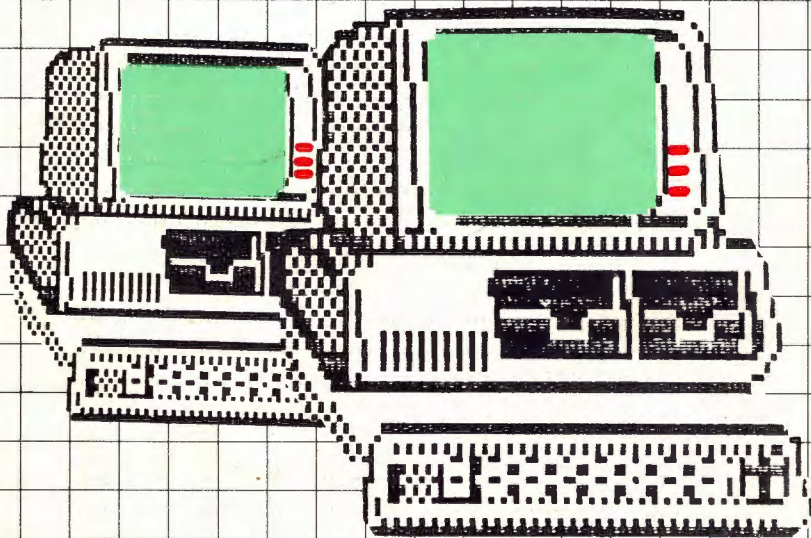


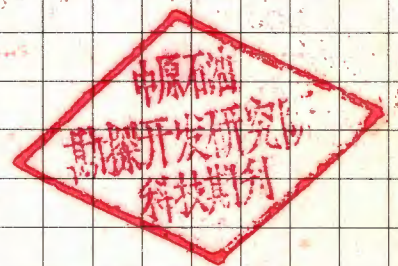
计算机应用研究

APPLICATION RESEARCH OF COMPUTERS



1989

5



《计算机应用研究》杂志社

《计算机应用研究》杂志办刊单位

四川省电子计算机应用研究中心

贵州省科学技术电子计算机中心

安徽省计算中心

新疆电子计算中心

吉林省计算中心

青海省测试计算中心

甘肃省计算中心

四川省电子学会

《计算机应用研究》杂志编辑委员会

主任委员：张执谦

副主任委员：李泽民

委员：余凯 张国栋 贾洪钧

曾光初 王小华 朱景生

《计算机应用研究》杂志社董事会

董事长：周赛渝

董事：唐珍 郑国基 陆慰椿

秦小竹 龚宇清 黎瑰常

计算机应用研究（双月刊）

（公开发行）

一九八九年

第六卷 第五期（总31期）

主 编：张执谦

副 主 编：李泽民

本期责任编辑：张明清

编 辑 出版：《计算机应用研究》杂志社

印 刷：西南冶金地质印刷厂

出 版 日 期：1989年9月30日

本刊通讯地址：成都市人民南路4段11号附1号

本刊邮政编码：610015

订 阅 处：全国各地邮局

总 发 行：成都市邮政局

每 册 定 价：1.30元

本刊邮发代号：62—68

国内统一刊号：CN51—1196

广告经营许可证川蓉工商广字005号

目 录

软 件 篇

屏幕字符显示形式的美化方法	陈庆章 (1)
一个文稿打印程序	何宇才 (3)
用 TRUE BASIC 语言读取 BASIC 数据文件	曹增强 (7)
dBASE III 的彻底解密	聂崇峡 (9)
在 Apple—II 显示屏上直接绘制高分辨率图形	任伍林 (11)
浅谈数据库索引文件的使用	孙春秋 张翼宗 (13)
甘肃省高考数据处理软件的设计及应用	周仲宁等 (15)
用计算机解 Reve 之谜	程贵平 (16)
FORTRAN 与 MACRO 之间的最佳数据交换	叶高等 (18)
APPLE II 微机体感诱发电位检测系统软件设计	彭利安等 (19)
简短的西文提示汉化程序	赵超 (22)
简介下拉式菜单	刘宇菁 (23)
“怡红夜宴图”验证程序设计	黄大海 (25)
让您的 PC 机奏出悦耳的曲子	刘康 (26)
NAG 应用软件生成系统简介	徐宝文等 (28)
用 BASIC 程序恢复误删文件的方法	于桦等 (30)

硬 件 篇

电摄影过程的单片机控制	胡盛斌 (31)
微机控制全自动电子乐队	罗建平 (35)
IBM—PC Z80 在线仿真器接口板的研制	徐飞虹 (37)

系 统 篇

通用快速数字信号处理系统的设计与实现	汪亚南等 (40)
多头高分辨率彩色显示系统	丁锋 (44)
AR—2463 打印机实现高分辨图形/字符屏幕拷贝	王南江 (45)
采用 NSC 810A 芯片的 IBMPC 接口	柳影等 (53)

维 修 篇

关于软盘零磁道破坏后的数据修复	阮高华 (56)
PDP11 RSX11M 操作系统中非正常文件的恢复处理	施剑秋等 (57)
KOSMIC KH—16 内存故障的检修	吴达 (60)
一种硬盘故障的修复方法	苗沛淋 (62)

信 息 篇

控制计算机图形质量的三个因素	王性卫 (62)
ESPRIT: 欧洲信息技术研究与开发的战略计划	刘椿年 (64)
培养 CAD 人才 推广 CAD 技术	陈冠清 (66)
NITLISP 语言通过鉴定等两则	李相彬 (67)
工科高等数学题库系统研制成功	张伟 (67)
我国色谱数据处理工作正向现代化水平迈进	吕宜男 (67)
计算机藏品管理系统软件简介	杨战 (68)
简介一种用于打印机的绘图程序库	孟洪 (69)
CX (川星) 286 微机简介	于作达 张顺全 (70)
新疆电子计算中心简介	周步祥 (封三)
信息三则	(21, 69, 封三)

新疆电子计算中心简介

新疆电子计算中心创建于一九七八年，系新疆区科委直属的计算机专业科研机构，是新疆区科技体制改革第一批试点研究院所之一。

新疆电子计算中心拥有一支政治上、业务上均训练有素、能打“硬仗”的科研队伍。全中心共有职工55人，其中科研人员占80%，高级职称2人，中级职称11人。十多年来，在党和政府的正确政策的指导和新疆区科委的直接领导下，共承担了二十多项科研项目，其中有七项已获新疆自治区科技进步奖；为了适应改革的形势需要，该中心还于一九八七年五月成立了计算机技术经营开发部，为新疆地区的计算机事业的发展作出了突出贡献，在广大用户中赢得了广泛的信誉。

新疆电子计算中心的主要服务业务范围是：

1. 计算机网络的开发应用与研究；
2. 软件工程、管理信息系统、教学软件等各种软件的开发应用；
3. 微机应用于工业监控及智能化仪器仪表

的开发应用；

1. 计算机辅助设计、制造、预测、决策、管理、诊断、教育等等。

5. 经营、维修各种微型机及外围设备，机房改造与装修；

6. 为用户提供各种计算机技术咨询，技术服务或机时服务，培训各种软、硬件技术人员。

另外，新疆电子计算中心还与新疆区主要电子技术科研、企业单位密切合作，是新疆电子集团公司的成员之一，该中心愿意在区内外、国内外与各计算机同行密切合作，共同努力奋斗，为中华民族的崛起奋勇拼搏！

热忱欢迎各单位与新疆电子计算中心接洽业务，并请来人来函联系。

地址：新疆区乌鲁木齐市北京南路40号

电话：76646，75415，37139，76638

电挂：3007

（本刊记者 周步祥）

FN-1型发热耐久控制台简介

吉林省计算中心研制成功的FN-1型试验台是按照国际IEC—335—1和B347061标准研制的，它的研制有利于测试上述标准，对保证和提高我国家用电器质量有着重大意义。

该试验台系统设计合理，功能强，并有耐久性、温度检测、绕组温升等九项功能，并可根据用户要求，增加或减少其功能。该系统利用TP801单板机控制，采用人机对话形式，操作简单、显示清晰、使用方便，大大提高了工作效率，减少了操作人员的劳动强度。

该系统系国内首创，具有国内先进水平。

APPLICATION RESEARCH OF COMPUTERS

Vol.6 No.5 (Total 31)

CONTENTS

THE ARTICLES ON SOFTWARE

A Beautified Method of Character-display Form on Screen.....	Chen Qingzhang (1)
A Printing Program of Manuscript.....	He Yucai (3)
Reading-out the Data File in BASIC by TRUE BASIC	Cao Zengqiang (7)
Thoroughly Deciphering on dBASE III.....	Nie Chongxia (9)
Directly Drawing the Graph with High-resolution on Screen of APPLE II	Sun Chunqiu and Others (13)
Designing and Applying about Data Processing System of College Entrance Examination in Gansu Province.....	Zhou Zhongning and Others (15)
Found a Clue to the Mystery of REVE by Computer.....	Cheng Guiping (16)
The Optimum Data Exchange between FORTRAN to MACRO.....	Ye Gaoying (18)
Designing the Program of Potential Detection System with Brought out on APPLE II.....	Peng Lian and Others (19)
A Brief Program in Chinese of Prompting in English	Zhao Chao (22)
Simple Introducing about the Push-down MENU.....	Liu Yuqing (23)
Designing about Verifying Program of "The Order Drawing of the Party at Night in Yi Hong Yard"	Huang Dahai (25)
Let Your PC Computer Play the Melody.....	Liu Kang (26)
Simple Introducing about the Generating System of NAG Applying Software	Xu Baowen and Others (28)
The Method of Restoring File Deleted by BASIC program.....	Yu Hua and Others (30)

THE ARTICLES ON HARDWARE

The Controlling Process of Taking Photography by Single Chip Processor.....	Hu Shengbin (31)
Controlling of Automatic Electron Orchestra by Microcomputer.....	Luo Jianping (35)
Developing for Interface of Line-emulator on IBM-PC Z80.....	Xu Feihong (37)

THE ARTICLES ON SYSTEM

Designing and Realizing of General-Rapid System for Digital Signal Processing.....	Wang Yanan and Others (40)
The Displaying System with More-head, Colour and High-resolution.....	Ding Feng (44)
Copying of High-resolution Graph/Character on Screen to AR-2463 Printer.....	Wang Nanjiang (45)
The Interface on IBM PC by NSC 816A Chip.....	Liu Ying and Others (53)

THE ARTICLES ON MAINTENANCE

- Restoring the Data Destroyed on Zero-magnetic Track of Disk..... Yuan Gaohua (56)
Renovating and Processing about Un-normal File in Operating System
of PDP 11 RSX11M.....Shi Jianqiu and Others (57)
Examining and Repairing for Internal Storage Bug in KOSMIC KH-16.....Wu Da (60)
A Repairing Method for Hard-disk Malfunction.....Miao Peilin (62)

THE ARTICLES ON INFORMATIONS

- Three Factors about Controlling Quality of Computer Graph.....Wang Xingwei (62)
ESPRIT --- Europe Strategic Plan with Reseaching and Developing
Informations Technology.....Liu Chunnian (64)
Fostering Talent in CAD, Spreading over CAD Technology.....Chen Guanqing (66)
NITLISP Language Adopted Appraisal and Others.....Li Xiangbing (67)
The Examination Questions Base System of Engineering Course
Higher-mathematics Unveiled.....Zhang Wei (67)
The Chromatogram-data-processing Striding toward Modernization
Levels in our Country..... Lu Yinan (67)
A Brief Introduction to the Ware-managment System by Computer.....Yang Zhan (68)
A Brief Introduction to A Plotting Program Library with Printer.....Meng Hong (69)
A Brief Introduction to Electronic Computing Centre of Xinjiang
..... Zhou Bu xiang (Page3 of the Cover)
Three Pieces of News

APPLICATION RESEARCH OF COMPUTERS

(Publicly Publishing) (Bimonthly)

September 1989 Volume 6 No.5 (Total 31)

Chief Editor: Zhang Zhi Qian

Vice-Chief Editor: Li Ze min

Responsibility Editor for this Issue: Zhang Ming Qing

Editor/Publisher: Editorial Department of this periodical

Address: 11—1 Four section Ren Ming Nan Lu, Chengdu

Post Code: 610015

Press: South West Metallurgical Geology Printing House
of Metallurgical Industry Ministry

Subscribe Address: The post of the All Localities of the Country

Central Dispatching in the Inside of The Country: The post of Chengdu

Unified Number in the Inside of the Country: CN51—1196

Number of the Post Publishing: 62—68

屏幕字符显示形式的美化方法

商业部安徽商业管理干部学院 陈庆章

应用软件的设计者们常常想改变屏幕字符的显示形式(styles),如将正常的显示形式改为黑白翻转(Reverse)、增添下划线(Underline)、明暗闪烁(Blink)、加亮(Bold),以及它们的结合如闪烁字符再加下划线等。这样做可使屏幕上显示的文本具有引人入胜的生动感。实现这一目标的方法有数种,如设计一个屏幕管理程序、采用屏幕写(Screen Writing)技术等。但这些方法或是工作量太大,或是引起速度上的较大牺牲,甚至还要对CONFIG.SYS文件和ANSI.SYS文件加以改造,对程序设计者并不方便。本文提供一种方法,既克服上述缺点,又简捷方便,原理也较易理解。它称为存储器写(Memory Writing)方法。

IBMPC屏幕显示是采用存储器映象(Memory-Mapping)方式。单显占用地址B000:0000,彩显占用地址B800:0000。在该地址范围内写入数据,即意味着使此数据显示在屏幕的相应位置上。由于整个屏幕分为25行,每行80个字符,每个字符占去二个字节,所以整个屏幕实际占用了连续的4000个字节。屏幕上任一显示位置自然可用行号列号来表字(行号列号均从0开始),但为方便,描述每一显示位置也可采用从0~1999指针值来表示,如第1行第2列的显示位置值是82,第23行第1列显示指针值是1814。即每一显示位置可以通过公式($80 * \text{行号} + \text{列号}$)来计算。

每个字符在显示存储区内占二个字节,一个字节是字符码,另一字节是显示形式。存储区中的显示形式代码规定如表1所示:

表1 不同显示形式的代码

Normal (low intensity)	07
Bold (high intensity)	0F
Underline	01
Blink	87
Reverse	70

既然了解了存储器中字符及其显示形式映射到屏幕上的道理,我们就可以通过程序来设定或修改显示形式。以下用Turbo Pascal语言为例作一示范。首先需要—个记录结构,用它能访问每个存储单元。见程序段G:

A)

Code Fragment G

```
type ScreenChar=record
    Value:Char; {actual character}
    Style:Byte; {text style}
end;
```

该段程序产生一个新的数据类型,构成一个字符的值和对应的字节内的显示形式。例如定义一个变量Spot,它表征类型ScreenChar,这样就把Spot归于二部分,即Spot.Value.和Spot.Style.对于单个字符,ScreenChar的二字节代表字符码和显示形式。

根据上面的介绍,如果我们选用的编程语言有一个较快地和有效地移动存储器的过程,那么就可方便地修改屏幕的显示形式。例如:Turbo Pascal的移动过程Move(Source, destination count),它可将存储器中特定数量的字节内容从源地址移向目标地址。作为一个例子,我们不妨用存储器写的方法来生成10个闪烁的星号“*”。注意,闪烁显示形式的代码是十六进制的87。有关程序结构是H、I、J和K所示:

A)

Code Fragment H

```
const Count=10;
var Asterisks:array[1..Count] of ScreenChar;
procedure CreateAsterisks;
begin
    for i:=1 to Count do
        begin
```

```

Asterisks[(2*i)-1]:=ord("*");
Asterisks[(2*i)]:=587;
end;
move(Asterisks,destination, TwiceCount);
end;

```

A)

Code Fragment I

```

const Count=10;
      TwiceCount=20;
var Asterisks:array[1..TwiceCount] of Byte;
procedure CreateAsterisks;
begin
  for i:=1 to Count do
  begin
    Asterisks[(2*i)-1]:=ord("*");
    Asterisks[(2*i)]:=587;
  end;
  move(Asterisks,destination, TwiceCount);
end

```

A)

Code Fragment J

```

const ScreenSize=2000;
ScreenBytes=4000;
LineSize=80;
type ScreenImage=array[1..TotalLines,1..LineSize] of ScreenChar;
      BlockImage=array[1..ScreenBytes] of ScreenChar;
var Lines:ScreenImage absolute $B000:0;
      Screen:BlockImage absolute $B000:0;

```

Code Fragment K

```

procedure CreateAsterisks;
begin
  for i:=1 to Count do
  begin
    Screen[origin+1].Style:=587;
    Screen[origin+1].Value:="*";
  end;

```

程序段H使用Pascal记录结构,十分易读。

注意:这里用来代表目标地址的变量并没有定义它。如果采用的语言没有记录只有数组,也能达到同样目的,但计算较为复杂些。这如程序段I所示。

不管使用那一种定义,首先要生成一系列星号,它们被夹在一行形式码中,然后移动这些已编码的星号串到目标区域。目标区域在哪里呢?回想一下前面提到的显示屏幕由2000个字符组成,它们被确定在相应的地址,这就需要能指定绝对地址位置。Turbo Pascal允许绝对地宣称变量,如在程序段J中那样。

在J段程序中为屏幕产生了二个不同的表示,两者都代表同样的存储器位置。Lines被设置成一个二维数组,而Screen是一个个屏幕字符数组,二个变量被定位在同样的地址,精确地重叠在屏幕存储器位置。我们可以使用一样的偏移量进入Screen数组,作为在Create Asterisks里的目标。确切地说,你能直接使用Screen数组排除移动,如K段程序中的表示。

这里我们所给的程序允许直接地向屏幕写,就好似屏幕是一个被展开的存储器一样。所设置的变量origin,它在1~2000之间取值。这样的表示法允许我们访问屏幕存储器,或者某个字节,或者整个行。通过Screen[(n* Linesize)+i]访问第n行第i位置。

对于更快速的显示,可通过建立一个存储器某部分的屏幕拷贝来实现。首先建立你所喜欢的显示形式,

然后执行一个块移动,来把数据立即传送到屏幕。

举例说,假设有一个字符串数组要显示,如通常的屏幕求助信息,命名这个数组为Help Lines并用其中元素填写一个屏幕拷贝,该拷贝命名为Screen Copy,然后移动该拷贝去显示,成为一个立即图象更新。我们再使用另外一个数组,名为Help Styles,它包含了Help Lines中每个字符串对应的显示形式码。上述过程如程序段L所示;

end;

Code Fragment L

procedure ShowHelp;

var ScreenCopy:BlockImage;

begin

fillchar(ScreenCopy,ScreenBytes,"");

for i:=1 to HelpSize do

for j:=1 to Length(HelpLines[i]) do

begin

ScreenCopy[i,j].Value:=HelpLines[i,j];

ScreenCopy[i,j].Style:=Helpstyles[i];

end;

move(ScreenCopy,Screen,ScreenBytes);

end;

这个过程允许我们在同一屏幕上实现不同的字符显示形式以及各种显示形式的复合,如闪烁的翻转字符。

通常我们可以自由地将二种或二种以上的显示形式加以复合,但并不是任何二种显示形式都能相结合的。表2展示了这些不同显示形式在字节里每个二进制位要求的置位情况。要获得可靠的显示形式,须按表中所示对显示字节置若干个1或0,标有×的位,意为0或1可任取。

表2 每种显示形式的置位情况

Bold	× × × × 1 × × 1
Blink	1 × × × × × × ×
Underline	× 0 0 0 × 0 0 1
Reverse	× 1 1 1 × 0 0 0

由于×是自由选取条件,那么产生同一种显示形式就存在好几种可采用的显示代码。如要生成加亮的字符,可使用的代码有:0F、5F、FF或者09。当然采取哪个代码要细心,如09既满足加亮字符显示形式,也满足字符下划线显示形式,选取09就会出现加亮字符再加下划线的复合显示形式。再如不能将翻转与下划线复合在一起,因为翻转显示形式的低位是0,而下划线的低位是1。

表3以十六进制和二进制二种形式列出了各种可以复合的显示形式,它们可作为在程序设计中的助记码。

表3 几种显示形式及结合的代码

Normal	00000111	07
Bold	00001111	0F
Underline	00000001	01
Blink	10000111	87
Reverse	01110000	70
Bold and Underline	00001001	09
Bold and Blink	10001111	8F
Underline and Blink	10000001	81
Blink and Reverse	11110000	F0
Bold Underline and Blink	10001001	89

一个文稿打印程序

北京大学计算机科学技术系 何宇才

许多报刊杂志都要求投稿人将文稿誊写在稿纸上,并规定每个汉字、单个英文字母、标点、符号、引号和括号均占一格,每两个英文单词之间空一格,等等。但是手抄难免不出错,尤其是一些外文字母的大小写常常不易区分。此外还有字迹是否工整的问题。为了满足一般稿件的誊写要求,笔者曾编写了一个文稿打印程序,自己觉得用起来比较得心应手。现借贵刊把它介绍给读者。

一、程序功能

该程序可以在20×20标准稿纸上打印,也可

以在普通打印纸或复印纸上打印。程序会问你使用那种纸。如果选择后者,程序会边打印边画上格子,并打印页号。用此种方法打印的稿件整齐、美观,但打印速度要慢一些。

原稿应预先输入到一个正文文件中,每行数不限。但标题部分最好不超过20个字。每当要打印一个文稿时,程序会提示你输入要打印的文件名。然后提示你准备好打印机,按任一键后开始打印。

程序从文件中逐行地读入正文,逐字地(或字符地)取字模(如果不是打印字模,还要进行

转换), 并送入行缓冲; 当行缓冲满时, 打印一行; 当一页满时, 送出该页纸并自动(或人工)装入一页纸, 等你按下任何一键后继续打印。在印完一个文稿后, 如要打印另一文稿, 则打入 'Y' 后, 程序又继续打印; 否则, 程序停止运行。

对一个文稿, 一般是全文打印。但有时因局部修改, 只要打印某一页或某几页时, 可以在提示: “请写入始页(默认1)和终值(默认999):” 后写入始页或终页。如界直接回车, 则使用默认值, 即全文打印。

为了适应多种形式的打印要求, 程序还允许在正文文件中使用3个编辑指示符: ! P, ! S和 ! E。! P(或! p)用于强制分段。一般地, 程序能识别以空格开头的自然段。但是如果一个自然段要顶格誊写, 则需在行首加指示符! P。例如, 要打印下面的行:

本段顶格誊写

则该行在输入时要写成:

! P本段顶格誊写

一般地, 一行印20个汉字或40个字符。但是如果有些行需密集打印, 则要在这些行的开始和末尾分别用指示符! S(或! s)和! E(或! e)标识。例如, 要打印下面的行:

本行密集打印

则在输入文稿时要写成:

! S本行密集打印

! E

密集打印时, 不识别自然段, 按行照印, 每行40个汉字或80个字符, 超过部分印在下一行。

这些指示符必须在行首, 可以独占一行, 也可以不独占一行。如文中一行开头正好是这些符号之一, 则应把! 双写。

二、编写技巧

该程序是在IBM 5550机上用笔者移植并汉化的Turbo Pascal 3.00B版本编写的, 全文附后。由于该Pascal系统有较强的串处理能力和调用BIOS中断及DOS中断的能力, 给编程带来很大方便。为了便于阅读, 程序中大多数标识符都使用了汉字。在不接收汉字标识符时均需改成英文字或汉语拼音。

该程序采用结构化、模块化设计。整个程序包含一个主程序和四个过程。主程序在设置初始信息后, 便进入了打印文件的主循环。初始信息主要包括两项:

(1) 设置打印参数; 每行点数和行走纸。这跟稿纸宽度、行距和打印机精度有关。这里设稿纸宽6.227英寸、行距0.458英寸, 打印机精度为每英寸180个点, 则每行点数为1121个, 行走纸为55单位(每单位1.5个点); 对非稿纸打印, 分三次走纸, 分别为16、16和23单位。程序中用控制序列Esc % 1 n1 n2和Esc % 5 n1 n2分别设置这两个参数, 其中n1和n2分别表示点数或走纸量的高位字节值和低位字节值。对不同型号打印机, 设置参数的控制序列也不同。程序中使用的控制序列只适用于IBM 5553打印机。

(2) 置打印缓冲初值。打印缓冲定义为有1121个元素的字符串数组, 每个字符串含3个字节, 对应24针打印机的一列。对稿纸打印, 只用一个行缓冲, 初始时全清零, 以后每打印一行都要清零; 对非稿纸打印, 要用三个缓冲, 其中两个用于画上、下格, 初始时置成适当值, 以后不再变动, 另一个和稿纸打印时类似, 但不都清零, 每56个点要画1个竖线。

过程“印一文件”具体打印一个文件。该过程从键盘接收一个正确的文件名, 有选择地接收起止页号, 接着进入了逐行逐字地打印的循环。在该循环中反复调用三个过程: 读一行, 印一字节或字, 印一行, 直到该文稿印完为止。过程“读一行”从正文文件中读一行正文到正文行中, 并处理有关编辑指示符、给出相应标记。过程“印一字节或字”并不真正打印一个字节或字, 只是根据正文行中第一个字节值决定取一字节或字的字模, 经转换后送入行缓冲。这里取字模采用IBM 5550机提供的BIOS 10H中断的10H号功能。在IBM PC机上如有24点阵打印字库, 则可根据汉字区位码直接取点阵送入行缓冲, 然后从正文行中删除该字节或字, 以便下次取字模。当行缓冲满时调用过程“印一行”。该过程打印一行, 如页满, 还要做换页处理。为了提高打印速度, 这里打印字符采用BIOS 17H中断的0号功能。

需要程序清单者, 请同作者或编辑部联系。

```

PROGRAM 印文档 (Input, Output, 源文件);
CONST
  字节 := ARRAY [0..7] OF Byte = ($80, $40, $20, $10, $5, $4, $2, $1);
  每页行数 := 20;
  每行格数 := 20;
VAR
  寄存器 : RECORD
    CASE Integer OF
      1 : (AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer);
      2 : (AL, AH, BL, BH, CL, CH, DL, DH : Byte)
    END;
  始页, 终页, 当前页, 当前行, 列计数, i, j, k, l, m, n, 间隔, 行下标 : Integer;
  字缓冲 : ARRAY [0..1, 1..3, 0..7, 0..1] OF Byte;
  上格, 下格, 行缓冲 : ARRAY [0..1120] OF String(3);
  源文件 : Text;
  源文件名, 正文行 : String [255];
  行点数, 走纸1, 走纸2 : String(5);
  回答, 字符 : Char;
  照印, 新段, 重读, 稿纸 : Boolean;
  字节数, 字节值 : Byte;

PROCEDURE 读一行;
BEGIN
  新段 := False;
  REPEAT
    Read(源文件, 正文行); REPEAT Read(源文件, 字符) UNTIL 字符 IN [#10, #26];
    重读 := (Length(正文行) = 2) AND (正文行[1] = '#') AND
      (Uppcase(正文行[2]) IN ['P', 'S', 'E']);
    IF (Length(正文行) > 1) AND (正文行[1] = '#') THEN
      CASE 正文行[2] OF
        'P', 'p': BEGIN 新段 := True; Delete(正文行, 1, 2) END;
        'S', 's': BEGIN 照印 := True; 间隔 := 2; Delete(正文行, 1, 2) END;
        'E', 'e': BEGIN 照印 := False; 间隔 := 16; Delete(正文行, 1, 2) END;
        'I' : Delete(正文行, 1, 1)
      END
    UNTIL (NOT 重读) OR (Eof(源文件));
    新段 := 新段 OR (Length(正文行) = 0) OR (正文行[1] = '#')
  END;
END;

PROCEDURE 印一字节或字;
BEGIN
  IF 正文行[1] = '#' THEN BEGIN 列计数 := 列计数 + 1; Delete(正文行, 1, 1) END
  ELSE
    WITH 寄存器 DO
      BEGIN
        ES := Seg(字缓冲); BX := Ofc(字缓冲);
        AX := $1000; CH := 0; CL := Ord(正文行[1]); 字节数 := 1;
        IF 正文行[1] > #127 THEN
          BEGIN
            IF (NOT 照印) AND Odd(列计数) THEN 列计数 := 列计数 + 1;
            CH := CL; CL := Ord(正文行[2]); 字节数 := 2
          END;
        Intr($10, 寄存器);
        行下标 := (12 * 间隔) * 列计数 + 间隔 * 字节数 DIV 2;

        IF (始页 <= 当前页) AND (当前页 <= 终页) THEN
          FOR i := 0 TO 字节数 - 1 DO
            FOR k := 0 TO 11 DO
              FOR j := 1 TO 3 DO
                BEGIN
                  字节值 := 0; m := k DIV 8; n := k MOD 8;
                  FOR l := 0 TO 7 DO
                    IF 字缓冲[i, j, l, m] AND 字节[n] <> 0
                      THEN 字节值 := 字节值 OR 字节[l];
                  行缓冲[行下标 + i * 12 + k][j] := Chr(字节值)
                END;
                Delete(正文行, 1, 字节数); 列计数 := 列计数 + 字节数
              END;
            END;
          END;

PROCEDURE 印一行;
BEGIN
  当前行 := (当前行 + 1) MOD 每页行数; 列计数 := 0;
  IF (始页 <= 当前页) AND (当前页 <= 终页) THEN
    WITH 寄存器 DO
      BEGIN
        Write(' ', 当前行); ClrEol;
        IF NOT 稿纸 THEN
          BEGIN
            FOR i := 0 TO 1120 DO
              FOR j := 1 TO 3 DO
                BEGIN DX:=0; AH:=0; AL:=Ord(上格[i][j]); Intr($17, 寄存器); END;
                Write(Lst, #13, 走纸1, #28)
              END;
            FOR i := 0 TO 1120 DO
              FOR j := 1 TO 3 DO
                BEGIN DX:=0; AH:=0; AL:=Ord(行缓冲[i][j]); Intr($17, 寄存器); END;
              IF NOT 稿纸 THEN
                BEGIN
                  Write(Lst, #13, 走纸1, #28);
                  FOR i := 0 TO 1120 DO
                    FOR j := 1 TO 3 DO
                      BEGIN DX:=0; AH:=0; AL:=Ord(下格[i][j]); Intr($17, 寄存器); END

```



```

END;
IF 当前行 > 0 THEN Write (Lst, #13, 走纸2, #28)
ELSE
  BEGIN
    当前页 := 当前页 + 1; Write (Lst, #13, #27+'S');
    IF (Length(正文行)<>0) OR NOT Eof(源文件) AND (当前页<=终页)
      THEN
        BEGIN
          GotoXY (5, 9); Write ('请换一页纸, 并按任何一键');
          REPEAT UNTIL KeyPressed;
          IF 稿纸 THEN Writeln (Lst, '':56, 当前页:4)
            ELSE Writeln (Lst, '第':56, 当前页:4, '页共' 页');
          Write (Lst, 行点数)
        END
      END;
    IF 稿纸 THEN 行缓冲 := 上格
    ELSE
      FOR i:=0 TO 1120 DO IF i MOD 56 <> 0 THEN 行缓冲[i] := #0+#0+#0
    END
    ELSE 当前页 := 当前页 + Ord (当前行 = 0)
  END;
PROCEDURE 印一文件;
BEGIN
  WITH 寄存器 DO
    REPEAT
      GotoXY (5, 6);
      Write ('请写入文件名([d:] [Y路径名] 文件名[.扩展名]): '); ClrEol;
      Readln (源文件名); 源文件名[Length(源文件名)+1] := #0;
      DS := Seg(字缓冲); DX := OfS(字缓冲)+1; AX := $1A00; MsDos(寄存器);
      DS := Seg(源文件名); DX := OfS(源文件名)+1; AX := $4E00; CX := $20;
      MsDos(寄存器);
      IF Odd(Flags) THEN Writeln(' 文件未找到. 请重输入, 或按^C停止')
      UNTIL NOT Odd(Flags);
      Assign (源文件, 源文件名); Reset (源文件);
      始页 := 1; 终页 := 999; Write (' 请写入始页(默认1)和终页(默认999): ');
      Readln (始页, 终页);
      Writeln ('':4, '文件在打印中...');
      GotoXY(5,9); Write('准备好打印机并按任何一键'); REPEAT UNTIL KeyPressed;
      当前页 := 1; 当前行 := 0; 列计数 := 0; 照印 := False; 间隔 := 16;
      IF 稿纸 THEN Writeln (Lst, '':56, 始页:4)
        ELSE Writeln (Lst, '第':56, 始页:4, '页共' 页');
      Write (Lst, 行点数);
      WHILE (NOT Eof (源文件)) AND (当前页 <= 终页) DO
        BEGIN
          读一行;
          IF (照印 OR 新股) AND (列计数 > 0) THEN 印一行;
          IF Length(正文行) = 0 THEN 印一行
          ELSE
            REPEAT
              印一字节或字;
              IF (照印 AND ((Length(正文行) = 0) OR (列计数 >= 每行格数+4) OR
                ((列计数 = 每行格数+1) AND (正文行[1] > #127)))) OR
                ((NOT 照印) AND ((列计数 >= 每行格数+2) OR
                ((列计数 = 每行格数+1) AND (正文行[1] > #127))))
                THEN 印一行
              UNTIL Length(正文行) = 0
            END;
          IF (列计数 > 0) OR (当前行 > 0)
            THEN FOR i := 当前行 + 1 TO 每页行数 DO 印一行
        END;
      BEGIN { 印文稿 }
        ClrScr;
        Writeln ('':24, 'IBM PC与IBM 5550正文稿纸打印程序');
        Writeln: Writeln ('':24, '北京大学计算机科学技术系 何宇才');
        Writeln: Write (' 是否打印在稿纸上(Y/N)? '); Read (Kbd, 字符);
        稿纸 := 字符 IN ['Y', 'y']; 行点数 := #27+'X1'+#4+#97;
        走纸1 := #27+'X5'+#0+#16; 走纸2 := #27+'X5'+#0+#23;
        FOR i := 0 TO 1120 DO
          IF (NOT 稿纸) AND (i MOD 56 = 0) THEN 行缓冲[i] := #17+#17+#17
            ELSE 行缓冲[i] := #0+#0+#0;
          上格 := 行缓冲; 下格 := 行缓冲;
          IF 稿纸 THEN 走纸2 := #27+'X5'+#0+#55
          ELSE
            FOR i := 0 TO 1120 DO
              IF (i MOD 56 = 0) OR (i MOD 4 = 0)
                THEN BEGIN 上格[i][1] := #1; 下格[i][3] := #64 END;
            回答 := 'Y';
            WHILE 回答 IN ['Y', 'y'] DO
              BEGIN
                印一文件: Writeln: Write (' 印另一文稿吗(Y/N)? '); Read (Kbd, 回答)
              END
            END. { 印文稿 }

