电脑编程技巧与维护月刊

圆圆圆年第 怨期　　总第 苑缘期　员怨怨源年 苑月创刊）　　每月 猿日出版

名誉社长　张　琪
社　　长　孙茹萍
副 社 长　毕研元
总　　编　王路敬
执行主编　叶　永
副 主 编　杨林涛
编　　辑　程　芳　管逸群
　　张　涛
公关部主任　黄德琏
出版发行部　毕　波
编辑出版　《电脑编程技巧与维护》杂志社
法律顾问　佟秋平 万亿中律师事务所雪
主办单位　中国信息产业商会
社　　址　北京海淀区学院南路 远愿号吉安大厦 源园苑室
投稿信箱
编辑部信箱
网　　址
邮　　编　员园园园员
电　　话　园员园-员园苑园苑园-员园苑苑缘
传　　真　园员园-员园苑园苑园-员园苑
照　　排　《电脑编程技巧与维护》杂志社电脑排版部
印　　刷　北京巨龙印刷厂
订 阅 处　全国各地邮电局
国内总发行　北京报刊发行局
邮发代号　愿园-苑缘
刊　　号　陨缘-员员园园 员园原愿缘园
　　怀园员员原愿缘园-员园
广告许可证　京海工商广字 园园缘苑
全年定价　怨缘员元
每期定价　苑缘员元

诚聘软硬件人才

北京飞天诚信科技有限公司从事软件防盗版产品的开发研究。推出的软件加密产品 砸粉悦 再源型 韵狗 接口的加密锁，成为市场上的主流产品。公司目前向网络安全领域进军，业务成倍增长。现诚聘软硬件人才：

＊ 软件编程人员：能非常熟练地使用 宇悦 悦垣垣 能独立完成完整的软件编程。编程经验 缘员园行以上）对 陨遭遭熟悉，对软件加密有一定程度了解，并对软件加密工作有兴趣和热情者。精通或了解 猿园位汇编，编写过 宰陨遭园下的设备驱动程序者优先考虑。

＊ 网络编程人员：熟悉局域网络的多种协议，栽没辑源 暑表用杂陨载 网络编程经验 缘员园行以上。

＊ 硬件开发人员：熟悉 孕悦机并口和 韵狗 接口编程、单片机开发、数字电路设计、云藻等工 耘云砸韵工 陨悦卡应用。

欢迎北京以外应届毕业，如果你在校是同学公认的计算机迷，电脑虫，上机动手能力、自学能力最强。那你不要犹豫，快将简历 耘原宅酱本公司。你正是我们寻求的人才。

学历、性别不限，提供宿舍，及面试费用。**确有真才实学。**

单位地址：北京市海淀区学院路蓟门饭店五号楼三层 员园园愿

耘源酱园员园原愿园员园援云藻等耘砸韵援宰宰宰援砸粉悦 再粉韵援暑

耘彦酱员园员园原愿园员园援孕耘陨援砸粉悦 联系人：黄先生

新技术追踪

缘苹果推出世界首台微型超级电脑等 怨篇

编程与应用起步

远汇编语言程序设计系列讲座 匀）

愿穴阅使用的一点技巧

怨将 阅宰郧格式三维模型直接转为 猿杂格式时若干问题的探讨

员远球藻算法对象的缺点与改进

员园宰蜜蝶算法下 悄酒杂写保护技术的实现

员源酝云悦应用程序框架打印预览功能分析与扩展

员愿在 穴月中实现电视般字幕滚动效果的几种方法

员怨穴月对 宰熄蜘苑中书签的编程

圆园用 阅藻蚤藻实现热点

编程语言

圆员在系统托盘中创建快捷菜单

猿远获取中断历史入口地址的实用程序

猿缘阅藻蚤藻之查询输入释疑

需要全本请访问 www.ertongbook.com

《电脑编程技巧与维护》杂志社授权佟秋平律师声明

《电脑编程技巧与维护》杂志社出版的《电脑编程技巧与维护》期刊，以荟萃国内外电子计算机先进信息，以领先性、实用性而享誉电脑报刊市场。但是近来《电脑编程技巧与维护》杂志社接到许多读者询问：“重庆科潮电子图书有限公司”在全国多种报刊上刊登邮购本社杂志 [illegible] 年合订本的广告。已有一些读者将定购款汇至“重庆科潮电子图书有限公司”后却未得到本社 [illegible] 年合订本，且与“重庆科潮电子图书有限公司”无法联系。“重庆科潮电子图书有限公司”的行为严重影响了《电脑编程技巧与维护》杂志社的声誉，也损害了读者的权益。为此，本律师郑重声明：

《电脑编程技巧与维护》杂志社的发行和销售渠道是报刊发行局和各地新华书店及部分代理书店。《电脑编程技巧与维护》杂志社并未授权“重庆科潮电子图书有限公司”刊登邮购广告。敬告广大读者如需订阅《电脑编程技巧与维护》杂志可向上述代销商订购，也可直接向本杂志社订购。希望广大读者提高警惕，辨清真伪。

特此声明

《电脑编程技巧与维护》杂志社法律顾问：佟秋平律师

[illegible] 年 [illegible] 月 [illegible] 日

专家论坛

[illegible] 面向 [illegible] 编程人员的 [illegible] 综述

可视化专栏

[illegible] 实现缩略图查看

[illegible] 实现全屏显示

[illegible] 的多态窗体

[illegible] 文件系统对象编程

数据库

[illegible] 中使用 [illegible] 数据

[illegible] 动态 [illegible] 语句及在组合查询中的应用

网络技术

[illegible] 中对模板文件的处理

[illegible] 实现数据库中图片上传与存储

[illegible] 实现在对话框中浏览网页

图形图像处理

[illegible] 为您的应用程序建立投影式立体窗口

[illegible] 图像在窗口中无闪烁移动的 [illegible] 实现方法

计算机维护

[illegible] 创建和编辑 [illegible] 的菜单

计算机安全

[illegible] “爱虫”病毒的 [illegible] 内核技术剖析

[illegible] 密码大揭秘

博士信箱

[illegible] 从中文 [illegible] 上网常见问题及其处理

网上征订：

[illegible]

品质服务才是硬道理

——从洪恩“海啸”看中国软件业的必然趋势

[illegible]年[illegible]月[illegible]日，在钓鱼台国宾馆金洪恩公司的“洪恩海啸世纪行”新闻发布会上传出惊人消息：洪恩软件要在中国软件业界掀起涨价的浪潮，“品质加[illegible]元，服务加[illegible]元”，洪恩软件要将中国软件推向以品质和服务为特征的新竞争时代……在全行业大规模的降价风潮以后，洪恩软件为什么要逆潮流而动不降反涨呢？

[illegible]年，中国软件业有史以来最大的降价浪潮——多家国内软件企业“跳楼价”力推低价软件，价位之低令人难以想象，从[illegible]元到[illegible]元不等；一时间软件再也不是专卖店中高高在上的高档消费品了，而是出没于街头巷尾的报摊上了，甚至还出现了低于[illegible]元的软件……

从业内比较权威的连邦软件销售排行榜和首都图书大厦软件销售榜的数据分析，即使在降价浪潮的颠峰时期，始终有一家公司的产品销售额名列榜首，它就是金洪恩。金洪恩的全称是北京金洪恩电脑有限公司，熟悉中国软件行业的人或者用过洪恩产品的消费者都知道，洪恩软件的产品品质在国内是首屈一指的，售后服务更是无人能出其左右。

“洪恩软件不降价；洪恩要涨价：品质加[illegible]元，服务再加[illegible]元。”洪恩继去年的《正版宣言》之后再次推出以“品质服务”为核心的洪恩“海啸”世纪行活动。通过三年的市场磨练，洪恩树立了中国教育软件第一品牌的形象，赢得了数百万用户的支持和信任，但洪恩人从激烈的市场竞争中也情形地认识到，对于一个[illegible]企业的发展，产品的品质和服务才是根本。“会员卡”、“投诉卡”、“会员会刊”、“投诉有奖”、“免费用户培训”、“用户服务车”、“社会监督员制度”等一系列的服务措施是一些颇有实力的[illegible]企业都做不到的。

连邦排行榜今年[illegible]到[illegible]月的数据显示，洪恩上榜的产品绝大部分在[illegible]元以上，甚至包括了[illegible]多元的《英语世纪行》和《电脑世纪行》套装礼包，倒是排行榜上不见了[illegible]元的软件。这些说明了一个软件是否能成功的根本不在于价格，还是品质和服务。

中国的软件行业比之发达国家还落后很多，比不上我们的邻国印度，但仍然有一些公司挺立[illegible]潮头，获得了社会和市场的认可。比之有些产品不断的“快枪手”公司，洪恩的产品推出很慢，但这种慢是对用户的负责。从内容、技术、美工、动画，洪恩的产品很难让人说出什么不是。从当年颇具传奇色彩的《开天辟地》的一张光盘，到今天[illegible]随心所欲[illegible]的[illegible]张光盘，洪恩的产品没有涨价，但无论是容量还好是服务却都在增值。所以洪恩之所以提出“品质加[illegible]元，服务再加[illegible]元”也来源于对自己产品的坚定信心。

对于[illegible]行业来说，判断一个企业的成功应该看这个企业的可持续发展能力。洪恩此次的“涨价”可以说是对自己的一次检验，洪恩希望借此次行动探索一条国内软件企业的发展道路。中国的软件业要想达到真正腾飞不能一味降价，不能也不可能谈什么与盗版拼价格，更不能靠媒体的盲目炒做和误导，而必须从产品的品质与服务上下功夫，而这种功夫应该是持久而连续的。试问有哪个企业的老总会亲自给投诉自己的用户回信？而洪恩总裁池宇峰就是这样。所以也就不难理解洪恩在“海啸”活动中开天辟地地提出自己的主张：“我们是中国软件的服务的典范，我们为用户提供优质的服务，所以我们要涨价。”作为一个几近成熟的中国软件企业，洪恩已经有了一套完整的用户服务体系，可以从售前、售中、售后上给予用户有力的支持。在伴随中国软件走过三年的风风雨雨之后，在近百万的用户热情洋溢的反馈信中，洪恩深刻体会到打击盗版，靠价格绝对是不可能战胜盗版的，惟有服务，增加大量盗版无法做到的增值服务，才是正版软件走向最终胜利的必由之路。

很希望有更多的人能够来关心我们的软件业，我们也希望中国的软件业能够涌现出更多的“洪恩”和“洪恩海啸”，真正使中国的软件业迎接[illegible]的挑战，真正地走向世界。

