



電子

ISSN 1000-1077

744

與電腦

形形色色的游戏棒
和游戏卡



• ELECTRONICS AND COMPUTERS •

电子工业出版社向 读者推荐计算机新书



DOS 使用大全 13.45 元

C 语言大全 12.35 元

UNIX 使用大全 11.65 元

dBASE III Plus 大全 12.85 元

IBMPC 编程指南 7.00 元

DOS / BIOS 使用详解 8.00 元

MS-DOS 高水平程序设计 5.50 元

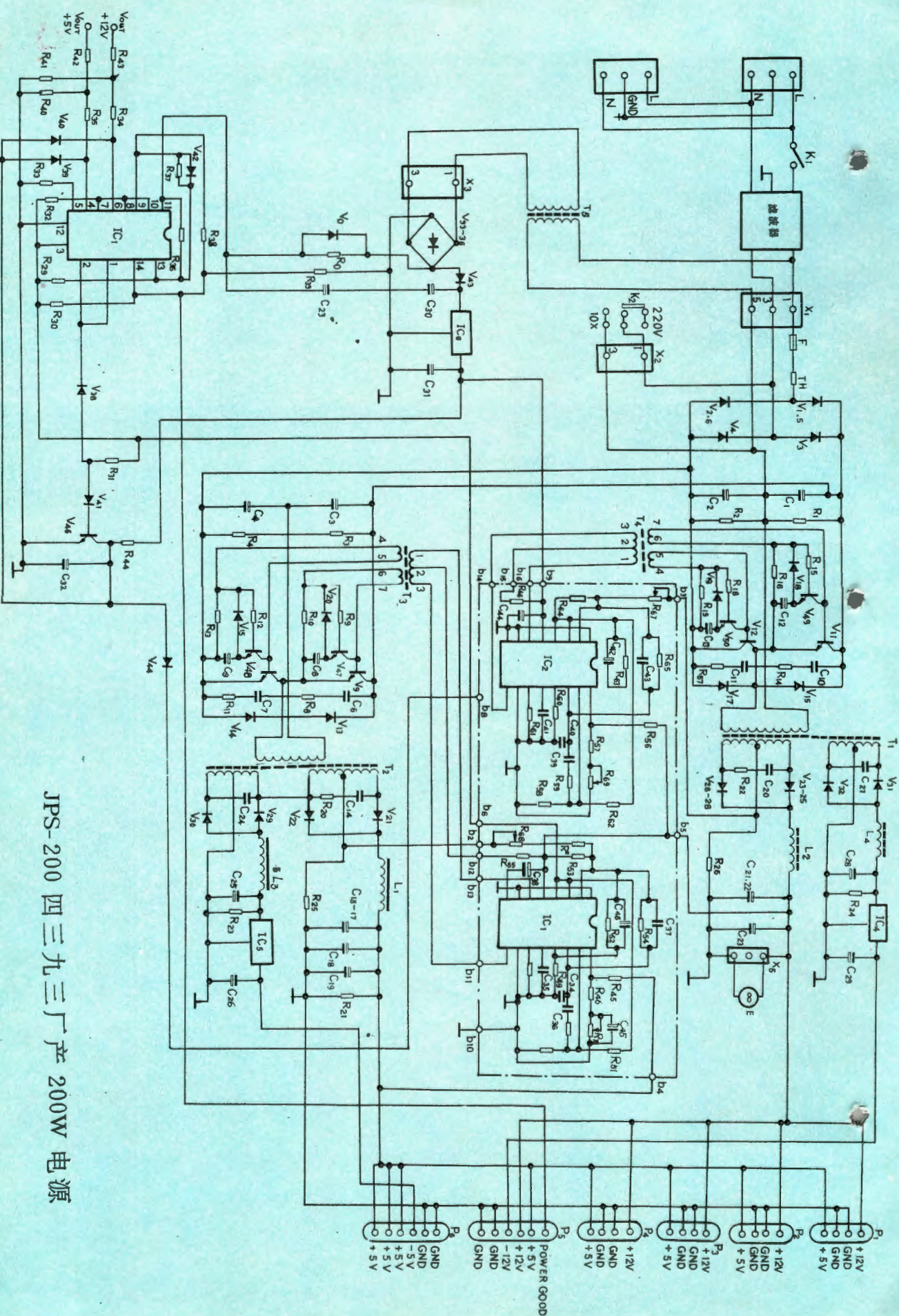
dBase 应用解疑 5.00 元



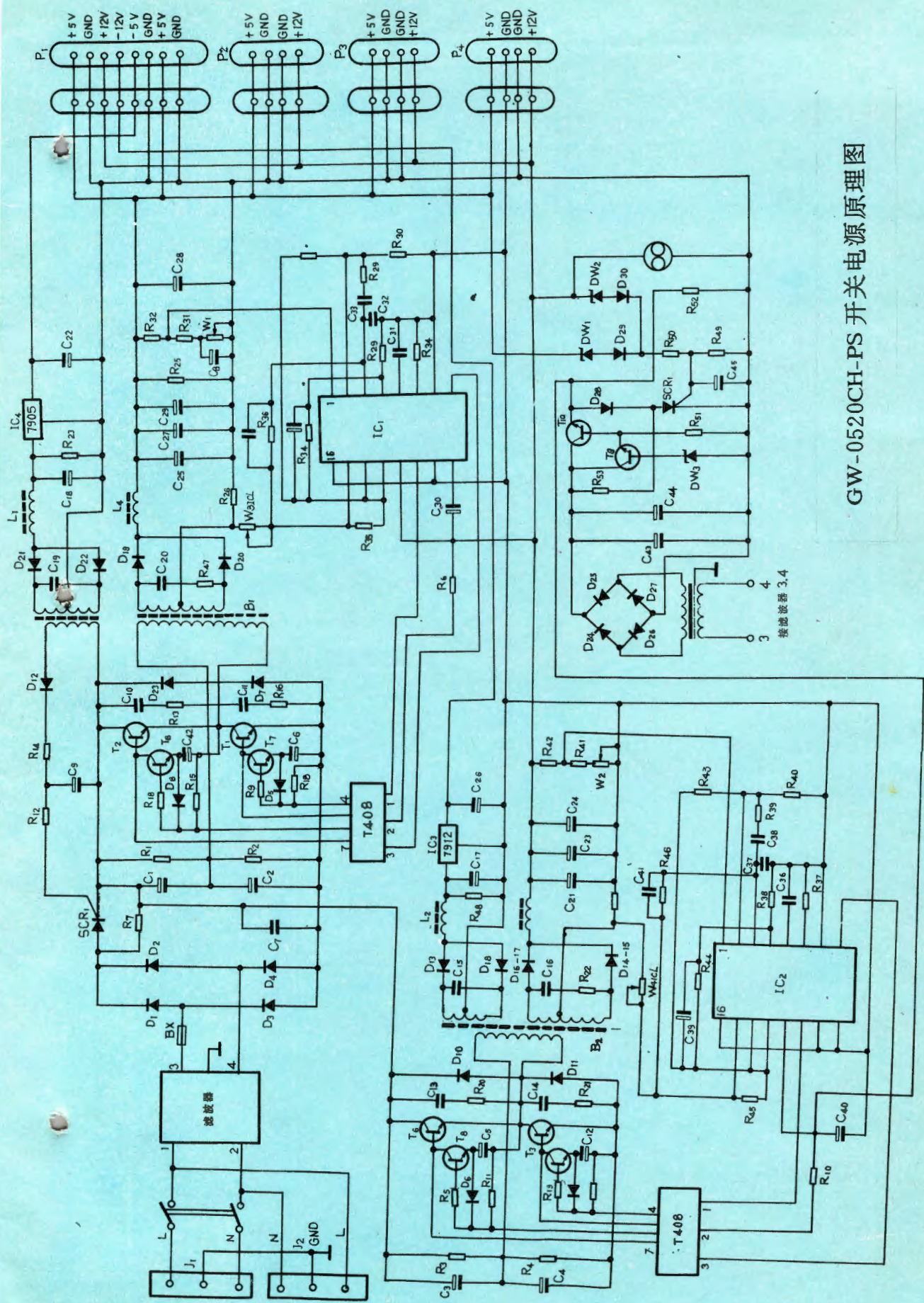
邮购汇款：北京 173 信箱发行部邮购科

邮编：100036 另加 15% 邮费

● 国内代号：2-888 定价：0.95 元



JPS-200 四三九三厂产 200W 电源



GW-0520CH-PS 开关电源原理图



電子

與電腦

創刊於 1989 年 1 月



形形色色的游戏棒和游戏卡

• ELECTRONICS AND COMPUTERS •

一九九一年

总期第80期

電子與電腦

目 录

• 综述 •

- 在中华机上扩充中文输入码的探索 胡志群、胡志广、胡沛云 (2)

• PC 用户 •

- 利用 PASCAL 扩充 dBASE III 的函数功能 王 越 张秀邦 (4)
如何动态生成有记忆功能的菜单 苏士俊 (6)
在 dBASE III 下实现菜单的光带键选 阮自力 (7)
双列分页打印程序 陶 锋 (8)
广州大学一号病毒解除与免疫 张文锋 (10)
谈谈 2.13 汉字系统在软盘上的使用 廖红旭 (11)
PC/XT 机病毒报警程序 王伟智 刘秉瀚 (12)
考生姓名编码自动生成程序 任绥海 (12)

• 学习机之友 •

- 把 APPLE DOS 移入 RAM 卡 傅 剑 (14)
将程序 CHAIN 升级为 APPLE DOS 命令 赵江龙 孙杰亮 (15)
向天坛机移植中华机汉字软件的一些体会与经验 张 斌 (17)
FORTH 问答: 堆栈和逆波兰表示法 丁志伟 (18)
中华学习机图书管理程序 王二珠 (19)
ProDOS 磁盘操作系统入门 (续) 廖 凯 (21)

• C 语言初阶讲座 •

- 第九讲 指针与函数、结构 李文兵 (23)

• 学用单片机 •

- 汽车转弯信号灯控制 张培仁 刘振安 (26)

• 学装微电脑 •

- Z80 CPU 与单片微处理器 8031 的交换 ... 高殿斌 (28)

• 电脑巧开发 •

- 在 APPLE 机接口卡中固化 BASIC 应用程序 解武杰 (30)

- 设计微电脑控制电路的小常识 王 林 (31)
型号不同的绘图仪绘制 TANGO 软件图形的方法 黄 健 陈 军 (32)

• 游戏机电脑 •

- 键盘电视游戏机的 FBASIC 语言及程序设计 (下) 于 春 (33)

• 维修经验谈 •

- 驱动器修理一例 张红平 (35)
维修小经验 张中弦 (35)
硬盘驱动器专用接口电路 501262 的测试 卜建辉 (36)
IBM-PC/XT 一种系统错误的分析 姜金友 (41)
用非格式化方法修复硬盘一例 周建辉 (42)
检修 IBM AS/400B35 故障二例 王志远 (42)
自己学修计算机 陈 勇 (43)
TEC-I 机磁盘驱动器 I/O 接口板故障检修一例 崔时珍 玄勇活 (45)
3070 打印机断针定位方法及 8088 汇编程序 龙兵生 (46)
打印机故障维修——以 STAR 或 GX-15 打印机为例 王 威 王 侯 (47)
普及型 PC 机上可用的印刷电路板辅助设计软件 smARTWORK 张保田 (48)
显示器故障检修一例 张 智 (48)

封面 形形色色的游戏棒和游戏卡

封二、封三 微型计算机电源原理图

封底 电子工业出版社计算机新书介绍

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部

(北京 173 信箱 邮政编码:100036)

印刷:北京三二〇九厂

国内总发行:北京报刊发行局

国内统一刊号:CN11-2199

邮发代号:2-888

国外代号:M924

出版日期:每月 23 日

主编:王惠民 副主编:王昌铭

责任编辑:张 丽

订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:0.95 元



電子
與
電腦

ISSN 1000-1077



• ELECTRONICS AND COMPUTERS •

744

一九九一年

总期第80期

電子與電腦

目 录

• 综述 •

- 在中华机上扩充中文输入码的探索 胡志群、胡志广、胡沛云(2)

• PC 用户 •

- 利用 PASCAL 扩充 dBASE III 的函数功能 王 越 张秀邦(4)
如何动态生成有记忆功能的菜单 苏士俊(6)
在 dBASE III 下实现菜单的光带键选 阮自力(7)
双列分页打印程序 陶 锋(8)
广州大学一号病毒解除与免疫 张文锋(10)
谈谈 2.13 汉字系统在软盘上的使用 廖红旭(11)
PC/XT 机病毒报警程序 王伟智 刘秉瀚(12)
考生姓名编码自动生成程序 任绥海(12)

• 学习机之友 •

- 把 APPLE DOS 移入 RAM 卡 傅 剑(14)
将程序 CHAIN 升级为 APPLE DOS 命令 赵江龙 孙杰亮(15)
向天坛机移植中华机汉字软件的一些体会与经验 张 斌(17)
FORTH 问答:堆栈和逆波兰表示法 丁志伟(18)
中华学习机图书管理程序 王二珠(19)
ProDOS 磁盘操作系统入门(续) 廖 凯(21)

• C 语言初阶讲座 •

- 第九讲 指针与函数、结构 李文兵(23)

• 学用单片机 •

- 汽车转弯信号灯控制 张培仁 刘振安(26)

• 学装微电脑 •

- Z80 CPU 与单片微处理器 8031 的交换 高殿斌(28)

• 电脑巧开发 •

- 在 APPLE 机接口卡中固化 BASIC 应用程序 解武杰(30)

- 设计微电脑控制电路的小常识 王 林(31)
型号不同的绘图仪绘制 TANGO 软件图形的方法 黄 健 陈 军(32)

• 游戏机电脑 •

- 键盘电视游戏机的 FBASIC 语言及程序设计(下) 于 春(33)

• 维修经验谈 •

- 驱动器修理一例 张红平(35)
维修小经验 张中弦(35)
硬磁盘驱动器专用接口电路 501262 的测试 卜建辉(36)
IBM-PC/XT 一种系统错误的分析 姜金友(41)
用非格式化方法修复硬盘一例 周建辉(42)
检修 IBM AS/400B35 故障二例 王志远(42)
自己学修计算机 陈 勇(43)
TEC-I 机磁盘驱动器 I/O 接口板故障检修一例 崔时珍 玄勇活(45)
3070 打印机断针定位方法及 8088 汇编程序 龙兵生(46)
打印机故障维修——以 STAR 或 GX-15 打印机为例 王 威 王 侯(47)
普及型 PC 机上可用的印刷电路板辅助设计软件 smARTWORK 张保田(48)
显示器故障检修一例 张 智(48)

封面 形形色色的游戏棒和游戏卡

封二、封三 微型计算机电源原理图

封底 电子工业出版社计算机新书介绍

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部

(北京 173 信箱 邮政编码:100036)

印刷:北京三二〇九厂

国内总发行:北京报刊发行局

国内统一刊号:CN11-2199

邮发代号:2-888

国外代号:M924

出版日期:每月 23 日

主编:王惠民 副主编:王昌铭

责任编辑:张 丽

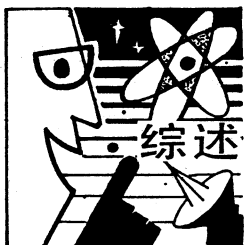
订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:0.95 元



在中华机上扩充中文输入码的探索

胡志群 胡志广 胡沛云

“杞码”是一种普及型的中文输入码，是广西民族学院李冠盛同志研制的，已经通过了技术鉴定。它具有编码规则简单、重码较少、易学、易记、易用等优点，很适合中、小学生应用。而中华学习机是我国普及中、小学电脑教育的机型，能把“杞码”开发应用于中华机是很有意义的。即将出版的《速查学生字典》及《速查实用成语词典》两书也把“杞码”作为正文的编排查检码。这就使得人们在学会“杞码”查字法的同时，也为学“杞码”汉字输入法打下基础。

由于中华学习机的内存有限，直接把“杞码”调入内存 RAM 中是不可能的，所以首先要将“杞码”的信息压缩，让它转换成存储码，再固化到 EPROM 中。本文就是说明按什么规律进行压缩，并说明在扩充程序中如何实现把输入码转换成存储码，再和已固化的码表比较，找出对应的汉字；同时讨论扩充程序与原机内中文管理程序的衔接及调用其子程序等问题。

一、如何打出原机内中文管理程序

为使我们的扩充程序与原有的中文管理程序能相互衔接，并尽可能的调用管理程序中有关的子程序，使扩充程序更简练，故必须打出机内管理程序进行分析。

中华机所用的微处理器 6502 可寻址 64K，其逻辑地址与物理地址不是一一对应的，物理地址分主存和辅存，整个中文系统是在辅存中，所以不能直接打出其管理程序，必须编制一定程序利用软开关的切换把它调入主存后才能由打印机打出。

程序中设四个单元，分别存放管理程序新、旧的首地址，然后令其逐一转移，在开始取数前，先执行子程序 \$C3AB\$，其目的是选择辅存 2。转移完毕执行子程序 \$C3B2\$，让其转主存 1 再返回。

二、码表的转换

一、二两级汉字共 6763 个，每个汉字有五位码。如果直接把杞码存入，则需要 33K 多的内存储单元，这对内存有限的中华机来说是不现实的。为此我们按一定的规律把杞码转换成存储码。这样每个汉字只要两个存储单元就可以了，节省了 3/5 的内存，只占用 13K 多一点就够了。

在杞码表中，应用 15 个英文字母作部首，再加上六个笔形码，在杞码中笔形码是用数字 1—6 表示，但 1—6 的数字键跟选择汉字提示的指令键冲突，因而改用部首没有用到的几个英文字母代替，Q、E、U、I、O、P 分别表示 1—6，现在反过来连同部首的 21 个英文字母分别用 1—21 的数字表示，其对应关系为：

Q E U I O P B C G H J K M N R S T W X Y Z
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21

这样我们把杞码所对应的五位数字码分别以 X_4, X_3, X_2, X_1, X_0 表示，则

存储码 = $X_4 \cdot 7^4 + X_3 \cdot 7^3 + X_2 \cdot 7^2 + X_1 \cdot 7 + X_0$
结果是小于或等于 16 进制的四位数，只用两个单元存储即可。

这个转换工作是让计算机完成的。我们只要把杞码以“T”文件的方式打入计算机，并转存在磁盘上，由于两级汉字的数据太多，故分成四个文件，通过一定程序自动的进行变换，把十进制转换成十六进制，并把转换后的数据自动的存贮在磁盘中。最后把这四个文件合并，就成为杞码的存储码了。

三、码表的固化

虽然把杞码转换成存储码，节省了 3/5 的信息量，但仍有 13K 多的字节，如果把它存在磁盘中，每次使用时再转入内存，这样余下给用户的内存 RAM 空间就太少了，因此有必要把这码表固化在 EPROM 中。我们将它固化在 U_7 的低地址部分，工作时由计算机直接到 U_7 查此表即可。

我们的码表只占用了 U_7 低地址的一部分，因而在固化前必须把我们的码表与 U_7 的余下数据合并后再写入新片中。以后要再写则直接利用磁盘的经整理后的数据即可。

四、扩充程序的编制

1. 编制换指针程序以解决程序间的衔接

中华机以“F₁”作为用户自定义的功能键。现在我们把它作为扩充输入码的状态转换控制键。换指针程序的任务是修改 \$391、\$392 的内容，使它从原来的 RTS 改为显示杞码状态字的入口地址，并把状态标志单元 \$03AE 置为“F₁”的键码（# \$94），把扩充的处理字符程序的入口地址送入 \$38F 与 \$390 单元。这样只要用户按下“F₁”键，就能显示“杞码：”这状态提示字。此后再键入的可显示字符（大于或等于 \$A0）或退格键（\$88）都转到扩充的输入处理程序入口地址去执行。

2. 编制退格键处理程序

退格键码 \$88 是通过累加器转到扩充程序的，可仿效区位及拼音输入时退格键的处理方式，并直接调用其子程序处理，不必考虑软开关的转换。

3. 编制杞码字符的处理程序

首先解决杞码字符的转换与显示。如前所述我们键入杞码时是用 Q、E、U、I、O、P 键代替杞码的数字码 1—6 的，但为了直观还必须通过一定的程序，使在字符区显示的仍然是原杞码中的数字。

杞码一般为 5 位码，不足 5 位的用“J”键结束。遇

到这样的结束符,由程序控制其在键入区自动补零,补足5位为止,无须通过键盘一一键入,因此可提高键入速度。

程序的一部分用于把输入的5位码转换成存储码。然后与码表进行比较。码表所表示的汉字是按区位码的顺序排列的,查找时设一个与区位码对应的指针。当输入码经转换后与码表比较,结果相同时可由指针找到它所对应的汉字,这样把对应于输入码的所有重码字都找出来,放在一定的区域中,并在汉字区显示头六个汉字。还可根据换页指令“>”键显示下6个汉字提示。或根据“<”指令键显示前一幕提示汉字。用户可在每一页中用1-6的数字键选提示字送系统。这种功能是调用拼音输入法所对应的子程序完成的。

以上是扩充程序的主要部分,还有一些细节,如出错时自动响铃并自动返回等。

五、编制使杞码运行的程序

我们是采用软硬件结合的方法,扩充程序没有固化,这样可以随时修改,以增加它的功能。因此在开始使用杞码输入前,首先要运行换指针程序,还要把字符处理程序由磁盘送入内存。这些任务是由BASIC程序实现的。此后凡按“F₄”键即可使用。

本项目在研制过程中得到华明公司袁晓岚先生、清华大学卓卓越老师以及我区计委计算站及计算中心等热情的支持与帮助,特此致谢。

附录:

“杞码”——又称笔型码的编码和检字方法

汉字的笔型分六种(见表一):横(包括提)、竖、撇(包括左向点)、捺(包括右向点)、左向折、右向折,分别用代码1、2、3、4、5、6表示,记作“杞”字笔型代码。

表一 汉字“杞”字笔型代码表

代 码	笔形名称	笔 形	同类笔形
1	横、提	→	— /
2	竖、垂	↓	丨
3	撇、左向点	↙	ノ ㄣ
4	捺、右向点	↘	ㄨ 丶
5	左折	↙	┐ ㄣ ㄣ ㄣ
6	右折	↘	┘ ㄣ ㄣ ㄣ

取码方法:

1. 字的起笔(即书写时的第一笔)不在表二所列常见部首内时,按写字的顺序依表一取五笔(前三笔加末二笔)“杞”字笔形代码,不足五笔的全取。

例:部 41452 干 112
乏 3454 盛 13521

2. 字的起笔在表二所列常见部首内时,先取相应部首代码,后按写字的顺序依表一取四笔(前二笔加末二笔)“杞”字笔形代码,不足四笔的全取。

例:萍 C4412 逛 G1154
马 M 伐 R1634

表二 汉字部首代码表

代 码	部 首
B	贝 广 宀 冫
C	艹 虫
G	广 彳
H	火 禾
J	钅 金
K	口 口
M	门 马 木 目
N	女
R	亻 彳 日 曰
S	纟 纟 石
T	土 扌
W	王
X	乚
Y	讠 衤(衣) 鱼 月
Z	足(足) ㄣ(竹)

(上接第32页)

```

1040 IF c1=44 THEN 1100
1041 IF c5=1 THEN 1070
1042 x(j)=c1
1043 j=j+1
1060 GOTO 1020
1070 y(k)=c1;k=k+1
1080 GOTO 1020
1100 c2=c2+1
1110 IF c2=2 THEN 1130
1111 c5=1
1120 GOTO 1020
1130 LPRINT CHR$(13)
1131 IF x(6)=200 THEN 1135
1132 x(4)=(X(4)-48)*100;X(5)=
(x(5)-48)*10;x(6)=x(6)-48
1133 x1=x(4)+x(5)+x(6)
1134 GOTO 1138
1135 IF x(5)=200 THEN 1137
1136 x(4)=(x(4)-48)*10;x(5)=x(5)-
48;x1=x(4)+x(5);GOTO 1138
1137 x(4)=x(4)-48;x1=x(4)
1138 IF y(6)=200 THEN 1141
1139 y(4)=(y(4)-48)*100;y(5)=(y(5)-48)*10;
y(6)=y(6)-48
1140 y1=y(4)+y(5)+y(6);GOTO 1158
1141 IF y(5)=200 THEN 1143
1142 y(4)=(y(4)-48)*10;y(5)=y(5)-48;y1=y
(4)+y(5);GOTO 1158
1143 y(4)=y(4)-48;y1=y(4)
1158 GET #1,g:g=g+8
1159 c3=ASC(a$)
1160 c3=c3-48
1161 x=c3*COS(0);y=c3*SIN(0)
1162 x2=x1+x;y2=y1+y
1163 LPRINT "M";x2;" ";y2
1164 LPRINT CHR$(13)

```

(下转第29页)



PC 用户

利用 PASCAL 扩充 dBASE III 的函数功能

沈阳工业学院计算机系 王越 沈阳市工程咨询公司 张秀邦

许多用户都感到 dBASE III 的计算能力较差,特别是函数功能欠缺,给系统的进一步开发带来困难。为此,我们分析了 dBASE III 与外部接口的数据交换方式,利用具有丰富函数功能、运行速度快的 TURBO PASCAL 语言完成了 dBASE III 的函数功能扩充。

一、基本思想

dBASE III 与高级语言间进行数据传送的方法,传统上一般以 ASCII 码文件(即 .TXT 文件)进行数据交换,这种方式在进行大批数据的交换时比较理想,而实际应用中往往需要把某几个参数直接传送出去,进行计算后返回。

dBASE III 提供了二种接口与外界交换数据:一是通过 CALL 命令,将参数直接传送给被调用的汇编程序,然后由汇编程序传送给高级语言程序。新版本的 dBASE III 可以用 CALL 命令传送给高级语言形成的 .BIN 文件,但这种方法较为复杂,对一般基层用户不太实用。另一种方法是利用 RUN 命令与宏代换函数 & 将参数传送给高级语言。RUN 命令的格式为:

```
RUN<命令文件名>WITH<参数表>
```

这里,<命令文件名>可为任意的后缀为 .COM 的可执行文件;<参数表>即为执行该文件需要的参数。dBASE III 规定,该参数表须为一系列值参。例如,我们可以写出下述调用格式:

```
RUN CALL-F WITH 'Y','=' , 'COS' , 3.14159
```

这里的自变量是常量;如果它是变量的话,设为 X,则下述调用

```
RUN CALL-F WITH 'Y','=' , 'COS' , X
```

dBASE III 认为是非法的。怎么解决这个问题呢? dBASE III 还提供了一个很有用的宏代换函数 &, 该函数的操作变量只能是字符变量,因此直接写成 &X 是不行的,只能先将 X 变换成字符串,然后代入该参数表。那么上述调用即为:

```
X1=STR(X,10,2)
```

```
RUN CALL-F WITH 'Y','=' , 'COS' , &X1
```

便可达到计算 $Y=\cos(X)$ 的目的。

根据这一原理,我们可以用高级语言编写一个命令文件,该程序对由 RUN 命令传送过来的参数进行接收、分析,然后计算并生成一个 dBASE III 命令文件,将结果写入该文件,在 dBASE III 中执行该文件后,便完成了计算结果的回送。

二、扩充软件实现

利用上述方法,我们编制了一个软件,实现了 dBASE III 函数的扩充。程序组成如下:

- CALL-F.PAS: 是一个用 TURBO PASCAL 编制的程序,经联接生成后,生成命令文件 CALL-F.

