段20毫秒的声音分成1000个小区域,然后把每一个小区(约0.02毫秒)用16位数形式来表示其声音的特征。接着,编码器将人耳最低听力限度、声音互相遮蔽效果等针对人耳能否听到的因素考虑进去,略去没有用的数据,最终达到压缩成只有原始信号的1/5。

此外,索尼这个系统在播放时还有一个特殊设计,有一个可以存入三秒钟声音的1兆字节小存储器,当发生剧烈震动一类短暂的不能读取声音信号的情况时,系统仍可以正常放音三秒钟。

### 收音机广播的压缩技术

数字压缩技术成为一大潮流后,不仅对有线电视、录音系统产生了影响,而且也波及收音机广播,虽然这方面的发展远落后于前两者。91年10月,在美国电子与电子工程师协会的纽约会议上,新泽西州洛克维尔的CCS公司总裁阿尔蒙·克莱格预测了数字式收音机的前景,他说:很多广播公司已经准备向听众提供无

噪音、无干扰的数字式高质量节目,既然采用了数字式,当然压缩技术就大有用武之地。他还认为,这种广播的频率分配可能有三种方式:一种是在一定范围内(比如1.5千兆赫)实行重新分配;一种是在现有AM和FM波段内分配频率,但必须考虑到数字式广播肯定比原有的广播波道要宽;最后一种是在FM的200千赫内,普通广播与数字广播混同使用。当然,这一切都有赖于联邦通信委员会做出新的决定。

## 应用计算机网络辅助 高中物理教学

北京和平街一中(100013)

肖美 王天谥 李燕萍

高中物理尤其是高一物理比较抽象,是历届学生深感难度极大的一门主课。多年的实践证明,采用计算机进行辅助教学,把抽象的、不易观察甚至无法观察的物理现象进行形象逼真的动态模拟,已经在提高学生兴趣、培养学生分析问题和解决问题的能力方面都取得了较好的效果。但是,使用分立的计算机进行辅助教学,并不能满足进一步教学改革的需要。这是因为采用分立的计算机进行辅助教学,教师难以控制各台学生机的进程,师生间的信息难以交流,教师难以掌握每个学生使用计算机学习的实际情况,从而难以进行管理和组织教学。为了进一步为计算机辅助教学创造更好的环境,我们开发了计算机教学网络,将教师机和多个学生机联网,实现资源共享和信息交流,保留并发展了传统教学和分立计算机辅助教学的优点,基本形成了一种新的辅助教学模式,经过多次实践,取得了预期的效果。

### 一、应用计算机网络辅助教学的过程

我们从今年开始,在高一物理课上,用计算机教学网络这个辅助教学手段,参与了“力的合成”、“运动的合成”、“物体的运动”一章复习课,“振动和波”等等物理教学课程,现以高一物理必修本第一册第八章第2节“气体的等温变化、玻—马定律”为例谈谈计算机网络在物理教学中的应用。

课程在计算机房进行,每两个学生一台计算机,共24台学生机,一台教师机,讲桌上有投影仪及玻—马

定律演示器。

教学过程大约分为六步：

(一)教师提出若气体的状态发生变化时,应如何研究状态参量的变化规律。人们研究事物的规律是从简单到复杂,因此我们先研究一个状态参量不变(即温度不变)的条件下,气体的另两个状态参量的变化遵循什么规律。

(二)先用玻—马定律演示器粗略地、定性地向学生演示实验,由于仪器较少,学生离得远,因此学生根本看不见仪器上的读数,只能大致看看水银面上下移动情况,粗略知道压强体积是增大还是减少,但具体改变多少看不见。

(三)由于上述仪器的演示看不清,因此用计算机细致地模拟整个实验的动态过程。由教师在教师机上操作,通过网络发送到每台学生机,教师一边操作一边讲解,学生通过屏幕看到了形象清晰逼真的整个实验过程,读出了各个状态的实验数据,比较了每次实验结果。

(四)教师由实验结果得出玻—马定律,总结出公式,讲解使用条件。由于每个学生都由计算机详细地观察了实验,这样,由实验上升为理论,对玻—马定律的理解就比较容易。

(五)教师向学生发送基本训练题,每个学生自己操作单机,完成各自的作业(题目是随机的,每个学生的题都不一样)。随着学生完成练习,每台学生机的正误率由联网随时反馈到教师机的显示屏上,使教师可以对每个学生的情况进行监视,了解学生练习中的信息,找出薄弱环节。

(六)针对上边练习中的薄弱环节,教师有目的地选择另一些例题发送给学生,并向学生讲解,纠正错误,巩固概念。

## 二、传统教学与计算机网络教学的比较

### 1. 形象生动,激发了学生的学习兴趣:

过去做玻—马定律演示实验时,由于大部分学生看不见仪器的读数,教师只能请一个学生上台来读,绝大多数人得不到参与的机会,学生在课堂上较被动,学习兴趣不浓。现在用计算机动态模拟,使每个学生都可看到详细而生动的实验全过程,亲自观察、读数和计算,边思考边讨论,从而提高了学习兴趣,活跃了课堂气氛,学生反映印象深、记得牢。

### 2. 信息双向传输,反馈迅速:

给学生进行基本训练的题,是从小题库中随机选出的,每人题目不一样,造成一个使学生不能抄答案的环境。练习时学生讨论的是解题思路和方法,并通过计

算机进行人机对话。答对了计算机给予肯定,答错了给予提示。学生不会做的题目,可以要求帮助,计算机给出模拟的物理过程,以及简要的文字说明、公式推导和答案。由于学生立即能知道自己答题的对错,因此增加了练习的兴趣,给学生创造了一个生动活泼的学习环境。

在学生练习时,教师可以通过网络系统对每个学生的情况进行监视,了解学生练习中的动态信息。教师一边监视,一边可将学生的情况进行汇总,及时表扬,提高学生兴趣。若发现了普遍性的错误,可以用网络将错误发给学生,向学生讲解,当堂就能补漏。

整节课以学生动脑为主,教师讲课为辅,充分发挥了学生的主观能动性。

### 3. 加强个别化教学,适应各类学生的需要:

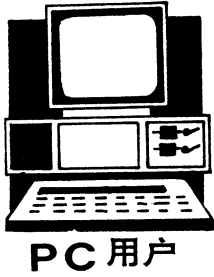
学生在学习中由于学习水平不同,总会存在差异。各班的尖子学生,他们理解力强,做题快。过去传统教学只能从中等生出发,开展教学活动,这样使尖子生不能充分发挥他们的才干,压抑了他们的聪明才智。但用计算机辅助教学,随机题库信息量大,可以使尖子生比别人有更多的练习机会。如在玻—马定律的例题中,我们让学生计算,“在开口向上的玻璃管中,用水银封闭了一部分气体,当开口向下放置时,气柱有多长?”有些同学做得较快,于是计算机又提出更深一层次的问题,“若玻璃管长为40厘米又如何求?”这些学生讨论得非常热烈,直到下课还在争论着。课后我们向学生了解情况,无论是学习好与差的学生反映均为良好。可见,计算机辅助教学不但能适应中等学生的需要,也有助于尖子生的培养。

### 4. 增大课堂容量,提高效率,减轻学生负担:

过去讲玻—马定律这一节,做实验、讲解概念后,最多只能做一道例题,大量的练习留在课后完成。现在应用计算机辅助教学,实验及概念讲解后,还能由易到难用标准化方式练习4道题,还加了两道层次不同的例题;学生的问题也及时通过网络而得以发现,并加以纠正。这样在45分钟内,学生既学到了新知识,又培养了能力,课堂上解决了问题,课下负担自然就减轻了许多。

## 三、结语

实践表明,计算机联网辅助教学不仅保留了传统教学和分立计算机辅助教学的各种优点,而且加强了师生间信息的双向交流,促进教师管理和组织教学,进一步提高学生学习兴趣。我们相信,随着教学方法的进一步完善,计算机联网辅助教学一定会在培养学生分析问题和解决问题的能力上发挥愈来愈大的作用。



# 也谈 FoxBASE+ 弹出式菜单的实现方法

呼和浩特内蒙古科技情报研究所(010020) 王瑞华

PC 用户

《电子与电脑》92 年第六期刊登的《dBASE III 和 FoxBASE+ 环境下实用多窗口弹出式菜单的实现方法》(简称《实用》)一文,为用户提供了方便的弹出菜单的实现方法。由于 FoxBASE+ (Rev. 2.10) 的速度比 dBASE III+ 快 5.90 倍,而且与 dBASE III+100% 兼容,所以目前大多数使用 dBASE III+ 的用户纷纷改用 FoxBASE+。但目前熟悉 FoxBASE+ 的用户不多,所以大多数用户只是把 dBASE III 的程序直接套入 FoxBASE+ 中执行。其实, FoxBASE+ 已经为我们提供了建立弹出式菜单的命令 MENU, MENU TO, @...PROMPT, @(<行>,<列>)MENU。下面将这些命令作一些介绍,并提供一些实例。

## 1 建立下拉式菜单命令 MENU

格式: MENU BAR<数组>,<数字表达式>  
MENU <数字表达式 1>,<数组>,<数字表达式 2>  
[,<数字表达式 3>]  
READ MENU BAR TO <变量 1>,<变量 2>  
[SAVE]

功能: 这个命令建立在顶行的菜单条,并由光标控制键来选择。光标通过的菜单标题将以反象显示,并显示这个标题所包含项目的下拉式菜单。按光标控制键的左右键,则显示相邻标题的下拉式菜单。

实现步骤:

1. 用 DIMENSION 定义顶行菜单棒的数组。
2. 用 DIMENSION 定义与各题目对应的下拉菜单的项目。
3. 用 MENU BAR 使定义的数组进入菜单棒。
4. 用 MENU 将下拉菜单与题目一一对应。
5. 用 READ MENU BAR 激活这个菜单系统。

### 例 1. 资料管理菜单程序

```
set status off
set echo off
set talk off
set message to 24
dimension top(3) && 定义菜单棒数组
top(1)='资料工作'
top(2)='资料借还'
top(3)='资料查询'
dimension zlgz(3) && 定义资料工作菜单数组
zlgz(1)='资料输入'
zlgz(2)='修改内容'
zlgz(3)='删除记录'
dimension jh(2) && 定义资料借还菜单数组
jh(1)='资料借出'
```

```
jh(2)='资料还入'
dimension zlcx(4) && 定义资料查询菜单数组
zlcx(1)='按书名查'
zlcx(2)='按类号查'
zlcx(3)='按作者查'
zlcx(4)='按内容提要查'
menu bar top,3 && 装配下拉菜单系统
menu 1,zlgz,3,3 && 资料工作菜单装入系统
menu 2,jh,2,2 && 资料借还菜单装入系统
menu 3,zlcx,4,4 && 资料查询菜单装入系统
row=1 && 初始化亮条的位置
col=1
read menu bar to row,col && 激活下拉菜单系统
```

## 2 建立上弹菜单命令 @(<行>,<列>)MENU

格式: @(<座标>)MENU <一维数组>,<数字表达式 1>[,  
<数字表达式 2>][TITLE<字符表达式>]

功能: 此命令可在任意位置建立一个上弹菜单。使用可选项 TITLE 可以在顶部显示菜单标题。

实现步骤:

1. 用 DIMENSION 定义一维数组,包括所有的菜单项。
2. @(<行>,<列>)MENU 指定菜单显示位置,数字表达式 1 是菜单所包括的项目数,表达式 2 是显示在屏幕上的菜单项目数,字符表达式定义菜单标题。
3. 用 READ MENU 激活这个菜单。

### 例 2. 建立上弹菜单程序

```
dimension ch(3,1)
ch(1)='图书采购'
ch(2)='图书编目'
ch(3)='图书流通'
store 0 to cc
@5,0 menu ch,3 title '图书管理系统'
read menu to cc
do case
case cc=1
do cg
case cc=2
do bm
case cc=3
do lt
endcase
```

## 3 建立条形光亮的菜单命令 @...PROMPT, MENU TO

格式: @(<行>,<列>)PROMPT<字符表达式>[MESSAGE<字符表达式>]

MENU TO <内存变量>

功能:这两个命令可以在任何希望的位置显示菜单选择项,可以用光标控制键选择后打回车,或直接键入提示的第一个字符。我们将重点介绍《实用》一文没有用到的 MESSAGE 命令。

实现步骤:

1. 用@...PROMPT 初始化菜单中的各选择项, MESSAGE 初始化与各项有关的提示信息。

2. MENU TO 将选择项的序号送入内存变量。

3. SET MESSAGE TO <数字表达式>定义提示信息的显示位置。

### 例 3. 建立条形菜单程序

```

set talk off
@2,1 clea
ch=0
do while ch<>4
@1,10 prompt '资料管理' message space(30)+'输入修改
资料信息'
@1,20 prompt '资料借还' message space(30)+'办理借书
还书手续'
@1,30 prom '资料查询'mess space(30)+'根据书名、作者
查询馆藏资料'
@1,40 prom '结束退出'mess space(30)+'工作结束,返回

```

主菜单'

```

set message to 24
menu to ch
do case
case ch=1
i=1
case ch=2
i=2
case ch=3
i=3
case ch=4
clear
clear all
exit
endcase
i=str(i,1)
do cd&i
enddo
clear
set talk on

```

需注意的两点:1. 如果您采用的是 FoxBASE+ (Rev2.00)版本,只能用第三种方式建立菜单。2. 如果是 Rev2.10 版本,则三者都可以使用。

## C 语言如何读取 dBASE 的库文件

昆明市北京路 333 号大院自动化站(650051) 李晓华

本文介绍用 C 语言读取 dBASE 数据库文件的方法。

### 1. dBASE 数据库文件结构:

数据库文件由头信息、结构表、记录集 3 部份顺序组成。

头信息主要包括 dBASE 标识符(dBASE I 与 dBASE III 使用不同的标识符)、记录长度、记录总数、上次更新数据库文件的日期。

结构表主要是描述数据文件的结构,由各字段组成。每个字段描述块有 32 字节。由于 dBASE I 与 dBASE III 所容纳的最多字段不同(dBASE I 为 32 个字段,dBASE III 为 128 个字段),所以其固定分配字节数也不尽相同,结构表由 0X0D 标志结束。

接下来是记录集,它存储数据库文件的全部记录,所占用的空间等于记录长度乘上记录总数,全部数据类型均以 ASCII 码存放。这部分是我们最关心的,特别要注意的是,由于每个记录把第一字节作为“删除标志”,尔后才是存放各个字段的值,所以记录长度应比

各字段长度之和多一个字节。

现以一个人事档案库 RDK.DBF 为例,该库文件的结构为:

姓名	籍贯	性别	出生年月	文化程度	政治面貌
c10	c6	c4	c8	c6	c6

### 2. 用 C 语言读取 dBASE III 库文件:

首先定义一个对应 RDK.DBF 各字段的数据结构:

```

/* readdbf.c */
#include <stdio.h>
typedef struct {
char start[1]; /* 记录的第一字节 */
char xm[10]; /* 姓名 */
char jg[6]; /* 籍贯 */
char xb[4]; /* 性别 */
char csny[8]; /* 出生年月 */
char whcd[6]; /* 文化程度 */
char zzmm[6]; /* 政治面貌 */

```

```

} rdk;
main(){
FILE *filedbf; /* dBASE III 数据库文件名 */
rdk *buff; /* 存放一条记录的缓冲区 */
char xml[11],jg1[7],xb1[5],csny1[9]
char whcd1[7],zzmm1[7]; /* 存放各字段 */
filedbf = fopen("RDK.DBF","rb"); /* 打开 dBASE III
的库文件 */
if(filedbf == '\0'){ /* 未找到.DBF 文件 */
puts("can't open .DBF file!");
exit(1);}
buff = (rdk *) malloc(sizeof(rdk)); /* 申请内存空间
*/
if(buff == NULL){ /* 无足够的内存空间 */
puts("out of memory");
exit(0);} /* 退出 */
while(fgetc(filedbf) != 0x0d); /* 找记录的开始 */
fgetc(filedbf); /* 跳一个记录 */
puts("姓名 籍贯 性别 出生年月 文化程度 政治面貌");
while(fread(buff,sizeof(rdk),1,filedbf) != NULL){
addeof(buff->xm,xml,10); /* 在各字段尾加一'\0'
*/
addeof(buff->jg,jg1,6);
addeof(buff->xb,xb1,4);
addeof(buff->csny,csny1,8);
addeof(buff->jg,jg1,4);
addeof(buff->whcd,whcd1,6);
addeof(buff->zzmm,zzmm1,6);
printf("%s%s %s%s %s %s\n",xml,jg1,xb1,csny1,
whcd1,zzmm1);
} /* 显示 */
}
int addeof(char *c1,char *c2,int n) /* 在各字段尾加一
'\0'的函数 */
{ while(n-- != 0) *c2++ = *c1++;
*c2 = 0;
}

```

说明:1. 读取文件时,应首先找到库文件结构表的结束标志 0X0D。

2. 由于文件的每条记录开始,都有一个字节作“删除标志”,所以读取时应过滤该字节。

3. 用 C 语言显示字符串时,需要有一个标志 '\0',所以,.DBF 库的每个字段末应加上 '\0' 标志后,才能正确显示。

4. 如库的字段为数据类型,且要进行数字运算时,需要用 atoi(s)函数把字符串转换成数字。

## 如何实现题库系统的图文并貌

天津师范大学计算机系 裴伟东

如何实现题库系统的图文并貌,是一直没有得到较好解决的一个问题。过去大多是在建立题库的同时,造出大量的图形符号,并利用它们拼成某种图形存放在题库中。这种方法的主要缺点是工作量太大,需要较多的人力和较长的时间才能完成。下面,根据我们在开发“离散数学题库系统”过程中的一些体会,向读者介绍一种简单实用的方法,供大家参考,并抛砖引玉。

### 一、绘图工具的设计

为实现题库系统的图文并貌,我们用 FoxBASE+2.00 和 2.13H 系统的特殊显示功能,首先设计了一个绘图程序 ht.prg,其功能、绘图方式及绘图原理如下:

#### 1. 功能

ht.prg 的主要功能是:

- (1) 绘制带有节点的树型图(有向图或无向图)。
- (2) 绘制表格、矩阵型图(有向图或无向图)。
- (3) 把绘好的图形特征信息存储在图形特征库中。

#### 2. 绘图方式

ht.prg 用 FoxBASE 的绘图语句将屏幕划分成三个主要的窗口,它们是:

- (1) 绘图窗口。该窗口规定了绘制图形的大小范

围,所使用的绘图语句只能把图形绘制在该窗口内。在整个图形的绘制过程中,从头到尾每一步的绘制图象都显示在该窗口内,以待用户给出是否满意的信息。

(2) 提示窗口。该窗口给出每一步要绘制的图象内容,包括节点、连线、数据(字母、汉字等也属数据内容)、它们的位置、连线的方向和特征等信息,待用户选择后,该步所绘制的图象即显现在绘图窗口中。

(3) 操作窗口。该窗口主要是选择提示窗口上的信息,并在每一步所绘制的图象显现在绘图窗口之后,询问用户是否满意、是否要修改、整个图形是否已画完等操作信息。

用户通过以上三个窗口的反复操作,最后将一个理想的图形绘制在绘图窗口内。

#### 3. 图形的存储方式

绘图程序 ht.prg 的另一个功能还在于它能存储一个图形的全部特征信息,为此,我们建立了一个存储图形特征信息的特征库 TU.DBF。考虑到各种图形内部有的有数据,有的无数据,数据的多少又和图形的复杂程度没有必然的联系等,为了节省特征库 TU.DBF 的存储空间,我们另行建立了一个数据库 DATA.DBF

用以专门存放图形中的数据;DATA. BF 通过 TU. DBF 中的一个字段与 TU. DBF 发生一一对应的关系,这两个库的结构如图 1.1 和图 1.2

Field	Field Name	Type	Width	Dec
1	X1	Numeric	2	
2	Y1	Numeric	2	
3	UJ1	Character	2	

图 1.1 DATA. DBF 库结构

图 1.1 中, X1, Y1 记录图形当前数据的位置,是文本方式下 X, Y 的坐标; UJ1 记录当前数据的内容。

Field	Field Name	Type	Width	Dec
1	X	Numeric	2	
2	Y	Numeric	2	
3	L1	Numeric	2	
4	L2	Numeric	2	
5	FX1	Character	20	
6	FX2	Character	20	
7	FX3	Character	20	
8	FX4	Character	20	
9	FX5	Character	20	
10	FX6	Character	20	
11	FX7	Character	20	
12	FX8	Character	20	
13	U1	Character	3	
14	V1	Character	3	
15	U2	Character	3	
16	V2	Character	3	
17	U3	Character	3	
18	V3	Character	3	
19	U4	Character	3	
20	V4	Character	3	
21	UJ	Numeric	1	

图 1.2 TU. DBF 库结构

图 1.2 中, X, Y 记录节点或连线的位置;是文本方式下 X, Y 的坐标; L1, L2 记录与当前节点或连线位置有关的点; FX1-FX8 记录当前连线的方向特征,它们分别为上、下、左、左上、左下、右、右上、右下等; U1, V1~U4, V4 为各种方向的位置点; UJ 是图形中有无数据的标记,当 UJ=1 时,表示该图形中有数据,数据内容存储在相对应的 DATA. DBF 中,当 UJ=0 时,表示该图形中没有数据。

以上两库中各字段的意义,包含了一个图形应具备的各种特征信息。

#### 4. ht. prg 的程序原理

简单地讲, ht. prg 就是从提示窗口获得图形特征信息,再把该信息所代表的图象用绘图语句绘制在绘图窗口,如果用户满意,则将该特征信息存储在 TU. DBF 和 DATA. DBF 中,整个图形绘制完毕时,描绘该图形的全部特征信息也就存储在固定的特征库里了。

鉴于树型图与表格、矩阵型图在绘制原理上的一致性,这里只给出 ht. prg 关于树型图的绘制原理。

(1)第一步是把一个树型图的全部节点画好;第二步是连结节点之间的连线并画好连线的方向。具体过程是从第一个节点连起,直到最后一个节点完毕,且这个过程可重复一遍;第三步是填好图形中的数据。

(2)程序中并没有存储节点的形状,这是因为树型图的节点形状不变,我们统一规定了它们的形状都为“。”。例如,要在坐标点(x,y)处画一个节点,可用语句 @x,y say “。”实现。可见,节点形状并不需要存储。

(3)连线和连线方向的画法是通过用 FoxBASE 语句和 2.13H 系统的特殊显示功能来实现的。例如,要把坐标(x<sub>1</sub>,y<sub>1</sub>)处的节点与坐标(x<sub>2</sub>,y<sub>2</sub>)处的节点连起来,可用下面两条语句来实现:

```
@1,1 say chr(14)+“Dx1,y1”
```

```
@1,1 say chr(14)+“Lx2,y2”
```

其中:(x<sub>1</sub>,y<sub>1</sub>)和(x<sub>2</sub>,y<sub>2</sub>)必须是图形方式下的坐标,它们与文本方式下的坐标可相互转换。例如,在 EGA-26 行显示模式下,设文本坐标为(x,y),则其图形坐标(x<sub>1</sub>,y<sub>1</sub>)的值可由下式计算:

$$x_1 = \frac{350}{26}x \approx 13.46x, y_1 = \frac{640}{79}y \approx 8.10y$$

## 二、图形显示工具的设计

由于 ht. prg 将绘好的图形以图形特征信息存储在特征库文件中,当调用图形时,就必须把其特征信息还原成真正的图形,为此,我们利用类似前面的原理,设计了一个图形显示程序 xt. prg,其功能和原理如下:

1. 功能 显示由 ht. prg 绘制过的图形。

2. 原理 xt. prg 原理简介如下。

ht. prg 本身就包含了一个绘图过程,只不过它是从提示窗口获得一个图象信息,再把该信息代表的图象绘制出来,而 xt. prg 的不同之处在于,它不是从提示窗口获得图象信息,而是从图形特征库中获得信息,再把该信息所代表的图象绘制出来。两个程序中所使用的绘图语句都是一样的。

## 三、题库中如何存取图形

由 ht. prg 的绘图原理我们知道, ht. prg 把绘制过的图形特征存放在固定的特征库 TU. DBF 和 DATA. DBF 中,它们目前还没有和题库中的题目产生一一对应的关系,那么,题库中的某个题目与它的图形怎样一一对应呢?图形的存取方式又是怎样的呢?下面,我们就这两个问题分别加以讨论。

(1)题库中题、图对应关系的实现

我们以一个具体的例子,来说明在录入题目时,某个题目与它的图形怎样产生一一对应的关系。

设“离散数学”中关于“树”部分的题目存放在 LS5. DBF 中,为方便说明起见,我们把该库的结构简化为 TM, BZ, TU 三个字段。其中:

TM——题目中除图以外其它必备字段的总和;

BZ——标志字段;当 BZ=1 时,表示当前题目有图;

TU——当前图形的文件名字段,其中文件名代表存储该图形特征信息的特征库文件名。

当录入这部分题目到 LS5. DBF 中时,某个题目与其图形的对应关系及存图方式流程如下:



1. 输入图形以外的题目内容到字段 TM。
2. 若题目无图,置 BZ=0,退出录入过程。
3. 题目有图,置 BZ=1。调用 ht. prg 绘图及存储图形特征信息。
4. 求出题目在 LS5. DBF 中的记录号(设为 25)。
5. 若 TU. DBF 中字段 UJ 值=0,即图中无数据,退出录入过程。
6. 若 UJ=1,将 DATA. DBF 拷贝到 LS525. DBF
7. 将 TU. DBF 拷贝到 LS525. DBF,将字符串“LS525. DBF”存入 LS5. DBF 当前记录的字段 TU 中。
8. 结束和退出录入过程

由录入流程可知,当题号为 25 的题目有图形时,置该库的当前字段 BZ=1,置字段 TU=“LS525. DBF”,其中,“LS5”表示“树”部分的题库,“25”表示该库中第 25 号题目,“LS525. DBF”表示该题目图形的图形文件名。由于题库名和记录号都是唯一的,于是,

“LS525. DBF”就与该题目产生了一一对应的关系。

#### (2)题库中图形的存取方式

题目录入流程中实际上已给出了图形的存放方式,即把某题目的图形特征信息存储在与该题目相关的一个特征库文件中,上例是“LS525. DBF”,也即每个图形对应于一个相关的特征库文件。取图时,比如在修改、删除或显示题目时,若 BZ=1,则显图程序 xt. prg 就自动调用与该题相对应的那个图形特征库(对上例的 25 号题而言,xt. prg 就自动调用 LS525. DBF)并显示其图形,从而实现了题库的图文并茂。

此外,由于每个图形存放的是数量不多的特征信息,可以把某类题目的所有图形信息全部存放在一张软盘上,既节省存储又方便使用。在打印图形方面,对没有图形的题目,直接按规定格式打印;遇到有图形的题目时,先将题目和图形显示在屏幕上,然后调用‘2. 13H 的特殊拷贝功能将其拷贝出来,最后恢复原来的格式继续打印,实现了试卷的图文并茂。

## 一个简单实用的磁盘加解密程序

甘肃庆阳长庆一中(745100) 任经海

如果别人发现你的磁盘上什么东西也没有,或者看到列目录时显示出的是些怪字符,那么他当然也不会不能运行你藏在该盘上的文件。

下面的程序先将磁盘目录区的第一扇区读出,放在偏移 200H 处,再将双面双密盘倒数第二扇区读入偏移 800H 处,然后将两区内容交换存盘。这样一来,别人就看不出你的盘上究竟有些什么文件。你如果想恢复原状,也只需再运行一次本程序即可。它是单次加密,偶次解密,你不必特别记住什么密码之类的东西。

AL 中是驱动器号、CX 为传送扇区数,如果你在别的驱动器上操作,或者文件比较多,不只一个扇区,那么就要改变以上参数。

有必要一提的是,INT25/INT26 绝对磁盘读/写中断,是用 RET 命令返回用户程序的,与别的中断不同(用 IRET 返回),调用结束后,F 标志仍留栈中,若不及时出栈,势必使此后的堆栈操作产生混乱。故而要求在用户程序中将栈顶指针 SP 复位。程序中的两个语句:ADD SP,2 语句和 POPF 就是完成这项工作的(作用相同)。

C>DEBUG

-A100

```
100 MOV AL,1
102 MOV BX,200
105 MOV CX,1
108 MOV DX,5
10B INT 25
```

```
10D POPF
10E MOV AL,1
110 MOV BX,800
113 MOV CX,1
116 MOV DX,2CE
119 INT 25
11B POPF
11C MOV AL,1
11E MOV BX,200
121 MOV CX,1
124 MOV DX,2CE
127 INT 26
129 POPF
12A MOV AL,1
12C MOV BX,800
12F MOV CX,1
132 MOV DX,5
135 INT 26
137 ADD SP,2
13A INT 20
-RBX ;将 BX 清零
:0
-RCX
:3C ;程序长度为 3CH
-NJJM.COM ;名为 JJM.COM
-W ;存盘
-Q ;退出
```

## 2.13HZ 汉字系统安装虚拟盘字库新探

北京铁路局防疫站(100038) 廖 凯

2.13H 是国内广大用户较受欢迎的一种汉字系统,而 MS-DOS 5.0 是目前功能最强的 DOS 操作系统,它可以将 DOS 的一部分装入到扩充存储器(EXTENDED MEMORY)内,而空出近 50K 的常规内存空间供用户使用。

本人在 MS-DOS 5.0 下安装 2.13 汉字系统时却发生了问题,就是在将 DOS 放入 HMA(1M 以上的第一个 64K 空间)后,将 HZK16 显示字库拷入用 RAMDRIVE.SYS 建立的虚拟盘后再读取汉字时,显示的汉字并不是所要的汉字。

经本人研究 FILE3.COM 程序发现,此程序在计算指定汉字的点阵数据地址时将基址设为 1M(见程序),这是因为用 V-DISK.SYS 建立的虚拟盘(MS-DOS 5.0 以下版)是在 1M 以上建立的,而在 MS-DOS 5.0 将 DOS 的绝大部分放入 HMA 后,再建立的虚拟盘是在 1M+64K 以上建立的。因此,FILE3.COM 在计算指定汉字的点阵数据地址时相差 64K。

若想计算正确,就要设基址为 1M+64K。也就是将程序的 ADC DX, +10 改为 ADC DX, +11, 这样,问题就彻底解决了。

```
U0261:
MOV AX, [SI+1A]
SUB AX, 0002
MOV DL, [030D]
XOR DH, DH
MUL DX
MOV DX, 0200
MUL DX
ADD [0124], AX
ADC DX, +10; 加基址 1M, 若低 16 位有进位, 则高 8 位
 加 1
; DL 和 AX 组成 24 位地址, 因此, DL=10H 即表示 1M
MOV [0122], DL; 写入高 8 位数据
MOV DX, 0103
MOV AX, 257F
INT 21
```

具体修改方法如下:

- ①DEBUG FILE3.COM
- ②E027A 11
- ③W 和 Q 退出

在配置文件 CONFIG.SYS 中应有如下设置:

```
DEVICE=\DOS\ANSI.SYS
DEVICE=\DOS\HIMEM.SYS
DEVICE=\DOS\RAMDRIVE.SYS 260 512 16/E
```

```
DOS=HIGH
FILES=25
BUFFERS=20,8
INSTALL=\DOS\DOSKEY
```

修改后重新启动系统,进入汉字系统后,所显示的汉字都很正常。

## 怎样从死循环程序返回 DOS

乌鲁木齐石油化工总厂化肥厂(830019) 方晨

有许多软件是不能返回 DOS 的(其中大量游戏程序),它占用计算机资源,直至重新启动。这些程序虽为使用者提供方便的专用功能,但对调试分析是不利的,因为这势必导致频繁的系统重启甚至关机。

要改变这种现象是可能的,只要使系统通过中断执行一段返回 DOS 的指令即可。这个中断可随时由某个键激活。如果 INT 5H 不被占用,也可将此指令置入 INT 5H,这样只要键入 Shift-Prtsc 即可返回。

用 Shift-Prtsc 返回 DOS 的程序如下:

```
.MODEL TINY ;置 Tiny 型
.CODE
ORG 100H ;COM 文件头定位
START: JMP MAINPRO ;转入装载程序
INT5H: ;新 INT 5H 首地址
MOV AH, 4CH; DOS 4CH 功能
INT 21H ;返回 DOS
RET
MAINPRO: MOV DX, OFFSET INT5H
MOV AX, 2505H ;置中断向量
INT 21H ;调用 DOS
MOV DX, OFFSET MAINPRO
INT 27H ;结束并驻留
END START
```

本程序名为 RETDOS.ASM,通过 TURBO 汇编:

- C> TASM RETDOS.ASM
- C> TLINK /T RETDOS.ASM

在死循环程序运行前装入,运行死循环程序后,只要按下 Shift-PrtSc 即可返回 DOS。

## BASIC 实现“卡拉—OK”

54685 部队技术部 于龙滋

卡拉—OK 录相带越来越受人们的喜欢,能否

用计算机模拟“卡拉—OK”呢?我用 BASIC 编制了一个程序,完成了音乐演奏和歌词显示效果。程序执行后,屏幕显示电子琴的演奏过程,同时随音乐依次显示相应的歌词。可同时输入9首歌曲供选择。该程序已在286微机中通过。本程序只输入了《在水一方》一首歌。

#### 程序说明

第2至270行,电子琴键盘显示。

第490至620行为演奏子程序,可完成音乐演奏、电子琴演奏过程显示及歌词显示功能。

第370、380行,计算发音频率。

第630至769行,选择歌曲。键入A—I可选择9首不同的歌曲,按ESC键退出。

第1000行以后,歌曲置数语句。歌曲A从1600行开始置数;歌曲B从1700行开始置数;歌曲C从1800行开始置数;歌曲D从1000行开始置数;歌曲E从1100行开始置数;歌曲F从1200行开始置数;歌曲G从1300行开始置数;歌曲H从1400行开始置数;歌曲I从1500行开始置数。

程序中第14、220、260、610、620的空格字串为两个字符;170~190号字串中的空格间隔为16字符。

```

2 COLOR 15,0,0
5 CLS;COLOR 0,12
10 FOR I=5 TO 74;FOR J=0 TO 1
11 LOCATE 1+J*8,I;PRINT " "
12 NEXT;NEXT
13 FOR I=0 TO 1;FOR J=2 TO 8
14 LOCATE J,5+I*68;PRINT " "
15 NEXT;NEXT
150 COLOR 15,0
170 LOCATE 11,15;PRINT"A—B—C—"
180 LOCATE 12,15;PRINT"D—E—F—在水一方"
190 LOCATE 13,15;PRINT"G—H—I—"
200 COLOR 0,15
205 FOR J=2 TO 8
210 FOR I=0 TO 15
220 LOCATE J,7+I*4;PRINT "|";" "
230 NEXT I
231 PRINT "|"
232 NEXT J
235 COLOR 15,0
240 FOR I=0 TO 12;FOR J=2 TO 5
250 IF I=2 OR I=6 OR I=9 OR I=13 THEN 270
260 LOCATE J,15+I*4;PRINT " "
270 NEXT;NEXT
275 COLOR 15,1,1
360 DIM M(88),O(70)
370 FOR I=7 TO 88;M(I)=36.8*(2^(1/12))^(I-6);
NEXT
380 FOR I=0 TO 6;M(I)=32767;NEXT
390 O(0)=0
400 O(39)=5;O(40)=7;O(41)=8;O(42)=9
410 O(43)=10;O(44)=11;O(45)=13;O(46)=14
420 O(47)=15;O(48)=16;O(49)=17;O(50)=18
430 O(51)=19;O(52)=21;O(53)=22;O(54)=23

```

```

440 O(55)=24;O(56)=25;O(57)=27;O(58)=28
450 O(59)=29;O(60)=30;O(61)=31;O(62)=32
460 O(63)=33;O(64)=35;O(65)=36;O(66)=37
470 O(67)=38;O(68)=39;O(69)=40;O(70)=42
480 GOTO 630
490 READ J,K,L
500 CMD$=INKEY$;IF CMD$="" THEN 540
510 IF CMD$=CHR$(27) THEN RETURN
540 IF J=-1 THEN RETURN
541 IF J<>-3 THEN 550
542 COLOR 14,4;LOCATE 16,6
543 PRINT " "
544 LOCATE 17,6
545 PRINT " "
546 LL=0;READ SS$;LOCATE 16,14;PRINT SS$
547 GOTO 490
550 Q=O(J)
555 LL=LL+L;COLOR 14,4;LOCATE 17,12+2*LL;
PRINT">"
560 IF J>64 OR J<39 THEN 590
570 IF INT(Q/2)-Q/2<>0 THEN COLOR 14,4;LO-
CATE 7,2*Q-1;PRINT"◎";GOTO 590
580 COLOR 14,4;LOCATE 3,2*Q-1;PRINT"◎";
590 SOUND M(J),K;IF J=0 AND K=1 THEN 600;
"SKIP NEXT FOR STACCATTO
595 SOUND 32767,1
600 IF J>64 OR J<39 THEN 490
610 IF INT(Q/2)-Q/2=0 THEN COLOR 15,0;LO-
CATE 3,2*Q-1;PRINT" ";GOTO 490
620 COLOR 0,15;LOCATE 7,2*Q-1;PRINT" ";
GOTO 490
630 LOCATE 15,15;PRINT "ENTER SELECTION =
>";
631 COLOR 14,4;LOCATE 16,6
632 PRITN " "
633 LOCATE 17,6
634 PRINT " "
650 IF INKEY$<>" " THEN 650
660 CMD$=INKEY$;IF CMD$="" THEN 660
670 IF CMD$=CHR$(27); THEN 850
685 IF CMD$="A" OR CMD$="a" THEN RESTORE
1600;GOTO 770
690 IF CMD$="B" OR CMD$="b" THEN RESTORE
1700;GOTO 770
700 IF CMD$="C" OR CMD$="c" THEN RESTORE
1800;GOTO 770
710 IF CMD$="D" OR CMD$="d" THEN RESTORE
1000;GOTO 770
720 IF CMD$="E" OR CMD$="e" THEN RESTORE
1100;GOTO 770
730 IF CMD$="F" OR CMD$="f" THEN RESTORE
1200;GOTO 770
740 IF CMD$="G" OR CMD$="g" THEN RESTORE
1300;GOTO 770
750 IF CMD$="H" OR CMD$="h" THEN RESTORE
1400;GOTO 770

```

```

755 IF CMD$ = "I" OR CMD$ = "i" THEN RESTORE
 1500;GOTO 770
769 GOTO 630
770 READ D
800 READ S$:LOCATE 15,35
805 COLOR 14,4;PRINT S$:COLOR 0,7
810 GOSUB 490
820 LOCATE 15,35;COLOR 15,0 PRINT"
"
830 GOTO 630
840 END
1200 DATA -2,"在水一方"
1205 DATA -3,0,0"绿草苍苍,白雾茫茫,有位佳人在水
 一方。"
1206 DATA 47,6,1,49,6,0,52,36,1,59,6,1,61,6,0,56,
 36,1,49,6,2,52,6,0,56,6,0,54,36,1,56,6,1,61,6,
 0
1207 DATA 56,6,0,59,36,1,59,12,2,61,36,1,63,6,1,
 61,6,0,59,6,0,56,36,1,47,6,1,59,3,0
1208 DATA 56,3,0,54,36,1,49,6,1,51,6,0,52,36,1
1215 DATA -3,0,0"绿草萋萋,白雾迷离,有位佳人靠水
 而居。"
1216 DATA 47,6,1,49,6,0,52,36,1,59,6,1,61,6,0,56,
 36,1,49,6,2,52,6,0,56,6,0,54,36,1,56,6,1,61,6,
 0
1217 DATA 56,6,0,59,36,1,59,8,2,61,36,1,63,6,1,61,
 6,0,59,6,0,56,36,1,47,6,1,59,3,0
1218 DATA 56,3,0,54,36,1,49,6,1,51,6,0,52,48,1
1220 DATA -3,0,0"我愿逆流而上,依偎在她身旁,无奈
 前有险滩,道路又远又长"
1222 DATA 64,18,1,63,6,1,61,6,1,63,6,0,64,12,1,
 56,6,1,57,6,0,59,36,1
1224 DATA 61,12,2,59,6,1,57,6,0,56,12,1,49,6,1,
 56,6,1,54,36,1
1226 DATA 56,6,2,57,6,0,59,18,1,61,6,1,59,12,1,
 57,6,1,56,6,0,54,36,1
1228 DATA 54,6,2,56,6,0,57,18,1,64,6,1,63,12,1,
 54,6,1,61,6,0,59,48,1
1230 DATA -3,0,0"我愿顺流而下,找寻她的方向,却见
 依稀仿佛,她在水的中央"
1232 DATA 64,18,1,63,6,1,61,6,1,63,6,0,64,12,1,
 56,6,1,57,6,0,59,36,1
1234 DATA 61,12,2,59,6,1,57,6,0,56,12,1,49,6,1,
 56,6,1,54,36,1
1236 DATA 56,6,2,57,6,0,59,18,1,61,6,1,59,12,1,
 57,6,1,56,6,0,54,36,1
1238 DATA 54,6,2,56,6,0,57,12,1,63,6,1,61,6,0,59,
 12,1,51,6,1,54,6,0,52,48,1
1240 DATA -3,0,0"我愿逆流而上,与她轻言细语,无奈
 前有险滩,道路曲折无已"
1242 DATA 64,18,1,63,6,1,61,6,1,63,6,0,64,12,1,
 56,6,1,57,6,0,59,36,1
1244 DATA 61,12,2,59,6,1,6,57,6,0,56,12,1,49,6,1,
 56,6,1,54,36,1
1246 DATA 56,6,2,57,6,0,59,18,1,61,6,1,59,12,1,
 57,6,1,56,6,0,54,36,1
1248 DATA 54,6,2,56,6,0,57,18,1,64,6,1,63,12,1,
 54,6,1,61,6,0,59,48,1
1250 DATA -3,0,0,"我愿顺流而下,找寻她的踪迹,却见
 依稀仿佛,她在水中伫立"
1252 DATA 64,18,1,63,6,1,61,6,1,63,6,0,64,12,1,
 56,6,1,57,6,0,59,36,1
1254 DATA 61,12,2,59,6,1,57,6,0,56,12,1,49,6,1,
 56,6,1,54,36,1
1256 DATA 56,6,2,57,6,0,59,18,1,61,6,1,59,12,1,
 57,6,1,56,6,0,54,36,1
1258 DATA 54,6,2,56,6,0,57,12,1,63,6,1,61,6,0,59,
 12,1,51,6,1,54,6,0,52,48,1
1260 DATA -1,-1,-1
1600 DATA -2," "
1611 DATA 52,20,1,43,20,1,44,20,1,45,20,1,46,20,
 1,47,20,1,48,20,1,49,20,1,50,20,1,51,20,1,52,
 20,1,53,20,1,54,20,1,55,20,1,56,20,1,57,20,1,
 58,20,1,59,20,1
1612 DATA 60,20,1,61,20,1,62,20,1,63,20,1,64,20,
 1,65,20,1
1613 DATA -1,-1,-1

```

## 实现1.2MB软磁盘之间直接拷贝简法

河南省人民银行计算中心(450002) 何崇乐

近读《电子与电脑》92年第4期上的《在单台驱动器上如何作软盘备份》(下称“杨文”),使我想起《软件报》92年第6期上的《一种快速简便的整盘文件拷贝法》(下称“软文”)及去年《中国金融电脑》(#9/91)上的《怎样实现1.2MB软磁盘之间直接拷贝》(下称“融文”);三文虽说用法不同,但似均为舍近求远,因此,愿以拙见就教于同行。

首先,不论在DOS 2.X(1)还是DOS 3.X下,即使对于双软驱机,也均可按单软驱机的盘拷法进行整盘拷贝。(此点DOS手册上本已有阐述;或如“杨文”所说:“其实,DOS的设计者为我们考虑了这个问题”……)

只是,若A驱为高密驱动器,因DOS 2.X管辖权所限,使DISKCOPY命令无法正确执行;换言之,符合我们要求的,只能是使用DOS 3.X操作系统。

其次,在高密A驱上由1.2MB盘直接(真正!)拷贝1.2MB盘的方法如下:

在DISKCOPY.COM所在目录下(否则,得指定

PATH),直接键入

```
DISKCOPY A: A:<CR>
```

即可;且其间的换盘信息和换盘操作,与在单软驱时的完全相同。

须指出的是,与 DOS 2.X 下的 DISKCOPY 操作不同,在 DOS 3.X 下,驱动器的指定既不能全部缺省,也不能部分缺省,而是缺一不可——必如上述方可。

以此与上述三文所用方法相较,可以看出:

“软文”所述方法,除了插盘次数要少些外,则无论从省时,还是从节省相对贵重的资源——硬盘——来看,均不理想。

而“融文”所说的方法,实际上既不“直接”,又属多此一举,更不理想。

至于“杨文”的方法:除非你必须用 COPY 命令来实现(个别文件的)复制,此时只好如“杨文”所述费事(建/补两个.SYS文件)外;只要想以 DISKCOPY 来实现复制(哪怕此后再删去无关文件),则显然以本法为简!何况,“杨文”的原意,也似在(全盘)“备份”。

正因如此,聊成此文,以对不知、不详“直接拷贝”法者有所裨益。

## 兼容机怎样运行 BASICA.COM

广西苍梧佛子冲矿计算室(543100) 邓文超

因 BASICA.COM 需 ROMBASIC 支持,而一般兼容机都没插有容量为 32K 的 ROMBASIC 芯片,所以多数兼容机不能运行 PC-DOS 的 BASICA.COM,而只能运行 MS-DOS 的 GWBASIC.EXE。极大多数 BASIC 语言教科书均以 PC-DOS 的 BASICA 为范例(最明显的是功能键的定义),况且 BASICA.COM 具有调用 DOS 命令(SHELL“命令”)等实用功能,因此多数用户更乐意使用 BASICA.COM。

下面介绍一个让兼容机也能运行 BASICA.COM 的方法,此方法使用了 DOS 的重定向功能,其优点在于每次运行 BASICA.COM 时只要打入 BAS 即可,无需多余的操作,就象在 IBM 机上使用 BASICA 一样简便。以下是一次性的准备工作,共六点:

1. 准备一张空软盘,将 DEBUG.COM 文件和 PC-

TOOLS.EXE 文件拷贝到该软盘上。并参照 DEBUG 的命令格式,在盘上建立一个借助 DEBUG 读取 ROMBASIC 参数的命令集文件,文件取名为 READROM.TXT:

```
A>TYPE READROM.TXT
M F600:0 L 8000 100
R CX
8000
N A:ROMBAS.DAT
W
Q
```

2. 将此软盘插入 IBM 原装机中,在 A>下打入 DEBUG< READROM.TXT,几秒钟后即可在软盘上生成一个名为 ROMBAS.DAT 的含有 ROMBASIC 参数的数据文件。

3. 将 BASICA.COM 拷贝到软盘上。

4. 启动 PCTools,选择软盘的 BASICA.COM 文件,按 F 键进入查找功能,按 F1 键变换查找方式为找 16 进制码,输入 00F6 并回车,开始查找。找到后按 E 键对 00F6 进行编辑,改为 0070,然后按 F5 键认可,按回车键返回,再按 G 键继续查找,找到再修改,直到全部 00F6 改为 0070 为止。退出 PCTools。

5. 参照 DEBUG 的命令格式建立以下装入 ROMBASIC 参数的命令集文件 BAS.TXT 于软盘上:

```
A>TYPE BAS.TXT
N A:ROMBAS.DAT
L
M 100 L8000 7000:0
Q
```

6. 再在软盘上建立以下启动 BASICA.COM 的批命令处理文件 BAS.BAT:

```
A>TYPE BAS.BAT
A:DEBUG <A:BAS.TXT
A:BASICA
```

以上 DEBUG 后跟的“<”符号为 DOS 的重定向符号。

至此,准备工作已完成,软盘上已含有:PC-TOOLS.EXE、READROM.TXT、DEBUG.COM、BAS.TXT、ROMBAS.DAT、BASICA.COM(已修改段地址 F600 为 7000)、BAS.BAT 共七个文件,其中后五个文件是在兼容机上运行 BASICA.COM 所必需的。

当要运行 BASICA.COM 时,插软盘入兼容机,打入 BAS 并回车即可进入 BASICA。

若要使用带执行文件名的方式进入 BASICA 的话,不妨按如下步骤进行:

```
A>DEBUG<BAS.TXT
A>BASICA 文件名(CR)
```

即先装入 ROMBASIC(每次开机装入一次即可),然后再象平常那样启动 BASICA.COM,非常方便。

# 香港病毒的消除和防范

南京东南大学四系四四教研组(210018) 孟桥

在一台计算机使用过程中,我们发现拒不执行打印命令的情况。经过仔细检查,排除了硬件故障的可能性。再用 CHKDSK 命令检查,发现可用内存比平时少了1K。于是怀疑有病毒作祟。用了几种病毒扫描软件均未发现异常。后用 CPAV 软件,查出其中有“香港病毒”(Hong Kong virus)。但是该软件因为缺乏对该病毒的分析资料,所以不能解除硬盘上的病毒体。

为了消除硬盘上的“香港病毒”,我们仔细分析了该病毒的机制和结构,并找出了一种较为简单的消除并防范该病毒的方法。

## 一、病毒的作用机制和危害

“香港病毒”载体存在于硬盘或软盘的绝对扇区0中,对于软盘,这里应该是引导程序所在之处。病毒在感染时将原来该扇区的内容存入第39道8区1头中。如该盘是360K软盘,则这是最后一个逻辑扇区,一般除非软盘已用满,否则不会用到它。但是对于1.2M、1.44M和720K的软盘而言,它正处于中间位置,极易破坏软盘上的数据。对于硬盘,此处放的是主引导程序,此时病毒体用其自身替代主引导程序,覆盖了原来的内容。此是其危害之一。

病毒体本身也具有一定的分区引导功能,但比较简单,缺乏对引导分区的检验、分区表的检查、系统错误提示、ROMBASIC引导等等功能。所以一旦系统不正常就会引起死机,而不会给用户以任何提示信息。此其危害之二。

当计算机开机启动时,病毒载体替代原来的主引导程序(硬盘启动时)或引导程序(软盘启动时)而首先进入系统内存并执行。它将病毒传播程序链接于DOS系统的中断INT 13H(磁盘I/O)中,将自身拷贝到内存高端,并通过改变BIOS参数区使可用内存减少1K字节,对病毒体进行保护。此其危害之三。

病毒体链接于INT 13H上后,每当系统进行磁盘I/O时,首先执行病毒传播程序,对被读写的磁盘进行感染,既传播了病毒,也降低了程序运行速度。对硬盘的感染则是在用有毒软盘启动时进行。此其危害之四。

当病毒载体把病毒传播程序链接于中断INT 13H之后,便使其内部计数器加一,同时写回磁盘。该计数器记录了病毒感染后机器被启动的次数。当计数器的值大于512时,病毒载体通过修改BIOS参数对打印口PRN1、串行通信口COM1进行“软锁”,使打印机、串行口操作不能正常进行。例如:当使用打印机时,屏幕上出现“No paper...”等字样。病毒的发作具有一定的潜伏期,故发作时往往已进行了一定范围的传染,

而发作时的症状极易被判断为硬件故障,给计算机的维修带来一定的困难。此其危害之五。

综上所述,“香港病毒”侵害计算机的最终目的在于“软锁”部分硬件外设,给使用者带来一些麻烦。但它不进攻计算机的数据系统,应该算是良性病毒。

## 二、“香港病毒”的消除和防范

目前一般常用的防毒软件都无法消除“香港病毒”。由于它存在于系统的引导区中,可以先将所有硬盘中文件进行备份,再重新格式化(包括低级格式化)硬盘,将文件重新拷入系统,最后消除(格式化)数据备份软盘(因为此时软盘上一定感染了病毒)。但是这样做工作量太大。这里介绍一个简单的方法,可以有效地用以消除和防范“香港病毒”。

对于软盘而言,消除该病毒的方法非常简单。首先判断是否感染了病毒。该病毒的特征是在偏移089h和08Ah处的码字为B1和90。判别软盘有病毒以后,只要将被隐藏的正常BOOT区写回即可。下面的一段程序KILLHKA.COM就是用于消除A盘的病毒的。它首先判断A盘是否有香港病毒。若有,则进行消毒:

```
C>DEBUG
-A100
XXXX:0100 MOV AX,0
XXXX:0103 INT 13
XXXX:0105 MOV AX,0201
XXXX:0108 MOV BX,0400
XXXX:010B MOV CX,1
XXXX:010E MOV DX,0
XXXX:0111 INT 13
XXXX:0113 JNB 117
XXXX:0115 INT 20
XXXX:0117 MOV AX,[0489]
XXXX:011A CMP AX,90B1
XXXX:011D JNZ 0115
XXXX:011F MOV AX,0201
XXXX:0122 MOV BX,0400
XXXX:0125 MOV CX,2708
XXXX:0128 MOV DX,0100
XXXX:012B INT 13
XXXX:012D MOV AX,0301
XXXX:0130 MOV BX,0200
XXXX:0133 MOV CX,1
XXXX:0136 MOV DX,0
XXXX:0139 INT 13
XXXX:013B INT 20
XXXX:013D
-RCX
CX 0000
:100
-RBX
BX 0000
:0
-NKILLHKA.COM
-W100
Writing 100 bytes.
```

-Q

以后只要在 DOS 提示符下执行 KILLHKA 就自动完成对 A 盘的查毒和解毒。

对于硬盘的消毒过程较之软盘略为复杂一些。先在正常、无病毒感染的机器上通过 DEBUG 程序建立一个消毒病毒程序 KILLHKB.COM, 具体过程如下:

(1) 拷贝正常的主引导区至内存中(偏移地址 0200H)

```

-A100
XXXX:0100 MOV AX,0201
XXXX:0103 MOV BX,200
XXXX:0106 MOV CX,1
XXXX:0109 MOV DX,80
XXXX:010B INT 13
XXXX:010D INT 20
XXXX:010E
-G=100

```

Program terminated normally

(2) 编制消毒病毒程序:

```

-A100
XXXX:0100 MOV AX,0
XXXX:0103 INT 13
XXXX:0105 MOV AX,0201
XXXX:0108 MOV BX,0400
XXXX:010B MOV CX,1
XXXX:010E MOV DX,80
XXXX:0111 INT 13
XXXX:0113 JNB 117
XXXX:0115 INT 20
XXXX:0117 MOV AX,[0489]
XXXX:011A CMP AX,90B1
XXXX:011D JNZ 0115
XXXX:011F MOV SI,0403
XXXX:0122 MOV DI,0203
XXXX:0125 MOV CX,8
XXXX:0128 CLD
XXXX:0129 REPZ
XXXX:012A MOVSB
XXXX:012B MOV SI,0570
XXXX:012E MOV DI,0370
XXXX:0131 MOV CL,90
XXXX:0133 REPZ
XXXX:0134 MOVSB
XXXX:0135 MOV AX,0301
XXXX:0138 MOV BX,0200
XXXX:013B MOV CX,1
XXXX:013E MOV DX,80
XXXX:0141 INT 13
XXXX:0143 JMP FFFF:0000
XXXX:0148
-RCX
CX XXXX
:400
-RBX
BX XXXX

```

:0

-NKILLHKB.COM

-W100

Writing 400 bytes.

-Q

此时,一个专用于消除硬盘的“香港病毒”的软件就形成了。把该文件拷入被感染的机器内并执行 KILLHKB,若该机未感染香港病毒,则程序正常退出,不作任何动作。若发现香港病毒,则自动消除,并进行重新启动,从而制止病毒的再次感染。

该消毒程序无须用健康系统盘重新启动系统,所以使用非常方便。它可放在系统自启动批文件 AUTOEXEC.BAT 中,起到防毒、消毒的作用。

## 谈数据文件“死而复生”的技巧

兰州军区司令部作战部(730000) 李瑛彬

在计算机世界里,一个个数据文件三天两头被判为“死刑”,绝非偶然。但也有一些文件被误判为“死刑”,那简直是软件研制过程中时间、人力、经费、成果的极大损失。下面介绍几种文件被删或突然丢失后被挽回的方法,供参考。

### 一、文件或目录被损坏后的恢复法

使用 DOS 命令 RECOVER.COM 进行恢复。恢复 A 软盘批量文件:

C>recover a:✓

恢复 A 软盘的 LYB.WSF 单个文件:

C>recover a:Lyb.wsf✓

### 二、文件被误删后的恢复法

使用 PCTools 工具软件中的 U 命令,步骤如下:

C>pctools✓

按 F10 键选择即要恢复的软盘驱动器,按 F3 键后用 U 键命令,此时系统提示所有被删除的文件,文件打头的字符全部为?号。如,恢复 A 软盘的 LYB.WSF 文件,系统提示为?YB.WSF。可将光标移至这个文件上,将?号改为 L,再打 G 键命令,文件马上就被恢复,然后按 Esc 键返回 DOS 系统。

### 三、写盘过程的文件挽救法

笔者正在往 A 软盘写数据文件 909XT,突然别人将盘拿走,出现以下现象。

Not ready error reading drive A

Abort,Retry,Ignore,Fail?

