

ISSN 1001-3695

計算機應用研究

1993

3

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



新潮電腦

XINCHAO'S COMPUTERS

時代新潮

GIVE YOU A STYLISH FEELING IN YOUR WORK



新潮系列微機
高科技的象征
國產微機的新里程

- 先進的設計和工藝
- 高度的兼容性
- 廣大的維修服務網



- 攝像機及其應用系統
- 網絡工程
- 開關電源
- 亞森 UPS 卡
- 系列工業控制計算機
- 稱量儀表
- 微機電子皮帶秤



四川新潮計算機產業集團公司
SICHUAN XINCHAO COMPUTER ENTERPRISES GROUP

地址：四川成都新濤路四號

電話：448100 443074

傳真：(028)444115 電挂：4615

郵政信箱：成都 606 信箱

郵政編碼：610051

四川新潮计算机产业集团公司

四川新潮计算机产业集团公司是中国西部地区第一家专门从事计算机及衡器等高科技产品的开发、生产、经营销售和技术服务的全民所有制企业集团。

集团公司人才荟萃,现有职工 2000 余人,其中,中高级技术人员 700 多名,注册资本 2000 万元,拥有全国第一条完整的表面组装(SMT)生产线和具有国际水平的电源测试设备以及引进德国申克公司的工业称量控制系统的制造技术,具有创亿元以上产值的生产能力,年出口创汇 2500 多万美元。在全国计算机行业中名列前茅,集团公司迄今已先后开发、生产了新潮系列微机 XC—PC、XC—286、XC—386、XC—486、10 余种开关电源及卡式 UPS 电源、CCD 摄象器及其应用系统、100 多项软件产品、工业控制机、应变计、传感器、应变仪、电子皮带秤、自动计量配料系统、网络系统以及办公自动化系统等,在全国各地均有经营、销售和技术服务网点。

集团公司的高级技术人员在 STDI 工业控制机、网络工程、通讯系统、机房工程、轻印刷系统、工程工作站、CAD 等方面,进行了卓有成效的开拓、研究和服务。迄今为止,上述各方面已先后荣获国家、部、省的几十次奖励,硕果累累,斐声国内外。

集团公司拥有一支技术层次较高的技术服务队伍,多年来在计算机及衡器的销售、维修、技术培训等方面为社会各界提供了优质的全面服务,为社会各界维修各类微机 4 万台次以上,培训计算机技术人才 9 千多名,同时,集团公司在全国及省内有重庆、绵阳、自贡等数十个分部和维修站,极大地方便了各界用户。

集团公司十分注重国际间的经济技术合作,在深圳、蛇口等地有华德、雅德、鼎峰、新欣、合力等 8 家中外合资企业,引进了国外先进的技术、先进的设备、先进的管理方法。美国著名的 IBM 计算机就指名用华德的开关电源,它获得 UL 登记和加拿大 CSA 以及法国 TUV 证书。所属雅德公司年产 40 多万块 OEM 级别的计算机主机板以及各种功能卡,全部销往欧、亚、美洲各国。

集团公司以“开拓、创新、团结、奉献”为企业精神,立足国内,面向世界,真诚欢迎国内外各界朋友与我们携手合作。集团公司将以精湛的技术,优秀的工作质量,高品质的产品,期待着为您服务的机会。

总经理:吕金才

电话:(028) 443074, 448100 传真:(028) 444115 地址:成都新鸿路 4 号 邮编:610051

- | | |
|---------------------|----------------|
| ●市场部:成都新鸿路公司本部 | 电话:448100—3201 |
| ●第一经营部:四川电子城二楼 | 电话:628680 |
| ●第三经营部:成都一环路南二段 9 号 | 电话:554901—188 |
| ●展销中心:成都一环路东二段 14 号 | 电话:441965 |
| ●维修中心:成都新鸿路公司本部 | 电话:448100—3206 |
| ●川南公司:滨江路壹幢 | 电话:221117 |
| ●重庆分部:重庆人民路 7 号 | 电话:352656 |
| ●绵阳分部:临园东路 371 号 | 电话:24722 |
| ●培训中心:成都新鸿路公司本部 | 电话:448100—3501 |
| ●机房工程部:成都水碾河东贸旅馆二楼 | 电话:434295 |
| ●电子分厂:成都新鸿路 17 号 | 电话:443074—39 |
| ●仪表分厂:成都新鸿路 17 号 | 电话:443074—38 |
| ●机械分厂:成都新鸿路 17 号 | 电话:443074—24 |

《计算机应用研究》杂志办刊单位

四川省电子计算机应用研究中心
贵州省科委计算中心
安徽省计算中心
吉林省计算中心
内蒙古电子计算中心
青海省测试计算中心
四川省电子学会

新疆电子计算中心
甘肃省计算中心
广西计算中心
山东省计算中心
河南省计算中心
云南省电子计算中心

《计算机应用研究》杂志社董事会

董事长:周赛渝

董 事:孙传江 陆慰椿 闫长荣 王升亮
李天健 郑国基 叶大卫 乔中南
冯德成 朱 华 黎瑰常

《计算机应用研究》杂志编辑委员会

主任委员:张执谦

副主任委员:李泽民

委 员:贾洪钧 曾光初 龚宇清 罗海鹏
张湘金 张国栋 范德元 李文华
刘启茂 崔振远 刘铁军 杨剑波
余 凯

1993年第3期(总第53期)

出版日期:1993年5月

责任编辑:张 钢

计算机应用研究(双月刊)

JI SUAN JI YING YONG YAN JIU

(公开发行)

刊 号: CN51-1196/TP(国内)
ISSN 1001-3695(国际)

邮发代号: 62-68(国内)
BM 4408(国外)

主 编:张执谦

副主编:李泽民

编辑出版:《计算机应用研究》杂志社

通讯地址:成都市人民南路4段11号附1号

邮政编码:610041

电 话:(028)-582666 转 2055

印 刷:成都市新都华兴印务有限公司

订 阅 处:全国各地邮政局

国内总发行:成都市邮政局

国外总发行:中国国际图书贸易总公司

广告经营许可证:川蓉工商广字 005 号

每册定价:1.80 元

计算机应用研究 第 10 卷 第 3 期(总第 53 期)

目 次

综述与评论

图纸扫描输入与处理技术的发展与应用 吴天智 韩可莉 李心泽(1)

研究探讨

一种改进的用封锁技术清除隐藏线算法的研究 王乐挺 朱秋萍(4)

菜单生成工具 CMUTOOL 的设计和实现 王泽兵 陈增武(7)

面向对象程序设计方法探讨 赵 炜(10)

MS-DOS 下点阵汉字的多功能综合处理及实现 唐贵川 李志蜀(12)

大型结构多阶模态快速动画显示方法 王成良(16)

大型集成式管理信息系统的设计方法 王宗军(19)

应用实践

下拉弹出窗口式菜单及其实现 张 华 王志新 葛宜远 路甬祥(22)

一个中文下拉菜单工具的设计与实现 袁楚明 周祖德 王焱清(25)

图文并茂的立体投影式窗口汉字下拉式菜单的设计 陈廷槐 董玉坤 张胜利 唐德江 赵春霞(28)

为软件配置良好的用户界面 午锁平(29)

EGA/VGA 图形汉字屏幕的保存与恢复 赵晓东(30)

C 实现 EGA/VGA 图形的存取与打印 方建民(33)

在高级语言中直接调用汉字显示中断的方法 瓮正科(35)

HP3000 系统应用研究 刘启茂 蔡红梅 方 红(38)

软件的封面设计 徐 琦(41)

标准尺寸图形打印 张红庆 胡 泊(44)

一个解决微机“死机”的有效方法 张鸿鸣 刘铁军 张 科(45)

如何实现 DOS 系统下应用软件到 XENIX 的转换 郑 英(46)

利用 TSR 程序实现假脱机绘图 李大宏(47)

染色工艺分布式微机控制系统 范德元 常国权 李 强 姜向荣 林川宏(48)

一种高速 FAX-PC 接口卡的应用研究 孙俊杰(52)

PC 总线接口中 8255 与 7135 的应用 崔亦飞(54)

一种高性能游戏机控制器 王让定 王小牛(57)

维护维修

M2024 打印机常见故障的修复 陈永红 穆大明(59)

微型计算机开关电源的维修 万承兴(61)

IBM AT 微型计算机直流电源故障一例 赵明生(62)

动态资讯

本刊启事 (58,62)

简讯广告 (封 2、6、9、27)

APPLICATION RESEARCH OF COMPUTER

Vol. 10 No. 3 (Total 53)

CONTENTS

SURVEY

Develop and Application with Scan—input and Processing Technology of Drawing Wu Tianzhi et al. (1)

RESEARCH

Research for Clear away Hidden Line Algorithm by A Improve Block Technology ... Wang Leting et al. (4)

Designing and Realizing of CMUTOOL--Generated Menu Tool Wang Zebing et al. (7)

Probe into The Programming Method about Object-oriented Zhao Wei(10)

Many Function Processing with Chinese Character in Dot Matrix on MS-DOS and It's Realizing Tang Guichuan et al. (12)

.....

The Displaying Method with Quick Animation of Large Structure Wang Chengliang(16)

The Designing Method for Large Integrated Management Information System Wang Zongjun (19)

APPLICATION

The Window Menu with Push-down and It's Realizing Zhang Hua et al. (22)

Designing and Realizing about A Menu Tool of Chinese with Push-down Yuan Chuming et la. (25)

Designing The Push-down Menu about Stereo Projecting Display by Graphic and Chinese Character in Full

..... Chen Tinghui et al. (28)

Disposing Good User Interface for The Software Wu Suoping(29)

Saving and Restoring for Graphic and Chinese Character Screen on EGA/VGA Zhao Xiaodong(30)

Access and Printing The Graphic on EGA/VGA by C Fang Jianmin(33)

The Method for Display Interrupt by Call Chinese Character in High Level Language Weng Zhengke(35)

.....

Application Research in HP3000 Liu Qimao et al. (38)

Designing to The front Cover of Software Xu Qi(41)

Printing to Graphic of Standard Size Zhang Hongqing et al. (44)

An Effective Method for Solving Dead Halt on Microcomputer Zhang Hongming et al. (45)

How Realize in Comesion Application Program from Dos to XENIX Zheng Ying(46)

Realizing of Spooling Plot by TSR Program Li Dahong(47)

The Distributed Microcomputer Control System in Dyeing Technology Fan Deyuan et al. (48)

An Application Research in Interface of FAX—PC Sun Junsheng(52)

The Application about 8255 and 7135 of Bus Interface Cui Yifei(54)

A High Function Controller of Game Wang Rangding et al. (57)

MAINTENANCE

Repairing with Common Fault of M2024 Printer Chen Yonghong et al. (59)

Maintenance for Switch Supply of Microcomputer Wan Chengxing(61)

An Example about DC Supply Fault of IBM AT Microcomputer Zhao Mingsheng(62)

INFORMATION

The Notice (58,62)

News in Brief (cover2,6,9,27)

图纸扫描输入与处理技术的发展与应用

东方电机厂 吴天智 韩可莉 李心泽 (四川省德阳市 618000)

摘要 把现有积累的数量浩大的图纸输入计算机,是推动大中型CAD系统尽快投入工程实用的关键之一。有针对性地采用先进的图纸扫描输入和处理技术手段,是低成本、高质量、高效率建立计算机图库,缩短设计周期,提高设计水平的合理选择,具有十分明显的社会效益和经济效益。本文综合介绍了图纸扫描输入原理及当今各类产品的结构、特性、数据的矢量化处理及最新的光栅/矢量混合编辑CAD技术。描述了图纸扫描输入与处理技术的发展与工程应用的先进性、经济性和实用性。

关键词 扫描输入 矢量化 光栅/矢量混合编辑

1 前言

计算机辅助设计(CAD)技术的迅速发展与应用,使设计工作告别手工绘图,进入计算机世界变成了现实。设计师应用计算机,借助其非凡的能力就可以快捷高效地作出优秀的设计。设计方式的这种变革还为更充分地利用现有图纸资源,荟萃已有设计成果创造了良好的条件。在CAD系统中,纸不再是图形的载体,而由计算机取而代之。某些设计工作如(机械工程、建筑工程、地图等)就能利用产品发展的一定的继承性,以原图(图形)为依托,在计算机上方灵活地加以修改编辑,生成新的图形,产生新的图纸。这样做,将大大地减少人工修改图纸、描绘图纸、管理图纸的工作,促进图纸编辑、修改、管理的自动化,提高效率,缩短周期,减少劳动强度,从而加快设计速度,提高产品设计水平,大力开发新产品。其社会效益和经济效益都是十分明显的。同时,计算机图库的建立发挥了计算机检索迅速,拷贝方便,存储量大、密度高、便于管理的先进性,提高图纸管理的现代化水平。

2 CAD系统常用的图纸输入方式

目前,CAD系统采用的图纸输入方式主要有以下三种。

在计算机系统上用键盘、数字化仪、鼠标器输入。工作时,操作人员对照图纸,在CAD软件的支撑下,通过屏幕菜单、数字化仪菜单、按钮菜单、键入命令名称/数据等途径人机交互式地建立计算机矢量图形。这种方式比较流行,对硬件平台的要求不高。如原图形上标明了数据,其输入准确。没有数据的图形,输入则可能出现较大的误差。还可能出现遗漏、错位、字符差错等输入错误。输入速度较慢,工作量大。

通过数字化仪输入。采用这种方式建立计算机图形时,把图纸贴在数字化仪板上,操作者手持鼠标器在图纸上逐个实体描图跟踪,就可轻易地将图形转变成成为光栅数据输入到计算机中,或在CAD软件的支撑下,采用矢量方式,建立矢量图形。它的特点是,数字化仪应用电磁技术,提供超高分辨率和优异精度,但是每套输入系统都要配备一台大型数字化仪,耗资较多,操作费力,输入出错的可能性依然存在。

图纸自动扫描输入。在先进的扫描式图纸输入系统上,图形被自动地扫描输入计算机形成光栅图形数据被显示和存贮,极大地提高了输入速度和保真度,降低作业难度,减少输入综合成本。有针对性地采用图纸扫描输入,是建立CAD输入子系统和计算机图库的合理选择。也是图纸输入的方向。

本文着重描述图纸扫描输入技术的发展与应用。

3 图纸扫描设备

图纸扫描设备是自动地采集图形信息并转变成数字信息的装置。其结构原理如图1示。

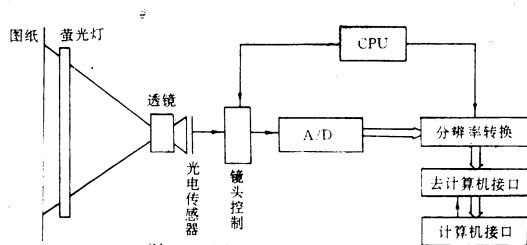


图1 图纸扫描机原理图

用CCD感应器或其它光跟踪镜头,包括最新的

光纤纤维镜头阵列构成扫描头。当扫描头与图纸按一定规则相对运动时,图纸上承载的图形即被反射(或透射)成光信号,经光电传感,光信号又转变成电信号,在数字化处理后形成光栅数据输入计算机存储起来。扫描产生的原始光栅数据量是较大的,典型的中等复杂程度的 A0 尺寸图纸以每英寸 200 点分辨率扫描,就要产生约 7.5MB 光栅数据量,应当进行压缩处理,数据量会大大减少。

随着光电、计算机等技术的发展,图纸扫描设备功能越来越强,性能越来越好,其主要特性有:

- 自动化程度高,操作简便;
- 扫描速度快;
- 分辨率高并可灵活选择;
- 可连续进行对比度调整,以达到最优采集;
- 和 CAD 主机联机。

目前,图纸扫描设备主要分成以下三种类型:

扫描附加器。扫描附加器设计独特,采用 CCD 感应器,安装在笔式绘图机的 Y 轴上,使一台机器具有了扫描/绘图双功能。工作时,扫描附加器沿 Y 轴运动,图纸沿 X 轴间歇移动,逐行扫描。由于扫描附加器感应面积小,扫描速度低,每秒钟仅扫描数平方厘米,对工程应用有局限性。但这种扫描附加器价格低廉,节省空间,能实施大幅面图纸的扫描。

图文扫描仪。这类扫描仪多为平板台式, A3/A4 幅面,价格便宜,品种较多。其中性能优良者能提供完美的扫描结果,配备的图形处理软件具有较强的后处理功能,如灰度编辑、旋转、缩放、插入或删除等。图文扫描仪用来输入图纸嫌幅面太小,大幅面的图纸要分块扫描,尔后在计算机上重新拼起来。它最适合于采集图片、技术文件资料。其它应用范围广泛,如专家工具、档案贮存、出版系统等。

大幅面图纸扫描机。它是一种工程图纸自动扫描和数字化设备,集中了扫描输入技术的功能,具有低成本、高质量、高效率采集图形数据的特点,并使系统整体化的工作变得简单容易。大幅面图纸扫描机既可作为一台独立设备使用,也可用作 CAD 系统的前端外部设备。操作简便,提供最高为每英寸 600 点的多种分辨率,能处理 A0 尺寸及加长加宽的图纸,扫描速度快。扫描机内置微处理器,灵活调整对比度以实现最优采集,高可靠性。这类扫描机最适合于工程规模输入图纸。目前某些产品的市场售价降至 1 万多美元。

除上述三类之外,还有视频摄象输入等方式。

图纸扫描机与计算机主机,相配的软件组合成一个完整的图纸扫描输入处理系统。应用光栅编辑软件,消除原图中的咖啡斑、老化斑,复新陈旧的图

面及对输入图纸进行修改和编辑。用转换软件完成光栅数据的矢量化,将数据输入到 CAD 系统,完成设计信息的转换。

4 常用的图形编辑修改方式

目前,绝大多数的 CAD 软件包支持的是矢量图形。它能够满足自动设计的要求,并可建立 IGES、SIF、DXF 或其它 CAD 系统的本机格式,直接通讯或联机交换提供 CAD 各系统间的链接与转换,实现图形数据共享。在这种背景下,常用的图形编辑修改方式主要有两种。一种是人机交互方式,即调用 CAD 软件包的编辑修改功能命令,删除、缩放、移动、旋转图形的某一部分或输入新的图形拼接,从而生成一幅新的图形。另一种方式为参数方式,自动设计系统要求采用这种方式。在该方式里,为拟定要变动的每一个实体编制一段程序,把确定其尺寸的数据作为变量处理,并将程序和绘图软件链接起来。当调入/输入设计计算或优化设计得出的实际尺寸数据时,新图形就自动地生成与标注。

人机交互以矢量方式建立的图形可以直接进入上述 CAD 系统,而扫描输入建立的光栅图形是不能直接进入矢量系统的。

5 光栅数据的矢量化处理

在图纸的扫描输入处理过程中,扫描问题前面已经解决,现在的问题是扫描形成的光栅数据到矢量数据的转换——即光栅数据的矢量化。从某种意义上讲,光栅数据的矢量化是应用图纸扫描输入技术的重要环节。

一般情况下,工程图纸的图形复杂,绝大多数现存的图纸来自手工绘制,图形上必然存在不规整的缺陷,比如线条欠光滑、宽度不均匀、圆弧曲率不一、落笔接头明显等,都会给矢量化处理带来困难。

目前,图纸扫描输入系统较多地采用自动/人机交互两种方式实现矢量化转换。

自从图纸扫描输入技术问世,自动矢量化技术便应运而生,不断发展。至今,国内外市场上自动矢量化软件很多,也不乏佼佼者。在采用自动矢量化技术的图纸扫描输入系统里,先运行光栅处理软件,消除光栅图形中的杂点(如噪声点)、咖啡斑、老化斑等,复新陈旧的图面。对光栅图形进行编辑、修改、字符辨识搜索后,调用系统的矢量化功能,对光栅数据进行自动矢量化处理,使之转换成矢量数据,显示矢量图形,形成矢量图形文件。自动化程度高是这种系统的突出特点。但由于手绘图纸的图形精度缺陷,即使最优秀的自动矢量化软件,也会产生矢量图形失真。通过计算机自动完成光栅数据向准确的矢量数据转换,国际上至今没有突破。例如,一条长线段可

能被误处理成几条顺序相连的短线段,一个圆被分成首尾相接的几段圆弧,虚线、中心线会变成一些线段。对字符自动识别时也有一些误差,如S与5、O与0、I与1不易区分等。在自动矢量化过程中,为减少图形精度缺陷的影响,采用了线条拟合方法。然而无论采用哪种拟合方式,矢量失真依然存在。如以线条中心线为依据拟合,人工设定线条的逼近允差,对曲率大的图形矢量失真较少。而以线条边界所限定的范围为拟合依据,曲率大的图形则会产生较严重的矢量失真。带有失真的矢量图形进入CAD系统就易引起错误,必须进行修正编辑后处理,使矢量图形与原始图形的所有实体几何特性一致,字符相同。修正编辑后处理通过专业人员对矢量图形的识别、比较、修改来实现,技术要求较高,工作量较大。一张中等复杂程度A0尺寸图纸的修正编辑后处理在图形工作站上约耗时2—4小时。

自动矢量化产生失真的根源是目前计算机尚不能智能识别图形缺陷,要解决这个问题难题很多。在此情况下,CAD覆盖技术的开发与应用为消除矢量化失真开辟了新路。在CAD覆盖技术的支持下,系统放弃计算机对图形的自动识别、拟合,改为人工辨识、人机交互作业完成矢量化工作,即所谓人机交互方式。八十年代末期推出的这类系统运行时,光栅图形直接显示在屏幕上,操作人员逐步实体调用CAD软件的作图命令,在对应的光栅图形上建立起矢量图形,光栅图形与新生成的矢量图形屏幕重叠,又以不同颜色层次区分。直接完成屏幕跟踪矢量化,并根除了矢量失真。改对照图纸为对照屏幕,避免了输入错误。图2给出了示例。

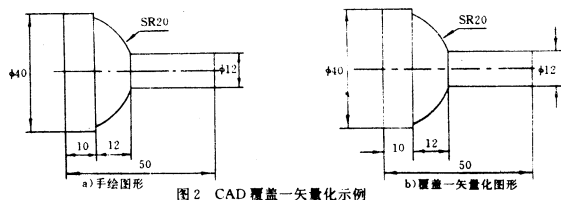


图2 CAD覆盖—矢量化示例

CAD覆盖技术提供了人机交互上的方便,可节省大量工时,PC机作为硬件平台减少了费用,加上对人员技术能力要求低,因此能够迅速、准确和低费用地把图纸转化为CAD矢量数据,有利于工程规模应用。CAD覆盖技术一推出,立即得到广泛应用,推动了图纸扫描输入技术的发展。

6 光栅/矢量混合编辑

在手工绘图方式中,一幅图形的生命仅存在于

绘图过程中,即从落笔开始到抬笔结束。如要改变这幅图形,哪怕是修改某个局部,必须在另一张纸上做一次重画,即使拿到CAD系统上去修改,也得把这幅图形输入。然而许多新项目的设计往往是以前某项设计的继承和翻版。在现存的大量图纸中蕴藏着丰富的可再生资源,不乏先进的技术和优秀的设计,往往只要把那些图形的某部分加以修改就产生一份新设计。为满足用户的需要,也往往要修改原设计的某些局部。

事实上,为局部的修改而重新画图是非常不合算的。即使应用图纸扫描输入技术,在图纸扫描输入处理中,光栅图形与矢量图形分立使得矢量化工作不可缺少。无论自动矢量化还是人机交互矢量化,工作量都是较大的,目标都是为适应一般的CAD系统。实际上,CAD中大量采用的并非自动设计,而是人机交互地对图形进行编辑修改,存在着无需矢量化的可能。最新的CAD光栅/矢量图形混合编辑技术针对局部修改的需要,把两者有机地融合起来,使工程图纸的扫描输入/局部修改进入了一个新的里程。

光栅/矢量图形混合编辑技术提供崭新、方便而有效的修改设计手段,让旧图纸焕发新生命(如图3示)。

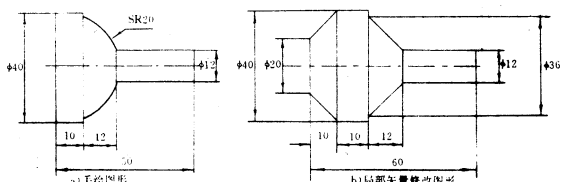


图3 CAD光栅/矢量混合编辑示例

该技术只需简单地扫描旧图纸,用现有的CAD系统,使扫描光栅图形不经矢量化、DXF或DXB处理就可加载为CAD系统中,拼接CAD矢量图形。用“粘贴”“切割”“隐显”等方式直接在光栅图形上建立修改部位的矢量图形,形成光栅/矢量混合图形,而不需要对图形全部实体跟踪一遍。在这个图形上,可以用CAD软件的所有矢量命令进行混合编辑,交互式地选择图形显示,支持多重参考图形,拼接完美。既省去了重新画图,又免除了费事的矢量化。整个过程如同用铅笔、橡皮在纸上改图一样直观方便。极大地提高了设计速度,缩短了出图周期。技术经济效果突出。当然,如果保留的原手绘图形缺陷过于明显,将导致新生成的混合图形不太协调。

7 结语

建立计算机图库是建立工程 CAD 实用系统的重要环节。先进的图纸扫描输入与处理技术展示了其应用的经济性和实用性。采用这一技术手段是低成本、高质量、高效率建立计算机图库的有效途径和缩短设计周期,提高设计水平的合理选择。而要达到最佳效果,必须针对应用对象的实际情况合理地选择图纸扫描设备、矢量化方式、修正编辑方式、图库的管理模式等,搞好系统配置,使其充分发挥作用,而无后顾之忧。这些技术的推广应用,还可以在更广

泛的设计领域产生更好的技术经济效益。

8 参考资料

- 1 Edit Your Paper Drawings in Auto CAD Overlay ESP.
- 2 LaYer Separation Package.
- 3 CAD Raster Process Your Scanned Drawings in Auto CAD and its Application.
- 4 Audre Intelligent Image System.
- 5 马灼奎,《关于图形输入方式的探讨》.

一种改进的用封锁技术消除隐藏线算法的研究

武汉东路新技术研究所 王乐挺 (430072)

武汉大学空间物理与电子信息系 朱秋萍 (430072)

摘要 本文探讨了针对 $Z=f(x,y)$ 型三维曲面图形的投影画法中,一种改进的利用封锁技术消除隐藏线的算法。并对文献[1]中的原始算法和本文的改进算法所产生的图形结果作了比较及讨论。

关键词 消隐算法 封锁技术 投影转换

1 引言

针对曲面图形 $Z=f(x,y)$ 的平面投影画法及用以增强平面图形立体效果的消除隐藏线的方法一直是计算机图形学中使人感兴趣的方面。通常的消隐算法,是先画离视点近的线段,由近至远绘制,如果一条线或线的一部分在以前画出的线所包围的区域之内(即被隐藏)则不画出来。如文献[1]中的屏蔽方法和文献[2]中高程线画法皆属此类。在文献[1]中,还介绍了一种从远离视点的线段开始画,由远至近的封锁消隐算法,其特点是函数图形较远部分先全部画出,当函数较近部分画出时,再删除那些隐藏部分。然而在观察角度上有较大局限性,且消抹掉了函数图形中可能显露出来的下凹峰。本文在此算法的基础上,提出了一种改进的封锁消隐算法,克服了原算法的不足。

2 三维图形平面投影画法的坐标转换

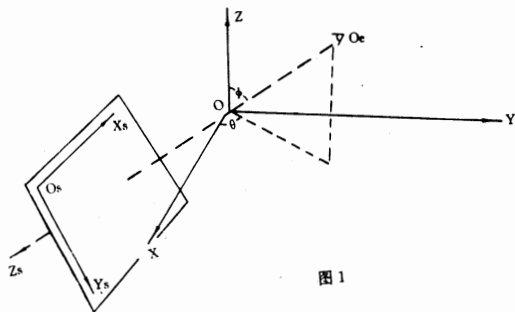


图1

文献[1]提出了一组方程,可将三维坐标转换成屏幕平面坐标。经过 $(X,Y,Z) \xrightarrow{\text{投影转换}} (X_s,Y_s,Z_s)$, 一个位于三维真实坐标系的 (X,Y,Z) 处的点将投影在屏幕平面的 (X_s,Y_s) 处,而 Z_s 反映了此点相对于视点的远近程度, Z_s 越小则此点离视点越近。真实坐标,屏幕坐标及投影角度等的示意图见图1。

3 由远至近封锁消隐算法

图1 X,Y,Z 为真实三维坐标, X_s,Y_s,Z_s 为屏幕坐标, O_e 为视点, θ, Φ 为观察角度(投影角度)。

设屏幕范围 $0 \leq X_s \leq X_m, 0 \leq Y_s \leq Y_m$ 。欲描述函数 $Z=f(x,y), -X_0 \leq X \leq X_0, -Y_0 \leq Y \leq Y_0$ 。

文献[1]中封锁消隐算法的原理用类 C 伪码语言描述如下:

定义绘图参数类型:

绘制算法{

给绘图参数赋值;

输入投影角度;

for ($x = -x_0; x \leq x_0; x = x + dx$)

{for ($y = -y_0; y \leq y_0; y = y + dy$)

{ $z = f(x,y);$

$(x,y,z) \xrightarrow{\text{投影转换}} (x_s,y_s);$

消隐画图{...

setcolor(背景色);

line (x_s,y_s,x_s,Y_m);

setcolor(作图色);

line ($x_s-dx_s,y_s-dy_s,x_s,y_s$);

...}

```

    })
}

```

可见,此消隐算法关键在于从当前作图点(X_s, Y_s)向屏幕底端平行屏幕Y轴以背景色作消隐线,将以前作的位于此点下部的隐藏点、线消抹掉。

此算法在 $0 < 90^\circ$ 或 $0 > 270^\circ$ ($0 \in [0^\circ, 360^\circ]$)的投影角度内对上凸图形的消隐十分成功,且依此法编程占用的存储空间小,确可谓优越。

不足之处,一是算法从对应于定义域边界端点($-X_0, -Y_0$)的点($-X_0, -Y_0, Z$)开始,沿X轴由负向至正向绘制一组平行于Y轴的曲线,当 $90^\circ < \theta < 270^\circ$ 时,绘线顺序相对于视点不再是由远及近,以此法消隐将出现较近的线段被较远线段消抹掉地错误绘制。二是函数曲面可能具有尖深的显露出来的下凹峰,被一消到底的消隐线抹去了。当从 $\Phi > 90^\circ$ 的投影角度绘图时,不属隐藏区域的下凹图形也会被消抹去。

4 改进的封锁消隐法

为确保由远及近绘线,我们认为应对对应于定义域边界四端点的空间四点($-X_0, -Y_0, Z_1$)、

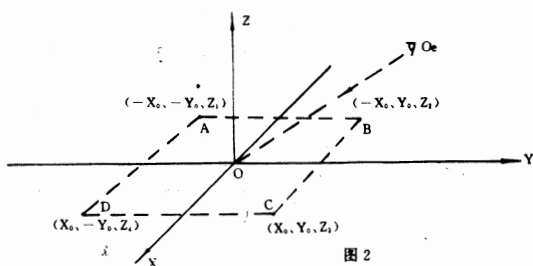


图2 $(-X_0, -Y_0, Z_1) \rightarrow Zs_1, (-X_0, Y_0, Z_2) \rightarrow Zs_2,$

$(X_0, Y_0, Z_3) \rightarrow Zs_3, (X_0, -Y_0, Z_4) \rightarrow Zs_4$

$(-X_0, Y_0, Z_2), (X_0, Y_0, Z_3), (X_0, -Y_0, Z_4)$ 作深度测定(示如图2),找出在当前投影角度下离视点 O_e 最远的点,作为绘线起点。查找方法为分别求各点对应的 Zs, Zs 最大的点即为最远点。算法如下:

$Peak[0] = Ax; Peak[1] = Ay * * ; Peak[2] = Bx;$

$Peak[3] = By;$

$Peak[4] = Cx; Peak[5] = Cy; Peak[6] = Dx; Peak[7] = Dy;$

start 算法{

int i;

float code[4], dmax;

for (i=0; i<=3; i++)

{x=Peak[2*i]; y=Peak[2*i+1];

z=f(x,y);

(x,y,z)——投影转换——>zs;

code[i]=zs;}

dmax=code[i]中的最大值(i=0,1,2,3);

for (i=0; i<=3; i++)

{ if (code[i]==dmax) break;}

return i;

}

以 $Peak[2*i], Peak[2*i+1]$ 为X,Y的起始值开始绘制就能保证由远及近的顺序了。

为表现函数图形中的下凹区域,我们认为应在绘制前先设立中界线(示如图3)。并认为出现在中界线上部的图形为上凸图形,下部的为下凹图形。在消隐时,由当前点作消隐线至中界线上而非至屏幕底端,从而同时符合了上凸图形靠前的较高线段隐藏靠后的较低线段和下凹图形靠前的较低线段隐藏了靠后的较高线段的屏蔽特点。

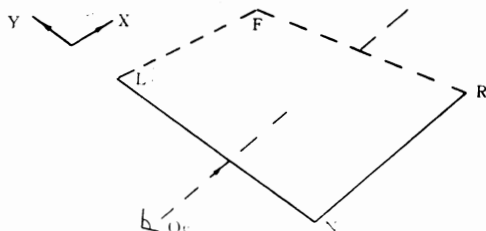


图3

图3 相对于视点 O_e, N 端点为最近点,由N点平行于X,Y轴向其两邻端点R,L作的曲线(图中实线)为中界线

中界线无需被绘出,只需将线上各点对应的(X_s, Y_s)点的 Ys 值放入设立的limit数组中对应于 Xs 的单元内即可。

其形成算法为:

Limit 算法{

float limit[Xm];

利用类似start算法的方法确定最近点;

/* 设最近点为N,两边界邻点为R,L. */

y=Ny;

for (x=Nx; x!=Rx+dx; x=x+dx)

{z=f(x,y)

(x,y,z)——投影转换——>(xs,ys);

...

for (linkx=xs-dxs; linkx<=xs; linkx=linkx+1)

{linky=linky+dys/dxs;

limit[linkx]=linky;}

x=Nx;

for (y=Ny; y!=Ly+dy; y=y+dy)

{同上;}

}

在绘制消隐线时,按下语句

```
setcolor(背景色);
```

```
Line (Xs,Ys,Xs,limit[xs]);
```

来作消隐,对上凸图形和凹图形都很成功。

考虑到描述的函数定义域分布的对称性 * *
*,让 start() 通过寻找最小 Zs 值来确定最近点,
就可同时确定最远点和开始建立中界线,而且可把
建立中界线的步骤放到消隐绘制函数中。具体编程
可按以下算法:

定义绘图参数类型;

```
float limit[Xm],Peak[8];
```

```
enum flag{true,false}F;
```

```
...
```

绘制算法:

```
int i;
```

```
给绘图参数赋值;
```

```
输入投影角度;
```

```
确定图形方式模式;
```

```
...
```

```
i=start();
```

```
F=true;
```

```
plot(Peak[2*i],Peak[2*i+1]);
```

```
F=false;
```

```
plot(-Peak[2*i],-Peak[2*i+1]);
```

```
}
```

```
void plot(a,b)
```

```
float a,b;
```

```
{if(F==true)
```

```
{以(a,b)为N点,建立中界线;}
```

```
else{以x=a,y=b为起点开始消影隐绘图;}
```

```
}
```

5 参考文献

- 1 Chan S. Park 著,陆吟芳,张燕生,郭超美译,《交互式微型计算机图形学》,清华大学出版社,(1988).
- 2 金廷赞著,《计算机图形学》,浙江大学出版社(1988).
- 3 徐金梧,刘治钢,张思宇,倪伟敏编译,《Turbo C 使用大全》,北京科海培训中心,(1990)
- 4 周公度编著,《结构化学基础》,北京大学出版社,(1989).

