

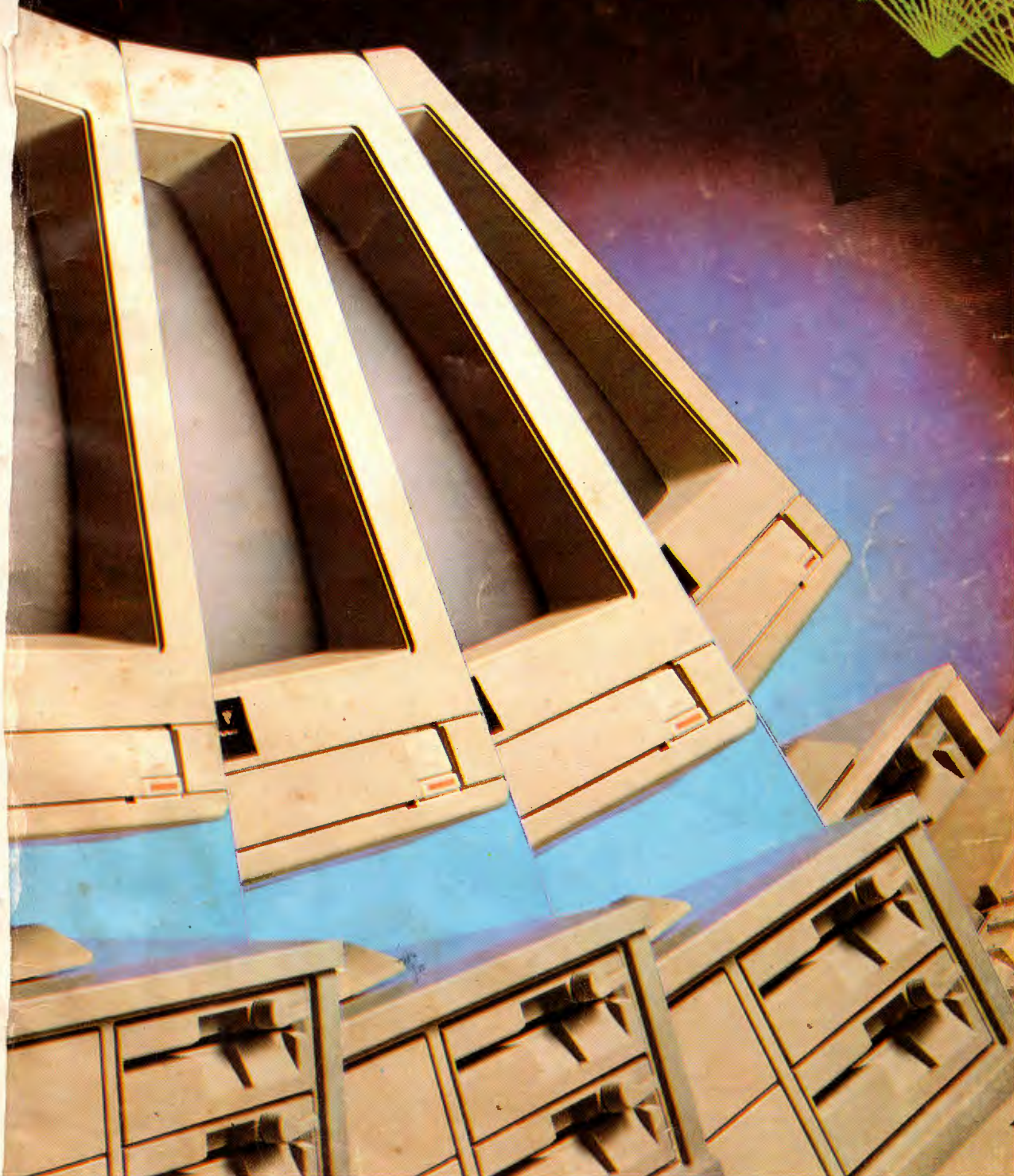
ISSN 1001-3695

B62

1992

计算机应用研究

APPLICATION RESEARCH OF COMPUTERS (计算机应用研究) 第5卷



《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 | 山東省計算中心 |
| 河南省計算中心 | 青海省測試計算中心 |
| 四川省電子學會 | |

《計算機應用研究》雜誌社董事會

董事長：周濟渝
董 事：唐 珍 孫傳江 陸慰梧
吳地興 鄭國基 陳建嶺
黎 蓉 閻長榮 黎曉常

《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 李文華
楊劍波 余 凱 劉啟茂

1992 年第 1 期（總第 45 期）

出版日期：1992 年 1 月

本期責任編輯：鄧偉文

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

山东省计算中心简介

山东省计算中心建于一九七六年,是山东省成立最早的计算机应用科学研究机构,隶属于山东省科学院。

山东省计算中心拥有较强的技术实力和丰富的设备资源。目前有高中级技术人员五十二名,初级技术人员五十余名;设有第一、二、三、四、五研究室、情报资料室、技术开发部、机房工程部;配有 IBM 4341 计算机、CALCOMP 大型绘图仪、TEKTRONIX—618 图形处理系统、APOLLO 和 SUN 工作站,并引进 REDUCE、ADINA、SAP5、SAP6 等应用软件,还配有各种类型的高档微型计算机。

山东省计算中心主要从事计算机应用系统分析、设计、开发、辅助设计(CAD)、数据检测及自控,并为科学计算、工程设计提供数学方法及计算机机时服务。多年来完成一批具有国内先进水平的研究课题。其中有数十项荣获国家部委、山东省和山东省科学院科学进步奖。同时完成了国家机关、省内外数百家企事业单位委托的技术开发性课题,取得了显著的社会效益和经济效益,受到各界人士的普遍好评。其中部分可供转让的科技成果有:1、BFC—1 型啤酒发酵罐微机温度控制系统;2、三维测力平台微机系统;3、工业厅局机关管理信息系统;4、办公自动化;5、企业管理信息系统;6、物资管理系统;7、计算机网络管理信息系统;8 造纸机石辊计算机辅助设计系统;9、实时销售系统;10、通用信息处理系统;11、武警被装业务管理系统;12、化工统计调优软件包;13、化工企业供销管理系统;14、化工设备信息管理系统;15、水电行业微机管理信息系统;16、TQC 应用软件;17、劳动合同制工人退休养老基金微机管理系统;18、个人收入调节税计算机辅助管理系统;19、劳资人事管理系统;20、金融信息管理系统。

山东省计算中心愿在计算方法研究、计算机软件开发、数据处理、过程控制、工业检测、电子产品研制、人员培训及机型选购、装机调试、机器维修等方面,与国内外企业厂家、科研机构、大专院校和学术团体,进行各种形式的技术合作、学术交流、友好往来,为计算机应用事业而努力奋斗。

通讯地址:山东省济南市经十路东首科院路

邮政编码:250014 电话号码:44236,615102—241



山东省计算中心计算机机房一瞥

2)计算数据是测植仪采集后加工取得的坐标点,无需难以精确测定的植面曲线上。

3)展平的植面不需要再作符植处理,即可作平面设计基片,同时提供准确的定位曲绕角,为后续部件曲绕处理奠定了基础。

4)全植面或半植面展平可以任选。

由于上述特点,本方法全面解决了制鞋 CAD 中三维数据向二维数据转换的技术难题,同时对制鞋手工设计中最带经验性操作的样片曲绕处理数学模型的建立,奠定了基础,这在国内还未见报导。加拿大 Cimtech 公司的 Descom 系统的二三维转换技术代表了当今同类系统的国际水平,本系统与之比较,不仅与 Descom 一样达到全植面展平,而且可作三维帮样展平,在技术上更胜一筹。

• 空间画笔三维款式设计。

手工设计方法之一是贴植法。设计人员在实际植面上设计款式,再经手工剥离下来展平成样片,经曲绕处理形成加工样片。在制鞋 CAD 中,目前国内已实现的三维设计方法有下列几种:

1)扫描法。在实物植面上画好款式,经三维数字化仪将款式线立体坐标送入计算机,由计算机完成二维样片转化。2)视图法。显示植形三视图于屏幕上,设计人员同时在三视图图上设计款式投影线,由计算机转化成空间曲线。3)查表法。用算法生成一定数量的植面三维点集,通过表映射建立植面投影二维点集与三维点集对应关系,使屏幕上拾取到的点转换成最接近的空间点坐标。扫描法对立体数字化仪功能要求较高,相应增大系统开销,同时扫描很费时间,设计效率不高。视图法用于象植面这样的异型几何曲面上,很难真实反映实际的空间曲线形态,这就要求设计人员除要有高度空间想象能力外,还要反复修改视图,才能达到设计要求,查表法在满足交互式设计要求时,除要较大空间开销外,同时所得三维坐标只是某种程度近似值,很难满足设计精度要求。

本系统提出了空间画笔概念,在曲面定位器(软件加平面数字化仪)控制下,画笔在植面三维造型图上连续移动,画笔的方向总保持与植面法线方向一致,将手工设计中植面的转动变成空间画笔的相对移动,极大增强了三维设计的真实感,深受设计人员欢迎。由于曲面定位器的作用,在三维设计完成后,立即取得平面展样,这更是已知方法中独一无二的。

• 实体造型。

作为一个好的 CAD 系统,总少不了实体造型内容。制鞋 CAD 的开发,主要就是为了加速款式翻新、缩短设计周期。款式设计好后,还要试做样鞋。因此,

在机助款式线设计完成后,立即显示成鞋试样特别重要。目前实体造型成功的例子,均在工程工作站或小型机上借助强有力的硬件作图功能完成的,但这在成本上还难以为我国制鞋厂广泛接受。本系统采用普通 386 微机,实现了多光源照射下彩色成鞋造型,并可自由旋转、换色和配跟,在低档设备上完成与高档设备效果相近的功能,大大提高了系统的性能价格比。

• 多层样片分解。

在手工设计中,当遇到样片互相重叠而形成多层样片时,分解样片既困难又繁琐。针对这种需求本系统首先提出并实现了多层样片设计与分解功能,极大提高了复杂款式的设计效率,同时将样片层次概念方便地引入成鞋造型中,达到多重覆盖显示的满意效果。

• 辅助设计功能配套实用。

整个制鞋设计包括若干设计环节,要真正达到提高辅助设计效率的目标,辅助功能配套实用是不可缺的条件。本系统提供的辅助设计功能覆盖了制鞋全部过程各个环节,与国内已开发同类系统比较,在此项指标上居于前茅。

• 设计数据库管理。

本系统提供设计数据库管理功能,可以对全部设计数据记录或文件进行拷贝、恢复、删除、查询、修改等操作,对几何和非几何数据作了较好的管理、同时提供式样评估、报表生成功能。

• 用户接口友好。

由于历史原因,目前我国大多数制鞋设计人员对于计算机还比较陌生,引入机助设计后,与传统设计方法比较,不可避免地会有一些较大改变(例如设计工具、设计操作对象、数量观念等)。这就更加要求系统具有良好的用户界面。本系统采用制鞋业专用术语在统一的汉化菜单控制下运行,无需直接使用任何操作系统命令。在运行中,有简明清晰的回显和提示信息,使设计人员随时知道他正在干什么,下一步该怎么办,以及放弃操作的 UNDO 功能,对设计人员容易应答出错的输入信息,均作错误检查,或自动排除,或纠正重输,避免了系统因误操作或误信息而中断的情况,保证系统正常运行,提供用户一个十分友善、简明可靠的操作界面。

三、结语

众所周知,由于我国人口众多,制鞋的产量在全世界占相当大的比重。但是,我国大中型制鞋厂面临两大困难。在出口方面,面对外国先进生产手段的挑战;在内销方面,面对个体制鞋业款式变换快的挑战。统计资料表明,国内一般大中型鞋(下转 52 页)

《计算机应用研究》第9卷第1期(总第45期)

目 录

软件篇

| | |
|-----------------------------------|--------------------|
| 用高级语言设计的数据库报表输出通用软件..... | 向南平(1)✓ |
| 用高级语言管理 dBASE Ⅲ 数据库结构..... | 向南平(4)✓ |
| 人体肾血浆流量微机计算方法..... | 张鸿鸣 刘铁军 田卫星 薛 赤(9) |
| 一种交互式快速原型图式语言——RPSGL | 马 江 黄 玲(11) |
| 选择组合查询的方法及其实现 | 田丽娃 刘鲜京(15) |
| 多窗口用户界面的设计与实现 | 史品志 刘文明 牛连江(17)✓ |
| 通用函数曲线作图程序 | 谢进一(19)✓ |
| 中文菜单的设计方法 | 曾向阳(21)✓ |
| IBM—FORTRAN 语言屏幕彩色字符显示功能的扩充 | 张 义(25) |
| 一个 Turbo—c 的多窗口大程序框架 | 唐常杰 韩仲清(27)✓ |
| 在西文 DOS 下显示汉字的一种方法 | 李日林(29)✓ |
| 动画显示程序设计及在模态分析中的应用 | 朱才朝(30) |
| DBASE Ⅲ 实现的中文屏幕游动显示 | 彭 立(32) |
| DOS 系统下的脱机输出 | 刘锦林(33) |
| FORTRAN 语言图形功能的扩充 | 邵庆良 沈尔秩(34) |

硬件篇

| | |
|-----------------------------------|-------------|
| 日历时钟芯片 M5832 与微机接口设计一种实现方法 | 卢正义(38) |
| 一种实用的双机并行通讯方法 | 周 立 王晓红(39) |
| PC 机与 TMS320 系列数字信号处理器的接口设计 | 王大可(40) |

系统篇

| | |
|-------------------------------|-------------------------|
| 单片机录像遥控播放系统的设计 | 陈惠生 侯建新 王德来 白智鹏 王 平(42) |
| 条形码技术在生产检测中的应用 | 姚景平 李 伶(46) |
| IBMPC—IBM PC 文件通信通用程序设计 | 郭继展(49) |
| 8031 单片机 DMA 功能的扩充 | 周鸣争(51) |
| IBM—PC/XT 中断扩展的具体实现方法 | 赖红威(53) |

维修篇

| | |
|----------------------------|-----------------|
| 微型机的故障维修技术 | 牛发海 张选阁 张双才(55) |
| 一种方便修复故障盘数据的实用程序 | 阮高华(57)✓ |
| IBM—PC/XT 硬盘不能自举维修一例 | 李洪涛(59) |
| PC/XT 软盘驱动器故障二例 | 周 力(59)✓ |

信息篇

| | |
|---------------------------------------|---------------------|
| 制鞋 CAD 开发应用刍议——兼谈 SMD—6410 技术特色 | 马在强 徐一清 张成武 刘 营(61) |
| 简讯 | (封二, 8, 50) |

APPLICATION RESEARCH OF COMPUTERS

VOL. 9. NO. 1 (Total 45)

CONTENTS

SOFTWARE

- The Common Software Designed by High—Level language for Data Base Report Xiang Nan Ping(1)
The Data Base Structure of DBASE ■ Managed by High —Level Language Xiang Nan Ping(4)
The Method for Computerizing Body Kidney Plasma Flow
..... Zhang Hong Ming ,Liu Tie Jun ,Tian wei Xing ,Xue Chi(9)
RPCGL;A Statecharts —Based Rapid Prototyping Graphic Language Ma Jiang , Huang Ling(11)
The Way and Realization of Selecting Combined Search Tian Li Wa ,Liu Xian Jing(15)
The Design and Realization for Multiwindow User Interface
..... Shi PinZhi,Liu Wen Ming,Niu Lian Jiang(17)
The Ploting Program for Common Functional Curve Xie Jin yi(19)
The Method for Designing Chinese Menu Zeng Xiang Yang(21)
Expanding the Function of Color Character by IBM— FORTRAN Language on the Screen
..... Zhang yi(25)
A Large Multiwindow Program Flowchart for Turbo—C Tang Chang Jie ,Han Zhang Qing(27)
A Method of Displaying Chinese Character on PC DOS Li Ri Lin(29)
Designing Cartoon Display Program and Used it in Modal Analysis Zhu Cai Chao(30)
Realizing Chinese Character Dynamic Display On Screen by DBASE ■ Peng Li(32)
The Off—Line Out Put in the DOS System Liu Jin Lin(33)
Expanding Graphic Function in FORTRAN Language Shao Qing Liang,Shen Er Chi(34)

HARDWARE

- Designing and Realizing Interface of Microcomputer and Clock Chip M5832 Lu Zheng Yi(38)
A Pratical Method of Two Commputer's Parallel Communication Zhou Li,Wang Xiao Hong(39)
The Interface of PC Computer and TMS320 Digital Signal Processor Wang Da Ke(40)

SYSTEM

- Designing Telecontrol Broadcast System of Recording Picture by Single Chip Processor
..... Chen Hui Sheng,Hou Jian Xin,Wang De Lai,Bai Zhi Peng,Wang Ping(42)
Application of Barcode Technology in Producing Field Yao Jing Ping,Li Ling(46)
Designing General File Communication Program of IBMPC— IBMPC Guo Ji Zhan(49)
The Extending Function of DMA for 80386 SCP Zhou Ming Zheng(51)
The Means of Realizing the Interruption—Expantion in IBM PC/XT Lai Hong wei(53)

MAINTENANCE

- The Technique of Maintaining Failure in the Microcomputer
..... Niu Fa Hai,Zhang Xuan Ge,Zhang Shuang Cai(55)
A Simple Utility Program of Recovering Data On Fault Disk Ruan Gao Hua(57)
A Maintenance Example about Hard Driver Self—Triggered Failure in IBM—PC/XT
..... Li Hong Tao(59)
Two Examples about hard Disk Driver Failure Zhou Li(59)

INFORMATION

- Discussing the Development and Application Technical Characteristic about SMD—6410
..... Ma Zai Qiang,Xu Yi Qing,Zhang Cheng Wu,Liu Ying(61)
News (Cover2,8,50)

用高级语言设计的数据库报表输出通用软件

中南工业大学资源开发工程系 向南平

摘要 任何一个实用的数据库管理系统,都必定要包含各类报表输出。但现有的各种用 dBASE III 形成的数据库管理系统中,几乎大部分都是由 dBASE III 系统本身的编程语句来实现报表输出,这使系统的通用性受到一定局限。本文给出一个用高级语言设计的、可在 DOS 操作系统下直接运行的数据库报表输出通用软件。

一、系统功能 为了保证系统的通用性,在系统的设计中,尽量考虑了在数据库报表输出中可能遇到的各种问题。系统功能如下:

1. 系统的工作文件是 dBASE III 的 DBF 数据文件,只需给出 DBF 数据文件名,系统就可自动从所给数据文件中获取其结构参数和数据;

2. 输出控制条件可选。可按起止记录号控制输出,也可按日期型字段的起止日期控制输出,也可按字符型字段的关键字符选择输出,还可按逻辑型字段的真假控制输出;

3. 任意字段的输出可选,序号栏也是可选栏;

4. 报表输出栏目名称、栏目宽度可任意调整;

5. 表头设计灵活,表头中各栏目名称可根据栏目宽度自动分行并居中排列,可生成下列三种形式的表头。

表头仅占一行:

(表一)

| | | | | | | | |
|----|------|-----|-----|------|----|----|------|
| 姓名 | 基本工资 | 书报费 | 洗理费 | 应发工资 | 水电 | 房租 | 实发工资 |
|----|------|-----|-----|------|----|----|------|

表头占二行:

(表二)

| | | | | | | | | |
|----|----|----------|---------|---------|----------|----|----|-------------|
| 序号 | 姓名 | 基本 工资 | 书报 费 | 洗理 费 | 应发 工资 | 水电 | 房租 | 实发工资 (元) |
|----|----|----------|---------|---------|----------|----|----|-------------|

表头占三行:

(表三)

| | | | | | | | | |
|--------|----|----------|---------------|---------|-----------------|----|----|-------------|
| 序 号 | 姓名 | 基本 工资 | 书报 报刊 费 | 洗理 费 | 应发 工资 (元) | 水电 | 房租 | 实发工资 (元) |
|--------|----|----------|---------------|---------|-----------------|----|----|-------------|

6. 可按用户要求生成如下含综合栏目的表头:

(表四)

| | | | | | | | | |
|--------|----|----------|------|----|-----------------|------|----|-------------|
| 序 号 | 姓名 | 基本 工资 | 付加工资 | | 应发 工资 (元) | 扣除部分 | | 实发工资 (元) |
| | | | 书报 | 洗理 | 付食 | 水电 | 房租 | |

7. 报表标题、副标题自动居中,其中报表副标题可选;

8. 对数值型字段,可选择是否需要页小计和总

计;

9. 可选择对任意个数值字段作横向统计(统计关系可为加减混合),统计栏目名称可选;

10. 对数值型字段的 0 值,可选择输出 0 或空格;

11. 数据行的行间横格线可选;

12. 报表可直接向打印机输出,也可向磁盘文件输出;

13. 可选择报表输出时左边空行数;

14. 可选择报表输出宽度限制,当报表总输出宽度超过限制值时,系统自动对其进行分解,分解后的报表各部分间衔接准确;

15. 系统自动分页,起始页号、页间空行数可选,还可选择页间暂停;

16. 自动打印页号和报表输出日期。

所有这些可选项,均保存在一个指定的报表格式文件中,第二次输出同样报表时,可不进入这些选择项,直接输出报表。此外,系统设计中还考虑了一定的排错和容错能力。

二、系统主控模块 由于系统是用高级语言设计而成,整个工作过程与 dBASE III 的管理系统无关,系统运行时直接调用 dBASE III 的 DBF 数据文件,并从 DBF 文件中获取其结构参数,因此,对 DBF 文件的结构的了解是设计本系统的先决条件,有关这方面的问题作者已有详细介绍(见参考文献),本文不再复述。以下是系统主控模块结构流程。

MAIN PROCEDURE

请用户输入工作库文件名(扩展名. DBF 可省略);

读工作库结构参数(字段数,记录长度,记录数,各字段名、属性、宽度、小数位);

请输入报表格式文件名;

IF 在指定路径存在该报表格式文件 THEN

读报表格式文件,并将各选择值赋给如下所述

相应变量;

ELSE

建报表格式文件,并将下列选择输入的结果存入报表格式文件;

请输入报表标题;

选择 1:报表要副标题吗?

IF 要 THEN 请输入报表副标题。

选择 2:向打印机输出或向文件输出?

IF 向文件输出 THEN 请输入文件名 ELSE 接通打印机。

选择 3:输入以什么为关键字控制输出?

CASE1:选记录号为关键字控制输出

请输入起止记录号

CASE2:选字符型字段内容为关键字控制输出

请输入关键字(长度不超过字段长度);

CASE3:选数值型字段内容为关键字控制输出

请输入数值的上下限(数值不限大小顺序);

CASE4:选日期型字段内容为关键字控制输出

请输入起止日期(日期不限先后顺序),(可只输入年或年月或年月日);

CASE5:选逻辑型字段内容为关键字控制输出

请输入逻辑关键字(T或F)。

选择 4:每页打印行数?

选择 5:数据行的行间是否要横格线?

选择 6:表格左边空?

选择 7:报表输出要限宽吗?

IF 要限宽 THEN 请输入限制宽度。

选择 8:打印起始页吗?

选择 9:页间空行(选 1 时页间暂停)?

选择 10:报表中数值型字段要页小计和总计吗?

选择 11:报表中数值型字段要横向统计栏吗?

IF 要 THEN 输入统计栏的组成关系(如表四的统计栏[实发工资],其输入形式为:2+3+4+5-7-8,式中数值对应于输出字段序号),系统根据该组成关系建立一个统计关系函数。

选择 12:报表中当数值为 0 时,要打印成空格吗?

选择 13:报表中有综合栏目吗?

IF 有综合栏目 THEN 请输入综合栏目个数;

FOR I=1 TO 综合栏目个数

请输入第 I 个综合栏的栏名;

请输入第 I 个综合栏由那几个字段组成,

如表三中有两个综合栏,输入形式为:3/4/5 和

7/8,

系统根据所输入的字符串分解出该综合栏所包

含的字段数及各字段序号;

NEXT I。

FOR I=1 TO 工作库字段数

显示第 I 字段字段名,字段属性,字段宽;

IF 第 I 栏不包含在统计函数关系式中,也不包含在综合栏目中 THEN

选择 14:该字段要输出吗?

IF 要输出 THEN

选择 14-1:要修改该字段的输出栏名吗?

IF 要 THEN 请输入栏名 ELSE 以原字段名为输出栏名;

选择 14-2:要修改该字段的输出宽度吗?

IF 要 THEN 请输入字段输出宽度 ELSE 以字段宽为输出宽度;

IF 输出宽度为奇数 THEN 输出宽度+1。

ELSE! 当字段包含在统计关系或综合栏目中时,该字段一定输出。

NEXT I。

关闭报表格式文件;

调报表预处理模块;

DO ! 以下为分页输出行报表

IF 总输出宽度>输出行宽限制 THEN

将报表以输出栏为单位分解成左右两部分,设左边栏目数为 N1,右边栏目数为 N2;

ELSE

令 N1=总输出栏目数,N2=0。

FOR I=1 TO 2

IF(I=1AND N2<>0)OR N2=0 THEN 输出前 N1 栏 ELSE 输出后 N2 栏;

IF 报表含综合表头 THEN

调综合表头输出模块:call print __title(page __ num,I,N1,N2);

ELSE

调普通表输出模块:call print __title2(page __ num,I,N1,N2);

调数据输出模块:call print __data(page __ num,I,N1,N2)

IF 页间空白=1 THEN

暂停,屏幕提示<C--继续,Q--退出>,并等待用户键盘响应;

IF 键入="C"OR 键入="c"THEN

继续输出;

ELSEIF 键入="Q"键入="q"THEN

终止输出,返回操作系统;

ELSE

按其它键系统不响应。

NEXT I

LOOP UNTIL 数据库记录输出完 OR 当以记录
序号控制输出时已输出至终止记录

END

三、报表预处理 报表预处理包括这几方面的内容:

1. 根据输出栏目的名称长度及报表中该栏目的实际输出宽度,计算各栏目名称应分几行输出,并由此确定报表表头的输出行数。

2. 为了满足中文报表栏名的输出,系统将输出栏宽取偶数,但若栏名是中西文字符混合时,就可能在栏名中出现中文字符被分割成两半而在两行上输出的情况。解决该问题的方法是先在栏名中截取一段相当于输出宽的部分,然后,判断在这段文字中所包含的西文字符(ASC II 值<160)的个数,当为奇数个时,在这段文字首加一个空格,这样就保证了栏名中的中文字符不会被错误的分割开来。

3. 确定各输出字段在数据库记录中的起始字节。这是因为,在数据输出时,系统是以记录为单位,每次从数据文件中读取等于记录长的字节数,并将其赋给一个字符串变量,而报表中各输出字段的内容,是根据该字段在记录中的起始字节和字段宽用取子串方式得到的。

四、普通表头输出模块

SUB print—title1 (page—num,I,N1,N2)

IF page—num=打印起始页(即第一页) THEN

IF N2=0 THEN

居中输出报表标题;

ELSE

I=1 时输出报表标题的前 K1 个字符,I=2 时
输出报表标题的后 K2 个字符。

IF (I=1 AND N2<>0) OR N2=0 THEN 输出前
N1 栏 ELSE 输出后 N2 栏;

IF 表头需三行输出 THEN

输出表头第一行;

对需三行或两行输出的栏目名称,在本行输出
其第一部分,

对仅需一行输出的栏目名称,在本行不输出;

输出表头第二行;

对需三行输出的栏目名称,在本行输出其第二
部分,

对需两行输出的栏目名称,在本行不输出,

对需一行输出的栏目名称,在本行输出,若栏名
长度小于输出栏宽时,居中输出;

输出表头第三行;

对需三行或两行输出的栏目名称,在本行输出

其剩余部分,

若该部分长度小于输出栏宽时,居中输出,

对仅需一行输出的栏目名称,在本行不输出;

ELSEIF 表头需二行输出 THEN

输出表头第一行;

对需两行输出的栏目名称,在本行输出其第一
部分,

对需一行输出的栏目名称,在本行输出,若栏名
长度小于输出栏宽时,居中输出;

输出表头第二行,对需两行输出的栏目名称,在
本行输出其剩余部分;

ELSE ! 表头仅需一行输出

输出表头各栏目名称,若栏名长度小于输出栏
宽时,居中输出,

END SUB

五、含综合栏目的表头输出模块 当报表表头
含有综合栏目时,系统固定表头按三行输出。与普通
表头不同之处在于,在综合栏目部分,表头第一行居
中输出综合栏目名称,第二行是输出一条横线,第三
行则输出包含在综合栏目中的各字段的栏名。其它
不包含在综合栏目中的各输出字段的栏目名称的处
理与普通表头一样。

本模块中比较难处理的是栏目中的各种制表符
的输出,普通表头中每输出一个栏名,同时就输出一个
制表符“|”,这里则要根据不同的情况输出六种形式
的制表符“|”、“┌”、“+”、“┐”、“└”、“┘”,限于篇幅,
本模块不再详细介绍。

六、数据输出模块 SUB PRINT __DATA (page
__num,I,N1,N2)

IF (I=1 AND N2<>0) OR N2=0 THEN 输出
前 N1 栏 ELSE 输出后 N2 栏;

设输出行计数器=表头所占行数;

IF 需要页小计 THEN 输出行计数器+3;

DO WHILE 输出行计数器值小于或等于设定页
长

DO WHILE 工作库还有记录可读

从工作库中读一条记录;

IF 该记录满足输出控制条件 THEN EXIT DO.

