

ISSN 1001-3893

# 计算机应用研究

1991

3

APPLICATION RESEARCH OF COMPUTERS 《计算机应用研究》雜誌





### 《計算機應用研究》雜誌辦刊單位

四川省電子計算機應用研究中心 新疆電子計算中心  
貴州省科委計算中心 甘肅省計算中心  
安徽省計算中心 廣西計算中心  
吉林省計算中心 青海省測試計算中心  
河南省計算中心 四川省電子學會

### 《計算機應用研究》雜誌社董事會

董事長：周賈渝

董事：唐 珍 鄭國基 陸慰椿  
秦小竹 龔宇清 黎瑰常

### 《計算機應用研究》雜誌編輯委員會

主任委員：張執謙

副主任委員：李澤民

委員：賈洪鈞 曾光初 龔宇清  
張國棟 羅海鵬 劉鐵軍  
崔振遠 楊劍波 余 凱

1991年第3期（總第41期）

出版日期：1991年5月 本期責任編輯：鄧嘉澍

計算機應用研究（雙月刊）  
（公開發行）

國內統一刊號：CN51—1196

主 編：張執謙

副主編：李澤民

編輯出版：《計算機應用研究》雜誌社

通訊地址：成都市人民南路4段11號附1號

郵政編碼：610015

印 刷：新都一中印刷廠

訂 閱 處：全國各地郵局

總 發 行：成都市郵政局

郵發代號：62—68



## 河南省计算中心简介

河南省计算中心成立于 1983 年,是在河南省科委的直接领导下,面向全省各行业,从事计算机应用、软件开发和应用数学研究的全民制科研单位。

该中心共有职工 166 人,专业技术人员 129 人,其中高级工程师、付研究员 10 人,工程师 43 人。该中心拥有一座 6100 平方米的科研实验大楼,配有从日本引进的具有汉字系统的 M—240D 大型通用计算机一套,高、中档微型机 59 台(套)及多种科研设备,该中心能为用户开展各种类型的计算机应用提供一流的工作环境及条件设备。

“七五”期间,该中心在计算机过程控制、管理信息系统、智能化仪器、计算机辅助设计、网络技术的应用、汉字处理、应用数学及软科学等方面进行了大量卓有成效的研究工作,共完成百余项科研项目,其中获得省科技进步二等奖 3 项,三等奖 5 项,获得国家科技进步三等奖 1 项。

该中心的新技术实验工厂是从事开发研制高技术产品的中试基地,并最早向全国推出了自行设计的 BD 和 BDJ(0.1 级、0.2 级)系列标准功率电能表。该中心的电脑公司,利用雄厚的技术力量,能为用户提供最佳的售后维修服务。目前,该公司已发展成为本省技术力量最强的计算机专业公司。

“八五”期间,该中心将集中优势力量,重点开展以下业务:

- 1、M—240 大型机算题服务、项目开发服务及微型机机时服务。
- 2、提供数值计算、数据处理(包括汉字录入)、计算机激光排版等服务。
- 3、承接国民经济各行各业微机应用项目。
- 4、研制开发高科技智能化、自动化仪器仪表。
- 5、承接电子技术应用项目的开发及设备研制。
- 6、研制开发并转让各类计算机系统软件及应用软件。
- 7、承接科技、经济发展中的实际数学问题,并重点开展软决策、系统工程、运筹学、计算数学等应用研究。
- 8、经营计算机设备及其零配件,承接计算机维修和机房设计装修等业务。
- 9、开展有关电子、电脑技术,应用数学方面的信息交流及其学术活动。
- 10、开展有关电子、电脑的科、工、贸方面的国内外技术合作。

通讯地址:郑州市花园路 27 号

邮政编码:450003

电 话:555801

### 《计算机应用研究》杂志社董事会

董事长:周赛渝

董 事:唐 珍 孙传江 陆慰椿

吴地兴 郑国基 陈建岭

黎 蓉 黎瑰常

董事会名单以此为准

表 1

A19	A18	A17	A16	Q0
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$\overline{\text{CAS3}}$ 由  $U_{42}$ (74LS138)的输出端  $Y_3$  来,测量  $U_{42}$ 的各输出端,发现第 4 脚( $\overline{\text{G2A}}$ )当  $\overline{\text{XMEMW}}$ 有效时为低电平(有效),不正常。该脚的信号是由  $U_{44}$ (24S10)的  $Q_0$  输出通过  $U_{41}$ (74LS00)送来的。测  $U_{44}$ 的  $Q_0$  为高电平,再检查  $U_{44}$ 的输入信号,  $A_3 \sim A_0$  为地址总线高四位,正常。 $A_4 \sim A_7$  由系统板上的开关和跳线插

头控制,均正常。修改并运行上面的程序,发现当  $A_{19} \sim A_{16}$  状态分别为 A、B 时,  $U_{44}$  的  $Q_0$  输出高电平,  $A_{19} \sim A_{16}$  状态分别为 C、D、E、F 时,  $Q_0$  输出为低电平。 $Q_1$ 、 $Q_2$  正常。 $U_{44}$  是 PROM, 当且仅当访问系统板上的 RAM 时,  $Q_0$  端应输出高电平。根据上述情况看,  $U_{44}$  的  $Q_0$  输出不正常,不过输入与输出端都没有损坏,可能是内部与  $Q_0$  有关的逻辑或存贮单元损坏。这种 PROM 无法配到,即便能配到也没有写入数据的编程器,所以无法更换。考虑到系统板上的开关(SW)和跳线插头的位置一般不会再改变,这样  $Q_0$  仅与  $A_3 \sim A_0$  (对应地址总线高四位)有关,可以用外加组合逻辑电路来取代 24S10 的部分功能( $Q_0$ )。根据对地址译码的要求,列出真值表如表 1 所示。从表 1 可以求得:

$$Q_0 = \overline{A_{19}}(\overline{A_{18}} + A_{17}) = \overline{A_{19}}A_{18}A_{17}$$

只要用一片四与非门即可实现上式的逻辑功能。选用 74LS00, 按图 3 连接好(74LS00 迭在 24S10 上面), 机器即恢复正常。

#### 结束语:

1 在这个例子中,由于 RAM 的地址译码电路损坏,使得 CPU 在访问显示缓存区的同时,也访问了 RAM 中特定的区域,从而造成这种较特殊的存贮器故障。

2 因为故障芯片不易更换,不得已采用外加组合逻辑电路的方法,这时应考虑附加的延时。此例中三级门共延迟约 30ns,实践证明没有影响。(图在 46 页)

## 介绍一例人民币金额小写变大写的实用小程序

应读者要求,此处推荐一例实用小程序。该程序用 dBASE II 编写,仅 40 余行,简短明了,效率很高(转换时间 < 1 秒)。程序清单如下:

```
dx="壹贰叁肆伍陆柒捌玖"
dy='元拾佰仟万拾佰仟亿拾'
set talk off
store 0.00 to s,flg
@10,20 say "请输入金额" get s pict "9999999999.99"
read
sr=""
zs=int(s)
dm=s-zs
store 1 to fl, i
```

```
do while zs >= 1
  s=int(zs/10)
  t=zs-10*s
  zs=s
  if t=0
    fl=0
  if i=1
    sr='元'
  endi
  if i=9 .or. (i=5 .and. s<1000)
    sr=substr(dy,i+1,2)+sr
  else
    if flg=1
      sr="零"+sr
    (下转 56 页)
```

## 《计算机应用研究》第8卷第三期(总第41期)

### 目 录

#### 软件篇

高速脱壳穿甲弹脱壳过程的计算机处理 .....	侯晚霞	崔东明(1)
实用代码字典的查找和对照显示技术 .....	张孟春	王国红(4)
在 XENIX 中连接外部设备的秘诀 .....	石文昌	(8)
微机对未来进行预测的方法 .....	张鸿鸣	李志英(10)
MCS-51 单片机开发工具软件 .....	林化	(13)
BASIC 语言动画技术应用实例 .....	黄焕如	(13)
DBMS 数据库汉字排序技术探讨 .....	张小朋	董 敏(18)
在单色汉字方式下的绘图软件 .....	吴正明	(20)
破译加密程序实例 .....	茹建平	(21)
如何灵活应用设置检索目录 PATH 命令 .....	张伟中	(23)
用 SQL * REPORT 设计统计报表 .....	曾明蓉	曾文方(24)
一种给 IBM-PC/XT 机硬盘加锁的实用方法 .....	李铭	(31)
一种屏幕上图块自动替换的有效方法 .....	鲍献华	(32)
印刷五线谱乐符的识别方法 .....	吕良	(26)
微机通用信息管理系统 Micro CDS/ISIS 的应用技巧 .....	万发仁	张晓东(29)
多路数据采集内存使用及存盘法 .....	任清珍	黄天成(33)
介绍一例人民币金额小写变大写的实用小程序 .....	叶志斌	(63)

#### 硬件篇

利用 GAL 设计的一种编码键盘接口 .....	赵英俊	胡 阳(35)
通用阵列逻辑 GAL 器件的分析研究 .....	朱焱	(38)
适用于计算机系统的硬时钟 .....	舒彦	(41)
一种利用资源共享技术的单片机开发装置 .....	荆予今	(43)
单片机与 A/D 转换器的简易连接 .....	孙劲夫	(47)
数字信号处理器与高速 A/D 卡接口技术 .....	庞文宁	朱科军(48)

#### 系统篇

MCS-51 系列单片机在数字逻辑分析与设计中的应用 .....	汪亚南	秦正南	黄丽莹(49)
----------------------------------	-----	-----	---------

#### 维修篇

排除 COMPAQ/20E 一故障之所得 .....	戴若兰	薛和华(57)
IBM-PC/XT 微机存储器故障排除一例 .....	周凡	包锦雄(61)

#### 信息篇

简讯 .....	(封2、28)
----------	---------

# APPLICATION RESEARCH OF COMPUTERS

VOL. 8 NO. 3 (Total 41)

## CONTENTS

### SOFTWARE

The Computer Process to Sabot Discard Process of the High Speed Armour—Piercing Sabot Discard Projectile .....	Hou Xiaoxia Chui Dingming (1)
Searching for Utility code Directory and Corresponding Display .....	Zhang Mengchun Wang Guohong (4)
The Secret of Setting up Peripheral Devices in XENIX .....	Shi Wenchang (8)
The Method of predicting Future by Microcomputer .....	Zhang Hongming Li Zhiying (10)
Developing Tool Software on MCS—51 Single—Chip Computer .....	Lin Hua (13)
Examples of Applying BASIC Language in Animated Cartoons .....	Huang Huanru (13)
Exploring the Technics of chinese characters sort for DBMS Data Base .....	Zhang Xiaopeng Dong Min (18)
A Plotting Software in Monochromatic Chinese Characters Modes .....	Wu Zhengming (26)
Examples of Decrypting Encrypted Program .....	Ru Jianping (21)
How to use the command of the Set Search Directory PATH Flexibly .....	Zhang Wei Zhong (23)
Designing Statistical Journal Sheet by Using SQL * REPORT .....	Zeng Mingrong Zeng Wenfang (24)
An Actual Method of Looking Hard Disk of IBM—PC/XT .....	Li Ming (31)
An Effective Method of Automatic Replacing Graphic Block on Screen .....	Bao Xianhua (32)
The Method of Recognizing the Printed Staff Note .....	Lu Liang (26)
The Application Skills of Microcomputer General Information Management System—Micro CDS/ISIS .....	Wan Faren Zhang Xiaodong (29)
Effective Memory Mapping and Saving in Multichannel Data Acquisition .....	Ren QingZhen Guang Tiaowu (33)
Recommend to an Example of Actual Program Exchanging Renminbis Quantity Lowercase for Uppercase .....	Ye Zhibin (63)

### HARDWARE

Designing Encoded Keyboard Interface With GAL .....	Zhao Yingjun Hu Yang (35)
Analysing and Researching into General Array Logic Device GAL .....	Zhu YAN (38)
The Hard clock Used in computer system .....	Su Yan (41)
A Developing Device of Single—Chip computer Used Resource Sharing Technique .....	Jing YiJin (43)
A Simple Interface between Single Chip Processor and A/D Converter .....	Sun Jingfu (47)
The Interface Technique between Digital Signal Processor and High—Speed A/D Card. ....	Pang Wenlin Zhu Kejun (48)

### SYSTEM

Applying MCS—51 Family Single Chip Computer in Mathematical Logic Analyses and Design .....	Wang Yanan Qin Zhengnan Huang Liyun (49)
---	--

### MAINTENANCE

Removing the Fault of COMPAQ/20E .....	Dai Ruolan Xue Huohua (57)
An Example for Debugging the Memory Defective in IBM—PC/XT .....	Zhou Fan Bao Jinxiong (61)

### INFORMATION

News .....	(Cover2, 28)
------------	--------------



# 高速脱壳穿甲弹脱壳过程的计算机处理

南京师范大学 侯晓霞 华东工学院 崔东明

**摘要** 本文给出了一个高速脱壳穿甲弹脱壳过程的计算机处理系统。利用该系统,能够重视高速脱壳穿甲弹的空间飞行姿态,并能对其整个脱壳过程进行动态模拟;同时,能定量给出弹体及弹托的空间位置。将计算机图形学技术应用于该领域在我国尚属首次。多次实验证明:其效果是很满意的。

## 一、引言

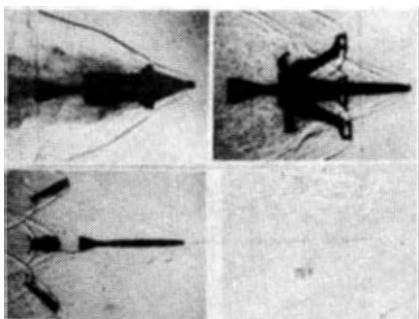
高速脱壳穿甲弹以其优良的穿甲性能成为目前反坦克武器的主要弹种之一,几乎所有的坦克炮和反坦克炮都装备有高速脱壳穿甲弹。高速脱壳穿甲弹是一种特殊类型的弹种,它由杆型弹体和马鞍型弹托两大部分组成,其不同于一般炮弹之处是脱壳穿甲弹出炮口后,弹托和弹体即开始分离,弹托逐渐脱离弹体而落下,弹体击中目标,完成战斗作用。由于脱壳穿甲弹特有的结构特征,其在发射后,特别是在近炮口区弹托的脱落过程中,会对弹体产生干扰影响弹体的命中精度,降低脱壳穿甲弹的战术技术指标。因此,高速脱壳穿甲弹的脱壳过程受到了世界各国军方的普遍重视,成为高速脱壳穿甲弹研究的一个重要领域。我国近几年在这方面的研究刚刚起步。

以往研究的方法均是由高速摄影拍摄出脱壳穿甲弹飞行过程中的姿态照片,通过对这些照片(绝大多数是阴影照片)的观察与分析,提出对其结构性能的改进意见,以及对其构造机理的进一步研究。但是,由于高速脱壳穿甲弹的飞行速度很快( $1300\text{m}\sim 1800\text{m/s}$ )再加上其拍摄环境非常恶劣(如近炮口区炮口焰的影响,中间各站气流与冲击波的干扰等),使得拍出的照片往往背景噪声很大,给观察与分析带来困难,尤其是近炮口区,弹体几乎被气流淹没,往往无法判断其空间位置与飞行姿态。(原始照片见图1)。再加上脱

壳穿甲弹脱壳过程本身是一个动态连续的过程,而测试站只能提供有限的静止的侧面的照片,使得无法实现对脱壳穿甲弹脱壳过程的全面观察,这无疑也影响到整个研究的水平。因此,如何全面、动态、连续地再现高速脱壳穿甲弹的脱壳过程,已成为这个领域急待解决的关键问题。有资料[1]报道,美国弹道研究所采用简化弹托结构,从照片上取个别特征点,通过手工绘图,给出各测试站的空间飞行姿态的简化草图及正面观察图,以弥补上述的不足,但对动态连续显示及定量计算等问题仍无法解决。为此,我们研制了高速脱壳穿甲弹脱壳过程的计算机处理系统。该系统能根据所拍摄的照片对高速脱壳穿甲弹整个脱壳过程进行各种处理,不仅能提供各摄影站的瞬间空间飞行姿态,还能对整个脱壳过程进行动画显示,并且能根据需要提供对脱壳穿甲弹脱壳过程中任一时刻,任一角度的观察,同时,提供有关数据的定量计算。利用该处理系统从动态连续的角度提供了对脱壳穿甲弹的全面观察研究,数据的定量计算还给设计者提供了科学依据。它从多方面弥补了以前的空白,取得了满意的效果。

## 二、脱壳穿甲弹飞行姿态的显示

由弹道靶道给出的正交多站阴影照片描述的是高速脱壳穿甲弹的脱壳过程。该过程是一个动态过程,弹托的分离是一种无规则的动态分离。由初始的三个(或四个)弹托紧紧包住弹体到逐渐张开,直至全部分离,只有12张照片(共有6个站,每个站有2张正交的阴影照片)给出的6个瞬间位置。而在每个位置上,弹体偏离中心多少,每一弹托的空间位置如何,几个弹托的分离是否平衡,它们与弹体之间的相对位置等都是我们感兴趣的。为了能准确地描述这些现象,在绘制该物体的三维立体图时,我们将弹体与每一个弹托都视为独立图形,使它们能分别进行各自的三维变换,以对应其不平衡性。(绘图坐标系的建立如图2所示。)经过分析,将这些变换分解成绕三个坐标轴的旋转与沿三个坐标轴的平移[2]。其各自的变换矩阵如下页:



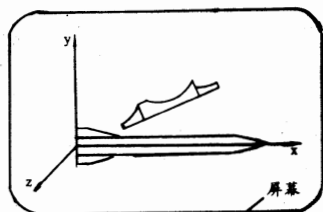


图2 坐标系的建立

图2 坐标系的建立

(1) 平移变换, 设新的中心点的坐标为  $(0_1, 0_2, 0_3)$ :

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0_1 & 0_2 & 0_3 & 1 \end{bmatrix}$$

(2) 绕 Z 轴旋转  $\theta$ : ( $\theta$  为正, 即逆时针旋转, 下同)

$$T_2 = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(3) 绕 X 轴旋转  $\alpha$  ( $\alpha$  可正可负)

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(4) 绕 Y 轴旋转  $\beta$ : ( $\beta$  可正可负)

$$T_4 = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

实际计算时, 以一个弹托为基本图形, 分别施以不同的  $T_1, T_2, T_3, T_4$  变换, 以对应于三个(或四个)不同的弹托位置。设该基本图形上的任一点  $(X, Y, Z)$ , 经变换后成为  $(X', Y', Z')$ , 即:  $(X', Y', Z') = (X, Y, Z, 1) \times T_1 \times T_2 \times T_3 \times T_4$ 。对经过上述变换后的坐标  $(X', Y', Z')$  再进行透视投影变换[3], 得到绘图坐标  $(X'', Y'')$ :

$$\begin{cases} x'' = -x' \sin\theta + y' \cos\theta \\ y'' = -x' \cos\theta \cos\varphi - y' \sin\theta \cos\varphi + Z' \sin\varphi \\ z'' = -x' \sin\theta \sin\varphi - y' \cos\theta \sin\varphi - Z' \cos\varphi + \rho \end{cases}$$

$$\begin{cases} x'' = d \times x_c / z_c \\ y'' = d \times y_c / z_c \end{cases}$$

其中  $(\rho, \theta, \varphi)$  为观察点的球面坐标,  $d$  为观察点到投影面的距离。

由于对每一弹托实现了全方位的坐标变换(即绕

三个轴的旋转与平移), 因此, 依据测试与识别系统提供的变换参数绘制出的脱壳穿甲弹飞行姿态的三维立体图收到了很好的效果。

另外, 我们设计了单站、多站对照、多站连续, 单弹托连续, 改变观察角等不同的显示方式, 充分利用图形变换及屏幕管理功能提供从不同方向、不同侧面, 不同角度对脱壳穿甲弹脱壳过程的观察。该观察方法明显优于文献[1]中美国弹研究所的结果。显示的部分结果见图3、图4、图5、图6所示。

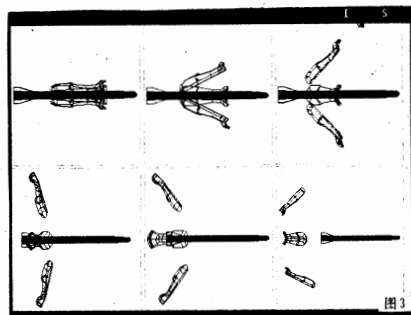


图3、多站对照(侧面)

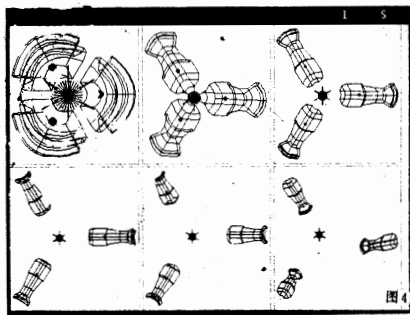


图4 多站对照(正面)

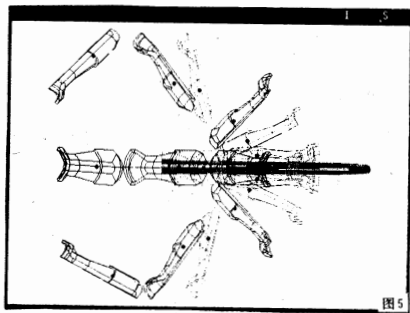


图5、多站连续



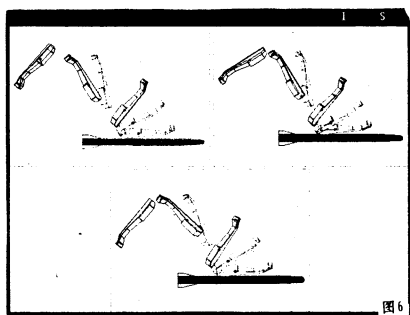


图6 单弹托连续

### 三、脱壳穿甲弹脱壳过程的动画显示

如前所述,由弹道靶道给出的12幅正交阴影照片,只描述了高速脱壳穿甲弹脱壳过程的六个瞬间位置,而脱壳穿甲弹的整个脱壳过程则是连续的动态的。为了能更好地反映出高速脱壳穿甲弹的整个脱壳过程,我们采用了计算机动画技术。

为了使动画效果逼真,首先我们采用加密显示站的方法,即在原有的六个站的基础上,通过插值的方法增加显示位置,使显示站点由原来的六个增至27个。其次是提高显示速度。由于该绘图程序中需用多种变换(平移、旋转、透视投影),因此占用计算机的时间较多,如果采用一边计算一边显示的方法,可能会因为显示速度太慢而失去其动画效果。为了提高显示速度,我们先把所有显示点的坐标(即27个显示位置的所有点坐标)全部计算出来,将其分别存放在不同的数组中,然后一次性地输出显示。这样去除了计算时间,得到了较好的动画效果。

脱壳穿甲弹脱壳过程的动画显示,弥补了原拍摄照片的静止性和间断性,从连续,动态的角度全面反映了整个脱壳过程,给观察和研究脱壳穿甲弹的飞行姿态提供了方便。利用此技术也可进行脱壳穿甲弹飞行姿态的模拟,给设计者提供依据。

### 四、弹托空间位置及弹体动规律的计算

高速脱壳穿甲弹弹托的飞散过程是一种无规则的运动,每个弹托的分离是不平衡的。定量描述这种不平衡性,对研究和改进弹托的设计有重要的参考价值。我们是采用下述方法对弹托的空间位置进行计算的。(示意图如图7)

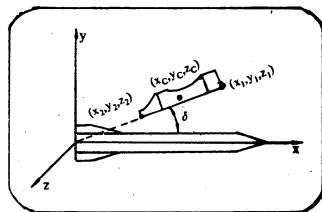


图7 计算弹托位置示意图

图7、计算弹托位置示意图

首先在每一弹托上取首尾两点  $A(x_1, y_1, z_1)$  和  $B(x_2, y_2, z_2)$ , (初始时  $A, B$  两点均在弹轴上), 这样  $A, B$  两点的连线即为该弹托的轴线。在每一站都分别对  $A, B$  两点施以相应的  $T_1, T_2, T_3, T_4$  变换, 得到它们在该站下的坐标  $A'(x'_1, y'_1, z'_1)$  和  $B'(x'_2, y'_2, z'_2)$ 。然后计算出由这两点组成的空间直线(即弹托的轴线)与弹体轴线(即  $x$  轴)之间的夹角  $\delta$

$$\delta = \arctan \frac{\sqrt{(y'_2 - y'_1)^2 + (z'_2 - z'_1)^2}}{x'_2 - x'_1}$$

$\delta$  即定量反映出每个弹托与弹体之间的相对位置, 若每个弹托  $\delta$  的相差很大, 就说明该弹托的分离非常不平衡。

对弹体的运动来讲, 我们最感兴趣的是它在各站的俯仰角  $\alpha$ , 偏航角  $\beta$  和攻角  $\delta$ 。这些参数对研究脱壳穿甲弹的气动力特性以及改进弹体的外形设计有很重要的意义。

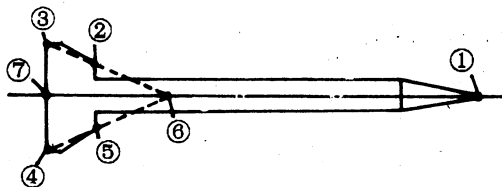


图8 确定弹轴端点坐标示意图

弹体的运动可看成是弹轴的运动。要确定弹轴, 实际上就是要确定弹轴的两个端点的坐标。(示意图如图8)。我们利用照片上弹体的五个特征点的坐标, 先由②③和④⑤确定出⑥的坐标, 再由⑥①和⑤④确定出⑦的坐标。对同一站的两张正交的照片(一张是  $xz$  平面的投影, 一第是  $xy$  平面的投影)上的特征点做同样的计算, 即可得出弹尖①的坐标  $(x_1, y_1, z_1)$  和弹尾中心⑦的坐标  $(x_{71}, y_{71}, z_{71})$ , 进而计算出[4]:

$$\alpha = \operatorname{tg}^{-1} \frac{y_1 - y_7}{x_1 - x_7}; \quad \beta = \operatorname{tg}^{-1} \frac{y_1 - y_7}{x_1 - x_7}$$

在短距离内,若认为速度方向不变,速度矢量一直与x轴平行,则攻角 $\delta$ 为:

$$\delta = \operatorname{tg}^{-1} \frac{\sqrt{(y_1 - y_7)^2 + (z_1 - z_7)^2}}{x_1 - x_7}$$

由此计算绘制出的 $x-\delta$ ( $x$ 为各站在总测量系统下的x坐标)曲线和 $\beta-\alpha$ 曲线很好地反映了弹体的章动规律。如图9所示。

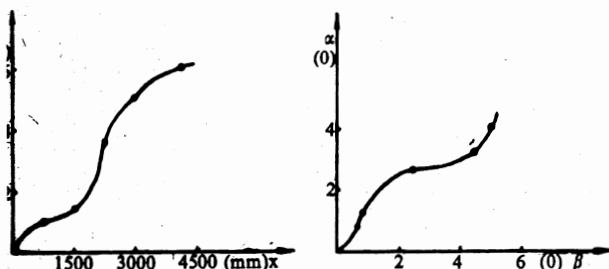


图9  $x-\delta$ 曲线和 $\beta-\alpha$ 曲线

## 五、结论

上述计算机处理系统我们已在APol110机上实现,其中图形部分我们调用了该机上的DOMAIN2D-GMR绘图软件包。应用该系统我们对多种高速脱壳穿甲弹的脱壳过程进行了处理,均得到了满意的效果。将计算机图形学技术应用于该领域在我国还是首次。它的设计思想还可广泛应用于航空、航天等部门,为高速飞行体的设计与研究提供服务,其应用前景是很广阔的。

## 参考文献

- [1] Schmidt, E. M. "Disturbance to the Launch of Fin-stabilized Projectiles" Journal of Spacecraft and Rockets, V. 1. 19, Jan-Feb, pp. 30-35 1982
- [2] 金延赞,《计算机图形学》,浙江大学出版社,1988
- [3] 蔡士杰,《计算机图形学入门》,江苏科技出版社,1986
- [4] 任国民,曾士论,《靶道测试技术》,华东工学院教材,1984

# 实用代码字典的查找和对照显示技术

新疆 张孟春 王国红

在干部人事管理等一些MIS系统中,许多设计者都大量使用了代码。随着大量代码的引入,在处理系统的输出时,如何构造一个合理的代码字典,设计一个好的数据结构和简洁的算法,使系统在将代码翻成汉字时有较高的效率,在系统输入时,又能方便地在屏幕上采用翻页的办法直接查找代码与汉字的对照表,供选择输入就成为系统设计时需要认真研究的问题。本文介绍了一种实用的代码字典的查找和对照显示输入技术。笔者曾在干部管理信息系统中使用,在长城0520-CH机FOXBASE环境下也取得了满意的效果。

## 一、字典库结构说明

库名 GDXS. DBF

字段名: NO, C, 2

字段名: DZ, C, 81

索引文件名: GDXS. IDX

索引关键字: NO+LEFT(DZ, 4)

字段NO存放各类代码的分类标志,用两位字符表示。字段DZ存放代码与汉字的对照表,字段宽度为

81个字符,与显示器能够显示的字符数相对应。按关键字NO+LEFT(DZ, 4)进行索引,是为了在某类代码数量比较多的情况下,第一次查找定位时能离目标更接近,提高查找速度。图一是某干部管理系统字典库中的国家地区代码和家庭关系代码部分,国家地区代码为国标,家庭关系代码为部标。在DZ字段中,每个代码的汉字后面必须留一位空格,做为截取汉字字符串的结束标志。

## 二、代码对应汉字的查找函数

(自定义函数见程序一)

算法充分利用 SEEK 命令定位速度快, LOCA 命令定位灵活的特点, 使用 SEEK 命令进行第一次定位, 使用 LOCA FOR <逻辑表达式> WHILE <逻辑表达式> 命令进行第二次定位, 使用 AT() 函数进行第三次定位并截取汉字字符串的基本设计思想。

函数第一参数为第一次定位时的索引关键字(注意: 在字典库进行索引时我们取了 DZ 字段的左边四位, 在用 SEEK 命令查时我们可根据实际情况取一位、二位、或四位), 第二参数为需要查找的代码值, 当第一参数或第二参数为空, 或者没有找到, 函数返回一个空串, 如找到函数返回对应代码的汉字串。现以查找家庭关系代码 75(姐姐)为例, 说明函数的调用过程。

\* ZK. PRG

SET TALK OFF

SET ECHO OFF

SET STAT OFF

USE

USE

.

.

.

CLEAR

USE GDXS INDE GDXS

G1 = "75"

? DZ1 ("GX" + LEFT (G1, 1), G1)

USE

.

.

.

RETURN

运行 ZK, 查找过程中, 第一次查找定位于 62 记录, 并回跳一条记录, 第二次在 61 和 62 两个记录范围内定位于 61 记录, 第三次在 61 记录中准确定位并截取汉字字符串“姐姐”显示在屏幕上。

## 三、翻页对照显示输入程序(见程序二)

程序利用 FOXBASE 的 READ() 读键盘函数, 其返回值为:

6 或 262 时表示 PgUp 键

7 或 263 时表示 PgDn 键

15 或 271 时表示输入超过末尾处

程序使用 CASE 语句判别不同的键值进行上翻、下翻或者输入操作。

功能:

从屏幕给定行开始, 使用上下翻页键查询某类代码的对照表, 并在给定屏幕行、列座标位置输入所选定的代码值。

入口参数:

XSCS1 = "\*\*\*\*\*"

第 1-2 位: 字典库代码分类标志

第 3-4 位: 给定分页显示的起始行

第 5-6 位: 输入代码时的行座标

第 7-8 位: 输入代码时的列座标

后面所跟的空格为输入代码的长度, 代码有几位就给几位空格。

出口参数:

XSCS1 为选择输入的代码值

下面以查询输入家庭关系代码为例, 说明程序的调用过程。

给定入口参数:

查询家庭关系代码: GX

分页显示的起始行: 20

输入时的屏幕座标: 18, 16

代码长度为两位

\* ZK. PRG

SET TALK OFF

SET ECHO OFF

SET STAT OFF

.

.

.

@ 18, 0 SAY "请输入与本人关系:"

XSCS1 = "GX201816"

DO XSSR

? XSCS1

.

.