苹果推出世界首台微型超级电脑

苹果日前推出世界上第一台微型超级个人电脑 [illegible]只有普通电脑四分之一大，装在一个明亮如水晶一样的透明方盒内，内部安装具有超级电脑功能的 ‘动力电脑处理器 ” [illegible]每秒钟的运算速度超过 [illegible]亿次！新产品加入了最新的科技如乙太网 [illegible]快速传递数据的 [illegible]和 [illegible]接口等，最令人称奇的是它的数码录像驱动器安在 [illegible]上面，只要轻轻按动开关键，光碟便徐徐升起。它的零售价格是 [illegible]美元，不包括显示器。

[illegible]超级商用计算机研制成功

[illegible]发布了一款号称世界上功能最为强大的商用超级计算机，每秒可完成 [illegible]万亿次运算。[illegible]公司称这台被命名为 [illegible]运算速度比普通个人计算机快 [illegible]万倍的超级计算机是上个月出售给美国能源部用于模拟核实验的计算机的商用机型。这台计算机的微处理器为铜质微处理器，而非硅微处理器，还采用了硅开关技术并安装了相应的软件，可用于电子商务和工程设计等用途，如飞机设计。[illegible]公司设计的这款型号为 [illegible]的商用超级计算机，目的是要充分利用超级计算机应用从科学研究转向商业领域 [illegible]如互联网业务、企业数据中心等 [illegible]新产生的巨大商机。

英特尔推出使用传统内存的奔腾 [illegible]芯片组

‘芯片巨头 ” 英特尔公司发布消息称，该公司将于明年推出一个使用传统内存的 ‘奔腾 [illegible]’ 芯片组。新推出的 ‘奔腾 [illegible]’ 芯片组将使用传统的 [illegible]内存，其 [illegible]外频 [illegible]运行速度将达 [illegible]完全能够发挥 ‘奔腾 [illegible]’ 处理器的强大功能。此前，英特尔公司的高档芯片组都只能使用 [illegible]内存芯片，而 [illegible]内存的成本极高，其价格至少是标准的 [illegible]内存的三倍。

[illegible]和 [illegible]提供远程 [illegible]缓存

[illegible]公司日前宣布将给它的 [illegible]小企业和远程 [illegible]分支办公室网络产品系列增加 [illegible]缓存功能。[illegible]公司将在它的 [illegible]、[illegible]和 [illegible]调制解调器，以及 [illegible]路由器和 [illegible]远程存储系统中捆绑 [illegible]公司的 [illegible]缓存软件。该软件能使与 [illegible]设备相连的 [illegible]共享 [illegible]浏览器的高速缓存空间，为联网的所有 [illegible]提供共用的 [illegible]缓存。

[illegible]推出新型图形工作站

美国图形工作站生产厂商 [illegible]简称 [illegible]宣布，将推出 [illegible]系列的新型图形工作系统，该系列产品在模式设计和计算机体系结构等各方面，均有突破性的创新。将为用户提供灵活的、极富弹性的、高性能的和具有伸缩性的工作系统。其中，[illegible]系列服务器、[illegible]系列的视觉化系统，不久即将开始发售。上述两个系列的即将问世的产品中，都应用了 [illegible]公司独特的 [illegible]模块技术，因而可以为用户在特定状态下，提供最优化的配置，并将根据顾客所在行业的不同需求，灵活地接纳或兼容新的商业和研究技术。如果希望进一步了解有关的产品信息，可访问以下网址：[illegible]或 [illegible]

微软公司即将推出全新操作系统

微软公司宣布它不久将发行新 ‘视窗 ” 软件的第一个试用版 —— [illegible]这个新的软件将把公司的两个计算机操作系统合二为一，以节省公司的开发时间和资金。除了建立在 [illegible]技术基础上这一最突出的优点外，[illegible]还具有更好的安全性，重新启动时间更短，它只需 [illegible]秒。这个叫作 [illegible]的软件试用版将于今年 [illegible]月发行，仅比 ‘视窗 ” 千禧版的发行时间晚一个月。 ‘视窗 ” 千禧版是向普通用户提供的一种 [illegible]产品的改进版。[illegible]也是针对家庭用户设计的，它是以 [illegible]为基础的，也就是用于 ‘视窗 ” [illegible]的技术重心是微软公司为网络大工作量计算机开发的一种操作系统。

[illegible]推出转速为 [illegible]月硬盘

美国 [illegible]公司宣布，该公司将推出转速为 [illegible]，单碟容量为 [illegible]月的 [illegible]英寸硬盘 [illegible]‘[illegible]’。该产品 [illegible]年 [illegible]月开始出售。至此，[illegible]公司的产品有使用单碟，容量为 [illegible]月的产品和 [illegible]碟 [illegible]月的产品共 [illegible]种。接口采用最大数据传输速度为 [illegible]的 [illegible]，[illegible]月产品在美国的单价为 [illegible]美元。预计在日本市场的价格为 [illegible]万 [illegible]日元。转速 [illegible]，单片容量 [illegible]月的产品还是首次上市。到目前为止的硬盘即使容量相同转速也只有 [illegible]，[illegible]公司在此之前已经推出了容量相同但转速只有 [illegible]的低速产品。

[illegible]新款大型机携 [illegible]亮相在即

[illegible]公司近日宣布即将推出新的大型机软硬件，旨在更好地支援庞大的电子商务网站。[illegible]将推出的下一代 [illegible]大型机代号 [illegible]，它采用 [illegible]，[illegible]的上市名称有可能也叫 [illegible]。此外，[illegible]公司还将推出大型机操作系统 [illegible]，该版本为运行于 [illegible]体系结构进行了优化。它为网络中的关键用户或工作组支持 [illegible]特色，给大型计算中心提供了更广泛的安全、[illegible]监控和存储功能。[illegible]还将改善对大型机群集管理的支持，确保关键用户或工作组优先使用网络应用。

英特尔正式推出主频为 [illegible]的微处理器

日前，英特尔宣布已正式推出工作频率为 [illegible]的微处理器，从而在芯片大战中再次领先对手 [illegible]公司。[illegible]公司曾于今年领先英特尔公司数日发布了工作频率为 [illegible]的微处理器，从而在芯片速度大战中稍占上风，该公司于今年 [illegible]日推出频率为 [illegible]的微处理器。此次，英特尔公司发布 [illegible]微处理器使该公司在高端芯片市场上再度占据领先地位。据英特尔公司称，新推出的 [illegible]微处理器批发售价约在 [illegible]美元左右，其发售数量比较有限。

为您服务网站
[illegible]
免费资源总汇
名优网站直达

汇编语言程序设计系列讲座（二）

第三讲 汇编语言程序的调试

李玉珏 徐 亮

开发汇编语言程序的最后一步就是对可执行文件进行调试，检查程序的功能是否正确。DOS提供程序调试工具DEBUG.COM，它不仅是动态调试器，也是二进制文件编辑器。DEBUG能提供一个动态调试程序的环境，可以帮助用户调试程序。

本讲，我们通过对第一讲例程序的调试，来说明DEBUG的使用方法。

需说明的是：本文中凡是用户键入的字符都使用加粗字符。

；进入DEBUG。可采用两种方式。直接调用DEBUG或是在调用DEBUG的同时装入被调试程序：

```
C>DEBUG                    ;采用直接调用 DEBUG 方式
-N BCDTOASC.EXE;然后使用命名命令 N 和装载命令 L,装入被调试程序
-L                         ;
C>DEBUG BCDTOASC.EXE       ;采用调用 DEBUG 的同时装入被调试程序 BCDTOASC.EXE 文件方式
```

；利用R命令，看看程序装入后，在执行之前各寄存器的内容：

```
-R          ;R 命令用来显示各寄存器的内容和下一条要执行的指令
AX=0000  BX=0000  CX=0051  DX=0000  SP=0050  BP=0000  SI=0000  DI=0000
DS=199E  ES=199E  SS=19B4  CS=19B0  IP=0000  NV UP EI PL NZ NA PO NC
19B0:0000 B8AE19        MOV     AX,19AE
```

；其中，八个标志位状态符号含义见下表

标志名称	溢出	方向	中断	符号	零	辅助进位	奇偶	进位
置位状态	OV	DN	EI	NG	ZR	AC	PE	CY
复位状态	NV	UP	DI	PL	NZ	NA	PO	NC

；利用R命令可设置有关标志位：

```
-R F                       ;设置零和进位标志位
NV UP EI PL NZ NA PO NC  -ZR CY
-R F                       ;看看修改后的标志寄存器的内容
NV UP EI PL ZR NA PO CY  -
```

；利用R命令可修改寄存器的内容：

```
-R DX                      ;显示 DX 内容
DX 0000
:1234                      ;修改 DX 内容
-R DX                      ;看看修改后的 DX 的内容
DX 1234
:
```

；利用反汇编命令U，将内存中的机器码以助记符的形式显示出来：

```
-U 0,002B;注意:U 命令中所指定的起始地址应在一条有效指令的开始处.
19B0:0000 B8AE19        MOV     AX,19AE
19B0:0003 8ED8          MOV     DS,AX
19B0:0005 8EC0          MOV     ES,AX
19B0:0007 BE0000        MOV     SI,0000
19B0:000A BF0900        MOV     DI,0009
19B0:000D B90900        MOV     CX,0009
19B0:0010 FC            CLD
19B0:0011 AC            LODSB
19B0:0012 8AD8          MOV     BL,AL
19B0:0014 240F          AND     AL,0F
19B0:0016 0C30          OR      AL,30
19B0:0018 AA            STOSB
19B0:0019 8AC3          MOV     AL,BL
19B0:001B 51            PUSH    CX
19B0:001C B104          MOV     CL,04
19B0:001E D2E8          SHR     AL,CL
19B0:0020 59            POP     CX
19B0:0021 0C30          OR      AL,30
19B0:0023 AA            STOSB
19B0:0024 E2EB          LOOP    0011
19B0:0026 BA0900        MOV     DX,0009
19B0:0029 B409          MOV     AH,09
19B0:002B CD21          INT     21
```

；利用跟踪命令T，可执行一条或多条指令。例如用T3则跟踪执行三条指令：

```
-T3                ;注意使用 T 命令一般情况下不要进入 DOS 功能调用程序和 BIOS 程序.
AX=19AE  BX=0000  CX=0051  DX=0000  SP=0050
BP=0000  SI=0000  DI=0000  DS=199E  ES=199E
SS=19B4  CS=19B0  IP=0003  NV UP EI PL NZ NA PO NC
19B0:0003 8ED8          MOV     DS,AX
AX=19AE  BX=0000  CX=0051  DX=0000  SP=0050
BP=0000  SI=0000  DI=0000  DS=19AE  ES=199E
SS=19B4  CS=19B0  IP=0005  NV UP EI PL NZ NA PO NC
19B0:0005 8EC0          MOV     ES,AX
AX=19AE  BX=0000  CX=0051  DX=0000  SP=0050
BP=0000  SI=0000  DI=0000  DS=19AE  ES=19AE
SS=19B4  CS=19B0  IP=0007  NV UP EI PL NZ NA PO NC
19B0:0007 BE0000        MOV     SI,0000
```

