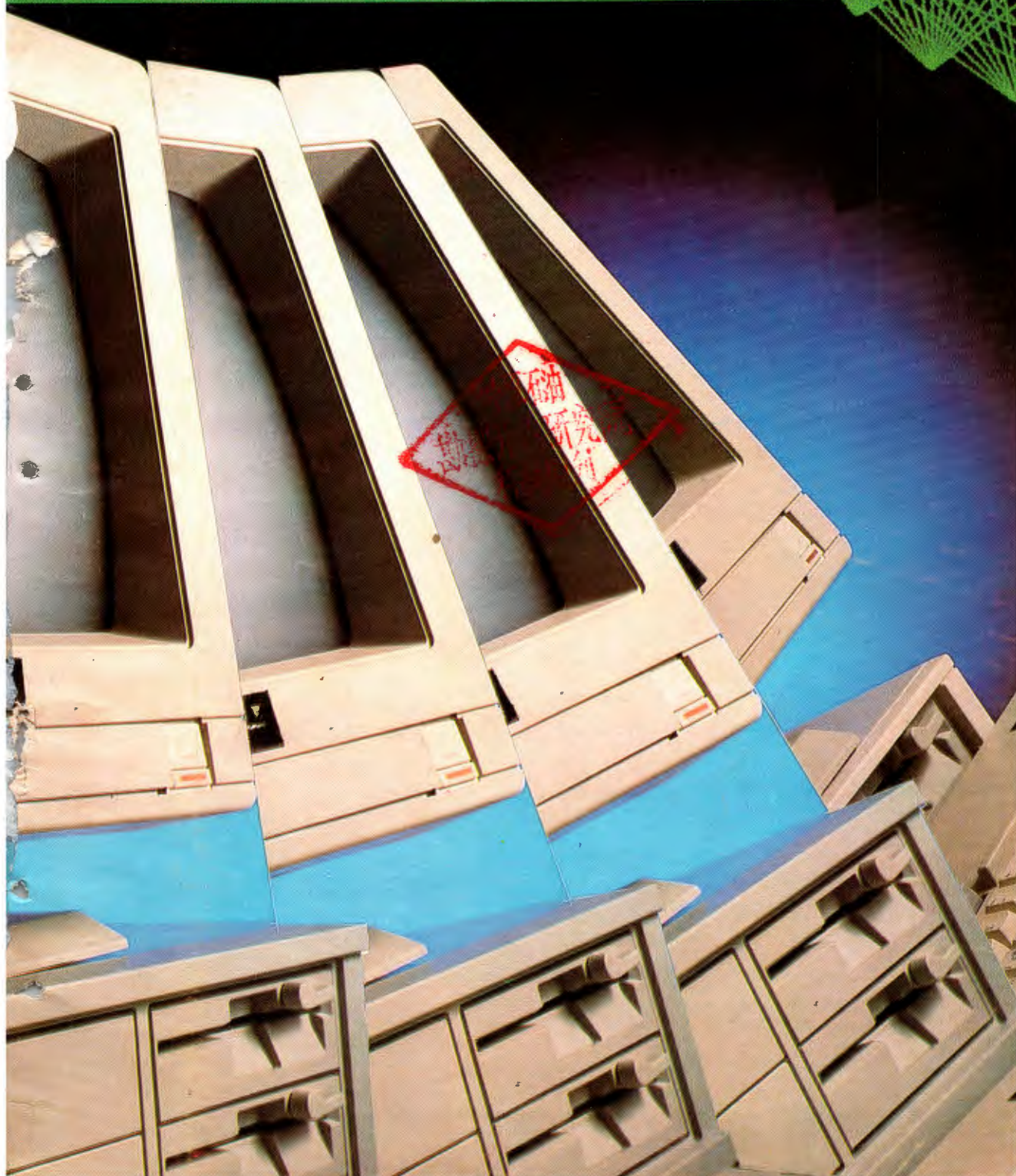


計算機應用研究

1991

4

APPLICATION RESEARCH OF COMPUTERS 《計算機應用研究》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：周賽渝

董 事：唐 珍	鄭國基	陸慰椿
秦小竹	龔宇清	黎琬常

《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4期（總第42期）

出版日期：1991年7月 本期責任編輯：鄧傢文

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

计算机应用研究第 8 卷第 4 期(总第 42 期)

目 录

软件篇

谈谈 SIMUL—TASK 386 仿真任务	符华儿(1)
数据库自动编程——程序变换法	龔正科(2)
dBASE Ⅲ 数据库结构自动编辑技巧	何金勇(7)
特殊稿件的稿纸打印	章建智(8)
电子 CAD 软件系统——TANGO	赵英俊 胡 阳(13)
用 TRUE BASIC 语言直接对 dBASE Ⅲ 数据文件进行操作	向南平(15)
介绍 IBM 局域网络支持程序	徐世坤 陈仲 姚学英(19)
IBM TOKEN—Ring 网络技术及功能	姚学英 徐世坤 陈仲(49)
LASER—PP40 汉化曲线绘制软件 1.00 版	周东方 肖杰 商胜伟 张跃元(53)
dBASE 和高级语言参数直接传递方法——桥地址思想	王敏生(23)
dBASE Ⅲ PLUS(FOXBASE)对桥地址的存取	王敏生(26)
dBASE Ⅲ 对桥地址的存取	王敏生(30)
dBASE(FOXBASE)与 BASIC 语言的直接参数传递	王敏生(33)
dBASE(FOXBASE)与 C 语言的直接参数传递	王敏生(36)
dBASE(FOXBASE)与 PASCAL 语言直接参数传递	王敏生(38)
dBASE(FOXBASE)与 FORTRAN 语言直接参数传递	王敏生(42)
dBASE(FOXBASE)与 COBOL 语言直接参数传递	王敏生(46)

硬件篇

打印机共享器的研制	余祥斌(54)
-----------------	---------

系统篇

急性心肌梗塞计量仪的设计与实现	卢正义(5)
单片机多点温度检测的实现	崔振远(57)
WHZ—851 自动比色分析仪研制与实验	周东方 侯志强 马庆海 王庆兰(60)

维修篇

GW0520CH 自检 RAM 到 3KB 死机检修一例	姚行中(58)
工业控制计算机中 RAM 数据保持和停电瞬间复位	戴晓黔(61)

信息篇

简讯	(58, 63, 64)
----------	--------------

APPLICATION RESEARCH OF COMPUTERS

VOL. 8 NO. 4 (Total 42)

CONTENTS

SOFTWARE

Discussing Simul—Task of 386	Fu Huaer(1)
Database Automatic Programming—Program Translating Method	Weng Zhenke(2)
Automatic Edit Skill for database structure in dBASE III	He Jinyong(7)
Printing Special Manuscript Paper	Zhang Jianzhi(8)
Software System for Electronic CAD;TANGO	Zhao Yingjun,Hu Yang(13)
Directly Operating to Data Files of dBASE III by Using TRUE BASIC Language	Xiang Nanping(15)
Recommending Support Program of IBM LAN	Xu Shikun,Chen Zhong,Yao Xueying(19)
Function and Technique of IBM Token—Ring Network	Yao Xueying,Xu Shikun,Chen Zhong(49)
The Software Version 1.00 Drawing Chinese Plot on Laser—PP40	Zhou Dongfang,Xiao Jie,Shang Shengwei,Zhang Yueyuan(53)
The Method of Direct Transmission between dBASE and High—Level Language Parameter—The Conception of Bridge Address	Wang Minsheng(23)
Accessing the Bridge Address by dBASE III PLUS(FOXBASE)	Wang Minsheng(26)
Accessing the Bridge Address by dBASE III	Wang Minsheng(30)
Directly Transmitting Parameter between BASIC and dBASE(FOXBASE)	Wang Minsheng(33)
Directly Transmitting Parameter between dBASE(FOXBASE)and c Language	Wang Minsheng(36)
Directly Transmitting Parameter between dBASE(FOXBASE)and PASCAL Language	Wang Minsheng(38)
Directly Transmitting Parameter between DBASE(FOXBASE)and FORTRAN Language	Wang Minsheng(42)
Directly Transmitting Parameter between dBASE(FOXBASE)and COBOL Language	Wang Minsheng(46)

HARDWARE

Research and Fabrication of Printer Sharer	Yu Xiangbin (54)
--	------------------

SYSTEM

Designing and Realization for Measurable Instrument of Acute Myocardial Infarct	Lu Zhengyi(5)
Realizing Temperature Detection for Multiple Point by Single chip Processor	Cui Zhenyuan(57)
Research and Experimentation of WHZ—851 Automatic Color Comparator	Zhou Dongfang,Hou Zhiqiang,Ma Qinghai,Wang Qinglan(60)

MAINTENANCE

How to REPAIR THE GW0520CH Computer while Testing the RAM of Itself and Halted Deadly to 3KB	Yao Xingzhong(58)
Keeping and Reseting the RAM Data while Instantaneous service Interruption in Industrial Control Computer	Dai Xiao qian(61)

INFORMATION

News	(58,63,64)
------------	------------

谈谈 SIMUL-TASK 386 仿真任务

广西计算中心 符华儿

Simul-Task386 是 UNIX 系统的一个实用软件,它提供了一个多任务、多用户的 MS-DOS 环境。它允许多个 MS-DOS 和/或者 UNIX 应用程序同时运行。本软件由两张软盘(1.44 兆/盘)的内容构成。它的主要功能有:

- 支持从 UNIX 系统命令行中引用 MS-DOS 命令,也支持从 MS-DOS 命令行中引用 UNIX 系统命令。

- 支持多任务和多用户操作,包括支持在远程 ASCII 终端上的基于文本方式的 DOS 应用程序的执行。

- 高度地汇集了 MS-DOS 和 UNIX 系统的共享文件。

- 支持一些 MS-DOS 应用程序的执行,比如: dBASE III、MFOXBASE+2.0 等。

正常运行 Simul-Task 的硬件要求是:

- 一台 AT&T6386 WGS 计算机或一台兼容的基于 Intel80386 的计算机。

- 最少要 2 兆内存,最好为 4 兆。

- 最少 40 兆硬盘,最好为 68 兆。

- 一个高密度 5.25 英寸或 3.5 英寸的磁盘驱动器。

- 一台 EGA、CGA 或单色显示适配器(MDA)做为主控台。

- 至少有 2 兆字节的磁盘空间供 Simul-Task386 文件使用。

- 支持串行终端。

它的软件要求是:

- UNIX SYSTEM V/386 Release 3.2 或更高版的操作系统。

- MS-DOS V 3.2 或更高版的操作系统。

Simul-Task386 作用在 UNIX 操作系统之内,它做为 UNIX 系统和 MS-DOS 之间的仲裁(arbiter)程序,激发 UNIX 操作系统解释、执行 MS-DOS 命令,并运行 MS-DOS 的应用软件。它实际上是把每一个 DOS 用户的任务,都当做 UNIX 下的一个进程来处理。因为 UNIX 系统是多任务和多用户的,因此 Simul

-Task386 环境下的 MS-DOS 也是多用户和多任务的,它允许用户共享资源,多个用户可在不同的终端上同时使用 DOS,一个用户可在不同的终端上同时执行多个任务。与此同时,其他终端用户还可运行 UNIX 软件。

对于纯 DOS 操作系统来说,它对设备、文件及其他资源都是独占的。独占性是 DOS 操作系统的特点。而共享和并发性是 UNIX 操作系统的特点。在 UNIX 系统中,每个用户都没有专用的软盘驱动器、打印机、硬盘和通讯口。Simul-Task386 允许用户在执行 MS-DOS 任务期间共享这些物理设备,这可通过选择 Simul-Task386 菜单上的相应菜单项来实现。比如:选择“Floppy Release”,便释放了当前 DOS 用户占用的软盘驱动器;选择“Printer Flush”,便可让 DOS 用户的打印进入 UNIX 的假脱机系统,排队等待打印;选择“Close serial”,便释放了一个固有的串行接口。在 Simul-Task386 中,首先进入 DOS 环境的用户占有软盘、硬盘、通讯串口等设备,必须选用上述有关菜单项释放相应的设备后,其他用户才能使用这些设备。其中,对硬盘 C 的释放是采用下面的方法进行。

在纯 DOS 中,文件的存取都与盘符打交道,比如: A、B 为软盘的盘符, C、D 为硬盘的盘符。而在 UNIX 中,所有外部设备包括软、硬盘都被当作文件(特殊文件)来处理。Simul-Task386 为这两者的联系建立了以下映射关系:

从 Simul-Task386 的 MS-DOS 角度看	从 UNIX 角度看
---------------------------------	------------

A: 盘	/dev/rdisk/fot
B: 盘	/dev/rdisk/flt
C: 盘	/usr/《用户名》/vpix/C;
D: 盘	/usr/dosapps/D;

其中, A、B、C、D 在 Simul-Task 386 中都是虚拟盘,它们对应的实际上分别是一个 UNIX 文件。它们是所对应的 UNIX 文件在 Simul-Task 386 中的映射。在 Simul-Task 386 中,用户可用有关的命令来建立虚拟盘(盘符可选用 A—Z)与 UNIX 文件的映射关系,一旦建立,用户在 DOS 中就可使用这些虚拟盘做

为盘符,随意使用。进入 UNIX 环境后,便可按指定的路径去访问那些与 DOS 盘符对应的文件,也可用 UNIX 的修改权限的命令去修改这些文件的属性。如属性改为大家都可读/写,那么,在 Simul-Task 386 中,该文件对应的虚拟盘符中的文件就是可共享的。比如:在 UNIX 中把文件/usr/(用户名)/vpix/C:的文件属性改为允许大家读写(用 Chmod 666《文件名》命令),这样,在 Simul-Task 386 的 MS-DOS 中,C 盘就是可共享的。DOS 的其他应用软件(普通文件)也可通过类似修改权限的方法来让大家共享。

纯 DOS 给用户提供的最大可用内存是 640K 字节。Simul-Task 386 也模拟地给每个 DOS 用户提供 640K 的可用内存,笔者曾在有 4 兆内存的多用户机上打开 8 个终端、一个主控台,其中 6 个终端进入 Simul-Task 386 环境,运行 DOS 程序,主控台和两台终端运

行 UNIX 程序,一切正常。如果 Simul-Task 386 为每个 DOS 用户都安排 640K 的实际内存,则 6 个 DOS 用户就占用了 3840K 字节内存,只剩下 256K 字节供 UNIX 的三个用户使用,平均每个 UNIX 用户只有 80 多 K,这显然是不太可能的,因此,笔者分析,Simul-Task 386 在为 DOS 用户分配内存时,使用了共享公用区、虚拟地址和复盖技术。

在 XENIX 操作系统下,也有类似 Simul-Task 386 功能的软件,它的名字是 VP/ix。正常运行 vp/ix 的硬、软件要求是:

- 386 机,运行 SCO XENIX/386 SYSTEM V 2.21、2.22 或更高版本。
- 至少 2 兆字节的内存。
- 至少 20 兆字节的硬盘空间。

数据库自动编程——程序变换法

新疆电子计算中心 龔正科

摘要 本文结合一个具体实例,较为全面地论述了利用程序变换法来实现数据库程序自动生成的若干技术细节,给出样板程序和生成器设计的方法。

一、程序生成原理 所谓程序生成是指通过一个给定的描述,经过一个程序变换成所希望的代码,从而省去人工大量的烦琐、不易、低效和重复劳动。目前,基于程序自动设计系统主要采用四种方法即过程法、演绎法、变换法和检查法,其中过程法和检查法仅仅是一种观点,真正赋予实践的情况很少,演绎法是基于数理逻辑的一种数学方法,应用于工程实践还有一段距离,变换法是最受欢迎的方法。

所谓程序变换法是指自动程序设计系统的输入是一种甚高级语言编写的程序,通过施用一系列变换,把这种输入转变成低级实现。实际上就是由一个程序转换成另一个功能等价的程序。

从程序变换法的定义可以了解到,一个程序生成系统应该由三部分组成,(1)一个模式(或称样板),(2)一组逻辑适用性条件,(3)一个动作过程。当找到一个模式实例时,就检查逻辑适用性条件,确定是否需要施用变换,如果需要,则求值动作以计算出新的一段代码,用以代替由模式匹配的代码,这种系统的典型结构

如图 1 所示。

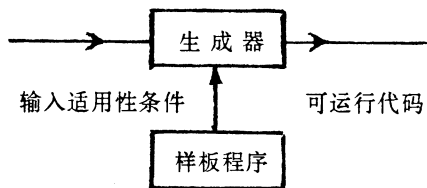


图 1 程序生成器的典型结构

下面我们根据该原理来论述数据库程序中样板程序和生成器的设计。

二、样板程序设计 样板程序是程序生成系统的基础,样板程序的质量决定了所生成的应用系统的质量,所以对样板程序必须千锤百炼。下面通过一个具体实例来说明样板程序的设计。

例:表 1 是一个学生情况表,要求设计一个编辑该文件的样板程序,其选择菜单要求具有两个选择项。

表 1 学生情况表(STUD. DBF)

学 号	姓 名	年 龄	性 别
SNO	SNAME	OLD	SEX
C(6,0)	C(8,0)	C(2,0)	C(2,0)
900101	欧阳奋进	20	男
900102	王望	21	男
900103	肖丽丽	20	女

★ PROGRAME NAME:TEST1. PRG

* 功能:编辑

* 生成参数 提示 1 和 2,条件 1 和 2 的字段名

* 参数行号:TS1=012,TS2=013,TJ1=020,TJ2=025

* 生成日期:

USE STUD. DBF

STORE. T. TO LOP

DO WHILE LOP

CLEAR

@10,10 SAY TS1

@12,10 SAY TS2

@14,10 SAY“0……返回”

STORE“0”TO KEY

@20,10 SAY“请输入数字[0,1,2,...]”GET
KEY PICTURE“!”

READ DO CASE

CASE KEY=“1”

STORE TJ1 TO TJJ1

@22,10 SAY“编辑条件”GET TJJ1

READ

EDIT FOR &TJJ1

CASE KEY=“2”

STORE TJ2 TO TJJ2

@22,10 SAY“编辑条件”GET TJJ2

EDIT FOR &TJJ2

CASE KTY=“0”

STORE .F. TO LOP

ENDCASE

ENDDO USE RETURN

程序 TEST1. PRG 就是根据要求所设计的样板程序,设计这种程序应该注意:①文件名将要在生成时被替换,所以它所在行和列不能随便改变,最好放在第二行;②程序生成日期在生成时根据当日系统日期填

写;③程序第 5 行是要进行变换的内容描述行,TS1,TS2 是菜单提示信息变量,TJ1,TJ2 是要进行编辑的条件变量。其中 USE STUD. DBF,如果把文件名改为变量,通过对话确定所打开的文件,这样就对任何文件都适合了。当然在实际中可以使该样板具有更强的功能。将若干个样板程序合在一起就形成所谓的样板程序库。

三、生成器的设计 根据用户输入的适用性条件,从样板库中提取相应的样板程序,启动生成器实现一系列变换,就能生成可执行代码。生成器实际上是一个字符串替代程序,它根据样板程序头部(第 5 行)所标识的变量内容,用户在运行变换器时,将这些变量赋值(输入),然后实现字符串替换,就生成代码。程序 TEST2. PRG 是针对例 1 的生成程序,它主要完成①将程序名改为用户的;②生成当日程序生成日期;③根据输入菜单信息替换掉样板程序中的 TS1 和 TS2 变量;④根据输入的条件信息(条件表达式初值)给条件表达式变量赋初值。它基本工作方式是利用一个数据库(字段名为 CC,类型为 C,长度=78),将样板程序拷入该库内,然后相应记录上根据输入信息处理替代,完后再将记录拷贝成. PRG 文件,就生成了可执行程序(SDEIT. PRG)。

★ PROGRAME NAME:TEST2. PRG

* 功能:生成学生文件的编辑程序

USE SC. DBF

DELETE ALL

PACK

APPEND FROM TEST1. PRG SDF

GO 2

REPLACE CC WITH SUBSTR(CC,1,16);+
“SEIT. PRG”

GO 6

REPLACE CC WITH SUBSTR(CC,1,13);+
DTC(DATE())

* 取参数和行号

READ

STORE“”+T3+“”TO T3

DO TEST3 WITH T3,TJ1

GO XH2

@22,10 SAY“输入提示 2”GET T2

READ

STORE“”+TRIM(T2)+“”TO T2


```

DO TEST3 WITH T2,TS2
GO XH4
@22,42 SAY "输入字段名"GET T4
READ
STORE""+T4+"'"TO T4
DO TEST3 WITH T4,TJ2
GO 4
DELETE NEXT 2
GO 5
STORE SUBSTR(CC,13,3)TO TS1
STORE SUBSTR(CC,21,3)TO TS2
STORE SUBSTR(CC,29,3)TO TJ1
STORE SUBSTR(CC,37,3)TO TJ2
STORE VAL(SUBSTR(CC,17,3)TO XH1
STORE VAL(SUBSTR(CC,25,3)TO XH2
STORE VAL(SUBSTR(CC,33,3))TO XH3
STORE VAL(SUBSTR(CC,41,3))TO XH4
* 区参数名和行号
GO XH1
STORE SPACE(20)TO T1,T2,T3,T4
@20,10 SAY"输入提示 1"GET T1
READ
STORE""+TRIM(T1)+"'"TO T1
DO TEST3 WITH T1,TS1
GO XH3
@20,42 SAY "输入字段名"GET T3
PACK
COPY TO SEDIT. PRG SDF
USE
RETURN

```

* PROGRAMME NAME:TEST3. PRG

* 替换内容

```

PARAMETERS TT,TTT
STORE 1 TO N
DO WHILE N<78
    IF SUBSTR(CC,N,3)=TTT
        REPLACE CC WITH SUBSTR(CC,1,N-
            1)+TT+SUBSTR(CC,N
            +3,78)
    ENDIF
    STORE N+1 TO N
ENDDO

```

RETURN

在 dBASE III PLUS 运行 TEST2. PRG 程序,屏幕
要求输入:

输入提示 1 1……根据年龄编辑 输入字段名
OLD

输入提示 2 2……根据性别编辑 输入字段名
SEX

完成之后,就生成 SEDIT. PRG 程序。

* PROGRAMME NAME:SEEDIT. PRG

* 功能:编辑

* 生成日期:01/27/90

USE STUD. DBF

STORE. T. TO LOP

DO WHILE LOP

CLEAR

@10,10 SAY'1……根据年龄编辑'

@12,10 SAY'2……根据性别编辑'

@14,10 SAY"0……返回"

STORE"0"TO KEY

@20,10 SAY"请输入数字[0,1,2……]"GET KEY

PICTURE"1"

READ

DO CASE

CASE KEY="1"

STORE'OLD'TO TJJ1

@22,10 SAY"编辑条件"GET TJJ1

READ

EDIT FOR &TJJ1

CASE KEY="2"

STORE'SEX 'TO TJJ2

@22,10 SAY"编辑条件"GET TJJ2

EDIT FOR &TJJ2

CASE KEY="0"

STORE. F. TO LOP

ENDCASE

ENDDO

USE

RETURN

参考文献:

1、徐家福:"软件自动化",《计算机研究与发展》,
1988,NO. 11

2、徐家福、戴敏、吕建:"算法自动化系统

NDADAS”,《计算机研究与发展》,1990,NO.2

设计的幻想与前景”,《计算机科学》,1989,NO.3

3, Charles Rich 和 Richard C. Waters: “自动程序

急性心肌梗塞计量仪的设计与实现

辽宁大学计算机系 卢正义

摘要 本文介绍急性心肌梗塞计量仪在急性心肌梗塞诊断中的应用,着重介绍该系统功能特点、系统设计与实现。

一、概述 快速测定急性心肌梗塞(AMI)范围的大小是心血管病研究的一个重要课题。目前确定AMI范围大小最有实际意义的是 Sobel 等人主张 CPK 酶法即肌酸磷酸酶法。此法一般在AMI患者发病后每二小时采血一次,直至CPK活性恢复正常为止。这样反复采血增加患者、医务人员负担,获得心肌坏死量也为时过晚,这对临床诊断很不利。为此我们研制AMI计量仪。根据快速推算AMI大小简捷方法,参阅文献[1],医务人员一般只通过三次采血就能快速测定CPK活性即:血清CPK活性序列、序列曲线、梗塞CPK克当量及克重量。为广大医务人员早期临床诊断、研究AMI病情提供具体梗塞定量指标。AMI计量仪经过中国医科大学附属一院、三院、沈阳军区总医院、沈阳市中心医院、沈阳市第四医院、中国人民解放军202医院、辽河油田第一职工医院等七家医院内科组织临床采血、计算验证:AMI克当量符合率为100%,克重量符合率为87.5%。AMI计量仪已通过省级技术鉴定。

二、AMI 计量仪系统功能 本系统具有以下功能:1. 操作简单、计算速度快

操作者只需连续输入七个原始数据即:三次测定血清CPK活性时间 t_1 、 t_2 、 t_3 ,相应的血清CPK活性值 $E(t_1)$ 、 $E(t_2)$ 、 $E(t_3)$ 以及AMI患者体重BW。仪器一般在15秒内快速计算出梗塞量,并且打印出血清CPK活性序列与曲线。

2. 自动记录、提示输入数据的次数

每输入一个原始数据后,显示提示符“,”输入七次后显示“E”。这样保证输入原始数据的次数正确。

3. 删改数据方便

可以删除、改正任意一个输错原始数据。

4. 可选择计算CPK或CPK-MB梗塞量

设有“C/B”选择键,显示“C”或“B”分别表示计算CPK或CPK-MB梗塞量。

5. 输出形式

LED数码管显示梗塞量。有两种打印格式,一种打印原始数据与梗塞量,另一种打印AMI血清CPK活性序列与序列曲线。

6. 梗塞量计算符合率

使用我国人平CPK消失率。AMI克当量符合率为100%,克重量符合率为87.5%。

三、系统构成 本系统采用微处理器Z80CPU,工作区2K RAM,8KEPROM固化系统软件。配制键盘、LED显示器以及TPμP-16微型打印机。其系统结构框图如图1所示。

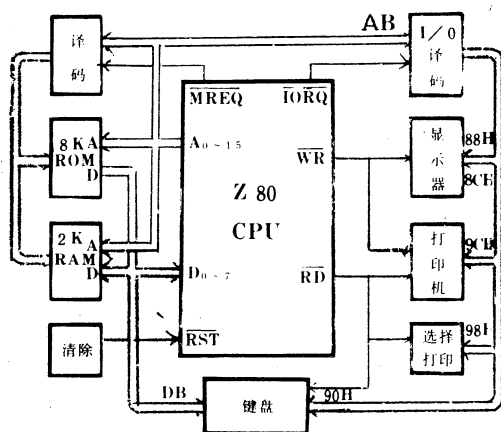


图1 系统结构框图

系统设计思想主要是通过键盘与显示相互配合,实现人机对话,方便操作。为此,键盘设有0~9、“.”、

“D”等12个数字键以及4个功能键即：I.S键、LIT键、C/B键、CR键。I.S键：显示克当量I.S内容。LIT键：打印AMI清单与CPK活性序列，序列曲线。C/B键：选择计算CPK或CPK-MB梗塞量。CR键：安排回车键处理程序，由它处理、管理以实现系统功能。

六位LED显示器如图2，线路简单规整，全部由集成片组成，只使用二片74LS273，二片MC1413和一

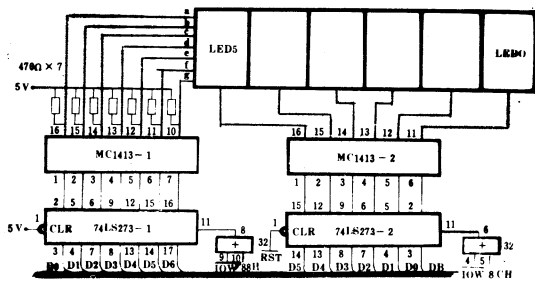


图2 LED显示器

片74LS32。通过执行一条I/O指令即：OUT(88H)，A，八D锁存器273-1片锁存显示数据的七段码以实现段控。同理，273-2片锁存位控电平，采取从高位到低位逐次扫描显示方法，实现位控过程。而MC1413是七路达林顿电流驱动器件，驱动LED数码管显示。

扫描显示软件只要保证段选、位控保持同步操作，显示数据保留1ms时间，其编程实现是容易的。实践证明：这样设计LED显示器调试方便，工作可靠，效果很好。

四、系统软件设计 系统软件包括：主程序、键扫描程序、显示程序、三个模块程序、四个功能键处理程序以及子程序库。由于采用模块化程序设计方法，因此程序结构紧凑、清晰、易于扩展。子程序库设置浮点数加、减、乘、除、平方、 e^x 、 $\ln x$ 、码制变换、曲线处理与打印等40个子程序及自定义字库。调用方便、灵活，系统功能丰富。系统主程序流程图如图3所示。

1. 程序运算误差分析

编制程序，考虑到运算层次繁杂和运算结果精度，系统采用四字节浮点数运算。浮点数基本格式是：1、字节阶码，其中附符1位、阶码7位。3字节尾数，其中符号2位，尾数22位。由于阶码表示范围： $-127 \sim +127$ ，尾数表示范围为： $0.5 \leq S < 1$ ，所以浮点数表示范围应是： $2^{-128} \leq 1 \times 1 < 2^{127}$ ，化为十进制数是：

$2.9 \times 10^{-39} \leq 1 \times 1 < 1.7 \times 10^{38}$ 。为提高精度，对运算梗塞值影响很大的函数 $\ln x$ 、 e^x ，采取契贝谢夫近似

