

1992



電子

ISSN 1000-1077

1000-1077

475

與電腦



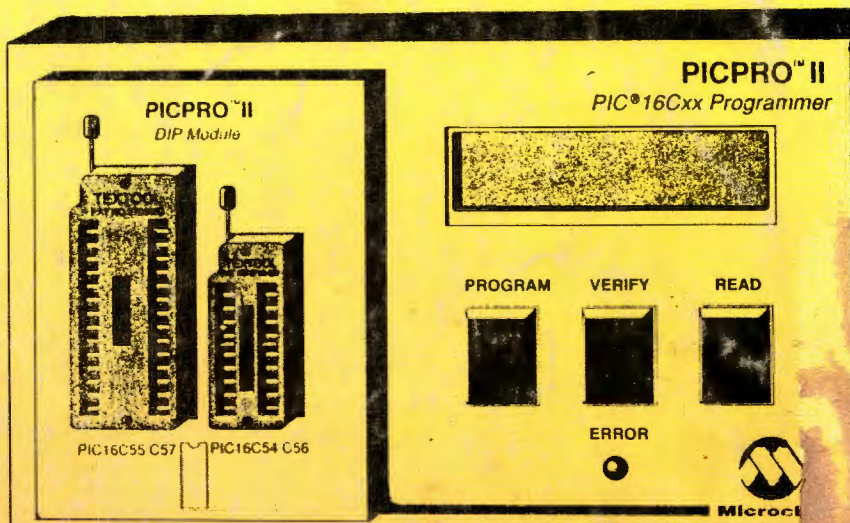
● ELECTRONICS AND COMPUTERS ●



中国电子器材深圳公司经销
美国Microchip公司 PIC16C5X 系列
PIC17C42 系列
单片机仿真开发系统

欢迎查询

欢迎选购



PIC-ICE™	PIC IN-CIRCUIT EMULATOR SYSTEM	开发系统及配件	29	—
PICPAK-I™	PIC Applications Kit	开发系统及配件	71	—
PICPRO-I™	PIC Programmer	开发系统及配件	71	—
PICALC/PICSIM	PIC Cross Assembler/Simulator Software	开发系统及配件	40	—
PICICEEM	PICPAK-I to PIC-ICE Conv. Upgrade	开发系统及配件	239	—
PICICEPOD-08	PIC-ICE Emulator Pod ONLY	开发系统及配件	228	00
PICICECARD	PIC-ICE PC Host Interface Card	开发系统及配件	2340.00	—
PP2DIP	PICPRO-I DIP SOCKET MODULE		990.00	792.00
PP2SOIC	PICPRO-I SOIC SOCKET MODULE		1170.00	936.00

地址：广东省深圳市振华路414幢(华发北路口)电子器材大厦
电话：354214 354345 350877 传真：350876
邮编：518031 电挂：8410
银行：工商银行华强分理处 帐号：22106500376
本公司上海经营部
地址：上海天目西路488号长安大厦商场部
电话：3173798 电挂：2685 邮编：200070

中国电子器材深圳公司

隆重推出美国 Microchip 的仿真器及 PIC16C5X 系列的单片机

型 号	名 称	封 装	单 价 片/元			
			1~49	50~99	100~999	1k~5k
PIC16C54—RC/P	MCU, 0.5K, CMOS, EPROM	18PDIP	44.00	37.40	30.26	18.72
PIC16C54—RC/SO	MCU, 0.5K, CMOS, EPROM	18SOIC	49.30	41.91	33.92	20.96
PIC16C54—XT/P	MCU, 0.5K, CMOS, EPROM	18PDIP	46.20	39.27	31.79	19.68
PIC16C54—XT/SO	MCU, 0.5K, CMOS, EPROM	18SOIC	51.80	44.03	35.62	22.00
PIC16C54—HS/P	MCU, 0.5K, CMOS, EPROM	18PDIP	50.60	43.01	34.77	21.52
PIC16C54—HS/SO	MCU, 0.5K, CMOS, EPROM	18SOIC	74.00	62.90	50.92	31.44
PIC16C55—RC/P	MCU, 0.5K, CMOS, EPROM	28PDIP	54.80	46.58	37.66	23.28
PIC16C55—RC/SO	MCU, 0.5K, CMOS, EPROM	28SOIC	63.00	53.55	43.27	26.80
PIC16C55—XT/P	MCU, 0.5K, CMOS, EPROM	28PDIP	57.50	48.88	39.53	24.48
PIC16C55—XT/SO	MCU, 0.5K, CMOS, EPROM	28SOIC	66.20	56.27	45.39	28.16
PIC16C55—HS/P	MCU, 0.5K, CMOS, EPROM	28PDIP	82.20	69.87	56.53	34.96
PIC16C55—HS/S	MCU, 0.5K, CMOS, EPROM	28SOIC	94.50	80.33	64.94	40.24
PIC16C56—RC/P	MCU, 1K, CMOS, EPROM	18PDIP	50.70	43.10	34.94	21.52
PIC16C56—RC/SO	MCU, 1K, CMOS, EPROM	18SOIC	56.80	48.28	39.10	24.08
PIC16C56—XT/P	MCU, 1K, CMOS, EPROM	18PDIP	53.20	45.22	36.72	22.56
PIC16C56—XT/SO	MCU, 1K, CMOS, EPROM	28SOIC	59.60	50.66	41.06	25.28
PIC16C56—HS/P	MCU, 1K, CMOS, EPROM	28PDIP	76.10	64.69	52.45	32.32
PIC16C56—HS/SO	MCU, 1K, CMOS, EPROM	28SOIC	85.20	72.42	58.65	36.16
PIC16C57—RC/P	MCU, 2K, CMOS, EPROM	28PDIP	62.80	53.38	43.27	26.64
PIC16C57—RC/SO	MCU, 2K, CMOS, EPROM	28SOIC	72.20	61.37	49.73	30.64
PIC16C57—XT/P	MCU, 2K, CMOS, EPROM	28PDIP	65.90	56.02	45.39	28.00
PIC16C57—XT/SO	MCU, 2K, CMOS, EPROM	28SOIC	75.80	64.43	52.19	32.16
PIC16C57—HS/P	MCU, 2K, CMOS, EPROM	28PDIP	94.20	80.07	64.94	40.00
PIC16C57—HC/SO	MCU, 2K, CMOS, EPROM	28SOIC	108.30	92.06	74.63	46.00
DSP320C/O/P	DSP 20.5MHz	40LD	112.90	95.92	79.31	63.60

地 址: 广东省深圳市振华路 414 幢(华发北路口)电子器材大厦

电 话: 354214 354345 350877 传 真: 350876

邮 编: 518031 电 挂: 8410

银 行: 工商银行华强分理处 帐 号: 22106500376

本公司上海经营部

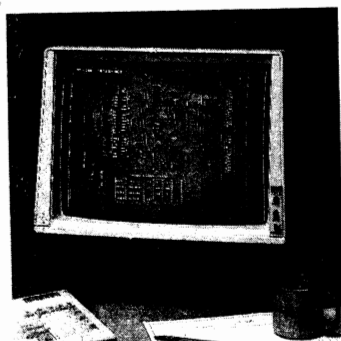
地 址: 上海天目西路 488 号长安大厦商场部

电 话: 3173798 电 挂: 2685 邮 编: 200070



ISSN 1006-1077
電子
與電腦

6



• ELECTRONICS AND COMPUTERS •

一九九二年

总期第87期

電子與電腦

目 录

• 综述 •

美国当今图像处理软件..... 吕问黎(2)

• PC 用户 •

dBASE III 和 FoxBASE+ 环境下实用多窗口弹出式菜单的实现方法..... 张志远(3)

CCDOS 2.1 特殊显示功能的妙用..... 李林枫(4)

在 DOS 中使用功能菜单的方法..... 金林樵(5)

PKZIP 压缩软件包的技术背景及使用..... 夏伟文(6)

如何使《UDOSV1.0》支持 AT101/102 键盘..... 杨 旭(9)

恢复硬盘启动功能的方法..... 王晓红(9)

• 学习机之友 •

无须驱动器的电子打字机程序..... 刘庆丰(11)

TOOL-KIT 使用详解..... 姜 宏(13)

利用 BOOT 程序对出租软件进行计次使用..... 任晓方(17)

中华学习机汉字双页显示..... 蒋建一(18)

高速排序程序的使用技巧..... 张世栋(19)

汉字编码打印程序..... 李 铁(21)

对《CEC-I 键控光标作图程序》的改进..... 李庆岱(22)

• 语言讲座 •

第六讲 简单程序和分支程序设计

..... 朱国江(23)

• 初级程序员级软件水平考试辅导 •

BASIC 基础..... 李志刚(29)

• 学用单片机 •

带 A/D 的单片机最小应用系统..... 张培仁 刘振安(31)

• 学装微电脑 •

7 段数码管读数装置..... 易齐干(34)

• 电脑巧开发 •

自带 EPROM 类单片机简易加密编程..... 庄 凯(38)

一种简单的键盘接口电路..... 罗许建(39)

• 电脑游戏机 •

第四章 F BASIC 语言的深入理解..... 于 春(40)

• 维修经验谈 •

维修打印头应注意的几个问题..... 蔡世清(44)

TH3070 点阵式打印机常见故障分析与检修(下)..... 刘立华(44)

• 读者联谊 •

普极型 PC 个人用户软件交流联谊活动问题解答(六)..... 王路敬(48)

封一:监视器 封二:资料 封三:MP-1 原理图 封四:广告

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部

(北京 173 信箱 邮政编码:100036)

印刷:北京三二〇九厂

国内总发行:北京报刊发行局

国内统一刊号:CN11-2199

邮发代号:2-888

国外代号:M924

出版日期:每月 23 日

主编:王惠民 副主编:王昌铭

责任编辑:施玉新

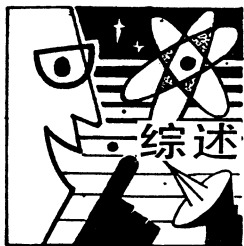
订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:0.95 元



美国当今图像处理软件

吕问黎

在美国丰富的软件种类之中,图像软件是一个相当引人注目的领域。这类软件主要有两大用途:图像显示和数据分析。前者主要是把书面的字句转化为视觉信号,经常在展销会、研讨会一类场所大显身手。后者则主要参予决策。在决策者和分析的信息之间建立起高速的信息交流,这一点正是决策者们从书面印刷材料中难以得到的;这类软件对于经常改变但主要输出形式又基本不变的各种数据(如股票市场信息)的处理尤为得心应手。此外,还有一部分图像软件专注于图像本身,不涉及其他信息,比如动画制作、宣传画绘制软件等。

现在美国市场上的图像软件的第一大特点是用户使用非常方便。用户自己无需有很高的绘画才能就可以制作出精美图像来。软件通过各种途径减少用户的麻烦,常设有大容量的图形库,提供现成的图像,以免用户自制之苦。比如 Computer Support Corporation 生产的 Arts Graphics Composer 软件存有 3700 个小型图像并有 15 种变化形式,而它的扩展版本的图像数上升到 5000 个。T/Maker Company 生产的 Clickart Images 是一个专门的图库软件,内有 1250 个如食品、电脑、旗一类的日常用品,140 多个动物、卡通形象和 240 个万圣节、复活节、情人节、舞会等场面,用户使用起来选择余地很大,绘图非常方便。

软件设计者对软件的操作程序也费尽心机使其尽量简化,如 Solution 国际公司出产的 Curator 软件是一个图像管理软件,用它寻找画面的方法就有五种,首先可以通过名称直接调入;如用户记不清全名而只输入名称的一部分也可调入相应的画面;还可以将所有画面名称列于屏幕上供选择;假如用户对画的名称一点儿也不记得了,软件会在屏幕上绘出多幅图像略图供用户选择;若这样还不行,用户也可逐一浏览图像。这个软件帮助用户整理、排列、打印、复制图像,它本身即是应方便使用这一要求而诞生的。

美国当今图像软件的另一大特点是作品质量上精益求精,技术上不惜花费大功夫。Autodesk 公司生产的 Autodesk Animator 是一个动画制作软件。它在动画显示时不仅运用了传统的翻页显示方法,而且采用了多种光学技术提高图形的三维立体性,还采用 Polymorphic(多形体)技术来直接使两个完全不同的图像相连接,而无需作者绘制中间的过渡画面。它还对字符的颜色等作了处理,还使各种字符也可以在画面上活动,产

生极佳的动画效果。软件还提供 20 多种工具来描绘作品细节,用户可以改变图像色调和色散程度,使图像变得不透明、柔和、有光泽或模糊,产生各种特殊显像效果。

MicroMind 公司出产的 MicroMind Director 是一个制作文字、图像、动画和小影片的软件,功能非常强大,制作成品的效果也令人赞叹不已。软件分两大部分,前一部分(Overview)控制着图像放映的先后次序和时间(最高达每秒 30 幅);后一部分(Studio)则是一个大工作室,有一套复杂的画面插入系统、动画操作系统和控制音响的程序。在细节上更是体现了软件的质量和制成品的质量,用户不仅可以对文字、音响效果等做微妙的控制,而且在画面与画面交替时,还可以采用电影中镜头交替的各种形式,如下一个镜头淡入,或者新画面逐渐推入将旧画面遮蔽,或者整个画面同时转化等。这些特殊的细节处理使人看了软件成品后觉得简直就是职业摄像师的作品一样。

图像软件的第三大特点是对硬件的开发达到了极高的程度。前述的 Autodesk Animator 软件可以容许同时有 4000 个物体参予动画过程,这对硬件的速度可谓利用到了极点。Deneba 软件公司的产品 Canvas 可以显示 1/1000 英寸的色点,可以有 1670 万种颜色,图像放大率可以在 3% 到 3200% 之间,图形旋转度可以小到百分之一度。

第四个特点是软件的通用性日增。前述的 MicroMind Director 软件制作的动画一旦完成后就可以单独拷贝,也就是说其他用户拷贝动画时无需将整个软件全部拷贝,拷贝后的动画作品可以在无原软件支持的条件下照常运行。通用性的另一特点是软件互相嵌套,也就是在使用一个软件时可以直接调用其它软件,而无需退出一个再调另一个。比如 Software Publishing 公司的 Harvard Graphics 软件就有专门模拟 DOS 的程序段,因而在这个程序运行时,DOS 系统支持的七个重要软件都可以照常调入运行。

当今美国图像软件中还有很多独创的技巧和新颖的构思,比如一般图像软件都内含一个拼写检查库,对用户的拼写错误予以自动纠正;有的软件可让几台微机同时使用,并分享重要昂贵的硬件设备(如高级打印机);还有的软件在必要的硬件支持下可以与摄像机、录像机相联通,从而使电脑图像产品可以录制在录像带上,录像带上的画面也可以进入电脑屏幕。



PC 用户

dBASE III 和 FoxBASE 环境下 实用多窗口弹出式菜单的实现方法

山东省计算机服务公司临沂分公司 张志远

目前开发和使用的一些新软件,正在流行着一种新的菜单界面方式:多窗口弹出式。这种方式的优点是:窗口位置、大小可以任意,光棒定位,清晰美观,选择不易出错,逐级弹出,逐级消失,并恢复被其复盖的原来屏幕画面,等等。这种菜单虽然好,但大都是用 C 语言等编写,大部分用户不熟悉。使用最多的是数据库管理系统,如 dBASE III 等,又不具备这种功能,编制应用程序时,仍使用数字或字母键选择,易出错、不美观。那么,能否利用编制一段子程序,使用时给出某些参数进行调用,以便实现上述功能,本人通过多次实践,在 dBASE III 和 FoxBASE 下,实现了这种多窗口弹出式菜单,现介绍给大家供参考。

该程序的实现过程是:

1. 将每行欲显示的菜单内容付给内存变量。
2. 在给定的位置开窗口,显示菜单内容。
3. 调用子程序,利用 Inkey() 函数,返回 ↑、↓、Enter 和 Esc 的键值,控制菜单光棒移动、选择和返回。
4. 弹出下一级菜单或执行该级被选项对应的功能模块。
5. 功能模块执行完后,再现当前级菜单(也可恢复所有前级菜单画面)。
6. 按 Esc 键,当前级菜单窗口消失,返回上一级窗口菜单。

其菜单实例见程序 CD1. PRG(一级)、CD23. PRG(二级)、CD32. PRG(三级)。

* CD1. PRG 一级窗口菜单

```
SET TALK OFF
SET COLOR TO /W
@0,0 CLEAR TO 24,79
PUBLIC Z
G1="系统管理"
G2="计划管理"
G3="固定资产"
G4="备件管理"
G5="人事管理"
G6="统计管理"
```

```
DO WHILE .T.
SET COLO TO W/B
@5,7 SAY G1
@6,7 SAY G2
@7,7 SAY G3
@8,7 SAY G4
```

```
@9,7 SAY G5
@10,7 SAY G6
DO IC WITH 5,7,1,6,"G"
IF Z=" "
RETU
ENDI
DO CD2&Z
ENDD
```

* IC. PRG 开窗口子程序

PARAM A,B,C,D,E

* A:起始行,B:起始列,C:行距,D:行数,E:内容
SET COLO TO R/G
@24,20 SAY "提示:上移↑下移↓选择 Enter 退出 Esc"

```
L=A
LE=LEN(G1)
SET COLO TO W/B
@A-1,B-1 TO A+D+(D-1)*(C-1),B+LE DOUBLE
DO WHILE .T.
SET COLO TO W/G
X=STR((L+C-A)/C,1)
@L,B SAY "&E&X" && 显示被选菜单
X1=X
```

W=INKEY(0) && 返回击入的键值

```
DO CASE
CASE W=5 && ↑上移
L=IIF(L=A,(D-1)*C+A,L-C)
CASE W=24 && ↓下移
L=IIF(L=(D-1)*C+A,A,L+C)
CASE W=13 && Enter 选择
Z=X
```

```
EXIT
CASE W=27 && Esc 退出
Z=" "
```

```
SET COLO TO /W
@A-1,B-1 CLEAR TO A+D+(D-1)*(C-1),B+LE
EXIT
ENDC
SET COLO TO W/B
```

```
@(&X1-1)*C+A,B SAY &E&X1  && 恢复
上次选单
```

```
ENDD
```

```
RETU
```

```
* CD23. PRG 二级窗口菜单
```

```
H1="台帐打印"
```

```
H2="台帐管理"
```

```
H3="报表输出"
```

```
DO WHILE .T.
```

```
SET COLO TO W/B
```

```
@8,19 SAY H1
```

```
@9,19 SAY H2
```

```
@10,19 SAY H3
```

```
DO IC WITH 8,19,1,3,"H"
```

```
IF Z=" "
```

```
RETU
```

```
ENDI
```

```
DO CD3&Z
```

```
ENDD
```

```
* CD32. PRG 三级窗口菜单
```

```
I1="台帐查询"
```

```
I2="台帐修改"
```

```
I3="台帐增加"
```

```
I4="台帐删除"
```

```
DO WHILE .T.
```

```
SET COLO TO W/B
```

```
@6,24 SAY I1
```

```
@7,24 SAY I2
```

```
@8,24 SAY I3
```

```
@9,24 SAY I4
```

```
DO IC WITH 6,24,1,4,"I"
```

```
IF Z=" "
```

```
RETU
```

```
ENDI
```

```
DO CD4&Z
```

```
ENDD
```

对于 FoxBASE+, 还可以使用下面更简便的方法实现, 这里仅给出一级窗口菜单程序 MU1. PRG, 其余下级弹出菜单依此类推。

```
* MU1. PRG 一级窗口菜单
```

```
SET TALK OFF
```

```
SET COLO TO /W
```

```
@0,0 CLEA TO 24,79
```

```
G1="系统管理"
```

```
G2="计划管理"
```

```
G3="固定资产"
```

```
G4="备件管理"
```

```
G5="系统退出"
```

```
DO WHILE .T.
```

```
SET COLO TO W/B
```

```
@6,7 TO 12,16  && 开窗口
```

```
SET COLO TO W/G
```

```
@7,8 PROM G1
```

```
@8,8 PROM G2
```

```
@9,8 PROM G3
```

```
@10,8 PROM G4
```

```
@11,8 PROM G5
```

```
N=1
```

```
MENU TO N
```

```
M=STR(N,1)
```

```
IF N=5
```

```
SET COLOR TO /W
```

```
@6,7 CLEA TO 12,16
```

```
EXIT
```

```
ENDIF
```

```
DO MU2&M
```

```
ENDD
```

CCDOS 2.13 特殊显示功能的妙用

安阳市建设银行电算科(455000) 李林枫

目前, CCDOS 2.13 汉字操作系统应用十分广泛, 利用其特殊打印功能打印文本文件和各类报表已极为普遍, 然而其特殊显示功能的使用者却为数不多。笔者发现, 在每屏仅能显示 10 行的中分辨率显示器上编制各类应用软件时, 由于客观条件限制, 人机界面很难处理, 计算机快速准确的提示功能得不到发挥, 因而编程工作倍受限制。如果把 CCDOS2.13 的特殊显示功能用于应用软件的设计中, 能够有效地缓解这一矛盾, 较好地

地进行人机界面的处理。

下面, 我们针对配置有中分辨率显示器的微机, 以 CCDOS 2.13F 为例, 说明在 FoxBASE 编程中特殊显示功能的具体应用。

数据库的内容常需查看, 利用特殊显示功能中的画矩形和横、竖线的命令, 可像手工制作表格一样把屏幕划分成几个窗口。由于构成窗口的矩形框和横竖线仅占用字符之间的空隙, 能充分利用有限的屏幕空间,

而且排列井然有序,紧凑美观。

利用 CCDOS 2.13F 的上滚当前页和下滚当前页命令,可实现显示内容的上下平移,效果甚佳。

CCDOS 2.13F 的特殊显示功能还可取消光标、设置颜色和清提示行,使显示屏幕清晰整洁,对于无图形功能的 FoxBASE 无疑是一种有效的功能补偿。

命令举例:

1. 在[0列,1行]至[20列,1行]之间做一横线。

@0,0 SAY CHR(14)+"H5,0,18,160]"

2. 在[20列,0行]至[20列,5行]之间做一竖线。

@0,0 SAY CHR(14)+"S5,160,0,90]"

3. 以[0列,0行]和[60列,8行]为对角线做一矩形框。

@0,0 SAY CHR(14)+"B5,0,0,480,144]"

4. 取消光标

@0,0 SAY CHR(14)+"I19,0]"

5. 建立光标

@0,0 SAY CHR(14)+"I19,1]"

6. 清提示行

@0,0 SAY CHR(14)+"I16,0]"

7. 使屏幕上[1列,1行]和[70列,8行]为对角线的矩形窗口内的字符以行为单位上下平移。

@0,0 SAY CHR(14)+"I6,1,,,1,1,8,70]"上滚一行

@0,0 SAY CHR(14)+"I7,1,,,1,1,8,70]"下滚一行

使用 CC-DOS 2.13 特殊显示功能应注意以下三

点:

1. 画点、画线和矩形框命令中的(X Y)座标均是用分辨率的点来描述的,而上滚/下滚当前页命令中的座标则是用行列值表示的。

2. CCDOS 2.13 的特殊显示命令因其版本不同而略有差异,使用时应查看相应的用户手册。

3. 特殊显示命令中的字母大小写均可。程序执行方式下,命令行末尾必须加],否则死机。

在 DOS 中使用功能菜单的方法

浙江杭州松木场河西 94 号(310007) 金林樵

菜单,是广大程序设计者广泛使用的选择分支方法。在 dBASE、FoxBASE 及 Turbo 系列软件中,菜单的设计给软件增添了勃勃生机。屏幕显示直观,选项移动灵活方便,给用户的使用提供了极大的方便。在 DOS 系统下,没有直接提供菜单功能。给用户在系统级上使用不同的系统设置了障碍。

我们知道,为了实现菜单功能,必须要有一个选择输入的过程,且能根据输入自动进行判断处理分支。但在 DOS 下,仅有一个 PAUSE 命令能实现任一键的输入,但它却不能将该输入值传送给 DOS 以加以判别。因此,只能设计一个过程。实现选项输入,并将该值传送给 DOS,随后用 DOS 提供的 IF、Errorlevel(n)Goto(标号)判断命令在一个批命令文件中依次实现上述菜单分支功能。

首先,用下附程序 MENU.COM 显示功能菜单,并等待用户的选择输入。最后将输入值放入 AL 中后带回 DOS。

然后根据用户输入用 IF、Errorlevel(n)Goto(标号)命令来实现分支转移。具体实现方法见下附程序 AUTOEXEC.BAT。

该批命令文件执行时,首先显示功能菜单,然后等待用户根据以上选项进行具体选择。输入后,其后的 IF Errorlevel 语句就根据用户的具体输入,作出相应的判断处理并实现分支。

以上程序已在 PC/XT 及其兼容机上调试通过。

MENU.COM 程序清单

C>DEBUG

-A

8EC9:0100 MOV DX,0110

8EC9:0103 MOV AH,09

8EC9:0105 INT 21

8EC9:0107 MOV AX,0C01

8EC9:010A INT 21

8EC9:010C MOV AH,4C

8EC9:010E INT 21

8EC9:0110 DB '0 CCDOS 4,0',A,D

8EC9:0120 DB '1 CCBIO 2,13H',A,D

8EC9:0133 DB '2 UCDO',A,D

8EC9:013F DB '3 BASICA',A,D

8EC9:014C DB '4 PCTOOLS',A,D

8EC9:015A DB '5 TURBO PASCAL,A,D

8EC9:016D DB '6 TURBO C 2.0',A,D

8EC9:017F DB '7 TURBO DEBUGGER 2.0',A,D

8EC9:0198 DB '8 MFOXBASE 2.10',A,D

8EC9:01AC DB '9 dBASE 1,10',A,D,A,D

8EC9:01BF DB 'SELECT(0-9)? \$'

-R CX

CX 0000

;D8

—N MENU.COM

—W

—Q

AUTOEXEC.BAT 程序清单

echo off

cls

menu

if errorlevel 57 goto J

if errorlevel 56 goto I

if errorlevel 55 goto H

if errorlevel 54 goto G

if errorlevel 53 goto F

if errorlevel 52 goto E

if errorlevel 51 goto D

if errorlevel 50 goto C

if errorlevel 49 goto B

if errorlevel 48 goto A

;A

cd\ccdos

..... ;CCDOS 4.0 各引导程序

goto K

;B

cd\213

..... ;2,13H 各引导程序

goto K

;C

cd\ucdos

..... ;UCDOS 各引导程序

goto K

;D

cd\BASIC

BASICA ;进入 BASICA

goto K

;E

cd\PCTOOLS

PCTOOLS ;进入 PCTools

goto K

;F

cd\TP

TURBO ;Turbo PASCAL

goto K

;G

cd\TC

TC ;Turbo C

goto K

;H

cd\TD

TD ;Turbo debugger

goto K

;I

cd\MFOXBASE

MFOXPLUS

goto K

;J

cd\DBASE

DBASE

;K

cd\

PKZIP 压缩软件包的技术背景及使用

崇文税务分局计算机室(100062) 夏伟文

“文件压缩存档(PKZIP)软件包 1.02 版”是美国“软件专业共享协会(ASP)”推荐,PKWARE 公司研制的产品。这个系列软件的前身是 PKARC,都属于使用“文档文件”来实现对原始文件的加密、压缩存储。所谓“文档文件”是指一种具有自己独特的内部结构标准的文件,当原始文件被储存时,按统一的标准压缩后存入“文档文件”中,用户可以方便地进行文件存取。国际上流行的许多软件都使用了该方法来分类存储各种文件,既可节省磁盘空间,又在一定程度上达到了加密的效果。对于用户来说,PKZIP 压缩软件又是自己加强文件管理的好助手。