；利用显示内存单元命令D，看看数据段的内容（其中，段值为实际装入的地址）：

```
-D 19AE:0  L28     ;显示以 0 为起始偏移量,长度为 28 个字节内的存单元内容.
19AE:0000  25 41 30 16 38 79 15 31-73 00 00 00 00 00 00 00   %A0.8y.1s.......
```


www.ertongbook.com

19AE: 0010 00 00 00 00 00 00 00 00 –00 00 00 24
. . . . . $

；上述信息中，左边是内存单元地址，中间部分是字节值，右边是字节值作为 粤栽悦[[码所对应的符号，对于非 粤栽悦[[码或非显示符号，用点或空格表示。

；利用运行命令 郧，可以实现全速运行和断点运行。通过断点运行，能对程序分段调试。

–G =7 0026 ；**在** G **命令中，第一个参数(等号后的地址值)是运行的起始地址，后面的地址是断点地址. 如果** G **命令没有指定起始地址，那么程序就以** CS: IP **的现行内容作为起始地址执行，直到遇到断点或程序正常终止而结束. 在** G **命令中允许设置** 10 **个断点，程序运行到任一断点地址时便立即停下来，并显示各寄存器的内容和下一条要执行的指令.**

AX =1937 BX =0073 CX =0000 DX =0000 SP =0050
BP =0000 SI =0009 DI =001B DS =19AE ES =19AE
SS =19B4 CS =19B0 IP =0026 NV UP EI PL NZ NA PO NC
19B0: 0026 BA0900 MOV DX, 0009
–D DS: 0 L28 ；**显示数据段内存空间的内容**:
19AE: 0000 25 41 30 16 38 79 15 31 –73 35 32 31 34 30 33 36
%A0. 8y. 1s5214036
19AE: 0010 31 38 33 39 37 35 31 31 –33 33 37 24
18397511337$
；**此时**, BCD **码已转换完毕，前** 9 **个字节为待转换的** BCD **码，紧接的** 18 **个字节为转换的结果，最后一个字节**(24H) **为** '$'.

；利用执行过程命令 孕可执行一条或是多条指令，也可以执行一个过程。

–p3 ；**注意**: **在此位置**(CS: IP **为** 19B0: 0026) **执行** P3 **命令之后，完成了** 9 **号功能调用，显示了 "**521403618397511337 **"字符串**.

AX =1937 BX =0073 CX =0000 DX =0009 SP =0050
BP =0000 SI =0009 DI =001B DS =19AE ES =19AE
SS =19B4 CS =19B0 IP =0029 NV UP EI PL NZ NA PO NC
19B0: 0029 B409 MOV AH, 09
AX =0937 BX =0073 CX =0000 DX =0009 SP =0050
BP =0000 SI =0009 DI =001B DS =19AE ES =19AE
SS =19B4 CS =19B0 IP =002B NV UP EI PL NZ NA PO NC
19B0: 002B CD21 INT 21
521403618397511337; 9 **号功能调用已执行完毕，显示转换结果信息**:
"521403618397511337 "
AX =0924 BX =0073 CX =0000 DX =0009 SP =0050
BP =0000 SI =0009 DI =001B DS =19AE ES =19AE
SS =19B4 CS =19B0 IP =002D NV UP EI PL NZ NA PO NC
19B0: 002D B44C MOV AH, 4C

；利用 砸**命令，不仅可以显示各寄存器的内容、修改通用寄存器和段寄存器的内容、修改标志寄存器的内容，而且可以修改指令指针 陨:**

–R IP ；**修改指令指针** IP, **使其指向** CS: 0026
IP 002D
: 0026
–T3 ；**在相同的位置**(CS: IP **为** 19B0: 0026) **下，执行** T3 **命令与** P3 **命令的区别**:

AX =0924 BX =0073 CX =0000 DX =0009 SP =0050
BP =0000 SI =0009 DI =001B DS =19AE ES =19AE
SS =19B4 CS =19B0 IP =0029 NV UP EI PL NZ NA PO NC
19B0: 0029 B409 MOV AH, 09
AX =0924 BX =0073 CX =0000 DX =0009 SP =0050
BP =0000 SI =0009 DI =001B DS =19AE ES =19AE
SS =19B4 CS =19B0 IP =002B NV UP EI PL NZ NA PO NC
19B0: 002B CD21 INT 21
AX =0924 BX =0073 CX =0000 DX =0009 SP =004A
BP =0000 SI =0009 DI =001B DS =19AE ES =19AE
SS =19B4 CS =0B4D IP =0445 NV UP DI PL NZ NA PO NC
0B4D: 0445 EAA004C305 JMP 05C3: 04A0
–

；对比执行 孕命令和 耘命令后的结果可知：执行 耘命令 只是运行了三条指令；而执行 孕命令 是运行了两条指令和一个过程。

；利用 耘命令，可以修改内存单元数据。

；修改之前，数据段的内容如下：

–D 19AE: 0 9
19AE: 0000 25 41 30 16 38 79 15 31 –73 %A0. 8y. 1s

；修改内存单元数据的操作如下：

–E 19AE: 0 ;
19AE: 0000 25. 01 41. 02 30. 03 16. 04 38. 05 79. 06 15. 07 31. 08
19AE: 0008 73. 09

；修改之后，数据段的内容如下：

– D 19AE: 0 9
19AE: 0000 01 02 03 04 05 06 07 08 –09 . . . . . .

；利用 粤命令，可直接修改内存中的指令代码。

–U 7 10 ；**修改前的目标代码**
19B0: 0007 BE0000 MOV SI, 0000
19B0: 000A BF0900 MOV DI, 0009
19B0: 000D B90900 MOV CX, 0009
19B0: 0010 FC CLD
–A 000D ；**利用** A **命令修改偏移地址为** 0DH **的指令**
19B0: 000D MOV CX, 0006
19B0: 0010
–U 7 10 ；**修改后的偏移地址为** 0DH **的指令代码**
19B0: 0007 BE0000 MOV SI, 0000
19B0: 000A BF0900 MOV DI, 0009
19B0: 000D B90600 MOV CX, 0006
19B0: 0010 FC CLD

；利用 匝命令，退出 调试程序.

–q ；**退出** DEBUG，**返回操作系统**.
C>

本讲要求：

员 了解 调试程序的功能。

圆 掌握进入和退出 调试程序的方法。

猿 掌握常用 调试程序命令及使用相应命令应注意的问题。

源 学会熟练使用 调试程序调试工具，能恰当地使用各种命令调试程序。如：运行程序可采用单条跟踪、过程跟踪、断点运行和全速运行。能够依据调试程序的实际情况而灵活选择恰当的命令。

收稿日期：圆园园园年 源月 圆园日）

宏阅使用的一点技巧

上海迅光科技系统有限公司

宏阅，听起来并不是个熟悉的名字，其实它是个数据库的开发工具。虽然它的名气还不怎么响，但用过了，就会有不少的感想。这里我想说说使用 宏阅的几点小技巧。

一、数据表间的过帐

在功能上，宏阅可能无法和 [illegible] 宏月等开发平台相比，但在有些方面却不能不说 宏阅比它们更胜一筹，例如开发速度。就从 宏阅实现数据表间的过帐来看，就有很多简便之处。举个例子说明会很清楚。例如在一个进销存的管理系统中，进货与出货都与产品的库存量息息相关，拿进货来说，我们希望在进货的同时对相应产品的库存进行增加的操作。在 宏阅中实现这一功能是很简便的，打开数据关联窗口，新增需要数据过帐的数据表（本例子中为三个，进货主档，进货明细档，以及产品库存档），其中数据的过帐是在明细档和产品库存档之间。用鼠标左键点击明细档的“数量”字段，将其拉至产品库存档的“库存量”字段，选择数据表过帐，这时会打开一个过帐明细窗口。其中“搜寻字段”是指在两个数据表之间的查找依据，本例中可以是产品的代号、名称等，而“目标字段”和“运算式”是过帐的主要内容，你可以在“运算式”中定义各种想要的公式，并选择是增或减等的操作，一个数据表的过帐功能就完成了。

二、数据表间的关联

当然在 宏阅中不仅是过帐简单易实现，而且实现数据表间的关联也是如此。同样在数据关联窗口，新增需要关联的数据表（在上例中的进货主档与明细档就存在着关联），首先要说明的是，在 宏阅中被关联的数据表要以关联字段为索引字段，也就是说只能与被关联数据表的索引实现关联，这是考虑到速度等因素的结果。如果我们是用进货的单据号码来实现关联，就点击主档的“单据号码”字段，拉至明细档的索引，选择数据表关联，打开关联明细窗口，其中最要说明的是“关联模式”与“删除模式”。“关联模式”可根据要关联的内容选择是一对多还是多对一，例如实现主档和明细档的关联，就是典型的一对多的例子，而要实现明细档与产品之间的关联，就是典型的多对一的例子了。如果选择“删除模式”，就可以实现在删除主表的同时，将明细档的被关联数据一并删除，这是个很重要的信息，千万不要漏选。

三、[illegible]语言

用到数据库，就无法不用到 [illegible]语言，它在做数据统计等方面表现十分灵活。在 宏阅中也专门有一个 [illegible]的精灵，帮助用户灵活准确的使用 [illegible]语言。同样在数据关联窗口，新增一个 [illegible] 打开精灵，新增要进行查询的数据表，可以是一个，也可以是多个，将要查询的字段直接拉至 [illegible]编辑器的窗口，当然也可以直接在编辑器窗口中输入 [illegible]语句。你还可以在 [illegible]浏览器中点击鼠标右键，选择要增加的语句，以及所用的字段等可以产生相同的效果。你还可以在结果浏览器中直接看到查询的结果。

以上所说的都是很基本、很简单的内容，但将他们适当的结合起来就可以实现很多复杂的功能。下面我想说说如何在应用中实现三者的结合。要完成的任务是，对进货主档进行增加、删除的操作，要求：[illegible]在新增时能在输入完仓库号后自动带入仓库名称；[illegible]新增后自动将单据代码带入明细档中；[illegible]输入完明细档后可以自动将相应的仓库中的相应产品的库存量增加；[illegible]输入明细档的产品时用下拉框，其中有产品代号，名称、该仓库的库存量等信息；[illegible]在删除主档的同时删除明细档的相关资料。

为了数据的冗余尽可能的减少，我在两个数据表间只建立一个关联字段。所以要实现自动带入仓库名称就要另想办法，将主档的仓库代号与仓库档的仓库代号两个字段关联，并在输入界面中显示主档的仓库代号，仓库档中相应的仓库名称，这样看上去就象在一个数据表中涵括了多种信息，而实际是将两个数据表的内容结合。这样就实现了第一个要求。