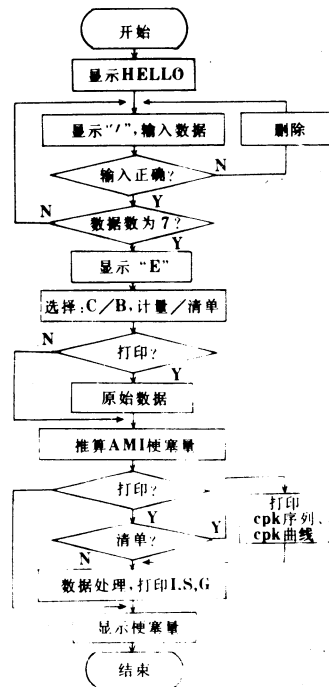


图3 系统主程序流程图

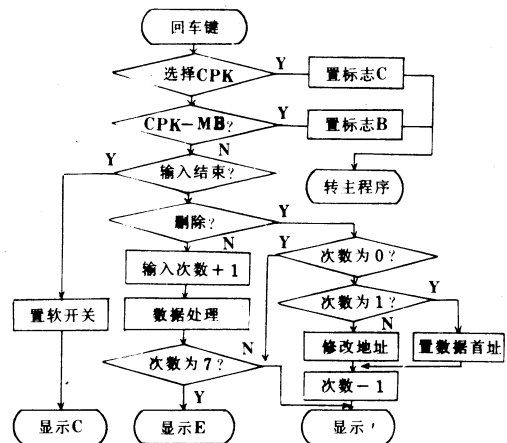


图4 回车键处理程序流程图

多项式算法编制子程序，从而保证系统运算精度。

2. 回车键处理程序

键盘设有一个回车键，其处理程序通过显示器实现人机对话联系，实现系统功能。其处理内容有：记录输入数据次数，不够七次显示提示符“'”，满七次显示“E”；根据各种情况置不同软件标志；删改输错数据、

修改地址;每次完成十进制数到BCD、ASCII、浮点数转换等,其流程图如图4所示。

3. 计算血清CPK活性总合子程序

设AMI患者每次采血时间为 t_0, t_1, \dots, t_n ,则相应测定血清CPK活性序列为 $E(t_0), E(t_1), \dots, E(t_n)$ 。

由此可推算出T小时内血清CPK活性总合即:

$$\int_0^T E(t)dt = \sum_{i=1}^{n-1} E_i \cdot \Delta t_i, (\text{采用梯形法})$$

$$\text{其中: } E_i = \frac{1}{2} [E(t_i) + E(t_{i+1})].$$

由于: $\Delta t_i = t_{i+1} - t_i = 1$,

$$\text{故: } \int_0^T E(t)dt = \sum_{i=1}^{n-1} E_i. \text{ 其流程图略。}$$

五、结束语 AMI计量仪有较好性能价格比,操作简单、携带方便、便于临床使用。在省级鉴定会上,与

会专家一致认为:“心肌梗塞计量仪的研制,在心血管病诊治中是一项具有国内领先水平的科研成果,为国内首创”。现在我们正进行普及推广工作,相信AMI计量仪在AMI诊治、研究中会得到更广泛的应用。

参考文献

[1]张恒久等:“推算急性心肌梗塞大小的一种数学方法”,《中华心血管病》杂志,1987,15(2),封三。

[2]Ahumada, G, et al: Progress in Cardiovascular diseases 18(5):405, 1976。

[3]王月霞编:《微型计算机汇编语言程序设计》,电子工业出版社,1985年。

[4]周明德编:《微型计算机硬件软件及其应用》,清华大学出版社,1983年。

dBASE III 数据库结构自动编辑技巧

河北省医院计算机室 何金勇

摘要 本文设计了一个对dBASE III数据库结构自动修改的实用子程序,应用本程序可在程序运行方式下象修改记录那样方便地修改数据库结构,并生成修改后的新数据库结构。

一、引言 用dBASE III语言编写的数据库管理系统,一般通用性较差,其主要原因在于不同单位的用户由于长期以来形成的习惯,或因实际情况的不同,使得对同一事物的管理略有差别,但程序员在编写管理程序时是不可能考虑到不同用户的所有情况的,即使是同一用户,由于管理的内容和形式在不断变化,同一管理程序也必然要随之变动,而修改程序时,最使程序员感到为难的是数据库结构的修改,对已经设计好的一个数据库管理系统来说,如果其中的某些数据库结构发生变化,就可能引起整个系统程序的变动,这无疑会花费大量时间和精力,而且还可能影响实际工作的完成。

一般介绍dBASE III的参考文献中,对数据库结构的修改只提供了MODISTRU命令,用此命令在dBASE III点状态下可以直接修改数据库结构,但在程序运行时使用此命令修改数据库结构时,程序是不能自动“记忆”哪些字段曾被修改,因此也就无法判别。

本文设计了一个在程序执行方式下自动修改库结构并生成新的数据库结构的子程序,使得用户可象修

改记录那样修改库结构中的字段。

二、程序及说明 程序名为H. PRG

PARAM KU1, KU2

CLEAR

SET TALK OFF

USE &KU1

COPY TO MIDD STRU EXTE

USE MIDD

GO BOTT

RE=RECNC()

IF RE<8

RE1=1

ELSE

RE11=RE-INT(RE/8)*8

IF RE11>0

RE1=INT(RE/8)+1

ELSE

RE1=RE/8


```

        ENDI
XG= "X"
@3,20 SAY "A-添加字段 X-修改字段"GET
XG
READ
USE MIDD
IF XG= 'A'
    APPE BLAN
ENDI
@0,0 SAY SPAC(79)
@0,0 say '字段名'+SPAC(3)+'类型'+SPAC
(3)+'宽度'+SPAC(2)+'小数'
DO WHIL RE1>0. AND.. NOT. EOF( )
    @1,0 CLEA
    I=1
    DO WHIL I<=8. AND.. NOT. EOF( )
        @I,0 GET FIELD-NAME
        @I,13 GET FIELD-TYPE
        @I,18 GET FIELD-LEN
        @I,24 GET FIELD-DEC
    READ
    IF RECN ( )<RE
        SKIP
    ELSE
        EXIT
    ENDI
    I=I+1
ENDD
RE1=RE1-1
ENDD
USE
CREA &KU2 FROM MIDD
ERAS MIDD. DBF
RETU
说明:

```

1. 调用本程序只须使用“DO H WITH NAME1, NAME2”命令,其中 NAME1 是库结构需要修改的数据库名,NAME2 是修改后的数据库名。

2. 程序设计为每屏可修改 8 个字段,变量 RE1 是总屏数,变量 KU1 接收 NAME1 的值,变量 KU2 接收 NAME2 的值。

3. “COPY TO MIDD STRU EXTE”命令语句生成的 MIDD 数据库是一个中间库,其结构如下:

字段	字段名	类型	宽度	小数
1	FIELD-NAME	字符型	10	
2	FIELD-TYPE	字符型	1	
3	FIELD-LEN	数字型	3	
4	FIELD-DEC	数字型	3	

MIDD 库中的记录是 NAME1 中的字段,记录个数是 NAME1 中字段个数。

COPY TO MIDD STRU EXTE 语句是一个扩展拷贝命令,使用它生成的数据库结构是不变的,总是和 MIDD. DBF 库结构一样。

4. CREA &KU2 FROM MIDD 是 CREA 命令的扩充形式,一般的 dBASE III 教科书中对此形式都未介绍,其格式如下:

CREA<新库文件名> FROM<中间库文件名>

功能:以中间库中的记录为新库的字段产生一个新的数据库文件。

中间库文件必须是象 MIDD 库的结构形式。

三、结束语 对本程序适当修改,即可使程序“记住”MIDD 库中的字段哪些被修改过,如生成 MIDD 库后再作一个备份 MIDD1,把 MIDD 修改完后,将其与 MIDD1 中的记录逐条比较,就能发现 NAME1 中的哪些字段被修改过。一般地说,同一类事务的管理程序的数据库结构差别不会很大,因此,程序员在设计管理程序时,只须规定库结构中的某几个字段可被用户修改,并使程序具有能够处理修改后的数据库结构的能力,这样,管理程序的通用性必将大大提高。

特殊稿件的稿纸打印

安徽芜湖供电局 章建智

用计算机进行文字和图表的输入、输出是很常见的。随着软件开发技术的发展,越来越受欢迎的编辑软

件的出现,使得文字和图表的编辑越来越简单。所以人们很自然地想到稿纸打印。

所谓稿纸打印,就是将编辑好的文稿以稿纸的形式打印输出。目前,常用的稿纸打印方法有两种:一是利用设备自身的功能,如 AR32 40 汉字打印机等;二是利用文稿打印程序,如 WP S 编辑软件等。显然,后者具有通用性。但仍有不足之处。如无法进行特殊打印,如上下标、上下划线等打印功能;无法向文件输出、不便于图表的剪贴等。

笔者经过长期的实践经验,用 BASIC 编制了一个较为实用的稿纸打印程序(附程序清单)。该程序可克服上述缺点。现介绍给读者,供参考。

一、程序功能 本程序在 2.13F 操作系统的支持下运行,具有屏幕输出、文件输出和打印输出功能。如原稿件中有图表,且希望将图表剪贴在稿纸上,这时可选择文件输出,然后用 CC ED 等编辑软件进行图表剪贴。程序以 20*20 标准公文纸打印输出,具有页码计数和换页功能。可使用 2.13F 中的大多数特殊打印控制码,可在稿纸的适当位置留有空白行,用于粘贴图表。

二、功能的实现 根据人们书写稿件的习惯,文章的标题,一般从稿纸的第五格写起;每自然段的首行行首,总要空两格;如果稿件中有图形或表格,则要在稿纸上留一定的空白行,供粘贴图表之用。本程序正是利用这一习惯编制的。

程序怎样识别“标题行”、“段落首行”和“空白行”呢？为了使程序能识别这些“行”，在编辑稿件时要求“标题行”的行首至少要空 8 个空格；“段落首行”的行首至少空 4 个空格，但不能多于 7 个空格。为了简单起见，“空白行”按“标题行”处理，但内容要求用一对双引号括起一个或若干个空格。以上三种“行”的格式如下：

标题行:

□□□□□□□□标题内容

段落首行

□□□□:段落首行内容

空白行:

[illegible]

□□□□□□□□”□□□□□□标题内容

若干个空格

注:以上“口”代表一个空格,下同。

程序规定“标题行”从第五格开始打印,标题的内容如果字数较多,则打印到第16格后,自动转到下一

行的第五格打印,同样也只能打印到该行的第16格,如果标题还没有打完则继续换行,一直到标题打印完毕。有时人们并不希望这样打印,而希望标题打印在一行的中间。这时可在标题内容的前面加上若干个空格,以使标题居中打印。但必须用双引号括起来。格式如下:

□□□□□□□□"□□□□□□标题内容

↑
若干个空格

同样,如果“段落首行”,不想从第二格开始打印,也可用上述方法。另外还可在“标题行”的行末或“段落首行”的行末加上若干个空格,所加的空格同样要用双引号括起来,以使如落款、署名、日期等内容打印在适当的位置。如“芜湖供电局”这五个字要从第八格开始打印,可用两种方法:

①当做“标题行”处理,格式如下:

□□□□□□□□”□□…□□芜湖供电局

↑加六个空格

②当做“段落首行”处理,格式如下:

□□□□“□□…□□芙湖供电局□□…□□”

↑
加十个空格

↑ ↑
加十六个空格

三、控制码的使用 本程序中可使用的特殊打印控制码,见下表:

格 式	功 能
'@字型'	置字型 (在稿纸打印中可选择字型有 A,E,I,M 四种)
'%0~7'	置打印背景 其中:0 无背景,1 网点,2 网格, 3 横线,4 竖线,5 右斜线, 6 左斜线,7 交叉线
';'	置抽点打印(轻打)
'\"'	置反白打印
'/'	置上划线打印
'_'	置下划线打印
'='	置正常打印(开机状态,清以上四项)

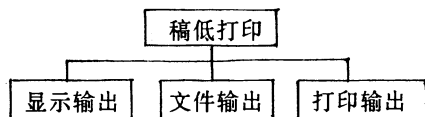
格 式	功 能
‘(’	置左旋 90 度打印
‘)’	置右旋 90 度打印
‘?’	置结束旋转(开机状态,清以上二项)
‘+’	置上标字符
‘-’	置下标字符
‘!’	置正常字符(开机状态,清以上二项)

控制码的使用方法同 2.13F,但要注意以下两点:①所有的控制码只能插在一行的中间或放在行首,不可放在行末;②两个或多个控制码不能放在一起使用。

《特殊稿件的稿纸打印》程序具有特殊打印功能:

打印的字体有:‘@A’宋体、‘@M’楷体、‘@I’黑体、‘@E’仿宋体;‘@A’背景有:1‘%1’网点‘%0’、2‘%2’网格‘%0’、3‘%3’横线‘%0’、4‘%4’竖线‘%0’、5‘%5’右斜线‘%0’、6‘%6’左斜线‘%0’、7‘%7’交叉线‘%0’、0‘%0’无背景;另外还有:‘,’抽点‘=’(轻打)打印,‘\’反白‘=’打印,‘/’上划线‘=’打印,‘-’下划线‘=’打印,‘(’左旋‘?’90度打印,‘)’右旋‘?’90度打印,上标‘+’ABCD‘!’打印,下标‘-’1234‘!’打印等。

程序框图如下:



程序清单:

```

10 REM 程序名:WGDY.BAS
20 REM 该程序可打印 2.13F 的控制码。
30 REM 芜湖供电局计算机室章建智编制。
40 CLS
50 KEY OFF
60 PRINT CHR$(14)+"[@M-200/150*5
  芜湖供电局计算机室]"
70 PRINT CHR$(14)+"[@A-220/220*12
  一九九零年七月]"
80 PRINT CHR$(14)+["@M-274/180*4 章
  建智]"
90 PRINT CHR$(14)+"b14,0,0,600,300]"
100 PRINT CHR$(14)+"b14,20,20,560,
  260]"
  
```

```

110 PRINT CHR$(14)+"[@m-100/50*2★
  文稿打印程序★]"
  
```

```
120 A$=INKEY$;IF A$=""THEN 120
```

```
130 DIM AA$(30,20),ZBF$(7)
```

```
140 REM *****
```

```
150 REM 主功能选择
```

```
160 REM *****
```

```
170 CLS:LOCATE 5,10:PRINT"本程序具有如
  下功能:"
```

```
180 LOCATE 7,10:PRINT"1. 显示输出 2. 文件
  输出 3. 打印输出"
```

```
190 LOCATE 9,10:PRINT"请选择功能号(0 退
  出)";
```

```
200 CH$=INKEY$;IF CH$=""THEN 200
```

```
210 IF CH$<>"1"AND CH$<>"2"AND CH$<>
  "3"AND CH$<>"0"THEN LOCATE 10,
  10:PRINT"功能号选择错误,重选。";FOR
  DD=1 TO 1000:NEXT DD:LOCATE 10,
  10:PRINT SPACE$(20);GOTO 190
```

```
220 IF CH$="0"THEN END
```

```
230 IF CH$="1"THEN DEV$="SCRN.";T
  =0;FOR I=1 TO 7:ZBF$(I)="" :NEXT
  I:GOTO 450
```

```
240 ZBF$(1)="┌";ZBF$(2)="┐";
```

```
ZBF$(3)="└
```

```
";ZBF$(4)="┘";ZBF$(5)="┌";ZBF
  $(6)="┐"
```

```
;ZBF$(7)="└";T=17
```

```
250 IF CH$="3"THEN DEV$="LPT1.";
  WIDTH DEV$,255
```

```
260 IF CH$="2"THEN CLS:INPUT"请输入
  要输出的文件名";DEV$
```

```
270 REM *****
```

```
280 REM 打印机选择
```

```
290 REM *****
```

```
300 CLS:LOCATE 5,10:PRINT"1. M-17 24,
  M-2024 等针径为 1/160 英寸的打印机"
```

```
310 LOCATE 7,10:PRINT"2. AR3240,LQ1500
  等针径为 1/180 英寸的打印机"
```

```
320 LOCATE 9,10:PRINT"请选择打印机
  ....";
```

```
330 D$=INKEY$;IF D$=""THEN 330
```

```
340 IF D$<>"1"AND D$<>"2"THEN LO-
  CATE 10,10:PRINT"打印机选错,重选。";
```



```

FOR DD=1 TO 1000;NEXT DD;LOCATE
10,10;PRINT SPACE¥(20);:GOTO 320
350 IF D¥="1"THEN HJ¥="18";ZH=73;
    TH=10 ELSE HJ¥="16";ZH=82;TH=3
360 REM * * * * *
370 REM 邮政编码
380 REM * * * * *
390 LOCATE 11,10;PRINT "是否要邮政编码
    Y/N(缺省为不要)? ";
400 YMY¥=INKEY¥;IFYMY¥=" "THEN
    400
410 IF YMY¥="Y"OR YMY¥="Y"THEN
    LOCATE 12,10;PRINT "请输入邮政编码
    (缺省为 241000)";:INPUT YM¥;IF YM
    ¥=" "THEN YM¥="241000"
420 REM * * * * *
430 REM 文稿名和打印份数的确定
440 REM * * * * *
450 OPEN DEV¥FOR OUTPUT AS #2
460 IF CH¥("<"1"THEN PRINT #2,"&" +HJ
    ¥+@A>,"
470 CLS :INPUT "文稿名":F¥:CLS :LO-
    CATE 5,10;PRINT "程序正在进行,请稍等
    ...";
480 IF CH¥="3"THEN INPUT "请输入打印份
    数";COUNT ELSE COUNT =1
490 REM * * * * *
500 REM 初始化
510 REM * * * * *
520 FOR I=1 TO 30;FOR J=1 TO 20;AA¥(I,
    J) = " ";NEXT J;NEXT I
530 I=1 :J=1;C1=0;PIE =0;TEMP¥="";
    TEMP1¥="";COUNT1=0
540 REM * * * * *
550 REM 主程序开始
555 REM * * * * *
560 OPEN F¥ FOR INPUT AS #1
570 WHILE NOT EOF (1)
580 IF I>=21 THEN 720
590 LINE INPUT #1,A¥
600 IF LEFT¥(A¥,1)=CHR¥(138) THEN
    A¥=MID¥(A¥,2)
610 IF A¥=CHR¥(13)+CHR¥(10) OR A¥
    =" " OR A¥=CHR¥(13) OR A¥=CHR

```

```

¥(10) THEN 820
620 IF LEFT ¥(A ¥, 8) = STRING ¥(8, 32)
    THEN 650
630 IF LEFT ¥(A ¥, 4) = STRING ¥(4, 32)
    THEN 700
640 Z=1;H=21;GOSUB 990;GOTO 820
650 IF J(>)1 THEN I=I+1
660 J=5;Z=5;H=17;GOSUB 990
670 IF J(>)5 THEN I=I+1
680 J=1
690 GOTO 820
700 IF J(>)1 THEN I=I+1
710 Z=1;J=3;H=21;GOSUB 990;GOTO 820
720 GOSUB 1260;FOR M=1 TO 20;FOR N=1
    TO 20;AA ¥(M,N)=" ";NEXT N;EEXT
    M
730 IF I=21 AND J=1 THEN I=1;GOTO 820
740 IF I=21 AND J(<>)1 THEN GOSUB 800;I
    =1;GOTO 820
750 IF J=1 THEN GOSUB 770 :GOTO 820
760 GOSUB 800;GOSUB 770;GOTO 820
770 FOR M=21 TO I-1;FOR N=1 TO 20
780 AA ¥(M-20,N)=AA ¥(M,N);AA
    ¥(M,N)=" ";NEXT N;NEXT M
790 I=I-20 :RETURN
800 FOR N=1 TO J-1
810 AA ¥(I-20,N)=AA ¥(I,N);AA ¥(I,N)
    =" ";NEXT N;RETURN
20 WEND
830 IF I>=21 THEN 850
840 GOTO 900
850 GOSUB 1260;FOR M=1 TO 20;FOR N=1
    TO 20;AA ¥(M,N)=" ";NEXT N;NEXT
    M
860 IF I=21 AND J=1 THEN I=1;GOTO 900
870 IF I=21 AND J(>)1 THEN GOSUB 800;I=
    1;GOTO 900
880 IF J=1 THEN GOSUB 770;GOTO 900
890 GOSUB 800;GOSUB 770
900 IF I=1 AND J=1 THEN 930
910 GOSUB 1260
920 IF CH¥("<"1"THEN PRINT # 2,"&20
    @A'";
930 CLOSE;INPUT"还有文稿要打印吗 Y/N",

```



```

YY
940 IF YY="y" OR YY="Y" THEN 170
950 END
960 REM *****
970 REM 赋值子程序
980 REM *****
990 GOSUB 1500
1000 IF LEFT$(A$,1)=CHR$(34) THEN A$=MID$(A$,2)
1010 IF RIGHT$(A$,1)=CHR$(34) THEN A$=LEFT$(A$,LEN(A$)-1)
1020 A1$=LEFT$(A$,1):A1=ASC(A1$):A2$=MID$(A$,2,1):IF A2$(">") THEN A2=ASC(A2$) ELSE A2=A1:A2$="":A$=A1$+A2$
1030 IF A1$=" " THEN PIE=PIE+1
1040 IF A1$=" " AND PIE=1 THEN TEMP$="":TEMP1$=""
1050 IF PIE(">")0 THEN 1070
1060 GOTO 1090
1070 TEMP$=TEMP$+A1$:A$=MID$(A$,2)
1080 IF PIE(">")2 THEN GOTO 1210 ELSE PIE=0:GOTO 1210
1090 IF INSTR(TEMP$,";")(">")0 OR INSTR(TEMP$,"\\")(">")0 OR INSTR(TEMP$,"\\")(">")0 OR INSTR(TEMP$,"-")(">")0 THEN TDMP1$="="
1100 IF INSTR(TEMP$,"(")(">")0 OR INSTR(TEMP$,")")(">")0 THEN TEMP1$="?"
1110 IF INSTR(TEMP$,"+")(">")0 OR INSTR(TEMP$,"-")(">")0 OR INSTR(TEMP$,":")(">")0 THEN TEMP1$="'"
1120 IF INSTR(TEMP$,"@")(">")0 THEN TEMP1$=""
1130 IF INSTR(TEMP$,"%")(">")0 THEN TEMP1$="%"
1140 IF INSTR(TEMP$,"%0")(">")0 OR INSTR(TEMP$,"=")(">")0 OR INSTR(TEMP$,"?")(">")0 OR INSTR(TEMP$,"!")(">")0 THEN TEMP1$="":TEMP$=""
1150 IF CH$="1" THEN TEMP$="":TEMP1$=""
1160 IF (A1<128 AND A2<128)OR (A1>160 AND A2>160) THEN 1180
1170 AA$(I,J)=TEMP$+LEFT$(A$,1)+" "+TEMP1$:A$=MID$(A$,2):GOTO 1190
1180 IF A2$(">")" " THEN AA$(I,J)=TEMP$+LEFT$(A$,2)+TEMP1$:A$=MID$(A$,3) ELSE AA$(I,J)=TEMP$+LEFT$(A$,1)+" "+TEMP1$:A$=MID$(A$,2)
1190 J=J+1:IF J=H THEN J=Z:I=I+1
1200 IF INSTR(TEMP$,"@")(">")0 THEN TEMP$=""
1210 IF A$=" " THEN GOTO 1220 ELSE GOTO 1020
1220 RETURN
1230 REM *****
1240 REM 打印子程序
1250 REM *****
1260 C1=C1+1
1270 IF CH$="1" THEN CLS:GOTO 1290
1280 LOCATE 5,20:PRINT "程序正在进行工作"
1290 FOR C=1 TO COUNT
1300 LOUNT1=COUNT1+1
1310 LOCATE 6,1
1320 PRINT #2,TAB(T);ZBF$(1);"—";:FOR K=1 TO 19:PRINT #2"—";:NEXT K:PRINT #2,ZBF$(7)
1330 FOR L=1 TO 19:GOSUB 1430
1340 IF CH$="1" THEN GOSUB 1460 ELSE GOSUB 1440
1350 NEXT L
1360 GOSUB 1430:PRINT #2,TAB(T);ZBF$(4);"—";:FOR K=1 TO 19:PRINT #2,"_ _";:NEXT K:PRINT #2,ZBF$(5)
1370 IF (YMY$="y" OR YMY$="Y") AND C1=1 THEN PRINT #2,TAB(3+T);"* 邮政编码";YMY$;
1380 PRINT #2,TAB(36+T);"第";C1;"页";TAB(60+T);"20×20"
1390 IF CH$="1" THEN 1420
1400 FOR K=1 TO ZH-61:PRINT #2,"";

```



```

NEXT K:LOCATE 6,1:PRINT STRING
Y(80,32)
1410 IF COUNT1 MOD TH=0 THEN PRINT
#2,""
1420 NEXT C:RETURN
1430 PRINT #2,TAB(T);ZBFY(2);AAY(L,
1);:FOR K=2 TO 20:PRINT #2,"|":;
PRINT #2,AAY(L,K);:NEXT K:
PRINT #2,ZBFY(2):RETURN
1440 PRINT #2,TAB(T);ZBFY(3);"—":;
FOR K=2 TO 20:PRINT #2"┐—":;
NEXT K:PRINT #2,ZBFY(6)
1450 PRINT #2,TAB(T);ZBFY(3);"—":;
FOR K=2 TO 20:PRINT #2,"└—":;
NEXT K:PRINT #2,ZBFY(6):RE-
TURN
1460 PRINT #2,TAB(T);ZBFY(3);"—":;
FOR K=2 TO 20:PRINT #2,"+-":;
NEXT K:PRINT #2,ZBFY(6):RETURN
1470 REM*****
1480 REM 去掉行首和行末空格子程序
1490 REM*****
1500 S1=1
1510 IF MIDY(AY,S1,1)=" "THEN S1 S1+1
=1:GOTO 1510
1520 AY=MIDY(AY,S1):S1=1
1530 IF AY=" "THEN 1560
1540 IF ASC(RIGHTY(AY,S1))=32 THEN
S1=S1+1:GOTO 1540
1550 AY=MIDY(AY,1,LEN(AY)-S1+1)
1560 RETURN

```

电子 CAD 软件系统——TANGO

华中理工大学 赵英俊 胡阳

随着微型计算机应用技术的普及和发展,人们越来越多地利用微机来进行电子线路设计(包括原理图的设计与模拟、印刷电路板的设计与加工等等)。就目前的情况来看,大多采用现成的电子 CAD/CAE 软件来进行这方面的工作。国内市场上也出现了许多电子 CAD/CAE 系统,如 SMARTWORK、Auto ROUTE/BOARD、REDBAORD-REDLOG、EE Desinger/System、PCAD 以及 or CAD 等等。这些软件价格不同,功能各异,使用起来也各具特色。这里介绍一个电子 CAD/CAE 系统——TANGO。

一、TANGO 系统的结构与特点 TANGO 软件是美国 ACCEL Technologies Inc. 在 1987 年推出来的新一代电子 CAD 软件系统,目前国内流行的有 V1.13~V3.12 等若干版本。该系统由原理图编辑软件(TANGO-Schematic)、印刷电路板设计软件(TANGO-PCB)和印刷电路板自动布线软件(TANGO-ROUTE)三部分组成,用它可以完成从原理图设计到印刷电路板工艺文件输出一系列复杂而繁琐的工作。

总的来讲,TANGO 软件具有“学习容易、操作简便、经济实用”的特点,的确不失为同类系统中的上乘

之作。

1. 对硬件环境要求低 TANGO 系统可以在 DOS 2.00 以上版本的操作系统的支持下,运行于任何 IBMPC/XT/AT/PS2 微机或兼容机之上,仅要求微机系统具有 384K 以上的内存,配有两个软盘驱动器或一个硬盘驱动器。TANGO 系统的图形可以通过 CGA、EGA、VGA、AGA 及 HEAR 图形卡在相应的显示器上显示出来。此外,还可以用 HP、DMP 及 ROLAND 系列的绘图仪和 EPSON 系列的打印机输出各种图形。鼠标器是一种辅助硬件,而不是必备硬件。

2. 学习容易 TANGO 系统采用菜单式命令结构,其操作命令不仅简单易学,而且可以在屏幕上随时查询。这样,即使不依靠操作手册,用户也能通过实际操作,很快熟悉全部操作命令。

3. 操作简便 使用 TANGO 系统进行设计时,全部操作均能用键盘很方便地完成,如果使用鼠标器则能提高工作效率。TANGO 系统配有极为丰富的元件符号图形库和元件封装图形库,这些图形库可以很方便地进行扩充和改建,使之满足各用户的需要。

4. 实用性强 由于 TANGO 系统独特的设计, 尤其是用它进行印刷电路板设计时自动布线功能的高布通率, 使之成为真正实用的电子 CAD 工具。

下面仅以原理图编辑软件(TANGO-Schem-atic)为例, 来看看 TANGO 系统的软件功能:

* 可以动态选择 A、B、C、D、E 五种图纸尺寸(最大为 42×32 英寸)。

* 不仅具有完备的元件符号图形库(按元件的生产公司或用途归为 14 类, 总数达 3000 多种), 而且可以由用户很方便地加以修改和扩充。

* 在一个总图名下, 可以生成和管理多达 99 张分图, 所有的图纸均能用打印机或绘图仪输出。

* 绘图时最小的移动步距为 0.1 英寸。

* 具有丰富的块操作功能, 可以对图形块进行标记、拷贝、移动和删除。

* 有四种线形一细线、粗线、虚线和总线。

* 可以生成元件连线表供后面设计印刷电路板时自动布线用, 并且还能把此连线表转换成其它电子 CAD 系统所能接受的格式。成套的 TANGO 系统大多装在七张 360KB 的 5 英寸磁盘上, 这些磁盘分别是:

- (1) TANGO-SCHEMATIC-EDIT
- (2) TANGO-SCHEMATIC-LIB1
- (3) TANGO-SCHEMATIC-LIB2
- (4) TANGO-SCHEMATIC-UTILITY
- (5) TANGO-PCB-EDIT
- (6) TANGO-PCB-LIB/UTILITY
- (7) TANGO-ROUTE

其中, (1)~(4)用于原理图的编辑, (5)和(6)用于印刷电路板的编辑, (7)用于自动布线。(1)中的 EDIT. *** 是原理图编辑程序, *** DRV 是图形驱动程序, DEMO. *** 是演示文件。(2)和(3)中全是元件图形符号库的库文件。(4)中的 POST.COM 用于生成连线表, PLOT.COM 用于绘图, COMPILE.COM 和 DECOMP.COM 分别用于元件图形符号库的合成与分解, NETTRAN.COM 则用来进行连线表的格式转换。(5)中有印刷电路板编辑程序 EDIT. *** , 图形显示驱动程序 ***. DRV 和演示文件 DEMO.PCB。(6)中有元件图形封装图形库和一些附加功能程序, 如校对网络表用的 NETCHECK.COM, 将网络表转换成连线表的 NET2WIRE.COM 和压缩元件库空间的 COMPACT.COM。(7)中则主要是用于自动布线的文件。

二、TANGO 系统的使用要点 由于篇幅有限, 这里仅介绍一下 TANGO 系统的安装和使用时的操作

要点。

1. TANGO 系统的安装

虽然 TANGO 系统可以在软盘上运行, 但是最好把它安装到硬盘上去, 这样可以大大加快运行速度。其安装步骤如下:

a. 在硬盘上为 TANGO 系统建立子目录, 其典型形式如图 1 所示。

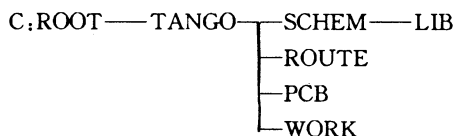


图 1 TANGO 的目录结构

b. 把 Tango-Schematic-EDIT 盘和 Tango-Schematic-UTILITY 盘上的文件拷贝到 C:\TANGO\Schem 目录中。

c. 把 LIB1 盘和 LIB2 盘上所有文件拷贝到 C:\TANGO\SCHEM\LIB 目录中, 形成元件图形库目录。

d. 把 Tango-PCB-EDIT 盘和 Tango-PCB-LIB\UTILITY 盘上的文件拷贝到 C:\TANGO\PCB 目录中。

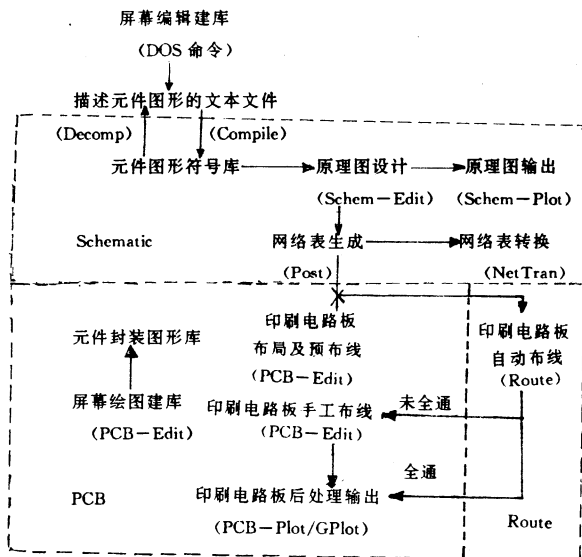


图 2 TANO 系统的工作流程图

e. 把 Tango-ROUTE 盘上的文件拷贝到 C:\TANGO\ROUTE 目录中。

f. WORK 子目录作为工作区,用来存放将要生成的各种图形文件。

完成了以上几个步骤之后,就可以单独使用硬盘来运行 TANGO 系统了。

2. 设置显示方式

TANGO 系统的三个部分均带有显示驱动程序。每个部分在第一次运行前都需要根据用户所用显示器的形式,执行相应的驱动程序,以设置适当的显示方式。为了便于操作,系统中备有四个批命令文件,即 CGA. BAT、EGA. BAT、HERC. BAT 和 AGA. BAT。这样用户就可以根据自己的需要,运行这些批命令文件来完成显示方式的设置。例如,使用 CGA 卡的用户,只需运行 CGA. BAT 即可。显示方式只需设置一次,不必每次都做。

3. 使用 TANGO 系统进行设计的基本流程

这个基本流程如图 2 所示,图中的虚线框即为 TANGO 系统的三个组成部分,汉字下方的文件名即为实现此功能所需的命令。

总的来讲,TANGO 系统中的 * * *. COM 文件都具有相对独立的功能,可单独运行。例如,可以用 SCHEM 中的 EDIT 编辑原理图(. S * * 文件)。绘完原理图之后,可用 POST 命令将其转换为网络表文件(. NET 文件)。网络表文件既可以与相应的布局文件(. PCB 文件)一起供 ROUTE 自动布线用,亦可通过 NETTRAN 命令转换成其它格式的网络表,供其它电子 CAD 系统使用。PCB 中的 EDIT 用于印刷电路板

的手工布局和手工布线,它最多可以布八层板,即一个电源层,一个地线层和六个信号层。在使用自动布线功能前,必须用它调用元件封装图形库安排好各元件在印刷电路板上的位置,做好回避区和印刷电路板的外形轮廓线,并做好关键通路的预布线,以形成一个布局文件(. PCB)。ROUTE 中的 ROUTE 命令则可以在(. PCB)文件和(. NET)文件的基础上进行自动布线,产生一个在原有文件名前加了“R”的(. PCB)文件。“R”表明该(. PCB)文件已做过自动布线。当然,用户也可以在运行 ROUTE 时通过人——机对话重新指定文件名。应当说明的一点是 ROUTE 的自动布线功能仅限于四层以内,即最多能自动布一个电源层、一个地线层和两个信号层。超出四层的印刷电路板都只能用 PCB—EDIT 手工布线。由于 ROUTE 的布通率极高,而一般的设计大多在四层以内,所以这并不妨碍 TANGO 的实际应用。

4. TANGO 系统的键盘操作

TANGO 系统中最频繁的键操作主要集中在两个 EDIT 文件中,即原理图编辑和印刷电路板编辑。整个系统的操作形式基本上是统一的,在使用过程中可以随时按“?”键求得在线帮助。此外,TANGO 系统还具有若干命令菜单供用户选择,这些功能选择项可单键触发,一般就是其命令字中最亮的那个键。

由于篇幅的限制,有关 TANGO 系统更详尽的特性、操作方法和实用技巧(此处从略),请大家查阅其使用手册。需此软件者可同作者或本刊联系。

用 True BASIC 语言直接对 dBASE III 数据文件进行操作

中南工业大学 资源开发工程系 向南平

dBASE III 数据库管理系统是实现管理科学化和现代化的一项有力工具,然而,尽管该系统本身已具有很强的数据处理能力,但对于大量复杂的计算及图形处理等综合能力就远不如其他高级语言了。因此,在开发数据库管理系统时,遇到类似的问题,总是借助于高级语言来解决。这就涉及到数据库文件与高级语言间数据的传递问题。

目前广泛使用的方法是先建立 dBASE III 的 DBF 数据文件,然后利用 dBASE III 的 COPY 命令把 DBF 文件的数据送入一个 TXT 文件,高级语言则是通过

对 TXT 文件进行操作,从而实现数据的传递。这种方法固然可行,但使用并不方便,每当数据库 DBF 文件进行了修改时,就必须重新生成 TXT 文件,这无疑是一种浪费。那么,能否直接对 DBF 文件进行操作,从而减少操作步骤呢?答案是肯定的,关键在于对 DBF 文件的了解程度。因此,本文将从剖析 dBASE III 数据库文件的结构入手,谈谈如何用 True BASIC 语言直接对 dBASE III 数据库文件进行操作的若干问题。

字节	内容	含义
1	83H 或 03H	DBF 文件标识字, 含 M 型字段时为 83H, 否则为 03H。
2-4	日期	最后一次修改 DBF 文件的日期, 依次为年、月、日, 各占一个字节, 其中年是以第二字节的内容加上 1900。
5-8	记录个数	DBF 文件中当前记录个数, 低位在前, 高位在后。因此数据库实际能容纳的最多记录数约为 22 亿, 即 5-8 字节不大于 FF FF FF 7FH。
9-10	地址	DBF 文件实际数据相对于文件开始位置的起始地址。
11-12	记录长度	存放记录长度, 低位在前, 高位在后。dBASE III 限制记录长度在 4000 个字符数内。即 11-12 字节不能超过 A00FH。
13-32		未用, 全为 00。

18	小数位数	当字段为实型时, 存放小数位数, 否则为 0。
19-32	未用。	

文件结构说明部分除字段名和类型标志用 ASCII 码表示外, 其余均为 16 进制数。文件说明部分以 0DH、00H 为结束标志。

2. 数据部分

DBF 文件的实际数据是紧接着 0DH、00H 之后存放。每个记录的第一个字节为记录删除标志, 当使用 DELETE 命令删除该记录, 但未使用 PACK 命令作物理删除时, 其内容为 2A(* 的 ASCII 码), 否则为 20 (空格的 ASCII 码); 其后连续以 ASCII 码形式按顺序和长度存放各字段数据, 其中, 字符型的数据是左对齐, 数值型的数据是右对齐, 不足字段长部分用空格补足。每个字符占一个字节。数据部分以 1A(CTRL+Z) 结束。

二、True BASIC 语言与 DBF 文件的数据传递

了解了上述 DBF 文件的结构特点, 就可解决高级语言与 DBF 文件的数据传递问题了。由于 DBF 文件是以字节为存储单位, 所以可利用 True BASIC 的字节文件直接对 DBF 数据文件进行读写操作。

设数据库文件为 DWZB.DBF, 其结构如下:

字段	字段名	类型	宽度	小数
1	点名	C	10	
2	X 坐标	N	12	4
3	Y 坐标	N	12	4
4	高程	N	9	3
5	点的等级	C	4	
	** 总计 **		48	

一、dBASE III 数据库文件的结构特点 DBF 文件由两大部分组成: 第一部分是文件结构的说明, 第二部分是实际的数据。以下分别说明各部分的结构特点。

1. 文件结构说明部分:

文件结构说明部分的长度为: (文件字段数+1)*32+2 个字节。其中前 32 个字节存放着有关文件类型、记录数、记录长度等重要信息, 各字节的内容及含义见表一。

从第 33 字节开始, 存放着各字段的说明部分, 每字段占 32 个字节, 有关字节的内容及含义见表二。

表二

字节	内容	含义
1-10	字段名	字段名最长为 10 个字符, 不足 10 个后面补 0。
11	未用	
12	类型标志	为 N、C、D、L、M。
13-16	未用	
17	字段宽度	由于只有一个字节, 所以最大宽度为 254。

用以下 True BASIC 程序可实现对该 DBF 文件数据的读取, 设要读取文件中的第 10 条到第 25 条共 16 条记录的内容, 并分别存储在 ZB 和 DM\$ 数组中。


```

DIM ZB(16,3),DM$(16,2)           ! 设置数组
OPEN #1,NAME "DWZB.DBF",ACCESS INPUT ,ORGNIZATION BYTE ! 打开文件
READ #1,BYTES (5+1)*32+2:A$        ! 读文件结构说明部分
READ #1,BYTES 9*48:A$              ! 读文件 的前 9 条记录
SET #1,RECSIZE 48                   ! 设置后续每次读取的字节数(=记录长度)
FOR I=1 TO 16
    READ #1:A$                      ! 读取等于记录长度的字节数并暂存在 A$ 变量中
    LET DM$(I,1)=A$[2:11]           ! 从 A$ 中提取点名
    LET ZB(I,1)=VAL(A$[12:23])      ! 从 A$ 中提取 X 坐标并将字符型数据转变成数值
    LET ZB(I,2)=VAL(A$[24:35])      ! 从 A$ 中提取 Y 坐标并将字符型数据转变成数值
    LET ZB(I,3)=VAL(A$[36:44])      ! 从 A$ 中提取 H 高程并将字符型数据转变成数值
    LET DM$(I,2)=A$[45:48]          ! 从 A$ 中提取点的等级
NEXT I
CLOSE #1                            ! 关闭文件
END

```

同样,若需对 DBF 文件中的某些记录进行替换,也可用这种方法,但需另设置一个输出文件。例如需要修改上述文件中的第 10 至 25 条记录,则先由原文件中读出文件结构说明部分,并写入新建的输出文件中;再读取数据部分,对不需修改的前 9 条记录,从原文件中读出后直接写入输出文件中,对需修改或替换的记录,则从原文件中读出并进行修改后再写入到输出文件中;最后再将后续记录从原文件中读出直接写入到输出文件中,直到数据终了标志 1AH。需要注意的是,新的记录内容应与原记录内容的字节对等。

三、用高级语言实现对 DBF 文件的分解和合并
在数据库的使用中,常常会遇到这样的问题,即,因为某些原因,需将原有库分解成几个子库或需将几个子库合并成一个大库。使用 dBASE III 本身提供的 COPY 和 APPEND 命令虽然也可完成,但当条件比较复杂时,并不是很方便的。例如,需对某一个库进行分解,分解的条件是取决于从另一个关连库里读出的内容。解决同样的问题,使用高级语言比用 dBASE III 可

能更方便、效率更高一些。

设有一个 DXD.DBF 库的结构如下:

字段 字段名 类型 宽度 小数

1	导线路线	C	12	
2	点名	C	10	
3	标识档案	M	100	
* * 总计 * *				123

DWZB.DBF 库与 DXD.DBF 库是关连库,前者储存的是点的坐标、高程及等级,后者储存的是点名、所在的导线路线及标识档案资料,二者的关连字段是点名(这里仅是为了说明问题,没有考虑这两个库本身设计的合理性)。现在需要按导线路线将 DWZB.DBF 库进行分解,采用下一段 True BASIC 语言程序就能很方便地解决这类问题。

```

DIM DM$(50)                        ! 开辟数组以存储点名
INPUT PROMPT "请输入待分出的导线路线号: ";DXLUH$
LET XKMC$=DXLUH$&"·DBF"           ! 新建的库即以导线路线号为名
CALL READ-DXD(DXLUH$,DM$,POINT-NUMBER) ! 调读 DXD.DBF 文件的子程序
CALL FENKU("DWZB.DBF",XKMC$,DM$,POINT-NUMBER) ! 调分库子程序
END
SUB READ-DXD(DXLUH$,DM$(),POINT-NUMBER) ! 读 DXD.DBF 文件的子程序
    OPEN #1,NAME"DXD.DBF",ACCESS INPUT ORGANIZATION BYTE
    READ#1,BYTES(3+1)*32+2:TOTAL$   ! 读文件结构说明部分
    SET #1,RECSIZE 128              ! 设置每次读取的字节数

```



```

LET POINT - NUMBER = 0
DO WHILE MORE #1                                ! 读 DXD. DBF 文件数据
READ #1:TOTAL $                                  ! 读 128 个字节
LET DXH $ = TRIM $ (TOTAL $ [2:13])              ! 将所读出内容的第 2-13 字节存在
                                                    DXH $ 中
IF DXH $ = DXLUH $ THEN                          ! 判断 DXH $ 是否等于所要求的导线路
                                                    线号
LET POINT - NUMBER = POINT - NUMBER + 1          ! 若是, 则点数加 1
LET DM $ (POINT - NUMBER) = TRIM $ (TOTAL $ [14:23]) ! 点号存于 DM $ 数组中
END IF

LOOP
CLOSE #1
END SUB

SUB FENKU (READ - DBF $, WRITE - DBF $, DM $ (), POINT - NUMBER) ! 分库子程序
OPEN #4:NAME WRITE - DBF $, CREATE NEW ACCESS OUTPUT, ORGANIZATION BYTE
ERASE #4                                           ! 清除原库中可能存在的记录
OPEN #3:NAME READ - DBF $, ACCESS INPUT, ORGANIZATION BYTE
READ #3,BYTES(5+1)*32+2:TOTAL $                  ! 从原库中读文件结构说明部分
WRITE #4:TOTAL $                                  ! 将文件说明部分写入新库
SET #3:RECSIZE 48                                ! 设置从原库中每次读取的字节数
DO WHILE MORE #3
READ #3:TOTAL $                                  ! 从原库中读出 48 个字节
LET A $ = TRIM $ (TOTAL $ (2:11))                ! 将读出内容的第 2-11 字节存于 A $ 中
FOR I=1 TO POINT - NUMBER
IF A $ = DM $ (I) THEN                          ! 与从 DXD. DBF 库中读出的点名比较
WRITE #4:TOTAL $                                ! 若是满足条件的点就写入新库
LET RECORD - NUMBER = RECORD - NUMBER + 1        ! 新库记录数加 1
END IF
NEXT I
LOOP
WRITE #4:CHR $ (26)                              ! 写 DBF 文件的结束标志
SET #4:POINTER begin                             ! 重置文件指针到文件开始处
WRITE #4:CHR $ (03)                              ! 写文件标识字, 若含 M 型字段, 则写入 CHR
                                                    $ (83)
WRITE #4:CHR $ (INT (date/1000))                  ! 写日期的年
WRITE #4:CHR $ (NUM (DATE $ [5:6])) & CHR $ (NUM (DATE $ [7:8])) ! 写月日
LET rec4 = int (record - number / 16 ^ 6)          ! 根据新库的记录数计算库中
LET rec3 = int ((record - number - rec4 * 16 ^ 6) / 16 ^ 4) ! 记录计数器各字节的内容
LET rec2 = int ((record - number - rec4 * 16 ^ 6 - rec3 * 16 ^ 4) / 256)
LET rec1 = record - number - rec4 * 16 ^ 6 - rec3 * 16 ^ 4 - rec2 * 16 ^ 2
WRITE #4:chr $ (rec1) & CHR $ (rec2) & CHR $ (rec3) & CHR $ (rec4) ! 写记录数
CLOSE #3
END SUB

上述程序是先从 DXD. DBF 库中读出所要求的导 线路线上的所有点名, 并存储在 DM $ 数组中, 然后,

```


读 DXZB.DBF 库,半判断所读出的每一个点是否为满足条件的点,如果满足条件即写入到新库中去。新库生成后,还需根据新库的实际记录数,修改文件结构说明部分中记录计数器各字节的内容。

以上说明了如何从大型数据库文件分解出满足条件的一个子库。如果需要由几个子库合并成一个大库,也可采用同样的办法。先将第一个子库的结束标志去掉,并把数据指针移到文件的结尾处,再从其他子库中读取记录,并追加到第一个库中,追加完成后写上文件结束标志。最后,根据文件总的记录数修改文件结构

说明部分中记录计数器各字节的内容。

综上所述,只要我们对数据库文件的总体结构有了比较深刻的了解,就可对其进行随心所欲的操作。进一步,如果对磁盘数据文件存储方式也比较了解的话,还可以做一些 dBASE III 本身所无法完成的工作。如 dBASE III 文件的恢复(用 USE 语句可以打开)及数据记录丢失后的恢复,还可以通过改变 DBF 文件标识字、记录计数器或其他有关字节的内容给 DBF 文件加密。

以上是笔者的一些肤浅的体会,仅供参考。

介绍 IBM 局域网络支持程序

吉林省计算机技术研究所 徐世坤 陈仲 姚学英

摘要 LAN 支持程序是其它网络软件及应用程序的基础。LAN 支持程序具有灵活性,用户可根据自己的需要作出不同的选择。此外,LAN 支持程序使用方法简单,便于掌握。本文介绍了 LAN 支持程序的使用及主要参数等。

一、引言 IBM 局域网络(Local Area Network,以下简称 LAN)是指 IBM 令牌环网络(Token-Ring Network,以下简称 T-R 网络)和 IBM PC 网络(PC Network)。IBM LAN 支持程序(IBM LAN Support Program)是为上面的两种网络在 DOS 环境下运行提供硬件设备支持接口及软件支持。

IBM LAN 指 T-R 网络和 PC 网络,而 PC 网络又分为 PC 宽带网和 PC 基带网。

T-R 网络是采用环型拓扑结构,使用单令牌(Token)控制的存取方法,通信介质是双绞线,通讯速率为 4MB。另外,IBM 还宣布了一种屏蔽双绞线上通讯速率为 16MB 的产品。

PC 宽带网是采用树型拓扑结构,使用 IEEE802.3 CSMA/CD 控制存取方法。通信介质是同轴电缆,通讯速率为 2MB。

PC 基带网采用星型单总线拓扑结构,使用 IEEE802.3 CSMA/CD 控制存取方法。通信介质是双绞线,通讯速率为 2MB。

两种网络使用的主要软件有 IBM LAN 支持程序(本文仅对目前较流行的 1.00 版所提供的功能作简单介绍)和 IBM LAN 程序等。

LAN 支持程序是一组设备驱动程序,(DEVICE DRIVE)。(设备驱动程序是为一些设备,如:打印机,键盘或适配器提供软件接口的程序)这些设备驱动程序是以“device=设备驱动程序名及参数”的格式存于计算机的配置文件 CONFIG.SYS 中,在网络启动时装入内存。

LAN 支持程序有以下作用:

- 提供 LAM 网络板支持及相关的程序接口。
- 为网络应用程序访问网络板提供接口;
- 允许用户在一个网络上运行另一个网络的网络程序。

LAN 支持程序所支持的网络板有:T-R 网用于总线结构的 I 型板(PC Adapter)、II 型板(PC Adapter II)和用于微通道结构/A 型板(Adapter/A);PC 网用于总线结构的 II 型板(Adapter II)和用于微通道结构的 II/A 型板(Adapter II/A)。

二、LAN 支持程序及使用

A. LAN 支持程序

LAN 支持程序是一组设备驱动程序,这些设备驱动程序都有 SYS 扩展名,文件名是这样构成:以 DXM 开头,以 MOD 结尾,在二者之间用两个字符来区别,

不同的设备驱动程序,如:DXMC0 MOD. SYS 为 T-R 网络板的设备驱动程序,=DXMG0 MOD. SYS 为 PC 网络板的设备驱动程序。

LAN 支持程序主要有以下几个文件:

• DXMA0MOD. SYS 该设备驱动程序被称为中断仲裁程序(interrupt arbitrator),它一般不带参数。这是一个必选的设备驱动程序,就是说所有不同的网络配置,都要选择这个设备驱动程序。

• DXMC0MOD. SYS(参数) 这是 T-R 网络板支持程序,其作用是为 T-R 网络板提供支持,并提供 IEEE802.2 程序支持。这个设备驱动程序可支持一块或两块 T-R 网络板。

• DXMC1MOD. SYS(参数) 这也是 T-R 网络板支持程序,其作用同上面的 DXMC0MOD. SYS,并且还支持 3270-PC 控制程序。也可支持一块或两块 T-R 网络板。

DXMG0 MOD. SYS(参数) 这是 PC 网络板支持程序,其作用是为 PC 网络板提供支持,并且提供 IEEE802.2 程序支持。此设备驱动程序支持一块或两块 PC 网络板。

DXMG1 MOD. SYS(参数) 这也是 PC 网络板支持程序,其作用同上面的 DXMG0MOD. SYS,并且支持 3270-PC 控制程序,也可支持一块或两块 PC 网络板。

在前面介绍的四个网络板支持程序中,每个网络板支持程序都可选择参数(参数及其如何选择在后面介绍),并且最多可支持两块网络板。在一般情况下,一台计算机中插入两块 T-R 网络板或两块 PC 网络板,其目的是为了实现在两个 T-R 网络之间的或两个 PC 网络之间通信和资源共享。这台计算机被称为网桥(Bridge)。

• DXMT0MOD. SYS(参数) 这是 NETBIOS 支持程序,其作用是提供 NETBIOS 程序支持,它有许多参数可供选用。这个设备驱动程序必须与中断仲裁程序和一个网络板支持程序一起使用。

B. LAN 支持程序的使用

现举例说明如何使用 LAN 支持程序。

如果你有 T-R 网络板一块,需要 IEEE802.2 和 NETBIOS 接口支持,那么在网络引导盘上要包含下面三个设备驱动程序,并且需要建立或编辑 CONFIG. SYS 文件,使之包含下面的三条命令:

```
device=dxma0mod. sys
```

```
device=dxmc0mod. sys
```

```
device=dxmt0mod. sys
```

这样,当用网络引导盘启动计算机时,你需要的设备驱动程序就装入内存了。

需要指出的是,在 CONFIG. SYS 文件中,中断仲裁程序要列在网络板支持程序之前,而 NETBIOS 设备驱动程序要列在网络板支持程序之后。

下面的表列出了 LAN 支持程序所支持的网络板类型和软件接口,用户可根据自己的要求选择设备驱动程序。

支持的 网络板	支持的软件接口			
	IEEE 802.2	IEEE 802.2 各 NET- BIOS	IEEE 802.2 和 3270-PC 控制程序	IEEE 802.2 NETBIOS 和 3270- PC 控制程序
T-R 网络 I 型板 II 型板 /A 型板	A0 C0	A0 C0 T0	A0 C1	A0 C1 T0
PC 网络 I 型板 II /A 型板	A0 G0	A0 G0 T0	A0 G1	A0 G1 T0

表中: A0 = CXMA0MOD. SYS, G0 = DXMG0MOD. SYS

C0 = DXMC0MOD. SYS, G1 = DXMG1MOD. SYS

C1 = DXMC1MOD. SYS, T0 = DXMT0MOD. SYS

(以后图同)

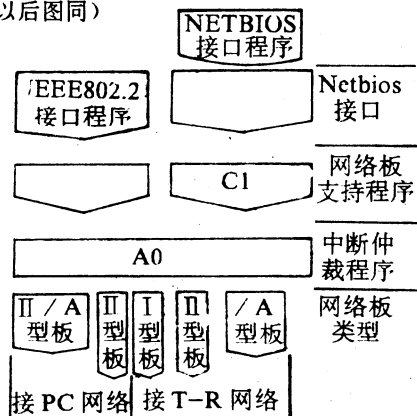


图 1

下面举例说明如何选择设备驱动程序来配置 CONFIG. SYS 文件。

例: 3270-PC 中接一块 T-R 网络板,它将使用 3270-PC 控制程序和 IEEE802.2 接口程序,但不用 NETBIOS 接口程序,那么在 CONFIG. SYS 文件中装入下面的设备驱动程序即可。


```
device=dxma0mod.sys
```

```
device=dxmc1mod.sys
```

例:计算机中有 T-R 网络板和 PC 网络板各一块,使用 IEEE802.2 接口程序和 NETBIOS 接口程序,不用 3270-PC 控制程序,那么在 CONFIG.SYS 文件中装入下面的设备驱动程序即可。

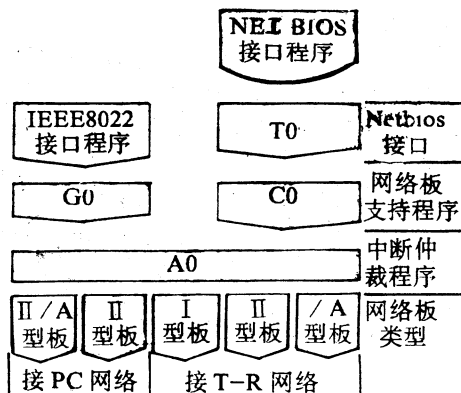


图 2

```
device=dxma0mod.sys
```

```
device=dxmc0mod.sys,D800
```

```
device=dxmg0mod.sys
```

```
device=dxmt0mod.sys
```

注意,当一块 T-R 网络板和 PC 网络板插入同一系统中时,在 dxmc0mod.sys 设备驱动程序中要带有参数“D800”。在同一计算机中插入两块不同类型的网络的网络板,其目的是为了实现 T-R 网络与 PC 网络之间的通信和资源共享,这台计算机被称为网间连接器(Gateway)。

当有两块网络板插入同一台计算机时(网桥或网间连接器),一块板必须设置成主板(PRIMARY),而另一块板设置成次板(ALTERNATE)。这一过程一般是通过设置板上的开关来实现的。另外,还要修改网络板设备驱动程序的参数。

C. CONFIG.SYS 文件中的其它配置

在 IBM LAN 中,每个网络结点(网络服务器或工作站)的 CONFIG.SYS 文件中仅包含前面介绍的设备驱动程序是不够的,为了更好地使用网络,可能还需下面的命令:

• BUFFERS=XX(XX 是十进制的 1~99)

使用这个命令的作用是,在 DOS 启动时可以得到磁盘缓冲区在内存中分配的个数。对于 IBM LAN 的服务器配置,一般要选择:BUFFERS=20。

• FCBS=m,n(m 是十进制的 1~255,n 是 0~255)

使用 FCBS 命令能够定义被同时打开的文件控制块(FCBS)的数量。在 IBM LAN 的服务器的配置中,若有共享文件,就要选用这个命令,一般要选择:FCBS=16,8 或更大一些。

• FILES=XXX(XXX 是十进制的 8~255)

• FILES 命令给出用户可以同时打开的文件的最大值。一般 FILES 给定的值必须足够大,以满足网络上应用程序运行的需要。

• LASTDRIVE=X(X 是英文字母的 A~Z)

