

E&C

1993

一九九三年 ● 总期第104期

11

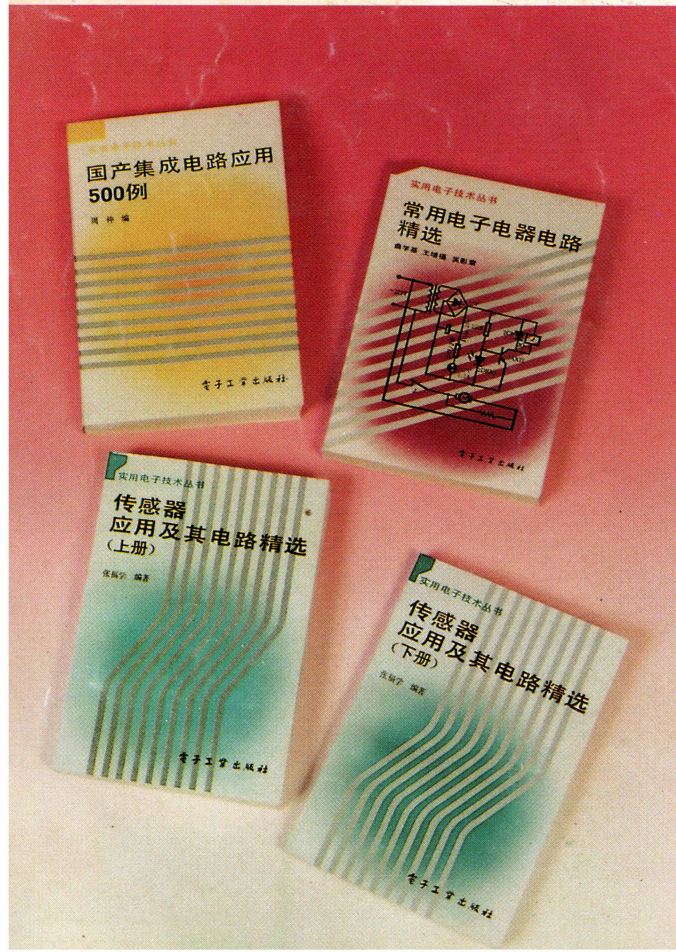
電子

ISSN 1000 1077

與電腦



ELECTRONICS AND COMPUTERS



衷 心 奉 献

●DP851

(DP851 单片机普及板)是一种模块式结构的 51 系列单片机系统,它有三种配置方式以实现不同类型的需要。

1. 开发系统板:主板+键盘板(或主板+PC机)可构成单板型单片开发器。主板上 有 16KB 的监控程序(键盘和 PC 机各占 8KB),具有置入、修改、调试、运行等多种开发功能。

2. 教学实验板:开发器+实验板可构成教学实验板、该板从教学的实际需要出发,具备了实验内容多、覆盖范围宽、结构灵活、性能可靠、价格低廉等优点,是一种大众型的教学实验设备。

3. 目标应用板:开发器+A/D、D/A 等扩展板可构成用户的目标系统(详细情况陆续介绍)。

●BJ10 系列

适用于仪器仪表智能化,数据采集,可编程控制等项目的研制开发及应用性计算机产品中的做为嵌入式主机板使用(详细情况陆续介绍)。

●BSY 系列

BSY01—07 分别具有 I/O 扩展;键盘及显示控制;A/D、D/A 变换等多种功能。用户可根据不同用途自行选择组合(详细情况陆续介绍)。

●TDS—3000 系列

TDS—3000 系列数字显示控制仪表:本系列仪表采用微机技术,实现电热函数的转换,彻底介绍了温度传感器的热电非线性问题,在整个测量范围内不存在非线性误差。微机定时控制的自动稳零系统,保证零点不会漂移。

本系列仪表还能配各种标准电压,标准电流输入,可与任何线性的传感器、变送器配套。还具有断偶保护及报警灯光显示,采样灯光显示等功能。

●SW 系列

本厂数字仪表共分三大类十个方面的一百多个品种,该仪表与不同传感器配合使用,可测量温度、压力、电流、电压等多种物理量,并能实现指示调节和报警。

各种仪表均可按用户要求带有 0—5V, 1—5V, 0—10mA, 4—20mA 等一组标准信号输出。

本仪表测量精度高,显示直观,抗震性能好,使用寿命长(详细情况陆续介绍)。

●为了便利用户,本厂愿为用户邮购各种电子元器件,各类芯片等。

地 址:北京西城区月坛南街 83 号

开户行:工商银行百万庄分理处

邮 编:100045

帐 号:014—047014—96

电 话:8512947

联系人:张文奇 庞文姬

电子工业出版社新书消息

书 名	定 价	作 者
标准集成电路数据手册——运算放大器(精)	32.00	崔忠勤等编
标准集成电路数据手册——运算放大器(平)	27.00	崔忠勤等编
标准集成电路数据手册——PAL电路(精)	61.00	手册编委会
标准集成电路数据手册——PAL电路(平)	56.00	手册编委会
标准集成电路数据手册——TTL电路(精)	40.00	童本敏等编
中外集成电路简明速查手册 TTL,CMOS 电路(精)	56.00	手册编委会
中外集成电路简明速查手册 TTL,CMOS 电路(平)	51.00	手册编委会
标准集成电路数据手册——通信电路(精)	35.00	手册编委会
标准集成电路数据手册——通信电路(平)	30.00	手册编委会
国内外功率晶体管实用手册(上.精)	35.00	本书编写组
国内外功率晶体管实用手册(下.精)	68.00	本书编写组
家用电器元器件手册(精)	56.00	张士炯编
家用电器元器件手册(平)	51.00	张士炯编
标准集成电路数据手册——音响电路(精)	40.00	《手册》编委会
标准集成电路数据手册——音响电路(平)	35.00	手册编委会
电视机用集成电路数据手册	60.00	手册编委会
国内外半导体光电器件实用手册(精)	80.00	光电子、光电器件协会
国内外半导体光电器件实用手册(平)	75.00	光电子、光电器件协会
标准集成电路数据手册——高速 CMOS 电路(精)	45.00	编委会
标准集成电路数据手册——高速 CMOS 电路(平)	40.00	
最新接口电路数据速查手册	16.00(估价)	
最新音响电路数据速查手册	16.00(估价)	
盒式收录机原理与检修技术	17.00	吴运昌等编著
家用电器维修技术基础	12.50	熊耀辉等编著
彩色电视机原理与检修技术	11.50	李运林等编著
黑白电视机原理与检修技术	14.00	全景才等编著
家用录像机原理与检修技术	16.00	文启暄等编著
家用电热电器原理与检修技术	8.50	宋国瑞等编著
家用组合音响原理与检修技术	16.00(估价)	
家用空调电冰箱原理与检修技术	16.00(估价)	
家用洗衣机、电风扇原理与检修技术	12.00(估价)	
如何延长录像机的使用寿命	3.90	马兰皋等编著
最新电子手表原理、使用与维修	2.90	王新廷等编著
万用表测量技巧(精)	17.50	沙占友
万用表测量技巧(平)	15.00	沙占友著
数字万用表的原理使用与维修	4.50	沙占友等编著
电梯技术——原理、维修、管理	9.80	史信芳等编
英汉信息技术标准语词典	18.50	林宁等主编
计算机应用指南	15.00	本书编委会
常用电子电器电路精选	9.50	曲学基等
国产集成电路应用 500 例	13.50	
传感器应用及其电路精选(上)	13.00	张福学编著
传感器应用及其电路精选(下)	9.00	张福学编著
实用光电控制电路精选	15.50	陈尔绍
优质电子元器件实用手册(上册)	25.00	
优质电子元器件实用手册(下册)	25.00	

购书请到北京万寿路电子工业出版社发行部邮购科 邮寄附加 15% 邮寄费 电话:8233693 开户行:北京市工商行
翠微路分理处 帐 号:661036-40 户 名:电子工业出版社发行部 邮编:100036

电子与电脑

一九九三年
总期第 104 期

目 录

- 综述 •
 - PC 机加解密软件及技术 陈 伟(2)
- PC 用户 •
 - 解决 TSR 程序一次性驻留内存的根本方法 宋立波(7)
 - FoxBASE 下一个新型的通用菜单控制程序 周丝溪 胡武海(11)
 - 在批处理文件中巧用 DOS 环境变量 黄庆程(14)
 - DOS 文件的 FCB 管理机制分析 崔来堂(16)
 - CMOS 中参数的保存及重置方法 金林樵(19)
 - 获得西山 DOS 五笔字型词组 丁 梅(20)
 - 最小汉字笔画库的设计和使用 黄焕如(21)
 - 背景音乐的原理及其实现 丁 塔(23)
 - C—WordStar 使用基础问与答 朱大公(24)
 - 计算机病毒的检测、消除和预防 苏民生(28)
- 学习机之友 •
 - CEC—I 中华学习机汉字系统简介 林永春(30)
 - 多功能标准化学成绩统计程序 汤永进(31)
 - ProDOS 系统内部结构剖析(续) 廖 凯(33)
- 学用单片机 •
 - 单片机软件抗干扰新设计 李凯里(35)
 - 一种简单实用的集散控制系统 王济浩(37)
- 电脑巧开发 •
 - LED 智能显示屏的结构及驱动、显示电路 张 艺(38)
- 维修经验谈 •
 - 激光印字机的维护与故障排除 崔晨荣(41)
- 电脑游戏机 •
 - 第二章 6527CPU 的显示系统(下) 于 春(42)
- IC 电路应用 •
 - 计算机中的新技术——Flash Memory 张 戟(45)
 - 模拟乘法器 IC 及使用方法(三) 李兰友(48)
 - 经验二则 王万春(49)
- 电脑通信 •
 - 微机通信网络的发展状况及趋向 薛兴华(50)
 - 如何升级你的网络工作站 薛 强(54)
- 读者联谊 •
 - 修改 BIOS 程序实现对 PC 机的加密 徐文军 李晓中(54)
 - 修改软金山系统,适应有缺陷的 386 主板 文 森(56)
 - CCED 使用次数的修改 朱 融(56)

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部
(北京 173 信箱 邮政编码:100036)

印刷:北京三二〇九厂

国内总发行:北京报刊发行局

国内统一刊号:CN11—2199

邮发代号:2—888

国外代号:M924

出版日期:每月 23 日

主编:王惠民 特约编审:苏子栋

责任编辑:张 丽

订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:1.60 元

PC 机加解密软件及技术

合肥市统计局计算站(230001) 陈 伟

加密与解密作为两门相克的技术生存至今,并得到不断的发展,就它们本身来讲都是有原因的。从加密者的角度来看,他们要设计出一种比较完善和安全的加密技术,来保护自身的经济利益,但作为解密者,他们则想“无偿”的得到软件的复制品,这样就导致这两种技术的相互促进与共同发展。

一、加密技术基础

磁盘加密技术分为三大方面:反拷贝技术、反跟踪技术和密文技术。对于一个好的加密系统来说,这三个方面都需要恰当运用。

反拷贝技术从本质上说就是在磁盘上做特殊标记的技术,通常我们把这类特殊标记统称为指纹。就现今的加密技术来说,一个成功的反拷贝技术在磁盘上做出的指纹是无法通过软磁盘控制器来复制的,并且做出的指纹是各不相同的(唯一的)。

反拷贝技术也是整个加密系统中最基本、最重要的技术,它的实现是建立在对 DOS 操作系统和软磁盘控制器深入了解的基础上。这方面的技术最初起源于 Apple II 机,由于 Apple II 的软盘控制器可以用软件方法来控制,因而在当时出现了较多和较好的反拷贝技术。当 PC 机刚刚出现时,由于人们对 PC 机和 PC-DOS 的不甚了解,使得早期的反拷贝技术在 PC 机出现的最初 4 年中没有多大的突破,主要停留在使用异常的 ID 参数上。

1984 年美国的 VAULT 公司推出了 PROLOK 加密系统。由于 PROLOK 加密系统首次成功的把良好的反拷贝技术、反跟踪技术和密文技术完美地结合起来,所以在它推出的近几年内,没有拷贝软件通过任何途径成功的复制了由它所保护的系统。可以说,PROLOK 加密系统是十分成功的,并且从现在的眼光来看 PROLOK 加密系统也不失为是个十分优秀的加密系统。正是由于 PROLOK 系统的推出,使人们充分的认识到一个成功的加密技术,所采用的方法一定是反拷贝、反跟踪和密文技术三方面的完美结合,也正是从 PROLOK 系统推出之日起,真正成熟和完美的 PC 机加密技术出现了。

从产生磁盘上指纹的方法来说,一共有两种:即软加密技术和硬加密技术。

软加密技术是直接通过软盘控制器来产生的,运用这种技术比较成功的有:1. 扇区间隙软指纹技术:它是利用磁头在操作时定位点的随机性偏差来完成的。当磁盘的某个扇区被重写时,磁头的定位点由于机械

偏差的原因,一定与重写数据前是不同的,当扇区中的数据被重写后,作为扇区间的间隔也一定发生变化,而且这些变化是随机性的,当然也就不能被拷贝软件通过软盘控制器来复制这种随机性的信息。2. 磁道接缝软指纹技术:大家知道作为圆形的磁道总有个首尾相接之处,并且由于机械方面的原因,在第 1 扇区的前面和第 9 扇区(360K 软盘)的后面都留有一段空白区域,而在这段空白区域(即磁道的接缝)上的数据是无法被格式化的,并由于软盘控制器转速波动的影响,使得第 9 扇区后置区的长度和内容是随机变化的,即使同一台软驱、同一片软盘、同一条磁道在不同时间上格式化出来的后置区的长度和内容都是不一样的,这种磁道中后置区的长度和内容的随机性,也同样具有了指纹技术中所要求的唯一性和不可复制性。

硬加密技术也分为两类:1. 利用特殊的设备制造出一般软盘控制器根本无法生成的指纹,比如:①采用激光打孔的 PROLOK 加密技术;②利用电磁手段破坏软盘上的编码格式的电磁加密技术;③为了保护软盘光洁度,而在软盘的某一区域上形成一层膜来达到破坏软盘磁性效果的掩膜加密技术;④成本极低的,仿激光加密效果的针孔加密技术等等。2. 加密卡技术:从硬件上产生指纹来保护软件的技术就是加密卡技术,它的成本较高,但安全性好,因为要实现加密卡技术的解密只有两种方法,一是复制加密卡,二是分析加密软件。

这两种加密技术在实际应用中都被广泛运用,并得到了较好的效果。可是由于用户考虑到制作加密系统的成本和设备,往往采用软加密技术,如国内十分流行的 LOCK89 加密软件就是采用了软加密技术。

作为反拷贝技术的核心就是指纹的判读程序。从安全方面来说,判断程序最好与被加密程序混为一体,因为只有这样才能大大加深解密的难度。对于一个用这种加密方法保护的系统来讲,唯一的解密方法就是彻底分析这个系统。这对于解密者来说,工作量是相当大的,而现在的加密系统一般只是把判断程序简单地加在被加密程序的外围,这样很容易被解密者剥去这层“外衣”,来达到解密的效果。当然如果要采用前一种方法必须要求软件开发者和加密者是同一个设计者,这样就有了较大的局限性,但现在的 CCED 4.0、超想自然码都是声称使用了这种技术。

随着加密技术的发展,现在作为反拷贝技术的指纹往往是不可复制的。因此解密者往往用跟踪程序的执行来分析和破解解密程序,这样就迫使加密者不得

不提高加密系统的反跟踪能力和密文技术。所谓反跟踪就是利用各种方法,阻止解密者对程序的分析 and 阅读,以达到增大解密难度,延长加密系统生命周期的目的。密文是将程序中的信息通过某种特定的算法转换而成的信息。

一个加密系统一定包含解密处理的模块,它主要用来识别钥匙盘的真假和使主程序能正常执行。现在的反跟踪技术主要就是为了保护这段模块的安全,它的主要功能有两种:1. 抗静态分析;2. 抗动态跟踪。前者主要是针对一些反汇编工具来说的,它的解决方法只要简单的密文技术足以可以抵挡;后者就相当复杂了,实现它的方法也比较多,灵活性较大,技巧性较强。它主要包括:

1. 破坏中断向量表:从现有的跟踪调试工具来说,大部分都利用了单步中断和断点中断,而这两个中断程序的入口地址分别存放在内存的 0:0004-0:0007H 和 0:000C-0:000FH 中,如果修改这些单元中的内容,就使调试工具无法跟踪加密系统;
2. 关闭显示屏;
3. 锁死键盘:因为用调试工具跟踪分析加密系统,是由解密者从键盘击入一些操作命令来进行的,并且还要从屏幕上显示出的结果来得到下一步的操作步骤。因此在加密系统无须从屏幕或键盘输出、输入信息时,完全可以关闭这些外围设备,来达到破坏跟踪加密软件的目的;
4. 采用逆指令流法:程序是从低地址到高地址逐条指令执行的,如果我们改变这种顺序执行指令的方式,使 CPU 按逆方向执行指令,那么这样的程序可读性将大大降低,而且可以阻止解密者对加密系统的跟踪;
5. 移动堆栈指针:堆栈是用来存放数据的,但它也是用来存放中断断点的,所以跟踪一定涉及到堆栈,因而我们也可采用移动堆栈指针的方法来破坏跟踪。

现在使用的反跟踪技术远远不止以上几种,但它们的的目的都是一样的。不过这些方法在执行时修改的内容一定要在加密系统退出时予以恢复,以免影响其它的操作。

密文技术起源较早。就加密系统而言,密文技术至少要考虑到两个方面:一是密码的时间、空间复杂度;二是密码的抗攻击强度。它主要的方法有:1. 换位法:将明文信息中的字符按照一定的规律进行重排;2. 替代法:用字符或字符块来替换明文信息;3. 乘积密码法:是换位法和替代法的相互结合,即先换位后替代的方法;4. 联邦保密标准(DES);5. 公开密钥密码系统(RSA)。后两种技术是最具代表性的现代密码技术,它们的实现比较复杂,在此不作叙述。

由于采用了反跟踪、密文技术,使得整个系统的软、硬件开销加大了,这主要集中反映在时间开销上。如果采用较复杂的运算,就可能影响到整个系统的效率问题,所以说一个好的加密系统必须恰当地使用这两方面的技术,以达到最好的加密效果和系统效率。但是由于加密系统的设计不可能不存在漏洞,这就给解密者造成了可乘之机,不过由于加密技术的不断发展和程序设计的不断完善,这种可能性也将变成最小。

二、加密技术分析和加密软件

最初的加密技术是在 Apple I 机上实现的,由于 Apple I 机的软盘控制器完全可以由软件来控制,这使得 Apple I 机上的加密技术显得十分凶悍,并且现今许多的磁盘加密技术都来源于 Apple I 机的磁盘加密技术。

最早的 PC 机加密技术是一种防搬保护技巧的软件,它通常是个短小的文件,主要的作用是去磁盘的特定位置搬运真正的主程序,并把控制权交给它。在当时甚至有些保护方法是利用了当时人们对 PC-DOS 系统的陌生来实现的,如用隐含文件的方法来保护软件,在我国流行很广的《武士道》游戏和早期的王码 DOS 就是采用了该技术。当然这些技术只能对付普通的 COPY 命令,对于整盘拷贝就显得很脆弱了。

稍后一些的加密在技术上、性能上有了一定的进步,足以对付 COPY 和 DISKCOPY 命令。这些技术有:1. 异常的 ID 参数:在磁盘的每一个扇区上都有 ID 字段,它包括 4 个参数:柱面号 C、磁头号 H、扇区号 R 和长度 N。标准的软盘在这些参数上都是有序的,打乱和改变其中有些参数,就可以使正常的拷贝命令失去效果;2. 人为生成 CRC 错误:软盘控制器在向磁盘写入数据时,会自动生成 CRC 校验码,在正常情况下,这些校验码是正确的,所以加密者可以故意制造出一种错误的 CRC 校验码来防止软件的复制;3. 额外磁道技术:一张未格式化的倍密度磁盘一共有 48 个磁道,而通过 DOS 格式化后只有 40 个磁道,并以此作为 DOS 的标准软盘。其实 41、42 磁道也能正常、安全地工作。如果我们利用复制软件不能复制 40 以上的磁道的特性,安排用 41、42 磁道存放一些加密信息,就可以达到防拷贝的目的;4. 不格式化方法:DOS 对未格式化的磁盘是无法使用的,但如果我们在格式化软磁盘时,对软盘中某一磁道的部分扇区不做格式化处理,这样就使 DOS 无法复制这张软盘。

从上面可以看出,早期的防拷贝技术作出的特殊标记都是建立在非 DOS 标准基础上的,这样的方法有着致命的弱点:即完全可以由 COPY PC、COPY-WRIT 这样的分析拷贝工具,以字节或位元为单位来成功地复制。

(一)PROLOK 加密系统

1984 年美国加州 New Bury Park 市的 VAULT 公司推出了硬加密技术的典型代表作:激光加密 PROLOK 系统。

PROLOK 系统主要是利用了激光的方向性好、温度高的特点,用激光光束来照射软盘上区域极小的面积,使被照射处的磁介质损坏,来达到制造指纹的目的。由于激光的方向性极好,因此可以在软盘上精确地制造出上万种不同的指纹。

另外 PROLOK 加密系统在反跟踪技术上也下了相当大的功夫:

1. 防反汇编:PROLOK 加密系统是以密文形式装

入内存的,并在程序的执行过程中分块解密,执行后又重新加密,所以在任何时刻,想从内存中得到完整的解密程序代码是不可能的。PROLOK 加密系统采用的密文生成算法主要是程序代码逐字节及程序上下搬移。

2. 破坏中断向量表:前面已经说过,象 DEBUG、Turbo DEBUG 和 CODEVIEW 这样的调试工具,在运行时都利用了单步中断和断点中断。而 PROLOK 加密系统为了破坏调试工具的跟踪,就把存放单步、断点中断的程序入口地址作为自身的数据区,并且频繁地在这个区域中存取数据,从而破坏了单步、断点中断向量。

3. 大循环、多出口:PROLOK 加密系统采用了大循环技术来提高软件的反跟踪能力,它主要在程序中故意设置多重循环,而循环的出口又有许多个,且多由程序动态设置,这样就使跟踪者很难判断出程序的真正出口,并使他们耗费大量的时间与精力。

4. 隐蔽转移:为了加大解密难度和降低程序的可读性,PROLOK 加密系统采用了隐蔽程序走向的方法。PROLOK 加密系统用自己子程序的入口或转移地址来代替中断向量表中的中断程序入口地址,并用相应的中断命令进入下一步的程序中继续运行。

5. 设置堆栈段:利用中断的调试工具在工作时需要大量的堆栈来保存工作数据,而 PROLOK 加密系统把堆栈指针设置在内存的低地址段内,这样既可以保存有限的数据,又防止调试工具大批量在堆栈中存放数据,还可破坏大部分中断向量表的内容,真是一个十分有用的方法。

(二)LOCK89 加密系统

作为使用软加密技术的 LOCK89 加密系统,推出的时间远远晚于 PROLOK 加密系统,但它在反跟踪技术上要远远好于 PROLOK 系统。

LOCK89 的反拷贝技术是由一名台湾学者提出的,它的特点是利用磁盘中非写入部分的信息的随机性,来达到反拷贝的目的,这是一个典型的软指纹技术。生成这种技术的具体方法是:格式化时是按小扇区而读出时是按大扇区,这种技术可以生成一些未经程序写入的信息,且十分可靠,并且是不可复制的。

LOCK89 在反跟踪技术上有许多的创新,可以说它是继 PROLOK 加密系统后,加密技术上的又一大突破。

1. 关闭外围设备:上面已经说到过关闭显示屏、锁死键盘的反跟踪技术,而 LOCK89 加密系统则是在程序的一段执行时间内,关闭了所有设备,并在程序中多次出现锁死键盘的语句,这样就大大加深了程序的破译难度。

2. 使用堆栈指针:PROLOK 加密系统只是把堆栈指针设置在地址低端,而 LOCK89 则用堆栈指针作为程序中的累计和寄存器,这样就彻底阻止了解密者通过对堆栈指针的修改来达到继续跟踪程序的目的。

3. 破坏中断向量表:PROLOK 加密系统部分地破

坏了中断向量表,而 LOCK89 加密系统则完全不同,它破坏了所有的中断向量表和部分系统数据区(即内存中的 0:0000H~0:047FH 中的数据),这使得调试工具在执行时,无法使用到任何中断。

4. 密文技术:PROLOK、LOCK89 加密系统在该点上是基本相同的。LOCK89 把整个加密程序分成了 7 层,而其中的第六层和第七层就是加密系统的解密处理模块,其余五层的主要作用就是保护这两层。这 7 层中的第一层是可以直接反汇编(明文)的,其后的六层都必须由上一层来还原成明文。

5. 利用具有隐蔽作用的指令:LOCK89 加密系统除了象 PROLOK 一样,通过修改中断向量表,来达到程序流向的转移,还利用了许多交换指令反复地修改中断向量表和系统数据区中的内容。

6. 程序代码与内存代码的保护:为了阻止解密者通过修改加密程序的关键指令,来达到解密的目的,LOCK89 在程序中多次设置了计算代码累计和的程序段,一旦这其中发生了任何变化都将会导致系统的非正常运行。这样也恰好阻止了断点中断的执行,因为用 G 命令设置的断点地址内容在执行时是由 INT 3 来代替的,那样就修改了该地址的内容。LOCK89 对于内存中的程序代码也同样的进行了保护。首先,LOCK89 不论加密程序是否在成功的执行,还是发现了错误而退出,加密程序都将清除内存中的程序代码;还有为了保证加密程序不被驻留程序截获,LOCK89 的命令参数都在命令行上一次性给出。

硬盘是一种大容量的外部存储器,在实际中被广泛运用,但由于硬盘的开放式使用,带来了其信息的不安全性。于是加密技术中的另一个分支出现了,但到目前为止硬盘的加密技术远远及不上软盘的加密技术,甚至有些是直接的利用了软盘加密技术。

1. 隐蔽主引导记录法:硬盘有两个引导记录:主引导记录和 DOS 引导记录,其中主引导记录尤为重要,它将在系统启动时被读取,并被执行来引导操作系统。它还包括了硬盘分区表。如果将这部分信息移走并隐蔽,这将导致无法用硬盘上的操作系统来引导系统,更重要的是由于没有硬盘分区表,即使用软盘引导了系统,也无法进入硬盘进行操作。使用这种技术的典型代表是 PCLOCK。

2. 改变系统设置法:286 及 286 以上档次的机器都配置了 CMOS,这其中存放的是机器各种部件的设置,我们只要把其中的硬盘设置设为空,这样就隐蔽了硬盘的存在。

3. 直接使用软盘加密技术法:我们可以直接在硬盘上使用指纹在软盘上的系统,只不过使用时,我们必须把钥匙盘放在机器的软盘驱动器中,由加密程序直接从软盘上判断钥匙盘的真假。所谓钥匙盘是指由加密系统制造出的具有指纹的软盘。

4. 子目录加密法:子目录加密的方法比较多,主要有:①采用隐舍子目录名的方法;②在子目录的名字中加个半个汉字的方法;③修改子目录的入口地址法;

④改变子目录的长度法。

5. 利用装配程序:使用特殊的装配程序把被保护的软件拷贝到硬盘上,并把软件在硬盘上的特殊信息(如起始簇号)写入被保护程序的特定单元中,执行时由被保护程序比较两者的大小来决定是否正常运行。

三、解密工具及技术

虽然说现在的大部分指纹已经无法用软盘控制器来复制了,但我们在拿到加密盘时,也不妨用现有的拷贝软件来试一试自己的运气(笔者曾经用 COPY I PC 5.0 成功的复制了 Auto-dBASE 2.10)。实在无法拷贝的才考虑破译,因为破译加密程序是解密者与加密者在精力、体力、智力及水平上的一次短兵接触,这其中也还不乏运气的因素。

1. PC Tools 工具:PC Tools 是美国 Oregon 州 Portland 市的 Central Point 软件公司的产品。最早的推出时间为 1985 年,至今已有 8 个版本面市。利用它可以静态地分析加密系统、破译任何的文件名加密方法、成功的拷贝一些加密程序,如早期的 UC-DOS。希望公司的 UC-DOS 由于采用了特有的反拷贝技术,所以一些流行的拷贝工具根本无法复制它,而 PC Tools 则能轻松地复制它的副本。与 PC Tools 功能相似的还有 Norton。

2. SOURCE 工具:这是一种静态的反汇编工具,利用它可以帮助解密者分析加密程序。

3. LockSmith:这是一种分析拷贝软件,它具有读写磁盘扇区,分析、修改磁盘参数的功能,与其功能相似的还有加拿大多伦多市的 Quaid 软件有限公司的 Explorer,值得一提的是 Explorer 直接提供了一些制作指纹的功能,比如它能生成:①CRC 校验码错误;②弱位技术;③在某扇区中写入长度小于扇区 ID 参数中标识长度的数据;④删除数据标识;⑤短删除数据。

4. PC-DOS 操作系统:①DEBUG:这是一个机器码调试工具,它具有丰富的命令和极强的动态跟踪能力,所以在解密中被广泛使用。与它兼容的工具还有 SYMDEB、Turbo DEBUG 和 CODEVIEW;②CHKD-SK:这是一个检查磁盘目录与分配表,并报告磁盘和内存状态的 DOS 命令,它能破解第一类、第四类的子目录加密法,并能彻底破坏第三类的子目录加密法。

5. UNLOCK89、UNGUARD:这是一类专门破解某种加密方法的工具,前者能轻而易举的破译用 LOCK89 加密的程序,而后者则能对付 PROLOK 加密系统。

6. COPYWRIT、COPY I PC:对于某种加密系统,最好的方法是用专用型解密工具来拷贝和破译,但这也有着巨大的局限性:①适应面较窄;②推出时间较相应的加密系统要晚得多;③由于推出时间晚,使得加密系统的生命周期变长,可能导致加密系统成为一个成功的系统。为了减少以上缺陷的作用,加拿大的 Quaid 软件公司和美国的 Central Point 软件公司推出了具有复制指纹功能的工具: COPYWRIT 和 COPY I PC。

这两者是国内外最流行的拷贝工具,它们可以分析加密软盘并依据分析结果自动复制出与源盘相同的指纹。严格地说它们只是一种高级的拷贝软件,因为它们所复制的目标盘与源盘具有同样的指纹,并没有从根本上来破译它。这类拷贝软件具有一定的通用性,但由于加密技术的不断涌现,使得它们的功能在一定程度上受到了限制,唯一的解决办法就是版本的更新,所以这两种软件每年、每月都有新版本上市。许多书刊上都做过两者的功能比较,普遍认为同期的 COPYWRIT 比 COPY I PC 功能要强些,但笔者并不这样认为,因为两者在性能上基本相似,也各有特色,很难从功能上比较两者的强弱,正如上面说到的一样,笔者曾经用 COPY I PC 5.0 甚至是 4.0(87 年版),成功地复制了 Auto-dBASE 2.1,而 COPYWRIT 91(91 年版)则不能,所以说在很大程度上很难比较两者在功能上的强弱。

其实在实际运用中,我们很少能成功的使用象 COPYWRIT 和 COPY I PC 这类拷贝工具复制当今的一些加密系统,究其原因有以下三点:①COPYWRIT、COPY I PC 流行广,加密系统从设计上也主要针对它们来研制;②现今的反拷贝技术有一大部分已经无法用软盘控制器来复制;③拷贝工具在设计上主要针对一些已经出现的加密系统,对刚刚或将要出现的加密系统缺乏预见性,更主要的是它们的产地在西方国家,对于国内一些流行的加密系统比较陌生。

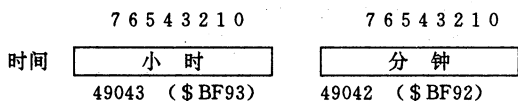
从理论上说,任何一种加密系统最终都可能被破译,只不过是时间的长短问题,如前面介绍的 PROLOK、LOCK89 加密系统都有专用型的解密工具。并且人们在解密过程中,发现解密工作是有规律和方法的:①产生与源盘相同的指纹;②无法生成指纹的,可以通过分析跟踪和修改加密程序来达到解密的目的,这是最艰苦也最有效的方法;③制作专用型解密工具。上面的第一种方法已经在许多方面没有用武之地了,第二种又显出劳动强度大、效率较低的缺点,因此解密者把目光投向了一个全新的领域:④监控 INT 13H:在 PC 机中软磁盘的操作都是通过调用中断 13H 来实现的,指纹的检测也不例外。因此我们可以编写一个驻留程序,监督控制 INT 13H 的执行,如果加密程序从软盘上读取指纹时,驻留程序可以用任何的方法来欺骗加密程序,使它认为任何的复制盘都是指纹盘。后面介绍的软件都是建立在这个方法上的。早在 80 年代加拿大的 Quaid 软件公司就利用这个方法破解了 PROLOK 加密系统,因为 PROLOK 加密系统在执行时要做一系列的检测,而 Quaid 公司的解密程序便设法使 PROLOK 认为已找到了激光孔,以迫使加密程序正常运行下去。

