

# 电脑 2 1990

中国软件行业协会会刊



## AD-386/ 33 型

- 80386-33 CPU 2 M RAM 64 K 高速缓冲
- 1.2M+1.44M软盘驱动器
- 100M硬盘
- 1024×768彩色显示器及VGA卡
- 软件、资料各一套
- AD-386系列机有25型、20型、16型等，配置与上相同

AD-386/25型

AD-386/20型

AD-386SX/16型

## AD-286/ 16 型

- 80286 CPU 1M RAM
- 1.2M+360K软盘驱动器
- 40M硬盘
- 1024×768彩色显示器及VGA彩卡
- 软件、资料各一套



## 广州经济技术开发区 夏港综合服务公司

本公司讲求信誉，用户第一，AD-386系列机全部免费保修两年，AD-286系列机全部免费保修一年。本公司售出机器除免费保修期间之外，其余均终身保用，如有维护，只收零件费用。本公司在南京、长沙、北京、哈尔滨都设有代理点。本公司并承接外来机器维修业务。

地址：广州市天河路33号之七

电话：752397

邮政编码：510075

# 电脑杂志社

广州经济技术开发区科技开发经营部

## 向《电脑》读者致意

应广大读者要求，为沟通厂家和读者的联系，电脑杂志社开办了科技开发经营部。在广大读者的关怀和支持下，我们将把该部办成技、工、贸一体的企业，为我国的计算机普及作出贡献！该部本着用户第一的原则，将开展优质服务。

### I、为您提供：

- 一、计算机整机、外设、零配件销售
- 二、承接计算机硬件、各种软件开发及网络工程。
- 三、提供维修服务。
- 四、售前售后技术培训。
- 五、为读者提供购机指南及其它咨询服务。
- 六、组织新产品开发，将读者科研成果转化为商品。
- 七、可组织和主持读者科研成果的鉴定会。
- 八、代理各专业厂家名牌产品。

### II、现货优惠供应

#### 1. PC 机每套 3000 元

配置：主频4.77 / 12Hz、640K内存、2×360K软驱、101键键盘及12寸"单显。

欢迎惠顾、垂询。

#### 2. 陕西厂中华学习机

地址：广州市德政北路 393 号

电话：330644      邮编：510055

广州市五山路科技东街 49 号

电话：516911-3273      邮编：510630

# 当今国内最理想的高抗干扰稳压净化电源产品



●电脑及高精尖娇而昂贵电器设备的最佳稳压电源●



## 铁塔牌 CWY 系列交流参数稳压器

产品符合国际标准【粤电采标证 (1989) 第 054 号】

●铁塔牌 CWY●高抗干扰●高可靠●长寿命●性能优●用途广●计算机必备●

## 北京亚运会和深圳大亚湾核电站选用产品

### ▲稳压范围宽:

单相 150~260V (实际可工作范围达 120~300V)

三相 260~450V (实际可工作范围达 245~480V)

### ▲恢复时间短: <10~40ms.

▲抗干扰能力强: 尖峰抑制: 常模输入 2KV

尖峰信号, 输出 <40Vp

▲独具负载短路自动安全保护功能。短路解除后自动恢复正常输出。

▲纯正弦波输出、精度高 (<1%)。

▲可靠性特别高, 长寿命, 可视为半永久性设备。

▲规格: 单相: 0.3~10KVA;

三相 1.5~100KVA。

☆权威人士指出: 计算机的故障 90% 来自电源, 您想消除这 90% 的故障, 请用铁塔 CWY 稳压净化电源。

☆在当代高科技领域里, 许多复杂的电子设备不时莫名奇妙地坏了, 这不仅仅与其本身质量有关, 而且与电源质量也很有关, 电源问题万万不可忽视。

☆铁塔 CWY 功能独特, 与高精尖娇而昂贵设备配套非常安全可靠, 使用与维护者极为放心, 用户效果效益特别显著。

☆敬请用户注意: 巨额的计算机与精密电器设备投资, 切勿忘记配套花钱少收益大的电脑保镖。

●铁塔牌 JH 系列—计算机、数控设备、精密测量仪器的理想电源。荣获 90 年省优秀新产品奖。

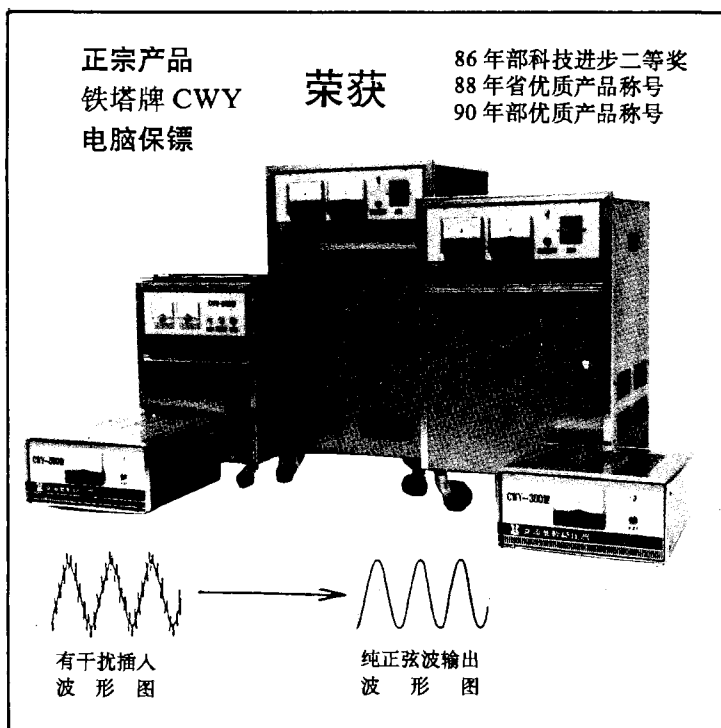
▲JH 系列交流稳压净化电源, 它是正弦波能量分配器与大功率滤波器两大先进技术之组合。

▲可靠性高, 性能远优于 614 类电子稳压器, 体积小、重量轻, 但价格与其相当。

▲精度高 (<1%)、效率高 (>93%)、响应速度快 (<10~40ms)、正弦波输出。

▲抗干扰力强 (尖峰抑制: 常模输入 2KV, 尖峰信号, 输出 <20VP)。

▲单相: 1、2、3、5、10 (KVA)。



★省级先进企业

★二级计量单位

## 广东省国营罗定无线电厂

★中国电源学会成员单位

★省定点电源设备专业生产厂

厂址: 广东省罗定县城逢源二巷 16 号

电话: 723559

电挂: 7193

邮码: 527200

长途区号: 0766

广州办事处: 江南西路青凤大街 30 幢

102 房王东岳

电话: 411450





# 南虹牌单色

NH9311型 31cm(12吋) 终端显示器  
NH9351型 35cm(14吋)

高清晰度

防眩目

平面方角

安全可靠

豪华外形

大量出口



白色



琥珀色



绿色



NH9312型 31cm 中华机显示器

## 国营广州无线电厂制造

地址：广州市员村一横路 邮 码：510656

通信处：广州市1411信箱销售处

电 话：516580 516215-2187 电 挂：0193

国内统一刊号：CN44-1188 邮发代号：46-1

# 广州白云山电源设备厂

## CWY系列高抗干扰稳压电源

我厂是生产稳压器、变压器、变压器铁芯的省电子局定点厂，具有十多年的生产历史，是我国生产各类电源设备及其配件的骨干企业。所生产的CWY系列高抗干扰交流参数稳压电源是我厂84年研制成功的国内首创稳压电源。通过国家技术部门鉴定。技术性能优越，比国内其他类型稳压电源有更明显的优越性。86年获国家科技进步奖。三相抗干扰稳压电源也由中国科学院广州分院通过技术鉴定。88年获产品专利权。89年获中国科学院技术进步三等奖。

本厂是生产CWY系列高抗干扰稳压电源的最早厂家，系列齐全。近年来经过工程技术人员研制攻关，对噪音、空载电流和漏磁干扰都取得了相当完善的解决，所以我厂产品与市场上同类型产品相比有更优越的性能，赢得了国内计算机用户、使用高精仪器设备等企、事业单位的信赖和广泛的应用，为我国电源系列发展作出卓越贡献。

购买时，请认明商标，提防有人剽盗本厂技术制造伪劣产品，使用户造成不必要的损失。

本产品已向中国人民保险公司办理了全国范围产品责任保险，用户可放心使用。

### 单相抗干扰稳压器系列

#### 一、型号及规格

型号	350	500	1K	2.2K	3.2K	5.2K	10K	15K
容量	350VA	500VA	1KVA	2.2KVA	3.2KVA	5.2KVA	10KVA	15KVA

#### 二、主要技术参数

●输入电压单相交流220V50Hz

●电压稳定度

输入电压范围	输出电压稳定度
175V~264V	$\Delta U_{\text{出}} < \pm 1\%$
160V~264V	$\Delta U_{\text{出}} < \pm 2\%$
140V~300V	$\Delta U_{\text{出}} < +2/-7\%$

●无过压危险，有自动短路保护特性

●总谐波失真度 $<4\%$ （开关式电源负载）

●对电网振铃干扰或尖脉冲干扰抑制能力符合国际计算机电源要求

●应变时间（输入电压跳变 $\pm 100$ 伏） $<10\sim 30\text{ms}$

●有效功率可达到80~90%

●音频噪声低，位于1.6米距离处50dB

全國範圍產品責任保險



高抗干擾  
安全可靠  
功能特殊  
電腦必備

### 三相抗干扰稳压器系列

一、规格：3KV、6KV、10KV、15KV、30KV

二、主要技术参数：●干扰脉冲抑制：输入脉冲 $<400$ 伏，输出不被发现；输入脉冲 $>1000$ 伏，输出 $<100$ 伏。

●稳压度调整率：输入 $-40\%\sim +40\%$ ，输出 $-4\%\sim +1\%$

●音频噪声低：位于1.6米距离处52db。

●效率高：89%。比一般电子交流稳压器效率高1/5。这对节能很有意义。

本单相、三相电源已与VAX11/750、VAX11/785、PDP11/44连机试验。运转正常，性能良好，广泛适用于计算机、自动控制设备，电子显微镜，X光CT，核磁共振断层扫描仪等高、精、尖设备等。与国外同类产品性能相等。

厂长：贝远娥 副厂长、工程师：陈自如 厂址：广州市沙河同和 邮政编码：510515

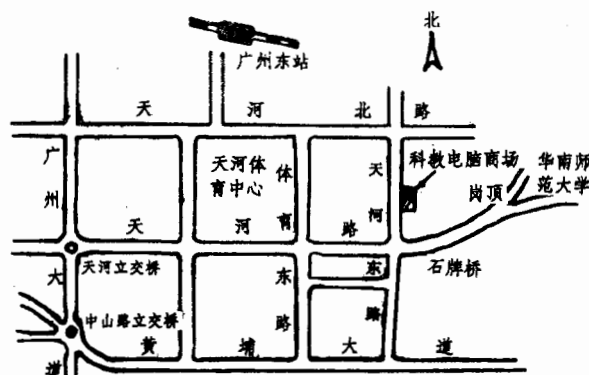
开户银行：广州农行白云营业所 账号：80-431032 电挂：0839 电话：705764转23 705665转343

注册商标 质量三包 欢迎来人来函订购 多谢惠顾

# 广州市科教电脑商场微机配置及价格一览表

机 型	超级 PC		超级 XT		说 明
CPU	8088V20	8088V20	8088V20	8088V20	可插 8087
频率	4.77 / 13MHz	4.77 / 13MHz	4.77 / 13MHz	4.77 / 13MHz	机箱按钮加速
内存	640KB	640KB	640KB	640KB	可扩至 1MB
软驱	360K × 2	360K × 2	360K × 2	360K × 2	可选 1.2M 或 1.44M
硬盘			20MB	20MB	可选 40MB
显示	高 / 中分单显(双频)	CGA640 × 200	高 / 中分单显(双频)	CGA640 × 200	单显 12", 彩显 14"
串 / 并	2 / 1 多功能	2 / 1 多功能	2 / 1 多功能	2 / 1 多功能	可选超级多功能卡
键盘	101 键	101 键	101 键	101 键	XT / AT 两用
价格	3300 元	5500 元	5000 元	7200 元	选 1.2M FDD 另加 200 元
机 型	CT286				
CPU	80286-16	80286-16	80286-16	80286-16	可插 80287
频率	10 / 22MHz	10 / 22MHz	10 / 22MHz	10 / 22MHz	按钮加速、液晶显示
内存	1MB	1MB	1MB	1MB	可扩至 4MB
软驱	1.2M+360K	1.2M+360K	1.2M+360K	1.2M+360K	可选 1.44M 3.5" 盘
硬盘	20MB	40MB	40MB	100MB	3.5" 高速硬盘
显示	CGA640 × 200	EGA640 × 350	TVGA1024 × 768	TVGA1024 × 768	可选 14" 高分单显
串 / 并	2 / 1	2 / 1	2 / 1	2 / 1	可选四用户卡
键盘	101 键	101 键	101 键	101 键	AT / 386 两用
价格	10000 元	12000 元	12700 元	14700 元	赠送机械式鼠标器
机 型	CT386DX / 33-64K Cache RAM				
CPU	80386DX-33	80386DX-33	80386DX-33	80386DX-33	可插 80387
频率	33 / 58MHz	33 / 58MHz	33 / 58MHz	33 / 58MHz	按钮加速、液晶显示
内存	2MB	2MB	4MB	4MB	可扩至 16MB
软驱	1.2M+1.44M	1.2M+1.44M	1.2M+1.44M	1.2M+1.44M	任选
硬盘	40MB	40MB	100MB	200MB	3.5" 高速硬盘
显示	EGA640 × 350	TVGA1024 × 768	TVGA1024 × 768	TVGA1024 × 768	任选
串 / 并	2 / 1	2 / 1	2 / 1	2 / 1	可选四用户卡
键盘	101 键	101 键	101 键	101 键	386 / AT
价格	21500 元	22000 元	25000 元	28000 元	赠送光电式鼠标器

地址: 广州市天河东路商业街东座 25-27 号  
 通信: 广州市 1753 号邮政信箱  
 邮编: 510620  
 电话: 511197、755147  
 电挂: 8360 FAX: 511197  
 开户银行: 广州中国银行天河支行  
 帐号: 271-015110027  
 免收运费 保用壹年



## 电脑应用

- 微型计算机遥控遥测无人值守广播电视发射机  
系统 ..... 郑德庆等 (2)  
视野测试 ..... 张建新 (4)  
微机数据处理工作的排序方法研究 ..... 杨宪泽 (5)

## 软件纵横

- MS-DOS 彻底剖析(十)IBM BIO 模块数据结构详析(上)..... 郭嵩山等 (11)

## 大学生之页

- 为可执行文件设置口令 ..... 谭毓安 (16)

## 使用与维修

- |                         |     |      |
|-------------------------|-----|------|
| AR-2463 打印机故障检修一例 ..... | 蔡 敏 | (18) |
| PC 机总线故障检测 .....        | 张文峰 | (19) |
| 硬盘不能启动的软维修方法 .....      | 徐云彪 | (20) |
| 软盘 Q 磁道损坏文件后的恢复 .....   | 黄 文 | (22) |
| 9P.EXE 中的错误及更正 .....    | 陈江海 | (22) |
| 对《电源维修一法》的几点看法 .....    | 常宝林 | (23) |

## 计算机辅助教学

- 计算机模拟化学中的酸碱滴定 ..... 许永飞 (23)

## 中华学习机

- 一种简便的进制转换方法 ..... 李 峰 (24)
- 解拆 CEC-I 五笔字型一例 ..... 傅 剑 (25)

中学天地

- APPLE 经验二则 ..... 张亚栋 (25)
- 对《电算 Waring 问题》程序的修正 ... 陈庆祥 (26)
- 一个屏幕快速绕卷程序 ..... 章文嵩 (27)

## 竞赛辅导

- 一九九〇年广东省青少年计算机程序设计竞赛  
试题分析及其选解(下) ..... 郎家炜 (28)

## 病毒防治

- 毛毛虫病毒的分析及排除方法 ..... 胡向东 (30)
- “428 病毒”的分析、诊断及防治 ..... 姚建华 (32)
- 一类感染.COM 文件病毒的消除方法 ... 方强 (34)

## 游戏乐园

- 游戏图形图象的巧妙再利用 ..... 张昌平 (36)

万花筒

- 音乐程序 ..... 水力 (38)

## 电脑用户

- ### 对 CCDOS4.0 (即西山 DOS) 的修改与补

- |   |     |      |
|---|-----|------|
| 充 .....   | 王大银 | (41) |
| 程序倒运行的实现方法 .....  | 华松青 | (42) |
| 汉化 Turbo Pascal 3.01A, Supercalc 3,<br>Wordstar 显示行参数修改 ..... | 宋运康 | (45) |
| TURBOPASCAL 程序中调用 TURBO C 模块<br>应注意的问题 .....                  | 陈文杰 | (47) |
| 电子系统交流电源的配置 .....   | 陆玉新 | (49) |

## 简讯

- 《LCDOS 四维形声码微机汉字操作系统》通过  
鉴定 ..... (3)

## 服务台

- 软件库 ..... (48)

## 广告索引

- |  |       |
|--|-------|
| 广州经济技术开发区夏港综合服务公司 ...                  | (封面)  |
| 电脑杂志社广州经济技术开发区科技开发经营<br>部开业 .....      | (封二)  |
| 广州白云山电源设备厂 CWY 系列高抗干扰稳<br>压电源 .....    | (I)   |
| PC1500 计算机新型 128K 电子记录模块开始<br>供应 ..... | (II)  |
| 广东省教育服务公司供应陕西计算机厂中华<br>学习机 .....       | (II)  |
| 广州市科教电脑商场微机配置及价格一览表 ...                | (III) |
| 当今国内最理想的高抗干扰稳压净化电<br>源产品 .....         | (封三)  |
| 南虹牌单色终端显示器 .....                       | (封底)  |

- 编 辑  
出 版
- 《电脑》编辑部  
电脑杂志社
- (地址:广州市石牌华南师范大学内  
邮政编码:510631 电话:516911-3273)
- 印 刷  
总发行处  
订 阅 处  
定 价  
出版日期
- 韶关二九〇研究所地图彩印厂  
韶关市邮电局  
全国各地邮电局、所  
1.00 元  
91 年 4 月 20 日



# 微型计算机遥控遥测无人值守 广播电视发射机系统

广州华南师范大学 郑德庆 郑云婷  
郑潮庆

**摘要:** 用国华II型计算机与 Z-80 单板机组成的 CTV-I 型微机遥控遥测无人值守广播电视发射机系统, 用于高山无人值守的电视广播差转台取得良好的效果。该系统结构紧凑, 操作方便, 工作稳定, 抗干扰能力强, 能更好地保证节目播出的准确。本系统对微机远距离无线控制广播电视发射机做了初步探讨, 有一定的推广价值。

## 一、前言

随着社会经济文化的发展, 人们对广播电视的依赖越来越强。自 1985 年我国开展卫星电视广播以来, 各县、区、乡地区, 为了扩大电视广播的复盖率, 提高收看效果, 纷纷建立 50~300W 的电视广播差转台, 这些差转台大多建立在本地附近的高山上, 采用差转广播。差转台建在高山, 交通不便, 环境恶劣, 值守工作人员生活极艰苦单调, 而且很不安全。因此, 研制高山差转台无人值守遥控遥测系统, 成为急待解决的问题。

## 二、系统概述

CTV-I 系统是由上位机 (用国华-II 型微机为核心构成的主控机) 与下位机 (用 Z80-CPU 单板机为核心构成的操作控制机) 构成的微型计算机遥控、遥测系统。它可对直线距离 10~15 公里的电视发射机 (差转机)、调频或调幅广播发射机实现多机 (台)、多功能及多个参数的控制与测量。

本系统主要功能有: 远程无线电控制 16 个开关量输出, 可分别控制 5~8 台发射机工作; 远程无线电测定 12 个模拟量输入, 可对发射机的 12 个参量进行实时遥测; 具有四路视频 (音频) 切换, 供广播或电视节目 (转播或自编) 之间的远程切换。

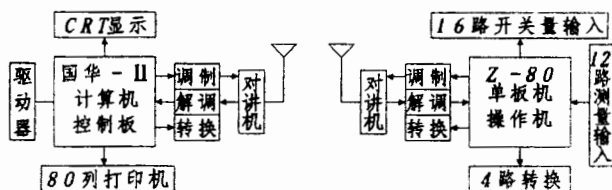
本系统的上位机与下位机之间的通讯与操作采用指令一应答测控方式, 即系统的操作系统全部在上位机, 它发出各种操作指令由无线信道到达下位机, 下位机按指令进行相应的操作与运行。

为使 CTV-I 系统操作简单易行, 用户的一切操作只需按上位机显示器显示的各级操作菜单指引按对应操作键, 即可命令下位机完成一系列的运行操作, 各种操作功能信号的发送与检查由上/下位机内的计算机自动执行, 不需工作人员干预。

为保障 CTV-I 系统不因操作失误而产生错误运行, 本系统采用标志互锁方式, 即使按错操作键, 也不会产生误运行。某些可以由计算机自行控制完成的操作, 则由系统本身自动完成, 从而保障操作的正确性, 大大减少操作失误。

## 三、硬件设计

CTV-I 型系统由微型计算机国华-II 型机和 Z-80 单板控制机及对讲机组成, 系统硬件方框图如图一:



图一

上位机由国华-II 型计算机和调制解调器组成。上位控制机配有显示器、打印机、软盘驱动器, 对发射机的各种遥控均通过“菜单”引导, 并把遥控所测数据显示在显示器屏幕上。Z-80 单板机 ROM 固化下位机的运行程序, RAM 供栈区与 I/O 接口暂存缓冲区用。16 个开关量输出能可靠地对 16 组 220V 继电器进行开关操作。12 路模拟量输入, 可对发射机的工作情况进行测量并经 A/D 转换成数字量, 送入 Z-80 CPU 进行运算, 转成相应参数发回上位机显示或打印。所有 I/O 接口均采用光电隔离。本系统还提供了 4 路视频或音频切换器, 供节目转换用。

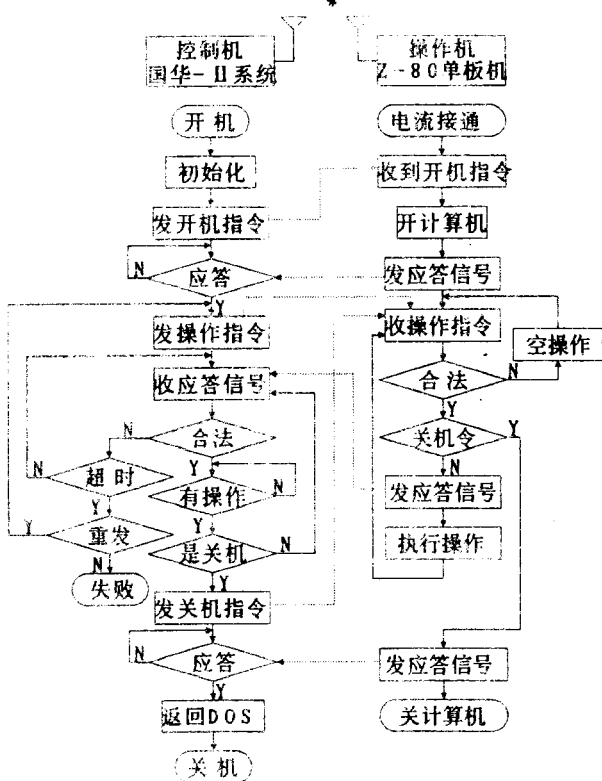
为实现上位机与下位机之间的无线通讯, 本系统采用通用的 5W 对讲机作为无线电通讯工具。接收与发送由上位机送出程控信号, 驱动一个小型继电器执行收发转换。对讲机振荡频率为 157MHz, 上位机与下位机的通讯采用指令一互答测控式。使用时, 上位机发出原先规定好的功能操作码, 经调制后送入对讲机发射。下位机天线接收此信号后, 进行解调, 恢复原数字操作码送入 Z-80 CPU 进行运算, 以完成各项操作。送入调制器的调制信号为十位异步串行数字信号, 通过二次调制后由对讲机天线发送。

解调电路是通过对讲机第一次解调后得到载有音频分量的数字代码, 送入有三级运算放大器和二级积分组成的整形与阈值检测电路进行解调, 恢复原来的功能代码供下位机操作。

## 四、系统软件

本系统软件由两部分组成, 其工作流程图如图二所示。





图二

### (一) CTV-I 系统软件组成

本系统软件用 BASIC 语言、6502 汇编语言及 Z-80 汇编语言写成, 有以下特点:

1. 操作用菜单引导进行。操作者利用在显示器上显示的菜单选择操作。每次操作结束屏幕上均显示操作结果和当时下位机工作状态。菜单层次清楚, 用户即使没有计算机知识, 也易学会操作。

2.为减少操作者因误操作而造成下位机控制失误,致使广播中断,系统软件内部各项操作均设立标志,且各项操作标志互锁。即使操作者出现误操作,下位机也不会响应。

3.系统软件中有主程序暂停功能,当广播节目正常投入运行时,CCTV程序有程序暂停功能,可将国华-II型计算机用于其他工作。此时上位机的对讲机每隔3分钟机收到下位机自动发来的脉冲音频信号,表示下位机操作机仍正常工作。

(二) 程序通讯协议与抗干扰措施:

1. 下位机始终处于接收状态，被动地按照上位机发出的指令进行操作，为了增强排除无用信号对下位机造成的误动作，上位机每个操作指令都发同样的 4 个操作功能代码。当下位机接到两个相同的功能代码，即可判断为正确信号，可转入按代码要求进行的相应操作，并向上位机发回一个规定代码，当上位机收到此代码后，就可认为本操作已被下位机所接收并已进行操作，同时将运行结果反映

在屏幕上，

2.按照发射机工作程序的要求,其某些操作均由程序自动控制完成,无须操作者介入,从而提高了控制自动化程度,也减少操作错误与难度。

3. 本系统采用自动复位与等待延迟转出程序。在系统运行时受到各种无线电磁波干扰时一方面能自动延迟一段时间后自动转出, 使下位机从程序混乱中转入正常运行; 另一方面在下位机程序中插入一条 I/O 指令, 当系统受干扰时, 可使这条指令失效, 导致单稳态电路工作, 从而使 CPU 复位, 驱使主程序重新运行, 使整个程序恢复正常工作, 以此提高了系统的抗干扰能力。

## 五、结束语

CTV-I型微机遥控遥测无人值守广播电视发射机系统已在广东省广宁县广播事业局的协助下投入运行,使用情况良好,操作方便,运行可靠,不失为各地广播电视台的优良配套设备。特别是本设备价格较低,易为用户所接受。在我国广大山区或边远地区,这种性能价格比高的产品,具有较大的推广价值。

本文经陈天钧教授审阅, 值此表示感谢。并向参加本系统研制工作的张依元、苑蓓蓓致意。

需要安装本系统的山区电视差转台,请与电脑杂志社联系。

## 《LCDOS 四维形声码微机 汉字操作系统》通过鉴定

【本刊讯】2月4日《LCDOS 四维形声码微机汉字操作系统》简称陆码，通过由广东省电力工业局组织的鉴定。《陆码》是广东省电力一局陆国华同志开发的，国家专利局已受理该项专利申请。

《陆码》综合了汉字的时空四维信息，它包含了汉字的字形、笔顺、字音、声调、使用频度等因素，一共只有四十个基本码元，每个汉字由四码唯一确定，每个词组也由四代表，又可以用形、音、形声、词组等四种方法兼容输入汉字。

用同一种编码可以方便地统一处理简体、繁体和日文汉字, 已有繁体版本, 10 行和 25 行版本, 可适用于 CGA、EGA 以及 VGA 的 IBM 兼容机, 286、386 等各种机型。

增强了区位翻页检字、字典学习、100种常用标点符号、声响提示、快速显示、提示编码等功能,人机界面友好。

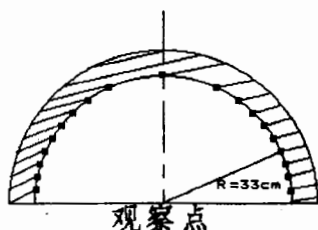
陆码汉字系统设计的主要对象是不用专门训练的普通操作者，如一般工作人员、工程技术人员、科学家、记者、教师、大中学生等，由于编码简单无重码、见字识码效率高、高频简码兼词组，对连续文本平均每个汉字仅须击 1.9 键，短期自学后很容易达到每分钟输入 60 个汉字，熟练者可以超过 200 字。

# 视野测试

山东省淄博市建设银行计算机科 张建新

从生理学的角度来说,正像视力、色盲度一样,每个人还应有一个视野度的指标。特别是作为飞行员、驾驶员,在高速行驶中要能够眼观六路随机应变,没有宽阔的视野是不行的。本人根据有关医学资料,利用 LASER310 微机设计了这套测试视野的装置。

测试方式如图一所示。利用一头架使测试者的双眼置于半球形测试屏的球心位置。在球带上分布有 27 个发光二极管。中心管为红色。测试者双眼凝视中心管。测试顺序为先右眼后左眼。运行程序后,计算机发出声响。中心向右第一管亮。若测试者可以看到则在键盘上按下 A 键,否则按 L 键。若按错则判为无效。测试屏自中心向两侧各为 90 度,分布有 13 个发光管。发光管顺序及角度如表一所示。其中 16 度为盲点,即在本装置中任何人都不能看到此点,理应按 L 键。从第 1 管至第 13 管依序发光,并伴有声响提示,测试者依观察按键。若某管发光 30 秒而不按键则判为无效。第 13 管发光后计算机统计有效按键并计算成绩,然后进行左眼测试。



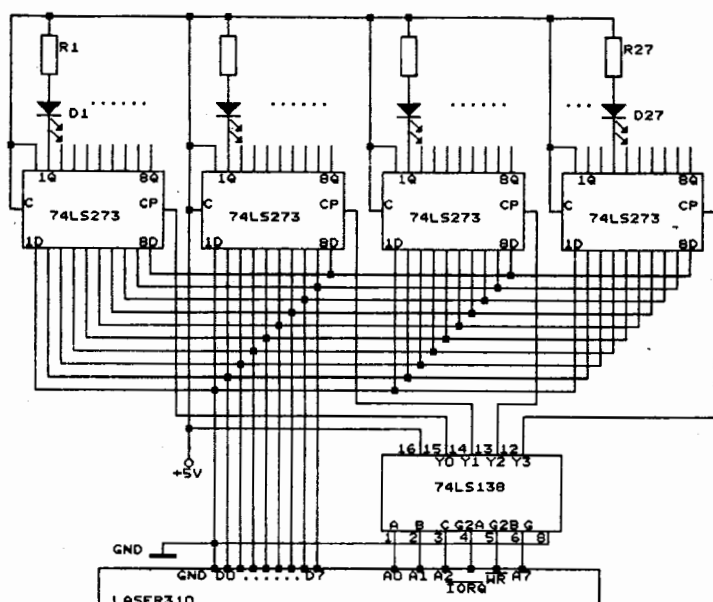
图一

硬件接口电路如图二所示。由 74LS273 锁存显示数据并驱动发光二极管。为满足功率要求,74LS273 工作在灌电流状态。由 74LS138 译码并产生选片脉冲 CP。4 片 74LS273 的地址分配为 80H-83H。

