

计算机使用技巧荟萃

——《计算机世界》报
学习版优秀文章选编

《计算机世界》报编辑部 编

计算机世界
CHINA COMPUTERWORLD

LOGIF 龙富

19

学习版

19

学习版

计算机世界
COMPUTERWORLD

计算机世界
CHINA COMPUTERWORLD

19

学习版

LOGIF 龙富

一机

五大功能



电子工业出版社

计算机使用技巧荟萃

——《计算机世界》报学习版优秀文章选编

《计算机世界》报编辑部 编

电子工业出版社

内 容 提 要

本书荟萃了《计算机世界》报 1993、1994 两年度 100 期 520 余例计算机用户用机技巧与实际解决问题经验。其内容涉及操作系统、数据库、应用软件、网络技术与多媒体、计算机语言、加密解密技术、整机/主机、磁盘与软驱器、显示器、键盘、电源与 UPS、打印机与绘图仪、笔记本电脑和其它。本书的出版为了扩展计算机用户和潜在用户的思维,达到触类旁通,使用户在现有环境中,更好地发挥各自的作用。

本书读者对象:非计算机专业人员,初级、中级计算机应用人员。

计 算 机 使 用 技 巧 荟 萃

——《计算机世界》报学习版优秀文章选编

《计算机世界》报编辑部 编

*

电子工业出版社出版

北京市海淀区万寿路 173 信箱(100036)

电子工业出版社发行 各地新华书店经销

北京科技大学印刷厂印刷

*

开本:787×1092 毫米 1/16 印张:53 字数:1712 千字

1995 年 9 月第 1 版 1995 年 9 月北京第 1 次印刷

印数:00001—10100 册 定价:52.00 元

ISBN 7-5053-3218-X/TP·1181

目 录

软件使用与技巧

操作系统

西文 DOS

通过 Int2eh 执行 DOS 命令	(20)
仿真 DOS 环境下 WS 下划线的去除	(25)
第六代磁盘操作系统——DR DOS 6.0	(36)
隐含子目录的又一种方法	(63)
解决 DOS 重入的根本方法	(74)
Setup 中应注意的几个基本问题	(94)
PC SHELL 7.0 中 FNT 字型库的利用	(95)
DOS 5.0 入网技巧	(146)
子目录及文件的移动和改名	(170)
Restore 不能恢复 Backup 备份件之原因及对策	(201)
DOS 命令的一些技巧使用	(221)
DOS 文件日期时间置换方法	(229)
全盘范围内删除指定的文件	(243)
利用 DOS 功能实现分页打印	(251)
DOS 文件安全性删除通用方法	(252)
一种用 EXE 代替 BAT 文件的技术	(258)
在 MS DOS 环境中实现事件驱动输入	(266)
新一代磁盘操作系统 MS DOS 6.0 特色	(270)
DOS 混装致软故障浅谈	(278)
MS DOS 5.0 系统下使用 DR DOS 6.0 的功能	(327)
DOS 5 新增命令介绍	(391)
对 DEL 命令的增强和扩充	(393)
西文 DOS 下的中西文文本阅读程序	(437)
DOS 未公开的 67H 号功能调用与程序打开多个文件	(460)
为 DOS 系统增加时钟显示功能	(468)
无硬盘微机使用技巧	(487)
使用 MS DOS 5.0 的体会	(488)
在西文环境下制作小字库显示汉字编程	(490)
降低硬盘 DOS 版本的方法	(502)
如何使 B 盘变成 A 盘	(550)
DOS 目录属性超级管理方法	(552)
拷贝前的空间测试	(560)

如何用好 MS DOS-6.0 中 CONFIG.SYS 的

新功能	(568)
一个增加 PATH 命令使用灵活性的程序	(570)
硬盘的写保护方法	(572)
对 COPY 命令的增强和扩充	(577)
将 BAT 转变成 EXE 又一法	(608)
MS-DOS 6 的磁盘压缩技术	(612)
增强 DOS 的存储命令功能一法	(622)
灵活运用 DOS 的变量	(630)
浅谈大容量硬盘与 DOS	(656)
使 DISCOPY 命令一次性读写复制软盘	(682)
DOS 中 TYPE 文件的改进	(702)
支持通配符 * 和 ? 的子目录改名程序	(720)
西文环境下制作小字库实用技巧	(831)
介绍一个能运行任意子目录里的 BAT、COM 和 EXE 文件的实用程序	(838)

中文 DOS 操作系统

HGC 卡单显环境下 2.13H 系统存在的问题及解决方法	(8)
2.13 在长城东海系列微机上的应用	(21)
给西山中文系统中图像文件加上用户自己的色板	(26)
使用 2.13H 打印程序的几点经验	(34)
为西山 CCDOS WBX.COM 加装扩展词组功能	(48)
汉字 CCDOS 下如何使用 ORACLE	(87)
汉字库在不同路径下的引导	(109)
BDDOS2.10 的一点改动	(109)
在西文 Windows 3.1 环境下 Super-CCDOS/WPS 使用方法初探	(146)
在 DOS5.0 下运行金山汉字系统	(148)
对于 2.13 打印程序与汉字系统配合使用方法的改进	(159)
如何提高 AutoCAD 图形显示速度	(175)
巧制 XENIX V 软盘登录的方法	(176)
SYMDEB 的简易汉化	(187)
让 M-6403 系统增加双拼双音双字词输入功能	(203)
利用硬字库在西文状态下显示汉字的方法	(205)
在 UC DOS 2.0 上安装长城机首尾码的方法	(252)
为 2.13 汉字系统增加繁体字打印功能	(278)