7. CHECK13、SIMINT13:这是一套由广大公司出品的解密工具,它们能够破译一些国内的加密系统。解密时由 CHECK13 监控加密程序的执行,当加密程序从源盘中读取指纹时,CHECK13 会从源盘中把指纹读出来并形成文件存放起来,正式执行时由 SIM-

INT13 通过监控 INT 13H, 并把 CHECK13 生成的文件调出来以骗取加密程序, 使它正常运行。

但是现今的许多加密程序为了对付这类解密工具, 在执行时首先检测内存, 如果发现可疑的驻留程序就停止运行, 这样就破坏了第七类解密工具的工作。为了解决这个问题, 人们通过探索从系统引导型病毒上学到了一种既能使程序驻留内存又能使加密程序无法检测到解密程序存在的方法: 系统数据区 0:0413H、0:0414H 中存放的是系统内存的大小, 如果减小它们的数量, 就将减少操作系统管理的内存, 那么运行于操作系统之上的加密程序也无法检测到不属于操作系统管理的内存中的情况, 这样我们就可以放心地把监控程序放入不属于操作系统管理的内存中。不过一般机器内存的大小都是有规律的, 如果加密系统通过检测系统数据区中的内存数据, 就可以发现这种监控程序的弊病。

(上接第 34 页)



此例程必须从一个 CLD 指令开始, 以一个 RTS 指令结尾。

要激活这个例程, 就要存入一个 JMP 到此例程的起始地址的指令到 DATETIM 单元 (\$BF06 ~ \$BF08)。

可以将这样一个例程放到内存的第 3 页 (\$300-3FF) 内。

二、中断处理例程

由于 Applesoft 语言的结构和编码, 在 Apple II 上中断需要对零页、堆栈和 6502 内部寄存器作特别处理, 为辅助可以处理中断的软件开发, MLI 提供一个连接中断驱动设备的规范。

要使用中断, 必须安装一个到四个中断接收例程到内存。你应该检查并修改系统位图, 确信例程不与 ProDOS 或其它同时执行的程序发生冲突。

一旦安装了一个例程, 你必须用 ALLOC-IN-

8. RCOPY02: 1991 年江西省科达通信电脑技术发展公司用上面的观点研制开发了 RCOPY02, RCOPY02 具有三大功能: ①拷贝软件; ②加密程序; ③保留现象。从根本上说这三个方面的原理是一样的。它通过专用键在某一时刻把计算机中运行的程序和处理的数及其它有关信息, 保存在软盘上, 下次使用时便可以把它们从软盘上读取出来, 并放入内存中相应的位置。RCOPY02 虽然是一个很有特色的软件, 但功能上并非完善: ①需要较多的软盘空间; ②生成的目标盘只能为自启动软盘; ③很难在 CCDOS 下使用; ④适应面较窄。

现今的加密、解密技术都不是一劳永逸的, 因为两者是相互影响、相互发展和相互依存的。加密技术的每一个新发现, 都将导致解密技术的新突破, 而解密技术的每一个新突破都又导致了加密技术的新发展, 所以它们之间的竞争是永无止境的。

TERRUPT 调用通知 MLI 接收例程的起始地址。在这调用安装完后, 才可以允许硬件中断。

当中断发生时, MLI 的中断处理程序保存 6502 的寄存器、零页 \$FA 到 \$FF 单元, 若堆栈的数据已超过 3/4, 则也保存堆栈的 16 个字节数据, 然后它调用每个接收例程(通过 JSR), 按照它们被安装的顺序逐一调用。每个安装的例程必须以 CLD 指令开始。

当调用可以处理相应中断要求的例程时, 它会进行工作, 在清除硬件中断并通过一个 RTS 返回时, 清除进位标志。当调用一个不能处理相应中断的例程时, 它返回并设置进位标志以便 MLI 知道应调用列表内下一个中断例程。

如上所述, 所有 6502 寄存器、\$FA 到 \$FF 单元、堆栈的 16 个字节(若栈内数据超过 3/4)被保存, 中断例程可以自由地用这些资源作为临时数据存储。

在没有安装和分配中断例程时要求一个中断, 那么系统将显示 SYTEM FAILURE 并停止程序的执行。

在结束一个中断驱动设备时, 会进行一个 DEALLOC-INTERRUPT 调用。 (待续)

(上接第 36 页)

升沿通过 U₁₄、U₁₅ 产生自动复位信号 XRST, 使系统立即进入复位状态(系统的另一路复位信号 YRST 是通常的上电复位和手动复位, 图中未画出)。

复位信号还经由 U₁₇、U₁₈ 提供一负触发信号给纠错系统清除端, 以建立新的起始状态。

图 2 还给出了某些信号状态。括号外表示稳定时的电平值, 括号内则表示纠错复位时的波形变化情况。

当然, 自动复位后, 系统软件还要做相应处理, 对此已有许多资料可供查阅。

最后为说明虚读方式所起的作用, 再看看程序 ROM 与其映像 ROM 内容的对应关系。这从下面的程序片断便一目了然, 就不再详述了。

程序 ROM	映像 ROM
· ·	· ·
· ·	· ·
· ·	· ·
19CD F5 F0	19CD 06 80
19CF 12 19 D5	19CF 05 80 80
19D2 C5 F0	19D2 06 08
19D4 C4	19D4 06
19D5 54 0F	19D5 06 80
19D7 24 90	19D7 06 80
19D9 D4	19D9 06
19DA 34 40	19DA 06 80
19DC D4	19DC 06
19DD 22	19DD 05
· ·	· ·
· ·	· ·
· ·	· ·

解决 TSR 程序一次性驻留内存的根本方法

辽宁省铁岭市委办公室(112000) 宋立波

TSR 程序驻留内存有两种方法:一是利用中断 INT 27H;二是利用 DOS 中断功能调用 INT 21H 的 31H 号子功能。这两种方法都能使程序重复驻留内存,如果某 TSR 程序已经驻留内存,下次再运行该程序时不再使其驻留内存,这该怎么办呢?在诸多报刊杂志刊登的实用程序中,已有很多人通过扩充 DOS 中断调用 INT 21H 的未被利用的子功能号方法来实现这一问题。如:《计算机世界》月刊 1993 年第 3 期刊登的郭永华同志的《实现 DOS 自动登录文件的方法》、《计算机应用研究》月刊刊登张鸿鸣、刘铁军等的《一个解决微机“死机”的有效方法》等文章都采用了该方法(具体使用方法见程序一)来实现 TSR 程序一次性驻留内存。其基本原理是:在主程序中调用一次中断 INT 21H,并且扩充 DOS 中断向量的子功能,在扩充的 INT 21H 子功能中判断被扩充的子功能号(该号 DOS 未用),根据判断结果设置相应返回码值,在下次重新运行该程序时根据该返回码即可判断该程序是否已驻留内存。在其它的报刊中也曾有人利用 DOS 的 SET 批命令设置某一环境变量的方法来实现程序的一次性驻留内存(见程序二)。这两种方法都不是理想的方法,经本人长期使用发现各有其致命的缺点:

(一)程序一所采用方法的缺点

1. 该方法需要修改中断 INT 21H,增加了系统的开销;

2. 此方法实际上是对 INT 21H 中断进行了扩充,这就有可能与其它对 INT 21H 进行功能扩充的程序相抵触;

3. 中断 INT 21H 可有 0FFH 个子功能,其中被 DOS 所利用的已经过半,当利用此方法的 TSR 程序稍一多就会发生后运行的某些 TSR 程序不能驻留内存现象,也会出现返回码与其它程序冲突等混乱现象。

(二)程序二所用方法的缺点

1. 此种方法需要设置相应的环境变量,这也有可能发生环境变量冲突现象;

2. 环境变量的多少受环境块大小的限制,环境变量一多,一是增加系统的开销,二是环境块根本放不下过多的变量;

3. 每个 TSR 程序都要编制相应的批处理程序,这增加了磁盘空间的开销和工作量,而且一旦批处理运行时发生某些意外,会造成环境变量已经设置而 TSR 程序已无法驻留内存的现象等。

由此可知以上两种方法虽能简单地使 TSR 程序一次性驻留内存,但不是解决 TSR 程序一次性驻留内存的根本方法。本人经查阅有关资料,摸索出了一种利

用 DOS 保留中断 INT 21H 的 52H 号功能调用实现 TSR 程序一次性驻留内存的方法,经笔者试用证明其的确是一种理想的方法,其原理如下:

一、DOS 管理内存的基本原理

DOS 管理内存是通过内存块来完成的,内存块包括两部分:(1)内存控制块 MCB;(2)内存分配块 MAB。内存控制块的大小为一节(16 字节),内存分配块的大小要根据具体驻留程序确定。内存块的具体结构如下:

内存控制块 MCB(16 字节)
程序段前缀 PSP(100H 字节) } 内存分配块 MAB
内存驻留程序部分

二、INT 21H 的 52H 号保留中断功能

INT 21H 的 52H 号中断功能调用主要是确定内存控制块链中第一个内存控制块 MCB 的具体地址,根据第一个 MCB 的地址利用公式可计算出下一个 MCB 的具体地址。其调用格式为:

```
MOV ah,52H
```

```
INT 21H
```

该操作的返回值为:ES:[BX-2]指向第一个 MCB 段地址,其偏移值为 0。下一个 MCB 具体段地址的计算公式为:下一个内存控制块 MCB 段地址=本内存控制块 MCB 段地址+本内存分配块 MAB 大小+1。内存控制块 MCB 的 16 字节具体内容如下:

偏移:	具体内容:
+00h	标识符 4DH 或 5AH,4DH 是一般内存控制块标识符,5AH 是最后内存控制块标识符,不是这两个值则说明内存控制块链错误。
+01h-02h	程序段前缀 PSP 段地址,如果内容为 0 则此内存块为自由块。
+03h-04h	内存分配块 MAB 节大小(一节为 16 字节)
+05h-0fh	保留

三、具体实现原理

该方法是通过在 TSR 程序的内存分配块 MAB 的 100H(即内存驻留程序的首字节,程序段前缀 PSP 后的第一个字节)处设置相应的驻留通行字,并在主程序中通过 INT 21H 的 52H 号子功能调用确定第一个 MCB 的段地址,根据第一个 MCB 的段地址分别计算出其它 MCB 的段地址值,这样就可判断各个 TSR 程序首处是否有本 TSR 程序的驻留通行字(采用此方法

的每个 TSR 程序要有自己的驻留通行字),以确定该 TSR 程序是否已经驻留内存中。

本人总结出的这个方法适合于任意大小的各种实用 TSR 程序,较其它方法有许多优点:1. 驻留通行字的长度可以是任意的,这样驻留通行字越长通行字重复的可能性就越低,可靠性比其它方法高;2. 系统开销较小,只有驻留通行字长度大小;3. 只要设置好自己的驻留通行字,不易与其它任何 TSR 程序相冲突,更不会造成机器的死锁现象等;4. 只要将此方法所用的一段小程序和相应的驻留通行字数据区加到各种实用软件中去即可实现 TRS 程序的一次性驻留内存,而不必象成形软件那样利用程序二所用的繁琐方法。

具体使用方法见程序三(COM 文件的使用格式)、程序四(EXE 文件的使用格式),在这两上例子中笔者利用了一个 TSR 程序,该 TSR 程序的功能是在屏幕的左上方显示字符“RESIDENT”,在实际工作中读者可将该 TSR 程序换成自己 TSR 的程序,就可实现您的 TSR 程序一次性驻留内存。程序清单中标号 start1 和 start2 之间为 TSR 程序驻留通行字判断部分,读者在使用本方法时只要将该段程序加到应用程序的起始执行部分即可,并在 TSR 首处设通行字。

程序一(一般人员采用的格式)

```
code    segment
        assume cs:code,ds:code
        org 100h
start:   jmp init
old21h  dd ?
new21h  proc far
        cmp ax,0ff00h
        jne aal
        mov al,0feh
        iret
aal:     push ds
        push es
        pop es
        pop ds
        assume ds:nothing
        jmp cs:old21h
new21h  endp
init:   mov ax,0ff00h
        int 21h
        cmp al,0feh
        jne next
        mov dx,offset masg
        mov ah,09h
        int 21h
        int 20h
next:   mov ax,3521h
        int 21h
        mov word ptr old21h,bx
```

此处为修改 INT21H 所增加的中断服务程序部分,该部分判断子功能号 FFH 是否被调用。

此段程序为判断 INT21H 的 0FFH 号子功能调用的返回值,并根据该值判断该程序是否已经驻留内存。

```
mov word ptr old21h+2,es
mov dx,offset new21h
mov ax,2521h
int 21h
mov dx,offset init
int 27h
masg   db '<hist>has been installed!',0ah,0dh,07,'$'
code   ends
end start
```

程序二(批处理命令实现的方法,以 CRT.EXE 内存驻留程序为例)

```
C>TYPE CRTRES. BAT
@echo off
if exist %tsr% goto end
set tsr=crt.exe
crt
goto end2
:end
echo 文件%tsr%已经驻留内存!
:end2
@echo on
```

程序三(COM 文件使用格式)

```
;resid1.asm
coseg   segment
        org 100h
        assume cs:coseg,ds:coseg
start:   jmp start1
pass     db 7 dup(0)
resid    db 00h
colr     dw 0000h
offlc    dw 0000h
seglc    dw 0000h
count    dw 0000h
linep    dw 0000h
address  dw 0000h
scri     db 'Resident has been installed!',07h,24h
appmeg   db 'R',05h
         db 'E',05h
         db 'S',05h
         db 'I',05h
         db 'D',05h
         db 'E',05h
         db 'N',05h
         db 'T',05h
         db 0
dir      proc far
        push ax
        push cx
        push di
        push si
        push ds
        push es
        pushf
```

该 TSR 程序存放驻留通行字处,本程序设置为 10 字节长度。读者在使用该方法时必须在该 TSR 程序最开始处置,其通行字长度和具体内容读者自己设置。

笔者设计的 TSR 程序


```

call dword ptr cs,off1c
inc cs;count
cmp cs;count,0010
jb return
mov cx,0000h
mov word ptr cs;count,cx
mov cx,word ptr cs:colr
mov es,cx
mov dx,word ptr cs;address
mov di,word ptr cs;linep
lea si,word ptr cs:appmeg
mov cx,0010h
cli
notall2: in al,dx
test al,01h
jnz notall2
mov ah,byte ptr cs:[si]
notall1: in al,dx
test al,01h
jnz notall1
mov byte ptr es:[di],ah
inc di
inc si
loop notall2
sti
return: pop es
pop ds
pop si
pop di
pop cx
pop ax
iret
dir endp
start1: mov ah,52h
int 21h
mov ds,es:[bx-02]
mov byte ptr cs;resid,00h
residbg: xor si,si
cmp byte ptr [si],4dh
jnz not4dh
add si,0110h ;主程序运行
lodsw ;前首先判断
cmp ax,0cbceh ;TSR 程序
jnz notpw ;是否已经驻
lodsw ;留内存中。
cmp ax,0c1a2h
jnz notpw
lodsw
cmp ax,0b2a8h
jnz notpw
lodsw
cmp ax,0d6f8h
jnz notpw
lodsb
cmp al,0
jnz notpw
mov byte ptr cs;resid,01h
jmp not4dh
notpw: mov ax,ds ;计算下一个
inc ax ;MCB 的段
xor si,si ;地址值。
add ax,[si+03h]
mov ds,ax
inc ax
jmp residp
not4dh: xor ax,ax
residp: jnz residbg
cmp byte ptr cs;resid,01h
jz exit
push cs ;TSR 程序
pop es ;没驻留则设
lea di,pass-03h ;置本 TS
mov ax,0cbceh ;R 程序的驻
stosw ;留通行字,
mov ax,0c1a2h ;该 TSR 程
stosw ;序中设置为
mov ax,0b2a8h ;CE,CB,C1,A
stosw ;2,B2,A8,D6
mov ax,0d6f8h ;F8,00
stosw
xor al,al
stosb
jmp start2
exit: push cs ;TSR 程序
pop ds ;已驻留提示
mov ah,09h ;部分
mov dx,offset scr1
int 21h
mov ax,4c00h
int 21h
start2: mov ax,0000h ;使 TSR 程序
mov ds,ax ;序驻留内存
cli ;的主程序部分。
mov ax,word ptr ds:[0070h]
mov word ptr cs;off1c,ax
mov ax,word ptr ds:[0072h]
mov word ptr cs;seg1c,ax
mov word ptr ds:[0070h],offset dir
mov word ptr ds:[0072h],cs
sti
cmp byte ptr ds:[0463h],0d4h
jnz notcolr
mov word ptr cs:colr,0b809h ;彩显 VRAM
;段地址
mov word ptr cs;address,03dah
jmp colrl
notcolr: mov word ptr cs:colr,0b009h ;单显 VRAM
;地址段
mov word ptr cs;address,03bah
colrl: mov word ptr cs;linep,0000h
mov dx,offset start1
mov cl,04h

```



```

        add dx,000fh
        shr dx,cl
        mov ax,3100h ;TSR 程序节驻留方式退出
        int 21h
coseg   ends
        end start

```

程序四(EXE 文件使用格式)

```

;resid2.asm
coseg   segment public'code'
dir     proc far
        assume cs:coseg
        jmp begin
pass    db 7 dup(0)      ;此处为 EXE 格式的 TSR
resid   db 00h          ;程序驻留通行字处,该
colr    dw 0000h        ;列设置为 10 个字节长度。

```

该段程序与程序三的相应部分完全相同。

```

begin:   push ax
        push cx
        push di
        push si
        push ds
        push es
        push cs      ;这两条程序一定要加入,否则
        pop ds       ;该 TSR 程序不能正常运行。
        pushf

```

该段程序与程序三的相应部分完全相同。

```

        iret
dir     endp
main    proc far
        assume cs:coseg,ds:dseg,ss:stacks
start1: mov ax,dseg
        mov ds,ax
        push ds
        mov ah,52h
        int 21h

```

该段程序与程序三的相应部分完全相同。

```

        cmp al,0
        jnz notpw
        pop ds
        mov byte ptr cs:resid,01h
        jmp not4dh1

```

```
notpw:   mov ax,ds
```

该段程序与程序三的相应部分完全相同。

```
not4dh:  pop ds
not4dh1: xor ax,ax
residp:  jnz residbg

```

该段程序与程序三的相应部分完全相同。

```

        mov word ptr ds:[0070h],offset begin
        mov word ptr ds:[0072h],cs

```

```

        sti
该段程序与程序三的相应部分完全相同。
colr1:  mov word ptr cs:linep,0000h
        mov dx,offset cs:main
        mov ax,3100h
        int 21h
main    endp
coseg   ends
dseg    segment 'data'
scr1    db '<Resident>has been installed!',07h,24h
dseg    ends
stacks  segment stack 'stack'
        dw 64 dup(0)
stacks  ends
        end main

```

注:程序四标注有“与程序三相应部分完全相同”字样的只有驻留通行字不同,程序四采用的驻留通行字为:CB,CE,A2,C1,A8,B2,F8,D6,00,这是为了节省篇幅,读者在应用时一定要设置好驻留通行字。

**把实验室搬进电脑,
红峰示波卡的宗旨!**

武汉红峰电子工业公司
(430070)

Tel:027-700564
FAX:027-703396

FoxBASE 下一个新型的通用菜单控制程序

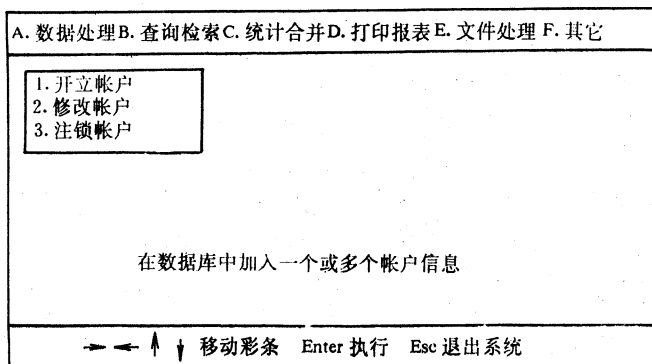
浙江省龙泉市计算机协会 周丝溪 胡武海

下拉式菜单是被编程者所广泛采用的一种菜单驱动形式。FoxBASE 为我们提供了实现该功能的现成语句——MENU。尽管这给编程者带来了很大的方便，但这一语句也有其诸多不尽人意的地方，首先是由于汉化的问题，屏幕效果不甚理想，其次是操作还不是十分方便。因为这种菜单是由一个横穿屏幕顶行的“菜单标题”和相应的“下拉菜单”组成的。只有当某一菜单标题被选中后，才会在此标题下下拉出相应的子菜单来。即从“菜单标题”到“子菜单”必须经过一次有效的选择，同样，从“子菜单”到“菜单标题”也必须选择一个“返回”项。因此，从操作效果上看它实际上相当于一个二级菜单。这无疑增加了击键次数，也在一定程度上影响了其直观性和方便性。

笔者在用 FoxBASE 开发的《银行帐户管理系统》(ZHGL.PRG, 清单附后)中,对改进这一不足作了一点小小的尝试,获得了比较满意的效果。该程序的菜单形式既具有下拉式菜单的基本效果,又克服了二次选键的不足,把两级菜单融为一体,改善了屏幕效果。同时,将菜单的所有内容以及相应的子程序名和提示信息等以文件的形式存于磁盘内,使之具有较强的通用性。要更改菜单内容或增减菜单选择项的数目,都无需修改程序本身,只需修改数据文件即可。因此,它可以很方便地被移植到其它的系统中去。

一、菜单的屏幕效果

该程序产生的菜单效果类似于 PC Tools 6.0 的菜单。它由一个横穿屏幕顶行的菜单条和若干个下拉菜单组成,菜单条包含有六个菜单标题,每个菜单标题对应一个包含若干个选择项的下拉菜单。当运行该程序时屏幕显示如下:



图中,菜单条的 A 标题醒目,同时彩条(红底黄字)停留在下拉菜单的第一个选择项上。此时若按 ↑ 或 ↓ 游标键即上下移动彩条,若按 → 或 ← 键即显示出右边或左边的一个下拉菜单窗口,原有下拉菜单随即消失,彩条的行位置与原先同,标题的醒目位置也随即改变,如此可往返、纵横随意选择,直至最终选中欲执行的项目时,按回车键即执行相应的功能模块。同时在选择的过程中,游标键(← → ↑ ↓)选择和编码选择是兼容的,从而使操作更为简捷、直观。

二、编程技巧

该程序并没有采用 FoxBASE 所提供的 MENU 语句,而主要是利用了以下几种编程技巧:

1. 利用了键值获取函数

FoxBASE 中的 READKEY() 函数,是一个键值获取函数,它可以在程序运行过程中截获最近一次击键的键值。譬如:按下 ↑ 键 READKEY() 返回的值为 4, ↓ 为 5, ← 为 0, → 为 1, Esc 为 12, Enter 为 15 等等。本程序就是利用这个函数来测试用户击键的键值,并据此判断用户的菜单选择的。

2. 巧用字符串截取函数

在顶行的菜单条选择中(即根据用户的选择改变菜单条的醒目位置),巧妙地利用了字符串截取函数 SUBSTR()。其方法是:①定义词组变量 WORDS 并赋予它一个包含所有菜单标题的字符串,每个标题串长为 10;②根据用户的选择确定 J1、J2 的值;③利用 SUBSTR() 函数,以表达式: WORD1 = SUBSTR(WORDS, J1 * 10 - 9, 10) 和 WORD2 = SUBSTR(WORDS, J2 * 10 - 9, 10) 从字符串 WORDS 中截取两个字符串;④以不同的颜色在指定的位置分别显示

这两个字符串,以达到改变标题醒目位置的目的。

3. 利用了二维内存变量数组及宏替换函数

程序中用了三组二维内存变量数组,第一组(MEU(I,J))赋予各子菜单选择项的内容,第二组(PRO(I,J))赋予各选择项对应的子程序名,第三组(REM(I,J))赋予各选择项相应的提示信息。然后,根据用户的选择确定三个内存变量数组的下标I、J的值,从而实现了下拉菜单窗口和彩条的移动以及相应的提示信息的显示。同时,将PRO(I,J)的值赋给内存变量FILE,然后执行一个DO &FILE语句,调用相应的子模块,使程序变得十分简洁。

4. 采用了内存变量文件

上述的词组变量及三组二维内存变量数组都以内存变量的形式存于磁盘内。其方法是:用编辑软件编写了一段FOX程序(CAMEU.PRG,清单附后),给这些变量赋值,最后用SAVE TO语句将这些变量写入一个内存变量文件(MENUDATA.MEM),使用时只需在FOX下执行一次该程序就可自动生成这个内存变量文件。然后在菜单程序中用RESTORE FROM语句将此文件的内容读入内存,供菜单显示用。

源程序清单:

```
ZHGL.PRG
*** 参数设置 ***
set menu off
set escape off
set status off
set message to 24
set talk off
set scoreboard off
*** 系统初始化 ***
restore from menudata && 读内存变量文件
move=" "
chara=" "
character=" "
cho=0
i1=1
i2=3
j1=2
j2=1
*** 给出屏幕菜单 ***
do while .t. && 循环 A 顶
set color to 2+
clear
@ 23,0 say repl ("=",80)
@ 24,13 say' →←↑↓ 移动彩条, Enter;
执行, Esc 退出系统'
set color to 6+/3
@ 1,0 say word && 显示菜单条
@ 2,0 say repl ("=",80)
** 根据用户选择确定数组的下标值 **
do while .t. && 循环 B 项
do case
case READKEY()=1
j1=j1+1
```

```
j2=j1-1
case READKEY()=0
j1=j1-1
j2=j1+1
case READKEY()=5
i1=i1+1
i2=i1-1
i1=i1-1 < CASE READKEY()=4
i2=i1+1
endc
ch1=uppe(character)
if ch1="A" .or. ch1="B" .or. ch1="C" .or. ;
ch1="D" .or. ch1="E" .or. ch1="F"
j2=j1
chara="y"
endif
do case
case uppe(character)="A"
j1=1
case uppe(character)="B"
j1=2
case uppe(character)="C"
j1=3
case uppe(character)="D"
j1=4
case uppe(character)="E"
j1=5
case uppe(character)="F"
j1=6
endcase
** 修正数组下标值 **
j1=iif(j1=7,1,j1)
j1=iif(j1=0,6,j1)
j2=iif(j2=7,1,j2)
j2=iif(j2=0,6,j2)
i1=iif(i1=column(j1)+1,1,i1)
i1=iif(i1=0,column(j1),i1)
i2=iif(i2=column(j1)+1,1,i2)
i2=iif(i2=0,column(j1),i2)
i1=iif(i1>column(j1),column(j1),i1)
i2=iif(i2>column(j1),column(j1),i2)
** 弹出菜单窗口 **
word1=subs(words,j1*10-9,10) && 截取两个字符串
word2=subs(words,j2*10-9,10)
n1=(j1-1)*13
m1=(j2-1)*13
if readkey()=0 .or. readkey()=1 .or. j1=2 .or. dys="y"
.or. chara="y"
set color to 2
@ 3,0 clear to 9,79 && 清除原有下拉菜单窗
set color to 6+/3
k=1
do while k<=column(j1) && 循环 C 顶
@k+2,n1 say"|" +meu(k,j1)+"|" && 显示下拉菜单窗口
k=k+1
enddo && 循环 C 底
```

```

dys="n"
chara=" "
if j2=6
@ 2,m1 say repl ("-",15)
else
@ 2,m1 say repl ("-",18)
endi
if j1=4
@ 2,n1 say" ┌───────────────────┐" &&.
@ k+2,n1 say" └───────────────────┘" &&.
else
@ 2,n1 say" ┌──────────┐" &&.
@ k+2,n1 say" └──────────┘" &&.
endif
endif
** 移动彩条 **
set color to 6+/3
@ i,m1+1 say" "+word2+" " &&. 覆盖菜单条的原醒目
标题
set color to 2+
@ 1,n1+1 say" "+word1+" " &&. 显示醒目标题
set color to 6+/3
@ i2+2,n1+2 say meu(i2,j1) &&. 覆盖原彩条
set color to 6+/4+
@ i1+2,n1+2 say meu(i1,j1) &&. 显示彩条
set color to 6+
@ 20,0 clear to 20,79
@ 20,10 say rem(i1,j1) &&. 提示信息
** 接收用户选择 **
character="^"
set color to 0,0
@ 21,0 say"" get character
read
if READKEY()=15.or.READKEY()=12.or.(val(char-
acter<=column(j1).and.val(character)>0.and.character
<>" ")
dys="y"
exit &&. 选中退出循环 B
endif
enddo &&. 循环 B 底
** 用代码选择时彩条的移动 **
if val(character)<=column(j1).and.val(character)>0
i2=i1
stor val(character)to i1
set color to 6+/3
@ i2+2,n1+2 say meu(i2,j1)
set color to 6+/4+
@ i1+2,n1+2 say meu(i1,j1)
set color to 6+
@ 20,0 clear to 20,79
@ 20,10 say rem(i1,j1)
endif
** 调用子模块 **
set color to 2
mms=""
if readkey()=12.or.(i1=5.and.j1=6)

```

```

do quit &&. 调用“退出”子模块
if cho=3
loop
endif
exit
else
file=pro(i1,j1) &&. 将过程名赋给变量 FILE
do &file &&. 调用相应的子模块
endif
if cho=2.or.cho=1
exit
endif
enddo
if cho=2
set status on
set scoreboard on
endif
close procedure
close database
if cho=1
quit
endif
return

```

CAMEU.PRG

```

*** 各下拉菜单选择项的个数 ***
dime column(6)
column(1)=3
column(2)=3
column(3)=2
column(4)=4
column(5)=3
column(6)=5
*** 菜单标题 ***
word=' A.数据处理 B.查询检索 C.统计合并;
D.打印报表 E.文件处理 F.其它'
words=' A.数据处理 B.查询检索 C.统计合并;
D.打印报表 E.文件处理 F.其它'
*** 各下拉菜单选择项的内容 ***
dime meu(5,6)
meu(1,1)="1.开立帐户"
meu(2,1)="2.修改帐户"
meu(3,1)="3.注销帐户"
meu(1,2)="1.单项查询"
meu(2,2)="2.多项查询"
meu(3,2)="3.模糊查询"
meu(1,3)="1.合并数据"
meu(2,3)="2.汇总数据"
meu(1,4)="1.打印统计表"
meu(2,4)="2.打印备案表"
meu(3,4)="3.打印汇总表"
meu(4,4)="4.打印对照表"
meu(1,5)="1.索引文件"
meu(2,5)="2.排序文件"
meu(3,5)="3.建立文件"
meu(1,6)="1.建立表头"

```



```

meu(2,6)="2. 设置参数"
meu(3,6)="3. 数据备份"
meu(4,6)="4. 拷回备份"
meu(5,6)="5. 退出系统"
*** 各功能模块的子程序(或过程)名 ***
dime pro(5,6)
pro(1,1)="INDATA"
pro(2,1)="MODATA"
pro(3,1)="DELDATA"
pro(1,2)="FINDA"
pro(2,2)="FINDB"
pro(3,2)="FINDC"
pro(1,3)="ADDA"
pro(2,3)="TOTADT"
pro(1,4)="PRTA"
pro(2,4)="PRTB"
pro(3,4)="PRTC"
pro(4,4)="PRTD"
pro(1,5)="INDFIL"
pro(2,5)="SORFIL"
pro(3,5)="CREFIL"
pro(1,6)="CREHD"
pro(2,6)="STDATA"
pro(3,6)="BACKDT"
pro(4,6)="RESDT"
pro(5,6)="quit"

```

```

*** 提示信息 ***
dime rem(5,6)
rem(1,1)=' 在数据库中加入一个或多个帐户信息'
rem(2,1)=' '
.....(略)
*** 将全部变量写入内存变量文件 ***
save to meudata
return

```

```

QUT. PRG
set color to 6+/4+,0/3
@ 10,10 clear to 14,63,
@ 12,20 prompt '1. 返回 DOS'
@ 12,35 prompt '2. 返回 FOX'
@ 12,50 prompt '3 取消'
menu to cho
cho=iif(cho=0,3,cho)
do case
case cho=2 .or. cho=1
set color to 2,3
return to master
case cho=3
return
endcase