以下为一程序范例:

```
5 DIM I (5)
10 FOR N = 1 TO 2: FOR I% = 1 TO 2000: OUT
129, 223: NEXT
20 FOR I% = 1 TO 13: READ A$: C = VAL
(LEFT$ (A$, 3))
30 C1 = VAL (RIGHT$ (A$, 3))
35 OUT 128, 255: OUT 130, 255: OUT 131, 255
```

```
40 OUT 129, 223: OUT C, C1
50 P% = 0: SOUND 20, 4
60 A$ = INKEY$
70 P% = P% + 1: IF P% = 280 THEN 120
80 IF A$ = " " THEN 60
100 IF (I% = 2) AND (A$ = "A") THEN I (I%) = 10:
GOTO 130
110 IF (I% < > 2) AND (A$ = "L") THEN I (I%) = 10:
GOTO 130
120 I (I%) = 0
130 NEXT
140 IF I (3) = 10 THEN I (3) = 4
150 IF I (2) = 10 THEN I (2) = 6
155 OUT 128, 255: OUT 129, 255: OUT 130, 255
160 FOR I% = 7 TO 12
170 IF I (I%) = 10 THEN I (I%) = 5: NEXT
180 I = 0: FOR I% = 1 TO 13: I = I + I (I%): NEXT
190 PRINT "SHI YE CS (" ; N ; ") = " ; I
200 IF N = 2 THEN 220
210 SOUND 28, 8: SOUND 28, 8
220 NEXT N: END
230 DATA 129191, 129127, 130254, 130253, 130251, 130247
240 DATA 130239, 130233, 130191, 130127, 131254, 131253,
131251
```



图二

# 微机数据处理工作的排序方法研究\*

西南民族学院 杨宪泽

## 摘要

本文是笔者近几年从事内排序方法研究工作的阶段小结,力图扼要、准确地向读者介绍国内外内排序方法的研究现状及发展前景,供从事这方面研究或应用中选择排序方法的读者参考。文中我们认为典型、实用、高效的排序方法,给出了结构图、BASIC程序或算法流程,涉及的理论分析、数学证明请参考给出的相关文献。笔者希望,此文能促进内排序研究学术交流,指导应用。由于本人水平和有关资料限制,错误之处,恳请指正。

关键词:排序,比较交换法,分布型、映射、多分量记录

## 一、引言

排序是计算机科学中很重要的一个子领域,它的发展不仅可以引起大批量数据处理变革,而且将影响软件工程。在众多的计算机自身系统软件和应用软件中,提高速度是共同面临的问题,使用好的排序算法及利用其基本思想又是解决问题的关键。近十几年来,虽然微电子技术飞速发展使微机速度增长很快,但在信息爆炸的时代,人们需要处理的数据量越来越大。排序算法时间复杂性有三种: $O(N)$ ,  $O(N \log_2 N)$ ,  $O(N^2)$ 。如果一小时内  $O(N)$  的排序算法能处理的数据量为  $3.6 \times 10^6$ ,  $O(N \log_2 N)$  的排序算法能处理的数据量为  $2 \times 10^5$ ,  $O(N^2)$  的排序算法能处理的数据量仅为 1897;若微机速度提高一万倍,那么  $O(N)$  的排序算法能处理的数据量变为  $10000 \times 3.6 \times 10^6$ ,

$O(N \log_2 N)$  的排序算法能处理的数据量却为  $9000 \times 2 \times 10^5$ ,  $O(N^2)$  的排序算法仅能处理数据量  $100 \times 1897$ 。这说明,排序算法效率的高低对能处理的数据量多少有决定性作用。当数据量急剧增大时,如果没有  $O(N)$  这样的高效率的排序算法,单纯依靠提高微机的速度是行不通的。

从另一个角度看,微机用于各行各业的事务处理工作中,数据处理,情报资料整理,企业管理等都必须检索。排序可以提高检索的效率,因此,众多领域中的大量数据元素的无序序列必须调整成有序序列,致使现今的微机系统中花费在排序上的时间占系统 CPU 运行时间很大比重。有人认为<sup>[1]</sup>在商用计算机中,其批处理系统的 15% 至 70% 的 CPU 时间用在排序上;也有人认为<sup>[2]</sup>排序工作占了计算机工作量 50% 左右。我们不敢肯定上述两种提法的正确性,但据我们的研究、考察资料表明,排序工作占了微机应用工作的 20% 至 30%。因此,为了提高微机工作效率,研究出更有效的  $O(N)$  的排序方法,是计算机应用科学的一个重要方面。

自从计算机问世以来,由于上述重要性,研究各种高效排序方法成为软件工作者必不可少的重要课题。迄今为止,有代表性的内排序算法,已提出几十种,且有不同的特点和效率。但可以认为,没有一个在任何情况下都十分完美的排序方法。对于它们的分类,没有统一的标准,有的认为可按特点归结为五类<sup>[2]</sup>,也有人把它们分成三类<sup>[3]</sup>。笔者认为把它们分成两大类更合适:一类是比较交换排序方法,一类是分布型排序方法。

比较交换排序算法是人们经常使用的一类排序方法。它有算法容易设计,不附加存贮空间的特点。典型的选择排序方法是一种比较交换排序算法,基本思想是:对于  $N$  个元素,首先通过  $N-1$  次比较找出最小元素,并将这一个元素与第一个元素交换(如果正好第一个元素是最小元素,不执行交换操作)。第  $i$  次在剩余的  $N-i+1$  个元素中找出最小元素与第  $i$  个元素交换,直到  $i=N-1$  为止,  $N$  个元素排成非递减序列。这一算法的时间复杂性满足递归式:

$$T(N) = \begin{cases} 0, & \text{对 } N=1 \\ T(N-1) + (N-1), & \text{对 } N>1 \end{cases}$$

250 DATA 129239, 129247, 129251, 129253, 129254, 128127, 128191

260 DATA 128, 223, 128239, 128247, 128251, 128253, 128254

DATA 语句中每个数据前三位为选片地址后三位为所亮管的数据。N 循环共 2 次,左、右眼各一次。I% 循环为一眼的测试。20-30 句把 DATA 语句中的数据分解转换。40 句用 OUT 命令使发光管点亮。60-80 句检测是否按键。100-120 句判断按键是否正确并定成绩。140-170 句将一眼测试结果转换成实际角度值。190 句输出最终测试成绩。

如果测试者不通过按键输入信号而希望使用按钮或开关,则可再设计一输入接口,相应地修改 60 语句为 INP 命令。

表一

序号	1	2	3	4	5	6	7	8	9	10	11	12	13
偏角(度)	10	16	20	30	40	50	55	60	65	70	75	80	90

上式的解,  $T(N) = \frac{N}{2}(N-1)$ , 是  $O(N^2)$ 。

分布型排序算法考虑了信息记录关键字值的分布情况, 不象比较交换排序法反复比较关键字而交换位置, 而是构造关键字值与一个数组(内存空间)的对应关系, 附加一定的存储空间换取时间。如基本的映射排序方法, 采用关键字求值与一个数组(附加存储空间)对应, 由于附加存储空间地址是有序的, 关键字的值计算结束只要按附加存储空间地址顺序调相应关键字所隶属的信息记录即可完成排序工作。这类排序算法, 虽然还不成熟, 但在一定条件下, 附加一定的存储开销, 期望时间复杂性是  $O(N)$ 。

本文力图扼要、准确地介绍国内外研究这两类内排序算法的现状、发展前景, 供从事这方面研究和应用中选择排序方法的读者参考。

## 二、比较交换排序法及其前景

比较交换排序法如上所述, 是实施对信息关键字反复比较和交换两种操作的排序方法。其优点带来了编程容易, 象上述选择排序法可用程序一实现:

程序中, A 是辅助变量, 10-50 句: 定义一个能容纳 N 个数据的数组, 输入 N 个待排数据, 60-110 句: 排序为递增序列, 若  $D(I) > D(J)$ , 实施交换, 反之不交换, 120-140 句: 输出已排序的数据。

```
10 INPUT N
20 DIM D(N)
30 FOR I=1 TO N
40 INPUT D(I)
50 NEXT I
60 FOR I=1 TO N-1
70 FOR J=I+1 TO N
80 IF D(I) > D(J) THEN 100
90 A=D(I): D(I)=D(J): D(J)=A
100 NEXT J
110 NEXT I
120 FOR I=1 TO N
130 PRINT D(I)
140 NEXT I
150 END
```

这一排序方法, 由于时间复杂性为  $O(N^2)$ , 速度较慢, 但是其算法容易理解, 编程简单, 程序易读等优点, 使不少人愿意在一般场合使用, 今后也仍然将在一般场合使用。

为了提高比较交换方法的排序速度, 不少人作了努力, 产生了许多典型的排序算法, 如 1959 年 shell 提出的 "a high speed sorting procedure", 又称 shell 排序; 1962 年 Hoare 提出的 "Quicksort"; 1964 年 Williams 提出的 "Heapsort"。Quicksort 可以认为是比较交换排序法研究的一个里程碑, 因为一直在比较交换法中具有最佳的平均性能。它的基本思想是: 在需要排序的 N 个元素中, 随机抽取一个元素 a, 使全体元素分成含有小于 a 的元素子集和含有大于 a 的元素子集, 递归地使用分治法, 最终可以完成排序。已经证明, Quicksort 时间复杂性在介于  $O(N \log_2 N) - O(N^2)$  之间。

Heapsort 的方法是把待排序的 N 个数据看成是一棵顺序存储的二分树, 将它改造成一个堆, 排序时把堆顶元素和堆底元素进行交换, 并让新的堆底(当前最大关键字数据)元素脱离堆, 即让堆的元素减少一个。这时新的堆在根节点破坏了堆的性质, 因此要重建堆, 对于建好的新堆又重复上述过程, 直到堆中仅含两个元素时, 排序工作即已完成。堆排序的中心就是建堆和重建堆, 方法较好, 由于它相当巧妙地利用了树的结构, 所以在比较交换次数上, 时间复杂性为  $O(N \log_2 N)$ , 在存储空间占用上, 属于最小存储类。Quicksort 与 Heapsort 相比, 以平均时间性能而言, Quicksort 最佳, 其所需时间最省, 因为 Heapsort N 小时, 优点并不突出, 理论上人们认为 Quicksort 在最坏情况下的时间性能不如 Heapsort, 但应用中往往难以遇到最坏情况, 所以人们一般认为 Quicksort 方法优于 Heapsort 方法(还包括优于 Lester Ford 和 Selmer Johnson 提出的归并排序方法等, 它们的时间复杂性也是  $O(N \log_2 N)$ )。

以后的二十多年, 比较交换排序方法没有多大突破, 仅产生了一些在特定环境附加特定条件的高效方法, 不具有典型性。这是因为已经证明, 在一个具有 N 个元素的序列的所有排列以相等的概率作为输入的条件下, 对 N 个元素的序列的所有排列以相等的概率作为输入的条件下, 对 N 个元素排序的任何一棵判定树的期望深度决不小于  $\lceil \log_2 N! \rceil$ 。

据斯特林公式:  $\lceil \log_2 N! \rceil = O(N \log_2 N)$

这一结果说明, 比较交换排序法时间复杂性下界是  $O(N \log_2 N)$ , 因此新产生的比较交换法要想在平均性能上优于 Quicksort 很不容易, 这就使得采用比较交换法的许多复杂的理论研究和较大的应用工作一直面临提高速度难题, 即使是深入研究和改进比较交换算法, 也只能事倍功半, 算法设计成功率很小, 不可能较大地提高速度。例如, 文献[4]的作者, 在改进堆排序方面作过努力, 结果仅减小了时间复杂性常数因子, 效率提高一倍。

1986 年, J. Hopcroft 和 R. E. Tarjan 获得了世界范围内计算机科学领域的最高奖 Turing 大奖, 奖励他们在算法与数据结构设计和分析的基础成就。R. E. Tarjan 等人一项重要成就就是提出了一种摊算复杂性分析 (Amortized Computational Complexity), 在这种新理论支持下, 1985 年 J. R. Sack 和 T. Strothotte 在隐式堆上建立了新的合并算法<sup>[5]</sup>, 1986 年 G. H. Gonnet 对隐式堆上的算法进行改进, 这些研究虽然还不完全成熟, 但它预示着, 如果新的复杂性度量标准建立了新的优秀的数据结构, 比较交换排序法的时间复杂性下界  $O(N \log_2 N)$  有可能被突破。

## 三、分布型排序法及其前景

分布型排序法中人们最熟悉的大概是基数法。基数法的优点是算法速度快, 为  $O(N)$ , 基本思想是: 设排序的元素由 K 位数字组成的十进制数, 各位数字看成一个分量, 编成 0, 1, 2, ..., 8, 9 十个桶, 有:

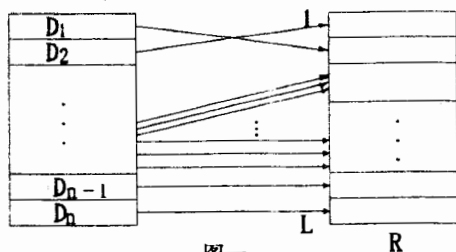


1. 将要排序的  $N$  个整数置入一个单向队列中。
2. 依次将队列中的每个数, 按倒数第  $i$  位数字的值  $X$  置入编号为  $X$  的桶中。
3. 按桶的编号顺序 (从小到大), 把每个桶中的数字 (按先进先出顺序) 依次重新置入单向队列中。
4. 重复 2, 3 的步骤, 直到  $i=K$  为止。这样, 就得到一个非递减的数字序列。

按基数排序的基本思想, 算法要使用较多的空间。因为基的个数一般为 10, 需要 10 个桶的队列, 每一个桶的队列最大长度可能得需要  $O(N)$  单元。这是由于各元素的某个分量为同一基数时, 经某遍扫描后, 所有元素将被置入一个桶中, 如果采用固定存贮分配, 将占用内存空间  $11 \cdot N$ , 不宜实际应用。链接结构的基数排序法改进了上述基数法, 附加存贮空间小于  $3N$ , 但据分析<sup>[6]</sup>和进行测试, 效率不如 Quicksort。主要原因是仅考虑了信息关键字某位与基的对应关系, 附加操作很占时间。然而, 基数法的意义在于, 它揭示了可以构造  $O(N)$  时间的排序算法途径, 为今后的研究奠定了基础。

几乎与基数法同时产生, 1956 年 Isaac, E. J 和 Singleton, R. C 提出了用地址计算排序<sup>[7]</sup>, 这是分布型排序算法的初期研究, 但并未得到足够重视: 其一, 当时的计算机内存很小, 以内存开闭交换速度不可取; 其二, 初期的研究有很大的局限性, 没有跳出反复比较和交换关键字方式的束缚。八十年代后半期, 我国一些人不约而同研究地址排序 (映射排序), 并投入应用。其原因是: ①随着我国微机在事务处理应用中的普及, 需要处理的信息量越来越大, 寻求  $O(N)$  的排序算法成为当务之急。②微机在事务处理中的某些应用具有特殊性。如高考分数排序, 竞赛名次排序, 全面质量评估得分排序等关键字最大值和最小值在一定范围内, 使用分布型算法可获得高效率, 这就促进了这类算法的研究和应用, 而其它一些工作又借鉴这些工作的成功基点, 使这类算法得到扩充, 发展; ③微电子技术飞速发展, 微机内存容量大大增加, 从而使这类算法以牺牲部分存贮开闭交换速度不会带来问题。

基本的映射排序思想是: 给定一组数据  $D_1, D_2, \dots, D_n$ , 可知或可以估计这组数据中的最大值  $D_{\max}$ , 最小值  $D_{\min}$ 。那么, 若开辟一个数组  $R(L)$  (如果  $D_{\min}$  为正,  $L=D_{\max}$ ; 如果  $D_{\min}$  为负,  $L=D_{\max}-D_{\min}$ ), 即可把数据送入数据值与  $R$  数组下标相等的对应元素中, 该方法原理结构见图一



图一

显然, 若  $D_1=50$ , 它对应  $R(50)$ ;  $D_1=500$ , 对应  $R(500)$ 。相同数据落在同一数组元素中, 用计数方式可知有几个。由于数组元素的下标是有序的,  $500 > 50$ , 数组元素的下标自然把数据一次定好了位置, 最后只要将非零元素按规定的打印出来, 相同元素按计数值打印多次, 排序即可完成。

映射排序彻底抛弃了将数据一一反复比较交换的方法, 时间复杂性是  $O(N)$ , 速度很快。映射排序提高速度附加了一定的存贮开销。

如由大至小排序 1000 个数据 ( $D_{\max}=1000, D_{\min}=1$ ) 的 Quicksort 需要时间 189 秒, 基数排序需要 98 秒, 映射排序仅需 26 秒。

```
10 N=1000: L=1000
20 DIM R(L)
30 FOR I=1 TO N
40 INPUT D
50 R(D)=R(D)+1
60 NEXT I
70 FOR I=L TO 1 STEP -1
80 FOR J=1 TO R(I)
90 PRINT I;
100 NEXT J
110 NEXT I
120 END
```

程序中, 30-60 句: 输入数据, 计算相同数据出现的次数。若  $R(35)=0$ , 没有 35 这个数据。70-110 句, 由大到小输出数据, 没有的数据由于  $R(I)=0$  不输出; 相同数据  $R(I) > 1$ , 连续输出。

认真研究基本的映射排序方法, 可以发现两个问题:

1. 数据中应包括正数、负数、小数和相同数据, 基本的映射排序没有提出小数的处理方法。
2. 如果  $L > N$ , 附加存贮开销很大, 排序效率也将大下降。

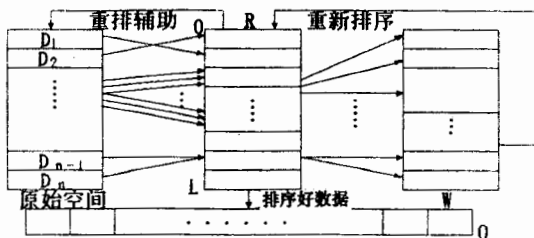
针对这两点, 文献[8]提出的改进措施如下:

(1) 在能估计或已知一组数据  $N$  的最大值  $D_{\max}$  和最小值  $D_{\min}$  以及出现的小数最多位  $T$  的情况下, 限制数组  $R(L)$  中  $L < 1000$ 。

(2) 若  $L = |(D_{\max} - D_{\min}) \times 10^{T+1} + 1|$  计算后  $L > 1000$ , 求  $L$  的位数  $q$ , 使  $L = |(D_{\max} - D_{\min}) \times 10^{T+3-q} + 1|$ 。

(3) 使数据  $D_i (i=1, 2, \dots, N)$  对应于  $R(L)$  中  $L = \text{INT}[(D_i - D_{\min}) \times 10^{T+3-q} + 1]$ 。

改进后的排序结构见图二



图二

图二中, R 数组限制为 1000, Q 数组存放排序好的数据, W 数组存放需要重新排序的数据。这一排序法由于限制  $R < 1000$ , 致使关键字值大的不同的数据 L 值相同, 落入同一级送入 W 空间。对于落入 W 空间的同一级数据, 有再排序过程。再排序时用原始空间辅助存贮, 直至同一级数据中  $D_{\max} = D_{\min}$  为止。

笔者称上述排序法为分级排序法, 已经证明, 一般情况下分级排序时间复杂度是  $O(N)$ 。分级排序首次大胆的将复杂性高的运算引入排序问题, 自始至终利用求值, 映射操作, 这是传统观念难以接受的。但事实上, 引入复杂性高的运算, 不一定必定增加问题求解的复杂性。高斯上小学时对  $(1+2+3+\dots+100)$  的计算  $(101 \times 50)$  即是一个有名的例子。更何况现代计算机内存极大, 还可带有硬件乘法器。

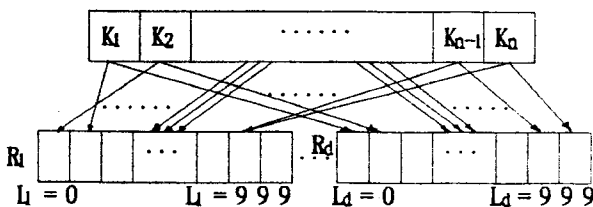
分级排序经严密的理论考察和应用检验, 暴露了如下缺点:

- 1、空间 R 独立于数据 N, 故对较小或大 N 均不适宜。
- 2、算法的平均复杂性为  $O(N)$ ; 最坏情况不是  $O(N)$ , 因为数学证明排序时间  $T < CN$ , 但 N 个数据经分级, 每一级含有 m 个数据。R < 1000 直接影响 m, 而 m 影响 C。由于 R 确定, 数据分布不均或值太大都可能导致不同数据落入同一级别使 m 发生变化, 引起 C 发生变化。这就是说, C 有一变化幅度, 而且很难事先确定, 致使一些特殊应用速度下降。

文献[9,10]提出分级排序的改进算法—子域映射排序法, 即把要排序的信息记录关键字描述成一个二维数组

$$K = \begin{bmatrix} K_{11} & K_{12} & \dots & K_{1d} \\ K_{21} & K_{22} & \dots & K_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ K_{n1} & K_{n2} & \dots & K_{nd} \end{bmatrix}$$

对于某一关键字  $K_{ij}$ , 描述成  $K_{i1}K_{i2}\dots K_{id}$ 。规定  $K_{ij}$  由三位数组成, 不够填充 0。排序在每一子域中进行, 即  $(K_{ij})$ ,  $1 \leq i \leq n$ ,  $1 \leq j \leq d$ , 结构见图三。



图三

整个排序过程见流程图。

流程中, P 空间为辅助记数空间, 记录映射于每一子域的相同数据。W 空间存贮已排序的关键字, T 为辅助变量。

$K_i$  切分按下式进行:

$$S_d = \text{INT} (K_i \times 10^{-3})$$

$$S_{d-1} = \text{INT} (S_d \times 10^{-3})$$

$$S_1 = \text{INT} (S_2 \times 10^{-3})$$

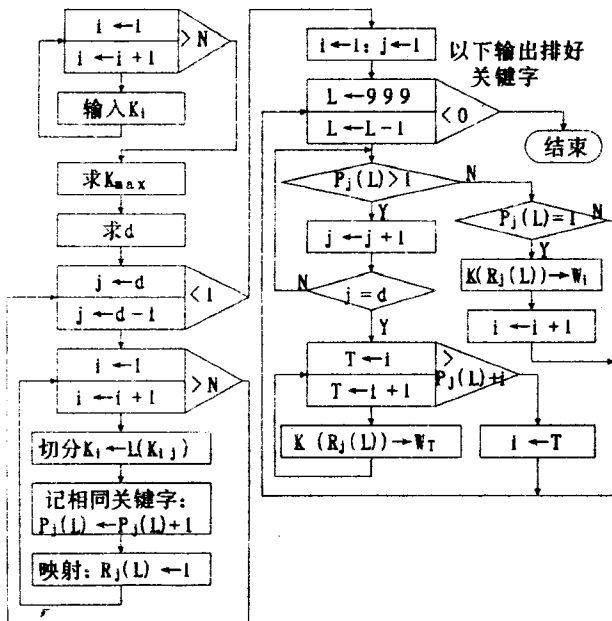
有:

$$K_{id} = (K_i \times 10^{-3} - S_d) \times 10^3 = K_i - S_d \times 10^3$$

$$K_{id-1} = (S_d \times 10^{-3} - S_{d-1}) \times 10^3 = S_d - S_{d-1} \times 10^3$$

: : :

$$K_{i1} = (S_2 \times 10^{-3} - S_1) \times 10^3 = S_2 - S_1 \times 10^3$$



流程图

子域映射排序考虑微机对数表达范围限制, 规定  $d < 13$ , 即  $K_{\max} < 10^{39}$ 。已经证明, 该算法时间复杂度是  $O(N)$ , 不要求数据均匀分布。并且, 只要知道 N, d, 从具体使用微机系统得知比较指令时间, 排序时间可以预估。

子域映射排序的缺点是: 空间 R 独立于数据 N, 如果 N 很小, 附加存贮空间  $2000d + N$  可能远大于 N, 加之使用乘法次数太多, 因此这一排序不适宜 N 小的情况。这再次说明, 没有一个在任何情况下都十分完美的排序方法。

在研究映射排序算法中, 八十年代还出现一种有代表性的改良算法值得一提[11,12]: 这种算法考虑了关键字 N 与附加在贮空间 R 关系, 依据基本映射排序方法思想, 又融入比较交换法在处理少量 N 速度快的特点, 算法步骤如下:

- (1) 计算 N 个关键字  $K_i$  ( $i = 1, 2, \dots, N$ ) 的极大值和极小值

$$K_{\max} = \max (K_1, K_2, \dots, K_N) = X(N),$$

$$K_{\min} = \min (K_1, K_2, \dots, K_N) = X(1),$$

若极差  $L = K_{\max} - K_{\min} = 0$ , 排序关键字取值相同, 无需进行排序; 否则,  $L > 0$ , 转 (2)

- (2)  $K_i$  分成  $R_j$  组

$$R_i = INT_1 (K_i - K_{min}) \frac{N}{L} + 1$$

$P(R_i) = P(R_i) + 1$ ; 记录落入各组的关键字个数。

(3) 把  $K_i$  送入  $W$  空间,  $P > 1$  的关键字采用链接方式, 记录链尾地址。

(4) 对于分成的  $N+1$  组,  $R_1$  中关键字最小,  $R_{N+1}$  中关键字最大, 各组采用比较交换法排好后依次送原始空间即完成排序。

这种算法附加存贮空间依赖于  $N$ , 如果关键字值均匀分布, 结果比基本映射排序方法效果好, 时间复杂性为  $O(N)$ ; 如果关键字值分布不均匀, 落入同一组关键字太多, 此方法失效。

通过上述讨论, 笔者认为分布型排序算法的研究和应用前景是乐观的。可以预测, 这类算法除了自身成为一类独立、成熟、高效、有用的排序算法外, 还可以引入一些以前认为已经成熟的排序算法中, 使它们的速度从  $O(N \log_2 N) - O(N^2)$  提高到  $O(N) - O(N \log_2 N)$ 。

#### 四、多分量记录的排序方法

当今的信息处理, 记录的分量越来越多(高考分数成绩排序, 记录分量达 10 个; 全面质量管理评估记录分量达 30 多个), 对它们进行排序, 移动分量的时间多于关键字比较所花的时间。而目前人们对不改变信息记录物理位置, 又符合常规事务处理需要的排序算法研究还未引起足够重视, 提供这方面的典型排序算法时间复杂性仍是  $O(N^2)$ 。

此外, 笔者注意到,  $dBASE III$  具有很强的数据处理能力, 在事务管理工作中发挥着越来越大的作用。其索引文件命令使用后, 索引文件中只包含排序的数据库文件的关键字内容和记录号, 不改变原数据库文件中的记录的物理位置, 符合日常事务处理要求。但索引文件命令的速度却慢到难以容忍的程度;  $dBASE III$  排序命令的速度较快, 方便了查询, 但对同一个数据库文件, 因不同需要按不同关键字排序, 就会变成几个数据库文件。这样修改原数据库文件时, 对排序文件也必须修改, 或者重新排序, 否则将造成数据的不一致性, 这一弱点限制了排序命令的使用。

笔者在此给出一个已经达到时间复杂性下界  $O(N)$  的不移动记录物理位置的排序程序, 其方法亦属分布型映射排序法。附加存贮开销为  $3M+N$  ( $M$  为关键字最大值)。提请注意, 由于  $M$  在一般常见问题中不大, 如高考总分  $M=700$ , 而大规模信息处理中  $N$  很大, 所以可以认为附加存贮开销小于  $2N$ 。因此, 程序的方法适宜广泛用于多分量记录的大规模信息处理工作。

对于这一方法, 理论分析和实验证明是高效率的: 用 BASIC 与  $dBASE III$  进行直接数据交换, 然后编程与  $dBASE III$  索引文件命令对比, 使用 90 西藏高考成绩 650 人数据, 结果是索引文件命令需要 9 分 25 秒时间, 此排序方法仅需 35 秒时间; 与  $dBASE III$  排序命令相比 (即使不考虑  $dBASE III$  排序命令的弱点), 数据量  $N=10000$

时, 结果是  $dBASE III$  排序命令需 6 分 25 秒时间, 此排序方法需 3 分 40 秒时间。

```
10 INPUT N, M, I
20 DIM D1 (N), D2 (N), D3 (N), P (M)
25 DIM Q (M), R (N), W (M)
30 FOR J=1 TO N
40 INPUT D1 (J), D2 (J), D3 (J)
50 P (D1 (J)) = P (D1 (J)) + 1
60 IF P (D1 (J)) > 1 THEN 80
70 W (D1 (J)) = J: Q (D1 (J)) = J: GOTO 90
80 R (W (D1 (J))) = J: W (D1 (J)) = J
90 NEXT J
100 X=1: K=1
110 FOR J=M TO 1 STEP -1
120 IF P (J) = 0 THEN 190
130 T=Q (J)
140 PRINT D1 (T), D2 (T), D3 (T), X
150 IF K=P (J) THEN 180
160 K=K+1
170 T=R (T): GOTO 140
180 X=X+1: K=1
190 NEXT J
200 END
```

程序注释:

10 句: 给定数据量  $N$ , 关键字最大值  $M$ , 最小值  $I$ 。  
20—25 句: 开辟链指针空间  $R$ , 记数空间  $P$ , 链指针空间  $Q$ , 链当前指针空间  $W$ ; 其中, 我们仅假设了一个记录含  $D1, D2, D3$  三个分量,  $D1$  为关键字。

30—90 句: (1) 从  $J=1$  开始, 输入记录各分量, 让  $P(D1(j)) \leftarrow P(D1(j)) + 1$ , 完成映射工作, 记录相同关键字出现个数。

(2) 若  $P(D1(j)) = 1$ , 作  $W(D1(j)) \leftarrow j$  和  $Q(D1(j)) \leftarrow j$ , 转 (4); 即据关键字映射所对应不同的单元, 构造链当前指针和链首指针, 链当前指针将为 (3) 中出现相同关键字提供链接地址, 链首指针是映射单元对应信息记录首地址。

(3) 若  $P(D1(j)) > 1$ , 作  $R(W(D1(j))) \leftarrow j$  和  $W(D1(j)) \leftarrow j$ ; 即有相同关键字, 据当前链指针地址链接起来, 此外, 还要再记录当前链指针, 为链接出现多个相同关键字作准备。

(4)  $j \leftarrow j+1$ , 直至  $J=N$  为止, 实施 (1) — (3)。

100—200 句: 从已被计数的最大空间地址开始, 据链指针关系输出信息记录, 给出对应名次 ( $X=X+1$ ), 结束。

这一方法说明了两个问题: 其一, 现代事务处理中出现的新问题, 必须有相适应的新排序方法。其二, 信息记录关键字 ≠ 信息记录。在这一问题上, 近年常见刊物上登载一定条件下适用而高效的排序算法把关键字集合与信息记录集合等同起来, 用关键字代替了信息记录。这种错误的出现, 关键字在多次移动后 (信息记录没有移动) 不再与信息记录有一一对应关系。因此, 这样描述的算法简化了排序任务 (关键字 = 信息记录), 适用范围比排序算法应涉及的范围小得多。事实上, 排序的定义是: 给定一个