*:本文提及的算法皆用类 C 伪码语言描述。

* *:Ax,Ay 分别表示 A 点的 x,y 坐标值,Bx,Cy 及后文中 Nx,Ry 等含义与此类同。

* *:函数定义域不对称分布时,可通过对表达式作小的改动,以达到对称分布。

SS—CEGA 双星汉卡汉字系统扩展程序

广州中山大学计算中心 肖俊良 (510275)

双星汉卡的设计既考虑了汉字字符显示又考虑了 EGA 显示,因而具有良好的兼容性。双星汉卡的 EGA 显示与标准的 EGA 完全兼容,汉字字符显示称为 014 板,其基本显示缓冲区在地址 0B800;0000,汉字显示缓冲区在 0B000;0000。一般直接访问显示缓冲区的西文软件仅把字符送到基本显示缓冲区,即使是汉字内码也是如此,因而把汉字作为高位 ASCII 码显示。如果要显示汉字,必须同时把汉字内码送到基本显示缓冲区和汉字显示缓冲区,这是西文软件在汉卡上的一般汉化方法。

本软件通过把基本显示缓冲区内的汉字内码中的区位码取出,再经下列的变换,把变换的结果的低字节送到基本显示缓冲区,高字节送到汉字显示缓冲区,最后把 1Fh 送入汉字显示缓冲区的下两个字地址处:区号 * 94 + 位号 如果位号 < 15

(区号 - 6) * 94 + 位号 如果位号 ≥ 15

即可在屏幕上显示汉字,达到一般的西文软件不必经过汉化即可直接显示汉字的目的。

本程序使用直接访问基本显示缓冲区的方法取

出汉字区位码加以处理。但调用显示中断 int 10h 的程序不必经此处理,这就很难区分哪些需要经此处理,哪些不必。为了防止这种现象,把 int 10h 的有关调用也改成直接送到基本显示缓冲区。

本软件的早期版本用于长城汉卡,汉字显示缓冲区在 0C000;0000。针对双星汉卡的汉字显示缓冲区作适当的修改后,可以显示汉字,光标却消失了。最后根据汉卡可以显示特殊的特点,把下划线属性作为光标使用,并保持光标闪烁的特性,经使用完全符合要求。

通过本程序,一般的西文软件不必经过汉化即可处理汉字,不论是否采用直接荧光屏存取的方法,只有下列情况例外:

- (1)使用多个显示页,并且基本的显示页不是 0。
- (2)不使用高位 ASCII,即图形字符。

本程序使用 IBM PC 汇编语言编写,在 80286 机及原版 DBASE III PLUS CLIPPER 5.01 软件下通过。

需要程序清单者可同作者或本刊联系。

菜单生成工具 CMUTOOL 的设计和实现

浙江大学计算机软件研究所 王泽兵 陈增武 (杭州 310027)

摘要 弹出式菜单和树形菜单是交互式菜单处理的两种常用结构,CMUTOOL 是基于树形结构的,用 C 语言开发的菜单生成工具。本文介绍了从系统功能树导出菜单结构的方法,给出了 CMUTOOL 的菜单表示方法、接口形式以及相应的数据结构,并以 C 语言形式给出了 CMUTOOL 的主要处理算法。

关键词 菜单 菜单项 功能树

1 系统功能与菜单结构

在交互式处理系统中,菜单是人与系统的接口界面,系统通过菜单来驱动各功能模块,用户通过点用菜单来执行系统功能。一个系统的功能逻辑上可用一个树形结构来表示,作为本文的示例,不妨假设下面是一个简单图形处理系统的功能结构树:

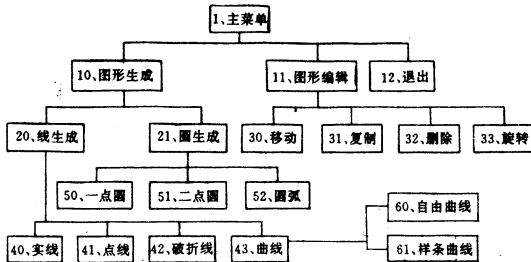


图1 功能树

图中节点的数字是由用户定义的用于标识的菜单号,它的具体作用后面将要详细介绍。

从上面的功能树可导出相应的菜单结构为:

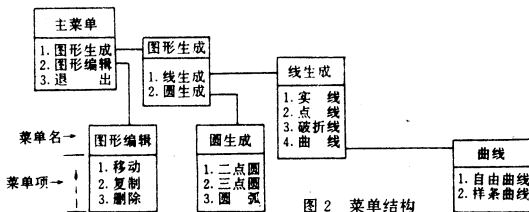


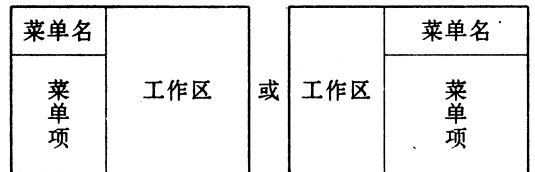
图2 菜单结构

从上图可见功能树和菜单结构存在如下关系:

- 树根 主菜单
- 树枝 子菜单
- 树叶 系统功能

这种树形菜单结构的应用特点可归纳为:

① 屏幕划出一块固定区域用作显示菜单,称为菜单区,其屏幕结构一般为:



② 菜单区分为两部分:菜单名和菜单项,如图2示

③ 光棒所在的菜单项称为当前菜单项,可移动光棒来改变当前菜单项

④ 菜单的点用有两种方式:

- CR 键确认当前菜单项为选中菜单
- 直接键入菜单项前的数字

⑤ 功能树中非叶节点对应的操作为:

- 显示相应的菜单,它的菜单由其儿子节点组成
- 等待用户点用菜单
- CR/菜单项前的数字:进入下层菜单
- ESC:返回上层菜单

⑥ 功能树中叶节点对应的操作为:

- 执行对应的系统功能,此时菜单区可用作提示
- 返回上层菜单

2 CMUTOOL 的菜单表示

为了降低算法的复杂度,CMUTOOL 采用一个一维表来表示菜单结构,表元素是一个菜单项,每一个菜单项是一个五元组,表示为:

$\langle no \rangle, \langle father \rangle, \langle oldson \rangle, \langle brother \rangle, \langle name \rangle$

这里 $\langle no \rangle, \langle father \rangle, \langle oldson \rangle, \langle brother \rangle$ 是非负整数, $\langle name \rangle$ 是字符串,它们分别表示:

$\langle no \rangle$: 该项菜单的标识号,也称为菜单号

$\langle father \rangle$: 对应的父节点的菜单号

$\langle oldson \rangle$: 大儿子菜单号

<brother>, 右兄弟菜单号

<name>, 菜单名

当某项菜单对应的 <father> 或 <oldson> 或 <brother> 不存在时用 0(NULL) 来表示, 这样 <father> 和 <brother> 为 NULL 的节点是根, <oldson> 为 NULL 的节点是叶。假设根节点的 NO 总是 1, 根据以上定义, 与图 2 对应的菜单结构可表示为:

no	father	oldson	brother	name
1	0	10	0	主菜单
10	1	20	11	图形生成
11	1	30	12	图形编辑
12	1	0	0	退出
20	10	40	21	线生成
21	10	50	0	圆生成
40	20	0	41	实线
41	20	0	42	点线
42	20	0	43	破折线
43	20	60	0	曲线
50	21	0	51	二点圆
51	21	0	52	三点圆
52	21	0	0	圆弧
30	11	0	31	移动
31	11	0	32	复制
32	11	0	33	删除
33	11	0	0	旋转
60	43	0	61	自由曲线
61	43	0	0	样条曲线

图 3 菜单表示

3 CMUTOO 的接口设计

CMUTOO 的接口由三部分组成:

3.1 菜单配置文件 菜单配置文件是由用户定义的 TEXT 文件, 它包括以下三项内容:

① 菜单区结构, 它由以下信息组成:

<x,y>: 菜单区的起始坐标

width: 菜单区的宽

height: 菜单区的高

step: 菜单项的高

② 菜单颜色配置, 它由以下信息组成:

bklr: 菜单区背景色

titlelr: 菜单名背景色

barclr: 光棒颜色

textclr: 字的颜色

③ 菜单结构, 它由以下信息组成:

n: 菜单项数

<no-1>, <father-1>, <oldson-1>, <brother-1>,

<name-1>...

<no-n>, <father-n>, <oldson-n>, <brother-n>,

<name-n>

3.2 菜单处理程序表 由前面讨论可知对于功能树非叶节点的处理程序是统一的, 其主要操作为显示菜单和等待用户点用菜单, 所以菜单处理程序表只需列出叶节点的处理程序。其结构为:

节点号 处理程序名...

3.3 接口函数

• CreatMenu: 读入菜单配置文件和菜单处理程序表, 建立菜单内部结构

• Dspmenu: 显示当前菜单

• DoFun: 执行对应的处理程序

交互式系统的主控算法可用接口函数定义为:

void main(void)

{

SysInit(); /* 系统初始化 */

Creatmenu(); /* 读入菜单配置文件和菜单处理程序表, 建立菜单内部结构 */

while(1){

DspMenu(); /* 显示当前菜单 */

DoFun(); /* 执行相应的处理程序 */

}

4 CMUTOO 的实现

4.1 数据结构定义

• 菜单节点结构

struct MenuNodeType{

int no; /* 节点号 */

int father; /* 父节点号 */

int oldson; /* 大儿子节点号 */

int brother; /* 右兄弟节点号 */

char *name; /* 菜单名 */

void (*fun)(void); /* 菜单处理程序指针 */

};

这里 fun 是一个函数指针, 指向该菜单项所对应的处理函数。

• 菜单结构

struct MenuInfoType{

int n; /* 菜单长度 */

int x,y; /* 菜单区的左上角坐标 */

int width,height; /* 菜单区宽和长 */

int step; /* 菜单项的高 */

int bklr; /* 菜单区背景颜色 */

int titlelr; /* 菜单名背景色 */

int barclr; /* 光棒颜色 */

int textclr; /* 菜单文字颜色 */

int len; /* 当前菜单项数 */

int cp; /* 当前菜单的下标 */

struct MenuInfoType *head;

/* 菜单表头指针 */

};

这里菜单表是一个线性表, 其表头指针为 head, 它是根据配置文件和处理函数表由菜单建立

算法动态生成的, len 是当前菜单的项数, 通过搜索菜单表得到, cp 是存放当前菜单在菜单表中的下标。

4.2 几个主要算法

设 Menu 定义为: struct MenuInfoType Menu

4.2.1 菜单表生成算法

void CreatMenu()

- ①读入菜单配置文件;
- ②根据菜单项数申请菜单表空间;
- ③根据菜单配置文件和处理程序表填菜单表;
- ④菜单表正确性检查;
- ⑤如果发现有错误则提示错误信息退出系统;

4.2.2 显示菜单算法

根据 Menu.cp 所指的菜单项, 在菜单区显示由它构成的子菜单, 主要有显示菜单名和通过搜索菜单表显示菜单项, 下面是用 C 形式描述的显示算法:

```
void DapMenu(void)
{
    int x, y, pos, n = 0, no;
    char buf[80];

    SetMenuPort(PUSH); /*设置菜单窗口 */
    FillBar(0, 0, Menu.width, Menu.height, Menu.bkclr); /*填菜单区背景色*/
    line(0, 0, 0, Menu.height);
    FillBar(1, 0, Menu.width, Menu.step, Menu.titlebkclr); /*填菜单名背景色*/
    x = (Menu.width - strlen(Menu.head[Menu.cp].name) * 8) / 2;
    DepStrXY(x, 2, Menu.head[Menu.cp].name, Menu.titlebkclr); /*显示菜单名 */
    pos = Menu.head[Menu.cp].oldson; /*取大儿子节点 */
    y = Menu.step * 2;
    z = 4;
    while(pos != 0) {
        /* pos != NULL */
        no = SearchMenu(pos); /*求菜单号为POS的项在表中的位置 */
        itoa(++n * 10, buf, 10);
        strcat(buf, Menu.head[no].name); /*产生菜单项 */
        DepStrXY(x, y, buf, Menu.textclr); /*显示菜单项 */
        y += Menu.step;
        pos = Menu.head[no].brother; /*下一菜单项 */
    }
    Menu.len = n; /*置当前菜单项数 */
    SetMenuport(POP); /*恢复原窗口 */
    DepMenuBar(1, Menu.barcclr); /*显示光棒 */
}
```

4.2.3 点用菜单算法

SelectMenu 算法的返回值为: CR 或 ESC, CR:

选中菜单, ESC: 返回上一层菜单。

```
int SelectMenu(void)
{
    int no = 0, key, n, pos = 1;

    while(!no) {
        if((key = ReadKey()) != 0) { /*接收键盘输入 */
            switch(key) {
                case ESC: no = Menu.head[Menu.cp].father; /*通到上一层菜单 */
                    break;
                case CR: no = GetMenuItemNo(pos); /*确认当前菜单, NO为选中的菜单号*/
                    break;
                case UP: MenuBar(pos, Menu.bkclr); /*关闭光棒 */
                    pos = (pos == 1) ? Menu.len : pos - 1; /*光棒上移 */
                    MenuBar(pos, Menu.barcclr); /*显示光棒 */
                    break;
                case DOWN: MenuBar(pos, Menu.bkclr); /*关闭光棒 */
                    pos = (pos == Menu.len) ? pos + 1; /*光棒下移 */
                    MenuBar(pos, Menu.barcclr); /*显示光棒 */
                    break;
                default: n = key - '0';
                    if(n > 0 && n <= Menu.len) /*KEY为菜单项前的数字? */
                        no = GetMenuItemNo(n); /*求相应的菜单号 */
                    else
                        Bell();
                    break;
            }
        }
    }
    Menu.cp = SearchMenu(no); /*求no在表中的位置 */
    return key; /*返回CR或ESC */
}
```

这里 SearchMenu 函数为: 求菜单号为 NO 的菜单在表中的下标, GetMenuItemNo 函数为: 求当前菜单的第 N 项的菜单号。

4.2.4 执行当前菜单处理程序算法

```
void DoMenu(void)
{
    int no = Menu.head[Menu.cp].oldson;

    (Menu.head[Menu.cp].fun)(); /*执行相应处理程序 */
    if(!no) /*当前节点是叶? */
        Menu.cp = SearchMenu(Menu.head[Menu.cp].father); /*返回父节点 */
}
```

5 结束语

利用 CMUTOOL, 只要根据系统分析得出的功能树, 定义好配置文件和菜单处理程序表, 即可生成系统的菜单, 这无疑将提高软件开发效率。CMUTOOL 菜单生成工具在 CAD 系统的开发中得到应用, 效果良好。

有线电视微机选台器在宁制成

南京凯达电子有限公司最近研制成功凯迪牌有线电视微机选台器, 该选台器采用微机技术和锁相频率合成技术。是一种高新技术产品, 普通电视机装上这种选台器后, 可使电视机由 VHF 频段的 12 个频道增加到 47 个频道, 能接收到所有的有线电视节目, 还能使原来不具遥控功能的电视机具有遥控功能, 并可防止儿童任意开关机, 还有利于电视机频道按键的维修和使用。(李相彬)

面向对象程序设计方法探讨

兰州高等工业专科学校计算机系 赵 炜 (730000)

摘要 面向对象的技术是 80 年代以来国外关注的问题,特别是近几年,面向对象的研究遍及计算机软、硬件系统的各个领域。面向对象程序设计是一种实用的程序设计方法学,它采用全新的方法来求解问题,其思想被认为是 80 年代的程序设计,从更高更广的角度研究面向对象的方法,已成为 90 年代的热门课题,这里就面向对象设计方法和传统的程序设计方法之比较,谈谈自己的看法。

关键词 程序设计方法学 结构程序设计 面向对象程序设计

1 引言

计算机科学技术的发展速度之快,是其它任何学科、技术都无法比拟的。自从世界上第一台计算机的诞生和第一个高级程序设计语言出现以来,时间也就是四、五十年,然而,各种各样的高级程序设计语言如雨后春笋般地发展起来,现已投入使用的少说也有几百种,但是在程序设计方法学上有重大突破的却很少。七十年代在程序设计方法学研究中获得了三个重要成果:①数据抽象机制,②异常处理机制,③并行控制机制。对方法学的研究,在八十年代出现了 Ada 语言,三个特性在 Ada 语言中得到了充分体现,可以毫不夸张地说,Ada 语言是第四代计算机具有代表性的语言之一。

在程序设计语言及其方法学的发展过程中,一个主要论题是研制用来处理抽象的工具。抽象第一次有意识地用在程序设计方法中是在五、六十年代。所谓抽象就是系统的简化描述或规格说明,它强调系统中的某一部分细节或特性,而忽略了其它部分。一个好的抽象应该强调对读者(即用户)至关重要的信息,而忽略至少目前不重要或转移视线的细节。到了七十年代早期,程序设计方面出现了一种新的方法学,这就是“逐步求精”或叫做“自顶向下程序设计”,它通过不断细化的中间阶段,使目标说明逐步过渡到最终代码。另外,逻辑程序设计以其强有力的知识表达能力,严密的逻辑基础和有效的实用系统为广大计算机工作者所欣然接受。人们在对其研究的基础上取得了许多可喜的成果。随着计算机科学技术的发展,软件设计方法不断涌现,面向对象程序设计(Object-Oriented Programming)就是近年来发展起来的新的软件设计方法。它集数据抽象、抽象数据类型、类继承为一体,使软件工程公认的模块化、信息隐蔽、抽象、局部化、重用性等原则,在面向对象语言机制下得以充分实现。由于面向对象技术对于软件工程学面临的困境和人工智能遇到的障碍都是

一个颇有希望的突破口,因而受到了计算机研究人员的高度重视,面向对象技术给软件重用带来了新的希望,同时使系统的可修改性、可移植性、可靠性和兼容性都大大的提高。面向对象的设计方法强调了系统设计之前的系统分析,强调以系统中的数据或信息为主线,全面、系统、详尽地描述系统的信息,建立系统的信息模型,指导系统的设计。从某种程度上讲,这是一种数据驱动的系统分析与设计方法。在 80 年代,随着计算机技术的普及,其应用领域不断扩大,诸如数据库系统、专家系统、实时过程控制系统、系统软件、自动代码生成、人工智能、计算机集成制造系统等。软件系统也变得愈来愈大,愈来愈复杂。尤其在程序设计方法学研究中,开始采用面向对象程序设计方法,已经开发出了十多种面向对象程序设计语言,如 Smalltalk、C++ 等,有人认为,九十年代,在程序设计领域中,面向对象设计方法的地位和作用将会超过八十年代盛行的结构程序设计。

2 面向对象的程序设计方法

面向对象方法的基本出发点是尽可能地模拟人类的思维方式,追求自然地刻画和求解现实世界中的问题。它以信息隐蔽和抽象数据类型概念为基础。在传统的程序设计方法中,由于将功能抽象和数据抽象分开处理,导致软件结构和问题结构的不一致性,从而给软件维护和软件理解带来困难。而面向对象方法把传统中所有资源,如数据、模块以及系统都看成对象,每个对象把一组数据和一组过程封装在一起,使得这组过程可对这组数据进行处理,并在定义对象时可以规定外界在其上操作的权限。这一方法直接、自然,可以使设计人员把主要精力放在系统一级上,而对细节问题可以较少地关心。面向对象程序设计方法可以概括为以下五个基本概念,它们构成了面向对象程序设计语言的核心:对象(Object)、类(Class)、实例(Instance)、方法(Method)和消息(Message)。

对象是数据以及可对其施行的操作所构成的独立单位的总称,也就是说对象是一个封装数据和操作的实体。这里数据就是实例变量,操作就是方法。数据描述了对对象的状态,操作可操纵私有数据,改变对象的状态。可以通过对象对被求解的问题及其诸要素进行抽象,它将自己的数据结构定义和功能实现封装起来,实现了信息隐蔽和数据抽象,体现了模块化。类是一组具有相同数据结构和相同操作的对象的描述。一个类定义的是一种对象类型,描述了属于该对象类型的所有对象的性质,类相当于传统程序设计语言中的类型,所不同的是类将数据结构及内部操作作为一个整体进行描述,充分体现了抽象数据类型思想。使用面向对象方法解决问题,首先要确定求解的问题中有哪些对象,对象之间的共性和个性表现为类中的公有成员和私有成员,以及类的从属关系——继承(Inheritance)。类的继承性使得子类可以继承其父类的特征和能力。消息是实现对象之间相互联系和作用的唯一手段,消息传递既可以发生在同一类的不同对象(不同实例)之间,也可以发生在不同类的实例(对象)之间,只有通过消息传递才能激活类中的相应动作,要求某个对象去执行所属类中定义的某个操作,实例是由某一特定类所描述的一个对象。由于上述面向对象方法具有普遍性,因而,类的概念是现代程序设计中的一个有力工具。

可以看出,面向对象的程序设计方法提供的概念——对象,是让程序员自己去定义解释空间对象,首先描述对象及其行为,然后用传统的方法实现。例如,数据堆栈可以作为一个对象,它必须有一个栈体存放数据,可按照后进先出的次序压入或弹出数据。栈空了不能弹出,栈满了不能压入。至于栈体中存放什么类型的数据,并不是堆栈行为的主要特征。面向对象提供一种机制,使得私有数据有其私有操作,即描述这个对象(数据)的独特行为。这样就向着减少语义断层的方向迈进了一步,在某些问题上可以直接模拟问题空间的对象,因而,写出的程序易于理解和维护。从存贮的角度看,对象是一片私有存贮,其中有数据及其操作。其它对象的操作不能直接操作该对象的私有数据,只有对象的私有操作才可以操纵它。从对象实现的机制看,对象是一台自动机,其中私有数据表示了对对象的状态,该状态只能由私有操作来改变。每当需要改变对象的状态时,只能由其它对象向该对象发送消息,对象响应消息后按照消息模式找出匹配的方法(操作),并执行相应的操作。

面向对象的设计方法可分为以下几个步骤:

(1)问题的定义;(1)对象需求分析;(2)对象划分。要求将所要解决的问题以对象为单元进行划分,并进行整体对象的需求分析。

(2)对象设计与定义;(1)对象的检索;(2)对象的定义。进行详细的设计和定义。

(3)对象的联系;(1)对象之间消息的传递;(2)系统的协调。要将各个对象联系起来,使它们形成一个协调的整体。

3 系统程序设计方法与面向对象方法之比较

面向对象的程序设计语言提出和发展是在七、八十年代。例如当时的C++、Smalltalk等,这正是传统语言往模块化、抽象化发展的时期。Smalltalk、C++或多或少也受到了影响,一个典型的例子是Object——C中的操作符重载,在Object——C中没有操作符的重载。另外,分布式数据处理与类有着不可分割的关系,分布式系统有着减少硬件成本和与先进通信设施相连接之优点,更主要是适合人们日常的数据处理方法,分布处理可提高有效性和可靠性,分布式数据处理是通信技术和计算技术的结合,适合于对于大部分处理是局部的,而又存在少量必要的、全局性处理的应用。这样我们就可以把局部操作看成不同的类,通过类之间的消息传递来实现全局性处理,实现了面向对象方法在分布式系统中的应用。可以说,以后过程式语言的发展都是枝节性的了。然而,传统的过程语言与面向对象的语言毕竟有本质的区别,这可以以下几点体现出来:

(1)对象与数据的区别。在传统语言中,要区分变量、常量、数据类型和过程等概念,对它们的处理要分别考虑。而对对象统一了数据和处理,它包含了数据及其对这些数据的处理,是一个能动的整体。

(2)消息传递与子程序调用的区别。在传统语言的子程序调用/返回中,存在着控制与被控制的关系,操作的接口与子程序的内部结构对调用者来说是可见的,并且有调用必有返回,存在严格的因果和次序关系。消息传递则不同,它统一了数据流和控制流。消息中只包含传送者的要求,它告诉接受者需要处理什么,至于如何处理,完全由接受者对消息的解释来决定,发送者对接受者不起任何控制作用,接受者也不必非得有回答消息不可,两者处于平等的地位。消息传递还具有并行性,即多个消息可同时进行传递。另外,发送消息和过程调用是不同的,发送消息只能触发自动机,同样的输入参数可因自动机的状态不同从而得出不同的终态。而调用过程时,只要输入相同,输出总是恒定的。

(3)类与类型的区别。类型只表示(下转56页)

MS-DOS 下点阵汉字的多功能综合处理及实现

四川大学计算机科学系 唐贵川 李志蜀 (成都 610064)

摘要 近年,中文操作系统和汉字处理软件不断更新,其应用已逐步深入到社会各个领域。但是,在某些专用场合它们仍然不能满足需要。为此,本文介绍了在非中文 DOS 下点阵汉字的多功能综合处理及实现,给出了一个创建汉字专有字库的简便方法;设计了适宜不同需要的汉字显示的三种算法;讨论了汉字的无级缩放,汉字的任意方向旋转变换和彩色文体效果的汉字产生等问题;还提到了点阵汉字旋转后的“缺漏”效应,并给出了相应的解决方法。

关键词 汉字显示 点阵 算法 缺漏 字模

1 引言

越来越多的应用软件鉴于节省内存空间,保证程序可靠运行等目的,一般不使用中文 DOS,但是又往往希望在程序运行中,能有少量汉字提示(例如生产流程中的控制信息等)以帮助理解,或是希望在屏幕需要的位置上以给定的方向和适当比例显示某些汉字。因此,在 MS-DOS 下建立专有字库以及实现汉字的多功能综合处理很有必要。

IBM-PC 系列监视器有两种显示模式:文本模式和图形模式。因为汉字是以图形方式呈现,且有比例旋转变换,再考虑到自定义特殊图形符号的可能,这里采用图形模式。

汉字在机器中的表示一般有两种,即矢量表示和点阵表示。矢量表示把汉字转化成能真实反映汉字字型的最小线段集合,点阵表示是以像素为单位来构建汉字笔划的。矢量表示具有存储量小,易于旋转放大等优点,但显示响应速度次于点阵汉字。由于点阵汉字使用面较广,在专用场合中点阵表达易于定位汉字,所以,本文仅就点阵汉字进行讨论。

实用汉字大小不等,位置各异,方向不一,这给汉字显示带来不便。因此,鉴于不同应用需要,本文设计了点阵汉字显示的三种算法,以供选择使用。

2 专有字库的建立

专有字库就是应用软件中所要用到的汉字字模数据集合。针对字模数据的获取方法,专有字库的建立有多种方法,可以先查出专有字库中的每个汉字的国标区位码(或内码),再由区位码(或内码)变换后在中文 DOS 的字库中索引得到字模;也可以将中文 DOS 中汉字字库文件(即 CCDOS 中的 CCLIB,金山软件中的 * .dot)调入内存,算出字库在内存中的起始地址,按国标区位码变换得到某汉字字模的相对偏移,并与字库首址相加,结果就是字模在内

存中的绝对起始地址,自该地址连续读字节若干(对 $n \times n$ 点阵字库,连续读 $n \times n/8$ 个字节,例如对 16×16 点阵,连续读 32 字节)即得字模数据。但是上述方法实现起来不方便,本文拟采用下面的方法。

分析汉字字处理软件 CWS 下的文本文件结构,不难发现该文件存储的是汉字的两字节内码。这促使我们用汉字字处理软件编辑专有字库中的所有汉字,得到专有字库汉字内码文件 INTER.COD,而后仅需编一程序读该文件,根据读取的汉字内码,依照国标区位码到内码的转换协定,计算出汉字字模偏移,在字库文件该偏移处,读取其字模。这样,就自动简便地建立起专有汉字字库文件了。

算法见下图 1,这里以 16×16 点阵字库示例。

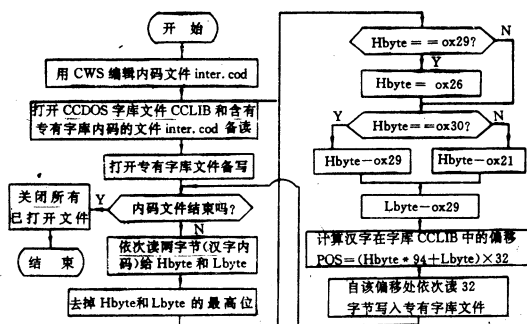


图 1 创建专有汉字字库流程图

专有字库建立后,可简单对每个汉字赋予一个专用内码(如对 m 个汉字的专有字库,可取自然数序列 $1, 2, 3, \dots, m$)。欲读取汉字的字模时,给出其对应的专用内码即可。

3 点阵汉字的多功能综合处理

建立专有字库后,就可以从中提取字模在屏幕上显示某个汉字。例如,在统计图形直方图中,其纵

坐标的标注习惯上就需要将文字旋转 90° 显示。可见,实用中汉字的显示有较大的随意性。

3.1 汉字的显示

这里提供了三种方法,如下所述。

①一般显示方法 这种方法最简单,CCDOS 中汉字显示就是采用这种方法。算法见图 2。

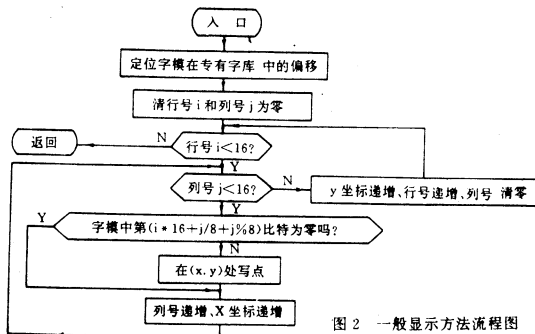


图2 一般显示方法流程图

显示过程入口参数有四个,即显示的汉字左上角屏幕坐标,水平 x 方向和纵向 y 方向的比例因子分别为 dx, dy , 还有汉字的专用内码(本文采用的是该汉字在专有字库中的自然数编号)。显示时按照专用内码索引得到字模数据,自 (x, y) 起,逐点判断字模中相应比特是“1”还是“0”,从而决定在屏幕上写还是不写点。稍作变动,可以旋转汉字九十度,一百八十度和二百七十度。实质上只是变换了写点方向来实施旋转的。

框图2为简明起见,未说明放大处理部分。其实放大处理很简单,即写点时,不只是写一点,而是自该点写一个 $dx * dy$ 点的小方块。

一般显示方法有下面几个特点,可在屏幕上任意 (x, y) 处显示汉字,汉字显示响应速度最快,但是只能提供仅四种显示方向,只能从原字模点阵成倍放大(如对 $16 * 16$ 点阵,可放大为 $32 * 16, 64 * 32$, 即 $(16 * dx) * (16 * dy)$ 类阵列,此处 dx 和 dy 为正整数)。

②任意方向显示方法 针对一般显示方法中的不足,即不能以任意方向(角度)旋转汉字而提出此法。

一般显示法是利用写点方向不同(自上而下,从左至右逐点写一行或自下而上,从左至右逐点写行或从上而下,自左至右写列...)来实施旋转的。要实现任意角度旋转,按下列公式转换即可。

设任意点坐标为

$$(x, y) = (T \cos \varphi, T \sin \varphi)$$

则旋转 α 角后,至新位置 (x', y') , 易知

$$x' = T \cos(\varphi + \alpha) = x \cos \alpha - y \sin \alpha$$

$$y' = T \sin(\varphi + \alpha) = x \sin \alpha + y \cos \alpha$$

变换矩阵为

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

这种显示方法见程序子函数 ListWord(), 较第一种方法能够以任意方向显示,但是显示响应速度不及第一法。尤当指出的是,用该法显示汉字时,当旋转非九十度或一百八十度或二百七十度时,显示出来的汉字笔划中有缺漏,即小的空洞(没有置亮的像素),看起来很不美观,当放大时缺漏现象更甚。我们在这里且称之为缺漏效应,见图3所示。



③改进的显示方法 上已述及汉字显示第二种方法中的缺漏效应。另外,第二种显示法中放大处理只能整数倍放大,不能以任意比例(比例系数为 $(0, +\infty)$)放大。为此,给出了改进的算法。

这种汉字显示方法思想与前者截然不同。假设有两个平面坐标系,一个为源坐标系,一个为目标坐标系。纯按字模显示的汉字在源坐标系中显示,旋转放大处理后的汉字显示在目标坐标系中。我们的出发点是对于目标坐标系中该汉字上的每一点,称目标点,经逆向运算算出该点(一个像素)在源坐标系中该汉字中的对应点,称源点(一个像素)。我们的写点准则即,若该源点(字模中的一个 bit)为“1”则写目标点,否则不写目标点。如此对于目标坐标系中的汉字中的每一点按该写点准则写点,就可完成旋转任意角度,放大任意倍数的汉字显示了。算法见图4。

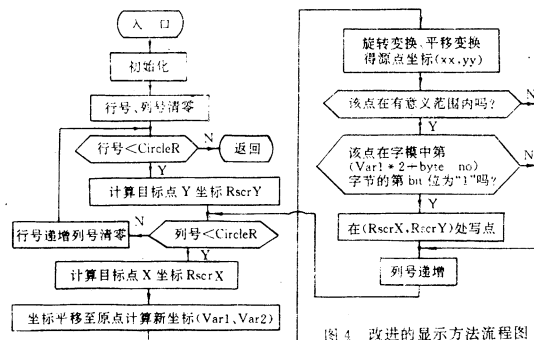


图4 改进的显示方法流程图

需提出,改进算法中放大比例可以无限制。理论上水平方向和纵向的放大系数分别为:

$$X \text{ ratio} \in (0, +\infty), Y \text{ ratio} \in (0, +\infty)$$

实际上由于屏幕不可能无限大,因此放大效果可以是把某个汉字缩小到一个像素点,可以放大到使字模中的一个比特充填整个屏幕,真正实现了无级缩放功能。根据实际需要,使用者可以设比例因子为不同的值,如 0.4, 0.9, 1, 1.5, 5, 10 不等,以满足实际需要。