实现第二个要求也并不难，只要将主档的单据代码与明细档的单据代码关联起来，在主档新增存档后，明细档新增时就会自动产生单据代码。

第三个要求，在说明数据过帐时已经很详细的介绍了实现方法，所要说的是这里的过帐是用主档中的仓库代码和产品代码结合与库存档中的库存量进行过帐。

要想将随心所欲的数据放在下拉框，用 [illegible]是最合适的了。首先用 [illegible]语言将要放在下拉框下的数据查询出来，然后贴入一个输入元件 [illegible]将它的数据表定义为 [illegible]既刚刚生成的 [illegible]，经过对显示字段、传回字段的选择，再将明细档的产品字段的“取代编辑器”的属性定义为 [illegible]既刚刚贴入的元件名称），就可以实现第四个功能了。

最后一个，就不用说了，只要在主档与明细档的关联中选中“删除模式”就可以了。

以上所介绍的只是 宏阅众多功能中的很少部分，当然在有关进销存的例子中还有更复杂的，在这里我只想借此来简单的介绍一下 宏阅在使用数据表中的巧妙之处。如果大家有兴趣，不妨去试试，相信在简短的时间内会让你有很大的成就感。

收稿日期：[illegible]年 [illegible]月 [illegible]日）

将 阅宰郧格式三维模型直接转为 猿孕杂格式时若干问题的探讨

何　英　陈立伟

摘　要　本文探讨了将 阅宰郧格式三维模型转换为 猿孕杂格式时若干问题的解决方法。

关键词　阅宰郧，猿孕杂格式，三维模型，转换

随着 粤耘悦粤阅和 猿孕杂转存载这两大著名软件的普遍应用，人们已越来越多地感受到两者功能上的强大互补性。在实际应用中，使用者常常是先利用在 粤耘悦粤阅上专门开发的建模软件（如建筑之星、德赛等），方便、准确、快捷地搭建出三维模型，然后，再利用 粤耘悦粤阅的文件转存功能（耘曾责燥则贼），将其转存为 猿孕杂格式，在 猿孕杂转存载中进行动画制作。但是，在实际应用中我们发现，这一格式转换过程往往难以顺利完成，常常会发现在转存时出现“悦燥灶增则贼藻则藻则则燥则”提示符后，就出现死机现象。或是转存的模型在 猿孕杂转存载中无法正常显示。对此，我们通过大量的试验和分析研究，发现了原因所在，找出了一套行之有效的解决方法，现详细介绍如下。

一、 原因分析

通过对不能正常转换的模型进行分析和试验后，我们发现模型中用 猿则藻葬贼藻（孕藻则泽）、孕藻则泽（孕藻则泽）等命令生成的实体，是造成转换异常的原因。其中：

猿则藻葬贼藻实体是造成转换死机的直接原因。

孕藻则泽（孕藻则泽）的不正确处理造成相应模型在 猿孕杂转存载中不能正常显示。

二、解决方案

员）把一个大模型适当地划分成若干个区域，对每个区域用 粤耘悦粤阅的 耘曾责燥则贼功能分别试转成 猿孕杂格式，以确定出现转换异常的区域，缩小处理范围；

圆）将试转出现问题的区域中的 猿则藻葬贼藻实体用带厚度的 孕造葬灶藻代替，平面实体用 猿燥造凿枣葬糟藻代替（建议边替代边试转，如不再出现问题即可终止这部分区域的工作）；

猿）对于由 孕造葬灶藻（孕藻则泽）方式生成的复杂曲面（如水塔、油罐等），应将其单独逐个转出，转出后即将其删除，以防转换时发生内存不够错误；

源）为保证转换后的 猿孕杂模型能在 猿孕杂转存载中正常显示，应将 孕造葬灶藻（孕藻则泽）生成的实体以实体方式（杂燥造凿）转出，其他部分用图层（造葬赠藻则）方式转出。

三、 技巧和方法

由于待转换的模型中含有大量的 猿则藻葬贼藻（孕藻则泽）、孕造葬灶藻（孕藻则泽）实体，因而如何快速定位实体及进行相应的处理，对实际应用至关重要，根据我们的经验，可采用下述方法和技巧：

员）分区域试转，以便快速定位错误区域；

圆）对有问题区域的相关实体，边修改边试转，一旦转换成功，即行停止，以节约时间；

猿）对上述中的复杂曲面，单独转出，分别生成独立的 猿孕杂格式文件，然后把其从模型中删除；

源）当所有区域都可成功转换后，即可确认整个模型已处于待转状态；

缘）对整个模型按图层（造葬赠藻则）和实体（杂燥造凿）两种方式各转换一次 猿孕杂格式，生成两个 猿孕杂文件，然后，将这两个 猿孕杂文件和前面转出的若干复杂曲面模型文件到 猿孕杂转存载中，用 陨皿责燥则贼进行叠加即可。（说明：虽然这样会有部分模型同时参加两次转换，出现重复，但可保证不出现缺漏。如想尽可能减少转换后 猿孕杂文件的大小，则应按前述原则，用实体（杂燥造凿）方式转时，删掉无关模型，用图层（造葬赠藻则）方式转时，也同样处理，这样可有效地缩小文件大小，但工作量很大。）

以上是我们在实际工作中探索总结出的一套解决 粤耘悦粤阅中将三维模型转为 猿孕杂格式的方法，实践证明非常有效，在此愿将其与同行共享。

参考文献

[员] 美 演陈[圆]葬灶凿藻灶 悦燥灶灶藻造造 著 徐有光等译 援粤耘悦粤阅 员猿源从入门到精通 援电子工业出版社，员怨怨缘年 猿月

[圆] 美 演陈[圆]葬灶凿藻灶 悦燥灶灶藻造造等著 黄心渊等译 援粤耘悦粤阅 员猿源三维转存载技术精粹 援华大学出版社，员怨怨缘年 怨月

收稿日期：圆园园缘年 源月 圆园日）

类对象的缺点与改进

骆名群 黄海云

在 Borland C++Builder 中，对象库包含一个 类类，用于管理指针列表。然而，类类含有许多缺点，其中最严重的缺点是 类缺乏类型的安全性及内存空间的自动释放。本文将探讨 类类的缺点，并提出改进方法。

一、 类的问题

类主要用来存储对象的指针，使用方便，但是这个类的主要缺点是缺乏安全性，因为它存储并维护的是空指针 Pointer * 。让我们看看它的 Add 方法的原型：

```
int _fastcall Add(void * Item);
```

Borland C++Builder 编译器能正确地将任何类型的指针转换为 Pointer * 型指针，因此，类的 Add 方法能接受你传给它的任何一个指针。这其中就产出了一个问题，当你想维护 类中的对象指针列表时，你能肯定 类中的指针具有相同类型吗？例如：

```
TList * ButtonList = new TList;        // create a list of buttons
ButtonList->Add(Button1);    // add some buttons to the list
ButtonList->Add(Button2);             //
ButtonList->Add( new TButton(this));      // OK so far
ButtonList->Add(Application);   // Application is not a button
ButtonList->Add(Form1);               // neither is Form1
ButtonList->Add((void *)534);
ButtonList->Add(Screen);
```

以上代码能正确编译通过，从中看出，类能处理任何对象，它在接受任何一个对象时不做任何类型检查。当你要引用 类中的一个对象时，由于它是一个空指针，你必须将它转换为一个正确类型的指针，具体方法如下：

```
TList * ButtonList = new TList;
ButtonList->Add(Button1);
. . . . . // 生成一个对象指针列表
TButton * button = reinterpret_cast<TButton *>(ButtonList->
Items[1]);
//引用对象
button->Caption = "I hope it's really a button";
delete ButtonList;
```

在以上代码中，类的 Items 数组返回一个空指针 Pointer * 。要引用其中的一个 类型的指针，必须将其转换为 类型，这种转换的前提条件是你能肯定 类的 Items 返回的指针原来指向一个 对象。也许有人认为用 C++ 的 RTTI 操作符来完成这种转换，然而不幸的是由于空指针不含有任何类型信息，上述缺点仍然无法消除。

类的另一个缺点是它不能自动删除其列表中的指针，要完全删除列表，必须遍历整个列表，将列表中的指针一个个删除。例如：

```
TList * ButtonList = new TList;    // 生成一个对象列表
ButtonList->Add(new TButton(Handle));
// 向列表中添加 TButton 对象
ButtonList->Add(new TButton(Handle));
ButtonList->Add(new TButton(Handle));
ButtonList->Add(new TButton(Handle));
. . .
. . .
int nCount = ButtonList->Count;    //引用列表中对象的个数
for (int j=0; j<nCount; j++)
    delete ButtonList->Items[j];    //  删除指针
delete ButtonList;
```

从表面上看，上述代码非常完美，但是深入一点，我们会发现一个缺点：删除列表中的某个指针时，该指针是一个空指针，而删除一个空指针与删除一个 类型的指针是不同的，因为删除一个空指针不能自动执行一个对象的析构函数，也就不能执行析构函数中的内存释放与重新分配操作，从而会造成内存空洞。

为了能正确删除 类对象中的指针，必须将它们的类型进行转换，这样编译器才能正确调用类的析构函数。因为 C++ 中的所有析构函数都是虚函数，我们可以将列表中的所有转换为一个公共的基类，例如：

```
TList * ControlList = new TList;
ControlList->Add(new TButton(Handle));
   ControlList->Add(new TEdit(Handle));
ControlList->Add(new TComboBox(Handle));
int nCount = ControlList->Count;
   for (int j=0; j<nCount; j++)
delete reinterpret_cast<TWinControl *>(ControlList->Items[j]);
delete ControlList;
```

若 类对象中的指针对象都是从 类（或 类）继承下来的，则上述代码工作正确，若某指针对象是从 类（非 类）继承下来的，则上述代码工作又有问题。

二、 改进方法

如果 类知道它所处理的对象，那么它的许多缺点就自动消除。基于此，在 类的基础上我们引进了一个新类 类，它是从 类继承下来的一个模板类，它在 类方法的基础上提供了强制类型检查，同时也提供了自动删除列表中的指针的一个方法。代码如下：

```
//---------------------------
#ifndef TTYPEDLIST_H
#define TTYPEDLIST_H
#include <classes.hpp>
```


www.ertongbook.com

```
template <class T>
class TTypedList : public TList
{
private:
    bool bAutoDelete;
protected:
T* __fastcall Get(int Index)
{
return (T*) TList::Get(Index);
 }
void __fastcall Put(int Index, T* Item)
{
TList::Put(Index, Item);
 }
public:
__fastcall TTypedList(bool bFreeObjects = false)
:TList(),
bAutoDelete(bFreeObjects)
{
}
int __fastcall Add(T* Item)
{
  return TList::Add(Item);
}
void __fastcall Delete(int Index)
{
    if(bAutoDelete)
      delete Get(Index);
    TList::Delete(Index);
}
void __fastcall Clear(void)
{
  if(bAutoDelete)
  {
      for (int j=0; j<Count; j++)
        delete Items[j];
    }
    TList::Clear();
  }
T* __fastcall First(void)
{
  return (T*)TList::First();
}
int __fastcall IndexOf(T* Item)
{
    return TList::IndexOf(Item);
  }
void __fastcall Insert(int Index, T* Item)
{
    TList::Insert(Index, Item);
}
T* __fastcall Last(void)
{
  return (T*) TList::Last();
}
int __fastcall Remove(T* Item)
{
    int nIndex = TList::Remove(Item);
    if(bAutoDelete && (nIndex != -1))
      delete Item;
    return nIndex;
  }
__property T* Items[int Index] = {read=Get, write=Put};
};
#endif
```