有线性  $R$  集合的  $N$  个信息记录  $a_1, a_2, \dots, a_n$ , 对其给定的关键字  $a_{11}, a_{12}, \dots, a_{1n}$  的排序 (有可能是多关键字排序, 本文不作讨论), 使  $a_1, a_2, \dots, a_n$  成为一个特定有序排列。显然, 关键字  $a_{1i}$  ( $i=1, 2, \dots, n$ ) 只是  $a_i$  的一个分量,  $a_i$  的分量可以很多。如果把  $a_i$  和  $a_{1i}$  混淆, 排序即使不出问题, 也只是一种特殊情况。若以此描述成一般算法, 显然简化了任务。

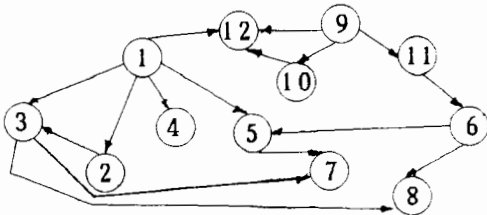
五、其它排序方法

在内排序方法中, 还有多关键字排序问题, 但这在日常事务处理中少见, 所以我们不予讨论。然而, 事务处理中还常有以元素前后左右关系来确定元素位置的, 这不同于依靠元素数值大小确定的排序方法。这种方法称拓扑排序。

例如: 有一张教学计划表 (表一), 学习其中某一门课程必须在学完相应的先修课才能进行。例如要学数据结构, 必须先学程序设计、离散数学、高级语言。因此, 先修条件定义了课程之间的领先关系, 这个关系可以用有向图 (图四) 表示, 其中, 结点表示课程, 有向边表示先决条件。

表一

课程代号	课程名称	先修课代号
1	程序设计基础	
2	离散数学	1
3	数据结构	1,2,5
4	汇编语言	1
5	高级语言	1,6
6	机器语言	11
7	编译原理	5,3
8	操作系统	3,6
9	高等代数	
10	线性代数	9
11	普通物理	9
12	数值分析	9,10,1



图四

排序的办法是构造一个节点的线性序列, 使得在此序列中不仅保持有向图中原有的节点之间的先后关系, 而且对没有向图中没有关系的两节点之间 (如图中的 9 和 1) 也建立一个先后关系 (或 1 先于 9, 或 9 先于 1)。具体算法, 参阅文献[13]。

上述教学计划经拓扑排序成为:

1, 2, 4, 9, 10, 11, 12, 6, 5, 3, 7, 8

实际工作中, 还有许多需要拓扑排序的例子, 如工程的施工, 产品的生产流程, 一个具体程序的流程等等。首先, 要满足每个环节的优先顺序, 无先决条件的, 则可并列进行。

此外, 排序方法研究还涉及外排序和并行排序。内排序和外排序的区别在于: 进行内排序时, 记录放于内存, 涉及的只是记录或其分量的存取交换问题。由于存取各内存单元速度基本相同, 提高效率的关键在于如何进行操作, 如何减少操作; 进行外排序时, 需要将记录从外存调到内存加工后再送外存存贮, 然后再进行下一步加工。这样就造成了内外存多次交换信息, 而这种交换信息所需时间通常比内存加工所用时间长得多, 因此效率高低一般取决于内外存交换的频繁情况, 取决于外存设备的速度, 靠算法极大地提高速度受限, 故本文不作讨论。

并行排序是目前一般微机所不能进行的, 它需要并行处理机, 所以本文也不讨论。但可以预言, 随着微电子技术的发展, 也许在不远的将来, 并行处理机会象今天微机一样普及, 那么并行排序算法的研究会突飞猛进, 在事务处理工作中发挥不可低估的作用, 其大型信息记录的处理速度将令人生畏。

\* 国家民委科研基金资助项目

参考文献

- [1]严蔚敏等, 数据结构, 清华大学出版社, 1988 年。
- [2]夏新国, 一种崭新的高速排序法, 软件报, 1990 年 19 期。
- [3]Nillaus Wirth, Algorithms = Data Structures = Programs, Prentice-Hall, 1976。
- [4]顾训豫等, 堆排序的改进算法及其复杂性分析, 计算机学报, 第 13 卷, 1990 年 4 期。
- [5]J. R. Sack, et al., An algorithms for merging heaps, Acta information 22 (1985), 177-186。
- [6]王本顺等, 数据结构技术, 清华大学出版社, 1988 年。
- [7]Isaac, E. J., et al., Sorting by Address Calculation, J. ACM, 3 (1956), 169-174。
- [8]杨宪泽, 分块快速排序法研究, 科学通报, 第 34 卷, 1989 年 11 期。
- [9]杨宪泽, 子域映射快速排序法研究, 科学通报, 第 35 卷, 1990 年 15 期。
- [10]杨宪泽, 基于映射排序方法的子域映射算法, 计算机应用, 第 11 卷, 1991 年 1 期。
- [11]Ehrlich, G., Searching and Sorting Real Number, J. of Algorithms, 2 (1981), 1-12。
- [12]张建中, 快速分组排序, 数值计算与计算机应用, 1988 年 3 期。
- [13]魏晴宇等, 数据结构, 中国人民大学出版社, 1988 年。■



# MS-DOS 彻底剖析 (十)

## IBMBIO 模块数据结构详析 (上)

郭嵩山 朱国庆 陈政

数据结构的设计是操作系统设计的关键之一，MS-DOS 的 IBMBIO 模块（磁盘 BIO）是表驱动程序，其运作是环绕在数据结构上所进行的各种操作。因此，我们在详细剖析 IBMBIO 模块时，先了解其数据结构，将有利于对整个模块各个部分的了解。IBMBIO 模块所用到的数据结构（包括 IBMBIO 初始化程序为建立 OS 运行环境而建立的数据结构）有堆栈、缓冲区、线性表和线性链表。

### 一、堆栈

在 IBMBIO 各个子模块运行过程中，需要建立各自的系统栈。此外，初始化程序（SYSINIT）II 在其运行过程中要根据系统配置而建立起堆栈运行环境。下面，分作介绍。

#### 1. 系统栈

系统栈是以传统形式的静态堆栈，其名称与内容如表 1 所列。这些供 IBMBIO 初始化时使用的系统栈，其所占的内存空间，在初始化程序任务完成后将随初始化程序本身所占资源被释放而消失。

表 1 IBMBIO 系统栈

栈名	所在模块	栈底	长度(Byte)
搬家子程序系统栈	SYSINIT I	0: 7BE2H	4
SYSINIT I 系统栈	SYSINIT I	70: 0000H	视系统配置而异
SYSINIT II 系统栈	SYSINIT II	CS: 096AH	使用 SYSINIT II 已执行过代码所占区域

由于 IBMBIO 模块的 SYSINIT II 程序在初始化过程中要将自身重定位到内存 RAM 的高端，以腾出空间来定位 IBMDOS.COM 模块，SYSINIT II 重定位后在高段的段址，其值为：

CS = RAM 最高端址 - 01FCH

#### 2. 堆栈运行环境

IBMBIO 建立的另一种重要的堆栈结构，由 SYSINIT II 在解释 CONFIG.SYS 中 STACKS 命令时作为 OS 运行环境之一建立在内存 DOS 内核和 COMMAND 常驻部分之间的区域，它将为用户提供中断发生时所需要用到的堆栈空间。

堆栈运行环境由堆栈框架头，中断修改部分和堆栈池三部分组成。其中堆栈框架头有 12H 个字节，包含有关堆栈池的信息；中断修改部分存放由 SYSINIT 修改的 14 个中断向量，该部分共有 7DEH 个字节；堆栈池由堆栈框架部分及框架说明块部分所组成。堆栈运行环境的结构如图 1 所示，堆栈框架说明块结构如图 2 所示，框架说明块长度为 8 个 Byte。



\* 偏移值指在堆栈池内的偏移

图 1 堆栈运行环境的结构

相应堆栈框架头偏移值	+07H
保存上次所用堆栈指针	+06H
留 用	+02H
标 志	+01H
	+00H

图2 堆栈框架说明块的结构

在图1中,堆栈框架的个数以及每个框架所占用的字节数是由CONFIG.SYS中配置命令STACKS=n,s所设置的。其中n为堆栈框架数,其从8~64,为每层堆栈框架的字节数,其值从32~512。如果不使用STACKS配置命令,n,s值根据机器默认值选取:

(1)对IBM PC/XT、PCjr等机型,n=0,s=0,也即不设置STACKS运行环境,不提供任何动态堆栈。在这种情况下,DOS不截取任何中断,而只使用DOS的内部栈。

(2)对其他机型,如IBM PC/AT及其向上兼容的微机,n,s隐含值分别为9及128Byte。

在每次发生硬件中断时,DOS就从堆栈池中取出一个尚未使用的堆栈框架供系统使用,待中断处理完毕,再将

这个堆栈框架释放到堆栈池中。

IBMBIO采用这种动态堆栈结构的意义在于,当一系列中断发生时,能使系统每次中断独自使用一个堆栈区,从而避免了由于一系列中断发生而导致可能产生的中断竞争。由于采用动态堆栈管理,提高了用户界面的透明度。

## 二、缓冲区

缓冲区主要用于暂存从各种设备输入等待处理的数据和准备输出到各种设备的数据。

IBMBIO模块所用到的缓冲区很多,其中包括传统形式的缓冲区,环形队列结构的缓冲区以及BUFFERS运行环境。

### 1.传统形式的缓冲区

表2列出了传统形式缓冲区名称、长度、用途和装入内存的位置(3.3版)。

### 2.环形队列缓冲区

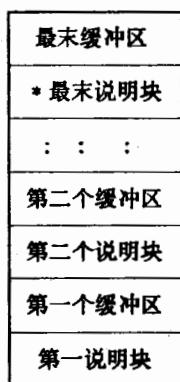
BIO的键盘输入缓冲区采用环形队列结构,用来存放键盘输入内容,每组(1个WORD)包括键符ASCII码和键盘扩展码,其长度视机型而异。对PC/XT及AT机,只用到前面15 WORD。该缓冲区安排在内存0040:001EH开始的20H Byte,其当前字符指针存放在0040:001AH WORD,可用单元指针存放在0040:001CH WORD。

### 3.BUFFERS运行环境

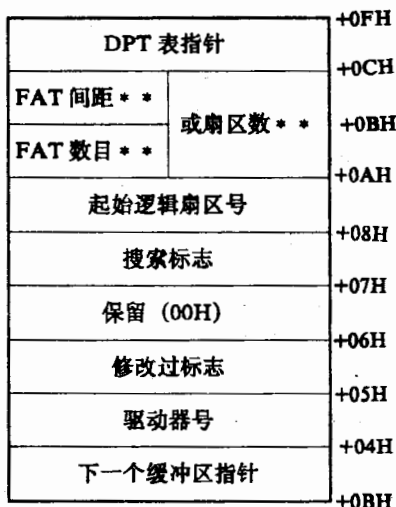
表2 IBMBIO模块中的缓冲区

名 称	地 址	长度 Byte	功 能
控制台设备 I/O 缓冲区	0070: 05D0H	1H	存放从键盘输入字符的扩展码
辅助设备 I/O 缓冲区	0070: 00BC~00BF	4H	每种设备用 1 个 Byte,依次存放 COM1、COM2、COM3、COM4 所输入的数据
块设备 I/O 缓冲区	0070: 0281~0480H	200H	存放从块设备读入 1 个扇区的内容
引导记录缓冲区	07C0: 0200~03FFH	200H	存放从引导区读入 1 个扇区的内容
卷宗名缓冲区	0070: 1DA4~1DAFH	0CH	存放从介质读入的逻辑卷宗名,该字符串以 00H 为结束标志
SYSINIT I FAT 缓冲区	RAM 最高段址—每个 FAT 占内存数	512*扇区数/FAT	供 SYSINIT I 程序用作存放所读出 FAT 表
搬家程序用 FAT 缓冲区	0000: 8000~83FFH	400H	供 SYSINIT I 搬家程序用作存放所读出 FAT 表
CONFIG 文件用磁盘路径缓冲区	CS: 1C2C~1C37H	0CH	存系统配置文件磁盘路径
国家格式信息文件磁盘路径缓冲区	CS: 1C38~1C46H	0FH	存系统配置文件磁盘路径
最高命令处理文件磁盘路径缓冲区	CS: 1C85~1C97H	13H	存最高命令处理文件磁盘路径
中断向量缓冲区	0070: 186BH~18A2H	38H	供 SYSINIT II 改变中断向量时保留原中断向量用,每 4 个 Byte 存 1 个中断向量,依次存 02H, 08H, 09H~0EH, 70H, 72~74H, 76H, 77H, 共 14 个中断向量
SHELL 配置命令参数缓冲区	CS: 081E~083DH	20H	存放系统配置文件 SHELL 命令参数值,即新的最高级别命令处理文件磁盘路径字符串

BUFFERS 运行环境实际上是由 SYSINIT II 根据系统配置命令在 RAM 的低端建立的具有链结构的缓冲区, 用于存放读盘数据, 每个缓冲区可存放一个扇区长数据。采用 BUFFERS 运行环境这种缓冲区链结构, 可以减少短期内重复读同一扇区的操作。这种运行环境, 为每个缓冲区配置一个说明块, 以保存指向下一个缓冲区的指针, 每个说明块长度为 10H Byte。通过各说明块, 将 BUFFERS 运行环境中各个缓冲区连接成一个链表。BUFFERS 运行环境结构如图 3 所示。



a) BUFFERS 结构



b) 说明块结构

图 3 BUFFERS 运行环境的结构

说明:

\* 指向 EMB 缓冲区运行环境链链头

\*\* +0A~+0BH 在不同操作时存放不同值, 在读写 FAT 时分别存放 FAT 间距或 FAT 数, 在读写一般扇区时存放数据区内所包含的扇区数

在图 3 中, 第一个缓冲区说明块保存指向 DOS 内核

中一个缓冲区指针, 最末一个缓冲区说明块保存了整个缓冲区链的链头首址, 该值保存在环境管理块 (EMB) 中。

在 MS-DOS 中, 缓冲区设置与读盘效率有密切关系, 缓冲区越多, 读盘次数愈少, 但相应的空间开销就会增大, 所以缓冲区个数的设置要适当, 其值是由 SYSINIT II 解释 CONFIG.SYS 中的 BUFFERS 配置命令的参数值来决定。如果用户未使用该配置命令, 则系统将选取其默认值, 其取值是:

128KB < 内存容量 < 256KB: BUFFERS = 5;

256KB < 内存容量 < 512KB: BUFFERS = 10;

512KB < 内存容量: BUFFERS = 15。

### 三、表格概述

在 IBMBIO 初始化程序执行过程中, 为 OS 运行而建立了多种表格型的数据结构, 它们是线性表和线性链表, 所谓线性表 (linear list), 顾名思义它是一组有序的数据结构, 表中的每个元素, 除第一个外, 都仅有一个直接前趋 (predecessor), 同时除最后一个外, 都有直接的后继 (successor), 前面提到的栈和缓冲区, 实际上都是一种特殊的线性表, IBMBIO 所用到的线性表, 是指其个体及字段均按次序在内存中顺序排列, 它占有内存的一段连续的空间。而线性链表是一种不要求连续存储空间的线性表, 其每一个体, 都是一个普通的线性表, 它们依靠指针相连, 也即每一个体, 都包含有指向后继个体的指针, 采用线性链表的好处在于各个体在内存中不必按顺序排列, 且其在内存中的位置可以在运行过程中有所改变。这样, 当插入或删除某一个体时, 只需在操作后修改前一个体指向该个体的指针。

IBMBIO 模块所用到 (或为系统而设置) 的表格较多, 表 3 列出其名称及结构。

下面, 分作详细介绍。

### 四、为 DOS 运行而建立的表格

IBMBIO 为 DOS 内核运行而建立的表格有内存控制块、环境管理块、FILES 运行环境和 FCBS 运行环境。

1. 内存控制块 (MCB: Memory Control Block) 内存控制块用于管理内存分配。当调用 DOS 内核 4BH 号功能 (EXEC 功能), 由父程序加载子程序或在外部程序或用户程序调用 DOS 48H 号调用 (分配内存) 时, DOS 为其申请内存空间, 建立一内存分配块, 并在该块头部设置长度为 1 节的块头, 以存放该内存分配块的信息, 该内存分配块的块头就称为内存控制块。其结构如下:

+00H	+01H	+03H	+05H	+0FH
标 记	分配块段号	分配块节数	保留	

其中各字段意义如下:

\* "标记" = 4DH, 表示本内存控制块非最后一块;

= 5AH, 表示本内存控制块为最后一块。

\* "分配块段号" = 0000H: 本内存分配空闲;

≠ 0000H: 本内存分配块已分配, 其值为程序 PSP 段址。

“分配块字节数”=本内存分配块大小(节)。

对于单用户的 MS-DOS 3.X, 内存分配是连续的, 各内存分配块通过标记连成内存分配块链, 链尾是遇到 MCB 标记为 5A 的内存分配块。IBMBIO 初始化程序为内存建立的第一个内存控制块及分配块, 其控制块标记为 5A (即只有一个分配块), 其分配块段号为 0000H (即空间可用), 其分配块字节数为整个可用内存空间。

2. 环境管理块 (EMB: Environment Manging Block)

DOS 通过建立环境管理块, 将 DOS 内核运行所用到的各个运行环境的地址及有关参数 (如驱动器号、FCB 数目及每个缓冲区字节数) 集中存放 (在 EMB 中), 以便于内核进行管理。该管理块由 IBMBIO 的 SYSINITII 建立在 DOS 内核中, 其结构如图 4 所示

### 3. FILES 运行环境

FILES 运行环境是由 SYSINITII 程序根据系统配置建立的文件句柄参数表的一种堆结构, 用于对文件操作和管理。所谓堆结构是指在系统个 FILES 运行环境中含有多文件句柄参数表, 这些表一个接一个地顺序排列, 其链首指针存于环境管理块 EMB 偏移+04H 处。图 5 示出了 FILES 运行环境的结构。

在图 5 中, FILES 运行环境是由环境头 (00~05H 共 6 个 Byte) 和各个句柄参数表组成, 其中句柄参数表个数由 CONFIG.SYS 中 FILES 配置命令参数值 (若无设定, 默认值为 8) 减去 5 个来决定。所扣除的 5 个是指标准设备文件句柄参数表, 因为这 5 个文件句柄参数表在 DOS 内核中构成另一个 FILES 运行环境, 由 SYSINITII 在初始化时将这两个 FILES 运行环境连成链结构。这种链结构

表 3 IBMBIO 所建立的表格

类 别	表 格 名 称	结 构	装入位置或指针	长度 (Byte)
为 DOS 与 BIO 通讯而建立	I/O 请求标题 (RH)	线性表	DOS 内核 ES: BX	14H
	设备标题 (DH)	链 表	0070: 016E~247H	每设备 12H
为 DOS 运行而建立	内存控制块 (MCB)	线性表	RAM 高端	10H
	环境管理块 (EMB)	线性表	指针存 CS: 7F4~7F7H	26H
	FILES 运行环境	链 表	链首指针存 EMB 偏移 04H 处	视系统设置而定
	FCBS 运行环境	链 表	链首指针存 EMB 偏移 1AH 处	同上
为块设备管理而设置	块设备控制块 (BDCB)	链 表	链首指针 0070: 024CH	51H
	磁盘参数表 (DPT)	链 表	链首指针存 EMB 偏移 00H 处	20H
	驱动器参数块 (DPB)	线性表	4 个 DPB 地址不同	13H
	BIOS 参数块 (BPB)	线性表	各类盘规格 BPB 地址不同	13H
	BPB 指针组	线性表	0070: 18A4H	视系统配置而定
	磁盘 I/O 基数表	线性表	(从 ROM 读出)0: 0522H (从引导记录读出)0: 7C2BH	11H
	磁盘路径表 (DPAT)	线性表	由 EMS 偏移 16H 指示	51H
	地址场	线性表	0070: 11D8H	FEH
为设备驱动程序而设置	命令代码入口地址转换表	线性表	0070: 0003~00AFH	视各设备而异
	列表设备循环检测次数表	线性表	0070: 05D1~05D9H	09H
为 SYSINIT I 运行而设置的	IBMBIO.COM 文件参数表	线性表	0000: 7BE2H	1BH
	软盘空间分配格式表	线性表	0070: 2A22H	20H
	支持大容量软盘机操作记录表	线性表	0070: 2A9AH	1EH
为系统配置文件而设置的	设备参数区 (DPA)	线性表	CS: 1CF9H	124H
	关键字表 (KT)	线性表	CS: 1CA6H	53H
	DRIVPARM 配置参数搜索表	线性表	CS: 1E71H	8H
其 他	错误代码翻译表	线性表	0070: 0271H 0070: 0279H	各 8H
	月份天数转换表	线性表	0070: 08B3H	0CH
	每月天数累加表	线性表	0070: 1B3CH	18H
	提示信息表	线性表	IBMBIO 模块内	



的最后一个 FILES 运行环境的下一个 FILES 运行环境地址置成 FFFFH 作为链尾标志。

#### 4.FCBS 运行环境

FCBS 运行环境也是由 SYSINIT II 建立的一种链表, 其结构与 FILES 运行环境完全相似 (如图 6 所示)。只是 04~05H 字段换成一次能打开的 FCB 数目, 其值由 CONFIG.SYS 的 FCBS 配置命令参数值 (如未设定, 默认值为 16) 来决定。FILES 运行环境链首指针存于环境管理块 EMB 偏移+1AH 处。在处理 FCBS 配置命令时, 还应用打开的 FCB 中 DOS 不能自动关闭的文件数来修改 EMB 相应字段的值 (偏移+1EH), 如该值未设定, 默认值为 4。

设备驱动程序链头地址	+25H
最末驱动器号	+22H
逻辑驱动器号	+21H
DOS 不能自动关闭的 FCB 数目	+20H
FCBS 运行环境链头地址	+1EH
磁盘路径表首址	+1AH
BUFFERS 运行环境链头地址	+16H
每个缓冲区字节数	+12H
现行输入设备程序地址	+10H
现行输入时钟设备程序地址	+0CH
FILES 运行环境链头地址	+08H
磁盘参数表 (DPT) 链头地址	+04H
	+00H

图 4 环境管理块 (EMB) 结构

最末文件句柄参数表	
:::	
第 2 个文件句柄参数表	+03BH
第 1 个文件句柄参数表	+06H
本环境拥有文件句柄参数表个数	+04H
下一个 FILES 运行环境地址	+00H

图 5 FILES 运行环境结构

最末 FCB	
:::	
第 2 个 FCB	
第 1 个 FCB	
本运行环境拥有一次能打开 FCB 数目	+06H
下一个 FCB 运行环境地址	+04H
	+00H

图 6 FCBS 运行环境结构

#### 参考资料

1. 郭嵩山, 陈学军: MS-DOS 3.3 BIO 初始化操作系统运行环境的建立  
计算机工程与应用 1990 年第 6 期。■

## 电脑世界用户佳音

CEC-I 中华学习机游戏杆(九针插头) 48 元/个

APPLE II 苹果机游戏杆(十六针双列插头) 48 元/个

IBM 游戏杆 (十五针插头) 58 元/个

以上游戏杆二模拟量二开关量, 使用进口日本万向电位器, 计算机按键, 进口插头, 现货供应, 款到即发。已含邮资。各类最新版 CEC-I, APPLE, IBM 电脑软件二千余盘, 最新版中文系统, 强劲工具软件, 管理, 文字编辑, 表格, 网络系统, 各类英语学习, 测试, 大小规模各类翻译软件, 英语试题库软件目录单免费函索, 欢迎“电脑”读者交流软件。来信写明机型, 详细地址, 联系人, 邮政编码。单位购买有正式税务发票报销。声明开票单位。

广州市解放北路桂花岗东 1 号

广州师范学院外语系电教室

联系人: 王德安

电话: 668410 或 663804 转 247

BB 机: 广州 181 电话台呼 68 7028

# 为可执行文件设置口令

北京理工大学计算机系 谭毓安

由于种种原因, PC 系列机的软件保护是很薄弱的, 和一些大型计算机系统相比有很大的差距。许多系统允许用户设置口令, 保护自己的软件。但 PC 机的系统软件却没有提供这一功能, 用户的软件可以被轻易地分析和仿制。有必要开发一种软件, 为 PC 机上的可执行文件设置口令, 防止非法用户使用和分析。为此, 我开发了设置口令字软件 PASSWORD, 可以给 DOS 环境下的 COM 文件和 EXE 文件设置口令。经过一段时间的使用, 收到了很好的效果。

## 一、设置口令字的原理

给可执行文件设置口令字的原理是: 由设置口令字软件加密文件, 把附加程序链接在文件的尾部, 再进行适当的处理, 使附加程序在执行文件时首先获得控制。附加程序中包含着检验用户输入口令是否正确的程序和对文件的内存映象进行解密的过程。

执行文件时, 首先执行附加程序。在接受用户输入的口令之后, 判断口令是否正确, 若口令不正确, 则停止执行, 否则对文件的内存映象解密, 再执行文件的正常程序。

可执行文件分为 COM 文件和 EXE 文件两类, 它们的结构和加载过程都有很大的区别。因此, 链接在它们尾部的附加程序是不同的, 为它们设置口令字时所进行的处理也不完全相同。下面分别介绍为它们设置口令的方法。

## 二、为 COM 文件设置口令

### 1. 使附加程序获得控制权

设置口令字软件把附加程序链接于 COM 文件的尾部。它把文件的头 3 个字节保存在附加程序中, 将头 3 个字节替换为一条 JMP 指令(一条段内 JMP 指令占 3 个字节), 这条 JMP 指令跳到附加程序的第一条指令。附加程序执行完后, 将内存中文件的头 3 个字节恢复为原来的内容, 再执行文件的正常程序。

在执行 COM 文件时, 位于文件头位置上的指令为第一条可执行指令。因此, 执行设置口令后的文件时, 由 JMP 指令跳到附加程序中, 使附加程序获得控制权。

### 2. 口令字正确性的判定

设置口令字软件在接收用户为文件设置的口令后, 把口令的某一函数计算出来, 将结果存放在附加程序中, 作为判定口令字正确性的依据。

在执行文件时, 附加程序在用户输入口令后, 计算口令的同一函数, 将计算出的结果与存放值相比较。若用户输入的口令与为文件设置的口令相同, 这两次计算出来的结果应该是相同的。因此, 若二者不等, 则说明口令字错误, 请求用户再次输入口令, 若三次之后口令仍不正确,

可认为是非法使用者, 停止程序的执行, 返回操作系统。

由于附加程序中存放的是口令的某一函数值, 因此, 破译者在分析时只能获得口令的这一函数值, 而不能获得口令本身。这种隐藏口令的办法对防止口令被破译是有效的。如果把口令直接放在附加程序中, 破译者就很容易获得口令。

### 3. 对文件的加密以及对内存中文件映象的解密

仅仅靠隐藏口令是不足以防止破译的, 因为破译者可以利用调试工具修改附加程序, 跳过附加程序的口令字判别部分, 从而使口令判别失效。

如果设置口令时对文件进行加密, 运行文件时在口令字正确的情况下再由附加程序对内存映象进行解密, 而加密密钥和解密密钥都是口令字的某一个函数, 也就是说和口令字有关, 破译者由于不知道口令字, 不能得到解密密钥, 因此无法继续执行文件。

设置口令字软件把 COM 文件全部加密, 加密密钥采用口令的一个函数, 根据口令计算出加密密钥, 对 COM 文件进行加密, 再把附加程序链接到它的尾部。

执行 COM 文件时, 首先执行的附加程序接受用户输入的口令, 在口令正确时根据口令计算出解密密钥, 用解密密钥对文件的内存映象解密, 解密后的 COM 文件映象和执行原文件时的内存映象是相同的。解密后, 将控制交给 COM 文件的正常程序。

## 三、为 EXE 文件设置口令

### 1. 使附加程序获得控制权

EXE 文件分为文件头和装入模块两个部分。文件头中包含着控制信息重定位项表。文件头中偏移为 14H 和 16H 的两个字分别是程序第一条指令在装入模块中的段内偏移和相对段值, 即这两个字确定了程序的第一条指令。在加载 EXE 文件时, 将起始段值加上相对段值作为 CS, 段内偏移作为 IP, 开始执行文件的正常程序。

设置口令字软件把文件头中偏移为 14H 和 16H 的两个字保存在附加程序中, 将它们替换为附加程序第一条指令在文件装入模块中的段内偏移和相对段值。这样, 在加载这个文件时, CS:IP 将指向附加程序的第一条指令, 执行附加程序。附加程序获得控制权。

### 2. 口令字正确性的判定

判定方法和 COM 文件附加程序的判定方法相同。

### 3. 对文件的加密以及对内存中文件映象的解密

EXE 文件的文件头不能加密, 只能加密它的装入模块。在执行附加程序时, 再对装入模块的内存映象解密。

若文件有重定位项, 加载时 DOS 将把起始段值加到由重定位项所确定的那些重定位字上, 这个过程叫重定

位。若对含有重定位项的装入模块进行加密后, 执行文件时 DOS 将把起始段值加到那些重定位字中。附加程序对内存中装入模块解密后这些重定位字的内容就是不正确的, 因此对有重定位项的文件要进行特殊的处理。

文件头中偏移为 06 的一个字指出了文件装入模块中重定位字的个数, 把这个字的值修改为 0, 这样在加载时 DOS 就不会进行重定位了。

设置口令字软件加密 EXE 文件的装入模块, 密钥选取为口令的某一函数。再把附加程序链接在文件的尾部。

执行文件时, 附加程序在用户输入口令正确的情况下根据口令计算出解密密钥, 用解密算法对文件装入模块的内存映象解密。

若原 EXE 文件有重定位项, 附加程序还要对装入模块进行重定位。将重定位项表中的每个表项的第一个字作为偏移, 第二个字加上起始段值后作为段值, 指向装入模块中的一个需要重定位的字, 把起始段值加到这个字上去。

这样, EXE 文件装入模块的内存映象和执行原文件时是相同的, 此时将控制交给原文件中的正常程序, 附加程序执行完毕。

#### 4. 文件长度的变化

EXE 文件头中偏移为 02 和 04 的两个字指出了文件长度。设置口令字后文件的长度肯定会增加, 增加的部分就是附加程序的长度。设置口令字软件要修改这两个字的值, 使它们反映出文件长度的变化。

#### 四、安全性分析

由于存放在附加程序中的仅仅是口令的某个函数结果, 由这个运算结果推测出口令是不可能的。这样, 在允许附加程序判断口令字正确性的同时, 避免了口令字被破解。计算过程越复杂, 被破解的可能性越小。计算时取口令中每个字符的 ASCII 值进行计算。

设置口令字软件加密文件的加密算法可任意选取, 但加密算法要有一定的复杂性。在设置口令字软件中, 采用了逻辑加密算法, 利用异或运算的特性加密。密钥的长度为 24 个字节, 而且每个密钥字节都和口令字有关, 在不知道口令的情况下, 靠穷举密钥的办法破译的可能性同样是很小的。

#### 五、使用说明

运行 PASSWORD 后, 首先输入要设置口令字的文件的文件名, 再输入赋予这个文件的口令, 为可靠起见, 口令要输入两遍, 前后两次输入的口令必须一致。口令可设置为任一以回车符结束的字符/数字串。

为某个文件设置口令字后, 运行此文件时, 将要求用户输入口令字, 输入的口命令字中的每个字符在屏幕上显示为“#”。若口令字正确, 则显示“Correct Password.”, 该文件可继续运行。若口令字不正确, 将显示“Incorrect password.”, 并要求用户重新输入口令。若输入三次之后口令仍不正确, 则显示“Illegal User?”, 返回 DOS。

例: 为宏汇编程序 MASM.EXE 设置口令字, 口令字设置为 8709046。运行口令字设置软件:

```
C>PASSWORD
Filename : MASM.EXE
PASSWORD : 8709046
PASSWORD : 8709046
SUCESS!
```

C>

运行 MASM.EXE 时, 必须输入口令字 8709046, 否则不能运行。例如:

```
C>MASM
PASSWORD: #####(键盘输入为
8709046 以及回车
符)
```

Correct password.