通过上述改进算法,圆满地解决了汉字旋转缩放后的缺漏效应问题,不论旋转角多大,比例因子多大,显示效果完全理想。但是,由于比例因子采用了浮点数,算法中难免不使用三角函数运算,致使汉字显示响应速度较慢。当然,还可以继续改进算法,使用某些技巧,再配上浮点协处理器就能加快显示响应速度了。

④评价 第一种方法至简,实现起来最容易,显示响应速度最快。当在旋转角度仅为零度(不旋)或九十度,或一百八十度,或二百七十度时,且放大比例只能在字模的基础上整数倍放大时选用第一种显示方法。

第二种显示方法在第一种方法的基础上解决了任意角度旋转问题,但是带来了缺漏效应。当放大比例较小时,显示效果不要求很好的情况下可以用第二种显示方法。

第三种显示方法实现思想与前二法迥异。它圆满地解决了缺漏效应问题,可以无级缩放,可以任意方向(角度)显示。可以满足任意场合需要。但就是显示响应速度最慢。当在第一种方法使用的条件之外的情况下,可以选用该法显示汉字。

3.2 其它

```
#include<graph.h>
#include<dos.h>
#include<stdio.h>
#include<malloc.h>
#include<math.h>
#define PI 3.14
#define CircleR 24
#define WModelL 32
#define DMtrL1 16
#define DMtrL2 8 /*DMtrL1/2*/
#define DMtrL3 15
#define AdjustV 4
/* the size of the special lib */
#define SLBsize WModelL*100
#define L 1
#define R 0
char *WordLib[SLBsize];
FILE *CLIBF;
/*-----*/
ImprovedLibWord(int SorX,int SorY,float XRatio,
float YRatio,double Rangle ,char *WordData)
{
register rr,oo;
int row,col,XYNr,xx,yy,RSorX,RSorY,bytemo;
int Conts1,Conts2,Conts3,Conts4,Conts5;
int var1,var2,pos;
unsigned char bit;
float XYRatio,cosra,sinra;
XYRatio=sqrt(XRatio*XRatio+YRatio*YRatio);
XYNr=(int)XYRatio;
Conts1=SorX+DMtrL2;
```

```
Conts2=SorY+DMtrL2;
Conts3=-DMtrL2-AdjustV;
Conts4=SorX+DMtrL3;
Conts5=SorY+DMtrL3;
switch ((int)Rangle)
{
case 90 :
cosra=0;
sinra=1;
break;
case 180 :
cosra=-1;
sinra=0;
break;
case -90 :
cosra=0;
sinra=-1;
break;
default :
Rangle=Angle/180.0*PI;
cosra=cos(Rangle);
sinra=sin(Rangle);
break;
}
for(row=0;row<CircleR;row++)
{
RSorY=(int)((Conts3+row)*XYRatio)+Conts2;
for(col=0;col<CircleR;col++)
{
RSorX=(int)((Conts4+col)*XYRatio)+Conts1;
for(rr=0;rr<XYNr;rr++)
```

```
for(cc=0;cc<XYNr;cc++)
{
var1=RSorX+Conts1;
var2=RSorY+Conts2;
xx=(int)((cosra*var1+sinra*var2)
/XRatio)+Conts1;
yy=(int)((cosra*var2-sinra*var1)
/YRatio)+Conts2;
if((xx>SorX)&&(xx<Conts4)
&&(yy>SorY)&&(yy<Conts5))
{
bytemo=(xx-SorX)/DMtrL2;
/* bytemo = 0 or 1 */
pos=(xx-SorX)*DMtrL2;
bit=0x80>pos;
var1=yy-SorY;
if((WordData[var1+var1
+bytemo*360] != 0)
_setpixel(RSorX+cc,RSorY+rr);
}
}
}
/*-----*/
void GetModel(char _W_code[],char *WordData)
{
unsigned char lbyte,lbyte;
unsigned long pos;
```

3.2.1 自定义特殊图形符号 鉴于应用程序的特殊性,可能需要一些国标 GB2312—80 中未说明的图形符号。这很容易,只需构建该特殊图形符号的字模,添加到专有字库中,赋予相应专用内码,就可以根据该内码调出字模显示了。

3.2.2 彩色立体汉字的显示 很简单,即在原显示汉字的位置上错开少许像素单位,再次显示另一颜色的同一汉字,就可以看到彩色的带立体效果的汉字了。见图 3 中(d)。

3.2.3 通用性 程序设计中注意到了通用性,采用符号常数定义。要在另一种点阵汉字库下显示汉字,只需修改少数几个符号常量即可。

利用 MSC 提供的一 Setpixel(x,y) 函数,可以在任意图形显示模式下显示汉字。

4 结果

根据以上思想,笔者在 PC 微机系列上用 MSC5.0 成功地实现了专有字库的建立和汉字的多功能处理。总结一下,该系统具有下述特点:

- 快速简便地建立专有图形字库,每个图形字模被赋予一个唯一的专用内码;
- 在屏幕上任意(x,y)处显示汉字;
- 以任意角度(方向)旋转汉字图形;
- 无级缩放的能力;
- 彩色立体效果的汉字显示;
- 适用任意点阵汉字显示的算法;
- 非中文 DOS 下显示汉字;

程序附参考文献之后,以供参考。

5 参考文献

- 1 唐荣锡等,《计算机图形学教程》,1990,4
- 2 金连甫,王译兵,《IBM—PC 图形处理入门》科学技术文献出版社重庆分社,第 1 版,1990,11


```

while((code=NextU8(j++))!=0){
    ListOneWord(x,y,j,(code-1));
    x+=16;
}
y+=18;j=0;
x=50;
getch();

_clearscreen(_GCLEARSSCREEN);
ListOneWord(50,50,3,3,15-1);
ListOneWord(100,150,3,3,15-1);
ListOneWord(10,10,1,1,15-1);
ListOneWord(120,50,10,2,15-1);
ListOneWord(300,50,2,5,15-1);
ListOneWord(80,200,130,6,4,15-1,R);
ListOneWord(80,450,100,6,5,15-1,L);
getch();

_clearscreen(_GCLEARSSCREEN);
ListOneWord(10,50,20,3,15-1);
ListOneWord(80,400,100,20,3,15-1,R);
getch();

_clearscreen(_GCLEARSSCREEN);
ListOneWord(80,200,100,4,4,15-1,R);
ListOneWord(80,450,100,3,3,15-1,L);
getch();

_clearscreen(_GCLEARSSCREEN);
ListOneWord(0,0,39,12,15-1);
ListOneWord(80,20,20,1,1,15-1,L);
getch();

_setvideocode(_DEFAULTNODE);
}

/-----#
GetCMode(Data(char incode[] ,char *data,
FILE *libf)
{
    unsigned char Mbyte,Lbyte;
    unsigned int i;
    unsigned long pos;
    Mbyte=incode[0]340x7f;
    Lbyte=incode[1]340x7f;
    if (Mbyte==0x29) Mbyte=0x26;
    Mbyte=((Mbyte==0x30)?(Mbyte-0x29)<
            (Mbyte-0x21));
    Lbyte=Lbyte-0x21;
    pos=(Mbyte+64*Lbyte)*32L;
    fseek(libf,pos,SEEK_SET);
    for (i=0;i<32;++i) data[i]=fgetc(libf);
}

/-----#

void CreateSpecialLib()
{
    FILE *rp,*wp,*libf;
    char *data=code[3],ch; int m=1,i;
    libf=fopen("occlib","r+b");
    wp=fopen("wordlib","w+b");
    rp=fopen("inter.cod","r+b");
    /* "inter.cod" may be edited by C-Ws etc..*/
    data[(char *)malloc(32*sizeof(char))]

    while (!feof(rp)){
        code[0]=fgetc(rp);
        code[1]=fgetc(rp);
        GetCMode(Data(code,data,libf);
        for(i=0;i<32;++i) fputc(data[i],wp);
    }
    fclose(wp);fclose(rp);fclose(libf);
}

/-----#

main()
{
    char ch;
    while(l!=1){
        _clearscreen(_GCLEARSSCREEN);
        _settextcolor(3);
        _outtext("\n\n\n\n\n\n");
        _outtext("\n _____");
        _outtext("\n ");
        _settextcolor(4);
        _outtext("MENU ");
        _settextcolor(1);
        _outtext("\n _____");
        _outtext("\n 1.Improved rotating example ");
        _outtext("\n 2.Fast rotating example ");
        _outtext("\n 3.Basic listing example ");
        _outtext("\n 4.Create special library ");
        _outtext("\n 5.Exit ");
        _settextcolor(3);
        _outtext("\n _____");
        getch();

        switch(ch){
            case '1': example1();break;
            case '2': example2();break;
            case '3': basicListing();break;
            case '4': CreateSpecialLib();break;
            case '0': return;

```

大型结构多阶模态快速动画显示方法

重庆大学计算中心 王成良 (630044)

摘要 本文采用 MS—FORTRAN 5.0 和宏汇编语言接口的方法将屏幕上所显示的某一阶模态图形通过访问 VRAM, 存贮在指定的磁盘文件中, 然后根据需要, 将文件内容读入 VRAM 中, 以实现模态的快速动画显示。该方法避免了一般方法因多阶模态图形需开辟多块内存区域来实现动画显示, 从而使内存不够的缺点。作为移植到微机上的 SAP5 和 ADINA 有限元分析程序的一个功能模块, 该方法可对大型结构的多阶模态进行快速动画显示。

关键词 动画显示 结构模态 视频缓冲区 VRAM 有限元分析

1 引言

随着 OS/2 操作系统和其它一些操作系统在微机上应用的普及, 原来 DOS 操作系统下解题规模十分有限, 要通过模块覆盖等技术才能在微机上运行的大型结构有限元分析程序如 SAP5, ADINA 等现已能方便地移植到微机上, 并在其上运行, 其解题规模也随着内存容量的增大而大大提高。对一个配置有 8M 内存的 80386/80486 微机, 再加上足够的外存, 其解题规模接近中型机的解题规模。在进行结构动力分析的后处理时, 为了使结构的实际振动更形象化, 避免阅读一大堆枯燥乏味的数据, 一般均采用模态的动画显示。

目前动画显示方法概括起来主要有以下三种:

- (1). 利用多个轮流显示的画面来实现动画;
- (2). 利用交替画擦法来实现动画;
- (3). 利用图形存取法来实现动画。

前两种方法由于其应用的局限性和本身的缺点, 一般仅用于非常简单的动画显示。第三种方法利用图形存取法, 即是利用计算机语言中某些内部函数, 如高级 BASIC 中的 GET 函数, C 语言或 MS—FORTRAN 5.0 中的 GETIMAGE 函数等等, 将屏幕上指定矩形范围内的图形保存在开辟的一块内存区域中, 可用相应的 PUT 函数, PUTIMAGE 等函数在指定位置重新显示所保存的图形。由于存取和显示直接和内存打交道, 因此可获得满意的动画效果。该方法仅适用于图形不复杂, 内存占用量少, 要保存的图幅数十分有限的场合, 当要保存多幅大量占用内存的复杂图形(如大型结构的多阶模态图形)来进行动画程序设计时, 显然该方法已不再适用。基于大型结构有限元分析程序均采用 FORTRAN 语言编写, 考虑到作为它们的一个功能模

块, 我们在 MS—FORTRAN 5.0 中调用汇编语言程序来直接读取屏幕所对应的 VRAM 中的内容, 并写于指定的文件中, 在动画程序设计时, 又可将文件内容恢复于 VRAM 中。该方法直接对 VRAM 进行读写, 因而达到了快速动画显示的效果, 避免了占用内存过多的情况, 且要保存的图幅数量不受限制。

2 图形的保存和恢复

现以 CGA 显示器在 320×200 分辨率图形方式下工作为例来说明如何实现图形的保存和恢复。

视频缓冲区 VRAM 具有 16K 字节的存贮空间, 其内存地址为 B8000H~BC000H, 整幅屏幕有 0, 1, ..., 199 共 200 行扫描线, 偶数序号的扫描行信息依次放于 B800H:0000H~1F40H 处; 奇数序号的扫描行信息依次放于 BA00H:0000H~1F40H 处。显示时, 先显示偶数行内容, 再显示奇数行内容, 且四个象素点占用 VRAM 一个字节。

保存图形时, 即是把 VRAM 中 16K 字节内容写入指定的文件中, 在 FORTRAN 程序中直接调用汇编程序 SAVEW, 格式如下:

```
CALL SAVEW(文件名)
```

恢复图形时, 即是把文件中内容读入 VRAM 中, 在 FORTRAN 程序中直接调用汇编程序 SHOWW, 格式如下:

```
CALL SHOWW(文件名)
```

两个汇编程序分别如程序 1 和程序 2 所示。

需要说明的是上述两个程序中的文件名不超过两个 ASCII 字符, 若指定多个 ASCII 字符为文件名时, 在参数传递部分稍作修改即可。

程序 1: 保存图形程序

```
.model large  
public savew
```

```

data segment public'data'
fln db'',0 ;file name
err1 db'cannot open file$',0ah,0dh
err2 db'not CGA 320×200 mode! $'
data ends
code segment
    assume cs:code,ds:data
sawew proc far
    push bp
    push ds
    mov bp,sp
    mov ax,data
    mov ds,ax
    les bx,[bp+8]
    mov ax,es:[bx]
    mov ds:word ptr fln,ax
    mov ah,3ch ;creat a file
    lea dx,fln
    mov cx,0020h
    int 21h
    jnc lab1
    jmp quit1
lab1,mov si,ax
    mov ah,0fh
    int 10h ;not CGA quit
    cmp al,4h
    je lab2
    jmp quit2
lab2 mov cx,16384; file length
    mov bx,si ;file code name
    push ds
    mov ax,0b800h
    mov ds,ax
    mov dx,0000h
    pop bp
    ret 4
sawew endp
code ends
end

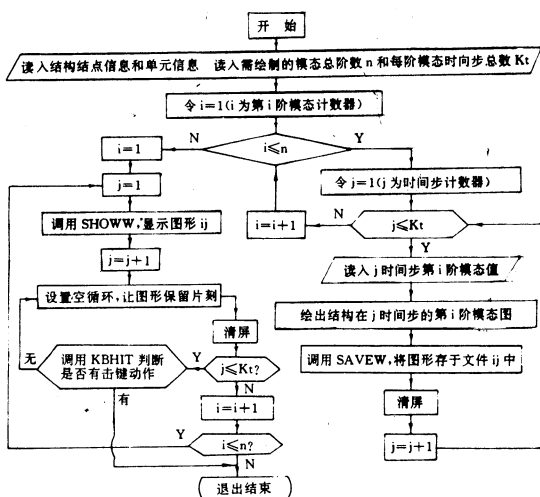
程序 2:恢复图形程序
.model large
public showw
data segment public'data'
fln db' ',0 ;file name
err1 db 'not find file$',0ah,0dh
data ends
code segment
    assume cs:code,ds:data
showw proc far
    push bp
    push ds
    mov bp,sp
    mov ax,data
    mov ds,ax
    les bx,[bp+8]
    mov ax,es:[bx]
    mov ds,word ptr fln,ax
    lea dx,fln ;open file
    mov al,0
    mov ah,3dh
    int 21h
    jnc cont
    jmp quit1
cont:mov bx,ax
    mov cx,16384
    push ds ;write to VRAM
    mov ax,0b800h
    mov ds,ax
    mov dx,0000h
    mov ah,3fh
    mov ah,40h
    int 21h
    pop ds
close:mov ah,3eh ;close file
    mov bx,si
    int 21h
    jmp quit
quit1:mov dx,offset err1
    mov ah,09h
    int 21h
    jmp quit
quit2:mov dx,offset err2
    mov ah,09h
    int 21h
quit:pop ds
    int 21h
    pop ds
    jmp quit
quit1:mov dx,offset err1
    mov ah,09h
    int 21h
    jmp quit2
quit2:pop ds
    pop bp
    ret 4
showw endp
code ends
end

```

3 大型结构多阶模态的动画实现

结构的一般运动是由模态迭加而成的,因此了解每一阶模态图形就可确定结构的振动形态。利用上述的 SAVED 和 SHOWW 子程序,在 MS—FORTRAN5.0 中,就可以实现大型结构多阶模态的动画显示。其程序框图如下。

在动画程序设计时,为方便从循环中退出,我们用汇编语言编制了一个子程序 Kbhit,供 FORTRAN 调用,设置 9 号键盘中断功能来判断有否击键动作,如有击键动作,则退出图形系统。程序 Kbhit 见程序 3。



4 例子与结论

作为该方法的一个示意性例子,下面我们附上一段用 MS—FORTRAN5.0 编写的完整的动画程序,读者可以以此为蓝本,进行多幅复杂图形的动画程序设计。该程序首先绘制了五幅大小不同的直线,填充椭圆图形,并分别存贮在文件 t1, t2, t3, t4, t5 中,然后采用由大到小,由小到大的循环方式显示各图形,获得逼真的动画效果。击任意键将退出图形状态,返回到操作系统下。

从上面的讨论可以看出,虽然该方法是针对大型结构多阶模态图形动画显示而言的,但此方法具有普遍性,可用于其它方面的动画程序设计。当采用 EGA/VGA 图形方式时,其 VRAM 始地为 0A000H,具有 64K 字节的存贮空间,由于保存和读取 VRAM 内容的磁盘文件为 64K 字节,因此,比起 CGA 图形方式 16K 字节的磁盘文件的保存和读取来,其动画显示速度受到一定影响,动画效果不如 CGA 图形方式。需要指出的是,当执行上述动画程序时,可将程序 1、2、3 先编译后,加入到 FORTRAN 库文件中去,作为库函数使用。为保证动画

速度,生成的图形文件应置于硬盘下。

用上述方法,我们编制了 SAPD 和 ADID 动画程序,作为 SAP5 和 ADINA 有限元分析程序的一个功能模块,并已在实际中获得较好的应用。

程序 4: 动画示例程序

```

call kbhit
enddo
INCLUDE 'FGRAPH.FI'
INCLUDE 'FGRAPH.FD'
integer * 2 dummy,
+xl(5)/10,60,110,140,150/,
+y1(5)/10,40,65,85,95/,
+x2(5)/310,260,210,180,170/,
+y2(5)/190,160,135,115,105/
character * 2 fn(8)/'t1','t2','t3',
+ 't4','t5','t4','t3','t2'/
record/xycoord/xy
IF(setvideomode($MRES4COLOR).EQ.0)
+STOP'Error:no color graphics!'
dummy=selectpalette(0)
idummy=setbkcolor($blue)
do i=1,5
dummy=setcolor(1)
call moveto(xl(i),y1(i),xy)
dummy=lineto(x2(i),y1(i))
dummy=setcolor(3)
dummy=ellipse(3,xl(i),y1(i),x2(i),y2(i))
call saved(fn(i))
read(*,*)
call clearscreen($GCLEARSCREEN)
enddo
call setviewport(0,0,319,199)
write(*, '(a\')'enter loop number:'
read(*,*)lp
do i=1,8
call showw(fn(i))
do j=1,lp
enddo
call clearscreen($GVIEWPORT)
goto 1
end

```

程序 3: 键盘中断程序

```

.model large
public kbhit
.code
kbhit proc far
cli
mov ax,3509h
int 21h

```



```
push es
sti
xor ax,ax
mov es,ax
in al,60h
test al,80h
jnz exit
in al,61h
or al,80h
out 61h,al
pop es
mov ah,4ch
int 21h
ret
```

```
exit:pop es
ret
kbhit:endl
end
```

5 参考文献

- 1 朱慧真编著,《汇编语言教程》,国防工业出版社,1988.10
- 2 戴梅萼编著,《微型计算机及其应用》,清华大学出版社,1991.11
- 3 明华译,《MICROSOFT FORTRAN 5.0 高级程序设计》,北京微宏电脑软件研究所
- 4 刘乃奇编著,《IBM PC 混合语言编程技术》,电子工业出版社,1990.9

大型集成式管理信息系统的设计方法

华中理工大学系统工程研究所 王宗军 (武汉 430074)

摘要 一个有效的管理信息系统(MIS),能够及时地为企业决策者提供各种数据及动态信息,从而辅助决策者进行合理决策及生产经营管理。本文以某大型矿务局为例,在调查研究的基础上,提出了大型企业的集成式管理信息系统(IMIS)设计思想,提出了大型IMIS的基本概念和基本结构,并介绍了该大型矿务局IMIS中数据库系统的设计和各个系统模块功能的形成,以及数据库的查询方式和各个系统中模块调用的方法。

关键词 管理信息系统 集成式管理信息系统 数据库 模块 查询

1 引言

信息系统的发展始于电子数据处理(Electronic Data Processing,记为EDP)阶段,经过管理信息系统(Management Information System,记为MIS)阶段,最终形成决策支持系统(Decision Support System,记为DSS)^[1]。在大型企业的生产管理中,通常上述的三种管理层次的信息系统是被有效地结合起来的。EDP负责接受MIS分配给车间、生产队、班组、工段等具体单位的计划任务,并作为MIS的数据站收集生产经营的原始数据;MIS负责接受来自DSS的战略计划数据,如生产计划指标、质量指标、产品结构、价格策略等,完成对各项指标的分解及专项计划之间、专项计划内部的协调平衡关系,并通过EDP下达执行;DSS则负责高层信息处理,与高级计划管理人员组成人——机交互系统,辅助和支持决策者进行经营管理和决策,并通过MIS发布计划决策的内容。从EDP、MIS和DSS这三者之间的关

系中不难看出MIS的重要地位和作用。我国计算机的应用起步较晚,目前MIS的开发尚属初级阶段,而且技术手段也还不成熟^[4]。DSS的开发则基本处于研究尝试阶段。笔者曾参加了国内一大型矿务局计算机管理系统开发的部分工作,并做了较深入的调查,进行了数据收集、整理和建立中心数据库(Central Database,记为CDB)及模块管理等工作,并对生产计划的决策支持做了研究^[2,3]。在开展这些工作的过程中,笔者深深认识到,建立一个有效的MIS,及时地为企业决策者提供所需的各种数据以及动态信息,对决策者进行合理决策、制定出合理的经营计划具有十分重要的意义。本文首先介绍IMIS(Integrated Management Information System)的基本概念、设计原则和基本结构,然后在此基础上讨论数据库系统(Database System,记为DBS)的设计及各子系统模块功能的形成,最后着重讨论数据库(DB)的查询方式及模块调用方式。

2 大型 IMIS 的基本概念、设计原则和基本结构

一个大型企业通常由下属的许多分机构组成,而且分机构中又有小的机构,这样便构成了大系统的递阶结构。在这个递阶结构中,信息可由下往上或由上往下传递。一般底层中具体的数据等信息是由下往上传递的,高层领导的计划指示等是由上往下传递的。此外,一个大型企业所面临的信息和数据又是十分繁多的。因此,如何有效地设计一 MIS,把各类信息和数据集成管理,为决策者创造一个良好的决策环境,便成为当务之急。基于此,笔者便提出了面向大型企业的大型 IMIS 的基本概念、设计原则及其基本结构。

2.1 IMIS 的基本概念 所谓 IMIS 是指针对某一应用领域,以 MIS 的方法和技术为核心,结合与之有关的 EDP 图形软件、数学模型、专家经验、数据库技术、通讯技术等技术方法,从而形成的集信息、管理、咨询于一体的系统。从这一意义上说,IMIS 扩充了一般 MIS 的概念,对大型企业的综合管理与决策更为适用。

2.2 大型 IMIS 的设计方法 笔者结合自己在研究工作中的体会,提出了大型 IMIS 的下述设计原则:

• 根据机构的不同职能划分子系统

笔者认为,根据大型企业主要管理机构的不同职能来划分 IMIS 的子系统,既符合大型企业生产经营管理的各部门自主管理且服从上级指挥的管理机理,又体现了大型程序设计的设计思路。

• 模块化程序设计

我们采用模块化程序设计原则,因而使得各子系统的目标十分清楚,而且各子系统内不同的模型设计成不同的模块,各个模型的输入输出采用规范的方式,由数据库管理系统(Database Management System,记为 DBMS)统一管理。这样,一旦一个子系统完成,就可装入该系统运行,而且各子系统中可以不断加入完成的模块来不断增强其功能;此外,整个 IMIS 可以不断地开发新的子系统同已有系统相集成,以扩大整个系统的功能,而不影响其它子系统的正常运行。

• 数据的一致性和保密性

大型 IMIS 在大型企业中能否发挥应有的作用来辅助决策者进行生产经营决策,在很大程度上取决于数据的一致性和保密性。依据这一原则,在 IMIS 的 CDB 设计中,应当把企业内的数据认真归并,去掉冗余,尤其要注重基础数据即原始数据的一致性和安全性。只有这样,系统的各子系统经模型运行的结果才有其参考价值,才能为决策者提供可信

赖的决策信息和依据。此外,应当采用一定措施,对系统的软件、硬件以及相关的设备甚至工作人员采取保密,只有这样才能使企业决策者信赖整个 IMIS,从而推动 MIS 在大型企业中的应用和发展。

2.3 大型 IMIS 的基本结构 我们通过对该矿务局的调查,详细了解了该矿务局的主要职能机构及其职能。该矿务局下属设有公司、处和矿等平行机构。公司包括运销公司、物资公司等,负责煤炭外运、物资供应与储备等;处有生产处、计划处等,各处职能分工不同,但都服务于该局且彼此有信息共享等联系;各矿则主要生产原煤以完成局下达的任务为目标。公司、处和矿又各有科、生产队等部门。另外,该矿务局还有一总调度室,汇集当天各公司、处和矿的各种情况及数据和信息,直接为局领导服务。根据这种递阶管理机构,我们得出了该矿务局 IMIS 的基本结构(如图 1 所示)。

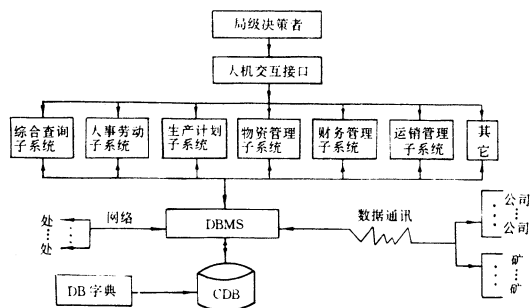


图 1 该矿务局 IMIS 的基本结构

在如图 1 所示的该矿务局 IMIS 中,CDB 中存放着全局的主要数据信息,各处通过计算机网络(该矿务局采用 3+ 网络系统)传送数据,各公司和矿则以数据通讯方式传送数据。

我们在对该矿务局 IMIS 的开发过程中采取的开发策略为:全面了解,整体规划,共同参与,模块集成。对该系统开发的实践表明,采用上述设计原则和开发策略,有助于对系统开发的管理和系统整体功能的不断完善,也有助于决策者及早地利用该系统提供的信息对生产经营进行管理。

3 大型 IMIS 中 DBS 的设计和各子系统模块功能的形成

3.1 DBS 的设计 MIS 是对信息进行管理的系统,它是一种协助决策者解决问题和作出决策的权威方法,它对所有决策人员都提供信息,所以企业一切活动可以紧密联系在一起,使整个企业作为一个系统来经营^[5]。对于一个大型企业来说,主要信息大都是以数据方式表示的,因此,对各类数据进行综合分析和系统整理是建立 MIS 的基础。

该矿务局 IMIS 的 DBS 由 CDB、DBMS 和 DB 字典组成,它是整个 IMIS 的基础。CDB 用来存放结构化的数据并按规定结构进行组织;DBMS 具有定义 DB 结构、操作 DB、数据安全保密和完整一致性控制和管理、维护和恢复 DB 等功能;DB 字典主要存放 DB 的结构信息。

在该矿务局 CDB 的 DATA.DBF 中,存有所有 DB 的字段名,而在 DATA.NDX 中存有相应表格与 DB 的对应关系。这里,我们把不同的表格作为不能的 DB,这也是我们系统设计的一个原则。以表格为单位建立 DB 的好处是便于制表和查询,编制计划汇总统计等也十分方便。

3.2 各子系统功能模块的形成 各子系统功能模块的形成是建立在已有 DBS 基础上的。在该局 IMIS 各子系统中存放了不同的模块,它们是彼此独立的,这样便于模块的添加、更新和删除。如在生产计划子系统中就有对各生产矿的两级生产优化模型:

第一级:全矿生产成本 C 最低

$$\begin{aligned} \min C = \sum_{i=1}^L a_i Y_i \quad \dots\dots\dots (1) \\ \text{s.t.} \begin{cases} Y_i \leq Y_{i\max} \\ \sum_{i=1}^L Y_i \leq Y_{\max} \\ Y_i \geq 0 (i=1, 2, \dots, L) \end{cases} \end{aligned}$$

其中, L 为水平数, a_i 为第 i 水平吨煤生产成本, Y_i 为第 i 水平原煤最优产量(由下一级决定), $Y_{i\max}$ 为第 i 水平限定生产能力, Y_{\max} 为全矿年最大生产能力。

第二级:各水平原煤产量最大

$$\begin{aligned} \max Y_k = \sum_{j=1}^m b_j X_j (k=1, 2, \dots, L) \quad \dots (2) \\ \text{s.t.} \begin{cases} (X_j \leq C_j) \\ X_j \geq 0 (j=1, 2, \dots, m) \end{cases} \end{aligned}$$

其中, m 表示约束条件个数, X_j 对 $j=1, 2, \dots, m$ 分别表示企业坑木约束、火药约束、钢材约束、电力约束、人员约束、资金约束、井下运输能力约束、通风约束等多种约束变量, b_j 为第 j 个约束条件对原煤产量的影响系数, C_j 分别表示上述 X_j 的限制值, Y_k 表示第 k 水平原煤产量。

采用线性规划方法对式(1)、(2)进行优化分析,可以得出生产成本和原煤产量的最优值,而且对不同的煤矿只需改变不同数据即可。此外,还可以进行灵敏度分析,找出生产成本的变动范围以及各水平原煤产量的变动区间,从而为决策者合理安排各水

平产量提供有力的帮助。一系列数学模型的建立,大大增强了各子系统的功能,也促进了 MIS 的发展。

4 大型 IMIS 中 CDB 的查询方式和模块调用方式

4.1 CDB 的查询方式

CDB 的查询方式采用下述两种:

· 菜单驱动方式

这是大部分 MIS 所采用的查询方式,优点是操作直观简单,缺点是对一个较大的系统的查询往往要进行多次的菜单选择,因而需要时间较多。决策者应用此方式查询时,只需按菜单提示选择即可,如选择查询目标的类型——>查询目标的形式(表格/数据项/文字说明/图形显示等)——>确定最后查询目标。

· 命令驱动方式

该方式的优点是它适宜从大量的查询目标中找出合适的项,缺点是要求用户输入较多,而且一旦输错的话会耽误查询时间。根据对该矿务局的了解,我们规定查询命令的形式为:

[查询单位][查询目标][查询时间]。决策者应用此方式查询时,首先要依次输入查询命令的三个组成部分的内容,最后确定查询目标的形式(表格/数据项/文字说明/图形显示等)。

查询目标或查询内容,由若干关键字构成,在用户回答时,以设置功能键的方式在提示窗口中显示出来。该矿务局的查询目标关键字按局级管理职能分为生产类、安全类、机电类、地质类、基建类等。此外,关于物资、生产、运销、财务、人员等情况,可以直接以表格形式查找。

4.2 模块调用方式 系统将用户输入的查询目标集转换为系统内部编码,再根据反映模块间逻辑关系的规则库,来确定模块的调用顺序。这样做一是为了提高模块组合的灵活性,二是为以后分析模块的加入作准备。每个规范的模块包括:模块名、编号、分类、功能描述、输入/输出接口、模块描述、模块体、预处理模块及查询目标等。

表格与 DB 文件名及其它查询内容都可以作为查询目标,DB 及数据项的关系都要转换成 PROLOG 中的事实,在调用 PROLOG 确定的调用模块及调用顺序后,再进行规范模块中各内容的输入,然后系统给出运行结果。

如果在大型 IMIS 各子系统中加入分析模块(包括参数调整、不确定因子设置、灵敏度分析等),将大大增强系统的功能,为决策者创造更好的决策环境。

(下转 43 页)

下拉弹出窗口式菜单及其实现^{*}

浙江大学 张 华 王志新 葛宜远 路甬祥(杭州 310027)

摘要 工控软件通常由相互独立的人机界面和系统控制功能模块等部分组成,本文针对人机界面程序设计,着重介绍了作者在工控机上实现下拉弹出窗口式菜单采用的方法。由于融洽的界面和良好的人机交互特性,使该系统更加便于推广和普及应用。文中程序全部采用 Turbo C 2.0 编程,并在工控机上调试和运行。

关键词 工控软件 下拉弹出窗口菜单 人机界面

1 概述

菜单是用户与计算机进行信息交流的重要接口,直接影响整个系统的人机交互性、可操作性及系统的推广和应用。所以,采用菜单作为整个系统与用户的界面,研究怎样合理地把系统所有的功能组织成一个经过精心安排的人机界面,提供一个高效、友好而与人更为融洽的操作环境,现在已成为人们关注的问题。

一般来说,菜单大都采用多层次结构,系统向用户提供的各种功能不是直接展示给用户,而是隐含在程序中,屏幕上只显示当前操作功能块提示,用户必须通过对当前窗口内的功能进行选择,方能到达可执行功能模块。当今流行的下拉弹出式菜单,从根本上改变了传统菜单的选择及生成方法,经过合理组织,使得整个系统的功能在屏幕上一目了然,用户在屏幕上可获得最大信息量,具有信息清楚、规范、操作灵活、效率高等特点,而容易为用户所接受。

作者介绍的菜单生成程序是一个通用的集成化应用软件,采用 TURBO C 2.0 编程,能提供一个彩色、汉化、多级下拉弹出窗口式菜单,用户只需将当前窗口的坐标、汉字信息、功能块操作信息送入程序中的相应数组,即可得到要求的下拉弹出式菜单,而不必改动整个程序。使用程序时,操作员只需根据汉字提示,并将光条移至所需功能处,并键入回车键,即可执行相应功能块。

2 工控软件下拉弹出式菜单的特殊要求

由于工控的特殊性,工控机通常不能带硬盘和软驱,而电子磁盘因价格高而不宜多用,再加上一般工控机的内存都不大。这样,就要求工控中的下拉弹出式菜单生成程序程序量尽量小,执行时所占的内存最小,同时,基于国情,无疑要求汉化软件的界面,由此也决定了程序只能采用图形模式。众所周知,工控软件中所显的汉字数是固定的,一般不会超出二百个汉字左右,因此,通常不会采用将 CCDOS 或 UCDS 进驻内存,或将整个汉字库装入电子磁盘的方法。作者通过编写一段程序,并根据要求建立一