使用 LASTDRIVE 命令可以给出在 DOS 下能同时使用的虚拟和实际驱动器的最后的一个。缺省值是 E,用户可根据自己的需要设置这个参数。

三、LAN 支持程序的主要参数 T-R 网络板的支持程序有两种参数可供选用:网络板地址和共享 RAM 地址段;PC 网络板的支持程序也有两种参数可被选用:网络板地址和工作区大小。这两个设备驱动程序的网板地址参数的意义和使用方法相同。NETBIOS 支持程序有许多可选用的参数,其参数是由关键字和值两部分组成。

为便于介绍,将 T-R 网络板的支持程序 DXMC0MOD.SYS 和 DXMC1MOD.SYS 用 DXMCnMOD.SYS 代替;PC 网络板支持程序 DXMG0MOD.SYS 和 DXMG1MOD.SYS 用 DXMGnMOD.SYS 代替。

A. 网络板支持程序参数

• 网络板地址 不论是 T-R 网络还是 PC 网络,其网络板都有一个地址,这个地址在网络板制造时被指定,并存入网络板的存储器中,网络板支持程序的“网络板地址”参数的缺省值指的就是这个地址。如果选择网络管理员,那么“网络地址”参数一般要指定。“网络板地址”这一参数是一个范围,用十六进制表示是 4000 0000 0000 到 40007FFFFFFF,习惯上用十进制表示,其范围是 4000 0000 0000 到 4000 7999 9999。此范围以外的值可能与网络的其它产品相冲突。值得注意的是,在同一网络中,每一块网络板的网络板地址是唯一的,否则在使用网络时将产生错误。

• 共享 RAM 地址段 T-R 网络板包含与计算机共享的 RAM,缺省值主板是 D800 段,次板是 D400 段(D800 段和 D400 段分别指内存地址的 D8000 和 D4000)。如果系统安装时使用了这些段,那么需要指定其它没被使用的地址段。正确的共享 RAM 地址段必须由四位十六进制数组成,用来表示五位十六进制地址的高四位。这个地址一般要 16KB 为界。T-R 网络的 I 型板需要 8KB,而 II 型板和/A 型板需要

16KB。

T-R 网络板支持程序带参数的格式:

DEVICE = DXMCnMOD. SYS addr0, mem0, addr1, mem1

其中:

addr0: 主板或只有一块网络板时的网络板地址。

mem0: 主板或只有一块网络板时的共享 RAM 地址段。

addr1: 次板的网络板地址。

mem1: 次板的共享 RAM 地址段。

规则:

- 主板参数在前, 次板参数在后。
- 用空格将参数表与“DEVICE=DXM CnMOD. SYS 分开。
- 用一个逗号将一个参数与其它参数分开。
- 省略的参数用逗号代替, 若省略的参数在参数表的尾部, 则不用逗号代替。

例: DEVICE = DXMCnMOD • SYS 4000 0000 0001

例中, “4000 0000 0001”是主板的新地址, 取代网络板地址的缺省值。所有的其它缺省值仍然保留。

例: DEVICE=DXMCnMOD • SYS, C200

例中, “C200”是主板的共享 RAM 地址段, 其它参数取缺省值。“C200”前面的逗号表示主板网络板地址缺省值。

例: DEVICE = DXMCnMOD. SYS 4000 0001, C200, 4000 0000 0002, AA00

例中, 主板和次板的网络板地址和共享 RAM 地址段全部修改。

工作区

PC 网络支持程序允许每块板内部工作区为 64KB。工作区的大小取决于应用程序的要求, 每块板工作区大小的缺省值为 8KB。

每块板的要求	工作区大小
缺省值	8KB
少于 16 个会话期	8KB
16—23 个会话期	12KB
24—32 个会话期	16KB

此表只是参考, 当会话期和信息量增加时, 工作区也要增大。

PC 网络板支持程序带参数的格式:

DEVICE = DXMGnMOD. SYS addr0, wrk0, addr1, wrk1

其中:

addr0: 主板或只有一块网络板时的网络板地址。

wrk0: 主板或只有一块网络板时的工作区大小。

addr1: 次板的网络板地址。

wrk1: 次板的工作区大小。

有关规则同前面的 T-R 网络板支持程序参数使用规则。

例: DEVICE = DXMGnMOD 4000 0000 0001,, 4000 0000 0002

例中, “4000 0000 0001”是主板的网络板地址, “4000 0000 0002”是次板的网络板地址。第二个逗号表示主板的工作区参数取缺省值。次板的工作区参数也取缺省值。

例: DEVICE=DXMGnMOD. SYS, 20

例中, “20”是主板的工作区大小(20KB)。所有的其它参数取缺省值。“20”前面的逗号表示主板的网络板地址省略。

B. NETBIOS 支持程序主要参数

NETBIOS 支持程序参数是由关键字和值组成。有许多可用的关键字, 下面仅对比较重要的几个加以介绍。

每个关键字都有其简写的形式, 表示方法在每个关键字后面的括号中。

STATIONS=(ST=)

• 指能被定义的 NETBIOS 接口设备驱动程序连接的工作站的最大数目。

• 必须是十进制的 0~254。

——实际上 T-R 网络中, I 型板最大值是 32, II 型板和/A 型板最大值是 64。

——关键字 EXTRA. STATIONS(作用是请求 NETBIOS 设备驱动程序以获得附加的连接站的数目)的值, 减小 NETBIOS 接口设备驱动程序的实际最大值。

• 若省略或选择 0, 则取缺省值 6。

SESSIONS=(S=)

• 指能被定义的 NETBIOS 接口设备驱动程序会话期的最大值。

• 必须是十进制的 0~255。

• 如果省略或选择 0, 取缺省值 12。

NAMES=(N=)

• 指可被定义的, 并且有网络板地址的 NETBIOS 接口设备驱动程序的最大值。

• 必须是十进制的 0~254。

• 如果省略或选择 0, 则取缺省值 17。

• 如果选择 1, 这个值被设置成 2。

下面举例说明参数的用法:

例: DEVICE=DXMT0MOD. SYS ST=64 S=64 N=32 C=32

例中, 主板或只有一块网络板选择部分关键字, 或者主板选择部分关键字而对次板全部的关键字取缺省值。关键字之间用空格分开。

例: DEVICE=DXMT0MOD. SYS /ST=16 S=32 C=32

例中, 对次板选择部分关键字, 对主板的全部关键字取缺省值。“/”后面是次板的关键字。

例: DEVICE=DXMT0MOD. SYS ST=16 S=32 C=16 /ST=8 S=8

例中, 主板和次板都选择关键字。关键字与顺序无关, 但要把主板与次板的关键字用“/”分开。

例: DEVICE=DXMT0MOD. SYS ST=32 S=32 C=16 /=

例中, 主板选择部分关键字, 并且次板定义与主板相同的关键字。

四、结束语 LAN 支持程序为 IBM LAN 提供了网络支持和它相关的软件接口, 并且提供了=NET-BIOS 的全部功能, 同时又使得网桥和网间连接器得以

实现。

为方便用户, 使用 LAN 支持程序, LAN 支持程序提供了一个配置帮助(Configuration Aid)程序, 这个配置帮助可帮助你选择设备驱动程序, 自动生成 CONFIG. SYS 文件中的相应命令。但是, 若使用配置帮助, 那么设备驱动程序不能使用参数(或者说, 全部参数都取缺省值)。

为方便用户使用设备驱动程序及参数, LAN 支持程序提供了一些信息, 这些信息主要是错误信息, 当用户使用设备驱动程序或参数有错时, 错误信息就会显示出来, 这将有助于你查找出错原因和改正错误。

参考文献

- IBM LAN Support Program User's Guide
- DXMINFO. DOC file on the LAN Support Program diskette
- IBM PC Local Area Network Program
- IBM Token — Ring Network PC Adapter Guide to Operation
- The IBM Token—Ring Network Decision

• 专题 •

dBASE 和高级语言参数直接传递方法—— 桥地址思想

中国科学院计算中心 王敏生

一、问题的提出 dBASE(FOXBASE)是世界上用得最多的数据库, 据估计, 全世界已有几百万台微机配备了 dBASE, FOXBASE 数据库, 我国有微机 40 多万台, 大约有 20 多万台微机配备 dBASE, FOXBASE 软件, 并利用它编制各种管理软件。全国大约有百余万人熟悉使用 dBASE, FOXBASE 软件, 我国的办公室自动化主要是在 dBASE(FOXBASE)的基础上编制各种应用软件, 上至国家计委的 OA 系统, 中央电视台的节目管理系统, 民航局的订票系统, 下至各省市计经委的办公室, 一般企业工厂的办公管理系统, 都采用 dBASE(FOXBASE)编程序, 单国家计委每年就要花几十万元用 dBASE(FOXBASE)编应用管理系统, 可见 dBASE(FOXBASE)的应用范围之广。

dBASE(或 FOXBASE)作为一种人机对话数据库

语言, 具有操作容易, 编程较为简单, 开发周期短的特点, 但是因为 dBASE(或 FOXBASE)自身的特点, dBASE(或 FOXBASE)实时功能不如 C, PASCAL, 科学计算不如 FORTRAN 语言, 事务处理不如 COBOL 语言, 图形功能不如 PASCAL 和 BASICA, C, 在 DOS 操作系统功能调用和系统接口及其 BIOS 系统接口方面大大不如 C 和汇编语言, 而这一切又大大与应用系统的运行开发效率有关系, 在这一方面, C 和汇编语言一花独秀, C 和汇编语言有很好的编程接口, C 和汇编语言可编制许多高效率, 占内存少的与系统功能有关的应用软件。如前后台处理, 通讯和图形等方面。这些方面都是 dBASE(或 FOXBASE)所难以或无法实现的, 这说明, 要开发高效率的应用软件, 必须把 dBASE 和高级语言混合使用, 这样才能充分发挥各种

语言的特点,取长补短,才能够大幅度提高应用软件开发运行效率。

在任何应用系统中,参数传递起着举足轻重的地位,它既是与用户沟通信息的手段之一,一方面通过它接受用户要求系统处理的数据;另一方面,可以通过它把计算机运行的结果告诉用户,让用户得到所需要的数据。另外,它又是程序中的语句进行交流,进程之间进行通信必不可少的工具。可以说,程序的走向是通过系统赋予变量的不同值来实现,系统中的参数,好比一座桥梁,把程序的语句连结成为一个功能独立的模块,把各进程有机连结成为一个功能齐备,能够高效率运行的计算机系统!

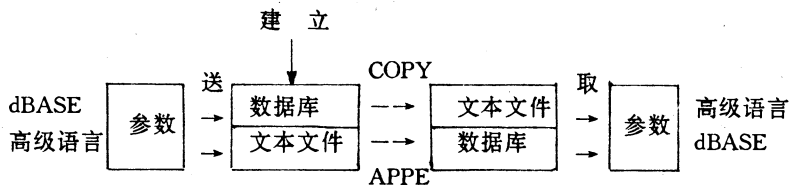
二、各种语言混合编程困难所在 dBASE(或FOXBASE)和高级语言混合编程,必须解决dBASE(FOXBASE)和高级语言参数传递(即变量的引用修改)问题。

在同种语言编程的系统中,如果处理得当(如说明为全局变量)则相同的变量名,系统总是把它解释为相同的内存地址,因此,在不同的子程序中,对同名变量的赋值、修改、引用,都是对同一个内存地址的存取,进

程之间的变量传递,就是通过此种途径来实现的。

在混合语言编程的系统中,对源程序的解释,编译是由两个以上语言解释(或编译)系统来完成的,由于语言系统对变量的管理自成一套,因而,同名变量,在不同的语言系统中,被解释成不同的内存地址,从而使参数传递变得困难,在混合语言的编程的系统中,对程序员的要求极为苛刻,要求程序员充分了解各种语言的内部管理,严格遵守各种语言对编程中参数传递的约定,并且运用编程技巧,才有可能解决参数传递,例如为了解决 Microsoft 公司所生产的软件 C, PASCAL, BASICA, FORTRAN, 汇编语言等之间参数传递, Microsoft 公司专门出了《各种语言的混合编程指南》一书,用以介绍混合语言编程中参数的传递方法,但其方法也只适应 Microsoft 公司所生产的各种语言软件,对别的公司生产的软件能否适应,不得而知。

dBASE(或 FOXBASE)是一种面向用户的数据库语言,它的内部管理人们所知不多,这便加重了 dBASE(或 FOXBASE)与高级语言参数传递的困难,在 dBASE(或 FOXBASE)与高级语言的混合编程系统中,现今惯用传递办法如图所示



从图中可以看出,为了传递一个参数,我们必须通过中间文件作为媒介来传递,这样的传递,使系统的开发、运行处于低效率之中,这极大地妨碍高层次应用系统的开发,因此,在 dBASE(或 FOXBASE)与高级语言的混合编程系统中,要高效率开发,运行应用系统,必须解决参数直接传递问题。

三、dBASE 和高级语言参数直接传递方法——桥地址思想 dBASE 和高级语言参数直接传递,是提高应用系统开发运行效率的关键,笔者在多年的实践中认识到,采用传统的方法,即把一个语言系统的参数值送往另一种语言的内存变量的内存位置,这样实现起来难度较大,而且价值不高,因而必须另辟捷径,才能彻底解决问题。

我们认为 dBASE(或 FOXBASE)与高级语言混合编程困难在于两个不同的语言解释,编译系统对变量的管理不同,因而两个系统的同名变量,实际是两个内存地址的代名词,传递参数要求两个语言系统的同名变量管理相一致是不可能的,但我们可从另一角度

来考虑这一问题,即在两个系统的变量中,设想有一系带,通过这一系带,在两个系统的变量中建立某种联系,从而实现参数传递,这就是我们的桥地址思想,简单而自然。

桥地址思想:我们在内存地址开辟一个区域,用以存放要传递参数的像(即内容),dBASE(或 FOXBASE)与高级语言的参数传递转化为 dBASE(或 FOXBASE)和高级语言对内存区域的存取,这个内存区域好像是为 dBASE(或 FOXBASE)与高级语言之间进行传递搭起的一座“鹊桥”,这就是桥地址的由来。

桥地址概念的引入,使得 dBASE(或 FOXBASE)与高级语言的参数传递这一问题分解为互不相干的 dBASE(或 FOXBASE)与高级语言各自自身对内存区域的存取问题,因此,桥地址的思想可推广应用于高级语言混合编程中的参数传递,dBASE(或 FOXBASE)与高级语言的参数传递正确与否,取决于 dBASE(或 FOXBASE)与高级语言对桥地址存取正确与否。

四、桥地址单元确定 1. 桥地址单元确定原则

我们认为:桥地址是 dBASE(或 FOXBASE)与高级语言进行参数传递的桥梁,它的选取应该是:1)桥地址应是稳定的,内存某个区域被确定为桥地址后,任何系统(包括系统软件,语言软件)的运行,如果不是为了进行参数传递,则不能对桥地址进行存取,只有这样,才能保护参数传递的正确进行。2)桥地址应是 dBASE(或 FOXBASE)和高级语言容易访问的内存单元,这样才能方便存取,简化编程。3)桥地址应该足够大,这样才能容纳所要传递的参数。4)对桥地址的存取规则应是统一的,事先在混合编程前确定好,例如:数值型数据是以 ASC II 码形式还是以二进位存放等等,这样才能使参数传递正确进行。

2. 中断向量表中部分区域作为桥地址区域(称为桥地址 1):

内存中中断向量表即地址 0:0000~0:400 中每 4 个字节位置对应一种中断类型 OH 至 OFFH,这 4 个字节用来存放该级中断处理程序所在的段号及偏移,一般说来,较低位置一般用于内部硬件中断和 DOS 中断,较高位置一般是保留的,服务于用户自己编的应用程序。因而,在较高位的中断地址一般都是 0,中断入口地址除非程序修改中断或设立中段,在程序运行过程中是不会改变的,为此,我们在中断向量表中划分一个单元内容都是 0 的区域来作为桥地址,例如 0:200~0:220,这样的设置,传递参数编程简单明了,但空间不足,稳定性差,程序的兼容性,通用性差。

3. 利用设置中断的方法,把中断处理程序的空间作为桥地址区域(称为桥地址 2)。大家都知道,中断驻留程序在驻留内存后,其驻留空间在运行别的程序时不会被复盖,另外,中断程序的入口地址可通过内存中中断向量表得到,因而利用中断驻留程序的空间作为桥地址空间满足桥地址单元确定的条件,缺点是必须多占用点内存。

上述两个方法可广泛应用于各种高级语言混合编程中的参数直接传递。

4. 利用 dBASE II PLUS(或 FOXBASE)二进制模块所占用的空间作为桥地址区域(称为桥地址 3): dBASE II PLUS(或 FOXBASE)可以将二进制文件加载到内存中去,作为 dBASE II PLUS(或 FOXBASE)的子程序或模块,而不是作为外部程序,另外, dBASE II(或 FOXBASE)还有释放该程序空间以作为它用的命令,因而,如果我们把二进制模块所占用的空间作为桥地址区域的话,其地址可通过某个中断入口地址而得到,则可大大增加编程的灵活性,缺点是这个方法只适应 dBASE II PLUS 或 FOXBASE 与高级语言进行

参数的传递,不能应用于各种高级语言混合编程中的参数直接传递。

桥地址 2 或 3 与桥地址 1 相比较,具有兼容性好,独立性强的优点,缺点是编程较复杂,运行时要多占用内存。

五、桥地址 2 的确立 为了保证程序的最大独立性,当我们设置中断时,有可能与原有的中断发生冲突,因此必须相应的保留原中断向量的入口地址,并在我们的程序中应有把控制交给原中断的处理。原中断入口地址的获得可通过调用 DOS 中断 21H 的功能 35H,新中断处理程序的设置可通过调用 DOS 中断 21H 的功能 25H,驻留内存处理可通过调用 DOS 中断 21H 的功能 31H。这时,DX 的值为驻留内存的长度,把有关控制交回原中断处理的功能在我们程序中通过 JMP DWORD PTR CS:INTOSEG 命令来实现。

在下面程序中,我们的程序是以中断 80H 的方式驻留内存,当驻留内存时,其段地址 SEGMENT ADDRESS 和偏移量 OFFSET ADDRESS 可从内存单元 20:2-20:3 和 20:0-20:1 得到,为了不与把控制交回原中断处理的功能所占有的内存单元冲突,我们的桥地址 2 的内存空间设置应为:

OFFSET ADDRESS + 40H ~ OFFSET ADDRESS + 14H

这样,可以最大限度保证对此地址的读取程序的独立性和兼容性。

下面是桥地址 2 的设立源程序 BRIDGE2 · ASM
CSEG SEGMENT PARA PUBLIC, CODE'

ORG 100H

ASSUME CS: CSEG, DS: CSEG, ES: CSEG, SS: CSEG

INIT PROC NEAR

MOV AH, 35H

MOV AL, 80H

INT 21H

MOV CS:INTOSEG, BX

MOV CS:INTOOFF, ES

MOV DX, SEG NEWINT

MOV DS, DX

MOV DX, OFFSET NEWINT

MOV AX, 2580H

INT 21H

MOV AX, 3100H

MOV DX, 640H

INT 21H


```

INIT ENDP
NEWINT PROC FAR
JMP NEW80H
INTOSEG DW 9090H
INTOOFF DW 9090H
NEW80H:JMP DWORD PTR CS:INTOSEG
NEWINT ENDP
CSEG ENDS
END INIT

```

读者只要把上面源程序进行汇编和连接,生成 BRIDGE2·EXE,并在 DOS 状态下执行之,即在内存单元划分了一个区域,作为桥地址区域。

六、桥地址 3 的确立 桥地址 3 是作为 dBASE (FOXBASE)的内部二进制模块存在于内存,故与桥地址 2 的设立不同在于不用 DOS 中断 21H 的功能 31H 来驻留内存。另外,要在程序中设立一缓冲区用于桥地址空间。

桥地址 3 的确立源程序如下

```

CSEG SEGMENT PARA PUBLIC 'CODE'
ORG 100H
ASSUME CS:CSEG,DS:CSEG,ES:CSEG,SS:CSEG
INIT PROC NEAR
MOV AH,35H
MOV AL,80H
INT 21H

```

```

MOV CS:INTOSEG,BX
MOV CS:INTOOFF,ES
MOV AX,20H
MOV ES,AX
MOV DI,0
MOV AX,OFFSET NOOPER
STOSW
MOV AX,CS
STOSW
RET
INIT ENDP
NEWINT PROC FAR
NOOPER DW 9090H
JMP DWORD PTR CS:INTOSEG
INTOSEG DW 0
INTOOFF DW 0
BUFFER DW 200 DUP(0)
NEWINT ENDP
CSEG ENDS
END INIT

```

在本文中,我们已经提出了解决 dBASE (FOXBASE)与高级语言直接参数传递的思想——桥地址思想,并给出了不同桥地址设立的方法,以后的文章将分别介绍 dbase 与高级语言的直接参数的方法,为了说明问题,我们只讨论传递参数——字符串字符个数不多于 32 个的情形。

dBASE III PLUS(FOXBASE)对桥地址的存取

中国科学院计算中心 王敏生

dBASE III PLUS(FOXBASE)与 dBASE III 的内部管理不同,这决定了 dBASE III PLUS(FOXBASE)与 dBASE III 对桥地址的存取方式也大为不同。

考虑到 dBASE,FOXBASE,C,PASCAL,BASIC,FORTRAN 等语言有数值型数据转化为字符型数值数据的功能,故我们只考虑 dBASE(FOXBASE)与高级语言直接字符串传递的实现。

dBASE(FOXBASE)的字符串变量是以 NULL(即 ASCII 码值 0)为结束符,为了充分照顾到各种高级语言字符串存放形式的差别,我们约定:在桥地址

中,第一个字节存放所要传递字符串的长度(以十六进制表示),以后顺次存放其字符串内容,最后存放字符串结束符 NULL(即 ASCII 码值 0)。另外,为了简单起见,我们设所要传递的字符串长度不超过 32 位。

一、dBASE III PLUS(FOXBASE)的内存模块管理
dBASE III PLUS(FOXBASE)提供了 LOAD 命令和 CALL 命令用来加载和运行二进制文件,这是 dBASE III PLUS(FOXBASE)对桥地址进行存取的唯一途径。

1. DBASE III PLUS(FOXBASE)内存模块管理命令说明:

LOAD(二进制文件名)[.(扩展名)]

LOAD 将二进制文件加载到内存,这时 DBASE 将加载到内存的文件视为子程序或内部模块,而不是作为外部程序。

CALL(模块名)WITH(内存变量)

CALL 命令将直接从内存运行由 LOAD 装入内存的二进制命令。

RELEASE MODULE(模块名)

RELEASE MODULE 从内存中删除一装载的模块,释放其占有的空间以做它用。

2. DBASE Ⅱ PLUS (FOXBASE) 内存模块设计须知:

dBASE 的二进制文件要正确运行,必须遵守下列规范

- 每个文件最大为 32000 字节
- 第一个可执行指令必须放在(ORG)偏移量为 0 处
- 程序存放或使用的空间不能超过内存的实际容量,因为 LOAD 用文件的大小来确定需要分配的内存容量。
- 在返回控制 DBASE 以前,程序必须恢复代码段 CS 寄存器和栈段 SS 寄存器的内容。
- 用远程返回 RET FAR,以便把控制交回 dBASE。

如何由汇编语言的文件生成二进制文件,按照下列步骤执行即可:

- 对程序进行汇编,生成目标文件(.OBJ)
- MASM(文件名)(文件名) NULL NULL;
- 连接 OBJ 文件,形成可执行文件(.exe);
- LINK(文件名) NULL NULL;
- 从可执行文件生成二进制文件(.BIN)
- EXE2BIN(文件名)

3. dBASE 内存变量管理

在 dBASE 中,当使用 CALL 命令中,CALL(模块名)WITH(特定内存变量),dBASE 把代码段和代码指针指向被调用模块的开端处,而数据段(DS)存放特定内存变量的段地址,BX 寄存器存放特定内存变量第一字节的偏移量,如果我们假设变量结束符存放地址为 ADDRESS,则从 DS:[BX]至 ADDRESS 是特定内存变量在内存中存放位置(以增地址形式存放)。

了解了 DBASE 内存模块中内存变量管理规则,我们可以编制二进制模块,把 DS:[BX]至 ADDRESS 的内存单元内容传送到桥地址单元,这就实现了 dBASE 对桥地址单元的写操作。另外,编制另一个模

块,把桥地址单元内容传送到 DS:[BX]及以后的内存单元,从而改变特定内存变量的内容,实现了 dBASE 对桥地址单元的读操作。不过,这时传递的参数长度不得大于特定内存变量的长度,传递完毕时,应注意在后面的单元赋予传递结束符 NULL。

二、dBASE (FOXBASE)对桥地址的写操作
dBASE (FOXBASE)对桥地址的写操作的实现实际上就是把 DS:[BX]的数据传递到桥地址单元的编程。注意到所要传递的数据是以 NULL 为结尾,另外,桥地址第一个单元是存放所要传递字符串的长度,则我们可以编写出下列程序:

1. dBASE (FOXBASE)对桥地址 1 写操作程序 PLUSWRIT. ASM

```
ABCD SEGMENT
ASSUME CS,ABCD
MOVEB PROC FAR
;
; MOV DS,[BX]→0020:[0]
;
PUSH AX
PUSH BX
PUSH CX
PUSH DX
PUSH ES
PUSH DS
PUSH SI
PUSH DI
MOV AX,20H
MOV ES,AX
MOV AX,1
MOV DI,AX
MOV SI,BX
MOV BX,0
MOV CX,20H
LL:LODSB
CMP AL,0
JZ PP
STOSB
INC BX
LOOP LL
PP:MOV DI,0
MOVAL,BL
STOSB
POP DI
```



```

POP SI
POP DS
POP ES
POP DX
POP CX
POP BX
POP AX
RET
MOVEB ENDP
ABCD ENDS
END

```

假设所要传递的参数为字符变量 AA, 则执行命令:

```

LOAD PLUSWRIT
CALL PLUSWRIT WITH AA

```

实现了 dBASE (FOXBASE) 对桥地址 1 的写操作。

2. dBASE (FOXBASE) 对桥地址 2 或 3 写操作程序 PLUSW. ASM

```

ABCD SEGMENT
ASSUME CS:ABCD
MOVEB PROC FAR

```

```

;
; PLUSW. ASM
;
; MOV DS:[BX]→BRIDGE ADDRESS
;

```

```

PUSH AX
PUSH BX
PUSH CX
PUSH DX
PUSH ES
PUSH DS
PUSH SI
PUSH DI

```

PUSH BX ; SAVE DBASE PARAMETER ADDRESS

DRESS

```

MOV AX, 20H
MOV ES, AX
MOV AX, 0

```

MOV DI, AX ; POINT TO INT 80H ENTRY

ADDRESS

MOV AX, ES:[DI]

ADD AX, 40H ; GET OFFSET BRIDGE ADDRESS

DRESS

INC DI

INC DI

MOV BX, ES:[DI]; GET SEGMENT BRIDGE

ADDRESS ADDRESS

MOV ES, BX

MOV DI, AX ; POINT TO BRIDGE ADDRESS

POP BX ; RESTORE DBASE PARAMETER

ADDRESS

PUSH AX

INC DI ; POINT SECOND BYTE IN BRIDGE

ADDR

MOV BX, 0 ; CHAR STRING COUNT INIT

VALUE

MOV CX, 20H ; CHAR STRING MAX COUNT

VALUE

LL; LODSB

CMP AL, 0

STOSB

JZ PP

INC BX

LOOP LL

PP; POP AX

MOV DI, AX

MOV AL, BL ; MOVE LENGTH OF CHAR

STRING TO

STOSB ; FIRST BYTE IN BRIDGE ADDRESS

POP DI

POP SI

POP DS

POP ES

POP DX

POP CX

POP BX

POP AX

RET

MOVEB ENDP

ABCD ENDS

END

假设前面已在内存地址单元架起桥地址 2 或 3,

执行命令:

LOAD PLUSW

CALL PLUSW WITH AA

将把字符型变量往桥地址 2 或 3 上写!