Microsoft (R) Macro Assembler Version 5.00

Copyright (C) Microsoft Corp 1981-1985, 1987

All rights reserved.

```
Source filename [ASM]:DPLASM;
51798 + 0 Bytes symbol space free
0 Warning Errors
0 Sevre Errors
```

C>

下面是用户不知道口令字的情况:

```
C>MASM
PASSWORD: #####(键盘输入为
8709047 以及回车
符)
```

Incorrect password.

```
PASSWORD: #####(键盘输入为
8709048 以及回车
符)
```

Incorrect password.

```
PASSWORD: #####(键盘输入为
8709049 以及回车
符)
```

Incorrect password.

Illegal user?

C>

可见, 不知道口令字的用户不能运行宏汇编软件 MASM.EXE。可用这种方法来达到保护该软件的目的。

## AR-2463 打印机故障检修一例

建设银行桂林中心支行计算机室 蔡敏

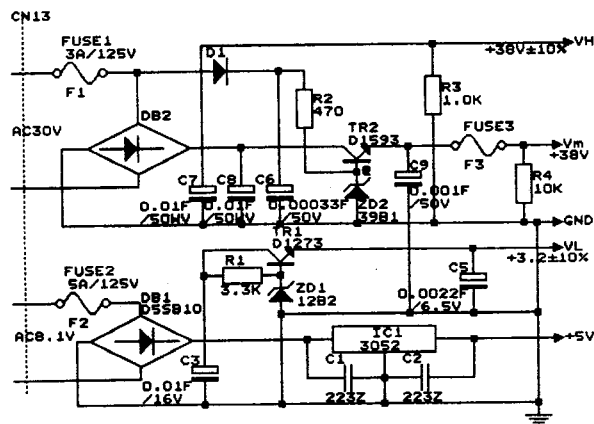
## 故障现象:

开机打印头有轻微抖动,并伴有“嗡嗡”声,字车没有初始化复位过程,联机指示灯不亮。

## 分析与维修:

AR-2463 打印机的结构基本上与其它类型的打印机一样。正常的打印机工作程序步骤是:加电电源指示灯亮,字体选择灯亮,字车向左移动到起始位置,联机灯亮,而完成整个初始化过程。

从故障的现象分析,该机的初始化过程不能执行下去,用遮光物在检测器上模拟初始化字车复位过程,初始化通过,联机灯亮。确定字车马达的激励电压没有送上去。图



@ 此处电压约 34V~36V

图 1

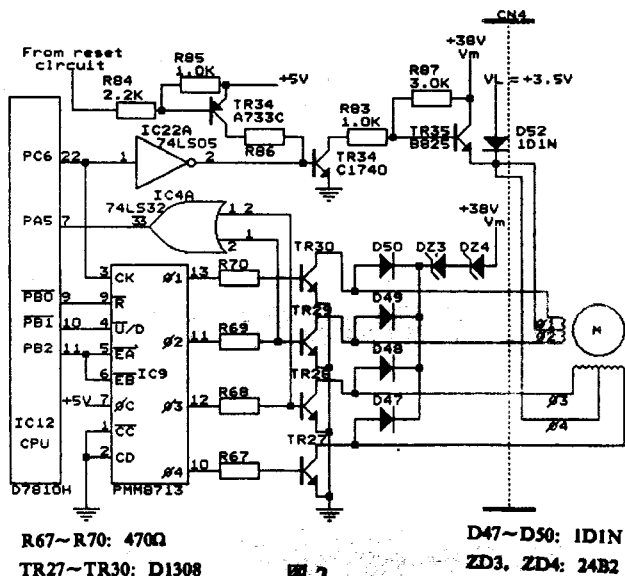


图 2

(1) 是 AR-2463 打印机电源原理图。检查中,发现电源电路中的 ZD2 被击穿, FuSe3 被烧断。根据经验,这种情况一般是大功率元件被击穿或线路短路造成过流、过载而损坏。

图 (2) 是字车驱动电路原理图。字车的逻辑控制是由一片 D7810H (CPU) 和一片 PMM8713 马达驱动控制芯片组成的控制电路,控制 TR27~TR30 四只功率管,送给相控信号,由 (CPU) 的 PC<sub>6</sub> (23 脚) 输出信号通过 TR34、TR35 对字车实行运动、静止的控制。

CPU 的 PC6 发出“0”指令,经过 IC22 (74LS05) 使 TR34 截止,此时,VL 经 D52 加到字车电机上使之保持静止。

当 CPU 发出“1”指令时,TR34 截止,V<sub>m</sub> 经过反馈电阻 R87 加在 TR35 的基极上,使 TR35 导通,将 V<sub>m</sub> 加到字车电机上,激励电机运动。由于 TR35 被击穿短路造成过载,使 FuSe3、ZD2 损坏。更换 TR35、ZD2 和 FuSe3,故障排除。

由于原型号器件在市场上不一定能买到,我们用其它参数相同的器件代用,如:TR35 (B825) 用 T1P42A 代用,ZD2 (39B1) 用 36V/1.5W 的稳压管代用,FuSe3 是一只可熔断玻封管,无法买到,但可以用>1A 的二极管或直接用低压 1A 保险管代用。

经过以上处理后,该打印机一切工作正常。

## 华源技贸商行敬告读者

本商行专营中华学习机,苹果机及各种微机,我们愿意为已拥有中华学习机或欲购买学习机的朋友提供热情的服务。本商行有陕西,佳木斯,北京等厂家生产的学习机以及与其相匹配的软盘驱动器,显示器及游戏棒,扩展卡等,我处有学习机资料 and 书籍三十余种,以及丰富的游戏,学习,工具和应用软件八百余种,除学习机外我处还经销各种档次的微机 (PC, XT, AT, AT/286, AT/386) 及各种档次的打印机及多种桌面印刷系统,并可维修各种微机,打印机,显示器, CRT 及激光印字机和复印机等。

欢迎人来函与我们联系,我们将竭诚为您服务,尽可能地满足您的要求。

地址: 北京市东城区沙滩五四大街 52 号  
 邮政编码: 100009 电话: 5127308  
 联系人: 于民

# PC 机总线故障检测

华南师范大学 张文锋

IBM-PC/XT, AT 机的故障中属总线故障是较多的, 故障主要表现为: 开机无反应, 无显示, 无信号声响, 出现“死机”。维修这类故障是比较困难的。为了避免盲目, 有目的进行检测分析, 本文就此给读者提供一种比较可靠、有效的方法。运用这种方法对绝大多数的总线故障源都能在很短的时间检测出来。

## 一、总线故障及现象

PC 机 (XT 或 AT 以及兼容机) 的总线根据传递信息不同分为地址总线, 数据总线和控制总线。同类总线按其所处的部位, 又分成局部总线, 系统总线和扩充总线等。总线上的部件是较多的, 多数为锁存器, 缓冲器。系统中主要的功能芯片都连接到相应的总线上。

当开机时 CPU 就立即向地址向发送地址数据, 从命令控制总线上发出命令信号, 从数据总线传递读写数据, 所有这些信息的任一位故障 (接地, 呈高电平或呈三态), 将使 CPU 无法正确执行 ROM BIOS 规定的软件, 以致使机器出现如下几种故障现象:

1. 开机后用示波器检测 CPU 的状态信息  $\overline{S_1}, \overline{S_0}$  发现它们瞬间有波形变化, 以后就呈现高电平状态, 机器进入停机状态。

2. 开机后  $\overline{S_1}, \overline{S_0}$  瞬间有波形, 但以后波形较固定, 机器进入死循环状态。

3. 开机后  $\overline{S_1}, \overline{S_0}$  立即变高电平。

4. 开机后机器停机, 用示波器检测数据、地址线发现有呈三态的现象。

应该指出, 总线故障会出现上述的几种现象之一, 但反过来, 不能说上述故障现象一定是总线故障。

对 XT 机来说: 当出现这种“死机”,  $\overline{S_1}, \overline{S_0}$  波形变得固定时, 我们应首先用示波器检测 PA 口 (8255 的 1~4 及 37~40 脚), 看看是否出现上、下跳动的现象 (0, 1 循环变化)。如果有, 可判断是自检 16KB 基本 RAM 出错, 即可能是 0 排 RAM, 延时器 TD1, 或 RAM 的地址多路器等故障造成。并不是本文所说的总线故障。对于 AT 机来说, 此种故障会显示“000000DDEE201”的错误信息, 其中 DD 是高 8 位故障芯片代码, EE 是低 8 位故障芯片代码。

## 二、总线故障的检测方法。

### 1、检测原理

在讲述总线故障检测前, 让我们先了解一下总线正常的操作。当一开机时, 机器就执行总线周期, CPU 执行取指令周期, 从地址引脚发送第一条指令的单元地址 FFFF0H; 并同时发出状态信号 ( $\overline{S_2}, \overline{S_1}, \overline{S_0} = 100$ ), 经

总线控制 8288 译码后产生  $\overline{MEMR}$  存储器读信号, 它们经过局部总线, 系统总线到扩充总线, 选通 ROM, 读取 ROM 的相应单元内容。该内容经扩充数据总线, 系统数据总线, 局部数据总线最后送入 CPU。对于 AT286 来说, 基本情况与 XT8088 一样, 不过 AT286 的取指周期的  $M/\overline{IO}$ ,  $COD/\overline{INTA}$ ,  $\overline{S_1}, \overline{S_0}$  为 1101。AT286 一次读取一字, 所以它从 OFFF0H 读出相应 16 位内容, 分别送到低 8 位和高 8 位数据总线上。

CPU 有个就绪信号 READY, 该信号有效时, CPU 就结束当前执行的总线周期, 否则处于等待状态。对 XT8088 来说 READY 为负时, AT286 的  $\overline{READY}$  为正时, CPU 不断插入等待状态, 当前总线周期一直延伸, 直到 READY 变正 (AT286 的  $\overline{READY}$  变负) 为止。

### 2、检测方法之一

开机前用万用电表的表笔插入扩充槽的 A10 与 B10 之间, 使 A10 接地。A10 是扩充槽上 I/O CHRDY 信号, 当它为 0 时, 就会使 CPU 的 READY 信号为无效状态。此时机器开机只执行第一个总线周期, 即取指令操作。由于 READY 无效, 机器就停留在这个周期, 并无限延时下去。在这个时刻, 总线上的信息保留稳定的状态。

对 XT8088 系统, 地址线 A19~A0 为 FFFF0H, 而数据线 D7~D0 为 EAH,  $\overline{MEMR}$  控制信号为 0 值, CPU 状态  $\overline{S_2}, \overline{S_1}, \overline{S_0} = 100$ 。我们可以通过逐位测量各数据位, 与正常情况进行比较, 一旦发现有异常, 就可以断定故障部位。通常地址总线从 CPU 开始向外进行逐位测试, 而数据总线应从 ROM 往 CPU 流向逐位测试。

对 AT286 系统, 由于开机是工作于实模式, 并取指令一次以 16 位进行, 所以 A0 没有使用, 测量地址总线由 A19 至 A1 为 OFFF0H。数据共有 16 位, 其低 8 位由一片 ROM 送出 EAH, 另一片 ROM 送出高 8 位 5BH。由于 80286 时序关系这时  $M/\overline{IO}$ ,  $COD/\overline{INTA}$ ,  $\overline{S_1}, \overline{S_0}$  为 1111 而不是 1101。

### 3、检测方法之二——“单步跟踪法”。

上述简易方法, 可检测出部分故障源, 但并不能检测出各种可能的故障。比如故障 A<sub>1</sub> 地址线接地故障就不能检测出。要能检测出各类总线的全部故障, 就需多跟踪几步, 执行多个总线周期, 充分对总线各位能否为 1 或 0 进行检验。为实现逐个总线周期进行操作, 需按下图制作一个电路板, 这个电路可以通过按键, 产生一个 I/O CHRDY 为正电平的方波, 使当前总线周期结束转入下一个总线周期; 一旦转入下一个总线周期, 电路的输入 I/O CHRDY 便变为无效低电平, 又使 CPU 停

# 硬盘不能启动的软维修方法

浙江省机电设备公司 徐云彪

计算机硬盘 0 磁头 0 磁道 1 扇区的硬损坏和软损伤, 使该硬盘无法启动, 甚至用软盘启动后也无法对该硬盘进行数据的存取, 这是我们计算机工作人员不希望看到的情况。目前已有不少文章介绍了上述问题的预防和维修方法, 多数采用的方法是: 用 DOS 提供的 FDISK 程序删除 DOS 分区, 然后重新创建新的 DOS 分区, 最后用 FORMAT 程序格式化新的 DOS 分区, 该方法虽然解决了硬盘不能启动的问题, 但硬盘中的数据无一例外地被抹掉了, 这对云需要保留硬盘中的某些数据的用户而言, 显然是一个重大损失。

基于上述原因, 笔者开发了“硬盘故障修理工具软件”并用该工具软件成功地修复了多台(次)不能启动的硬盘且 not 破坏盘中的任何数据和文件, 为了便于叙述, 现将该工具软件的运行结果的两份屏幕拷贝作得表一和表二。

表一中的各个字节依次取自硬盘 0 磁头 0 磁道 1 扇区偏移 1BEH 开始的 64 个字节, 它包含了四个分区的信息参数, 系统个分区的信息参数由 16 个字节组成, 其中各字节的含义表一中已有简要说明, 值得补充说明的是: 分区表中起始位置和终止位置中的扇区和磁道字节有如下关系: 扇区字节中二进制值的低六位(如表中的 D1H=11010001B, 低六位为 010001B=11H=17, 高二

表一:

引导	起始位置			系统	终止位置			引导相对扇区号				分区拥有扇区数				分区
	头	扇	道		头	扇	道	低	中	高	最	低	中	高	最	
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	1
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	2
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	3
80	04	01	03	04	04	D1	02	43	01	00	00	BC	FE	00	00	4

系统标志 00= ? 01= DOS12 02= XENIX 04= DOS16  
 字的说明 05=EXTEND 06= BIGDOS 64= NOVELL  
 75= PCIX DB=CP/M FF=BBT

引导标志字的说明: 00=No 即驱动器 A 启动 80=YES 即硬盘启动

表二:

硬盘分区表有关参数									
系统名	引导区	分区起始位置			分区终止位置			引导区相对扇区号	该分区拥有扇区数
		磁头	扇区	柱面	磁头	扇区	柱面		
?	No	0	0	0	0	0	0	0	0
?	No	0	0	0	0	0	0	0	0
?	No	0	0	0	0	0	0	0	0
DOS-16	Yes	4	1	3	4	17	770	323	65212

\*\*\*\*\*

留在这个新的总线周期处, 从而实现单步执行, 每执行一步我们都可用示波器检测各位信息值, 跟踪程序的执行, 直至认为找出故障为止。

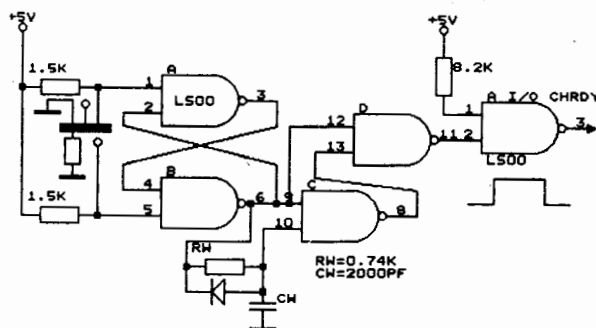
该电路主要要选择好  $R_w$  和  $C_w$  的值, 其值过小会使 CPU 不能实现单步执行, 过大会使 CPU 执行 2 个或多个总线周期才停下来。作者经过调试  $R_w$  选 740Ω,  $C_w$  选择 2000PF 后, 可以同时满足原装的 XT 机和 AT286 机使用。

采用单步跟踪电路板, 可以对 BIOS 程序进行跟踪, 其各步数据见表一。

表一:

XT 机			AT 机		
	地 址	数 据	地 址	数 据	
第一步(开机)	FFFF0H	EAH	FFFF0H(A <sub>0</sub> 不计)	EA5B	
第二步	FFFF1H	5BH	FFFF2H(A <sub>0</sub> 不计)	E000	
第三步	FFFF2H	E0H	FFFF4H(A <sub>0</sub> 不计)	F030	
第四步	FFFF3H	D0H	FFFF6H(A <sub>0</sub> 不计)	312F	
第五步	FFFF4H	F0H	DFF05AH(A <sub>0</sub> 不计)	D0E9	

注意: 当我们用跟踪进行测试时, 如果发现地址, 数据线有三态现象, 应进一步检测控制信号线。这种故障多为控制失效造成。另外, 这种检测也可以直接检测 ROM 正确与否。例如, 对于 XT 机第一步检测地址线值均正确为 FFFF0H, 而且 ROM 片选信号为低电平有效, 但 ROM 数据输出脚的值有错(不为 EAH), 这就有可能是 ROM 出错。



单步跟踪电路图



位 11B=3H=3) 存放真实的扇区号, 高二位存放的是磁道号的高位, 所以表一中的终止位置是: 扇区=17, 磁道=302H=3×256+2=770。引导区相对扇区号和分区拥有扇区数各由四个字节组成, 低字节在前, 高字节在后, 所以表一中引导相对扇区号=143H=1×256+4×16+3=323, 分区拥有扇区数=FEBCH=15×4096+14×256+11×16+12。

了解了上述知识后, 用户也就可以用汇编语言或在 DEBUG 下进行硬盘的软维修了, 方法如下:

#### A>DEBUG

-a100

```
IF22:0100 MOV BX,1F22
IF22:0103 MOV ES,BX
IF22:0105 MOV BX,0600 ;设置ES: BX用于存放硬盘读入的信息
IF22:0108 MOV AX,0201 ;AH=2读盘功能AL=1读一个扇区
IF22:010B MOV CX,0001 ;CL低六位存扇区号, CL高二位及
```

```
CH存磁道号
IF22:010E MOV DX,0080 ;DH中存磁头号, DL=80H表示硬盘C
```

```
IF22:0111 INT 13H ;执行BIOS中的INT 13号中断
IF22:0113 JB 0100 ;如果读硬盘操作失败, 则重试
IF22:0115 INT 3 ;硬盘0磁头0磁道1扇区已读入ES: 600
```

```
开始的内存区
;运行该程序, 下面是运行结果
AX=0000 BX=0600 CX=0001 DX=0080 SP=FFEE
```

```
BP=0000 SI=0000 DI=0000 DS=1F22 ES=1F22 SS=1F22
CS=1F22 IP=0115 NV UP EI NG NZ AC PE NC
IF22:0115 CC INT3
```

!-d7bc,7fd ;在屏幕上显示四个分区的参数

```
IF22:07B0 00 00 00 00 00 00 00 00 .....
IF22:07C0 00 00 00 00 00 00 00 00 .....
IF22:07D0 00 00 00 00 00 00 00 00 .....
IF22:07E0 00 00 00 00 00 00 00 00 .....
IF22:07F0 01 03 04 04 D1 02 43 01 .....C.
IF22:07F8 00 00 BC FE 00 00 ...Q..
```

-E XXXX ;XXXX表示需要纠正的错误参数的偏移量

;将所有错误参数纠正后, 继续下面的工作

-A100

```
IF22:0100 MOV BX,1F22
IF22:0103 MOV ES,BX
IF22:0105 MOV BX,0600 ;设置ES: BX以便将此开
```

```
始的内存写回硬盘
IF22:0108 MOV AX,0301 ;AH=3写盘功能AL=1
表示写一个扇区
```

```
IF22:010B MOV CX,0001 ;0磁道1扇区
IF22:010E MOV DX,0080 0磁头硬盘C
```

```
IF22:0111 INT 13
IF22:0113 JB 0100 ;写操作失败, 则重试
IF22:0115 INT 3 ;至此已将ES: BX开始的
```

```
内存信息写回硬盘
;运行该程序, 下面是运
```

行结果

```
AX=0050 BX=0600 CX=0001 DX=0080 SP=FFEE
BP=0000 SI=0000 DI=0000 DS=1F22 ES=1F22 SS=1F22
CS=1F22 IP=0115 NV UP EI NG NZ AC PE NC
IF22:0115 CC INT3
```

-Q ;硬盘软维修完毕, 退出  
DEBUG

最后为了硬盘的安全和正常运行, 建议各位计算机工作人员对硬盘0磁头0磁道1扇区作一个备份, 便于该扇区发生软故障时的恢复。制作备份时可将硬盘0磁头0磁道1扇区内容写到软盘的某一位置上, 恢复时便将软盘这一位置的内容写回硬盘0磁头0磁道1扇区, 下面是将硬盘0磁头0磁道1扇区内容备份到软盘A0磁头39磁道1扇区并且恢复它的操作示例。

#### ①制作备份示例

-a100

```
IF22:0100 MOV BX,1F22
IF22:0103 MOV ES,BX
IF22:0105 MOV BX,0200
IF22:0108 MOV AX,0201
IF22:010B MOV CX,0001
IF22:010E MOV DX,0080
IF22:0111 INT 13 ;读硬盘0磁头0磁道1扇区
IF22:0113 JB 0100
IF22:0115 MOV AX,0301
IF22:0118 MOV CX,2701 ;CH=27H=39磁道
IF22:011B MOV DX,0000 ;DL=0驱动器A
IF22:011E INT 13 ;写入A软盘0磁头39道1扇区
```

```
IF22:0120 JB 115
IF22:0122 INT 3
```

-E ;运行该程序, 下面是运行结果

```
AX=0001 BX=0200 CX=2701 DX=0000 SP=FFEE
BP=0000 SI=0000 DI=0000 DS=1F22 ES=1F22 SS=1F22
CS=1F22 IP=0122 NV UP EI NG NZ AC PE NC
IF22:0122 CC INT 3
```

-Q ;制作备份完毕, 退出  
DEBUG

#### ②恢复示例:

-a100

```
IF22:0100 MOV BX,1F22
IF22:0103 MOV ES,BX
IF22:0105 MOV BX,0200
IF22:0108 MOV AX,0201
IF22:010B MOV CX,2701
IF22:010E MOV DX,0000
IF22:0111 INT 13 ;读A盘0磁头39磁道1扇区
```

```
IF22:0113 JB 0100
IF22:0115 MOV AX,0301
IF22:0118 MOV CX,0001
```

```
IF22:011B MOV DX,0080 ;DL=80H硬盘C
IF22:011E INT 13 ;写入硬盘0磁头0磁道1扇区
```

```
IF22:0115 INT 3 ;至此已将ES: BX开始的
```

```
内存信息写回硬盘
;运行该程序, 下面是运
```

-E ;运行该程序, 下面是运行结果

## 软盘 0 磁道损坏后文件的恢复

福建省公安厅十一处 黄文

软盘驱动器经过一段时间使用后,较容易出故障。在使用有故障的软驱读写软盘时,很可能导致软盘的 0 磁道被划伤,出现不能读写现象。虽然软盘数据量不如硬盘大,但在没有备份的情况下损坏软盘上的文件,也是一件伤脑筋的事。笔者曾经遇到过这样的问题,并摸索到了解决的方法。

首先利用 ptools R4.11 的“磁盘及特殊功能”将 0 道损坏的软盘全盘拷贝到另一张完好的软盘上。在拷贝过程中可以见到读源盘时出现:

```
TRACK    0 1 2 3.....
SIDE     0      E.....
SIDE     1      E.....
```

即 0 道损坏。拷贝结束后报告目标盘可能是不能使用的。因为源盘 0 道读错误,目标盘 0 道没有被写上,即 0 道未拷贝,当然用 DIR 查看不到目标盘上的文件。接着用 C>CHKDSK /F A: 命令对目标盘进行恢复处理,提示:

```
XXX lost clusters found in XXX chains.
Convert lost chains to files (Y/N)?
```

回答 Y, 将恢复一个分离文件中的每一条链, 执行完后在状态报告中有一条:

XXXXXX bytes in XXXX recovered files 信息,并在盘上产生 FILE0000.CHK, FILE0001.CHK... FILEnnnn.CHK 一系列文件, 这些文件就是在刚才损坏 0 道盘上的文件。最后根据 FILEnnnn.CHK 的内容,逐一换成原文件名,具体恢复办法:若是文本文件,用 type 查看文件内容,根据内容换名,若是非文本文件,先把.CHK 扩展名换成.EXE 扩展名,运行之,看执行结果再换名,其他文件就不一一赘述。注意所恢复的文件长度均为 1024 的倍数,亦可根据长度推测原文件名。顺便提一下,0 磁道损坏的软盘重新格式化一下仍可使用。■

```
AX=0050 BX=0200 CX=0001 DX=0080 SP=FFEE
BP=0000 SI=0000 DI=0000 DS=1F22 ES=1F22 SS=1F22
CS=1F22 IP=0122 NV UP EI NG NZ AC PE NC
1F22:0122 CC INT3
```

~9  
:恢复工作完毕,退出  
DEBUG ■

## 9P.EXE 中的错误及更正

太仑工业学校 陈江海

9P.EXE 是 CCDOS4.0 中的九针打印机驱动程序,和 NEW9P.EXE 相比所输出的文本质量稍差一点,且字体也不及 NEW9P.EXE 的多,其原因是 CCDOS4.0 中配有众多的支持 24 针打印机的高级汉字打印驱动程序,9P.EXE 只是象征性地表示对九针打印机的支持,已显得不是太重要了。但是 9P.EXE 和 CCDOS4.0 整个系统一样具有较高的运行速度,和 NEW9P.EXE 相比要快得多。对于九针打印机的用户来说,使用它来输出一般汉字文本,可以提高工作效率,仍具有一定的实用价值。

9P.EXE 能输出从 A 到 H 的八种字体,在高级程序设计语言中利用 ESC 换码序列或在操作系统下利用 CTRL-F10 均可方便地变换字体。在使用中我们发现当输出 C、D、G、H 四种字体时,打印的汉字明显变形。打印机在打印这四种字体的汉字时是分两步完成的,先打上半部点阵,再打下半部的点阵,两部分合起来成为一个完整的汉字。经过仔细观察发现 C、D、G、H 四种字体的汉字,其上半部点阵和下半部点阵在衔接时有一部分明显重合,从而使印出的汉字在纵向上表现出不成比例的压缩,造成字体的明显变形。由此得出结论,字体的变形是由于上半部点阵打完换行时行距太小所致。用 DEBUG 对 9P.EXE 进行分析,找到其中控制换行的指令加以修改,即可解决此问题是。

具体作法如下:

①用 RENAME 命令将 9P.EXE 改为 9P.EEE;

②用 DEBUG 装入 9P.EEE 找到下面的代码段:

```
XXXX:0D92 B01B      MOV     AL,1B
XXXX:0D94 E83C00    CALL    0DD3
XXXX:0D97 B033      MOV     AL,33
XXXX:0D99 E83700    CALL    0DD3
XXXX:0D9C B001      MOV     AL,01
XXXX:0D9E F606010002 TEST    BYTE PTR [0001],02
XXXX:0DA3 7402      JZ       0DA7
XXXX:0DA5 B015      MOV     AL,15
XXXX:0DA7 E82900    CALL    0DD3
XXXX:0DAA B00A      MOV     AL,0A
XXXX:0DAF C3        RET
```

③用 A 命令将 XXXX: 0DA5 处的 MOV AL, 15 改为 MOV AL, 18;

④用 W 命令写盘后退出;

⑤用 RENAME 命令将 9P.EEE 改为 9P.EXE, 结束。经此修改后,字体即趋正常。■

# 计算机模拟化学中的酸碱滴定

国防科技大学化学系 许永飞

本文介绍的程序,虽然简练,但逼真地模拟了化学中酸碱滴定的全过程。程序中设置了两个软键 F1、F2 分别代表打开和关闭阀门。

$M_1$ . 滴定液(酸液)的浓度(MOL/L)。

$Ml_2$ . 被滴定液(碱液)的体积(Ml)

$M_2$ . 计算机内产生的随机数,作为被滴定液的原始浓度。

则达到滴定终点时消费滴定液的体积数为:

$Ml_1 = Ml_2 \times M_2 / M_1$ , 由此式取整数作为滴定所需时间,当计算机内部时钟达到这个时间时,使锥形瓶中的溶液变成红色,告知操作者,达到滴定终点。

具体原理和操作过程如下:

1. 输入滴定液浓度  $M_1$  和被滴定液浓度  $Ml_2$ 。

2. 计算机在彩色屏幕上画出实验装置图。

3. 当屏幕左上方出现一红色小圆圈时,按下  $F_1$ , 表示打开阀门,滴定开始,此时可见到滴定管下方“流出”一条红色溶体,同时管内液面逐渐下降。

4. 当锥形瓶中溶液变到红色时,按下  $F_2$ , 表示关闭阀门,滴定结束,“流下”的红线消失,液面停止下降,同时计算机记下从打开阀门到关闭阀门的时间  $MT$ , 并转化为体积数(时间秒与体积毫升数设定是等量关系),算出被测碱液的测量浓度:

$$MT_2 = MT \times M_1 / Ml_2$$

5. 计算机处理操作结果,打印出消费的滴定液的体积

$MT$ , 被滴定液的真实浓度和测量浓度并作比较,算出滴定误差:

$$X\% = (MT_2 - M_2) / M_2 \times 100\%$$

操作过程结束。

一次操作结果如下:

RUN

请输入滴定液的浓度  $m1 = ?$  0.5

请输入被滴定碱液的体积  $ml2 = ?$  20

Random number seed(-32768 to 32767)? 35

9374512

38

9374512

.95

$$\times \% = 1.338609\%$$

OK

本程序比较简单,特别是作图部分,有兴趣的读者可以更好地补充它,你不妨试一试。这个程序是用高级 BASIC 编成,在 IBM PC 机上通过的。

```
1 INPUT "请输入滴定液的浓度 m1="; M1
2 INPUT "请输入被滴定碱液的体积 ml2"; ML2
3 RANDOMIZE
4 LET M2=RND (0): PRINT M2
7 KEY OFF
10 SCREEN 1, 0: CLS
20 COLOR 10, 1
```

## 对《电源维修一法》的几点看法

太原机械学院自控系 常宝林

本刊 90 年 2 期刊登《微型计算机电源维修一法》,文中谈到了在线电解电容的容量下降,漏电加大的检测方法,对检修者是很有启发的,但有几处说法欠妥,在此提出我的看法,与杨培英同志商讨。