```

(注:本程序运行环境是:FoxBASE 2.0,CEGA 显示卡,DOS 3.0 以上)

在批处理文件中巧用 DOS 环境变量

福建省交通科学技术研究所(350004) 黄庆程

一、DOS 环境的概念及使用方法

DOS 的环境是一个由 DOS 设置的特殊存储区,其中存储的变量可供批文件或用户程序使用,可以用环境来存放任一种信息。实际存储 DOS 环境的内存区叫主环境块。环境中的每一项都是一个 ASCII 码字符串,其形式为:

环境变量名=环境变量的内容

由 DOS 自动建立的环境变量共有三个: PROMPT、PATH、COMSPEC。PROMPT 变量是在用户打入或从批文件中输入 PROMPT 命令时建立的。若用户从未使用过 PROMPT 命令,DOS 将自动建立一个环境变量 PROMPT,其内容为 \$n\$g。PATH 变量也是在用户打入或从批文件中输入 PATH 命令时建立的。若用户从未使用过 PATH 命令,则不建立这个变量。COMSPEC 变量是在用户打入或从批文件中输入 SET COMSPEC=...时建立的。若用户从未使用过这个命令,DOS 将自动建立一个环境变量 COMSPEC,其内容为 /COMMAND.COM。

同时,DOS 允许用户在命令行上,通过 SET 命令向环境内存块添加新设置的环境串或修改原有的环境串。SET 命令的格式共三种:

- ①SET
- ②SET 变量名=
- ③SET 变量名=变量内容

第一种格式是把环境中的各个变量在屏幕上显示出来。一种可能的结果是:

```

COMSPEC=C:\COMMAND.COM
PATH=C:\213;C:\TOOLS
PROMPT=$n$g

```

第二种格式删除所指定的环境变量。

第三种格式是建立(或修改)所指定的环境变量。

二、在批处理文件中巧用 DOS 环境变量

在 DOS 命令行下,只能通过 SET、PROMPT 和 PATH 命令来存取和修改环境字符串。而在批处理文件中,除了借助于这三条命令外,还可通过百分号(%)把环境变量包括起来的办法来引用环境变量的值。批

处理文件中使用 DOS 环境变量,可使批处理文件更加实用、灵活,能解决不少问题。下面给出几个巧用 DOS 环境变量实现的批处理文件实用例程。

①不同参数下重复执行一个 DOS 命令

有时我们需要用不同的参数一次又一次地重复运行同一个命令,例如用 DIR 查找所有扩展名为 EXE、COM、BAT 的文件时,必须依次键入 DIR *.EXE、DIR *.COM、DIR *.BAT,这很麻烦。下面提供的批处理文件例程 REPEAT.BAT 将较好地解决这个问题,提高工作效率。批处理文件例程 REPEAT.BAT 运行时将以不同的参数重复执行 DOS 命令。在使用该批处理文件时,需键入 REPEAT 及一个 DOS 命令和参数表。例如键入:REPEAT DIR *.EXE *.COM *.BAT 就可列出当前驱动器中所有扩展名为 EXE、COM 和 BAT 的文件。具体程序清单如下:

```
@ECHO OFF
IF "%1"="" GOTO INSTRUCT
IF "%2"="" GOTO INSTRUCT
SET CMD=%1
:DOCMD
ECHO %CMD% %2
%CMD% %2
SHIFT
IF NOT "%2"="" GOTO DOCMD
GOTO END
:INSTRUCT
ECHO ERROR:MISSING PARAMETERS
ECHO USAGE:REPEAT COMMAND PARAMETER
[PARAMETER...]
ECHO EXAMPLE:REPEAT DIR *.EXE *.COM *.
.BAT
:END
SET CMD=
```

该批处理例程巧妙地应用了几项批处理文件编程技术。其一是利用临时 DOS 环境变量 CMD 来存放 DOS 命令;其二是利用批命令的子命令 SHIFT 来变换参数,以遍历参数表中的所有参数。其三程序结尾处的“SET CMD=”用来释放临时环境变量 CMD 的空间,以节省内存,这是批文件编程中容易被忽略的技术。

②转换路径

在批处理文件中使用环境变量的一个最大好处是可以重新定义这些变量。通过下面两个批处理文件可实现路径的转换。先建立并运行批文件 SETPATH.BAT,用 4 个不同的变量名表示 4 个不同的路径:

```
@ECHO OFF
SET PATH1=C:\213
SET PATH2=C:\TOOLS
SET PATH3=C:\PROGRAM
SET PATH4=%PATH%
```

再建立一个转换路径的批文件 SWPATH.BAT:

```
@ECHO OFF
```

```
IF %1==1 PATH=%PATH1%
IF %1==2 PATH=%PATH2%
IF %1==3 PATH=%PATH3%
IF %1==4 PATH=%PATH4%
```

则在以后只要键入命令 SWPATH n 就可以转换成相应的路径。当 n 是 1 到 3 时相应地将路径转换到 PATH1、PATH2、PATH3。当取 4 时,则恢复到运行 SETPATH.BAT 以前设置的路径。

③随意添加路径

假如路径已经设置为 (PATH=C:\213;C:\TOOLS,现在欲再增加一条路径 C:\PP,这时不能仅仅键入:PATH=C:\PP,而必须键入:

```
PATH=C:\213;C:\TOOLS;C:\PP
```

当路径字符串很长时,这样重新打一遍是很麻烦的。如果建立如下这样一个批文件 ADDPATH.BAT,只须包含这样一行:PATH=%PATH%;%1,则只须键入:

```
ADDPATH C:\PP
```

就可把 C:\PP 添加到原来 PATH 命令指定的路径中去。

④防止内存驻留程序的重复装载

我们知道,并不是所有的内存驻留程序都有避免被多次装载的机制,为了避免一次又一次地装载同一内存驻留程序,节省内存开支,可在 AUTOEXEC.BAT 文件中插入命令行

```
SET TSR=NOT_INSTALLED
```

然后建立如下的批处理文件 RUNMYTSR.BAT (设内存驻留程序为 MYTSR.COM),这样当键入 RUNMYTSR 来装载内存驻留程序 MYTSR.COM 时,如果它已经被装载过,则该批处理文件将识别此事实并拒绝再次装载,显示内存驻留程序已经装载过的提示信息。具体程序清单如下:

```
@ECHO OFF
IF %TSR%==INSTALLED GOTO NOTICE
MYTSR
SET TSR=INSTALLED
GOTO END
:NOTICE
ECHO TSR PROGRAM IS ALREADY RESIDENT.
:END
```

以上仅给出四个巧用 DOS 环境变量实现的批处理文件实用例程,意在抛砖引玉,祈望能给广大电脑用户一点启迪,从而开发出更多更实用的批处理文件例程。

三、扩充 DOS 环境空间

DOS 对环境空间的隐含限制是 160 个字节,PATH 和 COMSPEC 变量至少占用 29 个字节,剩下的由用户程序使用。如果这个空间不够大,则必须对它进行扩充。上面四个批处理文件实用例程在运行时如出现“Out of environment space”错误信息,就表明环境空间不够,必须加以扩充,具体扩充方法如下:

1. 利用 DOS 的 SHELL 命令(适用于 DOS 3.2 及其以后版本)。

在 DOS 3.2 及其以后版本中,可以利用 SHELL 命令来扩充环境空间,最大可达 32K 字节。具体方法是在 CONFIG.SYS 文件中增添如下行:

```
SHELL=COMMAND.COM/E:XXXXX/P
```

其中,XXXXX 是一个十进制整数,以字节为单位,用来指定环境空间的大小,取值在 160~32768 之间。/P 使得仍然装入 COMMAND.COM 命令处理程序,在它被装入后立即执行 AUTOEXEC.BAT 文件。

2. 利用 DEBUG 修改 COMMAND.COM 文件(适用于 DOS 3.2 以前的版本)

在 DOS 3.2 以前版本的 COMMAND.COM 文件中,有一个字节代表环境空间的大小(以节为单位,1 节=16 字节),隐含值是 0A(即十进制 10)。这个字节在 COMMAND.COM 文件中的偏移量(在 DEBUG 下)是随 DOS 的版本不同而不同的:

DOS 版本	偏移量
DOS 2.0 和 DOS 2.1	ECF
DOS 3.0	F2C
DOS 3.1	D11

这个字节的内容可用 DEBUG 中的 E 命令修改成 0A~3E,即相当于环境空间为 160~992 字节。

DOS 文件的 FCB 管理机制分析

石家庄铁道学院(050043) 崔来堂

本文分析 FCB 的数据结构和有关中断调用;对 DOS 文件的 FCB 管理机制进行较为系统的讨论;给出利用该机制设计磁盘文件拷贝程序的实例。

一、DOS 文件的两种管理方式

由于程序以及数据的绝大部分都是以文件的形式驻留在磁盘上,因此,文件管理功能是 DOS 操作系统的主要内容。在 DOS 2.0 及以后各版本的系统功能调用中,为用户提供了两种文件管理功能:一种是传统方式,一种是高级方式。前者亦称 FCB 方式,它由用户在应用程序中按约定格式设计相应的文件控制块 FCB (File Control Block),并在其中填写待操作的驱动器号、文件名和扩展名等信息;文件建立或打开后,DOS 将向 FCB 中填充其它多项信息,以供对文件实施读写等各种操作。高级方式亦称 Handle 方式。它由用户在应用程序中给出一简练的字符串,此串由待操作文件的路径、文件名和扩展名组成,且以数字 0 结尾。DOS 确认后返回一个相应的文件编号,称为文件句柄(Handle),以后即按此句柄完成对文件的各种操作。

由于 FCB 方式具有 Handle 方式所不可替代的功能。因此本文主要针对 FCB 方式的管理机制及其应用问题进行讨论。

二、FCB 的数据结构分析

1. DOS 文件结构的三要素

在 FCB 方式中,DOS 文件的结构分为文件、记录块和记录三个层次,称结构三要素。一个文件由若干记录块组成,一个记录块包含 128 条记录,一条记录包含若干字节(默认值为 128)。记录是文件读写的最小单位,它按照两种方式进行编号:由记录块号和块内记录号而定位时,称绝对记录号;以文件的首记录为相对

0,向后依次作一维编号时,称相对记录号。

2. FCB 及其数据结构分析

由上述,FCB 是用户应用程序中设计的一个数据块,DOS 对该数据块的结构有严格规定。

就适用性而言,FCB 分为普通的和扩展的两种:前者长 37 字节,适用于普通文件;后者长 44 字节,适用于特殊文件(只读文件、隐含文件、系统文件、卷标、子目录等)。因它是在普通 FCB 的前面增加了一个 7 字节的“扩展头”,故也适用于普通文件。

就应用过程而言,FCB 分为未打开的和打开的两种状态:仅由用户填写了驱动器号、文件名和扩展名,以及特殊文件的属性的 FCB 称为未打开的 FCB;对未打开的 FCB 指定的文件实施了打开或建立操作之后,DOS 自动向 FCB 的字节位移 0CH—24H 处填入有关信息,形成打开的 FCB,以便进而对文件进行读写和关闭等实质性操作。

三、DOS 文件的 FCB 方式读写功能

读写文件是文件管理的主要内容。读写前,需在用户程序中由系统功能调用 1AH 设置磁盘传输区 DTA (Disk Transfer Area),作为一种内存缓冲区,用来存放应用程序与磁盘文件之间所交换的数据。

文件有顺序读写和随机读写之分,前者由 FCB 中的记录块号和当前记录号对记录进行定位,读写后使记录指针自动移向下一条记录;后者由 FCB 中的相对记录号定位记录,读写后记录指针不作变动。

1. 顺序读磁盘文件

步骤如下:

(1)打开文件:功能调用 0FH,格式:

```
MOV AH,0FH
```

```
MOV DX,OFFSET FCB 名字
```

INT 21H

DOS 按未打开的 FCB 指定的文件,在当前目录下搜索,有,则返回 AL=00H,使 FCB 变为打开状态,并对之进行初始化,即以下内容填写 FCB:

驱动器号:实际驱动器号(1=A,2=B,...)

当前记录块号:00H

当前记录号:00H

相对记录号:00H

记录长度:0080H

文件长度、建立日期和时间均按目录登记项中的相应域值填写。

若打开失败,则返回 AL=FFH。

(2)顺序读:功能调用 14H,格式:

MOV AH,14H

MOV DX,OFFSET FCB 名字

INT 21H

DOS 按照打开的 FCB 中的当前记录块号、当前记录号和记录长度,从文件中读出一条记录,并移动记录指针到下一条记录,读出的信息传送到 DTA,并在寄存器 AL 中返回结果的特征标志:

AL=00H:读成功

01H:文件结束,记录中未读到数据

02H:记录长度设置不当,超出所在段

03H:文件结束,记录中的数据不完整

2. 顺序写磁盘文件

步骤如下:

(1)建立文件:功能调用 16H,格式:

MOV AH,16H

MOV DX,OFFSET FCB 名字

INT 21H

DOS 以此功能调用在当前目录下新建一个目录项,文件建立后即被打开,使 FCB 处于打开状态,成功,AL=00H;否则,AL=FFH。

(2)顺序写:功能调用 15H,格式:

MOV AH,15H

MOV DX,OFFSET FCB 名字

INT 21H

DOS 按照打开的 FCB 中指定的当前记录块号、当前记录号和记录长度,向刚建立的新文件或已打开的旧文件,将 DTA 中的数据写入指定记录,并移记录指针到下一条记录,AL 中返回的特征标志与顺序读时相同,唯无 03H。

(3)关闭文件:功能调用 10H,格式:

MOV AH,10H

MOV DX,OFFSET FCB 名字

INT 21H

DOS 在当前目录下搜索指定文件,改写相应目录登记项的有关域,以反映对应 FCB 中的有关信息(如日期、时间和文件长度等)。

3. 随机读写磁盘文件

功能调用号分别为 21H 和 22H,其与顺序读写的区别在于:一是它们以打开的 FCB 中的相对记录号进行定位,二是操作后记录指针不移动。

FCB 的数据结构详如下表:

字节偏移	域内容	字节数	说明
F9H	FF 00 00 00 00 00	6	扩展的 FCB 起点,“扩展头”标志
FFH	文件属性	1	与目录登记项中属性字节的定义同
00H	驱动器号	1	普通的 FCB 起点 0 = 默认驱动器,1=A,2=B,3=C...
01H	文件名或保留设备名	8	不足 8 个字符时填充空格
09H	扩展名	3	不足 3 个字符时填充空格 未打开的 FCB 由用户填写至此
0CH	* 当前记录块号	2	以下各域在文件建立或打开后,由 DOS 自动填写,其中,打 * 号域的值,在文件读写前,可由用户重置。 各数值型域的值,在内存中均符合低字在前,高字在后,以及低字节在前,高字节在后的排列规则。
0EH	* 记录长度	2	
10H	文件长度	4	
14H	建立或最后修改日期	2	
16H	建立或最后修改时间	2	
18H	DOS 保留域	8	
20H	* 当前记录号	1	
21H	* 相对记录号	4	

4. 随机块读写磁盘文件

功能号分别为 27H 和 28H,它们一次可以读写若干条记录,操作前在 CX 中指明欲读写的记录数,操作后 CX 中返回实际读写的记录数,需要注意的是,此时在用户程序中设置的 DTA 应足够大,以能容纳下欲

读写的若干条记录,且应保证 DTA 不致溢出所在段。

读写前,相对记录号可由用户在 FCB 中直接设置,或由功能调用 24H 将 FCB 中的当前记录块号和当前记录号进行转换,然后写入相对记录号域中。

四、磁盘文件拷贝程序的设计

利用 FCB 方式下的文件顺序读写功能,笔者编写了磁盘文件拷贝程序 FCOPY,实现了如下的以及类似的操作:

C>FCOPY A:文件 1 名.扩展名 B:文件 2 名.扩展名

程序的思路是:命令执行时,DOS 先将命令字 FCOPY 之后的两个文件名参数,格式化为未打开的 FCB 格式(驱动器名、文件名和扩展名),分别送入程序段前缀 PSP 的+5CH 域和+6CH 域;进而利用上述 FCB 方式的有关功能调用,打开文件 1,建立文件 2,并以记录为单位,读文件 1 的信息至 DTA,写入文件 2,如此读写循环,直至将文件 1 的信息处理完毕,最后关闭两个文件,完成上述的拷贝工作。

程序中 DTA 容量的选择应当适中,太大,将占用更多的内存空间;太小,会影响拷贝速度,这里选取默认值 128 字节,程序运行后,读者将会发现:若被拷贝文件的字节长度恰是 128 的整数倍,则文件 2 的长度与文件 1 相等;否则,由于读文件 1 的最后一条记录时不足 128 字节,而文件 2 的最后一条记录仍写 128 字节,因此,文件 2 的长度将大于文件 1。如果读写的记录长度均取为 1 时,两个文件的长度将是永远相等的。

为了使问题简化,本程序假定被拷贝文件的长度不大于 64KB,并省略了各种提示信息。

与 DOS 的原内部命令 COPY 不同,本程序经汇编、链接,并转换为 COM 型文件后,将作为外部命令驻留在磁盘上。源程序如下:

;FCB 方式下的磁盘文件拷贝程序

```
code segment
org 100h
assume cs:code,ds:code,es:code
copypy proc near
cld
mov si,5ch ;向 FCB1 传送未打开
mov di,offset fcb1 ;的原文件格式信息
mov cx,0ch
repnz movsb ;
mov si,6ch ;向 FCB2 传送未打开
mov di,offset fcb2 ;的目标文件格式信息
mov cx,0ch
repnz movsb
mov ah,1ah ;设置 DTA
mov dx,offset rwdta
int 21h
mov ah,0fh ;打开文件 1
```

```
mov dx,offset fcb1
int 21h
mov ah,16h ;建立文件 2
mov dx,offset fcb2
int 21h
mov bx,offset fcb1+10h
mov ax,word ptr [bx] ;取文件长度
add ax,127 ;考虑记录长度的零头
mov cl,07h ;除以记录长度
shr ax,cl ;右移 7 位即除以 128
mov cx,ax ;得出记录数
reprw: push cx ;进行拷贝
mov ah,14h
mov dx,offset fcb1
int 21h ;读文件 1
mov ah,15h
mov dx,offset fcb2
int 21h ;写文件 2
pop cx
loop reprw ;记录数减 1,非零则循环
mov ah,10h
mov dx,offset fcb1
int 21h ;关闭文件 1
mov ah,10h
mov dx,offset fcb2
int 21h ;关闭文件 2
mov ah,4ch
int 21h ;退回 DOS
copypy endp
;数据区
fcb1 db 00
db 0bh dup(20h)
db 19h dup(0)
fcb2 db 00
db 0bh dup(20h)
db 19h dup(0)
rwdta db 80h dup(0)
code ends
end copypy
```

五、结束语

FCB 方式不支持磁盘的树形目录,只能在当前目录下进行操作;对 FCB 数据格式的要求比较苛刻,且它的读写方式也较为繁多。但是,当文件建立或打开后,FCB 中的多项重要信息对于用户是直观可知的,特别是,它适用于各种属性的文件,而 Handle 方式对于卷标和子目录则是无能为力的。因此,FCB 方式仍不失为 DOS 文件管理机制中的一种必要的方法。

CMOS 中参数的保存及重置方法

杭州松木场河区 94 号(310007) 金林樵

随着计算机技术的不断向前发展,286 以上档次的微机已成为人们选购的重点对象。在 286 以上档次的微机中,为在关机后继续有效地保存时钟、内存容量、软硬驱动器及显示设备的类型等许多至关重要的参数。使用了一枚靠可充电电池提供能量的 CMOS RAM 芯片(如 MOTORLA MC146818)。

我们知道,一般微机出厂时,厂方已设置好各种重要的参数,用户不必进行重新设置。但为便于用户选择修改,可在开机自检完成后按某一特定组合键(如“CTRL”+“ALT”+“ESC”),或在系统下直接运行 Setup 程序,就可自动进入显示及修改 CMOS RAM 中的配置参数。修改完毕后,选择写入就可将修改后的参数存入 CMOS RAM 中,以便下次开机时根据这些设置的参数决定该机的各种配置。

由于 CMOS RAM 中存有硬盘类型等相当重要的参数,所以,这些参数必须保存完好。一旦丢失这些数据,对于熟悉并了解这些参数的用户还可运行 Setup 程序进行重新设置;但对于不甚了解这些参数实际意义的一般用户,一旦设置错误,就可能造成硬盘上的重要数据丢失。因此,有必要对 CMOS RAM 芯片的结构有一个全面的了解。

CMOS RAM 中包含有实时钟等 64 字节内容。内部实时钟电路使用了前面的 14 个字节,其余用于保存其它各处设备的配置信息。CMOS RAM 的详细地址分配如附表一所示。

对 CMOS RAM 的读写非常方便。它有两个口地址,地址口的口地址为 70H;数据口的口地址为 71H。因此要对 CMOS RAM 进行读写时,首先必须将要读写的 CMOS RAM 地址送到 70H 口,随后就可以从 71H 口中得到所需数据。

具体使用方法如下:键入程序一。当输入 G=102 后,程序一将 CMOS RAM 中的 64 个参数读入 130H 开始的内存中。用户可用 D 130 观察这些数据,并根据需要用 E 命令修改其中的某些参数。

当按程序一中的要求键入完毕后,就在当前目录下生成了一个名为 CMOS.COM 的包含有 CMOS

RAM 中参数的文件。以后一旦系统参数意外丢失后,要恢复这些保存的参数时,只要在系统下运行 CMOS.COM 文件即可。

建立程序一

```
C>DEBUG
-A
4C50:0100 JMP 116
4C50:0102 MOV BX,0 ;将 CMOS 中的
4C50:0105 MOV AL,BL ;64 个参数读入
4C50:0107 OUT 70,AL ;130H 起始的内
4C50:0109 IN AL,71 ;存中
4C50:010B MOV [BX+130],AL
4C50:010F INC BX
4C50:0110 CMP BX,40
4C50:0113 JNZ 105
4C50:0115 INT 3
4C50:0116 MOV BX,10 ;将保存在文件
4C50:0119 MOV AL,BL ;中的 CMOS 参
4C50:011B OUT 70,AL ;数重新写入
4C50:011D MOV AL,[BX+130] ;CMOS
4C50:0121 OUT 71,AL
4C50:0123 INC BX
4C50:0124 CMP BX,40
4C50:0127 JNZ 119
4C50:0129 RET
-G=102 ;读出 CMOS 数据存在 130-16FN 内存中
-R CX ;设置文件长度为 70H 字节
CX 0000 ;内包括读出的 64 个参数
:70
-R BX
BX 0040
:0
-N CMOS. ;将参数保存在 CMOS.COM 文件中
COM
-W ;存盘
-Q ;退出
```


表 一

地址	说明	地 址	说 明
0	秒	10	软盘驱动器类型(位 7-4:A 驱,位 3-0:B 驱)
1	秒报警	11	保 留
2	分	12	硬盘驱动器类型(位 7-4:C 驱,位 3-0:D 驱)
3	分报警	13	保 留
4	时	14	设备字节(软驱数目,显示器类型,协处理器)
5	时报警	15	基本存储器低字节
6	星期	16	基本存储器高字节
7	日	17	扩展存储器低字节
8	月	18	扩展存储器高字节
9	年	19	硬盘类型字节(低于 15 为 0)
A	状态寄存器 A	1A-2D	保 留
B	状态寄存器 B	2E-2F	CMOS 校验和(10-2D 各字节和)
C	状态寄存器 C	30	扩充存储器低字节
D	状态寄存器 D	31	扩充存储器高字节
E	诊断状态字节	32	日期世纪字节
F	停止状态字节	33	信息标志
		34-3F	保 留

获得西山 DOS 五笔字型词组

东北勘测设计院物探公司微机室(130062) 丁 梅

西山 DOS 软件倍受广大计算机文字处理人员的青睐,它提供了五笔字型输入方法,而要实现五笔字型法的高速,离不开词组输入。词组输入的关键是要知道系统中提供了哪些词汇。初学者输入词组只是靠碰运气、试试看,经常是输了词组码而系统中又没有该词组,只能再用单字输入。本来词组输入是提高输入速度的一种方法,但是由于这样一“碰”、一“试”,反而降低了速度。如果你知道系统中存在哪些词组,就不会走弯路了。

本人经过对西山 DOS 的 WBX.COM 的剖析,找到了所有词组的存放位置,用 Pascal 编一小程序将其读出,写入磁盘文件。文件类型为文本文件,可用 WPS 的 N 或 D 功能显示出来,左边小写字母为词组编码。也可将其打印出来,供学习之用。

词组区在 WBX.COM 中的位置是 24562,所有词组按编码第一键分 25 个区,即从 a-y。每个词组只需要对后三键进行编码,词组的存放格式是“编码+词组”,每键码占 5 位二进制位,共 15 位占两字节,第一

字节的高位为 0,做为词组的分界标志,每区以 FFFF 为结束标志。知道了这些规则,用户便可编程序对其原有词组库进行修改或扩充。本文介绍的程序没有词组修改功能,有兴趣者可与本文作者联系。

```

PROGRAM WpsCZ(input,output);
uses crt;
var F:file;
    i,j:Longint;
    d:longint;
    k:integer;
    CZ:text;
    buf,ch:byte;
    a,a1:char;
procedure czbm(ch0:char;b1,b2:byte);
    var ch1,ch2,ch3:byte;
    begin
        ch1:=0;ch2:=0;ch3:=0;
        ch1:=((b1 shr 2)and 31)+96;
        ch2:=((b1 shl 3 )and 24)+((b2 shr 5)and 7)+

```

```

96;
ch3:=(b2 and 31)+96;
gotoxy (1,4);
write (ch0,chr(ch1),chr(ch2),chr(ch3));
write(cz,ch0,chr(ch1),chr(ch2),chr(ch3));
end;
begin
directvideo:=false;
assign(F,'C:$wps$wbx.com');reset(F,1);
assign(CZ,'cz.dat');rewrite(cz);
I:=1;
seek(f,24562);
gotoxy(1,5);
for a:='a' to 'y' do
begin
blockread (F,ch,1,k);
blockread(F,buf,1,k);
czbm(a,ch,buf);
repeat
blockread(F,ch,1,k);

```

```

if ch<128 then
begin
blockread(F,buf,1,k);i:=i+1;
writeln(cz);czbm(a,ch,buf);
gotoxy(1,3);
writeln(I);
gotoxy(1,5);
writeln(' ');
gotoxy(5,4);
end else
begin
if ch<>255 then
begin write(cz,chr(ch));
write(chr(ch));end;
end;
until (ch=255);
blockread(f,ch,1,k);
end;
close(F);close(cz);
end.

```

最小汉字笔画库的设计和使用的

江西拖拉机发动机厂(330044) 黄焕如

在汉字信息处理的实际工作中,经常会遇到汉字按笔画排序或者获取汉字笔画数的问题。

汉字字形远比西文字母的字形复杂,汉字的笔画繁简不一,少至一笔一字,多至三十多画一字,笔画的方向及形状变化也多,因此计算机显示汉字时,通常是把单个汉字离散成网点,每点以一个二进制数表示,这样就组成了该汉字的点阵字模。一般而论,任何汉字处理系统都配备了一个庞大的汉字点阵字模库。

以 CCDOS 为例,其汉字字模库分为一级汉字(3755个)和二级汉字(3008个),前者按拼音排列,后者按部首排列。显然汉字笔画没有在排列中起任何作用,也可以这样说,如果不采取其他措施,要想利用程序直接得到汉字的笔画数几乎是不可能的。

由于设置汉字笔画库是汉字信息处理工作中重要的环节,许多报刊上都登载过汉字笔画库的建立方法,其中许多汉字笔画库是利用 dBASE III 数据库建立的,这类汉字笔画库文件字节数特别大,调用速度慢,适用范围小。笔者在工作实践中找到了一种设置字节数最小的(一个汉字一个字节)汉字笔画库的方法,检索速度快,适用于宏汇编语言、dBASE III 及其他高级语言调用。

该汉字笔画库的设计原理,采用了建立同汉字区位码一一对应的汉字笔画库的办法,具体方法如下:

一、建立汉字笔画库

以现行通用的 CCDOS 为例,首先分区打印出全部汉字,同时将每个汉字的笔画(16进制数)标出,如:

```

      0  1  2  3  4  5  .....
1600 啊 阿 埃 挨 哎
      0A 07 0A 0A 08

```

利用 DEBUG 或其他软件建立汉字笔画信息库,内容就是全部标出的笔画数,例如:

```

C>DEBUG
-E100 0A 07 0A 0A 08.....
-RCX
CX 0000
:1A70 (6763个汉字和5个空字)
-NHQBH(汉字笔画信息库文件名)
-W

```

必须注意的是,如果汉字字模库有空白的汉字位置(例如55区只有89个汉字,有5个空字),必须在汉字笔画信息库相应的位置填写0字节。由于这项关键性工作繁琐并且很容易出错,务必特别仔细。

该汉字笔画信息库的特点是,每一字节(笔画数)均对应于区位码中的一个汉字。由于该库仅仅含有笔画数,一个汉字笔画占一个字节,即使选择一、二级汉字,文件字节数仅仅为6768个字节,可以认为在各种汉字笔画信息库中,该库文件的字节数最小。

二、用程序获取汉字的笔画数

在软件设计中,把汉字笔画信息库当作基本数据库处理,当输入或查找一个汉字时,首先取出该汉字的区位码,根据区位码在汉字笔画信息库中找到相应的地址,取出笔画数即可。

下面以 BASIC 语言为例,给出获取汉字笔画的实例,可供读者参考使用。根据其设计思想,无论是汇编语言、C 语言或者其他高级语言,获取汉字的笔画数,或者根据汉字的笔画数排序都是不困难的,限于篇幅不作进一步的讨论。

```

1  REM BASIC 语言获取汉字笔画数
10 ON ERROR GOTO 150
20 OPEN "HZBH"AS #1 LEN=94
30 FIELD #1,94 AS B$
40 INPUT "输入汉字:";A$
50 IF A$="" THEN 40
60 C1$=MID$(A$,1,1)
70 C2$=MID$(A$,2,1)
80 X=ASC(C1$)-160-15
90 Y=ASC(C2$)-160
100 GET #1,X
110 D$=MID$(B$,Y,1)
120 PRINT "(";A$;")";
130 PRINT "笔画数:";ASC(D$)
140 CLOSE #1;END
150 PRINT "输入的不是汉字!";END

```

三、在 dBASE III 中按笔画排序

在汉字信息管理系统中,经常需要按汉字的笔画排序,如果直接使用 dBASE III 的 SORT 或 INDEX 命令,得到的将是按汉字的 ASCII 码排序的结果。

dBASE III Plus 提供了与汇编语言程序的接口,且调用过程简明,传递参数方便而汇编语言处理数据的速度又是非常之快。用汇编语言编制获取汉字笔画程序,在 dBASE III Plus 下调用是较理想的方法。

如果按上面介绍的方法已经建立了汉字笔画库 HZBH,利用字处理软件编制程序 HZBH.ASM 如下:

```

CODE SEGMENT BYTE
    ASSUME CS:CODE,DS:CODE
HZ PROC FAR
    PUSH BX
    PUSH DS
    MOV AX,CS ;码段地址赋给数据段
    MOV DS,AX
    LEA DX,FILE ;HZBH 汉字笔画库
    MOV AL,0
    MOV AH,3DH ;打开文件
    INT 21H
    MOV BX,AX
    JC ERROR ;文件不存在出错
    MOV WORD PTR[HH],BX
    MOV CX,1A70H ;HZBH 文件字节数
    LEA DX,BUFF ;缓冲区
    MOV AH,3FH ;读文件
    INT 21H
    POP DS
    POP BX
    PUSH BX
    MOV DX,[BX] ;取汉字内码
    SUB DH,0A1H ;位码-161

```

```

    MOV BYTE PTR[WW],DH
    SUB DL,0B0H ;区码-160-16
    MOV AH,0
    MOV AL,DL
    MOV CX,5EH, ;区码×94
    MUL CX
    MOV CL,BYTE PTR[WW]
    MOV CH,0
    ADD AX,CX
    MOV SI,AX ;偏移地址
    PUSH DS
    MOV AX,CS
    MOV DS,AX
    MOV AX,WORD PTR[BUFF+SI]
    POP DS
    MOV [BX],AX
    MOV BX,WORD PTR[HH]
    MOV AH,3EH ;关文件
    INT 21H
    JMP DOS
ERROR:MOV AH,2 ;响铃退出
    MOV DL,07
    INT 21H
DOS: POP BX
    RET
WW DB 0 ;位码
HH DW 0 ;文件句柄
FILE DB' HZBH',0 ;汉字笔画库
BUFF DB 1A70H DUP(?) ;缓冲区
HZ ENDP
CODE ENDS
    END

```

在 DOS 下编译并链接成 EXE 文件,最后转换成 COM 类型文件,具体操作如下:

```

C>MASM HZBH
C>LINK HZBH
C>EXE2BIN HZBH

```

例如,某一数据库 GZHZ.DBF 内含字符性字段 XM(C,8),数字性字段 BH(N,2),要求按 XM 中第一个汉字排序,程序如下:

```

* * * 获取笔画数并按笔画排序 * * *
SET TALK OFF
LOAD HZBH && 装入 HZBH.BIN
USE GZHZ
DO WHILE .NOT. EOF()
    TT=XM && 取第一个汉字
    CALL HZBH WITH TT && 获得该汉字笔画数
    REPLACE BH WITH ASC(TT) && 笔画数送 BH
SKIP
ENDDO
GO TOP
INDEX ON BH TO ABC && 按笔画数索引
LIST && 显示
RETURN

```

背景音乐的原理及其实现

西安交通大学计算机系90461信箱(710049) 丁 塔

背景音乐的特点是当前台程序在运行的同时后台音乐照样演奏。例如,游戏中的音乐就是一种背景音乐。

PC 机上的 DOS 是单用户操作系统,并不支持多道程序的开发,那么如何实现需后台操作的背景音乐呢?我们知道 PC 机上提供了两个时钟中断 INT 8H 和 INT 1CH,并且两者都是每秒执行18.2次。INT 8H 主要用于调整时间等工作,而 INT 1CH 是空的主要供用户使用。时钟中断到来时,CPU 自动中断当前程序的执行而转入到时钟中断处理程序,中断结束后再返回到被中断的程序执行。这样,就为前后台工作提供了可能。本文提出的方法就是通过修改 INT 1CH 来安装背景音乐程序。程序中调用 sound 函数来发音,用变量 mcount 来计中断发生次数,以确定所需延时,从而演奏乐曲的。为了方便编程和引用,程序采用支持中断程序处理的 TurboC 编写并且作成库文件以便其它程序调用。(本程序在 PC 机上通过)源程序如下:

```
#include <string.h>
#include <conio.h>
#include <stdio.h>
#include <dos.h>
#include <stdlib.h>
/* 音符表,共四个音阶,分别用!~&,1~7,a~g,A~G 表示 */
char * mstrtab=" ①!0 $ % ^ & !234567abcdefgABCDEFG
G";
char soundword[300];/* 用于存放歌谱 */
int mswitch,mtimes,mcount;
void interrupt( * oldint1ch);
unsigned fm[28]={132,149,167,177,198,223,250,265,
297,334,354,397,445,500,530,595,667,707,794,891,
1000,1059,1189,1335,1414,1587,1782,2000};
char mch; size_t Len;
int MAXTIMES;/* 用于调节音节延续的时间 */
void interrupt newint1ch()
{int LOCATE;char * CP;/* 用于求音符所对应的频率 */
if (mswitch && mcount<=LEN){/* mswitch 为0时,音乐
停止 */
if(++mtimes>MAXTIMES){
mtimes=0;
mch=soundword[mcount];/* 依次获得每一个音符 */
CP=strchr(mstrtab,mch);
```

```
LOCATE=CP-mstrtab-2;/* 求出该音符频率 */
switch(LOCATE){
case-2:nosound();mtimes=q;break;/* 用于换音 */
case-1:nosound();break;/* 休止符 */
default:sound(fm[LOCATE]);
}
mcount++;
}
}
else nosound();
}
}
void initmusic()
{oldint1ch=getvect(0x1c);/* 保存原 int 1ch 的向量地址 */
setvect(0x1c,newint1ch);/* 安装音乐背景程序 */
MAXTIMES=4;
mswitch=1;
}
void closemusic()
{setvect(0x1c,oldint1ch);/* 恢复原 int1ch */
nosound();
}
void music(char * str)
{LEN=strlen(str);
strcpy(soundword,str);
mswitch=1;
mcount=0;
}
为了生成库文件,首先把本程序的说明部分截下,并在前面冠以 extern,生成一个头文件 music.h;并把本程序做成 music.obj 文件,再用 Turbo 系统的 tlink cs+music 即可。
引用时如下:(歌曲《世上只有妈妈好》)
#include <music.h>
main()
{initmusic();
MAXTIMES=8;/* 1拍约为0.5秒 */
music("66653355aa656666335655321^532222 222355 \
55633321111555321^15555");
getchar();/* 读键,暂停 */
closemusic();
}
```

C—Wordstar 使用基础问与答

新潮计算机集团公司(610051) 朱大公

25. 翻译外科技文献,边翻译边在 C—WordStar 下录入不失为一种很好的方法。翻译中经常遇到的一种情况是,由于对某一专用名词尚无统一译法,当通篇文章译完后发觉某种译法最贴切,应将全篇译名统一。C—WordStar 能方便地实现统一全篇译名吗?