最后根据改进的 [illegible] 类举个例子：

在 悦垣垣 [illegible] 中建一个新的应用程序，主要代码如下：

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
#include "typedlist.h"
//---------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
    : TForm(Owner)
{
}
//---------------------------
void __fastcall TForm1::CreateButtons()
{
// Create a list of buttons.
TTypedList <TButton> * ButtonList = new TTypedList <TBut-
ton>(false);
ButtonList->Add(new TButton(this));
ButtonList->Add(new TButton(this));
ButtonList->Add(new TButton(this));
for (int j=0; j<ButtonList->Count; j++)
{
    ButtonList->Items[j]->Caption="Button"+IntToStr(j);
    ButtonList->Items[j]->Left   = 250;
    ButtonList->Items[j]->Top    = 50 + j*25;
    ButtonList->Items[j]->Parent = this;
}
delete ButtonList;
  }
//---------------------------
void __fastcall TForm1::FormShow(TObject *Sender)
{
CreateButtons();
}
//---------------------------
```

以上代码在 悦垣垣 [illegible] 环境下中通过。

收稿日期：[illegible]）

宰蜜跐髒蘩愿下 惝礌狳写保护技术的实现

邓双成

惝礌狳是微机主板上的一块芯片，它保存着一些极为重要的系统信息，如系统时间、硬盘大小、软驱类型、开机密码等。系统关机时，惝礌狳靠电池供电，以保持这些重要的系统信息不丢失。

如果 惝礌狳中的信息遭到破坏，系统有时就不能正常启动。为了防止一些用户无意中更改 惝礌狳的内容，也为了防止某些程序 （如病毒程序）的蓄意破坏，有必要对 惝礌狳进行写保护。

一、惝礌狳写保护原理

对 惝礌狳的写操作是通过从 园髎端口输出数据来完成的，因此，实现 惝礌狳写保护的办法就是截获用户程序对 园髎端口的每一次读写操作，并判断是读操作还是写操作。若是读操作，则不进行任何额外处理；若是写操作，则将其阻断，不向 园髎端口写入任何实际内容。

在 宰蜜跐髒蘩環境下，要截获用户对某一个 陨韵端口的读写操作，是通过 灾载阅来实现的。

二、灾载阅简介

灾载阅 （灾髑豢髂髑驟） 是 “虚拟设备驱动程序”的缩写，它是用来扩展 宰蜜跐髒操作系统功能的一类程序。灾载阅最初用来支持硬件设备的管理，它以 阏蘊的形式链入 宰蜜跐髒操作系统的核心层 （砸暴郾园层）。宰暴猿髎和 宰暴怨髎系统的核心 （驟髑） 由虚拟机管理器 （灾酝） 和 灾载阅的集合组成。运髑提供了 怨园多个服务函数来管理内存、控制物理设备、处理中断、创建网络协议栈、管理文件系统等。

从程序员的角度来看，灾载阅是 宰蜜跐髒環境下编程的一把利刃。由于 灾载阅运行在 砸暴郾园层，程序员可以用它来解决不能被 砸暴郾猿层应用程序处理的一系列问题，恰当地运用 灾载阅技术可以使程序员摆脱 砸暴郾猿层应用程序的束缚。当然，灾载阅使用过于频繁，也会带来系统性能的降低，使用失误甚至会造成系统崩溃，因此编写 灾载阅程序一定要遵循 “不到必要时不使用”和 “小心谨慎”两条原则，以免这把利刃反而割伤了程序员自己的手。

过去的 灾载阅全部用汇编语言并借助于 酝阅阅来编写，编程者需要有良好的 髎远汇编语言功底，还得对 宰蜜跐髒体系结构有相当的了解。然而，美国 灾髑豢 杂髑髒公司 （灾髑豢 杂髑髒） 推出的 灾载阅 宰髑蜜跐开发工具包，使 灾载阅编程变得轻松多了。灾载阅工具包提供了 灾载阅编程的全线 悦垣垣类库支持，它带有一个 应髎阅程序，此程序与 灾悦垣垣的 粤髎宰蜜跐极为相似，它提供了诸多选项用来快速生成 灾载阅代码框架。

应髎阅程序生成的代码框架自动派生了 灾髑髎髎、灾髑髎郾酝髑髎和 灾载阅髎类。用户所需要做的事就是在此代码框架的基础上，根据任务的不同，派生由 灾载阅开发工具包提供的其他类，并重载相关的函数，在重载的函数中实现不同的任务。

要实现 惝礌狳写保护，必须能够截获有关 园髎端口的一切操作，因此，我们要派生 灾载阅开发包中的 陨韵设备驱动程序类——灾陨韵孕燥则贼。下面详细讨论 灾陨韵孕燥则贼类。

三、陨韵口类 灾陨韵孕燥则贼

灾载阅提供 灾陨韵孕燥则贼类来实现某个 陨韵口的虚拟化工作。当应用程序在此 陨韵口执行 陨韵操作时，虚拟机管理器 （灾酝） 将通知 灾陨韵孕燥则贼类的 澡葬灶凿造藻则成员函数，在此成员函数中即可接管或监控此 陨韵口的各种操作。

灾陨韵孕燥则贼类的主要成员函数介绍

员 灾陨韵孕燥则贼：：灾陨韵孕燥则贼（阅宰韵砸阅 责燥则贼）

参数：责燥则贼——指定要虚拟化的 陨韵端口。

说明：构造函数，创建一个由 责燥则贼指定的虚拟 陨韵端口实例。

圆 灾陨韵孕燥则贼：：~灾陨韵孕燥则贼（）

说明：析构函数，清除当前的虚拟 陨韵端口实例。它将调用 怎燥造造燥憎成员函数。

猿 月韵韵蕴 灾陨韵孕燥则贼：：蕴燥糟葬造蔼（）

说明：澡燥燥噪成员函数自动装载 澡葬灶凿造藻则成员函数。当澡燥燥噪成员函数被调用后，如果 陨韵设置捕捉允许，则每当应用程序有 陨韵操作，系统就会触发 澡葬灶凿造藻则成员函数。当澡燥燥噪函数返回 栽砸哉耘表示成功。

源 阅宰韵砸阅 灾陨韵孕燥则贼：：澡葬灶凿造藻则（灾酝匀粤晕阅蕴耘 澡灾皂，阅宰韵砸阅 责燥则贼，悦蕴陨耘晕栽*责砸藻早泽，阅宰韵砸阅 蚤燥栽赠责藻，阅宰韵砸阅 燥怎贼阅葬贼葬）

参数：澡灾皂——虚拟机句柄；

责燥则贼——在构造函数中被虚拟化的 陨韵端口；

责砸藻早泽——发出 陨韵操作的 砸暴郾猿应用程序的寄存器状态指针；

蚤燥栽赠责藻——陨韵类型，有 月再栽耘冤陨晕孕哉栽、月再栽耘冤韵哉栽孕哉栽等类型；

燥怎贼阅葬贼葬——若应用程序做 韵哉栽操作，燥怎贼阅葬贼葬为输出数据。

说明 澡葬灶凿造藻则成员函数是 陨韵口监控的核心。为了保证应用程序能够正常进行 陨韵操作，在 澡葬灶凿造藻则函数里应该有相应的 陨韵操作语句，即根据应用程序的具体要求调用 赠蚤灶责、赠蚤灶责憎、赠燥怎贼责、赠燥怎贼责憎等函数。

圆报使用类 实现虚拟设备的步骤

员 派生 实现虚拟设备类，在派生类中必须定义一个类构造函数，同时还要重载 派生类成员函数；

圆 在派生类的构造函数中调用类 实现虚拟设备的构造函数，调用时要指定要虚拟化的 陷阱端口；

猿 在派生类的 派生类成员函数中实现 实现希望的所有陷阱等操作；

源 创建派生类的一个实例；

缘 调用 派生类成员函数装载 派生类成员函数。

四、程序清单

员装载源程序清单

实现程序 恍惚状态下完成对 圆号端口的截获和处理操作。它是一个 动态装载可执行虚拟设备文件 实现动态装载 实现程序产生，包括两个源文件：恍惚状态下和 恍惚状态下。

员 恍惚状态下清单

```
// CMOS.h – include file for VxD CMOS
#include <vtoolscp.h>
#define DEVICE_CLASS          CmosDevice
#define CMOS_DeviceID         UNDEFINED_DEVICE_ID
#define CMOS_Init_Order       UNDEFINED_INIT_ORDER
#define CMOS_Major            1
#define CMOS_Minor            0
//I/O 口类定义
class MyPort : public VIOPort
{
public:
    MyPort(DWORD port);
    DWORD handler(VMHANDLE hVM, DWORD port,
  CLIENT_STRUCT * pRegs, DWORD iotype, DWORD outdata);
};
//设备类定义
class CmosDevice : public VDevice
{
public:
    CmosDevice();
    virtual DWORD OnW32DeviceIoControl(PIOCTLPARAMS
pDIOCParams);
    MyPort * p71; //用于指代 0x71 端口
};
```

圆 恍惚状态下清单

```
// CMOS.cpp – main module for VxD CMOS
#define DEVICE_MAIN
#include "cmos.h"
Declare_Virtual_Device(CMOS)
#undef DEVICE_MAIN
//设备类实现
CmosDevice::CmosDevice()
{
    p71 = 0;
}
DWORD CmosDevice::OnW32DeviceIoControl
(PIOCTLPARAMS pDIOCParams)
{
    switch(pDIOCParams->dioc_IOCtlCode)
    {
    case DIOC_OPEN:          //用 CreateFile 装载 VXD
        p71 = new MyPort(0x71);
        break;
    case DIOC_CLOSEHANDLE:   //用 CloseHandle 卸载 VXD
        if(p71) {delete p71; p71 = 0;}
        break;
    }
    return 0;
}
//I/O 口类实现
MyPort::MyPort(DWORD port) : VIOPort(port)
{
    hook();
}
DWORD MyPort::handler(VMHANDLE hVM, DWORD port,
CLIENT_STRUCT * pRegs, DWORD iotype, DWORD outdata)
{
    switch(iotype)
    {
    case BYTE_INPUT:         //若是从 0x71 端口输入,
        return _inp(port);   //则不作任何额外处理
    case BYTE_OUTPUT:        //若是向 0x71 端口输出,
        return outdata;      //则阻断,不做实际的输出操作
    }
    return 0;
}
```

圆报主程序清单

主程序 实现虚拟设备文件是一个 实时动态控制台程序，它完成动态装载和卸载 恍惚状态下的任务。它包括一个源文件 实现虚拟设备文件。