激活 UC DOS 中的副键盘	(291)	XENIX 系统瘫痪的应急修复方法	(743)
CCDOS 2.13H 光标显示的加强	(307)	XENIX 系统硬盘故障恢复一例	(744)
再谈给西山中文系统图像文件加上用户自己 的调色板	(336)	XENIX 终端用户自我保护一法	(790)
CCDOS2.13H 在 Novell 网络系统上的简单 共享	(350)	XENIX 系统中几种拷贝命令的比较	(795)
FoxBASE+2.10 中文环境下保存和恢复屏幕 的通用方法	(379)	XENIX 系统下超级用户口令被非法修改后如何 进入系统?	(796)
在 2.13 汉字系统上挂上任意输入码	(401)	Windows	
在 MS-DOS6 下安装 213H 汉字系统	(412)	MS Windows 3.0 环境下的汉字显示	(1)
汉字系统下显示 MSP 图像文件	(440)	多窗口重叠时窗口边界的绘制	(13)
改进 2.13H 在 VGA 方式下的显示性能	(499)	利用 Windows 实现任务切换	(261)
WMDOS 5.0 一处错误的补救及字笔汉字模 式的自动设置	(507)	在 Netware 上安装共享 Windows	(284)
2.13 系列显示汉字库的改进	(530)	“中文之星”多种双拼汉字输入法的键盘设置	(334)
2.13H 显示字库装入 XMS	(534)	在 Windows 3.1 中打印机的正确使用方 法	(385)
汉字系统 CCBios 2.13H 下如何使用 ORACLE	(547)	利用 Windows for Workgroups 的网络功 能实现文件与打印机的共享	(541)
UCDOS 下增加内存可用空间的方法	(715)	用 Windows 3.1 改变目录属性	(608)
UCDOS 3.0 的使用经验谈	(806)	Windows 与中文之星使用问答	(619)
多用户操作系统		Windows 启动图片的修改	(680)
也谈 XENIX 与 DOS 之间文件成批拷贝程 序的实现	(66)	在中文 Windows 中增添五笔输入功 能	(751)
UNIX 防止误删文件的简单方法	(93)	Windows 屏幕保护程序设计	(775)
建立 VAX VMS 操作系统的清屏命令	(94)		
如何在 DR-DOS6.0 下使用 2.13H	(137)	数 据 库	
关于 VAX/VMS 操作系统的清屏命令	(171)	dBASE	
谈多用户编程中的冲突处理	(275)	将 dBASE III 数据库结构转换成 informix 数据库 文件模式	(18)
XENIX 操作系统下目录名的简单加密法	(335)	数据库录入中的状态自动转换	(26)
一组 UNIX 系统运行管理工具	(372)	在 DOS 中直接访问数据库记录和结构	(88)
XENIX 中 MS-COBOL 如何生成下拉式菜单	(413)	大众化的源程序生成程序	(135)
XENIX 系统下终端打印的实现	(492)	数据库的通配符查询	(231)
UNIX/XENIX 操作系统下 FoxBASE 应用程序 如何实现终端打印	(506)	一个数据管理系统出错处理程序	(353)
XENIX 环境下的打印控制	(642)	关于数据驻留内存存在通信软件中的应 用	(384)
XENIX 系统下终端打印的实现方法	(678)	打印定向语句 SET PRINTER 的应用	(609)
XENIX 系统引导时常见软故障及其维护	(711)	数据库结构与源程序一体化打印输出方 法的实现	(785)
解决 DOS 与 XENIX 共用硬盘故障一例	(712)	利用库文件头对 .DBF 文件加密	(803)
XENIX 系统下多用户储蓄程序故障排除 数例	(716)	FoxBASE	
如何在 UNIX 环境下的 AUTOCAD 中标注 汉字	(719)	在 FoxBASE 上也能实现随机函数的功能	(107)
如何解决 XENIX 系统常见故障	(724)	FoxBASE+下自定义取汉字区位码和机内 码函数	(108)
VMS 环境中 VAX C 与 VAX FORTRAN 的 混合编程	(730)	FoxBASE+与汇编语言通用接口技术	(131)
XENIX 系统软故障分析几例	(743)	在 FoxBASE 下的英、汉输入转换	(156)
		如何在走纸的情况下使 PROW 置零	(158)
		在 FoxBASE 应用中需要注意的几个问题	(169)
		设计 FoxBASE+的通用项间运算程序	(181)

一种“数值为0字段输出用空白替代”的新方法	(184)	WPS 2.1 软件使用的特殊技巧	(305)
为 FoxBASE+ 增加几个新的内容命令	(207)	WPS 使用技巧三则	(453)
关于 Mfoxplus 2.0 中星期、月份的汉化	(227)	WPS 下 M-1724 打印机字体错位的校正	(467)
FoxBASE+ 受损数据库文件的修复	(240)	使 SPT 支持 9 针打印机	(503)
为 FoxBASE+ 增加一个函数	(250)	修改软金山系统适应有缺陷的 386 主板	(531)
用 FoxBASE 实现表格数据向数据库的传送	(299)	读取 WPS 文件的密码	(535)
美观实用的 FoxBASE+ 的调色板函数	(318)	WPS 版面高度测量仪	(632)
FoxBASE+ 源程序结构检测程序 FOXCHECK	(328)	恢复内存中 WPS 文件	(701)
FoxBASE+ 中 ON 命令的妙用	(426)	WPS 排版系统与高级语言的接口技术及应用	(753)
FoxBASE 屏幕作图功能的实现	(427)	如何解决 WPS 与 CR-3240 打印机的配合问题	(765)
FoxBASE 下三个小程序	(443)		
FoxBASE+ 应用软件中的容错处理	(497)		
巧用 FoxBASE 中的 INKEY 函数	(515)		
FoxBASE 环境下源程序结构的直接检测函数	(555)		
FoxBASE 中少字段、多记录数据的全屏幕输入方法	(562)		
FoxBASE+ 四舍五入函数失真的对策	(607)		
FoxPlus+ 下的图文混合管理的应用	(668)		
为 FoxBASE+ 数据库文件增加安全保护功能	(672)		
FoxBASE 在线帮助及其应注意的问题	(685)		
在 FoxBASE 下对三个锁定键的编程方法	(716)		
在 FoxBASE+ 状态显示放大汉字	(759)		
介绍一组在 FoxBASE 环境下数据库记录位置变换的通用程序	(762)		
在 FoxBASE 下实现屏幕浏览式检索的一个过程文件	(766)		
其它			
ORACLE 的分区及其使用	(72)		
一对多关系数据库的关联和连接	(386)		
SQL * FORM 查询中自设栈表的使用	(411)		
四则混合运算表达式自动识别与通用计算程序	(698)		
汉字查找中的一个问题	(800)		
实现 Informix 数据库表任意项组合查询及模糊查询	(808)		
应用 软件			
WPS			
WPS 造词功能的正确使用方法	(20)		
在 PC 机上用 WPS 编辑文件	(149)		
金山系统下内存的合理使用	(241)		
屏蔽 WPS 的版权信息显示	(244)		
找回 WPS 文件中被忘记的密码	(249)		
WPS 桌面印刷系统中 SPT 图形的放大	(265)		
WPS 文件转换成“华光”文件的实用程序	(300)		
		用 PCTools V7.0 修复微机硬盘掉电故障	(35)
		PC SHELL 7.0 中 FNT 字型库的利用	(95)
		快速磁盘搜索	(109)
		利用 PCTools 恢复软盘引导区	(181)
		利用 PCTools 快速鉴别软盘质量	(350)
		用 PCTools 恢复误删的文件或目录	(371)
		浅谈用 PC 工具如何恢复被 ZAP 命令删除的数据库	(837)
		字处理	
		修改 2.13H 汉字系统中 LQ1600 打印机驱动程序单/双向打印控制命令错误	(9)
		怎样使 WordStar 使用更方便	(9)
		优化的通用文本文件阅读器	(28)
		关于 VGA 图形的读取	(38)
		修改 Super-CCDOS 在 AST 386 SX 机上的显示错误	(43)
		自动转换输入方式的子程序	(63)
		24 点阵汉字的显示	(68)
		为 DEBUG 增加一个输出改向命令	(82)
		一个简化华光排版系统使用的程序	(97)
		用热键选择 CRT 的显示方式	(99)
		如何在 PC 机上读取四通中文打字机软盘上的文件	(107)
		直接写显示 RAM 区实现英文 DOS 下显示汉字	(110)
		关于 EGA 或 VGA 显示卡改变 WS 屏幕颜色的方法	(114)
		WS 的版本升级及显示色彩设置	(168)
		汉字操作系统下的俄文键盘	(176)
		“打印”屏幕至磁盘文件	(178)
		五笔字型编码字典打印程序	(179)
		一种检测设备驱动程序链的方法	(185)
		VGA 图形拷贝程序	(192)