1. 电解电容可等效为一阻值很大的电阻与电容并联。当漏电加大时,相当于  $R$  变小,其对外所呈现的阻抗  $X_c$  也变小,这就使得式①

$$V_c = V_o \frac{X_c}{X_c + Z_o} \quad (1)$$

中的  $V_c$  也随着变小,如果当漏电大到使  $V_c < 2.64V$  (其它条件与原文相同) 时,就会误把损坏的电容当作好电

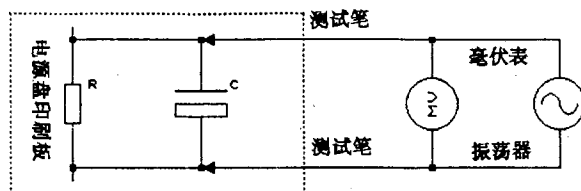
容。

2. 原文测试电路如图 1, 图中的电阻  $R$  为电源内阻,它与电容  $C$  相并联,测试电容时应考虑  $R$  对测试结果的影响,此时公式应修正为式②

$$V_c = V_o \frac{X_c // R}{X_c // R + Z_o} \quad (2)$$

若要忽略  $R$  对测试结果的影响,必须满足  $R \gg X_c$ , 此时音频信号发生器频率不能太低,选 100Hz 就不太合适了。

总之,用那种方法不能测电解电容漏电加大,对测量量下降也要考虑使用条件。



```

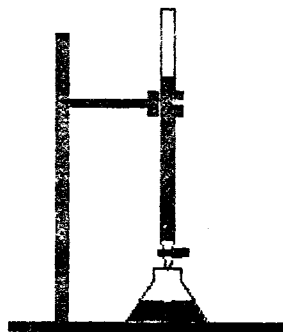
30 LINE (90, 20) - (95, 190), 3, BF
40 LINE (70, 190) - (190, 195), 3, BF
50 CIRCLE (93, 60), 3, 2
60 LINE (95, 58) - (130, 62), 2, BF
70 LINE (130, 53) - (134, 67), 2, BF
80 LINE (142, 53) - (145, 57), 2, BF
90 LINE (142, 61) - (145, 67), 2, BF
100 LINE (135, 5) - (141, 5), 1
110 LINE (135, 5) - (135, 141), 1
120 LINE (141, 5) - (141, 141), 1
130 LINE (135, 141) - (136, 147), 1
140 LINE (141, 141) - (140, 147), 1
150 LINE (134, 147) - (143, 150), 1, BF
160 LINE (136, 150) - (138, 160), 1: LINE (140, 154) -
(138, 160), 1
170 LINE (140, 146) - (144, 151), 1, BF
180 LINE (143, 146) - (147, 150), 1, BF
190 LINE (132, 158) - (144, 164), 1, B
200 LINE (132, 164) - (144, 164), 0
210 LINE (132, 164) - (120, 190), 1
220 LINE (144, 164) - (156, 190), 1
230 LINE (120, 190) - (156, 190), 1
240 LINE (126, 177) - (150, 177), 1
250 LINE (135, 5) - (141, 15), 1, B
260 PAINT (138, 13), 0, 1
265 LINE (135, 15) - (141, 141), 1, B
266 PAINT (138, 100), 3, 1
270 LINE (135, 15) - (141, 15), 3
280 LET J=0
290 LET T=0
340 LET ML1=INT (ML2 * M2 / M1) + 1
350 MI=0: ON TIMER (ML1) GOSUB 490
360 ON KEY (1) GOSUB 410
370 ON KEY (2) GOSUB 510
380 KEY (1) ON
390 KEY (2) ON: CIRCLE (10, 10), 5, 2
400 FOR Q=1 TO 3000: NEXT Q
405 GOTO 550
410 TIME$="0: 0: 0"
415 TIMER ON: LINE (137, 160) - (139, 180), 2, BF
420 IF MI=1 GOTO 480
430 LET J=J+1
440 IF INT (J / 80) <> J / 80 GOTO 470
450 LET T=T+1
451 IF T>95 THEN T=T-1
460 LINE (136, 14+T) - (140, 15+T), 0, BF
470 GOTO 420
480 RETURN
490 PRINT (138, 185), 2, 1: TIMER OFF
500 RETURN
510 U$=TIME$
520 LINE (137, 160) - (139, 177), 0, BF
530 LET MI=1
540 RETURN
550 SCREEN 0
560 A$=MID$ (V$, 4, 2): B$=MID$ (V$, 7, 2)
570 C=VAL (A$): D=VAL (B$)
580 GOSUB 600
590 END
600 MT=C * 60+D
605 PRINT MT, 消耗的滴定液的毫升数
610 LET MT2=MT * M1 / ML2

```

```

620 PRINT M2, 被滴定碱液的理想浓度
630 PRINT MT2, 被滴定碱液的实际浓度
640 PRINT "X%="; (MT2-M2) / M2 * 100; "%"
650 RETURN

```



## 一种简便的进制转换方法

福建南平师范学校 李锋

中华机上没有十进制与十六进制的转换，实为一憾事。常见报刊上发表有关文章，都是通过编程达到目的。下面介绍一种不须编制专用程序即可进行转化的方法。

熟悉程序在程序区存放格式的同志都知道，程序区是以一个程序做为基本单位存放。这个基本单位的开始两个字节存在一个链指针，指明下一个程序行以内存中那个单元开始存放，基本单位的随后两个字节是本程序行的行号，这是十六进制数，按高位在后，低位在前格式存放。接下去是程序行的内容，程序行的结尾 RETURN，在此用 0 代替。

了解以上知识后，不难设想，只要我们把须转化的十进制的数为行号，输入计算机，通过查找程序区的相应位置，即可得到相应的十六进制数。如要将 4096 和 53357 转化为十六进制数可进行如下操作：

```

NEW←
4096 PRINT←
53357 PRINT←
CALL-151←
0800-00 07 08 00 10 BA 00 0D
0808- 08 6D D0 BA 00 00 00

```

第一行程序是以 \$ 801 开始存放，其第三、第四个字节（即 \$ 803 \$ 804）内容为行号可结应的十六进制。故  $(4096)_{10} = (1000)_{16}$ ，同理  $(53357)_{10} = (D06D)_{16}$ 。

按照这种思路，我们可以方便地将 0~65535 间的整数转化为十六进制的数。假如将以上思路稍做变动，即可将十六进制的数转为十进制的数。限于篇幅，恕不赘言。

## APPLE 经验二则

南京市六合县第一中学 张亚栋

### 一、如何使随机数真正随机出现:

我们知道, APPLE 机有一条随机函数指令 RND (x), 但该指令造出的随机数并不是随机的, 每次开机, 出现的随机数都是一样的。例如:

```
10 FOR I=1 TO 5
20 PRINT RND (2)
30 NEXT I
```

每次开机后执行该程序得到的都是同样的 5 个数。这样, 某些用随机数出题的程序, 每次就会出同样的题目, 影响了使用效果。针对这种情况, 我们只需在程序开头加上 GET K\$: A = RND (PEEK (78) + PEEK (79) \* 256), 这样, 每次开机执行你自己的程序, 就不用担心随机数不随机出现了。

原理: 监控中的键盘输入子程序 (\$FD1B) 在等待键盘输入时, 不断把 (4E、4F) 加 1, 因此我们可以把 (4E、4F) 作为随机数基准。由于 RND (x) 函数是根据 x 与上一次随机数的值计算随机数的, 所以我们只要使每次开机后第一次执行的 RND 的自变量不同就可以造出不同随机数了。

### 二、谈在 BASIC 中执行监控指令且不破环栈指针:

目前在 BASIC 中执行监控指令一般把指令送入 A\$, 再执行 A\$ = A\$ + "ND823G": FOR I=1 TO LEN (A\$): POKE 511+I, ASCCMID\$ (A\$, I, 1)) + 128: NEXT: POKE 72, 0: CALL-144 但这种方法破坏了栈指针, 所以不能在子程序或 FOR……NEXT 循环中使用, 否则会出现错误信息, 如果直接调用监控 ROM 中的子程序, 那么在参数传递上又很麻烦, 不如使用监控指令来得直观形象。

针对这种情况, 我们只须把上面的 A\$ = A\$ + "ND823G" 改为 A\$ = A\$ + "N6: 68 68 60 N6G" 就可以既执行监控指令, 又不破坏栈指针了。

原来, 在执行 CALL-144 时, 计算机把返回地址压入堆栈, 但在 <法一>中没有在执行过监控指令后用 RTS 返回, 相反用 D823G 跳至 BASIC 解释程序中的下一条 BASIC 指令处理部分, 由于 "D823G" 相当于 JSR D823G, 所以又压入堆栈两个数据。这样, 在执行监控指令时先后压入了 4 个数据, 又不予弹出就回到了 BASIC, 因此使用堆栈的 FOR……NEXT 指令和 GOSUB 及 RETURN 就遭到了干扰, 从而出错。在我的方法中, 先用两条 PCA 指令把 "6G" 引起的人栈数据弹出, 再用 RTS 指令返回 BASIC, 这样就不致破坏指针了。■

## 解拆 CEC-I 五笔字型一例

舟山市电力公司变电工区 傅 剑

目前市面上流行的 CEC 硬汉字 WORDSTAR 1.0 (重庆版) 具有硬件运行环境低、功能齐全、输入容量大、形成文件兼容互换性好、并扩充了国家推广的五笔字型输入法等优点而受用户的宠爱, 笔者在使用该软件时发觉该系统的五笔字型输入法不但支持二级简码和选择式易学输入 (Z 键), 而且只占 0.8K 主存地址和 16K RAM 卡, 但该软件是直接利用 RWTS 读入整个系统程序并运行的, 因此笔者认为有必要把该输入法拆成 FILE (文件) 形式, 使用户能方便地把它运用到其它应用软件上。

### 具体解拆方法:

①启动 WORDSTAR 1.0 进入主菜单

②按 "X" 键退出系统

③输入: BSAVE WBZ, A\$8C00, L\$34A 回车

④输入如下机器程序并运行: \*300G 回车

```
* 300.33C
0300-AD 80 C0 AD 80 C0 A9 00
0308-85 F0 85 F2 A9 D0 85 F1
0310-A9 10 85 F3 A0 00 B1 F0
0318-91 F2 E6 F2 E6 F0 D0 F6
0320-E6 F3 E6 F1 A5 F1 C9 00
0328-D0 EC A9 00 85 F0 85 F2
0330-85 F3 A9 01 85 F1 AD 81
0338-C0 AD 81 C0 60
```

⑤输入: BSAVE WBZM1, A\$1000, L\$3000 回车

⑥输入如下机器程序并运行: \*300G 回车

```
* 300.33C
0300-AD 8B C0 AD 8B C0 A9 00
0308-85 F0 85 F2 A9 D0 85 F1
0310-A9 10 85 F3 A0 00 B1 F0
0318-91 F2 E6 F2 E6 F0 D0 F6
0320-E6 F3 E6 F1 A5 F1 C9 E0
0328-D0 EC A9 00 85 F0 85 F2
0330-85 F3 A9 01 85 F1 AD 81
0338-C0 AD 81 C0 60
```

⑦输入: BSAVE WBZM2, A\$1000, L\$1000 回车

⑧输入如下机器程序:

```
* 300.33C
0300-AD 8B C0 AD 8B C0 A9 00
0308-85 F0 85 F2 A9 D0 85 F1
0310-A9 10 85 F3 A0 00 B1 F2
0318-91 F0 E6 F2 E6 F0 D0 F6
0320-E6 F3 E6 F1 A5 F1 C9 E0
0328-D0 EC A9 00 85 F0 85 F2
0330-85 F3 A9 01 85 F1 AD 81
0338-C0 AD 81 C0 60
```

⑨输入: BSAVE YW, A\$300, L\$3D 回车

⑩输入如下 BASIC 主程序并存盘: SAVE HELLO 回车

```
10 D$=CHR$(4)
20 PRINT D$;"BLOAD WBZ, A$8C00"
30 PRINT D$;"BLOAD WBZM1, A$D000"
40 PRINT D$;"BLOAD WBZM2, A$1000"
50 PRINT D$;"BLOAD YW, A$300"
60 CALL 768: CALL 35840
70 PRINT D$;"PR#3: PRINT: PRINT"
80 END
```

# 对《电算 Waring 问题》 程序的修正

四川省南充一中 陈庆祥

贵刊在 1990 年第 6 期刊登了《电算 Waring 问题》一文，该文为验算“每一个自然数都可以表示为四个平方之和”的问题（即 Waring 在 1770 年提出的数学猜想）给出了一 BASIC 程序，并对  $N=1000, 100, 200, 300$  时求出了应有的答案。然而，经笔者分析与验证，发现该程序存在着三方面的问题。首先，程序中的 100 语句存在印刷上的错误：100 IF  $A^2+B^2+C^2+D^2=N$  THEN 300 中的 300 显然应该为 200（不存在 300 语句），再将修改后的程序运行：

```
10 INPUT N
20 FOR A=1 TO SQR (N+0.5)
30 FOR B=A TO SQR (N+0.5)
40 FOR C=B TO SQR (N+0.5)
50 FOR D=C TO SQR (N+0.5)
100 IF A^2+B^2+C^2+D^2=N THEN 200
150 NEXT: NEXT: NEXT: NEXT
200 PRINT N, A, " "; B, " "; C, " "; D
250 END
RUN
?29
29      6 6 6 6
RUN
?56
56      8 8 8 8
```

很容易看出，当输入 29 或 56 时，打印的结果是错误的，因为  $29 \neq 6^2+6^2+6^2+6^2$ 、 $56 \neq 8^2+8^2+8^2+8^2$ 。经试验，有 30 多个 100 以内的自然数根本不能表示成四个平方之和的形式。至于三位或更多位数的自然数尚未验证，但至少可以说明：“猜想”并不能适合每一个自然数。未经严格数学证明的猜想是不能作为定理使用的，所以，对于每一个输出的  $N$  值， $N=A^2+B^2+C^2+D^2$  能否在自然数范围内成立就具有两种可能的结果。这在程序中并不体现出来。既然 29 与 56 不能满足上式，为什么又会打印出上述错误的结果呢？这是因为对于这样的自然数，各次判断虽不能使条件成立，但退出循环后仍然要将 200 语句执行一次，打印出的是比各循环终值大 1 的数，而不是真正的解。

该文中说：如果把 250 改为 250 GOTO 150，还可得出更多的解，我们不妨照此理办后试试：

```
250 GOTO 150
RUN
?200
200      2 4 6 12
200      6 6 8 8
```

```
200      15 15 15 15
?NEXT without FOR error in 150
RUN
?29
29      6 6 6 6
?NEXT without FOR error in 150
```

运行后确实打印出全部解，但又冒出一组“假解”，而且接着打印出 NEXT without FOR error in 150 的错误信息。“假解”产生原因如前面所述，错误信息是因为打印“假解”后，仍无条件转向 150 语句执行，此时循环早已结束，循环栈已空，由于各循环变量未重新登录，当然执行不下去。（ $N$  取其他值时亦是如此）

发生上述错误的主要原因，是因为编程前对问题本身缺乏深入的了解，没有考虑到输入与输出数据的所有可能性，致使算法不够严谨造成的。对原文程序作简单修改即可克服上述毛病：

```
10 INPUT N
20 FOR A=1 TO SQR (N+0.5)
30 FOR B=A TO SQR (N+0.5)
40 FOR C=B TO SQR (N+0.5)
50 FOR D=C TO SQR (N+0.5)
100 IF A^2+B^2+C^2+D^2=N THEN PRINT N, A,
" "; B, " "; C, " "; D; T=T+1
150 NEXT: NEXT: NEXT: NEXT
160 PRINT "T="; T;
170 IF T=0 THEN PRINT "NO!"
180 END
RUN
?56
56      NO!
RUN
?200
200      2 4 6 12
200      6 6 8 8
T=2
```

上述程序虽能正确运行，但仍具有与原文程序一样的效率低下的缺点。当  $N=1000$  时，求出第一组解要用 3 分 40 秒，求出全部解要用约 1 小时（在 CEC-I 是运行），而不是原文所述的很快就可以打印出来。笔者采用预先计算  $N$  以内的平方数并用数组存放，选取序号与回溯的方法，重新编写了程序。以下程序当  $N=1000$  时，求第一组解仅用 6 秒，求全部解仅用 2 分钟，比原程序提高效率约 30 倍。

```
10 INPUT N: M=INT (SQR (N-4)): DIM A (M)
20 FOR I=1 TO M: A (I) = I*I: NEXT
30 FOR I=M TO 1 STEP-1
40 FOR J=I TO 1 STEP-1: P=A (I) + A (J)
45 IF P>N THEN 170
50 FOR K=J TO 1 STEP-1: Q=P+A (K)
55 IF Q>N THEN 160
60 FOR L=K TO 1 STEP-1: R=Q+A (L)
70 IF R=N THEN PRINT N, I, " "; J, " "; K, " "; L;
```



# 一个屏幕快速绕卷程序

湖南长沙国防科技大学 章文嵩

关于 APPLE 机上的屏幕处理问题。以前看到报上有关于屏幕绕卷处理的程序，程序长，而且绕卷速度慢，造成绕卷时有点失真。我想大概在两面的原因。第一，由于图形相应的内存单元对垂直方向向素跳跃分布，在屏幕绕卷时，必然产出屏幕行的首地址，在绕卷时用 JSR \$F411 子程序产生地址，势必影响绕卷速度。第二，那些程序几乎都是是一行一行地绕卷，这样也减慢了速度。

对于上述两个问题，我编了个程序，使得绕卷速度大大增快，效果很好。程序的主要原理：一开始地屏幕绕卷之前用 JSR \$F411 产生所有屏幕行首个单元的地址，把地址的高位存贮在 \$8E00-8EBF，而地址的低位存在 \$8F00-8FBF 内存单元里，避免了在绕卷时重复使用 JSR \$F411。绕卷时用 8 行移动一次，而不是一行一行地绕卷。还有程序本想设计思想非常巧妙，程序自身进行把屏幕的地址存入程序内存的有关单元而完成相互读写的过程，使得程序比较短。

本程序有向上绕卷和下绕卷的功能。向上绕卷的入口地址为 24576 (\$6000)，向下绕卷的入口地址为 24599 (\$6017)。\$E6 单元有特殊作用，决定哪一页向哪一页绕卷。例如先把荧屏显示第一页，给 \$E6 置 \$20，执行程序就能看到第二页图象逐渐绕卷到第一页。如果给 \$E6 置 \$40，则把第一页屏幕绕卷到第二页上。

还有 612E 单元值可以控制绕卷速度，置数值为 \$14 则速度较正常。

程序清单如下：

BR

\* 6000.6151

```
6000-20 36 61 A9 08 85 F4 A9
6008-00 85 F3 A9 B8 85 F1 A9
6010-C0 85 F2 20 F5 60 60 20
6018-36 61 A9 F8 85 F4 A9 B8
```

```
6020-85 F3 A9 00 85 F1 A9 F8
6028-85 F2 20 F5 60 60 BD 00
6030-8F 8D C5 60 8D CB 60 8D
6038-D1 60 8D D7 60 8D DD 60
6040-8D E3 60 8D E9 60 8D EF
6048-60 BD 00 8E 05 E6 49 60
6050-8D C6 60 18 69 04 8D CC
6058-60 69 04 8D D2 60 69 04
6060-8D D8 60 69 04 8D DE 60
6068-69 04 8D E4 60 69 04 8D
6070-EA 60 69 04 8D F0 60 60
6078-BD 00 8F 8D C2 60 8D C8
6080-60 8D CE 60 8D D4 60 8D
6088-DA 60 8D E0 60 8D E6 60
6090-8D EC 60 BD 00 8E 05 F0
6098-8D C3 60 18 69 04 8D C9
60A0-60 69 04 8D CF 60 69 04
60A8-8D D5 60 69 04 8D DB 60
60B0-69 04 8D E1 60 69 04 8D
60B8-E7 60 69 04 8D ED 60 A0
60C0-27 B9 00 20 99 00 20 B9
60C8-00 20 99 00 20 B9 00 20
60D0-99 00 20 B9 00 20 99 00
60D8-20 B9 00 20 99 00 20 B9
60E0-00 20 99 00 20 B9 00 20
60E8-99 00 20 B9 00 20 99 00
60F0-20 88 10 CD 60 A6 F3 86
60F8-F5 A6 F3 A5 E6 49 60 85
6100-F0 20 2E 60 8A 65 F4 AA
6108-20 78 60 E4 F1 F0 08 A9
6110-14 20 A8 FC 4C 01 61 20
6118-2E 60 A6 F5 A5 E6 85 F0
6120-20 78 60 8A 18 65 F4 85
6128-F5 C5 F2 F0 08 A9 14 20
6130-A8 FC 4C F9 60 60 A0 00
6138-98 48 20 11 F4 68 A8 A5
6140-26 99 00 8F A5 27 38 E5
6148-E6 99 00 8E C8 C0 C0 D0
6150-E7 60
```

```
T=T+1
150 NEXT
160 NEXT
170 NEXT
180 NEXT
190 PRINT "T+": T
200 IF T=0 THEN PRINT "NO!"
299 END
RUN
756
T=0      NO!
RUN
?1000
1000      28 14 4 2
```

```
1000 28 6 6
1000 28 10 10 4
1000 26 16 8 2
1000 26 14 8 8
1000 26 12 12 6
1000 24 18 8 6
1000 22 22 4 4
1000 22 20 10 4
1000 22 16 16 2
1000 22 16 14 8
1000 20 20 14 2
1000 20 20 10 10
T=13
```

# 一九九〇年广东省青少年计算机程序设计竞赛 试题分析及其选解 (下)

邬家炜

## 上机题部分

第二题: 这道题就其算法来说并不难, 难的是实现算法的编程技巧要求较高。因为题目要求用一个程序行来实现。从选手的答卷看, 归纳起来有两类型的问题: 第一类是属于审题不细心, 单纯地为了求快, 结果误了事, 把题目看成用一个程序实现, 结果用了几个程序行来表达算法, 这就不符合要求。另一类知道用一个程序行来写, 但由于编程技巧差, 常常要修改, 而这个程序行是较长的, 稍不注意就会超界, 有些选手虽然编出来了, 其程序的效率较差。据统计属于第一类错误的有 20 人, 占总人数的 66.67%, 而全对的只有 1 人这个数字不能不引起大家的关注。下面给出两种做法给大家参考。

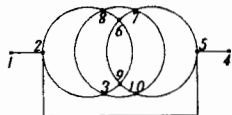
### 第一种做法:

```
10 A$(0) = "": B$(0) = "": B$(1) = CHR$(13): FOR I
= 10 TO 1000000: A$ = STR$(I): A$(1) = A$: L = LEN
(A$): L1 = INT(L/2): K1 = (L/2 - L1): B$ = LEFT$(A
$, L1): C$ = RIGHT$(A$, L - L1): X = VAL(B$) + VAL
(C$): K2 = (X * X) - 1: K = K1 * K2: PRINT A$(K): B$
(K): NEXT I
```

### 另一种做法:

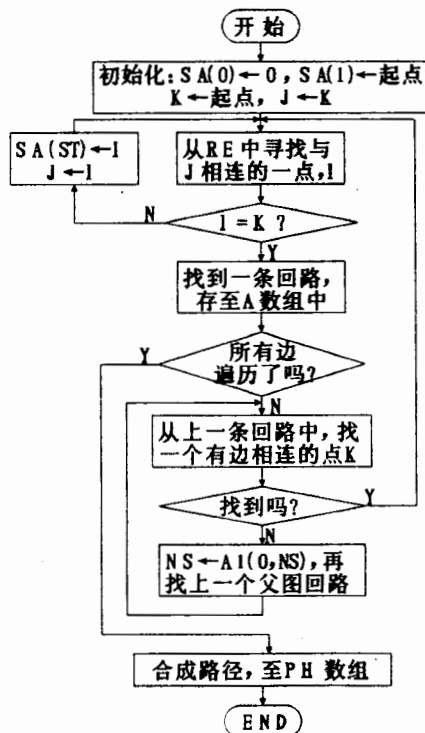
```
10 FOR I = 6 TO 999: A$ = STR$(I * I): L = LEN(A$):
M = VAL(LEFT$(A$, L/2)): N = VAL(RIGHT$(A$,
L/2)): B = M + N: B = B * B: C$(1) = STR$(B): P =
(B * I * I AND L/2 = INT(L/2)): PRINT C$(P): CHR$(
P * 13): NEXT I
```

第三题: 这道题难度大, 即使这类型的题考生会通过但未必是真正的理解其算法。下面在分析时, 我们先给图中的每个结点编码如下:



在图论理论中, 任一欧拉图可分成若干流通子图, 在找到遍历各路径的几个子图后, 则可将这几个子图的路径合起来, 遍历全图。下面给出算法的流程图及程序:

```
10 DIM PH(100), SA(100), A1(20, 20), A2(20, 20), RE
(20, 20)
20 T1 = 0: T2 = 0
30 READ N
40 FOR I = 1 TO N: FOR J = 1 TO N: RE(I, J) = 0: RE(J, I)
= 0: NEXT J, I
50 READ S, E
60 C = 0
70 READ I, J
80 IF I <> 0 THEN RE(I, J) = RE(I, J) + 1: RE(J, I) = RE
```



```
(J, I) + 1: C = C + 1: GOTO 70
```

```
90 RE(S, E) = RE(S, E) + 1: RE(E, S) = RE(E, S) + 1:
C = C + 1
```

```
100 SA(0) = 0: SA(1) = S: ST = 1: K = S
```

```
120 J = K
```

```
130 FOR I = 1 TO N
```

```
132 IF RE(J, I) <> 0 THEN 140
```

```
134 NEXT I
```

```
136 HOME: PRINT "INPUT GRAPH ARE WRONG.": GOTO
350
```

```
140 RE(J, I) = RE(J, I) - 1: RE(I, J) = RE(I, J) - 1:
```

```
C = C - 1
```

```
150 IF K <> 1 THEN ST = ST + 1: SA(ST) = 1: J = 1: GOTO 130
```

```
160 T2 = T2 + 1
```

```
170 FOR I = 1 TO ST: A1(T2, I) = SA(I): A2(ST, I) = 0:
```

```
NEXT I
```

```
180 A1(T2, 0) = ST: A2(T2, 0) = 1: A1(0, T2) = SA(0)
```

```
190 IF C = 0 THEN 240
```

```
200 NS = T2
```

```
210 FOR I = 1 TO A1(NS, 0)
```

```
212 FOR J = 1 TO N
```

```
214 IF RE(A1(NS, I), J) <> 0 THEN 230
```

```
216 NEXT J, I
```

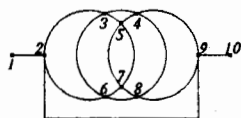
```
220 NS = A1(0, NS): GOTO 210
```

```

230 K=A1 (NS, I): A2 (NS, I) = A2 (NS, I) +1: I=J:
J=K: SA (I) =K: SA (0) =NS: ST=1: GOTO 140
240 I=1
250 FOR J=A2 (I, 0) TO A1 (I, 0)
260 IF A2 (I, J) <> 0 THEN 290
270 T1=T1+1: PH (T1) =A1 (I, J)
280 NEXT J: A2 (I, 0) =A1 (I, 0): GOTO 320
290 SG=0: GOSUB 1000
300 A2 (I, 0) =J: A2 (I, J) =A2 (I, J) -1
310 I=SG: GOTO 250
320 IF I>1 THEN I=A (0, I): GOTO 250
330 HOME: PRINT "PATH IS:"
340 FOR I=1 TO T1: RPINT PH (I): NEXT I
350 END
1000 FOR H=I+1 TO T2
1010 IF (A1 (I, J) =A1 (H, 1)) AND (A2 (H, 0) < > A1
(H, 0)) OR (A (H, 0) =1)) THEN 1015
1011 GOTO 1020
1015 SG=H: GOTO 1030
1020 NEXT H
1030 RETURN
2000 DATA 10, 1, 4, 1, 2, 2, 3, 2, 8, 2, 5, 3, 8, 3, 9, 9,
6, 9, 6, 6, 8, 9, 10, 6, 7, 8, 7, 7, 10, 3, 10, 7, 5, 10, 5,
5, 4, 0, 0
RUN
PATH IS:
1 2 3 9 6 8 7 10 9 6 7 5 10 3 8 2 5 4

```

下面介绍一种更简单的方法：这时设对图的结点编码如下：



```

5 DIM A (17, 2), B (17), C (17), D (17)
10 DATA 1, 2, 2, 3, 2, 6, 2, 9, 3, 4, 3, 5, 3, 6, 4, 5, 4,
8, 4, 9, 5, 7, 5, 7, 6, 7, 6, 8, 7, 8, 8, 9, 9, 10
20 FOR I=1 TO 17
30 READ A (I, 1), A (I, 2)
40 NEXT
60 I=0: D (0) =1: T=10
70 I=I+1
80 B (I) =B (I) +1: IF B (I) >17 THEN 500
90 IF C (B (I)) =1 THEN 80
100 IF A (B (I), 1) <> D (I-1) AND A (B (I), 2) <> D
(I-1) THEN 80
110 IF A (B (I), 1) =D (I-1) THEN D (I) =A (B (I), 2)
120 IF A (B (I), 2) =D (I-1) THEN D (I) =A (B (I), 1),
2)
125 C (B (I)) =1
130 IF I<17 THEN 70
140 FOR J=0 TO 16: PRINT D (J) "=>": NEXT J: PRINT
D (J): END
500 I=I-1: IF I=0 THEN END
510 C (B (I)) =0: GOTO 80
RUN
1==>2==>3==>4==>5==>3==>6==>2==>9==
>4==>8==>6==>7==>5==>7==>8==>9==>10

```

第四题：这道题难度大，它测试考生对 BASIC 的深

入了解是否透彻，例如对 POKE, PEEK, CALL 等语句的使用是否熟练，以及对 APPLE II 内存的分配，DOS 3.3 的内部命令的掌握情况。可见这涉及的知识面广，一个短短的程序要正确填写 21 个空缺也是不容易的。在这次竞赛中本题获满分的只有 2 人，获 24 分的有 2 人，其余都在 20 分以下。下面给出本题的参考答案。

```

2 DIM Z$ (16)
4 FOR I=0 TO 15
6 READ Z$ (I)
7 NEXT I
8 DATA 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
9 FOR I=0 TO 10
10 READ A
11 POKE 768 + I, A
12 NEXT I
13 DATA 32, 208, 248, 32, 83, 249, 133, 58, 132, 59, 96
15 HOME: PRINT "STAR-ADDRESS???"
16 FOR I=1 TO 4
17 GET A$: PRINT A$:
18 GOSUB 100
19 S (I) =V
20 NEXT I
21 PRINT
22 PRINT "END -ADDRESS???"
23 FOR I=1 TO 4
24 GET A$
25 PRINT A$:
26 GOSUB 100
27 E (I) =V
28 NEXT I
29 PRINT
30 POKE 59, S (1) *16+ S (2): POKE 58, S (3) *16+ S (4)
40 CH =E (1) *16+E (2): CL =E (3) *16+E (4)
50 CALL 768
60 IF PEEK (59) <= CH THEN IF PEEK (58) <= CL
THEN 50
70 END
100 FOR V=0 TO 15: IF Z$ (V) =A$ THEN RETURN
110 NEXT V: POP: GOTO 15

```

本程序的功能：只要用户在键盘上输入要查看的某个地址段的起始地址和终止地址，该程序就能把这段地址的指令以反汇编的形式显示出来。■

## 声 明

本刊 1990 年第 6 期《介绍一种快捷的汉字输入法-101 键扩展键盘的妙用》一文中介绍的对八笔字型操作系统的改进，经调查是上海市前进农场中心电脑室朱建平同志的科研成果。

特此声明。

本刊编辑部

毛毛虫病毒，又名 1575 病毒，是最近在 IBM PC 系列微机中流行的一种计算机病毒，它是一种文件型的病毒，能够迅速感染 COM 和 EXE 文件，并造成屏幕显示混乱、软盘驱动器运行不正常或死机等，对计算机的正常运行危害极大。

而原 COM 文件最前面的 0CH 个字节则被移到或加载到原 COM 文件尾巴的病毒程序部分偏移量为 0BH 的位置。

带毒的 COM 文件一般比原 COM 文件长 1575 至 1591 字节。

一般地，例如黑色星期五病毒，感染 COM 文件时，是把整个病毒程序（0710H 长）放在原 COM 文件的前头，因此，毛毛虫病毒在感染 COM 文件时，把病毒的主要部分放在 COM 文件后头，这在设计方面是有其特色的。

毛毛虫病毒对 EXE 文件感染时,和绝大多数的文件型病毒一样,是把病毒程序放在原 EXE 文件的尾巴后面,然后修改 EXE 文件的文件头,使带毒的 EXE 文件在执行时得到需要的 CS、IP、SS 和 SP 值,从而使之一开始就执行病毒程序。

带毒的 EXE 文件一般比原文件长约 1575 字节 (因此该病毒也称为 1575 病毒)。

毛毛虫病毒和绝大多数文件型病毒一样，是通过修改功能调用中断 int21H 的入口而达到传播病毒的目的。

毛毛虫病毒把 int21H 的入口改在  $\times \times \times \times: 04A8H$  处。

而在病毒 int21H 服务程序中,病毒设计者只是集中攻击 DOS 功能调用 11H (搜索第一个匹配文件) 和功能调用 12H (搜索下一个匹配文件)。

因此在 DIR 和 COPY 中使用通配符“\*”号时, 病毒 int21H 可迅速地感染多个 EXE 文件和 COM 文件,

笔者在检查一个硬盘时，硬盘中除隐含文件外，所有的 97 个 EXE 文件和 COM 文件都感染上毛毛虫病毒。

笔者在不带毒的软盘后动后,把三个不带毒的 EXE 文件拷具到硬盘中新开的一个子目录中,然后运行一个带毒的 EXE 文件,再退出,已发现毛毛虫病毒驻留内存并修改了 int21H.