RETURN

程序的返回值为所输入的代码值。

本文提供的字典库和程序全部可在 FOXBASE(25 行)下运行, 实际应用时还可扩充, 比如增加到页末, 到页首操作。



## NO CZ

GB 004. 阿富汗	008. 阿尔巴尼亚	010. 南极洲	012. 阿尔及利亚
GB 016. 美属萨摩亚	020. 安道尔	024. 安哥拉	028. 安提瓜
...			
GB 810. 苏联	818. 埃及	826. 英国	834. 坦桑尼亚
GB 840. 美国	849. 美属太平洋群岛	850. 美属维尔京岛	854. 上沃尔特
GB 858. 乌拉圭	862. 委内瑞拉	872. 威克岛	876. 瓦利和富图纳岛
GB 882. 西萨摩亚	886. 也门	890. 南斯拉夫	894. 赞比亚
GX11. 夫    12. 妻	21. 独生子    22. 长子	23. 次子    24. 三子	25. 四子
GX 26. 五子    27. 养子	28. 女婿    29. 其他儿	31. 独生女    32. 长女	33. 二女
GX 34. 三女    35. 四女	36. 五女    37. 养女	38. 儿媳    39. 其他女	51. 父亲
GX 52. 母亲    53. 公公	54. 婆婆    55. 岳父	56. 岳母    57. 继父	58. 继母
GX 61. 夫兄    62. 夫弟	63. 夫姐    64. 夫妹	65. 妻兄    66. 妻弟	67. 妻姐
GX 68. 妻妹    71. 兄	72. 嫂    73. 弟	74. 弟媳    75. 姐姐	76. 姐夫
GX 77. 妹妹    78. 妹夫	81. 伯父    82. 伯母	83. 叔父    84. 婶母	85. 舅父
GX 86. 舅母    87. 姨父	88. 姨母    89. 姑父	90. 姑母    91. 堂兄妹	92. 表兄弟
GX 93. 侄子    94. 侄女	95. 外甥    96. 外甥女	97. 亲属    98. 非亲属	

## 程序一

\* DZ1. PRG

PARA WDY1, WDY2

```
IF WDY2 == SPAC (LEN (WDY2), OR, WDY1 ==
SPAC (LEN (WDY1))
```

WDY=""

ELSE

IF LEN (WDY1) &lt;= 2

SEEK WDY1

LOCA WHILE NO=WDY1 FOR WDY2 \$ DZ

ELSE

SEEK WDY1

SKIP-1

```
SKIP IIF (NO=LEFT (WDY1, 2), AND, WDY2 $ DZ, 0,
1)
```

```
LOCA WHILE NO=LEFT (WDY1, 2), AND, (LEFT (DZ,
LEN (WDY1) - 2) = SUBS (WDY1, 3), OR, WDY2 $ DZ
FOR WDY2 $ DZ
```

ENDI

```
WDY=IIF (. NOT. FOUN (), "", SUBS (DZ, AT (WDY2,
DZ) + LEN (WDY2) + 1, AT ("", SUBS (DZ, AT
(WDY2, DZ) + LEN (WDY2) + 1)) - 1))
```

ENDI

RETU WDY

## 程序二

\* XSSR. PRG

XSCS=SUBS (XSCS1, 1, 2)

XSH=VAS (SUBS (XSCS1, 3, 2))

XSHH=VAL (SUBS (XSCS1, 5, 2))

XSL=VAL (SUBS (XSCS1, 7, 2))

XSCS1=SUBS (XSCS1, 9)

SELE 10

USE GDXS INDE GDXS

SEEK XSCS

IF. NOT. EOF ()

COUN WHILE NO=XSCS TO XSZS

SEEK XSCS

@ XSH, 0 CLEA

I=XSH

DO WHILE I&lt;24

@ I, 0 SAY TRIM (DZ)

SKIP

I=I+1

IF NO#XSCS

EXIT

ENDI

ENDD

XSJS=I-XSH

@. 24, 0 SAY "pgUp: 上翻 pgdn: 下翻"

DO WHILE . T.

```

@ XSHH, XSL GET XSCS1 PICT "@!"
READ
DO CASE
CASE LTRI (XSCS1) # XSCS1. AND. XSCS1 # SPAC
(LEN (XSCS1))
@ XSHH, 61 SAY "选择错误. 请重选 *"
LOOP
CASE READ () = 7. OR. READ () = 263
IF. XSJS >= XSZS
LOOP
ENDI
@ SXH, 0 CLEA TO 23, 79
I=XSH
DO WHILE I<24
@ I, 0 SAY TRIM (DZ)
SKIP
I=I+1
IF NO # XSCS
EXIT
ENDI
ENDD
XSJS=XSJS+I-XSH
CASE READ () = 6. OR. READ () = 262
IF XSJS <= 24-XSH
LOOP
ENDI
SKIP-24+XSH-IIF (MOD (XSJS, 24-XSH) = 0, 24-
XSH, MOD (XSJS, 24-XSH))
XSJS=XSJS-24+XSH-IIF (MOD (XSJS, 24-XSH)
=0, 24-XSH, MOD (XSJS, 24-XSH))
@ XSH, 0 CLEA TO 23, 79
I=XSH
DO WHILE I<24
@ I, 0 SAY TRIM (DZ)
SKIP
I=I+1
IF NO # XSCS
EXIT
ENDI
ENDD
XSJS=XSJS+I-XSH
CASE READ () = 15. OR. READ () = 271
IF LEN (TRIM (XSCS1)) = 0
EXIT
ENDI
XSJL=IIF (EOF (), RECC (), RECN ())
SEEK XSCS+LEFT (XSCS1, 1)
SKIP-1
SKIP IIF (NO=XSCS. AND. XSCS1 $ DZ, 0, 1)
LOCA WHILE NO=XSCS. AND. (LEFT (DZ, 1) = LEFT
(XSCS1, 1). OR. XSCS1 $ DZ) FOR XSCS1 $ DZ
IF. NOT. FOUN ()
@ XSHH, 61 SAY " * 选择错误. 请重选 * "
GO XSJL
LOOP
ENDI
EXIT
ENDC
ENDD
@ XSHH, 57 SAY SPAC (22)
ENDI
USE
RETU

```

\*\*\*\*\*

(上接 9 页)设备驱动程序与连接外部设备的根本内涵,需要指出的是,并不是说,在完成这些工作时,用户要亲自完成其中的每一个细节,因为驱动程序的安装程序及系统的连接程序已默默地为我们完成了其中的很多工作,以至于这些工作已对用户透明。

鉴于目前国内使用 XENIX 的用户较多,文中就使用 XENIX 的外部设备的有关问题展开了讨论。值得一提的是, XENIX 内核的基本结构与 UNIX 是类似的,因而,本文的内容适用于 UNIX 系统。

#### 参考文献

- (1)石文昌,“UNIX 输入/输出子系统及其移植实现”,中国科学院软件研究所硕士学位论文,1990. 4.
- (2)孙玉方、石文昌,“VAX 机的 RA81 硬盘在 UNIX 系统中的驱动原理”,《小型微型计算机系统》,1990,第 11 卷,第 2 期.
- (3)刘日升、孙玉方编著,《UNIX 操作系统原理与应用》,能源出版社.

# 在 XENIX 中连接外部设备的秘诀

广西计算中心 石文昌

**摘要** 在 XENIX 环境中,打印机、终端和磁盘等外部设备在物理上正确地接到主机上后,在逻辑上还要正确地连接到系统中才能使用。很多用户对设备与系统连接的真实含意不甚了解,在连接设备时,如连接打印机或终端时,或安装多用户卡及其驱动程序时,常常感到困难,特别是出现异常时,不知所措。本文根据作者在完成国家“七.五”科技攻关项目“UNIX 操作系统的移植及国产化”过程中对 UNIX 和 XENIX 操作系统的研究成果进行了剖析,揭示出在 XENIX 中连接外部设备的内在本质,同时探讨了这些知识在解决具体问题时的指导作用。

**一、引言** 和 DOS 单用户系统相比,用户在 XENIX 多用户系统中管理外部设备的任务要相对艰巨和复杂,其主要原因之一是用户对 XENIX 本身管理外部设备的内在机理不甚了解。其实,在进行外部设备的管理工作时,只要作到知其然亦知其所以然,则一切问题都将会迎刃而解。本文以方便应用为宗旨,剖析外部设备在 XENIX 系统中的本来面目,同时指出怎样用它来指导实际工作。

## 二、用户与 XENIX 外部设备间的操作界面

用户与 XENIX 间的操作界面可以用图一来说明。从软件的角度而言,用户不能直接与外部设备打交道。直接控制外部设备的是设备的驱动程序。用户访问外部设备的要求由 XENIX 操作系统的接口程序进行解释,并转换成对文件系统的操作,交由文件系统进一步处理,文件系统最终把用户的要求传递给设备驱动程序,由它真正驱动外部设备的动作。

用 户
XENIX 操作系统接口
文件系统
设备驱动程序
外部设备

图一

一般说来,用户对外部设备的访问,中间要顺次依靠 XENIX 的操作系统接口、文件系统和设备驱动程序这三个功能模块的配合作用才能实现。这三个功能模块是缺一不可的。操作系统接口和文件系统这两个模块是 XENIX 基本系统的标准结构,不会因外部设备不同而变化。设备驱动程序则不同,它高度依赖于具体的外部设备,不同的外部设备需要不同的驱动程序驱动。XENIX 基本系统为若干类型的外部设备,如标准硬盘、软盘、打印机和串行端口(终端)等,提供了驱动程序,对于这些类型的设备,无需另外安装设备驱动程

序就能使用,对于 XENIX 基本系统所支持的标准外部设备以外的其他外部设备,如某些多用户卡等,必须另外安装相应的设备驱动程序才能使用。

## 三、XENIX 设备驱动程序及其安装原理

XENIX 操作系统逻辑上要同时管理多个设备,它利用中断机制实现与设备操作的同步。XENIX 设备驱动程序由基本驱动程序和中断处理程序两大部分构成。基本驱动程序通过程序调用的方式被启动,用于主动地驱动设备的操作。中断处理程序由硬件中断机制启动,用于处理设备产生中断后的有关事务。

与设备驱动程序密切相关的两个重要参数是中断向量和设备地址,这是由硬件确定的两个值。在一些设备驱动器上,如多用户卡上,配有相关的开关部件,通过适当地设置这些开关可调整这两个值,以符合主机的要求。设备驱动程序是通过设备地址来控制设备的,设备驱动程序必须获得这个值才能工作。

在安装驱动程序时需要知道设备的中断向量。设备产生中断时,硬件产生对应于该设备的中断向量,中断机制以该中断向量为入口地址到中断向量表中取中断处理程序执行,因此,安装驱动程序时,要以中断向量为入口地址,把驱动程序的中断处理程序的调用接口填入到中断向量表中。

安装驱动程序时也要知道一个称为设备号的值。设备号由软件设置,它由两部分组成,一部分称为主设备号,一部分称为次设备号。主设备号标识插在主机上的各类设备,每类设备有一个主设备号,如硬盘和打印机各有一个主设备号。次设备号标识同一类设备中的不同设备,如一个多用户卡上的所有端口拥有相同的主设备号,而每个端口分别拥有一个彼此不同的次设备号。XENIX 内部用称为设备开关表的数据结构来组织设备驱动程序(设备开关表分字符设备开关表和块设备开关表,本文在不影响说明问题的前提下使用其



总称),当要驱动某设备的操作时,XENIX 操作系统的其它模块以该设备的主设备号为入口地址,到设备开关表中取基本驱动程序执行,而驱动程序进一步利用次设备号驱动具体的设备。因此,安装驱动程序时,必须以设备的主设备号为入口地址,把驱动程序的基本驱动程序的有关调用接口填写到设备开关表中。

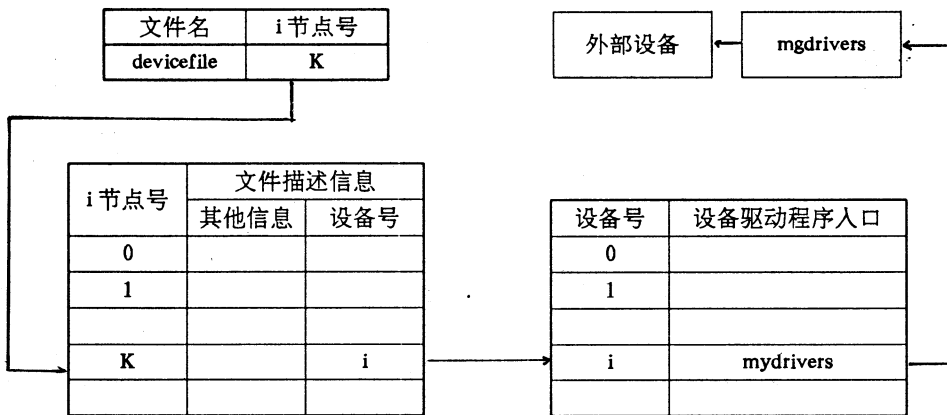
XENIX 操作系统内核是一个完整的可执行程序,设备驱动程序等其他模块都只是一些子程序的集合,不能独立执行,只能在操作系统内核中被调用。随设备驱动器(如多用户卡)一起提供给用户的设备驱动程序实际上是一些 C 语言的目标文件。驱动程序的安装过程本质上就是把驱动程序的目标文件和 XENIX 基本系统的目标文件进行链接,生成新的 XENIX 操作系统可执行程序的过程。通常所说的用 make 来重新生成系统,实际上就是完成这项工作。make 是一个用于对大型程序进行维护的工具软件,它根据事先定义的操作文件(缺省文件名是 makefile 或 Makefile)的内容执行既定的操作。XENIX 向用户提供的用于重新生成系统的供 make 使用的操作文件定义了更新 XENIX 操作系统可执行程序的方法。

#### 四、在 XENIX 中连接外部设备的意义

在 XENIX 系统中,设备被作为文件来处理。设备对应的文件称为设备特别文件,在用户面前,它们与普

通文件没有什么两样。用户使用与访问普通文件完全相同的方法,通过文件名访问外部设备。设备文件存放在目录/dev 中。一个文件在目录项中登记有文件名和 i 节点号等两项内容。XENIX 内部用 i 节点对文件加以描述,文件与 i 节点一一对应。不同文件的 i 节点组织在 i 节点表中,一个文件的 i 节点号是该文件的 i 节点在 i 节点表中的入口位置。i 节点详细记录文件的各种信息,其中有文件的属性、权限、历史及其在磁盘上的位置等,设备文件的 i 节点记录有该设备的设备号。

用户访问设备,不外乎两种途径,或者通过命令访问,或者在程序中访问。不管采取哪种途径,都需要依赖设备的驱动程序去完成最终的目的。驱动程序知道设备在地址空间中的地址,从而也就知道设备在机器中的具体位置。设备驱动程序存放在设备开关表中,设备所对应的设备号(严格地说是主设备号,为简单起见,以下不再区分主设备号、次设备号和设备号,笼统地称为设备号)就是驱动程序在设备开关表中的入口地址,这样,利用 i 节点中提供的设备号,便可寻址到设备的驱动程序,这为实现对设备的访问奠定了基础。由 i 节点表到设备开关表的过渡是从文件系统到 I/O 子系统的根本性过渡,而设备号就是实现这个过渡的关键媒介。描绘了设备文件到设备间的映象关系。



图二

安装了驱动程序之后,设备在设备开关表中找到了自己的位置,显然,仅有这些,用户还无法对设备进行访问,还必须在文件系统中为设备创建相应的文件,在 i 节点表中分配对应的表项,并建立起 i 节点到设备驱动程序间的映象关系。通常所说的对外部设备的连接就是完成这项工作。

相同的设备在不同的系统中可以有不同的文件名,在实际应用中,不要被这种现象所困惑。设备文件是设备在文件系统上的映象,而文件名只是一种代号,起什么名称是无关紧要的,关键的是设备文件与设备驱动程序间要建立起正确的映象关系。

#### 五、结束语

以上论述了在 XENIX 系统中安装(下转 7 页)

## 微机对未来进行预测的方法

吉林省计算机技术研究所 张鸿鸣 李志英

**摘要** 本文论述了对未来进行科学预测的重要意义,并介绍了微机预测未来的方法、步骤。

预测的目的在于认识自然和社会的发展规律,以及在各种历史条件下各种规律的相互作用。对未来的预测可以揭示发展趋势,使人们尽早发现未来的苗头,并能能动地控制其发展,使其为社会和人类服务。

预测的重要性在于它是经济活动中的第一环节,它是制订决策和规划的前提条件。在科学技术高度发展的今天,任何科学,任何技术的发展都离不开预测。一项科学的预测将有助于制订正确的决策,加速科学技术的发展,造福于社会。

预测不是目的,技术预测是提供一种信息,为制订决策和规划服务。在技术发展中,技术预测就是对未来的技术变迁作出概率估计。为此,研究微机对未来进行预测的方法十分重要。

微机对未来进行预测的方法大致以以下几个步骤来实现:

(1)预测对象分析;(2)预测模型建立;(3)预测方法确定;(4)预测微机程序;(5)预测结果分析;(6)预测价值评估。

从预测实现的步骤和层次中,我们就可以全面的了解微机对未来进行预测的方法,只有选择最佳的预测方法,才能得到精度高和有价值的预测结果。

### 一、预测对象分析

正确分析预测对象,是正确预测的关键和前提。预测对象分析就是分析预测对象的性质,也就是分析对象的内部矛盾及该矛盾诸方面所占的地位。这种分析需要进行大量地事实调查,查阅大量资料文献和集中各种经验。

同时,预测对象分析与预测目的有极其重要的关系。例如,当前城市人口增多,人民生活水平大大提高,用房增多,因此,占可耕地面积增多,以此作一预测。

(1)如果预测的目的只是为了警告有关部门,则问题就比较简单。可以用下述公式来说明:

$$\Delta = U \cdot M \cdot F \cdot (1 + K)^N - S$$

其中  $U$  为城市人口总数;  $M$  为平均每人占地面积;  $F$  为住房或商业用房中可耕地的份额;  $K$  为人口的增长百分数;  $S$  为新开垦的可耕地面积;  $\Delta$  为可耕地损失;  $N$

为年限。这里的  $U$ 、 $K$  和  $S$  是已知数,可用统计的方法得到。但  $M$  和  $F$  得到准确数字则需要大量调查研究和统计。但无论如何,有了上述公式和统计数字,这个预测结果很容易得到。

(2)若预测目的是解决因盖房过多而使耕地面积减少的问题,这就复杂多了。需要分析以下几个问题:

①城市住房的一般趋势;②城市人民的的生活方式;③商业、服务业方式及发展趋势;④城市能源的需求趋势及解决途径;⑤城市居住方式的可能变化。

这就不是能用一个简单公式所能解决的。

预测对象分析不仅与预测目的有关,而且还同预测理论有关。

在海洋捕捞业中,超级大国采用野蛮的掠夺方式,在公海上捕鱼。在他们看来,他们的船多,网也大,就可以多捕鱼。但若从生态平衡来看,需要提倡合理捕捞和合理繁殖,显然两者观点不同,因此理论也不同。前者是生态破坏理论,后者是生态平衡理论。所以预测的目的决定预测性质,预测范围和预测理论的应用,也就决定了预测的复杂程度。

值得注意的是,有些预测性质不象上面那样简单明了,而是隐含其它的预测之中。预测对象分析是微机对未来进行预测方法的第一步,预测者需要花费很大的力气才能作好。

### 二、预测模型建立

正确地分析了预测对象,弄清了预测性质,下一步骤就是建立预测模型。所谓预测模型就是用尽可能简单的形象方法描绘出所要预测的对象是什么样的,预测模型要在整体上表现出预测对象及其相关因素的联系、依存、变化和运动的关系。

建立预测模型与所采用的理论有关,应当说理论决定模型。不同的指导理论就有不同的模型,但这要取决于对预测对象的分析。有的预测对象需要用多种理论,因此,就会有多方案模型。为了深刻认识预测对象,人们在不同领域的预测中积累了很多经验,创造了许多模型。

(1)树状模型。它能形象地描绘出预测对象和相关因素的地位和联系。

(2)构造模型.它能直观地表达预测对象的组成和联系,可以清楚地表示预测对象的构造、系统和功能。

(3)程式模型.类似于化学工程中的流程图,它能表示预测对象的运动状态,因果关系及发展步骤,并能给出预测对象发展的控制步骤和重要程度。

(4)图示类比模型.预测对象是以其它事物的兴衰和长消为存在前提。

(5)先验模型.预测对象很复杂,需要借鉴历史。

(6)网络模型.对弄清多因素的复杂关系非常实用,特别对于规划性预测很方便。

(7)试验模型.特尔斐法就是其中一种。当预测者一无经验数据二又概念模糊时用此模型。市场预测和设计工作等大多采用这种模型。

(8)逻辑模型.它是根据事物合乎规律的发展为依据,由于预测者占有过去和现在的数据,就可以延伸推论。外推法和趋势外推法就是以这种模型为依据。

(9)公式化模型.它是以变量的函数组成的线性和非线性方程式。

上述模型基本上可分四大类:一是功能构造;二是机械组成;三是逻辑推论;四是公式化。究竟是那一类好,就要看哪一类模型能深刻而又简单的描绘预测对象。关于万物如何组成又如何分解,仍然是一个值得探讨的问题,只能具体问题具体分析。如果预测者不能清晰而透彻地说明预测对象的性质,他就不能建立有效的预测模型。而没有预测模型,也就不能进行任何预测。

### 三、预测方法确定

确定了预测模型,就比较容易进行预测了。依据预测对象的性质和预测模型,预测者就可以采用各种不同的方法进行预测未来。

1.数学法.将上述介绍的各种预测模型进一步用数学方法描述,有关因素就可以得到定量的说明,并可以得到计算机的算法,编程上机就可以计算出预测的结果。

在实际应用中将会发现,数学模型中的许多系数还是无法确定,或者精度不高。例如生长曲线方程

$$L = L_1 / (1 + aE^{-b})$$

其中  $a$  和  $b$  的值有时就很难确定,因此就无法得到确定的曲线形状。为了确定这些系数,就需要调查研究 and 试验,并对所取得的数据加以整理和分析。分析数据常用的方法就是概率统计中的回归分析和最小二乘法等。有些模型(树状模型和构造模型等),预测者为了能进行计算,常常对某些因素标准重要系数(这些系数是经验的,也有的是统计的),在必时还要作加权分析。

数学模型的好处就是能公式化,还可以变成微型

机的算法,并可程序化,说服力强。对于相关因素较少或性质较单一的预测事件,它简单明了,准确度高,易于用微机进行计算。特别对技术预测极易实现,但它要求基本数据准确可靠。

2、推理法.对于逻辑模型,经验模型和图示类比模型等很适用,它的理论基础是辩证逻辑。这类方法不完全是定性的,它的许多因素可以用数字或数学表达式表示,但对预测的整体或整个预测模型来说,还需要用推理或推论的方法做出许多解释,原因在于许多因素还比较模糊,数据不全。这类方法所得的结论可以是定量的。如预测汽车制造业的发展速度和前景等。也可以是定性的,如预测技术更新的可能性等。

3、实验法.民意测验的方法就是其中的一种。特尔斐法市场调查及某些科学试验预测都属于这一类,它要反复进行多次,得到的数据也要严格处理,这种方法的好处是简便、易行,特尔斐法和试验法几乎能用于一切预测。

### 四、微机预测程序

应用微机解决预测问题,事实上包括提出和分析问题、建立数学模型、选择计算方法、编写和调试程序、准备原始数据、实际上机计算以及评价计算结果的全过程。

任何实际计算问题都是由一定的自然规律支配的运动过程,反映在运动着的量与量之间有一定的数值关系。用一组数学公式来描述这个运动过程,并尽可能真实地反映出量与量之间的数值关系,这一工作称之为建立数学模型。这样的数学模型,并不一定适用计算机直接计算。因此需要选择适于计算机直接计算的计算方法,这就称之为算法。有了算法和初始数据,在编程序之前必须先画好程序框图。所谓程序框图就是在给计算机编程序之前,必须仔细考虑到可能发生的每一种情况。通常是先把各个步骤排好顺序,然后才能着手用计算机语言来编写程序。因此,程序框图是编写程序时不可缺少的一种方法。程序框图又称流程图。画出程序框图的好处在于能直观地反映计算问题中较复杂的相互关系,才能清楚地描述程序的计算步骤和流向。比较完整的程序框图有助于正确无误地编写程序。也有利于程序员之间的思想相互交流。

绘制程序框图应当仔细校对,并应当保证两点:①没有遗漏任何必要步骤;②绘成的程序框图在一切场合都适用。画程序框图的过程,细分起来还可包括框面构思、草拟粗框和不断改进等步骤,而且必须与程序的总体规划紧密配合。

所谓微机预测程序就是根据预测的问题,由人事先安排好计算步骤,把程序和预测问题中的原始数据输入计算机,计算机才能根据程序和原始数据计算出结果。编预测微机程序必须使用算法语言(BASIC和



FORTRAN等语言),算法语言已经是较高级的程序自动化语言了。它既与数学公式非常接近,又与计算机内部逻辑结构无关。算法语言使得程序设计人员可以不用考虑机器内部逻辑结构,因而可集中精力去推敲计算方法的逻辑和计算过程的描述。

动手编程之前,下工夫思考和分析自己的预测问题、整理计算公式和选择恰当的计算方法,是能取得更好预测结果的关键。把预测计算过程描绘出来的工作就是程序设计,因此,程序设计是预测过程的重要环节。

微机预测程序最好是模块化,应有主程序,又有若干子程序,由主程序调用若干子程序,来实现预测结果的计算。只有模块化,每个模块有自己对应的程序框图,这样的程序才能系统性强,便于阅读、修改和维护。专职搞预测最好建立预测数据库和算法库,积累大量的预测算法和数据,对今后进行预测创造了更方便条件,也能大大提高预测结果的精度和社会价值。

### 五、预测结果分析

预测结果的分析也就是预测精度的分析,分析预测结果的准确度。为了说明这一问题,有必要研究误差和相对误差两个概念。

如果 $X_1$ 为准确值, $X_2$ 为预测值,则 $e = X_1 - X_2$ 之差被称为误差,即所谓误差就是预测值与准确值的偏差。同样, $e' = e/X_1 = (X_1 - X_2)/X_1$ 之比被称为相对误差。即所谓相对误差就是误差 $e$ 在准确值 $X_1$ 中所占的百分比。因此,可以用 $e$ 和 $e'$ 来表示预测精度。

由此不难看出,影响预测结果的主要因素是基本资料、原始数据、预测方法和相关理论掌握程度。在进行预测因素的选择、模型的建立以及预测结果的计算过程中,都和原始数据和资料有关。数据和资料的完整与否,正确与否直接影响预测结果的精度。因此,为进行一项预测工作,必须投入相当大的力量去进行数据和资料的搜集和筛选工作,必要时还要进行抽样调查,这样才能得到有用和完整的数据及资料。

客观事物有各种各样的表现形态,所表现的统计规律也不同。所以如何针对不同的问题,选择不同的预测方法对保证预测结果的精度是十分重要的。这就要求预测者必须对预测对象所服从的统计分布特性,遵循的物理定律以及各种预测方法的原理、基本假设、适用范围等有极深刻的了解,才可以做到理论明确,方法简明,预测结果精度高。

预测物理模型是根据预测对象所在领域中的基本理论而建立的,它能反映预测对象的内在规律性。正如计量经济模型就可以反映出各经济元素之间的相互关系,为此,不仅需要大量的准确数据资料以及正确地判断内生变量和外生变量,同时还必须根据经济理论对

它做出一定的约束。不同的基础理论,决定不同的预测方法和不同的预测结果。因此,预测的本身就是一门综合性科学。

预测者在给出一项预测结果的同时,为了决策的需要必须给出预测的结果精度分析,采取一定的方法检验一下预测的效果如何。检验预测结果的精度方法很多,针对不同的预测问题有不同的检验方法。当然,不同的检验方法有各自的特点。回归分析法,它的回归效果检验就是一种精度分析法,通过显著性检验,对已得到的回归方程给出置信度,或者说按此预测所具有的成功把握性。同样对直观预测中的特尔斐法也有相应的精度分析法,它可以通过专家组的平均权威系数、专家组人数以及专家组意见的协调程度等,对专家的预测结果精度作出分析。

### 六、预测价值评估

预测价值也是很重要的问题,没有人想做无用的工作。如何评价一项预测的社会价值,是否是只有未来得以实现的预测才是有价值呢?应当说,预测的社会效果和它最后的正确性之间的关系取决于决策者对预测对象的可控制程度。有一类事物是人类无法控制的,因此,人们只有使自己的行为适应客观事物的结局,争取有利无害。在这种情况下,是否成为事实就是评价预测的社会价值的唯一标准。人类社会的发展是有它的客观规律性,然而这种规律的实现又是靠人能动性来实现,所以这类事物有些是部分可控。这时的预测社会价值不能按照它是否成为事实来衡量,而是依据它在帮助决策者作出正确而又及时地决策过程的效果来衡量。

只是由于做出了这种预测,才促成它成为事实的预测,人们称这种预测为自力预测。正如有人作出一个关于明年某地将出现经济不佳的预报,假如这一预报被广泛流传和接受,那么合理的响应是偿清一切债务,出售一切存货等,若相当多人都这样作起来,经济不佳的困境将不可避免地加速到来。

只是由于作出了预测才使预测结果不能实现的预测,虽然预测没有实现,但是对作出正确决策来说却有很重要的价值,这种预测叫自拆台预测。例如,五十年代时,我国就有人提出人口的增长应当控制,否则到本世纪末,将出现粮食和生产的的增长赶不上人口增长的局面。这一预测如果当时能被采纳,并且能实施有效地控制人口措施,无疑这个预测局面将不可能出现。遗憾的是,这一科学预测却被说成是唯心史观,误信为人口越多越好。其结果人口从五亿猛增到十亿多,如果再不采取措施,情况将会更严重。显然,若是从预测对领导作出正确决策时的效果来衡量,这个关于人口的预测具有极高的社会价值。

## MCS-51 单片机开发工具软件

四川省电子计算机应用研究中心 林化

随着计算机技术的发展,单片机以其高可靠性,超小型化和价格低等特点,在越来越多的领域得到广泛应用。MCS-51 单片机芯片在国内市场上一出现,即受到国内广大用户的欢迎。目前国内开发各类智能化仪器已开始大量采用 MCS-51 单片机,因此能有一种简单、实用、性能优良的 MCS-51 单片机开发环境,成为广大开发人员的心愿。

目前,MCS-51 单片机仿真器大多以 PC 机作宿主主机,使开发人员可以利用 PC 机的资源,但大多数仿真器配置软件产生的目标代码均需通过仿真器写入 EPROM,而不能直接利用 PC 机的 EPROM 写入器。尽管许多开发人员已有 PC 机的 EPROM 写入器,但仍要求购买仿真器相应的 EPROM 写入器和相应软件,从而增加了开发成本,并且绝大多数仿真器仿真内存有一定限制,如用户程序过大,在线仿真就有一定困难。

复旦大学的仿真器是国内应用最早和最广的仿真器之一,它允许用户程序最大 16K。我们开发一套智能化仪器时,使用的 EPROM 是 27256,在线仿真即遇上困难,同时,我们也遇到其它单位,由于没有购置仿真器 EPROM 写入器,或不稳定等故障造成写入器不能工作,使开发工作难以继续,因此我们参照复旦仿真器的相应软件,组织了一组 MCS-51 的开发工具软件包,包括编辑程序,MCS-51 汇编程序,通讯程序,转

换程序以及 EPROM 写入程序(对应 PC 机某一种 EPROM 写入器)转换程序把汇编程序产生的目标文件(格式同复旦汇编程序产生的目标文件)转换成后缀为 16 的二进制文件,该二进制文件可利用 PC 机 EPROM 写入器写 EPROM。

同复旦大学仿真器组合软件相同,汇编程序生成的目标文件中,前八个 byte 是用于同复旦 DSJ-51 型仿真器通讯的,其后为用户程序和数据,但这些程序和数据是以 ASCII 码形式出现的,因此转换程序的功能是除去目标程序中前八个 byte,然后将其后的数据由 ASCII 码转换成二进制,两个 byte 的 ASCII 字符转换成一个 16 进制数,转换后的 16 进制数存入后缀为 16 的二进制文件,转换完成后,可用 PC 机 EPROM 写入器写 EPROM,转换程序框图。该组合软件已在一些单位用于实际开发工作,本单位可负责提供这套软件。(可同作者直接联系)

已有复旦仿真组合软件的用户,利用转换程序可用 PC 机 EPROM 写入器,而不需购仿真器专用 EPROM 写入器,从而这组软件为开发人员节省了仿真器的专用 EPROM 卡和相应软件的费用。甚至可不用仿真器也能开发 MCS-51 单片机,这组软件不失为开发人员的一种简单、实用的开发工具。

## BASIC 语言动画技术应用实例

江西拖拉机发动机厂 黄焕如

提要:本文从实用的角度讨论了 IBMPC-BASIC 语言动画技术的实现,并由简到繁、从字体动画移动到图形动画变幻,列举了五个实例,这些实用程序简单易懂、动感性强、画面漂亮、可移植性好,很适合学习 IBMPC-BASIC 语言动画技术的读者使用,这些程序稍加修改也可应用于一些应用软件封面的制作。

计算机的图形学中的动画技术可以广泛应用于动画片、辅助教学、游戏、应用软件封面设计等方面。动画技术是指屏幕上显示出来的画面或其中一部分,能按一定的规则及要求屏幕上移动或变幻,从而使计算机显示的图形能动态变化的技术。IBMPC-XT 微型机 BASIC 语言(以下简称 BASIC)是一种简单易学、内容丰富和使用方便的程序设计语言,尤其是该语言具有较强的图形功能,为许多程序员所喜爱。

彩色图形显示器有两种基本的工作模式,即字母数字模式(A/N)和图形显示模式(APA),这两种工作模式可通过 SCREEN 语句来选择。字母数字模式中的

图形是由特殊字符或图形字符构成的,这种字符动画常见于一些游戏画面,在图形显示模式中,显示器屏幕上每个点(或称“像素”)均由程序控制其亮度和颜色,因而能显示出质量较好的图形,使用比较广泛,本文讨论的就是在 IBMPC-XT 微型机上彩色图形显示模式中的动画处理。

在图形显示模式中,按照分辨率的高低又可分为高分辨率、中分辨率和低分辨率三种模式,其中高分辨率模式屏幕显示:每帧 200 条线,每线 640 个点,每点只能取黑白两种颜色;中分辨率模式屏幕显示:每帧 200 条线,每线 320 个点,每点可以取 4 种不同的颜

色,低分辨率模式屏幕显示:每帧 100 条线,每 160 个点,每点可以取 16 种不同的颜色。目前 BIOS 中的驱动程序不支持低分辨率模式,而高分辨率下几乎没有彩色,因此一般均选择中分辨率模式。

在图形显示模式中,一般有两种产生动画的方式。一种是利用 GET 和 PUT 语句来保存和取出图形,称为“存取式”;一种是利用 PSET、LINE、DRAW 或 CIRCLE 等语句来描绘点、线或圆的动画图形,称为“轨迹式”。

### 一、“存取式”动画技术

为了方便使用,下面介绍“存取式”中 GET 语句的格式:

GET(X1,Y1)-(X2,Y2),Z

其中(X1,Y1)和(X2,Y2)分别是待保存的一个矩形区域信息的左上角和右下角点的坐标,Z是一个整形数组,该语句的功能是把指定矩形区域内每个象素的彩色码存放于数组 Z 中。必须指出,矩形区域的坐标应该容纳待保存在屏幕上的全部信息,数组 Z 的大小则必须足够容纳矩形中所有象素的彩色码。一般矩形区域坐标稍取大一些无妨,而数组 Z 的最小值应为:

$MIN = 4 + H * INT((W * B + 7) / 8)$

其中:H 和 W 是矩形区域的高和宽,B 是每个象素对应的二进制数目,中分辨率下 B 取 2,高分辨率下 B 取 1。

与 GET 语句对应的是 PUT 语句,PUT 语句的格式是:

PUT(X,Y),Z[,FS]

其中(X,Y)表示矩形的左上角坐标,Z 是由 GET 语句保存信息的整形数组名,FS 为运算选择方式(PSET、PRSET、XOR、OR、AND)。该语句的功能是把已保存的动画对象在屏幕上指定位置显示出来。