PKZIP 与 PKARC 相比,有显著的优点。一是使用了新的 Imploding 方法,压缩的程度更高,虽然压缩时

慢一些,但是提取时速度相当快。二是原来 PKARC 压缩存储可执行的二进制文件时有毛病,PKZIP 则没有这方面的限制。三是可以由用户指定口令字进行加密压缩,并支持 LAN 等局域网工作和其它并行工作方式,其安全性、适应性有了很大的提高。四是可以使用 -rP 参数自动搜索文件的路径名,并将其压缩在文档中,提取时加 -d 参数自动生成所需的子目录。五是增加了“自提取型文档文件(.EXE)”,既可以直接运行实现文档提取,也可以用 PKZIP 或 PKUNZIP 处理(不包括 -V 查看功能),适合于不同水平用户的要求。六是能根据 CPU 的型号,自己决定是否使用 32 位指令寻址方式和调用 80386 CPU 的加速器。

PKZIP 软件所管理的“文档文件”是后缀名为 ZIP

的文件,这种文件的标准结构如下:

局部文件头部信息+文件内容,局部文件头部信息+文件内容,……,核心目录,核心目录结束信息

局部文件头部信息和文件内容构成一个文件项,一般是每一个文件项存放一个原始文件。

“局部文件头部信息”包括以下字段:

1. 局部文件头标志:4 字节(0x04034b50)

2. 能够提取文件的(软件)版本号:2 字节

3. 通用的标志位:2 字节

第 0 位:如果置 1 则为加密文件

如果压缩类型为 6,则第 1 位置 1 生成 8K 浮动数据字典,置 0 生成 4K 浮动数据字典;第 2 位置 1 说明用了 3 棵 Shanno—Fano 树来完成数据的输出;第 2 位置 0 说明用了 2 棵 Shanno—Fano 树来完成数据的输出

4. 压缩方式:2 字节。有以下几种:

0——文件不压缩直接存储

1——文件以 Shrink 方式压缩

i——用第 i-1 个压缩因子进行 Reduce 压缩(i=2,3,4,5)

6——以 Imploded 方式进行压缩

5. 最后的文件(更新)时间:2 字节(按标准的 MS-DOS 格式来编码)

6. 最后的文件(更新)日期:2 字节(同上)

7. 32 位 CRC 校验码:4 字节

8. 压缩以后文件长度:4 字节

9. 压缩以前文件长度:4 字节

10. 文件名的长度:2 字节

11. 保留字段的长度:2 字节

12. 文件名(可变量)

13. 保留字段(可变量)

第 12. 和 13. 两项与下面提到的“文档注释”的长度之和不得超过 64K

“核心目录”的结构如下:

[文件头],[文件头]……[核心目录结束信息]

[文件头]除了包含“局部文件头部信息”的所有字段(不包括第一个)外,还包括以下字段:

1. 核心目录头部标志:4 字节(0x02014b50)

2. 适用的版本号:2 字节

高位字节代表操作系统,低位字节代表生成文档的软件版本号

3. 文档注释的长度:2 字节

4. 开始的磁盘(扇区)号:2 字节

5. 内部文件属性:2 字节

如果最低位置 1 则说明原始文件是 ASCII 或文本文件,否则为二进制文件。

6. 外部文件属性:4 字节 原始文件的属性(其类型依赖于 OS 环境)

7. 局部文件头部信息的相对偏移量:4 字节

据此可以找到“局部文件头部信息”的存储位置

[核心目录结束信息]包含的数据项有:

1. 核心目录结束标志:4 字节(0x06054b50)

2. 本磁盘核心目录数目:2 字节

3. 核心目录开始的磁盘(扇区)号:2 字节

4. 本磁盘核心目录中登记项数目:2 字节

5. 核心目录中登记项数目:2 字节

6. 核心目录的长度:4 字节

7. 核心目录相对于文件开始磁盘扇区号的偏移量:2 字节

8. ZIP 文档注释的长度:2 字节

9. ZIP 文档注释的内容(可变量)

压缩的方式主要有以下几个:

1. Shrink:这是一种改进过的动态 Ziv—Lempel—Welch 压缩算法。初始的码长是 9 位,最大的码长是 13 位。它改进的功能有:(1)码长由压缩软件控制,不会使码长没有必要地自动增长,避免编码时的数据阻塞。(2)当码表填满时,并不清除全部结点,只释放 Ziv—Lempel 树的叶结点,以供使用。

2. Reduced:这种算法实际上是两种不同算法的混合使用。第一种算法是压缩重复的字节序列,第二种算法根据第一种算法压缩得到的数据流,再应用一种概率压缩方法进行处理。

3. Imploding:该算法也是由两种不同算法组合而成的。第一种算法是用浮动数据字典来压缩重复的字节序列。第二种算法是用 2~3 棵 Shannon—Fano 树来实现对浮动数据字典编码输出的压缩。如果使用 3 棵 Shannon—Fano 树,第一棵树用来描述 ASCII 码字符集。后两棵树分别用来描述浮动数据字典中的数据项对——长度和距离信息。如果只使用 2 棵 Shannon—Fano 树,则描述 ASCII 字符的树被忽略,只存在后两棵树。

4. 加密压缩。每一个加密的文档都定义一个 12 字节长的加密文件头存放在文件内容区域的开始部分。每一个加密文件头初始时被设置成随机数,接着用 3 个 32 位密钥来加密,密钥由用户指定的口令字生成。等到原始文件的每一个字节都被加密后,密钥将用伪随机代码生成技术与文件的 32 位 CRC 校验码结合在一起,放在文件的某个地方。

解密的步骤如下:①32 位密钥被用户指定的口令字初始化。②读入 12 个字节的加密文件头,解密后用以更新 32 位密钥。③用这 32 位密钥对整个文件进行解密。

该软件可以在 IBM PC 及其兼容机上运行,要求 DOS 版本在 2.0 以上或者是 OS/2。PKZIP 需要 85K 的内存,PKUNZIP 需要 70K 的内存才能运行。该软件一次最多可以将 3900 个文件压缩入一个文档文件。为了使文档文件在不同的 OS 上统一,路径名中用“/”代替 MSDOS 中的“\”。

“文件压缩存档(PKZIP)软件包 1.02 版”包括下列应用程序及文件:

WHATSNEW. 102 PKWARE 1.02 版新增加的功能。

OMBUDSMN. ASP“软件专业共享协会(ASP)”有关于本软件包的通知。

DEDICATE. DOC 关于开始使用. ZIP 作为“压缩(Compressed)型数据文件”的括展名(后缀)的通知。

LICENSE. DOC 关于机关,单位(公司)取得本软件包使用许可证的手续。

MANUAL. DOC 本软件包的使用说明书。

ORDER. DOC 本软件包的使用登记与定购单。

README. DOC 本软件包简介。

REZIP. DOC 旧版(PKZ091,PKZ092),ZIP 文件的兼容与升级。

APPNOTE. TXT 本软件包的技术背景资料。

BIOSFIX. COM 停用 80386 CPU 加速器(32-bit accumulator;EAX)功能。

PKSFX. PRG 自动配合 ZIP2EXE. EXE 文件使用。

MAKESFX. COM 当 PK102. EXE 在当前目录时,执行 MAKESFX. COM 则生成 BIOSFIX. PRG 文件。

PKUNZIP. EXE 对已压缩的文档文件(*. ZIP)的提取程序。

PKZIP. EXE 对文件进行压缩存档的程序。

PKZIPFIX. EXE 用于修复已局部损坏的文档(*. ZIP)文件。

READ-1ST. EXE 本软件包的来源说明。

REZIP. EXE 用于将原用 PKZ090. EXE 或 PKZ092 压缩存档的文档文件改为 PKZ101. EXE 的格式存档。

ZIP2EXE. EXE 用于生成自提取(Self-extracting)文档文件。

PKZ102. EXE 本软件包内的全部文件均已存入这个自提取文档。

注:当我国软件保护法生效后,请您自觉按规定向版权法人交费。

下面简要介绍一下 PKZIP 软件包的使用。

(1)PKZIP. EXE 的使用格式:

PKZIP[-b[路径]] [可选参数] [-s(password)] [ZIP 文档文件] [@文件表] [文件...]

可选参数包括:

-d=删除文件 -l=显示版权信息

-f=更新文件(只更新以前已经存储的文件)

-u=更新文件(将已经存储的文件用最新版本更新,将没有存储的文件加入)

-b=在指定的驱动器上建立临时的 ZIP 文件

-c=加入/编辑文件注释

-C=只对新文件加入注释

-i=只将已经修改过的文件加入文档

-m[u,f]=将原始文件移动到文档内,原文件将被删除

-r=自动寻找当前目录下的子目录中的文件

-es=使用快速压缩 -a=加入文件

-ex=使用最大限度的压缩(默认)

-k=保持原有的 ZIP 日期

-z=增加/编辑文档文件的总注释

-o=将 ZIP 文件日期设置成最近的原始文件的日期

-P=将路径存入 ZIP 文档内(应与-r 参数合用,注意不要把大小写弄错)

-q=允许 ANSI 的注释

-x(通配符)=不包括指定的文件

-s(口令字)=用口令字给文件加密

-w/W(H,S)=包括/(或不包括)隐含/系统文件

-j/J(H,S,R)=对文件压缩存储时屏蔽/(或不屏蔽)隐含/系统/只读文件的属性

ZIP 文件=ZIP 文档文件名(包括路径名),默认的扩展名是. ZIP

文件 =将要压缩的原始文件名(包括路径名)可使用通配符如*,?。默认的是所有的文件

@文件表=包含了将要观察或压缩存储的文件名字的登记表

(2)PKUNZIP. EXE 的使用格式:

PKUNZIP [可选参数] ZIP 文档文件 [盘符:输出路径] [要提取的文件.....]

可选参数为:

-c[m]=把原始文件提取出来并送到屏幕[加 m 参数可逐屏或逐行显示]

-t=测试 ZIP 文件的完整性

-l=显示软件的版权信息

-d=建立输出路径中的目录用以存储提取出的原始文件

-o=覆盖已经存在的原始文件

-q=允许 ANSI 的注释

-s(口令字)=用口令字将文件解密提取

-n=只提取出较盘上已有文件更新的文件

-p[a,b,c][1,2,3]=将提取出来的文件送到打印机

[按 ASCII 码/按二进制码/送往 COM 通信接口]

[通信口编号]

-j/J(H,S,R)=在提取原始文件时屏蔽掉/(或不屏蔽)隐含/系统/只读等文件属性

-v[b,c,d,e,n,p,s,r]=观察 ZIP 文件[简要地列表/按 CRC 码/日期/括展名/名字/压缩百分比/长度排序/按逆序排列]

-e,x=分离出原始文件(默认)

-v 参数是 PKZIP 和 PKUNZIP 都可以选用的,其含义为:

-v[b,c,d,e,n,p,s,r,t]=观察 ZIP 文件[简要地列表/显示注释/按日期/括展名/名字/压缩百分比/长度排序/按逆序排列/详细列出技术细节]

(3)PKZIPFIX. EXE 使用格式如下:

PKZIPFIX ZIP 文件

PKZIPFIX 将会重建已经局部损坏的 ZIP 文件,并将从该文件中可能恢复的原始文件存到 PKFIXED. ZIP 文档中。

(4)自提取文档生成程序(ZIP2EXE. EXE)的使用

格式如下:

ZIP2EXE ZIP 文档文件

它可以把已生成的. ZIP 文档用本程序转变成同名的自提取型(. EXE)文件,使用这个文件时必需把PKSFX. PRG 文件与它拷在一个目录内或建立路径后使用。自提取型文件仍能用 PKZIP. EXE 或 PKUNZIP 进行处理。执行自提取型文件后将根据指定的参数提取出文件。执行的格式如下:

自提取文档名[可选参数][盘符:输出路径][文件.....]

可选参数包括:

-c[m]/-e/-d/-l/-o/-t/-n/-p[a,b,c]
[1,2,3]其含义与 PKUNZIP 的参数相同。

对于 1.1 版的 PKZIP 软件包而言,还可以加上一-s (psw)参数(其含义同上)。

自提取型文档一旦生成以后,可以用-d 参数删空存放其内部的无用文件,将它作为一个基础文档保留起来。需要自提取型文档时,只要拷出一个备份,就可以用 PKZIP 直接将原始文件压缩到该自提取型文档中即可,无需再用 ZIP2EXE 进行转换。

如何使《UCDOS V1.0》

支持 AT 101/102 键盘

辽宁省计量测试技术研究所(110006) 杨旭

中国科学院希望高级电脑技术公司开发的《超级组合式汉字系统》UCDOS V1.0,具有许多独到的优点,深受广大用户的欢迎。遗憾的是,该软件不支持 AT 101/102 键盘中部的两组小键盘(扫描码为 E0 xx)。不但不能发挥 AT 101/102 键盘的优势,反而造成许多不便,影响输入速度。因此,本人通过对其键盘程序的分析,加入一小段程序,恢复了这两组小键盘的正常功能。使 UCDOS V1.0 在 AT 及其兼容机上应用起来更加方便。用 DEBUG 修改其键盘程序的方法如下:(修改后的程序在 286 及 386 机上运行通过)

```
C:\>CD\UCDOS
C:\UCDOS>REN KB. EXE KB
C:\UCDOS>DEBUG KB
-A CS:9039
CS:9039 JMP A1C0
CS:903C
-A CS:A1C0
CS:A1C0 JNZ A1CD
CS:A1C2 OR AL,AL
CS:A1C4 JZ A1DD
CS:A1C6 CMP AL,E0
CS:A1C8 JZ A1D8
CS:A1CA JMP 900E
CS:A1CD OR AL,AL
CS:A1CF JZ A1DD
CS:A1D1 CMP AL,E0
CS:A1D3 JZ A1D8
CS:A1D5 JMP 926E
CS:A1D8 MOV AL,00
CS:A1DA MOV [0038],AX
CS:A1DD JMP 9048
CS:A1E0
```

-R CX

:A1B3

A1E0

-W

-Q

C:\UCDOS>REN KB KB. EXE

恢复硬盘启动功能的方法

福建泉州华侨大学机械系(362011)王晓红

操作系统型病毒感染或其他原因,破坏硬盘主引导记录的数据,将使硬盘不能启动。由于硬盘主引导扇区是非 DOS 部分,DOS 不能直接访问,因而常见的解决办法是低级格式化硬盘,重新分区,格式化。这样做,硬盘原来的数据全部丢失,造成巨大的损失。针对这一问题,本文介绍一种利用 ROM BIOS 的 INT 13H 来“救活”硬盘的方法,具体步骤如下:

1)准备一张带有 DEBUG. COM 的用户盘,并用行编辑命令建立 RDBOOT 和 WTBOOT 两个文件;

2)在一台带有同型号健康无毒的硬盘的机上键入:

A>TYPE RDBOOT | DEBUG

不同类型的硬盘的结构差别较大,分区表各不同,所以最好在机器正常的时候做上述两步,并存好磁盘待用。若有几种型号不同的计算机,只需改变上述两文件中的 N BOOTXT 句,如在 286 机上,则改为 N BOOT286。

3)当机器出现主引导扇区坏而不能启动时,用软盘启动后,插入上述软盘,然后键入:

A>TYPE WTBOOT | DEBUG

重新启动后机器工作正常。

RDBOOT 文件

A>type rdboot

a 100

mov ax,0201

WTBOOT 文件

A>type wtboot

n bootxt

1 200

```

mov bx,0200      a 100
mov cx,0001      mov ax,0301
mov dx,0080      mov bx,0200
int 13           mov cx,0001
int 20           mov dx,0080
                int 13
                int 20

g=100
r cx
0200            g=100
n bootxt       q
w 200
q

```

(上接 48 页)

有错,这往往是硬件故障造成。例如一台长城 0520CH 的彩色显示器,显示信息杂乱,经检查结果,显示卡故障。现在又发现,如机器感染上象圆点等一类病毒,一旦病毒发作,屏幕显示就出现“圆点”跳动干扰屏幕显示。这种问题的出现只要用解病毒软件清理一下即可恢复。

19. 打印机分哪两大类? 有哪几种打印方式?

打印机是微型计算机系统的主要输出设备之一。在微型机系统中,复印机是作为一个独立的部件与主机分离存在的。主机中都含有一个或多个打印机接口,这种接口多采用并行方式传送数据,即用 8 根数据线每次将一个字节的数据同时送出。计算机以中断或查询方式控制着打印机的动作,打印机服务程序是作为操作系统的一个组成部分常驻于主机内存中。

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

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

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

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

在现代的微型机系统中,上述两种打印方式往往是共存的,到底使用哪一种要视具体情况而定。有时,

用户可用键盘输入命令或通过程序中给定的指令来选择其一;有时由系统规定而不能改变。

20. 打印机控制器与打印驱动程序有什么关系? 使用汉字打印驱动程序应注意什么问题?

打印机控制器或称适配器可以是一块独立的选件板,也可以是“显示控制器/打印控制器”选件板的一个组成部分。打印控制器由命令译码器数据锁存器、总线缓冲器、控制锁存器、控制驱动器以及接收发送器等逻辑部件组成。命令译码器负责识别从 CPU 送来的具有系统分配给打印控制器所规定的端口地址的输入输出指令,并且译出“数据传送方向”、“读数据”、“写数据”、“写控制”、“读状态”、“读控制信号”等六个命令。数据锁存器用来暂存 CPU 送出的打印数据字节,然后一方面通过 D 形插头座送到打印机去打印,另一方面又回送到总线缓冲器,以便 CPU 必要时可以读入比较,进行故障测试与诊断。控制信号锁存器及集电极开路驱动器用来暂存并驱动四个控制信号:初始化、选通、自动输纸以及选择输入信号。这些控制信号在输出的同时也回送到另一个总线缓冲器,同样也是为诊断时用。打印机的五个状态信息:联机、认可、忙、缺纸、出错均为稳态电平信号,它们经由总线缓冲器被“读状态”命令选通后,即送往数据总线,由 CPU 的输入指令取回送入某寄存器中,供软件作分析打印机状态时使用。

有了打印控制器,还要有打印驱动程序的支持。通常打印输出控制选用查询方式或者中断方式来实现。在 IBM PC 的 ROM 常驻 BIOS 中,打印输出控制采用的是查询方式。需要指明的是,系统里提供的打印驱动程序仅支持英文字符的打印,即在 PC DOS 各种版本控制下的系统,只能输出字符信息,对于汉字信息处理时,就需要另配汉字打印驱动程序,这些程序要根据不同型号的打印机分别进行设计。

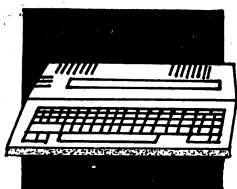
使用汉字打印驱动程序应注意下列问题:

(1)打印驱动程序应与打印机的型号配套。换言之,什么型号的打印机,要选择与该型号打印机配套的打印驱动程序。要注意,同一型号的打印机可有功能强弱不同的打印驱动程序。所以当你购买一型号打印机时,千万不要忘记让公司或厂家配给相应的打印汉字驱动程序。

(2)打印驱动程序应与汉字库文件配套。因为有的功能较简单的汉字打印驱动程序,仅要求 16×16 点阵 CCLIB 汉字库文件,功能稍强一些的打印驱动程序,不但要求 16×16 点阵 CCLIB 汉字库文件在内存,同时往往由于变换汉字字型的需要,要求 24×24 点阵 CLIB24 汉字库文件在硬盘上。

(3)打印驱动程序要与使用的汉字系统版本配套。汉字打印驱动程序运行的环境在汉字系统下,不同汉字系统下,同一打印机,有功能不同的打印驱动程序。

工作的实践经验告诉我们如果打印机打不出汉字来,或者打印汉字时死机,或打印的不是汉字就很有可能是上述三个问题没有解决好。



学习机之友

无须驱动器的电子打字机程序

重庆出版社科技编辑室(630050) 刘庆丰

各类型号的中华系列学习机均有较强的汉字功能。如小蜜蜂(XMF-1)机不但具有五种汉字输入法(其中的部首方式连选择键在内最多四键即可输入一汉字),而且固化了七千多单词的联想式词组,汉字输入速度已不逊于各类昂贵的电子打字机。但要使各种类型的学习机具备电子打字和汉字处理功能,必须要依赖于驱动器,对广大未配驱动器的学习机用户,要用电脑打字则必须将其内容逐一填入 PRINT 语句之中,且每行字数还不能超过八十汉字,相当繁琐。

若能使输入的汉字或西文字符自动变为 PRINT 语句中的内容,则可使学习机具备电子打字机的大部分功能,有相当的实用价值,本程序即是根据这一构想设计的。

使用方法

键入本程序并存带。每次使用时只需将其输入内存,运行后首先让你选择标题及正文的字号及字距等,接着即按日常书写习惯在句首空出两个汉字空间(也可用退格键删去)等候输入。对输入的所有汉字及西文,它均将其变为 PRINT 语句的内容,对双引号则以 CHR\$(34)的形式生成程序。当每行超过五十字符时自动加入末尾的双引号和分号,并另起一行继续书写 PRINT 语句,所以各自然段的字数可无任何限制,直至按下回车键时才自然分段。写作过程中若发现输入有误,可按退格键,程序将自动删去最后的一个汉字或西文或双引号,继续按可继续删除,若本语句行字符已全部删完,则删去上一行的结尾字符,同时行号自动减十。若按 CTRL-X 键,则删去上次回车后的本自然段全部内容。若按右行键,就又把光标划过的字符逐一添入到程序后,若误按回车使内容换行,可按 CTRL-B 键删去回车。

为提高人机界面的友好度,系统还设置了 CTRL-A 和 CTRL-C 两个功能键。后者用于查看已写字数和已生成的行号数,前者是一个用途较广的键,现将功能简述如下:

1. LIST 任何时候均可按下,以观察已完成的内容。列表结束后将自动返回等待输入状态。

2. RUN 按下后显示键入的全部文字内容,显示结束程序运行。若要继续,可键入 GOTO 50。在对程序进行手工改动后欲继续写作,则应键入 S%=行号: GOTO 50。

3. 选字号 当键入选好的字号后,程序即已将该号字体下每行的打印字数一并设好(按普通稿纸宽度计算),以 POKE 语句的形式生成一个单独的语句行。

它还能按使用者要求自行设置前置空格以供标题制作之用。同时,程序还提供了两种用于正文的标准字体,均按每行 20 字,每页 20 行设置:标 1 为标准二号字,标 2 为小号长方字,但每字间自动加入半汉字空间,用于文章引述,诗歌等需另置字体的场合。

4. 序号 这是为文内作段落标题和句首标号设置的,可免去写作时查索标号数字的麻烦。

文章全部写完后,若需作大的编辑加工,可用任何编辑 BASIC 程序的手段进行处理。若要存带,可先删去 0—500 行,也可连同这部分写作系统一并存储。后者在存带前最好加上一句 0 GOTO 1000 以免使用时发生混乱。

程序分析

本写作系统主要分为自动生成程序的机器语言核心程序和汉字输入与提供人机对话接口的 BASIC 程序两大部分。

由于电脑作打字时输入的并不全是欲打印的内容,还有很多是控制命令等,所以必须对输入的内容逐一进行识别,选择。这用 GET 语句是很理想的。但各学习机的 GET 都不能用于汉字状态,如 CEC 用 GET 采入汉字时得到的却是当时键入的拼音码或区位码,小蜜蜂机虽可键完一个汉字才采集一次,却只能获得以该汉字三内码最后一码开始的一串无用字符。

通过对内存的查询,找到了小蜜蜂在汉字 GET 时的一个特殊储存单元: \$B714。该单元在输入汉字时储存的是汉字内码的第二码,当输入 ASCII 码和控制符时内容为 35 或 33。据此就可以编出汉字 GET 的程序了。因为第一字节的引导码是可以自己加进去的,繁体汉字为 127,简体为 126。

作为人机界面的 BASIC 程序设计了较多的功能与提示,这部分读者可根据自己的情况进行增删。

位于 \$7800 至 \$79C9 间的机器语言程序是系统的核心,它又是由若干子程序组成,有较强的容错能力,对各种误操作均能响铃示警,并考虑了与各种机型的兼容,凡是汉字引导码为 127 或 126,且码长为三字节的各种苹果系列软硬汉字系统均可运用。为方便读者灵活运用,现将一些主要功能段及其入口地址简介如下:

\$780E: 查变量。调用前将单字节的字符串变量或整数型变量的 ASCII 码存入 \$FE 单元,调用后变量地址在 \$4C, \$4D 二单元中。

\$78C0: 根据 S%-值将新编行号写至程序尾部。

\$78EC: 将 A\$ 内容变为程序内容。灵活调用孩子

程序,可生成任何程序。但要注意的是程序中的保留字只能用 CHR \$(保留字内码)的形式生成。

\$ 7922:删去程序尾部的一个字符。

\$ 7987:删去上次回车后输入的所有内容

CEC 机使用本系统的方法

本程序是在 XMF 机上设置通过的。CEC 机若要使用本系统,需对程序作如下改动:

1. 程序首的 PR # 5 应改为 PR # 3。

2. 改动汉字输入语句。CEC 的汉字输入入口不同于 XMF 机,为此需编一段机器语言程序来作为汉字 GET 语句。具体作法是:

(1)加入以下两行程序

```
18 FOR X=768 TO 774:READ A:POKE X,A:
NEXT,DATA 32,129,158,141,16,3,96
```

```
55 CALL 768:A$=CHR$(PEEK(784)):IF
PEEK(784)=127 THEN CALL 768:A$=A$+CHR
$(PEEK(784)):CALL 768:A$=A$+CHR$(PEEK
(784))
```

(2)删去 95,100 两行全部内容。

3. 本程序中 240—250 三行为设置打印机的控制字符(字号,行宽,行距等),CEC 机用户可对此作自己的任意安排。方法是参照此程序将本机的控制码写入 A\$ 内(程序中的 CHR \$(185)即 POKE 的保留字代码)即可。

4. 由于 CEC 机不能任意使用汉字屏幕第十一行(但汉字状态下可有反白显示),故可将提示改为反相显示。即将本程序中所有用到 VTAB 11 的地方都改为 INVERSE,并删去 140 行的子程序。