出现这种情况时,千万不能乱击键或进行热启动。待把那张软盘插入 A 驱动器后,按 R 键即可将文件全部存入 A 软盘。

### 四、数据文件未做写处理丢失的恢复法

上述三种文件的特点是都有数据基础,因为它曾

驻留过磁盘。但有时在编写程序或打印文字材料时,在写盘失败后进行第三种方法处理仍不奏效,或干脆未做访问软驱的动作,直接返回 DOS,造成文件丢失的现象更为可悲。如出现这种情况,笔者在此提出严肃忠告,不要运行其它程序,更不能关机或热启动。因为此时文件内容仍驻留在内存之中。如果运行其它程序,会造成因其它程序占用内存较大空间而冲掉内存中的数据文件,重新关机或热启动会造成数据文件干脆丢失,而无法挽救。此时可用 DEBUG.COM 文件进行恢复。现假设本文未做访问软盘驱动器而突然丢失的情况为例,对其进行恢复的方法。

```
C>debug
-s 0 ffff 谈数据文件“死而复生”的技巧,搜索文件首地址
-66A0:5AB8 ;显示首地址
-na:909xt ;将恢复文件存入 A 盘
-rcx
-CX 0000 ;显示寄存器
:7190 ;根据文件长度给出偏移量
-w5ab8 ;存入首地址
Writing 7190 bytes
-q ;退出
```

用 DEBUG.COM 修改完之后,再用汉字编辑程序调入 A:909XT 文件就会发现文件完好无损。

---

## 数据自动存盘程序

四川大学数学系(610064) 汪海波

### 一、用途

(一)当有文字编辑软件进行文字数据录入时,突然的停电,将使您的工作前功尽弃。用手动存盘可以避免一些损失,但一来操作麻烦,二来由于疏忽不能及时存盘的事常常不幸地发生;而市场上出售的不间断电源价格昂贵。笔者编写的这个程序较好地解决了这个问题。本程序对录入的文字数据具有自动存盘功能,在一般情况下,几乎不影响用户的正常操作,而一旦停电或由于某种原因造成微机死锁,又未能及时存盘,本程序将使您很方便地恢复刚录入的文字数据。

(二)具有“黑盒子”的功能。将本程序放在硬盘上,在自动批文件里加上一两条命令,就具有对用户上机操作进行自动监视记录的功能。这对于微机管理员及时查出系统故障的原因,或教师了解学生上机作业情况都有很大的作用。

### 二、原理

本程序通过修改 INT 16H 键盘管理中断向量,指向新的中断程序,对键盘输入进行记录。当输入字符数达到64字节(可改成其它的数字)时,就自动存盘一次。键盘输入信息记录在数据文件 SAVE.DAT 中。

编写程序时,使用了下面一些技巧:

1)每次存盘时,移动文件读写指针到文件尾部,只存入64个字节。这在一瞬间即可完成,因此对用户的正常操作几乎没有影响。

2)每次启动本程序,自动将上次的记录文件 SAVE.DAT 改名为 SAVE.BAK,并重新建立新的记录文件 SAVE.DAT,从而保证 SAVE.DAT 中记录的是本次启动后的键盘输入。上次启动后的键盘输入可在 SAVE.BAK 中查到。

### 三、程序

本程序用汇编语言写成,经编译后的执行文件 AUTOSAVE.EXE 只有1K字节。

将文字编辑软件盘插入 A 驱动器,将含有编译程序 MASM.EXE 及连接程序 LINK.EXE 的软盘插入 B 驱动器,按下列步骤操作:

1)在 DOS 提示符下打 WS<CR>(CR 表示按回车键,下同)选择 N(编辑非文书文件)项,取名 B:AUTOSAVE.ASM,然后按下面的源程序清单输入计算机。输完后,打 Ctrl+KX,存盘退出。

2)在 DOS 提示符下打:

```
B)MASM AUTOSAVE;<CR>
```

```
B)LINK AUTOSAVE;<CR>
```

即生成可执行文件 AUTOSAVE.EXE。

### 四、使用

(一)在一张磁盘上建立文字编辑软件,并将 AUTOSAVE.EXE 拷入,插进 A 驱动器。打:

```
A)AUTOSAVE<CR>
```

则程序自动在 A 盘根目录下生成一个名为 SAVE.DAT 的文件,在此以后的键盘输入都将记入这个文件里。现在,就可以进入文字编辑软件,放心地进行文字数据的录入了。

这时,即使突然停电,也不必担心,所录文字数据都已存在 SAVE.DAT 中了。只需读入这个数据文件,删去其中的一些多余的文字符号(这些多余的文字符号是由于您的输入错误引起的),即可继续往下进行文字数据的输入了。

(二)“黑盒子”功能。将 AUTOSAVE.EXE 拷入硬盘根目录下,在自动批文件 AUTOEXEC.BAT 中加入一条命令:

```
AUTOSAVE
```

即可。这时,每次启动后,在 SAVE.BAK 中即可查到上次启动后的键盘输入情况。

### 五、说明

(一)本程序的记录功能很强,对所有可显示的键盘输入,包括错误的输入,都能如实地记录下来。因此,当用来恢复文件时,对这些多余的文字符号应予删除。

(二)在“黑盒子”功能中,执行本程序的命令在自动批文件中的位置对记录的内容将有很大影响。如果



AUTOSAVE 是 AUTOEXEC. BAT 的第一条命令, 则 SAVE. DAT 中记录的将只是 ACSII 码, 或汉字的外码(输入码), 而不能记录汉字的输入。如果需要记录汉字, 则 AUTOSAVE 必须放在装载汉字系统的命令之后。例如, 执行了 WBZX(五笔字型装入模块)之后, 再执行 AUTOSAVE, 就可记录用五笔字型输入的汉字。

(三) 由于每次开机启动都将刷新 SAVE. DAT, 因此如果多次开机, 则最后只能看到上次开机后的键盘输入情况。为了全面记录一段时期内的键盘输入, 可以在 AUTOEXEC. BAT 中 AUTOSAVE 命令之前, 加上一条命令:

COPY SS+SAVE. DAT SS>NUL 这样, 在 SS 中就可查到一段时期以来的全部键盘输入。注意对 SS 文件进行定期清理, 否则其长度将迅速增加, 而占去很大的磁盘空间。

(四) 本程序在 PC 机上运行通过。

(五) 欢迎读者对本程序的不足之处, 及其功能改进, 提出宝贵意见。

```
cseg segment para public
assume cs:cseg, ds:cseg, es:cseg

stk db 20h dup('stack') ;堆栈区
count dw 00h ;键盘计数
lim dw 40h ;每次存盘字节数(可改)
int16 dd 00000h ;原 INT 16H 中断向量
stk d dd 00000h ;保存原栈段指针
file1 db '\save.dat',0 ;数据文件名
file2 db '\save.bak',0 ;数据文件名
int16n proc far
pushf ;保存标志
or ah,ah ;非0号功能
jnz old ;转 old
popf ;0号功能
pushf
call dword ptr cs:int16;从键盘读字符
pushf ;返回 ah=扫描码
push ax ;al=字符码
push bx ;保存寄存器及标志
cli ;清中断
mov word ptr cs:stk d,sp ;保存栈段指针
mov word ptr cs:stk d+2,ss
mov bx,cs
mov ss,bx
mov sp,0FEh ;修改栈段指针
sti ;开中断
push cx
push dx
push ds ;保存寄存器
push cs
pop ds ;数据段指向代码段
cmp al,20h ;是否为 ACSII 码?
jnb save ;是,则转 save
cmp al,0dh ;是否为0dh(回车)?
```

```
jz save ;是,则转 save
jmp on ;否则,转 on
save: ;存键盘输入
lea bx,keep ;bx 指向缓冲区首址
add bx,count ;加上计数
mov [bx],al ;字符码存入缓冲区
inc count ;计数加1
call checkc ;转子程序
cmp al,0dh ;是否为“回车”
jnz on ;不是,转 on
mov al,0ah ;是,
jmp save ;则再存一个“换行”符
on: pop ds
pop dx
pop cx ;恢复寄存器
cli
mov bx,word ptr cs:stk d+2
mov ss,bx ;恢复栈段指针
mov sp,word ptr cs:stk d
sti
pop
pop ax
popf
iret ;中断返回
old: popf
jmp dword ptr cs:int16 ;转原中断
checkc proc near ;检验计数器程序
mov cx,cs:lim ;检验计数是否达到
cmp count,cx ;每次存盘字节数
jb retc ;未达到,则返回
call write ;达到了,则转存盘子程序
retc: ret ;返回
chechc endp
write proc near ;存盘子程序
push ax
lea dx,[cs:file1] ;
mov ax,3d01h
int 21h ;打开 save.dat 供写
jb return ;出错返回
mov bx,ax ;文件句柄送 bx
xor cx,cx
xor dx,dx
mov ax,4202h
int 21h ;移动文件指针到文件尾
jb closef ;出错,转 closef
lea dx,[cs:keep] ;dx 指向缓冲区
mov cx,cs:lim ;字节数送 cx
mov ax,4000h
int 21h ;写文件
closef: mov ax,3e00h ;关闭文件
int 21h ;计数清零
mov word ptr cs:count,0
return: pop ax ;返回
ret
write endp
keep dw 20h dup(?) ;键盘输入缓冲区
```

```
int16n endp
```

### 主程序

```

autosave proc far
start: push cs
 pop ds
 push cs
 pop es
 lea dx,file2
 mov ax,4100h
 int 21h ;删去 \save. bak
 lea dx,file1
 lea di,file2
 mov ax,5600h ;将 \save. dat 改名
 int 21h ;为 \save. bak
 mov ax,3c00h
 mov cx,0

```

```

int 21h ;重建 \save. dat
mov ax,3516h
int 21h ;取 INT 16H 中断向量
mov word ptr int16,bx;保存 INT 16H 中断向量
mov word ptr int16+2,es ;在 int 16 中
mov ax,2516h
lea dx,int16n ;修改 INT 16H 中断向量
int 21h ;使指向 int16n
lea dx,keep
add dx,lim
add dx,0100h
int 27h ;程序驻留退出
autosave endp
cseg ends start
end

```

## PC 机软件加解密技术剖析(续)

华南理工大学计算机系软件专业88级 李文亮

加密和解密是对立的不可分割的两个方面。没有无所不摧的矛,也没有攻不破的盾,同样也不存在万能的加密和解密方法。一般说来,加密都是针对某些破译方法的,若知道了加密方法,解密相对就是很容易的了。因此,就需要编写反跟踪程序段,以求在最大可能的限度下,阻止破译者的跟踪分析。

反跟踪的方法是多种多样的,这里主要介绍一些常用的方法,在此基础之上,用户可以充分发挥自己的想象力,创造出更多更好的方法。

### 1. 反汇编

通常程序在装入内存之后不再改变,因此用户可以用 DEBUG 等调试工具装入程序,然后利用反汇编命令得到程序清单。实用的加密系统则实现了对自身代码的逐段加密,在程序运行过程中逐块解密执行,并破坏已执行完的程序块。这样,任意时刻内存中都不存在完整的程序,只有当前程序块是可见的。对反跟踪程序的加密算法,一般要求算法简单易行,不要消耗太多的时间,而且要有较好的保密性。常用有以下四种方法:

(1)逻辑运算法。即将原代码与参量逐字节进行逻辑运算而形成密码。如采用逐字节异或(A XOR B=C 加密, C XOR B=A 解密)。如果参量也按一定的规律来生成,保密性更佳。

BASICA 文件的 P 方式存储就是根据这一原理来

实现加密的。下面来看 CCCC.COM 中的一个片断。

```

R
AX=160E BX=0195 CX=163E DX=0000 SP=E8DF
BP=0100 SI=01E5 DI=0000
DS=160E ES=160E SS=160E CS=160E IP=01DC
NV UP EI PL NZ AC PE NC
160E:01DC B94500 MOV CX,0045
-U 1DC 1E3
160E:01DC B94500 MOV CX,0045 ;这里数据段和代码
段相同
160E:01DF 90 NOP
160E:01E0 000C XOR[SI],CL ;异或一个字节代码
160E:01E2 46 INC SI
160E:01E3 E2FB LOOP 01E0 ;循环执行

```

(2)位移法。这种方法通过将一字节或字节的循环信息左或右移若干位来实现。如 FBH 循环右移3位,就变成7FH。循环的位数也可以通过某一算法来确定,同时,PC 机上的循环移位可以通过一条机器指令来完成,既有较强的加密性能,实现也很方便。这种方法的另外一种形式是通过把原代码加或减去某一个数值来形成密码。

(3)交换法。通过把字或字节信息中的某几位进行互换,以达到加密的目的。

(4)错位法。即根据指定的代码对换表,或根据一定的规律把原码变换为另一代码。如 A→K, F→+, …。这种方法的加密性能最强。但前者需要一块空间来存放对换表,而后者在规律被得知的情况下可轻易解密。

另外,还可以采用如 DES 等一些保密性更强的算法,但这样一来,解密过程就更为复杂,要以大量的时间为代价。由此可见,加密往往需要以时间和空间来换取加密性能。

单一地使用某种方法,是很容易被破解的。实用的加密系统都是巧妙地综合利用多种方法,实现分块多

层次可变的加密,最后还往往需要利用密钥来参与解密运算,这样就大大地增强了解密的难度。

对付上述手段,解密时只能逐块跟踪,逐块解密;积少成多,最终一定可以看到程序的全貌。同时,还要记下一些重要参数,如当前块的首末地址,使用到的一些参数块地址等。

反汇编的另一种方法,就是利用 PC 机的一些特殊指令,实现指令的重叠;或者在程序中穿插一些无用代码,造成反汇编的失败。

对付这种方法,用 DEBUG 反汇编的效果最差,最好利用 SOURCER 中的代码类型 FRAGMENT 来分析这个片断。SOURCER 是一个高级反汇编工具,它可以直接从内存或文件生成原代码或列表文件,并有多种说明,效果比较好。

最后介绍一种“指令逆行”的方法,它几乎达到了无懈可击的地步,具有最强的保密性。这种方法是作者在彻底剖析过数个加密系统之后,发现并提出来的一种新型的加密方法,并已经在 IBM 机上具体成功实现。附录3提供了全部详细的程序清单。

通常 PC 上机程序都是由低地址向高地址逐条指令排列和执行的,用调试程序可以方便地进行反汇编和跟踪。但是如果程序代码不按这种方式排列,而是由高地址向低地址反向排列,运行时也按这种反向顺序进行,那么,程序就无法反汇编导了。同时,跟踪程序总是按进行跟踪执行的(除了 TURBO DEBUGGER 可以在特定地情况下“逆向”执行指令外,但这种“逆向”还是要求指令是按由低地址向高地址顺序存储的),这样一来跟踪也就无法进行了。

借助于 PC 机的单步中断,可以很方便地实现以上构思。可以这样设想:当每一条指令执行完之后,将产生单步中断,利用该中断使一条指令复原,然后执行该指令;指令执行完后,又产生单步中断,根据两次中断的返回地址,即可算出该指令的长度,据此向低地址方向找到下一条指令的起始地址,然后再把指令复原,再执行,……。这样,程序的运行就好像是从高地址向低地址方向进行。

其实我们可以看到,指令代码是否逆向排列,程序是否逆向运行,这还不是关键。关键在于每条指令执行完后才产生下一条指令代码,每条指令都是在被执行时才恢复。这样一来,跟踪者就根本无法进行反汇编和跟踪,只能老老实实地一条一条指令地恢复。假如有数

百条指令,并且每条指令都被加密,那么,要把这么多的指令全部恢复过来,将是一件非常“痛苦”的事情。而加密者只需编写一小段程序,瞬间即刻完成加密,非常方便。根据这一设想,我们可以产生无数的方法,“指令逆行”只是其中的一种。

根据“指令逆行”的原理,逆程序段中一般不能含有循环和回向转移,因为在程序运行中已经将原来的代码破坏了;另外,中断调用时要作特殊处理,因为在中断处理程序中,指令是顺序排列执行的;同时,程序逆行过程中还要禁止硬件中断,防止在执行诸如 INT 08H 时钟中断等服务程序时逆行引起的错误。

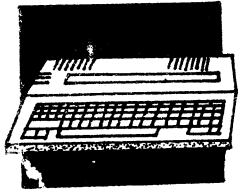
PC 机上程序要进入单步状态要有两个条件:一是设置了单步中断服务程序,二是把标识寄存器(FLAG)的中断允许位置为1。FLAG 的第八位为单步中断位,此位为1时,每执行完一条机器指令之后,CPU 就会产生单步中断,进入单步中断服务程序。在 DOS 下单步中断只有“IRET”一条指令,DEBUG 等调试程序每次执行 T 或 G 时,都把单步中断01和断点中断03改向自己的中断服务程序,这导致被调试程序的单步和断点服务程序永远无法执行。这样一来,调试程序就无法对逆程序进行跟踪了。我们用修改中断向量来实现单步中断程序的设置,用 IRET 来实现把单步中断位置1。

下面通过一个例子来说明该方法的实现。

由于8086系列芯片的指令最长为6个字节,所以在 INT 1中每次把指令代码前的5个字节移到指令代码后,就一定组成一条完整的指令;同时,这样作还破坏了上一条指令。这里用代码0F 0H(指令“LOCK”)来作为逆程序的结束标志,每条指令都用逐字节异或0F 0H的方式加密。也可以采用别的方法。在遇到中断调用指令(指令代码以“0CDH”开头)的时候,必须改为顺序运行的方式,并在中断调用完成后再恢复为逆行方式。中断后的六条指令就是为此而特意安排的。要注意的是,这里中断调用结束后的返回地址(由 LEA AX,G1确定)必须作一些修改,因为逆行后指令地址已经发生了变化。而且,逆程序段的入口处那条指令执行前不会产生单步中断,指令应该顺序放置。

这段程序仅作演示用,在 TURBO ASM 2.01上编译连接成 COM 程序后,可正确执行,对其它的编译器,密码段的跳转地址可能要作修改。

(待续)



## 学习机之友

# 推荐一个扩展 BASIC 语句——拆字符串语句

北大分校数学系(100083) 王 凯

拆字符串语句是在 CEC 机或 APPLE 机上实现的一个扩展 BASIC 语句,其功能是将一个字符串拆成若干子串,它可以按任意指定的分隔符,灵活快速地将一个字符串分解,并把分解后所得的子串赋给指定的多个变量,因此具有广泛的用途。

### 一、从改善数据结构说起

以下程序可列表显示某班学生的选课成绩。

#### 【程序一】

```

10 HOME
20 PRINT TAB(3)“姓名”TAB(13)“课程 A”TAB(20)
 “课程 B”TAB(27)“课程 C”
30 READ N;FOR I=1 TO N
40 READ N$,F1$,F2$,F3$
50 PRINT TAB(3)N$ TAB(14)F1$ TAB(21)F2$
 TAB(28)F3$;NEXT
60 DATA 30
100 DATA 张兴华,98,,84
110 DATA 王 涛,,78,82
120 DATA 李 英,,


```

在程序中学生的姓名、各门选课成绩是由 DATA 语句提供的,数据必须按规定的格式输入;前后次序绝不可颠倒,缺项(未选某一门课)时必须连续地输入数据分隔符——逗号。这种严格的规定对于数据的输入、核查、修改、增删都带来了不便,特别是数据个数较多时更是如此。

一种理想的格式是:同一个学生的各门成绩前的分隔符能换成不同的字符以示区别,例如换成 A、B、C,以分别代表三门不同的课程;而且它们的次序可任意掉换,缺项时也不必输入该项前的分隔符。按此设想,张兴华的成绩可表为 A98 C84或 C84 A98,而李英三门课均未选,其成绩处可以什么都不写。

无疑这种格式有极大的改进,它被采用有赖于使用拆字符串语句,现将程序一做如下修改。

```

5 CH=37120;REM 拆字符串语句解释程序的入口地址

40 READ H$;CALL CH EXP H$ ON “N$ AF1$ BF2
 $ CF3$”

100 DATA 张兴华 C84 A98
110 DATA 王 涛 C82 B78
120 DATA 李 英


```

修改后程序的功能完全没有变,但 DATA 语句中采用了理想的格式。

当循环变量 I=1 时,执行 40 句中的 READ 语句后,H\$ 的值为“张兴华 C84 A98”,再执行其后的 CALL 语句,实现了对拆字符串语句解释程序的调用,从而保留字 EXP 打头的拆字符串语句被执行,串变量 H\$ 被拆成:N\$=“张兴华”,F1\$=“98”,F3\$=“84”,而 F2\$ 为空串。

### 二、拆字符串语句的语法规则及基本功能

调用拆字符串语句的解释程序来执行该语句需采用以下复合语句:

(行号) CALL <入口地址> <拆字符串语句>

注意:CALL 语句与拆字符串语句之间只能有空格,不能把它们分为两个单独的语句。

拆字符串语句的语法规则为:

EXP<串变量> ON<串表达式>

这里 EXP 和 ON 是原 BASIC 保留字。EXP 之后的串变量是被拆串变量,ON 之后的串表达式经常是一个串常量或串变量,其值提供了拆字符串的依据,为简便起见,称此式的值为 ON 表,此表应符合以下规定:

<首串变量><字符1><串变量1>.....<字符 N><串变量 N>

其中的各串变量叫结果串变量,各字符叫分隔符,它们可以看作是其后的串变量的前缀,首串变量是不可缺的,而其后的分隔符和串变量必须成对。

结果串变量可以是简单变量,或下标变量,须注意的是:若其中的某一简单变量在执行此拆字符串语句前未曾赋值,那么在 ON 表中它绝不可排在任何下标变量之后!

除 CEC 机的汉字识别符(代码 \$7F)以外,分隔符可以是任意的 ASCII 字符,由于允许空格作分隔符,故 ON 表中不能有多余的空格。

当分隔符不只一个时,它们不必全相同,也不必全不同,下面说到“ON 表中某种分隔符的个数为 N”,指的就是该表中共有 N 个分隔符取的是同一个 ASCII 字符,此时表中可能还有别种分隔符。

拆字符串语句的基本功能有如下述。

1. 完全分解的情况,此时 ON 表中每一种分隔符的个数都不少于被拆串中该种字符的个数。

首先,解释程序依 ON 表提供的分隔符,把被拆串拆成若干子串(可能有的是空串),除首子串外,每个子串前都有一个 ON 表内的分隔符,可把它看作是此子串的前缀,每个子串内都不含任何分隔符,此时被拆串得到了完全的分解。

第二步,把首子串赋给首结果串变量,把其余每个

子串(不包括前缀)按下述规定赋给与它有相同前缀的结果串变量:对同一种前缀而言,依由前至后的次序,把第K个子串(空串也算数)赋给第K个结果串变量。

若ON表中某种分隔符的个数N多于被拆串中所含的该种字符的个数M,则后面的N-M个结果串变量变为多余的,它们均得到空串值,特别地,若被拆串为空串,则每个结果串变量均得到空串值。

执行下面的程序,可将含有加、减运算的算式拆开。

**【程序二】**

```
5 CH=37120
10 HOME:O$="B$+Z1$+Z2$-F1$-F2$=J$ "
20 READ A$:IF A$="00" THEN END
30 CALL CH EXP A$ ON O$
40 PRINT "B$:"B$","Z1$:"Z1$","Z2$:"Z2$","F1$:"F1$","F2$:"F2$","J$:"J$
90 GOTO 20
100 DATA 7+8-6-4+5=10
110 DATA 7+6-3=10
120 DATA 7-3+6=10
190 DATA 00
```

执行以上程序,三个算式均被完全分解,分解第一个算式将显示出:

B\$:7,Z1\$:8,Z2\$:5,F1\$:6,F2\$:4,J\$:10

分解第二、三两个算式后,显示出的结果均为:

B\$:7,Z1\$:6,Z2\$:,F1\$:3,F2\$:,J\$:10

其中Z2\$、F2\$的值为空串。

2. 不完全分解的情况,ON表中某分隔符的个数N少于被拆串中该种字符的个数M。

在此情况下,在被拆串所含的M个该种字符中,多余的后M-N个将失去分隔符的作用,因此在拆得的某些子串内还含有该种字符,被拆串得不到完全的分解。

例如在程序二中,增加以下两行:

```
130 DATA 8-7+4+2-5+3=5
140 DATA 10-7-2+4-3+4+2=8
```

执行程序二后,以上二算式均得不到完全分解。

在130句的算式中第三个加号失去了分隔符的作用,结果F2\$的值为:"5+3",在140句的算式中第三个加号和第三个减号均失去了分隔符的作用,结果Z1\$、Z2\$的值分别为:"4-3","4+2"。

不完全分解的情况似乎是完全不能容忍的,但实际上并非如此。假如我们在编一个题库检索程序时,允许用户通过键入题号的方式来检索,此时可在程序中用INPUT语句接收用户的键入。为方便用户,最理想的是规定用户的键入采用如下的格式。如检索第5至8题、第10题、第21题就键入:5-8;10;21及回车键,此时在INPUT语句中只需用一个串变量,例如A\$,来接收用户的键入,然后用拆串语句分两步(先按分号再按减号)分解串变量A\$,就可得到用户要检索的全部题号。

显然在做第一步分解时会产生这样的问题:如果ON表中的分隔符过多,常常会降低程序的效率,反之,不免出现不完全分解的情况。

其实这里有一个好的办法可解决困难,ON表中只需一个分隔符,可以多次执行拆串语句,逐步将A\$完全分解,读者不难看懂下面的程序。

**【程序三】**

```
.....
400 FOR I=1 TO N:N%(I)=0:NEXT
410 INPUT "请按规定格式键入要检索的题号:":A$
420 IF A$="" THEN 470
430 CALL CH EXP A$ ON "T$;A$"
440 CALL CH EXP T$ ON "T$-U$":IF U$="" THEN U$=T$
450 T=VAL(T$):U=VAL(U$)
460 FOR I=T TO U:N%(I)=1:NEXT:GOTO 420
470
```

在以上程序中下标变量N%(I)用来记录用户的选题情况,为了简明,程序中没有加入具有处理用户误操作一类功能的程序行。

**三、拆串语句的附加功能**

在程序二中,分解算式7+6-3=10与算式7-3+6=10所得的结果相同,有时我们可能对此感到不满,希望知道是先加后减还是相反。其实拆串语句的附加功能完全可以满足这样的愿望。

我们约定凡下面提到的序号均是由左至右,从0起数而得的。假设分解被拆串后,ON表中第K个串变量所得的子串在所有子串中的序号为Z(K),此子串的首字符在被拆串中的序号为S(K),则

$$Z(K)=PEEK(512+3*K)-128$$

$$S(K)=PEEK(513+3*K)$$

应说明的是:因要查看的单元位于键缓区,故只有在执行拆串语句后,还没有新的键入前查看,查到的值才有效,再有,如果某一结果串变量是多余的,则求得的Z(K)、S(K)的对应值自然失去原来的意义,此时Z(K)必<0,而S(K)的值是随机的。

在程序二中增加以下程序行:

```
50 PRINT "Z(K):";:FOR K=0 TO 5:PRINT PEEK(512+3*K)-128";:NEXT:PRINT
60 PRINT "S(K):";:FOR K=0 TO 5:PRINT PEEK(513+3*K)";:NEXT:PRINT
```

对以上所述的二算式,将分别列出Z(K)、S(K)的值:

```
... ..
Z(K):0 1 -85 2 -83 3
S(K):0 2 211 4 225 6
... ..
Z(K):0 2 -85 1 -83 3
S(K):0 4 211 2 225 6
... ..
```

因算式中加、减号各只有一个,第二、四两个结果串变量Z2\$、F2\$是多余的,故Z(2)、Z(4)为负值,Z(2)、S(2)、Z(4)、S(4)均无意义。

利用拆字符串语句的基本功能和附加功能,对一个字串进行一次或多次分解,常可较快地得到此字串内容的完整准确的分析。以下程序可用来检查一个字串中是否有大写元音字符:A、O、U、I、E,如果有则将其中的第一个这种字符及其位置序号列印出来。

**【程序四】**

```
5 CH=37120;F$(1)="A";F$(2)="O";F$(3)="U";F$(4)="I";F$(5)="E"
10 HOME;INPUT A$;I=1
20 CALL CH EXP A$ ON "B$"+F$(I)+"C$"
30 IF A$ <> B$ THEN PRINT "第一个大写元音字符为:"F$(I),"其位置序号为:"PEEK(516);END
40 I=I+1;IF I<6 THEN 20
50 PRINT "不含任何大写元音字符"
```

有些读者可能会想到,用 MID\$( )函数也可以实现同样的功能,但那样做,程序中必须要增加一重循环,执行的速度肯定也要慢得多。

**四、利用拆字符串语句可大量节省内存占用**

假设我们要编一个人事档案检索程序,档案内容保存在磁盘上,关于每一个人的信息分属姓名、性别、

……、单位、职务、……等 N 项,其中可能含可省缺项,为了能灵活快速地检索,通常在检索前要把一个文件完整地调入内存,现在让我们来看看,在此拆字符串语句能带来什么好处。

首先,拆字符串语句可改善数据格式,给建文件带来方便,第二,此语句还可用来节省内存的占用。

如果不用拆字符串语句,N 项信息得作为 N 个字段存于磁盘,调入内存时得占用 N 个下标变量,如果使用此语句,就可根据每一项的长短变化及是否可省缺,将 N 项信息组成较少的 M 个字段存于磁盘,调入内存时只占用 M 个下标变量。

一个串下标变量在变量表中要占三个单元,一个人的信息少占 N-M 个下标变量。假设 N-M=8,人数为50,便可节省近千个单元,对 CEC 机或 APPLE 机来说,这不能说是个小数目。

**五、拆字符串语句的解释程序**

解释程序在内存中是可浮动的,不应忘记的是:将其调入内存后,常需及时利用 HIMEM 语句来保护它。

```
9100-A9DD 20 C0DE 20 E3DF
9108- 20 60DDA0 00 B1 83 48
9110-C8 B1 83 48 C8 B1 83 48
9118-A9 B4 20 C0DE 20 7BDD
9120- 20 FD E5 A8 F0 5E A5 5E
9128-D0 02 C6 5F C6 5E A9 00
9130-99 01 02 B1 5E 99 00 02
9138- 88 D0 F8 A5 B8 48 A5 B9
9140- 48 84 B8 A9 02 85 B9 A2
9148- 00 A9 80 9D 00 02 8A 48
9150- 20 B1 00 20 E3DF 20 6C
```

```
9158-DDA0 00 98 91 83 68 AA
9160-E8 A5 83 9D 00 02 E8 A5
9168- 84 9D 00 02 E8 A5 B8 D0
9170- 02 C6 B9 C6 B8 B1 B8 C9
9178- 20 F0 D0 20 B1 00 F0 07
9180-C9 7F D0 C7 4C C9 DE 68
9188- 85 B9 68 85 B8 68 85 9F
9190- 68 85 9E 68 F0 68 85 9D
9198-A9 80 85 A7 86 A1 84 A0
91A0- 84 A6 B1 9E C9 7F D0 04
91A8-C8 C8 D0 0E A2 00 DD 00
```

```
91B0- 02 F0 0E E8 E8 E8 E4 A1
91B8- 90 F4 C8 C4 9D 90 E3 B0
91C0-0A E6 A7 A5 A7 9D 00 02
91C8- 20 84 A6 84 A5 8A A6 A0
91D0- 85 A0 BD 01 02 85 A2 BD
91D8- 02 02 85 A3 98 A0 00 E5
91E0-A6 91 A2 C8 A5 A6 9D 01
91E8- 02 18 65 9E 91 A2 C8 A5
91F0- 9F 69 00 91 A2 A4 A5 C8
91F8- C4 9D 90 A4 F0 CB 60
```

## 也谈内存信息的打印

北京航空航天大学机械厂(100083) 张亭

本刊91年第3期《超级8~40单元打印》一文中介绍的程序弥补了监控打印内存信息功能的不足,但该文程序仍然存在一些缺点:只适用于80列打印纸,若用132列打印纸还是存在相当大的浪费;打印行长不能少于8个单元,因此无法采用边角空白的废打印纸;存放在\$2000~\$20A6并不能避免与其他机器语言相冲突;不能连续打印多份;程序功能过剩,有嫌冗长,达到了167个字节。

本人以46个字节的简短程序解决了上述问题:可在一行中打印1~256之中任意个单元;可浮动,移到内存的任意空闲地址都能照常运行;运用简单的技巧就可连续多份打印。

使用之前,将首地址存\$06、\$07、末地址存\$08、\$09,程序首地址加上\$0A的单元(本例中为\$30A)存放一行打印的字节数,0表示256个(本例中为24,适用于80列打印纸)。

很多缺乏经验的人一听说打印就先用 PR # 立即命令接通打印机,结果把以后的操作步骤全部记录在打印纸上。本人主张不预先联机的打印,具体办法如下:

键入本程序,并正确置好首尾地址、一行字节数之后	
监控程序中	* 1P^300G
BASIC 状态有 DOS	]PR #1;FOR I=1 TO 3;CALL 768;?:?:NEXT
BASIC 状态无 DOS	]PR #1;FOR I=1 TO 3;CALL 768;?:?:NEXT;PR #0
备 注	1. 打印机假设在1号槽 2. P^表示 CTRL 与 P 同时按下 3. 假设共需打印3份,每份间隔2行

当该程序移到其他地址运行时,表中的300G和CALL768就都要作相应的调整。

0300— A0 03 B6 06 96 3C 88 10—F9 A9 18 85 FF 20 92 FD  
0310— A9 A0 20 ED FD B1 3C 20—DA FD 20 BA FC B0 0E  
C6  
0320— FF F0 E6 A5 3C 29 07 D0—E7 A9 AD D0 E5 60

## 增强中华学习机的音响功能

广东佛山永安路154号(528000) 杨云霄

中华机的发声电路只具备微机发声功能的最起码条件,只能奏出单调、无伴奏的单声部乐音。这与现今游戏机相比逊色多了,从而极大地限制了教学、游戏软件的音乐渲染力。虽然已有与它配套的音响合成部件出售,但价格昂贵、使用并不方便等,使音乐卡没能被大多数用户接受。

笔者从软件上着手,采用与众不同的发音原理编写这个程序。它大大地改善了主机的音响效果,使之可奏出双声部,各声道独立,并且可控制声道开或关、演奏速度可变、可奏出七个八度音(一般程序只能奏三或四个八度音)。在不改变原有乐音数据前提下,采用改变各声道音调控制单元的值,还可使同一首歌曲用几十种音调奏出,甚至和声。它可作为音乐程序中的子程序调用(用CALL24576或\$JSR\$6000)。

程序说明:音乐数据按音阶①音长①音阶②音长②(①、②各表一声道)存放。当乐曲为单声部时则令音阶②为音阶①高一个八度,对于乐曲为双声部但音长②与音长①不相等时则把较大值的那一个分为n个的出的个数与较小值的那个声道的下一个乐音按发音格式组合,其它如此类推。程序中\$6029单元控制演奏速度。\$602F、\$6030单元各控制一个声道的声调,它们的值均在\$00—\$FF之间,其对应单元的值越大则速度越慢、演奏音调越高。特别是当\$602F、\$6030单元的值为\$00时则关闭对应声道。音乐数据的存放是从\$610F始,以两个\$00值为结束。本程序设有录音输出,用于扩大音量,使音乐更动听。

乐谱数据表只列出C大调(即\$602F、\$6030两单元值为\$CC,其二进制代码11001100,“1”表示触发扬声器,“0”不触发)的四个八度。可通过改变这两个单元值来提高或降低,以获得七个八度。所有的拍子均分成较短的来演奏,使两声道和谐地交融于一起。

笔者打算近期内介绍其音乐数据生成的快速程序。同时,已编好几十首世界名曲,免费提供给读者。

乐谱数据表

1	1 \$ B5	1 \$ 56	1 \$ 26	1/16拍 \$ 3
#1	#1 \$ AB	#1 \$ 50	#1 \$ 23	1/8拍 \$ 7
2	2 \$ A1	2 \$ 4B	2 \$ 21	1/4拍 \$ 10
b3	b3 \$ 97	b3 \$ 47	b3 \$ 1E	2拍 \$ 22
3	3 \$ 8E	3 \$ 42	3 \$ 1C	4拍 \$ 52
4	4 \$ 85	4 \$ 3D	4 \$ 1A	
#4	#4 \$ 7D	#4 \$ 3A	#4 \$ 18	
5 \$ F5	5 \$ 76	5 \$ 36	5 \$ 16	
#5 \$ E7	#5 \$ 6F	#5 \$ 32	#5 \$ 14	
6 \$ DA	6 \$ 68	6 \$ 2F	6 \$ 13	
7 \$ CD	7 \$ 61	7 2C	7 \$ 11	
7 \$ C1	7 \$ 5B	7 \$ 29	7 \$ 0F	

注:对应为空格者表示此调溢出

\$ 6000	AE	2F	60	8E	79	60	A9	01
\$ 6008	20	35	60	20	4E	60	F0	08
\$ 6010	AD	30	60	CD	79	60	D0	03
\$ 6018	AD	2F	60	8D	79	60	AD	00
\$ 6020	C0	10	E8	8D	10	C0	60	0F
\$ 6028	61	B9	00	00	00	00	00	CC
\$ 6030	CC	00	00	00	00	AE	29	60
\$ 6038	8E	2A	60	0A	AA	BD	25	60
\$ 6040	85	00	8D	33	60	BD	26	60
\$ 6048	85	01	8D	34	60	60	A4	10
\$ 6050	D0	10	A0	01	B1	00	D0	0D
\$ 6058	AD	33	60	85	00	AD	34	60
\$ 6060	85	01	A9	00	60	0A	0A	AA
\$ 6068	AD	2A	60	20	F0	60	8E	2D
\$ 6070	60	8D	2C	60	88	B1	00	A8
\$ 6078	A9	CC	20	8B	60	A5	00	18
\$ 6080	69	02	85	00	A5	01	69	00
\$ 6088	85	01	60	E0	00	F0	4B	C0
\$ 6090	00	D0	04	A9	00	A0	01	8D
\$ 6098	2B	60	8C	2E	60	8E	2D	60
\$ 60A0	A9	00	8D	2C	60	A2	08	20
\$ 60A8	DB	60	AC	2E	60	88	D0	FD
\$ 60B0	CA	D0	F4	AD	2C	60	38	ED
\$ 60B8	2E	60	8D	2C	60	AD	2D	60
\$ 60C0	E9	00	8D	2D	60	90	13	AD
\$ 60C8	2C	60	38	E9	08	8D	2C	60
\$ 60D0	AD	2D	60	E9	00	8D	2D	60
\$ 60D8	B0	CB	60	0E	2B	60	90	0A
\$ 60E0	EE	2B	60	2C	30	C0	2C	20
\$ 60E8	C0	60	48	68	EA	EA	EA	60
\$ 60F0	8D	31	60	8E	32	60	A9	00
\$ 60F8	A2	08	4E	31	60	90	04	18
\$ 6100	6D	32	60	6A	6E	31	60	CA
\$ 6108	D0	F3	AA	AD	31	60	60	

# 计算各类债券储蓄收益利息 BASIC 程序

南京物资学校(210003) 陶文庆

随着改革开放发展经济的需要,城乡人民储蓄不断增长,国内证券市场正进一步开放、发展,购买国库券、企业债券的个人和单位越来越多。在债券发行时以面额价购债券的收益以及各类储蓄的利息,需要计算以予知或复核其效益;而中途以高于面额的款额从转让机构买入二手债券,其收益和收益率则是购买者更为关心的。怎样快速、准确地算出有关购、储的收益,以决定是购,是储?购多少,储多少?