程序实例一就是这两条语句的简单应用,该程序中第 30 行定义一整形数组,用来存放需移动的汉字信息,第 40 行和第 180 行均在 2.13D 以上版本汉字操作系统下(16 行显示,以下同)调取 24 点阵汉字,第 60~150 行将“设计:”和“黄焕如”字符串先后左右移动,然后一起向上移动定位,第 160~170 行将 24 点阵的“工资汇总和分配系统”汉字字符串整体向下移动定位,第 190~300 行是美化用的画边框、画小圆和填色等。

10 REM IBMP-C-BASIC 语言动画技术应用实例之一

20 SCREEN 1: COLOR 1,0: CLS: KEY OFF: DEFINT A

30 DIM AC(1600)

40 PRINT CHR\$(14)\*"[@A-50|10^0\*2 工资汇总和分配系统]"

50 LOCATE 9,2: PRINT“设计:”

60 GET(0,90)-(66,110),AC%,LOCATE 9,2: PRINT“

”

70 FOR X=4 TO 120 STEP 2: PUT(X,90),AC%,PSET: NEXT

80 FOR X=120 TO 60 STEP-1: PUT(X,90),AC%,PSET: NEXT

90 FOR X=60 TO 100: PUT(X,90),AC%,PSET: NEXT

100 LOCATE 9,21: PRINT“黄焕如”

110 GET(150,90)-(250,110),AC%: LOCATE 9,21: PRINT“ ”

120 FOR X=148 TO 180 STEP 1: PUT(X,90),AC%,PSET: NEXT

130 FOR X=180 TO 148 STEP-1: PUT(X,90),AC%,PSET: NEXT

140 GET(80,90)-(250,110),AC%

150 FOR X=0 TO 20: PUT(80,90-X),AC%,PSET: NEXT

160 GET(20,1)-(300,40),AC%

170 FOR X=0 TO 30 STEP 5: PUT(20,0+X),AC%,PSET: NEXT

180 PRINT CHR\$(14)\*"[@M 50|140\*3 江拖发动机厂微机室]"

190 LINE(26,20)-(293,179),3,B

200 FOR Y=5 TO 195 STEP 23

210 CIRCLE(4,Y),2,3:CIRCLE(298,Y),2,3

220 CIRCLE(12,Y+10),5,3:CIRCLE(305,Y+10),5,3

230 CIRCLE(18,Y+20),3,3:CIRCLE(315,Y+20),3,3

240 NEXT

250 FOR X=18 TO 290 STEP 31

260 CIRCLE(X,5),2,3:CIRCLE(X,195),2,3

270 CIRCLE(X+14,10),5,3:CIRCLE(X+14,190),5,3

280 CIRCLE(X+26,15),3,3:CIRCLE(X+26,185),3,3

290 NEXT

300 PAINT(0,0),3,3

### 二、“轨迹式”动画技术

“轨迹式”动画技术是指通过描绘一系列的点、线或圆,然后又擦去它们而产生的动态感,一般来说,该方法每次化在图形的建立、擦去、重画所需的时间将随着图形的复杂程度而增加,如果运动轨迹的计算比较复杂,则动画的速度将明显降低,因此该方法仅仅适用于动画对象比较简单的场合。

程序实例二就是“轨迹式”的简单应用,该程序中第 30~50 行画出两个椭圆并在中间部分涂色,第 140~220 行为一子程序,其功能是以一定的节拍使椭圆中间不断变换各种颜色、不同椭圆度的圆框,同时把第 70 行定义的字符串中一个个汉字从屏幕外依次移入



画面。顺便指出,汉字移入的动画感实际上也是利用“轨迹法”,显示、擦去、再在下一位置显示,如此循环就好象汉字在移动一样,当然这种简单的动画技术不需要使用 BASIC 语言的绘图语句。

```
10 REM IBMPC-BASIC 语言动画技术应用实例之二
20 CLS;SCREEN 1;COLOR 0,2;KEY OFF
30 CIRCLE (159,65),120,1,,,4;CIRCLE(159,65),75,1,,,52
40 CIRCLE (159,65),121,1,,,4;CIRCLE(159,65),76,1,,,52
50 PAINT(50,50),2,1
60 PRINT CHR$(14)“[@M-50|120*1^5 固定资产管理系统]”
70 YS$=“江拖发动机厂微机室”
80 K1=1:K2=10:K3=0:K4=10:K5=1
90 K=36
100 LOCATE 14,K:PRINT MID$(YS$,K1,2):K=K-2
110 GOSUB 140
120 IF K(>)K2 THEN LOCATE 14,K+2:PRINT“ ”:GO-TO 100
130 K1=K1+2:K2=K2+2:IF K1(>)25 THEN 90 ELSE END
140 K3=K3+1:K4=K4-1:K40=K4/15
150 IF K3>3 THEN K3=1
160 IF K4<2 THEN K4=10
170 CIRCLE(159,65),44,K3,,,K40
180 FOR I=1 TO 50:I=I+1:NEXT
190 CIRCLE(159,65),44,0,,,K40
210 K5=K5+1
220 RETURN
```

### 三、“存取式”和“轨迹式”结合的动画技术

比较复杂的动画技术往往还必须把“存取法”和“轨迹法”结合起来使用,把动画对象保存在某一数组中,然后使其按一定的轨迹移动,比较复杂轨迹的计算可以利用 BASIC 语言中的一些三角函数,动画的效果更好,程序实例三就是“存取法”和“轨迹法”结合的简单应用,其中第 60~280 行是在屏幕上描画一台微型计算机的图样,第 330~350 行画一个涂色的圆,存入数组后擦去,第 360~460 行使这个圆按正弦轨迹运动。值得注意的是,第 30 行定义的字符串在高分辨率下存入数组,然后在(第 290 行)中分辨率下取出,从而简单地实现了一屏同时显示两种字体的效果。

```
10 RIM IBMPC-BASIC 语言动画技术应用实例之三
20 DEFINT A,B: DIM B(280): DIM A(160): DIM JS(100)
30 YS$=“1991.1”
```

```
40 SCREEN 2: LOCATE 1,1: PRINT YS$: GET(0,0)-(110,16),JS
50 SCREEN 1,0: COLOR 0,2: CLS: KEY OFF
60 LINE(180,70)-(272,120),0,BF'画显示器
70 LINE(180,70)-(272,120),1,B
80 LINE (189,75)-(240,115),1,B
90 LINE(252,108)-(262,110),3,BF
100 CIRCLE(257,83),2,,,1: CIRCLE(257,95),2,,,1
110 I=10: WHILE I(>)0: LINE(272+I,70+I)-(272+I,120-I*1.4),1:I=I-1: WEND
120 LINE (160,125)-(292,148),0,BF'画主机
130 LINE (160,125)-(292,148),1,B
140 LINE (210,128)-(280,145),1,B
150 LINE (245,128)-(245,145),1,
160 LINE (210,136.5)-(245,136.5),1,
170 LINE (250,140)-(254,141),2,BF
180 LINE (214,131)-(215,132),2,BF: LINE (214,140)-(215,141),2,BF
190 FOR I=1 TO 30 STEP 2.5: LINE (170+I,134)-(170+I,141),3: NEXT I
200 I=10: WHILE I(>)0: LINE (292+I,125-I)-(292+I,148-I),1:I=I-1: WEND
210 LINE (278,115)-(300,115),1: LINE (160,125)-(180,114),1,: PAINT(180,123),0,1
220 LINE(169,154)-(283,154),1,:画键盘
230 LINE(169,154)-(145,180),1:
240 LINE(145,180)-(263,180),1:
250 LINE (283,154)-(263,180),1: PAINT (170,170),0,1
260 LINE (145,180)-(263,183),0,BF: LINE (145,180)-(263,183),1,B
270 I=0: WHILE I(<=)2: LINE (283+I,154+I)-(263+I,180+I),1:I=I+.1: WEND
280 FOR N=0 TO 9 STEP 3: FOR I=1 TO 84 STEP 3
285 CIRCLE (175-N+I+2,163+N),1.5,,,1:
NEXT I: NEXT N
290 PUT(80,68),JS
300 PRINT CHR$(14)“[@I-30|24*2^0 产供销和仓库管理系统]”
310 PRINT CHR$(14)“[@M 20|100*1 江拖发动机厂]”
320 LOCATE 12,9: PRINT“微机室”
330 CIRCLE (32,152),14,2: PAINT(32,152),2,2
```

```

340 GET (18,138)-(52,172),A
350 LINE(10,130)-(60,180),0,BF
360' 圆动画部分
370 FOR X=20 TO 280 STEP 4
380 B(X)=159-CINT(ABS(SIN(X*7.853981E-02)*
    X)\2)
390 NEXT
400 L=7:Y2=150
410 FOR D=0 TO 1:S=20+D*260:E=280-D*
    260
420 I=1:FOR X=S TO E STEP (4-8*D)*2
430 IF X=S THEN LINE(10,130)-(60,180),0,BF
440 Y=B(X),N=(170-Y2)\S,IFF THEN PUT(X2,
    Y2),A
450 PUT=B(X,Y),A:X2=X:Y2=Y:F=-1:
    NEXT
460 L=(L+7)MOD 64:NEXT
470 SS$=INKEY$' 按任一键退出
480 IF SS$=" " THEN GOTO 400 ELSE END

```

#### 四、利用随机函数的动画技术

在一些特殊的 BASIC 语言动画技术中,也可以既不使用“存取式”,也不使用“轨迹式”,而利用随机函数 RND 取一个 0~1 之间的随机数,经过某种运算后,再利用 BASIC 语言的作图语句产生各种大小不同的图形,也能产生很强的移动感。在这种方式下是否需要擦去图形的这一过程,就要根据实际情况决定,有时省略这一过程可能效果更好。程序实例四就是利用随机函数的动画技术的简单应用,其中第 30~60 行是画边框显示题头,70~90 行是产生随机函数并运算后生成矩形的左上角和右下角坐标,第 100~170 行是随着“”声产生各种大小不同、颜色各异的矩形(表示各种报表),在这个程序中就没有采用擦去图形的做法,而让这些矩形相互复盖,就好象各种报表争先恐后地出现,给人以极强的动画感。

```

10 REM IBMPC-BASIC 语言动画技术应用实例之四
20 SCREEN 1,0:COLOR 1,1:CLS:KEY OFF:I=1
   :DIM M(88)
30 LINE (50,16)-(270,43),1,BF
40 LINE (54,18)-(266,41),3,BF
50 LINE (62,19)-(257,40),2,B
60 LOCATE 3,11:PRINT"经济核算报表管理系统"
70 IX1=RND*250+35:IX2=RND*250+35
80 IX2=(IX1-IX2)/3+IX2
90 IY1=RND*110+55:IY2=170

```

```

100 LINE (IX1,IY1)-(IX2,IY2),RND*2+1,BF
110 LINE (IX1,IY1)-(IX2,IY2),0,B
130 LOCATE 13,11:PRINT"江拖发动机厂计算机室"
140 LINE (75,141)-(245,158),1,B
150 IA=ABS((IX1-IX2)*(IY1-IY2))
160 IS=(36400!-IA)/360+37:I=I+1
170 IF I/5=INT(I/5) THEN SOUND IS*10,3
180 SS$=INKEY$
190 IF SS$=" " THEN GOTO 70

```

#### 五、应用软件封面的简单制作

利用 BASIC 语言的动画技术设计一个应用软件的封面,能给计算机的屏幕增加美感,悦耳的乐曲令人精神振奋,尤其能提高操作人员的使用兴趣。根据以上介绍的动画技术,任意选择其中的动画技术加以组合,很容易设计出一个图文并茂的画面。以下提供一种简单制作方法:

1、首先利用 BASIC 语言丰富的绘图和音乐语句以及动画技术,绘制一幅应用软件封面(程序实例五 FM.BAS),该程序的特点和有关说明如下:第 30 行为歌曲《血染的风采》过门,采用背景演奏方式,使演奏和绘图同时进行,实际曲谱在 380~460 行;第 40 行将高分辨率下的字符串 YS2\$ 存入数组 JS,在 270 行中分辨率下显示;第 60~90 行绘制底图,可调整步长改变图形和速度;第 120~210 行绘制计算机图案;第 280~350 行在计算机图案的小小屏幕内,随着乐曲的节奏不断涌现各种各样、颜色各异的“财务报表”。

该程序在运行时可随时按任一键中断退出,否则反复演奏乐曲和涌现各种报表。

2、使用汉字编译 BASIC 编译和连接成 FM·EXE 文件。

C) BASCOM/O FM (使用 0 参数,以便 EXE 文件可脱离 BASRUN)

C) LINK FM; (连接成 FM.EXE 文件)

3、编制批处理文件

c) TYPE AUTOEXEC.BAT

ECHO 有 OFF

FM

DBASE<文件名>

值得注意的是,由于编译后的程序和解释 BASIC 下的程序执行时速度有一定的差异,各种计算机的时钟频率不同,故需根据实际情况调整程序,以使得封面图形变幻和乐曲协调一致。

10 REM IBMPC-BASIC 语言动画技术应用实例之五

```

20 DIM JS(100):YS$="江拖发动机厂":YS2$="

```

1991.1"

```

30 PLAY "MB T70 04g8g 16a16a8e16d16c4.
   03a804g8g16a16a8f16e16d204d4"
40 SCREEN 2;LOCATE 1,1;PRINT YS2$;GET(0,0)
   -(110,16),JS
50 SCREEN 1,0;COLOR 0,2;CLS;KEY OFF
60 FOR X=0 TO 319 STEP 5;LINE(160,100)-(X,
   0),2;NEXT
70 FOR Y=0 TO 199 STEP 5;LINE(160,100)-(319,
   Y),2;NEXT
80 FOR X=319 TO 0 STEP -5;LINE(160,100)-(X,
   199),2;NEXT
90 FOR Y=199 TO 0 STEP -5;LINE(160,100)-(0,
   Y),2;NEXT
100 FOR I=1 TO 50; CIRCLE(150,90),60-I,0;NEXT
   I
   110 PAINT (151,90),0,0
120 LINE(114,59)-(184,101),1,B
130 LINE(113.5,58.5)-(184.5,101.5),1,B
140 LINE(121,63)-(161,98),1,B
150 CIRCLE(172,70),2,,,1; CIRCLE(172,80),
   2,,,1
160 LINE(100,105)-(200,125),1,B
170 LINE(99.5,104)-(200.5,125.5),1,B
180 LINE(136,108)-(192,122),1,B
190 LINE(136,115)-(164,115),1
200 LINE(164,108)-(164,122),1
210 FOR I=1 TO 20 STEP 2;LINE(104+I,144)-
   (104+I,120),3;NEXT I
220 LOCATE 3,6
230 PRINT CHR$(14)"[@M-40|10^6*1 会计核算
   管理系统]"
240 LOCATE 14,9;PRINT "江"; LOCATE 14,13;
   PRINT"拖"
250 LOCATE 14,17;PRINT "发"; LOCATE 14,21;

```

PRINT"动"

```

260 LOCATE 14,25;PRINT "机"; LOCATE 14,29;
   PRINT"厂"
270 PUT(125,175),JS;GOTO 350
280 IX1=RND*250+35;IX2=RND*250+35
290 IX2=(IX1-IX2)/3+IX2
300 IY1=RND*110+55;IY2=170
310 LINE (IX1/7.5+120,IY1/6+65)-(IX2/7.5+
   120,IY2/6+65),RND*2+1,BF
320 LINE (IX1/7.5+120,IY1/6+65)-(IX2/7.5+
   120,IY2/6+65),B
330 IA=ABS((IX1-IX2)*(IY1-IY2))
340 IS=(36400!-IA)/360+37:I=I+1
350 READ DD$;IF DD$="G." THEN RESTORE 40
360 PLAY DD$;SS$=INKEY$
370 IF SS$=" " THEN GOTO 280 ELSE SYSTEM
380 DATA d8.,e16,o3,b8,a16,g16,e8,g8,a8.,o4,
   e16,o3,b16,a16,g16,a16
390 DATA o4,c2,o4,e8,e16,e16,e8,o3,b16,a16.,g4,
   e8,a8,a16,o4,c16,o3,b16
400 DATA a16,g8,a2,o3,a4,o4,c8,o3,a8,o4,a8,e16,
   d16,c8.,o3,a16,o4,d8,d16
410 DATA e16,f8,e16,d16,d2,o4,e8,e16,e16,e8,o3,
   b16,a16,g4.,e8,a8,a16,o4
420 DATA c16,o3,b16,a16,g8,a2,o3,a4,o4,c8,o3,
   a8,o4,a8,e16,d16,c8.,o3,a16
430 DATA o4,d8,d16,f16,e8.,d16,c2,o4,g8,g16,
   a16,a8,e16,d16,c4.,o3,a8,o4
440 DATA g8,g16,a16,a8,f16,e16,d2,o4,d8.,c16,
   d8,d16,e16,g8,g16,a16,g8,e8
450 DATA o4,g8,g16,a16,a8,f16,e16,d2,o4,d8.,
   c16,d8,d16,e16,g8,g16,a16,g8
460 DATA e8,d8.,e16,o3,b16,a16,g16,a16,o4,c8,
   c4.,o4,d4,d8,d8,a1,g1,G

```

(上接20页)方式输入汉字,转入汉字缩放和选择填充模式菜单,命令基本同E,Y结束,再进入图形移动菜单,Y结束。L装入图形或图形块后进入图形移动菜单。M块移动。N清屏。P屏打。S存盘,按屏打的范围全部存盘。T读一个图形符号,命令同H。W块存贮,范围自选。Q退出程序,退出前要询问存盘否。

有兴趣者可在本程序的基础上开发彩色绘图程序,将更加丰富多彩。本文只起抛砖引玉的作用。

**参考文献:**“一个用于汉字字幕的图形编辑程序”,《计算机应用研究》,1990年第五期

需此软件(程序清单和软盘)者,请同作者或本刊联系。

## DBMS 数据库汉字排序技术探讨

四川大学博物馆 张小朋

四川大学历史系 董 敏

数据库技术在我国被广泛应用,尤其是在 dBASE (I、II、PLUS、N 及 FOXBASE+) 系统下建立了众多的数据库,使大量的日常手工工作实现了自动化。在情报检索、文档、财会管理工作中,频繁地使用排序技术,使数据记录按一定的规则顺序排列。dBASE 系统提供的 SORT 和 INDEX 命令仅能实现西文字符的顺序排列,对汉字记录的排列是按照《通讯用汉字符集(基本集)及其交换码标准》(GB2312-80)顺序进行的。GB2312-80 共 6763 个汉字分为两级,一级 3755 个汉字按拼音顺序排列,二级 3008 个汉字按偏旁部首排列,其编码规则不一致,因此单纯使用 SORT 和 INDEX 命令对汉字排序的结果,既不是音顺排序,也不是笔画排序,不符合中国人对汉字的排序习惯。目前国内流行的汉字操作系统和字处理软件未提供适用的汉字排序手段,使大量的汉字排序工作不得不降为手工操作,当数据量较大时,其繁难程度可想而知。汉字排序主要有音顺和笔画两种,这里主要介绍最常用的笔画排序技

术。

西文字符的排序实际是按 ASCII 码进行的,而计算机中表示汉字的机内码是一种高位为“1”的两字节内码,不含笔画信息,因此要实现汉字的机器排序,首先必须构造一个汉字笔画基本库。我们选取 GB2312-80 作为汉字基本库,对全部 6763 个汉字和部首进行了笔画编排,形成了 BHBASE.DBF。其中共有 3 个字段,WORD 字段是 6763 个汉字和部首, BH 字段是每一汉字的笔画数, QB 字段是每一个汉字的起笔顺序。当笔画数相同时,将按起笔顺序排列。汉字笔画数和起笔顺序依据《汉字正字手册》的规则,起笔顺序是“一(横),丨(竖),丿(撇),丶(点),乙(折)”。在笔画基本库中还增加了全角字符“0~9”,其排序优先级高于汉字。有了笔画基本库就可以实现汉字的笔画排序,用户还可以随时追加库中没有的汉字和其笔画数及起笔顺序。并按 WORD 字段形成索引文件 Ma, ilword. IDX。

欲排字库	
字段 1...字段 M	
1 北京	.....
2 南京	.....
3 成都	.....
4 上海	.....
.	.
.	.
N .....	.....

复制并从预制的 SSSSS1.DBF  
中追加 BH1、QB1、BH2、QB2 四个字段

字段 1...字段 M BH1 QB1 BH2 QB2	
1 北京	.....
2 北京	.....
3 南京	.....
4 上海	.....
.	.
.	.
.	.
N .....	.....

在 BHBASE.DBS 中查询排  
序关键字笔画数和起笔顺序

排序生成字库	
字段 1...字段 M	
1 上海	.....
2 北京	.....
3 成都	.....
4 南京	.....
.	.
.	.
N .....	.....

0 按后四个字段排序,并删除后四个字段

字段 1...字段 M		BH1	QB1	BH2	QB2
1 北京	.....	5	2	8	4
2 南京	.....	9	1	8	4
3 成都	.....	6	1	10	1
4 上海	.....	3	2	10	4
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
N .....	.....	...	...	...	...



为了简化叙述和节省篇幅,以下仅涉及一个字段和前两个汉字的排序问题。排序基本原理是先将欲排序库复制成另一个库,并在其中追加4个字段,分别表示第一、第二个汉字的笔画数和起笔顺序,然后在笔画基本库 BHBASE. DBF 中搜索这两个关键字,将其笔画数和起笔顺序加到这4个字段中,用 SORT 命令根据这4个字段排序,并删除附加的4个字段,其结果就是按笔画顺序排列的汉字记录。

值得注意的是,在 Foxbase+ 下使用 SORT、INDEX 命令对较大数据库(长度超过 64K)进行操作时,系统可能出现死锁现象,这一般是由于系统运行环境的内存配置不合理所引起的,因此在进入 Foxbase+ 之前在根目录下生成一个包含以下内容的 CONFIG. FX 文件:

```
C)COPY CON CONFIG. FX
```

```
bucket=2
```

```
buffers=20
```

```
maxmem=20
```

```
pcache=10
```

```
^ Z
```

本程序包含主控、排序和笔画库追加记录以及一个过程文件四个部分,和 BHBASE. DBF、SSSS1. DBF 构成了完整的汉字笔画排序系统。程序中考虑了对一般常见错误的处理,由于通常不需要对笔画基本库修改,所以未给出这一功能。本系统需要 Foxbase+ 或 dBase+ 支持,并在 XT、AT、GW、LC 等机型上通过,在 286 机上对有 8600 条记录的数据库用此程序排序耗时约 21 分钟。

我们在上述基础上,完成了能根据三个字段、前四个汉字并采用下拉式菜单的全套汉字排序系统(包括笔画排序和拼音排序),投入实际运用后,极大地提高了汉字排序效率。

需本软件(含盘)者,请同本刊读者服务部联系。

(上接 40 页)逻辑图如图四所示。1~9、11 是输入端,12~19 是输出端,另外,13~18 又可作为反馈线的输入,因此 GAL16V8 中的 16 表示最多可有 16 个输入,8 则表示输出数目,另外,GAL16V8 有 8 个可编程的输出逻辑宏单元 OLMC,使得每个输出都可根据需要进行配置,配置的工作原理已在上文作了介绍。

GAL16V8 器件工作原理与图三所示意的完全一致。使用者通过 GAL 开发软件和 GAL 编程器确定出符合某种功能的熔丝图和 JEDEC 文件。JEDEC 文件被用作从开发的计算机环境到器件编程器之间的传输介质。将正确的 JEDEC 文件传送到 GAL 编程器上,通过编程器对 GAL 器件编程,就可以得到“与”阵列已被编程的信息,这时的 GAL 芯片就是符合某种设计功能的芯片了。

图四 GAL16V8 逻辑图

### 三、GAL 器件编程机理

通过对 GAL 器件的开发软件初步分析可知,对 GAL 器件编程的过程分两个步骤来实现:先是将 JEDEC 文件进行数据压缩,然后从计算机向 GAL 器件进行数据传送。

在数据压缩过程中,首先判断 GAL 芯片类型,将类型标志存入内存单元,该类型标志表示了芯片“与”阵列的输入线数量。例如 GAL16V8 是 32 条输入线,则类型标志为 20H。八个信息位按规律压缩为一位,经过多重循环,将  $64 \times 32 = 2048$  位熔丝点的信息压缩至 8

$\times 32 = 256$  位,82 位结构控制字进行同样的压缩,由于 82 不是 8 的整数倍,编程软件自动在该控制字前补上 6 个“0”,然后压缩。压缩后,数据存放在另一处连续的内存单元中。

数据准备好后,开始向 GAL 器件传送。传送时,先传送对芯片编程的指令,然后传送 256 位编程信息,数据之后是结构控制字。传送方式采用 BIOS 14H 号中断调用。每传送一字节的数据之后,编程软件都要测试编程器状态,以保证 GAL 器件的正确性。

当数据传送完毕,“GAL 器件编程完毕”的信息返回计算机,对 GAL 器件编程过程即告结束。

### 四、结束语

通过上面的分析研究,我们可以看到 GAL 器件设计合理,具有快速、集成度高特别是用户可定义其逻辑功能等特点,因而能适应各种应用的需要,GAL 器件的使用对微电子、计算机、自动控制,系统逻辑设计等领域有着极大地推动作用。

### 参考文献

- [1] 夏仁霖,“PAL 的发展及设计”,《电子技术》,1989, No. 2.
- [2] 叶晋达,尹周水,“可编程阵列逻辑(PAL)技术”,《微电子学与计算机》,1989, NO. 3.
- [3] 丁嘉种,刘凤云等,“可编程逻辑器件 PLD 基本原理·设计技术·应用实例”学苑出版社,1990. 3.

## 在单色汉字方式下的绘图软件

56025 部队计算所 吴正明

近年来,随着微型计算机的日益普及,各式各样的计算机绘图软件给广大计算机用户带来了极大的方便。但这些软件大多只能在彩显上运行,使众多的单色显示器用户只能将计算机用于字符处理,为使单色显示器充分发挥其特长,本文介绍一种在单色汉字方式下的绘图程序,它能在单色显示器上绘出生动优美的图形,其效果并不亚于 CGA 彩色显示器。单色显示器无颜色变化,但它有  $720 \times 350$  个像素点,比 EGA 彩色方式的像素点还多。可充分发挥其高分辨率的优势,利用点的变化来表现图形的层次。这对绝大多数无彩色打印机的用户制作“黑白图形”带来极大的方便。有一定的实用性。

### 一、功能及实现方法

程序用 TURBO PASCAL V4.0 编写,以菜单方式提供十种绘图功能,不懂计算机的人只要几分钟就能学会使用。

1、画图 E,程序主要调用 TURBO PASCAL 提供的 BAR 过程,在屏幕可视范围内画出任意大小的正方形,用任选的十一种填充模式填充矩形,矩形的大小和填充模式及每次移动步长可非常方便直观地任意调整。每次按键都有明确的提示。在绘制直线时,可将步长与笔粗调整到同样大小,在绘曲线时,为增加图形的美感可将步长调小,在特殊情况下,可将步长调到大于笔粗,出现断续的线条。每次移动后屏幕上都将显示笔粗左上角处的坐标,对于绘制一些精确度很高的图形非常有用。设有擦除功能,方法同画时一样。

2、调用汉字和图形字符 H,T,绘图时需要汉字和图形符号作注释,并且还要求有一定的变化和美感,本文主要考虑绘图为主,只调用了  $16 \times 16$  点阵的显示字符,经变化和放大之后效果也很好,汉字可横向和纵向任意放大、缩小,并可选择汉字和图符的显示模式,有十一种模式。在选择过程中字符能随选择的笔式进行变化,非常直观。如需要  $24 \times 24$  点阵的字模只要稍加改进就能实现。

3、活动方框取图形范围,本程序不是简单地绘出 XY 坐标,而是采取非常直观灵活的方框取范围的方式。充分利用了绝大多数单色显示器都配用的大力神单色适配卡(Hercmono)具有两页这一特点,变化的方框先在非活动页面上绘好,存入缓冲区,再显示在可见的活动页面上,直到方框的大小和位置都满意为止。方框可以任意步长向四个方向扩展和移动,直到按下 Y 键。方框内框住的图形为选定的范围。

4、图形块移动,方法与 3 相同,只是移动的图形块本身不能再变动,只有位置的变化。移动的图形块以 XOR 方式选加到原图形上,移动后再消去,非常直观

有趣。

5、图形块移动、拷贝,存贮和装入,这是一个完整的绘图程序不可缺少的功能。本程序在这些功能上就显得更加灵活,直观。几项功能的配合使用对制作幻灯片图、动画片、连环画及各种工艺图案等都非常有用。对经常出现的图形可反复调用,或稍加修改而用之,避免了很多重复劳动。程序运行本身就是一幅丰富多彩,具有动感的表演。

6、屏幕打印,考虑到不同的 CCDOS 操作系统屏打中断 INT5 和字符打印中断 INT17 的功能不一定相同,本程序使用了 MS DOS 中的 INT17 类中断来实现屏打。屏打范围从 16 线到 320 线止,打印机以 MX80 型九针打印机为例,其它机型稍加修改就能使用。

7、要实现汉字方式下的绘图,在 TURBO PASCAL 中不能调用 CRT 标准单元,在用有关 CRT 函数和过程时,可自己编写相应的过程,如本程序中的 gotoxy、cls、inkey 等过程都是自己设计的。在不绘图的 PACAL 程序中可以调用 CRT 和 GRAPH 两个标准单元,只要将 directvideo 变量设成 false。如不需 CRT 和 GRAPH 中的函数和过程,PASIC 程序不作任何处理就可直接使用汉字,方法同英文字符。

### 二、使用说明

在 CCDOS2.0 以上版本支持的单色显示器上运行,CCLIB 文件放在 C 盘的 CCDOS 目录下,情况不符者需修改相应的路径。显示器为  $720 \times 350$  的大力神单色字符图形驱动卡。

主菜单下的命令介绍:

C 块拷贝,E 绘图,子菜单中的命令为:0-9 选择填充模式,E 结束,W 画点,C 擦除,A 实填充模式,M 移动光笔,+、- 增加缩小步长,小键盘上的 +、- 为加粗、缩小光笔,↑ ↓ ← → 键为向上下左右移动光笔,Home End、Paup、PaDn 键为左上、左下、右上、右下方向移动光笔。H 读取一个汉字,可用任意(下转 17 页)

## 破译加密程序实例

上海市民用建筑设计院 茹建平

加密技术可分为数据加密、文件加密、磁盘加密及硬件加密。

数据加密是通过算法将明文转换成密文,使程序不可读,待执行时,由程序内部自身解密成可处理文件。文件加密出发点是首先保证程序不可复制,其次在程序内加入各种反跟踪措施。这包括改变中断入口、限时法、检测跟踪、堆栈技巧的应用、控制时序等手法、磁盘加密思想是利用非标准格式的磁盘或 CRC 校验值以及激光加密技术来实现。硬件加密包括基于磁盘的硬加密和非基于磁盘的硬加密,基于磁盘的硬加密如激光加密等。非基于磁盘的硬加密是利用数据加密/解密器,将密文印记做在硬卡上,使加密者极难复制,高级的加密硬卡上带有自身一套硬件系统,使密文转换成明文的过程在硬卡内完成,使解密更困难。

加密手法的不断改进,无疑给解密者带来了越来越多的困难,破译一个加密软件需要花更多的时间,采用更多的方法,具备更广更深的系统知识。

从解密成功的范例来看,破译加密程序,必须首先了解加密的大致几种方法,了解加密者设下每一陷阱的用意,所以一个好的解密者,应是一个好的加密者。

笔者曾对用于 AUTO CAD 的程序 DWGKEY. EXE 进行了跟踪。程序中采用了数据加密、磁盘防拷贝措施,包含了三种反动态跟踪手法。我们知道,任何加密软件,无论它的反拷贝技术如何安全,只要在反跟踪方面稍有漏洞,就很容易被解开。因此,加密软件主干部分、也是最显水平的是装备有若干种鲜为人知的好的反跟踪手法,一个好的反跟踪手法应具备①译破段是不可跳越的,不执行译码程序,程序就无法执行。②译破程序是不可被跟踪执行的,又是复杂的、隐蔽的,若跟踪,程序就无法执行。③不通过译码算法,密破是不可破译的。下面介绍 DWGKEY. EXE 的反跟踪手法。

反跟踪手法一。破坏 BIOS 的 INT1 和 INT3 中断处理程序入口,指令如下:

```
XOR BX, BX
PUSH BX
POP DS      ;设置 DS=0000
MOV BL, 04
MOV WORD PTR [BX], BX;破坏 INT1
MOV BL, 0C
```

```
MOV WORD PTR [BX], BX;破坏 INT 3
PUSH CS
POP DS
```

这段指令完成破坏 DEBUG 跟踪命令 T 和 G 的功能,因为每次键入 T 和 G 命令时,总要重新设置 1 号与 3 号中断的返回地址,若有改变返回地址的指令将引起中断转向错误。从指令段中看出,1 号中断向量处填入了 04,3 号中断向量处填入了 0C,达到了破坏目的,如果忽略了这段指令,将引起死机。

反跟踪手段之二。为了防止解密者打印出反汇编程序作静态分析,程序采用了暗码形式,解密者无法读通程序,也就无法正确设置断点,实施跟踪。

反跟踪手法之三。上述二种手法对于涉足解密领域不深的程序员有一定作用。但有解密经验的程序员可通过设置跳转指令,跳过破坏中断入口的指令,然后循着程序译码流,进行暗码自解过程,译破段指令如下:

```
MOV CX, 21      ;计数器
MOV AH, BYTE PTR [SI+03]
INC DX
PUSH DX
XXXX:39C3 LODSB
XOR AL, AH      ;译码算法
STOSB           ;恢复代码
LOOP 39C3
```

上述暗码自解后,就能进行钥匙盘的处理。所以仅有上述二手法是不够的。为了给程序员增加难度,程序中使用了颇为流行的穷举法,即程序中隐含了大量的破坏中断处理的指令,且都以暗码方式存在,不经过前段程序的执行,后段的暗码就无法恢复成明码,若恢复成明码后,又需对相对后继的暗码进行解码,而这种形式在程序内占了百分之九十以上。且每段中的破坏指令又迫使你必须小心从事,稍有不慎,就会前功尽弃。因此,破解穷举法无疑是对解密者信心与技巧的考验。