本程序在有驱动器的情况下亦可照常运行

```
2 PR# 5:PRINT
3 VTAB 3
5 PRINT"          汉字文稿写作系统":VTAB 8
6 PRINT SPC(9)"编制人:刘庆丰":PRINT SPC(13)"1991.
1."
8 FOR X=30720 TO 31177:READ A:POKE X,A:NEXT
9 DATA 56,165,175,229,255,133,74,165,176,233,0,
133,75,96,56,165,107,229,105,133,23,160,0,196,
23,144,3,76,221,251,177,105,197,254,24,240,7,
152,105,7,168,76,23,120,152,101,105,133,76,8
10 DATA 200,177,105,201,128,240,4,136,76,37,120,
40,165,106,105,0,133,77,96,56,165,74,229,255,
133,74,176,2,198,75,96,24,165,74,101,255,133,
74,144,2,230,75,96,160,0,152,145,74,200,145
11 DATA 74,200,145,74,160,0,24,165,74,105,1,145,
238,200,165,75,105,0,145,238,24,165,74,105,3,
133,175,165,75,105,0,133,176,96,160,0,152,133,
252,169,1,133,255,230,252,32,69,120,177,74
12 DATA 201,186,208,245,56,165,74,233,4,133,238,
165,75,233,0,133,239,96,169,211,133,254,32,14,
120,160,3,56,177,76,233,10,145,76,136,177,76,
233,0,145,76,96,169,2,133,255,32,0,120,165
13 DATA 74,133,238,165,75,133,239,169,211,133,
254,32,14,120,160,2,177,76,200,145,74,177,76,
```

```
136,145,74,169,4,133,255,32,81,120,76,93,120,
169,65,133,254,32,14,120,160,2,177,76,133,25,
200
```

```
14 DATA 177,76,133,23,200,177,76,133,24,169,3,
133,255,32,0,120,160,0,196,25,176,8,177,23,145,
74,200,76,12,121,165,25,133,255,32,81,120,76,
93,120,169,6,133,255,32,0,120,160,0,177
15 DATA 74,133,227,201,127,240,19,201,126,240,15,
169,2,133,255,32,81,120,177,74,133,227,201,34,
240,3,76,93,120,169,6,133,255,32,69,120,177,74,
201,34,208,3,76,93,120,169,1,133,255,32
16 DATA 69,120,177,74,201,59,240,8,32,221,251,
169,160,108,54,0,32,134,120,24,165,238,101,252,
133,74,165,239,101,253,133,75,32,168,120,76,41,
121,32,168,120,32,134,120,169,6,133,255,32,69
17 DATA 120,160,0,177,74,201,59,240,235,169,8,
133,255,32,81,120,76,93,120,32,134,120,169,6,
133,255,32,69,120,160,0,177,74,133,227,201,34,
240,3,76,221,251,32,134,120,165,252,133,255,32,
81,120,32,93,120,76,168,120
20 LOMEM:28672:HIMEM:30720:S%=1000:VTAB 11
22 PRINT"          行号自 1000 行开始?":GET A$
23 IF A$="N" THEN INPUT "多少?":S%
24 CALL 30912:A$=CHR$(186)+CHR$(34)
"          ":CALL 30956
26 GOSUB 180:S%=S%+10:GOSUB 140:IF A=1 THEN
29
28 INPUT "          1. 设空间,字距 0. 不设          ":A$:IF A$
="1" THEN 260
29 V=10:GOSUB 140
30 CALL 30912:PRINT"          "
35 A$=CHR$(186)+CHR$(34):CALL 30956
40 A$="          ":CALL 30956
45 IF FRE(0)<1000 THEN PRINT "FRE(0)/3="FRE
(0)/3
50 FOR I=1 TO 50
55 GET A$;A$=LEFT$(A$,1):PRINT A$;
60 IF A$=CHR$(34) THEN A$=CHR$(34)+CHR
$(231)+"(34)" +CHR$(34):GOTO 105
65 IF A$=CHR$(13) THEN A$=CHR$(34):CALL
30956:S%=S%+10:ZZ=ZZ+Z:Z=0:GOTO 30
70 IF A$=CHR$(8) THEN GOSUB 130:PRINT CHR$(8
*(PEEK(227)=126 OR PEEK(227)=127)):Z=Z-1:
GOTO 110
75 IF A$=CHR$(24) THEN CALL 31111:PRINT "\":
PRINT"          ":Z=0:GOTO 40
80 IF A$=CHR$(1) THEN 150
85 IF A$=CHR$(2) THEN CALL 31139:V=8*(PEEK
(227)=34):PRINT CHR$(V) CHR$(V) CHR$(V)
CHR$(V):GOTO 110
90 IF A$=CHR$(3) THEN 350
95 IF PEEK(46868)=35 OR PEEK(46868)=33 THEN 105
100 A$=CHR$(126)+CHR$(PEEK(46868))+A$
105 A$=A$+K$:CALL 30956:Z=Z+1
110 NEXT
```

```

115 A$=CHR$(34)+“,”;CALL 30956
120 S%=S%+10;CALL 30912
125 A$=CHR$(186)+CHR$(34);CALL 30956;GOTO 45
130 CALL 31010;IF PEEK(227)=160 THEN 130
135 RETURN
140 VTAB 11;HTAB 1;PRINT“
      ”;VTAB V;RETURN
150 VTAB 11;HTAB 1
152 INPUT “1. LIST 2. RUN 3. 选字号 4. 序号”;A$;V=
    10;GOSUB 140;ON VAL (A$) GOTO 155,500,26,
    300
155 LIST 1000,;GOTO 110
180 V=11;GOSUB 140
181 REM
182 INPUT “几号字? 1—8;字号 9;标 1 0;标 2”;A$;IF A
    $ (“0”OR A$) “9” THEN 180
185 IF A$ = “1”OR A$ = “7” THEN ZS=44
195 IF A$ = “2” OR A$ = “5” OR A$ = “8” THEN ZS=
    22
205 IF A$ = “3” THEN ZS=30
215 IF A$ = “4” THEN ZS=15
225 IF A$ = “6” THEN ZS=11
230 IF A$ = “9” THEN 245
235 IF A$ = “0” THEN 250
240 CALL 30854;CALL 30813;A$=CHR$(185)+
    “40963,”+A$+“,”+CHR$(185)+“40961,”+
    STR$(ZS);CALL 30956;K$=“”;A=0;RETURN
245 CALL 30854;CALL 30813;A$=CHR$(185)+
    “40963,2,”+CHR$(185)+“40961,20,”+CHR
    $(185)+“40962,20”;CALL 30956;K$=“”;A=1;
    ZS=20;RETURN
250 CALL 30854;CALL 30813;A$=CHR$(185)+
    “40963,1,”+CHR$(185)+“40961,30,”+CHR

```

```

$ (185)+“40962,20”;CALL 30956;K$=CHR
    $(160);A=1;ZS=40
255 RETURN
260 GOSUB 140;INPUT “本行共有几字?”;A
265 GOSUB 140;INPUT “字间留多少空格?”;B;QK=ZS-A
    -(B*(A-1)/2);KG=2*QK
268 INPUT “1. 前 1/3 2. 后 1/3 0. 正中”;A$;IF A$ =
    “1” THEN QK=KG/4
270 IF A$ = “2” THEN QK=KG*3/4
273 REM
274 IF QK<0 THEN PRINT “超宽! 重选”;GOTO 260
275 K$=“”;IF B<>0 THEN FOR I=1 TO B;K$=K$
    +CHR$(160);NEXT
280 V=10;GOSUB 140;CALL 30912;A$=CHR$(186)+
    CHR$(34);CALL 30956;A$=“”;FOR I=1 TO QK;
    A$=A$+“”;NEXT;PRINT A$;;CALL 30956;GO-
    TO 50
300 V=11
302 VTAB V;PRINT “1. 设序号 0. 写序号”;;GET A$;IF A
    $ (“1” THEN 315
303 HTAB 1
305 XH=1;PRINT “若序号不从头开始则按(1)”;;GET A
    $
308 IF A$ = “1” THEN INPUT “从几开始?”;XH
310 GOSUB 140;INPUT “1. 正中 2. 句首”;B$;GOTO 320
315 XH=XH+1
320 V=10;GOSUB 140;IF B$ = “2” THEN 330
325 A$=“”;FOR I=1 TO ZS-7;A$=A$+“ ”;NEXT;
    A$=A$+STR$(XH)+“.”;CALL 30956;PRINT A$
    $;S%=S%+10;GOTO 30
330 A$=STR$(XH)+“.”;CALL 30956;PRINT A$;;
    GOTO 50
350 VTAB 11;PRINT “行号=”S% “字数=”ZZ+Z;GET A
    $;V=10;GOSUB 140;GOTO 50

```

TOOL—KIT 使用详解

北大附中 姜 宏

(一)系统简介:

TOOL—KIT 软件是苹果公司推出的工具软件,之所以叫工具箱,是由于它里面包含着几个很有用的工具程序,这些程序是用户更好利用苹果机的基础。本文就是为介绍它们而写的。

这张磁盘里,有以下几类程序:

(1)工具类:这类文件可帮助用户设计 CAI 软件,比较典型的是 HRCG 程序。

(2)游戏类:这类程序大多是工具程序的示范,剖析这些程序,你会有意想不到的收获。

(3)字符集:可帮助用户编写游戏程序。

(二)使用详解:

1. 再定位载入程序——RBOOT, RLOAD

TOOL-KIT 这张软盘里,有一类很奇怪的文件,即 R 型文件。R 是英文 Relocatable 的简写,中文意思是再定位,在 DOS3.3 中,没有存取这类文件的命令。这使 R 类文件的功能无法全部发挥,苹果公司为此写了这两个程序,在介绍它们之前,我们先介绍一些基本知识。

根据苹果公司的资料,DOS 共可处理 8 类文件,它

们在目录道的代码及中文含意为:

- \$ 00—TEXT 数据文件(T 型)
- \$ 01—INTEGER 整数 BASIC(I 型)
- \$ 02—FP—BASIC(APPLE SOFT)(A 型)
- \$ 04—BINARY 二进制代码文件(B 型)
- \$ 08—SYSTEM 系统文件(S 型)
- \$ 10—RELOCATABLE 可再定位文件(R 型)
- \$ 20—新 APPLESOFT 文件(?)
- \$ 40—由 LISA 2.5 生成的 6502 汇编文件源程序(L 型)

但由于种种限制,DOS3.3 并没有加上对 S、R、? 和 L 四类文件的处理程序,LISA 2.5 在中国风行后,L 类文件被人注意了,一些汉字系统推出后,字库文件符号采用 S、?,那两类文件也被人注意了,而 R 类文件一直到 TOOL—KIT 推出之后才有人注意,那么,它的原理是什么呢?

我们先来看看下面两个简单的 6502 汇编程序:

```
程序 1:0800-A9 00      LDA # $00
```

```
      0802-4C 00 08      JMP $0800
```

```
程序 2:0900-A9 00      LDA # $00
```

```
      0902-4C 00 09      JMP $0900
```

这两个程序执行效果相同,只有一个代码不同,即: \$ 804 单元中的 \$ 08 与 \$ 904 中的 \$ 09,由于内存地址的问题,地址的高 8 位应根据地址值高位予以适当的修改。即当我们想把输入好的程序 1,拷贝至 \$ 900 作程序 2 时,应做如下工作:

* 900<800. 804M

(把 \$ 800—\$ 804 单元的内容拷贝至 \$ 900 开始的内存)

* 904,9

(把 JMP 的高 8 位地址改为 \$ 09,即 \$ 904)

象上面这样,一个 B 型文件要在不同地址的内存中运行,就要做较大的修改,由于这类问题,R 型文件应运而生,我们先看看 LOADAPA 程序的调用段:

```
140 PRINT CHR $(4);"BLOAD RBOOT"
```

```
150 CALL 520
```

```
160 ADRS=USR(0),"APA"
```

```
180 CALL ADRS
```

下面逐一分析各行程序:

140 行:调入 RBOOT 程序,该程序首地址为 \$ 208 (520)

150 行:启动 RBOOT,设定 USR 地址,调入 RLOAD 文件。

160 行:用 USR 命令调入 APA,这里对名为任意的 filename 文件,都可采用下法调用:

行号 ADRS=USR(0),"filename"

变量 ADRS 是程序的首地址,RLOAD 把要调入的程序转移至末尾地址为 HIMEM-1 的地址中,并按绝对地址修改好 APA 程序,这样,就可保证程序可在任意内存运行,如不想要 APA 的标题显示,在 180 行中用 CALL ADRS+3 即可。

180 行,调入 APA,设定 & 命令转移地址。

这里应介绍一个基本知识,即 B 与 R 型文件是不同的,B 型文件(如果是程序的话)一般只能在某一固定内存运行,而经 RBOOT 处理后的 R 型文件,则可以在任意内存中运行。

R 型的文件用处有很多,由于不想牵扯太远,故只介绍到此。

最后,简单介绍苹果公司编写的 3 个 R 型文件:

1)APA,是一个实用的程序设计辅助工具,收录在 TOOL—KIT 中。

2)HRCG,是一个可在高分辨页显示 ASCII 字符集的文件,也收在 TOOL—KIT 中。

3)ASSEMBLER,是一个可在 APPLE 机运行的汇编文件,使用方便,功能齐全,收在 ASSEMBLER 软件中,该软件的简化版本 EDASM 亦已收在 TOOL—KIT 中。

2. 程序设计辅助工具——APA

APA 程序全名为 Applesoft Programmer's Assistant,取其第一个英文字母,名为 APA。由于它是一个 R 型文件,因此只能用前文所述方法或执行 LOADAPA 调用。步骤如下:

1)插入 TOOL—KIT 盘,开机,按 E 键

2)打入 RUN LOADAPA,回车

3)过约 30 秒后,APA 调入内存。

下面的字幕显示出来:

```
APPLESOFT PROGRAMMER'S ASSISTANT
VERSION 1.0
(C) COPYRIGHT 1979, APPLE COMPUTER INC.
& RENUMBER(START),(INC),(FIRST),(LAST)
& HOLD
& MERGE
& COMPRESS
& SHOW
& NOSHOW
& AUTO(START),(INC)
& MANUAL
& XREF
& KEYS
```

屏幕上半部是版权说明,下半部是命令列示;下面,我们逐一介绍它们:

1)& RENUMBER 重排后第一行行号,重排后行号增量,欲重排程序始行号,欲排程序末行号:

它是用来重编一个程序的行号的,在 DOS3.3 里,有一个 RENUMBER 程序,这个程序里有个致命的弱点,请看下面的程序(在调入 DOS3.3 的 RENUMBER 环境下运行):

```
10 A=A*10
20 PRINT A
30 END
]& F100
]LIST
100 A=A*100
110 PRINT A
```


120 END

A=A*10 莫名其妙地变成了 A=A*100,这是由于 RENUMBER 设计的失误,错把“*”号当成转移命令了。而在 APA 中的 &RENUMBER 就没有这个问题。请看程序 1、2。

```
程序 1: ]LIST
        10 A=A*10
        20 PRINT A
        30 END
```

```
程序 2: ]& RENUMBER 100,10,10,30
        ]LIST
        100 A=A*10
        110 PRINT A
        120 END
```

如省略其中一个参量,只需在两逗号之间什么字符也不打就可以了。如下例:

```
& RENUMBER 100,10,,
```

则与 & RENUMBER 100,10,0,63999 相同,省略值如下:

第一个参数:100 第二个参数:10
第三个参数:0 第四个参数:63999

本命令只修改 GOTO, GOSUB, ON.....GOTO, ON.....GOSUB, ONERR.....GOTO 和 THEN 后面的行号, REM, LET, PRINT 后面的数字不予修改,另外如果有输入错误,或在程序中执行本命令,如下例:

```
]& RENUMBER 100,100,110,1000
```

这里,我们把新旧两个程序的行号交叉了,系统显示:

```
INTERLEAVED OR DUPLICATE LINE NUMBER
```

系统不会修改这个程序中的任何行号。

2)& AUTO 开始行号,行号增量

此命令将模拟 INT BASIC 的 AUTO 命令,自动产生行号,键入命令后,按下回车键,如果你紧接回车键按空格键,那么系统将显示开始行号,输完内容,按回车键,再紧接空格,系统又显示下一行号.....如果回车之后不紧接空格,那么系统还将按你的命令执行。如下例:

```
]& AUTO 10,10
] 10 HGR
] 20 HCOLOR=3
] 25 SCALE=1
] 30 DRAW 1 AT 140,80
] 35 FOR I=1 TO 1000:NEXT
] 40 XDRAW 1 AT 140,80
] 50 END
] LIST
10 HGR
20 HCOLOR=3
25 SCALE=1
30 DRAW 1 AT 140,80
35 FOR I=1 TO 1000:NEXT
40 XDRAW 1 AT 140,80
50 END
```

在 & AUTO 命令里,如省略第二个参数,系统按 10 计算,& AUTO 有一个奇怪的现象,当你键入 &ABC 后,也会与 &AUTO 产生同样的效果。这里,25、35 句前没有空格,因此,产生的不是整十的行号,而是用户自己输入的行号。

如果想取消 & AUTO 的功能,键入 & MAUAL 即可。

3)& MANUAL

执行这个命令后,系统退回原来的输入方式。

4)& XREF

列出这个程序中共用了哪些变量名及所在的行号。实型量直接用字母表示,整形量加“%”表示,字串量加“\$”表示,数组加“(”(左括号)表示。如:

```
10 A=10
20 B%=20
30 C$="MY NAME IS C$"
40 ABC(3,4)=5
50 DEE$(9,5)="MY NAME IS DE$"
60 ZZZ%(2,3)=100
]& XREF
```

```
A 10
AB( 40
B% 20
C$ 30
DE$( 50
ZZ$( 60
```

这里,变量名按英文字母的次序排列显示时只取前两个字符;行号在变量名的右边。

5)& KEYS

早期的苹果机,键盘上没有下横线“-”,反斜杠“\”及左方括号“[”,结果,连自己的商标“APPLE [”都产生不了。有了 & KEYS,我们可以用下面的键代替它们:

CTRL-O 产生“-”字符

CTRL-K 产生“[”字符

CTRL-L 产生“\”字符

如果想取消 & KEYS 的功能,键入 & MANUAL 即可。

6)& SHOW

该命令可显示控制字符,有些磁盘用控制字符加密文件名,也可用它解密,CTRL-字符的列印结果用反白显示。

7)& NOSHOW

恢复原来的显示模式,即取消 & SHOW 的效果。

8)& COMPRESS

有时,一个程序有许多 REM 语句,很占内存,为了节省内存,可用 & COMPRESS 命令清除 REM 语句。

9)& LENGTH

打印 BASIC 程序的长度。如:

```
]& LENGTH
PROGRAM LENGTH IS 131($ 0083) BYTES
10)& HOLD
```

将程序移到可用最高地址 HIMEM 的正下方,修

改 HIMEM, 将其保护起来。

(11)&. MERGE

恢复掩盖的程序。该命令可将两个程序连接起来:

]LIST

10 A=5

20 B=3

]&. HOLD

]LIST

]LOAD A-2

]LIST

30 C=4

40 D=7

50 PRINT A,B,C,D

]&. MERGE

]LIST

10 A=5

20 B=3

30 C=4

40 D=7

50 PRINT A,B,C,D

这些功能是怎么实现的呢?下面逐一分析它们,以便大家更好的利用。

(1)&. AUTO

本指令将 \$ 38, \$ 39 单元的值改至一个小程序的首地址。该子程序判断是否键盘上输入的是“回车+空格”,如是,显示第一个参数,同时把第一个参数值加上第二个参数值,等待用户输入,如此循环,便实现了上面所述的功能。

(2)&. MANUAL

本指令将 \$ 38、\$ 39 中的值换回原值(\$FD1B),不做任何检查。所以,它有切掉 &. AUTO 和 &. KEYS 功能。

(3)&. RENUMBER

本指令从 \$ 803 单元开始读每一字节,首先,判断其值是否为 \$ AB(GOTO 代码), \$ B0(GOSUB), \$ A5(ONERR), \$ C4(THEN)之一,如是,参照其后的参数把这些地址后面两字节修正,如不是,地址加 \$ 1,继续反复,如此循环,改正行号的工作就做好了。

(4)&. HOLD

此指令用 MOVE 程序把 BASIC 程序从程序区搬到(HIMEM—程序长度)的区域,执行这个指令后,FP、MAXFILES、HIMEM;等指令将不能执行,否则,程序有可能丢失。

(5)&. MERGE

此指令将藏起的程序和内存程序区的程序结合起来,同时根据行号大小将程序重新安排,如遇到重复的行号,就提出警告。

(6)&. COMPRESS

本指令从程序区开始处一直找到 PRGEND(\$AF、\$B0 单元指示的地址),如遇到 \$ B2(REM),就从这个单元开始删,遇到 \$ 00(一行的程序结束的标志),或遇到 \$ 3A(“:”代码)为止。如已找完,把空白的

单元删除,然后将消去的总单元数目打印出来。

(7)&. LENGTH

打印 \$ AF、\$ B0 的值减去 \$ 67、\$ 68 的值。

(8)&. SHOW

类似 &. AUTO, &. SHOW 改动的是 CSW(\$36, \$37 单元)的值,改动后的子程序检查输入、输出的是否是控制字符(\$81~\$9F),如是,则执行 AND # \$3F,即显示相应键符的反相字。

(9)&. NOSHOW

把 CSW 的值改为 \$ FDF0。

(10)&. KEYS

本指令修改 KSW 单元(\$38, \$39)的值,检查键符并作如下的变换:

CTRL- 0: \$ 8F → \$ DF(“-”)

CTRL- K: \$ 8B → \$ DB(“[”)

CTRL- L: \$ 8C → \$ DC(“\”)

如执行 &. AUTO, 则 &. KEYS 失效,如 &. KEYS 先执行,则 &. AUTO 亦失效。也就是说, &. AUTO 与 &. KEYS 不能兼容。

(11)&. XREF

本程序从程序区开始扫描,遇到变量之形式的内存单元,就将其挪出来。许多读者对变量存放格式很感兴趣,在这里,顺便介绍一下:

A 实数变量 例: A=40000

单元组	内容含义	内容
0	变量名的第一个字元	\$ 41
1	变量名的第二个字元	\$ 00
2	指数部分	\$ 90
3	小数部分	\$ 1
4		\$ 40
5		\$ 00
6		\$ 00

B. 整型变量 例: B%=257

单元组	内容含义	内容
0	变量名第一、二字元	\$ C2
1		\$ 80
2	变量值高位字节	\$ 01
3	变量值低位字节	\$ 01
4	未用	\$ 00
5		\$ 00
6		\$ 00

C. 字符变量 例: CD\$="ABC"

单元组	内容含义	内容
0	变量名第一字元(负 ASCII 码)	\$ 43
1	变量名第二字元(正 ASCII 码)	\$ C4
2	长度	\$ 03
3	字符串首地址低位字节	
4	字符串首地址高位字节	
5	未用	\$ 00
6		\$ 00

D. 整型数组变量 例 JK%(2,3)

单元组	内容含义	内容
0	名称第一字元(负 ASCII 码)	\$ CA
1	名称第二字元(负 ASCII 码)	\$ CB

2	长度低位	\$ 21
3	长度高位	\$ 00
4	数组维数	\$ 02
5	后一维长度高位	\$ 00

6	后一维长度低位	\$ 04
7	前一维长度高位	\$ 00
8	前一维长度低位	\$ 03

(未完待续)

利用 BOOT 程序对出租软件进行计次使用

无锡西潭压力容器厂技术科(214171) 任晓方

随着软件交流形式的多样化,出租软件也将成为一种使用形式。为了保护出租软件的权益,通常采用限时限次的方法来加以限制,这就产生了一个问题,即怎样才能确定使用次数。较为简单的方法是建立一个限时使用的计数文件,当其记录的使用次数达到额定的使用次数后,就停止其使用,或者是清除内存,或者干脆 INIT 磁盘,将其软件破坏掉,但是这种计次方法的计数文件容易被破解,也比较繁琐。

我通过分析 DOS 的引导过程,设计出了一种新的计次方法,即在引导 DOS 时就显示出该软件允许使用的次数,每次引导 DOS 时该使用次数自动递减,当使用完规定的次数后,DOS 就不能正常引导,而直接进入机内的 ROM BASIC 冷起动程序。

基本原理:我们知道 DOS 的引导过程是分三步进行的,第一步是执行软盘驱动器接口卡上的 ROM 引导程序,将软盘上的 0 道 0 扇区内容(即 BOOT1)读入内存,这一过程我们称做 BOOT0;然后执行 BOOT1 将软盘上的 0 道的 1—9 扇区内容(即 BOOT2)读入内存,再执行 BOOT2 将整个 DOS 读入内存,转入 DOS 的冷起动程序执行,至此 DOS 引导完毕。

在 DOS 中 \$BB00—\$BC55 为数据缓冲区,用来存放读/写磁盘的数据转换之用,位于磁盘上 0 道第 5 扇区和第 6 扇区的一部分,这一部分扇区的内容在引导 DOS BOOT1 阶段时不起任何作用,但仍被读入内存,因此我们可以在这一部分扇区内设计一段软件计次使用程序,在 BOOT1 完后执行该段程序,将使用次数单元值自动减 1,并利用已读入的 RWTS 程序修改磁道上的计次单元值。若计数单元值不等于 0,则转入 BOOT2 继续引导 DOS,否则转入 BASIC 冷起动程序执行,DOS 失效。

具体改造和使用方法如下:首先引导 DOS 格式化一张从盘,再用 COPY][工具软件或 LOCKSMITH 等其他能改写磁盘扇区的工具软件,将 0 道 0 扇区和第 5 扇区按下表划线部分修改后仍存回原扇区,全部修改完毕后这张盘就有了上述计数功能了,第 5 扇区的 \$DF 字节是设定的计次数,本例中为 \$12,使用次数为 \$DF 单元的值减 1,如(\$DF)=\$12,则可以使用 17 次。当其值为 1 时,则不能引导 DOS,转入机内的

BASIC 冷起动程序执行,从而使得出租软件得到保护。在程序中我另外在 \$1B00—\$1C55 开辟了一段数据缓冲区,以保护位于 \$BB00—\$BBFF 中的程序不被破坏,DOS 引导完后位于 \$BB00—\$BBFF 中的程序自动失效,该区域仍为 DOS 的数据缓冲区,用此盘引导的 DOS 格式化从盘则无上述功能,同正常 DOS 一样。