LOOP

从记录中截取各输出字段内容向报表中对应栏
目输出,并使输出行计数器+1;

若需要页小计和总计,则分别累加各数值型字
段的数值;

IF (N2=0 AND I=1) OR (N2<>0 AND I=2)
THEN

若需要横向统计,则按所给统计关系函数由对

应字段的值计算并输出统计值;

IF 需要输出行间横格线 THEN 输出行间横格线,并使行计数器+1;

LOOP

IF 需要页小计 THEN 输出页小计

IF 数据库记录输出完 OR 以记录序号控制出时已输出至终止记录 THEN 输出总计栏;

IF (N2=0 AND I=1) OR (N2<>0 AND I=2) THEN

输出报表生成日期、报表页号(page __ num);

END SUB

七、结束语 用高级语言设计数据库报表输出软件,避免了数据库编程语句中的一些缺陷,如输出行宽限制,输出行数限制等。更主要的是,用高级语言设计的报表输出软件,可以编译形成可执行的

EXE 文件,这样,既可在 DOS 状态下直接运行,也可移植到其它实用系统(包括用 dBASE II 形成的系统)中。

另外,由于 dBASE II 无图形功能,屏幕字符色彩和显示方式也太单一,因此用 dBASE II 无法产生让用户满意的屏幕效果。而高级语言(尤其是 Turbo Pascal 语言、C 语言)具有丰富的图形语句,可以使用窗口技术,设计多种不同色彩的下拉菜单,还可配以动画、音乐,使所设计的软件不仅具有通用性、实用性,而且新颖美观,更受用户喜爱。

参考文献 [1]向南平:“用 True BASIC 语言直接对 dBASE II 数据文件进行操作”,《计算机应用研究》,1991, NO. 4

[2]向南平:“用高级语言管理 dBASE II 数据库结构”,《计算机应用研究》,1992, NO. 1

用高级语言管理 dBASE II 数据库结构

中南工业大学资源开发工程系 向南平

摘要 在 dBASE II 系统的应用中,最使用户头痛的就是对数据库的结构进行修改。一般都是在 dBASE II 的点状态下用系统提供的 MODISTRU 命令来完成,当然,也可用程序方式解决。但无论那种方式,执行时都要相当谨慎,否则,就会导致数据的丢失。本文用高级语言设计了一个数据库结构管理系统,为解决这个问题提供一种新的有效途径。

dBASE II 数据库的结构管理,包括:显示数据库文件的结构,修改数据库文件的字段名、字段属性,调整字段宽度、字段顺序,增加或删除字段等。以下以 True BASIC 语言为例,介绍如何用高级语言来完成这些工作。

一、dBASE II 的 DBF 数据文件的结构

要用高级语言实现数据库文件的结构管理,对数据库 DBF 文件本身结构的了解是必不可少的。尽管该问题作者在文[1]中已有详细说明,但为了本文叙述的方便,也为了本文的完整性,以下再作一简要说明。

DBF 文件由两部分组成,即文件结构的说明部分和实际的数据部分。

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,为叙述方便,文中称它为第一类信息。第一类信息中各字节的内容及含义见下表。

| 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 |
| 5-8 | 记录个数 | DBF 文件中当前记录个数,低位在前,高位在后 |
| 9-10 | 首地址 | DBF 文件实际数据相对于文件开始位置的首地址 |
| 11-12 | 记录长度 | 存放记录长度,低位在前,高位在后 |

第 13 至 32 字节未用,全为 00H。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下表。

| 字节 | 1-10 | 11 | 12 | 13-16 | 17 | 18 | 19-32 |
|----|------|----|------|-------|-----|-----|-------|
| 内容 | 字段名 | 未用 | 类型标志 | 未用 | 字段宽 | 小数位 | 未用 |

文件结构说明部分除字段名和类型标志用

未订到本刊的订户,请直接同编辑部联系。

本刊

ASCII表示外,其余均为16进制数。结构说明部分以0DH、00H为结束标志。

DBF文件的实际数据紧接着0DH、00H之后存放。每个记录的第一个字节为记录删除标记,其后连续以ASCII码形式按顺序和长度存放各字段数据。数据部分以1AH(Cu+Z)结束。

二、DBF文件结构参数的读取和显示 无论要实现上述对DBF文件的结构管理中的那一部分,首先都必须获取工作库文件的结构参数。因此,先介绍读DBF文件结构参数子程序的设计。

```
sub read_dbf(#1, dbf_name$, zd_n, zkd,
rec_n, name$, kd(), sx(), xs())
input prompt"数据库文件名:", dbf_name$
open #1, name dbf_name$, org byte
read #1, bytes 32, rec1$
! 读第一类信息
let a2=unpackb(rec1$[9:9],1,8)
! 数据首地址低位
let a1=unpackb(rec1$[10:10],1,8)
! 数据首地址高位
let zd_n=(a1*256+a2-34)/32
! 计算字段数
let b1=unpackb(rec1$[8:8],1,8)
! 记录个数最低位
let b2=unpackb(rec1$[7:7],1,8)
! 记录个数次低位
let b3=unpackb(rec1$[6:6],1,8)
! 记录个数次高位
let b4=unpackb(rec1$[5:5],1,8)
! 记录个数最高位
let rec_n=b4+b3*16^2+b2*16^4+b1*16^6
! 计算记录个数
mat kd=zer(zd_n)
! 字段宽度数组
mat name$=nul$(zd_n)
! 字段名数组
mat xs$=nul$(zd_n)
! 字段属性数组
mat sx=zer(zd_n)
! 字段小数位组
let zkd=1
for i=1 to zd_n
    read #1, bytes 32: rec$
    let name$(i)=rec$[1:10]
    ! 读字段名称
```

```
let sx$(i)=rec$[12:12]
! 读字段类型
let kd(i)=unpackb(rec$[17:17],1,8)
! 读字段宽度
let xs(i)=unpackb(rec$[18:18],1,8)
! 读字段小数位
let zkd=zkd+kd(i)
! 统计记录长度
next i
print "数据库", dbf_name$, "的结构(字段名, 字段类型, 字段宽, 小数位):"
for i=1 to zd_n
    ! 显示各字段名, 属性, 字段宽, 小数位
    print "第", i, "字段:", name$(i), ",", sx$(i), ",", kd(i), ",", xs(i)
    if xs(i) <> 0 then print str$(xs(i)) else print
next i
print "数据库", dbf_name$, "的字段总数为", zd_n
print "数据库", dbf_name$, "的记录长度为", zkd
print "数据库", dbf_name$, "的总记录数为", rec_n
print "数据库", dbf_name$, "的建库日期为:",
print str$(unpackb(rec1$[2:2],1,8)+1900),"/",
print str$(unpackb(rec1$[3:3],1,8)),"/",
str$(unpackb(rec1$[4:4],1,8))
end sub
```

调用该子程序,就可获得所给数据库文件的结构参数,并显示在屏幕上。后面介绍的各功能模块都要调用该子程序,由该子程序传递的参数有:工作库文件名(dbf_name\$)、字段数(zd_n)、记录长度(zkd)、记录数(rec_n)、各字段名称(name\$(i))、各字段属性(sx\$(i))、各字段宽(kd(i))及字段小数位(xs(i))(i=1,2,...,zd_n)。

三、修改字段名和字段属性 字段名和字段属性的修改最为简单,因为它们仅涉及到文件结构的说明部分。先用字节文件的方式打开待修改的DBF数据文件,然后调用下面的子程序,就可在不影响DBF文件的数据的前提下同时完成对某字段(当然也可多字段)的字段名和属性的修改。值得一提的是,在dBASE II系统中这种同时修改是不允许的,因为这将导致数据的丢失。在下面的子程序中,order为待修改的字段序号,new_name\$为新的字段名,

property \$ 为新的字段属性。

```
sub change _1
call read __dbf( # 1, dbf __name $, zd __n,
zkd, rec __n, name $, kd, sx $, xs)
reset # 1, begin
read # 1, bytes 32 * order : rec $
! 读第一类信息和待修改字段以前各字段的说明
read # 1, bytes 32 : rec1 $
! 读待修改字段的说明部分
let name $ = (new __name $ & repeat $ (" ",
10))[1 : 10]
! 新名长度不足用空格补齐
if Ucase $ (property $) = "M" then
let rec $ = chr $ (131) & rec $ [2 : 32 *
order] & name $ & chr $ (0)
let rec $ = rec $ & Ucase $ (property $)
& rec1 $ [13 : 32]
else
let rec $ = rec $ & name $ & chr $ (0) & U-
case $ (property $) & rec1 $ [13 : 32]
end if
reset # 1 : begin
write # 1 : rec $
close # 1
end sub
```

子程序中的 IF 语句是判断修改后的字段属性是否为记忆型,若是则同时修改文件标识字,否则不修改标识字。(实际上完整的考虑应是看被修改后的库中是否包含有记忆型字段,限于篇幅,本文仅介绍最简单的情况。)

四、调整字段宽度 字段宽度的调整涉及到相应字段的结构说明部分的修改和数据的调整,见下面简要子程序。

```
sub change _2
call read __dbf( # 1, dbf __name $, zd __n, zkd,
rec __n, name $, kd, sx $, xs)
reset # 1 : begin
read # 1, bytes 32 : rec1 $
! 读第一类信息
read # 1, bytes 32 * (order - 1) : rec2 $
! 读调整字段之前的各字段说明
read # 1, bytes 32 : rec3 $
! 读调整字段的字段说明
input prompt "请输入新的字段宽度" : width1
call packb(width1 $, 1, 8, width1)
```

```
! 将整数压缩到 1 个字节中
let rec3 $ [17 : 17] = width1 $
! 替换字段宽度
if sx $ (order) = "N" then
! 当字段为 N 型时,输入小数位
input prompt "请输入字段小数位" : width2
call packb(width2 $, 1, 8, width2)
let rec3 $ [18 : 18] = width2 $
! 替换字段小数位
end if
let width __new. zkd + width1 - kd(order)
! 计算新库记录长度
call packb(c1 $, 1, 8, mod(width - new, 256))
call packb(c2 $, 1, 8, width - new - mod(width
- new, 256)) * 256)
let rec1 $ [11 : 11] = c2 $
! 替换记录长度的低位字节
let rec1 $ [12 : 12] = c1 $
! 替换记录长度的高位字节
open # 2 : name "FILE. ALT", create newold,
org byte
erase # 2
! 建临时文件"FILE. ALT"
write # 2 : rec1 $ & rec2 $ & rec3 $
! 向临时文件写调整后的内容
do
! 读结构说明部分的剩余部分并写入临时文件
read # 1, bytes 32 : rec $
if rec $ [1 : 1] = chr $ (13) and rec $ [2 : 2] =
chr $ (0) then exit do write # 2 : rec $
loop
for i = 1 to order - 1
let zkd1 = zkd1 + kd(i)
! 统计 order 字段之前的各字段宽度和
next i
! 以下按新的记录长度调整数据部分
set # 1 : recsize zkd
! 设置每次读取的字节数等于原库记录长度
for i = 1 to rec __n
! 读取原记录,改正 order 字段宽并写入临时文
件
read # 1 : rec $
let a $ = rec $ [1 : zkd1]
let b $ = rec $ [zkd1 + 1 : kd(order)]
let c $ = rec $ [zkd1 + 1 : kd(order) + 1 : zkd]
if sx $ (order) = "N" then
```



```

! N型字段在数值前加空格
let long=254+kd(order)
let b$=(repeat$(" ",254)&.b$)[long-
width1+1:long]
else
! 其它类型字段在字符后加空格
let b$=(b$&.repeat$(" ",254))[1:
width1]
end if
let rec$=a$&b$&c$
! 记录重新组合
write #2:rec$
next i
erase #1
! 清原文件内容
reset #2:begin
do while more #2
! 拷贝临时文件内容回原文件
read #2, bytes 2048:rec$
write #1:rec$
loop
write #1:chr$(27)
! 写 DBF 数据库文件结束字
close #1
unsave "FILE. ALT"
! 删除临时文件
end sub

```

五、调整字段顺序 限于篇幅,后面几个模块仅给出程序的简略结构流程,所叙述的有关操作可参见前面几个模块。

```

sub change _3
调 read _dbf 子程序,获取并显示原库的结构
参数
for i=1 to zd _n
输入新的第 i 字段为原库中第几字段
next i
建临时文件"FILE. ALT"
读原库的第一类信息,改写其中有关文件修改
日期的第 2,3,4 字节
将改写后的第一类信息写入到临时文件
按新的字段序列调整文件结构中各字段说明部
分的顺序
将改写后的数据文件字段结构说明部分写入到
临时文件
for i=1 to rec _n
逐条读取原库中记录

```

按新的字段序列调整数据记录
将调整后的数据记录写入到临时文件
next i

将临时库内容拷贝回原文件

end sud

六、给数据库增加字段

```
sub change _4
```

调 read _dbf 子程序,获取并显示原库的结构

参数

输入要增加的字段为第几字段,order

输入要增加的字段名称和属性,name\$,sx\$

if 新增字段为数值型字段 then

输入要增加的字段宽度和小数位,kd,xs

elseif 新增字段为字符型字段 then

输入要增加的字段宽度,kd

elseif 新增字段为逻辑型字段 then

设置字段宽为,kd=1

elseif 新增字段为日期型字段 then

设置字段宽为,kd=8

else(新增字段为记忆型)

设置字段宽为,kd=10

end if

建临时文件"FILE. ALT"

读原库的第一类信息,并完成下列修改:

改写其中有关文件修改日期的第 2,3,4 字节

改写其中有关实际数据相对于文件开始位置首地址的第 9,10 字节

(新库首地址=库首地址+32 字节)

改写其有关记录长度的第 11,12 字节。

(新库记录长度=原库记录长度+新增字段宽)

将改写后的第一类信息写入到临时文件

读原库的字段结构说明部分并在相应位置增加新字段的结构说明部分

将新的字段结构说明部分写入到临时文件

```
for i=1 to rec _n
```

逐条读取原库中记录

在原记录中相应位置按新增字段宽插入空格以形成新库的记录

将插入空格后的新记录写入到临时文件

next i

将临时库内容拷贝回原文件

end sud

七、从数据库中删除字段

```
sub change _5
```

调 read _dbf 子程序,获取并显示原库的结构

参数

输入要删除的字段为第几字段,order

建临时文件"FILE. ALT"

读原库的第一类信息,并完成下列修改:

改写其中有关文件修改的日期第 2,3,4 字节

改写其中有关实际数据相对于文件开始位置首地址的第 9,10 字节

(新库首地址=原库首地址-32 字节)

改写其中有关记录长度的第 11,12 字节

(新库记录长度=原库记录长度-kd(order))

将改写后的第一类信息写入到临时文件

按新的字段序列组合各字段结构说明部分,并写入到临时文件

for i=1 to rec _n

逐条读取原库中记录

在原记录中删除 order 字段的内容以形成新

记录

将新记录写入到临时文件

nest i

将临时库内容拷贝回原文件

end sub

八、结束语 上述子程序加上必要的屏幕菜单,就可形成一个实用的数据库结构管理系统。原程序经编译并最终形成 EXE 执行文件后,可直接在 DOS 系统下运行,也可在 dBASE II 系统中用 RUN 命令执行。

目前,dBASE II 系统在我国拥有众多的用户,本文给出的方法,将会有助于广大用户有效的提高其数据库系统的运行效率。

此外,作者还用高级语言设计了一个可由 dBASE II 的 DBF 文件直接输出报表的通用报表输出软件,将另文介绍(见本期第 1 页)。

参考文献

- [1] 向南平,"用 True BASIC 语言直接对 dBASE II 数据库文件进行操作",《计算机应用研究》,1991,NO. 4

食品罐头行业传佳音——杀菌釜微机监测控制系统显神通

四川省电子计算机应用研究中心集国内现有各杀菌釜控制系统之长处,研制开发出新一代《杀菌釜微机监测控制系统》,经过半年多的使用,证明该系统杀菌效果良好,对于保证杀菌质量,稳定杀菌工艺,提高罐头产品的合格率都有明显的效果,在国内同行业中处于领先水平,最近通过了四川省省科委主持的专家鉴定。

专家鉴定委员会一致认为:

(1)该系统在罐头杀菌过程中,从升温、恒温、反压降温等全部过程均实现了自动控制,并具有停电保护和数据自动恢复以及声音报警等功能,各项控制精度指标均达到设计指标(控温精度 $\pm 1^\circ\text{C}$,控压精度 $\pm 0.1\text{kg/cm}^2$,恒温时间精度 ± 0.5 秒)。

(2)该系统运行可靠,由于电脑与智能仪表对各项控制参数精度的提高,更符合杀菌工艺的要求,保证了产品质量(即色、香、味等感官指标)的一致性,所杀菌出来的罐头经卫生检验部门化验,完全符合国标 GB11671-89 罐头食品卫生标准。该系统还大大减轻了操作人员繁重的体力和脑力劳动,也大大降低了玻璃瓶的破碎率(由原来的 13.2%,降低为 10.9%),降低了产品成本,具有显著的经济效益。

该系统的研制成功,大大提高了产品合格率,保证了产品质量与安全生产,具有显著的社会效益和经济效益,标志着我国罐头食品生产跨上一个新的台阶。

(张钢 哲明)

旅游业计算机网络管理系统问世

为适应我国旅游业业务的不断拓展,由四川省电子计算机应用中心和西藏旅游总公司共同研制开发的旅游业计算机网络管理系统,目前在拉萨通过了西藏自治区科技局主持的专家鉴定。

旅游业计算机网络管理系统由总办、销售、财务、导游、车辆等多个子系统构成,通过 Novell 网,形成一个统一的整体,能广泛适应各大旅行社、各旅游公司业务工作的需要。

参加鉴定的专家认为:

该系统功能完善,基本覆盖了旅行社的主要业务工作,数据共享程度高,运行可靠,是一个先进实用的网

(下转 50 页)

人体肾血浆流量微机计算方法

吉林省计算机技术研究所 张鸿鸣 刘铁军 田卫星

白求恩医科大学 薛 赤

肾是泌尿系统最重要的器官。有效肾血浆流量(一般记为 $ERPF$)是反应肾脏血流动力学的一个重要指标,它可以了解肾脏的各种病理情况下肾血流的改变状态。

测定有效肾血浆方法很多,但是不管那种方法都要多次采血和收集尿液,都会给患者带来很大的痛苦和不便。因此,通过临床手段对有效肾血浆流量进行测定困难极多,不容易被所有患者所能接受。

由于新技术不断发展,尤其自从微型计算机问世以来,为人体有效肾血浆流量的测定开辟了新的途径。根据数学方法,应用微型计算机测定人体有效肾血浆流量更显示了其优越性。

人体肾脏清除率 C 系指在单位时间内肾脏将多少毫升血浆中的某物质清除掉,即为该物质的人体肾脏清除率。临床证明测定示踪剂(如磷碘马尿酸等)的清除率可以反应人体肾血浆流量。由于血液每次流经肾脏并非 100% 清除,因此,它反应的是有效肾血浆流量。

通过人体肾图来监测示踪剂在肾血浆中量的变化,这样可在记录仪上得到一条曲线,这条曲线被称为放射性肾图(简称肾图)。它不仅可反映示踪剂在肾内的变化,而且还反映肾外围组织及血管等的变化,此为基底。

本文就是将问题数学化,经过计算和数据处理来完成人体有效肾血浆流量的测定。由于我们运用了微机技术,利用微型计算机进行大量地计算和数据处理,大大地提高了有效肾血浆流量的测定准确率,也容易被大多数患者所接受。

一、建立模型。

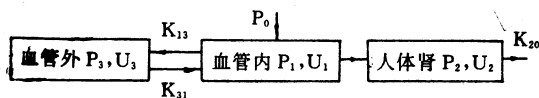
1. 房室模型。房室系指有一定容量的容器,里面装有某种服从物质守恒定律的单一物质,并假定任一时刻物质都均匀分布于容器之中。在分析问题,人们常常理想地把有机体看成一个或若干个有不同容量和不同浓度的房室,各房室间可以按一定规律进行物质交换,并可以直观地形成各类不同问题的物理模型和数学模型,从而可以知道物质在有机体内的理想动态分布规律是物质分布于系统的时间过程。

为了简化问题,常常把有机体理想化地看成一个有若干房室联系起来的系统。正如,对于能迅速分布于血浆及其它液体之间的物质就可以把有机体看

成为一个房室。如果同周围组织或血液及血液附近组织中的物质浓度有明显的不同,就可以把有机体看成是两个房室组成的系统。

建立模型就是把实际问题抽象成简化了的物理模型和定量了的数学模型。能否把实际问题简化成一个房室模型,关键在于:一是把实际问题简化成由若干个房室组成的系统;二是定出模型中应含房室的个数;三是定出房室间的联系方式。

2. 肾图房室模型。由于血液循环与肾本身的排泄是一个复杂过程,使运动机理难于分析,所以只有通过输入输出数据对结构进行识别,才能寻找同构或半同构的数学模型。为了能更准确地反映这个系统,经过反复实验,建立了这个系统的模型,如图一所示。



图一 三房室模型

这里的肾不是生理解剖的肾,而是排除肾所有血管后的肾,把它的血管归为一、三房室。其中 P_1 为所在子系统内, L 为磷碘马尿酸的量,单位是微居里 ($i=1,2,3$)。 K_{13} 为血管内到血管外 L —磷碘马尿酸的清除率,单位为每分毫升。 K_{12} 为血管内到肾的 L —磷碘马尿酸的清除率。从 K_{12} 的定义可知它就是肾脏清除率 C 。又由于肾脏清除率 $C=ERPF$, 所以就有 $K_{12}=ERPF$, 单位为每分毫升。 K_{20} 为排入尿中 L —磷碘马尿酸的清除率,单位为每分毫升。 P_0 为注入血管内的 L —磷碘马尿酸的量,单位为微居里。

根据物质守恒定律,并假定在某一时刻 L —磷碘马尿酸在各房室内均匀分布,则我们可以得到房室模型状态方程为:

$$\begin{cases} dP_1/dt = -K_{12}(P_1/U_1) - K_{13}(P_1/U_1) \\ \quad + K_{31}(P_3/U_3) \\ dP_2/dt = K_{12}(P_1/U_1) - K_{20}(P_2/U_2) \\ dP_3/dt = K_{13}(P_1/U_1) - K_{31}(P_3/U_3) \end{cases} \quad (1)$$

$$P_1(0) = P_0, P_2(0) = 0, P_3(0) = 0$$

设 $P_i = X_i (i=1, 2, 3)$, X_i 的单位为微居里, $K_{11}/U_1 = a_{11} (i, j=1, 2, 3)$, a_{11} 的单位为 1/分。并代入(1), 便得简化方程

$$\begin{cases} dx_1/dt = -a_{12}x_1 + a_{13}x_1 - a_{31}x_3 \\ dx_2/dt = a_{12}x_1 - a_{20}x_2 \\ dx_3/dt = a_{13}x_1 - a_{31}x_3 \end{cases} \quad (2)$$

$$x_1(0) = P_0, x_2(0) = 0, x_3(0) = 0$$

二、计算方法 由常微分方程定理可知, 应用特征根法解上述常微分方程(2)便可得出按房室分的三条曲线 $x_1(t)$, $x_2(t)$, $x_3(t)$ 。但由于 a_{ij} 未知, 故须用数理统计方法予以确定。经实验比较选择了单纯形搜索法和最小二乘法确定 a_{ij} 。

1、经画图可知肾图曲线是由 $x_1(t)$ 、 $x_2(t)$ 和 $x_3(t)$ 三条曲线叠加而形成的。其中 $x_1(t)$ 为血管内 L——磷碘马尿酸随时间变化曲线; $x_2(t)$ 为肾内 L——磷碘马尿酸随时间变化曲线; $x_3(t)$ 为血管内外 L——磷碘马尿酸随时间变化曲线

从而表明 $x_1(t)$ 和 $x_3(t)$ 是血管内外及非肾组织曲线, 也就是本底曲线的分解。因此根据反复实验选取了 $h(t) = 0.05(x_1(t) + x_3(t)) + 0.5x_2(t)$ 作为理论输出函数, 与肾图曲线拟合。那么 $0.05(x_1(t) + x_3(t))$ 就作为本底曲线, 目标函数为

$$I_j = \sum_{i=1}^N (h_i(t_i) - h^1(t_i))^2 \quad (3)$$

其中 $h_i(t_i)$ 是第 j 个点第 t_i 时刻搜索出的输出函数值。 $h^1(t_i)$ 为第 t_i 时刻从肾图上所测值, N 为采样总个数。

$$\text{当均方差为 } \sum_{j=1}^{M+1} (I_j - \bar{I})^2 < \varepsilon \quad (4)$$

时为一次搜索结束。其中 M 为参数个数, \bar{I} 为 I 的平均值 ($j=1, 2, \dots, M+1$)。

2、同时选若干个参数点, 设有 M 个参数要搜索, 这里选 $M=2$ 。任选一点作为初始点 A , 按给定边长 a 求得另外两点 B, C , 共有 $M+1=3$ 个点, 构成单纯形。若每一点为二维向量, 用最小二乘法可以得出目标函数为

$$\begin{cases} I_A = \sum_{i=1}^N (h(t_i) - F(t_i))^2 \\ I_B = \sum_{i=1}^N (h(t_i) - F(t_i))^2 \\ I_C = \sum_{i=1}^N (h(t_i) - F(t_i))^2 \end{cases} \quad (5)$$

其中 h, h 和 h 分别是 A, B 和 C 的函数, $F(t)$ 是理论函数。

这里第一步先比较 I_A, I_B 和 I_C 的大小, 找出其最小者, 为此可设 I_B 为最小者。第二步求 B 的反射点 B' ; 如果 $I_{B'} < I_B$, 则以 B' 代替 B 。按(5)式求 $I_A, I_{B'}$ 和 I_C , 再返回第一步进行比较; 如果 $I_{B'} > I_B$, 则沿反射方向对搜索不利, 故应向回收缩, 得点 B'' , 以 B'' 代替 B , 再返回第一步比较, 如果 $I_{B''}$ 仍最大, 则向 I 最小点假设 A 点压缩, 可得 A, C' 和 B'' 构成新的单纯形形返回到第一步, 直到均方差

$$\sum_{j=1}^{M+1} (I_j - \bar{I})^2 < \varepsilon$$

为止。这个反复过程被称为单纯形搜索法, 最小 I 的点为最佳点。

在 $K_{12} = a_{12}U_1$ 求出 a_{12} 。为此, 选一组参数为 a_{12}, a_{13} 和 a_{20} , 然后用单纯搜索法反复搜索, 直到拟合出误差最小的参数为止, 此时 a_{12} 就是所要求的。

3、解微分方程组。 首先把(2)式化为

$$\begin{cases} X(t) = AX(t) \\ X(0) = D_0 \end{cases}$$

$$\text{其中 } A = \begin{bmatrix} a_{12} + a_{13} & 0 & a_{31} \\ a_{12} & -a_{20} & 0 \\ a_{13} & 0 & -a_{31} \end{bmatrix}$$

$$D_0 = \begin{bmatrix} P_0 \\ 0 \\ 0 \end{bmatrix}, \quad X(t) = \begin{bmatrix} X_1(t) \\ X_2(t) \\ X_3(t) \end{bmatrix}$$

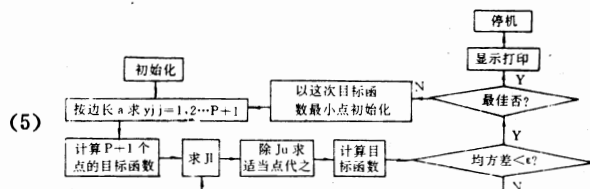
在 $D > 0$ 和 $D < 0$ 两种情况下, 用特征根法进行求解, 可分别得出通解

$$X(t) = \varphi(t) \cdot C$$

其中 $\varphi(t) = [ue^1 ve^2 we^3]$, 当 $t=0$ 时, $X(0) =]\varphi(0) \cdot C$ 可定出 C 值和 $X(t) = \varphi(t) \cdot C$, 其中 $(t) = \varphi[X_1(t)X_2(t)X_3(t)]$

当 $t=0$ 时, $X(0) = \varphi(0) \cdot C$ 可定出 C 值。因此, 便可由初值求其特解, 经过计算便可得出(2)式的解。

三、程序框图 根据上述计算过程和建立计算方法, 我们便可以画出程序框图, 并用 C 语言编写微机程序。仅经过 1—2 次搜索就可以求出最佳参数。其总框图如图二所示。



图二 程序总框图

四、讨论 由于肾图上得到的是脉冲数,而 $h(t)$ 的单位是微居里,与肾图进行拟合,则要乘上一个比例系数,即为

$$P_0/K = (1 \text{ 个微居里单位})/600$$

也就是一个微居里单位 $= (P_0/K) \times 600$, 其中 K 是注入 P_0 时肾的脉冲数(cmp)。一般当 $P_0 = 2$ 个微居里时, $\text{cmp} = 20000 \sim 50000$, 正常人注入 P_0 时肾的脉冲数是这个数字的 20%~25%。而进入每个肾只有 10%~12.5%, 所以 $K = 2000 \sim 3125$, 每一批药的 P_0/K 是一个常数。600 是肾图坐标上一个大格的脉冲数。

因此,每个状态变量 X_i 都要乘上一个因子 $P_0/K \times 600$, 经过整理,这时便可以得到 $K_{11} = U_1 a_{11} \cdot$

$(P_0/K) \cdot 600; K_{12} = U_1 a_{12} (P_0/K) \cdot 600; U_1 = 9000 \text{ ml}$, 则 $\text{ERPF} = K_{12} = a_{12} (P_0 \times 9000 \times 600)/K$ 。剂量是 ml/分。通过多次验证表明,用这种方法计算出的 ERPF 与临床的 ERPF 相吻合,正常人的 ERPF 应当为 600~750ml/分。

总之,由于计算 P_0/K 没有考虑 P_0 的离散情况,以及注入示踪剂时有注入血管外造成漏血的可能,所以计算时所有的 P_0 与实际的 P_0 有偏差,这也就导致出一般计算出的 ERPF 将有偏差。我们建立的数学模型与实际肾图曲线并不完全同构,三条指数曲线拟合相当困难,搜索出的 K_{12} 的精度也会受到影响。数据采样是肾图坐标纸上目测得到的,也肯定有偏差,自然精度也会受到一定影响。

一种交互式快速原型图式语言——RPSGL

武汉大学软件工程研究所 马江 武汉大学计算机科学系 黄玲

摘要 软件原型开发通常需要原型语言的支持。本文提出了一个基于 statecharts 的快速原型图式语言 RPSGL,并介绍了在软件原型开发环境(SPE)中实现的支持该语言快速生成原型的图形工具。该语言能较好地描述复杂系统动态行为信息,强有力地支持信息系统原型快速生成。

一、引言 快速原型方法是针对传统生命周期方法的不足而提出的。其基本原则是尽可能早地在系统开发的初期阶段,发现潜在的错误并立即修改,以小的代价开发高质量的产品。快速原型方法通常需要一个合适的软件原型开发环境的支持。为了方便原型设计者/用户快速生成、使用评价原型,在环境中提供一种原型语言来描述系统的动态行为和功能,是至关重要的。为此我们在集成化软件原型开发环境 SPE(Software Prototyping Environment)中,设计实现了一个基于 Statecharts[1]的支持信息系统原型的快速开发图式原型语言 RPSGL(A Statecharts - base Rapid Prototyping Graphic Language)。


本文首先提出该语言的基本成分,然后介绍支持该语言开发原型的图形编辑工具和图形解释工具,最后给出了利用该语言进行原型开发的过程模型。

二、RPSGL 图式语言的基本成分 statecharts

是以色列魏兹曼科学院的 D. Harel 于 1985 年提出的一种适于描述信息系统复杂行为过程的、基于状态的图示方法。由于它在刻画系统功能,尤其是在刻画动态行为方面很有优势,且其图形化表示形式已经过严格的理论验证。因此,我们针对 SPE 环境的特点选择了 statecharts 作为我们快速原型语言的基础。

并根据需要作了相应的修改,其基本成分如表 1 所示。

表 1. RPSGL 的基本图形元素

| | | |
|---|------------|------------------------|
|  | State | 状态或子状态 |
|  | Transition | 转换同标号一起用来描述状态间的转变及语义 |
|  | Andline | 用来描述并发状态 (concurrency) |
|  | Default | 缺省状态,用来描述初始子状态 |
|  | History | “历史”,记录最高层最近被访问的状态 |
|  | DHistory * | “历史”,记录最底层最近被访问的状态 |
|  | Diamond | 钻石标,用于隐藏状态名或转换标号 |
|  | Stub | 槽,用来在聚合状态时代表某一子状态 |
|  | Condition | 条件分枝结点 |
|  | Switch | 选择分枝结点 |
|  | junction | 连接结点 |

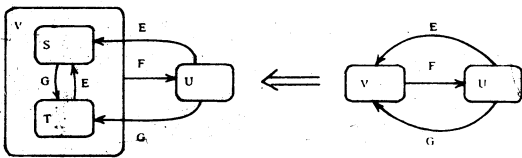


图1 状态的“异或”分解

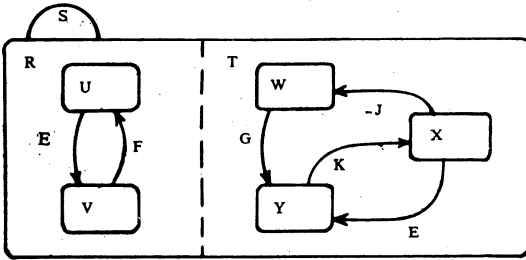


图2 状态的“与”分解

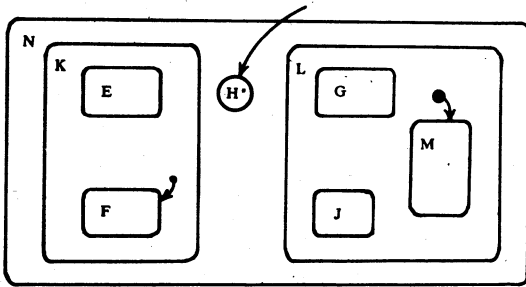


图3 状态的缺省和历史

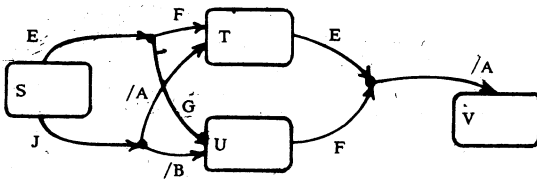


图4 连接节点

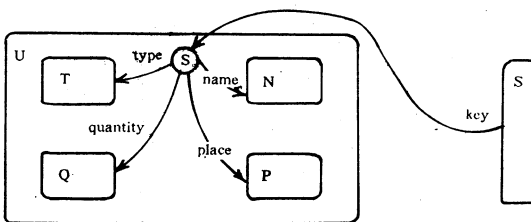


图5 选择节点(S)

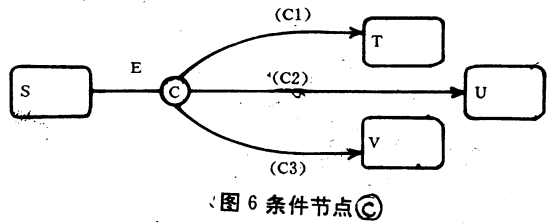


图6 条件节点(C)