了解了程序中所采用的反跟踪手法后,就能针对性的破解。

针对第一种反跟踪手法——改变中断处理程序入口。可利用实用程序 CODEVIEW 进行跟踪:

```
C>CV DWGKEY. EXE
```

用 U 子命令查看指令,起始指令是一段明码,中间嵌有破坏指令:

```
MOV BL, 04
MOV WORD PTR [BX], BX
MOV BL, 0C
MOV WORD PTR [BX], BX
```

由于填入中断向量处的数据在以后的执行中,不加引用,所以可将这段指令改为空指令。但进一步观察,此类破坏指令在每一段的暗码转明码后都存在,且填入中断向量处的数据都相同。换言之,经重置后的 INT 1 与 INT 3 中断处理程序入口均相同,因此可考虑在修改了的中断入口,填入跳转至正确中断入口的指令。正确的 INT 1 与 INT 3 处理程序入口,可在第一次执行 T 命令后,从内存处 0000:0004~0000:0007 与 0000:000C~0000:000F 获得。INT 1 正确入口设为 4C4D:113C, INT 3 正确入口应为 4C4D:1131。而破坏指令执行结果,将使 INT 1 处理程序入口置为 4C4D:0004, INT 3 的处理程序入口置为 4C4D:000C,弄清了处理程序入口改变规律后,就有办法避开它了。

以 INT 1 为例,其中断入口改变后,每次发生单步中断,执行完必要步骤,将转入 4C4D:0004。考虑人为地将 4C4D:0004 处改成跳转指令 JMP 4C4D:113C。这样修改后,单步中断就能转入正确入口。INT 1 的陷阱就被避开了。对 INT 3 可采用同样方法。经过上面的处理后,程序员已可放心地用跟踪命令实施跟踪了。

前面已经介绍穷举代码占程序长度的百分之九十左右,其中任何段密文都需前段程序执行转为明文。程序的断点很难正确设置,只能采用连续单步跟踪或局部明文断点跟踪。四十多 K 的暗码如用单步跟踪,需不间断地几天几夜,为了减轻程序员工作量,应设法使程序能分段跟踪,中间允许退出,执行过的程序段,下一次不再执行。这就需要保留每次结束跟踪时的寄存器值,并且在合适时机中断一天的工作。

假设 DWGKEY. EXE 装入时 IP 为 0000, CX 为 E800, BX 为 0, 目前已运行到如下状态,并准备退出跟踪,为以后的跟踪做准备。

```
37E6,5293 POP DX
37E6,5294 JMP 5298
37E6,5296 ADD BYTE PTR[BX+DI], DH
37E6,5298 XOR BX, BX
37E6,529A PUSH BX
37E6,529B POP DS
37E6,529C MOV BL, 04
37E6,529E MOV WORD PTR [BX], BX
```

```
37E6,52A0 MOV BL, 0C
37E6,52A2 MOV WORD PTR [BX], BX
37E6,52A4 PUSH CS
37E6,52A5 POP OS
37E6,52A6 MOV CX, 21
37E6,52A9 MOV AH, BYTE PTR [SI+03]
37E6,52AC INC DX
37E6,52AD PUSH DX
37E6,52AE LODSB
AX=48FA BX=000C CX=0000 DX=01DD
SP=FFFE BP=0000 ST=52B4 DI=52B4
DS=37E6 ES=37E6 SS=37E6 CS=37E6
IP=5293 ;寄存器值
NOVRFLOW UP DISABLE NEGATIVE NOT
ZERO;标志位
```

恢复断点寄存器值的指令如下:

>A 0000 ;装入程序时记下的 IP 值

```
37E6,0000 MOV AX, CS
MOV ES, AX
MOV DS, AX
MOV SS, AX
MOV AX, 48FA
MOV BX, 000C
MOV CX, 0000
MOV DX, 01DD
MOV SP, FFFE
MOV BX, 0000
MOV SI, 52B4
MOV DI, 52B4
JMP 5293
```

这段指令的功能,就是用于保留至目前为止的运行状态,下次跟踪就可在此基础上进行,而不需从头开始。

接下来的工作是:在 CV 命令级修改 CX, BX 值(恢复装入的文件长度),命名文件、记盘,重新装配文件头部信息模块。

通过几次这样的处理,穷举段执行完,进入校验钥匙盘阶段。这段指令只有一些磁道变形校验等,很容易读通,找到正确的出口地址,整个加密软件就被解开了。

加密手法多种多样,解密途径也不是唯一的,本文提供的方法也仅对某些加密软件有用,但只要花功夫,有耐心,加密程序是能够破解的。



# 如何灵活应用设置检索目录 PATH 命令

浙江省嘉兴市第二毛纺织厂 张伟中

**摘要** 本文简要介绍了使用设置检索目录 PATH 命令的技巧。

## 一、设置检索目录 PATH 命令的用途

该命令主要为 DOS 建立一张检索目录顺序表,当 DOS 调用的文件在当前目录中找不到时,就可按当前目录检索表中的目录顺序依次查找,直到找到为止。若找不到所找的文件,则显示出错信息。

PATH 命令仅仅对 DOS 的可执行文件(扩展名为 .BAT、.EXE 或 .COM)进行检索,而不理睬其它扩展名的文件,命令执行后也不改变当前目录。

## 二、设置检索目录 PATH 命令的应用

1、可将 DOS 的显示外部型命令当作隐含内部型命令执行。

由于 PATH 命令具有指定一串附加路径的特性,所以我们将一些常用的 DOS 外部型命令文件拷入到某一个子目录中,然后再用调试程序 DEBUG 或 PCTOOLS 将该子目录的属性改为隐含属性,这样当用目录命令 DIR 列文件目录时,该子目录将不会被列出。当其它目录要使用该子目录中的命令文件时,只要输入一条“PATH\子目录名”(命令文件所在的子目录),就可以使用该子目录中的任何一个命令文件。或者将“PATH\子目录名”这条命令放到根目录批处理文件 AUTOEXEC.BAT 中,这样当计算机启动或重新启动后,就可在根目录或任何一个子目录中使用该子目录里的命令文件。

具体使用步骤如下所示:

①在硬盘中建立一个子目录,任意起名为 ZWZ。

C>MD ZWZ (回车符)

②用 COPY 命令将一些常用的 DOS 外部型命令文件拷入到该子目录中。

C>COPY A: \* \* \* \ZWZ

③用 DEBUG 或 PCTOOLS 将子目录 ZWZ 的属性 10 改为 12(12 为子目录隐含属性)。

④在根目录批处理文件 AUTOEXEC.BAT 中增加一条“PATH \ZWZ”命令(或直接输入“PATH \ZWZ”命令)。

⑤计算机重新启动后,便可在硬盘根目录或任何一个子目录中使用子目录 ZWZ 内的命令文件,并且子目录 ZWZ 为隐含。

2、可使硬盘中各子目共用一个系统文件

利用 PATH 命令还可以使硬盘中各子目录共用一些常用的系统文件,如:DBASE II、BASICA、WORDSTAR 等。

具体使用步骤如下所示:

①在硬盘上建立一个子目录,任意起名为 FYJ。

C>MD FYJ

②将常用的系统文件拷入到 FYJ 子目录中

C>COPY A: \* \* \* \FYJ

③在根目录批处理文件 AUTOEXEC.BAT 中增加一条“PATH \FYJ”命令(或输入一条“PATH \FYJ”命令)。

④计算机重新启动后,就可以在硬盘根目录或各子目录中使用子目录 FYJ 中的系统文件。同样,若将子目录 FYJ 隐含起来,则用户使用子目录 FYJ 中的系统文件就象使用 DOS 的内部型命令一样。

## 三、结束语

灵活使用 PATH 命令,不仅可使硬盘中重复文件减少到最少,减少占用硬盘空间,而且可以使一些常用命令文件和系统文件单独放在一个子目录中,便于用户操作使用。

(上接 25 页) 采用 ORACLE 数据库,研究和开发了一个大型林业数据库应用系统。最初采用高级语言作为原型开发工具,在编制报表程序上花费很多时间和精力,修改很不方便。后来改用 SQL \* REPORT,大大提高了系统开发速度和效果。同高级语言比较,采用 SQL \* REPORT 制作统计报表,无论从结构上、可移植和可维

护性、工作量上以及所得到的效果等方面都是非常好的。(需要程序者请同本文作者联系)

## 参考文献

- [1]SQL \* PLUS user's Guide 美国 ORACLE 公司
- [2]SQL \* Report user's Guide 美国 ORACLE 公司

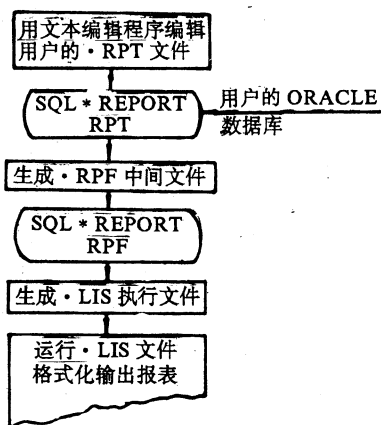
# 用 SQL \* REPORT 设计统计报表

四川省林科院计算中心 曾明蓉

成都科技大学计算机系 曾文方

**摘要** 本文在简介 SQL \* REPORT 系统支持工具的基础上,推出了一个统计报表实用程序。SQL \* REPORT 是 ORACLE 关系数据库系统中,众多的系统支持工具之一。专门用于制作电子邮件,数据统计报表等,是办公自动化企业管理的重要工具。

SQL \* REPORT 由两个程序组成:RPT(报表生成程序)和 RPF(报表格式化程序)。前者通过 SQL 语句提取数据库信息,后者根据正文中所包含的命令对这些信息进行格式化,并生成所希望的格式报表。图一所示,给出了 SQL \* REPORT 的运行流程。



图一 SQL \* REPORT 运行流程

图中:

(1) . RPT 文件:用户编制的 ASCII 文件,又称控制文件。它包含有用户正文信息、宏定义(调用)语句,SQL 查询语句以及 SQL \* REPORT 的有关定义、RPF 命令等。

(2) . RPF 文件:用 RPT 程序扫描编译控制文件(. RPT)后,生成的中间文件。它包含有从数据库检索出的数据信息,用户正文信息以及 RPF 语句。

(3) . LIS 文件:用 RPF 程序扫描编译中间文件(. RPF)后,生成的最终报表文件。

注意:这里的 . RPF 和 . LIS 两文件仍可使用文本编辑程序进行修改。

## 一、SQL \* REPORT 语句简介

SQL \* REPORT 语句分为三大类:RPT 语句和 RPF 语句。

(一)RPT 语句:有六种类型,用‘.’号打头表示。

1.说明语句:用在控制文件中,对数据、程序变量进行说明、赋值及初始化。它们是:

- DECLARE(程序变量名)(格式)
- SET(程序变量名)(字值)
- EQUAL(目的变量)(源变量)

2.宏定义语句:宏定义语句分两大类,即 SELECT 宏和过程宏,它们可用同一方式定义,格式为:

- DEFINE(SELECT 宏名)
- 或(过程宏名)
- 宏 体
- ..

其中,宏名是用户自起的。SELECT 宏体仅包含 SQL 查询语句;过程宏体中可含有 RPT 语句,用户正文和 RPF 语句以及其它的宏。SQL \* REPORT 的宏类似於高级语言中的过程(或子程序),仅当主程序调用它时才执行。

应当指出:一个宏不能在另一宏中定义,但它可以 在另一宏中被调用。

3.宏执行语句:有如下三类,它们的格式为:

- (过程宏名)
- EXECUTE (SELECT 宏名)(宏体)
- [(<头宏>)[(<足宏>)]]

当然,一个宏必须在其执行之前於程序中定义。

4.过程控制语句:用来控制程序的执行流向,如分支、无条件转向以及停止程序执行等。它们的格式为:

- &(标号名)
- GOTO (标号名)
- IF“(表达式)”THEN(标号 1)[ELSE(标号 2)]
- STOP

5.算术语句:RPT 提供在两个程序变量之间执行 ADD(加)、SUB(减)、MUL(乘)、DIV(除)的语句,运算结果存贮在第三个变量中。例如:

• ADD<目的变量><源变量 1><源变量 2>

6、其它语句:如 PRINT 语句、ASK 语句、TELL 语句、REM 语句等,它们的格式为:

```

• PRINT<程序变量名>[<变量名>]...
• ASK '<信息>'<变量>
• TELL '<信息>'
• REM<注释正文>
• COMMIT

```

(二)RPF 语句:主要用来完成报表的格式化,定义表,控制左右边界对齐、自动换页、控制页的顶和底边界、页数累计,控制修改打印机参数等。RPF 语句是用 '#' 号打头表示。下面列出主要的 RPF 语句和它们的功能。

1. #DT 定义表格输出格式,指定表的左右边界,每字段的起止位置。

2. #CUL 带不划线并将随后的正文放在一列的中央。

3. #PAUSE 暂停打印,等待操作员响应。

4. #PAGE 定义页的顶部、底部边界。

5. #NC 控制移到下一字段。

6. #NP 控制换页

7. #NS 控制跳过 n 行

8. #B 控制插入一空行

9. #T 调用指定的一张表

10. #TE 终止表

11. #L 对随后的正文行停止格式化,忽略列定义。

12. #F 在输出报表中为图表指定要保留空间的页数。

## 二、控制文件(·RPT)的编写方法

根据 RPT 语句在控制文件中所起的作用,可把控制文件分成程序说明、宏定义和主程序三大部分。现简介如下:

1. 程序说明部分:这部分包含了用说明语句 DECLARE、SET、EQUAL 对数据、变量等进行定义、赋值和初始化。具体用法请见后面例子。

2. 宏定义部分:有两种类型宏定义

1)SELECT 宏:在一个·RPT 文件中,可以定义多个 SELECT 宏,这些宏中可以包含 SELECT 语句中任何合法的 SQL 子句或参数。一般必须是一个带 INTO 子句的扩充的 SQL 语句,该 SQL 语句执行成功时,将检索到的数据库表中的相应信息暂存到规定的变量中。INTO 子句中的变量必须和 SELECT 子句中的列或表

达式具有相同的数据类型,并事先用 DECLARE 语句说明。

2)过程宏:一般由 RPT 语句、用户正文和 RPF 语句以及其它宏等组成。在调用时,宏中的 RPT 语句被执行,而用户正文和 RPF 命令原封不动被拷贝到中间文件(·RPF 文件)中。在·RPT 文件中,常遇到的过程宏如头宏,体宏和足宏,它们分别与报表的头(表头)、体(表体)和足(表尾)相对应。详细情况,请见后面例子。

3)主程序部分:在这部分中,可使用宏执行语句 EXECUTE 和 REPORT 来调用所定义的 SELECT 宏和过程宏的自动执行,生成报表。REPORT 语句驱动数据检索,使得相应的报表头宏、体宏、足宏命令被执行。

通常,SQL \* REPORT 的控制文件以 STOP 语句结尾。

## 三、一个完整的 SQL \* REPORT 报表控制程序

这是一个实用程序,运行本程序生成悬移质输砂量年统计报表(略)。表中的全部数据是按年和站名通过检索 ORACLE 数据库(SW1 基表)得到的。下面对程序作一简要介绍。

图(略)中行 22 至 33 定义二个 SQL 查询(SELECT SUM,SELYMMAX),从 SW1 基表找出年最大(YMAX)值,计算年平均值(YAVE)以及出现年最大值的对应日期。行 34 至 40 定义 SELSFC SQL 查询,从 SW1 基表中选出各行中的所有列。行 41 至 98 定义体(XYZBODY)、头(XYZHEAD)和足(XYZFOOT)三个过程宏。行 100 的 REPORT 语句驱动数据检索,并使头、体、足宏隐式执行。XYZHEAD 宏只执行一次,产生报表表头,输出从 SW1 基表中检索出的第一行数据。执行 XYZBODY 体宏,输出从 SW1 基表中检索出符合条件的第二行至最后一行数据。在处理查询的最后一行后,执行 XYZFOOT 足宏,输出报表年统计数据和报表结尾。下面给出在 VAX 机系统命令下产生报表的过程:

第一步:执行 RPT

```
$ RPT XYZ·RPT XYZ·RPF 用户名/口令
```

第二步:执行 RPF

```
$ RPF XYZ LP
```

其中 'LP' 输出设备的系统名或表示一个文件名(·LIS)

## 四、结束语

近半年来,我们在 VAX3500 上 (下转 23 页)

## 印刷五线谱乐符的识别方法

哈尔滨船舶工程学院计算中心 吕良

近几年来,由于计算机技术的迅速发展,给音乐领域这个分支带来了很大影响。从音乐格式的分析 and 比较等统计处理,到自动编曲、作曲和自动演奏以及包括这全部的音乐信息处理系统的研究正在逐步展开。为此,音乐数据的计算机输入也在普通键盘、乐谱信息输入键盘的基础上,开始尝试着乐谱识别的研究。

五线谱的识别有其不利的因素:

(1)五线谱是包含了细长直线成分和微细构造记号的二值图象,除了五线谱特有的记号外,一般还包含了与五线谱识别无任何关系的汉字、外语和数字等,它们的位置、大小和格式是很难决定其归属区域的。

(2)从面向幼儿的乐谱到管弦乐,难易程度差别很大,很难决定处理对象。

(3)通常五线谱有很多页,有必要考虑到输入方法和识别处理单位。即使只有一页,也要考虑有意义的联系,决定处理单位。

另一方面,在识别中也有有利的方面:

(1)五线谱内部结构通常是一定的,形状一般都比较简单,把这作为知识事先储存在系统中,再从上到下进行处理。

(2)在五线谱信息中,可以根据音乐规则进行辅助识别和检查。

### 一、特征提取和分类

五线谱经五线和小节线的检出和删除处理之后,乐符就被孤立出来,可以进行实际的识别。但是,许多熟悉的模式识别技术不能直接采用。在标准乐符中,有一些符号的几何形状不定,如图1所示。

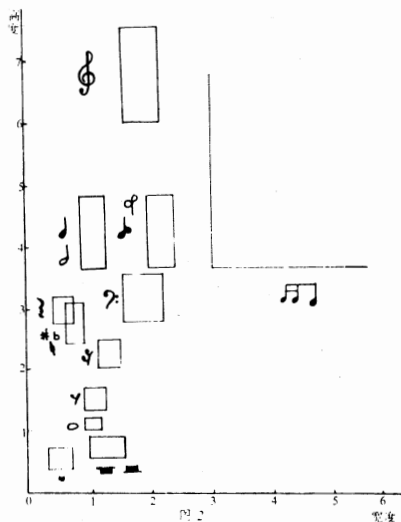


图1

在钢琴五线谱高音部有两个相同的符号,然而形状不同,因为要受五线谱低音部的影响。所以,对这类乐符常规的识别技术不是很有用的。本文提出了以小

节线为单位,利用乐符记号的特征和乐谱规则识别印刷五线谱,识别对象为钢琴乐谱。

乐符有很强的几何规则特性,首先利用乐符的几何特性进行分类,减少可能与未知符号相关的乐符集合。孤立地看乐符,每一种符号类型在高度和宽度都几乎不同于其它所有的符号。为了找到每种乐符的尺寸范围,要作标准乐符的测试。量度每种乐符的各个样本,将每种符号外接矩形的高度和宽度列成表。这种量度必须是归一化的,因为不同的样本有不同的比例。利用五线间隔的平均值作为归一化因子,用这个因子去除符号的高度和宽度值。这样转换成独立于绝对度量值的比值。每一种乐符量度归一化后,作归一化的高宽空间特性图。对每一种乐符都限制在某一个固定的区域内,这个区域共包含了该符号所有可能的高宽值的最小矩形区。乐符的空间特性如图2所示:



图中各个区域很少重叠,利用此图和乐谱规则对乐符进行分类。在五线谱中直接用于识别的任何信息,如标题、歌词等都在这里自动略去。同时,连接线、强弱符、装饰符等都不归为要识别的范围之内,在分类过程中自动地从中分离出来,并除去。

### 二、谱号、调号、拍号的识别

由乐理知识可知,这些符号限定了位置,即:

(1)谱号一定写在各段五线谱的开始;



(2)调号限制在谱号的后面或复纵线的后面;

(3)拍号一定写在曲子开始的五线谱号的后面,有时也接着复纵线写。

根据小节线位置或小节前的小节线种类,限定进行识别处理小节,使其后各小节对应于它的拍子、调子。

### 1. 谱号

谱号有两种:G—谱和F—谱。提取和五线有位置关系的谱号和被推定的连接成分,根据其外接矩形的大小(高、宽值)进行识别。需要说明的是,F—谱号即使通过其他测试也不能判别为F—谱号,除非两个其他图形识别为圆点,并且在F—谱号规定的位置为止。也有违反上述规则(1)的情况,如图3所示的第一小节和第八小节。



图 3

这样的谱号对五线的垂直方向位置不变,并且一定不同于在此之前的谱号,利用这个特点进行构造解析和识别。

### 2. 调号

调号由升号(♯)和降号(b)构成,根据其外接矩形的高度和宽度进行分类。由于♯和b的外接矩形的大小非常接近或相等,再由检查符号顶部的点来区别它们。对♯,最顶端的点发生在符号宽度右边三分之二处。对b,最顶端点在矩形左边三分之一处,见图4。

并且,由于调号的种类不同,写入的位置也不同,其位置规则如下:

①♯:F,C,G,D,A,E,B

②b:B,E,A,D,G,C,F

例如,对于G谱来说,如果第一个调号是b,则必须在五线的中线上。反过来,如果第一个调号在中线上,它一定是b。同样,如果第一个是b,若还有第二个

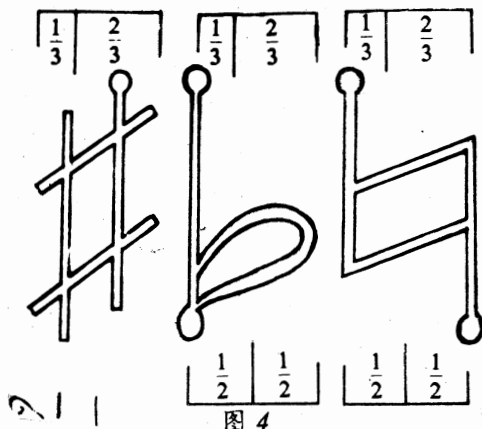


图 4

调号的话,一定在五线顶端空间上,且一定是b。如图5所示:



图 5

在上下每一段乐谱中,调号的描述是相同的,根据这个规则判别调号的种类和个数。

### 3. 拍号

拍号由分子、分母两个数构成,但这个数值被限定了。根据其外接矩形的大小和位置以及比较数字的交叉特征(数字笔道与指定直线相交的次数)进行识别。方法简单,同一般的数字识别类似,这里不再多述。

谱号、调号和拍号这些符号不表示其自身的音,而对后面的全部符号给予影响。它们所表明的信息不限制在此小节中,必须引入到其后的各小节中。

### 三、休止符、临时记号和音符的识别

这些乐符是表示音的基本符号,形状较单纯。但识别时有下列问题:

①频繁地出现和其它符号的接触、交叉、混入等,较难进行识别;

②五线谱中本来空白的地方有字,或五线消失时可能分割了本来连接的部分;

③因为细成分很多,可能出现符号中途断掉以致模糊不清;

④受其它符号的影响而改变其含意的情况。

①~③主要是图象处理问题,符号混入的问题可以在识别处理中加以考虑。④是意义分析上的问题,需要在识别中予以解决。

根据符号的外接矩形的大小和位置规则,将休止符、临时记号和音符分类。全休止符、二分、四分、八分、十六分休止符的区别首先根据其外接矩形的高、宽值,其次,每个休止符都有一个标准的垂直位置,并总在五线谱上占据同一个位置。十六分休止符( $\gamma$ )在五线的第四线和第一线之间;八分休止符( $\gamma$ )在第四线和第二线之间;四分休止符( $\gamma$ )在第五线和第一线之间;第三线在四分休止符中间;二分休止符在第三线上面;全休止符在第四线下面。如图6所示,根据这些规则识别休止符。

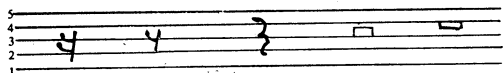


图 6

临时记号的外接矩形大小相近,必须再由检查符号的最高点和最低点来区别它们。#、b的判别同调号中的#、b的判别相同。#、b的判别:#的最高点发生在其宽度右的三分之二处,b最顶部的点在其宽度左边的三分之一处;b和b的判别:b的最低点发生在其宽度的左半边,b的最低点在其宽度的右半边,见图4。这种判别要求找到符号的最顶部的点和最底部的点的位置,这两点容易在乐符提取时用轮廓跟踪法得到。音符由于内部细微结构相对其他符号较为复杂,识别时先根据其外接矩形的高、宽值将其与其它符号分开,

然后对音符进行垂直投影,通过投影的峰的数目、位置将音符分段。再对各段进行水平投影,通过投影后的形状和所处位置检测音符符头和音高。

#### 四、结束语

我们以钢琴乐谱作为实验对象,目前识别对象号仅限于基本乐符,但这些乐符包含了所有重要的音乐信息(实际上,在许多应用中不要求完全识别)。今后识别对象的符号的扩展还是一个值得研究的课题。根据利用已经识别了的基本乐符信息以及把识别出来的符号从图象中消除,使剩余符号的识别处理比较容易。识别的时间由图象质量、五线谱的难易程度所决定。实验表明,识别的结果是令人满意的,乐符内部图形不好也能进行很好的识别。也有少数乐符被误识或拒识的情况,产生误识和拒识的主要原因是符号严重失真,或发生符号之间的接触。本识别方法有以下几个特点:

(1)方法简单,易于实现,识别速度快;

(2)具有一定的抗干扰和畸变能力;

(3)因在识别处理中交换各小节的信息仅限于有关的谱号、调号和拍号。所以,以小节为识别单位可以实现并列处理。

由于五线的识别技术还不是很成熟,对现有的识别方法进行的研究和改进还有待于进一步完善和提高。这些只是对印刷五线谱的识别进行的初步尝试。

#### 参考文献

[1] D. S. Prerav, "Computer pattern recognition of printed music" AFIPS Conference Proceedings, FJCC, Vol. 3, 153-162 (1971)

[2] 王积分等,《计算机图象识别》,中国铁道出版社,1988

[3] И. 斯波索宾著,汪启璋译,《音乐基本理论》,人民音乐出版社,1989,6

## 工具软件信息

当汉字编码设计完成后,最希望的莫过于亲手将自己的方案装入计算机系统。《通用嵌入式汉字编码系统生成软件》将为您实现这一愿望,并使你的编码方案获得最大的适应范围。

《通用嵌入式汉字编码系统生成软件》允许用户给系统定以不同的名称,并反映在汉字输入提示窗口中;可以转换四码长以下的各种汉字编码方案,包括 GB2312-80 中 6763 个标准汉字和二字词以上的编码,而且允许简码;用户能够通过本软件在一分钟内自动生成严格按 GB2312-80 顺序排列的 6763 个汉字文本供编码使用,并能自动对完成后的编码方案进行格式检查、纠错。本软件允许汉字编码方案的最终用户以个别和批处理的方式建立自己的“用户词组”文件,增加编码的适应力;同时最终用户可以利用本软件提供的功能使编码软件适应不同的汉字显示行数。

《通用嵌入式汉字编码系统生成软件》将不同的汉字编码方案自动生成操作软件,在完全不改变原汉字操作系统功能的前提下,通过热键自动嵌入或组合到 IBM 系列和兼容机的各种汉字操作系统中,如 CCDOS2. XX、CCDOS4. XX、GWBIO3. XX、CCDOS2. 13X、UCDOS、联想 DOS 等;生成的新编码操作软件不依赖于转换过程中的各种编码文件和转换系统本身,可由编码方案设计师提交最终用户使用,并且自适应各种分辨率和色彩的显示器以及软、硬汉字库,还具有万能键汉字输入外码替代功能,方便输入。

转让形式:软盘两张,含全套功能——为汉字编码设计者和为最终用户提供的功能。

系统报价:350 元。需此软件者,请同本刊联系。

# 微机通用信息管理系统 Micro CDS/ISIS 的应用技巧

吉林省农业科学院情报研究所计算机室 万发仁

吉林省四平地区农科学研究所情报室 张晓东

**摘要** Micro CDS/ISIS 是为管理结构化非数值数据库而专门设计的通用信息管理系统,它的设计思想是:面向非计算机人员。文中作者根据这一特点,经过对该软件的研究,总结出了部分简便的应用技巧和比较常见的几个问题及处理方法,希望对使用该软件的非计算机人员有一定借鉴作用。

## 一、概述

Micro CDS/ISIS 通用信息管理系统是目前功能比较齐全,操作简单的优秀信息管理软件之一,在国内许多单位尤其是图书,情报档案及办公室信息管理部门应用相当广泛,该软件是为管理结构化非数值数据库而专门设计,它的主要特点是数据库设计方便,且支持变长字段的处理。对于重复出现的数据元素,可以以子字段的方式进行处理。通过采用先进的 B\* 管理技术,使得该系统的检索速度相当可观,几乎不受文献数量的影响。它的另一个特点是面向非计算机人员,因此,用户无特殊使用要求可不必自己动手编制程序,而只需要具有一般的关于 Micro CDS/ISIS 基本操作知识,就可按照各级菜单提示,建立满足自己需要的各种数据库,并对其进行数据库维护、检索、编辑排版、打印、统计计算、转换等操作。有关 Micro CDS/ISIS 的使用方法及其特点已有过详细的论述,但 Micro CDS/ISIS 的一些应用技巧及存在问题的解决方法未见详细报导,考虑到 Micro CDS/ISIS 的使用者大多数属于非计算机专业人员,因此,笔者认为有必要总结出一些适合这部分人员特点,便于掌握的解决问题方法和应用技巧,以供参考。

## 二、数据库文件维护过程中的两个问题

近年来,国内已有众多用户把 Micro CDS/ISIS 作为优选的数据库管理软件。其最新版本 Micro CDS/ISIS (2.3 版)已经推出,与老版本 Micro CDS/ISIS (1.0 版)比较,增加了部分功能,并对其存在的一些问题作了修改,比较完善。不足之处是 V2.3 版本与 V1.0 版本相比,前者对运行环境要求较高,主要是 V2.3 版运行时要求内存空间大,一般(包括操作系统)大约需要内存一兆字节左右,这样除去操作系统所占内存外还必须 500K 左右的内存空间才能正常运行,这对于一些内存较小或没有汉卡的微机来说,如不进一步扩充内存是不能正常运行的。所以,尽管(2.3 版)功能比较完善,但因(1.0 版)运行空间小,操作方便,目前大多数用户仍偏爱使用老版本。根据这一特点,笔者将

以(V1.0)版本为例讨论 Micro CDS/ISIS 数据库文件系统及其在维护过程中两点问题的解决方法。

### 1. 不能对主文件进行正常检索

这是一个比较常见的问题,一般表现是:倒排文件发生混乱或无法实行正常倒排,因此,不能对主文件进行有效检索,如:一些新增加的记录检索不到,某些检索字段丢失或只能在主文件某一记录范围内实行有效检索等。对于这个问题,首先应分析一下 ISIS 倒排文件特点:ISIS 倒排文件系统是一个 B\* 文件。它由六个物理文件组成,即:CON, NO1, LO1, NO2, LO2, IFP 文件,这六个文件组成了一个能够快速存取的 B\* 文件,其中, CNT 是 B\* 树控制文件, LO1, LO2, NO1, NO2, 是节点和树叶文件。IFP 是对每个检索词登录列表文件。节点文件(LO1, LO2)用于小于等于 10 个字节检索词,(NO1, NO2)用于大于 10 个字节而小于等于 30 个字节文件。由 ISIS 倒排文件特点看出,这六个物理文件,其中任意一个出现问题,检索都将出现不正常。这对于一个较大型的数据库而言,其倒排文件运行时需要的内存空间是很大的,所以,欲保证对数据库实行有效检索,首先应注意主文件记录数一般不要超过软件所能承受的极限值(理论最大记录个数 V1.0 版本为三万条,但因各方面因素,实际微机上使用只能维护一万条左右),V1.0 版本主文件记录在极限值以内都可以进行正常倒排和检索。另外一点就是要考虑倒排字段(检索字段)不要设置过多,应把几个主要检索字段作为检索入口,进行倒排,如检索字段设置过多,它将占用较大的内存空间,一些内存较小的微机因空间不足将会使倒排文件无法正常运行。在使用精确词检索时,必须严格依照系统所能识别的方式来执行,任何微小的拼写差异系统也会拒绝受理,从而影响检索质量。其次应注意在进行修改或增加数据库记录结束后,随时更新倒排文件,并尽量避免出现倒排文件更新操作中的异常中断(如掉电)。如发现检索出现异常时,首先应确定检索词是否有误,主文件记录是否超出极限,倒排文件更新操作是否异常及检索字

段是否设置过多等。如无上述现象,则可待全部记录输入结束后,在确定主文件记录无任何错误的情况下,根据主菜单提示进入第三个系统程序模块:“倒排文件的维护”,选择该子菜单中的系统模块 F:“产生全部倒排文件”,运行后问题可得到解决。

## 2. 乱库的处理与预防

Micro CDS/ISIS 数据库文件系统由倒排文件和数据库文件组成,数据库文件由 .MST, .XRF 两个文件构成,.XRF 是记录号索引文件,.MST 是数据库主文件,所有记录都顺序的存贮在 .MST 文件中,该数据库文件系统在操作不当情况下,极易产生乱库现象(V1.0 版)。这是一个不容忽视的问题。据分析造成乱库的原因及现象有以下几点:①.突然掉电产生死机等意外事故或主文件记录超录过某一极限值。②.主文件有一条或几条记录出现错误或非法字符,此时无论进行那种数据库维护方式操作,运行到该记录时便产生死机现象。以至于无法修改和删除这些错误记录,同时也无法更新倒排文件,若要对主文件进行重新组织或备份,屏幕上则会出现“未更新倒排文件,主文件不能备份”的提示符,从而使整个数据库陷于瘫痪状态。③.浏览或修改主文件时每一屏的节奏过快,这是因为浏览或修改主文件时,在某一范围之内,.XRF 索引文件是顺序从 .MST 中抽取记录,而 .XRF 文件是以表的形式存贮的,表中每一项实际是一个指针,每一个指针都包含有主文件记录块号和该记录内容所在地址,当节奏过快时,极有可能造成这些指针分配上的混乱,或者给主文件记录标识上错误的内容,产生乱库现象。以上几点问题是造成乱库的根本原因,应在操作时尽量加以避免。其次应注意随时备份主文件。这样出现问题时不至于造成大的损失。如发现乱库,尤其在执行数据库维护操作过程中出现死机时,应首先弄清主文件中造成死机的记录号或记录范围,如果主文件已有备份,在损失较小的情况下,可重新初始化主文件,然后将备份的主文件重新拷入,若主文件没有备份或重新初始化主文件损失较大的情况下,采用如下方法解决:在主菜单提示下进入第四个系统程序模块“数据库备份和 ISO 格式转换”,此时不能选择模块 B:“主文件的备份”,因为此模块不具备任选某一个记录范围的备份功能,尤其在严重乱库情况下,这项功能已完全丧失。因此,应选择其中的模块 E:“本系统格式转换为 ISO 2709 格式”,按照提示将损坏记录中的前一部分记录(不包括损坏记录)按 ISO 2709 格式备份到软盘上,重复上述过程将后一部分备份另一张盘上,退出后,再进入该菜单中程序模块 I:“ISO 2709 格式专换为本系统格式”依