此时在新开的 C 盘子目录上运行 DIR \*.EXE←

便可发现在这个 DIR 过程中, 该子目录上所有的三个 EXE 文件均感染上毛毛虫病毒, 文件长度均加长了 1575 字节。

由于毛毛虫病毒感染迅速，有时疯狂到失控的地步，造成有时会使得软盘驱动器运行不正常。

毛毛虫病毒对 EXE 文件和 COM 文件只感染一次，毛毛虫的病毒标记是 0CH, 0AH, 是放在病毒程序的尾巴位置。

用 debug 命令看一个带毛毛虫病毒的 EXE 文件, 在 CS: 0725H 处, 用 d 命令便会看到这个病毒标记。

毛毛虫病毒通过修改时间硬中断 int1CH 的人口达到病毒表演的目的, 病毒 int1CH 的新人口设置在  $\times \times \times \times: 06B0H$  处。

用 debug 命令把一个带毒的 EXE 文件调入内存, 在 CS: 069AH 处有 16H 个字节, 20 07 0F 0A 0F 0A 0F 0A 0F 0A 0F 0A 0F 0A 0F 0A F7 0EE E 00

这就是屏幕上显示出来的毛毛虫图形。

病毒设计者把 ES 设定在彩色中分辨显示器基地址 B800H 处, 然后不断变换 DI 值, 把毛毛虫的图形直接写到显示区中, 显示在屏幕上。由于该显示是写在时间硬中断 int1CH 中, 而被频繁地调用, 故在彩色中分辨显示器上会看到一条彩色的毛毛虫从左到右, 从上到下地在屏幕上爬行。

由于不同类型的显示器的显示区基地址不同(例如单色显示器的基地址为B000H),因此在有些显示器中,毛毛虫病毒发作时也看不到毛毛虫,但程序运行一般都受到干扰或死机。

毛毛虫病毒程序中设置了反跟踪的“陷阱”，用 debug 把一个带毒的 EXE 文件调入内存 CS:100H 处。

病毒程序在 CS: 160H 至 CS: 16EH 中, 把中断向量表 0000: 0000 至 001FH 共 10H 个字节搬到 CS: 13CH 为首地址的区域中, 然后把堆栈指针 SS: SP 设定在 0000: 0008 位置上, 接着在 CS: 176H 处有一个调用子程序命令 CALL 01C5H, 此时进入 01C5H 子程序, SS: SP 已指向 0000: 0006。

这时若设置断点, 发断点中断时, 系统在堆栈中压入断点返回地址 CS: IP 与标志寄存器共 6 个字节, 已复盖了放在中断向量表 0000: 0004 的单步中断 int1CH 的入口地址, 造成 debug 程序死机。

要检测驻留在内存中的毛毛虫病毒，可用贴上写保护纸的软盘中的 debug 来测试机器内存中的中断向量表。

若中断向量表 0000: 0070H 中 int1H 的入口段内地址为 06B0H，和中断向量表 0000: 0084H 中 int21H 的入口段内地址为 04A8H，则可判断该内存中已驻留有毛毛虫病毒，此时应立即关机，并用不带病毒的软盘重新启动。

要判断一个 COM 文件是否带有毛毛虫病毒，可用 debug 把该 COM 文件调入内存 CS: 100H 处。

然后用 u 命令看 CS: 100H 至 CS: 10BH 处的指令，若感染上毛毛虫病毒，该 COM 文件为：

```

××××: 0100 push CS
××××: 0101 move AX, CS
××××: 0103 ADD AX, ××××(各COM文件因
                                长度不同而不
                                同)

```

```

××××: 0106 push AX
××××: 0107 mov AX, 0100
××××: 010A push AX
××××: 010B RETF

```

若该 COM 文件前面 0CH 个字节与上面相同，则该 COM 文件为带毛毛虫病毒的文件。

要判断一个 EXE 文件是否带有毛毛虫病毒，可用 debug 把该 EXE 文件调入内存 CS: 100H 处。

然后用 u 命令看 CS: 100H 至 102H 处，若有毛毛虫病毒，则

```

××××: 0100 JMP 014C
××××: 0102 NOP

```

再用 d 命令看 CS: 0725H 处，若有毛毛虫病毒标记 0CH, 0AH，则可判定该 EXE 文件已感染上毛毛虫病毒。

用 debug 清除感染上毛毛虫病毒的 COM 文件步骤如下：

(1) 用不带毒的写保护的软盘启动（假设该软盘有不带毒的 debug.COM 文件），进入“A>”提示符。

(2) A>debug C: ××××.COM←(假设要消毒的 COM 文件在 C 盘)

进入 debug 提示符“-”状态。

(3) -u100

确认该文件具有如前所述的 COM 文件带毒特征，若不是则退出。

(4) -d CS: 104

取出保存在 CS: 104H 中的原文件长度（以节为单位，每节 16 个字节），不妨设为 y，然后转换成字节为单位，得原文件长 16y 字节（比真正未染病毒前的 COM 文件稍长 0 至 16 个字节）

(5) -R CX 16y

修改带毒 COM 文件的长度为 16y。

(6) 计算 16y+010BH 的值，不妨设为 Z

(7) -m CS: Z LOC 100

把被转移到 CS: Z 处的原 COM 文件的 0CH 个字节搬回到 CS: 100H 处，还原原来的 COM 文件。

(8) -w

把修改后的 COM 文件存盘。

(9) -q

退出 debug，则得到的 ××××.COM 是已解毒的 COM 文件，且可正常运行。

笔者有一个 CDEBUG.COM，原文件长 11904，感染上毛毛虫病毒后文件长 13495，比原文件长 1591 字节。按上述步骤消毒后，文件长 11920，比原文件长 16 字节。可正常运行。

另有一个 command.com，原文件长 25332，感染上毛毛虫病毒后文件长 26919，比原文件长 1587 字节。按上述步骤消毒后，文件长 25344，比原文件长 12 字节，可以正常运行。

用 debug 清除感染上毛毛虫病毒的 EXE 文件步骤如下：

(1) 用不带毒的写保护的软盘启动（设 A 盘有不带毒的 debug.com 文件），进入“A>”提示符。

(2) A>debug C: ××××.EXE (假设要消毒的 EXE 文件在 C 盘)

进入 debug 提示符“-”状态。

(3) -u100 或 dCS: 725

确认该文件具有如前所述的 EXE 文件带毒特征，若不是，则退出 debug。

(4) -d CS: 123 134

记录原来未带毒时的 EXE 文件重要参数，其中 CS: 123H 中有原 CS 值，CS: 125H 中有原 IP 值，CS: 131H 中有原 SS 值，CS: 133H 中有原 EXE 文件的 SP 值。

(4) -q

退出 debug 状态，回到“A>”提示符。

(5) A>COPY C: ××××.EXE C: 111

要修改 ××××.EXE，先要把该文件拷贝为中间文件，此处取名为 111。

(6) A>debug C: 111

把中间文件调入内存 CS: 100H 处。

(7) -R CX

把显示出来的 EXE 文件长度低 16 位字节数减去 600H，（若不够减，则还要用 RBX 修改 BX，用 BX-1 赋值给 BX，然后用 10000H+CX-600H，计算新的 CX），修改文件 111 的长度。

(8) -d CS: 104

取得 CS: 104H 中保存的该 111 文件所占实际扇区数，用该数（占 2 个字节）减去 3，得到消毒后的文件所占实际扇区数，（比原带毒文件少占 3 个扇区）

(9) 用 e 命令修改 EXE 文件的 5 个重要参数，其中：

# “428 病毒”的分析、诊断及防治

山西大同机车厂设计处 CAD 室 姚建华

最近,我们在新购买的微机中发现了一种新的病毒。由于一开始并没注意到新机器会在磁盘格式化后的瞬间,病毒迅速出现,使其疯狂传染。这是我们在运行 AUTOCAD R10.0 软件包时,其中的 HLIB.EXE 文件在刚拷入硬盘时,则可以执行;如再运行则死机,这种现象反复出现几次,用病毒检测盘进行检测,没有任何病毒显示。为此笔者想到了可能是一种新的病毒或是某一病毒的变种。因为在我厂是首次发现,所以暂称此病毒为“428 病毒”。

经笔者使用 PCTOOLS 及 DEBUG 对该病毒进行检查、跟踪分析,发现其感染性和隐蔽性都相当强。至现在为止,由于发现还算及时,没有使其到处传播,就已在实验室得到了控制。

但是,笔者也要提醒微机用户注意,不要过分依赖于病毒检测、消毒盘。要认真作好重要文件的备份,且要养成一种良好的习惯,对任何来路不明的软件应慎重使用。笔者在这次“428”病毒的袭击下,就损失了许多文件。

这种病毒并不集中在 DOS 的初始加载程序 (IPL),而最主要的攻击对象是 COMMAND.COM 文件。由于其没有任何明显地影响机器正常运行,因而难于发现,往往发现时,一大批用户程序已染上病毒,消毒处理较复杂,而且必须对所有被感染病毒的程序逐个消毒处理,必须确保无一遗漏,否则必将再次蔓延。

## 一、病毒的症状:

1. 机器运行速度明显变慢。(如新购的且不了解其运行速度的微机,便无从感觉到它的变慢程度)
2. 文件被感染后的长度增长 1.6K 字节左右。
3. 有一部分被感染的 EXE 文件或比较大的 COM 文件,将由于程序太大而不能执行。会出现“Program too big

to fit in memory”的提示。

## 二、病毒感染机制:

1. 本病毒首先对 COMMAND.COM 文件进行攻击,使整个系统感染。(由于前两个系统文件隐含,该文件一般情况下处于第一位置,是病毒首先攻击的目标)

2. 对 COM 和 EXE 文件进行感染,感染后的程序运行时,先执行病毒程序。

3. 本病毒对程序不进行多重感染。(这不同于“犹太人病毒”)

4. 对 COM 型程序的感染是将文件的前 12 个字节搬入“病毒体”的第 12 个字节处,并将文件头的前 12 个字节进行改写。所改写的内容为:

0E 8C C8 05 26 00 50 B8-00 01 50 CB

对 EXE 文件进行感染是通过修改 EXE 文件的引导装入部分,而实现病毒的装人与执行。并不保留原引导部分的数据。

病毒体附于文件的尾部。

5. 此病毒对系统隐含文件以及除 COM 和 EXE 外的数据文件,不进行感染。

6. 病毒修改了 INT24 的处理程序,屏蔽了 DOS 所有的报警信息,使感染在用户难于发现的隐蔽状态无声息地进行。

7. 本病毒在感染 SUN286 微机上的程序时,该程序生成日期将变为当前的系统日期;而在感染 AST 386SX/16 微机上的程序时,程序生成日期不发生任何变化。

8. 当系统被感染上该病毒时,如果显示某一个扩展名为 COM 和 EXE 的文件时则该文件被感染;如果显示的是一批 COM 和 EXE 型的文件(如 DIR\*.com 或 DIR\*.exe),则病毒按照文件的存储顺序一个一个地被感

CS: 104H 改为消毒后的文件所占实际扇区数

CS: 10EH 改为未染毒前的 SS 值

CS: 110H 改为未染毒前的 SP 值

CS: 114H 改为未染毒前的 IP 值

CS: 116H 改为未染毒前的 CS 值

(10) -w

把修改后的 111 文件存盘

(11) -q

退出 debug 状态,返回 DOS 的“A>”提示符。

(12) A>COPY C: 111 C: \xxx\\*.EXE

把中间文件 111 拷贝成 xxx\\*.EXE,则该 EXE 文件已不带毛毛虫病毒,且可正常运行。

笔者有一个 CKFRI.EXE 文件,未感染毛毛虫病毒时

文件长 14942,感染该病毒后文件长 16519,比原文件长 1577 个字节。按上述步骤消毒后,文件长 14983,比原文件长 41 个字节,可正常运行。

另有一个 SCAN.EXE 文件,未感染毛毛虫病毒时文件长 46928,感染该病毒后文件长 48503,比原文件长 1575 个字节,按上述步骤消毒后,文件长 46967,比原文件长 39 个字节,可正常运行该文件。

文件长度解密后出现一些误差的原因是消毒过程中对文件实际占扇区数及文件长度 mod (512) 作简化处理,避免繁杂计算而引起的,但不会影响到原文件的正常运行。

我们可以根据上述消毒要点,编写毛毛虫的解毒程序,则操作会更简便,且效果会更好。■



染,操作一次感染一个文件。

### 三、病毒的破坏作用:

(1) 被感染的程序在执行过程中速度显著变慢。

(2) 由于程序的传染性,使磁盘空间的有效利用率降低,传染过程中使得机器不断的读、写磁盘,使机器时间的有效利用率降低。

(3) 破坏了文件的可执行性。

### 四、病毒的分析:

这是对 MORE.COM 程序进行跟踪分析:

-DEBUG MORE.COM

-D 100 980

屏幕显示:

14CC: 0100 0E 8C C8 05 26 00 50 B8

14CC: 0108 00 01 50 CB B4 09 CD 21; 病毒执行头,由它转到病毒程序。

14CC: 0110 CD 20 C6 06 E5 01 19 B4

14CC: 0118 0F CD 10 88 26 E6 01 BA

原 :  
文 :  
件 :

14CC: 0210 76 65 72 73 69 6F 6E 0D

14CC: 0218 0A 24 BB 5E 08 2E FF 1E; 以下部分是病毒文件的执行体。

14CC: 0220 EB 4A 90 60 14 8E 02 53; 画线的数据为该病毒的检测标志。

14CC: 0228 FF 00 F0B4 30 CD 21 3D; 画点的数据为原程序的文件的头。

14CC: 0230 03 1F 74 09 BA FC 01 FF ; 十二个字节。

14CC: 0238 FF D1 10 00 01 00 00 00;

14CC: 0240 01 D1 10 00 00 00 00 01

14CC: 0248 00 10 1E 05 00 8A 43 B7

14CC: 0250 9A 56 12 80 00 C1 0E FD

14CC: 0258 9D 59 1C 12 87 58 8E 02

病 :  
毒 :  
体 :

14CC: 0970 FF A0 0F 75 EB BF 00 00

14CC: 0978 81 FF 9E 0F 75 05 C6 06

14CC: 0980 B0 06 CF C3 43 0C 0A 00 ; 此7字节为病毒免疫标志。

14CC: 0988 00 00 00 00 00 00 00 00 ;

经反汇编后:

-U 100 980(回车)

```
14CC:0100 0E      PUSH  CS
14CC:0101 8CC8    MOV   AX,CS
14CC:0103 052600  ADD   AX,0026 ;CS=CS+0026H
14CC:0106 50      PUSH  AX ;保存CS值
14CC:0107 B80001  MOV   AX,0100 ;IP=0100H
14CC:010A 50      PUSH  AX ;保存IP值
14CC:010B CB      RETF   ;转到病毒体
                        CS<=CS+0026
                        H; IP=100
14CC:010C B409    MOV   AH,09 ;原文件
```

14CC:9112 C606E50119 MOV BYTE PTR[01E5],19

14CC:021A BB5E08 MOV BX,085E

14CC:021D 2E CS;

病毒可执行部分:

14CC:021E FF1EEB4A CALL FAR [4AEB]; 远调用转到病毒执行体。

14CC:0222 90 NOP

```
14CC:0223 60      DB      60
14CC:0224 148E    ADC     AL,8E
14CC:0226 0253FF  ADD     DL,[BP+DI-01]
14CC:0229 00F0    ADD     AL,DH
14CC:022B B430    MOV     AH,30
14CC:022D CD21    INT     21
14CC:022F 3D031F    CMP     AX,1F03
14CC:0232 7409    JZ      023D
```

病 :  
毒 :  
体 :

14CC:0975 BF0000 MOV DI,0000

14CC:0978 81FF9E0F CMP DI,0F9E

14CC:097C 7505 JNZ 0983

14CC:097E C606B006CF MOV BYTE PTR [06B0],CF

14CC:0983 C3 RET ;病毒执行完毕返回原程序。

对于 EXE 文件与 COM 文件的传染过程大致相同但它修改 EXE 文件的引导部分,使其在装载后便执行病毒程序。文件头的引导信息数据被重新计算后填入相应的位置,原数据丢失。由此,这种情况便很难恢复。(有时不可恢复。)

### 五、病毒的消除及免疫

(一) 对于 COM 型文件,只要把原文件头恢复即可同时把病毒体消除。

(二) 对于 EXE 型文件,除了消除病毒外,还得把原程序的引导头数据进行恢复,这需要经过分析、计算后,再恢复到其相应的位置。

(三) 对于被感染的系统,只要把 COMMAND.COM 文件进行消毒,然后重新启动,系统便恢复如初。(最好把该命令文件设置为隐含系统属性)

在文件的尾部增加防病毒免疫标志,即“328 病毒”的免疫标志为:

1. 对于 EXE 型文件,免疫标志为“B0 06 CF C3 45 0C 0A 00”。

2. 对于 COM 型文件,其免疫标志为“B0 06 CF C3 43 0C 0A 00”(对于以上的免疫标志有一个即可达到免疫,而不考虑是什么类型的文件。)

### 六、病毒的诊断

根据以上的分析,并从病毒的结构和免疫标志来看,诊断检测时用病毒的标志“EB 4A 90 60 14 8E”及免疫标志“B0 06 CF C3 45 0C 0A 00”为最可靠、准确。可以直接判断出是不是“428 病毒”。但是如果没有检测软件,可以用系统内部命令显示文件的长度变化来判断是否被感染;不过这样做比较消极,因为在你在进行判断时,病毒的感染也随之进行。■

# 一类感染.COM 文件病毒的消除方法

石家庄市 54 所程控中心 方强

对文件中病毒的消除,一般是在详细地分析了该病毒的工作原理后才能实现,而这个过程不是每个人都能进行的,况且从分析病毒到提出消毒方法,需要一段较长的时间。本文试图就某一类病毒进行归纳,从中总结出一般的规律,使得读者不用分析病毒的全部内容,就可以在 DEBUG 下用手工方法完成带病毒文件的消毒。

在感染 .COM 文件的病毒中,有一类病毒感染文件时是把病毒体加在文件的尾部,如维也纳病毒,音乐病毒,扬基病毒等。同时病毒替换了原文件开始处的几个字节,使得带病毒文件的第一条指令成为跳转指令,而跳转的目的地址一般为原文件尾部,即病毒程序的开始。

带病毒文件执行时,首先跳到原文件尾部执行病毒程序,在这个过程中,病毒首先恢复原文件开始处的内容,然后再进行其它处理,完后跳到  $\times \times \times \times: 100H$  处(程序放在  $CS: 100H$ )执行原文件内容,病毒恢复原文件开始处字节的操作对应着一段小程序,而这段程序一段均在病毒程序的开始,很容易找到并读懂。

消去文件中的病毒需要解决两个问题:

- (1) 恢复原文件开始处被替换的内容
- (2) 确定原文件的长度或近似长度

从前面分析中,只要我们找到病毒程序开始处还原原文件开始处内容的程序并运行之,就可得到原文件被替换的内容,这时观察  $CS: 100H$  处内容,就会发现有了变化。由于病毒体加在文件尾部,一般情况下我们可以通过被感染文件的第一条指令,即跳转指令来确定原文件的长度或近似长度。假如文件的第一条指令为  $JMP 2F80$ ,那么原文件长度就定为  $2F80H-100H=2E80H$ ,其中  $100H$  是 PSP 段的长度,但有的时候  $JMP$  后面的地址不一定正好指向原文件尾部而是指病毒体内部,那么消毒后的文件长度要比原文件长度要大些。实际上只要文件执行时控制权不是被病毒程序首先得到,那么消毒后文件长度的增加是无关紧要的。况且有的病毒在感染文件时可能对原文件长度做一个调整(一般加长),然后把病毒体加到后面,这时原文件的长度就无法准确得到。所以我们将第一条跳转指令的转移地址减去  $00H$  作为原文件长度,并不影响原文件的功能。下面我们给出这类病毒的消毒过程:

(1) 利用 T 命令,跟踪一次或两次后,用 U 命令反汇编病毒执行程序的开始部分,从中找到还原操作的程序。

(2) 利用 G 命令或 T 命令运行还原程序。

(3) 根据 (1) 得到的文件长度写成一个新文件即可。

这种方法简单易学,不用分析病毒的全部内容,只须读懂还原程序就能消毒。一般还原操作中,多采用  $REPZ MOVSB (MOV SW)$  指令,此时  $DI$  应为  $100H$ ,  $CX$  为移动的字节数(或字数),  $DS: SI$  存放着原文件被替换内容,  $ES: DI$  指向要移到的位置。如果以后碰到类似的病毒,即使你并不知道它是什么病毒,照样可以完成消毒。

下面我们运用这种方法来消除几种病毒。

## (一) 维也纳病毒的消除:

维也纳病毒是一种非驻留型病毒,它只感染 .COM 文件。带病毒文件执行时,病毒就发作,然后对当前目录或上一级目录中的一个 .COM 文件进行感染。病毒感染时把病毒体加在文件尾部,共长 648 个字节,同时修改原文件开始三个字节,使其为一条转移指令。DEMO1.COM 是一个带病毒文件,在 DEBUG 下我们很容易找到还原原文件头被替换内容的程序。

```
A>DEBUG DEMO1.COM
-R
AX=0000 BX=0000 CX=3108 DX=0000 SP=FFEE BP=0000
SI=0000 DI=0000
DS=1106 ES=1106 SS=1106 CS=1106 IP=0100 NV UP DI PL NZ
NA PO NC
1106:0100 E97D2E JMP 2F80
-T
AX=0000 BX=0000 CX=3108 DX=0000 SP=FFEE BP=0000
SI=0000 DI=0000
DS=1106 ES=1106 SS=1106 CS=1106 IP=2F80 NV UP DI PL NZ
NA PO NC
1106:2F80 51 PUSH CX
-U
1106:2F80 51 PUSH CX
1106:2F81 BA7931 MOV DX,3179
1106:2F84 FC CLD
1106:2F85 8BF2 MOV SI,DX
1106:2F87 81C60A00 ADD SI,000A
1106:2F8B BF0001 MOV DI,0100
1106:2F8E B90300 MOV CX,0003
1106:2F91 F3 REPZ
1106:2F92 A4 MOVSB
1106:2F93 8BF2 MOV SI,DX
1106:2F95 B430 MOV AH,30
1106:2F97 CD21 INT 21
1106:2F99 3C00 CMP AL,00
1106:2F9B 7503 JNZ 2FA0
1106:2F9D E9C701 JMP 3167
-Q
```

从  $CS: 2F80H$  到  $CS: 2F93H$  的程序为还原操作,原文件长度为  $2F80H-100H=2F80H$ ,故消毒过程如下:

A>DEBUG DEMO1.COM←

-G=2F80, 2F93--

-RCX--

:2E80--

-W--

-Q--

到此整个消毒过程结束。

## (二) 音乐病毒的消除

音乐病毒也是一种只感染.COM文件的病毒,它修改了中断21H,8H的入口地址,同时驻留内存,它不是感染当前正在执行文件,当用户进行某些DOS命令操作(如拷贝文件)时,病毒就会对磁盘上的某一个.COM文件进行感染,音乐病毒感染文件时把病毒体加在文件尾部,同时替换了原文件开头四个字节,病毒体长度近3K字节。

DEMO2.COM是一个带病毒文件,在DEBUG下我们可以跟踪到病毒程序的执行部分,从中可找到原操作。

A>DEBUG DEMO2.COM

-R

AX=0000 BX=0000 CX=3960 DX=0000 SP=FFEE BP=0000

SI=0000 DI=0000

DS=1106 ES=1106 SS=1106 CS=1106 IP=0100 NV UP DI PL NZ

NA PO NC

1106:0100 E97D2E JMP 2F80

-T

AX=0000 BX=0000 CX=3960 DX=0000 SP=FFFE BP=0000

SI=0000 DI=0000

DS=1106 ES=1106 SS=1106 CS=1106 IP=2F80 NV UP DI PL NZ

NA PO NC

1106:2F80 E9DD08 JMP 3860

-T

AX=0000 BX=0000 CX=3960 DX=0000 SP=FFFE BP=0000

SI=0000 DI=0000

DS=1106 ES=1106 SS=1106 CS=1106 IP=3860 NV UP DI PL NZ

NA PO NC

1106:3860 9C PUSHF

-U3860,3887

1106:3860 9C PUSHF

1106:3861 50 PUSH AX

1106:3862 53 PUSH BX

1106:3863 51 PUSH CX

1106:3864 52 PUSH DX

1106:3865 56 PUSH SI

1106:3866 57 PUSH DI

1106:3867 55 PUSH BP

1106:3868 1E PUSH DS

1106:3869 06 PUSH ES

1106:386A B8E033 MOV AX,33E0

1106:386D CD21 INT 21

1106:386F 3CFF CMP AL,FF

1106:3871 7423 JZ 3896

1106:3873 8CCE MOV SI,CS

1106:3875 8EC6 MOV ES,SI

1106:3877 8B360101 MOV SI,[0101]

1106:387B 81C60601 ADD SI,0106

1106:387F B90400 MOV CX,0004

1106:3882 BF0001 MOV DI,0100

1106:3885 F3 REPZ

1106:3886 A4 MOVSB

1106:3887 07 POP ES

-Q

从文件开始到病毒执行代码处有两个跳转指令,第一个是跳到原文件尾部,第二个跳过病毒的数据区,从CS:3873H到CS:2887H为还原操作,原文件长度为2F80H-100H=2E80H

下面是消毒过程:

A>DEBUG DEMO2.COM--

-G=3873, 3887--

-RCX--

:2E80--

-W--

-Q--

## (三) 扬基病毒的消除

扬基病毒是一种驻留型病毒,它对当前执行文件感染,既感染.EXE文件又感染.COM文件,感染时先把原文件长度调整到可被16整除的位置,然后把病毒体加在文件尾部,长约3K字节,故原文件长度无法准确获得。扬基病毒笔者未作全面分析,只是发现其开始处有一段还原操作,故.COM文件的消毒方法自然可得。

在DEBUG下观察带病毒文件DEMO3.COM的第一条指令以及病毒程序的开始部分。

A>DEBUG DEMO3.COM

-R

AX=0000 BX=0000 CX=39C5 DX=0000 SP=FFFE BP=0000

SI=0000 DI=0000

DS=1106 ES=1106 SS=1106 CS=1106 IP=0100 NV UP DI PL NZ

NA PO NC

1106:0100 E94E36 JMP 3751

-T

AX=0000 BX=0000 CX=39C5 DX=0000 SP=FFFE BP=0000

SI=0000 DI=0000

DS=1106 ES=1106 SS=1106 CS=1106 IP=3751 NV UP DI PL NZ

NA PO NC

1106:3751 E80000 CALL 3754

-U3751,3775

1106:3751 E80000 CALL 3754

1106:3754 5B POP BX

1106:3755 81EBD407 SUB BX,07D4

1106:3759 2E CS:

1106:375A C6875C00FF MOV BYTE PTR [BX+005C],FF

1106:375F FC CLD

1106:3760 2E CS:

1106:3761 80BF5B0000 CMP BYTE PTR [BX+005B],00

1106:3766 7418 JZ 3780

1106:3768 BE0A00 MOV SI,000A

1106:376B 03F3 ADD SI,BX

1106:376D BF0001 MOV DI,0100

1106:3770 B92000 MOV CX,0020

1106:3773 F3 REPZ

1106:3774 A4 MOVSB

1106:3775 0E PUSH CS

-Q

尽管从CS:3768H到CS:3773H为还原操作,但SI

# 游戏图形图象的巧妙再利用

海口市 38010 部队 张昌平

我们在 IBM-PC 及其兼容机上玩电子游戏时,发现许多游戏的图形图象非常漂亮精美逼真,于是我们产生了这么一个想法:能不能对这些漂亮精美逼真的游戏图形图象进行巧妙利用,并进行再加工为我所用?为此我们分析了 DOS 操作系统的所有中断向量,决定选用屏幕硬拷贝中断向量 5H 对游戏的图形图象进行巧妙截取。

**设计思想:**我们知道 IBM-PC 机 CGA 显示卡的图形屏幕总是被划分为 200 行,每行对应映射区中的 80 个字节,中分辨率时,即每屏  $640 \times 200$  象素点时,每个字节存放 8 个点,即每一位代表一个象素点;低分辨率时,即每屏  $320 \times 200$  象素点时,每个字节存放 4 个点,即每二位代表一个象素点。总数为 100 行的偶数行存放在起始地址为 B800H 的内存区中,而总数为 100 行的奇数行存放在起始地址为 BA00H 的内存区中。在游戏程序运行过程中,我们只要保存起始地址为 B800H 的 16K 字节屏幕缓冲区的内容到硬盘或软盘中,就等于保存了当前屏幕的游戏图形图象。我们对保存在硬盘或软盘中的这些漂亮精美逼真的游戏图形图象进行再加工,就可以为我所用,就可以移植到我们需要的地方,如移植到我们自己编的程序中或在高分辨力显示器中显示。

**实现方法:**我们按键盘的屏幕硬拷贝功能键时,系统就中断当前正在运行的程序,而调用运行中断向量 5H 指向的屏幕硬拷贝服务程序,因此我们只要修改中断向量 5H 的入口地址,当按屏幕硬拷贝功能键时,不是转入运行屏幕硬拷贝服务程序,而是转入运行保存屏幕缓冲区服