由于本程序只占用数据缓冲区所在扇区,所以对其他修改后的 DOS 从盘仍适用。另外,数据缓冲区共有 342 个字节,有较大的空间,可以设计出其他利用 DOS 的方法,如还可以设计一段程序进一步将使用次数已到的磁盘进行格式化,从而彻底保护出租软件,此加密方法同其他加密方法结合起来使用,能起到绝妙的效果。有兴趣的同志不妨试一试。

```

TRACK=0    SECTOR=0
00: 01A5 27C9 09D0 18A5 2B4A 4A4A 4A09 C085
10: 3FA9 5C85 3E18 ADFF 086D FF08 8DFE 08AE
20: FF08 3015 BD4D 0885 3DCE FF08 ADFF 0885
30: 27CE FE08 A62B 6C3E 00EE FE08 EEFE 0820
40: 89FE 2093 FE20 2FFB A62B 4C00 BB00 0D0B
50: 0907 0503 010E 0C0A 0806 0402 0F00 2064
60: A7B0 08A9 00A8 8D5D B691 40AD C5B5 4CD2
70: A6AD 5DB6 F008 EEBD B5D0 03EE BEB5 A900
80: 8D5D B64C 46A5 8DBC B520 A8A6 20EA A24C
90: 7DA2 A013 B142 D014 C8C0 17D0 F7A0 19B1
A0: 4299 A4B5 C8C0 1DD0 F64C BCA6 A2FF 8E5D
B0: B6D0 F600 0000 0000 0000 0000 0000 0000
C0: 0000 0000 0000 0000 0000 0000 0000 0000
D0: 2058 FCA9 C220 EDFD A901 20DA FDA9 AD20
E0: EDFD A900 20DA FD60 0000 0000 0000 0000
F0: 0000 0000 0000 0000 0000 0000 0000 B609

```

```

TRACK=0    SECTOR=5
00: 2058 FCA9 0820 5BFB A900 8524 CEDF BBD0
10: 1020 B8BB A225 20D0 BBA2 60BD 88C0 4C00
20: E020 B8BB A91C A01B 20E0 BBA9 B7A0 E884
30: 4885 49A9 00A0 0491 48A9 05C8 9148 A902
40: A00C 9148 A900 A008 9148 C8A9 BB91 4820
50: 04BD A9BC A0BB 20E0 BBA2 60A9 6C8D 4AB6
60: A9FD 8D4B B6A9 088D 4CB6 4C00 B760 0000
70: D4E8 E9F3 A0F0 F2EF E7F2 E1ED A0E9 F3A0

```

80: E1EC ECEF F7E5 E4A0 F5F3 E5A0 00A0 F4E9	C0: DFBB 2024 EDA2 1D20 D0BB 6000 0000 0000
90: EDE5 F3A1 008D 8D8D 8787 8787 A0A0 A0A0	D0: BD70 BBF0 0620 F0FD E8D0 F560 0000 0012
A0: A0A0 C9EC ECE5 E7E9 F4E9 EDE1 F4E5 A0F5	E0: 8D0A B88D 0EB8 8D20 B88D 25B8 8D3A B88D
B0: F3E5 A100 0000 0000 A200 20D0 BBA9 00AE	F0: 68B8 8C11 B88C 6BB8 8C80 B88C 90B8 6000

中华学习机汉字双页显示

安徽蚌埠医学院(233003) 蒋建一

中华学习机是具有汉字处理功能的计算机。它提供了汉字的输入、显示及打印等多项功能,为用户编制具有汉字处理功能的程序带来了方便,但它只能在高分辨率图形第二页显示汉字,而不能随意选用第一页或第二页,使得其汉字的显示有一定的局限性。

通过应用和分析学习机汉字系统程序,我编制了一个可实现汉字双页显示随意选择的汇编语言程序。程序使用说明:首先按所附程序清单键入全部机器码程序并存盘保存。以后只要 BRUN 该程序,便可自动进入汉字系统,并选用高分辨率第一页作为汉字显示页。

1C00- A9 00 8D 7B 04 20 00 C3
1C08- A9 00 85 06 A9 EC 85 07
1C10- A0 00 20 AB C3 B1 06 20
1C18- B9 C3 48 AD 83 C0 AD 83
1C20- C0 68 91 06 8D 82 C0 8D
1C28- 82 C0 C8 D0 E5 E6 07 D0
1C30- DF AD 81 C0 AD 81 C0 A9
1C38- 05 8D F4 F1 A9 E6 8D F5
1C40- F1 A2 6F CA F0 13 BD FF
1C48- 1C 85 06 CA BD FF 1C 85
1C50- 07 A0 00 A9 1F 91 06 D0
1C58- EA A9 1D 8D 88 F2 2C 82
1C60- C0 2C 82 C0 A9 20 85 E6
1C68- 20 F2 F3 A9 91 85 3C A9
1C70- 1C 85 3D A9 FF 85 3E A9
1C78- 1C 85 3F A9 00 85 42 8D

1C80- F6 03 A9 03 85 43 8D F7
1C88- 03 A0 00 20 2C FE 4C 14
1C90- 03 C9 31 F0 10 C9 32 F0
1C98- 4B 4C 76 DD 60 A9 20 85
1CA0- E6 2C 54 C0 60 20 0C 03
1CA8- 4C 5D 03 2C 52 C0 2C 57
1CB0- C0 A9 96 85 74 A9 33 85
1CB8- 36 A9 43 85 38 A9 03 85
1CC0- 37 85 39 60 8D 80 C0 8D
1CC8- 80 C0 20 CE EF 8D 82 C0
1CD0- 8D 82 C0 60 8D 80 C0 8D
1CD8- 80 C0 20 A3 F7 8D 82 C0
1CE0- 8D 82 C0 60 A9 40 85 E6
1CE8- 2C 55 C0 4C 17 03 2C 50
1CF0- C0 68 68 A9 E0 48 A9 02
1CF8- 48 4C 1A 03 00 00 00 00

1D00- ED 7A F0 11 F0 1B F0 26
1D08- F0 D4 F1 88 F3 64 F3 67
1D10- F3 E0 F3 E5 F4 1D F4 33
1D18- F4 6C F4 71 F4 76 F4 80
1D20- F4 85 F4 8F F4 98 F4 9F
1D28- F4 AA F4 AF F4 B2 F4 B7
1D30- F4 BE F4 C8 F4 D2 F4 E5
1D38- F4 EC F4 F3 F4 F8 F5 03
1D40- F5 08 F5 0B G5 0E G5 13
1D48- F5 16 F5 1B F5 2B F5 35
1D50- F5 4F F5 5A F5 98 F5 9F
1D58- F6 01 F6 61 F6 66 F6 AA
1D60- F6 B8 F6 DE F7 02 F7 10
1D68- F7 36 F7 63 F7 7E 00 00

若想换到第二页显示,只要键入 &2 便会立即转为第二页显示。(注:汉字显示页转换命令为 &1 选择第一页,&2 选择第二页)。无论何时,只要键入 &1 或者 &2 就能方便地实现汉字显示页的转换。本程序只要运行一次后,其内存中所占空间便可供再使用,不需保留。另外,本程序将汉字显示映射区改在 \$1D00 ~ \$1FFF。当选用汉字第一页显示时,用户空间为 \$800 ~ \$1CFF,和 \$4000 ~ \$9600。

敬告《电子与电脑》读者

今后凡读者来函、来款均请按下列地址及收件人邮寄:邮编 100036 北京 173 信箱 杂志收,并注明读者当地详细地址及邮编,字迹要清楚。

高速排序程序的使用技巧

山东淄博五中 33 级 1 班(255028) 张世林

在诸多排序方法中,笔者认为使用汇编语言编写的高速排序程序(以下简称高速排序程序)优点最大:

1. 使用简便。在你编写的 BASIC 程序中,只要有一句调用高速排序程序的命令,就可对指定的数组排序。2. 排序速度快。例如这篇文章中引用的高速排序,对具有 1000 个元素的一维实型数组排序,仅需 5 秒钟!

以上两条优点就大大方便了数据处理程序的编写及调试,下面就以《中华学习机》杂志 1990 年第 4 期刊登的汇编程序在考分统计中的应用为例,说明高速排序程序的使用技巧。

先简要介绍一下高速排序程序的使用方法:首先运行高速排序程序,然后在你所编的程序中加入: & 数组变量名(a),b。这样就可对指定的一维数组中,第 a 至 a+b 项,共 b+1 个数据进行排序。(注:高速排序程序及示范程序见附录一,其中示范程序的第 10 句及第 15 句是运行已存在磁盘中的高速排序程序,其文件文为:PMC'HART NEW)。

当您把附录中的高速排序程序输入计算机,存入磁盘就可下面介绍的两种方法,得心应手的使用它了。

方法之一:二次排序法

这种方法的设计思想是,把所要排序的数据,也就是各科考试的分数放入 2 个数组(这两个数组所放数据完全一样),再对其中一个数组进行快速排序,然后根据已排好次序的数组中各个数据在原数组中的位置,找出每个数据所对应的其它相关数据,如学号、姓名、性别等等。程序一就是根据这个设计思想编写的。它的功能就是对 N 名学生, Q 个学科的总成绩排序,并打印出名次表。

程序如下:

程序一:

```
10 INPUT N,Q;N=N-1;D$=CHR$(4)
20 DIM A(N),S(N),Y%(N)
30 FOR I=0 TO N:READ Y%(I)
40 FOR J=1 TO Q:READ A:A(I)=A(I)+A: NEXT J
50 S(I)=A(I): NEXT I
60 PRINT D$;"BRUN PMC'HART 2.0";&A(0),N
80 FOR I=0 TO N:V=0:PRINT I+1;:FOR J=0 TO Q
100 V=V+1:PRINT TAB(7);Y%(J);TAB(14);INT(S(J)/Q*100+.5)/100;TAB(23);A(I)
110 NEXT J;IF V>1 THEN I=I+(V-1)
```

120 NEXT I

200 DATA 1,89,99,78,2,78,65,67,3,88,99,100,4,57,86,97,5,78,65,67

]RUN

? 5,3

1	3	95.67	287
2	1	88.67	266
3	4	80	240
4	2	70	210
	5	70	210

程序一的几点说明:

1. 10~20 句是输入人数(N)和科目数(Q),并定义数组大小。

2. 30~50 句是读数据,把学号放入 Y%(N)中,每个学生的总分放入 A(N),S(N)。

3. 60 句运行排序程序对指定的 A(N)数组排序。

4. 80~120 句是第二次排序(对学号等其它相关数据)并打印名次表。变量 V 是用来避免名次并列时的重复打印。这种形式决定了名次并列时,学号小的在学号大的前面(见运行结果中的第 4 名)。

5. 200 句的数据存放格式为:学号在前,各科成绩依次在后。

6. 运行结果的 4 个纵行数据自左向右分别为:名次、学号、平均分、总分。

这种方法的优点在于程序简单,明白。但美中不足的是当人数一多,第二次排序时间必然会增长,如果您有一台高速打印机,就会出现打印机等主机的不正常现象。使用方法二,这种现象将根本消除。

方法之二:编码排序法

这种方法的设计思想是把所要排序的数(在此是考试总分)放大一定倍数,使它成为一个整数,把代表它在数组中的位置的数缩小一定倍数加在整数后面,成为整数的小数部分。再对这个“新”的数进行排序,排序完后再根据这个“新”数的整数与小数部分,求出实际考试总分,及对应的其它相关数据的位置(即这个新数在数组中未排序之前的位置),从而打印出名次表。这种设计思想是依据高速排序程序在排序时,总是从高位开始,高位数字大的必然大这一特点设计的。这样虽然各个数据在数组中位置不一样,但是把它们的位置作为低位数字加在各个数据后,对排序结果是毫无影响的。但必须注意的是,原来的数一定要扩大成为整数,代表位置的数一定要缩小为小数(即零点几)。这样相加所组成的“新”数才具有意义。这个过程很象编码,解码,因此我们就不妨称之为“编码排序”。程序

二就是根据这个设计思想编写的。它可对全年级学生排名次,并据各班前50名的总分,作为班级成绩,进行排序。(注:这个“前50名”是可以任意改变的),程序如下:

程序二:

```

5 D$ = CHR$(4); INPUT "How many students? "; N;
  N = N - 1; INPUT "How many classes? "; Q; DIM
  A(N), Y%(N), S(Q); Y1 = 10; Y2 = 10000
10 FOR I = 0 TO N; READ Y%(I), A(I); IF Y%(I) <
  1000 OR A(I) > 1000 THEN PRINT I + 1, Y%(I)
  + 1, A(I + 1); STOP
20 X = VAL (RIGHT$ (STR$ (Y%(I)), 2)); Y =
  VAL (MID$ (STR$ (Y%(I)), 2, 1)); IF X <
  = 50 THEN S(Y) = S(Y) + A(I); R(Y) = R
  (Y) + 1
30 A(I) = A(I) * Y1 + I / Y2; NEXT I; FOR I = 0 TO
  Q - 1; S(I) = S(I + 1) * 100 + I; NEXT I; PRINT
  : PRINT D$ "BRUN PMC'HART 2.0"; & A(0), N;
  & S(0), Q
35 INPUT "Hard copy? (Y/N) "; A$; IF A$ = "Y"
  THEN PRINT : PRINT D$; "PR #1"; PRINT
40 PRINT "1"; FOR I = 0 TO N; J = INT (A(I) *
  Y2 - INT (A(I)) * Y2 + .5); A(I) = INT (A
  (I)) / Y1; IF I > 0 THEN IF A(I) < > A(I - 1)
  THEN PRINT " "; I + 1;
50 PRINT TAB(8) Y%(J) TAB(17) A(I); NEXT I;
  PRINT : PRINT
55 PRINT : PRINT "1"; FOR I = 0 TO Q - 1; J =
  VAL (RIGHT$ (STR$ (S(I)), 1)); S(I) = (S
  (I) - J) / 100; IF I > 0 THEN IF S(I) < > S(I -
  1) THEN PRINT " "; I + 1;
60 PRINT TAB(8); J + 1 TAB(14) S(I); TAB(27)
  INT (S(I) / R (J + 1) * 100 + .5) / 100; TAB(
  37) R (J + 1); NEXT I; PRINT : PRINT D$; "PR #
  0"
200 DATA 1102, 56, 1102, 89, 1103, 98, 1201, 98,
  1202, 98, 1203, 90

```

] RUN

How many students ? 6

How many classes ? 2

Hard copy

? (Y/N) Y

1	1202	98
	1201	98
	1103	98
4	1203	90
5	1102	89
6	1101	56

1	2	286	95.33	3
2	1	243	81	3

程序二的几点说明:

1. 200句的数据存放格式为学号在前总分在后。学号的构成为:ABXX。A代表年级,B代表班级,XX代表班级名次。例如,高一年级一班第8名,其学号就是

1108。

2. 第5句是输入年级总人数N,班级数Q,A(N)放总分,Y%(N)放学号,S(Q)放各班前50名总分。变量Y1、Y2是在编码,解码时用。

3. 第10句中的条件语句,是判断所读数据有无错误,如有错误则立即打印出错误位置及总分、学号,并暂停。

4. 变量X、Y是根据学号构成特点,从学号中提取的名次及班级。

5. 第30句是对A(N),S(Q)编码,并排序。

6. 第40~50句是对A(N)解码,同时打印年级名次表。这种排序方式决定了名次并列时,学号大的在学号小的前面。

7. 第55~60句是对S(Q)解码,同时打印班级名次表。

8. 运行结果中三个纵行自左向右分别为名次、学号、成绩。下面的五个纵行自左向右分别为名次、班级、总分、平均分、人数。

应注意的是解码过程中,由于APPLE机的运算误差,因此解码不能是编码运算式的简单逆运算过程,请读者着重看一下程序二中的第40句、第50句的解码运算式。

当您掌握以上两种方法的原理,并应用到自己所编的程序上,您会发现:原来需要运行1个多小时,甚至几个小时的程序,现在只需十几分钟,并且还可以在在一定程度上缓解内存紧张问题。

如果有的读者想把这两种排序方法运用到中华学习机上,在程序中加入汉字,那么您可以用附录二中的汇编程序。(摘自《学生计算机世界》)。这是因为附录一中的汇编程序的结尾一部分,占用了中华学习机的汉字管理程序的工作单元,在汉字状态下运行程序会出现“死机”现象。

程序选自《学生计算机世界》

附录一:

```

0300- A9 0B 8D F6 03 A9 03 8D
0308- F7 03 60 20 E3 DF 84 3D
0310- 85 3C 20 BE DE 20 67 DD
0318- 20 52 E7 A5 50 85 3E 85
0320- 40 A5 51 85 3F 85 41 A7
0328- 41 87 3A 87 3B A5 40 46
0330- 41 6A A8 05 41 F0 D3 98
0338- 85 40 0A 0A 86 FB 26 FB
0340- 65 40 85 FA A5 FB 65 41
0348- 85 FB A5 3E 38 E5 40 85
0350- 42 A5 3F E5 41 85 43 A5
0358- 3B A4 3A 85 4E 84 4F 85
0360- FF 98 0A 26 FF 0A 26 FF
0368- 65 4F AA A5 FF 65 4E A8
0370- 8A 65 3C AA 98 65 3D A8
0378- 8A 20 F9 EA 18 A5 5E 65
0380- FA 85 60 A5 5F 65 FB 85
0388- 61 20 B6 EB B0 12 E6 3A

```

```

0390- D0 02 E6 3B A5 42 C5 3A
0398- A5 43 E5 3B 90 89 B0 B7
03A0- A0 04 B3 5E B1 60 91 5E
03A8- 8A 91 60 88 10 F4 A5 4F
03B0- E5 40 A8 A5 4E E5 41 90
03B8- D5 D0 A0 A5 4F C5 40 90
03C0- CD A9 00 B0 96
10 PRINT D$ = CHR$(4); PRINT
15 PRINT D$; "BRUN PMC'HART NEW"
20 INPUT N; DIM A(N); FOR I = 1 TO N; A(I) =
    RND(1) * 1000; PRINT A(I); NEXT
25 & A(0), N
27 PRINT; PRINT
30 FOR I = 0 TO N-1; PRINT A(I); NEXT
] RUN
?5
360.889731
628.262312
581.121379
768.268873
723.68035

```

```

768.268873
723.68035
628.262312
581.121379
360.889731

```

附录二

```

8FF0- A9 4C 8D F5 03 A9 00 8D
8FF8- F6 03 A9 90 8D F7 03 60
9000- 20 E3 DF 85 EC 85 EE 84
9008- ED 84 EF 38 E9 02 85 85
9010- B0 01 88 84 86 A0 01 B1
9018- 85 85 FE 88 B1 85 85 FF
9020- A9 01 8D B0 90 A9 00 8D
9028- B1 90 F0 11 18 A5 EC 69
9030- 05 85 EC 85 EE A5 ED 69
9038- 00 85 ED 85 EF A5 EC A4
9040- ED 20 F9 EA AE B0 90 AD
9048- B1 90 8D B2 90 20 7E 90
9050- E8 D0 03 EE B2 90 E4 FE
9058- D0 F3 AD B2 90 C5 EF D0
9060- EC 18 AD B0 90 69 01 8D
9068- B0 90 D0 03 EE B1 90 AD
9070- B0 90 C5 FE D0 B6 AD B1
9078- 90 C5 FF D0 A6 60 18 A5
9080- EE 69 05 85 EE 90 02 E6
9088- EF A5 EE A4 EF 86 FA 20
9090- B2 EB A6 FA C9 FF F0 01
9098- 60 A0 04 B1 EC 48 B1 EE
90A0- 91 EC 68 91 EE 88 10 F3
90A8- A5 EC A4 ED 4C F9 EA

```

```

10 INPUT N; DIM A(N)
20 FOR I = 0 TO N; A(I) = RND(1) * 1000; PRINT

```

```

A(I); NEXT; PRINT
50 CALL 198; & A(0); CALL 198
60 FOR I = 1 TO N; PRINT A(I); NEXT
] RUN
?7
594.331536
130.726507
834.264561
567.285335
920.549918
464.319447
829.422036
571.668958

834.264561
829.422036
594.331536
571.668958
567.285335
464.319447
130.726507

```

汉字编码打印程序

宝鸡市金陵路21号(721001)李 铁

中华学习机扩展操作系统 V2.11 以上版本可以使用五笔,表形,苍吉,首尾,电报等十余种编码输入汉字。但是,在以前发行的软盘中,没有专门的编码打印程序,对于初学汉字编码输入汉字的用户很不方便。为此,特编了下面这个汉字编码打印程序,使用如下:

启动 CEC-I SC V2.5 系统,调入需要打印的编码,然后执行本程序,便可为你打印一份码表。程序中,U 为每行字数,VV * K 为每页600字。ADD 为码表参数起址,V2.11 版改为 \$ACA7。

```

10 REM PRINT CHINESE CODE
20 IO#2;PRINT; HOME; PRINT
30 U = 10; VV = 6; K = 100
40 ADD = 9 * 4096 + 15 * 256 + 10 * 16 + 4
50 DEF FN A(X) = X + 28 + (X > 5) + (X > 14) + (X > 27)
60 D = PEEK(ADD + 10); Z = 64; S = 160
70 L = PEEK(ADD + 9)
80 I = PEEK(ADD + 14) - 159 + (PEEK(ADD + 15) - 160) * 100
90 T = PEEK(ADD + 12) + PEEK(ADD + 13) * 256 + D
100 BL = 2 ^ PEEK(ADD + 11)
110 M$ = "<<"; FOR E = 0 TO 5; S IF PEEK(ADD + E) < 128 THEN M$ = M$ + CHR$(PEEK(ADD + E)); ELSE; M$ = M$ + CHR$(FN A(PEEK

```

```

(ADD+E)-160))
120 NEXT E;M$=M$+"码>>"
130 POKE 1659,1;PRINT
140 FOR A=1 TO 1+4096
150 X=INT(A/K);E=FN A(X)

160 X=A-X*K;F=FN A(X)
170 IF X<>95 THEN 230
180 PRINT :A=A+5;S1=S1+1; IF S1<>VV
    THEN 380
190 S1=0;S2=S2+1
200 GOSUB 400
210 POKE 1659,0;INPUT A$;POKE 1659,1
220 GOTO 380
230 X=(A-I)*D
240 IF X/U=INT(X/U)THEN PRINT;
    PRINT I+X/D;"-";
250 PRINT CHR$(127);CHR$(E);CHR$(F);
260 X=X+T;F=0
270 FOR E=D-1 TO 0 STEP -1
280 F=F*256+PEEK(X+E)
290 NEXT E
300 A$=" ";FOR E=1 TO L
310 X= INT(F/BL^(L-E))
320 S=PEEK(ADD+17+X)
330 IF S=0 THEN A$=A$+" "
340 PRINT CHR$(S);
350 F=F-BL^(L-E)*X
360 NEXT E
370 PRINT A$;
380 NEXT A
390 GOSUB 400;POKE 1659,0;END
400 PRINT :PRINT :PRINT
410 PRINT SPC(20);M$;SPC(10);"区位:";A-VV
    *100;"-";A-1;SPC(5);"第";S2;"页"
420 RETURN

```

对《CEC—I 键控光 标作图程序》的改进

北京永外建筑磨石二车间(100075)李庆岱

1991年第12期《电子与电脑》“学习机之友”专栏发表的陈国良同志的《CEC—I 键控光标作图程序》，很有创意，为计算机绘图爱好者提供了一个非常好的工具。本人在学习该程序中，发现了几点可改进之处。

首先，原程序没有控制键用途说明，用时颇觉不便。为免去时时查书之苦，我试编了一个说明模块，见所附程序700—745语句。

其次，大多数计算机绘图爱好者，都有保存满意图形，供以后调用，欣赏的愿望。原程序无此功能，颇觉遗憾。我试加了一个存盘模块，见400—440语句。

再次，由于使用方向不同，我们往往调用盘中现有图形，经改进后加以利用。如能在同程序中调用已有的同系统下的图形，将是非常便利的。我尝试给原程序加了一个调盘模块，见500—520语句。它可以在调出图形后显示出来，并进入绘图状态，以利修改、欣赏。

最后，原程序无结束运行功能，只能按“CTRL—RESET”强行中断，似觉不完整。我为它增加了600句，能在结束时，问候使用者，亲切而方便。

几个模块的转向控制，利用了原程序语句行号的空置间隔，略增了几句。

为节约篇幅，原程序不再给出，详细增补程序如下：

```

161 IF P=211 THEN 400
162 IF P=193 THEN 500
163 IF P=197 THEN 600
164 IF P=212 THEN 700
.....
400 TEXT;HOME;VTAB 12;HTAB 3;PRINT"GIVE A
    NAME FOR THE PICTURE TO SAVE;"
405 INPUT M$;FLASH
410 HOME;VTAB 15;HTAB 20;PRINT"SAVING..."
415 NORMAL
420 PRINT CHR$(4)"BSAVE";M$;","A$4000,L
    $2000"
425 VTAB 20;HTAB 20;PRINT"GOOD!"
430 FOR I=1 TO 1500:NEXT
435 POKE -16304,0;POKE -16299,0;POKE -16297,0
440 GOTO 60
500 TEXT;HOME;VTAB 22;HTAB 10;PRINT"GIVE
    A NAME FOR LOADING;"
505 INPUT N$
510 POKE -16304,0;POKE -16299,0;POKE -16297,0
515 PRINT CHR$(4)"BLOAD";N$
520 GOTO 60
600 TEXT;HOME;VTAB 22;HTAB 15;PRINT
    "GOOD-BYE!!";
    END
700 TEXT;HOME;PRINT
705 PRINT"U=LEFT UP; I=UP; O=RIGHT UP"
710 PRINT;PRINT"J=LEFT; K=RIGHT;"
715 PRINT;PRINT"N=LEFT DOWN; M=DOWN; ,
    =RIGHT DOWN;"
720 PRINT;PRINT"Q=DRAW; W=XDRAW; ESC=
    CLEAR;"
725 PRINT;PRINT"O-7=CHANGE COLOR;"
730 PRINT;PRINT"A=LOAD; S=SAVE; E=QUIT;"
735 PRINT;PRINT"《HIT 'SPACE-BAR' TO
    RETURN》"
740 PRINT
750 GET E$;IF E$=" " THEN 435

```

原程序增加以上语句后，将多占用3个扇区的磁盘空间，但与增加的功能比较，还是划算的。

第六讲 简单程序和分支程序设计

南京大学大气科学(210008) 朱国江

从本讲开始介绍机器语言程序设计的方法和技巧,包括简单程序设计、分支程序设计、循环程序设计、子程序设计、堆栈程序设计及调用监控子程序等六个方面。

虽然我们分节介绍这六个方面内容,但实际上,一个实用程序常常是六个方面的有机结合。

简单程序是复杂程序的基础,也是初学者程序设计入门的必经之路。分支程序是循环程序的基础,用以解决各种各样的分支转移问题。循环程序完成大量的在处理形式上完全相同的重复性计算工作,是机器语言程序设计的核心和精华。子程序用以处理在功能上具有一定独立性,能够完成相同计算和操作,并被经常调用的程序段,用以简化设计并使结构清晰。堆栈是计算机信息处理中的重要概念,是保护和恢复现场的主要手段。监控子程序简短独立、功能完整、结构合理、简炼灵活,在程序设计中,合理调用能起到事半功倍的效果。

§ 1. 简单程序设计

简单程序一般是指整个程序自始至终按指令顺序逐条执行,而在执行过程中不会出现分支或转移,因而又称直线程序。

简单程序设计方法虽然比较单一,但包含的内容还是比较丰富的,即包括数据输入、进行计算、输出结果、结束停机等四个方面内容。也就是说它包含了任何一个程序设计中不可缺少的每一部份,反映了程序设计的结构和全貌。