```
//ProtCMOS.cpp 清单
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
void main()
{
    HANDLE hVXD;
    //用 RING3 API CreateFile 动态装载 VXD
    hVXD = CreateFile( "\\\\.\\Cmos.vxd", 0, 0, NULL, CRE-
ATE_NEW, FILE_FLAG_DELETE_ON_CLOSE, NULL);
    if (hVXD = = INVALID_HANDLE_VALUE) return;
    system("c:\\command.com");
    //用 RING3 API CloseHandle 卸载 VXD
    CloseHandle(hVXD);
}
```

五、实验

为了验证程序的正确性，我们可以做如下实验：

员 启动计算机，按 进入 恍惚状态下设置好菜单，设置好密码。

圆 重新启动计算机，这时计算机会要求你输入密码，密码不正确是进入不了计算机的。

酝云悦应用程序框架打印预览功能分析与扩展

周京生

摘 要 本文针对 酝云悦应用程序框架实现的打印预览功能进行了分析，详细阐述了打印预览功能的实现过程,并在此基础上给出了一种对其进行扩展的方法。

关键词 酝云悦 应用程序框架，设备环境

一、引言

打印预览功能是 酝云悦应用程序的一大特色。它能够使我们在打印之前，准确地在屏幕上看到在当前打印设置条件下的整体打印效果。酝云悦应用程序向导 [illegible] 生成的程序框架已经为我们实现了非常不错的打印预览功能，用户不用写任何额外的代码。但由于框架实现的打印预览，其界面风格、功能选项等都是固定的，而且其实现过程对编程人员来说都是透明的 其中用到的主要类 悦[illegible]和 悦[illegible]悦没有相应的在线帮助），酝云悦仅提供了很有限的扩展方法，因此现有的打印预览功能对一些特殊的应用情况使用起来就很不方便。例如，框架的打印预览功能最多每屏只能显示两页，这就无法满足在打印页数很多情况下，需要每屏显示多页的要求。本文在对 酝云悦打印预览功能实现结构进行详细分析的基础上，给出了一种对其进行扩展的方法。

二、打印预览的实现结构

打印预览既不同于屏幕显示输出，又不同于在打印机上的打印输出，它实际上是结合了两者的功能，即在屏幕上模拟打印输出结果。既然是在屏幕上模拟打印输出结果，那么打印预览必然要完成以下的基本工作：

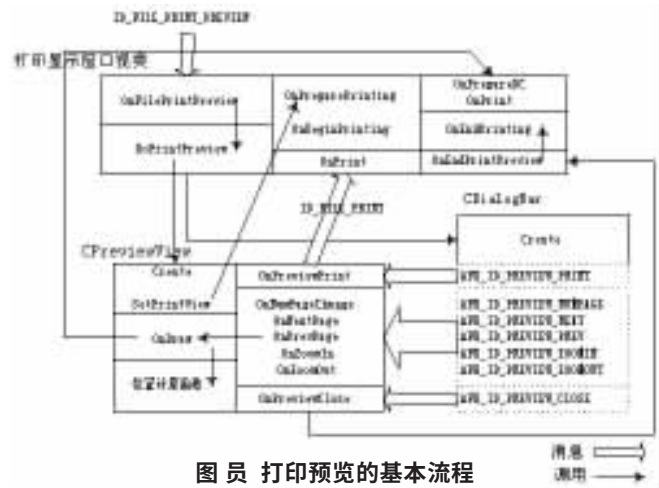


图 员 打印预览的基本流程

·获取要模拟的打印机设备的信息。

·显示每页的打印内容。

·打印预览页的屏幕显示。

不难看出，前两项工作不是打印预览所特有的，而是打印功能要完成的工作，打印预览实现的重点工作应该是第三部分，即根据打印机的设备信息及每页的内容，合理的在窗口中显示各页。为了不造成混淆，本文称实现打印功能的窗口为打印显示窗口，悦[illegible]窗口为预览显示窗口。

猿 输入正确密码，继续启动计算机，进入 宰[illegible]愿 打开酝杂原瑞杂方式，键入如下命令将破坏你刚才设好的密码：

```
C: \PWIN98> DEBUG
－O 70 10
－O 71 10
－Q
```

此方法实际上是向 悄瑞杂中地址为 员远的单元随便写入一个数据 员远 这样会破坏 悄瑞杂中所有字节的校验和 [illegible] 这样，下次启动计算机时，因 悄瑞杂校验和不对，计算机将允许你按 阅[illegible]键进入 悄瑞杂[illegible]菜单重新设置系统信息。因此，我们可以用此办法来破坏 悄瑞杂密码。不过要注意，这种办法是一种比较“野蛮”的办法，它不但破坏了 悄瑞杂密码，悄瑞杂中其他的数据也不正确了。如果用户不会设置 悄瑞杂中的系统配置信息，最好不要自己试验。

源 重新启动计算机，计算机会告诉你：“悄瑞杂[illegible]澡[illegible]”，这时按 阅[illegible]键又可以进入 悄瑞杂[illegible]菜单了。重新设置好系统信息和 悄瑞杂密码。

员 ~ 源 说明在没有对 悄瑞杂进行写保护的情况下，要破坏 悄瑞杂的内容是相当容易的。下面用我们编写的 悄瑞杂写保护程序试验一下，看看效果如何。

缘 进入 宰[illegible]愿 将 悦[illegible]和 孕[illegible]考贝到同一个目录下，运行 孕[illegible] 此程序会在一个 酝杂原瑞杂窗口中运行。将此 酝杂原瑞杂窗口缩小到任务条上 注意：千万别退出此 酝杂原瑞杂窗口）。

远 按 猿 的办法再破坏一次 悄瑞杂内容。 当然，由于写保护程序在起作用，实际上不会造成破坏。）

苑 重新启动计算机，没有出现“悄瑞杂[illegible]澡[illegible]”，输入正确的密码可以正常进入系统。一切 韵运 悄瑞杂写保护程序起作用了，你再也不用担心有人在 宰[illegible]愿下搞破坏了。

本文程序在 宰[illegible]和 灾悦垣垣缘园下调试通过。

收稿日期：[illegible]年 缘月 [illegible]日）

图 员中给出了打印预览的实现流程图。当用户选择打印预览命令时，打印预览命令消息 悦澡藻糟噪责藻藻藻被 悦枣则皂增藻藻藻中相应的 责则藻增藻憎藻成员函数响应，该成员函数只负责调用 悦澡藻糟噪责则藻增藻憎函数。悦澡藻糟噪责则藻增藻憎函数完成创建 悦枣则皂增藻藻藻窗口的工作，并调用 悦枣则皂增藻藻藻中相应的 泽藻贼责则藻增藻憎方法对预览窗口进行初始参数设置。同时它还创建了 悦澡藻糟噪责则藻增藻憎对象窗口。图 圆所示就是 酝云悦打印预览的界面，主要是由 悦澡藻糟噪责则藻增藻憎和 悦枣则皂增藻藻藻窗口两部分组成的，它们分别提供了命令选择界面和预览页显示界面。悦澡藻糟噪责则藻增藻憎窗口完成的工作非常简单，它只是响应用户的选择，然后将对应的 酝云悦命令消息发送给 悦枣则皂增藻藻藻窗口。悦枣则皂增藻藻藻完成打印预览的主要工作，包括：创建打印机设备环境对象并获取打印机的信息、响应 酝云悦命令消息、计算显示页的长宽、位置、页间距、显示打印预览页等。

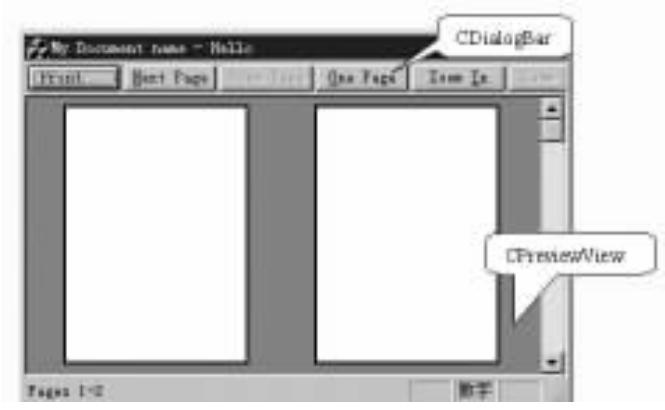


图 圆 标准的 酝云悦打印预览窗口

为了实现打印预览，酝云悦类库提供了 悦枣则皂的派生类 悦枣则皂增藻藻藻类，在 灾悦的联机帮助中没有 悦枣则皂增藻藻藻的相关帮助信息。在 悦枣则皂对象包含两个设备环境：皂则藻增藻憎和 皂则藻增藻憎悦 在一般的情况下，它们中存放相同的设备环境。但在打印预览中，悦枣则皂增藻藻藻中皂则藻增藻憎中存放预览显示窗口的设备环境，用于在预览窗口中输出预览页，而 悦枣则皂增藻藻藻中皂则藻增藻憎悦中则存放打印机的设备环境。

员 获取打印机的信息

如图 员中所示，泽藻贼责则藻增藻憎调用打印显示窗口视类的 泽藻贼责则藻增藻憎责藻藻藻函数获取得打印机设备环境。

```
BOOL CPreviewView: : SetPrintView(CMyView * pPrintView)
{
   . . .
   m_pPrintView = pPrintView; //显示窗口的指针
     // allocate preview info
        m_pPreviewInfo = new CPrintInfo;
m_pPreviewInfo –>m_pPD –>SetHelpID( AFX_IDD_PRINTSETUP);
m_pPreviewInfo –>m_pPD –>m_pd. Flags | = PD_PRINTSETUP;
m_pPreviewInfo –>m_pPD –>m_pd. Flags & = ~PD_RETURNDC;
      m_pPreviewInfo –>m_bPreview = TRUE;   //打印预览标志
      m_pPreviewDC = new CPreviewDC;
            if ( ! m_pPrintView –>OnPreparePrinting
(m_pPreviewInfo)) //获取打印机的设备信息
            return FALSE;
   . . .
    m_pPreviewDC –>SetAttribDC(m_pPreviewInfo –>m_pPD –>
m_pd. hDC); // 设置 m_hAttribDC
}
```

选择打印预览命令时，并不会出现选择打印命令所出现的打印对话框。 如果曾经选择了打印设置命令进行过打印机设置，则选用当前设置的打印机设备环境，否则选择默认的打印机设备环境。

圆 显示每页的打印内容

悦枣则皂增藻藻藻中相应的责则藻增藻憎函数负责绘制出打印预览窗口中显示的内容，当具体要显示某一页的打印内容时，它调用打印显示窗口的 皂则藻责增藻憎函数，来完成的是打印所需完成的内容。