COM,其目的就是接收由 dBASE III 程序提供的参数,对其识别后,根据计算公式计算出函数结果,回填到一个 TRANSFER.PRG 文件中。它的调用格式为:

```
RUN CALL-F WITH<结果变量>,<=>,<函数名>  
>,<自变量>
```

- PASCAL-F.PRG: dBASE III 命令文件,将接收的参数进行类型转换等预处理后,调用 CALLF.COM 计算,并执行 TRANSFER.PRG 文件,实现函数结果的回送。该程序调用时要求提供四个参数,从左到右依次为:<结果变量>,<=>,<函数名>,<自变量>。

- TRANSFER.PRG: 回送结果文件,程序结构为:

```
PUBLIC <结果变量>
```

```
<结果变量>=<计算结果>
```

正确调用 CALL-F.COM 后,执行它便得到计算结果。

在该软件中,CALL-F.COM 对用户透明,用户可不去管怎样正确调用它,而只要提供 PASCAL-F.PRG 要求的参数形式即可。

下面是一个将数据库 TEMPT.DBF 中 ANGLE 取出,计算 COS 值后,将结果写到另一字段 COS-VALUE 中的应用程序。数据库结构如下:

Field	Field name	Type	Width	Dec
1	ANGLE	Numeric	10	
2	COS-VALUE	Numeric	10	5

程序为:

```
set talk off
use tempt
set devi to print
@ prow()+1,0 say '计算前数据库中数据显示:'
@ prow()+1,0 say 'ANGLE(角度)'
@ prow(),35 say 'COS-VALUE(余弦值)'
do while .not.eof()
    @ prow()+1,0 say ANGLE pict'9999999999'
    @ prow(),35 say cos- value pict'9999.999999'
    skip
enddo
go top
* 数据库测试
do while .not.eof()
    * 弧度换算
    TEMP-MV=ANGLE * 3.14159/180
    * 计算函数值
    do pascal-f with 'sd','=' , 'cos' ,TEMP-MV
    replace cos- value with sd
    skip
enddo
```

```

go top
@ prow()+3,0 say '计算后数据库中数据显示:'
@ prow()+1,0 say 'ANGLE(角度)'
@ prow(),35 say 'COS-VALUE(余弦值)'
do while .not. eof()
    @ prow()+1,0 say ANGLE pict'9999999999'
    @ prow(),35 say cos-value pict'9999. 999999'
    skip
enddo
use
* 变量测试
abc=3.14159/3
do pascal-f with 'y',' ','cos',abc
@ prow()+2,0 say 'r='
@ prow(),5 say y pict'99999. 9999'
@ prow(),10 say ' '
set devi to screen
return

```

运行结果:

计算前数据库中数据显示:

ANGLE(角度)	COS.VALUE(余弦值)
30	0.00000
60	0.00000
90	0.00000
120	0.00000

计算后数据库中数据显示:

ANGLE(角度)	COS-VALUE(余弦值)
30	0.86602
60	0.50000
90	0.00000
120	-0.50000

Y=0.5000

细心的读者会发现,上述程序对数据库字段操作时,先将 ANGLE 的值赋给一临时内存变量 TEMP.MV,这是因为 dBASE II 的 RUN 命令只承认内存变量而不承认数据库字段变量的缘故。

有关程序清单附后,本软件只写了 COS 函数,读者可根据需要扩充,其实现大同小异,十分容易。

* CALL-F. PAS

program Call-functions-of-PASCAL;

Type str80=string[80];

var

i:integer;
operator:str80;
operand,result:real;
filevar:text;

procedure error(ss:str80);

begin

writeln(chr(7),ss);

assign(filevar,'transfer. prg');

rewrite(filevar);

close(filevar);

exit;

end;

function check-str(ss:str80):boolean;

var

i:byte;

begin

if length(ss)<=0 then error('call function error!');

check-str:=true;

for i:=1 to length(ss) do

if not (ss[i] in ['0'..'9','a'..'z','A'..'Z',' ','-','_']) then check-str:=false;

end;

function upcase-str(ss:str80):str80;

var

i:byte;

begin

if length(ss)>0 then

for i:=1 to length(ss) do ss[i]:=upcase(ss[i]);

upcase-str:=ss;

end;

function change-type(ss:str80):real;

var

p:integer;

value:real;

begin

val(ss,value,p);

change-type:=value;

end;

procedure write-result(ss:str80,result:real);

begin

assign(filevar,'transfer. prg');

rewrite(filevar);

writeln(filevar,'Public ',ss);

writeln(filevar,ss ' ',result:20:5);

writeln(filevar,' ');

close(filevar);

end;

begin

for i:=1 to paramcount do writeln(paramstr(i));

if (paramcount<5)

or (upcase-str(paramstr(1))<>'WITH')

or (not check-str(paramstr(2)))

or (paramstr(3)<>'=')

then error('Parameters Format Error!');

operator:=upcase-str(paramstr(4));

operand:=change-type(paramstr(5));

if operator='COS' then begin

result:=cos(operand);

write-result(paramstr(2),result);

end;

end.

* Name of the procedure:PASCAL-F. prg

Parameter LetVar,Equal,FName,SelfVar

str-data=str(selfvar,15,4)

run callf with &LetVar &Equal &FName &str-data

do transfer

return

如何动态生成有记忆功能的菜单?

北京计算机学院 苏士俊

菜单生成技术,也常用来减少汉字输入。但简单的、常规的菜单却有时满足不了用户的功能要求。比如在录入学生或职工档案时,把民族的汉字名称和编号写成菜单,则用户只需输入相应的编号。但菜单中写出全部民族,不仅增加了选择的困难,也影响屏幕的美观,实际上有的民族可能根本用不上。因此可用动态菜单生成技术弥补其不足。

技术问题是这样解决的:在系统中专门设置一个民族编码提示库,用它来支持系统运行。库内只存放民族名称和对应的编码(用记录号代替编码也是可取的)。在用户输入民族字段的数据时,打开民族编码提示库(这一工作由程序中的语句完成),将其内容按一定格式显示在屏幕上,并询问有无需要的民族(Y/N)?这时用户即可选择相应的编码,代替汉字输入。当菜单中没有需要的民族时,只需键入 N,即可转去向民族提示库中增加一条新记录(汉字的民族名称和对应的编码)。该民族的名称和编码一旦写入数据库,下次再显示菜单时,新的民族和编码会永远出现在菜单中。这种有记忆办法的动态菜单会极大地方便用户。

下面是一个生成动态菜单的示范程序,凡是用户使用过的民族都可以自动出现在民族提示菜单中,其菜单内容会随系统运行而动态变化,用户不妨试一试。

存放民族名称的 MZTSK.DBF 库已有 7 条记录,其库结构和数据如下:

.LIST STRUCTURE

```
库文件结构-- 文件名:C:\MZTSK.dbf
库文件中的记录数量: 7
库文件的最后更新日期: 01/23/91
  字段  字段名      类型      宽度      小数
  1    民族名称    字符型      4
** * 总计 ** *      5
```

.LIST

```
记录号: 民族名称
  1    汉族
  2    回族
  3    苗族
  4    壮族
  5    满族
  6    白族
  7    傣族
```

动态生成菜单的程序如下:

* 民族菜单显示:

```
SET TALK OFF
RK=" "
Y12=" "
DO WHILE .T.
  CLEAR
```

```
@3,8 SAY " * * 民族提示 * * "
```

```
USE MZTSK
```

```
SET HEADING OFF
```

```
LIST
```

```
SET HEADING ON
```

```
IF RK="Y"
```

```
  EXIT
```

```
ENDIF
```

```
@1,8 SAY "如果没有需要的民族,"
```

```
@1,28 SAY "请按 N 键,否则按其它键!"GET Y12 PICTURE "I"
```

```
* 增加新民族:
```

```
IF Y12<>"N"
```

```
@1,0 SAY SPACE(80)
```

```
EXIT
```

```
ELSE
```

```
  APPEND BLANK
```

```
  RK="N"
```

```
DO WHILE RK<>"Y"
```

```
@2,10 SAY "新民族:"GET 民族
```

```
READ
```

```
@2,30 SAY "认可吗?"(Y/N)?"GET RK PICTURE "I"
```

```
  READ
```

```
  @2,0 SAY SPACE(80)
```

```
  ENDDO
```

```
ENDIF
```

```
ENDDO
```

```
* 选择民族:
```

```
N=0
```

```
@1,8 SAY "请选择民族编码:"GET N PICTURE "99" RANGE 1,RECNO()-1
```

```
READ
```

```
@1,40 SAY "以下操作略..."
```

```
USE
```

```
SET TALK ON
```

```
RETURN
```

①下面是用户操作时显示的一个屏幕画面:

如果没有需要的民族,请按 N 键,否则按其它键!

```
* * 民族提示 * *
```

```
1  汉族
```

```
2  回族
```

```
3  苗族
```

```
4  壮族
```

```
5  满族
```

```
6  白族
```

```
7  傣族
```

②为了同时显示几十个民族并使屏幕更加美观,可直接用下面的程序段代替上述程序中的 LIST 语句,这时屏幕菜单每行显示 5 个民族:

示 @3,8 SAY "***** 民族提示 *****"

```
DO WHILE .NOT. EOF()
    K=1
    STRING=""
    DO WHILE .NOT. EOF() .AND. (K<=5)
        STRING=STRING+STR(RECNO(),3,0)+
            ":"+"民族+"
        K=K+1
    SKIP
    ENDDO
    @ROW()+1,8 SAY STRING
ENDDO
```

③下面是程序自动生成的一个“动态菜单”，最初含 7 个民族，现在含 8 个民族。

```
***** 民族提示 *****
*****
1:汉族 2:回族 3:苗族 4:壮族 5:满族
6:白族 7:傣族 8:哈萨克
```

在 dBASE III 下实现 菜单的光带键选

泉州五中电脑室 阮自力

屏幕的菜单显示状况及选择方式如何，直接影响着用户的心理。本人编写的这一程序在 IBM PC 及其兼容机已运行通过，效果很好，这一程序具有一定的技巧性，使用起来方便灵活。为方便阅读作几点说明：①本程序既可通过→、←、↑、↓及回车键选择菜单项外，还可直接键入项编码进行选择。②程序使用的函数 PEEK(1023+PEEK(1050))能指示出用户的按键内容，这里使用的 72、75、77、80 对应 4 个箭头键，28 对应回车键。③程序安全性能良好，不会因为操作失误而可能影响系统的数据。④如果选择菜单项时，不慎按到指定键外的其它键，程序能响铃警告，并让你重新选择。

```
SET TALK OFF
A1="1. 输入"
A2="2. 修改"
A3="3. 查询"
A4="4. 统计"
A5="5. 报表输出"
A6="6. 使用说明"
A7="7. 退出"
XZ=""
DO WHIL .T.
    L=1
    CLEA
    @0,28 SAY "泉州五中学籍管理系统"
    @1,24 SAY "....."
    DO WHIL L<8
```

```
KL="A"+STR(L,1)
@L+1,28 SAY &KL
L=L+1
ENDDO
L=1
DO WHIL .T.
    KK="A"+STR(L,1)
    @L+1,28 GET &KK
    CLEA GET
    SET CONS OFF
    WAIT " " TO XZ
    H=PEEK(1023+PEEK(1050))
    DO CASE
```

```
    CASE H=28
        XZ=STR(L,1)
    CASE H=77 .OR. H=80
        @L+1,28 SAY &KK
        L=L+1
        IF L=8
            L=1
        ENDIF
        LOOP
    CASE H=75 .OR. H=72
        @L+1,28 SAY &KK
        L=L-1
        IF L=0
            L=7
        ENDIF
        LOOP
```

```
ENDCASE
SET CONS ON
DO CASE
    CASE XZ="1"
        DO SURU. PRG
    CASE XZ="2"
        DO XIUGAI. PRG
    CASE XZ="3"
        DO CAXUN. PRG
    CASE XZ="4"
        DO TONGJI. PRG
    CASE XZ="5"
        DO BAOBIAO. PRG
    CASE XZ="6"
        DO SHIYONG. PRG
    CASE XZ="7"
        CLEA ALL
        RETURN
    OTHE
    ? CHR(7)
    LOOP
ENDCASE
EXIT
ENDDO
ENDDO
```


双列分页打印程序

中国科学院化学所 陶锋

在编写程序时,常常需要把源程序打印出来,以便检查。我在用 C 语言、汇编语言等的编程中,感觉到,每个语句所占用的字节并不多,但在打印时,它要占据一行,不但浪费纸张,更重要的是打印的程序所占的纸张太长不便于检查程序。能否使被打印的程序,在每页纸上打印出两列的程序清单呢?这样可以使打印出来的程序,在纸张的长度上基本达到缩短 1 倍的目的,便于程序清单的检查和保存。

现在打印的程序不少,但没有现成的程序可利用。本人动手用 Turbo C 语言的 1.5 版本,编写了一个适合于 IBM-PC 机编程打印的程序,希望能为有兴趣者所用。程序特点:

1. 此程序可在命令行上接收多个要打印的文件名,而不产生混乱。一个程序打印完后,从另一页开始打印另一个程序。命令格式为:

dcdpp filename1 filename2...

2. 在要打印程序清单时,屏幕提示“Will the filename be printed with line number (y/n)?”,此时键入 y,则在被打印的程序清单中加入行号。

3. 本程序可在行宽 80 个字符的纸上,每页打印双列程序清单。一页打印完后,等待键入任意键,打印另一页。

程序说明:本程序可在行宽 80 个字符的纸上输出二列,每列由 54 行组成,每行最多为 34 个字符。这些参数均为宏定义,可按具体要求进行调整。程序首先定义了一个二维数组,将待打印的程序 2×54 行,读入这个数组中。若源程序行超过 34 个字符,自动截取前 34 个字符,将余下的放入下一行中,如此下去。数组第一维是行号,第二维中存放这一行的内容。

函数 dprint() 将待打印的程序按双列形式读入二维数组中。函数 nextline() 每读入一个 34 个字符的字符串时,行号自动加 1,每行内容少于 34 个字符时,则用空格补足。当其读入了 2×54 行时,调用 printout() 函数将其在打印机上输出。函数 printout() 将读入二维数组中的内容按 2 列的形式在打印机上输出(源程序清单就是用本程序打印的)。函数 ys() 将每行开头的空格或 TAB 键压缩掉。

```
/* DOUBLE COLUMN PAGING PRINTOUT */
```

```
#include "stdio.h"
#include "fcntl.h"
#include "conio.h"
#include "io.h"
#include "dos.h"
#include "bios.h"
#include "graphics.h"
#include "mem.h"
```

```
#include "stdlib.h"
#define LL 96
#define COL 2
#define CSIZE LL/COL-14
#define PL 108
#define DL 54
#define MARGIN 2
#define MAX 200
```

```
char buff[PL][CSIZE];
char *tt;
int ln[PL];
int col,l,p,page,key;
```

```
main(argc,argv)
int argc;
char *argv[];
{
    int p,qq;
    if (argc<2){
        fprintf(stdout, "\nUsage: dcdpp filename[.ext]\n");
        abort();
    }
```

```
    setmode(fileno(stdout), O_TEXT);
    for(p=1; p<argc; p++){
        ys(argv[p]);
        tt=argv[p];
        pp=wherey();
        do {
            gotoxy(18,qq);
            printf("\nWill the %s be printed with line number (y/n)?",tt);
            key=getch();
        } while (key!= 'y' && key!= 'n');
        clrscr();
        dprint();
        unlink("tmpll");
    }
```

```
/* Delete space or tab at beginning of lines */
```

```
ys(fname1)
char *fname1;
{
    char buf[MAX];
    int i,j,flag;
    FILE *fp1, *fp2, *fopen();
    if ((fp1=fopen(fname1, "r"))==NULL)
    {
        printf("\n Can't read file %s!",fname1);
```

```

abort();
if ((fp2=fopen("tmpl", "w"))==NULL)
{
printf("\n Can't write file %s!", "tmpl");
abort();
flag=1;
while(flag){
memset(buf, MAX, '\0');
if ((fgets(buf, MAX, fp1))==NULL)
flag=0;
if(flag){
j=0;
while (((buf[j])==' ')|| (buf[j]==''))&&.(j<
MAX))
j+=1;
for(i=j; buf[i]!='\0'; i+=1)
putc (buf[i], fp2);
}}
fclose(fp1);
fclose(fp2);
}

/* READ FILE AT D---D MODE */
dprint()
{
FILE *fp; *fopen();
int line,c;
extern int page;
if ((fp=fopen("tmpl", "r"))==NULL)
{
fprintf(stdout, "\n Can't open file %s! \n", "tmpl");
abort();
}
line=0; p=0; l=0;
page=1; c=getc(fp);
while(c!=EOF){
ln(l)=++line;
while(c!='\n'&&c!=EOF){
if(p>=CSIZE){
nextline();
ln[l]=++line;
buff[l][p++]=c;
c=getc(fp);
}
nextline();
if(c!=EOF)
c=getc(fp);
}
while(l!=0){
ln[l]=0;
nextline();
}
fclose(fp);
}

/* READ THE NEXT LINE */

```

```

nextline()
{ int i;
while(p<CSIZE)
buff[l][p++]=' ';
if(++l>=PL){
printout();
for (i=0; i<=(61-DL); i++)
fprintf(stdprn, "\n");
l=0;
}
p=0;
}

/* PRINT THE FILE OUT */
printout()
{ int l,i,lpos,d,p,col;
extern int page;
char line[LL];
fflush(stdprn);
gotoxy(18,5);
printf("Replace paper, then hit any key to continue!");
getch();
gotoxy(18,5);
printf("Printing now, please wait! .....");
if (key=='n'){
fprintf(stdprn, "          %10s

%5d PAGE", tt, page++);
fprintf(stdprn, "\n");
fprintf(stdprn, "-----
-----");
}
else{
fprintf(stdprn, "          %10s

%5d PAGE", tt, page++);
fprintf(stdprn, "\n");
fprintf(stdprn, "-----
-----");
}
fprintf(stdprn, "\n");
fprintf(stdprn, "\n");
for(l=0; l<DL; l++){
for(i=0; i<LL; i++){
line[i]=' '; lpos=0;
for (col=0; col<COL; col++){
if (key=='n')
p=lpos;
else{
d=ln[l+DL*col];
p=lpos+2;
while(d>0){
line[p--]=d%10+'0';
d=d/10;
}
p=lpos+4;
}
}
}
}

```



```

for(i=0;i<CSIZE;i++)
line[p++] = buff[l+DL*col][i];
if(key == 'n')
lpos += CSIZE + 3;
else
lpos += CSIZE + 6;
}
p = LL;
while((p>=0)&&(line[p-1] == ' '))
p--;
for(i=0;i<p;i++)
putc(line[i],stdprn);
putc('\n',stdprn);
}
for(l=0;l<MARGIN;l++)
fprintf(stdprn, "\n");
}

```

广州大学一号病毒 解除与免疫

华南师范大学计算机系 张文锋

一、病毒特点:

1. 感染性十分强,由于它属于引导区病毒,修改了中断 13 的向量,所以只要有关于 INT 13H 的操作就首先执行病毒源程序从而传染磁盘,即使用无系统的数据盘启动时,只要磁盘中有病毒,也会感染硬盘。

2. 采用 stone(大麻)标志(××××:0000JMP 07CD:0005)并且采用小球病毒的标志(5713)。不但如此,它还具有自己标志(地址[01BA]开始四个字节容为:A1 0D 0A 00),所以它对 stone,小球免疫。

3. 由于标志多,往往导致一些解毒工具(如 scan 等)的误操作,执行解大麻病毒的程序,结果越解越糟。

4. 占据主引导区,转移原引导区。

a. 软盘(360K):把原引导区调到 719 扇区(即最后),所以影响不大。

b. 硬盘:把主引导区调到 1 道 1 面 F 扇区。对于不同的硬盘只要分区的第一个柱不同,那么对盘破坏就不同,这是因为分区引导区的位置前后移到的位置不同,倒置目录区和数据区的首扇区有所变化,有时会占据目录区,有时会破坏 IBMBIO.COM 文件。

c. 对 1.2 M 软盘:与 360K 一样引导区调到 719 扇区。

二、解毒与免疫

1. 对软盘解毒

```

A>DEBUG
-L 100 0 0 1
-D 29D
××××××××:××××××××病毒一号
:
: 广州大学黄永钊
制造!

```

```
-L 100 0 719 1
```

```
-W 100 0 0 1
```

```
-Q
```

2. 对软盘免疫(对象为无病毒的盘)

```

A>DEBUG
-L 100 0 0 1
-E 2BA(连续加四字节标志)
2BA. A1 2BB. 0D 2BC. 0A 2BD. 00
-W 100 0 0 1
-Q

```

3. 对硬盘解毒

```

A>DEBUG
-A 100
××××:0100 XOR AX,AX
INT 13
MOV AX,0201
MOV BX,0200;读正确引导区
MOV CX,010F
MOV DX,0180
INT 13
MOV AX,0301
MOV BX,0200
MOV CX,0001;写回主引导区
MOV DX,0080
INT 13
INT 3

```

```
-G=CS:100
```

```
-Q
```

A>SYS C:(如果不能启动时才恢复 DOS 系统程序)

4. 对硬盘免疫

```

A>DEBUG
-A 100
××××:100 MOV AX,0201
MOV BX,0200
MOV CX,0001
MOV DX,0080
INT 13
INT 3

```

```
-G=CS:100
```

```
-E 3BA
```

```
3BA. A1 3BB. 0D 3BD. 0A 3BD. 00
```

```
-G=CS:100
```

```
-Q
```

(本人对许多软盘和硬盘按上步骤解毒都通过)

谈谈 2.13 汉字系统在软盘上的使用

韶关大学计算机系 廖红旭

2.13 系列汉字系统(以下简称 213)以其卓越的汉字处理功能赢得国内广大用户,同时,随着硬件性能价格比的提高,基本配置的 PC 机如 Boy 机和长城 0520-EM 机等,开始普及到中小学和家庭。

现在学校和家庭一般使用的汉字系统是电子工业部六所的 CCDOS4.0。它提供了驻留字库的功能,看上去很方便,实际上由于字库驻留不完全,很多字因为顾此失彼而找不到,给用户带来不便。如今基本配置的 PC 机内存都有 640K。驻留国标(GB)一、二级字库后仍从容运行巨大的 dBASE III,而 213 就具有驻留国标一、二级字库的功能,因此把该功能移植到软盘上是很有意义的。通常 213 是安装到硬盘上使用的(213 系统盘共有 29 片,加载高点阵字库和繁体字库时硬盘容量不得低于 8M)。下面就以 2.13F 为例,给出较完整的移植步骤,望能与同行们共同讨论。

一、系统盘的操作步骤:

1. A>FORMAT B: /S

2. A>CDEBUG HZK16

-RCX

CX EF40

:6900

-W

-NCCCC.COM

-L

-A10E

1BCB:010E INT 20

1BCB:0110

-G=100

-A100

1BCB:0100 JMP 010E

1BCB:0102 NOP

1BCB:0103

-A10E

1BCB:010E CALL 0257

1BCB:0111

-A13B

1BCB:013B JMP 013F

1BCB:013D

-E1820 55 AA

-W

3. A>COPY CONFIG.SYS B: (拷贝 213 系统文件)

A>COPY ANSI.SYS B:

A>COPY HZK16 B:

A>COPY FILE2.COM B:

A>COPY CCCC.COM B:

A>COPY CH21.COM B: (不同 CRT 选不同显示模块)

A>COPY CON B:AUTOEXEC.BAT

ECHO OFF

CLS

ECHO

PLEASE WAIT!!