```

用TRUE BASIC语言读取BASIC数据文件

北京航空航天大学 曹增强

摘 要

BASIC语言是应用广泛的高级语言之一。TRUE BASIC 是严格按照美国国家标准推出的一种结构化程序语言。它具有结构清晰、易读、易编程等特点,并具有很强的数据处理功能。本文通过分析两种语言数据文件各自的特点,提出了用TRUE BASIC 语言读取BASIC数据文件的方法。

一、BASIC数据文件的特点

BASIC数据文件有二种:顺序文件和随机文件。

顺序文件中的数据按每行送入的次序一个接着一个按顺序存放,每个数据按同样的方式读出。数据的存放形式是ASCII码。

随机文件的数据通常以二进制格式存贮。数据能随机存取,数据信息在被称作记录的单元中被存放和取出,每个记录都已被编了号码。随机文件的数据在被置入缓冲区时,数值必须用MKD\$,MKI\$,MKS\$转换为字符串,以字符串方式存放,在读出时再用CVI、CVS、和CVD转换为数值。

二、TRUE BASIC数据文件之特点

TRUE BASIC 数据文件有三种:正文文件、记录文件和字节文件。

记录文件是一“记录”或一些存贮单元的序列。文件中的记录都有唯一的数据,它既可以是字符串也可以是数字。文件的指针可用SET PO—INTER语句指向任一记录。数字和字符串都是以内部存贮形式写入的,所以记录文件不能直接在屏幕上显示。记录文件的记录必须设置长度。

正文文件是符号行的序列,每行必须由可显示的字符构成,按行按顺序存取。

字节文件不限定任何结构。只有符号串表达式的字节能够被写入到字节文件中,并且按照当

前文件所指向的位置顺序地放置于文件中,每写入一个字节指针向下移动一个字节。字节文件在读取时只能赋给符号串变量,无论何时从字节文件中读取数据都必须告诉TRUE BASIC 要读入多少字节来赋给字符串变量。

从以上可以看出:顺序文件和正文文件结构比较相近,随机文件和字节文件的结构比较相似。以下就用TRUE BASIC的正文文件和字节文件分别调用BASIC的顺序文件和随机文件以达到BASIC语言和TRUE BASIC语言数据共享之目的。

三、TRUE BASIC读取BASIC 数据文件

TRUE BASIC 的数据文件都具有扩展名TRU,因而TRUE BASIC将调用的BASIC数据文件也必须有扩展名TRU,否则TRUE BASIC将无法找到要调用的数据文件。

如下所示,程序一产生一顺序文件DAT. TRU。运行程序结果见下页。

程序二为TRUE BASIC 一正文文件调用DAT. TRU 数据文件。由于顺序文件和正文文件都是按行存取的,所以TRUE BASIC 每行要读的数据必须与BASIC 每行写入的数据相对应。另外TRUE BASIC 还需知道读哪个文件的数据量。

以下为程序二清单和运行结果。我们可以看出程序二读取的数据完全正确。

```

1 rem program1.bas---create a sequential file
10 open "dat.tru" for output as #1
20 input "name":n$
30 if n$="done" then close#1:end
40 input "department":d$
50 input "data hired":h$
60 write #1,n$,d$,h$
70 print:goto 20

```

程序一,basic 产生一顺序文件

```

run
name?zhang li
department?2
data hired?01/12/1978

```

```

name?wang hai
department?7
data hired?01/03/1967

```

```

name?li mei
department?6
data hired?03/05/1975

```

```

name?done
ok

```

```

rem program2.tru---call a basic's sequential file
input prompt "number of data that will be read?":number
input prompt "name of file?":nam$
open #1:nam$,organization text,access outin
for i=1 to number
    input #1:n$,d$,h$
    print n$,d$,h$
next i
close#1
end

```

程序二, true basic 调用basic 的顺序文件

```

run
number of data that will be read?3
name of file?dat
zhang li      2      01/12/1978
wang hai      7      01/03/1967
li mei        6      03/05/1975
ok

```

程序三为BASIC生产一随机文件FILE·TRU。以下为程序清单和运行结果。

程序四是一TRUE BASIC字节文件调用随机文件FILE·TRU。随机文件是以记录的形式存取的。所以TRUE BASIC也须以整个记录的形式来读取BASIC的随机文件。为了得到记录中每个字段的值,TRUE BASIC必须被告知每个字段的字节数。我们首先把整个记录赋给一字符串,然后从这个字符串中根据每个字段的字节数得到每个字段的值。为了调用方便,我们在产生BASIC的随机文件时把数值仍以字符串的形式输入,在TRUE BASIC读取时用

```

1 rem program3.bas-----create a random file
10 open "file.tru" as #1 len=32
20 filed#1,20 as n$,4 as a$,8 as p$
30 input "2-digit code?":code%
35 if code%<99 then close:end
40 input "name":n$
50 input "amount":amt$
60 input "phone":tel$:print
70 lset n$=x$
80 lset a$=amt$
90 lset p$=tel$
100 put #1,code%
110 goto 30

```

程序三,basic 产生一随机文件

```

run
2-digit code?01
name?cao zang qiang
amount? 1
phone?2017251

```

```

2-digit code?02
name?ruan hong
amount?1
phone?1230451

```

```

2-digit code?03
name?huang xiao ming
amount?2
phone?2348970

```

```

2-digit code?04
name?lou chun xiang
amount?1
phone?5678932

```

```

2-digit code?99
ok

```

函数VAL(A\$)转换为数值即可。另外由于VAL(A\$)的特殊要求我们在输入数值时不能采用指数形式。

程序四清单和运行结果见下页。我们可以看出TRUE BASIC读取的数据完全正确。

四、结束语

在TRUE BASIC读取BASIC数据文件应注意:

- 1 BASIC产生的数据文件要有扩展名TRU。
- 2 TRUE BASIC应正确使用VAL(A\$)函数。


```

rem program4.true--call basic's random file
dim numbyte(30)
input prompt"name of file?":name$
input prompt"length of record?":l1
input prompt"number of record?":l2
input prompt "number of record's text?":l3
mat numbyte=zer(l3)
print"number of text's bytes"
for i=1 to l3
input numbyte(i)
next i
open #1:name name$,organization byte
for i=1 to l2
read #1,bytes l1:b$
print b$[1:numbyte(i)]:
print val(b$[numbyte(i)+1:numbyte(i)+numbyte(2)]):
print b$[numbyte(i)+numbyte(2)+numbyte(3):l1]
next i
close#1
end

```

程序四,true basic 调用 basic的随机文件

```

run
name of file?file
length of record?32
number of record?4
number of record's text?3
number of text's bytes
?20
?4
?8
cao zeng qiang      1      2017251
ruan hong            1      1230451
huang xiao miny      2      2348970
lou chun xiang       1      5678932
ok

```

本文的所有程序均在IBM PC1×T机上运行通过。

参考文献

- (1) “用TRUE BASIC语言直接读DBASE III数据库文件” 李允俊,《小型计算机系统》,第9卷,第7期。
- (2) 《IBM PC BASIC程序设计语言》,同济大学出版社,1985年7月。
- (3) 《TRUE BASIC》,中科院软件研究所。
- (4) 《在“长城0520A”型计算机上实现FORTRAN和BASIC的数据文件的相互调用》,武警黄金部队黄金地质研究所,黄晓辉。

dBASE III 的彻底解密

重庆建筑工程学院 聂崇峡

dBASE III是用PROLOK程序加密的。除了dBASE III之外,还有许多程序都用PROLOK程序加密。因此,对dBASE III的彻底解密具有较普遍的意义。

笔者曾写过一篇文章(《微型计算机》1988/5),介绍PROLOK程序的加密方法。但对于不太熟悉系统的读者来讲,解密仍很麻烦。这里笔者提供一个BASIC程序,运行这个程序,即可对dBASE III进行解密。每个记录长度为8字节,DBASE.EXE共有16674个记录。

由BASIC程序生成的DBASE.DAT文件,从它的1d50H起到最后,就是dBASE III未经加密的程序。把它剪接出来,再换名成EXE为后缀的文件即可。剪接过程如下:

```

-n dbase.dat (找执行文件的头)
-1
-d 1e50 140 (这里包含PSP 100H字节,故为1D50+100H=1E50H)
0DA8: 1E50 4D 5A 9C 01 D6 00 9B 07-00 02 92 0C FF FF 3B 25
0DA8: 1E60 00 01 2B 75 FA 02 64 18-1C 00 00 00 02 00 2F 18
0DA8: 1E70 08 00 2F 18 0A 00 2F 18-18 00 2F 18 1A 00 2F 18
0DA8: 1E80 28 00 2F 18 2A 00 2F 18-38 00 2F 18 3A 00 2F 18
-r (寻找CS的值)
AX=0000 BX=0001 CX=CB00 DX=0000...

```

```

DS=0DA8 ES=0DA8 SS=0DA8 CS=0DA8...
0DA8: 0100
-h da8 1d5 ( 计算要写的文件的内存段起点cs+1d5h )
0F7D 0BD3
-rcs
CS 0DA8
: f7d
-h cb00 1d50 ( 计算长度cx-1d50h )
E850 ADB0
-rcx
CX CB00
: adb0
-n d1.dat ( 命名一个新文件, 若用EXE作后缀, 则写后不需再换名 )
-w
Writing 1ADB0 bytes
-q

```

对不熟悉上述操作的读者, 可以把160行的I=1改为I=1+938。这样, 不需要对输出文件进行剪接, 只要把它换成EXE后缀的文件。如

```
COPY DBASE.DAT D1.EXE
```

这时, d1.exe就可以启动了。它与DBASE.EXE作用一样。

PROLOK加密时, 如果被加密的某些数据组(8字节为一组)为全0, 则这时产生的加密信息正好是键值。键值共有8字节(其中有4字节是另外4字节的非值)。一般地, 执行文件的头的重定位区和文件的尾部, 通常有一些全0信息, 加密后形成8个字节一组的循环相同信息, 据此可以找出键值。如下为DBASE.EXE的键值。

```

-n dbase.exe
-l
-d 3b00 120
0DA8: 3B00 75 7D 8A 82 1C 68 E3 97-75 7D 8A 82 1C 68 E3 97
0DA8: 3B10 75 7D 8A 82 1C 68 E3 97-75 7D 8A 82 1C 68 E3 97
.....

```

把键值按6 3 2 7 8 1 4 5排列, 填入120行的DATA语句中。

对于dbase II, 这里已经填好了。但对于其它的用PROLOK程序加密的程序, 这个过程是不可缺少的。

键值是加密时随机产生的。

如果读者会寻找键值, 则本程序可以同一时期的用PROLOK加密的许多程序进行解密, 且几乎不需要了解它的加密过程。有兴趣的读者不妨试一试。

```

100 REM dBASE III decode program. nov 21. 1987
110 REM 6 3 2 7 8 1 4 5
120 DATA &h68,&h8a,&h7d,&he3,&h97,&h75,&h82,&h1c
130 FOR I=1 TO 8, READ K(1), NEXT I
140 OPEN "r", #1, "dbase.exe", 8, FIELD #1.8 AS ZY
150 OPEN "r", #2, "dbase.dat", 8, FIELD #2.8 AS YY
160 I=1, J=1, K=1
170 GET #1 I, I=I+1, MY="", PRINT I:
180 AHY=MIDY(ZY, 6, 1), ALY=MIDY(ZY, 3, 1)
190 BHY=MIDY(ZY, 2, 1), BLY=MIDY(ZY, 7, 1)
200 CHY=MIDY(ZY, 8, 1), CLY=MIDY(ZY, 1, 1)
210 DHY=MIDY(ZY, 4, 1), DLY=MIDY(ZY, 5, 1)
220 MY=MY+CHRY(ASC(AHY) XOR K(1))+CHRY(ASC(ALY) XOR K(2))
230 MY=MY+CHRY(ASC(BHY) XOR K(3))+CHRY(ASC(BLY) XOR K(4))
240 MY=MY+CHRY(ASC(CHY) XOR K(5))+CHRY(ASC(CLY) XOR K(6))
250 MY=MY+CHRY(ASC(DHY) XOR K(7))+CHRY(ASC(DLY) XOR K(8))
260 ISET YY=MY, PUT #2, J, J=J+1
270 IF EOP(1)=-1 THEN END ELSE 170

```

在Apple-Ⅱ显示屏上直接绘制高分辨率图形

兰州铁一院职工工学院 任伍林

摘 要

本文给出了 Apple-Ⅱ 微机在高分辨率图形显示状态下显示屏幕光点与内存地址间对应关系的计算公式,同时提供了一个仅通过键盘操作就能够随心所欲地在显示屏上绘图的基本程序。

我们知道,若在显示屏上进行分辨图形的绘制,一般要经过画草图,找出必要的座标点,根据草图编写绘图程序等步骤。而本文所给出的绘图程序则无需通过上述步骤,只要在键盘上操作就可以直接在显示屏上进行图形创作。下面就该程序所涉及到的有关问题叙述于后。

一、屏幕光点(像素)与RAM地址的对应关系

以Apple-Ⅱ显示屏高分辨率图形第2页(HGR2)为例。在HGR2状态下,显示屏全屏幕由53760个光点(162行×280列)组成,这些光点的亮与不亮即可形成图形。

根据RAM的每个内存单元的各位(高位除外)所贮信息是1或0,每一RAM单元可控制屏幕一行内连续7个光点的亮与不亮。若称屏幕上一行内的连续7个光点为一个显示单元,则每行的280个列光点就构成了40个列显示单元(0~39)。这样,全屏幕53760个光点即形成7680(192×40)个显示单元,它们与RAM中7680个内存单元对应。对于Apple-Ⅱ HGR2,用以控制全屏幕各显示单元的RAM地址从16384(\$4000)到24575(\$5FFF)。

屏幕显示单元与RAM地址间并不是一一对应关系,就屏幕行而言,其与RAM地址的对应划分为:

(1)先将0~191行分为三大区,每区间隔64行,每大区首行所对应的RAM地址差为40(十进制)。

大区号	每大区所含屏幕行	大区首行所对应的RAM地址
0	0 1 63	16384
1	64 127	16424
2	128 191	16464

(2)每一大区又分为8个小区,每小区间隔8行,每小区首行所对应的RAM地址差为128。如0号大区:

小区号	每小区所含屏幕行	小区首行所对应的RAM地址
0	0 7	16384
1	8 15	16512
2	16 23	16640
3	24 31	16768
4	32 39	16896
5	40 47	17024
6	48 55	17152
7	56 63	17280

(3) 每一小区内含有8个连续行, 每行所对应的RAM地址差为1024。如0号大区0号小区:

0号小区所含屏幕行	各行所对应的RAM地址
0	16384
1	17408
2	18432
3	19456
4	20480
5	21504
6	22528
7	23552

由上分析可见, 屏幕第Y行所对应的行地址HD可用下式计算:

$$HD = S + 40 * A + 128 * B + 1024 * C$$

这里: S: 表示首行地址, HGR2时,

$$S = 16384;$$

A = INT(Y/64): 确定Y行所在的大区号;

B = INT(Y/8): 确定Y行所在的小区号;

C = Y - 8 * B: 确定Y行位于B小区第几连续行;

(INT(X)为取整函数)

若位于第X列Y行处的光点用(X, Y)表 = (0 ≤ X ≤ 279, 0 ≤ Y ≤ 191), 则该点所处的列显示单元 L = INT(X/7)。这样光点(X, Y)所对应的RAM地址RD为:

$$RD = HD + L$$

另外, 由于每一RAM单元RD控制每一行的连续7个光点的亮与不亮, 故要准确控制点亮某光点(X, Y)还须使用下述指令来实现:

POKE RD, M

这里: M = 2^(X-L*7); RD, L同前。

比如, 要使位于X=139, Y=94处的光点发光, 通过计算得 L=19, RD=23995, M=2⁶=64 (相当于二进制1000000)。故用指令 POKE 23995, 64即可。

据此, 我们就可以通过键盘操作不断改变X, Y的值, 使RD、M的值相应发生变化, 并利用POKE RD, M指令, 从而达到控制亮点在全屏幕上自由移动的目的。

二、画线、点

通过屏幕上亮点的移动, 我们可确定出需画的起点座标(X₁, Y₁)及终点座标(X₂, Y₂)。为了能够快速画线, 在此我们不使用Applesoft BASIC语言的HPlot X₁, Y₁ TO X₂, Y₂指令, 而是使用了AppleSoft BASIC解释程序中专用于画线的汇编子程序。为了调用那些子程序, 还需使用如下几个6502汇编语言命令:

LDA #A, LDX #X, LDY #Y
及JSR \$M, RTS

用来实现画点和线。画点时\$A的取值为Y₁座标 (以2位十六进制数表示); \$X和\$Y分别取X₁座标 (以4位十六进制数表示)的低2位和高2位; \$M取值为\$F457。当画线时, \$A和\$X分别取X₂座标的低2位和高2位; \$Y取值为Y₂座标; \$M取值为\$F53A。这里\$457和\$F53A分别为画点或线的子程序入口地址。RTS用于返回主程序。

为方便使用, 这一过程作为汇编语言, 可用POKE指令将其写入某段内存内, 需要时用CALL指令调用。

三、程序使用说明

本文给出了用BASIC语言编写的直接在显示屏上绘制高分辨率图形的操作程序, 使用它就可通过键盘控制在显示屏上随心所欲地绘制出任何由点和线组成的图形。具体操作步骤为:

(1) RUM此程序, 屏幕显示:

"DRAWING NAME": (图名) 此时可键入你所画图形的名称。尔后, 屏幕将显示出绘图范围框, 并在屏幕中央显示一亮点。

(2) 通过I、K、J、L四键控制亮点上、下、左、右移动。

(3) 画线: a) 把亮点移至线段起点; b) 分别压下I键和Q键; c) 移亮点至线段终点并压下P键即完成画线。画点时完成a)、b)操作后直接压下P键即可。

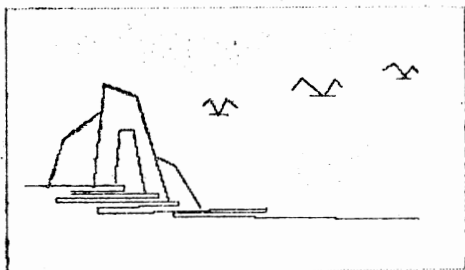
(4) 抹去某线段, 操作过程与画线同, 只是要将b改为压下0 (零) 键和Q键。

(5) 不存盘而结束绘图压Z键。

(6) 结束绘图并将图形存盘,同时压CTrL及A键。

(7) 再次显示图形可用指令: HGR2及BRUN(图形名)。

下图为笔者在屏幕上直接构思绘图后的拷贝。



四、源程序

LIST

```
10 HOME
20 INPUT " DRAWING NAME:";M1$
100 A2(1) = 169:A2(2) = 127:A2(3) = 133:A2(4) = 228
200 AS = 768:A1(1) = 169:A1(3) = 162
210 A1(5) = 160:A1(7) = 32:S1 = AS
300 HGR2 : HCOLOR= 6
305 HPL0T 0,0 TO 278,0 TO 278,191 TO 0,191 TO 0,0
310 X = 139:Y = 94:S = 16384:M = 8
320 RD = 17979
330 POKE RD,M
340 POKE 49168,0:P = PEEK (- 16384)
345 IF P = 48 THEN A2(2) = 123: GOSUB 1050:A = PEEK (49200): GOTO 340
346 IF P = 49 THEN A2(2) = 127: GOSUB 1050:A = PEEK (49200): GOTO 340
350 IF P = 81 THEN X1 = X:Y1 = Y: GOTO 370
360 IF P = 80 THEN X2 = X:Y2 = Y: GOTO 390
365 GOTO 400
370 GOSUB 2000: FOR I = 1 TO 20:AA = PEEK (49200): NEXT I
380 GOTO 400
390 GOSUB 2050
393 POKE AS,96
395 FOR I = 1 TO 20:AA = PEEK (49200): NEXT I
396 CALL 768: IF AS > 1022 THEN AS = 768
400 POKE 49168,0:P = PEEK (- 16384)
05 IF P=1 THEN590
410 IF X <= 0 OR X >= 278 THEN X = 139: GOTO 570
420 IF Y <= 0 OR Y >= 191 THEN Y = 94: GOTO 570
430 IF P = 73 THEN POKE RD,0:M = 2 ^ (X - L * 7):Y = Y - 1: GOTO 550
440 IF P = 74 THEN POKE RD,0:X = X - 1:M = 2 ^ (X - L * 7): GOTO 550
450 IF P = 76 THEN POKE RD,0:X = X + 1:M = 2 ^ (X - L * 7): GOTO 550
460 IF P = 75 THEN POKE RD,0:H = 2 ^ (X - L * 7):Y = Y + 1: GOTO 550
```

```
470 POKE 49168,0:P = PEEK (- 16384)
480 IF P < 128 THEN 470
490 IF P = 218 THEN TEXT : HOME : END
950 GOSUB 1000: POKE RD,M
570 GOTO 330
590 PRINT CHR$(4);"BSAVE";M1$;"A";16384;"L";8192
600 TEXT : HOME : END
1000 A = INT (Y / 64):B = INT (Y / 8)
1010 C = Y - 8 * B:L = INT (X / 7)
1020 HD = S + 40 * A + 128 * B + 1024 * C
1030 RD = HD + L
1040 RETURN
1050 FOR I = 1 TO 4
1055 POKE AS,A2(I)
1060 AS = AS + 1
1065 NEXT I
1075 RETURN
2000 N = X1: GOSUB 3000
2005 A1(6) = M1(4) * 16 + M1(3):A1(4) = M1(2) * 16 + M1(1)
2010 N = Y1: GOSUB 3000
2015 A1(2) = M1(2) * 16 + M1(1)
2020 A1(8) = 87:A1(9) = 244
2025 GOSUB 4000
2030 RETURN
2050 H = X2: GOSUB 3000
2055 A1(4) = M1(4) * 16 + M1(3):A1(2) = M1(2) * 16 + M1(1)
2060 N = Y2: GOSUB 3000
2065 A1(6) = M1(2) * 16 + M1(1)
2070 A1(8) = 58:A1(9) = 245
2075 GOSUB 4000
2080 RETURN
3000 I = 1:M1(1) = 0:M1(2) = 0:M1(3) = 0:M1(4) = 0
3010 IF N < 16 THEN M1(I) = N: GOTO 3035
3015 CN = INT (N / 16):M1(I) = N - 16 * CN
3020 IF CN < 15 THEN I = I + 1:M1(I) = CN: GOTO 3035
3030 I = I + 1:N = CN: GOTO 3015
3035 RETURN
4000 FOR I = 1 TO 9
4010 POKE AS,A1(I)
4020 AS = AS + 1
4030 NEXT I
4050 RETURN
```

浅谈数据库索引文件的使用

河北医学院第四医院计算机室 孙春秋

河北省卫生厅 张翼宗

内 容 提 要

数据库的索引文件结构主要是为了能够迅速的检索而设计的。在正常使用之外,我们还可以利用索引文件达到某种目的或取代一些命令以简化程序的编制或缩短执行时间。但如果使用不当,也有可能使问题变得复杂。本文对一些与其有关的命令作了归纳,并且根据作者的使用情况给出了几种索引文件的使用方法,同时也对与索引文件有关的问题作了讨论。

数据库的索引文件结构主要是为了能够迅速的检索而设计的。按索引关键字段检索时,索引文件能够迅速地找出一个满足关键字的记录。

必须使用索引文件的命令有:

1. FIND、SEEK

用于在关键字的数据查询;

2. SET RELATION TO

用于把现用数据同另一工作区中的数据库进行联接;

3. BROWSE命令中 ^ Home 状态的 SEEK功能

用于在BROWSE状态按关键字查询数据。这时要求同时打开按此键字索引的索引文件。

与索引文件有关的命令有:

1. TOTAL

当求现用文件数值字段的和且将结果记入第二的数据文件时,要求按关键字段进行索引或排序;

2. UPDATA

使用已标识源文件中的数据修改现用数据库文件中的数据记录,要求按关键字段进行索引或排序;

3. REPORT

在报告格式文件中已定义分组作小计,要求按关键字段进行索引或排序;

另外,要产生无重复项目的数据表时,也需要借助索引文件。具体实施方法是在进行索引以前执行一次SET UNIQUE ON。

在正常使用之外,我们可以利用索引文件达到某种目的或取代一些命令。

1. 利用索引文件改变数据库记录的显示顺序

有时候,我们需要以输入方向相反的顺序去处理数据,这就要求把后输的数据先显示出来。改变显示的一种方法是用CO BOTTOM 把记录指针指定到最后,然后以SKIP-1 反向追踪;另一种方法就是使用索引文件。此索引文件的关键字可取负的记录号[-RECNO()]。

2. 代替SORT命令进行排序处理

数据库,尤其是数据库II,利用SORT 命令进行排序要比索引处理慢得多,对翻译型数据

库干脆取消了此命令。因而用索引文件弥补这一缺陷是非常有效的。具体步骤如下:

USE数据库

INDEX ON关键字表达式TO索引文件

COPY TO新数据库

则新数据库就是已经排序的数据库。

这里的新数据库是以升序方向进行的,需要降序处理时可以再使用1的方法改变其方向。这种方法同时弥补了 SORT 命令不能以表达式作分类关键字的缺陷。

索引文件的关键字表达式必须是同一种类型,如果不同时应当进行一些转换,一般转换为字符型。但表达式中涉及到日期型字段时,最好用SET DATE ANSI把日期制为“年.月.日”的显示方式,否则产生的索引文件将不按日期的先后顺序进行排序。这是因为系统对日期的默认方式是“月/日/年”。

3. 代替INSERT命令插入记录

在日常工作中,我们经常需要在数据库的某些位置插入记录。虽然可以使用命令 INSERT 实施,但是用它插入数据时,记录要进行重写,当记录较多时就会花费大量时间。甚至有的数据库版本在插入记录的同时还会丢失尾部的记录,因而需要较多的记录插入时,可以采用索引文件的方法进行处理。使用合适的索引文件能够自动把新输入的记录调整到适当的位置。然后把索引后的文件拷贝到新的文件中,则新文件就是我们需求的文件。

虽然索引文件十分有用,但如使用不当也会出现一些问题。

1. 对批替换命令的影响

当索引文件打开后,用REPLACE命令替换关键字加范围或条件进行限制时,往往不能达到所希望的结果。这是因为计算机对一条记录进行处理后,系统就会自动把本记录调整到适当的位置。由于记录的移动,处理不能依次按原来顺序进行,而是相对于索引文件新的记录位置向下滚动。因而使用本命令只能修改当前记录,如要继续下去,就得反复调整记录指针位置,逐条进行。

甘肃省高考数据处理软件的设计及应用

甘肃省计算机中心 周仲宁 胡彦发 王子西 者建武

一、系统概述

本软件是在WANG VS/80计算机上开发研制的,用WANG COBOL和WANG BASIC高级语言编制程序,采用中西文混合输出。本软件具有数据处理、统计计算、中西文报表输出三大功能。

二、设计思想和程序模块

1. 设计思想:

- ①确保原始数据和处理结算的绝对正确。
- ②提高计算机的效率,确保按时完成任务。
- ③合理管理数据,确保处理的顺利进行。
- ④软件要便于阅读、调试、修改和扩充。
- ⑤软件操作要简单方便。

2. 程序模块:

本软件约有80多个程序模块,主要的有:

录入模块:录入考生各单科的成绩和考生简况。

数据处理模块:建立考生考号文件,对数据进行校验、修改、删除、查询、缺考检查处理,特殊情况处理等。

统计计算模块:考生分数的统计计算,各类

分档表的统计计算等。

汉字处理模块:处理有关反字的信息,打印输出表格的汉字部份等。

输出模块:打印西文部份,如成绩通知单,简明情况表,剪贴条等。

三、应用情况

本软件自1986年开发研制成功以来,已经连续三年顺利地完成了甘肃省高考数据处理的工作,按时保质完成了任务,得到了甘肃省招生办公室的充分肯定和高度评价。本软件的开发研制成功,为高等学校招生的数据处理工作提供了现代化手段,对提高招生工作质量,缩短招生工作的周期,科学分析考生成绩起积极作用。

参考文献

1. 卢开澄:《组合数学算法与分析》下册,清华大学出版社 1983年。
2. 朱三元编译:《软件工程指南》,上海翻译出版社,1985年。
3. D.E.Knuth: The Art of Computer Programming, vol.3, Addison-Wesley Publishing Company, 1973.