三、dBASE(FOXBASE)对桥地址的读操作 把桥地址单元的数据送往 DS:[BX]就是 dBASE(FOXBASE)对桥地址单元的读操作,编程中必须注意到,桥地址单元的第二个单元及以上的内容才是高级语言所要传递的字符参数的实际内容。

1. dBASE(FOXBASE)对桥地址 1 读操作程序 PLUSREAD. ASM

```
ABCD SEGMENT
ASSUME CS,ABCD
MOVEB PROC FAR
;
; MOV 0020:[1]→DS:[BX]
;
PUSH AX
PUSH BX
PUSH CX
PUSH DX
PUSH ES
PUSH DS
PUSH SI
PUSH DI
MOV AX,DS
MOV ES,AX
MOV DI,BX
MOV AX,20H
MOV DS,AX
MOV AX,0
MOV SI,AX
MOV AH,0
LODSB
CMP AL,0
JZ NOCH
MOV CX,AX
LL:LODSB
STOSB
LOOP LL
MOV AL,0
NOCH:STOSB
PP:POP DI
POP SI
POP DS
POP ES
POP DX
POP CX
```

POP BX

POP AX

RET

MOVEB ENDP

ABCD ENDS

END

执行下列命令

STOR SPACE(32)TO AA

LOAD PLUSREAD

CALL PLUSREAD WITH AA

这时,AA 实际内容就是桥地址 1 所存放的字符串内容的拷贝。

2. dBASE(FOXBASE)对桥地址 2 或 3 读操作程序 PLUSR. ASM

```
ABCD SEGMENT
ASSUME CS,ABCD
MOVEB PROC FAR
;
; PLUSR·ASM
;
; MOV CHARS IN BRIDGE ADDRESS→DS;
[BX]
;
PUSH AX
PUSH BX
PUSH CX
PUSH DX
PUSH ES
PUSH DS
PUSH SI
PUSH DI
MOV AX,DS;POINT TO DBASE PARAMETER SEG ADDRESS
MOV ES,AX
MOV DI,BX;POINT TO DBASE PARAMETER OFFSET ADDR
MOV AX,20H
MOV DS,AX
MOV DI,0;POINT TO INT 80H ENTRY ADDRESS
STOSW
ADD AX,40H
MOV BX,AX;BX=OFFSET BRIDGE ADDRESS
```



```

STOSW ;AX =SEGMENT BRIDGE ADDRESS
MOV DS ,AX
MOV DI,BX ;POINT TO BRIDGE ADDRESS
INC DI ;POINT SECOND BYTE IN BRIDGE
ADDR
MOV CX,20H ;CHAR STRING MAX COUNT
VALUE
LL;LODSB
CMP AL,0
STOSB
JZ PP
LOOP LL
MOV AL,0
STOSB
PP;POP DI
POP SI

```

```

POP DS
POP ES
POP DX
POP CX
POP BX
POP AX
RET
MOVEB ENDP
ABCD ENDS
END
执行下列命令:
STOR SPACE(32)TO AA
LOAD PLUSR
CALL PLUSR WITH AA

```

这时,我们把已架起桥地址 2 或 3 后桥地址所存放的字符串内容直接赋给字符变量 AA。

dBASE III 对桥地址的存取

中国科学院计算中心 王敏生

dBASE III 与 dBASE III PLUS (FOXBASE) 的内部管理不同,与外部系统的接口是通过命令 RUN 来实现的,因而 dBASE III 与高级语言的直接参数传递的方法独具特色。

一、dBASE III 的内部命令简介 Ashton-Tate 公司的 dBASE III 软件,其内部提供两条命令一个函数,用以实现对内存单元的存取,这些命令(函数)不对外公开,因而在升级版本 dBASE III PLUS 中被取消了,解决 dBASE III 对桥地址存取问题,必须借助于这些命令与函数。

1. 段地址的设置:SET SEGMENT TO 定义存储区的当前段,命令 POKE 与函数 PEEK 将是在这个当前段的基础上对段偏移量的内存单元进行的读写操作。

命令格式 SET SEGMENT TO ADDRESS

注释:ADDRESS 可以是常数值,数值变量或数值表达式,当 ADDRESS 是字符数值变量时,前面必须加宏替换函数 & 使之成为数值,提醒读者注意的是,ADDRESS 的取值应在 0~65535 整数范围内,而且只能用十进制表示。

下面的命令都是合法的:

- SET SEGMENT TO 16
- SET SEGMENT TO 2 * 32
- STOR 50 TO AA
- SET SEGMENT TO AA
- SET SEGMENT TO AA+5
- STOR '50' TO AA
- SET SEGMENT TO &AA

下列命令是非法的:

- SET SEGMENT TO 0×12 (不是十进制)
- SET SEGMENT TO 66000 (大于 65535)
- SET SEGMENT TO '1234' (不是数值常数)

2. 从指定内存单元阅读字节函数 PEEK(N)

函数格式:V=PEEK(N)

注释:N 是在 0~65535 范围内的整数,N 如上面的 SET SEGMENT 所定义的那样,是从当前段开始的一个偏移量,它指出要阅读的内存地址。取回的值是一个 0 到 255 范围内的整数。PEEK 是 POKE 的反函数。为保证读操作的正确进行,N 应该是常数值,数值变量或是数值表达式。

下面命令把内存单元 0010:000A 的内容赋予变量 AA:

- SET SEGMENT TO 16
- AA = PEEK(10)

3. 向内存单元写字节命令 POKE

命令格式: POKE N, M

注释: N 必须是在 0 到 65535 范围内变化, 它指出将要写入的内存单元地址, 它是在 SET SEGMENT 命令中所定义的从当前段开始的偏移量。

M 是将写到指定存储单元的数据, 其值必须在 0 到 255 的范围内变化。

N, M 只能是常数或是字符型数值变量, 当是字符型数值变量, 在引用前要加 & 宏替换函数符号。

例: 下面命令都是合法的:

- STOR '10' TO AA
- POKE &AA, 81 或
- POKE 10, &AA

但当 AA=10, 则 POKE 11, AA 将把偏移量为 000B 的内存单元清零。

二、dBASE III 对桥地址单元的写操作 在下面的程序中, 我们把去掉右置空格的字符串的长度置于桥地址首单元, 然后把字符串内容顺次写到桥地址以后单元, 最后在桥地址单元上写上字符结束符 NULL

1. dBASE III 对桥地址 1 的写操作程序

```

** DB3WRITE. PRG
* * DBASE III — WRITE DBASE CHAR
STRING TO BRIDGE ADDRESS
SET TALK OFF
CLEA
STOR SPACE(32) TO CHOICE
@4,30 SAY 'DBASE III STATE I'
@6, 30 SAY ' INPUT CHAR STRING
PLEASE!'
@8,28 GET CHOICE
READ
CHOICE = TRIM(CHOICE)
LENGTH = LEN(CHOICE)
PP = '0'
SET SEGMENT TO 32
IF LENGTH = 0
POKE 0,0
RETURN
ENDIF
IF LENGTH < 10

```

```

LENGTHCH = STR(LENGTH,1,0)
ELSE
LENGTHCH = STR(LENGTH,2,0)
ENDIF

```

POKE 0, &LENGTHCH

II = 1

IIC = '1'

DO WHILE II < (LENGTH + 1

CHAR = SUBSTR(CHOICE, II, 1)

ASCCHAR = ASC(CHAR)

DO CASE

CASE ASCCHAR < 10

CHASC = STR(ASCCHAR, 1, 0)

CASE ASCCHAR > 99

CHASC = STR(ASCCHAR, 3, 0)

OTHERWISE

CHASC = STR(ASCCHAR, 2, 0)

ENDCASE

POKE &IIC, &CHASC

II = II + 1

IF II < 10

IIC = STR(II, 1, 0)

ELSE

IIC = STR(II, 2, 0)

ENDIF

ENDDO

CH = '0'

POKE &IIC, &CH

RETU

2. dBASE III 对桥地址 2 或 3 的写操作程序

dBASE III 对桥地址 2 或 3 的写操作, 首先必须从中断 80H 的入口地址, 读取桥地址 2 或 3 所在的段号及偏移量, 从这确定出桥地址 2 或 3 在内存准确位置, 在这个基础上实现 dBASE III 对桥地址 2 或 3 的写操作。

```

** DB3WRITE. PRG
* * DBASE III — WRITE DBASE CHAR
STRING TO BRIDGE ADDRESS
SET TALK OFF
CLEA
STOR SPACE(32) TO CHOICE
@4,30 SAY 'DBASE III STATE I'
@6, 30 SAY ' INPUT CHAR STRING
PLEASE!'

```



```

@8,28 GET CHOICE
READ
CHOICE =TRIM(CHOICE)
LENGTH=LEN(CHOICE)
PP='0'
SET SEGMENT TO 32
A0=PEEK(0)
A1=PEEK(1)
A2=PEEK(2)
A3=PEEK(3)
SEG =A3*256+A2
OFF=A1*256+A0+64
SET SEGMENT TO SEG
OFFC=STR(OFF,6,0)
OFFC1=STR(OFF+1,6,0)
IF LENGTH =0
POKE &OFFC,0
POKE &OFFC1,0
RETURN
ENDIF
IF LENGTH<10
LENGTHCH=STR(LENGTH,1,0)
ELSE
LENGTHCH=STR(LENGTH,2,0)
ENDIF
POKE &OFFC,&LENGTHCH
II=1
DO WHILE II<(LENGTH+1
CHAR =SUBSTR(CHOICE,II,1)
ASCCHAR=ASC(CHAR)
DO CASE
CASE ASCCHAR<10
CHASC =STR(ASCCHAR,1,0)
CASE ASCCHAR>99
CHASC=STR(ASCCHAR,3,0)
OTHERWISE
CHASC=STR(ASCCHAR,2,0)
ENDCASE
IIC=STR(OFF+II,6,0)
POKE &IIC,&CHASC
II=II+1
ENDDO
CH='0'
IIC=STR(OFF+II,6,0)

```

```

POKE &IIC,&CH
RETU

```

三、dBASE Ⅲ 对桥地址的读操作 在下面程序中,首先读取桥地址的首单元内容,这是传递到 dBASE 字符串的长度,然后以这个长度读取以后的字符给变量 AA。

1. dBASE Ⅲ 对桥地址 1 的读操作程序

```

** DB3READ. PRG
* * DBASE Ⅲ READ CHAR STRING FROM
BRIDGE ADDRESS
SET TALK OFF
CLEA
SET SEGMENT TO 32
W=PEEK(0)
IF W=0
RETURN
ENDIF
AA=''
II=1
DO WHILE II<(W+1
W1=PEEK(II)
AA=AA+CHR(W1)
II=II+1
ENDDO
RETU

```

2. dBASE Ⅲ 对桥地址 2 或 3 的读操作程序

```

** DB3READ. PRG
* * DBASE Ⅲ READ CHAR STRING FROM
BRIDGE ADDRESS
SET TALK OFF
CLEA
SET SEGMENT TO 32
A0=PEEK(0)
A1=PEEK(1)
A2=PEEK(2)
A3=PEEK(3)
SEG =A3*256+A2
OFF=A1*256+A0+64
SET SEGMENT TO SEG
W=PEEK(OFF)
IF W=0
RETURN
ENDIF
AA=''

```



```

II=1
DO WHILE II<W+1
W1=PEEK(II+OFF)
AA=AA+CHR(W1)
II=II+1
ENDDO
RETU

```

四、dBASE Ⅲ 通过桥地址实现参数传递 在上面的基础上,我们可以通过桥地址在 dBASE Ⅲ 之间传递字符串参数,用以检验上面对桥地址读写操作的正确性,下面是 dBASE Ⅲ 通过桥地址 1 传递参数的检验程序

***dBASE Ⅲ

```

** TRANSFER CHAR STRING BY BRIDGE1
ADDRESS
SET TALK OFF
PUBLIC AA ,CHOICE
DO DB3WRITE
STOR TRIM(CHOICE) TO CHI
DO DB3READ
IF CHI=AA
WAIT 'TRANSFER SUCCESS!'
ELSE
WAIT 'TRANSFER FAIL!'
ENDIF
RETURN

```

dBASE(FOXBASE)与 BASIC 语言的直接参数传递

中国科学院计算中心 王敏生

摘要 在这一篇文章中,我们将解决 BASIC 语言对桥地址存取的问题,重点介绍 dBASE(FOXBASE)与 BASIC 语言通过桥地址 1 实现参数直接传递,至于 dBASE Ⅲ 与 BASIC 语言的直接参数传递,dBASE(FOXBASE)与 BASIC 语言通过桥地址 2 或 3 实现参数直接传递可参考别的系列文章。

一、BASIC 语言直接内存存取语句 BASIC 语言有直接内存存取的语句和函数,这是 BASIC 语言对桥地址进行存取的基本手段。

1. 定义存储区当前“段”的语句:

命令格式:DEF SEG [=ADDRESS]

命令注释:该命令相当于 DBASE Ⅲ 的命令 SET SEGMENT TO,定义了命令 POKE 与函数 PEEK 把所有操作的实际物理地址做为进入这个段的偏移量。

ADDRESS 是一个 0 到 65535 范围的数值表达式,若 ADDRESS 是以十六进制表示的,前面应加符号 &H 加以说明。

下面的语句把当前存储区段地址设置为彩色/图形监视器接口板的屏幕缓冲区。

```
100 ' SET SEGMENT TO COLOR SCREEN
BUFFER
```

```
110 DEF SEG =&HB800
```

2. 从指定内存单元阅读字节的函数

函数格式:V=PEEK(N)

函数注释:N 是在 0 到 65535 范围内的整数,N 如 DEF SEG 语句所定义的那样,是从当前段开始的一个

偏移量,它指出将要阅读的内存单元地址,返回值将是一个 0 到 255 范围内的整数,PEEK 是 POKE 的反函数。

3. 向内存单元写数据语句

语句格式:POKE N,M

语句注释:N 是 0 到 65535 之间的整数,它指出将要写入的内存单元地址,N 是一个如 DEF SEG 语句所定义的从当前段开始的偏移量。M 是将写到存储单元的数据,其值必须在 0 到 255 范围内变化。POKE 的反函数是 PEEK,利用 PEEK 和 POKE,可以实现高效率地在字符串变量和桥地址单元互相传递数据。

二、BASIC 语言对桥地址的写操作 BASIC 语言对桥地址的读写操作的思想基本与 dBASE Ⅲ 对桥地址读写操作相同,处理不同之处在于 BASIC 语言本身特点,如 BASIC 语言的字符串变量为变长,至多可至 254 字符,故在程序中,必须有截取前 32 个字符的处理。

1. BASIC 语言对桥地址 1 的写操作

首先通过 DEF SEG 语句把当前段定义为 20H,然后利用 POKE 实现对桥地址 1 的写操作,具体程序

如下:

```
1 REM——WRITE.BAS
2 REM——BASICA CHAR STRING WRITE
TO BRIDGE ADDRESS
10 PRINT "BASICA STATE! INPUT STRING
PLEASE"
15 INPUT ABC$:ABC$=LEFT$(ABC$,
32)
```

```
20 A1=LEN(ABC$)
30 DEF SEG=&H20
80 POKE 0,A1
90 IF A1=0 THEN END
100 A2=0
110 A2=A2+1
120 A0$=MID$(ABC$,A2,1)
130 A0=ASC(A0$)
140 POKE A2,A0
150 IF A2=A1 THEN GOTO 190
160 GOTO 110
190 A2=A2+1:POKE A2,0:END
```

2. BASIC 语言对桥地址 2 或 3 的写操作

同样,我们通过 DEF SEG 语句设置当前段为 20H,然后通过 PEEK 函数求出桥地址 2 或 3 的段号及偏移量,然后通过 POKE 命令实现对桥地址 2 或 3 的写操作。

```
1 REM——WRITE.BAS
2 REM——BASICA CHAR STRING WRITE
TO BRIDGE ADDRESS
10 PRINT "BASICA STATE! INPUT STRING
PLEASE"
15 INPUT ABC$:ABC$=LEFT$(ABC$,
32)
20 A1=LEN(ABC$)
30 DEF SEG=&H20
40 B0=PEEK(0):B1=PEEK(1):B2=PEEK
(2):B3=PEEK(3)
50 SEGADD=B3*256+B2:OFFADD=B1*
256+B0+64:I=OFFADD
60 DEF SEG=SEGADD
80 POKE I,A1
90 IF A1=0 THEN 190
100 A2=0
110 A2=A2+1:I=I+1
120 A0$=MID$(ABC$,A2,1)
```

```
130 A0=ASC(A0$)
140 POKE I,A0
150 IF A2=A1 THEN GOTO 190
160 GOTO 110
190 I=I+1:POKE I,0:END
```

三、BASIC 语言对桥地址的读操作 参照 dBASE

■ 对桥地址读操作的处理,我们可以较容易编写出 BASIC 语言对桥地址读操作的程序。

1. BASIC 语言对桥地址 1 的读操作

```
1 REM——READ.BAS
2. REM——BASICA READ BRIDGE ADDRESS
10 DEF SEG=&H20
15 A0=PEEK(0)
20 IF A0=0 THEN 40
30 PRINT "BASICA STATE! NO CHAR IN-
PUT IN DBASE STATE!"
35 END
40 A1=PEEK(1)
50 ABC$=CHR$(A1)
60 A1=1
70 IF A1=A0 THEN 180
90 A1=A1+1
100 A2=PEEK(A1)
110 A0C$=CHR$(A2)
120 ABC$=ABC$+A0C$
130 IF A1=A0 THEN 180
140 GOTO 70
180 PRINT "BASICA STATE! THE STRING
IN DBASE STATE IS "ABC$
190 END
```

2. BASIC 语言对桥地址 2 或 3 的读操作

```
1 REM———READ:BAS
2 REM———BASICA READ BRIDGE AD-
DRESS
10 DEF SEG=&H20
15 A0=PEEK(0):A1=PEEK(1):A2=PEEK
(2):A3=PEEK(3)
20 SEGADD=A3*256+A2:OFFADD=A1*
256+A0+64:DEF SEG=SEGADD
25 I=OFFADD:A0=PEEK(I):IF A0>0
THEN 40
30 PRINT "BASICA STATE!NO CHAR INPUT
IN DBASE STATE!"
35 END
```



```

40 I=I+1 : A1=PEEK(I)
50 ABC$=CHR$(A1)
60 A1=1
70 IF A1=A0 THEN 180
90 A1=A1+1 : I=I+1
100 A2=PEEK(I)
120 A2C$=CHR$(A2)
130 ABC$=ABC$+A2C$
140 A1=A1+1 : I=I+1
150 IF A1>A0 THEN GOTO 180
160 GOTO 100
180 I=I+1:POKE I,0
190 PRINT"BASICA STATE! THE STRING IN
DBASE STATE IS "ABC$
200 END

```

四、DBASE (FOXBASE) 与 BASIC 语言的直接参数传递 1. BASIC 语言通过桥地址 1 传递参数的方法: 在上面的基础上, BASIC 语言可通过桥地址来传递参数, 以检验 BASIC 语言对桥地址读写操作的正确性, 下面是其检验方法, 在 BASIC 状态下:

- A. 装载 BASIC 语言对桥地址 1 写操作程序
LOAD WRITE
- B. 运行之, 完成把字符串往桥地址 1 的写操作:
RUN
- C. 装载 BASIC 语言对桥地址 A 读操作程序
LOAD READ
- D. 运行之, 完成把字符串往桥地址 1 的读操作:
RUN

在执行 D 后, 通过比较输入与输出字符串, 基本可以确定 BASIC 语言对桥地址读写的正确性。

2. DBASE (FOXBASE) 与 BASIC 语言直接参数传递:

DBASE (FOXBASE) 与 BASIC 语言直接参数传递通过下面的程序来完成。我们在长城 286B 机上运行的 BASIC 版本是随机带来的软件 NBASIC, 在实际应用中读者要根据不同微机上不同的 BASIC 语言版本作些改动即可。

```

** DBASE ■ PLUS STATE
** TRANSFER CHAR STRING
** BETWEEN DBASE AND BASIC
** BY BRIDGE ADDRESS 1
LOAD A:PLUSREAD
LOAD A:PLUSWRITE
SET TALK OFF

```

```

DO WHILE .T.
AA=SPACE(32)
CLEA
STOR'' TO CHOICE1, CHOICE2
@1,10 SAY'*****'
@2,10 SAY' * DIRECT PARAMETER TRANSFER
*'
@3,10 SAY' * DBASE ■ PLUS (---)BASIC * '
@4,10 SAY'*****'
@5,10 SAY' * 1. BASICA --->DBASE ■ PLUS * '
@6,10 SAY' * 2. DBASE ■ PLUS --->BASICA * '
@7,10 SAY' * 3. QUIT
@8,10 SAY'*****'
*'
@10,10 SAY' * * SELECT 1---3 PLEASE !
* * ' GET CHOICE1 CLEA TYPEAHEAD
READ
DO CASE
CASE CHOICE1='1'
RUN \NBASIC WRITE
CALL PLUSREAD WITH AA
AA=TRIM(AA)
AAL=LEN(AA)
STOR'' TO CHOICE2
IF AAL=0
@15,20 SAY'DBASE ■ PLUS STATE !'
@17,20 SAY'NO CHAR INPUT IN BASICA '
GET CHOICE2
READ
LOOP
ELSE
@15,20 SAY'DBASE ■ PLUS STATE !'
@17,20 SAY'THE CHAR STRING IN BASICA
IS'
@19,22 GET AA
CLEA GETS
@20,10 SAY SPACE (20) GET CHOICE2
READ
LOOP
ENDIF
CASE CHOICE1='2'
@15,20 SAY'DBASE ■ PLUS STATE !'
@17,20 SAY' INPUT CHAR STRING
PLEASE'

```



```
@19,20 GET AA
READ
AA=TRIM(AA)
CALL PLUSWRIT WITH AA
RUN \NBASICA READ
```

```
CASE CHOICE1='3'
RETU
ENDCASE
ENDDO
```

dBASE(FOXBASE)与 C 语言的直接参数传递

中国科学院计算中心 王敏生

摘要在这一篇文章中,我们将解决 C 语言对桥地址存取的问题。重点介绍 dBASE(FOXBASE)与 C 语言通过桥地址 2 或 3 实现参数直接传递,至于 dBASE(FOXBASE)通过桥地址 1 实现参数直接传递 DBASE II 与 C 语言直接参数传递请参考别的系列文章。

一、C 语言对内存单元直接存取的手段 C 语言与别的程序设计语言不同之处是具有处理指针的能力,及其在处理指针时所表现出独有的灵活性,使得 DBASE(FOXBASE)与 C 语言之间的直接参数传递比起与 BASIC, FORTRAN, COBOL 语言的直接参数传递要容易些。

C 语言直接读写桥地址要求使用 FAR 指针,如果 C 编译器不支持 FAR 指针,则不能对桥地址直接访问。C 编译用两种方式支持 FAR 指针,一种是常用的 FAR 关键字扩展,另一种是使用大存储模式编译器,其所有指针值为 FAR,在本文中,我们使用 FAR 改进型。另外,本文所有 C 语言程序都是用 TURBO C 1.5 以上版本编译,当然,你可以用 MICROSOFT 或其它公司的 C 编译软件编译。

C 语言的内存地址用十六进位 8 位整数来表示,前 4 位表示段地址,后四位表示偏移量,如十六进制表示某内存某个单元 20:11,则在 C 语言用 0X00200011(0X 表示十六进制)来表示。

如果 V 是一指向某个内存单元 MEM-BYTE 的字符型指针, C 为一字符变量,则

```
*V=C
```

表示把字符 C 的内容送往内存单元 MEM-BYTE,

```
而 C=*V
```

则表示把内存单元 MEM-BYTE 的内容赋给字符变量。

二、C 语言对桥地址的写操作 C 语言的字符串存放方式与 dBASE(FOXBASE)的存放方式相同,另外 C 语言的字符串也是变长的,其最大长度不能超过预先在程序定义好的长度,依照这些特点,我们可以编写出 C 语言对桥地址写操作的程序。

1. C 语言对桥地址 1 的写操作

```
/* write.c——c direct write bridge address */
#include<stdio.h>
#include<process.h>
#define bridge-address 0×200
main()
{
    int i,l;
    unsigned char ch[32],ic;
    char far *v, far *address-point;
    printf("\n c state! \n");
    printf("input char string please! \n");
    scanf("%s",ch);ch[32]='\0';
    i=strlen(ch);
    ic=(char)i;
    address-point=(char far *)bridge-address;
    v=address-point;
    *V++=ic; /* SAVE LENGTH OF CHAR
STRING */
    for(i=0; i<ch[i]=='\0'; i++) *V++=ch[i];
    *v='\0';
```


}

2. C 语言对桥地址 2 或 3 的写操作

/* Write.c——c direct write bridge address */

#include<stdio. h>

#include<process. h>

#define off—address 0×200

#define seg—address 0×202

main()

{

long int bridge—address;

unsigned short int i, segint, offint;

unsigned char ch[32], ic;

unsigned short int *seg, *off;

long int address;

unsigned char far *v, far *address—point;

printf("\n * * * * STATE! * * * *\n");

printf("INPUT CHAR STRING PLEASE ! \n");

scanf("%s", ch); ch[31]='\\0';

i=strlen(ch);

ic=(char)i;

seg=(unsigned short int *)seg—address;

off=(unsigned short int *)off—address;

segint=*seg; offint=*off;

(long int)offint+0×40

address—point=(unsigned char far *)bridge—ad—

dress;

v=address—point;

* V++ = ic; /* SAVE LENGTH OF CHAR

STRING */

for (i=0; ! ch[i]=='\\0'; i++) * V++=ch[i];

* v='\\0';

}

三、C 语言对桥地址的读操作 C 语言对桥地址的读操作是先读取桥地址首单元是否等于 0, 如等于 0, 表示没字符传送, 否则, 则顺次读取以后单元的内容, 一直读取到 NULL 为止, 这个过程其实就是写操作的反操作。

1. C 语言对桥地址 1 的读操作

/* read.c——c direct read bridge address */

#include<stdio. h>

#include<process. h>

#define bridge—address 0×201

main()

{

int i;

unsigned char ch[32];

char far *v, far *address—point;

address—point=(char far *)bridge—address;

v=address—point;

ch[0]=*V++;

if(ch[0]=='\\0')

}

printf("\n C STATE! NO CHAR INPUT IN DBASE STATE! \n");

return

}

for(i=1; ! *v=='\\0'; i++)

{

if (i==32)break;

ch[i]=*V++;

}

ch[i]='\\0'; /* THE CHAR STRING END CHARACTER */

if (i==32) ch[i]='\\0';

printf("\n C STATE! THE STRING IN DBASE IS\n");

printf(" * * %s * ", ch);

return

}

2. C 语言对桥地址 2 或 3 的读操作

把上面的程序稍加改动, 首先从中断 80H 的入口地址求出桥地址 2 或 3 的准确位置, 然后让字符指针指向桥地址, 其它与对桥地址 1 的读操作相同, 即可实现对桥地址 2 或 3 的读操作。

/* readc.c——c direct read bridge address */

#include<stdio. h>

#include<process. h>

#define seg—address 0×202

#define off—address 0×200

main()

{

long int bridge—address, c;

unsigned short int i, segint, offint;

unsigned char ch[32];

unsigned short int *seg, *off,

unsigned char far *v, far *addres—point;

seg=(unsigned short int *)seg—address;


```

off=(unsigned short int *)off-address;
segint=*seg;offint=*off;
bridge-address=(long int)segint*0×10000+
(long int)offint+0×41;
address-point=(unsigned char far *)bridge-
address;
v=address-point;
ch[0]=*v++;
if(ch[0]!='\0')
{
printf("\n C STATE ! NO CHAR INPUT IN
DBASE STATE ! \n");
return;
}
for (i=1; ! *v=='\0'; i++)
{
if(i==31)break;
ch(i)=*v++;
}
ch[i]='\0';/* THE CHAR STRING END
CHARACTER */
if (i==31) ch[i]='\0';
printf("\n C STATE! THE STRING IN DBASE
IS\n");
printf(" * * %s * * ",ch);
return;
}

```

四、dBASE(FOXBASE)与 C 语言直接参数传递

把上面几个源程序进行编译,连接,生成 EXE 文件,在这个基础上,可以完成 dBASE(FOXBASE)与 C 语言的直接参数传递:

1. C 语言通过桥地址实现参数传递

在 DOS 状态下,顺次执行下列命令,可以完成 C 语言通过桥地址实现参数的传递。

C>WRITE✓

C>READ✓

2. 桥地址 2 的建立:

桥地址 2 的建立,既可以在 DOS 状态下建立:

C>BRIDGE2✓

也可在 dBASE 状态下建立,

RUN BRIDGE3✓

桥地址 2 一经建立,如果重复建立的话,则原先建立的桥成为只占内存空间,不能发挥作用的废桥。

3. 桥地址 3 的建立

桥地址 3 的建立,须在 DBASE 状态下,通过运行下列命令来建立,

LOAD BRIDGE3✓

CALL BRIDGE3✓

上述命令可以重复运行,对内存空间没有影响,只是影响运行效率而已。

另外,RELEASE MODULE BRIDGE3 命令释放桥地址 3 所占有的内存空间以作它用。

4. dBASE(FOXBASE)与 C 语言直接参数传递

下面是 C 语言向 dBASE(FOXBASE)借助于桥地址 3 传递参数的程序,至于 DBASE(FOXBASE)向 C 语言的参数传递可以参考本程序与别的系列文章:

* 建立桥地址 3

LOAD BRIDGE3

CALL BRIDGE3

* C 语言向桥地址 3 写参数

RUN WRITE

STOR SPACE(32) TO AA

* dBASE(FOXBASE) 向桥地址 3 读参数

LOAD PLUSR

CALL PLUSR WITH AA

则字符变量 AA 的内容就是 C 语言所要传递的字符参数。

dBASE (FOXBASE)与 PASCAL 语言直接参数传递

中国科学院计算中心 王敏生

摘要 在此文中,我们将解决 PASCAL 语言对桥地址存取的问题,重点介绍 dBASE III 与 PASCAL 语言直接参数传递,至于 dBASE(FOXBASE)与 PASCAL 语言的直接参数传递可参考别的系列文章。

PASCAL 语言对内存直接存取操作可以通过指针,直接内存存取数组,绝对地址定义和嵌套过程等手段来实现,读者可以很方便地利用这些手段来实现对桥地址的读写。

PASCAL 语言的字符串长度是变长的,但最大长度不能超过预先定义好的长度,PASCAL 语言的字符串占用其最大长度加一的存储空间,第一字节包含字符串的当前动态长度,后面的字节存储字符串的字符,这也是 PASCAL 语言的一个特色。

本文中所有程序都是在 Turbo PASCAL 4.0 以上版本编译连接的。

一、绝对地址变量与对桥地址的存取 PASCAL 语言有一类变量叫做绝对地址变量。说明到内存特定地址的变量叫绝对变量,绝对变量的说明是在类型后面加上一个 ABSOLUTE 子句。

语句: CRTMODE; BYTE ABSOLUTE \$0040; \$0049 定义了存储地址为十六进制的 40:49 的绝对变量 CRTMODE。

利用绝对变量的定义,可以快速实现 PASCAL 语言对桥地址的读写操作。

①绝对变量对桥地址 1 的写操作:

```
PROGRAM P3;
{MOVE PASCAL STRING TO dBASE BY ABSOLUTE VARIABLE}
{¥F+}
VAR
N;STRING[32];
N1;STRING[32] ABSOLUTE ¥0020;¥0;
BEGIN
    WRITELN(' THIS IS PASCAL STATE
!');
    WRITELN(' INPUT THE STRING
WILL BE TRANSFERED TO dBASE ');
    READ(N);
    MOVE(N,N1,33);
```

END.

