

# 計算機應用研究

1990

2

APPLICATION RESEARCH OF COMPUTERS 《計算機應用研究》雜誌社







北京四通新技术产业股份有限公司

计算机事业部

## 隆重推出

### ● 万能编程测试卡

无所不能  
开发必备  
十位一体

### ● MICE-II 微机在线仿真器

硬件侦错  
系统调试  
逻辑分析



**MICE-II**

- |                 |                |
|-----------------|----------------|
| 1. EPROM        | 2. EEPROM      |
| 3. 8751 系列      | 4. 8748 系列     |
| 5. Z86 系列       | 6. BPROM(PROM) |
| 7. 逻辑电路测试       | 8. RAM 测试      |
| 9. PALS GAL 编程  |                |
| 10. PEEL/PLD 编程 |                |

仿真CPU: 8085 8086  
8088 Z80等  
是开发调试微电脑的必要  
工具。

## 北京四通集团公司万能编程测试卡简介

北京四通公司计算机部新推出万能编程测试卡,该卡功能卓越,包罗目前市场上各种编程卡的功能。集 EPROM 编程卡、EEPROM 编程卡、8751 单片机编程卡、8748 单片机编程卡、Z86 单片机编程卡、PROM 编程卡、PAL 编程卡、GAL 编程卡、逻辑电路测试仪、RAM 测试仪于一体,是开发者的有力伙伴,是此类产品中功能最全、设计最新的编程卡。

## 稿 约

1. 本刊宗旨系“应用促进研究,研究指导应用”,故来稿要求“新、实、短、精”,贵在新颖,重在实用,字数在 3500 字以下。
2. 恕不受理复印、复写稿及圆珠笔誊抄稿,请用钢笔正楷书写于标明字数的方格稿纸上;若用计算机打印,亦需按标准稿纸格式打印,每页均要求标明字数。
3. 文稿中的插图和程序清单均要求一式两份,编注号码,并注明在文稿中的位置。图纸要求正规描绘,程序清单要求黑白反差大,图纸和程序的宽度为 7cm 或 14cm 或 21cm,便于编辑、印刷。
4. 来稿要求附中英文标题及其摘要。
5. 请作者自留底稿。人手所限,恕不退稿。
6. 要求作者一稿一投。稿件刊出,赠阅样本,酌致稿酬。稿件寄出半年后若未见录或未收到录用通知,作者可函询或自行处理。

本刊编辑部

## 技术资料邮购消息

本刊有以下技术资料可供邮购(含邮费):

1. 《1988 年全国计算机应用研究学术交流会论文集》,每册 16 元;
2. 《1989 年全国计算机应用研究学术会论文集》,每册 39 元;
3. 《APPL II 微型机实用维修技术》,每册 2.5 元;
4. 《静电复印机维修指南》,每册 11 元;
5. 《IBM-PC 磁盘操作系统》,每册 3.75 元

需以上资料者,请同本刊唐大利同志联系;数量有限,欲购从速。

开户银行:成都跳伞塔分理处

帐号:89501299

此外,本刊将于最近出版《计算机病毒大观》论文集,热忱欢迎投稿和订阅!

## 好 消 息

四川省电子计算机应用研究中心引入国外技术开发的下列产品颇受青睐:

- 一. ICT-55A 智能通用数字集成电路测试仪—可检测 54/74 和 CMOS4000 系列及其兼容的同类芯片以及部分静态 RAM,同时可自动分析上述芯片的型号、功能。批发价:3200 元,零售价:4300 元。
- 二. 超级通用学习机苹果机读写卡—可读写 2716~27512 芯片, Vpp 程控自动选择。批发价:400 元(10 块以上);单价:550 元。
- 三. 超级通用 PC 机读写卡—可读写 2716~27512 芯片 Vpp 程控自动选择。批发:650 元(10 块以上);单价:850 元。

联系人:本刊读者服务部唐大利

本刊编辑部



## 《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）

一九九〇年

第七卷 第二期（总第34期）

主 编：张执谦

副 主 编：李泽民

本期责任编辑：齐墨之

编 辑 出 版：《计算机应用研究》杂志社

内 文 印 刷：西南冶金地质印刷厂

封 面 印 刷：四川省印刷制版中心

出 版 日 期：1990年3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

## 代数多项式的Horner—差商混合算法

辽宁大学 徐继锋

代数多项式是函数逼近应用最广泛的数学工具。研究多项式的计算机快速算法很有实用价值。

目前最常用且计算步骤最简的算法是 Horner 算法。即对  $n$  次多项式

$$P_n(x) = \sum_{k=0}^n a_k x^k$$

有递推公式:

$$\begin{cases} b_n = a_n \\ b_{i-1} = b_i x + a_{i-1} \end{cases} \quad i = n, n-1, \dots, 1$$

其中  $b_0 = P_n(x)$ 。

显然, 每计算一个函数值需要  $n$  次加法  $n$  次乘法。

本文把 Horner 算法与差商相结合, 推出了 Horner—差商混合算法, 它对计算多项式在批量均匀加密点上的值有极大的优越性, 一般比 Horner 算法节省机时一半以上。下面以应用最广泛的三次多项式为例, 给出 Horner—差商混合算法的原理及应用程序。设

$$P_3(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0$$

把区间  $[a, b]$  分成  $n$  等份, 分点  $x_i = a + ih$ ,  $i = 0, 1, \dots, n$ ,  $h = (b-a)/n$ , 现计算  $P_3(x_i)$ 。做差商表:

$P_3(x_0, x_1, x_2, x_3)$	$P_3(x_1, x_2, x_3, x_4)$	...		
$P_3(x_0, x_1, x_2)$	$P_3(x_1, x_2, x_3)$	$P_3(x_2, x_3, x_4)$	...	
$P_3(x_0, x_1)$	$P_3(x_1, x_2)$	$P_3(x_2, x_3)$	$P_3(x_3, x_4)$	...
$P_3(x_0)$	$P_3(x_1)$	$P_3(x_2)$	$P_3(x_3)$	$P_3(x_4) \dots$

过  $x_0, x_1, x_2, x_3$  做  $P_3(x)$  的插值多项式, 由差商定义,  $P[x_0, x_1, x_2, x_3]$  是这个插值多项式的首项系数, 又根据插值性质知该插值多项式就是  $P_3(x)$  本身, 所以  $P_3[x_0, x_1, x_2, x_3] = a_3$ , 同理  $P[x_i, x_{i+1}, x_{i+2}, x_{i+3}] = a_3, i = 1, 2, \dots, n-3$ 。

Horner—差商混合算法计算步骤为:

1. 用 Horner 算法先计算出  $P_3(x_0), P_3(x_1), P_3(x_2)$
2. 计算  $P_3(x_0, x_1), P_3(x_1, x_2), P_3(x_0, x_1, x_2)$
3. 计算  $Q_1 = 3h \cdot f[x_0, x_1, x_2, x_3] = 3ha_3$ ,  
 $Q_2 = 2h, Q_3 = h$
4. 对  $i = 0, 1, 2, \dots, n-3$ , 计算

$$P_3(x_{i+1}, x_{i+2}, x_{i+3}) = P_3(x_i, x_{i+1}, x_{i+2}) + Q_1$$

$$P_3(x_{i+2}, x_{i+3}) = P_3(x_{i+1}, x_{i+2}) + Q_2 \cdot P_3(x_{i+1}, x_{i+2}, x_{i+3})$$

$$P_3(x_{i+3}) = P_3(x_{i+2}) + Q_3 \cdot P_3(x_{i+2}, x_{i+3})$$

显然, 计算  $P_3(x_0), P_3(x_1), P_3(x_2)$  的步骤就是 Horner 算法, 从  $P_3(x_3)$  开始, 到  $P_3(x_n)$ , 每计算一点只用 2 次乘法, 3 次加法, 比 Horner 算法少一次乘法, 虽然第 2, 3 两步为第 4 步做准备, 需要 7 步乘除法和 3 步加减法, 但当  $n$  较大时, 第 4 步所节省的乘法数很快就可以给它找回来。

例: 计算  $2x^3 + 3x^2 + 4x + 5, x \in [1, 10], n$  分别取 50, 100, 200, 300, 500。在 IBM PC—XT 机上分别用 Horner 算法及 Horner—差商混合算法所用机时对比如下表。

$n$	Horner 算法	Horner—差商混合算法
50	2.5 秒	1 秒
100	5 秒	2 秒
200	10 秒	3.5 秒
300	15 秒	5 秒
500	25 秒	8 秒

# 数据库设计最小复盖方法

新疆电子计算中心 瓮正科

## 摘要

本文系统地介绍了具有牢固理论基础和成套算法的数据库设计方法—最小复盖方法, 该方法 是作者经过所谓的“通用数据库设计方法”的高度概括、提炼和简化得出的。利用该方法设计的 逻辑数据库, 不仅冗余小, 而且数据库完整性好, 同时所得到的逻辑数据库满足最佳4NF。

最小复盖方法(原名通用数据库设计方法)由 M. Vetter 和 R.N.Maddison 提出。该 方法牢固地建立在数据相关理论上, 具有系统的算法, 是数据库设计中能应用计算机进行自 动设计的新途径。该方法追求的目标是, 逻辑数据库尽可能对应于最重要、最常用的外部模型, 以便使为进行从逻辑数据结构到外部数据结构的变换所需要的操作达到最小; 同时也使得外部数 据到逻辑数据库的更新操作达到最小。该方法基本思想是从实体关系和关联关系出发, 依据数据 相关理论, 建立不可约关系模型, 然后利用传递闭包推导出最小复盖, 选择适当的最小复盖集 合, 转化为最佳4NF关系模型, 从而得到逻辑数据库。该方法的基本步骤为, 真实世界的概念 化, 确定不可约关系模型, 计算传递闭包, 确定最小复盖关系, 推导最佳4NF的逻辑数据库。

## 一、真实世界的概念化

在概念化阶段主要要做两件事。第一是确定 实体, 确定实体的名称, 属性和实体关系的关键 字。第二是确定关联集合, 得到关联关系, 并确 定其关联属性。分述如下:

确定实体:           实    体

雇员 (ER9) (工号 (E#), 雇员名(EN)),  
(PK: E#);  
部门 (ER10) (部门号 (D#), 部门名  
(DN)), (PK: D#);  
经理 (ER11) (经理号 (M#), 经理名  
(MN)), (PK: M#);  
技能 (ER7) (技能号 (S#), 技能名  
(SN)), (PK: S#)。

确定关系:           关    联

雇员工作关系 (E#, D#)  
经理工作关系 (M#, E#)  
被领导关联 (M#, E#)  
技能关系 (S#, E#)

另外, 分析人员分析出某个雇员掌握某种技 能水平的关系:

掌握关系 (ER8) (E#, S#, 水平(SL),  
(PK: E#, S#)。

关联关系是进行后面推导计算的关键, 因 此分析人员应该将所有尽可能了解得到的关联关 系分析出来, 否则, 就不可能全面地反映真实世 界的全貌。

附Horner—差商混合算法程序

```
10 INPUT "A,B,N="; A,B,N
20 DIM P3(N),A(3),B(3)
30 FOR I=0 TO 3
40 PRINT "A(",I,")=";
50 INPUT A(I)
60 NEXT I
70 H=(B-A)/N; B(3)=A(3)
90 FOR M=0 TO 2
100 X=A+M*H
110 FOR K=3 TO 1 STEP -1
120 B(K-1)=B(K)*X+A(K-1)
```

```
130 NEXT K
140 P3(M)=B(0)
150 NEXT M
160 P1=3*H*A(3); P2=2*H; P3=H
190 F01=(P3(1)-P3(0))/H; F12=(P3(2)-P3(1))/H
210 F02=(F12-F01)/P2; B=P3(2)
230 FOR I=3 TO N
240 F02=F02+P1; F12=F12+F02*P2
260 B=B+P3*F12; P3(1)=B
280 NEXT I
290 END
```

## 二、确定不可约关系模型

所谓不可约关系是, (A) 包括一个基本键, 可能是复合的; (B) 最多还有另一个属性, 也可能是复合的. 当且仅当“另一个属性”包含对某个实体的引用, 而且该实体是组合的基本键所标识时, 这个属性才是复合的. 总之不可约关系应表示为彼此独立的“原子事实”关系. 根据概念化阶段得到的诸关系建立起一个不可约关系模型. 为了后面计算方便, 作两点简化:

1. 实体关系不参加计算, 因为实体关系的属性若和其他实体发生联系, 必然在关联集合中反应出来, 这里不妨先将它“隐藏”起来, 最后在逻辑数据库中为它设置一个关系即可.

2. 关联关系中具有自身属性的关系, 该属性不被其他关联关系所引用的关系也作为实体关系来处理.

确定不可约关系模型:

雇员工作关系:  $(E\#, D\#)$ , 一个雇员只能在一个部门工作, 而一个部门可以有許多雇员,  $\therefore E\# \rightarrow D\#, (PK: E\#)$ .

经理工作关系:  $(M\#, D\#)$ , 一个部门只有一个经理, 而一个经理只能在一个部门任职. 该关系中  $D\#$  或  $M\#$  都是候选键,  $\therefore M\# \rightarrow D\#,$  还有  $D\# \rightarrow M\#$ .

被领导关系:  $(M\#, E\#)$ , 一个经理可以领导多个雇员, 但一个雇员只能接受一个经理的领导.  $\therefore (M\# \rightarrow E\#)$ .

技能关系:  $(S\#, E\#)$ , 一个技能可以被多个雇员掌握, 而一个雇员可以掌握多种技能.  $\therefore (S\#, E\#) \rightarrow S\#, (S\#, E\#) \rightarrow E\#$ .

至此得到不可约关系模型.

## 三、确定传递闭包

依据不可约关系模型, 就可以画出一个函数依赖有向图, 如图2所示. 将有向图用矩阵表示如图1所示.

利用A矩阵就可以推导出传递闭包, 其算法如下:

求解传递闭包算法:

```
FOR I=1, N      * N...矩阵的维数
FOR J=1, N
IF I<>J.AND.A(I,J)=0
      * 矩阵A中非1的元素
      * 皆为0.
FOR K=1, N
IF A(I, K)=1.AND.A(K, J)=1
A(I, J)=1      * 得到一个新关系
K=N+1
ENDIF
NEXT K
ENDIF
NEXT J
NEXT I
结束.
```

A	S#	S#, E#	E#	D#	S#
S#					
S#, E#	1		1		
E#				1	1
D#					1
M#				1	

图 1: 有向图的连接矩阵A

B	S#	S#, E#	E#	D#	S#
S#					
S#, E#	1		1	①	①
E#				1	1
D#					1
M#				1	

图 2: 不可约关系模型有向图

通过计算得到新的矩阵和有向图如图3和图4所示.

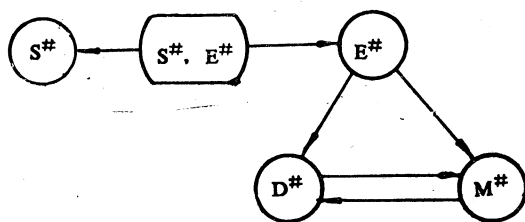


图 3: 传递闭包有向图的连接矩阵 B

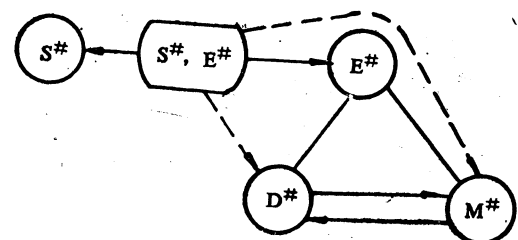


图 4: 传递闭包有向图

传递闭包的结果产生出一些新的关系, 这些关系若没有实际意义就把它删除掉, 仅将有意义的留下。本例中新产生的 $(S\#, E\#, M\#)$ 和 $(S\#, E\#, D\#)$ 关系语义上不成立, 所以删除掉, 结果仍同图1和图2所示。值得注意的传递闭包所产生的新关系并不是都不成立, 本例是一特例。

#### 四、确定最小复盖

根据传递闭包的结果图1和图2来确定最小复盖。所谓最小复盖就是一套没有冗余的基本关系。值得提醒的是最小复盖不是唯一的。通过推导可以得到一套最小复盖集合, 可以根据用户的需要选择合适的一套基本关系。确定最小复盖的算法如下:

求解最小复盖算法:

设:  $S^z = \{ER_1, ER_2, \dots, ER_z\}$

代表原有的一套基本关系, 并让:

$MC\{S_n^m\}$

代表最小复盖集合, 其中  $z, m$  分别是  $S$  和  $MC$  的基数,  $n$  代表序号, 开始  $MC$  是一个空集合。

步骤1: 找出  $S^z$  中每一个基本关系的最长路径:

步骤2: 如果  $ER_r$  是  $ER_i, ER_j$  最大路径的复合, 即:

(a)  $ER_i, ER_j, ER_r$  都属于  $S^z$ ;

(b)  $ER_r = \{ER_i, ER_j\}$ ;

(c)  $ER_r$  具有最大路径。

那么就把  $ER_r$  从  $S^z$  中删掉, 得到剩余集合  $S^{z-1}$

步骤3: 如果剩余集  $S^{z-1}$  没有可删除的元素, 就把其  $S^{z-1}$  放到  $MC$  中去, 转步骤4, 否则, 修改剩余基本关系的最长路径, 重复步骤2。

步骤4: 结束。

上述算法中, 若存在  $ER_r'$  满足  $ER_r$  的条件, 那么选择  $ER_r'$  又得到另一个最小复盖集合, 这就是所谓的最小复盖不唯一性。

接上例, 设:

$S^0 = \{ER_1, ER_2, ER_3, ER_4, ER_5, ER_6\}$

步骤1: 计算每个关系的最长路径。

$ER_1$	$D_1=1$
$ER_2$	$D_2=1$
$ER_3=\{ER_4, ER_5\}$	$D_3=2$
$ER_4=\{ER_3, ER_6\}$	$D_4=2$
$ER_5$	$D_5=1$
$ER_6$	$D_6=1$

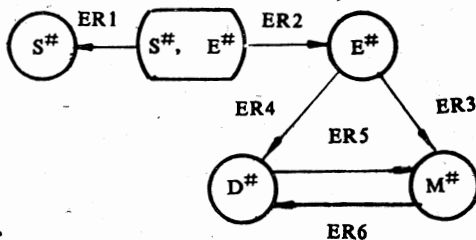


图5: 基本关系有向图

步骤2: 删除具有最长路径的复合关系。

可选择的关系是  $ER_3$  和  $ER_4$ 。

1. 若删除  $ER_3$  关系, 剩余集合  $S^1 = \{ER_1, ER_2, ER_4, ER_5, ER_6\}$  由于  $ER_3$  的删除,  $ER_4$  的距离也变短了, 所以  $S$  中再也没有可删除的元素, 则:  $MC = S$ 。结束。

2. 若删除  $ER_4$  关系, 则得到的  $MC' = S^1 = \{ER_1, ER_2, ER_3, ER_5, ER_6\}$ 。

$MC$  和  $MC'$  都是最小复盖, 根据用户需要选择其中一个集合即可。

#### 五、确定最佳4NF关系逻辑数据库

基于不可约关系模型, 通过传递闭包和最小复盖的计算, 得到的基本关系就是 4NF, 另外, 一开始“隐藏”起来的关系一般也是 4NF 的关系。4NF 关系的数据模型, 由于关系数目太多, 并不是一个好的关系模型, 所以需要求最佳 4NF 的关系模型, 算法如下。

求解最佳 4NF 关系模型算法:

1. 将具有相同键的关系放在一起, 得到一个子集  $S_i$ ;

2. 将每个  $S_i$  进行合并, 得到的关系, 其所有非键属性  $A_1, A_2, \dots, A_n$  都函数依赖于同一个键  $K$ , 得到:

$ER_i = \{K, A_1, A_2, \dots, A_n\}$



该关系既没有非完全函数依赖,也没有多值函数依赖)。

3.合并结束后,若没有实际意义(并非数学意义上)的传递依赖,就得到最佳4NF关系模

型,转第5步;否则,进行第4步。

4.用投影法消去实际意义的传递依赖。

5.结束。

根据该算法进行上例计算,关系如下:

从最小复盖得到的基本关系

ER1 (S#, E#)	(PK: S#, E#)
ER2 (S#, E#)	(PK: S#, E#)
ER4 (E#, D#)	(PK: E#)
ER5 (D#, M#)	(PK: D#)
ER6 (M#, D#)	(PK: M#)

开始“隐藏”的关系

ER7 (S#, SN)	(PK: S#)
ER8 (S#, E#, SL)	(PK: S#, E#)
ER9 (E#, EN)	(PK: E#)
ER10 (D#, DN)	(PK: D#)
ER11 (N#, MN)	(PK: M#)

合并同键关系,得到:

ER12 (S#, SN)	(PK: S#)	技能关系
ER13 (S#, E#, SL)	(PK: S#, E#)	掌握关系
ER14 (E#, EN, D#)	(PK: E#)	雇员关系
ER15 (D#, DN, M#)	(PK: D#)	部门关系
ER16 (M#, MN, D#)	(PK: M#)	经理关系

这组关系中不存在传递依赖,所以得到的关系集即为最佳4NF关系逻辑数据库。

## 六、结 束 语

最小复盖方法是通过把1NF的关系分解为不可约的关系,经过传递闭包的计算,确定最小复盖,然后计算出最佳4NF关系模型,构成逻辑

数据库。这种数据库具有冗余最小;没有存储操作异常;真实世界的数据库到逻辑数据库的数据库操作最少;从逻辑数据库中查询操作也是最少;另外在保持和真实世界具有依赖相一致的情况下,保证了数据库的完整性。这些优点是其它方法难以达到的。因为该方法以一套完整的算法和牢固的理论为依据。

## 一个不易破解的文件名加密法

湖北省荆州师专计算机室 瞿 波

APPLE机用户常在文件名中夹进不可见的控制字符(同时按CTRL键和其它字母键)来对文件名进行加密,以保护磁盘文件。但在使用控制字符时,大都不用CTRL-H,原因大概是因为CTRL-H是个功能键,无法通过按键将其加进文件名中。其实,是有办法将CTRL-H加进文件名中的,而且CTRL-H加进文件名中后还很难破解呢!

在文件名中夹进一般的控制字符,如CTRL-A、CTRL-E等,其保密性已不是什么很强了,因为APPLE DOS手册中提供了一个解密程序,运行这个解密的程序后再列磁盘目录。则文件名中的控制字符就会以闪烁的方式显示出来,从而被破密,但是这个解密的程序对CTRL

-H无效,它不能将CTRL-H以可见的形式显示出来,也就是说它不能破解含有CTRL-H的文件名。

那么,如何将CTRL-H加进文件名中呢? CTRL-H相当于一个功能键,从键盘上按CTRL-H,将使光标左移一格,使前一次按键所产生的字符成为待定位符。显然是不能直接从键盘上按CTRL-H将其加进文件名中的,而只有采用间接的方法,通过CHR\$函数进行转换才能奏效。CTRL-H的ASCII代码是8,假设欲加密的文件名为A(CTRL-H)B,则可进行如下操作完成文件名的加密。

```
PRINT CHR$(4) "SAVE A" CHR$(8) "B"
```

# 窗口软件的编写

深圳水产供销公司 刘宇菁

## 1. 概述:

窗口是屏幕上具有某种特殊用途的部分。通常这块区域是方形。一个屏幕上可以同时出现好几个窗口,使用者可以在各个窗口中进行工作。

实际上,窗口的理论基础相当简单:在程序执行时,程序中的各个工作分别使用各自的窗口,当工作开始时,它所使用的窗口就被启动,而在工作完成之后,该窗口便消失;假如某项工作在执行期间被别的工作中断,则此工作将暂时停止执行和使用其窗口,但此窗口并不会消失,尔后新的工作便启动它的窗口,将它盖在其它窗口的上面。如果没有窗口,则每一项工作都会清除屏幕,不仅打断了使用者的注意力,而且容易引起混淆。

## 2. 设计思想

如果要正确、快捷地使用窗口,则必须通盘进行考虑。

由于窗口一般覆盖一块比较大的区域,所以,即使在很快的电脑上执行,ROM-BIOS也无法满足速度上的要求,因此,为了突出窗口的特点,就必须以极快的速度呈现和清除窗口。我们建议读者直接存取屏幕缓冲区。

无论如何,在窗口出现时,该区域内的信息必须先储存起来,然后才能显示窗口;当使用完窗口时,必须将屏幕该区域内的信息还原。

列目录时,该文件名仅显示一个B,那个解密的过程也解不了它。

当你自己要调用这个文件时,也应采用间接的方法,如:

```
PRINT CHR$(4) "RUN A "CHR$(8) "B" /
```

使用CTRL-H进行加密吧,它的保密性你已经知道了。

附:APPLE DOS手册中的解密程序如下:

使用窗口时,不能超出窗口的边界。由于C语言常用的输入/输出函数,不能处理避免超出窗口边界的工作,因而必须使用专门为窗口所写的输入/输出函数。

基于以上的一些考虑,一个窗口系统应当有以下几个方面的内容:

- ①建立和初始化窗口的视框
- ②启动窗口和消去窗口
- ③更改窗口大小和移动窗口
- ④专门的窗口输入输出函数
- ⑤保存和恢复屏幕原内容

下面,我们分别详细说明。

## 3. 窗口的视框:

如要正确地制作出窗口,则所有的窗口函数必须能在任何时刻引用窗口的一些属性,定义见实例程序。

在使用窗口之前,必须先初始化窗口的视框。本文的这个功能为make-window函数,其主要工作是:

- ①检查窗口的设计是否合理
- ②为保存屏幕信息申请缓冲区
- ③初始化窗口的各个属性

详细程序见实例。

## 4. 启动窗口和消去窗口

启动一个窗口,使用window函数,其主

ILIGT

```
10 DATA 201,141,240,21,201,136
20 DATA 240,17,201,129,144,13
30 DATA 201,160,176,9,72,132
40 DATA 53,56,233,64,76,249
50 DATA 253,76,240,253
60 FOR I = 768 TO 768 + 27
70 READ V: POKE I,V: NEXT I
80 POKE 54,0: POKE 55,3
90 CALL 1002
```

## 一个用户友好的文件装载窗口\*

四川大学计算机科学系 唐常杰

许多优秀的软件,例如PC—TOOL及Turbo系列的Pascal、Basic、C等等,都有一个文件装载窗口。当用户在File窗口中选点Load时,出现一个目录窗口,列出指定目录下的文件和子目录名,用户用上下左右等箭头依动光条,选中文件后按〈Enter〉,则选点文件被装入进行处理,按〈Esc〉则退出。文件装载窗口有三个优点:

1. 用户不需键入文件名。
2. 用户不知道目录中有何文件时,也可以先浏览,后选择。
3. 不会出现“File not found”之类的错误,因目录中列出的文件都是存在的。

本文提供一个可供Turbo Pascal 4.0或5.0调用的文件装载单元的源程序ULoadF.Pas。用户只要在自己的程序中加入uses ULoadF语句,就可在用户指定的窗口中实现上述浏览选点文件功能。

这一工具已用于Print64 (5.2版),及国家自然科学基金8688303项目中。经过实践考验,证明其技术细节周到,坚固性强,源程序可读性好,并优于Turbo系列文件的类似界面,例

如按上下箭头时,采用InsLine而不是换屏,减少闪烁,此外,屏幕显示比Turbo软件整齐。

为便于阅读,已用分隔线划分了模块并标注了过程的层次,注解中标有行号。

清单第1~379行为文件装载单元ULoadF, (U—unit, F—File)。3~12行为单元的接口部分InterFace,它提供了用户可以调用的过程及其参数的名称和类型。由接口可知,ULoadF提供了三个过程,其中SetHires Mode和SetHires Color是用户用于CCDOS时设高分辨模式和颜色用,当ReverseChar为False时是反相字。Browse—Load—File是本单元中心。参数X1, Y1, X2, Y2是用户指定的窗口左上角和右下角坐标。Path是用户给出的路径或掩码(例如\Basic或\dBase\\*.DBF)。如用于CCDOS,应令useCCDOS为True,则可显示汉字,否则为False。变参SelectedF是用户用箭头键选定的文件全名,交给调主进程处理。

第14~379行是单元实现部分,可视其为一个黑箱,用户不须了解其内部细节也能使用单元。第17~32行分别实现一级过程SetHires

要工作是:

- ①设置好合适的显示区始址
- ②保存原屏幕内容
- ③显示窗口的表头和画边框
- ④置窗口内游标到(0, 0)

消去一个窗口,使用deactivate函数。详细程序见实例。

### 5. 更改窗口大小和位置

使用者在使用一个窗口时,或许希望更改窗口的大小和位置。本节提供两个函数size和move。

### 6. 窗口输入/输出函数

为了方便使用窗口,必须编写多种专供窗口使用的输入输出函数。本节只提供5个函数:

window—xy, window—getche, window—gets, window—patchar, window—puts。

### 7. 保存和恢复

见实例中的save—window—video和restore—window—video

### 8. 实例程序(因篇幅所限,从略)



Mode和SetHiresColor过程。32~378行实现一级过程Browse—Load—File。主要的数据结构是第49行的文件清单类型。查目录的结果(文名和属性)存放在有500个元素的数组中,为了节省栈,第50行使用了指针类型(用Heap,堆,即动态空间),其余变量和类型名均可顾名思义。

二级过程FName—Len12(61~76)在文件名中添加空格,凑成长为12的整齐外形,改善显示效果。过程MakeFNameList(79~141行)搜索指定目录,并在内存的堆空间中建立文件名清单。其中包含了搜索,处理错误等。SetColor(143~161行)将选定的文件名改色显示,在CCDOS中用反相字,在PC—DOS中用紫底黄字。未选文件名用兰底白字。

Browse(162~373行)浏览选点文名,其中又有三个三级子过程。过程gotocolRow(172~184行)在用户指定的显示目录窗口中把光条移到第Col列Row行,如果文名是已选定的(Selected=True)则显示紫底黄字。过

程DisplayOnePage(186~214行)在目录窗口中,从给定的文件清单中文件序号起显示一个页面。过程WriteOneLine(216~227行),在上下移动箭头接触上下边界时,用此过程写一行目录。二级过程Browse主体从229~373行。首先显示一页(231),然后读键盘,分析是否〈ESC〉,〈Enter〉或箭头键之一(234~263行),然后分情况处理(268~365)。二级过程Browse—Load—File的主体在372~378行。其中258—263行为递归浏览子目录。

清单中382~422行给出一测试程序,说明如何使用这一单元。394~398行交互式地读入路径名,默认为当前目录。399行现在设useCCDOS为False,如果你用于CCDOS改其为True。401~407行是为CCDOS显示准备。409行调浏览装载过程。412~420显示选择结果,418行把被选的文件打印出来。(如果不是Text文件,则可能打印出一些怪字符)。