2. 对BROWSE编辑状态的影响

使用索引文件后,如果用BROWSE命令对关键字段进行修改,也会由于记录指针的重新定位不能按原来的顺序依次指向刚刚修改过的记录的下一条记录。对数据库Ⅱ解决此矛盾的方法是BROWSE命令同NOFOLLOW连用。

3. 对运行速度和磁盘空间的影响

索引文件不改变数据的物理存储地址。当数据较多,并且索引文件的逻辑记录位置与其物理地址差异很大,且对数据库进行序列操作时,例如使用条件FOR,打开的索引文件将明显减慢其运行速度;输入数据的时间也将延长。并且,索引文件要占用磁盘空间。当关键字较多时,有时索引文件要比数据库还要大。

因而,什么时候使用索引文件较合适是值得考虑的。可用可不用时,最好不用。

再者,数据库结构的修改,也会对索引文件产生影响。如果修改的数据库结构涉及了索引文件的关键字段,即使仅仅改变了它的长度,也一定要重新以字段进行索引。而且最好不用REINDEX命令,这是因为在这种情况下使用REINDEX处理过索引文件被打开后,很容易出现“记录没有被索引”的错误。

虽然一般数据库教科书中没有专门的介绍,事实上,汉字索引的排列顺序是以区位码为基准的。但在一级字库中的汉字是按汉语拼音排序的,并且这些字都是常用字,因而,很容易使人产生汉字是以拼音排序的感觉。了解了这些之后,我们就不难理解为什么在索引文件中,“腾”(TENC)字的位置要排在“中”(ZHONG)字之后。

用计算机解Reve之谜

江西省轻工业设计院 程贵平

在教科书中,常常用汉诺塔(Tower of hanoi)作为递归过程的例子。当盘子增多时,移动次数急剧增加,为了减少移动的次数,英格兰学者Dudeney于1907年提出了汉诺塔的一个推广,称为Reve之谜,其描述如下:

有四条凳子和八块大小均不相同的乳酪,把四条凳子排成一行,再把所有乳酪叠放在一端的凳子上,大块乳酪只能放在比它小的乳酪下面。要求你把所有乳酪都移到另一端的凳子上,每次只能移动一块乳酪,并且任何时候不得把大块乳酪放在小块乳酪的上面。

Reve之谜和汉诺塔相似,只不过是多了一个缓冲区而已。因此,在解决Reve之前,让我们先看一下汉诺塔的移动方式。

设 n 为乳酪块数, s_1, s_2, s_3 为凳子的编号,则汉诺塔算法描述为:

```

procedure hanoi (n,s1,s2,s3: integer)
begin {move n cheeses from s1 to s2 using s3}
  if n=1 then
    writeln (s1, '→', s2)
  else
    begin
      call hanoi (n-1, s1, s3, s2);
      writeln (s1, '→', s2);
      call hanoi (n-1, s3, s2, s1);
    end;
end

```

因此,汉诺塔的移动过程描述为:把上面的 $n-1$ 块乳酪从 s_1 移到 s_3 上,再把最大的那块乳酪从 s_1 移到 s_2 上。然后把 s_3 上的 $n-1$ 块乳酪移到 s_2 上。设汉诺塔的移动次数为 $h(n)$,则:

$$\begin{aligned}
 h(n) &= 2h(n-1) + 1 \\
 &= 2^2h(n-2) + 2 + 1 \\
 &= 2^k h(n-k) + 2^{k-1} + \dots + 2 + 1
 \end{aligned}$$

由于 $h(1)=1$,当 $k=n-1$ 时有:

$$h(n) = 2^{n-1} + 2^{n-2} + \dots + 2 + 1 = 2^n - 1$$

同理,Reve之谜的移动是把 n 块乳酪中的 $n-f(n)$ 块乳酪移到一个凳子上,然后对剩下的 $f(n)$ 块乳酪作汉诺塔移动。再 $n-f(n)$ 块乳酪移回。

设Reve之谜的乳酪块数为 n ,剩下 $f(n)$ 块乳酪进行汉诺塔移动,凳子的编号为: s_1, s_2, s_3, s_4 ,则其算法为:

```

Procedure Reve (n, s1, s2, s3, s4: integer);
begin {move n cheeses from s1 to s2 using s3 and s4}
  if n=1 then
    writeln(s1, '→', s2)
  else
    begin
      Reve (n-f(n), s1, s4, s3, s2);
      hanoi (f(n), s1, s2, s3);
      Reve (n-f(n), s4, s2, s3, s1);
    end;
end;

```

设Reve之谜的移动次数为 $R(n)$,则

$$R(n) = 2R(n-f(n)) + 2f(n) - 1$$

$$(1 \leq f(n) < n)$$

式中 $R(n)$ 的值与 n 和 $f(n)$ 有关,当 n 为定值时, $f(n)$ 为多少时, $R(n)$ 才会最小呢?由于 $R(n)$ 中有两个变量,不能由以上式子推出 $f(n)$ 同 n 的关系函数,在此我们只有借用计算机来求 $R(n)$ 最小时的 $F(n)$ 值,其程序如下:

```

5 REM 求R(N)最小值时的F(n)值
10 DIM R(30), F(30)
20 R(1)=1: F(1)=1
30 INPUT M
40 FOR N=2 TO M
45 R(N)=2^N
50 FOR J=1 TO N-1
60 X=2*R(N-J)+2^J-1
70 IF X<R(N) THEN R(N)=X: F(N)=J
80 NEXT J
90 NEXT N
95 LPRINT " N", "R(N)", "F(N)"
100 FOR N=1 TO M
110 LPRINT N, R(N), F(N)
120 NEXT N
130 END

```

由 $F(n)$ 的运行结果可知: $F(n)$ 值的改变在 n 为如下值时: 1, 3, 6, 10, 15... $n_{i-1}+i$ ($i \in \mathbb{N}$)

因此,由以上数列的公式 $n_i = n_{i-1} + i$ 可推出 n_i 的表示式为 $n_i = i(i+1)/2$ 即 $n = i(iH)/2$. 由此可求出 i 的表示式为: $i = (\sqrt{8n+1}-1)/2$ 通过对 i 求整便得到了 $F(n)$ 的值, $F(n) = \lfloor (\sqrt{8n+1}-1)/2 \rfloor$ 其FORTRAN的表示式为:

$F(n) = \text{ifix}(0.5 * (\text{sqrt}(1+8 * \text{float}(n)) - 1))$

同样可求出 $R(n)$ 的表示式, 由运行结果有:

$$R(2) = R(1) + 2^{F(1)} \quad \dots\dots ①$$

$$R(3) = R(2) + 2^{F(2)} \quad \dots\dots ②$$

$$R(4) = R(3) + 2^{F(3)} \quad \dots\dots ③$$

\vdots

$$R(n) = R(n-1) + 2^{F(n-1)} \dots\dots n-1$$

以上 $n-1$ 式相加得:

$$R(n) = R(1) + 2^{F(1)} + 2^{F(2)} + 2^{F(3)} + \dots\dots + 2^{F(n-1)}$$

由于 $F(1) = F(2)$, $F(3) = F(4) = F(5) \dots\dots$

$$\therefore R(n) = 1 + 2 * 2^{F(1)} + 3 * 2^{F(3)} + 4 * 2^{F(6)} + \dots + \left(n - \frac{F(n)(F(n)+1)}{2} \right) * 2^{F(n)} \quad \dots\dots ④$$

$$2R(n) = 2 + 2 * 2^{F(3)} + 3 * 2^{F(6)} + \dots + 2 \left(n - \frac{F(n)(F(n)+1)}{2} \right) * 2^{F(n)} \quad \dots\dots ⑤$$

由④-⑤得:

$$-R(n) = -1 + 2^{F(1)} + 2^{F(3)} + 2^{F(6)} + \dots - F(n) * 2^{F(n)} - \left(n - \frac{F(n)(F(n)-1)}{2} \right) * 2^{F(n)}$$

$$\therefore R(n) = \left(F(n) + n - \frac{F(n)(F(n)+1)}{2} - 1 \right) * 2^{F(n)} + 1$$

其中 $F(n) = \lfloor (\sqrt{8n+1}-1)/2 \rfloor$

笔者在IBM-5550机上, 用FORTRAN语言实现了Reve之谜的移动, 并验证了运行结果. 对于五个或五个以上凳子的移动方式其原理同Reve之谜的移动方式, 在此不多加讨论.

Reve之谜的程序清单如下:

```

c program reve.for
c 乳酪块数(M),移动次数(II),凳子(S1,S2,S3,S4)
c Move M Cheeses From S1 to S2 Using S3 and S4
  integer s1,s2,s3,s4,m,ii
  read(*,*) m
  ii=0
  call reve(m,1,2,3,4,ii)
  write(*,*) 'm',m,'ii',ii
end
c subroutine of reve
  subroutine reve(m,s1,s2,s3,s4,ii)
  integer s1,s2,s3,s4,ii,n,m
  if (m.eq.1) then
    write(*,*) s1,' -> ',s2
    ii=ii+1
  else
    n=ifix(0.5*(sqrt(1+8*float(m))-1))
    call reve(m-n,s1,s4,s3,s2,ii)
    call han(n,s1,s2,s3,ii)
    call reve(m-n,s4,s2,s3,s1,ii)
  end if
end if
return
end
c subroutine of han
  subroutine han(n,s1,s2,s3,ii)
  integer n,s1,s2,s3,ii
  if (n.eq.1) then
    write(*,*) s1,' -> ',s2
    ii=ii+1
  else
    call han(n-1,s1,s3,s2,ii)
    write(*,*) s1,' -> ',s2
    ii=ii+1
    call han(n-1,s3,s2,s1,ii)
  end if
  return
end

```

FORTRAN与MACRO之间的最佳数据交换

核工业部西南物理研究院 叶高英 邓慧忱

关键字: 数据交换, 共用存储区

一、数据交换环境

随着计算机应用领域的不断扩大, 同一计算机上运行的不同程序设计语言程序之间进行数据交换已引起越来越多的注意和讨论。文〔1〕讨论了在 VAX-11/750 计算机上运行的一个实时数据采集处理系统 (DAS) 与其它并行系统程序在实时环境下的大批量数据交换问题。在用 FORTRAN 语言书写的 DAS 软件系统与用宏汇编 MACRO 书写的低级设备驱动程序之间进行数据传输时, 我们利用 VAX/VMS 操作系统提供的 PS (Program-Section) 程序 (数据) 块定义工具, 实现了两种不同语言程序之间的最佳数据交换即, 以直接内存存取方式传送数据。本文下面详细介绍这一方法

二、程序实现

一般说来, 两种不同语言交换数据可以通过存取共用外部文件实现, 只要文件结构、格式及数据的组织能够被两种语言各以某种方式检索、读、写即可, 但这种交换方式需要大量的排队 I/O 操作。在实时环境中, 由于既要交换大批量数据又要求快速时间响应, 因而应尽可能避免此种数据传输形式。

另一种交换形式是带数组参数的过程 (或子程序) 调用。文〔2〕给出了在 IBM/PC 微型机上可行的这类数据交换例程。应当指出对于复杂软件设计而言, 象例程中所介绍的那种自我修改地址的方法是一种冒险, 它给程序正确性测试和验证设置了难以逾越的障碍, 犯了大忌。此外, 由于编译系统为数组分配的存储地址被动态地修改了, 这意味着在 FORTRAN 程序中调用汇编子程序之前对数组进行的任何存取操作、数组中已记录的所有数据都面临被遗弃的危险。因此, 汇编程序首先必须对已有数据进行转储, 既耗费时间, 且双倍使用存储空间造成浪费。

我们在 VAX 计算机上设计的 FORTRAN 与 MACRO 程序之间交换数据程序完全克服了上述不足之处。将 FORTRAN 中定义的公共数据区与 MACRO 中用 PS 结构组织的数据块用同名实现存储共享, 从而以直接内存存取方式进行数据交换。下面给出一个简单而又完整的程序实例。

```

PROGRAM TRANS
Test the transformation of data between
FORTRAN and MACRO program fragments

INTEGER COUNT, MCA(4096)
COMMON /TRANSDATA/MCA,COUNT

DO I=1, 4096
  MCA(I) = I
END DO
COUNT = 2000
CALL SEND
TYPE *, MCA
END

MACRO:

.PSECT TRANSDATA, NOEXE, RD, WRT, LONG
.M: .BLKL 4096
C: .BLKL 1

.PSECT SEND, NOWRT, EXE
.ENTRY SEND, "m(R7, R3, R4, R9)
MOVAL M, R7
MOVL C, R3
LOOP: DECL R3
MOVL (R7), R4
MULL #2, R4
MOVL R4, (R7)+
CMPL R3, #-1
BCTR LOOP
RET
.END

```

很显然, MACRO 中定义的数据段在结构上与 FORTRAN 中共用数据区 /TRANSDATA/ 完全一致, 而同名又能保证其公用性。

应当指出, 对于某个特定 FORTRAN 编译程序, 在编译处理共用数据区时需指明一些特征信息供尔后地址分配时识别使用, 这是需要在汇编中模拟给出的。我们这里指明的非执行、可读、可写及长字存取并不能保证与前者完全一

APPLE IIe 微机体感诱发电位检测系统软件设计

第三军医大学野战外科研究所

彭利安 刘连生 陈恒胜 杨诗球 谭正中

指导 李 朴

人体脊髓, 皮质体感诱发电位 (SPEP, CSEP 总称 SEP), 能够比较直接而客观地反映中枢神经、外周神经的功能状态, 并对神经系统损伤病变的程度和部位作出判断。目前国内有条件研究或初步应用的单位, 多数是从国外引进设备, 研究与应用起来, 一时难以得心应手, 均感难度大, 再因价格昂贵, 一般用户不敢问津, 为能适应国情, 普及基层, 一般能力和条件的医院科室买得起, 用得上, 我们在为 APPLE IIe 微机研制了简单、新颖、经济、实用的硬件接口, 构成系统 (图1) 的前提下, 自行设计了功能较全的软件程序。为使用户能够人机对话, 选用 APPLESOFT 会话式语言; 为提高机器工作速度, 选用宏汇编; 为解决信息量大, 内存容

量小, 显示分辨率低, 运算速度受限制的矛盾, 充分考虑编程技巧, 达到工作要求, 用于临床监护和早期诊断, 取得了良好效果。

一、提取信号

小至零点零几微伏的脊髓、皮质体感诱发电位 (SPEP, CSEP), 深埋在比其大得多的各种心电、肌电等自发电位及外界电场干扰之中, 一般的模拟或数字滤波难以分离出来, 根据它总是出现在刺激后一定时间的规律, 严格地以刺激时刻为准, 对每次所得的取样信息逐点叠加平均, 增大信号, 削弱噪声取出诱发反应, 公式为:

$$S_k = \frac{\sum_{I=1}^N Y_k}{N} \dots\dots\dots (1)$$

理论上讲, 叠加N次, 信号振幅增加N倍, 噪声幅度则只能增加 \sqrt{N} 倍, 实践证明, 叠加次数足够多, 效果越理想 (图2)。

试叠1000次, 亦即刺激1000次, 每刺激1000次取1000个时间序列, 则有百万个数据, 每字节放一个信息量, 最后做总的叠加, 需占1兆字节, 仅64K的本机, 既容纳不下, 也无法处理。

选用边采样边叠加, 在本次采样等待时间, 将上次采样值与前I次值累加, 多叠65000次, 也只需3个字节放一个数据, 1K序列值仅占3K字

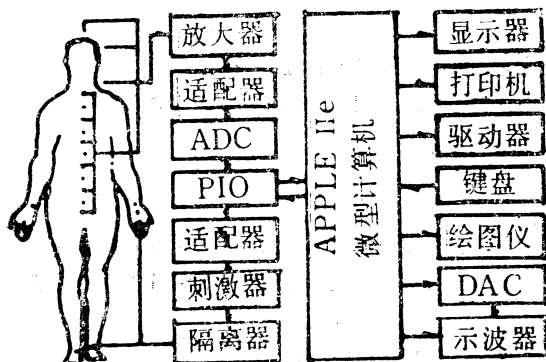


图1 SPEP 检查硬件框图

因此, 因此在link遍有可能报告“类型冲突”警告信息, 不过对于我们介绍的使用要求并无影响, 故可不予理睬。

参考文献

〔1〕叶高英、邓慧忱:《并行数据采集的集中

管理》, 研究报告, 1988, (将在《研究年报》上发表)

〔2〕万钟济、宋知用:《高级语言与汇编语言间的数据传送》,《微型计算机应用》, 第八卷 第一期, 1987.1,

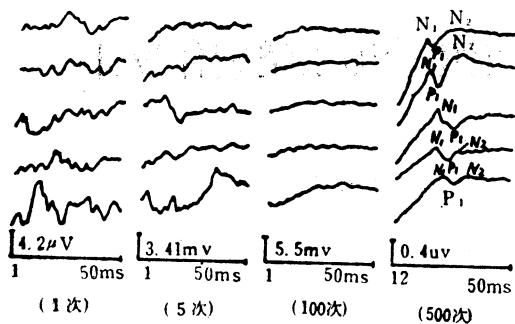


图2 不同叠加平均次数的脊髓诱发电位

节, 平均搬移后少至1K, 压缩了数据量, 节省了内存, 一台似乎胜任不了的机器, 慨然胜之有剩。

叠加次数取2的整数次方N, 平均时将叠加后时间序列值统统右移N位, 既简单又省时, 虽然限制了选择次数的随意性, 但已满足需要, 提高了工作速度。

二、分析计算

波峰幅值、峰值潜伏时为SEP重要检查指标, 峰值的大小有无或者是说异常或消失, 反映了神经通路的损伤病变产生了障碍, 先曾设计了一种全程式自动分析计算法。

试以刺激伪迹左边延迟空间基线值数点平均, 作为随机应变、基于实际的活动基线, 亦称基线窗口, 视为分析计算的参考点, 即:

$$S_b = \frac{\sum_{I=1}^N Y_1 + \sum_{I=1}^N Y_2 + \sum_{I=1}^N Y_3 + \cdots + \sum_{I=1}^N Y_j}{j} \quad (2)$$

由此得出人体体表CSEP波幅算法:

若 $S_k > S_{k-1}$; $S_k \geq S_{k+1}$;

又 $S_k \geq S_{k+2N}$; $S_k \geq S_{k+3N}$; $S_k \geq S_{k+4N}$; $S_k \geq S_{k+5N}$; $S_k \geq S_{k+MN}$;

还 $S_k > S_{k-2N}$; $S_k > S_{k-3N}$; $S_k > S_{k-4N}$; $S_k > S_{k-5N}$; $S_k > S_{k-MN}$;

则 $S'_k = (S_k - S_b) \times \frac{10}{256} \quad (3)$

或 若 $S_k < S_{k-1}$; $S_k \leq S_{k+1}$;

又 $S_k \leq S_{k+2N}$; $S_k \leq S_{k+3N}$; $S_k \leq S_{k+4N}$; $S_k \leq S_{k+5N}$; $S_k \leq S_{k+MN}$;

还 $S_k < S_{k-2N}$; $S_k < S_{k-3N}$; $S_k < S_{k-4N}$; $S_k < S_{k-5N}$; $|S_k - S_{k-MN}| \geq Y$

则 $S'_k = (S_k - S_b) * \frac{10}{256} \quad (6)$

式中 $N=1$ 或 5 ; $M=3$ 或 8 ; $Y=2$ 或 4 。

凡满足算式条件值统算为峰, 然后根据波峰出现的时刻, 峰值的大小、波形的形状, 即可认定峰否, 相应其值自然所知。

峰值潜伏时的算法为:

令 $M = I * \Delta T$

如 $M > D_T + F + R$

则 $N_i L = I * \Delta T - D_T$

$P_i L = I * \Delta T - D_T \quad (7)$

按上各式逐点计算, 故要多费机时, 截取典型段, 只对产生反应的有用区间分析, 运算时间减半有余。

人体体表SPEP甚微, 前述算法偶有不变, 又曾设计了既适用CSEP又适合SPEP的键移造型游标自动分析法;

$$N_1 = (Y_{ni} - Y_+) * 10/256$$

$$N_i \sim P_i = (Y_{pi} - Y_{ni}) * 10/256$$

$$P_i \sim N_i = (Y_{ni(i+1)} - Y_{pi}) * 10/256 \quad (8)$$

或;

$$P_1 = (Y_{p1} - Y_+) * 10/256$$

$$P_i \sim N_i = (Y_{ni} - Y_{pi}) * 10/256$$

$$N_i \sim P_i = (Y_{p(i+1)} - Y_{ni}) * 10/256$$

P_1, N_1 —正、负首峰; Y_{ni}, Y_{pi} —峰点座标值; $N_i \sim P_i, P_i \sim N_i$ —峰—峰幅度值。峰值

潜伏时间(7)式。

主程序设计流程见图3,两法首段相似,后段各异,仅述后者。程序启动后,键盘回答被试者姓名、检查位置、条件参数,待系统同步,即刻采样叠加平均,一次性展示图象,一旦电位图形始段下方出现造型游标“1”,指击“→”、“←”键,将其移至欲设的参考点位置,置“↓”键,瞬间闪出以被选点为中心的十字架“+”,参考点即被建立,同时在曲线的上方、十字架的右边再现与前同状的造型峰点游标,照例击用“→”、“←”键,将其移至的确为峰的适中点,重按“↓”键,以此反复,峰点上下依次闪现波峰序名,箭头指示,峰值峰峰值,峰值潜伏时等结果。打印全屏幕内容并将图形数据结果存盘,按“↑”键;不计算不打印只存盘,按“M”键;清除重来者,按空格键;强行停机,并行按下CT-Re键。

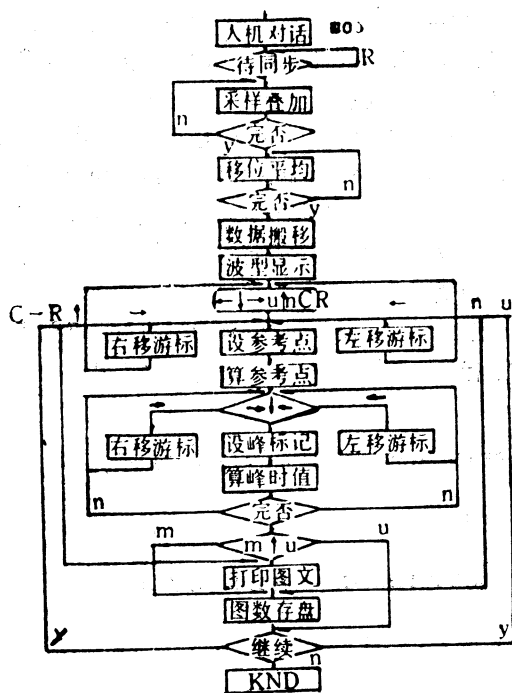


图3 主程序流程图

三、结果处理

屏幕显示:不同节段部位的CSEP、SPEP有不同形状的电位波形,数据结果,一次性展示

于屏幕,给人以直观感,更有利于分析。分辨率低的本机,难以示出图像全程,压缩四倍取全程,二倍压缩取首段、中段或末段。

打印记录:检查位置、电位图形、分析结果等全屏内容压缩六倍打印成窄条状检查报告单,供医务人员分析参考。

图形数据存档:为把实验或检查电位图形原状地保存下来,便于事后的统一分析,或是作为一种原始资料,以便事后查证,将其存入磁盘,长期保存,形成档案,何时查阅,可自动检索、再分析、再计算。

病历档案存贮:键盘键入病人病历,可自动修改、打印、存盘,随时可见。

四、程序功能

为使本机使用灵活方便,功能齐全,试用两种方案编制了二种适合动物SEP、人体SEP(CSEP、SPEP)及其它同步与非同步等多种信号图形的采样叠加、平均、分析、计算、打印、存档、检索、再分析、再计算以及档案文件的书写、存贮、检索等40多种功能30多个程序。

单板机控制

炭黑袋滤系统简介

该系统是吉林省计算技术研究所与长春市煤气公司共同研制的,是用新技术改造旧设备成功的一例。在炭黑生产行业中属国内首创。该系统采用软、硬件结合实现控制,有体积小、集成度高、稳定可靠、操作简便、便于维修及修改参数灵活等特点。具有较大的经济效益和推广价值。

该成果荣获省级科学技术进步四等奖。现已在大连、鞍山等地推广使用,效果很好。

简短的西文提示汉化程序

南京市3801信箱47分箱 赵超

西文软件的汉化主要完成两步工作,一是使西文软件能在汉字操作系统中正常运行,二是将西文提示改为汉字提示。前者主要通过改善系统调用的标准化程度和兼容性、使用统一的内部码等方法解决。最近上海某单位已开发出一套“嵌入式”汉字操作系统,据称,任何西文软件不需作任何修改均可在该系统下运行,给西文软件的汉化提供了一条捷径。后一问题的解决不存在技术上的难度,但工作量较大,通常的做法是用汉化了了的DEBUG调出欲汉化的可执行文件,再寻找西文提示行,逐行将西文提示修改为汉字提示。这个过程涉及地址计算,DEBUG内部命令的使用、文件的读、写等操作,普通计算机用户很难进行。笔者用BASIC语言编制了一个短程序,利用这个程序普通录入人员借助一本字典即可对西文软件的提示符进行汉化。

这个程序由可显示字符串搜索和汉字字符串代换两大部分组成。10~180行是搜索部分,这部分程序逐字扫描目标文件,若连续出现两个可显示ASCII码字符则认为是提示字串的起始点。在一串连续可显示字串后若出现一个不可显示ASCII码字符则认为是字串的终止点。起点和

终点的位置分别存放在数组BPOINT和EPOINT中。程序的20行定义有效可显示字符;40行读入欲汉化程序文件名;50行打开文件并将记录长度定义为1字节。60行对文件名有效性进行检查;100行顺序读出一个字节数据;110行判别是否为可显字符;120行判别并记录字串的终止点;150行处理连续可显字串;160行判别第二可显示串、确认字串起始点并记录该点;运行至此就完成了一串可显示字符的识别和记录。由于这部分程序运行需较长时间,180行设置了搜索完成音响报警功能,用以招徕操作员进行下面的代换工作。

190~390行是代换部分,程序逐行显示西文提示,操作员在下面键入对应的中文指示即能完成该行字符的代换。对于一些无意义的字串,操作员应键入回车跳过。190行显示共搜索出字串的行数;220行键入欲修改行的行号;240行进行有效性检查;270行显示原西文提示字符;300行键入相应的中文提示并进行有效性检查;350行写入串文提示。

笔者利用该程序对DOS3.3中的一些西文程序进行汉化效果很好。(源程序清单附后)

```

10 DIM BPOINT(3000),EPOINT(3000)
20 ALLCHARS=".,:?!abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNPOQRSTUVWXYZ0123456789"
30 KEY OFF:CLS
40 LOCATE 2,1:INPUT "文件名: ";FILENAME$:LOCATE 3,1:PRINT "
50 OPEN FILENAME$ AS 1 LEN=1:FIELD #1,1 AS ENGCHAR$
60 IF LOF(1)=0 THEN CLOSE:BEEP:KILL FILENAME$:LOCATE 3,1:PRINT "未找到! 请再键
入: ";GOTO 40
70 FL=LOF(1):I=0:N=1
80 WHILE I<FL
90 LOCATE 3,1:PRINT "正在处理第" I "字节"
100 I=I+1:GET #1,I
110 IF INSTR(ALLCHAR$,ENGCHAR$)<>0 THEN 140
120 IF MARK$="ENG" THEN EPOINT(N-1)=I-1:MARK$="NOT ENG"
130 GOTO 170
140 IF MARK$="ENG" THEN 170
150 I=I+1:GET #1,I
160 IF INSTR(ALLCHAR$,ENGCHAR$)<>0 THEN BPOINT(N)=I-1:N=N+1:MARK$="ENG"
170 WEND
180 IF INKEY$<>" " THEN BEEP:LOCATE 3,1:PRINT "请敲空格键开始汉化! ";GOTO 180
190 LOCATE 1,1:PRINT "行号 = 1 ---"N-1
200 FOR I=1 TO N-1

```

浅介下拉式菜单

深圳深汉电子技术有限公司技术部 刘宇菁

1. 概述:

所谓菜单,就是一系统选择项,用户从中进行选择。以前的菜单是使用数字或选择项的第一个字母来选取选择项,而且这种菜单有时会清掉整个屏幕的内容,我们称这种菜单为老式菜单。当前最流行的菜单有两种:突现式菜单和下拉式菜单,通常突现式菜单只有一个层次,即突现式菜单内的选择项并无子菜单,实际上,突现式菜单是下拉式菜单的一个特殊子集。因此,本文介绍下拉式菜单。

所谓下拉式菜单(Palldown-menu),简单的说,即此种菜单内的选择项还可以是另一个菜单,另一方面,屏幕上可以允许数个下拉式菜单同时有作用。一般在设计显示时,里层的下拉式菜单位于外层的下拉式菜单之下,故取名为“下拉式”,但只要具有上述特性,而不论显示位置如何,即里层的菜单不一定位于外层的菜单之下,都应当是下拉式菜单。下拉式菜单与老式菜单相比,有以下一些优点:

- ①色彩鲜艳、丰富。
- ②用户不需打入所选的选择项,而可以移动亮光柱进行选择。

③可以在屏幕任何地方出现,不会清去其它部分的信息。

④可以不进行选择就退出此层菜单,而老式菜单则做不到这点。

因此,在程序中使用下拉式菜单,不但丰富了程序的内容,而且还极大地方便使用者。正因如此,下拉式菜单已成为当前国际流行软件的主流之一。

2. 设计思想

欲建立和显示下拉式菜单,则程序必须能直接控制屏幕,因此,这种程序就与硬件和操作系统有关。

在启动一个下拉式菜单时,此菜单会暂时盖掉屏幕上的一些信息;而在使用者做了选择之后,屏幕应当回复原先的状态。

从以上的过程可以看出,要编写一个含下拉式菜单的程序,必须有以下几个方面的内容:

- ①建立一个菜单的框架结构,即此菜单在何处显示,菜单的大小,菜单的条目和条目数,菜单的热键(使用此键快速进行选择),菜单是否正在显示等。
- ②保存此菜单要盖掉的屏幕上那部份信息。

```

210 LOCATE 3,1:PRINT "
220 LOCATE 3,1:INPUT "请输入行号: ",F
230 IF F=0 THEN 250
240 IF F> N-1 OR F<0 THEN BEEP:LOCATE 2,1:PRINT "行号错!"
      ":GOTO 210:ELSE I=F
250 LOCATE 2,1:PRINT "请汉化第" I "行
      :LOCATE 4,1:FOR J=1 TO 80*4:PRINT " ":NEXT
260 LOCATE 4,3:FOR J=BPOINT(1) TO EPOINT(1)
270 GET #1,J:PRINT ENGCHARS:
280 NEXT J:PRINT
290
300 LOCATE 5,1:INPUT CHINAS:IF LEN(CHINAS)>(EPOINT(1)-BPOINT(1))+1 THEN BEEP:LOC
ATE 3,1:PRINT "太长! 请再键入 ":LOCATE 5,1:PRINT "
      ":GOTO 300
310 L=1:CHL=LEN(CHINAS):LOCATE 2,1:PRINT "键入汉字长" CHL ":" 字节, 从"EPOINT(1):"到
      "BPOINT(1)" 字节存入 "
320 FOR K=BPOINT(1) TO EPOINT(1)
330 IF CHL=0 THEN 360
340 IF K>BPOINT(1)+CHL THEN CHMID$=" " ELSE CHMID$=MID$(CHINAS,L,1)
350 PRINT #1,CHMID$:PUT #1,K:L=L+1
360 NEXT K
370 NEXT I
380 BEEP:LOCATE 3,1:PRINT "汉化全部完成!"
390 END
      ":CLOSE
  
```

- ③把菜单显示在设定好的区域。
- ④获取键盘信号, 以得到使用者的选择。
- ⑤在使用者从菜单选择某个条目之后, 程序必须将屏幕还原。

下面, 我们详细阐述下拉式菜单的上述几项内容。为了简明起见, 本文暂介绍使用字符方式的下拉式菜单。

3. 框架结构

使用下拉式菜单的程序, 在执行过程中, 必须不时地引用某个下拉式菜单的各种信息。提供下拉式菜单视框的最佳方法是建立能存放所有与下拉式菜单有关的信息的一个结构数组, 定义方法如下:

Struct menu—frame

```
{
    int startx, starty; 菜单左上角的座标
    int endx, endy; 菜单右下角的座标
    char ** menu; 指向菜单条目的指针
    char * keys; 指向热键的指针
    unsigned char * P; 指向保存此菜单要
        盖掉区域的信息的缓冲区的首指针
    int border; 是否需要画边框
    int count; 菜单的条目数
    int active; 是否正在显示
} menu—frames (MAX—FRAME);
```

至此, 我们已经定义好下拉式菜单的数据结构。在使用下拉式菜单之前, 必须先初始化一个视框。为节省篇幅, 见实例程序中的 make—menu函数。

4. 保存屏幕某部分信息

要想将屏幕上的资料储存起来, 必须读取屏幕上每一个位置目前的值, 并将它储存起来。

显示在屏幕上的字元是存放在显示卡中的 RAM 内, 此块 RAM 的起始地址在单色显示卡中为 B0000000H, 而在彩色显示卡中为 B8000000H。每个要显示在屏幕上的字元需占此块 RAM 的两个字节, 第一个字节存放显示字元的 ASCII 码, 第二个字节存放显示字元的屏幕属性。单色显示卡的屏幕属性只有几种:

正常的黑底白字 属性字节的值为7
反白字元 属性字节的值为70H
划底线的字元 属性字节的值为1

彩色显示卡的属性字节如下设定:

位	意 义
0	蓝色前景
1	绿色前景
2	红色前景
3	高亮度
4	蓝色背景
5	绿色背景
6	红色背景
7	闪烁字元

在字符方式下, 一般的模式为25行×80列。上述说明, 主要用于存取屏幕和设计字元的属性。

一般说来, 控制屏幕的方法有三种: 第一种是使用DOS的功能调用, 第二种方式是使用BIOS的功能调用, 第三种方法是直接存取屏幕缓冲区, 比较而言, 使用第三种方法可以制作任何类型的下拉式菜单。因此, 本文的程序是直接使用屏幕缓冲区。

本节功能见实例程序中的 make—menu—video函数。

5. 显示下拉式菜单

此功能见实例程序中的 pulldown函数。此函数是本章的主要内容, 请读者细细领会。

6. 恢复原屏幕信息

在使用者从菜单中选取某个条目之后, 程序必须将屏幕还原。见实例程序中的 restore—meau—video函数。

7. 读取使用者的回应

使用者在进行选择时, 一般有两种方式: 第一种是按一个热键而快速选取某条目; 第二种方式是使用“箭号键(arrow key)”或空格键将亮光柱移到某一条目上, 然后再按ENTER键。