个小汉字库。屏幕上所显示的汉字提示符由小汉字库中取出,这样就大大节省了存贮空间。

3 设计原理

3.1 理论依据 下拉弹出窗口是在终端屏幕上划分出的一个矩形区域,可以模拟物理终端的行为。窗口可以被重新显示在终端屏幕上,当然,也可以从终端屏幕上消失。只不过在当前信息消失时,要恢复该窗口上一次的屏幕信息,窗口之间也允许全部或局部覆盖。由此可见,程序实现时要用到画面存贮、重放技术,而势必用到以下功能函数:

① `imagesize(x1,y1,x2,y2)` 返回存放屏幕上一矩形区域所需字节数

`size=imagesize(x1,y1,x2,y2)`

其中 $(x1,y1)$, $(x2,y2)$ 分别是一个待保存的矩形区域信息的左上角和右下角的坐标,由 `size` 可知返回要存放屏幕上的该矩形区域所需字节数。

`buf=malloc(size)`

其中, `malloc(size)` 按照要保存的矩形区所需字节数分配内存, `buf` 是用来存贮矩形区域信息的缓冲区。

② `getimage(x1,y1,x2,y2,buf)` 将指定区域位图保存到内存

其中 $(x1,y1)$, $(x2,y2)$ 及 `buf` 与 ① 相同,该语句的功能是把指定矩形区域内的图形信息保存在 `buf` 缓冲区。

③ `putimage(x1,y1,buf,op)` 将以前保存的位图送回到屏幕

其中, `buf` 是由 `getimage(x1,y1,x2,y2,buf)` 存储的图形信息缓冲区指针 $(x1,y1)$ 是将 `buf` 所指向的内存中的图象拷贝到屏幕上的起始位置, `OP` 是图象以何种方式写到屏幕上,其有效方式有 5 种,但由于下拉弹出窗口式菜单的特点,一般只用 `COPY_PUT` 即复制的方式。

④ 将以前保存的位图送回到屏幕之后,有一个为人们忽视的问题,即释放以前保存的位图所占的内存空间,因为如果内存不释放,位图所占的空间在整个程序中将一直保留下去,在工控机内存很小的

情况下很快就会出现内存扰乱等现象,使程序无法正常运行下去。所以,在每级菜单返回时,都必须使用函数 `free(buf)` 释放内存(其中 `buf` 为位图指针)。

根据数据结构,我们将下拉弹出窗口式菜单抽象为一树的模型,再根据调用关系,把所有结点组织起来成为一棵菜单树。其中,树的根结点为主菜单,树的每一个非终结点为菜单管理、连接模块,树的每个终结点为数据输入以及系统功能的可执行功能模块。程序首先进入主菜单,当键入特定功能热键 `F10`,程序即处于菜单树的第一结点上,根据屏幕上的提示,移动光标键、回车键以及 `ESC` 键,即可实现树的上溯和下推,也就是菜单窗口的下拉弹出以及回退;当进止终结点时,即执行系统功能模块;当回止根结点之后,即提示是否返回操作系统。由此可见,要方便、简洁实现菜单树的上溯和下推,必须建立一个标志堆栈,用来记录已上溯止或已下推过的结点,并通过将已下推过的结点标志压入堆栈,已上溯的结点标志弹出堆栈,来使程序清楚地知道当前所处的结点位置,便于编制程序。

3.2 程序设计 实现下拉弹出窗口式菜单的程序,一般分为键管理程序、窗口生成程序、光条生成程序、菜单连接程序、数据输入程序以及系统功能块执行程序六部分,下面给出了各部分编程方法及程序清单:

3.2.1 键管理程序 `TURBO C 2.0` 提供的键输入函数有 `getch()`、`gets()`、`scanf()`,但这些函数对于键盘上的一些特殊键,例如游标移动控制键、左移箭头、右移箭头及 `F1...F10`、`ALT` 键、`SHIFT` 键和 `CTRL` 键等都无能为力。然而,这些键在程序中往往具有特殊的用途,如何获得这些信息呢?这就需要采用函数 `bioskey(cmd)`,其中 `cmd=0` 时,返回下一个键盘输入的值,若是一般 `ASCII` 码字符,则低 8 位为相应的 `ASCII` 码;若为特殊码时,低 8 位为 0,高 8 位为相应的扩充码。

3.2.2 窗口生成程序 窗口生成即为菜单树的下推,程序应实现将位图缓冲区链指针压入堆栈,读取现行窗口位图并存放于内存,同时生成窗口以及显示相关的汉字信息等功能。

3.2.3 光条生成程序 光条的生成一般都是采用读取位图并保存,随后调用一个光条生成程序的办法来实现的。由于工控机的内存比较小,而应该尽量减少其内存的占用。因此,作者利用光条生成函数来生成一个光条,采用覆盖的方法,即当光条移动后,仍调用该函数,并通过改变入口参数(改变调色板),来改变光条底色、汉字颜色。

3.2.4 菜单连接程序 首次进入菜单程序,第一个结点就是菜单连接模块,用于实现当前窗口中各提示功能的选择,并由此进入下一级菜单。一般由键管理程序,菜单树堆栈指针管理程序等组成。

3.2.5 数据输入程序 数据输入方式有两种,一是由人工通过键盘键入数据,二是由 `A/D` 或其它方式输入数据。其中,后一种方式一般都做在系统功能块中。本文采用前一种方式。

3.2.6 系统功能块执行程序 由系统所需要实现的功能要求组成,一般由用户自行编制程序放置在菜单树的终结点,即可实现调用。

上面叙述了作者开发的生成下拉弹出窗口式菜单的主要程序块(主要源程序清单见文末附 1),只要将这些模块灵活地进行拼集,即可得到所需要的菜单,定能够获得理想的效果。

图 1、图 2 所示分别为作者开发的“塑机综合测控系统”人机界面主菜单及主程序流程图,图 3 给出了数据输入流程。

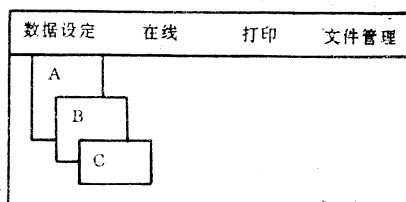


图 1 主菜单界面

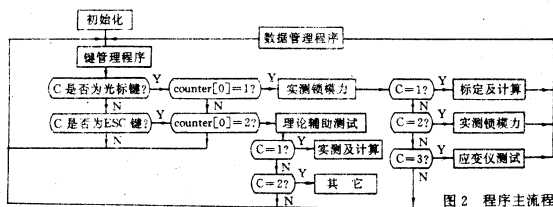


图 2 程序主流程

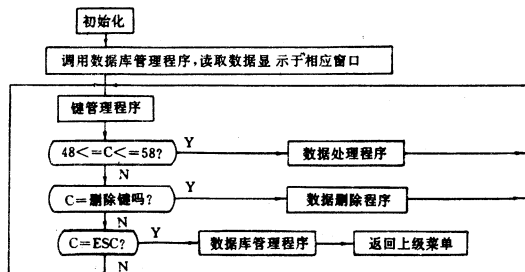


图 3 数据输入流程

4 结论及展望

在作者开发的“注射成型机控制系统”“塑机综合测试系统”“塑料成型专家系统”和“混合搅拌检测系统软件”中,上述方法已得到成功地应用。由于采用了本文介绍的方法,使得系统的人机界面良好,易于操作。即使对系统不了解的人员,也能顺利地按提示进行操作。这一切为系统的推广及应用打下了良好的基础。

在工控行业中,人机界面一直是个非常棘手的问题,因为它一方面与选用的硬件密切相关,但同时

对系统的推广和普及应用起着非常大的作用。怎样协调两者之间的关系,亦即在有限的硬件资源上,编制与人更加接近的人机界面,就成为当今工控软件的一大难题。如今,由于工控机的价格逐年下降,使硬件资源的选择范围越来越大;再加上“把思想变成

程序的工具——雷奥”的推出,这一切都使得即使不懂得程序如何编制的人员,也有可能在这段时间内开发出优秀而实用的人机界面。由此,工控软件人机界面的开发,将越来越向高层次、图形化发展,同时开发周期也将大大缩短。

```

/* ***** 附1 主要函数功能块源程序清单 ***** */
menukey()
{
    int key=0;
    while((c=biockey(0))>=0){
        if((c<=8==0)(c==8)>>8;key=1;goto ahl;);
        if((c<=8)>0)(c==c<8;c==8)>>8;goto ahl;);
    }
    ahl;
    return(key);
}

/* 近邻值 key 为 1, 说明是特殊码, 反之, 就是 ASCII 码, 其值保留在全局变量 C 中。 */
/* 窗口生成程序 */
windowcreat(int x1, int y1, int x2, int y2, int c_outCT, int c_outCT, int c_outCT, int c_outCT)
{
    int i;
    unvo(int x1, int y1, int x2, int y2); /* 保存位图 */
    setfillstyle(SOLID_FILL, CYAN); /* 生成窗口 */
    bar(x1, y1, x2, y2);
    for(i=0; i<=c_outCT; i++){
        c_outCT = x1 + 10 * y1 + 10 * y1 + 20 * c_outCT; c_outCT;
        WHITE, 0, Lib _buffer); /* 显示汉字 */
    }
}

/* 位图读取及保存程序 */
unvo(int x1, int y1, int x2, int y2)
{
    counterbuf++;
    size[counterbuf]=imagesize(x1, y1, x2, y2);
    buf[counterbuf]=malloc(size[counterbuf]); /* 分配位图内存 */
    else printf("ERROR"); /* 内存不够提示了出错信息 */
}

/* 其中, counterbuf 是位图缓冲区指针, 为全局变量。 */
menubw(int x, int y, int c_outCT, int c_outCT, int setcolor1, int setcolor2)
{
    setfillstyle(SOLID_FILL, setcolor1);
    bar(x, y, x+c_outCT, y+16); /* 变量 */
    c_outCT(x, y, c_outCT, c_outCT, setcolor, 0, Lib _buffer); /* 显示汉字 */
}

/* 连接程序清单 */
menu(int x1, int y1, int x2, int y2, int bi, int biend, int c_outno, int c_outof)
{
    int key;
    if(counterbuf!=bi-1)
        windowcreat(x1, y1, x2, y2, c_outCT, c_outCT, c_outCT, c_outCT); /* 生成窗口 */
    counter [bi]=1; /* 菜单树结点下推标志置位 */
    for(key=0; key<=c_outCT; key++){
        c_outCT(x1+10, y1+18+key, 18, c_outno+c_outCT, c_outCT, c_outCT, 0, 0, Lib _buffer); /* 显示窗口内的所有汉字信息 */
        menubw(x, y, c_outCT, c_outCT, BLACK, WHITE); /* 显示光条 */
        begin;
        key=menukey();
        if((key==1)&&(c==72)|(c==80)&&(counterbuf==0)){
            menubw(x, y, c_outCT, c_outCT, WHITE, BLACK); /* 光条置位 */
            if(c==80){
                if(counter[bi]<=c_outCT) counter[bi]++; /* 菜单树下推标志点标志进位 */
                if(counter[bi]<=c_outCT) counter[bi]=1; /* 同级菜单选择项初始计数 */
            }
            if(c==72){
                if(c==counter[bi]>0) counter[bi]--; /* 菜单树下推标志点标志进位 */
                if(counter[bi]<=0) counter[bi]=c_outCT; /* 同级菜单选择项初始计数 */
            }
            menubw(x, y, c_outCT, c_outCT, BLACK, WHITE); /* 显示光条 */
        }
        if(key==0&&(c==CR)){
            if(bi(biend) c[bi]=1; /* 菜单树结点标志置位 */
            goto end;
        }
        if(key==0&&(c==ESC)){
            counter[bi]=0; /* 菜单树上推标志点标志进位 */
            voun(); /* 将以前保存的位图送回屏幕 */
            if(bi==2) c[bi-1]=0; /* 菜单树结点标志置位 */
        }
    }
}

goto end;
}
else goto begin;
end;
}

/* 其中, counter 为菜单树结点标志进位, C 为菜单树结点标志进位。 */
/* 将以前保存的位图送回屏幕 */
voun()
{
    putimage(x[counterbuf], y[counterbuf], buf[counterbuf], COPY_PUT); /* 将以前保存的位图送回屏幕 */
    free(buf[counterbuf]); /* 释放当前位图所占内存 */
    counterbuf--; /* 当前位图缓冲区出栈 */
}

/* 数据输入程序清单 */
keyinput(int x1, int y1, int x2, int y2, int c_outline, int bi, int lin, int outline)
{
    int injecta[4]=(130, 130, 130, 130);
    int injecty[4]=(16, 34, 52, 16);
    char ahlfi[4][10];
    int key, i=0, j=0;
    star;
    for(i=0; i<=8; i++){
        if(ahlfi[i][j]!='\0') break;
        ahlfi[2][i+injecta[lin][j]+i+8, y1+injecty[lin][j], BLACK); /* 显示光标 */
        begin;
        key=menukey();
        if((G(=7)&&(key==0)&&(c==33&&(127)){
            ahlfi[i][j]=c+i+1, ahlfi[i][j]!='\0';
            ahlfi[2][i+injecta[lin][j]+(i-1)+8, y1+injecty[lin][j], WHITE); /* 显示光标 */
            ahlfi[3][i+injecta[lin][j], y1+injecty[lin][j]); /* 字符串置位 */
            outtextxy(x1+injecta[lin][j], y1+injecty[lin][j], ahlfi[i][j]); /* 显示字符串 */
            ahlfi[2][i+injecta[lin][j]+i+8, y1+injecty[lin][j], BLACK); /* 显示光标 */
            /* 数据输入 */
            if(key==1&&(c==BS)){
                if(i==0){
                    if(G(=1), ahlfi[i][j]!='\0';
                }
                ahlfi[2][i+injecta[lin][j]+i+8, y1+injecty[lin][j]);
                ahlfi[2][i+injecta[lin][j]+(i+1)+8, y1+injecty[lin][j], WHITE);
                setcolor(BLACK);
                outtextxy(x1+injecta[lin][j], y1+injecty[lin][j], ahlfi[i][j]);
                ahlfi[2][i+injecta[lin][j]+i+8, y1+injecty[lin][j], BLACK);
            }
            /* 数据删除 */
            if(key==1&&(c==8)|(c==72)|(c==77)|(c==75)){
                ahlfi[2][i+injecta[lin][j]+i+8, y1+injecty[lin][j], WHITE);
            }
            if(C==80|(c==77)){
                if(G(=c_outline) j++)
                    if(i==c_outline) j=0;
                i=0;
            }
            if(c==72)|(c==75){
                if(j==0) j--;
                if(G(=0) j=c_outline-1;
                i=0;
            }
        }
        goto star;
    }
    /* 字符串光标移动 */
    if((key==0)&&(G(=7)&&(c==33&&(127)){
        sound(1500); delay(100); nosound();
    }
    /* 数据输入提醒 */
    if((G(=7)&&(c==33&&(c==93)|(c==97&&(c==127)) goto begin;
    if((key==0&&(c==ESC)){
        voun(); /* 将以前保存的位图送回屏幕 */
        if(bi==2) C[bi-1]=0; /* 菜单树结点标志置位 */
        goto end;
    }
    else goto begin;
    end;
}

```

5 参考文献

- 徐金梧等编译,《Turbo C 使用大全(1.5—2.0)》,北京科海培训中心,1990。
- 张国锋编译,《面向对象的程序设计和 C++ 语言》,北京科海培训中心,1990。
- 《人机接口应用生成程序》,希望电脑公司,1991 年。
- 《把思想变成程序的工具——雷奥》,中国科学技术出版社,1991。

* 浙江省自然科学基金资助项目

一个中文下拉菜单工具的设计与实现

华中理工大学 袁楚明 周祖德 王焱清 (武汉 430074)

摘要 本文介绍了用 TURBO PROLOG 语言设计中文下拉菜单的方法,并给出了完整的源程序,为用户提供了一个强有力的工具。

关键词 菜单 下拉菜单 中文菜单

1 引言

TURBO PROLOG 语言由于具有很强的逻辑推理、知识表达能力,同时它运行速度快,内存需求小,特别适合于在没有复杂的数值计算、数据处理的情况下开发人工智能软件,因而它被众多的微机用户所接受和青睐。

TURBO PROLOG 的工具库 TOOLBOX 提供了丰富多样的菜单工具谓词,为用户开发具有良好人机界面的软件提供了方便。利用这些工具谓词,用户可以开发具有象 TURBO 语言那样优美界面的软件。目前,由于众多用户的英语水平不高,在开发实用软件时往往用汉字显示(即设计中文菜单)。我们知道,汉字在图形模式下才能被显示。然而,在图形模式下用户将无法使用 TOOLBOX 的菜单工具谓词,这给用户带来了极大的不便。我们分析其源程序发现,这些工具谓词是基于窗口谓词,通过在屏幕上开窗实现的。在实践中,我们发现 makewindow 谓词在图形模式下不能自如地使用,这是因为谓词 removewindow 在图形模式下不能删除窗口所致。这样,尽管这些菜单工具谓词在图形模式下能实现汉字显示,但当结束某一菜单项的选择时在屏幕上开的窗口不能被清除。而对于下拉菜单,其边框在汉字系统下显示一些杂难的汉字(这是由于边框的 ASCII 码与汉字的内码重合而引起的),且结束下拉菜单的操作时,下拉菜单仍留在屏幕上,不能恢复屏幕原来的内容。

在开发诸如实用专家系统时,通常要使用汉字显示进行人机对话,这时借助中文 DOS 的支持是必要的。希望电脑公司推出的 UC DOS 汉字系统,不但支持 EGA 和 VGA,实现 25 行汉字显示,与西文兼容,而且具有彩色显示功能,同时其汉字的显示字库可调入扩展内存,系统可以进行裁减,这样大大地节省了内存,特别是 UC DOS 2.0 版本所需内存更少,如果你只需进行汉字显示的话,通过裁减,UC DOS 只占据 10K 左右的内存。下面介绍在 UC DOS 汉字系统的支持下,用 TURBO PROLOG 语言设计中文菜单的方法。

2 中文下拉菜单的设计

要设计一个中文菜单,首先应考虑并解决以下几个问题:

2.1 汉字显示 TURBO PROLOG 的谓词 write 支持汉字显示,但不支持色彩。而其 BGI 谓词 outtext、outtextxy 虽支持色彩,但不支持汉字显示。利用下列方法可实现汉字的彩色显示:

①用谓词 write 将汉字显示在屏幕上;

②用谓词 getimage 保存屏幕上汉字的位图信息;

③用谓词 putimage 通过异或操作将保存的汉字位图信息重显在屏幕上。

例:

```
cursor(0,0),
write("主 菜单"),
getimage(0,0,63,17,Bm),
.....
putimage(240,100,Bm,1),
.....
```

程序中的汉字在中文 WS 下编辑输入。

2.2 菜单项的显示与选择 作为一个工具,在屏幕的有效范围内,对菜单及子菜单的项数应没有限制。用一醒目的颜色显示一条光柱,光柱所在的位置即为欲选择的某一项对应的位置,按箭头键使光柱作相应的移动,按 ENTER 键即作相应的选择。

2.3 屏幕的保存与恢复 菜单显示时会消除屏幕原有的内容;下级菜单显示时会消除上级菜单的部分内容;执行菜单的某一选择项时也会覆盖菜单的部分显示内容。当完成操作或从下级菜单退到上级菜单,或撤消菜单的显示时,必须恢复屏幕原来的内容。这样在菜单显示前或操作前,必须把欲显示的屏幕区域信息保存下来,这通过 getimage 和 putimage 谓词来实现。

2.4 窗口谓词的利用 TURBO PROLOG 语言优良的窗口功能是其它众多语言(如 TURBO C、TURBO PASCAL、TURBO BASIC 等等)所不能媲美的,特别是它能自带编辑器,给用户带来了极大地方便。在图形模式下,由于 removewindow 谓词失

效,因此,窗口谓词不能被自如的利用。但是,如果不利用其优美的窗口功能,那是十分遗憾的。我们可以把窗口作为图形模式下的一个视口,当完成窗口操作后,用 `clearviewport` 清除窗口。这样使窗口在图形模式下得到充分的应用。具体方法如下:

- ①用 `makewindow` 谓词在屏幕上开窗口;
- ②在窗口内进行输入输出操作;
- ③根据窗口的位置,计算该窗口在图形模式下的位置(即左上角和右下角的坐标);
- ④根据窗口的位置设置一视见区。视见区的范围要包含窗口在内;
- ⑤用谓词 `clearviewport` 清除视见区,从而窗口也被清除;
- ⑥通过自定义的谓词 `getmaxview` 返回屏幕默认的最大视见区。

例: `makewindow(7,18,18," ",5,30,3,28),`

```
.....
setviewport(240,60,464,112,0),
clearviewport,
getmaxview,
.....
getmaxview:-
    getmax(Maxx),getmaxy(Maxy),
    setviewport(0,0,Maxx,Maxy,0).
```

在图形模式下利用窗口,可以实现屏幕的滚动显示和汉字的彩色显示。按照这种方法,TOOLBOX的工具谓词如 `makestatus`、`lineinput`、`menu` 等在图形模式下也可以使用。

附录的程序清单是我们用 TURBO PROLOG 语言开发的一个中文下拉菜单工具的源程序。程序说明如下:

①谓词 `pulldown(COLOR,MENULIST)`

实现下拉菜单的调用。其中 `COLOR` 为菜单的颜色属性,而 `MENULIST` 是一个用 `CURTAIN` 函子构造的一张表,它包含每一菜单项的正文;函子 `CURTAIN` 有 3 个参数:

`curtain(X,MENU,SUBMENU)`

其中 `X` 对应于水平菜单的起始坐标,`MENU` 是该菜单项的名,`SUBMENU` 是对应于该菜单的垂直菜单的项表;

②谓词 `pdwkeyaction(KEY,...)` 说明按键动

作;

③谓词 `pdwaction(CH,SUBCH)` 确定菜单选择项的执行动作。它使得可以在程序执行命令时,下拉菜单仍保留在屏幕上(只要屏幕上的内容没有覆盖菜单),`CH` 和 `SUBCH` 分别对应于用户按 `ENTER` 键后实际选中的水平菜单的编号和相应的垂直菜单(下级菜单)的编号;

④数据库谓词 `pdwstate` 用来记录菜单的状态;

⑤数据库谓词 `pullbase(X1,Y1,X2,Y2)` 用来记录垂直菜单的矩形框范围。

读者在使用该工具程序之前必须做以下工作:

- ①应用程序置图形模式;
- ②在应用程序中包含键盘定义;
- ③在应用程序中用本文前述方法保存各菜单及其子菜单项的汉字;各项汉字的字数最好不超过四个;

然后用谓词 `pulldown` 调用即可。

例如:假设应用程序中各菜单及其子菜单项汉字的位图信息保存在变量 `B1`、`B2`、`B3`、`B4`、`B5` 和 `B11`、`B12`、`B13`、`B14`、`B21`、`B22`、`B31`、`B32`、`B33`、`B34`、`B41`、`B42`、`B43`、`B44` 中,那么,在程序的目标段中调用谓词:

```
pulldown(9,[curtain(35,B1,[B11,B12,
    B13,B14]),
    curtain(135,B2,[B21,B22]),
    curtain(235,B3,[B31,B32,
    B33,B34]),
    curtain(335,B4[B41,B42,
    B43,B44]),
    curtain(435,B5,[[]])]).
```

将应用程序生成 EXE 文件,在 UC DOS 汉字系统的支持下运行 EXE 文件即可。

3 结论

本工具程序在 UNIS386 上运行通过,作者利用它产生应用程序的 EXE 文件后,在 IBM PC 的其它兼容机(只要其显示器支持 UC DOS)上均能运行。我们把它用于开发故障诊断实用专家系统,效果良好。因此可以说,该工具绝对实用可靠,对于欲开发汉化软件的 TURBO PROLOG 用户是一个强有力的实用工具。

附录: 程序清单

```

domains
  MENUITEM=CURTAIN (INTEGER, STRING, STRINGLIST)
  MENULIST=MENUELEM*
  STOP=stop () : cont ()
database=pullbase
pdwstate (INTEGER, SYMBOL, INTEGER, INTEGER, INTEGER)
pullbar (INTEGER, INTEGER, INTEGER, INTEGER)
predicates
  pulldown (INTEGER, MENULIST)
  pdwkeyaction (KEY, INTEGER, SYMBOL, INTEGER, INTEGER, STRING, MENULIST, STOP)
  pdwaction (INTEGER, INTEGER)
  writelist (MENULIST)
  getfirstpos (MENULIST, INTEGER)
  changedwstate (PULLBASE)
  changedbar (PULLBASE)
  listlen (STRINGLIST, INTEGER)
  listlen (MENULIST, INTEGER)
  index (INTEGER, MENULIST, MENUELEM)
  getpos (MENUELEM, INTEGER)
  getstep (MENULIST, INTEGER, INTEGER, INTEGER, INTEGER)
  check_ar (INTEGER, INTEGER, INTEGER, INTEGER, INTEGER, INTEGER, INTEGER)
  check_af (INTEGER, INTEGER, INTEGER, INTEGER, INTEGER, INTEGER, INTEGER)
  morebar (INTEGER, INTEGER, STRING)
  getpdwlist (INTEGER, MENUELEM, STRINGLIST)
  writemenu (INTEGER, INTEGER, STRING, MENULIST, STRINGLIST, INTEGER)
  writemenu1 (INTEGER, INTEGER, STRING, STRINGLIST, INTEGER)
  writemenu2 (INTEGER, INTEGER, STRING, MENULIST, INTEGER)
  writewlist (INTEGER, INTEGER, STRINGLIST)
  check_xyl (INTEGER, INTEGER, INTEGER, INTEGER)
  check_xy2 (INTEGER, INTEGER, INTEGER, INTEGER)
  findlen (INTEGER, INTEGER, MENULIST, STRINGLIST, INTEGER)
  movevbar (INTEGER, INTEGER, INTEGER, STRING)
  clearpull
  clauses
    writelist ([]) :- !.
    writelist ([certain (X, Word, _), T]) :- putimage (X, Word, _), writelist (T).
    getfirstpos ([certain (X, _), _], X).
    changedwstate (X) :- retract (pdwstate (X, _), pullbase), fail.
    changedwstate (T) :- asserta (T, pullbase).
    changedbar (X) :- retract (pullbar (X, _), pullbase), fail.
    changedbar (M) :- asserta (M, pullbase).
    listlen ([], 0).
    listlen ([_], T), N :- listlen (T, N), N+X.
    index ([_], H), H :- !.
    index ([_], T), Element :- N1+1, index (N1, T, Element).
    getpos (certain (POS, _), POS) :- !.
    getstep (List, X), Xn, L, Step :-
      listlen (List, L), index (L, List, C),
      index (2, List, C), index (L, List, Cn),
      getpos (C, X), getpos (C2, X2),
      getpos (Cn, Xn), Step=X2-X1.
    check_ar (X, Xn, Step, CH, X, CH) :-
      X=Xn, X=Xn, Step, CH=CH.
    check_ar (X, X1, Xn, Step, CH, CH) :-
      X=Xn, X=Xn, Step, CH=CH.
    check_af (X, X1, Xn, Step, CH, CH) :-
      X=X1, X=X1, Step, CH=CH.
    check_af (X, X1, Xn, Step, CH, CH) :-
      X=X1, X=X1, Step, CH=CH.

```

```

movevbar (X, X, Cur_bar) :-
  putimage (X, 4, Cur_bar, 1),
  putimage (XX, 4, Cur_bar, 1).
getpdwlist (X, certain (X, _), List) :- !.
writelist ([_], T) :- !.
writelist (X, Y, H) :-
  putimage (X, Y, H, 1),
  writelist (X, Y, T).
check_xyl (Y1, N, Y, Y1) :-
  Y=31, (N-1)+19, Y-Yn, 1,
  Y-Y1+19, Y-Y1+19.
check_xy2 (Y1, N, Y, Y1) :-
  Y=31, (N-1)+19, Y-Yn, 1,
  Y=31, Y-Y+19.
check_xy2 (Y1, N, Y, Y1) :-
  Y=31, (N-1)+19, Y-Yn, 1,
  Y=31, Y-Y+19.
findlen (X, CH, List, List, N) :-
  index (CH, List, CM), getpdwlist (X, CM, List),
  listlen (List, N).
movevbar (X, Y1, Y, Cur_bar) :-
  Xp=X+10,
  putimage (Xp, Y1, Cur_bar, 1),
  putimage (Xp, Y, Cur_bar, 1).
clearpull :-
  pullbar (X, Y, _), !.
  getmaxx (Maxx), getmaxy (Maxy),
  setviewport (X, Y, Maxx, Maxy, 1),
  clearviewport,
  setviewport (0, 0, Maxx, Maxy, 0).
writemenu (X, CH, Cur_bar, List, N) :-
  N>0, !.
  writemenu1 (X, CH, Cur_bar, List, N),
  writemenu2 (X, CH, Cur_bar, List, N),
  writemenu1 (X, CH, Cur_bar, List, N) :-
  X=X+83, Y1+31, Yn=Y1+(N-1)+19,
  Yb=Y+27, Xr=X+5, Yb=Yb-5,
  X2=Xr-5, bar (X, 27, Xr, Yb),
  rectangle (Xr, 27, X2, Yb),
  Xp=X+10, Y22=Y1+19,
  writelist (Xp, Y1, List),
  putimage (Xp, Y1, Cur_bar, 1),
  changedwstate (pdwstate (X, down, Y1, Y22, CH)),
  changedbar (pullbar (X, 27, Xr, Yb)).
writemenu2 (X, CH, Cur_bar, List, N) :-
  changedwstate (pdwstate (X, down, 0, 0, CH)),
  changedbar (pullbar (X, 27, X, 27)).
pulldown (Color, List) :-
  setfillstyle (1, 12), bar (0, 0, 63, 17),
  getimage (0, 0, 63, 17, Cur_bar),
  setfillstyle (1, Color), getmaxx (Maxx),
  bar (0, 0, Maxx, 26), writelist (List),
  getfirstpos (List, X),
  putimage (X, 4, Cur_bar, 1),
  changedwstate (pdwstate (X, up, 0, 0, 1)).
repeat,
  pdwstate (X, DOWN, Y1, Y2, CH),
  readykey (KEY),
  pdwkeyaction (KEY, X, DOWN, Y1, Y2, CH, Cur_bar, List, CONTINUE,
  CONTINUE=stop, !.
pdwkeyaction (right, X, up, Y1, Y2, CH, Cur_bar, List, cont) :-
  getstep (List, X), Xn, L, Step,
  check_ar (X, X1, Xn, Step, CH, L, XX, CH),
  morebar (X, XX, Cur_bar),
  changedwstate (pdwstate (XX, up, Y1, Y2, CH)).
pdwkeyaction (left, X, up, Y1, Y2, CH, Cur_bar, List, cont) :-
  getstep (List, X), Xn, L, Step,
  check_af (X, X1, Xn, Step, CH, L, XX, CH),
  morebar (X, XX, Cur_bar),
  changedwstate (pdwstate (XX, up, Y1, Y2, CH)).
pdwkeyaction (cr, X, up, Y1, Y2, CH, Cur_bar, List, cont) :-
  findlen (X, CH, List, List), N, N>0, !,
  writemenu1 (X, CH, Cur_bar, List), N,
  SUBCH=0,
  not (pdwaction (CH, SUBCH)).
pdwkeyaction (down, X, down, Y1, Y2, CH, Cur_bar, List, cont) :-
  findlen (X, CH, List, List), N,
  N>0, N>1, !,
  check_xyl (Y1, N, Y, Y1), !,
  movevbar (X, Y1, Y, Cur_bar),
  changedwstate (pdwstate (X, down, Y, Y1, CH)).
pdwkeyaction (up, X, down, Y1, Y2, CH, Cur_bar, List, cont) :-
  findlen (X, CH, List, List), N,
  N>0, N>1, !,
  check_xy2 (Y1, N, Y, Y1), !,
  movevbar (X, Y1, Y, Cur_bar),
  changedwstate (pdwstate (X, down, Y, Y1, CH)).
pdwkeyaction (right, X, down, Y1, Y2, CH, Cur_bar, List, cont) :-
  getstep (List, X), Xn, L, Step,
  check_ar (X, X1, Xn, Step, CH, L, XX, CH),
  morebar (X, XX, Cur_bar),
  clearpull,
  findlen (XX, CH), List, List, N,
  writemenu (XX, CH, Cur_bar, List, List), N.
pdwkeyaction (left, X, down, Y1, Y2, CH, Cur_bar, List, cont) :-
  getstep (List, X), Xn, L, Step,
  check_af (X, X1, Xn, Step, CH, L, XX, CH),
  morebar (X, XX, Cur_bar),
  clearpull,
  findlen (XX, CH), List, List, N,
  writemenu (XX, CH, Cur_bar, List, List), N.
pdwkeyaction (cr, X, down, Y1, Y2, CH, Cur_bar, List, cont) :-
  Y2>0, !, SUBCH=round ((Y2-31)/19),
  not (pdwaction (CH, SUBCH)).
pdwkeyaction (cr, X, down, Y1, Y2, CH, Cur_bar, List, cont) :-
  Y2>0, !, SUBCH=round ((Y2-31)/19),
  not (pdwaction (CH, SUBCH)).
pdwkeyaction (cr, X, down, Y1, Y2, CH, Cur_bar, List, cont) :-
  clearpull, retractall, pullbase,
  changedwstate (pdwstate (X, up, 0, 0, CH)).