CH、CEGA 卡 VRAM 区汉字显示的读写方法···	(204)	谈“四码”的科学推算和惰性检索·····	(255)
CEGA、CVGA 中文显示卡扩展图形的编程···	(213)	利用虚拟盘应用中文字表编辑软件 CCED···	(351)
键盘操作水平测试程序·····	(216)	微机常见软件故障分析·····	(559)
“科印”排版系统无法正常打印的原因·····	(219)	维护修改程序的技巧·····	(564)
PC 机与四通中文打字机文本的相互转换·····	(232)	多种内存模式下的通用编程技术·····	(593)
鼠标激活内存驻留程序方法·····	(244)	PAL、GAL 的 JEDEC 文件到 ABEL 源文 件的转换·····	(648)
在 M-6403 汉卡下使用 CR3240 打印机制作 表格·····	(251)	微机热启动和 RESET 开关复位启动的软件 实现及其应用·····	(667)
EGA/VGA 中文彩色屏幕窗口的快速存储 与恢复·····	(281)	如何在向量巨型机上高速实现字符运算·····	(682)
自动设置 CapsLock 状态·····	(297)	286 微机参数设置信息丢失的处理·····	(714)
文本编辑自动定时存盘的通用方法·····	(335)	NumLock CapsLock ScrollLock 三键状态的 程序控制法·····	(769)
显示速度随机可变、随机切换分屏和打印、可 预先指定起始行的一个有益尝试·····	(346)	微机速度的软件调整·····	(771)
北大(华光)表格的自动转换·····	(360)	图像处理	
西文图形方式下 16 点阵汉字的快速放大显示 为华光、北大排版系统增加、补充勾边、立体、 半阴半阳字功能·····	(372)	滚屏时颜色不正确的原因及解决方法·····	(30)
如何使 PE2 软件使用更加得心应手·····	(388)	在 DOS 状态下随意扩充屏幕和键盘的有效 方法·····	(42)
对热启动按键的安全处理·····	(401)	VGA 图形方式下的读/写像素·····	(54)
正确设置修改中断后的中断返回标志·····	(417)	CEGA、CVGA 中文显示卡编程·····	(69)
关于计算机音乐软件的设计方法·····	(423)	图形图像数据基于像素的压缩与还原·····	(118)
213 24 点阵汉字显示的补正·····	(445)	有趣的图形光标·····	(127)
在 DOS 提示符下直接调用打印机控制指令的 方法·····	(452)	VGA 图形控制器直接编程技术·····	(138)
通用下拉式多级同步显示菜单设计·····	(466)	VGA 高分辨率屏幕图形的汉字显示及图形 打印技术·····	(142)
浪潮 CVGA 卡文本方式下汉字的显示及放大···	(476)	一种书写动画的制作方法·····	(277)
打印机输出表格参差不齐的解决方案·····	(493)	一个生成矢量汉字的简便方法·····	(285)
将四通文件转换为方正(华光)的小样文件···	(509)	一套自成系统的图形截取、编辑系统·····	(311)
不同显示卡下正确设置光标类型·····	(518)	在 ORACLE 库中调用 C 语言图形函数·····	(316)
文件压缩实用程序·····	(549)	Unix System V 3.2.2 以上版本中图像显示 的方法·····	(319)
对文本文件中软回车、控制符的灵活处理·····	(594)	屏幕图形的再利用·····	(398)
PE I 功能剖析·····	(661)	旋转汉字显示的快速算法及其 C 程序·····	(419)
如何显示高质量的点阵汉字·····	(665)	一个由 C++ 实现的图形窗口程序包·····	(441)
巧妙实现同一驱动器下文件的直接拷贝·····	(687)	BMP 图像文件的格式分析与显示·····	(538)
101 键码与 84 键码相互转换程序·····	(690)	16 色 PCX 图像文件的恢复·····	(556)
WordPerfect 5.1 宏命令的使用技巧·····	(704)	利用位覆盖技术实现图形实体的明暗处理···	(561)
西文状态下快速显示汉字的新方法·····	(722)	PC 图形背景的设置及 16 色 PCX 图形文 件作 2 色图形使用的方法·····	(584)
黑白打印机硬拷贝彩色图像方法·····	(745)	一种提高显示器彩色能力的简单方法——Dither ·····	(717)
一个实用的屏幕拷贝程序·····	(763)	屏幕图形的随机转储、编辑、显示和打印·····	(814)
“绿色”打印程序·····	(787)	VGA/TVGA 图形处理中的平滑滚动技术···	(819)
架起四通机 M240X 与 PC 机之间共享数据的 桥梁·····	(817)	计算机病毒	
其它		文件类病毒的通用免疫程序设计·····	(164)
在有效期外运行规定有效期软件的通用程序···	(835)	挽救被病毒破坏的硬盘的一种方法·····	(322)
计算机自动生成试卷·····	(10)		
一种解决试题库中题图一体的方法·····	(187)		
	(235)		

1759 病毒的分析与清除	(323)	Turbo C 应用程序中时钟自动显示	(257)
变形病毒的发展小议	(391)	C 语言的 FILE 结构剖析	(262)
新型病毒“848”	(451)	用 Turbo C 实现文书文件的自由删除	(330)
修复 DIR-I 病毒导致的磁盘软故障	(507)	用 C 语言快速地给 AUTOCAD 点变量赋值	(345)
利用 CPAV 检测计算机病毒动态	(584)	数字金额转换为大写金额的 C 语言程序	(345)
流行计算机病毒惯用伎俩之综析	(598)	用 C 语言检测打印针的状态	(363)
Client/Server 与 Fourth Shift	(638)	分析不同 C 语言编译函数中“++”运算符 的差异	(366)
二叉树病毒的检测与清除方法	(639)	用 Turbo C 语言实现屏幕图形压缩存储及 打印	(369)
Random-Formatting Virus 的分析报告	(726)	再谈 C 语言中多维数组的动态分配	(376)
对 Random-Formatting Virus 病毒的诊断 与清除	(788)	对 Turbo C 与流相联的缓冲区的几个附注	(453)
1100 病毒的清除	(804)	按属性搜索文件程序	(461)
给软硬盘安装一个安全保护开关	(825)	用 C 语言实现双列目录显示	(531)