务程序。这样我们就达到了录取游戏图形图象的目的。

我们用宏汇编语言实现了游戏图形图象录取程序,并在东海 0530 微机上调试运行通过。源程序大约 96 行,取名 INT5.ASM,编译连接生成可执行文件取名 INT5.COM。在运行游戏程序前,首先运行 INT5.COM,程序取代屏幕硬拷贝中断服务程序并常驻内存。当运行游戏程序时,假如你发现屏幕上显示的游戏图形图象非常漂亮精美逼真,并有保存再利用价值,那么请你按屏幕硬拷贝功能键,系统自动把当前屏幕上的图形图象保存在硬盘或软盘中。录取的游戏图形图象文件名为 BBB.n, n 为 A、B、C、D、E……等。BBB.A 为第一幅游戏图形图象, BBB.B 为第二幅游戏图形图象。游戏图形图象文件 BBB.n 的长度为 16000 个字节,前 8000 个字节对应起始地址为 B800H 映射区中的偶数行,每行 80 个字节,共 100 行;后 8000 个字节对应起始地址为 BA00H 映射区中的奇数行,每行 80 个字节,共 100 行。

我们用 TURBO-PASCAL-4.0 语言编制了一个在东海 CEGA 显示卡上显示录取的游戏图形图象的小程序,源程序约 50 行,取名 XS.PAS,编译连接生成可执行文件取名 XS.EXE。当 XS.EXE 运行时,首先提示你输入录取的图形图象文件名,然后就在屏幕 X 坐标为 100, Y 坐标为 100 的地方重现你录取的游戏图形图象。

INT5.COM 不仅能录取游戏图形图象,也能录取其他程序的图形图象。程序编译连接运行步骤如下:

C> MASM INT5.ASM

C> LINK INT5.OBJ

C> EXE2BIN INT5.EXE INT5.COM

C> DEL INT5.EXE

C> INT5

附源程序如下:

```
code segment
    assume cs:code,ds:code,es:code
    org 100h
start: jmp begin
main proc far
    cli ;关中断
    push ax
    push bx
    push cs
    push dx
    push si
    push di
    push ds
    push es
    push bp
    push cs
    pop ds
```

的值依赖于 BX,故我们还得得到 BX 的值,这可运行 CS: 3751H 到 3759H 之间的程序,原文件长度可定为 3751H-100H=3651H,消毒过程如下:

A> DEBUG DEMO3.COM

-G=3751, 3759

-G=3768, 3773

-RCX

:3651

-W

-Q

实际上 JMP 3751 不是跳到原文件尾部,而是跳到病毒体的中部。经实验如果文件长度再减去 7D1H,那么就更接近原文件长度了,另外我们发现从 CS: 3751H 直接执行到 CS: 3775H,其间不会产生转移,所以可以 G=3751, 3775 代替消毒过程中的两条 G 命令。

为避免意外,在消毒前最好对文件备份,消毒后应检查文件是否可运行,另外消毒应在无病毒环境下进行。■

```

inc count ;计数
mov al,count
add al,40h
mov file[4],al ;产生扩展名为26个字母之一的图象文
件名
mov ax,0b800h ;传送偶数行屏幕图象到图象缓冲区
mov es,ax
mov si,0

h11:
mov al,es:[si]
mov ds:buff[si],al
inc si
cmp si,8000
jc h11
mov ax,0ba00h ;传送奇数行屏幕图象到图象缓冲区
mov es,ax
mov di,0

h22:
mov al,es:[di]
mov ds:buff[di+8000],al
inc di
cmp di,8000
jc h22
push cs
pop ds
push cs
pop es
mov ah,3ch ;打开图象文件
mov dx,offset file
mov cx,20h
int 21h
jc h33
mov bufr,ax
mov ah,40h ;写图象缓冲区内内容到图象文件
mov bx,bufr
mov cx,16000
mov dx,offset buff
int 21h
mov ah,3eh ;关图象文件
mov bx,bufr
int 21h

h33:
pop bp
pop es
pop ds
pop di
pop si
pop dx
pop cx
pop bx
pop ax
sti ;开中断
iret ;返回断点
file db "bbb.x" ;图象文件名
dw 0
bufr dw 0 ;文件句柄
count db 0 ;计数器
buff db 16000 dup(0) ;图象缓冲区
dw 0
main endp
begin:
push cs
pop ds

```

```

mov dx,offset main ;置中断5H向量
mov ax,2505h
int 21h
mov dx,offset begin+1 ;程序驻留内存退出
int 27h
code ends
end start

```

```

program yxtx(input,output);
uses dos;
type
Registers = record
case Integer of
0: (AX,BX,CX,DX,BP,SI,DI,DS,ES,Flags: Word);
1: (AL,AH,BL,BH,CL,CH,DL,DH: Byte);
end;
var
fname:string[80];
ok:boolean;
tt:file of byte;
xx:array[1..10] of integer;
x0,y0,q,h,i,k,ff,color:integer;
bb:byte;
regs:registers;
procedure die; {写点子程序}
begin
with regs do begin ax:= $3500;
xx[1]:= color+3;xx[2]:= x0;xx[3]:= y0;
bx:= ofs(xx[1]);bp:= seg(x[1]);
intr($10,Dos.Registers:=regs);end;
end;

begin
write('file NAME: ');
readln(fname); {提示输入录取的游戏图形图象文件名}
assign(tt,fname);
({$I-} reset(tt);ok:= (ioread=0));{$I+};
if ok then begin
for q:= 0 to 1 do
for h:= 0 to 99 do
for i:= 0 to 79 do begin
read(tt,bb);
for k:= 0 to 3 do begin ff:= bb;
color:= hi(ff shr (k+1) * 2) and $03;
x0:= 100+i * 4+k; y0:= 100+h * 2+q;
if color < > 0 then die;
end; end; close(tt); end;
end.

```

## 小广告

请邮购 SHARP MZ-731 计算机汇编及反汇编语言磁带和中文说明书, 价 38 元 (含邮费), 增有 SHARP 公司未公开发表的 102 条指令, 款到发货。

汇款请寄: 南宁市第二水文队 黄学克 (收)

邮政编码: 530007

# 音乐程序

武汉锅炉厂设计处计算机室 水力

现在,大多数人们用 BASIC、FORTRAN 等高级语言编写程序、游戏、音乐软件也不例外,却忽视了用汇编语言设计一些程序。由于汇编语言是面向机器的语言,它有许多不可忽视的优点,其突出的优点是:第一,汇编语言具有目标码短、执行速度快、能够充分利用硬件资源及能够进行精确控制的优点。这些优点是高级语言远远不能比拟的。第二,汇编语言是一种低级语言,它与硬件的关系最密切。常常看到这样的现象,有些人已经学会了 BASIC 或 FORTRAN 之类的高级语言,但对计算机究竟是怎样工作的却知之不多,因而影响了对计算机的全面掌握和使用。在这里,我们在学习 IBM-PC (及兼容机) 的汇编语言同时,编制了一个音乐程序,以增加学习的兴趣。通过编制这样一个程序,更进一步了解了 IBM-PC 机的内部结构及各个芯片的用途,对以后编写其他的软件是一个很好的帮助。

## 一、预备知识:

我们知道,IBM-PC (及兼容机) 是一个以 8088 微处理器为中央处理机的核心部分之准 16 位机。其支撑 8088 工作的辅助电路有:

1. 振荡器及时钟信号发生器 8288A。主振荡器元件的频率经过三次分频后得到 4.77MHz 的中央处理机时钟信号。那么,每个时钟信号的周期为:  
 $1 / 4.77\text{MHz} = 210\text{ns}$ 。

2. 四通道 20 位直接内存访问 (DMA) 控制器 8237A。它用三个通道提供输入输出 (I/O) 设备与存储器之间的高速数据传送。第四个通道专门对动态存储器 RAM 进行刷新。

3. 三通道 16 位定时器和计数器电路 8253。它在系统中的作用分别是:0 号通道用于动态存储器刷新的定时器;1 号通道用于日期时间的基准;2 号通道用作扬声器的声调发生器,以便发生各种需要的音调。

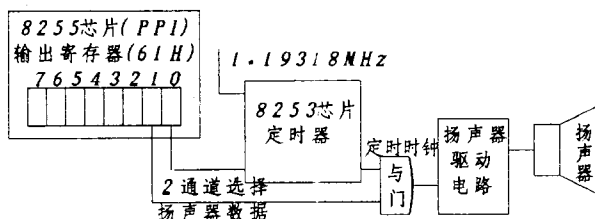
4. 八级中断优先控制器 8259A。它可实现对输入的八个中断信号排出优先级的顺序:0 号定时器、键盘控制电路、同步通信控制器中断、异步通信控制器中断、硬磁盘中断、软磁盘中断、打印控制器中断等。

另外,还有一个通用的可编程外设接口---8255 芯片。它有很多不同方式的配置方法。在系统主机板上,支持各种设备及信号,包括键盘、扬声器、配置开关和其他几种信号。这个芯片包括三个端口,叫做 PA、PB 和 PC。分别分配给它们的端口地址是 60H、61H、62H。此外,8255 芯片还有一个 8 位的命令寄存器,可用 I/O 地址 63H 存取。在开机时,BIOS 把 99H 的值写入 8255 的命令寄存器中去,将这个芯片进行初始化,使三个端口分

别设置为:PA、PC 端口为输入口,PB 端口为输出口。但我们经常使用 PB 端口为输入输出口。

## 二、指导思想:

怎样用汇编语言编程,使计算机发出美妙悦耳的音乐呢?这个问题的提出,归根结底是要搞清楚计算机中发出声音的扬声器是怎样工作的。首先,让我们来看一个扬声器驱动系统的示意图。



从图上我们不难看出,有二个可以驱动扬声器的信号。由 PPI 输出寄存器来选择其中一个或两者同时驱动扬声器。当第 0 位为 1 时,由 8253 定时器驱动扬声器;第 1 位为 1 时,扬声器发声直到第 1 位变为 0 时为止。

我们知道 8253 芯片上的 2 号通道是作为扬声器的声调发生器,改变通道中的数据,以便达到发出各种音调的目的。送给 2 号通道改变音调是频率值起作用的。频率值的计算为: (定时器时钟) / (普通音频) = (频率值)。例: C 调 1 的音频为 523Hz,那么 (频率值) =  $1193180 / 523 = 2281$ 。这样,我们只要向 8253 定时器中的 2 号通道寄存器,即 (42H) 口,送入 2281 值。扬声器就会发出音频为 523Hz 的声音来。

但我们怎样来选择 2 号通道呢?这是由 8255 芯片输出寄存器--- (61H) 口和 8253 定时器的命令寄存器--- (43H) 口共同来定义的。很有必要进一步看看 8255 芯片 (61H) 口和 8253 芯片 (43H) 口每位的情况。

8255 芯片 (61H) 口:

7	6	5	4	3	2	1	0
---	---	---	---	---	---	---	---

- 第 7 位: 选择 PA 口的来源,同时 1=键盘认为;
- 第 6 位: 0=禁止键盘时钟;
- 第 5 位: 0=允许由扩充槽的错误信号;
- 第 4 位: 0=允许 RAM 工作;
- 第 3 位: 0=启动盒式带马达;
- 第 2 位: 口 PC 位 0—3 的来源;
- 第 1 位: 扬声器数据;
- 第 0 位: 定时器 2 号通道选通。

由上可知,我们要发出声音,只需将 8255 芯片 (61H) 口的第 1、0 位置“1”即可。(码为 03H)

8253 芯片 (43H) 口:



7	6	5	4	3	2	1	0
---	---	---	---	---	---	---	---

第 7、6 位：是 0、1、2 通道选择，00=0 号通道，01=1 号通道，10=2 号通道；

第 5、4 位：是读写闸门寄存器方式，00=闸门含计数寄存器值，01=只读写高字节数据，10=只读写低字节数据，11=先读写低字节数据，然后读写高字节数据；

第 3、2、1 位：是定时器模式，有 0—5 种，其中模式 3 (011) 的输出产生对应的方波，且频率等于定时器时钟信号的频率，除以闸门寄存器的值；

第 0 位：0=二进制，1=BCD。

在这里，我们清楚地看到，在设置命令寄存器—(43H) 口时，必须设置 2 号通道为先读写低字节数据，然后读写高字节数据的方式、定时器模式 3 和作为二进制码输入。即 (43H) 口的码为 B6H。

综上所述，产生声音的最简单方法是，选择定时器 2 号通道，使它产生“成音”频谱 (20Hz~20KHz) 的方波。应用模式 3 及适当的闸门寄存器的初值来设定。若我们把 8255 芯片 (61H) 口的第 1、0 位都置成“1”，则定时器的方波输出将输入给扬声器。由于扬声器及其放大器电路无法适应于方波的陡峭变化，因此波形将被圆滑成较为悦耳的声音。一旦产生声音后，它将保持到关闭扬声器的时刻。持续时间的确定取决于你所希望的节拍。在 4/4 拍时，每一小节有 4 拍，一个全音符持续时间为 4 拍，一个二分音符持续时间为 2 拍，一个四分音符持续时间为 1 拍，一个八分音符持续时间为 1/2 拍，一个十六分音符持续时间为 1/4 拍。一个持续时间我们不难用 LOOP 指令来设计。我们查看 LOOP 指令的时钟周期数，将可发现有两个不同的数字，第一个为 17，代表 LOOP 指令若跳转时所需的时钟周期数；另一个为 5，代表 LOOP 指令不跳转时所需的时钟周期数。(即减一后 CX=0 情况)。所以当 CX 为 1 时的延迟时间为：(1\*17+5)/4770000 秒。若我们想得到 10ms 的延迟时间，那么它的 CX 值为多少呢？有算式，(CX\*17+5)/4770000=0.01

即 CX = (47700-5)/17=2806

也就是说，需要执行约 2806 次，才能获得 10ms 的延迟时间。

### 三、程序：

我们设计的音乐程序，是根据《梁祝》选曲—化蝶来的。

；———音乐程序———

```
stack segment para stack stack
stack ends
code segment
assume cs: code, ds: code, es: code
org 100h
begin: jmp start
f dw 330, 392, 440, 523, 587, 440, 523,
392, 784, 1047, 880, 784, 659, 784, 587
```

```
dw 587, 659, 494, 440, 392, 440, 523,
587, 330, 523, 440, 392, 440, 523, 392
dw 659, 784, 494, 587, 440, 523, 392,
330, 392, 330, 392, 392, 440, 494, 587,
440,
dw 392, 440, 523, 587, 784, 659, 587,
659, 587, 523, 440, 392, 330, 60000, 523
dw 440, 523, 440, 392, 330, 392, 440,
523, 392, 60000
dw 330, 392, 440, 523, 587, 440, 523,
392, 784, 1047, 880, 784, 659, 784, 587
dw 587, 659, 494, 440, 392, 440, 523,
587, 330, 523, 440, 392, 440, 523, 392
dw 659, 784, 494, 587, 440, 523, 392,
330, 392, 330, 392, 392, 440, 494, 587,
440
dw 392, 440, 523, 587, 784, 659, 587,
659, 587, 523, 440, 392, 330, 60000, 523
dw 440, 523, 440, 392, 330, 392, 440,
523, 392, 60000
dw 12, 9, 3, 9, 3, 3, 3, 6, 9, 3, 3, 3,
3, 3, 24
dw 9, 3, 6, 6, 9, 3, 6, 6, 9, 3, 3, 3, 3,
3, 24
dw 9, 3, 6, 6, 3, 3, 18, 3, 3, 3, 3, 3,
3, 3, 3, 18, 3, 3
dw 9, 3, 6, 6, 6, 3, 3, 6, 3, 3, 6, 6,
12, 3, 3, 3, 3, 3, 3, 3, 3, 36, 12
dw 12, 9, 3, 9, 3, 3, 3, 6, 9, 3, 3, 3,
3, 3, 24
dw 9, 3, 6, 6, 9, 3, 6, 6, 9, 3, 3, 3, 3,
3, 24
dw 9, 3, 6, 6, 3, 3, 18, 3, 3, 3, 3, 3,
3, 3, 3, 18, 3, 3
dw 9, 3, 6, 6, 6, 3, 3, 6, 3, 3, 6, 6,
```

```

12, 3, 3, 3, 3, 3, 3, 3, 3, 36, 12
kf      db 0
flag    db '歌名:《梁祝》选曲——化蝶'
        db 0dh, 0ah, ' ', 24h
start:  mov dx, offset flag
        mov ah, 9      ; 功能9调用
        int 21h
sing:   mov di, offset f ; 频率送 di
        mov si, offset t ; 延续时间送 si
        mov cx, 213     ; 循环 213 次
singl:  push di
        push si
        push cx
        mov bx, [si]    ; 把 si 的内容送 bx
        mov al, 0b6h    ; 8253 初始化
        out 43h, al
        mov dx, [di]+
        mov cl, kf
        sal dx, cl
        mov bp, dx
        mov dx, 12h
        mov ax, 34dch   ; 计算频率
        div bp
        out 42h, al     ; 送低地址
        mov al, ah      ; 送高地址
        out 42h, al
        in al, [61h]    ; 61 端口数据送 al
        push ax
        or al, 03h      ; 开扬声器
        out 61h, al
        mov ax, bx
        mov bx, 10
        mul bx
delay:  mov cx, 2806     ; 发音时间延续
dl10ms: loop dl10ms
        dec ax
        jnz delay
        pop ax          ; 关扬声器
        out 61h, al
        pop cx
        pop si
        pop di
        mov ah, 1
        int 16h
        jnz fh
        add di, 2
        add si, 2       ; 下一个频率和时间
        loop singl
        inc kf
        cmp kf, 3
        jnz 1100
        mov kf, 0
1100:  jmp sing
fh:    ret
code   ends
end     begin
    
```

#### 四.程序说明

在程序的数据区中,有标号为 f 和 t 的两片数据。分别表示音频数据和延时数据。我们定  $1=C$  4/4, 在手册中我们可查得, C 调的音频数据, 中间 C 调前后两个八度音符与它们的频率分别是:

音符	频率	音符	频率
C	130.810	C <sup>*</sup>	523.250
D	146.830	D	587.330
E	164.810	E	659.260
F	174.610	F	698.460
G	196.000	G	783.990
A	220.000	A	880.000
B	246.940	B	987.770
C	261.630	C	1046.500
D	293.660	D	1174.700
E	329.630	E	1318.500
F	349.230	F	1396.900
G	392.000	G	1568.000
A	440.000	A	1760.000
B	493.880	B	1975.500

\* 中间 C 调。在程序中设计的频率应四舍五入。

我们就是根据歌谱和这张表, 写出了这片数据。对于歌谱中的节拍, 我们定义: 1 拍为 1.2 秒 (即 12 个 100ms), 1/2 拍为 0.6 秒, 1/4 拍为 0.3 秒等等。其余说明见程序中。■

(接 44 页)

```

        cmp byte ptr [di], 0cdh
        jnz rep3
        mov ax, [bp+6]
        mov flag, ax
        and ax, 0feffh
        mov [bp+6], ax
        mov cx, 19
rep2:
        mov ax, offset cont
        mov [bp+2], ax
        mov ax, [bp+6]
        and ax, 0feffh
        mov [bp+6], ax
        jmp int1 exit
        dw offset exit
        dw 0
        endp
        ends
        end start
rep3:
        inc di
        dec si
rep1:
        mov al, [si]
        mov [di], al
        inc di
        dec si
        dec cx
        jnz rep1
int1 exit:
        pop cx
        pop si
        pop di
    
```

## 小 启

1. 《电脑》杂志没有合订本, 编辑部现有少量的杂志, 有 88 年 2、4、5 期; 89 年 2、4、5、6 期; 90 年只有第 6 期, 91 全年, 其它各期全部售完。以上每本邮购 1.10 元, 欢迎读者汇款到编辑部购书。

2. 凡来信询问有关问题, 请附贴足邮票的回邮信封。多谢合作。

# 对 CCDOS 4.0 (即西山 DOS) 的修改与补充

铁道部株洲电力机车厂技校计算机室 王大银

现行的 CCDOS 4.0 (即西山 DOS) 在汉字输入方法和输入速度方面比早期的 CCDOS 有了很大的提高, 可提供全拼双音, 双拼双音, 区位, 五笔字型, 表形码, 层次四角及电报明码, 共七种输入方案, 特别是其精华部分“双拼双音”给我们展示了一个崭新的双音编码输入体系。经笔者所办训练班的测验, 其输入文件的速度已达到 7300 字/小时, 常见字的单字测验速度也达到 3700 字/小时。这种双音输入方法很有必要全面推广。

但笔者在使用西山 DOS 的过程中遇到以下几个问题: 1. 该系统只能用硬盘启动。2. 该系统的 ASCII 字符不如 CCDOS 2.1 美观。前一个问题对没有硬盘的用户不能不说是一个相当大的遗憾。为解决上述两个问题, 笔者对 CCDOS 4.0 作了如下的修改。

## 一. 修改 CHLIB.COM 中的参数, 使之能用软盘启动西山 DOS

西山 DOS 中有一个字库读取模块 CHLIB.COM, 经分析发现其中有几处对硬盘进行操作的参数, 只要将它们修改为对 A 盘的操作, 就可以解决这个问题。

我们知道, IBMDOS 中, 磁盘读写是通过 13H 号中断来实现的, 绝对磁盘读(按扇区读)则是 25H 号中断。西山 DOS 中的 CHLIB.COM 在为字库申请内存及读字库时, 主要用到这两个中断。

调用 25H 号中断时, 驱动器代码 (0=A, 1=B, 2=C) 放入 AL 寄存器; 调用 13H 号中断时, 驱动器代码 (0=A, 80=C) 则放入 DL 寄存器。这样, 我们只要在 CHLIB.COM 中找到 25H 和 13H 号中断, 并将 A 驱动器的对应代码分别置入 AL 和 DL 寄存器。这样就可由 A 驱动器启动西山 DOS 了。

具体步骤如下:

1. 查找 INT 13, INT 25 及字库名“CCLIBJ.DOT”的地址。

(1) A>DEBUG CHLIB.COM

利用 DEBUG 对 CHLIB.COM 进行调试

(2) S100, 6414, CD 13

在 DEBUG 状态下查找指令 INT 13 的地址, 这里 CD 13 为 INT 13 的机器码, 而 6414 则为 CHLIB.COM 的长度加 100H, 这时机器显示如下三个地址:

2247:3D2F, 2247:3D11, 2247:379D

这里 2247 为代码段地址, 用户显示时可能有所不同。

(3) S100, 6414, CD 25

解释同上, 这时机器显示下面三个地址:

2247:3CD8, 2247:3D49, 2247:3D99

(4) S100, 6414, CCLIBJ.DOT

查找字库名“CCLIBJ.DOT”的地址, 机器显示 2247:3E9D

2. 修改有关内容:

找到指令 INT 13 和 INT 25 的地址后 (共 6 个地址), 再利用反汇编 U 命令找到指令 MOV AL, 02 和 MOV DL, 80。将其操作数 02 和 80 都改为 0。经查这 6 个操作数所在的地址为:

3D25, 3CCE, 3D38, 3D8E, 3D0E, 379A

可以利用 E 命令修改这六个地址的内容为 0:

E3D25 00

E3CCE 00

E3D38 00

E3D8E 00

E3D0E 00

E379A 00

对于文件名 CCLIBJ.DOT 我们可以用 D3E70 命令看到其前面有一个盘符 C: (地址为 3E9A) 将其改为 A: 即可。

A3E9A

2247:3E9A DB A:

3. 存盘

以上步骤完成后, 就可以将文件存盘。

RCX

CX:6314

RBX

BX:0000

W

这样修改后的 CHLIB.COM 将使西山 DOS 只对 A 盘进行操作。

## 二. 制作西山 DOS 系统盘

CCDOS 4.0 (即西山 DOS) 是一种汉字处理功能非常强的汉字操作系统, 经以上修改后, 我们可以把它广泛应用到 IBM-PC/XT 或 AT 及其兼容机上 (对带硬盘的用户则不需以上修改)。

这里介绍一种就没有硬盘的情况下, 制作西山 DOS 系统盘的方法。

将磁盘 A 格式化 (带系统), 然后将文件 CHLIB.COM, VDKEY.COM, CCLIBJ.DOT 拷入 A 盘将西山 DOS 的几个输入模块 PY.COM, WBX.COM, TELE.COM, BXM.COM, CCSJ.COM 拷入另一张盘 B 盘中 (不一定全部拷入, 视用户的需要而定)

在盘 A 中用 COPY CON AUTOEXEC.BAT 命令建

# 程序倒运行的实现方法

四川大邑 56025 部队 90 分队计算所 华松青

通常的程序都是由低地址向高地址逐条指令的顺序执行，这样的程序可以用调试程序进行反汇编，也可以方便地进行跟踪，但是如果我们程序的机器码不按这种顺序排列，而是由高地址向低地址排列，运行时也让其按这种反向顺序执行，即从高地址向低地址顺序执行，那么这样的程序既无法跟踪，也无法反汇编。

由于 8086 系列微机很容易设置单步中断，给以上想法的实现带来了可能，可以设想当执行完一条指令后将产生单步中断，我们利用该中断使一条指令复原，然后执行该指令，指令执行后又产生中断，根据此次中断和上次中断的返回地址即可算出该指令的长度，这样向低地址方向就可以找到下一条指令的起始地址，然后再把指令复原，再执行……；这样，程序的运行就好像是从高地址向低地址方向运行。

显然这样的程序中不能包含有循环和向回的转移指令，因为在程序运行中已将原来的代码破坏掉，转移过去已无法再恢复原来的指令；另外，对调用软中断的过程要

立批文件：

```
A>COPY CON AUTOEXEC.BAT
ECHO OFF
CHLIB
VDKEY
B:PY
B:WBX
ECHO ON
```

这里只读入两个输入模块（拼音与五笔字型），读者可视自己具体情况而定。

## 三、修改字符发生器

西山 DOS 的 ASCII 字符从外形上看，笔者认为没有 CCDOS 2.1 的字符美观，可以用下面的方法将 CCDOS 2.1 的字符字模移置到西山 DOS 中。

CCDOS 2.1 的字符发生器（即字模）在文件 CCCC.EXE 中，西山 DOS 的字符发生器在文件 CHLIB.COM 中，我们可以找到 CCCC.EXE 中第一个字符“!”的字模首址 2BA0，（CCCC.EXE 中字符“!”的字模为 30, 78, 78, 30, 30, 00, 30, 00），然后找到 CHLIB.COM 中“!”的首址为 2350，再将 CCCC.EXE 中从 2BA0 开始的 744(468H)个字节的内容移置到 CHLIB.COM 中。

注意，因 CCCC.EXE 中字符是 8×8 点阵，每个字符的字模占 8 字节，而 CHLIB.COM 中字符为 8×16 点阵，每个字符的字模占 16 字节，前者高度为后者高度的一

半，这样我们就必须在 CCDOS 2.1 字符字模的前后分别增加 4 个 0（或者前 5 个，后 3 个），然后移置到 CHLIB.COM 中字模的相应位置上。

具体步骤如下：  
1、查找 CCCC.EXE 中“!”的首址  
(1) A>REN CCCC.EXE CC  
A>DEBUG CC  
-S100, FFFF, 30 78 78 30 30 00 30  
-2247:2BA0  
-D2BA0, 3D07（此时可按 Ctrl-PrtSc 将显示的字模打印下来）

(2) 同样我们可以找到 CHLIB.COM 中“!”的首址为 2350  
(3) 用 DEBUG 将 CHLIB.COM 读入内存，然后将打印出的字模每 8 个字节前后共加 8 个 0，再依次键入从 2350 开始的 468H 个单元中。

这样，我们重新启动西山 DOS 后，就可以看到美观漂亮的 ASCII 字符了。

## 结束语：

任何一种软件都是在不断调试，不断完善的过程中完成的，特别是象汉字操作系统这样一些拥有广大用户的软件，更需要用户从应用过程中提出一些更好的方案和意见。以上只是谈了笔者对西山 DOS 的一些修改建议，其中可能有不准确之处，还望读者从中提出宝贵意见。

能组成一条完整的指令，当遇到指令代码为 0 时，则认为倒运行结束，恢复为正常状态；当遇到指令码为 0CDH 时，表明是一条 INT 指令，为了保证中断程序的正确运行，必须变为顺序运行方式，并在 INT 指令执行完后再恢复成逆运行方式，逆运行部分 INT 13H 后的 7 条指令就是为此目的安排的。由于倒运行部分的第一条指令前不会产生 INT1，所以倒运行部分的第一条指令应为单字节指令或顺序放置，本例中为一单字节指令 CLI，OFFADDR 初始值为第一条指令的地址，这样通过 INT1 的返回地址就可算出刚执行的指令长度，从而得到下一条指令的地址，并把该地址存入 OFFADDR，这样循环下去便使程序逆运行起来了。为了说明逆运行中转移指令的情况，本例逆运行部分有三条转移指令，第一条在 INT 13H 后，为 JC NEG RUN，它是当逆运行读盘失败时，退出逆运行方式，并重新初始化逆运行指令码，再运行逆运行部分，由于出现 INT 指令时，INT1 例程已将运行方式变为顺序运行方式，正是由于这一条件的存在，才使 JC NEG RUN 指令得以实现，在一般情况下，逆运行中采用向回的转移是不允许的；第二、三条转移指令分别为 JNZ ERROR 和 JMP EXIT，这两条指令均为向前转移并为近转移，同时也没有转出逆运行程序部分，所以无需任何改动便可在逆运行中实现。

程序中，标号 PROM START 到 CONT 之间的代码为逆运行部分，为了说明程序的运行状况这段程序我们以顺序方式写出，而这部分的实际代码为 PROM 这部分数据，NEG RUN 后的一段程序把 PROM 这段数据移到了 PROM START 处，由于 PROM 的数据实际上是 PROM START 部分的逆向代码（即最后一字节放到第一字节，倒数第二字节放到第二字节……），所以这两部分的长度一定是相等的，当我们把该程序编译，并用 DEBUG 观察时，会发现 PROM 并不完全是 PROM START 部分程序的逆向代码，有个别字节被改动了，这些改动是必须的，本例中改动了两处，第一处为 INT 16H 后的 JC NEG RUN，因为当代码被逆向安排后，该指令的地址已和顺序方式下的地址不同，所以转移的偏移也必然不同；另一处改动为 JC NEG RUN 指令下的第三条指令 MOV AX,OFFSET GGGG，原因基本是一样的，因为代码逆向安排后，GGGG 标号位置已发生了变化，所以 OFFSET GGGG 也发生了变化，必须更改。

该例改写完 A 盘 39 磁道 0 面 1 扇区后，将缓冲区 BUF 清 0，然后等待键盘输入一字符后才继续运行，当我们看到 A 驱动器工作灯亮后又熄灭时，便是在等待键盘输入，此时敲任一键便可进入逆运行状态，如果在修改 A 盘扇区时出错，将重试 5 次，如果仍不成功，则显示：“DISK OPERATION ERROR!”，然后退出；在实验时，我们分别可以试两种情况，首先插入一张已格式化好的盘在 A 驱动器，运行该程序，当等待键盘输入时敲入任一

键，则应显示“OK!”退出；另一种情况，当等待键盘输入时，换上一张未被该程序修改过的盘，再键入任一，则应显示“ERROR!”并退出。

该程序为表演性的，但运用这种方法可以为自己的应用程序增添光彩，我们已把该方法运用到应用程序的加密中，效果良好。

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

start:
    mov ax,0
    push ds
    push ax
    push cs
    pop ds
    push cs
    pop cs
    mov cx,5

repeat1:
    push cx
    mov ax,0201h
    mov bx,offset buf
    mov cx,2701h
    mov dx,0
    int 13h
    pop cx
    jnc edit
    dec cx
    jnz repeat1

error1:
    mov dx,offset string3
    mov ah,09
    int 21h
    mov ah,4ch
    int 21h

edit:
    mov si,offset buf
    mov al,0e9h
    mov [si+10h],al
    mov cx,5

repeat2:
    push cx
    mov ax,0301h
    mov bx,si
    mov cx,2701h
    mov dx,0
    int 13h
    pop cx
    jnc countinue
    dec cx
    jnz repeat2
    jmp error1

countinue:
    mov al,0
    mov di,offset buf
    mov cx,512
    cld
    repz stob
    push ds
```