次重新拷入并组织主文件,然后将剩余部分已损坏的记录重新进行输入,经更新倒排文件后即可恢复主文件及各项功能。

## 三、几点数据库维护技巧

### 1. 主文件的初始化与备份

(1)主文件的初始化 Micro CDS/ISIS 对数据处理的优点之一是不定长处理,这样可节省大量的磁盘空间,但是,在对主文件记录进行经常性修改时,仍会浪费许多空间,其中主要是因为对主文件记录修改时,数据库主文件 .MST 所占空间增长很快,而 ISIS 文件管理系统无法管理这些不能使用的空间。为解决这一问题, Micro CDS/ISIS 提供了一个主文件重新组织的功能,用于收回由此而浪费的空间。执行时,可根据主菜单提示选择第四个系统程序模块 E:“数据库备份和 ISO 格式转换”,选择该子菜单中的模块 C:“主文件的重新组织”运行时,系统首先生成一个备份文件(.BKP),除了不需要的 .XRF 文件外,该文件的结构及格式与 .MST 文件相同,且不备份那些已作过删除标志的记录。备份文件的恢复是顺序的,原来已标记为逻辑删除的记录,此时已被标记为物理删除,定期的执行这一功能是回收空间的有效方法。

(2)主文件的备份 主文件的备份是数据库维护过程中一个重要环节, Micro CDS/ISIS 软件本身具有主文件备份功能,不足之处是,如有任意一条记录未更新倒排文件(处于倒排悬挂状态),该功能将丧失。它的主要缺点是:即浪费时间又不能从主文件中任选记录范围进行备份。采用 DOS 的 BACKUP/BESTORE 命令比较理想,但需要备份某一记录范围时,便显得无能为力。因此建议使用 ISO 2709 格式转换备份和恢复主文件,速度虽然慢一些,但备份的记录范围可任选,还可在备份盘上作出标记,运用时比较灵活方便。

(3)数据录入技巧 建库过程中的数据录入需耗费大量时间。因此,提高中文数据库的录入速度是一个值得注意的问题,如能灵活掌握 Micro CDS/ISIS 软件中的“静态隐含值”和“动态隐含值”,则可带来极大的方便。“静态隐含值”是在定义数据库结构建立数据库输入工作单时定义的,它可以在修改输入工作单时修改并删除,也可在数据录入时进行修改。对于数据录入项目中出现频率很高或者基本不变的字段,完全可以用“静态隐含值”预先定义在输入工作单上,而对于周期性出现的录入字段如:文献库的刊名,举办单位等,可采用“动态隐含值”每次进行输入操作之前将其定义好,待这一输入周期输入结束退出后,便自动消失。下一次输入另一种文献时可重新定义,这两(下转 60 页)

# 一种给 IBM-PC/XT 机硬盘加锁的实用方法

大连理工大学管理学院 李铭

笔者通过实践,摸索出一种给硬盘加锁的简便方法。它的基本思想是,在软盘上先用 DEBUG 分别编写一个给硬盘加锁程序(LOCKH.COM)和给硬盘解锁程序(UNLOCKH.COM)。当用户暂时不使用硬盘且不希望他人使用硬盘时,执行硬盘加锁程序,关掉电源。此时,若别人没有硬盘解锁程序,不论用硬盘或软盘启动,都无法进入硬盘,从而对硬盘的内容起到了保护作用。当用户需要使用硬盘时,可用软盘启动系统,待系统启动完毕后,执行硬盘解锁程序,再重新启动系统,就可进入硬盘。

程序的实现方法是根据微机的 ROM 系统的工作原理进行的,也就是当机器启动时,ROM 系统的其中一项工作就是判断主引导扇区的结束标志(55AA)是否正确,若标志不正确且用硬盘启动时,则直接进入磁带 BASIC,无法使用硬盘,若用软盘启动,也无法进入硬盘。硬盘加锁程序与硬盘解锁程序正是利用了 ROM 系统的这一工作原理编写的。执行硬盘加锁程序时,破坏了主引导扇区的结束标志。而执行硬盘解锁程序时,则恢复主引导扇区的结束标志。

下面是利用 DEBUG 程序编写硬盘加锁程序(LOCKH.COM)与硬盘解锁程序(UNLOCKH.COM)的方法:

A>DEBUG

—N LOCKH.COM

—A 100

```
0EDA:0100  PUSH  DS
0EDA:0101  MOV   AX,0000
0EDA:0104  PUSH  AX
0EDA:0105  MOV   DX,0080
0EDA:0108  MOV   CX,0001
0EDA:010B  MOV   BX,0200
0EDA:010E  MOV   AX,0201
0EDA:0111  INT   13
0EDA:0113  MOV   AH,00
0EDA:0115  MOV   [03FE],AH
0EDA:0119  MOV   DX,0080
0EDA:011C  MOV   CX,0001
0EDA:011F  MOV   BX,0200
```

0EDA:0122 MOV AX,0301

0EDA:0125 INT 13

0EDA:0127 RETI

—RCX

CX 0028

:28

—W

Writing 0028 bytes

—N UNLOCKH.COM

—A 100

0EDA:0100 PUSH DS

0EDA:0101 MOV AX,0000

0EDA:0104 PUSH AX

0EDA:0105 MOV DX,0080

0EDA:0108 MOV CX,0001

0EDA:010B MOV BX,0200

0EDA:010E MOV AX,0201

0EDA:0111 INT 13

0EDA:0113 MOV AH,55

0EDA:0115 MOV [03FE],AH

0EDA:0119 MOV DX,0080

0EDA:011C MOV CX,0001

0EDA:011F MOV BX,0200

0EDA:0122 MOV AX,0301

0EDA:0125 INT 13

0EDA:0127 RETF

—RCX

CX 0028

:28

—W

Writing 0028 bytes

—Q

A>



## 一种屏幕上图块自动替换的有效方法

合肥工业大学机械系 鲍献华

美国 AutoDESK 公司的 AutoCAD 图形软件包是国内公认的先进支撑软件,国内已开发的微机 CAD 软件几乎都是在 AutoCAD 上开发的。正由于它是通用软件,不可能满足各类用户的特殊要求,有时候,用户觉得现有的 AutoCAD 功能还不够强。如在 CAD 制图时,往往会出现这种情况,当你从存有大量图块的图形库中调出一个图块 A 插入到图中的许多位置中后,突然发觉图块调错了,实际应插入的是图块 B,而不是图块 A。虽然此时可用 AutoCAD 的图块功能重新定义、修改图块 A 和 B,但这样做并不好,因为图块 A 和 B 都是正确定义的、有用的图块。而 AutoCAD 又没有这种在屏幕上实现图块自动替换的功能,那么能否找到一种既简单又方便的方法实现这一目的呢?

笔者利用 AutoCAD 的内嵌语言 AutoLISP 编写了一个名为 CB. LSP 程序,实现了块的自动替换,见所附程序。使用时将 CB. LSP 调入内存(或者事先将 CB. LSP 程序存入 ACAD. LSP 中,靠 ACAD. LSP 自动载入内存),象使用 AutoCAD 外部命令一样,在 COMMOND 命令状态下,打 CB 然后回车,此时系统就向用户提示:输入新的图块名,待输完后敲回车,系统又向用户提示在屏幕上选择被替换的图块,当你这样做后,屏幕上的图块立即被新图块所替换。如要继续替换,则打一回车键,重复前面的选择过程,否则敲任一其它键,替换结束。

CB. LSP 程序是针对图块的替换而编写的,实际上,许多制图作业过程都可以利用 AutoCAD 的内嵌语言 AutoLISP 定义成外部命令,使用时将会大大提高制图效率。AutoLISP 是 AutoCAD 二次开发强有力的工具,在 AutoCAD 所及之处都能找到它的力量之所在。随着用户开发的深入,AutoLISP 的潜力将会不断地被挖掘出来。

附:CB. LSP 程序:

```
(defun c:CB()
  (setq ac t)
  (while ac
    (setq newblock (getstring "\n 输入新块
名:"))
    (setq slength (strlen newblock))
```

```
    (if (=slength 0)
      (princ "\n 新块名输入无效,重
输!")
      (setq ac nil)
    )
    (setq endtest 32)
    (prompt " 选择要替换的块:")(terpri)
    while (=endtest 32)
      (setq ss nil)
      (while (not ss)
        (setq ss (ssget))
        (if (not ss)
          (princ "\n 选择无效,重选!")
          )
      )
      (setq ssl (ssllength ss))
      (setq k 0)
      (while (<k ssl)
        (setq ename (ssname ss k))
        (setq edate (entget ename))
        (setq edate
          (subst (cons 2 newblock) (assoc 2
edate) edate)
        )
        (entmod edate)
        (setq k (+k1))
      )
      (prompt "\n 如继续替换,请打回车键,否
则打任一其它键:")
      (setq endtest (cadr (grread)))
    )
    (princ)
```

# 多路数据采集内存使用及存盘法

武汉水利电力学院 任清珍 黄天戌

**摘要** 多路数据采集有它的独特性。因数据量大,故内存的使用分配显得很重要。本文介绍一种适合于 IBM-PC/XT 及其兼容机多路数据采集系统的数据存储法。用宏汇编语言编程,科学使用内存,并通过软中断调用 PC-DOS 的磁盘信息存取功能,实现一种内存使用灵活,存盘迅速简单,查询保存方便的数存法。

多路数据采集广泛地用于模型或原型观测系统中。所用的中心微机通常选 IBM-PC/XT 或其兼容机。因需采集的通道多,如果连续工作时间长,即使对具有 640KB 内存的 IBM-PC/XT 微机来说,也存在着一个合理,灵活使用内存问题。

再之,多路数据采集要求速度快,采样周期可控以及各种状态转换灵活,故在采集, A/D 转换和存贮部分一般采用宏汇编语言编程。下面通过一个在 128 路电导率测试系统中对内存的安排和存盘的设计,提供一种用宏汇编语言编写,方便灵巧的多路采集数据存储法。

## 一、内存使用——开辟存盘缓冲区

图 1 是装入多路采集执行程序(·EXE 文件)之后, IBM-PC/XT 机的内存分配示意图。

用户执行程序装入内存后,其代码、数据和堆栈是分开开放的。采集的数据可存到图中所标出的采集数据可存区中。

### 1. 开辟所需的存盘缓冲区

可以看出,尽管采集数据存放区域看起来很大,但它的容量毕竟是有限的。许多采集工作中,采集路数多,采样周期短,连续工作时间长,若无止尽地向该区存放数据,则势必会出现内存装满的现象。

为解决上述问题,可在采集数据可存区中开辟一段所需格式和容量的存盘缓冲区(图 1 中斜线部分),把采来的数据经过适当地处理后存放在这有限的内存区中。当存放的数据达到了规定的容量后,由 DOS 启动外设,将这一完整的存盘缓冲区的内容作为一个文件存盘(存盘缓冲区的长度也就是磁盘文件的长度)。然后,存盘缓冲区的指针回到初始位置,以保证后续采集工作可重新使用这一区域。

这种把所采的数据存放在有限的存盘缓冲区中,再由 DOS 成批地向外设(软硬盘)转移的方法,达到了一种所希望的无限量数据采集存储的效果。

### 2. 存盘缓冲区的格式设计

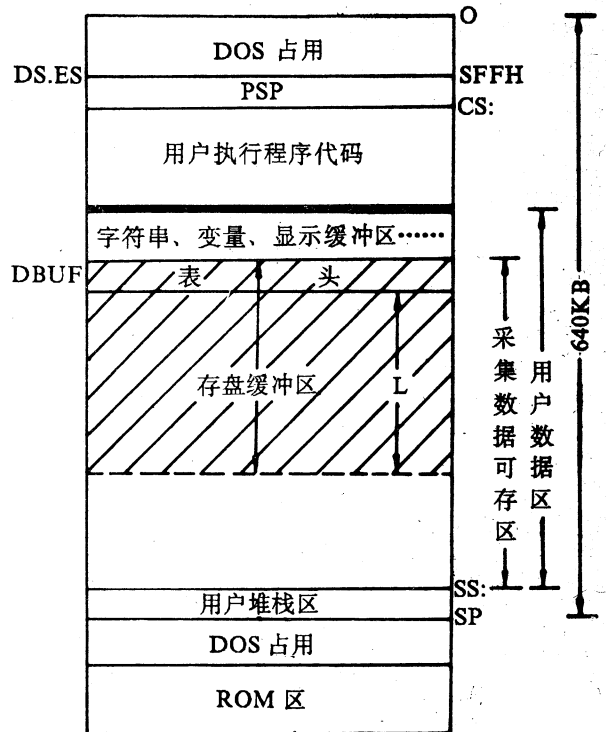


图 1 内存分配设计

图 1、内存分配设计

格式的设计包括文件表头,数据存放格式和数据打印排列格式等。

存放的数据采用 ASC II 码。把采集来的原始数据经过标定,补偿等必要的运算,转换成符合要求的 ASC II 码放到存盘缓冲区内。这样,所存的磁盘文件就可在 DOS 系统下用 TYPE 命令查询和打印。这比采用二进制的原始数据文件需做二次处理方便省时的多。由此也看出预先设计存盘缓冲区的优点,不必担心内存不够。

### 3. 存盘缓冲区的长度设定

存盘缓冲区的长度(即容量)设定是很灵活的,用户可根据自己采集工作的需要预算确定。预算方法为:

$L \approx M \times N$  (字节)

其中:  $L$ —数据区长度(字节)

$M$ —一次采样中,所采集的全部数据以及相应的空格,回车换行符等所需字节

$N$ —每个存盘文件所含的采集次数

比如在 128 路电导率测试系统中,  $M$  预算为 680 字节,  $N$  设为 25, 则:  $L \approx 680 \times 25 = 17000$  (字节) 如要设计一个 64 路系统,  $M$  可预算为 340 字节, 若仍设  $N$  为 25, 则:  $L \approx 340 \times 25 = 8500$  字节。

从上述关系也可以看出, 对于同样的  $L$ , 如果采集系统的路数太多, 则在设计中可适当减少  $N$ , 反之则可增大  $N$ 。例如: 设计时你想固定存盘缓冲区的  $L$  为 17000 字节, 那么预算:

对 64 路系统:  $M = 340$ , 则  $N = \frac{17000}{340} = 50$  (次) 对

128 路系统: 设  $M = 680$ , 则  $N = \frac{17000}{680} = 25$  (次)

## 二、存储的实现—DOS 扩展文件功能调用

1 编制存盘子程序清单如下:

```
SAVE    PROC NEAR
        LEA DX, FNAME; 搜索文件
        MOV CX, 0
        MOV AH, 4EH
        INT 21H
        CMP AL, 18; 无, 建立文件
        JE CRETE
        MOV AX, 3D01H; 有, 打开文件
        INT 21H
        JMP SAVEA
CRETE:  MOV AH, 3CH; 建立文件
        INT 21H
SAVEA:  MOV FHEAD, AX; 保存文件代号
        LEA DX, DBUF; 文件写盘
        MOV BX, FHEAD
        MOV CX, 17104
        MOV AH, 40H
        INT 21H
        MOV BX, FHEAD; 关闭文件
        MOV AH, 3EH
        INT 21H
        RET
SAVE    ENDP
```

DOS 2.0 以上的版本可使用扩展文件功能调用(用软中断 21H)。这种调用和用户程序接口, 读写操作都较简单, 并支持目录。它可以不使用 FCB 和记录方式, 而直接采用 ASCII 串文件名和文件代号。文件名的 ASCII 串表示形式为: [d:] [Path] f; lename [· 副名], 0, 文件代号是一个 16 位二进制控制字。

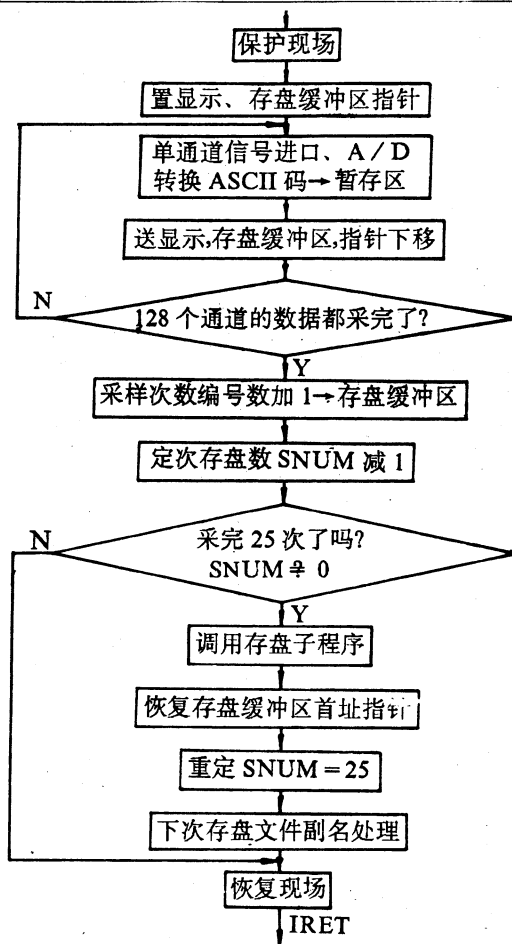


图 2、采集中断服务程序(128 路)

文件打开后, DOS 在自己的保留区建立一个有关文件名, 驱动器, 路径及读写指针等信息的控制块, 而只给用户一个文件代号即可。

这种调用的编程也简单。步骤是: 给文件名, 代号开辟内存单元(本例中文件名为 FNAME DB "C:\SAMP\DAT.00", 0)→分配足够长的内存缓冲区(正好用上述开辟的存盘缓冲区)→功能号调用 4EH, 3CH, 3DH 去搜索, 建立和打开文件→读写长度(字节数)送 CX→40H 进行文件写盘→3EH 关闭文件。

2 在采集中调用存盘子程序

图 2 是采集中断服务程序。每采 25 次后调用一次存盘子程序。文件的副名在程序中变化。随着采集的进行, 以 25 次采样为间隔, 在硬(软)盘的 SAMP 子目录中, 得到了 DAT.00, DAT.01, DAT.02... 多个可永久保存, 查询打印方便的文件。采集工作无内存不够之忧, 系统安全可靠。

其中: FNAME—文件名

FHEAD—文件代号

DBUF—存盘缓冲区首址

17104—存盘缓冲区总长度(L 及表头)

## 利用 GAL 设计的一种编码键盘接口

华中理工大学 赵英俊 胡阳

**摘要** 本文介绍一种利用一片 GAL 实现的编码键盘接口。该接口可以管理 14 个按键,具有处理按键抖动和多键闭合的功能,能够很方便地与各类微机相联,适用于小型智能仪器及其它微机应用系统。

### 一、工作原理

本接口实际上属于独立联接式的编码键盘接口,其功能框图如图 1 所示。K0...KD 共 14 根键输入线接到 14-4 优先权编码器的输入端。编码器的输出则接到一个 4D 触发器的输入端。触发器的时钟由一个 R-S 触发器的 Q 端控制,锁存的数据最后由三态门输出。这里的 R-S 触发器的置位端有 14 个输入。当 K0...KD 中有任何一个键被按下时,R-S 触发器就会输出一个置位信号(H)。该信号一方面用作锁存器的时钟,使之将按键编码数据保存起来;另一方面还可以用作为是否有键被按下的指示信号供查询或申请中断用。CPU 打开三态门,就能直接读入按键的编码(键值),勿需由软件再作识别。

从框图不难看出该接口具有以下特性:

1、自动去抖 由于 4D 触发器的时钟输入取自 R-S 触发器的 Q 端输出,Q 端一旦被置位后,只要 R 端不变,无论 S 端如何变化,Q 端都将保持 H 态不变。这样就使触发器只锁存了一个数据,从而消除了抖动的影晌。

2、保持键值,等待处理 正是由于 R-S 触发器的作用,该接口可以用其中的 4D 触发器保持键值不变,直到 CPU 响应之后,向 R-S 触发器发出复位信号,才使接口能够接受下一次按键。这样,无论 CPU 在干什么事,都不会发生丢失按键的情况。当然,这样也就限制 CPU 除非对最初的按键作出响应,否则后面的按键都无效。

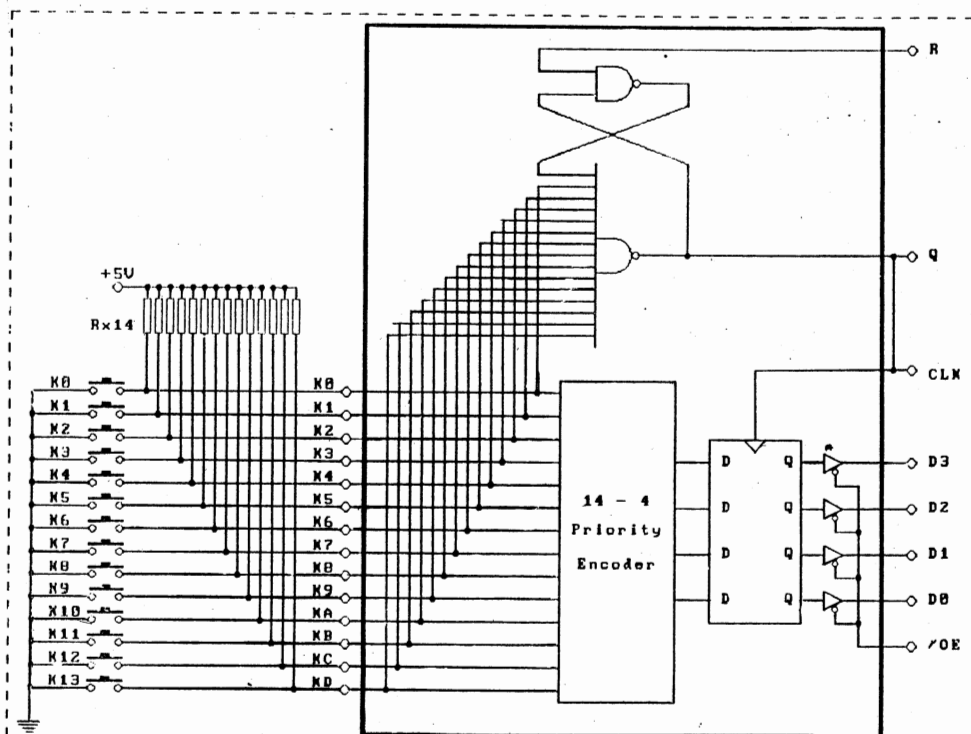


图 1 GAL 功能框图

3、可以处理多键闭合 由于键盘编码采用的是优先权编码器,因此在任意时刻无论有多少键被按下,编码器都只会输出一个有效的键值,即优先权最高的那个键的键值,从而避免了多键闭合的影响。

4 与数据总线兼容 因为键值输出采用三态门,所以该接口可以直接挂到 CPU 的数据总线上。

## 二、用 GAL 实现接口的方法

上述的编码键盘接口如果用普通的 TTL 集成电路来实现,大约需要六、七块芯片,而用 GAL 器件只需一片 20V8 即可完成。可见在微机应用系统中适当地选用 GAL 器件,不仅可以保密,而且可以提高设计效率。

GAL20V8

KB INTERFACE

Y. J. ZHAO 1990. 10. 30.

JPJK

CLK KO K1 K2 K3 K4 K5 K6 K7 K8 K9 GND

/OE KA KB KC KD D0 D1 D2 D3 Q R VCC

/Q=/R

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* K9 \* KA \* KB \* KC \* KD \* /Q

D0<sub>i</sub> = K0 \* /K1

+K0 \* K1 \* K2 \* K /K3

+K0 \* K1 \* K2 \* K3 \* K4 \* /K5

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* /K7

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* /K9

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* /K9 \* KA \* /KB

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* /K9 \* KA \* KB \* KC \* /KD

D1<sub>i</sub> = K0 \* K1 \* /K2

+K0 \* K1 \* K2 \* /K3

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* /K6

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* /K7

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* K9 \* /KA

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* K9 \* KA \* /KB

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* K9 \* KA \* KB \* KC \* KD

D2<sub>i</sub> = K0 \* K1 \* K2 \* K3 \* /K4

+K0 \* K1 \* K2 \* K3 \* K4 \* /K5

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* /K6

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* /K7

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* K9 \* KA \* KB \* /KC

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* K9 \* KA \* KB \* KC \* /KD

+K0 \* K1 \* K2 \* K3 \* K4 \* K5 \* K6 \* K7 \* K8 \* K9 \* KA \* KB \* KC \* KD

实现上述功能的 GAL 编程逻辑设计输入文件如图 2 所示。GAL 的引脚定义如图 3 所示。其中 K0...KD 为 14 个按键输入线,D0...D3 为键码输出端,Q 为按键指示输出,R 为复位输入,CLK 和/OE 分别为时钟和输出允许。

在 K0...KD 中,以 K0 的优先级别最高,KD 的优先级别最低,其它顺次排列。平时,K0...KD 被拉至高电平,如果 CLK 上有脉冲出现,接口输出的编码为 1110B。K0...KD 的有效编码依次为 0000B...1101B。CPU 通过判读得到的键码,便可得知键盘的工作状态。另外,Q 端输出既可以作为 CPU 的中断申请信号,也可以作为一种状态标志供 CPU 查询用。

```

D3: =K0 * K1 * K2 * K3 * K4 * K5 * K6 * K7 * /K8
+K0 * K1 * K2 * K3 * K4 * K5 * K6 * K7 * K8 * /K9
+K0 * K1 * K2 * K3 * K4 * K5 * K6 * K7 * K8 * K9 * /KA
+K0 * K1 * K2 * K3 * K4 * K5 * K6 * K7 * K8 * K9 * KA * /KB
+K0 * K1 * K2 * K3 * K4 * K5 * K6 * K7 * K8 * K9 * KA * KB * /KC
+K0 * K1 * K2 * K3 * K4 * K5 * K6 * K7 * K8 * K9 * KA * KB * KC * /KD
+K0 * K1 * K2 * K3 * K4 * K5 * K6 * K7 * K8 * K9 * KA * KB * KC * KD

```

DESCRIPTION

图2 GAL 编程的源程序

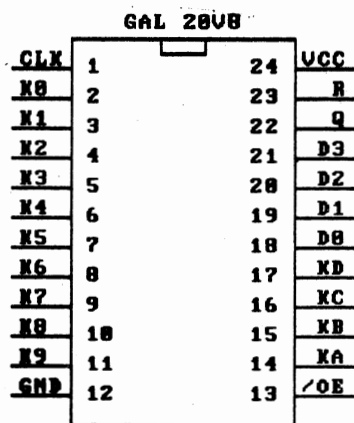


图3 GAL 引脚图

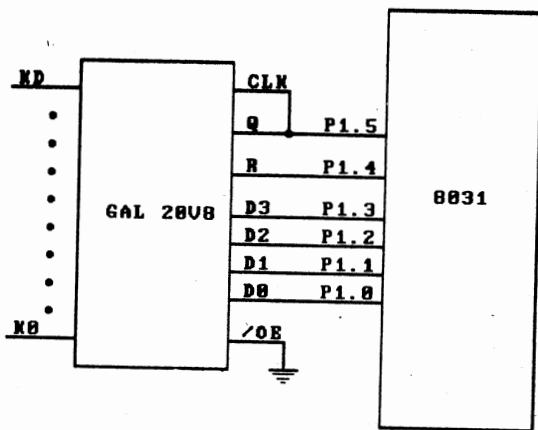


图4 GAL与8031的联接图

将图2所示的源程序用FAST MAP 软件包汇编之后,得到相应的JED 文件。用适当的编程器将这个JED 文件固化到一片20V8 的GAL 芯片中,就得到了这种编码键盘接口。

### 三、应用举例

这种用GAL 器件设计成的键盘接口,使用起来灵活方便。CPU 可以根据应用的具体需要,用查询方式或中断方式来管理它。这里仅举一个与8031 单片机系统联接的实例(图4)供读者参考。

为了简明起见,这里只画出8031 系统中与键盘接口有关的部分,按键与键盘接口的联接请参阅图1 中的相应部分。该系统采用查询方式管理键盘。8031 的P1 口与键盘接口相联,其中P1.0...P1.3 分别接D0...D3 作键码输入,P1.4 接接口的复位端R,P1.5 接接口的按键指示端Q,接口的输出允许端/OE 则直接接地,使输出直通。

与键盘输入有关的源程序如图5 所示。

STRAT;

SETB P1.4;键盘接口初始化

TEST;

JNB P1.5,Y;查询是否有键被按下

MOV A,P1;读入接口信息

AND A,#0FH;屏蔽高4 位,获取键值

RL A;键值加倍

MOV DPTR,#TABLE;指向转移表首址

JMP A+DPTR;以键值为索引查表转移

NEXT;

CALL DELAY;延时等待按键释放

CLR P1.4;清除接口,为下次按键作准备

AJMP START;重新开始

TABLE;

AJMP KBROUT0;转向按键0 服务程序

AJMP KBROUT1;转向按键1 服务程序

AJMP KBROUT2;转向按键2 服务程序

⋮

AJMP KBROUTD;转向按键D 服务程序

DELAY;

.....;延时子程序

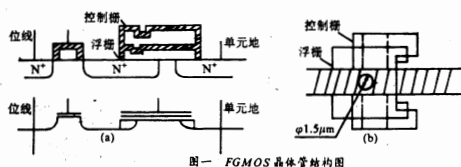
图5、键盘接口用的源程序



# 通用阵列逻辑 GAL 器件的分析研究

西南交通大学计算机系 朱焱

**摘要** 本文分析讨论了当今最新 PLD 类产品——通用阵列逻辑 GAL 器件的设计原理,器件的结构,并对其内部编程机理作了初步探讨。



图一 FGMOS 晶体管结构图

通用阵列逻辑 GAL 是 1986 年问世的新型 PLD 器件,现已逐步成为 TTL、门阵列、PAL 器件及其它各种 PLD 器件的理想替代品。在逻辑设计、计算机硬件设计、微电子等系统中使用 GAL 器件,可以发现系统设计得到简化,系统功能却大大提高,由于 GAL 器件具有比 SSI/MSI 更高的功能集成度,因此系统体积降低,功耗减少,系统可靠性提高,使设计过程有了较大的飞跃。

## 一、GAL 器件的设计原理

### (1) GAL 采用了 E<sup>2</sup>CMOS 电路工艺。

这种电路工艺具有无直流功耗,负载管与驱动管

跨导不需按一定比例,脉冲上下沿对称,高电平可达电源电压等特点,这些特点使 E<sup>2</sup>CMOS 与其它工艺相比具有低功耗,集成度高,抗干扰能力强,速度快,可试验性等优点。

### (2) GAL 采用了 FGMOS 作为编程器件。

FGMOS 晶体管结构如图一所示。GAL 器件能够多次擦除反复编程的关键在于晶体管的浮栅与漏区重叠部有一个直径 1.5 μm 的圆,此处的绝缘层厚度约有  $1 \times 10^{-5}$  mm,而其余的部分绝缘层厚度均大于  $5 \times 10^{-5}$  mm。(见图一(b))。当浮栅上有一正电压时,浮栅上的正电荷越过极薄的绝缘层而泄放,浮栅为不带电荷状态,通常表示成“1”。当控制栅接地电值,编程脉冲加到位线,浮栅上有一起始负电压,由于电容分压,浮栅电压上跳,一些正电荷从 N<sup>+</sup> 区穿过很薄的绝缘层到达浮栅上,使浮栅积累正电荷,使 FGMOS 管处于导通状态,并表示为“0”。浮栅上一旦有了电荷,由于它们被绝缘层包围着,因此不易丢失。

### (3) GAL 采用可编程输出逻辑宏单元 OLMC

OLMC 结构如图二所示,它主要由四个部分构成。

• 积项选择多路开关 (PTMUX); • OLMC 的或门接受来自“与”阵列 8 个积项中的 7 个,另一个积项作

这里应当说明两点,一是所有的按键服务程序都要以 LJMP NEXT 指令结尾,这样才能为接收下一次按键作准备;二是要注意在适当的时候给接口中的 R-S 触发器复位,不能过早,否则可能造成一次按键,多次响应的错误。因此应根据具体的情况,确定延时子程序 DELAY 延时的长短。一般情况下,取 10—20ms 为宜。

在此基础上,我们可以将图 5 扩展成一个既能防误发又能进行连发的键盘管理程序,其流程框图如图 6 所示。主要工作原理如下:系统在主程序中以 0.1 秒为周期查询键盘,因为手动按键的速度一般不超过 5 次/秒,所以不会丢失按键。如果微机连续读到相同的键值,则表明该按下后没有释放。若连续读到的次数小于 5,则表明按键时间小于 0.5 秒微机不作进一步处理,从而避免了因按键释放过慢所造成的误发。如果连续 5 次读到同一键值,则表明操作者要求连发,于是微机以 0.1 秒为周期进行相应操作。