这里提供了一个适用于各类储蓄、发行债券、二手债券收益利息计算的 BASIC 程序。只需要按微机提示键入相应内容的数据,电脑就会“动脑筋”帮助你:1、计算各类储蓄的利息、本息合计;2、计算购买各种发行期债券的到期利息和本息合计;3、计算中途买入二手债券的利息、本息合计及所购二手债券的后期(到期)收益率。

操作中,电脑要求键入“年利率%(.##);月息%(-#.##)”,这时如果是给月息,请在数据前加负号;如月息六厘六(6.6‰),应键入“-6.6”,以便识别;给年利率,则输入百分号前的数据即可:如年利率为10%,应键入“10”,年利率为9.5%,应键入“9.5”。为方便计算,本程序以月为单位计天数,一年按360天,一月按30天计。程序已在紫金Ⅱ及 LASER310上运行验证,其他机型也可移植。

```
10 INPUT "中途买入请键入8;发行期买债券或储蓄请键入9";B
20 INPUT "年利率%(.##);月息%(-#.##)";L;
 IF L < 0 THEN L = -(L/1000);GOTO 30
25 L = L/100/12
30 INPUT "发行(存入)年,月";F1,F2
40 INPUT "到期年,月";D1,D2
50 INPUT "买(存)入(债券为面额)多少元?";N
60 IF B = 9 THEN 200
70 INPUT "每100元买入价(元)";R;R = R/100
80 INPUT "买入年,月";Z1,Z2
90 GOSUB 260
100 RPINT "到期所购债券本息合计";N+(N*L*Y);"元"
110 PRINT "其中收益(利息)为";N+(N*L*Y)-(N*R);"元"
120 PRINT "您买这些债券用了";N*R;"元"
130 PRINT "到期收益率是:";
140 IF Z2 = D2 THEN T = (D1 - Z1) * 12 * 30
150 IF Z1 = D1 THEN T = (D2 - Z2) * 30
160 IF Z2 > D2 THEN T = ((D1 - Z1 - 1) * 12 + D2 + 12 - Z2) * 30
```

```
170 IF D2 > Z2 THEN T = ((D1 - Z1) * 12 + D2 - Z2) * 30
180 PRINT (N + (N * L * Y) - (N * R)) / (N * R) / T * 360
 * 100;"%"
190 RUN
200 GOSUB 260
210 PRINT "到期利息是:";N * L * Y;"元"
220 PRINT "本息合计=";N + (N * L * Y);"元"
240 RUN
260 IF F2 = D2 THEN Y = (D1 - F1) * 12
270 IF F1 = D1 THEN Y = D2 - F2
280 IF F2 > D2 THEN Y = (D1 - F1 - 1) * 12 + D2 + 12 - F2
290 IF D2 > F2 THEN Y = (D1 - F1) * 12 + D2 - F2
299 RETURN
```

## 哥德巴赫猜想 BASIC 程序的改进

浙江金华县傅村镇校(321037) 卢良红

本刊1991年第3、第10期发表了陈君佐同志的关于验证 Goldbach 猜想的 BASIC 程序,文后所附的大量结论,在我校师生中引起了强烈反响。笔者连忙将它送入中华学习机中运行,却意外的发现其运行速度实在太慢了,特别是奇数猜想,在探求1991的3285组解时,竟花费了近3个小时。仔细分析原程序,发现在每探求一解时,都要从头计算各质素数,耗费了不少无效循环运算,致使运行速度慢、且不易连续探求。笔者经多方编程试验,终于找到了解决的方法:首先采用素数的快速求解法,把所需验证的N以内所有质素数都存在数组A%(4000)之中,然后采用数据搜索法求解。这样大大减少了无效循环运算,节省了占机时间,提高了程序运行效率。在验证1991年的3285组解时,仅耗时12分10秒。在偶数猜想中,将数从6验证到2000实际耗时仅53分钟,速度比原程序提高了十余倍。

```
10 INPUT N;DIM A%(4000);A%(1) = 3;K = 1
20 FOR I = 6 TO N + 1 STEP 6;FOR J = I - 1 TO I + 1 STEP 2;E = SQR(J);FOR L = 1 TO K;IF (J/A%(L)) = INT(J/A%(L)) THEN 60
30 IF A%(L) > E THEN 50
40 NEXT L
```



```

50 K=K+1:A%(K)=J
60 NEXT J,I
70 PRINT "1,偶数猜想":PRINT"2,奇数猜想":PRINT"3,
 END"
80 PRINT "请选择1,2,3":GET A:ON A GOTO 90,160,
 260
90 L=2:FOR I=6 TO N STEP 2:P=0
100 IF A%(L)<I-3 THEN L=L+1:GOTO 100
110 E=L:FOR J=1 TO K:IF A%(J)>I/2 THEN 150
120 F=A%(J)+A%(E):IF F>I THEN E=E-1:GOTO
 120
130 IF F=I THEN P=P+1:PRINT P)"T"="A%(J)+"
 A%(E)
140 NEXT J
150 NEXT I:GOTO 70
160 L=2:FOR M=9 TO N STEP 2:A=0:C=1
170 IF A%(L)<M-6 THEN L=L+1:GOTO 170
180 D=L:FOR I=1 TO K:B=0:IF A%(I)>M/3 THEN
 PRINT "END":GOTO 250
190 E=D:FOR J=I TO K:IF A%(J)>(M-A%(I))/
 2 THEN 240
200 F=M-A%(I)-A%(J)
210 IF F<A%(E) THEN E=E-1:GOTO 210
220 IF F=A%(E) THEN A=A+1:B=B+1:PRINT A"
 C" "B" "M"="A%(I)+"A%(J)+"A%(E):IF J=I
 AND B=1 THEN D=E
230 NEXT J
240 C=C+1:NEXT I
250 NEXT M:GOTO 70

```

## 数组的动态删除

成都商业学校(610091) 冯端品

在PC机的BASIC语言中有一条ERASE语句,可以把不再需要的数组删除,以后又可以重新定义。这对于使用大型数组的程序节省内存空间是非常有用的。但是苹果机、中华学习机等却没有相应的语句,这

对于内存容量本就很小的八位机来说,实在是一个很大的遗憾。

我分析了苹果机、中华学习机上变量在内存中的存储结构后,发现只要对数组变量区的数据作适当的调整,同时修改相应的指针,就可以起到删除数组的作用。于是我用6502机器语言编了一段程序,解决了这个问题。

首先运行程序(\*300G或]CALL768),然后在你的程序中要删除某个数组时,先使用该数组一次(例如给数组中任意一个元素赋值),后面跟冒号和&命令,即可删除该数组。

后附的BASIC程序是删除数组的一个例子,其中30句就是删除数B,40句重新定义了数组B,且下标上界比原来的大,程序照样能顺利运行。

可以删除任何类型的数组,不管是实型数组还是整型数组,不管是数值型还是字符型,也不管它们是几维的,都可以删除。但一次只能删一个数组。

机器语言程序的地址是浮动的,可以移往任何区域,只是原来\$301、\$306单元的数据要作相应的调整,使它们指向新的入口地址。

```

300— A9 0B 8D F6 03 A9 03 8D
308— F7 03 60 A5 6B 85 42 A5
310— 6C 85 43 A5 6D 85 3E A5
318— 6E 85 3F A0 00 B9 81 00
320— D1 42 D0 08 C8 B9 81 00
328— D1 42 F0 21 A0 02 18 B1
330— 42 65 42 48 C8 B1 42 65
338— 43 85 43 68 85 42 A5 43
340— C5 3F 90 D7 D0 07 A5 3E
348— C5 42 B0 CF 60 A0 02 38
350— A5 6D F1 42 85 6D C8 A5
358— 6E F1 42 85 6E A0 02 18
360— A5 42 71 42 85 3C C8 A5
368— 43 71 42 85 3D A0 00 20
370— 2C FE 60
10 DIM A$(1,2),B(3),C%(2,1)
20 A$(1,0)="ABC":B(1)=1:C%(0,1)=2:PRINT A
 $(1,0),B(1),C%(0,1):PRINT
30 B(0)=0:&
40 DIM B(5):B(5)=7:PRINT A$(1,0),B(1),C%(0,1),
 B(5)
50 END
RUN
ABC 1 2
ABC 0 2
7

```

# FORTH 语言讲座

## 第一讲 概述

北京西四北三条10号 FORTH 应用研究会(100034) 丁志伟

应部分读者要求,从本期开始,我们将连续刊载丁志伟先生撰写的 FORTH 语言介绍,有关 FORTH 语言的特点及学习意义,在这篇概述中已有很好的说明,不再赘述。

前一时期,笔者在杂志上发表过几篇介绍 FORTH 语言的文章。这种语言的独到之处,引起了许多读者的兴趣,因此陆续收到不少来信,询问有关信息、寻找资料及软件,想进一步了解这个语言。FORTH 有很多特点,与 BASIC、PASCAL 和 C 语言差别较大,相比之下其应用不如这些语言那么广泛,资料也较少。据笔者知,有关书籍在国内出过十几种,大多已无法得到。港台美日资料也有一些。各种计算机刊物上有关 FORTH 的文章虽时有所见,但较分散,找起来并不容易。

软件方面,由于 FORTH 语言实现和改造比较方便,因此版本较多,估计国内至少有二十多种,较为流行者,APPLE 机上的,有 FIG FORTH、FORTH79、FORTH II 等;PC 机上,有 PC/FORTH2.0 版和 3.2 版、F-PC3.5 等。这个讲座就是以 APPLE 机上的 FORTH I D1.0 和 PC 机上的 PC/FORTH2.0 为基础,介绍 FORTH 语言。

本讲座是为了帮助广大想学习 FORTH 而又缺乏资料的读者而办的,考虑到《电子与电脑》读者有众多业余爱好者,因此将介绍得较为详细具体。

如果需要资料及软件,请与北京西四北三条10号,中国软件行业协会 FORTH 应用研究会秘书处联系(邮编100034),也可以打电话:655661。

FORTH 是一种计算机高级语言。

计算机语言出现已近半个世纪,发展到今天,已经出现过上千种高级语言。其中人们普遍使用的有几十种。FORTH 也在这几十种之中。每一种高级语言都有其特点。比如 BASIC 和 LOGO 是为初学者设计的,FORTRAN 是科学计算语言,系统设计常用 PASCAL 和 C 语言,COBOL 多用于事务处理,LISP 和 Prolog 是人工智能语言,Post Script 是一种类似 FORTH 的语言,多用于排版和高级打印机,等等。

FORTH 是什么样的语言?为什么叫做 FORTH?它有什么特点?已经有那么多高级语言,为什么还要学习使用它呢?这些问题,正是这个讲座所要解答的。

### 名称及由来

FORTH 语言是由美国计算机专家 Charles

Moore 在60年代于美国佛吉尼亚无线电天文台工作时发明的。当时将其用于控制巨大的射电天文望远镜。他在 FORTH 名著《STARTING FORTH》(中译本《FORTH 入门》)的前言中是这么讲的:

若干年前,我是把 FORTH 做为程序员和计算机之间的操作界面而开发的。

传统的计算机语言无法满足我的各种要求。诸如功能、灵活性和简捷等等。为了使这个语言全面包含高水平程序员所需要的各种性能,我在设计 FORTH 时没有遵循其他语言中的常规做法。这些性能中最重要的一点就是其扩充能力,即凡是所需要的功能都能被加入到原有语言系统中。

当我第一次把各种设想组合成一个程序实体,在当时的“第三代计算机”IBM1130上运行,其效果如此之好,以至于我认为它是“第四代计算机语言”。最初想把它命名为 FOURTH(第四代),但由于 IBM1130 机只能接受5个字符的标识符,于是将 FOURTH 改为 FORTH,这就是其名称的由来。

正因为如此,人们又常把这种语言称为第四代计算机高级语言。顺便说一句,FORTH 这个词,在英文中是向前的意思,这种语言确实有一种向前的精神。它虽不象 BASIC、C 等语言那样风行,却也吸引了众多爱好者,一直在稳步发展。关于计算机语言的第几“代”如何划分,是有不同看法的,这里不做讨论。不过 FORTH 的确具有一系列特点,足以使它与其他高级语言相区别,也正是由于这些特点,一些人感到难以接受,另一些人又会发现,某些用其他语言不易解决的问题,用 FORTH 来解决却很方便,或者在其他语言无法使用的场合,这个语言却能很好地工作。

### FORTH 语言的特点

FORTH 语言在概念、结构、程序表现形式及使用操作各方面都与大多数高级语言有不少差别。它具有一系列特点。如:扩充能力极强、词典结构、堆栈及相应的逆波兰表示法、完全结构化、非常高的时空效率、独特的解释/编译方法、穿线代码、溶汇于高级语言的结

构化汇编和可以方便地反编译,等等。这些特点,又是互相关联的。由于这些特点的存在,实际了解这个语言,对于喜欢接受新事物的人,能发现这是一片新天地;对于只了解传统高级语言的人,也会有耳目一新的感觉,能加深对计算机语言本质的认识。

在众多特点中,最重要的是对系统的可扩充(或者说可扩展)能力。限于篇幅,这里无法全面介绍各种特点,下面将重点介绍系统扩充,同时介绍其他与之关联的特点。

为了使问题更清楚,先来看看不具备扩充能力而大家又较为熟悉的 BASIC 语言在这方面的情况。比如,某些 BASIC 只能使用 10 进制数,而在与机器深层系统打交道时,使用 16 进制会更方便。这时会需要编一段 10 进制~16 进制转换子程序。这样的子程序较为通用,可以用于别处,编其他程序时可以将它搬过来加以使用。随着时间及经验增长,每个编程者都会积累一批这样的通用子程序。它们相当于制造业中的通用部件,利用它们可以减少编程工作量。

可是事情并不完美。使用子程序,总要把它和主程序一同使用,程序接口常常是容易发生问题的地方。变量的用法,参数如何传递,都有可能出错。虽说编写工作量减少了,但仍要精心对待。同时,由于子程序要与主程序一同使用,程序规模不但不会因为编程量减少而缩小,反而更加容易膨胀。随着程序规模增加,处理上也会越来越复杂。这样的语言,程序和系统是截然分开的,系统本身不会随着编程者的经验增加而变化。好象旅行,每次只能从一个起点出发,越是远行,就非得背上更大的包袱。

很多人都看到了这点。自然会产生一种愿望,能否有这样的一种方法,所需的功能,只要编一次程序,系统就会记住,以后用到时只要直接引用就行了,即系统具有学习能力。遗憾的是,大多数语言做不到这点,它们的功是固定的,不对用户开放。其实想开放也难以实现,这些语言内部结构太复杂,一般人难以理解。APPLE II 机问世以来,不知有多少人 BASIC 解释程序做过剖析,一部分人还做过改进工作,但成绩并不大,一般不过是打几块补丁而已。现在人们普遍使用的仍然是厂家提供的固化 BASIC。随着软件技术的发展,现在虽然已经出现了 PC 机的 Turbo C 这样的系统,操作上很方便,可以使用众多函数库。但问题的实质并没改变:系统本身的功能都是由厂家提供的,用户只能被动接受,想改进还是不太可能。

现在来看看 FORTH。

首先举一个简单的实际例子。本文写作期间,笔者正在使用 PC/FORTH 2.0 编程,这是国内 PC 机上最普遍的版本。它能利用 PC 机的彩色显示方式,但平常却并没有使用彩色。为了改进显示效果,笔者定义了一组色彩功能模块。这种模块在 FORTH 语言中被称为 WORD,国内翻译成“单词”,也叫“词”或“字”,这个版本提供的几个与色彩有关的词中,有两个很重要。一个是 FOREGROUND,用于改变前景颜色,它决定显示

字符的颜色;一个是 BACKGROUND,用于改变背景。想改变以后显示字符的颜色,先给出一个颜色编号,再使用 FOREGROUND。比如红色编号是 4,要用红色,就可在键盘上打入

```
4 FOREGROUND
```

以后显出的字符就是红色了。

利用这个 FOREGROUND,就可以定义出一组颜色词。程序是这样的(␣表示空格,下同):

```
:BLACK 0 FOREGROUND ; (黑)
:BLUE 1 FOREGROUND ; (蓝)
:GREEN 2 FOREGROUND ; (绿)
:LBLUE 3 FOREGROUND ; (浅蓝)
:RED 4 FOREGROUND ; (红)
:PINK 5 FOREGROUND ; (粉)
:YELLOW 6 FOREGROUND ; (黄)
:WHITE 7 FOREGROUND ; (白)
```

这是 FORTH 中最常用的编程方式,也叫定义。这几行程序,直接由键盘输入即可。当然也可以进入编辑状态,存入文本文件中,然后再编译。两种方式效果都一样,都会使 BLACK、BLUE……WHITE 这几个词进入系统,之后就能根据这几个词改变颜色了。接下来打入

```
RED
```

系统会用红色显示,再打入

```
WHITE
```

以后显示内容又会变为白色。

这种词可以在程序中被引用。与之类似,还能借助于 BACKGROUND 改变背景颜色。

讲述以上的例子,目的是想使读者通过实例了解 FORTH 语言所具有的扩充能力。对具体操作不理解也不要紧,以后会详细讲到的。这里的定义,又是编译过程,与其他语言相比,具有以下特点:

\* 每个新功能都要起一个词名,这里是 BLACK、RED、WHITE 等。如何起名,一般没有限制,编程者应该尽量选择含义清楚,便于记忆的名称。

\* 新词在使用上,格式与原系统一致,一经定义,就可以直接引用这些词,不必理会它们是如何定义的,极为方便。

\* 新定义的单词,一旦进入系统,就与原有系统浑然一体。系统并不区分哪个词是原有的,哪个词是新定义的。所有单词,在系统中地位都是一样的,实际上,一般的 FORTH 系统,大部分单词都是利用 FORTH 自身的扩充能力而扩充出来的。

\* 编译过程是一遍扫描,直接在内存中产生目标代码。不象其他语言那样在磁盘中产生中间文件,也不需要表格区,因此非常快速,开销极小。

\* 产生的目标代码非常简短,经常比汇编还要短,一个系统中可以定义出很多新词,其数量只受内存限制,大多数情况下不必担心系统中存放不下。

\* 新词一经编译,便进入系统,不必保存源程序。简单的定义可以直接从键盘输入,使系统立即增加新词,随后可以立即使用。

以上介绍了扩充能力,下面再来看看其他特点。

**词典结构。**为了便于使新词放入系统,FORTH采用了词典结构。系统的主要部分被称为词典,其中是按顺序存放的一个个单词。这些单词相当于其他语言中的命令、语句、函数和子程序等,实际上就是一个个个功能模块,系统的功能就是这些词的功能,每个词有一个词名。一般情况下,新词是通过词典中原有词组合而定义出来的,每定义一个新词,就把它放入词典顶端。整个系统就是利用滚雪球的方法建立起来的。反之,又可以用删除顶端一部分单词的方法来缩小词典规模。由于可以方便地扩充和压缩,使得系统非常灵活。可以看出,它是一个开放的系统。词与词之间是由链表连接在一起的,查询、添加、删除等,都要通过这条链,有人把它称为FORTH系统的生命线。

词典结构体现了FORTH语言的一个原则:简明。它易于理解,进行各种处理也非常方便,易于保证程序的正确性。简明还使得用户分析这个开放系统成为可能,突出的一点,是可以方便地对单词进行反编译。另一方面,它的功能又很强,可以解决各种复杂问题。

**模块化和结构化。**软件工程学倡导模块化和结构化编译方法。这要求所使用的语言具有与之适应的机制,FORTH语言正是具备了这种机制的语言。词典中一条条单词就是一个个模块,单词构成方法,能够较好地符合模块化所要求的便于分解、隐藏信息和模块独立性。FORTH又是天然的结构化语言,适宜于自顶向下的设计表达,具体编程时则是自底向上生成各个单词。在程序流程的控制结构方面,FORTH语言也非常好。它可以方便地实现各种条件判断、循环和分情况结构。如果所需要的结构在一个版本中不存在,还可以由用户自己扩充出来,这点笔者深有体会。值得一提的是,FORTH中没有GOTO这个词,因为不需要。在其他语言,对于GOTO使用不当时,会造成非结构化因素,经常是引起程序混乱的祸根,但有时不用GOTO却又难以解决问题。FORTH不存在这样的问题,这与它的高度模块化有关。更值得一提的是,FORTH中还能方便地容纳与高级成分相一致的结构化汇编,可以省去在普遍汇编中必不可少的标号。

**堆栈。**进行模块化编程时有一个非常重要的问题要解决。就是模块之间的参数传递。在其他语言中,一般是利用变量,可能因此带来复杂的问题。很多程序错误,都是因为对变量使用不当引起的。FORTH没有沿用其他语言中的做法,它使用了堆栈,这是与其他语言的重大区别之一。各种运算、操作,在需要处理和传递数据的地方,一般都是通过堆栈进行的,相应地,采用了一种特殊的运算表示法,称为逆波兰表示法,又叫后缀表示法。堆栈和逆波兰,有其自身的性质,内容很丰富,以后还要重点介绍。

**快速。**随着词典的层层扩充,高层次单词的速度会有所降低。因此词典低层的词应该非常快才行。FORTH速度非常快,而且还可以将某些对速度有重要影响的词(即所谓“瓶颈”部位的词)用汇编来编写,

以进一步提高速度。

**紧凑。**FORTH系统占内存很小。在其他语言由于内存紧张难以运行的场合,如各种8位微机、单片机和单板机系统,FORTH却能舒适地运行。

由于快速和占内存小,在很多场合下,这个语言常常成为取代汇编的首选高级语言。

**集成化环境。**使用FORTH时,解释、编译和磁盘存取都在同一个操作界面,使用上统一而方便。编辑器非常小,可以常驻内存,进入和退出编辑都很迅速。加之FORTH程序一般比较短,使得编程、调试都很方便,系统兼有解释和编译两种功能,在解释状态下,具有很好的交互功能,能象BASIC和LOGO那样方便地进行人机对话;另一方面,词典中的词都经过编译,运行效率又非常高。集成化环境,对于其他语言是近年来才发展起来的技术,比较复杂;而对于FORTH来说,是早已具备的性能,而且较为简单、自然。

### 发展简史

FORTH语言是在天文台开发的,最初用于射电天文望远镜的控制,随着美国射电望远镜的出口,流传到了世界各地。由于它的成功,70年代中期,国际天文学会把FORTH确定为标准计算机语言。

在最初10年,由于缺乏教科书和它的软件比较昂贵,这个语言传播速度比较缓慢。1978年,一些FORTH爱好者成立了一个FORTH兴趣小组(FORTH Interest Group 简写FIG),这是一个非赢利的组织。他们从一个通用模型出发,在当时流行的8080、8086、6800、6502和PDP-11等9种计算机上实现了FIG FORTH版本。由于他们的版本是公开版权的,又易于移植,大大促进了FORTH的广泛传播。现在我们国内在APPLE II及兼容机上,还流传着这个版本。早几年,宇航出版社的《FORTH系统》介绍的就是这个版本。PC机上普遍使用的PC/FORTH2.0也是在这个版本基础上发展起来的。

FIG不断发展,现在已成为遍布世界、会员达十几万的组织。FIG还很重视语言的标准化工作,先后制定了FORTH-78、FORTH-79、FORTH-83等标准。据介绍,现在正在进行新的标准修订工作,不久即将公布。

在国外,FORTH在天文、军事、航空航天、工业自动化、图形、医疗仪器、工作站等众多领域都有广泛应用。在国内虽不及国外广泛,但也取得了许多可喜的进展。

这个语言引进我国是在1980年,将其汉化,是在中国推广的重要内容,目前已有多种汉化版本。1987年,成立了中国软件行业协会FORTH应用分会,简写CFIG,使FORTH语言在我国发展进入了一个新阶段。学习它的人越来越多。1990年3月,机电部仪器仪表司决定将FORTH做为仪表行业的标准语言。

由于其自身的特点,加上广大爱好者的努力,可以预计FORTH语言还将有一个较大的发展。



# BJS-51单片机实验系统

## 模/数、数/模转换实验(续)

北京工业大学(100022) 张俊谋

### 三、实验操作和编程：

在实验时,应注意以下各项:

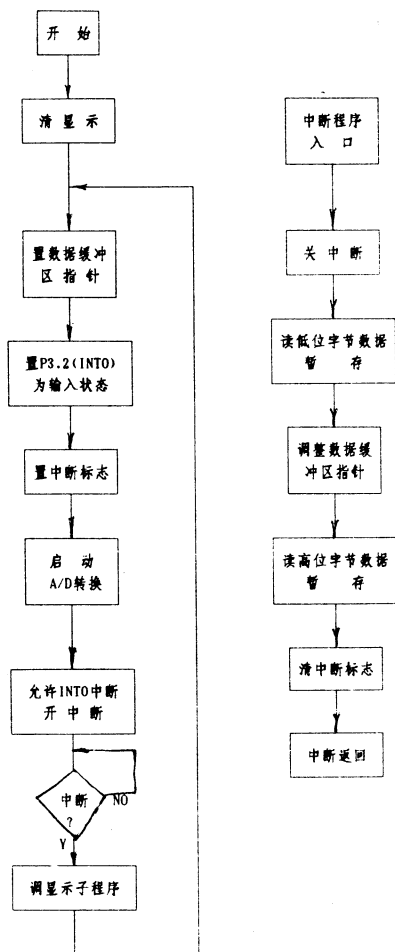
1. 实验前必须根据教师拟定的实验要求,选择相应的插块位置。

例如:在控制转换方式下,模拟信号为4~20mA的电流信号,希望采用中断方式输入数据,测定精度高,拟用外部参考电压。

根据上面的要求,选定插块位置。

- 中断响应方式            W1 置于2-3位置
- 控制转换方式            W2 置于1-2位置
- 直接输出方式            W3 置于2-3位置

框图1:



地址	机器码	行号	标号	汇编语言	注 释
8003		1		ORG 8003H	中断0 中断服务程序入口
8003	028280	2		LJMP INT01	转入中断服务程序
		3		:	
8200		4		ORG 8200H	控制转换, 中断方式采集
8200	9140	5	MAIN1:	ACALL CLR	清显示
8202	7820	6	MA12:	MOV RO,#20H	数据缓冲区首地址
8204	D2B2	7		SETB P3.2	设P3.2为输入口
8206	D2D5	8		SETB PSW.5	设置中断标志
8208	907800	9		MOV DPTR,#7800H	启动转换
820B	F0	10		MOVX @DPTR,A	
820C	D2A8	11		SETB IE.0	置外部中断0
820E	D2AF	12		SETB IE.7	开中断
8210	20D5FD	13	HERE1:	JB PSW.5,HERE1	未中断,等待
8213	7A20	14		MOV R2,#20H	
8215	128400	15	HERE1:	LCALL DISP	调显示转换结果子程序
8218	7D01	16		MOV R5,#1	
821A	128480	17		LCALL DELAY	调延时子程序
821D	DAF6	18		DJNZ R2,HERE1	
821F	128202	19		LJMP MA12	
		20		:	
8280		21		ORG 8280H	中断服务程序
8280	C2A8	22	INT01:	CLR IE.0	禁止外部中断0
8282	C2AF	23		CLR IE.7	关中断
8284	907800	24		MOV DPTR,#7800H	
8287	E0	25		MOVX A,@DPTR	读取低位字节
8288	F6	26		MOV @RO,A	暂存数据
8289	08	27		INC RO	调整缓冲区地址
828A	907400	28		MOV DPTR,#7400H	
828D	E0	29		MOVX A,@DPTR	读取高位字节
828E	F6	30		MOV @RO,A	暂存数据
828F	C2D5	31		CLR PSW.5	清中断标志
8291		32		RETI	中断返回

- 使用外部参考电压 W4 置于1-2位置
- 4~20mA 电流输入(差分) W5置于1-2位置 (电流)W6 插上短路块

2. 根据实验要求拟定程序框图和编制软件.

- (1)根据要求程序如框图1.
- (2)根据程序框图, 编制的实验程序如表.

3. 本实验中所使用的仪器要求比较高, 比如数字电压表宜采用5位半, 至少也应保证在四位半的水平.

#### 四、带数据寄存器的8位 D/A 转换芯片—DAC0832

1. 芯片特点:

- 分辨率为8位(精确到1/256)

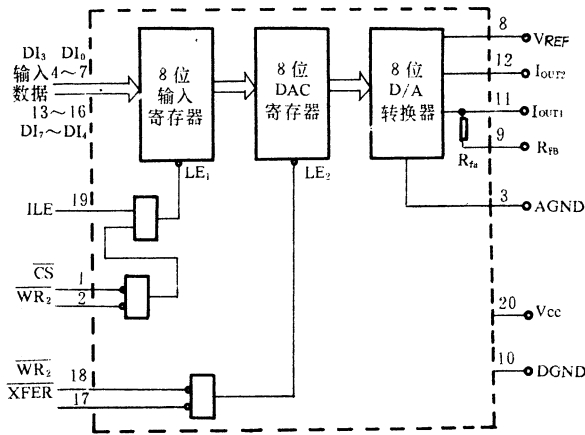


图4

- 电流稳定时间: 1 $\mu$ S
- 数字输入可实现双缓冲、单缓冲或者直接输入
- 只需在满量程下调整其线性度
- 单电源供电(+5V~+15V)
- 低功耗: 20mW

2. 芯片的结构原理与管脚功能:

DAC0832的原理性结构及外部管脚如图4所示。它由8位输入寄存器、8位 DAC 寄存器、8位 D/A 转换电路及控制电路所组成, 为20脚双列直插式封装。

各管脚的功能如下:

- DI0~DI7 8位数据输入端
- ILE 输入寄存器允许信号, 高电平有效
- CS 片选信号, 低电平有效
- WR1 输入寄存器写选通信号, 低电平有效
- WR2 DAC 寄存器写选通信号, 低电平有效
- XFER 传送控制信号, 低电平有效。与WR2一起用来选通 DAC 寄存器
- VREF 基准电压输入端
- Rfb 反馈电阻输入端
- Iout1 电流输出端1, 其值随 DAC 寄存器内容线性变化
- Iout2 电流输出端2。Iout1+Iout2=常数
- Vcc 电源电压输入端
- AGND 模拟地
- DGND 数字地

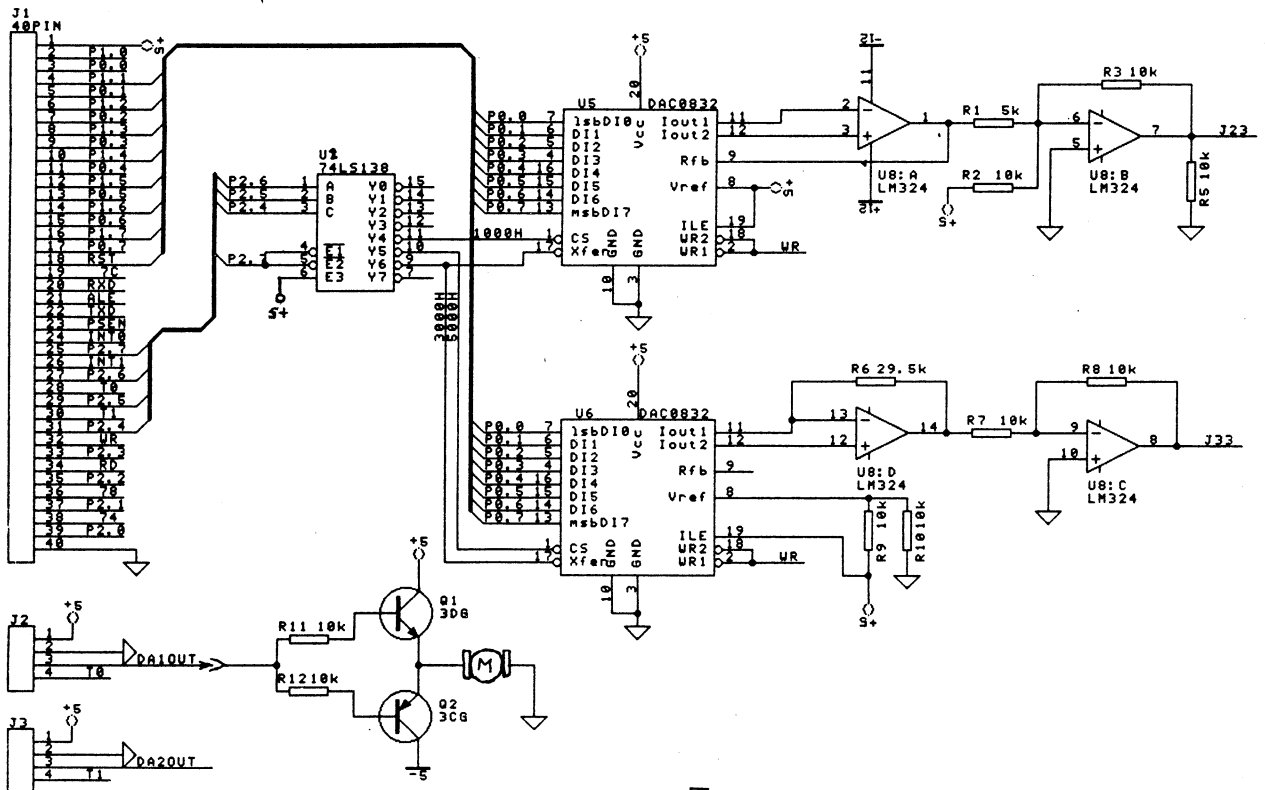


图5

### 五、D/A 转换实验电路的设计

为了满足对 D/A 转换实验的要求,实验电路采用双 D/A 转换器,并且可用于控制微型电动机的启、停和正反转,电原理如图5所示。

该电路包括 D/A 转换器、转换控制电路、输出电路及电机控制电路等几部分。

1. 转换控制电路:这部分由 U2(74LS138)构成,实际上是一个译码电路,由高位地址线 P2.4~P2.7 通过该译码器译出 1000H、3000H、5000H 三个地址供本部分电路使用。1000H 用于控制 U5 的写入寄存器的写入操作;5000H 用于控制 U6 的输入寄存器的写入操作;而 3000H 则用于将两个 DAC0832 输入寄存器中的数据通过 DAC 寄存器送入 D/A 转换电路部分而实现 D/A 的输出。

2. D/A 转换电路:这部分包括两个 DAC0832(U5 和 U6)。从前述控制电路的说明中可以看出,这两个 D/A 转换器采用的是双缓冲的接法,这种接法可以保证两个 D/A 转换器输出同步。

3. 输出电路:这部分包括运算放大器 LM324 和若干电阻。LM324 是一个四运放集成电路。从图5中可以看出 U8:A 和 U8:B 构成 U5(DAC0832)的输出电路。U8:A 是 I/V 转换器,U8:B 是一个偏置电路,使得在插座 J23 上得到  $-5.0V \sim +5.0V$  的双极性输出,用于控制电动机。图中 U8:D 和 U8:C 构成 U6(DAC0832)的输出电路,U8:D 是 I/V 转换器,U8:C 则是一个反相器,使得在插座 J33 上得到  $0 \sim 5V$  的单极性输出。

4. 电机控制电路:这部分包括晶体管 Q<sub>1</sub>(3DG)、Q<sub>2</sub>(3CG),电阻 R<sub>11</sub>、R<sub>12</sub> 和电动机 M,由 U8:B 输出的  $-5V \sim +5V$  的电压经晶体管驱动之后,带小电机作正转、反转实验。

程序1

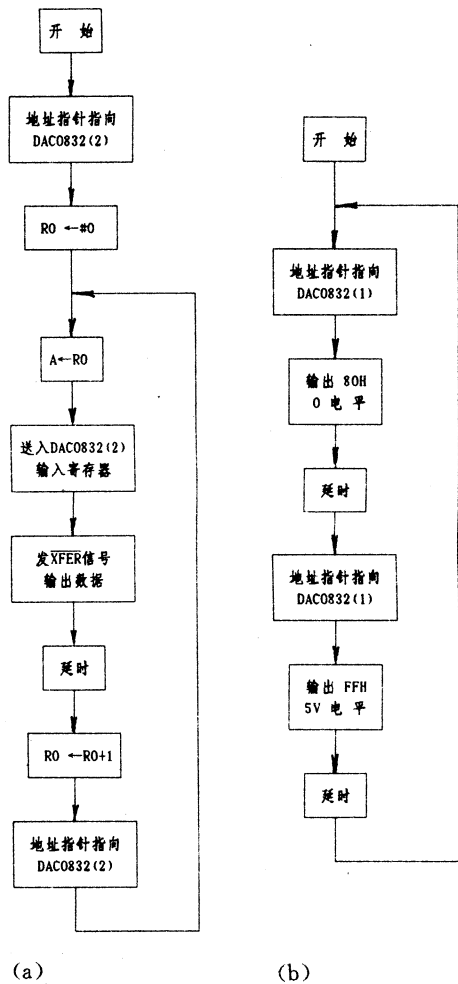


图6 D/A 转换实验程序框图

地址	机器码	行号	标号	汇编语言	注释
8000		1		ORG 8000H	
8000	4120	2		AJMP DASAW	
		3		:	
8220		4		ORG 8220H	
8220	905000	5	DASAW:	MOV DPTR, #5000H	地址指针指向 DAC0832(2)
8223	7800	6		MOV RO, #00H	输出数据从 00H 开始
8225	E8	7	WW:	MOV A, RO	
8226	F0	8		MOVX @DPTR, A	送数据到 DAC0832(2)
8227	903000	9		MOV DPTR, #3000H	完成 D/A 转换输出
822A	F0	10		MOVX @DPTR, A	
822B	128235	11		LCALL DELAY	延时
822E	08	12		INC RO	D/A 转换数据从 00H 开始
822F	00	13		NOP	每次加 1
8230	905000	14		MOV DPTR, #5000H	
8233	4125	15		AJMP WW	
		16		:	延时子程序
8235	7B02	17	DELAY:	MOV R3, #02H	
8237	7CFF	18	T3:	MOV R4, #0FFH	
8239	7DFF	19	T4:	MOV R5, #0FFH	
823B	DDFE	20	T5:	DJNZ R5, T5	
823D	DCFA	21		DJNZ R4, T4	
823F	DBF6	22		DJNZ R3, T3	
8241	22	23		RET	

若小电机转动时带动一小叶片,小叶片转动到某一位置时将光电耦合器的光阻断,转速越快,每秒钟阻断光的次数越多,光电耦合器输出的脉冲频率越高,将该脉冲接入8031的T0或T1口,通过编程就可测出电机转速,并可在数码管上显示出来。

### 六、实验操作和编程:

在上述实验电路的基础上,通过编程将一组数码送入D/A转换器,执行变换指令后可在运放输出端得到相应的一组电压值,如果是按某种规律送入一组数码,则输出电压也将按某种规律变化,如正弦波、三角波、方波等等;如果两路输出均按正弦波变化,且相互间相位差为90°,这样两组电压输入示波器的x,y轴,在示波器上可以观察到圆形或椭圆的轨迹。

### 程序2

地址	机器码	行号	标号	汇编语言	注释
8000		1		ORG 8000H	
8000	4180	2		AJMP DAMOTOR	
		3			
8280		4		ORG 8280H	
8280	901000	5	DAMOTOR:	MOV DPTR,#1000H	地址指针指向DAC0832(1)
8283	7800	6		MOV A,#80H	先送数据80H
8285	F0	7		MOVX @DPTR,A	送数据到DAC0832(2)
8286	903000	8		MOV DPTR,#3000H	完成D/A转换输出0电平
8288	F0	9		MOVX @DPTR,A	
828A	519A	10		ACALL DELAY1	0电平持续时间
828C	78FF	12		MOV A,#0FFH	再送数据0FFH
828E	901000	11		MOV DPTR,#1000H	地址指针指向DAC0832(1)
8291	F0	13		MOVX @DPTR,A	送数据到DAC0832(1)
8292	903000	14		MOV DPTR,#3000H	完成D/A转换输出电平5V
8295	F0	15		MOVX @DPTR,A	
8296	516A	16		ACALL DELAY1	5V电平持续时间接触动
8298	4150	17		AJMP DAMOTOR	
		18		延时子程序	
829A	7B02	19	DELAY1:	MOV R2,#0FFH	
829C	7CFF	20	DEL:	DJNZ R2,DEL	
829E	22	21		RET	
829F	DDFE	22	DELAY2:	MOVZ R2,#50H	
82A1	DCFA	23	DEL2:	DJNZ R2,DEL2	
82A3	DBF6	24		RET	

如果D/A转换器输出的模拟电压,再送给功率驱动部件执行控制任务,如直流小电机的正、反转及快、慢转等。

如上述可做的实验项目很多,下面仅以输出三角波(锯齿波)和控制直流小电机为例,说明编程以及实验操作方法。

### 1. 输出三角波实验编程和操作:

根据要求程序框图如图6a所示。

由上述程序框图,编制的三角波实验程序如程序1:

运行上述程序,用示波器在运算放大器LM324(U8:B)的第7脚可以观察到三角波(锯齿波)为了得到清晰的波形,应该适当调节示波器频率开关和同步旋钮。在实验时,改变延时子程序中R<sub>3</sub>预置的数值,可以得到不同斜率的锯齿波。

### 2. 控制小电机实验编程:

根据控制小电机正、反转和快、慢速的要求,得到程序框图如图6、b所示。

由程序框图编制的实验程序如程序2:

运行上述程序,就可在DAC0832(1)(U5)输出电路的输出端获得一串脉冲,经放大便可驱动小电机移动。

改变程序中语句:828A、828C、8296,可使输出脉冲的直流电平及持续时间变化,从而达到使电机正转、反转,快转和慢转的目的,在不同转动情况下,828A、828C及8296各单元的内容如下表:

存储单元	正向快转	反向快转	正向慢转	反向慢转
828A	519A	519A	519F	519F
828C	74FF	7400	74FF	7400
8296	519A	519A	519F	519F

### 结束语:

到本期为止,有关BJS-51单片机实验系统系列讲座已经结束。我们将在“电子与电脑”杂志的帮助下,将讲座的内容汇集成册出版。研制这样一个有关单片机的教学实验系统是我们的初次尝试,不足和错误一定不少,敬请广大读者赐教。有意见和愿购买该教学实验系统的同志请与北京市单片机应用技术协会技术服务部盛志纯同志联系。地址:北京西城三里河西口18号楼,邮编100045。





# 中华学习机巧测电容

四川重庆市20信箱62分箱 邓本富

我们知道,在CEC—I中华学习机上有一个九针游戏杆插座。这个插座可提供4个模拟量和3个开关量的输入,用BASIC语言中的PDL(n)函数可以读取n号(n=0~3)摇杆的编码值(0~255)。实际上,该编码值就是学习机内部定时器组件的延时循环次数。机内监控程序的循环时间为11个时钟周期,共计10.8μS。如果在模拟量输入端接入被测电阻,用一个简单的BASIC程序调用监控程序,测出延时循环时间t(t等于10.8μS乘以延时循环次数),通过RC电路充电时间关系式 $V_c = V_{cc}(1 - e^{-t/RC})$ ,即可算出被测电阻的阻值。关于这一点,清华大学出版社出版的《苹果机和中华学习机应用实验与实用制作》一书中已有详述,本文不再赘述。

在《实用制作》一文中,对测量电容须增加接口电路,由于接口电路要用STB选通信号控制工作,而STB信号输出只有APPLE—II机上的16脚游戏I/O接口才有,在CEC—I机上无此选通信号输出,所以无法实现。而笔者通过分析认为,在CEC—I上,不用增加接口电路,完全可以测量电容。下面谈谈测量方法。

电容的测量原理和测量电阻类似,仍然利用RC电路充电的时间关系。CEC—I中用NE558(U19)作为定时器组件,其内部由4个独立的555时基电路所组成,图1示出其四分之一,相当于一路模拟量输入接口(对应PDL0)。虚线为测量电容时所加。电阻R\*和电容C(C=C<sub>1</sub>+C\*)构成时基电路RC充电回路。当触发输入端(3脚)接到一个负向脉冲时,输出端(1脚)输出为高电平,负脉冲消失后,电源电压V<sub>cc</sub>通过R\*对C充电,2脚电压逐渐上升,当2脚电压超过2/3V<sub>cc</sub>时,时基电路迅速翻转,1脚输出变为低电平。只要测出1脚高电平的持续时间,通过算式 $t \approx 1.1RC$ 即可得到C的容量值,C\*的具体计算公式为:

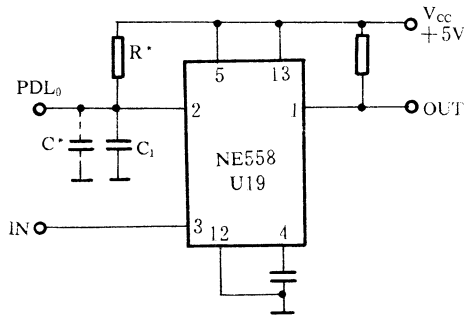


图1

$$C^* = 10.8M/1.1R^* - 0.022(\mu F)$$

注:式中R\*的单位为欧姆

测量电容的BASIC程序清单如下:

```

10 DIM A(20)
20 K=0
30 FOR I=1 TO 20:A(I)=PDL(0):K=K+A(I):NEXT I
40 M=K/20:IF M>=255 THEN PRINT "CONDENSER TOO LARGE":PRINT "CHANGE THE RESISTANCE":END
50 C=10.8*M/1.1*R-0.022
60 PRINT "C=";C;"μF"
70 FOR I=1 TO 3000:NEXT I:GOTO 20

```

测量时,连接方法如图2。把R\*接入+5V和PDL(0)端,C\*接入GND和PDL(0)端,运行BASIC程序即可。若换接不同的R\*,则C\*的可测范围亦随之变化。R\*越大,可测C\*的最大值越小,但分辨率提高。

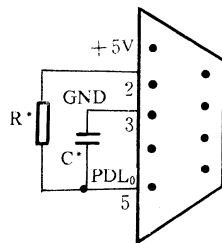


图2

下面是R\*与C\*的大致关系

R*	C* min	C* max	分辨率
10k	0.0006μF	0.2μF	0.001μF
1k	0.03μF	2μF	0.01μF

这种电容测量方法,经笔者在中华学习机上实践,效果良好,有兴趣的读者不妨一试。

# 家庭如何配置中华学习机

上海交通大学闵行二部(200240) 聂铁轮

1. 首先应当清楚,购买学习机用来完成何种任务。根据不同的要求,购买适当的学习机及其软件。

CEC-M型是一种初级的中华学习机,不能显示彩色图案,不能外接驱动器,其内固化 BASIC 解释程序,局限于对儿童进行普及型教育,其工作范围有限。

CEC-I型,这种计算机对于家庭比较适用,它不仅与 Apple I 型计算机完全兼容,而且具有汉字输入显示功能,可外接软盘驱动器,CEC-I型的汉字功能已经固化在计算机的主机板上,可随时输出汉字,显示器的接口可接 PAL 制式彩色或黑白电视机,另外还设有游戏控制杆接口。

2. 对 CEC-I 型功能的扩充,及外围设备的配置:

在 CEC-I 型机主机板上有一个与 Apple I 兼容的输入、输出插座,用来扩充 CEC-I 型机的功能。

1. Z-80卡:由于 CEC-I 型中华学习机使用的操作系统是 DOS 系统,所以在 CP/M 操作系统下,数以千计的软件无法使用,我们可以在扩充槽口上插入 Z

-80卡,在 Z-80卡中有固化的 CP/M 操作系统,就可以使用 CP/M 操作系统下的大量软件,Z-80卡售价160元左右。另外还需一台软盘驱动器。

2. 使用 dBASE-II 系统,如果我们想在 CEC-I 上建立 dBASE-I 管理系统,至少需要56K的内存,CEC-I 型机内存为69KRAM,可以满足需要。由于 dBASE-I 管理系统是用8080汇编语言编写的。需在扩充槽口插入 Z-80卡,再购买一台软盘驱动器,将含有 dBASE-I 的软盘调入内存,就可使用了。有时我们需要把汉字库调入内存,使用汉字,如果内存中存在汉字库和 dBASE-I 管理程序,可能有些内存的字节溢出,发生这种情况,就要在扩充槽上插入扩充卡,如16K的 RAM 等。

3. 如果你要在屏幕上进行汉字编辑并打印编辑内容,可以购买 CPC 卡,此卡有汉字编辑和打印机驱动功能,使用很方便。如果仅仅局限于打印,可购买打印卡,接在扩充槽上。

## 92' 全国软件出版工作研讨会 在湖北宜昌召开

1991年10月15日至20日,由中国软件行业协会软件出版分会与湖北省新闻出版局联合举办的“92' 全国软件出版工作研讨会”在宜昌市召开。国家新闻出版署与机械电子工业部对会议的召开给予了大力支持。国家新闻出版署刘泉副署长亲笔至函祝贺,并对软件出版工作表示极大的关注;技术发展司、图书司、版权司的有关领导出席了会议,宜昌市的党政领导到会祝贺。《计算机世界》报打来电报预祝会议圆满成功。来自北京、上海、天津、华东、东北、西北、西南、华中等地区的出版社、出版管理部门、软件开发单位以及有关报刊等36个单位的48位代表参加了会议。湖北省新闻出版局副局长田胜立同志与软件出版分会秘书长宋玉升同志主持了会议。

新闻出版署技术发展司司长高永清同志在大会上对于近年来电子出版物的发展作了重要讲话。

这次研讨会共收到论文21篇,大会报告16篇,对于国内外软件出版概况,软件出版工作流程及文档管理,软件的选题与组稿,软件编辑岗位责任制问题,软件出

版物的质量与商品化,软件的版权保护问题以及软件的发行与软件市场等进行了深入的研究与讨论。代表们一致认为,软件出版是我国软件产业中不可忽视的一支力量,工作取得了长足的进步,但是还存在许多问题和困难,例如,软件出版的法律效力不明确;发行渠道不流畅,软件编辑人员的工作还需要有关领导的确认和大力支持。

软件出版在我国已初具规模,开拓了新的出版业务,国家新闻出版署极为关心软件出版事业的发展。在当前国际上电子出版物发展迅速并处于激烈的竞争的情况下,应进一步加强对包括计算机软件、数据库、电子书刊在内的电子出版物的规划、领导与管理,这对于我国电子出版事业的发展,具有重要的意义,特别是要研究解决软件以国际标准书号出版的法律效力及今后电子出版物的版权保护问题,制定一系列软件出版工作规范与管理条例,加强编辑队伍建设,协调软件发行工作,把我国软件出版事业推进到一个新阶段。



## 电脑游戏机

# F BASIC 语言的游戏程序编程技巧

## 第四讲 游戏程序的设计过程(中)

山东苍山机械电子化学工业局(277700) 于春

### (6)模块 H:

成龙若按 PS \$ 中记录的顺序打开箱子则发声,打印积分,并令箱子消失。每打开一个箱子加10分。若箱子全部打开则转救人成功程序。流程图见图5:

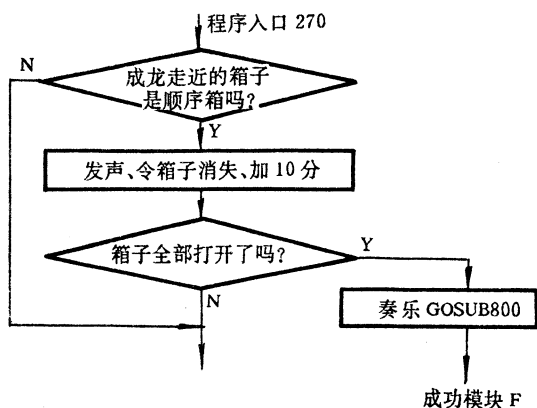


图5 模块 H 的流程图

```
270 REM "OPEN BOX" (打开箱子)
275 IF S1 $ <> MI. (PS $,PS,1) T. 295
280 PL. "O4C0#CD#DEF#FG#G"
285 LOC. PX+SX,PY+SY;P. "┌";
 T=T+10;LOC. 8,22;P. ↑
290 PS=PS+1;IF PS>PL T.PL. "O4C3D
 EFG;O4E3 FGAB";G. 110(成功处理)
295 RE.
```

注:┌表示空格,下同

### (7)模块 I:

妖怪自动运动。轮流测试妖怪在 BG 面的坐标。判断妖怪周围是否为楼板或楼梯,使妖怪在楼板上左、右移动,在楼梯上上、下移动。使用变量:

KD:妖怪的运动方向。

F:妖怪动作的编号。KN=2时,F=0,

1KX,KY:妖怪在 BG 面的坐标。

JX,JY:成龙在妖怪上下左右的判别。

SK \$ ,SR \$ ,SL \$ ,SD \$ ,SU \$ :妖怪

前后左右的背景数值。流程图见图6。

对应程序如下:(程序入口300)

```
300 REM "BOGY MOVE"(妖怪运动)
```

```
305 F=F+1;IF F>KN-1 T.F=0
310 KD = VCT(F);IF KD>0 T. 320
315 KD=E(F)
320 KX=(XPOS(F)-12)/8;JX=SGN(PX-KX)
325 KY=(YPOS(F)-16)/8;JY=SGN(PY-KY+1)
330 SK $ =SCR $ (KX,KY);SR $ =SCR $ (KX+1,KY);
 SL $ =SCR $ (KX-1,KY);SD $ =SCR $ (KX,KY+
 1);SU $ =SCR $ (KX,KY-1)
335 IF (SR $ +SL $ <> "┌")AND(SD $ +SU $ <> "┌")
 T. 345
340 IF (KD=3 AND JX=-1)OR(KD=7 AND JX=1)OR
 (KD=5 AND JY=1)OR(KD=1 AND JY=-1) T. 390
342 KD=(KD+4)+8*(KD>3);G. 390
345 IF JX * JY <> 0 AND RND(2)=1 T. 355
350 IF JX=1 AND SR $ <> "┌" T. KD=3;G. 390
352 IF JX=-1 AND SL $ <> "┌" T. KD=7;G. 390
```

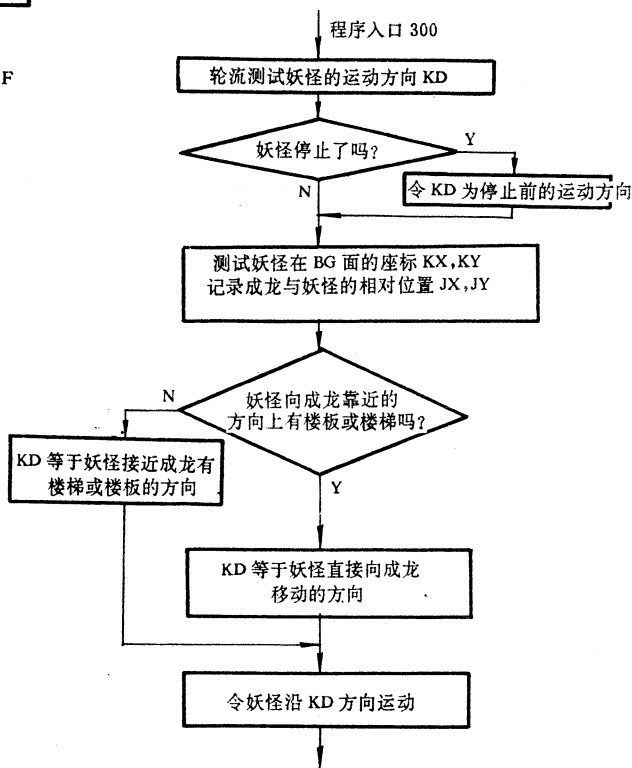


图6 模块 I 的流程图

```

355 IF JY=1 AND SD$ <> "┘" T.KD=5;G. 390
360 IF JY=-1 AND SU$ <> "┘" T.KD=1;G. 390
365 S=RND(4)+1;ON S GOTO 370,375,380,385
370 IF SR$ <> "┘" T.KD=3;G. 390
375 IF SL$ <> "┘" T.KD=7;G. 390
380 IF SD$ <> "┘" T.KD=5;G. 390
385 IF SU$ <> "┘" T.KD=10
390 DE. M. (F)=SP. (11,KD,1,4,0,RND(4));POS. F,KX
 * 8+12,KY * 8+16;M. F
395 E(F)=KD;RE.

```

(8)模块 F

救人成功处理。令所有定义的卡通消失。画室内地板。定义成龙、金凤相向奔跑、拥抱。成龙起始座标(0, 104),金凤起始座标(240,104)。音乐停止后,清屏、画路。成龙、金凤手拉手向屏幕左方运动(回家)。成龙起点座标(220,104),金凤起点座标(230,104)。流程图见图7,对应程序清单如下:

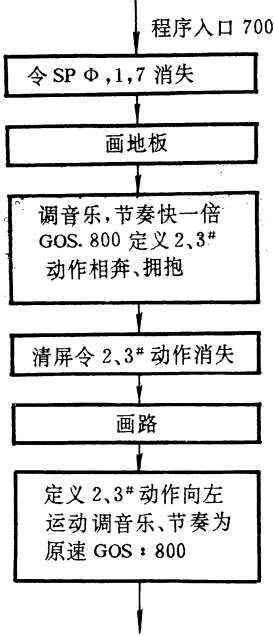


图7 模块 F 的流程图

```

700 REM"VICTORY1"(救人成功)
705 ERA 0,1,7;CLS;SP. 3
710 F. I=0 TO 27;LOC. I,12;P. CH. (208);N.
715 PAL. B 0,10,10,21,49
720 F. I=2 TO 3
725 DE. M. (I)=SP. (I-2,4 * I-5,2,58,0);POS. I, (I-2)
 * 240,104;M. I;N.
730 PL. "M0V12M1V9Y2T1;M0V12T1;M1V0T1";FF=0;
 GOS. 800
750 REM"VICTORY2"
755 ERA 2,3;CLS;PAL. B 0,5,22,39,56
760 F. I=0 TO 27;LOC. I,12;P. CH. (197);N.
765 F. I=2 TO 3

```

```

770 DE. M. (I)=SP. (I-2,7,4,230,0)
775 POS. I, (I-3) * 8+230,104;M. I;N.
780 PL. "M0V12M1V12Y1T4;M0V9T4;M1V2T4";FF=1;
 GOS. 800
785 RE.

```

(9)模块 K

结束处理,若是成龙被妖怪抓住或者时间到则打印游戏结束,询问是否继续游戏;若救人成功,在显示成功画面后,直接询问是否继续。两种情况用一个选项程序,即读1#操纵器的 START 钮和 SELECT 钮。若继续游戏按 START 钮;否则按 SELECT 钮。模块流程图和程序清单见管理程序。

(10)模块 J:

音乐模块。为简化程序,成龙与金凤散步音乐、每

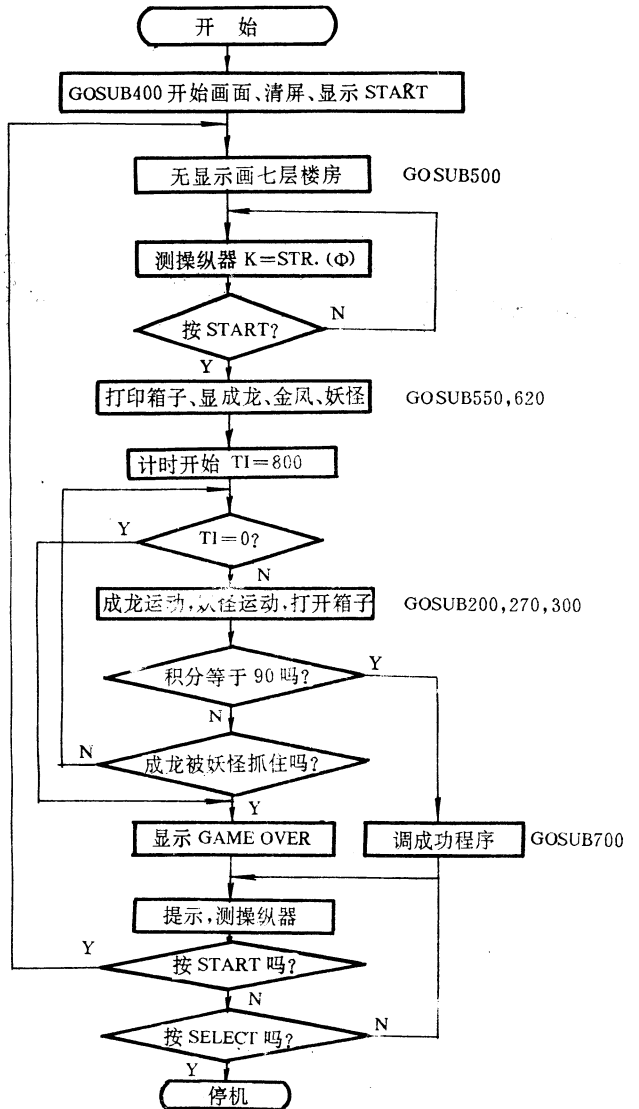


图8 管理程序流程图

场的开始音乐和救人成功用同一个旋律,不同的是在成龙与金凤相向奔跑时要求的激烈奔放的音乐是把音乐节奏加速一倍,其它使用正常节奏。在山鹰叼走金凤时有一段急促紧迫的音乐。

### 主题音乐 (三重奏)

```
800 REM "MUSIC1"
805 PL. "O4G5E3GA4G1;O4C5O3GA;O1C3CC5C3C"
810 PL. "E3DCDE4D1;GDG;C5C3CC5"
815 PL. "E3 G7G3E4G1;AGGO4C;C3CC5C3CC5"
820 PL. "A3O5C6O4G3A;DCO3G;C3CC5C3CC5"
825 PL. "DE1DC7;CEC;C5C3CC5"
830 PL. "G5E3GA4G1;O4CO3GA;G3GG5G3G"
835 PL. "E3DCDE4D1;GDG;G5G3GG5"
840 PL. "E3G7G3E4G1;AGGO4C;G3GG5C3CC5"
845 PL. "A3O5C6D3G;ECD;C3CC5C3C"
850 PL. "DE1DC7;GDC;C5C3CC5"
855 IF FF=0 THEN FF=1;G. 805
860 RE.
```

### 叼走金凤音乐

```
900 REM "MUSIC2"
905 PL. "M0V10M1V8Y3T2;M0V3T2;T2"
910 PL. "O4C3CO3GA;O3C3CO2GA;O2E5G"
915 PL. "Q4CCO3GADG6;O3CCO2GADG6;CGDG"
920 PL. "D3G6A3O4E6;D3G6A3O3E6;DGAO3E"
925 PL. "Q3A3O4E6;O2A3O3E6;O2AO3E"
930 PL. "C3CO3GA;C3CO2GA;O2CG"
935 PL. "O4CCO3GA;O3CCO2GA;CG"
940 RE.
```

### (11)管理程序

管理程序完成整个游戏过程的控制。其流程图见图8。

管理程序如下:

```
10 REM "MAIN PROGRAM" (主程序)
15 GOS. 400 (开始画面)
20 CLE. ;CLS;CG.RND(2);RND(3);CGEN2
25 LOC. 8,8;P. "START!!!"
30 SC. 0,1;PL. "M0V12M1V12Y1T3;M0V9T3;M1V2T3";
 FF=1;GOS. 800
35 SP. O. ;PL. "V15T1;V15T1"
40 IF SCR $ (1,2)<>" " T. 50
45 GOS. 500 (画七层楼)
50 SC. 0,0;V. ;P. " " TIME = " " PRESS "
 START"
55 K=STR. (0);IF K=0 T. 55
60 PL. "O3C1EE;O3E1FG"
65 GOS. 550 (显示九个箱子)
70 GOS. 620 (显示成龙、金凤、妖怪)
75 TI=TI-1;LOC. 7,0;P. TI;" " (计时开始)
80 IF TI=0 T. 115
85 GOS. 200 (用操纵器控制成龙)
90 GOS. 270 (打开箱子的处理)
95 GOS. 300 (妖怪自动运动)
100 IF CR. (7)>=0 T. PL. "O1B2AGFEDC";G. 115 (救
 人失败)
105 G. 75
110 GOS. 700;G. 130 (救人成功)
115 CLS;LOC. 9,6;P. "GAME-OVER!"
120 SP. OF.
130 LOC. 2,20;P. "CONTINUE/start-END/select"
135 S=STR. (0);IF S=0 T. 135
140 IF S=1 T. RUN 20
145 IF S=2 T. CLS;LOC. 10,4;P. "GOOD-BY!";LOC. 0,
 21;E.
150 G. 135
```

现在“成龙救金凤”游戏程序全部编写完毕。把各模块程序合成即得模块结构程序清单。读者可自己合成。

或“电脑与通信”栏目稿件)

## 致读者与作者

应广大读者要求,我杂志明年新增“IC 与应用”、“电脑与通信”两栏目。“读者联谊”栏目将调整为几个小栏目,有‘读者信箱’、‘购机指南’、‘求答’、‘转让’等,以便更好地为读者服务。欢迎大家踊跃投稿。内容范围大致为:有关国内外 IC 的发展及最新 IC 的应用资料,如开关电源(图像处理、模糊逻辑、通讯)IC 等。有关点对点及网络通信,办公室自动化方面。字数一般在5000字以下。稿件一经利用,即付稿酬。

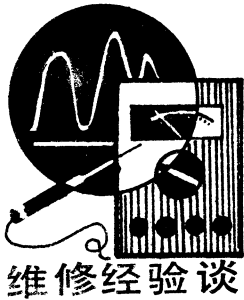
来稿请寄:北京173信箱杂志编辑部收  
邮编:100036 (请在信封上注明“IC 与应用”

## 敬告读者

本刊92年第3期登出的 PAL 卡,因卡上的调试点需要用户自行调整后才能达到预期效果。

为此购买者请与编辑部直接咨询。本刊93年撤消七号信柜邮购点。

《电子与电脑》编辑部



# LQ-1600K 打印机接口专用集成电路 M54610P 的简易修复方法

陕西省计量测试研究所(710048) 赵继文

## 1. 专用集成电路 M54610P 简介

LQ-1600K 打印机并行接口电路由一片专用集成电路 M54610P 构成。其内部逻辑电路如图1所示,各引脚功能列于下表中。

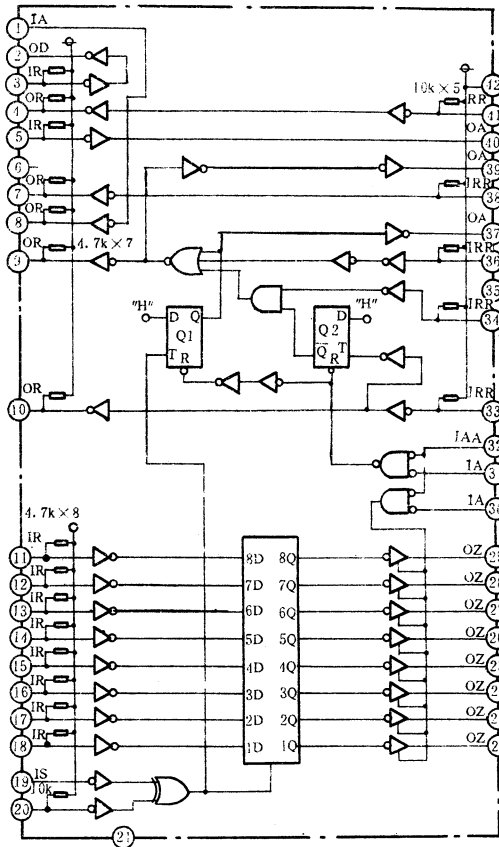


图1

## 2. M54610P 的一种简易修复方法

针式打印机自检正常,而联机不打印或打印错码,都是由于接口电路损坏或接口电缆连接器产生故障而造成。本文介绍 LQ-1600K 打印机并行接口电路损坏的一种简易修复方法。其故障现象是:开机自检正常,但不能联机打印,缺纸灯常亮,实际上并不缺纸。经用万用表测试 CN1(与计算机相连接的接插件的第12针,它的输出表示 PE(缺纸)信号,测试结果为常高电

M54610P 引脚功能表

引脚号	名称	信号方向	功能
1	BI1	入	PE(缺纸信号)
2	BO1	出	送往复位电路的INIT信号
3	$\overline{BI2}$	入	来自主计算机的INIT信号
4	$\overline{BO5}$	出	来自主计算机的ERROR信号
5	BI3	入	READY(准备就绪)信号
6	GND	—	地
7	BO4	出	送往主计算机的 PE (缺纸)信号
8	$\overline{BO1}$	出	送往操作面板的PELED (缺纸灯)信号
9	BUSY	送往主计算机	BUSY(忙)信号
10	$\overline{ACK}$	出	$\overline{ACKNLG}$ (应答)信号
11	DIN8	由主计算机送入	8位并行数据
12			
13			
14			
15			
16			
17			
18			
19	STB	由主计算机送入	STROBE(选通)脉冲
20	PSW	由 CPU 送入	BUSY 信号输出时间 低电平;STROBE 的上升沿 高电平;STROBE 的下降沿
21	GND	—	地
22	DOU1 DOU2 DOU3 DOU4 DOU5 DOU6 DOU7 DOU8	由 CPU 送入	8位并行数据
23			
24			
25			
26			
27			
28			
29			
30	RD	由 CPU 送入	$\overline{READ}$ (读)脉冲
31	$\overline{WR}$	由 CPU 送入	$\overline{WRITE}$ (写)脉冲

32	$\overline{CS}$	由 G/A 送入	Chip select(片选)信号
33	ACKI	由 CPU 送入	Acknowledge(应答)信号
34	BSSL	由 CPU 送入	BUSY 信号的选择 低电平:当 ACKI 变高时, BUSY 信号变为低; 高电平:当 CS 和 WR 都变低 时,BUSY 信号变为高。
35	GND	—	地
36	EBSY	由 CPU 送入	BUSY(忙)信号
37	BSYF	送往 CPU	BUSY(忙)脉冲
38	BI4	由 CPU 送入	$\overline{PE}$ (缺纸)信号
39	RDY	—	无用
40	BO3	出	送往操作面板 RYLED (准备就绪指示灯)信号
41	$\overline{BI5}$	入	来自 CPU 的 ERROR (出错)信号
42	Vcc	入	+5V 电源

平。该信号来自 M54610P(代号5A)的引脚7,从图1可看出引脚7的内部是一只独立的反相门电路,7脚是反相门电路的输出端,38脚是输入端,它的输入信号来自用作 CPU 的  $\mu$ PD7810HG(代号7B)。再测38脚,结果也为高电平,说明 CPU 送来的信号是正常的,确认反相门电路损坏。与 PE(缺纸)信号及 PELED(缺纸指示灯)信号有关的电路用图2表示。由于输入、输出端分别为38和7引脚的反相门 Q16损坏,虽然 CPU 送出的信号为正常的高电平,但经 CN1的第12针反映到主计算机上的 PE 信号为有效的高电平,所以不能联机打印;

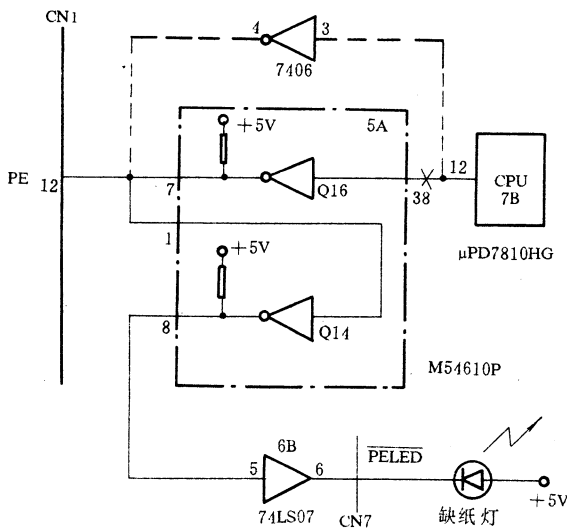


图 2

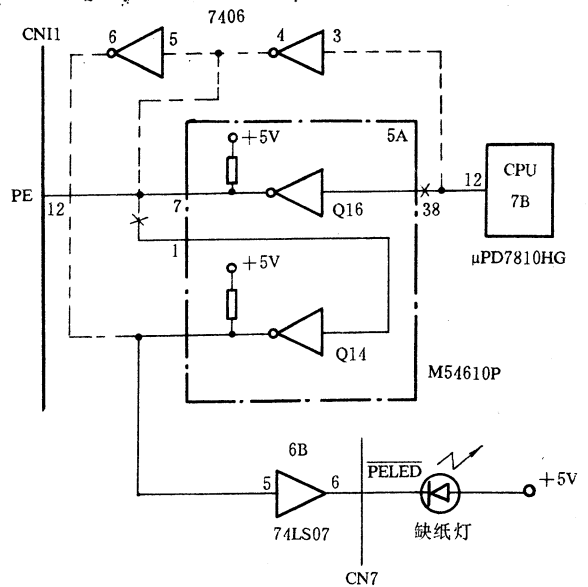


图 3

从图2可以看到5A的引脚1和引脚7相连接,由于引脚7是高电平经反相门 Q14后,从5A引脚8上输出低电平,经74LS07(代号6B)后仍输出有效的低电平缺纸指示灯信号 PELED,所以缺纸灯常亮。按理应用 M54610P 更换5A,但考虑到 M54610P 专用集成电路价格昂贵,且市场上也不易购到,故采用外加一片7406来替换5A中所损坏的反相门电路 Q16。具体方法是:在主板电路板上将5A的引脚38与7B的引脚12的印制连线割开,将7406的引脚3接到7B的引脚12上,7406的引脚4接到5A的引脚7上,用原 Q16输出端的上拉电阻作为7406的上拉电阻。在图2中用“×”号表示要割开的点,用虚线表示要连接的线。并将7406的引脚14和7分别接到+5V和地上,其余引脚向上翻起,以避免造成短路。用这种简易可行的方法修复 M54610P 后,打印机故障立即排除。

其所以会损坏接口集成电路,绝大多数是由于用户带电插拔接口连接器,或带电拨动机械式多路转换器造成。因为这种带电操作在连接器与接口集成电路相连接的各个端口都会产生突发性冲击电流,若冲击电流过大,集成电路质量又欠佳时,就会损坏电路。本文前述的现象是反相门 Q16损坏。实际工作中也可能反相门 Q16和 Q14都同时损坏,损坏后在5A的引脚8上也始终为高电平,此时经同相门6B后仍为高电平。这时缺纸灯虽不亮,但由于5A的引脚7是常高电平,故打印机仍将产生不联机打印的故障。若遇到这种情况时,可用7406芯片中另外一只反相门来替换 Q14,具体方法见图3,除采取前述分割、连接步骤外,还将7406的引脚4和5串联起来,引脚6连接到5A的引脚8上,再将5A的引脚1和8之间的印制线割开即可。

# GW286计算机硬盘控制卡

## 故障处理一例

湖南黔阳县委办公室(418100) 杨远成

故障现象:开机后,机器不能由硬盘启动,用DOS软盘可启动系统,但不能进入硬盘,并显示:“无效的驱动器标识符”。

故障分析:首先,按Ctrl+Alt+Esc键,检查CMOS RAM系统配置参数均为正常。然后在A驱动器运行GW286诊断程序,对硬盘进行诊断时屏幕显示“1714 DRIVE NOT READY FIXED DISK DRIVE C”。即硬盘驱动器未准备好,无“准备好”信号。通过对以上故障现象分析,可能有以下两种原因,一是硬盘本身的问题,主要是硬盘0磁道信息丢失或硬盘0磁道损坏;二是硬盘控制卡或驱动器等方面的问题。

处理方法:首先检查第一种情况,使用FDISK进行硬盘DOS分区,运行失败,并显示:“Error reading fixed disk”。再进行低层格式化也失败。由此可排除产生第一种情况的可能性。接着对第二种情况进行检查,关机后打开主机机壳,取出硬盘控制卡,更换该控制卡,开机,硬盘能够正常启动进入系统。又经检查,原硬盘控制卡也没发现什么问题。再次关机,重新装回原硬盘控制卡,并将各连接部件安装牢固,使其接触可靠,再重新开机,机器由硬盘引导启动成功,且原硬盘上全部信息完好无损,故障得以排除。由此分析可见,该故障为硬盘控制卡及其连接电缆接口部位松动或接触不良所致。

---

## TP801单板机的常见故障

江苏邳县港务局第一港埠公司中控室  
(221300) 花成

---

TP801单板机是目前国内应用比较普遍的一种以Z-80为CPU的微机,由于其性能好,指令丰富、功能强,可扩展性好,集成芯片较通用等特点,而被广泛应用于各工科院校计算机原理实验机,工业控制采样机,

成为实现智能化的得力助手。但由于单板机在使用时比较频繁,因此总会出现一些常见故障,使用者应能分析排除这些常见故障,使之能正常运行工作,下面对TP801单板机中一些常见故障作简要分析:

### 一、电源故障

TP801单板机工作在+5V电压下,假如把电源接在+15V或把电源极性接反,则在工作时,数码显示管出现忽有忽无,时强时弱的不稳定信号,更严重的是,这种人为故障特别容易损坏CPU芯片,造成整机无法工作。

### 二、2716EPROM损坏

2716EPROM芯片内固化了许多监控程序,当出现故障时,会出现监控失调,比如:正在输入程序时,会突然莫名其妙地跳到其他单元或出现其他错误信息,排除该故障的方法之一便是更换一块新的2716EPROM芯片,这是最保险的一种办法。

### 三、2114RAM损坏

由于TP801单板机主要是对内存操作,因此,2114RAM芯片损坏是一种最常见的故障,判断该种芯片是否损坏的方法有:

1. 在用户读写存储区内用NEXT键或LAST键进行存写或读取,当发现某一单元无法读写或出现混乱时,则一般情况下,是RAM芯片损坏。例如低四位或高四位数据始终不能改变,而其它RAM区却正常读写,单板机在未损坏的区域内仍能执行各种功能。这种故障排除很简单,只要根据译码电路和出错单元号更换一块新的RAM芯片即可。

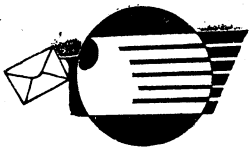
2. 该种情况下单板机根本无法进行,数码显示管内容出错或根本没有显示。碰到这种情况,人们往往误认为是CPU或显示电路部分出了故障,但这时最容易出故障的却是地址2C00—2FFFH的RAM区,因为这一区域(1K)中有监控程序使用的暂存区,用户堆栈工作区,用户程序寄存器存放区等,如果这1KRAM出了故障,势必使整个单板机无法运行,所以当出现此种故障时,应该仔细考虑这个问题。

### 四、MON键或STEP键失效

MONITOR监控键及SINGLE STEP单步执行键是TP801单板机上最重要的两个功能键。如果这两个键都失效的话,则很有可能是Z80-CTC出了故障。因为MON键和STEP键要实现其正常功能,必须通过Z80-CTC向CPU请求屏蔽中断不可。如果STEP键正常,而MON键失效,则损坏的可能性最大的是MC14538单稳态触发器,因为MON键是通过此单稳态触发器向Z80-CTC发送计数脉冲的。

上面所述的只是TP801单板机的几种常见故障,单板机一般用于现场控制,容易受干扰,只有正确地维护好单板机,才能减少单板机的故障。





# 普及型 PC 个人用户软件交流联谊活动 问题解解答(十二)

读者联谊

北京中国农科院计算中心(100083) 王路敬

## 49. 硬盘的主引导扇区数据破坏后不对硬盘进行低级格式化能否恢复?

不管 IBM-PC 还是兼容机一些意外事故造成硬盘数据丢失, 自举失败, 从软盘启动也无法进入硬盘, 如果立即对硬盘进行格式化处理, 会造成大量数据损失, 在上述故障中其中很重要的原因之一有可能硬盘的主引导扇区数据被破坏。这时不要动不动就对硬盘进行低级格式化处理, 可以将一个相同微机系统中硬盘上的主引导扇区读出来写在软盘某一扇区, 然后再将写入软盘这一扇区的主引导扇区的信息写回被破坏的硬盘主引导扇区。可用下面汇编小程序实现之。

程序一 正常硬盘引导扇区的内容写入软盘

```
A)DEBUG
-A 100
3356:0100 MOV AX,0201
3356:0103 MOV BX,0200
3356:0106 MOV CX,0001
3356:0109 MOV DX,0080
3356:010C INT 13 ;读硬盘主引导扇区
3356:010E MOV AX,0301
3356:0111 MOV BX,0200
3356:0114 MOV CX,0008
3356:0117 MOV DX,0000
3356:011A INT 13 ;写软盘8扇区
3356:011C INT 20
3356:011E
-G
-Q
```

## 程序二 将写入软盘上的硬盘主引导程序写入硬盘第1扇区

```
A)DEBUG
-A 100
3356:0100 MOV AX,0201
3356:0103 MOV BX,0200
3356:0106 MOV CX,0008
3356:0109 MOV DX,0000
3356:010C INT 13 ;读硬盘8扇区
3356:010E MOV AX,0301
3356:0111 MOV BX,0200
3356:0114 MOV CX,0001
3356:0117 MOV DX,0080
3356:011A INT 13 ;写硬盘主引导扇区
3356:011C INT 20
3356:011E
-G
-Q
```

## 50. PC-DOS 3.30 硬盘分区法与以前的版本有何不同?

PC-DOS 3.30 的命令较之以前的版本新增加 DOS 区内分区功能, 即建立 DOS 基本区和扩展区, 但有别于供其他操作系统的“非 DOS 区”。DOS 3.30 的硬盘分区命令 FDISK 选择菜单可以选择以下五个方面的功能之一:

1. 建立 DOS 分区
2. 改变活动 DOS 分区
3. 删除 DOS 分区
4. 显示分区信息
5. 选择另一个硬盘驱动器

如果需要在另一硬盘驱动器上使用 FDISK 是选择5。

DOS 3.30 版的 FDISK 建立的 DOS 基本区 C 盘的上限为 32MB, 扩展区最多可建 23 个逻辑盘(标识 D-Z), 其总容量达 726MB。以长城 286BH 机配置的 HH-1050 40MB 硬盘为例, 可分为 10M 的基本区及 30MB 的扩展区。这种分法下的 C 盘能接受 DOS 2.0 或 2.10 版本的访问, 较之 30MB 及 10MB 顺序为佳。这样为某些系统软件象 CC-DOS 2.13 在大容量硬盘上的使用提供了方便。

## 51. 怎样用 DM 软件修复硬盘 0 磁道“假”坏?

一台 IBM PC/XT286 微机不能从硬盘启动。也不接受 FORMAT 进行格式化, 用低级格式化程序例如 LOWFMT.EXE 也能通过, 但当进行 FDISK 之后, 再用 FORMAT C:/S 进行格式化时却进不去, 最后出现“0 磁道坏, 格式化失败”的错误信息。这时候最好先不要怀疑是硬件问题, 用 DM 软件。DM (DISK MANGER) 磁盘管理软件具有硬盘物理格式化、硬盘分区、0 磁道移道、改变系统配置和检测介质状态等多种功能, 是处理硬盘软故障常用的软件。使用时要特别注意屏幕显示信息。

操作步骤如下:

(1) 运行 DM 程序, 键入 DM 表示选择缺省分区; 若键入 DM/M 表示人工定义分区。

A)DM

(2) 在主菜单下选“1”, 进入物理格式化菜单:

```
Initialization Menu
Select an option(R):1
```

(3) 如保留缺省分区, 键入“Y”后按回车:

```
Is the above DEFECT - LIST accurate for this
drive?(y/n):Y
```

(4) 键入硬盘编号

Select an option(R):C

Enter Interleave Value(3):1

(5) 键入“Y”后回车,开始进行物理格式化:

THIS WILL DESTROY ANY EXISTING DATA ON THIS DRIVE!

Continue?(y/n):y

(6) 物理格式化结束后,按“R”退回到主菜单,选择“P”,进行预置分区设置:

(P)artitioning menu,

Select an option (R):P

(7) 核对分区表,如不修改,键入“Y”后按回车:

Does the above PARTITION TABLE regular modification?(y/n)Y

(8) 预置分区菜单键入“P”后,根据显示信息进行确认:

Prepare ALL DOS, Write/Read-only partition?(y/n):Y

THIS WILL DESTROY ANY EXISTING DATA ON THIS DRIVE!

CONTINUE?(Y/N): Y

(9) 如在分区内只放置一个系统,键入 Y 后回车:

Place a system on the partition?(Y/N):Y

(10) 写入卷标号

Enter 1-11 character Volume Label (RETURN for none):×××

(11) 分区设置结束后,先插入系统盘传送系统,再逐级退回 DM,如能从硬盘启动,则故障排除。

在此,也顺便说一下,286 档次微机例如长城 286BH、EX 等大都配置 40MB 硬盘,DM 软件可以将该容量的硬盘为 C 盘 32MB, D 盘 8MD,如果出现在 D 盘上安装某个系统,例如 CC-DOS 2.13G 后,逻辑盘 D:“丢失”,显示“无效的驱动器”。此时要求在 C 盘根目录下建立两项条件:

(1) 在 CONFIG. SYS 中添加 DEVICE = DM-DRVR. BIN。

(2) 拷贝 DMDRVR. BIN

两者缺一不可,缺前项屏幕无提示,而后者则显示该文件不存在或者无效。

52. 怎样解决从硬盘启动后,屏幕显示:Missing Operating System 错误信息后死机?

硬盘测试成功后便进入 ROM BIOS 的 INT 19 自举程序,把硬盘上 0 头 0 柱 1 扇区的主引导记录和分区信息表读入内存,若检查读入的分区表无引导标志则进入 ROM BASIC,若分区信息表中活动分区多于 1 个,则显示“无效分区”从而进入死循环。若活动分区唯一,则把活动分区的第一扇区内容读入 0000:7C00H 处。若读入错则显示“系统装入错误”进入死循环。若读入正确,则检查读入内容的最后两个字节是否为 AA55,若为 AA55,则把控制权交给活动分区的操作系统,否则显示“Missing Operating System”错误信息而死机。

此时,可将正常 DOS 系统盘上的引导扇区写入硬盘 PC-DOS 分区引导扇区中。操作如下:

A)DEBUG ;进入 DEBUG

-L 100 1 0 1 ;DOS 系统盘插入 B 驱动器,读引导扇区

-W 100 2 0 1 ;写硬盘 PC-DOS 引导扇区

-Q ;退出 DEBUG

53. 高密度软盘向普通 360KB 软盘复制文件较简单的方法是什么?

CPU 为 286、386 的中高档微机系统均配置了两个软盘驱动器,一个高密度,一个 360KB。例如 GW286BH,其中 A 驱动器为高密度 1.2MB 软盘驱动器。高密软盘文件传送到 360K 软盘上比较简单的方法是间接拷贝法。这种方法是在硬盘上建立一个子目录,将高密度软盘上的文件拷贝在某一子目录下,然后进入该子目录利用 RESTORE 命令将文件备份到 360K 软盘上。但这种方法的缺点是由于在硬盘上频繁地进行拷贝,删除等操作,造成硬盘上的碎片大量增加,使磁头在硬盘上的读写时间延长,从而加速磁头的磨损。

54. 怎样在同一个硬盘上安装两个不同的汉字系统?

