

微小型计算机



开发与应用

MICRO—MINICOMPUTER
DEVELOPMENT & APPLICATION



1991 2

微小型计算机开发与应用编辑部

微小型计算机开发与应用（公开发售）

编辑：《微小型计算机开发与应用》编辑部

发行：天津市邮局

出版：天津市电子计算机研究所
天津市计算机学会

印刷：天津武清县长宏印刷厂

地址：天津市河西区友谊路宾馆南道5号

订购处：全国各地邮局

邮政编码：300061

定价：0.95

邮局代号6—87

津工商广字0146号

国内统一刊号CN12—1122 ISSN1001-8786

《微小型计算机开发与 应用》编辑委员会

顾问 郭平欣
主编 黄侃
副主编 王治宝 张凤枝
委员 (以下按姓氏笔划排列)

于万源 于清汶
王治宝 王镭
王士禧 王寿松
付园明 许镇宇
朱植松 曲庭维
李凤祥 刘连棣
陈力为 吴锦声
房家国 张凤枝
夏纪寅 夏业勋
袁维本 曹东启
黄侃 黄宝良
章渭臣 梅克定
童宣明 裴少峰
薛大中

1991年第2期目录

(总第52期)

计算机软件计

- 地震信息图象动态显示的实用软件
.....王孝铨(1)
一种新颖的办公自动化工具——电脑
传真排版系统.....张群(5)
“大麻”病毒综析.....孙德坤 蔡思伟(15)
UNIX/XENIX文件系统结构与误删除
文件的恢复.....刘长江(20)

应用实例

- 远距离多点大规模数据采集系统设计的研究
.....傅景义 贾景云 耿仁义 陈熙亮(23)
微机在化工设备工艺性补图中的应用
.....华振明(26)
打印机图文输出的实现.....黄明(30)

经验点滴

- 非标准硬盘驱动器的使用.....陈庆章(35)
一个专为清洗盘洗磁头设计的程序
.....张胜(42)
中英文电子打字机与微型机联机通信
.....奚忠方(44)

信息动态

- Microsoft的当前技术动态和未来发展战略
.....张一鸣(46)

网络技术

- 国产PC机如何联入3⁺以太网.....林鑫(47)

CONTENTS

SOFTWARE

The Utility Software for Dynamic Display of Seismal Picture	Wang Xiaoxian (1)
A Novel Tool of Office Automation-Facsimile Composing System Using Computer	Zhang Qun (5)
A Summary of "MARIJUANA"	Sun Dekun, Cai Siwei (15)
The Structure of UNIX/XENIX File System and the Restoring of Wrongly Deleted File	Liu Changjiang (20)

APPLICATION EXAMPLE

A Study on Designing the Remote Multi-Point Large Scale Data Acquisition System	Fu Jingyi, Jia Xueyun, Geng Renyi, Chen Xiliang (23)
Application of Microcomputer in Making the Drawing for Standard Parts of Chemical Industry Equipment	Hua Zhenming (26)
Implementation of Graphic & Character Output by Printer	Huang Ming (30)

EXPERIENCE

Installing a Non-Standard Hard Disk Drive	Chen QingZhang (35)
A Practical Program Designed for Cleaning the Disk	Zhang Sheng (42)
Network Communications between Chinese-English Electronic Typewriter and Microcomputer	Xi Zhongfang (44)

INFORMATION AND TRENDS

New Products and Future Development Strategy of the Microsoft	Zhang Yiming (46)
---	---------------------

NETWORK

How to Install the Personal Computer Made in China to 3 ⁺ Network	Lin Xin (47)
--	----------------

地震信息图象动态显示 的实用软件

福建省地震局地震研究所 王孝铤

摘要 本文介绍地震信息图象动态显示软件的总体结构和基本功能, 论述和探讨了程序设计中的—些实用方法和技巧。

一、问题的提出和本软件 目标

地震预报至今仍是一个科学难题, 处于探索阶段。自1966年邢台大地震发生以来的廿多年间, 为了探求地震孕育、发生和发展的规律, 我们积累了大量十分宝贵的资料。在处理资料时, 除了进行计算之外, 人们还采用各类图件来帮助分析, 传统的图件有其局限性。随着计算机应用的深入, 有理由希望使已有的宝贵资料得到更充分, 更灵活的展现和利用。

例如, 在地震预报和地震活动性研究中, 地震震中分布图是经常要使用的图件。通过它, 人们可以观察到在某段时间内、某个地区、某种强度的地震分布的情况。然而, 这种震中分布图只是静态的, 从中还难以直观地看到在这块区域内, 这些地震发生、发展的变化过程。为了解这些地震的迁移规律, 有人就在图上震中旁边注上发震时间, 或者再用箭头连接起来形成有向图。这样做, 图面就很难清晰, 地震较多时更令人眼花缭乱。本文要介绍的软件就是要使静态的图件动起来。当给定必要的参数后, 屏幕上首先显示地图经纬度线和地理要素, 接着便依时间顺序, 按着与实际发震时刻成比例的时间间隔逐一显示发生的地震, 並伴有闪亮和声响。这样, 我们就能更直观地观察到地震如何随时间而迁移。使用者可以边观察、边思索地震活动的发展规律。

地震信息时序图也是地震预报和研究工作常用的图件。通常是以时间为横坐标画轴, 以观测值或经过处理的计算值作为纵坐标在米格纸上描点连线。人们为了保存连续的图件, 就将一张张图件粘贴起来, 折叠成册, 需要时再展开来观察其资料随时间变化的情况, 进而分析和捕捉地震发生的前兆信息。然而这样的时序图不便于将一条曲线做分段比较, 也不便于对不同曲线进行对比。固然计算机的应用使人们方便地把图件搬上屏幕, 但是一帧一帧地显示仍然缺少灵活性。本软件提供动态显示的方法, 可以克服上述缺点。当给定必要的参数后, 首先根据用户选择的曲线种类(日值线或五日均值线, 或……)建立相应的无限时间横坐标轴; 一旦数据文件读毕, 便形成全部信息的时序曲线。使用者只要在键盘上通过适当的按键, 便能或快或慢地左右滚动其时序图, 从而在屏幕窗口内观察到任意时段曲线的变化过程。一屏设上下两个窗口, 它们分别独立地动作, 这样, 可十分灵活而简便地进行两条曲线间或者一条曲线的不同时段间的对比分析, 对于突出异常等感兴趣的部分也可放大显示。

地震信息图象的动态显示, 使静态的图件在一定程度上活动开来, 为分析预报和研究人員更有效、更形象地使用已有的资料提供了一个良好、方便的工具。

二、总体结构和基本功能

地震信息图象动态显示软件的总体结构如图1所示。

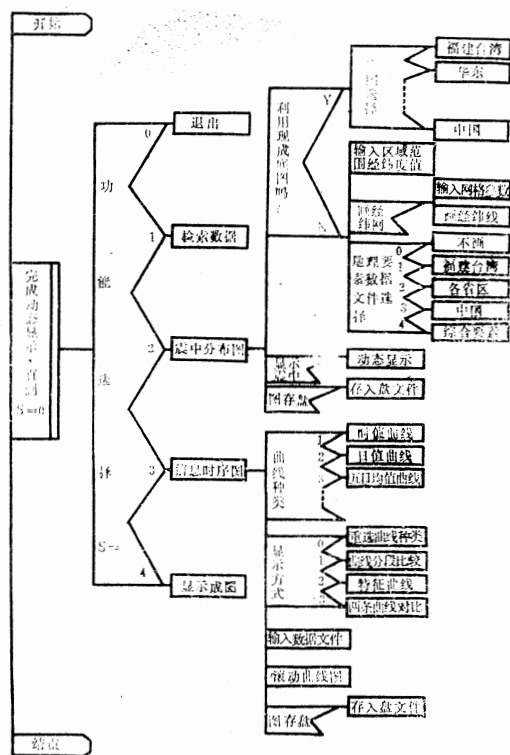


图1 DSSD软件PAD图

下面就震中分布图和信息时序图动态显示的基本功能分别加以介绍。

1. 震中分布图动态显示

为了显示地理要素和地震，我们使用了三类文件：

第一类是地理要素数据文件。在本软件中已经事先准备好中国各省和福建台湾地区地理要素的数据文件，其中包括有国界、省界、海岸线、主要城市和若干地震台站数据。对于它们，在程序执行绘图时可以选择使用。

第二类是工作底图成图文件，它是利用地理要素数据文件事先画好而存盘的底图，如中国地图、华北地图、福建台湾地图……。这些图件可以便直接调用。

第三类是地震目录数据文件。这要由使用者准备。文件中每个记录包括地震发生的

时间和震中经纬度值（地点）。该数据文件可以利用编辑程序编辑而成，也可以通过本软件的数据检索模块从地震目录数据库中检索得到。

当程序运行进入本模块时，将通过人机对话的方式输入有关参数。回答结束后，本软件就按用户指出的条件对该区域内、给定时间里发生的地震动态显示。人机对话需要回答的问题是：

(1) 利用成图吗？若回答Y，则显示菜单，有10个成图供选择；若回答N，即表示需要现画底图，则再请用户回答如下提问：

- 1) 选定地图范围，即输入该区域的最南、最北纬度和最西、最东经度值。
- 2) 是否要画经纬度线？若要画，则键入经、纬度线间隔各多少度？
- 3) 选用哪一个地理要素数据文件来画图？用户做菜单选择。

(2) 要动态显示震中吗？(Y/N)

本软件对地理要素和地震的显示，全采用等积圆锥投影的方法。为方便操作，这里并不要求用户回答投影比例尺是多少。这个值是不容易一下给好的。我们专门安排一小段程序，它根据键入的区域范围经纬度值，自动计算求得一个合适的比例尺值，此值将使给定的区域经过投影后能在屏幕正中位置形成一个大小适宜的窗口，这窗口充分大而又不超出屏幕范围。经纬度线、地理要素都落在此窗口内，在给定区域内的地震也都在此窗口内动态显示，而窗口之外的通通筛去。动态显示震中时，用1秒钟代替10天，这样10年内的地震发生过程可在几分钟内模拟完成。当然，这个时间比例因子可以另选。

假若觉得有必要保存这份震中分布图，可以将它存在盘文件中，以后需要时，这成图文件可随时调出，迅速重现。图2就是成图文件在打印机上拷屏的结果。

2. 信息时序图动态显示

为动态显示信息时序图，用户要准备用于显示的一个或两个数据文件，可以利用本

软件中数据检索模块从地震前兆数据库中检索而得到。

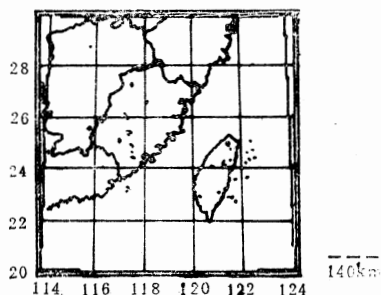


图2 震中分布图

当程序进入本模块时,人机对话主要是:

(1) 询问曲线的种类:是时值线还是日值线,或是五日均值线……。用户做菜单选择后,本软件便自动生成相应种类的无限时间坐标轴。对于日值曲线的时间坐标轴,已经考虑到了闰年平年、大月小月天数不同的情形。

(2) 提问信息的显示方式:是一条曲线做不同时段比较还是做特征显示?或是进行两条曲线对比?用户再做菜单选择。

(3) 输入要做动态显示的数据文件名。当从数据文件读数结束后,全部信息的时序曲线也就生成了。此时,屏幕上就从初始日期开始显示其曲线(对日值线来讲,显示40天),而时间轴上年月日刻度值与曲线上的点一一对应,它将跟着曲线同步左右移动。纵坐标则由程序根据数据的实际取值范围自动调整好并标上刻度值,在时序曲线移动过程中,它保持不变。

动态显示是这样进行的,使用者通过在键盘上按某些键使曲线滚动:

- 按空格键或<键,屏幕窗口内的曲线则左移1天,即所示的时间后移1天。
- 按>键,曲线则右移1天。
- 按数字键,曲线则快步左移。比如按数字1键,则后移1个月时间的曲线,按数字2键,则后移2个月,余类推。
- 按键盘上数字键下面一排字母键,曲线则相应地快步右移。比如按字母Q

键,则前移1个月时间的曲线,按字母W键,则前移2个月,等等。

- 按字符\键,则切换窗口。开始时,动态显示在下窗口内进行,利用上述按键来移动曲线,此时上窗口内图象不动。当按\键后,就切换到上窗口动作,而下窗口内图象保持当前状态不变,直到再按\键又切换回来。两个窗口的操作是分别独立进行的。
- 同时按控制键和字母S键,则将屏幕上当前窗口内图象存入盘文件。
- 按ESC键,则退出动态显示状态。

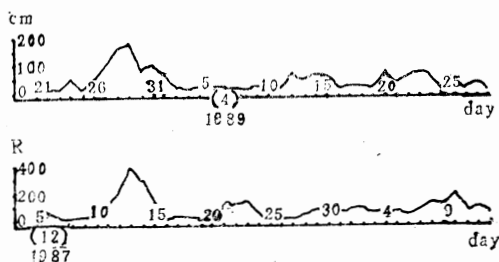


图3 曲线对比图

图3反映了两条曲线经动态显示进行比较的情况。

三、设计方法和主要特点

1. 交叉程序设计

为实现信息图象的动态显示,我们选用了功能很强的结构化程序设计语言, True BASIC来编制软件。由于数据主要来自数据库,我们就利用 dBASE III 库的数据检索模块来形成用于动态显示的数据文件。True BASIC模块和dBASE III模块是相互独立的。为了使它们能在统一管理下交互运行,我们通过操作系统DOS命令来统一调度。为此,采用批处理的技术[1],建立若干 DOS批命令模块。它一方面提供高层的菜单提示,另一方面用以控制不同语言程序的自动转换执行。下面举一个例子说明这三个系统转换运行的情况。

开始执行本软件,按DSSD后便显示主菜单。

这菜单显示,实际上是执行批命令文件

DSSD.BAT的结果。在本软件中,我们还建立了0.BAT、1.BAT、2.BAT、3.BAT和4.BAT这些批命令文件。不妨假定在主菜单下,我们选择2,也就是说要进行震中分布的动态显示,那么按下2之后就去执行2.BAT这个批命令,该批命令文件内容如下:

```
echo off
cls
cd\xx1
hello map.trc
pause
cls
cd\
dssd
```

它的作用是转到C盘子目录XX1,启动True BASIC系统执行map程序,并在True BASIC环境下进行动态显示。执行完成后回到DOS状态,重新在根目录下显示主菜单,供进一步选择操作,直到回答0时为止。

在整个运行过程中,交叉程序设计把几种语言融在一起,取长补短,是开发较强功能的软件的可行途径。

2. 模块化层次结构

该软件采用自顶向下为主、自底向上为补充的设计方法^[2]。总思路是从大到小,从粗到细,逐步求精,直到能实现具体的功能模块,从而形成如图1那样的模块化层次结构。这些功能模块大都是按逻辑功能划分而编写的独立子程序,基本上符合功能内聚。而各模块之间只是通过数据文件传送数据,其联系为数据耦合。为保证运行的安全可靠和结构清晰,我们把操作系统安排在根目录中,而True BASIC和dBASE III系统分别安排在各自己的子目录中。由于采用模块化层次结构,整个软件的可读性、可维护性都较好。

3. 高效的动画功能

这是本软件的一个重要特点,不仅可以在屏幕上显示美观的图象,而且能够产生形象悦目的动态画面。这里主要使用True BASIC语言中BOX语句独特的图形显示功能

[3]。其一般方法是,利用BOX KEEP语句把屏幕上某一方块内图象仍存于变量vax\$中,接着利用BOX CLEAR语句将该方块内的图象抹去(实即用屏幕背景颜色涂满),再用BOX SHOW语句将保存于vax\$中的图象在某指定位置上显示。在循环语句中恰当地配合使用这三个语句,并给好位置参数,就会产生理想的动画效果。

我们还利用PICTURE语句建立画图子程序,使得图象可以在不同窗口中和窗口转换中灵活地进行动态显示。

4. 通用的数据传送软接口

动态显示所需数据可能从dBASE III库中检索取得,也可能从别的语言程序中得到,还可能由自己编辑数据文件来提供。它们的格式各异,且与True BASIC语言的格式要求不一定相符,为了使无论从哪方面来的数据文件都能为程序方便地读入,我们利用错误处理程序的陷阱技术专门设计一个通用的输入子程序,它能在数据文件中筛选非信息的部分而正确读入所需要的数据。该数据输入子程序与INPUT语句是相容的,包含并扩充了INPUT语句的功能,还有一定的容错性。实践表明,这个数据传送软接口为交叉程序设计的实现提供了很大的方便。

5. 自动处理技巧

为使动态显示软件在运行中尽量少一些人工干预并提高效率,在程序设计中多采用自动处理技巧。比如在震中分布动态显示模块中,为了显示工作底图,本来应该给定比例尺才能做投影计算,从而产生一定大小的地图,然而在屏幕大小一定的情况下,这个比例尺很难一下给合适的:太大了屏幕显示不下,太小不能充分利用屏幕,显示图可能不清晰。鉴于此,专门编制一小段程序,综合考虑给定范围的经纬度差值与屏幕大小的关系,以求得比较合适的比例尺值。又比如在信息时序图中纵坐标轴上刻度值也是自动生成,若人工去做还要事先知道动态显示的曲线的最大、最小值。我们把这项繁琐的工

一种新颖的办公自动化工具

—电脑传真排版系统

天津市电子计算机研究所 张群

摘要 电脑传真排版系统是集中西文文体编辑、排版、传真格式转换、传真文稿校样、打印拷贝、传真发送等诸多功能于一身的,在IBM-PC XT/AT及其兼容机和CCDOS 4.0等操作系统下独立运行的微机汉字传真通讯系统。该系统将文艺理论排版技术与传真通讯技术相结合,为办公自动化领域增添了一个新的工具。本文着重介绍了该系统的软硬件结构、功能、软件设计和用于G■类传真的一维压缩编码与帧格式。

一、引言

电脑传真(简称PC-FAX)系统作为微机的一种新型通讯产品,问世已经许多年了。它的出现使传真通讯与计算机及网络技术相结合,实现了融文字、图像为一体,经由直拨电话线路,将图像文件在PC与PC或PC与传真机之间的远程通讯。微机兼备传真机的功能,不仅扩大了自身的应用领域,而且为微机增添了一种新型外设,为微机的发展注入了新的活力。越来越受到人们的重视。

以往引进的传真通讯软件,只能以图像方式处理经扫描器或传真机传送的汉字画面。无法将计算机中的汉字文件直接传真,更谈不上版面编排了。为了弥补这一领域的

缺陷,使电脑传真技术具有我们民族的风格与特色,我们将微机轻印刷技术与传真通讯技术结合起来,开发了一个我们称之为电脑传真排版的汉字传真通讯系统。

二、电脑传真排版的业务流程

电脑传真排版,是对通过各种手段输入到微机中的中西文文稿进行传真版面的编排设计,生成的图象编码文件可直接用于传真。图1基本反映了传真排版的整个过程。

首先,利用各种文本编辑软件对稿件进行输入或修改,将相应的排版命令嵌入其中,形成中西文字与排版命令混合的排版文本文件。接下来确定版面基本格式的版式参数。版式有自定义和系统标准版式两种。它好象