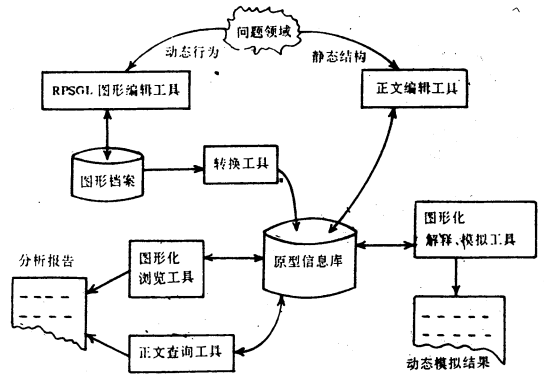


图7 RPSGL环境的组成

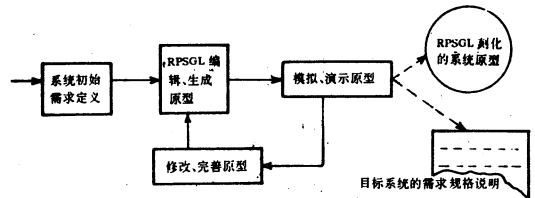


图8 利用RPSGL进行原型开发的过程模型

(1) 状态, 子状态

状态是 RPSGL 的核心成分, 这里状态的含义可同有穷状态机中的状态, 作同样理解。状态可以进行“异或”(XOR)分解和“与”(AND)分解, 变成复合状态。复合状态是由若干个子状态组成, 原则上, 复合状态中子状态的个数和状态层次没有限制, 这样体现了自顶向下、逐步细化的思想。原子状态是用来模拟一个实际的物理活动, 是不可再分的状态。

(2) 转换

转换, 用来描述状态之间的变迁。转换上的标号 (Label) 包含三方面的信息:

- (a) 激发转换的刺激事件 (Stimulus Event)
- (b) 转换发生必须满足的条件 (Condition)
- (c) 伴随转换发生而同时产生的动作 (Action)

一般, Label 的格式为 $S(C)/A$ 。其中 S、C、A 均可以为空, 但 S、C 不能同时为空。S 为空时, 只要 C 满

足,则发生转换,C为空时,只要S出现,转换也发生。通过转换可以实现状态间的通讯。另外,转换的起始状态和目标状态还可以是不同层次的状态。转换实际上体现了RPSGL的语义及控制机制。

(3)层次

RPSGL利用状态的“异或”分解、刻画状态间的层次关系(由状态父女关系构成)。RPSGL的层次概念,体现了抽象(聚化 Cluster)和求精(精化 Refinement)的思想,提供了处理复杂问题的手段。如图1所示,状态V是状态S,T的抽象。

(4)并发(正交性)

RPSGL利用状态的“与”分解,刻画子状态间的并发关系,提供了描述系统的并发生为的手段。如图2所示,状态R,T是状态S的两个并发子状态。

(5)缺省和历史状态

缺省状态一般由用户指定,它相当于有穷自动机中的初始状态。如图3中的状态M和F。

历史状态,记录状态的历史信息,根据它系统能返回最近被访问的状态。(H)记录最高层子状态入口,(H₁)记录最低层子状态入口,如图3所示,,H*能使系统进入最底层子状态(E,F,M,G,J)中的最近被访问的一个。若不是(H₁),而是(H),则系统进入最高层子状态(K,L)中的一个,按缺省状态进入状态M或F。

(6)附加成分

为了增强描述能力,成为转换弧的长度及标号中的重复部分,RPSGL提供了一个附加部分,如连节点②,选择点③,条件节点④等等。(参见图4,图5,图6)

三、支持RPSGL开发原型的环境及工具

基于上述的REPSGL的基本图式语言成分,我们在一个集成化软件开发环境SPE中,开发实现了该语言的图形编辑工具及图形解释模拟工具,以便有效地支持RPSGL进行快速原型的产生、模拟演示及修改完善的全过程。

3.1 SPE环境的组成

集成化软件原型开发环境SPE,是我们在VAX station II/GPX图形工作站上研制开发的一个支持面向对象的软件信息系统原型开发的环境。其组成如图7所示。

3.2 RPSGL图形编辑工具

RPSGL图形编辑工具是一个面向上述语言基本图形成分(简称图元),基于鼠标选择、操作图元,进行自动隐式的RPSGL语法检查,提供丰富的图形编辑功能的工具。其主要功能如下:

(1)图形存储管理功能:

对上述RPSGL图示语言建立图形档案,由统一的存储、管理机制进行存储和管理,图形档案是在图形编辑过程中由RPSGL图形编辑工具自动建立并修改的。对使用者而言,图形档案是透明的。

(2)作图功能

支持RPSGL图元的快速生成。主要包括:(a)生成状态/子状态;(b)生成并发状态;(c)生成状态活动名;(d)生成转换弧;(e)生成转换标号;(f)定义缺省状态;(g)定义历史状态;(h)生成附加成分等等。

RPSGL的所有图形生成,均采用鼠标选择控制关键点决定每个图形元素的构成,自动生成,如状态由两个关键点构成,转换由一组关键点构成。在作图过程中,按鼠标左键进行选择关键点,中间键放弃(UNDO)前次所选关键点,右键确认。在按下右边键确认以后,图形编辑工具严格地按RPSGL语法进行自动语法检查,提示错误信息。

(3)图形编辑功能

对RPSGL描述的图形提供灵活、方便的编辑功能,主要包括有:移动,复制(包括全局复制,局部复制),删除,重置大小,信息隐藏(将状态活动名或转换称号隐藏而用“◇”标记),信息再现(将“◇”隐藏的信息再现)。

所有的图形编辑过程,均由鼠标选择相应的图形进行交互式编辑,并进行全屏幕的自动RPSGL语法检查。若发现错误,可按中间键放弃所做操作,也可继续移动鼠标,直到没有错误、满意为止。

(4)图形浏览功能

直观方便地为用户提供了对RPSGL所描述信息的宏观或微观的了解,全局或局部的浏览,主要包括:

(a)精化:在显示窗口内,显示低一层的状态

(b)聚化:在显示窗口内,显示高一层的状态

(c)放大:按一定比例放大窗口内的图形

(d)缩小:按比例缩小图形

(e)复原:按原来的尺度显示图形

(f)画面移动:将整个画面在显示窗口上、下、左、右平移,浏览。

(5)图形区选择:对准备编辑图形,设置图形区域,以便编辑操作对整个图形区域中所有图形元素进行。

(6)正文编辑:对状态的活动和转换标号中的刺激事件、条件、动作等功能性描述进行说明、定义。目前,这些功能性描述均由C语言支持。

(7)读图:将存放在外存的已定义好的RPSGL图形读入。

(8)写图:将已编辑好的 RPSGL 图形写到外存存储。

(9)SHELL:直接转入 SHELL 窗口,可方便地调用 SHELL 命令,进行相关的处理。

(10)刷新:刷新当前的图形档案。

3. RPSGL 图形解释、模拟工具

利用上述图形编辑工具,编辑好的 RPSGL 描述的目标系统的原型可能还存在的一些问题:(1)语义上有错误;(2)不符合用户的需求。为此我们提供了 RPSGL 语言的解释执行工具,来模拟、演示原型的动态行为,以便及时发现错误、修改错误,完善原型,最后得到满足用户需求的原型(系统的规格说明)。

RPSGL 的图形解释、模拟工具主要具有如下的功能:

(1)解释状态中的活动:每个状态中都定义了一些活动,当解释器进入一状态时,就模拟该状态中的活动。

(2)实现状态间的转换:当一状态解释完毕时,解释器检查该状态所有可能发生的转换(包括在刺激信号缓冲区寻找适当的刺激信号,及判断转换的条件是否为真)。选择满足条件的转换执行。转换发生时,若有动作则执行该动作。

(3)识别状态层次的变化:为了自然、形象地解释,解释器对所有状态间的层次关系都作了相应的记录。

(4)模拟子状态的并发活动:状态的“与”分解导致正交子状态的出现。正交子状态的活动是并发进行的。我们充分利用了 Ultrix 操作系统提供的创建进程,进程间信号传输以及进程间通讯等机制,来模拟、演示正交子状态间的并发活动。

(5)广播通讯:利用刺激信号缓冲区,实现所有正交子状态间的信息共享,达到广播通讯的要求。

(6)语义方面的检查:主要针对在演示执行原型过程中,发现的某些异常情况,如转换的不确定性、死锁等进行语义检查。若发现错误,需利用图形编辑工具进行修改。

在图形解释工具模拟、演示 RPSGL 原型过程中,使用者相当于真实系统的外部环境。如果由软件、硬件实现的状态活动越多,由人模拟的状态活动越少,这个原型就越接近于真实的系统。

通过一个友好的图形多窗口用户界面,用户/设计者可方便灵活地使用 RPSGL 图示语言进行图形直接操作,图形会话,交互式地设计开发应用系统原型。

四、利用 RPSGL 进行原型开发的过程模型

使用 RPSGL 进行原型开发的过程模型可由图 8 所示

开发过程的每个阶段具体描述如下:

①系统初始需求定义:在该阶段,要求开发者相当程度地理解用户的要求。开发者将要开发的应用系统功能/性能,有一个大致的设计方案。

②原型定义生成:对上述设计方案,使用 RPSGL 图形编辑工具,编辑产生用 RPSGL 描述的原型。

③模拟、演示原型:利用图形解释工具对已编辑好的 RPSGL 原型,进行模拟、演示。在解释执行过程中,用户根据他所期望的系统行为评价原型的实际行为,且将反馈信息提供给开发者。

④修改、完善原型:根据解释过程中用户的反馈,利用 RPSGL 图形编辑工具进行修改、完善原型,重复③,④,直到用户认为该原型已充分体现了想象中系统的主要功能为止。

⑤需求定义的完善:对最终得到的 RPSGL 原型进行分析,在此基础上,修订初始需求,并产生形式化规格说明。

五、结论 对复杂系统进行原型设计、开发,往往需要有一个能描述系统动态行为的原型语言支持。为此本文提出了一个基于 statecharts 的快速原型图式语言 RPSGL,开发实现了该语言的图形编辑工具和图形解释工具,支持利用该语言进行快速原型开发。该语言的特点是:形象、直观、自然、灵活,易于使用、易于修改,具有强有力地描述系统动态行为能力。因而,可方便地对系统的行为、功能或用户界面作原型。目前,我们已在一个集成化软件原型开发环境 SPE 中,利用该语言,成功地支持了许多应用领域原型的设计开发,显示出该语言的巨大潜力。

参考文献

- [1] Harel, D. "Statecharts: A Visual Formalism for Complex System," Science of Computer programming(8) 1987.
- [2] "The Operational Versus the Conventional Approach to Software Development" Comm, ACM, Vol. 29, Feb 1984.
- [3] "The Language of STATEMATE", Technical Report, i _ Logic Inc, Burlington, May 1987.
- [4] 陈冀军, "SBOM——一个用于软件原型的操作模型", 武汉大学硕士论文, 1988, 6.
- [5] 马江, 何克清, "The Design and Implementation of an Integrated Software Prototyping Environment (SPE)". Proc. of the ICYCS'91, July 1991.

选择组合查询的方法及其实现

山东省计算中心 田丽娃 刘鲜京

摘要 本文提出了选择组合查询的方法,并在 FOXBASE 环境下实现了该方法。该方法实现简单,灵活方便,通用性强,有效地解决了由用户自行选择组合查询问题。

一、问题的提出 在数据库管理系统(DBMS)下工作,查询即为关系的选择输出或选择投影输出。一般在设计程序时,程序员需根据用户提出的需求,编写相应的查询程序来实现用户的要求。若用户事后要求改变查询需求,则要改变程序中的查询条件。采用选择组合查询方法,可由用户自行定义查询条件,即由用户根据工作要求,通过直观的输入模式引导用户做出查询选择,然后由系统自动生成查询条件,而无需程序人员再变更程序。该方法可作为一个标准模块,在信息管理系统(IMS)中调用。

二、设计思想及其数据结构 所谓选择组合查询是指选择操作的命题公式是系统根据用户的输入自动组合生成的。例如在 FOXBASE 环境下,Display、List、Locat 等命令均可用来选择查询,关键在于如何生成命题公式,即条件表达式。这就是本方法所需解决的根本问题。设有数据库 A,其属性集合为:

$A = \{A_1, A_2, \dots, A_n\}$ 要对 A 进行随意的组合查询。因条件表达式的一般形式为:

(条件表达式) ::= <逻辑运算符> <字段名变量>

<关系运算符> <输入变量>

所以,令逻辑运算符为 L,字段名变量为 C,关系运算符为 R,输入变量为 B,条件表达式为 F,则有:

$$F = LCRB \dots\dots\dots (1)$$

又设 L 的集合 $L = \{L_1, L_2, \dots, L_m\}$

C 的集合 $C = \{C_1, C_2, \dots, C_m\}$

R 的集合 $R = \{R_1, R_2, \dots, R_m\}$

B 的集合 $B = \{B_1, B_2, \dots, B_m\}$

则(1)式可改写为:

$$F = L_1 C_1 R_1 B_1 L_2 C_2 R_2 B_2 \dots\dots L_m C_m R_m B_m \dots\dots (2)$$

其中, L, R 均为确定的, $C \in R$ 也是确定的。B 为用户输入模式,供用户选择输入。设用户选择输入为 BK_1, BK_2, \dots, BK_n 的值 ($n \leq m$),则系统自动组合生成条件表达式:

$$F = L_1 C_1 R_1 BK_1 L_2 C_2 R_2 BK_2 \dots\dots L_n C_n R_n BK_n \dots\dots (3)$$

并执行选择输出操作。

由此可见,选择操作的条件表达式不是确定的,

它将因用户输入不同而变化。输入模式一旦确定, F 也就随之确定。因此我们把这种选择条件表达式组合生成的查询称之为“选择组合查询”。

例如:一个被查询的数据库 AA,其数据结构为:

```
Strucure for database, A: \AA. DBF
Number of data records:      7
Date of last update : 11/30/90
Field  Field Name  Type      width  Dec
  1  A1          Character  6
  2  A2          Character  2
  3  A3          Numeric   3
  4  A4          Date      8
  5  A5          Character  8
  6  A6          Character 30
* * Total * *                58
```

显然,在组合条件表达式 F 时,不但要给出 L, C, R, B, 还要再给出字段类型。由此我们需定义字段类型。字符变量集合:

$$T = T_1, T_2, \dots, T_m$$

以便在组合条件表达式 F 时,可根据不同的数据类型进行相应的处理。这样一来,令 F 的初值为“T”,对任一非空的 $B_i (i=1, 2, \dots, m)$ 在组合条件表达式时,有:

$$\text{当 } T_i = 'N' \text{ 时, 则 } F = F + L_i + C_i + R_i + V.1(B_i)$$

$$T_i = 'C' \text{ 且 } R_i = '$' \text{ 时, 则 } F = F + L_i + B_i + R_i + C_i$$

$$T_i = 'C' \text{ 且 } R_i \neq '$' \text{ 时, 则 } F = F + L_i + C_i + R_i + B_i$$

$$T_i = 'D' \text{ 时, 则 } F = F + L_i + C_i + R_i + \text{"CTOD}(B_i)\text{"}$$

这样,当输入模式 B 确定后, F 即可唯一。

由此可见, F 的组合形式取决于集合 C, L, R, T 和 B, 改变其中任何一个,都将影响到 F 的组合形式和查询结果。因此,选择组合查询方法可分两部份:一是输入模式 B 的生成;二是条件表达式 F 的生成。设计通用的选择组合查询方法,应能够处理不同的查询模式而不必对方法本身进行任何修改。根据这一思想,设查询模式的数据结构 QK 为:

Structure for database, A:\QK DBF

Number of data records: 14

Data of last update 12/26/90

| Field | Field Name | Type | Width | Dee |
|-------|------------|-----------|-------|-----|
| 1 | KM | Character | 8 | |
| 2 | B | Character | 10 | |
| 3 | W | Numeric | 2 | |
| 4 | C | Character | 10 | |
| 5 | T | Character | 1 | |
| 6 | L | Character | 5 | |
| 7 | R | Character | 2 | |

** Total **

39

结构中, B 对应 B_i 用以说明 B_i 的意义, 以提示用户输入; W 为 B_i 的初始化位数, 以便在 B_i 输入时确定字段值位数; C 对应 C_i , 表示 B_i 所对应的字段名, T 对应 T_i , 以确定对 B_i 的处理方法; L 对应 L_i ; R 对应 R_i . 为实现以上方法, 我们设计了一个输入模式生成程序 QMG. PRG 和一个条件表达式 F 的生成程序 QFPG. PRG. 用户可根据各自的需求, 由 QMG 模块输入 B 的数据值, 再由 QFPG 模块生成相应的查询条件表达式, 从而得出所需的查询数据. QMG. PRG 和 QFPG. PRG 模块程序框图见图 1·2 图所示。

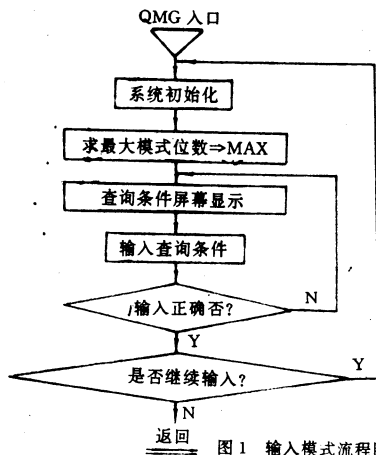


图 1 输入模式流程图

三、性能分析及完善 上述选择组合查询方法, 允许处理 1~99 个输入模式选择项。其能够组合生成的条件表达式种类数为:

$\sum_{i=1}^m C_i^n$ (m 为查询模式库中的记录数) 为了使用方便, 直观, 我们还增加了查询结果显示, 见图三所示, 或打印输出模式和集合 L、R、C、T 的定义模块, 使得查询结果输出更直观, 关系定义更为方便。

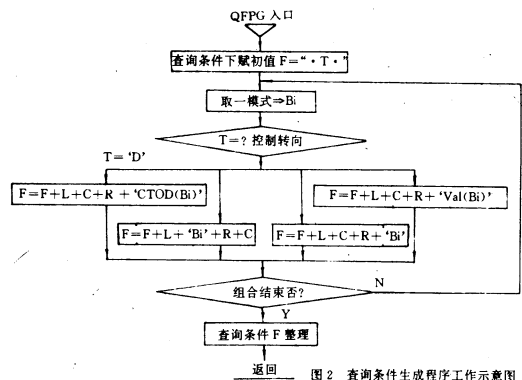


图 2 查询条件生成程序工作示意图

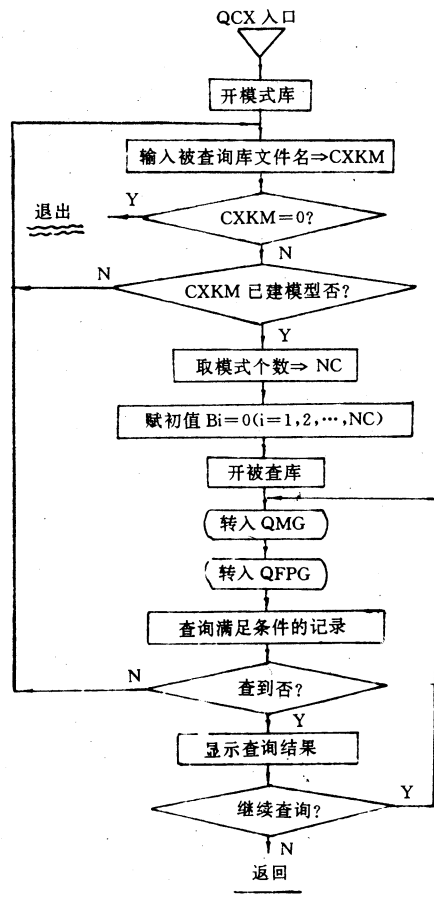


图 3 显示查询建模结果流程图

参考文献

- [1] 凌连生等著,《新颖关系数据库管理系统》,中国科学院计算技术研究所,1983.3.
- [2] 史济民著,《数据库管理及应用》,高等教育出版社,1987.10.
- [3] 郑若志等著,《数据库原理与方法》,湖南科学技术出版社,1983.7.

多窗口用户界面的设计与实现

中国科学院自动化所 史品志 刘文明 牛连江

摘要 本文重点讨论了采用弹出式技术结构用户界面的基本方法。恰当地运用该技术的关键,是设计弹出式工具箱、鼠标工具箱和容错信息包。该方法对设计其它类型的用户界面也不无裨益。

一、概述 在设计软件系统中,用户界面的设计是改进系统质量的关键技术。一个友善的用户界面,对提高系统的使用效率和实用性能尤为重要。

目前国际上流行的用户界面技术主要可分为以下五类:

(1)下拉式技术;(2)弹出式(或堆栈式)技术;(3)分层式技术;(4)Two—Views;(5)生成式技术。

在许多工作站和个人计算机软件环境上均提供了用户界面设计工具。典型的系统有用户接口管理系统(User Interface Management System),对话管理员(Dialog Manager)或接口建造者(Interface Builder),这些工具使用户更容易建立友好的用户界面[7, 12]。

一个发展趋势是对构造用户接口采用直接的处理编辑器[1, 2, 6]。直接处理方法的优点是:设计者可直接看清接口是什么样子,并且很容易改变接口特性。缺点是对大量的接口对象难以做出一致性的更改。

近期的研究中,有人建议用描述语言和交互编辑器组成一个“Two—Views”的编辑器,它可以同时对两种描述进行存取[5]。

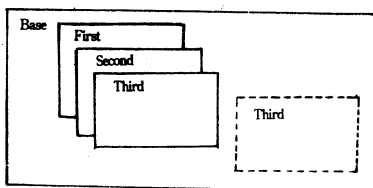
作为商品化软件系统,最常见的是采用弹出式技术来构造用户界面。目前 Turbo C2. 0、Microsoft C6. 0的用户工作平台均采用该种技术。基于我们研究开发的体会,本文重点讨论利用弹出式技术开发多窗口的的基本方法,相信它对于开发其他种类的用户界面技术也会起到触类旁通的作用。我们采用多窗口用户界面技术设计的彩色图象处理系统,其结构如图1示。

构造多窗口用户界面的关键是建立弹出式窗口工具箱、鼠标工具箱和容错信息包。

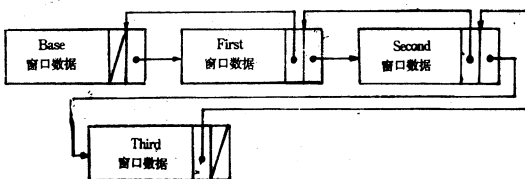
下面各节中将分别讨论。

用户窗口按视频 RAM 的存储方式可分为两类,一类是文本方式的用户界面,另一类是图形方式的用户界面。前一类用户界面通用性强,不受图形卡的限制,但不如图形方式的用户界面功能强。图形方式的用户界面更为灵活,并可方便地形成汉字界面,但受图形卡的约束。限于篇幅,此处仅讨论文本方式的用户界面,它对开发图形方式的用户界面技术也不无裨益。

二、弹出式窗口堆栈 弹出式技术的重要概念是基于堆栈来管理窗口。如图2所示的多窗口和对应的堆栈结构。堆栈结构是一个双向链表。这两个指针连接“下面的”窗口和“上面的”窗口。“上面的”和“下面的”是指:每个窗口都保存它背后的图形(除某种类型的窗口外),因此,必须对窗口弹出顺序保持记录,以使,在去除弹窗(栈弹),其背后的窗口能正确恢复。



a). 弹出式窗口 虚线指出顶层Third被移动后的位置



b). 窗口堆栈

图2. 弹出式窗口与对应的堆栈

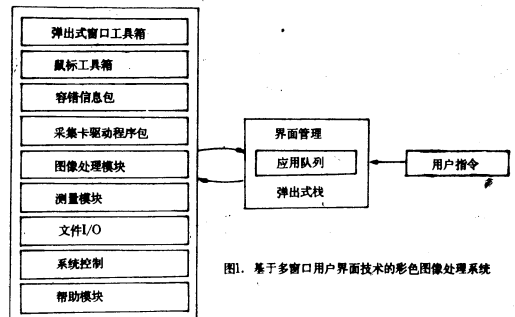
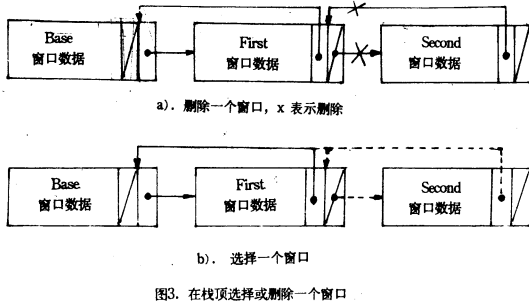


图1. 基于多窗口用户界面技术的彩色图像处理系统



三、窗口交换

管理弹出式窗口有2个基本管理例程,即:窗口的选择与删除,窗口的隐藏与显示。

(1) 选择与删除窗口的工作均在堆栈顶部来进行。删除一个窗口时,相当于删除栈顶的节点如图3a所示。它把 Second 窗口作栈顶,并修改尾指针。选择一个窗口时把该窗口节点加到栈顶,如图3中b所示。

(2) 隐藏和显现窗口

当选择或删除一个窗口时,它是面向弹出式窗口堆栈而言的,与此同时,对终端要同步进行显现窗口和隐藏窗口。值得强调,所有的交换窗口均在堆栈顶部进行。这是为了保证,无论窗口多少,均是按有序管理,不会出现杂乱无章的现象。据称,Sun 工作站的弹出式窗口可达64层深度。那么几万个功能如果不是在一套有序管理下进行界面理可能会一团糟。特别是在高集成化软件中更为至关重要。

(3) 窗口移动

如图2a)中的 Third 窗口被移动。移动后,Second 窗口的内容可在终端上显现出来。它主要是通过交换堆栈中两个窗口的内容及重新刷新屏幕来实现。

四、鼠标工具箱

构造鼠标工具箱应保证:

(1)、功能应是完备的,即实现鼠标驱动器可完成的全部功能。

(2)、能保证在终端或图象(图形)监视器上可靠切换。

(3)、面向高级语言的逻辑接口。

鼠标工具与鼠标驱动程序之间的关系如图4所示。

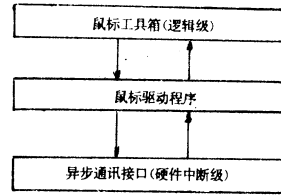


图4鼠标工具箱与驱动程序、异步通讯口的关系

通常在设计多窗口用户界面时,需要考虑键盘和鼠标两者都能使用,键盘的优先级高于鼠标,即两者同时被检测时,先响应键盘的命令。

五、容错信息包 该信息包的任务是处理错误信息。它按下面的方式工作:涉及两个窗口:一个是处理错误信息;另一个是给用户提示信息。窗口在信息出现时弹出,一直持续到用户响应时消隐。信息窗的管理与窗口交换管理的方式类似,此处不再累赘述。

六、结束语 采用弹出式技术设计多窗口用户界面是行之有效的方法。它比分层式技术管理简单。但在设计窗口移动时,不如分层式更为灵活。原因是:弹出式由于采用堆栈式管理,每次只能激活栈顶窗口移动,然而,分层式则可激活任一层窗口移动。采用生成式技术设计的多窗口功能更强,可随用户需求生成窗口。其不足之处是系统开销大,响应慢。

目前,在多窗口用户界面设计中,下拉式菜单技术已渐过时,采用弹出式和分层式技术仍居主流。随着个人计算机的容量不断增大,速度不断提高,窗口技术不断改进,Two-Views 和生成式技术有可能在未来几年中发展更为迅速。

参考文献

- [1] Brad A. Myers User-Interface Tools, Introduction and Survey. IEEE Software, 6(1), 15-23, January 1989.
- [2] G. Avrahami, K. P. Brooks and M. H. Brown, A Two-View Approach to Constructing User Interfaces, Computer Graphics, 23(3), 137-146, July 1989.
- [3] Greg Nelson, Juno, a constraint-based graphics system. Computer Graphics, 19(3) 235-243, July 1985.
- [4] H Rex Hartson and Deborah Hix, Human-Computer Interface Development: Concepts and Systems. ACM Computing Surveys, 21(1), 5-92 March 1989.
- [5] Luca Cardelli. Building User Interfaces by Direct Manipulation. In Proc. ACM SIGGRAPH Symp. on User Interface Software, pages 152-166, October 17-19 1988.
- [6] Mark A. Linton, John M. Vlissides, and Paul R. (下转 38 页)

通用函数曲线作图程序

空军领航学院 谢进一

一、问题的由来 使用计算机绘制各种函数曲线,是许多用户经常遇到的任务。一般的作法是根据给定曲线的函数关系式,编制一个相应的作图程序。这样做的局限性显而易见,函数关系改变,程序就得做相应的修改。这样的程序基本不能交流。为克服这一弊病,不少图形软件采用了“曲线库”的方法,即编出若干种常用工程(函数)曲线作图子程序,置于“曲线库”中,通过交互方式,供用户作图时选择。但是由于这种方法仍是基于“一线一程”模式,遇到库中没有的函数曲线作图要求,仍是无能为力。随之而来的是,这类软件的推广应用亦受到影响。如果能编制一个通用曲线作图程序,使用时,用户只需输入欲绘制曲线的函数表达式,及必要的坐标参数,计算机就可自动绘出相应曲线,无论怎样修改函数表达式,都不用改动程序,这将极大地方便用户,也利于推广及交流。本程序即以上述想法为设计目标,实现了“万线一程”。

二、函数表达式的翻译算法 要想根据屏幕输入的函数表达式绘 制出相应的曲线,关键问题是找到将该表达式(实际是一字符串)转换成计算机的操作序列的方法。这种转换,就称之为翻译算法,它本是各种编译程序的主要功能,因此,我们可借鉴编译原理,构造自己的函数表达式翻译算法。本程序采用的是一遍扫描翻译算法,即:对表达式从左至右一遍扫描,边扫描,边分析扫描对象,并根据对象的不同性质进行不同的操作。扫描完毕,即可求出相应的函数值。例如:对表达式 $2 * \sin(3+X)$ 的翻译过程为:

- (1)分离出操作数 2,压入数据栈;
- (2)分离出运算符 $*$,压入符号栈;
- (3)分离出标准函数名 \sin ,查取相应的函数符,压入符号栈;
- (4)分离出左括号 $($,压入符号栈;
- (5)分离出操作数 3,压入数据栈;
- (6)分离出运算符 $+$,压入符号栈;
- (7)分离出自变量 X ,将循环变量 X 当前值压入数据栈;
- (8)分离出右括号 $)$,转运算;
- (9)取操作数 X , 3,运算符 $+$,做 $3+X$ 运算,结果压入数据栈;

(10)取 $(3+X)$ 的结果, \sin 函数码,做 $\sin(3+X)$ 运算,结果压入数据栈;

(11)取 $\sin(3+X)$ 结果,操作数2,运算符 $*$,做 $2 * \sin(3+X)$ 运算,结果压入数据栈,结束。其算法描述如表。

说明: 1. $L[g]$:表达式字符串当前子串首字符;

2. $\#$:表达式字符串结束标志;

3. $W1, W2$:算法定义的双目运算符,为 $+, -, *, /, ^$;

4. 本算法用到两个栈,一为数据栈,用于存放常数, X 变量当前值及运算的中间结果,最终的结果(Y 值)亦放在内;一为符号栈,用于存放运算符,标准函数符及分层用的左括号;

5. GEN_w :双目运算子程序(过程),其算法为:

- ①从数据栈取出两个操作数;
- ②从符号栈取出一个运算符;
- ③做运算符指定的双目运算;
- ④结果压入数据栈,返回。

表1 表达式一遍扫描翻译算法

| 符号栈顶 $L[g]$ | 运算符 $W1$ | 函数符 | 左括号 |
|----------------|---|--------------|----------------|
| 常 数 | 常数串转数值,压入数据栈, NEXT | 同 左 | 同 左 |
| 变 量 | 循环变量 X 当前值压入数据栈, NEXT | 同 左 | 同 左 |
| 函 数 | 函数名串转函数符,压入符号栈, NEXT | 同 左 | 同 左 |
| (| 左括号压入符号栈, NEXT | 同 左 | 同 左 |
| 运算符 $W2$ | 若 $W1(W2)$, 则 $W2$ 入符号栈, NEXT 若 $W1) = W2$, 则 GEN_w | 求函数值,结果压入数据栈 | $W2$ 入符号栈 NEXT |
|) | GEN_w | 同 上 | 弹出左括号 NEXT |
| 结束符 # | GEN_w | 同 上 | 返 回 |

三、程序实现时的几个问题及对策 表达式翻译这一核心算法解决后,还有一些问题需要解决,

如:怎样区分表达式中各种不同成份?怎样实现标准函数的运算?怎样将求得的函数计算点(X,Y)转换成屏幕位置坐标(Xs,Ys)?针对这些问题,本程序采取了如下对策:

1. 表达式不同成份的识别.对初等函数表达式进行分析可知,一切初等函数不外乎都是由常数、函数、自变量、括号及运算符这几种成份组成,为了简化成份识别,人为规定自变量用X表示,表达式书写遵照计算机规定,这样,当对表达式扫描时,可做如下表处理:

表2 表达式成份识别

| L[g] | 识别 | 分离处理 | 结束条件 |
|---------------|--------|---------------|--------|
| 0..9 | 常数首字符 | 逐位取码,转换成数值 | 非 0..9 |
| A..Z 不含 X | 函数名首字母 | 逐字节取函数名 | 非 A..Z |
| X | 自变量 | 用循环变量 X 当前值代换 | 下一字符 |
| (| 分层符左括号 | 取相应的 ASCII 码 | 下一字符 |
| +, -, *, /, ^ | 运算符 | 取相应的 ASCII 码 | 下一字符 |
| # | 表达式结束符 | | 全式结束 |

2. 标准函数运算.首先要构造一张函数名→函数码对照表,如:

| 函数名 | 函数码 | 函数名 | 函数码 |
|--------|-----|------|-----|
| ABS | 1 | LN | 6 |
| ARCTAN | 2 | SIN | 7 |
| COS | 3 | SQR | 8 |
| EXP | 4 | SQRT | 9 |
| INT | 5 | | |

其次,设计一个求值过程,它的功能是:根据给定的函数符作相应函数运算.由于函数是单目运算(只要求从数据栈中取出一个操作数),为了与双目运算符区别,人为规定,双目运算符从00H开始编码,函数符则从80H起编码.