```

4 参考文献

- 1 辛达雅,《最新 TURBO PROLOG 使用大全》,中科院希望电脑技术公司,1990,11

- 2 潘金贵等,《TURBO PROLOG 工具库》,南京大学出版社,1988,6

并排打印及分页打印程序

本人用 C 语言在 IBM-PC 机上编制了一个可在宽行打印纸上进行并排打印及分页打印的程序,可在一页宽行打印纸上打印 60 行程序,每行可并排打印两个语句行,每个语句行限定在 67 个字符内,每一语句行字符若超过 67 个,则自动打印到下一行,并使得一页宽行打印纸上左右两边的语句行各自连续,而不是两个连续的语句行被分割在左右两边,从而便于装订成册,节约纸张,阅读方便。本程序可在所有各种类型宽行打印机上打印各种语言的程序清单,效果甚理想。需此程序者可同本人联系(陈金友,通讯地址:兰州市 27 支局 10 号信箱 75 号,邮码:732750。)

图文并茂的立体投影式窗口汉字下拉式菜单的设计

黑龙江八一农垦大学 陈廷槐 董玉坤 张胜利 唐德江 赵春霞(密山 158308)

美观的窗口菜单是软件设计者追求的主要目标之一,因为这将给软件增添光彩。目前,一些国外引进的商品软件,如:Pctools 7.0、PW等,都在下拉式菜单的基础上增加了窗口的立体投影效果,窗口看上去犹如悬挂在空中,十分漂亮。有的还在背景上点缀了一些图纹,使用户有一种美的感受,增强了用户界面的友好性。

有关汉字窗口菜单的设计杂志上发表过许多文章,C语言被认为是一种行之有效的工具语言,作者用 Turbo C 实现了立体投影窗口中文下拉式菜单,可在 VGA、EGA 下与 WPS、UCDOS、王码及 2.13 兼容。现将实现的全过程介绍如下,并给出了演示程序,在演示程序中设计了一幅山水画作为背景,使窗口犹如悬挂在山水间,非常新颖。

1 投影窗口的实现

立体投影窗口的原理很简单,即设定两个不同颜色的屏幕区域,将其错位重叠即可产生立体效果,这在 Turbo C 2.0 中很容易实现,用下面的代码即可产生立体窗口:

```
textbackground(color1);
window(x1-1,y+1,x2-1,y2+1);
clrscr();
textbackground(color2);
window(x1,y1,x2,y2);
clrscr();
```

其中, textbackground(color1) 为设置窗口的投影颜色, textbackground(color2) 为设置窗口的颜色, window(x1,y1,x2,y2) 为 Turbo C 提供的文本窗口函数, X1 和 Y1 为窗口的左上角座标, X2 和 Y2 为窗口的右下角座标, clrscr 为清屏函数, 在此用于使窗口颜色得以显示。

2 汉字窗口输出

在 Turbo C 语言中, 汉字窗口输出是下拉式菜单设计的难点之一。在 Turbo C 语言中, 许多显示函数如 cputs(), cprintf() 等, 为追求快速显示效果, 通常直接将字符串送到屏幕显示缓冲区的方法显示字符, 这就无法直接利用这些函数向屏幕输出汉字。一些作者曾介绍在图形方式下显示汉字的方法, 即从汉字磁盘字库中或从汉字系统的内存字库中取汉字字模, 再用画点阵的方法向屏幕输出汉字, 在图形状

态下用这种方法建立窗口菜单时, 可以丰富窗口背景颜色, 另外在磁盘汉字库直接取字模可以节约内存。但在磁盘取字模速度较慢, 从内存中取字模则兼容性较差。Turbo C 提供了一个全局变量 directvideo, 如将其设置为零。即 directvideo=0 (缺省为 directvideo=1), 则在一些系统下 (如 UCDOS) 可用 cputs(), cprintf() 等函数直接向窗口输出汉字。但对大多数汉字系统 (如 2.13、UCDOS 等) 则还不能向窗口输出汉字, 如将屏幕设置为图形方式, 再将全局变量 directvideo 设置为零 (directvideo=0) 即可在多种汉字系统下, 用 cputs(), cprintf() 等函数向窗口输出汉字。

利用 Turbo C 提供了图形驱动函数 initgraph(), 可很方便地将屏幕设置成图形方式, 下面的代码将使屏幕自动设置成高分辨率 (假定图形驱动程序在当前目录下)

```
int g_driver=DETECT, g_mode;
initgraph(&g_driver, &g_mode, " ");
```

将屏幕设置成图形方式后, 再将全局变量 directvideo 设置成零 (directvideo=0), 即可在 window(x1,y1,x2,y2) 设置的窗口中输出汉字。

3 屏幕背景设置

由于将屏幕设置成了图形方式, 所以可用下述代码设置屏幕背景颜色 textbackground(color);

```
clrscr();
```

也可用 setbkcolor(color) 来设置背景。在演示程序中, 使用 fractal 方法产生一幅风景画作为背景, 使窗口的立体效果更加突出 (程序附后)。

4 关于演示程序的说明

本文的演示程序提供了完整的汉字立体窗口下拉式菜单的实现过程, 稍加修改即可成各种实用菜单。背景为一幅风景画。与取字模方法比较, 有代码小、兼容性好的特点。在 VGA 上可与多种汉字系统兼容, 在 EGA 下也可运行, 但在 EGA 下使用时, 需将函数 draw—fractal() 中的相应参数加以修改, 否则背景的山水画不能显示。但对窗口菜单无影响; 另外, 如在 2.13 下运行时, 需先将光标关闭, 否则由于光标的影响, 在汉字的下方将出现黑色的下划线。此外, 用本文的方法, 在 2.13 系统下, 在 CGA 上也可向窗口输出汉字。

```

// WxWidgets example c++
#include <wx/wx.h>
#include <graphics.h>
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#include <process.h>
#include <math.h>
#include <iostream>
#define MAXLEVEL 1000
#define WATERLEVEL 8
#define SUN 8
#define SKY 3
void tahoma();
double fractal(MAXLEVEL);
void subdiv0();
void fractal0();
void draw_fractal0();
int select();
char *name[3]={"显示","退出","再见"};
char *name[3]={"您好","感谢使用本程序","再见"};
main()
{
    int x=0;
    int driver=DTECT_0_mode;
    initgraph(&g_driver,&g_mode,"");
    directmode=0;
    textcolor(RED);
    gotoxy(15,10);
    cputs(name[0]);
    sleep(1);
    clrscr();
    textbackground(BROWN);
    window(15,10,45,11);
    clrscr();
    textbackground(S);
    window(14,11,44,17);
    clrscr();
    textbackground(I);
    window(15,11,45,18);
    clrscr();
    textcolor(I);
    textcolor(O);
    cputs(name[1]);
    textbackground(S);
    window(17,13,43,13);
    textcolor(O);
}

gotoxy(4,1);
cputs(name[0]);
sleep(select(a));
if(select(a))
    textbackground(BROWN);
    window(16,10,44,10);
    clrscr();
    window(16,11,44,17);
    clrscr();
    textcolor(I);
    gotoxy(4,3);
    if(a==0)
        cputs(name[1]);
    else
        cputs(name[2]);
    int sleep(1);
    clrscr();
    exit(0);
}

int select()
{
    int i;
    key;
    while(key!=p)
    {
        key=i+get_key0();
        if(i==key,0)
            switch(key,c[i])
            {
                case S:
                    textbackground(RUID);
                    window(17,13,p,43,13+p);
                    clrscr();
                    textcolor(I);
                    gotoxy(4,1);
                    cputs(name[p]);
                    if(p<1)p++;
                    else
                        p=0;
                    textbackground(GREEN);
                    window(17,13,p,43,13+p);
                    break;
            }
        else
            switch(tolower(key,c[0]))
            {
                case I:
                    p++;
                    return false;
            }
        while(key!=c[0]==I);
    }
    get_key0();
    union REGS r;
    r.h.ax=0;
    return int86(0x16,&r,&r);
}

void tahoma()
{
    rectangle(0,0,getmaxx(),getmaxy()-40);
    setcolor(SKY);
    fractal(0,0,MAXLEVEL,0.5,80,10);
    draw_fractal0();
    setfillstyle(SOLID_FILL_SKY);
    floodfill(1,1,SKY);
    setcolor(WHITE);
    fractal(250,MAXLEVEL,0.7,30,6,14);
    draw_fractal0();
}

setfillstyle(SOLID_FILL_WATER);
floodfill(1, getmaxy()-40, WHITE);
setfillstyle(SOLID_FILL_WHITE);
setcolor(WHITE);
circle(getmaxx()-100,80,SUN);
floodfill(getmaxx()-100,80,WHITE);
}

void fractal(int y1,int y2,int maxlevel,double h,double scale,int num)
{
    int first,last;
    double ratio,std;
    first=0;
    last=(int)(pow(2,(double)maxlevel));
    fract(ffirst,y2);
    fract(last,y2);
    ratio=1./O/pow(2,(0,h));
    std=sqrt(scale);
    subdiv0(first,last,std,ratio,num);
}

void subdiv0(int p1,int p2,double std,double ratio,int num)
{
    int midpt;
    double scmid;
    midpt=(p1+p2)/2;
    if(midpt==p1)&&midpt==p2)
        fract(ffirst,y2)-fract(flast,y2)/(2*(double)(num-8)/8.0*std);
    std*=sqrt(ratio);
    subdiv0(p1,midpt,scmid,ratio,num);
    subdiv0(midpt,p2,scmid,ratio,num);
}

void draw_fractal(void)
{
    int i,n,sinc;
    l=(int)(pow(2,(double)MAXLEVEL);
    sinc=getmaxx()/l+3;
    moveTo(0,0);
    for(i=0,x=0;i<l;i++,x+=sinc)
        lineto((int)fract(i));
}

```

为软件配置良好的用户界面

甘肃省计算中心 午锁平 (兰州 730030)

摘要 本文介绍了用 TURBO—PASCAL 4.0 编写汉字放大以及多彩色下拉式菜单屏界面的实现过程。

1 开发背景

软件用户界面的好坏给人们的印象非常重要，使用的方便程度也直接影响着软件的推广应用。尤其是在人们已经熟悉了如 windows、PC—Tools、WPS 等一系列五颜六色的屏幕显示与多窗口设计；移动上、下、左、右箭头连同光标条一起滚动；按下〈ENTER〉键就选中了处理项目的优秀软件后，对软件的用户界面要求也随之提高了。笔者用 PASCAL 语言对最近开发研制的普通微机图像处理系统软件也试着加了一个用户界面。以使得软件菜单在屏幕上显得丰富多彩；操作起来也随心所欲。效果甚佳。

2 功能介绍

在菜单屏幕上采用汉字放大技术,用湖绿色大字将软件名称打在屏幕上,起到点题醒目的目的。用深蓝色色带条米黄色字标出版本和日期,开发单位等。主菜单用白框围起,用红底黄字衬托起菜单目录。用户只需按下(↑)(↓)箭头即可选择项目。选择时随着箭头的移动彩色色带也一并移动。若选中时色带底色为深蓝色,字样为棕色,未选中的字条带

均为底色大红色,字样米黄色,非常醒目。若要执行按下<ENTER>键即可。

3 实现方法

3.1 使用 TURBO—PASCAL 语言的 graph 单元, 将 16×16 或 24×24 的字模点阵先读出存盘(在进入编译前先在汉字系统下将所需的汉字标题输入在一个变量单元去), 用 write 和 GETpixel 过程实现。见过程程序 graphhz{汉字字模存盘}。本过程仅使用一次即可。

3.2 用 Read 和 putpixel 取出并显示汉字字模(在中西文系统下均可)用 selfillstyle 和 bar 过程可实现放大字样。见过程 DISPHZ{屏幕汉字显示与放大}。

3.3 选用 `<↑>` `<↓>` `<→>` `<←>` `<cr>` 等键编制下拉式菜单,再采用覆盖汉字字条的技术使之光条带跟着上下箭头的移动而移动。见过程 `handlefunkey`、`setxy`、`condition`。

3.4 选用 Gotoxy 和 setxy 过程进行光标定位处理,用 Readkey 过程来接收字符。

4 程序实例清单

[illegible]

2 夏国华,“大型汉字配上倒影的初步试验”,《计算机应用研究》,No. 2. 1990

山东矿业学院经济研究所 赵晓东 (泰安市 271019)

关键词 保存 恢复 页面 图形控制器 定序器 输出寄存器

存贮 EGA/VGA 屏幕上的图形汉字,关键在于

掌握好对图形控制器寄存器(口地址为 3CEH 和 3CFH)的正确控制,由图形控制器寄存器决定 4 个页面中哪一个页面读出的数据送至 CPU。图形控制器寄存器见表一。

表一 图形控制器寄存器

序号(3CE)	图形控制器寄存器(3CF)
00	设置/重置寄存器
01	设置/重置允许寄存器
02	颜色比较寄存器
03	数据移位与功能选择
04	读出页面选择寄存器
05	模式寄存器
06	混合寄存器
07	颜色忽略寄存器
08	位屏蔽寄存器

EGA/VGA 屏幕图形汉字的保存,寄存器的访问可分为两步:第一步是索引(索引号 4),第二步是数据寄存器数据,具体算法见程序一。

2.2 EGA/VGA 的屏幕恢复

对于恢复 EGA/VGA 屏幕上的图形汉字,关键在于掌握好定序器输出寄存器(口地址为 3C4H 和 3C5H)的正确控制,由它来决定向哪一个页面写入数据。定序器输出寄存器见表二。

表二 定序器输出寄存器

序号(3C4)	定序器寄存器(3C5)
00	重置寄存器
01	时钟模式
02	彩色页面写允许
03	字符发生器选择
04	存储模式

EGA/VGA 屏幕图形的汉字恢复,其步骤同保存类似,也分为两步。第一步是索引(索引号 2),第二步是数据寄存器数据,具体算法见程序二。

3 程序的几点说明

(1) 程序针对的是 VGA 下的模式 12H(640×480, 16 色),对于 EGA 的模式 10H(640×350, 16 色),可把程序中的 mov cx,38400d 改为 mov cx,28000d 即可。

(2) FoxBASE+ 和 dBASE 调用汇编语言时,DS 和 BX 寄存器存放由 WITH 选项传递参数的第一个字节。

(3) savescr.asm 产生一个临时文件,将视频缓冲区的数据写入临时文件。

(4) restscr.asm 是 savescr.asm 的逆过程,即将临时文件中的数据写回视频缓冲区中相应的页面。

(5) 临时文件最好放在虚拟盘中,这时屏幕的保存和恢复是相当快的,完全达到实际应用的要求。

(6) 该程序尚需进一步考虑的是临时文件的压缩

存贮和还原显示,以进一步减少空间的占有量和数据的冗余度。

(7) 程序经编译、连接和用 EXE2BIN 命令转换后即可调用,程序在 Stone 286,UCDOS 2.0 下调试通过。

程序一:

```

; * * * * *
; * SAVESCR.ASM——VGA 图形汉字屏幕保存程序 *
; * Copyright (C) 1992,12 *
; * Made by Zhao Xiaodong *
; * Economic Centre,SIMT *
; * Usage: *
; * 在 FoxBASE+下,call savescr with(“文件名”) *
; * * * * *
code segment BYTE PUBLIC 'CODE'
    assume cs:code,ds:code,es:code,ss:code

savescr proc far
    jmp program

para _bx dw?
para _ds dw?
scrfile db 12 dup(0) ;File name
;
fpscr dw 0 ;File pointer
endfile db 1ah ;EOF
program: mov cs:para _bx,bx ;save register BX
        mov cs:para _ds,ds ;save register DS
;
;transfer parameters
        mov si,bx
        mov ax,cs
        mov es,ax
        mov di,offset scrfile
        mov cx,0

trans: lodsb
        cmp al,0
        jz end_trans
        inc cx
        stosb
        jmp trans

end_trans: stosb
;
        mov ax,cs
        mov ds,ax
;create output files
        mov dx,offset scrfile
        mov cx,00h
        mov ah,3ch
        int 21h
        mov fpscr,ax
;save screen to scrfile
        mov ah,00

```

```

        mov al,04      ;选择读地址选择寄存器
        mov dx,3ceh
        out dx,al
lab1:   push ax
        mov al,ah      ;选择读页面 0,1,2,3
        mov dx,3cfh
        out dx,al
        mov bx,cs;fpscr 送文件描述字
        mov cx,38400d   ;640*480/8
        mov ax,0a000h
        mov ds,ax
        mov dx,0        ;DS,DX=0A000:0000
        mov ah,40h
        int 21h
;
        pop ax
        inc ah
        cmp ah,04
        jne lab1
        mov ax,cs
        mov dx,ax
;write EOF to scrfile
clsfile: mov bx,fpscr
        mov dx,offset endfile
        mov cx,01
        mov ah,40h
        int 21h
        mov bx,fpscr
        mov ah,3eh      ;close file
        int 21h
;
        mov al,04      ;restore VGA register
        mov dx,3ceh
        out dx,al
        mov al,00
        mov dx,3cfh
        out dx,al
;
        mov bx,cs;para _bx ;restore BX
        mov ds,cs;para _ds;restore DS
        retf
savescr endp
;
code    ends
end

```

程序二:

```

;*****
;*  RESTSCR.ASM——VGA 图形汉字屏幕恢复程序 *
;*  Copyright (C) 1992,12                      *
;*  Made by Zhao Xiaodong                       *
;*  Economic Centre,SIMT                        *

```

```

; * Usage, *
; * 在 FoxBASE+下,call restscr with("文件名") *
;*****
code    segment BYTE PUBLIC 'CODE'
        assume cs:code,ds:code,es:code,ss:code
restscr proc far
        jmp program
para _bx dw?
para _ds dw?
restfile db 12 dup(0)      ;File name
;
fpscr    dw 0              ;File pointer
program: mov cs;para _bx,bx ;save register BX
        mov cs;para _ds,ds ;save register DS
;
;transfer parameters
        mov si,bx
        mov ax,cs
        mov es,ax
        mov di,offset restfile
        mov cx,0
trans:   lodsb
        cmp al,0
        jz end_trans
        inc cx
        stosb
        jmp trans
end_trans: stosb
;
        mov ax,cs
        mov ds,ax
;open input files
        mov dx,offset restfile
        mov al,0
        mov ah,3dh
        int 21h
        mov fpscr,ax
;restore screen from scrfile
        mov ah,01
        mov al,02          ;选择彩色页面写允许
        mov dx,3c4h
        out dx,al
lab1:   push ax
        mov al,ah          ;选择页面写(1,2,4,8)
        mov dx,3c5h
        out dx,al
        mov bx,cs;fpscr 送文件描述字
        mov cx,38400d   ;640*480/8
        mov ax,0a000h
        mov ds,ax
        mov dx,0        ;DS:DX=0A000:0000
        mov ah,3fh

```



```

        int 21h
    ,
        pop ax
        shl ah,1
        cmp ah,10h
        jne labl
        mov ax,cs
        mov ds,ax
;close file
clsfile:  mov bx,fpscr
        mov ah,3eh    ;close file
        int 21h
    ,
        mov al,02      ;restore VGA register
        mov dx,3c4h
        out dx,al
        mov al,0fh

```

```

        mov dx,3c5h
        out dx,al
    ,
        mov bx,cs;para _bx ;restore BX
        mov ds,cs;para _ds ;restore DS
        retf
restscr endp
;
code     ends
end

```

4 参考文献

- 1 刘旭东,“FoxBASE+2.10 及其实用工具”,《计算机用户》1992 年第 4 期。
- 2 戴水贵编著,《DOS/BIOS 功能调用及程序实例》,海洋出版社。

C 实现 EGAVGA 图形的存取与打印

上海空间电子设备研究所 方建民 (201800)

关键词 C 语言 图形存取 常驻内存 热键 图形打印

本方案提供两组 C 实用程序:G_wr.c(图形存贮)、G_rd.c(图形读取)和 G_pr.c(图形打印)。

G_wr.c 常驻内存,并由热键 F12 触发,完成对 EGAVGA 图形的存贮。EGAVGA 图形的当前显示页存在于 a000:0000 开始的 $(640 \times 480) \div 8 \times 4$ 个字节。依次为蓝、绿、红、亮各 640×480 位(假设显示模式为 640×480)。G_wr.c(HUGE 模式下编译)程序清单如下:

```

#include<stdio.h>
#include<dos.h>
#define size 38400
void interrupt intpsv()
{
    FILE *fp;
    register i;
    static char busy=0;
    char far *p=(char far *)0xa0000000;
    char far *t=(char far *)1050;
    geninterrupt(80);
    if(*t!=*(t+2)){
        t=t+*t-25;
        if(*t==(char)134){

```

```

            if(! busy){
                busy=! busy;
                fp=fopen("a:g_scr","wb");
                if(! fp)goto end;
                outportb(0x3ce,4);
                for(i=0;i<4;i++){
                    outportb(0x3cf,i);
                    fwrite(p,i,size,fp);
                }
                outportb(0x3ce,3)
                outportb(0x3cf,0);
                fclose(fp);
                end : busy=! busy;
            }
        }
    }
}
main()
{
    union REGS r;
    struct addr {char far *p;};
    struct addr far *a=
        (struct addr far *)36;

```

```

struct addr far *int9=
    (struct addr far *)320;
if(*int9->p==*(int9+1)->p){
    int9->=a->p;
    setvect(0x9,intpsv);
}
else exit(1);
r.x.dx=1024;
r.h.ah=0x31;
int86(0x21,&r,&r);
}

```

G_rd.c 完成 EGAVGA 图形的读取与显示。按任意键返回执行前状态。程序清单如下:

```

#include<stdio.h>
#include<dos.h>
#define size 38400
main()
{
    FILE *fp;
    int driver,mode;
    register i;
    static char bury=0;
    char far *p=(char far *)0xa0000000
    driver=DETECT;mode=9;
    initgraph(&driver,&mode,"")
    fp=fopen("a:g_scr","wb");
    if(! fp)goto end;
    outportb(0x3c4,2);
    for(i=0;i<4;i++){
        outportb(0x3c5,0x1<<i)
        fread(p,1,size,fp);
    }

    outportb(0x3c4,2);
    fclose(fp);
    end;busy=! busy;
}

```

G_pr.c 利用 DOS 功能调用 05h,完成 EGAVGA 图形的打印输出。本例中只打印了蓝色信息。可根据具体需要对本例作些改动。如果身边有彩打,可以通过查阅随机手册对本例作一改进,使它能为打印出漂亮的彩色图形。G_pr.c 程序清单如下:

```

#include<stdio.h>
#include<dos.h>
main()
{

```

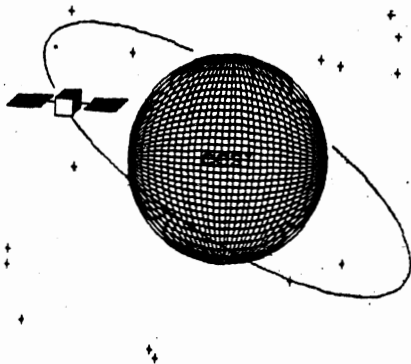
```

    FILE *fp;
    register i,j,k;
    char c[480][81];
    lp(0x0d);lp(0x0a);
    lp(27);lp('3');lp(23);
    fp=fopen("a:g_scr","rb");
    for(i=0;i<480;i++){
        for(j=0;j<80;j++){
            c[i][j]=fgetc(fp);
        }
    }
    for(i=0;i<480;i++){
        c[i][80]=0
    }
    for(j=0;j<81;j=j+3){
        lp(27);
        lp(' ');
        lp(39);
        lp(224);
        lp(1);
        for(i=0;i<480;i++){
            lp((int)c[i][j]);
            lp((int)c[i][j+1]);
            lp((int)c[i][j+2]);
        }
        lp(0x0d);lp(0x0a);
    }
}

lp(int x)
{
    union REGS r;
    r.h.ah=05;
    r.h.dl=x;
    intdos(&r,&r);
}

```

以下图形由 EPSON LQ-1600K 打印输出。



在高级语言中直接调用汉字显示中断的方法

新疆电子计算中心 龔正科(乌鲁木齐 830011)

摘要 本文通过 2.13H 的特显命令程序的分析,指出 INT 10H 接口程序存在的问题,通过具体实例给出在高级语言下调用汉字显示中断的方法,为用户设计软件的友好界面提供一条有益的途径。

关键词 操作系统 功能调用 软件 汉字系统

在 2.13H 系统的特殊显示命令中专门设置一个显示中断调用的命令,用它来实现若干功能如窗口滚动,提示行操作等等,显得很方便。

1 显示中断的接口程序

在特显命令中,有一个命令叫 I 命令,它的调用关系如下:

中断 INT10 调用 CHR[\$(14) + "I 寄存器参数串"]

其中:寄存器参数串为 AH,AL,BH,BL,CH,CL,DH,DL

这个命令可以在高级语言、操作系统命令和 dBASE 中直接作为打印命令形式来调用显示中断。它的原理是在特显命令程序中增加了一个命令转换接口,其基本程序如下:

```
L_02F0:
    CMP     AL,49H           ;是 I 吗?
    JNZ     L_032E           ;不是
;执行 INT 10H 中断
;格式:CHR(14)+'I,AH,AL,BH,BL,CH,CL,DH,DL]'
    MOV     DI,0E0H          ;寄存器临时存放地址
L_02F7:
    CALL    L_0103           ;取寄存器值
    STOSB                    ;保存
    CMP     BYTE PTR [SI],2CH ;是“,”吗?
    JNZ     L_0303           ;不是,表示参数已结束
    INC     SI               ;继续读参数
    JMP     L_02F7
L_0303:
    XOR     AL,AL            ;AL=0
L_0305:
    CMP     DI,0E8H          ;是否已读到全部的寄存器
    JZ      L_030E           ;是
    STOSB                    ;没有,则余下的寄存器赋值 0
    JMP     L_0305
L_030E:
```

```
MOV     DI,0E0H             ;寄存器表起始地址
MOV     AX,[DI]             ;取 AX
MOV     BX,[DI+02]          ;取 BX
MOV     CX,[DI+04]          ;取 CX
MOV     DX,[DI+06]          ;取 DX
XCHG    AL,AH               ;交换 AL,AH
XCHG    BL,BH               ;交换 BL,BH
XCHG    CL,CH               ;交换 CL,CH
XCHG    DL,DH               ;交换 DL,DH
CALL    L_01E39             ;执行 INT 10H
JMP     L_01E7
```

这个程序实际上一个 INT 10H 的接口程序,它沟通高级语言与 INT 10H 的关系,直接调用显示中断服务程序。这个程序根据命令格式将命令参数读到当前段地址,偏移量为 E0 的地方,然后,将其参数赋给相应的寄存器,进行汉字 INT 10H 的功能调用。这段接口程序设计为了简单,只将数据传递给 INT 10H,而不能从 INT 10H 中断服务程序得到数据,这导致许多调用如当前显示方式、读当前光标位置、读当前点信息等等需要得到数据的功能调用不能使用,这些问题都是通过源程序分析后得出的结论,原说明书中未提这一点。

2 调用方法

关于显示中断的调用输入输出关系,读者可以参阅相应的资料。下面讨论调用方法。使用 I 命令的基本方法是利用显示命令,例如:

操作系统:

```
C)SC"IAH,AL,BH,BL,CH,CL,DH,CL]"
```

其中 AH,AL,BH,BL,CH,CL,DH,DL 为寄存器参数串。

dBASE:

```
@ 1,1 SAY CHR(14)+"IAH,AL,BH,BL,CH,CL,DH,CL]"
```

BASIC:

```
PRINT CHR $(14)+"IAH,AL,BH,BL,CH,CL,DH,CL]"
```

上述调用方法中应特别注意如下几点:

2.1 寄存器的位置不能错,个数不能少,当某个寄存器不用时,赋于“0”字符,寄存器的个数如果少于

8 个,则最后的寄存器将自动赋于“0”。

2.2 这些寄存器的参数都用十进制的字符形式,不可以用十六进制,或者其他进制,因为特显程序的内部处理是读这些数字符,将其转换为十六进制处理。

2.3 碰到使用变量作为参数时,在写调用命令时,需将值转换为字符,可用 STR()函数。

2.4 要特别注意数值的换算关系。

3 窗口滚动的调用方法

在高级语言中,当行座标到底或者最上面时,系统报告出错误,如果程序中能够自动滚屏,效果将会大大改善。更进一步,如果将指定的窗口进行滚动,效果会更好。窗口滚动分为向上滚和向下滚,其调用参数如下:

向上滚动当前页

输入参数:[AH]=6

[AL]=滚动行数,=0 全滚

[CH,CL]=滚动窗口左上角字符行,列值

[DH,DL]=滚动窗口右下角字符行,列值

[BH]=空行填充字符属性

向下滚动当前页

输入参数:[AH]=7

[AL]=滚动行数,=0 全滚

[CH,CL]=滚动窗口左上角字符行,列值

[DH,DL]=滚动窗口右下角字符行,列值

[BH]=空行填充字符属性

设置窗口滚动只要 AL 的值不等于 0 即可。行、列值是字符值即行为 0~24,列 0~79。

下面的例子是在屏幕设置一个窗口,然后向上滚动窗口。

```
* -----
* 程序名:tggil01.prg
* 功能:窗口上滚
* -----
set talk off
* 将屏幕置成绿色
set color to n/g
clear
set color to
* 画一个框
@ 1,1 say chr(14)+"D280,350B180,300"
*
@ 4,36 clear to 18,55
store 4 to nn,mm
do while mm<100
    store mm to nn
    store str(nn,2) to tt
    if nn>18
        * 上滚
        @ 1,1 say chr(14)+"16,1,0,0,4,40,18,40"
        store 18 to nn
```

```
@22,0
wait
endif
@ nn,40 say tt+tt+tt+tt+tt+tt+tt
store mm+1 to mm
enddo
set color to
return
```

将这个程序输入计算机,运行,当按住任一键,窗口内的信息将自动上滚,下滚与上滚方法相同,留给读者去做。

4 提示行操作

通过显示中断调用,可以对提示行进行操作,提示行操作分为四个动作,第一是清提示行,第二是显示字符,第三是定提示行光标,第四是以 TTY 方式显示字符。提示行操作的输入参数如下:

输入参数:

[AH]=16

[AL]=功能号

=0 清提示行,光标移动行首

=1 在提示行当前位置显示字符,[DL]=字符的 ASC I 码

=2 设置提示行光标位置,[DL]=光标位置

=3 在提示行以 TTY 方式显示字符,[DL]=字符的 ASC I 码

注意提示行汉字显示,每次只能对一个汉字作用。下面是提示操作的程序实例。

```
* -----
* 程序名:tggil03.prg
* 功能:提示行操作
* -----
set talk off
clear
* 清提示行
@1,1 say chr(14)+"i16,0,0,0,0,0,0,0]"
* 在提示行光标处显示字符
@1,1 say chr(14)+"i16,2,0,0,0,0,0,40]"
@1,1 say chr(14)+"i16,1,0,0,0,0,0,65]"
wait
* 显示汉字
@1,1 say chr(14)+"i16,2,0,0,0,0,0,70]"
@1,1 say chr(14)+"i16,1,0,0,0,0,0,181]"
@1,1 say chr(14)+"i16,2,0,0,0,0,0,71]"
@1,1 say chr(14)+"i16,1,0,0,0,0,0,165]"
* 以 TTY 方式显示字符
@ 10,10say"以 TTY 方式显示字符"
@1,1 say chr(14)+"i16,2,0,0,0,0,0,40]"
@1,1 say chr(14)+"i16,3,0,0,0,0,0,65]"
return
```

5 光标操作 光标操作是指根据用户的要求,如何将光标移动到指定的位置的操作,下面是调用说明。

AH=2 设定光标位置

输入参数:[BH]=页号(仅西文下有效)

[DH]=字符行号(0~24)

[DL]=字符列号(0~79)

```

* -----
* 程序名:tgil05.prg
* 功能:光标操作
* -----
set talk off
* 将屏幕置成绿色
set color to n/g
clear
set color to
store 10 to dh,dl,ch,cl
*
store 0 to nn
do while nn<>13
    nn=inkey()
    do case
        case nn=5    && 上
            store dh-1 to dh
            if dh<0
                store 0 to dh
            endif
        case nn=24    && 下
            store dh+1 to dh
            if dh>24
                store 24 to dh
            endif
        case nn=4      && 右
            store dl+1 to dl
            if dl>79
                store 0 to dl
            endif
        case nn=18     && 左
            store dl-1 to dl
            if dl<0
                store 79 to dl
            endif
    endcase
    @ 1,1 say chr(14)+"I2,0,0,0,0,0,"+str(dh,2)+" ",
    +str(dl,2)+" "]"
    @ 1,1, say str(dh,2)+" " +str(dl,2)
enddo
set color to
return

```

6 写点操作

写点操作是指在屏幕上按指定颜色号显示一个亮点,它是画直线、曲线等图形的基本操作,写点操作的调用关系如下。

AH=12 显示点

输入参数:[AL]=点的值,若最高位为1,则写

入值与原点异或

[CX]=点的列位置(X坐标)

[DX]=点的行位置(Y坐标)

下面的程序是首先在屏幕的四个极点显示,然后用写操作画一条线。

```

* -----
* 程序名:tgil04.prg
* 功能:写点操作
* -----
set talk off
clear
* 显示四个座标点
@ 1,1 say chr(14)+"I12,2,0,0,0,0,0,0]" && (0,0)
@ 1,1 say chr(14)+"I12,2,0,0,0,0,0,1,223]" && (479,0)
@ 1,1 say chr(14)+"I12,2,0,0,2,127,0,0]" && ch 2=512+127=639
@ 1,1 say chr(14)+"I12,2,0,0,2,127,1,223]" && dl 1=256,256+223=479
* 画一条线
store 0 to tt
store 0 to nn,mm
do while nn<640
    store 0 to mm
    do while mm<256.and. nn<640
        do case
            case tt=0
                @ 1,1 say chr(14)+I12,3,0,0,0,"+str(mm,3)
                +",0,100]"
            case tt=1
                @ 1,1 say chr(14)+I12,3,0,0,1,"+str(mm,3)
                +",0,100]"
            case tt=2
                @ 1,1 say chr(14)+I12,3,0,0,2,"+str(mm,3)
                +",0,100]"
        endcase
        store mm+1 to mm
        store nn+1 to nn
    enddo
    store tt+1 to tt
enddo
return

```

通过上述例子,可以清楚地看出,通过特殊显示命令中的I命令和高级语言建立非常简单的接口关系,使用方法也比较简单。

HP3000 系统应用研究

华中理工大学 刘启茂 蔡红梅 方红 (武汉 430074)

摘要 本文首先简要介绍了 HP3000/925LX 系统在我校图书馆的应用概况,然后从系统应用的更新与完善、数据库的建设、期刊篇名数据库及其检索系统与基础研究信息库及其检索系统的建立等方面入手,具体论述了 HP3000 系统的应用研究及其成果,并提出了今后进一步研究与开发的初步设想。

关键词 HP3000 系统 软件更新 数据库 检索

1 HP3000 系统应用概况

我校图书馆引进的 HP3000/925LX 系统⁽¹⁾于 1991 年 4 月初安装,同年由学校下达“HP3000 系统的开发与应用”研究课题,经过近一年的应用实践与研究,正在逐步实现图书馆自动化管理,初步研究开发了几个专用数据库及其检索系统。目前该系统除用于采购、编目、典藏等业务管理外,还积极开展了面向读者服务,其中有流通部的借还书、预约、罚款、查询、期刊篇名检索以及申报基础研究课题的查新等,这不仅给读者提供了很大方便,而且大大节省了读者的时间,取得了较好的经济效益和社会效益。图 1 为 HP3000 系统现阶段的应用功能示意图。