FILE2

CCCC

CH21

ECHO ON

^Z

不显示版权可作下步:

4. A>CDEBUG CH21.COM

-A123

1BCB:0133 JMP 019C

1BCB:0135

-W

经过上面几步,用户就能使用 213 的驻留汉字库功能了。若想使用 213 别的小巧功能如预选字表、键盘重定义等,就必须把 213 的 YX1.COM 文件、KEY.COM 文件等拷贝到用户的工作软盘内运行之。

二、两个重要的向量入口。

向量名	入口地址	注意
-----	------	----

INT 16	CS:9A80	原 INT 16 拷贝为 INT 7E
--------	---------	---------------------

INT 10	CS:184A	原 INT 10 拷贝为 INT 78
--------	---------	---------------------

了解这两个向量的入口,分析和修改它们都是方便的。

三、进一步修改(兼谈键盘扫描码)

使用过长城机的同行都知道,它有方式 1~4,全/半角转换和英文这 6 个键。而标准的 101 键盘没有。在汉字录入的响应和方便程度方面,前者确比后者方便。能否利用标准的 101 键盘仿真长城机键盘呢?这就要先了解键盘 I/O 的基本原理。

我们知道,键盘的每一个非变换键被按下,硬件电路就会调用监控程序 INT 16(键盘 I/O 中断)产生相应的 ASCII 码和扩展 ASCII 码(又称扫描码)。键盘上除 SHIFT、CTRL、ALT、SCROLL、CAPSLOCK 等 5 个键外,每一个键都有一个对应的扫描码,但不一定有对应的 ASCII 码。组合键的 ASCII 码等于零值。通过判断 ASCII 码或扫描码可以知道哪一个键(或组合键)被按下过。这在 PCTOOLS 等工具软件及 CCBios 中的键盘管理模块中广泛使用。(附录中给了常用的组合键的扫描码)。

知道上述这些原理,修改 2.13F 的键盘管理模块 CCCC.COM,使标准 101 键盘仿真长城机键盘是容易的。当然,被重定义的功能键原功能将丢失。本文旨在提出思想方法。步骤如下:

A>CDEBUG B:CCCC.COM

-E9BC1 4F

-E9BD9 52

-E9BF4 47

-E9C00 49
-E9C0C 53
-E9C18 51

这样,101 标准键盘仿真如下:

INSERT(区位) HOME(首尾) PGUP(拼音)
DELETE(快速) END(全/半角) PGDN(英文)

附录:常用组合键扫描码表

扫描码	字符组合
54H~5DH	SHIFT+F1~F10
5DH~67H	CTRL+F1~F10
68H~71H	ALT+F1~F10
72H	CTRL+PRTSC
75H	CTRL+END
76H	CTRL+PGDN

PC/XT 机病毒报警程序

福州大学 王伟智 刘秉瀚

许多病毒常采用修改内存总量,使病毒程序常驻内存,这样,用户软件就不能复盖病毒程序。

PC 机系统内存总量参数存放在 0:0413H 单元中,并以 K 为单位,例如,640K 内存的配置,0:0413H 单元内容为 280H。对一般 XT 系统,内存配置为 640K、512K 和 256K。如果系统启动后,内存总量不等于这几种参数,系统就有可能传染上病毒。用户如能知道这个情况,就可及时采取措施,避免许多修改内存总量的病毒的传播。

根据这个思想,我们设计了一个病毒报警程序,把本程序的调用命令存放在 AUTOEXEC. BAT 文件中,系统引导结束,自动执行病毒报警程序。如发现内存总量减少,立即报警,提醒用户注意病毒将传染;如无减少,则转去执行正常操作。本程序可发现“小球”病毒、“大麻”病毒、“巴基斯坦”病毒、“磁盘杀手”和“毛虫”病毒等。实验表明,本程序对于操作系统型病毒和传染了 COMMAND.COM 的外壳病毒,并且利用修改内存总量使病毒程序隐蔽起来的情况效果很好。

本程序仅能发现病毒在活动,但不能确定病毒类型,也不能除去病毒。对于修改 INT 21H 的病毒,程序中显示报警提示的中断应改为 INT 10H。病毒对 INT 10H 服务程序的修改较少出现,这样,可避免报警时病毒进行传播的可能性。修改方法可参见 DOS 手册。

```
CODE SEGMENT
    ASSUME CS, CODE, DS, CODE, ES, CODE
STACK SEGMENT PARA STACK 'STACK'
    DB 20 DUP(0)
STACK ENDS
    ORG 100H
BEGIN: JMP START
```

```
WARN DB 'WARNING: TOTAL MEMORY RE-
      DUCED! ', 0AH, 0DH
      DB 'BEWARE OF VIRUS INFECTION!!!', '$'
```

```
START: PUSH DS
      XOR AX, AX
      MOV DS, AX
      MOV BX, 0413H      ;取 413H 单元内容
      MOV AX, [BX]       ;到 AX
      POP DS
      CMP AX, 0280H      ;内存是 640K 吗?
      JZ RET              ;正常则退出
      CMP AX, 0200H      ;内存是 512K 吗?
      JZ RET              ;正常则退出
      CMP AX, 0180H      ;内存是 384K 吗?
      JZ RET              ;正常则退出
      CMP AX, 0100H      ;内存是 256K 吗?
      JZ RET              ;正常则退出
      LEA DX, WARN       ;不正常则报警
      MOV AH, 09H
      INT 21H
      RET: INT 20H
CODE ENDS
      END BEGIN
```

考生姓名编码自动生成程序

甘肃庆阳长庆一中 任绥海

随着标准化考试的日益深化和完善,考生的姓名等汉字信息也可以由机器来阅读处理了。如《考生机读报名卡》就要求考生将汉字信息以编码的形式涂写出来。要涂写编码,首先当然应该知道编码。国家招生委员会连续两年编印出版了《汉字编码简明对照表》,供考生自己查找。遗憾的是它们都只收录了两千五百多个常见姓名用字,有相当一部分学生根本就找不到自己的姓名编码,不得不向上一级主管部门伸手求援;即使是“榜上有名”者,一个学校,几百考生,要想从那几张对照表上一个一个查出自己的姓名编码,也绝非易事。

实际上,凡了解计算机汉字操作系统的人都能看出,所谓姓名编码原不过是汉字区位码,是根据国家有关标准编制的一种没有重码现象的汉字输入码。它是各种编码中最容易掌握的、与机内码联系最紧密的一种编码,而且可以用很简单的方法获得。下面这个 BASIC 程序,就可以利用各考点现有考生姓名数据库,一次生成姓名编码对照表,并以文件的形式存放在磁盘上,供编辑软件随意编排成各种格式,多次打印,既准确无误,又省时省力。

使用本程序前,只需将数据库文件中的姓名字段

转换成标准数据格式(SDF)即可。

以下是程序及打印实例,供您参考。

```
10 KEY OFF;CLS;LOCATE 6
20 SOUND 440,8;INPUT"考生姓名文件名?",WJM$
30 OPEN WJM$ FOR INPUT AS #1
40 WJM1$=WJM$+".DAT"
50 OPEN WJM1$ FOR OUTPUT AS #2
60 PRINT STRING$(30,"-")
70 PRINT #2,STRING$(30,"-")
80 IF EOF(1) GOTO 280
90 INPUT #1,A$;A$=RIGHT$(A$,6);PRINT A$;
100 PRINT #2,A$;
110 FOR Q=1 TO 6
120 A$(Q)=MID$(A$,Q,1)
130 A(Q)=ASC(A$(Q))-160
140 A1$(Q)=STR$(A(Q));IF A(Q)<0 THEN A1$(Q)=" "
150 A1$(Q)=RIGHT$(A1$(Q),2)
160 IF LEFT$(A1$(Q),1)="-"AND RIGHT$(A1$(Q),1)<>" "THEN A1$(Q)="0"+RIGHT$(A1$(Q),1)
170 NEXT
```

```
180 PRINT STRING$(4,32);
190 PRINT #2,STRING$(4,32);
200 FOR Q=1 TO 6
210 PRINT A1$(Q);
220 PRINT #2,A1$(Q);
230 IF Q=2 OR Q=4 THEN PRINT " ";PRINT #2," ";
240 NEXT
250 PRINT;PRINT STRING$(26,"-")
260 PRINT #2,"";PRINT #2,STRING$(26,"-")
270 GOTO 80
280 CLOSE;CLS;SOUND 440,8;LOCATE 10,18;COLOR 7,2
290 PRINT"姓名编码文件已经以";WJM1$;"的名字存于盘上
300 LOCATE 12,37;PRINT"再见!";COLOR 7,0 END
```

李蕊	3278	4079	唐莉梅	4438	3282	3523
王建伟	4585	2908	4616	金志宏	2980	5430 2674
王志梅	4585	5430	3523	尹虎	5092	2702
符永宏	2391	5132	2674	邓燕	2143	4964

(上接第 25 页)

```
scanf("%s",new->name);
scanf("%s",new->city);
scanf("%s",new->zip);
new->next=last;
last=new;
} while((i++)<3);
do {printf("%s\n",last->name);
last=last->next;
} while(last);
}
```

A>exp9-9

王宜滨	天津	300071
李文华	湖北	500065
卢元来	北京	600073
林俞锡	东北	154600
林俞锡		
卢元来		
李文华		
王宜滨		

可以改变连接方向的程序,如练习 9.10 所示。

A>type exp9-10.c

```
#include <stdio.h>
```

```
#include "stdlib.h"
```

```
#include "alloc.h"
```

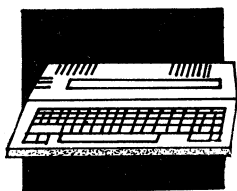
```
struct address {char name[9];
char city[7];
char zip[7];
struct address *next;
```

```
};
main()
{struct address *new,*last=0;*top=0;
int i=0;
do {if((new=malloc(sizeof(struct address)))==0)
{printf("out of memory\n");
exit(0);
}
if(top==NULL)
new->next=NULL;
scanf("%s",new->name);
scanf("%s",new->city);
scanf("%s",new->zip);
last->next=new;
last=new;
} while((i++)<3);
do {printf("%s\n",top->name);
top=top->next;
} while(top);
}
```

A>exp9-10

李一民	天津市	3000074
鲁晓春	北京市	1000009
王里黎	河北省	5000002
林太态	山东省	4000003
(null)		
李一民		
鲁晓春		
王里黎		
林太态		

(南开大学王玉华同志参加了程序调试)



把 APPLE DOS 移入 RAM 卡

学习机之友

舟山市电力公司变电工区 傅 剑

一般的 APPLE II PLUS 或中华学习机都配置有 16K RAM 卡(即 \$D000~\$FFFF RAM 辅存),在 DOS 3.3 中把该卡作为存放 INTBASIC 解释语言的地址,因此也把该卡称作语言卡。从大量的应用程序中可以看出,我们很少使用 INTBASIC 编制程序,因此语言卡经常被闲置,笔者为充分发挥现有设备的效益,特编制一程序,把 DOS 移入语言卡中。从而使 CEC BASIC 的最大编程容量高达 42KB 之多,由于妥善地处理了

DOS 与监控、CEC BASIC 解释程序的接口问题,因此当 DOS 移入语言卡后,所有的 DOS 命令和功能与未移入之前完全一样(除不能再使用语言卡)。该程序用 6502 汇编语言编写,因为需处理多种接口问题,所以内容比较长(约占 7 个扇区),在此以机器码的方式列出,有兴趣的读者可用反汇编命令将其列出分析。

程序的使用方法很简单,只要输入下列程序并存盘后,用 BRUN“程序名”命令即可。

4000-08 78 A0 04 B9 00 00 48	4148-9D 41 90 07 A5 02 DD 9C	4290-A9 E5 85 01 20 C7 42 A9
4008-88 10 F9 A9 00 8D 65 43	4150-41 B0 03 4C CB 40 8A 18	4298-A0 85 00 A9 E5 85 01 20
4010-AD D2 03 C9 9D F0 19 EE	4158-69 04 AA E0 1C B0 03 4C	42A0-C7 42 A9 E6 85 00 A9 F6
4018-65 43 C9 AF B0 09 A9 8C	4160-BF 40 AD 83 C0 AD 83 C0	42A8-85 01 20 BB 42 A9 7D 85
4020-85 00 A9 43 4C 4A 43 A9	4168-A2 23 A9 BF 85 03 A9 FF	42B0-00 A9 F7 85 01 20 BB 42
4028-FC 85 00 A9 43 4C 4A 43	4170-85 01 A0 00 84 02 84 00	42B8-4C D8 42 A0 02 B1 00 D9
4030-AD 83 C0 AD 83 C0 AD 00	4178-B1 02 91 00 C8 D0 F9 CA	42C0-D2 42 D0 0D 88 10 F6 A0
4038-E0 49 A5 8D 00 E0 CD 00	4180-F0 55 C6 03 C6 01 D0 F0	42C8-02 B9 D5 42 91 00 88 10
4040-E0 8D 81 C0 8D 81 C0 F0	4188-00 9D 56 64 9D 02 6C 9D	42D0-F8 60 AD 00 E0 20 E2 BE
4048-0C EE 65 43 A9 C8 85 00	4190-04 C3 AA 3A EE B7 02 00	42D8-A2 1E BD 32 44 9D 51 DE
4050-A9 43 4C 4A 43 A9 4C 8D	4198-00 00 78 9D 84 A8 FD AA	42E0-CA 10 F7 A2 05 BD 2C 44
4058-68 A5 A9 04 8D B3 AE A9	41A0-97 B3 00 B6 E0 B7 00 B8	42E8-9D FA FF CA 10 F7 A2 0C
4060-FB 8D CD AE A9 AF 8D CE	41A8-11 BA 69 BA 96 BA 56 BC	42F0-BD 51 44 9D BB F5 CA 10
4068-AE A9 0B 8D F9 AE 20 D1	41B0-A8 BF C8 BF 00 C0 D6 D7	42F8-F7 A9 4C 8D C6 E0 A9 BB
4070-9E A9 00 8D B3 AA A9 88	41B8-E5 ED F9 FD FD FD FD FE	4300-8D C7 E0 A9 F5 8D C8 E0
4078-85 00 A9 41 85 01 A2 00	41C0-FE 65 D2 1B 24 48 0C 8E	4308-A2 0F BD 5E 44 9D E6 F7
4080-20 A1 40 E6 02 D0 02 E6	41C8-DA ED 8B 95 27 33 39 3F	4310-CA 10 F7 A9 4C 8D 36 E8
4088-03 18 A1 02 69 40 81 02	41D0-45 4B 51 57 5D 63 69 A2	4318-8D 94 E3 A9 E6 8D 37 E8
4090-E6 02 D0 02 E6 03 88 88	41D8-AF BD BF 43 9D 00 BE E8	4320-A9 F7 8D 38 E8 A9 EE 8D
4098-D0 E9 20 A1 40 D0 E4 F0	41E0-D0 F7 BD BF 44 9D 00 BF	4328-95 E3 A9 BE 8D 96 E3 A9
40A0-1C A1 00 85 02 20 B6 40	41E8-E8 D0 F7 A2 11 BD E8 F7	4330-00 8D 5F EA A9 06 8D 42
40A8-A1 00 85 03 20 B6 40 A1	41F0-9D D5 BF BD BB F5 9D E6	4338-DE A0 1D B9 6E 43 99 75
40B0-00 20 B6 40 A8 60 E6 00	41F8-BF CA 10 F1 A9 D5 8D C1	4340-EA 88 10 F7 A9 FC 85 00
40B8-D0 02 E6 01 60 A2 00 86	4200-EA 8D E4 F7 A9 BF 8D C2	4348-A9 43 85 01 AD 82 C0 A0
40C0-04 BD 9A 41 85 02 BD 9B	4208-EA 8D E5 F7 A9 E6 8D 0E	4350-00 B1 00 F0 06 20 F0 FD
40C8-41 85 03 A2 00 A1 02 20	4210-DD A9 BF 8D 0F DD A9 6F	4358-C8 D0 F6 A0 04 68 99 00
40D0-8E F8 A4 2F C0 02 D0 60	4218-8D 02 DD A9 BF 8D 03 DD	4360-00 88 10 F9 A9 00 F0 02
40D8-B1 02 C9 9D 90 5A C9 C0	4220-A9 78 8D 04 DD A9 BF 8D	4368-28 60 28 4C 06 BF C8 C5
40E0-90 22 C9 D0 90 52 A2 0B	4228-05 DD A9 87 8D BB DE A9	4370-CC CC CF A0 A0 A0 A0 A0
40E8-B1 02 DD B6 41 D0 0E 88	4230-BF 8D BC DE A9 81 8D C6	4378-A0 A0 A0 A0 A0 A0 A0 A0
40F0-B1 02 DD C1 41 D0 06 BD	4238-DF A9 BF 8D C7 DF A9 4C	4380-A0 A0 A0 A0 A0 A0 A0 A0
40F8-CC 41 4C 28 41 A0 02 CA	4240-8D BA DE 8D C5 DF A9 C4	4388-A0 A0 A0 A0 8D 8D C3 C1
4100-10 E6 30 34 88 C9 B7 D0	4248-8D A4 DD 8D B7 DD A9 BF	4390-CE A7 D4 A0 C5 D8 C5 C3
4108-0F B1 02 C9 E8 90 20 C9	4250-8D A5 DD 8D B8 DD A2 0B	4398-D5 D4 C5 A0 A2 C4 CF D3
4110-F9 B0 1C 38 E9 13 D0 10	4258-BD 20 44 9D 56 DD CA 10	43A0-AD D5 D0 A2 AE 8D 87 C4
4118-C9 B5 D0 13 B1 02 C9 A4	4260-F7 A2 12 BD ED FF 9D B2	43A8-CF D3 A0 CE CF D4 A0 C1
4120-90 0D C9 D1 B0 09 69 2B	4268-E5 CA 10 F7 A9 B2 8D 78	43B0-D4 A0 CE CF D2 CD C1 CC
4128-91 02 C8 A9 BF D0 07 A0	4270-F3 A9 E5 8D 79 F3 A9 94	43B8-A0 B4 B8 CB A0 CC CF C3
4130-02 B1 02 18 69 40 91 02	4278-85 00 A9 DD 85 01 20 C7	43C0-C1 D4 C9 CF CE AE 8D 00
4138-38 A5 2F 65 02 85 02 A9	4280-42 A9 CD 85 00 A9 DD 85	43C8-8D 8D C3 C1 CE A7 D4 A0
4140-00 65 03 85 03 A6 04 DD	4288-01 20 C7 42 A9 7C 85 00	43D0-C5 D8 C5 C3 D5 D4 C5 A0

43D8-A2 C4 CF D3 AD D5 D0 AE
 43E0-A2 8D 87 CE CF A0 CC C1
 43E8-CE C7 D5 C1 C7 C5 A0 C3
 43F0-C1 D2 C4 A0 C6 CF D5 CE
 43F8-C4 AE 8D 00 8D 8D A0 A0
 4400-A0 A0 A0 A0 A0 A0 A0 C4
 4408-CF D3 A0 CE CF D7 A0 C9
 4410-CE A0 C8 C9 C7 C8 A0 CD
 4418-C5 CD CF D2 D9 AE 8D 00
 4420-9F BF AA BF AD BF B3 BF
 4428-B9 BF BF BF 8D BF 93 BF
 4430-99 BF 4C 00 BF 4C 06 BF
 4438-4C 0C BF 4C 15 BF AD D4
 4440-BF AC D3 BF 60 AD D2 BF
 4448-AC D1 BF 60 4C 1E BF C5
 4450-BF 8A F0 07 C9 06 B0 03
 4458-4C D1 E0 4C C9 E0 68 68
 4460-AD FE BF 48 AD FF BF 48
 4468-AD B6 EA 4C 39 E8 08 2C
 4470-83 C0 2C 83 C0 28 60 8D
 4478-FA BF 08 68 8D FB BF 68

4480-8D FC BF 68 8D FD BF A9
 4488-BE 48 A9 AE 48 AD FD BF
 4490-48 AD FC BF 48 AD FB BF
 4498-48 FD FA BF 28 8D 82 C0
 44A0-60 20 B8 BE CD 00 E0 08
 44A8-AD 00 E0 28 60 AD 72 EA
 44B0-8D FE BE AD 73 EA 8D FF
 44B8-BE 20 B8 BE 4C 00 00 20
 44C0-AF BE 4C BF DD 20 AF BE
 44C8-4C 84 DD 20 AF BE 20 FD
 44D0-EA 4C DE BE 20 AF BE 20
 44D8-B5 F7 4C DE BE 20 AF BE
 44E0-20 51 E8 4C DE BE 8D 82
 44E8-C0 20 65 D6 20 AF BE 4C
 44F0-FF E4 8D 82 C0 4C D2 D7
 44F8-20 B8 BE 4C 1B E5 20 B8
 4500-BE 4C 24 ED 20 B8 BE 4C
 4508-48 F9 20 B8 BE 4C 0C FD
 4510-20 B8 BE 4C 8E FD 20 B8
 4518-BE 4C DA FD 20 B8 BE 4C

4520-ED FD 20 B8 BE 4C 8B FE
 4528-20 B8 BE 4C 95 FE 20 AF
 4530-BE 20 81 DE 4C DE BE 20
 4538-AF BE 20 BD DE 4C DE BE
 4540-20 B8 BE 6C 36 00 20 B8
 4548-BE 6C 38 00 20 B8 BE 6C
 4550-FA FF 8D 82 C0 6C FC FF
 4558-20 B8 BE 6C FE FF AD B6
 4560-EA D0 03 8D 82 C0 6C C5
 4568-BF 6C C7 BF 8D 82 C0 6C
 4570-C9 BF 8D 82 C0 6C CB BF
 4578-8D 82 C0 6C CD BF 8D 82
 4580-C0 6C CF BF 00 00 00 00
 4588-00 00 00 00 00 00 00 00
 4590-D5 BF E6 BF 01 00 00 00
 4598-00 00 00 00 00 00 00 00
 45A0-00 00 00 00 00 00 00 00
 45A8-00 00 00 00 00 00 00 00
 45B0-00 00 00 00 00 00 00 00
 45B8-00 00 00 00 00 AF BE FF

将程序 CHAIN 升级为 APPLE DOS 命令

保定地区商业学校 赵江龙 孙杰亮

CHAIN 用来实现 BASIC 程序的链接与数据的传递,弥补了 APPLESOFT 的一大不足。但当初 APPLE 公司在发表它时,并未将其开发成 DOS 命令,而只作为一个 B 类文件保留在系统主盘上,供用户在需要时调用。目前,对 CHAIN 的研究性文章已为数不少,许多作者均从各自的角度提出了不同的改进方案,但他们大都着眼于对 CHAIN 功能的拓展上。这种仅将该功能程序当作一般的应用程序来调用的处理方式,影响了 CHAIN,包括改进后的 CHAIN 的实用价值。

笔者在对系统进行了认真分析之后,成功地将该程序升级为新的 DOS 命令。此后,APPLESOFT BASIC 程序的链接将和别的 DOS 功能一样,用标准的命令格式,以预定命令激活的方式来实现。这对于在应用程序开发过程中需要频繁传递数据的用户,具有实际价值;对于一般的用户,由于采取了一定的技术处理,用户可用的空间并不见减少,而且还保证了最大限度的兼容。为了同用于整数 BASIC 的 CHAIN 命令相区别,该命令取名为“SCHAIN”。

为了对 DOS 进行有效的扩充而又不影响 DOS 原有的全部功能,我们模仿 DOS 对命令的处理方式,在 DOS 和 BASIC 之间增加一个新的拦截判断层次,专门截获处理 SCHAIN。由于本问题把该命令约定为 DOS 命令, SCHAIN 一定属于 DOS 的拦截判断之列,因而仅对 DOS 传给 BASIC 的非 DOS 命令进行拦截即可。

考虑到 DOS 3.3 的特殊性,在程序方式下,

SCHAIN 将不为 DOS 所接受。原因是在该执行方式下,系统规定以“CTRL-D”字符打头的输出内容只应是 DOS 命令,不属于 DOS 的则认为是错误命令。非 DOS 命令时就直接由 DOS 提示出:“SYNTAX ERROR IN XX”的错误提示信息。因此,加入的拦截处理程序将接收不到该命令。本文对此进行了调整,将 DOS 的这种错误判定功能全部交给了 BASIC,该调整见步骤 6。这样,对于任何一个命令,属于 DOS 就转去执行 DOS 命令,是 SCHAIN 则进行程序链接,是错误命令的话,将由 BASIC 解释程序最终判定为错。

在运行空间的选择上, SCHAIN 放在 \$BA69~\$BC55 的 DOS 区域。原 CHAIN 占用的是 \$208~\$3B5,由于第二页同作键缓冲区, CHAIN 极易受到干扰,以致在链接过程中即遭到破坏;而 DOS 的两个资料缓冲区却被固定在 DOS 内部,当作系统的一部分享受保护。系统的这种安排是不合理的。我们参考了姚政曙“从 DOS 中获取用户空间”一文(《苹果园》87 年第 3 期)提出的方法,将资料缓冲区连同读翻译表一起,与 CHAIN 原占有的空间进行了对调。具体的操作方法在本文后面的部分列出。

升级步骤如下:

1. 引导标准 DOS3.3, CALL-151 进入监控;
2. 在监控状态下键入:

B811:2 NB880:2 NB890:2 NB8CB:2 NB921:2
 NBEC1:2 NBEC7:2 NB80A:3 NB80E:3 NB820:3 N

B825;3 NB83A;3 NB868;3 N B8CE;3 NB8D2;3
NB910;3 NB86B;2 NB90A;C0 2 NB91B;C0 2
NB92B;C0 2 NB64A;4C B3 08

3. 键入:

B6B3;A2 0 BD 6C 3 9D 56 3 E8 E0 6A
D0 F5 A6 2B 6C FD 8 0

4. 执行

356<BA96.BAFFM<CR>

键入程序 1;

5. 键入:

A008;4C 69 BA <CR>

将 DOS 的转移指针指向 SCHAIN;

6. 去掉 DOS 的错误识别功能, 键入:

A000;EA EA EA EA EA EA EA EA <CR>

7. 插入新软盘, 执行 INIT 命令。

程序一:

BA69- A0 FF A2 00 AD 00 02
BA70- C9 84 D0 01 E8 C8 BD 00
BA78- 02 E8 C9 A0 F0 F8 D9 8B
BA80- BA F0 F2 B9 8B BA F0 0A
BA88- 4C A4 9F D3 C3 C8 C1 C9
BA90- CE 00 20 12 BC 20 31 BC
BA98- 20 84 E4 A9 07 85 8F A5
BAA0- 69 A6 6A 85 9D 86 9E E4
BAA8- 6C D0 04 C5 6B F0 05 20
BAB0- 0B BB F0 F3 85 9F 86 A0
BAB8- A9 03 85 8F A5 9F A6 A0
BAC0- E4 6E D0 07 C5 6D D0 03
BAC8- 4C 6D BB 85 9D 86 9E A0
BAD0- 00 B1 9D AA C8 B1 9D 08
BAD8- C8 B1 9D 65 9F 85 9F C8
BAE0- B1 9D 65 A0 85 A0 28 10
BAE8- D3 8A 30 D0 C8 B1 9D A0
BAF0- 00 0A 69 05 65 9D 85 9D
BAF8- 90 02 E6 9E A6 9E E4 A0

BB00- D0 04 C5 9F F0 BA 20 15
BB08- BB F0 F3 B1 9D 30 46 C8
BB10- B1 9D 10 41 C8 B1 9D F0
BB18- 3C C8 B1 9D AA C8 B1 9D
BB20- 85 9C 86 9B C5 B0 F0 02
BB28- B0 2B 88 88 B1 9D 48 38
BB30- A5 6F 85 94 F1 9D C8 91
BB38- 9D 85 6F C8 A5 70 85 95
BB40- E9 00 91 9D 85 70 68 18
BB48- 65 9B 85 96 A5 9C 69 00
BB50- 85 97 20 9A D3 A5 8F 18
BB58- 65 9D 85 9D 90 02 E6 9E
BB60- A5 9D A6 9E A0 00 60 C4
BB68- C1 CF CC 84 8D A9 00 85
BB70- 95 85 9C 85 97 A9 69 85
BB78- 9B A9 71 85 96 A9 EF 85
BB80- 94 20 9A D3 A2 01 B5 69
BB88- 95 9B B5 6D 95 96 B5 6F
BB90- 95 94 CA F0 F1 20 9A D3
BB98- A5 94 85 A1 A6 95 E8 86