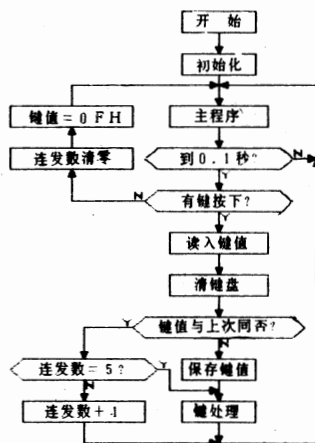


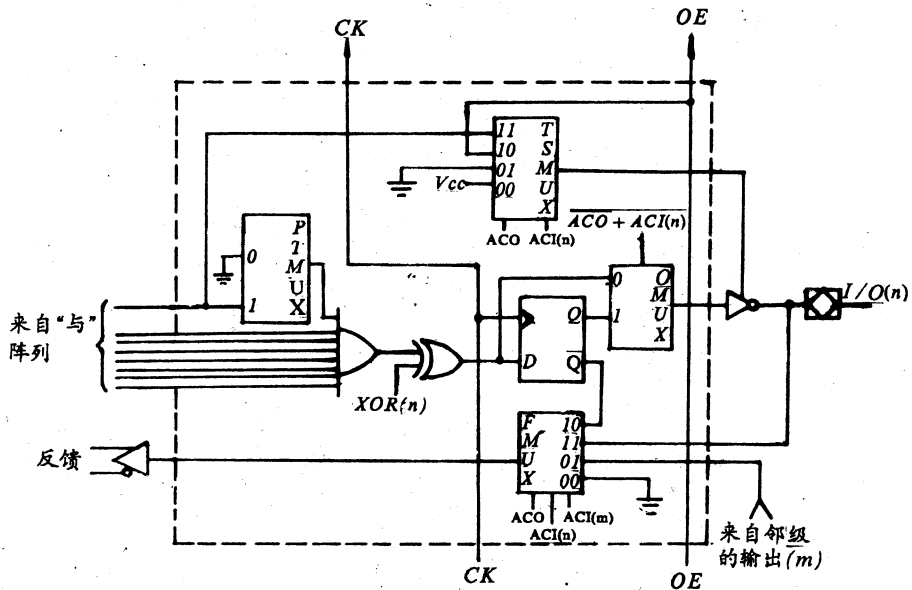
图 6 键盘管理程序框图

为 PTMUX 的“1”输入端,“0”端为地。PTMUX 的功能是选择“1”端或“0”端作为或门的另一输入项。

• 输出选择多路开关 (OMUX): OLMC 的“与-或”逻辑操作的结果一路送到 OMUX 的“0”端,作为组合型结果;另一路触发 D 寄存器,作为寄存器型结果送到 OMUX 的“1”端。OMUX 在 AC0 和 AC1 的控制下,选择组合型或寄存器型输出。

• 输出允许控制选择多路开关 (TSMUX): 该开关在地、Vcc、OE 和“与”阵列输出之一四路中选择一个信号作为 OLMC 输出三态门控制信号。

• 反馈选择多路开关 (FMUX): 该开关选择本级 OLMC 输出、邻级 OLMC 输出、地或  $\bar{Q}$  作为反馈信号,送回“与”阵列作为输入信号。



图二 OLMC 结构图

由上分析可知,四种多路开关在 SYN、AC0、AC1(n) 的组合控制下,使 OLMC 能产生五种有效的宏单元功能。列表示意。

宏单元功能	SYN	AC0	AC1(n)	XOR
专用输入方式	1	0	1	
专用组合型输出	1	0	0	0/1(低/高有效)
组合型输出	1	1	1	0/1(低/高有效)
寄存器型器件中的组合输出	0	1	1	0/1(低/高有效)
寄存器型输出	0	1	0	0/1(低/高有效)

SYN、AC0、AC1、XOR 的意义是:

SYN: 同步位。“0”表示 GAL 器件具有寄存器型输出能力;“1”表示 GAL 器件具有组合型输出能力。

AC0: 结构控制位。该位对于 8 个 OLMC 是公用的,它与各个 OLMC(n) 各自的 AC1(n) 配合,控制四个选择多路开关。

AC1: 结构控制位。对于八个 OLMC,每个有各自的 AC1(n)。

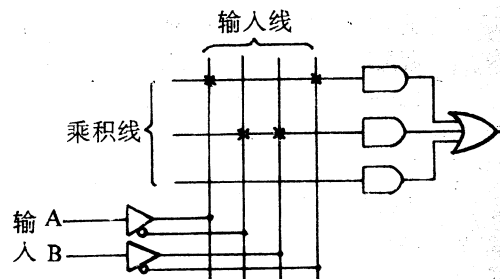
XOR: 极性控制拉。每个 OLMC 都有一个各自的 XOR(n) 而输出宏单元的极性是可编程的,它由或门输出的数据与极性控制位经异或门后产生。

数据	控制位	XOR 输出
0	0	0
1	0	1
0	1	1
1	1	0

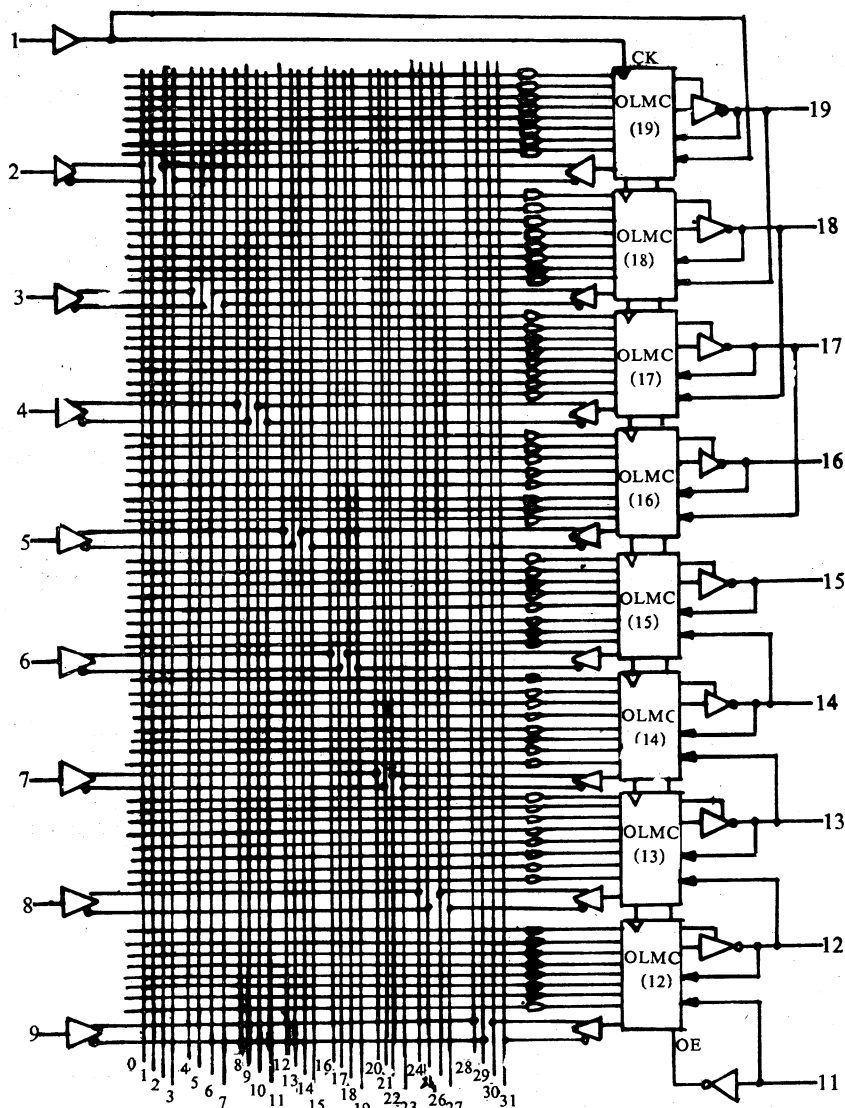
由上表可知,数据的极性取决于极性控制位的状态。

## 二、GAL 器件的结构

(1) GAL 器件是 PLD 系列的新产品,PLD 的阵列结构是由“与”阵列和“或”阵列组成的。“与”阵列和“或”阵列的不同结构形成了三种 PLD 阵列类型—PROM(“与”阵列固定,“或”阵列是可编程的),PAL



图三 一个 PAL 电路



图四 GAL 器件(GAL 16V8)逻辑图

(“与”阵列是可编程的,“或”阵列是固定的);PLA (两种阵列均可编程)。

由于对大多数应用来说,要求阵列大小较灵活,因此 PAL 与 PLA 均优于 PROM,另外对“或”阵列的可编程灵活性要求并不强烈,因此,PAL 阵列结构已成了工业标准的 PLD 结构。

图三是一个简单的 PAL 电路。A、B 两输入端通过缓冲器得到两个互补的输出,然后送到乘积线上,各输入线与乘积线的交叉点构成一个 4×3 的可编程“与”阵列,在“与”阵列上,“X”表示该处熔丝未被编程,表

示逻辑“0”,与该点相连的输入构成“与”表达式中的一项;如果交叉点上无“X”,则表示该处熔丝已熔断,表示逻辑“1”,与该点相连的输入不构成“与”表达式中的一项;如果交叉点上出现“·”,则称之为硬连接,与该点相连的输入项始终构成“与”表达式中的一项。图三的输出  $F = A\bar{B} + \bar{A}B + B$ ,原因是:第一条乘积线上 A、 $\bar{B}$  两个输入与乘积线相连;第二条乘积线上  $\bar{A}$ 、B 两个输入与乘积线相连;第三条乘积线上 B 输入项与乘积线硬连接,不受功能变化编程的影响。

(2)GAL 器件(以 GAL16V8 为例)的(下转 19 页)

# 适用于计算机系统的硬时钟

四川省电子计算机应用研究中心 舒彦

**摘要** 在许多实时计算机控制系统、计算机实时监测系统中,特别需要一种十分精确稳定且不停电运行的硬时钟。这种硬时钟必须具有与计算机便于接口、外围电路简单的特点。大规模集成电路 Lm8361 就是一种既简单又经济的选择。

## 一、工作原理

Lm8361 是日本三洋公司生产的 P-MOS 集成电路。如图一所示,它与专用的显示块联接,可构成一数字显示钟。时钟运行时,各笔段信号输出为静态显示,某一笔段点亮时,输出端有相对  $V_{ss}$  端的高电压。其引脚定义如下:

2—22 足分别为十时、时、十分、分段信号标准七段码。28,29 足分别为电源、地,因 LM8361 是 P-MOS 电路,只能供给一负向电压。

33,34 足为时钟快速调整、慢速调整键,35 足为 50/60Hz 输入端,它作为时钟运行的基准频率。

36 足为 50/60Hz 选择端。此为高,选择 50Hz 基准频率输入;此为低,选择 60Hz 频率输入。

1,40 足为时钟上午、下午标志,

39 足为 1Hz 频率波形输出。

## 二、时钟构成

鉴于时钟输出笔段为静态输出,我们用缓冲器 74LS244 分别对时、十分、分的各段信号采样和处理显示。为了使各段信号输出稳定而不产生干扰,在每一段的输出端接入一负载电阻,使其有微弱的电流流过,如图三所示。用 2M 晶振产生的矩形波经四级 CD4017 计数器(十进制计数器)分频成为 200Hz 的波形,再由 CD4018(2—10 计数器)四分频成为 50Hz 信号。那么此时,36 脚接高电平。二个 74LS138 译码器由地址信号  $A_0 \sim A_7$ ,产生译码信号作为十时、时、十分、分各自缓冲器的选通信号,三个口总结如下:

在十分的采样中,加入 AM 标志,以判定上、下午时间。由此用 Z—80 程序作如下采样:

```
TIME IN A,40H      TRAN  PUSH BC
CALL TRAN 转换子程序  PUSH DE
LD B,A             LD C,A
IN A,42H           LD HL,TAB
CALL TRAN  TRAI     LD A,(HL)
LD D,A             CP A,C
IN A,41H           JR NZ,TRAN2
```

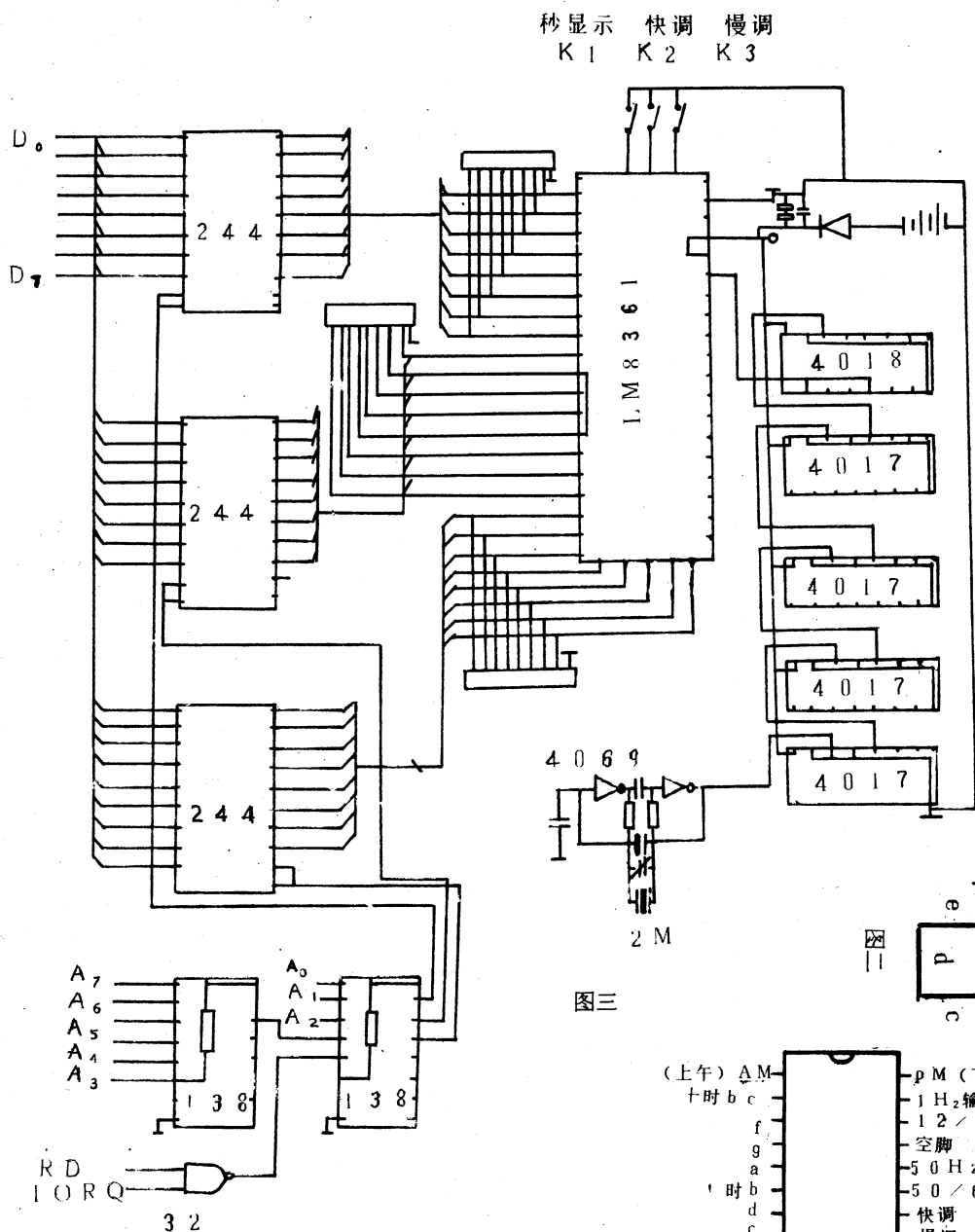
```
BIT O,A(判上、下午)  LD A,L
JR NZ,TM1            POP DE
CALL TRAN           POP BC
LD C,A              RET
JR TIM2 TRA2        INC HL
TIMIAND A,FEH       JR TRA1
CALL TRAN           LD C,A
LD A,B              ADD A,OCH(下午处
理)
TM2                 LD B,A
RET                 RET
TIAB DB FCH,60H,DAH,F2H,66H,B6H,BEH,
FFH,FFH,F6H,FDH,61H,DBH,CEH
DB 的佰是笔段字形码。
可见寄存器 B、C、D 分别放置时、十分、分的值。
为了做到时钟不停电运行,加入普通干电池作为
备用。如图  $V_{DD}$  加一二极管 D 再接 4.5V 电池。当系统
一旦停止供电,由电池向 LM8361,50Hz 产生电路供电,
保持时钟继续运行。
```

## 三、小结

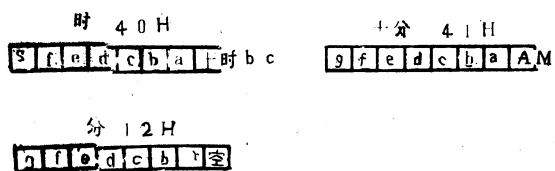
硬时钟的精度取决于 50Hz 矩形波的精度。且时钟快慢可由微调电容加以调整。由于是选用精度达  $10^{-6}$  的 2M 晶体,实践证明,系统误差小于 4 秒/天。并且时钟运行基本不受环境温度变化的影响,可以适用于各个恶劣的工业现场,即便是计算机系统“死机”,时钟仍能正常走时,收到较为满意的效果。如精度要求更高,则可选用高级恒温晶体。同时,仍可以开发 LM8361 定时输出控制功能作为计算机系统定时控制,或将其 1Hz 输出信号,作为中断信号充当时间信号源。

## 参考文献

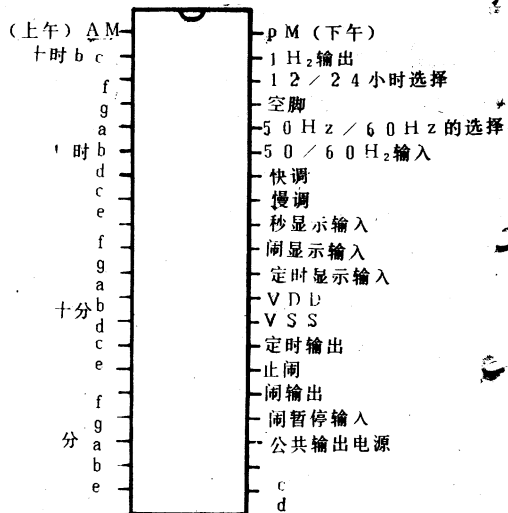
- 1、《电子报》1984 年第 17 期
- 2、《CMOS 设计手册》,三叶出版社(澳门)



图三



附图:



图一

# 一种利用资源共享技术的单片机开发装置

航空航天部重庆巴山仪器厂 荆予今

**摘要** 单片机的应用,已渗透到各行各业。单片机开发装置的价格,又让广大工程技术人员望而却步。本文介绍的这种利用资源共享技术研制的单片机开发装置,由于充分地利用了Z-80单板机的板上硬件资源及软件资源,因而使开发仿真板的硬件成本降到最低。对于拥有Z-80单板机的技术人员,只要再花不多的投资,就可以得到一台高性能价格比的Z-80, MCS-48, MCS-51 三用开发仿真装置。

单片机自80年代中进入我国后,由于其价廉、体积小,抗干扰能力强等优点,得到了各行各业的重视,正在被迅速应用于各种自动控制、智能仪器、设备中。然而,由于单片机不具有象单板机一样进行自我开发的能力,所以必须借助于开发装置进行软件的开发和硬件的调试。以目前市售的开发装置而论,虽然品种很多,但大多价格昂贵,使大多数中小企事业单位望尘莫及;即便是大型企事业单位,也只能购买少量开发装置,而对于大多数工程技术人员和课题组,仍不能满足科研需要。笔者在调查中发现,目前在很多单位中,都有大量闲置的Z-80单板机。如何充分发挥这些单板机的作用,使它在新的技术形势下再立新功,是加快单片机开发应用的一个重要方面。因此笔者应用资源共享技术,设计了一种具有较高性能价格比的单片机开发装置。该开发装置可实现MCS-48, MCS-51系列单片机的软件开发、调试,通过仿真插头又可以实现在线仿真。具有单步、断点、单步调用、全速等必要的开发手段,基本上不占用单片机的内部资源。

## 一、采用资源共享技术的意义

作为一种开发装置,应该具有一些必要的开发手段,这包括软件的装入与转贮功能,程序的编辑功能;诸如单步、单步、连步、断点运行程序等各种调试功能、固化程序的EPROM写入功能以及软硬件的在线仿真功能。这些功能的实现,需要大量的外围设备及软件。这也是市售的开发装置所以昂贵的一个原因。

事实上,一台完好的Z-80单板机已具备了上述各种功能的大部分。因此只要在此基础上增加1个单片机开发仿真板,使其充分享用Z-80单板机原有的硬件资源,再在Z-80监控的基础上开发出单片机的监控软件,就可以构成一台功能齐全的单片机开发仿真装置。JS单片机开发仿真板就是根据这个思路研制的。其价格只及市售MCS-51开发装置的1/9~1/10,却能开发48、51两种单片机。

这个开发仿真装置,从软件上讲,它同时享有Z-80、及单片机两种软件。而对Z-80CPU和单片微处理

器来说,它们又是共享一套硬件资源,当然这种硬件资源的共享不可能是两种微处理器同时对硬件资源的访问,而只能是分时进行的。

## 二、如何实现单板机的资源共享

资源共享,具体说,就是Z-80CPU和单片机共用一套地址总线,数据总线及控制线,实现对单板机上各种硬件资源的控制。但是如何才能使Z-80CPU和单片机既共享总线,而又不发生相互干扰呢?查Z-80CPU手册可知,Z-80CPU有二根特殊的控制线 $\overline{\text{BUSRQ}}$ 和 $\overline{\text{BUSA}}\text{K}$ 。当 $\overline{\text{BUSRQ}}$ 有效时,Z-80CPU在执行完当前指令后将交出三总线(三总线呈高阻状态)而处于暂停状态,这时外设(如单片机)可以接管三总线而实现对Z-80单板机上存储器、显示器、键盘、I/O口的控制。这种状态叫做单板机的单片机状态,在此状态下,可实现单片机软件的运行、调试。而当 $\overline{\text{BUSRQ}}$ 无效时,Z-80CPU又可通过 $\overline{\text{BUSA}}\text{K}$ 控制单片机交出总线,从而又恢复到单板机的Z-80状态。在Z-80状态,可利用单板机的监控程序实现对单片机程序的输入、打印、编辑、转贮,固化等操作。这样通过 $\overline{\text{BUSRQ}}$ 信号的有效和无效,就方便的实现了单板机从Z-80状态到单片机状态的互换,从而实现了硬件资源的分时共享。在Z-80与单片机的共同配合下,实现单片开发装置的所有必要功能。

## 三、开发装置的电路简介

通过以上介绍,读者就能够明白为什么应用资源共享技术,就可以大幅度降低单片机开发装置的硬件成本,从而获得较高的性能价格比。

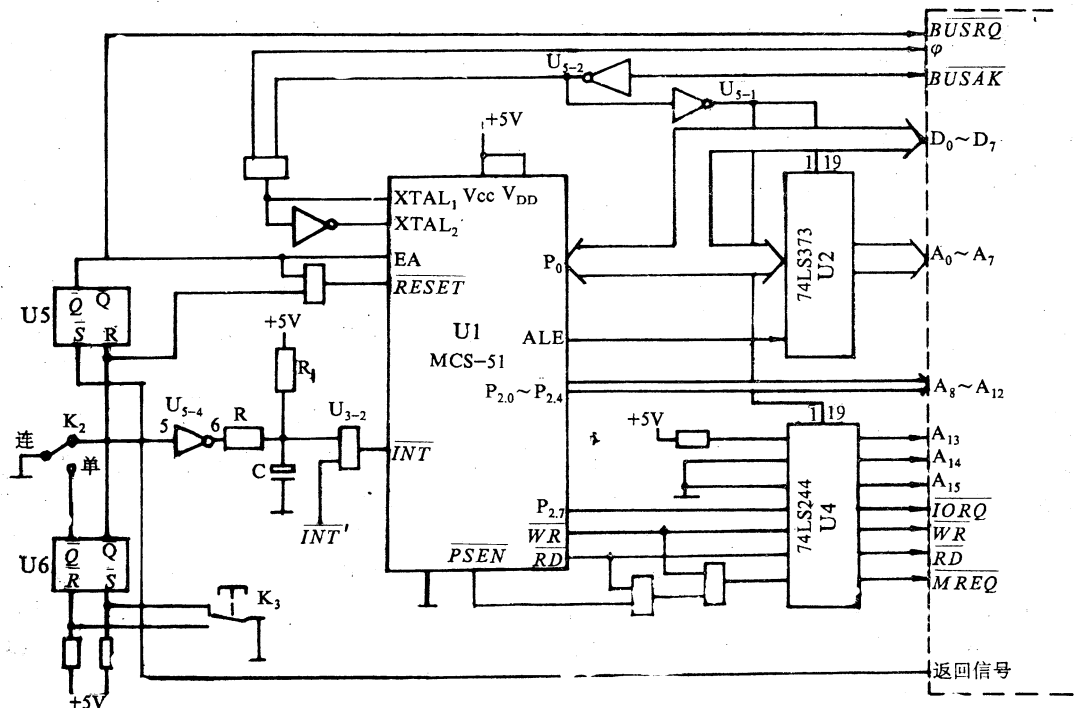
### 1. 数据、地址线的连接

图一是利用资源共享技术开发MCS-48系列单片机的简化原理图。74LS373用以锁存单片机 $P_0$ 口的低四位地址。而 $P_0$ 口输出的数据信息直接与单板机的数据总线相连。 $P_2$ 口的低四位输出的高4位地址也直接与单板机的地址总线的A8~A11相连。查INTEL单片机手册,MCS-48单片机在 $\overline{\text{RESET}}$ 有效及EA为低电平时, $P_0$ 处于高阻状态, $P_1$ 、 $P_2$ 口处于高阻上拉的输



人方式。因此单片机对于整个系统而言,是处于挂断状态,不会影响 Z-80CPU 的工作,而当单片机工作时,首先  $\overline{\text{BUSRQ}}$  信号有效, Z-80 在每个机器周期查询到此信号后,会立即响应总线请求中断释放三总线,因此

也不会对单片机的工作产生影响。但是在实际在线仿真调试时,为了防止 Z-80 信号对用户机的外围电路产生影响,一般都在 DB、AB 高八位加三态隔离门。



图一

单片机复位后,是从 0000H 开始执行程序,一般用户程序也是从 0000H 地址开始存放指令。由于单板机上从 0000H 开始用的前 2K 地址空间已被 Z-80 监控占用,所以为了不与单板机的监控发生冲突,开发装置采用了存储器分段技术,使单片机复位后,从 Z-80 单板机存储器的 2000H 单元开始取指令。也就是说限制单片机访问单板机 0000H~FFFFH 的空间。具体作法是当系统处于单片机状态时,把地址线 AB 常接高电平, A14、A15 接地。即限定单片机只能访问单板机的后 8K 空间。而这 8K 空间,正好是单板机提供给用户的 RAM 空间。这样对于单片机而言,复位后仍是从 0000H 开始执行程序,而这个地址正好是单板机的 2000H 地址。

## 2. 控制线的模拟

熟悉 Z-80 CPU 的读者都知道, Z-80 CPU 与 MCS-48 的控制线并不相同,为了使单片机能访问主机上的硬件资源,必须对 Z-80 的控制总线进行模拟。Z-80 的控制线较多,常用的有 MREQ, IORQ, RD,

WR。其中, WR 与单片机上的 WR 功能相似, RD 信号在单板机上只在访问 I/O 设备时有用。这两根信号线可用单片机上同名的控制线直接代换。MREQ 信号用于各种情况的存储器访问,非常重要,根据其功能可用单片机的 PSEN、WR、RD 三线相与后合成。IORQ 是输入输出请求线,单片机如果模拟出这个信号,就可以访问单板机上的 PIO、CTC,及键盘、显示系统。但是单片机已把 I/O 口集成到片子内部,片外本身并没有任何一个控制线与其相似,所以只得采用 P2 口的一根 I/O 线做为 IORQ,并通过软件使其产生 IORQ 信号。这样就可以实现 I/O 资源的共享。

## 3. Js 开发仿真板与单板机的连接

对于使用 TP-801 单板机的用户,只要在扩展区焊上一排单列插座,就可以把开发仿真板直接插在扩展区上。如果不是用 TP-801 单板机,而是 CMC-80 等其它型号的 Z-80 单板机,则可以通过一个 40 线印制板插座与开发板相连,40 线插座的另一端通过电缆焊到单板机的三总线上。

#### 四、单片机开发装置的工作过程

首先应在 Z-80 状态把待调式的单片机程序输入单板机 2000H 开始的存储器中, 然后开始调试。按下微动开关  $K_2$ , 使 R-S 触发器翻转,  $U_6$  的  $\bar{Q}$  上升为高电平, Q 端清零。由于  $U_6$  的 Q 端直接连到 Z-80CPU 的  $\overline{BUSERQ}$  线上, Z-80CPU 迅速响应总线请求中断, 交出三总线, 并发出  $\overline{BUSAK}$  应答信号。 $\overline{BUSAK}$  信号打开单片机的时钟控制门, 并做为使能信号, 打开  $U_2, U_4$  输出, 允许单片机接管三总线。当  $K_2$  开关复原后, R-S 触发器 Q 端为高,  $U_5$  输出高电平, 单片机的  $\overline{RS}$  端变高, 复位状态结束, 系统进入单片机状态。单片机首先检查有无单步, 然后从外映象区中读回内部各寄存器、累加器、片内 RAM 的内容, 再根据用户设定的 PC 地址开始执行用户程序。在执行用户程序的时候, 遇到下列三种情况之一, ①断点, ②单步, ③人工强迫中断用户程序, 单片机先把当前 PC 值、累加器 A 及各片内单元的内容送往片外映象区, 以备检查, 然后使  $U_5$  的  $\bar{S}$  端为低,  $U_5$  触发器置位,  $\overline{BUSERQ}$  信号  $\overline{BUSAK}$  信号相继解除,  $U_2, U_4$  的三态门关闭, 输出呈高阻, 系统又回到了 Z-80 状态。这时就可以在 Z-80 监控的支持下, 检查单片机指令运行结果, 并进行各种修改。

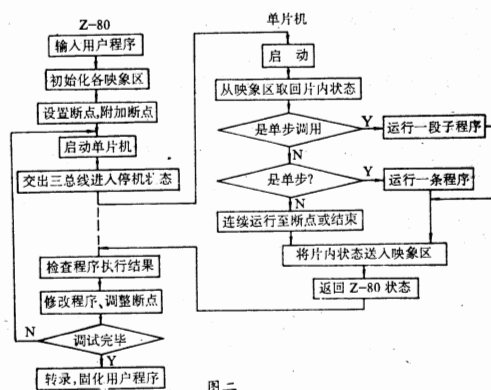
#### 五、单步功能的实现

单步运行程序, 是开发装置的一个重要调试手段, 单步功能是利用  $\overline{INT}$  及片内定时器或 CTC 实现的。从图一可以看出, 当  $K_2$  置于连步时, 反门 5 脚始终处于低电平端。而当  $K_2$  置于单步时, 反门 5 脚始终处于低电平端。而留  $K_2$  置于单步时, 反门 5 脚的输入电平就置于 R-S 触发器的  $\bar{Q}$  控制之下。当  $K_2$  按下时,  $\bar{Q}$  为高, 反门 6 脚输出为低, 电容器上的电荷可以通过  $R_1$  迅速放掉; 而把  $K_2$  松开时  $\overline{R-S}$  的 Q 端上升为高, 单片机  $\overline{RESET}$  端置位, 单片机开始工作。同时随着 R-S 触发器的翻转, 反门 6 脚输出为高。但是由于 RC 电路的延时作用,  $\overline{INT}$  脚并没有和  $\overline{RESET}$  脚同时升高。这时可令单片机启动后, 首先检查  $\overline{INT}$  脚, 如果为低则设置单步标志, 否则不设标志。单片机在读回片内状态, 进入用户程序之前, 再检查单步标志。如果有单步, 则设定定时器初值, 并开中断。当系统进入用户程序后的第一个机器周期内, 定时器发出溢出中断, 单片机在执行完一条用户指令后, 马上响应中断, 并把片内各单元状态送入片外映象区, 然后返回 Z-80 状态。至此, 完成一个单步操作。

#### 六、监控程序简介

本开发装置由于采用了资源共享技术, 利用 Z-80CPU 和单片机双机分时工作, 共同完成开发装置的

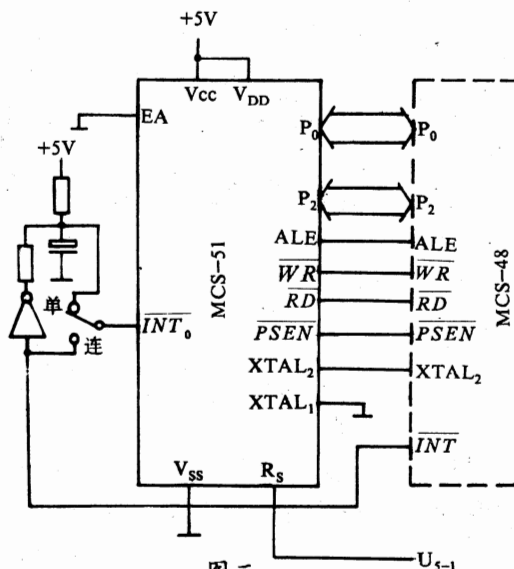
各种功能, 因此相对于单片单板机类型的开发装置, 软件的规模要小得多, 但功能却增强了。图二是本开发装置的工作流程图。值得说明的是由于监控程序的单片机部份采用了浮动地址编程, 所以对于使用 RAM 扩展板的用户, 为了能获得较大的存贮空间, 可以根据需要, 将监控程序定义到任何适当的位置。



图二

#### 七、8031 的开发

本开发装置用于 8031 的开发时, 只要再增加一个 40 脚插座和少量硬件就可以实现。其工作原理也是与上述相同的。图三是开发板 8031 部份的原理图。



图三

## 八、存储器的分配

当与 TP-801B 单板机相配时,本开发装置用于 MCS-48 的开发时,用户可以得到完整的 4K 程序存储器,直接数据区约 6 页,监控程序只占用 MCS-48 内部几个单元。

当用于 MCS-51 的开发时,用户有 7.5K 连续的程序存储器及数据区,而监控程序只占用 MCS-51 内部两级堆栈。

这个存储区分配,对于大多数智能仪表、自动化控制,已是足够了。本开发装置经过几年使用,已发挥了较好的作用,它既是单片计算机的学习机、又是功能齐备的开发装置。(现有功能它可和市售的单片单板

机相比)一些原本买不起单片机开发装置的单位,现在只要花不多的钱,买一块本文介绍的 JS 开发仿真板,配上已有的单板机,就可组成一台功能齐备的 MCS-48, MCS-51 两用开发装置。

## 参考文献

[1]徐爱卿等:《单片微型计算机及其应用》,北京航空学院出版社,1986,4

[2]王长胤,鄢定明,刘大凯编:《微计算机原理及其应用》,湖北科技出版社,1984

[3]林匡定等译:《Intel 单片微机手册》,上海市微型电脑应用协会,1987。

(上接 62、63 页)

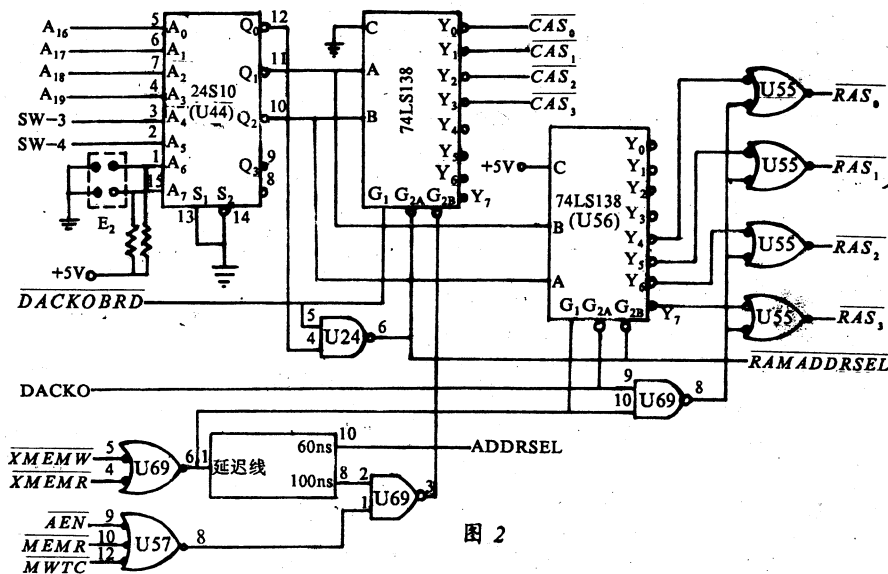


图 2

## 作者来函更正

贵刊 91 年第 2 期第 58 页图 3 中 74LS04 的 6 足与 9 足少划一短路线,特此更正,请谅!