函数符=80H+函数码

例如,SIN的函数码为7,其函数符为87H.有了以上准备后,标准函数运算变得极为简单,只不过是查表、求值这样两个简单的过程,至于其相应的出入栈及运算时机,在翻译算法一节已有阐述,这里不再赘述.

3. 函数计算点与屏幕显示点的坐标换算.对于任意初等函数表达式,给出自变量X一个确定的值,应用上述算法,即可求得函数Y一个确定的值,而这个点(X,Y)尚不能直接显示到屏幕上,需要进

行必要的坐标变换.下面以CGA640×200图形方式为例,说明其转换方法:

①用户坐标系的确定

H:用户横坐标(X轴)总刻度(相当于屏幕横向所分份数);

V:用户纵坐标(Y轴)总刻度(相当于屏幕纵向所分份数);

U_x, U_y:用户坐标系原点位置;

X₀, X_a, dx:自变量X起止值及步长

②计算点(X,Y)→屏幕点(X_s,Y_s)转换算法:

$X_s = \text{INT}(O_x + (U_x + X) * B_x - 1)$;

$Y_s = \text{INT}(O_y + (U_y + Y) * B_y - 1)$;

其中,O_x,O_y为屏幕标系左下角值,本例为0,199; B_x, B_y为坐标换算系数本例中 B_x = 640/H, B_y = -200/V;对每个计算点,经上述转换后,即可显示于屏幕.

四、演示程序及说明 演示程序用Turbo Pascal V4.0编成.在PC/XT上通过运行,适用于任意初等函数.由于篇幅所限,只给出了主程序及翻译算法主过程的程序清单.不难看出,在本程序中加上打印模块及必要的菜单选择控制,即可得到一个实用的通用曲线作图程序.

源程序清单:

```

Program PolyDraw;
Uses Graph;
Var X,Y,Ox,Oy,Ux,Uy,Bx,By,H,V,X0,Xa,dx,:
    FunStr:String;
Xs,Ys,I,Graph Driver,GraphMode,Integer;
Procedure YValue;
Var FunCode,Dc,g,a,p,Opt:Byte;
    Astack:Array[0..50]of Real;
    Wstack:Array[0..5]of Byte;
    L:String;
    Opr1,Opr2:Real;
begin
    L:=FunStr;g:=1;a:=0;p:=1;Dc:=255;
    Wstack[0]:=Dc;
    while g<=Length(L)do
    begin
        Case L[g] of
            '0'..'9':begin StrVal;Inc(g);end;
            'A'..'W','Y','Z','a'..'w','y','z':begin F-
            Code;Inc(g);end;
            'X','x':begin Astack[a]:=X;Inc(a)Inc(g);
            end;

```



```

'(', begin Wstack[p] := Dc; Inc(p); Inc(g); end;
'+', '-', '*', '/', '^'; begin
Case Wstack[p-1] of
0..4; begin
if (Int(Wstack[p-1]/2) * 2) >= (Int(OptCode
(L[G])/2) * 2
then GENw
else begin
Wstack[p] := OptCode(L[g]); Inc(p); Inc(g);
end;
end; {0..4}
129..138; ALUFun;
255; begin Wstack[p] := OptCode(L(g); Inc(p);
Inc(g); end;
end; {Case}
end; {w2}
')'; begin
Case Wstack[p-1] of
0..4; GENw; 129..138; ALUFun;
255; begin Dec(p); Inc(g); end;
end; {Case}
end; {}
'##'; begin
Case Wstack[p-1] of
0..4; GENw; 129..138; ALUFun;
255; begin Y := Astack[0]; EXIT; end;
end; {Case}
end; {#}
end; {Case L[g]}
end; {while}

```

```

end; {Procedure Yvalue}

begin
Graph Driver := 0; GraphMode := 0;
Initgraph(Graph Driver, Graph Mode, 'C:\');
SetViewPort(0, 0, GetMaxX, GetMaxY, True);
Write('Input FunStr, please, '); Readln(FunStr);
if FunStr[Length(FunStr)] <> '#' then
Write('Input H, V, Ux, Uy, Xo, Xn, dX, ');
Readln(H, V, Ux, Uy, Xo, Xn, dX);
ClearViewPort;
Ox := 0; Oy := 199; Bx := GetMaxX/H; By := -Get-
MaxY/V;
X := Xo; Yvalue;
Xs := Trunc(Ox(Ux+X) * Bx);
Ys := Trunc(Oy(Uy+Y) * By);
MoveTo(Xs, Ys);
repeat
X := X+dx; Yvalue;
Sx := Trunc(Ox(Ux+x) * Bx-1);
Ys := Trunc(Oy(Uy+Y) * By-1);
LineTo(Xs, Ys);
until X >= Xn;
end

```

(vepwat)

参考文献

- [1] 《计算机程序设计技巧》 (美) D. E. 克努特
著 国防工业出版社
- [2] 《程序设计语言编译原理》 陈火旺 钱家骅
孙永强 编 国防工业出版社
- [3] 《编译方法》 金成桂 编 高等教育出版社

中文菜单的设计方法

湘潭大学机电系 曾向阳

摘要 本文通过对各种形式菜单特点的分析,提出了一种用 Turbo-pascal 语言在 IBM-PC/XT/AT 上实现的下拉式中文菜单的设计方法和设计技巧。

一、菜单概述 对于一个多功能软件,为方便用户使用,提高软件和用户之间的友善性,往往把软件的各项功能以一系列选择项的形式列出,用户可根据需要从中选取所需的某项实现相应的操作。这一系列的选择项就是菜单。良好的菜单形式可使用户方便和有效地使用软件,使软件的使用简单易学,操作方便。

菜单有多种形式,其实现的方法一般有以下几种:(1)把菜单的各项按顺序列出,并用序号标出,选择菜单的某一项时只要键入此项对应的序号字母即可;(2)把菜单的各项按顺序列出,每一项用一不同的特征字母(通常用大写形式或用不同的颜色)标

记,选择菜单的某一项时只要键入此项对应的特征字母即可;(3)把菜单的各项按顺序列出,每一项用一不同的特征字母(通常用大写形式或用不同的颜色)标记或不标记,选择菜单的某一项时键入此项对应的特征字母或按光标键移动光柱即可。

前两种菜单一般只能是一次显示,选择某一项后菜单即消失,这种菜单也称之为闪现式菜单或老式菜单。这种菜单只要用 If 语句或 Case 语句,根据各项的序号或特征字母转到相应的功能处理子程序即可实现,实现容易,方法比较简单。缺点是显示菜单时会把屏幕上原有的内容清除掉,菜单选中后又

不能重选,另外这种菜单一般只有一个层次,不能实现多级子菜单,不能实现动态菜单功能,同时当菜单项实现的操作比较复杂时不易实现,与一些先进的输入工具(如鼠标)不易连接。

后一种菜单一般利用键盘上的光标移动键或空格键来选择菜单的各项。菜单的实现一般要利用计算机的系统资源或在图形方式下才能实现,可实现多级菜单显示而形成所谓下拉式菜单。可方便地实现动态式菜单,使用方便易学,可充分利用显示屏幕的颜色使菜单美化,可以不选择就退出当前一级子菜单,选择以后也可重选。

以上所述是针对西文菜单而言,而中文菜单则又具有其相应的特点。

中文菜单由于键盘上没有对应的汉字键,用上述第二种方法就不易实现。如果用特征汉字来代替特征字母,则由于受汉字输入速度或方法的限制,很不方便。另外显示一屏中文菜单的项数远不及西文项数多,软件功能稍多则用老式菜单就不易实现。因此中文菜单最好采用下拉式菜单的形式。

中文显示是图形方式下的文本汉字显示,汉字的显示缓冲区与图形的显示缓冲区是共用的,在 CCDOS 下不易实现汉字与图形的同时显示。

对一些要求内存量较大的软件,在 CCDOS 下使用往往内存不够,希望在 MSDOS 或 PC DOS 下使用。但这时如使用中文菜单,则必须使用一些特殊的技术,为解决显示的速度问题还需要一些特殊技术。

二、下拉式中文菜单的实现方法:要实现一个下拉式中文菜单必须做以下工作:

1. 菜单的显示。这里要解决以下问题:菜单的项数,每一菜单项的名称和功能,菜单的显示位置,菜单的选择键等,还有汉字的显示问题和装饰问题。

2. 屏幕的保存与恢复。菜单显示时会消除屏幕的原有内容,下级菜单显示时也会清除掉上级菜单的部分显示内容。当从下级菜单退回到上一级菜单或撤消菜单显示时,必须把清除掉的屏幕内容恢复原样,这样在菜单显示前必须先把欲显示的屏幕区域信息保存下来。

3. 菜单项的选择

必须用醒目的颜色显示一条光柱,光柱所在的位置即为相应欲选择的某一项对应的位置。按光标移动键应使光柱相应移动。还必须获取键盘的输入信号。

要实现一个下拉式中文菜单,一个合适的数据结构是必须的。好的数据结构可使程序实现简单,结构清晰,提高速度,节省内存等。程序中我们采用了以下的数据结构:

每一菜单项采用表一所示的数据结构:(表一)

MenuItem=Record

ItemName:string[40];{每一菜单项的名称}

NextProc:pointer;{指向此菜单项的指针,

可为实现此项功能子过程的地址}

End;

整个菜单采用表二所示的数据结构:

(表二)

MenuType=Record

Title: String[40];{菜单标题}

Item: Array[1..10]of MenuItem;

{菜单项数,对 CGA 方式最大为10行}

X,Y: Word; {菜单左上角的
的屏幕坐标}

CountLine, {实际菜单项数}

CountWord, {最长一项菜单
的汉字字符数}

Selected: Byte; {当前选择项的序
号}

WindScr: Pointer;{保存屏幕原有内
容}

WindSize: Word; {需保存屏幕内
容的大小}

Key: KeyType;{菜单选择时所
使用功能键的集合}

End;

或表三所示的数据结构:

(表三)

MenuType=Record

Title: String[40];{菜单标题}

Item: Array[1..10]of MenuItem;

{菜单项数,对 CGA 方式最大为10行}

X,Y: Word; {菜单左上角
的屏幕坐标}

CountLine, {实际菜单项
数}

CountWord, {最长一项菜
单的汉字字符数}

Selected: Byte; {当前选择项的序
号}

FileName:String; {屏幕保存磁盘文
件的文件标识符}

Key: KeyType;{菜单选择时所
使用功能键的集合}

End;

三、中文菜单的显示问题:CCDOS 下,在屏幕上

显示中文信息只能用 Write 或 Writeln 语句实现,但显示时存在硬件光标闪烁问题,同时不能充分利用图形功能使显示美化,显示的位置也不易确定,移动式光柱不易实现。若使用图形方式,则应将 CRT 单元中的标准变量 DirectVideo 设定为 false,这样才能使用 CRT 单元的功能使文本汉字与图形同时显示,并且可使用文本窗口使显示更容易。这时菜单容易装饰,显示位置可随意,移动式光柱可用图形功能来实现。对 EGA 和 VGA 等高分辨率显示器还可利用其丰富的彩色实现彩色汉字显示。

MSDOS 或 PC DOS 西文方式下的汉字菜单显示,一般是利用 CCDOS 的汉字库,将欲显示的汉字字模点阵取出存放于一数组中,经过适当变换再以位图的形式显示在屏幕上。这样,菜单可显示在屏幕的任意位置,不受屏幕文本方式行列的限制,并可任意调整行距和字距。如果采用 CCLIB 的 16 点阵汉字库,则字模点阵可用下面的数据结构来表示:

```
Words=Array[1..16]of Word;
```

字模点阵变换成的位图数据可用如下的数据结构来表示:

```
HZImageType=Array[1..18]of Word;
```

这里的前两个字分别表示位图点阵的宽和高的像素数,对 16 点阵字库各为 15。字模点阵变换为位图数据后就可利用 GRAPH 单元的位图过程将汉字显示在屏幕上。

在西文方式下显示汉字菜单时,为提高显示速度,可采取以下一些措施:建立专用汉字库,减少字模检索的时间;先将整个菜单的每一个汉字的字模取出,经过处理后再用位图过程一次显示到屏幕上;汉字库存入硬盘或虚拟盘。

四、屏幕的保存与恢复问题 由于汉字显示是在图形屏幕方式下实现的,因此可利用位图过程将欲保存的屏幕信息保存在数组中或指针变量中,恢复时再用位图过程来实现。这种方法实现的屏幕信息的保存与恢复速度很快,实现方便,但要占用一定的内存空间,特别是当菜单层次较多时则占用的内存量较大。

也可将由位图过程获取的屏幕信息保存在磁盘文件中,恢复时再调用磁盘文件中的信息。为提高磁盘文件的存取速度,可利用块读和块写过程,并且可把屏幕数据保存在硬磁盘上或虚拟盘上。

五、程序的结构 作者编写了两个单元,分别实现了 CCDOS 和 PC DOS 下的中文菜单显示。用户只要在主程序中对菜单数据记录中的每一项赋以相应的值即可实现不同的中文菜单。CCDOS 下中文菜单单元的接口部分如表四所示:

(表四)

```
Unit CCMenu;

Interface

uses Dos,Crt,GCrUnit,Graph,KbdUnit;

type MenuItem=Record

    ItemName:string[40];
    NextProc:pointer;
End;

MenuType=Record

    Title:string[40];
    Item:Array[1..9]of MenuItem;
    Col,Row:byte;
    CountLine,CountWord;
    Selected:byte;
    WindScr:pointer;
    WindSize:integer;
    Key:KeyType;
End;

var ExitMenu:Boolean;

Procedure CallProc(Sub:pointer);
Procedure ExitProc;
Procedure NullProc;
Procedure GeneralMenuProc(Menu,MenuType);
Implementation
MSDOS 或 PC DOS 下中文菜单单元的接口部分如表五所示
(表五)

Unit HZMenu;

Interface

uses Crt,Graph,KbdUnit;

type Words=array[1..16]of word;

HZImageType=record

    HZ,array[1..18]of word;
end;

WordImage=^ WordImageArray;
WordImageArray=array[1..10,1..10]of HZImageType;

MenuItem=Record

    ItemName:string[40];
    NextProc:pointer;
End;

MenuType=Record

    Title:string[40];
    Item:Array[1..10]of MenuItem;
    x,y:word;
```

```

        CountLine, CountWord,
        Selected; byte;
        WindScr; pointer;
        WindSize; word;
        Key; KeyType;
    End;
var CCLibrary; file of Words;
    ExitMenu; Boolean;
Procedure CallProc(Sub; pointer);
Procedure ExitProc;
Procedure NullProc;
Procedure GeneralMenuProc(Menu; MenuType);
Implementation
CCDOS 下中文菜单单元演示主程序如表六所示
(表六)
Program CCMenuDemo;
uses CCMenu, SetGrMod;
var MainMenu, SubMenu; MenuType;
{ $F+ } Procedure MainMenuProc; { $F- }
begin GeneralMenuProc(MainMenu);
end;
{ $F+ } Procedure SubMenuProc; { $F- }
begin GeneralMenuProc(SubMenu);
end;
Procedure DefineMainMenu;
begin
    FillChar(MainMenu, SizeOf(MainMenu), 0);
    with MainMenu do
    begin Title := '主菜单';
        Item[1]. ItemName := '主菜单主菜单';
        Item[1]. NextProc := @SubMenuProc;
        Item[2]. ItemName := '主菜单主菜单';
        Item[2]. NextProc := @SubMenuProc;
        Item[3]. ItemName := '主菜单主菜单';
        Item[3]. NextProc := @ExitProc;
        CountLine := 3;
        CountWord := 6;
        x := 1; y := 1;
        Selected := 1;
    end;
end;
Procedure DefineSubMenu;
begin
    FillChar(SubMenu, SizeOf(SubMenu), 0);
    with SubMenu do
    begin Title := '子菜单';

```

```

        Item[1]. ItemName := '子菜单子菜单';
        Item[1]. NextProc := @SubMenuProc;
        Item[2]. ItemName := '子菜单子菜单';
        Item[2]. NextProc := @NullProc;
        Item[3]. ItemName := '子菜单子菜单';
        Item[3]. NextProc := @ExitProc;
        CountLine := 3;
        CountWord := 6;
        x := 60; y := 5;
        Selected := 1;
    end;
end;
begin
    ExitMenu := false;
    SetCGAMode(High);
    DefineMainMenu;
    DefineSubMenu;
    GeneralMenuProc(MainMenu);
    ClearDevice;
end.
MSDOS 或 PC DOS 下中文菜单单元演示主程序如表
七所示
(表七)
Program PC DOS — CCMenuDemo;
uses Graph, HZMenu, SetGrMod;
var MainMenu, SubMenu; MenuType;
{ $F+ } Procedure MainMenuProc; { $F- }
begin GeneralMenuProc(MainMenu);
end;
{ $F+ } Procedure SubMenuProc; { $F- }
begin GeneralMenuProc(SubMenu);
end;
Procedure DefineMainMenu;
begin
    FillChar(MainMenu, SizeOf(MainMenu), 0);
    with MainMenu do
    begin Title := '主菜单';
        Item[1]. ItemName := '主菜单主菜单';
        Item[1]. NextProc := @SubMenuProc;
        Item[2]. ItemName := '主菜单主菜单';
        Item[2]. NextProc := @SubMenuProc;
        Item[3]. ItemName := '主菜单主菜单';
        Item[3]. NextProc := @NullProc;
        CountLine := 3;
        CountWord := 6;
        x := 1; y := 1;

```

```

        Selected:=1;
    end;
end;
Procedure DefineSubMenu;
begin
    FillChar(SubMenu,SizeOf(SubMenu),0);
    with SubMenu do
    begin Title:='子菜单';
        Item[1].ItemName:='子菜单子菜单';
        Item[1].NextProc:@SubMenuProc;
        Item[2].ItemName:='子菜单子菜单';
        Item[2].NextProc:@NullProc;
        Item[3].ItemName:='子菜单子菜单';
        Item[3].NextProc:@NullProc;
        CountLine:=3;
        CountWord:=8;
    end;
end;

```

```

        x:=60;y:=20;
        Selected:=1;
    end;
end
begin
    ExitMenu:=false;
    SetCGAMode(High);
    DefineMainMenu;
    DefineSubMenu;
    Assign(CCLibrary,'C:\CCLIB');
    Reset(CCLibrary);
    GeneralMenuProc(MainMenu);
    Close(CCLibrary);
    ClearDevice;
end.

```

IBM—FORTRAN 语言屏幕彩色字符显示功能的扩充

四川石油测井研究所 张 义

一、引言 众所周知,IBM—PC 及其系列兼容机的 BASIC 语言为屏幕文本字符(包括汉字)的输出提供了彩色显示功能,为 BASIC 应用软件的编制提供了色彩鲜艳的屏幕模式。FORTRAN 语言是一种功能较强、应用极为广泛的通用程序设计语言,它对复杂的科研、教学、工程和财政计算来说尤其适用。但是,FORTRAN 语言却令人遗憾地缺乏彩色文本输出功能,这使得 FORTRAN 应用软件的屏幕显示只能用单调的黑白输出方式。鉴于此,作者根据实际工作的需要,编制了一个用于 FORTRAN 语言屏幕文本显示模式的宏汇编子程序,该子程序可使用户在 FORTRAN 语言环境中,以某一确定的屏幕位置用十六种色彩进行文本输出,为 FORTRAN 应用软件开发提供了灵活多变、丰富多采的屏幕显示功能。

二、实现方法 开发工作是在 GW—286 主机、CEGA 显示卡上进行的,软件环境为:MS—DOS3.0 (或 CCDOS2.0)及其以上版本操作系统、宏汇编3.0 及 FORTRAN3.31 编译系统等。试验表明,本文所介绍的程序和方法在 IBM—286、386 等机种及 CGA、EGA、VGA、COLOR400 等显卡的彩色显示器上都能使用,具有一定的实用推广价值。

彩色字符输出子程序 PRT.ASM 见程序一。

该子程序可在彩色屏幕的指定位置上以指定的

色彩输出一串字符(或汉字),这一功能是通过通过对 INT 10H 显示管理的功能调用来实现的。PRT 子程序已设计好与 FORTRAN 语言的接口,其调用格式为:

```
CALL PRT(IC,IX,IY,'STRING',NC)
```

其中,参数 IC 为输出字符串所需显示颜色的色号,其值和颜色的对应关系为:

| | | |
|-------|--------|---------|
| 0——黑 | | |
| 1——蓝 | 6——棕 | 11——浅青 |
| 2——绿 | 7——白 | 12——浅红 |
| 3——青 | 8——灰 | 13——浅洋红 |
| 4——红 | 9——浅蓝 | 14——黄 |
| 5——洋红 | 10——浅绿 | 15——高亮白 |

此外,如果想使显示的字符具有背景色,则可置 IC 为 16 加上对应的颜色号,IX、IY 分别为屏幕行、列数, $0 \leq IX \leq 25$, $0 \leq IY \leq 80$, STRING 为输出字符串,其内容必须用单引号引导,NC 为所显示的字符串之字符个数(注意,STRING 为汉字时,一个汉字占两个字符宽度)。

程序二是一个光标定位子程序,其功能是将光标定位于屏幕上指定位置。如果用户把程序一(PRT.ASM)和程序二(LOCATE.ASM)配合使用,可以使光标定位于所显示的字符串之后的某一位置,从而使某些应用软件(如菜单程序)的文字提示和输

入项的光标位置更符合人们的习惯。

以上两个程序都是汇编程序,均已设计好和 FORTRAN 的接口,它们可直接由 FORTRAN 语言调用。程序三是一个调用上述二子程序的 FORTRAN 例程,该程序由 FORTRAN3.31 编译系统编译通过。

关于子程序的存放和调用方式,读者可将子程序用 LIB 库管理程序加入 FORTRAN.LIB 库中,也可单独将这两个汇编程序和读者的其他应用子程序形成一个用户库 USER.LIB,这样,用户在使用过程中,不必将各子程序模块一一列出和主程序模块相连接,而只需和库文件连接即可。

三、源程序

```
;-----
;功能:屏幕字符输出
;调用:CALL PRT(色彩,行,列,'输出字符串',
字符个数)
```

```
FRAME    STRUC
SAVEBP   DW?
SAVERET  DD?
FRAME    ENDS
DATA     SEGMENT PUBLIC 'DATA'
DATA     ENDS
CODE     SEGMENT 'CODE'
DGROUP   GROUP DATA
ASSUME    CS:CODE, DS:DGROUP, ES:
DGROUP, SS:DGROUP
```

```
PUBLIC PRT
PUSH BP
MOV BP,SP
LES BX,[BP+0Eh]
MOV DL,[BX]
LES BX,[BP+12h]
MOV DH,[BX]
LES BX,[BP+06h]
MOV CX,[BX]
LES BX,[BP+16h]
MOV AH,[BX]
LES BX,[BP+0Ah]
PUSH BP
MOV BP,BX
MOV AL,00H
MOV BL,AH
MOV BH,00H
MOV AH,13H
INT 10H
POP BP
```

```
MOV SP,BP
```

```
POP BP
```

```
RET 20
```

```
PRT    ENDP
```

```
CODE   ENDS
```

```
END
```

程序一,屏幕彩色字符输出子程序 PRT.ASM

```
;-----
```

```
;功能:光标定位
```

```
;调用:CALL LOCATE(行,列)
```

```
FRAME STRUC
```

```
SAVEBP DW?
```

```
SAVERET DD?
```

```
FRAME ENDS
```

```
DATA SEGMENT PUBLIC 'DATA'
```

```
DATA ENDS
```

```
CODE SEGMENT 'CODE'
```

```
DGROUP GROUP DATA
```

```
LOCATE PROC FAR
```

```
PUBLIC LOCATE
```

```
PUSH BP
```

```
MOV BP,SP
```

```
MOV DH,00
```

```
MOV DL,00
```

```
MOV BH,01
```

```
MOV AH,02
```

```
INT 10H
```

```
LES BX,[BP+0Ah]
```

```
MOV DH,[BX]
```

```
LES BX,[BP+06H]
```

```
MOV DL,[BX]
```

```
MOV BH,00
```

```
MOV AH,02
```

```
INT 10H
```

```
MOV SP,BP
```

```
POP BP
```

```
RET 8
```

```
LOCATE ENDP
```

```
CODE ENDS
```

```
END
```

程序一、光标定位子程序 PRT.ASM

```
C-----
```

```
PROGRAM MAIN
```

```
CALL PRT(12,5,14,'四川石油管理局
测井研究所试验程序 MAIN.FOR ',40)
```

```
CALL PRT(14,8,26,'本行显示颜色为
```

```

黄色;IC=14',24)
CALL PRT(11,10,26,'本行显示颜色为
浅青色;IC=11',26)
CALL PRT(20,12,26,'本行显示颜色为
红色,蓝底;IC=16+4',30)
CALL PRT(10,14,26,'本行显示颜色为
浅绿色;IC=10',26)

```

```

CALL PRT(13,18,26,'请输入一个整
数:',15)
CALL LOCATE(18,41)
PAUSE
END
程序三、FORTRAN 主程序 MAIN. FOR

```

一个 Turbo-C 的多窗口大程序框架

四川大学计算机系

唐常杰 韩仲清

本文给出了一个用 Turbo-C 实现的多窗口菜单驱动程序,该程序已在 IBM PC/XT 上调试通过,运行时能够显示颜色、窗口和菜单。用户选择1、2、3,就可以分别在三个不同的窗口中运行三个子程序,然后又回到主菜单,若选Q则退出。在使用中,用户只需要把子程序换成自己的函数,并按需要调整各个窗口的大小和颜色,即可以很快地开发出一个具有相当规模的多窗口菜单驱动式的程序。这种多窗口菜单驱动技术,常常作为开发大程序的一个框架。

下面将实现多窗口菜单驱动技术的 Turbo-C 源程序介绍给读者。

第11~20行是一个菜单函数。注意:这里用的是 Cprintf,如果改用 Printf,则无颜色和窗口;而且这里只用了一个 Cprintf 而不是每显示一行就用一个 Cprintf,利用\n\f换行回车,这样既缩短了代码又节约了运行时间。

第22~52行为画窗口边框的函数。

第53~75行为设置窗口函数,这儿共设置了四个窗口。统一地设置边框的大小和颜色,这样便于维护和修改。

第76~85行为选菜单函数,可设置默认值。

第87~100行分别为三个子模块,可以用自己的函数取而代之。

第102~126行为主函数,采用循环语句中套 Switch 的办法来分流。

程序中加了汉字注解,有较好的程序设计风格和可读性,便于读者模仿和改进。

/* -----

文件名:ProgFram.C 大程序框架

用法:模仿此程序框架,可组织大程序。

有颜色,窗口等附加效果。

----- */

```

#include<stdio.h> /* 6 */
#include<conio.h>
#include<string.h>
#include<alloc.h> /* 9 */
/* ----- */
void Menu() /* 菜单 11 */
{ char MenuStr[] =
    "          Menu          \n\r"
    " ----- \n\r"
    " 1.SubProgram 1          \n\r"
    " 2.SubProgram 2          \n\r"
    " 3.SubProgram 3          \n\r"
    " Q. Quit                 \n\r"
    " ----- \n\r"
    cprintf(MenuStr);} /* 20 */
/* ----- 窗口边框 ----- */
void Frame(int UpperLeftX,int UpperLeftY,
           int LowerRightX,int LowerRightY,
           int FramColor,int BackColor)
{int I; /* 25 */
const int UpLeft=201, /* 左上角 */
HorizonBar= 205, /* 水平线 */
UpRight =187, /* 右上角 */
VertBar =186, /* 竖线 */
LowerLeft= 200, /* 左下角 */
LowRight =188; /* 右下角 */
window(1,1,80,25);
textcolor(FramColor);
textbackground(BackColor);
gotoxy(UpperLeftX,UpperLeftY);
putch(UpLeft); /* 36 */
for(I=UpperLeftX+1;

```

```

    I <= (LowerRightX-1); I++;
    putch(HorizonBar);          /* 39 */
    putch(UpRight);
    for (I = (UpperLeftY+1);
        I <= (LowerTightY-1); I++)
    { gotoxy(UpperLeftX, I);
      putch(VertBar);           /* 44 */
      gotoxy(LowerRightX, I);
      putch(VertBar);           /* 45 */
      gotoxy(UpperLeftX, LowerRightY);
      putch(LowerLeft);
      for (I = (UpperLeftX+1);
          I <= (LowerRightX-1); I++)
      { putch(HorizonBar);       /* 51 */
        putch(LowRight);        /* 52 */
      }
    }
    /* ----- */
void SelectWindow(int WindowNo)
/* 置颜色和窗口 */           /* 55 */
{ int x1, y1, x2, y2,
  CharColor, BackColor;
  switch(WindowNo)           /* 58 */
  { case 0: x1=3; y1=3; x2=70; y2=23;
    CharColor=YELLOW; /* 60 */
    BackColor=GREEN; break;
    case 1: x1=7; y1=6; x2=50; y2=17;
    CharColor=LIGHTGREEN; /* 63 */
    BackColor=BLUE; break;
    case 2: x1=30; y1=4; x2=75; y2=10;
    CharColor=RED; /* 66 */
    BackColor=LIGHTGRAY; break;
    case 3: x1=2; y1=10; x2=60; y2=23;
    CharColor=MAGENTA; /* 69 */
    BackColor=BLACK; break; }
  Frame(x1, y1, x2, y2, WHITE, BLUE);
  textcolor(CharColor);      /* 72 */
  textbackground(BackColor);
  window(x1+1, y1+1, x2-1, y2-1);
  clrscr(); } /* 清窗口 75 */
/* ----- */
void ChooseMenu(char * question,
  char * choice, char * DefaultAnswer)
{ printf("%s[default is %s]",
  question, DefaultAnswer);
  cscanf("%c", choice); /* 81 */
  /* 如果改用 gets(), 则无颜色 */

```

```

    if (* choice == '\r') /* 83 */
        * choice = * DefaultAnswer;
    }; /* 菜单选答, 有默认值 */ /* 85 */
    /* ----- */
void SubProgram1() /* 示意性子程序 1 */
{ printf("\n\r this is SubProgram 1"
  "\n\r Press key...");
  getch(); } /* 90 */
/* ----- */
void SubProgram2() /* 示意性子程序 2 */
{ printf("\n\r this is SubProgram 2"
  "\n\r Press key...");
  getch(); } /* 95 */
/* ----- */
void SubProgram3() /* 示意性子程序 3 */
{ printf("\n\r this is SubProgram 3"
  "\n\r Press key...");
  getch(); } /* 100 */
/* ----- */
main() /* 主程序 */ /* 102 */
{ char * Choice; int stop;
  Choice = (char *) malloc(2);
  clrscr(); /* 105 */
  do { SelectWindow(0); stop=0;
    Menu(); /* 显示菜单 */
    ChooseMenu("Choice[1—3, Q]",
      Choice, "1"); /* 选择 */
    switch(* Choice) /* 按选择分流 */
    { case '1': SelectWindow(1);
      SubProgram1(); break;
      case '2': SelectWindow(2);
      SubProgram2(); break;
      case '3': SelectWindow(3);
      SubProgram3(); break;
      case 'q': /* 117 */
      case 'Q': stop=1; break;
    } /* 准备退出 */
  }; } /* 120 */
  while(!stop); /* 检查退出条件 */
  clrscr(); /* 122 */
  printf("Good bye!")
    "\n\r Press <Enter>...");
  getch();
} /* 126 */