BBA0- A2 A0 06 B9 66 BB 20 5C
BBA8- DB 88 D0 F7 C8 B1 B8 F0
BBB0- 0A C9 8D F0 06 20 5C DB
BBB8- C8 D0 F2 A9 C9 85 36 A9
BBC0- BB 85 37 20 51 A8 4C FB
BBC8- DA 20 93 FE 20 51 A8 20
BBD0- 65 D6 38 A5 69 E5 E7 85
BBD8- 9D A5 6A E5 E8 85 9E A2
BBE0- E9 20 03 BC E8 20 03 BC
BBE8- A2 01 B5 ED 95 6F 95 3E
BBF0- B5 A1 95 3C B5 69 95 42
BBF8- CA F0 EF A0 00 20 2C FE
BC00- 4C D2 D7 18 B5 00 65 9D
BC08- 95 82 E8 B5 00 65 9E 95
BC10- 82 60 CA CA 86 B8 A9 02
BC18- 85 B9 AD 59 AA 8D 38 BC
BC20- BA 8E 59 AA A9 95 48 A9
BC28- BA 48 20 5B A7 20 83 9F
BC30- 60 AD 38 BC 8D 59 AA 60

(上接第 22 页)

BASIC(BAS)、二进制(BIN)、文本(TXT)和变量(VAR)等文件分别经 SAVE、BSAVE、OPEN 或 STORE 命令自动地产生。行号 110 至行号 190 建立一个菜单, 将文件类型显示给用户。行号 200 至行号 290 用户从 1~8 之间选择文件类型。行号 300 和行号 305 是要求用户输入路径名或文件名。行号 320 用 CREATE 命令产生文件名和文件类型。CREATE 命令必须用“,T 类型”参数产生所需要的文件类型。只有目录文件的产生

不需要 T 参数。

如果用户选择了用户定义文件类型, 那么程序由行号 310 转至行 350, 允许用户选择所需的文件号(从 1 至 8)。行号 380 将两个串变量相加, 而后返回到行号 320 产生文件及文件类型。

ProDOS 系统程序(SYS)、浮动代码(REL)和用户定义(\$F#)类型文件可以用 BASIC 产生。

〈未完待续〉

北京安华科技市场西城经营部

——主要经营: 电子计算机、电子元器件

各种微机:

• 中华学习机: CEC—I ¥900.00 小蜜蜂—I
¥920.00

• 家庭 PC 机: ¥3150.00

主频: 4.77/12 兆 RAM: 640KB 软驱: 360KB×2
101 键盘

14"双频单显

Smper286: ¥6950.00

主频: 10/21 兆 RAM: 1MB 可扩 5MB 5 1/4 软驱: 1.2MB
360KB 各 1

硬盘 3.5"40MB. 101 键盘 14"双频单显 数显机箱

选配: 14"CGA 彩显加 1500 元 14"1024×768 VGA 彩显加
3800 元

各种: APPLE II、CEC—I、PC 机接口板; 九针、24 针打印
机

还承接各种微机的修理业务。

对教育部门、个人购机实行优惠。

地址: 北京西北大街 120 号 电话: 65.6342

邮码: 100034

联系人: 孙义强 宁函华 BB 机: 126 呼 45198 42616

向天坛机移植中华机汉字软件的一些体会与经验

张 斌

天坛机及其加强型问世以来,便以与中华机汉字90%以上的兼容度而受到好评。但是,也有一部分中华机的汉字软件在天坛机上不能直接运行。我曾经试着移植了一批汉字软件,包括‘人事档案’、‘个人办公系统’、‘家庭账目管理’、‘备忘录’等,现在,我就把在移植过程中出现的问题和几点经验做一简单的说明。其中,出现的问题大致可分为以下几点。

(1) 汉字状态下图象的移植。

这是中华机汉字软件移植中的一个关键性问题。中华机的汉字屏幕是显示在高分辨率第二页上的,即内存地址 4000H~5FFFH。而天坛机的汉字屏幕占用的是高分第四页,即内存地址 8000H~9FFFH。故中华机上图文并茂的软件到天坛上便只见文不见图了。如果图形是以 34 扇区 B 类文件的形式存在盘上的,那么在读文件时只需把起始地址从 4000H 改到 8000H,图形就会显示出来了。但有一些较高级软件中的图形是以压缩形式存放的(如通讯录),在它运行时先调入一个解压缩文件和压缩的图形文件,然后再把图形还原到高分第二页上去。对此解决的方法有两种,一是先把还原好的图形存盘,然后修改主程序,直接把图形调入高分第四页。这种方法的优点是简单,省事,缺点是占用磁盘空间较大,调盘时速度较慢。另一种方法是修改解压缩文件,使它把压缩的图形直接还原到高分第四页。这就要求移植者对机器的内部结构、6502 语言以及一些常用的系统子程序有个大概的了解才行。这种方法的优点是使用时速度较快,对主程序的改动不大,缺点是移植的周期较长。

还有一些程序,如‘个人办公系统’是先把图形调入高分辨率第一页做些处理,再通过调用机器码子程序把图形移入高分第二页。对于此类程序的修改主要是针对机器码子程序的修改,即把图形移动的目的地从高分第二页改到高分第四页就成了。

(2) 汉字状态下特殊字符的移植

在中华机的硬汉字库中包含了一些特殊的图形字符,如‘①’、‘②’、‘Σ’等等。而在天坛机中所对应的不是空字符就是“怪”字符。比如中华机中的‘①’对应于天坛机的就是大写的‘X’;‘:’对应是大写的‘B’等等。这就是为什么有些中华机的汉字软件在天坛机上可以运行,但在中文中间总是加杂有“怪”字符的原因。对于这类问题解决的方法一般有两种:一是硬替换,一种是软替换。硬替换法就是直接用天坛机中的字符替换掉。比如用‘:’、‘山’两个字符替换掉‘B’;用两个减号替换掉‘!’等等。这种方法的特点是快速、简便,适用于画面要求不高,特殊字符不太多的场合。缺点是显示效果不如软替换法。而软替换法就是把所有的中华机图形字符

制成型表,然后用造型表替换掉那些“怪”字符。我在“小学数学”软件移植过程中,考虑到原来的显示比较生动活泼,适合于小学生使用,如使用硬替换法就会影响画面的质量,因而采用了软替换方法。这种方法的优点是显示灵活,方便,画面质量较高,效果好。自己甚至可以造出中华机没有的字符。缺点是大量显示造型表时会影响速度,移植难度大,周期长。

(3) 汉字状态下与 DOS 的配合问题

在中华机上,汉字系统并不侵占 DOS 区,因而一些经过改进的快速 DOS 可以在汉字系统下正常运行。而在天坛机中的汉字系统侵占了 9600H~9FFFH 这一部分的 DOS 区作为汉字显示页,这样使得一些快速 DOS 无法正常运行。经我实验在汉字状态下真正能正常运行的 DOS 只有标准的 DOS3.3。所以,如果你的汉字软件不能正确的打开和读写文件时,请换一个标准 DOS3.3 试试看。

(4) 汉字状态下英文字符大小写问题

在我移植的“小学英语”软件中,最突出的问题就是在汉字状态下英文大小写字符的问题。在英文中,一般在句子开头的第一个字母必须大写,在人名、地名、缩写和其它一些专有名词的第一个字母也必须大写。但在天坛机的汉字状态下,英文字符没有小写状态。如果把试题和答案全部改成大写字符,不仅工作量成倍增加,而且会使一部分考题失去意义。为了解决这一矛盾,我采用了特殊字符移植问题中的软替换法。即把二十六个小写字符全部都做成造型表。然后在应显示小写字符的位置上用造型表替换。

关于移植过程中的问题就介绍这些,其中难免有失误之处,欢迎大家批评指正。下面,我将在移植中积累的一些小经验介绍一下。

(1) 去掉原程序中所有 CALL 43089 语句。

(2) 在 PR # 3 语句后面必须紧跟一条 PRINT 语句。

(3) 在 MUSIC 语句后必须跟一个 PR # 3 及 PRINT 语句。

(4) 有些程序中 DOS 命令不能正确执行,就在这条命令前加一条空 DOS 命令,即 PRINT CHR \$(4)

(5) 在汉字状态下 GET 语句不能读进控制字符,用下面子程序模拟:

```
9998 POKE $C010,0;A=PEEK($C000);A$=CHR$(A);IF A<128 THEN 9998
9999 RETURN
```

(6) 汉字状态下有时要知道光标所处位置,下面语句把横座标放入 H,纵座标放入 V:

```
H=PEEK(214);V=PEEK(215)
```


FORTH 问答:堆栈和逆波兰表示法

北京机工印刷厂动力设备科 丁志伟

甲:你上回曾说 FORTH 是面向堆栈的语言,就是汇编语言常用的后进先出栈吗?

乙:正是。不过对于 FORTH,堆栈的作用相当重要,使用起来内容也更丰富。

甲:逆波兰是怎么回事?和堆栈是什么关系?

乙:逆波兰是一种运算表示法,在算术逻辑运算时与堆栈互为表里。解释起来有些麻烦,这样吧,我问你,用 BASIC 做 $3+5$,再显示出来,你怎么做呢?

甲:可以用 $A=3+5$;PRINT A。FORTH 是怎样的呢?

乙:用 $3\ 5+.$ 显示 8。(✓表示回车)

甲:看起来有些新鲜,怎么不是 $3+5$?变量是怎么表示的?数又存到哪里?还有……

乙:你看,逆波兰是有不少特点,对这么一个简单的算式,能让你提出这么多问题。我一个个回答。这种表示法,一般不用变量,而把数据存在堆栈里。

甲:在这个例子里,怎么用到堆栈的?

乙:FORTH 每输入一个数都要放到栈顶,取出时也从栈顶取。这里先输入的是 3,存入栈顶,见图 1a。再输入 5,栈顶就变成 5,3 并不消失,而在次栈顶,见图 1b。

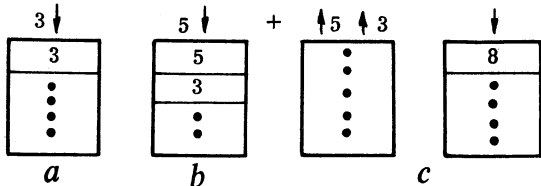


图 1

甲:遇到“+”呢?也入栈吗?

乙:不,栈中只能存放数据,“+”则是从栈里取出两个数,相加后把两数之和再送入堆栈。此时栈顶是 8,见图 1c。3 和 5 都已不复存在了。

甲:噢,这就是逆波兰。这个名字有点奇怪,你能告诉我它是什么意思吗?

乙:这种表示法是波兰逻辑学家 J. 卡西维兹发明的,因而得名。实际上有两种,一是波兰表示法,一是逆波兰法。我们平常所用的运算表示法,比如 $A+B$ 两数相加,都是把运算符“+”放在中间,写做 $A+B$,称为中缀表示法,而波兰和逆波兰则是分别把运算符放在前边和后边,表达式写出来就是 $+AB$ 和 $AB+$,又称为前缀法和后缀法。FORTH 用的是后缀法,就象你看到的 $3\ 5+$,请注意 3 和 5 之间要有空格,否则有可能误认为 35。其他高级语言的编译系统常把它做为中间语言表示法。

甲:把运算符放在后边有什么好处吗?

乙:FORTH 中,数据存放于堆栈,后缀法是与之相应的表示法。它不用括号,运算的先后顺序由表达式决定,这样系统处理起来简单,因此就能提高速度。

甲:不用括号?比方 $(a-b)*c$ 用逆波兰怎么表示呢?

乙: $a\ b-c\ *$ 。先做 a 减 b ,其结果乘以 c 。首先, a 和 b 先入栈,见图 2a,遇到“-”时,得出 $a-b$,见图 2b。随后 c 再入栈,见图 2c,遇到“*”时, $(a-b)$ 与 c 相乘,得出 $(a-b)*c$,见图 2d。

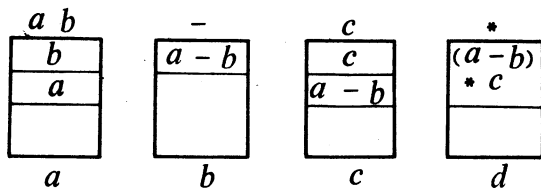


图 2

甲: $(a-b)$ 与 c 相乘,这里的汉语表达法,正好可以做后缀表达法。

乙:你真细心。这样理解起来就方便多了。

甲:虽说如此,还是不太习惯。

乙:刚开始这是难免的,时间一长就能适应了。

甲:堆栈内数据变化也不易弄清,有好办法吗?

乙:这一般是用在括号内加注释的办法来解决。 $a\ b-c\ *$ 就可以这样注释:

$a(--a)\ b(a--ab)\ -(ab--a-b)$
 $c(a-b--a-bc)\ *(a-bc--a-b*c)$

括号在 FORTH 中是注释用的,不会被执行。括号内,虚线左边是执行前栈状态,右边是执行后栈状态。将来熟练了,就不必注释得这么细了。

甲:这下清楚多了。前边的 $3\ 5+.$ 中的“.”相当于 BASIC 中的 PRINT 吗?

乙:差不多,它从栈中取出一个数据,显示在屏幕上,栈状态是 $(n--)$ 。这里 n 代表 16 位有符号数,这是 FORTH 中常用的数据类型。

甲:这样执行“.”后栈中数就消失了,如果还想用又怎么办呢?

乙:那就先用 DUP,意思是复制栈顶数据,栈状态是 $(n--nn)$ 。把这个式子改成 $3\ 5+DUP.$,输出后栈中还留有一个 8。由于堆栈的重要性,FORTH 安排了一组对堆栈进行操作的动词,DUP 就是其中之一。其余的列表如下:

SWAP($n_1n_2--n_2n_1$) 栈顶和次栈顶两数对换
 DROP($n--$) 丢掉栈顶一个数
 OVER($n_1n_2--n_1n_2n_1$) 复制次栈顶
 ROT($n_1n_2n_3--n_2n_3n_1$) 旋转,把栈顶第三项旋转到栈顶

甲:能举几个例子,说明它们的用法吗?

乙:行。你输入 1 2 3,再... ,该显什么?

甲:堆栈是后进行先出,该显 3 2 1。想按原来的顺序显出 1 2 3,看样子上边的几个动词该派上用场了。

乙:此时用 ROT. SWAP. 或 SWAP ROT... 都能显出 1 2 3。我给你出个题,现在栈顶状态是(-abc),想要做 (a+b)/c,用逆波兰该怎么做呢?

甲:我看看,先把 a 和 b 移到最上边,该用 ROT ROT+,栈顶就是 (c a+b)。现在该做除法了吧。

乙:还不行。除法要求被除数放在次栈顶。

甲:那就先用 SWAP。总起来就是 ROT ROT+SWAP /。

乙:完全正确,不防再给它加上注释,继续熟悉一下。

甲:DROP 有用吗?

乙:很有用。为了说明,我先介绍两个动词:

/MOD($n_1n_2--n_3n_4$) n_1 除以 n_2, n_3 :余数, n_4 :商

MOD ($n_1n_2--n_3$) n_1 除以 n_2, n_3 :余数

我所用的版本里,没有 MOD,在只需要余数时,就用得着 DROP 了。比方 10 3 /MOD DROP. 显 1。

甲:用 /MOD DROP 代替 MOD,是否有些麻烦?

乙:如果嫌麻烦,可以定义一个 MOD。方法是,
;MOD /MOD DROP;

甲:你最好仔细讲解一下定义是怎么回事。

乙:因为篇幅关系,这回不行了,下次再说吧。

中华学习机图书管理程序

甘肃西和县西路乡王磨村 王二珠

不少人手头都有几十本书籍,对它们的管理是件麻烦事情。但若用本文程序管理,就方便多了。

特点:采用模块结构,使用方便。而且在检索时,除了采用多种检索方法外,不论用那种方法检索,都会列出书的开本,这就对寻找这本书带来了方便。最后还显出所检索的书有多少本和它们的总价值。

使用时应注意:

- (1)所用盘上应有 CHAIN 程序。可用 FID 拷贝。
- (2)首次使用应先建立书库,以后再也不用建立了。
- (3)图书库文件名已取了 TSK. L100,也可改为用输入语句输入。
- (4)如把主控程序 TSGL 作为“哈罗”程序存在盘上,那使用就更方便了。
- (5)除了 TSGL 程序外,其它程序都不能单独起动。

主控程序(TSGL)

```
10 D$=CHR$(4);PRINT D$;"MAXFILES 1"
20 PRINT D$"PR #3";PRINT:HOME
30 TT$="*****";
PRINT TT$;PRINT TAB(1);" ";TAB(14);"图书管理";
TAB(33);" ";PRINT TT$
40 PRINT " 功能表: ";PRINT "1. 建立书库 2. 图书检索
3. 图书扩充 4. 修改图书 5. 删除图书 6. 退出系统"
50 INPUT "你用哪一个(1-6)?" ;Y$;A=VAL(Y$);IF A
<=0 OR A>6 THEN 50
60 IF A=6 THEN HOME;VTAB6;PRINT TAB(15);"再
见!";CHR$(19);NEW
70 O$=D$+"OPEN TSK. L100, L100";C$=D$+"
CLOSE";R$=D$+"READ TSK. L100, R";W$=D$+"
WRITE TSK. L100, R";IF A=1 THEN 110
80 PRINT O$;PRINT R$;0;INPUT N,F$,Z$,L$,B
$,T,L,S;PRINT C$;P=L;PRINT "共有记录";P;"个"。
90 PRINT D$;"BLOAD CHAIN, A520"
100 PRINT;ON A-1 GOTO 150,160,170,170
110 INPUT "是第一次使用吗? (Y/N)";Y$;IF Y$="Y"
```

THEN 130

```
120 PRINT "请选用 '扩充图书' 项!";CHR$(19);GOTO 20
130 INPUT "请输入现在日期(年. 月.):";T;E=0;N$="书
名";Z$="作者";L$="类别";B$="出版社";L%=0;S
=0
140 PRINT O$;PRINT W$;0;PRINT E;" ";N$;" ";Z
$;" ";L$;" ";B$;" ";T;" ";L%;" ";S;PRINT C$;
PRINT "完成!";CHR$(19);GOTO 20
150 CALL 520"图书检索"
160 CALL 520"增加图书"
170 CALL 520"修删图书"
```

图书检索:

```
10 PRINT D$;"PR #3";PRINT;ONERR GOTO 190
20 HOME;INPUT "是否检索?";Y$;IF Y$="N" THEN
190
30 DIM N$(P),V$(P),E(P),L$(P);HOME;PRINT TT
$:PRINT TAB(1);" ";TAB(14);"图书检索";TAB(33);
" ";PRINT TT$
40 PRINT " 功能表: ";PRINT "1. 全部 2. 编号 3. 类
别 4. 出版社";PRINT "5. 书名 6. 作者 7. 单价 8. 时间 9. 退出"
50 INPUT " 你选哪一个? (1-9)";Y$;ZJ=0;C1=
0;A=VAL(Y$);IF A<=0 OR A>9 THEN 50
60 ON A GOSUB 80,90,100,160,140,150,170,180,190
70 GOTO 20
80 N1=1;N2=P;GOSUB 200;RETURN
90 INPUT "起始号: ";N1;INPUT "结束号: ";N2;GOSUB
200;RETURN
100 HOME;FOR I=1 TO 14;READ H$;PRINT I;" ";H$;
" ";NEXT;PRINT;RESTORE
110 DATA 语文,数学,物理,地理,化学,生物,历史,英语,
政治,科普,技术,电子,棋术,计算机
120 INPUT " 你选哪一个?(1-14)";Y;IF Y<1 OR Y
>14 THEN 120
130 FOR I=1 TO Y;READ H$;NEXT;RESTORE;E$=
" ";I$=" ";G$=" ";GOSUB 230;RETURN
```

```

140 INPUT "请输入书名: "; E$; G$ = " "; I$ = " "; H$ =
    " "; GOSUB 230; RETURN
150 INPUT "请输入作者名: "; G$; E$ = " "; I$ = " "; H$ =
    " "; GOSUB 230; RETURN
160 INPUT "请输入出版社名: "; I$; E$ = " "; H$ = " "; G$ =
    " "; GOSUB 230; RETURN
170 INPUT "请输入起始单价: "; S1; INPUT "请输入结束单
    价: "; S2; T1=0; T2=0; GOSUB 260; RETURN
180 INPUT "请输入起始时间: "; T1; INPUT "请输入结束时间: ";
    T2; S1=0; S2=0; GOSUB 260; RETURN
190 PRINT D$; "RUN TSGL"
200 D=N2-N1+1; K=0; PRINT O$
210 FOR I=1 TO D; B=N1+I-1; GOSUB 390; BJ=0; GOSUB
    380; IF T=0 THEN C1=C1+1
220 NEXT; PRINT C$; G$ = " "; I$ = " "; H$ = " "; E$ =
    " "; GOSUB 310; RETURN
230 K=0; PRINT O$
240 FOR I=1 TO P; B=I; GOSUB 390; IF L$=H$ OR F$=
    E$ OR Z$=$G$ OR B$=I$ THEN BJ=0; GOSUB
    380
250 NEXT; PRINT C$; GOSUB 310; RETURN
260 K=0; PRINT O$; IF S1=0 AND S=0 THEN 290
270 FOR I=1 TO P; B=I; GOSUB 390; IF S>S1 AND S<S2
    THEN BJ=S; GOSUB 380
280 GOTO 300
290 FOR I=1 TO P; B=I; GOSUB 390; IF T>T1 AND T<
    T2 THEN BJ=T; GOSUB 380
300 NEXT; PRINT C$; GOSUB 310; RETURN
310 HOME; U=0
320 FOR I=1 TO K
330 IF E(I)=0 THEN 350
340 PRINT V%(I); " "; N$(I); TAB(25); E(I); TAB
    (32); L%(I); GOTO 360
350 PRINT V%(I); " "; N$(I); TAB(24); I$; H$; G$;
    TAB(32); L%(I)
360 U=U+1; IF U=INT(U/8)*8 THEN GET Q$
370 NEXT; GET Q$; PRINT "总数="; U-C1; "本"; TAB
    (20); "价值="; ZJ; "元"; CHR$(19); RETURN
380 K=K+1; V%(K)=X%; N$(K)=F$; E(K)=BJ;
    L%(K)=L%; ZJ=ZJ+S; RETURN
390 PRINT R$; B; INPUT X$, F$, Z$, L$, B$, T, L%,
    S; RETURN

```

增加图书

```

10 PRINT D$; "PR #3"; PRINT; HOME; ONERR GOTO 130
20 INPUT "你想增加图书吗? (Y/N)"; Y$; IF Y$="N"
    THEN 130
30 HOME; PRINT TT$; PRINT TAB(1); " * "; TAB(14);
    "增加图书"; TAB(33); " * "; PRINT TT$; Q$="请输入";
    DIM N$(7)
40 DATA 书名, 作者姓名, 类别, 出版社名, 出版日期, 此书
    开本, 此书单价
50 FOR I=1 TO 7; READ H$
60 PRINT Q$; H$; INPUT N$(I); IF I>4 AND VAL(N
    $(I))<=0 THEN PRINT "请重新输入!"; GOTO 60

```

```

70 NEXT; IF P=0 THEN P=1; B=P; GOTO 100
80 PRINT O$; FOR I=1 TO P; PRINT R$; I; INPUT X, F
    $, Z$, L$, B$, T, L%, S; IF T=0 THEN B=P; P=X;
    PRINT C$; GOTO 100
90 NEXT; P=P+1; B=P; PRINT C$
100 PRINT "请稍等, 正在写入!"; PRINT O$; PRINT W$;
    P; PRINT P; " "; N$(1); " "; N$(2); " "; N$(3); " "; N
    $(4); " "; N$(5); " "; N$(6); " "; N$(7); PRINT R$;
    0; INPUT X, F$, Z$, L$, B$, T, L%, S
110 PRINT W$; 0; PRINT X; " "; F$; " "; Z$; " "; L$;
    " "; B$; " "; T; " "; B$; " "; S; PRINT C$; PRINT "完成!"
120 INPUT "你还要增加吗? (Y/N)"; Y$; IF Y$="Y"
    THEN RESTORE; GOTO 50
130 PRINT D$; "RUN TSGL"

```

修删图书

```

10 PRINT D$; "PR #3"; PRINT; HOME; ONERR GOTO 150
20 IF A=4 THEN H$="修改"; GOTO 40
30 H$="删除"
40 PRINT "你想"; H$; "图书吗? (Y/N)"; INPUT Y$; IF
    Y$="N" THEN 150
50 HOME; PRINT TT$; PRINT TAB(1); " * "; TAB(14);
    "图书"; H$; TAB(33); " * "; PRINT TT$; DIM P$(7)
60 PRINT "请输入要"; H$; "记录的书名或编号"; INPUT
    J$; IF VAL(J$)<>0 THEN 90
70 PRINT O$; FOR I=1 TO P; PRINT R$; I; INPUT E, P
    $(1), P$(2), P$(3), P$(4), P$(5), P$(6), P$(7); IF
    P$(1)=J$ THEN PRINT C$; GOTO 110
80 NEXT; PRINT C$; PRINT "无此书名!"; GOTO 140
90 X=VAL(T$); IF X>P OR X<=0 THEN 60
100 PRINT O$; PRINT R$; X; INPUT E, P$(1), P$(2),
    P$(3), P$(4), P$(5), P$(6), P$(7); PRINT C$
110 U$=" "; PRINT "本记录如下: "; PRINT E; U$;
    FOR I=1 TO 7; PRINT P$(I); U$; NEXT; PRINT
120 PRINT "是否"; H$; "? (Y/N)"; INPUT Y$; IF Y$=
    "N" THEN 140
130 ON A-3 GOSUB 160, 220
140 PRINT "还要"; H$; "吗? (Y/N)"; INPUT Y$; IF Y$=
    "Y" THEN 60
150 PRINT D$; "RUN TSGL"
160 FOR I=1 TO 7; GOSUB 200; IF Y$="Y" THEN GOSUB
    210
170 IF I>4 AND VAL(P$(I))<=0 THEN PRINT "请重新
    输入!"; GOSUB 210; IF VAL(P$(I))<=0 THEN 170
180 NEXT; RESTORE; GOSUB 230; RETURN
190 DATA 书名, 作者, 类别, 出版社, 日期, 开本, 单价
200 READ K$; PRINT K$; "修改? (Y/N)"; INPUT Y$;
    RETURN
210 PRINT K$ "改为"; INPUT P$(I); RETURN
220 FOR I=1 TO 4; P$(I)="{ "; NEXT; P$(5)=
    "0"; P$(6)="0"; P$(7)="0"; GOSUB 230; RETURN
230 PRINT "请稍等, 正在"; H$; "!"; PRINT O$; PRINT W
    $; E; PRINT E; " "; P$(1); " "; P$(2); " "; P$(3); " ";
    P$(4); " "; P$(5); " "; P$(6); " "; P$(7); PRINT C$;
    PRINT "完成!"; RETURN

```


ProDOS 磁盘操作系统入门(续)