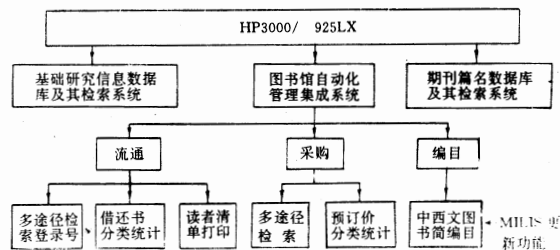


图 1 HP3000 系统应用功能图

下面将就 HP3000 系统应用研究作一系统论述。

2 应用软件的更新与完善

为了使 HP3000 系统尽快投入使用,必须尽可能利用国内外现有成果,在反复调查、论证基础上,我们引用了上海交通大学研制的 MILIS 系统⁽²⁾,将之作为我校图书馆自动化管理的基础,使系统得以有可能在去年九月初首先在流通部正式投入使用,这一成果受到了全校广大师生的普遍欢迎与好评,并在去年先后在校刊和长江日报上作了报道,从而扩大了图书馆在学校和社会中的影响,为今后的自

动化工作奠定了基础。通过一段时间的应用实践与探索,根据我校实际,我们舍弃了 MILIS 系统中的一些不足之处,对 MILIS 系统有计划、有步骤地作了必要的更新与完善,增加了一些实用独特的功能。现将 MILIS 主要更新功能介绍如下:

2.1 多途径查找登录号 图书流通自动化管理的特点是要求流通事务特别是借还书响应时间要快。因而它是建立在 HP3000 配有的 IMAGE 网络型 DBMS 之上的,同时,为了使系统尽快投入使用,采用了边流通边建库方案⁽³⁾,而在建立的流通书目数据中只包括索书号和条码号,且老书的条码号与登录号不一致,这样,就给实际应用带来了一定困难,如丢书读者只有查到所丢图书的登录号后方能确定其赔偿额,为此,我们在 MINISIS 基础上,建立一只包含索书号、书的条码号和登录号的临时数据库,且对条码号和索书号建立快速查找文件,借助于根卡及时手工录入,同时,通过 MINISIS 与 IMAGE 的接口,将条码号和索书号传入流通数据库的书目数据集(ITEM),于是,读者赔书即可在 MINISIS 数据库中通过索书号和条码号来查找登录号,既方便又省时。

2.2 流通分类统计 由于 MILIS 系统中的流通统计不能满足我们的需要,为此,我们以每天借还书产生的两个数据文件 CHTABLE 和 DCHTABLE 为入口,重新设计流通统计功能⁽⁴⁾。

重新设计的流通统计程序具有以下功能:

(1)按中图分类法各大类,准确统计出每天、每月、每年各类读者借还各类图书的种、册数。

(2)统计每天、每月、每年各类读者到馆人数。

2.3 打印读者清单 为了帮助丢失借书证的读者迅速、方便查找其借书证的条码号,以便及时进行挂失处理,我们设计了一程序用于打印读者清单,此清单包括条码号和相应的读者姓名,按单位集中,按条

码号排序⁽¹⁾。

2.4 建立中西文图书简编目库 在中西文图书的编目中,由于标准数据源的获取时间常滞后于图书到馆时间,若全部图书均按标准 MARC 格式著录,先填写工作单,再手工输入,无疑将花费很大人力、财力和时间。为此,我们从标准书目数据库中抽取题名与责任者、ISBN、统一书刊号、出版发行项、索书号、中图分类号、责任者、馆藏复本数等字段建立一 PS 库作为简编目库,并建立相应的 VPLUS 屏幕格式,卡片打印格式,以便输入数据和打印卡片,待取得标准数据后,再用标准数据代替简编目数据,这样既可提高编目效率,缩短编目周期,使新到馆的图书尽快与读者见面,又可及时为读者提供基本的查询服务。

2.5 采购统计功能的更新 在图书采购中,特别是西文图书的采购,由于图书定价的货币单位不同。因而在各类价格统计中,不能简单将数字相加。为此我们利用 MINISIS 中的 COMPTEP 模块重新设计了采购预订总价的统计⁽¹⁾。新的采购总预订价统计既可统计每批书的种数、册数,同时可统计出一批书所需各种货币量。

3 数据库的建设

3.1 数据源 在图书馆自动化管理中,数据库的建设是一个关键问题,书目数据库的检索、卡片制的废除等都依赖于书目数据库的建设,只有设法加速数据库的建设,保证数据的质量,才能为整个图书馆自动化提供良好的基础。

在书目数据库的建设中,我们的基本原则是:

3.1.1 充分利用标准数据源 为了实现资源共享,保证数据的高质量与规范性,我们建库的首要原则是充分利用标准书目数据。对于西文书刊,我们利用上海交通大学的 CD-ROM 光盘进行检索,而对中文图书,则利用北京图书馆提供的“中国国家书目机读目录”、产品进行检索,对命中的数据进行自动化编目。

3.1.2 积极采用已有书目数据 为了加速书目数据库的建设,除利用标准数据源外,还需积极采用已有书目数据。在我们采用 HP3000/925LX 进行图书馆自动化管理之前,已研制成功了期刊网络系统,用于期刊的自动化管理⁽²⁾,进库数据已达六千多条。如果在 HP3000 上重新建立期刊书目数据库,则造成大量和重复劳动,带来相当大的浪费,为此,我们将已有的期刊数据也作为一个重要的数据源。

3.1.3 认真进行手工编目 对于缺乏以上二种数据源的书刊,其建库则采用手工编目。

3.2 数据源的利用 对于已有的不同种类的书目

数据源,如何将其分别转换成 MINISIS 数据库中的数据呢?这一数据转换全过程可参见图 2。

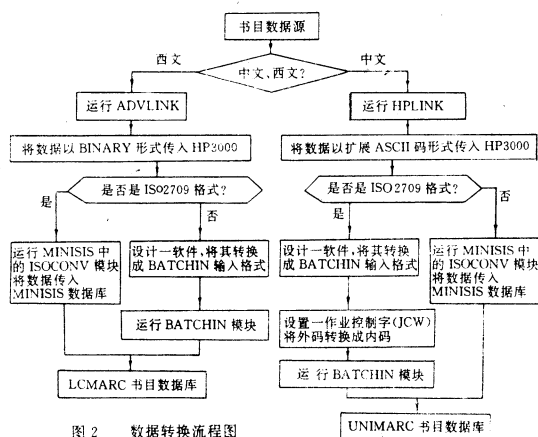


图2 数据转换流程图

4 期刊篇名专用库及其检索系统的建立

中国科技情报所重庆分所发行的“中文科技期刊篇名数据库、机读产品收录国内外期刊 4212 种(含港台刊 102 种),年题录量 18 万条,每条记录含:分类号、篇名、著者、出处、主题词等。我们已按专业分类购置了该产品的 25 万条记录。由于该产品系经过加密的数据源,按一般途径不能显示其内容,无法了解其数据结构,为了利用这一数据源建立期刊篇名专用库及其检索系统,充分利用 HP3000 容量大、速度快,以及 MINISIS 的强大的情报检索功能,更好地为读者服务,我们对该产品的数据结构进行了具体分析,在 PC 机上将其转化为可利用的 dBASE 数据文件,最终传入 HP3000 的 MINISIS 数据库,并在 HP3000 上,利用 MINISIS 设计了多种快速检索途径、用户使用提示菜单以及输出打印格式,满足了读者的需要,取得了良好的社会效益。

图 3 即为中文科技期刊篇名数据库建立流程图:

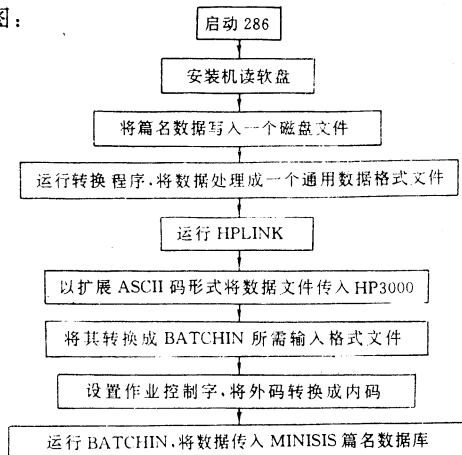


图3 中文科技期刊篇名数据库建立流程图

5 基础研究信息数据库及其检索系统的建立

为了充分利用现有设备资源,扩大 HP3000 系统的应用,我们开展了包括国家自然科学基金高校博士点基金、教委优秀青年教师基金、霍英东基金、国家社会科学基金、国家青年社会科学基金等六大基金的数据库建设及其检索系统的建立工作。建立该库的主要目的是为基金申请者提供多途径、多层次查新服务,以提高申报获准率,同时为上级主管部门提供审查依据,以帮助决策避免重复资助。

建立六大基金数据库,关键是以下两方面工作:

5.1 HP3000 上利用 MINISIS 建立六大基金库及其检索系统 关于六大基金的建库,以前也曾有些单位做过一些工作,但其设计目的主要是用于统计分析,不适于检索查新之用,如在 dBASE III 支持下的自然科学基金库,每条记录达 70 余项,学科代码、专业职务代码等均无相应的中文名称,记录中也无内容提要,而其他基金库连关键词字段也没有,显然对查询很不适用,除此之外,这些系统还受 PC 机容量及速度的限制,以及支持软件 dBASE III 对字段的定长限制。

为了解决以上问题,经过反复论证,我们决定以 MINISIS 做为支持软件建立基金数据库及其检索系统,目前可从关键词、基金类型和学科代码等途径进行单项或组配检索。具体结构参见图 4:

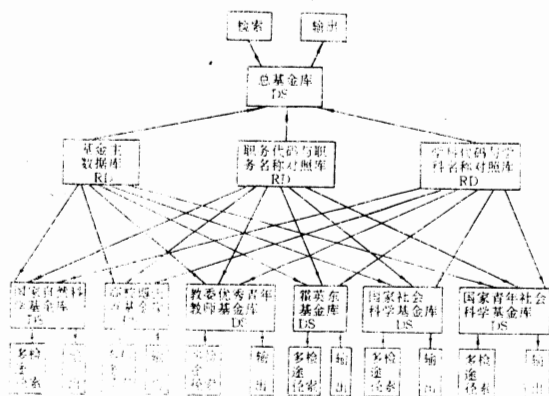


图4 基金数据库结构功能图

除此之外,还利用 VPLUS 设计良好的用户界面,同时利用 MPEXL 操作系统提供的帐户结构,给不同用户以不同的能力,并适当设计保密字,以确保数据的安全性。

5.2 已有数据的利用 充分利用 dBASE III 上已有基金库的数据,是加速基础研究信息数据库及检索系统建设,使其尽快投入使用的关键。为了利用已有数据源,首先必须在 dBASE III 中取出所需的数据,然后利用 PC 机与 HP3000 的仿真通讯技术将数据由

微机传入 HP3000,再进行格式转换,将其传入 MINISIS 的基金主数据库,并对其进行加工。具体做法如图 5。

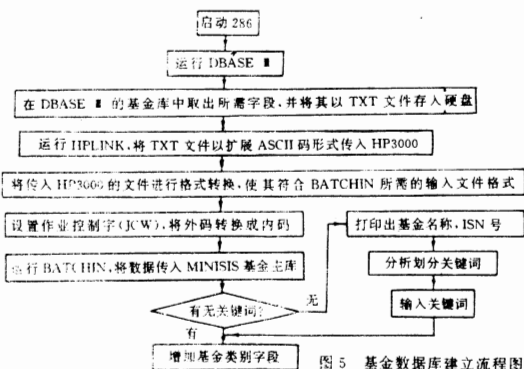


图5 基金数据库建立流程图

6 MINISIS 数据库数据的交换

在前面我们已谈到,不论在建立何种数据库时,我们都充分利用本单位或兄弟单位已有的标准数据或非标准数据,同样,为了做到资源共享,促进单位间的相互协作,避免数据的重复输入,我们也积极为其它单位提供已有的 MINISIS 数据库数据。

为了单位间的数据交换,我们可根据不同的需要将 MINISIS 数据库数据转换成以下两种便于用户利用的文件形式。

6.1 ISO 格式文件:为了将 MINISIS 数据库数据转换成 ISO 格式文件,需按以下步骤进行转换:

6.1.1 定义一个 CD 数据库,将所需转换的字段在 CD 中定义。

6.1.2 定义一个 ISO 文件,并指出使用软盘或磁带存储 ISO 格式文件。

FILE ISO; DEV=DISC

OR DEV=TAPE

6.1.3 运行 ISOCONV,使用 LOAD 命令将 MINISIS 数据库中数据写到 ISO 文件上。

6.1.4 如若 ISO 文件存入磁盘,则将 ISO 格式文件由 HP3000 传入 PC 机,存入磁盘,这样,既可以磁带、也可以软盘为介质,将数据提供给用户。

6.2 “通用数据格式”文件:为了将 HP3000 上的 MINISIS 数据库数据转换成 dBASE 数据库可利用的“通用数据格式”文件,必须利用 MINISIS 中的 PRINT 模块:

6.2.1 利用 PRINT 模块,设计一个打印格式文件,按照所需字段,将文件的输出格式编辑成 dBASE 数据库可以接受的通用数据格式。

6.2.2 定义一输出文件 OUTFL,并指出输出设备为磁盘。

6.2.3 运行 PRINT 模块,将 MINISIS 数据库数据输出到所定义的输出文件中。

6.2.4 将输出文件传入 PC 机存入软盘。

这样就可以将 MINISIS 数据以软盘形式提供给 dBASE 用户。

7 结语

经过近一年的实践与探索,我们在 HP3000 上已进行了一些应用研究。随着实践的深入,将会遇到更多的问题。譬如:在图书馆自动化管理系统中,订购查重、编目查重希望能成批处理;在基金数据库及其检索系统中,希望增加一些人工智能检索功能,这些都是我们亟待解决的问题。除此之外,在系统管理方面,如系统的配置、系统后备、磁盘空间的管理、设备的管理等也进行了一定的探索,积累了一些经验,

同时,在 HP 扫描仪,HP 激光打印机,HP286 组成的系统上也进行了初步开发,我们希望能有机会与其它 HP 用户互相交流,借鉴经验,进一步开发已有资源,扩展系统的功能与用途,以发挥其最大效益。

8 参考文献

- 1 刘启茂、方红、蔡红梅,“MILIS 系统软件的应用与更新设计。”《计算机应用研究》,1992 年第 4 期。
- 2 刘启茂、严春兰,“一种期刊网络系统实用化方案的实现。”《计算机应用研究》,1992 年专辑第 1 号。
- 3 刘启茂、蔡红梅,“图书馆自动化与 HP3000 系统”,《计算机应用研究》,1992 年第 2 期。
- 4 刘启茂、蔡红梅,“高校图书馆的自动化初探”,《计算机应用研究》,1991 年专辑第 2 号。

软件的封面设计

机电部第二十七研究所 徐 琦 (河南驻马店 463000)

摘要 本文通过了解 SPT 图文编辑系统中图形数据文件的结构,提出了一个 SPT 图形数据信息再利用的方法。图形的生成、加工比较简单,显示处理也不复杂。同时也讨论了图形方式下趣味清屏的实现等。通过编程实验,不但给软件增添了风采,同时也从中享受到计算机软件设计的不少乐趣。

关键词 软件封面 SPT 图文编辑系统 图象重现

随着计算机技术的发展,计算机的图形、图象也越来越精美、越逼真。如果我们能将这些精美逼真的图画截取下来,作为我们自己的软件封面,那一定有很多朋友感兴趣的。但是,精美逼真的图画的生成、存储、显示等技术比较复杂,加工、实现起来也比较困难。我们可以利用目前较流行的 WPS 桌面印刷系统中的 SPT(Super-Star)或 Windows 中的绘图工具 PaintBrush 或者两者结合起来制作我们的软件封面、(【参见 1】),图形的生成、存储、显示加工等也比较容易。

1 软件封面图画的制作

WPS 桌面印刷系统是目前较为流行的汉字处理软件,它的 SPT 图文编辑系统更让人爱不释手, SPT 中的汉字大小可达 480×480 ,且可自定义大小,没有有些汉字系统中汉字放大以后产生的锯齿,美观大方,并且具有宋、楷、仿宋、黑等四种字体。西文有 11 种字体等(PaintBrush 不具有汉字处理功能,我们可以把 PaintBrush 中的图形数据文件转换成 SPT 数据文件,然后在 SPT 图文编辑系统中进行汉

字处理。【参见 1】)。

用户首先根据计算机分辨率的大小设置封面图画尺寸如 640×480 、 640×350 等,然后根据用户自己的创作意图在 SPT 中对封面图画进行布局设计。不理想的地方可以擦掉重画,直到用户满意为止。

然后将绘好的封面图画存盘,此时应注意两点:

- ①要选用 Super-Star 的文件类别(S);
- ②采用非压缩的存储格式(E)进行存储。

对于其它的文件类别及存储格式在画面重现时要复杂一些。

2 SPT 图形数据文件的结构(Super-Star 文件类别,非压缩存储格式)

SPT 图形数据文件有一个 64Byte 的文件头, 00H~0FH 字节为 SPT 图形数据文件的标志, 1CH~1FH 为作者姓名(C0 EE C3 F7 李明), 22H~23H 为图形画面宽度(以像素为单位), 24H~25H 为图形画面高度(以像素为单位)。

紧接在 SPT 图形数据文件头之后,自 40H 字节开始为图形数据信息,其长度为版面宽度 \times 版面高

度/8(字节)。图形点阵数据的结构很简单,它是按照从左到右、自上而下的次序来描述各点的数据的(每字节描述 8 个点)。

3 应用程序中画面的重现

在应用程序中设置一字节型数组,大小为 SPT 图形数据文件的大小,用以存放图形数据信息,对于 640×480 大小的图画,应这样设置:

```
var
```

```
Buffer:Array[1..38464]Of Byte;
```

然后打开 SPT 图形数据文件,将数据信息读入 Buffer 数组中,计算出图形的尺寸大小:

```
Assign(F,'G.SPT');
```

```
Reset(F,1);(无类型文件)
```

```
BlockRead(F,Buffer,SizeOf(Buffer),Rec);
```

```
Close(F);
```

```
Row:=Buffer[36]*256+Buffer[35];
```

```
Col:=Buffer[38]*256+Buffer[37];
```

图形数据的每一位即对应于屏幕中的一个点,如果某一位为 0,则说明屏幕相应位置上无点(即该点颜色与背景颜色相同);如果某一位为 1,说明屏幕相对位置上有点,则在该位置上显示出该点(与背景颜色不同),直到整个图形数据文件全部处理完毕,即将保存的图形重新显示在屏幕上了。

若想使画面自上而下逐渐显示,此时可建立 HorDownPrint 过程:

```
Procedure HorDownPrint(Row,Col;Word;Color;Byte);
```

```
Var
```

```
i,j;Word;
```

```
m,k;Byte;
```

```
Begin
```

```
For i:=0 To Col-1 Do
```

```
For j:=0 To (Row-1) Div 8 Do
```

```
Begin
```

```
m:= $ 80;
```

```
For k:=0 To 7 Do
```

```
Begin
```

```
If(Buffer[65+j*Row Div 8+j]And m)(>)0 Then
```

```
putpixel(j*8+k,i,Color);
```

```
m:=m Div 2;
```

```
End;
```

```
End;
```

```
End;
```

参数 ROW 为图形宽度, COL 为图形高度, COLOR 为颜色值。

HorDownPrint 过程包含有三个循环;外循环处理整个屏幕;第二个循环处理屏幕中一行的数据信息;内循环完成一个字节中 1bit 象素的处理,如果该

bit 为 1,则在屏幕相对应位置上以 Color 颜色值画一象素点。

如果想使画面从左至右逐渐显示,可建立 VerRightPrint 过程:

```
Procedure VerRightPrint(Row,Col;Word;Color;Byte);
```

```
Var
```

```
i,j;Word;
```

```
m,k;Byte;
```

```
Begin
```

```
For i:=0 To (Row-1) Div 8 Do
```

```
For j:=0 To Col-1 Do
```

```
Begin
```

```
m:= $ 80;
```

```
For k:=0 To 7 Do
```

```
Begin
```

```
If(Buffer[65+j*Row Div 8+i]And m)(>)0 Then
```

```
PutPixel(i*8+k,j,Color);
```

```
m:=m Div 2;
```

```
End;
```

```
End;
```

```
End;
```

参数 Row、Col 及 Color 与 HorDownPrint 过程参数相同。此过程的执行过程同 HorDownPrint 过程类似,只是循环次序不同。

笔者编写了几个用来重现图形(西文方式、中文方式均可)的过程:如画面自下而上显示的 horupPrint 过程;从右至左显示的 verLeftPrint 过程;从上下向中间显示的 HorDownUpMidPrint 过程;从两边向中间显示的 VerLeftRightMidPrint 过程以及它们的逆向显示过程等。如果在不清除原来图画的基础上重新以另一颜色值重画画面,则可达到屏幕的变色效果(前景、背景均可变)。

```
Procedure PartHorDownPrint(StartX,StartY,EndX,EndY;  
Word;Color;Byte);
```

```
Var
```

```
i,j;Word;
```

```
m,k;Byte;
```

```
Begin
```

```
For i:=0 To StartY To EndY Do
```

```
For j:=0 To StartX Div 8 To EndX Div 8 Do
```

```
Begin
```

```
m:= $ 80;
```

```
For k:=0 To 7 Do
```

```
Begin
```

```
If(Buffer[65+i*Row Div 8+j]And m)(>)0 Then
```

```
PutPixel(j*8+k,i,Color);
```

```
m:=m Div 2;
```

```
End;
```

```
End
End;
```

为了突出图画의局部信息,可以采用与其它背景和前景均不相同的颜色值在原来位置上重现该局部画面,PartHorDownPrint 可实现这一过程。参数 StartX、StartY 为局部画面区域的左上角坐标,EndX、EndY 为右下角坐标,ROW 为整个画面的宽度。

局部画面的重现过程与整个画面的重现过程相同,笔者也编写了几个与整画面显示相类似的局部画面重现过程,两者可以合二为一。

4 画面的清除

Turbo Pascal 提供了两个清除画面的标准过程:ClearDevice 和 ClearViewport。这两个清除画面的过程可使画面的前景瞬间消失,虽然速度极快,但给人们一种枯燥、呆板的感觉,如果给计算机配上几种有趣的清屏过程,达到象电视特技镜头那样的效果,肯定可以给软件增添不少的风采。

过程 HorDownErase 从屏幕上部开始一直清到屏幕的底部,即屏幕上部的图画首次消失,直到屏幕底部画面消失为止,原理是以背景颜色从屏幕上部开始自左至右画一直线,直到画满整个屏幕,即把屏幕清除了。

```
Procedure HorDownErase;
Var
  i: Word;
Begin
```

```
SetColor(GetBKColor);
For i:=0 To GetMaxY Do Line(0,i,GetMaxX,i);
End;
```

笔者编写了几种趣味清屏过程,其基本原理都是把屏幕前景颜色换成背景颜色。读者可以依此原理编写出更加丰富多采的清屏过程:如从屏幕底部向上清屏,从左向右从右向左清屏,垂直拉幕式清屏,十字线清屏,方格状清屏等。

5 复杂图形(象)的重现

复杂图形图象的截取及重现可参见参考文献 2 中《后台截取 EGA/VGA 图象》一文,读者可以在此基础上对画面或叠加、或剪取等处理,使画面的屏幕显示更加美观、漂亮。

6 结束语

笔者曾按照上述方法应用 Turbo Pascal 5.5 制作一自动演示程序,经使用效果良好,不但给软件增添了不少风采,同时也给操作人员带来了无穷的乐趣。

本文所介绍的软件封面的制作方法,并不是一种十全十美的方法,比如显示速度较慢等。读者可在此基础上加以改进,比如将图形(象)信息直接送入计算机图形显示缓冲区,则显示速度将大幅度提高。

7 参考文献

- 1 《计算机世界月刊》,1992.3.
- 2 《计算机世界月刊》,1992.5.
- 3 《Turbo Pascal 5.5 技术参考大全》,中国科学院希望高级电脑技术公司,1991 年元月。

(上接 21 页)

5 结束语

本文针对大型企业的递阶管理机构、信息繁多、数据量大等特点,以一大型矿务局为例,提出了 IMIS 的基本概念、设计原则,并介绍了它的基本结构,对 IMIS 中 DBS 的设计和子系统功能模块的形成、CDB 的查询方式和模块调用方法作了较详细的讨论。笔者相信,随着 MIS 的开发研究的不断深入,IMIS 在大型企业中的优势必将充分发挥出来,它的应用,将大大提高企业的管理效率,并为企业决策者进行合理决策创造一个可依赖的良好的决策环境。

6 参考文献

- 1 M. C. Er, Decision Support Systems: A Summary, Problems and Future Trends, Decision Support Systems 4 (1988).

- 2 Wang Zongjun (王宗军), Decision Support System for Production Planning, Proc. of 11th International Conference on Production Research (ICPR), 1991.
- 3 王宗军,“生产计划分析与优化支持系统 (PP-AOSS)”,《第六届全国系统与控制科学青年学术年会论文集》,1990.
- 4 张俊普,“面向企业计划管理的决策支持系统”,《中国计算机用户》,第 21 期,1987.
- 5 R. G. Murdick and J. E. Ross, Introduction to Management Information Systems, Prentice-Hall, Inc., Englewood Cliffs, 1977.
- 6 C. D. Darix and M. H. Olson, Management Information Systems, McGraw-Hill, 1985.
- 7 王冬,“国家经济信息系统的数据库系统”,《中国计算机用户》,第 8 期,1988.

一个解决微机“死机”的有效方法

吉林省计算机技术研究所(130012)

长春机械工业学校计算机室

张鸿鸣 刘铁军

张科

摘要 本文主要概述用汇编语言编写程序,并用键盘中断解决微机“死机”的问题,为调试程序提供方便。

关键词 死机 死循环 汇编语言 键盘

对于一个程序员来说,很难使自己编制的程序不需修改就能通过。可见,编制程序必须通过调试才能完成。因此,调试是编制程序不可缺少的一个重要环节。

在调试程序的时候,往往会遇到“死机”现象。“死机”的原因主要是由于程序陷入了死循环,这时就不得不重新启动微机。这样不仅浪费了时间,而且影响到机器的使用寿命。

能不能在“死机”的时候用键盘中断死循环呢?我们试着用汇编语言编写了一段程序(见程序清单)。程序通过修改该键盘中断程序,达到了这一目的。由于键盘中断是 IRQ1,它有很高的优先权,所以可以解决了大部分“死机”问题,为调试程序提供了方便。

1 原理

大家知道,当你每次按下键盘时无论有没有程序在运行都会引起一次 INT9H 中断,中断结束返回中断点。当 CPU 接到一个中断请求时,首先将当前有关信息保存在堆栈中(包括状态标志 CS 和 IP),然后将中断入口地址放入 CS,IP 寄存器,这样就可以转入相应的例行程序。返回时,恢复 CS、IP 和状态标志,使能够继续执行刚刚被中断的程序。保存信息的过程如下:

- ```
(1)(sp)=(sp)-2
 ((sp)+1,(sp))=flags
(2)(sp)=(sp)-2
 ((sp)+1,(sp))=cs
(3)(sp)=(sp)-2
 ((sp)+1,(sp))=ip
```

在返回时,过程恰好相反。只要我们修改堆栈中 CS 和 IP 的值,就能在返回时不再回到原来的中断点,而执行用户自己的程序。

### 2 程序

本程序就是根据这一原理编写的,并取得理想的效果。其程序清单如下:

```
code segment
assume cs:code,ds:code
org 100h

start: jmp begin

int9off dw? ;原键盘中断偏移量
int9seg dw? ;原键盘中断段址
hk1 equ 04 ;“CTRL”键
hk2 equ 30h ;“B”键
int2loff dw? ;原 21h 中断偏移量
int21seg dw? ;原 21h 中断段址
comp: push ax ;AX 入栈
 in al,60h ;读键盘
 cmp al,hk2 ;是“B”键吗?
 jnz sint9 ;不是转原键盘中断
 mov ah,02 ;2号功能
 int 16h ;调用键盘 I/O 中断
 and al,hk1 ;“CTRL”键未按下转原键盘中断
 jz sint9 ;

int9: pop ax ;
 pop ax ;寄存器 AX 出栈
 pop ax ;
 push cs ;寄存器 CS 入栈
 mov ax,offset exit ;寄存器 AX 指向 EXIT
 push ax ;
 push ax ;AX 入栈
sint9: pop ax ;
 jmp cs:dword ptr int9off ;进入原键盘中断程序
exit: mov ah,4ch ;返回 DOS
 int 21h ;

;新的 21H 中断入口
int21: cmp ah,0ffh ;调用 FF 号功能吗?
 jnz sint21 ;不是转原 21H 中断
 mov al,0aah ;寄存器 AL 赋 0AAH
 iret ;中断返回
sint21: jmp cs:dword ptr int21off ;执行原 21H 中断
begin: mov ah,0ffh ;调用 0FFH 功能
 int 21h ;
 cmp al,0aah ;安装过否?
 jnz install ;未安装过进行安装
 push cs ;
 pop ds ;CS=DS
 mov ah,09 ;
 mov dx,offset mess1 ;显示已安装信息
 int 21h ;
 mov ah,4ch ;
 int 21h ;返回 DOS
;安装程序
```



```
install: mov ax,3509h ;
 int 21h ;取键盘中断向量
 mov int9off,bx ;
 mov int9seg,es ;存键盘中断向量
 mov ax,2509h ;
 mov dx,offset comp;修改键盘中断向量使指向 COMP
 push es ;
 pop ds ;
 int 21h ;
 mov ax,3521h ;
 int 21h ;取 21H 中断向量
 mov int21off,bx;
 mov int21seg,es;存 21H 中断向量
 mov ah,25h ;
 mov dx,offset int21;修改 21H 中断向量使指向 INT21
 int 21h ;
 mov ah,09 ;调用 09 号功能
 mov dx,offset mess2;
 int 21h ;显示“OK!”
```

```
 mov dx,offset begin ;
 int 27h ;驻留并返回 DOS
 mess1 db "Break—key has been installed!". 0dh. 0ah. 24h
 mess2 db "OK!". 0dh. 0ah. 24h
 Code ends
 end start
```

程序中为了避免程序的重装入,修改了 21H 中断。经过实践,发现该程序不但可以在“死机”的时候正确返回 DOS,而且还能够中断其它正运行的程序。例如:PE 和 PCTOOLS 等软件。

### 3 用法

在调试程序前执行本程序,使其驻留内存。当遇到“死机”的时候,按 CTRL-B 即可返回 DOS。

## 如何实现 DOS 系统下应用软件到 XENIX 的转换

西南财经大学 郑 英 (成都 610074)

XENIX 是一种多用户、多任务的分时系统,以其高效率、高性能、应用程序良好的可移植性,赢得了越来越多的用户。可是大多数用户以前都是 DOS 用户,他们用 DOS 系统下的高级语言开发了不少应用程序,如在 DOS 系统下开发的 dBASE III 软件。那么,这种单用户应用程序如何转换到 XENIX 系统下使用?这是用户所希望做到的,同时 XENIX 也是有能力做到的。

一般地说,DOS 下的高级语言程序只需转到 XENIX 下,稍作改动就可执行。

下面就以 DOS 系统下 dBASE III 程序为例,按步骤说明如何实现到 XENIX 的转换。

#### 1. 在 XENIX 下,安装多用户 FOXBASE+。

(1)进入 /tmp 目录: # cd/tmp

(2)拷贝安装程序:将装有 FOXBASE+ 的盘插入 0 号驱动器,键入命令:

# tar xvf/dev/fdo96ds15 in stall

(3)执行安装程序: # /tmp/in stall

(4)系统中各用户只要打入命令 \$ FOXPLUS 即可进入 FOXBASE+。

2. 将 DOS 系统下 dBASE III 应用软件拷到软盘,把软盘插入 A 驱动器,准备把软盘上应用软件拷到 XENIX 下。

(1)如果软盘上所有的文件都要移植,那么只需

在操作系统提示符下键入 foxpget。

(2)如果只选择软盘上部分文件,那么可用 doscp 命令。

例: \$ doscp A:fill. prg /usr/user/ful. prg

将 A 盘上 fill. prg 拷到 XENIX 目录/usr/user 下。

应注意的是,由于 XENIX 的文件名区分大小写,所以应遵循通常的约定,即 FOXBASE+ 的文件名和程序名都应是大小写。foxpget 实用程序将自动地将 DOS 的数据库文件名转换成小写。如果我们用 doscp 命令转换 DOS 系统下的文件时,首先得把 DOS 系统下的文件名全都转换成小写的文件名。

多用户 FOXBASE+ 与 dBASE III 在语法和语义上是兼容的,应用程序能正确地在 dBASE III 下运行,则同样能正确地在多用户 FOXBASE+ 下运行。但是,多用户程序与单用户程序有一个重要区别:前者允许多用户共享数据库,并且能对共享数据库对产生的碰撞问题进行管理。因此,单用户应用程序要稍稍修改一下,增加管理数据库状态的命令,避免多个用户共享数据库,对数据库进行修改操作时发生死锁。

这样就实现了 DOS 系统下应用软件到 XENIX 的转换。

# 利用 TSR 程序实现假脱机绘图

苏州城建环保学院计算中心 李大宏 (215008)

**摘要** 本文介绍了利用 CAD 软件包生成的绘图文件和 DOS 3.3 中 Print.com 程序完成假脱机绘图的步骤和方法。

**关键词** 假脱机绘图 绘图文件

## 1 引言

随着近几年来 CAD 技术的推广和普及,很多单位每天要用计算机绘制大量的图纸,大多数 CAD 软件包都不包含假脱机绘图程序,因此普遍采用联机绘图,从而占用了大量时间,使设计者无法利用 CAD 系统进行设计,造成了资源的巨大浪费。本文介绍了用假脱机绘图解决这一问题的方法。假脱机绘图程序是典型的 DOS 可执行文件,它设计用来把绘图文件以后台控制方式送到被指定的绘图设备,而允许用户继续在 CAD 系统工作。假脱机程序完全独立于 CAD 系统,它的活动不受 CAD 系统的控制。

## 2 实现方法

经过分析发现 DOS3.3 中的 TSR 程序 Print.com 就能作为假脱机绘图程序,假脱机绘图有两种不同情况:一是在绘图机上进行,二是在打印机上进行。前者较复杂。

现以如下硬件环境:AST386,RS232-C 串行口,惠普公司 DraftPro 绘图机,EPSON LQ-1600 打印机,绘图机接在 COM1,打印机接在 PRN 为例分两种情况加以介绍:

### 2.1 在绘图机上实现假脱机绘图的步骤

**2.1.1 设置异步通讯适配器** 本例中 DraftPro 绘图机开关采用其缺省设置:<sup>(1)</sup>波特率 9600 波特,非校验,关闭监听,仿真和扩展置在 normal 状态。命令如下:

```
C>mode com1:9600,n,8,1,p
```

#### 2.1.2 假脱机程序驻留

```
C>Print/D:com1/U:1
```

该命令只需使用一次,发出的这个命令要增加内存中 DOS 的驻留长度。

**2.1.3 假脱机绘图** 首先将 CAD 系统中的图形文件转为绘图文件,一般较好的 CAD 软件包,如 AutoCAD EDA(Electronics Design Automation),都具有这一功能,然后将要绘制的绘图文件加入打印队列。命令如下:

```
C>print 绘图文件名/P
```

要绘制其它绘图文件,只要同样将该文件加入队列即可。

## 2.2 在打印机上实现假脱机绘图

### 1. 假脱机程序驻留

```
C>Print/D:Prn
```

### 2. 假脱机绘图(同上第三步)

## 3 讨论

**3.1** 假脱机程序必须在启动 CAD 系统前驻留,不能在系统 DOS SHELL 中使用,否则将引起难以预料的结果。

**3.2** 当 PRINT 命令有要处理的数据时,输出设备不能用于任何其它用途,如使用 Shift prtsc、LPRINT 等。

**3.3** 特别应提到的是 AutoCAD 生成的绘图文件一般采用绝对坐标(如...PU;PA-5699.-3736;PD;PA-4021,-3736;PU;...),这样当文件中含有超出绘图机上实际纸的幅面坐标时,绘图机则不能正常工作,解决办法有两种:1. 换大幅面绘图纸,2 改 CAD 软件中设置,转换成合适的绘图文件。

**3.4** 本方法适用于 HP Draftpro、HI DMP-60 绘图机,EPSON LQ1600 打印机,AutoCAD、EDA 软件包。对于其它外设及 CAD 系统,还请读者参阅有关说明。

## 4 结束语

这种方法简便、实用,而且程序容易获得,可以使各单位更充分的利用现有的 CAD 资源,特别是对需要大量绘图的企业,效果更加明显。

## 5 参考文献

- 1 希望编译:《AutoCAD 10.0 计算机绘图软件包》,海洋出版社,1991.2.
- 2 李振格主编:《AutoCAD 11.0 计算机绘图软件包》,海洋出版社,1991.5.
- 3 席一凡等编:《IBM PC 3.30 计算机磁盘操作系统》,陕西电子编辑部,1987.12.
- 4 《HP Draftpro DXL/EXL Plotters User's Guide》,HEWLETT PACKARD,Inc. 1989.2.
- 5 王建华:“Auto CAD 与其它程序快速交换图形数据方法”,《计算机应用研究》,1992.3.
- 6 《EDA 系统用户手册》,中兴科技实业开发公司,1988.6.