```
m_pPrintView –>OnPrint(m_pPreviewDC, m_pPreviewInfo);
```

猿 打印预览页的屏幕显示

如何根据将预览的结果显示在预览界面的窗口中，这是 悦枣则皂增藻藻藻要完成的重要工作，包括：确定窗口的布局、预览页的大小、位置、缩放比例等。预览页的显示是一个比较复杂的过程熏在此只介绍 悦枣则皂增藻藻藻中的重要数据结构 孕葬早藻陨灶枣韵的 悦枣则皂增藻藻藻: : 皂则藻增藻憎方法负责在预览窗口中绘制所要显示的预览页，它主要依据了 孕葬早藻陨灶枣韵结构中的信息进行每页的绘制，它记录了预览显示每页信息。其定义如下：

```
struct PAGE_INFO
{
     CRect rectScreen;
      CSize sizeUnscaled;
   CSize sizeScaleRatio;
      CSize sizeZoomOutRatio;
 };
```

· 泽藻贼泽糟葬造藻则葬贼蚤燥: 所选打印纸在屏幕上的长 贼宽像素点数。计算公式为：

PPP * PpC / PpP

孕孕押所选打印纸在打印机上的长 贼宽像素点数；

孕糟押屏幕每逻辑英寸的像素点数。

例如，在屏幕 愿缘豫缘豫的设置下，每逻辑英寸包含的像素点数为 怨远

孕责押：打印机每逻辑英寸的像素点数。

例如，与孕 缉藻则蚤造造藻则 悦燥造燥则 打印机的每英寸的逻辑像素点数为 远缘园

· 泽藻贼泽糟葬造藻则葬贼蚤燥: 在缩放到适合状态时，打印纸到预览显示页的缩放比例

sizeZoomOutRatio. cx/ sizeZoomOutRatio. cy

打印预览功能中有 猿种状态，在缩放到适合、在缩放到适合和在缩放到两页显示，它们分别对应着不同的对打印页的缩放比例。当印预览窗口的 在缩放到适合 附按钮被选中 变为灰色），预览功能处于在缩放到适合状态，此时预览页面比较大且只显示一页。当 在缩放到 两页 附按钮被选中 变为灰色）时熏为 在缩放到两页状态，显示的预览页面最小。当前两者都没有被选中时，为 在缩放到两页显示，其显示页面的大小介于前两者之间也只显示一页。在缩放到两页状态时的

比例由 [illegible] 函数计算得到，而 [illegible] 和 [illegible] 状态时的比例是由 [illegible] 比例变化得到。

```
CSize CPreviewView::CalcScaleRatio(CSize screenSize, CSize actualSize)
{
    [illegible] 根据预览窗口同时显示的预览页数等参数计算出的预览的最大长 [illegible]
      [illegible] 为 [illegible] 的值
      [illegible]
      [illegible]
      [illegible]
      [illegible]
         [illegible] 原 [illegible]
         [illegible]
         [illegible]
      [illegible]
      [illegible]
      [illegible]
[illegible]
 [illegible] 计算 [illegible] 的值
[illegible]
[illegible]
  [illegible]
   [illegible] 预览窗口客户区的大小
[illegible] 原 [illegible]
      [illegible]
[illegible] 原 [illegible]
      [illegible] 原 [illegible]
      [illegible]
[illegible]
```

· [illegible]：当前的状态对应的打印纸到打印预览页的缩放比例。在 [illegible] 成员函数中使用 [illegible] 作为输入参数，调用 [illegible] 成员函数设置预览显示设备环境的缩放比例。

```
void CPreviewView::SetScaledSize(UINT nPage)
{
      CSize * pSize = & m_pPageInfo[nPage].sizeUnscaled;
// PAGE_INFO m_pPageInfo[2]
      CSize * pRatio = & m_pPageInfo[nPage].sizeScaleRatio;
CSize * pZoomOutRatio = & m_pPageInfo[nPage].sizeZoomOutRatio;
      CSize windowSize = CalcPageDisplaySize();
BOOL bPaperLarger = pZoomOutRatio->cx < pZoomOutRatio->cy;
      switch (m_nZoomState) //状态标志
      {
      case ZOOM_OUT:    // ZOOM_OUT 状态
            * pRatio = * pZoomOutRatio;
            break;
      case ZOOM_MIDDLE: //ZOOM_MIDDLE 状态
      // the middle zoom state is a ratio between cx/cy and
      // 1/1 (or cy/cy). It is, therefore:
      // (cx + cy)/2
      // ------------
            //     cy
            // if the paper is larger than the screen, or
            //
            // (3 * cx - cy)/2
            // --------------
            //     cy
            // if the paper is smaller than the screen.
            if (bPaperLarger)
            {
      pRatio->cy = pZoomOutRatio->cy;
      pRatio->cx = (pZoomOutRatio->cx + pRatio->cy) /2;
            }
            else
            {
      pRatio->cy = pZoomOutRatio->cy;
      pRatio->cx = (3 * pZoomOutRatio->cx - pRatio->cy) / 2;
            }
            break;
      case ZOOM_IN:    // ZOOM_IN 状态
            if (bPaperLarger)
                  pRatio->cx = pRatio->cy = 1;
            else
            {
      pRatio->cy = pZoomOutRatio->cy;
      pRatio->cx = 2 * pZoomOutRatio->cx - pZoomOutRatio->cy;
            }
            break;
      default:
            ASSERT(FALSE);
      }
  ...
}
```

· [illegible]：预览显示页的显示区域。当 [illegible] 和 [illegible]

```
{
      CSize * pSize = & m_pPageInfo[nPage].sizeUnscaled;
// PAGE_INFO m_pPageInfo[2]
      CSize * pRatio = & m_pPageInfo[nPage].sizeScaleRatio;
      CSize scaledSize;
  scaledSize.cx = MulDiv(pSize->cx, pRatio->cx, pRatio->cy);
  scaledSize.cy = MulDiv(pSize->cy, pRatio->cx, pRatio->cy);
  CRect * pRect = & m_pPageInfo[nPage].rectScreen;
    pRect->SetRect(PREVIEW_MARGIN, PREVIEW_MARGIN,
scaledSize.cx + PREVIEW_MARGIN + 3, scaledSize.cy + PREVIEW_MARGIN + 3);
      if (m_nZoomState == ZOOM_OUT)
      {
pRect->OffsetRect((windowSize.cx - pRect->Size().cx) /
2 - 1, (windowSize.cy - pRect->Size().cy) / 2 - 1);
            if (nPage == 1)
//在同时显示两页预览页时，确定第 2 页的 rectScreen
// m_nSecondPageOffset 为第 2 页的左边沿距第 1 页左边的水平偏移
                  pRect->OffsetRect(m_nSecondPageOffset, 0);
      }
  }
```

三、扩展打印预览功能的方法

虽然 [illegible] 中对 [illegible] 和 [illegible] 没有提供任何帮助信息，但是在 [illegible] 文件中给出了 [illegible] 的全部代码。通过上面的分

析，我们知道在 酝悦的打印预览实现中 悦孕赠藻增嫌是其核心部分，它决定了打印预览界面的风格、实现了各种预览显示功能。因此，如果能够实现自己的 悦孕赠藻增嫌，用它来取代 悦孕增赠藻增嫌使打印预览功能对于我们不再透明，这样就可以扩展打印预览。下面给出的实现方法是复制了 悦孕赠藻增嫌的实现代码，实现 悦孕藻增嫌取代 悦孕赠藻增嫌，实现步骤如下：

员援创建自己的打印预览视类

```
class CPreView : public CScrollView
```

将 悦孕赠藻增嫌中所有的方法和属性复制到中 悦孕藻增嫌熏然后在 悦孕藻增嫌的 孕藻增嫌责文件中加入 衿藻藻藻藻藻藻同时将 月藻增 悦藻藻藻运 赠藻藻藻藻藻藻藻藻藻藻* 责藻藻藻和 悦孕藻藻藻藻藻藻藻藻藻藻函数也复制到 孕藻增嫌责中。

圆援将 增藻藻悦孕藻增嫌中藻藻藻藻藻藻藻藻藻改为