网络技术

NetWare V3.11 采用工作站作为网络共享打印服务的方法	(157)	怎样用 Turbo C 写控制系统的程序	(790)
关于“四通”文字处理机与微机联网中几个问题的解决	(203)	PASCAL	
Novell NetWare 使用经验	(246)	用高级语言实现显示器的定时关闭	(364)
如何在 Novell 3.11 网上安装使用绘图机	(380)	TURBO Pascal 生成文本文件的一个缺陷及处理	(385)
NetWare 工作站不能进网的原因分析及解决方法	(382)	为 Turbo Pascal 增添覆盖加载功能	(481)
微机局域网服务器的故障分析与维修	(431)	汇编语言	
CMOS 设置造成 Novell 网安装失败问题的解决	(602)	MASM 5.0 中 LINK 时死机的原因及解决方法	(4)
在 Novell 网中无盘工作站上使用金山 DOS 及 WPS 的方法	(628)	如何测定 CPU 的类型	(40)
多媒体通信产品——会议电视	(677)	用 SYMDEB 替代 DEBUG	(44)
网络打印机的使用方法	(686)	鼠标器模拟键盘程序 MOUSEKEY——让程序支持鼠标	(102)
网络发展的新趋势	(692)	怎样在机器中加扩大容量硬盘	(112)

计算机语言

C 语言	不用 SETUP 软件修改 CMOS 内容的方法 ... (260)
利用 C 语言实现音乐简谱的识别与演奏 (64)	建立一个方便实用的汇编程序库 (295)
用 C 语言编制类似 FoxBASE 中的 INKEY(n) 函数 (71)	检测 TSR 程序是否已驻留的方法——探讨 与评价 (331)
Turbo C 键盘宏定义 (91)	实现 TSR 程序“阶层式”管理的通用程序 设计技巧 (339)
TURBO C++ 2.0 屏幕图形写入磁盘文件及 由磁盘文件写屏幕的方法 (114)	DEBUG (Ver5.0) 的汉化方法 (412)
一种 Turbo C2.0 环境下的汉字处理方法 (137)	在主引导扇区上实现多个操作系统的选择 引导 (462)
汉字操作系统下使用 TURBO C 的小技巧 ... (158)	TSR 程序释放内存资源的两种基本编程方法... (469)
用 C 语言调用 SPT 图形一法 (202)	CMOS 的配置信息及设置方法 (604)
exit () 函数在批处理中的应用 (228)	用 DBPACK 压缩 DEBUG 重定向文件 (609)
C 语言多维动态分配的新方法 (248)	多路中断 INT2FH 的分析与应用 (652)

CLIPPER 5.01 与汇编语言的接口	(703)	AST/286 内存板故障的维修	(15)
FORTAN		Super-286 总线故障的排除	(35)
FORTAN 语言直接读取 FoxBASE 数据库文件	(480)	AST P386 微机常见故障的排除	(111)
用 FORTRAN 形成 SCR 文件时摆脱空格影响的 一种方法	(545)	兼容机主板的检测与故障定位	(115)
在 FORTRAN 语言中利用系统资源	(828)	Sun 工作站的 EEPROM 浅析	(160)
COBOL		AST P386/25 处理板故障的判断与排除	(162)
怎样实现 LEVEL I-COBOL 索引文件到 MS COBOL 索引文件的转换	(16)	IBM AT 286 主机故障维修一例	(163)
PROLOG		充分利用好微机的配置信息减少微机修理中的 “误诊”	(219)
FoxBASE+ 与编译型高级语言的通用接口 ...	(173)	AST-386 特有的口令方式——服务模式	(235)
用 PDC PROLOG 建立知识库和推理机	(675)	浅谈购买微型机的步骤	(422)
CAD		长城微机屏幕信息转储程序	(434)
高级语言程序调用 AutoCAD 图形	(60)	内存文本恢复简法	(444)
OrCAD PCB 打印输出功能的改进	(477)	AST/286 机 CMOS RAM 故障一例	(458)
其它		USER 机的密码设置与清除	(496)
GW-BASIC 使用点滴	(397)	扩展内存和扩充内存的比较及使用	(509)
面向对象的程序设计方法 OOP 及其实现	(528)	高档微机的应急维修一法	(546)
高级语言界面设计思路	(635)	芯片散热不佳显示器屏幕模糊	(697)
		Super PC/XT 故障检修一例	(707)
		兼容微机 Super 386SX/20 板故障维修	(750)
		PA-7100LC: 具有 MPEG 视频解码功能的 PA-RISC 芯片	(801)
		微机内存分布及优化使用	(823)

磁盘与驱动器

加密解密技术

AS/400 小型机上具有安全保密的全屏通用 菜单技术	(84)
内存驻留程序安全性要素分析	(153)
一种常见的软件加密方法	(180)
微机暂离锁的软件实现	(367)
怎样对 .fox 文件进行加密	(464)
常用文件压缩软件命令参数速查	(472)
成批文件的一种简易加密方法	(621)
背包口令——一种准确安全识别用户的方法 ...	(625)
谈 DOS 操作系统下批处理程序的加密	(629)
给你的键盘加软锁	(679)
数据库数值字段的加密压缩	(706)
用 FoxPro 对文件加密	(781)

硬件维修

整机/主板

紫金多用户卡的故障排除	(6)
-------------------	-----

硬盘系统参数的获取	(11)
微机磁盘维护的有效工具	(19)
1.2MB 软盘驱动器故障检查与调整	(22)
IBM PC-XT 机硬盘故障检修一例	(36)
软盘驱动器非损坏性故障的处理	(57)
软磁盘的选购、使用及维护	(83)
软盘驱动器的维护	(117)
硬盘软故障排除一例	(125)
ST225 型温盘驱动器故障排除	(134)
硬盘的结构及启动流程	(150)
SUN SPARC 兼容机软驱故障解析一例	(159)
软盘驱动器非损坏性故障处理一则	(163)
驱动器工作异常的一种解决方法	(218)
增加 DRIVER.SYS 设备命令的几种用法	(220)
软盘驱动器维修二则	(271)
加电自检出现“601”故障的分析及维修	(272)
IBM-PC/XT 硬盘的修复	(273)
软驱特殊故障检修一例	(274)
1.2MB 软驱写保护电路故障的修复	(274)
巧“换”软盘驱动器	(291)
硬盘驱动器维修八例	(292)
微机硬盘故障诊断与维修方法优化	(308)