我们知道,这里所提出的任务,如果在手抄情况下完成,除了逐个查找,逐个手工改正外,别无它法。如果让 C—WordStar 来作,可以做得又快、又好,无一遗漏,这就是 C—WordStar 的查找与替换功能,使用的命令为 ^ QF 和 ^ QA。这两个命令的功能比较强,使用灵活,不过操作上也相应复杂一点。在此我们仅就本题的情况,说明 ^ QA 命令的具体操作方法。首先假定待改正的译名在书写上仍是一致的。例如,本应译为“中央处理单元”,现在均书写为“中央处理器”。操作方法分为以下几步:

(1) 首先把光标移至文本头。^ QA 命令的功能是从光标所在处开始,到文本结束为止的范围内查找并替换字符串。为实现查遍整个文本并实施改正,本步是必要的;

(2) 发 ^ QA 命令。此时,C—WordStar 首先要操作者回答待查找的字符串,也就是要被替换的字符串,如上面举例中的“中央处理器”,该字符串称为目标字符串。目标字符串回答完毕以回车结束。紧接着,C—WordStar 问操作者用什么字符串去替换它,如上面举例中的“中央处理单元”,该字符串称为源字符串。回答完源字符串后仍以回车结束。此时系统给出“选择?”这样一个提示;

(3) 击(?)键,屏幕出现五项提示。在此我们选择 N,也就是说输入字符 N 并回车。

此时,整个文本中的目标字符串“中央处理器”都自动地、毫无遗漏地全部被源字符串“中央处理单元”所替换,从而完成全文译名的统一。

26. 查找字符串时,为什么常加长或缩短待查找的字符串?

查找字符串的主要目的是替换或修改该字符串。加长或缩短待查找字符串都是为了准确地、无遗漏地找到所需的字符串。我们先看一下加长待查找字符串的情况。例如,我们以“中央处理器”为源字符串去替换目标字符串“中央处理单元”。实际上,我们的目的是用“器”替换“单元”。如果把目标字符串定为“单元”二字,则由于文本中出现“单元”的可能性远大于出现“中央处理单元”的可能性而造成替换出错。例如,把“内存单元”替换成“内存器”,这当然是不允许的。加长待查找

字符串多在查找并自动替换时使用。下面看一下缩短待查找字符串的情况。这种情况是以增加待查找字符串的数量来实现无遗漏的替换或修改。例如,文本中混用了“扩充内存”和“扩展内存”,此时就可以缩短目标字符串,既不将其定为“扩充内存”,也不将其定为“扩展内存”,而将其定为“内存”二字。当使用 ^ QF 或 ^ QA 命令查找或替换时,既可以找到“扩充内存”,又可以找到“扩展内存”,当然也可以找到“内存”、“内存单元”、“内存容量”……等。那么,到底怎样使用 ^ QF 或 ^ QA 命令来具体实现修改或替换呢?我们以使用 ^ QF 命令为例加以说明。使用 ^ QF 命令后,系统要求操作者回答待查找的字符串,例如“内存”。回答完待查找的字符串后击<ESC>键,注意不要用回车键结束。系统从当前光标位置开始往文件尾执行查找,当找到待查找的字符串时,光标停留在该字符串尾部的下一个字符处。此时,可以在下述三种操作中选择一种:

(1) 在光标当前位置处进行其它操作,如插入、删除、修改等。例如,把当前光标位置前的“扩充内存”修改为“扩展内存”。修改完后用 ^ L 命令继续往下查找;

(2) 不理睬刚才找到的字符串,用 ^ L 命令继续往下查找。例如,保持当前光标前的“扩充内存”不修改,继续往后查找;

(3) 使用 ^ QV 命令使光标返回到前一次使用 ^ L 命令处,为修改字符串提供又一次机会。

通过多次使用 ^ L 命令,总可以查到文件尾,同时系统提示用户已经没有要查找的字符串。

使用 ^ QA 命令实现替换与此类似,仅须注意以下几点:

(1) 回答完目标字符串后以回车结束,而回答完源字符串后以<ESC>键结束;

(2) 找到要替换的字符串时,光标在该字符串首字符处闪动,同时在状态行提示用户是否要用源字符串去替换目标字符串,回答 Y 执行替换,回答 N 则不执行替换。然后像使用 ^ QF 命令一样,可以在三种操作(见前述)中选择一种,直到本命令执行完毕。

上面结合本题所说明的 ^ QF 和 ^ QA 命令的使用方法是这两个命令的简易使用方法。这两个命令还有所谓标准使用方法,请参见第27题。

27. 使用 ^ QF 和 ^ QA 命令时,操作者按照系统要求回答完待查找的字符串(^ QF)或用来替换目标字符串的源字符串后,有时用回车键结束回答,有时又是击<ESC>键结束回答,这是为什么?

^ QF 命令和 ^ QA 命令有所谓简易用法和标准

用法之分。当操作者击<ESC>键结束回答时,便是取简易用法;用回车键结束回答时,便是取标准用法。前面第24题、第25题和第26题中的处理,大多数情况下都是取简易用法,现对标准用法作一些说明。

当用回车键来结束对[^]QF和[^]QA命令的回答时,系统会出现“选择?(?提示)”这样一个提示,它的含义是请你键入一个问号“?”并回车,屏幕上就会出现下一个提示信息。该提示信息包括五项,告诉操作者下一步该怎么做下去,各项含义如下(选择后四个选择项时,应以回车结束):

第一项告诉操作者,如果键入回车键或<ESC>键,就按照简易用法执行[^]QF或[^]QA命令。

第二项为n。它表示此时可以键入一个整数n。n的含义对[^]QA和[^]QF是不同的。对于[^]QA命令,n表示只查找和替换前n次出现的目标字符串。在这n次内,如不作替换操作,可击<ESC>键跳过查找到的字符串,不必像简易用法中那样必须用Y或N回答,然后再用[^]L命令才能继续往下查找和替换。数字n的大小无限制,只要为自然数即可,所以当n很大时,可以方便地完成在整个文本中的查找和替换。对于[^]QF命令,n表示只查找第n次出现的字符串,所以当n很大时,就有可能查不到要找的字符串而文本中又确实有指定的字符串。

第三项为B。它表示此时键入B,则[^]QF或[^]QA命令便从当前光标位置处开始倒着往文本头的方向执行。

第四项为N。它表示执行[^]QA命令时,只要找到一个目标字符串便自动用源字符串去替换它,不必再确认是否要替换。该项对[^]QF命令无意义。

第五项为G。它表示执行[^]QF或[^]QA命令时,将不管光标置于文本何处均从文本头搜索至文本尾。

上述五项,第一项与第二、三、四、五项是互相排斥的,也就是说,如果选择了第一项就不能选择后四项中的任何一项;只要不选择第一项,后四项可以组合使用。例如,选择BG表示让C-WordStar将光标移至文本尾,然后从文本尾到文本头反向搜索;选择NG表示让C-WordStar在整个文本中查找,对于[^]QA命令还表示,如果查找到目标字符串,便自动用源字符串去替换目标字符串,不再需要操作者的人工干预。

28. C-WordStar 中的表格处理是什么含义?怎样实现表格处理?

在笔者的教学过程中发现,初学者往往把表格处理的含义理解为怎样生成中文表格特有的表格线。其实,表格线的生成不属于C-WordStar表处理的功能。下面我们举例说明表格处理功能的真实含义。有一张职工简况表如下所示:

职工简况表

姓名	性别	年龄	文化程度	籍贯	工龄	工资
陈家名	男	35	大专	上海	15	92.50

刘晓义	男	24	大学	成都	2	68.00
赵新秀	女	45	工人	南京	27	128.70

通过观察我们发现这种表格和一般的文本有两点不同:

1. 各栏目下的具体内容所占字符数可能不同。如姓名一栏,遇上复姓复名,则有4个汉字,少数民族的姓名可能更长,因此要留出足够的空间。这样一来,输入3个汉字的姓名时,按通常的方法就要键入多个空格,显得很不方便。

2. 工资一栏为带有小数点的数值,希望小数点位对齐,如不采取措施,每次都数格数,实不可取。

表格文本的这两个特点,实际上只是文本格式上的差别。既然表格也是一种文本,我们不妨把它称为表格文本。在处理表格文本时,应尽可能减少击键次数,并使表格中各栏目的起始列数和数字的小数点位对齐,这就是C-WordStar表格处理功能的真实含义。

为实现表格处理功能,须借助于[^]ON和[^]OI两个命令。下面介绍这两个命令的具体使用方法。

[^]ON命令清除定位点。所谓定位点就是各栏目的起始列和小数点所在列的位置。清除定位点的目的是为了重新设置定位点。使用该命令后,C-WordStar让使用者从下述两种方式中选择一种来清除定位点。其一是按下<ESC>键,清除光标所在处的定位点。此种方法一次只能清除一个定位点,而且发[^]ON命令前应将光标移至要取消的那个定位点所在的列上;也可以先输入一数字n,再按<ESC>键,n列的定位点被取消。其二是击<A>键,取消全部定位点。

[^]OI命令设置定位点。设置定位点前一般先清除所有定位点。发[^]OI命令后,C-WordStar亦会提供两种设置定位点的方法供操作者选择。其一,按下<ESC>键,在光标所在处设置一个定位点;也可以先键入一个数字n,击回车键,第n列便被设置为一个定位点。要设置多个定位点,应多次使用[^]OI命令。其二,键入(<#>)号后再跟一数字n,击回车键,则在第n列设置了一个小数定位点。小数定位点的作用是这样的,在前一个定位点和小数定位点间输入一个数,该数的小数点一定处于小数定位点。

有了上述两个命令便可以进行制表工作了。首先用[^]ON命令消除所有定位点,然后重复使用[^]OI命令设置多个定位点和小数定位点,每一个定位点或小数定位点间的字符数多少按表格栏目相应的宽度设置。输入数据前查看一下状态行的状态指示,如果为INSERT ON,就用一次[^]V命令或按一下<Ins>键,使插入开关断开,其目的是为光标的移动带来方便。此时便可以输入数据了。仍以前述表格为例,先输入表头,以回车键结束。接着输入栏目这一行,输完姓名二字后,按一下<TAB>键(也可以使用[^]I命令),光标会自动跳到下一个定位点,省去了击空格键和对准一个栏目起始列的麻烦,体现出了表格处理的第一种功能。当输入数字时,只要定义了小数点位,数字输入完毕,小

数点便会自动对准小数点位。这就是表格处理的第二种功能。本例中输入工资一栏的数据便属于这种情况。输入完一行数据后的处理分两种情况。第一种情况是表格宽度与右边界一致,此时既可以用回车键结束本行,也可以用 \wedge I命令(或击<TAB>键)结束本行,此时光标会自动跳到下一行行首,实质上就是跳到下一个定位点。第二种情况是右边界比表格宽度宽,此时建议采用击回车键结束本行,如果用 \wedge I命令结束本行,光标跳到下一定位点而不是下一行行首。按照上述方法可以处理整个表格。

29. 使用 \wedge OS命令有时能使屏幕上的文本行间距加宽,有时又不能,原因何在?

这种情况的确存在,但这不是 \wedge OS命令功能不完善,而是使用者对该命令的含义、使用方法和对文本的作用范围还没有完全搞清楚。

使用 \wedge OS命令时应按屏幕提示回答一个具体的数字,其取值范围为1~9,一般取2或3,取2表示两文本行间有一空行,余类推。如果认为数字1表示两文本行间有一空行,而在使用 \wedge OS命令时输入了1,结果必然是文本行间的间距没有拉开。另一种情况是没有搞清楚 \wedge OS对文本的作用范围。当我们把文本录入计算机后,编辑、修改总是免不了的,如果文本行的间距太小,阅读起来会十分困难。例如,在分辨率为720×350的单色显示器上就存在这一问题。为阅读方便起见,应该使用 \wedge OS命令,但在使用 \wedge OS命令后发现文本行的间距并没有加大,其原因在于,一般情况下, \wedge OS命令只对使用该命令后输入的文本有效。为使 \wedge OS命令对以前输入的文本有效,应该使用一次 \wedge B命令。只要注意了上述两点,便不会出现使用 \wedge OS命令后屏幕文本行的间距不改变的情况了。

30. 使用 \wedge OS命令能使打印输出文本的行间距加宽吗?

使用 \wedge OS命令不能使打印输出文本的行间距加宽,也就是说 \wedge OS命令只能改变屏幕上文本的行间距。要想改变打印输出文本的行间距,原则上应借助于汉字打印驱动程序。对于不同的汉字打印驱动程序,调整打印文本行间距的方法各不相同,应参考相应汉字打印驱动程序的使用说明。对于目前流行的新型号打印机,本身自带打印汉字字库,一般不需要专门的汉字打印驱动程序,可以先调整好打印机的行间距,再在C-WordStar下打印,同样可以获得行间距符合要求的打印文本。

31. 怎样调整段落?

我们知道,在C-WordStar中凡出现一个硬回车符,便形成一个段落,同时标志列上显示出“ \langle ”符。调整段落的含义包括合并段落、分割段落和重排段落等内容。如果文本在屏幕上的显示如下所示:

```
××××××××××
××××××作。
为了××××××××××
××××××。
```

根据段落的定义,上面的文本为两个段落。

怎样合并段落。参见上面的文本,把光标移至第二段文本的开头,即“为了××××”的“为”字处,击键,第二段文本便接到第一段文本的句号后。我们可观察到该行上原来的硬回车符“ \langle ”消失,出现符号“+”,“+”号的含义是,本行右边还有内容,由于屏幕宽度有限,这些内容显示不出来,当光标移至这些内容上时,它们会在屏幕上显示出来。为了观察屏幕上两个段落合并成一个段落后的情形,可使用一次 \wedge B命令重排段落。

怎样分割段落。分割段落是合并段落的逆问题,操作正好相反。以上面已经合并了的段落为例,把光标移至“为了”的“为”字下,击回车键,此时该段文本从“为”字处断开直到段尾形成一个新的段落。此时屏幕上可以看到两个硬回车符“ \langle ”,可见一个段落被分割成了两个段落。作分割段落操作前,应检查一下状态行,状态行右端应显示INSERT ON字样,如果没有INSERT ON,应击一次<Ins>键。另外,新段落形成后,应按照汉语的书写习惯,在自然段开始处插入4个空格,然后使用一次 \wedge B命令。

重排段落。重排段落又称为排版,所使用的命令为 \wedge B。重排段落的含义是按左右边界设定的宽度重排文本,使文本排列整齐划一。 \wedge B命令重排段落只对光标所在处起到光标所在的段落结束这一段文本有效。如果整篇文章分成多个段落,要想使用 \wedge B命令重排所有段落,必须对每一个段落都使用一次 \wedge B命令,这种方法的等效操作是使用 \wedge QQ \wedge B命令。 \wedge QQ \wedge B命令可以从光标所在处开始逐个段落地重排,直到文本结束,从而避免了对每个段落发 \wedge B命令。

32. 按照汉语的书写习惯,文章的标题一般都设计为居中放置。屏幕显示时怎样实现标题居中?打印输出时又怎样实现标题居中?

屏幕显示时实现标题居中的方法很简单,首先把光标移至标题所在行,然后使用 \wedge OC命令即可。在大多数情况下我们需要的是打印输出的文本标题居中,而不是屏幕显示时文本标题居中。打印输出时实现文本标题居中分两种情况。第一种情况是,如果标题和正文选择同样大小的字形,则屏幕上居中的标题打印输出时也同样是居中的。第二种情况是,标题使用的字形较正文使用的字形大,此时屏幕上居中的标题打印出来便不居中了。这时应将屏幕上的标题左移,具体移到什么位置可用下面的方法计算。首先作如下一些约定:

1. 设文本左右边界字符数之差为 n_1 。例如左边界(用 \wedge OL命令)为1,右边界(用 \wedge OR命令)为69,则 $n_1 = 69 - 1 = 68$;

2. 文本标题的宽度,包括空格在内,以字符数为单位,记为 n_2 。例如,文本标题有6个汉字,则 $n_2 = 12$;

3. 根据标题所使用字型的横向点阵数和正文所使用字型的横向点阵数确定二者的比值,记为 n_3 。例如,标题使用32×32点阵汉字打印,正文使用24×24点阵汉字打印,标题的横向点阵数为32,正文的横向点阵数

为24,那么 $n_3 = 32/24 = 1.33$;

如果设标题在屏幕上的起始列数为 n ,则

$$n = (n_1 - n_2 \times n_3) / 2$$

计算结果如为小数,可以取整,也可以取整后加1,只要使标题和文本看上去美观就行了。

按照上面的举例,我们可以计算出

$$n = (68 - 12 \times 1.33) / 2 = 26$$

此时,把标题的第一个汉字放到屏幕的第26列,打印出来的标题便是居中的。

33. 怎样使打印输出文本的宽度符合特定的要求?

打印输出文本的宽度取决于三个因素,字型、一行中的字符数和字间距。首先定好字间距和字型,试打印

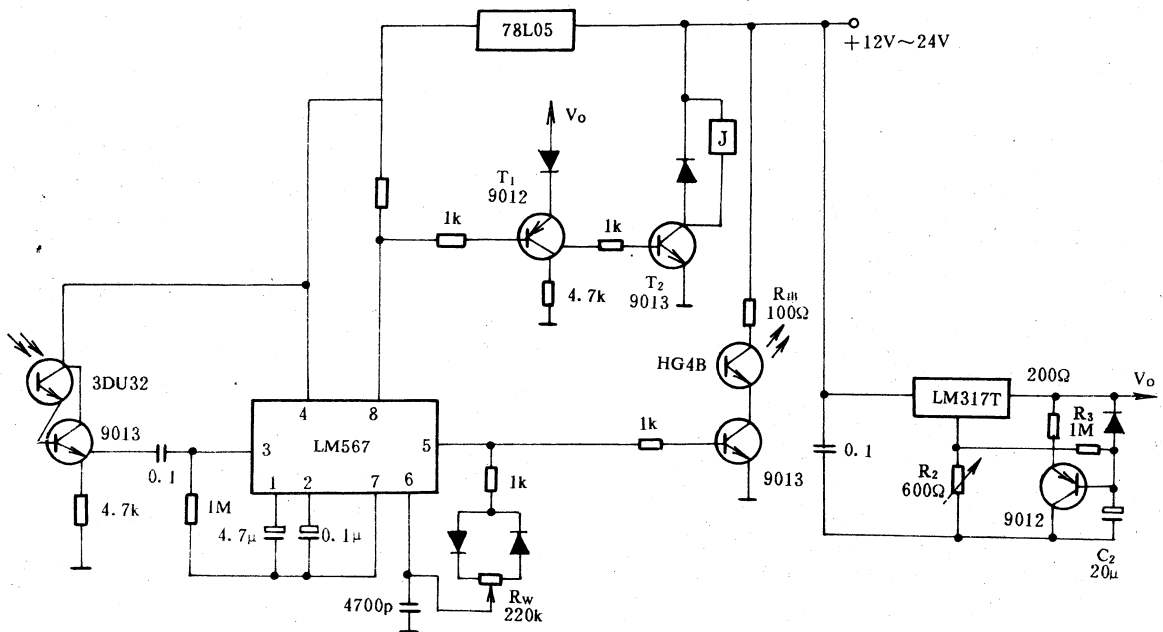
两三行文本,如果对字间距和字型不满意还可以调整,直到满意为止。然后根据文本宽度的要求,确定一行中安排多少汉字。接着用 ^OL 和 ^OR 命令确定屏幕文本宽度,文本宽度应为汉字数的2倍。用 ^OL 和 ^OR 所确定的文本宽度很可能与当前屏幕上的文本宽度不一致,此时便应使用 ^B 命令重新排版。最后打印输出。在上述步骤中,初学者为使屏幕美观往往先使用 ^B 命令排版,但由于字型、字间距的改变,一般情况下均要重新设定屏幕文本宽度,这就导致不得不重新使用 ^B 命令排版。这种不正确的操作次序降低了工作效率,应予克服。

(上接第40页)

发光二极管 HG413 向外发出,再通过一只光敏三极管 3DU32 接收到其输入端脚③。由于接收的信号频率与 f_0 始终相符,因此,③脚输出端为低电平, T_1 、 T_2 管导通,并驱动继电器吸合。

图中 T_1 管的电源由 LM317T 供给,LM317T 构成慢启动稳压电源,目的在于防止开启或关闭电源时冲击电流对继电器触点颤动造成的不利影响。 V_0 的启动速度由时间常数 $R_3 \cdot C_2$ 决定。输出电压则通过改变 R_2

的阻值进行调节,并使 V_0 符合下面情况时方为合适:将发射管与接收管位置对正,LM567 的④脚输出稳定的低电平,这时 T_1 管刚好导通。实验中 V_0 约为 4.2V 左右。 R_w 是为了调节振荡频率的占空比,注意占空比不可过大,以免 R_w 过热烧毁。LM567 ④脚的输出状态以及⑤脚振荡信号的占空比,最好借助于示波器进行观察。实际应用中因发射管,接收管引线较长,需用屏蔽线克服干扰。本电路结构简练。调整容易,且控制效果好,有效距离可达 3m 以上。



计算机病毒的检测、消除和预防

苏民生

四、混合型病毒

1. 新世纪(XqR)病毒

这是既感染 COM 文件和 EXE 文件,又感染硬盘主引导扇区的病毒,兼有文件外壳型和系统型两类病毒特点,可称为“混合型”,指上述两类之混合类型。

病毒程序中的字符内容有“New Century of Computer Now!”,“新世纪”由此得名。病毒表现部分激活时显示给 XqR 的一封信,因此又称为 XqR 病毒。它是一位自称 YaQi 于 1991 年 5 月在长沙制做的“国产病毒”,可用公安部提供的 KILL V46.01 检查和消除。(SCAN 和 CLEAN 高版本,对于侵占 MBR 区的 XqR 病毒可做为 [Azusa] 识别和消除,对于侵占文件的 XqR 病毒尚无法解决。)

使用系统区带病毒硬盘引导系统,或者执行带病毒的 COM 文件或 EXE 文件,都可以使病毒程序进入系统,驻留内存(占 4K),修改系统参数及一些 中断服务程序入口地址,形成病毒环境。

病毒程序首先感染硬盘系统区。在病毒环境下执行 DIR 命令时,被查看的 COM 文件和 EXE 文件被感染。每当执行 COM 文件或 EXE 文件,也被感染。但是,对每个文件只感染一次。

被感染硬盘的前 6 个扇区(以主引导区开始)被病毒程序占用,其中第 2 扇区(0 道 0 面 2 区)存放原来的 MBR。

被感染文件长度增加 3073 至 3104 字节不等,即先将原文件长度增加为 32 整数倍,然后再加上 3072。

XqR 病毒于 5 月 4 日将每个运行的 COM 文件或 EXE 文件都删除,并显示一封致 XqR 的信。此外,星期天开机一段时间后封锁键盘,无法正常工作。

2. Flip 病毒

Flip 病毒也是混合型病毒,感染 COM 文件和 EXE 文件,也感染硬盘系统区。

系统区被感染的硬盘,其隐藏扇区数不小于 6。第 1 扇区,即 MBR(主引导记录)所在扇区,被病毒引导部分占据,MBR 则被移至第 2 扇区,第 3 至 6 扇区则被病毒程序主体部分占据。

被感染的 COM 文件和 EXE 文件,都是在原来文件之后续接病毒部分,并对 COM 文件开头几条指令或 EXE 文件头进行一些修改,使得执行该文件时先要执行附在其后的病毒程序。值得注意的是,对于不同的文件,感染后的病毒部分不大相同,因此有的文献将 Flip 称为“多形性病毒”,即认为它有不同的存在形式。续接的病毒部分,长度一般为 2343(927H)字节,也有

只增大 2153 字节的。

使用被感染的硬盘引导系统,或者执行带病毒的 COM 文件或 EXE 文件,都会使病毒进入系统,驻留内存,形成病毒环境。在病毒环境下执行的 COM 文件或 EXE 文件都会感染病毒,并且只感染一次。如果硬盘未被感染,将会通过磁盘读/写命令首先感染硬盘。

Flip 病毒有时破坏文件分配链表,从而破坏文件;有时会使系统引导失败。

五、病毒的检测方法

发现病毒有主动和被动两种情形。

被动的情形,是指在系统运行中出现异常后,人们才“感觉”到病毒的存在。主动的情形,是指人们有意识地对磁盘或文件进行检查,或对系统运行过程进行监控,设法去识别和发现病毒。

如果比较细致地观察系统运行情况和磁盘存储、读写情况,有时可以发现一些异常,诸如以下情况:

- 屏幕出现预料之外的异常显示,如跳动的小球(小球病毒)、从左向右逐行“吃掉”字符的小虫子(1575/1591病毒)、异常字符或图形等等。

- 在没有安排向磁盘读写数据的情况下,磁盘的指示灯闪亮。多发生在病毒攻击磁盘或文件(试图传染)的情况。

- 执行程序的速度显著变慢。

- 打印机接而不通,发生在 2708 病毒侵入系统时。

- 当用软盘引导系统时,结果却实际上转成由 C 盘启动,而软盘已经插在 A 驱动器上。此时可能 A 盘片已被 GenB 病毒或 E99500 病毒感染,并已传染给 C 盘。

由于病毒的隐蔽性,当人们被动地察觉病毒存在时,它可能已经造成某种程度的破坏,而且它必定已经传染给相当多的磁盘或文件,包括从它实际进入系统至被发觉之日这一段期间使用过的未贴写保护签的软盘。当这些软盘用在其它机器时再传播出去。

因此,现在更加需要主动检查和监测病毒的存在和入侵。这已经成为对机器进行管理的一项日常工作。

检测病毒的依据是病毒的特征,分为静态和动态两种。所谓静态特征主要有以下几方面:

第一,引导区病毒程序占据磁盘 BOOT 区或硬盘主引导区。这两个区的内容,不同的磁盘都是大体相同的。硬盘的分区表,不同的系统各自不同,但主引导程序则只有几个版本,如开头代码为 FA 33 C0...或 FA 2B C0...,而引导区病毒侵入后,0 面 0 道 1 区开头几个字节就被改变,例如变成 EA 05 00...(大麻病毒),E9 4B

00... (2708病毒), EB 68 90... (新世纪病毒), E9 95 00... (暂称为 E99500病毒)。Genp 病毒和1991病毒(暂名)的前3个字节是 FA 33 C0, 和某种 MBR 开头相同, 但在病毒程序中可看到明显标记: Genp 程序1B0至1BD 是字符串 RMBDRMCCQBDWRM, 而1991程序12C至12D 字节代码为9119, 这些都可做为辨认标记。因此, 只要查看硬盘0面0道1区, 就可以及时发现病毒感染。对于软盘和硬盘的 BOOT 区, 也是一样的, 可通过查看该区及时发现病毒。

第二, 文件型病毒使得感染病毒后的文件长度增大, 有的病毒改变文件形成日期, 通过查看文件目录(DIR 命令)可以发现这类变化。对于常用软件, 包括系统启动文件, 都应有一个正常情况下的文件目录表, 以备查看时用于对照。

第三, 文件型病毒程序中总有一些固定代码或字符, 有的病毒还要给感染的文件另加一些标记, 可以通过查看文件中有没有这类特征或标记, 来检查文件是否感染病毒。例如, 感染以色列病毒的 COM 文件, 最后5字节被标记为 MSDOS, 1575/1591病毒则将被感染 COM 文件最后3字节加上标记代码43 0C 0A, 而对 EXE 文件所加标记为45 0C 0A。

第四, 针对 D2这类系统型病毒特征, 可以查看文件目录项, 包括根目录和子目录。如果发现某些文件目录项1A至1B 字节内容相同, 而本应为0000的15至16 字节内容已不是全0, 那么这张盘或说这些文件已被 D2病毒感染。

利用上述病毒静态特征进行检查, 可以用 DEBUG、PC Tools、NU (Norton Utilities) 等通用软件工具, 也可以依照这些原理编写专门的程序。病毒诊断程序通常都和消除程序编在一起, 有的是专门针对某一种病毒, 更多的则是针对一大批病毒, 或者是包括全部“已知病毒”。