②绝对变量对桥地址 1 的读操作

```
PROGRAM P4;
{MOVE DBASE STRING TO PASCAL BY ABSOLUTE VARIABLE}
{¥F+}
VAR
N;STRING[32];
```

```
N1;STRING[32]ABSOLUTE ¥0020;¥0;
BEGIN
    MOVE(N1,N,33)
    WRITELN ('THIS IS PASCAL
STATE!');
    WRITELN('THE STRING IN DBASE
IS:',N);
END.
```

用绝对变量实现对桥地址的读写,具有编程简单,传递效率高的特点。但由于 PASCAL 语言字符串的特色,字符串不是以 NULL 为结束符,故 PASCAL 语言与 dBASE(FOXBASE),C 等语言进行参数传递时,要注意在这些语言进行桥地址读的时候,一定要以桥地址首单元的内容为长度来读取以后的字符。

二、指针处理和嵌套过程对桥地址的读写操作 利用 TurboPASCAL 的指针处理可象 C 语言那样较方便的实现对桥地址的读写。

Turbo PASCAL 的函数 PTR 将把段地址及偏移量地址转换成指针类型。

语句: P:=PTR(\$0020;\$0000)将给指向内存地址为 \$0020;\$0000 的指针变量 P 赋值。

1. 指针处理实现对桥地址 1 的写操作

```
PROGRAM P5;
{MOVE PASCAL STRING TO dBASE BY POINTER}
TYPE
N1=STRING[32];
P1=^N1;
VAR
{¥F+}
P:P1;
N;STRING[32];
BEGIN
    WRITELN('THIS IS PASCAL STATE!');
    WRITELN('INPUT THE STRING
WILL BE TRANSFERED TO dBASE');
    READ(N);
    P:=PTR(¥20,¥0);
    MOVE(N,P^,33);
END.
```

另外,Turbo PASCAL 允许你使用 INLINE 指令来实现汇编指令过程嵌套进 PASCAL 语言的使用,这方面的内容较丰富,在这里我们就不作介绍。

2. 嵌套汇编过程对桥地址 1 的读操作。

PROGRAM P8;

```
{MOVE dBASE STRING TO PASCAL BY IN-
LINE}
{¥F+}
```

TYPE

```
STR1=STRING[32];
PROCEDURE PP(VAR NN;STR1);
BEGIN
  INLINE(
    ¥8B/¥46/<NN/{MOV AX,[BP+NN];AX=
(OFFSEG) ADDRESS OF NN }
    ¥89/¥C7/{MOV DI,AX}
    ¥1E/{PUSH DS}
    ¥56/{PUSH SI}
    ¥90/{NOP}
    ¥1E/{PUSH DS}
    ¥07/{POP ES}
    ¥B8/¥20/¥00/{MOV AX,0020}
    ¥8E/¥D8/{MOV DS,AX}
    ¥BE/¥00/¥00/{MOV SI,0}
    ¥B9/¥21/¥00/{MOV CX, 0021; CX =
COUNT OF NN}
```

```
¥F3/¥A4/{REPZ MOVSB}
¥B0/¥00/{MOV AL,0}
¥AA/{STOSB}
¥5E/{POP SI}
¥1F/{POP DS}
END;
VAR
  N;STR1;
BEGIN
  PP(N);
  WRIELN('THIS IS PASCAL STATE1');
  WRITELN('THE STRING IN dBASE IS1',N);
END.
```

三、直接内存存取数组对桥地址的读写操作

TurboPASCAL 实现预定义数组 MEM,利用这可以以字节为元素直接访问存储器单元。

语句:MEM[\$0040:\$0049]=7

将把数值 7 存于内存 \$0040:\$0049 字节内。

而语句:I:=MEM[\$0040:\$0049]

则把内存 \$0040:\$0049 的内容赋给变量 I

利用直接内存存取数组对桥地址 2 或 3 的存取程序如下:

1. 对桥地址 2 或 3 的写操作。

PROGRAM P1;

```
{MOVE PASCAL STRING TO dBASE BY DI-
RECT MEM}
{¥o+}
```

{¥F+}

VAR

```
N;STRING [32]
NSEG,NOFS,INTSEG,INTOFS;WORD;
I,NBYTE,I0,I1;BYTE;
BEGIN
  WRITELN('');
  WRITELN(' * * THIS IS PASCAL STATE!
* * ');
  WRITELN(' INPUT THE STRING WILL BE
TRANSFERED TO dBASE ');
  READ(N);
  NSEG:=SEG(N);
  NOFS:=OFS(N);
  INTSEG:=¥20;
  INTOFS:=¥0;
  I:=¥0;
  I0:=MEM[NSEG:NOFS];
  I1:=I0;
  WHILE I<=I0 DO
  BEGIN
    NBYTE:=MEM[NSEG:NOFS+I];
    MEM [INTSEG:INTOFS+I]:=NBYTE;
    I:=I+¥1;END;
    MEM[INTSEG:INTOFS+I1+1]:=0
  END.
```

2. 对桥地址 2 或 3 的读操作程序:

PROGRAM P2;

```
{MOVE dBASE STRING TO PASCAL BY DI-
RECT MEM}
```

{¥O+}

{¥f+}

VAR


```

N:STRING[32];
NSEG,NOFS,INTSEG,INTOFS:WORD;
I,NBYTE:BYTE;
BEGIN
NSEG:=SEG(N);
NOFS:=OFS(N);
INTSEG:=¥20;
INTOFS:=¥0;
I:=¥0;
WHILE I<=¥20 DO
BEGIN
NBYTE:=MEM[INTSEG:INTOFS];
MEM[NSEG:NOFS+I]:=NBYTE;
I:=I+¥1;
INTOFS:=INTOFS+¥1;
END;
MEM[NSEG:NOFS+¥21]:=0;
Writeln(' ');
Writeln(' * * THIS IS PASCAL STATE! *
* ');
Writeln('THE STRING IN dBASE IS :',N);
END.

```

四、dBASE III 与 PASCAL 语言的直接参数传递

把上述程序编译,连接成可执行文件,并适当改名,对桥地址 1 写操作的文件改名为 WRITE·EXE,读操作文件改名为 READ·EXE。

1. PASCAL 语言通过桥地址 1 传递参数执行下列命令:

```

C>WRITE✓
C>READ✓

```

将实现 PASCAL 语言通过桥地址 1 传递参数。至于 PASCAL 语言与 C 语言,BASIC 语言通过桥地址传递参数可参考上面。

2. dBASE III 与 PASCAL 语言通过桥地址直接传递参数:

下面的程序,实现了上述功能!

```

* * DB3MAIN. PRG———dBASE III STATE
* * TRANSFER CHAR STRING
* * BETWEEN dBASE III AND PASCAL
* * BY BRIDGE ADDRESS1
PUBLIC AA,CHOICE ,W
SET TALK OFF
STOR' ' TO S1

```

```

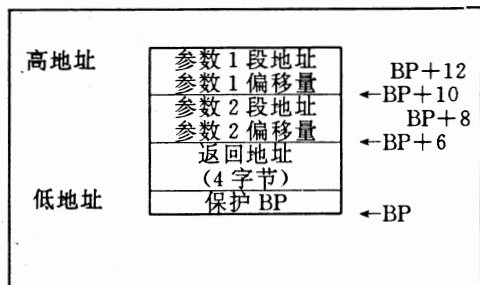
DO WHILE . T.
CLEA
STOR' ' TO CHOICE1
@1,10 SAY' * * * * *
@2,10 SAY' * DIRECT PARA TRANSFER
BETWEEN *
@3,10 SAY' * dBASE III ↔PASCAL *
@4,10 SAY' * * * * *
@5,10SAY' *1. PASCAL →dBASE III *
@6,10SAY *' 2. dBASE III →PASCAL
@7,10 SAY' 3. QUIT *
@8,10 SAY' * * * * *
@9,10 SAY' SELECT 1-3 PLEASE !' GET
CHOICE1 READ
DO CASE
CASE CHOICE1 ='1'
RUN A:WRITE
DO DB3READ
IF W =0
@10,0 CLEA
@10,10SAY 'NO CHAR TRANSFER 'GET S1
READ
STOR' ' TO S1
ELSE
@10,10SAY 'THE CHAR STRING IN PASCAL
IS !'
@10,45 GET AA
CLEA GETS
@10,78 GET S1
READ
STOR' ' TO S1
ENDIF
CASE CHOICE1 ='2'
DO DB3WRITE
RUN A:READ
STOR' ' TO S1
@10,78 GET S1
READ
CASE CHOICE1 ='3'
RETU
ENDCASE
ENDDO

```


dBASE(FOXBASE)与 FORTRAN 语言直接参数传递

中国科学院计算中心 王敏生

由于 FORTRAN 语言本身的特点, FORTRAN



语言没有直接内存读写的手段,故 FORTRAN 语言要通过调用汇编语言才能实现对桥地址读写操作。在这一篇文章中,我们将重点介绍 Fortran 语言如何实现对桥地址读写操作。

一、FORTRAN 语言调用汇编语言

FORTRAN 语言调用汇编子过程,要注意:

1. FORTRAN 语言调用子程序时,其堆栈结构变化如左图,(以两个参数为例说明):

这正好与汇编中的结构 STRUC 定义相吻合。

2. 为此,我们设立结构 STRUC 来描述,为要传递的参数分配内存空间,STRUC 定义如下:

```
FRAME STRUC
SAVEBP DW?,保留 BP 的值
SAVRET DD?,返回地址
ARGADD DD? 参数 1 的地址
FRAME ENDS
```

3. 汇编语言子程序的数据段同结构分配的空间要在同一段中,这可通过语句 DGROUP 来实现。

4. 汇编子程序的返回语句必须根据压入栈参数的总长度来实现返回,一般说来是用 RET 4 * N 返回,这里 N 为参数个数。

二、FORTRAN 语言对桥地址的写操作

FORTRAN 语言的字符串是定长的,在输入过程中,不够之处用空格填充,这是 FORTRAN 语言字符串的一个特点,另外,在汇编子程序中,因为传递参数的需要,数据段 DS 和附加段 ES 会发生变化,故在程序中必须予以保护,才能保证整个程序的正常运行。

1. 汇编子程序对桥地址 1 写操作程序 TOMEMO.ASM

```
;MOVE FORTRAN CHAR STRING TO MEMORY
```

```
; TOMEMO.ASM
```

```
FRAME STRUC
```

```
SAVEBP DW ?
```

```
SAVEDS DW ?
```

```
SAVEBP DW ?
```

```
SAVRET DD?
```

```
CHADDR DD?
```

```
FRAME ENDS
```

```
;
```

```
-DATA SEGMENT WORD PUBLIC 'DATA'
```

```
-DATA ENDS
```

```
CSEG SEGMENT 'CODE'
```

```
DGROUP GROUP -DATA
```

```
;
```

```
ASSUME CS ; CSEG, DS ; DGROUP, ES ;
DGROUP, SS ; DGROUP
```

```
PUBLIC TOMEMO
```

```
TOMEMO PROC FAR
```

```
PUSH ES
```

```
PUSH DS
```

```
PUSH BP
```

```
MOV BP, SP
```

```
LES AX, [BP] + CHADDR
```

```
MOV SI, AX
```

```
MOV AX, 20H
```

```
MOV ES, AX
```

```
MOV AX, 0
```

```
MOV DI, AX
```

```
MOV AX, 20H
```

```
MOV CX, AX
```

```
STOSB
```

```
AA: LODSB
```

```
STOSB
```



```

DEC CX
JNZ AA
MOV AL,0
STOSB
POP BP
POP DS
POP ES
RET 4*1
TOMEMO ENDP
CSEG ENDS
END

```

2. 汇编子程序对桥地址 2 或 3 写操作程序

```

TOMEMO. ASM
;MOVE FORTRAN CHAR STRING TO
;ENTRY ADDRESS OF MEMORY IS
;ENTRY ADDRESS OF INT 80H
;TOMEMO . ASM
FRAME STRUC
SAVDS DW?
SAVES DW?
SAVSI DW?
SAVDI DW?
SAVBP DW?
SAVRET DD?
CHADDR DD?
FRAME ENDS
—DATA SEGMENT WORD PUBLIC 'DATA'
—DATA ENDS
CSEG SEGMENT 'CODE'
DGROUP GROUP —DATA
ASSUME CS : CSEG, DS: DGROUP, ES:
DGROUP,SS:DGROUP
PUBLIC TOMLMO
TOMEMO PROG FAR
PUSH DS
PUSH ES
PUSH SI
PUSH DI
PUSH BP
MOV BP,SP
LDS AX,[BP]+CHADDR
MOV SI,AX
MOV AX,20H
MOV ES,AX

```

```

MOV AX,0
MOV DI,AX
MOV AX,ES:[DI]
ADD AX,40H
INC DI
INC DI
MOV BX,ES:[DI]
MOV ES,BX
MOV DI,AX
MOV AL,20H
STOSB
MOV CX,20H
AA:LODSB
STOSB
DEC CX
JNZ AA
MOV AL,0
STOSB
POP BP
POP DI
POP SI
POP ES
POP DS
RET 4*1
TOMEMO ENDP
CSEG ENDS
END

```

3. FORTRAN 语言对桥地址写操作的程序:

```

C TOMEMORY. FOR — MOVE CHAR IN
FORTRAN TO MEMORY
CHARACTER*32 CHADDR
WRITE(*,10)
WRITE(*,20)
10 FORMAT (1H0,'THIS IS FORTRAN
STATE!')
20 FORMAT (1H0,'INPUT CHAR STRING
PLEASE!')
READ*,CHADDR
CALL TOMEMO (CHADDR)
RETURN
END

```

在本程序中,如果在连接过程中,与之连接是 1 中的 TOMEMO,则实现了对桥地址 1 的写操作,如与之连接是 2 中的 TOMEMO,则实现了对桥地址 2 或 3

的写操作。

4. FORTRAN 语言与汇编语言子过程的连接

1) 对汇编语言子程序进行汇编,生成目标文件 TOMEMO.OBJ;

2) 把汇编后的目标程序拷到 FORTRAN 程序所在的目录;

3) 用 FL 进行编译,生成目标文件 TOMEMORY.OBJ C>FL/C TOMEMORY.FOR;

4) 用 LINK 进行连接,生成可执行文件 TOMEMORY.EXE,

C>LINK TOMEMORY +TOMEMO;

这样,执行文件 TOMEMORY.EXE 实现了 FORTRAN 语言对桥地址的写操作。

注意:我们所用的编译系统是 MICROSOFT 公司的 FORTRAN 4.0 编译软件。

三、FORTRAN 语言对桥地址的读操作 在 FORTRAN 语言中,由于其字符串定长这一特色,决定了在向桥地址阅读字符时,如果读到的字符为 NULL,而前面读取的字符个数不足 32 时,必须继续读取空格字符,直到满 32 为止,这是在编制汇编语言子程序应注意的事项。

1. 汇编子程序对桥地址 1 读操作程序 TOFORT.ASM

```
, MOVE CHAR STRING IN MEMORY TO
FORTRAN
```

```
,TOFORT.ASM
```

```
FRAME STRUC
```

```
SAVEES DW?
```

```
SAVEDS DW?
```

```
SAVEBP DW?
```

```
SAVRET DD?
```

```
CHADDR DD?
```

```
FRAME ENDS
```

```
,
```

```
-DATA SEGMENT WORD PUBLIC 'DATA'
```

```
-DATA ENDS
```

```
CSEG SEGMENT 'CODE'
```

```
DGROUP GROUP -DATA
```

```
,
```

```
ASSUME CS: CSEG, DS: DGROUP, ES:
```

```
DGROUP,SS:DGROUP
```

```
PUBLIC TOFORT
```

```
TOFORT PROG FAR
```

```
PUSH ES
```

```
PUSH DS
```

```
PUSH BP
```

```
MOV BP,SP
```

```
LES AX,[BP]+CHADDR
```

```
MOV DI,AX
```

```
MOV AX,20H
```

```
MOV DS,AX
```

```
MOV AX,1
```

```
MOV SI,AX
```

```
MOV AX,20H
```

```
MOV CX,AX
```

```
AA:LODSB
```

```
CMP AL,0
```

```
JZ BB
```

```
STOSB
```

```
DEC CX
```

```
JNZ AA
```

```
JMP TOEND
```

```
BB:MOV AL,20H
```

```
STOSB
```

```
DEC CX
```

```
JNZ BB
```

```
TOEND:POP BP
```

```
POP DS
```

```
POP ES
```

```
RET 4*1
```

```
TOFORT ENDP
```

```
CSEG ENDS
```

```
END
```

2. 汇编子程序对桥地址 2 或 3 读操作程序 TOFORT.ASM

```
, MOVE CHAR STRING IN MEMORY TO
FORTRAN
```

```
,ENTRY ADDRESS OF MEMORY IS INT 21H
```

```
,TOFORT.ASM
```

```
FRAME STRUC
```

```
SAVDS DW?
```

```
SAVES DW?
```

```
SAVSI DW?
```

```
SAVDI DW?
```

```
SAVBP DW?
```

```
SAVRET DD?
```

```
CHADDR DD?
```

```
FRAME ENDS
```



```

-DATA SEGMENT WORD PUBLIC'DATA'
-DATA ENDS
CSEG SEGMENT'CODE'
DGROUP GROUP -DATA
ASSUME CS: CSEG, DS: DGROUP, ES:
DGROUP,SS,DGROUP
PUBLIC TOFORT
TOFORT PROG FAR
PUSH DS
PUSH ES
PUSH SI
PUSH DI
PUSH BP
MOV BP,SP
LES AX,[BP]+CHADDR
MOV DI,AX
MOV AX,20H
MOV DS,AX
MOV AX,0
MOV SI,AX
LODSW
ADD AX,41H
MOV BX,AX
LODSW
MOV DS,AX
MOV SI,BX
MOV CX,20H
AA:LODSB
CMP AL,0
JZ BB
STOSB
DEC CX
JNZ AA
JMP TOEND
BB:MOV AL,20H
STOSB
DEC CX
JNZ BB
TOEND:POP BP
POP DI
POP SI
POP ES
POP DS
RET 4*1

```

```
TOFORT ENDP
```

```
CSEG ENDS
```

```
END
```

3. FORTRAN 语言对桥地址读操作程序 TOFORTRA.FOR

```
C TOFORTRA. FOR -- MOVE CHAR IN
MEMORY TO FORTRAN
```

```
CHARACTER * 32 CHADDR
```

```
CALL TOFORTRA(CHADDR)
```

```
WRITE(*,10)
```

```
WRITE(*,20)CHADDR
```

```
10 FORMAT (1H0,' THIS IS FORTRAN
STATE!')
```

```
20 FORMAT (1H0,' THE CHAR STRING IN
dBASE IS',A32)
```

```
RETURN
```

```
END
```

在上面的程序中,与之连接是 1 中的 TOFORT,则生成的可执行文件实现对桥地址 1 的读操作,如与之连接的是 2 中的 TOFORT,则是实现了对桥地址 2 或 3 的读操作。

四、dBASE(FOXBASE)与 FORTRAN 语言直接参数传递。

①FORTRAN 语言通过桥地址实现参数传递:

执行下列命令,可以完成 FORTRAN 语言通过桥地址实现参数传递:

```
C>TOMEMORY✓
```

```
C>TOFORTRA✓
```

②FORTRAN 语言与 PASCAL 语言的参数传递顺次执行下列命令:

```
C>WRITE✓
```

```
C>TOFORTRA✓
```

实现了 PASCAL 语言向 FORTRAN 语言的参数传递。

```
C>TOMEMORY✓
```

```
C>READ✓
```

则实现 FORTRAN 语言到 PASCAL 语言的参数传递。

③在前面的系列文章中,把 C,或 PASCAL,或 BASIC 语言与 dBASE(FOXBASE)的参数直接传递程序中的高级语言对桥地址写的可执行文件换成 FORTRAN 语言对桥地址读写的可执行文件,就可以实现 dBASE(FOXBASE)与 FORTRAN 语言的参数直接传递!

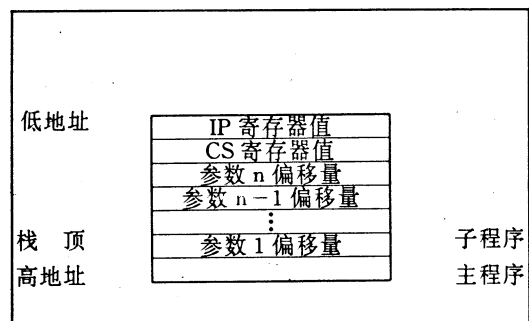
dBASE(FOXBASE)与 COBOL 语言直接参数传递

中国科学院计算中心 王敏生

摘要 COBOL 语言对桥地址的读写操作是通过调用汇编子程序来实现的,在本文中,我们将重点讨论如何实现 COBOL 语言对桥地址的读写操作。

一、COBOL 语言调用汇编语言:COBOL 语言调用汇编语言不同于 FORTRAN 语言调用汇编语言之处是:参数的段地址是当前数据段。

1. COBOL 语言调用子程序员,其堆栈结构变化如下:



2. 设立结构 STRUC 来描述堆栈变化,为要传递的参数分配存储空间:

```

;PARM STRUC
;SAVEBP DW?
;SAVEIP DW?
;SAVECS DW?
;PARMN DW?
;
;PARM1 DW?
;PARM ENDS

```

3. 汇编子程序的返回语句必须是 RET 2 * N, 这里 N 为参数个数。

二、COBOL 语言对桥地址的写操作 COBOL 语言的字符串同 FORTRAN 语言一样,是定长的,不足之处用空格填充,另外,必须保护数据段和附加段的值,这是编制汇编子程序时必须注意的一个问题。

1. 汇编子程序对桥地址 1 的写操作:

```

;WRITE1. ASM

```

```

;MOVE CHAR STRING IN COBOL
;TO BRIDGE ADDRESS
ASSUME CS:CSEG
;PARM STRUC
;SAVEBP DW?
;SAVEIP DW?
;SAVECS DW?
;PARM1 DW?
;PARM ENDS
CSEG SEGMENT PARA
PUBLIC WRITE1
WRITE1 PROC FAR
PUSH BP
MOV BP,SP
MOV BX,[BP]+PARM1
PUSH ES
PUSH DS
PUSH SI
PUSH DI
MOV SI,BX
MOV BX,20H
MOV ES,BX
MOV DI,0
MOV AX,20H
STOSB
MOV CX,20H
REPZ MOVSB
MOV AL,0
STOSB
POP DI
POP SI
POP DS
POP ES

```



```

POP BP
RET 2
WRITE1 ENDP
CSEG ENDS
END
2. 汇编子程序对桥地址 2 或 3 写操作:
;WRITE2. ASM
;MOVE CHAR. STRING IN COBOL
;TO BRIDGE ADDRESS
;THE ENTRY ADDR OF BRIDGE ADDR
;ENTRY ADDR OF INT 80H
ASSUME CS,CSEG
PARM STRUC
SAVEBP DW ?
SAVEIP DW?
SAVECS DW?
PARM1 DW?
PARM ENDS
CSEG SEGMENT PARA
PUBLIC WRITE2
WRITE2 PROG FAR
PUSH BP
MOV BP,SP
MOV BX,[BP]+PARM1
PUSH ES
PUSH DS
PUSH SI
PUSH DI
MOV SI,BX
MOV BX,20H
MOV ES,BX
MOV DI,0
;
MOV AX,ES:[DI]
ADD AX,40H
INC DI
INC DI
MOV BX,ES:[DI]
MOV ES,BX
MOV DI,AX
;
MOV AX,20H
STOSB
MOV CX,20H

```

```

REPZ MOVSB

```

```

MOV AL,0

```

```

STOSB

```

```

POP DI

```

```

POP SI

```

```

POP DS

```

```

POP ES

```

```

POP BP

```

```

RET 2

```

```

WRITE2 ENDP

```

```

CSEG ENDS

```

```

END

```

3. COBOL 语言对桥地址写操作程序

```

* WRITE. COB

```

```

* MOVE CHAR STRING IN COBOL TO
BRIDGE ADDRESS

```

```

IDENTIFICATION DIVISION.

```

```

PROGRAM-ID. WRITE.

```

```

ENVIRONMENT DIVISION.

```

```

DATA DIVISION.

```

```

WORKING-STORAGE SECTION.

```

```

01 PARM1 PIC X(32).

```

```

PROCEDURE DIVISION.

```

```

THIS-IS-IT .

```

```

DISPLAY' * * THIS IS COBOL STATE! * *'.

```

```

DISPLAY'INPUT CHAR STRING PLEASE'.

```

```

ACCEPT PARM1.

```

```

CALL'WRITE1'USING PARM1.

```

```

STOP RUN.

```

上面的 COBOL 程序实现了 COBOL 语言对桥地址 1 的写操作,如果稍加改动,把 CALL'WRITE1' USING PARM1 改变成 CALL'WRITE2'USING PARM1,则就实现 COBOL 语言对桥地址 2 或 3 的写操作。

4. COBOL 语言与汇编语言子程序的连接

1) 对汇编语言子程序进行汇编,生成目标文件 WRITE1. OBJ

2) 对汇编后的目标程序文件拷到 COBOL 程序所在的目录。

3) 用 COBOL 进行编译,生成目标文件 WRITE. OBJ

```

C>COBOL WRITE;

```

4) 用 LINK 进行连接,生成可执行文件 WRITE. EXE

C>LINK WRITE+WRITE1;

这样,可执行文件 WRITE. EXE 与 COBOL 编译系统的公用运行时间库 COBRUN. EXE 实现了 COBOL 语言对桥地址的写操作。

注意:我们所用的编译软件是 MICROSOFT 公司的 COBOL 1.12 编译软件。

三、COBOL 语言对桥地址的读操作 COBOL 语言的字符串与 FORTRAN 语言的字符串一样,都是定长的,故在向桥地址阅读字符时,如果读到字符为 NULL 而前面读取的个数不足 32,这时,应继续读取空格字符,直到满 32 个为止。

1. 汇编子程序对桥地址 1 的读操作程序 READ1.

ASM

```

READ1. ASM
;MOVE CHAR STRING IN BRIDGE ADDRESS
;TO COBOL VARIABLE
ASSUME CS,CSEG
PARM STRUC
SAVEBP DW?
SAVEIP DW?
SAVECS DW?
PARM1 DW?
PARM ENDS
CSEG SEGMENT PARA
PUBLIC READ1
READ1 PROC FAR
PUSH BP
MOV BP,SP
MOV BX,[BP]+PARM1
PUSH ES
PUSH DS
PUSH SI
PUSH DI
MOV DI,BX
MOV AX,DS
MOV ES,AX
MOV CX,20H
MOV DS,CX
MOV AX,1
MOV SI,AX
AA,LDSB
CMP AL,0
JZ BB
STOSB

```

```

DEC CX
JNZ AA
JMP TOEND
BB;MOV AL,20H
STOSB
DEC CX
JNZ BB
TOEND:POP DI
POP SI
POP DS
POP ES
POP BP
RET 2
READ1 ENDP
CSEG ENDS
END
;READ2. ASM
;MOVE CHAR STRING IN BRIDGE ADDRESS
;TO COBOL VARIABLE
;THE ENTRY ADDR OF BRIDGE ADDR
;ENTRY ADDR OF INT 80H
ASSUME CS,CSEG
PARM STRUC
SAVEBP DW?
SAVEIP DW?
SAVECS DW?
PARM1 DW?
PARM ENDS
CSEG SEGMENT PARA
PUBLIC READ2
READ2 PROC FAR
PUSH BP
MOV BP,SP
MOV BX,[BP]+PARM1
PUSH ES
PUSH DS
PUSH SI
PUSH DI
MOV DI,BX
MOV AX,DS
;
MOV ES,AX
MOV CX,20H
MOV DS,CX

```



```

MOV AX,0
MOV SI,AX
;
LODSW
ADD AX,41H
MOV BX,AX
LODSW
MOV DS,AX
MOV SI,BX
;AA:LODSB
CMP AL,0
JZ BB
STOSB
DEC CX
JNZ AA
JMP TOEND
BB:MOV AL,20H
STOSB
DEC CX
JNZ BB
TOEND:POP DI
POP SI
POP DS
POP ES
POP BP
RET 2
READ2 ENDP
CSEG ENDS
END

```

3. COBOL 语言对桥地址 1 读操作程序 READ.

COB

* READ. COB

* MOVE CHAR STRING IN BRIDGE ADDRESS TO COBOL

```

IDENTIFICATION DIVISION.
PROGRAM—ID. READ.
ENVIRONMENT DIVISION.
DATA DIVISION .
WORKING—STORAGE SECTION.
01 PARM1 PIC X(32)VALUE SPACES.
01 PARM2 PIC X VALUE SPACES.
PROCEDURE DIVISION.
THIS—IS—IT
CALL'READ1'USING PARM1.
DISPLAY' * * * * THIS IS COBOL STATE!
* * * *'.

```

DISPLAY' THE CHAR STRING IN BRIDGE ADDRESS IS'.

DISPLAY PARM1.

ACCEPT PARM2.

STOP RUN.