软盘逻辑损坏的修复一法	(357)	显示器的维修几例	(496)
软驱读错误故障排除一例	(358)	全点阵液晶显示器与 8031 的连接	(516)
零磁道损坏软盘的返新	(358)	微机单显电源部分故障维修	(565)
为什么软盘驱动器的灯会常亮不灭	(400)	彩色显示器故障维修一例	(623)
磁盘逻辑损坏的简便修复	(422)	中环 CT100 终端故障维修三例	(624)
使报废硬盘再生利用一法	(423)	监视器选购指南	(694)
用文本文件备份与恢复硬盘主引导扇区的 方法	(459)	对几种常见显示适配器“代码”的简释	(696)
硬盘引导区系列错误的排除	(486)	电感线圈管脚氧化造成显示器行幅“萎缩”	(696)
硬盘不能启动故障维修几则	(488)	故障多发地——大功率器件	(697)
软驱故障排除一例	(508)	单色、彩色显示器的维修两例	(709)
AST 286 微机硬盘故障维修三例	(547)	显示器故障修复一例	(710)
硬盘驱动器保养之拙见	(548)	一种能自动探测显示卡类型的方法	(832)
巧妙恢复硬盘主引导信息	(575)		
对 DOS 不识别硬盘软故障的修复	(658)		
软驱机械故障维修三例	(713)		
修复软驱故障的简易方法	(713)		
软磁盘驱动器适配器故障排除两例	(714)		
软盘驱动器常见故障原因分析及检修	(739)		
软驱的一类故障的分析及排除	(746)		
也谈“巧‘换’软盘驱动器”	(747)		
修复软盘 0 磁道的“怪招”	(748)		
软盘数据抢救一法	(748)		
软盘驱动器故障维修一例	(748)		
高密软驱磁盘更换信号的检测及处理	(748)		
如何鉴别真、假 JANUS 和 MAXELL 软磁盘	(752)		
对调 A 驱和 B 驱	(753)		
用软盘复制命令修复软盘	(770)		
软盘驱动器读写故障的检修	(771)		
续谈 AST 微机硬盘软故障的排除	(793)		
AST 286 硬盘无法启动故障的修复	(794)		
120M 硬盘分区信息丢失故障的排除	(800)		

显 示 器

测试 CRT 的模式	(5)
AST SVGA 彩色显示器维修实例	(100)
显示器适配器的维修	(171)
自动关闭监视器——适用于任何类型的监视器	(188)
显示器的维修六例	(233)
再谈 EGA/VGA 中文彩色屏幕上任一位置窗口 的瞬间存储与恢复	(288)
改变 VGA 的 16 种显示颜色	(358)
GW-300 显示器无光栅故障的分析与维修	(429)
显示卡故障维修	(433)
CTX-7E1 显示器巧修一例	(457)
Compaq 彩色显示器——故障的修复	(459)

键 盘

用逻辑笔检修键盘一例	(281)
键盘的应急修理	(288)
键盘小故障维修一例	(575)
键速的任意调整	(782)

电源与 UPS

UPS 电源的正确使用方法	(59)
SENTECK UPS-500 电源故障维修一例	(60)
中小型(400W~600W)UPS 常见的两种故障 检修	(217)
主机板蓄电池故障维修一例	(218)
微机电源常见故障及其检修方法	(348)
微机系统供电的可靠性浅谈	(484)
电源特殊故障检修一例	(495)
MAXIPAC SXM 10000 UPS 的正确安装与 维护	(566)
SANTAK UPS-500 故障检修	(567)
现代办公设备电源部件的维修实例	(615)
SENTECK UPS-500 型电源常见故障的分析 与维修	(616)
LQ-1000 稳压管损坏造成死机故障	(618)
山特 500W UPS 维修三例	(662)
后备式 UPS 蓄电池的“复活”一例	(663)
SANTAK——500VA 后备式 UPS 维修经验	(664)
U-1100 5kVA 型 UPS 电源故障排除	(689)
微机开关电源维修几例	(774)
Santak-500 型后备式方波电源故障维修	(812)
Santak-1000 后备式 UPS 不能逆变故障的排除	(813)

打印机及绘图仪

IBM PC/XT 兼容机打印适配器故障检修	(56)
------------------------------	------

MS—Windows 3.0 环境下的汉字显示

Microsoft 公司于 90 年代初推出的 Windows 3.0 以其丰富的窗口函数、友好统一的图形用户等优良性能而风行全球,国内也有不少单位相继应用其汉化版本。如果手头只有西文的 Windows,能否同样地显示汉字呢?笔者通过把汉字字模直接映射到屏幕的方法,解决了此问题。

这种方法是点阵汉字库为基础的,以 16×16 点阵的显示字库为例,每个汉字由 32 个字节表示,其排列如图一所示。这样的点阵结构就好比用一个 16×16 的网络罩在汉字上,相交位为 1,其他位置 0。所有这 32 个字节被称为汉字的字模。另一方面,在 Windows 的图形环境下,屏幕图形均由像素构成,如果能够将字模的点阵信息映射到屏幕上对应的像素,就能显示出相应的汉字。这就是本方法的基本思想。

这里的关键是找出要显示汉字的字模来。众所周知,汉字库是按区位顺序存放的,每区有 94 个汉字或符号。设某个汉字的区号为 m ,位号为 n ,则它的字模在汉字库的起始位置应为:

$$((m-1) \times 94 + n - 1) \times 32$$

下面给出一个示例程序的完整清单,包括 MAKE 文件、资源文件、模块定义文件、头文件和 C 源程序。利用 MAKE 文件生成可执行程序 SHOW_CH.EXE,在 Windows 环境下执行该程序,能按屏显示汉字库的所有符号与汉字。笔者这里选用了 XSDOS 的显示字库 CCLIBJ.DOT。