但是, 所谓“已知病毒”不能包括新出现的病毒。因此, 针对静态特征进行检查, 对于新出现的病毒是无能为力的。病毒的发展很快, 不断出现“新品种”, 而且有些老病毒稍加修改(改变已知特征和表现方式)就成为与原来不同的“变体”, 如果仅仅跟在新出现病毒之后编写检测程序, 就永远比病毒的发展落后一步。已经普遍使用的 SCAN/CLEAN 软件1992年3月的版本 8.4B89宣称可检测/消除534种病毒, 如包含变体则有1263种, 而经过不到半年时间, 1992年8月的版本 8.7B95则宣称可检测/消除的病毒已达685种, 包含变体则为1401种。任何稍有能力的人在原有病毒基础上的小小的“改进”, 都可写出“未知病毒”, 因而, 可使编写检测/消除病毒程序的人疲于奔命。因此, 虽然上述

方法是基本的, 特别对于消除文件病毒是必须的, 但不能仅仅依靠这种方法。一些著名的检测/消除软件, 以及一些防病毒卡, 在检测时也都针对病毒的动态特征进行。所谓“动态特征”包括以下几方面:

第一, 病毒进入系统时, 通常要改变用户可用内存大小, 以便将病毒程序驻留内存。

第二, 病毒程序通常都要改变一些常用中断服务程序的入口地址, 如系统计时中断 INT 8, 磁盘服务中断 INT 13H, DOS 中断 INT 21H 等, 以便拦截中断服务程序, 即在运行程序中在中断服务程序之前“插入”一段病毒程序。

第三, 病毒程序在其活动过程中, 还可能改变某些应当由操作系统进行的设置, 从而产生异常。

第四, 病毒程序进行传染时, 试图改变不应改变的磁盘内容或文件内容, 例如系统病毒试图侵入磁盘引导区或硬磁盘主引导区, 有些文件病毒要将自身附在刚刚读入的文件上并试图把感染后的内容写回到文件中。

针对以上动态特征, 可以用特定程序监控系统运行, 随时检查是否发生与病毒特征有关的异常, 一旦发现, 立即发出警告, 或者提请用户根据具体情节进行处理, 或者不执行可疑程序。一些软件系统提供的 BOOTS SAFE 功能用于保护/监控磁盘系统区, 对于文件(COM、EXE 等)的保护/监控也有适当的程序来实现。它们不仅可以检测到已知病毒, 也可以检测到未知病毒, 对于这两种情形的差别在于: 对于已知病毒, 可以进一步指出它是何种病毒, 并进而消除它, 而对于未知病毒, 则无法提供更多的信息。

防病毒卡是将检测/消除病毒软件固化而产生的, 依据的是相同的原理。和检测/消除软件不断进行版本更新一样, 防病毒卡也在不断改进或升级, 才能应付病毒发展的新的挑战。

无论是用通用软件工具(DEBUG、PC Tools、NU 等)还是用专门软件检测病毒, 都要求在“清洁的”环境中进行。不少病毒为了隐蔽自己, 都采用了特殊手段, 使得在病毒环境下不能检测出病毒。由于硬盘很容易被攻击, 通常都需要备份一张带系统的好的软盘, 可用它引导系统, 并将一些必要的数据存放在这张盘上, 如硬盘主引导区内容(特别是分区表)和硬盘每个分区的引导区(BOOT)内容。这张盘制做好之后应立即贴上写保护签, 保证它不被任何病毒感染。工具软件和专门的检测、消除病毒软件, 为了使用方便, 通常可以放在硬盘特定子目录中, 但为了安全起见, 都应当备份在软盘上并加写保护。

(未完待续)

CEC— I 中华学习机汉字系统简介

辽宁省新宾县永陵林场(113206) 林永春

本刊93年4期中,王志超同志撰文详细地介绍了CEC— I 中华学习机的系统程序,给读者阅读和使用这些系统程序带来了方便。在中华学习机中还固化有汉字系统程序,该文没有介绍。由于CEC— I 中华学习机带有硬汉字库,汉字编码规范,用于开发汉字应用软件比同等级其它机型更为便利,所以有必要多了解一些机上的汉字系统程序。现将笔者对汉字系统程序分析了解的一部分汉字系统程序的功能和使用方法介绍给读者,希望它给读者开发CEC— I 中华学习机的汉字功能带来帮助。

一、CEC— I 汉字系统使用方法

CEC— I 中华学习机的汉字系统程序固化在辅存\$EC00—\$FFE5区间,调用时要通过位于\$C300处的汉字接口与驻机的其它系统程序连接。\$C300处主辅存转换的子程序有四个:

- \$C3A4 在辅存\$200—\$BFFF区间读操作,位于这部分的汉字系统有汉字点阵字库和拼音码表。
- \$C3AB 在辅存\$D000—\$FFFF区间读操作。位于这部分的是汉字系统程序。
- \$C3B9 读写操作返回主存\$D000—\$FFFF区间
- \$C3B2 读写操作返回主存\$200—\$BFFF区间
调用汉字系统子程序应先用JSR \$C3AB切换辅存ROM。调用结束后再用JSR \$C3B9返回主存ROM。

二、CEC— I 汉字系统子程序简介

在下面的子程序中,地址后面标有“#”号的子程序为对应《中华学习机CEC— I 技术参考手册(软件)》(清华大学出版社)第四部分附录A中的各系统程序框图,请读者在阅读汉字系统程序时参照。

- \$EC00 第一次进入汉字状态时的初始化子程序,其功能是:清高分辨率图形显示第二页;设置汉字系统输入/输出入口;显示标题;设置打印机初值。
- \$ECCD 取出显示代码寄存器(\$D6—\$D7)中代码对应的字符点阵送入字符点阵暂存区(\$94D0—\$94F1)。
- \$EDD1 将系统当前状态(字母、拼音、区位、用户扩充)显示到屏幕第11行上。
- \$EE77 显示键盘输入的状态提示字符,即F1—F5所定义的状态字符。
- \$EE9F 在第11行左侧显示二个汉字和冒号。
- \$EF7D 将学习机汉字内码转换为异形国标码。
- \$EFAC 将异形国际码转换为学习机汉字内码。
- \$EFCE# 输出处理程序(CSW)。完成字符、汉字和控制

- \$EFE7 命令码的输出,代码要放入累加器A中。汉字必需用三字节的机内码。
- \$F068# 输出处理程序。功能与\$EFCE相同,要求汉字用二字节异形国标码,字母显示码的最高位必须为0。
- \$F09E# 汉字字符显示程序。
- \$F0AD ASCII字符显示程序。
- \$F0B5 将字符点阵暂存区中的ASCII码送图形显示区显示。
- \$F123 将字符点阵暂存区中的汉字图形送图形显示区显示。
- \$F182 翻滚屏幕。
- \$F18D 将字符点阵暂存区清零。
- \$F1A6 将ASCII码送入屏幕映射区。
- \$F1E0 将ASCII码送入屏幕映射区。
- \$F29E# 计算屏幕座标地址。
- \$F2F3 控制字符处理入口。
- \$F30D 执行控制字符[CTRL—G],使扬声器叫一声。
- \$F32C 执行控制字符[CTRL—H;即←]。光标退一格。
- \$F34D 执行控制字符[CTRL—K]。从光标处开始清屏至页末。
- \$F369 执行控制字符[CTRL—L]。清屏幕,光标移到左上角。
- \$F36E 设置输出为正相显示方式。
- \$F373 设置输出为反相显示方式。
- \$F37F 显示或清除状态提示字符。
- \$F388 暂停输出,按任意键恢复输出。
- \$F3C7 从光标处清屏至行末。
- \$F3F1 输出回车符,光标移至下行首。
- \$F45D# 屏幕翻滚处理。
- \$F782 打印处理程序。
- \$F7A3# 产生光标,等待按键。
- \$F7EF# 输入处理程序(KSW)。
- \$F846 控制键处理入口。
- \$F8D0 屏幕编辑入口。功能同系统行编辑子程序。
- \$F8FB 字母、拼音、区位状态转换。
- \$F90B F4、F5键处理程序。通常用于用户扩充功能。
- \$F935 状态显示处理。
- \$FA62# 删除光标处字符,光标退一格。
- \$FADD# 区位方式处理。
- \$FB28 拼音方式处理。
- \$FF0C 将选中的提示字送入系统区。
- \$FF25 显示下一幕汉字提示。
- \$FF8A 显示前一幕汉字提示。
- \$FF91 显示标点符号提示。
- \$FF98 显示算术符号提示。
- \$FFB8 显示制表符符号提示。
- \$FFB8 清除拼音提示。

多功能标准化学生成绩统计程序

襄樊铁路运输技校(441001) 汤永进

学生成绩的统计分析是教育部门计算机的一大用武之地。近年来随着教育统计学理论的深入普及,标准差、标准分数等新型统计项目逐渐成为常规(国家教委已正式下文今后高考将以标准分划线录取),过去公开发表过的一些较简单的统计程序已不能满足需要。笔者特将经我校长期实践精练而成的一个程序推出与同行们共享。

该程序的主要特色是:

(1)功能多而强,既能统计单项成绩(满分不限百分),又能汇总多项成绩(可任意自定比例进行),还可随时复查已统结果,较全面地包括了成绩统计的基本科目。在具体统计项目中,除传统的总分、平均分、分数段、各项比率和排名等外,还增加了标准差、标准分的统计和成绩分布直方图的显示,适应了不同需要。

(2)操作简便,每班仅首次遇及需逐人输入姓名(名单随即存盘),以后的统计均自动由盘读取名单,操作者照单输数即可。整个过程均以人机对话进行,数据输入时自动根据需要转换字母或拼音状态,分数输错允许连续回退,良好的界面和防误性可使非专业人员也易运用自如。

(3)输出悦目,表格精美,各项数据均按小数点对齐或右对齐工整排列,显示时分段自动(亦可随时手动)暂停待键以便操作者驻看。本程序还特别兼顾屏显与纸印效果,二者版面完全一致,即使不接打印机也可从屏幕上观看到整齐的表列。

(4)程序精悍,篇幅显著小于一些具同等功能的文件组合软件,占用内外存少而运行效率高。特别是采用了近年来为人瞩目的新颖映射排序(由笔者进行了映域浮动、数组兼能等克服其弱点的重大改进),因无交换大大加快了统计速度,对于一个普通大小的班级当原始分数为整数时,其数据输入后的运算等待时间仅约数秒!

下面是对该程序基本功能模块的说明:

10~90行为系统的启动和菜单总控。

100~190行为统计单位(班或其它集体)名单的读取或建立(盘上无名单时)。其中140行定义的数组 M\$(N)放学生姓名、F(N)放原始分数、BF(N)放标准分数、H\$(N)放序号为 N 的学生在排序后的新序号, P\$(X)初放分数为 X 的学生的序号、排序后放该生的名次。

200~280行为单项成绩的输入,注意输项目名时包括考项和科项(例“93二期末数学”)。程序中的变量 D 是精度标志(若原始分出现小数点时设为 1),控制排序时灵活配用数组空间,满分 M 最大可达 P\$数组下

标量的定义值。

300~390行为总计多项成绩时的数据读取与加和。有关各单项成绩须先统好在盘,如缺项会给出提示自动返回。

400~460行统计总分 ZF、人平分 PF、分数段 D(I)及各段百分率 L(I),其中下标为 0 和 1 者分别放及格、优秀人数及其百分率。

470~490行为排序(调子程进行)、统计标准差 BC 和标准分 BF(I)。标准分取值形式采用基本式(以零为平均线),如本地区推行的是非负修正式则可在 490 行的循环中给 BF(I)加上该式的修正常数。

500~540行为数据存盘。所有的原始数据及除排序结果外的统计后数据均存入,存后跳往 600 输出。

550~590行为查阅已统成绩时的数据读取及排序。采取即时排序是因本程序排序速度奇快、且其记序数组大而离散,若存之反而耗费时空。

600~880行输出统计结果,可反复进行。遇暂停按任意键即继续,输出时按任意键则暂停。

900~930行为输入状态自动转换子程序,调 900 转拼音,调 910 转字母。

1000~1100行为排序子程序。变量 DD 为一系数,可将带小数的原始分数变换为整数,变换后映射为 P\$数组的下标,其数组值取为该分学生序号(按 3 位等长放入,同分者串加),然后进行整理,将已按名次成序但间断存放于 P\$数组中的序号转送 H\$数组连续化,同时产生确切名次回送已腾空的 P\$数组。该改进的映射排序法较之原来具有可处理小数且使用数组少等特点(若只获得名次而不要求输出按名次排列则 H\$数组还可取消),效率极高。

本程序为便于发表精简了一些为输入高度自动化服务的冗长程序段,有兴趣的读者可根据本校的具体设置,将班名、项名等制作成 DATA 数据表放入程序(或以文本文件独自放盘),运行时自动读出展列供输入选择,可使操作更加简易。

该程序适用于 CEC-I 系列中华学习机。稍加改动(如开关打印机的命令)即可用于 Apple 兼容机的软汉字等系统。

本文作者还有适用于编辑稿件、信函的工具软件《CEC 文稿书信超级打印室》,需要者请与作者联系。

```
10 D$=CHR$(4);G$=CHR$(7)+CHR$(7);S$=CHR$(19)
15 OP$=D$+"OPEN";RD$=D$+"READ";WT$=D$+"WRITE";CL$=D$+"CLOSE"
20 A$(0)="-";A$(1)=" ";A$(2)="+":A$(4)
```

```

=“0”;P=49152;PP=49168
30 PRINT D$;“PR#3”;PRINT;HGR2;VTAB 1
40 PRINT TAB(5);“《标准化学生成绩统计系统》”;PRINT
50 PRINT “*****请选择*****
***”
51 PRINT TAB(10);“(A)单项成绩统计”
52 PRINT TAB(10);“(B)多项成绩总计”
53 PRINT TAB(10);“(C)已统成绩查阅”;PRINT
55 PRINT TAB(5);“(按 A/B/C 进入,Esc 键退出)”;G$
60 E=PEEK(P);IF E<128 THEN 60
70 POKE PP,0; IF E = 155 THEN HOME:PRINT“再
见!”;G$;END
80 E=E-192;IF E<1 OR E>3 THEN 60
90 HOME :PRINT G$
100 GOSUB 900;INPUT“单位名称:”;W$
105 POKE 222,255;ONERR GOTO 120
110 PRINT OP$;W$;PRINT RD$;W$;INPUT N
120 POKE 216,0;R=PEEK(222)<9;ON R=0 GOTO
140;PRINT CL$
130 GOSUB 910;INPUT“人数:”;N
135 GOSUB 900;PRINT“各人姓名:”
140 DIM M$(N),F(N),BF(N),H$(N),P$(3200-N
*5)
150 FOR I=1 TO N
155 IF R=1 THEN PRINT I;“:”;
160 INPUT“”;M$(I);NEXT
170 IF R=0 THEN PRINT CL$:GOTO 200
180 PRINT OP$;W$;PRINT WT$;W$;PRINT N
190 FOR I=1 TO N;PRINT M$(I);NEXT;PRINT CL
$
200 GOSUB 900;INPUT“项目名称:”;M$;T$=W$+
M$
210 GOSUB 910;ON E GOTO 220,300,550
220 INPUT“满分数:”;M;M=M/100
230 PRINT“各人分数:(输错可空按回车退回)”
240 FOR I=1 TO N
250 PRINT I;“:”;M$(I);:INPUT“:”;X$
260 IF X$=“”THEN I=I-1+(I=1);CALL -998;GO-
TO 250
270 F(I)=VAL(X$)
275 IF F(I)<>INT(F(I)) THEN D=1
280 NEXT;GOTO 400
300 INPUT“总计项数:”;S;DIM X$(S),B(S)
305 PRINT“是否比例加和?(Y/N)”;GET Y$;PRINT Y
$
310 PRINT“各项名称:”
315 FOR I=1 TO S;GOSUB 900
320 PRINT I;:INPUT“:”;X$(I)
325 IF Y$<>“Y” THEN B(I)=1;GOTO 340
330 GOSUB 910;INPUT“该项比例(%)”;B(I);B(I)=B
(I)/100
340 NEXT;ONERR GOTO 390
350 FOR I=1 TO S
355 PRINT OP$;W$;X$(I);PRINT RD$;W$;X
$(I);INPUT X;INPUT Y
360 M=M+X*B(I);IF Y=1 THEN D=1

```

```

365 FOR Y=1 TO N
370 INPUT X;F(Y)=F(Y)+X*B(I)
375 INPUT X;BF(Y)=BF(Y)+X*B(I)
380 NEXT;PRINT CL$;NEXT;GOTO 400
390 PRINT;PRINT X$(I);“数据缺,请先进行其单项统
计!”;G$;FOR I=1 TO 2000;NEXT;RUN
400 HGR2;VTAB 2;PRINT“机内正在统计,请稍候……”;
G$
410 FOR I=1 TO N
420 F(I)=INT(F(I)*10+.5)/10;ZF=ZF+F(I)
425 IF E=2 THEN BF(I)=INT(BF(I)*100+.5)/100
430 X=F(I)/M/10;D(X)=D(X)+1;NEXT
440 PF=INT(ZF/N*10+.5)/10
450 D(2)=D(2)+D(1)+D(0);D(9)=D(9)+D(10);D
(1)=D(8)+D(9);D(0)=D(6)+D(7)+D(1)
460 FOR I=0 TO 10;L(I)=INT(D(I)/N*1000+.5)/
10;NEXT
470 GOSUB 1000;Y=0
480 FOR I=1 TO N;X=F(I)-PF;Y=Y+X*X;NEXT
485 BC=INT(SQR(Y/N)*100+.5)/100
490 IF E=1 THEN FOR I=1 TO N;BF(I)=INT((F(I)-
PF)/BC*100+.5)/100;NEXT
500 PRINT“统计完毕!数据存盘(按一键开始)”;G$;:
GET X$;PRINT
510 PRINT OP$;T$;PRINT WT$;T$
515 PRINT M;PRINT D
520 FOR I=1 TO N;PRINT F(I);PRINT BF(I);NEXT
525 PRINT ZF;PRINT PF;PRINT BC
530 FOR I=0 TO 10;PRINT D(I);PRINT L(I);NEXT
540 PRINT CL$;GOTO 600
550 PRINT“放好磁盘,读取数据(按一键开始)”;G$;:
GET X$;PRINT
560 PRINT OP$;T$;PRINT RD$;T$
565 INPUT M;INPUT D
570 FOR I=1 TO N;INPUT F(I);INPUT BF(I);NEXT
575 INPUT ZF;INPUT PF;INPUT BC
580 FOR I=0 TO 10;INPUT D(I);INPUT L(I);NEXT;
PRINT CL$
590 PRINT“正排序,请稍候……”;GOSUB 1000
600 PRINT“统计结果公布如下:”;G$;PRINT
610 POKE 1915,3;POKE 1659,1
620 PRINT TAB(15-LEN(T$)/3);T$;“成绩”:
PRINT
630 PRINT“^^^^^^^^(1)总体成绩分析表^^^
^^^^^^”;PRINT
640 PRINT“统计人数: 总分: 人平分分数: 标准差:”
650 PRINT TAB(4);N;TAB(11);ZF;TAB(19);PF;
TAB(28);BC
660 PRINT“优秀人数:优秀率: 及格人数:及格率:”
670 PRINT TAB(4);D(1);TAB(11);L(1);“%”;TAB
(21);D(0);TAB(28);L(0);“%”;S$
680 IF M<>1 THEN PRINT“(分数段系折算为百分之统
计)”
690 PRINT“分数段: 人数:比率 :直方图(5%/#):”
700 FOR I=9 TO 2 STEP -1
710 PRINT SPC(I=2);I*10*(I>2);“~”;I*10+9+

```

```

(1=9);“:”;
720 PRINT TAB (10+(D(I)<10));D(I);TAB(14+(L
(I)<10));L(I);TAB(18)“% ”;
730 IF L(I)>2.4 THEN FOR Y=0 TO L(I)/5-0.5;
PRINT “#”;NEXT
740 PRINT;NEXT I;PRINT S$;G$;POKE PP,0
750 PRINT“^^^^^^^^(2)个人成绩排列表^^^^
^^^^^^^^”;PRINT
760 PRINT“1.按统计顺序排列: I按名次排列:”
770 PRINT“名次-姓名-分数-标分 名次-姓名-分数”
780 FOR I=1 TO N
790 X= F(I) * DD;PRINT TAB(4-LEN(P$(X)));P
$(X);
800 PRINT TAB(5-(LEN(M$(I))>9));M$(I);TAB
(12-D);F(I);
810 X= ABS(BF(I));PRINT TAB(15);A$(SGN(BF
(I))+1);A$(3+(X>0 AND X<1));X;
820 X=VAL(H$(I));Y=F(X) * DD;PRINT TAB(24
-LEN(P$(Y)));P$(Y);
830 PRINT TAB(25-(LEN(M$(X))>9));M$(X);
TAB(32-D);F(X);

```

```

840 IF PEEK(P)>127 THEN PRINT S$;POKE PP,0
850 PRINT;NEXT;PRINT
860 POKE 1659,0;PRINT“再显印一遍?(Y/N)”;G$
870 GET X$;PRINT;IF X$=“Y” THEN 600
880 RUN
900 POKE 942,140;GOTO 920
910 POKE 942,129
920 PRINT CHR$(18)CHR$(18);
930 RETURN
1000 DD=(1+9*D)/(1+4*(M>3)*D)
1010 FOR X=1 TO N;Y=F(X) * DD
1020 P$(Y)=P$(Y)+RIGHT$(STR$(1000+X),3)
1030 NEXT;K=1
1040 FOR Y=100 * M * DD TO 0 STEP -1
1050 ON P$(Y)=“”GOTO 1090;L=LEN(P$(Y))
1060 IF L<4 THEN H$(K)=P$(Y);GOTO 1080
1070 FOR X=1 TO L STEP 3;H$(K+X/3)=MID$(
$(Y),X,3);NEXT
1080 P$(Y)=STR$(K);K=K+L/3
1090 NEXT;HGR2
1100 RETURN

```

ProDOS 系统内部结构剖析

北京铁路局卫生防疫站(100038) 廖 凯

三、综合技术

这部分将描述一个系统通常做什么事,并为完成它们提供某些技术。

(一)确定机器配置

对一个系统程序而言知道正在运行中的 Apple I 的型号常是有用的。在系统整体页面内的 MACHID 字节标识机器类型、存储空间和是否有一个 80 列文字卡或 Thunderclock 卡。

有两个位(bit)区分 Apple I, Apple Ie 或 Apple II 仿真模式。这区别对两件事最有用:

1. Apple Ie 具有小写功能。屏幕信息可以用大写和小写显示,如果机器不是 Apple Ie(或者它是无 80 列文字卡的 Apple Ie),则变为大写。

2. Apple Ie 有其它的 Apple 机不具有的键(最明显的是箭头键和 DELETE 键)。在所运行的系统中,软件能够对这些键进行使用,并且屏幕信息也应作相应调整。

存储容量

机器可能存在的存储容量是 48K、64K 和 128K。一个系统程序在确定移动它自身时使用这些值。注意辅助 64K 区域不能包含 MLI 调用的代码。

80 列文字卡

若 80 列卡按照 UCSD Pascal 规定设置在槽口 3, 则此位(bit)被设置。此规定允许许多 80 列卡(包含 Ap-

ple Ie 的)的功能可以经一个 JSR \$C300 来接通。

80 列卡可以用下面的指令来关闭:

```

LDA # $15 ;关闭视频固件的
JSR $C300 ;打印它到视频固件

```

控制 Apple Ie 显示的这软开关在使用普通的 Apple I 时无效。它们在 Apple Ie 参考手册中有解释。在监控 ROM 内的 RESET 子程序(\$FF59)内的四个子程序被所有的 Apple I 机型所使用。

(二)使用日期

系统程序经常使用当前的日期:用一个修改的日期标记文件,在列表上作为标记,或在屏幕上显示。无论什么用途,总是希望获得最近的设定。

保存系统日期和时间单元(\$BF90-\$BF93)为以后使用,然后清除它们。接着使用 GET-TIME 调用。如果有一个时钟/日历卡一个安装的计时子程序,则系统日期和时间单元将为非零,这是你应使用的日期和时间。若 GET-TIME 调用无效,你或者使用上述的值,或者提示用户给出当前的日期和时间。在系统启动时,日期和时间单元设为 0。

如果没有系统时间并调用 GET-TIME 无结果,则使用 GET-FILE-INFO 调用并用最后修改的日期和时间作为系统设定。若用户修改时间,要放置这些值到系统日期和时间单元,则一个 SET-FILE-INFO 调用将修改时间给下一个 GET-FILE-INFO。

(三) 系统程序系统设定值

在一个目录内的每个文件项目有一个两字节的辅助类型字型。这字段包含二进制文件的载入地址或文本文件的记录长度的信息。对于系统文件不使用它。若你的系统程序有一个少量的系统设定信息,你想从程序的一个执行保存到下一个,则此字段是一个储存它的好地方。

要改变此字段的内容,使用 GET-FILE-INFO 调用读取文件的项目用相同的参数表保存修改的值到文件的项目内。

(四) 查找一个卷

由于不总是知道所有联机的卷的名字,用户有时需要用槽口及驱动器号代替卷名来指定卷。在存取 ProDOS 文件之前,必须把槽口和驱动器号转换为一个卷名,用户可以按以下步骤进行:

1. 把槽口及驱动器号装入设备号(unit_num)。此号用于指定由 ON-LINE 调用所需的设备。

2. 在 ON-LINE 调用内使用设备号。此调用将返回一个带有计数字节的卷名。此卷名前面没有斜线。你在其它 ProDOS 调用使用此名时,必须在卷名前插入一个斜线将计数字节加1。

(五) 使用 RESET 向量

在 Apple II 内,按下 CTRL-RESET 键将无条件转到 RESET 向量(在内存 \$3F2处)。用户可以在任何时候按 CTRL-RESET,但当文件被打开时按了 CTRL-RESET 键,ProDOS 不能保证磁盘的完整性。

用户的程序应该在 RESET 向量单元内放置一个显示劝告用户关闭任何打开的文件的信息的子程序的入口地址,然后关闭文件。

四、ProDOS 系统程序规范

为了使软件具有兼容性,ProDOS 系统程序有以下规定:

1. 尽可能使用相同的术语。若你的实用程序执行由 BASIC 系统程序、FILER、CONVERT 程序、或 EDITOR/ASSEMBLER 所使用的功能,则用相同的术语。

2. 在所有显示文件列表的软件内使用相同的目录格式,没必要执行40列和80列两种格式。

若你决定执行自己的目录格式,就要识别文件类型并显示3字母缩写词。

3. 标准的 Apple II 警声已被柔和音调所代替。用户可以利用它编个音乐程序,下面是它的代码:

```

SPKP EQU $C030;使扬声器发声
*
LENGTH DS ; ;音调宽度
*
* 这是监控 ROM 的延迟子程序
*
WAIT SEC
WAIT2 PHA
WAIT3 SBC #1
BNE WAIT3
    
```

```

PLA
SBC #1
BNE WAIT2
RTS
    
```

```

*
* 产生一个动听的小音调
* 退出时设置 Z-标记(BEQ)作为转移
* 破坏累加器的内容
*
    
```

```

BELL LDZ # $20;音调宽度
STA LENGTH
BELL LDA # $2 ;短延时,发声
JSR WAIT
STA SPKR
LDA # $24 ;长延时,发声
JSR WAIT
STA SPKR
DEC LENGTH
BNE BELL1 ;重复 LENGTH 次
RTS
    
```

第七章 增加例程到 ProDOS

本章介绍可以与 ProDOS MLI 一起使用的各种例程的规范和特性。这些程序被连接到 MLI 并与 MLI 相互作用。在此章介绍的例程有时钟/计时例程、中断处理例程和磁盘驱动例程。

一、时钟/计时例程

许多 MLI 调用具有日期和时间参数,这些参数可用于标注目录标题和文件项目。每当需要建立或修改时间时,MLI 按照以下步骤进行:

1. 若调用的参数表内日期或时间字段非零,则 MLI 使用指定的值。在其它情况下,MLI 使用 DATETIME(\$BF06)调用。在这单元开始的3个字节包含一个跳转到时钟/日历程序的 JMP 指令。根据系统设定,它们包含跳转到一个已知 RTS 指令的 JMP 指令。这例程可能或不可能修改系统日期和时间单元。

2. 若在一个 DATETIME 调用之后,系统日期和时间单元非零,MLI 使用这些值。其它情况下 MLI 使用已在文件的改过字段内的值(如果文件正被建立,则是零)。

若你的系统有一个 Thunderclock 卡,则从时钟卡读取当前日期和时间的例程被自动安装,并且 JMP 到此例程的起始地址放入 DATETIME 单元。若你想从其它类型的时钟/日历卡上自动地读取日期和时间,你必须自己安装一个相应的例程。这子程序应能从卡上读取日期和时间,并用下面的格式放置到 \$BF90-\$BF93内:

	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0												
日期	年							月							日													
	49041 (\$BF91)														49040 (\$BF90)													

(下转第6页)

单片机软件抗干扰新设计

山东烟台大学 李凯里

工业环境不可避免地存在着各种电的干扰,经常对单片机应用系统的正常运行造成影响。轻则使输入输出数据出错,重则导致程序完全失控,即产生所谓的软件故障。后者是对系统长期可靠运行的最大威胁。

软件故障的机理已经清楚,无非是在取指令期间,程序计数器 PC 本应指向指令码,却因某种干扰而误指向了非指令码(包括数据区的数据或程序区中的操作数)。接下去 CPU 便执行一条非正常指令,极易引起一系列错误使程序脱轨。

为防止程序脱轨,就需要设计纠错系统。理想而言,这样的系统应能尽快地发现干扰,甚至在 PC 指针刚出错的那个周期内就发现并纠正之。我们称这样的系统为高定位精度(High Location Precision)的纠错系统。

现存的几种克服软件故障的措施,大致可分为两类。一是完全依赖软件的,如指令冗余和软件陷阱技术。另一类是基于硬件的,其典型方法如程序运行监视系统(Watchdog 即 WTD)等等。它们虽然在一定程度上能起到纠错作用,但往往是当 CPU 执行了若干条错误指令,或者系统出现了明显的错误标志后方才发现并加以纠正的。也就是说,它们都不能算是高定位精度的纠错系统。

在对可靠性要求较高的场合,人们总是希望纠错系统的定位精度尽可能地高。是否能够实现这样的纠错系统?本文给出的方案将做出回答,即实现高定位精度的纠错系统,至少原则上是可行的。

一、故障机理及对策

单片机虽然采用了哈佛结构,但在外部程序区内仍然存在着干扰的可能(即指令码与操作数相混淆)。本文的方案主要针对这一问题而设计。

我们设想,一个好的纠错系统必须既能对 CPU 的取指状态进行检测,还能同时对程序区中的指令码与操作数进行识别。当两者匹配时,认为程序运行正常。否则,一定是发生了干扰,应立即复位使程序转入正轨。

有些 CPU 芯片(象 Z80 系列、Intel 8086 等)自身带有输出引脚以表明取指状态。在 Z80 中为 \overline{M}_1 信号,在 Intel 8086 中为 \overline{S}_0 、 \overline{S}_1 、 \overline{S}_2 状态信号。这使得对其检测变得容易些。单片机 CPU 芯片没有这样的信号(注意, \overline{PSEN} 信号无法直接区别取指状态)。因此需要设计外部电路,用来产生一个与指令码周期相同步的信号作为取指状态。

另一方面,为了识别程序区中的指令码与操作数,还需要采用虚读技术。即设置一个与程序 ROM 芯片相对应的辅助 ROM 芯片(或称映像 ROM)。其内存数据用来表示与程序 ROM 相对应地址上的数据的属性(指令码/操作数)。工作时,这些虚读字节被送到纠错系统,看其是否与 CPU 的取指状态相匹配,为系统做何反应提供依据。关于虚读技术的细节,请参阅杂志“微型计算机开发与应用”1992年第3期。

二、构造取指状态信号

MCS-51 系列单片机的 \overline{PSEN} 信号在取操作数时也产生输出,因此不能直接用作取指状态信号。但经适当处理,仍可资利用。

对单片机指令系统进行分析后可知,每条指令总有其固定的时钟周期数,且有倍数关系,即分别是 12、24、48 个时钟周期。对应于 \overline{PSEN} 信号则分别为 2、4、8 个周期(姑且称其为 \overline{PSEN} 周期)。唯一的例外是 MOVX 类指令,它虽然为 24 时钟周期,却只含有 2 个 \overline{PSEN} 周期,可以将其按 12 时钟周期指令对待。