.....
作交给计算机去做,产生的结果更合理,还减少了人工干预,增进人机界面的友好性。

四、结语

本软件是作为地震监视和震情分析会商系统的一部分而设计的。它可以作为这个系统的一个子系统,但它又是相对独立的部分,可以作为单独的应用软件来使用。毫无疑问,该软件同样适用于非地震的信息图象

动态显示。从初步使用来看,本软件自动化程度高,人机界面友好,具有较强的实用性和可扩展性。

参考文献

- 1 张福炎等,微型计算机 IBM PC的原理与应用,南京大学出版社,1984。
- 2 仲萃豪等,程序设计方法学,北京科学技术出版社,1985。
- 3 朱禹等,汉字True BASIC 语言结构化程序设计简明教程,科学出版社,1988。

一本书, 根据版面要求来选择不同的参数。为了保证排版的效果和程序的正常运行, 在排版之前, 应对待排文本进行查错处理, 就好像计算机语言编译系统的编译过程一样。一旦查出错误, 会自动生成错误列表。供显示或打印。便于用户对照修改文本, 直至通过语法检查进入排版工序。排版工序根据版式参数和文稿中嵌入的命令进行版面设计, 按页生成点阵画面。最后经过传真格式转换, 得到数据压缩编码、符合国际规约的传真格式文件。此时, 用户既可直接在屏幕上校样或通过打印机硬拷贝校样, 若不满意, 可修改原文本, 改变文本内容或排版命令, 重复上述过程, 直至获得满意的版面。也可直接在校样后向目标机(传真机或PC机)发送。

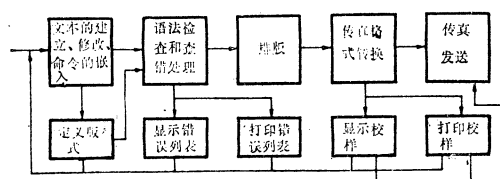


图1

三、系统构成

1. 硬件

- 主机为IBM-PC XT/AT、386、长城系列或兼容机。内存至少512KB, 10兆以上硬盘, CGA显示适配器。640×200BW图型方式。

- 9或24针点阵打印机。

- PC-FAX传真电路板。该板作为微机传真通讯的物理接口和电脑传真排版软件的重要设计依据, 它由一块传真通讯主电路板和一块Modem子板组成。子板为Rockwell公司生产的、由数块专用超大规模集成电路组成的R96F Modem板。调制方式采用QAM 16点正交调幅信号调制方式。传输速

率为9600、7200、4800、2400bps等多种, 与CCITTGⅢ、T4和T30传真规约相兼容; 主电路板是R96F与计算机和电话机及公共电话网络的网间接口。它与微机采用八位数字信息通道, 与电话机采用模拟信息通道。两种信息通道均经R96F进行数据交换, 该板构成了电脑传真所需的硬件基础。

- 公共电话网络和电话机。它是传真通讯的传输媒介。主机通过传真电路板上的公共电话网络接口、电话机接口与公共电话网络相连。象打电话一样, 只要拨通(当然也可由微机程控自动拨号)传真号码, 在网络程控交换机的控制引导下, 即可将传真文件向目标机发送, 或接收对方发送的文件。

2. 软件

- 操作系统为CCDOS 4.0或其它具有不驻留汉字库于内存功能的汉字操作系统。

- 与汉字操作系统相适应的打印驱动程序。

- WS字处理程序。

- 宋、仿宋、繁、黑、楷五种字体的24×24点阵汉字库。每种字库含7000多汉字及日、英、俄, 希腊文及汉语拼音字母和各种符号。

- 与传真接口板配合使用的PC-FAX。传真通讯软件。该软件具有成组、多方向, 定时、循环等多种形式的收发传真文件功能。可以用菜单或命令等多种方式运行。既可在PC与PC机、PC与传真机之间实现速率自适应的传真通讯, 也可将PC-FAX用作一个高速Modem, 人为选择传输速率, 在PC与PC机之间实现任何类型DOS文件(如可执行文件、数据库文件、程序文件、文本文件等)的传输。电脑传真排版软件的传真通讯部分就是以该软件为基础设计的。

- 电脑传真排版软件。集中西文文本编辑、普通轻印刷版面编排处理、传真转换、传真校样、打印、发送等诸多功能于一身的中西文排版传真通讯软件。使用该软件可在微机上对汉字文本进行字体、字号、行

长、页深、行距、表格、分栏、图空、花边、横竖排等多种样式的版面设计与编排。现场校样后，可实地传真发送。该软件也是本文介绍的中心。

四、电脑传真排版系统的功能与软件设计

根据排版行业的工艺流程，软件功能集文本编辑、版式定义、语法检查、轻印刷文艺理论排版、显示校样、打印校样、传真收发、DOS命令接口等各工序环节于一体，形成所谓集成工作环境。以使整体结构紧凑、功能多样，便于使用者掌握。

为便于开发，维护和扩充，程序用Turbo C语言编写，采用模块化结构，辅以公共函数库。一级模块有11个，二级模块有20个，通过总控模块分层调用。图2既示意了系统功能，又展示了程序模块的结构。

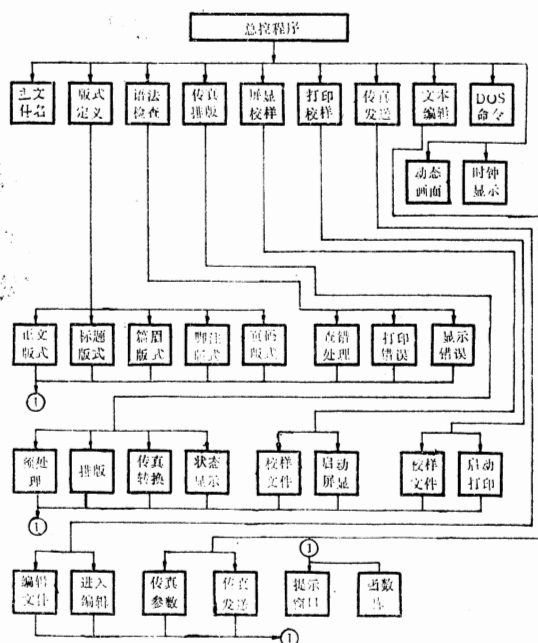


图2

1. 交互式工作环境与窗口技术的应用

随着软件技术的发展，软件设计不再单纯满足于实现既定的功能，在软件与使用者的交互性能上也提出了高要求，力求软件能有一个良好的用户界面，便于使用者清楚地

观察整个运行过程和各环节的状态，准确无误地进出系统的各个功能层次。

用人机对话完成各种操作，已成为目前计算机的一种主要方式。其实，这种人机交流本身就是一种语言。当今有多种技术支持这种人机交互语言的设计，但最为流行的是窗口技术。

窗口一般有可见的边界与屏幕其它部分分开。经过技术处理，窗口可来去自如，需要时弹出，不需要时消隐。其尺寸、格式可多样化。各窗口之间可相对独立，也可互有关联。由于一个窗口可在另一个窗口的某一位置上弹出，即使每个都占用屏幕的大部分，也可在一屏上看到几个窗口。利用窗口技术，可使屏幕的有限空间处理多种不同信息的显示，设计出复杂但却层次清楚的面，给用户一种了解整个工作过程的全景感。

窗口有分层式和堆栈式等多种，前者比后者有更多的特色，而后者由于弹出和消隐速度快，而比前者效率高。根据电脑传真排版软件的具体情况，我们选用了分层式窗口。图3是一个分层式窗口的例子。

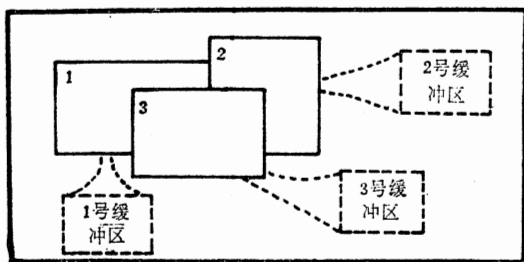


图3

图4是生成这种窗口的流程图。

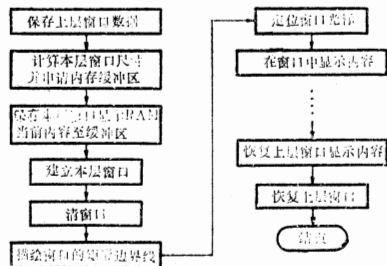


图4

为了在当前窗口使用完毕后，能消隐并返回到上一层窗口，每层窗口生成时，需保存

上一层窗口的数据(如窗口坐标、宽度、高度等),同时需开辟一块内存区域存放该窗口所占区域的原显示内容,以便将来通过逻辑运算消隐或移动窗口。待勾画出窗口边界,消除新建窗口中的旧内容并定位光标后,printf()函数的输出均在窗内进行。利用CCDOS的软中断INT 10H的6和7号子功能(AH=06和AH=07)可上下滚动窗口内容,进而实现上下翻页。这样,一个窗口就形成了。消隐或移动窗口内容时,Turbo C语言的putimage()函数提供了很方便的功能。该函数可将事先保存在缓冲区中的原窗口画面与显示RAM某指定区域进行与、或、非、异或、同或、拷贝等逻辑操作,其中拷贝操作既可消隐又可移动本层窗口画面。

应用分层式窗口技术,我们设计了电脑传真排版软件的交互环境,主要表现在以下几个方面。

(1) 窗口式中文控制菜单

总控程序负责协调软件的运行,控制程序的流向。软件启动后,整个屏幕分成三个窗口如图5所示。

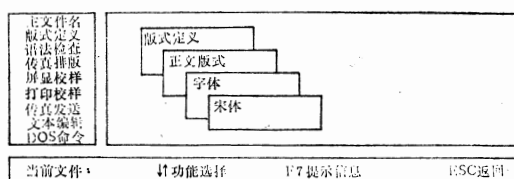


图5

左框为滑块式主菜单窗口。使用↑↓键移动滑块光标,可选择进入某个功能选项。右框为二级模块子菜单和工作窗口。以分层方式面向用户。图中以版式定义选项的工作过程为例。第一层窗口为版式定义选项的二级滑块式菜单;第二层窗口为子版式选择;第三层窗口为所选子版式的参数选择;第四层窗口为参数范围选择、输入窗口。定义文章版式的过程实际上是软件交互式处理用户键入的各种形式数据的过程。为了方便用户,减少输入错误,程序设计用交互方式在窗口中模拟表格,每项数据按栏填入,每栏均有汉

字说明。程序自动检验数据是否有效,自动处理数据输入域、光标属性和位置。下框为操作过程中的按键提示窗口,具有呈上启下的作用。内容随功能选择和运行状态而变。

(2) 上下文相关的帮助信息窗口

一个新的软件,其人机交互语言总是陌生的。新手往往不知道系统能够做什么?具体怎样操作。本软件利用窗口弹出技术为一至二级菜单的功能选项提供了对应的在线帮助信息。只要按动F7键,当前帮助窗口便弹出,利用↑↓、Pgup、Pgdn键可在窗口内浏览帮助信息的全部内容。用后,按Esc键,帮助窗口便会消隐,并不干扰程序对整个屏幕的使用。其实现方法为:

放入指针数组中的各个帮助信息与相应模块对应,由各信息文本的首指针、文本长度为参数,形成一个列表结构。当程序运行时,表的指针随之做相应移动。这样,当键扫描程序捕捉住F7键时,程序根据列表的当前指针,确定当前帮助窗口的信息长度和位置,由具有上下滚动和翻页功能的显示程序在弹出的窗口中显示行文。最后消隐窗口,恢复屏幕原貌。

2. 文本编辑

此功能项打开DOS二次命令处理器之前,保存当前屏幕画面,然后运行WS字处理程序,为用户提供一个编辑、修改文稿的环境和手段,退出后恢复原屏幕。

3. 版式定义

版式是排版的一个术语。面向一本书,它确定了版面的基本格式,每一个源文本都对应了一个版式。排版前省略版式定义或进入版式定义后不改变各参数的缺省值,则系统默认采用标准版式,否则为自定义版式。

版式由五个子版式组成。依次为正文、标题、脚注、篇眉、页码子版式。正文子版式决定了每页版面的行长、页深以及缺省字体、字号、行距、字距等参数的选择;标题子版式定义了多达四级的标题格式(包括标题的定值);脚注子版式定义了注码的类型

以及注文的格式；篇眉子版式定义了正文行之上出现的、常用来说明每页内容纲要的文字行的版面格式和篇线形式。

版式一经确定，排版程序将以此作为版面编排的依据之一，处理文章的版面编排。除非在文章中出现专门的命令来设置文章格式和字属性。此时，以文中命令为准。

版式定义模块主要是通过建立分层式输入窗口，辅以查错和光标滑块技术，实现版式参数的正确输入，将超限和非法参数拒之门外。最终形成格式如表1所示的，后缀为.ver的版式文件。

表1 版式文件

序号	名 称	序号	名 称
	正文：	19	定位
1	行距	20	等级
2	字体		四级标题：
3	字号	21	行距
4	行长	22	字体
5	页深	23	字号
	一级标题：	24	定位
6	行距	25	等级
7	字体		脚注：
8	字号	26	行距
9	定位	27	字体
10	等级	28	字号
	二级标题：	29	注码类型，
11	行距		篇眉：
12	字体	30	行距
13	字号	31	字体
14	定位	32	字号
15	等级	33	定位
	三级标题：	34	眉线有无
16	行距	35	眉线形式
17	字体		页码：
18	字号	36	页码有无

4. 语法检查

传真排版系统提供四大类，共计30余条排版命令供用户使用。这些命令嵌入文本中正确与否，决定了版面的预定效果能否实

现，排版工序是否正常进行。语法检查模块就是为了把住这道关卡而设立的。

语法检查模块不仅在文字与命令混合的文稿中识别命令，对其进行类似于计算机语言编译系统那样的词法检查，而且还能分析命令间的嵌套关系正确与否和一些语法逻辑问题。只有通过语法检查的文本，才能保证排版程序的正常运行。其处理流程如图6所示。

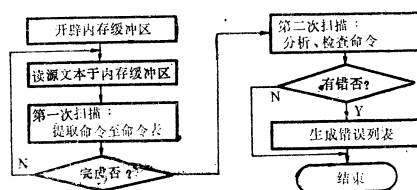


图6

首先将文稿读入内存一号缓冲区。通过两次扫描完成整个语法检查过程。第一次扫描识别一号缓冲区中的命令，将其提取到二号缓冲区，并顺序排放形成一个命令表。若文稿没有全部处理完，就利用覆盖技术，继续读文稿于一号缓冲区，直至扫描完整个文稿。第二次扫描对二号缓冲区中的命令进行分析、查错。每一条命令都有相应的分析处理程序，由该模块主程序控制这些子程序的调度。一旦发现错误，则按表2格式生成后缀为.err的错误列表文件供在相应窗口中显示或打印使用。

表2 错误列表

错误序号	位置	类型	提示信息
1	10	1	<input type="checkbox"/> 错误
2	21	3	字号越界
...

对命令的分析、检查主要包括以下几个方面：

- 命令书写是否正确。
- 命令参数值是否超限。
- 命令参数个数、类型及分隔符是否正

确。

- 开关命令的嵌套关系是否正确。
- 语法逻辑有无问题。

5. 排版

排版模块是本软件的核心模块。它以文稿、版式文件和文稿中嵌入的命令为依据，进行既定的版面设计，按页生成传真格式文件。文件名依次为“主文件名.NN”(NN为00~99)。排版模块能实现的版面效果由表3的排版命令集合表给出。

表3 电脑传真排版命令集

分 类	功 能 名 称
行排类	自然段结束命令 取消段头空命令 字空命令 字体命令 字号命令 字体反显命令 行调整命令 文章结束命令
版面处理类	篇眉定义命令 随文脚注命令 图空命令 分栏命令 分栏命令 标题命令 换页命令
特殊排版类	定义横向往点命令 采用横向往点命令 取消横向往点命令 着重线命令 加减字距命令 加减行距命令 文字填充命令 花边命令
表格排版类	表格状态命令 基本制表命令 表格抽线命令 表文填充命令 表文竖排命令 表文横排命令

整个排版分为预处理、排版处理、传真格式转换三部分。其流程如图7所示。

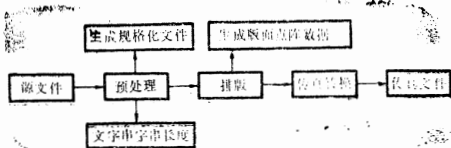


图7

(1) 预处理

预处理主要是清除源文稿中本软件认为非法的字符，按每条命令的作用域划分字符串

串的段落，生成规格化的中间文件和各字符串长度表4，为正式排版作准备。

表4

规格化文件	字符串长度数据表
命令 1	第1字符串长度
字符串 1	第2字符串长度
命令 2	第3字符串长度
字符串 2	...
命令 3	...
字符串 3	...
...	...
...	...
...	...
命令 n	第n字符串长度
字符串 n	

(2) 排版处理

版面按页划分，规格为A4幅面。分辨率为每毫米3.85个扫描行，页宽 1728 个扫描点，长度为215毫米。页深1075个扫描行，长度为277毫米。每个扫描点对应一比特，故每行216个字节。一个标准A4幅面的点阵文件，其长度为 $216 \times 1075 = 232200$ 字节。整个版面的使用划分如图8所示。

传真文件的幅面与格式

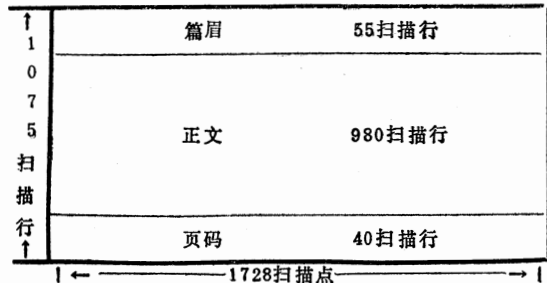


图8

内存使用情况如图9所示。

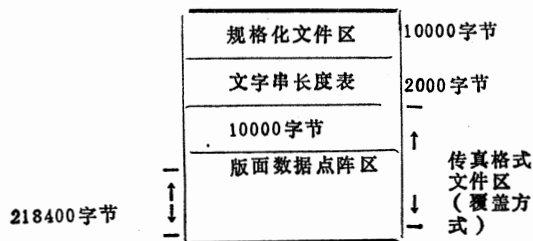


图9

首先在字符串长度表缓冲区的下面开辟一个228400字节的数据点阵区，其中218400字节作为版面数据点阵区（考虑到在页面的

两侧留出足够的空白边缘,故实际页面的有效宽度为1680个扫描点,这样218400字节的缓冲区就足以容纳一页点阵数据),程序在识别规格化文件中逐条命令的基础上,调用相应的命令处理程序,对紧接每条命令后面的文字串(长度可从字符串长度表中查阅)进行排版。若当前处理的是字体、字号等版式文件中也涉及的参数,则以命令中出现的参数值为准。为了保证版面效果,按排版行业的要求,还要进行行末、页末禁则处理。最终形成一页完整的版面点阵于内存缓冲区中。