因此, 在上述过程中, 既可能使用一般字母键, 也可能使用特殊键。由于C语言中的 get—char() 之类的函数只能传回字元代码, 故无法使用。本文为此设计了专门的函数 get—key

获取回应功能见实例程序中的 get—resp函数。

8. 实例程序 (因程序太长, 篇幅所限实例程序略去, 需要者, 请同作者或本编辑部联系)。

“怡红夜宴图”验证程序设计

东南大学计算机系 黄大海

在《红楼梦》第六十三回“寿怡红群芳开夜宴”中，曹雪芹描写了怡红夜宴时行酒令的热闹场面。为了追求无限丘壑的艺术效果，曹雪芹没有明确指出参加夜宴的总人数以及参加者的座次。书中只给出了行酒令时掷骰子的过程：晴雯开掷，五点到宝钗；宝钗十六点到探春；探春十九点到李纨；李纨未掷，由下家黛玉十八点到湘云；湘云上家黛玉，下家宝玉，湘云九点到麝月；麝月十九点到香菱，香菱六点到黛玉；黛玉二十点到袭人。（选自1982年新版红楼梦）。此外，从书中还可以看出，总人数不少于十四人，不多于二十五人。

由于各种版本对夜宴中掷骰子的点数记载有出入，一百多年来“怡红夜宴图”成了红学家们争论的焦点之一。俞平伯、周绍良等许多专家对此做过研究，并绘制出各自的怡红夜宴图。现在要求编写一个程序来验证《红楼梦》各种版本记载掷骰子点数是否正确，并对正确版本给出参加者总数以及座位次序。

刚拿到这个问题，或许我们一时想不出如何去编写程序。那么不妨来模拟一下怡红夜宴的掷骰子过程。先剪一些红方块代表座位，剪一些白方格写上已知参加者的名字。然后取若干红方块围成一圈，并把“晴雯”放在一个红方格上。以后根据书中记载的点数，顺某一方方向依次去安置每一个已知参加者。这里有一点需要说明，在按点数去数座位时，可以包含投骰子者，也可从他的下家数起。假定我们采用前一种方法。

在取十四个座位时，我们发现放下“湘云”后，“黛玉”不是她的上家，下家却成了“晴雯”。在取十五个座位时，“宝钗”又和“探春”争抢一个座位。只有在取十六个座位时，才没有两个人争抢一个座位或“黛玉”一人占几个座位的现象。如图1。

通过对出错情况和正确答案的比较分析，我们可以看出：在假定了数座位方法和总人数前题

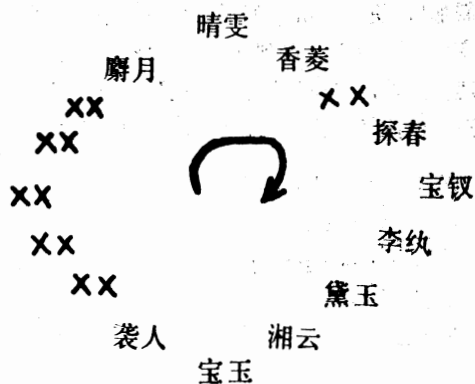


图1 怡红夜宴图

下，如果按某版本掷骰子点数去安排座次，既没有争座位又没人多占座位，那么该版本记载是正确的。否则至少可以认为，在数座位方法、总人数和版本记载三者中有一个是错的。如果通过穷举总人数和数座位方法仍无正确答案，则可断定该版本记载有错。

下面的任务是编写程序。我们可以用数组单元代表座位，它们初值为0，表示无人坐。给每个已知参加者一个编号，例如从1到10。然后把数组看成是一个环形缓冲区，按掷骰子次序和点数向数组内填数字。

不过这里还有一个难点，就是如何表达“李纨下家是黛玉”和“湘云上家是黛玉、下家是宝玉”最简单方法就是，在放置完参加者后，查看“李纨”、“黛玉”、“湘云”和“宝玉”四人是否依次排坐在一起，这一点从上述问题中推出。

显然，判别某次试排是否成功的条件是：一，存在某个单元要求填入不同的数，也就是说，在填某单元时发现单元值已不是零。由于每个人只被数骰子点数点到一次，因此不会出现同一个数填入同一单元两次的现象。二，在放置后发现“李纨”、“黛玉”、“湘云”和“宝玉”

四人不是依次排在一起。

下面给出相应程序,该程序输入数据如表1所示。它由数组N、C组成,分别填写掷骰子的点数和到达者的编号。在验证不同版本时只要按版本记载的点数修改N数组即可。不过,到达湘云的点数需加1,使“李纨不掷下家黛玉掷十八点到湘云”变为“李纨十九点到湘云”以减少不必要的处理。此外,“李纨”、“黛玉”、“湘云”和“宝玉”依次编号为9, 5, 6, 7, 以利于编写程序。

```
10 DIM N(9), C(9), B(25)
20 FOR I=1 TO 9
30 READ N(I), C(I)
40 NEXT I
50 FOR I=1 TO 9 STEP -1
60 FOR M=14 TO 25
70 FOR L=0 TO M-1
80 B(I)=0
90 NEXT L
100 X=L
110 FOR I=1 TO 9
120 Y=(X+N(I)-L) MOD M
130 IF B(Y)<>0 GOTO 270
140 B(Y)=C(I)
150 IF C(I)=4 THEN Y=X
160 NEXT I
170 FOR I=5 TO 7
180 Y=(Y+1) MOD M
185 IF I=7 AND B(Y)=0 GOTO 200
190 IF B(Y)<>I GOTO 270
200 NEXT I
210 PRINT "The number of participants: ";M
220 FOR I=0 TO M-1
230 PRINT B(I); " ";
240 NEXT I
250 PRINT
260 GOTO 300
270 NEXT M
280 NEXT L
290 PRINT "This version is wrong!"
300 END
310 DATA 0,1,5,2,16,3,19,4,19,6,9,8,19,9,6,5,20,10
```

通过上述程序的设计过程可以看出,当我们拿到一个问题而又不能立即想出编程方法时,不妨先建立一个模拟环境,并试着去解决问题,从多

表1

下标	N	C	姓 名
1	0	1	晴 雯
2	5	2	宝 钗
3	16	3	探 春
4	19	4	李 纨
5	19*	6	湘 云
6	9	8	麝 月
7	19	9	香 菱
8	6	5	黛 玉
9	20	10	袭 人

* 填点数加1

次的失败和成功中找出规律,悟出编程的方法。作者建议,在采用模拟方法时应注意以下问题。

第一,在模拟环境下寻找解决问题方法时,要尽量摆脱编程意识的纠缠,特别是不要把数据结构硬塞到模拟环境中,否则会束缚我们的思路。

第二,对每一次试解问题步骤和结果都应加以记录、分析。试解失败时,要找到陷入困境的原因并设法避免它。成功时,要找出带有规律性的东西,并在模拟环境下反复验证它。这样才能悟出解决问题的方法。

第三,在把人的解法转变为计算机算法时,要注意人与计算机各自的特点,选择一种既能较好地表达模拟环境又有利于计算机处理的数据结构。并根据该结构进一步细化人的解法,使之过渡为计算机算法。

让您的pc机奏出悦耳的曲子

南京无线电仪器厂CAD室 刘 康

摘要: 本文从实用的角度,介绍了如何在PC机上奏出美妙动听的音乐的两种方法,对初学者尤为适宜。

音乐是如此深邃地激动人心,以致成为一门老幼咸宜、人人都爱的艺术。事实上,当我们在应用编程的时候,尤其是在调出成品软件的时候,倘若能在程序的恰当位置加上一个或几个精妙的音乐程序段,真可谓锦上添花也!那么怎样才能编制音乐程序以及应该具备哪些基本知识才行呢?

一、基本知识

1. 乐理部分

- 1 — C
- 2 — D
- 3 — E
- 4 — F
- 5 — G
- 6 — A
- 7 — B

“#”和“+”，表示升高半音；

“—”，表示降低半音；

“.”，附点音符，表示再延长紧前演奏音符一半的时间；

“Q”，表示休止符，停顿一段时间，音长由实际乐谱确定。

注意：E→F和B→C只升高了半音，故无E⁺、F⁻、B*等存在。

2. 控制命令部分

①音长控制符：Ln.

(1 ≤ n ≤ 64)

例：L4，表示四分音符，演奏时间为一拍；

L8，表示八分音符，演奏时间为1/2(半)拍。

②速度控制符：ML、MN、MS.

ML，置演奏速度为慢速；

MN，置演奏速度为中速，即为Ln的7/8；

MS，置演奏速度为快速。

③演奏方式设定：MB、MF.

MB，称为后台演奏，当用其演奏时，可以边执行别的语句边奏乐，单独演奏时应不多于32个音符，当然加上一些技术也可以用于多于32个音符的演奏；

MF，称为前台演奏，当用其演奏时，别的语句暂停执行，是计算机的默认方式。

④演奏音符：Nn.

例：N32，表示“5”；

N33，表示“5*”；

N20，表示“5”；

N41，表示“3”

(常用演奏音符见附。)

⑤音程控制符：On.

O₂，表示低音音程；

O₃，表示中音音程；

O₄，表示高音音程。

⑥休止控制符：Pn.

例：P4，表示休止一拍；

P2，表示休止二拍；

P8，表示休止半拍。

⑦速度调节符：Tn.

(32 ≤ n ≤ 255)

n值愈大，速度愈快；

计算机默认值是：12Q.

说明：在实际编程中，所有的控制命令符，不见得个个都要出现，在一段程序中，也不一定都同时出现。

二、两种编程方法

当我们编程的时候，常常希望方法快捷、简便、易记、从实用的角度本文介绍两种方法。

(一) 音程+音值+音长

(二) 音长+演奏值

下面举例说明：(注①表示第一种方法，⑥表示第二种方法)

1. 《是你给我爱》

| 1 3 3 1 | 35 23 30 | 1331 | 23 5 -- |

① O₃C4E4E4C4E8G8D8E8E4P4C4E4
E4C4D8E8O2G2.

⑥ L4N25L4N29L4N29L4N25L8N29 L8
N32L8N27L8N29L4N29P4L4N25 L4
N29L4N29L4N25L8N27L8N29L2
N20L4N20

2. 《彩云飞》

| 3 5 6 | 1 2 1 6 5 | 65 32 12 | 3-- |

① O3C4G4A4O4C4D16C16O3A8 G4A8
G8E8D8C8D8E2.

⑥ L4N29L432L4N34L4N37L16N39 L16
N37L8N34L4N32L8N34L8N32L8
N29L8N27L8N25L8N27L2N29L4N29

NAG应用软件生成系统简介

南京航空学院 徐宝文 伍俊明 李红霞 袁明 黄雅茜

一、基本思想

传统软件开发一般要经过需求定义与分析、设计、编码、测试几个阶段，在通常的软件开发中，大部分时间都花在软件的设计、编码与测试上。因此，提高软件生产率就必须尽可能减少设计、编码与测试工作量，最好能直接根据用户需求产生相应的目标代码。NAG 就是为此目的研制的一个系统，用于管理软件的自动生成。研制这一系统主要基于以下认识：在各种管理软件系统中，它们的宏观结构是大致相同的，绝大多数功能的处理方式也是类似的，如屏幕格式设计与处理、数据的录入、查询、统计与删除方法、数据有效性校验、报表生成等。我们可以把它们存在的诸多共性规格化、通用化，使之成为各种管理软件的基本框架，再把对具体系统的“个性”的处理与之结合起来，就可以研制一个具体管理软件系统。这就是NAG系统的研制依据，本系统可根据用户需求直接生成所需目标程序，而且目

标程序一经生成不需其它任何辅助手段就可单独使用。目标程序由两部分组成：其一是基本部分，这是任何具体管理软件系统都需要的；其二是专用部分，它特定于具体应用需求。

二、设计要点与系统结构

由于dBASE 是国内比较流行的关系数据库管理系统，考虑到系统的实用性，我们选用dBASE 语言作为目标语言（在第一版中生成的是dBASE I 程序，第二版是dBASE II 程序），为方便起见，实现工具也选用了dBASE。系统在执行时，首先接收用户需求，进行分析处理，形成各种辅助数据库文件，再利用它们生成可执行目标程序以及供目标程序执行使用的数据库文件（结构）。在生成过程中，目标程序中的语句是作为数据库记录进行处理的并存放在数据库文件中，每形成一个功能模块就转换成可执行dBASE 程序文件。

本系统的主要研制目标是服务于稍懂或不懂

另外，“>”和“<”也能改变音程高低。

例：| 356 | 1̣ 2̣ 1̣ 6̣ | 5 | 65 32 12 | 3- - |

可以写成：O3C4G4A4>G4D16C16<A8
G4A8G8E8D8C8D8E2.

建议：采用第一种方法编程，好处有四。

1. 看着歌谱的同时，就能编；
2. 不用查表，不宜出错；
3. 便于检查；
4. 键入的字符少，省机时。

本人编了一个小程序供读者尝试。

《海霞》组曲之一《童年》

10 PLAY "MF□MN□T12Q□03"

20 PLAY "G8>C8D8E8D2<A8>D8
D8C8<G2A8>D8D8C8<A8G8>D4 C8
<A8G8A8G2"

30 END

修改演奏速度，反复演奏，您一定会觉得趣

味无穷。

(常用演奏音符对照表见下)

演奏音符对照表

演奏值	音符	演奏值	音符
N 20	5	N 35	6*
N 21	5*	N 36	7
N 22	6	N 37	1
N 23	6*	N 38	1*
N 24	7	N 39	2
N 25	1	N 40	2*
N 26	1*	N 41	3
N 27	2	N 42	4
N 28	2*	N 43	4*
N 29	3	N 44	5
N 30	4		
N 31	4*		
N 32	5		
N 33	5*		
N 34	6		

计算机的用户,因此,它向用户提供的接口是交互式的,这种方式的优点是,用户在使用系统之前几乎不需进行培训,容易掌握,出错率低。

本系统主要由需求定义分析、生成、辅助数据库三大部分组成,如图1所示。需求定义分析部分用于接收并分析用户需求、生成字段信息库等数据文件,并向驱动程序提供有关信息控制系统其余部分的执行。生成部分根据需求定义分析部分生成的各种辅助数据库文件及其它有关信息,在驱动程序控制下生成目标程序及其它辅助数据库文件(包括供目标程序运行时管理使用的数据库文件的结构。在这些部分生成或使用的(辅助)数据库文件主要有字段信息库、条件库、统计库、报表库、系统库、用户库等,其中,有的是在需求定义分析部分生成在生成部分(及目标程序)维护与使用的,有的是在生成部分定义并在该部分(及目标程序)中维护与使用的,有的是在生成部分定义由目标程序使用的,在生成部分建立的数据库有的只有文件结构,具体内容由目标程序录入。

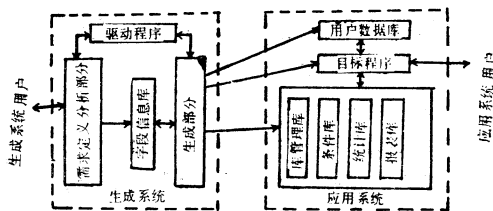


图1 NAG系统结构

三、系统功能

由本系统生成的软件具有库管理、转贮、插入、查询、删除、修改、统计、报表生成等各种功能。库管理的目的是对库文件名、口令等进行登记、修改、核对、撤除等管理;转贮用于实施各数据库文件的数据传送;插入模块用于插入记录;查询有多种形式,既可单字段查询、部分字段查询又可查询整个记录;删除记录时,可由用户进行确认;修改操作分全屏幕修改、单项手工修改、单项自动修改等三种;统计的式样可由用户自定;报表格式可由用户预先确定或在使用时临时选定。

在进行查询、修改、统计、制表、转贮、删除时都要根据一定的条件进行。在本系统中,所生成软件的条件设置有两种方法,一种是快速条件,主要借助索引进行,另一种可称为普通条件,通过会话由系统自动组合,用户使用时不需要掌握dBASE的表达式用法。如,用户可以在会话方式下提出“找出武汉大学毕业的50岁以下的男性教授”的查询要求,系统会据此组合相应的条件表达式。这种方式的优点是既方便于一般用户,又具有较高的灵活性。

无论是dBASE I 还是 dBASE II 都对每个数据库文件允许使用的字段数有限制(分别为32与128个)。因此当某应用系统中用户字段数超过这个界限时,一般要用多个数据库文件,但这样既会影响速度,又会影响外存利用率。为了克服此不足,本系统在遇到用户字段数较多的情况时,把有关用户字段组合到一个dBASE字段中。这样,一个数据库文件就可以存放数百个乃至上千个用户字段,只要每个记录的总字节数不超过原有限制。另一方面,有些用户字段的字符数可能要超过dBASE规定的254个,(这时对dBASE II 虽然可以用辅助(备忘)文件,但这有诸多不变。)对这种情况,系统把一个用户字段分存到多个dBASE字段中。这两种扩充都是由生成系统自动进行的,用户无需知道实现细节。

四、结 语

应用软件生成系统的一个突出优点是,系统面向应用,面向用户,具有很高的非过程性。借助应用软件生成系统,用户只需向系统提供他要干什么,而不必关心怎么干。比起手工研制的软件,用该系统研制的软件具有整齐划一、用户接口标准统一、使用方便、维护容易等优点。目前我们已经取得了初步成功,正考虑在此基础上作进一步完善。

这项研究得到了南京航空学院青年科学基金的资助,在此谨致谢意。

用BASIC程序恢复误删文件的方法

广州南方医院 于 桦 郭松亮

在使用IBM—PC/XT及兼容机的过程中,用户可能会由于操作失误把不该删除的文件删除了,只要没有新的存盘动作发生,恢复误删的文件是完全可能的。用户可以采用DEBUG程序的“L、D、E、W”命令恢复误删的文件。这种方法比较直观,但要涉及到磁盘目录的地址、驱动器号、扇区号等内容,对于不熟悉DEBUG程序和DOS内部结构的用户来说是十分困难的。是否可以用高级语言的程序直接从软件上恢复误删的文件,答案是肯定的。关键是要解决直接读取目录区问题,BASIC语言不具备直接读/写目录区的功能,汇编语言却能够很容易地实现,二者结合起来即可方便地恢复误删的文件。

1. DOS的内部结构和目录区

双面的5 1/4吋软盘,每面各有40道,每道有9个扇区,每个扇区有512个字节。盘上除文件外,还记录着引导程序,两份文件定位表和根目录。图1是DOS磁盘的映象图:

引导程序
文件定位表1
文件定位表2
根目录
数据区

图1: DOS磁盘的映象图

DOS在0道的5~8扇区建立一个目录区,长度是3.5K,可存放112个目录项,一个文件仅占一个目录项。每个目录项占32个字节,它们的内容及含义如图2所示:

0	7	8	A	B	C	F
文件名	类型名	属性	未用			
未用	时间	日期	簇号	文件长度		
10	15	16	17	18	19	1A 1B 1C 1F

图2: 文件目录项

其中文件名的第一个字节表示目录项的状态:

00H 从未用过的目录项
E5H 已删除文件的目录项
2EH 子目录标记项
其它 文件名的首字符

当文件被删除时,只是在文件名的第一个字节做了“E5H”标记,而文件的内容仍留在磁盘上。只要没有新的存盘动作发生,把目录中该文件名的删除标记用文件名第一个字母的ASCII码替代就可达到恢复误删文件的目的。

2. 用BASIC程序恢复误删文件的方法

①调用汇编子程序把目录区的内容读入内存。语句20、30、40、110。

利用POKE语句把机器语言的代码装入内存,由CALL语句调用子程序,把指定驱动器目录的内容读入内存。

②显示磁盘的全部文件名。语句210~270。

将目录以每屏10个文件名显示出来,并在已删除文件名的第一个字符用“*”作标记,用户可选择C——继续显示,S——停止显示,0...9——选择本屏某个已删除的文件作为要恢复的文件。

③恢复误删文件的目录项并写回磁盘。语句280~310。

修改要恢复文件名的目录项的第一个字节,将其删除标记“E5H”用该文件名第一个字母的ASCII码代替,并调用汇编子程序把恢复后目录区的内容写回磁盘。

本程序简单实用,运行程序时只需输入驱动器号,就可一次恢复多个已删除的文件。该程序在IBM—PC/XT微机上通过,操作系统为CC—DOS2.10或CC—DOS2.13A。

程序清单附后。

```

10 DEFINT A-Z; DIM MLQ(15, 127), NAMES$(127), NA(127)
20 DEF SEG=&H7000: STARTR=0: FOR I=0 TO 30: READ M$: POKE STARTR+I, VAL("&H"+M$): NEXT
30 DATA 55, 8B, EC, 8B, 76, 08, 8A, 14, 8B, 5E, 06, 8B, 07, 8B, D8
40 DATA B6, 00, B5, 00, B1, 06, B0, 08, B4, 02, CD, 13, 5D, CA, 04, 00
50 STARTW=50: FOR I=0 TO 30: READ M$: POKE STARTW+I, VAL("&H"+M$): NEXT
60 DATA 35, 8B, EC, 8B, 76, 08, 8A, 14, 8B, 5E, 06, 8B, 07, 8B, D8
70 DATA B6, 00, B5, 00, B1, 06, B0, 08, B4, 03, CD, 13, 5D, CA, 04, 00
80 PRINT "请输入驱动器号 (A, B, C) ---- ": AS=INPUT$(1)
90 IF ASC(AS)>96 THEN QDQ=ASC(AS)-97 ELSE QDQ=ASC(AS)-65
100 IF QDQ<0 OR QDQ>3 THEN 80
110 DZ=VARPTR(MLQ(0, 0)): CALL STARTR(QDQ, DZ)
120 NO=0: N1=0: FOR I=0 TO 127: FOR J=0 TO 5
130 N0$=HEX$(MLQ(J, I)): N1$=RIGHT$(N0$, 2): N2$=LEFT$(N0$, 2)
140 IF J=0 AND N1$="E5" THEN N1$="2A"
150 NNS=NNS+CHR$(VAL("&H"+N1$))+CHR$(VAL("&H"+N2$))
160 NEXT J: NAS=LEFT$(NNS, 8)+" "+MID$(NNS, 9, 3)
170 N1$=LEFT$(NAS, 1): IF N1$="X" THEN 190
180 IF N1$<"A" OR N1$>"Z" THEN N1=N1+1: IF N1>3 THEN 210 ELSE 200
190 NAMES$(NO)=NAS: NA(NO)=I: NO=NO+1
200 NNS="": NEXT I
210 CLS: PRINT TAB(30) "目录区表"; TAB(55) "第 屏": PRINT TAB(10) "-----"
220 LOCATE 8, 10: PRINT "( G-Cont S-Stop 0..9-Select)"
230 FOR I=0 TO INT(NO/10)+1: LOCATE 1, 58: PRINT I+1: FOR J=0 TO 9 STEP 2
240 LOCATE INT(J/2)+3, 14: PRINT J, " "; NAMES$(I*10+J), TAB(44) J+1, " "; NAMES$(I*10+J+1): NEXT J
250 AS=INPUT$(1): IF AS="C" OR AS="c" THEN 270 ELSE IF AS="S" OR AS="s" THEN END
260 IF AS>="0" AND AS<="9" THEN N1=I*10+ASC(AS)-48: GOTO 280 ELSE 250
270 FOR J=1 TO 5: LOCATE J+2, 14: PRINT SPACE$(50): NEXT J, 1: END
280 CLS: PRINT "文件 名 --- "; NAMES$(N1): INPUT "原文件名 --- ", AS
290 IF ASC(AS)>90 THEN AS=CHR$(ASC(AS)-32)
300 N1$=HEX$(ASC(MID$(NAMES$(N1), 2, 1))): N2$=HEX$(ASC(LEFT$(AS, 1))): MLQ(0, NA(N1))=VAL("&H"+N1$+N2$): NAMES$(N1)=AS+RIGHT$(NAMES$(N1), 11)
310 DZ=VARPTR(MLQ(0, 0)): CALL STARTW(QDQ, DZ)
320 PRINT "是否需要继续恢复已删除的文件 ( Y/N ) ?": AS=INPUT$(1)
330 IF AS="Y" OR AS="y" THEN 210

```

电摄影过程的单片机控制

华中理工大学计算机系 胡盛斌

摘要: 本文以U-BIX复印机为例, 介绍了电摄影的原理并给出了单片微型机控制复印机的主要电路。

激光印字机和复印机的文字图形输出都采用了电摄影技术: 将光线扫描到旋转的光导鼓上, 形成静电潜象, 再转印到纸上, 经定影后输出。不同之处在于, 复印机是对原稿直接扫描形成有明有暗的光线, 而激光印字机载有文字信息的光

点是由计算机通过接口传送过来的。下面以复印机电摄影过程为例, 介绍单片机的控制电路。

一、电摄影过程简介

一般的复印机结构如图1所示。图中1为光学部件, 2为光导鼓(硒鼓)、3是充电极, 4是显

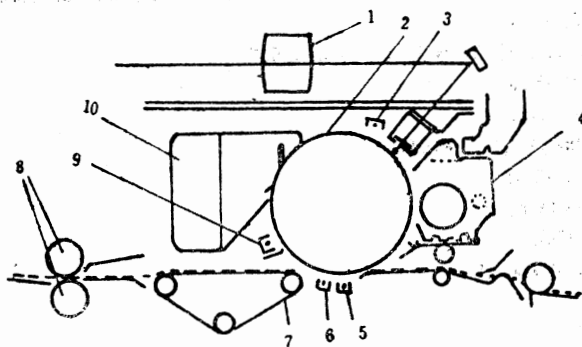


图1

象部件, 5为转印极, 6是分离极, 7是输纸机构, 8是定影辊, 9是清洗极, 10是清扫部件。

复印时工作程序如下:

1. 充电(电晕充电方式), 让硒鼓带上正电荷。

2. 曝光、充电后的硒鼓表面受光线照射时, 接触光线的部分电荷消失, 其余部分仍留有电荷, 从而形成静电潜象。

3. 显象(二元磁刷方式)、硒鼓上由正电荷形成的静电潜象吸附带负电荷的墨粉, 在鼓表面形成粉体图象(图2)。

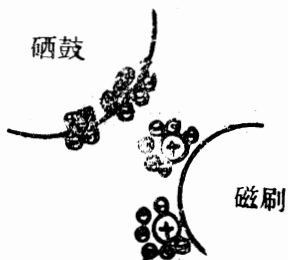


图2

4. 转印(电晕转印方式)、硒鼓表面由负电荷构成的粉体图象被纸张后面的正电荷吸引, 离开鼓表面被吸附到纸上(图3)。

5. 分离(交流电晕分离方式)、硒鼓和纸张由于静电吸力而无法分离。此时在纸张后面提供正、负电荷, 使表面上的电荷取消, 达到纸张和硒鼓分离的目的(图4)。

6. 定影(热辊定影)、纸张上的墨粉图象经过定影辊时, 在定影辊压力和高温的作用下, 软化后定影在纸上, 形成永久性图象。

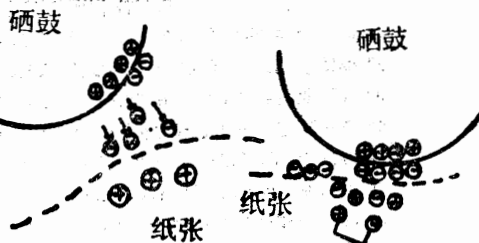


图3

图4

7. 消电(排纸消电与硒鼓消电, 交流电晕消电方式)、对剩余在纸上和鼓上的电荷, 通过正负电荷中和而消除。

8. 清洗(刮板清洗)、将鼓表面剩余的墨粉清洗回收, 保证硒鼓能循环使用。

二、单片机控制电路

U-BIX160RE/1600MR是日本产复印机, 其电气控制部分如图5所示。

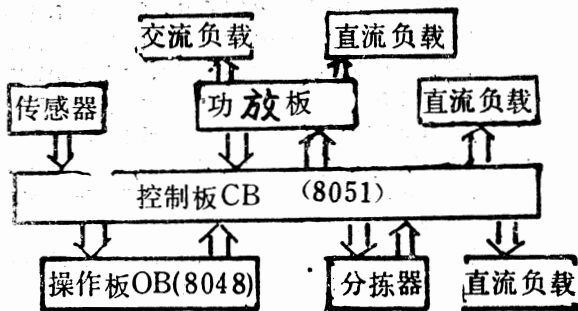


图5

1. 控制板CB

此板主要包括单片机8051(或8031)和输入、输出电路两部分。

8051是INTEL公司第二代单片机, 内部有8位CPU, 4K字节ROM和128字节RAM, 2个16位定时器/计数器, 可寻址64K字节外部数据存储器器和64K字节外部程序存储器, 8031除内部无ROM外, 其余与8051相同。

控制板CB电路框图见图6, 从图中可看出, 三个可编程输入输出接口8255中, IC₂PA口, IC₃PA口、PB口和IC₄的C口为输入端口, IC₂PB口、PC口, IC₃PC口和IC₄PA口、PB口为输出端口。

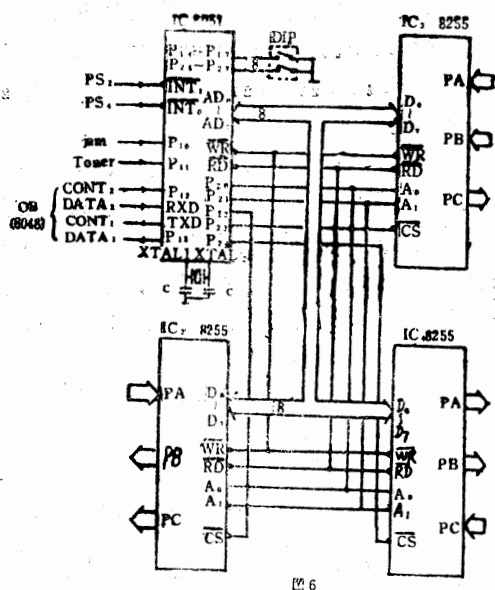


图6

当采用不带ROM的单片机8031时,需外接EPROM2732和锁存器74LS377,此时8031的 \overline{EA} 端应接低电平,以从外接的EPROM2732中读取指令(图7)。

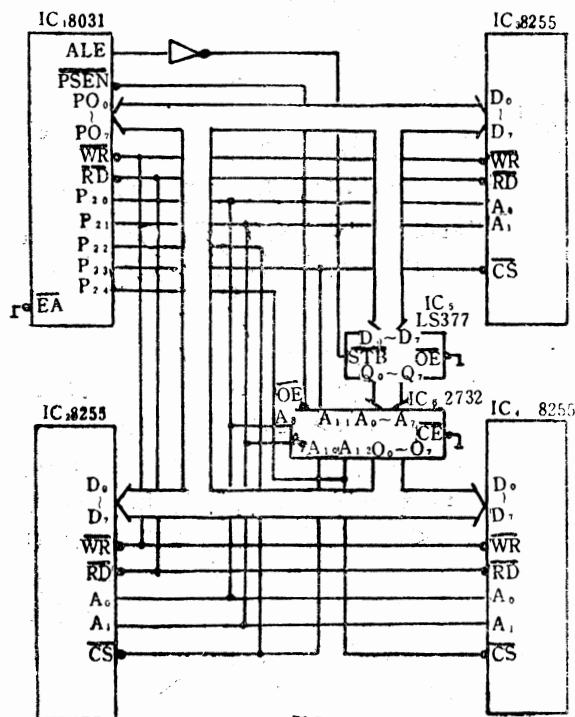


图7

P0口和P2口的输出驱动器以及P0口的输入缓冲器是用于访问外部存储器的, P0口分时输出外部存储器低位地址 $A_0 \sim A_7$ 和读、写的数 $D_0 \sim D_7$, P2口输出外部存储器高位地址 $A_8 \sim A_{12}$ 。用地址锁存允许信号 \overline{ALE} 反相后作为锁存器74LS377的选通信号 \overline{STB} , 将P0口上出现的地址锁存在74LS377中, EPROM 2732的输出允许信号 \overline{OE} 接在8031的程序选通允许信号 \overline{PSEN} 端, 作为外部程序存储器取指的读选通信号。

下面介绍8051 (8031) 的接口信号。

输入信号

IC₂ PA口: 光学部件放大、缩小传感信号 $PS_7 \sim PS_{12}$, 上、下纸盒微型开关信号 MS_6 和 MS_7 。

IC₃ PA口: 起始位置光电传感器信号 PS_3 , 光学部件放大、缩小越位检测信号 PS_5 、 PS_6 , 门开关 MS_6 , 纸暂停开关 MS_2 , 纸尺寸开关 RS_4 , 计数器C(K)、C(PM)。