廖 凯

第五章 程序设计

本章将结合 APPLESOFT 命令和 ProDOS 命令,介绍如何在 ProDOS 系统下进行程序设计。由于 DOS3.3 与 ProDOS 最大区别之一是在数据存储方面,所以将着重介绍文本文件在 ProDOS 系统下的使用。在这章中的大部分程序是在 80 列显示方式下运行的。如果要在 40 列方式下运行,只要略微改动一下即可。

一、APPLESOFT BASIC 基础

这部分只对 APPLESOFT BASIC 作简单的介绍,不熟悉 BASIC 语言的初学者,请参阅有关方面的书籍。

1. 基本指令

END 使程序停止运行
FOR/NEXT 循环执行 FOR 与 NEXT 之间的程序行
GOTO 使程序转到指定的行号去执行
HOME 清除屏幕并将光标移到左上角
IF/THEN 条件语句,如果 IF 的表达式值为真,就执行 THEN 子句;如果为假,就执行下一个语句
INPUT 由当前输入设备接收数据给计算机
NEW 清除当前内存中的程序及所有变量
PRINT 输出字符符号到屏幕或指定的输出设备

2. 开始工作

将 ProDOS User's Disk 磁盘插入驱动器 1 中并启动计算机,过一会儿屏幕将显示主菜单,按 B 键跳出主菜单,进入 BASIC 状态。打入 HOME 指令,清除屏幕,光标移至左上角,再打入 NEW 指令,清除内存程序。

BASIC 执行方式有两种:立即方式和间接方式。立即方式是指直接打入语句或指令,按回车键后立即执行。如刚才的 HOME 和 NEW 指令。间接方式是指每个语句都有一个行号,程序需 RUN 指令来执行。大多数 APPLESOFT BASIC 指令均可在两种方式下执行。

下面的程序是在屏幕上打印 1 到 10 的数字:

```
10 FOR I= 1 TO 10
20 PRINT I
30 NEXT I
40 END
```

运行后,我们可以看到每当 FOR/NEXT 循环的变量 I 递增 1 时,打印语句就将变量 I 的值打印出来,如果将行号 10 改为:

```
10 FOR I=1 TO 10 STEP 2
```

运行结果则是印出 1 至 10 之间的奇数,这是因为变量 I 每次递增 2。如果我们将行号 20 改为:

```
20 PRINT I;或 20 PRINT I,
```

运行程序后,你会看到它们的变化是在屏幕输出格式上。分号不允许输出字符之间有空格,而逗号则可以,它是按制表域输出字符。

下面的程序是计算汽车每加仑汽油跑多少英里:

```
10 HOME
```

```
20 INPUT"NUMBER OF GALLONS IN A FULL TANK?";FT
30 INPUT"ENTER THE TOTAL MILES DRIVEN?";TM
40 MPG=TM/FT
50 PRINT"MILES PER GALLON IS";MPG
70 END
```

运行后程序要你输入加仑数和英里数给计算机,计算机会自动计算并将结果打印出来。在上面的程序中再加入以下语句:

```
35 IF TM=0 THEN 70
60 GOTO 20
```

这样就可以连续计算了,直到你输入的英里数为零,程序才停止运行。

二、数组和串变量

这部分讨论 ProDOS 如何处理串变量的输出。

1. 有关指令

ASC 返回串表达式值第一个字符的 ASCII 码
CALL 使程序转向指定的内存单元开始的机器语言子程序
CHR \$ 返回一个与表达式值相对应的 ASCII 字符
DIM 定义数组,在内存中分配变量空间
GET 等待用户由键盘输入单个字符,无需按回车键
GOSUB 使程序转向子程序,当遇到 RETURN 时,程序立刻转回到 GOSUB 语句的下一句
INVERSE 使字符反相输出到屏幕上
LEFT \$ 返回字符串中最左边的若干字符
LEN 计算字符串的长度
MID \$ 返回字符串中指定的若干字符
NORMAL 使字符正常输出到屏幕上
ONERR GOTO 当错误发生时,转到指定的语句
PEEK 返回指定内存地址单元中的值
READ/DATA 当有 READ 指令时,计算机寻找 DATA 指令,将 DATA 语句中的数据赋给 READ 语句中的变量
REM 用于书写程序的注解,运行时忽略不计
RETURN 使程序回到上一个 GOSUB 后面的语句
RIGHT \$ 返回字符串中最右边的若干字符
STOP 终止程序执行并显示行号
VAL 将字符型数据转换成算术型

2. 数组

数组用于识别一组同一名称的变量,数组的每个元素用赋给它的编号来识别。数组可用于串变量、整数或实数。下面的程序将说明如何组成数组 G。

```
10 DIM G(4)
20 FOR I=0 TO 4
30 READ G(I)
40 NEXT I
50 DATA 22.68,1002,601,-.033,98
```

```

60 REM Print the numbers with another loop
70 FOR H=0 TO 4
80 PRINT G(H)
90 NEXT H
100 END

```

此程序首先定义一个数组 G,它有五个元素 G(0)至 G(4),然后经循环语句和 READ/DATA 语句将数据赋给数组的五个元素,最后由 FOR/NEXT 语句和 PRINT 语句将数组五个元素的值顺序打印出来。

3. 串变量

串变量表示一个字符串。串变量将 \$ 号放在变量的最右边以区别于数值变量。在给串变量赋值时,字符串必须用引号括起来。如:

```

A$="a string 12345"
A$="good morning"

```

下面的程序是询问用户是否使用 80 列卡,即屏幕显示是否是 80 列。程序会根据用户的回答来决定是用 40 列还是用 80 列方式显示磁盘目录。由于 ProDOS 使用分级文件结构,所以程序设计者必须注意路径。部首 (PREFIX) 被设为要 CATALOG 的磁盘。

```

1 REM Program to CATALOG or CAT different ProDOS disks.
4 HOME
5 D$=CHR$(4)
10 INPUT "Is the screen in 80-column mode?";A$
20 IF LEFT$(A$,3)="YES"OR LEFT$(A$,3)="yes"
   THEN 50
30 IF LEFT$(A$,2)="NO"OR LEFT$(A$,2)="no"
   THEN W=1;GOTO 50
40 HOME;GOTO 10
50 INPUT "Enter prefix to be CATALOGed:";P$
55 ONERR GOTO 170
60 IF LEFT$(P$,1)<>"/"THEN P$="/" + P$
70 PRINT D$;"PREFIX";P$
80 IF W=1 THEN 120
90 PRINT D$;"CATALOG"
100 GOTO 150
120 PRINT D$;"CAT"
150 END
170 POKE 216,0
180 GOTO 50

```

行号 5 是将 ASCII 码为 4 的字符(即 CTRL-D)赋给变量 D\$。行号 10 等待用户输入大写或小写的 YES 或 NO,行号 20 和行号 30 分析 INPUT 语句并进行比较。若符合条件就转至行号 50;若不符就执行行号 40。行号 50 是等待用户输入要被列目录的部首,行号 60 是分析用户输入的部首,若卷目录前面没有斜线,那么将自动加上一个。行号 55 是错误处理子程序,如果输入了错误的部首,那么程序将转向行号 170 处理。行号 70 至行号 150 是程序根据分析结果,以 80 列或 40 列显示方式列印目录。

4. 进一步介绍 PREFIX

• 22 •

当你初次启动系统时,系统没有指定磁盘的部首。在没有使用 PREFIX 命令的情况下,你可以将驱动器号与命令一起使用,如同 DOS3.3 一样,对磁盘进行操作,而不会有路径错误发生。

在立即方式下执行 PREFIX 命令时,当前的部首将被显示出来。此命令也可在程序方式下设置部首:

```

5 D$=CHR$(4)
10 INPUT "NEW PREFIX:";P$
20 PRINT D$;"PREFIX";P$
你也可以将部首存储在一个串变量中:
5 D$=CHR$(4)
10 PRINT D$;"PREFIX";INPUT";P$

```

5. CREATE 命令

ProDOS 具有产生任何文件类型的能力。下面是一个用 CREATE 命令来产生不同文件类型的程序。CREATE 命令最常用于产生目录文件。

```

100 HOME;D$=CHR$(4)
104 PRINT D$;"PR #3"
110 PRINT "Create A File Type"
120 PRINT "1. Applesoft Program BAS"
130 PRINT "2. Applesoft Variable VAR"
140 PRINT "3. Binary BIN"
150 PRINT "4. Directory DIR"
160 PRINT "5. ProDOS System Program SYS"
170 PRINT "6. Relocatable Code REL"
180 PRINT "7. Text TXT"
190 PRINT "8. User defined $F#"
200 PRINT;GET N$;N=VAL(N$)
210 IF ASC(N$)<48 OR ASC(N$)>56 THEN 200
220 IF N=1 THEN A$="BAS"
230 IF N=2 THEN A$="VAR"
240 IF N=3 THEN A$="BIN"
250 IF N=4 THEN A$="DIR"
260 IF N=5 THEN A$="SYS"
270 IF N=6 THEN A$="REL"
280 IF N=7 THEN A$="TXT"
290 IF N=8 THEN A$="$F#"
300 PRINT;PRINT "Give the pathname or appropriate file
   name to CREATE a ";A$ "type file";PRINT
305 INPUT";F$
310 IF N=8 THEN 350
320 PRINT D$;"CREATE";F$",";A$
330 GOTO 400
350 PRINT;PRINT "Enter the User Defined number between 1
   and 8 inclusive";PRINT
360 INPUT";NO$
370 IF VAL(NO$)<1 OR VAL(NO$)>8 THEN 350
380 A$="$F"+NO$
390 GOTO 320
400 END

```

注:行号 104 是启动插在 3 号槽口的 80 列卡,后同。对于中华学习机可将此句删去或将槽口号改为 80 列卡所插的槽口号。(下转第 16 页)

第九讲 指针与函数、结构

李文兵

1. 函数指针

指针不仅可以指向一般变量、数组,而且还可以指向函数。我们把指向函数的指针叫做函数指针,其一般定义形式如下:

数据类型 (*标识符)();

例如: int (*p)();

表示 p 是一个指向函数的指针,且说明 p 所指向的函数的返回值是整型数据。

函数指针的用法,如练习 9.1 所示。

```
A>type exp9-1.c
main()
{ int (*p1)(),print1();
  int (*p2)(),print2();
  p1=print1;
  p2=print2;
  (*p1)(123);
  (*p2)(456);
}
print1(x)
int x;
{ printf("%d\n",x);}
print2(y)
int y;
{ printf("%d\n",y);}
```

```
A>exp9-1
123
456
```

在该程序的函数体中:

1、2 行为说明语句,说明 p1、p2 为函数指针,print1()和 print2()为函数;

3、4 行为函数指针赋值语句,用来使 p1 和 p2 分别指向函数 print1()和 print2();

5、6 行为函数的函数语句方式调用,即利用函数指针 p1 和 p2 分别调用函数 print1()和 print2(),其中 123 和 456 为代替形参的实参。

再看一个例子,如练习 9.2 所示。

```
A>TYPE EXP9-2.C
#include "time.h"
#include <stdio.h>
#include "stdlib.h"
static int (*cmpf)();
#define MAX 10
main()
{ int quick(int *,int,int(*)()),cmp(int,int);
```

```
int i,array[MAX];
long now;
srand((unsigned)time(&now));
for(i=0;i<MAX;i++)
printf("%d ",array[i]=rand());
quick(array,MAX,cmp);
printf("\n\n");
for(i=0;i<MAX;i++)
printf("%d ",array[i]);
printf("\n\n");
}
quick(item,count,cmp)
int item[],count,(*cmp)();
{ cmpf=cmp;
  qs(0,count-1,item);
}
qs(l,u,item)
int l,u,item[];
{ int i,j,x,y;
  i=l;j=u;
  x=item[(l+u)/2];
  do { while((item[i]>x)&&(i<u))i++;
      while((item[j]<x)&&(j>l))j--;
      if(i<=j)
      { y=item[i];
        item[i]=item[j];
        item[j]=y;
        i++;j--;
      }
  } while(i<=j);
  if(l<j)qs(l,j,item);
  if(i<u)qs(i,u,item);
}
int cmp(x,y)
int x,y;
{ if(x>y) return(-1);
  else if(x==y) return(0);
  return(1);
}
```

```
A>exp9-2
15250 7082 21072 29365 24173 1478 22943 7166 10947 23544
29365 24173 23544 22943 21072 15250 10947 7116 7082 1478
```

该练习是快速排序程序。快速排序是靠函数 qs() 实现的,qs() 为递归函数,2 次递归调用其本身。快速排序原理是以数组的中间元素为基础值,把比中间元素值大的放在中间元素的一边,小的放在另一边。把小的放在中间元素的左边,是升序;而把大的放在左边,

是降序。按升序,还是按降序,qs()中是靠连结 item()与 x 的关系运算符决定的。这里,qs()中是按降序处理的。如图 9.1 所示,第 1 次选的中间元素其值为 5,以它为基准进行比较,则把比 5 大的换到 5 的左边,把小的放在 5 的右边。

原数组: item[0]..... item[17]

1	4	2	6	3	7	4	9	5	1	6	4	7	6	8	2	9	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

x=item[(0+17)/12]的结果为: x=5

第 1 次排序后:

9	8	6	6	7	7	6	9	5	1	4	3	2	4	2	1	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

图 9.1 qs()降序第 1 次排序后情况

改变 item[i]与 x 间、item[j]与 x 间不等号的方向,即可改为升序排序。

指向函数的指针用 static 说明为全局变量,即:

```
static int (*cmpf)();
```

其中 cmpf 为指针名,用它指向比较函数 cmp(),即:

```
cmpf=cmp;
```

2. 返回指针的函数

说明语句:

```
int *p();
```

表示 p 是一个函数,一个返回整型指针的函数。返回指针的函数的用法,如练习 9.3 所示。

```
A>type exp9-3.c
```

```
#include <stdio.h>
```

```
main()
```

```
{char *cpy(char *,char *);
```

```
char s[]="abcd";
```

```
char d[]="efgh";
```

```
printf("%s",cpy(s,d));
```

```
}
```

```
char *cpy(s,d)
```

```
char *s,*d;
```

```
{int i;
```

```
char *p;
```

```
p=d;
```

```
if(*s==NULL) exit(-1)
```

```
do { *d++=*s++;
```

```
}while(*s!=NULL);
```

```
return (p);
```

```
}
```

```
A>exp9-3
```

```
abcd
```

该程序是从 s[] 中把字符串拷贝到 d[] 中。这时函数 cpy() 的返回值为指针 d。

3. 指针参数与函数参数值调用

在 C 语言里,函数的参数值有两种调用方法:

(1) 值调用 (call by value) 值调用的例子如练习 9.4 所示。

```
A>type exp9-4.c
```

```
main()
```

```
{int i,j;
```

```
char a;
```

```
i=110;j=220;a='c';
```

```
func(i,j,a);
```

```
}
```

```
func(i,j,k)
```

```
int i,j;
```

```
char k;
```

```
{ printf("arg1=%d\narg2=%d\narg3=%c\n",i,j,k);
```

```
}
```

```
A>exp9-4
```

```
arg1=110
```

```
arg2=220
```

```
arg3=c
```

该程序调用函数 func() 时,引用 3 个参数。

(2) 参考调用 (call by reference) 这是使用指针

的值调用,其例如练习 9.5 所示。

```
A>type exp9-5.c
```

```
main()
```

```
{int i=111,j=222;
```

```
char a='a';
```

```
func(&i,&j,&a);
```

```
printf("i=%d\nj=%d\na=%c\n",i,j,a);
```

```
}
```

```
func (i,j,k)
```

```
int *i,*j;
```

```
char *k;
```

```
{ printf(" arg1=%d\narg2=%d\narg3=%c\n", *i, *j, *
```

```
k);
```

```
(*i)++;
```

```
*j++;
```

```
*k++;
```

```
}
```

```
A>exp9-5
```

```
arg1=111
```

```
arg2=222
```

```
arg3=a
```

```
i=112
```

```
j=222
```

```
a=a
```

从程序中,我们可以看到,函数 func() 的三个参数 i,j,k 都是指针,也就是说,i,j,k 为指针参数。这样,在调用它时,必须使其参数与之对应,即也必须是指针类型,因此,要在一般变量 i,j,a 前面冠以取址运算符 &,即使其三个参数为:

```
&i,&j,&a
```

不返回值的函数,可用关键字 void 来说明,如练习 9.6 所示。

```
A>type exp9-6.c
```

```
#include <stdio.h>
```

```
main()
```

```
{void pnt(char *);
```

```
char s[]="abcd";
```

```
pnt(s);
```

```

}
void pnt(s)
char *s;
{ do {printf("%c", *s++);
} while (*s != NULL);
}

```

A>exp9-6
abcd

从练习 9.6 可以看出,在 Turbo C 中可用 void 类型来显式说明一个无返回值的函数,除此之外,void 还有如下一些有用法:

①用来说明一个空参数表 如练习 9.7 所示。

```

A>type exp9-7.c
void putmsg(void)
{ printf("Hello,world\n")
}
main()
{ putmsg(); }

```

A>exp9-7
Hello,world

②用来作一个特殊结构,如:

```
(void) getch();
```

其功能是暂停执行,直到按任何键为止。

③说明一个 void 指针。注意,这不是空指针,而是建立一个数据目标为任何类型的指针,对它们的类型无需知道。可把任何指针赋值为 void 指针,反之亦然,而无需事先安排。

void 指针不能使用 *。

4. 指针与结构

(1) 指针型结构成员 指针变量同样可以作结构成员。下面的结构定义中,就把指针变量 pname 当作了结构的成员。

```

struct person {
    int number;
    char * pname;
}

```

这样的结构的用法,如练习 9.8 所示。

```

A>type exp9-8.c
#include <stdio.h>
struct person {
    int number;
    char * pname;
}
main()
{ int i;
  struct person qp[10];
  for(i=0; i<10; i++)
    qp[i].pname=NULL;
  qp[1].number=100;
  qp[1].pname="aoi";
}

```

```

qp[2].number=200;
qp[2].pname="akai";
/* print out */
i=1
do { printf("%d %s\n",
    qp[i].number, qp[i].pname);
    i++;
} while (qp[i].pname!=NULL);
}

```

A>exp9-8
100 aoi
200 akai

(2) 结构指针 指针不仅可以指向变量、数组、指针、函数,也可以指向结构。我们把指向结构的指针,叫做结构指针。结构指针在链表、二叉树等数据结构中是很有用的。设一链表的结构为:

```

struct address {
    char name[20];
    char city[40];
    char zip[5];
    struct address * next;
};

```

如果定义如下指针 top:

```
struct address * top
```

则指针 top 就是指向该结构的一个结构指针。这时,就可使用->运算符,引用该结构的各个成员,如下所示:

```

top -> name
top -> city
top -> zip
top -> next

```

连接结构数据可使用引用自身的结构,如练习 9.9 中的结构 address。练习 9.9 可连接 4 个新的结构,并能读出全部第一成员姓名。

```

A>type exp9-9.c
#include <stdio.h>
#include "stdlib.h"
#include "alloc.h"
struct address { char name[9];
    char city[7];
    char zip[7];
    struct address * next;
};

```

```

main()
{ struct address * new, * last = 0;
  int i = 0;
  do { if ((new = malloc(sizeof(struct address))) == 0)
    { printf("out of memory\n");
      exit(0);
    }
    if (last == NULL)
      new->next = NULL;

```

(下转第13页)



汽车转弯信号灯控制

(KDC—Ⅲ 最小系统应用实例)

张培仁 刘振安

汽车控制面板上有一个控制杆。此控制杆有三个位置：

中间位置时，汽车不转弯；向上时，汽车左转；向下时，汽车右转。汽车转弯时，要求左右尾灯，左右头灯和仪表板上的2个指示灯发出闪烁的信号。当应急开关合上时，所有6个信号灯都应闪烁。汽车刹车时，2个尾灯发出不同闪烁信号。如刹车时正在转变，则相应的转弯表1转弯信号工作的真值表

输入信号				输出信号			
刹车开关	应急开关	左转开关	右转开关	左头灯和仪表板灯	右头灯和仪表板灯	左尾灯	右尾灯
0	0	0	0	断	断	断	断
0	0	0	1	断	闪烁	断	闪烁
0	0	1	0	闪烁	断	闪烁	断
0	1	0	0	闪烁	闪烁	闪烁	闪烁
0	1	0	1	闪烁	闪烁	闪烁	闪烁
0	1	1	0	闪烁	闪烁	闪烁	闪烁
1	0	0	0	断	断	通	通
1	0	0	1	断	闪烁	通	闪烁
1	0	1	0	闪烁	断	闪烁	通
1	1	0	0	闪烁	闪烁	通	通
1	1	0	1	闪烁	闪烁	通	闪烁
1	1	1	0	闪烁	闪烁	闪烁	通

闪烁信号不受影响。

汽车转弯或应急状态下，外部信号灯和仪表板指示灯的闪烁频率为1Hz，称低频信号。当停靠开关合上时，外部信号灯以高频(约30Hz)频率闪烁，以适应低亮度背景的使用场合，增加亮度。

硬件电路原理如图1所示：

图中用1.1KΩ电阻与发光二极管串联模拟车灯，两旁并上1.5KΩ电阻消除了故障检测电路引起的发光二极管余光现象。故障指示灯可以检测所有信号灯是否有断路现象，从P3.4的电平来判断。

对汽车转弯灯控制系统功能可以用数字电路来实现，但灵活性不够。

从表一中列出转弯信号灯的真理值表，我们用单片机最小系统来完成逻辑电路要完成的功能。

系统软件：

主要二部分，一部分是主程序，另一部分是中断服务程序。

主程序开始是输入，输出口线说明和变量定义。以后就是初始化，置定时器0，总允许中断，启动定时器0，等待等几个过程。定时器0和一个软件计数器SOB—DIV产生一秒的定时信号，以实现1Hz 闪烁功能。

中断服务程序：

主要先断是否1秒到。1秒到重置

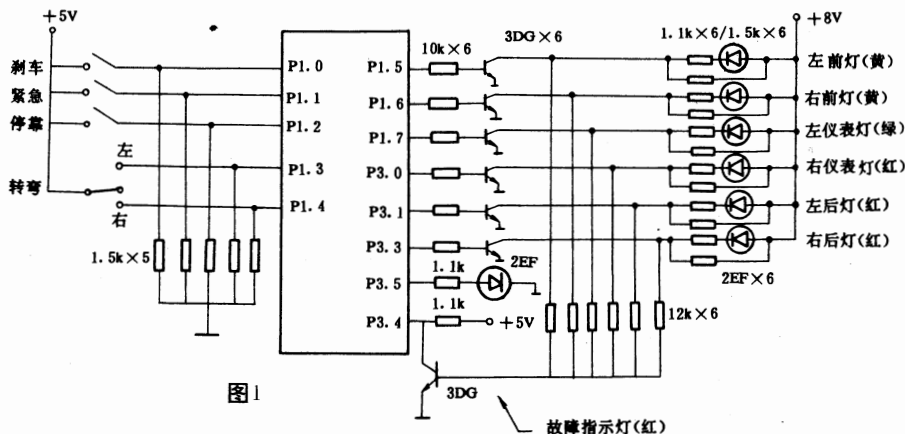


图1

T₀时间常数和 SOB-DIV 的值,并检查信号灯是否有故障。如果1秒未到,产生了30Hz,占空比是62.5%的信号,输出相应仪表左右灯,车前,车后灯,最后返回。

因为 MCS-51是一个功能很强布尔处理机,这方面指令很多,使运算速度快,程序可读性好。这是单片机的很大优点。单片机是功能很强易于进行数字逻辑运算的处理机。程序清单如下:

```

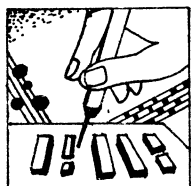
BRAKE EQU P1.0      ;刹车
EMERG EQU P1.1      ;应急
PARK EQU P1.2       ;停靠
L-TURN EQU P1.3     ;左转
R-TURN EQU P1.4     ;右转
L-FRNT EQU P1.5     ;左前灯
R-FRNT EQU P1.6     ;右前灯
L-DASH EQU P1.7     ;左仪表灯
R-DASH EQU P3.0     ;右仪表灯
L-REAR EQU P3.1     ;左后灯
R-REAR EQU P3.3     ;右后灯
TO EQU P3.4         ;故障检测输入
S-FALL EQU P3.5     ;报警
SUB-DIV EQU 20H     ;软件计数器
SUB-DIV.0 EQU 00H
SUB-DIV.1 EQU 01H
SUB-DIV.2 EQU 02H
SUB-DIV.7 EQU 07H
HI-FREQ EQU SUB-DIV.0
LO-FREQ EQU SUB-DIV.7
DIM EQU PSW.1 ;
ORG 1000H
LJMP INIT
ORG 100BH
MOV TH0, #0F8H      ;重置定时器0
PUSH 0D0H
AJMP UPDATE
ORG 1040H
INIT: MOV TL0, #00H   ;定时器0置初值
MOV TH0, #0F8H
MOV TMOD, 11110001B
MOV SUB-DIV, 244
MOV IE, #10000010B
SETB TR0
ASD: SJMP ASD
UPDATE: DJNZ SUB-DIV, TOSERV
MOV SUB-DIV, #224
SETB L-FRNT         ;对线路检测故障
SETB R-FRNT
SETB L-DASH
SETB R-DASH
SETB L-REAR
SETB R-REAR
CLR L-FRNT
JB TO, FAULT
SETB L-FRNT
CLR L-DASH

```

```

JB TO, FAULT
SETB L-DASH
CLR L-REAR
JB TO, FAULT
SETB L-REAR
CLR R-FRNT
JB TO, FAULT
SETB R-FRNT
CLR R-DASH
JB TO, FAULT
SETB R-DASH
CLR R-REAR
JB TO, FAULT
SETB R-REAR
JB TO, TOSERV
FAULT: CPL S-FALL      ;使报警闪一次
TOSERV: MOV C, SUB-DIV.1 ;30Hz 信号
ANL C, SUB-DIV.0
ORL C, SUB-DIV.2
ANL C, PARK
MOV DIM, C
MOV C, L-TURN         ;控制左仪表灯
ORL C, EMERG
ANL C, LO-FREQ
MOV L-DASH, C         ;控制左前灯
MOV F0, C
ORL C, DIM
MOV L-ERNT, C
MOV C, L-TURN         ;控制左后灯
CPL C
ANL C, BRAKE
ORL C, F0
ORL C, DIM
MOV L-RFAR, C
MOV C, R-TURN         ;控制右边灯
ORL C, EMERG
ANL C, LO-FREQ
MOV R-DASH, C
ORL C, DIM
MOV R-FRNT, C
MOV C, R-TURN
CPL C
ANL C, BRAKE
ORL C, F0
ORL C, DIM
MOV R-REAR, C
POP 0D0H
RETI                  ;返回
END