纵观排版命令集,利用不同命令的组合可以排出许多种形式的版面。而实现这30余条命令的核心是公共函数库中的getpixel、make-form、removeline等几个函数。它们实现了文艺理论排版最基本的功能,绝大多数命令都是在它们的基础上实现的。现简要给予介绍。

1) getpixel函数

该函数是一串字符的字模提取、字体、字号、字距选择、正反显示、按版面格式排放于数据点阵区指定区域的多功能函数。只要给出字库指针、间距大小、机内码首址、机内码个数、版面排放指针及字号、反显的标志参数,就可得到一串文字的局部版面。众所周知,一页版面由若干行组成,而该函数恰恰实现了行排功能,所以它是排版的基干

函数。其处理流程如图10所示。

初始化主要是进行一些指针的预置和缓冲区的申请;各种字体的汉字库(24×24点阵)均贮存于硬盘中,每个汉字占72个字节,读取一个汉字的字模,首先要计算该字模在库中的起始位置,汉字机内码与字库的区位排列有一定的换算关系,按此关系式得到的数值定位库指针,连续读出72个字节即可;24针的汉字库每个字模按纵向排列,需转换成横排,才能用于排版。见图11。

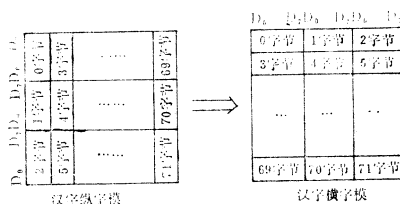


图11

下面是实现这两个步骤的源程序。

```
int i, k, m, n, p, q;
unsigned long int pp;
unsigned char vert_buf[72], hori_buf[12];
FILE *fbase;
fbase=fopen("某种字体汉字库", rb);
pp=((((区号&0×7f)-0×21)*94+(位号&0×7f-0×21))*72;
fseek(fbase, pp, SEEK_SET);
for(i=0; i<72; i++){
    hori_buf[72+i]=0×00;
fread(vert_buf, num, 1, fbase);
for (n=0, q=0; q<3; q++){
    for (p=0; p<8; p++){
        for (m=0, K=0; k<3; n++, k++){
            for (i=0; i<8; m+=3, i++){
                if ((vert_buf+n)&(0×80u)
                    i))
                    (hori_buf+72+m+q) |= (0×80u>>p);
            }
        }
    }
}
```

字号的选择是通过对读取、转换后的字模进行放缩实现的;反显是通过对字模取反得到的;字模传送到数据点阵区要解决的主要是像素点的定位问题。由于每个像素对应1 bit,这就要求程序有严谨的位操作功能。

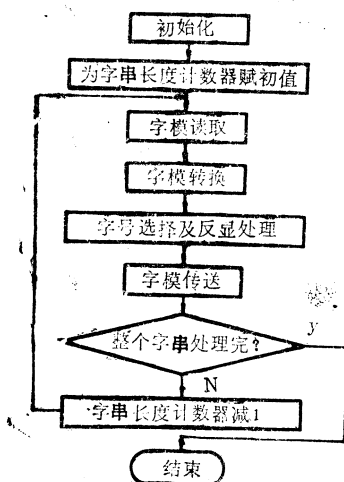


图10

定位一个点可以说需要三维坐标,即行、列及所在字节的第几位来决定。字模数据是连续的,由于放置的起始位置不一定是字节的整数倍,再加上字距的影响,故点的计数和定位就较为繁琐。程序中除了设置数据点阵区当前字节指针外,还设置了当前字节右侧空 bit 数计数器及行字节计数器来解决象素的定位问题。

2) 表格函数—make-form () 和 remove-line ()

表格分成基本表格和复杂表格两种。如下图所示12所示。

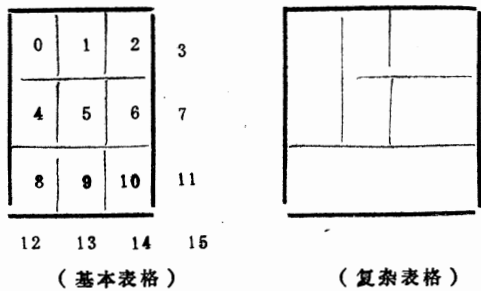


图12

基本表格是由N行M列个方盒子组成的方型表格。make-form () 函数就是为基本制表而设计的。它是根据调用者提供的栏宽、栏数、行长、行数、位置等制表参数,经计算得出表格横纵线个数、每条横线和纵线的长度(每条线均占两个扫描点的宽度)、每条线交叉点的位置指针及所在字节的Bit位置等一系列数据,按照如图所示的方盒子顺序填写如表5所示的数据结构,以便为表格抽线和表文填充时使用,然后依次画线形成基本表格。

表5 表格数据结构

0号盒子表文填充指针	n号盒子表文指针右侧空位
1号盒子表文填充指针	0号盒子抽线指针右侧空位
...	1号盒子抽线指针右侧空位
...	...
n号盒子表文填充指针	...
0号盒子表文抽线指针	n号盒子抽线指针右侧空位
1号盒子表文抽线指针	0号盒子长度、宽度值
...	1号盒子长度、宽度值
...	...
n号盒子表文抽线指针	...
0号盒子表文指针右侧空位	n号盒子长度、宽度值
1号盒子表文指针右侧空位	表格竖线条数
...	
...	

remove-line 函数是表格抽线函数。复杂表格是由基本表格抽线得到的。该函数以抽线参数和基本制表所形成的表格数据结构为依据。抽任何一条线段都要考虑周围线段的状况来决定抽线的长度和形式。具体地说,相邻盒子的共用线段在抽线时有一个算法问题。如图13,抽横线要以左侧为基准,判断三条线是否已全被抽掉。若全被抽掉,则C点抽线不仅要抽掉cd长度,而且还要抽掉ac长度。否则,应只抽cd长度。当然判断顺序也有规定。须先左上线,后左下线,再中间横线。

抽纵线如图14所示。以上面为基准。若三条线全被抽掉,则抽掉ad长度,否则只抽cd长度。判断顺序也应左横线,后右横线,再竖线。

抽线后,原表格数据结构也需相应改变,以标志出哪些线段被抽,保留了哪些线段,以便再次抽线和表文填充时使用。表文填充就是根据数据结构中给出的各盒子左上角的指针及每个盒子的长度和宽度填写各盒子表文内容的。

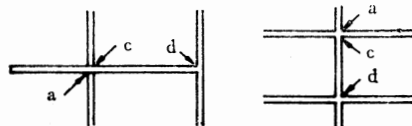


图13

图14

3) 传真格式转换

排版后生成的点阵数据(每页约218400字节)须转换成传真压缩格式的编码数据才能用于传真。这种转换的原理为:

对点阵数据扫描计数,统计每扫描行上连续白点和黑点的个数(即游程长度),查传真编码表并按规则组合成一维传真编码。具体地说:一维传真编码分黑、白游程两种。如表6表7所示:

每种又分为终止码(Terminating code)和组合基干码(Make-up Code)两种,每个游程长度(黑或白)均由一个终止码或一个组合基干码加一个终止码组成。

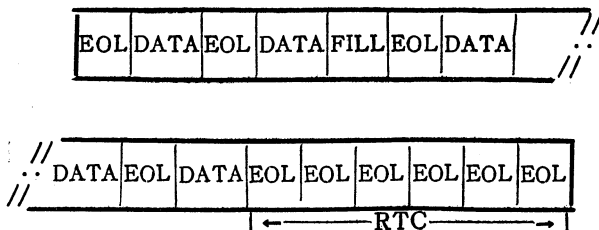
表6 终止码表

白点游程长度	编码值	黑点游程长度	编码值
0	00110101	0	0000110111
1	000111	1	010
2	0111	2	11
3	1000	3	10
...
61	00110010	61	000001011010
62	00110011	62	000001100110
63	00110100	63	000001100111

表7 组合基干码表

白点游程长度	编码值	黑点游程长度	编码值
64	11011	64	0000001111
128	10010	128	000011001000
192	010111	192	000011001001
...
1664	011000	1664	0000001100100
1728	010011011	1728	0000001100101
EOL	000000000001	EOL	000000000001

游程长度为0~63个点的,用对应的终止码进行转换;64~1728个点的,用对应的组合基干码加剩余长度对应的终止码进行转换。组合基干码的游程长度是64的整数倍,而终止码的游程长度小于64。对版面点阵压缩编码后,还需按传真通讯帧格式将整幅画面的传真编码打包处理。格式如下:



其中: EOL 行结束标记和帧起始标记
 DATA 每个扫描行一维压缩编码数据
 FILL 为了保证每行数据最小传输时间添加的由一串0组成的填充位
 RTC 6个EOL组成的文件结束标记

一般200余K字节的版面点阵经压缩编码后可降至30~90K字节。

6. 传真发送

排版后生成的传真格式文件通过该模块实地发送。该模块调用公共函数库中的传真发送函数,以窗口方式让用户选择要发送的文件和传真号码,缺省文件为“主文件名.01”。然后打开DOS二次命令处理器,执行一个由该模块生成的、由一系列传真通讯命令组成的DOS批命令文件。这组命令由传真通讯软件识别,该软件通过屏幕给出传输速率和通讯过程的状态信息,自动拨号,自动检测是否为多页文件。若是,则按页序传送整个文件。传送后删除中间文件并恢复屏幕原画面。

7. 屏显校样

排版生成的传真文件可以在该模块下进行屏幕显示校样。操作者可通过窗口选择要校样的文件,缺省为“主文件名.01”。通过按键操作可对版面进行放缩[+、-键]、旋转180°[F3键]、上下左右移动[↑↓←→键]、正反显[I、i键]、横向放缩[X、x键]、纵向放缩[Y、y键]、观察下一页[F7键]等许多种手段的校样。

8. 打印校样

该程序是排版结果进行校样的另一种工具。它对传真格式文稿打印硬拷贝。处理的缺省文件为“主文件名.01”,通过在窗口的参数域中输入不同的参数值,可对文稿进行横向放大,纵向放大及横纵向同时放大等多种形式的打印。

9. 其它

动态画面程序由总控程序在软件运行之初调度运行。它在屏幕上展示了PC机、传真机、公共电话网络及计算机外部设备进行传真通讯的动态流程,为用户提供了一个了解传真通讯概貌的直观画面。

时钟程序是在屏幕右下角生成一个带有时、分、秒的数字时钟。该程序实际是截获了操作系统的1C号中断。将该中断向量指向自行编制的时钟处理程序,形成新的中断服务程序。1C号中断是硬件激发的中断。每秒

该流程如图15反映了传真排版的整个工作过程。

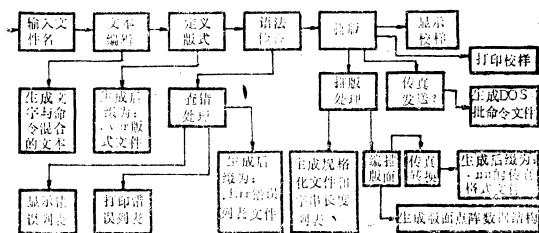


图15

电脑传真排版系统的实用价值在于国内大量拥有PC及其兼容机的用户,只要配备了传真电路板和电脑传真排版系统软件,就可以使微机兼备传真机的功能,成本却大大低于购买传真机的费用;经过排版生成的传真格式文件不仅可原样复制到传真机,也可放缩输出到打印机。这就使用户兼得轻印刷排版与传真排版两大功能。

鉴于目前的技术条件、时间限制和我们的水平，该系统主要是从实用角度出发，功能设计上力求实现文艺理论排版的最基本的功能，随着时间的推断，我们将改进工作，提高系统的功能，向更深的层次拓展。参加此项工作的还有赵民益和龚鸣鸣二位同志。

日立公司近日宣布，他们已经研制成功“神经计算机”。

新型计算机的“大脑”由八条超大规模神经集成电路组成，每条电路包括1152个彼此相连的电子“神经细胞”。这样，整个计算机结构形成八个稠密的人造神经元网，人造神经元类似精巧的人脑神经组织内闪电式思维跳跃那样控制着强大信息流的通过。这种系统能以每秒钟运算23亿次的前所未有的速度处理信息，就是说，它比现有超级电子计算机快9倍。

“神经计算机”能够解决一系列只有人的直觉思维可以解决的“创造性”算题，即确定最佳解题方案，排列“推理链”，确立目标明确的算法，分辨形象，“按惯例”运算。而且新型“神经电子计算机”能够自学，在积累起来的信息基础上产生新知识和获取经验。现在该公司已生产这种新型号计算机使用的软件。

“大麻”病毒综析

中国矿业大学

孙德坤 蔡思伟

摘要 本文对大麻病毒的表现形式,常用的诊断方法做了全面的论述,尤其对病毒程序进行了详细的分析。在此基础上,除提供了一般的消除大麻病毒的方法外,还提供了“反大麻病毒”实用程序清单。最后对修复由于病毒所破坏的文件提出了具体方法。

一、大麻病毒的症状

与小球病毒一样,大麻病毒也属于PC系列机上的操作系统病毒。该病毒程序在启动机器时被装入内存0000:7c00处执行。

病毒程序执行时,屏幕上出现“Your PC is now stoned! ...”。如果你的机器启动后,出现上述显示,毫无疑问,用来启动的磁盘已染上了大麻病毒。由于大麻病毒的潜伏期长,即使启动机器时不显示大麻标志,只要留心观察,用硬盘启动机器比原来用的时间稍长或造成死机,也可以断定,硬盘上可能已染上了大麻病毒。在染有大麻病毒的环境下,用户按常规调试程序,进行磁盘读写,有时会出现两个毫不相干的文件残缺的连在一起。在DOS环境下用DIR命令检查目录,会发现后半部分的文件名已面目皆非,并且出现一片杂乱无章的字符。(从理论上讲,大麻病毒允许显示96个文件)。此现象表明,盘上已染上了大麻病毒。

由于大麻病毒在机器启动后常驻内存,并伺机发作传播,此时,只要对A驱动器中的未加写保护的盘进行读写操作,就会染上病毒。如果硬盘原来并无病毒,用带病毒的软盘启动计算机后,对硬盘进行读写操作,也会把大麻病毒传染到硬盘上去。

二、大麻病毒的诊断

对于被染有病毒的360k,双面软盘,病

毒驻留在逻辑0扇区。它把原驻留此扇区的正确引导程序搬到了逻辑0B扇区(。此扇区是FDT所占的最后一个扇区,可以存放第97至112这些文件的目录项。由于病毒程序的这种操作,便导致0BH区中的目录项被覆盖,从而造成文件的“丢失”。

当硬盘染上大麻病毒后,病毒程序侵占了硬盘的0磁头,0道1扇区。该扇区是一个隐含扇区,即主引导扇区。把主引导程序搬到了0磁头,0道7扇区。这样,对于逻辑0扇区定在1磁头,0道1扇区的硬盘,正常引导程序仍在隐含扇区中,故仍可以启动机器,只是具有传染性;而对于逻辑0扇区设在0磁头、0道2扇区的硬盘,0磁头、0道7扇区是硬盘的FAT表区,所以病毒程序即破坏了FAT表,从而造成文件“丢失”。本文针对后一种情况讨论。

1. 软盘上大麻病毒的诊断

在无病毒环境下按如下步骤操作。

A>DEBUG

-L0001; 读A盘逻辑0扇区

-U0007; 显示两条指令内容

××××:0000 JMP 07c0:0005; 对应四个扇区。

××××:0005 JMP 00A1;病毒标志字节:
EAH, 05H, 00H, CoH

-D 18A 1B7; 显示“Your PC is now
stoned...Legalise marijuana”

若有上述情况发生,可确认A驱动器中的盘已染上了大麻病毒。

2. 硬盘上大麻病毒的诊断

如上所述, 大麻病毒侵入硬盘而驻留在0磁头, 0道1扇区, 该区是隐含扇区, 因此, 用DEBUG或PCTOOLS的一般命令是无法读出的。具体方法加下:

```
A>DEBUG
-R IP
:0000
-A 0000
××××:0000 MOV AX, 0201
      MOV CX, 0001
      MOV DX, 0080
      MOV BX, 0100
××××:000c INT 13; 读0柱、0头1扇区
-G=00 0E          ; 执行程序
-U 100 107         ; 显示 "JMP
                   07c0:005"
-D 28A 2B7; 显示 "Your PC is now
                   stoned!"
```

通过上述步骤可以确诊硬盘是否染有大麻病毒。

三、大麻病毒的工作机理

大麻病毒在磁盘上占1B8H 个字节的存储空间。它是通过机器运行时, 往往总是“机械”地读入磁盘第一扇区的内容执行这一“漏洞”而设置物理位置的, 以大麻病毒程序代替原来的引导程序。并修改INT13指针, 使得INT13截流到病毒程序之中。

整个病毒程序分为两部分。第一部分用于大麻病毒的自引导, 并实现对硬盘主引导区的感染。第二部分是病毒的“INT13”部分, 主要实现对A驱动器执行读写时实施传染。

第一部分其入口在大麻病毒的四个标志字节(EAH, 95H, 00H, C0H) 的后面紧跟的跳转指令JMP 000A1中体现出来。下面列出头尾几条指令, 供读者查阅时参考。

```
-L 0 0 0 1
```

```
-U 0 0 07
```

JMP 07c0:0005; 对应于标志字节

JMP 00A1 ; 第一部分入口

```
-U 00A1,0187
```

××××:00A1 XOR AX, AX

××××:00A3 MOV DS, AX

××××:00A5 CL

:

××××:0183 INC CL

××××:0185 INT 13

××××:0187 JMP 014E

在分析程序前, 将程序中用到的几个BIOS工作单元做一介绍。

0:[004E]、0:[004C]这两个单元用于存放INT 13的段地址和偏移量。

0:0413 单元为系统保存的以K字节为单位的内存容量存放单元。

0:043F 单元是磁盘驱动器马达开关状态单元。

0:46C 为机器内部时钟计数器。

程序分析如下:

××××:0005 JMP 00A1; 跳转到病毒第一部分。

××××:00A1~00AB ; 设置堆栈。
00AC~00B5 ; 保存原INT13入口。

00B8~00C4 ; 可用内存减2K用来存放病毒并根据当前内存的大小计算最高段地址。

00C6~00CF ; 修改INT13入口为病毒第二部分入口。

00D3~00DE ; 把病毒移到内存高端。

00F4 ; 判断带毒盘符是否为0, 即判断当前为何盘启动。

00F8~0103 ; 把0道0磁头7区的硬盘主引导读入内存。

0106~010E ; 把A盘逻辑B扇区上的引导程序读入内存。

0111 ; 测试和内定时计算器的
低位字节中的低3位是否
全为0。若是, 则显示提
示符。

011B~0128 ; 显示“病毒”提示。

012A~0139 ; 读硬盘第一个扇区到内
存。

013B~014C ; 检查硬盘第一扇区, 如
果最前的4个字节与病毒
标志字节相同, 则不再传
染, 否则“传染”。

014F ; 置带毒盘符为0

0154~0155 ; 执行正常引导程序。

0159~0168 ; 把带毒盘符记为2, 并把
硬盘主引导写入 0道0磁
头7扇区。

016F~017D ; 把硬盘的4个分区信息表
移到病毒后面。

017E~0185 ; 把“大麻”病毒和分区
信息表一同写到硬盘第
一扇区, 完成传染硬盘。

018A~01B7 ; 病毒提示数据区。

第二部分从××××:0015~××××:
00A0。

-U 0015 00A0

0015~0023 ; 判当前盘操作是否在A 驱动
器上进行的。

0029~0036 ; 检测A 驱动器中马达是否处
于开状态。

003A~0052 ; 读A盘0面0道1区到内存。

006F~0074 ; 判当前盘是否已被传染。

0079~0085 ; 把正常引导程序写到A盘的
1面0道3扇区。

008B~0096 ; 把病毒程序写到A盘的0面0
道1扇区, 完成传染。

四、大麻病毒的消除

消除盘上的大麻病毒基本上有两种途
径。

1. 用DEBUG工具软件进行, 其步骤如
下:

对于软盘:

A>DEBUG

-L 100 0 0b 1; 读入A中病毒盘的0b扇区。

-W 100 0 0 1; 正常引导写回0与 0道1 扇
区。

对于硬盘:

A>DEBUG

-R IP

:0000

-A 0000 ; 从偏移量0开始汇编

××××:0000 MOV AX, 0201

MOV DX, 0080

MOV CX, 0007

MOV BX, 0100

INT 13

MOV AX, 0301

MOV DX, 0080

MOV CX, 0001

MOV BX, 0100

××××:001A INT 13; 写回正常引导。

-G=00 1C

-Q

2. 利用“反大麻病毒”程序进行消毒

明白了大麻病毒的机理之后, 便可以用
不同种类的语言编制“反大麻病毒”的程
序。笔者曾写出了“反小球病毒”和“反大
麻病毒”两个程序。都是用汇编语言编写的。
因本文讨论大麻病毒, 故将“反大麻病毒”
程序清单附后。有兴趣的同行不妨一试。

五、被病毒破坏的文件修复

大麻病毒被清除之后, 接下来的工作是
尽量恢复原来磁盘上的文件。对于软盘, 病
毒破坏了文件目录表(FDT)的最后一个扇
区, 运行“反大麻病毒程序 STONE.EXE
”, 使得0BH扇区恢复成一个可再填写目录
项的FDT表区, 接着可以恢复软盘上所丢失
的文件的目录项。对于硬盘, 病毒破坏了文
件分配表(FAT)的逻辑06扇区, 恢复也
较为困难。在做文件恢复之前, 必须对FDT
表和FAT表有一个清楚的了解。

为管理磁盘上已建立或将建立的文件，DOS在盘上的固定扇区建立了一张文件目录表。该表描述文件名、子目录名，卷标及其相应信息，每个文件的“簇链”的簇头。簇头之后的簇链则存放于文件分配表中。对于

文件名	扩展名	属性	保留区	时间	日期	起始簇	长度
位移00	08	0B	0C	16	18	1A	1C 1F

为了统一管理和分配磁盘空间，DOS建立了一张文件分配表FAT，FAT是簇号的集合。为了确保以后存取文件时能搜索到每一个不连续的扇区位置，DOS将一个文件分配到的簇号用一条“簇号链”连起来。文件的第一个簇号在FDT中。在簇号链中，除最后一个簇号有特殊标记外，其余的每一个簇号位置上，都记载着文件下一个后续的簇号。对于DOS 2·X及以下版本，在FAT中，对每一簇号有一个12位的登记项，即占1.5个字节。下面给出恢复文件的方法。

1. 软盘上文件的恢复

(1)用无毒盘启动系统，执行DEBUG。

(2)将有病毒的盘的逻辑AH区调入内存（因BH区已遭破坏）。查寻最后一个目录项的位移1A至1C的内容（起始簇）。计算该簇号在FAT中的位移，然后依次在FAT中把该文件所占簇号链找出。具体步骤如下：

a) 当前簇号乘以1.5
b) 上述乘积的整数作为FAT的位置。
c) 如果当前簇号为偶数，则该位移处的1个16位bit中低12位指示下一个簇号，否则高12位为下一个簇号。

d) 若该位移处为FFH，则是文件的簇尾。

(3)在FAT 中找到这个簇尾，则紧跟其后的位移处就是丢失文件的开始，记下这个位移值。

(4)将这一位移值除以1.5并取整，所得值即为丢失文件的簇头号。

(5)在逻辑BH区用DEBUG 的E命令重新填入丢失文件的目录项内容，并将簇头号

360K软盘，FAT表在逻辑1至4扇区，FDT表在逻辑5至0B扇区；对于10MB 硬盘，FAT表在逻辑2至11H扇区，主目录FDT表在FAT表之后占20H个扇区。FDT表的每一登记项占32个字节，结构如下所示。

填入相应位置。至此便恢复了一个文件。

(6)重复上述步骤，再恢复其他丢失文件。

2. 硬盘上文件的恢复

对于硬盘，大麻破坏的是FAT区中的一部分。恢复文件的步骤如下：

(1)先找FDT表，查出相应目录项中的“簇头”号和“字节长度”值。计算该文件应占的簇数及第一簇在FAT中的位移。

(2)用DEBUG中的S命令搜索硬盘数据区中的内容。多找几次直至找到丢失的文件块。（一次可调入80个扇区到内存，用S命令时最好能附上一个丢失文件块具有的“标志字节”）

(3)找到丢失的文件块后，则根据该块所在扇区号计算出对应的簇号。

(4)根据第(1)步得到的值和丢失块对应的簇号，结合前述查找簇链的方法（见软盘丢失文件的恢复）在FAT 中找到对应文件的簇链，并将当前查到的丢失的文件块所在的簇号填入正确位置。

(5)重复上述步骤，可找到丢失的各文件块（即一个文件的各个部分），恢复相应的FAT中的内容，并在最后一个簇号内填入FFH结束标志。从而完成了一个文件的恢复工作。

按上述方法，继续恢复其他丢失的文件。

以上介绍的丢失文件恢复的方法是可行的。笔者曾对由于大麻病毒的侵入而遭破坏的一些重要文件进行过恢复，结果是令人满意的。

附：反大麻程序清单

```

A>type dd, asm
data segment para' data'
    d_flag      db 80h
    stone       db 512 dup(0)
    rootd       db 16 dup(0, 31 dup(0f6h))
    help1       db 0dh, 0ah, '软盘(1)硬盘(2)
                  请输入1或2$'
    help3       db 0dh, 0ah, '在B驱动器中插
                  入软盘, 按下任一键$'
    help4       db 0dh, 0ah, '输入Q键退出,
                  其它键继续$'

data ends
code segment para' code'
    assume cs:code, ds:data, es:data
sd proc near
    mov ah, 0
    mov dl, 1
    int 13h
    mov ax, 0201h
    mov dx, 0101h
    mov cx, 0003h
    mov bx, seg stone
    mov es, bx
    mov bx, offset stone
    int 13h
    mov ax, 0301h
    mov dx, 0001h
    mov cx, 0001h
    mov bx, seg stone
    mov es, bx
    mov bx, offset stone
    int 13h
    mov bx, seg rootd
    mov es, bx
    mov bx, offset rootd
    mov ax, 0301h
    mov dx, 0101h
    mov cx, 0003h
    int 13h
    ret
sd endp
hd proc near
    mov ah, 0
    mov dl, 80h

```

```

    int 13h
    mov ax, 0201h
    mov dx, 0080h
    mov cx, 0007h
    mov bx, seg stone
    mov es, bx
    mov bx, offset stone
    int 13h
    mov ax, 0301h
    mov dx, 0080h
    mov cx, 0001h
    mov bx, seg stone
    mov es, bx
    mov bx, offset stone
    int 13h
    ret
hd endp
start proc far
    assume ds:data
    push ds
    xor ax, ax
    push ax
    mov ax, data
    mov ds, ax
    lea dx, help1
    mov ah, 9
    int 21h
    mov ah, 1
    int 21h
    cmp al, '1'
    jnz start3
    mov d_flag, 0
    lea dx, help3
    mov ah, 9
    int 21h
    mov ah, 1
    int 21h
    call sd
start2: cmp d_flag, 80h
    jz exlt
    lea dx, help4
    mov ah, 9
    int 21h
    mov ah, 1
    int 21h

```

UNIX/XENIX文件系统结构与误删除文件的恢复

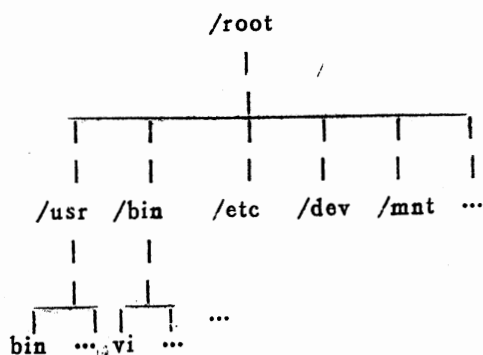
新疆独山子乙烯工程指挥部 刘长江

摘要 本文将介绍UNIX文件系统结构和如何利用UNIX文件系统的结构来恢复被误删除的文件。

随着UNIX/XENIX多用户分时操作系统的发展和国际标准化组织制定的UNIX国际标准POSIX的推出,UNIX用户日益增加,但使用它的人都感到的UNIX/XENIX环境中缺少一个诸如DOS用户使用的PCTOOLS这样的软件工具来恢复被删除的文件,一般用户都认为,文件一旦被删除则认为永久性的损坏,其实,这种认识是不对的,在UNIX/XENIX环境下,依然可以用一定的手段将被误删除文件恢复起来。

一、UNIX/XENIX文件系统结构

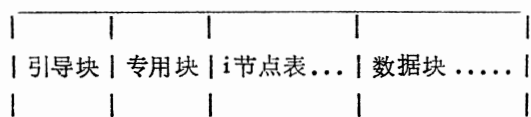
UNIX/XENIX支持一个树型的文件系统,其典型结构如下:



```
start3: nop
        call hd
        or al, 20h
        cmp al, 'q'
        jz exit
        jmp start
```

在UNIX/XENIX系统中,对外部设备、目录等均按文件处理,将其分为设备文件、目录文件和普通文件。如细分则可分为字符设备文件、块设备文件、可执行文件、正文文件等,对于每一个文件其内部表示则通过一个i节点来给出,一个i节点包括了文件数据大小、磁盘文件布局及有关的说明。

一般UNIX/XENIX文件系统具有如下结构:



其中引导块占一个物理块,专用块占用一个物理块,i节点表占用物理块数和数据块占用物理块在系统建立时确定。

UNIX/XENIX文件系统对建立和删除文件的管理采用如下方式,如果用户建立一个新文件,则UNIX通过creat()系统调用查询空闲i节点表,看是否有空闲的i节点,如果有空闲的i节点,则将这个空闲i节点分配给该文件,有了文件i节点,再查询空闲数据块表,如有空闲数据块,则将该空闲块分配给该文件正文部分,一块不够,则依次将各块分配给该文件,物理块大小,一般定为512字节的倍数(本文中的所有程序均按一

```
        jmp start2
exit: ret
start endp
code ends
end start
```

个物理块1024字节计算)。当我们删除一个文件时,系统首先找到该文件对应的i节点,然后通过i节点中数据块地址找到对应的数据块,将其链接到空闲数据块表中,再释放i节点,即将其链入空闲i节点表中,在此过程中,除将用户目录中该文件的i节点号置0和清理i节点外,并没有对数据块的内容进行修改,除非i节点和数据块已经被另行分配给其它的新文件了,否则我们就有可能根据物理盘区上的内容将文件恢复起来,这个前提和DOS环境下是一致的,也就是误删除文件后,未对该盘或该文件系统进行写操作。

我们知道,如果我们已知一个文件的i节点号,则我们可以用下面的命令:

```
lseek ( fd, 数据块大小×2+64× ( i  
节点号-1 ), 0 )
```

将文件指针指向i节点表中该文件实际i节点位置,从中读取64字节内容(每个i节点占64个字节),即为i节点内容,根据i节点内容,我们可以找到数据块号,从而可找到磁盘文件的具体内容。

经过上面的分析我们可以看出,恢复被误删除的文件要经过三个步骤:

1. 得到被删除文件的i节点号。
2. 通过i节点得到被删除文件的在盘区中的绝对地址。
3. 通过绝对地址读出被删文件的内容。

下面我们分别对其进行介绍。

二、如何得到文件的i节点号

通过上面的分析我们显然可以看出,得到一个被删除文件的i节点号成为问题的关键,我们知道,一个文件一旦被删除后,其i节点被释放了,其i节点号也无法得知,所以必须在文件被删除前或被删除时将其i节点号记录下来,以备恢复时用。

我们知道,用od -c命令可得到当前目录的文件目录表,而文件目录表中则记录着

文件的i节点号和文件名,因而我们可以另建一个删除文件命令del,并将其放在/usr/bin目录下,其内容如下:

```
od -c > inode  
rm $1
```

以后我们删除文件时就使用del命令,这样每次删除文件前,我们在当前目录的inode文件中保存了当前目录的文件目录表,从而根据该文件,可以看到被删除文件的i节点号和文件名,就可以根据其i节点号作恢复文件的准备工作了,由于该文件带有前缀,所以一般列目录命令无法看到该文件名,不影响用户的使用习惯。

采用od -c命令,得到的i节点号是用ASCII方式解释的,而我们一般习惯于用十进制表示法,并且我们取i节点内容时所用lseek中的i节点号也是用十进制数表示的,如果不用od -c命令,而采用od -d命令,i节点号是用十进制数表示的,但文件名也是用十进制数表示的,显然这是很不方便的,为此我们设计了如下的小程序:

```
#include<stdio.h>  
#include<sys/types.h>  
#include<sys/ino.h>  
#include<sys/dir.h>  
main ( )  
{  
int fd;  
struct dinode inodel;  
fd=open(".", 0);  
while (read(fd, &inodel, sizeof(inodel))!=0)  
printf ("%d\t", inodel.i_d_ino);  
printf ("%s\n", inodel.i_d_name);  
close (fd);  
}
```

这样就可以得到十进制表示的i节点号和字符方式表示的文件名。

三、如何得到被删文件的数据区的绝对地址

根据前面的介绍,我们已经可以知道被删除文件的*i*节点号,分析UNIX/XENIX的文件系统我们可以得知,*i*节点号只能帮助我们知道*i*节点的内容,并不能直接通过*i*节点号来得到文件的实际内容,但是我们只要有了*i*节点号就可以得到*i*节点中的具体内容,通过分析*i*节点的数据结构我们就可以得到文件数据块号,根据数据块号,就可以读取该块内容,即为删除文件的内容。我们可以用以下的程序来读取*i*节点的内容:

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/ino.h>
main()
{
    int fd, i, *ip;
    char s[4];
    struct dinode add;
    fd=open("/dev/root", 0);
    lseek(fd, i节点号-1)×64+2048, 0);
    read(fd, &add, sizeof(add));
    printf("%d\n", add.di_size);
    for(i=0; i<=38; i++) {
        s[i%3]=add.di_addr[i];
        if(i%3==0){
            s[3]=0;
            ip=(int *)s;
            printf("%d\n", ip[0]);
        }
    }
}
```

有了上面的程序我们可以很方便的得到文件的绝对地址。此项工作应在文件被删除之前进行,该程序可与前一程序联合起来使用,构造成新的del命令,这样在inode文件中,我们就有了被删除文件的*i*节点号、总字节数、和十三个数据块地址。一旦有了文件正文的绝对地址,我们就可以着手下面恢复文件的工作了。

四、通过绝对地址读出被删文件的内容

有了文件数据块的数据块号,就可以计算出该块的物理地址,我们可以用c语言很方便的读出文件的正文,其程序如下:

```
#include<stdio.h>
main()
{
    char buff[1024];
    int fd;
    fd=open("/dev/root", 0);
    lseek(fd, 1024×数据块号, 0);
    read(fd, buf, 1024);
    write(1, buf, 1024);
    close(fd);
}
```

这样就读出了一个数据块的内容,用户可将其写入另一文件中,依次将全文读完,被删除的文件也就恢复起来了。在这里我们要注意以下问题:

在不同的机器、不同版本的UNIX/XENIX上,其*i*节点的数据结构可能略有不同,用户可查阅相应的数据结构,如有些机器上用40个字节来表示十三个数据块地址,每个地址占三个字节,转换时要注意,在求地址程序中我们对此进行了处理,另外在*i*节点中只有十三个数据块地址,也就是说直接寻址最大也就是13K字节,实际上只有前十个地址是直接块地址,超过直接寻址能力的文件将采用间接寻址,我们可以根据*i*节点数据得到文件的大小,从而可以判断出是否需蚤间址。如果需要间址,在十三个数据块地片中就会有一个或一个以上的地址号不是该文件的数据块号,而是一个地址块号,如常用第十一个地址表示一级间址,如果采用四字节表示一个地址的话,一个地址块1024字节则可以表示256个数据块,也就是文件容量可以增加256K,依次类推。如果一级间址仍不够用的话,也可以有二级间址、三级间址,但这种情况是很少见的,因为用户文件一般多在几K至几十K之间。

远距离多点大规模数据采集系统设计的研究

天津纺织工学院 傅景义 贾景云 耿仁义 陈照亮

摘要 本文通过以8031单片机为CPU构成的远距离多点粮库测温系统的实例,论述分析传输电路的硬件设计,并引伸推广以任何并行输出端口进行串行输出的设计方法。

一、问题的提出

粮库测温系统是多点大规模数据采集系统,当微机系统远离库区或库区过大时,即形成远距离多点大规模数据采集系统。如图1所示。

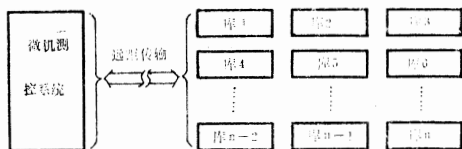


图1 远距离多点大规模数据采集系统示意

系统设计的关键是选点,即如何唯一确定地选通任一测量点,取入该点的瞬时状态数据值送至微机指定存储区。

常规的选点方式是译码方式,借助于外部译码器实现多点选择。但随着点数的增多,要增加译码输入的线数,以至传输线增多,加大了成本。

采用串行输出选点方式,则大大节省了传输线路的投资,且对远距离传输还可采用电流环电路,从而增强了系统的抗干扰性能。

二、远距离多点大规模数据采集硬件系统

以粮库测温系统为例,将每个库定义为一个数据采集区,整个系统为若干区的集合。各个区之间,采用串联形式。由微机侧引出五条控制线,分别为+Vcc、GND…选点发送数据DATA,选点发送移位脉冲 ϕ ,及模拟量传输线AS。

每个区的选点检测电路视点数不同而异,以每区40点为例:由一片8位移位寄存器74LS164为主接收芯片。采用五片CD4051八选一模拟通道开关芯片为辅构成选点检测电路。系统原理图如图2所示。

写文件时最好一次将所有数据块读入内存或将结果写入另一个系统,如/usr文件系统中,以防止新文件的数据块与已删除并正在恢复文件的数据块重复,造成数据覆盖而丢失数据。

采用上面介绍的方法,虽然给我们提供了一个恢复UNIX/XENIX系统中误删除文件的手段,相对而言,这还是比较麻烦的,但有了这种思想我们可以开发相应的工具软件来做上述工作,本文只是给出思路,限于篇幅,不再做更进一步的讨论。

另外,我们也可以采用从空闲i节点表中将被删文件的i节点号删除,将其i节点号重新填入文件目录表中,重新恢复i节点的

内容,然后将已释放的数据块重新申请回来,即从空闲数据块表中删除这些被该文件占用的数据块号,用这种方法也可以将被误删除的文件恢复起来,而且看起来更正统一些,但这种方式仍然要保存被删文件的i节点号和i节点的值,同时它对用户的要求更高,要求用户对UNIX/XENIX文件系统和系统调用有更深入的理解,否则一些误操作会损坏文件系统。上述方法我们在多种机型上的UNIX/XENIX通过,实践证明,这是一种十分有效的方法,有兴趣的读者不妨一试。

参考文献

- 1 《UNIX分时系统程序员手册》
- 2 《UNIX操作系统分析报告》刘日升、孙玉方
- 3 《C语言程序设计基础》史美林、苏云清

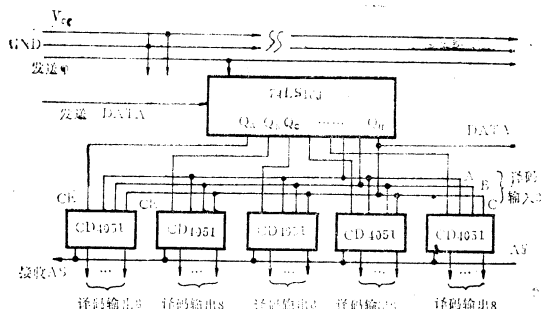


图2 多点数据采集各区硬件组成电路图

模拟量传输线AS,将被选通点处的瞬时状态值传送至微机A/D转换器进行A/D转换。选点发送数据DATA和选点发送移位脉冲 ϕ 由微机串行输出端口输出,对8031单片机,即为方式0移位寄存器工作方式,由P30和P31两引脚输出。其波形图如图3所示。

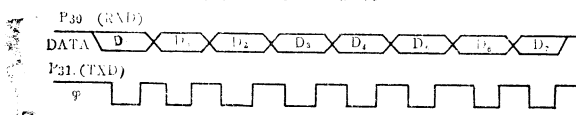


图3 移位寄存器式串行输出波形图

为了抑制干扰信号,一般对选点发送数据DATA和选点发送移位脉冲 ϕ 经光电隔离器隔离后送出。当距离较远时,采用电流环电路:

此时由微机串行输出端口引出的信号称为发送DATA和发送 ϕ ,至检测区接收到的信号称为选点DATA和选点 ϕ 。两信号的隔离与传送方式基本一样。仅是对 ϕ 信号在选区较多时,要附加驱动环节。而DATA信号系由各级74LS164接续传递,故无需附加驱动。

由于8031工作于方式0时,其波特率固定为主时钟频率 f_{osc} 的十二分之一。对 $f_{osc} = 6\text{MHz}$ 时。其波特率为500KHz。因而不能使光电隔离器件有效跟随并使电流环正常工作。

为了使电流环及光电隔离器件正常工作,同时考虑到用户系统不需要高速传输的特点,采用人工软件改变传输波特率。

三、远距离多点大规模数据采集软件系统

1. 一位数据信息的发送

为了满足接收寄存器74LS164的正常工作,必须按照图3所示的波形时序发送数据DATA和移位脉冲 ϕ ,仅在速度上放慢,为此,采用程序指令输出方式。仍以8031的P30作DATA, P31作 ϕ 线输出为例,其程序流程如图4所示。

以发送一位信息“1”为例,且移位寄存器为下降沿锁存。程序如下:

```

:
SETB P31; 置 $\phi$ 初态
SETB P30; 发送“1”
CALL 延时1; 稳定DATA
CLR P31; 发移位脉冲 $\phi$ 
CALL 延时1; 2倍延时1
SETB P31; 一位信息发送完
CALL 延时1; 变更下一位信息
:

```

2. 一字节数据信息的发送

发送一个字节信息的实质,是进行八次发送一位信息,因而程序予设计计数器常数为8。将待发送字节由存储区取至A中,进行移位判断,确定DATA状态后,调用发送一位信息的子程序,从而完成一字节信息发送,流程如图5所示。

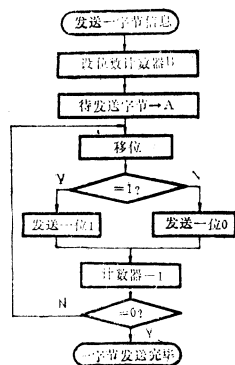
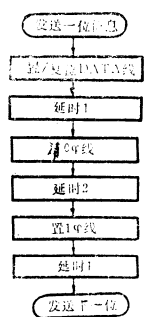


图4 发送一位信息流程 图5 发送一字节信息流程

3. 选点数据信息的发送

(1) 被选通区接收字节信息特征分析
设有n区进行数据采集,如图2所示硬件系统,则非选通区接收的字节信息必为0FFH。而被选通区接收字节信息 $Q_A \sim Q_E$ 之中仅有一位为0,其余为1, $Q_F \sim Q_H$ 状态在 $Q_A \sim Q_E$ 不变

的情况下,由000B变化至111B,从而对一片模拟量开关电路CD4051进行依次选通各个通道,随后,将 $Q_A \sim Q_E$ 之中的“0”状态移动一位,重复上述的 $Q_F \sim Q_H$ 变化过程,直至 $Q_A \sim Q_E$ 均经过“0”态,表明该区的40路模拟通道已依次被选通一次,因而,对 $Q_F \sim Q_H$ 的处理为加一处理方式,而对 $Q_A \sim Q_E$ 的处理为左移位处理方式。

(2) 选点数据信息特征分析

由于非选通区接收的字节应一律为FFH,故每次发送的选点数据信息为若干个FFH和一个选通区字节信息。

设数据采集区数为 n ,被选通区距第 n 区(最后一区)距离为 K ,则 $K_{\min}=0$,即被选通区为第 n 区, $K_{\max}=n-1$,即选通首区。

由此可知,选点数据信息的组成为图6所示。

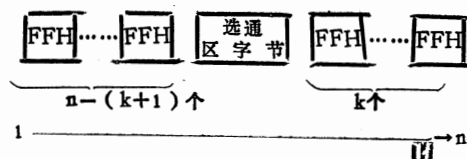


图 6

当 $K=0$ 时,首发选通区字节,随后发出 $(n-1)$ 个FFH。

当 $K=n-1$ 时,首发 $(n-1)$ 个FFH,随后发出选通区字节。

当 $0 < K < (n-1)$ 时,首发 K 个字节FFH,随后发出选通区字节,再发 $n-(K+1)$ 个字节FFH。

通过上述的分析可知,无论选区数 n 是多少,对于选点数据信息在存储区内所占的字节数并无关系,而仅仅影响在每次选通时,所发送的FFH的个数。

(3) 指定区点号的选点数据信息发送

在数据采集过程中,若指定要求采集某一区点号的瞬时状态时,实质是对指定的一片模拟通道开关CD4051中某一通道的选通。因而软件系统的任务是:

a. 根据区点号求出 K 值。

b. 根据区点号求出选通区字节内容,即

选通字,

c. 调用选通数据发送程序并执行一次。

求取 K 值和选通字的方式很多,可通过计算或查表进行,在此不一一赘述。

(4) 循环检测时选点数据信息发送:

在大规模数据采集系统中,应用最为广泛的是循环(即巡回)检测方式,由第 n 区的最后一点,直至首区的第一点,依次选通。为此系统软件设计方案为:

置 K 初值为0;

置选通区字节 $Q_A \sim Q_H$ 状态为11110000B

调用选通数据发送程序。即

发送0个FFH; K 个FFH

发送选通字;

发送 $n-(0+1)$ 个FFH; $(n-1)$ 个FFH

调整选通字:如前述 $Q_F \sim Q_H$ 按加1处理,经8次后 $Q_A \sim Q_E$ 按移位方式处理,经5次结束。

置 $K=K+1$ 并判断 $K=n-1$?

依次类推,直至全部选通完毕。

采用上述的硬软件系统,以最廉的硬件成本;最少的干线传输;可靠的抗干扰能力;简明易懂的软件结构,构成远距离多点大规模数据采集系统。A/D转换器只需一个通道即可依次读取各点的模拟量值。

尤为可引伸推广的是,对没有串行输出端口的微机系统,如常用的8039、Z80等,可通过并行输出端口的任意两条线,参照本文介绍的方式,实现串行选点数据输出。

系统的缺点在于点数越多,则每次选通发送的位数也增多,从而使速度降低,在本来就较低的波特率情况下,显得速度不大理想。但对于如粮库测温系统等非实时控制及大惯性系统的数据采集,具有其一定的实用价值和优良的性能价格比,对于发送每一位数据信息时,控制 ϕ 的上升沿与下降沿之间的时间间隔。需依据所选用的光电隔离器件响应速度来确定延时子程序的时间值。巡检方式中选通字的调整为二重循环程序结构。

微机在化工设备制造工艺性补图中的应用

燕山石油化工公司机械厂

华振明

摘要 本文对微机应用在化工设备制造工艺性补图中的必要性、可能性等作出了分析,并给出了整个系统的设计方案。此系统于88年予以实现,并获得了很好的经济效益和社会效益。

一、引言

在化工设备的制造中,标准零部件的制造图纸,设计单位往往是不提供的,由制造单位根据“化工设备标准手册”及其它有关标准自己制图,其工作量很大。以燕山石油化工公司机械厂为例,按每年生产4000吨化工设备计算,每年需补50000张零部件图。而这些图纸以前是完全靠技术人员手工完成的。

人工制图存在的问题是:

1.工作效率低,故任务多时需要加班加点突击完成。另外由于技术人员做的是大量的机械式的重复劳动,故大家感到枯燥心烦。

2.准确率低。由于人工画图,图纸的准确率只能达到95%,而图纸一旦有错误,零部件就成了废品,给企业造成很大的经济损失。

为此,提出将这部分工作由计算机来完成,对于提高企业的经济效益无疑是很有意义的。

二、系统设计

1.基本思想

把常用的标准零部件图存放在图形库中,把每种图形及工作压力所对应的各种尺寸参数存放在数据库中,这样,当用户给出一个图代号和主参数后,计算机根据图代号从图形库中检索出所需的图形;根据主参数从数据库中确定其它参数,把这一组参数加在所需的图形上,就可输出所需的图纸。如图1所示:

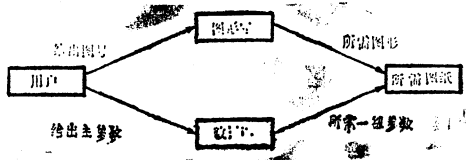


图1

而目前微型计算机对图形、数据的存储和处理都已解决,图形的输出也已解决,并且具备较强的汉字功能,更提供了方便。

2.系统模块划分

系统模块划分如图2所示。

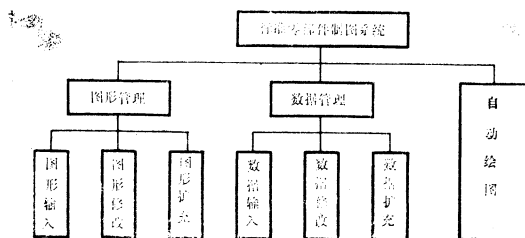


图2

3.系统性能要求

(1)自动化程度高。由用户任意选定一种零部件,给出主参数后,不用人工干预,计算机自动输出所需的图纸。

(2)处理速度快。从用户选定图形、给出主参数到图纸输出几分钟完成。

(3)准确率高。只要人工不发生错误,系统输出的图纸准确率应100%。

(4)操作简单。一般技术人员,不懂得计算机内部结构也能很快掌握。

4.图形库的建立

化工设备标准零部件图有几十种,必须输入到计算机内。输入的方法有二种。

(1) 采用扫描仪

优点: 速度快

缺点: a: 需要价格昂贵的扫描仪。

b: 图形所占的磁盘空间大。

c: 图形上不能设置尺寸变量参数。

故这种方法无法采用。

(2) 采用AUTOCAD软件包AUTOCDA是一个在微机上作图功能较强的软件包, 它具有如下优点:

a. 绘图功能强。它提供的基本绘图命令已满足我们绘制标准零部件图的要求, 此外还提供了尺寸标注。

b. 软件开放好。AUTO CAD 能为用户输出几种格式的数据形式和接口, 以供高级语言调用。系统中的图形数据可用FORTRAN BASIC等高级语言及其它实用程序(如DBASE等)对其进行处理, 处理结果可以返回AUTO CAD中, 重新生成图形。这为图形调用数据文件的数据提供了方便。

c. 硬件配置通用性好。AUTO CAD 硬件运行环境较为通用, 一般的微机PC/XT或AT及其兼容机都可运行。

d. 利用它绘制的图形可以以文件形式存放磁盘中, 可以随时调用。

e. 图形文件所占的磁盘空间较小。一个零部件图约占 20K。故整个系统一般微机10M的硬盘足够用了。

根据以上分析, 决定采用AUTO CAD 输入零部件图形。

5. 数据库的建立及数据结构

每种零部件, 在一定的参数下都对应着一系列其它参数, 必须把这些数据存放在计算机内, 以便系统在使用时, 当用户选定一种图形和给出公称参数后, 能检索出其它的一系列参数。

存入数据可以采用大家较为熟悉的DBASE, 也可采用高级语言的数据文件。采用DBASE 的优点: 查询数据, 修改数据方便(不需编程), 但向图形文件传递数据的

接口复杂; 而BASIC语言的数据文件, 数据的输入、查询。修改较为不便(需编程), 但向图形文件传递数据的接口简单。

故决定采用BASIC语言的数据文件。

数据文件的结构, 基本上参照“化工设备标准手册”的表格结构。

以对焊法兰为例:

表格结构如表1所示。表2为数据文件的结构, (见下页)。

6. 接口设计

这是整个系统实现的关键问题。

由于一种图形对应着几种尺寸参数, 故应把图形上的尺寸参数做成变量, 它由用户给定的主参数从数据库中检索确定。但AUTOCAD 本身没有存放数据的数据文件, 只有高级语言才有数据文件。由前知, 我们把数据存放在 BASIC 的数据文件中, 这就要解决 BASIC 数据文件中的数据向图形文件传递的问题。也就是解决 AUTO CAD 与 BASIC 语言程序的接口问题。

AUTO CAD 与高级语言之间的接口方法, 目前较常用的有“DXF”和“SCR”方法。由于“OXF”方法使用时限制多、转换速度较慢、效率较低、需要编写很多的程序; 而“SCR”方法接口简单, 故决定采用“SCR”方法。

7. 输入输出设计

(1) 输入

人机对话采用汉字屏幕提示方式, 由用户通过键盘选择图形和给出主参数。另外, 考虑到用户在输入过程中产生误输入, 故在程序中应增加纠错措施。

(2) 输出

图形的输出, AUTO CAD 本身已解决, 现讨论采用什么输出设备问题。

图形的输出常用有绘图仪和打印机, 考虑到绘图仪使用消费较大(常需要更换绘图笔, 而图笔较贵); 操作不方便(需经常换笔和换纸); 速度较慢(特别是图纸中技术条件说明等都是汉字, 用绘图笔描制速度很

慢)和不能一机多用。故决定采用普通打字机(M1724),普通打印机有图形软件支持,输出的图纸和绘图仪绘制的图纸一样很漂亮,同一台打印机既可作图形输出机,又可作管理字符打印机,做到一机多用,减少系统成本。采用打印机的另一个优点,可以采用多层涂碳打印纸,一次可输出4份图纸,满足实际要求,提高工作效率。打印机更换一条色带才几元钱。

8.运行设计

为便于用户操作(在整个过程中尽量减少人工干预),和提高整个系统的运行速度,整个绘图过程由批处理文件统调。

三、程序设计

表2

名 称	代号	类型	长度
公称直径	A\$	C	4
颈部外径	B\$	C	4
内 径	C\$	C	4
外 径	D\$	C	4
螺栓孔中心圆直径	E\$	C	4
连接凸出部分直径	F\$	C	4
连接凸出部分高度	G\$	C	4
法兰厚度	H\$	C	4
法兰高度	K\$	C	4
颈部最大直径	T\$	C	4
圆弧半径	M\$	C	4
螺栓孔直径	N\$	C	4

表1 mm

Pg=6kg/cm ²															
公称通径	管子											螺栓		法兰理论重量 (比重7.85 kg)	
	外 径	颈 部 外 径	内 径	外 径	螺栓孔中心圆直径	连接凸出部分直径	连接凸出部分高度	法兰厚度	法兰高度	颈 部 最大直径	圆 弧 半径	螺栓孔直径	数 量	螺 纹	≈
	D _g	d _o	D _n	d _i	D	D ₁	D ₂	f	b	h	D _m	r	d		
10	14	15	8	75	50	32	2	12	25	22	4	12	4	M10	0.344
15	18	19	12	80	55	40	2	12	30	28	4	12	4	M10	0.402
20	25	26	18	90	65	50	2	12	32	36	4	12	4	M10	0.534
25	32	33	25	100	75	60	2	14	32	42	4	12	4	M10	0.777
32	38	39	31	120	90	70	2	14	35	50	4	14	4	M12	1.08
40	45	46	38	130	100	80	3	14	38	60	4	14	4	M12	1.22
50	57	58	49	140	110	90	3	14	38	70	4	14	4	M12	1.41

软件设计的内容较多,不能做详细说明,只能把软件设计中较为主要的部分加以叙述。

1. 图形文件的建立

采用AUTO CAD的绘图功能,将常用的零部件图做成图形文件,存放在A218\tu子目录中,形成一个图形库。图形文件的命名方法如下:

LO XX XX
图号

2. 数据文件的建立

用BASIC生成数据文件的方法生成的数据文件存放在A218\DATA子目录中,形成一个数据库。

数据文件的命名方法如下:

DATA XX XX
公称压力
图号

数据文件生成框图如图3。

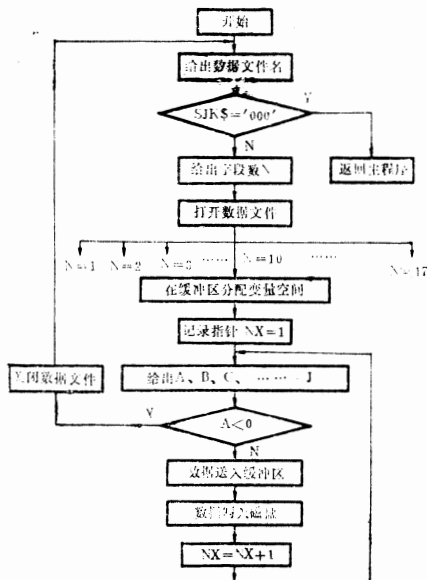


图3.

3. 接口生成程序

从数据文件向图形文件传递的数据是由执行接口生成程序来解决。

对接口生成程序应具有如下几个基本功

能:

(1) 由用户给出的图号,能打开相应的数据文件。

(2) 由用户给出的主参数,能从数据文件中检索出图纸上需要的其它参数。

(3) 把一系列参数组成 AUTO CAD 能识别读取的SCR文件。

接口生成程序文件名的命名方法如下:

PF XX
图号

4. 主控程序

整个系统由主控程序统调,主程序结构如图4。

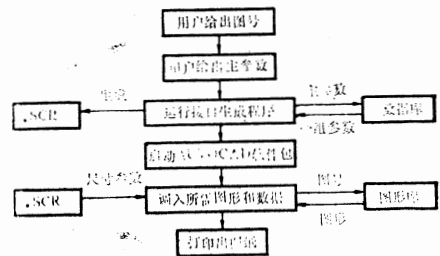


图4

四、结束语

本系统于88年由助理工程师薛生贵、杨解生、赵功雨等予以实现。提高功效近6倍,准确率达100%。该系统89年2月通过了中国石化总公司的鉴定,同年4月在总公司范围内推广,并获中国石油化工总公司89年度优秀计算机软件及应用成果三等奖。

但该系统在运行速度上还须进一步提高。

参考文献

- 1 微型计算机硬件软件及其应用 清华大学出版社。 1987年。
- 2 AUTO CAD计算机绘图软件 中国科学院希望公司。 1988年。

打印机图文输出的实现

苏州市经济信息中心 黄明

摘要 本文介绍了利用打印机替代绘图仪输出图形和文字的一种设计思想,重点说明了图文缓冲区的组织和计算方法,以及在此基础上设计图文输出控制命令的具体过程。由于选择了合理的数据结构和算法,根据这思想实现的作图软件,在图文打印控制方法、图形输出质量、运行时效等方面,都有较大的改进,具有一定的实用价值。

我们将结合TURBO PASCAL 4.0 的语言特色,着重针对上述几个存在的问题,介绍一种实现的思路和方法。

一、引言

以图形表示数据,在信息管理系统中有一定的意义,与数字表式相比,图表具有简洁、直观的特点,很受管理人员的重视。计算机的图表输出有两种形式:屏幕显示和硬拷贝输出。屏幕上作出的图形,作为硬拷贝输出的纸上作图,一般需要连接绘图仪等硬件装置。如果能以普通联机的字符行输出设备——打印机,替代绘图仪在图纸上输出图形和文字,则能充分发挥打印机的效能,降低联机系统费用,满足用户的硬输出要求。

打印机替代绘图仪的方法在一些文章中已有所介绍,但其中有几个不足之处:1.输出图形的数据缓冲区,因为数据结构不甚合理,占用的存贮区过大,限制了图形的输出面积;2.在打印机设置成位图状态时,只可以输出图形,而无法打印中文汉字和ASCII字符;3.发往打印机的作图命令不够完善,例如,通过数组传送打印点阵的信息,或是以一个函数形式计算打印像素在缓冲区中的位置,控制打印机的方式不方便;4.没有考虑输出图形的形状分布,在某一个坐标轴方向,由于打印像素坐标值改变的梯度过大,往往使点阵分布不均匀,这就影响了输出图形的美观。

二、设计思想

TURBO PASCAL 4.0中的图形标准单元GRAPH,提供了很强的屏幕作图功能,用户调用这个单元,可以在屏幕上绘制多种图形。我们借鉴了屏幕作图的形式,由一系列子程序实现作图参数的设置、图文缓冲区的赋值及排序、缓冲区内容输出等操作,这些子程序及相关部分组成一个图文输出单元,通过“接口”供用户调用。模仿TURBO PASCAL 4.0中屏幕作图的特点,用户可以根据打印机与屏幕上作图的相似性,方便地设置打印参数,使用图文输出命令。

由于打印机既要输出图形,又要打印文字,因此,设计中开辟了图、文两个缓冲区,对此分别进行操作相同但内容不同的处理,抽象处理的过程如图1所示。

图文输出的实现过程由五个部分组成。

1.开辟缓冲区及初始化处理 用作图、文缓冲区的数据结构是两个类型不同的动态内存变量构成的链表,因此,建立两个新的动态变量,并分别设置指向它们的指针,作为图、文缓冲区的首地址。

2. 作图打印命令设置 这类命令由自定义的图文输出单元中的过程实现,通过设定各作图命令,向缓冲区传送像素点坐标值、文字信息等参数。这类作图命令包括画点、画线、画块、画圆和输出文字等数条。

3. 输出坐标值计算并插入缓冲区 这部分是实现打印参数到缓冲区的过渡,对于每一条作图打印命令,都要通过这个过程把命令中的输出内容按计算得到的坐标值顺序插至缓冲区的相应位置。这一部分完全由程序内部实现,无需用户的命令干预。

4. 图、文缓冲区的内容打印输出 图形以点阵形式输出,文字打印则利用操作系统的字符输出功能实现,图表中点阵和文字坐标值的计算单位是不同的。因联机的打印设备是24针点阵式打印机,所以图形缓冲区中的坐标需要按行坐标值进行以24为单位的数值转换,使打印机完成一次行输出动作时,能实现24个点阵行的图形打印。同时,要判断当前打印行是否有文字打印,若有,则使打印针回车,并从文字缓冲区中读出内容,输出注释文字。

5. 缓冲区释放 分别清除图形、文字缓冲区占用的堆栈段内存。

上述是图文输出的设计思想,实现过程中有关程序设计的一些技巧和方法,将在下面分别予以介绍。

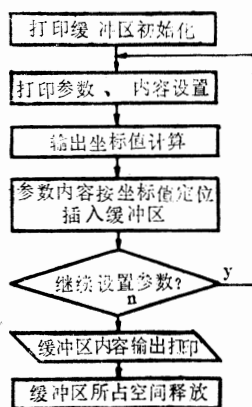


图1 图形(文字)缓冲区上的抽象处理过程

三、缓冲区数据结构和计算

缓冲区数据结构选择的好坏将影响可使用的缓冲区容量,直接限制了输出图形点阵最大行、列坐标的取值。我们拟用动态链表结构作为缓冲区,因此,它占用的是 TURBO PASCAL 存贮映象中的堆存贮区。

图形中某个点的含义是指在坐标系中确定的位置上是否置一个像素,它没有相对的输出内容可言,只需以一个二元坐标向量表示;一幅图形就能用一个元素值或为“1”、或为“0”的二维数组表征。从需要输出的图形分析,其中大部分图形的空白点要多于置像素点,因此,作为一个重要的特征,可以把图形的数组认为是一个二维的稀疏矩阵。

我们用伪地址法表示稀疏矩阵中值为“1”的元素行、列向量,即图形中的像素坐标,因而,动态单链表中只需一个整数数据域即可表示一个像素点。伪地址的计算式如下:

$$M = (X-1) \times n + Y$$

其中, X 、 Y 为像素点坐标值, M 为 (X, Y) 点的伪地址值, n 为矩阵列向量的最大个数。在程序设计中,图形数据域中的伪地址值用长整数 Longint 类型定义。由此可见,一幅图形中置像素点的少量元素用稀疏矩阵中的伪地址法表示,比所有的图形元素用二元法放入缓冲区,要节省许多内存空间。同理,如果在处理的图形中置像素点多于空白点,则令缓冲区中的元素为空白点地址,在缓冲区内容输出时,位图变量值作逻辑非(not)运算后打印即可。

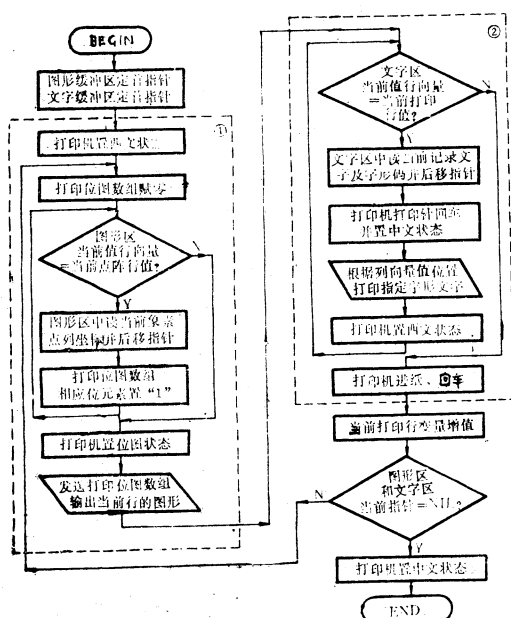
图形缓冲区的链表按照伪地址数值的升序排列。作图语句设置的各个坐标参数,在计算、排序后,逐一插入已存在的链表中。这是在缓冲区链表上的写入操作,它实际包括坐标值转换、数值排序和记录插入三个步骤。

文字缓冲区的数据结构也是一个动态链表,其中存贮的是值为“1”的稀疏矩阵元素的伪地址值和文字内容,作为矩阵元素行

列向量的原型则是输出字符串的起始行列坐标值。链表的构成与图形缓冲区相似——按坐标值伪地址升序排列的单链结构，但因输出文字还包含文字的内容和字形控制码，所以链表的数据域由三项组成：伪地址值，长整型；文字内容，字符串型；字形控制码，子界型。

四、缓冲区内内容输出打印

图文输出的特征是文字可以作为图形的标题或注释一起输出，使图形、文字相互覆盖。在设计中，图形输出是控制打印机处于位图状态 (Bit Image Print) 实现，文字则是调用操作系统的字符打印功能输出。因此，若是在中文操作系统的支持下，则可以直接打印中文汉字。其设计的关键在于打印机完成了一次24点阵行的输出后，需要根据文字缓冲区中的内容决定，是使打印针回车输出文字信息，还是进纸继续下一行的图形输出。输出打印的处理流程如图2所示。



①图形输出处理部分

②文字输出处理部分

图2 图文输出打印的处理流程图

上述处理过程中，有几点需要加以说明。

1. 打印位图数组反映了以24个点阵行、 n 个点阵列为单位的区域内图形象素的分布情况，作为打印图形的点阵控制码发往打印机，它是二维字节型数组，类型定义为：

$BI = \text{Array}[0..n-1, 0..2]$ of Byte
(n 为输出图形点阵的最大列数)

数组的一维下标表示打印针列号，二维下标为打印位变量的序号。在第 X 列($0 \leq X \leq n-1$)上，有 $BI(X, 0)$ 、 $BI(X, 1)$ 、 $BI(X, 2)$ 三个字节(Byte)，共24位(Bit)，它们的取值分别对应着打印机一列24个针点的落针状态，这与打印机的硬件参数相一致。我们设定输出图形的空白点多于象素点，因此，打印位图数组所有的初值都赋零。

2. 图形缓冲区的读出是基于稀疏矩阵存贮方式的操作，其中存放的是由象素点坐标计算得出的伪地址值。因缓冲区是升序链表结构，在输出图形的第 i 打印行($i=1, 2, \dots$)位图数组赋值时，若正在处理位图数据的第 p 位($0 \leq p \leq 23$)，则判断链表当前指针关键字 Key 是否满足条件：

$$Key \leq ((i-1) \times 24 + p) \times n$$

若条件成立，则读出 key 值，按伪地址写入缓冲区的计算式逆运算，求得象素点坐标的列向量 j ，以下列计算式实现一个象素点的位赋值操作：

$$\begin{aligned} \text{令 } a &= p \div 8, b = p \bmod 8 \\ \text{则 } BI[j, a] &= BI[j, a] + (1 \text{ shl} \\ &\quad (7-b)) \end{aligned}$$

其中： \div —整除， \bmod —取余数， shl —逻辑左移

然后，缓冲区指针后移，继续判断条件，如此反复，直至条件不成立或者链表历遍完毕，完成一个点阵行的内容输出。因而，图形缓冲区的读出操作是以点阵行、列为逻辑循环单位进行的。

3. 文字缓冲区的读出，在每次位图数组赋值完毕、打印输出后进行，它以打印行(24个点阵行)为基本单位；同理，一个打印列为24个点阵列。文字缓冲区中当前指针关键

字的地址范围判断与图形区类似,只是求得的列向量,作为输出字符串的前导空格数目,与文字等信息一起送往打印机输出。

4.图文输出过程中的一些控制命令与联机的打印机型号和使用的操作系统有关。我们以M-1724点阵打印机和CCDOS V4.0为例予以介绍,其它种类的机器和系统请参阅各自的使用说明书。其一,打印机中西文状态切换命令,这与操作系统有关,控制序列为“Esc IZ”,这是因为位图打印必须要在西文状态下进行,而后又要输出中文;其二,位图打印状态控制命令“Esc G ab”,其中a、b为最大输出列限制符,算式与要求如下:

$$0 \leq \text{最大输出列值 } n = (a \times 256 + b) \leq 2176$$

在设置成位图状态后,循环 n 次,每一次用WRTE语句发送3个字节变量的控制码,可完成一个 $24 \times n$ 区域内的像素点打印;其三,进纸行距控制命令“Esc Jc”,设置的是 $c/120$ 英寸的进纸距离,我们选定 c 值为18,此时图形能完成拼接。

图文打印处理过程是整个设计的关键。

五、图文输出 TURBO PASCAL 4.0的具体实现

图文输出面向用户的整个过程是完全透明的,用户只需通过提供的接口调用过程打印图文。我们用 TURBO PASCAL 4.0的“单元”结构组织这些程序,定义名为“P-GRAPH”的自定义单元为有关图文输出的常数、数据类型、过程和初始化程序体的集合,编译成“.TPU"文件。用户打印图形时,在主程序体首部的USES子句中列出PGRAPH单元名字就可使用。因而,用户不必涉及具体的程序体,屏蔽了功能实现的详细过程。

图3说明了 PGRAPH 单元中接口部分

Interface、实现部分Implementation 和初始化部分Initialization的组成内容。单元中的一些结构和编程方法介绍如下。

1.最大输出点阵行值和列值作为常数设定,根据程序运行时的内存分配和图形象素的分布率分析,可以打印一幅最大约为 2000×2000 点阵的图形。因而,这两个常数取值均为2000。

2.类型定义表示图文缓冲区和文字缓冲区的动态内存变量的记录定义,每个记录结构包括数据域和指针域两部分。

3.图文输出提供的命令由作图、打印、释放缓冲区三类过程构成。作图过程实现了设置象素点坐标、图形位置和文字起始行列及内容等参数,包括画点语句PPlot、画线语句PLine、画块语句PBar、画圆语句PCircle、画椭圆语句PEllipse以及文字输出语句 PText,而且可以设置图形的线型和填充模式等控制符,各语句调用与TURBO PASCAL 4.0屏幕作图语句的使用方法相似;打印过程把图、文缓冲区的内容在转换后发送至打印机输出,这是一个无参调用过程P-Print,其中用到的打印位图数组是这个过程的局部变量;释放缓冲区过程 PDispose也为无参过程,可在图文输出后调用,清除图形和文字缓冲区所占的内存空间。

4.为了提高作图参数插入缓冲区的运算速度,采用了双链表的排序合并算法,即对一作图语句中设置的各象素点参数按伪地址值进行预排序,组成辅链表,经过比较后插入缓冲区的主链表中。这个过程只改变两个链表中的指针,无需移动记录,且减少了比较次数,因此,执行算法所需要的附加空间和时间开销明显优于其它排序方法。过程Merge 实现这个功能。

5.在图形象素点坐标值的计算中,如果仅以X轴方向作自变量递增,会产生如图4所示点阵分布不均匀的现象。如果采用下述方法,能避免这种情况,即当图形在一个区间内某点的切线与X轴正方向夹角在 45° 至 135°

之间时,则以列向量 y 值递增求得坐标值;在处理圆、椭圆等圆锥曲线时,坐标值以圆心角自变量结合坐标轴自变量的递增方式计算。修正后的输出图形如图5所示。

图文输出的具体实现,与TURBO PASCAL 4.0语言的编程特点有关,其中还涉及到一些合适的计算方法。

六、结语

利用打印机图文输出,能够满足信息管理中一般的绘图需要。本文介绍的实现方法,经过实际应用表明,在输出图表质量、作图打印控制方法和运行时效等方面,是令人满意的。

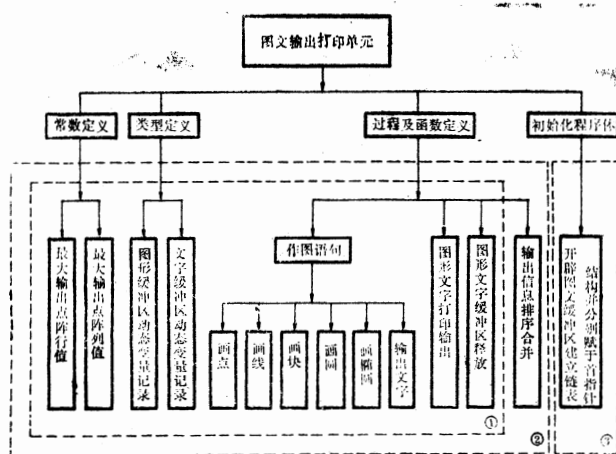
但是,这个实现方法也存在着某些可以改进之处。如:图文输出功能无法在其他语言环境中调用,也不能嵌入数据库管理软件

件;图形中的字符只可以按照操作系统设定的字形输出,不能进行无级缩放,虽然在读字模后写入图形缓冲区能解决这个问题,但无疑增加了运行时间;可移植性较差,实现方法在一定程度上依赖于打印机的型号和操作系统的种类。

打印机图文输出的实现是对信息管理工具的一个补充,图表打印将提高管理的质量。为了使这个方法更实用化,还需要作进一步的改进。

参考书目

- 1 TURBO PASCAL (4.0) 使用手册,中国软件技术公司 软件交易中心, 1988.12.
- 2 李启炎、宋秋杰, PASCAL 程序设计语言, 同济大学出版社, 1985.1.
- 3 许卓群、张乃孝等, 数据结构, 高等教育出版社, 1987.5.
- 4 沈演明、曾希君, 也谈用打印机绘制图形, 计算机世界月刊, 1989.10.



- ①接口部分内容
②实现部分内容
③初始化部分程序体

图3 图文输出打印单元PGRAPH结构图

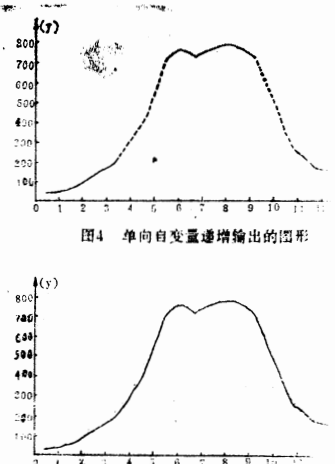


图4 单向自变量递增输出的图形

图5 修正后点阵均匀分布的图形

非标准硬盘驱动器的使用

商业部安徽商业管理干部学院

陈庆章

摘要 PC AT机所认可的硬盘驱动器类型有14种,这14种类型的参数表存放在ROM中。如果需要安装使用一个非标准的硬盘驱动器,由于与规定的标准参数不完全吻合,常常会导致硬盘的功能减少。本文从AT机启动后如何确定驱动器类型及相应的参数的内部处理过程入手,分析介绍了通过对驱动器类型参数表的改造以适应非标准硬盘驱动器的方法,并给出了具体实用的参数表数据和重新指定参数的设备驱动程序。

现在市场上出售的许多硬盘驱动器(以下简称驱动器)与IBM PC AT 所认可的标准相比,都可以说是非标准的。要安装使用这样一个驱动器,必须对驱动器进行修改以满足IBM所规定的14种驱动器类型。由于存在一些修改,用户通常不能完全利用被安装驱动器的全部功能,如存贮量的减少。本文通过分析 AT 机决定驱动器类型的内部处理过程,给出一个按规格改制的专门设备驱动程序,用它来调整驱动器类型参数以适合一个非标准的驱动器。由此,就能100% 的利用任何驱动器了。

一、传统的决定驱动器类型的方法

要决定一个新驱动器的类型和参数,可以把生产厂家的参数说明(即:柱面数、磁头数,为预先补偿柱面,定位区)与IBM的驱动器类型表加以对照(见表1)。如果新驱动器的所有说明与表1中的某一栏(也可称入口)完全相同,这个新驱动器就是标准驱动器。一般情况,是有些差异的。此时,应当从表中找出若干个与新驱动器参数最接近入口作为考虑对象,根据这些被选定的入口,找出一个柱面数是最接近的(入口的柱面数要比新驱动器的柱面数要少才可),就以此入口的类型作为新驱动器的类型。如果同样柱面数的入口有几个,就以最接近的写预先补

偿柱面和定位区作为选定的驱动器类型。

表1 驱动器类型参数表

类型	柱面数	磁头数	写预先补偿	定位区
1	306	4	128	305
2	615	4	300	615
3	615	6	300	615
4	940	8	512	940
5	940	6	512	940
6	615	4	No	615
7	462	8	256	511
8	733	5	No	733
9	900	15	No	901
10	820	3	No	820
11	855	5	No	855
12	855	7	No	855
13	306	8	128	319
14	733	7	No	733

例如:要安装一个Micropolis 1024 型驱动器,它有6个磁头和1024个柱面,没有写预先补偿。根据表1各入口数据,最接近匹配的驱动器类型是5。因此,在回答高级诊断程序的提问时,通过设定这个驱动器的类型为5,用户就能使用940个柱面。注意:941到1024之间的柱面将无法使用,这将损失了约4M字节容量!并且,虽然指定为类型5,也不会执行写预先补偿。

要克服由于类型不配而引起的损失,我们可以设想修改类型5的各项数据,即修改类型5的入口,使它与用户要安装的驱动器参数一致,这样,就可以完全利用新驱动器了。那么问题已归结到,驱动器类型参数表

放在AT机的什么位置呢？AT机是如何找到这份参数表和决定驱动器类型的呢？

二、AT机如何决定驱动器类型参数

所有的AT机和AT兼容机都包含有一个RAM芯片，它靠内部电池供电，用于当计算机断电时保留数据。AT机的高级诊断磁盘可以操纵该RAM芯片。进入RAM中的数据包括：日期，时间、存贮容量、软盘驱动器数，硬盘驱动器数和硬盘驱动器类型。当AT机启动时，它要去读该RAM中的数据以决定驱动器数和驱动器类型。根据读来的这些数据去查对驱动器类型参数表以确定驱动器参数。驱动器类型参数表放在AT机的ROM中，包含有14个入口，每个类型占一个入口（如表1所示）。1型驱动的参数就是第一个入口，如此类推。

后来，有很多AT机把这张参数表扩展到23个入口，甚至有的是47个入口。遗憾的是这些类型和参数都没有标准化，以致今天所生产的许多驱动器仍然不能与参数表中的数据完全匹配。本文所给出的方法和程序可以让用户任意指定驱动器参数以和要安装的驱动器参数相吻合。所指定的这些参数被置入一个新的驱动器参数表并把它放在RAM中，然后AT的驱动器类型指针就改为指向这个新的类型参数表。

三、驱动器类型参数的存放和访问

接通电源后，AT就创建一个指针指向ROM中的驱动器参数表，这些指针是置入AT的矢量中断表中的。矢量中断表占用RAM底部的1024个字节(0到3FF)。这个表中的每个入口占4个字节，标有数字从INT0到INTFFh。AT把指向第一个驱动器的指针放在

INT41h（占104h到107h），把指向第二个驱动器的指针放在INT46h（占118h到121h）。这些指针作为两个字来存贮的，一个是段（Segment），一个是偏移量（offset），段占后二个字节。

使用Debug程序，我们可以测试出指针。采用命令是：