编译好的ULoadF.Tpu只有8688字节,这一工具可使软件产品界面提高一个档次。

```
Unit ULoadF;
(=====)
Interface                                     ( 3 )
  Uses Dos, Crt, graph;
  Procedure Browse_Load_File(
    X1,y1,x2,y2 : byte; { window }
    Path       : string;
    UseCCDOS   : boolean;
    Var SelectedF : string);
  Procedure SetHiresMode;
  Procedure SetHiresColor(ColorNum : byte;
    ReverseChar : boolean);

Implementation                               ( 14 )
(=====)
(----- SetHiresMode -----Level 1 -----)
Procedure SetHiresMode;                       ( 17 )
  Var graphDrive, graph_Mode : integer;
begin
  DirectVideo := false;                       ( 20 )
  GraphDrive:=cga; Graph_Mode:=cgaHi;
  initGraph(GraphDrive, Craph_Mode,'');
end;

(----- SetHiresColor -----Level 1 --)
Procedure SetHiresColor(colorNum : byte;
  reverseChar : boolean);
```

```
begin                                         ( 28 )
  setBkColor(colorNum);
  If reverseChar
    then textBackGround(colorNum);
end;                                         ( 32 )

(--- Browse_Load_File ---- Level 1 ----)
Procedure Browse_Load_File(                 ( 35 )
  X1,y1,x2,y2 : byte; { window }
  Path       : string;
  UseCCDOS   : boolean;
  Var SelectedF : string);
{ Broese dir in window ,and use arrow
key to select file}
Type                                           ( 42 )
  string12 = string[12];
  FName_Attr = record { F --- File }
    name:string12; {file Name }
    Attr:byte {Attr -- attribute}
  end; { record }                             ( 47 )

  FListType=array[1..500] of FName_Attr;
  FListPtrType = ^FListType;                 ( 50 )
  Const InitColor = False;
        SelectColor = True;                  ( 52 )
  Var MaxRow, MaxCol, whereRow, whereCol,
      Row, Col, TotalFNum, CurrFNum,
```

```

PageStartFNum, LastRow : integer;
FName : string12; {File name}
FListPtr : FListPtrType;
ListOK : boolean; { 58 }

```

```

{---- FName_Len12 -----Level 2 --}
Function FName_Len12(

```

```

    Name: string12) : string12;
{fill space, let name be of length 12}
begin

```

```

    if pos('.', Name) = 0 then { 65 }
        while Length(Name) < 12
            do Name := Name + ' '

```

```

    else { 68 }

```

```

        begin
            while Pos('.', Name) < 9 do
                insert(' ', Name, Pos('.', Name));
            while Length(Name) < 12 do
                insert(' ', Name, Pos('.', Name) + 1);
            end;

```

```

        FName_Len12 := Name
    end; { 76 }

```

```

{---- MakeFNameList --- level 2 -----}
Procedure MakeFNameList;

```

```

{Search Dir, Make FName list in memory}.
Var

```

```

    SearchP : string; (P--Path) { 82 }
    Error, i : byte;
    DirInfo : SearchRec; { in DOS Unit }

```

```

begin
    SearchP := Path; { 86 }
    ListOK := True;

```

```

    FindFirst(SearchP, Anyfile, DirInfo);
    Error := DosError;
    if (Error = 0) and { 90 }
        (DirInfo.Attr <> directory)

```

```

    then {Get current Path, cut FName}
        while (Path[Length(Path)] <> '\')
            and (Path[Length(Path)] <> ':')

```

```

        do delete(Path, Length(Path), 1);
        if (Error <> 0) or { 96 }
            (DirInfo.Attr = directory)

```

```

        then { 98 }
            begin { serch sub directory }

```

```

                if (Path[Length(Path)] <> '\')
                    then Path := Path + '\';

```

```

                if copy(SearchP,
                    length(Path) - 2, 3) <> '.*.*'
                    then SearchP := SearchP + '\.*.*';

```

```

                FindFirst(SearchP, Anyfile, DirInfo);
                Error := DosError; { 107 }
                if Error <> 0 then

```

```

                    begin
                        WriteLn('Directory ', Path,
                            ' not found!');
                        writeln('Press <Enter> for',
                            ' continue ...'); readln;
                        ListOK := false; Exit;
                    end { 115 }
                end;

```

```

                GetMem(FListPtr, sizeof(FListPtr));
                { allocate dynamic memory }
                TotalFNum := 0;

```

```

                while DosError = 0 do { 121 }
                    begin

```

```

                        FName := DirInfo.Name;
                        if (FName <> '.') and
                            (FName <> '..') then
                            begin { 126 }

```

```

                                with FListPtr^[TotalFNum] do
                                    begin
                                        name := FName;
                                        Attr := DirInfo.attr;
                                        end; Inc(TotalFNum)

```

```

                            end; { 132 }

```

```

                        FindNext(DirInfo); Error := DosError;
                    end; {while }

```

```

                    LastRow := TotalFNum div MaxCol + 1;
                    for i := TotalFNum to (LastRow) * MaxCol
                        do FListPtr^[i].name := ' ';
                    if LastRow <> 0 then { 139 }
                        TotalFNum := LastRow * MaxCol;
                    end; { MakeFNameList } { 141 }

```

```

{----- SetColor -----Level 2 -----}
Procedure SetColor (Selected: boolean);

```

```

{----- Do_Color -----Level 3-----}
Procedure Do_Color(Txt, Back: byte);

```

```

    begin textcolor(Txt);
        textbackground(Back);
    end; { 150 }

```

```

{-----Level 3-----}
begin { SetColor }

```

```

    if not UseCCDOS then { 153 }
        begin

```

```

            if Selected then { 155 }
                Do_Color(yellow, lightMagenta)

```

```

            else Do_Color(white, blue);
        end else { UseCCDOS }

```

```

            if Selected then textbackground(White)
            else textbackground(black)

```

```

        end; { SetColor } { 161 }

```

```

{---- Browse -----Level 2-----}
Procedure Browse ;
{Browse dir and use Arrow key to select}
Var ch:char; ArrKeyPressed: Boolean;
    CurrLine , TotalLine ,
    LineStarFNum : Integer ;

{--- GotoColRow -----Level 3-----}
Procedure GotoColRow(Col, Row : byte ;
                    Selected : boolean);
{ Set current selected position
  at ( Col, Row ) in name_table}
begin
    CurrFNum := PageStartFNum +
                (Row-1)*MaxCol+Col - 1;
    SelectedF := FListPtr^[CurrFNum].name;
    SetColor(Selected);
    gotoXY((Col-1)*15+1, Row); { 180 }
    write(FName_Len12(SelectedF):12 );
    whereCol:=Col; whereRow := Row ;
    { remember new position }
end; { 184 }

{----- displayOnePage ---Level 3-----}
Procedure displayOnePage
    ( PageStartFNum : Integer);
Var FNumInPage, OldCol, OldRow : integer;
begin
    SetColor(InitColor); ClrScr; { 191 }
    OldCol:=whereCol; OldRow:=WhereRow;
    FNumInPage:=1;
    whereRow:=1; whereCol:=1;
    CurrFNum := PageStartFNum; { 194 }

    while ( CurrFNum <= TotalFNum ) and
        (FNumInPage <= MaxCol * MaxRow) do
    begin
        FName := FListPtr^[CurrFNum].name;
        Write(FName_Len12(FName) : 12);
        if whereCol < MaxCol then
            begin write(' ');
                inc(whereCol); { 204 }
            end
        else if WhereRow < MaxRow then
            begin inc(whereRow);
                gotoxy(1, whereRow); whereCol:=1;
                end; { 219 }
            inc(CurrFNum); Inc(FNumInPage);
        end; { while }
        WhereCol:=OldCol; WhereRow:=OldRow;
    end; { 214 }

{----- writeOneLine ----- level 3 ---}
Procedure writeOneLine(

```

```

    LineStarFNum : integer);
Var i : Integer; { write one line FName}
begin
    SetColor(InitColor); { 220 }
    for i:=1 to maxCol do
    begin
        with FListPtr^[LineStarFNum-1+i] do
            write(FName_Len12 (Name):12);
            if i < maxCol then write(' ');
        end; { 226 }
    end;
{-----level 3-----}
begin { Browse } { 229 }
    PagestartFNum := 1;
    displayOnePage(PagestartFNum);
    gotoColRow(1,1 ,SetColor );
    repeat { 233 }
        ch := readkey ;
        if not ( ch in [#13, #0, #27] ) then
            begin
                write('^G'); ArrKeyPressed:=False;
            end
        else if ch=#27 then SelectedF:=#27
            { Abort selection } { 240 }
        else if ch=#0 then
            { Arrow key or F1-F10 been pressed }
            begin
                ch:=readkey; ArrKeyPressed:=true;
            end
        else if ch=#13 then { 246 }
            { a file name selected }
            begin { * }
                ArrKeyPressed:=False;
                { Full Name : } { 250 }
                SelectedF:= Path+SelectedF;
                {Is it sub Directory ? }
                CurrFNum := PageStartFNum +
                    (whereRow-1)*MaxCol+whereCol-1;
                with FListPtr^[CurrFNum] do
                    if ( Attr = directory ) then
                        begin { ** } { 258 }
                            { recursive for sub dir }
                            Path:=SelectedF;
                            Browse_Load_File(
                                X1,y1,x2,y2, Path,
                                UseCCDOS, SelectedF);
                            end; { ** } { 263 }
                        end; { * }

                If ArrKeyPressed then { 266 }
                case ch of
                    #75:begin { left arrow } { 268 }
                        gotoColRow(whereCol, WhereRow,
                                    InitColor);

```



```

    dec(whereCol);
    if whereCol=0      { 272 }
    then whereCol:=maxCol;
    gotoColRow(whereCol,
                whereRow, SelectColor);
end;
#77:begin { right arrow }      { 277 }
    gotoColRow(whereCol, WhereRow,
                InitColor);
    inc(whereCol);
    if whereCol=MaxCol+1      { 281 }
    then whereCol:= 1;
    gotoColRow( whereCol,
                whereRow, SelectColor);
end;
#72 : if (whereRow = 1) { up key }
    and (PageStartFNum > Maxcol)
    then      { 288 }
    begin
        gotoColRow(whereCol,
                    WhereRow, InitColor);
        dec(PageStartFNum ,MaxCol);
        gotoxy(1,1);
        InsLine ;      { 294 }
        writeOneLine(PageStartFNum);
        gotoColRow(whereCol,
                    WhereRow, SelectColor);
    end      { 298 }
    else if (whereRow > 1) then
    begin
        gotoColRow(whereCol, { 301 }
                    WhereRow, InitColor);
        dec(whereRow);
        gotoColRow(whereCol,
                    WhereRow, SelectColor);
    end;      { 306 }
#80 : { down key }
    if ( whereRow = MaxRow ) and
    ((PageStartFNum+MaxCol*MaxRow)
    <= TotalFNum )
    then { down }
    begin      { 312 }
        gotoColRow(whereCol,
                    MaxRow, InitColor);
        gotoxy(X2-X1 -2, MaxRow );
        textBackGround(blue);writeln;
        writeOneLine(PageStartFNum
                    +MaxCol*MaxRow );
        inc(PageStartFNum, MaxCol);
        gotoColRow(whereCol, MaxRow,
                    SelectColor);
    end
    else if whereRow < MaxRow then
    begin      { 324 }
        gotoColRow(whereCol,
                    WhereRow, InitColor);
        inc(whereRow);
        gotoColRow(whereCol,
                    WhereRow, SelectColor);
    end;      { 330 }
#71 : { Home key }
    begin      { 332 }
        PageStartFNum:=1;
        displayOnePage(1);
        gotoColRow (1, 1, SelectColor);
    end;
#79 : { End key }      { 337 }
    if PageStartFNum <
    totalFNum -(maxCol*MaxRow)+1 then
    begin
        PageStartFNum := 1      { 341 }
        totalFNum -(maxCol*MaxRow)+1;
        displayOnePage(PageStartFNum);
        gotoColRow (1, 1 , SelectColor);
    end;
#73 : { PgUp key }      { 346 }
    if PageStartFNum > 1 then
    begin
        dec ( PageStartFNum,
                (maxCol * MaxRow) );
        if PageStartFNum < 0
        then PageStartFNum :=1 ;
        displayOnePage(PageStartFNum);
        gotoColRow(1, 1, SelectColor);
    end;
#81 : { PgDn key }      { 356 }
    if PageStartFNum <
    (TotalFNum -maxCol*(MaxRow)) then
    begin
        inc ( PageStartFNum,      { 360 }
                (maxCol * MaxRow) );
        displayOnePage(PageStartFNum);
        gotoColRow(1, 1, SelectColor);
    end;
end { case } ;
until ch in [#27, #13];      { 366 }
if ListOK then
    FreeMem(FListPtr, sizeof(FListPtr));
end; { Browse }      { 373 }

{----- Level 2 -----}
Begin { Browse_Load_File }      { 372 }
    window(x1,y1,x2,y2);
    SetColor(InitColor);clrScr;
    MaxRow := (Y2-Y1+1 );
    MaxCol:= ( x2- x1 + 4 ) div 15;
    MakeFNameList; Browse
end; { Browse_Load_File }      { 378 }

```

# 自动生成DBASE III 菜单源程序的方法

新疆电子计算中心 周步祥

## 一、前言

随着我国微型计算机的日益增多, DBASE III 关系型数据管理系统的应用领域越来越广泛, 它在办公室自动化、企业管理、科研管理和财务等方面的应用具有许多独特的优点。纵观它在各个领域的应用形式, 绝大多数的应用系统都是用菜单来驱动的, 也就是说当我们在某一个领域开发一个应用系统时, 我们必须编制大量的菜单程序, 对于一个熟练的程序员来说, 虽然编制一个菜单程序是比较容易的, 但编制大量的菜单程序却增加了开发人员的体力上的负担, 如果能有一种工具软件能够根据我们的要求自动生成菜单源程序, 那么不仅能够减轻系统开发人员的工作量, 提高系统开发效率, 而且能够使系统中各级菜单的格式、结构更加规范, 更加统一 (本文中所述的菜单源程序均是指 DBASE III 菜单源程序)。

## 二、生成菜单源程序的方法

### 1. 菜单源程序的结构分析

由菜单驱动的应用系统都是一种树型结构, 这就要求菜单源程序的工作过程是: 首先是在屏幕上显示菜单的内容, 然后通过键盘选择所选项, 最后是根据选择的内容去执行不同的子程序。换句话说一个 DBASE III 菜单源程序应该由以下三个部分组成: 菜单内容显示部分、选择机构以及根据选择内容转移的部分 (图一示出了一个菜单源程序的基本框图结构)。

用 DBASE III 的命令来写图一所示结构的源程序是很容易的, 这里不再赘述。这里我们所要强调的是这种菜单源程序都具有的结构。事物共同具有的东西称为共性, 要区别不同的事物还必须找出他的特殊性。这里结构是菜单源程序的共性, 那么不同的菜单, 形成它们的源程序的不同之处又是由哪些因素决定的呢?

```
end. { Unit }
```

```
Program Test_Load_File_Unit; { 382 }
($M 4084, 0, 16384) { stack and Heap }
Uses Dos, Crt, graph, ULoadF;
Var
  Count      : integer;
  UseCCDOS   : boolean;   { 387 }
  DirPath, SelectedFileName : string;
```

```
Begin { Test_Load_File_Unit }
  CheckBreak:=true; { 391 }
  {TextMode(C80);} {Color 80X25}
  ClrScr;
  write('Enter dir path :');
  readln(DirPath);
  if (DirPath = '') { 396 }
  then getDir(G, DirPath); {Default dir}
  if DirPath[Length(dirPath)]='\ '
  then dirPath:=dirPath+'*.x';
```

```
writeln(DirPath); UseCCDOS := False;
{ if use ccdos, let UseCCDOS:= true }
if UseCCDOS then { 401 }
```

```
begin
  directVideo:=False;
  SetHiresMode;
  SetHiresColor(Yellow, false);
end
else directVideo:=True; { 407 }
```

```
Browse_Load_File(10, 8, 70, 18, DirPath,
  UseCCDOS, SelectedFileName);
```

```
ClrScr;
if SelectedFileName=#27 { 412 }
then writeln('Selection is aborted ')
else
begin
  writeln('SelectedFileName= ',
    SelectedFileName);
  Exec('\Command.Com', '/C Type '+
    SelectedFileName);
```

```
end;
Readln; TextMode(LastMode);
End. { Test_Load_File_Unit } { 422 }
```

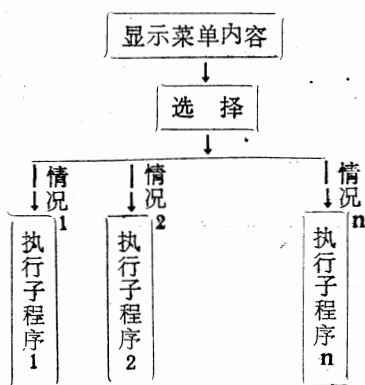


图1

通过分析,我们发现形成不同菜单的源程序的不同处有以下几点:

- (1) 屏幕背景不同,即清屏的范围不同;
- (2) 菜单有无标题不同;
- (3) 菜单有无方框不同,即在不在菜单内容上加上方框;

- (4) 菜单的显示位置不同;
- (5) 菜单的内容不同;
- (6) 菜单的各项转向子程序不同;
- (7) 菜单源程序的文件名不同。

## 2. 自动生成菜单源程序的软件设计

通过以上分析,我们找到了菜单源程序的共性以及他们的特殊性,自动生成菜单源程序的软件就是基于这两点进行设计的,首先是通过键盘接收各个菜单源程序的不同点,然后将这些东西加到他们共同具有的结构当中去就形成了一个我们所需要的菜单源程序。

## 三、自动生成菜单源程序实例

我们用BASICA语言编写了一个能够自动生成菜单源程序的程序,生成的菜单源程序是DBASE III—Plus源程序,菜单是下拉式的,控制菜单选用了↑↓键、HOME键以及回车键,↑↓键用上下移动光标;回车键用来确认所选择的项目;HOME键用于返回上一级菜单或退出主菜单。

下面给出了自动生成DBASE III—Plus下拉式菜单的源程序的程序清单,在使用该程序时

只需要回答计算机提出的七个问题(即给出菜单的特殊性),计算机将会根据你回答的内容自动生成DBASE III—Plus的菜单源程序。程序的90—240行是用来获取每个菜单源程序的特殊性的,在通过键盘输入这些内容时必须按照提示的内容的要求输入,例如在输入菜单源程序文件名时必须加上后缀.PRGM,还必须说明一点的是,为了编程的方便,菜单的分支所指向的子程序的文件名的取名方式是:文件名前缀十分支号。例如某一菜单的分支子程序的文件名的前缀是AA,则它的第一条分支的文件将自动设定为AA1,第二条分支为AA2,以此类推。程序清单中的250—360行是将按要求形成的DBASE III—Plus菜单源程序写入规定的文件中(文件名也是由用户在前面输入的)。

在使用该程序时,生成一个菜单源程序(抛开键盘输入的时间不足一分钟,这样就大大提高了编制菜单源程序的效率。

本程序是在具有25行汉字显示功能的微机上设计的,生成的菜单源程序是DBASE III—Plus的下拉式菜单的源程序,如果要在其它种类的机上生成DBASE III—Plus或DBASE III的其它形式的菜单源程序只需要将本程序中的有关显示行数及其它个别语句作适当的变动就可以了。

## 四、结束语

自动生成源程序是一个非常复杂的问题,本文只是根据实际工作中的具体问题,从提高编程效率的角度介绍了一个简单的方法,本程序在完成《新疆区科委办公室办公自动化辅助支撑系统》的过程中得到了良好的运用,它只是根据本系统设计思路及要求加以总结及编制的,要想形成一个完整系统,还必须进行进一步的充实、完善。

## 好消息

如同“扫黄”一样,对严重威胁计算机信息安全的病毒也须群起而扫之!

为了扫毒和免疫,本刊将于最近出版《计算机病毒大观》论文集,热忱欢迎赐稿和订阅!

此论文集每册估价10元。

联系人: 本刊编辑部 邓嘉澍



```

10 CLS: KEY OFF: DIM A$ (20): DEF INT A-Z
20 LOCATE 2, 15: PRINT "*****"
30 LOCATE 3, 15: PRINT "*****"
40 LOCATE 4, 15: PRINT "    自动生成 DBASEIII 菜单源程序    "
50 LOCATE 5, 15: PRINT "*****"
60 LOCATE 6, 15: PRINT "                    一九八九年十一月                    "
70 LOCATE 7, 15: PRINT "*****"
80 LOCATE 8, 15: PRINT "*****"
90 LOCATE 10, 5: PRINT "建菜单前清屏范围 (0, 0-23, 79    不输为不清屏: (    )"
100 LOCATE 10, 57: INPUT " ", A$, A1$
110 LOCATE 10, 67: INPUT " ", B$, B1$
120 LOCATE 11, 5: INPUT "菜单标题 (不输为没有标题): ", BT$
130 LOCATE 12, 5: INPUT "菜单左上角位置 (0, 0-23, 79): ", WZ$, WZ1$
140 LOCATE 13, 5: INPUT "菜单内内需加方框吗 (Y/N): ", KW$: IF KW$ < > "y" AND
    KW$ < > "Y" AND KW$ < > "n" AND KW$ < > "N" THEN 140
150 LOCATE 14, 5: INPUT "菜单源程序文件名 (后加缀 .PRG): ", JMW$
160 LOCATE 15, 5: INPUT "菜单分支文件名前缀: ", QZ$
170 LOCATE 12, 50: PRINT "菜单内容"
180 H=VAL (WZ$)
190 IF BT$ < > " " THEN H=H+1
200 IF KW$ = "Y" OR KW$ = "y" THEN H=H+2
210 FOR I=1 TO 23-H
220 LOCATE 12+I, 50: INPUT " ", A$(I): IF KD < LEN(A$(I)) THEN KD=LEN(A$(I))
230 IF A$(I) = " " THEN HS=I-1: I=24+H
240 NEXT I
250 OPEN WJM$ FOR OUTPUT AS #1
255 PRINT #1, "set talk off"
260 PRINT #1, "@": A$: " ": A1$: "clear to": B$: " ": B1$
270 PRINT #1, "e=1"
280 IF BT$ < > " " THEN PRINT #1, "@": WZ$: "-1,": WZ1$ "say,": BTs "1"
290 IF KW$ = "Y" OR KW$ = "y" THEN PRINT #1, "@": WZ$: " ": WZ1$: "to":
    VAL(WZ$)+HS+: " ": VAL(WZ1$)+KD+5: "dunble": WZ=VAL(WZ$)+1:
    WZ1=VAL(WZ1$)+1 ELSE WZ=VAL(WZ$): WZ1=VAL(WZ$)
300 FOR I=1 TO HS
310 PRINT #1, "c": MID$(STR$(I), 2, LEN(STR$(I))-1): "=1" A$(I): " "
320 NEXT I
330 FOR I=0 TO HS-1
340 PRINT #1, "@": WZ+I: " ": WZ1: "say c": MID$(STR$(I+1), 2, LEN(STR$(
    (I-1))-1)
350 NEXT I
370 PRINT #1, "set color to n/w"
380 PRINT #1, "@": WZ: " ": WZ1: "say cl"
390 PRINT #1, "el=str(e,1)"
400 PRINT #1, "do while .1."
410 PRINT #1, "@ 1, 65 say time()"
420 PRINT #1, "ks=inkeA()"
430 PRINT #1, "do case"
440 PRINT #1, "    case ks=5"
450 PRINT #1, "        set color to n/w"
460 PRINT #1, "        @": WZ: "-1: "+e: " ": WZ1: "say o&el"
470 PRINT #1, "        set eolor ton/w"
480 PRINT #1, "        e=mod(e-1, "HS+1: " )"
490 PRINT #1, "        if e=0"
500 PRINT #1, "        e=": HS

```

```

510 PRINT #1, "    endif"
520 PRINT #1, "    ei=str(e,1)"
530 PRINT #1, "    @ "; WZ-1, " "; +e, " "; WZ1; " say c&el"
540 PRINT #1, "    case ks=24"
550 PRINT #1, "    set color to n/w"
560 PRINT #1, "    @ "; WZ-1, " "; +e, " "; WZ1; " say c&el"
570 PRINT #1, "    set color to n/w"
580 PRINT #1, "    e=mod(e+1, " ; HS+1; " )"
590 PRINT #1, "    if e=0"
600 PRINT #1, "        e=1"
610 PRINT #1, "    endif"
620 PRINT #1, "    el=str(e,1)"
630 PRINT #1, "    @ "; WZ-1; " +e, " WZ1; " say c&el"
640 PRINT #1, "    case ks=1"
650 PRINT #1, "    set color to w/n"
660 PRINT #1, "    @ "; WZ-2; " "; WZ-2; " clear to "; WZ+HS+1; " "; WZ1+KD+4
670 PRINT #1, "    exit"
680 PRINT #1, "    case ks=13"
690 PRINT #1, "    p= / /"
700 PRINT #1, "    set color to w/n"
710 PRINT #1, "    do while .not.p$ 'YN'"
720 PRINT #1, "    @ 23, 20 say '您是要进行' +c&el+ '吗 (Y/N) ?' get p pict 'PY'"
730 PRINT #1, "    read"
740 PRINT #1, "    enddo"
750 PRINT #1, "    if p='Y'"
760 PRINT #1, "        do "; QZ$; " &el"
770 PRINT #1, "        exit"
780 PRINT #1, "    else"
790 PRINT #1, "        @ 23, 0 clear to 23, 79"
800 PRINT #1, "        set color to n/w"
810 PRINT #1, "    endif"
820 PRINT #1, "endcase"
830 PRINT #1, "enddo"
840 PRINT #1, "    @ "; WZ-2; " "; WZ1-2; " clear to, " ; WZ+HS+1; " "; WZ1+KD
850 PRINT #1, "return"
860 CLOSE

```

## 汉字下拉式菜单

深圳水产供销公司 刘宇菁

### 一、概述

随着微型计算机在国内的普及化, 为方便国内用户, 必须使英文菜单汉字化。使用汉字菜单, 用户不需打入汉字, 可以快速地进行操作。CC-DOS 汉字操作系统已经广泛地用于 IBM PC 微型系列机及其兼容机上, 这就有了使用汉字菜单的可能性。

本文以《浅介下拉式菜单》一文(见本刊1989年第5期)为基础进行叙述。

### 二、设计思想

由于CC-DOS是用图形方式实现的, 因此在菜单系统中的与屏幕相关的部分必须重写, 而且为了快速, 直接读写屏幕的方法与显示卡是相关的。本文的实例程序以CGA为蓝本。

CGA的显示缓冲区起址为B8000000H, 使用640×200模式时(CCDOS初始化成这个模式), 每一个二进制位代表一个像点, 0为不显示这点, 1为显示该点。CGA的扫描线从上到

## 用BASIC语言实现全屏幕数字编辑

湖北丰山铜矿计算机站

段宝珠

BASIC语言以其简单易学而得到广泛应用,但由于功能比较低,编写具有全屏幕编辑功能的应用程序比较困难。这里向读者介绍一段经长期使用效果基本满意的全屏幕数字编辑子程序,以求推进BASIC语言的更好应用。

这是从实用数据处理程序中摘录的一段数字编辑子程序,稍加整理,增加了一组模拟数字(第120行),在IBM-XT机上调试通过,可以独立运行演示。其主要功能及对应行号简要介绍如下(括号内的数字为程序行号):

1. 用光标移动键可控制光标在全屏上下左右移动,并定位于数据项的第1位(570-600);

2. 光标可从任一位置跳跃定位于数据的第一项或最末一项(550-560);

3. 此程序段专用于编辑数字型数据,若误敲非数字键则不接受并发出声响告警,负数的负号必须在第1位敲入,正数的正号一律省略,小数点多于1个时不告警,但第2个小数点及以后的数字无效(500-530);

4. 录入数据过程中可用退格键删除光标左侧的数字,当敲入的数字全部删除后将恢复显示原始数据(480-490);

5. 有两种方式退出编辑:①废除编辑(不存盘)退出:Ctrl+Q(420,240);②保留编辑(存盘)退出:Ctrl+W或Ctrl+End(430,250);

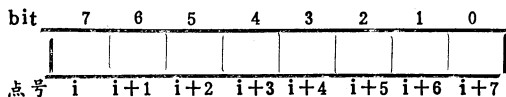
6. 每一数据项最多可敲入15位数字(包括负号和小数点),当敲入第16位时不再接受并告警(440),敲入的数字值应在-99999999.99至99999999.99之间,目的是保护屏幕格式不被冲乱(670-690);

7. 录入数字时可以用回车键结束,也可以用光标移动键或退出键结束(390,450);

第130-170行定义了一组常量,其中MAX:数据项最大序号;REND:光标停止右移的数据项序号;第140-160行的12项为光标移动键和功能键编码;第170行的4项分别为负号,小数点,0和9的编码(可查阅ASCII编码集)。

因篇幅所限程序中删去了翻页处理段,并对程序行长度作了适当调整,以求压缩版面,但牺牲了可读性,不过对熟悉BASIC语言的读者读懂这段小程序并不困难。当数据项序号大于92时可分几页编辑,用PgUp和PgDn实现前后翻页,这里不再赘述。

下从0开始标计,共200线,在偶数扫描线内的像点存于起址为B8000000H的缓冲区中,在奇数扫描线内的像点存于起址B8002000H的缓冲区中。像点在字节内的顺序是按照屏幕位置从左至右储存,即:



以上这些,都是编程资料。

三、实例程序:(因程序较长,篇幅所限,从略)

## 书 讯

本刊现有以下技术资料欢迎函购(含邮费):

开户银行:成都跳伞塔分理处;

帐 号:89501299

资料名称	每册单价(元)
1989年全国计算机应用研究学术会论文集	39.00
APPLE II 微型机实用维修技术	2.50
静电复印机维修指南	11.00
IBM PC磁盘操作系统	3.75