```

在西文 DOS 下显示汉字的一种方法

信阳陆军学院实验中心计算机室 李日林

一、问题的提出 目前,应用于 IBM-PC 及其兼容机上的软件十分丰富,给用户实际问题带来了极大的便利。但是,由于机器内存容量的限制,汉字 DOS 本身又要占用大量的内存空间,例如 CC-DOS2.1 的字库为 230K 多,系统启动后字库常驻内存,这样内存空间所剩不多,所以,在汉字 DOS 下使用较大型的软件,有时就会因内存不够而感到无能为力;另一方面,有时候我们自己编制的应用软件或程序中仅需要数量很少的汉字,例如,显示标题、菜单及提示信息等,如果在汉字 DOS 下使用它们,就会白白浪费大量的内存空间。许多用户在日常使用中也想出一些办法来解决这一问题,如提高编程质量以简化程序,扩充内存容量等等,但是,简化程序来节约内存是相当有限的,扩充内存总会需要额外的开销。那么,能不能在西文 DOS 下直接运行较大型的软件,而又能显示我们所需要的汉字呢?答案是肯定的!笔者通过对 CCDOS2.1 的汉字库 CCLIB 及显示驱动模块进行分析研究,较好地解决了上述问题。本文介绍在西文 DOS 下显示汉字的实现方法,具体的程序设计并给出一个程序实例,编程语言为 PC-BASICA,程序在 IBM-PC/XT 上运行通过,使用效果很好,供有兴趣的同行参考使用。

二、实现原理及程序设计方法 要想在西文 DOS 下显示汉字,首先必须得到汉字的字模点阵信息。众所周知,16×16点阵的汉字库中每个汉字的字模占32个字节,作为文件的一个记录。那么怎样确定某汉字在字库上的记录号呢?这和汉字库的结构及该汉字的区位码均有关系。

CCDOS2.1 的汉字库 CCLIB 中存放着 GB2312—80 中的符号和汉字的点阵信息,但两者的内容并不一一对应,其对应关系如表一所示。可以清楚地看出 CCLIB 中略去了不常用的部分(如6区和8区)及空白区,这样做是为了节省宝贵的内存空间。由此,我们可以根据下述关系式确定某汉字在 CCLIB 中的记录号(RECON):

$$\text{RECON} = (\text{QM} - 1) \times 94 + \text{WM}$$

其中 6 当 QM=9 时

$$\text{QM} = \text{QM} \quad \text{当 } \text{QM} > 1 \text{ 及 } \text{QM} \leq 15 \text{ 时}$$

$$\text{QM} - 8 \quad \text{当 } \text{QM} > 16 \text{ 时}$$

(式中 QM、WM 分别是汉字的区码和位码)。

表一 汉字库与 GB2312—80 的对应关系

| 汉字库的区号 | GB2312—80 的区号 | 内 容 |
|--------|---------------|----------|
| 1~5 | 1~5 | 符 号 |
| 6 | 9 | 制 表 符 |
| 7 | 7 | 大、小写俄文字母 |
| 8~79 | 16~87 | 一、二级汉字 |
| 无 | 6 | 大、小写希腊字母 |
| 无 | 8 | 拼音符号与字母 |
| 无 | 10~15 | 空 区 |
| 无 | 88~94 | 空 区 |

确定了汉字在汉字库中的记录号,我们就可以用 BASICA 中 GET 和 PUT 语句所具有的功能来实现汉字的显示。GET(X1,Y1)-(X2,Y2),A% 语句把以(X1,Y1)为左上角坐标,(X2,Y2)为右下角坐标的矩形区域中点的颜色读到数组 A% 中;PUT(X1,Y1),A% 语句则是将数组 A% 中的元素放到以(X1,Y1)为左上角坐标的矩形区域。数组 A% 的字节数 BITN 由下式确定:

$$4 + \text{INT}(((X2 - X1) + 7) / 8) * (Y2 - Y1)$$

所以数组元素的个数为 BITN/2,对于 16×16 点阵的汉字,由上式可算得数组的元素为 18,其中前两个元素 A%(0)、A%(1)的值为 0010H,也就是点阵区域的长和高,A%(2)~A%(17)中的值则为汉字的字模信息。

根据上述讨论,可以确定在西文 DOS 下显示汉字的程序算法:

(1)进入高分辨率图形方式,设定常数 A%(0)=16,A%(1)=16;

(2)打开字库文件 CCLIB,根据要显示的汉字的区位码确定记录号,并读出该记录;

(3)将字模点阵数据分别放入数组 A%(2)~A%(17)中;

(4)用 PUT(X1,Y1),A% 语句将汉字显示在指定位置。

三、程序实例 本文给出在西文 DOS 下显示汉字的一个程序实例,使用时只需用 PC-DOS 启动,并根据提示输入字库名及汉字的区位码即可。如果您编写的软件中需要显示汉字,可以将本程序改

为一个专门显示汉字的子程序,根本无需启动汉字操作系统,并且一定会发现,可用的内存空间比以往在汉字操作系统下工作时要大得多,以前无法运行的软件现在可以运行了!

在西文 DOS 下显示汉字,确实具有很大的实际意义,相信许多用户都有同感,本文仅提出一种设计思想,希望能起到抛砖引玉的作用,使更多的用户受益!

```
10 'XSHZ. BAS LRL/1991.4.20
20 ' THIS PROGRAM IS USED TO DISPLAY CHINESE IN PC-DOS
30 KEY OFF;CLS
40 SCREEN 2;OUT 985,12
50 DIM A%(18);A%(0)=16;A%(1)=16
60 INPUT "PLEASE INPUT HZKNAME: ";ZK$
70 OPEN ZK$ AS #1 LEN=32
80 FIELD #1,32 AS A$
90 CLS
100 FOR I=1 TO 19
110 LOCATE 1,10+3*I;READ QWM
120 GOSUB 220
130 NEXT I
140 DATA 2722, 5113, 4225, 5135, 0122, 5258,
4687,4636,0336,0347
```

```
150 DATA 0351, 4734, 4752, 4230, 2626, 5554,
1144,4882,0123
160 LOCATE 3,13;PRINT STRING$(56,"*")
170 LOCATE 5,1;INPUT "PLEASE INPUT DISPLAY
LOCATION(X,Y): ";X,Y
180 IF X<5 OR X>23 OR Y<1 OR Y>79 THEN
170
190 LOCATE 7,1;INPUT "PLEASE INPUT QWM: ";
QWM
200 LOCATE X,Y;GOSUB 220
210 CLOSE;END
220 ' SUBROUTINE;DISPLAY CHINESE
230 QM=INT(QWM/100);WM=QWM-QM*100
240 IF QM=9 THEN QM=6
250 IF QM>15 THEN QM=QM-8
260 RECON=(QM-1)*94+WM
270 GET #1,RECON
280 FOR J=1 TO 16
290 A%(J+1)=CVI(MID$(A$,2*J-1,2))
300 NEXT J
310 H=POS(0)*8-8;L=CSRLIN*8-8
320 PUT(H,L),A%
330 RETURN
```

动画显示程序设计及在模态分析中的应用

重庆大学 朱才朝

摘要 本文论述了在 IBM PC 机上进行动画显示的基本原理方法及加快动画速度的汇编编程。作为一个实例,介绍了动画技术在模态分析中的应用。

一、引言 计算机动画显示技术(Computer Animation),是指屏幕上显示出来的画面或画面中的一部分,能够按一定规则或要求在屏幕上进行活动,使计算机显示输出的图形具有更强的真实感。动画显示是工程分析和设计中一种最直观、最形象、最实时的输出方式。

在计算机上实现动画,从原理上讲与电影动画片的制作是一样的,它们都是利用人们眼睛的一种视觉暂留现象,即当一幅画面在眼睛前面移去以后,画面的内容并不会在视网膜上立即消失,而是要保留在视网膜上短暂的一段时间,如果在这段时间内有另一幅与其相似的画面出现,就会同时有两幅画面在视网膜上,其区别仅在于很短时间一个

较小的动作差别。一组连续的幅面不停地出现,就自然形成一个连续动态过程。在计算机上实现动画,我们可以利用计算机及外设,采用编程实现。在程序中对动画对象反复地进行如下操作:先显示出该对象的图形,再进行变换,然后擦去原来的图形,最后再显示出变换后所得的新图形。

二、动画程序设计 在 IBMPC 上实现图形动态显示有各种方法,其实质都是连续显示一系列静止画面使画面活动起来。下面介绍一些常用方法及各自特点。

1. 利用多个轮流显示的页面来实现动画。

在字符显示模式下,16K 字节的缓冲区被分为 8 页(屏幕宽为 40 时)或 4 页(屏幕宽为 80 时),IBMPC

BASIC 语言提供了一个 SCREEN, AP, VP, 利用它可任指定其中某一页为当前显示页面 (VP) 或工作页面 (AP), 且两者可以不同。这样, 我们可在显示某一页面图案的同时, 可在另一当前工作页面画出新的图案, 然后再把工作页面设置显示页面。这样, 新的图案代替旧的图案。如此反复, 就好像画面在不断变化。

这种动画实现方法直观、简单, 但闪烁厉害。且只能在字符状态下使用, 图案比较粗糙。

2. 利用交替画—擦法实现动画。

所谓交替画—擦法是利用绘图语句在某一位置画出结构图后, 紧接着用相反的方式重画该位置结构图, 即从屏幕上擦掉它, 然后又用绘图语句在另一位置画出其结构图, 再擦掉它, 这样“画—擦—画……”循环下去获得动画效果。

这种方法要求绘图速度快。在图案简单的情况下是可行的。否则会出现明显闪烁, 以至无法满足要求。

3. 利用图形存取法来实现动画。

所谓图形存取法, 是利用 IBM PC 机的高级 BASIC 语言中提供的图形功能语句 GEJ 和 PUT 或 TRUE BASIC 语言中 BOX Keep 和 BOX show 两语句, 将某一矩形区域内的图形存入某一数组或变量中, 然后根据需要将此图形显示到指定位置, 使用这些语句可避免擦去和重构, 提高了动画速度。对复杂图形及需填色和阴影图案也能获得满意的动画效果。

4. 利用汇编语言提高动画速度。

动画的实现还是要受到计算机的运算速度 (硬件决定) 和图形变换连续性的影响, 有时候我们可以利用汇编语言直接设计绘图和图形变换子程序, 来提高动画速度, 实现高水平动画。下面介绍利用汇编语言实现两个基本图形一点、线的程序框图。

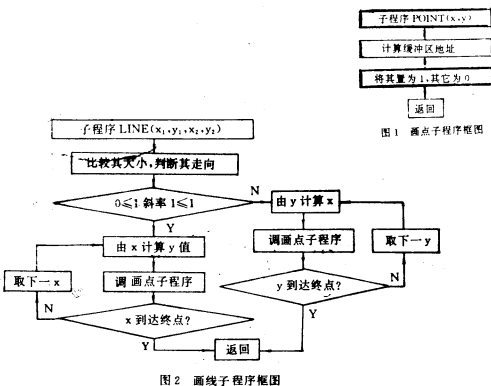


图2 画线子程序框图

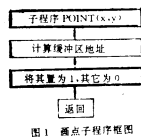


图3 画点子程序框图

三、动画模式技术 动画技术应用于模式显示十分有用, 它能够直观地、实时地显示各阶振型, 运动轨迹, 快速地向设计人员提供估计, 修改的理论依据。且可避免制作样机, 因而大幅度地节省研制费用, 缩短周期。为有效地分析和解决问题提供了极大的方便。

动画模式是由物体在平衡位置时的形状加上不同时刻的振动量组成。本人根据上节介绍的实现动画显示途径中的存取法, 利用 TRUE BASIC 语言编制了实现振型动画显示的程序, 其框图如下图 (3)。

作为一个实例, 我们在计算一悬臂梁固有频率和振型的基础上, 利用这一程序得到其某一阶固有频率下的动态显示过程。

当然这只是一个简单的例子, 随图形算法的研究和大规模集成电路技术的发展, 必将加快动画技术的发展。

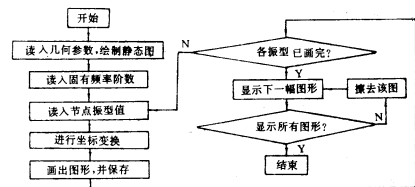


图3 振型动画显示程序框图

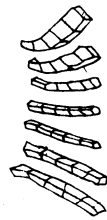


图4 悬臂梁振型动画显示

参考文献

- [1] 张福炎、蒋新儿、李滨宇:《微型计算机 IBM PC 原理与应用》, 南京大学出版社, 1984 年。
- [2] 孙开明:“三维动画技术在 IBM PC 机上的实现及应用”,《微型计算机》, 1988. 1。
- [3] John·G·Kemeny·Thomas E·Kurts 等:《TRUE BASIC》, 中科院软件研究所译。

DBASE III 实现的中文屏幕游动显示

四川省渠江水泥厂微机室 彭 立

一、问题的提出 我们经常在电视上看到这样一种现象,当电视节目发生变动时,电视台的通知会在电视屏幕的最下面一行从右至左逐字游动显示,非常醒目,那么,我们是否可以在一般的微机配置的 CRT 显示屏上达到同样的目的呢?回答是肯定的,请参看下面的分析。

二、设计思想 根据我们所提出的问题,要达到游动显示的目的,我们可以充分利用计算机运行速度快的特点,不妨这样设想:对给定的中文字符串,我们可以从头到尾依次截取——最初截取第一个字,把它显示在屏幕的最右端;而后,截取头两个字,在屏幕上占据最右的两个位置;再截取头三个字……,当屏幕上的汉字字数达到40个时,改为从第二个字开始截取40个字显示在屏幕上,以后再从第三个字,第四个字起……,同样截取40个汉字予以屏幕显示,如此这般,用户将看到屏幕上的中文字符串从右到左不断移动,并且,显示屏的最右一侧也不断地增添着新的内容,直至所给定的中文字串的内容全部在屏幕上显示完一遍,最后一个字位于屏幕的最左端为止。

三、程序介绍

1. 变量说明

X——待显的中文字符串

Z——X 显示次数的控制变量

I——在屏幕显示汉字字数 ≤ 40 时,I以2递增,控制所截取的字符串长度;在屏幕显示汉字的字数 > 40 时,I以1递增,控制着显示完后继字符所需的次数

K——从头到尾显示 X 一遍所需的总次数(直至给定字符串的最后一个字显示在屏幕的最左端)

M——屏幕显示行最左列的控制变量

N——截取字符串的起始位置控制变量

J——屏幕暂停时间控制变量

2. 程序说明

程序的主体是“DO WHILE I \leq K”这一循环。在这一循环语句中,又有一个条件判断语句“IF M $<$ 1”,通过这一条件判断,程序截取相应长度的字符串,并在相应的屏幕位置显示出来。

3. 注意事项

为保证屏幕显示的结果不出现混乱,截取字符串时,控制变量必须以偶数递增,从而保证截取到一个

完整的汉字。

四、使用效果 在设计管理程序的过程中,我们经常要在屏幕上为用户提供一些相关信息,帮助用户更好的工作,采用以上方法设计信息行,一方面增强了动感,丰富了人机界面,使屏幕设计更为灵活;另一方面,能更好地提高计算机操作人员的注意力,尽可能地减少误操作,以取得更好的工作成效。

五、程序清单

```
C>type c:\pl1.prg
set talk off
x="长城计算机公司于八九年宣布了一项重要声明
如下所示"
z=1
do while z<=2
    i=2
    n=1
    k=39+len(x)
    set talk off
    set exact on
    m=79
    do while i<=k
        if m<1
            m=1
            n=n+2
            y=subs(x,n,80)
        else
            y=subs(x,n,i)
        endi
        clear
        @5,m say y
        j=1
        do while j<=20
            j=j+1
        endd
        m=m-2
        if i>40
            i=i+1
        else
            i=i+2
        endi
        endd
        z=z+1
    endd
endd
retu
```

DOS 系统下的脱机输出

四川省建筑设计院 刘锦林

摘要 本文介绍一种实用的脱机输出方法,利用此种方法在主机输出的同时可运行其他软件;当数据全部进入缓冲器后,即使关闭主机电源,输出仍可继续完成。

DOS 系统中有一个外部命令 PRINT.COM 用于字符文件的假脱机输出,对于提高主机使用效率能起一定作用,但是它的能力是相当有限的。脱机后若是运行一个内存需求量较大的程序其结果前后台操作都变得相当缓慢。特别是脱机一个上百 K 的 DWG 图形文件而同时在前台屏编一个数十 K 的文件时,几乎无法忍受那种缓慢。脱机的目的主要是希望再次使用 CPU,而 PRINT.COM 离开主机 CPU 的控制却无法从任何一个内存地址调数据到输出口。因此,仅仅想依靠扩展内存来提高脱机输出的效率,并没有抓住问题的本质。目前一些 386 486 机型上配置的 SUPERSPL 之类的软件也仅仅是利用了扩展内存。只有想法减轻主机 CPU 的负担才能得到最好的效果。

本文介绍一种实用的脱机输出方法,利用这种方法在输出的同时主机上可以同时运行其它较大的软件,如 PK, TBSA, AUTOCAD 等,大大提高了主机使用效率,当数据全部进入缓冲器之后即使关闭主机电源,输出过程仍能继续完成。

笔者自今年四月以来使用四川华明电脑公司研制的 DBF—B 型数据传输缓冲器在 286 386 486 机型上实现了脱机输出。效果很好,相当于多了一台主机提供你使用,其花费不过 3200 元。

AUTOCAD 2.5 以上的版本已具有产生脱机文件的功能。这种文件是一个字符文件, AUTOCAD 按照你所选定的绘图机类型由 DWG 矢量文件产生一个该类型绘图机执行代码的字符文件,它的字节数要比 DWG 文件大得多。进入 AUTOCAD 输出一张 1# 图大约需要 40 分钟(笔速 4"—8"/秒),这段时间内主机完全被占用。AUTOCAD 一面把 DWG 文件转换成绘图机代码一面向绘图机传送,而且是转换一段传送一段。由于未退出 AUTOCAD 所以无法运行其它程序。如不采用缓冲器在 DOS 下用 PRINT 脱机的话,当文件字节数超过数十 K 时,脱机后主机再运行普通一个屏幕编辑程序,主机反应都很慢。击键后数秒才有反应(286),即使是 386, 486 机这种反应也很明显。目前一些 386, 486 机型上虽然配备了如:磁盘 CACHE, 内存 CACHE, SUPERSPL 之类的功能,但

如果真要脱机输出一个数百 K 的 DWG 文件,仅仅依靠主机内部的 RAM 效果仍然不理想。特别是缓冲内存的大小很难选定,小了作用不大,大了输出时比不设缓冲内存还要糟糕。即使选择了一个合适的大小,其效果也远不如加接外部缓冲器。要想同时运行 AUTOCAD, TBSA 之类的软件是极其困难的。造成这种现象的原因主要是 DOS 不是一种多用户多任务的操作系统而且访问内存的能力一般仅 640K, 尽管 SUPERSPL 之类的软件可以访问扩展内存,但这种访问必须通过主机 CPU 才能实现。接入缓冲器后完全克服了这一现象。当初缓冲器容量仅 128K 但效果已经十分明显,目前容量已经扩大到 1M,这对于 DWG 文件的输出就更加有价值。DWG 文件转化为 PLT 文件,字节数增加 1.6—4.4 倍甚至更多,一张内容复杂的 1# 框架施工图其 PLT 文件字节数可近 1M,用缓冲器脱机输出后,主机仍然可以立即运行 AUTOCAD 这么大的软件,对输出看不出任何影响。这时主机起了两部主机的作用。在 AUTOCAD 中产生一个脱机文件十分简单,你可仍旧按 PLOT 命令的操作过程进行,只是在出现 Write the plot to a file? (N) 的提示时敲 Y,在后来的文件名询问时直接敲回车,就能生成一个与原图形名同名的 *.PLT 脱机文件。退出 AUTOCAD 在 DOS 提示符下用 PRINT *.PLT 脱机绘图机立即开始工作,屏幕上马上又出现 DOS 提示符 C)。这时你可以任意使用主机。

PRINT.COM 是 DOS 系统的外部命令,它有一些任选项如:

/D: 用于指定脱机设备,其参数可以为 PRN(隐含)、COM1、COM2...等

/B: 指定内部缓冲大小(隐含 512)

/Q: 指定脱机文件个数(隐含 10)

/S: 指定时间片,可为 1—255(隐含 8)

还有 /U:, /M; 等,弄清这些任选项的作用可以使 PRINT.COM 的使用更加灵活,有关详细资料请查阅 DOS 手册。

有些关于 AUTOCAD 的资料上,在介绍伪脱机绘图时,介绍了诸如:建立伪脱机绘图目录、确立伪脱机绘图文件名……等等步骤。其中提出了如 AU-

TOSPOOL 这样的文件名,在 DOS 系统下根本行不通(文件名长于8字符)。况且这些资料交流仅仅介绍了如何生成脱机文件,究竟怎么输出这些文件却没有交待。

经我们试验根本无须经过这些繁琐步骤,仅按前面介绍的方法完全可以生成 *.PLT(绘图机)*.LST(打印机)脱机文件。

除了 PRINT 可以传送数据,DOS 命令中的 TYPE COPY 也可以用来传送数据。使用并口发送数据要比串口快得多,一个字节数为556921的数据文件经不同口发送,在 AST486/25上的试验结果是:串口为10分39秒,并口为2分6秒。可见采用并口发送要快五倍多。(大概每分钟200K 左右)为使数据尽快脱离主机建议尽量使用并口发送。缓冲器本身具有串——并转换功能无论连接串行或并行输出设备均很方便。

以上试验均使用“COPY 文件名[COM1,PRN……]”的命令格式。如果要向 COM1传送,事先必需使用 MODE 命令对串口进行设置。如 MODE COM1:96,N,8,2具体格式由需要而定。

传送时间的计算是用自编程序进行的。使用 TYPE COPY 命令来传送数据借助提示符 C>的出现可以比较清楚的观察到传送过程何时结束。使用 PRINT 传送数据则较难判断数据传送何时全部结束,这种情况下数据传送是分时间片进行的(可用/S,XXX 来确定时间片),每次传送数据前总会从硬盘读取数据,所以硬盘指示灯按一定时间间隔闪烁。可以根据这个现象判断传送过程是否结束。更简单

的方法是按经验来判断,在 AST486/25的机器上如果敲 PRINT 命令后同时运行一个其它的程序,大约每分钟还可以传送数据120K 左右。在确认数据已经脱离主机后,如果接有外部缓冲器的话即使关闭主机电源,输出过程仍可继续完成。使用 PRINT 命令的另一个好处是发命令后可以立即使用主机。

所以,笔者认为利用外部缓冲器来弥补 DOS 寻址能力薄弱,利用缓冲器的 CPU 弥补 DOS 单用户任务的缺点,使用 DOS 现有的 PRINT.COM 产生立即脱机的效果,利用缓冲器的串——并转换功能联接串行绘图设备,不失为一种简单可行而又经济的方案。

缓冲器大大减轻了主机内存和 CPU 的压力,这对寻址能力一般仅640K 的 DOS 系统来说,实在太有意义。由于 CAD 软件的普遍使用,各单位的绘图需求量将日益增大,如果386,486主机仅仅为了输出图形文件或打印文件被长时间占用,其使用率就太低了。多用户系统上也能使用这种设备,效果同样好。为提高主机使用效率延长主机寿命,笔者认为应当大力推广脱机输出,以使各单位有限的设备发挥更大的效益。

编者按:四川华明计算机技术公司开发的 DBF—B 高速数据缓冲器使外设与主机间实现了真脱机,串行传输速率达1200~9600BPS,并具有并——并、并——串、串——串、串——并传输转换,缓冲容量显示和串行传输出错,自检出错显示等功能,无需任何软件支持,即可投入工作,缓冲容量分为128K、512K、1024K 字节等档次。现已批量投产。

FORTRAN 语言图形功能的扩充

合肥工业大学计算中心 邵庆良 沈尔铁

摘要 本文将讨论高级语言功能扩充的方法;介绍了在 FORTRAN 语言中扩充图形功能的实现过程并附有相当实用的汇编语言和 FORTRAN 语言子程序。

一、问题的提出 FORTRAN 语言由于有极强的科学计算功能,因此自1957年以来一直是科学和工程领域中使用的最广泛的一种程序语言。然而,随着计算机技术的发展,计算机要处理的问题已不再只是科学计算,而是多方面的,如:数据管理、分析计算、绘制图形等。特别是 FORTRAN 语言缺乏图形功能,给用户带来诸多的不便。本文分析了 FORTRAN 与汇编

语言 CALL 调用的实质,提供了 FORTRAN 语言图形功能的扩充方法。

二、FORTRAN 语言 CALL 调用的实质 高级语言允许部分编译,形成的目标模块由连接器(Linker)装配在一起形成执行文件。连接时缺省的模块,如函数调用及 CALL 过程均认为在指定库(FORTRAN.LIB)中,因此,扩充的模块可放入 FORTRAN.LIB

中,从而在连接时就不需特别指出。这样,用户可以通过自行编制的模块来扩充或更新 FORTRAN 库达到扩充功能的目的。

扩充图形功能是通过调用汇编语言的图形显示程序来实现的。在 FORTRAN 语言程序调用汇编语言子程序时,要解决的一个关键问题是两种语言之间的调用接口。在微机,上, FORTRAN 语言程序调用汇编子程序仍然使用 CALL 语句,而实际的数据传送是通过一个公共的堆栈进行,调用时的地址也保存在该堆栈中。如图1所示为 FORTRAN 程序在调用汇编语言时堆栈中的内容。

在编写汇编子程序时应注意以下几点:

● far 过程的名子必须被指出为 public。

● 对 DATA 段(segment)的组 group,一个 group 语句必须用 group IDGROUP 包括。

● 假设 CS 指向代码段, DS, ES, SS 指向 DGROUP。

● 必须保存和恢复调用者 BP 和 SP。调用者 CS 和 IP 用 RET4 * n(n 为参数个数)指定恢复,假如要改变调用者的 DS,则也必须恢复。

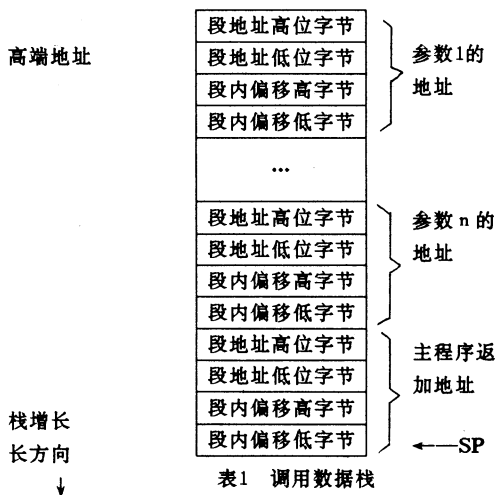


表1 调用数据栈

三、图形功能模块的设计 采用汇编语言编制图形显示程序,至少需要编写设置显示方式模块和图形方式下画点模块。如再加上屏幕色彩设置模块就可以实现各种色彩的多种图形。图2至图4为这三个基本模块的汇编子程序。

SCREEN(MODE)

```
STACK    STRUC
          SAVEDS    DW?
          SAVEBP    DW?
          SAVERET   DD?
          MODE      DD?
```

```
STACK    ENDS
DATA     SEGMENT PUBLIC 'DATA'
DATA     ENDS
CODE     SEGMENT
DGROUP   GROUP DATA
          ASSUME CS:CODE,DS:DGROUP,ES:
          DGROUP,SS:DGROUP
SCREEN    PROC FAR
          PUBLIC SCREEN
          PUSH BP
          PUSH DS           ;保护现场
          MOV BP,SP        ;设置栈指针
MODE0:    LES BX,[BP]+MODE
          MOV AH,ES:[BX]   ;传颂 MODE 参数
          CMP AH,0H
          JNE MODE1
          PUSH AX
          MOV AH,0H        ;调用0号模块
          MOV AL,02H       ;设置80×25黑
          INT 10H          ;白文本显示方式
          POP AX
          JMP ENDD
MODE1:    CMP AH,01H
          JNE MODE2
          PUSH AX
          MOV AH,0H        ;调用0号模块
          MOV AL,04H       ;设置320×200彩
          INT 10H          ;色图形显示方式
          POP AX
          JMP ENDD
MODE2:    CMP AH,02H
          JNE ENDD
          PUSH AX
          MOV AH,0H        ;调用0号模块
          MOV AL,06H       ;设置640×200黑
          INT 10H          ;白图形显示方式
          POP AX
          JMP ENDD
ENDD:     POP DS
          POP BP           ;恢复现场
          RET 4            ;返回
SCREEN    ENDP
CODE     ENDS
          END
```

图2 设置显示方式汇编子程序
COLOR(N1,N2)

| | | | | |
|--------|---------------------------|-----|--------|---------------------------|
| STACK | STRUC | | Y | DD? |
| | SAVEDS | DW? | X | DD? |
| | SAVEBP | DW? | STACK | ENDS |
| | SAVERET | DD? | DATA | SEGMENT PUBLIC ' DATA ' |
| | N2 | DD? | DATA | ENDS |
| | N1 | DD? | CODE | SEGMENT |
| STACK | ENDS | | DGROUP | GROUP DATA |
| DATA | SEGMENT PUBLIC ' DATA ' | | | ASSUME CS:CODE,DS:DGROUP, |
| DATA | ENDS | | | ES:DGROUP,SS:DGROUP |
| CODE | SEGMENT | | SETDOT | PROC FAR |
| DGROUP | GROUP DATA | | | PUBLIC SETDOT |
| | ASSUME CS:CODE,DS:DGROUP, | | | PUSH BP |
| | ES:DGROUP,SS:DGROUP | | | PUSH DS ;保护现场 |
| COLOR | PROC FAR | | | MOV BP,SP ;调置栈指针 |
| | PUBLIC COLOR | | | MOV AH,12 ;调用12号模 |
| | PUSH BP | | 块 | |
| | PUSH DS ;保护现场 | | | LES BX,[BP]+COLOR |
| | MOV BP,SP ;设置栈指针 | | | MOV AL,ES:[BX] ;传送 COLOR |
| | PUSH AX | | 参数 | |
| | PUSH BX | | | LES BX,[BP]+Y |
| | MOV AH,0BH ;调用11号模块 | | | MOV DX,ES:[BX] ;传送 Y 参数 |
| | LES BX,[BP]+N2 | | | LES BX,[BP]+X |
| | MOV BH,ES:[BX] ;传送 N2参数 | | | MOV CX,ES:[BX] ;传送 X 参数 |
| | LES BX,[BP]+N1 | | | INT 10H |
| | MOV BL,ES:[BX] ;传送参数 | | | MOV SP,BP |
| | INT 10H | | | POP DS |
| | POP BX | | | POP BP ;恢复现场 |
| | POP AX | | | RET 12 ;返回 |
| | POP DS | | SETDOT | ENDP |
| | POP BP ;恢复现场 | | CODE | ENDS |
| | RET 8 ;返回 | | | END |
| COLOR | ENDP | | | |
| CODE | ENDS | | | |
| | END | | | |

图3 设置背景色和调色板汇编子程序

;SETDOT(X,Y,COLOR)

```

STACK  STRUC
        SAVEDS      DW?
        SAVEBP      DW?
        SAVERET     DD?
        COLOR       DD?