简单程序仅能完成一些简单操作,一般不单独使用。而且,任何一个实际程序绝非单纯由简单程序组成,但初学编程从简单程序入手实为必要,因为它是构成复杂程序的基础,又是初学者入门的必经之道。

现在,举一些简单程序的设计实例。

〔例1〕数据块转移

将起始地址\$2000—\$2004的5个字节内容转移到\$4000—\$4004单元中去。

由于转移的数据较少,问题十分简单,直接写出程序6.1。

程序6.1:

```
1000- AD 00 20    LDA    $2000
1003- 8D 00 40    STA    $4000
1006- AD 01 20    LDA    $2001
1009- 8D 01 40    STA    $4001
100C- AD 02 20    LDA    $2002
100F- 8D 02 40    STA    $4002
```

```
1012- AD 03 20    LDA    $2003
1015- 8D 03 40    STA    $4003
1018- AD 04 20    LDA    $2004
101B- 8D 04 40    STA    $4004
101E- 60          RTS
运行前: 2000-0A 0B 0C 0D 0E
         4000-00 00 00 00 00
运行后: * 1000G✓
         * 4000-4004✓
         4000-0A 0B 0C 0D 0E
```

程序6.1用的是取数和送数两条指令,反复进行同样的操作共5次。显然,若要传送250个字节的数据,则程序将要写上500个语句,这不仅没有必要,而且几乎也是不可能的,因此,对于大量数据的传送,直接程序有很大的局限性。对于少量数据的传送,程序6.1既简单又实用,复杂程序中的数据传送常采用类似的结构。

〔例2〕数据交换

将\$06单元和\$08单元中的内容互换。设原来(06)=AA,(08)=BB。交换后应有(06)=BB,(08)=AA。

对这类简单问题设计方法很多,这里借助于一个暂存单元\$07,做中间寄存器来互换。见程序6.2。

程序6.2

```
1000- A5 06    LDA    $06
1002- 85 07    STA    $07
1004- A5 08    LDA    $08
1006- 85 06    STA    $06
1008- A5 07    LDA    $07
100A- 85 08    STA    $08
100C- 60      RTS
运行前: * 6:AA✓
         * 8:BB✓
运行后: * 1000G✓
         * 6✓
         0006-BB
         * 8✓
         0008-AA
```

分析程序6.2时,应紧紧抓住这样一个事实:数据从源地址传送到目的地址,目的地址的内容被源地址内容更新,而源地址内容不变。这也是分析一切传送指令的要领。

〔例3〕求反码

设原码为\$6A,设计一个程序求出它的反码。结果为\$95。

6502指令系统中没有求反码的指令,但可以将一

个8位二进制数与 \$FF 进行异或处理,异或结果即为该8位二进制数的反。

即: $6A = 01101010$
 $+ FF = 11111111$
 $\hline 95 \quad 10010101$

故有程序6.3。

程序6.3:

```
1000- A9 6A    LDA  # $6A
1002- 49 FF    EOR  # $FF
1004- 20 DA FD JSR  $FDDA
1007- 60      RTS
```

运行: *1000G✓

95

〔例4〕求补码

设原码为 \$09,设计一个程序求出它的补码。结果为 \$F7。

6502指令系统中也没有求补码的指令,但可根据原码取反加1就是补码的定义,容易写出程序6.4。

程序6.4:

```
1000- A9 09    LDA  # $09
1002- 49 FF    EOR  # $FF
1004- D8      CLD
1005- 18      CLC
1006- 69 01    ADC  # $01
1008- 20 DA FD JSR  $FDDA
100B- 60      RTS
```

运行: *1000 G✓

* F7

〔例5〕拆字

拆字又叫字的分离或字的分解。

设原字为 A3(10100011),将其分成两段,每段四位,高4位放入 \$6001单元中低4位位置上,低4位放入 \$6000单元低4位位置上,并使 \$6001和 \$6000两个单元的高4位全部清零。

即: $(6000) = 00000011 = 03$

$(6001) = 00001010 = 0A$

解本题的思路可分为二步:第一步将原字 A3取出来,屏蔽高四位(AND # \$0F),留下低4位(0011),存入 \$6000单元中;第二步将原字 A3再一次调出来,用4次逻辑右移一位移令 LSR,这样原高四位的值 A(1010)全部移入低4位,而高4位变0,再存入 \$6001单元。故有程序6.5。

程序6.5:

```
1000- A9 A3    LDA  # $A3
1002- 29 0F    AND  # $0F
1004- 8D 00 60 STA  $6000
1007- A9 A3    LDA  # $A3
1009- 4A      LSR
100A- 4A      LSR
100B- 4A      LSR
100C- 4A      LSR
100D- 8D 01 60 STA  $6001
1010- 60      RTS
```

运行: *1000G✓

* 6000✓
 * 6000-03
 * 6001✓
 * 6001-0A

〔例6〕组字

组字又称语句组合,字组合。例如,将 \$01和 \$02单元中的低4位组成一个字,放在 \$03单元中。其中 \$01单元的低4位放入 \$03单元高4位; \$02单元的低4位放入 \$03单元低4位。

即: $(01) = 6A$

$(02) = 43$

$(03) = A3$

设计本题程序的要领是先拆字后组字,见程序6.6。

程序6.6:

```
0300-A9 6A    LDA  # $6A } (01)=6A
0302-85 01    STA  $01 }
0304-A9 43    LDA  # $43 } (02)=43
0306-85 02    STA  $02 }
0308-A5 02    LDA  $02取43
030A-29 0F    AND  # $0F屏蔽高四位
030C-85 03    STA  $03存数(03)=03
030E-A5 01    LDA  $01取6A
0310-0A      ASL
0311-0A      ASL } 左移四位
0312-0A      ASL }
0313-0A      ASL }
0314-85 04    STA  $04 存数(04)=A0
0316-18      CLC      清进位,0→C
0317-A5 03    LDA  $03 取03
0319-65 04    ADC  $04 加 A0
031B-85 03    STA  $03 存结果 A3
031D-60      RTS      结束
```

非常有趣的是,程序6.6中的 ADC \$04指令,可以换成下面三条指令中的两条:

```
AND  $04
ORA  $04
EOR  $04
```

您能指出是哪两条指令呢?为什么?

当您换上逻辑运算指令后,CLC 指令是否还需要?为什么?

值得指出的是,程序6.6还可以进一步化简,完全可以不用 \$04单元暂存,因为在执行第七条指令 STA \$03后, \$03单元已存好03的值;而在执行四条左移一位指令 ASL 后, \$01单元的值为 A0。因此,可以将 \$01单元的内容和 \$03单元中的内容进行组合。方法是修改 \$0314单元开始的内容。

```
如:0314-A5 03 LDA  $03
    0316-05 01 ORA  $01
    0318- 85 03 STA  $03
    031A-60      RTS
或 0314-A5 03 LDA  $03
    0316-45 01 EOR  $01
```

0318-85 03 STA \$03

031A-60 RTS

程序6.6还可以进一步简化,可以省去\$01-\$04的4个零页地址的选用,同样完成字的组合任务,见程序6.7。

程序6.7:

```
1000- A9 43   LDA  #$43
1002- 29 0F   AND  #$0F
1004- 85 06   STA  $06
1006- A9 6A   LDA  #$6A
1008- 0A      ASL
1009- 0A      ASL
100A- 0A      ASL
100B- 0A      ASL
100C- 18      CLC
100D- 65 06   ADC  $06
100F-20 DA FD JSR  $FDDA
1012- 60      RTS
```

运行: *1000G

A3

总之,一道题目的解法是多种多样的,学习用不同方法编程,可以训练思维,扩大思路,学会方法,熟习指令,提高编制程序的能力和技巧。

〔例7〕求平均值

为简单起见,仅求两个数\$3A和\$22的平均值。结果应为\$2E。见程序6.8。

程序6.8:

```
1000- 20 58 FC JSR  $FC58
1003- A9 3A     LDA  #$3A
1005- 18        CLC
1006- 69 22     ADC  #$22
1008- 4A        LSR
1009- 20 DA FD JSR  $FDDA
100C- 60        RTS
```

程序6.8在使用上有很多不足之处:

- 不能进行N个数的累加和求平均
- 和数只能限于一个8位数,不能有进位
- 只适用于无符号数或正数,LSR相当于÷2

因此,本例和本节中的其它几个例子,处理的都是一些最简单问题。而对一些比较复杂的问题,如:N个数据的累加和,大量数据的传送,多个数据的交换,判断奇偶数,确定正负数,输入输出码的转换,字符串的比较,多个数据块的连续搬移等等诸如此类的问题,简单程序设计是不能胜任的。故有必要介绍其它一些程序设计方法,以补不足。但不管怎样,简单程序设计方法,仍是一切初学者程序设计入门的必然阶梯,因为它处理的问题简单、直接,用的指令少,没有分支、转向和循环,直观、形象,易于初学者接受。通过上述实例分析,我们初步勾画了程序设计的部份面貌,也为程序设计方法和技巧抛出了引子,希望起到开阔视野、启迪思维的作用。

§2. 分支程序设计

在程序设计中,除极简单的问题外,一般都比较复

杂,解决问题的程序常常带有分支,即根据不同条件,使程序转向不同的地方执行。因此,我们十分需要一种具有判断能力的控制转移手段,以便对程序进行有效的控制。而计算机则按给定的条件进行分析、比较、判断后决定程序的走向,这就是分支程序设计。

6502指令系统中为我们提供了众多的转移指令,这种指令使程序在执行过程中出现分支,其重要性在于它们使计算机具有某种判断功能,可以根据具体条件满足与否,来决定是进行分支转移还是继续执行下一条指令。

我们可以根据问题的性质,选择不同的转移指令,控制程序的流向,以解决各种各样的分支问题。必须着重指出,灵活而准确地使用转移指令,特别是条件转移指令,是高质量汇编语言程序设计不可缺少的部份,而且使用的技巧性很强。

现举若干实例,作为分支程序设计入门的引导。

〔例1〕数的比较

一数放在\$06单元,一数放在\$07单元,比较两者是否相同。相同给出00标志,不同给出FF标志。

对于数或字符串的比较是否相同的问题,最好用异或指令EOR,它的功能是将存储器同累加器内容异或,并将结果送累加器,见程序6.9。

程序6.9:

```
1000- A5 06     LDA  $06
1002- 45 07     EOR  $07
1004- D0 06     BNE  $100C
1006- 85 0B     STA  $0B
1008- 20 DA FD JSR  $FDDA
100B- 60        RTS
100C- A9 FF     LDA  #$FF
100E- 85 09     STA  $09
1010- 20 DA FD JSR  $FDDA
1013- 60        RTS
```

运行前: *6:AA

*7:FF

运行后: *1000G

FF

运行前: *6:AA

*7:AA

运行后: *1000G

00

程序6.9中,在执行异或指令EOR后,有两种情况:两数相同,结果为0;两数不同,结果非0。因此,标志寄存器P中的零标志Z保留了执行EOR指令后结果的情况,前者Z=1,后者Z=0,所以,选用BNE指令也是顺理成章的,它是根据Z标志是0还是1,发生转移或者继续执行下一条指令。两数相同,结果为0,Z=1,执行下一条指令,0→(08),显示(A)=0的结果并停机;两数相异,结果非0,Z=0,转移到\$100C地址,取不同标志FF→(A),FF→(09),显示(A)中内容FF并结束。

〔例2〕判断奇偶数

在\$06单元中存放一个数B,若B是奇数,给出01标志;若B是偶数,给出00标志。

任何一个数(正数,负数或零),将其展开成一个8位二进制数时,则第0位不是“1”就是“0”。若原数是偶数或零时,则第0位一定是“0”;若原数是奇数时,则第0位一定是“1”。这个事实告诉我们,为了判断数的奇偶性,可以改为判断第0位是“1”还是“0”。

如果我们将一个数先右移一位,再左移一位,则第0位一定是“0”。

这样,我们就可以根据运算前后的值相同与否,判定数的奇偶性,结果相同为偶数,结果相异为奇数。故有程序6.10。

程序6.10:

```
1000- A5 06    LDA  $06
1002- 4A        LSR
1003- 0A        ASL
1004- C5 06    CMP  $06
1006- D0 05    BNE  $100D
1008- A9 00    LDA  # $00
100A- 4C 0F 10 JMP  $100F
100D- A9 01    LDA  # $01
100F- 20 DA FD JSR  $FDDA
1012- 60        RTS
```

运行前: * 6: C7 输入一个奇数

运行后: * 1000G

01 给出奇数标志

运行前: * 6: 34 输入一个偶数

运行后: * 1000G

00 给出偶数标志

〔例3〕找两数中较大者

两数分别存入\$6000和\$6001单元中,将大数找出并放入\$6002单元中。

取第一个数,与第二个数比较,若第一个数大,则存入\$6002单元,反之,第二个数大,将第二个数调入累加器A,再用STA \$6002指令,存入\$6002单元。见程序6.11。

程序6.11:

```
1000- AD 00 60    LDA  $6000
1003- CD 01 60    CMP  $6001
1006- B0 03       BCS  $100B
1008- AD 01 60    LDA  $6001
100B- 8D 02 60    STA  $6002
100E- 60         RTS
```

运行前: * 6000: 4F 3F

运行后: * 1000G

* 6002

* 6002-4F

运行前: * 6000: 6D 80

运行后: * 1000G

* 6002

* 6002-80

这段程序的思想是用CMP指令来比较两个数,然后利用BCS指令进行程序分支。在比较指令之后配合条件转移指令,是分支程序设计中经常使用的一种方

法。

应该指出的是,在选用CMP指令时,要注意几点:

•CMP指令是累加器A的内容同存储器M的内容比较,比较实际上是做一次减法操作,即 $A-M$

•CMP指令同减法指令SBC区别之一是,借位标志 \bar{C} (表示C标志取反)不参加减法运算,前者 $A-M$,后者 $A-M-\bar{C}$ 。

•CMP指令同减法指令SBC的另一个区别是,减法运算结果不送入累加器A,即CMP执行后不改变A的内容,前者 $A-M$,后者 $A-M-\bar{C} \rightarrow A$ 。

•执行CMP指令后,影响P寄存器中的标志位C、Z、N,尤其是影响标志位C。若 $A \geq M$,表示够减无借位,C置1;若 $A < M$,表示不够减有借,C置0。因此,可由C的状态判断A同M中两数谁大谁小。

因此,本例在选用CMP指令后,自然配合使用BCS指令,因为BCS指令是根据P寄存器中C的状态决定分支转移或继续执行下一条指令,即 $C=1$ 时转移, $C=0$ 时继续。

〔例4〕确定正负数

一个数(正数或负数)存放在\$06单元中,设计一个程序能判断该数是正数还是负数,并给出正数标志00、负数标志01。

由于本例中只判断正数或负数性质,而且仅限于对一个数的判别,因此,选用BPL指令是合适的,因为它以N作为判别标准, $N=0$,表示正数,执行分支动作; $N=1$,表示负数,则执行下一条指令。N标志的状态可由执行LDA指令后,P标志寄存器中的N标志位取得,因为LDA指令对符号位N及零标志位Z皆有影响。故有程序6.12。

程序6.12:

```
1000- A5 06    LDA  $06
1002- 10 05    BPL  $1009
1004- A9 01    LDA  # $01
1006- 4C 0B 10 JMP  $100B
1009- A9 00    LDA  # $00
100B- 85 07    STA  $07
100D- 60        RTS
```

运行前: * 6: 7D 输入一个正数

运行后: * 1000G

* 7

* 0007-00 给出正数标志

运行前: * 6: 96 输入一个负数

运行后: * 1000G

* 7

* 0007-01 给出负数标志

程序6.12中安排了一条无条件转移指令JMP,它是编程人员事先安排好的,它控制程序没有任何条件地实现转移,即将程序由通常的顺序执行状态转向另一个指定的目标地址,这和条件转移指令的执行情况不同,后者则是在满足某种条件的情况下,才使程序转移,即根据P寄存器中各标志位状况来确定程序是否转移。

〔例5〕输入码转换

如从键盘上按下一位10进制数,将其转换成16进制数。

10进制数的代码是0—9,程序的设计必须保证小于0的和大于9的键码都不要,而且还应保证按任何字符和运算符都不应出错,能自动返回等待正确输入;同时,也应安排一个控制键,例如回车键,让其程序自动停机。因此,本程序的设计是包含多个条件语句的分支程序设计。见程序6.13。

程序6.13:

```
1000- 20 1B FD   JSR   $FD1B
1003- C9 8D     CMP   # $8D
1005- F0 11     BEQ   $1018
1007- C9 B0     CMP   # $B0
1009- 90 F5     BCC   $1000
100B- C9 BA     CMP   # $BA
100D- B0 F1     BCS   $1000
100F- 20 DA FD   JSR   $FDDA
1012- 20 8E FD   JSR   $FD8E
1015- 4C 00 10   JMP   $1000
1018- 60        RTS
```

程序6.13运行后,等待操作者从键盘键入输入的数码,若在0至9之间,则显示 B0—B9(0—9对应的ASCII码),否则按任何字符、符号都无效,转向 \$1000 开始重新接受按键输入,若要停机,则按 RETURN(它的ASCII码是8D)键。程序中用了三个监控子程序, \$FD1B:等待键盘输入, \$FDDA:输出一个字符的代码子程序, \$FD8E:换行。

〔例6〕16进制数转换成ASCII码

计算机键盘与屏幕显示器都采用ASCII码表示字符。一个16进制数字可以用代码0—9或者用字母A—F表示。这两段内容在正常方式的屏幕显示时,对应的ASCII码是B0—B9及C1—C6。一个需要显示的数字必须先换成对应的ASCII码,然后再送入显示器。

要完成16进制数向ASCII码值转换,其方法是:如果属于0—9数字,则只要将它们加上B0;如果属于A—F字母,则需将它们加上B7。所以,16进制数字转换成ASCII码的程序,实际上是一个分支转移程序:即判断输入码属于0—9还是A—F,是前者加B0,是后者加B7。故有程序6.14。

程序6.14:

```
1000-A5 06      LDA   $06
1002-C9 0A      CMP   # $0A
1004-90 03      BCC   $1009
1006-18         CLC
1007-69 07      ADC   # $07
1009-69 B0      ADC   # $B0
100B-85 07      STA   $07
100D-20 DA FD   JSR   $FDDA
1010-60        RTS
```

运行前: *6:F✓

运行后: *1000G✓

*C6

或: *1000G✓

*7✓

*0007-C6

运行前: *6:05✓

运行后: *1000G✓

*B5

或: *1000G✓

*7✓

*0007-B5

本题也可用另法求解,其思路是先把数字0的ASCII码B0加到所有待转换的16进制数上去,然后与BA比较,以判断是否属于0—9,再决定是否加07。故有程序6.15。

程序6.15:

```
1000- A5 06      LDA   $06
1002- 29 0F      AND   # $0F
1004- 09 B0      ORA   # $B0
1006- C9 BA      CMP   # $BA
1008- 90 02      BCC   $100C
100A- 69 06      ADC   # $06
100C- 85 07      STA   $07
100E- 20 DA FD   JSR   $FDDA
1011- 60        RTS
```

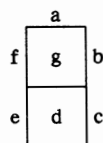
程序6.15的运行方法同程序6.14。

〔例7〕10进制数转换成7段代码

7段代码是七段发光二极管显示器显示字符所采用的代码,对共阴极7段显示器来说,所用7段代码显示10进制数字的对应关系为:

数字	代码	数字	代码
0	3F	5	6D
1	06	6	7D
2	5B	7	07
3	4F	8	7F
4	66	9	6F

7段代码的安排是:



	64	32	16	8	4	2	1
	g	f	e	d	c	b	a
	2^6	2^5	2^4	2^3	2^2	2^1	2^0

在一个字节中构成7段代码的方法是:最高位总是0,其余7位由低位向高位顺序为a、b、c、d、e、f、g各段代码(1亮,0不亮)。

例如,当a、b、c、d、g均为1时,则显示10进制数3,即:

$$\begin{aligned}
 &2^0 \times 1 + 2^1 \times 1 + 2^2 \times 1 + 2^3 \times 1 + 2^4 \times 0 + 2^5 \times 0 + 2^6 \times 1 \\
 &= 1 + 2 + 4 + 8 + 0 + 0 + 64 \\
 &= (79)_{10}
 \end{aligned}$$

=(4F)₁₆

所以,要求编一个程序,输入0—9这10个10进制数中的任一代码,显示出相应的16进制数。

程序可以这样安排,在\$6000中存放一个10进制数,比较是否在0—9之间,是的,取对应的16进制代码;不是的,显示0。故有程序6.16。

程序6.16:

```
1000- A9 00    LDA    # $00
1002- AE 00 60  LDX    $6000
1005- E0 0A    CPX    # $0A
1007- B0 03    BC9     $100C
1009- BD 00 70  LDA    $7000,X
100C- 8D 01 60  STA    $6001
100F- 20 DA FD  JSR    $FDDA
1012- 60      RTS
```

程序6.16运行前应先设置:

7000-3F 06 5B 4F 66 6D 7D 07

7008-7F 6F

* 6000;5✓

* 1000G✓

* 6D

本题可以进一步简化,只要在\$06单元中放置0—9当中的任一10进制数码,运行程序6.17后,立即得到结果。

程序6.17:

```
1000-A4 06    LDY    $06
1002-B9 00 70  LDA    $7000,Y
```

1005-20 DA FD JSR \$FDDA

1008-60 RTS

程序6.17非常简捷,运行前应设置\$06单元和\$7000—\$7009单元中的值。

如(06)=03

7000-3F 06 5B 4F 66 6D 7D 07 7F 6F

* 1000G✓

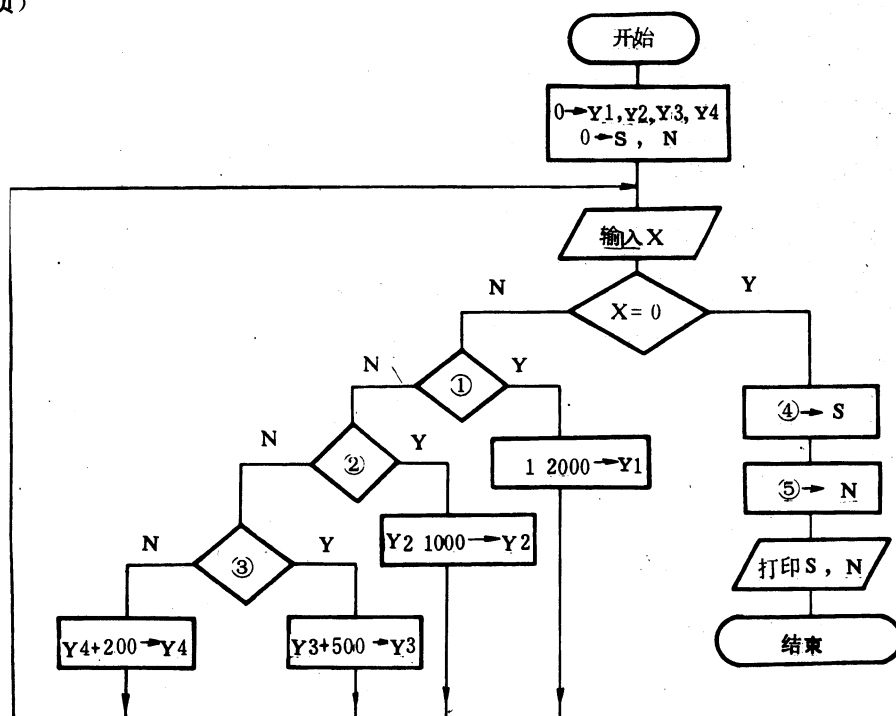
显示:4F

以上我们通过几个实例,介绍了分支程序设计的一些问题。可以看出,分支程序解决和处理的问题,要比简单程序广泛和多样,在设计方法和应用技巧上,又比简单程序丰富和灵活。

应该着重指出的是,学习和掌握分支程序设计有一定的难度,其困难所在是如何选用条件转移指令,以及什么条件转移和转向到那里去,这需要经常编程和大量调试。但有一点是至关重要的,必须牢牢掌握标志寄存器P的各个有关标志位是0还是1的状况。因为标志寄存器P中保留了每条指令执行后的情况,如执行后结果为全零或非0;是正数或负数;有进位或无进位;有溢出或无溢出等,这些情况即为判断条件分支的重要依据,也是学习本节内容的要领和程序设计的重点。

还有一点要说的,仅仅掌握了简单程序、分支程序的设计方法,还不能解决名目繁多形式各异的实际问题,特别是不能有效地解决大量的在处理形式上又完全相同的重复性计算问题。因此,循环程序的设计方法和技巧,应该作为重点来介绍,请看下讲内容。

(上接30页)



答案:①

②

③

④

⑤

初级技术员级软件水平考试辅导

PC BASICA 试题

北京市计算中心(100005) 李志刚

一、填空

① $(2\sin 45^\circ + \cos 30^\circ) / (3 |X| \ln(Z-5))$ 的 BASIC 表达式是_____

② 等效于 SWAP X,Y 的语句组是_____

③ 表达式 $\text{SQR}(\text{VAL}(\text{RIGHT} \$ ("1245", 2)) + 4)$ 的值是_____

④ 将内存中的 BASIC 程序以 PROG 为文件名,用 ASCII 码方式存入 A 盘的命令是_____

⑤ 在使用 FOR/NEXT 循环语句时,要求内循环的 NEXT 语句必须出现在外循环的 NEXT 语句之前,是由堆栈的_____性质决定的。

六、阅读下列说明,从选择答案中找出正确答案,把对应的字母填入括号内。

① 标准程序流程图是指国家标准()

供选择的答案:

A. GB2312—80 B. GB1526—89

C. ISO 5807—85

② 程序的三种基本控制结构是(),它们的共同特点是()。

供选择的答案:

A. 过程、子程序、函数 B. 顺序、条件、循环

C. 递归、队列、堆栈 D. 调用、返回、转移

E. 不能嵌套使用 F. 只能写简单程序

G. 只有一个入口和一个出口 I. 已经用硬件实现,只需调用

③ 结构化程序设计的一种基本方法是()。

供选择的答案:

A. 筛选法 B. 枚举法

C. 归纳法 D. 逐步求精法

④ 结构化程序设计语言中()GOTO 语句。

供选择的答案:

A. 必需有 B. 没有 C. 可有可无 D. 限制

⑤ 在子程序头,一般要写出一些变量,这些变量被称为()。它们应与调用该子程序的语句中的

()在个数、类型、顺序等方面相一致。

供选择的答案:

A. 条件参数 B. 形式参数 C. 入口参数

D. 实际参数 E. 出口参数 F. 局部参数

G. 全局参数

三、读下列程序,从选择答案中找出正确答案,把对应的字母填入括号内”

① 程序:

10 READ A,B,C

20 RESTORE

30 DATA 1,2,3,4

40 READ G,F,E,D,C,B

50 DATA 4,5,6,7

60 PRINT A;B;C;D;E;F;G

70 END

运行结果是:()

供选择的答案:

A. 1 2 3 4 5 6 7

B. 1 4 5 4 3 2 1

C. 1 5 4 4 3 2 1

D. 7 6 5 4 3 2 1

② 程序:

10 INPUT "n=";N

20 M=N+1

30 F=1

40 GOSUB 60

50 PRINT F;END

60 M=M-1

70 IF M<>1 THEN GOSUB 60

80 F=F*M

90 M=M+1

100 RETURN

运行结果是:()

RUN

5↓

A. 24

B. 48

C. 96

D. 120

③ 此程序运行后打印的结果是()

程序:(设真为-1,假为0)

供选择的答案:

10 FOR A=-1 TO 0

A. 0 0 -1 -1

20 FOR B=-1 TO 0

B. 0 1 -1 -1

30 PRINT (A AND B)+(NOT A) C. -1 0 -1 -1

40 NEXT B

D. 1 0 -1 -1

50 NEXT A

60 END

④ 此程序运行的结果是()

程序:

供选择的答案:

10 Y=14

A. 269

20 DEF FNA(X)=X²+2*X

130.5

30 Y=Y+FNA(FNA(3))

B. 269 130.5

40 PRINT "Y1=";Y

C. Y1=269 Y2=130.5

50 Y=(Y-FNA(2))/2

D. Y1=269

60 PRINT "Y2=";Y

Y2=130.5

70 END

⑤ 运行此程序,打印结果是()。

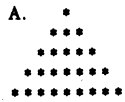
程序:

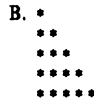
供选择的答案

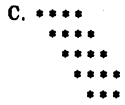
```

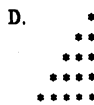
10 N=120 FOR X=1 TO N
20 FOR X=1 TO N
30 PRINT " * ";
40 NEXT X
50 PRINT
60 N=N+1
70 IF N>5 THEN 90
80 GOTO 20
90 END

```

A. 