# 染色工艺分布式微机控制系统

内蒙古电子计算中心(呼和浩特 010010)

范德元 常国权 李强 姜向荣 林川宏

**摘要** 本文介绍了条染生产的微机控制系统及其系统功能、软硬件设计。

**关键词** 条染工艺 分布式控制系统 温控

## 1 序言

染色是纺织生产中的关键环节,是一个复杂的物理化学过程。其目的在于使毛条或整批纺织品具有一定的坚固色泽。在染料、媒染剂的选择、染浴PH值等诸多因素都得到保证的条件下,染色温度和上染时间以对羊毛染色的影响至关重要,它们直接影响上染率。根据资料(1)提供的关于不同染料上染曲线分析可知,上染率和温度、时间有严格的对应关系,温度不同,染料上染率不同。上染阶段要求严格控制升温速度。

染色工艺从升温开始,分为升温、保温、降温等多个连续阶段。升温过程使用蒸汽直接加热,降温过程使用冷却水直接冷却。升温时不能太快,否则容易染花,尤其是上染阶段,一定要严格保证升温速度,要求动态偏差在 $\pm 1^\circ\text{C}$ 范围内。

正因为如此,各厂家均精心设计了許多先进实用的工艺程序,要求操作者认真操作,严格执行,如图(1)。

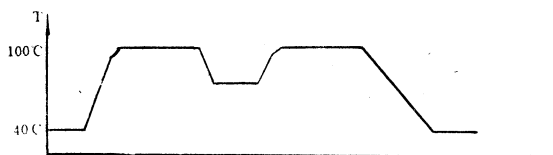


图1 染色工艺曲线

遗憾的是我国在染色机上现大多仍采用人工控制的方法,由于染色现场温度高、湿度大、蒸汽弥漫、酸碱度大,人工操作困难,难于严格执行升温、保温、降温的工艺程序,人为因素影响大。再加上设备陈旧,测量手段落后,以及其他随机因素的影响,染色质量不太稳定,色牢度、色差波动大,染花事故时有发生,造成工时、原料、染料和能源的浪费。

另外,在染色过程中,对于不同的织物染色,如羊毛、丝绸、锦纶、混纺织物染色等,染色过程的升温、沸染、冷却速率和时间不尽相同。如要试验新工艺,寻求最佳染色工艺曲线,以提高染色质量,缩短

染色周期,则需要不断摸索及修改工艺曲线。

## 2 自动控制系统的构成

根据用户要求和实地调研,染色微机控制系统必须满足以下要求:

- (1) 快速跟踪给定温度变化速率,严格遵守升温、保温、降温速率和时间,使各阶段温度控制精度保持在 $\pm 1.0^\circ\text{C}$ 以内。
- (2) 需自动切换蒸汽/冷水调节阀,以实现升温段的升温速率和降温段的降温速率的控制。
- (3) 鉴于溢流漂洗、满水位蒸煮、半水位调温等要求,应对液位能自动进行位置检测,严格控制浴比。
- (4) 对于循环泵和排水阀进行控制,且在其停转或异常时自动报警。
- (5) 由于现场与机房距离较远,必须建立直观、可靠、方便的操作人员和计算机之间的互话接口,便于协调工作。
- (6) 编制简明、标准的染色工控语言级命令,利于操作,提供新产品、新工艺探索的手段。
- (7) 具有断电保护功能,可以适应生产中因特殊事故引起的跨越任意工步继续工作的需要。
- (8) 具备实时显示、打印染色工况参数,实际与给定曲线功能,可以保存多条工艺程序,也可以保存历史信息,以利追踪分析。
- (9) 生成统计报表。
- (10) 具有高的抗干扰性能,可以长期连续稳定运行。
- (11) 便于日后的联网升级。

据此,受控染色机示意图如图(2)。

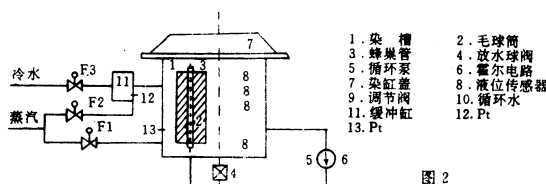


图2

面对现场八台染色机,计算机系统应该具备 A/D16 路, D/A24 路, I/O 出 40 路, I/O 入 64 路。从分散控制, 集中管理的角度出发, 以形成二级实时分布系统为宜, 系统框图如图(3)。

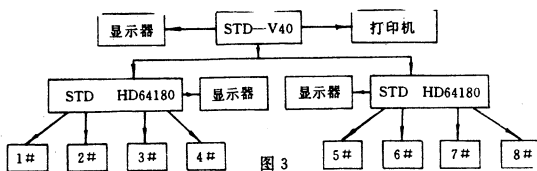


图 3

主控机所以选用 STD-V40 是因为其兼备了个人通用计算机和工业控制机的优点。前者以其丰富的软件资源为软件开发提供了便利, 鲜丽的色彩为创造形象、生动、直观的图形画面准备了条件; 巨大的内存和海量外存存储众多的工艺曲线和历史数据成为现实, 利于质量的追踪分析。后者因其有高的抗干扰能力, 使得可以在复杂的工业环境中连续、可靠的运行。

前沿控制选用 STD 工控机。CPU 为 HD64180, 是八位计算机, 它是专为工业现场应用而设计的, 可靠性高, 抗干扰能力强, 故障下恢复时间很短。每台 STD 工控机直接控制四台染色机的染色过程, 控制系统原理图如图(4)。

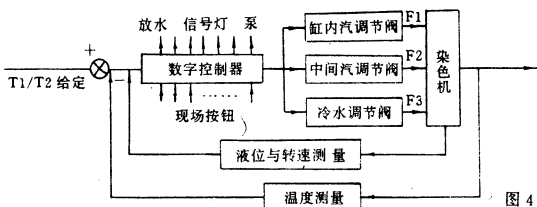


图 4

鉴于温度控制是本系统的核心, 因此在模拟量的输入通道上选用了 12 位 A/D, 以提高温度检测与控制的精度。对于常温染色机, 温度检测的范围为  $+20^{\circ}\text{C} \sim +100^{\circ}\text{C}$ , 所选温变为 0.5 级表, 所用无源 V/I 变化的电阻为锰铜线绕电阻, 其精度为 0.05%。A/D 转换器设计成输入阻抗为  $\text{G}\Omega$  级, 且 12 位 A/D 的分辨率为  $1/(4096-1)$ , 即 0.025%, 根据误差理论分析, 系统的测量误差为

$$\Delta E_n = \sqrt{(0.5\%)^2 + (0.05\%)^2} \approx \sqrt{(0.5\%)^2} = 0.5\%$$

这样做, 意在整体测量精度近似由温变的精度来决定, 即 0.5 级。而温度的分辨率为:

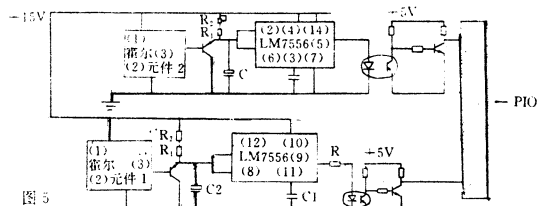
$$(100-20^{\circ}\text{C})/(4096-1) = 0.02^{\circ}\text{C}.$$

这样系统在进行控制决策时, 可以根据温度的细微变化, 改变输出幅度, 使系统温控偏差远远小于

$\pm 1^{\circ}\text{C}$ , 而在  $0.5^{\circ}\text{C}$  以下。

染槽液位信号的测取是控制浴比的重要依据。采用五根一组的下垂式的液位计, 并且选用耐腐蚀性能卓越的银铜条为插针。测量电路的工作电流为微安级, 既保证信号的可靠测取, 又避免了阳极的腐蚀。

循环泵的工作状态监视是避免染花事故发生的重要环节。系统使用两路对称的霍尔元件电路来对其监测, 且为“或”关系。只要有一路发现停泵, 立即发出中断信号, 计算机即自动报警, 对染色机停止加汽, 有效地防止染花事故的发生, 其电路图如图(5)。



### 3 软件设计

软件设计是数字控制器设计的主要组成部分。它以硬件为背景,根据实际被控对象的要求与操作来确定软件方案。

前述染色自动控制系统的种种功能都是依靠软件和硬件的合理配合来实现的。众多的硬件线路和设备在软件的驱动下,完成了本系统的复杂操作。

本系统在软件的功能方面可看作由以下几部分组成。

前沿机的控制程序

前沿机的人机界面程序

中控机的控制程序

中控机的人机界面程序

上述程序的简要情况为:

**3.1 前沿机控制程序** 该部分程序可以同时控制四台染机工作。将现场的温度信号、液位信号、循环泵工作状态等主要信号经过各自的输入通道变成计算机需要的信息。

控制程序依据上述外界信息,按控制算法向每台染机的三个调节阀,发出合适的开度控制信号,保证放水阀、水泵的正确状态,向操作工发出相应的操

作信号,这样便保证了染色过程高精度的逼近工艺曲线。

为了使用户提出的工艺生产曲线为计算机能理解并执行,在本程序中设计并采用了工业控制语言级命令。依据这些命令,任何一个熟悉染色工艺的人均可以设计并输入工艺,并使之执行。

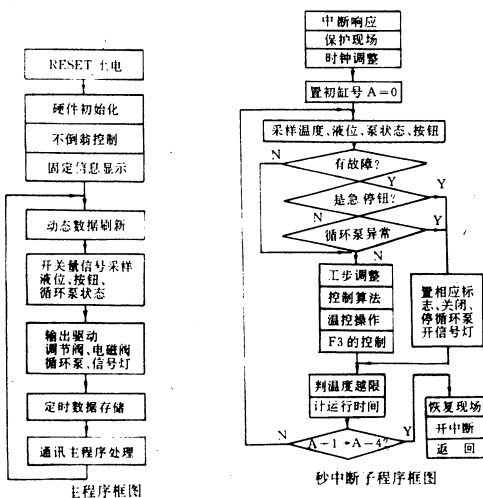
**3.2 前沿机的人机界面程序** 该程序可以随时将采集到的各种信息以人们习惯的方式显示到 12' 绿色屏幕上,还将工作步序、工作时间等显示出来,为操作人员的正确操作提供直观可靠的数据。

该程序向操作人员提供了多种键盘命令,这些命令经由前沿机的键盘输入后即可解释执行。命令是围绕着染色全过程而设置,这些命令主要由启动、结束、强迫停止、跨越工步、设通讯、手动控制、设定时钟等构成。

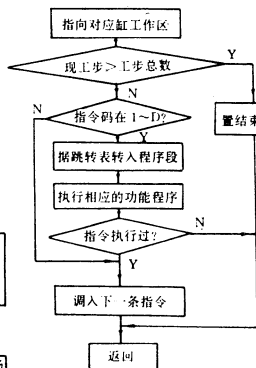
该程序还可以将编制好的工艺表由键盘逐字节输入,提供了设计新工艺的可靠手段。

由于以上程序的功能完善,实际上前沿机是一个完善的独立系统,可单独工作。

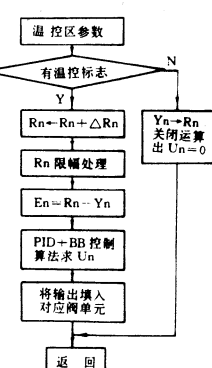
前沿机的程序全部采用了汇编语言编程。部分框图见下图。



秒中断子程序框图



工艺执行程序框图



温控控制操作框图

**3.3 中控机的控制程序** 中控机的程序操作对象主要是两台前沿工控机,在时钟驱动下,不断的从前沿机提取各种“数据块”,并把这些“数据块”变成文件形式。这些数据文件作为显示、打印的原始数据,同时可以保留下来作为历史资料,以及在前沿机排除故障后,作为继续控制的依据。

控制程序由汇编语言以及高级语言交互实现。控制功能是自动实现的,无需人工干预。

控制程序中的一部分是专管异步通讯的。由于

该程序与前沿机的相应程序配合默契,使通讯速率很快,通讯误码率几乎为零,这就保证了分布式染机群控系统有了可靠的支持环境。

**3.4 中控机的人机界面程序** 这部分程序的功能主要由屏幕上的一组功能菜单体现。

由于中控机上的程序均采用模块式管理,高级语言与汇编语言程序之间采用文件方式传送数据,在内存中开辟了一个信箱,用来互相传送运行标志及控制命令信息。这样的结构使程序之间相关性减

小到最小程度,从而大大提高了程序的可靠性及可维护性。

中控机的功能菜单中有几条是直接针对染色过程控制用的,例如启动、停止等等,并为用户提供了直接输入工艺曲线及设计新工艺的功能。也就是说在中控机上的操作可完全替代在前沿机的操作。

另外还有一些功能是前沿机所不具备的,如显示图形、曲线的功能、打印功能。

中控机采用了高分辨彩色显示系统,显示方面主要用高级语言编程,显示具有快速和动感,画面直观、形象。

打印功能采用了彩色打印机,可同时打印网格、标准曲线及实际运行曲线,对照直观、方便,且能长期保留。

#### 4 控制策略

**4.1 对象特性的确定** 染色温度控制系统包括三套对象动态特性,即蒸汽加热使染槽温度升级的过程,冷水冷却使染槽温度下降过程,以及冷水与蒸汽同时在预热器中混合形成 40℃ 的入槽水温。被控量是染槽和预热温度,控制量为蒸汽和冷水量。我们以染槽温度上升过程为例说明测试过程。

我们采用阶跃响应法来求取加热过程对象的动态特性。利用计算机直接输出定值电流来驱动 DQ—I 去推动曲柄绕阀,从而给出一个阶跃的蒸汽信号,然后由计算机记录温度变化的过程。得到如下的阶跃响应曲线。

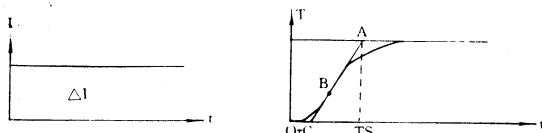


图7 阶跃响应曲线

所得响应曲线呈 S 形,为非周期性自恒过程。我们用惯常的方法,把它分解成一惯性环节与纯滞后环节的串联。其对象的传递函数为:

$$G(S) = [K / (Ts + 1)] * e^{-\tau s}$$

其中:

$\tau$ ——纯滞后时间

$Ts$ ——时间常数

$K$ ——增益

$$K = \frac{[\Delta T / (T_{\max} - T_{\min})]}{\Delta I / (I_{\max} - I_{\min})}$$

其中  $\Delta T$  是对象最终稳定温度值和起始温度值之差。 $T_{\max}$  和  $T_{\min}$  分别为对象输出温度信号的最大值和最小值; $\Delta I$  为计算机输出的电流信号的阶

跃值。 $I_{\max}$  和  $I_{\min}$  分别为计算机输出信号的最大值和最小值。

染槽降温过程的对象特性与加热过程相似,但是  $\tau$ 、 $Ts$ 、 $K$  数值不同,40℃ 水混合过程的对象特性的  $\tau/T$  比前者大些。但不管是哪一种均满足  $\tau < Ts$  的关系。这应该归结为染槽中有循环泵的原因。

**4.2 控制算法** 当有  $\tau < Ts$  时,PID 控制算法可以得到较好的控制效果。我们决定选用这个万能数字调节器。

由于本系统给定值在各个时间段里斜率不同,而且升温段和降温段中控制特性也不相同,另外即便是同一升温阶段,斜率实际上在 40℃~70℃ 再由 70℃~98℃ 的升温时系统本身的热惯性也不尽相同。所以我们选择了分段变参数改进型的 PID 控制算法。为了加快调节过程。外加积分分离。其算式的差分方程式:

$$\Delta U = K_p(e_n - e_{n-1}) + AK_i e_n + K_n(e_n - 2e_{n-1} + e_{n-2})$$

$$U_n = U_{n-1} + \Delta U$$

$$A = \begin{cases} 1 & |e_n| < \epsilon \\ 0 & |e_n| \geq \epsilon \end{cases}$$

由于所用气动曲柄绕阀没有自动保持前次开度的能力,所以最后呈现出位置量  $U_n$ 。

由于这是管道系统,非线性十分严重,我们选择了调节阀工作于等百分比特性,对其进行了补偿;使用了四字节浮点运算,提高了运算精度;采样时间的选取在兼顾  $T < 1/5Ts$  的前提下,并考虑到电气转换器与阀门的滞后时间,我们取其较小值,以求好的调节品质。

对于信号的数字滤波,我们采用连续 6 次采样,去大小,求平均的方法去除高频干扰。综合采用平均值和中值法;对于工频干扰则是在前次滤波有效值测取之后相隔 10ms 再进行一次。

#### 5 结束语

系统在内蒙古第二毛纺厂使用 10 个多月来,具有良好的控制与跟踪性能,工作可靠,抗干扰能力强,操作方便,控温精度高,升温段温度偏差  $2\sigma \leq \pm 0.5^\circ\text{C}$ ,保温段温度偏差  $2\sigma \leq \pm 0.3^\circ\text{C}$ ,完全达到了设计要求,显著提高了染色质量和染机工作效率,节约了能源,已经取得显著的经济效益和技术效益,受到了厂方的好评与肯定。

#### 6 参考文献

- 1《染料应用手册》第三分册 上海市纺织工业局
- 2 陆政道、季新,《自动控制原理及设计》
- 3 刘值、郭木河、何克忠,《计算机控制》

# 一种高速 FAX—PC 接口卡的应用研究

西安电子科技大学电子计算机系 孙俊杰 (710071)

**摘要** 本文所介绍的接口卡是一个用于高速传真机(FAX)与 PC 系列微机之间的通信接口,它可以使 FAX 机兼做微计算机外设,又可以借助传真网实现微机之间、微机与 FAX 之间的远程通信。这里对它的构成及工作原理均做了较为详尽的论述。

**关键词** 传真机 OCR 传真数据 转换技术(FDC) 热敏打印机

## 1 前言

传真机(FAX)可以在两地之间高速地传输文件、图形。而且还具有图象处理、数据通信、复印等功能。这已是众所周知的事情。同时,我们也注意到 FAX 以其低造价(目前国内较低售价为 4 千元左右)、高性能冲击着的每个角落,在我国的应用也是与日俱增,据有关部门统计,目前已达数十万台,八五期间预计达八十万台,九五期间可达二百万台,同时,计算机的应用也相当普及。为了进一步提高办公室自动化的水平,为了在“中文信息处理”中使计算机具有更有效的输入、输出手段,我们开发研制了一种 FAX—PC 卡,其效果较好,而又是简易可行的。

## 2 问题的提出

如前所述,FAX 通信网与计算机的应用,已成为“办公室自动化”的重要内容,而传真机可以高速地传送图文信息,计算机则可以高速地处理信息,如果在两者间实现有机的连接,相辅相成地工作,则可产生更大的效益。我们所研制的 FAX—PC 卡,就是应这一需求而进行的。同时我们还注意到,要使两者真正有机的结合,还要做大量的软件工作,才能实现各种不同的用途。其中一个重要的内容即是“传真—数据转换技术(FDC)”,它可以帮助我们实现人—机(计算机)对话通信,其主要内容是将电码转换为图形码,以及将图形码转换为电码的技术。目前,FDC 技术已为国内外人士所关注。

## 3 传真机的信号

这里传真机的信号是指我们所采用的 FAX 的两组信号,一组是打印头控制信号,我们知道,传真机内部有一个热打印头,它单独构成模块,与主机板间用电缆及接插件相连,它的作用是将图文信号复印在纸上,为此,主机板向它提供必要的信号,其中包括行信号、位信号以及数据信号,在 VF200FAX 中分别叫作 THSTB、THCLK、THSIG 信号。在 NEFAX—22 中,分别叫作 DLATCH、DCLOCK、DDATA 信号,在我们所开发的这两种 FAX 中,其三种信号的形式是完全相同的,其波形如(1)所示。

此外,还有其它一些信号,但我们所需用的仅三

种。从图中可以看出:其位信号和数据信号是一一对应的,且都是间歇的,每两行信号间有一段间歇时间。而行信号则处于每两行之间的间歇期。位信号的频率为 1MHz,行信号的频率为 0.1KC。此外,由于我们是在其接插件上引取信号,所以具有一定的方便性,又由于 FAX 在这几个信号上具有普遍性,因此其应用方式具有一定的普遍性。所采用的另一组信号叫作 CCD 控制信号,在 FAX 中有一个 CCD 模块,也是由电缆与主板相连,在其接插件上取得一组信号如图(2)所示,其中,SAPL 为位时钟信号,频率为 1MHz, $\Phi$ SG 为行信号,表示一行的起始。我们就是利用上述两组信号实现 FAX 与微机的有机连接。

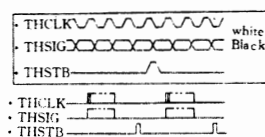


图1 传真机信号图(1)



图2 传真机信号图(2)

## 4 FAX—PC 卡的基本功能

所构成系统具有如下几种基本功能:

- (1) 可做为图文 OCR(光读取设备)使用,将图文原稿以原码方式,以 1MB 高速读入 PC 机,并显示、存盘。
- (2) 将 PC 机内存贮的图文信息或键盘输入的图文信息由 FAX 打印输出,其速度和清晰度均高于普通打印机。
- (3) 由外地 FAX 或 PC 机传来的图文信息,在本地 FAX 和 PC 机接收即可实现 FAX 与 PC 机的通信,也可实现两地 PC 间的通信。
- (4) 本地 PC 机间与 FAX 机的信息,传送到远程 PC 机或 FAX 机。

## 5 FAX—PC 卡的构成与工作原理

FAX—PC 卡的构成与工作原理可分为两个方面来加以说明:第一,做为 OCR 或接收传真网信号时,其原理框图如图(3)。由图中可以看出。



其一端连接 PC 机 I/O 插槽,由于 PC 机系列所具有的兼容性,本接口对于 PC/XT、PC/AT、286、386、486 等都是适用的。本接口另一端与 FAX 中打印头接插件相连,以取得上述所需 FAX 信号。其工作原理是这样的:首先,PC 和 FAX 开机后,由于 8255 处于高阻状态,所以微机和传真机的原有功能均不受影响,可以各自正常工作。只有当 PC 机启动输入程序时(其流程图如图(4)),本接口板才开始工作,第一步运行对 8255 并行口的初始化程序,初始状态参数可置为 92,并送出第一个控制信号,以便选通输入线驱动器;在与 PC I/O 插槽的连接中采用了 AEN 信号,其作用是有效地避免总线竞争的产生。其地址译码部分采用了 74LS30 门电路构成,以便有效地选通本接口,地址的选通宜采用系统闲置地址。

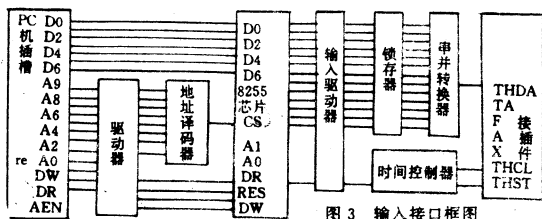


图3 输入接口框图

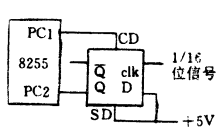


图5 D型门电路原理图

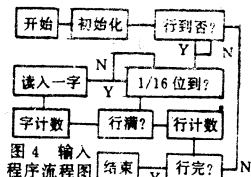


图4 输入程序流程图

其驱动部分则采用 74LS244 即可。初始化以后,程序进入行信号查询方式,其目的是查询 FAX 的第一个行信号是否到达,而这个信号的到达与否取决于如下过程:首先,FAX 机进入复印工作,其行信号即进入 74LS74D 触发器。这个电路的连接方式如图(5)所示,由于这种方式在本接口中还要用到几处,所以我们把它简称为 D 型门电路,行信号的到来使这个电路触发为“1”状态,这个“1”信号再通过线驱动器到达 8255,而程序首先查询的就是这个信号。查得之后,立即发出一个信号,使 D 型门电路归“0”,以待下一行信号的到来,应用此 D 型门电路保证了查询的可靠性。此后程序转入下一级查询,即查询 1/16 位信号是否到达,这个信号的来源如下:位时钟信号(THCLK)首先到达时间控制器部分,这个控制器主要由 74LS293 构成,这是一个二——八——十六进制计数器,它的作用是把 THCLK 信号进行 16 分频,即产生 1/16 位信号,再进入另一个 D

型门电路,再通过线驱动器进入 8255,这就是主程所查询的 1/16 位信号。同样也要进行归“0”操作。然后主程进入读字操作,即通过 8255 的 A、B 口读入一个 16 位字。这里说明一点,这比读字节操作大大提高了工作速度。在一般机器上即可达到 1MB 的数据传送速度。数据字的来源如下:与 THCLK 同步的 THDATA 信号首先进入串并转换电路,这个电路以 THCLK 信号为时钟,以 1/16 位信号为并行翻转信号,可以想见,这个 THDATA 信号就会每 16 位为一组并行进入锁存器,再经过线驱动器到达 8255,这就是所取得的数据信号,主程取得数据后,按一定格式送入内存,并计数。在一行的字未取完的情况下,再继续查询 1/16 位信号,以取得第二个字节,直到把一行的数据取完为止,再进入下一行信息的查询与读取,如此周而复始,直到全文信息的读完。这就是本接口的基本工作原理。采用本接口后,就可以把原稿信息如实地送入微计算机,或在 FAX 接收外地稿件的同时,将信息同时送入微计算机,并可立即显示出来,可以上下,左右滚动。并且,其输入速度远高于一般的扫描仪,其数据的输入速度可达 1MB 以上,而扫描仪的最大传送速度为 9600B。另外,本接口采用原码输入方式,因此,进入微机后无需译码,但可以编辑、压缩、存贮、只要你启动相应软件就可以了。第二,本接口另一个方面的功能是,FAX 兼做热敏打印机使用,以及借助于 FAX,将微机内部文件传送到外地的 FAX 或微计算机。在实现这一功能时,所研制的电路与上述接口在原理上相类似,可以共用同一个 8255,其工作原理基本上是输入时的逆过程,不同的一点是它与 FAX 相连时,所采用的信号不同,这里仅给出其原理框图及流程图(图 6)(图 7)所示。其工作过程则不必介绍了。

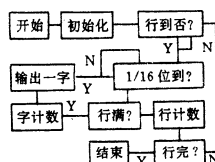


图7 输出程序流程图

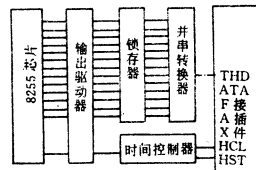


图6 输出接口框图

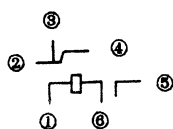


图8 数据通路控制图

- ① +5V
- ② THDATA
- ③ 串并转换器
- ④ 串并转换器

还有两个问题说明如下:一是在 I/O 程序的初

始化模块中,将自动设置 I/O 线驱动器的工作状态,当输入时,只有输入线驱动器处于工作状态,而输出线驱动器则置为关闭状态,当输出操作时,则给以相反的设置。这样,I/O 两部分线路则互不影响;另一点要说明的是在做热敏打印使用时,其 THDA-TA 信号要通过一个小继电器进行控制,其原理图如图(8)所示。在不打印时相通,其数据信号由 FAX 供给,FAX 照常工作,当由微机做输出打印时,相通,其数据信号由 PC 机供给,这个功能的切换也是在 I/O 程序初始化阶段完成的。

## 6 结束语

综上所述,我们可以看出,我们以较简易的方法,实现了 FAX 机与微计算机的有机连接。其效果是令人满意的,其输入输出一页图文的时间仅需十几秒钟,且有相当高的分辨率。同时,微机和传真机的原有功能均不受影响,因此,这是个一举多得的方

案。同时我们还注意到,只要在软件上做进一步的充实,它可以发挥出更大的作用,例如在模式识别、编辑排版方面,在微机数据库的远程传送及管理方面,在民用、军用地图的传送与管理方面都有很高的应用价值。

不当之处,望多指教,欢迎联系。

## 7 参考文献

- 1 村上治(日),《传真机新的通讯设备》,北京,中国铁道出版社,1988.5,65—130
- 2 周明德,《微型计算机接口电路及应用》,北京,清华大学出版社,1981.4,42—79
- 3 周子耀,《传真三类机原理与维修》,北京,原子能出版社,1991.6,21—99
- 4 《Panafax VF—210 Service Manual》,Japan,1989,5—1—7—22

# PC 总线接口中 8255 与 7135 的应用

中国矿业大学 崔亦飞 (徐州 221008)

**摘要** 本文介绍了可编程芯片 8255、7135 在 PC 总线接口中的具体应用方法,并给出了原理图和时序图。

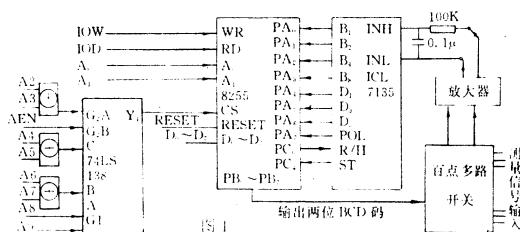
**关键词** 总线接口 可编程芯片 数据采集

用 IBM—PC 总线结构构成的监测系统可直接将采集的数据进行各类复杂的数据处理。该系统用于井下各类非电量信号的监测与分析。使用 PC 机采集数据其数据贮存形式有其通用性,可直接在 286、386 等更高级微机上进行处理。

## 1 系统 I/O 的使用及 I/O 读写周期

数据采集关键在 I/O 接口。解决 I/O 问题首要的问题是 PC 机内存使用分配情况及 I/O 地址的设置,这样才能合理分配,采集程序及采集数据在内存的存放位置,这是关键问题。IBM—PC 机内存采用浮动地址。在内存中的单元号只是程序所在段的内存偏移地址,而非绝对物理地址单元。PC 机中的 8088CPU 采用 I/O 独立编址方式,使用专门的 I/O 指令进行操作,存储器地址和 I/O 接口地址读取指令不同,其 CPU 内部操作也不同,控制的信号线也不同。PC 机用 I/O 方式与外设进行数据交换时,地址线使用低十位,其范围为 000H 至 39FH,此范围内的地址单元,其中大部分口已被微机外设占用,还有一些可供用户使用。本系统使用了 300H 至 303H 这几个地址,这原来是预留留给试验卡的,由于一般 PC 机都无此卡,所以我们选用此 I/O 地址,这些地

址是分配给 8255 接口芯片使用的,其分配如下:300H:A 口,301H:B 口,302H:C 口,303H:控制口。系统使用 PC62 总线插槽的信号是:A<sub>9</sub>~A<sub>0</sub>,AEN、 $\overline{\text{IOR}}$ 、 $\overline{\text{IOW}}$ ,系统译码电路及系统原理见图 1。其中:



A<sub>9</sub>~A<sub>0</sub> 为地址线。AEN 为地址允许线。AEN 为高时,由 DMA 控制器控制地址总线,数据总线及读写线,作为控制 I/O 通道信号,应使用 AEN 信号,而不是 ALE 信号,这是由于 AEN 信号可区分出 DMA 状态。

$\overline{\text{IOR}}$  为 I/O 读信号, $\overline{\text{IOW}}$  为写信号。

I/O 读总线周期时序图见图 2,它由五个时钟周期组成,即在 T<sub>3</sub> 与 T<sub>4</sub> 状态间插入了一个等待状态 T<sub>w</sub>,在 T<sub>1</sub> 状态 8288 送出地址锁存允许信号