IC₃ PB口: 纸尺寸簧片检测开关 $RS_1 \sim RS_3$, $RS_6 \sim RS_7$, 无纸检测开关 MS_3 、 MS_4 。

IC₄ PC口: 分拣器输出(4), 恒定电压调节器输入CVR, 浓度检测TLD。

输出信号

IC₂ PB口: 光学部件缩小、放大离合器 MC_1 和 MC_2 , $MC_4 \sim MC_6$, 透镜驱动电机 M_6 , 供粉电机 M_3 。

IC₂ PC口: 阻塞传感及浓度传感 PS_1 , 刮板电机 M_5 , 偏压控制电路输出A、B、C、D。

IC₃ PC口: 第二走纸离合器 MC_3 , 第一走纸电磁铁 SD_1 , SD_2 , 主电机 M_1 , 计数器C(T), C(K)和C(PM)。

IC₄ PA口: 高压控制 HV_1 (充电)、 HV_2 (转印)、 HV_3 (分离)、 HV_4 (消电), 消电CEL控制A、B、C。

IC₄ PB口: 主继电器 RL_1 , 固体继电器SSR, 恒定电压调节器输出CVR, 分拣器输出SORTER(4)。

8051 (8031) 外部中断信号有二个: 一个接编码器光电传感信号 PS_2 ($\overline{INT_1}$), 一个接纸暂停(重启动)信号 PS_4 ($\overline{INT_0}$)。

控制器是如何工作的? 8051 (8031) 首先检测上、下纸盒开关 MS_6 和 MS_7 , 通过操作面板OB控制芯片8048显示出上(或下)纸盒位置, 根据 MS_6 和 MS_7 信号通过8048在面板上显示是否缺纸。由装入纸张尺寸的大小, 通过 IC_3PB , PA 两个端口读入簧片开关 $RS_1 \sim RS_7$ 的状态, 根据不同尺寸的纸张而分别调用ROM中的机器动作时序表数据。机器动作时序表是以光导鼓硒鼓的旋转角度为标准的。在硒鼓轴的传动机构上装有鉴相园板。园板上有180条狭缝(缝宽1mm)。在硒鼓旋转一周360°时, PS_2 产生180个脉冲信号(周期约17.7ms), 每个脉冲对应旋转角度2°。8051对该信号进行计数, 并和ROM中动作时序表数据相比较, 在鉴相板脉冲数达到预期数值时, 8051通过8255接口输出有关动作信号。

按下复印按钮, 单片机8051通过鉴相园板计数信号 PS_2 开始计数。计到115时, 8051发出信号, 使第一供电电磁铁吸合, 把纸张从存纸盒送出。硒鼓转到256°(计数值为128)时, 8051发出信号, 离合器 MC_1 通电, 使光学部件向左移动, 准备曝光。其余动作也都按硒鼓旋转角度预先设计成数据固化在ROM中。控制示意图见图8。

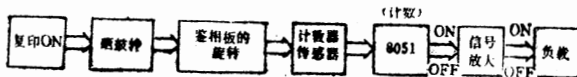


图8

2. 操作面板OB

这部分电路由8048单片机和键盘、显示灯等部分组成, 主要完成以下任务:

1)、键入信号的识别

8048从端口P2发出列扫描脉冲, 从端口P1依次读入行信号。若该行有键按下, 则从P1口对应端读入信号为1, 否则为0。依照有1时的行值和列值, 从8048内的ROM中键值码表查得相应的键值。对数字0~9的显示键及浓、淡、上、下纸盒指示键, 则转相应显示程序显示。对复印, 停止/清除, 浓度等控制键, 则通过8048与8051的通讯, 由8051作相应的处理。

2)、指示灯LED的显示

3)、与控制板CB进行数据信息通讯

8048的 P_{14} 、 P_{15} 接收8051发来的 $DATA_1$ 和 $CONT_1$ 。 P_{26} 和 P_{27} 向8051发送 $DATA_2$ 和 $CONT_2$ 。

例如按下操作面板上的复印键, 8048扫描键盘查得是复印键, 向8051发出信息。8051在接收完一帧时, 标志位RI将激活串行口中断, CPU将转至地址为0023H的内存单元, 开始执行中断服务程序, 清除特殊功能寄存器SCON中RI标志, 按动作时间表发出信号。

操作面板电路框图见图9。

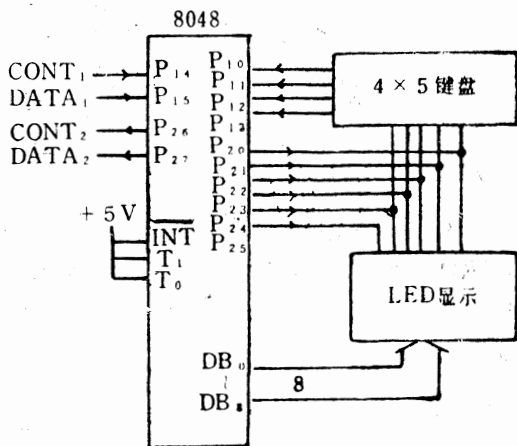


图9

三、几个传感器电路

1. PS_1 阻塞传感器、浓度传感器

PS_1 设计成检测反射光信号。可见光LED₁作为卡纸阻塞测试, 红外光LED作为墨粉浓度检测。

1)、阻塞(Jam)传感

通过电阻 R_a 和 R_b 将 V_1 电压设定为7.7V, 正常工作时保持 $V_1 > V_2$ 的状态, 所以比较器输出为低, 若纸张缠绕在硒鼓上, 随着过强的光反射, PS_1 的输出电流也变大。因此, 比较器输入端出现 $V_1 < V_2$, 此时比较器输出为高。CPU测试到此信号, 一方面向操作面板单片机8048发出信号, 使面板上显示卡纸信号; 另一方面使复印过程停下来, 等待修理, 排除故障。

2)、浓度(Toner)传感

正常工作时墨粉量较充分, 墨粉控制用浓度

片的图象清晰, 光反射较弱, Toner端输出变小, 比较器输入端为 $V_1' > V_2'$, 比较器输出为低。当墨粉不充分时, 墨粉浓度比的图象呈灰白状, Toner端输出变大, 就有 $V_1' < V_2'$, 比较器输出为高。8051从此信号知墨粉浓度不够, 随之启动供粉电机 M_3 补加墨粉(图10)。

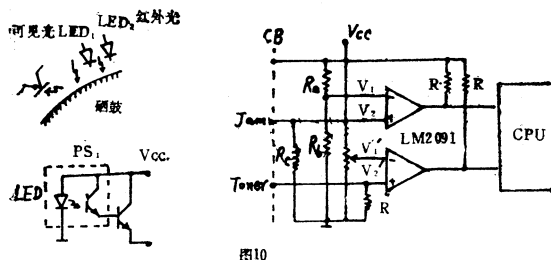


图10

2. 定影灯 L_2 开关电路及温度控制

由图11可以看出, 定影灯 L_2 的开关是由固体继电器SSR(Solid State Relay)控制的。

SSR这个开关元件当其输入端有电压输入时, 其输出两端闭合, 在控制板8051的ROM中装有定影辊的温度数据表。热电偶CA将温度变化量送入温度控制元件 CU_1 , 使 CU_1 根据温度

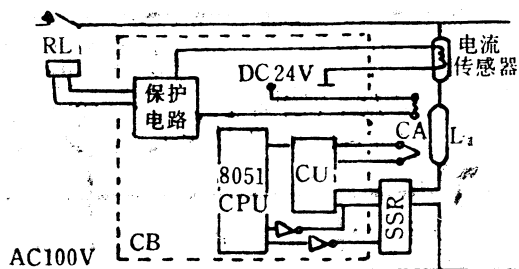


图11

变化正确控制定影灯 L_2 的开关SSR, 将温度保持在CPU给定的温度范围内。

当定影辊过热时, 电流传感器的输入信号会使保护电路动作, 关断继电器 RL_1 , 使交流100V不能加在定影灯上。

参考文献:

1. Intel Component Data Catalog, January 1981.
2. U-BIX 160RE/1600MR SERVICE HANDBOOK, KONISHIROKU PHOTO IND.CO.LTD,TOKYO JAPAN, 1983.4.

微机控制全自动电子乐队

甘肃省计算中心 罗建平

摘要: 本文介绍一种全自动电子乐队的整机电路。它由一部微机(或单板机)根据存入的乐谱控制多路电子音色模拟器, 其输出经电子音色合成器混成后模拟一个电子乐队的演奏。它可用作新创乐曲的试听工具或其它演奏场合。

一、系统工作原理简介

本文介绍的全自动“电子乐队”, 实质上是一个由微机控制多路电子音色模拟器并行自动演奏, 每一个电子音色模拟器产生一特定乐器音色, 从而实现乐队效果的电子合成器。事先将一首乐谱存入微机内存, 并在内存中为每个乐器分谱开辟一个分乐谱缓存区, 用以存放记录音调, 音长, 音强三个参数的向量序列, 如图2所示。

程序采用单微机多道控制(群控)的方法, 即根据分乐谱缓存区的个数, 在演奏时, 依次取

出各缓存区的相应条向量, 输出到微机接口, 从而并行控制各个电子模拟器, 各电子模拟器的输出经电子音色合成器合成后, 为听者呈现出电子乐队的演奏效果。

二、系统的构成及控制线分配

电子模拟器功能结构如图1所示。其中 S_0, S_1 (=00或01或10或11)用于控制电子模拟器的音强, 可分别对应四个等级: 强、中、弱、无声。 $H_0 \sim H_{11}$ 可控制32度音高, 共四个八度音程, 并由 h/e 来控制是全音还是半音。为了保证音调

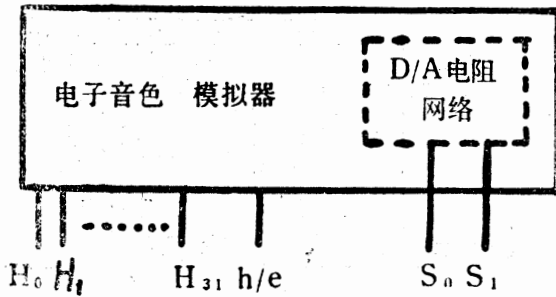


图1

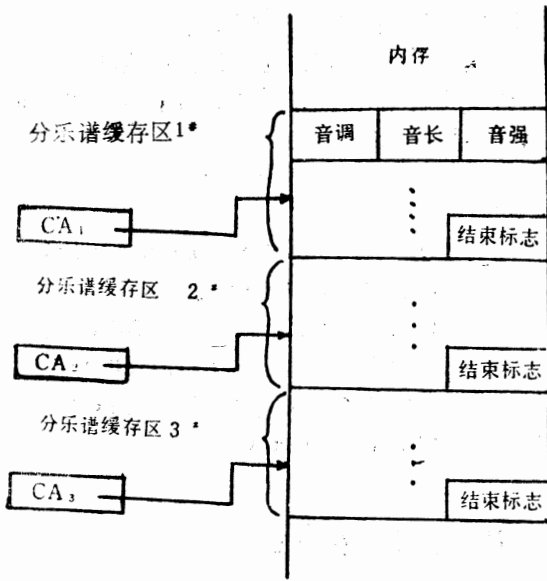


图2 分乐谱缓存区

1*当前向量地址指针

2*当前向量地址指针

3*当前向量地址指针

的准确度，其音高由电子模拟器内部电路确定，外部仅由 H_i ($i=0, 1, 2, \dots, 31$) 来进行断开控制。而音强精确度要求略次，故可用模拟器内简单的D/A电路来解码。

基本系统由一部以8086/8088为CPU的微机或单板机、一片并行接口芯片8255、若干译码电路及分离式电子模拟器与电子音色合成器组成，其相互连接见图3所示。

并行接口芯片8255共有三个数据输出口 ($PA_0 \sim PA_7$, $PB_0 \sim PB_7$, $PC_0 \sim PC_7$)，每口为八个二进制位，各口分别担任一路电子模拟器的控

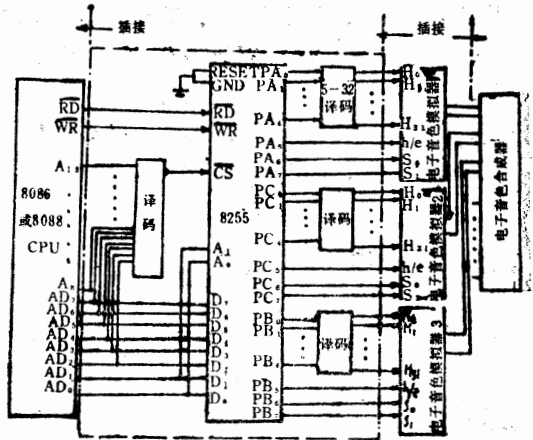


图3 系统连接

制。 $PX_0 \sim PX_4$ (X为A, 或B, 或C, 以下类同) 通过5位输入译码器获得32位输出, 用以控制32个音调开关 $H_0 \sim H_{31}$, 给定时刻仅有一位为有效。 PX_5 与 PX_6 、 PX_7 直接连到电子模拟器的 h/e 和 $S_1 S_0$, 作为全/半音及音强控制。三个电子模拟器的输出接入电子音色合成器的三个输入通道, 以获得多乐器音色并行演奏的效果。

三、演奏定时控制

演奏程序利用可编程定时器8253的一个通道来获得基本时间节拍(8253连接图从略)。8253被编程为每隔1/32拍的时间(或根据需要定为其它时间)向CPU定时中断一次。

分乐谱缓存区中任一向量的音长一项, 为对应音调应持续的时间长度, 它为基本节拍(即1/32拍)时间的整数倍。

程序中为每一分乐谱缓存区设置一个“当前向量地址”变量 CA_i ($i=1, 2, 3$)，每当8253向程序中断一次, 程序便按 CA_i 提供的“当前向量地址”依次扫描各缓存区中当前向量中的“音长”单元, 并将单元内容减一。当结果为0时, 表示该向量所示音调的持续时间已完成, 从而修改 CA_i 使之指向下一向量, 并把新向量中有关内容(音调、音强)送到对应的并行口 $PX_0 \sim PX_7$, 以使对应的模拟器发出新的音调及音强。否则不作任何修改, 让其继续原有音调。若要使乐谱能重复演奏, 可为每个缓存区设立一个临时变量 t_i , 用它来代替完成音长单元的计数功能, 而不

IBM-PC Z80在线仿真器接口板的研制

上海石化总厂职工大学 徐飞虹

摘要: 本文提出了以IBM-PC接口板形式, 研制微机仿真开发机的方法。这使仿真开发机结构大大简化, 效率显著提高, 成本大幅度下降。这就为自制微机仿真开发机创造了条件。本文详细叙述了它的硬件结构和设计方法。

一、前言

目前国内外十分重视微机开发系统的研制。国内广泛流行的仿真开发机结构如图1。IBM-PC机用作人机对话工具, 通过RS-232C串行口实现与仿真机的通信。一切开发功能由仿真机本身完成。由于它是一个有CPU、ROM、RAM、I/O及监控软件的独立微机系统, 因此结构复杂, 价格昂贵(3000元以上)。它主要有如下缺点:

(1) 由于以串行方式与IBM-PC机通信, 使工作速度大大下降, 出错率提高。额外的串行通信硬件和软件, 使硬件成本上升, 软件编制复杂, 效率下降。

(2) IBM-PC机仅仅被用作人机对话工具, 没有充分利用它内部软硬资源(如CPU、RAM、电源、时钟)的巨大潜力, 造成资源的极大浪费。

(3) 仿真机体积笨拙, 加上PC机和目标系统(开发对象)一起, 使工作台上空间颇感紧张, 操作不便。

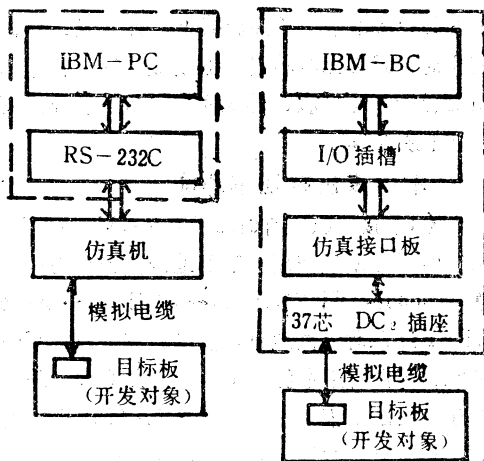


图1 两种仿真机的比较

作者研制的仿真机如图1。它仅仅由一块接口板, 一根37芯扁平电缆及一个40芯IC插脚组成。使用时, 将接口板插入PC机内任一I/O插槽中, 将40芯IC插脚插入目标系统(开发对象)的Z80芯片插座中, 用37芯电缆将二者相联。这种仿真机的工作原理与前述仿真机类似, 仅仅在

致破坏音长单元的内容。另外, 每个缓存区末均设有一个结束标志, 以停止该道的演奏。

四、基本系统的扩充及布线考虑

此基本系统能控制三路电子模拟器并行演奏, 但对一些乐器种类要求较多的乐曲尚不能满足需要。为此只需另行增加一片或数片并行接口芯片8255和相应的译码电路以及电子模拟器, 连接同上; 并在内存中增加分乐谱缓存区数目, 扩大演奏程序的扫描范围即可。为了便于系统规模的增减及随意更换音色模拟器, 可将8255及有关译码电路做成一块分离式的标准插件, 使得可任

意插接若干块到CPU的总线上; 同时, 电子音色模拟器与该插件之间亦做成插接式的。而演奏定时控制程序亦设计成相应的通用形式。

参考文献

1. 美国微处理器支持芯片手册, [美] T.J. 拜厄斯编, 机械工业出版社, 1988, 3;
2. 接口与通信, 黎康保, 朱宏兴, 西北工业大学出版社, 1987, 9;
3. 华星电子琴手册, 1987。

于用PC机内的CPU、RAM取代仿真机的CPU、ROM和RAM; 直接由PC机提供+5V电源和时钟。整个仿真机监控软件以磁盘文件形式存在磁盘上, 运行时调入PC内存。这样就省去了前述仿真机中的CPU、ROM、RAM、电源、时钟, 串行通信硬件, 仿真机的一切功能在PC机直接控制下工作。这就使结构大大简化, 效率显著提高, 成本大幅度下降。整个硬件成本才150元左右。该接口板可像PC机内其它接口板一样长期插在I/O插槽中, 不会影响PC机正常工作。在使用时, 只要把37芯扁平电缆插头插入露在PC机后盖上的接口板37芯插座内。因此操作方便, 不占机外空间。

本仿真机有如下功能:

(1) 接口板上有8KB仿真RAM, 它可“出借”给目标系统使用。它的地址范围可由用户任意在 0000~1FFFH, 2000~3FFFH, ... E000~FFFFH 八种不同地址中选择以满足目标系统的需要。用户可用本仿真机的行汇编命令, 编制源程序。编制结束, 直接在仿真RAM中生成机器码, 然后可对源程序运行、调试和修改, 直至满意为止。这就对调试目标系统的软硬件提供了便利。

(2) 仿真机在软件控制下可实现行汇编, 反汇编, 单步, 单周期, 断点, 显示修改寄存器, 显示修改内存存储器, 输入, 输出, 数据块填充, 传送, 检测RAM、ROM, 跟踪, 复位等功能。利用PC机上已有的各种软件(如行编辑, 交叉汇编, 反汇编)使整个开发过程变得更为方便。

(3) 它与作者研制的IBM-PC高速通用多功能EPROM编程器接口板^[1]一起组成了一台实用的开发系统, 可随时对EPROM或单片机进行高速编程, 拷贝和比较检验。

(4) 仿真机具有逻辑分析仪的某些功能, 对检查目标系统的硬件故障十分有用。

二、硬件结构

如图2, 接口板由I/O控制电路、仿真CPU、I/O地址译码器、8K仿真RAM, RAM地址选择电路, 总线控制电路和机器周期步进电路组成。具体电路如图3和图4。简述如下:

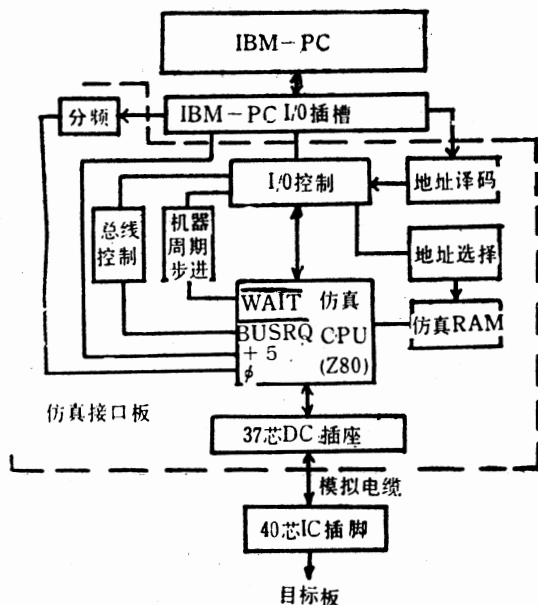


图2 IBM-PC Z80仿真器框图

(1) I/O控制电路:

它由2块并行接口芯片8255组成。它所提供的6个I/O口用来实现PC机和本接口板的数据通信。设计I/O地址译码器(74LS30), 使8255(1)的口地址为1F8H~1FBH, 8255(2)的口地址为1FCH~1FFH。查阅IBM-PC技术手册, 这个地址范围没有和PC机内其它I/O口地址冲突。将PC机内的AEN信号接入地址译码器的输入, 是为了确保PC机进行DMA操作时, 本接口板停止工作。8255(1)的A口, B口设置为输入, C口为输出。A、B口用来读取Z80 A₀~A₁₅的地址信号, C口用来向Z80传送数据和指令。8255(2)的A口为输入, B、C口为输出。A口用来监视和读取Z80控制信号, 以便了解Z80处于什么机器周期。B口用来发出控制信号以实现Z80总线请求电路, 机器周期步进电路的控制。C口用于选择仿真RAM的地址范围。

(2) 仿真CPU (Z80):

它有2种用途。其一是当将目标系统断开时, 它与仿真RAM组成一台Z80微机系统; 其二是当接上目标系统时, 它代替目标系统中的CPU (Z80), 执行目标系统各种实时控制任务, 因

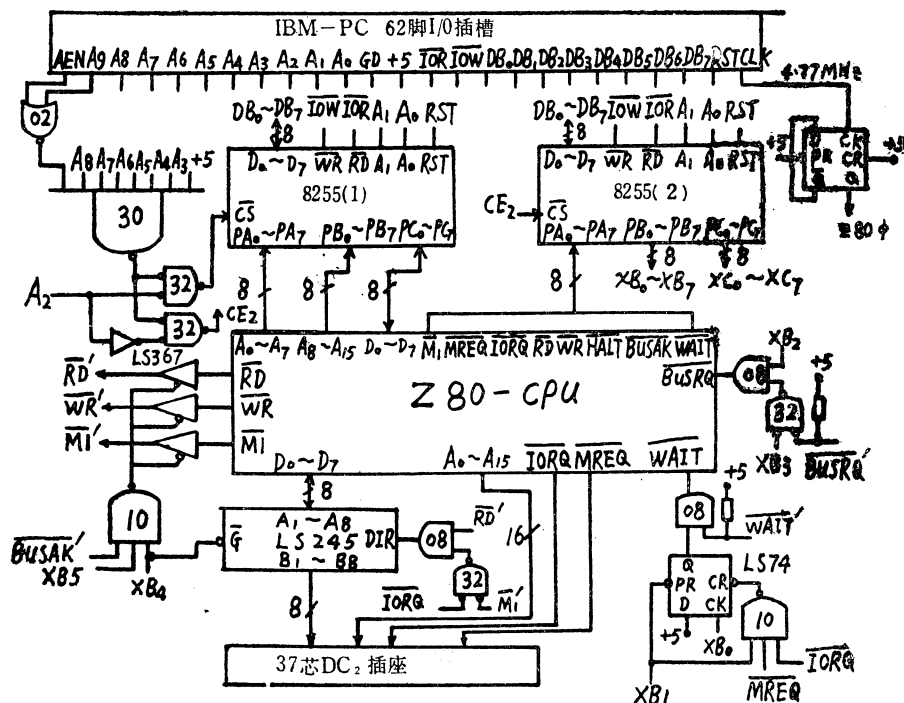


图 3 IBM~PC Z80 仿真接口板电路原理

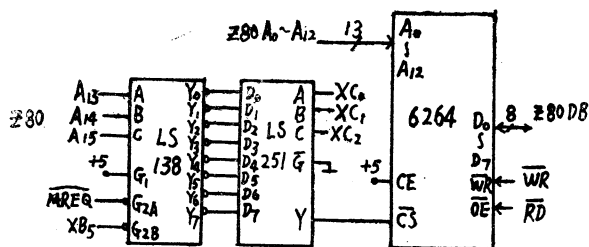


图4 仿真RAM地址选择

此称为在线仿真CPU。目标系统的工作状态及故障一般都在CPU三总线的状态上有所反映,由于仿真CPU的三总线的信号又可通过8255传送给IBM—PC,因此就可通过软件随时将目标系统的各种状态和运行结果在PC机CRT上显示,用户可用仿真机的各种命令来对目标系统进行调试。仿真CPU的 $A_0 \sim A_{15}$, \overline{IORQ} 和 \overline{MREQ} 直接与目标系统相连,而 $D_0 \sim D_7$ 及 $\overline{M_1}$, \overline{RD} , \overline{WR} 则通过缓冲器(74LS245和74LS367)和目标系统相连。这样做有2个目的:其一,当IBM—PC通过8255(1)C口向Z80传送数据时,要先将74LS245关掉,以避免这些数据干扰目标系统。其

二, 当仿真CPU向仿真RAM传送数据时, 先关掉74LS245和74LS367, 可防止这些数据写入目标系统。在实时运行目标系统程序时要将245和367接通, 它们还增大了总线的驱动能力。

(3) 仿真RAM和地址选择电路:

如图4仿真RAM选用6264 (8K×8)。它的13根地址线, 8根数据线及 \overline{WR} , \overline{OE} 脚直接与仿真CPU对应脚相连。选片信号则接8~1多路数据选择器74LS251的输出脚Y。按图中所接, 74LS138的输出脚Y₀的地址范围为0000~1FFFH, Y₁为2000~3FFFH, ……Y₇的地址范围为E000~FFFFH。控制74LS251的选择脚A₁B、C的电平, 就可选择其输出脚Y与Y₀~Y₇中的一根线相联, 从而使6264的选片脚 \overline{CS} 在选定的地址范围内有效。对8255(2)的C口写不同的控制字, 就可实现对6264的地址选择。当目标系统不需仿真RAM时, 可使8255(2)的PB5置“1”, 从而禁止74LS138工作, 进而使6264停止工作。

(4) 总线控制电路:

当Z80的 $\overline{\text{BUSERQ}}$ 有效并进入总线响应周期时,它的总线处于高阻状态。从图3可见,仿真CPU的三总线既与8255(1), (2)相连,也与目标系统和仿真RAM相连。有时需要实现8255和仿真RAM之间的数据传送,这时就必须使Z80让出总线以免发生数据冲突。这可通过8255(2)的PB3向Z80 $\overline{\text{BUSERQ}}$ 提出总线请求。此外,目标系统的总线请求亦可通过74LS32和74LS08加到Z80 $\overline{\text{BUSERQ}}$ 脚。

(5) 机器周期步进控制电路:

本仿真器对Z80指令的仿真是以机器周期为基本单位一步一步进行的。现以仿真Z80指令“JP2050H”(其机器码为:C3H, 50H, 20H)为例说明本电路原理。在取指周期起始时刻, $\overline{\text{IORQ}}$ 和 $\overline{\text{MREQ}}$ 均为高电平,阅图3,设此时的XB1(即8255(2)的PB1)也为高电平,则74LS10的输出为“0”,置74LS74的输出Q为“0”,这样就使Z80的 $\overline{\text{WAIT}}$ 脚为“0”,提出等待请求。Z80是在T2周期检查 $\overline{\text{WAIT}}$ 信号的,当它发现为“0”就插入等待周期TW,这时,虽然 $\overline{\text{MREQ}}$ 已变“0”,导致74LS74的置0端 $\overline{\text{CR}}$ 恢复为“1”,但Q状态仍维持“0”不变。因此,只要不对XB0和XB1加控制信号,Z80就一直处于等待状态。这时可将C3H通过8255(1)的C口送上Z80DB,然后使XB0产生正脉冲,使74LS74Q端为“1”,撤消了Z80 $\overline{\text{WAIT}}$ 脚上的请求,Z80脱离Tw进入T3周期,它在T3前沿读取DB上的C3H,进入了下一个机器周期。在第二个机器周期,以同样方法使

Z80读取50H,在第三个机器周期读取20H。这样Z80就从8255(1)的C口读入了一个完整的指令JP2050H。经指令译码,Z80将2050H放入程序计数器PC,并将2050H送上地址总线A₀~A₁₅,我们可通过8255(1)A、B口读取A₀~A₁₅看看它是否为2050H,并将此信息送PC机CRT显示给用户。通过对Z80 $\overline{\text{WAIT}}$ 端的这种控制过程,就可使仿真CPU以机器周期为单位一步一步地执行PC机送来的指令,并可随时对执行结果检测显示。这种执行→检测→执行的过程是本仿真器最本质的工作机理。若要使Z80连续执行一般程序,则可使XB1为“0”,它迫使74LS74Q端置“1”而不受 $\overline{\text{IORQ}}$ 和 $\overline{\text{MREQ}}$ 的影响,Z80 $\overline{\text{WAIT}}$ 端就不会出现“0”,从而保证指令的连续执行。

三、软件编制:

由上所述,仿真器的各种功能均由IBM-PC通过8255(1), (2)的6个口发出各种控制字,指令或数据实现的。可见整个软件是由IBM-PC直接控制的。因此本软件以8086汇编语言编制。软件按仿真器各个命令划分成一个个模块,此外还需编制许多基本操作的子程序。软件的工作量相当大,整个程序约12KB左右。限于篇幅,在此不再叙述。

参考文献

- [1]《微小型计算机开发与应用》1988年第5期:
徐飞虹:“IBM-PC通用EPROM编程器接口板的研制”

通用快速数字信号处理系统的设计与实现

电子科技大学 汪亚南 陈尚勤

〔摘要〕本文介绍一种基于TMS32010单片机的快速数字信号处理系统。该系统由两部分组成:即TMS32010单片机开发系统和数字信号处理信道。系统结构特点是以TP801作支撑,借用TP801的资源来实现TMS32010的功能。对于开发系统部分,着重介绍存储器连接。在信道部分,则主要内容是A/D转换器和输入输出接口。全文最后介绍了软件固化,并列出了部分软件。本系统可用于短波数传系统自适应均衡、误码检测及一切需用快速数字信号处理的领域。

(一) TMS32010单片机开发系统

TMS32010 于 83 年问世。由于它与通用微处理器之间的显著差别,因而广泛用于数字信号处理领域。它的最富于吸引力的特点是大多数指令皆为单周期指令,指令的执行时间为 200ns。所以“32010”是用于快速数字信号处理,特别是数字图象处理的理想器件。

单片机由于本身结构的原因,使它对硬件和软件的支持能力有限。因而在研制单片机应用系统时,必须配备一定的开发工具。TMS32010也是如此。

TP801 是我国广大计算机用户常用并拥有的计算机设备之一,Z80 汇编语言亦为广大用户所熟知。因此,使用 TP801 作为 TMS32010 的支持硬件确具有现实意义。实际处理时,TP801 可脱离,故不影响 TMS32010 的快速处理(每秒 500 万次)的性能。

1. 系统结构

基于 TMS32010 单片机的快速数字信号处理系统,其框图如图 1 所示。

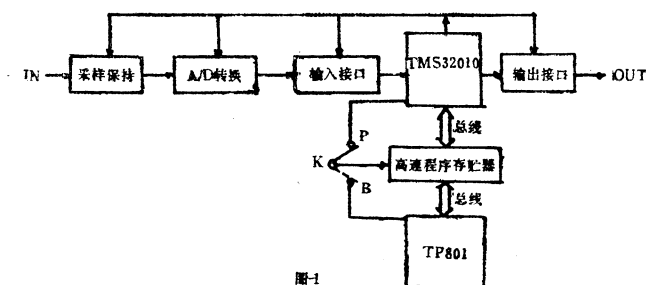


图 1

由图看出,这是包含两个 CPU 的计算机应用系统。

TP801 带有 8 片静态 RAM 2114 并通过 S-100 总线实现系统连接。TP801 的功能是用作程序输入与调试。

TMS32010 的内部不带程序存储器,因而必须外接。并要求选用高速存储器芯片。TMS32010 的作用是运行程序和提供控制信号。

高速程序存储器采用 4K×1 位静态 RAM 2147,使用 16 片组成 4K×16 位程序存储器。该存储器不仅与“32010”相连,而且也同 TP801

连接。

开关 K 是本系统的转换装置。

当 K 打至“B”时,TP801 运行。从单板机的角度看,2147 是它的扩展 RAM。利用单板机上的键盘可控制程序存储器的读写。因此,TMS32010 应用程序的机器代码可通过 TP801 键盘送入 2147。