B. 

C. 

D. 

四、读程序,从供选择答案中将程序执行屏幕显示的错误信息对应的字母填入括号内。

①():

```

10 FOR I=2 TO 3
20 DIM A(I)
30 A(I)=I
40 PRINT A(I),
50 NEXT I
60 END

```

②()

```

10 DIM A(5)
20 FOR I=1 TO 10
30 READ A(I)
40 PRINT I;A(I)
50 NEXT I
60 DATA 0,9,8,7,6,5,4,3,2,1
70 END

```

③()

```

10 FOR I=1 TO 5
20 PRINT TAB(1);I*I;PRINT SIN(I)
30 NEXT I
40 END

```

④()

```

10 READ A$
20 IF A$=EWS THEN PRINT P$;END
30 P$=P$+A$
40 GOTO 10
50 DATA THE,IS,A,PEN,YES

```

供选择的答案:A. Duplicate Definition

B. Syntax error

C. Subscript out of range

D. Type mismatch

五、下面是模拟儿童画竞赛的鼓励奖抽奖程序,请补充填空。

抽奖原则:随机产生10个四位整数,取其中最大数的十位数字与最小数的个位数字组合成中奖号码的两位尾数。

```

程序:10 RANDOMIZE
20 DIM A(10)
30 FOR I=1 TO 10
40 A(I)= .....
50 PRINT A(I),

```

```

60 NEXT I
70 IF A(1)<A(2) THEN SWAP A(1),A(2)
80 FOR I=3 TO 10
90 IF _____ THEN SWAP A(1),A(I);
GOTO 110
100 IF _____ THEN SWAP A(2),A(I)
110 NEXT I
120 PRINT"最大数";A(1),"最小数";A(2)
130 A$=_____
140 B$=_____
150 PRINT"中奖号码"A$+B$
160 END

```

六、阅读下列程序说明和 BASIC 程序,请补充填空。

[程序说明]

本程序是将一正整数序列 {K1,K2,...,Kn} 重排列成一个新的序列。新序列中,比 K1 小的数都在 K1 的前面(左面),比 K1 大的数都在 K1 后面(右面)。

例如,当 n=9 时,序列 {6,8,9,1,2,5,4,7,3} 经重新排列后成为: {3,4,5,2,1,6,8,9,7}

[程序]

```

10 INPUT "N=";N
20 IF N<2 THEN 10
30 _____
40 FOR I=1 TO N
50 INPUT A(I);IF _____ THEN 50
60 NEXT I
70 FOR I=1 TO N;PRINT A(I),;NEXT I;
PRINT
80 I=1
90 FOR I=2 TO N
100 IF _____ THEN 170
110 X=A(I)
120 FOR J=I TO 2 STEP -1
130 _____
140 NEXT J
150 A(1)=X
160 _____
170 NEXT I
180 PRINT
190 FOR I=1 TO N;PRINT A(I),NEXT I
200 END

```

七、“合理化建议”奖金发放以年经济效益为依据,具体发奖标准是:

等级	年经济效益(万元)	奖金(元)
一	100以上	2000
二	10~99	1000
三	1~9	500
四	不足1	200

请填写流程图中编号处的内容,使之能根据读入的年经济效益 X,按发奖标准兑奖,并统计发奖总数 S(单位万元)和合理化建议总数 N。(下转28页)



带 A/D 的单片机最小应用系统

合肥中国科技大学计算中心(230026) 张培仁 刘振安

A/D 变换广泛应用于工业控制、自动化、数据处理、智能仪器中。和单片机接口的 A/D 芯片主要有三类：双积分式 A/D 变换器，逐次比较式 A/D 变换器和 V/F 式 A/D 变换器。下面分别介绍它们和单片机的接口和硬软件结构。

智能热电偶温度测试仪中是以 8031 最小系统为核心，使用高精度 $4\frac{1}{2}$ 位积分式 A/D 变换器作为采样。该仪器的系统框图如下：

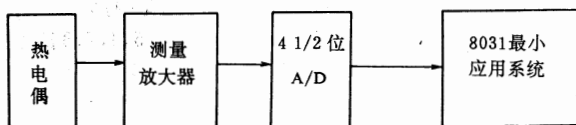


图1

在实验室和工业系统中，通过热电偶测温度经常遇到的问题是由于热电偶测出是毫伏数。最后要查阅热电偶手册，查出相应的温度，而使用单片机可以控制 $4\frac{1}{2}$ 位 7135A/D 片不断进行采样，再经过单片机的数据处理后显示出相应温度值。

用测量放大器把热电偶信号从毫伏放大到几伏。测量放大器由四个运放组成的直流放大器。共模抑制比可达 90dB，基本上满足了三种热电偶在工业测量中的要求。放大器必须加以屏蔽，以尽量减少干扰和噪声。放大器模拟地与单片机数字地严格相互分开，最后再联在一点，使地线上压差达最小。

双积分 A/D (7135) 与单片机连接电气原理图如图 2 所示。

A/D 7135 的特性如下：

- (1) 总计数达 ± 20000 个计数，精度为 ± 1 个数。
- (2) 输入阻抗大于 $10^9 \Omega$ 。
- (3) 自校零，保证零电压输入时读数为 0。
- (4) 采用 BCD 码扫描输出。
- (5) 设有 6 个 I/O 辅助信号可方便地与单片机接口与控制。

(6) 输出与 TTL 电路兼容。

7135 由 $\pm 5V$ 供电， V^+ 接 $+5V$ ， V^- 接 $-5V$ ，DGND 为数字地，即接 $\pm 5V$ 电源地。 U_R 为基准电压的高电位输入端，AGND 为模拟地，它是模拟信号的地，又是基准电压的低电位端。INT 为积分器输出，接积分电容。BUF 为缓冲放大器的输出，接积分电阻。AE 为自动调零输入端，接自动调零电容。 CR^+ 为基准电容的高电位端， CR^- 为基准电容低电位端。INHI 为模拟信号输入的正端。INLO 为模拟信号输入的负端。R/H 为运行或保持操作控制端。D1, D2, D3, D4, D5 为位扫描输出端，其中 D1 为个位，D5 为万位。B1, B2, B3, B4, B8 为 BCD 码数据，B1 为低位，B8 为高位。POL 为信号极性输出，信号为正时，POL 为 1，OR 为超量程状态输出，UR 为欠量程输出。BUSY 为忙信号输出，它指示 A/D 是正在进行转换还是已经转换完毕。STB 为数字选通

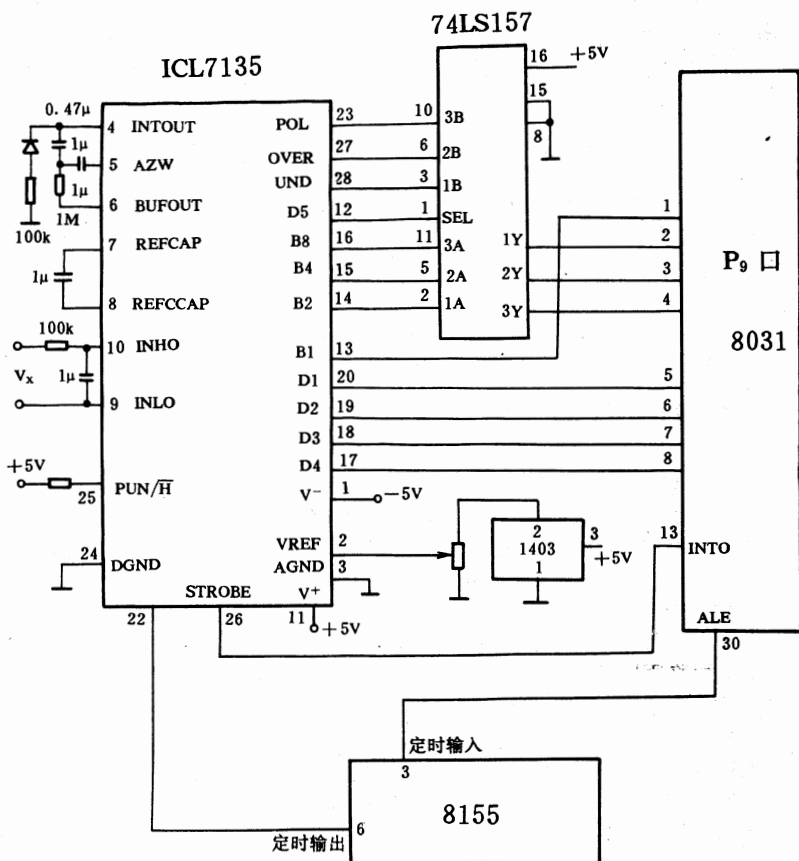
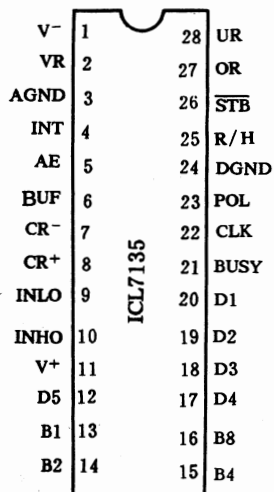


图2



ICL7135 管脚图

图3

程信号也一同被读入。在每一位数据产生后,都会产生中断请求,在中断服务中设置一个中断次数计数器。单片机根据第几次中断来判断是第几位产生的中断。准确将数据每一位存入所对应的缓冲区内。

7135输出定时波形如图4:

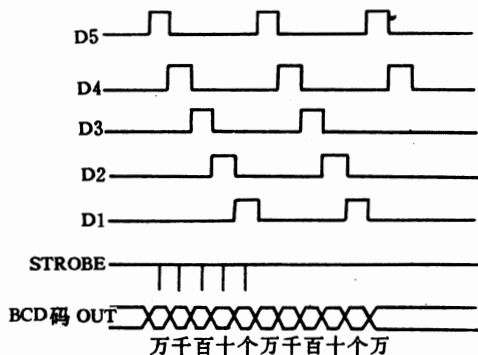


图4

单片机将数据读入内存以后,经过处理,将温度信号由8155的PA与PC口送显示。A口是段码,C口是位码,即选中六个数码管的那一个数码管。

数据输入在INT00中断服务子程序中完成。该子程序中用计数器计中断次数,决定是哪一位的中断,从而存入相应内存中,第一次中断将万位,极性,欠过量程一起读入8031中的RAM中,第二次中断将千位,第三次将百位,第四次将十位,第五次将个位读入,这样不断循环。

8031中RAM的数据内容和格式说明(与7135A/D变换有关的)如下:

40H:A/D输入极性。

41H:过量程标志。

42H:欠量程标志。

43H:A/D BCD 码十位,个位。

44H:A/D BCD 码千位,百位。

45H:A/D BCD 码万位(低三位)

程序框图(中断服务程序即 A/D 控制部分)

程序清单:A/D 部分

```

;
int00:      mov ie, #00h
            push a
            mov a, r0
            push a
            mov a, 47h
            inc a,
            mov 47h, a
            cjne a, #01, i1
            ajmp ino
i1:          cjne a, #02, i2
            ajmp in8
i2:          cjne a, #3, i3
            ajmp in9
i3:          cjne a, #4, i4
            ajmp ina
i4:          cjne a, #5, i5
            ajmp inb
i5:          ajmp ine
ino:         mov a, p1
            jnb acc. 3, in1
            mov 40h, #0
            ajmp in2
in1:         mov 40h, #0ffh
in2:         jb acc. 2, in3
            mov 41h, #0
            ajmp in4
in3:         mov 41h, #0ffh
in4:         jb acc. 1, in5
            mov 42h, #0
            ajmp in6
in5:         mov 42h, #0ffh
in6:         jb acc. 0, in7
            mov 45h, #0
            ajmp ine
in7:         mov 45h, #1
            ajmp ine
in8:         mov a, p1
            anl a, #0fh
            swap a
            mov 44h, a
            ajmp ine
in9:         mov a, p1
            anl a, #0fh
            mov r0, #44h

```



```

xchd a, @r0
ajmp ine
ina:
mov a, p1
anl a, # 0fh
swap a
mov 43h, a
ajmp ine
inb:
mov a, p1
anl a, # 0fh
mov r0, # 43h

```

```

xchd a, @r0
clr a
mov 47h, a
pop a
mov r0, a
pop a
mov p1, # 0ffh
mov ie, # 84h
reti

```

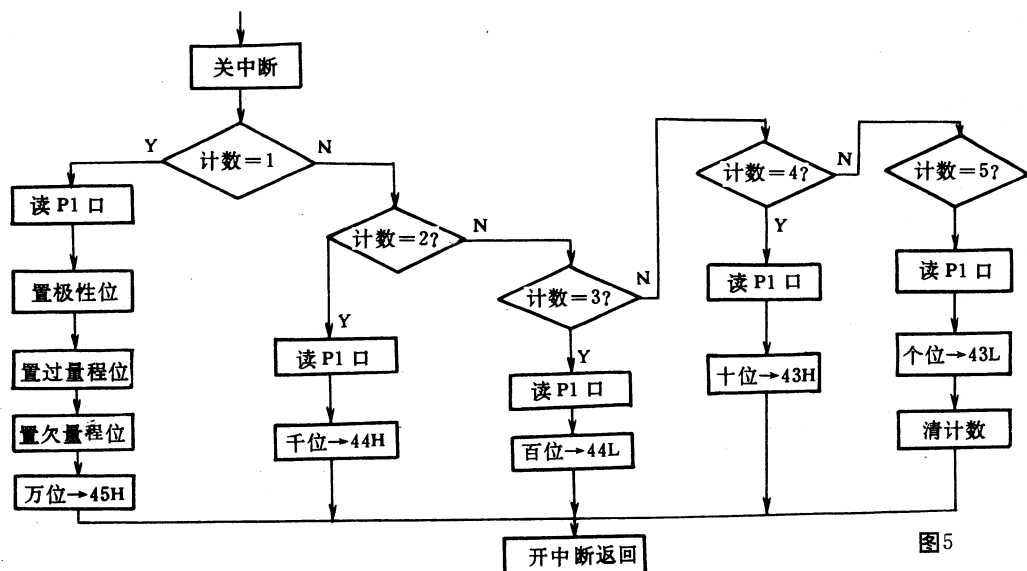


图5

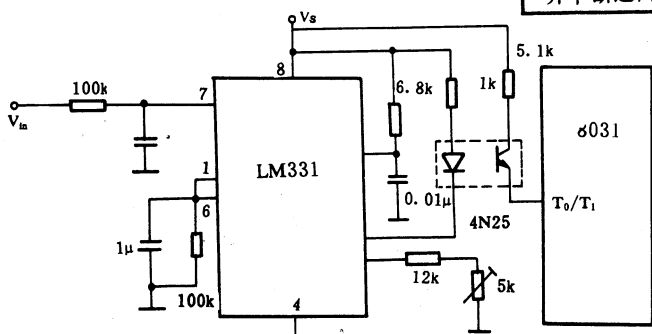


图6

图7是LM331原理图,它的原理简介如下:

输入比较器将输入电压 V_{IN} 与 V_X 比较,当 $V_{IN} > V_X$ 时,启动单脉冲定时器并导通频率输出晶体管和开关电流源,定时器的定时周期 $t = 1.1R_L C_L$,在这个周期中电流 i 向电容 C_L 充电,使 V_X 上升,当 V_X 上升至 $V_X > V_{IN}$,电流 i 关断,定时器自动复位,同时, C_L 逐渐通过 R_L 放电直到 $V_X < V_{IN}$ 为止,然后比较器再次启动定时器,开始下一个循环。

V/F 的变换程序是很简单的。单片机初始化时把 T0 清 0,以后启动 T0,定时器计输入脉冲。输入脉冲计数多少是和输入电压成线性关系。

使用 V/F 转换器作 A/D 时具有独特的优点。当单片机采用 V/F 型 A/D 时有如下一些优点:

- (1) 接口简单,占 8031 资源少。
- (2) 频率信号输入灵活。一般从 T0 或 T1 进入单片机。
- (3) 抗干扰性能好,容易进行光电隔离。
- (4) 便于远距离传输。

主要用于非快速过程 A/D 变换。
用 LM331 和 8031 做接口图如下:

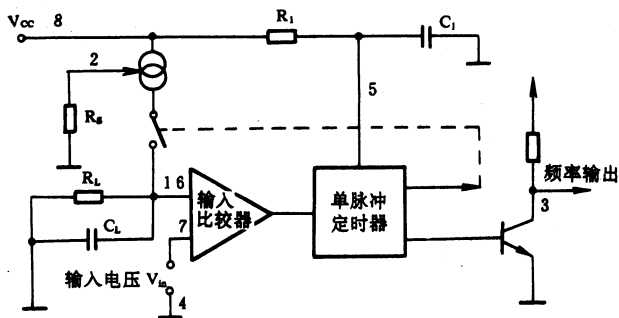


图7



学装微电脑

7段数码管读数装置

易齐干

本文介绍使用光导纤维检测7段数码管每段发出的光的信息;以及光—电信息转换和微电脑组成的读数装置。

同一距离检测有无光信号的状况

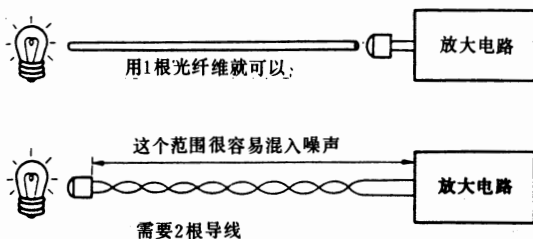


图1

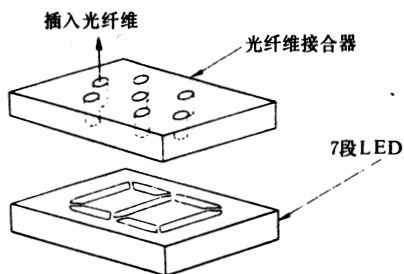


图2

表1

标号	助记符	地址	机器语言	注释
START	LD A,90H	00	3E 90	设置CW
	OUT (03H),A	02	D3 03	
	LD SP,0100H	04	31 00 01	设置SP
J1	LD HL,00D0H	07	21 D0 00	设置最前面数据的地址
J2	IN A,(00H)	0A	DB 00	输入数据
	LD B,(HL)	0C	46	
	CP B	0D	B8	与数字数据进行比较
	JP Z,J3	0E	CA 1E 00	
	INC HL	11	23	
	LD A,(HL)	12	7E	
	AND A	13	A7	字数据结束?
	JP NZ,J2	14	C2 DA 00	
	LD A,9EH	17	3E 9E	
	OUT (02H),A	19	D3 02	输出E
	JP J1	1B	C3 07 00	
J3	LD A,L	1E	7D	
	ADD A,10H	1F	C6 10	数字数据地址+16
	LD L,A	21	6F	
	LD A,(HL)	22	7E	
	CPL	23	2F	
	OUT (02H),A	24	D3 02	
	JP J1	26	C3 07 00	

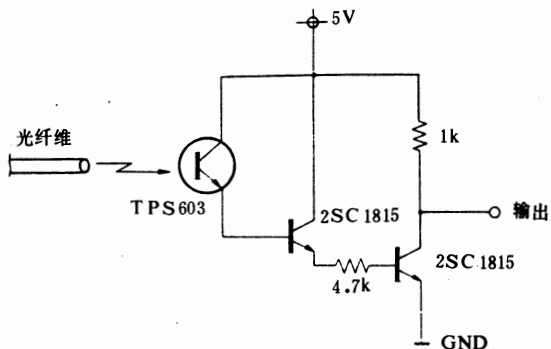


图3

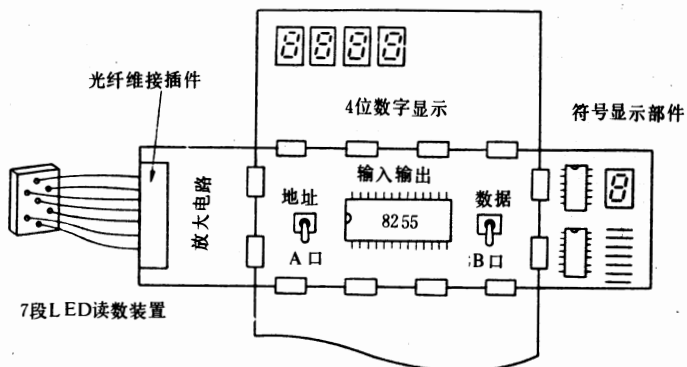


图4

标号	助记符	地址	机器语言	注释
DATA M	DB 03H	D0	03	0 数字数据
	DB 9FH	D1	9F	1
	DB 25H	D2	25	2
	DB 0DH	D3	0D	3
	DB 99H	D4	99	4
	DB 49H	D5	49	5
	DB 41H	D6	41	6
	DB 1BH	D7	1B	7
	DB 01H	D8	01	8
	DB 09H	D9	09	9
	DB 00H	DA	00	结束
DATA F	DB 03H	E0	03	0 显示数据
	DB 9FH	E1	9F	1
	DB 25H	E2	25	2
	DB 0DH	E3	0D	3
	DB 99H	E4	99	4
	DB 49H	E5	49	5
	DB 41H	E6	41	6
	DB 1BH	E7	1B	7
	DB 01H	E8	01	8
	DB 09H	E9	09	9

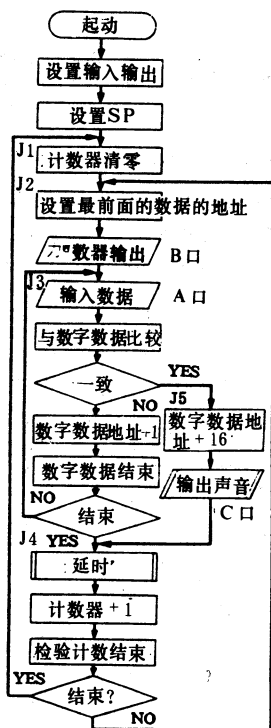


图5

光的信息转换为电的信息过程中,光导纤维的传输工具。与使用光敏晶体管直接转换,电缆根数可以减少一半。

另外,光电转换元件与放大电路距离愈近,噪声干扰愈少,效果愈好。

这就是使用光导纤维的优点。如图1所示。

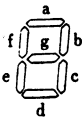
光导纤维的接合器由5mm厚丙烯酸塑料加工而成。要求光导纤维的中心尽量接近每段LED的中心部位。如图2所示。

光电信息转换部份电路如图3所示。光敏晶体管(东芝TPS603)将光信息转换为电信息。

光导纤维传输来的光照射到光敏晶体管的基极,

表2

7段LED 的显示	A口								16进制 表示
	a	b	c	d	e	f	g	—	
0	A7	A6	A5	A4	A3	A2	A1	A0	03H
1	0	0	0	0	0	0	1	1	9FH
2	1	0	0	1	1	1	1	1	25H
3	0	0	1	0	0	1	0	1	0DH
4	0	0	0	0	1	1	0	1	99H
5	1	0	0	1	1	0	0	1	49H
6	0	1	0	0	0	0	0	1	41H
7	0	0	0	1	1	0	1	1	1BH
8	0	0	0	0	0	0	0	1	01H
9	0	0	0	0	1	0	0	1	09H



发射极与集电极则导通。再经过两个晶体管(2SC1815)逐级放大,输出端或是L电平、或是H电平。

要读出7段管和小数点时,必须有8个如图3所示电路。

7段数码管读数装置与μP-80教育套件端口A相连,符号显示部件与端口C相连。如图4。程序要求输入7段数码管读数装置的LED灯亮信息,由符号显示部件显示。如果读到0~9数字以外的字符时,显示“E”。

编程的基本思想是将0~9数据预先写入存储器,与端口A输入数据进行比较,两者一致,端口C输出该数据;如果不一致,输出“E”。

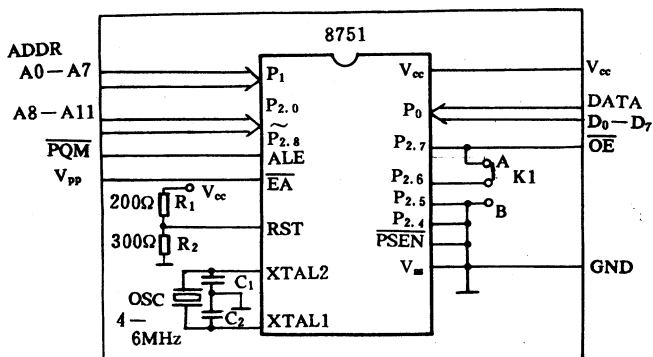
流程图如图5所示。程序清单如表1所示。7段数码管显示编码表如表2。

类似这种程序,随着读数装置不同的输入,仅改变微电脑输出数据,能够完成各种控制工作,例如,灯亮的启动;继电器驱动等等。有兴趣者不妨一试。

(上接38页)

校 验	V _{IHL}	0	1	1	0	0	×	×
加密编程	V _{IHL}	0	0*	V _{PP}	1	1	×	×

注:“1”是逻辑高电平
“0”是逻辑低电平
“×”是任意状态
“V_{PP}”是+21V±0.5V
“0*”是50ms 低电平脉冲
“V_{IHL}”是RST输入高电平,2.5V≤V_{IHL}≤V_{CC}+0.5V