```
-DO:104 L 4
```

Debug给出的响应是：

```
0000:0100 11E4 00 F0
```

这说明在位置104h的矢量是指向地址：F000:E411，这是一个在驱动器参数表中的16字节入口。要看看这个入口的内容，再键入命令：

```
-D F000:E411 L 10
```

Debug的响应是

```
F000:E410 67 02 04 00 00 00 2C
```

```
01 00 00 00 00 00 67 02 11
```

```
F0000:E420 00
```

驱动器参数表的每个入口内容占16个字节，每个入口的内容格式由表2详细的注明（见表2）。

表2 驱动器类型入口格式

Byte	Size	含义	备注
0	字	柱面数	
2	Byte	磁头数	
3	字	没用	
5	字	写预先补偿	
7	Byte	没用	
8	Byte	控制字节	磁头数在8以内为00h 磁头数超过8为08h
9	Byte	没用	
10	字	没用	
12	字	定位区	
14	Byte	每个磁道分段数	通常是17
15	Byte	没用	

刚才这个例子是以AT的驱动器C为例的，是驱动器类型2。参数表的入口是依次紧

挨的,紧跟着刚才显示的后面16个字节就是驱动器类型3的参数,位于刚才显示之前的16字节就是驱动器类型1的参数。

要看到整个驱动器参数表,采用命令如下:

```
-D F000:E401 L E0
```

Debug的响应如清单1所示(见清单1)。

当然,如果你考虑你的机器的驱动器参数表中的类型数超过14,你可采用刚才类似的方法看到更多的信息。如果对清单1中的数据执行16进制到10进制转换程序,立即就会发现,清单1中的数据与表1的数据是完全对应的,每个入口内容的格式也与表2描述的是一样的。

Listing1: The entire drive Parameter table as displayed by a DEBUG dump.

```
F000: E400      32 01 04 00 00 80 00-00 00 00 00 00 31 01 11
F000: E410 00 67 02 04 00 00 2C 01-00 00 00 00 00 67 02 11
F000: E420 00 67 02 06 00 00 2C 01-00 00 00 00 00 67 02 11
F000: E430 00 AC 03 08 00 00 00 02-00 00 00 00 00 AC 03 11
F000: E440 00 AC 03 06 00 00 00 02-00 00 00 00 00 AC 03 11
F000: E450 00 67 02 04 00 00 FF FF-00 00 00 00 00 67 02 11
F000: E460 00 CE 01 08 00 00 00 01-00 00 00 00 00 FF 01 11
F000: E470 00 DD 02 05 00 00 FF FF-00 00 00 00 00 DD 02 11
F000: E480 00 84 03 0F 00 00 FF FF-00 08 00 00 00 85 03 11
F000: E490 00 34 03 03 00 00 FF FF-00 00 00 00 00 34 03 11
F000: E4A0 00 57 03 05 00 00 FF FF-00 00 00 00 00 57 03 11
F000: E4B0 00 57 03 07 00 00 FF FF-00 00 00 00 00 57 03 11
F000: E4C0 00 32 01 08 00 00 80 00-00 00 00 00 00 3F 01 11
F000: E4D0 00 DD 02 07 00 00 FF FF-00 00 00 00 00 DD 02 11
F000: E4E0 00
```

那么,AT起动以后,是如何访问到驱动器参数表的呢?AT引导以后,系统首先读Config.SYS文件并执行之。Config.SYS文件中,系统装有特定的设备驱动程序。被指定的设备驱动程序直接装在DOS所占区之上的存贮区,以后各设备驱动程序依次顺序装入前一个设备驱动程序之上。以后当要执行一个程序时,DOS就把相应代码也装入,位置在最后一个设备驱动程序之上。一旦完成装入工作,由设备驱动程序所占用的存贮区就由操作系统保护起来,直到关闭电源为止,装到该存贮区的数据也被永远保持。由

于在一个设备驱动程序里装有规格化的驱动器类型参数表,在机器引导处理过程期间,这个表就被装入受保护的RAM区中。只要电源是接通的,就可供访问。

本文所给出的设备驱动程序,名为ATDRIVE.SYS,它就是要迫使AT接受驻留在RAM中的规格化的驱动器参数。

四、设备驱动程序

清单2是用汇编语言写的设备驱动程序ATDRIVE.SYS(见清单2),并加有详细注释。

Listing2

```
;
; Device driver to specify nonstandard hard
; disk drives for the AT
;
```

```

0000          code    segment public 'ATCODE'
0000          atdrive proc    far
                        assume cs:code, ds:code, es:code
0000          org     0
0000 FF FF FF FF header    dd    -1          ; link to next device=end of list
0004 8000                dw    8000h         ; device attribute word
0006 0036 R              dw    strat         ; strategy entry point
0008 0041 R              dw    intr         ; interrupt entry point
000A 41 54 44 53 4B 54 41 db    'ATDSKTAB' ; device name
42

;
; double word pointer to request header
;
0012 ? ? ? ? ? ? ? ? rh_ptr    dd    ?          ; pointer to request header
;
; nonstandard hard disk drive type definition table
;
0016          disk_drive_1:
0016 0280                cyls_1    dw    640          ; number of cylinders
0018 08                  heads_1   db    8           ; number of heads
0019 0000                dw    0           ; not used
001B FFFF                wpc_1     dw    0FFFFh        ; write precompensation
001D 00                  db    0           ; not used
001E 00                  ctl_1     db    0           ; control byte
001F 0000                dw    0           ; not used
0021 00                  db    0           ; not used
0022 0280                land_1    dw    640          ; landing zone
0024 11                  sect_1    db    17          ; sectors per track
0025 00                  db    0           ; not defined
0026          disk_drive_2:
0026 0132                cyls_2    dw    306          ; number of cylinders
0028 04                  heads_2   db    4           ; number of heads
0029 0000                dw    0           ; not used
002B 0080                wpc_2     dw    128          ; write precompensation
002D 00                  db    0           ; not used
002E 00                  ctl_2     db    0           ; control byte
002F 0000                dw    0           ; not used
0031 00                  db    0           ; not used
0032 0131                land_2    dw    305          ; landing zone
0034 11                  sect_2    db    17          ; sectors per track
0035 00                  db    0           ; not defined
;
; device driver strategy routine
;
0036          strat      proc far          ; save address of request header
0036 2E, 89 1E 0012 R      mov word ptr cs: [rh_ptr], bx
003B 2E, 8C 06 0014 R      mov word ptr cs: [rh_ptr+2], es

```

```

0040 CB          ret
                strat      endp
;
; device driver interrupt routine
;
0041          intr      proc      far
0041 50          push     ax          ; save general registers
0042 53          push     bx
0043 52          push     dx
0044 1E          push     ds
0045 06          push     es
0046 57          push     di
0047 0E          push     cs          ; make local data addressable
0048 1F          pop      ds
0049 C4 3E 0012 R les     di, [rh_ptr] ; ES:DI now=request header
004D 26, 8A 5D 02 mov     bl, es: [di+2] ; make bx=command code
0051 32 FF          xor     bh, bh
0053 83 FB 00       cmp     bx, 0          ; initialization command?
0056 74 05          je      init_st       ; yes-initialize drive pointers
0058 B8 8003       mov     ax, 8003h      ; no-error
005B 74 07          je      init_ed
005D          init_st
005D E8 0072 R      call    ini
0060 C4 3E 0012 R les     di, [rh_ptr] ; restore ES:DI=request header
0064          init_ed,
0064 0D 0100       or      ax, 0100h      ; set done flag in status word
0067 26, 89 45 03 mov     es: [di+3], ax
006B 5F          pop      di
006C 07          pop      es
006D 1F          pop      ds
006E 5A          pop      dx
006F 5B          pop      bx
0070 58          pop      ax
0071 CB          ret          ; return to DOS
;
; initialize device driver
;
0072          init      proc      near
0072 06          push     es          ; save address of request header
0073 57          push     di
0074 33 C0       xor     ax, ax          ; set extra segment to bottom of
                                     memory
0076 8E C0       mov     cs, ax
0078 BF 0041     mov     di, 41h        ; put INT number into di register
007B D1 E7       shl     di, 1          ; multiply by 4
007D D1 E7       shl     di, 1
                                     ; set interrupt vector for first
                                     drive

```

```

007F B8 0016 R      mov     ax, offset disk_drive_1
0082 FA             cli             ; disable interrupts
0083 26: 89 05       mov     es: [di], ax
0086 26: 8C 4D 02    mov     es: [di] +2, cs
008A FB             sti             ; re-enable interrupts
008B BF 0046        mov     di, 46h   ; set interrupt vector for second drive
008E D1 E7          shl     di, 1
0090 D1 E7          shl     di, 1
0092 B8 0026 R      mov     ax, offset disk_drive_2
0095 FA             cli
0096 26: 89 05       mov     es: [di], ax
0099 26: 8C 4D 02    mov     es: [di] +2, cs
009D FB             sti
009E B4 09          mov     ah, 9     ; print init message
00A0 BA 00B4 R      mov     dx, offset ident
00A3 CD 21          int     21h
00A5 5F             pop     di       ; restore address of request header
00A6 07             pop     es
                                ; set first usable memory address
00A7 26: C7 45 0E 0072 R  mov     word ptr es: [di+14], offset init
00AD 26: 8C 4D 10      mov     word ptr es: [di+16], cs
00B1 33 C0          xor     ax, ax   ; set return status
00B3 C3             ret
                                init
                                endp
=000D              cr     equ       0Dh   ; ASCII carriage return
=000A              lf     equ       0Ah   ; ASCII line feed
=0024              eom    equ       '$'   ; end-of-message signal
00B4 0D 0A 0A      ident db         cr, lf, lf
00B7 41 54 20 4E 6F 6E 2D db       'AT Nonstandard Disk Drive Table Installed'
53 74 61 6E 64 61 72
64 20 44 69 73 6B 20
44 72 69 76 65 20 54
61 62 6C 65 20 49 6E
73 74 61 6C 6C 65 64
00E1 0D 0A 0A 24   db         cr, lf, lf, eom
                                intr     endp
                                atdrive  endp
00E5              code    ends
                                end

```

程序的开始部份是为第一个和第二个驱动器安排的驱动器类型定义表。如果系统只有一个驱动器，可将驱动器2的表删除。在该表之后是处理策略程序。

接下来是中断程序。这个程序化处理命令代码0（即初始化功能），对任何其它代码

则返回一个错误。通常，当驱动程序被明确时，首先接收一个命令代码，它对应执行那一个命令功能。这里给的设备驱动程序只处理命令代码0。一旦一个驱动程序被装入并且在下一个设备驱动程序被装入之前，DOS就自动地将初始化命令代码送给每一个设备驱

动程序,在该设备驱动程序初始化以后,它就再不会被调用了。事实上,在初始化之后它还存在的原因,仅仅是为了保持RAM中的规格化驱动器类型参数表。

初始化程序计算中断41h的地址,然后把指向驱动器表1的指针放入这个地址(在指针没有改变之前,不能中断,在这之后可以中断,这就防止了中断程序去读一个被改变了的指针)。对于驱动器2的中断46h,上述过程是一样的,如果仅有一个驱动器,设置驱动器2的代码指针就不必再包括了。工作完成后,驱动程序显示出“AT Nonstandard Disk Drive Table Installed”的信息。最后,设备驱动程序的末地址被设定在初始化程序的开始。指向中断41h和46h的指针已改为指向规格化驱动器表,DOS不可能对这份驻留在RAM中的表进行写操作,因DOS的启动地址设定在最后一个设备驱动程序装入后的位置。现在,需要利用驱动器参数的任何程序(例FORMAT和FDISK)就会找到驻留在RAM中的规格化参数表,而不是原来的驻留在ROM中的表。

五、重新引导和安装

汇编程序的输出文件是可执行的格式,需要转换为设备驱动程序所要求的命令格式。因此,在汇编之后,可输入下面命令:

```
EXECZBIN ATDRIVE.EXE AT  
DRIVE.SYS
```

这样把ATDRIVE.EXE转换为ATDRIVE.SYS。下一步工作是把ATDRIVE.SYS拷贝到引导目录,然后在Config.SYS文件中写入一句:

```
DEVICE=ATDRIVE.SYS
```

现在,当重新引导计算机时,新的设备驱动程序就起作用了。

用户还应当创建一个可引导的具有新的ATDRIVE.SYS和Config.SYS文件的系统

磁盘,当然上面也要有FORMAT和FDISK文件。到此,就可确信,新的磁盘驱动程序在每次引导系统时已被装入。

在完全能利用新的硬盘驱动器前,仍要有三步工作要做:低级格式化、磁盘分区和高级格式化。

六、格式化

软盘驱动器的低级格式化和高级格式化利用一个Format命令就完成了,硬盘不是。硬盘要分二步走:先低级格式化再高级格式化。

低级格式化在介质上写上分段标志(Sector markers)和在每个段上写上虚拟数据。执行低级格式化需要专门的程序,并且该程序并不在DOS盘上,而是在AT高级诊断盘上。在进行格式化和分段以前,要确保机器已经利用新的驱动器类型参数表引导。工作步骤如下:利用新的驱动器表从系统盘上引导计算机,把AT高级诊断盘放入驱动器A,使驱动器A为你的当前(default)驱动器,随即键入COMMAND.COM。

现在选择菜单上的选项0,即通过系统检查程序。从子菜单上再选择选项0,运行测试程序,选择选项17来测试驱动器。而后再选择选项7,C来进入格式化菜单,最后选择选项2,C来执行一个非条件性格式化。格式化从最高数字的柱面开始,你应该注意到,所显示的开始柱面的数字与你的规格化参数表中的值是相等的。

在磁盘上要创建一个分区表,首先要从软盘上重新引导系统,该软盘自然是含有新的驱动器表的。然后运行FDISK来将驱动器分段。运行了FDISK后,系统从软盘驱动器上重新引导。

下面应进行高级格式化。高级格式化建立引导磁道,创建一个具有文件分配表和根目录的文件系统,以及测试磁盘上的每个磁

一个专为清洗盘洗磁头设计的程序

广东商学院

张 胜

软盘驱动器的日常维护中，经常是用清洗盘来清洗磁头。清洗盘用过几次，就会发现盘上最外圈变脏而盘面的其它地方似乎没有用过。事实也正如此。平时我们清洗，通常是打一命令或任给一个文件，使驱动器转起来。这里所给的命令必定也与文件操作有关。涉及到文件，计算机总是先去磁盘上的文件目录表中查看指定的文件（或者是盘上所有文件），而文件目录表是在零道上（最外道）的几个扇区内。这种方法只能使磁头停在磁道的上方。此时，所插的盘是清洗盘，自然找不到什么文件。这样驱动器转上几秒钟，返回一出错信息就停转了。上述所见，通常的方法只能用到清洗盘上靠外面的那一圈（零道附近的一圈），清洗盘的使用率很低。其次，一般清洗时间要半分钟一分钟，这需要反复打那命令（文件），使用很不方便。下面介绍一个程序，它是专为用清洗盘洗磁头设计的。使用时只要给出驱动器名，清洗便从0道开始，转上几秒钟，再进到1道，这样，边转边进道，一直进到最内道（39道），然后又回到0道。如此反复不停。若要结束清洗，只要在键盘上任击一键。用此程序来清洗磁头有这样的优点：①整张清洗盘面都可利用（0~39道）。②磁头除了通常在磁道的园周方向有运动外，半径方向也产生进道运动，增强了清洗的效果。③所进行的每一步都有信息提示，使用非常

方便。④不会因程序原因而损坏误插进去的有信息盘。

此程序的汇编文本附在本文的最后。将其用MASM（或ASM）汇编，再经LINK连接生成EXE文件。存放在硬盘上，供随时调用。下面以其名为DISKCLEA.EXE为例，说明其使用过程。

C>DISKCLEA ; 键入文件名

PLEASE ENTER THE DRIVER (A or B)

; 屏幕提示

A ; 键入A, A

; 驱动器回转

HIT ANY KEY THEN STOP ; 屏幕提示

┐ ; 任击一键

CONTINUE TO CLEAN? (Y/N)

; 屏幕提示

N ; 键入N

C> ; 返回系统状态

汇编文本