当 K 打至“P”时,TMS32010 启动。此时,从单片机的角度看,2147 应该是属于 TMS32010 的。单片机立即运行转贮在 2147 中的应用程序。如果该程序需要修改,可再启动 TP801。如此反复几次,直到 TMS32010 应用程序被调试到完全令人满意为止。实际上,开关 K 的功能是由一般控制程序实现的。

2. 存储器连接

存储器连接为本系统的技术关键,也是较复杂的部分。由上所述,TMS32010 与 TP801 共用外接程序存储器,每即程序存储器既与 TMS32010 连接又与 TP801 连接。存储器芯片的信号线分为三种:地址信号、数据信号和控制信号,系统连接可用下图说明。

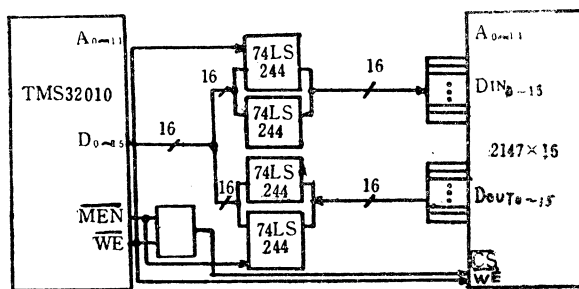


图 2

图 2 是存储器与 TMS32010 之间数据线及控制信号线的连接。二者之间采用了数据缓冲器,74LS244。2147 有两条控制信号线 WE 和 CS。其中 WE 直接与 TMS32010 的 WE 相连,而后者的 WE 与 MEN 相与后作为 2147 的 CS 信号。

图 3 是 2147 与 TP801 的连接。TP801 的数据线、地址线以及控制信号皆与 TMS32010 有很大不同。

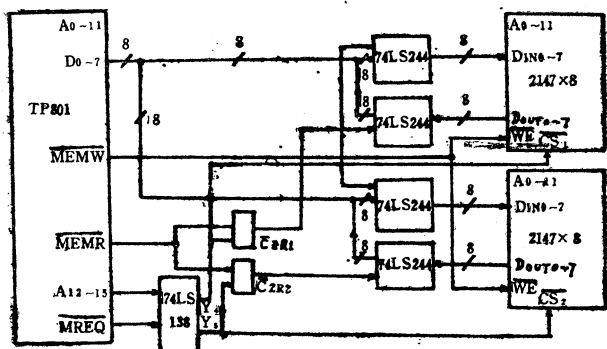


图-3

TP801的数据线是八位的,为了能对 $4K \times 16$ 位的程序存储器寻址,则必须能寻址 $3K \times 8$ 位的存贮单元。

TP801的地址线为16根,其中 $A_{12} \sim 15$ 与 \overline{MREQ} 通过译码器产生两个片选 (Y_4 和 Y_5) 信号。

TP801 的读写控制信号有 \overline{MEMR} 和 \overline{MEMW} 。其中 \overline{MEMR} 与 Y_4 相或后得到 C_{ZR1} ; \overline{MEMR} 与 Y_5 相或后得到 C_{ZR2} , 而 \overline{MEMW} 则作为存储器的 \overline{WE} 信号。

无论是地址线、数据线或读写线皆有二选一即选中TMS32010或是TP801的问题。因而,在系统内,必须采用二选一控制逻辑,现使用数据选择器74LS157,它的选择端S与单板机上的Z80—PIO接口的B口(PB_7)连通。当 PB_7 处于高电平,亦即 $S=“1”$ 时,选择B(单板机);当 PB_7 为低电平,亦即 $S=“0”$ 时,选择A(单片机)。二选一控制逻辑如图一4所示。

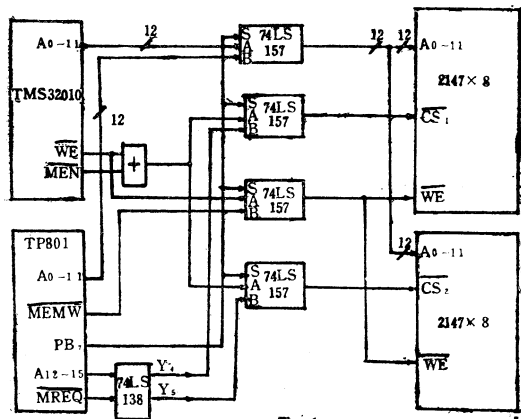


图-4

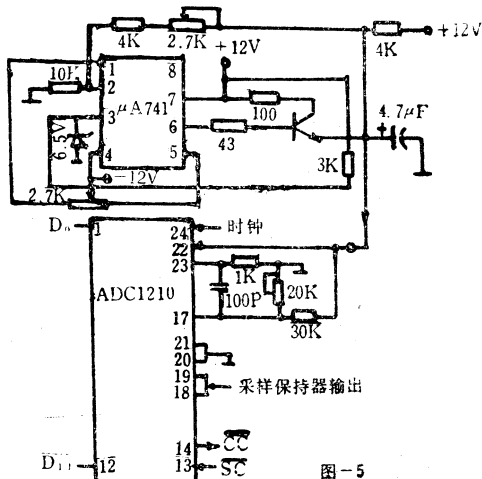


图-5

(二) 数字信号处理信道

本系统的输入信号是带有失真的模拟信号,而输出是经过处理以后的数字信号序列。由于不需要控制某种对象和通过显示器显示结果,因此,输出未接入D/A转换器。

1. A/D转换电路

模拟信号不能直接进入计算机加工处理,故需经过A/D转换。图一5为A/D转换器及其 V_{REF} 稳压电路。

ADC1210是12位A/D芯片,CMOS器件。A/D的参考电源 V_{REF} 为10.24V,由专用稳压电源供给。稳压电路如图一5所示。其中稳压管选用2CW7—6.5V, V_{REF} 值的高低直接影响A/D转换的精度。对A/D转换器进行调试时,要进行满量程和零漂移调节。

A/D转换器的时钟,由TMS32010输出的5MHz时钟经40次分频得到,即125KHz。

2. 输入/输出接口

TMS32010 单片机共使用两组(4片)8212并行接口。其中一组作为输入接口,另一组作为输出接口。

在输入接口电路中,8212的 $\overline{DS_1}$ 直接接地, $\overline{DS_2}$ 信号是由TMS32010的 \overline{DEN} 和74LS138的输出 Y_6 相或非后得到。

在输出接口电路中,8212的 $\overline{DS_1}$ 同样直接接地, $\overline{DS_2}$ 接高电平。这样,8212的数据缓冲器

始终是打开的。TMS32010的 \overline{WE} 与三~八译码器的输出 Y_6 相或非后的信号作为输出接口 8212 的选通信号。一旦执行 OUT 指令时,就可输出数据。

(三) 软件固化

利用 TP801 单板机支持 TMS32010 的快速数字信号处理系统,结构简单,操作方便,称得上是价廉物美的支持硬件,我们把该系统制作成双层结构,顶层放置 TP801,以便于操作;底层放置 TMS32010。两层之间通过 S-100 总线插座使用 40 芯扁平电缆连接。但是,美中不足的是所有程序都是由键盘手工输入,既费时,又易出错。因此,在 TMS32010 应用程序调试完毕后,可把它注入到 EPROM 中去。这样,只要一开机,EPROM 中写入的程序便传送到 2147 中。

EPROM 写入器是软件固化必不可少的工具,写入器的所有操作都是在程序控制下进行的。在 TP801 单板机内部装有 EPROM 写入电路,单板机面板上有两个备插 EPROM 的插座 PROM1 和 PROM2。本系统采用的是最常用的 EPROM2716。为了对 EPROM 进行写入,除有上述硬件电路装置外,还要通过 EPROM 的写入程序这一软件才能解决 EPROM 的写入问题,将用户需要的程序写入到 EPROM 中。

下面把本系统中用到的几个程序段分别用 Z80 汇编语言和 TMS32010 汇编语言写出于后。

程序1: 二选一控制

```

LD A, CFH
OUT (83H), A
LD A, 00H
OUT (83H), A

TP801-C      TMS32010
LD A, FFH    LD A, 00H
OUT (81H), A OUT (81H), A

HALT
  
```

程序2: EPROM编程

```

LD HL, 0819H
LD DE, 4000H
LD BC, 12
LDIR
LD HL, 0A40H
LD DE, 5000H
LD BC, 13
LDIR
JP 0000H
  
```

程序3: A/D转换

```

LACK >14
SACL >2
LACK >32
SACL >3
OUT >2, 3
OUT >3, 4

PAF NOP
BIOZ ST
B PAF
ST IN 7, 6
OUT 7, 6
LARK AR1, 28
LARP AR1
PAU NOP
BANZ PAU
B PAF
END
  
```

结束语

以上为我们所研制的快速数字信号处理系统的片断介绍。实验结果表明,本系统方案是切实可行的,应用本系统能够得到令人满意的效果。



多头高分辨率彩色显示系统

南京供电局 丁 锋

一、引言

随着PC机的普及,在办公自动化及工业监控的实际使用中,常常需要同时显示几个不同的画面,并希望用多个键盘对相应的显示器进行操作,这样如果用多台PC机固然可以解决问题,但有两个缺点,一是浪费硬件资源,投资高;二是不能共享软件。我们选用KVS-D₄图形显示卡(适用于PC/XT和PC/AT)开发而成的多头高分辨率彩色显示系统实现了由多键盘控制多头显示,并且不需要修改DOS和其它系统软件,硬件简单,程序设计方便,取得了较好的效果。

二、硬件结构

图1为系统结构示意图,系统选用KVS-D₄图形显示卡,该卡主芯片为82720智能图形显示控制器,自带128K×8Bit的RAM和2M×1Bit的汉字库,具有较强的绘图功能,可直接插在PC总线上,口地址由跳接线选择,显示器可以选用SHARP等640×400分辨率的工业监视器,图形卡与显示器接口和PC原有图形卡相同。

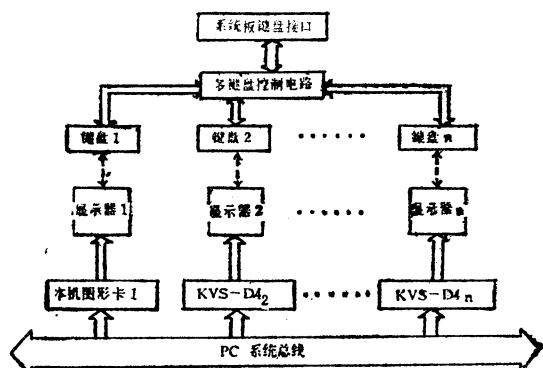


图1 硬件结构示意图

多键盘控制电路是为实现了键盘和显示器的一一对应控制而设计的,它利用各键盘发往主机的键盘时钟信号(KBDCLK),分频后作为选通脉冲向主机插入特定字符,并与主机软件配合识别相应键盘输入,电路还将各键盘时钟信号与键盘

数据信号逻辑组合后接到主机键盘接口,该电路特点是简易、低廉,使用方便,且可改造成普通键盘。

二、程序设计

利用KVS-D₄的语言支持库可以方便地用汇编语言、编译FORTRAN、BASIC等高级语言来作图和显示汉字。以大家熟悉的编译BASIC为例,如想选择I/O地址为251H,250H的显示卡作图,画一绿色矩形可以用以下语句实现:

```
I0% = 250H
I1% = 251H           ; 赋口地址
CALL SPORTS(I0%, I1%)
                        ; 选择显示卡
C% = 4                ; 选绿色
CALL (SCOLRCC%)
CALL SBOX(X%, Y%, X1%, Y1%)
                        ; 画矩形
```

为了节省篇章,有关具体使用方法可以查阅空军第五研究所编写的《系列高分辨智能彩色汉字图形显示系统手册》。

图2给出的框图介绍了实现多头动态显示和根据键盘输入实现功能选择的一种方法,

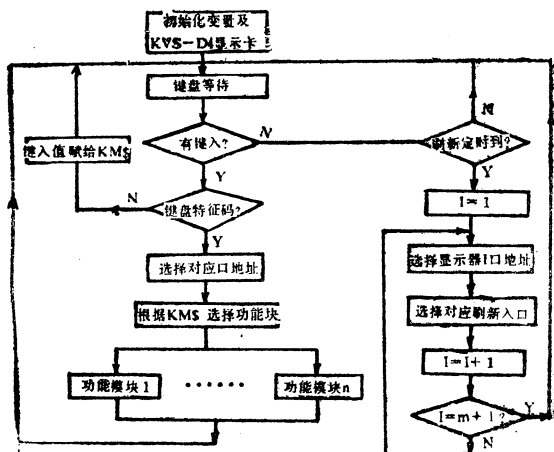


图2 系统程序框图

AR—2463打印机实现高分辨图形/字符屏幕拷贝

西安电子科技大学计算机运动技术研究中心 王南江

AR2463 高速汉字打印机带有由用户选择不同字形的硬字库,并能通过调节拔码开关将打印机设置成3070或M2024/M1724打印机兼容的工作方式,目前它是一种非常流行的高速24针打印机。但是对GW0520C—H微机或配置CH汉卡和高分辨彩色图形适配器的XT/286或AT机的用户,在使用BASIC或汇编语言绘制屏幕图形时,按屏幕打印键Shift+Prtsc不能进行正确的屏幕图形打印,打印当前屏幕的字符信息。虽然用户以将打印机拔码开关设置成3070打印机兼容工作方式,并在开机后运行3070屏幕拷贝程序P3070SG.COM,这样便可按Shift+Prtsc键将图形及字符(包括汉字)拷贝到打印机上,不过这种3070兼容工作方式对2463的命令格式不支持,打印机的一系列操作命令,必须按照3070打印机的规约设置,2463自身多种字体和命令不得不舍弃,这对打印机的用户来说为之遗憾。为了使AR2463不改变工作方式而能对屏幕进行正常拷贝,可以用DEBUG程序对P3070SG.COM

进行跟踪并修改,使打印机在2463工作方式下通过按Shift+Prsc键将屏幕图形/字符包括汉字正常拷贝下来。

P3070SG.COM是长城微机开发公司编写的外部I/O管理模块,它和GWBIOS3.00的其它模块连接在一起。当该模块运行驻留内存后,在任何时候只要按下Shift+Prtsc键或者程序中执行中断INT5H调用,便转向该模块将屏幕的图形与字符拷贝至打印机。程序的粗框图见图1。

GWBIOS3.00允许用户将自己编写的该屏幕打印模块驻留内存,并和GWBIOS3.00的其它模块连接在一起使用。这样做可使I/O入口地址不改变,避免了由于地址的改变而造成的软件出错问题,并可以转回到内部I/O处理而不需要重新开机。GWBIOS3.00在基本I/O中能设置有关的四个I/O管理模块入口,这四个模块是:

1. 键盘的系统接口,调用向量:INT16H
2. 打印机的系统接口,调用向量:INT17H

其中m为显示器数,n为功能模块数,程序功能模块化设计有利于各显示头通用,也便于管理和维护,程序中键盘接收部分的键盘特征码为插在正常键入字符后的各键盘识别字符,由键盘控制电路自动完成。另外,程序中的定时刷新部分主要是为动态显示考虑的,这段程序在无键入和无功能服务要求的空暇时间内不断对各显示器扫描,根据需要选择显示器刷新入口更新显示的画面式数据。

三、绘图软件介绍

为了使应用多头高分辨彩色显示系统更方便,我们开发了一套绘图软件,它既可以在一般高分辨率的PC机上编辑图形,也可以直接在KVS—D₄支持的显示器上编辑图形,能用键盘方便的绘制点、线、圆、矩形、填色多边形等功能和移位、转向、复制、放大等功能,此外还可以用来确定动态显示数据的位置等参数,并能方

便地与主程序接口,绘图软件既可以将图形作为数据文件存储、调用,也可以将图象点压缩存储、调用;两种方式都具有速度快、调用方便的特点,该软件也可以推广在一般PC机上使用。程序用编译BASIC编写。

四、结束语

南京供电局用该系统组成的电网调度自动化系统实时终端于89年4月在南京通过了部级验收,与会专家认为该系统实时性强、投资省、画面推出速度快,多键盘使用互不干扰,操作方便,此外由绘图软件绘制画面也很美观。该系统在其它行业实时监控领域及办公自动化方面均有一定推广意义。

该系统在研制过程中得到河海大学赵鸣梁同志,空军第五研究所张苏宁等同志及南京供电局沈兵兵、宋健生、梁萍同志的协作,一并致谢。

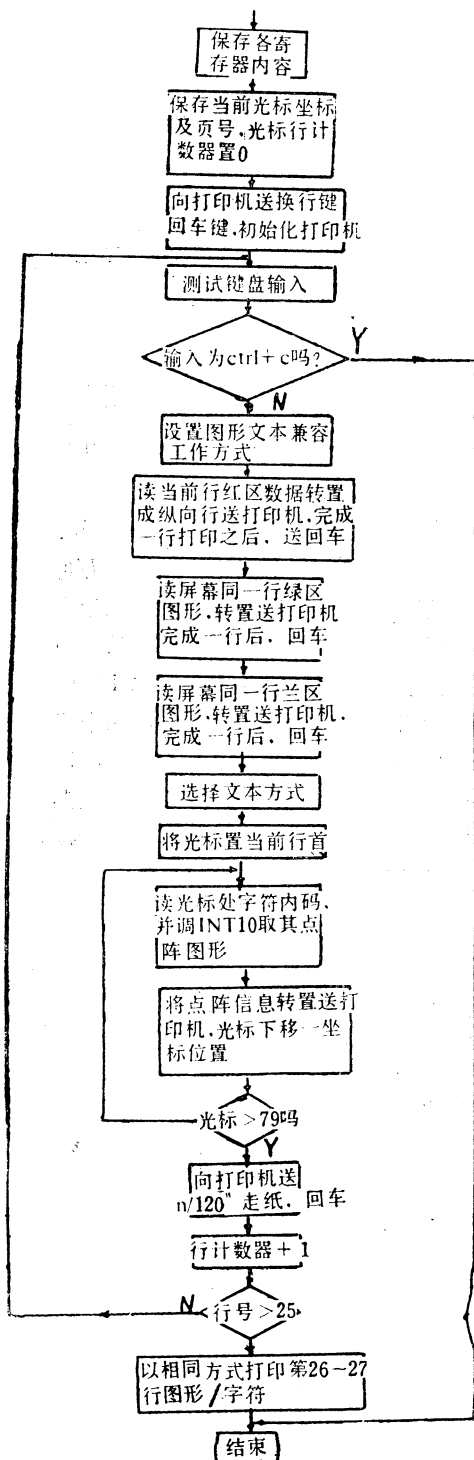


图1 程序框图

3. 屏幕打印接口, 调用向量: INT5H

4. 显示的系统接口, 调用向量 INT10H
在进入这四个I/O模块时, 如果 [AH] = OFFH, 就意味着调用内外I/O处理模块。

[AH] = OFFH;

输入参数:

[AL] = 0 设立新的处理模块入口地址

[BP]: [SI] = 新的处理模块的入口地址

[AL] = 1 取消外部处理模块控制权
该I/O处理权返回给内部I/O模块。

输出参数:

如果使用 [AL] = 0 设立新的处理模块入口, 那么

[AL] = 0 设立成功。

[AL] = 1 设立不成功。

[BP]: [SI] = 原模块的地址。

如果使用 [AL] = 1 取消外部处理模块就没有输出参数。

根据这个规定, P3070 SG.COM 在运行时首先完成内外I/O模块的切换, 并将外部I/O模块驻留在内存。程序如图2所示。

紫金3070打印机和AR2463打印机的打印头纵向由24针组成, 当打印模式设置为24点影象模式时, 各打印针是独立控制, 故可以用点阵形式, 打印各种图形。每打一纵向行由3个字节的数据表示各打印针的击打状态。3070与2463打印机的不同在于每个字节中数据位与打印针的对应关系对于这两种机型是高低相反。当然在设置24点影象打印模式时命令是两种规约如图3所示。对于模块的取屏幕点阵我们可以不必改动。只需用DEBUG程序对P3070.SG.COM中有关部分进行修改, 并适当扩充由于程序长度增加而占用的驻留空间便可达到预期目的。

程序的修改过程如下。

(一) 纵向24点影象打印模式的设置

紫金3070打印机是通过向打印机送控制码 ESC I n1n2+图形点阵数据来完成模式设置。P3070 SG.COM在CS: 1532处为点影象设置。

子程序, 在CS: 1586处为3070正向走纸命令处理子程序。子程序1140为打印机端口访问子程序, 调用1140为向打印机端口送[AL]值,

-debug p3070sg.com

-u cs:1532 1549

1975:1532 BB0005	MOV	BX, 0500	;	设置纵向列机数n=5*256列
1975:1535 B01B	MOV	AL, 1B	;	向打印机端口送
1975:1537 E813FC	CALL	114D	;	(ESC)的ASCII码
1975:153A B049	MOV	AL, 49	;	[AL] = "I" ASCII码
1975:153C E80EFC	CALL	114D	;	将[AL] → 打印机端口
1975:153F 8AC7	MOV	AL, BH	;	
1975:1541 E809FC	CALL	114D	;	
1975:1544 8AC3	MOV	AL, BL	;	将纵向列数送打印机。
1975:1546 E804FC	CALL	114D	;	
1975:1549 C3	RET		;	

-u cs:15b6 15d4

1975:15B6 B01B	MOV	AL, 1B	;	设置n/120"正向换行
1975:15B8 E892FB	CALL	114D	;	向打印机端口送代码
1975:15BB B057	MOV	AL, 57	;	1B 57 n1 n2 n3 n4
1975:15BD E88DFB	CALL	114D	;	
1975:15C0 B030	MOV	AL, 30	;	
1975:15C2 E888FB	CALL	114D	;	
1975:15C5 B030	MOV	AL, 30	;	
1975:15C7 E883FB	CALL	114D	;	
1975:15CA B031	MOV	AL, 31	;	
1975:15CC E87EFB	CALL	114D	;	
1975:15CF B036	MOV	AL, 36	;	
1975:15D1 E879FB	CALL	114D	;	
1975:15D4 C3	RET		;	

按AR2463打印机的命令规约, 设定24点影象模式, 向打印机送控制码“ESC J d₄”, 其中d₄为4位整数, 每行最大点数可由1224点至2448点。在修改时为不影响程序的其他部分, 将

24点影象模式设置子程序改写在程序尾部, 地址起址CS: 17FD, 将正向n/120"走纸设置分别放置在原子程序地址和已经跳过无用的地址CS: 1535以后的空间内。其过程如下:

-A 1532

198D: 1532 MOV MOV BX, 0500H

198D: 1535 JMP 17FD

198D: 1538

-A 17FD

1975:17FD	MOV	AL, 1B	;	将ESC的ASCII码值送AL
1975:17FF	CALL	114D	;	[AL]送打印机端口
1975:1802	MOV	AL, 4D	;	[AL] ← "M"的ASCII码
1975:1804	CALL	114D	;	送打印机端口
1975:1807	MOV	AL, 1B	;	向打印机送ESC
1975:1809	CALL	114D	;	
1975:180C	MOV	AL, 3E	;	向打印机送">"
1975:180E	CALL	114D	;	ESC+ ">" 设置2463为单向打印

```

1975:1811      MOV     AL, 1B      ; 向打印机送
1975:1813      CALL    114D      ; ESC+ "J" 将2463
1975:1816      MOV     AL, 4A      ; 设置为24点影像模式
1975:1818      CALL    114D      ;
1975:181B      MOV     AL, 31      ; 设置打印列数
1975:181D      CALL    114D      ; 向打印机送四
1975:1820      MOV     AL, 32      ; 位数1280
1975:1822      CALL    114D      ; 即640列X2
1975:1825      MOV     AL, 38      ;
1975:1827      CALL    114D      ;
1975:182A      MOV     AL, 30      ;
1975:182C      CALL    114D      ;
1975:182F      RET

```

—A 151B6

```

1975:15B6      MOV     AL, 1A      ; 向打印机端口送
1975:15B8      CALL    114D      ; <SUB>+ "F"
1975:15BB      MOV     AL, 46      ; 设定n/120英寸间距
1975:15BD      CALL    114D      ;
1975:15C0      MOV     AL, 1B      ; 向打印机端口送
1975:15C2      CALL    114D      ; <ESC>+ "T"
1975:15C5      MOV     AL, 54      ; 设定n/120英寸
1975:15C7      CALL    114D      ;
1975:15CA      MOV     AL, 31      ; 送纸
1975:15CC      CALL    114D      ; 向打印机送
1975:15CF      JMP     1538      ;
1975:15D2      NOP              ; n=16
1975:15D3      NOP              ;
1975:15D4      RET

```

—A 15138

```

1975:1538      MOV     AL, 36      ;
1975:153A      CALL    114D      ;
1975:153D      MOV     AL, 0A      ; 向打印机送换行
1975:153F      CALL    114D      ; 回车。
1975:1542      MOV     AL, 0D      ;
1975:1544      CALL    114D      ;
1975:1547      JMP     15D4

```

(二) 数据的纵向行转置

P3070SG.COM是以屏幕上将横向行点阵转置成纵向行数据送打印机。前面讨论了3070与2463打印机的数据与所控制的打印针的对应关系,对转置子程序的修改必须注意2463的点阵数

据与打印针的位置关系,原程序的转置子程序在地址CS: 154A,是用循环左移来实现将数据点阵存入DX, BX寄存器内、程序CS: 14a6处将寄存器中数据进行调整,完成送打印机前的准备工作。

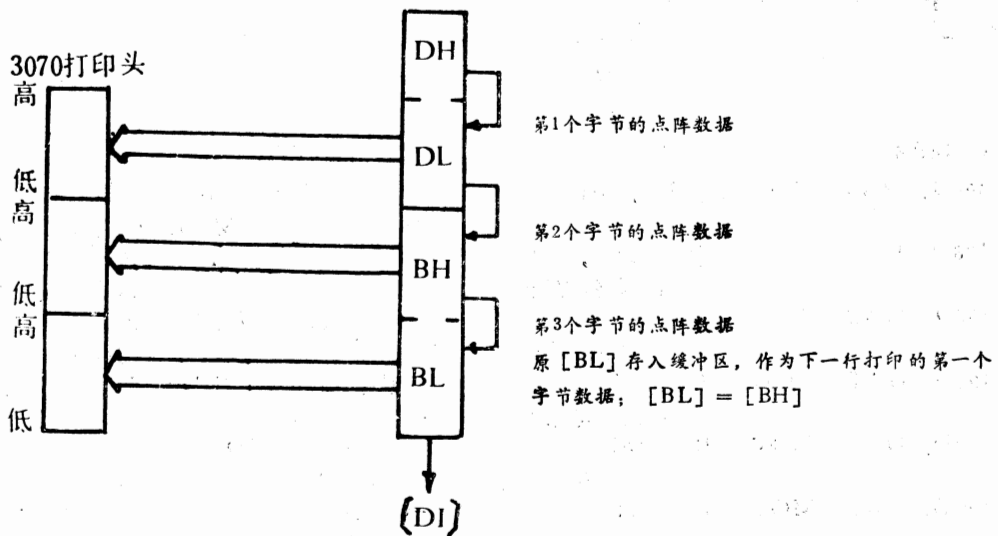
```

-u CS: 154a
198D: 154A RCR AL, CL ;
198D: 154C RCL BX, 1 ; 用循环左移将
198D: 154E MOV AH, BL ; CF内容放入
198D: 1550 RCL DX, 1 ; 寄存器。完成
198D: 1552 RCR AH, 1 ; 数据的纵向
198D: 1554 RCL BX, 1 ; 行转置
198D: 1556 RCL DX, 1 ;
198D: 1558 RET ;

-u CS: 14A6
198D: 14A6 MOV [DI], BL ;
198D: 14A8 MOV [DI+03], BL ; [BL] 存入缓冲区
198D: 14AB ADD DI, +06 ;
198D: 14AE MOV BL, BH ; [BH] → BL
198D: 14B0 MOV BH, BL ; [DL] → BH
198D: 14B2 MOV DL, DH ; [DH] → DL
198D: 14B4 CALL 1559 ;

```

这样BL, BH, DL内容为原BH, DL, DH的内容。



向打印机送数据:

```

-u CS: 1559
198D: 1559 MOV AL, DL
198D: 155B CALL 114D
198D: 155E MOV AL, BH
198D: 1560 CALL 114D
198D: 1563 MOV AL, BL
198D: 1565 CALL 114D
198D: 1568 RET

-A 154A
198D: 154A RCR AL, CL
198D: 154C RCR BX, 1
198D: 154E MOV AH, BH
198D: 1550 RCR DX, 1
198D: 1552 RCL AH, 1
198D: 1554 RCR BX, 1
198D: 1556 RCR DX, 1
198D: 1558 RET

```

修改时将转置部分反其道而行, 用循环右移来完成。过程如下:

-A 151A

198D: 151A

198D: 151A MOV CL, 04

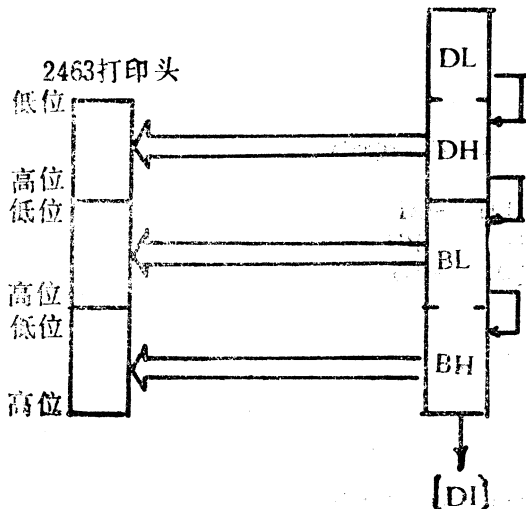
198D: 151C RCR BH, CL

198D: 151E MOV AL, BH

198D: 1520 AND AL, OF

198D: 1522 MOV [DI+03], AL

经过转置的纵向行数据被暂存在两个十六位寄存器中, 如图所示:



第一个字节的点阵数据

第2个字节的点阵数据

第3个字节的点阵数据

原[BH]入缓冲区以作为下一行的第1个字节点阵数据。

-A 14A6

198D: 14A6 MOV [DI], BH

198D: 14A8 MOV [DI+03], BH
; 调整寄存器

198D: 14AB ADD DI, +06
; 中点阵数据

198D: 14AE MOV BH, BL
; 以便向打印

198D: 14B0 MOV BL, DH
; 机送数

198D: 14B2 MOV DH, DL

-A 1559

198D: 1559 MOV AL, DH

198D: 155B CALL 114D

198D: 1560 MOV AL, BL

198D: 1563 CALL 114D

198D: 1565 MOV AL, BH

198D: 1568 CALL 114D

-A CS: 16ED

198D: 16ED MOV BX, 182F ;

198D: 16F0 MOL CL, 04 ;

198D: 16F2 SHR BX, CL
; 单位转换为“节”

198D: 16F4 INC BX
; 增加1节裕量

198D: 16F5 PUSH BX ;

-R CX ✓
CX: 16FD

; 172F ✓

-W ✓

-Q ✓

C>COPY P3070SG.COM P2463
COM ✓

于是你便得到了一个修改好的程序。运行 P2463.COM, 便出现“STAR2463外部屏幕打印模块”提示, 在任何时候, 只要按 Shift + Prtsc 键便可将屏幕图形及字符拷贝至打印机。

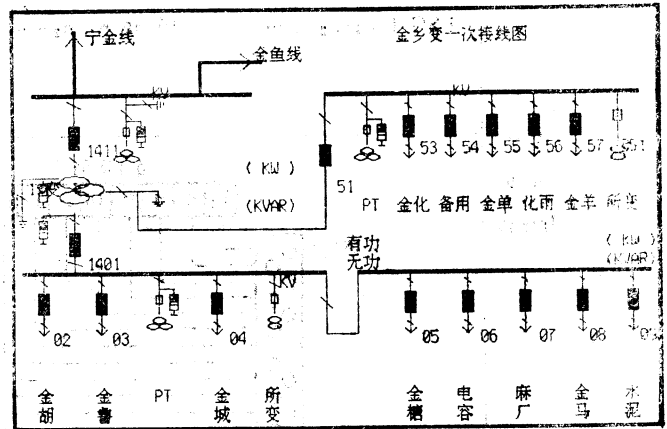
最后修改扩大驻留内存的空间容量并将运行时的提示信息(在程序数据区 CS: 1660)修改成“STAR2463打印机屏幕管理模块”。

debug p3070sg.