ALE, 8088 送出 I/O 地址  $A_0 \sim A_{15}$  在 ALE 后沿将其锁存或缓冲送出, 在  $T_2$  状态由 8288 送出  $\overline{\text{IOR}}$  信号, 该信号在  $T_4$  状态结束。被选中的 I/O 设备地址必须在  $\overline{\text{IOR}}$  信号结束前 (上升沿) 30 多毫微秒前将数据送到数据总线, 并持续到  $\overline{\text{IOR}}$  结束后至少  $10\mu\text{s}$  确保 8088 能取到数据。在  $T_4$  状态, 8088 从数据总线将送来的数据取走。I/O 写总线周期时序图见图 3。图中指出了要写入的 I/O 口地址及发  $\overline{\text{IOW}}$  信号的过程, 从图看出同 I/O 读总线周期一样。只是此时在  $\overline{\text{IOW}}$  信号产生后的  $112\text{ns}$  时间内 8088 已将要写入 I/O 设备口地址的数据送到数据总线, 此数据必须一直保持在  $T_4$  将时钟上升沿开始后  $10\mu\text{s}$  以后为止, 否则 I/O 设备可能取不到完整的数据。在  $T_4$  状态 I/O 设备将 CPU 送出的数据取走。

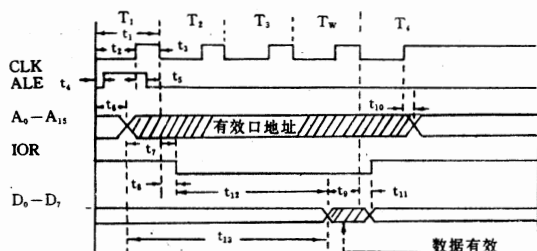


图2 I/O读总线周期

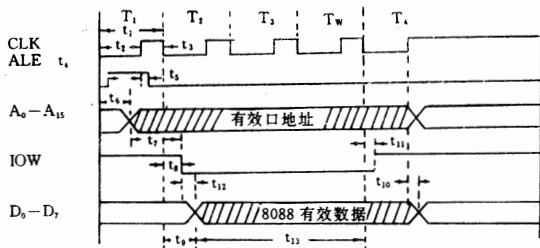


图3 I/O写总线周期

## 2 系统中 8255 与 7135 的使用方法

8255 可编程外围接口芯片是通用并行输入/输出接口芯片, 很适用于作 8088 CPU 的外围接口。它具有三个端口 (A 口、B 口、C 口) 能在三种方式下工作。8255 各口及控制寄存器靠  $A_{15}A_0$  线选择,  $\overline{\text{CS}}$  为片选信号线, 8255 的口地址可由此三条线来确定。我们采用 0 工作方式或 1 工作方式。8255 一方面受 8088 控制与 8088 总线传递数据, 另一方面控制 7135 并采集 7135 输出的数据, 此外还控制多路转换通道, 7135 为高精度 41/2 位双积分式 A/D 转换器, 分时扫描输出 41/2 位 BCD 数据, 转换速率为每秒 4 至 25 次, 输出速率除最高位为 201 个时钟周期

外, 其他为每位 200 个时钟周期, 这样一次输出时间为 1001 个时钟, 这种扫描输出速率对 8088 来说准确采集数据是不成问题的。此外 7135 测量范围在  $0 \sim \pm 2\text{V}$ , 因此温度及应变信号在 7135 前端还应通过相应的放大器放大再进行 A/D 转换, 这里不细述。通道转换的控制使用 B 口, 由 IBM PC 控制, 输出两位 BCD 码, 控制两片 16 选 1 模拟开关芯片 4067, 由此控制 100 组干簧管继电器开关。

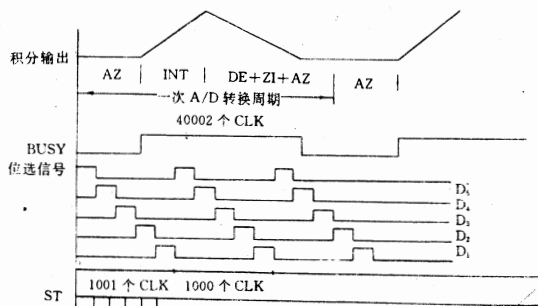


图4

8255 要准确采集 7135 数据, 关键问题是利用 7135 数据输出时的信号:  $D_5 \sim D_7$  为位选通信号,  $B_1$ 、 $B_2$ 、 $B_4$ 、 $B_8$  为数据 BCD 码输出。位信号出现时的 BCD 码为该位有效数字。此外每次 A/D 转换完毕输出的第一组数据在输出正脉冲中间有一个  $\overline{\text{ST}}$  发出的负脉冲, 这个信号在数据采集中是可以利用的。

在不用  $\overline{\text{ST}}$  信号只使用数据线查询电平时, 其接线如图 1。图中  $\text{PC}_4$  接  $\overline{\text{ST}}$  可以不用。这样采集数据的方式是查询方式, 当查询到出现位信号时, 将数据采集出来送内存。其参考程序如下:

```
MOV AL, 90
MOV DX, 303
OUT DX, AL
MOV AL, 0F
MOV DX, 302
OUT DX, 300

LOOP1: IN AL, DX
TEST AL, 10
JZ LOOP1
MOV (200), AL

LOOP2: IN AL, DX
TEST AL, 10
JNZ LOOP2
MOV (201), AL

LOOP3: IN AL, DX
TEST AL, 20
JZ LOOP3
```

```

MOV (202),AL
LOOP4: IN AL,DX
TEST AL,20
JNZ LOOP4
MOV (203),AL
LOOP5: IN AL,DX
TEST AL,40
JZ LOOP5
MOV (204),AL
RETF

```

而在某些情况下必须使用 $\overline{ST}$ 线。当 $R/H$ 发一个宽度大于 300ns 以上的正脉冲后,查到 $D_5$ 的高电平后,查 $\overline{ST}$ 线信号, $\overline{ST}$ 信号只有 1/2 个 A/D 时钟周期,通常在 5 $\mu$ S $\sim$ 7 $\mu$ S 左右,直接查 $\overline{STB}$ 线往往查漏,而采用 8255 工作方式 1 时需要此信号。方式 1 下的输入时序如图 5, $\overline{STB}$ 为选通输入信号线,IBF 为输入缓冲信号,在输入状态下,为接收到 $\overline{ST}$ 信号将 $\overline{ST}$ 接 $\overline{STB}$ 线,当 7135 转换周期结束时, $\overline{ST}$ 的低脉冲向 8255 发出请求读信号 $\overline{STB}$ ,将数据锁存在 8255 的 A 口,此时 IBF 线为“1”,8088 CPU 查 IBF 为“1”时,确认缓冲器已有数据,这样避免查询 $\overline{STB}$ 而将输入

缓冲器信号作为标志位使用,控制 CPU 的读取,而不控制外设送数。因为外设送数速度较 CPU 取数慢得多,大约在 200 个时钟周期 2ms 左右送一位数字。IBF 也没有合适的控制信号,因为 A/D 一组信号的 5 个 $\overline{ST}$ 信号是连续发出的,无等待联络,因而只有将 IBF 作为状态位查询。IBF 为“1”说明缓冲器数据满,发 $\overline{IOR}$ 信号,8255  $\overline{RD}$ 接到此信号,数据送入 8088 CPU, $\overline{RD}$ 信号结束时,IBF 复“0”,准备接收下一组数据。

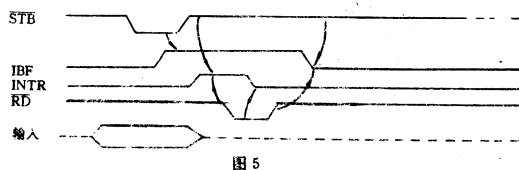


图 5

从上述分析可以看出 8255 采用工作方式 1,对 7135 来讲,信号的可靠性大大提高。由于现场试验所测信号是井壁压力变化及冻结井壁温度变化,信号变化是缓慢的,根据这一实际情况,7135 采集速率(每秒 4~25 次)是可以满足的。

(上接 11 页)对数据结构的描述;而类对数据结构及施于其上的操作进行统一描述,这充分体现了抽象数据类型思想。另外,类通过继承将类和类之间有机地联系起来,从而达到代码的共享与重用,而类型并无此种机制。

人们开始寻求将面向对象的设计思想应用于传统的程序语言上,来构筑良好风格的程序。虽然人们习惯于自己固有的程序设计方法,一下子很难适应一种全新的方法。我们可以采用以下手段来实现面向对象的思想:模块 $\Rightarrow$ 类、记录 $\Rightarrow$ 实例、过程 $\Rightarrow$ 方法、过程调用 $\Rightarrow$ 消息传递。通过这些联系,就可以把良好的传统程序设计风格用于面向对象方法。当然,这都不是简单的对应,例如,模块应分为定义和实现两部分,定义性模块描述了类的外部特征,实现性模块描述其内部实现。

#### 4 小结

面向对象的程序设计方法受到人们越来越多的重视,它具有许多明显的优点,正在冲击着传统的程

序设计风格。当然,面向对象程序设计也有其不足之处,只有把面向对象程序与面向函数的程序设计和面向逻辑程序设计三者有机地结合起来,才能相互补充,达到完善的结合,这些都是值得进一步探讨的。

在本文的形成过程中,承蒙徐德启付教授的仔细审阅并提出宝贵意见,特此致谢!

#### 5 参考文献

- 1 Peter Wenger, "Ada 的成就与不足",《计算机科学》,88. 1.
- 2 朱海滨, "面向对象方法学的研究",《计算机科学》,90. 5.
- 3 麦中凡译,《C++ 程序设计语言》,清华大学出版社。
- 4 Brad J. Cox,《Object-Oriented Programming》,1983.
- 5 Sally Shlaer, Stephen J. Melor,《Object-Oriented Systems Analysis》.

## 一种高性能游戏机控制器

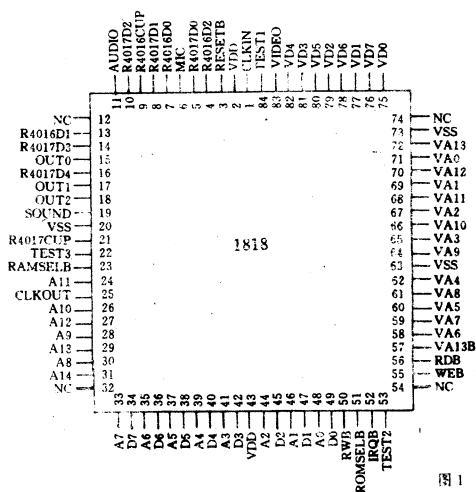
西北师范大学计算机科学系

王让定 王小牛(兰州 730070)

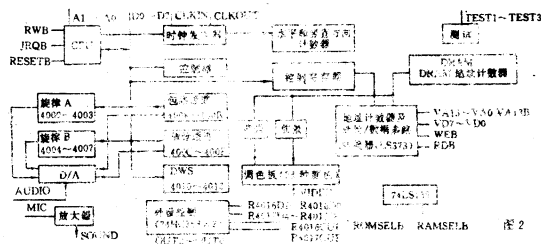
**摘要** 本文介绍了一种新型的高集成度的游戏机控制芯片。用该芯片开发的游戏机体积小,重量轻,且兼容于任天堂系列游戏机。

**关键词** 集成化芯片 控制器 游戏机

一般的游戏机主要由 CPU 6527、PPU 6538、译码器 74LS139、三态缓冲器 74LS368、锁存器 74LS373、PRAM 6116、VRAM 6116 及附加电路组成。由于芯片多,线路复杂,对游戏机的性能有一定的影响。最新研制的集成化芯片 1818 可使游戏机的开发大大简化,该芯片集成了游戏机中除 VRAM 和 PRAM 以外的几乎所有芯片和必须的电路,因此采用该芯片,只需少量的线路,便可构成一台任天堂系列游戏机。下面对 1818 做一简单的介绍。



采用 1818 构成的游戏机系统框图如图 3 所示:



## 2 功能

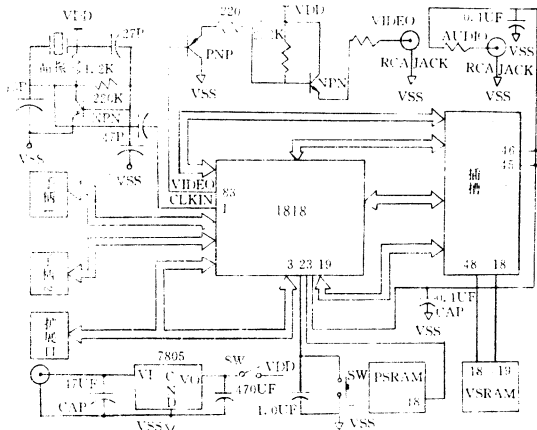
1818 集成了一片兼容于 6502 的 8 位 CPU、一个声音处理器、一个图像处理器 PPU (picture processor unit) 及外部地址译码器,集成度高,性能稳定,具有很强的功能,下面一一说明:

8 位 CPU 包括 16 位程序计数器、8 位 ALU 和累加器、状态寄存器、两个通用寄存器 X 和 Y、8 位堆栈指针、16 位地址总线和 8 位数据总线。该 CPU 为 PPU 提供了 256 字节的 DMA 功能,为手柄 1、手柄 2 及扩展口提供了两个 I/O 口 (\$4017H 和 \$4016H),一个三位的控制口 (OUT2~OUT0)。

声音处理器包括两个旋律通道、一个噪音通道、一个包络通道和一个数据波形合成器,每个通道都有四个口控制其操作。

PPU 能控制前景和背景两种目标的显示,前景是如汽车、子弹、人等可以移动的目标;背景是如森林、房子、可以滚动的图像等大的目标。PPU 在电视屏上控制显示的分辨率为  $256 \times 240$ ,一屏可以显示 64 个前景,每个前景至少需要 4 个字节,在水平扫描行上一次最多显示 8 个前景,超过 8 个以后的图像帧将被丢失并返回信息给 CPU。基本的前景或背景单元是一个  $8 \times 8$  点阵的数据帧,每个点阵可以显示 16 种颜色,前景有  $8 \times 8$  点阵和  $8 \times 16$  点阵两种。有两页背景的图像,无论在水平还是垂直方向上,页与页之间都可以迅速的切换或滚动;调色板可以定义 28 种颜色,每种颜色的定义需要 6 个 bit 位。自动电视同步信号独立于程序。视频合成输出信号有 NTSC 和 PAL 两种制式。

## 3 系统框图



## 4 系统时钟

1818 芯片的系统时钟有 NTSC 和 PAL 两种制式,CPU 及 PPU 在这两种制式下的操作时序如图 4 和图 5 所示:

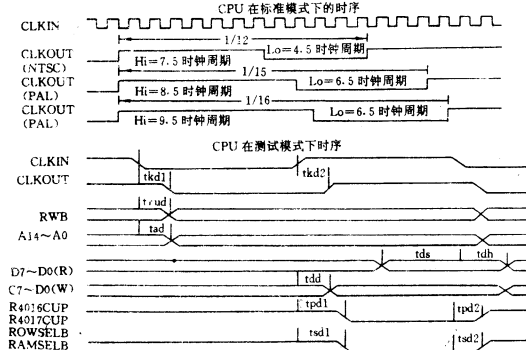


图 4

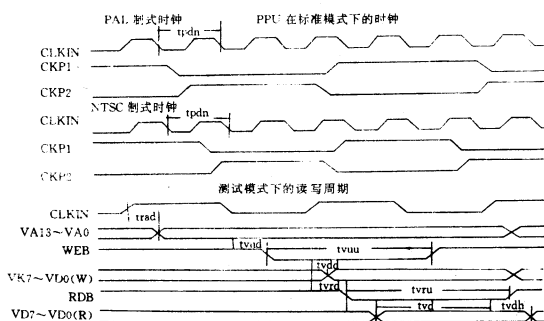


图 5

## 本刊启事

为了更快地推动、促进我国计算机产业的迅猛发展,及时地为计算机科研、开发、生产等部门提供计算机行业的最新发展动向,加速高新尖端技术、最新软硬件的开发、移植、引进,及时地为广大读者及计算机爱好者奉献更多更新的计算机专业技术资料,并使众多作者、译者脱颖而出,《计算机应用研究》杂志社现在开展优惠出版各类计算机专业技术资料业务,欢迎广大作、译者踊跃赐稿。具体出版业务欢迎来函来电来商议。联系人:张钢(邮编:成都 610041)

《计算机应用研究》杂志社启

## M2024 打印机常见故障的修复

黑龙江大学 陈永红 穆大明 (哈尔滨 150080)

**例1:故障现象:**开机后,除风扇转动外,无其它动作,面板上所有的指示灯均不亮。

**故障的分析及排除:**打开机壳,发现+36V 电源部分的保险管 F3(4A)烧断,从插座 J<sub>11</sub>上取下接口板,除电源插头 P<sub>10</sub>外,取下主板上的其他插头,用万用表测+36V 和 0V 之间的阻值,出现短路。打开电源开关,+36V 和+5V 都无输出。从电源部分分析,因击穿能直接导致+36V 短路的元件有:整流管 DB2、可控硅 SCR2 和 C<sub>47</sub>,经逐个检查无异常情况,初步断定部分正常,短路可能发生在控制和驱动电路。在控制和驱动电路中,使用+36V 电源的有:走车和换行两电机的驱动电路、打印头针驱动电路以及打印头保护电路。由于保险管 F1(2.5A)完好,故电机驱动电路正常;在打印头保护电路中,能直接造成电源短路的可能是起保护打印头线圈作用的差动 M18K(≠G7),经检查确定 M18K 正常;而在打印头针驱动电路中,从原理图上分析,这部分没有能直接造成+36V 短路的元器件。最后,把各部分的+36V 电源线彼此断开来检查,经进一步查找,发现这部分的+36V 和 0V 之间有一小电容 C<sub>10</sub>被击穿,而这只小电容在原理图上没有标注,断开电容 C<sub>10</sub>,打印机恢复正常工作。

**例2:故障现象:**同上。

**故障的分析及排除:**由于接通打印机开机后,可听到风扇的旋转声,所以证明交流电流已供电,而指示板无显示,打印头不动,可见问题出在打印机+5V、+36V 电源供电部分。打开打印机后盖,用电表测量在打印机+5V、+36V 两端,发现均无电压输出。由于打印机的内部工作就是靠+5V、+36V 供电来完成,现在+5V、+36V 没有输出,所以指示板无显示,打印头不动作。再检查+5V、+36V 供电线路,发现 F1(2.5A)保险管、F4(4A)保险管烧断,换上新的保险管后,打印机工作正常。

**例3:故障现象:**开机即烧毁保险管 F3。

**故障的分析及排除:**保险管 F3(4A)是保护+36V 电路的。+36V 给打印电路、升行及字车电路供电。保险管 F3 开机即烧毁,说明这几部分电路或+36V 电源的变换、控制部分的某处有短路发生。为迅速确定短路点,依次拨去保险管 F1(2.5A),断开升行及字车电路;再拨去打印头供电插头,仅剩下电源部分,加电保险丝仍烧毁。很明显,短路点在电源部

分。经过反复分析,最后按电路板全面检查电源电压。结果发现,电容 C<sub>70</sub>(50V 4.7μ)短路。C<sub>70</sub>与电容 C<sub>47</sub>并联,共同对+36V 电压提供滤波。更换 C<sub>70</sub>,打印机恢复正常工作。

**例4:故障现象:**开机后所有指示灯均亮,但打印头不动。

**故障的分析及排除:**当打印机开机以后,首先要测试 B534007 接口板上四块芯片(6116)的好坏,当有一块芯片坏时,打印机即处于停机状态,所有指示灯均亮。因此,先检查 RAM 芯片的好坏。通过测量,所有芯片均正常,接着检查与其有关的外围电路,当检测到#B1、#C1(型号 SN7406N,为六倒相缓冲器)时,发现这两块芯片已被击穿,导致 RESET 信号持续输出,从而使指示灯发亮,更换同型号的两块芯片后,打印机工作正常。这种故障主要是由于带电插拔电缆线或数据传输线引起的。

**例5:故障现象:**打印机一直处于缺纸状态,虽装上打印纸,缺纸灯仍亮。

**故障的分析及排除:**对 M2024 打印机和其他大多数 24 针打印机,检查是否缺纸靠的是装在字辊下部的一个光电传感器来完成,正常时传感器发的光经纸面反射后被传感器接收并产生出一电讯号,给出有纸信息,对这种故障,大部分是由于长时间的用机且不清洁,致使灰尘等污物玷污传感器表面,清洁传感器表面后即可排出故障。然而,由于打印机长时间使用,不可避免地有时就会出现传感器已坏的情况,若当时又没有适合的传感器更换,这里介绍一种解决该故障最简单的办法。在 M2024 打印机和其他多数 24 针打印机中,除该传感器可检测有无纸外,在后部的盖子下方(通常是打印机靠孔输进纸的地方),还有一个接触式探纸机构,可以用胶带把该机构粘住,这样一来,就不会再出现缺纸信号了。但用该法处理后,不论打印机有纸与否,均按有纸处理,所以使用打印机前一定要注意装好打印纸。

**例6:故障现象:**打印时自动走纸滚筒工作正常,但纸不能前进。

**故障的分析及排除:**可能是压纸弹簧过硬,或压纸轮损坏。拆下打印机操作面板,拆下打印机走纸驱动滚筒,发现压纸轮因受高温腐蚀或其它方面影响而严重变形(该压纸轮是用橡胶材料制做),当将 clever Position 置 close 位置时,压纸轮被卡滞在引



导走纸板的导向槽内,不能转动,将纸卡死。更换压纸轮。由于压纸轮是用橡胶材料制做,因此如果市场上暂时买不到压纸轮,可将压纸轮拆掉,用小刀将变形凸起部分削掉,但要保持压纸轮为圆柱形且光滑。用上述方法处理后,安装好走纸驱动滚筒和操作面板,即恢复良好。

**例 7:故障现象:**打印机工作时,无论是打印中文还是西文,都发现字迹不清晰,有时是白纸,听得见打印针动作的声音。

**故障的分析及排除:**可能是打印针损坏,针头固定螺钉松动或针头距打印滚筒太远。打开打印机操作面板上的活动玻璃,拆掉色带盒,检查针头固定螺钉的固定均良好。松开针头固定螺钉,用手将针头向前推靠近打印滚筒并顶住不松,拧紧针头固定螺钉,装上色带盒和玻璃板,即恢复正常。说明是打印针头距打印滚筒太远。

**例 8:故障现象:**打印汉字时笔划残缺不全,自检打印时字符的字型正常,但点阵稀疏,其密度约只有正常字符点阵的一半。

**故障的分析及排除:**我们知道 M2024 打印机的奇数针和偶数针(各 12 根)分别由 ODDENB 和 EVNENB 两脉冲信号控制,ODDENB 信号有效时,选通偶数针,EVNENB 信号有效时,选通偶数针,若其中一个信号不正常,就可能造成一半针不工作,所以首先检查 ODDENB 和 EVNENB 信号。在打印过程中用逻辑笔测试上述两脉冲信号,其中 ODDENB 端有规律地出现脉冲,而 EVNENB 端测无脉冲。由电路分析可知,ODDON 和 EVNON 信号控制的,ODDON 和 EVNON 经单稳电路 #A1(74LS221)产生足够宽的脉冲信号,然后由晶体管 TR2 和 TR1 放大后输出。因此,既然 EVNENB 端无脉冲,首先用逻辑笔检查 EVNON 端,测试结果正常(有脉冲),再测试三极管 TR1 的基极,结果无脉冲,故初步判断故障可能出现在 #A1 芯片及周围的元件上(如 C1,VR1,RA2 等)。更换一片 #A1,故障仍未排除,然后换电容 C1,测试 EVNENB 端有脉冲出现,打印也正常。再用数字表测 C1 的电容,几乎为 0,所以故障是由 C1 电容失效所引起的。由于 C1 电容失效,使 #A1 的 12 脚 Q 无脉冲输出,造成打印机偶数针不工作,换 C1 后故障排除。

**例 9:故障现象:**打印出的文字中有一条连续黑线,自检时印出的字符上黑线仍存在,从字符上方算起大约在第 8 个点的位置。

**故障的分析及排除:**为查找故障原因,与另一台同样的机器互换打印头,上述故障仍未消除,这就排除了打印头本身损坏的可能。接着查找打印头驱动

电路,由电路原理图可知,M2024 的打印锁存信号(PDA1—24)直接接到 6 个打印头驱动器 M5467P,每片驱动器驱动 4 根打印针,既然初步判断第 8 根针工作不正常,首先检查 #G6 芯片,因为 2、4、6、8 号针都是由此芯片驱动的。以对比的方式测试 #G6 的四个输出端,发现第 8 根针的驱动端 Q<sub>4</sub> 对地电阻特小(只有 1 欧姆),而其它端则大得多。将 Q<sub>4</sub> 端空后继续测试其阻值仍为 1 欧姆,所以怀疑 #G6 内部损坏,更换一片 M5456P 后,故障现象消失。由于第 8 根打印针的驱动电路损坏,使相应的驱动线圈总保持适当的电压,因而造成 8 号针一直处于工作状态,从而打印出一条连续的黑线。

**例 10:故障现象:**联机打印时,打印出几个杂乱字符后,一直跑纸。

**故障的分析及排除:**首先用好的同型号打印机取代原打印机,故障消除,说明原打印机有问题。对原打印机进行自检,以区别是主板还是接口板的故障,经过自检正常,说明问题出在接口板(B534007)上。通过检查发现 J<sub>11</sub> 的第七脚始终是低电平,分析电路图可知有两种可能,一是电容 CA<sub>2</sub> 的 7~1 击穿,二是 #C<sub>2</sub>(M5324P)的 1~2 这个非门损坏。将 CA<sub>2</sub> 焊下,用万用表测量 7 脚到 1 脚之间阻值为无穷大,CA<sub>2</sub> 是好的。问题只能出在 #C<sub>2</sub> 上,将 #C<sub>2</sub> 换下,发现 1 脚与 7 脚之间短路,而 #C<sub>2</sub> 的第 7 脚是地,所以使得 J<sub>11</sub> 一直处于接地状态,更换后故障消除。

**例 11:故障现象:**当 Control-P 打印机被启动后,打印一行后就连续走纸,但打印机的指示灯均正常。

**故障的分析及排除:**将主机与打印机各自进行自检,均工作,打印正常。用替换法检查是打印机适配器的还是打印机的问题,换一只好的打印机适配器,M2024 仍然联机不打印,并且死主机。将 M2024 也换一台,联机打印才正常。这说明打印机适配器和 M2024 打印机均有故障。经检测发现打印机适配器板上的 U8(74LS240)反码 3 态输出缓冲器的第 2 脚接地,使“BUSY”信号不正确,更换 U8 后打印机适配器的故障排除。检测 M2024 打印机各接口信号线的静态电平,发现打印机的“BUSY”信号也不正常,总为高电平。查为打印机的 B534007 接口板上发送给主机的“BUSY”的信号不正确。当打印机不处于“忙”状态时,BUSY=0;当打印机处在“忙”状态时,BUSY=1。由于 B534007 板上的 #A3(LS02)与 #B1(LS05)集成电路坏,此时的 LS05 的 10 脚总为“1”,表现为打印机没有准备好,所以主机的数据送不出来。更换 #A3(LS02)和 #B1(LS05)故障排除。

**例 12:故障现象:**打印机自检时前面的部分字符打成黑块,后面的字符和汉字则打印正常。

**故障的分析及排除:**因为打印机能进行自检,说明监控程序得到了执行。自检打印时部分字符点阵成了一个黑块,这好似在安装某些软字库时由于没有将全部字符点阵装入硬盘中,而又要打印全部字符点阵,就会打印出一些黑块一样,说明有些字符点阵缺少。由于打印的黑块是在前面,且汉字打印正常,因此,可以确认是固化基本字符集的 ROM 芯片 IC24(27512)损坏,更换后故障排除。

**例 13:故障现象:**打印机开机后就不停地走纸。

**故障的分析及排除:**当打印机驱动程序与实际安装的打印机类型不相符时,打印机打印时就会无休止地走纸。本例的故障可能是固化打印机自检驱

动程序的 ROM 发生了问题。监控程序是固化在 IC25 中的,因此,是 IC25 中的程序坏,更换 IC25 后故障排除。

**例 14:故障现象:**打印时字符缺划严重。

**故障的分析及排除:**静态检查 TR13~TR28 正常,用万用表测 VH1~VH4 三个高压发现 VH2 无电压指示。进一步检查见到保险管 F202 损坏,换一个 3.15A 的保险管后故障排除。同样,保险管 F203 和 F204 熔断后也会出现类似的故障现象,因为 VH2、VH3、VH4 三个高压是提供给针驱动三极管集电极作为工作电压的。则对应的针驱动三极管就无法正常工作。因此打印时字符要缺划。而如果是保险管 F201 熔断,则公共三极管的工作电压就全部被切断了,这样打印头根本无法出针打印。

## 微型计算机开关电源的维修

江西师范大学计算机系

万承兴

(南昌 330027)

微型计算机中使用的开关电源种类繁多,功率小的用自激式开关电源,采用单管自激振荡方式,这种电源线路简单一些。功率大些的多用双管半桥式开关电源,开关管的导通和截止受触发信号的控制,因而增加了控制电路,这部分电路的好坏直接影响电源的工作状态。在控制电路中,有的给 IC 控制芯片加了辅助电源,有的不带辅助电源,IC 芯片(像 TL494C)的工作电压+V<sub>cc</sub>是直接由+12V 加上的,开关电源的启动是由电源开启时产生的冲击电压使开关管振荡,输出+12V 电压为 IC 控制芯片提供电源,控制芯片又产生触发脉冲,维持开关管的振荡和电源正常工作。

对于控制电路的检查,主要检查 IC 芯片的功能,对有辅助电源的 IC 芯片,可利用本身的电源,在辅助电源正常的情况下,断开开关管集电极电源加电检查;对无辅助电源的 IC 芯片,可用外加电源+12V 检查,对于 TL-494C 芯片,把第 4 脚和线路断开,把+12V 电源加到 12 脚,在 14 脚应有+5V 输出,8 脚和 11 脚应有相位相差 180°,幅度为 2V 左右的方波输出,周期为 20<sup>μs</sup>到 40<sup>μs</sup>,否则 IC 有故障。

有辅助电源的开关电源,对开关管的放大倍数要求不高,一般大于 10 即可,而没有辅助电源的开关电源,要求开关管的放大倍数大于 20,一般为 30 倍左右,且两个开关管要尽量配对,即放大倍数要求一样,否则,开关电源不能正常工作。

我们在维修工作中,碰到这样几种故障:

### 1、开机就烧保险丝

这种情况一般为短路故障,常见为整流二极管,滤波电容和开关管短路,只要查出短路元件,换上好的,电源就能正常工作。

2、电源因输入交流电压过高烧坏开关管和推动管,无电压输出。

对于带辅助电源的,检查出损坏元件,换上好的开关管、推动管等元件之后,电源就能正常工作,而对于没有辅助电源的,在换上好的开关管之后,有时仍不能启动,对有关元件逐个查找,也没发现异常现象,加电检测开关管集电极有 300V 直流电压,在量开关管基极电压时,电源启动工作了,输出电压亦正常。对电路原理进行分析,电源是靠开机时的冲击电压起振的,如果开关管电流放大倍数小,开机时的冲击电压就不能使其振荡,因而没有输出,而用万用表测开关管的基极电压,相当于给开关管一个触发信号,因而使其振荡,输出+12V 电压使控制电路工作。换上电流放大倍数大的配对开关管后,电源工作正常。

3、微机在使用中没接稳压器,因交流电压大的波动,造成主机不能工作。

对电源检查,发现直流+5V 只有 4.6V,低于其要求的最低值 4.75V,使主机集成芯片不能正常工作,打开电源盒,对输入电路进行检查,测得滤波电容有一只漏电,换上相同型号的电容后,故障排除。

## IBM AT 微型计算机直流电源故障一例

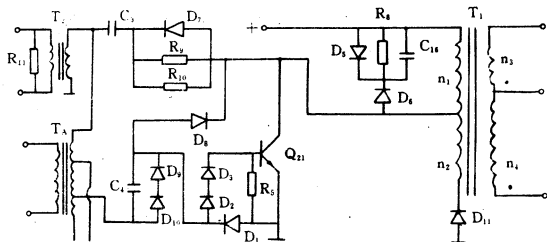
林业部白城林业学校

赵明生

(吉林省白城市 137000)

**故障现象:**经常烧该机直流稳压电源中的功率开关管 Q21。

**检查与分析:**该机中的功率开关晶体管 Q21 与高频变压器 T1 及相关元件组成它激式脉宽调制变换驱动线路(如附图)。Q21 的好与坏直接影响 5 伏和 12 伏的输出值,若 Q21 击穿则可造成无输出现象。



IBM AT 微机直流稳压电源中的直流变换器驱动部分线路(局部)

本例,在主机使用中突然停止工作,开机后测直流电源输出插头无 5 伏及 12 伏电压。说明直流电源出现故障。拆开电源外壳详细检查发现开关管 Q21 集电极与发射极之间击穿。取一同型号的晶体管换上后,一切正常,但没用多久又出现此类现象,经查又是 Q21 被击穿。这种情况一般说来是因过载引起过流或 Q21 的保护元件损坏失去保护作用所致。

保护 Q21 不被击穿的电路及元件主要有:

①为了避免因过流而烧 Q21 管,在高频变压器

的原端设有保护网络,高频变压器 T1 和脉宽调制组件 TDA1060 是主要器件。

②若 5 伏或 12 伏直流的输出端任何一路有过流现象时,D6 和 D3 及 TDA1060 是主要保护 Q21 不被击穿的器件。

对于①和②两方面都是通过 TDA1060 使输出的调制脉冲宽度变为零而实现保护作用的。

③在直流变换器驱动线路中,并联在高频主变压器 T1 原边绕组上的 D5 和 D6、电容 C16 和电阻 R8 以及连接在 Q21 集电极上的 R10、D7、C3 组成一个防止晶体管 Q21 的集电极上出现两次电子击穿所设置的消振缓冲电路。D8 和 D1 是分别用于抑制可能出现在 Q21 的集电极与发射极之间的感应反偏电压幅度过大而设置的保护二极管。

根据以上分析,首先检查了过载情况,没有发现问题,所以上述①、②中的元件不必测量。只对③中的 D5、D6、D8、D1、C16、C3、R9、R10、R8 进行逐个测量,在测试中发现 D1 已经开路。由于 D1 开路,破坏了由 D1 和 D8 构成的保护网,当电网电压波动幅度较大时或由于其它原因,可能引起 Q21 的集电极与发射极之间反偏电压增大。当反偏电压超过规定值时可将 Q21 击穿。更换 D1 后再更换一个新的功率开关晶体管 Q21,直流稳压电源恢复正常。至今未坏。

推广应用开发研究计算机的良朋益友

传播计算机产品广告信息的忠实媒介

欢迎订阅《计算机应用研究》杂志

●刊号 CN51-1196/TP  
ISSN1001-3695

●邮发代号 国内:62-68  
国外 BM 4408

●全年每套订价:10.8 元(国内)  
\$ 12 元(国外)

●全国各地邮局均可订阅

●国外总发行:中国国际图书贸易总公司