```

图4 图形方式下画点汇编子程序

如果仅有上面介绍的画点子程序,则在画复杂图形时,需要反复调用画点子程序,其结果必然降低效率。因此,在实际应用中,还要增加一些标准图形生成子程序,如画线子程序、画圆子程序、画弧子程序、填色子程序等。作为例子我们给出了画线和画圆子程序,如图5所示。用户可以据此编写和扩充其它的图形生成子程序,也可以建立适合于自己专业的专用图形子程序。

```

subroutine line(x1,y1,x2,y2,N)
  integer x1,y1,x2,y2,N
  d1=iabs(x2-x1)
  if(iabs(y2-y1).gt.d1) d1=iabs(y2-y1)
  if(d1.eq.0.)return
  dx=(x2-x1)/d1
  dy=(y2-y1)/d1
  x=x1+0.5
  y=y1+0.5
  do 10 i=1,int(d1)
    call setdot(int(x),int(y),N)

    x=x+dx
    y=y+dy
10  continue
    erturn
end

subroutine circle(x0,y0,r,n)
  integer x0,y0,r,n,x,y,x1,y1
  pi=3.141593
  theta=pi/90.0
  a=theta
  y1=y0
  x1=x0+r
  do 10 i=1,180
    x=r*cos(a)+x0
    y=r*sin(a)+y0
    a=a+theta
    call Line(x1,y1,x,y,n)
    x1=x
    y1=y
10  continue
    return
end

```

图5 画直线和圆的子程序

四、FORTRAN 库的扩充和调用 FROTRAN 库管理程序 LIB. EXE 是一个实用程序,它可以帮助你建立、组织和维护 FORTRAN 扩展库。使用 LIB. EXE 库管理程序,将我们自己设计的图形生成文件(OBJ)扩充到 FORTRAN 扩展库中去,就可以实现 FORTRAN 库的功能扩充。下面是扩充的过程。

(1)汇编或编译成 OBJ 文件:

C>MASM 文件名; 或 C>FOR1 文件名;
C>FOR2

(2)利用库管理程序(LIB. EXE)扩充 FORTRAN 库:

```

C>LIB
Library name:FORTRAN
Opration: + 目标文件名(如 SCREEN)
Listfile:      (回车不建立交叉列表文件)

```

Output library: (回车存入原库文件名下)

在使用图形功能时,在主程序中用 CALL 语句调用,数据传送用哑实结合的方式,编译过程与未扩充前相同。下面是前面给出的几个图形生成模块的参数说明。

●屏幕显示方式设置 SCREEN(MODE)

MODE=0, 80×25黑白文件显示方式

1, 320×200彩色图形显示方式

2, 640×200黑白图形显示方式

其它显示方式,用户可以自己添加。

●背景和调色板设置 COLOR(N1,N2)

N1——背景颜色号(0—45)

N2——调色板号(0或1)

●画点 SETDOT(X,Y,COLOR)

X,Y——点的坐标

COLOR——点的颜色号

●画线 LINE(X1,Y1,X2,Y2,COLOR)

X1,Y1,X2,Y2——直线两端点的坐标

COLOR——点的颜色号

●画圆 CIRCLE(X0,Y0,R,COLOR)

X0,Y0——圆心坐标

R——圆的半径

COLOR——圆周线的颜色号

五、结束语 我们针对 FORTRAN 语言缺乏图形功能所带来的不便,自行扩充了图形功能。在本文中介绍了设计的基本思想和扩充方法以及实现过程。这种扩充方法简单易行,用户根据这种方法,将会很容易地克服在实际应用中 FORTRAN 语言缺乏图形功能所带来的困难。

参考文献

[1]查良琦等编著《实用微型计算机图形学》,电子工业出版社,1990.6.

[2]《IBMPC 计算机系统/应用软件开发》,中国科学院希望电脑技术公司,1988.1.

日历时钟芯片 M5832 与微机

接口设计一种实现方法

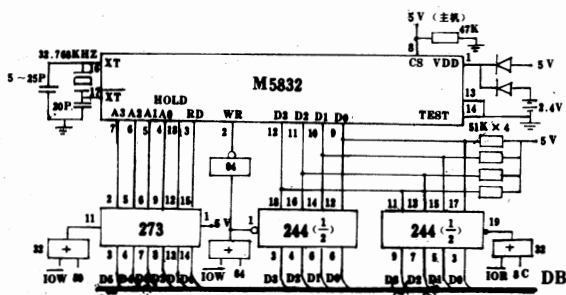
辽宁大学计算机系 卢正义

在智能化仪表、微机应用系统中,经常要记录采集数据或工作执行时间。日历时钟芯片 M5832 由于具有功耗低、功能齐全,易于调整的特点,因此为我们使用提供了很大方便。

M5832 接口硬件电路常接 Z80—PIO, 以实现其功能,这是一种典型应用。但是在实际应用中, Z80—PIO 两个 I/O 端口 A、B 往往被 A/D 转换、控制电路所占用。为此我们设计另一种接口电路实现了 M5832 的功能。如图接口硬件采用 74LS237、74LS244、74LS32 各一片,一位反相器可使用系统中 74LS04。电路中端口 80H 控制 74LS273 锁存器,通过数据总线 D_0-D_7 锁存读写时间的地址、计数保持 (HOLD) 和读信号。端口 84H 控制时间、日期从数据总线 D_0-D_7 通过 74LS244 缓冲器写入,此时 WR 为高电平, RD 为低。端口 8CH 控制读出年月日时分秒通过 74LS244 从数据总线 D_0-D_7 输出,此时 RD 为高电平, WR 为低。系统不工作时, M5832 片选端 CS 接地,此时它在备用电池供电下工作。系统工作,电源接通,片选端 CS 接 5 伏,这样通过 Z80 汇编语言程序可以对 M5832 进行读写以实现校时,显示或打印时间的目的。

软件实现很简便,对 M5832 进行读写基本思路是:首先置计数保持 (HOLD) 为高电平,冻结计数;然后给出时间相应的地址,读写时间数据;最后置计数

保持 (HOLD) 为低电平,这样 M5832 解冻,继续计数工作。由于时间数据是在给出时间地址以后 $6\mu s$ 才能得到,系统主频为 2M,则周期 T 是 $0.5\mu s$,而一条空操作指令周期 T 为 4,这样三条 NOP 指令执行时间应是 $6\mu s$ 。因此在给出时间地址以后执行三条 NOP 指令就足以保证读出正确。



在单片机应用系统中,由于 8031 单片机 P_1 口、 P_3 口都具有锁存功能,因此采用上述接口,电路十分简单,只需略加改动:用一片 74LS244 代替原 74LS273,共输入端与 $P_1.0-P_1.5$ 相连,共控制端 1 脚, 19 脚与 $P_3.0$ 接,原 74LS244 片控制端 1 脚接 $P_3.1$, 19 脚接 $P_3.2$,去掉原 74LS32,其它线路不动。这种接口电路无论在 Z80—CPU 还是在 8031 单片机应用系统中都成功实现了,效果很好。

(上接 18 页)

Calder. Composing User Interfaces with Interviews. IEEE computer, 22(2):8-22, February 1989.

[7] Paul J. Asente. Editing Graphical Objects, Using Procedural Representations. Ph. D. thesis, Dept. of Computer Science, Stanford University, Stanford, CA, 1987.

[8] Paul R. McJones and Garret F. Swart. Evolving the UNIX System Interface to Support Multithreaded Programmes. In Proc. winter 1989. USENIX Technical Conference, pages 393-404, USENIX Association, Berkeley, CA, 1989.

[9] Paul Rovner. Extending Modula-2 To Build Large,

Integrated Systems.

[10] Prototyper, Smether Barnes, Portland, OR.

[11] R. Daris, 1980, "Interactive Transfer of Expertise, Acquisition of New Inference Rules". In B. L. Webber and N. J. Nilsson (ed). "Readings in artificial intelligence", Palo Alto, Calif.: Tioga, pp. 410-428.

[12] T. Parlidis, Algorithms for Graphics and Image Processing, Computer Science Press Inc. 1982, Rockville, pp. 2-20.

一种实用的双机并行通讯方法

东南大学热能所 周立

南京师范大学物理系 王晓红

摘要 本文提出一种简单实用的、全双工的并行中断通讯物理链路,介绍了它的设计、工作及特点。

一、引言 在工业控制和管理中,经常要遇到多机通讯协调管理的问题,如果采用计算机网,一方面成本较高,另一方面计算机网对于所用机型又有限制。因而经常采用 RS—232或 RS—422串行通讯,但其速度较慢,常常不能满足实时性很强的场合。笔者为多组逆变器系统的协调控制设计了一种高质量的双工并行通讯接口,采用中断方式在 PC 主机和 STD 工业控制机之间进行通讯,已投入现场运行。因其具有较高的性能价格比,现已被多个项目采用。

二、方案的实现 并行通讯接口硬件如图一所示。采用 intel8255作并口基本单元,7400与非门、7432或门以及7431延时门作通讯握手信号处理。

1、电路设计原理说明

根据8255并口通讯的时序要求,推导出接口调整电路时序,进而设计出接口电路。该通讯链路在完成数据发送与接收时的时序如图二所示。

2、通讯物理链路工作原理

并口 J₂ 被初始化为 B 口输出、B 口输入的方式 1, 并口 J₃ 被初始化为 A 口输入、B 口输出的方式 1, 中断请求 INTR 反应了各口的状态, 接至各个计算机的中断申请端 (例如中断管理芯片 8259 的 INTR_x 端)。数据收发过程如下: 当发送方发出数据后, 发送方的 INTR 由高电平有效变为低电平无效, 接收方的 INTR 被接口电路同时置为有效, 当接收方未读取数据 (CPU 执行读口操作) 之前, 发送方和接收方的 INTR 保持不变, 只有当接收方读取数据后, 接收方的 INTR 无效, 同时发送方的 INTR 被接口调整电路置为有效, 再次申请 CPU 发送数据。当所有数据都发送完毕后, CPU 即屏蔽发送中断申请。

三、总结

该并行通讯链路具有优越的性能:

1、双工方式,发送和接收独立工作,互不影响。

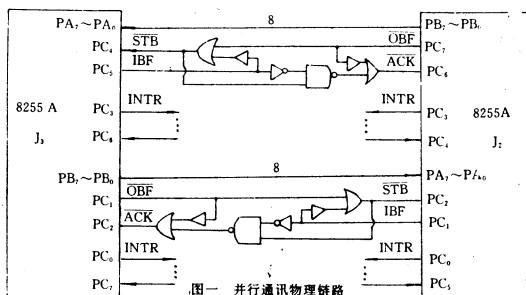
2、中断方式,实现了高速并行通讯。

3、握手信号采用硬件调节，免却软件调节握手关系之苦。

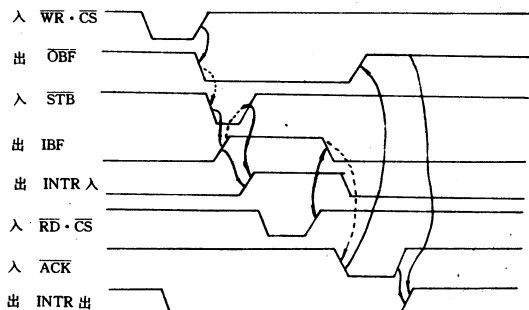
4、适用于各种机型之间高速通讯。

5、也可以用于查询方式。只需将 INTR 分别接至 8255 未用的 C 口某些位端,例如将并口 J₂ 的 INTR_A 接到 C₄ 端,INTR_B 接到 C₅ 端,软件查询 C₄ 和 C₅ 的状态即知 INTR 状态。

6、采用延时门设计增加了通讯可靠性,可用于环境恶劣的电力实验室,而满足同样时序关系的D触发器电路通讯出错率较高。



图一 并行通讯物理链路



图二 并口发送与接收时序

PC 机与 TMS320 系列数字信号

处理器的接口设计

机电部第二十七研究所 王大可

摘要 本文介绍了 PC 机与 TMS320 系列数字信号处理器的一种接口方法,为利用 PC 机强大的功能快速开发 TMS320 系列 DSP 开辟了一个途径。

一、问题提出 TMS320 系列数字信号处理器以其独特的高速、实时及灵活性,在图象处理、语言处理、频谱分析、数字滤波、实时控制等方面得到广泛应用。

但是,使用 TMS320 需得到开发工具和应用上如全速仿真器、软件模拟程序等的支持。对于没有这些开发工具的单位来说,给开发利用 TMS320 系列带来了一定的困难。为此,我们自己动手设计了一个 PC 机与 TMS320 数字信号处理器的接口。

二、PC 系列机扩展插槽的信号分布 在设计 PC 机与 TMS320 数字信号处理器的接口之前,需了解 PC 机扩展插槽的信号分布及作用。图 1 为 PC 系列机扩展插槽的信号分布。

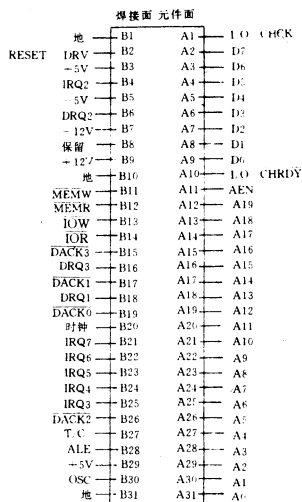


图 1 PC 系列机扩展插槽信号分布

PC 系列机扩展插槽中,有:8 位双向数据总线;20 位地址总线;6 根中断信号线;3 根 DMA 控制线;4 根电源线以及其它各种控制线共 62 根。按其功能可分为五类:

1. 地址线 A19~A0(20 根)

地址线用来指出内存地址或输入输出地址。当用来指出输入输出地址时,A19~A10 无效。

2. 数据线 D7~D0(8 根)

3. 控制信号线 (21 根)

控制信号线中,6 根中断请求线(IRQ2~IRQ7);3 根访问内存请求线(DRQ1~DRQ3);4 根 DMA 请求认可信号线(DACK0~DACK3);2 根控制内存进行数据读出或写入的 $\overline{\text{MENR}}$ 和 $\overline{\text{MENW}}$ 信号线;1 根 ALE 信号线;1 根 T/C 信号线; $\overline{\text{IOR}}$ 和 $\overline{\text{LOW}}$ 一般用来指出 CPU 正在执行的是输入指令还是输出指令,所以,数据总线上的数据是与输入输出控制器有关的;RESETDRV 是系统总清信号,使系统初始化;AEN 信号用于 DMA 操作,当它为高电平时。所有地址线、数据线及 $\overline{\text{IOR}}$ 、 $\overline{\text{IOW}}$ 、 $\overline{\text{MENR}}$ 、 $\overline{\text{MENW}}$ 等均受 DMA 控制器的控制,而不再受 CPU 的控制。

4. 状态线 (2 根)

5. 辅助线和电源线(11 根)

PC 系列机通道上输入输出设备的编址可达 512 个。其中大部份已被其它选件板如:软硬盘控制器、彩色图形适配器、打印机适配器等占用。根据应用要求,我们在剩余端口地址中选取 3COH~3CFH(实际只用了 3COH~3CBH)地址作为我们 PC 机与 TMS320 专用数字信号处理机的接口地址。

三、接口硬件设计 硬件接口电路一般有两种工作方式,一种是内存工作方式,另一种是输入/输出(I/O)工作方式。内存工作方式是把 TMS320 专用数字信号处理机的存储器作为 PC 机的内存,故而存取灵活,存储速度快,缺点是连线较多,不利于调试,且受外界干扰的影响较大。而 I/O 工作方式是把 TMS320 专用处理机的存储器作为 PC 机的一个外部输入输出设备,不需要外地址线,缺点是随机存取不方便。综合各种因素,我们采用 I/O 工作方式。引出线共有 15 根,其中 8 根数据线;4 根低位地址线;1 根“3C”信号线;1 根 $\overline{\text{IOR}}$ 信号线;1 根 $\overline{\text{IOW}}$ 信号线。采用 25 芯 D 型插头座连接很方便。

接口电路的框图如图 2 所示。一部分插在 PC 机的扩展插槽中,主要完成 3COH~3CFH 地址中“3C”

位选通译码,以及数据线,控制线和低4位地址信号线的缓冲。另一部分安装在跟踪系统中,主要完成各端口地址的译码、地址计数及与 TMS32020 专用处理机的连接。

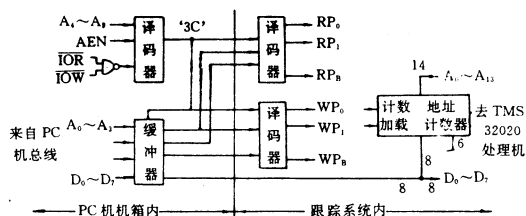


图2 接口电路框图

“3C”位选通译码电路输出“3C”信号,此信号低电平有效。它是当地址线 $A_9 \sim A_4$ 为 CH, AEN 信号为低电平(非 DMA 工作期间。DMA 工作期间 CPU 被挂起,无法控制数据线、地址总线及 \overline{IOR} 、 \overline{IOW} 等信号线),且有输入输出指令执行时才有低电平输出。

“3C”位选通信号分别与 \overline{IOR} 、 \overline{IOW} 信号配合,完成端口的输入、输出选通工作。地址线的低4位经缓冲器后接至4—16线译码器的输入端,以产生各分机的选通信号及地址计数器的加载、计数的缓冲信号。在我们设计的接口电路中,可以与10个分机通讯,根据 $A_0 \sim A_3$ 信号的不同,分别选择不同的分机。根据需要,还可以扩展。

地址计数器原框图如图3所示。地址计数器采用可预置的二进制计数器,输出端口 3CAH(WPA)为地址计数器的加载控制脉冲,当它有效时,PC 机将 PC 机数据总线上的数据加载至地址计数器的预置端。

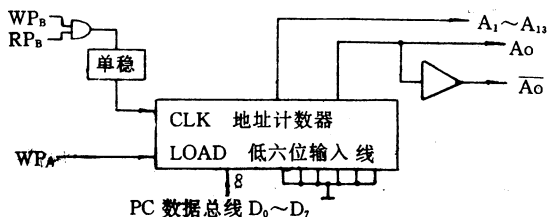


图3 地址计数器原理框图

由于 PC 机数据线只有8位,在我们的智能电视实时跟踪系统中 TMS32020 专用数字信号处理机的地址线需要13根,因此我们将 PC 机加载地址计数器的8位起始地址作为 TMS32020 处理机的段地址,

以此为基础计数。TMS32020 处理机的数据总线为16位,我们选取地址计数器输出端最低位 A_0 控制 TMS32020 处理机中程序、数据高、低8位存储器的选通。地址线 $A_1 \sim A_{13}$ (存储器地址线为 $A_0 \sim A_{12}$) 中低5位加载时为0,相当于段地址长度为32个单元。

输入输出端口 3CBH 触发一单稳态电路产生一脉冲作为地址计数器的计数脉冲,单稳态的作用是使地址计数器在完成读写操作之后再改变其状态。单稳态时延选择在200ns 左右。

接口电路设计中还包括接口命令锁存器及状态查询电路。由于这部分电路为各分机所拥有,故在接口原理框图中未画出。

当 PC 机在 3COH~3C9H 端口执行输出命令时, $WP_0 \sim WP_9$ 信号有效, WP_n ($n=0 \sim 9$) 控制将 PC 机的控制命令信息打入命令锁存器中。命令信息包括:

$\overline{RS} \sim \overline{n}$;使 n 号分机进入复位状态;

$\overline{HOLD} \sim \overline{n}$;使 n 号机进入保持状态;

$\overline{OE} \sim \overline{n}$;PC 机请求接管存储器总线信号。当 $\overline{OE} \sim \overline{n}$ 与 TMS32020 的 \overline{HOLDA} 同时有效时,PC 机接管存储器总线。

当在 3COH—3C9H 端口执行输入命令时, $RP_0 \sim RP_9$ 信号有效,各分机的工作状态通过一缓冲器反映到 PC 机的数据总线上。PC 机可通过状态查询了解各分机的工作状态。

为了不使各分机的存储器同时与 PC 机接通造成数据冲突,设计 PC 机放弃控制电路。系统开机时使各分机的命令锁存器输出三态,这样各分机存储器都由 TMS32020 芯片管理,互不干扰。

接口电路在使用时,应先送地址计数器的起始段地址,再送 TMS32020 挂起命令,然后向 TMS32020 处理机中传送程序、数据,或从 TMS32020 处理机中读回数据到 PC 机中。

四、结束语 本文介绍的 PC 机与 TMS32020 接口,是我们开发利用 TMS32020 而设计的专用接口。在我们的双目标智能电视实时跟踪系统中应用,取得了良好效果。加快了设计和开发周期,给我们的硬、软件设计、调试工作带来了很大的方便。通过修改其硬件电路,同样也可以改造为与其它处理器的接口。

参考文献

- [1]张福炎、蒋新儿、李滨宇《微型计算机 IBM PC 的原理与应用》,1984年,南京大学出版社。
- [2]中国科学院声学研究所译《数字信号处理器 TMS32020 用户手册》,1987年。
- [3]内部(SDG—2A 型双目标双模式数字图像跟踪器技术手册),1991年。

单片机录像遥控播放系统的设计

总后医专 陈惠生 侯建新 王德来 白智鹏 王 平

摘要 本文介绍单片机录像遥控播放系统,该系统以 8031 单片机为微处理器,配以 8155、8255、8279 以及串行编码发送及解码芯片 ED5026、ED5027 为主要接口电路,组成录像遥控播放系统,该系统具有双向遥控功能和定时开关机功能。

一、硬件设计 根据系统要求,为了满足系统对遥控功能的要求,系统采用 INTEL 8031 单片机及 8255A、8155、8279 以及大规模数字电路 ED5026、ED5027 等外围接口芯片组成录像遥控播放系统。系统硬件结构见图 1.1,该系统由以下六部分组成:

1.1 8031 单片机系统

这一部分由 INTEL 8031、74LS373 锁存器、E-PROM 2732 及振荡、复位电路组成,8031 单片机的工作频率为 6MHz。

1.2 键盘及按键显示电路

在进行系统电路设计时,为了简化接口电路,缩短研制周期,采用 INTEL 8279 键盘显示芯片。这种芯片同时提供键盘和显示接口,对于扫描方式和其它功能都可通过编程进行选择,能自动进行按键及显示扫描,自动消除按键抖动及重键控制,扫描不占用 8031CPU 的时间,提高了 CPU 的运行速度。键盘由数字键 0~9 和四个功能键组成,按显示由 3 位 LED 显示器组成,前两位显示教室号码,第 3 位显示频道号。采用右端入的输入方式,即计算器式输入,数字从最右边即第 3 位开始输入,逐渐向左移动。

8279 芯片的键盘/显示方式采用内译码方式,因此 SL0~SL2 直接作为 LED0~LED2 的位选线,低电平有效,在同一时刻,SL0~SL3 四条输出线只有一条为低电平,输出波形见图 1.3。

1.3 教室号码与电视频道显示电路

在主机前面板上,每个教室对应着一个 LED 显示器,它显示相应教室里电视机的工作状态,工作时显示该教室里电视机的频道号,关机时不亮,并可切换显示时钟。此部分电路由 8255、LED 驱动电路、位选电路及 LED 显示器组成。8255 芯片的 A 口输出显示频道的段码,C 口输出位选数据,经 7402 或非门运算后驱动位选信号放大器,然后驱动 LED 显示器的位选线。

1.4 中心控制机遥控信号发送与检测电路

该部分电路是主机对各个教室的电视机进行遥控的控制接口,同时也是检测教室控制终端对播放

中心录像机进行控制的检测接口,由并行输入输出接口芯片 8155、编码芯片 ED5026、解码芯片 ED5027、74LS244 以及射随驱动电路、波形整形电路、终端请求控制中断检测电路组成。8031 单片机通过的 8155 的 PA 口,并由两片 74LS244 控制地址信号流向遥控信号编码发送器 ED5026 的 AO~A7 或接收来自教室控制终端录像机控制信号接收器 EED5027 的地址编码线 AO~A7。8155 的 PB 口作为双向口,即可通过 1/2 的 74LS244 向编码器 ED5026 的 D0~D3 写控制码;又可通过另外 1/2 的 74LS244 读终端录像机控制请求信号解码器 ED5027 的控制功能码。达到遥控教室里的电视机和在教室里通过终端控制器遥控播放中心的录像机的目的。

1.5 录像机控制电路

录像机控制电路由 8 片 74LS367、两片 7432、三一八译码器 138、8031 的 P1 口以及 8155 的 PC 口组成,由 8155 的 PC 口发出的三位数据经译码器 138 译码后和录像机有效信号进行与运算后选中所要操作的录像机所对应的 367 芯片,使之导通,将 8031 单片机 P1 口发出的功能控制信号送到相应的录像机上,进行操作。

1.6 电视终端控制器电路

此电路是遥控播放系统的教室控制核心,由信号接收、电源控制、频道切换、录像机遥控编码发送四部分组成,电路图见图 1.2。

1.6.1 信号收部分

由波形整形电路、ED5027 解码电路组成。完成对线路的脉冲信号进行整形解码接收并锁存的任务。

1.6.2 电视机电源控制部分

此部分由三极管 9013、电阻 R1 及固体继电器 SCR1 组成。根据 ED5027 的控制信号线 D3 的电平高低来开关电视机的电源。

1.6.3 电视频道选择部分

此部分由 138 译码器、驱动放大器及继电器组成。根据 138 译码器的信号来切换频道。

1.6.4 录像机遥控编码器扩编码发送器

由5个微型按键、4532编码芯片、ED5027编码器以及其它逻辑电路组成。ED5027根据138译码器的信号确定录像机号的编码,再由录像机控制编码器进行功能编码,然后自动检测外部信号线上是否有信号传输,当外线上无信号时编码发送器ED5026将频道信号和功能信号发送到信号线传输入线上,由系统主机接收并进行相应的控制。

二、软件设计 本系统通过合理的软件设计降低了硬件成本,充分发挥了硬件功效,教室号与频道LED器件采用了动态扫描方式,减少了硬件成本及功耗。根据系统控制的需要,键盘扫描、教室频道显示、电视控制器对放像机的控制均采用中断的方式。键盘中断占用INT0,教室频道显示中断占用CT0,放像机控制中断占用INT1,时钟中断占用CT1,其中设置CT1中断级别最高。系统软件由五个功能模块组成。

系统内所有I/O接口芯片采用与外部数据统一编址,由于外围芯片不多,采用线选法进行编址。根据硬件连接方法,确定各I/O接口的地址,各外围接口的地址见表1。

ED5026 地址编码

教室终端 ED5026 001—120;

中心控制接收录像机反向控制信号的ED5026的地址编码121—128。

2.1 主程序

此模块完成中断入口地址的设置、工作区的初始化以及各个I/O接口芯片的初始化,完成对各个教室中电视机的开闭及频道选择工作;对各个放像机进行操作;检查定时启动和定时关机标志单元,根据标志判断是否进行定时开机或定时关机;设置时钟;设置定时启动时间、定时关机时间。根据系统要求,主程序中设置8031定时器中断优先标志位PT1为1,定时器1中断级别最高。流程图见图2.1。

2.2 键盘中断子程序

此中断子程序较简单,每当按键后,由此子程序把按下的键值从8279芯片的FIFO存储器中读至键值缓冲区20H单元。程序流程图见图2.2。

2.3 教室频道显示中断子程序

该中断利用CT0定时器,在方式0工作,计数器为13位,10ms中断一次保证动态扫描教室频道LED显示器时不产生抖动。并判断是否显示时钟,若显示时钟,则把时钟显示缓冲区的显示码送最后6位LED显示,否则最后6位显示教室频道号。程序流程图见图2.3。

2.4 电视控制器中断子程序

此中断级别最低,当电视控制器上的编码器输出的串行信号经信号检测电路和播放中心控制发送器ED5026的14脚发出的信号进行逻辑与运算,若此时ED5026的14脚的电平为低,则运算后的一个电平下降前沿引起INT1中断,主机则进行相应的中断处理,根据收到的控制码对相应的放机像进行控制。

2.5 时钟子程序

此中断利用CT1定时中断,该中断级别最高,采用方式1工作,计数器为16位,每100ms中断一次,每十次中断进行一次处理,包括向显示缓冲区写显示码,改写时钟计数器,判断定时启动或定时关机时间是否到,时间到则写入时间到标志。

三、信号的传输与抗干扰 由于该系统采用长距离双绞线传送串行数字信号,必须采取可靠的抗干扰措施,保证信号的传输准确无误。

3.1 采用波形整形电路对信号进行整形

本系统的信号是采用双绞线传送串行数字信号,由于线路较长,控制信号传送到终端后波形有些失真,为了保证控制的可靠性,在中心控制机的信号发送端采用射随放大电路,降低信号源的内阻,在电视控制的接收端采用门电路对接收到的信号进行波形整形,由此保证控制可靠,满足系统的要求。

3.2 采用开关式射随电阻

在本系统的信号接收电路中,所有的控制信号接收器都是并接在信号传输线上的,因此,当接收电路的数量增大时,所有终端控制器的录像机反向控制发送电路的射随电阻将变小,即 $R = R_s / N$,式中 R 为并接后电路的总阻值, R_s 为每个发送器的射随电阻, N 为网络中的所接入的电视控制器的个数。这样,会引起信号衰减失真变形,为了保证信号能可靠地传送,我们采取了开关式射随电阻的方式,即在信号射随放在器的发射极电阻电路中串接一开关式工作的三极管,只有控制器工作时开关管才导通,使射随放大器的发射极电阻接通,控制器如没有发送录像机控制信号,开关管不导通,射随放大器的发射电阻呈开路状态,网络中总电阻 $R = R_s / N1$ 。式中 $N1$ 为同一时刻同时进行反向控制录像机的控制器的个数。因此提高了射随放大电路的发射极电阻,降低了线路的损耗。提高了系统的负载能力和抗干扰能力。

3.3 软件设计措施

由于传输线路较长,线路的电感对信号的传输影响较大,为保证各种控制操作准确无误,在软件设计上采取措施消除电感的影响,根据ED5026和ED5027芯片的特性,由中心控制机发向各电视控制终端的控制信号写入8155I/O口后,控制信号发送

装置 ED5026 的发送脚 14 的电平,使其延时 5ms 后再取消此发送信号,保证接收端可靠接收;对于电视控制端发送到中心控制机上的录像机控制信号,中心控制机接到中断信号后再延时 5ms,保证信号准确,系统工作可靠。

四、总结 经过我校电教中心实际使用证明,该系统具有工作稳定,性能可靠、自动化程序高的优点。具有定时播放和定时关机功能,可实现播放中心无人值守,教员可随意对播放中心的录像机进行控制操作,极大地提高了工作效率。

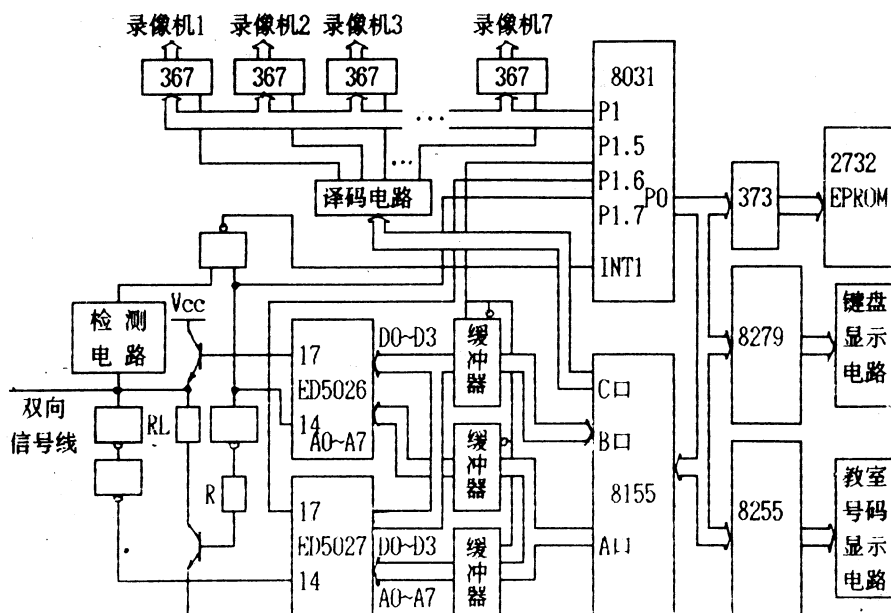


图1.1 系统硬件结构图

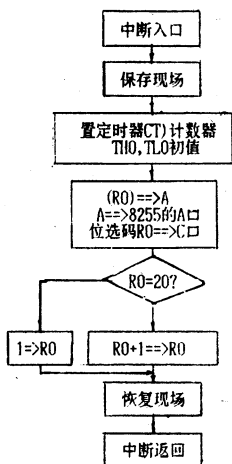


图2.3 教室与频道显示中断子程序流

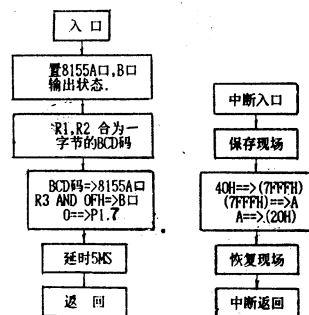
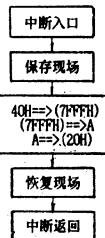