把上面程序中的语句

CALL'READ1'USING PARM1 改动为:

CALL'READ2'USING PARM1

则实现 COBOL 语言对桥地址 2 或 3 读操作。

四、dBASE(FOXBASE)与 COBOL 语言直接参数传递:1. COBOL 语言通过桥地址实现参数传递

执行下列命令,实现了 COBOL 语言通过桥地址传递参数:

C>WRITE

C>READ

2. COBOL 语言与 dBASE(FOXBASE)直接参数传递:

把系列文章中的 dBASE(FOXBASE)与高级语言的直接参数传递程序中高级语言对桥地址读写的可执行文件相应换为 COBOL 语言对桥地址读写的可执行文件,就实现了 dBASE(FOXBASE)与 COBOL 语言的直接参数传递。

IBM TOKEN—Ring 网络技术及功能

吉林省计算机技术研究所 姚学英 徐世坤 陈仲

摘要 本文介绍了 Token—Ring 网络的体系结构、令牌访问、控制协议、网络的硬件配置、令牌环网的功能,具有应用参考价值。

一、引言 IBM Token-Ring 网络是美国 IBM 公司 1985 年向世界推出的网络产品,适用于具有总线型结构的微机连网。1987 年又推出了新型网络适配器,它适用于具有微通道体系结构的微机连网。该网采用环型拓扑结构,令牌访问控制协议传输信息。它是一个标准化的网络,符合 ECMA(European Computer Manufacturers Association 欧洲计算机厂家协会)标准 89 和 IEEE(Institute of Electrical and Electronic Engineers 美国电子电气工程师学会)标准 802.2 和 802.5。1988 年初吉林省计算机技术研究所国内首先对该网汉化。汉化后的 IBM 令牌环网显示的各级菜单、出错信息、站间通信及用户程序均可使用汉字。并先后在十个单位推广应用。运行三年多来,证明该网络系统能够满足机关、企事业单位自动化管理的要求。

二、IBM Token-Ring 网络的 SNA SNA 是 IBM 公司利用其产品为用户提供计算机网络的“系统网络体系结构”。最早于 1974 年提出,以后陆续发表了一些新的版本,使得网络体系结构逐渐完善。

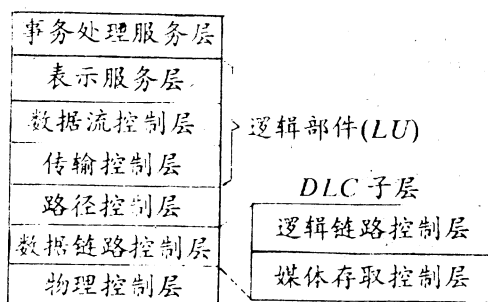
1. 分层网络体系结构 SNA 向用户提供透明的、顺序的数据流通道,这种通道独立于网络的物理结构和拓扑结构,独立于路径选择,独立于传输介质。这种数据流通道被称为会话期或会议。会话期是两个网络可寻址部件之间的临时逻辑连接。用户要通信时,以一定的格式发出命令,由网内的数据流控制层负责建立会话期。通信结束后,通知该层结束会话期,拆除逻辑连接。通信双方称为“端点用户”,它可以是终端旁工作的人,也可以是主计算机中的应用程序。

SNA 定义七个功能层(如图一),每一层都执行专门的功能。例如事务处理服务层的功能如下:

- 提供端点用户对网络的访问;
- 向表示服务层请求服务;
- 与另一个 SNA 基底产品的事务处理服务层通信。又如物理控制层的功能如下:
- 管理结点和连到结点上的传输设备之间的物理接口;
- 执行来自数据链路控制层的服务;
- 与另一个 SNA 基底产品的物理控制层通信。

从事务处理服务层和物理控制层的功能可以看到,每一层除执行本层的功能外,每一层又为相邻层提供不同的服务。对于低层的变动,高层的功能不受影响。因此,为了增强 SNA 的功能,允许采用新的媒体和链路控制。早期的 SNA 版本支持租用 SDLC,点对点通信和物理层上的多结点连接。以后的版本增加了拨

号功能,卫星链路,并支持 X.21 和 X.25 接口。IBM



图一 SNA 结点的分层

Token-Ring 网上的任意两个结点之间能够直接连接。为了充分利用对等层的连接特性,IBM Token-Ring 网上的 SNA 结点可以使用 IEEE 802 委员会定义的局域网媒体存取控制和逻辑链路控制代替 SDLC 数据链路控制。媒体存取控制包含一组关于局域网连接的程序。它的功能主要是监测和诊断环路故障,并能自动恢复环路正常工作状态,处理各工作站初始状态,网络的地址管理,网络路由选择等。逻辑链路控制与具体的媒体存取方法无关,它提供一个面向数据链路连接的服务。这种服务包括对数据顺序发送的支持和一组综合性数据链路层错误恢复的技术。作为分层体系结构,SNA 能够与这个链路控制层结合在一起,而使网络高层功能不受任何影响。这一点对网络的可移植性非常重要。这也是 SNA 结点能够在 IBM Token-Ring 网上存在的原因,也就是说,SNA 支持 IBM Token-Ring 网络链路控制协议。

2. 先进的程序对程序通信(APPC)

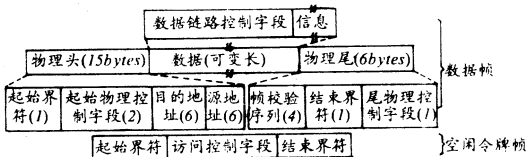
微处理和存储技术的发展,使小的系统能够支持对较大的应用程序的开发成为可能。这样的系统比传统的 SNA 外围结点需要较大范围的连接。应用程序之间的有效通信要求有公用应用程序接口和程序对程序通信的协议。SNA 结点类型 T2.1 允许小系统建立对等层通信,并具有由公用应用程序接口和公用会话期协议提供的先进的程序对程序通信功能。

在 IBM Token-Ring 网络上,SNA 结点 T2.1 采用物理连接,并使用公用数据链路控制。先进的程序对程序通信和 SNA 结点 T2.1 还提供了一些附加支持:允许结点初始会话期,独立地交换信息和终止会话期。由于在局域网和广域网中,SNA 使用同样的通信协议,同样的应用程序接口。因此,一个用户能够在 IBM

Token-Ring 网络工作站上同远程的主机或其它计算机中的应用程序之间传输信息,就同他与其它工作站之间交换信息一样。近年来,IBM 公司研制出的 SNA 低入口(Low-entry)网络协议,通过它提供的目录和中间结点路径选择服务,增进了局域网上结点的扩展能力,加强网络的连接性能和与网中任意结点通信的透明度。使用这样的协议,IBM Token-Ring 网络上的 SNA 结点能够与整个网(指广域网)上任何地方的资源通信。

三、令牌访问控制协议 IBM Token-Ring 网络采用的令牌访问控制协议符合 IEEE 802.5。令牌(Token)是一个具有一定字节数的帧格式。环网上信息的传输均以帧格式进行。数据帧和空闲令牌帧的格式如图 2 所示:

数据帧由物理头、数据字段和物理尾三部份组成。

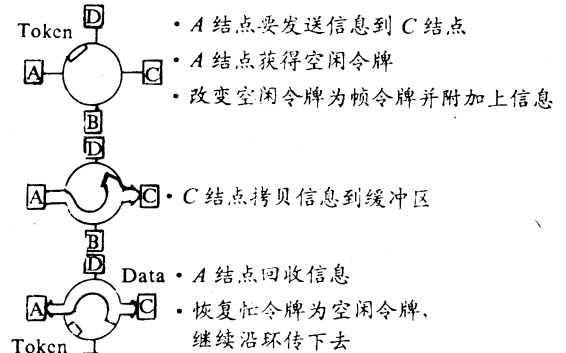


图二 数据帧和空闲令牌帧格式

物理头由起始界符、起始物理控制字段、目的地址和源地址四部份组成,共 15 个字节。物理尾由帧校验序列、结束界符和尾物理控制字段三部分组成,共 6 个字节。数据字段由数据链路控制字段和信息域二部分组成,为可变量。空闲令牌帧由起始物理控制字段的第一个字节访问控制字段,起始界符和结束界符三部分组成,共 3 个字节。访问控制字段中的第 4 位为令牌特征位。它被置成 0 时为空闲令牌,它被置成 1 时为忙令牌。

在环路上任何时候只允许一个令牌存在。可能是忙令牌,也可能是空闲令牌。令牌沿着环路单方向的从一个结点传送到另一个结点。环路上各结点都有机会获得空闲令牌并向媒体上发送信息。一旦某个结点要发送信息,又得到了空闲令牌,则立即将空闲令牌改变成忙令牌,在忙令牌后紧跟着目的地址(接收站地址)和源地址(发送站地址),后面就是要发送的信息。这时在环路上传送的是一个带着忙令牌头的数据帧。其余各结点依次收到该数据帧,各自比较目的地址,与自己的地址不符,立即将它释放。与自己的地址相符,则把

收到的信息拷贝到缓冲区中。忙令牌继续沿环传下去,绕环一周后返回到原结点。该结点从环上回收信息,并核对该信息,若发现错误,重新发送该信息。若没有发现错误,则把忙令牌恢复成空闲令牌,继续传下去。如图 3 所示。使我们看到令牌在环路上访问控制的过程。



图三 令牌访问控制协议

环路上设有检测装置,若检测到令牌丢失,会自动生成一个新的空闲令牌,继续传下去,使网络正常工作。

四、IBM Token-Ring 网络的硬件配置 IBM Token-Ring 网络是一种星形连接的环形拓扑结构,环上最多结点数为 260 个,最大传输率为 16mbps。该网可连接 IBM 的大、中、小、微各类计算机,外部设备和通讯控制器……等设备。构成令牌单环网的基本配置为网络接口适配器、IBM8228 多站访问器、结点和传输媒体等。

1. 网络接口适配器 每个网络结点(如网络服务器、工作站)必须有一个网络接口适配器。不同的微机使用不同型号的网络接口适配器,Ⅰ型主要用于总线型的微机,如 PC,PCXT,PCAT,AST,PS/2-30 等;Ⅱ型主要用于构成桥接器,如:在 AT 机上插二块Ⅱ型网络接口适配器,并运行桥路管理程序,就构成桥接器,可以把两个以上的单环网连在一起。/A 型主要用于具有微通道体系结构的微机,如 PS/2、50、60、80。网络接口适配器的主要功能为:Token 的识别与生成,帧格式的构成,识别和传输,传输错误检测等。

2. IBM8228 多站访问器(MAU) IBM8228 是构成环路的重要部件,它不需要外接电源,在环路上不占用网络地址,起着无源集中器的作用。一个 MAU 最多能连接 8 个结点,MAU 与 MAU 之间可串行连接。

MAU 内有开关元件,可形成旁路支路和辅环路,可自动越过网上没有信号的发生了故障的设备,重新形成环路,保证网络正常工作。因此,使用 MAU 可增加网络的可靠性和可维护性,给环网的配置带来很大的。

3. 网络服务器和 workstation 凡上网的微机中至少有一台被设置成网络服务器,允许设置多个服务器。在网络中,服务器通常是活动中心,因为它有与其它计算机共享的设备或数据。它的硬盘,打印机及信息,常常被多台 workstation 使用。网上的工作速度主要取决于服务器的速度,作为网络服务器最好具有尽可能大的内存和硬盘,尽可能快的处理速度。如:PS/2-80 或 AST/386 等。IBM 的 PC、XT、AT、PS/2 系列机其兼容机都可以作为网络 workstation。根据对它们的不同的要求,可分别把它们设置成信息处理器、收发器和发送器。

4. IBM 的电缆系统 IBM 的电缆系统是一系列标准化产品。目前有四种类型:TYPE1、2、3、为铜质导线,TYPE5 为光纤纤维。TYPE1、2 为屏蔽双绞线,在环路上最多能连接 260 台设备。TYPE3 为普通双绞线,在环路上最多能连接 72 台设备。这三种类型的电缆既可以作为网络适配器与 MAU 之间的连接媒体,又可以作为 MAU 之间的连接媒体。光纤电缆不能用在 MAU、中继器的环路连接上,只能用于 MAU、中继器的环路连接上。MAU 之间用双绞线连接而且加中继器,最长距离为 750 米。使用光纤电缆连接而且加中继器,最长的连接距离可达 2 公里。

以上为单环网的基本配置,如果环路上的设备数多于 260 个,可以使用桥接器把两个单环网连接在一起,也可以把多个单环网通过多个桥接器与主干网连接在一起。令牌环网还可以通过网间连接器(Gateway)与具有不同协议的网连接在一起,例如令牌环网可以与以太网互连在一起,互相通信。还可以通过调制解调器与远程和计算机通信。

五、令牌环网的功能 IBM Token-Ring 网面向 PC 机配置了较丰富的软件,可在 DOS3.2(或更高版本)支持下工作。基本软件如下:

- IBM PC LAN 程序
- IBM PC LAN 信息管理程序
- IBM PC LAN 安装辅助程序
- IBM PC LAN 支持程序

除此之外还有环网诊断程序、桥路管理程序、IBM PCNET SNA 3270 仿真程序、IBM 令牌环网与 PC 网互连程序、APPC/PC 程序等。由于令牌环网合理的结构、丰富的软件、先进的管理,使其具有很强的功能。

IBM PC LAN 程序允许网上的计算机互相通信,

共享资源。它的基本功能如下:

1. 具有四种网络设置 在启动网络时,对连在网上的每一台微机要分别进行设置,至少设置一台微机作为网络服务器,以便资源共享。由于这四种设置的功能不同,系统软件占内存的多少不同,因此要根据需要进行设置。可以设置成服务器(占内存 350K),信息处理器(占内存 160K),收发器(占内存 68K),发送器(占内存 50K)。服务器可以完成 8 种功能,而发送器只可以完成 2 种功能。对网络上每台微机进行设置后,均可自动生成批命令文件,再次启动时不必重新设置。也可以取消设置后再重新设置。

2. 具有两种运行方式 令牌环网有两种运行方式,菜单方式和命令方式;菜单方式分四级,都是汉字的条文,供用户选择。对网络不熟悉的用户,使用菜单方式,操作简单,容易掌握。对网络熟悉的用户,使用命令方式更方便,省去了菜单逐级调用的时间。共有 14 条命令,每条命令都是多种格式,多个参数。凡是在菜单方式下完成任务,在命令方式下也可以完成。

例 1:NET SEND* 今天下午一点开会

这条命令发出后,连在网上的所有计算机名都收到该条信息。

例 2:NET SHARE DISK=C:\HOW /RWC

与其他网络用户共享你的硬盘 C 的根目录,硬盘 C 的网络名为 DISK,该硬盘 C 有一个口令字为 HOW, RWC 表示其他用户可以在该硬盘 C 上读,写和建立文件。

3. 具有很强的信息处理功能

• 发送信息,能以点对点方式从一台计算机把信息发送给另一台计算机,也能以广播方式从一台计算机把信息发送给网上所有的计算机。

• 能随时接收来自其它计算机的信息,并能立即将信息显示在屏幕上。也能把发来的信息存贮起来,以后查看。当你工作忙,收到的信息不希望显示在屏幕上,可以建立日志文件,存放收到的信息。

• 能用附加名接收信息,每个计算机名可增加 1 至 12 个附加名。

• 能寄发信息,如果你不想在你的计算机上接收更多的信息,或者你有事需要关机时,你可以寄发信息。例如:有甲、乙、丙……等网上用户,乙让丙寄发信息,乙关机离开,甲给乙发送信息时,丙代替乙接收到信息。

4. 设有网络请求键,在实际应用中非常方便,但只有在服务器和信息处理器设置时,才能使用网络请求键。使用网络请求键,可以中断应用程序去执行网络任

务,完成网络任务后直接返回到应用程序。

例如:某人正在使用字处理程序,她想打印文件而又忘记连打印机(网上打印机),如果她能使用网络请求键(ctrl-Alt-Break),两步可以完成:字处理程序

Ctrl-Alt-Break
↔
主菜单(连打印机)
Ctrl-BREAK+Enter

如果她不能使用网络请求键,需四步才能完成:

字处理程序 $\xrightarrow{KD \ X}$ DOS $\xrightarrow{NET \downarrow}$ 主菜单(连打印机)
 $\xrightarrow{WS \downarrow}$ \xrightarrow{ESC}

除了上述几个主要功能外,还有一些功能,如可以改变网络启动的缺省值,可以随时查看网络状态,可以查看网络的错误信息表,对于每个网络任务有暂停和继续的功能等。

六、结束语 综上所述,IBM Token-Ring 网由于采用媒体访问方式,加之合理的结构,先进的工艺,丰富的软件,使其在传输速度,实时性,适应的地理范围,上网的用户数均优于 CSMA/CD 总线网,传输速度和数据和吞吐能力高于令牌总线网。特别是它很容易以光纤电缆作为网络传输媒体,可以使网络性能大幅度地提高。

参考文献

[1] IBM 资料,《IBM Token-Ring Network Technology》

[2] IBM PC Local Area Network Program User's Guide

LASER-PP40 汉化曲线绘制软件 1.00 版

郑州信息工程学院二系 周东方 肖杰 商胜伟 张跃元

摘要 汉化曲线 CAD1.00 版,在 IBM PC/XT 及其兼容机上,实现了以中文对话方式对任意曲线进行单色和彩色的各种显示,支持 Laser-PP40 描绘器对任意曲线进行单色或彩色的各种绘制。

一、系统配置 1. 适用微机:IBM PC/XT、PC/AT、AST/286、386/AT 及其兼容机。

2. 盘片配置:5 $\frac{1}{4}$ 寸软盘二张(高显),5 $\frac{1}{4}$ 英寸软盘二张(中显)。

3. Laser-PP40 描绘器一台。

4. 自备系统,独立运行。

二、Laser-PP40 汉化曲线 CAD1.00 版功能介绍

Laser-PP40 汉化曲线 CAD 软件使用带有标准接口的香港伟易达公司生产的 Laser-PP40 描绘器,在人机中文对话方式下实现了对函数曲线与数据曲线的自动描绘,各种曲线拟合描绘和显示。其基本功能有:

1. 按照需要绘制任意形式的二维直角坐标系。

a. 直角坐标系的横轴、纵轴的起始值,终止值,刻度点数可随意确定。

b. 横纵轴的刻度值(指数形式时为其系数部分)印出到小数点的后两位(见图 1)。

c. 横纵轴的刻度值大于 1000,或小于 0.01 时,自动以指数形式标出(见图 1)。

d. 横纵轴的标志可设为任意字符(见图 1)。

e. 可根据需要自动标出坐标零线和坐标方框(见图 1)。

2. 按照需要绘制任意形式的极坐标系和对数坐标系。极坐标系坐标值为极轴长 ρ 和极轴与横轴正向的逆时针夹角 θ 。对数坐标系横轴同一般直角坐标系,纵轴刻度以指数值均分标注。其它特点与一般直角坐标系 a-e 点基本相同。

3. 按照需要绘制任意形式的曲线

a. 绘制任意形式的函数曲线。

b. 绘制任意形式的数据曲线。

4. 绘制曲线时有多种输入方式可供选择。

a. 数据方式:直读数据方式,数据文件方式。

b. 函数方式:标准函数直接输入方式(见常用各种语言标准函数)和一般函数文件输入方式。

5. 曲线颜色可有四种选择。

可任意选择黑色、蓝色、绿色和红色。

6. 可按照需要对曲线进行任意多次数的加深。

7. 同一坐标系内可描绘任意多条曲线,可自动标注每条曲线的名称或取消自动标注改为特殊标志设置。

8. 具有对十六种线型的随意选择功能和线型自动设置功能。

9. 对曲线的某一段可按需进行局部放大。

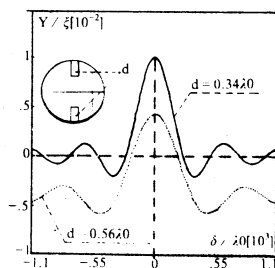


图 1. CAD 曲线图例

10. 具有在曲线图上,沿四个方位旋转标注六十四种不同字号的字符和字符串的功能(见图 1 中各字符)。

11. 能在曲线图中描绘出各种特殊用途的图案(见图 1)。

12. 绘制曲线的全过程横纵轴向总误差恒小于 0.1mm。

三、结束语 Laser-PP40 汉化曲线 CAD1.00 版实现了以人机对话方式,直接完成平面直角坐标系的绘制,极坐标系的绘制,对数坐标系的绘制,具有曲线输入方式,曲线颜色和曲线深浅度的选择,坐标轴标志设定,曲线标志设定,坐标系中特殊用途图案设置,特殊字符和一般字符描绘,字号选择和串字符旋转描绘等十几个方面的各种功能,可获得美妙的艺术曲线和画面。经过近两年以来的广泛使用,说明一般非计算机专业人员,也只需要十几分钟的学习,即可掌握。作为绘图工具,用起来十分简捷,是有效性和实用性独到的曲线绘制工具,可广泛用于各种曲线的绘制,节省大量的时间和人力。

参考文献

[1]《Laser-PP40 描绘器使用手册》,香港伟易达公司,1984

[2]《AutoCAD 应用与软件开发指南》,中国科学院希望高级电脑技术公司,1988.3

打印机共享器的研制

湖北 潜江 江汉石油管理局测井工程处计算室 余祥斌

摘要 本文从实际应用出发,介绍了打印机共享器的设计原理,拟出了线路框图,并给出了具体线路图。

一、前言 打印机是计算机不可缺少的外部设备。

目前的配置是一台微机配备一台打印机。在微机集中的地方,这样的配置造成打印机的使用效率不高。能不能使多台微机共用一台打印机,就象一些大型机的许多终端可以共用一台打印机一样,从而提高打印机的使用效率呢?通过分析研究,我们认为可以的。以下是我们对两台微机共享一台打印机的共享器的研制,该共享器达到了预期设想目的,具有以下特点:

1. 接口的所有信号线的电平与 TTL 电平兼容;
2. 无须对两台微机的软件和硬件做任何修改;
3. 全自动控制,无任何手动开关;
4. 打印机空时,两台微机都可以使用打印机。当一台微机正在使用打印机时,能可靠地抑制另一台微机使用打印机,并发给另一台微机一个 ERROR 信号,提示对方本机正在使用打印机。两台微机使用打印机的优先级一样;

5. 该共享器使用打印机输出信号连接器上的电

源,无须另配电源;

6. 如用多个共享接口,可连接成多台微机共享一台打印机的情况。例如:三台微机共享一台打印机按如

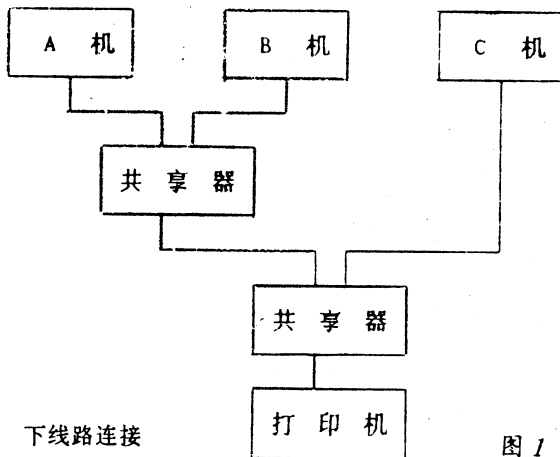


图 1

7. 该原理可以推广至其它一些外设,如绘图仪等。

二、设计原理 两台微机使用一台打印机的原理图如下所示:

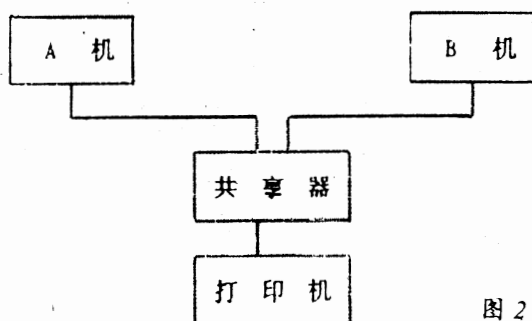


图 2

标准并行接口的打印机具有以下一些接口信号:

1. STROBE 读入数据的选通脉冲
2. DATA1—DATA8 八位并行数据信号
3. ACKNLG 数据接收响应信号
4. BUSY 忙信号
5. PE 纸尽信号
6. SLCT 打印机选中信号
7. AUTD 自动走纸一行信号
8. 0V 逻辑地电平
9. 地地信号
10. +5V 打印机+5V 电源信号

11. INIT 初始化打印机信号

12. ERROR 打印机出错信号

要实现一台微机正在使用打印机时,自动地抑制另一台微机使用打印机,则必须在一台微机使用打印机时,发出一个电平信号用来限制另一台微机使用打印机。分析微机的控制信号,未发现有这种电平信号。然而,进一步分析得知,只要微机使用打印机,便会发出数据选通脉冲 STROBE,该脉冲的宽度大于 $0.5\mu\text{S}$ 。通过转换线路,我们可以将 STROBE 数据选通脉冲信号转换为低电平信号,该低电平信号有两个作用:一是打通本机的控制信号和数据信号通路;二是封锁另一台微机到达打印机的控制信号和数据信号,并给另一台微机发出—ERROR 信号,提示另一台微机,该机正在使用打印机。当该机使用完打印机后,控制低电平信号通过一定的延时时间后,恢复到高电平。此后,两台微机都可以使用打印机。在线路设计中,我们将两台微机的优先级定为一样,打印机响应优先发出 STROBE 数据选通脉冲信号的微机。

三、线路框图与具体线路 以下是两台微机共用一台打印机的线路框图

打印机共享器的具体线路图如附图 1 所示

图中:CLA 是 A 机发出的 STROBE 信号所产生的低电平控制信号

CLB 是 B 机发出的 STROBE 信号所产生的低电平控制信号

该线路用了以下一些元器件:

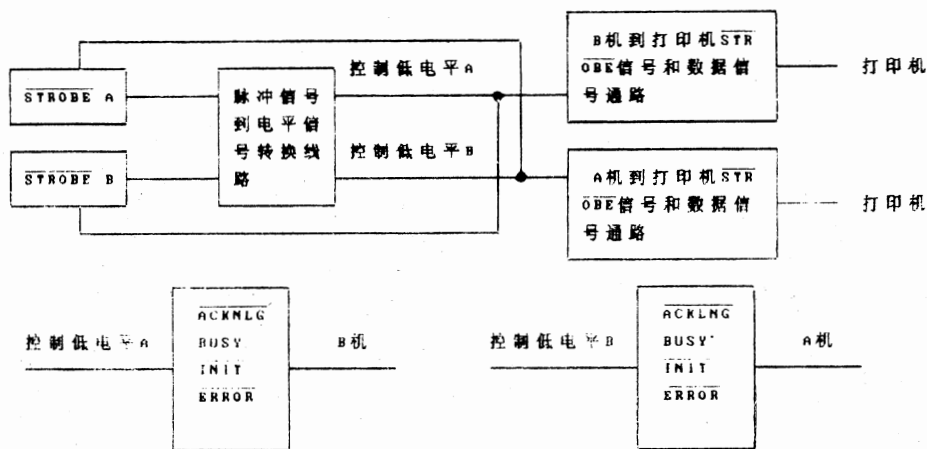


图 3 打印机共享器线路框图

单片机多点温度检测的实现

河南省计算中心 崔振远

工业生产中,几乎都离不开温度测量与控制,近几年发展起来的计算机温度检测,对于合理利用能源、提高产品产量和质量具有十分重要的意义。只是此类巡检系统中核心设备大多采用微、小型机,造价较高;而采用单片机设计成专用的多点温度检测设备,对于提高系统可靠性,降低成本无疑有重要意义。

一、基本设计思想 基于设计较为通用的检测设备,可能工作于各种不同类型的环境,提高设备可靠性和抗干扰能力应作为一个重要的指导思想。在设计中我们采用如下几点措施:

(1)考虑到信息可能长距离传输以及系统多运行于电磁干扰严重的环境,我们从硬件结构及软件设计上均采用了一系列措施。在硬件上我们采取严格地“级”间屏蔽及带通滤波抑制干扰。

(2)考虑到设备对环境的适应能力,在线路结构选择上尽量考虑选择对元、器件稳定性及线性性能要求不严格的线路,特别注意降低元器件热稳定性的要求。这样不可避免地要造成线路的复杂性,但是过于复杂的设备不可避免地造成系统可靠性的降低,对此一方面需要权衡利弊选取合理的结构方案,另一方面充分发挥单片机的潜在功能,尽量简化设计。这样有较多的对比方案可供选择。

(3)提高测量精度及可靠性,降低整体系统的成本,不仅应在硬件设计中下功夫,更重要的是充分利用系统的核心——计算机的软件功能上着手,在软件设计中采取相应措施,利用软件实现系统抗干扰,自校正等措施,对测量结果进行误差校正,对不同的测温元件及测量范围采用不同计量方程及校正因子。

二、基本功能 (1)检测点数:8,15,20,30,40,60

(2)测温范围:0~1800℃

(3)测温元件:可用铂热电阻、铜热电阻、热电偶等

(4)分辨率:0.2℃(0~1000℃)

(5)工作环境:温度0~40℃,相对湿度<80%

(6)具有数字显示,打印输出各点温度,并能设定各点报警温度,超界蜂鸣报警,显示和打印警报点情况。

三、结构原理

单片机多点温度检测系统原理如图1。下面介绍其主要部分:

(1)测温元件:如前所述可采用热电偶、热电阻等。热电阻采用三线制接法,不需外接调整电阻,并能减少因连接导线长短不一所引起的影响。热电偶输入时自动进行冷端补偿。

(2)采样开关:采用模拟多路转换技术进行各通道信号获取,在此之前有电信号生成电路,把测温元件得到的信号进行标准化处理。

(3)信号放大:由采样开关来的毫伏级的电压信号经集成放大器放大后,送到单片机控制的A/D转换器中量化处理。

(4)模数转换:模数转换把由信号放大器送来的表示温度高低的电压信号量化成计算机所能接受的数字信号。根据需要可采用不同方式,为提高系统抗干扰能力,我们采用如图2所示,由计算机控制的单基准双斜积分器作为A/D转换器。采样积分和比较积分的计数频率采用单片机的时钟频率。由于这种方法采用了平均值转换,所以对“过零”干扰及常态干扰有较好的抑制能力。

(5)单片机系统:采用美国英特尔公司产品MCS-51系列单片机和相应的存贮芯片以及接口芯片构成一个完整的计算机系统。系统中存贮器8.5KB。系统所采取接口电路如图2所示。

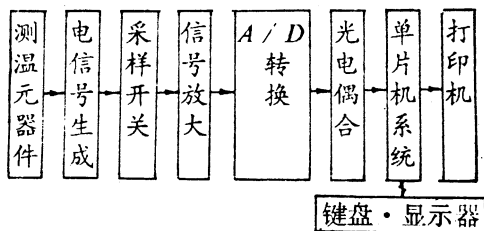


图1 系统结构原理图

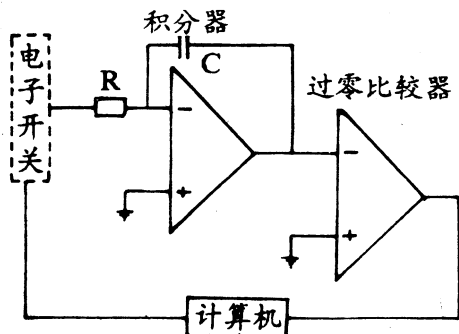


图2 双斜率积分型 A/D 转换器

- a. 显示器及键盘接口。
- b. 采样开关控制接口。
- c. 报警器接口。

d. 标准八位并行接口。

该系统所有运算,检测功能由计算机程序控制。

四、有关说明: (1)系统软件:该检测系统主要包括如下软件 a. 基本程序:主要完成设备自校,自身检验及顺序检测功能以及基本初始参数的设定等功能。b. 数据处理程序:完成计算机内码温度数据到实际温度数据的转换及误差分析校正。对于不同测温元件及不同的测温范围提供不同的计值和校验方程,这种软件对提高系统测量精度,对于降低设备成本、降低对元

器件及电路要求有重要意义,其中误差处理和修正程序,提供在一定硬件设备条件下获得较高的测量精度的可能。c. 显示和打印程序:不言而喻,为显示及打印测试结果而设置的程序。

(2)该系统设计中仅考虑了多点温度检测,实际上增加热源控制和执行机构,即可完成热系统控制功能。在设计中采用了较灵活的通用接插板方式,软件设计方案中也留有相应的接口,增加控制功能并不困难。

GW0520 CH 自检 RAM 到 3KB 死机检修一例

武汉二炮指挥学院计算机室 姚行中

我们在维修长城 0520CH 机时,遇到几台都是上电自检到 3KB 时死机的现象,此时测量 CPU 的 CLK、READY,以及 AD_0-AD_7 , A_8-A_{15} 正常,但 8088 的 $S_2S_1S_0=111$,即总线周期过渡状态。其原因是决定 RAM 芯片 4164 行列地址选通允许的延时芯片 DL-1 输出幅度不够造成的。这部分电路和 PC/XT 的结构与工作原理有所不同,如图 1 所示。

由图 1 可见,当 CPU 对主板上的 512KBRAM (00000H-7FFFFH)读或写时, U_{7-1} 输出高电平,该信号一方面经 T_6 反相后在 T_{6-10} 和 T_{6-2} 两引脚上得到低电平,作为主机 RAM 的上半部和下半部行选通信号 RAS,将地址总线低 8 位 A_7-A_0 固化在所有 4164 芯片中;另一方面该高电平信号经 W_{7-6} , W_{7-8} , P_{7-6} 和 Y_{7-10} 脚后获得高电平的 RAMSEL 信号,送 DL-1-1 脚,经 DL-1 延迟 60ns 后,在 DL-1-10 脚输出高电平,经 R_{7-11} , R_{7-10} 和 R_{7-5} 后,在 R_{7-6} 和 R_{7-8} 获得高电平的主机 RAM 上半部和下半部的 CAS 允许信号,把由门阵 ACI-C-1001 送来的, CAS_0-CAS_7 经与非

门后送到相应各组 RAM 的 \overline{CAS} 端,但只有被 ACI-C-1001 译码选中的某组 RAM 的 \overline{CAS} 端,才有低电平的列地址选通信号,把由 ACI-C-1001 送来的高 8 位 $A_{15}-A_8$ 固化在该组 RAM, CPU 即可对该组 RAM 进行读或写。另外,与 PC/XT 不同, DL-1-8 输出信号(比 DL-1-1 延迟 100ns),送主机面板 RAM 显示存储器 f5(LS374)的 CLK 端,以显示出 CPU 正在访问的 RAM 区间。

正常情况下 DL-1 的输入输出波形如图 2 所示。图 3 为从故障机器上测得的波形。

在开机瞬间, DL-1-10 还是正常的,但随着温度升高,其波形下降为图 3 所示的临界工作状态,多数故障机器是自检到 3KB 时 DL-1 不能正常工作。偶而使自检超过 3KB 甚至启动操作系统。这种故障现象的不确定性和 DL-1-10 输出幅度的可变性给维修带来了困难。DL-1 在电路板上的座标为 X8,型号为 DT14CB201 KAPPA-TAIWAN。

(图 1、图 2、图 3 见下页)

简讯

传播信息的窗口
交流学术的讲坛

结识同行的良机
展销产品的市场

第四届全国计算机应用研究学术交流及 新产品、新成果展示会即将在蓉城拉开帷幕

翘首期待的第四届全国计算机应用研究学术交流及新产品新成果展示会,即将于今年 9 月 5 日至 9 日在西南交通大学成都总部(成都火车北站西侧九里堤)新落成的明诚堂学术厅拉开帷幕。热忱欢迎社会各行各业计算机界广大同仁、公司、厂家、用户踊跃参加,光临指导!

本届会议将开展学术论文评奖、颁奖活动,颁发学术论文录用证书,举行专题学术报告,展示计算机推广应用的新产品、新成果,发布新闻信息,内容丰富多彩。会后将出版学术论文专辑。

参展单位与会代表请于 9 月 4 日在西南交通大学成都总部西大门招待所报到;参观单位、团体及个人需参观者,请同本刊联系。

联系电话:582666 转 2055

784160 转 746

电 挂:成都 5938

联系人:张钢 邓嘉澍 唐大利

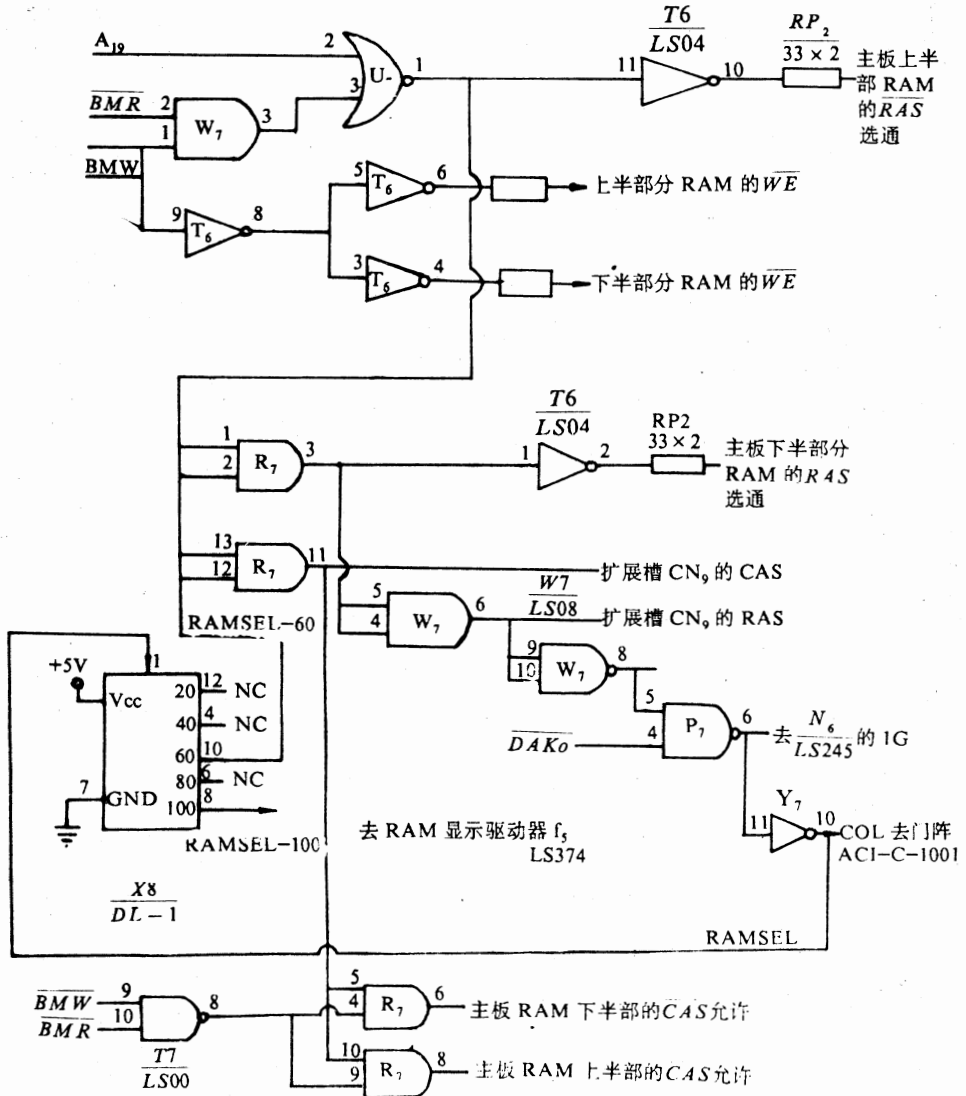


图1 RAM 行列地址选通或允许

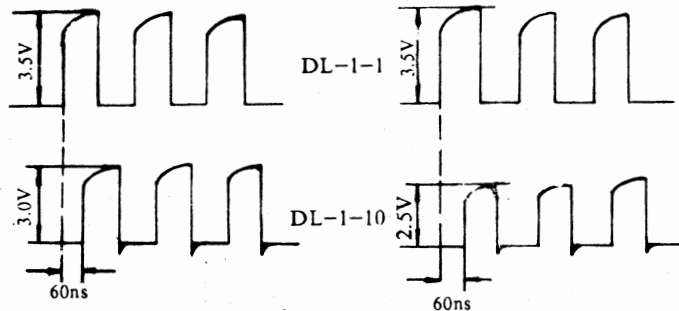


图2 正常 DL-1 的波形

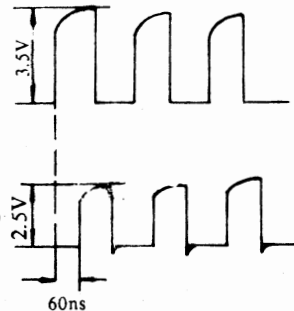


图3 故障状态的 DL-1 波形

WHZ-851 自动比色分析仪研制与实验

郑州信息工程学院 周东方 侯志强 马庆海
郑州 153 中心医院 王广兰

摘要 用 CMC-80 双板微处理机对 721 型分光光度计进行改造,研制成功了 WHZ-851 型自动比色分析仪。这种比色分析仪具有自动贮存、转换标本、设置各种检测项目、显示、打印诸功能,且生产成本低廉,使用方便,具有很高的应用价值。

一、比色仪系统组成 该仪器由微处理机, Laser-PP40 描绘器,改装的 721 型分光光度计及系统软件组成,如图 1、2 所示。

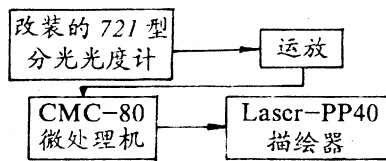


图1 WHZ-851型比色仪系统组成框图

二、实验结果 该仪器在直接测试结果的同时读取原表头的吸光度(简称A),现以葡萄糖氧化酶测定血糖为例,用不同含量 50mg/dl, 100mg/dl, 150mg/dl, 200mg/dl, 300mg/dl, 400mg/dl,

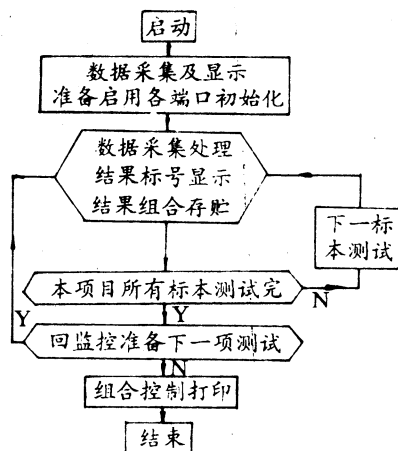


图2 比色仪系统软件流程

500mg/dl 的葡萄糖标准液各测 20 次,实验证明:自动测试结果的准确度和精密度均高于表头 A 计算出的结果(见表 1)。

表 1、两种测试结果准确度和精密度比较

葡萄糖含量	比色分析仪				表头 A			
	\bar{X}	SD	CV%	准确度 %	\bar{X}	SD	CV%	准确度 %
50	49.5	0.22	0.44	-1.13	51.0	0.55	1.08	1.96
100	101.3	1.97	1.94	1.31	101.5	2.3	2.27	1.45
150	150.6	0.87	0.58	0.40	153.4	0.92	0.60	2.27
200	199.3	3.99	2.01	-0.35	196.9	4.69	2.39	-1.74
300	299.0	1.79	0.60	-0.35	305.0	1.98	0.65	1.68
400	407.0	5.53	1.36	1.76	409.0	7.33	1.79	2.25
500	508.7	5.34	1.05	1.74	513.0	8.46	1.65	3.25

其中: \bar{X} 为均值, CV% 为变异系数, SD 为标准差, 准确度% = (真值 - 均值) / 真值 × 100%

三、结束语 本仪器目前能一次显示贮存 21 份标本的数据, 每一人份标本可测出 37 种不同的检验项

目,并能将结果按所测项目和要求组合打印在一起,比色部分在 721 型分光光度计的基础上,将手推式(每只只装 4 份标本)改为自动转盘式(每次可装 40 份标本),减少了反复换液的烦琐手续。本仪器比原来手工比色、计算等提高功效 20 多倍,特别在组合打印、自动连续比色等方面国内尚未见报道。它适用于临床生物化学分析,工业、农业及科研等化学分析工作、具有广

泛的应用前景。

参考文献

[1]《LaserTM-PP40 描绘器使用手册》,香港伟易达公司,1984。

[2]《CMC-80 微处理机使用说明书》,郑州信息工程学院,1985。

工业控制计算机中 RAM 数据保持和停电瞬间复位

机电部第十一设计研究院 戴晓黔

一般的工业生产过程都要求使用于工业控制的计算机具备掉电时能保持实时数据,记下停电时生产过程的状态,以便重新上电时,受控的生产过程能从停电时的状态开始继续运行。反映工业过程实时状态的实时数据都是由计算机实时采集并存储于随机存取存储器 RAM 中。目前国产的许多用于工业控制的计算机没能很好地解决这个问题。停电时, RAM 中的数据容易丢失。结果经常在重新上电时需要人为干预去设置反映生产过程状态的数据,工业控制过程才能继续。

笔者就怎样避免 RAM 中的数据丢失进行了一些探讨。

停电时 RAM 中的数据丢失,一般说来有两种原因;一种是由于电源的原因;另一种是由于 RAM 芯片的控制线上出现错误信号,对 RAM 进行了误操作,改变了 RAM 的存储内容,引起数据丢失。

由于电源的原因引起的 RAM 数据丢失,具体分析起来有二种情况。一种是由于备用电源的电池过度放电,使得 RAM 的电源电压过低。另一种是在主工作电源停电时,备用电源投入不及时,使 RAM 的电源电压产生波谷状“L”脉冲,从而引起数据丢失。

由于 RAM 芯片的控制线上出现的错误信号引起的 RAM 数据丢失也可分为二种情况。一种是主工作电源处于稳态正常情况, RAM 控制线上出现错误信号,引起数据丢失。另一种是主工作电源处于过渡过程,上电或掉电瞬间 RAM 控制线上出现错误信号,引起数据丢失。

下面,我们就以 Z80 系列微机芯片组成的工业控制机为例,讨论避免 RAM 数据丢失的方法。

首先解决由于电源的原因引起的数据丢失。笔者

设计了如图 1 的 RAM 数据保持电源电路。图中电池 BAT 采用 4 节额定电压为 1.25V,额定贮能为 0.5 安时的可充电镉镍电池。当 220 伏交流电存在时,稳压器 7805 的输出电压 V_c 通过二极管 D_3 向集成电路

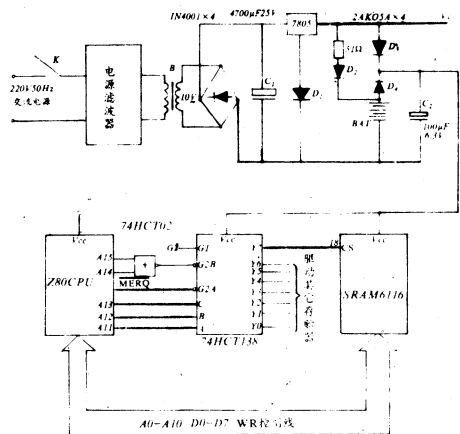


图 1

74HCT138(地址译码器)和 SRAM6116 供电,由于电解电容 C_2 的作用,使电源的自动切换平缓而无脉冲产生。当 220 伏交流电停止时,电池通过 D_4 向 74HCT138 和 SRAM6116 供电。当有工作主电源 V_c 存在时, V_c 还可以通过 51 Ω 电阻及二极管 D_2 向电池充电。这样使得当交流电源存在时,电池所存能量总是处于额定状况。

交流电源断掉后,电池能够维持多久呢?如图 1 所示,停电后电池将通过 5 个回路放电。1、是通过充电回路二极管 D_2 反向泄漏,其漏电流 $I_{D2} < 15\mu A$; 2、是通

过二极管 D_3 反向泄漏,其漏电流 $I_{D3} < 15\mu A$; 3. 通过电解电容 C_2 泄漏,采用低漏型 CD11D 电解电容,其漏电流 $I_{C2} \leq 0.004CU$,其中 I_{C2} 单位为 μA , C 为电容器的电容值,单位为 μF , U 为额定电压,单位为 V ,则 $0.004 \times 100 \times 6.3 = 2.52(\mu A)$ 即 $I_{C2} \leq 2.52\mu A$; 4. 集成电路 74HCT138 的静态电源电流 $I_{138} \leq 2\mu A$; 5. SRAM6116 的静态电源电流: $I_{6116} \leq 20\mu A$ 则电池总的工作电流;

$$I_B = I_{D2} + I_{D3} + I_{C2} + I_{138} + I_{6116} \\ \leq 15 + 15 + 2.52 + 2 + 20 = 54.52(\mu A)$$

(装好电路后,我们实测值为 $15\mu A$)

则电池能维持的时间:

$$T = \text{电池额定安时数} \div \text{电池工作电流} \\ = 0.5 \div (54.52 \times 10^{-6}) \\ = 9171(\text{小时}) \\ \approx 382(\text{天})$$

根据目前一般工业控制机的工作状况,这么长的保持时间是足够的了。而且当交流电上电后,将对电池进行充电,所以只要连续断电时间不超过 382 天,电池都能维持 RAM 所需电源电压。

对于因为 RAM 芯片的控制线上出现错误信号,对 RAM 进行误操作从而引起数据丢失的情况应当怎样避免呢?

当工作电源处于稳定状态(有电或无电两种状态)时, RAM 控制线上出现的错误信号往往是由电源回路和控制系统的输入输出回路引入及环境电磁干扰所引入。如图 1 在交流电源引入端加上交流电源滤波器,再通过变压器隔离,直流电源端滤波电容,另外再通过直流稳压器 7805 就可以有效地滤掉交流电源引入的干扰信号。为防止从输入输出回路引入干扰信号,可在输入输出接口电路中加入光电耦合电路和滤波电路。为防止环境中的电磁波干扰,只要采用具有良好接地的金属机箱进行屏蔽,一般也就能够达到要求。

上电或掉电瞬间的情况又是怎样呢?上电时,由于上电复位电路的作用, CPU 处于复位状态,其地址总线和数据总线都为高阻抗状态,而且所有的控制输出信号全为无效状态,因此一般上电时 CPU 对 RAM 不会发出错误信号,掉电时的情况就不同了。如图 2 所示波形,当电源电压下选到 4.75V 以下时, CPU 的复位端 (RESET) 电平仍为“1”电平。由于电源电压已不正常, CPU 就会向 RAM 发出错误的控制信号,引起 RAM 数据丢失。

要避免在电源掉电瞬间所引起的 RAM 数据丢失,主要就是要解决图 2 中 Δt_1 时间内怎样封锁 RAM

不允许 CPU 对其进行误操作或者使 CPU 复位不发出错误信号。

图 3 所示电路解决了这个问题。

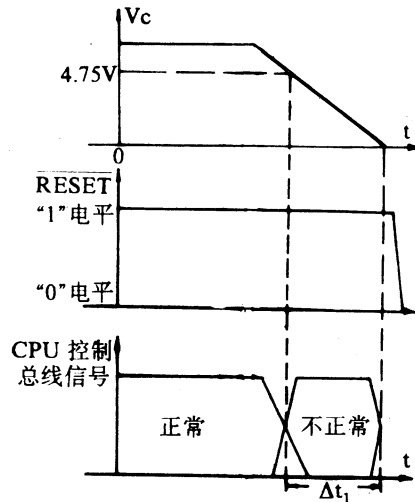


图 2

其工作原理是:我们应注意到图 1 中,稳压器 7805 的公共端处加入了一锗二极管 D_1 ,使得直流工作电源电压稳定时为 5.3 伏。此时图 3 中三极管 T_1 的射极基极间电压为 $5.3 - 4.3 = 1V > 0.7V$ 晶体管导通电压,则 T_1 导通,则 T_2 、 T_3 组成的复合管截止。则 T_2 发射极(图 3 中 A 点)电压为 5.3V, B 点电压 $V_B = 5.3 \times (1000 \div (1000 + 47)) \approx 5.1(V)$,系“1”电平,通过 74HCT04 两次反向,则 CPU 的 RESET 脚为“1”电平, CPU 未复位,正常工作。而 G_1 点电压为:

$$V_{G1} = 5.3 \times 1000 \div (1000 + 10) \approx 5.2V$$

也为“1”电平,则存储器地址译码器 74HCT138 未被封锁而正常工作。

当交流电源掉电时,一旦直流电源电压下降到小

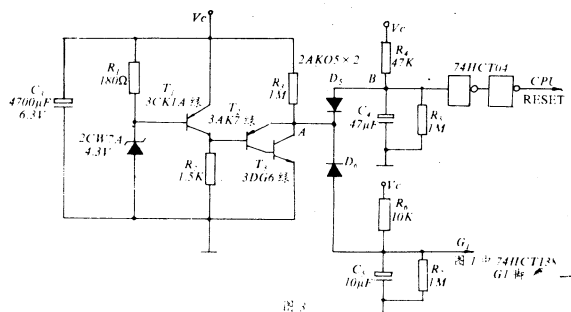


图 3

于5V时,则三极管 T_1 射极基极间电压就小于 T_1 的导通电压0.7V,则晶体管 T_1 截止,于是 T_2T_3 组成的复合晶体管饱和导通。则 C_4 通过 D_5 及 T_2T_3 放电; C_5 通过 D_6 及 T_2T_3 放电,则B点电压很快降到 D_5 导通电压及 T_3 饱和电压之和,即0.6V;从而CPU因RESET信号电平变为“0”电平而被复位,同时G1点电压也很快降到0.6V,从而74HCT138的选通端G1亦被封锁,不允许CPU对SRAM6116再进行任何操作。使得在掉电瞬间起到保护RAM数据的作用。

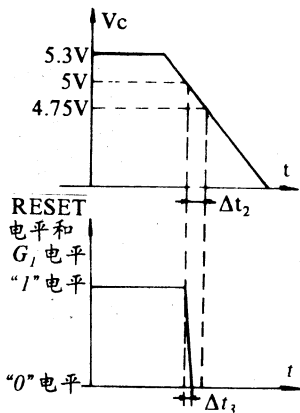


图4

掉电时电路工作时序分析如下。如图4的波形,电源电压从+5V下降到4.75V所需时间这样计算:我们从最快的情况去考虑,只考虑图3中 C_3 4700 μ F电解电容所贮电能,同时认为系统所耗工作电流为极限情况1.5A,则有:

$$\Delta t_2 \cdot I_{\max} = \Delta V \cdot C$$

这里 Δt_2 为电源电压从5V下降到4.75V所需时间单位微秒, I_{\max} 为系统所耗最大电流1.5A, ΔV 为

5-4.75=0.25V,C为电容值单位微法。于是有:

$$\Delta t_2 = 0.25 \times 4700 \div 1.5 \approx 783(\mu S)$$

现在计算B点和G1点电压从“1”电平下降到“0”电平所需时间 Δt_3 。我们以其最慢的情况讨论,考虑 C_4 和 C_5 只通过 T_3 放电,则放电时间常数: $\tau = R_2(C_4 + C_5) / \beta_{T_2}\beta_{T_3}$

这里 β_{T_2} 为三极管 T_2 的放大倍数, β_{T_3} 为三极管 T_3 的放大倍数,因为 T_2T_3 均选择色点为绿色的管子,其放大倍数均大于80。故有

$$\tau = 1500 \times (47 + 10) \div (80 \times 80) \approx 13(\mu S)$$

$$\text{则 } \Delta t_3 = 2.3\tau \approx 30(\mu S)$$

$$\text{而 } \Delta t_2 \text{ 为 } 783\mu S$$

$$\text{可见 } \Delta t_3 \ll \Delta t_2$$

故能及时地封锁6116,并复位CPU,使RAM中的数据得到有效保持。

上电时图3所示电路的工作情况是这样的。当 V_c 电压上升尚未达到5V以前, T_1 保持截止, T_2T_3 保持饱和导通,则G1点和B点保持为“0”电平。从而6116被封锁,CPU被复位,RAM数据不会丢失。当 V_c 上升到5V后, T_1 导通, T_2T_3 截止,此时通过 R_4 给 C_4 充电,其时间常数:

$$\tau_4 = 47 \times 47 = 2209(\text{ms})$$

通过 R_6 给 C_5 充电,其时间常数:

$$\tau_5 = 10 \times 10 = 100(\text{ms})$$

可见当电源电压 V_c 达到5V后,6116的封锁先解除而CPU的复位信号后消逝,这样便不影响整机的正常运行。

本文所介绍的电路,已进入实际应用。曾进行过连续上万次人为上、掉电实验,它均能有效地保持RAM中存贮的数据。

武侯网络工程公司竭诚为您服务

本公司经营各种计算机及网络工程销售与维修业务,承接CAD、多用户系统、计算机网络的设计实施。凡经该公司出售的机器,均免费保修两年。

通讯地址:成都一环路南二段11号

总经理:况光名

电话:581252

邮编:610013

中文电脑打字摆擂台

为了推动普及型汉字编码输入方法的发展,挑选优秀的普及型汉字编码方法。中华侨情编委会组联部和桂林市肖码技术开发部订于1991年10月在桂林市联合举办普及型汉字编码方法擂台友谊赛,对前三名优胜者给予奖金,凡比赛名次超过肖码中文输入方法者,均给予物质奖励。欢迎海内外人士参赛、参观。

*同时展销肖码中英文电子打字机、词处理通信机、桌面印刷系统、汉卡、肖码神指。

*每月10日举办中英文电脑打字学习班。学习结束择优聘用和向有关单位推荐。

*承接电脑打字、价格同机械打字。

*肖码是简便易学快速的中文输入方法。详见《计算机世界》报1991年5月1日第5版、《人民日报》海外版1991年5月13日第1版的报导。其特点是:

1. 易学:采用汉语拼音、笔画知识,只要具有小学文化水平,几十分钟即可学会。

2. 易用:只用10个数字编码,单手击键,不用练英文指法。

3. 快速:字、词码长1~5位数字,平均每字1.5个数字,无重码,可高速盲打、听打,可达速记员的输入速度。

4. 面广:适用面广,肖码软件可挂接到各种汉字系统,在只有512K内存,无硬盘的微机中都能运行肖码软件。

5. 多用:在同一输入状态下,可输入中文、英文、数字、表格而不用按任何转换键。

乍看,肖码似乎是“天方夜谈”,但是,只要您花一小时看完肖码技术资料,亲自上机操作,您就会从心里发出:肖码确实易学、易用、快速、面广、多用。特别是肖码神指,为一双电子手套,手套的10个手指代表10个数字,一个或两个手指按任一物体即可输入一个数字或一个计算机命令。使用肖码神指,不用英文键盘,它使得枯燥的中文输入,变得如同做游戏一样轻松自如,彻底解决了计算机中文输入难的问题,包君一定会“不用不知道,一用丢不掉”。

*汇款5元,即寄肖码技术资料、比赛章程,招生简章及肖码系列电脑产品报价单。

*桂林市肖码技术开发部地址:广西桂林市七星路2号,邮码:541004 电话:442474 联系人:肖水清

*北京总代理:先锋集团华清电子工程公司,地址:北京市中关村丁字路口,电话:2564879。

《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