联系人:本刊读者服务部 唐大利

```

100 '----- << 全屏数据编辑 >> -----
110 CLS:KEY OFF
120 DIM NUM$(90),Z$(15):FOR I=0 TO 90:NUM$(I)=10##I:NEXT I
130 MAX=90 : REND=MAX-22 : CTRLQ=17
140 HOME=71 : ENDB=79 : PGUP=73 : PGDN=81
150 UP=72 : DOWN=80 : LEFT=75 : RIGHT=77
160 ENTER=13 : BACK=8 : CTRLW=23 : CTRL=117
170 NEGATIVE=45 : POINTS=46 : ZERO=48 : NINE=57
180 SCREEN 0,0,0:WIDTH 80
190 LOCATE 1,1:COLOR 15,9:PRINT SPACE$(32);"edit of number";SPACE$(33);
200 LOCATE 25,1:PRINT SPACE$(32);"edit of number";SPACE$(33);:COLOR 7,0
210 FOR I=1 TO MAX:GOSUB 270:IF I<=69 THEN COLOR 3:PRINT "|";
220 NEXT I:I=1:CHANGE$=""
230 GOSUB 330:IF CHANGE$="" THEN 230
240 CLS:IF CHANGE$="F" THEN END
250 FOR I=1 TO MAX:PRINT USING "#####.####";NUM$(I);:NEXT I:END
260 '
270 '----- 数据显示 -----
280 COL=INT((I-1)/23):ROW=(I-23*COL)+1
290 LOCATE ROW,COL*20+1:COLOR 6:PRINT USING "###";I;:PRINT " ";:COLOR 7
300 IF ABS(NUM$(I))<.00005 THEN PRINT SPC(15);:RETURN
310 PRINT USING "#####.####";NUM$(I);:RETURN
320 '
330 '----- 定位编辑 -----
340 COL=INT((I-1)/23):ROW=(I-23*COL)+1
350 LOCATE ROW,COL*20+1:COLOR 15:PRINT USING "###";I;:PRINT " ";
360 COLOR 14,4:PRINT USING "#####.####";NUM$(I);:COLOR 7,0
370 LOCATE ROW,COL*20+5:J=0
380 K$=INKEY$:IF K$="" THEN 380
390 IF LEN(K$)=1 THEN 410 ELSE BJM=ASC(RIGHT$(K$,1)):IF J>0 THEN GOSUB 670
400 GOSUB 540:RETURN
410 SJM=ASC(K$)
420 IF SJM=CTRLQ THEN CHANGE$="F":RETURN
430 IF SJM=CTRLW THEN CHANGE$="T":IF J=0 THEN RETURN:ELSE GOSUB 670:RETURN
440 IF J>14 AND SJM<>ENTER AND SJM<>BACK THEN BEEP:GOTO 380
450 IF SJM=ENTER THEN 460 ELSE 480
460 IF J=0 THEN GOSUB 270:IF I=MAX THEN BEEP:RETURN ELSE I=I+1:RETURN
470 GOSUB 670:GOSUB 270:IF I=MAX THEN RETURN:ELSE I=I+1:RETURN
480 IF SJM<>BACK THEN 500
490 IF J=0 THEN BEEP:RETURN:ELSE GOSUB 640:IF J=0 THEN RETURN:ELSE GOTO 380
500 IF SJM<>NEGATIVE THEN 520
510 IF J=0 THEN Z$(0)=K$:J=1:PRINT K$;:GOTO 380 ELSE BEEP:GOTO 380
520 IF SJM<ZERO AND SJM<>POINTS OR SJM>NINE THEN BEEP:GOTO 380
530 Z$(J)=K$:J=J+1:PRINT K$;:GOTO 380
540 IF BJM=CTRL= THEN CHANGE$="T":RETURN
550 IF BJM=HOME THEN IF I=1 THEN BEEP:RETURN:ELSE GOSUB 270:I=1 :RETURN
560 IF BJM=ENDB THEN IF I=MAX THEN BEEP:RETURN:ELSE GOSUB 270:I=MAX :RETURN
570 IF BJM=UP THEN IF I=1 THEN BEEP:RETURN:ELSE GOSUB 270:I=I-1 :RETURN
580 IF BJM=DOWN THEN IF I=MAX THEN BEEP:RETURN:ELSE GOSUB 270:I=I+1 :RETURN
590 IF BJM=LEFT THEN IF I<=23 THEN BEEP:RETURN:ELSE GOSUB 270:I=I-23:RETURN
600 IF BJM=RIGHT THEN IF I>=REND THEN BEEP:RETURN:ELSE GOSUB 270:I=I+23:RETURN
610 BEEP:RETURN
620 '
630 '----- 光标定位 -----
640 Y=CSRLIN:X=POS(0):LOCATE Y,X-1:PRINT " ";:LOCATE Y,X-1:J=J-1:RETURN
650 '
660 '----- 字符串转换 -----
670 NUM$="":FOR K=0 TO J-1:NUM$=NUM$+Z$(K):NEXT K:BUFF#=VAL(NUM$)
680 IF BUFF#>999999999.99# OR BUFF#<-999999999.99# THEN BEEP:RETURN
690 NUM$(I)=BUFF#:RETURN

```



# 屏幕划到与翻页技术

通信工程学院图书馆计算机室 王 渤

**提要:** 本文介绍运用关系数据库, 在屏幕上实现形象地签到, 以及用户的信息显示的翻页技术。使数据库更好地发挥其数据管理的作用, 并可将该文中所介绍的方法移植到其它语言中, 为广大用户更灵活的开发新颖的数据库语言, 提供一种灵活的技术方法。特别是对图书馆的期刊验收划到, 更具其形象直观的特点。

## 一、引言

目前在PC机上使用的关系数据库, 大多是DbaseIII+的各种版本, 而另一新颖数据库FOXBase+也正日受用户青睐(在此也向广大用户推荐之, FOXBase+已有汉化版本, 实用于IBM PC和VAX机, 较DbaseIII+又增加了许多新功能和新函数, 且比原DbaseIII+的许多功能都有所加强, 原DbaseIII+的程序不用任何修改即可在FOXBase+中运行, 经编译过的FOX程序运行速度将快于Dbase6倍多), 我们利用它的INKEY( )函数配合使用@语句, 实现了屏幕划到及翻页技术。

## 二、INKEY函数

INKEY( )函数是一接收击键函数, 形式为: INKEY(X), 自变量X是整数形, 表

示等待击键的时间(秒数), 该函数返回一整数, 对应击键的ASCII码值。若自变量取值“0”, 则将无限期等待, 我们采用自变量取0值时。如:

K=INKEY(0)

击键Ctrl-W则回送其ASCII值K=23

该功能类似于WAIT语句, 但WAIT接收的是击键“字符”, 且只能识别其ASCII值大于32(空格)而小于126(≈)的字符, 在这个局限内WAIT语句也可实现INKEY( )的功能, 如:

WAIT / / TO P;  
K=ASC(P)

## 三、屏幕划到问题分析

先看下面这张表, 这是从屏幕上硬拷贝下来的(期刊划到卡的形式):

刊名：计算机技术				刊号：ABC004								架号：372	
年	卷	一	二	三	四	五	六	七	八	九	十	十一	十二
1984	23	■			V	V	V	V	V	V			V
		V	V	V	V	V		V		V	V		
1983	23	V	V	V	V	V		V	V	V	V	V	
		V		V			V	V	V		V		
半月刊		编者：北京计算机中心										来源：公开	

→右移←左移↑上移↓下移(sp)删所打的VHS打VENTER回车CTRL-END退出

根据表下说明, 这张表在屏幕上用光标键可在表内任意移动光标“■”, 打入“INS”键则光标处出现“V”, 打空格键则删掉之, Ctrl-End则将这些信息保存退出, 上表中期刊每年共24期, 即半月刊。此表及其内容下次划到时再

重新调出。

要实现上述功能, 几个关键问题是:

①将原来的系统光标“—”改为与划到卡方格等同(或略小)的方块光标“■”

②设计击键控制, 使光标在划到卡中能随意

移动,并用指定键划到

③屏幕划到信息的保存和恢复

#### 四、技术实现

针对上述三个关键问题,分别用以下方法解决:

①字块形光标,用@. .GET语句实现,用@. .SAY语句清除光标,光标的大小用GET后字符变量的长度决定

②击键控制,即用选择语句CASE来判断IN-KEY的接收值

③设二维数组VA(2,12),每一元素对应屏幕上的一期,记录“到”或“否”,如:VA(1,2)对应表中第一行第二格;期刊库中有一“划到字段”,用来记录划到情况,用数字加一“.”表示所到期数,如:1.2.7.表示该刊已到第一二七期。

下面是一例程序,实现对半月期刊库的划到(用类语言编写,读者若有兴趣可再联系原程序):

第一步:调出欲划到的期刊划到卡(可通过多种途径查找)

```

;
;
USE DBQK ; 打开期刊库
@1,1 SAY "请输入欲划到的期刊名:"
GET NAME
READ
SET INDEX TO DBQKIND*
; 打开刊名索引
SEEK NAME ; 查找欲划到期刊
; (利用GDOS划线语句在屏幕上画表,
此处略)

```

@3,0 SAY "→右移←左移↑上移↓下移  
(sp) 删所打的√ IHS打√ CTRL-END退出"

@1,1 SAY "刊名:" + 刊名 + "刊号:"  
+ 刊号 + "架号:" + 架号 ; 显示表头

@2,1 SAY "年卷 一 二 三 四 五  
六 七 八 九 十 十一 十二"

```

;
X=3; 左上角划到方格坐标为X=3,Y=9
Y=9
STOR " " TO VA
I=1
IF I<10

```

```

IK=STR(I,1) + "."
ELSE
IK=STR(I,2) + "."
ENDIF
DO WHILE I<=24 ; 查24期中那一期已到
J=AT(IK,划到字段)
IF J=0 ; 第IK期未到
I=I+1 ; 查下一期
Y=Y+3 ; 光标下移一格
ELSE
IF I>12
X=X+1 ; 超过12期则换行
Y=9
@X,Y SAY VA(X-2,I-12)
ELSE
@X,Y SAY VA(X-2,I)
ENDIF
I=I+1
Y=Y+3
ENDIF
ENDDO ; 调出原表结束

```

第二步:实现具体划到,在屏幕上修改划到卡,先赋初值

```

;
X=3
Y=9
I=1 ; VA的第一个元素
DO WHILE .T.
@X,Y GET VA(X-2,I); 建立光标
KEY=INKEY(0) ; 等待输入
@X,Y SAY VA(X-2,I); 消除光标
DO CASE
CASE KEY=4 ; 键入 "→"
Y=Y+3
I=I+1
IF Y>45; 光标在第一行尾则转为第二行
I=1
Y=9
X=X+1
ENDIF
IF X>4
X=3
ENDIF
CASE KEY=19 ; 键入 "←"
Y=Y-3
I=I-1
IF Y<9 ; 转为第一行
I=1
Y=9
X=X-1
ENDIF
IF X<3
X=3
ENDIF
CASE KEY=5 ; 键入 "↑"
IF X>3
X=X-1
ENDIF
CASE KEY=24 ; 键入 "↓"
IF X=3

```

```

X=X+1
ENDIF
CASE KEY=23 ; 键入 "Ctrl-End"
EXIT ; 跳出循环
CASE KEY=22 ; 键入 "Ins"
VA(X-2, I)="z"
CASE KEY=32 ; 空格键
VA(X-2, I)=" "
ENDCASE
ENDDO ; 划到结束

```

第三步：保存划到结果，将划到卡转化为数据保存

```

I=1
J=1
REPLACE划到字段 WITH " "
DO WHILE I<=24
K=1
IF I>12
J=2
K=I-12
ENDIF
IF I<10; ; 把VA变换成每一期的标志,
II=STRING(I, 1) 如: 2.为第二期3.为
第三期.....
ELSE
II=STRING(1, 2)
ENDIF
IF VA(J, K)="z"
REPLACE划到字段 WITH TRIM (划
到字段)+II+"."
ENDIF
I=I+1
ENDDO
; 程序结束!
RETURN
;

```

从以上程序可见，击键与屏幕显示需经过变换，可根据自己的意愿改用其它键，如用：“J”，

L, I, K”分别作为光标键的“左右上下”等，此时即可用“WAIT”语句代之。（注：上例程只对表中前两行进行划到操作，读者有兴趣很容易将其改成全屏操作，及实现各种期刊划到）

该方法若用汇编语言设计会产生更令人满意的效果。

## 五、翻页技术

实际上这也是上面技术的其中一部分的应用，此只举一例说明：

```

;
USE DBASC
GO TOP
@3, 0 SAY "↑翻上页↓翻下页End退出"
LOP=0
I=1
DO WHILE LOP<>6 ; End键
GO I ; 此处是显示程序，略
LOP=INKEY(0)
DO CASE
CASE LOP=5 ; "↑"
IF I>1
I=I-1
ENDIF
CASE LOP=24 ; "↓"
IF .NOT. EOF
I=I+1
ENDIF
ENDCASE
ENDDO
RETURN

```

同样道理，若有一张表格在一屏显示不下，也可以用此法分页显示（也可左右分页），此不赘述。

# 交互式任意大小屏幕图形的存贮与再现

华中理工大学电信系 李 晖

## 一、功能

在设计动画、菜单等时，常常要将设计的图形存贮在磁盘上，以便在后续设计或程序中调用。在BASICA中，有BSAVE和BLOAD语句可完成上述功能。但要存、取用户指定的任意区域内图形时，用它们则很不方便，且一般要占用大量磁盘空间。为此，笔者设计了一键盘交互式程序，实现了下述功能：

1. 只要通过光标键来指定屏幕上任意两

点，即可把以这两点为对角顶点的矩形区域内图形作为数据文件存于磁盘，并可实现把已存于磁盘的图形文件重现于屏幕上原位置（具有极好的定点功能）。由于只存贮指定区域内图形，因此生成的数据文件大小仅仅取决于该区域本身大小（自然可以极小）。

2. 因为具有定点重现功能，所以可方便地实现多个已存贮图形的重构。

3. 可任选高分辨率或中分辨率两种图形模

式。

4. 可编译成 .EXE 文件, 这样在运行某些 .EXE 或 .COM 文件时, 只要能从中退出, 便可用本程序对这些程序的图形进行摄象 (或多或少还有点 WINDOWS 中摄象功能的味道)。

5. 本程序中的存贮和重现图形的思想方法可广泛移植于用户的应用程序中, 可容易地在用户自己的程序中实现存取图形的功能。这是某些应用绘图软件如 PCPAINT 等所无法实现的。

## 二、原理

IBM PC 的彩色图形显示器有两种图形模式: 中分辨率模式  $320 \times 200$ , 高分辨率模式  $640 \times 200$ 。在这两种模式下, 屏幕上的每一个象素点分别由屏幕存贮区中一个字节的两 (中分辨率), 一位 (高分辨率) 来映象。BASICA 中的 GET 语句就是通过把屏幕上的一矩形区域所对应的屏幕存贮区中的相应字节 (位) 的内容存于一数组中, 来实现存下该矩形区内每一象素的彩色码值, 从而达到暂存图形的目的。

对于语句  $GET (XS_1, YS_1) - (XS_2, YS_2)$ , S%, 由于  $XS_1$  (或  $XS_2$ ) 所对应的映象位可能并不在整字节处, 因此如果从分析数组 S% 的内容与图形的对应关系着手, 势必十分繁琐, 事实上这也是不必要的。但若从另一方面着手, 即把数组 S% 中的每一个元素作为数据文件的一个记录都存下来, 而在重现时再把数据文件的每一个记录分别返赋给数组 S% (从元素 S% (0) 开始) 的元素, 再利用 PUT 语句, 就可以把原象重现。

另外, 为实现在原位置重现图形, 把  $XS_1$ ,  $YS_1$  也分别作为一个记录存下。

由于 BASICA 中无可变数组, 为了能存贮任意大小的图形 (最大为满屏幕), 特把暂存图形的整数数组 S% 定义为最大情况 8102 个元素 (16204 字节)。实际应用时, 在用户指定矩形的一对对角顶点  $(XS_1, YS_1)$ ,  $(XS_2, YS_2)$  后, 由公式

$$IIM = \text{INT}(4 + (YS_2 - YS_1 + 1) * \text{INT}(((XS_2 - XS_1 + 1) * \text{BIT} + 7) / 8) + 1)$$

可算出所需的实际字节数, 而整数数组 S% 则要用  $\text{ENDN} = \text{INT}(IIM / 2) + 1$  个元素来暂存图

形, 所以在存贮数据文件时只用存  $(\text{ENDN} + 1)$  个数组 S% 的元素。而为了方便以后读数据文件, 把  $\text{ENDN}$  也作为一个记录存下。在读数据文件时, 先读出  $\text{ENDN}$ ,  $XS_1$ ,  $YS_1$ , 再把余下的内容 (记录 4 至  $\text{ENDN} + 4$ ) 全部赋于数组 S% 的第 0 ~  $\text{ENDN}$  个元素中, 这样, 用 PUT  $(XS_1, YS_1)$ , S%, PEST 便可重现图形于原处。

由于要对磁盘文件进行操作, 在程序中加入了错误陷阱, 提示可能出现的诸如 "File not found", "Disk full" 等错误信息, 可保证程序在意外情况下仍然能正常运行。

## 三、操作菜单

为实现能任意指定存贮区域, 在程序开始设计了键盘交互式主控程序。

### A. 存图形

1. 选按 "1" (高分辨率)、"2" (中分辨率彩色) 或 "3" (中分辨率黑白) 键, 来选择与当前屏幕图形兼容的图形模式。

2. 通过上、下等八个光标键可任意移动当前图形光标——一个象素点 (程序开始置它于屏幕中心。为醒目, 本程序使它不停闪烁)。在光标到达用户指定区域的左上顶点时, 按 "1" 键; 再移动光标至指定区域的右下顶点, 按 "2" 键, 这样用户要存贮的图形区域便被指定 (在按 "Ctrl-S" 开始存入磁盘之前, 可重复上述过程来重新确定存贮区域)。

3. 按 "Ctrl-S", 便会出现一小窗口, 要求输入文件名, 注意该文件名不能超过八个字符 (此时图形已被暂存于 S% 中, 屏幕上被覆盖亦无妨! )。

### B. 重现图形

按 "R" 键便可重现已存于磁盘上的图形文件 (也要先回答文件名)。

在存取过程中, 屏幕右上方都有倒数数字显示进程。并可按 "Ctrl-Break" 中断程序执行。整个程序可按 "ESC" 键正常退出。

另外, 为方便用户, 还设计了缺省回答方式: 若不按 "1", 直接按 "Ctrl-s", 将只存贮以当前光标为左上顶点的  $51 \times 41$  矩形区域; 若

只按“1”，直接按“Ctrl-S”，将只存贮以

(XS<sub>1</sub>, YS<sub>1</sub>)为左上顶点的51×41矩形区域。

为避免覆盖原有图形，这些菜单并不显示，但在后面附的源程序清单中还作为注释保留。

#### 四、结束语

本程序已在 IBM PC, IBM PC/XT, IBM PC/AT, GW0520-CH等多种机型上调试通过。

```

10 REM      -- SAVE & REPLAY IMAGE --      7/10/1989      LIHUI
50 DIM S$(8102)
51 ON ERROR GOTO 1000
55 KEY OFF
60 AS=INKEY$:IF AS="" THEN 60
65 IF AS="1" THEN 80
70 IF AS="2" THEN 90
75 IF AS="3" THEN 95
78 GOTO 60
80 SCREEN 2,0:CCC=639:X=320:Y=100:BIT=1:GOTO 102
90 SCREEN 1,0:CCC=319:X=160:Y=100:BIT=2:GOTO 102
95 SCREEN 1,1:CCC=319:X=160:Y=100:BIT=2:GOTO 102
102 'LOCATE 3,4:PRINT CHR$(4);" SMART SAVE ";CHR$(4)
104 'LOCATE 3,21:PRINT "1--Affirm start point 2--Affirm end point.R --- Replay
"
105 'LOCATE 4,21:PRINT "Ctrl-S --- Begin Save Ctrl-Break --- Stop "
106 'LOCATE 2,21:PRINT CHR$(24);" ";CHR$(25);" .ect --- Move point Esc -- Exit
"
150 IF POINT(X,Y)<1 THEN SPO1=0 ELSE SPO1=1
154 PSET(X,Y):FOR H2=1 TO 5:NEXT:PRESET(X,Y)
160 IF SPO1=1 THEN PSET(X,Y):SPO1=0
162 AS=INKEY$:IF AS="" THEN 162
164 IF RIGHT$(AS,1)=CHR$(80) THEN Y=Y+1:GOTO 195
166 IF RIGHT$(AS,1)=CHR$(72) THEN Y=Y-1:GOTO 195
168 IF RIGHT$(AS,1)=CHR$(77) THEN X=X+1:GOTO 195
170 IF RIGHT$(AS,1)=CHR$(75) THEN X=X-1:GOTO 195
172 IF RIGHT$(AS,1)=CHR$(71) THEN X=X-1:Y=Y-1:GOTO 195
174 IF RIGHT$(AS,1)=CHR$(79) THEN X=X-1:Y=Y+1:GOTO 195
176 IF RIGHT$(AS,1)=CHR$(73) THEN X=X+1:Y=Y-1:GOTO 195
178 IF RIGHT$(AS,1)=CHR$(81) THEN X=X+1:Y=Y+1:GOTO 195
180 IF AS="1" THEN VV=VV+1:XS1=X:YS1=Y:IF VV>2 THEN VV=1
182 IF AS="2" THEN VV=VV+1:XS2=X:YS2=Y:IF VV>2 THEN VV=2
184 IF AS=CHR$(19) THEN GOSUB 200
186 IF AS="R" OR AS="r" THEN GOSUB 500
188 IF AS=CHR$(27) THEN END
190 GOTO 162
195 IF X<0 OR X>CCC OR Y<0 OR Y>199 THEN BEEP:GOTO 162 ELSE 150
200 GOSUB 208
201 GOSUB 325
202 GOSUB 400
205 RETURN
208 VV=VV+1
209 ON VV GOTO 210,240,270
210 XS2=X+50:YS2=Y+40:XS1=X:YS1=Y
214 IF XS2>CCC OR YS2>199 THEN LOCATE 3,15:PRINT "Too close to edge":GOTO 162
216 IIM=INT((4+41*INT((51*BIT+7)/8+1))/2+1):GET(XS1,YS1)-(XS2,YS2),S$:RETURN
240 XS2=XS1+50:YS2=YS1+40
242 IF XS2>CCC OR YS2>199 THEN LOCATE 3,15:PRINT "Too close to edge":GOTO 162
246 ENDN=INT((4+41*INT((51*BIT+7)/8+1))/2+1):GET(XS1,YS1)-(XS2,YS2),S$:RETURN
270 IIM=INT(4+(YS2-YS1+1)*INT((XS2-XS1+1)*BIT+7)/8+1)
274 ENDN=INT(IIM/2)+1:GET(XS1,YS1)-(XS2,YS2),S$:RETURN
325 A=2:B=4:C=10:GOSUB 650:LOCATE 3,15:INPUT"File name:";FILES
330 IF LEN(FILES)>8 THEN 325
340 FILES=LEFT$(FILES,8)
360 RETURN
400 A=2:B=4:C=10:GOSUB 650
405 LOCATE 3,15:PRINT "Please wait .... "

```



```

410 OPEN FILES FOR OUTPUT AS #1
415 SSS=ENDN:WRITE #1,SSS
416 SSS=XS1:WRITE #1,SSS
417 SSS=YS1:WRITE #1,SSS
420 FOR FIL=0 TO ENDN
430 SSS=S%(FIL)
440 WRITE #1,SSS
450 LOCATE 3,30:PRINT (ENDN-FIL)
460 NEXT FIL
470 CLOSE #1
480 BEEP:LOCATE 3,15:PRINT "    Complete ... "
490 RETURN
500 GOSUB 325
505 A=2:B=4:C=10:GOSUB 650
510 FIL=0:LOCATE 3,15:PRINT "Please wait ... "
520 OPEN FILES FOR INPUT AS #1
530 IF EOF(1) THEN 590
532 INPUT #1,SSO:ENDN=SSO
533 INPUT #1,SSO:XS1=SSO
534 INPUT #1,SSO:YS1=SSO
536 IF EOF(1) OR FIL=ENDN+1 THEN 590
540 INPUT #1,SSO
550 S%(FIL)=SSO
555 LOCATE 3,30:PRINT (ENDN-FIL)
560 FIL=FIL+1
570 GOTO 536
590 CLOSE #1
595 BEEP:PUT(XS1,YS1),S%,PSET
600 RETURN
650 FOR I=A TO B:LOCATE I,C:PRINT SPACES(25):NEXT I:RETURN
1000 A=2:B=4:C=10:GOSUB 650
1010 IF ERR=53 THEN LOCATE 3,15:PRINT "File not found      "
1020 IF ERR=61 THEN LOCATE 3,15:PRINT "Disk full          "
1030 IF ERR=64 THEN LOCATE 3,15:PRINT "Bad file name       "
1040 IF ERR=70 THEN LOCATE 3,15:PRINT "Disk write protect  "
1050 IF ERR=71 THEN LOCATE 3,15:PRINT "Disk not ready      "
1060 IF OS=CHR$(19) THEN RESUME 201
1080 IF GS="r" OR GS="R" THEN RESUME 500
1090 GOTO 150

```

## 大型汉字配上倒影的初步试验

九江炼油厂计算机站

夏国华

第一印象是十分重要的。包装、装璜、书刊封面设计者和服装设计师以及推销员、公关人员等对此更具有发言权。甚至婚姻大事的成功与否有时也与第一印象密切相关。可见，第一印象是一门学问，是一种艺术。

有些软件设计师，往往在其设计的系统正式运行前以醒目的大型字符（尤其是大型汉字）显示在屏幕上，其目的不仅在于宣传主题，而且是

为了雅观，力图给观众留下美好的第一印象。

先将欲放大的汉字显现在屏幕上，然后读取组成这些汉字的点，最后将这些点以园或矩形取代之，是将汉字显示放大的方法之一。笔者通过试验发现，若将这些放大后的汉字配上相应的倒影并在同一屏幕上显示，其宏观效果是甚佳的。富有立体感的倒影，令人回味，倍感真切。

下面笔者简介自己编的二个小程序，供有兴趣

趣者上机(机型: IBM PC机或PS/2机)证实、评赏、引用, 诱发有关者创新或介绍更好的方法。

程序(1), 须先在CCDOS下输入和运行, 目的是把汉字点阵数据存盘; 程序(2)中103句之前的作用是以矩形代替点使汉字放大, 103句之后是处理倒影。倒影分成三部分, 旨在增强立体感。

程序(1)仅需运行一次, 以后运行程序(2)时无论是中文还是西文状态均可以。运行程序(2)的结果是白底红字, 每个汉字约为2.2CM×5.2CM, 汉字位于屏幕的中上部, 其倒影位于中下部且向左倾斜。当然, 根据个人爱好, 亦可把汉字及其倒影设计成不倾斜或倾斜(本例是汉字不倾斜而其倒影倾斜), 只需对程

序(2)稍作修改便可实现。

程序清单如下:

#### 程序2

```
10 KEY OFF:CLS:SCREEN 1:COLOR 7,4:'取点 放大
30 DIM C(128,16)
46 OPEN "GRAPHIC.DAT" FOR INPUT AS #2
50 FOR X=1 TO 128:FOR Y=1 TO 16
60 INPUT #2,C(X,Y)
70 IF C(X,Y)=0 THEN 90
81 A=X*2+20:B=Y*4+30
82 LINE(A-2,B-5)-(A,B),2,BF
90 NEXT Y
100 NEXT X
103 CLOSE
230 FOR X=1 TO 128:FOR Y=16 TO 1 STEP-1
250 IF C(X,17-Y)=0 THEN 280
260 A=X*2+20+Y-1:B=Y*3+90
263 IF Y<7 THEN 276
265 IF Y<13 THEN 275
271 LINE(A-.4,B-4)-(A,B),2,BF:GOTO 280
275 LINE(A-1,B-2+9/(Y))-(A,B),2,BF:GOTO 280
276 LINE(A-1,B-3)-(A,B),2,BF
280 NEXT Y
290 NEXT X
291 LOCATE 8,23
292 END
```

#### 程序1

```
10 CLS:'读点存盘
20 KEY OFF:SCREEN 1,0:COLOR 9,0
30 DIM C(128,16)
40 LOCATE 1,1:PRINT "油罐检定计算系统":'欲放大的汉字
50 OPEN "GRAPHIC.DAT" FOR OUTPUT AS #1
60 FOR X=1 TO 128:FOR Y=1 TO 16
70 C(X,Y)=POINT(X,Y)
80 WRITE #1,C(X,Y)
90 NEXT Y,X
100 CLOSE:END
```



图1

# 最优箭线网络图的绘制算法

西北工业大学 周 明

**摘要:** 本文介绍了一个过程化的绘制最优箭线网络图的算法, 并给出了算例。该算法易于用计算机来实现。

## 1. 前 言

最优箭线网络图在其实际应用中深受人们欢迎。文献[1]介绍了其绘制法则, 从而为绘制最优箭线网络图提供了基本的途径和方法。但文献[1]中提供的方法, 既要考虑工序的紧前工序, 又要考虑工序的紧后工序, 这样势必增加对已知条件的多余要求, 特别是在用计算机实现该方法时, 要占用较多的内存。这里, 我们提出一个经过改进的过程化的方法, 它所考虑的已知条件仅仅是各工序与其紧前工序间的逻辑关系, 从而可以使最优箭线网络图的绘制过程易于用计算机来实现。

## 2. 逻辑关系到网络关系的转换算法

最优箭线图的绘制过程, 实际上是将工序间的逻辑关系转换为网络关系的过程。我们按照工序明细表一个工序一个工序地来处理。假定当前所处理的工序为 $x$ , 其紧前工序为 $a_1, a_2, \dots, a_e$ , 经过归纳分析, 共有下面的几种情况。

### 2.1 当前所处理的工序无紧前工序

则工序 $x$ 的始点为整个工程的起点。如图1所示, 可画出工序 $x$ 。

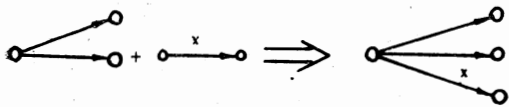


图 1

### 2.2 当前所处理的工序有一个紧前工序

它又可分为二种情况。

一种情况是该紧前工序的终节点的入度=1, 则将这个紧前工序的终节点作为工序 $x$ 的始节点, 画出工序 $x$ , 如图2所示。

另一种情况是该紧前工序的终节点的入度>1。如图3所示, 先将紧前工序 $a_1$ 从合节点中分

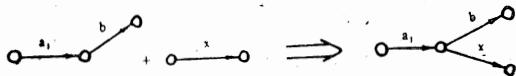


图 2

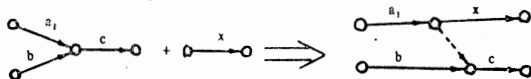


图 3

出来, 然后从 $a_1$ 的终节点引一虚工序到原合节点, 再以 $a_1$ 的终节点为工序 $x$ 的始节点, 画出工序 $x$ 。

### 2.3 当前所处理的工序有 $e$ 个紧前工序

它又可分为三种情况。

第一种情况是, 进入各紧前工序终节点的所有工序都是当前所处理工序的紧前工序, 且各紧前工序尚无出度。从这 $e$ 个紧前工序中任选一个作为代表工序, 并定义它的终节点为合节点。以该合节点作为工序 $x$ 的始节点, 画出工序 $x$ 。其余 $(e-1)$ 个紧前工序, 若将它们的终节点合并到合节点上而不会引起某两个工序相重叠的话, 则进行合并处理; 对哪些能够引起重叠的紧前工序, 则从其终节点引一虚工序到合节点。如图4所示。