图2.1... 控制子程序流程图 图2.2 键盘中断子程序流程图



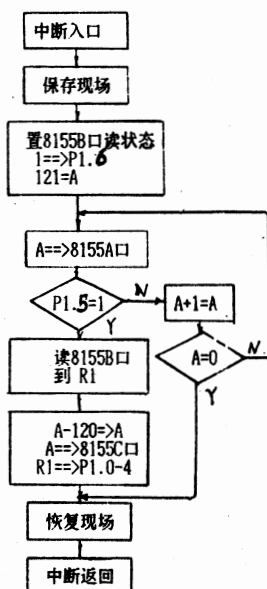


图 2.4 电视终端控制录像机中断子程序流程图

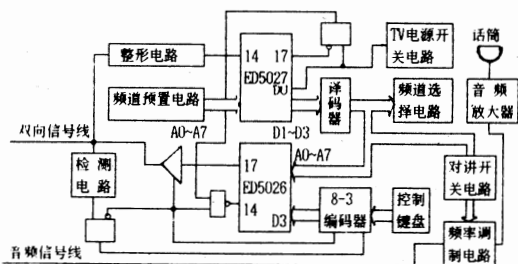


图 1.2 电视控制器结构图

表 1、系统 I/O 接口的各个口地址

| 外围接口名称 | 片内地址单元数(字节) | 地址编码 |
|--------|-------------|--------------------------------------|
| 8279 | 2 | 7FFE _H ~7FFF _H |
| 8155 | I/O | DF00 _H ~DF05 _H |
| | RAM | 9F00 _H ~9FFF _H |
| 8255 | 4 | FF7C _H ~FF7F _H |

参考文献

- [1] 单片机应用系统设计 系统配置与接口技术 何立民编著
- [2] 单片微型计算机原理与应用 陈建泽 兰铮编
- [3] 新编的 ED5026/5027 编译码集成电路
——无线电杂志 1991.2 广宇

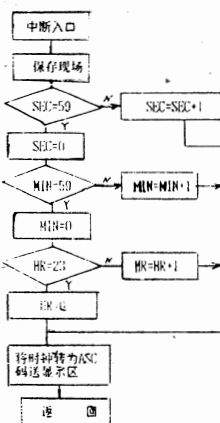


图 2.5 时钟中断子程序流程图

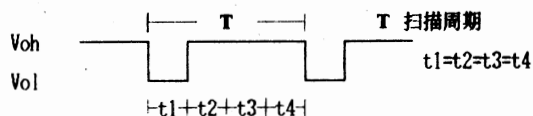


图 1.3 8279位选线SL0-SL3的扫描波形图

条形码技术在生产检测中的应用

中国科学院成都计算机应用研究所 姚景平 李伶

摘要 本文介绍一种将条形码技术应用在生产现场的微机条码检测系统——MCBCDS。文内简要介绍条形码技术及条形码系统的组成,着重论述了所研制的三级分布式微机条码检测系统的体系结构、功能与实现,以及条码技术与计算机相结合的处理方法。

关键词 条形码技术、条形码码制、中间服务站、三级分布式结构、数据正确性与合法性

一、前言 随着国外先进卷烟生产设备的引进,国内大多数卷烟厂的自动化生产水平和管理水平普遍较高,有些厂的计算机应用已经渗透到车间一级。目前,卷烟厂迫切需要一种能实时了解车间卷烟成品包装产量的数据检测系统,为此,我们为用户研制了一套满足这一要求的微机条形码检测系统 MCBCDS。

任何实时系统都必须考虑速度和精度问题,而计算机的数据输入设备和输入方式直接影响到这两个因素。在众多的自动化或半自动化输入设备中,条形码技术与其它相比更成熟,更方便,越来越受到人们的关注,因而我们选择了条形码技术,将它应用到该卷烟厂的生产检测过程中。

二、条形码技术 条形码是由许多预定的不同宽度的深色条形和浅色条形组成的印刷图形,是一种代表数字和字母数字信息的二进制序列。一个条形码系统包括条形码符号体系、媒质、印刷手段、扫描器及译码器五个部分。条形码符号体系是指条形码的码制和结构,目前,已有多种不同的码制,如UPC/EAN、九三码、工业五二码、矩阵五二码、交错五二码等。在实际应用中,是将条形码印刷到某种媒质上,常用的媒质是具有粘性的标签、卡片等。扫描器有一个光敏笔尖,用它划过条形码后,深浅相间的条形产生强弱不同的反射光,转换成相应的二进制电平后,译码器对这个电平序列译码,得到数字信息。将条形码系统与计算机相结合的技术就是条形码技术,条形码技术具有输入出错率低(约为三百万分之一),速度快等显著优点。

早在四十年代后期,这种条形码技术就开始出现在美国及欧洲一些发达国家的各个应用领域,近年来开始在中国出现,它可用于销售业务、铁路运输业务、图书馆管理等等领域中,随着计算机技术的发展,相信条形码技术的应用将愈来愈广泛。

三、微机条码检测系统 MCBCDS

1、系统的背景:

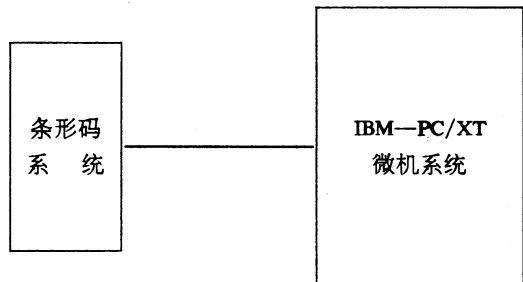
本所为卷烟厂车间已经研制了卷接、包装的自动检测系统,可随时检测到各烟机的各种原材料的消耗及烟支产量,但生产流水线的最后一道工序,即卷烟成箱(在卷烟厂称之为“件”)包装的计量目前采用的却是一种与较高自动化生产水平和管理水平极不相称的人工收牌方法,这种方法既慢且易出差错,车间管理人员不可能随时了解件包装产量的具体数据,微机条码检测系统 MCBCDS 就是为改变这一状况而研制的。

2、系统体系结构的设计与优化:

(1) 设计

我们已经确定了采用条形码技术作为系统的输入手段,系统的目标是使用条形码技术快速、准确地检测各车间卷烟成品产量的实时数据,并能进行数据统计,日、月、季、年数据的存档及报表处理等功能。那么,采用一种什么样的体系结构是合适的呢?

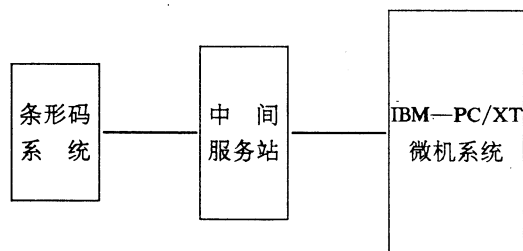
首先,将条形码系统与 IBM-PC/XT 微机系统相结合是必然的。我们的初始模型是采用一种两级分布结构,即直接将条形码系统作为 IBM-PC/XT 微机系统的一个外部设备,所有数据处理,包括数据采集、数据正确性、合法性判定、数据统计、数据文件存档及报表处理等工作均由 IBM-PC/XT 微机承担,该模型插图如下:



采用这种模型的系统能够快速、准确地反映实时数据,系统结构简单,易于实现,但这种模型有一重要缺陷是,它将长时间占用 IBM-PC/XT 的时间,使

其不能用于其它工作,必须为这种模型的系统专门配备一台 IBM—PC/XT 微机系统,这大大增加了系统的开销,显然这种体系结构不宜采用。

为此,我们提出另一种模型,即所谓三级分布式系统结构,其主要原理是增加“中间服务站”作为整个系统的第二级,如图 2:



“中间服务站”首先必须能够快速、准确地检测到实时数据,此外,其成本宜低于 IBM—PC/XT 微机系统,由它承担数据采集、数据正确性、合法性判定、数据统计及实时数据显示等数据处理工作,只在需要时,才使用为车间其它事务处理配备的 IBM—PC/XT 微机,从“中间服务站”收集数据进行累计、存档及报表处理,显然,这种体系结构有相当的优越性。

(2) 优化:

“中间服务站的结构可以从以下三种方案考虑:
采用单片微型机:

大多数单片微型机芯片本身的 I/O 和处理功能都较弱,要构成本系统要求的中间服务站,必须开发一套较完整的 I/O 接口用于远距离和近距离串行数据通信以及实时数据显示,为此,开发工作量较大,比较繁杂。

采用 STD BUS 结构:

STD BUS 是目前相当流行的工业标准总线,它具有高可靠性、实时性、灵活性特点,结构上采用模块化小板结构,实现板级功能分散,若将这种技术用于本系统的中间服务站,则需要配备 CPU 模板、串行通信模板以及必需的终端模板,这样容易在有些模板上造成硬件资源冗余,成本虽低于一台 IBM—PC/XT 微机系统,但资源冗余造成的额外开销是不合理的。

采用价格低廉但用途广泛的 TP801A 单板机:

TP801A 单板机本身的功能基本完整,对它作一定的扩充后可实现系统的各项功能,从系统功能要求、成本及开发工作量三方面考虑,这种结构比较适宜,因而被选用。

3、MCBCDS 的结构:

所研制的 MCBCDS 采取了模型 2 的结构,“中间服务站”在 MCBCDS 中称为“微机条码检测子系统”,即 MCBCDS 由条形码系统、微机条码检测子系

统、IBM—PC/XT 微机三级硬件结构及相关软件组成,三级硬件结构通过专用电缆连接在一起。本系统在实际应用时,条形码系统位于车间内生产流水线的最后一道工序的出口处,微机条码检测子系统和 IBM—PC/XT 微机均在机房内,两处相距约 200 公尺,考虑到车间的噪音干扰因素,采用 RS—485 接口连接条形码系统和微机条码检测子系统,而微机条码检测子系统与 IBM—PC/XT 微机之间直接用 RS—232C 连接,见图 3:

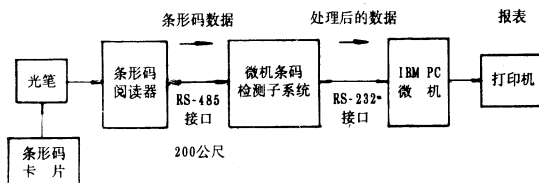


图3. 微机条码检测系统MCBCDS结构图

数据流动方向反映了系统的工作过程:条形码阅读器前端的光笔读入条形码卡片,由条形码阅读器对该条形码进行码型判别和校验,并将其转换成标准的代码,条形码阅读器再将测得的条形码数据代码传送给微机条码检测子系统进行实时数据处理,最后在 IBM—PC/XT 微机需要时将处理后的数据传送给 IBM—PC/XT 微机进行文件存档及报表处理。

(1) MCBCDS 的条形码系统:

MCBCDS 的条形码系统包括智能条码阅读器、光笔、条形码卡片及条形码生成软件。

智能条码阅读器

智能条码阅读器采用进口 PC—WAND300 型可编程条码阅读器,这种条码阅读能阅读多种码制,如 UPC/EAN 码、九三码、交错五二码、矩阵五二码等。它包含有标准 RS—232C 及 RS—485 信通讯接口,可以以点一点方式或多点方式与计算机相连,此外,它还包含一整套完整可靠的通信协议,以便与计算机进行通信,确定通信方式的各种参数均可通过程序设定,而且具有记忆功能,一旦设定就保持直到再改变它。

码制的选择及条形码卡片:

码制直接影响系统的可靠性、成本及处理方法,在选择时,我们从以下三个方面考虑,其一,应用系统的数据类型是数字型,还是字母数字型,本系统为数字型;其二,选择适当信息密度的码制,信息密度

与条形码的印刷成本成正比,最后,根据不同的应用场合选择合适的码制,本系统为工业应用领域。

根据以上三个标准,我们选择交错五二码,这种码制的优点是“编码简便容易,信息密度高,具有通用性,在工业上用得最广,尽管它只能对数字编码,但这对本系统已经足够了。

该系统需编码的信息包含:班次、烟机台号、烟牌、成品包装件号及检验和用专门设计的条形码生成软件生成,并采用激光打印机印刷,最后用表面透明度好、光滑的塑料硬膜制成卡片,这样制作的卡片读出率高,达95%,不会产生误识,见图4:



图4. 生产成品检测条码卡

(2) 微机条码检测子系统:

它是整个系统的核心部分。我们选用了模型2的第三种结构,即利用价格低廉,用途广泛的TP801A单板机,对它进行扩充开发后研制成了能完成以下功能的微机条码检测子系统:

·利用扩充的串行 RS—422A/485 接口接收条形码数据;

·数据正确性判定,重划判定;

·数据统计;

利用扩充的数码显示与控制板进行实时数据显示;

·利用扩充的串行 RS—232C 接口向 IBM PC/XT 微机传送处理后的数据。该子系统的结构图如下:

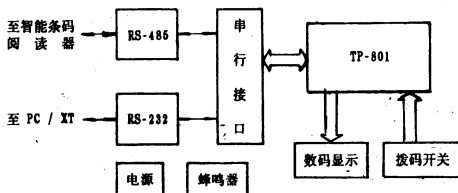


图5. 微机条码检测子系统

4. 应用系统软件设计特点:

(1) 数据结构:

考虑到 TP801A 有限的存贮空间,在系统软件设计时采取不直接存贮条形码件号,而将件号与存贮单元的 bit 位一一对应的数据结构进行原始数据的存贮,某位为“0”时,表明相应的条形码卡片尚未读入,反之。

件号与 bit 位的对应关系可描述如下:

设 p_i 为指向 i^* 烟机号的起始单元的指针, N 为本次读入的 i^* 烟机的一个件号,其合法值为 $1-200$,则代表件号 N 的单元与 P_i 有相对位移为:

$$\text{offset} = \text{FIX}(N/8) \quad , \text{FIX 为取整函数。}$$

N 在地址为 $P_i + \text{offset}$ 单元中相应的 bit 位为:

$$\text{bit} = \text{MOD}(N/8) - 1 \quad , \text{MOD 为}$$

取余函数。

如果 $\text{Dbit} = "0"$, 表示 N 为尚未读入的件号,如果 $\text{Dbit} = "1"$, 表示 N 已读入,本次读入为重划读入。

(2) 两级通信:

由于该系统采取三级分布式硬件结构,因而通信涉及到两个层次,即:

第一层次:条形码系统与微机条码检测子系统之间的通信。

通信双方的协议必须相匹配,所设定的通信参数为:

波特率——2400 baud;

传输数据位数——7 位/字符;

奇偶校验——偶校验;

传送数据结构——以一个完整的条形码(9 个字符)为一个数据块,一个完整的条形码可表示为:

$$C_{m1}C_{m2}C_bC_{s1}C_{s2}C_{s3}C_{s4}C_vC_{end}$$

班次烟机字符 C_{m_i} ($i=1,2$)

如果 $1 \leq C_{m1} * 10 + C_{m2} \leq 32$, 为甲班, 烟机号 $= C_{m1} * 10 + C_{m2}$,

如果 $33 \leq C_{m1} * 10 + C_{m2} \leq 64$, 为乙班, 烟机号 $= C_{m1} * 10 + C_{m2} - 32$, 其中 $0 \leq C_{m1} \leq 9$

烟牌字符 C_b : $1 \leq C_b \leq 3$

件号字符 C_{s_i} : ($i=1,2,3,4$):

$0 \leq \text{件号} = C_{s1} * 1000 + C_{s2} * 100 + C_{s3} * 10 + C_{s4} \leq 200$, 其中 $0 \leq C_{s1} \leq 9$

检验和字符 C_v :

$$0 \leq C_v = \sum_{c=m1,m2,b,s1,s2,s3,s4} C_i - \text{FIX} \left(\frac{\sum C_i}{10} \right) * 10 \leq 9$$

块结束符 C_{end} : $C_{end} = CR$ 。

应答信号——根据一个条形码中各符间的约定关系进行检验,并给出相应应答。

访问方式——查询方式;

第二层次:微机条码检测子系统与 IBM-PC/XT 微机之间的通信:

通信协议规定如下:

波特率——2400baud 或更高;

传送数据位数——8 位/字符;

奇偶校验——无奇偶校验;

停止位/字符——1 个停止位/字符;

访问方式——IBM-PC/XT 微机为查询方式,

微机条码检测子系统为中断方式;

传送数据结构——以块为单位每个数据块包括表示一个机台件号的所有字节及相应的检验和字节;

应答信号——为保证通信正常,提供了一组行之有效的应答信号。

(3)多层次的数据正确性、合法性判定:

·条形阅读器对码制进行了限定,禁止读入交错五二码以外的其它码制,能识别印刷质量差或错误编码的条形码。

·微机条形码检测子系统能识别出非当前班次、当前烟牌的条形码卡片,能识别重划读入的卡片,还能检测出通信线路造成的传输错误。

·当处理后的数据传送到 IBM-PC/XT 微机

后,为该系统配置的系统软件能采用“排除法”识别出读入的作废卡片。

以上为该系统采用的多层次的数据正确性、合法性判定策略,它使系统检测到的数据准确可靠。

(4)实时数据的定时刷新显示及定台显示:MCBCDS 设有实时数据显示机制,可定时按机台顺序循环显示各机产量的实时数据,并可定台显示。

四、结束语

经历几个月的研制调试,MCBCDS 目前已正式在卷烟厂投入运行,效果良好。

条形码技术在生产检测中的成功应用,无疑是使生产检测领域受益非浅,相信不论是工业生产检测,还是铁路运输,图书馆管理等,都将从条形码技术的应用中大获效益。

张炳泉,王晓宇二同志为本项工作给予很大支持和帮助,参加 MCBCDS 研制工作的还有王庆华、余劲松、许承卓等同志,谨在此一并致谢。

五、主要参考文献

[1]江惠民,条形码系统与符号体系

[2]“自动识别的输入设备”《国外自动化》,1985 年第 3 期

[3] Barcodes in the Workplace Putting a system together, Electronic Design, Vol. 30, 1982.

IBM PC—IBM PC 文件通信通用程序设计

河北廊坊炮兵导弹学校 郭继展

计算机网络的重要任务之一就是进行文件通信,而 PC 机(含其兼容机)之间的文件通信又是大量的。本文的“通用”两字是指本程序适用于 PC 机间所有类型的文件:数据文件,如顺序文件、随机文件、WS 文件、数据库文件等,二进制代码文件,如 EXE 文件、COM 文件等,以及任何语言编写的程序。

一、程序设计思想 1. 程序用 BASIC 语言编写,同时具备两种功能:发送文件和接收文件。

2. 一个文件,不管它是什么文件,都把它看作随机文件,每个字符 8bit 位,记录长度取文件长度的小于等于 128 的最大因数(即使是随机文件,也不管它原来的记录长度)。显然,文件长是素数时,记录长为 1。

3. 为了得到文件长度,可先把它当作一个顺序文件打开,测得长度后关闭它。

4. 文件长、记录长相除,可得记录数。发方用紧凑格式(“,”间隔)发送文件,收方用 INPUT \$ 语句

——接收存盘,最后发方发送“EEE”作为结束标志,这实际上实现了检查和校验。

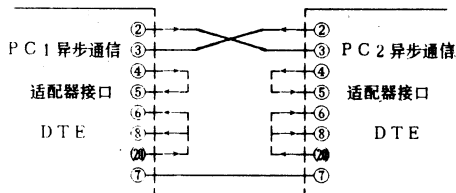
5. 实行“半双工”通信,“握手信号”定为:接收方先运行程序,等发送方传送文件名、文件长、记录长,验证记录数是否为整数后发回“接收准备好”或“参数错误,重传”的信息。收方收到“接收准备好”后开始发送文件。

6. 程序最初设计时,利用通信中断陷阱技术,执行 XON/XOFF 协议,各参数调好后,统统去掉这些语句,以简化程序、加快传输速度(现程序已达 9600bps)。

7. 程序运行万一有误,自动进入出错陷阱,给出已传记录数和错误类型号。

二、硬件连接 每台 PC 机 RS-232C 控制规程异步通信适配器一个,插在大板上空闲槽内。对标准的 DTE 连接方式(NULL 电缆连接,七根线)进行改革,变为无控制信号 MODEM 电缆连接,只需三根

线,省掉四根线。接口控制信号,⑤脚允许发送通过短路④脚请求发送获得,⑧脚数据终端准备好,⑧脚载波检测通过短路②脚数据终端准备好获得,线路如图:



三、程序及运行 程序在 DOS3.0 及 BASICA 下调试成功,试过十几种不同类型的文件几十个,传输无一失误,BASICA 启动时,设置通信缓冲区 8000 字节、文件缓冲区 2000 字节(再少一些也无妨,发送文件的 PC 机也可不设),命令为 "BASICA/C:8000/S:2000"。程序如下:

```
10 CLS;DEFINT A-N;ON ERROR GOTO 320
20 OPEN "COM1:9600,N,8,1RS,CS,DS" AS #1
30 PRINT TAB(24); "IBM PC 间文件通信通用程序";PRINT
40 PRINT TAB(24); "1. 本 PC 向另一 PC 发送文件"
50 PRINT TAB(24); "2. 本 PC 从另一 PC 接收文件"
60 PRINT;PRINT TAB(26); "键入选择号 1/2 ";:INPUT K
70 IF K=2 THEN 210
80 IF K<>1 THEN PRINT TAB(26); "选择错误,重选!";:GOTO 60
90 PRINT TAB(26); "要发送的文件名";:INPUT W$
100 OPEN W$ FOR INPUT AS #2;N=LOF(2);PRINT TAB(26); "文件长";N
```

```
110 FOR L=128 TO 2 STEP -1;Q=N/L
120 IF Q=INT(Q) THEN PRINT TAB(26); "选取记录长";L;CLOSE #2;GOTO 140
130 NEXT L
140 OPEN W$ AS #2 LEN=L;FIELD #2,L AS T$
150 PRINT #1,W$;PRINT #1,STR$(N);PRINT #1,STR$(L)
160 INPUT #1,A$;IF A$(">") "接收准备好" THEN 150
170 PRINT TAB(26); "开始发送文件....."
180 FOR I=1 TO Q
190 GFT #2;PRINT #1,T$;
200 NEXT I;PRINT #1,"EEE";GOTO 310
210 PRINT TAB(26); "等发送方传来文件名、文件长、记录长....."
220 INPUT #1,F1$;INPUT #1,N$;INPUT #1,L$
230 PRINT TAB(35);F1$,N$,L$;N=VAL(N$);L=VAL(L$)
240 Q=N/L;IF Q<>INT(Q) THEN PRINT #1,"参数错,重传";:GOTO 210
250 PRINT TAB(26); "文件改名吗(不改敲回车)";:INPUT F$
260 IF F$="" THEN F$=F1$
270 OPEN F$ AS #2 LEN=L;FIELD #2,L AS T$
280 PRINT #1,"接收准备好";PRINT TAB(26); "接收文件....."
290 FOR I=1 TO Q;LSET T$=INPUT$(L,#1);PUT #2;NEXT
300 INPUT #1,A$;IF A$(">") "EEE" THEN PRINT "传输错误,结束标志不对!";:STOP
310 PRINT TAB(26); "传输顺利结束!";:CLOSE;END
320 PRINT "发生错误! 记录号 I=" I, " 错误编号";:ERR;STOP
```

(上接 8 页)

络管理系统;系统采用了面向对象的设计思想和正确的技术路线,通用性强,易移植,易扩充,用户界面友好,易于为普通用户操作使用;系统采用了先进的 Novell 局网技术,采取了有力的技术措施,加强了数据的保密性和安全性。

旅游业计算机网络管理系统的研制和开发成功,使我国旅游业务工作更加科学化、程序化,对提高各旅行社的运行效率、管理水平、社会效益、经济效益,都将起着较大的促进作用,大大促进了计算机在我国旅游业务管理工作中应用的进一步普及和深化。

(张钢 哲明)

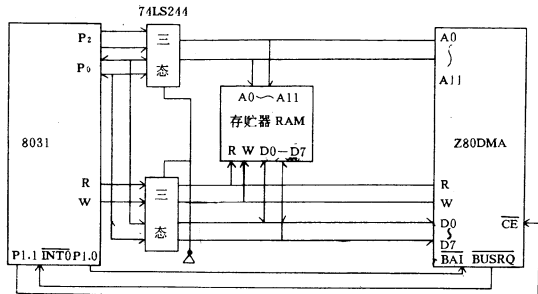
8031 单片机 DMA 功能的扩充

安徽机电学院电气系 周鸣争

摘要 本文介绍了一种将 8031 单片机与 Z80DMA 连接实现单片机 DMA 功能的方法。该方法硬件结构简单,软件编程方便。弥补了 8031 单片机因不具备 DMA 功能的缺陷。文中详尽论述了该方法的硬件工作原理及软件编程要点。

一、问题的提出 众所周知,在单片机与其 I/O 接口及其它计算机之间实现数据通信的方法很多,概括起来主要有两种:并行传送和串行传送,而在并行传送中又有程序查询、中断及 DMA 三种传送方式。其中查询方式和中断方式每传送一次数据,就需要进行 I/O 的输入、输出、修改存贮单元地址、判断 I/O 过程是结束以及保护和恢复现场等过程的处理,因此,在整个传送过程中浪费了大量的时间,这对大批量的数据传送是很不理想的。而 DMA 传送方式,数据传送整个过程完全由硬件设备完成,无需 CPU 干预,因而它是一种通信速率高,信息传输量大的理想通信方式。近年来,8031 单片机由于其功能强,性能/价格比高,在工业过程自动控制及智能仪器仪表中越来越得到广泛的应用,但由于它不具有 DMA 功能,给在一些要求通讯速率较高系统中(如多机系统)的应用带来很大的不便。为此,笔者针对 8031 单片机和 DMA 结构特点,提出了一种对 8031 扩展 DMA 功能的方法,该方法灵活方便,具有较高的性能价格比。

二、硬件结构及工作原理 8031 单片机 DMA 功能扩充如图一所示:



图一 硬件结构原理图

图中,我们采用 Z-80 DMA 作为 DMA 控制器,它可以使用仅传送、仅检索、既传送又检索三类操作,检索和传送最高速率可达 1.25 兆字节,并在传送中产生双地址。在仅传送方式中,首先从一个出入口读出数据,然后再写入另一端口,一个字节接一个字节的进行。按 DMA 功能,这些出入口可以为系统的主存贮器或任一 I/O 高速外设置,这样在 DMA 控制下,数据可从一个外设写到存贮区,或从存贮器

的一个存贮区写到另一个存贮区,或将存贮区中的内容写到外设上。在传送中,DMA 产生双地址(读出地址和写入地址),其地址的寻址方式,可以是固定方式,也可以是从顺序方式,也可以是从一个起始地址起用增 1 的方式(或减 1 的方式)的增量地址。操作的长度(字节数)由字组长度寄存器用编程方式给定,可寻址字组长度可达 64K 字节。传送时产生的两个地址是读周期时用的出入口地址和写周期时用的出入口地址。

Z80DMA 控制器共有四种操作方式:

1、字节方式:每次只传送一个字节,每传送一个字节后返回到 CPU 控制。

2、字组方式:当所选出入口准备就绪,操作持续时间可与 DMA“准备就绪”的有效时间一样长,当 DMA 准备就绪输入为“无效”或“字组结束”或寻找相匹配的字节时,才返回 CPU 控制。

3、连续方式:在返回 CPU 控制之前完成整个检索和(或)整个数据组的传送。

4、周期窃用方式:在本方式中,DMA 操作发生在正常存贮器再生时间内,因为不占用 CPU 操作时间,而只是窃用存贮器再生时间来进行传送,故称周期窃用传送。

DMA 操作管理的顺序是:当 DMA 一旦已编程,就处于“启动”操作状态,当“准备就绪”信号为有效时,DMA 发出总线请求(BUSRQ),CPU 如果响应,就发出一个总线响应信号,这个信号加到 BAI 脚,当 DMA 接收 BAI 信号时,就开始程序所要求的操作。

8031 单片机本身没有 DMA 功能,也就是说它没有 DMA 申请信号(BUSRQ)的输入端,也没有使各总线输出处于高阻状态的 DMA 响应信号的输出,为此,在图一中,我们使用 INT₀ 作为 DMA 申请信号的输入端,使用 P1.0 既作为对 DMA 请求的回答信号,又作为三态缓冲器 74LS244 的控制信号,以使 8031 在收到 DMA 申请信号后,产生中断,在其中断服务子程序中,首先使 P1.0 变为低电平输出,在允许 DMA 控制器使用系统总线的同时,将 8031 的数据线 P₀ 口和地址线 P₀、P_{20~23} 口与存贮器 RAM 断开,呈高阻状态;使 8031 挂起。而在 DMA 传送期间,

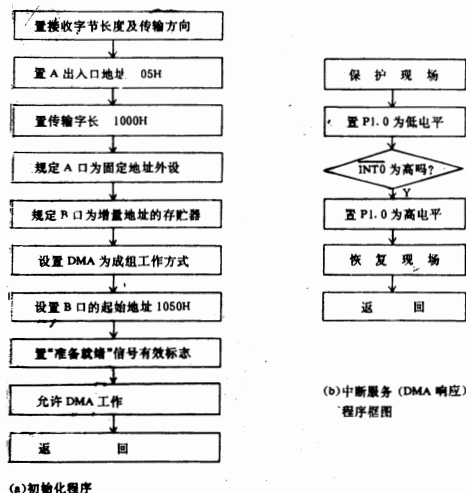
8031 在其服务程序中一直监视 $\overline{\text{BUSRQ}}$ 的电平变化, 当 $\overline{\text{BUSRQ}}$ 由低变高后(即 DMA 传送结束)。立即使 $\text{P}_{1.0}$ 变高, 接通 8031 与存储器之间的总线连接。并返回到主程序的断点处, CPU 继续工作。由于图中采用 INT_0 作为 DMA 申请信号的输入端, 因而使得 DMA 传送具有较强的实时性。同时 CPU 对 DMA 的响应过程均由软件(中断服务子程序)实现, 硬件结构十分简单, 具有一定的灵活性及较高的性价比。

三、控制软件设计 由 Z80DMA 控制器的工作原理可知, 要实现 DMA 传送, 首先必须要对其进行初始化, 完成对其有关参数的设定。Z80DMA 有两个基本状态即 DMA 有效状态和 DMA 无效状态。在允许状态下, DMA 可以获得对系统总线的控制权, 并且可直接进行两个口之间的数据传送。在禁止状态下, DMA 不能请求总线控制权也不能传送数据或启动中断, 当 DMA 加电或以任何方式复位时, DMA 就自动处于禁止状态。无论 DMA 处于什么状态, CPU 均可给 DMA 控制写命令, 但对其进行写命令后仍将自动使 DMA 处于禁止状态中, 一直要等到 CPU 发生一条允许命令为止。因此, 对 DMA 初始化就是根据应用的具体要求, 去选择其某一种功能和工作方式。在实现对 DMA 初始化编程中, 要使用到内部 1A、1B、2A、2B、2C、2D 六个基本命令字^[1], 这六个命令字的结构已决定了各控制/状态字使用时的前后次序, 笔者在 8031 单片机某一应用系统中, 利用图一的硬件结构, 把参数采集部件当源口(A 出入口), 存储器当目标口(B 出入口), 进行 DMA 数据传送。且要求:

- 1、数据信息由采集部件传送到 DMA 中。
- 2、字组长度设为 1000H
- 3、存储器从 1050H 地址单元起存放采集数据。
- 4、采集部件的地址为 05H。
- 5、来自采集部件的“准备就绪”信号为低电平有效。

按照上述的要求, 对 DMA 初始化及 8031 响应 DMA 的控制程序框图如图 2 所示。

在执行初始化程序时, 应先使 $\text{P}_{1.1}$ 端口输出为低电平, 这是由于在图一中, $\text{P}_{1.1}$ 端口作为 DMA 控制器的片选信号 $\overline{\text{CE}}$ 的控制信号。



图二 控制软件流程图

四、结束语 综上所述, 利用 8031 单片机 INT_0 和软件控制程序实现 DMA 申请输入及响应输出方法, 具有硬件连接简单, 软件编程方便的特点。笔者在多个单片机应用系统中应用表明, 该方法工作可靠, 实现方便, 有效地解决了 8031 单片机与其它外设及计算机之间的 DMA 传送问题。该方法只要稍加改动亦可完成 8031 与其它类型的 DMA 控制器(如 INTEL 8257 等)的连接。为 8031 单片机 DMA 传送方式的实现提出了一种实用的方法。

参考文献

- [1] 周明德:《微机原理及应用》, 清华大学出版社, 1982
- [2] 陈章龙等:《实用单片机大全》, 黑龙江出版社, 1989

(上接 63 页)厂, 从设计到批量生产一般要三个月周期, 而包括方案选择在内的设计阶段就要花一个月时间。设计阶段花费时间太长, 是造成上述困难的一个重要原因。随着人们消费观念的改变, 款式翻新成了主要矛盾。本系统的扩大应用为解决这一瓶颈问题注入了科技动力。同时我们也清醒地看到, 面对我国因制鞋业对科技进步的更高要求, 比照国外同类系统的差距, 我们还有许多研究与开发工作要做。但

是 SMD-6410 在众多竞争者中获得的成功, 坚定了我们争取更大进步的信心。只要我们坚持瞄准国际先进水平和技术, 紧密结合中国国情, 结合生产实际, 我们的系统会更加成熟、更加完善, 为推动传统制鞋业步入现代化的行列作出应有贡献。

在本系统的研制和开发过程中, 我们得到不少同志的关心和支持, 特别是和张执谦所长的关心和支持分不开的, 借此深表感谢。

IBM——PC/XT 中断扩展的具体实现方法

中国保险管理干部学院计算机系 赖红威

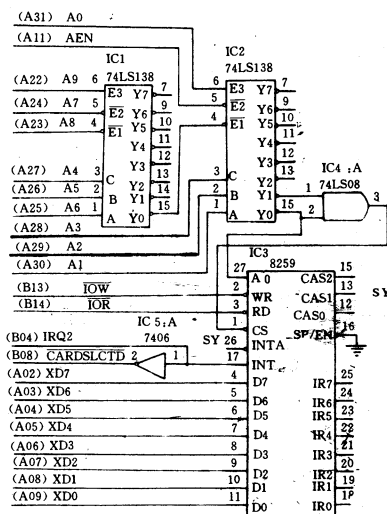
摘要 本文就 IBM——PC/XT 中断扩展的实现,给出详细的电路图与程序清单,只要您按图索骥,即可实现中断的扩展。

一、问题的提出 现在利用 IBM——PC/XT 计算机实现实时控制的地方越来越多,而在实时控制中又常常使用中断技术。虽然机器本身配有 8 级硬中断,但基本外设已占去多数,留给用户的已所剩无几。因此,用户在设计自己的控制系统中往往觉得硬中断不够用。为解决这一问题,笔者设计了一个扩展了 8 级硬中断的简单电路。如能掌握该电路,在此基础上很容易实现更多级的中断扩展。该电路在实际使用中效果甚佳。

二、硬件的设计 在 IBM——PC/XT 中,CPU 所接收的硬中断请求是由中断控制器 8259 A 提供的(关于 8259 A 的详细说明请参阅有关书籍)。要想实现中断的扩展,也就是要使主(系统板上)8259 A 与从(扩展板上)8259 A 实现级联。而在 8259 A 的级联中,必须要保证的硬件条件是:①各 8259 A 要有自己的独立地址;②级联线 CA S0—CA S2 要并联;③INTA 线要相互联接;④必须确保各 8259 A 所发的中断向量能被 CPU 可靠接收。图一给出的是满足上述条件的一个扩展了 8 级中断的电路图。在该图中作为从 8259 A 被分配的地址是 I/O 地址 20 1H 和 20 3H。由于 I/O 插槽上没有 CA S0—CA S2 和 INTA 的信号线,所以在实际应用中要将扩展板上的这四根线用导线和系统板上的 8259 A 相对应的各线相互联接起来。下面的问题是如何使 CPU 能够在它的第二个 INTA 周期内读到从 8259 A 发出的中断向量。由于这里的从 8259 A 是采用非缓冲方式,所以它应与系统板上的主 8259 A 直接共数据总线。然而,作为系统板上的主 8259 A,它的数据总线又是和 I/O 插槽 J8 直接相联的。因此,只要将中断扩展板插在 J8 上就可实现主、从 8259 A 直接共数据总线。这里要注意的是 J8 上 CA RD SL CT D 信号,当它为高电位时,数据是由 CPU 输出到扩展板上的,低电位时数据则是由扩展板向 CPU 输入的。因此,在中断响应期间 CA RD SL CT D 应为低电位,从 8259 A 的中断向量才能被 CPU 正确接收。为此,应将从 8259 A 的 INT 取反后与 CA RD SL CT D 相联。

三、软件设计 程序如下。由于该程序已作了详细的注解,这里就不再赘述。只是要注意的是,为

了在调试中不出现死机现象,在返回 DOS 前要将主 82 59 A 的预置命令字、(IC W)恢复到初始状态。



图一

STACK SEGMENT PARA STACK
DB 32 DUP(?)