程序清单如下:

```
E:\Windows>type show_ch.m
all:show_ch.exe
```

```
show_ch.res:show_ch.rc show_ch.h
rc -r show_ch.rc
show_ch.obj:show_ch.c show_ch.h
rc -c -AS -Gsw -Oas -Zpe show_ch.c
```

```
show_ch.exe:show_ch.obj show_ch.def show_ch.res
link /NOD show_ch,,slibcew libw,show_ch.def
rc show_ch.res
```

```
E:\Windows>
```

```
E:\Windows>type show_ch.h
```

```
#define IDM_OPEN 103
#define IDM_PROC 110
#define IDM_ABOUT 100
#define IDM_CL 111
#define IDM_FMS 112
#define IDM_CLOSE 102
```

```
int PASCAL WinMain(HANDLE,HANDLE,LPSTR,int);
```

```
BOOL InitApplication(HANDLE);
BOOL InitInstance(HANDLE,int);
long FAR PASCAL MainWndProc(HWND,unsigned,
WORD,WORD);
```

```
void Show_Chinese(HDC,int,int);
```

```
E:\Windows>
```

```
E:\Windows>type show_ch.def
```

```
NAME Show_ch
```

```
DESCRIPTION 'Show Chinese in Windows'
```

```
EXETYPE Windows
```

```
STUB 'WINSTUB.EXE'
```

```
CODE PRELOAD MOVEABLE DISCARDABLE
```

```
DATA PRELOAD MOVEABLE MULTIPLE
```

```
HEAPSIZE 1024
```

```
STACKSIZE 5120
```

```
EXPORTS
```

```
MainWndProc @1
```

```
E:\Windows>
```

```
E:\Windows>type show_ch.rc
```

```
#include "windows.h"
```

```
#include "show_ch.h"
```

```
FlexibleMenu MENU
```

```
BEGIN
```

```
POPUP "&Help"
```

```
BEGIN
```

```
MENUITEM "&About Flexible...", IDM
```

```
ABOUT
```

```
END
```

```
POPUP "&File"
```

```
BEGIN
```

```
MENUITEM "&Open", IDM_OPEN
```

```
MENUITEM "&FMS", IDM_PROC
```

```
MENUITEM "&Close", IDM_CLOSE
```

```
MENUITEM "&About", IDM_ABOUT
```

```
END
```

```
POPUP "&Option"
```

```
BEGIN
```

```
MENUITEM "&FMS", IDM_FMS
```

```
MENUITEM "&C1", IDM_CL
```

```

END
END

```

```

E:\Windows>

```

```

E:\Windows>type show_ch.c

```

```

/*****

```

```

PROGRAM: Show_Ch.c

```

```

WinMain() — main program

```

```

InitApplication() — initializes window data and registers window

```

```

InitInstance() — saves instance handle and creates main window

```

```

MainWndProc() — processes messages

```

```

About() — processes messages for "About" dialog box

```

```

*****/

```

```

#include "windows.h"

```

```

#include "string.h"

```

```

#include "stdio.h"

```

```

#include "Show_Ch.h"

```

```

HANDLE hInst;

```

```

HPEN hBlueSolidPen;

```

```

HPEN hYellowSolidPen;

```

```

HPEN hGreenSolidPen;

```

```

HPEN hRedSolidPen;

```

```

HPEN holdPen;

```

```

HBRUSH holdBrush;

```

```

HBRUSH hBlueBrush;

```

```

*****/

```

```

FUNCTION: WinMain (HANDLE, HANDLE, LPSTR, int)

```

```

PURPOSE: calls initialization function, processes message loop

```

```

*****/

```

```

int PASCAL WinMain (hInstance, hPrevInstance, lpCmdLine, nCmdShow)

```

```

HANDLE hInstance;

```

```

HANDLE hPrevInstance;

```

```

LPSTR lpCmdLine;

```

```

int nCmdShow;

```

```

{

```

```

MSG msg;

```

```

if (!hPrevInstance)

```

```

{
    if (!InitApplication(hInstance))

```

```

    return (FALSE);

```

```

if (!InitInstance(hInstance, nCmdShow))

```

```

    return (FALSE);

```

```

    • 2 •

```

```

while (GetMessage(&msg, NULL, NULL, NULL)) {

```

```

    TranslateMessage(&msg);

```

```

    DispatchMessage(&msg);

```

```

}

```

```

return (msg.wParam);

```

```

}

```

```

*****/

```

```

FUNCTION: InitApplication (HANDLE)

```

```

PURPOSE: Initializes window data registers window class

```

```

*****/

```

```

BOOL InitApplication (hInstance)

```

```

HANDLE hInstance;

```

```

{

```

```

WNDCLASS wc;

```

```

wc.style = NULL;

```

```

wc.lpfnWndProc = MainWndProc;

```

```

wc.cbClsExtra = 0;

```

```

wc.cbWndExtra = 0;

```

```

wc.hInstance = hInstance;

```

```

wc.hIcon = LoadIcon(NULL, IDC_APPLICATION);

```

```

wc.hCursor = LoadCursor(NULL, IDC_ARROW);

```

```

wc.hbrBackground = GetStockObject(WHITE_BRUSH);

```

```

wc.lpszMenuName = "FlexibleMenu";

```

```

wc.lpszClassName = "FlexibleWClass";

```

```

return (RegisterClass(&wc));

```

```

}

```

```

*****/

```

```

FUNCTION: InitInstance (HANDLE, int)

```

```

PURPOSE: Saves instance and creates main window

```

```

*****/

```

```

BOOL InitInstance (hInstance, nCmdShow)

```

```

HANDLE hInstance;

```

```

int nCmdShow;

```

```

{

```

```

HWND hWnd;

```

```

hInst = hInstance;

```

```

hWnd = CreateWindow (

```

```

    "FlexibleWClass",

```

```

    "Show Chinese in MS—Windows 3.0",

```

```

    WS_OVERLAPPEDWINDOW | WS_VSCROLL | WS_HSCROLL \
    | ES_AUTOHSCROLL | ES_AUTOVSCROLL,

```

```

    0, 0,

```

```

    GetSystemMetrics(SM_CXSCREEN),

```

```

    GetSystemMetrics(SM_CYSCREEN),

```

```

    NULL, NULL, hInstance, NULL);

```

```

if (!hWnd)

```

```

    return (FALSE);
ShowWindow(hWnd,nCmdShow);
UpdateWindow(hWnd);
return (TRUE);
}

/* *****
FUNCTION: MainWndProc (HWND, unsigned, WORD,
LONG)
PURPOSE: Show Chinese in screen
***** */

long FAR PASCAL MainWndProc (hWnd, message,
wParam, lParam)
HWND hWnd;
unsigned message;
WORD wParam;
LONG lParam;
{
HDC hDC;
PAINTSTRUCT ps;
LPSTR FileName;
char bv, * Buffer, * tem;
int i=0,j=0,l,f,n,x,y;
unsigned pixel_value;
long m;

switch (message) {
case WM_CREATE:
hRedSolidPen = CreatePen (0,1,RGB(250,0,0));
hGreenSolidPen = CreatePen (0,1,RGB(0,250,0));
hBlueSolidPen = CreatePen (0,1,RGB(0,0,200));
hBlueBrush = CreateSolidBrush(RGB(0,0,200));
break;

case WM_PAINT:
x=0;y=0;
hDC = BeginPaint(hWnd,&ps);
holdPen = SelectObject(hDC,hBlueSolidPen);
tem = Buffer;
FileName = "c:\cclibj. dot";
holdBrush = SelectObject(hDC,hBlueBrush);
Rectangle(hDC,0,0,640,420);
for (m=0;m<8000;m++) {
f = _lopen(FileName,READ);
if (f!=-1) {
_llseek(f,m*32,0);
_lread(f,Buffer,32);
_lclose(f);
for (i=0;i<16;i++) {
for (j=0;j<2;j++) {