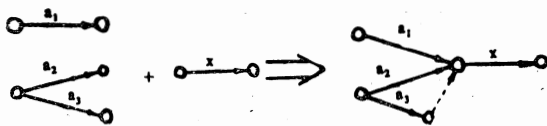


图 4

第二种情况是, 进入各紧前工序终节点的所有工序都是当前所处理工序的紧前工序, 但某些紧前工序有出度。先就无出度的紧前工序按第一

种情况处理,得一合节点,并以它为始节点画工序x;然后分别从有出度的节点引指向合节点的虚工序,如图5所示。假如所有的紧前工序终节点都有出度的话,如图6所示,则增设一节点为合节点,以它为始节点画工序x,并分别从有出度的节点引指向该合节点的虚工序。

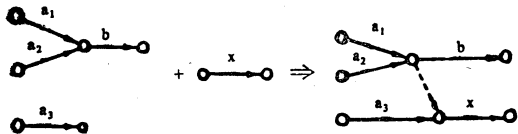


图5

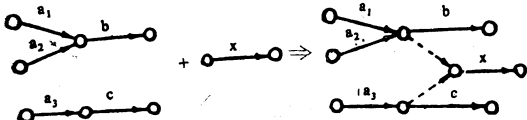


图6

第三种情况是,进入紧前工序终节点的某些工序不是当前所处理工序的紧前工序。先就所有

进入工序都是x的紧前工序的那些节点按第一种情况处理,得一合节点,并以它为始节点画出工序x;然后对其余包含有未处理的紧前工序的各节点,分别增加一节点,以将工序x的紧前工序和非紧前工序分开,并从该增加的节点引指向合节点和原节点的虚工序。如图7所示。

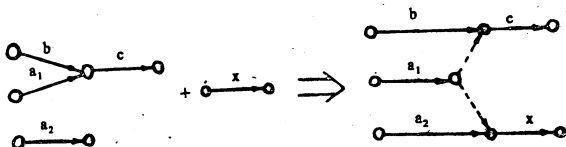


图7

#### 2.4 工程终点的处理

按上述方法处理完所有工序后,再仿2.3节中的第一种情况,将所有无出度的节点合并为工程的终点。

上述过程实际上是给出了最优箭线图的绘制算法,下面我们给一算例。

### 3. 算例

某工程任务的工序明细表如下:

工序代号	a	b	c	d	e	f	g	h	i	j	k	l
紧前工序	—	—	—	a, b	a, c	a, b, c	d	e	e, f, g	h, i	f	j

按上述算法绘制出的最优箭线网络图如图8所示。

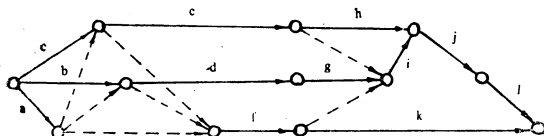


图8

这个算法很容易用计算机来实现,它在我们研制的图像仪辅助网络计划技术软件中得到了应用。

#### 参考文献

- [1] 黄沛钧、程国平,“最优箭线图的绘制法则”,《系统工程理论与实践》,1986年第1期
- [2] 李随成,“最优箭线网络图中虚工序的正确运用”,《系统工程理论与实践》,1988年第1期。

## 并行处理语言occam及其在图象处理中的应用

西南交通大学 诸昌铃

**摘要** 本文扼要地介绍了并行处理语言occam的一些基本概念,在此基础上给出了笔者完成的中值滤波和多模板匹配等有关图象处理程序的主要结构,用以说明并行处理程序及occam语言的应用。

并行处理是从体系结构入手提高运算速度的一种有效方法。除了在硬件上需要多个处理器

并行地工作且能按一定拓补结构互相通信之外,要实现并行处理还必须有一种能够处理并行机制

的语言。由英国INMOS公司与牛津大学程序设计研究室共同开发的occam语言,就是一种较理想的并行处理语言。occam语言的产生与并行处理器Transputer的研制是分不开的,它不仅进行并行处理所需的语言,同时也是描述并行硬件系统的语言。随着Transputer的发展和推广应用,occam语言也日益受人垂青。

### occam语言概要

#### 一、基本定义

和一般语言一样,occam对名字、数据类型、运算符等均有严格的定义,现简要介绍如下:

名字:在occam语言中,一个对象(变量、通道、常数等)都可以给予任意长的名字;一个名字必须以字母起头,后面可以跟以字母、数字和逗号组成的其余部份;大小写字母在occam中是区别对待的。

数据类型:occam语言中把数据分为整型数据INT、字节型数据BYTE和布尔型数据BOOL。其中BYTE型数据用来表示0到255之间的整型数,通常用以代表字符的ASCII码值。BOOL型数据只有两个可能的取值,即TRUE和FALSE。在对一个数据进行操作之前,应先对其类型加以说明,说明的格式为:

数据类型 变量名 {, 变量名}:

其中数据类型为保留字INT、BYTE和BOOL之一,冒号:作为说明语句的结束符。例如:

```
INT x, y:
```

说明变量x和y是整型数。

字符与串:字符的值为BYTE型数据,在书写时可用带单引号的字符来表达,如'a'。串作为字符数组来处理,书写时用带双引号的串来表示,如"hello"。

数组:在occam语言中,按如下形式对数组数据进行说明:

{元素数}{(元素数)}数型 名字:

如:

```
{8} BYTE char:
```

表明char是一个有8个BYTE型数据元素

的一维数组。而

```
{8}{8} INT plane:
```

表明plane是一个有8×8个INT型数据元素的一维数组。

数组在说明后,可以用名字加以引用并在名字后的方括号内写明下标,如

```
char {2}, plane {3} {2}
```

运算符:occam分别对算术运算、布尔运算与比较判别等规定了相应的运算符。

+, -, /, \*, REM是为算术运算符,其中REM是求除法余数的算符。上述各算术运算符具有相同的运算优先级,因此在同一表达式内存在多个运算符的情况下,必须用圆括号来指定运算的顺序。

PLUS、MINUS和TIMES作好模2运算的加、减、乘算符。

AND、OR和NOT是为逻辑算符。

判断测试所需的比较符有: =, >, <, < >, > =, < =等。

通道与通道类型说明:通道是指从一个处理到另一个处理的点到点的链接,用以进行处理间的通信,通道类型说明用于指定能够通过该通道传递的数据的类型,其说明格式为:

CHAN OF 数型 通道名 {, 通道名}:

如:

```
CHAN OF INT chan:
```

说明通过通道chan可以传递整型数。

省略:在occam中可以用一个名字来代替表达式的值,用以下形式进行省略:

```
VAL 数型 名字 IS 表达式:
```

如:

```
VAL INT exp IS ((x+y) / (p+g)):
```

也可通过省略定义常数,如:

```
VAL BYTE Esc IS 27:
```

#### 二、原处理

所谓一个‘处理’就是‘开始——执行某些操作——结束’,‘原处理’就是组成occam程序的最基本的处理。在occam语言中原处理只有赋值、输入、输出三种。

赋值处理:对一个表达式求值并将其值赋予



一个变量,其一般形式为:

变量名:=表达式

如  $x:=2, y=2+3$

输入处理:从一个通道输入一个值并将其赋予一个变量,输入的符号用问号?表示,其形式为:

通道名 ? 变量名

如  $\text{chan} ? x$  即从通道  $\text{chan}$  输入一个值给变量  $x$ .

输出处理:对一个表达式求值并从指定的通道输出出去,输出符号为惊叹号!其格式为:

通道名 ! 表达式

如  $\text{chan} ! 2$  即从通道  $\text{chan}$  输一个值2.

输入输出处理是解决并行处理系统中各并行部份相互通信的原处理,这两个处理必须是配对的,即在一个通道的一端存在一个输出处理而在另一端存在一个输入处理.只有当一个通道的输出与输入两侧均已准备好时,通信才能进行,换言之,一个输入处理只有当与之配对的输出处理准备好时才能进行,反之一个输出处理也只有在与之配对的输入处理准备好后才能进行.

### 三、结构

在程序结构方面除了像一般高级语言那样具有顺序、循环、条件、选择、重复、过程及函数等形式之外,occam作为并行处理语言更具有并行结构的特殊功能.下面分别对各种结构作一介绍:

SEQ结构:SEQ是Sequence的缩写,表示顺序结构,即依顺序执行在其控制之下的各个处理.一个SEQ结构只有在其构成的最后一个处理完成之后才结束.在程序的书写上被SEQ控制的各处理写在比SEQ缩进两个字符的起始处(即正对着字母Q),这样不但程序整齐易读,而且occam也容易辨认哪些处理是在该SEQ的控制之下.这种缩写形式在occam的其他结构中也普遍适用. SEQ结构举例如下:

SEQ

$\text{chan3} ? x$

$y := x + 1$

$\text{chan4} ! y$

PAR结构:PAR是PARAllel的缩写,表示并行结构,即所有在PAR控制之下的各个处理在同一时刻开始,当构成PAR的所有处理均一并完成时,该PAR结构才结束.在PAR控制下的各并行处理其书写的先后顺序其实是无关紧要的. PAR结构的举例如下:

PAR

SEQ

$\text{chan3} ? x$

$x := x + 1$

SEQ

$\text{chan4} ? y$

$y := y + 1$

即两个顺序结构(都是先输入后赋值)的并行执行.当然这个例子只将并行结构作了强调,并不完整.首先,它像前面的例子一样没有对变量( $x, y$ )和通道( $\text{chan3}, \text{chan4}$ )加以说明;另一方面在这个例子中不论对通道  $\text{chan3}$  还是  $\text{chan4}$  都只有输入处理,而未表示出相应的输出处理,所以单独执行这个PAR结构,必然会在输入处理  $\text{chan3} ? x$  和  $\text{chan4} ? y$  处死锁,因为没有向  $\text{chan3}$  或  $\text{chan4}$  提供数据的输出处理.从某种意义上来说PAR结构的关键问题,就是解决在其控制之下的各并行处理之间的通信.死锁问题不仅会出现在输入输出不配对的情况下,而且会在输入输出不能同时准备好的情况下发生,如:

PAR

SEQ

$\text{chan3} ? x$

$\text{chan4} ! 3$

SEQ

$\text{chan4} ? y$

$\text{chan3} ! 2$

尽管在两个并行执行的SEQ中对通道  $\text{chan3}$  和  $\text{chan4}$  都具有输入输出两种处理,但由于两个SEQ都把输入处理放在前面,故哪一个SEQ都不可能从相应的通道上得到数据,依然死锁.只有改变SEQ之一的内部顺序,就可避免死锁.但如果两个SEQ的内部顺序同时改变则仍旧死锁.

WHILE结构:在occam中循环是由WHILE 判别式组成,当判别式之值为真时,执行WHILE所控制的处理,执行完再循环到判别,直至判别式之值为假.例如:

```
x:=0
WHILE x>=0
  SEQ
    input ? x
    output ! x
```

这是一个简单的数据传送程序，它循环地执行从input输入一个数据，并把它从output输出，直至该数据之值为负。

IF结构：条件结构根据条件判别式的取值决定执行某种相应的处理，其一般形式为：

```
IF
  条件式1
    相应的处理1
  条件式2
    相应的处理2
```

如：

```
IF
  x=1
    chan1 ! y
  x=2
    chan2 ! y
```

应注意的是occam语言并未提供ELSE结构，也没有默认的ELSE功能，因此在使用IF结构时，必须考虑所有可能的条件以免死锁。在上例中如果x的取既非1又非2程序将出现死锁，为此应加上条件式

$(x < > 1) \text{ AND } (x < > 2)$  及相应的处理，如chan3 ! y等。

ALT结构：ALT是ALternative的缩写。在多输入处理的情况下往往需要按照一定的条件或输入通道准备好的情况来选择执行其中的一个输入处理。例如根据哪个通道先准备好就执行那个输入处理的ALT结构有以下形式：

```
ALT
  chan1 ? x
    进一步的处理
  chan2 ? x
    进一步的处理
  chan3 ? x
    进一步的处理
```

过程：过程的定义由关键字PROC开始，并有以下形式：

```
PROC 过程名 ({, 形参})
  过程体
```

最后的冒号是过程定义的结束符。过程的引用有如下形式：

过程名 ({, 实参})

函数：occam设计了函数结构，但在大多数目前的版本中尚未提供。

重复：为了使SEQ、PAR、ALT及IF等结构具有重复功能，occam提供了一种重复方式：

REP 重复变量=基 FOR 计数

其中REP表示SEQ、PAR、ALT、IE之一，重复次数从基开始到计数值所指定的数为止。如：

```
SEQ i=0 FOR 5
  input ? x
```

即顺序执行input ? x 5次

又如对一个先进先出FIFO缓冲器的结构，用occam来描述，就可采用重复的PAR结构。设该缓冲器有20个缓冲单元，每次从输入端输入一个数据的同时，这20个缓冲单元的内容依次递传，直至输出端输出一个最末端的一个数据。

```
→FFFFFFFFFFFFFFFFFFFFF→
输入          缓冲器          输出
```

可以看出包括输入端、输出端及各缓冲单元之间的链接，共需21个通道，我们可以定义一个数组通道，由普通PAR结构写成的程序为：

```
{21}CHAN OF INT chan: PAR
  SEQ
    chan[0] ? f0
    chan[1] ! f0
  SEQ
    chan[1] ? f1
    chan[2] ! f1
  .....
  SEQ
    chan[20] ? f20
    chan[21] ! f20
```

用重复的PAR写起来则简单得多：

```
{21}CHAN OF INT chan:
  PAR i=0 FOR 20
    SEQ
      chan[i] ? f
      chan[i+1] ! f
```

即该SEQ并行地重复执行20次。不难看出重复的PAR结构是描述流水线的一种有力工具。

### 图象处理应用举例

随着对图象质量要求的不断提高，一幅图象

的图案越来越多,而从处理速度方面来看也要求尽量加快,因此在图象处理中采用并行处理就成了一种应时之需。用occam语言实现的图象处理程序涉及到并行处理的很多方面,现仅举两例侧重介绍流水线结构与多机多任务结构的实现。

### 一、采用流水线结构实现的中值滤波器

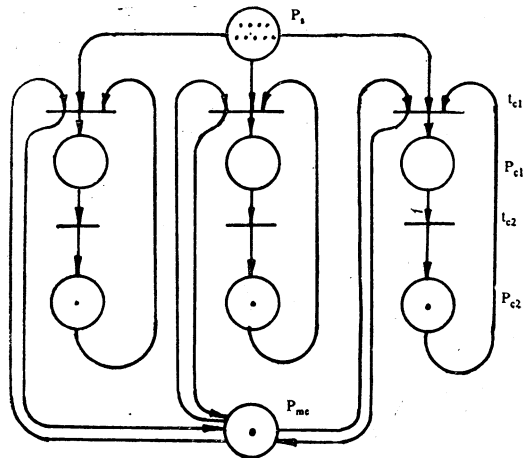
以 $3 \times 3$ 的二维中值滤波器为例,每次运算的核心是在9个相邻的图案上对其灰度级进行排序,并取出中间值作为所讨论的图案的灰度级。因此中值滤波就是一个排序过程。采用9个并行单元组成的流水线,在进行9次比较后即可得出结果。流水线结构示意图如下:

$\rightarrow p_0 \rightarrow p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_4 \rightarrow p_5 \rightarrow p_6 \rightarrow p_7 \rightarrow p_8 \rightarrow$   
 $ch_0 \ ch_1 \ ch_2 \ ch_3 \ ch_4 \ ch_5 \ ch_6 \ ch_7 \ ch_8 \ ch_9$   
 排序前的数据从通道 $ch_0$ 进入 $p_0$ ,原来 $p_0$ 中的数据与新输入的未排序数据进行比较,把大的(big)留下,小的(small)从 $ch_1$ 输出给 $p_1$ 。处理 $p_1$ 到 $p_8$ 的情况与 $p_0$ 类似。经过9次比较操作的时间,在 $p_0$ 中得到最大值、在 $p_1$ 中得到次大值,在 $p_8$ 中得到最小值,而 $p_4$ 中即是中值。上述过程是为了理解排序而设想的,对中值滤波来说只要求到中值即可,因此对中值以下的数据没有必要进一步排序,所以 $p_5$ 到 $p_8$ 是不必要的。此外对于每一个处理,比较的次数是递减的,即 $p_0$ 比较8次、 $p_1$ 为7次、 $p_2$ 为6次等等。这样相应的occam结构为:

PAR i=0 FOR 5	五个并行处理 ( $p_0 \sim p_4$ )
SEQ	
chan[i] ? bigger	首先输入第一个数据
SEQ j=0 FOR 8-i	进行8-i次比较
chan[i] ? smaller	输入下一个数据
IF	
smaller <= bigger	进行比较,把大数留下, 小数送到下一个处理去
chan[i+1] ! smaller	
smaller > bigger	
SEQ	
chan[i+1] ! bigger	
bigger:=smaller	
chan[i+1] ! bigger	最后输出本处理结果

### 二、多模板匹配

模板匹配是图象识别中的一种经典手段,经常需要对一幅图象进行多块模板的匹配。在并行处理的情况下,各个模板可以分配到不同的并行处理中同时进行匹配运算,但有时模板个数多于并行处理的数目,在此情况下就涉及到任务分配问题,而且由于各模板的具体情况不同因此各任务所需时间也不一致,这就要求任务在并行处理间进行动态分配。以具有4个并行处理,而需对9个模板进行匹配的情况为例,任务分配的过程可由图1来描述。图1是Petri网图形式,其中a



## 任务分配实现方法

PROC task.parallelism()

... variable and channel declarations

PROC tmpl.1()

... template1

:

PROC tmpl.2()

... template2

:

:

. -- proc tmpl.3() -- proc tmpl.8()

:

PROC tmpl.9()

... template9

:

SEQ

PAR

SEQ

-- control unit

workload.number:=0

token.1 ! workload.number

workload.number:=workload.number+1

token.2 ! workload.number

workload.number:=workload.number+1

token.3 ! workload.number

workload.number:=workload.number+1

WHILE workload.number &lt; 9

ALT

reply.1 ? do

SEQ

token.1 ! workload.number

workload.number:=workload.number+1

reply.2 ? do

SEQ

token.2 ! workload.number

workload.number:=workload.number+1

reply.3 ? do

SEQ

token.3 ! workload.number

workload.number:=workload.number+1

terminator:=20

reply.1 ? done

token.1 ! terminator

reply.2 ? done

token.2 ! terminator

reply.3 ? done

token.3 ! terminator

SEQ

-- parallel image processing unit 1

process.1:=TRUE

WHILE process.1=TRUE

SEQ

token1 ? command.1

IF

command.1=20

process.1:=FALSE

command.1 &lt;&gt; 20

SEQ

IF

command.1=1

tmpl.1()

command.1=2

tmpl.2()

:

. --if command.1=n, then tmpl.n()

:

command.1=9

tmpl.9()

reply.1 ! answer.1

SEQ

... parallel image processing unit 2

SEQ

... parallel image processing unit 3

... other processing

9个模板作为9个过程供并行处理调用

一个控制处理和三个图象处理并行, 以任务工作负荷号代表相应模板

token 通道作为控制处理  
向图象处理分配任务的通道  
先向各图象处理分配一个任务

已分配任务数小于9

以 reply 通道作为图象处理 向控制处理反馈空闲状态的通道

当某一图象处理已完成前一任务处于空闲状态, 则分配下一任务

以任务号 20 代表结束标志

从 token 通道接受任务

如果任务号是 20, 结束处理

否则

按任务号调用相应的模板匹配过程

完成一次任务后发回空闲状态

并行处理图象之二

并行处理图象之三

# Turbo BASIC语言与汇编程序的连接方法

四川省地震局计算中心 辛 华 李谊瑞

**摘要:** 本文提出了用Turbo BASIC语言调用汇编语言程序的一种方法,较详细地阐述了实现的技术思路,并给出了一个程序实例。

## 一、问题的提出

IBM—PC系列微型机及其兼容机,在我国各行各业应用十分广泛,在使用过程中,很多用户都特别喜欢采用IBM—PC BASIC语言编写各种应用程序。但在实际应用中,如通讯网络系统、微机实时控制中,由于运算速度、内存分配、接口控制等多种因素,需要高级语言程序和汇编语言程序混合使用才能满足要求。这是因为在这些系统中不仅需要具有实时控制、采样检测等功能;同时还需具有较先进的过程监控、数据处理等功能。而高级语言和汇编语言在处理这些要求时各具有不同的特点,所以在很多应用领域中,必须结合两种语言的特点,发挥各自的特长。采用BASIC语言调用Intel 8086汇编语言程序,在一些资料上都有过介绍。

我们在研制“计算机远程数据通信网”软件系统时,主体程序采用了Turbo BASIC语言。这是因为该语言具有快速的编译编辑和交互性及跟踪调试系统,编译后的目标程序执行速度比解释BASIC语言快4—100倍;还采用了先进的信息流控制结构,并允许递归运算;同时还具有普通BASIC语言的所有特色。其中CRC16校验等程序直接采用Intel 8086汇编语言编写,这是因为它比Turbo BASIC语言方便灵活、速度快。这两种语言进行交叉程序设计,满足了本软件系统的需要。下面简要介绍Turbo BASIC语言调用Intel 8086汇编程序的一种方法。

## 二、两种语言相连接的方法

在Turbo BASIC语言中,汇编语言程序可以直接作为一个过程被主程序调用(使用Inline过程)。其调用方法如下:

```

CALL 过程名
:
END
SUB 过程名 Inline
$Inline * .COM
END SUB

```

这对一些要求运算速度或需完成某些特定功能(如直接调用BIOS、DOS功能以及对系统直接进行管理)的程序设计者来说,无疑提供了一个良好的开发环境。但是,Turbo BASIC语言对此又加了一定的限制,即汇编语言程序必须是\*.COM文件。由于\*.COM文件中只能有一个代码段,这一限制给需在程序中使用数据段(常数、表格等)的使用者带来了不便。

如果我们的程序中必须要使用常数、表格时,可以将它们放在代码段中作为代码段的一部份。源程序经过汇编、链接、转换形成\*.COM文件后,“数据段”的偏移地址已成为了一个绝对偏移值。这样的\*.COM文件在DOS下执行是没有问题的。因为在DOS的加载过程中,\*.COM文件被调入内存时,总是从一个新的物理段开始存放,即从一个偏移值为0的新的物理段开始存放(偏移值0000H—00FFH为程序段前缀,偏移值0100H处为程序开始执行地址),所以这时“数据段”的绝对偏移值就正

具体的模板匹配可以采用过程调用的方式来实现,此处不再赘述。任务分配的过程是通过

assignment与各parallel IP间的通信来完成的,具体实现方法如下:(见31页)



确地指向了“数据段”本身。但在 Turbo BASIC语言中,情形却就不同了。由于 $\bullet\bullet$ COM文件是作为一个过程被嵌入在 Turbo BASIC程序中,经过编译形成的 $\bullet\bullet$ EXE文件的代码段中,即包含了 Turbo BASIC本身的程序执行代码,也包含了 $\bullet\bullet$ COM文件的执行代码,换句话说, $\bullet\bullet$ COM文件中的代码段不再处于程序开始执行处。虽然“数据段”也被装入了内存,可这时却无法通过其绝对偏移值找到它的位置。如何解决“数据段”偏移值不正确的问题,可通过如下两种途径:

1. 既然“数据段”已作为代码段的一部份被装入到内存,仅是偏移值有错。可对编译后的 $\bullet\bullet$ EXE文件用Debug或其它工具软件对“数据段”的偏移值加以改写。这样设计出的程序可移植性非常差,只要源程序——Turbo BASIC稍有改动或 $\bullet\bullet$ CON的嵌入位置稍有变动,就只得相应改写一次“数据段”的偏移值。所以这种方法虽然可行但不实用。

2. 前面我们已知道了“数据段”在 $\bullet\bullet$ COM文件中的绝对偏移地址。根据 $\bullet\bullet$ COM文件的结构, $\bullet\bullet$ COM文件被加载进内存后,从偏移值为0000H到偏移值为00FFH为程序段前缀,偏移值为0100H处为程序开始执行地址。那么“数据段”到程序开始执行处的偏移值就可以通过其绝对偏移值减去0100H求出来(我们称作相对偏移值),并且这一相对偏移值的大小是固定不变的。因此,我们只要在 $\bullet\bullet$ COM文件中加上一段求取相对偏移值的程序,便可实现“数据段”的浮动。

### 三、程序实例

综上所述,我们给出一个 Turbo BASIC语言调用 Intel 8086 汇编语言程序的实例。

Turbo BASIC语言程序:

```
...
CALL TT (参数, ...) TT: 过程名
...
END
SUB TT LnLine
$inline T.COM T.COM: * .COM 文件
END SUB
```

Intel 8086 汇编语言程序 (T.ASM):

```
CODE SEGMENT
ORG 0100H
ASSUME CS: CODE, DS: CODE
PROC FAR
START: PUSH BP
      MOV BP, SP ; 保存BP,
      ; 以便正确返回
      CALL B
      JMP L
B: PROC ; 求“数据
   ; 段”相对偏移子程序
   PUSH BP
   MOV BP, SP
   MOV AX, [BP+2]; 取指令
   JMPL的偏移值
   SUB AX, 06H ; 求STA-
   ; RT: 的偏移值
   MOV DI, AX
   MOV AX, OFFSET DATA
   SUB AX, 0100H
   ; “数据段”DATA 在汇编程序中的
   ; 偏移值减去0100H, 得到 DATA 相
   ; 对START的偏移值。加上DI, 得到
   ; DATA 在实际 Turbo BASIC 中
   ; 的偏移值。存入DI以备后面使用。
   ADD AX, DI
   POP BP
   RET
B: DATA: DB.....
      .....
L: PUSH DS ; 保存DS, 以便正
   ; 确返回 Turbo
   ; BASIC.
   ;
   ; POP DS
   ; POP BP
   ; ENDP
   ; ENDS
   ; END START
```

### 参考文献

- [1] 荣海、王复车, “BASIC语言调用汇编程序的一种方法”, 《计算机世界月刊》, 1988年4期。
- [2] 叶红、魏鹏译, “Turbo BASIC语言使用手册”, 北京科海培训中心, 1988年3月。

## 快速高效编制报表的简易方法

西安空军工程学院计算机教研室 方荣耀

随着计算机广泛使用, 采用 dBASE III PLUS 的用户越来越多, 该软件以众多的命令和功能丰富的函数支持各种应用软件。

计算机在企业管理中, 编制报表工作是应用软件中的重要一环。对于复杂表格, 即使是高级电子制表软件也不能胜任。尤其是在纵、横方向

上均为多级表头的表格,只能按该报表设计一个打印程序来实现。编辑这个打印程序时,其过程较为复杂和繁琐,开发周期长,工作量很大。况且随着改革的深入发展,报表的模式不断改变,维护这些打印表格的程序也随之带来很大困难。

打印复杂表格的困难就在于表格线的处理。无论是编辑或修改打印表格程序,在处理表头各栏的表格线时,总是以输入相应的区位码实现的。表格横线较长时,输入相应的区位码就容易出错,该横线极易造成时长时短。当进行增删时,稍不小心,又易出现半个汉字而发生错位,碰到这种现象,实在令人头疼。要调整正确,只能耐心地依靠移动光标数数来解决。既麻烦又费时,实在太不方便了,远不能满足实际应用的需要。

随着dBASE III PLUS的深入使用,笔者在开发“工资管理系统”和“财务管理系统”应用软件时,遇到了大量的表格处理工作要做,快速实用的制表方法亟待解决,经仔细地分析研究和大量的制表实践,终于找到一个快速、高效、简易、可行编制复杂表格的方法,克服了上述种种困难。具体做法如下:

### 一、设计表格元素

任何表格无论多么复杂,即使是多层表头的表格,要用到的制表符归纳起来共有十一种:“┌、┐、└、┘、—、+、├、┤、┴、┬、─”,称之为表格元素,把它们分别赋给内存变量。如ZSJ=“┌”,ZX=“—”……。也可把它们放在内存文件中保存起来。如:

HSDUS、MEM

```
ZSJ      pub  C  "┌"
YSJ      pub  C  "┐"
ZKJ      pub  C  "└"
YKJ      pub  C  "┘"
ZX       pub  C  "—"
ZGX      pub  C  "+"
SGX      pub  C  "├"
HGX      pub  C  "┤"
BX       pub  C  "┴"
ZBK      pub  C  "┬"
YBK      pub  C  "─"
```

12 variables defined.

53 bytes used

244 variables available. 5947 bytes available

## 二、使用字符重复显示函数

### REPLICATE

字符重复显示函数的格式为:

REPLICATE (〈字符型表达式〉, 〈数值型表达式〉)。该函数可使参数〈字符型表达式〉的字符串重复多次,具体重复的次数由参数〈数值型表达式〉的整型值来决定。灵活地使用这个函数,可以非常方便的得到复杂表格所需的各种表格线。

如:ZX="—" 〈CR〉

? REPLICATE (zx, 5) 〈CR〉

运行结果为: "-----"。

在实际使用中进一步发现,当表格中有连续某几栏宽度相同时,只需将其中的一栏格线赋给变量,然后利用该函数就可得到连续几栏相同宽度的格线。

如:ZH1="┌——" 〈CR〉

? REPLICATE (ZH1, 4) 〈CR〉

运行结果为: "┌——┌——┌——┌——"。

由以上二例可见,灵活地使用此函数,可以迅速准确地得到表格中所需的任意表格线。

随着改革的深入发展,办公室自动化程度不断提高,各种报表的格式也在不断变更。使用上述编制报表的方法,维修打印表格程序中的格线就易如反掌。如果需要增加栏目,只需增加描述该栏目的一个变量。若只改变某栏宽度则更易,只需改变重复显示函数的数值型参数即可,非常省事。

### 三、一个实例

这里介绍一份实际应用的“核(准)决算通知单”的表格形式。如表一所示。

它既适用于各种会计凭证、各类发票,也适用于各类复杂报表。采用上述编制报表的方法,编制本报表打印程序,显得非常简单。根据给定表格设定好各栏的宽度,再将表中给定的内容组织好循环,用输出语句?输出表头、表体和收尾部分,实用程序的清单附后。如图1所示。

由这个实例可知,采用这种方法实现表格输出,有以下明显优点:



```

DO WHILE GG>0
STOR ' ' TO W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11
MH=SUBS(11,JJ,2)
YYO=SUBS(KK,YL,20)
YYI=SUBS(KK1,YL,20)
YL=YL-20
JJ=JJ-2
DD=LEN(STR(AMH))
SS=DD
I=0
DO WHILE DD>0
I=STR(DD,1)
WMI=SUBS(STR(AMH),SS-DD+1,1)
DD=DD-1
ENDD
? bx+spac(2)+bx+yy0+bx+yy1+bx+z1+u11+bx+z1+u10+bx+z1+u9+bx+
z1+u0+bx+z1+u7+bx+z1+u6+bx+z1+u5+bx+z1+u4+bx+
z1+u3+bx+z1+u2+bx+z1+u1+bx+spac(2)+
bx+spac(2)+bx+spac(2)+bx
? s1
ss=ss-1
endd
? BX+REPL(S10,2)+SPAC(16)+BX+SPAC(20)+BX+REPL(S10,14)
? zbx+ZX+XGX+ZX+XGX+s4+s4+XGX+s5+s5+XGX+repl(s9,13)+zx+ybx
? bx+'附'+BX+SPAC(90)+BX
? BX+SPAC(2)+BX+SPAC(90)+BX
? BX+'注'+BX+SPAC(90)+BX
=打印表卷收尾部分
? ZXJ+ZX+XGX+REPL(ZX,49)+TXJ
? SPAC(6)+'财务处长:-----'+SPAC(50)+'助理员:-----'
?
?
?
?
?
SKIP
ENDD
CLOSE DATA
SET CONSOLE ON
SET PRINT OFF
RETURN

```

图1

1. 只要使用者会编程,就能用上述方法编制复杂表格的打印程序,这种打印表格的程序设计非常简单、明瞭、直观,易读、易懂、易修改。

2. 编辑打印表格程序时,省时、省力、省事,准确可靠质量高。它省去了以输入区位码直接处理表格线的方法,而把制表符存于内存变量中,均以汉字“全形”处理,决不会有半个汉字字节出现。因而不会出现表格线缩短、增长、断裂、错位等故障。再也不用移动光标数数那种耗费大量时间与精力的笨拙方法了。从而大大地缩短了程序开发和维护的周期。

3. 越是复杂的表格,用这种方法设计越显得简单快速高效,能提高工效二至四倍,使得各种应用软件能得心应手地满足用户的要求。

### 参考文献

1. 陈璇等:《微机网络数据库设计使用指南汉字dBASE III PLUS》,科海培训中心 1987.
2. 张福炎等:《IBMPC的原理与应用》,南京大学出版社 1986.
3. 杨润生主编:《办公自动化教程》,湖南大学出版社 1986.