做上述分析是要说明,若根据不同指令码将 \overline{PSEN} 信号简化为三种间断的信号(2 分频、4 分频、8 分频)之一输出,那么这种信号就能唯一地表示取指周期而不再含有取操作数周期。譬如指令 INC DPTR (A3H),因为是 24 时钟周期指令,可将 \overline{PSEN} 进行 4 分频,得到的输出一定是下一条指令的取指时刻。

这样我们首先解决了 CPU 取指状态的检测问题。其次还需要在映像 ROM 芯片里根据程序的指令码排列,分别设置不同的识别字节,并通过虚读方式提供纠错系统。例如可按图 1 进行设计。其中 D_0 、 D_1 、 D_2 位表示该指令码的 \overline{PSEN} 周期数, D_7 产生 GATE 信号以控制复位状态。对于指令码, GATE 设为 0,表示不复位(程序正常运行)。如果是操作数,则 GATE 设为 1(允许复位)。其他位暂不用。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
-------	-------	-------	-------	-------	-------	-------	-------

GATE

\overline{PSEN} 周期

图1 指令码/操作数识别字节

三、高定位精度(HLP)纠错系统的实现

图 2 给出 8031 单片机 HLP 抗干扰系统的原理性方案。

图中 U_5 为程序 ROM U_4 的映像单元,其输出数据

被 U_6 锁定。我们以 \overline{PSEN} 的上升沿为基准进行分频，得到三路不同 \overline{PSEN} 周期 (1/2、1/4、1/8 周期) 的信号至 U_{12} 。究竟输出哪一路则由 U_6 锁定的指令码识别字节的 D_0 、 D_1 、 D_2 位确定。 U_{13} 的输出称为 ACT 信号，它的上升沿总是代表取指时刻。ACT 起两个作用，一是送给 U_6 以锁定下一时刻的 \overline{PSEN} 周期数和当前的 GATE

信号 (D_7 位)。同时又输入至 U_{14} 用来产生复位信号。而能否发生复位则取决于该时刻锁定的 GATE 状态。如果是指令码，总有 $GATE=0$ ，当然不会产生复位。否则 $GATE=1$ ，即表示发生了干扰 (因为只有锁定了操作数识别字节时才会使 $GATE=1$)，这时 ACT 的上

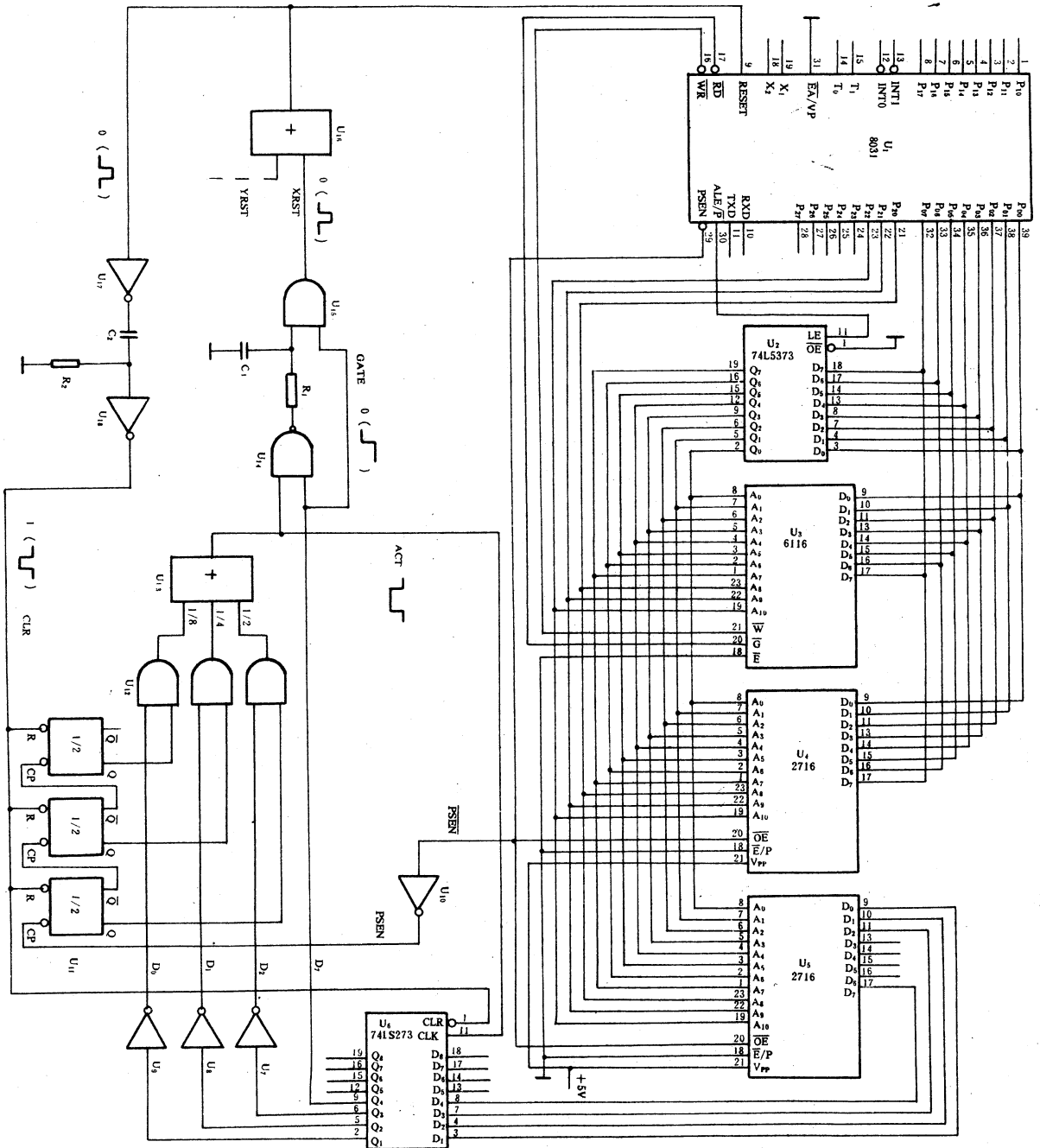


图2

(下转第6页)

一种简单实用的集散控制系统

山东工业大学自动化系(250014) 王济浩

随着计算机技术的不断发展,工业生产中对于自动控制的要求越来越高。从单台设备控制,发展到一条流水线,一个车间甚至整个工厂的复杂控制。完成这种复杂控制任务一般应采用集散式计算机控制系统。所谓集散式控制系统通常由两级以上计算机网络构成。下位控制机完成设备一级的监控,上位管理机收集数据,发出命令,完成管理功能,在上位机之上还可以有更高一级的上位机。上下位机之间通过串行通信联接起来,形成一个完整的控制网络。

集散式控制系统的规模大小、复杂程度差别很大,价格也很悬殊。我们从具体的经济和技术情况出发,设计了一套小型的集散式控制系统。该系统上位管理机采用 IBM-PC 或其兼容机。这种机型通用性好,自由度大,功能强,价格低,特别是具有汉字和图形功能,并有丰富的软件支持,可以完成各种复杂的管理任务。下位机采用 MCS-51 系列单片机。单片机速度较快,功能较强,使用灵活,而且在我国开发和应用都较成熟,特别是具有多机通信功能,非常适合于组成集散式控

制系统。本文主要谈谈两级计算机之间的通信。

一、通信标准

控制系统网络采用了总线式结构,如图1所示。所有下位控制机全部挂在上位 PC 机的串行通信线上,下位控制机之间不进行通信,只在上位机和下位机之间通信。

在 PC 机中,一般都有一块串行通信板。该板完成串并数据转换和串行数据接收、发送的任务,采用 RS-232C 通信标准。这块板使用简单,不加调制解调器时,只用三条线即可完成通信功能。其不足之处是带负载能力差,通信范围小,不超过十几米,很难满足一般集散控制系统的需要。为了充分利用这块现有的串行板,并且进一步扩大通信范围,我们研制了一块通信转接板,接在串行板和通信线路之间,这样就把通信标准从 RS-232C 标准变成了 RS-422A 标准。通信转接板线路如图2所示。

图2中的 MC1488和 MC1489是实现232标准通信的一对芯片。前者发送,完成 TTL 电平到232标准电平的转换;后者接收,完成从232标准电平到 TTL 电平的转换。MC3487和 MC3486是实现422标准通信的一对芯片。前者发送,把 TTL 电平变成422标准电平;后者接收,将422标准电平变成 TTL 电平。

PC 机串行接口板送出的信号,已经由发送芯片 MC1488送出,变成了232标准电平。进入通信转接板后,先由 MC1489接收,变成 TTL 电平。然后再经过 MC3487发送出去,变成了422标准电平,一直送到下位控制机。再由下位机 RXD 口外接的 MC3486完成串行接收。

PC 机接收串行通信的过程与此相反。下位机的 TXD 发出信息→MC3487发送→通信转接板的 MC3486接收→MC1488发送→PC 机的串行接口板接收,完成了从下位机到上位机的通信。

通信标准改变以后,采用了平衡传输方式,带负载能力和抗干扰能力大大提高,通信距离可以扩大到1200米以上,完全可以满足一般集散控制系统的要求。

二、通信约定

MCS-51 系列单片机的串行通道是一个全双工的串行通信口,既可以双机之间通信,也

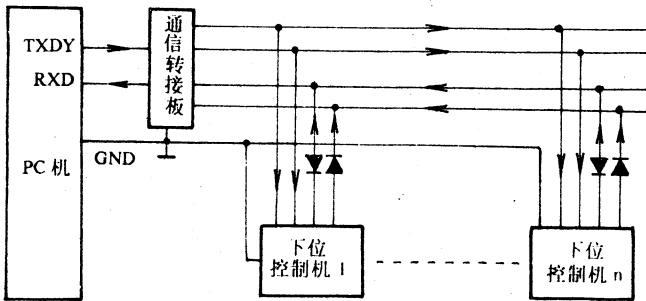


图1

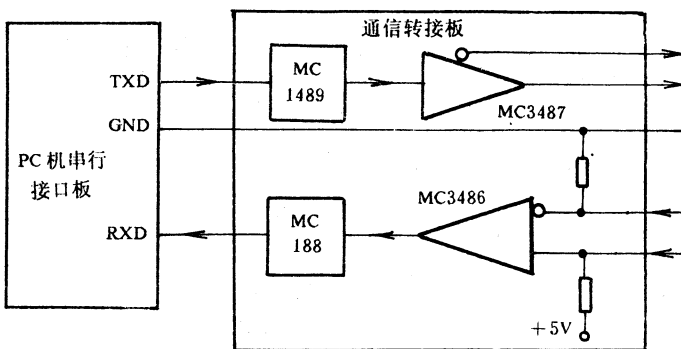


图2

可以实现多机通信。当串行口工作在方式2或方式3时，若特殊功能寄存器 SCON 的 SM₂由软件置为“1”，则为多机方式。若 SM₂为“0”，则为9位异步通信方式。

在多机通信时，MCS-51发送的帧格式是11位，如图3所示。其中第10位是 SCON 中的 TB8，它是多机通信时发送地址 (TB8=1) 或发送数据 (TB8=0) 的标志。串行发送时自动装入串行帧格式的相应位。在接收端，一帧数据的第10位信息被装入 SCON 的 RB8 中，接收机根据 RB8 以及 SM₂ 的状态确定是否产生串行中断标志，从而可以响应或不响应串行中断，这样就实现了多机通信。

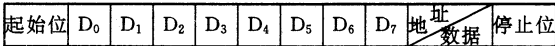


图3

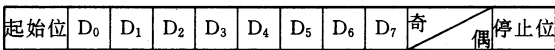


图4

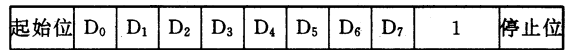
PC 机的串行通信由接口芯片 8250 完成。它并不具备多机通信功能，也不能产生 TB8 或者 RB8。但我们可以灵活使用 8250，用软件完成上述功能。8250 可以发送几种字长。其中一帧最长为 11 位，格式如图 4 所示。与 MCS-51 发送的帧格式相比，差别仅在第 10 位。即 PC 机的 8250 发送的第 10 位是奇/偶校验位，而不是相应的地址/数据标志。可以采用软件编程的方法使 8250 的奇/偶位形成正确的地址/数据标志。

在串行通信中，一般数据都采用 7 位数码的 ASCII 码，最高位 D₇ 不用，因而在 PC 机发送信息时，可根据发送的是地址或数据，以及 7 位有效数码的奇偶性，硬性把 D₇ 位置为“0”或者“1”，这样就完全符合了 MCS-51 多机通信的要求。

三、通信控制方式

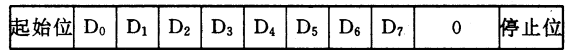
通信的控制方式采用轮流呼叫方式，即由上位管理机轮流呼叫每一个下位机。某一个下位机收到针对本机的呼叫后，判明若是上位机发的某个命令，则加以执行；若是上位机要求发送数据，则向上位机发送数据。校验方式可以采用发送校验和的方式，也可采用其他方式。

命令类



下位机地址 命令码 未用

数据类



数据 ASCII 码 未用

图5

通信的信息分成两类。由上位机向下位机发的命令类和由下位机发向上位机的数据类，其格式见图 5。

LED 智能显示屏的结构及驱动、显示电路

青岛经济技术开发区电子技术公司 张艺

八十年代以来，大屏幕智能显示屏作为一项高科技产品正引起人们的高度重视。采用计算机控制，将光、电融为一体的大屏幕智能显示屏，已经应用到工业、交通、商业广告、新闻发布、体育比赛、模拟军事演习、电子景观等领域。

LED 智能显示屏的像素采用 LED 发光二极管，将许多发光二极管以点阵方式排列起来，便构成 LED 阵列。LED 智能显示屏的优点是利用低电压扫描驱动，耗电省、成本低、清晰度较高、寿命长。我公司研制的 LED 智能显示屏分为单色、彩色两种。采用电流控制型发光二极管，亮度分为普亮、高亮和超高亮度，以适用于不同的要求，可配数十幅图象。

下面将该 LED 智能显示屏的结构作一介绍，并对驱动、显示板的电原理作一分析。

一、系统结构

该系统由微型计算机（编辑控制微机，即上位机）

及工作控制微机（即下位机）、驱动电路、显示屏及电源组成。

其方框图如图 1 所示。

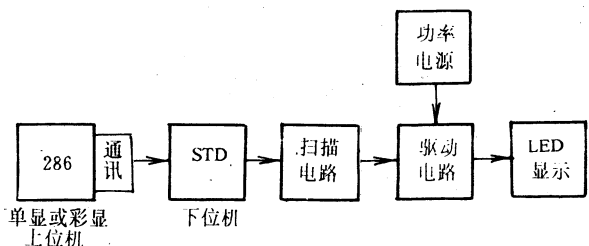


图1 系统结构框图

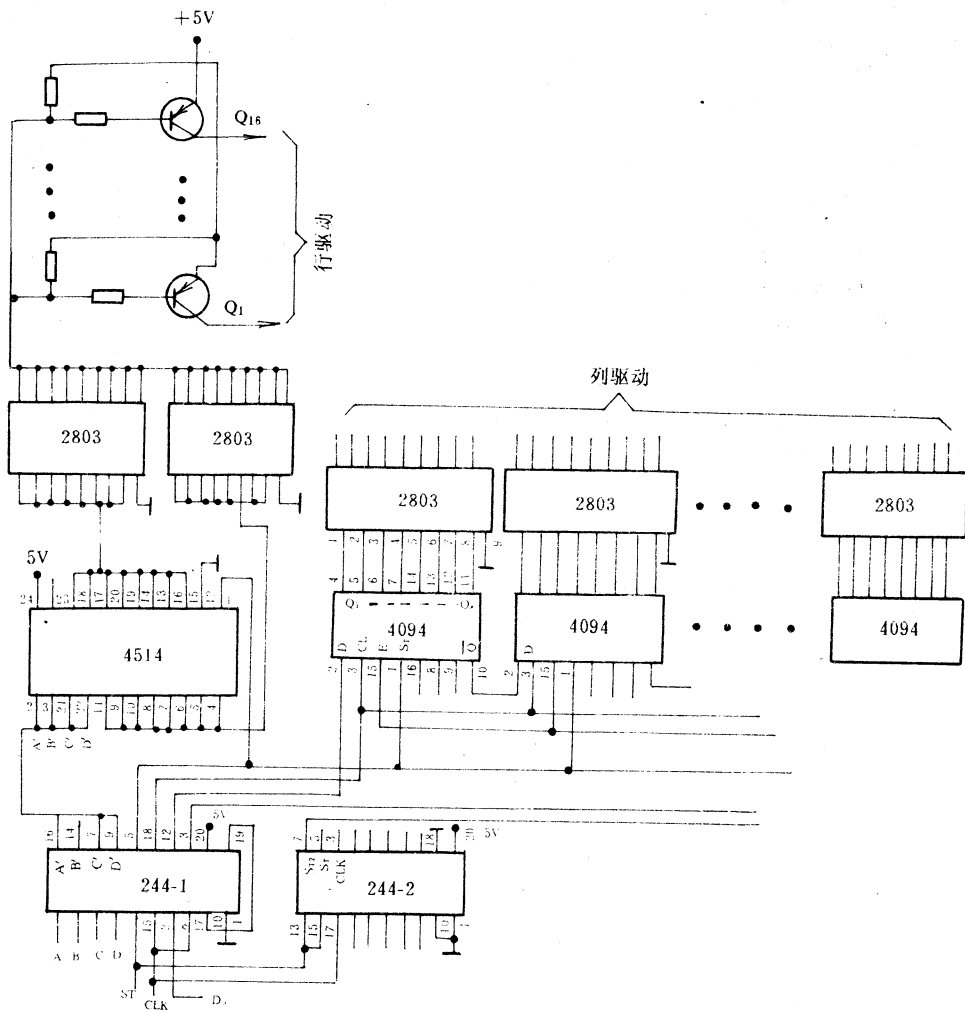


图2 驱动板原理图

编辑控制微机负责完成对图文的编辑及向工作控制微机发送所需显示的信息。工作控制微机对显示系统进行全面控制，驱动电路部分接受来自工作控制微机的命令和要显示的图文信息，传到显示屏上显示。各单元互相配合，各司其职。

二. 驱动电路

驱动电路组成如图2所示。

1. 结构

驱动电路分为行驱动和列驱动。行驱动由两只HC244、一个4514、十六个TIP127pnp达林顿功率三极管组成。一只127对应于一行LED。2803为八输入端八输出端的npn三极管阵列，每一输出端与一只127对

应。HC244是八缓冲器/线驱动器/线接收器，为八个三态门。CD4514为4—16线译码器。列驱动由多个MC4094和2803对组成，MC4094为八位并行输出串行移位寄存器。

2. 工作原理

行驱动：从图2所示，HC244—1在4、6、13、11、端接收扫描电路传来的行信号A、B、C、D，为并行通信方式，在16、14、7、9相应并行输出A'B'C'D'至4—16线译码器4514的2、3、21、22端，输出端为16个，分为2组，各为8脚，其中11、9、10、8、7、6、5、4脚为一组，18、17、20、19、14、13、16、15为另一组。当有选通信号ST经244的15脚输入，从其5脚输出到达4514的1脚时（即此脚为高电平时），4514的两组输出端中必有一组根据其

输入信号有相应的输出(即为高电平),此信号传至相应的功率三极管阵列2803,其相应的输出端为低电平。此时,相应的127的基极为低电平,达林顿管导通,其集电极将高电平加于LED阵列的对应的行上,即LED的正极上。其顺序从Q1到Q16。这就是行驱动电路的逐行扫描过程。行信号A、B、C、D的顺序变化范围从0000、0001、0010、.....1111,来一个选通信号ST,行信号顺序就变化一次。其频率由扫描电路决定。行驱动电路是时序逻辑电路。

列驱动:从图2所示,32个MC4094八位并行输出串行移位寄存器,每个的输出端4、5、6、7、14、13、12、11脚均接其对应的2803的八个输入端1、2、3、4、5、6、7、8脚。第一个4094的串行移位输出端Q(即10脚),连到下一个相邻的4094的数据输入端D端(即2脚),依次顺序连接直到第32个4094为止。各个4094的使能端15脚E均接高电平,使其都处于工作状态。来自扫描电路的时钟脉冲CLK通过244的2脚输入,在其18脚输出到32个4094的3脚。数据D_n经过244的8脚,在其12脚输出到第一个4094的2脚,各个4094的选通端即1脚,均并联与4514的1脚一起接到244的5端,即ST输出端。在时钟脉冲到来时,来自扫描电路的数据串D开始输入第一个4094。在时钟脉冲的不断作用下,数据脉冲不断输入,并在4094的输出端串行移位。第一个4094被灌满后,又移向第二个,依次第三个,.....直至第三十二个。数据灌满32个4094需移位256次,也就是说,一个数据串为256位。在256个时钟脉冲的作用下,当移位256次后,就从扫描电路来一个选通信号ST。这个选通信号通过244,作用到各个4094的1脚和4514的1脚时,每个4094的8个输出端均根据数据串D而输出相应的高电平或低电平。各个对应的2803的输出端也相应地输出相反的电平,低或高。2803在此做功率输出,亦是倒相器。这就是选通信号ST的一个周期,也就是数据串D的一个接收周期。周而复始。数据的异同取决于扫描电路。该列驱动电路也是一个时序逻辑电路。

三. LED 显示电路

1. 结构

结构图见图3所示。

采用成品化的LED矩阵板拼装成大屏幕显示屏,不仅可以简化安装工艺,而且可以提高使用寿命及图

象分辨率,减少故障。图3是32×16 LED单色矩阵板的结构,分为共阳共阴两种阵列,行为共阳,列为共阴。每块板为2个字,每个字为16×16阵列,即为单个文字显示单元。

2. 原理

行、列驱动电路各对应驱动LED阵列的行和列,在选通信号S的作用下,在整幅画面的显示中,LED阵列有亮有暗。这个阵列中的某个象素的亮暗状态可用一个函数表示,即:

$$Z_{ij} = (x_i, y_j)$$

$$i = 1-16 \quad x_i = 0, 1$$

$$j = 1-32 \quad y_j = 0, 1$$

只有当第i行为高电平, $x_i = 1$, 而同时第j列为低电平时, $y_j = 0$, 即 \bar{y}_j 。此时该象素点亮, 其余情况均不亮。由亮暗相间的象素组成了图案和文字。

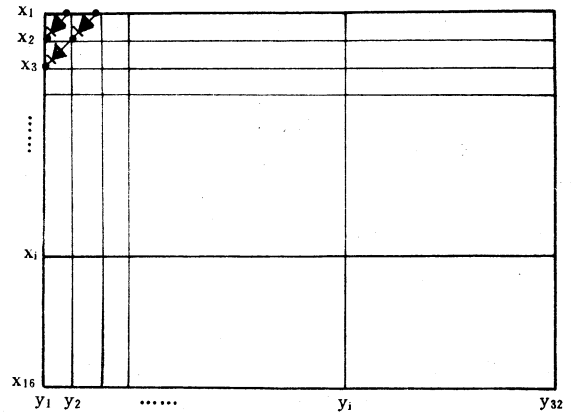


图3 LED 阵列电路的结构

四. 结语

以上仅对我公司研制的LED智能显示的结构和驱动、显示电路的原理作一介绍和分析。对于显示屏的其它部分因篇幅关系,在此不作详述。实践证明,该显示屏结构合理,驱动、显示部分可靠、实用。

大屏幕智能显示屏集电子、计算机、半导体等多项技术之大成,是高科技的结晶,具有广阔的发展前景和应用范围。

LM567单管红外发射与接收电路

首都师范大学 解永勃 孙继安

LM567锁相环音频译码集成电路,专门用于解调某一频率的单音信号。其⑤、⑥脚外接的阻容元件决定内部压控振荡器的中心频率 $f_0 = \frac{1}{1.1RC}$ 。当输入端③

脚信号频率与其设定的中心频率 f_0 相符时,⑧脚即由高电平变为低电平。本电路(见图)利用以上特点,巧妙的将LM567自身的中心频率由⑤脚经放大后,由红外

(下转第27页)

激光印字机的维护与故障排除

广西师范大学(541004) 崔晨荣

激光印字机是世界上最吸引人的打印机新技术之一,它是激光、微电子和机械技术的综合应用。激光印字机问世以来,以其工作时噪音极低、打印速度快、分辨率高、性能稳定等突出特点,以及可打印出各种高质量的文本、图形、图像、曲线、表格等优越性能,越来越博得人们的青睐。

作为一种十分理想的计算机输出设备,近年来,激光印字机已越来越多地配套应用我国各行各业的计算机系统。由于激光印字机性能优越,价格高于其它打印输出设备,而国内介绍激光印字机技术的资料有限,若因不注意日常维护或使用不当,就可能产生不该发生的故障甚至损坏设备,严重影响使用,给单位造成不必要的损失和增加经济负担。因此,如何正确使用和维护激光印字机,掌握常见故障的排除方法,愈发显得重要。下面介绍几点激光印字机使用中可行的维护方法及常见故障的排除方法。

一、激光印字机的日常维护

在使用激光印字机过程中,需经常对其进行清洁保护。以避免因不注意清洁保养或使用不当而造成的不该发生的故障。

需清洁的主要部件为:①转印电极丝;②传输器条板;③传输器锁盘;④输纸导向板;⑤静电消除器。

转印电极丝是一种非常精细的钢丝,可将吸附着墨粉的负电荷从感光鼓传到打印纸上。印字机使用一段时间后,会有一些残留的墨粉在电极丝周围。清洁时用毛刷或略浸酒精或清水的棉签清洁电极丝周围的区域。清洁此部位时一定要格外小心,不要弄断横跨在电极丝上方的几根单丝线。

清洁传输器条板和传输器锁盘的方法,是用软布略蘸清水擦净银白色长条板和擦掉传输锁盘上积存的纸尘。

输纸导向板位于粉盒下方,它的作用是使纸张通过粉盒传输到定影组件。清洁时用软布略蘸清水擦净导向板的表面。

静电消除器的位置与转印电极丝的水平位置在一起,清洁时须用印字机所带小刷清除掉静电消除器周围的纸屑和墨粉。(经常清洁此部位可减少卡纸现象)

另外,激光印字机的感光鼓为有机硅光导体,存在

着工作疲劳问题。因此,刚使用过的感光鼓不宜连续工作。最好放置一星期左右或更长的时间后再使用,效果就较好,一般建议用2只以上粉盒互换使用,可避免感光鼓的疲劳。

注意:①请不要推开感光鼓护盖,使鼓露光。

②清洁上述工作之前必须预先关闭电源。

二、有关打印质量问题及解决办法

通常,印字机出现墨粉不足信息时,印字样张上会出现字迹不清或浅淡现象,这说明粉盒内的墨粉不足。此时,可掀开打印机上盖,取出粉盒,双手水平握住,横向轻轻摇晃粉盒几次,使盒内墨粉均匀分布,然后再使用,即可消除字迹不清或浅淡现象。若按此方法无法排除,则说明需要装新墨粉或更换新的粉盒。

粉盒内有感光鼓和墨粉。当在纸样上出现不规则的划痕,或浅或深并带有一定的周期性(十几厘米重复一次),则是由于粉盒内的感光鼓受磨损造成的。此时,须更换新粉盒,现象就会消失。一般情况,不要随便碰感光鼓,并不可清洁。除非在绝对必要时,也只能用镜头纸或软布(柔软、不起毛)沾墨粉轻拭。

当在纸样上无规律的出现空白圆点时,一般说明纸的性能不好或纸的表面有潮湿点而不能接受墨粉,或者是转印电极丝较脏造成,解决的办法是换用其它性能好且干燥的纸张,或清洁打印机内部的转印电极丝及周围区域。

当印字纸样上出现黑色条纹或在纸纵向出现模糊的墨粉时,多数情况是定影辊清洁衬垫被污染和损坏造成。解决措施是清除清洁衬垫表面的墨粉或更换它。

三、激光印字机卡纸现象及排除

如果激光印字机出现卡纸错误,多数情况是在纸的传递通道上有纸屑、碎纸等,而减慢了纸行进的速度所致。通常,卡纸出现在三个位置:①纸样输送区;②传送导向区;③定影组件和最终传输区。

一般排除的方法为:打开印字机上盖,观察上述三个纸通道区,进行必要的清洁或打开卡纸部位将所卡的纸张取出。但需注意的是,若卡纸出现在定影组件区,当打开定影组件后,不要通过定影组件向后取纸,而要将卡住的纸向印字机前面提取出,以免造成被卡纸的墨粉掉进定影组件中,引起打印质量问题。

第二章 6527 CPU 的显示系统(下)

山东苍山机械电子化学工业局(277700) 于 春

五、显示系统的配色

1. 6527 CPU 的配色区

我们知道在 F BASIC 状态下,当使用 CGSET 指令选定背景和卡通配色板后,就可以在 DEF MOVE、DEF SPRITE 等指令中,使用选定配色板的配色代码。由于每个配色板有12种颜色,故两个配色板有24种颜色供选用。当我们欲自己组合颜色时,可使用 PALETB、PALETS 指令进行定义,定义后就可以在显示画面中使用。另外,我们还可以在背景配色板或卡通配色板的 0# 配色代码的重新设定中改变底背景的配色,使用极为方便。但是,我们自己设定的配色数据存于系统 RAM 区的什么位置呢?查遍 RAM 区也找不到。原来6527 CPU 工作系统有一专用的配色区,这一配色区位于 PPU 内 \$ 3F00 ~ \$ 3FFF 中。

2. 6527 CPU 配色区的地址分配

在 PPU 的 \$ 3F00 ~ \$ 3FFF 这 256 个单元中,存储了当前使用的背景、卡通配色面板中的各颜色代码。它们的地址分配见表七。

表七 PPU 配色地址分配

地 址	用 途
3F00	底背景配色
3F01~3F03	背景配色板,0# 配色代码的颜色代码存储单元
3F05~3F07	背景配色板,1# 配色代码的颜色代码存储单元
3F09~3F0B	背景配色板,2# 配色代码的颜色代码存储单元
3F0D~3F0F	背景配色板,3# 配色代码的颜色代码存储单元
3F10	底背景配色
3F11~3F13	卡通配色板,0# 配色代码的颜色代码存储单元
3F15~3F17	卡通配色板,1# 配色代码的颜色代码存储单元
3F19~3F1B	卡通配色板,2# 配色代码的颜色代码存储单元
3F1D~3F1F	卡通配色板,3# 配色代码的颜色代码存储单元
3F20~3F2F	映射于 \$ 3F00 ~ \$ 3F0F
3F40~3F4F	映射于 \$ 3F00 ~ \$ 3F0F
.....	
3FE0~3FEF	映射于 \$ 3F00 ~ \$ 3F0F
3F30~3F3F	映射于 \$ 3F10 ~ \$ 3F1F
3F50~3F5F	映射于 \$ 3F10 ~ \$ 3F1F
.....	
3FF0~3FFF	映射于 \$ 3F10 ~ \$ 3F1F

注—— 1. \$ 3F04、\$ 3F08、\$ 3F0C、\$ 3F14、\$ 3F18、\$ 3F1C 等最末一位是 4、8、C 的单元没有使用,一般置入 0FH。

2. 由于 CPU 的配色区位于 PPU 内,故一般称 \$ 3F00 ~ \$ 3FFF 为 PPU 配色区。

例四、写出在 F BASIC 状态下执行 CGSET 1,0 和 CGSET 0,2 命令后配色单元的内容。

执行 CGSET 1,0 后配色单元内容如下:

```
3F00—0F 30 21 02 0F 30 27 18
3F08—0F 30 27 16 0F 29 36 17
3F10—0F 36 16 02 0F 27 30 19
3F18—0F 35 25 17 0F 30 27 16
```

执行 CGSET 0,2 后配色单元内容如下:

```
3F00—0F 2C 15 07 0F 27 21 12
3F08—0F 29 36 17 0F 30 26 07
3F10—0F 30 26 12 0F 30 15 12
3F18—0F 30 12 16 0F 30 26 19
```

以上运行结果可以使用程序 No. 2—8、2—9 稍加改造而读出。读者可试读验证。

当执行 PALETB 0,3,5,7,9 后,底背景变为蓝色,相应的配色单元变为:3F00—03 05 07 09 0F 27 21 12

读者可以用写入 PPU 的方法,把各颜色代码直接写入 PPU 配色单元。

3. 说明:

(1) 在配色区,每连续的四个单元为一组,每组的后三个单元存背景页颜色代码。

(2) 在各配色单元中,只有 PPU 地址的末位为 0 的单元是底背景配色单元,其它地址无效。

(3) 如果把 \$ 3F00 ~ \$ 3FFF 分成 16 行,那么偶数行之间互相等效(包括 0 行,即 0、2、4、6……),也就是说对 \$ 3F60 ~ \$ 3F6F 操作等效于对 \$ 3F00 ~ \$ 3F0F 操作,都作用于背景页。同样,奇数行之间互相等效,它们都作用于卡通页。

六、图形库结构与卡通块设计

1. 图形库结构

在任天堂游戏中,无论是背景还是卡通都采用了点阵块显示的方法。这种方法的优点是:画面处理速度快,占用系统内存少。

(1) 图形库在内存中的位置

图形库占用 PPU 内存的 \$ 0000 ~ \$ 1FFF 的 8K 空间。一般使用 8×8K 的 2764 存放,它共分两个区:第 I 区、第 II 区。

第 I 区占用 \$ 0000 ~ \$ 0FFF 共 4K 空间,存放 256 个 8×8 点阵的卡通图形块(包括字符)。

第 II 区占用 \$ 1000 ~ \$ 1FFF 共 4K 空间,存放 256

个8×8点阵的背景图形块(包括字符)。

我们可使用程序 No. 2—7 读出图形库数据以进行研究。

(2) 图形结构

在图形库中,从\$0000单元开始,每连续的16个字节为一段,定义一个8×8点阵的图形块,实际上每一个8×8点阵的图形是由两个8×8点阵的子图叠加而成的。图形结构如下:

a. 把16个字节分成两部分,前八个字节为第一部分,后八个字节为第二部分。这两部分分别对应于一个8×8点阵的子图形。

b. 把两个子图叠加,构成一个8×8点阵的图形块。这样,每个点就由两位二进制数表示,从而可以使用四种颜色组出各种各样的图形块。

下面通过实例说明图形块的定义方法和定义过程。

例五、定义一个图形块。要求:用0#配色为底色;用1#配色围成一个方框;在方框内有一个罗马数字“1”,要求用2#配色写横;3#配色写竖。图形块如图7所示。

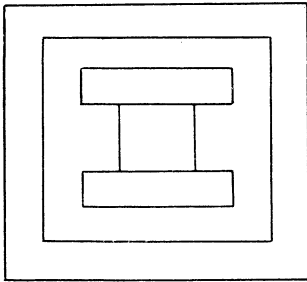


图7

定义过程如下:

a. 画一个8×8的方格(如图8)

b. 在最外圈的小方格中填入1#配色代码01,使其构成1#配色的方框。填入配色代码数字时,要求一个方格中的数字分两行填写,填入“01”时,上一行填“0”、下一行填“1”。(下同)

c. 从第3列第三行开始填入四个“10”,构成“1”的上横。从第3列第六行开始填入四个“10”,构成“1”的下横。

d. 从第4列第四行开始填入两个“11”,从第4列第五行开始填入两个“11”,构成“1”的一竖。

e. 把剩余的方格均填入“00”。

f. 把每一格中上行的八位二进制数写在方格的左边;把下行的八位二进制数写在方格的右边。

g. 先自上而下抄下左边的八个数据,再抄右边的八个数据。这16个数据就是例五的图形数据。它们为:

00 00 3C 18 18 3C 00 00
FF 81 81 99 99 81 81 FF

列:	1	2	3	4	5	6	7	8	
行:一 00	0	0	0	0	0	0	0	0	FF
	1	1	1	1	1	1	1	1	
二 00	0	0	0	0	0	0	0	0	81
	1	0	0	0	0	0	0	1	
三 3C	0	0	1	1	1	1	0	0	81
	1	0	0	0	0	0	0	1	
四 18	0	0	0	1	1	0	0	0	99
	1	0	0	1	1	0	0	1	
五 18	0	0	0	1	1	0	0	0	99
	1	0	0	1	1	0	0	1	
六 3C	0	0	1	1	1	1	0	0	81
	1	0	0	0	0	0	0	1	
七 00	0	0	0	0	0	0	0	0	81
	1	0	0	0	0	0	0	1	
八 00	0	0	0	0	0	0	0	0	FF
	1	1	1	1	1	1	1	1	

图 8

若以空白“□”代表0#配色,以“▨”代表1#配色,以“▩”代表2#配色,以“■”代表3#配色,对图8进行涂色后,图形效果将如图九所示。

2. 卡通图形的设计

卡通图形的设计过程与背景图形大同小异,所不同的是:卡通图形一般都是16×16点阵的,一个卡通图形要由四个8×8点阵的卡通块组成。

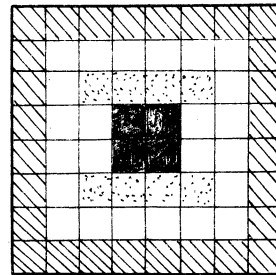


图9

因此,在定义卡通图形时,要把一个图形分为四份,每一个卡通块占图形的四分之一。下面我们验证一下F BASIC中玛丽的定义数据。

查阅卡通图形库数据区,在\$0040~\$007F连续的64个单元是“玛丽走2”的点阵数据,现抄录如下:

0040—00 00 07 1F 3F 1F 0F 00
0048—03 0F 00 02 04 1E 00 03
0050—00 00 E0 F0 F0 F0 C0 80
0058—C0 E0 E0 50 D0 30 00 E0
0060—01 11 1F 0F 07 01 00 00
0068—07 0F 0F 0E 06 07 0E 01
0070—00 80 80 A0 E0 C0 00 00
0078—E0 E0 60 60 60 C0 C0 C0

把以上数据填入4个8×8的方格表中(从略。读者可试填一下)。

若仍按例五对配色代码的代表色块约定,对以上4个8×8方格表涂色,将得到“玛丽走2”的图形。见图10。

凡玩过《超级玛丽》游戏的朋友都知道,游戏开始时玛丽只有16×16点阵大小,当吃到红蘑菇后,身体立即长大为32×32点阵大小。但F BASIC 图形库中没有这么大的卡通图形,它的图形数据将是怎样的呢?实际上并不神秘,它不过是把玛丽的16×16点阵的图形放大一倍而已。下面我们讨论32×32点阵玛丽图形的设计方法。

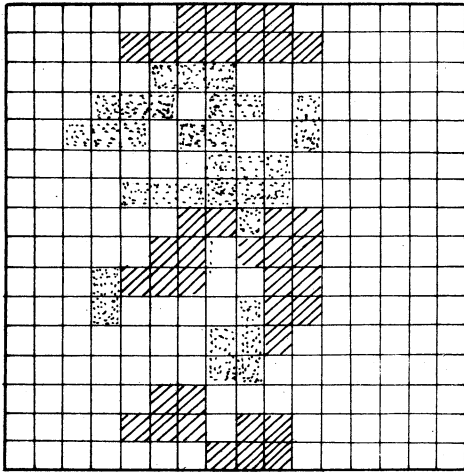


图10

例六、试求32×32点阵的“玛丽走2”的图形数据。

现在,我们已经掌握了16×16点阵的“玛丽走2”的图形数据,那么求32×32点阵的图形数据将变得十分容易。凡实践过图形的方格放大的朋友都清楚:当把图形放大一倍时,只要把方格尺寸比原图方格尺寸扩大一倍,然后照原图填入每个方格的内容,就得到放大的图像。因此,对于例六的问题,只要把原16×16点阵图形中的4个8×8点阵图形块分成16个4×4点阵块;然后把每一个4×4点阵块的数据填入8×8点阵中,即4×4点阵中的一个方格中的数据要填入四个相邻的方格中;最后把这16个8×8点阵的图形块组合在一起,就完成了32×32点阵“玛丽走2”的图形设计。

下面,我们仅以原图形库中“玛丽走2”的第一个8×8点阵块放大为例,介绍图形放大的具体方法。

(1)读出“玛丽走2”的第一个图形块数据(即图形库第I区序号为4的SP块数据):

0040—00 00 07 1F 3F 1F 0F 00

0048—03 0F 00 02 04 1E 00 03

(2)把以上16个数据分为四组,每组4个,排列如下:

i. 00 00 07 1F

ii. 03 0F 00 02

iii. 3F 1F 0F 00

iv. 04 1E 00 03

(3)把以上四组数据中的每一个数据按高、低位拆开,高位为一组,低位为一组,共组成八组。排列如下:

A { i. 0 0 0 1
ii. 0 0 0 0 } B { i'. 0 0 7 F
ii'. 3 F 0 2 }
C { iii. 3 1 0 0
iv. 0 1 0 0 } D { iii'. F F F 0
iv'. 4 E 0 3 }

(4)把每一组数字按表八的规则替换成相应的数据。

表八 图形放大数据换算表

原数据	替换数据	原数据	替换数据
0	00	8	C0
1	03	9	C3
2	0C	A	CC
3	0F	B	CF
4	30	C	F0
5	33	D	F3
6	3C	E	FC
7	3F	F	FF

以上各组替换完毕后的数据如下:

A { i. 00 00 00 03
ii. 00 00 00 00 }
B { i'. 00 00 3F FF
ii'. 0F FF 00 0C }
C { iii. 0F 03 00 00
iv. 00 03 00 00 }
D { iii'. FF FF FF 00
iv'. 30 FC 00 0F }

(5)把以上各组数据,每一个重写一次,使每行由4个数据变为8个数据。则有:

A { i. 00 00 00 00 00 00 03 03
ii. 00 00 00 00 00 00 00 00 }
B { i'. 00 00 00 00 3F 3F FF FF
ii'. 0F 0F FF FF 00 00 0C 0C }
C { iii. 0F 0F 03 03 00 00 00 00
iv. 00 00 03 03 00 00 00 00 }
D { iii'. FF FF FF FF FF FF 00 00
iv'. 30 30 FC FC 00 00 0F 0F }

以上A、B、C、D四组数据,即为4号卡通块放大一倍后的图形数据。我们把它写入方格验证一下(仍采用例五的约定),涂色后的图形见图11。

可见,放大的图形数据都是正确的。

仿照以上五个步骤,可以写出剩下的三个SP块5、6、7号的放大数据。有兴趣的读者可以自己做一遍。下面仅给出最后结果(以A、B、C……为序)。

E { 00 00 00 00 FC FC FF FF
F0 F0 FC FC FC FC 33 33 }
F { 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 }
G { FF FF FF FF F0 F0 C0 C0
F3 F3 0F 0F 00 00 FC FC }

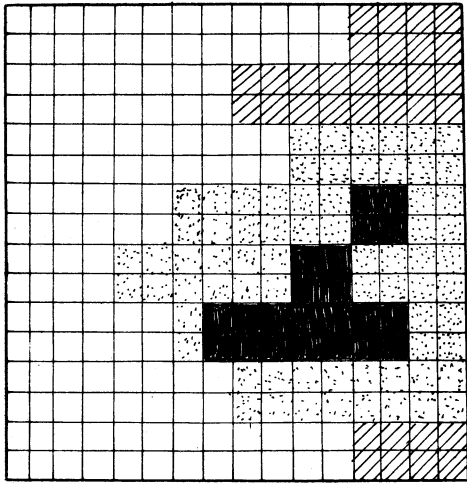


图11

H { 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00
 I { 00 00 03 03 03 03 00 00
 00 00 00 00 00 00 00 00
 J { 03 03 03 03 FF FF FF FF
 3F 3F FF FF FF FF FC FC

K { 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00
 L { 3F 3F 03 03 00 00 00 00
 3C 3C 3F 3F FC FC 03 03
 M { 00 00 C0 C0 C0 C0 CC CC
 FC FC FC FC 3C 3C 3C 3C
 N { 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00
 O { FC FC F0 F0 00 00 00 00
 3C 3C F0 F0 F0 F0 F0 F0
 P { 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00

读者可把以上12组数据填入12个8×8方格以对比放大后的效果。

通过以上讨论,我们基本弄通了图形块设计的方法步骤,同时也解释了我们的电脑游戏机中为什么没有《魂斗罗》、《双截龙》等游戏中的背景画面和卡通图案的缘故。明白了这一道理,我们可以设计自己喜欢、需要的图形库图形,通过仿真器写入 EPROM,替换出 BS 卡中的图形库存储器 2764,就可以使用自己设计的图形了。今后我们还将陆续刊登一些游戏中的卡通、背景图形数据,以供读者选用。

计算机中的新技术——Flash Memory

西安交通大学 张 戟

快速擦写存储器(Flash Memory)是 Intel 公司于 80 年代末 90 年代初推出的一种新型存储器,由于它众多的优点而深受用户的青睐。Flash Memory 的两个主要特点是可以整体电擦除和按字节重新编程,是完全非易失的。由于 Flash Memory 的突出性能,使它问世后获得了迅速的发展。Intel 公司已决定由 Flash Memory 来完全替代 E²PROM,不再生产 E²PROM 产品。虽然 E²PROM 具有可以按字节进行电擦除的特点,但只有很小一部分应用领域有这种特殊要求。下面将从三个方面来介绍 Flash Memory。

一、Flash Memory 的工作机理

Intel 公司的 ETOX—I (Eprom Tunnel Oxide) 快速擦写存储器的工艺是由标准的 CMOS EPROM 工艺发展而来的,它的单元结构与 EPROM 的结构一样。两者的主要差别是栅极氧化层厚度不同,EPROM 的氧化层厚度通常为 325 Å,而 ETOX—I 的氧化层较薄,为 100~120 Å,正是这一点使后者具有电擦除功能。

在编程方式下,Flash(以下都这样简称)的工作机理和普通的 EPROM 完全一样。控制栅上施加编程电压 V_{pp}(12v),漏极上施加比 V_{pp}稍低的电压(7V),源极

接地。在漏源极之间的电场作用下,热电子穿越沟道,但在控制栅上的高电压吸引下,这些自由电子越过氧化层进入浮置栅,当浮置栅获得足够多的自由电子后,就在源漏极间造成一个导电沟道。

Flash 的擦除机理与 EPROM 的机理完全不同。在 EPROM 中,靠紫外光来中和浮置栅上的电荷以达到擦除目的。对 Flash,高电压 V_{pp}施加在源极上,控制栅接地,在此电场作用下,浮置栅上的电子就越过氧化层进入源区,被外加电源中和掉。如图1所示。

二、Flash Memory 的主要性能特点:

1. 高速芯片整体电擦除——芯片整体擦除时间约 1S,而一般 EPROM 需花 15~20min 进行擦除。
2. 高速编程——采用快速脉冲编程算法。对于 28F256 芯片,每个字节的编程花费 100μS,对 28F512、28FD10 和 28FD20,每个字节的编程仅花 10μS。对于上述 4 种芯片,整个芯片编程时间分别为 4S、1S、2S 和 4S。对于 28F256A 芯片,整个编程时间为 0.5S。
3. 最少 10,000 个擦除/编程周期,通常可达到 100,000 个周期。
4. 采用 12V±5% 的编程电压。
5. 高速度的存储器访问——最大的读取时间不超

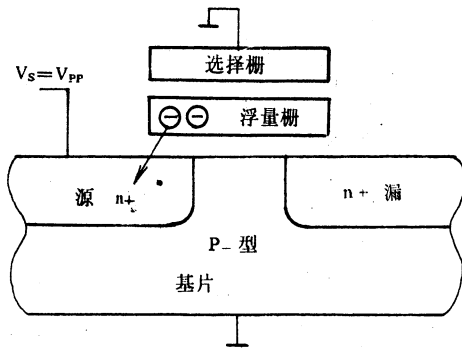


图1 擦除时的 Flash

过 135ns。6. CMOS 的低功耗——最大工作电流为 30mA，备用状态下的最大电流为 100 μ A。

7. 内部的命令寄存器结构可用于微处理器/微控制器(单片机)兼容的写入接口。

8. 抗噪声特征—— $\pm 10\%$ 的 V_{cc} 允差，通过 Intel 的独特 EPI 工艺过程实现了最高程度的死锁保护，在地址和数据线上承受 100mA 电流、 $-1V \sim V_{cc} + 1V$ 的电压仍能防止死锁。

9. 与 E^2 PROM 相比，Flash 具有密度大、价格低、可靠性高的明显优势。下表列出了两者的密度和可靠性方面的比较数据：

性能比较	Intel ETOX™ Flash	E^2 PROM
存储体单元晶体管数	1	2
单元尺寸(1 μ 特征尺寸)	15 μ^2	38 μ^2
循环周期故障	0.1%	5%

Intel 公司 1991 年提供的 DIP 封装的 28F256—200 (200 μ S 的 256K 位的 Flash) 批量报价约为 4.4 美元/片，与 E^2 PROM 相比，价格相当低廉。

三、Flash Memory 的应用领域

20 年来，计算机的发展突飞猛进，CPU 的性能几乎每两年提高一倍。而作为 I/O 的软盘和硬盘驱动器的性能却要 10 年才提高一倍。磁盘至少在 3 个方面严重影响了计算机的发展：

1) 磁盘驱动器是机电装置，它是计算机中最易磨损、寿命最短的部件。

2) 由于访问磁盘是低速的操作过程，每字节约需 15ms，大大低于 CPU 访问内存的速度(每字节 80~120ns)，因而磁盘操作形成了系统的数据“瓶颈”，是影响系统速度的关键因素。

3) 磁盘驱动器的功耗大，体积大。不利于计算机的

小型化。

多年来，计算机生产商一直在探索取代磁盘的途径，出现了所谓的“固态盘”(Solid State Disk)。在一些计算机系统中采用的 RAM DISK 就是一种固态盘，它实际上是带电池供电的 SRAM 卡。但这种 SRAM 的密度不高(1 个存储单元含 4~6 只晶体管)，采用电池来保持数据也不是十分可靠，且这种卡的体积仍较大，价格也贵，因而很难取代软盘和硬盘。但是随着 Flash 的出现，就使得固态盘真正有可能在很多领域内取代软、硬盘。这种替代是极具革新意义的，也是 Flash 的一个重要应用领域。它完全摒弃了机械装置，具有高达 1 百万小时的平均故障间隔时间，使计算机的寿命大为延长。Flash 的读取时间为 120~250ns，因而消除了机电式磁盘驱动器所造成的数据“瓶颈”，Flash 是高密度的 CMOS 芯片，由它做成的固态盘功耗低、体积小，特别适用于便携式计算机(如笔记本式计算机和膝上型计算机)。Intel 公司推出的一种 4M 字节的 Flash 存储器卡，典型工作电流为 40mA，典型备用电流为 800 μ A，功耗远比磁盘驱动器低，其体积为 $54 \times 85.6 \times 3.3\text{mm}^3$ ，也比最小的软盘驱动器(2.5 英寸盘)的体积小很多。这种固态盘已逐渐形成了一种被国际个人计算机存储器卡协会(PCMCIA)所认可的标准，称之为“PC 卡”。这种卡使用起来很方便，在便携式计算机中已获得广泛应用。目前这种卡的主要缺点是价格比较贵，但这不会成为其推广应用的主要障碍。

除了上述的 PC 卡之外，在 PC 机中采用 Flash 作为其植入式(Embedded)的存储器，可以从多方面提高计算机性能。若采用 Flash 来储存计算机的 BIOS(基本 I/O 系统)代码，可以使计算机厂商灵活地改变 BIOS，以适应微计算机技术迅速发展的趋势。微机系统厂商实行开放性的政策，更需要有灵活可变的 BIOS。品种繁多的硬件卡和成百上千种流行软件包，使得任何一种 BIOS 都难以兼容所有这些硬件和软件。如果 BIOS 是存储在 Flash 中的，则软硬件生产商可以在提供新的硬件卡或软件包时，同时提供一张含有适用的 BIOS 的盘(软盘或固态盘)，把新的 BIOS 重新装入机内 Flash 存储器中，以运行新的软硬件。

由于 Flash 具有整体电擦除和按字节重新编程的功能。所以 Flash 很适合于数据采集系统。采集的数据可以周期性地由芯片取出，进行分析。擦除后，成为空白芯片，可反复使用至少 10,000 次。也可以用多个芯片构成数据采集的“滚动窗口”。HP 公司的 HP54504A 双迹数字存储示波器的采集系统中就大量使用了 Flash 芯片。

对于需周期性地修改被储存的代码和数据表的应用场合，Flash 是十分理想的器件。因为若采用 EPROM，修改一次代码，需花 15~20 分钟，而用 Flash 只要花 1 秒钟。而且擦除和重新编程可以在同一个系统中或同一个编辑器插座中进行。在组装或装配阶段，为了对硬件进行诊断，常需要根据情况编写各阶段的硬件调试程序。若用 Flash 替代 EPROM，则芯片可直

接焊在电路板上,既提高了系统的可靠性,也增加了调试的灵活性。

在系统售后服务中,有时也要修改程序,如用 E-PROM 的系统修改就会很繁,有时干脆要把整块电路取下送回厂里返修,这种售后服务费用很昂贵。若采用 Flash 就可以通过串行通信口对电路上的芯片直接进行改写,大大降低了费用支出。

最后我们谈谈采用 Flash 来固化操作系统。

长期以来,操作系统是装到软盘或硬盘中的,很少采用固化的措施,这是因为系统设计者认识到操作系

统是在不断更新和改进的,固化反而会带来很多麻烦。凡是用过计算机的用户都会深刻体会到,开机后,等待磁盘驱动器稳定转动起来,然后把操作系统从软盘或者硬盘通过低速的 I/O“瓶颈”装载到系统内存中,是一个相当“漫长”的过程。如果采用 Flash 来固化操作系统,将会显著地缩短开机后用户等待的时间,而同时又不影响方便地更新操作系统。

总之,Flash Memory 是一种具有诱人的应用前景的新型存储器,从单片机应用系统、个人计算机系统到各类计算机系统,都有它的用武之地。

中国计算机学会
北京市单片机应用技术协会

联合举办“单片机普及函授班”

招 生 简 章

为在我国青少年和科技人员中普及计算机硬、软件知识,推动单片机的普及应用,计算机学会普及工作委员会在举办多期“MP-1型学习机”函授班的基础上,准备将“学装微电脑”的普及活动档次升级,让学习者使用单片机解决他那里的实际问题。为此,中国计算机学会与北京市单片机应用技术协会决定联合举办“单片机函授班”。每年举办两期(春季、秋季),并根据条件与可能,委托地方学会开办函授点。举办函授班的宗旨是,扩大青少年和科技人员的知识面,增强他们的动手能力,以适应社会对新技术、新知识的需求。并在适当的时候举办“学装微电脑”的成果比赛,为将我国的计算机普及活动推向新高度作出不懈的努力。

一、招生对象

具有中等以上文化程度的微电脑爱好者、中学师生、各行业的技术人员、校内外科技辅导员、对单片机应用开发有需要的或有兴趣的业余爱好者等。

二、教学计划与学习要求

每年举办两期函授班(春季和秋季)。每期函授班共四个月,分为四个教学段进行学习。

向每位学员提供学习所需的全部材料,并配备一套“DP-851单片机普及板”作为教学实验设备(含主板、键盘板、实验板、电缆、工具、电源盒、说明书)。

要求学员按各阶段的教学安排寄交作业及实验报告,并提出疑难问题。由指导教师批改作业,解答问题,寄给学员。学习结束后举行开卷考试,按作业和考试成绩发给“全国单片机普及及技术函授班结业证书”。

三、报名和开学日期

春季班报名从10月1日起到12月31日止,2月10日前寄出教材及实验设备。

3月1日开课,6月30日结业。

秋季班报名从5月1日起到7月31日止,8月10日前寄出教材及实验设备。

9月1日开课,12月31日结业。

四、教材及学费

全部学费438元,包括教材费及实验设备费在内。教材包括《单片机普及函授班教材》上、下册,由清华大学吴文虎教授和北京广播电视大学李广弟副教授执笔(共二十多万字)。

实验设备使用北京市单片机应用技术协会研制的“DP-851单片机普及板(专利号:93301539.9)”,提供《DP-851单片机普及板使用说明书》一册。学习结束后,该设备将成为您进行单片机开发应用的得力工具。

五、报名办法

报名者可写信到中国计算机学会办公室,索取“学员登记表”。学员填好后连同学费一并交齐。函授班将发出“录取通知书”及学习用全套教材、教学大纲和教学实验设备。

地址:北京2704号信箱,邮编:100080,联系人:宁伟成

注:计算机学会已指定“电子工业出版社广州科技开发公司”为广东地区的函授指导站。广东的学员可就近报名、邮教具。学会统一教学、发证。联系人:王惠民,邮编:510630,地址:广州市石牌华南师大北区一号203。

模拟乘法器 IC 及使用方法(三)

李兰友

模拟乘法器 IC 广泛应用于测量仪器中。本文介绍一些实例。

1. 可变增益放大器

利用模拟乘法器 IC 可实现电压控制型可变增益放大器。可变增益放大器的原理图如图1所示。

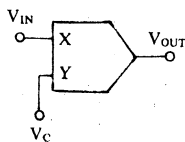


图1

乘法器的输出 V_{OUT} 为

$$V_{OUT} = (X \cdot Y) / 10$$

当 X 输入为输入电压 V_{IN} , Y 输入端加上控制电压 V_C 时,

$$V_{OUT} = (V_C / 10) \cdot V_{IN}$$

当 V_C 的电压在 $0 \sim 10V$ 范围内变化时,则增益可在 $0 \sim 10$ 内变化。

使用 AD534 构成的压控可变增益放大器如图2所示。AD534 的引脚 4 (SF) 通过外接电阻 R_1 和电位器 VR_1 , 将标度因数 SF 调整设定为 $4V$ 。这时,输出 V_{OUT} 计算为:

$$V_C \cdot V_{IN} = 4(R_3 / (R_2 + R_3)) V_{OUT}$$

当 $R_2 = 39k, R_3 = 1k$ 时,

$$V_C \cdot V_{IN} = \frac{V_{OUT}}{10}$$

$$V_{OUT} = (V_C / 0.1) \cdot V_{IN}$$

当 V_C 取 $0 \sim \pm 5V$ 时,增益可在 $0 \sim \pm 50$ 内变化。

本电路中, VR_1 用做增益微调, V_{IN} 输入为 $\pm 5V_{P-P}$, 电容 $C_C = 4700p$ 用做相位补偿。工作频率为 $0 \sim 20kHz$ 。

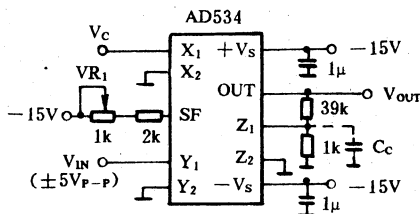


图2 增益在 $0 \sim \pm 50$ 内变化的放大器

2. 射频功率计

射频功率计用于测量信号的功率。当信号电压为直流时,其功率为

$$P = V^2 / R$$

当输入信号为交流时,即 $V_{IN} = \sqrt{2} V_{rms} \sin \omega t$ 时,瞬时功率为 $P = V_{rms}^2 (1 - \cos 2\omega t) / R$, 而其中直流分量为:

$$P' = V_{rms}^2 / R$$

使用 AD834 做成的射频功率计电路如图3所示。高频波的功率由终端负载电阻的功率值表示。这里 $R_1 = 50\Omega$, 当 $V_{IN} = 1V_{RMS}$ 时,功率 P 可表示为

$$P = V_{IN}^2 / 50\Omega = 20mW$$

并用 $20mW$ 时输出电压为 $2V_{rms}$ 设计 IC_2 的增益。

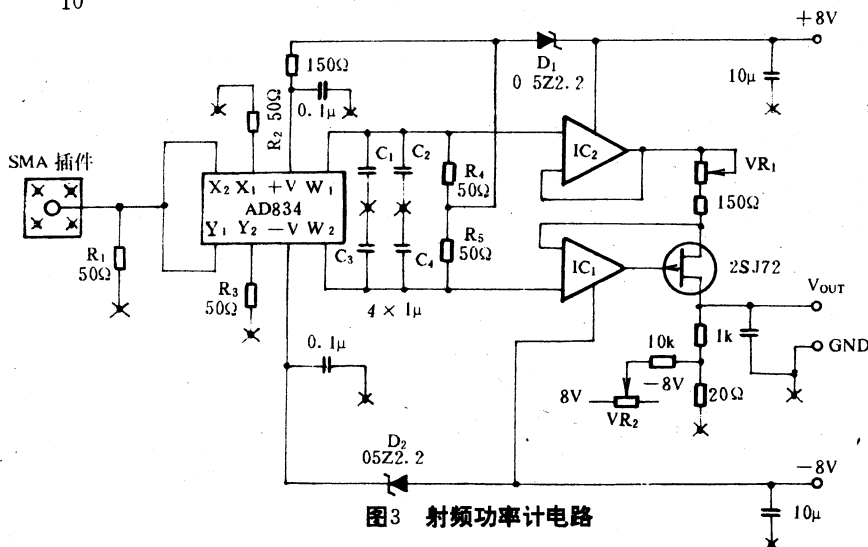


图3 射频功率计电路

由于 AD834 的负载电阻 R_4 和 R_5 为 50Ω , 当 AD834 输入为 $1V_{rms}$ 时, 输出电压 V_A 为 $0.4V$ 。

电路中 $C_1 \sim C_4$ 构成低通滤波器。 $C_1 = C_2 = 1\mu F$ 时, 可测量 $100KHz$ 附近的信号, 当测更低频率时, 可并联 $10\mu F$ 的钽电容。

因设定 $V_A = 0.4V$ 时 $V_{OUT} = 2V$, 因而 IC_2 的增益 G 设定为 5 , 即 $G = 5$ 。

如图所示, G 可由下式计算:

$$G = R_7 / (R_6 + VR_1)$$

因而, 通过调节 VR_1 , 可使 G 在 $4 \sim 6$ 内变化。

电位器 VR_2 用于失调电压调节, 调节范围为 $\pm V_s (R_{10}/R_{11})$ 。

图4为图3电路的输入—输出特性, 图5为频率特性。从图中看出, 本射频频率计可用至 $200MHz$ 附近。

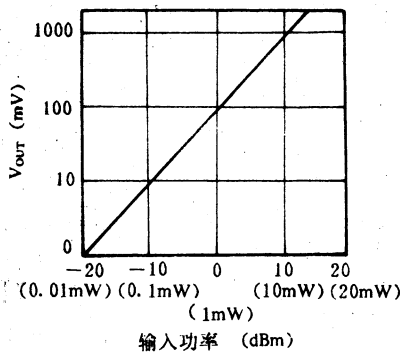


图4 输入输出特性

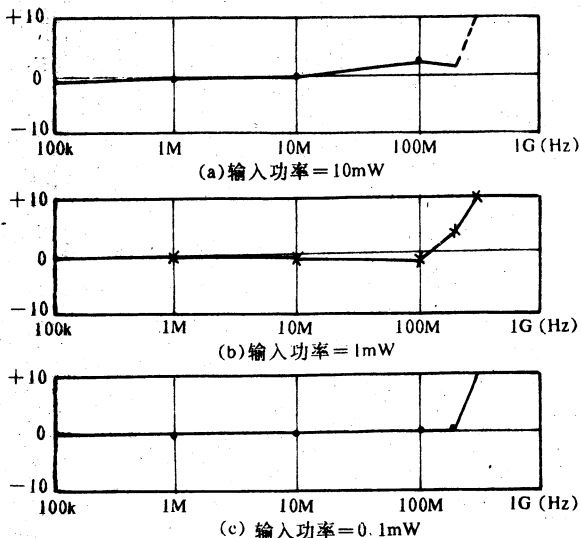


图5 实测频率特性

本仪器采用 $\pm 8V$ 电源供电, 经稳压管 D_1, D_2 降压 ($2.2V$ 左右), 使 AD834 的工作电压为 $5.8V$ 左右。这样, 通过在 $+V$ 端外接 150Ω 电阻, 使 $+V$ 端子电压为

$$5.8V - 150\Omega \times 0.011 = 4.15V。$$

电路调整步骤如下:

- (1) 在无输入信号状态下, 调节 W_2 使 $V_{OUT} = 0$
- (2) 输入 $1V_{rms}$ 信号, 调节 W_1 , 使 $V_{OUT} = 2V$ 。

电路设计应注意以下两点:

- (1) 良好接地。
- (2) SMA 接插件连接。如图6所示, 将电缆屏蔽外皮分四部分分别接地, 中心导体尽量少裸露在外面。

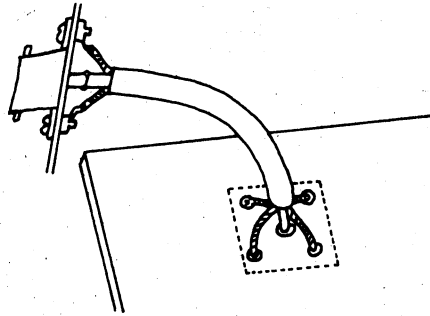


图6 SMA 接法)