胡红

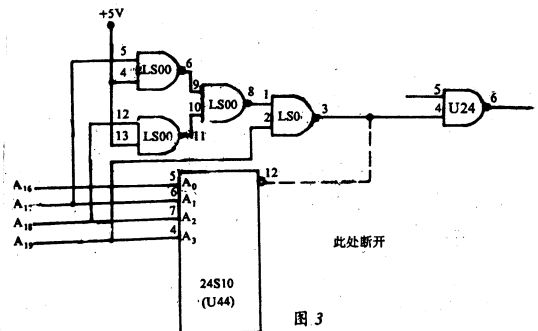


图 3

## 单片机与 A/D 转换器的简易连接

重庆大学 孙劲夫

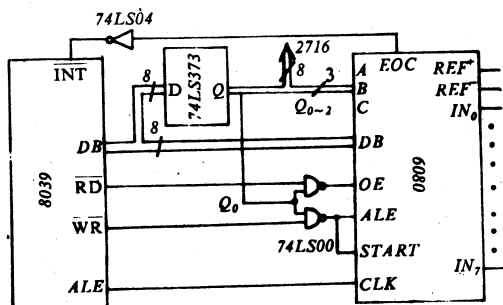
**摘要** 本文讨论单片微机 Intel 8039 两种常用 A/D 转换器芯片 ADC0809 和 ICL 7135 的接口问题,给出了简便实用的硬件电路及软件程序。

仪器仪表的智能化技术越来越多地使用了单片微型计算机和 A/D 转换器,常常会碰到单片机与 A/D 的接口电路问题。本文介绍的单片机与 A/D 转换器的接口电路,简单实用,价格低廉,适合各种不同的场合。单片机选用 Intel 公司的 8039HL 芯片,时钟周期为 1.4  $\mu$ S 到 15  $\mu$ S,包括有 CPU, 128 字节 RAM 和三个 8 位的 I/O 口,具有定时、计数、数据运算处理和输入输出的功能。

### 一、与 ADC0809 的连接

ADC0809 是一种常用的 8 路 8 位双积分型 A/D, 常用钟频为 640KHz, 完成一次转换需 100  $\mu$ S, 单一正 5 伏电源供电, 输入电压信号为 0~5 伏, 片内带有地址译码器和锁存器以及多路开关, 以选通 8 路通道中的一路, 输出有 TTL 三态锁存缓冲器, 便于与各种微处理接口匹配。

Intel 8039HL 与 ADC0809 的接口电路如图一:

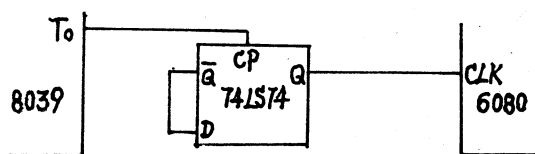


图一 8039 与 0809 的接口电路

图一的接口电路以中断方式采集数据, 0809 作为 8039 的外部数据存储器, 由 8039 的间接寻址指令访问, WR 与 RD 信号与通道选通的低位信号与非后接到 0809 的 OE 端与 ALE 端, 实践中, 与非门输出接一上拉电平, 以保证 TTL 与 CMOS 电路在高电平时的匹配。

图一中, 8039 的 ALE 信号为 400KHz (对于 6MHz

晶振), 0809 完成一次转换需 160  $\mu$ S。在需要高速采样的场合, 可将图一电路稍加改动如图二。



图二 高速采样时 0809 时钟信号的连接

图二

图二中, T0 作为 8039 的时钟信号输出, 频率为 2MHz, 经二分频后作为 0809 的时钟信号, 此时完成一次转换需 64  $\mu$ S。相应的软件上在初始化时需将 T0 端置为时钟信号输出方式。

采样程序为:

```
MOV R0, INH; 关采样通道数,
MOV R1, DATAH; 送数据区指针,
MOVX R0, A; 启动 0809,
HLT; JMP HLT; 等待,
RET; 返回,
```

中断服务程序为:

```
MOVS A, R0; 从 0809 取数,
MOV R1, A; 保存数据,
ET; 开中断,
RETR; 返回。
```

### 二、与 ICL7135 的连接

ICL7135 是专用于仪器仪表的  $4\frac{1}{2}$  位 A/D 芯片。

其特点是 CMOS 工艺又积分工作方式, 具有很高的抗工频干扰的能力, 转换结果以多路调制扫描 BCD 码输出, 缺点是转换速率较慢 (2—10 次/秒), (下转 62 页)

# 数字信号处理器与高速 A/D 卡接口技术

中国科学技术大学 庞文宁 朱科军

**摘要** 本文介绍将 TMS320C25 处理器可寻址的后 32K 数据存储器直接作为高速 A/D 卡的采样缓冲区的接口技术。该技术在采样速率同处理速度相匹配的前提下,可充分发挥数字信号的实时处理能力。

**一、引言** 从科学研究到文化娱乐,数字信号处理几乎无处不在。如果说八十年代是微型计算机的年代,则九十年代将是数字信号处理器(DSP)大显身手的年代。

DSP 是数字信号处理器(Digital Signal Processor)的简称。八十年代初在美国最先进入市场。数字信号处理器以其高速运算的独特优点,成为科研、技术工程等诸多方面不可缺少的工具。数字信号处理器芯片的用法和通用微处理器相似,可以用不同的程序实现不同的功能,然而它比微处理器的运算能力要强二、三个数量级。

信号要变成数字才能处理,但大多数信号是模拟量,要处理信号首先要对它进行模数变换,而各种应用范围信号的频率不同,所以要求模数变换的采样率也不同。怎样达到采样率与处理速度相匹配呢?可以从两方面考虑:

1. 从发挥某一种数字信号处理器的最高处理速度为出发点,选择采样率,这样得到的是一种通用系统。

2. 从某一应用的要求出发,以所需的采样率为前提,配以适应的数字信号处理器,得到的是一种高效的专用系统。

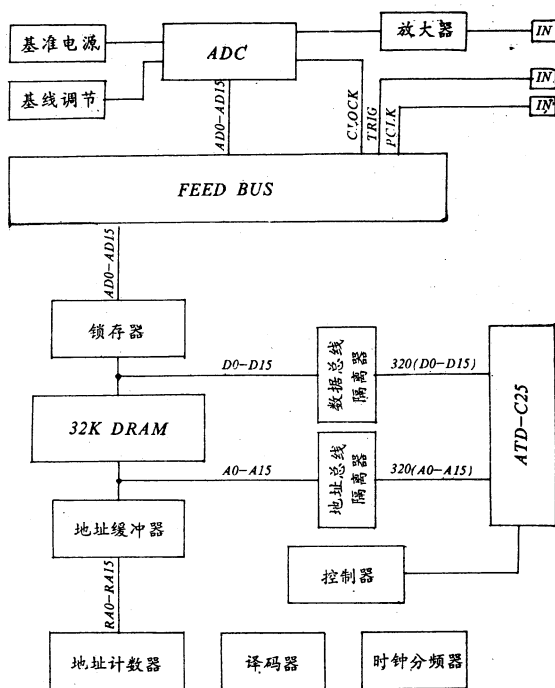
不管那一种方法,都希望把数字采集部分同处理部分直接耦合。对采集部分说,要能连续输出数据而不丢失数据,对处理部分说,不能因为数据的采集而插入等待周期,处理器必须能以它的全速进行操作。

我们采用的是第一种方法。所做的工作是在 ATD—C25 上完成的。ATD—C25 是北京中软计算机研究所研制的 TMS320C25 开发/并行处理系统,有众多的用户,在国内有较大的影响。

## 二、硬件结构

F9321 包括高速 ADC 采集卡(F9321A)和接口卡(F9321M0)。F9321 系快速电子学实验室系列产品编号。图 1 表示了 F9321 与 ATD—C25 构成一个高速采集和高速处理系统的框图。

高速 ADC 采集卡,最高采样率为 10MSPS,分辨率



8bit。它是由 ATD—C25 控制的,C25 通过接口卡和数  
据馈送总线控制高速采集卡的工作。实验室内部称这  
组数据馈送总线为 FEEDBUS(Fast Electronics Lab inter-  
nal Exchanging Data Bus)。FEED 总线有 16 位数据线,3  
根控制线,串行数据和控制线,备用线,电源线和地线。  
这充分考虑到扩展的可能性。F9321A 采集的数据通过  
接口缓冲锁存,直接存储在 C25 可寻址的后 32K 数据  
存储器中。C25 的存储器为 16 位字长,F9321A 只用低  
位字节,而将高位字节置 0。

F9321M 接口板上包括数据,地址总线隔离缓冲  
器,总线转换控制器,地址计数器,I/O 译码器,时钟及  
时钟分频器,采样长度锁存器。接口板上有插座同  
ATD—C25 上处理器仿真插针直接连接。在接口板上  
配置了 C25 可直接寻址的后 32K 数据存储器。F9321

# MCS—51 系列单片机在数字逻辑 分析与设计中的应用

电子科技大学 汪亚南 秦正南 黄丽莹

**摘要** 本文通过具体实例介绍 MCS—51 系列单片机的布尔处理器在逻辑电路分析与设计中的应用,并与通用微处理器及 MCS—48 单片机的同一功能作比较,从而体现 MCS—51 系列单片机的优越性。继而进一步介绍当前利用 51 单片机进行程序逻辑开发的实例,将更有利于单片机的推广应用。

## 一、引言

MCS—51 系列单片机由于其硬件特性和结构的改进以及具有新的指令系统,因而可用于较复杂的实时控制和计算机智能终端等。MCS—51 系列的三种机型(8051、8031、8751)的引脚是完全兼容的,这使得开发的工作量较小,而灵活性很大。它是世界上第二代标准的微控制器,也是当前工业控制系统中理想的机种,被誉为“控制领域中最最佳的 8 位微机”。

MCS—51 系列单片机的最突出特点之一是其内部具有按位操作的位处理器,又叫布尔处理器。布尔处理器实际上是一个完整的 1 位机。虽然它只是 8051 整个结构中的一部份,但它有自己的指令系统、累加器、可寻址的 RAM 和 I/O 口,故也可以看成是一个独立的位处理器。所以, MCS—51 单片机是把 8 位微机和 1 位微机结合在一起,这在微机技术上是一种创新。1 位机在开关量检测、逻辑电路仿真和实时控制方面非常有效;而 8 位机在运算处理、数据采集等方面有明显优势。因而使得 MCS—51 单片机足以胜任各种复杂系统中的应用。

## 二、布尔处理器及其逻辑仿真应用

MCS—51 单片机把 8 位机和布尔处理器的硬件资源复合在一起,构成布尔处理器是 CPU、累加器、程序存储器、数据存储器 and I/O 口,并有 17 条专用指令组成的指令集。

• CPU 和累加器 8051 的 CPU 具有一些特殊的硬件逻辑,使 CPU 能够按位操作。在布尔处理器中,原来 CPU 的进位标志 C 作为按位操作的位置累加器使用。位置累加器 C 是布尔处理器的核心。位操作指令可在任何一个可寻址的位和进位标志 C 之间完成逻辑与/或的位操作,并把结果送回到进位标志。

• 程序存储器 由于位处理指令仍是以字节为单位(指令长度分别为 1、2、3 字节),故布尔处理器中的程序存储器仍使用原程序存储器。

• 数据存储器 存尔处理器可用的数据存储器是内部 RAM 中字节地址为 20H~2FH 中的一段,共 16 字节,128 位,按位寻址的地址为 00H~7FH。位操作指令使得在片内数据 RAM 中的 128 个位和在专用寄存器中的 128 个位能直接寻址。

并不影响 ATD—C25 的工作,仅是把后 32K 数据存储器搬到接口板上而已,原来在 ATD—C25 开发的程序照旧可以使用。接口板上的控制部分,接受 C25 发来的指令,把它们变成各种控制信号,通过 FEED BUS 传到 F9321A 上。接口板使用了一些 C25 的 I/O 口来传递命令。凡符合 FEED BUS 规范的 ADC 采集卡都可与 F9321M 连接。这样就为今后的开发留出了余地。

## 三、工作原理

ATD—C25 作为采样部分的控制者,C25 将采集部分初始化(包括采样长度,采样频率选择)后,指定后 32K 数据存储器作为采集部分的数据缓冲区。采样长度可由用户自己决定,程控选择后由采样长度锁存器锁存,当触发信号到时,F9321A 开始采集数据,记录的数据直接写入 32K 数据存储器,此时 C25 被禁止使用后 32K 数据存储器,但不影响 C25 其它工作。当采样结束时,接口板就向 C25 发出中断请求,通知 C25 处

理器采样已结束,同时将后 32K 数据缓冲器交还给 C25 使用,即 C25 可立即处理采样的数据。由于这 32K × 16RAM 是双端口,不需插入等待周期,因此它的处理速度不降低。

当 ADC 采集卡工作时,ATD—C25 的后 32KRAM 的数据总线、地址总线通过总线隔离缓冲器处于高阻状态,切断 ATD—C25 与这块数据存储器的联系。而 ADC 采样缓冲区的地址由地址计数器给出。记录采样数据通过锁存器写入后 32K 数据存储器。采样结束后则自动释放总线。在接口控制器的控制下,使得 ATD—C25 与 ADC 共享后 32K 数据存储器,使得它们之间可协调工作。

## 参考文献

- [1]李昌立:“数字信号处理器 DSP 及其新成果”,《计算机世界月刊》,1988. 12.
- [2]《TMS320C25 用户手册》内部翻印。



CPU 中累加器 ACC 与寄存器 B 既可用于按字节操作,又可用于按位操作,有 16 个位地址。它们可用于存放位变量或用作软件标志。

• I/O 口 布尔处理器的 I/O 口,因 8051 有 4 个 8 位的 I/O 口,并且每位都可独立操作,故这 4 个 8 位 I/O 口共有 32 个布尔口。每个布尔口都可通过软件设置作为输入、输出或状态输入、选通输出。此外,为了对定时器和串行口等外设进行控制和测试,设有 16 个位寻址寄存器(TCON · 0~TCON · 7,SCON · 0~SCON · 7);为了对 5 个硬件中断源进行控制,设有 11 个位寄存器(IE · 0~IE · 4,IE · 7,IP · 0~IP · 4);还有程序状态字寄存器中 7 个标志位寄存器(PSW · 0, PSW · 2~PSW · 7);P3 口用于 I/O 口及外围设备的控制,有 8 个位寄存器(P3.0~P3.7)。所以,布尔处理器可用的位寄存器总共有 42 个。连同布尔 I/O 口则共有 66 位可以位寻址的位寄存器(有 8 位重迭),它们分布在片内 RAM 区的 80H~FFH,即特殊功能寄存器内。

布尔处理器可作多种应用。这里仅介绍它在逻辑仿真方面的应用。众所周知,逻辑函数有四种表示方法,即真值表、函数表达式、逻辑图和卡诺图,并且它们之间可以相互转换。利用 MCS—51 布尔处理器进行逻辑仿真时,首先必须将逻辑图或真值表转化为相应的逻辑函数表达式,而布尔处理器给逻辑表达式直接转换成软件提供了简单的方法,它不需要大范围地传输数据,屏蔽字节、测试和转移的树形网络就可以解决复杂的逻辑问题。

#### 1 组合逻辑电路求解

在组合逻辑电路分析中,常遇到求解组合逻辑方程的问题。由于 MCS—51 有丰富的位操作指令,用来解这类方程特别方便。

例 1:已知某一 TTL 组合逻辑电路,如图 1 所示。其布尔代数式如下: $Q = U \cdot (V + W) + (X \cdot \bar{Y}) + \bar{Z}$ ,

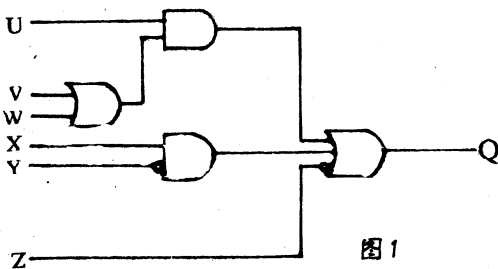


图 1

此逻辑函数表达式可用卡诺图或布尔代数方法进行化简。但随着逻辑式的复杂程度上升,化简难度也将增加。现用 MCS—51 布尔指令系统求解该 U—Z 六个布尔变量的函数,则可非常简化。

设 U、V——P1 口的输入脚(P1.0—和 P1.1)

W、X——两个外围控制器的状态位(P1.2 和 P1.3)

Y、Z——程序中设置的软件标志(P1.4 和 P1.5)

Q——所求的结果。此为 P1 口的输出脚(P1.7),可从累加器得此结果,改变 P1 口的输入变量值,则 Q 改变。

程序清单 1:

```

1:                                ORG 8000H
2:8000 7590FF                    MOV P1, #0FFH
3:8003 A291                      MOV C, P1.1
4:8005 7292                      ORL C, P1.2
5:8007 8290                      ANL C, P1.0
6:8009 92F0                      MOV F0H, C
7:800B A293                      MOV C, P1.3
8:800D B094                      ANL C, /P1.4
9:800F A095                      ORL C, /P1.5
10:8011 72F0                     ORL C, F0H
11:8013 E4                      CLR A
12:8014 33                      RLC A
13:8015 22                      RET
14:                                INCLUDE IN HASHTAB,

```

例 2:今给定逻辑表达式  $Q = [(C + D) \cdot B \cdot \bar{A} + E \cdot F \cdot G + H]$  其逻辑图和逻辑判断流程图,如图 2—(a)和(b)所示。设

P1.0——变量 A 输入端口 P1.4——变量 E 输入端口  
P1.1——变量 B 输入端口 P1.5——变量 F 输入端口  
P1.2——变量 C 输入端口 P1.6——变量 G 输入端口  
P1.3——变量 D 输入端口 P1.7——变量 H 输入端口

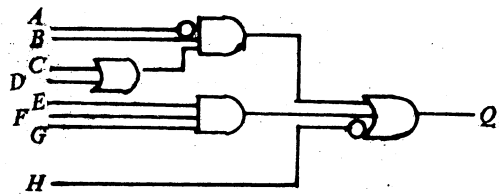


图 2—(a)逻辑图

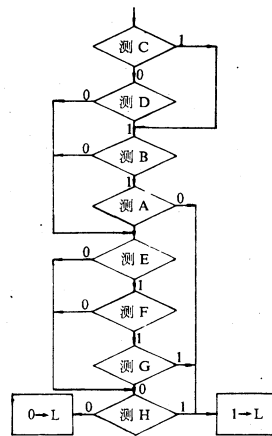


图 2—(b)逻辑判断流程图

## 程序清单 2:

1:			ORG	8000H	
2:	8000	209203	TESTC,	JB	P1.2, TESTB
3:	8003	309306		JNB	P1.3, TESTE
4:	8006	309103	TESTB,	JNB	P1.1, TESTE
5:	8009	309010		JNB	P1.0, SETBL
6:	800C	309406	TESTE,	JNB	P1.4, TESTH
7:	800F	309503		JNB	P1.5, TESTH
8:	8012	209607		JB	P1.6, SETBL
9:	8015	309704	TESTH,	JNB	P1.7, SETBL
10:	8018	C2F0		CLR	F0H
11:	801A	8006		SJMP	DONE
12:	801C	D2B0	SETBL,	SETB	P3.0
13:	801E	A200		MOV	C
14:	8020	8001		SJMP	TT
15:	8022	C3	DONE	CLR	C
16:	8023	E4	TT,	CLR	A
17:	8024	33		RLC	A
18:	8025	22		RET	
19:			INCLUDE IN HASHTAB,		
20:		TESTC,		8000	
21:		TESTB,		8006	
22:		TESTE,		800C	
23:		SETBL,		801C	
24:		TESTH,		8015	
25:		DONE,		8022	
26:		TT,		8023	

若使用 51 单片机的布尔变量逻辑操作指令, 将使程序更为简化和直观。

## 程序清单 3:

1:		ORG	8000H	
2:	8000	7590FF	MOV	P1, #
3:	8003	A292	MOV	C, 0FFH
4:	8005	7293	ORL	C, P1.2
5:	8007	8291	ANL	C, P1.3
6:	8009	B090	ANL	C, P1.1
7:	800B	92F0	MOV	F0H, /P1.0
8:	800D	A294	MOV	C, P1.4
9:	800F	8295	ANL	C, P1.5
10:	8011	8296	ANL	C, P1.6
11:	8013	72F0	ORL	C, F0H
12:	8015	A097	ORL	C, /P1.7
13:	8017	E4	CLR	A
14:	8018	33	RLC	A
15:	8019	92F0	MOV	F0H, C

16, INCLUDE IN HASHTAB,

例 3: 8 输入端的组合逻辑电路

已知该组合逻辑电路的逻辑图如图 3 所示。

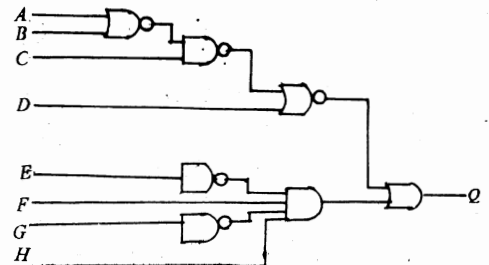


图 3 逻辑图

由此可得布尔表达式  $Q = \overline{A} \cdot B \cdot C \cdot \overline{D} + E \cdot F \cdot \overline{G}$

• H

解<一>:采用 MCS-48 系列单片机实现,则求解逻辑函数 Q 的模拟程序及其软件流程图为图 4-(a)和(b)。

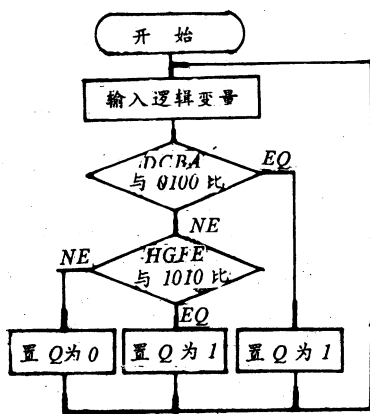


图 4(a) 软件流程图

源程序 4:

A>type

```
START: MOV A, #0FFH
        OUTL P1, A
LOOP: IN A, P1
        MOV R2, A
        ANL A, #0FH
        XRL A, #04H
        JZ THERE
        MOV A, R2
        ANL A, #0F0H
        XRL A, #0A0H
        JZ THERE
        ANL P2, #7FH ; Q=0
        JMP LOOP
THERE: ORL P2, #80H
        ANL P2, #0BFH ; Q=1
        JMP LOOP
HERE: ORL P2, #80H ; Q=1
        JMP LOOP
```

图 4-(b) 逻辑函数的模拟程序

解<二>设 8 个逻辑变量 A~H 从 P1 口输入, Q 从 P2 口(P2.7)输出。若用 MCS-51 系列单片机,则模拟

程序如程序清单 5 附后。

程序清单 5 表明无论逻辑电路的函数表达式多么复杂,其布尔代数式均可用逻辑运算指令表示。并且用布尔变量逻辑指令求解逻辑方程的模拟程序是很简便的。

利用软件实现硬件逻辑的功能,可省去逻辑门电路的设计,而且可使系统运行可靠,减少故障率。

把硬件逻辑转变为逻辑仿真程序是将硬件软化,它所带来的优点是明显的。但这将使运算处理速度降低, MCS-51 单片机即使采用 12MHz 晶体运行布尔仿真程序,至少也需几十微秒。例如, 51 单片机运行仿真程序 1, 最快也需 20 微秒, 即从 I/O 口输入变量到逻辑处理后, 从 P1.7 口输出结果要延时 20 微秒。因此, 对于高速逻辑仿真, MCS-51 布尔处理器难以实现。

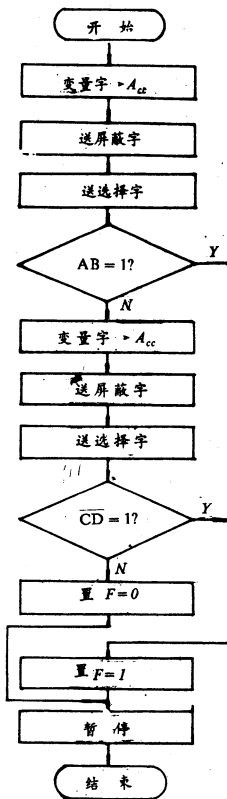


图 5

图 5 逻辑函数模拟流程图

## 2 组合逻辑函数的模拟

在实时控制和智能仪器应用领域中,对逻辑判断

或逻辑函数的模拟很有实用价值。现举一例以说明组合逻辑函数模拟程序的设计。

给定  $F(A, B, C, D) = A\bar{B} + \bar{C}D$ , 现用程序模拟该函数。

通常, 用程序来模拟硬件逻辑都要先画出程序框图。见图 5。

设计思路: 采用多级的判断分支程序结构。对每个逻辑乘项用一个判断环节模拟, 对各个逻辑乘项之间的逻辑加运算则用多级判断模拟。程序清单 6 附后。

当函数自变量数目不超过 8 时, 可在内存 RAM 中选一工作寄存器, 代替实际应用中的端口寄存器, 用以存放函数的自变量, 再选用程序状态字寄存器 PSW 存放函数的逻辑值。本例中, 设函数自变量的安排如下, 当各变量取不同的逻辑值 1 或 0 时, 则 PSW. 5 的 F 将根据  $A\bar{B} + \bar{C}D$  取值。

R <sub>1</sub>	A	X	B	X	C	D	X	X
----------------	---	---	---	---	---	---	---	---

PSW		PSW. 5	
-----	--	--------	--

### 三、继电器控制逻辑的程序模拟

在工业控制中, 许多硬件逻辑结构是由一些继电器的串、并联支路构成。给出一组继电器触点逻辑结构, 则可画出其等效逻辑电路。这类控制逻辑, 也完全可用单片机来实现。下面列举用 8031 单片机系统对继电器控制逻辑的模拟实例, 在某继电器自动检测、逻辑运算和控制系统中, 有各种功能的继电器, 其中有些继电器开关的作用为完成逻辑运算, 这可利用 8031 单片机的布尔处理器的位来代替这些继电器开关。如何实现用 8031 布尔处理器的位来代替逻辑运算继电器开关呢?

1 等效原理: 逻辑运算器的一个开关  $K_i$  可由布尔处理器中的一位可寻址位  $B_i$  来代替。当开关  $K_i$  断开时, 给 8031 的 P1 口 (如 P1. 0) 输入一个“0”信号, 由程序控制置  $B_i$  位为“1”, 表示  $K_i$  断开; 当开关  $K_i$  合上时, 则给 8031 的 P1. 0 口输入一个“1”信号。由程序控制清除  $B_i$  位为“0”, 表示该逻辑运算器开关  $K_i$  接通。

#### 2 等效逻辑电路

图 6-(a) 逻辑运算继电器开关的触点控制逻辑

(b) 等效逻辑电路

(c) 改用 8031 的直接寻址位进行逻辑运算后所等效的一个输出位 (开关)

现分别用  $B_1 \sim B_7$  (位) 代替  $K_1$  等开关。

当  $B_i = "0"$  时, 对应于  $K_i$  接通; 当  $B_i = "1"$  时, 对应于  $K_i$  断开。因此, 该等效关系成立: 即

$$OK = B_1 \cdot (B_2 + B_3 + B_4 + B_5 + B_6 \cdot B_7)$$

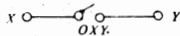
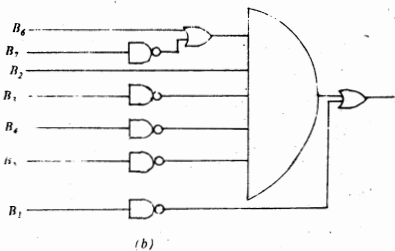
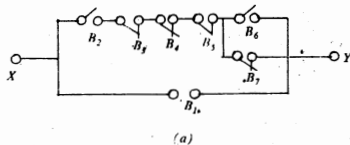


图 6

### 3 逻辑运算电路的模拟程序

在图 6-(C) 电路中, 设 OK 为 8031 系统中输出接口的 D5 位, 该输出接口的口地址为片内 RAM51H 单元。B<sub>1</sub>、B<sub>2</sub>、B<sub>3</sub>、B<sub>4</sub>、B<sub>5</sub>、B<sub>6</sub>、B<sub>7</sub> 为片内 RAM22H 单元的 D<sub>0</sub> ~ D<sub>6</sub> (位地址为 10H ~ 16H)。该电路的逻辑运算程序清单附后。

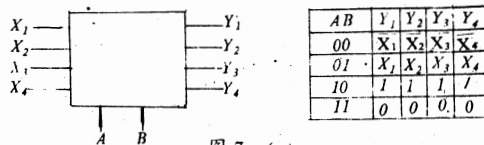


图 7-(a)

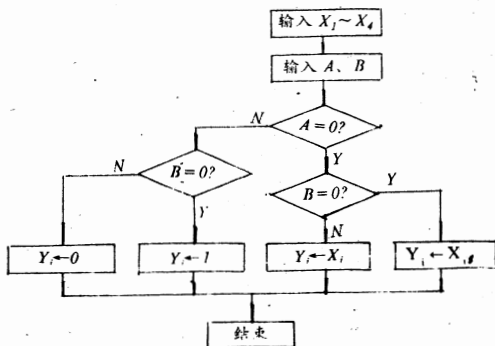


图 7-(b)

#### 四、程序逻辑的开发

硬件逻辑电路有多种实现方法,如分立元件、集成电路、可编程逻辑器件和大规模集成电路的方阵列等。由上述诸多例子也可看出:硬件逻辑电路可通过程序逻辑来实现。程序逻辑是一个强有力的逻辑设计手段。采用程序逻辑可便于设计的修改和更新。

当前,各种微机控制系统也多用程序逻辑来实现硬件逻辑结构。尤其是时序逻辑系统,皆采用程序控制而不用硬件逻辑实现。这里介绍利用单片机进行程序逻辑的开发。

MCS-51 系列单片机可用来开发程序逻辑。为此,只需输入硬件逻辑电路的真值表、状态转移表、状态表等表格形式,系统就能自动生成单片机控制的模拟程序,省去编程和调试,这样可以加速程序逻辑的开发。程序逻辑开发主要包含两个部分:数据生成程序和程序逻辑执行程序。首先运行数据生成程序,在运行时,输入硬件逻辑的原始真值表或状态表,利用生成程序将这些表格转换成执行程序所需的形式。并能写入到单片机的存储器中去。

程序逻辑执行程序完成由输入值到输出值的转换。即实现具体的控制功能,然后把生成的数值和执行程序一起写入到 EPROM 中。

1 实例:模拟一个具有双控制信号端的 4 路输入/输出电路,其输入与输出关系,如图 7 所示。

图 7-(a) 输入-输出关系 (b) 模拟程序框图在控制信号 A、B 作用下,电路输出  $Y_i$  可为 0, 1,  $X_i$  和  $\bar{X}_i$ 。

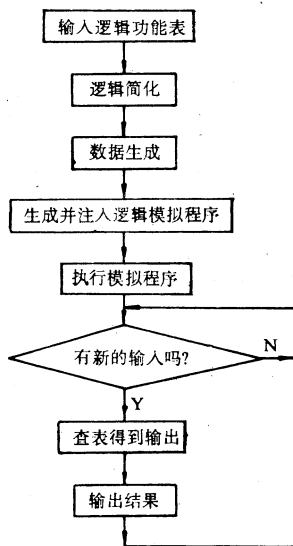


图 8: 程序模拟过程框图

2 程序模拟过程 本例的程序模拟基本过程,如图 8 所示。

- 输入逻辑功能表(即真值表或状态表),本例中有 6 个输入变量(A, B,  $X_1$ ,  $X_2$ ,  $X_3$ ,  $X_4$ )和 4 个输出变量( $Y_1$ ,  $Y_2$ ,  $Y_3$ ,  $Y_4$ )。因而真值表将有  $2^6=64$  列故必须化简。

- 逻辑简化,用来简化原始的状态表或真值表,经过化简以后,将大大压缩生成的数据量,减少数据占用的存储空间,此项工作由逻辑简化程序完成。

- 数据生成,由数据生成程序将化简后的真值表阵列进行数据变换成执行逻辑模拟程序所需要的形式,并连同模拟程序一起写入存储器。

- 执行模拟程序,查表求得输出。在计算机中,程序设计经常会遇到表格和查表。表格是属于一种数据存储结构,其中线性表是最常用、最简单的一种数据结构。所谓线性,就是一组有序的数据元素,根据不同的线性表格,查表的方法不同,对于某些有序表格的查找,通常采用计算查表法。例如,从表格基地址开始存放数据元素,在查表时,先得到欲查找的关键项或“字”(标识该记录的数据项称为关键项)的位置,该位置是相对于表格基地址的相对位置,即偏移量,再由偏移量算出待查找的关键项的地址。