```

Z80 CPU 与单片微处理器8031的交换

高殿试

Z80 CPU 微电脑系统必须有一系列的外围 LSI 芯片支持。例如,需要 EP—ROM 存储器、RAM 存储器、I/O接口等,这样,使控制系统较复杂。

单片微处理器将 CPU、ROM、RAM 与 I/O 集成在一块 LSI 芯片中,体积缩小,集成度提高,为微电脑应用开发带来了很大方便,目前,较为流行的8位单片微处理器是 MCS51系列单片微处理器。

MCS51系列的特点

8051系列单片微处理器的主要特点

- 1) 8位 CPU 为中央处理器。
- 2) 能够进行位操作。
- 3) I/O 口有32条双向引脚,也能实现逐个引脚存取。
- 4) 内藏128字节的数据 RAM。
- 5) 有两个可编程的16位定时器/计数器。
- 6) 内藏双重异步接收发送装置。
- 7) 有五个中断源,是具有两级优先权的中断系统。
- 8) 64K 字节的数据存储器和程序存储器严格分开。

单片微处理器8031

MCS-51系列单片微处理器概要如表1所示。

8051片内设有程序存储器 ROM,适用于大批量的产品中引用。

8751片内设有程序存储器 EP—ROM。可以根据需要对 EP—ROM 进行程序写入或程序擦除。必须具备专用的 EP—ROM 写入器。

8031片内没有程序存储器。因此8031所有的取数均来自外部存储器,很容易搞到的2716或者2732 EP—ROM 存储器。

使用8031的中央处理部件

μ P80微电脑套件是按功能划分的部件,通过接插件的连接,能够完成各种各样的实验。

“实践篇”的六块部件:键盘部件、读写部件、存储器部件,中央处理器部件、输入输出部件和4位数字显示部件。

本文讨论用8031替代中央处理部件的 Z80 CPU。这样,六块部件连接起来就组成一套简易 MCS—51系列开发系统,8031价格便宜,使用方便。如果再配备 EP—ROM 写入器,将为 MCS—51系列单片微处理器的应用开发提供极大方便。

8031中央处理部件的电路图如图1所示。

(1) 数据总线 $D_0 \sim D_7$:

数据总线 $D_0 \sim D_7$ 使用 P0.0~P0.7引脚。

和外部进行8位数据的双向传输。在实验过程中,下方连接存储器部件,可以看作外部程序存储器,上方连接输入输出部件,看作外部数据存储器。与 ALE 信号同步,输出8位数据。

选择外部程序存储器由 $\overline{\text{PSEN}}$ 信号控制。

P₀端口与外部数据存储器进行8位数据的输入、输出由 $\overline{\text{RD}}$ 、 $\overline{\text{WR}}$ 信号控制。

(2) 地址总线 $A_0 \sim A_7$:

在与 ALE 信号同步同时,随8位数据的输出输入低8位地址 $A_0 \sim A_7$ 由74LS373输出。

表1 MCS—51系列单片微处理器一览表

ROM 型	无 ROM 型	EPROM 型	ROM 容量	RAM 容量	16位定时器	电路类型
8051	8031	(8751)	4KB	128B	2	HMOS
8051AH	8031AH	8751H	4KB	128B	2	HMOS
8052AH	8032AH	8752BH	8KB	256B	3	HMOS
80C51BH	80C31BH	87C51	4KB	128B	2	CHMOS
83C152	80C152	—	8KB	256B	2	CHMOS
83C51FA	80C51FA	87C51FA	8KB	256B	4	CHMOS
83C51FB	80C51FB	87C51FA	16KB	256B	4	CHMOS
83C51GA	80C51GA	87C51GA	4KB	128B	2	CHMOS





电脑巧开发

在 APPLE 机接口卡中固化

BASIC 应用程序

解武杰

如果 APPLE 机在特殊场合单一使用,把应用程序固化在接口卡中,既便于携带又增加了系统的可靠性。为此,下面给出一种比较简单的电路,其原理如图1:

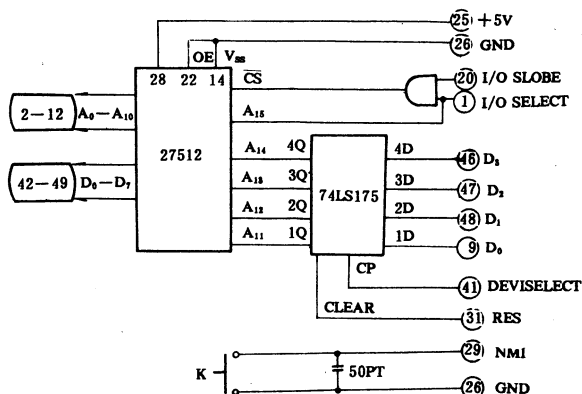


图1

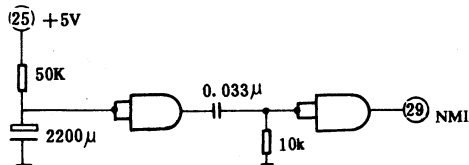


图2

用一块27512EPROM 存储管理程序和用户的基本应用程序,应用程序可达32K。74LS175四D触发器产生 EPROM 片选信号。非屏蔽中断信号 NMI 通过按钮开关 K 控制。这样,主机启动后首先进入 BASIC 状态。若要进入用户程序,用开关 K 触发即可产生非屏蔽中断信号。机器原有功能未受影响。电容 C 用作抗干扰,防止误触发。

设接口卡占用2号插槽,应把原主机监控程序 \$FFFA 和 \$FFFB 单元内容分别改为00和 C2,同时用短接帽短接2号插槽。这样,当用开关 K 触发 NMI 即可进入卡上管理程序。运行管理程序的结果是把用户程序以2K 字节为单位调入内存,再给0页有关单元置初值,然后执行用户程序。

管理程序清单:

C200-	20 FB DA	JSR \$DAFB
C203-	A9 08	LDA # \$08
C205-	85 FF	STA \$FF
C207-	A2 00	LDX # \$00

C209-	8A	TXA
C20A-	8D A0 C0	STA \$C0A0
C20D-	A0 00	LDY # \$00
C20F-	98	TYA
C210-	85 3C	STA \$3C
C212-	A9 01	LDA # \$01
C214-	85 42	STA \$42
C216-	A9 C8	LDA # \$C8
C218-	85 3D	STA \$3D
C21A-	A5 FF	LDA \$FF
C21C-	85 43	STA \$43
C21E-	A9 CF	LDA # \$CF
C220-	85 3F	STA \$3F
C222-	A9 FF	LDA # \$FF
C224-	85 3E	STA \$3E
C226-	20 2C FE	JSR \$FE2C
C229-	EA	NOP
C22A-	A5 FF	LDA \$FF
C22C-	18	CLC
C22D-	66 08	ADC # \$08
C22F-	85 FF	STA \$FF
C231-	E8	INX
C232-	C9 0F	CMP # \$06
C234-	D0 D5	BNE \$D5
C236-	A9 08	LDA # \$08
C238-	85 68	STA \$68
C23A-	A9 01	LDA # \$01
C23C-	85 67	STA \$67
C23E-	A9 33	LDA # \$33
C240-	85 6A	STA \$6A
C242-	85 6E	STA \$6E
C244-	85 6C	STA \$6C
C246-	A9 BC	LDA # \$BC
C248-	85 69	STA \$69
C24A-	85 6B	STA \$6B
C24C-	85 6D	STA \$6D
C24E-	85 AF	STA \$AF
C250-	A9 01	LDA # \$01
C252-	85 AE	STA \$AE
C254-	20 66 D5	JSR \$D566
C257-	60	RTS
C258-	00	BRK

C259- 00 BRK
C25A- 00 BRK

该管理程序的 CMP# \$nn 语句中 nn 应根据用户 BASIC 程序长度在 01H 至 0FH 之间选择。\$6A, \$6E, \$6C 中存放用户程序未地址高字节。\$69, \$6B,

\$6D, \$AF 中存放用户程序未地址低字节,其赋值也应根据 BASIC 程序长度而定。

管理程序在 EPROM 中固化时从 0200 地址开始,用户程序从 8000 地址开始。

如果想在开机后直接进入用户程序,用图2电路可产生 NMI 信号。

设计微电脑控制电路的小常识

王 林

微型计算机在过程控制的开发应用中要进行不同程度的接口扩充、接口芯片与微型计算机总线的挂接、其 CPU 承受负载的能力都是有限的,设计者如不很好注意,就会带来许多困难,下面就 Z80—CPU 为例来说明其带负载的能力。

在计算机总线上,传送的均为“1”“0”信号,挂在总线上的各个部件对“1”“0”信号进行识别,从而完成各种操作。

以正逻辑为例,在微计算机中,“1”“0”所代表的电压范围,当逻辑1时,必须输出大于 2.4V 的电压,而要输出逻辑0,则必须输出小于 0.4V 的电压。在电路中,器件输出的电压除与本身的输出能力有关外,还与负载有关。Z80—CPU 输出口在输出逻辑1时,要满足逻辑1的电压值,负载电流最大不应超过 250 μ A,或 $R_L \geq 2.4V/250\mu A = 9.6K\Omega$;而输出逻辑0时,灌入电流不能大于 1.8mA,方能使输出电压不大于 0.4V。

在微计算机中,与总线接口的器件有 MOS 器件,也有 TTL 器件。MOS 器件的功耗很小,在小系统中一般不会对 CPU 的负载能造成威胁,而 TTL 器件一般较 MOS 器件功耗大,但由于 TTL 器件速度快,驱动能力大,不易静电损坏,因此,在接口电路中大量应用,在总线上挂的 TTL 器件较多时,要满足负载条件,一般要加总线驱动器。

表一列出 74 系列中六个反相器作为输入负载和输出驱动情况下的技术条件,从而不难看出,Z80—CPU 输出口在以反相器输入端为负载时最多可带几个反相器。

CPU 输出“1”时 $250\mu A/20\mu A = 12.5$ 个门

CPU 输出“0”时 $1.8mA/0.36mA = 5$ 个门

因此,Z80—CPU 在以 74LS04 的输入端为负载时,最多可带 5 个门,否则就必须加总线驱动才能保证正确的逻辑输出。

		74LS04	7404	74S04
输出 驱动	1	输出电压降到 2.4V 时能供给的最小电流	400 μ A	400 μ A
	0	输出电压升到 0.4V 时能吸入的最小电流	8mA	16mA
输入 负载	1	输入电压升到 2.4V 时的最大电流	20 μ A	40 μ A
	0	输入电压降到 0.4V 时的最大电流	0.36mA	1.6mA

对其它电路在使用中也存在以上问题,读者可查阅有关手册,以了解其使用参数。

微机通用系统板测试卡

适合 8088 兼容机、286 兼容机和 386 兼容机系统板的维修测试设备,国家“七·五”重点攻关项目,获北京第 2 届国际博览会金奖。

·通用性强,适用于所有采用 PC/AT 总线的各种机型。

·使用方便,测试卡的使用与普通控制卡完全相同。

·无特殊的设备要求,测试时使用原微机的 CGA 或 EGV 或兼容的显示系统。

·显示的结果直观明确,可将故障测试定位到芯

片/模块一级。

·资料齐全,备有详细的使用说明和维修方法介绍,并以集中方式免费培训。

·负责终生保修,并作维修方面的技术后援。

微机通用测试卡 全国统一零售价 3800.00 元,有资料备索。

《电子与电脑》编辑部办理邮购经销业务。

邮局汇款:北京 173 信箱《电子与电脑》编辑部 邮编 100036

包装邮资费另加 10 元。

型号不同的绘图仪绘制 TANGO 软件图形的方法

湖北黄石机械自动化研究所 黄健 陈军

TANGO 的原理图输出程序 Schematic—PLOT 可以在多种打印机、绘图仪上输出各种格式的原理图,其图形的大小可任意设置,甚至 x、y 方向的比例因子均可独立调整。本文以 SHARP 的 CE—515P 绘图仪为例,介绍与 TANGO 不兼容的绘图仪绘制 TANGO 原理图的简便方法。

该法分两步进行。

(一)首先通过 Schematic—PLOT 将欲输出的原理图文件转化为绘图指令序列文件。

(二)根据您的绘图仪编一个转换程序,将第(一)步中得到的绘图指令序列文件中的绘图指令转换成您的绘图仪的相应指令,使绘图仪绘出图形。

第(一)步

1. 运行 PLOT.EXE 文件,屏幕显示主菜单:

Tango—Schematic PLOT TER MAIN MENU

Active Directory;D:\TANGO

Sheet Filename;

Sheet Size;B

Free memory;416624 Bytes

Orientation;Normal Border Plot;ON

Plotter Pen;1 Pin Numbers;OFF

Communications Menu Scale Menu

Dir Load Plot

X offset;1.00 Y offset;0.50

Pen Speed;9

Quit

>

2. 按 L(Load)键,装入欲转换的原理图文件。

按 O (Orientation) 键,选择原理图按横向(Normal)还是按纵向(Rotated)打印绘图。

按 C(Communication Menu)键,进入通讯协议子菜单。

按 L(Plotten Language)键,选择到 Roland DXY—800。

按 P(Plotter Output Device)键,直到出现 File-name。

按 F(Filename)键,输入欲得到的绘图指令序列文件名。

按 Q(Quit)键,回到主菜单。

按 S(Scale Menu)键,进入比例选择子菜单。

按 S(Plotter Scale)键,指定输出图形的比例。对于 CE—515P 绘图仪我们设置为 0.400:1.000。

根据菜单提示还可选择 x 方向比例因子、y 方

向比例因子等,这里不再详述。

按 Q(Quit)键,回到主菜单。

按 P(Plot)键,转换开始。

当转换结束后,便得到了绘图指令序列文件。

第(二)步

编制转换程序。下列程序是针对 CE—515P 绘图仪编写的,供大家参考。

```
1 LPRINT CHR $(13)
2 LPRINT CHR $(27);"a"
3 LPRINT CHR $(13)
4 LPRINT CHR $(27);"b"
5 LPRINT CHR $(13)
6 DIM x(10),y(10)
7 x=0;y=-999
19 INPUT "File Name:";f$
20 OPEN "r",#1,f$,1:FIELD
   #1,1 AS a$
30 b=LOF(1)
31 LPRINT "M";x;"",y
32 LPRINT CHR $(13)
33 LPRINT "I"
40 g=5
50 FOR i= TO b
60 GET #1,g:g=g+1
61 c=ASC(a$)
70 IF c=77 THEN 100
80 IF c=68 THEN 150
81 IF c=13 THEN 300
82 IF c=67 THEN 400
90 LPRINT a$;
95 NEXT i
96 CLOSE:END
100 LPRINT "M";
110 GOTO 95
150 LPRINT "D";
160 GOTO 95
200 GOTO 95
300 LPRINT CHR $(13)
310 GOTO 95
400 GOSUB 1000
410 GOTO 60
1000 c2=0
1001 j=4;k=4;c5=0
1002 x(5)=200;x(6)=200;
   y(5)=200;y(6)=200
1020 GET #1,q;q=q+1;i=i+1
1030 c1=ASC(a$)
```

(下转第3页)

键盘电视游戏机的 FBASIC 语言 及程序设计(下)

于 春

三、FBASIC 语句和程序

先举一个简单的 FBASIC 程序的例子,看如何运用 FBASIC 语言解题。

1、语句与程序示例

设全班某课程的成绩,A 等(100分)的10人,B 等(90分)18人,C 等(80分)21人,D 等(70分)5人,求全班的平均成绩。

解此题可用 FBASIC 语言编出如下程序:

```
10 LET A=10
20 LET B=18
30 LET C=21
40 LET D=5
50 LET U=A+B+C+D
60 LET G=100*A+90*B+80*C+70*D
70 LET F=G/U
80 PRINT F
90 END
```

这是一个完整的 FBASIC 程序,可以看出,它是由若干行组成。一般一行写一个语句,上面9行共有9个语句。也可以一行内写多个语句,不过同一行内语句间要用冒开“:”隔开。

每一个语句让计算机完成一定的操作,若干语句的有序集合就构成一个程序。

一个语句一般可分成三部分:

a、标号 每个语句前,都有一个数字(本例中10~80),这个数字叫行号或语句标号。行号是无符号整数,一般情况下,计算机按行号的大小由小到大执行各语句。行号范围0~65534,不要求连续,一般上下行间都留出一定间隔以便修改补插新的语句。各语句送入计算机不一定按行号顺序输入,送入后,计算机会自动按行号大小顺序整理排列好。

b、语句定义符 它规定计算机执行一定的操作,如上例中的 LET、PRINT、END。

c、语句体 是跟在语句定义符后面的,需要执行的具体内容,如上例中的 A=10等。每个语句不一定都有这三部分,但行号一定

要有。一行有几句,则这几句共用一个行号。每个程序一般以 END 语句结束。只要后面没有别的语句,允许省略 END。

2、赋值语句(LET 语句)

上例中10~70行具有共同的形式:

〈行号〉LET〈变量〉=〈表达式〉

这类语句叫赋值语句。它的含义是:将等号右边的表达式先算出来,然后将其值赋给左边的变量。执行此语句之后,左边的变量就获得了一个新的确定的值。上例中10句执行后,变量 A 得到一个确定的值10;60句执行后,变量 G 得到一个确定的值4650。

说明几点:

a、语句说明符 LET 可以省略。前面说过,一行可以写几个语句,因此,上述例题可以简写成如下形式:

```
10 A=10:B=18:C=21:D=5
20 U=A+B+C+D:G=100*A+90*B+80*C
+70*D
30 PRINT G/U
40 END
```

b、赋值语句中的等号与数学上的等号不同,在数学上 $X+Y=Z$ 表示等号两边完全相等,也可以写成 $Z=X+Y$ 。而赋值语句的 $Z=X+Y$ 是指把 X、Y 的当时值相加后送到 Z 中去而不管当时 Z 已经是多少,此语句执行后 Z 得到一个新值。又如

```
100 X=X+1
```

是经常用的一个赋值语句,它表示把 X 的当前值加1之后送给 X 作为 X 的新值。这在代数学中显然是不成立的。因此要特别注意这一区别。

另外,赋值语句的左边只能是一个变量,不能是多个变量或表达式。如只能写 $Z=X+Y$,决不能写成 $X+Y=Z$ 。同样 $4*Y=20$ 也是错误的。

c、也可以对字符串变量赋值。如

```
10 LET A$="FCS-90 COMPUTER"
```

此句是将右边字符串送入左边串变量中,此时 A\$ 获得新值。A\$ 的新值就是一个确定的字符串。注意字符串都要带上双引号。且字符串只能赋给串变量,数值只能赋给数值变量,不能交叉赋值。

d、赋值语句是用得最多的一种语句,不但可以用

它给变量赋值,更主要的是可以用它完成各种运算。因为赋值语句等号右边的表达式可以表达各种复杂的算术运算和字符串运算,在执行此语句时,先将右边表达式的值计算出来,再赋给左边的变量,所以任何复杂的算术运算和字符串运算都可用赋值语句来完成。如

$$X = \frac{3a}{2c + \frac{b}{5+c}}$$

只要给定了 a、b、c、值,用一个赋值语句就可以完成 X 的计算:

```
10 X=3*A/(2*C+b/(5+c))
```

3、程序的执行(RUN 指令)

上述关于程序和语句的例子,如果没有给以运行命令,电脑不执行,只是把输入的程序存在内存里。电脑内只有一个计算计划,不会产生任何结果。

使电脑执行运算的命令是 RUN 指令。当 RUN 输入后,计算才按程序的要求一步一步执行,直到结束。执行一次 RUN 后,程序并没有被破坏,可以多次执行。

4、打印语句(PRINT 语句)

打印语句的作用是在显示器上显示或在打印机上打印出变量或表达式的值或字符串。

a. 打印语句的格式

〈行号〉PRINT〈输出项〉

输出项可以包含多个变量,表达式及串变量,各量之间用逗号或分号隔开。如

```
10 A=100:B=200
20 A$="HOW ARE YOU"
30 PRINT A$,A,(A+B)/2
40 END
```

输入 RUN 语句后,结果如下:

```
HOW ARE YOU      100      150
```

b. 打印语句的输出格式

〈1〉标准格式:(分区格式)

PRINT 语句中的输出项之间如用逗号分开则输出的格式叫标准格式,或叫分区格式。

显示屏一行28个字符。根据输出项字符的多少,自动分成四个区,三个区、两个区。如果最后一区打不下,多的部分会自动移到下一行第一区。

〈2〉紧凑格式

输出项之间用分号隔开,则输出为紧凑格式。例如:

```
10 A$="GOOD"
20 B$="MORNING"
30 C$="*"
40 PRINT A$;C$;B$
50 END
```

RUN 屏幕显示

```
GOOD*MORNING
```

输出项之间不留空格,紧凑打印。若上例30句改为

```
30 C$=" "
```

RUN 屏幕显示

GOOD MORNING

这是 C\$ 为空格,空格也是字符。

〈3〉单列打印

在循环语句中,若输出项后不打标点,则单列打印。如

```
10 FOR A=0 TO 5
20 PRINT A
30 NEXT
40 END
RUN 屏幕显示
```

```
0
```

```
1
```

```
2
```

```
3
```

```
4
```

```
5
```

〈4〉隔行打印

若在打印语句的上一行或下一行加入 PRINT 空语句,则输出隔一行打印。例

```
10 FOR A=0 TO 3
20 PRINT
30 PRINT A
40 NEXT
50 END
RUN
```

```
0
```

```
1
```

```
2
```

```
3
```

5、列表与清内存语句(LIST、NEW)

一个程序编写完毕或运行后,欲察看程序或进行修改可使用 LIST 语句列出程序表。

LIST 语句是一个直接指令,可不编行号,直接输入电脑执行。LIST 指令可以随意、部分或全部列出行号。其格式为

LIST〔m〕〔-n〕

中括内为任选项。如:

LIST m——列出行号为 m 的那一行程序。

LISTm——列出 m 行后所有程序。

LIST-m——列出起始行至 m 行程序。

LISTm-n——列出 m 行至 n 行程序。

LIST——列出全部程序。

一个程序运行完毕,要进行新的程序设计时,必须清除原有程序(以免干扰新编程序),即须进行清内存。清内存有多种方法,如关机后重新开机。常用的是使用 NEW 指令。

NEW 指令也是直接指令,执行后,内存中所有数据全部清除。

两条指令的使用很简单,不再举例,读者可以自己练习。



驱动器修理一例

张红平

机型:APPLE II 型计算机。

故障现象:主机、打印机、显示器均工作正常只是驱动器无法取出磁盘信号,驱动器上的指示灯亮一两秒钟就熄灭了。

故障分析:驱动器正常情况,指示灯一般要亮十几秒到几十秒(由磁盘内容决定时间长短)后熄灭,显示器显示磁盘内容并提示问题或者让用户选择。根据故障现象,驱动器亮,说明主机电源已送出,只是没有接通控制信号和读写文号,计算机误认为磁盘没有放进去,因此显示器上没有任何磁盘信息。据本人修理多台驱动器所知,大多是人为损坏,很可能是驱动器与主机的20线杆头插反造成主机电源对驱动器信号电路的损伤,驱动器接口插头各脚作用如下:上面一排从1、3、5、7脚为接地线,19、17、15、13脚为正12伏电源线,

下面一排为步进电机A、B、C、D相,10、14、16、18、20脚分别为写令,驱动器启动、读数据、写数据、写保护、和负12伏电源,据此插头的分布情况,上面一排是电源,而下面一排主要为控制信号和读写信号,如果插头插反正好是上面一排电源对驱动器的下面控制电路,就会造成驱动器控制电路损坏。

检修:据以上分析,打开驱动器拆下电路板,仔细观查发现主要控制信号都由集成块74LS125进行转换,因此认为此集成块损坏的可能性是较大的,小心取下此集成块,测试发现区别较大,最后花两元钱换掉此集成块,试机,一切正常,此机修复。

注:驱动器内部电路大多为双面电路,拆卸时最好用两个烙铁两人同时进行,望同行们在遇到类似故障时多加分析故障产生的原因,少走弯路。

维修小经验

张中弦

钟控收音机的时钟集成块 失控故障的检修

有一钟控收音机其时钟集成块是MM5387AA/N,故障是不能定时收音、响闹,当设在睡眠功能时,按住SLEEP(睡眠)键才能收音,一放手就立即自动关机。检测集成块各脚电压,发现各控制输入脚均接近0V,唯有打盹输入脚(第24脚)接近电源正极电压9V,检查打盹按键(SNOOZE)并无短路。原来故障出自集成块内部软击穿,导致打盹输入常通呈高电平,这相当于一直按住打盹按键,使定时收音、闹响及睡眠功能失灵。后经试验用一只150Ω的电阻将打盹输入脚与电源负极相连,使其电平拉低,电路即可恢复正常工作。时钟集成块一般若损坏设置输入部分,可按此法排除,只要适当选取电阻的阻值就可,理论上说阻值越小越好,但阻值过小当按下按键时会使电源电压下降太大。若时钟集成电路块是完全击穿的话则只有更换集成块了。

PC8300 微电脑读取 程序特殊故障的排除

一台PC8300微电脑正常使用时发现键入“LOAD”及程序名,按回车键后,显示器一片空白,插放程序磁带也没有显示及不能读取程序,按SHIFT-BREAK不能暂停读取程序过程,只能按ENTER-RESET键或断电重新启动。检查外围元件并无损坏,经测量发现接口芯片C4005的TPEIN端(第2脚)工作时的电平偏高,估计为C4005内部软击穿导致电平升高所致。后经试验将一只2.2KΩ的电阻接在C4005的第2脚与电源负极之间,故障即消除,使得一块价值五十多元的C4005接口芯片恢复正常工作。

硬磁盘驱动器专用接口电路 501262 的测试

卜建辉

501262 适用于 ST506/412 标准工业接口,使用在美国 MICROSCIENCE 公司的产品 HH-725, HH-825, HH-830, HH-1050 及 HH-4050 等系列硬磁盘驱动器中,由于美国 MICROSCIENCE 公司的硬磁盘驱动器在国内有较多的用户,通过该篇文章,能给用户及硬磁盘驱动器维修人员提供帮助。

该集成电路产生索引,在索引脉冲的同步下能产生伺服区和数据区,同时产生准备好信号寻道完成信号,读写控制,时钟分频,步进脉冲缓冲作用。

1. 引脚功能及定义(见表 1)

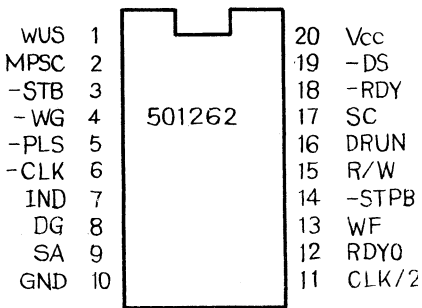


图 1

表 1

WUS	写不安全信号	输入
MPSC	CPU 控制寻道完成	输入
-STB	步进脉冲	输入
-WG	写门	输入
-PLS	索引脉冲	输入
-CLK	时钟	输入
IND	索引	输出
DG	数据区同步信号	输出
SA	伺服区同步信号	输出
GND	地	
CLK/2	时钟二分频	输出
RDY0	准备好信号输出	输出
WF	写故障	输出
-STPB	步进脉冲缓冲输出	输出
R/W	读/写	输出
DRUN	数据运行	输出
SC	导道完成	输出
-RDY	准备号	输出
-DS	驱动器选中	输入
Vcc	+5V	输入

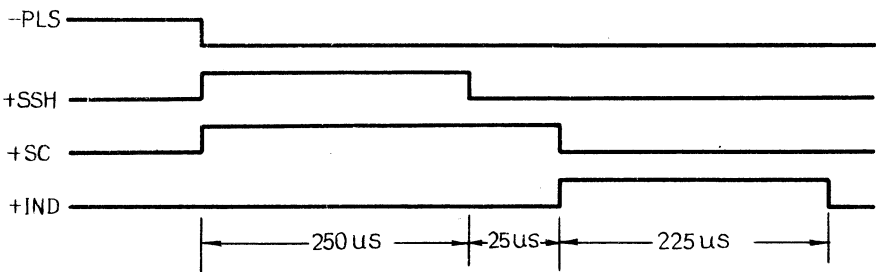


图 2

2. 逻辑关系

$$(1) +RDY0 = (+DS) * (+RDY) * (-WF)$$

$$(2) -STPB = (-STP) \text{ 脉冲下降沿的一个时钟脉}$$

冲

$$(3) +SC = [+MPSC] * (STPNG) * DS$$

$$\text{在这里 } (+STP) * (+RDS) = 1 \text{ 时 } STPNG = 1$$

$$MPSC = 1 \text{ 时 } STPNG = 0$$

$$(4) +R/-W = -((+WG) * (+DS) * (-WFW) * (-DG) * RDY * (-STPNG) * (-MPSC))$$

$$(5) +SA = SSH + (-RDY) + STPNG$$

在这里 SSH 见图 2

(6) +DG 见图 2

(7) +IND 见图 2

(8) -CLK/2 时钟二分频

3. 该集成电路的测试及方法

(2) 测试过程

- 将专用集成电路插入测试仪中的 IC 插座
- 开电源

c. 按 START 开关,开始测试.

e. 测试完后,指示灯亮及显示相应的代码,绿灯表示通过,红灯表示失败.

(3) 测试程序

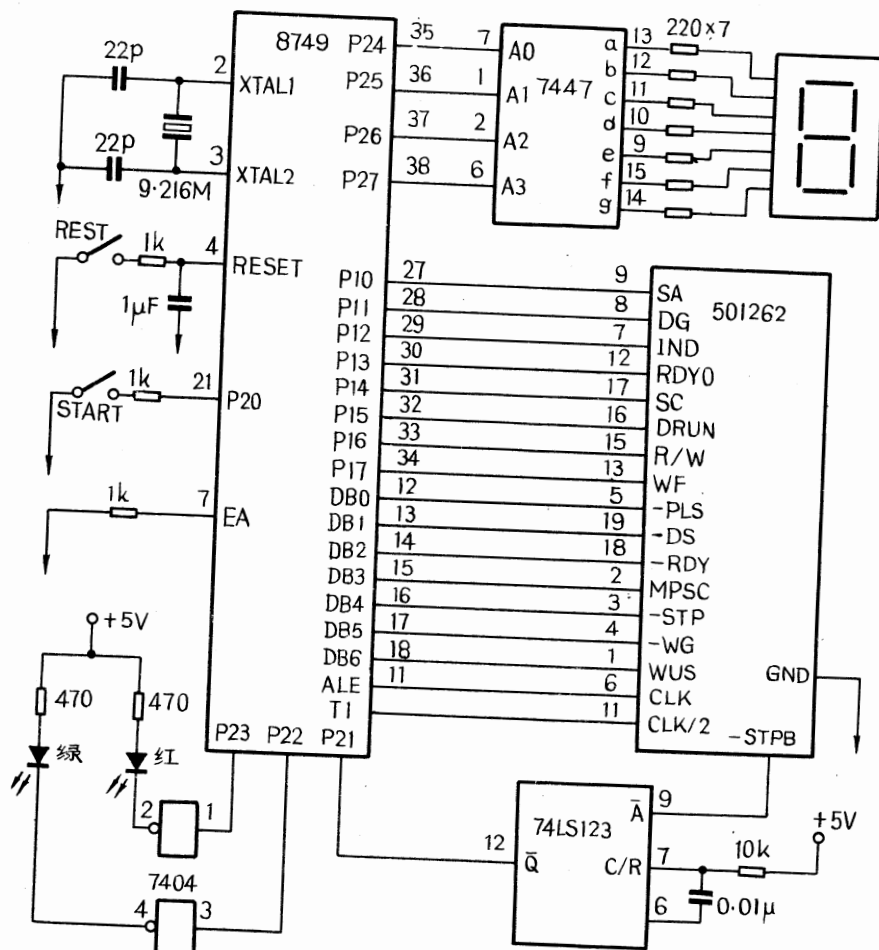


图 3

```

ORG 00H
JMP RESET
ORG 0AH
RESET: CLR A
      OUTL BUS,A
      CPL A
      OUTL P1,A
      MOV A,03H
      OUTL P2,A
      DIS TCNTI
      DIS I
      SEL RB0
      SEL MB0
      ORL P2,F0H

```

```
START:  IN A,P2
        ANL A,01H
        JNZ START
        MOV R2,20H
        CALL DELAY2
        IN A,P2
        ANL A,01H
        JNZ START
        ANL P2,F3H
        MOV R6,00H
        MOV R5,10H
PULSE:  MOV A,FBH
        OUTL BUS,A
        MOV R2,20H
```

```
CALL DELAY2
ANL BUS,FEH
MOV R3,00H
MOV R4,00H
IN A,P1
ANL A,07H
XCH A,R3
MOV R1,2DH
CALL DELAY1
IN A,P1
IN A,67H
XCH A,R4
NOP
IND: IN A,P1
```

3.4 程序流程图

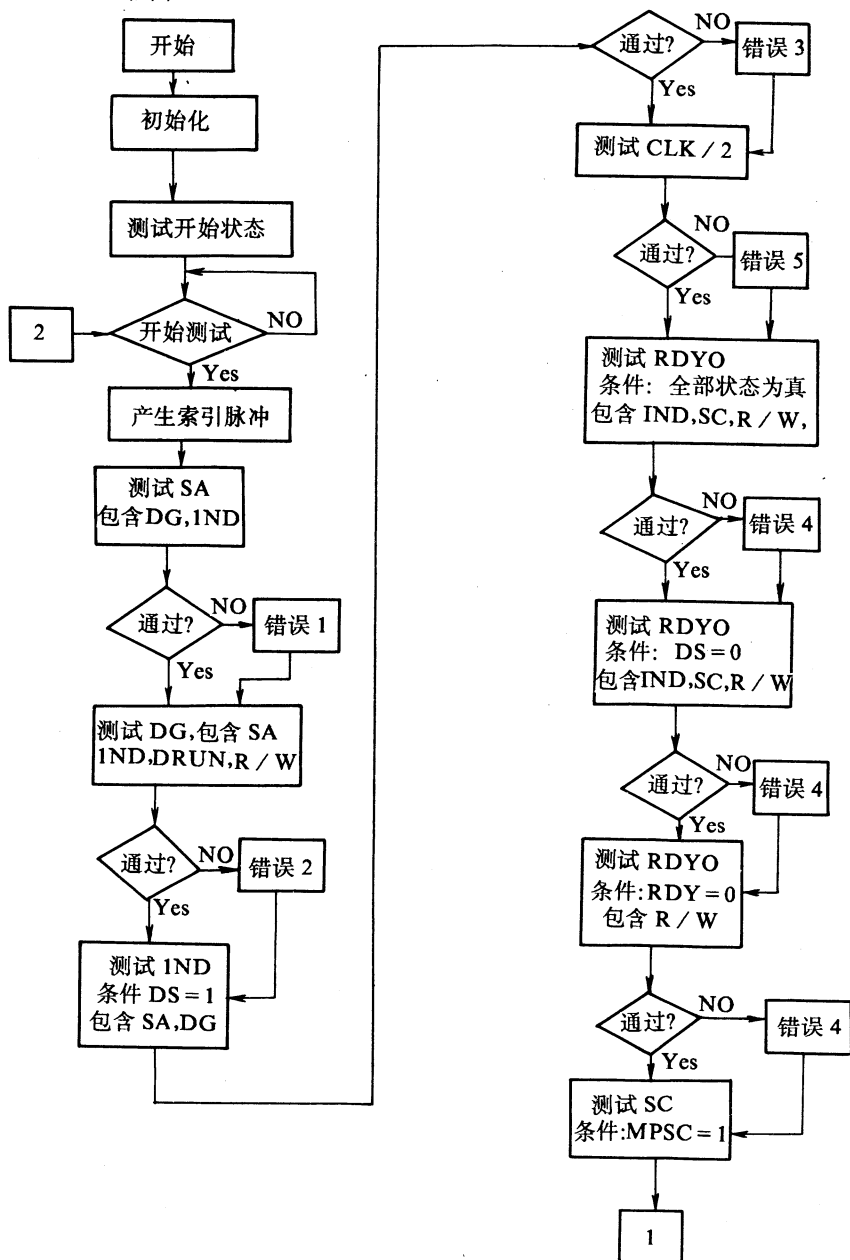


图 4

ANL A, 07H
JZ IND1
CALL ERROR
ANL P2, 3FH
IND1: ANL BUS, FDH
IN A, P1
ANL A, 07H
ANL A, 04H
JZ IND2

CALL ERROR
ANL P2, 3FH
IND2: MOV R1, 23H
CALL DELAY1
IN A, P1
ANL A, 07H
JZ TST1
CALL ERROR
ANL P2, 3FH

TST1: XCH A, R3
XRL A, 03H
JZ TST2
CALL ERROR
ANL P2, 1FH
TST2: XCH A, R4
XRL A, 42H
JZ PCNT1
CALL ERROR

```

ORL P2,F0H
ANL P2,9FH
PCNT1: MOV A,FBH
        OUTL BUS,A
        DJNZ R5,PULSE
CLK2:  MOV A,D0H
        MOV T,A
        STRT CNT
        MOV R1,5DH
        CALL DELAY1
        STOP TCNT
        JTF RDYO
        CALL ERROR
        ANL P2,5FH
RDYO:  MOV A,79H

```

```

        OUTL BUS,A
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,08H
        JNZ RD1
        CALL ERROR
        ANL P2,4FH
RD1:   MOV A,7BH
        OUTL BUS,A
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,5CH
        XRL A,40H

```

```

        JNZ RD2
        CALL ERROR
        ANL P2,4FH
RD2:   MOV A,7DH
        OUTL BUS,A
        MOV R1,10H
        CALL DELAY1
        IN A,P1
        ANL A,48H
        XRL A,40H
        JZ SCT
        CALL ERROR
        ANL P2,4FH
SCT:   MOVA,79H
        OUTL BUS,A

```

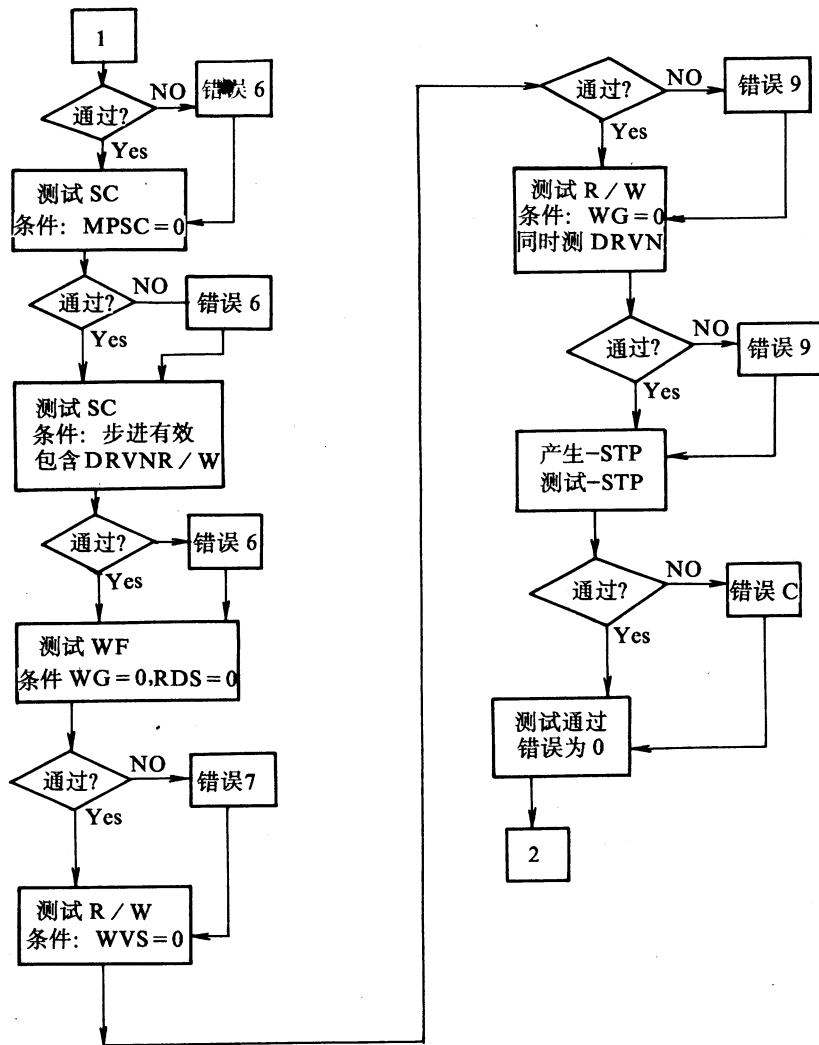


图 5

MOV R1,10H	ANL P2,6FH	IN A,P2
CALL DELAY1	WFT: ANL BUS,5FH	ANL A,02H
IN A,P1	IN A,P1	JNZ STP2
ANL A,10H	ANL A,80H	CALL ERROR
JZ SC1	JNZ SC4	ANL P2,AFH
CALL ERROR	CALL ERROR	STP2: ANL BUS E1H
ANL P2,6FH	ANL P2,7FH	IN A,P2
SC1: MOV R2,60H	SC4: MOV A,79H	ANL A,02H
CALL DELAY2	OUTL BUS,A	JZ STP3
MOV A,71H	MOV R1,50H	CALL ERROR
OUTL BUS,A	CALL DELAY1	ANL P2,AFH
IN A,P1	ANL BUS,77H	STP3: DJNZ R5,STP1
ANL A,10H	IN A,P1	GOOD: MOV A,R6
JNZ SC2	ANL A,10H	XRL A,AAH
CALL ERROR	JNX WRT	JZ AIN
ANL P2,6FH	CALL ERROR	ANL P2 0FH
SC2: MOV A,73H	ANL P2,6FH	ORL P2,04H
OUTL BUS,A	WRT: ANL BUS,BFH	AIN: JMP START
MOV R1,10H	IN A,P1	
CALL DELAY1	ANL A,40H	延迟子程序 1
IN A,P1	JNZ WR1	DELAY1:DJNZ R1,DELAY1
ANL A,30H	CALL ERROR	RET
JZ SC3	ANL P2,9FH	延迟子程序 2
CALL ERROR	WR1: ANL BUS,DFH	DELAY2:MOV R1,FFH
ANL P2,6FH	IN A,P1	DLY: DJNZ R1,DLY
SC3: ANL BUS,65H	ANL A,60H	NOP
MOV R1,50H	JZ TSTP	DJNZ R2,DELAY2
CALL DELAY1	CALL ERROR	RET
MOV A,71H	ANL P2,9FH	错误子程序
OUTL BUS,A	TSTP: MOV R5,10H	ERROR: ORL P2,08H
IN A,P1	STP1: MOV A,F1H	ORL P2,F0H
ANL 70H	OUTL BUS,A	MOV R6,AAH
XRL A,40H	MOV R2,90H	RET
JZ WFT	CALL DELAY2	
CALL ERROR		

电子工业出版社计算机类新书介绍

代号:	书 名	邮购价
B1580	Microsoft Windows V3.0 用户手册	16.10
B1581	Microsoft Windows V3.0 程序设计指南和工具	24.20
B1582	Microsoft Windows V3.0 程序员参考手册	30.50
B1530	计算机环境的可移植操作系统界面 POSIX.1	23.00
B0890	DEC NET 计算机网络的结构与实践	46.00
B1671	现代计算机技术博览(一)	28.80
B1570	UNIX 使用大全	14.00
B1660	UNIX 系统 V 实用指南	12.70

邮局汇款:北京万寿路 173 信箱电子工业出版社发行部邮购科。

银行汇款:

户 名:电子工业出版社销售服务部。

开 户 行:北京市翠微路分理处。

帐 号:661036—40 邮政编码 100036

汇款同时请用正楷字写明书代号、册数。

IBM-PC/XT 一种系统错误的分析

姜金友

在 IBM-PC/XT 及其兼容机上,即使系统有两个 5 英寸软盘驱动器,DIR B:时偶尔也会出现以下信息: Insert diskette for drive B: and press (or strike) any key when ready,其意是在 B 驱动器内插入软盘后再任敲一键,按键后发现 DIR B:执行的就是 DIR A:,以后每次 DIR B:都是对 A 驱动器进行操作。若再执行 DIR A:,又会出现以上提示,只是 B:变成了 A:,让你在 A 驱动器内插入软盘。一旦出现上述情况,以后所有对软盘的文件操作都将会有类似现象发生。这些表现同只有一个 5 英寸软盘驱动器的情况相同。也就是说系统此时认为你的微机只有一个 5 英寸软盘驱动器。发生上述情况后,只要重新启动,一般就不会再出现以上现象。

究其原因,起初还以为是病毒所至,经检查后发现系统并没有染毒。再一分析,发现用 INT 13H 对软盘扇区进行访问,一切正常。因为 INT 13H 是调用 BIOS 对软盘进行访问的,故此可以肯定以上现象只与 DOS 系统有关。

下面再看看 DOS 系统是如何引导的?

开机后,系统首先执行 0 道 0 面第一扇区的引导模块,该模块将 IO. SYS(或 IBMBIO. COM)读入内存,IO. SYS 由以下三个部分组成:

1. 初始化程序;2. BIOS 接口程序;3. 其他服务程序。

其中初始化程序有一项就是确定软盘驱动器数,该数目是由系统配置开关 DIP 所决定的。主机通过 8255A 可编程外围接口芯片读取 DIP 开关的设置,该开关的第七、八两位表示系统连接的 5 英寸软盘驱动器数,其组合如下:

SW-8	SW-7	软盘驱动器数
ON	ON	1
ON	OFF	2
OFF	ON	3
OFF	OFF	4

在 IBM PC/XT 中,8255A 工作于无联络信号的基本输入输出方式(方式 0),其 24 根外围数据线分成三组:PA0-PA7、PB0-PB7、PC0-PC7,BIOS 中使用的 I/O 地址分别为:60H、61H、62H。PC0-PC7 工作于输入状

态,用来读取 DIP 开关的设置。从 62H 口读入数据的第二、三位分别表示 SW-7 与 SW-8 的状态。IO. SYS 就是通过从口 62H 读数据而获得软盘驱动器数的,并将其结果存入内存。经多次查找分析发现,结果存放在内存 0 段某一单元内,该单元的偏移量随 DOS 版本不同而不同。下面是几种不同版本的偏移量:

DOS 版本	偏移量
2. 00	0acfH
2. 10	0ae0H
3. 10	0b2cH
3. 20	1abfH
3. 30	0bd8H
4. 01	0becH

对于 DOS3. 20 以下版本,如果偏移量处数值为 01H,则表示系统只有一个 5 英寸软盘驱动器;如果为 00H,则表示系统有二个 5 英寸软盘驱动器;DOS3. 20 (含 3. 20)以上版本则正好与上面相反,01H 表示系统有二个 5 英寸软盘驱动器,00H 表示系统只有一个 5 英寸软盘驱动器。因此只要改变以上偏移量处的值,系统所识别的驱动器数目就要发生变化,例如对于 DOS4. 01 系统,只要完成以下操作(键入划线部分即可),系统就认为只有一个软盘驱动器。此时 DIR A:与 DIR B:结果完全相同。

```
C)>DEBUG
-e0:bec 00
-q
```

导致本文开头所说情况的原因就是因为基本输入输出模块读取 DIP 开关错误所至,因此如果你的计算机是 PC/XT 型,且带有双软盘驱动器,只要在 DOS 系统引导后,检查该版本所对应偏移量处的值是否为双驱动器所对应的值,若不是就改过,这样即可避免再发生上述错误。

对于 PC/AT 等系列微机,尚未发现上述现象,但若将以上偏移量处的值改为单驱动器所对应的值,也会有上述现象发生。

用非格式化方法修复硬盘一例

西安石油学院 周建辉

我室一台 IBM-PC/XT 微机开机出现死锁现象,用软盘启动也不能联上硬盘,判定为硬盘主引导记录(主 BOOT)坏。因机上有大量科研成果资料,我们决定用非格式化方法来修复此硬盘。

通过分析 XT 机、AT 机和 386 机的主引导记录,发现它只起到一个 ROM-BIOS 和引导 DOS(BOOT)之间过渡作用,主要功能是找到合法硬盘的活动分区并把控制权交给该分区的 BOOT 以引导 DOS。所以主 BOOT 都大同小异,特别是同一容量硬盘在常规使用下更是相似甚至相同。因此我们的做法是利用 DEBUG 将好盘主 BOOT 代替同容量坏盘主 BOOT,具体操作如下:

第一步:拷贝好的主 BOOT 到软盘,在 DEBUG 下输入下述小程序将好的主 BOOT 读入内存 $\times \times \times \times$:0400;再用命令“-W400 1 12 1”将好主 BOOT 写入 B 驱动器上的软盘。如图一所示。

```
A>debug
-a100
21A2:0100 mov dx,0080
21A2:0103 mov cx,0001
21A2:0106 mov bx,0400
21A2:0109 mov ax,0209
21A2:010C int 13
21A2:010E
-g=100 10e
AX = 0000 BX = 0400 CX = 0001 DX = 0080 SP =
FFEE BP = 0000 SI = 0000 DI = 0000 DS = 21A2 ES = 21A2
SS = 21A2 CS = 21A2 IP = 010E NV UP EI PL NZ NA PO
NC 21A2:010E 47      INC      DI
-w400 1 12 1
```

图一 拷贝好的主 BOOT 到软盘

第二步:将好的主 BOOT 写入坏盘。用软盘启动已坏的机器,在 DEBUG 下用命令“-L400 1 12 1”将 B:驱动器上软盘中的好主 BOOT 读入内存 $\times \times \times \times$:0400 位置;再用下述小程序将好的主 BOOT 写入坏盘,即完成修复工作。如图二所示。

我们用上述办法恢复了一个 20M 硬盘,其上的所有数据完好。事实证明,只要对 PC 机引导机制及主 BOOT、BOOT 工作过程进行分析,就可以许多情况下无须通过格式化就可修复硬盘。

```
A>debug
-
-1400 1 12 1
-
-a100
21A2:0100 mov dx,0080
21A2:0103 mov cx,0001
21A2:0106 mov bx,0400
21A2:0109 mov ax,0301
21A2:010C int 13
21A2:010E
-g=100 10e

AX = 0000 BX = 0400 CX = 0001 DX = 0080 SP =
FFEE BP = 0000 SI = 0000 DI = 0000 DS = 21A2 ES = 21A2
SS = 21A2 CS = 21A2 IP = 010E NV UP EI PL NZ NA PO
NC 21A2:010E 47      INC      DI
```

图二 将好的主 BOOT 写入坏硬盘

上面操作是在双软驱情况下,但此方法也可在单软驱情况下进行。另外分区表信息可以 DOS 版本和硬盘容量进行重构;但分区表信息还与硬盘使用情况(如是否分区,怎样分区)有关,因而作一个主 BOOT 备份是有益的。

检修 IBM AS/400 B35 故障二例

王志远

(一)IBM 5337 激光打印机故障检修

故障:打印出来的字体拉丝

原因:在感光鼓上残留有纸片,未被卡纸传感器检出。

分析:IBM 5337 激光打印机是一种电子照相式打

印机。它的工作原理是:用充电器使 OPC 鼓表面带电,被调制的激光束打在鼓表面,使鼓表面电位发生变化,产生不可见的潜象。显象部件的墨粉被吸入鼓表面被激光照射的部分,从而在鼓上产生映象。转写器将鼓上的墨粉吸到打印纸上,经喷光定影,将映象加热并熔于纸上。当鼓上残留有纸片,而未被卡纸传感器检出,打印机不产生错误信息。而残留的纸片象一把刷子碰触在转写后的打印纸上,因为此时打印出来的字体还

(下转第 29 页)

自己学修计算机

陈 勇

随着近几年来微型计算机的不断普及,微型计算机的维修工作日趋变得重要起来。微机一旦出现故障,往往会耽误工作,如果请人维修还会花费大量金钱和时间,如果我们微机使用者本人能够学习一些微机简单故障的修理知识,那么不但能够丰富自己的计算机知识而且还能节约大量请人修机所需的时间和金钱。本文仅以 IBM PC 机为例作一介绍,并愿与计算机同行们共同切磋探讨。

一、IBM PC 配备的两种故障辅助诊断功能

每一台 IBM PC 机都配有两种故障辅助诊断功能,以帮助维修人员确定引起故障的可能原因,第一种辅助诊断功能是开机后的自检过程,这一过程叫作开机自检(POST)。第二种辅助诊断功能是一种诊断软件,一般与随机配置的操作手册说明书放在一起,较之第一种辅助诊断功能,这一诊断软件能对计算机的各种系统、电路板和外设进行更广泛的测试。

两种辅助诊断功能的不同之处就在于:两者的测试范围不同。后者比前者测试的范围更广一些。IBM 公司没有为用户提供充分的有关如何使用这两种辅助诊断功能的资料,也没有提供一些简单的错误信息,即,没有像“你的随机存/取存储器坏了”这样简单明了的信息,而只有用错误代码表示的信息,比如,“03400201”,这看起来很不直观,但是,一旦我们查清代码的含义,那么我们就不但能够知道出的是什么故障,还能知道故障的确切位置。另外,由于开机自检所产生的错误代码与诊断软件所产生的错误代码是一样的,所以只要我们查清一种代码含义,就可以知道另一种的含义。