向用户推荐——PIC16C5X 系列和 17C42 单片机

中国电子器材深圳公司供稿

美国 Microchip(微晶片)科技公司,是以生产 RISC 微控制器(单片机)和 EPROM 存储器为主的半导体厂商。其产品在国际市场上有相当强的竞争能力,该公司近年来着力开发亚太市场,对我国单片机市场很感兴趣,该公司生产的 PIC16C5X 系列和 PIC17C42 单片机已开始向国内供货。本文将简要介绍其性能,供用户参考。

一、PIC16C5X 系列单片机(采用 CMOS 工艺,8 位,内含 EPROM)

1. 引脚及说明

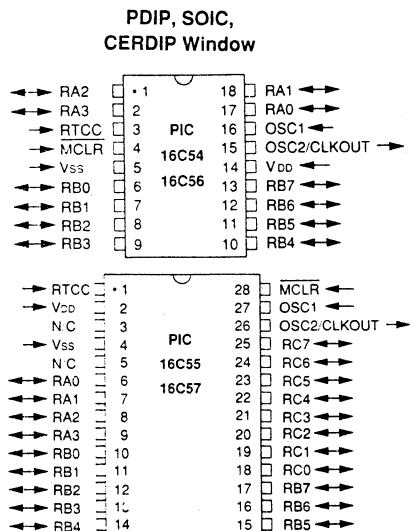


图 1

图中 PDIP 为塑料双列直插式封装,SOIC 为小引线封装,CERDIP,Window 为陶瓷双列直插式封装,开窗式。

现将芯片列表如下(电源电压:4.0~5.5V;工作频率:DC~20MHz;每种均有 PDIP,SOIC,CERDIP Window 三种封装方式)。

型 号	EPROM	RAM	I/O 口	引脚数
PIC16C54	512×12	32×8	13	18PIN
PIC16C55	512×12	32×8	21	28PIN
PIC16C56	1K×12	32×8	13	18PIN
PIC16C57	2K×12	80×8	21	28PIN

从图 1 中可以看出 PIC 系列单片机由于采用 RISC 技术,内部集成度很高,使得引脚十分简单,便于外电路的设计与使用。

(1)OSC1(输入),OSC2(输出) OSC1 可外接 RC,晶体振荡器或其他时钟电路。OSC2 在选用 RC 振荡器时,可用作 CLK 输出,其输出频率为 OSC1 输入频率的 1/4。

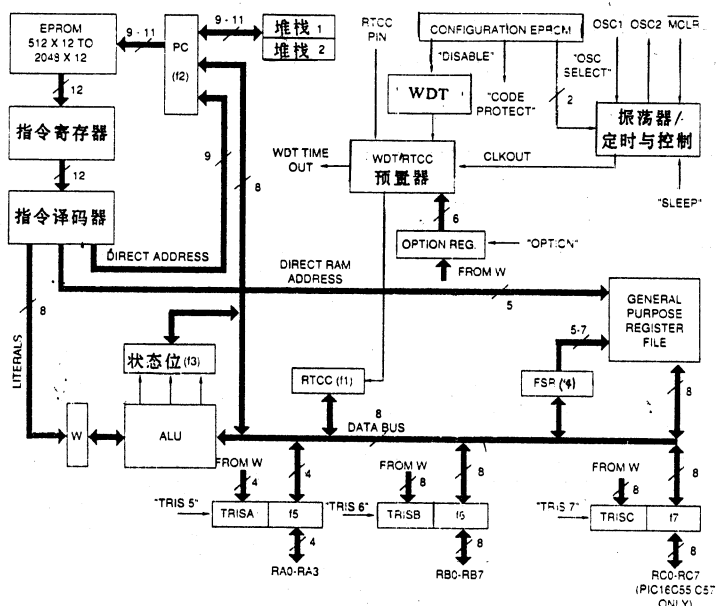
(2)RTCC(real-time clock counter,实时时钟计数器,输入) RTCC 是外加可编程控制时序。内部有 RTCC 寄存器,该寄存器可用程序设置和读取。用户可根据不同的外接时钟频率,利用程序设置 RTCC 和其他内部信号组成有关控制信号。

(3)MCLR 系统 RESET 信号。

(4)RA0~RA3,RB0~RB7,RC0~RC7(输入/输出) 用户编程的双向 I/O 口(RC0~RC7 只有 28PIN 的 PIC165X 才有)。

(5)VDD 和 VSS +5V 电源和地。

2. 内部结构及特点:



(1)从图2可以看出,PIC16C5X单片机内部除含有一般单片机的通用结构如ALU,PC,指令寄存器,指令译码器,状态寄存器等之外,还有一些特殊的结构。如通用寄存器阵列,实际上是一个可寻址的 32×8 (或 80×8)的RAM。该32个寄存器分别以F0~F31表示。均可用指令直接寻址,其中F0~F7是单片机本身专用的寄存器。如F2是起PC+1作用的,F5是设定RA0~RA3口性质的寄存器等等。F4~F31可用作RAM。

(2)二级堆栈。

(3)有监视计时器功能(Watchdog timer;WDT)它是芯片内部用来巡视工作状况的“看门狗”。其作用是当该计时器计数到终点时,会自动使单片机RESET,并重新启动。其作用是可防止在不正常的情况下“死机”或进入无限循环,提高芯片的抗干扰能力。也可以不使用此保护功能。

(4)有33条指令;指令字长12位;单周期,指令周期为DC~200ns。可进行位操作;有直接,间接和相对寻址方式。

3. EPROM的保密

PIC16C5X内含EPROM。EPROM的编程需用专用的PIC-PAK-I, PIC-ICE或PICMASTER等仿真开发系统。

EPROM中有四个专用位,其中一个为保密位,当程序写入后,将此位烧录为“0”,这样,任何人用任何器材都不能拷贝或读出EPROM中的程序。另一位是WDT的使能位,将此位烧录为“0”,则无WDT功能。其他二位是用来选择振荡方式的。

EPROM中还有16位留给用户专用,只能在PIC仿真开发系统中读写这些位,与程序的运行无关。

PIC16C5X系列中采用无窗口PDIP,SOIC封装的,其中的ROM用户可以用仿真器自己烧录,亦可加保密位,外人不能读出和擦除。因而它们适合定型的产品选用。

二、PIC17C4高性能8位CMOS EPROM单片机

PIC17C42是新推出的一种性能高于PIC16C5X系列的单片机。在增强的功能方面主要有:

(1)55条指令,多为单周期指令,指令字长16位,指令执行周期250ns。有直接、间接、相对和立即等寻址方式。

(2)寻址范围 $64K \times 16$ 片上2K,(可外接2K~64K)。

(3)片内EPROM $2K \times 16$ 。

(4)通用寄存器 232×8 ,其中有48个专用寄存器,其余可用RAM。

(5)16级堆栈。

(6)11级中断。

(7)33个I/O口。

(8)二个高速PWM输出(10 bit, 15.6KHz)。

(9)带有波特率发生器的UART口。

(10)有EPROM保密位。

(11)44PIN PLCC(塑料无引线芯片载体)封装。和P44PIN PQFP(塑料四边引线扁平封装)。图3为40PIN双列直插封装。

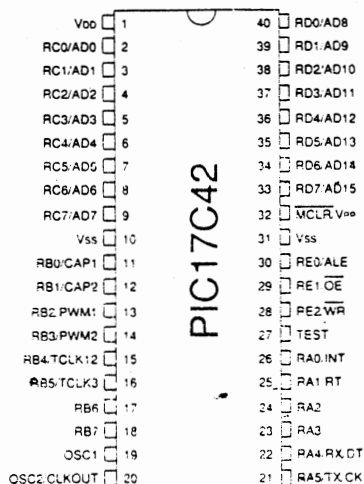


图3

三、专用仿真器

Microchip公司为开发PIC16C5X系列,PIC17C42单片机推出多种仿真器。

1. PICPAK-I——PIC16C5X普及型微控制器开发系统,该系统主要有:

(1)EPROM编程器(仅适用于PIC16CX系列),可以同PC机连用。

(2)PICALC宏汇编器,可实现源码到目的码的转换。

(3)PICSIM模拟器,可对PIC16C5X进行指令级模拟。支持PC机的MSDOS系统。

2. PIC-ICE——PIC16C5X在线仿真开发系统

(1)EPROM编程器,可独立也可同PC机或286,386及486连用。仅适用于PIC16C5X系列。

(2)宏汇编器。

(3)仿真器系统,用于PIC16C54,16C55和16C57的在线仿真。

3. PICMASTER-16——通用在线仿真器

(1)EPROM编程器,使用于PIC16C5X系列和PIC17C42。

(2)宏汇编器。

(3)仿真器,可用于对PIC16C5X和PIC17C42的在线仿真。适用在Microsoft Windows 3.0环境下的286,386和486机连用。

通过以上介绍,相信有兴趣的用户,对Microchip公司的PIC16C5X,PIC17C42单片机及其开发仿真系统会有一个概貌性的了解,相信会有越来越多的用户选用Microchip公司的单片机来开发自己的产品。



自带 EPROM 类单片机 简易加密编程方法的探索

河南周口地区水科所(466001) 庄 凯

计算机软件设计人员都为软件的加密工作搅尽了脑汁。但是,任何软件加密的方法,都不能绝对保护被加密软件不被非法拷贝或破译。在各种单片机应用中,最简单、可靠的加密方法是采用具有保密位的、自带 EPROM 程序存储器的单片机芯片。这类芯片,一旦将保密位编程(即加密编程),系统将禁止采用任何外部方式访问片内程序存储器。然而,对这类单片机芯片加密编程,往往要具有专用编程器或万用编程器,这对于只有普通开发设备的地方开发此类芯片是有困难的;另一方面,各类单片机技术手册、开发机手册以及许多单片机应用资料,只说明编程方式的要求,并不详细说明编程与加密的关系、以及实现的方法步骤,这使得许多人在开发过程中容易以失败而告终。因此,将会大大影响这类单片机的普及应用。本文拟以 MCS-51 系列 8751H 单片机为例,探索采用普通开发机对其加密编程的简易方法步骤。

一、创造一个可编程、加密的硬件环境

一般普通开发机大多数不具备对 8751 芯片编程的条件,往往只能对普通 EPROM 芯片编程,因此,我们要先创造一个硬件环境。

首先,根据 8751H 编程方式表(见附表)、时序图以及其他条件(参阅单片机技术手册),在具有 2764 芯片编程器的基础上(可以是任何芯片编程器,这里仅以 2764 为例),制造一个适宜于 8751 各种编程方式要求的简易硬件环境来。具体方法是比较 8751H 编程方式的要求条件和 2764 编程的一般条件:

1. 先找出共同点直接代用。例如:8751H 的 ALE 引脚在编程时,需要输入 50ms 负编程脉冲,而 2764 的 \overline{CE}/PGM 引脚也需要输入 50ms 负编程脉冲,因此,可以用 2764 的引脚信号 \overline{CE}/PGM 代替 8751 的 ALE 引脚的输入讯号。

2. 再找出不同点,使用简单电路满足其要求。例如,附表中 RST 引脚的编程要求必须满足 $2.5V \leq V_{IHL} \leq V_{CC} + 0.5V$,否则不能编程,这在 2764 各引脚中是没有的,因此必须设法满足它,这里可以采用附加电阻分压的方法解决。

附图就是使用上述方法为 8751H 设计的简易编程板电路。图中,方框内是 8751 编程板,电路很简单,可制成一块小印刷电路板(甚至可以不用印刷板);方框外是 2764 的引脚名;将框内电路通过电缆和仿真插头插入任何编程器的 2764 插座上,即可对 8751H 编程。

另外还有一类开发机,如江苏省启东计算机厂生

产的 DVCC-51-ED 型开发机,自备对 8751 编程和校验的功能,但不能加密编程,这只要将原机中编程区里 8751 插座的 P2.6 引脚从接地处断开,象本文附图一样接入一个单刀双掷开关 K1,就可以方便的实现编程、加密编程的功能和转换。这类机器最好还要核一下 8751 的 RST 引脚电压是否能可靠满足编程要求。

二、编程步骤及其之间的关系

将制成的 8751 编程板插入开发机编程器 2764 插座中,并把 8751H 芯片插入编程板,然后分别按照下面的步骤操作:

1. 把编程板上的开关 K1 拨向 A 端,启动开发机和编程板进行 8751 查空,在保证全空状态后,按照所用机型对 2764 编程的方法开始编程,直到正常结束编程。切记要保证 V_{PP} 应符合 8751 的要求。

2. 开关 K1 拨向 A 端,按照对 2764 的查空、转移、校验等方法可对 8751 进行相同的操作。例如编程后的校验等。

3. 在编程、校验合格后,将开关 K1 拨向 B 端,按照对 2764 编程的方法再次对 8751 编程,即保密位编程或加密编程。这个过程一瞬间即可完成。应该说明,不同的开发机编程软件其反应是不同的:对于边编程边校验的编程方式,第一个单元编程后立即呈现校验错误,编程停止,这时,加密编程已经完成;对于只编程不校验的编程方式,机器将一直编程到地址终端,但只有第一单元编程是有效的,其余时间是无效运行,所以当显示器显示过第一单元地址后,可立即停止编程,这时加密编程也已完成。应当注意,一旦完成这个步骤,便不能再经校验等方法验证,只能通过应用系统的运行来证明。

上述每个步骤的功能都是独立的,第一步只编程不加密,第二步可对 8751 进行编程以外的各种操作,第三步只对 8751 加密,因此,第三步只有在前两步先完成的基础上才是有效的,否则,仅进行第三步操作是无用的。

该方法同样适用于其他此类单片机简易编程板的设计。

8751H 编程方式表

方 式	RST	PSEN	ALE	EA	P2.7	P2.6	P2.5	P2.4
编 程	V_{IHL}	0	0*	V_{PP}	1	0	×	×
禁 止	V_{IHL}	0	1	×	1	0	×	×

(下转 35 页)

一种简单的键盘接口电路

南京市1206信箱数控所(210012) 罗许建

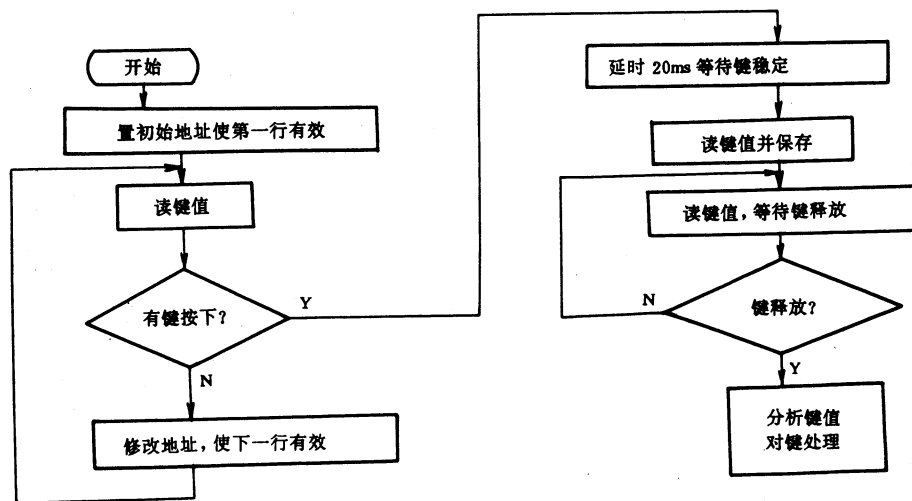
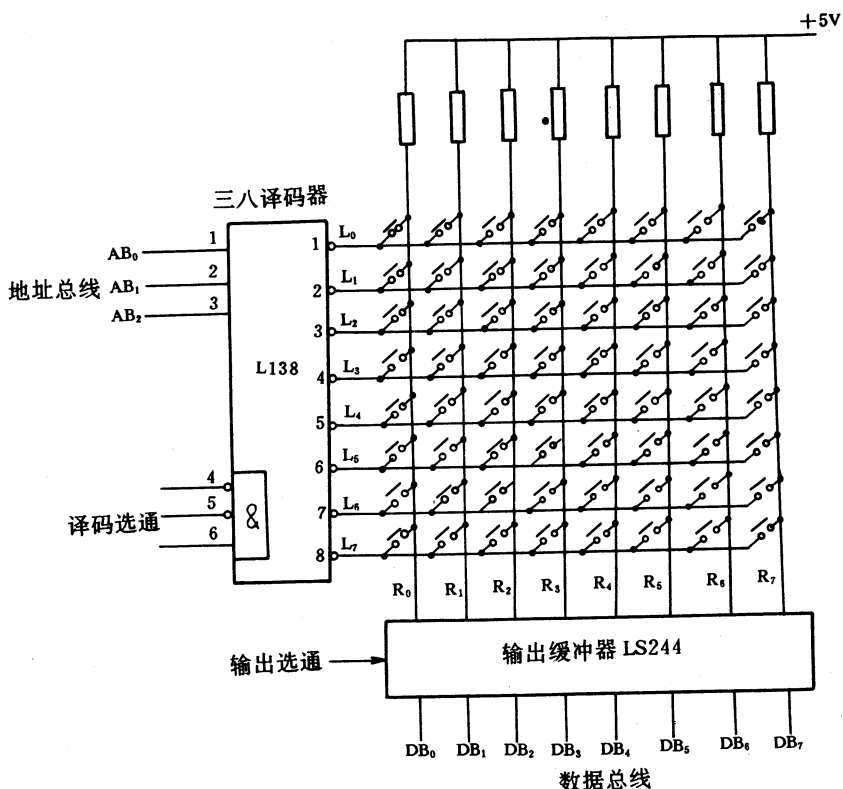
下面介绍一种简单的键盘接口电路,它由一个三八译码器和一个LS244缓冲器组成,构成一个由64个键组成的8×8矩阵。

如图,当三八译码器工作使得某行线有效时(低电平),如果这根线上有键被按下,则相应的列线为低电平,其它的列线为高电平。键盘扫描过程就是执行一系列的读操作,使每根行线依次为低,读列线内容。若读出数据中某位为“0”,则对应于该行、列的键被按下。这里的行线,列线都可以任意增加(增加一个三八译码器或输出缓冲器),因而扩展十分方便。

由于按键是机械结构,因而在键闭合的过程中会产生抖动。这样每按一次键,会产生不止一次状态变化,抖动时间一般在十几毫秒之内。一般用软件延时来等待按键的稳定,然后才读入键信息。

按键时,如果同时按下几个键,则读出的数据中有几个位为“0”。用户可根据实际情况对重键进行处理。

下面是键盘功能识别流程图:





电脑游戏机

第四章 FBASIC 语言的深入理解

山东苍山机械电子化学工业局(277700) 于 春

3. LEN

LEN 语句的功能是计算字符串的长度,即测量字符串共有多少个字符,其中包括空格、标点(除逗号)、运算符等。

语句格式为

LEN(字符串)

例11.

```
10 I. "PLEASE INPUT",A $
20 A=LEN(A $)
30 P. "LEN("A $" )="A
40 G. 10
RUN 显示
PLEASE INPUT? 键入 ABCD+3214
LEN(ABCD+1234)=9
..... ? 键入 A#B/"
LEN(A#B/" )=5
..... ? 键入 XYZABCDEFG
LEN(XYZABCDEFG)=10
...
```

4. LEFT \$ 和 RIGHT \$

LEFT \$ 语句的功能为从字符串的左边取出字符形成子字符串。其简写为 LEF.

RIGHT \$ 语句的功能为从字符串的右边取出字符形成子字符串。其简写为 RI.

语句格式为

LEF. (字符串,n1)

n1表示从字符串左边取 n1个字符。

RI. (字符串,n2)

n2表示从字符串右边取 n2个字符。

例12.

```
10 A $="ABCDEFGF"
20 B $="1234567"
30 F. I=2 TO 6
40 P. LEF. (A $,I),
50 P. RI. (B $,I)
60 N.
RUN 打印
AB 67
ABC 567
ABCD 4567
ABCDE 34567
ABCDEF 234567
```

5. MID \$

MID \$ 语句在随机手册中关于语句功能的介绍为“取出某个字符的字和数。”有点令人费解,建议改为“取出某个字符串中指定位置和个数的字符组成子字符串”。

字符串”。

MID \$ 简写 MI.

语句格式为:

MI. (字符串,取出位置,取出个数)式中:字符串最多可以有31个字符。取出位置均从左边数。

例13.

```
10 A $="FCS-90 F BASIC"
20 F. I=1 TO LEN(A $) ST. 2
30 P. MID $ (A $,I,3)
40 N.
RUN 打印
FCS CS- S-9 -90
90F 0FB FBA BAS
ASI SIC
```

从示例可以看出,MID \$ 语句兼备了 LEFT \$、RIGHT \$ 两个语句的功能,且比这两个语句使用更灵活、更方便。因此,一般使用 MID \$ 语句较多,而 LEFT \$、RIGHT \$ 两个语句使用的较少。

6. INSTR

INSTR 语句的功能是检索某一子字符串在另一字符串中的位置。

INSTR 简写 INS.

语句格式为

INSTR(被查串,子串)

该语句随机手册中没有介绍,请读者添上。

例14

```
10 A $="ABCDEABC"
20 B $="BCDEA"
30 C $="DE"
40 B=INS. (A $,B $):
C1=INS. (A $,C $):
C2=INS. (B $,C $)
50 P. "B $="B,"C $="C1"-C2
RUN 显示
B $=2 C $=4-3
```

例15. 设计一个绘制心脏图形的程序。

```
10 CLS
20 A $="AIIIIIIIIIGIGGGGH"
30 B $="HHBHHBBEBEEEE"
40 K $="ABCDEFGHI"
50 I $="ABCGHIDEF"
60 X=13:Y=22:I=0
70 GOS. 100:SWAP A $,B $:GOS. 100
80 IF I=1 T. LOC. 0,0:END
90 IF I=0 T. SWAP K $,I $:I=1:G. 70
100 L=LEN(A $):F. P=1 TO L
```

```

110 K=P
120 IF I=0 T.140
130 K=L+1-P
140 C=INSTR(K$,MID$(A$,K,1))-1
150 V=C/3:H=C MOD 3
160 X=X+H+3*(H=2):Y=Y+V+3*(V=2)
170 LOC.X,Y:P.CHR$(207):N.:RE.

```

RUN

(打印出一心的图形。)

由于 F BASIC 语言没有画曲线的语句和函数(如正弦、余弦、求平方根等),因此,仅使用前三章介绍的语句画一心形将很困难。本例巧妙地使用了字符处理语句 INSTR、MID\$、LEN 就画出了心形。

为便于理解,特对程序说明如下:

本程序共分四部分:

20~60行为变量赋值。

100~170行为绘图子程序。

80 行为结束判断处理语句。

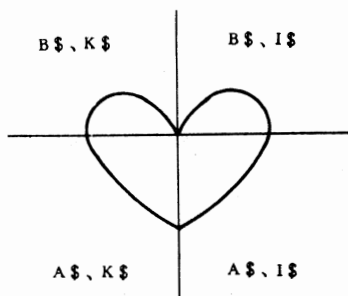
70、90两行为绘图控制语句,这两行是本程序的关键行。

程序的执行过程如下:

- (1) 变量赋值后,通过70行第一个 GOSUB 语句转绘图子程序,画出心形的左下部分。这时 A\$、K\$参与绘图。返回70行。
- (2) A\$、B\$交换后,由第二个 GOSUB 语句转绘图子程序,画出心形的左上部分。这时 B\$、K\$参与绘图。
- 返回70行,执行90行。(由于 I=0跳过80行)
- (3) K\$、I\$交换。令 I=1,返回70行。由第一个 GOSUB 语句转绘图子程序,画出心形的右上部分。这时 B\$、I\$参与绘图。返回70行。
- (4) B\$、A\$交换后,由第二个 GOSUB 语句再转绘图子程序,画出心形的右下部分。这时 A\$、I\$有效。

返回70行,由于 I=1,执行80行程序运行结束。

若把心形分成四部分,各字符串在绘图中的有效区域如下图。



关于画图子程序的分析,读者可以列一张表,详细列出 P、K、C、V、H、X、Y 的变化,即可看出用字符串语句绘图的巧妙性和方便之处。

另外,上例中160行使用了特殊判断式(H=2)、(V=2)。这种判断式的值如下:(以 H=2为例)

```

当 H<2 (H=2)=0
当 H=2 (H=2)=-1
当 H>2 (H=2)=0

```

同样,若判断 H>2,则有

```

当 H<2 (H>2)=0
当 H=2 (H>2)=0
当 H>2 (H>2)=-1

```

即判断式以括号内为形,以判断内容为体。当判断内容成立时,判断式的值为-1;当判断内容不成立时,判断式的值恒为0。请读者记住这种判断式及使用方法,游戏程序设计中经常用到。

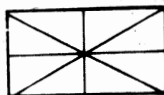
在随机手册、说明书及一般书籍中仅介绍了如何使用 F BASIC 语言进行计算程序设计和使用卡通图案进行游戏程序设计,而忽略了绘图程序设计的介绍。这使不少使用者产生了误解,认为 F BASIC 语言不能设计绘图程序。其实, F BASIC 语言也有着很强的绘图功能。例15就是一个证明。尽管它看起来不如 BG GRAPHIC 方便,但还有胜过 BG GRAPHIC 的方面。特别是 BG GRAPHIC 在 Y 轴方向上有下面三行不能绘图,从而使绘图有效面积缩小了13%,这就限制了它的使用。而用程序绘图则纵横自如,不受限制,充分利用了有效画面。作为 F BASIC 语言的提高篇,再介绍几个设计程序绘制图案的实例,以开阔思路,使读者掌握设计绘图程序的方法和技巧。

例16. 编制程序绘制米字形图案,用方框围起来。

```

10 CLS
20 F.X=0 TO 25
30 LOC.X,0:P."*"
40 LOC.X,21:P."*"
50 LOC.X,10:P."*"
60 LOC.X,(21-21*X/25):P."*"
70 LOC.X,21*X/25:P."*"
80 N.
90 F.Y=0 TO 21
100 LOC.13,Y:P."*"
110 LOC.0,Y:P."*"
120 LOC.25,Y:P."*"
130 N.
RUN
打印

```



例17. 编制绘出长方体图案的程序

```

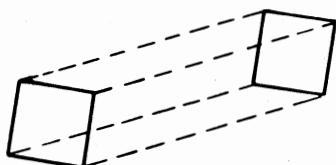
5 CLS
10 F.X=0 TO 9
20 LOC.0,X:P."*"
30 LOC.X,0:P."*"
40 LOC.9,X:P."*"
50 LOC.X,9:P."*"
60 F.X=17 TO 26
70 LOC.X,9:P."@"
80 LOC.X,18:P."@"

```

```

90 F.Y=9 TO 17
100 LOC. 17,Y:P. "@"
110 LOC. 26,Y:P. "@" :N.
120 F.X=1 TO 16
130 LOC. X,9 * X/17:P. CH. (205)
140 LOC. X,9 * X/17+9:P. CH. (205):N.
150 F.X=10 TO 25
160 LOC. X,9 * X/17-4:P. CH. (205)
170 LOC. X,9 * X/17+5:
    P. CH. (205):N.
180 E.
RUN    打印(如图2)

```



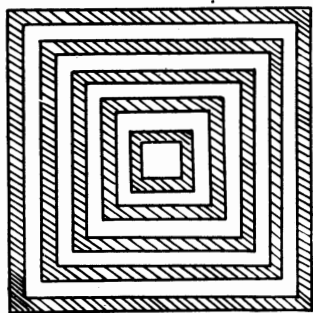
为便于理解,上两例均使用了单循环。也可以采用多重循环以简化程序。读者可以自己练习。

例18. 编程绘制五个正方形套在一起的图案。

```

10 CLS: X=13: Y=11: A$=CH. (253)
20 F. N=1 TO 10 ST. 2
30 F. L=-N TO N
40 LOC. X+L,Y-N:P. A$
50 LOC. X+N,Y+L: P. A$
60 LOC. X-L,Y+N: P. A$
70 LOC. X-N,Y-L:P. A$
80 N. : N.
90 E.
RUN    打印(如图3)

```



以上程序可以再加一段程序,使方框从里圈陆续显示,显示出全部图像后,再从外圈开始陆续消失,全部消失后再陆续显示。如此,周而复始,产生较强的动感。读者可自己练习。

还可以编制绘制五角星、八角形、圆等图形的程序。由于篇幅所限,不一一列举,读者可自己练习。

五、出错处理语句

F BASIC 语言共有程序出错处理语句五条,随机手册中均未介绍,请读者自己添上。

在一个较复杂的游戏程序中,不可能把各个方面都考虑周全而编制相应的判断处理程序(那样程序将变得很庞大)。但在游戏过程中,情况千变万化,可能出现如出界、溢出等错误。出现错误时,计算机将停止运行,给出出错提示,从而使游戏中断。往往刚玩出一个好的积分,因游戏中断再从头开始,很令人扫兴。若使用出错处理语句,就可以消除这一弊病。使用出错处理语句后,当程序运行出错时,计算机自动转出错处理子程序,处理后,再返回继续下面的程序运行。这样可以把错误处理在程序的运行中,即自动处理。因此,引入出错处理语句,方便了程序的运行。

1. 出错转移语句(ON ERROR GOTO)

ON ERROR GOTO 简写 O. ERR. G.