u cs:0100

1975:0100 E9C815
1975:0103 0000
1975:0105 0000
1975:0107 0000
1975:0109 0000
1975:010B 0000
1975:010D 0000
1975:010F 0000
1975:0111 0000
1975:0113 0000
1975:0115 0000
1975:0117 0000
1975:0119 0000
1975:011B 0000
1975:011D 0000
1975:011F 0000

```
JMP 16CB
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
ADD [BX+SI], AL
```



-u cs:16cb 1714
1975:16CB B80018
1975:16CE CD10
1975:16D0 BA6016
1975:16D3 B409
1975:16D5 CD21
1975:16D7 B800FF
1975:16DA CD05
1975:16DC 0AC0
1975:16DE 7403
1975:16E0 E9E700
1975:16E3 8CC8
1975:16E5 8EC0
1975:16E7 8ED8
1975:16E9 B449
1975:16EB CD21
1975:16ED BB6016
1975:16F0 B104
1975:16F2 D3EB
1975:16F4 43
1975:16F5 53
1975:16F6 B448
1975:16F8 CD21
1975:16FA 7303
1975:16FC E9F500
1975:16FF B801FF
1975:1702 CD05
1975:1704 8CCD
1975:1706 BE5711
1975:1709 B800FF
1975:170C CD05
1975:170E 5A
1975:170F B80031
1975:1712 CD21
1975:1714 0A0D

```
MOV AX, 1800
INT 10
MOV DX, 1660
MOV AH, 09
INT 21
MOV AX, FF00
INT 05
OR AL, AL
JZ 16E3
JMP 17CA
MOV AX, CS
MOV ES, AX
MOV DS, AX
MOV AH, 49
INT 21
MOV BX, 1660
MOV CL, 04
SHR BX, CL
INC BX
PUSH BX
MOV AH, 48
INT 21
JNB 16FF
JMP 17F4
MOV AX, FF01
INT 05
MOV BP, CS
MOV SI, 1157
MOV AX, FF00
INT 05
POP DX
MOV AX, 3100
INT 21
OR CL, [DI]
```

; 建立工作方式0。可以
; 识别汉字内码
;
; 显示程序运行提示
; “B1053.00外部管理模块”
; 测试模块状态
;
; 已有外部模块转入出错处理
; 转17CA
;
; 无外部模块
;
;
; 释放当前程序段
;
; 提示信息以上为模块区
;
; 单位转换
; 增加16字节裕量
;
; 申请内存
;
;
; 内存不够出错
; 删除测试向量
;
; 新模块段号
; 模块起始地址
; 建立向量地址
;
;
; 驻留程序退出
;

图2 用户自编的外部拷屏模块

	控制码	代 码	功 能
紫金 3070 打印机	ESC I $n_1 n_2 + \dots$	1B 49 $n_1 n_2 + \dots$	发送24点图形 $N = n_1 \times 256 + n_2$, $0 \leq N \leq 2448$ 设定 24点图形及列数, 故接着发送的图形字 节数是 $3 \times N$ 。
AR— 2463 打印机	ESC J $d_4 + \dots$	1B 4A “ d_4 ”	发送24位定影象, 用 d_4 指定每行点数 d_4 值是用4位数字表示, 最大值是2448, d_4 之后是影象字阵的数据

图3 设置24点影象打印模式的命令规约

采用NSC 810A芯片的IBM PC接口

成都水电部勘测设计院电算中心 柳 影

四川德阳市东方汽轮机厂 孙立海

本文提供一种较先进的 IBM PC 的接口设计, 采用可编程—阵列—逻辑芯片 National Semiconductor NSC 810A 接口, 特点是较之采用 Intel 8255 和 Motorola MC 6821 芯片接口, 减少了接口的复杂性和芯片个数, 使接口更简单、更可靠。

(一) NSC 810A 芯片简介

NSC 810A 是一个 40 引脚的双列直插式芯片。它内部有一个 6 位和两个 8 位的双向口、两个 16 位多方式定时器/计数器及 128 字节的 RAM, 它能对几种不同存取方式的 I/O 端口进行程序化, 同时还具有单位的设置和复位能力。芯片引脚见图 1。

(二) 硬件说明

在此, 对 IBM PC 的接口卡、总线及时序就不加赘述了, 而只讨论在 NSC 810 接口设计时应注意的几个问题, 接口硬件电路图见图 3。在

图中 DIP 开关用来设置接口卡的地址, 通过比较器 74LS682 芯片对地址译码。AEN(地址允许)信号和比较器中的信号相比较, 当该信号为有效高电平时, 表示 DMA 控制器已控制了地址总线、数据总线、读命令信号(存储器和 I/O)和写命令信号(存储器和 I/O)。ALE(地址锁存允许)信号用来对 I/O 通道上的设备锁存有效的地址, 当在 I/O 通道上使用 AEN 信号时, 这个信号是在 I/O 通道上的有效地址标识器。

接口设计中采用了一个 16R6 PAL 芯片它只对需要的 26 个寄存器单元进行译码, 16R6 PAL 芯片是一个 20 管脚的双列直插式可编程—阵列—逻辑芯片, 关于该芯片详细说明, 请查阅 PAL 使用手册。图 2 给出了读周期的时序, NSC810A 使用了 23 个寄存器, 且有三个未用, 见表 1。在此要注意的是, 相连空间的 26 个字节被用于 I/O

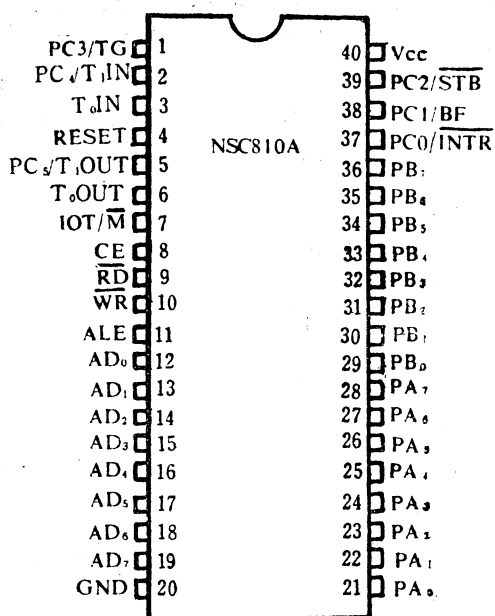


图1 NSC810A芯片输出管脚

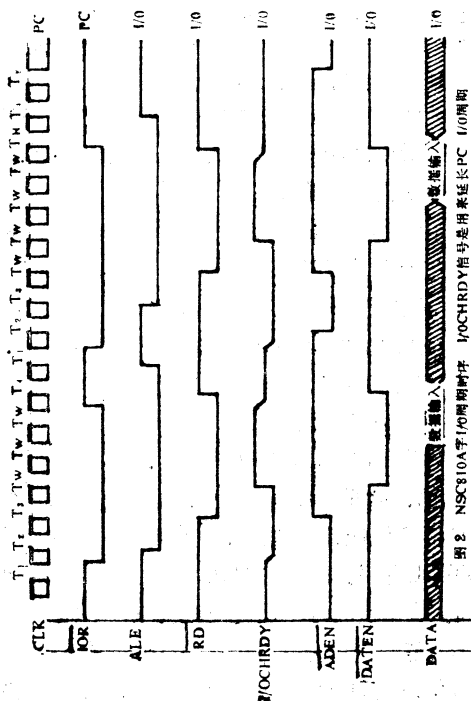


图2 NSC810A字/IO周期时序图

表1 National. NSC810A寄存器编码, 定时器口须先访问低字节, 随后访问高字节, 这个访问次序与PC字I/O周期相匹配

A ₄	A ₃	A ₂	A ₁	A ₀	寄存器	选取方式
0	0	0	0	0	A口(数据)	R/W
0	0	0	0	1	B口(数据)	R/W
0	0	0	1	0	C口(数据)	R/W
0	0	0	1	1	不用	—
0	0	1	0	0	DDR-A口	W
0	0	1	0	1	DDR-B口	W
0	0	1	1	0	DDR-C口	W
0	0	1	1	1	方式一定议寄存器	W
0	1	0	0	0	A口一位清除	W
0	1	0	0	1	B口一位清除	W
0	1	0	1	0	C口一位清除	W
0	1	0	1	1	不用	—
0	1	1	0	0	A口一位设置	W
0	1	1	0	1	B口一位设置	W
0	1	1	1	0	C口一位设置	W
0	1	1	1	1	不用	—
1	0	0	0	0	定时器0(LB)	*
1	0	0	0	1	定时器0(HB)	*
1	0	0	1	0	定时器1(LB)	*
1	0	0	1	1	定时器1(HB)	*
1	0	1	0	0	STOP定时器0	W
1	0	1	0	1	START定时器0	W
1	0	1	1	0	STOP定时器1	W
1	0	1	1	1	START定时器1	W
1	1	0	0	0	定时器0方式	R/W
1	1	0	0	1	定时器1方式	R/W

*写模数寄存器, 读定时器

映象, 所以必须是低地址字节时存取定时器口, 而高地址字节与IBM PC的字I/O操作相匹配。

由于NSC 810A芯片采用了复合的地址和数据线, 所以在I/O存取期间, 必须保证有足够的时间用于芯片上的数据和地址计时, 这就要求读/写脉冲不少于三个周期, 并且ALE至少有两个周期是高电平, 并使用I/O CH RDY信号延迟PC的I/O周期。

NSC 810A是CMOS芯片, 它虽大大降低了功耗, 但同时带来了一个问题, 它必需的是CMOS逻辑电平, 然而, IBM PC总线使用的是TTL电平。通常, CMOS输入高电平需要的是3.5伏以上的电压, 而在满负荷时的一个LST-TL(低功率肖特基TTL)输出电平需要的是2.4伏以上的电压。鉴于上述情况, 故所有接至

NSC 810A的总线信号必须经过缓冲。我们建议缓冲器最好是采用HCT系列的芯片, 这是由于它们输入和TTL相兼容, 输出与CMOS相兼容。但是, 74HCT245的驱动能力仅大约是74LS245的百分之三十五。为增加其驱动能力可采用两个缓冲器芯片串联的方法, 即74LS245出来的数据总线与74HCT245芯片相联接, 然后再接至NSC 810A芯片。此外, 在PAL芯片的输出线上, 采用了一个2.2KΩ的上拉电阻, 通过该电阻提高逻辑高电平输出电压, 使TTL输出能直接驱动NSC 810A(见图3所示)。

高电平激活NSC 810A的芯片选择信号, 74HCT238用于从地址线A10到A12来的芯片选择译码。

(三) 软件说明

因为Intel 8088处理器使用专用指令(IN, OUT)来进行I/O空间的存取, 所以使用的语言中必须有存取这些I/O指令的功能, 否则实施I/O是很困难的。本文中我们采用了Turbo Pascal语言, 该语言使用两个“数组”Port [0...\$3FF] 和Portw [0...\$3FF] 来建立I/O指令的集合, 它分别作为字和字节存取I/O单元。因此, 从I/O单元300H读一个字节的指令是K:=Port [\$300], 用类似简单的指令也可向一个端口写。例如, 若要在地址300H, 存取接口卡上第五个NSC 810A芯片的定时器, 使用Turbo Pascal语言可定义如下:

```
const
CARD=$300;
TIMER-1=$012;
NSC-5=$2400;
```

若要用该程序读一个16位的值, 可用:

```
timer-Value:=
Portw [CARDor NSC-5or TIMER-1];
```

在此, 符号的实际值是与编程者无关的。

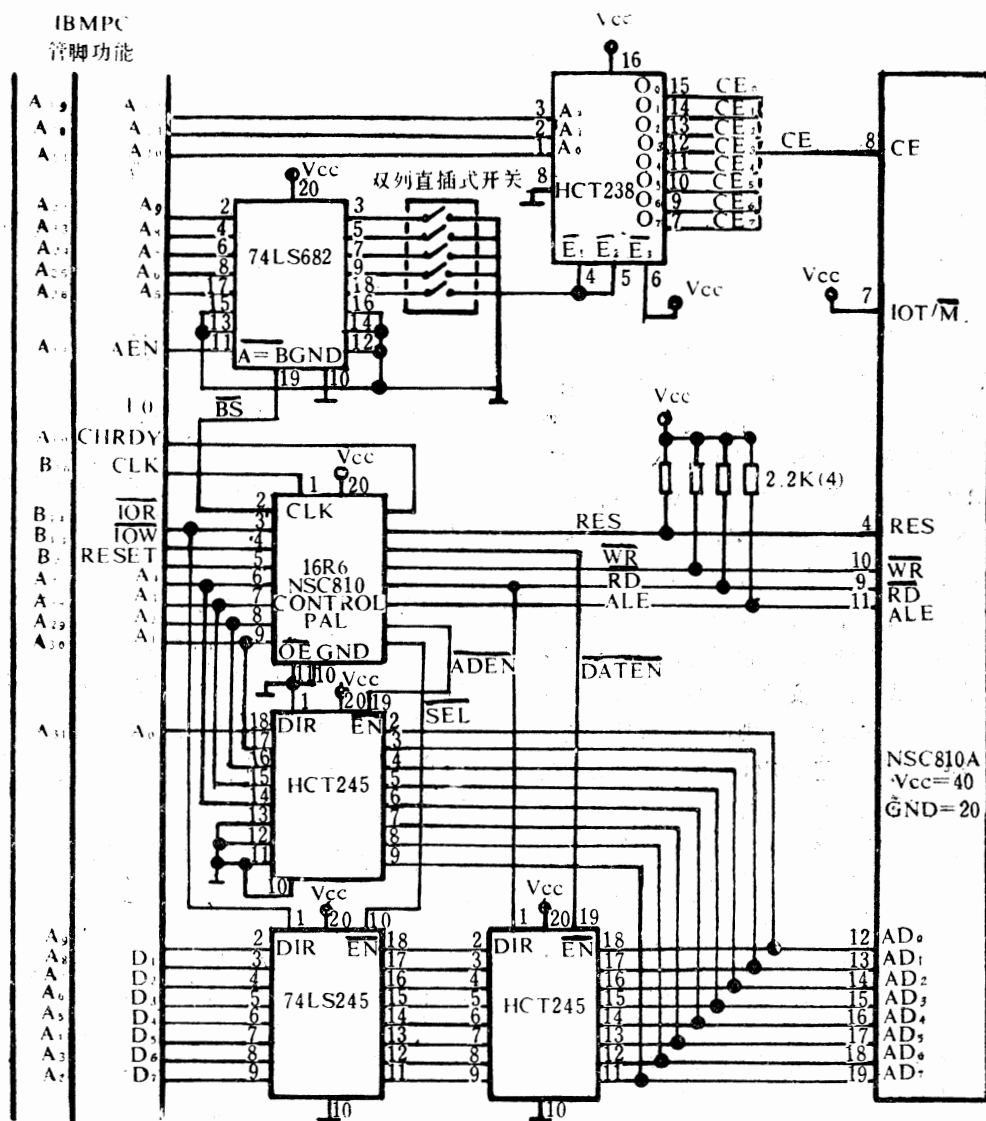


图3 NSC 810A接口硬件电路

本刊今年栏目设置

- 1、软件篇；
- 2、硬件篇；
- 3、系统（软、硬件兼具）篇；
- 4、维修（含维护）篇；
- 5、信息（含综述、编译、短讯等）篇；

目前来稿中1稍多，2，3，4，5偏少，新颖、短小、实用的2，3，4更
显少，热诚欢迎广大作者根据稿约要求积极来稿。

本刊编辑部

关于软盘零磁道破坏后的数据修复

解放军总后基地指挥部自动化站 阮高华

摘 要

本文针对软盘零磁道破坏后,其盘内文件数据无法读写的问题,进行了分析解剖,并详细介绍了其数据修复的方法步骤。

我们在使用已有存贮信息的旧软盘时,经常会出现软盘零磁道破坏现象;发生这种情况时,用DIR或其它显示命令,系统均给出“扇区未找到, A驱动器读错误”,并给出退出,再试、强制执行三种选择信息(SECTOR NOT FOUND ERROR READING DRIVE A, ABORT, RETRY, IGNORE),如果敲除 A以外的键,系统均重复上述信息,只有敲A键退出,因而对软盘上的文件进行读写修改等一系列操作均失效。以前我们遇到这种情况时,通常是弃之不用或重新格式化再建文件,但假如被破坏的软盘上装有大量的文件数据,则这种浪费是不能容忍的。最近,我们使用DEBUG调试程序对零磁道被破坏后的磁盘进行了分析解剖,发现文件数据基本上是完好的,如果我们能将这些未遭破坏的文件数据(或稍加修改后的数据)移至另一张零磁道完好的软盘上,问题就迎刃而解了。而DEBUG调试程序正好提供了这种功能。以下是我们对零磁道被破坏后的磁盘进行数据修复所采取的方法步骤:

一、修复盘文件目录,即将零磁道被破坏的软盘上的文件各部分移至零磁道完好的软盘上,具体分三步进行:

1. 先格式化一张软盘(即零磁道完好之软盘,以下简称新盘),预备作数据转储之用。
2. 将零磁道遭破坏的磁盘(以下简称坏盘)插入A驱动器,用DEBUG装入命令(LOAD)把盘文件目录装入内存中:

```
C>DEBUG
-L ES:0100 0 5 6
```

上面L命令表示从A驱动器的盘上装入数据,并把数据存放在以ES:0100开始的内存中;从A盘的第5扇区开始,传送6个连续扇区的数据。

用转储命令(DUMP)显示盘文件目录内容,如果发现有文件名破坏现象,即用输入命令

(ENTER)进行修改。

```
-D ES:0100
-E (地址)
```

上面E命令所带的地址参数是需进行修改的字节地址。

3. 将新盘插入A驱动器,用写命令(WRITE)将内存中调试过的盘文件目录信息写到新盘上:

```
-W ES:0100 0 5 6
```

上面W命令表示从ES:0100单元开始的数据写到A驱动器的软盘中,由第5扇区开始,连续写6个扇区的数据。这里要注意上面W命令所带的四个参数一定要与前面L命令带的参数相一致。否则会引起数据混乱。至此,盘文件目录部分即被移至新盘上了。

二、修复文件内容,即将坏磁盘中的各个文件的数据部分移至新盘上,可按以下二个步骤进行:

1. 将坏软盘插入A驱动器,根据盘文件目录,先用命名命令(NAME)将第一个文件装入内存中:

```
-N A: (文件名)
-L
```

上面L命令不必指定参数,表示从默认驱动器的盘上,装入由NAME命令指定的文件,并把它存放在ES:0100单元开始的存贮器中。

2. 将新盘插入A驱动器,将内存中的第一个文件内容写到新盘上。

```
-W
```

上面W命令不必指定参数,表示把从ES:0100单元开始的文件以文件说明FILESPEC写到默认驱动器(即A驱动器)的软盘上。这样,第一个文件就完整地转储到新盘上了。尔后同理操作,将坏磁盘上的所有文件一一移至新盘上。数据全部修复完以后,即可对新盘上的文件进行读写修改等各种操作了。

PDP11 PSX11M 操作系统中非正常文件的恢复处理

中国科学院福建物质结构研究所

结构化学开放实验室衍射技术组 施剑秋 吴镛金

摘 要

本文涉及在PDP11系列计算机RSX11M操作系统中处理外来的ASCII磁带文件以及由于设备故障或操作错误导致文件不能读出或被破坏等几种情况下, RSX11M几个系统实用程序和几个我们编制的程序的灵活使用。对结晶学SDP用户和其他计算机用户有一定的启发性。

一、如何读外来的ASCII文件磁带

问题的提出,是如何读外来的ASCII文件磁带。不同的操作系统所建立的磁带上的文件格式绝大多数是不一样的。因此,外来的ASCII磁带文件如果用RSX11M的实用程序BRU, FLX, PIP等直接读出时总是扭曲的,不能用的。但RSX11M有个实用程序DMP,它既可按文件方式(包括按文件中的记录方式)又可按块方式卸出磁带上的内容。特别是当用户完全不了解外来带的属性(块大小,记录长度)和情况(文件数目,文件名等)下,用DMP块方式卸出磁带上的内容是大有裨益的[1]。用以下命令:

```
>RUN $DMP
```

```
DMP>TI:/AS=MM:/BL:1:10.  
/RW(按块卸出)
```

就可了解外来带的属性,就有可能编制自己的程序来读外来带了。

用DMP按块方式将磁带内容卸到盘上,可得到ASCII的文件FILDMP、DMP。这种文件是可读而不可用的。我们首先是写一个程序TRSDMP,将FILDMP、DMP转换成可用形式。但这种读法有三个不可取之处:1. DMP卸出文件FILDMP、DMP所占的磁盘量很大,2. 这种转换费时,3. 由于外来的ASCII磁带文件格式是五花八门的,所以TRSDMP程序也不可能编得通用。因此我们考虑能否直接从带读盘,一次成型呢?但是磁带驱动程序不支持直接的FORTRAN I/O,我们利用了系统库和结晶学库中几个磁带I/O子程序编制了一个通用的读外来ASCII带程序READMT。它可以直接读

大部份定长的ASCII磁带文件,并有着在磁带上根据指定给予定位(跳过若干个文件或跳过若干块)的功能。这种读带方式既省时又省磁盘容量。

二、READMT程序用于非正常文件的处理

问题的意义不仅在于READMT程序能读ASCII磁带文件,而且在于该程序的灵活使用。有一个时期PDP 11/70机的磁带机故障。用系统实用程序FLX, PIP读带上结晶学数据文件CAD4、DAT时总出现ERROR DURING DIRECTORY I/O的错误,数据读不出来。用DMP卸出时发现Logical block 0, 000001—Size 14, bytes(文件头)有Parity Error(奇偶错误),但Logical block 0, 000002—Size 512, bytes以下(文件内容本身)没有这种奇偶错误。说明故障仅发生在文件存取方式,而不会发生在块存取方式。而READMT程序可以按块存取方式工作。于是我们就用READMT程序将CAD4、DAT送到11/70上,绕过了磁带文件方式的ERROR DURING DIRECTORY I/O错误。

另一次,一个用户磁带在11/70上联机时,一个突然的干扰后,磁带首部被破坏,带上的文件不能用FLX读出,而带上要读的是非ASCII的数据文件。虽然READMT程序能按块读出,但却不能处理非ASCII文件。怎么办?前已述及,READMT程序有着在磁带上正确定位的功能。这样,我们就利用READMT程序跳过磁带头上被破坏的坏区后,参照原磁带文件目录,

将磁带定位在要存取的文件之前,然后用正常的FLX命令将非ASCII文件读出。

这个例子启发我们,既然带头被破坏,数据仍可读出,那末磁带不慎被清零(FLX/ZE)后带上的文件是否仍可得到恢复呢?答案是肯定的。磁带被清零时,只是由磁带文件系统在带首连续写上三个EOF(End of File)标记,但带上的信息在没有被重写上之前必然依旧存在。果然,在清零之后,通过READMT程序,或该程序配合FLX就能读出被清零带上的ASCII的或非ASCII的文件。

以上涉及的是定长记录文件的处理,下面述及变长记录文件的恢复处理。

三、FLX 操作失误后文件的恢复处理

正确的结晶学数据文件CAD4、DAT传送命令应该是:

```
>FLX DR:=MM:[g,m] CAD4、
DAT/IM:64.
```

但是某个用户在整理磁带时用了错误的命令:>FLX DR:=MM:[g,m] CAD4、DAT(1)然后将磁带清零。在这种情况下,FLX传送开关缺省到FA(格式化ASCII)格式。在这种DOS-11到FILES-11FA格式传送中,CR

—LF符从记录的尾部被除去,且所有的NULL字符,Vertical tabs字符均被删去。又因为FA不加:64数值,FLX产生变长记录(2)。这时若根据第二部分所述,文件仍可读出。但用户一错再错,又使用了以下命令:

```
>FLX MM:/DO=DR:CAD4、DAT
/IM:64./RS(2)
```

```
和 >FLX DR:=MM,CAD4、DAT/IM:
64.(3)
```

这样,磁带被重写,正确的文件已不复存在。用PIP列出(3)以后的数据文件,成锯齿形(见图1。)这时无无论用哪一个系统实用程序或READMT均不能将其矫正。

由于此时文件已是变长记录,若用DMP按块卸出观察显然不够明确,而按记录卸出更便于观察。

```
DMP>TI:/AS=DR:CAD4、DAT/RC
```

卸出格式如图2,记录大小仍是64.字节,但记录中杂乱无规地带有CR,LF字符。也就是说该文件是假的定长记录,由(2),(3)操作所致,真的变长(无规)记录,由(1)操作所致。而正常的CAD4、DAT文件按记录卸出,应如图3.所示。图2、3一比较,关键问题就是顺序

```
21      MD-15T
.9 1.3 5
.9
22 1.00 23.00 0.50 0.35 1.50 1.05 6 0
.500 0.300 4 30
0 0 0
23 3 BISECT 37 0.00 1
```

图1

```
DUMP OF DL1:[101,101]ERROR.DAT;1 - FILE ID 63,163,0
RECORD NUMBER 00. - SIZE 64. BYTES
```

```
000000      ^J      2 1      1 M 0 - 1
000020      5 T
000040
000060      1 7 3      4 ^M ^J . 0      1 . 3 5
```

图2

DUMP OF DLO:1301.1011CAD4.DAT:1 - FILE ID 165,113.0
RECORD NUMBER 00. - SIZE 64. BYTES

```
000000  ^J  2 1          1 K H 1 6 A
000020
000040
000060  7 3  4 . 0  1 . 3  5 . 9  ^M
```

图3

```
21      1MB-15T                      173 4.0 1.3 5.9
22  1.00 23.00 0.50 0.35 1.50 1.05 6 0.500 0.300 4 30 0 0 0
23 3 8ISECT 37  0.00 10.00  0.002778 -0.035847 -0.000393
24 ZIGZAG HLK  -28 -28  28  0  0  13  0  0  34
25 1 34 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
26 3600 0.030 0.850 300 0.150 0.5862111 0.6432642  0
    1-10 5 19H*SR*S 15.48 46.17  2.65 38.24 1 1.17 786
```

图4

读出错误数据后如何去掉错误数据中杂乱无章的CR, LF字符, 然后将62个非CR, LF字符顺序排列, 前头加上LF, 后头加上CR后写进新的文件中。程序DATCOR解决了这个问题。矫正后的文件如图4。这种处理方式可推而广之, 处理任何外来的变长的ASCII磁带文件。在不少场合, 外来的源程序文件录在带上是变长记录的。可以用FLX加FA:n, 开关读到盘上, 对DATCOR程序略加修改, 即可将其矫正。DATCOR的处理是基于记录的。还有一种处理也基于记录方式。

四、磁盘设备奇偶错误下文件的恢复处理

当盘上有大量的I/O作业时, 易产生一种错误。ERROR 39 (ERROR DURING READ), FCS-4 (PARITY ERROR ON DEVICE)。任务在写一个文件时不报出错误, 但以后读这个文件时, 就报以上的错误。有一次刚刚收集完的数据文件CAD4.DAT出现这类错误, 它不能用FLX, PIP, DSC甚至DMP读出。这种错误的起因也并非由盘上的坏块所致。因为几次运行实用程序VFY加/RC(读检查)检查该文件时, 出错的VBN和LBN都不尽相

同, 这就排除了坏块的可能。VFY/RC检查(一次读入3~4块时)会报两种类型的错误:

(1) ERROR STARTING AT VBN n1, n2 LBN n1, n2—ERROR CODE—4 和
(2) ERROR AT VBN n1, n2 LBN n1, n2—ERROR CODE—4 根据VFY使用说明[3], 如果仅有(1)类错误信息而无(2)类错误信息出现, 错误可能是多块读操作失败, 但一块一块地可能可读。所以, 当再用VFY/RC:1读检查时, (1)类错误信息就减少了, 但(2)类错误信息依然存在。VFY说明还指出, 如果列在(2)类错误信息中不可读块的VBN在block-used-count之外, 文件的数据部分仍可读。也就是说, 数据信息在盘上仍存在, 只不过按块读不行了。而系统实用程序FLX, PIP, DSC等均是以为块为单位进行I/O的, 故对此错误无能为力。于是我们就编了个程序DATHAN, 将该错误文件按记录读出, 再写进新的文件, 解决了这个问题。

以上是我们在长期的系统管理中遇到各种不正常文件时的心得体会和处理方法。现归纳成文, 以期对结晶学SDP用户或其它计算机用户特别是PDP用户有所启发, 从而收到抛砖引玉的效果。(需要源程序清单者可同作者或本刊联系)

KOSMIC KH-16 内存故障的检修

江苏省无锡市港口机械厂微机室 吴 达

计算机内存故障是常见而多发的,只要多实践,边摸索、边积累经验。利用“插拔法”、“替换法”、“交互法”、“试探法”等多种方法查出引起故障的 RAM 芯片,换后即能轻而易举地修复。

KOSMIC KH-16 是 IBM PC 兼容机,其内存配置为 512KB;其中多功能板上有 256KB,分成 4 组;分别是“BANK1”、“BANK2”、“BANK3”、“BANK4”。扩充板上也有 256KB;也分成 4 组,分别是“BANK5”、“BANK6”、“BANK7”、“BANK8”。每组由 9 片,64K×1 位动态随机存储器芯片组成,每组有 1 片为奇偶校验位,其余 8 位为二进制 0 到 7 位,每组存储器芯片的存储容量为 64KB。RAM 芯片的排列,扩充板与多功能板上稍有不同;屏幕显示与 RAM 芯片实际对应位置二者有很大的差异。

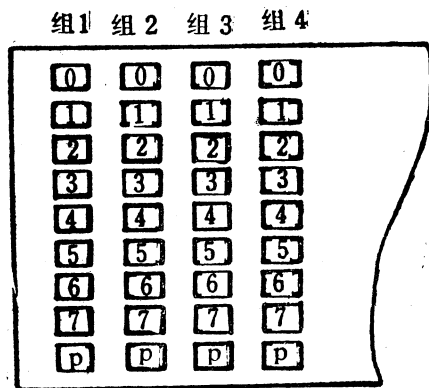
开机后,主机就进行自检。正常时显示器屏幕显示如下。如图<1>

```
KOSMIC KH-16
Press [Alt-Ctrl-Del] will skip RAM
Test
RAM Test
Bank 1: 0 00000000
Bank 2: 0 00000000
Bank 3: 0 00000000
Bank 4: 0 00000000
Bank 5: 0 00000000
Bank 6: 0 00000000
Bank 7: 0 00000000
Bank 8: 0 00000000
```

图<1>

正常为全“0”,只要在“Bankx”中出现一个或几个“X”,表示存储器芯片发生故障或是动态特性不一致,主机自检就中断锁死。

一、若是“X”出现在“Bank1”至“Bank4”中,就要检查多功能板上的 RAM 芯片。RAM 芯片在多功能板上的排列及屏幕显示与 RAM 芯片的对应位置见图<2>。



RAM 芯片在多功能板上的排列

```
Bank 1: 0 0 0 0 0 0 0 0
Bank 2: 0 0 0 0 0 0 0 0
Bank 3: 0 0 0 0 0 0 0 0
Bank 4: 0 0 0 0 0 0 0 0
```

RAM 芯片对应位置: P 7 6 5 4 3 2 1 0

屏幕显示与 RAM 芯片对应位置

图<2>

五、参考文献

[1]、FILE DUMP UTILITY (DMP)
Digital: 《PDP 11 RSX-11 Utilities Manual》

15-1~15-6

[2]、FILE TRANSFER PROGRAM
(FLX)

同上, 5-1~5-16

[3]、FILE STRUCTURE VERIFICATION UTILITY (VFY)

同上, 13-1~13-14

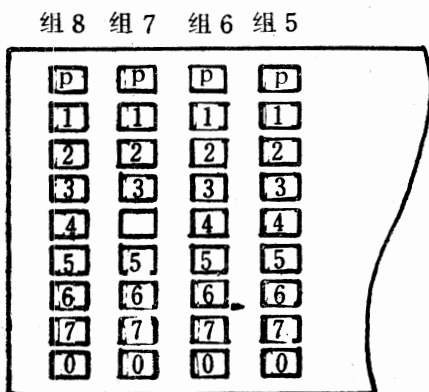
在多功能板上,主机依次从组1到组4,先检测本组的奇数位1、3、5、7。当奇数位RAM芯片全好时,再检测本组的偶数位0、2、4、6。若是奇数位或奇、偶数位的RAM芯片都有故障时,屏幕上只显示奇数位的“X”。

①例:屏幕显示Bank 2: 0 000000 X 0 “X”在组2位1,但不一定光是组2位1的RAM芯片有故障,组2位0、2、4、6也有可能也有故障。检修时,先用好的RAM芯片替换组2位1,若故障并没有消失,但“X”的位置变成偶数位时,可依次再替换屏幕指示的偶数位。若是“X”在位3、5、7时也需注意到这一点。

②若是屏幕上显示偶数位的RAM芯片有故障时,可按多功能板上RAM芯片的排列位置,准确找到有故障的芯片加以排除。

二、若是“X”出现在Bank5至“Bank 8”中,就要检查扩充板上的RAM芯片。RAM芯片在扩充板上的排列及屏幕显示与RAM芯片的对应位置见图<3>。

在扩充板上,主机按组5到组8先检测本组的偶数位0、2、4、6。当偶数位RAM芯片全好时,然后再检测奇数位1、3、5、7。若偶数位或



RAM芯片在扩充板上的排列

Bank 5: 0 0 0 0 0 0 0 0
Bank 6: 0 0 0 0 0 0 0 0
Bank 7: 0 0 0 0 0 0 0 0
Bank 8: 0 0 0 0 0 0 0 0

RAM芯片对应位置: P 0 1 6 3 4 5 2 7

屏幕显示与RAM芯片对应位置

图<3>

奇、偶数位的RAM芯片有问题时,屏幕只显示偶数位“X”。

①例:屏幕显示 Bank 5: 0 X 0000000, “X”在组5位0,但不一定只是组5位0芯片有故障,组5位1、3、5、7有可能也不好。检修时,先用好的RAM芯片替换组5位0;若故障没有消除,但“X”变成奇数位时,可再用好的RAM芯片替换屏幕指示的奇数位。若“X”在位2、4、6时同样也要考虑到这一点。

②若屏幕上显示奇数位的RAM芯片有故障时,可按扩充板上RAM芯片排列位置,准确找到有故障的芯片,加以替换。

三、本机型的奇偶校验位在主机内存自检时是不检测的(我曾多次试过把奇偶校验位的RAM芯片都拔去“置0”或故意把有的奇偶校验位的RAM芯片的插脚不插进插座,结果自检时并不出错),偶尔有几次到内存检测完毕,在调操作系统或运行应用软件过程中才出现错误信息:

PARITY CHECK ERROR

Do you Want Resume, Press “F1”

KEY

这时只要用好的RAM芯片依次替换各组的奇偶校验位“P”即可。

若是手头没有RAM芯片备件,可采用应急措施:

①把奇偶校验位置“0”,用拔下的奇偶校验位的芯片替换出错的RAM芯片。

②可把以前主机自检内存出错时,替换下来的RAM芯片作为备件使用(在专门检测仪器上测试出来电气性能失效的,当然不能再用),你往往会获得意外的成功。RAM芯片在某个位置上不适用,但换个位置又工作得很正常,这是由于RAM动态特征不一致的原故。

根据我的经验,若是某组RAM芯片中坏一片或几片时,最好能换上和主机上原有RAM同型号的芯片,最好把一组RAM芯片都换掉,(换下来的可作备件保存)以保持RAM动态特性的一致性和匹配性。若是更换组件后故障仍没有排除,那要考虑RAM印刷板有无问题,并仔细检查RAM插座有无损坏、弯曲、碰线、接触不良等等,一一排除后内存故障肯定会迎刃而解。

一种硬盘故障的修复方法

平顶山高压开关厂计算中心 苗沛淋

故障现象：一台IBM-PC/XT机，利用硬盘不能自举，但利用软盘启动机器后，关于硬盘的操作可以进行。不过，在读写时，常常出现：

Data error reading drive C

Abort, Retry, Ignore?