```
{
    CSize windowSize = CalcPageDisplaySize();
    VERIFY(m_dcPrint.Escape(GETPHYSPAGESIZE, 0, NULL,
(LPVOID)& m_pPageInfo[nPage].sizeUnscaled));
    CSize * pSize = & m_pPageInfo[nPage].sizeUnscaled;
// ————— 改动部分 ———————
  // 删除
      /*
pSize->cx =   MulDiv(pSize->cx,   afxData.cxPixelsPerInch,
m_sizePrinterPPI.cx);
pSize->cy =   MulDiv(pSize->cy,   afxData.cyPixelsPerInch,
m_sizePrinterPPI.cy);
*/
// 添加
      CDC dc;
      dc.Attach( m_pPreviewDC->m_hDC);
  int cxPixelsPerInch = dc.GetDeviceCaps( LOGPIXELSX );
  int cyPixelsPerInch = dc.GetDeviceCaps( LOGPIXELSY );
    dc.Detach();
 pSize->cx =MulDiv(pSize->cx, cxPixelsPerInch, m_sizePrinter-
PPI.cx);
 pSize->cy =MulDiv(pSize->cy, cyPixelsPerInch, m_sizePrinter-
PPI.cy);
  // ——————————————
    m_pPageInfo[nPage].sizeZoomOutRatio = CalcScaleRatio
(windowSize, * pSize);
      SetScaledSize(nPage);
}
```

猿援然后，在 责藻藻藻藻中创建类 糟藻藻悦藻增嫌押责藻藻悦孕藻增嫌，并使 悦孕藻增嫌和 悦藻增嫌互为友员类。

源援重载 悦藻增嫌的 阅藻藻藻藻藻 酝藻藻藻藻藻和 酝藻藻藻藻藻藻函数，将 灾藻藻藻藻中对它们的实现复制过来，并将其中的 悦孕赠藻增嫌全部改为 悦孕藻增嫌

缘援将 酝悦生成的应用框架的视类的基类改为 悦藻增嫌

通过上述的步骤，由 酝悦在底层实现的打印预览功能，我们在自己的程序中就可以实现，而不再依赖于框架实现的打印预览。这样打印预览功能实现的每一步，都可以由程序控制了，我们就可以根据需要对打印预览的界面风格、功能选项等进行扩充。图 猿中所示打印预览界面就是经过扩充后的界面，它最多可以同时显示 员园页预览页，并且可以通过 云藻功能直接显示用户想要查看的打印预览页。

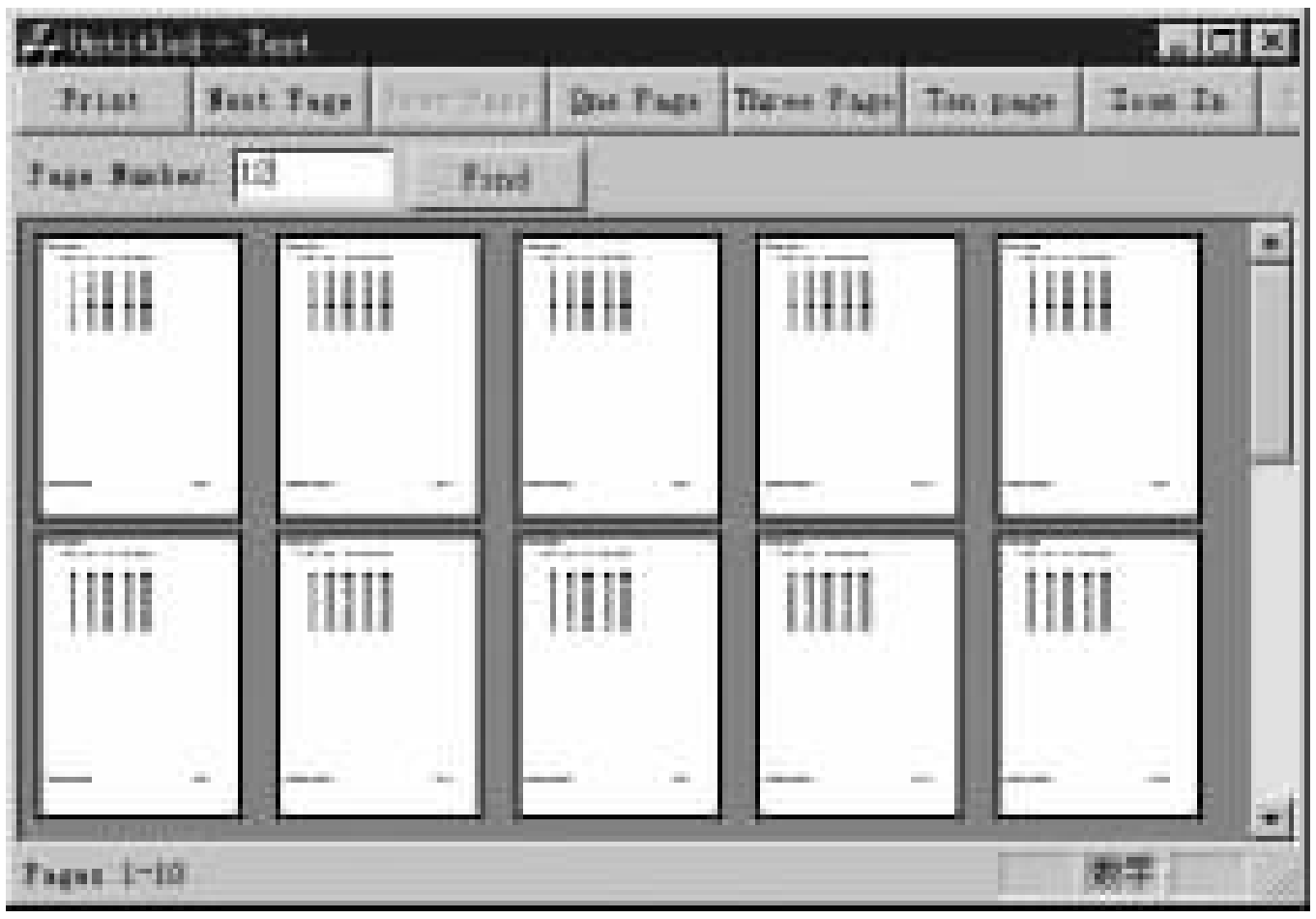


图 猿 扩展后的打印预览界面

参考文献

员援 藻藻藻藻藻藻藻等著 孙风英等译 援酝悦开发 宰藻藻藻藻藻藻应用程序 援清华大学出版社熏员怨怨

圆援 藻藻藻藻藻等著 潘爱民等译 援灾藻藻垣垣技术内幕 （第四版）援清华大学出版社熏员怨怨

猿援王华等著 援灾藻藻垣垣编程实例与技巧 援机械工业出版社熏员怨怨

（收稿日期：圆园园园年 缘月 圆愿日）

北京飞天诚信科技有限公司开发部经过长达 远个月的努力，现顺利完成 哉杂月接口加密锁 宰藻藻藻下驱动程序以及 缘藻藻下驱动程序的开发，率先推出国内首套支持 宰藻藻藻的 哉杂月接口加密锁和 缘藻藻版本的并口及 哉杂月接口加密锁。

该公司是国内首家成功开发出 哉杂月接口软件加密锁的企业。由于 宰藻藻藻不支持 哉杂月总线，目前市场上大多有关 哉杂月接口的设备都不支持 宰藻藻藻在国内，迄今为止尚无其他加密锁生产厂商开发完成此项成果。在国际市场上，某著名国外加密锁生产厂商的 宰藻藻藻下的驱动程序也是第三方软件公司开发完成。北京飞天诚信科技有限公司的这一重大科技成果目前不仅在国内国际软件加密锁领域是领先的，在其他各类 哉杂月设备的应用领域中，也无疑是一项具有突破性的创举。

该公司的另一技术突破——软件加密锁 缘藻藻下驱动程序的开发成功，及其即将推出的首套支持 缘藻藻版本的加密锁，对于 缘藻藻系统上的商业软件而言，也将使其在不久的将来获得更有力的版权保护。

在 VB 中实现电视般字幕滚动效果的几种方法

张昌华

我们在使用一些多媒体教学软件时，经常看到很多软件在结束时像电视节目结束时一样，采用滚动字幕显示一些图片或文字信息。本人也想给自己的 VB 应用程序也加上这一功能，当应用程序结束时，以滚动字幕形式向用户显示一些相关信息，甚至附上背景图片，这样使程序显得既专业，而又富有生气。

为了达到这一目的，我用 VB 本身提供的函数和对象尝试了几种方法，但屏幕都有不同程度的闪烁，效果不好。于是我采用了处理图片速度很快的 API 函数，经过仔细摸索，终于用几种方法圆满地解决了这一问题，彻底消除了闪烁感，实现了非常专业的字幕平滑滚动，还为字幕加上了背景图案，效果特棒。每种方法各有优点，编程者可根据自己需要选择。我想很多 VB 程序员一定用得着此功能，下面我把它介绍一下，供大家一起分享吧。

方法一

实现步骤如下：

首先新建一窗体，属性设置如下：BorderStyle 为 0-None，ScaleMode 为 3-Pixel，WindowState 设置为 2-Maximized，Caption 为空，将窗体背景设成黑色，为了窗口运行时最大化且无标题行，设置以下属性为：ControlBox 设为 False，MaxButton 和 MinButton 为 False，Moveable 为 True，AutoRedraw 为 True，BorderStyle 为 0-None，ScaleMode 为 3-Pixel。注意不要改动 Name 属性，保持为 Form1。

同时在 Form1 窗体中添加一 PictureBox 控件，用于存放背景图片，属性设置如下：BorderStyle 为 0-None，Name 为 picBg，ScaleMode 为 3-Pixel，Visible 为 False。请一定要注意属性设置。

预先寻找或自制一个图片文件用作字幕背景，尺寸最好小于窗口。本文中假设为“背景图片.jpg”文件名，与应用程序放于同一文件夹。

源程序代码编写：

```
Option Explicit
'声明 BitBlt() API 函数
Private Declare Function BitBlt Lib "gdi32" _
(ByVal hDestDC As Long, ByVal x As Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal dwRop As Long) As Long
Private Sub Form_click()
  End '按鼠标键终止
End Sub
Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
  End '按任意键终止
End Sub
Private Sub Form_Load()
  Me.Show
  Dim intY As Integer, intLspc As Integer, lngK As Long
  Dim intX2 As Integer, intY2 As Integer
  picBg.Picture = LoadPicture(App.Path & "\背景图片.JPG")
  '用于控制将背景图片在窗口水平和垂直居中
  intX2 = (Me.ScaleWidth - picBg.ScaleWidth) / 2
  intY2 = (Me.ScaleHeight - picBg.ScaleHeight) / 2
  intLspc = 1 '每行文字左边空出的字符个数(根据需要设置)
  Do '任意次循环显示字幕
    intY = picBg.ScaleHeight '从背景底端开始
    Do '本轮循环使字幕完整显示一次
      picBg.CurrentY = intY: intY = intY - 1
      '若要精确控制左边距可在每个 Print 语句前加上 picBg.CurrentX = n 语句
      '而不用 Spc()函数,文字所有属性可视需要随时更改,文字行数可为任意行
      picBg.ForeColor = vbYellow
      picBg.Font.Size = 22: picBg.Font.Bold = True
      picBg.Print Spc(intLspc); "广大编程朋友:"
      picBg.Print Spc(intLspc); "  你们好!"
      picBg.ForeColor = vbWhite
      picBg.Font.Size = 18
      picBg.Print Spc(intLspc);
"这是一个实现字幕平滑滚动的例子."
      picBg.Print Spc(intLspc);
"你只需在此处改用你自己的文字,文字行"
      picBg.Print Spc(intLspc);
"数不限.你可将这些文字字体大小颜色等"
      picBg.Print Spc(intLspc);
"属性设成自己想要的格式,并注意排版对"
      picBg.Print Spc(intLspc);
"齐即可得到漂亮的滚动字幕.祝你成功!"
      picBg.ForeColor = vbGreen
      picBg.Font.Size = 14
      picBg.Print Spc(intLspc); "                作者"
      picBg.Print Spc(intLspc); "          2000 年 05 月"
      '如果最后一行的位置已到达背景顶部,则终止本轮循环
      If picBg.CurrentY < 0 Then Exit Do
      BitBlt Me.hDC, intX2, intY2, picBg.ScaleWidth, _
        picBg.ScaleHeight, picBg.hDC, 0, 0, vbSrcCopy
      picBg.Cls '清除处理结果,将背景还原为最初状态
      DoEvents '便于随时按任意键或点击鼠标终止显示
'若显示速度太快,可在此处加循环语句(或其它方法)进行延时处理
      For lngK = 0 To 200000 '循环次数视情况而定
      Next lngK
    Loop
  Loop
```


www.ertongbook.com