## MIGHTYPRESS 打印接口卡的特殊用法

广西师范大学 唐汉雄

Apple II 型计算机及与其兼容的国产微机是目前我国较普遍使用的一种机型。MIGHTYPRESS 打印接口卡是适用于这类微机的并行打印接口,它广泛适用于 EPSON, NEC, BMC 等多种类型的打印机,可在 Apple 的各种操作系统如 DOS3.3, PASCAL, CP/M 下使用。它比常用的 EPSON 打印接口卡有更强的图形打印功能,它可以正常、反相、放大、右旋 90°、倍密度及高分辨第一、二页并列打印输出,也可对屏幕实现硬拷贝。现将它的主要用法简单介绍如下。

### 一、接口卡的正确安装

在 MIGHTYPRESS 接口卡上方设有一个八位 DIP 小开关,编号为 #1~#8 (实际只用 #1~#4)。接口卡与不同型号的打印机连接时,各个开关的状态应按下表设置。

常用的 Fx-80, Fx-100, Sakata, SP-100, CP-80, Ep-1000 等均属于 EPSON 系列打印机,可按上表位置设定。接口卡可以插在 0~7 任意槽号上,但习惯上将卡插在 #1 槽上。安装接口卡时一定要关闭主机电源,以免造成不必要的损坏。

打印机类型	DIP小开关			
	# 1	# 2	# 3	# 4
EPSON系列	ON	ON	ON	ON
NEC8023/C.Iton Pro	OFF	ON	ON	ON
Anadex	ON	OFF	ON	ON
IDS	OFF	OFF	ON	ON
Okidata 84	ON	ON	OFF	ON
Other Oki.	OFF	ON	OFF	ON
APPLE D.M.	ON	OFF	OFF	ON
Mann, Talley	OFF	OFF	OFF	ON
BMC Bx-80/Spirit80	ON	ON	ON	OFF

在Applesoft状态下MIGHTYPRESS有自检功能, 自检时, 先用PR#n (n为槽号) 打开打印机 (本文提到的打印命令均需在打开打印机后才起作用), 然后键入CTRL-IV 命令 (即按住CTRL 键的同时击 I 键, 全松手后再按 V 键), 打印机就会先打印全部字符集, 再后打印出“TEST OK”字样, 这表明接口卡的工作状态正常。

## 二、打印命令的特殊用法

### 1. 文本格式命令

①CTRL-InN 在LIST BASIC程序时, 打印机自动设定每行字符数 (行宽) 为 40, 需要改变行宽时, 可用CTRL-InN命令 (n=1~255)。执行此命令时, 字符只传送到打印机而不在屏幕上显示。如设n=80, 在程序行中, 此命令可写成: PRINT CHR\$(9); “80N”

②CTRL-InL 本命令使随后的打印每行左边留几个空格 (n=0~255, 打印机加电后 n 值为零)。如设n=10, 在程序中可写成: PRINT CHR\$(9); “10L”

③CTRL-IL 本命令是设置打印每行字符左边不留空格。用了上面 CTRL-InL后可用此命令取消。在程序行写成: PRINT CHR\$(9); “L”

④CTRL-InP 本命令是设置每页的行数 n (页宽n=1~127, 加电初始化后 n 值为零)。打印机在打完n行后留出6行空行。一般打印机每页的行数为66, 如设n=60, 则在程序行写成: PRINT CHR\$(9); “60p”

⑤CTRL-ID 本命令是恢复文本打印的初始化设定值 (即设置行宽 n=0, 页宽 n=0, 左边留空n=0, 恢复屏幕显示等)。在程序中写为: PRINT CHR\$(9); “D”

上述命令均可以立即运行方式执行。例如要求打印出左端预留 5 个空格、每行打印75字符、每页打印60行字符的BASIC程序清单, 可先输入CTRL-I5L CTRL-I75N CTRL-I60P CTRL-X (这里各个命令之间不允许有空格, 命令结束用 CTRL-X 代替 RETURN, 目的是防止打印机价出“SYNTAX ERROR”的信息), 然后打入LIST命令即可。

### 2. 屏幕开关命令

①CTRL-II或CTRL-IO 本命令使字符同时传送到屏幕和打印机, 用了上面的CTRL-InN 命令后若希望屏幕同时显示, 可补充此命令, 但若已设n>40会使CTRL-InN失效, 只取n=40打印。在程序中可写成: PRINT CHR\$(9); “I” 或 “O”

②CTRL-InI 本命令是设置屏幕显示并置行宽为n (n=1~255, 但未用80列显示板时屏上最多每行显示40字符)。如设n=30则写成: PRINT CHR\$(9); “30I”

③CTRL-IN 本命令是关闭屏幕显示并设行宽为零。在程序中写为: PRINT CHR\$(9); “N”

④CTRL-IS 本命令是将当前的40字符文字幕硬拷贝到打印机输出。若使用Videx 80列卡则可用CTRL-IW对80列屏幕进行硬拷贝。

⑤CTRL-I<sub>n</sub>S 本命令是将当前40字符文字幕的第几行开始的内容硬拷贝到打印机输出。对使用Videx 80列卡则用 CTRL-IW 命令 (n=1~23)。④⑤在程序中的写法与上相仿。

### 3. 改变命令字符

①CTRL-I CTRL-n 本命令是把命令字符CTRL-I改为CTRL-n。使用本命令后, 凡是命令中的CTRL-I均可用CTRL-n代替。如设CTRL-n为CTRL-Y则在程序中写成: PRINT CHR\$(9); CHR\$(25)

这里的CTRL-n原则上可选任意控制字符,但要避免使用CTRL-M (RETURN), CTRL-X, CTRL-H及用于打印机软件上的控制字符。

②CTRL-n CTRL-I 本命令是把命令字符CTRL-n改回CTRL-I。

上述命令也可以立即运行方式执行。

#### 4. 图形打印命令

①CTRL-IG 本命令是硬拷贝高分辨第一页的图象到打印机输出,图形打印在纸的中央部分。在程序中可写成:PRINT CHR\$(9); "G"

②CTRL-IG2 本命令是在纸中央打印高分辨第二页图形。程序中写成:PRINT CHR\$(9); "G2"

③CTRL-IGS 本命令是并排打印高分辨图形第一页和第二页。有的打印机位置不够,第二页图形打印不完整。此时可加入"L"(纸左端不留空)或"R"(右旋90°的命令打印出并排的图形。在程序中可写为:PRINT CHR\$(9); "GS"

④CTRL-IGD 放大一倍打印图形,本命令与两页并排打印命令"S"连用时失效。使用本命令如打印位置不够,可采用旋转90°并放大的方法打印。程序可写为:PRINT CHR\$(9); "GD"

⑤CTRL-IGR 把图形顺时针方向旋转90°。在程序中写成:PRINT CHR\$(9); "GR"

⑥CTRL-IGI 使图形以及反相方式打印。在程序中写成:PRINT CHR\$(9); "GI"

⑦CTRL-IGE 加重(倍密度)打印图象。(适用于EPSON, NEC8023A, C.Itoh Pro和APPLE D.M.等打印机)。使用本命令时,打印机对每个普通点打印成紧挨着的两个点。打印所需的时间是普通打印的两倍。在程序中写为:PRINT CHR\$(9); "GE"

⑧CTRL-IGL将高分辨图形第一页打印在纸的最左端。程序为:PRINT CHR\$(9); "GL"

⑨CTRL-IGM将文本与图形混显的高分辨率图形硬拷贝在打印机输出。程式为:PRINT CHR\$(9); "GM"

上述图形打印命令很多可以联合使用,例如PRINT CHR\$(9); "GLDIRE2",它是将高分辨第二页图象在纸的最左端以放大、反相、右旋90°和倍密度方式打印出来。本命令也可以立即运行方式执行,方法是先按住CTRL键的同时击I键,然后全松开再单独按GLDIRE2各键,最后用CTRL-X, RETURN, SPACE BAR之一作为结束键即可。虽用三者均能打印图形,但执行结果有所不同:按回车会在屏幕和打印机上输出"SYNTAX ERROR"字样。这是因为此命令既不是DOS命令也不是BASIC命令的缘故。按空格键作为结束键则可使打印卡保持在图形打印状态,当要再次打印图形时直接键入子命令(即I、E、R、S、D、I、M、L字母)即可,而不必再次使用CTRL-IG命令。

MIGHTYPRESS打印接口卡的其它用法因不太常用且限于篇幅,此处就不一一详述了。

### 三、应用举例

#### 例一、见程序1

LIST

```
10 TEXT: HOME
20 PR# 1
30 PRINT CHR$(9); "30N"; CHR$(9); "I"
40 FOR I = 161 TO 254: PRINT CHR$(I);: NEXT I
50 PRINT CHR$(9); CHR$(25); REM
  CHANGES THE CTRL-I TO CTRL-Y
60 PRINT CHR$(25); "10L"; CHR$(25); "50R"
70 LIST 50
80 PRINT CHR$(25); "D"
90 PRINT "MIGHTYPRESS INTERFACE"
100 PRINT CHR$(9); "S"
110 PR# 0: END
```

程序1

说明:第30行设定每行打印30字符并使打印内容同时送屏幕和打印机。50行把命令字符CTRL-I改为CTRL-Y。60行设置打印每行左端留10个空格,右边第50字符起留空。80行恢复屏幕显示和文本打印的初始化设定值,命令字符也恢复为CTRL-I。100行是对屏幕进行硬拷贝。



## 例二、见程序 2

说明：第60行打印高分辨第一页图形。100行打印高分辨第二页。110行打印高分辨第二页、放大一倍、顺时针右旋90°的图形。限于篇幅，图略去。

## 会议征文

四川省计算机学会情报网成立大会将于今年5月16~18日在成都地区隆重召开，现向本网各情报研究单位公开征文，内容为计算机及电子产品国内外情报及情报研究，论文可直接寄至本刊编辑部。李泽民收；与会者，请将论文打印60份带来。

JLIST

```

10 TEXT : HOME : VTAB 21
20 INPUT "FILENAME PIC.1?";F1$
30 PRINT CHR$(4);"BLOAD";F1$;"
  ,AS2000"
40 POKE - 16304,0: POKE - 1629
  7,0: POKE - 16300,0: POKE -
  16301,0
50 PR# 1
60 PRINT CHR$(9);"G"
70 INPUT "FILENAME PIC.2?";F2$
80 PRINT CHR$(4);"BLOAD";F2$;"
  ,AS4000"
90 POKE - 16299,0: POKE - 1630
  2,0
100 PRINT CHR$(9);"G2"
110 PRINT CHR$(9);"GDR2"
120 PR# 0: END

```

程序 2

## 多功能电脑编辑打印软件竭诚为各界计算机用户服务

功能简介：此软件(TX Ver4.0)曾荣获1989年度全国计算机应用研究学术交流会“优秀论文二等奖”，具有丰富的编辑和打印功能。它是运用价值工程原理，为弥补目前流行的字处理软件 and 高级排版软件之不足，提高各行业办公效率和质量而开发的，最突出的优势是：新颖实用，独具特色，投资低廉，功能丰富，操作简便，效果理想。其主要特点：(1)占用资源少。TX占用内存和磁盘空间少，可在硬盘或软盘上实现排版和打印功能。(2)适应范围广。可用于各行业进行(中文、英文、中英文混合的)公文、书信、文学作品、社会科学类书籍以及源程序清单的排版打印输出。(3)排版功能强。可根据用户确定的页宽自动排版。(4)打印方式多。具有“单/双/三/四栏+单/双面/+标准文本/清样文本”等16种打印方式；行、页、栏间距可调；可连续打印多份内容完全相同和主送单位(或个人)不同而正文完全相同的文本；一行内可打印四种字体(宋、仿宋、黑、楷体)及多种字号；可打印输出实线表格。(5)扩充性能优。TX既是一个独立的软件，又可作为用户应用系统的一个模块，由其它高级语言调用。

(6)工作效率高。以排版效果为基准看，其效率比c-Wordstar至少快30倍以上。(7)经

济效果好。TX可以大大地提高用户处理文书的效率和质量。同时，由于提供了双面打印功能，打印用纸量可比过去节约50%。有了TX，即可使“微型计算机=微型计算机+中英文电脑打字机”，从而节省了大量投资。总之，TX软件智能度高，使用简便，操作灵活，与c-Wordstar配合使用，便可在一台微型计算机上完成各种编辑、排版和印刷工作，是实现办公自动化的得力工具。

硬件环境：IBM-PC/XT/AT及其兼容机，任意型号的24针针式打印机。TX Ver4.0A适用于单/彩色/、高/中分辨率显示器；TX Ver4.0B适用于彩色、中分辨率显示器。

软件环境：任何版本的汉字操作系统及打印系统。

转让形式：5英寸软盘1片，使用手册随盘。

(购买时请注明所需版本号)

服务准则：信誉第一，用户至上；一次购买，终身保用；版本更新，免费复制。

转让价格：本刊订户(持当地邮局证明)购买，每片100元；其他用户购买，每片240元。

收款单位：本刊编辑部。开户行：成都跳伞塔分理处；帐号：89501299

联系人：唐大利

## 居民身份证底卡的微机打印程序

福建宁德师专计算机室 苏亚华

我国实行居民身份证是一件意义重大的策略。根据有关部门的决定,从今年十月份开始要在全国实行居民身份证查验制度。但是,目前许多地方印制身份证的速度很慢,严重影响了这项工作的进度。特别是填写身份证底卡,是一项十分繁琐的工作。现在多数地区还是采用手工填卡,不仅字体不规范,而且效率极低。这里介绍一种用微机处理居民身份证底卡的方法。用此方法我们已经打印了几万张底卡。实践证明,不但质量好,且方法简单,工效高。只要有一台IBM-PC或其兼容机及24针打印机,每人每天可以输入并打印底卡200张以上。

该方法在CCDOS 2.10和C-dBASE III环境下实现。

### 一、存储数据

微机处理身份证底卡,首先要将底卡上的记录输入计算机,存储起来。由于成批填卡,有大量数据是重复的。如:住址前半部分,签发日期,编号前六位等。为避免重复输入,我们把这些数据作为公用数据放在一个附库FK.DBF中,而其它一些数据放在身份证库SFZK.DBF中。打印时先到附库中取出公用数据,再从身份证库中逐个取记录打印。这样,大大减少了输入数据量,提高了输入速度。FK.DBF只有一个记录,如果公用数据有变动时,只要在打印前修改一下FK.DBF的内容即可。

除上述数据外,还有些数据可以由其它数据生成。如出生日期由编号第7至第12位得到,又如编号最后一位为奇数时性别为男,为偶数则为女。有效期限可以根据出生年月计算出来。经过分析,我们发现,一个身份证记录中,真正独立的数据只有姓名,民族,住址后半部和编号后9位这四个数据是独立的。因此,我们建立一个数据库SFZ.DBF,只包括上述四个字段。这样,不仅占用空间少,处理速度快,而且大大提高了

输入工效。而且,其中民族与住址部分往往有许多是相同的,可以先不输入,最后用REPLACE命令统一修改。这样,每个记录数据的输入量减至最低,一般只要输入一个姓名和9位数字即可。充分发挥了计算机处理的优越性。

附库FK.DBF和身份证库SFZ.DBF的库结构如下所示。

```
数据库结构——数据库      : C : FK.dbf
数据库中的数据记录个数      : 1
数据库的最后更新日期        : 06/16/89
  字段   字段名   类型   宽度   小数
  1      住址     字符型  18
  2      签发日期  字符型  12
  3      编号     字符型   6
**总计**                      37
```

```
数据库结构——数据库      : C : SFZ.dbf
数据库中的数据记录个数      : 4
数据库的最后更新日期        : 06/16/89
  字段   字段名   类型   宽度   小数
  1      姓名     字符型   8
  2      民族     字符型   2
  3      住址     字符型  18
  4      编号     字符型   9
**总计**                      38
```

### 二、打印底卡

进入dBASE III状态,运行打印程序,默认盘上应有附库FK.DBF和身份证库SFZ.DBF。打印时,先从附库中取出公用数据赋给变量ZZ, QF与BH,然后依次取身份证库记录打印。先取编号后9位数码,分解出出生年,月,日;根据最后一位数的奇偶确定性别;并计算有效期限。然后取其它几项数据。处理完一张底卡的四个记录后,一次打印一张底卡。程序十分简单,大家可以自己从中分析具体的数据处理方法。用这个程序打印一张底卡约需35秒,一台打印机一小时可打卡100张。同时,此程序可以毫不修改地用编译dBASE III编译成.EXE文件运行,速度更快。

```

*打印身份证底卡
set talk off
clear
N=0
@ 1,15 say '注意: 打印的数据库为 SFZ.DBF'
@ 2,15 say '公用数据: 编号, 住址, 签发日期'
      应在副库 FK.DBF
@ 3,15 say '请输入起始序号 (0退出): '
      get N pict '9999'

read
if N=0
    return
endif
use FK
BH=编号
ZZ=住址
QF=签发日期
use SFZ
count to M
go N
do while recno()+3<=M
    K='1'
    do while K<='4'
        XM&K=姓名
        MZ&K=民族
        ZZ&K=住址
        BH&K=编号
        X=val(subs(BH&K,9))
        if X=2*int(X/2)
            XB&K='女'
        else
            XB&K='男'
        endif
        X=subs(BH&K,1,2)
        if X='0'.or.X='9'
            YEAR='18'+X
        else
            YEAR='19'+X
        endif
        MONTH=subs(BH&K,3,2)
        if MONTH='0'
            MONTH=''+substr(MONTH,2,1)
        endif
        DAY=subs(BH&K,5,2)
        if DAY='0'
            DAY=''+subs(DAY,2,1)
        endif
        CS&K=YEAR+' '+MONTH+' '+DAY
        do case
            case BH&K<='42'.or.BH1>='8'
                QX&K='长期'
            case BH&K<='62'
                QX&K='20年'
            other
                QX&K='10年'
        endc
        K=str(val(K)+1,1)
        skip
    enddo
    CLEA
    wait '按任一健打印一张底卡'

```

```

D=chr(27)
A1='0018'
A2='0002'
A3='0034'
A4='0045'
SET PRINT ON
set device to printer
?? D+"IB"
@ proW(),8 say XM1
@ proW(),31 say XM2
? D+"W"+A1
?? D+"IA"
@ proW(),19 say XB1+space(10)+MZ1
@ proW(),65 say XB2+space(10)+MZ2
? D+"W"+A1
@ proW(),16 say CS1
@ proW(),62 say CS2
? D+"W"+A1
@ proW(),16 say ZZ
@ proW(),62 say ZZ
? D+"W"+A2
@ proW(),16 say ZZ1
@ proW(),62 say ZZ2
? D+"W"+A3
@ proW(),13 say QF+space(11)+QX1
@ proW(),59 say QF+space(11)+QX2
? D+"W"+A1
?? D+"IP"
@ proW(),7 say BH+BH1
@ proW(),44 say BH+BH2
?? D+"IA"
? D+"W"+A4
?? D+"IB"
@ proW(),5 say XM3
@ proW(),28 say XM4
? D+"W"+A1
?? D+"IA"
@ proW(),19 say XB3+space(10)+MZ3
@ proW(),65 say XB4+space(10)+MZ4
? D+"W"+A1
@ proW(),16 say CS3
@ proW(),62 say CS4
? D+"W"+A1
@ proW(),16 say ZZ
@ proW(),62 say ZZ
? D+"W"+A2
@ proW(),16 say ZZ3
@ proW(),62 say ZZ4
? D+"W"+A3
@ proW(),13 say QF+space(11)+QX3
@ proW(),59 say QF+space(11)+QX4
? D+"W"+A1
?? D+"IP"
@ proW(),7 say BH+BH3
@ proW(),44 say BH+BH4
?? D+"IA"
? D+"W"+A4
?
set print off
set device to screen
*
enddo
clos data

```

# 一个实用的微机共享存贮器通讯接口

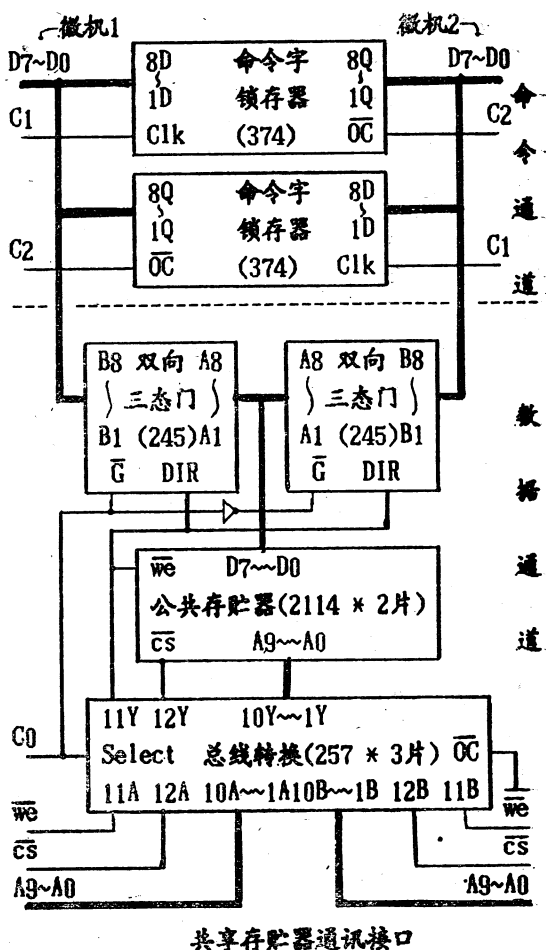
合肥电子工程学院 刘宏生

目前,微处理机在数字控制、仪器仪表等领域的应用越来越广泛,但鉴于微处理机的速度和存贮容量的限制,在复杂系统的开发和设计时,一般都采用分布式微机系统。

分布式系统区别于单机系统的最显著之处是通讯,因此通讯接口是分布式系统的重要环节,通过接口设计的关键点是解决通讯竞争。由于通讯接口是微机之间交换信息的必然通道,在某一时刻只能被一个微机占用,而另一微机如需使用则只有等待,通讯等待必将影响系统的实时性。如若某个时刻两个微机同时争用,即产生通讯竞争。因此在设计通讯接口时,首先要保证通讯的可靠性,其次要尽量满足通讯的实时性要求。

本文作者设计了一个共享存贮器通讯接口,较好地解决了上述问题,如下图所示。该接口具有较高的通用性和实用性。

本设计将通讯的信息通道分为命令通道和数据通道,对于命令信息由于它们非常重要而不允许等待,因而通过命令通道来传输。微机1和微机2均可随时通过C<sub>1</sub>控制线将命令字写入相应的锁存器中,而它们又可随时通过C<sub>2</sub>控制线,将对方的命令字取出,因而不存在通讯等待问题。为防止一微机在写命令字,而另一微机几乎与此同时取命令字,由于命令数据写入锁存器尚未稳定就被读出而出错,在软件设计时,取命令字设



共享存贮器通讯接口

打印程序见图1。这个程序是用于3070打印机的,其后部一些指令是控制走纸的打印机控制代码。如果使用其它型号的打印机,则这几个代

码要作相应的修改。具体可参看打印机使用手册。

附打印的底卡样张,见图2。

姓名 王祖贤

性别 男 民族 汉

出生 1951 年 5 月 3 日

地址 福建省福鼎县前歧镇芦竹

1988年12月31日签发 有效期20年

编号 352224510503301

(a) 正面

别名

出生地

文化程度

身高

单位职业

何地迁来

备注

图2

宗教信仰

籍贯

婚姻状况

兵役状况

米 血型

派出所

(b) 反面

# 气功信息模拟方法

中科院安徽光机所 奚居雄 张健如  
中国科学技术大学 刘振安

**摘要:** 本文提出用微机模拟气功信息的方法, 经临床验证, 效果很好。

## 一、引言

气功是我国古老文化中的一颗灿烂的明珠, 也是我国医学宝贵遗产的一部分, 是一门综合性的边缘科学。气功不仅为国内人民所喜爱, 在国外也享有很高的声誉。正确地练功, 可以强身祛病, 激发潜能, 开发智力。气功师还可以用自己释放出的外气为患者治病, 甚至对一些疑难病症也能起到神奇的效果。但是气功师每次发气都要损失自身的能量、消耗本身的元气, 所以很难满足众多患者治病之需要。而且气功师每次发功之效果无疑也并不完全相同。因此, 我们可以对经过临床验证是有效的发功过程进行对比研究, 去伪存真, 得到理想的发功治病过程, 用计算机进行模拟, 再用此模拟方法去调制  $\text{CO}_2$  激光束, 使  $\text{CO}_2$  激光以外气的形式, 模拟气功师发功的强弱变化作用于人体, 为患者治病 (这种仪器称做电脑激光气功医疗仪)。本文仅介绍这种气功信息模拟方法。

## 二、气功信息模拟方法

要想实现气功模拟, 首先要将气功师发出的外气用测量仪器测量并记录下来。再对这些大量记录与临床实验进行反复比较, 挑出治病效果好的过程曲线, 然后把这种生理信息离散化、量化, 以数字信号的形式存放起来。使用时, 用数模转换的方法把它转换成模拟量输出。该模拟量就是所需要的气功信息。致于气功信息再现, 可以有很多方法。电脑激光气功医疗仪是用  $\text{CO}_2$  激光器进行再现的, 即用气功模拟信息去调制  $\text{CO}_2$  激光光束, 使激光束按气功师发功的信息变化规律而变化, 起到了模拟气功的作用。气功信息模拟的原理框图如图1所示。数字信号储存在只读存储器中, 由中央处理单元根据监控程序按照一定的规律调用。中央处理机选用 Z80-cpu。

## 三、设计方法

总体结构如图2所示, 现对各组成部分分别叙述如下:

计成连续取两次, 若两次取值相同则有效, 否则, 再连续取两次。实践证明, 这样做是十分可靠的。

对于大量的数据信息则通过公共存储器数据通道来传输, 该公共存储器是两微机均可寻址的公用空间, 显然二者不能同时对它进行存取, 这里使用了两个措施来保证通讯的可靠性:

(1) 在命令字中设通讯握手标志位。

(2) 使用优先级结构

在某微机需使用数据通道时, 首先置命令字的通讯握手标志位, 向对方提出申请, 然后再取对方的命令字, 查看通讯握手标志位, 如对方回答允许则使用。如某时刻两微机同时申请, 则由微机1负责判优, 微机1拥有对多路总线转换器的

控制权 (图中  $C_0$  线), 它根据其当前的工作以及微机2的命令字进行判优, 以决定谁先使用数据通道。当微机1的控制线  $C_0$  为低电平时, 图中左边双向三态门  $\bar{G}=0$ , 右边  $\bar{G}=1$ , 使得公共存储器的数据线与微机1接通, 而与微机2断开, 同时, 总线转换器的A线进入Y线, 使得微机1的地址线、片选线  $\overline{\text{CS}}$ 、读写线  $\overline{\text{WE}}$  进入公共存储器以及双向三态门的DIR引脚, 显然此时将由微机1对公共存储器进行存取。当  $C_0$  为高电平时, 则由微机2对公共存储器进行存取。

本通讯接口与 Z80, MCS-51 等微机连接简单, 占用两边微机的资源很有限, 可方便地用于多微机实时控制等场合。

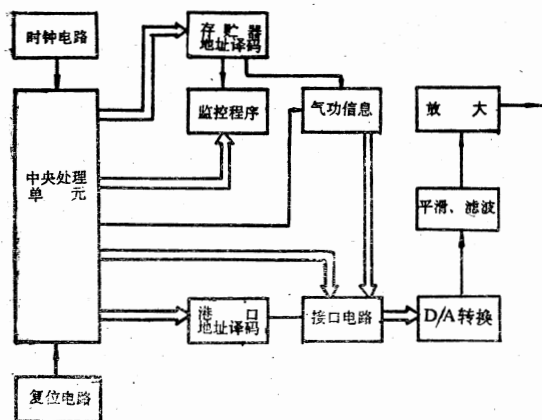


图1 气功信息模拟原理框图

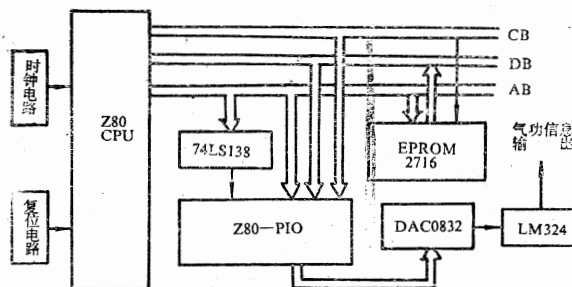


图2 总体结构框图

## 1. 芯片选择

(1) 中央处理器：选 Z80 作中央处理器。其数据总线为 8 位、控制总线有 13 根、本系统仅

用到 6，为保证稳定可靠及提高抗干扰能力，对未用的 7 根均作了相应处理。

(2) 数模转换：选用 DAC 0832 芯片作为数模转换芯片，用以把储存在内存中的数字气功信息变成模拟量气功信息，也就是把数字信号转换成与其数值成正比的电压信号。其原理框图如图 3 所示。