```
STACK SEGMENT PARA STACK
        'STACK'
        DB 64 DUP(0)
STACK ENDS
DATA SEGMENT PARA PUBLIC
        'DATA'
        DRIVER DB 0
        K_BUF DB 80 DUP(0)
        K_BS DW 0
        BUFFER DB 512 DUP(0)
        M1 DB CR, LF, 'PLEASE
                ENTER THE DRIVER
                (A or B)', CR, LF, '$'
```

道。用户必须为每个驱动器分段做一个高级格式化。要确保将ATDRIVE.SYS和Config.SYS文件拷贝到现行分段（驱动器C）的根目录。

通过本文所介绍的方法，相信你能在你的AT机上使用并且是100%的使用任何一种硬盘驱动器了。

M2	DB	CR, LF, 'HIT ANY KEY, THEN STOP',	J4,	JMP	J1
		CR, LF, '\$'		MOV	DL, LF
M3	DB	CR, LF, 'CONTINUE TO CLEAN? (Y/N)',		CALL	K_DIS
		CR, LF, '\$'		MOV	CL, 0
M4	DB	CR, LF, 'ENTER ERR -OR!!!', CR, LF,	J5,	MOV	BX, K_BS
		'\$'		DEC	BX
				INC	BX
				MOV	AL, [BX]
				CMP	AL, CR
DATA	ENDS			JZ	J7
CODE	SEGMENT	PARA PUBLIC		CMP	AL, ' '
		'CODE'		JZ	J5
START	PROC	FAR		CMP	CL, 0
	PUSH	DS		JZ	JD
	MOV	AX, 0		JMP	ERROR
	PUSH	AX	JD,	SUB	AL, 'A'
	MOV	AX, DATA		JZ	J6
	MOV	DS, AX		CMP	AL, 1
	MOV	ES, AX		JZ	J6
	ASSUME	CS, CODE		SUB	AL, 32
	ASSUME	DS, DATA		JZ	J6
	ASSUME	ES, DATA		CMP	AL, 1
CR	EQU	0DH		JZ	J6
LF	EQU	0AH		JMP	ERROR
BS	EQU	08H	J6,	MOV	DRIVER, AL
J14,	MOV	DX, OFFSET M1		MOV	CL, 1
	CALL	S_DIS		JMP	J5
	MOV	BX, OFFSET K_BUF	J7,	CMP	CL, 1
	MOV	K_BS, BX		JZ	J8
J1,	CALL	K_IN		JMP	J14
	MOV	DL, AL	J8,	MOV	DX, OFFSET M2
	CALL	K_DIS		CALL	S_DIS
	CMP	AL, BS		MOV	DL, DRIVER
	JZ	J2		MOV	DH, 0
	MOV	[BX], AL		MOV	CL, 1
	INC	BX	J9,	MOV	CH, 0
	CMP	AL, CR	J10,	MOV	BX, OFFSET BUFFER
	JZ	J4		MOV	AL, 1
	JMP	J1		MOV	AH, 2
J2,	CMP	BX, K_BS		INT	13H
	JNZ	J3		MOV	AH, 1
	JMP	J1		INT	16H
J3,	DEC	BX		JNZ	J12
	MOV	DL, ' '		PUSH	CX
	CALL	K_DIS		XOR	CX, CX
	MOV	DL, BS	J11,	LOOP	J11
	CALL	K_DIS	JJ,	LOOP	JJ

中英文电子打字机与微型机联机通信

空军电子技术研究所

冀忠方

摘要 本文着重介绍KL-8701中英文电子打字机与长城0520CH微机联网通信中的硬件连接, 以及中英文电子打字机与0520CH联机时汉字区位编码转换。

一、前言

中英文电子打字机, 目前在国内流行的有: 中信公司的文豪mini5H, 科理的KL-8701, 四通MS-2401、MS-2402、MS-2403, 天津夏普的WL-1000C, 深圳的智能9250A, 常州的DAS-2416等几十种。但这些中英文电子打字机, 有的内存容量小, 有的没有进网联网软件, 有的没有通信协议和规程, 有的虽然能与计算机之间传输数据, 但通信规程十分简单。所以它们普遍存在不能与各种计算机组网联网, 下面介绍科理KL-8701中英文电子打字机和长城0520CH微型计算机联机通信的具体做法。

二、硬件接口和连线及

通讯协议

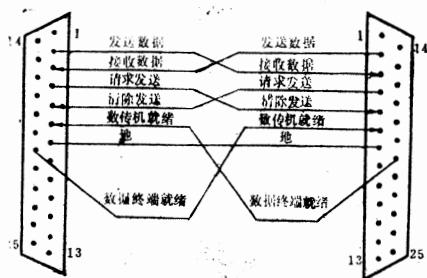
计算机与计算机、计算机与其他设备实现互相联网和通信, 首先要解决硬件接口和连线问题。0520CH有两个串行口(RS-232), 一个并行口(接打印机)。KL-8701对外有多种接口选件, 我们选购通信接口板(RS-232)。KL-8701与0520CH之间的连线, 见图1所示。

三、信息转换

KL-8701有国家标准的一二级汉字,

```
POP      CX
CMP      CH, 39
JZ       J9
INC      CH
JMP      J10
J12:     CALL K_IN
J13:     MOV  DX, OFFSET M3
         CALL S_DIS
         CALL K_IN
         MOV  DL, AL
         CALL K_DIS
         CMP  AL, 'Y'
         JZ   J15
         CMP  AL, 'Y'
         JZ   J15
         RET
J15:     JMP  J14
ERROR:   MOV  DX, OFFSET M4
         CALL S_DIS
```

```
JMP      J13
K_IN     PROC NEAR
         MOV  AH, 8
         INT  21H
         RET
K_IN     ENDP
K_DIS    PROC NEAR
         MOV  AH, 2
         INT  21H
         RET
K_DIS    ENDP
S_DIS    PROC NEAR
         MOV  AH, 9
         INT  21H
         RET
S_DIS    ENDP
START    ENDP
CODE     ENDS
END      START
```



KL-8701 0520CH
RS-232C插头 RS-232C插头

图1 KL-8701与0520CH通信电缆接线图

但它的一二级汉字所对应的汉字编码与0520CH的汉字编码有区别。所以要想使中英文电子打字机与计算机联机通信,还要解决数据信息的相互转换,经过一段时间的摸索和分析,我们掌握了KL-8701与0520CH汉字编码的规律,并找出了其互相转换的算法。

1. KL-8701的汉字编码转换成0520CH的区位码算法。

$$(1) I_1 = \text{INT}((S+1)/2)$$

$$(2) I_2 = \text{INT}(S/2)$$

$$(3) QWM_H = 128 \cdot I_1 + 64 \cdot \text{INT}(S/63)$$

$$(4) QWM_L = 63 + 95 \cdot (I_2 - I_1 + 1) + R + (I_1 - I_2) \cdot \text{INT}(R/64)$$

2. 0520CH的区位码转换成KL-8701汉字编码的算法。

$$(1) I_3 = \text{INT}(QWM_L/159)$$

$$(2) I_4 = \text{INT}(QWM_L/128)$$

$$(3) S = 2 \cdot (QWM_H - 128 - 64$$

$$\cdot \text{INT}(QWM_H - 128) / 70) - 1 + I_3$$

$$(4) R = QWM_L - 63 - 95 \cdot I_3 - I_4 \cdot (1 - I_3)$$

KL-8701的一个汉字编码为二个字节,其中算法中的S为高八位,R为低八位,INT为取整数,*是乘法,/是除法,QWM_H是区位码的高八位,QWM_L是区位码的低八位。

四、注意事项

1. KL-8701的数据信息转换成0520CH数据时,应注意以下几点。

(1) KL-8701的有些ASCII码字符同0520CH相同,例如十个阿拉伯数和二十六个英文字符等。

(2) 有些字符象·×÷±∞≠\$是KL-8701自己定义的,它们分别为A₀、A₁、A₂、A₃、A₄、A₅、A₆。

(3) KL-8701还定义了一些功能符,其范围为十六进制的F040至F07E。

2. 0520CH的数据信息转换成KL-8701数据时,应注意以下几点。

(1) 把在各种方式下输入的汉字转换成区位码,并找出与KL-8701相同的ASCII码字符。

(2) 把·×÷±∞≠\$七个字符编码转换成KL-8701的编码A₀—A₆。

(3) 把0520CH中的回车、换行等编码转换成KL-8701中的相应功能符。

按上述方法联机通信,使用中取得较满意的效果。

Microsoft的当前技术动态和未来发展战略

天津市电子计算机研究所

张一鸣

不久前,应机电部邀请,美国Microsoft公司副总裁 Mr Joachim Kempin 一行数人对我国进行短暂访问。探讨有关与中国软件业合作的可能性和中国国内软件保护与标准化等问题。并在清华大学举行了技术研讨会,介绍有关Microsoft公司当前的技术动态和未来发展战略问题。

Microsoft公司代表、远东项目经理Mr Ben Hsu介绍说,随着VLSI设计和工艺技术的飞速发展,提供了性能上越来越先进而价格却愈来愈低廉的半导体处理器,从而促进了微机的迅速发展。而这些用户绝大多数都是在MS-DOS基础上使用微机的。截止90年10月,正式登记注册的MS-DOS用户有5千多万个,而估计非法拷贝的用户至少是前者的一倍。MS-DOS之所以受到这样普遍欢迎,除了它对各种硬件的广泛支持外,还在于它为软件建立单一的二进制标准格式,这是迄今任何一个操作系统尚没能做到的。

两年前Microsoft在推出OS/2时,曾预计OS/2将在几年内逐步取代MS-DOS,但后来发现这是不现实的,这是由于难以计数的用户投入巨额资金在MS-DOS上开发的大量成果是不能轻易丢掉的。因此公司决定将继续大力支持开发MS-DOS。现在有40多名工程师在改进发展MS-DOS,这比过去最多时只有9个人要多数倍的力量。做结果为,新推出的MS-DOS 4.0版和4.01版有了许多新特色,它表示在:

- 1.对命令有所增加。
- 2.对扩展内存的有限度使用。
- 3.冲破了MS-DOS V3.X对硬盘文件系统空间不能超过32MB的限制,允许单个文件系统分区达到2GB。
- 4.改进了文件管理系统,使其与OS/2

V1.1的文件管理系统兼容。

- 5.增加了面对文本的类似Window的DOS用户Shell。使用户抛开了传统的输入字符串命令的DOS命令方式而进入选菜单发命令方式。

Mr Ben Hsu在关于MS-DOS下一步发展时说,MS-DOS要更好地支持图形功能,使用扩展内存,并通过OS/2的LAN Manager进入OS/2世界。要支持更大容量的硬盘和更高分辨率的显示终端,并将支持3.5"的CD-ROM光盘,更快、更容易安装和使用。特别是通过Windows软件,一改MS-DOS面向字符串处理的传统形象而进入图形世界。今年6月,公司公布了新一代产品Windows V3.0,短短的4个多月,就已经通过零售卖出了一百多万份,还有50个OEM厂家通过许可证销售。并已经在其上开发了1000多个应用软件。Windows提供的软件开发工具SDK使得软件开发工作更方便。它有2万多个功能。Windows的运行最小需640KB内存,最大支持16MB,还有类似多任务的功能。在Windows下开发的各类应用软件具有命令、用法的一致性。而且与设备无关,可以充分发挥应用集成软件的特点。各大软件厂商都在加紧把自己的产品向Windows下移植,如Lotus, Wordperfect, Aston-Tate, Borland, Software Publishing等。据了解,明年将有1500多个支持Windows的新的应用软件开发出来。明年6、7月份,Microsoft将会推出把MS-DOS+Windows装在一起的ROM芯片提供用户,以便为用户省出更多的内存空间。另一项重要新产品是在Windows下的手写体英文识别设备。

关于MS-DOS和OS/2的关系, Mr Ben

国产PC机如何联入3+以太网

济南机车工厂 林鑫

摘要 3+网是一种较为成熟的局域网，它可以面对多种型号的微型计算机进行联网，但是，由于目前国产机兼容性的限制，在直接上网时均存在一定的问题，本文从在3+网上微机联网时的基本原理入手，利用3+网提供的丰富的软件支持，从而顺利地解决了国产IBM兼容机联入3+网的各种难题。

3+以太网是3com公司于1986年2月份推出的第三代微机网络产品，现已有1.00、1.10、1.20、1.31等版本，是局域网中较为成熟的网络之一，在我国不少部门及单位都采用了3+以太网进行了微机联网。

3+以太网特别适合于IBM PC/XT/AT及其兼容机连网，就我国目前来说，联网上机的均采用原装PC/XT/AT等显然成本太高，而主要应采用我国自己生产制造的IBM系列兼容机进行联网，如长城A，长城CH，浪潮机，紫金AT，长城286，曙光机等。

下面就谈一谈国产PC机的联网原理和方法：

一、上网原理

用户站上网之前，必须用工作站启动盘重新启动工作站，下面首先让我们来看一下工作站启动盘的主要文件目录：

COMMAND.COM	NB.COM
ETH.SYS	PRO.SYS
CONFIG.SYS	BUF.SYS
IDP.SYS	SPP.SYS
LGL.SYS	RUNMWDS.SDM
ANSI.SYS	AUTOEXEC.BAT
EDLIN.COM	GWINT16.COM
RIP.SYS	BUF.SYS

在启动盘主文件目录中，有一个配置文件CONFIG.SYS，是一个DOS每次启动时均需检索执行的文件，它的主要任务是对计算机进行初始化配置，下面是一个工作站CONFIG.SYS文件的实例：

device=Eth503.SYS	；数据链路驱动程序
device=Pro.SYS 8 20 2	；进程调度程序
device=buf.SYS	；缓冲区管理程序
device=idp.SYS	；网间数据报管理进程
device=SPP.SYS	；顺序报文分组管理程序
device=lgl.SYS	；登录管理程序
device=rip.SYS	；路径选择管理程序

Hsu讲到，今后3~5年内，MS-DOS将会占有微机市场的65~85%，OS/2会占15~35%。Windows与MS-DOS将与OS/2在相当长一段时间内并存，但各自占有不同的应用领域：OS/2做为中、高微机的操作系统，充当网络服务器；而MS-DOS和Windows做为普遍微机系统环境。这样，旧的软、硬件资源得以继续保留，而新机器又能提供更多更快的服务。两者通过OS/2中的LAN Manager来联接，达到共享资源和数据的目的。

目前，MS-DOS和OS/2格式不同，无

法沟通信息。下一步要提供一个SMK(Software Migration Kit)以便把在Windows下开发的应用软件转化为OS/2 V1.X格式的。第二步把DOS环境和OS/2 V2.X环境结合起来，构成一个BCL(Binary Compatibility Layer)，以允许在OS/2 V2.X的保护方式下与Windows的应用软件在指令级兼容。总而言之，Windows+MS-DOS与OS/2会兼容发展，Windows将是开发环境，而OS/2将是最终的集成系统。明年，OS/2将主要瞄准网络服务器市场。

```
device=buffers=6
device=files=20
device=lastdrive=e
```

文件中每个device语句行均代表一特定的功能，其中有一个语句行device=eths0s.sys，是PC机上网的关键所在，它的功能是装入网络驱动程序。

对于一个网络驱动程序，必须指定出一系列与工作站计算机相匹配的参数后，才能正常运行，这些参数是：DMA通道，中断级，传输类型，DMA类型，I/O基址等。

装入网络驱动程序的device语句的标准形式是：

```
device=Eth503.sys[/parameter:
value][.../parameter:value]
```

表达式parameter:value是可以根据需要而加上的，没有次序也没有数量的限制，不过，每组参数之间必须有一个空格隔开，并且参数表达式中不允许出现空格及其一些非法字符，控制符等。

下面以Eth503.sys驱动程序为例，列表说明Eth503.sys所带参数及其值的确定方法。(如表1)。

二、如何上网

现在3+以太网，网络连接板有好几种产品，3C501，3C503，3C505B等，不同的网络连接板带有不同的适配器，同时也对应有不同的网络驱动程序。

任何一台PC机，准备联入3+以太网时，

首先必须确定这台机器的D参数，M参数，I参数，T参数和网络连接接板的I/O基址，必须确定数据传输方式（粗电缆或细电缆），第二步就是在一台IBM PC机上，利用一个编辑器对工作站启动盘上的CONFIG.SYS文件进行修改；第三步，利用工作站启动盘启动该工作站即可上网运行网络程序。

以下是长城0520CH的上网过程。

经初步分析及查阅有关资料，长城0520CH的数据传送模式是单字节DMA方式，中断级为3，DMA通道值为1，假定采用粗同轴电缆联结，网络连接板I/O基址为350H，现将这台机器联入3+以太网。

在另一台IBM PC机上将一盘新的工作站启动盘上的CONFIG.sys文件利用Edlin命令进行编辑修改，只须将其中的

```
device=eth503.sys
```

改为：device=eth503.sys/M：

```
1/A:350/T:2
```

将电缆联结好，在此0520CH上用修改后的工作站启动盘启动该计算机就可上网运行网络程序了。

三、小结

在异种机联入3+以太网时，一般都只须修改网络驱动程序所带参数即可上网，对其一些如进程调度程序等所带参数可以采用其缺省值，如发现对网络驱动程序所带参数修改正确无误后仍无法联网的，可以考虑修改其它有关参数。

表1 3C503网络驱动程序参数简表

参 数	功 能
A	设置I/O地址用，缺省值为300H，可选值为300H，310H，350H，250H，280H，2A0H和2E0H，此地址可以通过跳线设置
D	设置DMA通道，缺省值1，可选值为1，2，3
M	取决于用户工作站计算机的数据交换类型，可能值为1，2，3，和4，对于IBM PC，XT及PS/230及其兼容机缺省值为3，PC AT及其兼容机缺省值为4，单字节DMA方式值为1，程序I/O方式值为2，DMA通道方式值为3
T	取决于工作站与网络间的联结介质，缺省值为1，可选值为1，2当使用板上收发器时值为1，使用外接收发器时值为2。