ON ERROR GOTO 语句的功能是设置错误转移入口,以便对错误进行拦截处理。语句格式为:

ON ERROR GOTO 行号

式中行号为出错处理子程序的首行号。

该语句的意思是,当程序出错时转去执行X×行。

2. 错误处理返回语句(RESUME)

RESUME 简写 RESU.

RESUME 语句的功能是通知 BASIC 从出错处理子程序返回,并恢复执行原程序。

语句格式为

RESUME [NEXT][行号]

当程序中设计了 ON ERROR GOTO 语句后,在程序出错时,则令 BASIC 转去执行出错处理子程序。处理完后要返回到原程序,使用 RESUME 语句可以实现返回。返回语句有三种情况:一种是 RESUME 语句后跟 NEXT,返回时,程序就会在出错语句的下一行恢复运行;第二种是在 RESUME 语句后跟行号,返回时,程序就会在规定的行号恢复运行;第三种情况是省略 NEXT 和行号,程序就会返回到原出错语句,由于原出错语句有错,又转出错处理子程序,从而进入死循环状态。所以,使用时要牢记不要省略 NEXT 和行号。

例19.

```

10 CLS
20 O. ERR. G. 70
30 I. "Please Input";X
40 A=10000/X
50 P. "10000/"X="A
60 G. 30
70 P. "X="X;
80 P. "0 Bu Neng Zuo Chu Shu"
90 RESU. 30

```

RUN

Please Input? 键入 30

10000/30=333

Please Input? 键入 0

X=0 0 Bu Neng Zuo Chu Shu

Please Input?

从上例运行可以看出,当输入0后即打印零不能

做除数,可继续输入。虽然出错,运行并不停止。这就是出错处理语句的作用。

3. 测试出错行号语句(ERL)

ERL 语句的功能是给出最近一次出错语句所在的行号。这是为测试出错行号而设置的语句。语句格式为:

ERL

4. 测试出错代码语句(ERR)

ERR 语句的功能是给出最近一次出错语句的出错代码。语句格式为

ERR

每一类错误都有一个确定的代码。错误的类型不同,出错代码也不同。当程序没有错误时,ERR=0(注:

表一 常用错误代码表

错误类型 代码	错误表示 简写	错误说明
0	NF	无 FOR 却有 NEXT
1	SN	语法错误
2	RG	无 GOSUB 却有 RETURN
3	OD	DATA 语句中的数据不足
4	IL	语句的叙述或称呼错误
5	OV	演算结果超限(溢出)
6	OM	内存不足。
7	UL	GOTO、GOSUB、IF 的行号未定义。
8	SO	排列变数的添字在规定外。
9	DD	排列设双重定义。
10	DE	用零做除数。
11	TM	变数的类型不一致。
12	ST	字符串超过31个。
13	FT	式子过于复杂
14	CC	CONT 无法继续执行程序。
15	UF	
16	MO	参数所需要的指令未指定。
17	TP	无法从磁带读出资料。
18	NR	
19	RE	
20	NB	
21	UP	

0同时也是—类错误代码)。如某—类错误发生时,ERR便记录下这个错误的类型代码。当又有新的错误发生时,ERR 又会变成新的错误类型代码。即 ERR 中所存的总是最近一次发生的错误的代码。F BASIC 共有22个错误类型代码,下表分别列出各错误类型代码所代表的错误信息简写和部分说明。

掌握了每种错误代码所代表的错误类型,就可以设计相应的错误处理子程序。

5. 定义出错代码语句(ERROR)

ERROR 简写 ERR.

ERROR 语句的功能是定义用户自己的出错代码。

在程序设计中,有时要根据出错类型做相应的转移处理。而往往有些人设定的错误在 BASIC 系统中并未设定,如 X 不能大于1000或者 X 不能等于500等。这些实际上都不是错误,不过是在程序设计中一些特定场合的需要而人为设定的。这些错误没有设定的错误代码,因此必须由用户自己设定。

ERROR 语句的格式为

ERROR n 式中:n取值0~21。

例20. 设定出错代码的演示。

```

10 CLS:ON ERROR GOTO 100
20 I. "Please Input";X
30 IF X>1000 T. ERR. 10
40 IF X=500 T. ERR. 15
50 IF X<100 T. ERR. 20
60 P. "OKOK":G. 20
100 IF ERR=10 T. P. "X>1000"
110 IF ERR=15 T. P. "X=500"
120 IF ERR=20 T. P. "X<100"
130 P. "ERL="ERL,"ERR="ERR
140 RESU. 20
RUN
    Please Input? 键入 1234
    X>1000
    ERL=30      ERR=10
    Please Input? 键入 500
    X=500
    ERL=40      ERR=15
    Please Input? 键入 200
    OKOK
    Please Input? 键入 50
    X<100
    ERL=50      ERR=20
    Please Input?
    :
```

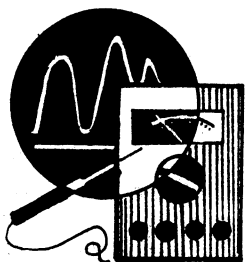
消 息

电子工业出版社《电子与电脑》杂志编辑部将于8月31日在北京举办为期四周的微型机二级维修技术学习班。

培训内容:PC/XT/AT(8088、80286)主机板、软硬盘及适配器、显示器及适配器、打印机及适配器、键盘、主机电源、UPS 的工作原理、维修方法,设有典型故障演示课以提高实际维修能力与技巧。

培训费:390元 资料费:70元(实报实销)

联系人:中国人民大学信息中心(100872)胡野红
电 话:255.5431-2518(或2514)



维修经验谈

维修打印头应注意的几个问题

西安532信箱(710061) 蔡世清

打印头是打印机的关键部件之一,也是一个精密的机电组合件。打印机使用久了,断针是打印头中常见的故障。根据我的体会,修复断针时应注意以下几个问题。

1. 对于有断针的打印头,不宜仅补穿上好针完事,而应该拆除全部打印针,将空头仔细进行清洗(用酒精棉球),彻底清除其中的油污。对拆下的打印针,也要逐根用酒精棉球将其上面的油污擦净。擦洗时要格外细心,以免把针折断。这样装出来的打印头如同新的一样。

2. 穿针前先拿一根新针做标准,把拆下的旧针一一与之对照,比较长度,并去掉长度不够的旧针。做此项工作时,一定要仔细、准确。同时要注意,新针往往比

未磨损的旧针长,头脑中要留出余量,以免把合格的旧针也剔除掉。

3. 穿针时宜将增补的新针穿在同一边附近位置。有的打印针(如 TH3070R2)其新旧长度之差约2mm,穿好后露出部分过长,如用油石或细砂布打磨,则易引起折断。对此,宜在穿针前用手术剪轻轻剪掉1mm左右,余下部分待穿好后再用油石或细砂布打磨校正。

4. 用油石或细砂布打磨打印头时,开头用力要较轻。宜将细砂布或油石平放于玻璃板上,打印头一定要和细砂布或油石面垂直,切勿歪斜。用力要逐渐加大,不能用猛力。当全部打印针与陶瓷定位片在同一平面上时,这个打印头就算装好了。

TH3070点阵式打印机常见故障分析与检修(下)

国务院办公厅秘书局技术处(100017) 刘立华

【故障实例五】印字连点故障

这个故障现象和“印字漏点”刚好相反,它的表现形式是在不该打印的地方,仍打印了,某一针(或几根针)连续出针在打印的字形中出现了一条连续的横线,这种故障完全出在电路上,检查和排除比较简单。

和“印字漏点”故障的检索一样,首先要确定是哪一针(或几针)连续,然后参照“印字漏点”故障中的针与各芯片的对应关系,检查六片 TD62064和三片 LS374。若是确实判定这几个芯片没有问题和话,再检查另外一个 LS374(IC46)。这个故障的范围,就在这十个芯片之内,而且这种故障一般都是出在 LS374芯片上。

【故障实例六】打印不出字故障

这种故障的表现形式为在自检或联机打印时,字车正常走动。走纸也正常,唯独打印无字形,即打印头不出针,检查这种故障的流程图如图6所示。

在上边流程图中,实际上包含着三个方面的检查内容。依次是7.5中断故障,出针控制信号故障及出针驱动信号的故障,图中“打印头插头插好”与“打印头线

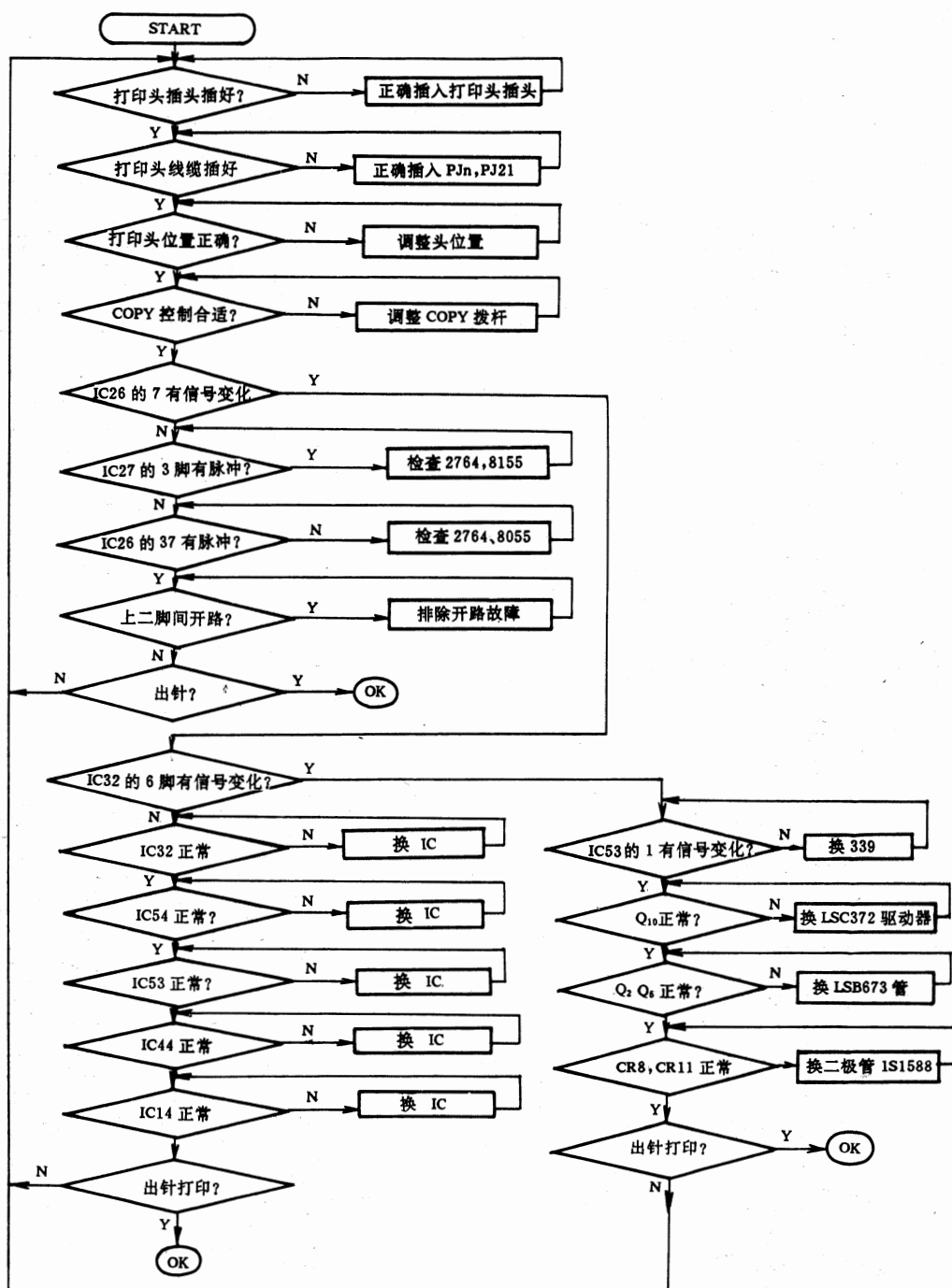
缆插好”两步,是很重要的一个检查内容,如果插得不对。最严重的后果将会是烧坏打印头的线圈。

电路检查中,IC54的型号为“555”,被称作“时基电路”,主要作为时间控制用,检查这个芯片,主要是检查一下它的第三脚有无信号变化就可以了,其它脚的输入与输出都与第三脚是密切相关的,也就是说,如果 IC53(339)的第二脚和第十四脚有信号变化,而“555”(IC53)的第三脚无信号变化,则说明“555”芯片已经坏了。

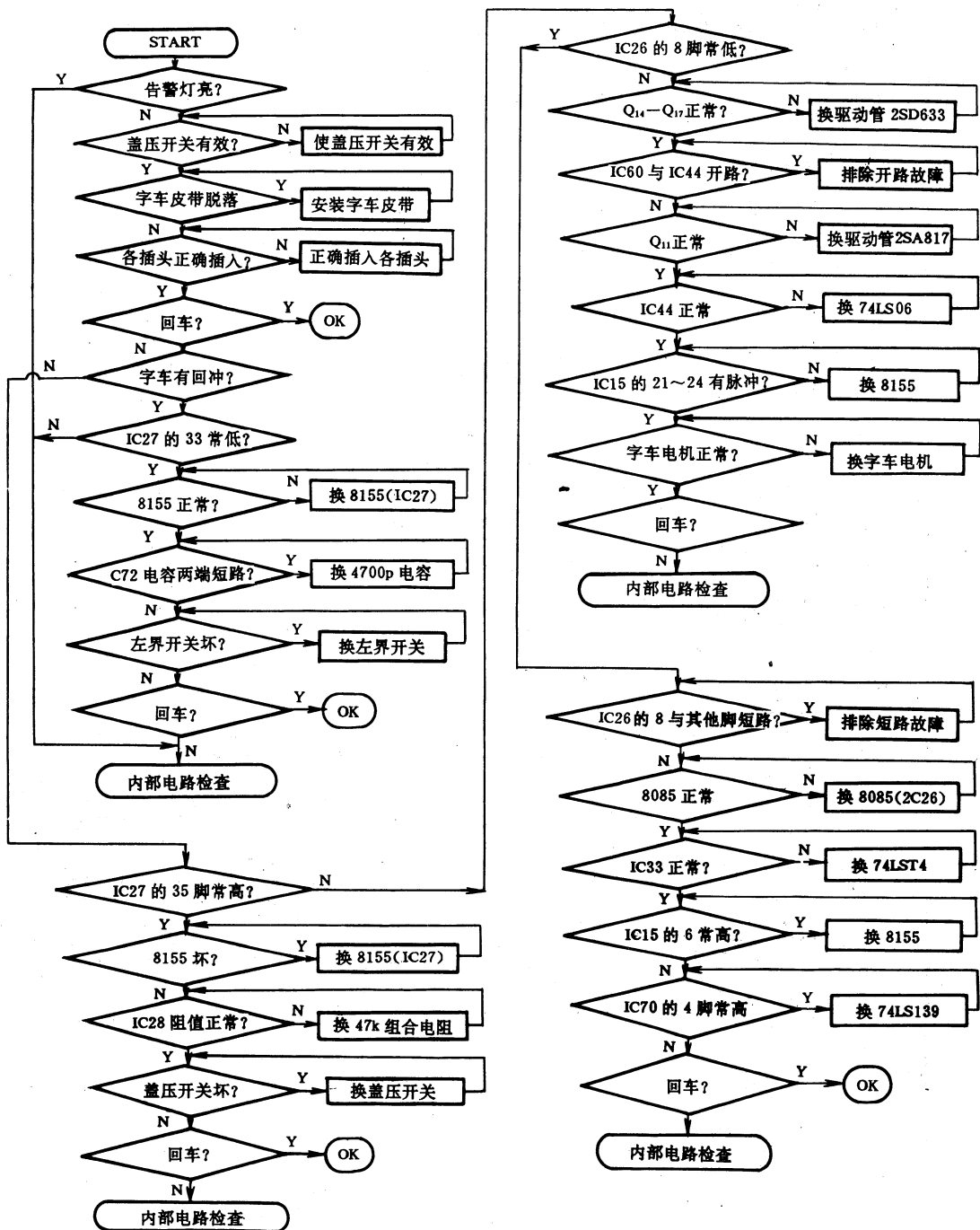
【故障实例七】加电后不回车故障

打印机加上电而打印字车不动或不回到“左界”是打印机最常发生的故障,这种故障由于其牵涉面比较广,因而隐蔽性也比较强,所以检查起来也比较棘手。

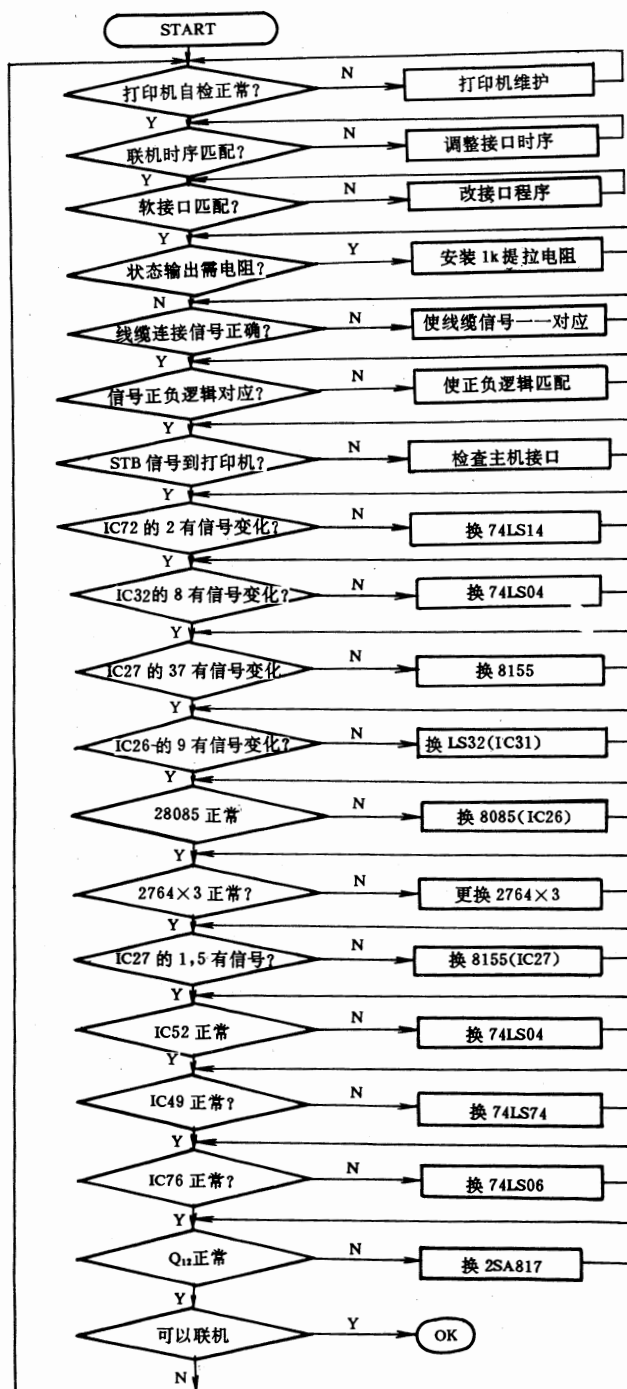
在检查这种故障时,首先要检查一下,看看保险丝是否已熔断。若换好保险丝仍然熔断,则说明电源有毛病。若电源没问题,那么,还需检查一下打印机控制板上的插头是否插错、插反或未插,在确认以上检查无误后,即可进入电路部分的检查,加电不回车的检查流程图如图7、图8所示。



图(6)“打印不出字故障”检修逻辑图



图(7) “加电后不回车故障”检修逻辑图



图(8)“并行接口不能联机故障”检修逻辑图



普及型 PC 个人用户软件交流联谊活动 问题解答(六)

读者联谊

北京中国农科院计算中心(100081) 王路敬

17. CGA、MCGA、EGA、CEGA、CMGA、VGA、TVGA 各表示什么含义?

它们分别代表微机系统显示器视屏标准。

CGA 是 Color Graphics Adapter(彩色图形适配器)的缩写。这是 IBM PC 系列机一种显示标准,该标准有 7 种工作方式,其中 4 种字符方式,3 种图形工作方式,它们是:

- 40×25 字符方式 2 种颜色
- 40×25 字符方式 4 种颜色
- 80×25 字符方式 2 种颜色
- 80×25 字符方式 4 种颜色
- 320×200 图形方式 4 种颜色
- 320×200 图形方式 2 种颜色
- 640×200 图形方式 2 种颜色

IBM PC/XT、GW0520A 等机即为这种显示标准

MCGA 是 Multi-Color Graphics Adapter(多色图形适配器)的缩写。它是低档 IBM PS/2 系列机的视屏标准。它除能提供 CGA 所有工作方式外,增加了下述两种图形工作方式:

- 640×480 图形方式,2 种颜色
- 320×200 图形方式,256 种颜色

EGA 是 Enhanced Graphics Adapter(增强型图形适配器)的缩写。EGA 能仿真 CGA 的所有功能,支持 CGA 的所有工作方式,并且新增加了下列几种工作方式:

- 80×25 字符方式,2 种颜色(单色)
- 320×200 图形方式,16 种颜色
- 640×350 图形方式,16 种颜色
- 640×350 图形方式,4 种灰度(单显)
- 640×350 图形方式,16 种颜色

CEGA 是 Chinese Enhanced Graphics Adapter(中文增强型彩色显示系统)的缩写。该显示标准与 CGA 全兼容。同时增加了与 EGA 全兼容方式,640×350 图形方式颜色由 8 种提高到 64 种,并做到了与 VGA 的 640×480 高分辨率图形方式兼容。

CMGA 是 Chinese Monochrome Graphics Adapter(中文增强型单色多灰度显示系统)的缩写。这种显示标准与 Hercules 单色方式(720×350)全兼容,与 CGA 彩色显示方式全兼容,具有 16 级灰度,可自动把彩色转换成灰度,增加了 640×480 单色高分辨率图形方式。

VGA 是 Video Graphics Array(视屏图形阵列)的缩写。这是 IBM 公司为其 PS/2 系列中高档机设计的一种高性能视屏标准。VGA 与 EGA 高度兼容,它能支持

EGA 提供的所有工作方式,并且增加了下列工作方式:

- 640×480 图形方式,2 种颜色
- 640×480 图形方式,16 种颜色
- 320×200 图形方式,256 种颜色

自从 IBM 新的图形标准 VGA 卡推出以后,各种高性能的高分辨率彩卡不断问世,TVGA 卡就是其中的一种。TVGA 卡不仅支持 CGA、EGA、VGA 的图形标准,而且提供了比 VGA(640×480)标准更高的视频分辨率(1024×768)和图形功能。采用 1024×768 高分辨显示模块进行 CAD 一类的图形处理,无论是图形质量还是工作效率都有较大的提高。一般 TVGA 卡配置到处理速度较快的 386 机上。由于 TVGA 卡性能高,价格便宜,因此,在新购置的各类微机中,都可按照用户的要求配置这种彩卡。

18. 使用显示器应注意哪些问题?常见故障有哪些?

首先要注意的是显示器的电源开关,有些显示器没有电源开关,它是和主机电源开关在一起,开启主机电源时,显示器电源也随之被打开。不过大多数显示器电源不和主机在一起,它有自己的电源开关。在计算机系统启动时,一般都要先打开显示器开关,然后再开主机开关,以防止瞬间电流脉冲影响主机,关闭时则要求先关主机,然后再关显示器。

第二,有些是专门的显示器可供不同型号或种类的微机使用,因此,它们背面可能有选择开关,使用时要把系统配置选择开关拨到相应的位置,否则屏幕上出现杂乱无章的信息。

第三,显示器一般都要有亮度、对比度、色彩等调节选配,我们可以根据自己的爱好,调节选择一定亮度、对比度和色彩等。

在使用中常见故障:屏幕一片黑,无任何显示内容。碰到这类故障首先要区别是系统本身的故障还是显示适配器或是显示器本身的问题,可根据故障表现出的其他现象逐个判别。例如 PC 机单色显示器若加电后屏幕上没有任何显示,随即听到一长二短的鸣叫声,这说明可能是系统板上的 5、6 两位开关设置不对或显示板问题。屏幕滚动也是常见故障之一,这种现象一般是显示器的帧频不同步,调整显示器的垂直同步旋钮即可恢复;如果调整无效,就需要检查适配器的输出波形是否正常了。屏幕有时出现杂乱无章的字符或图形。出现这种故障现象的实质就是数据在传送过程中

(下转 10 页)