0832 有两个寄存器，可以进行两次缓冲操作，因此具有更大的灵活性。 $\overline{CS}$  为片选信号，它与  $\overline{ILE}$ （输入寄存器允许信号）信号共同对写信号  $\overline{WR1}$  进行控制，用以将 8 位数字量输入信号锁存于 0832 内部的输入寄存器中。而  $\overline{WR2}$  写信号与传送控制信号  $\overline{XFER}$  同时有效时，又将输入寄存器中的内容传送到 D/A 寄存器中锁存起来。这时 D/A 转换器开始工作。 $I_{OUT1}$  为转换后的模拟量输出。 $I_{OUT1}$  和  $I_{OUT2}$  之和为一常数，则  $I_{OUT2} = \text{常数} - I_{OUT1}$ 。为了提高抗干扰性能，分别设有模拟量地（AGND）和数字量地（DGND），以保证在电路上能把模拟信号地与数字信号地分别归类相结，然后再把数字信号地和模拟信号地在一处互接完成单点接地。 $V_{REF}$  为基准电压，采取二次稳压以提高稳定性，从而保证转换的精度。电源电压  $V_{CC}$  可为  $5 \sim 15V$ ， $R_f$  为反馈电阻，由于片内已具备反馈电阻，故可直接与外接运算放大器的输出端相接。本设备是通过扩充的 Z80-PPIO 与 DAC 0832

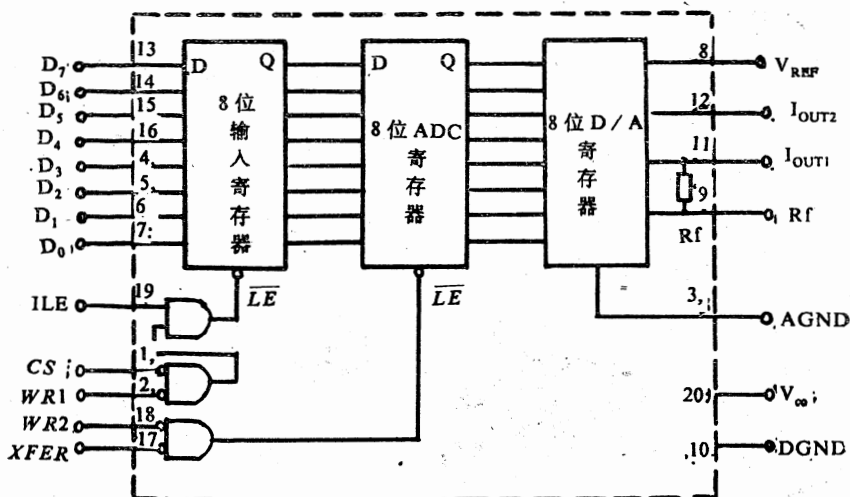


图3 0832原理框图

相接,其接法如图4所示。

(3) 接口芯片: 选 Z80-PIO 为 缓冲寄存

器。它是并行接口芯片并具有可编程的特点。外设可通过它直接与cpu交换信息,结构如图5所示。

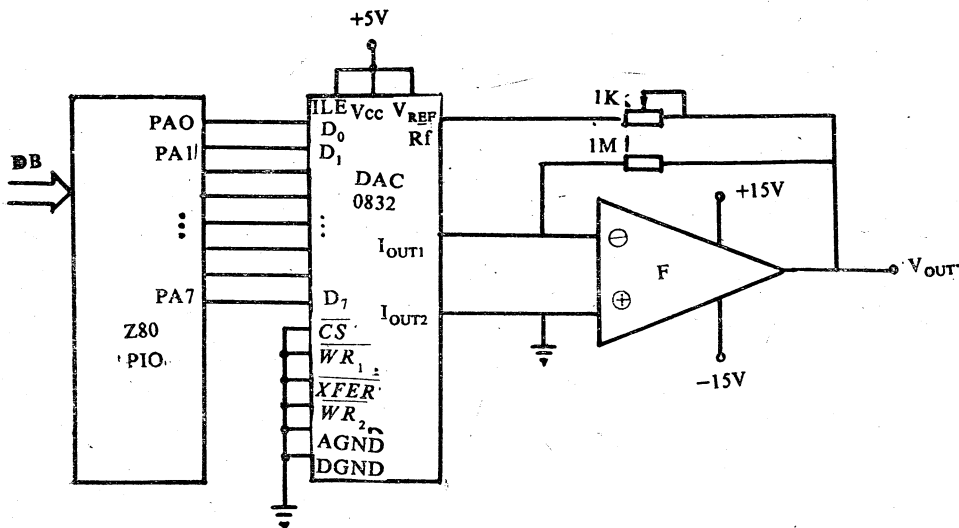


图4 0832与放大器及PIO的接线图

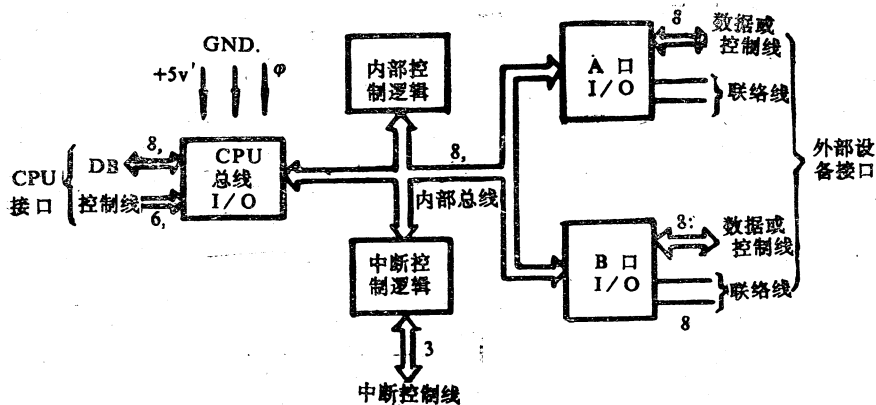


图5 Z80-PIO结构图

(4) 存储器: 通常, 在计算机中都有一些随机读写存储单元用来做为工作单元, 以存放程序运行中的变化量。本机为提高可靠性, 只选取只读存储器2716做存储器。而通过巧妙地编写监控程序, 使得本机可以不用随机读写存储器, 这是本机的一个特点。EPROM2716的接线图容易查到, 这里不再给出。

## 2. 线路原理简介

Z80 的地址总线、数据总线和控制总线连接着EPROM2716和 Z80-PIO, cpu 的系统控制

信号  $\overline{MREQ}$  直接接到 2716 的片选输入端上, 信号  $\overline{IORQ}$  直接接到 Z80-PIO 的 36 脚上。

存储器地址是由 cpu 的地址总线直接分配的。EPROM2716 的 11 根地址线接到 cpu 地址线的  $A_0 \sim A_{10}$  上,  $A_{11}$  接 2716 使能端 20 脚上, 当  $A_{11}$  为低电平时, 2716 地址有效。即 2716 的地址为  $000H \sim 7FFH$ 。高端地址  $A_{12} \sim A_{16}$  可取任意值。用 "X" 号代表任意值, 地址范围可表示为  $X000H \sim X7FFH$ 。本机工作过程中, 用到两组独立地址。即  $0000H \sim 07FFH$  和



# 体感诱发电位检测系统的设计与应用

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 刘英炳

脊髓、皮质体感诱发电位 (SPEP、CSEP), 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, 并对神经系统的损伤病变的程度和部位作出判断, 目前国内有条件进行研究或开展某项临床应用的单位, 多数是从国外引进设备, 但因价格昂贵, 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, 我们研制了简单、经济、实用、功能较全的APPLE I e微机体感诱发电位 (SEP) 检测系统。

## 一、硬件结构与工作原理

(详见本刊1989年第6期)

## 二、软件设计

SEP尤其是SPEP小至零点零几微伏, 深埋在比其大得多的各种心电、脑电、肌电及外界电场的干扰噪声之中, 用一般方法难以分离出来。根据它总是在刺激后一定时间才产生反应的规律, 严格以刺激时刻为准, 对每次所得数字信息序列逐点叠加平均, 增大信号振幅, 削弱背景噪声, 提高信噪比, 取出诱发反应, 数学公式为:

$$S_K = \frac{\sum_{I=1}^N Y_K}{N} \dots\dots (1)$$

反应电位的波峰幅值是反映神经 (脊髓) 功能状况的重要指标, 峰值的大小有无, 标志着神经功能的好坏, 提示了损伤病变的程度, 机器根据②③式自动寻找计算:

$$N_1 = (Y_{n1} - Y_+) \cdot \frac{10}{256}$$

$$N_i \approx P_i = (Y_{pi} - Y_{ni}) \cdot \frac{10}{256}$$

$$P_i \approx N_i = (Y_{n(i+1)} - Y_{pi}) \cdot \frac{10}{256} \dots\dots (2)$$

或

$$P_1 = (Y_{p1} - Y_+) \cdot \frac{10}{256}$$

$$P_i \approx N_i = (Y_{ni} - Y_{pi}) \cdot \frac{10}{256}$$

$$N_i \approx P_i = (Y_{p(i+1)} - Y_{ni}) \cdot \frac{10}{256} \dots\dots (3)$$

$N_1$ 、 $P_1$ 为负负首波峰值,  $N_i \approx P_i$ 、 $P_i \approx N_i$ — $i$ 个正负、负正峰峰值。

从刺激至产生诱发反应波峰的时间为峰值潜

2000H  $\approx$  27FFH。

Z80-PIO的地址由74LS138译码决定。本机选用80H  $\approx$  83H。80H和82H用以选中A口数据寄存器和控制寄存器; 81H和83H分别用以选中B口数据寄存器和控制寄存器。由程序设定其A口为DAC 0832的缓冲寄存器。PIO的 $P_{A0} \approx P_{A7}$ 接到0832的 $D_0 \sim D_7$ 上。PIO B口设定为输入, 根据该口的输入信号, 进行功能选择和定时控制。

放大器LM324用来把0832转换的气功信息

放大到规定电平以对CO<sub>2</sub>激光器进行调制。

## 四、结束语

用本文介绍的气功信息模拟方法, 很好地再现了气功信息, 并用其调制CO<sub>2</sub>激光输出, 使CO<sub>2</sub>激光器的输出按照气功信息变化来照射人体相应的穴位, 进行治疗。经大量临床应用, 对许多疾病都具有很好的疗效。

在生物医学领域中, 有许多模拟量都可以采用这种方法进行再现, 以供教学、研究和诊断用, 具有广阔的推广应用价值。

潜伏时(期),它是衡量神经(脊髓)功能的另一个重要指标。潜伏时的延长,意味着神经损伤病变后产生了障碍,减低了传导速度,机器将自动根据④式依次报告对应的峰潜伏时:

$$\text{令 } M = I \cdot \Delta T$$

$$\text{如果 } M > D_T + F + R$$

$$\text{则 } NiL = I \cdot \Delta T - D_T$$

$$PiL = I \cdot \Delta T - D_T \dots (4)$$

$NiL$ 、 $PiL$ — $i$ 个 $N$ 、 $P$ 波峰潜伏时。

主程序设计流程如图1,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,一次性展示结果图象,一旦坐标曲线始段下方出现参考点造型游标“1”指击“←”“→”键,将其移至欲设的位置,按下“↓”键,即刻闪现以被选点为中心的“十”字架,参考点被建立,同时在其右边、曲线的上方再现与前同状的游标,照例击用“→”“←”键,顺序移至的确为峰的适中点,再按“↓”键,峰点上下方依次闪现波峰序号、峰值峰峰值、峰值潜伏时等结果,欲打印并将图形数据结果存档时,按“↑”键,单独存盘按“M”键,清除重来按空格键,强行停机同时按下CTRL-Res键。

### 三、检测方法

受试者舒适放松地仰卧于安静的常温房间的检查床上,刺激电极分别置于正中神经、胫后神经皮肤表浅处,先用酒精棉球脱脂,点滴清水,用绷带系紧,地线极分别置于小臂、小腿中部。

刺激正中神经检测上肢或胫部CSEP,记录电极分别置于头颅中线旁开7-8厘米,两外侧耳道孔与颅顶正中点(CZ)的连线后方2公分,即中央后回手区(C3、C4);颈5(C5)等,记录外周神经传导速度置Erb点,参考点同置前额中央(FPZ)。

刺激胫后神经检测下肢CSEP,记录极置于头颅中线正中点(CZ)后方2厘米,即中央后回的下肢区域,测下肢外周神经传导速度置正极于腓窝中点内侧(PF),参考极分别置于FPZ、髂前上髁(IC);查躯干胸腰段SpEP,正负两

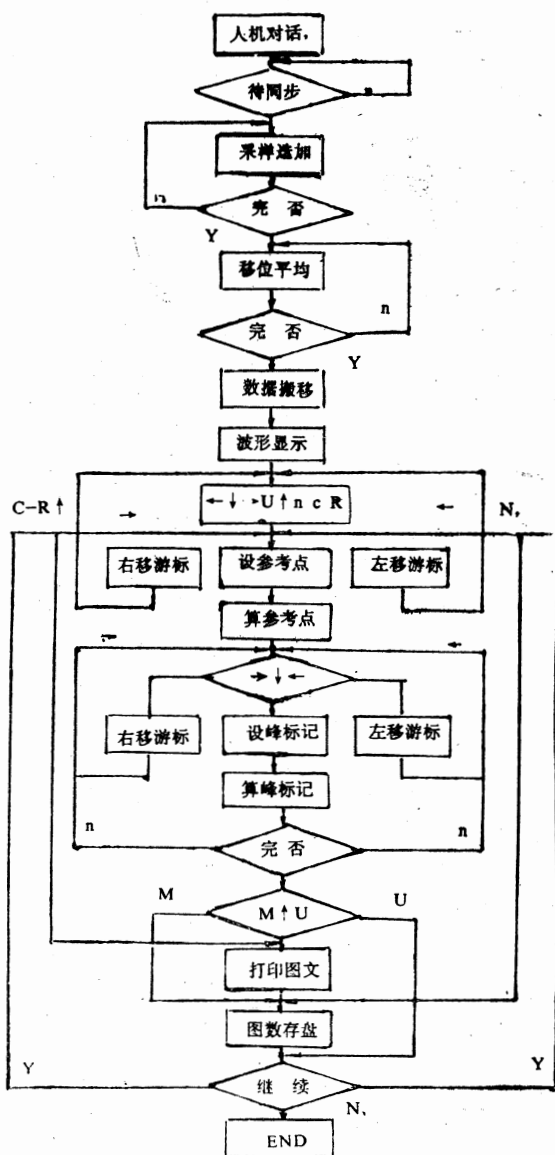


图1. 主程序流程图

极总是每间隔一个椎体测完一次向上递移一个位置,得到L5-L3、L3-L1至T9-T7跨椎体节段的SPEP。

记录电极与皮肤接触点,同样先用酒精棉球脱脂,涂上导电膏,耦合皮肤电阻 $6k\Omega$ 以下。

刺激器予置产生波宽0.2ms、频率4HZ,强度超过刺激阈,见足趾微动而宜的刺激脉冲。放大器放大数十万倍,频率范围1-3kc,分析时

间 $50 \sim 100\text{ms}$ , 平均次数512。

#### 四、应用结果

本系统经过大量动物实验, 获得了不同损伤时间、不同损伤程度的反应电位波形、数据(本文略), 探索了不同损伤时间、诱发电位与神经功能三者之间的变化规律及相互关系。先对男18例, 女12例, 年龄23岁至55岁的正常人分别做了CSEP、SPEP等检测, 获得了正常人体的标准波形(图2、3)及正常值(见表1、2、3)。然后在4个医院7个科室检测250多个病人, 获得了各种异常波形(图1、2、3)及数据(本文略), 临床符合率达96%以上, 一些疑难病症得到了早期诊断。

仅以SPEP的结果表明, 脊髓损伤节段以下两侧总能获得异常电位。少数损伤时间甚长, 营养条件太差, 长期缺乏活动锻炼, 肌肉严重萎缩者, 两侧记录不到电位的现象也有存在; 被损伤节段和该节段以上各段有的电位异常存在, 有的完全消失。异常存在者, 体征症状及功能表现为不完全损伤即不完全性截瘫; 完全消失者, 体征症状及功能表现为完全损伤即完全性截瘫。电位

表1. 两侧外周神经电位正常值 ( $n=15$ )

N1波		胫后神经		正中神经	
		Pf-IC		Erb-FPZ	
峰幅值 ( $\mu\text{V}$ )	L	$0.52 \pm 0.27$		$1.40 \pm 0.52$	
	R	$0.65 \pm 0.35$		$1.25 \pm 0.54$	
峰潜时 (ms)	L	$8.16 \pm 0.73$		$10.10 \pm 1.42$	
	R	$8.25 \pm 0.79$		$9.90 \pm 0.68$	

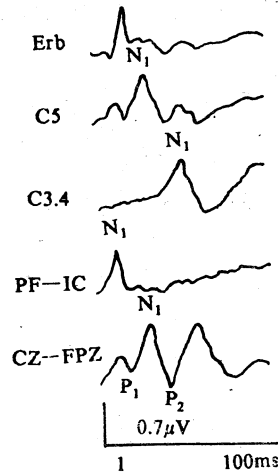


图2. 正常人体的CSEP波形

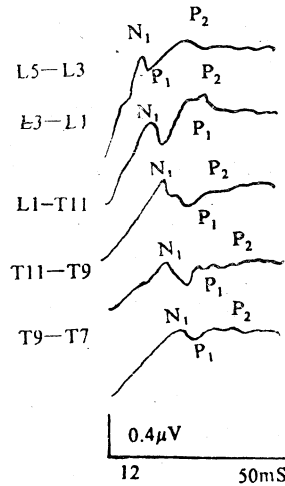


图3. 正常人体的一个椎体的SPEP波形

表2. 脊髓两侧电位正常值 ( $n=15$ )

N1波		胫后神经					正中神经	
		L5-L3	L3-L1	L1-T11	T11-T9	T9-T7	C5-FPZ	
峰幅值 ( $\mu\text{N}$ )	L	$0.37 \pm 0.18$	$0.36 \pm 0.18$	$0.41 \pm 0.15$	$0.34 \pm 0.14$	$0.20 \pm 0.07$	$0.90 \pm 0.26$	
	R	$0.42 \pm 0.13$	$0.41 \pm 0.17$	$0.44 \pm 0.02$	$0.34 \pm 0.09$	$0.22 \pm 0.11$	$0.64 \pm 0.19$	
峰潜时 (ms)	L	$17.73 \pm 1.08$	$18.91 \pm 1.42$	$21.13 \pm 1.06$	$21.46 \pm 1.15$	$22.48 \pm 0.97$	$12.95 \pm 0.97$	
	R	$17.94 \pm 1.21$	$18.66 \pm 1.52$	$20.82 \pm 1.15$	$21.40 \pm 1.30$	$22.15 \pm 1.10$	$13.48 \pm 1.06$	

表3. 两侧皮质体感诱发电位正常值 ( $n=15$ )

N1波		胫后神经		正中神经	
		Cz'-PFZ		C'3-PFZ	C'4-PFZ
峰幅值 ( $\mu\text{V}$ )	L	$0.73 \pm 0.51$		$0.80 \pm 0.34$	$0.80 \pm 0.34$
	R	$0.70 \pm 0.43$		$0.67 \pm 0.37$	$0.67 \pm 0.37$
峰潜时 (ms)	L	$40.17 \pm 3.23$		$19.43 \pm 1.40$	$19.43 \pm 1.40$
	R	$39.13 \pm 2.53$		$19.42 \pm 0.93$	$19.42 \pm 0.93$

从某节段开始异常或消失,则说明脊髓(神经)就损伤在该节段或以上段,刺激左右两侧结果不同,体征症状及功能自然而异,为临床的早期诊断和监护,提供了可靠的依据,增添了新的方法、新指标、新设备。

### 五、医学价值

可对周围神经损害,患脊柱间盘突出、轴索病、病毒性脑炎综合症等一系列有损于神经根从神经的外伤与疾病作出判断;可以客观地判断脊髓损伤的部位与大概水平,一般要比体征检查更符合实际;可以监护容易引起神经并发症的脊柱畸形矫形手术,探测脊髓机能状况,监护脊髓功能;诊断外周神经、末梢神经的损伤与疾病;判断脑干、丘脑、大脑皮层不同部位的损害;还

可对脑死亡、精神病和糖尿病等多种疾病提供诊断依据。

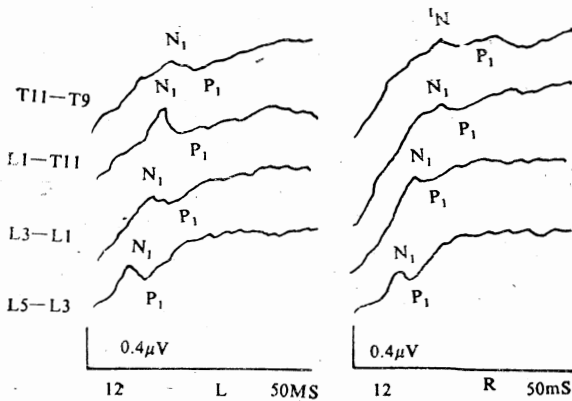


图4. T<sub>11</sub>、T<sub>12</sub>节段骨折的异常波形

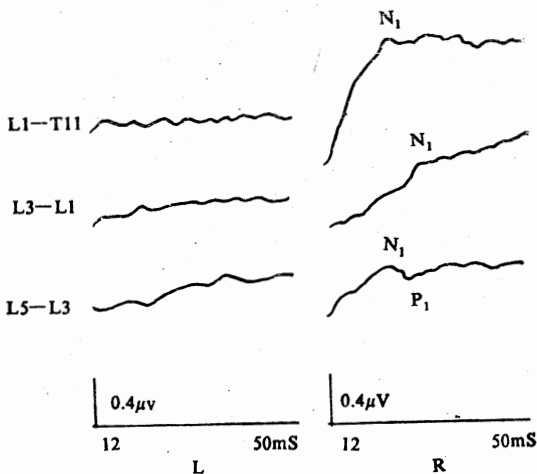


图5. L<sub>1</sub>、T<sub>12</sub>节段骨折的异常电位

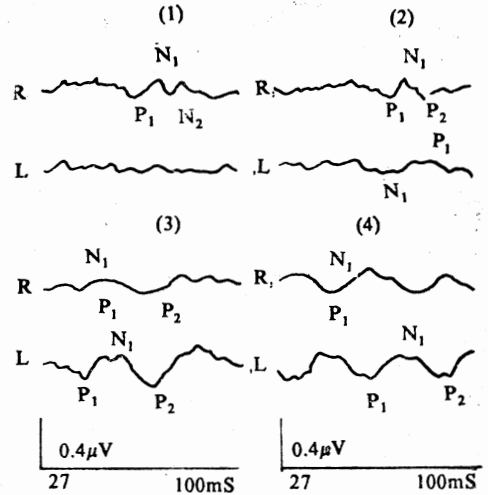


图6. 足体感区CZ-FPZ点记录的CSEP异常波形

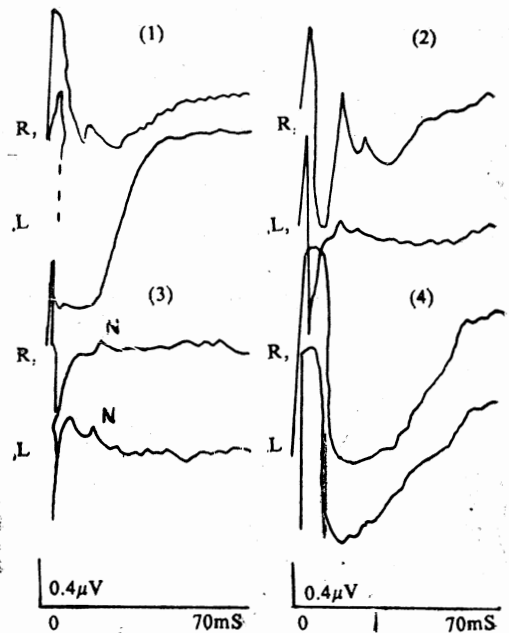


图7. 脑窝PF-IC点记录的CSEP异常波形

# Z-80微机系统中大量扩展I/O接口和存储器的巧妙方法

西南交通大学 计算机系 方旭明

我们一般认为Z-80微型计算机指令系统输入/输出类指令对于I/O的操作只有八位地址,即 $A_0 \sim A_7$ ,而实际上I/O类指令都是对十六位地址进行操作的,即对于指令IN A, (n)由(n)提供地址低八位,Acc提供地址高八位,其余指令都是由(C)提供地址低八位,(B)提供地址高八位。因为对于地址高八位的操作没有直接体现在指令的助记符中,所以往往被忽视。如果我们充分理解其含义,巧妙地设计我们的系统,就可以大量地扩展I/O接口或存储器。

## 一、扩展I/O接口

在Z-80微机系统中,若只对地址 $A_0 \sim A_7$ 进行译码,则可以产生256个I/O口地址信号,即寻址256个外设接口;若我们统一对地址 $A_0 \sim A_{15}$ 进行全译码,则可选择 $2^{16}$ 个口地址,要注意的仅仅是在执行I/O指令之前,要送高八位地址给 $A_8 \sim A_{15}$ 。典型电路如图1所示。当然,在

实际系统中我们根据所需I/O口的数量来决定译码器中参加译码的地址总线数目。可取 $N = \lceil \log_2 M \rceil$ ,其中N为参加译码的地址总线数目,M为外设接口数目。

## 二、扩展存储器

对于微机系统中一些不需要CPU频繁管理的存储器(如带CRT控制器的显示缓冲区等),可以采用口地址的方式来寻址,这样可以达到以牺牲少量口地址来换取大量存储器单元的目的。如图2为TP-805微型计算机中采用的显示缓冲区VRAM的连接电路。

在图1中由74LS139译出I/O口32个片选信号 $E_0H \sim FFH$ ,以此作为控制74LS244及74LS245的门控信号。通过巧妙安排CPU地址线与存储器RAM的地址线对应关系就可以对其8K字节进行读写操作。由于只有在修改画面内容时才需要更新VRAM内容,所以,没有必要让其占

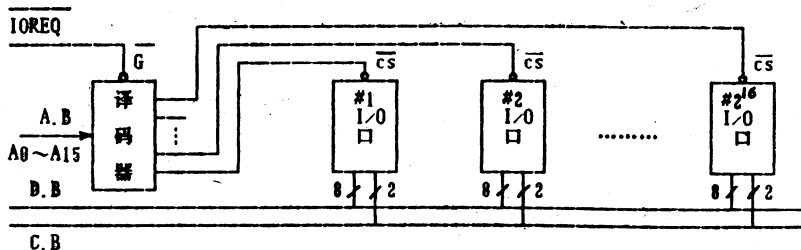


图1

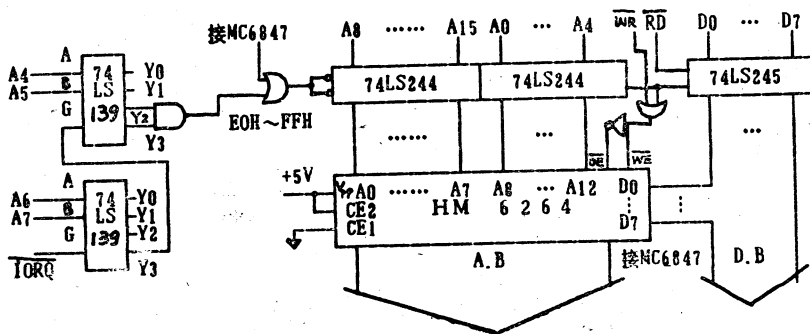


图2

用过多的用户资源。

CRT清屏程序如下:

```

LD  A,0F0H ; 给CRT控制口
OUT (0DCH),A ; 送方式控制字
LD  C,0E0H ; 置全图形方式
LD  B,00H ; 置VRAM地址
LD  B,00H ; 低字节初值
LD  B,00H ; 置VRAM地址
LD  B,00H ; 高字节初值
LOOP2: LD A,00H ; 置全暗光点
LOOP1: OUT (C),A ; 往VRAM写数
INC  B ; 高八位地址加一
JR   NZ,LOOP1 ; 不足256字节,
            ; 循环
INC  C ; 低八位地址加一
LD  A,0F8H ; 不足6K字节,
            ; 循环
CP   C ;
JR   NZ,LOOP2 ;
HALT ; 结束。

```

在此程序中用  $E0H \approx FFH$  32个口地址 就完  
成了6K字节VRAM的读写操作。

如图3所示为TP-STD8201EPROM编程  
卡电路原理图, EPROM编程卡的核心任务就是  
要将内存的内容固化到某一EPROM中, 或实现  
两EPROM的拷贝, 因此, 就没有必要让拷贝或  
被拷贝存储器固定地占据用户大块存储器区域。

我们来分析一下图3是如何妙用OUT(C),  
A指令的, 由(B)送出高八位地址送上一片锁  
存器74LS273, 由(C)送出的低八位地址产生  
I/O口片选信号00H或01H来选择二片锁存器273  
或一片锁存器374, 由(A)送出数据至D0—  
D7进入下一片锁存器74LS273。对于地址线A1  
4的译码分别作为EPROM的编程信号。详细工  
作过程不在此赘述, 可参考有关资料。这里要  
说明的仅仅是此电路用了两个口地址00H和01H,

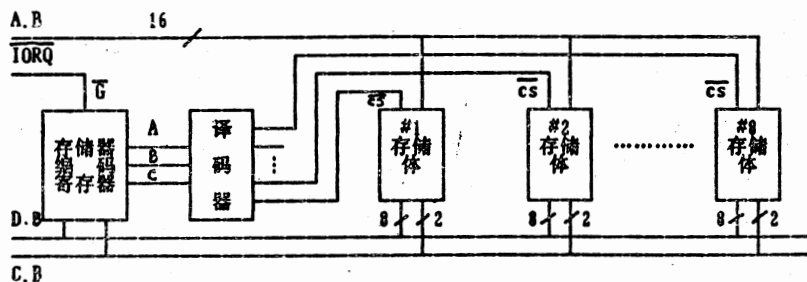


图3

就实现了对2716、2732、2764、27128等EPROM  
的编程写入, 而不是象TP-801A单板机那样固定  
地占有一个存储器区域 (PROM2, 1000H  $\approx$   
17FFH)。请看 EPROM操作过程。