和 Data error writing drive C

Abort, Retry, Ignore?

打入“R”，多数情况下不能成功。用检测盘检测，发现硬盘发生了1703号错误。这意味着硬盘不能做为系统盘使用了，只能用做数据盘。这对用户来说，损失太大了。

故障原因分析：发生此故障，是因为UPS失灵而未及时修理，市电又因种种原因经常停电，造成硬盘工作时掉电，读写磁头不能复位，再开机时磁头在盘面上机械移动造成盘面划伤，将第零号柱面的某部分划伤，使DOS引导程序不能正常写入，因而不能自举。要解决此问题，只需避开有划伤的柱面，重新选择引导磁道，对硬盘进行格式化即可。

修复方法及步骤：

经过试验，我们采取以下方法进行。

1. 用FDISK程序对硬盘的DOS分区重新划分，以第3柱面为开始柱面，第304柱面为结束柱面，然后退出FDISK程序。

2. 利用FORMAT命令对硬盘进行格式化，打入：A>FORMAT C:/S/V <CR>格式化完成后，硬盘可用空间约9.99兆字节。

3. 利用COPY命令将CCDOS2.20拷贝到硬盘上，使硬盘上有下列九个文件：COMMAND.COM, FILE1.EXE, CCCC.EXE, CCLIB, ANSI.SYS, CONFIG.SYS, NEW9P.EXE, AUTOEXEC.BAT。

4. 完成上述三项任务以后，取出软盘驱动器里的软盘，启动机器（冷、热启动均可），即可进入CCDOS2.20操作系统了。

到此，系统即可由硬盘启动了，且工作完全正常。利用本方法修复的硬盘，虽然容量减少了一部分，但权衡之，仍是得大于失，不失为一种好方法。

控制计算机图形质量的三个因素

浙江医科大学生物医学工程系 王性卫

当前，越来越多的计算机辅助设计(CAD)软件被广大计算机用户使用。CAD软件中一般均采用了图形编程技术，因此了解一些与图形编程有关的知识是大有裨益的。本文主要对控制IBM个人计算机及其兼容机图形质量有关的因素作一介绍。

一般而言，有三个因素控制个人计算机图形的质量，即图形适配器、监视器及程序语言。其中图形适配器是最重要的因素。

• 图形适配器

个人计算机一般采用下列主要的五种图形适配器：VGA, EGA, MCGA, CGA及Pcjr

显示子系统。表一表示IBM PC及其兼容机使用的各种图形适配器的显示方式。

CGA, CGA即彩色/图形适配器(Color/Graphics Adapter)，是IBM PC及兼容机广泛采用的一种图形适配器。CGA能产生320×200 4包和640×200 2色屏幕显示方式。所谓320×200指的是横向有320个象元，纵向有200条光栅扫描线；象元则是在显示屏幕上能控制的最小图象单元。

由于CGA在四色显示方式时分辨率很低，只有320×200，而在640×200高分辨率时只有两种色彩可控选择，因而不适合于高级图形程序。

表一: IBM PC 机及其兼容机图形适配器的显示方式

	320×200 4色	640×200 2色	160×200 16色	320×200 4色	320×200 16色	640×200 4色	640×200 16色	640×350 16色	640×350 2色	640×480 2色	640×480 16色	320×200 256色
CGA	+	+										
EGA	+	+			+		+	+	+			
MCGA	+	+								+		+
VGA	+	+			+		+	+	+	+	+	+
PCjr sub- system	+	+	+	+	+	+						

MCGA: MCGA是多彩色图形适配器(Multi Color Graphics Array)。这种图形适配器用于CPU为8086的IBM PS/2系列30型和25型中。MCGA除了能产生标准的320×200 4色和640×200 2色显示方式外,还能产生640×480 2色及320×200 256色显示方式。

EGA: EGA是增强型图形适配器(Enhanced Graphics Adapter)。EGA除了能产生CGA的二种显示方式外,还能产生其他四种图形方式:320×200 16色、640×200 16色、640×350 16色及640×350 2色显示。

VGA: VGA是视频图形适配器(Video Graphics Array),它用于CPU为80286和80386的IBM PS/2系列的50型、60型中。VGA除了能工作在CGA、MCGA和EGA的显示方式外,还能产生640×480 16色的屏幕显示方式。

PCjr视频子系统(PCjr video subsystem):它是由部分CGA和部分EGA构成的。除了能提供标准的CGA显示方式外,PCjr视频子系统还能提供160×200 16色,320×200 4色,320×200 16色及640×200 4色工作方式。其中320×200 16色和640×200 4色工作方式是高级图形程序中最有用的。

• 显示器

图形适配器确定后,与之适配的显示器的类型就限定了适配器的输出。目前,主要有四类显示器用于IBM PC机及其兼容机。它们是标准彩色显示器(SCD),增强型彩色显示器(ECD),模拟视频显示器及可变频显示器。

标准彩色显示器(Standard Color Dis-

play, SCD): SCD的最大分辨率为640×200,它能生成最多16种彩色。在配用CGA时,标准彩色显示器可以显示320×200 4色和640×200 2色;如果配用EGA,还可以提供320×200 16色和640×200 16色显示。由于标准彩色显示器的光栅扫描线上有200条,因而不能提供640×350 16色显示。

增强型彩色显示器(Enhanced Color Display, ECD)的最大分辨率为640×350,能产生最多16种彩色。这种显示器能从64种色彩的调色板中选择16种可显示的彩色,这是与标准彩色显示器的不同之处。增强型彩色显示器在配用CGA时,可显示320×200 4色及640×200 2色;如果配用EGA则除了能显示以上两种方式外,还可显示320×200 16色,640×200 16色及640×350 16色。

模拟视频显示器(Analog Display)用于IBM PS/2系列中。不同于采用数字技术的SCD和ECD,这种显示器采用一种连续可变的信号来产生更多的色彩并能提供更细微的灰度等级。配用了VGA后,这种显示器可以显示所有的CGA和EGA的图形显示方式;它还能提供320×200 256色和640×480 16色的显示。

可变频显示器(Variable Frequency Display)能显示数字的和模拟的信号。这种显示器可以显示所有CGA, EGA, MCGA和VGA的图形方式。它甚至还可以提供诸如752×410 16色的非标准显示方式。

• 程序语言

不论个人计算机采用何种图形适配器和显示器,为了最终生成图形,还必须依赖程序语

ESPRIT: 欧洲信息技术研究与开发的战略计划

北京工业大学计算机科学系 刘椿年

摘要 本文简要介绍欧洲信息技术研究与开发的战略计划 ESPRIT 的组织实施、和迄今已取得的效果, 供国内计算机科技界参考。

八十年代以来, 西欧、北美、日本和我们国内均在信息技术领域开展了相当规模的高技术研究与开发活动, 这些活动都是国家级的甚至是国际区域性的联合, 旨在增强九十年代以至下世纪的高技术产品竞争能力。

西欧联合进行的 ESPRIT 计划有其特色, 其规模之大, 理论与实践相结合的策略, 迄今已取得的效果等均值得我们加以认真研究。本文根据最新资料, 对 ESPRIT 计划的预备阶段, 第一阶段和目前正在进行的第二阶段的方针、组织、实施情况和效果作一介绍, 并提出作者本人对 ESPRIT 计划的特点的看法。

ESPRIT 计划的准备阶段 (1979—1983)

1979年欧洲共同体开始了一项有限目标的在信息领域的研究计划, 重点是微电子学。这一研究计划导致了1982年底开始的 ESPRIT 计划的预备阶段。ESPRIT 是英文“欧洲信息技术研究与开发的战略计划”的缩写。这一预备阶段进

行了一年多, 其结果令人鼓舞, 主要的成果是使欧洲的信息技术工业界为区域性的全面合作做好了准备, 而这一合作就是通过 ESPRIT 计划来实施的。

ESPRIT 计划的第一阶段 (1984—1988)

ESPRIT 计划第一阶段共投资15亿欧洲货币单位, 主要目的有三:

1. 推动欧洲信息技术工业界在预赛性的研究与开发方面的合作;
2. 为欧洲信息技术工业界提供它们在90年代初所需要的基本技术;
3. 为标准化工作铺平道路。

所谓预赛性, 是指 ESPRIT 计划研究的内容带有一定的超前性质, 但它的效果应在数年内体现出来, 即提高欧洲信息领域高技术产品的竞争力。

第一阶段通常简称为 ESPRIT I。ESPRIT I 在1984年, 1985年和 1986 年分别征集在微

言。目前有许多种语言可供用户选用, 我们不可能在此逐一介绍。这里, 只就常见的不同版本的 BASIC 语言的图形能力作一简介。

IBM BASICA 2.00, 2.10, 3.00, 3.10 及与之兼容的其他 BASIC 版本可支持标准的 320 × 200 4色和 640 × 200 2色显示方式。GW-BASIC 除了支持 CGA 的显示方式外, 还能支持 160 × 200 16色, 320 × 200 4色, 320 × 200 16色及 640 × 200 4色的显示方式。

IBM BASICA 3.21, Microsoft 的 ϕ nich BASIC 2.0 和 Borland 的 Turbo BASIC 1.0 则支持 EGA 的显示方式。如果采用了 EGA,

就应当选用上列三种版本的 BASIC。否则, EGA 只能当 CGA 用。

IBM PS/2 系列机中采用的 IBM BASICA 3.30 能支持 PS/2 的 VGA 以及所有 EGA 和 CGA 显示方式, 但不能支持新版本的 MCGA 和 VGA 的图形方式。Turbo BASIC 1.0 是第一个支持 VGA 的 640 × 480 2色和 640 × 480 16色的 BASIC 版本; Quick BASIC 4.0 则能支持 VGA 和 EGA 的显示方式。

了解了控制个人计算机图形质量的三个因素后, 我们就可以根据不同的需要, 合理选择硬件配置及程序语言, 以便更好地实现编程的目的。

电子学、信息处理系统(软件技术,高级信息处理技术)和信息技术应用(办公室自动化,计算机集成化制造)三个方面的科研计划申请。每一个申请必须是多国合作的,还必须是学术界与工业界合作的。尤其每一个申请必须有来自至少两个欧洲共同体成员国的工业界单位参加。一旦申请获得通过,就可以从欧洲共同体获得50%的研究经费,其余一半由申请单位自筹。研究小组的成员单位共享研究成果。

共有1000多份科研计划申报了上来。ESPRIT I 从各成员国邀集专家进行独立的评议。评议标准极严,大约有五分之四的申请被拒绝,其中有些计划还是具有相当高的水平的。

最后获准参加ESPRIT I 计划的共有420个研究单位,2900余名研究人员。除了欧洲的许多大公司之外,还有许多大学、研究所和中小企业参加了ESPRIT I 的研究工作。下表给出到1988年底ESPRIT I 的实施情况:

ESPRIT I 项目总数	226
1988年之前完成	37
1988年内完成	54
1988年底仍在进行	135
参加的单位	526
小型公司(50人以下)	62
中型公司(50—500人)	84
其他公司	181
大学和研究所	199

到1988年底,总计226个项目中已有130个项目获得了168项具体成果。其中42项已应用于市场产品,50项已应用于即将投放市场的产品,46项已通过技术转让被公司采用,30项已成为国际标准或正被国际标准化组织考虑。

ESPRIT I 确已大大地推动了欧洲信息产业界的多层次合作。过去,欧洲信息产业公司之间很少合作,而是寻求与美国公司合作。通过ESPRIT I 在科研方面的合作,1983—1986年期间欧洲信息产业公司之间在产品开发、市场开发和合资等方面的商业协议增加了7倍,达到了与美国公司所订协议的数量。

ESPRIT 计划的第二阶段(1988—1992)

ESPRIT 计划的第二阶段投资总额为32亿

欧洲货币单位。它的重要特点是基础研究与工业应用并重,并把合作范围扩大到欧洲共同体之外的一些欧洲国家(奥地利、瑞士、瑞典、挪威、芬兰)。

第二阶段计划称为ESPRIT I,其目标为:

1. 建立欧洲新一代集成式信息处理系统;
2. 提高欧洲信息技术在广泛领域的系统环境方面的应用能力;
3. 在信息技术的适当领域中开展基础研究,作为工业研究与开发的补充。

ESPRIT I 的首次征集科研项目申请的工作已经结束,共有156个项目入选,投资总额为18亿欧洲货币单位。研究工作已于1989年上半年开始。ESPRIT 有一份出版物称作“Synopses”,详细列出各个项目的研究目标和技术途径。

ESPRIT I 的一个重要特点是给予基础研究以一定的重视。美国对今后五年的计算机科学基础研究共拨款4.5亿美元,而欧洲各国在这方面却呈下降趋势。针对这一情况,从1987年以来ESPRIT 计划经过长期讨论和酝酿,决定在ESPRIT I 中开始对信息技术的有限领域内的基础研究给予支持。到1988年10月底,已决定对62个基础研究课题(涉及285个组织)提供资助,同时成立若干工作小组促进合作,总投资额达6300万欧洲货币单位。所选定的基础研究领域为微电子学、计算机科学、人工智能与认识科学中可望经过欧洲的全面合作而获得突破的领域,并且带有多学科合作的性质。

小 结

从上面简单综述不难看出ESPRIT 计划有如下一些特点值得我们借鉴:

1. 高技术科研具有一定超前性质,但必须于数年内促进高技术产业的发展和提高产品竞争力方面显示出具体效果。
2. 基础研究具有更加超前的性质,但必须慎重选择有可能在近期获得突破的领域加以重点支持,不能把有限的资金平均分配。
3. 工业界和学术界以高技术科研合作促进

培养CAD人才 推广CAD技术

南京师范大学计算中心 陈冠清

计算机科学技术的迅猛发展,推动着各行各业计算机技术的应用,近年来,CAD技术更为引人瞩目,它是计算机科学与工程相结合的一种崭新的技术,CAD技术是对传统设计方式的革命,它能提高设计质量和速度,CAD技术能适应“多品种少批量”的生产要求,产品更新快,竞争能力强,是增强企业自身竞争力的手段,为工厂企业和工程界所关注。

CAD技术在大中型工厂企业中已经引起了重视,但在中小型企业尚属起步。我们在南京市对28个中小企业进行了抽样调查,今就中小企业对CAD需求的主要表现,采用CAD技术的做法,以及发展CAD技术中遇到的问题归纳如下:

1. 中小企业对CAD需求的主要表现

目前中小企业对CAD的需求,在很大程度上取决于企业领导。

a, 企业经济效益较好,领导有长远眼光,愿在CAD方面进行投资做些探索的占10%左右。

b, 尽管企业经济效益不太好,领导愿意进行技术储备,为新产品开发准备手段的占10%左右。

c, 企业规模小,产品单一,技术要求不高,不考虑CAD技术的占30%—50%。

d, 因CAD效益不好,应用尚有距离,领导处于观望的占20%—30%。

2. 中小企业采用CAD技术的做法

总的说来,中小企业在上CAD项目时,均持谨慎态度,从经济效益,技术人才和目前需求

出发,大多数单位采取“交学费,试试看”的办法,即利用工厂原有微机(AT, 286, 386等)加上一些配置,使之变成一个低档的CAD硬件系统。“花钱不多,可以起步”。它们大多数从“辅助绘图”开始,加上部分电算程序,查表程序,构成了企业中实用的软件包。

3. 中小企业在发展CAD技术中遇到的问题

中小企业在发展CAD技术中主要遇到以下四个问题。

a, 资金问题。中小企业经济条件差,普遍要求政府在资金上给予支持,在贷款还款方面给予优惠。

b, 领导认识。企业领导有短期行为,他们只对自己承包的任期内负责,对技术储备不感兴趣。有少数领导自己是科盲,不懂CAD的用途,当然谈不上采用CAD技术的兴趣了。

c, 专业技术人员素质差。中小企业的专业技术人员本来就不多,工作负担较重,没有机会,没有时间,没有精力去学习CAD知识,这给应用CAD技术带来困难。

d, 中小企业中现有的CAD系统运行情况较差,影响了其它企业采用CAD的积极性。

4. 培养CAD人才、推广CAD技术

计算机可以用于各行各业,但真要使计算机应用搞得好,必须依靠广大的专业技术人员。CAD技术属于计算机应用,而且专业性很强。要想使CAD技术在中小型企业推广使用,必须大力培养既懂专业又懂计算机的两用人才。中小企业在发展CAD技术中遇到的问题,除了资金

了高技术产业的经济合作,而且这种合作是多国性的。ESPRIT计划的成功提示我们国内,全国性的集学术界与工业界科技精英于一体的高技术科技和高技术产业合作是大有可为的事业。

参考文献

G. Metakides, ESPRIT, BASIC RESEARCH ACTIONS, Bulletin of EATCS, No37, 1989.2, pp59—86

问题外,其余都与CAD人才有关。

这样的两用人才,既可从本专业中培养,也可从计算机专业中培养,但从本专业中培养比较容易实现。这种两用人才,要求他们精通本专业,懂得一定的计算机知识就可以了。本专业人员要补充的计算机知识较少,如懂得数据结构,数据库原理,软件工程,1—2种高级语言大致就

可以了,不要求他们去了解计算机的内部结构。而计算机专业人员转向某个应用专业,需要补充的知识就多了,他们须要从该专业的基础知识补起,甚至还要懂得该专业的制造工艺。

迅速培养CAD人才,是推广CAD技术的关键所在。

NITLISP 语言通过鉴定

日前机电部召开的软件鉴定会上通过了东南大学研制的NITLISP语言。这种语言是以GCLISP语言为基础,再进一步加以扩充和改进而成,实现了汉化。该语言除保持了GCLISP

语言的全部功能外,还具有中文信息处理及显示、模块管理和窗口管理等功能。

(李相彬)

智能化双全臂假肢在宁研制成功

一种国内首创的智能化双全臂假肢由东南大学和江苏康复中心研制成功,为无臂伤残者带来了福音。这种假肢装在伤残者上体和残肢上,伤残者可根据意愿操纵触压开关,将信号输入装在左上臂壳体内的微机,在微机的控制下来做举臂、曲肘、手腕翻转,抓握等动作。它可帮助伤残者

完成喝水、开门、取物、写字、打电话等日常生活中的多种动作,达到生活自理的目的。该假肢重约3公斤,使用方便、造价低。据计算它的造价仅是国外同类产品的1/3。

(李相彬)

工科高等数学试题库系统研制成功

西安交通大学计算机系软件教研室 张伟

本试题库系统是在国家教委领导下,由西安交通大学、北京理工大学、华南理工大学和上海高校工科数学协作组四单位研制成功的,关于1989年2月16日由国家教委组织在北京正式通过鉴定。

该成果体现了自然科学与教育科学相结合的特点,库存4200道试题均建立了题卡,确定了反映试题各项性能的指标,能根据用户各种要求由

计算机迅速自动地生成试卷,同时输出题解、评分标准和各种反馈信息表。生成的五种水平的标准试卷,符合工科院校高等数学的教学要求,适用于工科《高等数学》课程的各类院校的常规考试。

该题库系统规模大,功能全,与国内各试题库相比,处于领先地位,有较大的社会效益,对其它课程试题库的建设也有重要的参考价值。

我国色谱数据处理工作正向现代化水平迈进

——上海计算所致力开发色谱数据处理机系列产品

色谱仪是物质成份分析和检测的重要工具。而色谱数据处理机就是采用微机对色谱仪实现色谱数据处理,从而使数据处理速度比手工处理提

高十倍,特别是可以解决人工无法完成的复杂成份、超微量组份的精确定量。该机国外在70年代末问世。我国则全部倚仗舶来机,且用户需求量

计算机藏品管理系统软件简介

南京博物院 杨 战

最近笔者已为南京博物院设计出实现了编目、查询、检索、统计、打印等项目自动化管理的计算机藏品管理系统软件,代替了传统的人工劳动,使现代科学进入了古老的博物院,实现了“今为古用”。

在现代化藏品管理中采用计算机技术有下列优越性:

1. 我国有着几千年悠久的历史,有光辉灿烂的文化,我院大多数藏品均系历史上遗留下来的珍贵文物,其中有很多是当今世界的孤本。由于日常工作中需常提用这些文物,易对文物造成一定破损,用计算机管理可改善此状况。

2. 在文物考古研究中,常从大量卡片中去查询所需资料,再对照实物研究,或遇到专题文物研究时,从各个不同库房中去查阅各种卡片,工作量大,效率低,由于文物资料分散,使专题研究缺少总体概念,今用计算机处理可迅速检索、查考,提高工作效率和研究质量。

3. 用计算机处理为全院藏品目录与藏品的总数建立完整和确切的资料,形成一个完整的资料体系。

4. 以前文物是按社会属性等特征入藏,因科研需要,今后各文物按质地入藏,而不便于科研工作开展,传统人工保管方式对文物保管有利。

用计算机管理后,可克服上述矛盾,将卡片内容输入电脑后,可以在电脑中建立各种所需的专题内容库和自动组合为各种需要的专题资料,以提高效率,提高准确性。

5. 应用计算机处理技术,完全代替了人工的查询、登记卡片等各项工作。从而节省大量人力。

本套系统具有用于全院藏品的建档、查询,便于各种文物专题研究等功能,可将全院藏品的各种信息贮存在电脑的存贮器中,以永久性记住这些信息,代替卡片的存档。如需要对某种质地相同的文物进行研究,可将质地相同的文物的分类号和文物名称等信息归集在一起,形成一个专用卡片库供使用,这就有点象从大量的卡片中抄下分类号和文物名称等,将其放入一个卡片柜中一样,其工作质量优于人工质量;而且可根据需要立即打印出各藏品目录、藏品登记卡等资料。实现按用户要求建立专题文物库,统计文物件数,按质地分卡片,按社会属性分卡片等要求。使用了计算机之后,从而改变了我们博物院工作中,在文物建档管理等方面的许多落后状况,使我院的文物管理走向办公自动化,使我们博物院工作进入现代化的行列。需此软件者,可同作者联系。

又很大,国家去年因此要花费大量外汇用于进口。1980年底,上海计算所通过社会调研获取这一重要信息后,迅即组织力量着手色谱数据处理机的研制工作,这在国内尚属第一家。经过八年耕耘,已先后开发出市场紧俏的七种CDMC型号的色谱数据处理机,经专家和广大用户确认均达到了80年代国际上同类产品的水平,完全可与进口的同类型数据处理机相媲美,现已被广泛应用于石油、化工、医药、能源、生化、食品、环

保、国防、生物工程等领域。今年,上海计算所将陆续推出色谱数据处理机IBX型和H型两个新型号机应市。

专家和众多的用户认为,上海计算所研制生产的色谱数据处理机系列产品,为我国气相色谱、液相色谱、薄层、凝胶色谱及其现代分析仪器的微机化及联用开拓了良好前景;同时每年可为国家节约大量外汇。

(吕宜男)

简介一种用于打印机的绘图程序库

吉林省地矿局物探队计算站 孟 洪

计算机自动绘图在事务管理、科学计算以及辅助设计等各个领域都起着重要作用。在不具备专用绘图设备的条件下,点阵式打印机也是一种较好的绘图工具。这方面的例子已有很多,但大多采用BASIC语言,且多是专用的绘图软件包。此处简介一种以FORTRAN语言编制而成的、面向工程技术人员和管理人员的绘图程序库,它提供了用户与打印机之间的交互界面,使用者不必了解打印机的具体指令,仅仅是灵活地调用和组合库的各种功能子程序,就能开发和绘出各种各样的图形和图表。

该程序库基本特征如下

(一)绘图子程序采用FORTRAN—77语言编写,发挥了这种高级语言的模块化特点,弥补了BASIC语言程序运行速度慢的缺陷。用户根据自己的制图需要,相应地利用程序库中的绘图子程序编写主程序段,并将其单独编译后与该绘图程序库(.OBJ文件)连接到一起形成目标程序。

(二)程序库研制机型为DUAL—68000,并已在IBM—PC机上实现。原则上可移植到其它任何具有FORTRAN—77语言的其它机器,占内存约30K左右。

(三)在LQ—1500串、并型打印机上成图,即参照该种打印机的指令集编写的具体绘图指令。可分成8针和24针两种绘图系统,8针型绘图速度快,24针型速度较慢,但图形美观,线条流畅。该打印机与FX系列打印机兼容,而其点阵绘图原理适用于其它的9针或24针点阵打印机。

(四)绘图面积宽。以宽行打印纸横向长为极限,约为34.5cm,纵向长度可无限制。

绘图程序库共由16个子程序组成,其中9个可由使用者在主程序中直接调用。包括定原点、清除帧缓冲区、画点、画线、绘圆、绘矩形、圆形充填和矩形充填等功能。只要稍加熟悉每个子程序的作用和调用参数,就可用它来编制绘图主程序段。需此绘图程序者,可同作者联系。

好消息

应广大计算机应用研究科技工作者和本刊众多作者要求,为了振兴我国的信息产业,本社将于本年11月6日至11日在成都地区举办1989年度《全国计算机应用研究学术交流会》,会议的主题是:探讨计算机软件维护及硬件维修在计算机推广应用中的战略地位;交流软件维护、硬件维修中的实践经验;展望软件维护、硬件维修的发展前景。围绕这一主题,也相应涉猎计算机应用中具有推广价值的其它典型课题。会议的宗旨是:探讨学术,交流经验,沟通信息,增进友谊,开拓视野,取长补短,提高效益。报名及征稿截止日期为1989年10月20日。

欲知有关详情,请同本社邓嘉澍、唐大利同志联系。

热诚欢迎计算机界专家及各行各业推广应用计算机的广大科技工作者踊跃应征,光临指导!热诚欢迎本社新朋故友前来叙旧情、增新谊,传经送宝!

《计算机应用研究》杂志社

1989年8月

CX (川星) 286微机简介

最近, 位于四川省绵阳市的国营华西电子计算机厂在四川省电子工业厅召开了“川星286微机技术评定暨展销会”, 隆重推出了该厂经过一年多努力, 在过去生产长城0520CH微机的基础上, 开发、创新、研制成功的川星286微机。该机的主要特点如下:

一 高速度

一般286档次的微机其时钟频率可达12.5MHZ, 而川星286微机可达16MHZ, 且可保证整个微机系统安全可靠的运行。经过电子科技大学(中国)微机研究所试用和测试, 证明川星286不但大大优于GW0520CH, 而且还大大优于GW286B。测试结果如下表:

测试内容	GW0520CH	GW286B	CX286
测试内存读写速度	28	3	2
测试内存与硬盘交换数据速度	234	76	42
测试内存与CPU字符交换数据速度	120	25	19
测试内存与CPU字节交换数据速度	112	22	18
测试内存与CPU字交换数据速度	112	23	18
测试内存与CPU长字交换数据速度	120	25	19
测试短整数加运算速度	166	31	20
测试整数加运算速度	212	39	24
测试长整数加运算速度	327	54	33
测试实数加运算速度	103	17	12
测试双精度加运算速度	111	19	14
测试双精度乘运算速度	41	7	6
测试双精度除运算速度	53	8	6
测试综合运算速度	69	16	12

(注: 时间单位“秒”)

二 高可靠性

由于采用了VLSI大规模系列门阵片, 川星286具有高集成度。整个主板除RAM芯片外, 不到二十块芯片(与同档次的286机主板相比, 尺寸亦大大减小)。该机的1.2MB和360KB软盘驱动器及40兆硬盘驱动器等部件亦选择了世界名牌产品。因此, 大大提高了可靠性。四川省供销社微机室等用户在恶劣环境下试用川星286时, 将电源电压升高至240伏及降低至180伏时, 系统仍正常可靠地工作。

三 优良的汉字处理功能

川星286充分考虑了目前国内外处理汉字的特点, 采用IBM 101 键键盘, 但又具有长城系列机键盘的优点(如键上标有偏旁部首等), 为用户输入汉字带来了极大的方便。

四 高度的兼容性

川星286采用DOS3.2操作系统, CEGA显示方式(带ROM汉字库), 与IBM PC/XT, IBM PC/AT, 长城, 浪潮等机种完全兼容, 各种软件可不经修改直接在该机使用, 如四川省计经委经济信息中心, 四川省供销社系统等单位的软件及未来Ⅲ汉字输入码, 《大众》码, 汉化的GW1—2—3集成软件等, 均可在CX286机上正常运行。

五 良好的可扩展性

RAM: 1兆, 可扩展至4兆, 最大可达16兆。

ROM: 128KB。

可扩展硬盘一个。

可扩展80287

具有I/O扩展槽8个。

DMA通道7个。

六 优良的性能价格比

综上所述, 川星286微机性能在同档次的微机中处于领先地位, 但售价仅为25000元/套, 3台以上优惠价为24000元/套, 低于同档次的286微机。因此, 川星286微机的性能价格比处于同档次微机的领先地位。

国营华西电子计算机厂 张顺全 四川省电子计算机应用研究中心 于作达