```

mov ax,0
mov ds,ax
mov si,4
cli
mov ax,[si]
mov es:int1 ip,ax
mov ax,[si+2]
mov es:int1 cs,ax
mov ax,offset int1
mov [si],ax
push cs
pop ax
mov [si+2],ax
pop ds
sti
mov ah,0
int 16h

neg run:
mov cx,count
mov si,offset prom
mov di,offset prom start
cld
repz movsb
pushf
pop ax
or ax,100h
push ax
push cs
mov ax,offset exit
push ax
iret

prom start:
db 0
cli
mov ax,0201h
mov bx,offset buf
mov cx,2701h
mov dx,0
int 13h
jc neg run
mov ax,flag
push ax
mov ax,offset gggg
push cs
push ax
iret

gggg:
mov si,offset buf
add si,10h
cmp byte ptr [si],0e9h
jnz error
mov ax,0
jmp exit

error:
mov ax,1

exit:
sti
db 10 dup(?)

cont:
cmp ax,1
jz error2

mov dx,offset string1
mov ah,09
int 21h
jmp exit1

error2:
mov dx,offset string2
mov ah,09
int 21h

exit1:
cli
push ds
mov ax,0
mov ds,ax
mov si,4
mov ax,es:int1 ip
mov [si],ax
mov ax,es:int1 cs
mov [si+2],ax
pop ds
sti
mov ah,4ch
int 21h

buf
string1 db 512 dup(?)
string1 db 0dh,0ah,"OK! $"
string2 db 0dh,0ah,"ERROR! $"
string3 db 0dh,0ah,"DISK OPERATION error! $"
int1 ip dw ?
int1 cs dw ?
prom db 0,0fbh,0,1,0b8h,90h,4,0ebh,0,0,0b8h,6
db 75h,0e9h,3ch,80h,10h,0c6h,83h,2,6
db 0beh,0cfh,50h,0eh,1,0c9h,0b8h,50h,4,0d9h,0a1h
db 0c0h,72h,13h,0cdh,0,0,0bah,27h,1,0b9h,2
db 6,0bbh,2,1,0b8h,0fah
db 10 dup(0)

count dw 3bh
int1 proc
cli
push bp
mov bp,sp
push ds
push es
push ax
mov ax,[bp+4]
mov cs,ax
push cs
pop ds
mov ax,[bp+2]
sub ax,offaddr
sub offaddr,ax
mov ax,offaddr
mov [bp+2],ax
push di
push si
push cx
push es
pop ds
mov si,ax
cmp byte ptr [si],0
jz rept2
mov di,ax
mov cx,5

```

(转 40 页)

# 汉化 Turbo pascal 3.01A, Supercalc3, Wordstar 显示行参数修改

重庆市大足师范 宋运康

随着显示装置的不断更新,多种分辨率的彩色或单色显示装置层出不穷。按 IBM 标准,已有 CGA、EGA、VGA 三个系列。各种中文操作系统的不断推出,使汉化软件与多种显示装置的矛盾显得更为尖锐。新软件要求自适应多种显示装置已成趋势。CCDOS 4.0 似乎是汉化软件率先适应这种趋势的代表。一些推出较早的汉化软件,如 Turbo pascal 3.01A、dbaseⅢ、Supercalc3、Wordstar 由于多种原因(包括自身的很多优点),仍有相当广泛的应用。但这些软件推出时主要面对 600×200 的显示装置,汉字显示行数才限于 10 行。修改这些软件的显示行控制参数以适应多种显示装置,显然是一件十分有意义的工作。本文拟就汉化 Turbo pascal 3.01A、Supercalc3 1.30、电子工业部六所的汉化的 Wordstar 的显示行参数修改过程作一简单介绍,目的是希望能有更多的用户自己参与这项工作。

大家知道,各种软件实现屏幕显示的方式不外两类:一是调用 BIOS 功能,即 10 号中断(INT 10);二是把显示内容直接送显示缓冲区实现。前者较为简单,但显示速度稍慢。后者速度快,近期推出的许多新软件常采用后者,如 Turbo pascal 4.0、5.0、推出时间较早的 Turbo pascal 3.0、dbaseⅢ、Supercalc3、Wordstar 等大都采用第一种。这就给我们修改这些软件的显示行控制参数提供了方便:查找上述各软件中的 10 号中断的上滚功能调用(屏幕上滚入口参数:AH=6, AL=上滚行数, AL=0 表示填充格, BH=显示属性, CX=左上角行、列号, DX=右下角行、列号)。尤其是上滚清屏,一般不难找到显示行控制参数。这些参数来自某存贮单元,或是直接给的立即数。若是后者,修改立即数即可。若是前者,查找该存贮单元的赋值语句,修改所赋值即可。有的软件常常同时采用两种方式。

用 DEBUG 的查找命令 s 查找 b8 00 06 (即查找 mov, 0600 语句)虽常易于找到 DH 的入中参数,但显然必须继续查找所有的 cd 10 (即 int 10 语句,本文涉及的两个软件都各只十来处),逐处分析 DH 的入口参数是否需要修改。但是必须指出,即使所有的 int 10 调用处的入口参数都已检查处理,但修改结果可能仍未完全满足要求。这是因为可能还有别的语句(如 cmp xx, xx 语句)涉及到最大显示行数。对修改结果一般应作较长时间检验(本文的修改结果已都经一年左右时间试用,无错误发现)。当然根本的解决办法是分析完整的汇编程序清单或跟踪调试,但那工作量是很大的,也超出了本文讨论的范围。

以下分别说明三个常用软件的显示行参数修改。

## 一、汉化 Turbo pascal 3.01A 的行参数修改

C> debug turbo.com

用 r 命令可以从寄存器 cx 中得知文件长度 9d00,然后  
a100 9e00 b8 00 06(9e00 由 9d00+100 得到)得到下列

三处地址

0239

02b1

9ca5

用 u 命令分别对这三处进行反汇编,很快可以找到下列语句,清楚地表明 016b 是行参数存贮单元。

-u 9e95, 9eb7

0913:9C95 3C03 CMP AL,03 ; 显示模式识别  
0913:9C97 2E CS:  
0913:9C98 C6066B0119 MOV BYTE PTR [016B],19; 若是西文  
模式送 19 到存贮单元

0913:9C9D 7606

JBE 9CA5; 跳转到 9ca5 处对西文模式清屏

0913:9C9F 2E

CS:

0913:9CA0 C6066B010A

MOV BYTE PTR [016B],0A; 若是中

文模式送 10 到存贮单元

0913:9CA5 B80006

MOV AX,0600; 置上滚清屏功能参数

0913:9CA8 51

PUSH CX; 原 cx 值进栈

0913:9CA9 52

PUSH DX; 原 dx 值进栈

0913:9CAA 53

PUSH BX; 原 bx 值进栈

0913:9CAB 31C9

XOR CX,CX; 对 cx 置零

0913:9CAD B70F

MOV BH,0F; 置显示属性

0913:9CAF 2E

CS:

0913:9CB0 8A366B01

MOV DH,[016B]; 从行参数存贮单元

取得行参数

0913:9CB4 B250

MOV DL,50; 置列参数

0913:9CB6 CD10

INT 10; 清屏

用 s 命令查找 cd 10(即查找 int 10 语句)或查找 6b 01(查找 01 6b 单元),逐处反汇编后分析,不难找到下列语句。类似于上列语句,不再注释。

-u 9d62, 9d79

0913:9D62 2E

CS:

0913:9D63 C6066B0119

MOV BYTE PTR [016B],19

0913:9D68 E8F6BB

CALL 5961

0913:9D6B 2E

CS:

0913:9D6C 803EFA9D03

CMP BYTE PTR [9DFA],03

0913:9D71 7606

JBE 9D79

0913:9D73 2E

CS:

0913:9D74 C6066B010A

MOV BYTE PTR [016B],0A

0913:9D79 C3

RET

-u 9d96, 9dbe

0913:9D96 3C03

CMP AL,03



```
0913:9D98 760F JBE 9DA9
0913:9D9A 2E CS:
0913:9D9B C606B010A MOV BYTE PTR [016B],0A
0913:9DA0 2E CS:
0913:9DA1 C706309C9E06MOV WORD PTR [9C30],063E
0913:9DA7 EB0D JMP 9DB6
0913:9DA9 2E CS:
0913:9DAAC606B0119 MOV BYTE PTR [016B],19
0913:9DAF 2E CS:
0913:9DB0 C706309C9E0FMOV WORD PTR [9C30],0F9E
0913:9DB6 B84000 MOV AX,0040
0913:9DB9 8ED8 MOV DS,AX
0913:9DBB B80052 MOV AX,5200
0913:9DBE E8AFFE CALL 9C70
```

用 c 命令修改 9ca4,9d78,9d9f 单元的值 0a 为所需值即可。如对 21 行汉字屏幕,把 0a 改为 14。

## 二.汉化 supercalc 1.30 版的行参数修改(文件长度是 9e00b)

用 s100 9f00 b8 00 06 可以得到地址 205c,再用 u 命令可看到:

```
-u 2054, 2060
0913:2054 B90000 MOV CX,0000
0913:2057 BA4F17 MOV DX,174F; 无须修改 17
0913:205A B707 MOV BH,07
0913:205C B80006 MOV AX,0600
0913:205F CD10 INT 10
```

用 s100 9f00 cd 10 可以得到地址 1ffd,2005 等,然后用 u 命令可以看到:

```
-u 2002, 200c
0913:2002 B80600 MOV AX,0006
0913:2005 CD10 INT 10
0913:2007 C606A6010A MOV BYTE PTR [01A6],0A; (01a6 即为行参数存贮单元)
0913:200C E83D27 CALL 474C
```

用 c 命令修改 016a 和 200b 的值 0a 为所需值(21 行汉字屏幕为 14)即可。

三.汉化 wordstar(电子工业部六所)的行参数修改(文件长度是 5500b)

用 s100 5600 cd 10 可以得到地址 2d3b 等,再用 u 命令可看到:

```
-u 2d25, 2d3c
0913:2D25 B406 MOV AH,06
0913:2D27 B000 MOV AL,00
0913:2D29 8A2E4802 MOV CH,[0248]
0913:2D2D 8AF5 MOV DH,CH; (0248 即为行参数存贮单元)
0913:2D2F B100 MOV CL,00
0913:2D31 8A164902 MOV DL,[0249]
0913:2D35 FECA DEC DL
0913:2D37 8A3E8B02 MOV BH,[028B]
0913:2D3B CD10 INT 10
-u 4e76, 4e84
0913:4E76 B80006 MOV AX,0600
0913:4E79 B90000 MOV CX,0000
0913:4E7C BA4F17 MOV DX,174F; (修改 17 为所 14)
0913:4E7F 8A3E8B02 MOV BH,[028B]
0913:4E83 CD10 INT 10
-u 507f, 508c
0913:507F B80006 MOV AX,0600
0913:5082 BB000F MOV BX,0F00
0913:5085 B90000 MOV CX,0000
0913:5088 BA4F17 MOV DX,174F; (修改 17 为 14)
0913:508B CD10 INT 10
-u 3017, 301d
0913:3017 A04802 MOV AL,[0248]
0913:301A 3C17 CMP AL,17; (修改 17 为 14)
0913:301C 7301 JNB 301F
-u 4f32, 4f38
0913:4F32 A04802 MOV AL,[0248]
0913:4F35 3C09 CMP AL,09; (修改 09 为 13)
0913:4F37 7303 JNB 4F3C
```

还须修改 0248 单元的值为 14(都以 21 行汉字显示为例)。■

## IBM-PC/AT (0530) 微机系统原理及维修技术

### 培 训 班

电脑杂志社培训部将于 1991 年 6 月 27 日至 7 月 17 日在广州举办 IBM-PC/AT(0530)微机系统原理及维修技术培训班。

该班主要讲授 PC/AT 系统板;软、硬盘驱动器及适配器;高分彩显及适配器;M1724 打印机以及多种电源,UPS 电源电路的工作原理和维修技术。同时还详细地分析 PC/AT ROM BIOS 的工作原理。也就是说该班既是 PC/XT 维修技术的提高班,也是 PC/AT 维修技术学习班。

该培训班采用《IBM-PC/AT 微机系统原理及维修技术》作教材,该书分上下册和下册。其中上册十章和一个附录,主要内容包括:80286 的结构特点,IBM-PC/AT 主机板电路工作原理,高分辨彩色显示器及适配器电路原理,软、硬盘驱动器及适配器电路原理,M1724 打印机电路原理,多种微机电源以及多种常见的故障及排除方法,步骤。

下册共十章,主要是分析 PC/AT 微机 ROM BIOS 的工

作原理和各类中断服务程序实用程序的工作原理。在前九章中着重剖析 ROM BIOS 中各功能模块的程序流程图。第十章是 ROM BIOS 源程序清单,并附有各条指令的详细中文注释。所以这套书既是一套很好的 PC/AT 维修技术培训班的教材,又是一套 PC/AT (0530)微机的完整技术资料 and 维修技术指南。(该书每套定价 49 元,邮购加收 10% 挂号及邮杂费,杂志社有售。)

培训费 400 元(含上机实验费),书杂费实收,凡参加学习班者按 7 折优惠供应一套《IBM-PC/XT 电脑故障检修》录像教学片。食宿自理,统一安排。

#### IBM-PC/AT 维修技术学习班回执

姓名	性别	工作单位及详细地址	邮编	单位意见

请按此表格式面好填妥后于 6 月 10 日前寄达电脑杂志社。

邮局汇款地址:广州石牌华南师范大学电脑杂志社

银行汇款为:广州中国银行天河支行

账号:271-015170240 户头:电脑杂志社

联系电话:516911-3273, 330644

# TURBO PASCAL 程序中调用

## TURBO C 模块应注意的问题

国家建材局武汉建筑材料工业设计研究院 陈文杰

TURBO PASCAL 与 TURBO C 均是 Borland 国际公司开发的使用面广, 效率十分高的程序设计语言。然而, 关于二者之间的接口问题, 在有关资料上极少介绍, 本文就此问题谈谈在 TURBO PASCAL 程序中调用 TURBO C 程序模块时应注意的几个问题。

1. 在 TURBO C 模块中说明的数据不能被 TURBO PASCAL 程序存取, 共享数据必须被说明在 TURBO PASCAL 程序中。

2. 编译 TURBO C 程序模块必须采用小型内存模式。

3. 在 TURBO C 程序模块中不能采用 TURBO C 库函数, 原因在于它们没有正确的段地址。但是, 如果你有 TURBO C 库函数的源程序, 则不受此限制, 因为你可以将你使用的有关 TURBO C 库函数连同你的 TURBO C 程序模块重新编译形成正确的 .OBJ 文件结果, 供 TURBO PASCAL 程序连接调用。(TURBO C 库函数源程序可以向 Borland 国际公司购买获得)。

4. TURBO C 程序模块必须被编译形成正确的 .OBJ 文件, 然后, 通过在 TURBO PASCAL 程序中加入编译指令(\$L)将这个 .OBJ 文件连接进入 TURBO PASCAL 程序中。

5. 为了获得正确的 TURBO C 程序模块的 .OBJ 文件在 TURBO C 集成环境下采用如下命令:

TC / CCTOPAS 程序名

然后 ALT F9

在 TURBO C 命令行环境下采用如下命令:

TCC 程序名

6. 编译和执行 TURBO PASCAL 程序

下面请看两个程序实例:

```
/* CP1.PAS */
program cp1; (程序一)
uses crt;
{$L CP1.OBJ}
/* 将 TURBO C 形成的 .OBJ 文件连接起来 */
function sqc(i:integer; j:integer; k:integer)
:word; external;
/* 此函数定义在 TURBO C 程序模块 CP1.C 程序中 */
var
t:integer;
fa:word; /* 共享的数据变量, 必须说明在 PASCAL 中 */
```

```
begin
```

```
t:=4;
```

```
writeln(sqc((t-3), 2, t));
```

```
writeln(fa);
```

```
end.
```

```
*****
```

```
/* CP1.C */ (程序二)
```

```
typedef unsigned int word;
```

```
extern word fa; /* 此变量在 CP1.PAS 中定义 */
```

```
word sqc(int i, int j, int k)
```

```
{
```

```
fa=j+1;
```

```
return(i*j+k);
```

```
}
```

具体操作如下:

在 TURBO C 环境下编译 CP1.C, 生成 CP1.OBJ

1/ TC / CCTOPAS CP1.C

2/ ALT-F9

在 TURBO PASCAL 环境下编译和运行 CP1.PAS

以上只是笔者在实践中获得的几点心得, 不当之处请指正。 ■

## 思索电脑公司开业

广州市天河思索电脑公司于 3 月 30 日在广州天河五山路高科技产业区正式开业。该公司是一间推广高新技术产品、提供优质服务的全新型的全民所有制企业。

该公司主要从事计算机工程、系统应用软件、自动控制系统、CAD、网络工程、信息管理以及办公自动化管理等多方面的研究开发和推广应用工作, 并逐步从商业性销售转变为技术性销售。

“思索电脑”组建了一支训练有素、精明能干、具有全新意识和开拓进取精神的技术队伍, 公司的职员全是受过高等教育和专业技术培训的多方面人才。

该公司将本着对客户负责、对社会负责的精神, 坚持以一流的质量、一流的服务为广大客户提供应用软件开发的技术支持; 提供保证系统软件的支持; 并成为第一家提供设备三年免费保修服务以及报修 24 小时内作出妥善处理的新型电脑公司。 ■

# 计算机爱好者 软件库

计算机爱好者软件库本着质量第一, 用户第一的精神在软件交流工作中将开展优质服务。欢迎读者多提意见和建议。只有读者的热心支持, 软件库才能办好。欢迎个人、单位来软件库代售各自开发的软件。欢迎个人、单位交换软件。

一、I类软件 APPLE 类(为保证质量, 仅单面拷贝) 每片收10元, IBM 类软件每片收16元。II类软件按软件后标价计收。个人购买9折收费, 邮包费每次收5元。

二、订购软件请注明使用机型、主选目录和备选目录, 避免兼容性问题发生给您带来损失。

三、订购软件清单的每个软件后, 请注明片数及单价。

四、收到款后一周内寄出。软件寄出后一个月内(凭包裹单邮戳)不能运行的软件可免费退换。超过一个月, 如软件损坏, 可退回重新复制收费 IBM 每片10元, APPLE 每片7元, 另加邮费5元。

五、来信询问软件问题时请注明用户编号及软件编号, 并附上贴好邮票的标准回邮信封, 请在信封上写清地址和邮政编码。

六、汇款: 广州市石牌华南师范大学内电脑杂志社收。

注: 方括号内数字为片数, 圆圈内J指游戏杆, K指键盘。

## IBM 机软件

### 工具类

IT111 DOS外部命令集锦(二)(EXTENSIONS#2) (1) 在本软件库, IT 106 软件我们已介绍了DOS 外部命令集锦(-), 本期再为各位介绍一种性能丰富的软件。它能提供16个增强的DOS 外部命令, 其中包括有把一个或多个文件的若干行文本合并到一个或多个指定的文件中的MERGE 命令; 有把一个或多个文件从一个目录搬至同一盘的另一个目录, 移动之后的文件可以改名的MV 命令; 有可以选择打印文件若干行的PRNT 命令; 有修改子目录, 有修改子目录名的REMDIR 命令; 有修改一个或多个文件名的RENAM 命令; 有询问删除一个或多个文件的RM 命令; 有可以进行“流水作业”的编辑工具, 方便对文件增加修改和删除; 有允许用户直观进行多行数据操作的SELECT 命令; 有分析文件行数、字数等各种数据的STAT 命令; 有分析文件行数、字数等各种数据STAT 命令; 有指定搜寻单个字符的命令; 有灵巧的管道操作TEE 命令; 有修改文件某些字符的TXLAT 命令; 有合并包有重复行文件UMIQ 命令; 有查看或修改磁盘卷名为VOLM 命令和搜索具有相同的指定文件名的磁盘及目录。

IT112 PC实用程序集(UTILITIES ECETERA) (1) 本软件包括一系列的实用和程序, 其中有通讯用的程序BLATHER, 能够发送或接收文件, 用作远程终端, 修改通讯参数等功能; 这是一个功能很强通讯用程序, 窗口屏幕编辑示范程序DOORS, 用户可以从中学学习到如何建立另一个窗口并保证原窗口数据不丢失; 此外, 还有列目录程序COVER, 查找文件程序FINDFILE; 打印程序, PORE 该程序对DOS 的MORE 外部程序功能改进, 能够每打印55行暂停一次, 以便于用户操作。

IT113 BASIC窗口设计工具箱(Basic Windowing Tool box) [1] 30元/片 这是一个为方便BASIC 程序员写出更有技巧和趣味程序而提供的工具箱, 适用于使用标准解释型BASICT 和编译型BASIC 使用参考原稿。视窗关闭后原先复盖内容可以重视。总之, 这是一个饶有趣味, 功能较

强的BASIC 语言程序设计工具, 有兴趣读者不妨一试。

IT114 密码编制/译码系统(ENCODE/DECODE) (1) 60元/片 这是一个通过电子传递的文件编制密码和接收者译码而设计的软件, 以达到资料传递的保密性, 它包括三种档次密级: 普通级、机密级、绝密级的密码编制和解密程序。同时在编码过程中还采用了数据压缩技术, 以节省传递费用。

IT115 无线电爱好者工具箱(Ham Radio) (1) 本工具箱提供了一批实用软件, 其中包括传导线上电阻和电抗计算程序Smith, 电路分析系统NETWORK 等。NETWORK 程序可由用户将电路中各元件的数据输入指定文件, NETWORK 运行时读入指定文件, 并根据文件内容生成一个数组, 计算出电路输出端的频率特性, 并将电路分析结果输出屏幕或所设立的输出文件中。

IT116 电子表格用户指南(EXPRESS CALC) (2) 100元 这是一个比LOTUS-1, 2, 3 更易用的电子表格编制软件, 它适用于制定有关贷款、工资、月销售统计、税利统计及投资分析等表格, 能与PC-FILE, FILE EXPRESS 等用户数据库连接, 能从这些数据库中提取资料。简而言之, 这是适用一个商业与财务统计用的有力工具。本软件附有7页的中文使用说明。

IL46 PBASE关系数据库(PBASE V 1. 2) (1) 100元/片 PBASE 是一个以SQL 为基础的可编程关系数据库, 具有查询、插入、修改、删除、书写报告等数据处理的功能, 该数据库建立在表操作基础上的, 它本身提供的语言和命令可用于开发应用程序。该数据库具有较强的窗口的功能, 可以在屏幕的任意位置开窗口, 以显示有关信息和在窗口内执行PBASE 命令。本软件附有近万字的中文使用说明。

IL47 PROLOG系统(A, D, A PROLOG V1. 91) (1) 100元/片 这是一个由A, D, A(tomata Design Associates)开发的适用于IBM PC 类机型的环状结构的PROLOG, 被称为实现人工智能和决策的第五代开发工具, 本系统包括一个解释器(PDPROLOG.EXE 文件), 一个屏幕编辑器(PROLOGED.COM), 此外还有包含有自然语言分析器(Lou Schumacher)的压缩档案ATV, ARC, 有用于医学知识、病情诊断专家系统的压缩档案EXPERT, ARC 等一批实用软件。

IW16 字处理器W-ed (1) 这是一种小型、快速、直观的编辑器, 对于编辑小型程序特别方便。该软件有一个HELP 菜单, 便于用户学习使用。

IW17 文件打印格式工具ROFF4 (1) 60元/片 这是一个用C 语言开发专门用于文件打印格式化处理的工具, 它可以产生用户自定义的特殊符号或字体, 在打印机没有配置、反向滚动和退格功能时实现退格, 可以往上、下标, 具有较好的脚注的功能, 支持下划线、多重打印等功能, 其对打印控制比WORDSTAR 要强得多。

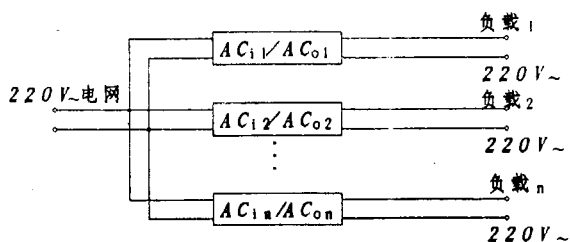
配备宏观预处理器功能也是本软件重要特色之一, 它可以为用户预示打印结果, 使用户避免许多因打印格式不符而造成的返工浪费。本软件附有长达8页的中文使用说明。

# 电子系统交流电源的配置

中国电子器材公司西南公司 陆玉新

众所周知,发电厂发出的电力总是有一定的功率限制的。在用电高峰超过其负荷能力时,电业局要采用降压供电,这就导致电网形成了欠压,在用电低峰时负载很轻,使电网电压升高造成过压;当电网功率因数太小时,供电网就要采用加入或断开并入电网的电容器来调节功率因数,这时会在电网中激起1KV以上的尖峰电压;电力通过高压输电线输送,而高压输电线是一个悬浮的不接地系统,在其周围发生闪电、汽车打火、无线电波发射、电弧辐射时,可能在输电线中引起千伏以上的尖峰电压;此外,用户本身大功率设备的不断接通与断开,特别是大功率马达接通瞬间需要很大的启动电流(持续时间可达几十秒),在输电线路内阻上产生很大压降,若正巧赶上几台大功率马达同时启动,电网电源线上几乎相当于短路,其影响相当于暂时停电;大功率的机械设备,其电源开关往往是由继电器控制的,开关触点很容易引起反弹,在反弹瞬间会在电网中引起600伏左右的尖峰电压。这些现象就是电网中产生电压瞬变(下陷、浪涌、尖峰等)的原因。这些噪声叠加在交流供电电网上沿电力线路传输,使用户在得到电力供应的同时,也受到电网上噪声干扰,并且在千万个用户之间引起相互干扰。理论分析和实测都表明,供电电网上的电源,既有持续时间较长的过压、欠压,过欠压以及停电等电源电压的慢变化,也有持续时间较短浪涌、下陷及尖峰等,这表明供电电网上的电源是极不干净的。

调查表明,灵敏电子设备、计算机和以数字处理为基础的电气装置所代表的负载量不大于市电总负载的0.01%。很显然,计算机制造厂、精密电子设备制造厂以及各行各业的电子设备用户、净化电源生产厂都不能左右或改变这一现状,市电供应部门也不会采取措施来防止影响。包括计算机在内的电子设备正常运转的电源噪声干扰。在一般情况下,配电导线的尺寸和接地技术要求是基于多数电气装置和重型电气设备的负载制定的,它并不考虑计算机等信息设备以毫秒级开关速度开关时所要求的低阻抗。这就迫使电子系统用户根据用电设备的要求和电网供电质量,组建与之相适应的电源系统。作为一个普遍可能接受和行之有效的系统,如下图所示,广大电子系统用



户通过一个个独立的,以同源输入和独立输出为主要标志的电源系统ACi/ACo来维持和保证用户局部范围内有一个纯净的电源环境。这种电源环境能有效地克服供电系统污染对电子设备正常运行的影响,以及避免不同用户之间的相互干扰。集中起来,该电源系统应该具有如下特点:

(1) 稳压范围宽。例如,即使电网电压在(120~300)伏,甚至更宽的范围内变化时,亦应保证其输出电压稳定在220伏;

(2) 抗干扰能力强。在电源开、关机或电网供、停电时,其输出无过冲,过渡特性好。这对于有效地保护计算机、精密电子仪器以及各种医疗设备,广播电视设备,程控电话等是至关重要的;

(3) 输入和输出相互隔离。这是避免各个独立电子系统运行相互干扰的有效方法;

(4) 可以带载启动。即不必严格遵照先开电源,后开电子设备的规定。这就可以避免电网无计划停电对电子系统安全性的威胁,它对设备维护人员的硬件知识要求较低;

(5) 寿命长。即要求其平均无故障时间(MTBF)长,这将有利于工作在无人值守的环境,它能长时间的连续不断地运行,最好能经受得起短路的考验。

目前国内外有很多电源生产厂家,其产品原理,所用器件各异,用户对象不同,性能价格相差悬殊。广大电子系统用户在选用交流稳压电源来改善自己电子设备的供电质量和供电环境时,应根据具体情况予以比较选择。根据对计算机、程控电话站、工业自动控制设备、激光照排系统、精密电子系统、高级音响、彩扩行业、医疗电子设备部门用户的调查和信息反馈,目前市场上销售的诸多交流稳压电源,其综合特性以广东省国营罗定无线电厂生产的“铁塔牌”交流参数稳压电源为最佳。交流参数稳压电源是通过周期性地改变谐振电路中的储能元件(电感器)的参数(电感量)将电源能量馈入系统并稳定电压的,它除了具有上述几个明显的特点之外,通常还设置有多种屏蔽和接地端,有利于避免或消除电子系统中各种用电设备在运行中的相互干扰,这就使得交流参数稳压电源在激烈的市场竞争中处于非常有利的地位。更特别值得注意的是,目前该类电源已经有了地区性的企业产品标准,并且为地区主管部门(广东省技术监督局)批准执行,该产品标准为电源生产专业厂家广东省国营罗定无线电厂制定,这就为该类电源的生产、检验提供了一个可靠的依据。该厂“铁塔牌”交流参数稳压器并于1990年获机电部优质产品称号,成为国内电源界罕见的部优产品。可以预言,该类电源将逐步取代614等型号的电子交流稳压器,成为90年代电子系统的主流电源。近几年来,全国各地用户,尤其是许多重要部门,如:海关、部队、深圳大亚湾核电站、西昌发射基地、十一届亚运会等,使用了广东国营罗定无线电厂生产“铁塔牌”交流参数稳压器后,都获得了显著效益,被用户称为“正宗产品”、“电脑保标”。这一事实也充分说明了参数稳压器正逐步成为电子系统的主流电源。■

# 广东省教育服务公司供应陕西计算机厂中华学习机

陕西计算机厂中华学习机全国中华学习机测试评比第一，全国总销量占有率超过 50%，是学校家庭用计算机的最佳选择。

陕西计算机厂中华学习机  
广州特约销售维修点（门市部）：广州市广仁路四号之二  
广东省教育服务公司

电话：352553

邮码：510036



## PC 1500 计算机新型 128K 电子记录模块开始供应

广州市袖珍计算机技术服务中心新近投产一种 GD-128K 电子记录模块，其特点是体积小（87×57×13mm）、容量大（128K 字节）、脱机保存时间长（半年以上）、价格低。该模块便于野外采集数据和存储程序，其除具有 E-BASIC80 多个指令外，又新增了计算机内存可靠性检测指令。只需 6 秒钟就可对用户存储区的内存进行检测，以保证数据采集的可靠性及计算数据的准确性。

该种电子记录模块还可增加万字汉卡功能。欢迎用户邮购。免收邮费，保用 2 年，终身保修。

GD-128	720 元	DG-96H（带汉卡）	730 元
GD-96	620 元	GD-64H（带汉卡）	620 元
GD-64	510 元	GD-32H（带汉卡）	490 元
GD-32	380 元	GD-128H（带汉卡）	820 元

### 广州袖珍计算机技术服务中心

地址：广州东风东路 745 号

电话：751025-235 邮政编码：510080

开户银行：广州建行天河支行环市东办

银行帐号：208-2612123-72