例: 把内存3000H  $\approx$  3100H的内容写入宿片  
0000H开始的单元中去。

```

LD  HL,0000H ; 送EPROM首地址
LD  DE,3000H ; 送内存首地址
LD  BC,0100H ; 送字节数
EP1: LD  A(DE) ; 从内存取数
      OUT (01H),A ; 送374锁存以备写入
      LD  A,H ;
      OR  0C0H ; 置CE, PER信号
      PUSH BC ; 保护字节数
      LD  B,A ; 送EPROM地址高字节
      LD  A,L ; 送EPROM地址低字节
      LD  C,00H ; 送口地址
      OUT (C),A ; 向EPROM送地址及控制信号
      POP BC ;
      CALL D50MS ; 延时50MS, 让信息可靠
                  ; 写入
      LD  A,H ;
      AND 7FH ;
      PUSH BC ;
      LD  C,00H ;

```

```

OUT (C),A ; 撤除编程信号, 停止写
            ; EPROM

```

```

POP  BC ;
DEC  BC ; 修改指针
LD   A,C ;
OR   B ;
JR   Z,EP2 ; 判是否传完, 未完继续
INC  HL ; 修改指针
INC  DE ;
JR   EP1 ; 未完, 循环
EP2: HALT ; 结束。

```

### 三、存储器的体选择

在某些场合, 计算机要求有较大的存储器容  
量, 如具有汉字库的微型计算机系统, 其显示缓  
冲区往往很大, 对于Z-80CPU其地址线只有16  
根, 即使采用全译码方式其直接寻址能力也只有  
64K。在此情况下, 我们可以借助 I/O的选择功  
能来实现存储器的扩充寻址——体选择。其主要  
思想是: 将存储器划分成若干个存储体, 每个存  
储体内部的寻址信号仍然由16位地址线提供, 而  
每一个存储体是否被选中, 则应由存储体选择信

# 六十五吨玻璃窑炉微机控制系统

吉林省计算机技术研究所 阎立恒 刘晓光

## 一、前言

玻璃熔窑微机控制系统,是利用微机技术改造加热设备的项目。玻璃制品生产大致分为配料、熔制、成型加工等过程,产品产量的增减,质量的好坏,成本的高低,决定玻璃熔制质量好坏,也决定玻璃熔窑各项热工参数稳定程度,玻璃熔窑又是玻璃加工的主要耗能设备。

该微机控制系统主要任务,是节约能源,熔制出高质量玻璃熔液,满足成型时的工艺要求。该系统是个实时性控制系统,包括多回路多参数闭环调节控制,熔窑温度和油压闭环调节控制,开关量过程检测与控制,三个料道出口料滴数计量,自动火焰换向,熔窑自动加料,风压检测,油流量的计量等多种功能。

用常规仪表进行控制,虽然比手动控制有改进,但调节仪表的精度低,只能实现单回路调节控制,抗干扰能力差等等,不能对熔窑热工参数

实现较为理想的控制。

我们利用的微机控制系统与其它微机控制系统相比较,具有自己的特点:

该系统稳定可靠,抗干扰能力强,控制对象规模较大,测量点路数多,温度检测点的选择和方案经济合理,控制精度高,根据需要可以实现在线修改参数,改变工艺状态,火焰换向时控制收敛好,过渡过程快……等。

## 二、系统组成和作用

1. 玻璃熔窑微机控制系统硬件总体框图如图一:

图中分析检测信号传感器、微机、电控装置、机械部份。四部份通过信号相应对接构成一个闭环调节控制系统。机械部份是完成具体任务的执行机构;电控装置部份是指挥各部分协调工作,信号匹配的纽带;微机是构成整个控制系统的核心;检测传感器,把熔窑各种信号采集进

号来决定。这个选择信号由译码器提供。如图4所示。

存储器的操作过程如下:

1. 先送体选择信号,即通过 I/O 指令输出并锁存体选择信号,编码寄存器锁存的信号送译码器选中某一个存储体。

2. CPU 通过地址总线在选中的存储体体寻址,并进行读写操作。

在上面的原理图中,我们也可以认为整个内存由19根地址线组成,其中低16根由原CPU地址总线提供,高三根I/O口地址低三位来提供。采用这种方法寻址有得也有失,即得到了空间,失去了时间。显然,数据存取周期延长了。另外,每增加一个存储体,就要牺牲一个口地址。从理论上说,处理器整个内存可以扩充到 $2^{16} \times 64K$ 字节。

以上介绍的是针对Z-80CPU的微型计算机系统的,对于其它类型的处理器原理是类似的,也可以参考上述方法。

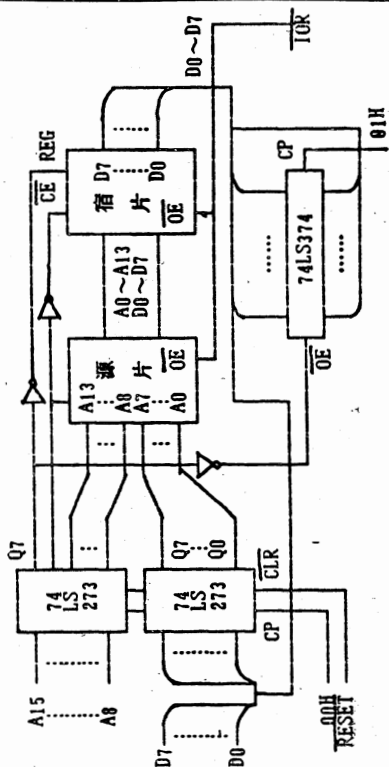
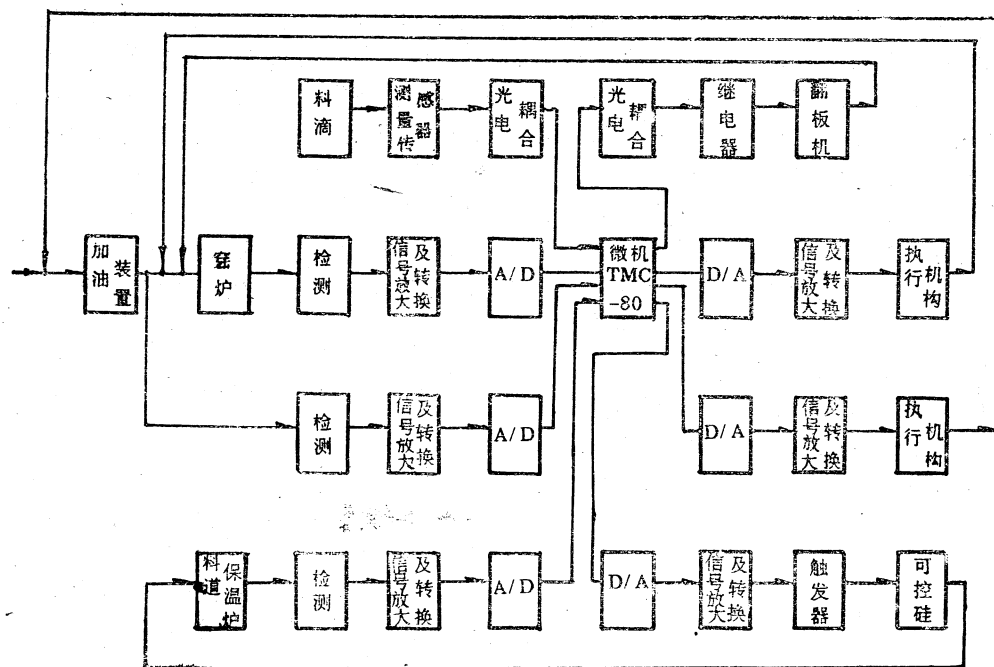


图4





图一. 系统硬件总体框图

来,并将采集的各种信号作输入,存贮和运算处理,处理的结果发出相应的命令,通过输出信息,驱动机械部份执行机构动作,使玻璃窑炉按着预期工艺指标运行,稳定熔窑各种热工参数,形成一个完整的闭环调节控制系统,同时微机还将实时数据和过程发生的情况及时地通知操作人员,建立起人机联系。

2. 日出料量65吨玻璃熔窑,该窑连续工作熔化面积为 $40\text{m}^2$ ,工作池面积 $12\text{m}^2$ ,在窑炉结构上采用全窑保温,属于玻璃池窑炉。利用重油或油渣加热,每侧有油枪6只,两侧共12只,熔窑有三个料道出口,料道均设保温措施。

玻璃熔窑运行过程是个复杂的燃烧过程。熔窑温度受油温、油压、一次风压、二次空气量和温度、空气压力、窑压、烟道抽力等多种因素影响。为使微机在玻璃熔窑控制方面显示出比仪表控制的优越性,合理开发利用微机的功能,用单机实现多回路多参数的检测和控制。

3. 控制和检测回路的选择:具有检测参数多,系统控制规模较大的特点。从生产和实际需要出发,微机选择14路模拟量入,4路模拟量输

出。有八路开关量入,7路开关量输出,这些控制回路和检测点有:

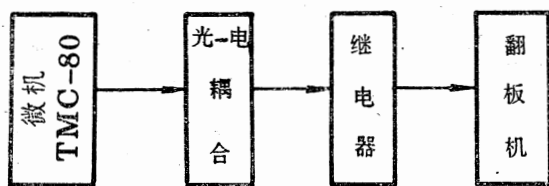
- (1) 熔窑温度回路; (2) 燃油压力回路;
- (3) 燃道1控制回路; (4) 燃道2控制回路;
- (5) 工作池检测; (6) 窑温检测;
- (7) 油压检测; (8) 一次风检测;
- (9) 油温度检测; (10) 南蓄热室温度检测;
- (11) 北蓄热室温度检测; (12) 碛西温度检测;
- (13) 碛东温度检测; (14) 料道温度检测;
- (15) 油流量计量;
- (16) 窑容液面检测与自动加料控制;
- (17) 火焰换向时间显示和自动换向控制;
- (18) 三个料道料滴计数并累计;
- (19) 熔窑温度、油压、风压、工作池温度……等几处自动报警。

#### 4. 火焰换向时控制收敛好,过渡过程快;

火焰换向是玻璃熔窑重要控制部分。熔窑参数最大波动产生在火焰换向过程中,换向前后熔窑温度和其它参数,如窑压、油压、风压、烟道含氧量……等都需要较长时间才能稳定下来。微机具有过程、顺序控制和执行编程功能。每次换向

过程中包括几个动作：例如关闭一侧管道上的油阀，雾化空气阀，二次空气交换方向；打开另一侧管道上雾化空气阀，油阀等。这些动作是预先设定时间顺序进行，动作过程中可允许在一定范围内对时序作必要的修改或调整。用软件实现阀位控制，使系统在火焰换向过程中加快，与“PID调节器”控制换向时相比快一分钟，熔窑的热工参数收敛较快，换向后熔窑热工参数很快稳定。

该系统调节控制过程中，使用“交差法”控制，使燃油充分燃烧，减少环境污染。图二是微机与“PID调节器”控制火焰换向时记录曲线，相同条件相比，不难看出微机控制火焰换向过渡过程快，熔窑温度稳定。换向过程控制框图如下：



#### 5. 熔窑温度检测点选择合理：

该窑熔化面积为 $40\text{m}^2$ ，经过实验及对熔窑结构分析，以大铂顶部为检测点，并选用铂铑—铂热电偶直接采集火焰温度，经过运行表明窑炉检测点的选择是正确合理的，检测火焰温度提高了系统的精度，铂铑——铂热电偶对温度信号反应比较灵敏稳定，把灵敏度较高的信号输入微机，经过微机运算处理再输出控制油阀和一次风阀产生火焰，从时间上说虽然还稍有些滞后，但控制上已接近熔窑的实际温度值，用微机运算处理从检测信号到输出时间比较短，在一个周期里对于输出执行三次调节控制，使熔窑温度变化幅度小，便于达到稳定运行状态。

#### 6. 自动加料装置：

熔窑液面控制，玻璃熔液流出使液面变化，液面传感器检测信号输入微机，微机控制加料机自动改变加料速度，保持液面稳定。

#### 7. 报警、显示功能：

有油压、风压报警。炉温、油温、磁工、磁

西、磁中温度报警等功能。30分钟换向有声音和屏幕显示功能。CRT屏幕显示有：油压、风压、油温、炉温、蓄热室，料滴计数…等14个参数显示功能。数码管显示控制台有温度、油压及火焰燃烧侧的温度和油压功能。

#### 8. 修改参数：

根据需要设置在线修改控制参数，熔窑温度、油压及换向时间给定值，改变工作状况，定时输出熔窑14种参数，满足检查统计管理要求。

### 三、电控装置硬件设计

#### 1. 微机的选择：

微机是该系统的核心部件，它的特性、功能对整个控制系统支持的传感器和接口电路都有很大的影响，微机除了在给定时间内，完成执行机构调节和控制外，力求体积小、成本低、本身损耗小、操作简便，微机本身抗干扰性能强，长期运行稳定可靠的机器。

本控制系统采用了TMC—80微机一台，来完成巡回检测，过程控制，实时数据处理，修改参数等全部工作。是一种可编程控制器，智能终端及带有CRT显示等。该机具有灵活性好，可扩展性强，可靠性高的特点。系统配有14吋显示器，80行打印机，磁带机和专用直流稳压电源（ $\pm 5$ 伏， $\pm 12$ 伏， $+25$ 伏），EPROM为32K，RAM为32K，Z—80CPU，158条指令，有较强的输入输出及多机中断处理功能，该系统的应用程序固化在EPROM里，动态数据在RAM中运行。

系统扩充到16路模拟量入，输入信号满量程为 $500\text{mv}—5\text{v}$ ，精度为0.1%，八路模拟量输出，输出信号为5伏，精度为0.5%，32点开关量输入输出供用户使用。

#### 2. 输入与输出电路

模拟量输入传感器信号有：（1）铂铑—铂热电偶、镍铬热电偶。（2）压力传感器。（3）液面指示传感器。（4）接近电子开关…等。熔窑测温传感器选择是很重要的，因为检测信号误差会影响系统的精度，我们选用铂铑—铂热电偶，测温范围（ $0—1600^{\circ}\text{C}$ ）实际应用在（ $1430^{\circ}\text{C}—1480^{\circ}\text{C}$ ），将传感器信号经放大后输入到微机，不降

低精度, 又不受其它干扰, 采用高精度放大器, 控制系统精度没受到影响。

接近电子开关是把料道生产过程中料滴数采集来, 用开关量输入微机, 计数和积累结果, 便于产量管理。

熔窑液面控制电路, 液面料位传感器输出开关量信号输入微机, 微机输出开关量控制加料机, 实现自动加料。

模拟量输出电路, 气动薄膜调节阀为执行机构, 执行机构的状态表示开度, 开度的大小控制油的流量, 微机输出0—5伏信号, 经v/I转换电路为0—10毫安电流信号, 再经电—气转换器, 将电信号转变为气动信号, 驱动薄膜调节阀, 进而达到控制熔窑温度目的。

玻璃熔窑要求30分钟火焰换向一次, 采用开关量控制直接驱动外电路, 通过外电路控制翻板机, 自动实现火焰换向过程。

#### 3. 料道加热器控制:

该熔窑有三个料道出口, 玻璃熔液流经料道出口到成型机, 这段料道要求玻璃熔液温度相当稳定, 才能满足成型时的工艺要求(温度、粘度、重要均用温度实现), 三个料道出口均采用碳化硅棒加热, 每台功率为15瓩, 采用可控制供电闭环调节控制电路, 经微机控制可控硅触发器, 控制了可控硅输出直流电压, 保证了碳化硅棒加热器工作状态, 使料道温度范围稳定在 $\pm 2^{\circ}\text{C}$ 范围

#### 4. 报警驱动电路:

是监视设备运行状态的环节, 包括油温、油压、风压、窑室工作温度、熔窑温度……等报警功能。除了在CRT全屏幕上有标记(↑超高, ↓超低)外, 还有供操作人员声音报井, 直到设备正常运行后报警结束。

### 四、应用软件设计

该系统应用软件全部使用汇编语言程序, 其特点是执行速度快, 占用内存空间少, 并能充分发挥CPU的功能。软件实现闭环调节控制, 控制过程中自动优化修改控制参数, 熔窑热工参数稳定。在火焰换向过程中进行阀位控制, 使系统控制参数收敛快, 保证了系统稳定运行。

主要控制模块有:

#### 1. 窑炉温度控制模块;

#### 2. 油压控制模块;

#### 3. PID运算模块;

#### 4. 检测模块包括:

风压检测, 油压检测, 工作室弦顶温度检测, 弦顶东侧温度检测, 弦顶西侧温度检测, 南蓄热室温度检测, 北蓄热室温度检测, 料滴个数计数和累计, 即产品数量统计模块, 流量计数和累计模块, 窑池温度检测;

#### 5. 数字滤波程序模块;

#### 6. 服务程序模块包括:

画面和汉字显示, CRT中断的设置, 换向时间显示, 修改参数, 初始化, 数码管显示: 有窑室温度和油压;

#### 7. 位控(I/O)模块, 换向时间;

#### 8. 报警模块;

#### 9. 热电偶冷端补偿模块。

该系统总的软件框图如图四所示:

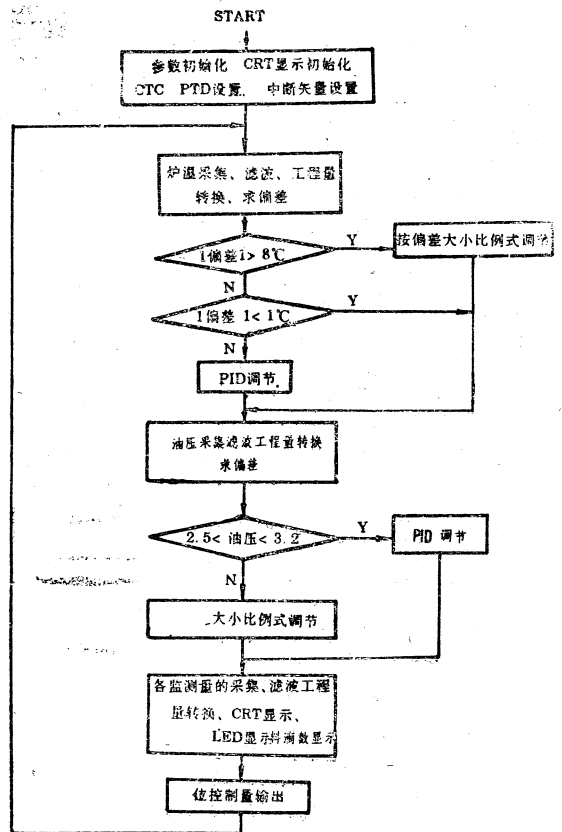


图4-1 主程序框图

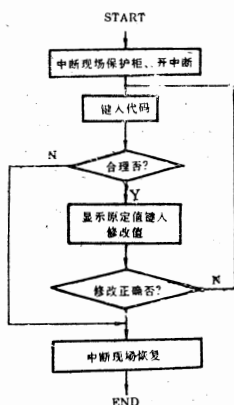


图4-2 键入中断流程图

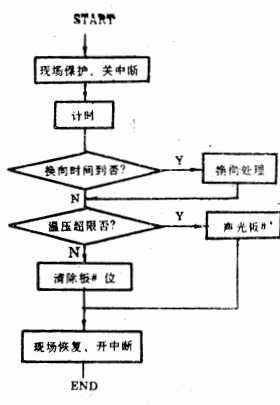


图4-3 CTC流程图

## 五、防止干扰问题

该微机控制系统所处的工作环境恶劣，上面是用于加运料的3吨电动葫芦，侧面是自动加料机，下面是控制翻板机的电动机和贮油罐控制阀，没有专用电源，机房也没屏蔽。开始投入时因现场严重干扰，系统不能投入正常运行，经过我们努力，在防止电压波动和电磁干扰方面作了许多工作，使该微机控制系统投入现场工作，长期稳定运行工作可靠。

采取措施是：

1. 交流稳压电源，车间电网电压受起动设备影响，瞬间电压可降到180伏，超出交流稳压电源的稳压范围，采用一空芯电感接到交流稳压电源的前端，控制了因电网电压瞬间波动影响，效果明显，使微机系统能投入电网。

2. 加强电源滤波，装设电源低通滤波器，只允许正常周波数交流电通过，对于电网波动幅度小及杂散的脉动、高频和低频干扰予以抑制和吸收。把交流和直流电源这两个环节，相应强加处理，使那些杂散干扰在进入微机前全部滤掉，保证微机能正常工作。

3. 优化电路设计和安装，控制电气开关转换，气动调节阀驱动电路和执行机构动作引起冲击及电磁干扰，采用光电隔离技术，布线走向合理，元器件布局合理，埋设专用地线，模拟地，数字地的处理……等均为现场要解决的。

4. 使用软件滤波程序，采集的信号接近实际值。

5. 为了增强抗干扰性，设置了自启动功能，并保持现场画面和显示数据，保证了系统的可靠性。

几年来该微机系统运行，长期工作稳定可靠

## 六、结果与讨论

1. 该系统已投入运行二年多，运行结果表明，实现了原来的设计指标，长期运行稳定可靠。该系统是多回路多参数闭环调节控制，用一台微机实现多路控制、检测、在线修改参数、料道保温、自动报警、液面检测、自动加料和产品管理等，使微机的功能得到充分开放利用。温度检测点的选择和传感器信号都保证了系统的精度，控制对象为日出料量为65吨玻璃液的熔窑规格大，投资较少，抗干扰能力强，该系统设计方案是经济合理的。

2. 玻璃窑炉是一次性起动设备，窑温随时受多种因素影响，其中一种因素发生变化，窑温跟着受影响，是个滞后的惯性环节，系统采用：铂铑—铂热电偶；并直接检测火焰温度作为主要输入信号，火焰温度反应速度灵敏，能提高控制系统精度，二者都起到了加快系统调节和控制反应速度作用。检测火焰温度和控制喷油枪调节滞后量较小，合理的选择采样时间和控制周期，除了能减小系统的惯作用外，还能使火焰长时间稳定燃烧，窑室温度变化一般在 $\pm 4^{\circ}\text{C}$ 。

3. 熔窑温度检测点的选择与火焰换向：

正确选择测温点，是工作过程中应重视的问题，因为检测点的选择（位置、热电偶插入窑室内深度），直接影响精度，经过长期运行，对于选择大磁顶部为检测点，运行结果表明测温传感器和测温点的选择是合理的。

“PID调节器”控制火焰换向过程，熔窑热工参数波动较大，熔窑温度变化一度下降 $25^{\circ}\text{C}$ ，后上升 $15^{\circ}\text{C}$ ，振荡三次，才能稳定下来。

微机有实时控制功能，自动选择控制参数进行阀位闭环控制，使火焰换向过程熔窑热工参数收敛的快，使换向过渡过程加快，熔窑温度很快稳定下来，换向过程窑温下降 $10^{\circ}\text{C}$ ，后上升 $5^{\circ}\text{C}$ ，超调一次，便可稳定于正常运行状态。

参考文献（略）

## 多级系统中IBM-PC/XT和STD总线通讯

黑龙江矿业学院 周树杰 党群 梁华 李广才

STD总线具有抗干扰能力强, 扩展容易等特点, 我们曾用它和IBM-PC/XT微机组成多级系统, 很好完成了现场的要求。本文以一生产信息管理系统中在线收集生产原始数据为例来谈一下PC机STD总线组成的多级系统的优点, 组成多级系统的主要要求以及一种比较简单通讯方式的组成和简单工作过程。

很多生产信息管理系统是运行在PC机上的, 为提高系统功能, 用户希望能把生产数据如流量, 浓度等通过传感器直接送到计算机内, 而不通过键盘人工键入。传感器的信息可以通过插在PC机扩展槽中的扩展卡直接送入微机, 传感器的信息也可先送给STD总线及有关模块, 它们做些简单处理, 存贮等工作, 等PC机需要这些数据时, 再由STD总线送给PC机。后一种方法较前一种方法有如下优点。

首先, 它增加了系统的灵活性。因为监测传感器的任务交给了STD总线及其模块, PC机在不需要传感器数据时, 可以做其它工作, 甚至可以关掉电源。当需要传感器的数据时, PC机执行相应的通讯程序, 把STD总线及其模块的数据传给PC机, 这样可提高系统的灵活性, 提高PC机使用效率。这一点对微机比较少的单位显得更重要。其次, STD总线扩展方便, 可较方便地扩充连接的传感器数量和类型。STD总线

抗干扰能力强, 可安装在现场。再次, 这种做法较前一种做法容易维护, 又省下了PC机扩展槽可做其它工作。

STD总线和PC机间通讯要满足下列基本条件:

1. 通讯过程中, 人不需要干预STD总线模块的工作。如通讯过程中不需要通过人工启动STD总线模块的通讯程序等。

2. STD总线和PC机之间连线简单可靠。

3. STD总线和PC机通讯至少要半双工, 因为PC机和STD总线之间要互送信息, 比如为了避免干扰引起的误动作, PC机向STD总线传送口令, STD总线及其模块核对口令, 口令正确了才能发送数据。

考虑到上述条件, 又注意到STD总线和PC机的特点, 我们选择了STD总线、CPU模块、传感器接口模块和串行接口模块组成了一个智能接口, 该接口利用串行接口模块和PC机的RS-232接口之间进行通讯。具体连线如图1所示。

这样做PC机不做任何硬件改动, STD总线的串行接口板上备有对外接线端, PC机和智能接口连线是普通双绞线。它们之间的通讯是异步半双工基带传输方式。

它的具体工作过程简述如下:

智能接口的通讯功能是通过一个中断服务程

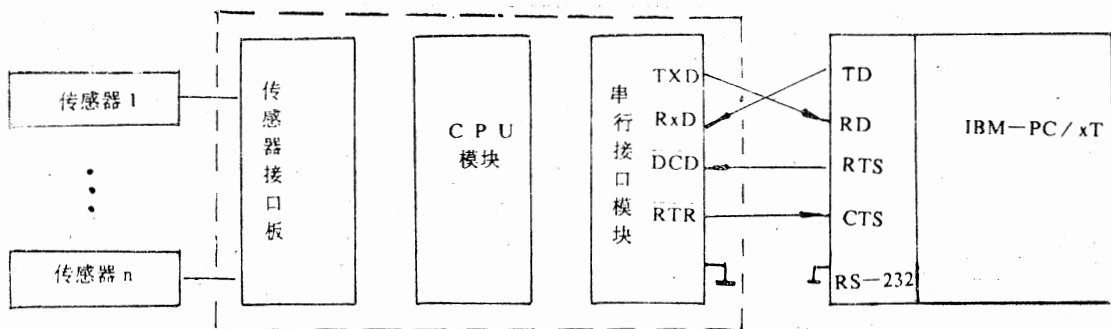


图1

序来完成的。当PC/XT机通过RS-232的RTS端把要求通讯的信号传送给串行接口模块的DCD端时,智能接口则执行中断服务程序。其程序框图如图2所示。

中断服务程序的主要功能是完成数据代码转换,根据PC机发来的特征字,决定清数据区后并向PC发送数据等。

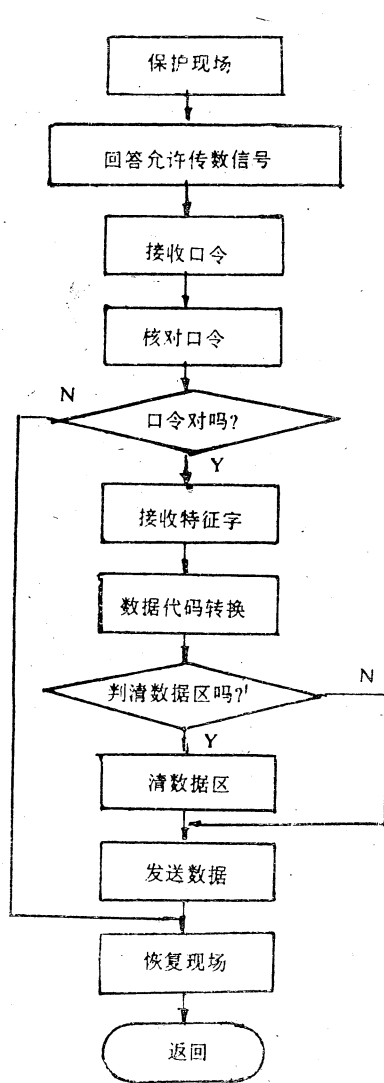


图2

PC的通讯子程序框图如图3所示。

我们在某选煤厂的生产管理系统中采用了上述方案。该选煤厂有五条自制电子皮带秤,我们把皮带秤的信号送给智能接口,智能接口和信息中的微机相连,信息中心微机通过计算机网络和其它科室微机相连,经过近一年实践运行证明此方案完全可行。

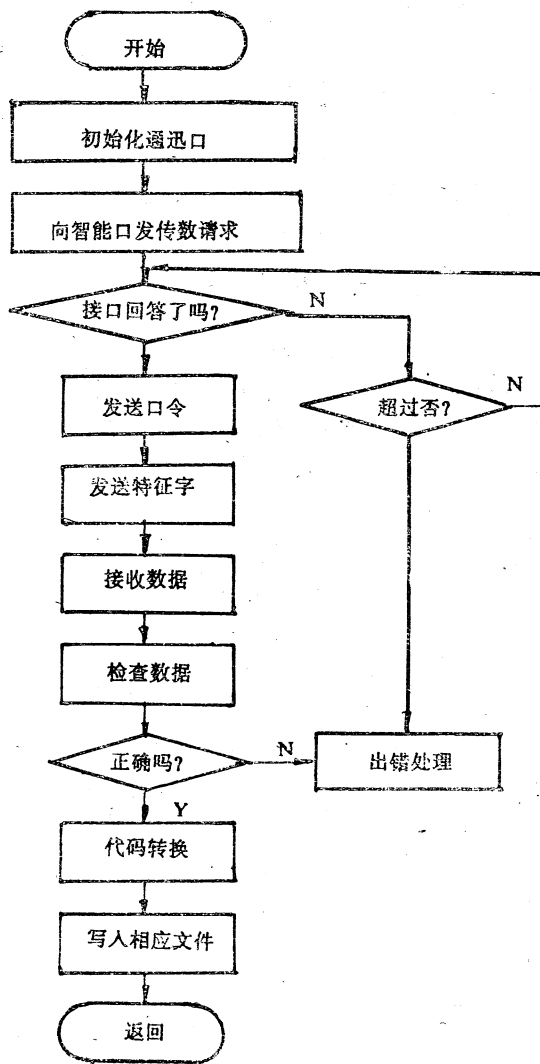


图3

# VAX计算机接口板DHV11的故障分析

浙江大学电机系 高济平 蔡新尧

**摘要:** 本文介绍了MICRO VAX I 计算机常见的由于带电拔下或插上终端接插件而引起的DHV11接口板的损坏。分别从软件和硬件二个方面来诊断故障所在,分析故障原因,从而排除故障,以保证计算机的正常工作。

MICRO VAX I 计算机是美国DEC公司生产的32位超级微型机,它的接口板大多数都是采用DHV11八路全双工异步转换板。DHV11是八路异步直接存贮访问多路器,它提供主机与EIAS232—C终端之间的连接,使访问直接存贮器减少了由于终端频繁使用而引起的系统开销。DHV11在每条线上使用全调制—解调制控制,在全双工方式工作时,编程或跨接线可选择速度最高为38400位/秒。DHV11在每条线上都支持分速发送和接收。

## 一、故障现象

我们的MICRO VAX I 计算机最近运行不正常。在系统引导时,经常发生突然中断,机器死机。这时敲任何键都不起作用,因此无法确定故障所在。有时当系统引导出来后,发现有一个终端TXA6:不能与主机联机。但是该终端在脱机时,终端本身工作正常。

## 二、故障诊断

在开机过程中,怎么能判断引导系统失败,机器死机呢?