到目前为止已推出了各种不同版本的汉字操作系统,CC-DOS 2.0/2.10,CC-DOS 2.13A-H,CC-DOS 4.0,GWBIOS 3.00 等等,各具特色,在各种机型上广泛使用着。为了发挥出某一汉字系统特殊作用,希望在同一个硬盘上能够存放两个不同的汉字操作系统,实现这一想法的基本思路是将 C 盘分为两个分区,分别安装一种汉字操作系统,使用时采取改变活动分区和分区的系统类型的办法来达到两个不同操作系统共存的目的。

例如在长城 0520DH 机 20MB 硬盘安装 CC-DOS 4.0 和 GWBIOS 3.00,操作步骤如下:

1. 启动 PC-DOS 3.20,运行该系统下的硬盘分区命令 FDISK 删除 DOS 分区,并重新建立 DOS 分区。将 101-611 柱面的硬盘空间建立为 GWBIOS 3.00 使用的 DOS 分区。这时可在主菜单下选择 1,当询问是否将硬盘全部用于 DOS 时回答 N,询问柱面数的开始柱面号时回答 510T 101,然后按“ESC”退出 FDISK。

2. 用 PC-DOS 3.20 版的 FORMAT/S 命令格式化硬盘,按照 GWBIOS 3.00 的安装方法,在硬盘上安装该操作系统。具体操作可分为如下几步:

(1) C)COPY A: \*.\* C: 拷贝 PC-DOS 3.20 到硬盘

(2) C)COPY A: \*.\* C: 拷贝 GWBIOS 3.00 到硬盘

(3) C)RESTORE A: C: 将 CLIB24 字库文件恢复到硬盘

3. 用 PC-DOS 2.0/2.10 系统盘重新启动,运行该版本的 FDISK。由于该版本不能访问 PC-DOS 3.20,因此此时 FDISK 将 PC-DOS 3.20 版本 FDISK 建立的系统分区视为 NON-DOS 分区,这样就可以很

方便地将剩余硬盘空间全部或者部分建立 DOS 2.0/2.10 的 DOS 分区。操作方法仍是在主菜单下选择建立 DOS 分区功能并回答柱面数和开始柱面号,若将所有剩余空间用于该 DOS 分区,两次回答均按回车键。然后按“ESC”退出 FDISK。

注意做这一步时,对有的微机可能会出现 DOS 2.0/2.10 版的 FDISK 并不视第一分区为 NON-DOS,这时可使用 PC-DOS 3.20 系统提供的 PARTED.EXE 将第一个分区即 GWBIOS 的分区,改变为 NON-DOS,具体做法是运行 PARTED,在 Menu 中选择 TYPE,再选(1),然后选 Compatible DOS,最后退出 PARTED 程序即可。

4. 用 PC-DOS 2.0/2.10 的 FORMAT/S 命令格式化硬盘该 DOS 分区,然后分别将 CC-DOS 4.0 的 5 张软盘拷贝或恢复到硬盘上。至此,在硬盘上存放 GWBIOS 3.00 和 CC-DOS 4.0。如果这时用硬盘启动,将立即进入 CC-DOS 4.0 汉字系统。

5. 当要改变操作系统,从硬盘启动进入 GWBIOS 3.00 汉字系统,不能用各版本的 FDISK 改变活动分区的办法,因为它不能改变分区系统类型。上面提到的 PC-DOS 3.20 提供的 PARTED.EXE,它具有改变系统类型等很多功能。此时将该程序分别拷贝到硬盘的两个分区中,在 GWBIOS 3.00 和 CC-DOS 4.0 两种汉字系统中均可用它来改变系统类型和活动分区。这样可在任一汉字系统下运行 PARTED.EXE 程序,根据菜单提示选择 TYPE,再选(1)或(2)然后选 Bootable DOS,即将选择的分区类型改为 DOS 系统了,同时将该分区置成活动分区,把另一分区的系统类型改为 NON-DOS,再选择 Quit,退出 PARTED,重新启动系统即可进入所希望的汉字系统了。

在这里顺便说一下,新一代长城系统随机 PC-DOS 3.20 系统盘(1)中有文件:

PFC. BAT  
PARTED. EXE  
DISHZ. COM  
HARDRIVE. SYS

这几个文件可以将大于 33 兆的硬盘作为“D”盘进行格式化并提供用户使用。使用方法如下:

- (1) 将 PC-DOS 3.20 插入 A:,键入命令“PFC”
- (2) 在命令 Menu 中选择 CREATE 及 Compatible DOS
- (3) 在命令 Menu 中选择 FORMAT 及(2)
- (4) 在命令中选择 QUIT 退出 PARTED. EXE
- (5) 在 C 盘(启动盘)CONFIG. SYS 文件中增加语句:  
DEVICE=HARDRIVE. SYS
- (6) 重新启动系统。

### 55. 在高密软盘驱动器中如何格式化 360KB 的软盘?

在高密软盘驱动器中格式化 360KB 软盘只要在 FORMAT 命令中带参数“/4”即可完成,不一定非放

在 360KB 软盘驱动器中进行。命令格式与操作的显示信息如下:

```
C>FORMAT A:/4
Insert new diskette for drive A:
and strike ENTER when ready
Format Complete
362496 bytes total disk space
362496 bytes available on disk
Format another? (Y/N)
```

上述信息清楚地说明在高密软盘驱动器可以格式化 360K 软盘。

### 56. 40MB 硬盘为什么格式化后变为 32MB 了?

长城 286BH、286EX、AST286 等许多 286、386 的参数,执行系统配置 40MB 硬盘,但格式化后只有 32MB,少了 8MB。这是由于 PC-DOS 系统基本 DOS 分区只能达到 32MB。换言之,在对硬盘进行格式化时,只建立了基本 DOS 分区 (Primary DOS Partition),大于 32MB 字节硬盘就要建立扩展 DOS 分区 (Extended DOS Partition) 否则硬盘 DOS 基本分区以外的硬盘空间将造成浪费。

建立扩展 DOS 分区可用硬盘分区命令 FDISK,首先建立基本 DOS 分区,可以指定开始柱面号和柱面数,然后可建立扩展 DOS 分区,也分别给出。但有一点应注意,各分区的柱面数之和一定不能大于硬盘柱面总数。分区以后,返回 FDISK 对 C 盘和 D 盘分解进行 DOS 格式化。

利用磁盘管理软件更容易实现上述操作。

## 培 训 消 息

电子工业出版社电子与电脑杂志将于 93 年 2 月 20 日在北京举办为期一学期的微机硬件维修技术代培班;于 93 年 3 月 31 日举办为期四周的微机硬件二级维修技术学习班。授课内容包括:8088、80286 主机板和 80386、80486 简介;3.5"、5.25"软驱(高低密度)及软卡;硬驱硬卡;CGA、EGA 显示器及 VGA 简介;打印机及打印卡;主机电源;主键盘;小型不间断电源(UPS)等工作原理及实用维修技术,并通过演示课练习故障排除,代培学员除学习上述内容外还将在老师指导下参加各部分实际的维修工作。

代培费:1300.00元(名额限4—5人)

学 费:410.00元

资料费约120.00元(实报实销)

上课、实习、食宿(费用自理)均在中国人民大学校园内。

联系人:北京中国人民大学信息中心

胡野红

电话:255.5431转3305 邮编:100872

## C—DBAG 中文数据库应用生成器 F/D \* AG V5.0 零售价:3900元

使用本系统只需按照提示输入用户需求,不需编程,各种管理软件,在功能变动时可任意改动,适用于网络及单用户环境。

功能:生成 DBASE+FoxBASE+源程序;生成任意格式报表;自动生成数据字典及文档;工程图形生成与管理。

组成:下拉/弹出式菜单生成器;查询模块生成器;数据录入与维护模块生成器;数据库文件结构生成器;彩色统计图形模块生成器;任意格式报表生成器;数据字典与文档生成器;

### 灵科汉卡 零售价:1200元

西文软件无需汉化即可直接使用中文,直接写屏,支持1024×768,640×480,640×350图形方式。支持中文排版功能。中文简体与繁体,中文与西文之间壹键转换。支持针式打印机、激光打印机及鼠标。精密宋、楷、黑、仿宋体平滑放大输出,能识别中西文窗口。显示字库不占用内存,且显示速度大大提高。

### 灵科加密器 零售价:180元

可加密任何 EXE 和 COM 文件,也可对任何源程序实施加密,如 dBASE, FoxBASE 等,加密文件个数不限。安装方便,只要外插于打印机接口上,且对打印机工作无丝毫影响,加密手段方便,保密性极高。具有强有力的反跟踪功能,使解密成为梦想。

### 灵科加密卡 零售价:450元

具有灵科加密器的一切功能,并且做到了真正的软硬件加密结合的功能。具有开机口令功能,并且可以方便地修改口令或关断开机口令功能。彻底避免了计算机的非授权使用,减少了计算机病毒的感染机会。本卡有2K字节可供编程人员进行二次开发。不用任何其它硬件设备,只需本公司提供的软件工具或函数即可完成向卡内读写加密数据或程序,写入次数以上五万次。本卡极适合保密性高且成本较高软件加密或大、中型 MIS 系统加密使用。

### 灵科 MIS 管理新工具

#### 程序结构管理及菜单生成系统 零售价:500元

辅助生成结构图,程序文档;自动生成 WINDOWS、C 语言、dBASE 菜单源程序;支持鼠标操作,模拟显示。

#### 程序流程图管理系统 零售价500元

支持顺序判断(IF..ELSE)、循环(WHILE)、选择(SWITCH)逻辑结构图及相应文档;自动生成 C 语言

逻辑调用程序;支持鼠标操作;支持6层嵌套。

经 销:中国电子器材华北公司

地 址:北京市海淀区万寿路西街五号

通信地址:北京144信箱

邮政编码:100036

联系人:石立军 魏 国

电 话:81.1810 81.0920

本公司经营计算机及外设通信设备,欢迎来电函联系。

## 个人电脑联谊活动

为了帮助中小学及个人电脑用户学好用好计算机,解决用户对软件“买难、用难、交流难”的矛盾,星星电脑部在《电子与电脑》编辑部的支持下,开展中华学习机、苹果机、PC 机用户及爱好者联谊活动,欲参加联谊者将登记表填写寄至长春市岭东路115号,星星电脑咨询服务部,随信附手续费、回邮资共1.20元,复函回寄简章、联谊证、软件目录等。

### 主要活动内容:

1. 相互交换软件、资料;
2. 代销成员自开发的实用软件;
3. 代转让旧机器;
4. 免费咨询相关信息和技术问题;
5. 优惠提供硬件、软盘、资料;
6. 联谊成员交纳30元活动费可成为中华、苹果机沙龙成员,沙龙成员可享受免费拷贝沙龙现有的上千种软件等待遇,详见91年《电子与电脑》第一期。
7. 赠阅电脑信息文摘《家庭电脑》一期。

### 活动形式:

1. 外地会员通过邮寄进行。
2. 本地会员定期举办联谊会,地点设在市少年宫  
联络部地址:长春市岭东路115号星星电脑部  
邮 编:130031  
联系人:董韶华  
开户行:吉林路城市信用社  
账 号:11029424  
电 话:42002 42727 47823(住宅)  
来访路线:乘61路或1路车终点下南走,岭东商场  
对面。

# BJS-51 的监控程序

北京航空航天大学 盛焕鸣

```
0000: 00 C1 FF D2 04 02 00 C2 02 08 EC 02 60 3B 02 08 02A0: F0 21 29 61 BC 01 3C 74 01 11 28 90 60 25 74 A2
0010: ADA5 12 02 60 43 02 0A 26 21 30 02 60 4B 02 08 02B0: F0 A3EEF0 A3 74 22 F0 12 60 25 E4 92 E0 91 BE
0020: D9A5 A5 02 60 53 64 D2 02 0A 56 02 60 5B 78 0F 02C0: 40 0B A2E0 90 60 25 74 92 F0 12 60 25 21 29 EC
0030: D8FE 22 00 00 00 00 00 00 00 90 75 81 30 75 98 02D0: 60 0E B4 01 0E 74 01 11 28 A830 A61F 18 A61E
0040: DA75 88 81 75 89 12 75 A8 01 75 B8 00 75 D018 02E0: 02 09 DC90 02 FC11 1E 21 29 D202 EC60 F1 74
0050: 75 20 20 C2 08 4D 4F 56 03 F4 E4 F5 8DF5 8B 20 02F0: 01 11 28 A830 A61F 18 A61E 61 CC3F 3F 0D 24
0060: B0 FDD2 8E 11 2E 20 B0 F1 30 B0 FD11 2E 20 B0 0300: BC 01 E074 01 11 28 90 60 25 74 E5 F0 A3EEF0
0070: FDC2 8E 78 FF 08 C3 E5 8B 94 40 F5 8B 50 F6 E5 0310: A374 22 F0 12 60 25 91 BE 40 0DC0 E0 90 60 25
0080: 8D94 00 F5 8D 50 EEE5 8B 21 20 20 E7 01 08 E8 0320: 74 F5 F0 D0E0 12 60 25 21 29 D200 BC03 B4 90
0090: F4 04 F5 8D 75 89 23 D2 8E 02 11 70 C2 98 C2 01 0330: 60 28 79 06 78 1AE0 F6 A308 D9FA 8A 82 8B 83
00A0: 12 10 A5 42 4A 4A 59 2D 2D 35 31 2D 56 31 2E 30 0340: E0 C0 E0 D1 AD 40 02 D2 03 AA 82 AB 83 8E 82 8F
00B0: 2D 39 31 2E 37 8D 75 81 3F 75 3E 30 75 3F 40 75 0350: 83 D0 E0 F0 30 00 08 78 0F 79 00 D9 FED8 FAA3
00C0: D0 00 C0 D0 A2 AF 92 07 D0 D0 C2 AF 85 81 30 75 0360: AE 82 AF 83 30 03 D5 21 29 90 60 24 E0 91 BE 90
00D0: 81 30 C0 D0 C0 83 C0 82 C0 E0 90 60 20 D0 E0 F0 0370: 60 24 F0 21 29 90 60 24 E0 F5 D0 E8 C0 D0 75 D0
00E0: A3 E5 F0 F0 A3 D0 E0 F0 A3 D0 E0 F0 A3 D0 E0 F0 0380: 18 91 BED0 D0 F8 75 D0 18 21 29 90 60 24 E0 F5
00F0: 75 D0 18 30 02 1F 30 98 0B C2 98 E5 99 54 7F B4 0390: D0 E9 C0 D0 75 D0 18 91 BED0 D0 F9 75 D0 18 21
0100: 03 02 D2 03 74 02 11 28 A8 30 E6 B5 1F 07 18 E6 03A0: 29 12 09 3D 7F 03 A1 F4 E5 30 14 14 91 BE 04 04
0110: B5 1E 02 D2 03 30 01 03 02 09 42 30 02 08 02 09 03B0: F5 30 21 29 12 EC 60 10 B4 02 02 D2 02 74 01 11
0120: DC 12 09 42 61 A4 20 04 F8 C2 03 C2 00 F1 25 74 03C0: 28 A830 A61F 18 A6 1E D201 D204 12 09 3D 02
0130: 2A 11 08 71 D2 91 27 90 60 00 AA 82 AB 83 79 00 03D0: 09 DC90 60 00 11 16 B4 7F 02 80 08 A2 E5 82 E6
0140: 78 0F E0 F5 F0 31 9B B4 00 02 41 E3 B5 F0 17 A3 03E0: 50 02 C2 E5 F0 B4 0D 02 81 0C B4 08 04 91 10 61
0150: E0 B4 0D 02 21 73 B4 20 02 21 73 F5 F0 08 31 9B 03F0: D5 B4 18 0B A8 82 B8 00 02 80 DA 91 10 80 F5 B4
0160: B5 F0 02 80 EA 18 09 8A 82 8B 83 31 9B 08 30 D5 0400: 7F 04 91 10 80 CFA3 A8 82 B8 20 C9 A8 82 08 22
0170: FA 80 CF 20 00 18 E9 25 E0 F5 F0 A5 31 12 60 60 0410: A8 82 B8 00 01 22 15 82 74 08 11 08 12 08 EA 74
0180: 93 C0 E0 83 E5 F0 04 93 F5 82 D0 83 E4 73 E9 90 0420: 08 11 08 22 D3 80 01 C3 90 5F FF 7C 00 A3 D8 01
0190: 02 70 93 FE 11 0E 12 08 E6 61 07 02 17 E0 C2 E7 0430: 22 40 04 E0 B4 20 F6 A9 82 7D 00 7E 04 7F 05 89
01A0: 22 41 43 C3 C2 42 41 53 49 C3 42 49 D4 C4 53 C3 0440: 82 E0 B4 20 09 09 D8 F7 EC 60 01 0C 80 54 89 1A
01B0: 44 C3 44 C9 44 50 54 D2 44 D8 C5 CC 4C C3 C7 CD 0450: E0 B4 2C 08 0D 0D 09 0C D8 E5 80 46 A3 D8 02 80
01C0: 4DC5 4D D0 50 53 D7 D2 52 45 C7 52 B0 52 B1 53 0460: 04 E0 B4 2C F7 A9 82 DF 02 80 37 EAB5 82 03 E4
01D0: C4 53 D0 53 D8 D4 43 C6 43 2DC3 44 C1 44 49 B4 0470: 80 0F 15 82 E0 44 20 C3 94 30 B4 0A 00 40 02 94
01E0: 44 4D FF 45 C4 FFFF 41 43 C3 C2 44 50 C8 44 50 0480: 27 F5 F0 54 F0 70 31 EF 20 E0 04 AB F0 80 D8 E5
01F0: CC 49 C5 49 D0 50 43 4F CE 50 B0 50 B1 50 B2 50 0490: F0 C4 42 1B C0 82 74 28 2DF5 82 EBF0 D0 82 0D
0200: B3 50 53 D7 52 43 41 50 32 CC 52 43 41 50 32 C8 04A0: 80 C5 0C E8 60 0C EC B5 1E 02 80 06 7F 05 09 D8
0210: 53 42 55 C6 53 43 4F CE 53 D0 54 43 4F CE 54 4C 04B0: 8E 0C ECA 2E 1 92 06 22 90 1F BE 11 1E 22 FE 11
0220: B0 54 48 B0 54 4C B1 54 48 B1 54 4C B2 54 48 B2 04C0: 0E 74 2D 11 08 74 01 12 0A 65 71 D2 91 24 74 01
0230: 54 4D 4F C4 54 32 43 4F CE 00 02 8B 02 97 10 02 04D0: 11 28 C3 EC 70 01 D3 EE 22 90 60 23 E0 FF 11 0E
0240: 02 A4 05 E1 06 19 07 28 04 F0 04 D9 07 2A 02 CF 04E0: 15 82 E0 91 BE 90 60 22 EEF0 05 82 EFF0 21 29
0250: 05 53 06 6F 02 EA 03 2C 03 2A 06 C9 03 69 03 A1 04F0: 7B 10 74 01 11 28 A5 71 BF 20 00 40 4F 8E 82 8F
0260: 03 00 03 75 03 8B 07 99 03 A8 07 E6 03 B5 1F C5 0500: 83 C0 82 C0 83 BC 03 0A 74 03 11 28 AC 1E AD 1F
0270: E0 F0 83 82 A8 B8 87 80 90 A0 B0 D0 CACB 99 98 0510: D2 00 74 02 11 28 D0 83 D0 82 10 00 04 AC 82 AD
0280: 81 88 8A 8C 8B 8D CCCC 89 C8 00 90 60 20 E0 91 0520: 83 F1 25 ED 11 0E EC 12 09 AE 12 17 8A 20 03 11
0290: BE 90 60 20 F0 21 29 90 60 21 E0 91 BE 90 60 21 0530: 20 06 04 DB EC 80 0A E5 83 B5 1F 12 E5 82 B5 1E
```

0540: 0D AE 82 AF 83 74 01 12	0A 65 F1 25 21 29 40 D1	0830: B4 2E 02 80 80 B4 0D 02	80 0D B4 58 02 01 8C C0
0550: 80 EF 01 7A 01 74 01 11	28 C2 05 11 16 B4 1B 02	0840: F0 12 07 D6 D0 F0 F0 A3	E582 70 B4 E5 83 70 B0
0560: 80 35 B4 3A F4 7A 00 B1	B4 60 27 F8 B1 B4 F5 83	0850: 90 0A 22 11 D9 80 28 BC	0428 AA 1E AB 1F 74 03
0570: B1 B4 F5 82 EC 60 0A EE	25 82 F5 82 EF 35 83 F5	0860: 51 56 AC 1E AD 1F 74 04	51 56 D0 83 D0 82 E0 B5
0580: 83 B1 B4 B1 B4 F0 A3 D8	FAB 1 B4 EA 60 CBD 2 00	0870: 1C 02 EE F0 A3 E5 83 B5	1B F4 E5 82 B5 1A EF 02
0590: 80 C7 11 16 B4 1B FB C0	82 C0 83 90 1F A6 11 1E	0880: 01 29 D0 83 D0 82 BC 03	03 02 02 E3 11 D5 E5 83
05A0: D0 E0 11 0E D0 E0 11 0E	30 00 05 90 05 D1 11 1E	0890: 11 ADE 5 82 11 AD 11 E6	E0 12 07 C1 B4 2E 02 80
05B0: D2 05 21 29 11 16 B1 C6	C4 F9 11 16 B1 C6 49 C0	08A0: DE B4 0D 02 80 04 02 12	C0 F0 A3 80 DF C0 E0 C4
05C0: E0 2A FAD 0 E0 22 20 E6	04 C3 94 30 22 C3 94 37	08B0: 54 0F 24 10 83 11 EC D0	E0 C0 E0 54 0F 24 05 83
05D0: 22 43 68 65 63 6B 73 75	6D 20 45 72 72 6F 72 0D	08C0: 11 EC D0 E0 22 30 31 32	33 34 35 36 37 38 39 41
05E0: 24 7F 08 79 80 E7 F5 F0	F4 F7 E7 65 F0 04 A7 F0	08D0: 42 43 44 45 46 74 0D 01	EC 11 D5 E4 93 B4 24 01
05F0: 70 02 7F 10 E4 F8 79 00	F1 25 E8 11 0E 74 3A 11	08E0: 22 11 ECA 380 F5 74 3A	11 EC 74 20 C0 E0 30 98
0600: 08 12 08 EA E6 11 0E 08	09 E9 B4 08 03 12 08 EA	08F0: 1C C2 98 E5 99 54 7F B4	13 09 30 98 FDC 2 98 E5
0610: E9 30 E4 ED DFE 0 21 29	71 BC 02 20 D1 5D 79 20	0900: 99 54 7F B4 03 02 D2 03	B4 10 02 B2 08 C2 A8 39
0620: D2 90 7E 00 EBD 1 3E E5	82 D1 3E EDD 1 3E ECD 1	0910: 99 FDC 2 99 D0 E0 F5 99	D2 A8 30 08 03 12 1F B1
0630: 3E E0 D1 3E D1 AD 40 F9	E4 9E D1 3E 21 29 7F 09	0920: B4 0D 18 30 99 FDC 2 99	75 99 0A C0 F0 75 F0 70
0640: CE 2E CEC 2 90 D3 D1 54	13 92 90 DFF 9 7A 12 D1	0930: E4 D5 E0 FDD 5 F0 F9 D0	F0 74 0D 22 75 90 09 B7
0650: 56 D1 BC 22 7A 06 7B 7D	DB FED A FA 22 90 60 28	0940: 11 D9 A8 30 E6 F5 83 11	AD 18 E6 F5 82 C0 82 C0
0660: 79 06 78 1A E0 F6 A3 08	D9 FA 8A 82 8B 83 22 79	0950: 83 31 AE 90 60 20 E0 31	AE A3 E0 31 AE E5 30 14
0670: 20 7E 00 D1 91 F5 83 D1	91 F5 82 D1 91 FDD 1 91	0960: 14 31 AE 90 60 23 E0 11	AD 15 82 E0 31 AE 90 60
0680: FC D1 91 F0 D1 AD 40 F9	D1 91 BE 00 02 21 29 A1	0970: 24 E0 F5 D0 F5 F0 E8 75	D0 18 31 AE 85 F0 D0 E9
0690: AB 7F 08 E4 20 97 FD 7A	03 D1 56 20 97 F6 D1 54	0980: 75 D0 18 31 AE 7A 08 E5	F0 33 C0 E0 40 06 74 30
06A0: A2 97 13 DF F9 CE 2E CE	D1 54 D1 BC 22 E5 82 B5	0990: 11 EC 80 04 74 31 11 EC	D0 E0 DA ED 74 42 11 EC
06B0: 1C 07 E5 83 B5 1D 02 C3	22 A3 D3 22 C0 E0 11 0E	09A0: 11 EAD 0 83 D0 82 51 74	11 D5 20 01 2F 22 11 AD
06C0: D9 04 79 20 F1 25 D0 E0	22 BC 03 45 D1 5D 75 A0	09B0: 74 48 11 EC 11 EA 22 50	43 20 20 20 20 41 43 43
06D0: E0 7A 80 12 17 2B B5 F0	11 0E BE 00 01 0F D1 AD	09C0: 20 42 20 20 20 53 50 20	20 44 50 54 52 20 20 52
06E0: 40 F1 12 17 06 74 9B F2	21 29 DAE 7 90 05 DA 11	09D0: 30 20 20 52 31 20 20 50	53 57 0D 24 90 60 24 E0
06F0: 1E 90 1F AB 11 1E EF 11	0E E1 20 02 08 D5 FF 12	09E0: C0 E0 15 82 E0 C0 E0 15	82 E0 C0 E0 15 82 E0 F5
0700: 14 4D 02 14 F6 01 0F B1	AD 40 C9 75 90 FF 74 9B	09F0: F0 15 82 E0 D0 82 D0 83	D0 D0 85 30 81 20 02 0C
0710: F2 21 29 90 05 DA 11 1E	90 1F AB 11 1E EF 11 0E	0A00: 20 01 0C C0 D0 A2 07 92	AF D0 D0 22 30 03 06 C2
0720: EE 11 0E E1 0B 02 08 D5	C2 00 C2 01 02 12 E2 02	0A10: 01 C2 02 B2 03 10 04 05	D2 AFD 2 89 22 D2 AFD 2
0730: 13 00 0E B4 20 00 40 05	54 7F 02 08 EC 02 08 EA	0A20: 89 32 4E 6F 0D 24 C2 A8	30 98 FDC 2 98 E5 99 54
0740: E5 82 54 0F F9 E1 49 79	00 F1 25 A5 41 E5 83 11	0A30: 7F D2 A8 B4 03 02 D2 03	B4 10 04 B2 08 80 E7 30
0750: 0E E5 82 11 0E 74 3A 11	08 12 08 EA 30 00 04 E4	0A40: 05 13 B4 0D 02 01 D5 B4	08 01 22 B4 0A 01 22 B4
0760: 93 E1 64 E0 F1 2F 30 03	02 E1 8E 30 06 0E E5 82	0A50: 7F 01 22 11 EC 22 90 60	28 14 23 25 82 F5 82 E0
0770: B5 1E 09 E5 83 B5 1F 04	F1 25 E1 8E A3 09 E9 B4	0A60: FE A3 E0 FF 22 90 60 28	14 23 25 82 F5 82 EE F0
0780: 08 03 12 08 EAE 9 30 E4	D0 20 06 BB DB B9 AE 82	0A70: A3 EF F0 22 AC 82 AD 83	E4 93 F9 C0 82 C0 83 B4
0790: AF 83 74 01 12 0A 65 21	29 ECB 4 01 FA 90 60 28	0A80: D6 02 E1 E3 B4 D7 02 E1	E8 54 0F B4 01 02 E1 96
07A0: E0 F9 F1 25 E9 11 0E 12	08 E6 E7 F1 C1 B4 0D 02	0A90: 90 0C 56 70 03 90 0B B6	B4 02 03 90 0B D6 B4 03
07B0: 80 0C B4 2E 02 80 E0 C0	19 F1 D6 D0 19 F7 09 80	0AA0: 03 90 0B F6 B4 04 03 90	0C 16 B4 05 03 90 0C 36
07C0: E1 C0 82 C0 83 11 0E 74	2D 11 08 71 D2 90 60 00	0AB0: E9 54 F0 C4 23 C0 E0 93	F5 F0 D0 E0 04 93 85 F0
07D0: E0 D0 83 D0 82 22 C0 82	C0 83 91 24 74 01 11 28	0AC0: 83 F5 82 E4 93 70 06 D0	83 D0 82 61 B2 20 E7 11
07E0: D0 83 D0 82 EE 22 74 01	11 28 8E 82 8F 83 C0 82	0AD0: B4 05 00 50 07 FA 11 EA	D8 FC 80 02 11 ECA 3 80
07F0: C0 83 74 01 33 11 28 BC	02 5D 02 12 35 82 8E F0	0AE0: E2 D0 83 D0 82 B4 80 02	61 9E C0 E0 20 E5 19 20
0800: E0 B5 F0 43 E5 83 F5 A0	11 ADE 5 82 54 F0 F9 11	0AF0: E4 06 F1 4D F1 FB 80 19	B4 D8 09 A3 E4 93 11 AD
0810: AD 11 E6 E3 11 AD 11 EA	E9 54 0F 04 09 B4 10 F3	0B00: F1 FB 80 0D F1 8C 80 F8	20 E4 04 D1 FC 80 02 F1
0820: 11 D5 E5 83 11 ADE 5 82	11 AD 11 E6 E0 12 07 C1	0B10: 32 83 D0 E0 54 0F 23 C0	82 C0 83 A5 02 90 0B 21