```

```

bv = * (Buffer++);
for (l=0;l<8;l++)
if (((bv>>(7-l)&0x01)!=0)
SetPixel(hDC,x+j*8+l,y+i,RGB(0,
250,0));
}
}
Buffer = tem;
x += 16;
if (x>580) {
x = 0; y += 16;
}
if (y>410) {
y = 0;
Rectangle(hDC,0,0,640,420);
}
}
SelectObject(hDC,holdBrush);
SelectObject(hDC,holdPen);
EndPaint(hWnd,&ps);
break;

case WM_DESTROY:
DeleteObject(hBlueSolidPen);
DeleteObject(hBlueBrush);
PostQuitMessage(0);
break;

default:
return (DefWindowProc (hWnd, message, wParam,
lParam));
}

return(NULL);
}

```

最后补充一点,本文采用的字库是16×16点阵的显示字库,它为横向排列结构。如果选用精度更高的打印字库(常为24×24,32×32,48×48),以上程序必须稍作修改,特别要注意的是,打印字库为纵向排列方式。

字节0	字节1
字节2	字节3
⋮	⋮
字节30	字节31

图1 显示字库的结构

王闯舟 (93.1.6,131)

MASM 5.0中 LINK 时死机的原因及解决方法

MASM 5.0较之 MASM 4.0增加了不少功能,但经常使用 MASM 5.0的用户会发现这个问题:MASM 5.0编译通过的程序在 LINK 时却会“死机”。笔者在此讲解发现这个问题及寻找解决这个问题的过程,这样不单是解决了这一具体问题,用户还可从笔者解决这个问题的过程中可获得一种通用的解决这类问题的方法。

笔者曾经写过一个程序:将文件 PAPER 读入内存并对之进行处理,程序如下:

```
CODE SEGMENT
    ASSUME CS:CODE,DS:CODE
    ORG 100H
START: JMP BEGIN
FNAME: DB 'PAPER',0
BUFFER: DB 40000 DUP(0)
BEGIN: LEA DX,FNAME
    MOV AX,3D00H
    INT 21H
    JC OPEN_ERR
    MOV BX,AX
    LEA DX,BUFFER
    MOV AH,3FH
    INT 21H
    MOV AH,3EH
    INT 21H
    .
    .
    .
CODE: ENDS
END START
```

为节省篇幅,文中省略了打开文件出错的处理和文件的处理过程,

该程序在用 MASM 5.0编译时通过,但在 LINK 时却死机。笔者怀疑是否 LINK.EXE 文件被破坏,但 COPY 一个好的 LINK.EXE 后,问题仍然存在。无奈,笔者怀疑是否文件的处理过程导致死机,因此将文件的处理过程去掉,只剩下打开文件、读文件和关闭文件,LINK 时仍然死机。笔者又将读文件过程去掉,只剩下打开文件和关闭文件,在 LINK 时仍然死机。最后,笔者干脆使程序不做任何工作,而直接结束,程序如下:

```
CODE SEGMENT
    ASSUME CS:CODE,DS:CODE
    ORG 100H
START: JMP BEGIN
FNAME: DB 'PAPER',0
```

```
BUFFER: DB 40000 DUP(0)
BEGIN: MOV AH,4CH
    INT 21H
CODE: ENDS
END START
```

结果 LINK 仍然死机。这时笔者又将语句 FNAME DB 'PAPER',0 去掉,只剩下语句 BUFFER DB 40000 DUP(0),结果仍然死机。此时笔者怀疑是否因为 BUFFER 定义较大所致,因此 40000 去掉一个 0 改为 4000,LINK 时经过短通过后,将 4000 改为 10000,LINK 时经过很長時間后通过。由此注意:笔者只是将缓冲区大小由 4000 改为 10000,LINK 的时间就大大延长(此时可以推测:缓冲区大小为 40000 时,程序并未死机,只是处理的时间太长)。笔者以前使用 MASM 4.0 时,从未遇到这种情况,于是笔者想是否将缓冲区定义为 DUP(?) 时情况也是如此呢?于是将 DUP(0) 改作 DUP(?),重新 LINK 时很快通过,又将 10000 改作 40000,LINK 时仍然很快通过。由此可知,这是由于定义缓冲区时缓冲区被赋予了初始值才导致 LINK 时“死机”的。

因为 LINK 使用的输入文件是 .OBJ 文件,可以推测:将缓冲区定义为 DUP(0) 时将缓冲区定义为 DUP(?) 时,.OBJ 文件是不相同的,于是笔者对这两种情况分别进行编译,缓冲区定义为 DUP(0) 时的 .OBJ 文件如下页所示(.OBJ 的大小为 103):

可以看出:DUP(0) 时的 .OBJ 文件比 DUP(?) 时的 .OBJ 文件多了 17 个字节(从 141H 至 151H),这 17 个字节构成了 .OBJ 文件的重复块代码记录,可见在 DUP(?) 时没有重复块代码记录。

至此,我们知道,如果程序中定义了一个有初值的较大缓冲区,LINK 时就会“死机”。实际使用时,缓冲区大多数没有初值,因此在定义缓冲区时应把写成 DUP(0) 的习惯改为 DUP(?)。但在某些特殊情况下,缓冲区确实需要初始化为一特定值,在此情况下为避免 LINK 时“死机”,可将缓冲区定义语句 DUP(XX) 改作 DUP(?),而在程序中使用下面的几条指令来实现缓冲区的初始化:

```
LEA DI,BUFFER
MOV CX,XXXXX ;XXXXX 为缓冲区的大小
MOV AL,XX
CLD
REP STOSB
```

这样,既初始化了缓冲区,又很快地通过 LINK。

从笔者查找这个问题的过程中可以看出:笔者使用的是逐步缩小范围以定位问题的方法。对这种莫名

其妙的问题,可采用这种方法找出解决问题的方法,而不必去分析软件(比如本文的 LINK)。

```
C:\>DEBUG TL1.OBJ
```

```
-RCX      ;大小为103
```

```
CX 0067
```

```
-D100 167
```

```
36CA:0100 80 0E 00 0C 54 45 53 54-4C 49 4E 4B 2E 41 53 4D    ....TESTLINK.ASM
36CA:0110 E9 98 07 00 00 04 43 4F-44 45 44 98 07 00 60 47    i....CODEC...\G
36CA:0120 9D 02 01 01 19 88 04 00-00 A2 00 D1 A0 07 00 01    .....Q...
36CA:0130 00 01 E9 00 00 6E 9C 08-00 84 01 00 01 01 43 9D    ...i..n.....C.
36CA:0140 F5 A2 0E 00 01 03 01 40-9C 01 00 01 00 00 00 01    u.....e.....
36CA:0150 00 6C A0 08 00 01 43 9D-B4 4C CD 21 89 8A 07 00    .l...c.4LM1.....
36CA:0160 C1 00 01 01 00 01 AB E8                                A....+h
```