在开机前,打开主机前端面板。以至在开机过程中可以观察到:面板上DUA0:和DUA1:的二只Ready指示灯的绿色发光二极管闪烁或一只闪烁,这说明了闪烁的发光二极管所指示的那只硬盘正在工作。若二只发光二极管都亮了,而且一动也不动,(即不闪烁)。同时屏幕上显示停止不动,这就说明了系统引导中断了。

这时我们采取强制引导系统的方法:把主机后盖打开,把允许/禁止中断开关▽从“○”位置向上肉到“●”位置上,这样允许强制中断。然后按键盘上的Break键(即F5键),按如下过程引导:

```
>>>B/1 DUA0:
2...1...0
SYSBOOT>SET/STARTUP OPA0:
SYSOBT>CONTINUE
Micro VMS Version V4.4 09-Mar-1988
10:00.....
$SET DEF DUA0: {SYS0.SYSEXE}
$@SHUTDOWN
```

此处说明一下:为什么要先关机一次?因为当计算机是非正常关机时,(即在开机或关机时,机器突然中断而死机)这时机器没有正常关机,有些硬件设备没有及时拆卸(这里指软件拆卸)。这样在下次开机引导时,会显示如下的信息:

```
% MOUNT-I-REBUILD, Volume was
improperly dismouned; rebuild in pro-
gress.
```

该信息说明:由于不正确地拆卸磁盘,会破坏该磁盘上的结构信息,因此需要重新建立该磁盘上的文件结构。

在正常关机后,再按Break键:

```
>>>B DUA0:
```

或者按主机前端面板上的Restart/run键,也可以引导系统开机。

但是为了更有效地监视开机引导,及时发现故障所在,也可以做以下工作:

```
>>>B/1 DUA0:
SYSBOOT>SET/STARTUP OPA0:
SYSBOOT>CONTINUE
$RUN INSTALL
INSTALL>EDIT/OPEN/HEADER/
SHARED
$SET DEF DUA0: {SYS0.SYSMGR}
$EDIT SYSTARTUP.COM
```

把该文件中SET NOCONTROL=Y改成SET CONTROL=Y,把SET NOVERIFY改成SET VERIFY。这样随时监视系统引导的全过程,在引导中发现错误可以按CTRL/Y或F6键强制退出引导。也可以发现在



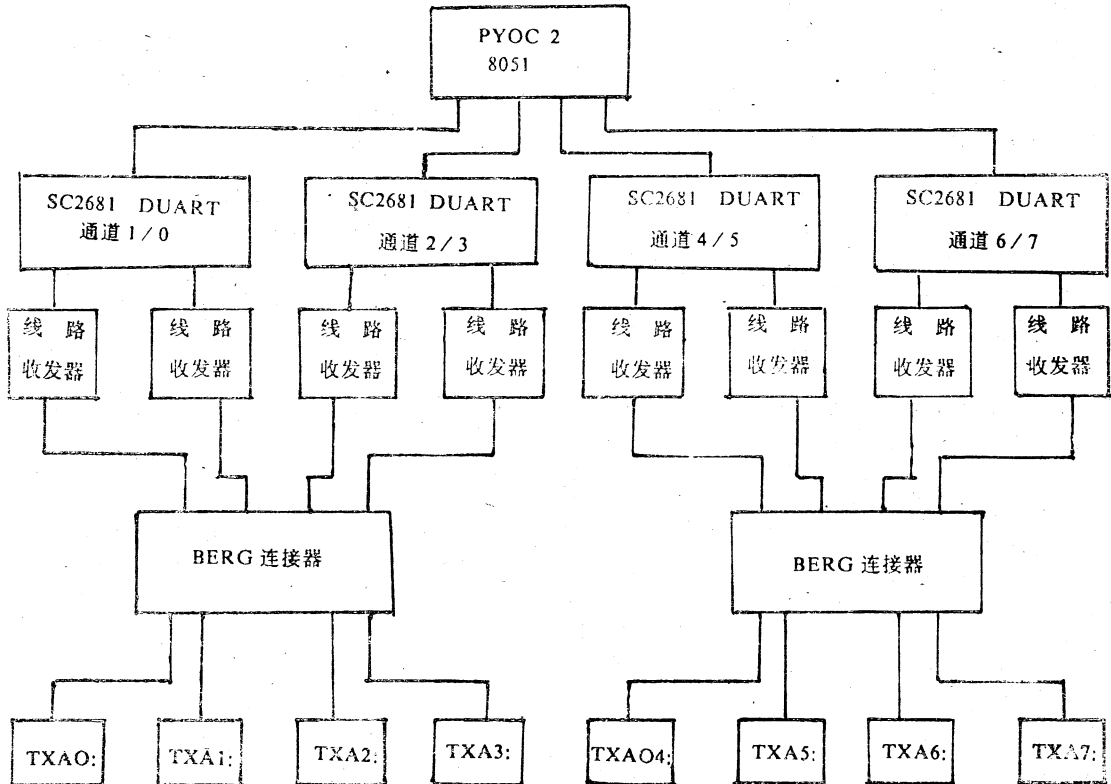
引导系统时,当系统中断死机时,操作系统正在执行哪一条命令,从而可以发现故障所在。

在做了上述工作后,发现系统引导中在设置TXA7:和TXA6:时,机器死机。初步判断故障与TXA7:和TXA6:有关。同时通过更换终

端和电缆线的方法,排除了终端和电缆线可能损坏的因素,确定是TXA7:和TXA6:的接口电路坏掉了。

### 三、故障分析

经查阅电原理图,框图如下:



DHV11接口板上的proc 2是一片8051单片机它减轻了主机系统的许多数据处理任务,控制4个SC2681双通用异步接收发送器DUART(Dual Universal Asynchronous Receiver Transmitter)。DUART执行数据的串行/并行或者并行/串行之间的转换,每一个DUART有二个通道,也就是接二个终端。由此看出TXA6:与TXA7:是共用一片DUART。现在TXA6:与TXA7:二个终端都不能与主机联机,说明了第4片DUART已经损坏了。

根据情况调查,我们分析:TXA7:是我们专门用于IBM-PC计算机与VAX计算机之间连机,进行IBM-PC计算机与VAX计算机之间的程序、数据的转换,平时不大常用,只是在需要

转换时才接上去。而在拔下或插上时一般情况下都是带电操作的,即VAX机和IBM-PC机都没有关机,从而引起烧掉了接口电路,也就是损坏了通道6/7共用的DUART,使得TXA6:也不能与主机联机。

### 四、注意事项

计算机的终端的接插件在拔下或插上时一定要切断电源,即先关机,后拔插接插件!

同时请注意:DHV11接口板是多层细线腐蚀PCB印刷电路板,自己不能随便更换元器件。若要更换,则一定要非常小心。在DHV11板上只有带插座的8051单片机是可以由用户更换的。

但是国内的DEC公司的VAX计算机维修站都规定:凡是用户自己焊接过的板子(即自己维

## IBM-PC/XT 软盘驱动器故障两例

林业部白城林业学校 赵明生

### 例 1. 故障现象：软盘驱动器主导电机转速过高而不能引导DOS

分析与维修：在开机加电后，主机自检通过，但在引导DOS时，屏幕显示：“Disk boot failure”信息，即“磁盘引导失败”。引起这类故障的原因有如下几种：①驱动器的磁头太脏；②选头电路故障；③步进（STEP）控制信号不正常；④主导电机转速不对。本例是由第④种故障引起的。

修理时，先取下磁盘驱动器，打开底盖即可看到在底部有一片测速盘，测速盘有两圈条纹，

在日光灯照射下，如果内圈条纹看上去不动时（在启动驱动器时观察），转速为50周/秒。这是正常情况。若外圈条纹不动其转速为60周/秒。而本机在观察时发现，其测速盘的内圈条纹是逆着测速盘的转动方向转动。这可证明转速超过正常转速。此时笔者调整调速电位器 $R_4$ （见图1），不起作用，说明不是因 $R_4$ 触点位置变动而使转速变动的。检查调整管 $Q_1$ （TIP110）的C和E极间没有击穿。经分析知：若图1中的LM2917N组件中的功放三极管开始（即LM2917N的5和8脚之间），会使 $Q_1$ 的基极电位升高，使 $Q_1$ 饱和导通，使 $U_c \approx V_c$ ，从而加在直流电动机上的电

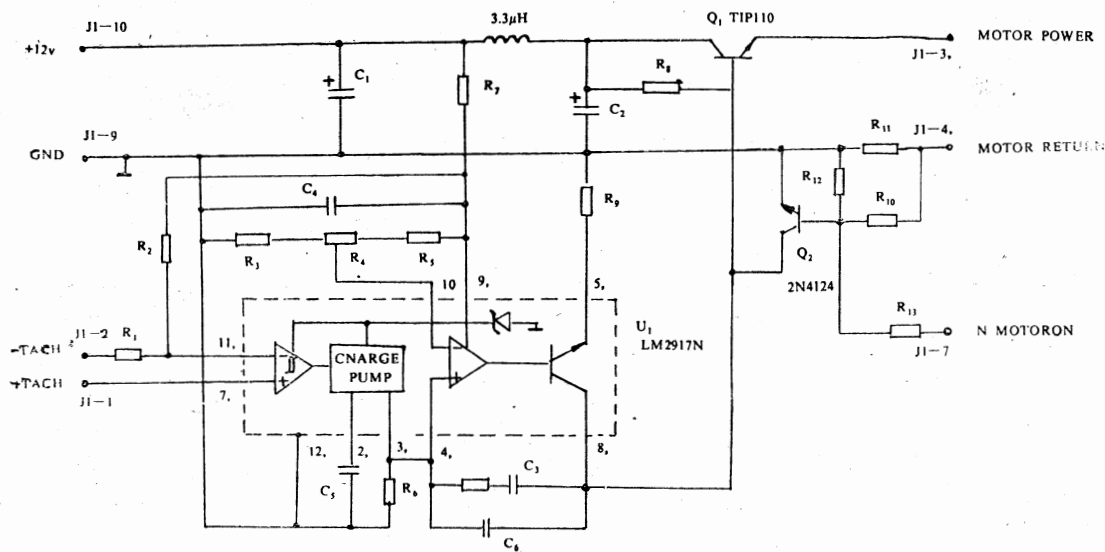


图1

修过的) 维修站不接收维修，或者在维修时加价。因此若故障较大的话，最好还是送DEC公司的VAX计算机维修站。

DHV 11接口板在带电拔下或插上终端接插件时，并非都会引起损坏。但是“不怕一万，只怕万一”，还是按照操作规程办事。

这里要说明一下，接口电路的损坏并非都是DUART的损坏，有时只是损坏线路收发器，也就是电平转换电路，EIA/TTL的电平从12V与3V之间的电平转换。这种情况下，更换元器件比较容易，因为这些元器件是常用的，在市场上很容易买到的。

# IBM-PC 微型计算机及兼容机故障检修

烟台师范学院计算机实验中心 王峰

在使用PC机时,我们遇到些故障现象,今我们将排除这些故障的分析方法介绍给广大读者。

## 一、主机硬盘不能引导系统,对硬盘执行FORMAT程序时,屏幕提示0道坏。

Track 0 bad-disk unusable-DISK UNAVAILABLE FOR SYSTEM DISK.

故障分析与检修:

我们机器用的是ST-412全高10兆硬盘。从现象上看是引导记录、文件分配表和目录驻留的0磁道损伤,所以从硬盘上无法启动系统。首先用LOWFORM对硬盘进行初始化,后用FDISK命令分区,再用FORMAT C:/S对硬盘重新系统格式化,把DOS装入硬盘,如果仍不能启动就认为是硬盘0磁道损伤。我单位机器经过上述操作后,DOS仍不能启动,我们用重新定义0磁道的办法来避开已损伤的0磁道。ST-412全高10兆硬盘采用开环定位,驱动硬盘磁

头移动的步进电机的步进相序是以8磁道一循环的,因此可把新的0磁道定义在原来的第8磁道。在系统未加电的状态下,先在步进电机轴和挡光板接触的部位做一精确的记号,目的是记住它们未调整前的相对位置,再用内六角扳手松开步进电机轴上0磁道检测器挡光板上的2个螺丝,只要用手指在轴上转动挡光板即可,不要过于松动。此时启动系统,在DEBUG下用一段小程序把磁头移到第8磁道。具体作法见程序清单1(见64页)。

程序中的DH=01表示1号磁头,DL=80表示硬盘驱动器,CH=08表示第8磁道,CL=01表示1扇区,AH=0C表示查找磁道,它们都是硬盘I/O功能调用INT13所需要的寄存器参数。此外,程序的MOV CX,0801中的08可以根据需要改成09、0A、0B、0C,这样就可把0磁道定义在第9、10、11、12磁道。

亦可以在DEBUG下用N命令给这个小程序

压超过正常值,导致电机转速增高。一般来说,出现电机转速不对时,若调整R<sub>4</sub>不起作用,多数是LM2917N组件损坏。但该组件价格较高,检修时应先对外围元件进行检查。特别是该组件的1和11脚是和测速发电机相接的,而测速发电机又与主导电机同轴,若该接线断路或测速发电机内部绕组开路,就不能反馈速度信号来控制主导电机的转速,使驱动器转速不准。这点应引起修理者的注意。

## 例2. 故障现象:机器工作时经常产生随机性读错误

分析与维修:在寻道等电路无故障条件下,产生随机性读错误,一般是由机械故障引起的。如驱动器主轴不同心或轴与孔因磨损出现间隙增大,使磁盘横向晃动,从而使计算机发生随机性读错误。

本例拆开磁盘驱动器后,用手轻晃主轴,能感觉到晃动,说明间隙过大。更换后故障消失。另外,检查主轴是否晃动没可采用测波形法。即:取一片已格式化的磁盘,在任意一磁道上用RDOS写上全“1”或全“0”码,之后,再步进到这一磁道上。用示波器看测试点TP5的波形时,若第三个波形晃动大于480ns时,表示主轴间隙过大或不同心。应更换。(见图2)

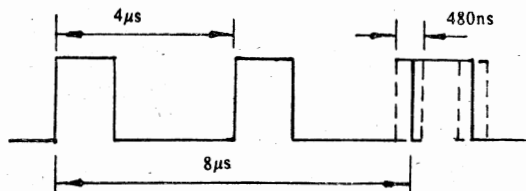


图2

起个名字如: WANG.COM, 再用W命令把这段程序记在盘上。以后要用这段程序时只要在DOS提示符下打入: WANG就行了, 用起来很方便。

此时用三用表观察0道开关的输出信号(P8—4)的电压幅道, 同时很细致地顺时针转动0磁道检测器挡光板, 当信号幅值在2.25V左右时, 再用内六角扳手紧固挡光板, 操作时注意不要转动步进电机轴。然后启动系统, 重新对硬盘进行系统格式化并装入DOS, 这样硬盘就可以重新使用了。

## 二、“Ctrl+p”时不打印, 屏幕出现:

No paper error. Writing device PRN  
Abort, Retry, Ignore?

故障分析与维修:

已确定打印机联机完毕且打印纸在机上, 由现象可分析产生故障的原因是由于打印适配器第15脚ERROR信号变为低电平所致(正常时应为高电平)。这可能是联接电缆有问题, 检查联接电缆, 发现接在打印适配器端第15脚的连线, 在打印机端没有接在第32脚上(fault)呈悬空状态, 但不是断了。后将该连线改接到32脚上, 故障排除。

分析其原因, 该联接电缆当一端接在打印适配器第15脚, 而另一端悬空的状态下, 由于受外界杂散信号干扰, 使适配器误发ERROR信号, 造成故障。

进一步实验, 将打印适配器第15脚连线焊下, 开机后键入Ctrl+P, 打印时又出现原故障。此时将与打印适配器第15脚的有关芯片74LS125芯片的性能有关, 我们用一新芯片替换原来的旧芯片后, 故障排除, 打印又恢复正常, 从而说明原74LS125芯片的性能已降低。

从中我们可以认为, 打印适配器与打印机的联接电缆, 除了那些必须要联接的外, 打印适配器一端的第15脚与打印机一端的第32脚必须要联接, 这样即使有时芯片性能稍差些, 亦能正常使用。

## 三、打开机器电源后, 自检正常, 惟不读软盘。

故障分析与检修:

开机自检正常, 可见软盘驱动器适配卡主要部分是好的。但因为没有从软盘中读出数据, 由此可判定很可能是适配卡上的数据通路部分有问题, 应重点检查此部分。

在软盘驱动器中插入DOS盘, 使机器处于读盘状态(不断作热启动即可)。在此时用示波器观测适配卡上的数据通道, 即 $u_1$ (74LS240)的读数据输入端第8脚, 输出端第12脚, 均有波形。再观测数据分离、解码电路即 $u_{22}$ (74LS112),  $u_{23}$ (74LS161),  $u_{25}$ (74LS112)及 $u_{26}$ (74LS02)的有关输入输出, 在读盘时均有波形输出, 似乎并无芯片损坏。进一步只能怀疑是否是数字锁相环路出了问题, 使检读数据窗口的频率发生变化, 致使数据分离出错以致导致读数据错。用示波器观察 $u_{19}$ (74LS191)的 $Q_0$ 输出(第7脚), 即数据窗口(DATAW INDOW)波形, 发现其频率偏低(正常的频率为250kHz—写时钟(WRITE CLOCK)频率500kHz的一半)。因此, 可断定数字锁相环路有故障。用万用表测相位检波器 $u_{21}$ (MC4044)第8脚的输出电压仅2伏左右(正常时为4.1V左右)。故使压控振荡器 $u_2$ (MC4024)第6脚输出的频率变低(正常时应为4MHz)。至此, 故障范围缩小到相位检波器 $u_{21}$ (MC4044), 输入选择器 $u_{24}$ (74LS153)以及500kHz写时钟的频率是否正常这几个疑点上。用示波器观测500kHz写时钟频率基本正常。但用示波器观测四中选一输入选择器 $u_{24}$ (74LS153)的输入输出端时, 发现其第一组的输入端1Y(第7脚)无波形, 始终为“1”, 尽管此时其相应的4个输入端1C0—1C3均有波形输入。由此可判定此片74LS153损坏。正因为1Y端无波形输出, 使相压比较器 $u_{21}$ 的比较信号输入端(第1脚)亦无波形(即比较信号的频率为0), 因而反馈信号(由 $u_{21}$ 第3脚输入)的频率大大高于比较信号, 使相位比较器输出电压变低, “企图”降低压控振荡器的振荡频率, 从而使反馈信号(由压控振荡器的输出经八分频

后,从 $u_{10}$ 第6脚输出,并经 $u_{24}$ 的第二组通路反馈到 $u_{21}$ 的第3脚)的频率向比较信号的频率靠拢。当然,由于相位比较器和压控振荡器的工作特性范围有限,实际上不可能把反馈信号的频率亦降低为0。更换 $u_{24}$ (74LS153)芯片后,故障排除。

#### 程序清单1:

A>DEBUG

-AL00

0BF3:0100 MOV DX,0100

0BF3:0103 MOV CX,0001

0BF3:0106 MOV AH,0C

0BF3:0108 INT 13

0BF3:010A

-N WANG.COM

-R CX

CX 0000

:000F

-W

WRITING 0000F BYTES

-Q

A>WANG

## 为汽车制造提供现代测试手段 微电脑汽车性能综合测试仪问世

一种专门用于鉴别汽车质量和性能的微电脑汽车性能综合测试仪,已由上海计算所和中国第二汽车制造厂联合研制成功。据二汽实际使用后认为,该测试仪已达到日本、西德同类产品水平,且功能更全,将会改变我国已往长期依赖进口此类仪器的局面。

该测试仪操作简捷,结构紧凑,功能齐全,小型价廉,携带方便。它的研制成功,为我国的汽车质量和性能评定提供了高精度的综合测试仪器,对我国STD总线技术的进一步开发利用,也将起到推动作用。

(吕宜男)

## 推进我国工业岔管的分析设计现代化 岔管有限之分析和CAD软件系统研制成功

岔管有限之分析和CAD软件系统,系上海市科学技术发展基金项目,已由上海计算所副研究员华伯浩等自行研制成功。9月25日在沪通过技术鉴定。专家确认,该软件是国内第一个自行研制在河波罗工作站上先进的岔管自动分析和设计系统,数据准备自动化程度很高,分析结果准确,制图精密,达到了近年来引进国际同类软件的先进水平,其中,系统的有限之分析模块的功

能和效率在某些方面已超过了当前美国著名的SAP系列的各个版本。

岔管有限之分析和CAD软件系统的研制成功,对改进岔管、管接头和离心机转子的结构设计,缩短设计用期,提高产品质量,具有重要意义,对形成我国在这些行业中的设计制造体系也将起到积极的促进作用。

(吕宜男)

## 为水工建筑设计提供决策依据 填筑土石坝空间非线性有限之应力分析通用软件面世

采用土石材料代替混凝土建筑水电站的拦水坝,是当代世界上水电工程建设中的一项高新技术,

对于开发我国丰富的水电资源具有极大的经济价值。为此,国家科技攻关中安排了“高土石坝

筑坝关键技术问题的研究”项目,而填筑土石坝空间非线性有限之应力分析通用软件是其中的一个子项目,已由上海计算所、成都科技大学和水电部昆明水电勘测设计院研制成功,并通过了部级鉴定,最近还荣获1988年上海市科技进步二等奖。

该项成果已在我国第一个通过世界开发银行贷款、施工管理公开向国外招标的贵州省鲁布革

水电站的建设中得到了应用。由于对该电站的拦水高土石坝心墙采用就地的风化料作了准确的计算分析论证,从而为工程节省投资300万元作出了贡献。今年9月,这个电功率为80万千瓦的水电站已投入商业运行。对水工建筑设计采用这种三维有限之非线性分析程序的,这在国内尚属首次。

(吕宜男)

## 国内首次采用条形码识别技术的计算机会务 签到应用系统获上海市人代会和人大常委会好评

一项能够实时传送会议代表签到信息的计算机会务签到应用系统,日前在沪通过由上海科学院主持的验收鉴定。该系统是由上海计算所、上海市煤气公司和上海市人大常委会办公厅联合研制成功的。

专家们认为,这项成果在实用化、产品化方面是相当成功的,具有国内先进水平,特别是系统中应用条形码识别新技术,这在国内尚属首次,建议有关部门尽早组织推广应用。

(吕宜男)

## 电脑等精度通用计数器研制成功大批投产

国家科技攻关项目——EE3366型电脑等精度通用计数器由南京电讯仪器厂研制成功,日前通过部级鉴定大批投产,填补了国内一项空白。

这种计数器目前国外只有美国有。美国的产品型号是HP5345,被美国列为禁运品。这种仪器是研究、开发雷达、导航、遥感技术的重要设备、是国防、科研和文教领域迫切需要的仪器。它配有微电脑,功能多、精度高、速度快、能自动测量窄脉冲调制微波信号多种参数,并能自我检验测量数据是否正确。

该仪器的主要特点是:用程序控制进行倒数频率测量,保证了测量精度在同闸门时间内是恒等的,而与测量频率无关,实现了等精度测量;采用高灵敏度,超宽带DC—500MHz放大器,给频率与脉冲提供了具有现代水平的前置信号处理能力。该仪器主要由微电脑、GB—IB标准接口及测量、键盘显示等部分组成,其中测量部分由A、B两个DC—500MHz的放大整形电路、超高速主门电路、事件计数器、时间计数器、及控制电路、时基钟电路组成;键盘显示部分由功能、数字、副功能等27个按键组成,另外还有11位LED数字显示以及必要的译码、扫描、锁存等电路;电脑部分由MC6800、ROM、RAM以及译码器组成。遥控信号控制通GB—IB标准接口实现。在电脑控制下,具有测频、测周期、测时间间隔、测频率比以及累计,自校、自诊断等七项主要功能。另外,还有为多种转换而设置的数据显示移动和预置偏移量两种副功能。为了满足对直流及射频脉冲调制信号测量的需要,还设置了从单次、100μs至1000s的口档闸门。为使用户操作方便,按键分为主、副、数据三种类型,按类分块,可以迅速找到所需按键。同时,考虑到保持面板整洁美观,采用一键多义的方式,即一个按键可以作多种用途。

(李相彬)

## 计算机应用研究 第7卷 第2期 (总第34期)

## 目 录

## 软 件 篇

代数多项式的Horner—差商混合算法.....	徐继锋 ( 1 )
数据库设计最小复盖法.....	瓮正科 ( 2 )
一个不易破解的文件名加密法.....	瞿 波 ( 5 )
窗口软件的编写.....	刘宇菁 ( 6 )
一个用户友好的文件装载窗口.....	唐常杰 ( 7 )
自动生成dBASE Ⅲ菜单源程序的方法.....	周步祥 ( 12 )
汉字下拉式菜单.....	刘宇菁 ( 15 )
用 BASIC 语言实现全屏数字编辑.....	段宝珠 ( 16 )
屏幕划到与翻页技术.....	王 勃 ( 18 )
交互式任意大小屏幕图形的存贮与再现.....	李 晖 ( 20 )
大型汉字配上倒影的初步实验.....	夏国华 ( 23 )
最优箭线网络图的绘制算法.....	周 明 ( 25 )
并行处理语言OCCam及其在图象处理中的应用.....	诸昌铃 ( 26 )
Turbo BASIC 语言与汇编程序的连接方法.....	辛 华 李谊瑞 ( 32 )
快速高效编制报表的简易方法.....	方荣耀 ( 33 )
MIGHTYPRESS 打印接口卡特殊用法.....	唐汉雄 ( 36 )
居民身份证底卡的微机打印程序.....	苏亚华 ( 40 )

## 硬 件 篇

一个实用的微机共享存贮器通讯接口.....	刘宏生 ( 42 )
气功信息模拟方法.....	奥居雄等 ( 43 )
体感诱发电位检测系统的设计与应用.....	彭利安等 ( 46 )
Z80微机系统中大量扩展 I/O 接口和存贮器的巧妙方法.....	方旭明 ( 50 )

## 系 统 篇

六十五吨玻璃窑炉微机控制系统.....	周立恒 刘晓光 ( 52 )
多级系统中IBM-PC/XT和STD总线通讯.....	周树杰等 ( 57 )

## 维 修 篇

VAX计算机接口板 DHV11 的故障分析.....	高济平 蔡新光 ( 59 )
IBM-PC/XT 软盘驱动器故障两例.....	赵明生 ( 61 )
IBM-PC微型计算机及兼容机故障检修.....	王 锋 ( 62 )

## 信 息 篇

消息十则.....	(封二, 13, 39, 64, 65)
-----------	----------------------



## APPLICATION RESEARCH OF COMPUTERS

Vol.7 No.2 (Total 34)

## CONTENTS

## SOFTWARE

- The hybrid Algorithm with Horner-Difference Coefficient of  
Algebraical Multinomial ..... Xu Jifeng ( 1 )
- The Minimal Overlay Method for Database Design.....WengZhengke ( 2 )
- A Method of Filename EncryPtion which Isn't Easy to Disencrypt  
..... Qu Bo ( 5 )
- Writing the Aperture Software ..... Liu Yujing ( 6 )
- A User Aperture of Loading File in Friendship ..... Tang Changjie ( 7 )
- The Method for Automatic Producting the Menu Souce Program of  
dBASE II..... Zhou Buxiang ( 12 )
- The Push-down Menu with Chinese Characters..... Liu Yujing ( 15 )
- CompiLing Numeral with Full Screen by BASIC Language  
..... Duan Baozhu ( 16 )
- Partiting Screen and Overturning Page ..... Wang Bo ( 18 )
- Storing and Restoring for Interactive Screen Graphic with Any Size  
..... Li Hui ( 20 )
- The First Test about Large-scale Chinese Characters and Their  
Shadow.....Xia Guohua ( 23 )
- The Algorithm of Drawing Optimum Network Graphic with Arrow  
Lines.....Zhou Ming ( 25 )
- The Language OCCAM with Concurrent Processing and It's Applying  
in Image Processing.....Zhu Changqian ( 26 )
- The Linking Method between Turbo BASIC Language and Assembler  
Program..... Xin Hua and Others ( 32 )
- The Simply Method of High Speed Writing the Report Table  
.....Fang Rongyao ( 33 )
- The Specical Method for Printing Interface of MIGHTYPRESS  
..... Tang Hanxiong ( 36 )
- The Printing Program for Resident Identity Card on Microcomputer  
..... Su Yahua ( 40 )

## HARDWARE

- A Practical Communication Interface for Sharing Storage on  
Microcomputer..... Lin Hongsheng ( 42 )
- The Analog Method of QIGONG Informations ..... Xi Juxiong ( 43 )
- Designing and Applying for Detection System of Induced Potential by  
Body Sense.....Peng Lian ( 46 )
- The Ingenious Method of Expanding I/O Interface and Storage on Z80  
Microcomputer System..... Fang Xuming ( 50 )

## SYSTEM

- The Controlled System for Sixty-five Ton Glasskiln by Microcomputer  
..... Yan Liheng ( 52 )
- The Bus Communication between IBM-PC/XT and STD in Multilevel  
System..... Zhou Shujie ( 57 )

## MAINTENANCE

- Defect-analysis of DHV11 Interface Board in VAX Computer  
..... Gao Jiping ( 59 )
- Two Defect-exampl for Floppy Disk Driver on IBM-PC/XT  
..... Zhao Mingsheng ( 61 )
- IBM-PC/XT and It's Compatible Breakdown Overhauled  
..... Wang Feng ( 62 )

## INFORMATION

- Ten Pieces of News ..... (The Cover2, 13, 16, 39, 65)

## APPLICATION RESEARCH OF COMPUTERS

(Publicly Publishing) (Bimonthly)

March 1990 Volume 7 Number 2 (Total 34)

Main Editor: Zhang Zhi Qian

Assistant Main Editor: Li Zemin

Responsibility Editor for this Issue: Qi Mo Zhi

Editor/Publfsher: Editorial Department of This Periodical

Address: 11—1 Four Section Ren Ming Nan Lu, Chengdu, P.R.C

Press: South west metallurgical geology Printing house of

Ministry of Metallurgical Industry

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

Central Dispatching in Inside the Country: The Post of Chengdu

Unified Number in Inside the Country: CN 51—1196

Number of the Post Publishing: 62—68

Post Code: 610015