0B20; 73 61 33 61 39 61 47 61	51 61 60 61 6C 61 78 61	0E10; D0 4D 55 4C 03 41 42 00	43 4A 4E 45 02 41 2C 23
0B30; 8C 61 96 D083 D082 61	B2 D083 D082 71 5B 74	0E20; D2 53 57 41 50 02 41 00	46 41 03 20 41 00 43 4C
0B40; 23 11 ECF1 8C 61 B0 D0	83 D082 71 5B D1 FC 61	0E30; 52 03 41 00 43 50 4C 03	41 00 49 4E 43 03 C0 44
0B50; B2 D083 D082 71 5B F1	32 61 B2 74 2C 11 EC22	0E40; 45 43 03 C0 41 44 44 03	41 2C C0 41 44 44 43 02
0B60; D083 D082 71 5B 74 41	11 EC 61 B2 D0 83 D082	0E50; 41 2C C0 4F 52 4C 03 41	2C C0 41 4E 4C 03 41 2C
0B70; 71 5B 74 43 11 EC 61 B2	D083 D082 71 5B 74 23	0E60; C0 58 52 4C 03 41 2C C0	4D4F 56 03 C1 4D4F 56
0B80; 11 ECF1 8C 71 5B F1 FB	D1 FC 61 B2 D0 83 D082	0E70; 03 80 53 55 42 42 02 41	2C C0 3F 00 43 4A 4E 45
0B90; 71 5B F1 4D 61 B0 D083	D082 F1 8C 61 B0 C0 82	0E80; 02 41 2C C2 58 43 48 03	41 2C C0 44 4A 4E 5A 02
0BA0; A883 A3F1 4D 88 83 D0	8271 5B F1 4D F1 FBA3	0E90; C2 4D 4F 56 03 41 2C C0	4D4F 56 03 C4 49 4E 43
0BB0; F1 FBA3F1 FB 22 0C 76	0C 7A 0C 7F 0C 83 0C 88	0EA0; 03 F0 44 45 43 03 F0 41	44 44 03 41 2C F0 41 44
0BC0; 0C 8C 0C 91 0C 95 0C 9A	0CA0 0C AB0C B3 0C BB	0EB0; 44 43 02 41 2C F0 4F 52	4C 03 41 2C F0 41 4E 4C
0BD0; 0C C1 0C C6 0C D3 0C E0	0CE 6 0C ED0C F1 0C F6	0EC0; 03 41 2C F0 58 52 4C 03	41 2C F0 4D 4F 56 03 F1
0BE0; 0C FB 0D 00 0D 05 0D 0C	0D13 0D 18 0D 1F 0D 24	0ED0; 4D 4F 56 03 C3 53 55 42	42 02 41 2C F0 4D 4F 56
0BF0; 0D 29 0D 2F 0D 3A 0D 45	0D 4A 0D 50 0D 55 0D 5B	0EE0; 03 F7 43 4A 4E 45 02 F6	58 43 48 03 41 2C F0 44
0C00; 0D 60 0D 65 0D 6A 0D 76	0D 83 0D 92 0D 9B 0D A1	0EF0; 4A 4E 5A 02 F2 4D 4F 56	03 41 2C F0 A3E493 C0
0C10; 0DA70DAE0DB9 0DC4	0DCA0DD0 0D D8 0DE1	0F00; 82 C0 83 8D 83 8C 82 A3	AG82 AD83 A320 E7 0B
0C20; 0DE9 0DF1 0DF9 0E 01	0E 08 0E 11 0E 18 0E 21	0F10; 25 82 F5 F0 E5 83 50 01	04 80 08 25 82 F5 F0 E5
0C30; 0E 28 0E 2E 0E 34 0E 3A	0E3F 0E 44 0E 4B 0E 53	0F20; 83 34 FFD0 83 D082 11	ADE5 F0 11 AD74 48 11
0C40; 0E 5A 0E 61 0E 68 0E 6D	0E72 0E 7A 0E 7C 0E 84	0F30; EC 22 E920 E3 04 74 40	11 EC 74 52 11 ECE9 20
0C50; 0E 8B 0E 91 0E 98 0E 9D	0EA2 0E A7 0E AE0E B6	0F40; E3 07 54 01 24 30 11 EC	2254 07 80 F7 A37A 01
0C60; 0E BD0E C4 0E CB0E D0	0ED5 0E DD0E E2 0E E8	0F50; E4 93 F5 F0 C0 82 C0 83	90 02 70 E4 93 60 22 B5
0C70; 0E EF0E F5 00 55 4E 4F	50 00 4A 42 43 03 D2 4A	0F60; F0 02 80 04 A30A 80 F3	90 01 E6 DA02 80 08 A3
0C80; 42 04 D2 4A 4E 42 03 D2	4A43 04 E0 4A 4E 43 03	0F70; E4 93 20 E7 F6 80 F8 A3	E493 11 EC20 E7 08 80
0C90; E0 4A 5A 04 E0 4A 4E 5A	03E0 53 4A 4D 50 02 E0	0F80; F6 D0 83 D0 82 80 A2D0	83D0 82 22 A3E493 11
0CA0; 4D 4F 56 03 44 50 54 52	2C 23 D8 4F 52 4C 03 43	0F90; AD74 48 11 EC22 E9 20	E4 04 74 08 80 02 74 0F
0CB0; 2C 2F D0 41 4E 4C 03 43	2C 2F D0 50 55 53 48 02	0FA0; 04 C0 E0 83 20 E7 14 11	ECD0 E0 80 F3 41 4A 4D
0CC0; C0 50 4F 50 03 C0 4D 4F	56 58 02 41 2C 40 44 50	0FB0; 50 20 20 80 41 43 41 4C	4C20 80 D0E0E493 F5
0CD0; 54 52 00 4D 4F 56 58 02	40 44 50 54 52 2C 41 00	0FC0; F0 A3 E493 8C 82 8D 83	A3F5 82 E5 F0 23 23 23
0CE0; 4C 4A 4D 50 02 D8 4C 43	41 4C 4C 20 D8 52 45 54	0FD0; 54 07 53 83 F8 45 83 11	ADE5 82 D083 D082 F1
0CF0; 00 52 45 54 49 00 4F 52	4C 03 C4 41 4E 4C 03 C4	0FE0; 8F 61 AF90 1F 8D 80 03	90 1F 99 E4 93 20 E7 05
0D00; 58 52 4C 03 C4 4F 52 4C	03 43 2C D0 41 4E 4C 03	0FF0; 11 EC A380 F6 D0 83 D0	8261 B2 0C BC00 01 0D
0D10; 43 2C D0 4D 4F 56 03 D5	4D4F 56 03 43 2C D0 43	1000; 22 58 E4 F5 D0 F5 0E F5	0D75 81 55 D1 F8 30 B5
0D20; 50 4C 03 D0 43 4C 52 03	D0 53 45 54 42 02 D0 4D	1010; 1E 11 A5 0D 4D 43 53 2D	35 31 20 54 69 6E 79 20
0D30; 4F 56 58 02 41 2C 40 52	30 00 4D 4F 56 58 02 40	1020; 42 61 73 69 63 20 56 32	2E 32 61 8D 02 18 46 D2
0D40; 52 30 2C 41 00 52 52 04	41 00 52 52 43 03 41 00	1030; 42 D2 41 02 18 67 30 98	FDE5 99 C2 98 54 7F B4
0D50; 52 4C 04 41 00 52 4C 43	03 41 00 4F 52 4C 03 C1	1040; 03 03 02 10 CEB4 61 00	40 07 B4 7B 00 50 02 54
0D60; 41 4E 4C 03 C1 58 52 4C	03 C1 4A 4D 50 03 40 41	1050; DF 22 74 0D 30 99 FDC2	99 F5 99 B4 0D 0A 12 09
0D70; 2B 44 50 54 52 00 4D 4F	56 43 02 41 2C 40 41 2B	1060; 23 A5 91 A5 12 00 80 04	C3 D5 50 04 75 50 08 D3
0D80; 50 43 00 4D 4F 56 43 02	41 2C 40 41 2B 44 50 54	1070; 22 30 98 0A 11 36 B4 13	05 11 36 B4 11 FB 22 74
0D90; 52 00 49 4E 43 03 44 50	54 52 00 43 50 4C 03 43	1080; 20 11 54 50 FA 22 54 0F	70 03 20 48 07 C2 48 24
0DA0; 00 43 4C 52 03 43 00 53	45 54 42 02 43 00 4D 4F	1090; 03 83 11 54 22 30 31 32	33 34 35 36 37 38 39 41
0DB0; 56 58 02 41 2C 40 52 31	00 4D 4F 56 58 02 40 52	10A0; 42 43 44 45 46 D0 83 D0	82E4 93 A310 E7 04 11
0DC0; 31 2C 41 00 49 4E 43 03	41 00 44 45 43 03 41 00	10B0; 54 80 F6 11 54 E4 73 02	18 49 FF FFFF FFFF FF
0DD0; 41 44 44 03 41 2C 23 D0	41 44 44 43 02 41 2C 23	10C0; 75 A0 60 78 60 90 10 D5	E493 00 B4 FF 01 22 F2
0DE0; D0 4F 52 4C 03 41 2C 23	D0 41 4E 4C 03 41 2C 23	10D0; A3 08 80 F4 22 90 10 E0	22 FF 11 C0 02 07 F3 FF
0DF0; D0 58 52 4C 03 41 2C 23	D0 4D 4F 56 03 41 2C 23	10E0; 02 8A 02 96 02 A4 12 40	05 E1 12 E0 07 28 04 F0
0E00; D0 44 49 56 03 41 42 00	53 55 42 42 02 41 2C 23	10F0; 04 D9 07 2A 18 93 1A B8	02 EA 05 53 16 A7 16 A5

1100;	19 64 19 40 19 4C 1A D5	19 58 03 69 03 A1 03 00	13F0;	09 EF 91 10 00 7A 80 12	26 56 91 1C 00 74 90 F2
1110;	03 75 03 8B 06 19 07 99	03 A807 E6 03 B5 FFFF	1400;	EF91 16 00 19 EE F3 19	12 06 54 E3 E3D0 1A 22
1120;	ED12 3F B1 FD22 FF FF	FFFFFFFFFFFFFF FFFF	1410;	D2E7 C2E6 F3 22 C2E6	C2E7F3 22 C2E7D2E6
1130;	BC03 13 12 06 5D 75 90	E7 75 A07F E5 83 B5 1D	1420;	F3 22 C083 C082 C0E0	90E403 74 88 F0 90 E4
1140;	08 E5 82 B5 1C 03 02 01	29 79 03 74 9C F3 19 EF	1430;	02 E0 20 E7 F9 90 E4 00	D0E0F0 C0 E0 90 E4 03
1150;	F3 19 EEF3 19 E3 F0 E5	83 B5 1D 90 40 07 E5 82	1440;	74 05 F0 74 04 F0 D0E0	D082 D083 22 75 A078
1160;	B5 1C 00 50 03 02 01 29	A3 0E EE79 01 0F 41 69	1450;	74 C7 75 A070 78 03 74	90 F2 22 E7 F3 19 EAF3
1170;	30 98 FDC2 98 20 B5 03	02 80 00 C2 01 02 00 A0	1460;	19 00 00 E3 E3 22 FFFF	0ABA00 01 0B EBB5 1D
1180;	20 E6 03 54 0F 22 54 0F	24 09 22 90 80 70 12 03	1470;	06 EA B5 1C 02 C3 22 50	PCD3 22 FF FFFF FFFF
1190;	D5C0 1A 75 A060 78 B0	7C 00 90 60 70 E0 B4 0D	1480;	BC03 14 12 06 5D 85 1F	83 85 1E 82 D1 00 F0 A3
11A0;	02 80 0D 31 80 C4 FA A3	E0B4 0D 08 74 00 4A 0C	1490;	D1 1B 40 F8 02 01 29 90	14 9F 12 00 1E 61 14 3F
11B0;	D0 1AC3 22 31 80 4A F2	0C 08 A3E0 B4 20 03 A3	14A0;	3F 8D FFFF FFFF FFFF	BC02 E9 12 06 5DD1 00
11C0;	80 DBD0 1AD3 22 E5 82	70 02 15 83 15 82 22 00	14B0;	B4 FF 07 D1 1B 40 F7 02	01 29 EB 12 00 0E EA 12
11D0;	00 31 8B 74 01 12 00 28	8E 82 8F 83 C0 82 C0 83	14C0;	09 AE 74 3A 12 08 ECD1	00 12 09 AE 74 0D 12 08
11E0;	74 02 12 00 28 D0 83 D0	82 A9 1C 78 B0 E5 83 B5	14D0;	ECD1 1B 40 03 02 01 29	20 03 02 84 AE 80 F6 FF
11F0;	1F 08 E5 82 B5 1E 03 02	01 29 E4 93 F5 F0 E2 A3	14E0;	C0E0 B5 7C 03 D0 E0 22	74 20 12 08 EC 05 7C D0
1200;	08 B5 F0 E5 C0 82 C0 83	D9 1F D0 1F D0 1E 74 01	14F0;	E0 80 EDF FFFF 11 C0	12 1A FB 02 07 F3 FF 91
1210;	12 0A 65 8E 82 8F 83 31	C6 00 E5 83 12 00 0E E5	1500;	01 02 03 01 01 02 01 01	01 01 01 01 01 01 01 01
1220;	82 12 00 0E 12 08 EA 21	D3 E4 93 F5 F0 E2 A3 08	1510;	03 02 03 01 01 02 01 01	01 01 01 01 01 01 01 01
1230;	B5 F0 B2 41 08 75 D0 18	21 D0 00 86 00 FF FFFF	1520;	03 02 01 01 02 02 01 01	01 01 01 01 01 01 01 01
1240;	74 01 12 00 28 8E 82 8F	83 C0 82 C0 83 BC 03 22	1530;	03 02 01 01 02 02 01 01	01 01 01 01 01 01 01 01
1250;	74 03 12 00 28 A8 1E AA	1F85 1F A074 02 12 00	1540;	02 02 02 03 02 02 01 01	01 01 01 01 01 01 01 01
1260;	28 D0 83 D0 82 12 07 25	E5 83 B5 1F 08 E5 82 B5	1550;	02 02 02 03 02 02 01 01	01 01 01 01 01 01 01 01
1270;	1E 03 02 01 29 E0 F5 F0	E2 B5 F0 0A 08 A3 B8 00	1560;	02 02 02 03 02 02 01 01	01 01 01 01 01 01 01 01
1280;	03 0A 05 A0 41 68 E5 83	12 00 0E E5 82 12 00 0E	1570;	02 02 02 01 02 03 02 02	02 02 02 02 02 02 02 02
1290;	74 3A 12 08 ECE0 12 00	0E 12 08 EA 12 08 EAEA	1580;	02 02 02 01 01 03 02 02	02 02 02 02 02 02 02 02
12A0;	12 00 0E E8 12 00 0E 74	3A 12 08 ECE2 12 00 0E	1590;	03 02 02 01 02 02 01 01	01 01 01 01 01 01 01 01
12B0;	12 07 25 20 03 02 41 7C	02 01 29 FFFF FF FF	15A0;	02 02 02 01 01 02 02 02	02 02 02 02 02 02 02 02
12C0;	B4 2C 0E 15 82 E5 82 B4	FF 02 15 83 74 0D 02 08	15B0;	02 02 02 01 03 03 03 03	03 03 03 03 03 03 03 03
12D0;	8C 12 07 D6 02 08 A9 FF	FFFFFFFFFFFFFF FFFF	15C0;	02 02 02 01 01 02 01 01	01 01 01 01 01 01 01 01
12E0;	D2 01 7B 10 74 01 12 00	28 8E 82 8F 83 C0 82 C0	15D0;	02 02 02 01 01 03 02 02	02 02 02 02 02 02 02 02
12F0;	83 74 02 12 00 28 D0 83	D0 82 61 24 00 07 44 FF	15E0;	01 02 01 01 01 02 01 01	01 01 01 01 01 01 01 01
1300;	20 01 03 61 1E 00 B4 20	00 40 0E 10 E7 03 02 08	15F0;	01 02 01 01 01 02 01 01	01 01 01 01 01 01 01 01
1310;	EC 12 08 EC 74 2C 02 08	EC 74 3F 02 08 EC 02 00	1600;	75 A0 E0 79 02 78 03 74	90 F2 EBC2 E6 C2 E7 F3
1320;	0E 02 07 66 E5 82 54 0F	02 07 44 00 22 00 00 00	1610;	19 EA F3 19 00 00 E3 E3	22 FFFF 0A BA 00 01 0B
1330;	85 82 E0 20 E2 06 E0 C0	E0 02 03 43 C0 20 E0 85	1620;	EBB5 1D 06 EAB5 1C 02	C3 22 50 FC D3 22 FF EB
1340;	E0 2F A2 78 92 E1 A2 79	92 E0 A2 7A 92 E3 A2 7B	1630;	70 03 EA 60 31 85 1D 83	85 1C 82 E0 85 83 1D 85
1350;	92 E2 D0 2F C0 E0 02 03	43 18 FFFF FF FF	1640;	82 1C 1C BC FF 01 1D 85	1F 83 85 1E 82 F0 30 00
1360;	FFFFFFFF FFFF FF FF	FFFFFFFFFFFFFF FFFF	1650;	02 D1 67 85 83 1F 85 82	1E 1E BE FF 01 1F 1A BA
1370;	FFFFFFFF FFFF FF FF	FFFFFFFFFFFFFF FFFF	1660;	FFD3 1B BB FF CF 22 78	30 79 00 D9 FED 8 FA 22
1380;	C0 19 75 A0 E0 79 02 78	03 74 80 F2 74 40 F3 D0	1670;	FFED 70 03 EC 60 28 00	00 85 1B 83 85 1A 82 E0
1390;	19 22 D2 E7 C2 E6 F3 22	C2 E6 C2 E7 F3 22 C2 E7	1680;	A385 83 1B 85 82 1A 85	1F 83 85 1E 82 F0 30 00
13A0;	D2 E6 F3 22 18 C0 1A 79	00 78 03 74 80 F2 E0 F3	1690;	02 D1 67 A3 85 83 1F 85	82 1E 00 1C BC FF DA 1D
13B0;	F5 F0 09 EEF3 09 EF 71	92 00 7A 20 12 06 56 71	16A0;	BDFD D6 22 FFD2 00 BC	03 2A 12 06 5DC 3 EC 9A
13C0;	9E 00 74 90 F2 EF 71 98	00 19 EEF3 19 12 06 54	16B0;	C0E0 ED 9B C0 E0 EA 9E	EB 9F 50 15 D0 19 D0 18
13D0;	E3 D0 1A 22 E4 FF FF FF	FFFFFFFFFFFFFF FFFF	16C0;	E8 2E FEE 9 3F FF E8 FA	E9 FB 00 00 D1 2F 02 01
13E0;	C0 1A 79 00 78 03 74 80	F2 E0 F3 F5 F0 09 EEF3	16D0;	29 D0 1D D0 1C 00 00 D1	71 02 01 29 FFFF FFFF



16E0:	FF FF FF FF FF FF BF	40 00 40 07 8E 82 8F 83	19D0:	74 BF F2 75 A0 70 78 02	EF F2 75 A0 78 74 07 F2
16F0:	02 07 34 02 01 29 92 E7	19 89 2AC0E0 C0F0 43	19E0:	75 A0 70 78 00 E222 C0	1F 7F 64 DF FED01F 22
1700:	D0 18 22 02 10 CE C0 19	75 A0E079 02 78 03 74	19F0:	C0 1F C0 1E 7F 64 7E 64	DE FE DFFA D0 1E D0 1F
1710:	80 F2 74 40 F3 D0 19 22	D2 E7C2E6F3 22 C2 E6	1A00:	22 7A 00 12 19 74 12 19	E7 12 19 B2 B5 F0 11 0E
1720:	C2 E7 F3 22 C2 E7 D2 E6	F3 22 18 C0 1A 79 00 78	1A10:	BE00 01 0F 12 06 AD40	E8 75 A078 74 07 F2 22
1730:	03 74 80 F2 E0 F3 F5 F0	09 EEF3 09 EF 12 17 18	1A20:	DAE1 12 19 F0 12 19 B2	12 08 AD7A 80 12 19 74
1740:	7A 40 12 06 56 12 17 24	74 90 F2 EF 12 17 1E 19	1A30:	12 19 F0 12 19 B2 B5 F0	03 02 1A 0F DAEF90 78
1750:	EE F3 19 12 06 54 E1 A6	1A22 E493 F9 C0 83 C0	1A40:	00 74 07 F0 90 05 DA12	00 1E 90 1F AB12 00 1E
1760:	82 75 83 15 F5 82 E4 93	FA F5 7D23 54 FE 25 7C	1A50:	EF12 00 0EE E 12 00 0E	D3 22 C0 83 C0 82 C0 1D
1770:	F5 7C D0 82 D0 83 E4 C0	E0 93 12 00 0E D0E0 04	1A60:	C0 1C C0 1E C0 1F 12 19	74 12 19 F0 0E BE 00 01
1780:	DAF5 74 0C 12 17 90 02	0A 78 75 7C 00 02 17 5A	1A70:	0F 12 06 AD40 F0D01F	D0 1E D0 1C D0 1DD0 82
1790:	C0 E0 B5 7C 03 D0 E0 22	74 20 12 08 EC 05 7C D0	1A80:	D0 83 02 1A 44 D0 1F D0	1E D0 1C D0 1DD0 82 D0
17A0:	E0 E1 90 00 14 F9 E3 E3	D0 1A 22 50 44 12 15 D1	1A90:	83 E0 F5 F0 12 19 B2 B5	F0 AA0E BE 00 01 0F 12
17B0:	FF FF FF FF FF FF FF FF	FFFFFFFF FFFF FF	1AA0:	06 AD40 ED02 1A 19 12	19 B2 B4 FF 97 0E BE 00
17C0:	FF FF FF FF FF FF FF FF	FFFFFFFF FFFF FF	1AB0:	01 0F 12 06 AD40 F0 22	BC 03 11 12 06 5D 8B 83
17D0:	FF FF FF FF FF FF FF FF	FFFFFFFF FFFF FF	1AC0:	8A 82 EE F0 12 06 AD40	F9 02 01 29 90 02 FC 12
17E0:	C2 D5 E8 83 B4 FF 04 D2	00 79 00 30 E7 04 D2 D5	1AD0:	00 1E 02 01 29 BC03 F4	12 06 5D 8F 83 8E 82 EA
17F0:	C2 E7 22 41 43 C3 C2 42	49 D4 43 2DC3 C4 44 C1	1AE0:	FEEB FF 12 19 B2 F0 A3	EEB5 1C 07 EFB5 1D 03
1800:	44 C3 44 C9 44 50 54 D2	44 D8C5 46 C9 C7 CC CD	1AF0:	02 01 29 0E BE 00 01 0F	02 1A E390 60 56 E0 B4
1810:	4D C5 4D 50 CA 4D 50 CB	4D 50 CD4D 50 D2 4D 50	1B00:	A5 1F A3 E0 B4 A5 1A 75	A0 60 78 58 79 30 E2 F5
1820:	D8 50 53 D7 D2 52 45 C7	52 B0 52 B1 53 C3 53 C4	1B10:	83 F7 08 E2 F5 82 19 F7	08 7A 03 E2 F0 A308 DA
1830:	53 D0 53 D8 D4 FF C4 FF	41 43 C3 C2 44 50 C8 44	1B20:	FA 22 C0 82 C0 83 C0 E0	90 61 00 74 C3 F0 90 61
1840:	50 CC 49 C5 49 D0 50 43	4F CE 50 B0 50 B1 50 B2	1B30:	03 E0 20 E5 FC D0 E0 90	61 02 F0 C2 90 D2 90 D0
1850:	50 B3 50 53 D7 52 43 41	50 32 CC 52 43 41 50 32	1B40:	83 D0 82 22 54 48 49 53	20 50 52 4F 47 52 41 4D
1860:	C8 53 42 55 C6 53 43 4F	CE 53 D0 54 43 4F CE 54	1B50:	20 49 53 20 44 45 56 45	4C 4F 50 45 44 20 42 59
1870:	4C B0 54 48 B0 54 4C B1	54 48 B1 54 4C B2 54 48	1B60:	20 53 48 41 4E 47 20 48	55 41 4E 20 4D 49 4E 47
1880:	B2 54 4D 4F C4 54 32 43	4F CE 00 FFFF FFFF FF	1B70:	20 39 31 2C 38 20 51 2C	F0 68 F8 A3 DAF8 51 2C
1890:	02 02 CF EC 60 FA B4 01	12 74 01 12 00 28 A8 30	1B80:	68 60 1C 74 0A 51 4C 01	FA 90 41 F1 E0 90 41 4E
18A0:	A6 1F 18 A6 1E 02 09 DC	02 1ACCB4 02 FA 74 02	1B90:	F0 90 41 F2 E0 90 41 4F	F0 74 FF 51 4C 01 FA 90
18B0:	12 00 28 8F 83 8E 82 C0	83 C0 82 75 A0 60 78 56	1BA0:	41 D0 E0 54 18 7F 08 F5	82 78 40 E0 C0 82 88 82
18C0:	74 A5 F2 08 74 A5 F2 08	E5 83 F2 08 E5 82 F2 08	1BB0:	F0 08 D0 82 A3 DFF4 74	FF 51 4C 01 FA 7A 80 90
18D0:	7A 03 E0 F2 08 A3 DA FA	D0 82 D0 83 74 02 F0 A3	1BC0:	41 80 78 00 51 2C F0 A3	68 F8 DAF8 51 2C 68 60
18E0:	74 18 F0 A3 74 EA F0 02	18 99 C0 D0 A2 AF 92 07	1BD0:	B8 74 0A 51 4C 01 FA 90	41 4E E0 90 41 F1 F0 90
18F0:	D0 D0 C2 AF 85 81 30 75	81 30 CDD0 C0 83 C0 82	1BE0:	41 4F E0 90 41 F2 F0 7A	80 90 41 80 78 00 E0 FB
1900:	C0 E0 90 60 20 D0 E0 F0	A3 E5 F0 F0 A3 D0 E0 F0	1BF0:	51 4C A3 EB 68 F8 DAF6	51 4C 61 9F 7A 20 90 41
1910:	A3 D0 E0 F0 A3 D0 E0 F0	75 D0 18 75 A0 60 78 58	1C00:	00 78 00 E0 FB 51 4C A3	EB 68 F8 DA F6 79 20 7A
1920:	A9 30 E2 F5 83 F7 08 E2	F5 82 19 F7 08 7A 03 E2	1C10:	40 E7 FB 51 4C 09 EB 68	F8 DAF6 7A 20 90 41 60
1930:	F0 A3 08 DA FA 90 60 56	74 FFF0 A3 F0 02 03 A1	1C20:	E0 FB 51 4C A3 EB 68 F8	DAF6 51 4C 61 9F 7B 00
1940:	BC 03 06 12 06 5D 12 1A	01 02 01 29 BC 03 06 12	1C30:	7A 00 78 70 51 2C F6 08	6B FB 1A B8 74 F6 85 70
1950:	06 5D 12 1A 5A 02 01 29	BC 03 06 12 06 5D 12 1A	1C40:	83 85 71 82 51 2C F0 A3	6B FB DA 06 51 2C 6B FB
1960:	91 02 01 29 BC 02 0A 12	06 5DEAFEEB FF 12 1A	1C50:	70 16 E5 82 B5 73 EDE5	83 B5 72 E8 51 2C 6B 70
1970:	A7 02 01 29 75 A0 78 74	FF F2 75 A0 70 74 80 78	1C60:	07 74 FF 51 4C 02 10 FA	74 0A 51 4C 02 10 FA 12
1980:	03 F2 75 A0 78 74 7F F2	75 A0 70 78 01 EEF2 75	1C70:	12 2C 90 41 4B F0 12 12	2C 15 82 F0 12 12 2C 90
1990:	A0 78 74 BF F2 75 A0 70	EF 78 02 F2 75 A0 78 74	1C80:	41 4D F0 12 12 2C 90 41	4C F0 90 41 50 E0 D2 E0
19A0:	3F F2 75 A0 70 E0 F5 F0	78 00 F2 75 A0 78 74 09	1C90:	D2 E1 F0 02 FF FFE4 90	41 4C F0 A3 F0 02 FFFF
19B0:	F2 22 75 A0 78 74 FF F2	75 A0 70 78 03 74 90 F2	1CA0:	12 12 2C 90 41 4F F0 12	12 2C 90 41 4E F0 74 80
19C0:	75 A0 78 74 7F F2 75 A0	70 78 01 EEF2 75 A0 78	1CB0:	90 41 50 F0 02 FFFF 51	2C 90 41 4B F0 51 2C 90

ICC0: 41 4A F0 90 41 50 74 80	F0 02 FF FF 74 FF 51 4C	1E60: 7A 0A 0D 20 20 32 3A 20	31 31 2E 30 35 39 30 4D
ICD0: 02 10 FFFFFFFF	FFFFFF FF FFFF FFFF	1E70: 48 7A 0A 8D 30 98 FD C2	98 E5 99 30 99 FDC2 99
ICE0: FFFF FFFFFFFF	FFFFFF FF FFFF FFFF	1E80: F5 99 B4 31 02 D1 8E B4	32 02 C1 9AC1 2E 90 60
ICF0: FFFF FFFFFFFF	FFFFFF FF FFFF FFFF	1E90: 73 74 66 F0 90 60 7C 74	65 F0 01 90 00 FFFF F7
ID00: FFFF FFFFFFFF	FFFFFF FF FFFF FFFF	1EA0: 02 60 70 02 60 79 FF FF	FFFFFF FFFFFFFF
ID10: FFFF FFFFFFFF	FFFFFF FF FFFF FFFF	1EB0: FF F7 FF FFFFFFFF	FFFFFF FFFFFFFF
ID20: FFFF FFFFFFFF	FFFFFF FF FFFF FFFF	1EC0: FF FF FF F7 FFFFFFFF	FFFFFF FFFFFFFF
ID30: FFFF FFFFFFFF	FFFFFF FF FFFF FFFF	1ED0: FF FF FF FFFF F7 FF FF	FFFFFF FFFFFFFF
ID40: FFFF FFFFFFFF	FFFFFF FF FFFF FFFF	1EE0: FF FF FF FFFFFFFF F7	FFFFFF FFFFFFFF
ID50: 51 2C FA 51 2C FB 51 2C	F5 76 51 2C F5 75 51 2C	1EF0: FF FF FF FFFFFFFF	FFF7 FF FFFFFFFF
ID60: 78 7D 76 00 D6 C4 08 76	00 D6 51 2C 78 7B 76 00	1F00: FF FF FF FFFFFFFF	FFFFFF F7 FFFFFFFF
ID70: D6 08 C4 76 00 D6 AC 76	AD 75 B1 F7 31 54 60 56	1F10: FF FF 00 FFFFFFFF	FFFFFF FFFF F7 FFFF
ID80: B1 EA C0 83 C0 82 31 4A	B1 DDB1 FD F5 70 D0 82	1F20: FF FF FF FFFFFFFF	FFFFFF FFFFFFFF F7
ID90: D0 83 B1 DDB1 FD 85 83	74 85 82 73 B5 70 05 A3	1F30: FF FF FF FFFFFFFF	FFFFFF FFFFFFFF
IDA0: B1 DD 80 D8 EA 51 4C EB	51 4C 8A 83 8B 82 B1 DD	1F40: FF F7 FF FFFFFFFF	FFFFFF FFFFFFFF
IDB0: B1 FD 51 4C 85 73 82 85	74 83 E5 83 51 4C E5 82	1F50: FF FF FF F7 FFFFFFFF	FFFFFF FF FFFFFFFF
IDC0: 51 4C B1 DDB1 FD 51 4C	51 2C 85 73 82 85 74 83	1F60: FF FF FF FFFF F7 FF FF	FFFFFF FFFFFFFF
IDD0: AC 76 AD 75 80 C9 74 FF	51 4C 02 10 FAE5 82 78	1F70: FF FF FF FFFFFFFF F7	FFFFFF FFFFFFFF
IDE0: 7B 12 10 67 E5 83 12 10	67 22 78 7B 12 10 60 F5	1F80: 22 0D 0F F0 52 45 54 2E	0D 80 FF FFFF 58 43 48
IDF0: 82 12 10 60 F5 83 22 EB	70 02 1A 1B 22 90 7E 51	1F90: 44 20 20 41 2C 40 52 30	80 58 43 48 44 20 20 41
IE00: E0 20 E1 0E B1 EA 74 F0	55 83 70 03 43 83 80 E4	1FA0: 2C 40 52 31 A0 FF 4E 65	78 74 20 61 64 72 3A 20
IE10: 93 22 31 65 B1 EAE 0 31	78 22 F9 74 79 28 F8 E6	1FB0: 24 02 1B 22 F5 90 C2 B4	D2 B4 30 B3 FD 22 48 45
IE20: D1 25 D9 F7 22 7F 02 7E	FF DE FE DF FA 22 90 1A	1FC0: 58 20 3F 0D 24 D2 03 24	0A 00 80 03 02 02 E3 90
IE30: 1A 75 A0 60 78 70 7A 12	E4 93 F2 00 A3 08 DAF 8	1FD0: 20 00 93 C0 E0 74 28 93	D0 19 B4 3E EF E7 09 C2
IE40: 12 10 A5 58 54 41 4C 0A	0D 43 48 4F 49 43 45 3A	1FE0: B5 54 2F 09 89 83 A4 25	F0 75 82 A7 02 3F EC FF
IE50: 0A 0D 20 20 31 3A 20 20	36 2E 30 30 30 30 4D 48	1FF0: FFFFFFFF FFFF FFFF FF	FF FF FFFF FF FFFF FD