利用这两种检验手段,在配合以其他一些检验手段和识别故障的方法,我们就可以检验出所有微机故障的百分之九十五。

· 利用开机自检功能(POST)维修计算机

每次开机后,微机的开机自检功能首先对内部的主要系统进行一次快速检查,所需时间依 RAM 容量的多少从几秒钟至三分钟不等,在接下来的期间内,POST 对系统主板、RAM、电源、键盘、外部机架和一些适配器插卡进行检验。在此期间,如果 POST 没有发现故障,那么微机屏幕上将不会有任何显示,同时计算机也不会有任何反映,如果,POST 发现了错误,它将用故障信息代码以及可看到的信息向你发出信号。

每当我们打开计算机,总会听到微机喇叭发出几声“嘟嘟”声,以及软硬磁盘动作时发出的声音,这些声音都是在开机后装入自检程序时发出的,而且,这些声音是有规律的。但往往我们并不在意这些声音,岂不知这些声音对帮助我们发现问题是很有用的。我们都可

能有过这样的经历:打开计算机后,开始我们的工作之前,我们发现微机发出的声音与往常有所不同,但由于计算机仍然能正常引导启动,所以,这些异常的声音并未引起我们足够的重视,于是我们就继续下一步的工作了,然而,当过了一段时间,微机出现死锁或我们编写的程序不能正常存盘时,我们才发现微机出现了故障,并认识到开机时那些声音的重要性。对于 IBM PC 机来说,如果开机后一切工作正常,那么我们会看到微机有以下一系列的反映:光标闪烁几秒钟,至到 POST 结束为止。接下来,会看到 A 盘驱动器的指示灯亮起来,并可听到一声短促的“嘟”声,随之 A 盘驱动器的指示灯一闪之后就又熄灭了。当 A 盘驱动器的指示灯再次亮起来之后,微机开始装入引导程序。如果,有任何不同于上述情况的现象发生,则说明微机发生了故障。

最初,IBM PC 本想为微机配备盒式磁带机构,用于装入和存储程序,虽又立即又放弃了这种打算,不过微机内仍保留了磁带机接口,由一种长驻内存的 ROM BASIC 来管理磁带机。有时当你试图装入一个程序时,你会发现你所面对的确是 BASIC 的画面,这说明 ROM 已经抢先把磁带 BASIC 引导入内存。

当我们所要装入的程序能正常装入时,我们会听到软盘驱动器发出声音,否则,微机会自动引导磁盘 BASIC。

表 1 列出了 POST 所有的错误代码和错误提示,其中 X 代表一位数字,比如,屏幕上并不会出现“XXXXX201”这样的内容,而只会出现现象“0340201”这样的信息,“0340201”这一代码的后三位数字“201”代表 RAM 有故障,而这一代码的前四位数字则代表前排右起第二个 RAM 模块出现了错误。

表 1 POST 的故障信号和代码表

信号和代码	故障所在位置
无任何现象 (nothing happens)	电源或其他
连续的“嘟嘟”声	电源
重复发出短“嘟”声	电源
一声长“嘟”,一声短“嘟”	系统板
一声长“嘟”,二声短“嘟”	监视器
无显示	监视器
磁带 BASIC 显示	磁盘驱动(常常是 A)
101, 131	系统板
201	RAM
XXXXX201 和同位核对 X	RAM
同位核对 X	电源
301, XX301	键盘
601	软盘驱动器
1701	硬盘驱动器
1801	扩展槽

值得一提的是,一些提示严格地讲并不是 POST 的一部分,比如,“nothing happens”这一既不是错误代

码,也不是磁盘 BASIC 的有关显示,但它的出现表明微机有故障存在。有些故障信息是以声音的形式表示的,有时伴随声音的出现也会有错误代码。

当微机提示出错误信息或代码后,就要用诊断盘进一步检查,在微机尚未配备诊断程序的情况下,我们可以对微机进行一些简单的检查,比如,当我们接通计算机电源的时候,计算机没有任何反映,很可能是计算机并未加电,造成这种故障的原因可能在以下几个地方:墙上的电源插座坏了;导线可能断了;微机内部电源可能出了故障;微机内某个地方短路引起电源断电。

• 利用诊断盘检修计算机

我们往往只在微机出现故障时才使用诊断盘,这是不对的,我们应当经常使用诊断盘对微机进行检查,以便极早消除一些尚未造成大错的隐患。诊断盘能对微机进行更全面的检验,它还能进行多功能测试,这一点是非常重要的,因为 POST 或单项检验无法对一些模块工作时好时坏这种故障进行检验,而使用诊断盘的多功能测试,便会迎刃而解。

要使用诊断盘,首先把诊断盘插入 A 驱动器,然后从新启动微机(同时按下 Alt, Ctrl, Del 三个键);或关闭计算机等候 5 秒钟后从新打开计算机,随后便是前面讲过的微机正常启动所发出的有规律的声音和喇叭发出的“嘟嘟”声。在 POST 运行完毕后,驱动器 A 的指示灯会亮起来,扬声器也会发出一声短“嘟”声,随后是驱动器装入程序读磁盘时发出的声音。

在使用诊断盘时,我们可以把故障存入磁盘,当然这是以你的微机和软盘驱动器能正常工作为前提条件的。

一般我们是把错误信息在打印机上输出。

当执行诊断程序时,诊断盘首先检验软盘驱动器,当要执行诊断程序时,微机会停下来并且会发出一声长“嘟”和一声短“嘟”,此时,你应当用一张存有无用信息的软盘代替诊断盘,这是因为执行诊断程序时会破坏盘上的信息。

表 2 列出了诊断盘所产生的初始错误代码、信号、提示信息。与诊断盘所产生的同类内容相比,诊断盘能产生某一部分系统或电路工作正常的信息,而 POST 确不能,这些表示工作正常的代码都是以 0 结束(比如系统板工作正常的代码是 100,软盘驱动器工作正常的代码是 600)

当你的微机监视器出现故障时,你可以通过诊断盘产生的代表不同错误信息的声音对微机进行检验,在有关操作手册上给有测试和屏幕顺序的信息(见表 3),但没有与各种声音相对应的错误信息,所以你应当先运行诊断程序,并记录下相关信号的出现和意义。

二、找出功能故障

找出功能故障并不象你所想象的那么难,只需比消除一个故障多知道一些信息。引起某一功能故障的原因常常有多种可能。

消除一个故障往往需由简单到复杂一步一步地进行检查。比如,你的微机完全不能工作了,你就要对微

表 2 诊断程序代码

代码	故障位置
02X	电源
1XX	系统板
20X	RAM
XXXX	RAM
XX20X	RAM
30X	键盘
XX30X	键盘
4XX	单色显示器
5XX	彩色显示器
6XX	磁盘驱动器
7XX	8087 算术协处理器
9XX	打印机适配器
11X	同步通信
12X	同步通信
13XX	游戏适配器
14XX	打印机
15XX	同步数据链路控制通信
17XX	硬盘驱动器
18XX	扩展槽
20XX	二进制同步通信适配器
21XX	二进制同步通信适配器

表 3 诊断盘在以声音形式进行提示情况下的运行过程

1. 微机加电;在 POST 期间光标闪烁;短的“嘟”声;装入诊断程序
2. “对选择项进行选择”屏幕(0——运行诊断程序;1——格式化磁盘;2——拷贝磁盘;9——退回到系统盘);键入 0 运行诊断程序
3. “配制设备”屏幕;“嘟”声;键入“Y”
4. “诊断功能选择”屏幕(0——运行一次诊断程序;1——运行诊断程序多次;2——使用记录功能;9——退出诊断程序);键入“0”或“1”;程序开始运行
5. “键盘”(键盘测试)屏幕;按回车键;短“嘟”声
6. “显示属性”屏幕;键入“Y”;短“嘟”声
7. “设置字符显示方式”屏幕;键入“Y”;短“嘟”声
8. “80×25”显示;键入“Y”;短“嘟”声
9. “40×25”显示;键入“Y”;短“嘟”声
10. “320×200 图形显示方式”;键入“Y”;短“嘟”声
11. “640×200 图形显示方式”;键入“Y”;按任意键 8 次,短“嘟”声;键入“Y”
12. 一声长“嘟”,一声短“嘟”之后将要测试磁盘驱动器,把一存有无用信息的软盘插入 A 驱动器;按回车键;驱动器 A 的指示灯亮,驱动器也发出声音;两声“嘟”声;键入“Y”
13. 对驱动器 B 重复 12 中的过程;不要在意 RAM 驱动的结果
14. 稍等一会儿,一声短“嘟”表示诊断程序的结束

机内外进行检查,首先你要检查墙上的电源插座的电源是否正常,再检查微机电源是否与插座接触良好。然

后,把微机外的设备都去掉,比如象电源接线板之类,而把微机直接与墙上的电源插座相连,这样可以检验故障是否出在微机以外设备上,(微机内部的开关大部分是自检的,风扇能转动,说明微机已经加上了电,否则,可能是电源或开关出了故障),接下来,对微机内部电源和电路进行检验,你可以使用电压表对微机电源进行测试,也可以按下列步骤进行:先把与微机电源相连的所有器件和电路都断开,看此时,电源是否定有电压,如果有,则说明是某一与电源相连的器件出了故障,我们再每次一个地把断开的器件与电源连接起来,当电源与某一器件连接后,出现故障,则说明此器件出了故障。需要注意的是,每次断开器件或连接器件时,都要把电源关闭。

上述检验故障的思想方法是:先对故障原因做出两种可能的推测,然后消除其中之一,则证明是另一种可能原因造成的。这种检验故障的思想方法也适用于其他一些故障检验,比如,某一程序不能正常装入,引起这种故障的可能有两种:或是磁盘驱动器出了故障,或是程序的故障;若是磁盘驱动器的故障,则引起磁盘

驱动器故障的原因又有两种:或是驱动器本身,或是微机内部的电路板。把装有程序的软盘放在另一个软盘驱动器 B 上,如果,装入成功了,就说明是 A 驱动器出了问题,否则,说明与驱动器 A 相连的插卡或系统电路板出了故障。

三、引起计算机故障的其他一些原因

有时对计算机不正确的操作以及对所用软件使用不当,也会造成计算机功能故障。

比如,某位操作员把一块多功能电路板插入计算机插槽中,当他试图使用它时,计算机出现死锁现象,而当他把此插卡拿掉时,计算机仍然不能恢复正常工作。这是为什么呢?原来,当他插入新的插卡时,实际的通信接口变为三个,而计算机只能管理两个通信接口,这就使计算机发生混乱。另外,由于他插入新的插卡改变了系统板上的开关设置,即使去掉了插卡,设置仍指向去掉的插卡上的 RAM。所以微机仍然不能运行。

总之,只要一步一步仔细检查,对于大部分计算机故障我们自己是完全可以修理的。

TEC—I 机磁盘驱动器 I/O 接口板故障检修一例

崔时珍 玄勇活

笔者最近检修了一台 TEC—I 机(苹果机)的磁盘驱动器的 I/O 接口板,收集了一点经验,现介绍如下,愿与同行交流。

故障现象:

本机在不连接磁盘驱动器的 I/O 接口板时,能进入固化 BASIC 状态。但是,连接之后,屏幕上显示长方形大白块,不能启动磁盘驱动器。

故障分析及故障排除:

磁盘驱动器的 I/O 接口板是由并串行转换电路,多路器,马达控制电路,磁盘驱动器选择电路等组成。并串行转换电路,一方面,负责把磁盘驱动器中读出的串行数据转换成并行数据,再送入主机;或者,把主机送来的并行数据转换成串行数据,再写入磁盘驱动器。另一方面,根据从磁盘驱动器中读出的状态信息或从主机中送来的控制信息,向多路器送工作控制信息。多路器依据工作状态,把主机送来的启动马达和选择磁盘驱动器的信息,再送到马达控制电路,启动马达,同时对磁盘驱动器进行选择。

为此,笔者首先检查了磁盘驱动器选择电路的两个输出端 ENBL1 和 ENBL2(74LS132 又输入四与非斯密脱触发器的 6 脚和 8 脚),发现两个输出端都为高电平,所以,两台驱动器都不能启动。正常时,6 脚首先为

低电平,接通 1 号驱动器。经过对该集成电路逻辑分析之后,确定未坏。为此,顺着此端向上找,发现多路(74LS259 带锁存器的三位译码/选择器)的输出端 9 脚为低电平,工作控制输入端 14 脚为高电平,都不正常。正常读写时,14 脚应有图①所示的脉冲信号,输出端 9 脚应有高电平。

这时,有几种可能性,或者多路器损坏,或者前级某元件损坏,或者包括多路器,前级元件都损坏。为先排除第一种可能性,拨掉了多路器之后,检查了 74LS323(八位双向通用移位寄存器)控制输出 2 脚,发现仍然是高电平。再检查其它管脚,发现 6 脚输入/输出端波形也失常,脉冲幅度只有 1V 左右,再拨掉该集成电路之后,测 6 脚管座的脉冲是正常的。由此可以确定 74LS323 已损坏。更换之后再开机,故障消除了。

特别注意,绝对不允许开机的情况下,插拨接口板或集成电路,否则,会损坏元件。

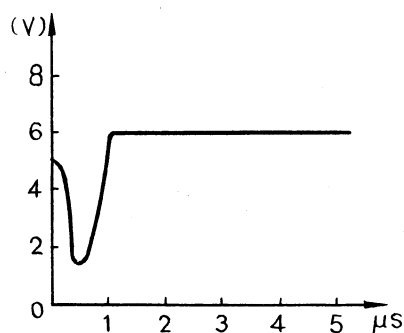


图 1

3070 打印机断针定位 方法及 8088 汇编程序

龙兵生

本文针对紫金 3070 24 针汉字打印机断针问题,介绍一种用来测试断针位置的简易方法,并给出一个使用 8088 汇编语言编制的测试软件。该软件只要修改一个字节,即可用来测试其它型号的打印机断针定位问题,故具有较强的通用性。

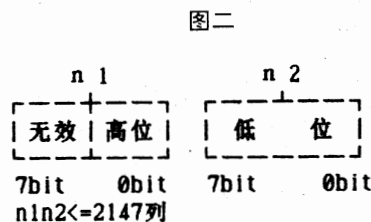
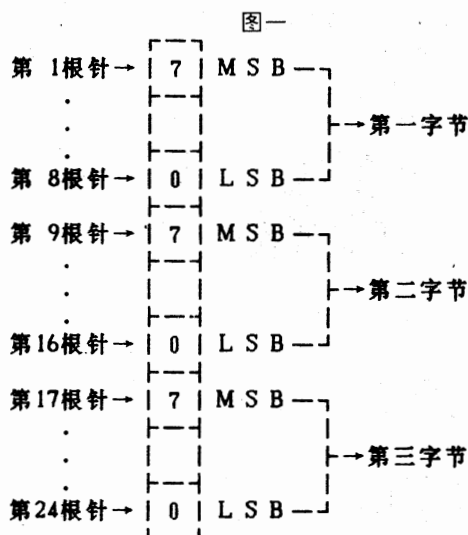
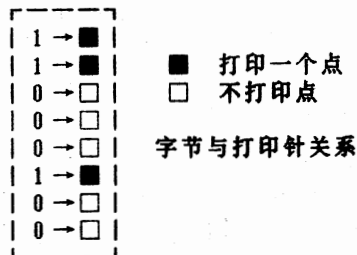
计算机外部设备之一的打印机,由于打印针具有一定的使用寿命,用户使用一段时间以后,或多或少都会遇到打印机发生断针现象(从打印输出的图文信息即可确认)影响打印输出字符的美观。为迅速进行断针的更换,需确定断针的准确位置之所在,这虽然可通过人的肉眼对打印信息和打印头部位进行观察得出答案,但准确性较差且需反复校验,费时费力。为此,笔者根据自己在工作中碰到的这类问题及解决方法,以紫金 3070 24 针汉字打印机断针为例,介绍一个简易的测试方法和程序,断针测试的准确率达 100% 且一次完成。其它类型的打印机断针定位问题,只需将本文提供的程序作一字节改动,即可用来进行断针测试。

测试打印机的断针之位置,基本原理就是设法直接控制打印机的某一根针来打印一条直线,据此观察所测试的针是否可以打印出直线来。若能够打印出直线,那么可以确认该针完好无损,打印不出直线来,那么该针已经打断。记下该断针的编号,然后用一根好针去替换(打印针的编号顺序我们规定为:最上面的一根针为一号针,其次为二号针、三号针……最下面的一根针为二十四号针),恢复该位置针的打印功能。为了实现直接控制打印机一次只出一根针打印信息,需要了解打印机 24 根打印针与打印信息字节的对应关系。3070 打印机的打印针与打印字节的相互关系是这样的(如图一所示):

图象数据为“1”时,对应的针打印一个点,为“0”时对应的针不打印。与打印头对应的数据分三字节传送(如图二所示),第一字节的最高位对应第一根针,最低位对应第八根针;第二字节的最高位对应第九根针,最低位对应第十六根针;第三字节的最高位对应第十七根针,最低位对应第二十四根针。图象数据的传送列数用 $n1, n2$ 表示,如图三所示, $n1$ 和 $n2$ 不能同时为零。打印点列数 N 与 $n1$ 和 $n2$ 的关系如下: $N = 256 \times n1 + n2$ 。

了解了打印信息与打印机 24 根针的对应关系以后,我们可以向打印机发送控制码序列 $ESC + 49H + n1 + n2$,将打印机设置成图象打印方式,从程序中对打印信息字节的内容,根据所选定的针号进行有目的的控制。

制,一次只出一根针来打印 768 个点组成的直线(在程序中我们取 $n1 = 3, n2 = 0$,得到打印列数为 $N = 256 \times n1 + n2 = 3 \times 256 + 0 = 768$)。例如,将 3 字节数 128, 0, 0 传送给打印机后,第一根针出针打印一个点,其余 23 根针不动作,连续打印 768 次,即可打印出由第一根针打印出来的一条 768 点组成的直线。直线的点阵长度用户可以取其它的值,以打印出来的直线便于观察为准。



用户将本文提供的 8088 汇编语言源程序 DAJC.ASM 用字处理软件键入计算机,经编译(MASM DAJC;〔回车〕),连接(LINK DAJC;〔回车〕)以后,生成可执行文件 DAJC.EXE。此后,用户需要检测 3070 打印机断针位置时,只要在操作系统提示符下键入:

A>DAJC〔回车〕

程序运行后,屏幕提示“检测第几号针(1—24),按其余键程序结束)?”用户可输入 1—24 中的任一数字

来检测相应的打印针打印一条由 768 个点组成的实线条,并在直线打印完后自动打印出第 XX 根针打印字样,以便于用户记下断针位置。若用户键入 1—24 以外的其它键,则程序自动识别用户意图而结束测试,返回 DOS。

打印机故障维修—— 以 STAR 或 GX-15 打印机为例

王威 王侯

一、可以自检,但不能与主机连:

I_{c7} (74LS05), I_{c20} (74LS32), I_{c3} (74LS374)

I_{c18} (74LS14)

二、有自检现象,但不走纸

1. 走纸马达坏,检测方法:按 LF, FF 看其是否走纸,如走则是好的,否则是坏的。

2. 如走纸马达是好的,则走纸马达控制电路有问题,查一下 TA2, I_{c22} (74LS05)

三、开机能自检,但打印乱字符:

1. 看一下打印机连线是否插好或损坏。

2. 用橡皮打并行接口擦一下,再查一下 I_{c19} (74LS374)

四、打印机有自检功能,但乱打空字符

1. 打印头控制电路不好,则查一下以下片子: I_{c16} , I_{c18} , I_{c8} , I_{c21} , I_{c22} , I_{c17} , I_{c1} , TR_3 — TR_{11}

2. 如以上没问题,查一下打印头,看针是否有损坏的

针控端

97531 +5V 2468

各针控制端与 +5V 端为 7Ω ,或电流为 600mA。

3. 如以上都没问题,则 2764ROM 坏

五、开机有自检动作,但打印不成形的字符(缺点少划)

1. 打印头本身坏,或有的针被堵塞。

2. 2764ROM 字符库坏

六、开机有自检动作,但打印头不向右移。

1. 行打印马达坏:检查方法:关机,把头向右移到头,开机,如打印头回到左侧则好的,否则马达坏。

2. 行打驱动电路坏:查 TA1, I_{c23} 。

中国计算机学会 联合举办“学装微电脑”函授班 电子工业出版社

第四期招生简章

近几年来,我国的计算机普及教育活动有了很大的发展,许多人学习了计算机语言,甚至具备了编制程序的能力。但是,如何进一步利用计算机的接口电路作一些开发应用,以及计算机出了故障敢于动手修理,许多人仍缺乏硬件方面的知识。我们办函授班的宗旨是:在广大计算机爱好者中培养具有一定计算机硬件基础知识和动手能力的人才。

一、招生对象

具有中等文化程度的微电脑爱好者、中学师生、各行业的技术人员、计算机维修人员,也为解放军培养两用人才服务。

二、教学计划与学习要求

每期函授班为四个半月(每周 6 学时,分为三个单元授课)。

要求学员按时完成每个单元的学习,将作业和实验报告寄回,提出学习中遇到的疑难问题。教师批改作业,评定成绩,解答问题,复信给学员。

学业结束后,根据学员的作业总成绩,经考试合格者,由中国计算机学会发给“全国学装微电脑函授学习结业证书”。

三、招生日期

截止至 1992 年 1 月 30 日

四、教材及学费

全部学费 275 元,包括:MP-1 型学习机全套元器件(以 6502CPU 为中心的 IC 电路 21 块,元器件 102 个,电源一只,印刷电路板一块,实验板一块),实验指导书和补充讲义等教材(已购到 MP-1 学习机者交 60 元)。

五、报名办法

报名者可写信到中国计算机学会办公室(地址:北京 2704 号信箱,邮码 100080,联系人:宁伟成)索取“学员登记表”。学员填好后连同学费一并交齐,函授班将发出“录取通知书”,由电子工业出版社寄发套件、全套教材。

普及型 PC 机上可用的 印刷电路板辅助设计 软件 smARTWORK

张保田

八十年代以来出现了微机上使用的印刷电路板辅助设计软件(CAD), smARTWORK 是常用的一种。该软件对微机的要求低, 便于掌握使用, 可绘制双面印板, 具有自动布线功能, 所绘制的底图可直接照相制板。

硬件要求和运行环境:

主机: IBM—PC、PC/XT, 国产 05 系列, 各种 286、386 等, 要求 256KB 内存。

显示系统: smARTWORK 工作在 CGA 方式, 所以要有彩色图形适配器和彩显, 可同时显示印板的两面。在单色适配器配单显情况下, 如 SUPER—PC 和 HH—PC, 可先运行 MKF.EXE 文件仿真 CGA 模式, 再使用 smARTWORK 软件。

输出设备: 可选用 FX—100 针式打印机或 DMP 系列绘图仪。可打印绘制 1:1 图纸用于校对或 2:1 图纸用于照相制板。

软件要求 PC—DOS 2.0 以上版本支持及 smARTWORK 盘两张。

软盘上的 EDIT.EXE 文件是“自动布线编辑软件”, 运行后有两种工作方式, 即布线和命令方式。布线的主要控制键是:

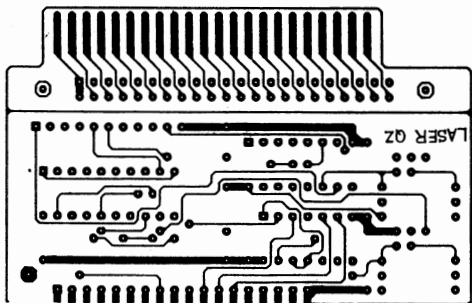
箭头键控制光标上下左右运动, 光标每移动一个步距相当于实际尺寸 1.27mm(0.05 英寸), 使用数字键可快速移动光标。

F3—置焊盘;

F4—清除焊盘;

F1—自动布线, 光标在起点时按一下 F1, 再把光标移到终点按下 F1, 就在起、终点之间自动布线, 以最短路径为标准进行选择。

F2—清除连线;



F5—F7 粗细线转换, 粗线为 1.27mm(0.05 英寸), 细线为 0.3mm(0.012 英寸)

F8—重复布线, 在布一组相同走向尺寸的连线时可重复使用 F8 键。

ALT/F1——单显方式, 只显示印板的一面。

ALT/F7——显示全图;

ALT/F9——显示网格;

其他 ALT 和 F 组合键都用于彩显选择色彩。

击回车键转入命令方式, 是常用的几个命令是:

HELP——帮助, 屏幕上显示 smARTWORK 的功能菜单, 共三页。

SAVE 文件名. 扩展名——把当前板图存入磁盘。

LOAD 文件名. 扩展名——调入板图。

smARTWORK 的功能很多, 本文介绍的只是最主要的部分。附图是用该软件绘制的 LASER 打印卡的印刷板底图, 现已缩小为 1:1, 可以看出是相当漂亮的。

近年来出现了功能更强的 CAD 软件包, 如 TANGO、E. ESYSTEM(电子工程), 但它们对机器要求较高, 有兴趣的读者可参阅有关专业资料。

显示器故障检修一例

张智

故障现象:

一台长城 0520 计算机, 打开显示器光栅正常, 打开计算机后屏幕全黑, 无任何显示, 几分钟后, 屏幕隐约可见字符, 半小时后屏幕显示正常。

故障检查与分析:

将显示器与同型号显示器交换后, 故障排除, 说明故障出在显示器上, 据电器修理经验, 怀疑该故障为显象管座受潮, 导致聚焦极对地有放电现象, 使聚焦电压远低于正常电压, 所以, 在开机时屏幕上无任何显示, 随开机时间的增长, 机内温度升高, 显象管座内的湿气逐渐蒸发, 聚焦极对地之间的绝缘电阻增高, 使屏幕显示正常。

故障排除:

拔掉电源, 打开显示器机壳, 小心拔下显象管座, 向管座封装间隙内注满乙醇, 一分钟后倒掉乙醇, 用电风吹干后, 插上管座, 开机故障仍存在, 再次拔下管座, 从电路板上焊下显象管座, 用小刀和小平口螺丝刀, 将管座的密封间隙塑料盒撬开, 发现密封盒内的聚焦极受潮严重氧化, 产生绿色氧化物, 盒内四壁表面吸附有杂质, 用小刀仔细刮除氧化物及杂质, 用乙醇清洗后, 在白炽灯上烘干, 再在盒内表面涂上一层硅脂, 装上密封间隙塑料盒, 用 502 胶水密封, 装好显象管座, 故障排除。

通过以上故障的检修, 充分说明, 维护好机房洁净, 经常除尘除湿, 是一个不可疏忽的问题。