STACK ENDS

DATA SEGMENT PARA 'DATA'
INTA00 EQU 20H
INTA01 EQU 21H
INTA10 EQU 201H
INTA11 EQU 203H

DATA ENDS

CODE SEGMENT PARA PUBLIC 'CODE'

START PROC FAR

ASSUME CS, CODE, DS, DATA, SS, STACK

PUSH DS

SUB AX, AX

PUSH AX

MOV AX, DATA

MOV DS, AX

; 设置中断号

PUSH DS

MOV AX, CS

微型机的故障维修技术

西北工业大学 牛海发 张选阁 张双才

摘要 本文从理论和实践上讨论了PC系列微型机的故障维修技术方法,每种方法均列举了实例。

微型机系统硬件出现故障,维修的关键在于查找故障源。而故障源查找主要是根据零部件、插接件和元器件的作用,产生故障后的影响,及可能产生的故障类型;通过测量、分析、比较、再进行判断,进而来确定故障的部位。实际维修中,都是采取一些简单而又实用的维修方法。较复杂的故障,采用多种方法结合,逐步查找维修,直到把故障排除掉为止。下边就以笔者所用的一些实用技术方法介绍如下:

1、敲击与手压法

实际中,经常会遇到微机运行时好时坏的故障现象,这是由于组装或生产厂家的产品质量原因。这种故障大多数是由于接触不良或虚焊造成。排除这类故障可先采用敲击或手压法。所谓敲击法就是:对可能产生故障的部件,用小橡皮锤之类敲击物轻轻敲打插件板或部件,看是否会引起出错或停机故障。所谓手压法就是:故障出现后,关闭电源对插接的部件或插头插座或芯片用手压牢。再重新开机试验看故障是否排除。这种方法对新买的微机故障排除非常有效。

例1、一台 SUPER AT 微机买来不久工作就不正常,采用小橡皮锤轻敲外壳,内存出错;关闭电源把内存芯片用手压一遍,再开机试,机器工作正常。由此得知是主板上内存芯片接触不良所引起的故障。

2、插拔法

插拔法是:通过插拔机内的一些插件板或芯片来判断故障原因和部位的一种方法。微机有些故障出现后,可很快判断出在那块插件上;而有些故障不易一下判断,因各插件板故障都可能引起该故障现象。在检修中为了找出故障所在,采用逐个拔掉各扩展槽上的适配器卡,每拔一块后,开机试一,看机器是否恢复正常。当拔到那块后,机器恢复正常,就证明该适配器或器件有故障。

例2、有一台 IBMPC XT 微机,开机完全无反应,检查+5V 电源有问题,但是电源部件都正常,把扩展槽所有插板拔掉,机器恢复正常运行。断定故障在适配器上,然后插一块试一次,终于找到了一块适配器因+5V 电源因短路引起的故障,短路排除后机器

便恢复正常。

3、替换法

所谓替换法就是:用一台好的设备替换一台可疑设备,看故障是否消除;或把一个可能故障的元组件,接到一台正常机器上,看是否出现相同故障的方法。这种方法适用于判断系统故障的部位、而特别适用于判断能插拔芯片的好坏。

例3、一台 SUPER XT 微机的 A 驱动器工作不正常;不知软盘驱动器故障,还是驱动器读写电路故障。用替换法把 B 驱改为 A 驱后,结果运行正常。断定故障在原 A 驱上,进一步检修,即可排除故障。

例4、ALPHA 分时系统机一台 AM-50 终端,开机后无任何反应。拔掉 8031 单片机芯片,插入一片好的 8031 芯片,再开机该终端工作恢复正常;故障是原 8031 单片机损坏造成。

4、比较法

比较法是:用正确的特性与错误的特性进行比较,以便把故障确定到具体元器件的方法。按比较性质分为:

(1)电压比较:如电源的电压,键盘、显示器等的信号线,在正常工作下值一定,可实测数值与正常数值比较。

(2)波形比较:对那些有规律性的信号(如时钟、复位等),做是否与正常信号相同的波形比较。

(3)静态阻抗比较:关机后,用万用表电阻档测量有关电阻与正常值比较。

(4)输出结果比较:如按键偶尔断续产生错误,把出错码与正确码比较;打印机打印字符出错部分与正确部分比较。

通过上述比较方法,就可以找出出错的规律,进而确定故障的原因再加以修复。

例5、有台打印机自检检测正常、串行打印亦正常,但并行口连机时打印字符出错。如打印 D59E5 却变成 @19A1,即 E 变成 A 字符了,逐位比较它们的 ASCII 码,是因二进制的 D2 位 1 信息变成 0 信息的缘故。其故障范围在多路锁存器或先进先出的缓冲存储器部分,再用万用表测量这部分 D2 脚对地的电阻,其值为零(正常应为千欧级),换去损坏的组

件,测量电阻值恢复正常,开机打印也正常,故障即排除。

5、直接观察法

直接观察法是:利用人的眼、耳、手、鼻检查机器是否有火花;异常音响;过热烧焦现象;或电源短路、过流、过压和保险丝熔断等现象;观察有关插接件是否松动——引起接触不良,或虚焊、脱焊、断线、短路、元器件锈蚀、损坏等故障。

例 6、一台 PC 286 微机键盘按键后发送信息出错,采用直接观察法发现几个键已接触不良,经维修后恢复正常。

例 7、一台 M-1724 打印机连机打印时出现: NO PAPER(实际有纸)故障。拆开打印机直接观察,发现开关即将断线而造成接触不良。排除故障方法是:先用酒精棉球擦洗后,再重新将所断线焊接好。之后开机连机打印正常,故障即排除。

6、直接测量法

人工诊断微机故障时,为尽快查找到故障源并缩小故障范围,常采用各种测量工具进行一些元器件特性的测量,如用万用表静态测量元件的各点电平或其它参数,以便分析判断故障原因的方法。测量法能方便的检查出 CPU、接口和存储器出现的故障。

例 8、一台 SUPER XT 微机开机后主机或外设的保险马上烧断。经检查是电源板上有短路,开机后烧断保险丝只可能有两种情况引起。采用直接静态测量法检查,测量了有关的电源脚对地线的电阻,器件 1488 的 14 脚(+12V)和 1 脚(-12V)的静态电阻仅在欧姆级,与正常值不同,说明该组件损坏。更换后再测量两脚间电阻值恢复正常。开机后亦工作正常,故障排除。

7、诊断程序自检法

IBM PC XT/286/386 及其兼容机的 BIOS 中,有一个自检诊断程序 POST,利用它开机后对系统本身各组件能进行逐一测试,按不同类型故障或是显示出错代码或是发出响声警告,便于以此确定故障原因。这种方法对于主机故障,特别是内存故障能给予某芯片的确切位置。

例 9、有一台 IBMPC XT 微机,开机自检到 RAM 时屏幕上出现:

192KB OK

30000 04 201 字样信息,仍旧往下继续执行。这种内存没有检测完就提示出的上述故障信息,是由

于 RAM 芯片损坏引起的。一般都可由显示的出错信息确定出损坏芯片部位。其 192KB OK 信息说明在检测中,前 192KB 内存是好的;而 30000 04 201 是真正的错误代码,第一位是三,说明故障出在系统板的 BANK 3 组;第 1—5 位表示 30000 单元有错;第 6—7 位为 04 表示第三位 RAM 芯片损坏。第 8—10 位为 201 表示内存出错。故障部件为系统板上 BANK 3(第四排)的 D2 位(第四列)RAM 芯片,更换后,开机自检通过,故障排除,在排除这类故障时应该注意:因 RAM 芯片可拔插,所以应先看看 RAM 芯片上否接触好等,若没问题,再按错误信息找出故障的 RAM 芯片,更换一片好的即可。

8、高级诊断法

使用 IBM 公司的高级诊断程序(一般在软盘上保存),测试每个机器部件(A 软驱正常下),确定机器元器件好坏的方法,称为高级诊断法。

例 10、有一台 M-1724 打印机打印不正常,或无法启动。用高级诊断程序对打印机进行测试,结果显示:9XX 错误代码,然后拔掉打印机信号电缆,再加电启动高级诊断程序对打印机测试。屏幕显示 90X 错误信息,或 INSERT WRAP PLUG AND PRESS "ENTER" 信息,都证明故障在打印机适配器,否则打印机故障。再结合其它方法检修,就可排除故障。

9、综合检修法

微机系统有时出现的故障较复杂,采用某一种方法不能直接找出故障原因,这时就要用多种方法检修。在检修微机故障时,采用多种方法检修,叫做综合检修法。

例 11、一台 SUPER AT 微机,在 DOS 操作系统支持下运行用户程序,突然程序飞了。产生这种故障的原因很复杂,用多种方法综合检查,如先用拔插法去试探故障原因在哪个范围;再把可疑部件用“交换法”检查,若程序飞不了,故障部件找到;若程序还飞,则故障不在插件板的元器件上,很可能是虚焊或其它接触不良,就可用“敲击与手压法”检修故障;若还找不到故障原因,就可能是时序配合不当、或用户程序对硬件环境要求不合理;也可能是软件的稳定性比较差;否则可用测量法逐点检查可疑部分,就可找到故障点进行检修,使机器恢复正常运行。

专家系统检修法是微机微修站普遍使用的一种方法,许多报刊和资料上都有广告或介绍,在此不述,有兴趣者请参阅有关资料和文章。

一种方便修复故障盘数据的实用程序

总后基地指挥部自动化站 阮高华

零磁道逻辑故障是发生在旧软盘中的一种常见故障。当用户对其进行读写操作时,屏幕均显示“扇区未找到,A驱动器读错误”信息,使用户束手无策。对此故障,笔者曾在《计算机应用研究》89年第5期上介绍了采用动态调试程序DEBUG进行修复的方法步骤。这种方法的优点是简便易行,不受软、硬件环境的限制。缺点是此方法要直接涉及到磁盘的目录地址、驱动器号、扇区号等参数,操作稍有不慎,则后果不堪设想。因此,一些不太熟悉DEBUG及DOS内部结构的用户希望能使用简单的键选菜单式方法,修复故障盘数据,为此,笔者采用汇编语言编写了一个数据修复程序,可以满足不同用户所需。

我们知道,磁盘文件的完整结构是由文件的目录区和数据区两部分组成的,因此,程序的功能设计主要分为读写目录和读写文件两方面。主要设计步骤如下:

1、显示功能菜单,接收用户的键选输入。

2、选择读写目录功能时,提示用户将故障盘插入A驱动器,非故障盘插入B驱动器,准备完毕后任按一键。

3、用中断INT25H将故障盘中的目录扇区读入内存。

4、用中断INT26H将内存中的目录扇区内容写入非故障盘中。

至此,故障盘中的文件目录扇区内容即修复至非故障盘中,可用DIR命令显示查看。当然此时只有文件名,而无实际内容。

5、选择读写文件功能时,提示输入文件名。

6、用中断INT21H,0AH读取待修复的文件名。

7、用中断INT21H,42H读取待修复的文件长度。

8、用中断INT21H,3FH将待修复的文件内容读入内存。

9、用中断INT21H,0AH,42H,40H将文件名、文件长度和文件内容写入非故障盘中。

至此,即完整地修复了一个文件,接着可进行下一个文件的修复操作。

本程序采取汉字菜单驱动,提示信息帮助,操作简单,使用方便。此外程序还具有较好的容错功能,即使出现误操作也不会导致不良后果。该程序已在

PC/XT、长城-286EX等微机上通过,数据修复效果明显。现将程序清单介绍给大家,只要将其编译、连接成EXE文件,即可执行之。

程序清单如下:

```
stack    segment para stack 'stack'
stapn    db 256 dup(?)
top       equ length stapn
stack     ends
data     segment para public 'data'
mbuff     db 49
          db ?
          db 50 dup(?)
messl     db 30 dup(' '), '数据修复程序', 38 dup(' ')
          db 30 dup(' '), '1——读写目录', 38 dup(' ')
          db 30 dup(' '), '2——读写文件', 38 dup(' ')
          db 24 dup(' '), '请输入 1/2 (q 键退出):', '$'
dxslp     db 15 dup(' '), '读文件,请输入盘符、文件名:', '$'
dxslp1    db 15 dup(' '), '写文件,请输入盘符、文件名:', '$'
soft1     db 15 dup(' '), '请插入源盘至 A, 目的盘至 B, 任敲一键继续!', '$'
meseg     db 15 dup(' '), '文件名错! 请敲一键重新输入!', '$'
handle     dw ?
datbuff1  db 60416 dup(?)
datbuff    db 3584 dup(?)
data      ends
mcode     segment
mproc     proc far
          assume cs, mcode, ds, data, es, data
clear      macro
          mov ax, 7
          int 10h
endm
```

```

mcal macro mes,numb,dxfs,wjm,wjc
    mov dx,offset mes
    mov ah,numb
    mov al,dxfs
    mov bx,wjm
    mov cx,wjc
    int 21h
endm
cpdx macro qdqh,sqh,sqs,dxcz
    mov al,qdqh
    mov dx,sqh
    mov bx,offset datbuff
    mov cx,sqs
    int dxcz
endm
qwjm macro
    mov dx,offset mbuff
    mov ah,0ah
    int 21h
    mov bl,mbuff+1
    mov bh,0
    mov [mbuff+bx+2],0
endm
qwjc macro
    mov ah,42h
    mov al,2
    mov bx,handle
    mov cx,0
    mov dx,0
    int 21h
endm
start: push ds
        xor ax,ax
        push ax
        mov ax,data
        mov ds,ax
        mov es,ax
lop:    clear
        mcal mess1,9
        mcal 0,1
        cmp al,'q'
        jz exit
        cmp al,'1'
        jz rmla
        jmp open
exit:   mcal 0,4ch

rmla:   clear
        mcal soft1,9
        mcal 0,8
        cpdx 0,5,7,25h
        add sp,2
wmlb:   cpdx 1,5,7,26h
        add sp,2
        jmp lop
error:   clear
        mcal meseg,9
        mcal 0,8
open:    clear
        mcal dxslp,9
        qwjm
        mbuff+2,3dh,2
        mov handle,ax
        qwjc
        mov cx,ax
        push cx
        mcal mbuff+2,3dh,0
        jc error
        mov handle,ax
        pop,cx
read:    mov ah,3fh
        mov bx,handle
        mov dx,offset datbuff1
        int 21h
        push cx
write:   clear
        mcal dxslp1,9
        qwjm
        mcal mbuff+2,3ch,,,0
        jc error1
        mov handle,ax
        qwjc
        pop cx
        mov ah,40h
        mov bx,handle
        mov dx,offset datbuff1
        int 21h
close:   macl 0,3eh,,handle
        jmp lop
error1:  clear
        mcal meseg,9
        mcal 0,8

```



```

      jmp write
mproc      endp

```

```

mcode      ends
end start

```

IBM-PC/XT 硬盘不能自举维修一例

河北水利专科学校计算机教研室 李洪涛

故障现象 IBM-PC/XT 硬盘不能自举,可以由 A>转入 C>,但 C 盘只能读不能写,如 A 驱动器引导,自检完内存后机器“嘟……”一声长鸣,A 驱动器指示灯不熄,Ctrl-Alt-Del 失效。

一、分析与维修 根据故障现象,应从以下几部分查检:

1. 硬盘驱动器逻辑板的写入部分。
2. 软硬盘适配器及有关插槽。
3. 电源输出及其它。

首先因硬盘不能写入,在 A>下用 COPY 命令不断向 C 盘复制文件,用 50MHz 示波器查出驱动逻辑板的 U₁₈有问题,14 脚有波形而 15 脚无波形,可能 U₁₈坏。U₁₈为 CA3096E 是只封装 5 只三极管元件,其中 2 只为 PNP 型,3 只为 NPN 型。

更换 U₁₈,如无可此芯片可用国产 3CG₃ 和 3DG₇ 代用,焊下 U₁₈,装上一只 18 脚插座,将 3DG₇ 和 3CG₃ 配对按下图插入,经试效果良好。

其次,因硬盘不能自举,为保障原记录不被破坏,可用生成 C 盘的系统盘插入 A 驱动器,启动,在 A>下键入 SYS C;回车,显示器显示:system transferred,再 copy command com c:↙,热启动,硬盘仍不自举。

在 A 驱动器插入 DOS 系统盘,列目录查看有无 LowFoRm. EXE;FDISK. com;FoRmat com。

A> LOWFORM. EXE↙

A> FDISK. COM↙

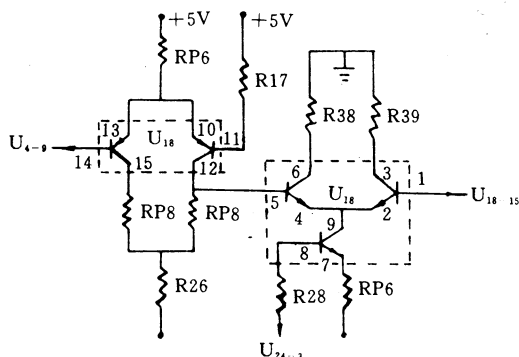
A> 1↙

A> Y↙显示器显示

| Partition | Status | Type | Start | End | Size |
|-----------|--------|-------|-------|-----|-----------|
| 1 | A | DOS | 0 | 304 | 305 |
| Total | disk | space | is | 305 | cylinders |

A> FORMAT C:1S↙

重新启动机器故障消除,(注意硬盘原有记录已被破坏)用 COPY 命令考入全部硬盘备份文件。



PC/XT 软盘驱动器故障二例

福州大学自动化所 周力

例一故障现象 开机后,不能引导 DOS 系统,不能正确地读/写,清洗磁头后,虽能读能引导 DOS 系统,但写入操作时有出错。执行 FORMAT 命令重新格式化后,该盘能正常执行读/写命令,并能引导 DOS 系统,但盘的有效存储空间只有 179712BYTES,仅为正常值的一半。

故障分析 由于经该驱动器格式化后的软盘能正常引导和正常读写,故可断定主轴驱动、磁头定位、“00”磁道开关、索引电路、读放电路、写/抹控制

均正常。故障范围可缩小到磁头选择电路、以及“1”磁头与“写”驱动之间的有关电路。顺便说明,由于能执行 FORMAT,因此可以肯定“0”磁头以及与“0”磁头有关的外围线路是好的。现以 JA551 软盘驱动器的有关电路为例进行分析见附图(其它型号的电路也大致相同)。图中红、白、黑、绿分别是磁头读/写线圈和抹线圈的引出线的颜色,RP6 是电阻排,HA16642 是大规模集成片,其功能包括读放、微分、限幅、临零整形、读/写控制等。DP2、DP3、DP4、DP5、

制鞋 CAD 开发应用刍议——兼谈 SMD—6410 技术特色

四川省电子计算机应用研究中心 CAD 室

马在强 徐一清 张成武 刘 营

摘要 本文详尽介绍了国内外制鞋 CAD 开发应用的现状和自己开发的 SMD—6410 系统的技术性能及其特色。

关键词 制鞋 CAD 开发应用

一、国内外开发应用概况

从七十年代末至今,国际上制鞋 CAD 开发研制工作经历三个阶段。第一阶段是二维设计系统。经手工制作好的平面样片,数字化后输入计算机,由系统解决几何变换、放码及排料等问题,而后经绘图设备绘出加工样片,第三阶段是二、三维混合设计系统。除上述功能外,增加了三维楦面数据采集功能,解决了楦面造型、楦面展平、平面设计、三维设计等问题。第三阶段是制鞋 CAD 与 CAM 广泛结合,利用 NC 技术开发一系列 NC 机械,作为 CAM 硬件支持。同时增加诸如设计数据、成本计算、原辅材料管理、定单管理等一系列管理信息,在统一的数据库基础上,将 CAD、CAM 及管理信息集成在一起形成完整的集成系统。这些开发工作极大推动了古老制鞋工业步入现代化的进程。

这当中比较典型的系统有如下几家。

(1) 英国 BUSM 公司开发的 CRISPIN 系统。该系统采用工程工作站做主机,并带一台微机控制的下料机,样片从数字化仪输入,可以做放码、排料直至切割样片工作,对我国报价 20 万美元一套。

(2) 加拿大 CIMTECH 公司推出,后为美国 MICRODYNMICS 公司收购的 DESCOM 系统。该系统利用三维楦面测量仪,将楦面数字化为 2 万多个点输入计算机,在此基础上,作楦面展平、平面设计,设计好的平面款式再返回三维状态显示样鞋线框图。同时开发了 NC 下料机,NC 缝纫机,NC 鞋底模具制造加工机械,DESCOM 系统是目前国际上功能比较齐全、实用性较高的系统,在加拿大本土,以及美国、法国及英国都有销售分公司。该系统(NC 机械中只包括立体数字化仪及下料机)对我国厂家报价为 40 万美元一套。

(3) 美国 CAMSCO 公司的 APEX 系统,是一个立体帮样设计系统。在楦面上画好款式线,经楦面测量仪数字化后送入计算机,并将其展平、放码、做用料计算等工作。

(4) 美国 GERBER 公司的 ShoeMaker 系统,是运行在 SGI 工作站上的一套专用制鞋 CAD/CAM/CIM,满足所有二维及三维制鞋要求,是目前依附于工程工作站销售的主要制鞋 CAD/CAM 产品之一。此外,ShoeMaker 还在数据库基础上,建立了 CIM 集成系统。

除此而外,法国、西德、日本、意大利及我国台湾省都不同程度地研制和开发了一系列制鞋 CAD 应用系统和制鞋业信息管理系统。在日本甚至搞成一个远程的制鞋 CAD/CAM 网络系统。这一开发工作,几乎涉及到所有工业发达的国家和地区,呈现一个利用现代计算机技术改造武装传统制鞋业的世界性潮流。

在研制开发的同时,应用也呈现一派生机。据不完全统计,在八八年下半年,光欧洲几个发达国家采用 CAD/CAM 技术和设备的制鞋厂家和研究单位已达九十多家,而后几年应用部门正按每年翻番的速度增长。我国台湾省面对该省制鞋厂家纷纷向海外投资辟厂的趋势,也广为采用制鞋 CAD/CAM 技术拯救在台的制鞋工业。目前来我国做生意的工程工作站商家在推销软件上产品中,往往也包括制鞋 CAD/CAM 系统。

如前所述,国外制鞋 CAD/CAM 系统一般非常昂贵,非国内厂家所能承受。在世界开发和应用潮流冲击下,我国计算机工作者和制鞋研究人员以及厂家纷纷转向自己的开发研制工作。86 年下半年,上海大学工学院与福州皮鞋工业研究所开发了“PFS—1 微机辅助皮鞋优化系统”,并通过技术鉴定,开创国内二维制鞋 CAD 的先例。继后,围绕轻工部制鞋研究所六十年代提出的三角控制楦面轮廓展平方法,湖南大学,北京皮鞋工业研究所等众多单位都编制了展平图的绘制程序。成都科技大学、长春光机所、轻工部制鞋研究所、航空航天部 303 所,重庆大学等不少单位,在微机或工程工作站上也轰轰烈烈地展开了独立或合作的开发工作。其中影响较大的

是轻工部制鞋研究所和航空航天部 303 所合作开发的“CAD/CAM 在鞋植、鞋帮设计加工中的应用”。但是,由于一方面设备投资过大,一般鞋厂根本无法问津,另一方面一些带根本性的与生产紧密相关的关键技术配套问题尚未解决,因而不能直接推广应用。可以说,自 86 年以来,在国内也形成一个开发热潮。但国内应用情况却不太理想。由于经济原因,国内无一家鞋厂引进国外制鞋 CAD 产品,在 SMD-6410 投入市场以前,没有开发出实用化的系统。

SMD-6410 是四川省计算中心研制开发的三维制鞋辅助测量和辅助设计系统。该系统历经三年的研制开发工作后,又经一年多的应用完善提高。在国内众多的同类开发项目中以其功能齐全、实用性强、辅助设计效率高、性价比优异、技术先进、适合国情而率先开始进入国内市场、受到越来越多的制鞋厂家的青睐,并荣获四川省九零年科技进步二等奖。

二、SMD-6410 的主要功能及技术特色

在广泛调研和查阅资料的基础上,SMD-6410 制定了瞄准国际八十年代先进水平,从解决植面三维数据采集入手,开发二三维设计系统的第一阶段目标。在植面数据采集、植面展平、平面样片设计、三维样片设计、成鞋造型、样片放码、用料计算、优先排料的全部设计过程均采用 CAD 技术,达到功能配套、实用、技术先进、系统性价比高的设计要求。完成的 SMD-6410 完全实现了第一阶段目标,并为第二阶段的 CAM 开发奠定了良好的基础。

为了达到设计目标,SMD-6410 开发了下列支撑环境和软件:

- 与大专院校合作,完成了实用廉价的三维植面测量仪研制

- 符合 CORE 标准的三维图形库
- 与图形环境兼容的专业汉字信息显示与管理软件

- 全植面及特殊边界测量及设计用特征数据生成软件

- 全植面及三维款式展平软件
- 采用空间画笔的多层三维款式设计软件
- 多种输入设备下,多层平面款式设计软件
- 多层样片分解、工艺装配结构图、曲翘处理、放加工余量、绷帮裕度、标注分针眼软件

- 帮样及底样扩缩软件
- 植面 B 样条拟合、消隐线框图、三维轮廓图显示软件

- 成鞋浓淡图象旋转造型软件
- 样片库管理软件
- 样片图及排料图绘制软件

- 优化排料及制鞋工艺卡管理软件

整个系统在多重汉化菜单统一管理下运行,有较完善的用户输入操作提示,输入信息合法性检查及 UNDO 功能,提供用户一个统一、友善、简明、可靠的接口。

全部系统程序量超过五万余条。

跟同类系统相比 SMD-6410 具有显著的技术特色。

联机植面数据采集。

制鞋的设计工作是以鞋植作为基础的,设计数据都是直接或间接来源于鞋植。但长期以来,由于缺乏价格适中的植面数据测量设备,设计数据全部由人工测量,而鞋植又是不规则的空间几何体,致使测量工作既比较困难又不易测准。在这种条件下,国内兴起的制鞋 CAD 研制工作所使用的鞋植面数据大多也只能由人工测量输入,少数使用部颁数据(往往与厂家实用鞋植有出入),也有使用昂贵的精密机械测量用的三坐标测量机测取实验数据。但不管怎样,都未能从根本上解决好制鞋 CAD 所需要的原始数据采集问题,严重阻碍了它的发展。国外从八十年代中期开始为制鞋 CAD 开发了机械测量设备,有力推动了制鞋 CAD 的飞速发展。本系统在国内首先采用自行研制的价廉实用的植面测量仪作为植面数据自动采集联机设备,从根本上解决了困扰人的制鞋 CAD 原始设计数据采集问题,开始了国内同类系统先河,在这方面达到国际同类系统相同水平,而且具备更优的性价比。

- 有限元植面展平法。

早在六十年代,轻工部制鞋研究所组织国内制鞋业科研生产部门,研究出三角控法下近似植面轮廓展平方法。由于该法制图过程复杂,所绘展平图还要作符植处理,要用较多数学知识,难为一般设计人员接受,随着计算机普及和国际制鞋 CAD 浪潮冲击,此方法成为国内制鞋 CAD 作植面展平的基本方法,但由于需输入植面 40 余个特征部位长度值,大多是沿植面曲线,不仅手工难以测量,就是全植面数字化后也很难计算准确,故影响了此方法的实用性。此外,国内还有在此方法基础上一些非本质变化的新方法,也未能避免上述困难。再有把植面前、中、后三部分近似当作规则几何曲面的方法,由于难以适应植型千差万别变化,实用性差。本系统在总结前人工作成果基础之上,采用了有限元计算方法及为理想的收敛准则,圆满解决了植面展平问题,跟其它方法比较,它具备下述显著特点:

1) 这是一种植面展平方法,而不仅仅是植面轮廓展平,因此能提供更丰富的机助设计信息。