将缓冲区定义为 DUP(?)时的.OBJ 文件如下(OBJ 文件的大小为86) :

```
C:\>DEBUG TL2/OBJ
```

```
-RCX
```

```
CX 0058      ;大小为86
```

```
D 100 156
```

```
36CA:0100 80 0E 00 0C 54 45 53 54-4C 49 4E 4B 2E 41 53 4D    ...TESTLINK.ASM
36CA:0110 E9 96 07 00 00 04 43 4F-44 45 44 98 07 00 60 47    i....CODED...\G
36CA:0120 9D 02 01 01 19 88 04 00-00 A2 00 D1 A0 07 00 01    .....".Q....
36CA:0130 00 01 E9 00 00 6E 9C 08-00 84 01 00 01 01 43 0D    ...i..n.....C.
36CA:0140 F5 A0 08 00 01 43 9D B4-4C CD 21 89 8A 07 00 C1    u...C.4LM.....A
36CA:0150 00 01 01 00 01 AB 43                                .....+c
```

王振祥 (93.1.6,139)

测试 CRT 的模式

目前个人计算机的显示器及其适配器的类型非常多,有单色的,也有彩色的。单色中又分 MDA、CGA、VGA 等等;彩色的也分 CGA、EGA、VGA 等等。在微机显示器及其适配器的开发与使用过程中,常常需要知道当前正使用的是什么模式?下面给出一个程序可以很方便地测出当前正使用的模式。

程序是用 TURBO C 2.0 写成的。利用它所提供的函数,经编译、连接后,生成 EXE 文件,就可在命令行上直接键入 CRTMODE 来运行。程序已经在各种微机上使用过,包括 IBM-PC、XT、AT,以及各种兼容的286、386等微器。

```
#include <stdio.h>
#include <graphios.h>
```

```
#define P(note) printf(note)
#define PV(format,value) printf(format,(value))
#define PM printf(" mode is ")
#define PD printf("\n\nDetected graphics _driver is ")
void main(void)
```

```
{ int g_driver,g_error,g_mode;
detectgraph(&g_driver,g_mode);
if(g_driver<0)
{ p("No graphics hardware detected!\n");
exit(1);
}
switch(g_driver)
{ case 1: PD;P("CGA,");
switch(g_mode)
{case 0: PM; P("CGAC0 320 * 200");
case 1: PM; P("CGAC1 320 * 200");
case 2: PM; P("CGAC2 320 * 200");
case 3: PM; P("CGAC3 320 * 200");
case 4: PM; P("CGAHI 640 * 200");
}
break;
case 2: PD; P("MCGA,");
switch(g_mode)
{case 0: PM; P("MCGAC0 320 * 200");
case 1: PM; P("MCGAC1 320 * 200");
case 2: PM; P("MCGAC2 320 * 200");
```



```

case 3: PM; P("MCGAC3 320 * 200");
case 4: PM; P("MCGAMED 640 * 200");
case 5: PM; P("MCGAHI 840 * 480");
}
break;
case 3: PD; P("EGA,");
switch(g_mode)
{ case 0: PM; P("EGALO 640 * 200");
case 1: PM; P("EGAHI 640 * 350");
}
break;
case 4: PD; P("EGA64,");
switch(g_mode)
{ case 0: PM; P("EGA64LO 640 * 200");
case 1: PM; P("EGA64HI 640 * 350");
}
break;
case 5: PD; P("EGAMONO,");
PM; P("EGAMONOH 640 * 350");
break;
case 6: PD; P("IBM8514,");
switch(g_mode)
{ case 0: PM; P("IBM8514LO 640 * 480");
case 1: PM; P("IBM8514HI 1024 * 768");
}
break;
case 7: PD; P("HEROMONO,");
PM; P("HEROMONO 720 * 348");
break;
case 8: PD; P("ATT400,");
switch(g_mode)
{ case 0: PM; P("ATT400C0 320 * 200");
case 1: PM; P("ATT400C1 320 * 200");
case 2: PM; P("ATT400C2 320 * 200");
case 3: PM; P("ATT400C3 320 * 200");
case 4: PM; P("ATT400MCD 640 * 200");
case 5: PM; P("ATT400HI 640 * 200");
}
break;
case 9: PD; P("VGA,");
switch(g_mode)
{ case 0: PM; P("VGAIO 640 * 400");
case 1: PM; P("VGAMED 640 * 350");
case 2: PM; P("VGAHI 640 * 480");
}
break;
case 10: PD; P("PC3270,");
PM; P("PC3270HI 720 * 350");
break;
}
P("\n\n\n\tTHANK YOPU VERY MUCH FOR USING
THE CRTMODE PROGRAM!");
}

```

何 渝 (93.1.6,141)

紫金多用户卡的故障排除

ZJ286 8I/O 卡(8串一并),串口采用 RS-232C 标准,并口采用 Centronics 标准。该卡适用于 DOS 系统和 Xenix 系统支持下的多用户,串口由通用可编程异步接口片(USART)INS8250B 和一些电平转换芯片组成;并口是由一些中小规模 TTL 通用电路组成。

并口常见故障与维修

打印口出现故障一般有两类:一类是插上 I/O 卡后主机无法启动;第二类是不影响主机启动,但不能打印。

(1)插上 I/O 卡后主机不能启动:

这种故障一般是 I/O 卡中被损坏的芯片将系统总线的地址线或数据线钳位成固定电平,或者是 I/O 卡上的 +5V 电源与地短路。用万用表和逻辑笔测量,地址线和数据线正常时均应有脉冲。若电源、数据线和地址线均正常,可再查 $\overline{IDR}$ 、 $\overline{IOW}$ 、Busy、ACK 信号

是否正常,如果“忙”信号(Busy)和“回答”信号 ACK 出现故障,会导致主机死锁或等待。当上述信号均正常时,一般不影响主机启动。

(2)不影响主机启动,但联机不打印:

当打印机处在联机状态下,主机送出打印数据后,打印机不打印,这种现象一般分为锁死主机和不锁死主机两种状态。送出打印数据后锁死主机的情况较常出现,这种故障常常由命令译码器、数据收发器和控制信号几部分引起。

检查这类故障时,一般可先用随机诊断程序进行检查,但随机诊断程序功能不强,画面提示仅出现 PASS 或 ERROR,只能大致判断 I/O 卡是否有故障,而不能具体定位。但可借助汇编语言和系统资源,自己动手编制功能强的诊断程序,限于篇幅,本文不再叙述诊断程序的编制方法。现介绍一种用 DEBUG 对打印口定位的简易方法。在 DOS 操作系统下调用动