(待续)

经验二则

我使用的是北方电脑公司生产的 BF-PC-BOY 型微机, 原机配的显示卡是单色显示卡, 在运行具有图形显示方式的软件 (如游戏软件、教学应用软件及自编软件) 时, 必须先运行图形转换软件 (如 MKB、MKF 等)。最近我试用了一块双频显示卡, 效果不错。在启动 DOS 后, 不用运行图形转换软件就能直接运行绝大多数 CGA 显示方式的软件。但是在进行汉字操作 (如 CCDOS 4.0) 时, 由于是 CGA 显示方式, 所以每屏只显示 11 行, 使用起来很不方便。由该机的使用说明可知, 主机板上靠后部的 DIP 开关的第 5 位和第 6 位是设置显示方式的, 即: 5 为 ON 及 6 为 OFF 时是图形方式。我在外壳前面板上增加了一个小开关, 用来控制第 5 位的通或断。当要运行图形软件时, 将开关闭合; 当要进行汉字操作时, 则把开关断开, 而每一次拨动开关, 都必须按一下 RESET 按钮。请注意双频显示卡的跳线应设置为 CGA 方式。经上述改动后, 使用起来就非常方便了。

另外早些时候购买的 BF-PC-BOY 机都没有高速主频显示灯。我经试验后发现主机板上实际上已有高速主频显示灯的接口, 只是没有使用。第 6、7 两位就是高速主频显示灯接口引线, 6 为正, 7 为负。在前面板上原已有发光二极管位置, 但是没有开孔。细心用小刀刻一长方形孔, 能放入小长方形发光二极管就行, 然后焊上导线接到第 6、7 位, 改动即完成。当主频为高速时, 新装的发光二极管应发光。主频的切换仍使用同时按下 Ctrl 键 Alt 键和小键盘上 + 键的方法实现 (王万春供稿)。

微机通信网络的发展状况及趋向

薛兴华

一、微机通信网络的基本组成

1. 网络的基本构成

微型计算机通信网络是计算机技术和通信技术相结合而形成的一种通信新方式,主要是满足数据通信的需要。它将不同地理位置具有独立功能的微机、终端及附属设备,用通信链路连接起来,并配备相应的网络软件以实现通信过程中的资源共享而形成通信系统。

一个微机网络由通信子网和资源子网构成。通信子网分为传输介质和通信设备,传输介质既可以是专用的双绞线、同轴电缆及光缆等,也可以是公用通信线路,如电话线、出租电缆等。局部网使用闭路电视网线路效果甚佳。通信设备是指通信处理机、传输线路、交换设备和调制解调设备,以及用于卫星通信的地面站、微波站、集中器等。

资源子网由各种用户计算机和终端组成,通过通信子网来共享资源。它除带有数量众多的终端机外,还有计算机系统本身具备的磁盘机、磁带机、绘图仪等设备,与各网络节点连接的主机构成一个完整的计算机系统。终端设备可以不经主机直接连到结点通信处理上。在局部网中,可将公用的大容量磁盘机直接接到通信子网中供用户使用。国内的微机和小型机可以通过通信子网享用网内大型机的激光打印机、数据库内的信息,或软件包内的程序;或是分担负荷,当某一计算机任务过重时,可以把一部分任务交给网内空闲的计算机协助处理,以达到分布计算和分布处理的目的。

通信子网的主要功能是进行数据传输、交换及通信控制;资源子网的主要功能是提供网中共享的硬件、软件和数据等资源,并进行数据处理。通信子网把资源子网中的各种资源连接起来,便可实现资源子网中各种资源之间的信息交流和资源共享。在资源子网内的用户计算机称为主机,在通信子网内的分组交换机称为结点机。

如果微机用户距离交换局较远,而且各用户终端相当集中地聚集在一起,则可采用多路复用器把一些邻近的微机用户线首先集中在多路复用器的输入侧,然后将各用户的信息用频分复用或时分复用的方式送入网内的本地交换局。如果在同一条微机用户线上连接有若干个数据终端,则可采用集中器定期地轮流询问各个终端,各个终端只有当被查询时才有资格发送信息。

2. 微机网络的分类

微机网络是在公用电话通信网基础上发展起来的。由于在电话网上采用调制解调技术使得数据通信逐步形成网络。

按通信距离分,微机网络分为局部地区网络和远程网络。局部网一般在几百米到十公里左右距离之内,如在一个工厂、一所大学、一幢大楼范围内。而远程网络一般在几百公里、几千公里或几万公里。

按网络使用目的分,有共享资源网络、数据处理网络、数据传输网络和计算机复合系统(LCN)。

按网络结构分,有星形网络,以一台计算机为中心,以放射状连接若干终端或微机;环形网络,信息传输线路构成一个封闭的环,环中每个结点有一个站和一个中继转发器;总线网络,如以太网,各计算机或直接连网设备都经收发器连到电缆上;分布网络,各结点之间有多条路径相通。

3. 微机网络资源共享

建立微机通信网络的目的之一就是实现资源充分共享。这些“资源”是指网中所具有的硬件(各种类型的计算机、终端及其配套设备)、软件(各种规程、协议、程序和操作系统)和数据库等。资源共享是指微机通信网络内所有用户均能享受和使用网中资源的任意部分或全部,其目的是为了将地理位置分散的多台计算机实现实时地、集中地数据信息处理。

(1) 共享硬件资源

对微型用户,可通过网络分享大型计算机主机或特殊的外围设备,可降低建设网络工程投资。主要有利用硬件必不可少的软件来处理数据;送去软件及数据;共享某类硬件;共享计算机本身及必不可少的软件及只共享计算机硬件本身。

(2) 共享软件资源

可以避免软件研制开发工作的重复劳动,一套完整的网络软件系统可供计算机通信网中各个用户共同使用。软件资源共享有两种:一是通过共享的软件进行数据处理,二是以子程序调用方式利用共享软件。

(3) 共享数据库资源

将同一类型的文化、数据和用户手册供大家使用,如建立电话簿的单位、电话号码等数据库,供电信部门等调用、备查及其它终端用户查找。这样可以避免数据或文件在多处重复设置,如科技情报资料部门可将大量的文献存入大容量的存储器内,别处需要时只调用所需要的部分即可。

微机网络不仅可以满足局部地区的一个公司、企业、学校和办公机构等的的数据、文件传输需要,而且可以在一个国家甚至全世界范围进行信息的交换、储存、处理,同时可以提供话音、数据和图像的综合服务,具有诱人的发展前景。现在,计算机通信网络已从单机运行和脱机远程通信发展成面向终端的联机系统,这

种联机系统可以实现资源共享和多重处理。目前广泛使用的计算机通信网络实质上是面向微机终端的联机系统中的主机用通信线路连接而形成的计算机网络。

应当看到,没有联机网络,计算机终端用户就很难得到服务。信息只有在运行畅通的动态网络中,才能最大限度地发挥其社会效益。只有建成一个覆盖面广、能够高效地收集、加工、处理、存储反馈信息,并有一套完整科学的信息管理制度网络,才能创造一个数据库服务的良好环境,采集和分配好数据库资源,为用户提供利用的机会。

二、微机通信网络的交换方式

计算机网络与数据通信按照使用方式可分为电路交换网、报文交换网和分组交换网。这些交换网都是遵循国际电报电话咨询委员会(CCITT)建议的统一标准和国际标准化组织(ISO)制订的标准化建议开发建设的。计算机及数据通信网由相互连接的节点集合构成,其中数据从源点经由节点网传输到终点。节点由传输路径连接。从一个站进入网络的数据,通过节点转发至终端。

1. 电路交换网

电路交换网是一种高速、高质量的、可与任意对方进行数据通信的网络,适用于高密度和较长电文的数据通信及数字传真通信等。

电路交换方式与电话通信相同,但速率和质量比电话交换高得多。其传输信道采用PCM技术的数字信道,对各种速率的通信电文实行时分多路复用后再传输。交换设备使用时分交换机,可直接交换多路复用数字信号的通信电文。

电路交换的通信有三个阶段:

* 电路建立

在任何数据能够传送之前,必须建立一条端到端(站到站)的电路。

* 数据传送

数据可以是数字的(如终端到主机)或是模拟的(语音);信道及传输也可以是数字的或模拟的。连接是全双工的,数据可沿两个方向传输。

* 电路拆线

经过一定时间的数据传输之后,连接要终止,一般是由两站中的一个站进行拆线。

电路交换的连接路径是在数据传输开始之前建立的,路径上每一节点间必须保持信道容量,每个节点必须具有可用的内部交换能力以处理所“请求”的连接。交换机应具备建立路由的智能。电路交换网进行数据交换时,两个站都必须同时可用,两站间的网络部分具备专用资源。

2. 报文交换网

报文交换是另一种适于数字数据交换的方式,如电报、电子邮件、计算机文件及事务查询应答等。它不必象电路交换那样在两个站间建立一条专用路径。在电路交换网中,每个节点是一个电子交换装置或者机电交换设备,而报文交换节点则是一个通用小型计算

机,带有足够的外存,以便在报文进入时进行缓冲存储。

报文交换比电路交换有许多优点:

* 链路效率提高,一条节点到节点的信道可以在时间上由许多报文共享;

* 网络可以在接收机使用前存储报文;

* 可以将一份报文发至多处终点,复制报文抄件;

* 一个节点有若干报文排队等待传输,可优先发送某种报文;

* 可对报文进行差错控制和校正;

* 能根据速率和码制转换,实现不同数据率的两个站的联系;

* 对送到的无用报文可以截住,或存储,或发至别处终端等;

但报文交换网也存在一些不足之处,如经过网络的延迟较长,差别较大;不适于实时及会话式通信,也不适用会话式终端——主机间连接。

3. 分组交换网

电路交换在每次通信前都要有一定的接续时间,整个交换设备和信道在接通时间内只能为一对用户服务,设备和信道利用率低,对高速计算机来说,浪费很大。报文交换虽然利用缓冲存储器,但是由于电文长度不一,存储容量得不到充分利用,在克服电路交换和报文交换缺点基础上发展起来的分组交换网,收发信息不是直接在终端之间相互传递,而是从始发端发送的信息先在网内交换机中存储,分成一定大小的“分组”,每一组再通过网络进行高速传输,送至接收终端。这种新型交换方式近几年得到各国的广泛使用,是目前世界计算机网络重点开发应用的交换技术。我国的公用计算机通信网络也是采用的分组交换网。

分组交换技术在进行工作时将每份较长的电文划分成某一固定长度的若干段,每一段电文加上报头(标题)组成一个“分组”,在同一条信道由依次传送不同来源的不同分组,这样既节省了缓冲存储器容量,提高了缓冲存储器的利用率,又加快了信息的传输速率,能更好地配合计算机工作。其主要优点有:

* 线路利用率提高

用户数据是以继续的一个一个分组的形式在网上交换节点机之间的数据通道上传输的,用户不发数据时不占用交换机之间的数据通道,这时线路可供其它用户传输数据分组,故线路利用率明显提高。

* 允许不同速率、码型和规程之间的终端互通

分组在交换机之间是以存储转发方式传输的,交换机可以对速率变换、电码变换、规程变换等加以处理,实现不同速率、码型和规程的终端设备互通。

* 实现多种类型计算机互连和数据的共享

分组交换网的建立可实现不同种计算机终端之间的互连,在网上连接各种类型的数据库,以提供资源共享,并可与PSTN网、Telex和其它电路交换网互连,增强了网路的灵活性。

* 具有高质量的服务功能

由于分组在传输过程中提供差错检测和重发功能,故可提供高质量低误码率的传输,特别适用于银行业务、远程加载和数据处理等服务。

* 能提供低成本的数据交换业务

分组交换的资费多数国家采取与距离无关的政策,只计算占用电路的时间和通过的业务量,一般比电路交换和租用专线便宜。

4. 网络交换技术比较

电路交换	报文交换	分组交换
专用传输路径	无专用路径	无专用路径
连续数据传输	报文传输	分组传输
对会话式足够快	对会话式太慢	对会话式足够快
不存储信息	存储报文以备重发	分组存储到投送完毕
对整个会话确立路径	对每份报文确定路由	对整个会话确定路由
有呼叫建立延迟,可忽略传输延迟	有报文传输延迟	有呼叫建立和分组传输延迟
如被叫忙则发忙信号	没有忙信号	如分组未能投递可通知发送者
机电式或计算机化交换节点	报文交换中心带外存设施	小的交换节点
固定带宽传输	动态使用带宽	动态使用带宽

从电路交换、报文交换和分组交换的技术比较看,电路交换对于大量的会话式数据通信来说,其数据通道的利用率低,不同速率、码型和传输规程的用户之间不能互通。报文交换存在对会话式终端——主机间连接不适用、经过网络延迟较长等缺点。因此,从80年代中期至今,绝大多数国家在兴建微机数据网络时都是采用的分组交换技术。随着综合业务数字通信网(ISDN)的进一步实用化,分组交换业务将进入ISDN,并发展成为最新的宽带分组交换——异步转移模式(ATM),使数据传输和交换的质量明显提高。

三、微机通信网络的国际规程

整个微机网络的数据通信是由很多节点和连接节点的传输路线组成。微机网络使得一个地方的数据处理用户可以使用另外一个地方的数据处理功能或服务内容。在一个网络内,不同层次之间进行通信,遵循接口协议。两个网络连接对应发生相互作用,遵循同层协议。

数据通信是在各种类型的用户终端和计算机之间、以及不同型号的计算机之间进行的。它和电话、电报通信方式不同,在电话通信中,用户终端只是电话机,在电报通信中只是电报机。而数据通信规程要能适应各种类型的数据终端。数据通信的终端设备是计算

机,网内控制设备也是计算机。

微机网络的数据通信大多是利用现有的电话网和用户电报网、新型数字通信网或分组交换网,故用户终端进入数据通信网的形式多种多样,这就需要有各种相应的接口,加上用户使用的计算机用途不同,网络须作出相应的配合,而计算机网络通信具有全球性,故对数据通信的规程应具备统一性和兼容性。为统一国际标准建设公用计算机网络,CCITT 专门为利用公用电话网的数据通信提出了 V 系列建议;为利用数字通信网和分组交换网的数据通信制订了 X 系列建议。

适用于公用数据通信网常用的 CCITT 建议规程有:

X.1 公用数据网国际用户业务分类

X.2 公用数据网内的国际用户业务功能

X.3 公用数据网内分组装/拆(PAD)功能

X.20 公用数据网起止式传输业务用的数据终端设备和数据电路终接设备之间的接口

X.21 公用数据网内同步式数据终端设备和数据电路终接设备之间的通用接口

X.25 公用数据网内分组式数据终端设备和数据电路终接设备之间的接口

X.75 在分组交换的公用数据网的国际电路上用于传送数据的终端和经转呼叫的控制规程

X.121 公用数据网的国际编号制度

X.150 公用数据网络中 DTE 和 DCE 的测试环路

1984年国际标准化组织 ISO/TC97提出了计算机网络的7层协议,即开放系统互连型(OSI),为各国制造厂家开发生产网络产品和为用户选用分组交换网提供了兼容标准,促进了分组交换网的发展。现在,北美、西欧和日本使用分组交换网十分普遍,计算机联网范围不断扩大,互连性和共存性日趋普及。1988年美国联邦政府就规定,政府部门购买计算机必须具有 OSI 功能。到1990年,OSI 已正式成为联邦信息处理标准。现在美国 IBM、AT&T、DEC、CDC、DG、HP、Telonet、Unisys、王安公司等计算机厂家,为获得高速数据传输速率,都宣传他们的产品具有“开放系统互建”标准。

四、微机通信网络的发展趋向

现在,计算机通信技术、数据库技术和基于两者基础上的联机检索技术已广泛应用于信息服务领域。传统的以报刊、人工采集、会员单位组织的信息服务方式已逐步被以数据库形式组织的信息由用户联机检索所替代,信息量和随机性增大,信息更新加快,信息价值明显提高,信息处理和利用更加方便。世界各国都公认,计算机网络与数据通信是90年代信息服务业大有发展前途的领域。

自从80年代中期以来,计算机网络发展十分迅速,现在各国都通过建成的公用数据网享用各类数据库的资源,为发展高新技术和国民经济各个领域服务。它作为联机检索中心和用户之间的纽带,使数据交换业务更加具体实用化。已投入运营的部分网络见下表:

名称	开办国家	简况
SITA	法国巴黎	供航空公司航线管理使用
TYMNET	美国	国家医疗图书馆使用
TELENET	美国等14个国家	公用网,约200个节点可通过电话网访问
CYLIX	美国	公用卫星网
DECNET	美国	私用网,DEC
ES	BELL	电子交换
DATAPAC	加拿大	公用网,可通过电话网访问
VENUS	日本	公用网
EPSS/IPSS	英国	国际分组交换服务
Transpac	法国	可通过电话网访问,最大数据长度为128、256字节等
Euronet	欧洲共同体(9国)	可通过电话网访问,以欧洲情报网(EIN)技术为基础
CNPAC	中国	公用网,北京、上海、广州设节点交换机

进入90年代后,世界微机网络及数据通信的发展方向已十分明确,即从低速数据传输向高速发展,以适应高质量信息通道的需要;从专用网络向公用网络过渡,以适应计算机终端日益普及和进入国际数据网的需要;从面向终端的数据通信发展到计算机联网;普遍采用分组交换技术,并向快速分组交换发展;从单一的分组交换网进入综合业务数字网(ISDN),并实现宽带分组交换,即异步传输模式(ATM)。今后微机网络及数据通信发展趋势主要表现在以下几方面:

1. 普遍采用分组交换方式

分组数据交换(Packet Data Switching)正向低成本、长距离传输分组数据网结构发展,以扩大数据业务(Data Service)的服务区域。在大城市将设置多台分组数据交换机(Packet Data Exchange)今后几年各国将重点开发微型分组交换机,用微处理机制节点机,这样具有体积小、操作灵活、经济方便、便于施工和维护。分组交换网将通过卫星通信来传输,使数据传输(Data Transmission)手段更加先进,交换网将形成连续性业务和分组交换业务都能处理网络的布局。卫星通信将沟通许多国家境内的节点(Node),使分组交换网的用户能够利用世界各国的计算机,将电文传送到世界各

个角落。

用户接口计算机将与交换计算机分开,具有完全不同的功能,用户接口计算机将把所有的终端设备(Terminal Equipment)的传输转换成一种标准的方式、代码和规程,使目前完全不兼容的终端能互连。用户接口设备将能够与普通传真机或其它模拟设备互相传输;接口计算机将能在传输前把电文压缩,以提高传输效率。此外,还能处理分组无线电终端,系统内可接入便携式数据终端。

2. 从单一的数据网向综合业务数字网方向发展

现在,综合业务数字网(ISDN)已成为计算机网络最重要的发展领域。实现 ISDN,第一步是在电话网中从用户到用户完全数字化,目前欧洲国家、美国、日本等进展极快,到去年底已初具规模;第二步在 ISDN 中提供分组交换业务。从各国发展趋向看,ISDN 的分组交换业务最终将取代现有的公用分组交换网;单一的数据网将过渡到 ISDN。

ISDN 是计算机与通信相结合的产物。它不仅能传送语言信息,而且还能够传递数据、字符、电视、图像、传真等各种形式的信息。ISDN 实现了信息收集、传送、处理和控制在一体化,并具有高度数字化、智能化和综合化。用户只需占用一个号码就可以在一条电路上同时打电话,使用传真机发送图表、文件,通过计算机终端设备调用有关数据库资源、检索需要的资料及处理信息业务等,是90年代各国开发建设的重点信息技术领域。

3. 实现宽带综合业务数字网——异步传输模式(ATM)

随着计算机网络技术的发展,ISDN 采用电路交换和分组交换两种方式来实现多样化的高级通信业务已暴露出不足之处,如宽带 ISDN 是固定的64Kb/s 速率,难以实现活动图像、高速数据等宽带业务(Wideband Service)。应运而生的 ATM 克服了电路交换和分组交换的局限性,是发展高速 B-ISDN 最有效的交换技术,被公认是21世纪微机通信网络理想的传输、交换方式。CCITT 已确定将 ATM 作为组建 B-ISDN 的最佳传输方式。

ATM 采用异步时分复用(ATD)技术,实质是快速分组交换(FPS)方式。它适用于语音、数据、高清晰度电视(HDTV)等各种宽带通信业务。近几年,各国为适应巨大的微机用户市场需要,明显加快对 ATM 交换技术的研制开发,都力图尽早解决 B-ISDN 的关键技术,顺利建成 B-ISDN,实现计算机通信网络传输交换的现代化。

4. 开发更多的用于联机网络的数据库

与计算机通信网络相辅相成的是数据库的开发和建设。离开数据库资源,微机网络为用户服务就成为空谈,网络建设就失去意义,因此开发更多的数据库是今后微机网络追求的目标之一。

近几年世界上用于联机网络的数据库发展迅猛增长,平均年增长率为50%。美国是数据库生产发展最快

的国家,占全世界商用的70%。日本数据库商用服务发展也十分迅速,据统计,12家主要的联机服务的数据库生产厂家拥有的注册用户口令数由1990年的9万个增长到1991年的20多万个;26家主要数据生产兼代销的联机服务公司的联机用户口令数已达到30多万个。日本

数据库业务的90%以上是通过联机来提供的,用户使用的主要设备是微机和电话线,通信速率主要是1200bps。数据库资源将使微机通信网络大放异彩,微机终端功能更加神奇。

如何升级你的网络工作站

山东济宁医学院附属医院计算机室(272129) 薛强

MSDOS Ver5.0无疑是一个极为成功的操作系统,它的出现被称为MSDOS历史上最重大的一次升级。它把人们对DOS 4.0的不满情绪一扫而光,充分显示出Microsoft公司的大家风范。国内用户纷纷将其操作系统升级到5.0,它已经成为新的标准平台(以前是DOS 3.3)。其中大多数用户是从MS-DOS 3.3直接升级到5.0的。然而也出现了一些问题,有些汉字系统和应用软件在DOS V5.0下不能得到很好的支持(已有一些文章专门介绍这方面的问题)。更为严重的是:由NOVELL公司推出的风靡世界的NETWARE-386 Ver3.10、3.11系列没有提供与DOS 5.0的接口(当时DOS 5.0尚未推出),而NOVELL网络目前在世界范围内的市场占有率已达80%,拥有相当数量的用户。网络用户只能使用DOS 3.3或DOS 4.01。而无法享受DOS 5.0的诸多强大功能。笔者作为一网络用户,对此深感遗憾。为了让广大网络用户能充分享用这一高版本的操作平台,真正做到与世界水平保持一致,笔者对DOS 5.0和NETWARE386 V3.10、V3.11进

行了深入的研究,发现了一个简单的修改方法,该方法极为安全而有效,而且修改后的NETX.COM对网络系统和DOS系统无任何影响。以NET4.X为例,仅须修改两个字节即可。

具体方法如下:

```
COPY NET4.COM NETX.COM
DEBUG NETX.COM
-E9533 E9↓
-EAF73 E9↓
-W
-Q
```

需要指出的是经过修改后的NETX.COM在原DOS 4.X下可完全正常运行,并在理论上支持DOS的未来版本,真正成为NETX.COM。

笔者经过近三个月的使用,证明效果良好,有兴趣的读者不妨试一试。

修改BIOS程序实现对PC机的加密

北京 徐文军 李晓中

一、问题的提出

在计算机应用的过程中,为了系统和文件的安全,一些用户采用了系统、磁盘、文件等加密技术,对于一些专用系统(如教学训练中的某种专业训练系统),不需要操作者(学员)使用自己的软盘,而只用系统专用盘片。如果要防止操作者使用自己的磁盘或无关的软件上机运行,采用以上几种加密方法难以解决。为此便提出了这样的设想,能否通过修改PC机的BIOS芯片中的程序,使其在自举前具有识别非法使用者和非法引导盘的能力。

通过对ROM BIOS程序的分析,发现实现这个设

想是可能的。ROM BIOS芯片中装入BIOS程序时,还有一些空间是闲置的。以“0”填充,我们可将增加的口令字程序和确认引导盘的程序分块存放于这些闲置区中,在BIOS POST执行完并开始自举时,转向我们增加的程序。这样便增加了口令字功能和确认引导盘功能。如是系统专用启动盘,则正常实现自举(执行INT 19H),否则系统将锁死。

二、实现方法

识别非法使用者是通过在BIOS程序中增加一段“口令字”识别程序来实现的,在BIOS地址E420H处有指令

MOV BYTE PTR[003F],01

INT 19H

我们将其改为:

E420 JMP E6D2

E423 DB 4A

E424 DB 59

地址 E6D2 处便是口令字程序,为了程序的灵活性我们增设了一个选择以确定是否执行口令字程序。这个选择是通过 E423 处是否为 594AH 为实现的。确认非法系统盘是通过将系统盘格式化为非标准化磁盘来实现的(如双面倍密盘格式化为 42 道)。进入系统时,首先对 A 驱进行读操作,若读 42 磁道成功则表明为合法系统盘,否则为非法系统盘,提示磁盘错误,系统锁死;若是合法系统盘,则去执行 INT 19H 自举程序。

在 BIOS 芯片中无足够大的区域将整个程序存放在一块,因此采用了分块存放的方法,通过无条件转移指令联系在一起。

本文介绍的方法是在 JUKO 2.30 芯片基础上实现的,对于其它芯片,则读者需要分析 BIOS 程序以便找出适当的空间存放增加的程序,又不损坏 BIOS 功能。

用户可能要提出,非法使用者可以在系统盘启动后再使用自己的软盘,对于这种现象的预防,则要修改 INT 13H 中断,在读/写等磁盘操作时,先检查磁盘的第 42 道内容如果符合约定,则进行正常的 INT 13H 操作否则警告用户此盘为非法用户盘,且系统自动退出。

附反汇编程序如下:(段址为 0F000H)

```

E420 E9AF02    JMP     E6D2
E423 4A       DEC     DX
E424 59       POP     CX
E425 CD19     INT     19
E6D2 C6063F0001 MOV    BYTE PTR [003F],01
;判口令字
E6D7 0E     PUSH   CS
E6D8 1F     POP     DS
E6D9 B04A    MOV    AL,4A
E6DB B459    MOV    AH,59
E6DD 2E     CS;
E6DE 3B0623E4 CMP    AX,[E423]
E6E2 7403    JZ     E6E7
;判是否执行口令字
E6E4 E92911    JMP    F810
E6E7 8D368AF0 LEA   SI,[F08A]
;口令字提示串地址
E6EB 8307    MOV    BL,07
E6ED E91C01    JMP    E80C
E6F0 CD19     INT     19
E80C AC     LODSB
E80D 08C0    OR     AL,AL

```

```

E80F 7406    JZ     E817
E811 B40E    MOV    AH,0E
E813 CD10    INT     10
E815 EBF5    JMP    E80C
E817 B108    MOV    CL,08
E819 BE08F8  MOV    SI,F808
;口令字存放地址
E81C 30E4    XOR    AH,AH
E81E CD16    INT     16
E820 3A04    CMP    AL,[SI]
E822 7402    JZ     E826
E824 EBF5    JMP    E824
E826 46     INC    SI
E827 FEC9    DEC    CL
E829 75F1    JNZ    E81C
E82B E9E20F  JMP    F810
;判非法系统盘
F810 B008    MOV    AL,08
F812 B52A    MOV    CH,2A
F814 30C9    XOR    CL,CL
F816 31D2    XOR    DX,DX
F818 31DB    XOR    BX,BX
F81A 8EC3    MOV    ES,BX
F81C BB007C  MOV    BX,7C00
F81F B402    MOV    AH,02
F821 CD13    INT     13
F823 7303    JNB    F828
F825 E90207  JMP    FF2A
F828 B90008  MOV    CX,0800
F82B B88787  MOV    AX,8787
F82E 26     ES;
F82F 2907    SUB    [BX],AX
F831 7403    JZ     F836
F833 E9F406  JMP    FF2A
;提示非法系统盘
F836 43     INC    BX
F837 43     INC    BX
F838 E2F4    LOOP  F82E
F83A E9B3EE  JMP    E6F0

```

在地址 E089 处有指令:

```

E089 7401    JNZ    E08C
E08B       HLT
E08C.....

```

应改为指令:

```

E089 EB01    JMP    E80C
.....

```

这是因为 BIOS 要对自身进行校验,若发现有错或被修改,便要死机。所以必须修改。

由于篇幅有限,本文只提供了口令字和读盘的主要程序,没有提供完整的程序,读者可以参考这个思路,编写适合自己口味的程序。

修改软金山系统,适应有缺陷的386主板

南宁教委信息中心(530022) 文 森

日前,笔者在市场上发现有一批廉价的386主板,金山汉字系统5.10版在其上不能使用。现象是运行SPLIB.EXE后,再运行SPDOS.COM,屏幕提示“Super Chinese Card— I has not been installed”,对不带卡的金山系统而言,意思就是字库未载入。经跟踪分析,笔者修改了金山系统的SPLIB.EXE,SPDOS.COM两个文件,避开了主板缺陷,解决了上述问题。现将有关原理及修改过程公诸报端,以飨同行。

这种主板的主要缺陷在系统内存的0000:04feh位置。查一查手册我们可以发现内存地址0000:04f0h—0000:04ffh是DOS约定的应用程序通信区。金山5.10版就使用了其中的0000:04fch—0000:04ffh四个字节在程序间通信,由SPLIB.EXE在检查、载入字库后写入有关的标志和地址信息。SPDOS.COM程序要在检查这些标志和地址信息无误后方能往下执行。而在这种386主板上,0000:04feh这一地址的内存不能写入,固定为90h,这就是导致金山系统不能运行的原因。值得一提的是,由于IBM PC系列微机在自检时,对前16K的内存是默认正确的,所以这一缺陷在机器自检时是不能发现的。

找到原因后,笔者便对SPLIB.EXE和SPDOS.COM进行修改,将他们使用的通信区地址从0:04fch

—0:04ffh移至0:04f0h—0:04f3h,从而避开了缺陷点,使金山系统得以正常运行。

使用PC Tools进行的具体修改过程如下:

1. 运行PC Tools,在文件状态下移动光条选中SPLIB.EXE文件。
2. 按“F”键选择“搜寻(FIND)”功能。
3. 按“F1”功能键切换到16进制输入方式,键入“BEFC04”。
4. 当串找到后,按“E”键进入修改方式,将“FC”改成“F0”,按“F5”功能键确认后,退到文件选择状态。

再选文件SPLIB.EXE,重复上述操作,所不同的是这次搜寻的串为“BFFC04”,同样将其中的“FC”改为“F0”。

选择文件SPDOS.COM,重复上述操作2—4。不同的是搜寻的串改为“A1FC04”,同样将其中的“FC”改为“F0”。

再选文件SPDOS.COM,重复上述操作2、3,搜寻的串为“1EFE04”,找到后将“FE”改为“F2”,确认后退出PC Tools。经过以上修改,你的金山5.10版就可以在以这种386主板为基础的机器上运行了。同时,从修改原理我们不难看出,修改后的金山系统,照样可以在主板无缺陷的微机上运行。

CCED 使用次数的修改

武汉铁道部车辆工厂(430012) 朱 融

CCED编辑软件,因其使用方便、直观,深得用户的好评。本人的一套CCED(Ver01—15—1989)使用时间长了以后,屏幕提示“CCED使用期限已到,需要更新版本?”,此后无论键入什么字符均不能进入系统。经分析该系统中的某一文件肯定设置了计数器,每调用一次则次数减一,直到不能使用为止。

根据以上分析,在A盘上建立两个子目录C1和C2,将能够继续使用和使用期限已到CCED分别拷入这两个子目录中,CCED一共有九个文件,用PC Tools中的文件比较功能(O功能)分别对这九个文件进行比较,结果发现除CCED.EXE文件以外,其余文

件内容均相同,由此可知,计数器就在该文件中。

这两个文件中有两个字节的内容不一样,屏幕显示如下:

```
CCED.EXE (on Drive A) is being COMPARED to
CCED.EXE (on Drive A)
```

```
File MISMATCH in logical sector num 0000160 at displacement 008
```

```
Mismatched bytes: "0" (30 (in 1st file)) "9" (39) (in COMPARE TO file)
```

```
Press any key to continue
```

```
File MISMATCH in logical sector num 0000160 at displacement 009
```

```
Mismatched bytes: "1" (31) (in 1st file) "9" (39) (in COMPARE TO file)
```

```
Press any key to continue
```

使用PC Tools中的文件编辑功能(E功能)根据以上地址进行修改,将使用期限到期的文件改为“99(3939)”,这样您的CCED又可使用99次了。

对于其它类似的问题可以采用相同的方法进行处理。