真值表阵列中的每项皆包含输入变量和输出变量,其中  $ABX_1X_2X_3X_4$ (输入部分)用  $P_i$  表示,  $Y_1Y_2Y_3Y_4$ (输出部分)用  $P_o$  表示。在执行模拟程序时,当取得一个新的输入值  $P$ ,且满足条件  $P \wedge P_i = P$  时,则对应的  $P_o$  即为要查找的输出,也就是从  $Y_1Y_2Y_3Y_4$  输出。符号“ $\wedge$ ”表示两项的按位逻辑“与”操作。以后再等待下一个新的输入。这样就实现了硬件逻辑功能。由于单片机具有丰富的位操作指令,故所用指令数目较通用微机为少,因此,这更有利于程序模拟的开发。

对于时序逻辑电路的模拟也是利用查表法。此时是查状态表。由于时序逻辑电路的输出不仅与当时的输入有关,而且与前时刻的输入状态有关,故在系统中要有状态寄存器,这可利用内部 RAM 单元。在取得新的输入值时,结合状态寄存器中的内容查表,求得输出和下一状态。下一状态则存入状态寄存器以备下次查表时用。因此,在程序逻辑开发中,组合逻辑和时序逻辑有不同的数据生成程序及模拟查表程序。

当输入输出变量数目较多时,则硬件逻辑电路复杂,成本也高,特别适宜于采用程序逻辑。但程序逻辑存在运行速度较低的缺陷,因此,它不能完全取代硬件逻辑。

## 程序清单 5:

```

1:                                ORG      8000H
2:      8000      7590FF      MOV      P1      #0FFH
3:      8003      A292      MOV      C,      P1.2
4:      8005      B091      ANL      C,      /P1.1
5:      8007      B090      ANL      C,      /P1.0
6:      8009      B093      ANL      C,      /P1.3
7:      800B      92F0      MOV      F0H,      C
8:      800D      A295      MOV      C,      P1.5
9:      800F      B094      ANL      C,      /P1.4
10:     8011      B096      ANL      C,      /P1.6
11:     8013      8297      ANL      C,      P1.7
12:     8015      72F0      ORL      C,      F0H
13:     8017      E4      CLR      A
14:     8018      33      RLC      A
15:                                INCLUDE IN HASHTAB;

```

## 程序清单 6:

```

                                ORG      8000H
8000      75D000      MOV      PSW,      #00H
8003      7914      MOV      R1,      #14H
8005      E9      MOV      A,      R1
8006      54A0      ANL      A,      #0A0H
8008      6420      XRL      A,      #20H
800A      700D      JNZ      REL
800C      7A14      MOV      R2,      #14H
800E      EA      MOV      A,      R2
800F      540C      ANL      A,      #0CH
8011      6408      XRL      A,      #08H
8013      7004      JNZ      REL
8015      C2D5      CLR      PSW.5
8017      8002      SJMP      DONE
8019      D2D5      REL;      SETB      PSW.5
801B      22      DONE;      RET

```

INCLUDE IN HASHTAB;

REL; 8019

DONE; 801B



## 程序清单 7:

```

1:      0000      A216      MOV      C,      16H
2:      0002      8215      ANL      C,      15H
3:      0004      7214      ORL      C,      14H
4:      0006      7213      ORL      C,      13H
5:      0008      7212      ORL      C,      12H
6:      000A      7211      ORL      C,      11H
7:      000C      8210      ANL      C,      10H
8:      000E      400B      JC      OFF
9:      0010      E551      MOV      A,      51H
10:     0012      54DF      ANL      A,      #0DFH
11:     0014      F551      MOV      51H,      A
12:     0016      90A000     MOV      DPTR,      #0A000H
13:     0019      F0       MOVX     @DPTR,      A
14:     001A      22       RET
15:     001B      E551      OFF:     MOV      A,      51H
16:     001D      4420      ORL      A,      #20H
17:     001F      F551      MOV      51H,      A
18:     0021      90A000     MOV      DPTR,      #0A000H
19:     0024      F0       MOVX     @DPTR,      A
20:     0025      22       RET
21:                                     INCLUDE IN HASHTAB,

```

(上接 63 页)

```

endi
endi
else
    sr=substr(dx,t+t,2)+substr(dy,i+i,2)+sr
    store l to fl,flg
endi
i=i+1
endd
if dm=0
    sr=sr+'整'
else
i=i+i
dm=100 * dm
t=int(dm/10+.01)
rem=int(dm-10 * t+.1)
if t<>0

```

```

sr=sr+substr(dx,t+t,2)+'角'
if rem<>0
    sr=sr+substr(dx,rem+rem,2)+'分'
endi
else
    sr=sr+'零'+substr(dx,rem+rem,2)+'分'
endi
endi
?? sr

```

该程序的基本设计思想是:首先把欲转换的数据整数和小数部分分离,然后再把各位分离(可以从高位到低位,也可从低位到高位,既可以用数学方法,也可以用字符串的方法分离)。若分离出的某位不为0,就把它所对应的大写数字及位级连接到大写字符串中,否则,要视其所在的位置和前后关系做相应标记或处理。

(郑州黄河大学 叶志斌)

## 排除 COMPAQ/20E 一故障之所得

甘肃计算中心 戴若兰 薛和华

### 一、故障现象分析与定位

在 compaq/20e 计算机上增置一个 1.44MB 3.5 吋软驱。完成了物理安装与逻辑安装后,在该驱动器中插入软盘片,且列目录清单。B>Dir 屏幕显示:General Failure error reading drive B, Abot, Retry, Fail?

按 A↙, R↙, F↙ 任一健,均死机。

为了断定是否软盘片故障,换一片插入,现象相同。因机器原先运行正常,3.5 吋软驱是新加的,故考虑,软盘驱动器有故障。为判断软驱是否能写,插入软盘片,进行格式化 Format,格式化可进行,但待各磁道格式化完成后,屏幕仅出现:Completed Format,但盘容量小结不显示,却又显示:General Failure error reading drive B, Abot, Retry, Fail?

按 A↙, R↙, F↙ 任意一健,均死机。

看起来,故障似乎说明,驱动器可写不可读,可能是读电路故障。为进一步验证,插入软盘片进行拷贝 COPY,却一个文件也拷不上。

为了断定该驱动器是否有故障,另换一驱动器装上,结果现象相同。这说明故障不在驱动器本身。接着又更换了与主机相联的电缆,现象不变。至此,如是硬件故障,只能是位于主机主板上的软盘适配器。但考虑机器原先运行正常,适配器故障的可能性不大。因而故障极有可能是软故障。故将在 B 盘中插过的软盘片插入其它驱动器,竟也发生相同现象。故此可断定故障出在软件上,是系统软件出了故障,还是病毒侵犯?用消毒软盘检查该驱动器读过的软盘,显示:

Scan 1.0 by public security ministry of china scanning for 78 viveres.

Scanning boot sector.

General Failure error reading drive B, .....

这说明,软盘上的信息已被破坏得面目全非,消毒程序也根本无法读,肯定 DOS 系统出了问题。用消毒软盘检查 C 盘,屏幕显示:

Scan 1.0 by public security ministry of china scanning for 78 viveres.

Foud vivus 6.4 in partition table.

即在分区表里发现了 6.4 病毒,而这种病毒是用该软件包杀不死的。

### 二、故障排除

我们知道硬盘分区表和 DOS 主引导程序一起位于硬盘 0 头 0 柱 1 扇区,该扇区是硬盘的隐含扇区排在硬盘逻辑 0 扇区之前。即排在 DOS 分区之前。用 PC-TOOLS 和一般的 DEBUG 命令都不能将其读出,需编一简单的程序将其读出。该程序是:

A>DEBUG↙

-A↙

```
CS:100  MOV DL,80↙      ;驱动器号
          MOV DH,0↙      ;0 号磁头
          MOV CH,0↙      ;0 号柱面
          MOV CL,1↙      ;1 号扇区
          MOV BX,CSEG↙    ;置缓冲区
          MOV AL,1↙      ;读扇区数
          MOV AH,2↙      读磁盘
          INT 13↙        调用硬盘 I/O 驱动
          INT 3↙          设置断点
```

-G=100

这样将硬盘 0 头 0 柱 1 扇区信息读出。该扇区的信息布局是:

000H	主引导记录 (240)字节
0F0H	全 0 (206 字节)
IBEH	第一分区表(16 字节)
ICEH	第二分区表(16 字节)
IDEH	第三分区表(16 字节)
IEEH	第四分区表(16 字节)
IFEH	55H
	AAH

即从 IBEH 字节开始,每 16 个字节组成一分区表,其中 0 字节是自举标志,第四字节是系统标志。1—3 字节是分区的起始地址。1 字节是磁头号,第二字节的低 6 位是扇区号,其最高两位和第 3 字节共同组成 10 位的柱面号。5—6 字节是分区终止地址,5 字节是磁头号,6 字节的低 6 位是扇区号,其最高两位和第 7 字节共同组成 10 位的柱面号,8—11 字节是相对扇区号,12—15 是实用扇区号。

该系统硬盘容量是 84.2MB,6 个磁头,831 柱面,

每道 33 扇区,每扇 512 字节。

在系统初始化时曾分为两区,C 区是 0~299 柱,约 30MB,D 区是 300~830 柱,约 54MB。显示分区表信息:

D21BE 21FE

数据是:

CS,21BE

```
80 01 01 00 04 05 61 2B 21 00 00 00 E7 E7 00 00
00 00 41 2C 05 05 E1 3E 08 E8 00 00 B2 9A 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
55 AA
```

d 1000 11ff

```
122E:1000 B5 7F 72 92 58 E9 B8 00-00 00 00 1A AF 00 F0 06 ...r.X.....
122E:1010 01 80 9F 00 7C 00 00 05-7C 00 00 00 00 00 80 ....|...|.....
122E:1020 FC 02 72 0D 80 FC 04 73-08 80 FA 80 73 03 E8 05 ..r....s....s...
122E:1030 00 2E FF 2E 0B 00 50 53-51 52 06 1E 56 57 0E 1F .....PSQR..VW..
122E:1040 C 07 BE 02 00 33 C0 9C-FF 1E 0B 00 B8 01 02 BB .....3.....
122E:1050 00 02 B9 01 00 32 F6 9C-FF 1E 0B 00 73 05 4E 75 .....2.....s.Nu
122E:1060 E4 EB 2E 33 F6 BF 00 02-B9 03 00 FC F3 A7 74 21 ...3.....t!
122E:1070 B8 01 03 BB 00 02 B9 03-00 B6 01 9C FF 1E 0B 00 .....
122E:1080 72 0F B8 01 03 33 DB B9-01 00 32 F6 9C FF 1E 0B r....3....2....
122E:1090 00 5F 5E 1F 07 5A 59 5B-58 C3 16 16 1D 1D 1A 1D - ^ ..ZY[X.....
122E:10A0 37 37 37 37 37 37 55 7B-78 78 73 6E 36 37 5D 62 777777U{xxsn67]b
122E:10B0 79 39 37 23 3B 37 26 2E-2F 2E 1D 1D 1D 1A 1D 00 y97#;78.../.....
122E:10C0 33 C0 8E D8 FA 8E D0 BC-00 7C FB A1 4C 00 A3 0B 3.....|..L...
122E:10D0 7C A1 4E 00 A3 0D 7C A1-13 04 48 48 A3 13 04 B1 |.N...|...HH....
122E:10E0 06 D3 E0 A3 11 7C A3 4E-00 8E C0 B8 1F 00 A3 4C .....|.N.....L
122E:10F0 00 C7 06 0F 7C 06 01 BE-00 7C 33 FF B9 00 01 FC ....|...|3....
122E:1100 F3 A5 FF 2E 0F 7C 33 C0-CD 13 0E 1F 33 C0 8E C0 .....|3....3...
122E:1110 B8 01 02 BB 00 7C 80 3E-0A 00 00 74 31 B9 06 00 .....|. >...t1...
122E:1120 BA 80 00 CD 13 0E 07 FE-06 08 00 80 3E 08 00 80 .....>...
122E:1130 72 1A C6 06 08 00 7A FC-BE 9A 00 AC 3C 00 74 0C r.....z....<.t.
122E:1140 32 06 03 00 B4 0E B7 00-CD 10 EB EF EB 54 B9 03 2.....T..
122E:1150 00 BA 00 01 CD 13 0E 07-72 1D B8 01 02 BB 00 02 .....r.....
122E:1160 B9 01 00 BA 80 00 CD 13-72 0D BE 00 02 33 FF B9 .....r.....3..
122E:1170 03 00 FC F3 A7 75 0E C6-06 0A 00 00 C6 06 08 00 .....u.....
122E:1180 00 FF 2E 13 00 B8 01 03-BB 00 02 B9 06 00 CD 13 .....
122E:1190 72 E5 BE BE 03 BF BE 01-B9 21 01 F3 A5 C6 06 0A r.....|.
122E:11A0 00 01 B8 01 03 33 DB B9-01 00 CD 13 BE BE 04 BF .....3.....
122E:11B0 BE 01 B9 20 00 F3 A5 EB-BE 0F E3 0A 3E 21 80 01 .....>|..
122E:11C0 01 00 04 05 61 2B 21 00-00 00 E7 E7 00 00 00 00 ....a+|.....
122E:11D0 00 41 2C 05 05 E1 3E 08-00 00 B2 9A 01 00 00 00 .A,...>.....
122E:11E0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
122E:11F0 00 00 00 00 00 00 00 00-00 00 00 00 00 55 AA .....U.
```

将分区表数归纳,可得出

分区	自举标志	系统标志	起始地址	终止地址	相对扇区	实用扇区
主	80	04	1头0柱 1扇	5头299柱 33扇	21H (33)	E7E7H 25MB
	00	05	0头300柱 1扇	5头830柱 830柱	E808H 59400	019AB2H 54MB

可见,主分区实用扇区数不对,按公式:每第道扇区数×柱面号×磁头数+每道扇区数×磁头号+相对扇区

算出主区实用扇区:

$$33 \times 299 \times 6 + 33 \times 5 + 21 = 59400 = E808H = 30.4MB$$

即 21CA 和 21CB 字节的内容应是 08E8H。

主引导程序块占扇区的前 240 字节(0~0EFH),但此时打出的主引导块如下:

而正常系统的主引导块是:

122E:1000	FA 33 C0 8E D0 BC 00 7C-8B F4 50 07 50 1F FB FC	.3..... ..P.P...
122E:1010	BF 00 06 B9 00 01 F2 A5-EA 1D 06 00 00 BE BE 07	.....
122E:1020	B3 04 80 3C 80 74 0E 80-3C 00 75 1C 83 C6 10 FE	...<.t.<.u.....
122E:1030	CB 75 EF CD 18 8B 14 8B-4C 02 8B EE 83 C6 10 FE	.u.....L.....
122E:1040	CB 74 1A 80 3C 00 74 F4-BE 8B 06 AC 3C 00 74 0B	.t.<.t.....<.t.
122E:1050	56 BB 07 00 B4 0E CD 10-5E EB F0 EB FE BF 05 00	V.....^.....
122E:1060	BB 00 7C B8 01 02 57 CD-13 5F 73 0C 33 C0 CD 13	.. ...W...s.3...
122E:1070	4F 75 ED BE A3 06 EB D3-BE C2 06 BF FE 7D 81 3D	Ou.....}.=
122E:1080	55 AA 75 C7 8B F5 EA 00-7C 00 00 49 6E 76 61 6C	U.u..... ..Inval
122E:1090	69 64 20 70 61 72 74 69-74 69 6F 6E 20 74 61 62	id partition tab
122E:10A0	6C 65 00 45 72 72 6F 72-20 6C 6F 61 64 69 6E 67	Ie. Error loading
122E:10B0	20 6F 70 65 72 61 74 69-6E 67 20 73 79 73 74 65	operating syste
122E:10C0	6D 00 4D 69 73 73 69 6E-67 20 6F 70 65 72 61 74	m. Missing operat
122E:10D0	69 6E 67 20 73 79 73 74-65 6D 00 00 00 00 00 00	ing system. ....
122E:10E0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:10F0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1100	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1110	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1120	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1130	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1140	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1150	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1160	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1170	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1180	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:1190	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:11A0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:11B0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 80 01	.....
122E:11C0	01 00 04 05 61 2B 21 00-00 00 08 E8 00 00 00 00	....a+!.....
122E:11D0	41 2C 05 05 E1 3E 08 E8-00 00 B2 9A 01 00 00 00	A,...>.....
122E:11E0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
122E:11F0	00 00 00 00 00 00 00 00-00 00 00 00 00 55 AA	.....U.

可见主引导块亦被破坏得面目全非了。病毒程序替换了原主引导程序。而将主引导程序移到别的扇区,对于一般用户没有必要去分析病毒机理,只要杀死病毒,使机器恢复正常,程序和数据不受损失即可。故可用复盖法恢复系统。即用前面的读盘程序将一相同操作系统版本的正常硬盘上的0面0柱1扇内容读出,再用 C>DEBUG/W CS:CSEG DRIVE S1

将其写入一已格式化软盘(其中:DRIVE是驱动器号A=0,B=1,S是写入扇区号,自定。CSEG是缓冲地址,自定)。将此软盘插入被破坏的系统(该系统不可再用硬盘启动)。用 L CS:CSEG 0 S 1 读入内存,再将前面算出的正确的实用扇区数E808H,用DEBUG的e命令进行修改。再用程序;

MOV DX,0080/;硬盘0磁头

MOV CX,0001/;0柱1扇

MOV BX,CSEG/;置缓冲区

MOV AX,0301/;写1个扇区

INT 13/;调硬盘I/O驱动

• INT 3/;设置断点

写到需恢复的硬盘。至此整个系统便恢复,且用消毒软件包检查未发现病毒。

6.4 病毒侵犯破坏软盘的逻辑0扇区,即引导扇区。对软盘可直接用DEBUG的读写命令,即用 L CS:CSEG 0 0 1 把一相同操作系统版本相同盘容量的引导程序读出,再用 W CS:CSEG 0 0 1 写到被破坏的软盘上即可。

122E,2000	2E FF 2E 17 7C E9 B8 00-00 00 00 12 2E 00 F0 06	.... .....
122E,2010	01 80 9F 00 7C 00 00 05-7C 00 00 00 00 00 80	.... ... .....
122E,2020	FC 02 72 0D 80 FC 04 73-08 80 FA 80 73 03 E8 05	...r...s...s...
122E,2030	00 2E FF 2E 0B 00 50 53-51 52 06 1E 56 57 0E 1F	.....PSQR..VW..
122E,2040	0E 07 BE 02 00 33 C0 9C-FF 1E 0B 00 B8 01 02 BB	.....3.....
122E,2050	00 02 B9 01 00 32 F6 9C-FF 1E 0B 00 73 05 4E 75	.....2.....S. NU
122E,2060	E4 EB 2E 33 F6 BF 00 02-B9 03 00 FC F3 A7 74 21	.....3.....t!
122E,2070	B8 01 03 BB 00 02 B9 03-00 B6 01 9C FF 1E 0B 00	.....
122E,2080	72 0F B8 01 03 33 DB B9-01 00 32 F6 9C FF 1E 0B	r...3...2....
122E,2090	00 5F 5E 1F 07 5A 59 5B-58 C3 16 16 1D 1D 1A 1D	.....ZY[X.....
122E,20A0	37 37 37 37 37 37 55 7B-78 78 73 6E 36 37 5D 62	777777U{xxsn67Ib
122E,20B0	79 39 37 23 3B 37 26 2E-2F 2E 1D 1D 1D 1A 1D 00	y97#,78.../.....
122E,20C0	33 C0 8E D8 FA 8E D0 BC-00 7C FB A1 4C 00 A3 0B	3..... ..L...
122E,20D0	7C A1 4E 00 A3 0D 7C A1-13 04 48 48 A3 13 04 B1	.N... ...HH....
122E,20E0	06 D3 E0 A3 11 7C A3 4E-00 8E C0 B8 1F 00 A3 4C	..... .N.....L
122E,20F0	00 C7 06 0F 7C 06 01 BE-00 7C 33 FF B9 00 01 FC	.... ... 3....
122E,2100	F3 A5 FF 2E 0F 7C 33 C0-CD 13 0E 1F 33 C0 8E C0	..... 3....3...
122E,2110	B8 01 02 BB 00 7C 80 3E-0A 00 00 74 31 B9 06 00	.... .>...tl...
122E,2120	BA 80 00 CD 13 0E 07 FE-06 08 00 80 3E 08 00 80	.....>...
122E,2130	72 1A C6 06 08 00 7A FC-BE 9A 00 AC 3C 00 74 0C	r.....z.....<.t.
122E,2140	32 06 03 00 B4 0E B7 00-CD 10 EB EF EB 54 B9 03	2.....T.
122E,2150	00 BA 00 01 CD 13 0E 07-72 1D B8 01 02 BB 00 02	.....r.....
122E,2160	B9 01 00 BA 80 00 CD 13-72 0D BE 00 02 33 FF B9	.....r...3..
122E,2170	03 00 FC F3 A7 75 0E C6-06 0A 00 00 C6 06 08 00	.....u.....
122E,2180	00 FF 2E 13 00 B8 01 03-BB 00 02 B9 06 00 CD 13	.....
122E,2190	72 E5 BE BE 03 BF BE 01-B9 21 01 F3 A5 C6 06 0A	r.....!.....
122E,21A0	00 01 B8 01 03 33 DB B9-01 00 CD 13 BE BE 04 BF	.....3.....
122E,21B0	BE 01 B9 20 00 F3 A5 EB-BE 0F E3 0A 3E 21 00 00	.....> ..
122E,21C0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00	.....
122E,21D0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00	.....
122E,21E0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00	.....
122E,21F0	00 00 00 00 00 00 00 00-00 00 00 00 00 55 AA	.....U.
42B0,1000	EB 28 90 49 42 4D 20 20-33 2E 33 00 02 01 01 00	.(.IBM 3.3....
42B0,1010	02 E0 00 60 09 F9 07 00-0F 00 02 00 00 00 00 00	.....
42B0,1020	00 00 00 00 00 00 00 00-00 00 FA 33 ED B8 C0 07	.....3....
42B0,1030	8E D8 C4 1E 1C 00 88 16-FD 01 0A D2 79 08 89 1E	.....y...
42B0,1040	24 00 8C 06 26 00 8E C5-8E D5 BC 00 7C FC 1E 36	\$...&..... ..6

(上接 30 页)项隐含值的灵活运用即可加快录入速度,又可节省宝贵时间,而且不影响原数据库任何功能。

Micro CDS/ISIS 的倒排文件,不象通常其它检索系统的倒排文件那样,每个可检索字段都要分别有一个倒排文件,而是对于任何给定的数据库使用一个倒排文件,这是它的显著特点。因此,在数据录入时,对于给定的倒排文件字段选择项目,任意一条重复信息都可不必录入,这样即节省时间又节约内存,一举两得。

#### 四、结束语

Micro CDS/ISIS 为信息管理工作提供了一个非常

方便的现代化管理手段,并为用户创造了一个良好的使用环境,目前它的独到之处已越来越引起各界广泛关注,相信该系统的广泛应用将对我国的微机信息管理起到积极的推动作用。

#### 参考文献

- [1] 吴广印 Micro CDS/ISIS 软件的开发 第八届全国机检会议论文
- [2] CDS/ISIS (Mini—Micro Version, Release 1.0) USER MANUAL

42B0,1050	C5 36 78 00 BF 2A 7C B9-0B 00 F3 A4 1F C6 06 2E	.6x..* .....
42B0,1060	00 0F BF 78 00 B8 2A 7C-AB 91 AB FB 8A 16 FD 01	...x..* .....
42B0,1070	CD 13 A0 10 00 98 F7 26-16 00 03 06 0E 00 E8 73	.....&.....s
42B0,1080	00 E8 79 00 BB 00 05 53-E8 A0 00 5F BE 71 01 B9	..y....S...-..q..
42B0,1090	0B 00 90 F3 A6 75 57 83-C7 15 B1 0B 90 90 F3 A6	.....uW.....
42B0,10A0	75 4C 26 8B 47 1C 99 8B-0E 0B 00 03 C1 48 F7 F1	uL&..G.....H..
42B0,10B0	3D 14 00 7F 02 B0 14 96-A1 11 00 B1 04 D3 E8 E8	=.....
42B0,10C0	32 00 FF 36 24 00 C4 1E-6D 01 E8 30 00 E8 5B 00	2..6\$...m..0..[.
42B0,10D0	2B F0 76 0D E8 1D 00 52-F7 26 0B 00 03 D8 5A EB	+..v....R.&....Z.
42B0,10E0	E9 5B 8A 2E 15 00 8A 16-FD 01 FF 2E 6D 01 BE 8B	..[.....m...
42B0,10F0	01 EB 54 90 01 06 24 00-11 2E 26 00 C3 A1 18 00	..T...\$...&....
42B0,1100	F6 26 1A 00 91 A1 24 00-8B 16 26 00 F7 F1 92 8B	..&...\$...&....
42B0,1110	0E 18 00 F6 F1 2A CC 91-FE C5 86 E9 D0 CE D0 CE	.....*.....
42B0,1120	0A F1 86 F2 87 CA 8A 16-FD 01 C3 BF 05 00 B8 01	.....
42B0,1130	02 CD 13 72 03 B0 01 C3-80 FC 11 74 F8 33 C0 CD	...r.....t.3..
42B0,1140	13 4F 75 EA BE D5 01 E8-1D 00 BE AB 01 E8 17 00	.Ou.....
42B0,1150	33 C0 CD 16 36 C7 06 72-04 34 12 EA 00 00 FF FF	3...6...r.4.....
42B0,1160	B4 0E BB 07 00 CD 10 AC-3C 2A 75 F4 C3 00 00 70	.....<\$u....p
42B0,1170	00 49 42 4D 42 49 4F 20-20 43 4F 4D 49 42 4D 44	.IBMBIO COMIBMD
42B0,1180	4F 53 20 20 43 4F 4D 00-00 00 00 0A 0D 4E 6F 6E	OS COM.....Non
42B0,1190	2D 53 79 73 74 65 6D 20-64 69 73 6B 20 6F 72 20	-System disk or
42B0,11A0	64 69 73 6B 20 65 72 72-6F 72 24 0A 0D 52 65 70	disk error\$..Rep
42B0,11B0	6C 61 63 65 20 61 6E 64-20 73 74 72 69 6B 65 20	lace and strike
42B0,11C0	61 6E 79 20 6B 65 79 20-77 68 65 6E 20 72 65 61	any key when rea
42B0,11D0	64 79 0A 0D 24 0A 0D 44-69 73 6B 20 62 6F 6F 74	dy..\$...Disk boot
42B0,11E0	20 66 61 69 6C 75 72 65-24 43 6F 70 72 2E 20 43	failure\$Copr. C
42B0,11F0	4F 4D 50 41 51 20 31 39-38 33 2D 38 37 00 55 AA	OMPAQ 1983-87. U.

通过对上述故障的分析和排除,可得到的启发是对于一般用户,可备一个工具软盘,用上述方法将不同版本,不同容量的硬盘的主引导扇区,引导扇区,软盘的引导扇区写到此软盘上,一旦发生引导扇区被破坏故障,只消用相同版本,相同容量的扇区内容替换就

是。对硬盘主引导扇区的分区表信息可用 FDISK 命令查得分区柱面号,再用公式计算出扇区号对分区信息进行修改。这样就可免去对磁盘重新格式化。

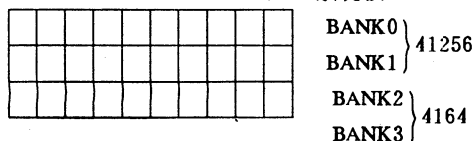
用上述方法我们对数台计算机和数十张软盘进行了消毒,都恢复了健康。

## IBM-PC/XT 微机存贮器故障排除一例

上海建筑材料工业学院 周凡 包锦雄

我们在维修 IBM-PC/XT 微机时遇到一例较特殊的故障,现将分析与排除过程介绍如下。该机的 RAM 配置为 640KB,由动态存贮器 41256 和 4164 各两组构成,且全部装在系统板上(见图1)。开机自检时 RAM 只有 608KB 正常,并显示错误信息“98000 201”,按“F1”键强制运行后,机器基本上能够工作。

IBM-PC/XT 系统板



608KB 的物理地址是 98000H,这说明第 3 组 RAM 的前 32KB 正常,后 32KB 出错。因为 4164 的十六根地址线的高、低八位是复用的,所以排除地址线损

坏的可能性;而数据线由于前 32KB 正常也可以排除。还有一种情况是芯片内部存贮单元损坏,尽管可能性较小,但由于系统板上 RAM 安装时均使用了插座,更换较为方便,所以首先换掉了第 3 组 RAM,开机后故障现象不变,从而排除了 RAM 损坏的可能性。

因机器基本可以使用,所以用软件工具来检查。利用 DEBUG 跟踪调试程序中的 E、D 命令,发现从 98000H 开始的一段地址上无论写什么,读出的信息总有许多是屏幕上显示的字符所对应的 ASCII 码。进一步编写了如下的测试程序:

```

0C10:0100 B80098      MOV     AX,9800
0C10:0103 8ED8         MOV     DS,AX
0C10:0105 8EC0         MOV     ES,AX
0C10:0107 FC           CLD
0C10:0108 29FF         SUB     DI,DI
0C10:010A B90040      MOV     CX,4000

```

```

0C10: 010D B8AAAA    MOV    AX,AAAA
0C10: 0110 BA55FF    MOV    DX,FF55
0C10: 0113 F3          REPZ
0C10: 0114 AA          STOSB
0C10: 0115 4F          DEC     DI
0C10: 0116 FD          STD
0C10: 0117 89FE        MOV    SI,DI
0C10: 0119 B90040      MOV    CX,4000
0C10: 011C AC          LODSB
0C10: 011D 38E0        CMP    AL,AH
0C10: 011F 751F        JNZ    0140
0C10: 0121 88D0        MOV    AL,DL
0C10: 0123 AA          STOSB
0C10: 0124 E2F6        LOOP   011C
0C10: 0126 20E4        AND    AH,AH
0C10: 0128 7416        JZ     0140
0C10: 012A 88C4        MOV    AH,AL
0C10: 012C 86D6        XCHG   DH,DL
0C10: 012E 20E4        AND    AH,AH
0C10: 0130 7504        JNZ    0136
0C10: 0132 88E2        MOV    DL,AH
0C10: 0134 EBD5        JMP    0115
0C10: 0136 FC          CLD
0C10: 0137 47          INC     DI
0C10: 0138 74DD        JZ     0117
0C10: 013A 4F          DEC     DI
0C10: 013B BA0100      MOV    DX,0001
0C10: 013E EBD5        JMP    0115
0C10: 0140 90          NOP

```

上述程序运行后,显示器上的字符全部消失。后来发现,这时机器仍在运行 DEBUG,按“Q”、“ENTER”键,可退出 DEBUG 返回系统,再执行 BASICA 解释程序(对显示区重写),显示恢复正常。这说明 RAM(980/0H

~9FFFFH)和显示缓存区(B8000H~BBFFFH)之间可能有不正常的联系。为进一步弄清两者之间的关系,将彩显及适配器换成单显及适配器,开机自检显示 RAM 只有 576KB 正常,错误信息为“90000 201”。此时将上面程序中的第一句改为“MOV AX, 9000”,运行后情况与前相同,从而证实了上述判断(单显适配器中,缓存区地址为 B0000H~B0FFFH)。为确定故障点,在 DEBUG 下编制并运行下面的程序:

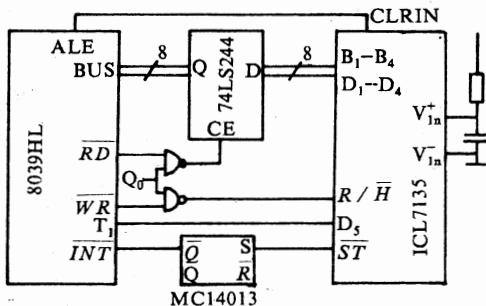
```

1000: 0100 B80098    MOV    AX,9800
1000: 0103 8ED8      MOV    DS,AX
1000: 0105 BE0000    MOV    SI,0000
1000: 0108 8804      MOV    [SI],AL
1000: 010A EBFC      JMP     0108
1000: 010C 90        NOP
1000: 010D 90        NOP
1000: 010E 90        NOP
1000: 010F 90        NOP
1000: 0110 90        NOP
1000: 0111 90        NOP

```

同时使用双踪示波器观察有关点的波形。将 XMEMW 送到 Y<sub>a</sub>,并选用 Y<sub>a</sub> 通道作为扫描触发,Y<sub>a</sub> 则用来测量 XMEMW 有效期间地址线和译码电路输出的电平。从地址总线高四位 A<sub>19</sub>~A<sub>16</sub>上测得“1001”(对应 9H),地址正确;测 CAS3 有效,译码也正确。再将上面程序中第一句改为:“MOV AX,A000”(A0000H~AFFFFH 为保留区),运行后测得 A<sub>19</sub>~A<sub>16</sub>为“1010”(对应 AH),地址正确,测 CAS3 仍有效,译码错误。从图 2 可见(在 46 页)

(上接 47 页)输出无锁存能力。8039 与 7135 的连接电路如图三。



图三 8039 与 7135 的接口电路

图三、8039 与 7135 的连接口电路

图三中,Q<sub>0</sub> 为来自地址锁存器的低位信号,7135 的位信号 D<sub>1</sub>~D<sub>4</sub> 以及数据信号 B<sub>1</sub>~B<sub>4</sub> 通过一数据驱动缓冲器接到 8039 的 BUS 口。7135 作为单片机的外部数据存储器,通过寄存器间址可访问。8039 对 7135 定写作可启动 7135,7135 完成一位转换后发出一个 ST 信号引起 8039 中断,在中断服务程序中,8039 确

定 T<sub>1</sub> 或 74LS244 的高 4 位(D<sub>1</sub>~D<sub>4</sub>)中某一位为高电平,表明相应位的转换数据已稳定,取出存入相应内存单元。中断五次后,一个数据就转换完成,转入相应数据整理和处理程序。中断服务程序流程如下:

