

E & C
1993
●一九九三年 ●总期第100期
7

電子

ISSN 1000 1077

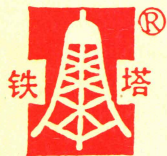
與 電腦



● ELECTRONICS AND COMPUTERS ●

理想的高抗干扰特宽稳压净化电源

铁塔牌 CWY 系列交流参数稳压器



抗干扰力强！抗雷击力强！

正宗产品！电脑必备！高精尖电器必备！

省优部优



国际金奖

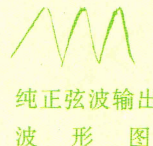
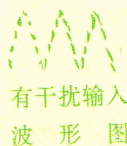
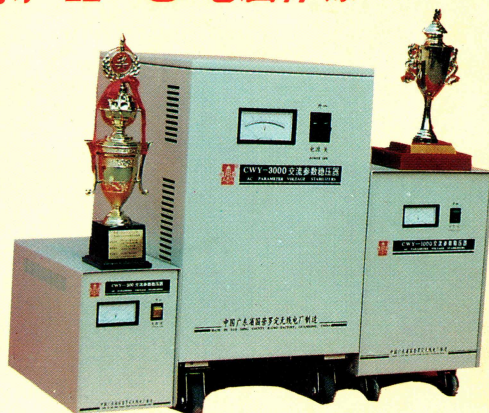
北京亚运会、大亚湾核电站和西昌卫星发射中心等单位都选用本产品！

★铁塔牌 CWY 系列交流参数稳压器：集隔离变压、稳压、抗干扰、净化功能于一体，具有稳压范围特宽（单相 120~300V，三相 260~460V）、精度高、纯正弦波输出、响应快、抗干扰力强、抗雷击力强、高可靠、寿命长、用途广等优异性能。

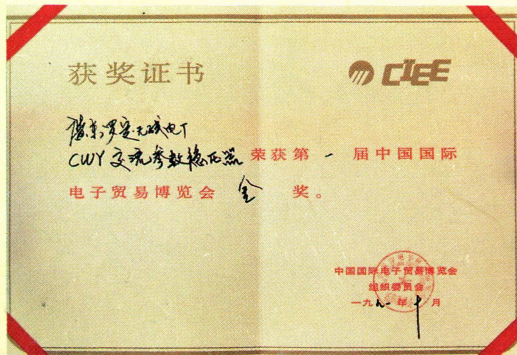
★铁塔牌 CWY 广泛适用于银行、邮电、通讯、科研、院校等企业事业部门的大中型计算机、微电脑、传真机、医疗器械、复印机、电脑打字机、高级音响等高精尖电器配套使用，效果极佳。

★铁塔 CWY 为您消除因电压不稳和各种电干扰以及雷击损坏设备的隐患。铁塔 CWY 为您提高产品质量以及提高产品与企业声誉起关键作用，并助您取得显著的经济效益。

铁塔电源 ● 稳如铁塔
正宗产品 ● 电脑保镖



- 规格：单相 0.3~30KVA，三相 1.5~100 KVA。
- 产品符合国际标准（机电部采标证字第 01517 号）
- 产品生产许可证（证书编号：XK—09—507—093）。



国内领先 国际水平

★连续七年被评为重合同守信用企业
省级先进企业

广东罗定市无线电厂制造

厂址：广东省罗定市逢源二巷 16 号 邮码：527200
电话：(区号 07666)727888、728888、723559(传真)
电挂：7193 广州联系电话：(区号 020) 4411450
北京联系电话：(区号 01)6065795 王东岳、张子光
开户帐号：广东罗定工商银行 203—00430004—060

● 国内代号：2-888 定价：1.60元

普及微電子與電腦知識，

傳達技術革命信息，

造就新型科技人才，

振興我國電子工業。

江澤民

一九八四年十一月

熱烈慶祝《電子與電腦》

雜誌的創刊，希望它在普及電

子和信息知識、培養年青一代

掌握電子技術上發揮積極的

作用。

李鵬

一九八〇年
十月廿三日

电子工业出版社广州科技公司

综合经营部邮购信息

本公司已于93年6月在穗正式开业。利用有关宣传媒介,通过邮购为读者服务,是公司的业务之一。

本公司自93年6月开始,将在《电子与电脑》杂志上介绍深圳、珠江三角洲及港台有关公司、厂家的一些电子产品,电脑配件,电脑软件以及电子版新书等,欢迎广大消费者惠顾垂询。



语言钟表系列

开关电源

邮购商品一览表

品名	规格	单价	邮费
DAKE 打印机共享器	2 × 1 (配 2 根电缆)	320 元/台	5 元/台
DAKE 打印机共享器散件(含外壳)	500 套以上起售	195 元/套	按实际费用结算
单片机应用系统开关电源	5V / 10A, +12V / 1A, -12V / 1A; 外形尺寸: L158 × W98 × H48mm ³	175 元/台	5 元/台
单片机应用系统开关电源	5V / 15A (LED 大屏幕显示用) 尺寸同上	175 元/台	5 元/台
单片机应用系统开关电源	5V / 10A, +12V / 1A, -5V / 2A 尺寸同上	175 元/台	5 元/台
软件狗 WATCH DOG	软件硬加密工具 体积: 60 × 50 × 15mm ³	200 元/台	5 元/台
袖珍固体留言录放机(不能放磁带)	存贮时间 20 秒	82 元/台	5 元/台
桌式语言报时钟 (带石英钟)	可自动每小时报时一次, 带闹	75 元/台	5 元/台
桌式语言报时钟	同上	60 元/台	5 元/台
盒式语言报时钟	同上	55 元/台	5 元/台
语言报时手表	同上	60 元/只	5 元/只

以上产品, 均为正式厂家产品, 如出现质量问题, 可在二个月内退换。

邮购注意事项:

- 1、来函及汇款要用挂号邮件寄来。
- 2、收货人的邮编、地址、姓名均要用正楷书写清楚。
- 3、以上商品均可批发, 批发价另议。

电子与电脑

一九九三年
总期第 100 期

目 录

- 综述 •
 - 祝贺“电子与电脑”第 100 期 吴鸿适(2)
 - 计算机与信息研究综述 贾洪卫(2)
- PC 用户 •
 - 弹出式菜单中屏幕滚动的具体实现
..... 董立平 李钢锲(4)
 - 普板型微机汉字系统及 2.13H 打印驱动程序的改进
..... 曲 峰(6)
 - DOS 内部命令的修改和扩充 姜金友(7)
 - Super PC/XT 附加存储器的使用 袁卫国(8)
 - 快速筛法求证素数的程序设计 王 存(11)
 - 文件(目录)的快速查找 刘炳文(13)
 - 巧用长城 0520EM 运行汉字 dBASE III 辛启亮(14)
 - C—WordStar 使用基础问与答 朱大公(15)
 - CCDOS 4.0 解密的方法 范恒钦(16)
 - 1741 病毒的检测及其清除 邹肇辉(17)
- 学习机之友 •
 - 谈 6502 软中断功能的开发应用 苏 华(19)
 - 汉字文稿打印程序 冯惠民(21)
 - ProDOS 系统内部结构剖析(续) 廖 凯(22)
- 语言讲座 •
 - FORH 语言讲座 丁志伟(24)
- 初、中级程序员软件水平考试辅导 •
 - C 语言中的字符串处理及指针变量在字符串处理中的应
用 李 宁(29)
- 学用单片机 •
 - DP—851 监控程序简介 罗明宽 车金相(39)
 - 实用汉字库芯片的制作 陈超波(43)
 - 51 单片机学习及应用装置 裴 巍(45)
- 电脑巧开发 •
 - 微机打印机共享器原理与设计 朱明程等(34)
 - IBM—PC 总线扩展技术和开放工作台 朱世鸿(47)
 - 烟台大学计算机应用开发中心让您的微电脑一展其音乐
才能 (45)
 - 微型计算机遥控键盘 顾正平(50)
- 电脑游戏机 •
 - FBASIC 语言编程特技的深入研究 于 春(53)
- 电脑通信 •
 - 用电流环路实现远程快速数据传输 袁学文(55)
- 传真机专题讲座(五) •
 - 传真机的安装和使用 张建军 张景生(57)
 - 电脑通信大家谈
MODEM 卡的使用——PC—PC 远程通信(二) ... (59)
- IC 电路应用 •
 - 模拟乘法器 IC 及使用方法(一) 李兰友(61)
- 读者联谊 •
 - DOS 应用技巧集锦 李 际(63)
 - 加速 WPS 的实现 邓 海、白奉礼(56)

机械电子工业部电子工业出版社主办

编辑、出版:《电子与电脑》编辑部
(北京 173 信箱 邮政编码:100036)
印刷:北京三二〇九厂
国内总发行:北京报刊发行局
国内统一刊号:CN11—2199
邮发代号:2—888
国外代号:M924

出版日期:每月 23 日

主编:王惠民 特约编审:苏子栋

责任编辑:施玉新

订购处:全国各地邮电局

国外总发行:中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证:京海工商广字 147 号

定价:1.60 元

祝贺“电子与电脑”第 100 期

吴鸿廷

“电子与电脑”是一份知识性、实用性、趣味性和消息性,兼顾普及和提高的科技刊物,主要面向初级和中级、专业和业余学习、从事电子计算机软、硬件设计、制作、测试、应用和维修的各社会阶层。八年多来到本期为止,这份刊物已发行到第 100 期,发行量逐年稳步增长。这是一个不小的成绩。

进入 90 年代以来,随着改革开放以及市场经济的兴起,我国已进入了电子与信息社会的时代。无论是在国民经济的工农业生产、水利交通、铁路航空、医药卫生、轻工纺织、地矿勘探、通信运输、电视广播、文化教育、核能利用、行政管理、金融财政、商务贸易以及国防领域的武器装备、侦察对抗和作战指挥等都离不开电子计算机的应用。特别是所有的高科技研究项目和高科技产品无一不采用电子与电脑相结合的技术手段,

很多产品几乎全部融电子与电脑于一体。在我国目前电子计算机的应用还不够普遍时,“电子与电脑”刊物的兴办对推广和普及电子计算机的应用作出了有益的贡献。

从创刊到第 100 期,标志着刊物从幼小、成长到壮大,必然经历过许多艰辛和克服过不少困难。这是一个有历史意义的里程碑。之所以取得这些成绩,离不开领导的支持、全体编辑出版和发行人员的努力,自然也还有广大读者、作者的关心、爱护和鼓励。

作为读者的一员,我衷心祝贺“电子与电脑”在这 100 期中所取得的可喜成就,同时也希望它在今后不断地总结经验、改进提高,进一步成长壮大,为我国社会主义建设事业作出更多贡献。

计算机与信息研究综论

湖北大学[430062] 贾洪卫

当今世界科学技术迅猛发展,学科知识交叉渗透、高度综合,信息交流与日剧增。信息的获得、选择、分类、筛选、组织、存储、传播、交流、利用已得到社会广泛的重视。信息同能量、物质成为人类社会三大资源之一,也是繁荣现代科学技术和发展社会生产力的三大支柱之一。任何人类创造物都是信息和物质、能量的结晶体。物质、能量奉献于人类的是材料和动力,而信息则为人类增进智慧和提高创造力。信息对经济的推动作用将逐渐大于能量和物质的作用,越来越成为生产力和经济成就的关键。信息在社会活动、生产实施、科学实验中将大大提高人类思维劳动的效率,导致劳动方式的巨大变化。人类社会正在进入一个信息化的时代,信息是人类社会进步的一种极为重要的经济资源。在当代商品经济高度发展而形成的国际、国内市场激烈的竞争态势中,以及在科学技术突飞猛进、迅速交替而形成世界复杂多变的政治经济大气候、大环境相互制约的抗衡态势中,信息已成为决定一个国家、企业经营成败和影响国民经济管理的关键因素,计算机技术和通信技术相结合,使人类社会的信息处理的传输出现一种全新的局面,它在人类走向信息时代的过程中,

信息资源的开发、加工、服务过程中起了先导的作用,信息是人类智慧的产物,特别是通过计算机技术处理的信息更是现代最高级、最复杂、具有高度逻辑性、预见性和选择行动的信息。

1. 计算机用于信息存储与检索

计算机与信息的关系主要是什么呢?首先从信息的基本特征看,信息具有 5 个基本特征:具有认识性、能成为一种资源、具有无限性和共享性、是可以传递和存储的、是可以检索和加工交换的。信息处理的最有效的手段是利用计算机和现代通信技术对信息的加工、存储、检索、传递、控制发挥出其高度的效率。实践证明,计算机有信息转换快、存储量大、处理速度快、重复性强和不易疲劳等优点,而计算机网络把这一优点进一步扩大,各终端用户可充分利用每台计算机所存储的大量信息以及由它所提供的各种服务项目,极大地提高工作效率和各种信息资源的利用率。在计算机可直接阅读的高密度磁盘上存储了大量的可供检索的信息,尤其是近年来研制成功的光盘存储器,存储密度高,是一种极有发展前途的存储设备。例如,引人注目的 CD-ROM 只读存储器是一种直径不到 5 英寸、厚

度不到 0.1 英寸的圆盘。可存七亿三千万字节的数据信息,瞬间可以随意读出符合目的的文字、图像信息。最近,世界上又研制成功超高密度光学存储材料,每平方厘米可存储几百亿字节信息。在信息量剧增的今天,用计算机控制的高密度存储信息通过现代通信技术,加快了信息的交流与利用,使信息源的范围大大扩展,并通过计算机网络传输到各个终端用户,具备了很高的信息资源共享性。

2. 计算机用于信息资源共享

在信息时代信息量爆炸性增长的情况下,如果不用计算机和现代通信设施且采用数据库形式来收集、存储和提供信息,则信息建设是不可想象的。数据库是现代信息产业的基本建设,是信息系统的核心。计算机用于信息资源共享,首先建立和开发统一标准的数据库,然后建设一个用于通信和信息处理的公共网络,其次开发信息传递和利用现代最新技术。以计算机为代表的高技术产业在世界范围内迅速兴起和发展,信息这一资源就显得更为重要了。这是因为在现代新兴产业发展中,其功能系统的各个环节都必须依赖于信息的准确性、快速性和灵敏性。一个国家拥有多少自己的数据库、能用数据库提供多少服务,是其现代化水平的标志之一。美国是数据库建设起步最早发展最快的国家,目前全世界商用数据库的三分之二是美国生产的。据统计,美国每年应用计算机来处理的各种信息量相当于 4000 亿人年,是美国现有人员工作信息量的 2000 多倍。在西方一些发达国家,以计算机处理信息为主的社会经济效益来看,在整个国民经济增长率中要占 40—60%。在日本,其工业所以能在高技术领域里向美国提出挑战,并取得引人瞩目的成就,其中一个重要因素就在于他们对信息工作的重视,并努力设法采用以计算机技术为代表的各种现代化信息处理手段和重视信息资源的开发,来加快国际间最新成果信息源的开发和利用。计算机是一种强有力的信息资源处理工具,它极大地提高人类处理信息的能力,掌握了它就会获得科技进步的主动权。

3. 计算机与信息科学

随着高度信息化社会的进展,开发信息处理技术方向也开始发生转变,由重视电子计算机、通信网络、新闻传播媒介等硬件转变为重视人工智能、信息检索和帮助人们做出判断的软件,形成了信息科学。信息科学以信息论和控制论为理论基础,与电子学、计算机和自动化技术以及生物学、数学和物理学等学科相结合,是一门边缘科学。它着重研究机器、生物和人类对于各种信息的获取、变换、传输、处理、利用和控制的一般规律,目前正处于发展阶段。信息科学在与计算机结合方面取得了一些成果,尤其是第五代计算机的出现与模糊数学理论结合建立了信息识别理论,这个领域的工作对于信息科学的发展必将产生巨大的影响。第五代计算机系统将具有以很高水平来解决问题的能力,将是知识信息处理系统。在这些系统中,智能水平将会大大改进到接近人类的水平。所谓知识信息处理,实际上

是计算机科学的分支领域——人工智能,这类问题的求解系统通常叫做专家系统,它是九十年代信息处理的一个主要领域,为这类问题设计的计算机,是第五代计划的主要课题。信息科学是研究信息现象及其规律的科学。它的主体结构是包含了信息论、控制论和系统论三者。而人工智能理论是这三者的综合应用。第五代计算机系统最重要的特征是人工智能特征,信息科学将在这个计算机系统上迅猛发展。

4. 计算机技术与信息技术

现代的信息技术花样繁多,数不胜数,渗透在生产、科学研究、国防以至社会生活和家庭生活的各个方面,真是令人眼花缭乱,但是,不管信息技术是怎样千姿百态、无可计数,我们仍然可以给它一个十分明确的定义:凡是可以扩展人的信息功能的技术,都是信息技术。比如,计算机技术是一种信息技术,因为它可以扩展人的处理信息等功能。信息技术最重要的代表是传感技术、通信技术和计算机技术。传感技术主要包括信息识别、检测、提取、变换以及某些信息处理技术,它是人们信息感觉器官功能的扩展和延长。通信技术大体上包含信息检测、变换、处理、传递、存储以及某些信息控制与调节技术,它是人的信息神经系统功能的扩展和延长。计算机技术主要包括信息存储、检索、处理、分析、决策以及控制等技术,它是人的思维器官功能的扩展和延长。传感、通信、计算机技术代表了信息技术的主要方面,三者相辅相成,尤其是通信与计算机技术两者性质更为接近。传感技术产生信息源泉。通信技术使信息流通,是信息社会的生命线。计算机技术与传感、通信技术三者互相渗透、互相合流在一起形成功能多样、威力强大的计算机智能综合信息网络系统,形成了信息科学技术突飞猛进,势如破竹的壮观局面,引起和促进了整个科学技术领域的革命。

5. 计算机与信息革命

计算机的发明和应用,使人们在信息传输、接受、处理方式、方法和能力方面产生了深刻印象。计算机可以代替人脑的部分功能,扩展人的智力,使人们得到快速、准确和完整的信息,在人类的生产,生活和科学实验中引起一场新的影响极为深远的信息革命。

信息革命以信息科学作为理论基础和技术指南,它的直接目标与必然结果,是要扩展人类的信息功能,包括获取、传递和处理信息的功能,特别是要扩展处理信息的功能,这相当于延长人类的感覺器官、神经系统和思维器官。信息革命带来信息工业的大发展,尤其是计算机工业的迅速发展。美国硅谷的创立和发展就是信息革命的胜利果实。

计算机是人类获取信息、传输和处理信息的好帮手。用计算机来管理农业、工业生产使经营管理自动化,使生产者能够做出最佳决策,导致农业、工业生产方式的根本性变革。信息革命还带来社会产业结构的变化。由于计算机的使用,提供信息服务的行业 and 方式发生了变化。例如,在新闻业、编辑、排版、印刷以及新闻的获取和播发都在向计算机化方向发展,在金融业,

计算机化的银行已经实现记帐、转帐、存款、取款的自动化,并且提供保险、不动产、证券、信用卡和数据处理等计算机服务项目;在经济领域,提供商品和金融行情等经济信息通过计算机终端屏幕快速地了解世界经济信息;在家庭,通过计算机网络上的终端设备,人们可以坐在家办公、听课、查找资料、撰写论文和玩计算机游戏。人们还可以用计算机自动管理微波炉、电子洗碗机、电热器、空调器等家用电器把自己从繁重的家务

劳动中解放出来。综上所述,信息革命将使人类社会发生深刻的变化,计算机在信息革命的发展中扮演了极重要的角色,信息科学技术革命将大大改变整个科学技术的内容、结构、体系和方向,改变科学的思维方式和发 展图景,将有力地推动科学技术本身突飞猛进的向前发展而且也将对工业、农业、国防、交通、文教、卫生、商业、贸易、经济、管理、社会以至家庭生活各方面产生强烈的冲击。

弹出式菜单中屏幕滚动的具体实现

郑州纺织工学院计算中心(450007) 董立平
郑州纺织机械厂生产处 李钢铨

具有立体投影的弹出式菜单是目前国内外许多流行的优秀软件所采用的菜单界面方式。如 PCSHELL、C 等集成软件,它以“→”、“←”、“↑”、“↓”四个光标键和回车键进行选择,具有新颖的窗口画面和良好的用户界面,其最大的优点是:窗口位置、大小可以任意,光棒定位,清晰美观,选择不易出错。

本文给出菜单项目用醒目的光棒在具有立体投影的弹出式菜单中左右、上下滚动屏幕,用 PgUp、PgDn 键翻页显示的实现方法。源程序用 Turbo C 2.0 编译,在 AST286、386 上运行通过。

1. 实现目标

熟悉计算机的用户都知道,在目前比较流行的 PCSHELL、Turbo 等系列的集成软件中,其弹出式菜单中有时由于菜单项目较多,在规定的菜单窗口中不能一屏显示完菜单的全部条目,此时只好用上下滚动屏幕显示菜单以实现菜单项目显示的“连续性”和“双向性”。本文笔者模仿了 Turbo C 集成开发环境中按 F3 弹出一个显示指定目录下的文件的菜单,等待用户按上、下、左、右光标键和 PgUp/PgDn 键将醒目的光棒移到要选择的文件处,然后按回车即可进入下一步要处理的工作。当要显示的菜单项目数大于窗口的显示行数时,若当前醒目项在窗口的最后一行时,按“↓”键后,则把要显示的下一行菜单条目都显示在窗口的最后一行上,即实现屏幕上滚显示;若当前醒目项在窗口可显示的第一行且窗口以外还有待选择的菜单项时,按“↑”键后,则把当前醒目项所在行之外的上一行菜单条目显示在窗口的第一行,即实现屏幕下滚显示;同样,若按左右光标键则使醒目项在窗口菜单中左右乃至上下行移动,按 PgUp/PgDn 键则可实现屏幕的上下翻页显示。

2. 实现方法

首先,要编写显示弹出式菜单项目的字符串显示函数 write—string()。为提高显示速度,本文采用直接向显示缓冲区写字符的方式来向屏幕快速地显示菜单

项,实践证明,这种方法使屏幕的显示在视觉上没有停留感,真正体现了菜单的弹出式效果。

其次,要实现菜单项在窗口中向上或向下的滚动功能,为此,要利用 ROM BIOS 的显示器中断 10H 服务的 06H 号或 07H 号功能调用来确定屏幕上的一个矩形文本窗口,并且使这个窗口向上或向下滚动一行或多行,其中,中断服务 06H 号功能是在窗口底部插入空行(07H 功能是在顶部插入),而窗口的顶部行被卷上且消失(07H 是底部的行消失)。至于窗口中菜单项目用醒目的光棒左右、上下乃至翻页选择的具体实现,其关键是要记住当前光棒显示的菜单项在窗口中所处的位置以及它在整个菜单数组 menu[] 中的下标,然后以它为中心来实现光棒在整个菜单窗口中的正确移动,其实现的具体细节由于要考虑的方面比较多,读者可以参看源程序去琢磨。

再次,由于本弹出式菜单是用菜单数组 menu[] 来存放窗口中的菜单项,故用户只需将菜单内容写入菜单数组,然后直接调用弹出式菜单函数 menu—display() 即可。本文是用当前工作目录下的文件名来初始化菜单数组,从而真正达到模拟 Turbo C 集成工作环境下按 F3 所弹出的文件菜单,菜单数组的初始化是用 C 语言的库函数 findfirst()、findnext() 组合起来对磁盘目录搜索而完成的。因而,本程序不仅实现了弹出式菜单中屏幕菜单项用醒目的光棒左右、上下移动乃至翻页显示的功能,而且还同时实现了当前工作目录下文件目录的列表显示功能,程序运行方式如同 DOS 系统下的 DIR 命令一致,还对具有只读、隐含、系统等属性的文件同样具有列表显示功能。

最后,谈一下具有立体投影效果的窗口技术设计的基本原理:先将设定的窗口投影颜色,在投影区内进行清屏处理;再在以上位置稍加错位后,将按设定窗口颜色进行清屏的结果叠加在投影面积上,这样便建立了一个具有立体投影效果的窗口。Turbo C 2.0 提供了大量的屏幕管理库函数,如 window()、textcolor()、

textbackground()等,为西文彩色窗口菜单的设计提供了基本条件,读者可参看源程序得知。

附源程序清单如下:

FILENAME:tc-menu.C

```
#include<dos.h>
#include<string.h>
#include<dir.h>
#define MAX1 200 /* 菜单最大项目数 */
#define ROW 4 /* 每行菜单项目数 */
#define COL 9 /* 每列菜单项目数 */
#define LEN 14 /* 菜单项长度 */

void write_string(int x,int y,int attrib,char *s)
/* 显示字符串;x,y:显示行列;attrib:显示属性;s:字符串 */
{ char far *v;
v=(char far *)6xb8000000;
v+=(x*160)+y*2;
while(*s){*v++=*s++;*v++=attrib;}
}

void up_down(int num,int top,int left,int bottom,int right)
/* 使窗口菜单项上下滚动一行;num:调用功能号 */
/* 其余参数指明窗口滚动的范围 */
{ union REGS regs;
regs.h.ah=num;regs.h.al=1;regs.h.bh=0;
regs.h.ch=top;regs.h.cl=left;
regs.h.dh=bottom;regs.h.dl=right;
int86(0x10,&regs,&regs);
}

menu_display(unsigned char menu[][13],int max,int top,
int left)
/* 弹出式菜单函数;menu:菜单数组;max:菜单项数;top,
left:菜单区左上角坐标 */
{ union REGS r,regs;
register int i=0,j=0;
int v1,v2,bottom,right,mark=1;
int s0,s1,w=0,a=0,b=0;
static int coord_x=0,coord_y=0;
char str[65];
window(1,1,80,25);textbackground(1);clrscr();
right=left+ROW*LEN+2;bottom=top+COL+3;
/* 选择菜单窗口的投影窗口 */
window(left+2,top+1,right+2,bottom+1);
textbackground(0);clrscr();
/* -----显示菜单边框----- */
v1=bottom-top;v2=right-left;strcpy(str,"\xda");
for(i=1;i<v2;i++) strcat(str,"\xc4");strcat(str,"\
xbf");
write_string(top-1,left-1,0x70,str);strcpy(str,"\
xb3");
for(i=1;i<v1;i++)
{ write_string(top+i-1,left-1,0x70,str);
write_string(top+i-1,right-1,0x70,str);}
strcpy(str,"\xc0");
```

```
for(i=1;i<v2;i++) strcat(str,"\xc4");strcat(str,"\
xd9");
write_string(bottom-1,left-1,0x70,str);
/* -----弹出式菜单的主体----- */
for(;;){/* -----显示一页----- */
if(mark)
{ window(left+1,top+1,right-1,bottom-1);
textbackground(7);clrscr();
if(b)s0=++w;else s0=w;
for(i=0;i<COL;i++)
if(max>=s0)
write_string(i+1+top,j*LEN+left+1,0x70,menu[s0+
+]);
else {coord_x=0;coord_y=0;}
mark=0;
}
/* -----菜单醒目项的控制----- */
switch(a){
case 1:if(w>max)w--;
else w=w+coord_y-ROW;a=0;break;
case 2:if(<w+ROW)<=max)w=w+ROW;a=0;break;
case 3:if(b=0;
else ++w;a=0;break;
case 4:w=w+coord_x*ROW+coord_y;a=0;break;
default;break;
}
/* -----显示菜单醒目项----- */
write_string(coord_x+1+top,coord_y*LEN+left+1,
0x0b,menu[w]);
r.h.ah=0;(int86(0x16,&r,&r);
write_string(coord_x+1+top,coord_y*LEN+left+1,
0x70,menu[w]);
if(r.h.ah=28||r.h.ah=1)break;/* 回车或 Esc 退出 */
/* 选择 */
switch(r.h.ah){
case 72:if((w-ROW)>=0&&coord_x==0)
{ a=1;
up_down(7,top+1,left+1,bottom-3,right-2);
window(left+1,top+1,right-1,top+2);
textbackground(7);clrscr();
w=w-coord_y-ROW;
for(i=0;i<ROW;i++)
write_string(1+top,i*LEN+left+1,0x70,menu[w+
+]);
}
else if(coord_x!=0){ coord_x--;w=w-ROW;}break;
/* * */
case 80:if((w+ROW-coord_y)<=max&&coord_x==
COL-1)
{a=1;
up_down(6,top+1,left+1,bottom-2,right-2);
window(left+1,bottom-2,right-1,bottom-1);
textbackground(7);clrscr();
s1=w;w=w-coord_y+ROW;
for(i=0;i<ROW&&w<=max;i++)
write_string(COL+top,i*LEN+left+1,0x70,menu[w+
+]);
```



```

+]);
if (w>max)
if ((s1+ROW)<=max){w=s1+ROW;a=0;}
else coord_y=--i;
}
else if ((w+ROW)<=max){coord_x++;a=2;}
else if (w==max)
{coord_x=0;coord_y=0;w=0;mark=1;}break; /* ↓ */
case 75:if (coord_y==0&&coord_x==0) break;
coord_y--;w--;
if (coord_y<0&&coord_x>0)
{coord_x--;coord_y=ROW-1;}break; /* ← */
case 77:if (coord_x==COL-1&&coord_y==ROW-1&&(w+1)<=max)
{a=3;b=1;mark=1;coord_x=coord_y=0;}
else if (w<max)
{coord_y++;a=3;}
if (coord_y>(ROM-1)){coord_x++;coord_y=0;}
} break;
case 73:if ((w-coord_x*ROW-coord_y-1)>0)
{w=w-coord_x*ROW-coord_y-1;a=4}
if ((w+ROW-COL*ROW)>=0) w=w+ROW-COL*
ROW+1;
else w=0; mark=1; }break; /* PgUP */

```

```

case 81:if (max>=(w+COL*ROW-coord_x*ROW-
coord_y))
{w=w+COL*ROW-coord_x*ROW-coord_y-ROW;
a=4;mark=1; }break; /* PgDn */
}
}
/* 主程序 */
main(int argc,char * argv[])
{int j=0,done;
unsigned char menu[MAX1][13];
struct fblk f;
if (argc==1) strcpy(argv[1],"*.*");
/* 用当前工作目录下的文件名初始化菜单数组 */
done=findfirst(argv[1],&f,0xff);
while (! done)
{if ((f.ff_attrib&0x10)! =0x10)
strcpy(menu[j++],f.ff_name);
done=findnext(&f);
}
j--;menu_display(menu,j,6,11);/* 调用菜单函数,显示
在6行11列 */
clrscr();
}

```

普及型微机汉字系统及 2.13H 打印驱动程序的改进

曲 峰

一、普及型计算机汉字系统的改进

在 93 年第 2 期的《电子与电脑》上,张济生先生(《普及型计算机汉字使用问答》的作者)为广大 PC 机用户提供了两套系统,使一些无硬盘的用户能够使用功能较全的汉字系统。

本人在普及机汉字系统的基础上作了一些改进。对于 286 主机(具有 1M 内存):第一,对 DOS 进行了升级。对 DOS 系统分别采用 DOS 3.31 和 DOS V5.0, DOS 3.31 版非常适用于只有一个驱动器的用户;DOS 5.0 版由于 DOS 占用了 HMA,所以虚盘上的空间很小,只能运行一些小程序,如 BASIC 等,但其主存用户空间比 DOS 3.31 版节省近 45K 的内存空间。此版本非常适用有二个驱动器的用户,单个驱动器的用户也可使用。第二,在原基础上增加了显示模块,可支持 CGA、EGA、VGA、COLOR 400 和单色显示器,用户可根据自己的情况进行选择,均用菜单显示,使用非常方便。对于具有双频卡的用户,可用 MODE MONO 或 MODE CO80 命令设置单显或 CGA 模式。第三,按照功能的不同,用菜单和子目录进行了分类,使用户一目了然。第四,对原程序进行了改进和压缩,使程序执行更快,而且节省了磁盘的空间。第五,增加一些实用程序,如 PC Tools、EDIT 和 MI 等。

对于只有 640K 内存的用户,可使用 DOS 3.31 版的压缩汉字系统,支持 CGA、EGA、VGA 和单显。

如果系统设置不适合用户使用环境,可相应地修改 AUTOEXEC.BAT 文件。

经过以上改进,汉字系统运行得更快,更省内存(DOS 5.0 版本最大可供用户使用的空间为 577K),均可运行如 FoxBASE 等大型软件。

用户在使用 WordStar 和 CCED 时,应根据自己所使用的显示器,来修改其显示模式和行数。WordStar 用 CWSET.EXE 进行设置;对于 CCED,需进入 CCED 后按 Shift+F7 进行设置。以上两个软件,缺省为 CGA 显示模式和 15 行内容显示。

系统详细使用说明请参考《2.13 系列汉字系统用户手册》及有关资料,这里不再作介绍。

二、修改 2.13H 打印驱动程序的一处错误

2.13H 汉字系统的打印功能很强,它可以在许多汉字系统下使用,但在驱动 LQ1500 系列打印机时,出现不能双向打印的问题。本人通过分析 PRTA.COM 程序,发现是原程序有一处错误。LQ1500 打印子程序是与 AR3240 子程序共用的,只是在单/双向控制码有些区别。由于原程序的错误,使 LQ1500 打印子程序用的是 AR3240 单/双向控制码。见下面的程序:

LQ1500(AR3240)单/双向控制子程序

```

1F66:0DCB CMP Byte Ptr [0A42],FF
1F66:0DD0 JNZ 0DD7
1F66:0DD2 MOV [0A42],AL
1F66:0DD5 MOV AL,3E
1F66:0DD7 CMP AL,3E ;是单向控制符 '>' 吗?
1F66:0DD9 JZ 0DDF ;是,转
1F66:0ddb MOV AL,00 ;双向,AR3240 为 '02'
1F66:0DDD JMP 0DE1
1F66:0DDF MOV AL,01 ;单向,AR3240 为 '00'
1F66:0DE1 PUSH AX
1F66:0DE2 MOV AL,1B ;'ESC'
1F66:0DE4 CALL 0550
1F66:0DE7 MOV AL,55 ;'U'
1F66:0DE9 CALL 0550
1F66:0DEC POP AX

```

LQ1500(AR3240)打印机初始化子程序

```

1F66:10EA CMP AL,30 ;是 '0' 号 AR3240 吗?
1F66:10EC JNZ 110E ;不是,转
1F66:10EE MOV Word Ptr [0DF9],5241
1F66:10F4 MOV Word Ptr [0DFB],3233
1F66:10FA MOV Word Ptr [0DFD],3034

```

```

1F66:1100 MOV AX,0200 ;设置单双向控制码
1F66:1103 MOV [0DDC],AH ;设置双向控制码
1F66:1107 MOV [0DE0],AL ;设置单向控制码
1F66:110A MOV AL,05
1F66:110C JMP 1133
1F66:110E CMP AL,35 ;是 '5' 号 LQ1500 码?
1F66:1110 JNZ 1129 ;不是,转
1F66:1112 MOV Word Ptr [0DF9],514C
1F66:1118 MOV Word Ptr [0DFB],3531
1F66:111E MOV Word Ptr [0DFD],3030
1F66:1124 MOV AX,0001 ;设置单/双向控制码
1F66:1127 JMP 1100 ;应为 JMP 1103
1F66:1129 CMP AL,31 ;是 '1' 号 P1351 码?

```

修改时只要用 DEBUG 将 PRTA.COM 调入内存,用 A 命令将 1127 处的 JMP 1100 改为 JMP 1103,存盘即可。

注:用户如果需要此汉字系统,请与电子工业出版社社办闫莉联系,每套 2 张,定价 25 元,邮购请寄北京 173 信箱电子工业出版社,社办闫莉收,邮政编码:100036,电话:821.4062

DOS 内部命令的修改和扩充

合肥矿山机器厂(230011) 姜金友

DOS 由以下四个主要部分组成:引导块;IBM-BIO.COM;IBMDOS.COM;COMMAND.COM,其中 COMMAND.COM 负责接受键盘命令,并执行相应的命令子程序;如果是象 DIR、COPY、RENAME 等内部命令,就直接由驻留在内存的代码执行;如果是外部命令,则调用相应的可执行程序执行。DOS 内部命令是由 COMMAND.COM 初始完成并驻留内存的,其优先级最高,利用 DEBUG.COM 将 COMMAND.COM 调入内存,可以找到一个 DOS 内部命令表。每个命令以 ASCII 文本形式出现,后随一个四字节的跳转指针指向其代码。该命令表的位置随 DOS 版本不同而不同,对于 DOS 3.30 版本,命令表在偏移 5448H 处。

对于 COPY 命令,其内存入口地址在 CS:2A15H,若内存为 640K,则 CS=9B2AH,只要从 9B2A:2A15H 反汇编就可得到 COPY 命令的内部代码,这段代码对应于 COMMAND.COM 的位置在 4075H 处。

从以上分析可知,修改 DOS 内部命令只要修改 COMMAND.COM 即可,有以下两种方法:

1. 直接修改每个命令的 ASCII 字符串,其余不变。例如将 COPY 命令改为 DUPL,可按以下步骤进行:

C>DEBUG COMMAND.COM

—E54AB "DUPL"

—W

—Q

运行 COMMAND.COM 以后再打入 COPY 命令将不会执行,而键入 DUPL 则会执行原来的 COPY 命令。须注意的是命令表的 ASCII 字符串一定要大写,如果要使某个命令不能执行,只要简单地将其改为小写字符串即可。例如为了防止删除文件,可将 "DEL" 命令改为 "del"。

2. 修改内部代码。此时必须找到 COMMAND.COM 里对应的命令代码位置,下面将几种不同 DOS 版本的 COPY 命令代码位置列出(在栏为 DOS 版本,中栏为 COPY 代码在 COMMAND.COM 内的位置,右栏为 COPY 代码在 640K 内存中位置):

2.00	2B8CH	9CB0:1ACCH
3.10	39A1H	9B79:2661H
3.20	3BB1H	9B6E:2701H
3.30	4075H	9B2A:2A15H

通过 COPY 代码位置就可算出其它命令代码位置,例如 DOS 3.30 的 CHCP 命令,其内存入口在 9B2A:15D2H,对应的 COMMAND.COM 内位置=

4075H+(15D2H-2A15H)=2C32H。

下面就是将 DOS 3.30 的 CHCP 命令改为隐藏文件命令 HIDN 的过程:

```

C)DEBUG COMMAND.COM
-E5473 "HIDN"           ;修改 CHCP 为 HIDN
-A2C32
××××:2C32  PUSH  CS
××××:2C33  POP   DS
××××:2C34  PUSH  CS
××××:2C35  POP   ES
××××:2C36  JMP   2C94
××××:2C38  NOP
-A2C94
××××:2C94  MOV   DX,15ED
××××:2C97  MOV   AH,09      ;显示提示信息
××××:2C99  INT   21
××××:2C9B  MOV   AH,0A;输入须隐藏的文件名
××××:2C9D  MOV   DX,15D9
××××:2CA0  MOV   BX,DX
××××:2CA2  MOV   AL,14
××××:2CA4  MOV   [BX],AL
××××:2CA6  INT   21
××××:2CA8  INC   BX
××××:2CA9  MOV   AL,[BX]
××××:2CAB  MOV   AH,00
××××:2CAD  ADD   DX,AX
××××:2CAF  INC   DX
××××:2CB0  INC   DX
××××:2CB1  MOV   BX,DX
××××:2CB3  MOV   AL,00
××××:2CB5  MOV   [BX],AL
××××:2CB7  MOV   AH,43     ;修改文件属性
××××:2CB9  MOV   DX,15D9
××××:2CBC  INC   DX
××××:2CBD  INC   DX
××××:2CBE  MOV   AL,01
××××:2CC0  MOV   CX,0022
××××:2CC3  INT   21
××××:2CC5  MOV   BH,FF
××××:2CC7  CMP   AH,BH
××××:2CC9  JZ    2CD2
××××:2CCB  MOV   DX,1613    ;显示错误信息
××××:2CCE  MOV   AH,09
××××:2CD0  INT   21
××××:2CD2  MOV   BX,15D9;重新恢复 DOS
                        暂存区,以免再次
××××:2CD5  MOV   AL,00;装入 COMMAND.COM
××××:2CD7  MOV   [BX],AL
××××:2CD9  INC   BX
××××:2CDA  CMP   BX,15ED
××××:2CDE  JNZ   2CD7
××××:2CE0  MOV   AH,4C     ;结束
××××:2CE2  INT   21
-F2C39 2C4C 00
-E2C4D 0A 0D "Please input file or directry name: $"
-E2C73 0A 0D "File or directry can't accKSess! $"

```

-W

-Q

再次加载 COMMAND.COM 后,键入 HIDH 命令即可隐藏指定文件或子目录。这就意味着 DOS 内部命令得到了扩充。读者还可以对一些不常用的 DOS 内部命令按上法进行扩充,须注意的是扩充部分的代码不要覆盖了其它命令代码,以免影响这些命令的执行。如 DOS 3.30 的 CHCP 命令代码从 2C32H 到 2D0AH,扩充的命令代码也必须在这个范围内,否则就要影响 SET(内存入口地址:CS:16AAH)等命令的执行。

Super PC/XT 附加存储器的使用

新华社重庆支社(630020) 袁卫国

近一、二年,随着市场上 PC 机价格大幅度下降,许多价廉物美的 PC 机已进入家庭,其中较为常见的是小机箱的 Super PC/XT,以及类似的产品(如“新潮”PC/XT)。这类 PC 机基本配置较齐全,一般都有 640K 内存,2 台 5 英寸软驱,双频双显。特别是 200W 开关电源预留了硬盘、3 英寸软驱的插头,及其在机箱中的位置,使其很容易扩展成为一台性能较强的 PC/XT 机,而价格却不到 3000 元,故深受一般家庭欢迎。

但是,这类机器一般都存在着软、硬件资料缺乏的问题,使其功能的充分发挥受到一定限制。例如,这些新型的 PC 机在硬件结构上与以前的 IBM PC 有很大的不同,突出表现在机内的 RAM 上。该机一般主板上插有 4 片 514256(512K)、4 片 51464(128K),除此以外,尚有一些 RAM 插座,主板上总共可插 1MB 的 RAM。如果花上一百多元钱,再购买 4 片 514256,将内存扩展为 1MB,对于一般无硬盘的家用 PC 来说,是必要的。方法是:将 4 片 51464 拔下,然后将 4 个 514256 空插座插上 RAM 芯片(电路板上各 RAM 型号标记),最后将系统开关 SW3、SW4 拨到 OFF 位置即可。此时开机自检,显示 1024K OK,说明安装成功。

但此时如果用 VDISK.SYS 的 /E 开关在增加的 384K 内存中建立虚拟盘,你定会失望:系统根本不承认有扩展内存。据其随机资料称:A0000H—FFFFFH 可作页面存储器地址,即支持 EMS 规范,但用 EMS.SYS 启动、RAMDRIVE.SYS 的 /A 开关建虚盘,仍然不能工作。

VDISK.SYS 的 /E 开关所支持的是只有在 80286 CPU 的保护方式下可以访问的扩展存储器(Extended Memory),故以 8088(或 NECV20)为 CPU 的 PC 机理理所当然地不可能使用它。而 RAMDRIVE.SYS 的 /A 开关所支持的是 EMS 规范的扩充存储器(Expanded Memory),许多 PC 机的内存扩展板都支持它。令人遗

憾的是, Super PC/XT 增加的 384K 内存既不同于前者, 也不同于后者, 所以无法用现成的软件来利用它。

通过对该机 BIOS 的分析, 发现这增加的 384K 内存是采用类似于中华学习机那样的存储体切换技术来实现的, 姑且称之为附加存储器。附加存储器的地址是 2000:0000—7000:FFFF 共 6 个 64K 体。切换方法是:

```

切换到附加内存: 切换到主存:
MOV AL,1        MOV AL,0
OUT E0,AL       OUT E0,AL

```

有两种办法可以使附加内存用于虚盘: 一是专门编写一个 VDISK 程序; 再者就是模仿扩展存储器或扩充存储器。显然, 前者较复杂, 且附加内存只能作虚盘, 利用率太低; 而后者实现起来较容易, 且除了作虚盘外, 还可支持其它使用扩展内存的软件。故笔者采取了后一种办法中的模仿扩展存储器的方法。

AT 机的扩展存储器是通过 BIOS 的 15H 号中断来管理的, 其中关于扩展存储器的最重要的子功能是功能调用 87H——成块传送和功能调用 88H——取扩展内存容量。VDISK.SYS 以及其它许多使用扩展内存的程序都要用到它们。所以, 将 Super PC/XT 的 BIOS 的 15H 号中断扩充上述两个子功能, 就能够实现对扩展存储器的模仿。

88H 子功能的扩充非常容易, 只要将附加存储器的大小测出(以 KB 为单位), 用 AX 寄存器返回即可。

87H 子功能是在标准存储器(0—640K)和扩展存储器(1M—16M)之间成块传递数据。在实时地址(即类似 8088)调用此功能时, 必须通过建立描述符表来辨认存储区域的源区域和目的区域地址(详细情况可参考电子工业出版社出版的《DOS/BIOS 使用详解》一书)。根据这张描述符表, 我们可以这样来处理: 若地址小于 A0000H, 就在主存储器内存取; 若地址大于 FFFFFH(1M 字节), 就换算成相应地址(如 100000 对应于 2000:0000, 110000 对应于 3000:0000 等等), 在附加存储器内存取(即对应于扩展存储器)。

根据上述原理, 笔者编写了一个设备驱动程序: EXTMEM.SYS, 其源程序为 EXTMEM.ASM(见程序清单)。这是一个标准的设备驱动程序, 其规则可参见电子工业出版社出版的《IBM PC 编程指南》一书。该程序的汇编与一般的 EXE.COM 文件完全一样, 例如可按以下步骤汇编、链接:

```

MASM EXTMEM.ASM,EXTMEM;
LINK EXTMEM,EXTMEM
EXE2BIN EXTMEM EXTMEM.SYS

```

其中第三步可有可无, 即 EXTMEM.SYS 既可是 EXE 文件, 也可是 BIN 文件。

现在, 在 CONFIG.SYS 中加上:

```

DEVICE=EXTMEM.SYS
DEVICE=VDISK.SYS 384 /E

```

然后启动 DOS 系统, 你就会发现你有一个 384K 的且 not 占主内存的虚盘了! 为了证实这一点, 你用 PC Tools 的 system Info 功能查看系统配置便可发现具有

扩展存储器 384K。

笔者用 CCDOS 2.13H 将 16 点阵字库放在虚盘(注意: 对无硬盘的机器要将 FILE3 D2 改成 FILE3 C2)的功能时, 将联想词库调入内存后, 还有 513K 的自由内存, 而虚盘上还剩 1 百多 K 空间。拥有 Super PC/XT 及类似 PC 机的朋友, 化少量的经费, 把机器容量扩展到 1MB, 就可以在您心爱的 PC 机上享用到一些只有 AT 机才有的功能了。

;文件名: EXTMEM.ASM

```

CSEG          SEGMENT PUBLIC 'CODE'
               ASSUME CS:CSEG,DS:CSEG,ES:CSEG
DEVICERAM     PROC FAR
               DD 0FFFFFFFH
               DW 8000H
               DW DEV-STARTEGY
               DW DEV-INTERRUPT
               DB 'Extended Memory Drive'
EXTRAM        DW ? ;保存附加存储器容量
;设备驱动程序子功能入口表
FUNCTIONS     DW INITIALIZE ;初始化子功能
SUB1          DW ERR ;设备驱动程序其余 12 个
SUB2          DW ERR ;子功能不用,直接退出
SUB3          DW ERR
SUB4          DW ERR
SUB5          DW ERR
SUB6          DW ERR
SUB7          DW ERR
SUB8          DW ERR
SUB9          DW ERR
SUB10         DW ERR
SUB11         DW ERR
SUB12         DW ERR
KEEP-ES      DW ?
KEEP-BX      DW ?
DEV-STARTEGY:MOV CS:KEEP-ES,ES
               MOV CS:KEEP-BX,BX
               RET
DEV-INTERRUPT:PUSH ES
               PUSH DS
               PUSH AX
               PUSH BX
               PUSH CX
               PUSH DX
               PUSH SI
               PUSH DI
               PUSH BP
               MOV AX,CS:KEEP-ES
               MOV ES,AX
               MOV BX,CS:KEEP-BX
               MOV AL,ES:[BX]+2
               SHL AL,1
               SUB AH,AH
               LEA DI,FUNCTIONS
               ADD DI,AX
               JMP WORD PTR[DI]

```

```

ERR:      OR     ES:WORD PTR[BX]+3,8103H
          JMP     QUIT1
QUIT:     OR     ES:WORD PTR[BX]+3,100H
QUIT1:    POP     BP
          POP     DI
          POP     SI
          POP     DX
          POP     CX
          POP     BX
          POP     AX
          POP     DS
          POP     ES
          RET
;扩展内存驱动程序:
BEGIN:    CMP     AH,88H
          JNZ     NEXT
          MOV     AX,WORD PTR CS:[EXTRAM];
          取附加 RAM 容量
          IRET
NEXT:     CMP     AH,87H
          JZ      RAM
          DB     0EAH ;转 BIOS 的 INT15
INT15_KEE DD    ? ;INT15 向量,由初始化程序填
入
RAM: PUSH  BX
          PUSH  DX
          PUSH  ES
          PUSH  SI
          PUSH  DS
          PUSH  DI
          PUSH  CX
          MOV   AX,ES:[SI+12H];ES:SI 为描述符表首
          址
          PUSH  AX
          MOV   AX,ES:[SI+14H];取源地址
          MOV   AH,00H
          PUSH  AX
          MOV   AX,ES:[SI+1AH]
          PUSH  AX
          MOV   AX,ES:[SI+1CH];取目的地址
          MOV   AH,00H
          MOV   DH,AL
          MOV   CL,0CH
          SHL   AX,CL
          AND   DH,0F0H
          OR    DH,DH;DH 非 0 目的地址则为扩展地址
          JZ    N1
          MOV   DH,01H
          ADD   AX,2000H
N1: MOV   ES,AX;换算后目的地址放在 ES:SI
          POP   SI
          POP   AX
          MOV   DL,AL
          MOV   CL,0CH
          SHL   AX,CL
          AND   DL,0F0H
          OR    DL,DL;DL 非 0 源地址则为扩展地址
          JZ    N2
          MOV   DL,01H
          ADD   AX,2000H
N2: MOV   DS,AX;换算后源地址放在 DS:DI
          POP   DI
          POP   CX;CX 为传递的字(2 字节为一字)数
LP:  MOV   AL,DL
          OUT  0E0H,AL
          MOV  BX,DS:[DI]
          MOV  AL,DH
          OUT  0E0H,AL
          MOV  ES:[SI],BX
          ADD  DI,02H
          JNC  N3
          MOV  AX,DS
          ADD  AX,1000H
          MOV  DS,AX;若跨段,DS+1000H
N3:  ADD   SI,02H
          JNC  N4
          MOV  AX,ES
          ADD  AX,1000H
          MOV  ES,AX;若跨段 ES+1000H
N4:  LOOP  LP
          POP  DI
          POP  DS
          OP   SI
          POP  ES
          POP  DX
          POP  BX
          ADD  SP,04H
          POP  AX
          AND  AL,0FEH ;CF=0
          OR   AL,40H ;ZF=1
          PUSH AX
          SUB  SP,04H
          XOR  AX,AX
          OUT  0E0H,AL ;切回主存
          IRET
EOP: ;驱动程序尾
INITIALIZE: LEA  AX,EOP;初始化程序
          MOV  ES:WORD PTR[BX]+14,AX
          MOV  ES:WORD PTR[BX]+16,CS
;检测附加内存容量
          MOV  AL,01H
          OUT  0E0H,AL
          MOV  BX,00H
          MOV  AX,2000H
L1:  MOV   DS,AX
          MOV  AX,0AA55H
          MOV  WORD PTR DS:[0],AX
          CMP  WORD PTR DS:[0],AX
          JNZ  Q1
          MOV  AX,DS
          CMP  AX,8000H
          JZ   Q1

```


ADD	AX,1000H	MOV	WORD PTR CS:[INT15-KEEP],BX
ADD		MOV	WORD PTR CS:[INT15-KEEP+2],ES
BX,40H		LEA	DX,BEGIN
JMP		MOV	AX,CS
SHORT L1		MOV	DS,AX
Q1: MOV	WORD PTR CS:[EXTRAM],BX	MOV	AX,2515H
MOV	AL,00H	INT	21H
OUT	0E0H,AL	JMP	QUIT
;保存 15H 中断向量和设置新的 15H 中断		DEVICERAM	ENDP
MOV	AX,3515H	CSEG	ENDS
INT	21H	END	DEVICERAM

快速筛法求证素数的程序设计

邯郸市财政局(056002) 王存

如何在计算机上运用筛法求证素数,以往不少报刊资料曾陆续作过一些介绍,但在如何既使求证速度提高,又使程序变得通俗实用方面,却大都属于东鳞西爪,浅尝辄止。

为了给有关科研人员和教育工作者进行数论研究及教学提供方便,本文作者设计编制了一种快速求证素数的程序(见附录)。该程序利用 Eratosthenes 筛法并通过筛法优化,使求证素数的速度显著提高,所得结论准确无误。并且由于全面考虑了数论专业的工作习惯,从而使程序的实用性进一步增强。

在程序设计方面所采取的主要做法是:

一、创立两轮筛选法,使求证素数的速度从根本上得以提高。

为了加快素数的求证,作者采取纯素数因子筛选与纯奇数因子筛选相结合的办法,对被检验是否为素数的整数进行两轮分步筛选。其第一轮筛选,即纯素数因子筛选,乃是通过把 100 以内的素数分别作模而检验所指定整数是否与 0 同余为基本条件,来判断该整数是否为素数。当然,如果经此第一轮筛选尚未能得到证明,则就进入第二轮筛选。其第二轮筛选,即纯奇数因子筛选,乃是从奇数 101 开始,每次加 2 作模而再度检验所指定整数是否与 0 同余,并以此是否同余及其它相关条件为依据,来进一步判断该整数是否为素数。

与两轮筛选同步进行,程序上还通过设置筛选终止关卡以控制筛选进程。其具体做法是:当检验同余过程中的模一经大于或等于被检验整数的平方根,便强令筛选终止,并最终确认该被检验整数为素数。

运用上述筛选方法,与将小于被检验整数 i 的所有自然数分别作模而检验 i 是否与 0 同余的作法相比,求证速度提高的倍数远大于 $2\sqrt{i}$ 。实践证明,运用所附程序求证素数确实相当快速。例如,在 CPU 为 286 的微机上验证 2147483647 这样一个数值为 20 多

亿的整数为素数,大约仅需要 12 秒钟。

成功地实现两轮筛选,程序设计的主要技巧在于:

(1)采取将 100 以内的素数先赋值予一维数组,尔后再依照数组序列进行循环筛选。这样做,便使得第一轮筛选,亦即纯素数因子的筛选变得自成体系且过程简单。
(2)通过设置两轮筛选衔接标志器的状态,构成两轮筛选间具有接替性的有机联系。其基本做法是:事先将两轮筛选衔接标志器 W 置为 0。进入第一轮筛选后,如果被检验整数与 100 以内的某素数同余,则在断定它或为素数或为合数的同时,将标志器 W 置为非零整数。反之,如果被检验整数与 100 以内的素数均不同余,则说明它有大于 100 的因子,尚需进入第二轮筛选,这时,则使标志器 W 保持原来的 0 值不变。是否进入第二轮筛选,就可依据标志器 W 的数值状态来确定:当其为 0 时进入,否则不进入。

二、全面考虑求证素数的专业习惯,使程序充分满足用户的实用需求。

运用筛法求证素数,不外乎处理两种情况:一是检验某一数是否素数,二是检验某一整数区间究竟哪些数是素数。为了程序编制的便利,把这两种情况归化为同一种类型,即整数区间类型 $[X, Y]$ 。这样,当检验某一数是否为素数时,就视 $X=Y$ 而加以处理。但在使程序面向用户时,则是考虑了以下三种情形:

1. 一些特殊数值的处理。若输入数据出现 $X>Y$ 、 $X<1$ 、 $Y<1$ 以及 $X=Y=1$ 的现象,则程序将不作任何处置而自行退出。

2. $X=Y>1$ 的处理。当检验某一整数是否为素数时,程序以 $X=Y$ 的方式进行处理。此时,不仅要求输入 X,而且要求以同样的数值输入 Y。为方便用户,设定了两种输入办法:一是当 X 输入后, Y 以与 X 相同的数值输入;二是当 X 输入后, Y 以“-0”代之输入。

对于求证结果,将以两种提示性方式打印:当 X

为素数时,打印出“整数 X 是素数”;当 X 为合数时,打印出“整数 X 不是素数,它有素因数……”。

3. $X \neq Y (1 < X < Y)$ 的处理。这是检验特定整数区间究竟哪些数是素数。当把 X、Y 输入后,程序将首先打印出“整数 X 与 Y 之间的素数:”一行字。尔后通过检验,若 X、Y 之间没有素数,则打印出“不存在”三字;否则,便相继打印出 X、Y 之间的所有素数,最终打印出“共有 m 个素数”的结束语。

所附程序用 Turbo C 编制。它不仅求证素数快速,能充分满足用户的实用意愿,而且还颇具功能扩展潜力,例如,被检验整数的字长位数可进一步扩充,纯素数因子筛选的数组可随意增大等。

```
# include<stdio. h>
# include<conio. h>
# include<string. h>
# include<math. h>
# include<float. h>
# define V 26
main()
{
    long int i=0,j=0,k=0,l=0,m=0,n=0,U[V],W=0,X
    =0,Y=0,Z=0;
    clrscr();
    gotoxy(1,4);
    printf("\n\n\n      请输入待检验的整数所处的");
    printf("自然数范围(0<X<=INTEGER<=Y):");
    printf("\n\n          X=");
    scanf("%ld",&X);
    printf("          Y=");
    scanf("%ld",&Y);
    if(Y== -0)
        Y=X;
    if (X>Y || X<1 || Y<1 || (X==1 && Y==1))
        {printf("\n\n      输入数据不符合要求,");
        printf("出现了 X>Y 或 X<1 或 Y<1 或 X=Y=1 的情
        形,故退出!");
        while (i<500000)
            i=i+1;
        exit(0);
        }
    gotoxy(20,18);
    printf("请打开打印机,马上进行打印:");
    while (i<500000)
        i=i+1;
    gotoxy(19,18);
    printf("正在求证并将打印,请等待……");
    U[1]=2;U[2]=3;U[3]=5;U[4]=7;U[5]=11;U[6]
    =13;U[7]=17;U[8]=19;U[9]=23;U[10]=29;U[11]=
    31;U[12]=37;U[13]=41;U[14]=43;U[15]=47;U[16]
    =53;U[17]=59;U[18]=61;U[19]=67;U[20]=71;U
    [21]=73;U[22]=79;U[23]=83;U[24]=89;U[25]=97;
    if(X==Y)
        {k=1;
        while(k<26)
            {if(X%U[k]==0)
```

```
{if(X==U[k])
{m=1;W=1;}
else
{j=U[k];W=-1;}
k=26;
}
k=k+1;
}
if(W==0)
{j=101;m=1;
Z=(long int)sqrt(X);
while(X%j! =0 && j<=Z)
j=j+2;
if(X%j==0 && X! =101)
m=0;
}
}
if(m==1)
printf(stdprn,"整数 %ld 是素数.",X);
else
printf(stdprn,"整数 %ld 有素因数 %ld",Y,j);
}
if (X! =Y)
{printf(stdprn,"\n      整数 %ld 与 ",X);
printf(stdprn,"%ld 之间的素数:\n",Y);
if (X==1)
X=2;
for(i=X;i<=Y;i++)
{z=(long int)sqrt(i);
k=1;W=0;
while(k<26)
{if(i%U[k]==0)
{if (i==U[k])
{m=m+1;n=(long int)(m/10+1);W=1;
if(m%10==1)
printf(stdprn,"\n");
printf(stdprn,"%12ld",i);
}
}
else
{j=U[k];W=-1;}
k=26;
}
k=k+1;
}
}
if(W==0)
{j=101;
while(i%j! =0 && j<=Z)
j=j+2;
if((i%j! =0 && j>=Z) || i==101)
{m=m+1;
if(m%10==1)
{printf(stdprn,"\n");
if(n==55)
{n=0;
for(l=0;l<11;l++)
printf(stdprn,"\n");
}
}
}
}
```



```

    n++;
}
fprintf(stdprn, "%12ld", i);
}
}
if(i<Y)
    i=i+1;
else
    break;

```

```

}
if(m<1 && 7<X)
    fprintf(stdprn, "\n 不存在!!");
else
    fprintf(stdprn, "\n 共有 %ld 个素数。", m);
}
fprintf(stdprn, "\n");
}

```

文件(目录)的快速查找

国防大学电化教学中心(100091) 刘炳文

文件查找也称文件定位,即确定文件或目录在磁盘上的位置。随着计算机技术的发展,磁盘的容量越来越大,可以存放大量的文件。文件较多时,通常按类别建立若干目录。每个目录下可有数以十计甚至数以百计的文件。在这种情况下,查找指定的文件往往是一件十分复杂和繁琐的事。为此,国外已推出专门用于文件查找的软件。这里介绍一种方法,可以在复杂的层次目录结构中方便地查找指定的文件。为了便于理解,我们先简单介绍 DOS 下的 CHKDSK 和 FIND 命令。

CHKDSK 用来查看、分析指定或缺省磁盘上的目录和文件分配表,提供磁盘和内存状态的报告。它有两个选择项:/V 和/F。当选择项为/V 时,将显示指定盘上的所有目录和文件以及所发现的错误、可用磁盘容量及内存空间。例如:

CHKDSK A:/V
其显示结果如下:

```

Directory A:\
  A:\BEEP.BAT
  A:\LIU.FOR
Directory A:\BAS
  A:\BAS\SEARCH.BAS
  A:\BAS\COUNTER.BAS
Directory A:\BAS\POL
  A:\BAS\PCL\TABLE.POL
  A:\BAS\POL\LIST.POL
Directory A:\WPS
  A:\WPS\DEFUN.WPS
  A:\WPS\STAR.WPS
  A:\WPS\WORDPRO.WPS
Directory A:\BAT
  A:\BAT\BEEP.BAT
  A:\BAT\FASTCOPY.BAT
  A:\BAT\KEY.BAT
1457664 bytes total disk space
3072 bytes in 4 directories

```

FIND 命令用来搜索一个或一组文件中的指定字符串,并把含有该字符串的所有行送到标准输出设备,如:

FIND "The dentist said" story.doc
将显示文件 STORY.DOC 中含有字符串"The dentist said"的所有行。

在缺省情况下,DOS 命令从键盘上输入,在显示器上输出命令的执行结果,键盘和显示器分别被称为标准输入输出设备。但是,利用 DOS 的重定向功能,可以使操作结果输出到一个文件中去。例如:

CHKDSK A:/V > CHKOUT
将把 CHKDSK 命令的执行结果输出到文件 CHKOUT 中。

查找指定文件的操作通过 CHKDSK 和 FIND 命令以及重定向功能来实现,通常把它们放在批处理文件中。例如,我们可编写一个名为 SEARCH1.BAT 的批处理文件,其内容如下:

```

echo off
if "%1"==" " goto :error
echo 现在开始查找指定文件,请稍候...
chkdsk/v > temp
echo 查找结果为:
find "%1" temp
erase temp
echo 查找结束...
goto: end
:error
echo 在 SEARCH 命令后必须给出文件名,再试一次...
:end

```

程序首先检查用户是否键入了要查找的文件名,如果没有(即参数%1 为空),则转到错误处理程序(:ERROR),提示用户键入需要查找的文件名。如果键入了文件名,则显示相应的信息,并开始查找。

程序 SEARCH1.BAT 执行 CHKDSK /V 命令,把结果重定向输出到临时文件 TEMP 中,接着执行 FIND 命令,找出指定的文件名,然后删除临时文件。如果键入完整的文件名,则所找到的文件名通常是唯一的;如果键入文件名的一部分,则可能找到多个文件名。两次执行情况如下:

```

(1)
A>SEARCH1 WORDPRO
屏幕上将显示:
现在开始查找指定文件,请稍候...
查找结果为:
——temp
A:\WPS\WORDPRO.WPS

```

查找结束...

执行上述批命令,查找指定的文件 WORDPRO.
该文件在 A:\WPS 目录下。

(2)

A)SEARCH1 BAT

现在开始查找指定文件,请稍候...

查找结果为:

```

---temp
      A:\BEEP.BAT
Directory A:\BAT
      A:\BAT\BEEP.BAT
      A:\BAT\FASTCOPY.BAT
      A:\BAT\KEY.BAT

```

键入 SEARCH1 BAT 后,查找到 4 个扩展名为 .BAT 的文件,其中 1 个在根目录下,3 个在子目录 A:\BAT 下。

程序 SEARCH1. BAT 可查找指定的文件,但不能区分要查找的是文件名还是目录名。此外,该程序只能查找当前驱动器上的指定文件,不能由用户指定驱动器名。对 SEARCH1. BAT 稍作修改,即可查找指定驱动器上的目录或文件。修改后的程序如下(文件名为 SEARCH2. BAT):

```

echo off
if "%2"==" " goto :error
if "%1"=="DIR" goto :dir
if "%1"=="FILE" goto :file
echo 在命令中必须输入“DIR”或“FILE”,再试一次...
goto :end
:dir
echo 查找%3 驱动器上指定目录%2:
chkdsk %3 /v|find "Directory" > temp
find "%2" temp
erase temp
goto :end
:file
echo 查找%3 驱动器上指定文件 %2:
chkdsk %3/v > temp
find "%2" temp
goto :end
:error
echo 在命令中应指定“DIR”或“FILE”,再试一次...
end

```

上述程序可以查找指定驱动器上指定的文件或目录。执行时需要输入三个参数,分别为查找类型(FILE 或 DIR)、文件或目录名、驱动器号。例如:

SEARCH2 DIR S C:

查找 C 驱动器上名为 S 的目录名。执行结果如下:

```

查找 C:驱动器上指定目录 S:
---temp
Directory C:\WS
Directory C:\WPS
再如:
SEARCH2 FILE BAT C:

```

用来查找 C:驱动器上的批处理(BAT)文件。执行结果如下:

```

查找 C:驱动器上指定文件 BAT:
---temp
C:\AUTOEXEC.BAT
C:\WPS\XSDOS.BAT
C:\WPS\WP.BAT
C:\WPS\WS.BAT
C:\TC.BAT

```

SEARCH2. BAT 中的“|”是管道操作符,它连接两个命令,把第一个命令的输出和两个命令的输入连起来。这里的管道操作

CHKDSK %3/V | FIND "Directory" > temp

把 CHKDSK 命令的输出作为 FIND 命令的输入。

执行 SEARCH1 或 SEARCH2 命令时,输入的文件名或目录名必须大写,因为在 DOS 下显示的文件名和目录名一律为大写。

巧用长城 0520EM 运行汉字 dBASE III

山东烟台商校八九统计(264000) 辛启亮

长城 0520EM 机型是中国长城计算机公司 1988 年推出的准 16 位型机,该机为 512K RAM,固化汉字库 ROM,美中不足的是只有一个 360K 的软盘驱动器,为运行汉字 dBASE III 带来很多困难,很易死机。本人经过多次实践,发现可利用 EM 机汉字系统(GW-BIOS)占用内存少的特点开发一个 64K 的虚拟磁盘,以很好地解决这一问题。

在系统盘中建 CONFIG. SYS 文件,其内容如下:

```

device=VDISK. SYS 64 128 15
device=CKB9. SYS
device=GRD. SYS
files=20
buffer=10

```

这样在机器启动后就开辟一个 64K 的虚拟磁盘,在运行汉字 dBASE III 前,将要使用的库文件,命令文件等拷入虚盘,并在 dBASE III 的系统盘(至少要含有 DBASE. EXE 和 DBASE. OVL 两个文件)上建立文件 CONFIG. DB,其内容如下:

DEFA=C:

这样系统就自动到 C 盘(虚拟磁盘)上寻找库文件等,使用虚盘的最大优点就是存取速度快,但必须注意以下几个问题:

1. 在运行 dBASE 时,dBASE III 的系统盘不能离开驱动器 A,否则在退出 dBASE 时就会死机。
2. 退出 dBASE 后,已修改过的虚拟盘中的文件必须从 C 盘拷出来,以防重新启动后丢失数据。

(7)冒号:。文本头提示符。表明光标已经移至本文头。使用者一般可不予以理会。

8. 输入文本过程中怎样防止已输入文本丢失?

输入文本过程中,已输入文本丢失的情况时有发生。例如,突然断电或电源电压大幅度波动;病毒激活造成键盘死锁或系统瘫痪;C-WordStar 软件本身出错等。遇到这些情况,如未采用防患于未然的措施,则已输入的文本便会丢失。这可以说是C-WordStar 设计上的一个缺陷。补救的办法一般有两种,其一,在输入文本的过程中,随时使用^KS 命令存盘,将内存中已经存在的文本写往磁盘,生成一磁盘文件。使用^KS 命令后,紧接着再使用一次^QP 命令,光标会自动回到刚才的位置,此时操作者便可以继续输入。要注意,使用^KS 和^QP 命令之间一定不能击其它任何键,否则光标会处于文本头。如果文本很长,移动光标是很费时的。其二,设计一个模拟^KS 和^QP 命令的程序,让其驻留内存,在输入文本的过程中,让它定时模拟^KS 和^QP 命令,以代替操作者的人工操作。上述两种补救办法都可以在一定程度上弥补C-WordStar 本身的不足,当遇到断电等不测事故时,使丢失的文本减少到一定程度,即只会丢失最后使用^KS 命令存盘到断电发生这一段时间内输入的文本。

9. 输入文本过程中应注意些什么?

输入文本虽然不是十分复杂的操作,但是不同的操作者输入的效率却有很大的差别。除了击键指法是否正确熟练,对汉字编码方式的理解和运用是否纯熟等共性方面的因素外,对C-WordStar 功能的理解是否深刻也有很大关系。这里指出的一些注意事项可供初学者参考。首先,要正确使用硬回车。初学者为了使输入的文本实现右边缘对齐,往往在一行结束时使用硬回车,即一行结束时击回车键。这不仅因为要判断一行是否结束而分散精力,降低输入效率,而且由于一行一回车,将造成排版功能不起作用。正确的方法是,根本不理睬一行是否结束,只管埋头输入,仅当文本的一个自然段结束时才使用一次回车键。其次,输入文本的过程中,不得随意键入不可见字符,特别是一些控制字符。当文本输入完毕,需要打印输出时,在打印的过程中,有时会出现莫名其妙的死机现象。造成死机的原因固然很多,但输入文本的过程中,自觉或不自觉输入了不可见字符,便是原因之一。由于不可见字符不会在屏幕显示,因此查找困难,一种无可奈何的处理办法是,从死机的字符(或汉字)开始重新输入几个或十几个字符(或汉字)。如果造成死机的原因确实是由于输入了不可见字符,那么死机的现象自然就不复存在了。第三,输入文本的过程中,一定要随时使用^KS 存盘,以便把遇到断电等不测事故时所丢失的文本减少到可允许的程度。作为一个建议,一般情况下,对初学者来说十分钟便应该存盘一次。第四,输入文本的过程中,要注意标点符号的使用。使用标点符号的原则是,凡是键盘上有的标点符号,一律使用键盘上的标点符号,只有中文中特有,而西文中没有(即键盘上没有)的标点

符才使用纯中文的标点符号。例如,文本中大量使用的逗号,一律使用键盘上的逗号“,”;句号则使用纯中文的句号“。”。同时也要防止另一种倾向,为了省事,句号也使用键盘上的句号“.”。按照笔者建议的原则使用标点符号,不仅可保证较高的输入效率,同时也维护了汉语言文字的规范化和严肃性,又为发挥C-WordStar 的排版功能留下了充分余地。在怎样正确使用^B 命令的有关问题中,对此有更多的说明。最后,输入文本的过程中,不必过分注意文本的格式,而应该集中精力,全神贯注地输入文本,以提高输入效率。这里所提到的文本格式,主要包括文本宽度,右边缘是否对齐,标点符号是否处于行首等,甚至还可包括文字是否有错,标点符号是否用错等,这些问题一律留待校对和处理文本格式时解决。

CCDOS 4.0 解密的方法

湖南省外贸学校电教中心(410114) 范恒钦

CCDOS 4.0 是一个较好的汉字系统,在 IBM 兼容机中应用广泛。它的核心由三个文件组成,CC-CC.COM 是系统装入文件,CCCC.OVR 是键盘管理文件,而 CCCC.OV1 或 CCCC.OV2 为显示模块,前者用于彩卡,后者用于单色卡。以上三个文件中 CC-CC.COM 首先被执行,此文件并未加密,但它内部有两个解密子程序,它们用于对 CCCC.OVR 进行解密,它们的入口地址分别是 XXXX:02BF 和 XXX:X02EA,其解密方法是类似的。不过前者每次解密四个字节,而后者可解密任意字节(不超过FFFF)。键盘管理模块 CCCC.OVR 被解密后才能被执行。而在 CCC-C.OVR 中又有一解密程序,它负责为后面要执行的显示模块解密。它的解密方法不同于前两个子程序所用方法。总体看来,三个核心文件是这样执行的:先执行 A,A 为 B 解密,再执行 B,B 为 C 解密,最后执行 C。

CCCC.COM 中的解密方式是先将一个字节的密文取反,再减去数 X,然后再与密钥作异或运算,将结果加上数 X,从而得到原文。麻烦的是解密下个字节时,X 并非固定不变,它在 0F 到 00 之间循环变化,而密钥也是随 X 而变的。也就是说,哪怕是同一个原文,例如字母 A,由于处于程序的不同位置而被加密成不同的代码。

以下是可解密任意字节的那段子程序,DX 中放欲解密文首址,CX 中放欲解密字节数。

```
xxxx 02EA    PUSH BX;保存主程序中 BX 的值
xxxx 02EB    MOV SI,DX ;密文首址送指针 SI
xxxx 02ED    MOV DH,0F ;X 初值为 0F
xxxx 02EF    MOV DI,SI ;密文首址送 DI
xxxx 02F1    LODSB ;取一字节密文放 AL
xxxx 02F2    XOR BH,BH ;BH 清零
```



```

xxxx 02F4    MOV BL,DH ;X 值送 BL
xxxx 02F6    NOT AL ;密文取反
xxxx 02F8    SUB AL,BL ;再减去 X
xxxx 02FA    CS;
xxxx 02FB    XOR AL,[BX+0177];再异或上密钥
xxxx 02FF    ADD AL,BL ;再加上 X
xxxx 0301    STOSB ;将已得到的原文放回原处
xxxx 0302    DEC DH ;X 值减一
xxxx 0304    JNS 0308 ;X 不为负则转 0308
xxxx 0306    MOV DH,0F ;否则 X 值又回到 0F
xxxx 0308    LOOP 02F1 ;若字节数未滿,返回继续
                处理
xxxx 030A    POP BX ;恢复 BX 值
xxxx 030B    RET ;返回主程序
xxxxx 0177 DB 23,65,72,AC,29,DF,AA,67,55,ED,
                78,91,AB,73,5A,AC

```

不难发现,以上程序每处理 16 个字节其密钥就会重复。如果将以上解密过程逆过来,我们就可得到对应的加密程序。但需注意,加法的逆运算是减法,减法的逆运算是加法,而异或的逆运算仍是异或。

下面我们再来看看 CCCC.OVR 中的解密程序,它首先将密文与密钥 1 作异或,再减去数 X,数 X 是在 0B 至 00 中循环变化,密钥 1 也随 X 而变。然后与密钥 2 作异或,再加上数 Y,而 Y 在 0A 至 00 中循环变化,同样,密钥 2 是随 Y 变的。可以看出这种密文比上一种复杂。数 X 有 12 个值,数 Y 有 11 个值,共有 12×11=132 种不同的组合。换句话说,要每译 132 个字节,密钥(密钥 1+密钥 2)才会产生重复。显然比上一种更难破译。以下是这段程序。(再次提请注意:这段程序原来是加密的,这里写的是被上一程序解密后的结果。)

```

xxxx 855B MOV CX,AX;本次需解密的字节数送 CX
xxxx 855D MOV AH,3E;调用号送 AH
xxxx 855F INT 21;关闭显示模块文件
xxxx 8561 MOV DH,0B;X 初值为 0B
xxxx 8563 MOV DL,0A;Y 初值为 0A
xxxx 8565 MOV SI,0100;密文首址送指针 SI
xxxx 8568 MOV DI,SI;原文存放首址送 DI(也是 0100)
xxxx 856A PUSH DS;压 DS
xxxx 856B POP ES;DS 送 ES
xxxx 856C LODSB;取一字节密文
xxxx 856D XOR BH,BH;BH 清零
xxxx 856F MOV BL,DH;X 值送 BL
xxxx 8571 CS;
xxxx 8572 XOR AL,[BX+8242];密文与密钥 1 异或
xxxx 8576 SUB AL,BL;减去 X
xxxx 8578 MOV BL,DL;Y 值送 BL
xxxx 857A CS;
xxxx 857B XOR AL,[BX+8238];密文与密钥 2 异或
xxxx 857F ADD AL,BL;加上 Y 得到原文
xxxx 8581 STOSB;存回原处
xxxx 8582 DEC DH;X 值减一
xxxx 8584 JNZ 8588;不小于零转 8588
xxxx 8586 MOV DH,0B;否则 X 回初值
xxxx 8588 DEC DL;Y 值减一
xxxx 858A JNS 858E;不小于零转 858E
xxxx 858C MOV DL,0A;否则 Y 回初值
xxxx 858E LOOP 856C;字节数未滿,返回
xxxx 8238 DB 34,4F,23,67,9D,92,38,67,23,4B
xxxx 8242 DB 28,3A,62,19,86,65,6D,2C,81,8E,AE,
                43,43,43

```

1741 病毒的检测及其清除

四川射洪县人民银行微机室(629200) 邹肇辉

近一个时期,我们在不同单位的十几台微机上连续发现了一种文件型病毒,著名的 KILL46.0 和 CPAV 1.0 均不能将之清除。

该病毒长 1741 字节,按通常的命名法,称之为 1741 病毒。当运行带毒文件时,它便驻留内存高端,占用 2K 空间,并修改 INT 21H 中断向量,使之指向病毒体。由于病毒控制了 INT 21H 的 4B 功能,操作系统只要调入 .EXE 或 .COM 文件,在执行前都将通过病毒体而且被加以改造;在文件尾挂上一段病毒程序,并修改 .EXE 文件头的文件长度、初始化 IP 值、代码段偏移值三项,对于 .COM 则修改文件的头三字节。病毒附着在可执行文件后的长度并不固定,一般在 1741—1760 之间,视它在加挂 1741 字节病毒体前在原文件后添加几个 00H 而定。该病毒可重复感染。

1741 病毒的表现,基本上是良性的。但它感染了 COMMAND.COM 文件,则使汉字系统出错甚至死

机。从这点看,它又表现出恶性病毒的特征。

我们对 1741 病毒进行了一段时间的跟踪分析,发现病毒体在感染了可执行文件后,会把该文件的文件名记录在自己体内固定偏移处,并且在另一固定偏移处有一特征字符串“VUWRQSP”。我们便以此两点作为判定文件是否染毒的标志。另外,病毒还将 .EXE 文件头中被改动的部份和 .COM 文件的头三字节保存在自己体内,我们只需将它们恢复到原位处,并从尾部截断病毒体,即可达到彻底清除 1741 病毒的目的。

S1741.C 程序实现了上述想法。该程序用 Turbo C 2.0 编译后,即可运行。用法:S1741 [盘符]。

运用该消毒程序,我们为二十多台微机清了毒,有效地遏止了 1741 病毒的蔓延。

```

/* S1741.C 程序清单 */
#include <stdio.h>
#include <string.h>
#include <fcntl.h>

```

```

#include <stdlib.h>
#include <io.h>
#include <dir.h>
#include <dos.h>
char driver curdriver,path[MAXDIR],curpath[MAXDIR];
int p;
pathm();
dirm();
scan(char *);
main(int argc char * argv[])
{
curdriver=getdisk();
getcwd(curpath,MAXDIR);
if(argc>=2&&argv[1][1]=';')
(driver=toupper(argv[1][0])-'A';
setdisk(driver);
driver+='A';
}else
driver=curdriver+'A';
chdir("\\");
pathm();
printf("\n\n");
setdisk(curdriver);
chdir(curpath);
}
pathm()
(struct fblk dirment;
chdir("\\");
dirm();
getcwd(path,MAXDIR);
p=findfirst(" * . * ",&dirment,0x3f);
while(! p)
{
if((dirment.ff-attrib& $ 0x10)==FA-DIREC)
{chdir(dirment.ff_name);
printf("\n\\%s\n",dirment.ff_name);
dirm();
chdir(path);
}
p=findnext(&dirment);
}
}

dirm()
{
struct fblk ffb;
int done;
done=findfirst(" * . exe",&ffb,0);
while(! done)
{
scan(ffb.ff_name);
done=findnext(&ffb);
}
done=findfirst(" * . com",&ffb,0);
while(! done)
{

```

```

scan(ffb.ff_name);
done=findnext(&ffb);
}
}

scan(char * comm1)
{
FILE * fi;
char * comm2, * comm3, * comm4, * ptr1, * ptr2;
char buf[1760];
int x,y,handle;
long length;
comm2="VUWRQSP";
comm3=". COM";
comm4=". EXE";
if((fi=fopen(comm1,"r+b"))==NULL)
{printf("\nFile %s Can not open! \n",comm1);
return;
}
fseek(fi,-0x8fL,SEEK-END);
fread(buf,0x8fL,1,fi);
ptr1=strstr(buf,comm2);
fseek(fi,-32L,SEEK-END);
fread(buf,32L,1,fi);
ptr2=strstr(buf,comm1);
if((ptr1==NULL)|| (ptr2==NULL)){printf("No 1741
virus
in %s \r",comm1);
fclose(fi);}
else
{printf("\Fcund 1741 virus in %s!!! \n",comm1);
if(strstr(comm1,comm3)!=NULL)
{
fseek(fi,-1741L,SEEK-END);
fread(buf,3,1,fi);
fseek(fi,0L,SEEK-SET);
fwrite(buf,3,1,fi);
fclose(fi);
handle=open(comm1,O_WRONLY|O_BINARY);
chsize(handle,filelength(handle)-1746L);
close(handle);
printf("1741 virus already cleaned! \n);}
if(strstr(comm1,comm4)!=NULL)
{fseek(fi,-1332L,SEEK-END);
fread(buf,2,1,fi);
fseek(fi,20L,SEEK-SET);
fwrite(buf,2,1,fi);
fseek(fi,-1328L,SEEK-END);
fread(buf,2,1,fi);
fseek(fi,22L,SEEK-SET);
fwrite(buf,2,1,fi);
fseek(fi,0L,SEEK-END);
length=ftell(fi)-1741L;
x=length/512+1;
y=length%512;
fseek(fi,2L,SEEK-SET);

```

```

putw(y, fi);
fseek(fi, 4L, SEEK_SET);
putw(x, fi);
fclose(fi);
handle = open(comm1, O_WRONLY|O_BINARY);
chsize(handle, filelength(handle) - 1741L);

```

```

close(handle);
printf("1741 virus already cleaned! \n");
}
}
}

```

谈 6502 软中断功能的开发应用

苏 华

熟悉 8088 系列处理器的人都知道软中断指令 INT 在调用系统功能方面的重要应用。例如,在调用 BIOS 的键盘输入处理功能时要用 INT 16H 中断,调用显示功能时用 INT 10H 中断,而通过 DOS 去调用类似功能或文件管理、内存分配等功能时则使用各种功能号的 INT 21H 中断。6502 处理器也有一条相应的软中断指令 BRK,可为我们提供类似于 INT 指令功能的使用条件。但在苹果机或 CEC-I(以下统称学习机)中并未加以利用,仅是作为调试机器语言程序的一种手段。在八位机面临完成其历史作用的今天,似乎也没有什么必要再去研究它。但是,如果您想把身边的学习机改造成特定用途的专用机,而又不想让别人轻易地解读、剽窃您的软件,或者仅仅是让初学者借以理解计算机的中断处理机制,也还是值得加以探讨的。

一、6502 cpu 的中断机制

不论是在程序中通过 BRK 指令设置的软中断,还是外部设备有输入输出请求时向 6502 发出的硬中断信号,都会引起 cpu 作出下述一系列响应:(硬中断)

1. 执行完正在处理的指令;
2. 将中断处理后的返回地址(中断返回地址)按高、低字节顺序送入堆栈保存;
3. 将状态寄存器 P 的内容(状态字)存入堆栈;
4. 转去 \$FFFE、\$FFFF 单元内指定的地址(中断向量)去作中断响应处理。

中断响应处理的内容视设计要求而定。它可能是读入一个按键的键码,也可能是向打印机送出一个 ASCII 码,或者是设计要求的其它功能。值得一提的是,学习机在处理键盘输入或向打印机输出时均未利用中断功能,而是采用设备查询方式。例如在需要时检查 \$C010 单元的最高位,判断是否有键按下以作相应的处理;再如向打印机输出之前,反复检查 \$C1C1 单元(设打印卡插在 1 号接口槽),待最高位为零,表示打印机不忙碌时,才通过 \$C090 单元送出 A 寄存器中的字符代码。而在 IBM-PC 系列机方面,则是利用按键引发的硬中断来处理键盘输入的,其好处是 cpu 只有在有键按下时,才暂时放下当前的操作,转入相应的中断处理,把键码暂存入键盘缓冲区再继续运行原来的程序,不用专门安排键盘查询,提高了处理效率。中断机制使得 cpu 在进行其它工作时,多次输入的键码得

以保存(在键盘缓冲区允许范围内),而在学习机上只能留下最近一次输入的键码。

由于中断时 cpu 只保存返回地址和状态字,在设计学习机的中断处理程序时一般应作如下安排:

1. 保存 A、X、Y 各寄存器的内容;
2. 转入具体的中断处理;
3. 恢复保存的 A、X、Y 寄存器内容;
4. 通过 RTI 指令返回并继续被中断的主程序。

RTI 指令从堆栈中取出状态字到 P 寄存器,再取返回地址的低、高字节送入程序计数器,它对 A、X、Y 寄存器没有影响,这样 cpu 便可以保持中断时的现场继续主程序的运行。

二、6502 cpu 怎样区分硬中断和软中断

6502 的状态字由下面七个标志位组成。

7		4				0
N	V	B	D	I	Z	C

只有在执行 BRK 指令中断时, B 标志位才置 1。cpu 正是根据这个标志位来区分软中断和硬中断的。前面说过,不论是硬中断还是软中断,6502 都是到 \$FFFE、\$FFFF 单元找中断向量。这两个单元的内容在学习机上分别是 \$40 和 \$FA,最初的中断处理便从监控的 \$FA40 地址开始:

FA40— STA	\$45	; 暂存 A 寄存器内容
FA42— PLA		; 从堆栈取状态字至 A 分析
FA43— PHA		; 状态字仍保存在堆栈中
FA44— ASL		; 状态字左移三位,将
FA45— ASL		; B 标志移到 A 寄存器
FA46— ASL		; 的最高位
FA47— BMI	\$FA4C	; B=1, 软中断转 \$FA4C
FA49— JMP		; B=0, 去 \$03FE、\$03FF 单
	(\$03FE)	
		; 元指定的地址处理硬中断
FA4C— PLP		; 软中断, 从堆栈恢复状态字
FA4D— JSR	\$FF4C	; 将 A、X、Y、P、S 寄存器内容
		; 分别存入 \$45~\$49 零页单元
FA50— PLA		; 从堆栈依次取出返回
FA51— STA	\$3A	; 地址低、高字节值,分
FA53— PLA		; 别存入 \$3A、\$3B 零页
FA56— STA	\$3B	; 单元

FA54— JMP ;去 \$03F0、\$03F1 单元指定 (\$03F0)
;的地址处理软中断
FA59— JSR \$F882 ;显示 \$3B、\$3A 内容(返回地址)
FA5C— JSR \$FADA ;显示保存于 \$45~\$49 单
;元的 A、X、Y、P、S 寄存器内容
FA5F— JMP \$FF65 ;重新进入监控待命状态。

软中断处理程序的入口地址低、高字节分别放在 \$03F0 和 \$03F1 单元内,在开机时已被设定为 \$59 和 \$FA,因此执行 JMP(\$03F0)的结果实际上就是执行上面最后三条指令,不再返回主程序而重新进入监控,并显示各寄存器在中断时的内容。我们在开发程序时,将某一处指令的操作码用 BRK 指令代换来设置断点,就可以对程序进行调试。

三、学习机软中断功能的开发应用

BRK 是单字节指令,监控系统的反汇编程序也是把 BRK 作为单字节指令处理的,但实际上软中断的返回地址并不紧接在 BRK 指令之后,而是隔开一个存储单元。如果我们在 BRK 指令的下一个存储单元内放置一个功能代码,在 \$03F0、\$03F1 单元设置自定义的软中断处理程序入口地址,并在软中断处理中根据不同的功能码作相应的处理,就可以实现类似 PC 机的 INT 指令的功能。

为使软中断处理后能正确返回主程序,应把返回地址及状态字按原来的顺序重新送入堆栈,然后才取功能号分析。软中断处理程序的开始部分如下:

```
LDA $3B ;取返回地址高字节至 A
PHA ;送存堆栈
LDA $3A ;取返回地址低字节至 A
PHA ;送存堆栈
LDA $48 ;取状态字至 A
PHA ;送存堆栈
LDA $3A ;取返回地址低字节分析
BNE L1 ;不为零则转 L1 减 1
DEC $3B ;否则返回地址高低字节也需减 1
L1: DEC $3A
LDY # $00
LDA ($3A),Y ;取功能码至 A 寄存器分析
BEQ 0 号功 能入口地址
CMP # $01
BEQ 1 号功 能入口地址
:
```

在各功能处理程序的出口,可以通过下面两条指令返回主程序:

```
JSR $FF3F ;恢复中断时的 A、X、Y、P 寄存
RTI ;器内容及中断返回
```

上述程序结构只是为了叙述方便而加以简化,开发者可以精心设计甚至采取一些隐蔽技巧,使程序难以被解读。有一点要注意:由于 DOS 也使用 \$45~\$48 单元,如果在您设计的功能调用中要实现 DOS 操作,如读写磁盘等,就需事先把保存在 \$45~\$48

单元的 A、X、Y、P 寄存器内容转存到其它安全的地方,如零页的 \$06~\$09 单元(S 寄存器内容通常不需保存),并另行设计恢复这些寄存器内容的程序段。当然,在中断返回时是否恢复这些寄存器则视需要而定。

从理论上讲,可以为 BRK 指令设置 256 种功能码,远超过实际需要。如果精心选择 6502 的 151 条指令的某些操作码作为功能码,就可以增加程序的隐蔽性。功能号较多时,可以用查表的办法,按功能号将入口地址(减 1!)高、低字节,送入堆栈,再用 RTS 指令进入相应的处理程序。

下面的简单例子只利用软中断实现单一的功能:对被加密的程序 1 经程序 2 解密、运行后再用程序 2 加密,因此未采用上述软中断处理逻辑。已加密的程序 1 选用某个特征码(本例为 \$BB)作结束符:

```
程序一
2000— 0F 9C 09 79 37 CC F9 B9
2008— A1 5B FA 88 B4 27 98 7E
2010— F9 4F BB
程序 2
0360—A9 4C LDA # $4C ;密钥为 $4C,也是 JMP
0362—85 FC STA $FC ;指令的操作码
0364—A0 00 LDY # $00 ;设定程序 1 起始索引值
0366—A9 20 LDA # $20 ;$FA、$FB 单元内容
0368—85 FB STA $FB ;指向程序 1 首地址
036A—A9 00 LDA # $00
036C—85 FA STA $FA
036E—38 SEC ;让进位标志参与加密操作
036F—A5 FC LDA $FC ;取密钥
0371—6A ROR ;循环右移一位
0372—85 FC STA $FC ;暂存密钥单元
0374—B1 FA LDA($FA),Y ;取程序 1 一个字节
0376—49 BB EOR # $BB ;判断是否已结束
0378—F0 0D BEQ $0387 ;结束则退出加密操作
037A—B1 FA LDA($FA),Y ;否则与密钥
037C—45 FC EOR $FC ;作异或运算
037E—91 FA STA($FA),Y ;存回加密解密结果
0380—C8 INY ;索引值加 1
0381—D0 EC BNE $036F ;继续加解密
0383—E6 FB INC $FB ;满 256 次地址高字节增 1
0385—D0 E8 BNE $036F ;继续加/解密处理
0387—60 RTS
```

将程序 1 未加密的原本用程序 2 加密后存盘。运行时将程序 2 及已加密的程序 1 调入内存,执行程序 3 通过软中断运行程序 1。软中断入口地址为 \$0300,其低、高字节值应事先置入 \$03F0、\$03F1 单元。

```
程序 3
6000—00 BRK
6001—4C 00 20 JMP $2000 ;6001 单元存密钥
软中断处理程序:
0300—A5 3A LDA $3A ;将返回地址减 1
0302—D0 02 BNE $0306
0304—C6 3B DEC $3B
```

```

0306—C6 3A   DEC $ 3A
0308—A5 3B   LDA $ 3B       ;调整后的返回地
030A—48      PHA                ;址重新送入堆栈
030B—A5 3A   LDA $ 3A
030D—48      PHA
030E—A5 48   LDA $ 48       ;状态字送入堆栈
0310—48      PHA
0311—A0 00   LDY # $ 00
0313—B1 3A   LDA ($ 3A),Y     ;取密钥(此处为 4C)
0315—85 FC   STA $ FC        ;存入密钥单元
0317—20 66 03 JSR $ 0366       ;对程序 1 解密
031A—40      RTI              ;返回 $ 6001 去执行程序 1

```

运行看看有什么结果:

```

* 3F0:0 3✓
* 6000G✓

```

这种加密技巧当然不难破解,本文的目的只是借以讨论软中断的机制及一种可能的应用方面,更高超的技巧及种种应用,还有待进一步去开发。

(上接第 32 页)口方安,系统键处理及联机功能强,并配有详尽的教材,故采用本系统作为大专院校单片机教学实验机型,能有效增强学生动手能力,并易于教学,且成本低廉。

系统配置

一、主控模块

CPU 为 8031,8M 晶振、单字节指令执行时间最快能达 2μS;

2764 8K ROM(监控占用 4K),6264×2 16K 仿真/用户程序调试区,6264—I 带掉电保护;

8255 并行输出用端口,ADC0809 8 路 8 位 A/D 转换、转换时间 100μS;

标准 RS232 串口一个、8279 键盘显示控制,不占用 CPU 机时,24 键六位 8 段 LED 显示器,仿真插座,三总线插座,并行口插座。

二、功能模块

单价:(元/台)

1. 仿真/用户程序调试运行空间扩展 RAM 卡 580
2. 用于智能巡检仪表、工业多点检测扩展卡(最大 64 点) 250
3. EPROM 写入卡(可写 2716~27256) 150
4. 高精度采集卡(12 位) 420

三、系统工具软件及配件

低密软盘 1 张,具有 MCS—51 汇编源文件编辑、汇编、反汇编、双向通讯功能,并配有系统监控程序清单。

配件:与主机通讯电缆一根,五芯电源线一副。

中国长江轮船总公司 武汉协和祥电子技术开发公司

地址:武汉武珞路 323 号 邮编:430070

电话:(027)713295、725559(宅)

开户行:工行科技小区办 帐号:67—59

联系人:张 伟

汉字文稿打印程序

浙江省海宁市中医院 冯惠民

本刊杂志曾介绍过汉字文稿打印程序,但只能运行于配有 ZH/L310 卡的 LASER 310 机,且程序复杂。经移植并简化后的本程序,由于采用了 STC 软汉字系统,所以能适用于任何型号的 Apple 兼容机。程序中的特殊制表符以及文稿中的标点符号,可以用软汉字系统的造字功能建造;第 10 语句中 D 的赋值,由文稿中汉字数确定。

文稿输入在 DATA 语句中,每句 20 个汉字;文稿中所有非整行空格,须在汉字输入状态下以空字符区位码输入。输入完毕后,最后一条 DATA 语句,请输入制表符“—”,以示结束。

```

10 D$=CHR$(4);D=300;DIM B$(20);PRINT D$ "
   STC B0,VB,L90";CALL 5576
20 P=INT(D/301+1);FOR K=1 TO P;FOR J=1 TO 15
30 READ A$;N=1;FOR I=1 TO 20;B$(I)=MID$(A
   $,N,3);N=N+3;NEXT
40 IF B$(I)="-" THEN GOSUB 100;GOTO 70
50 GOSUB 100;FOR I=1 TO 20;PRINT "|";B$(I);
   NEXT;PRINT "|";PRINT
60 NEXT;GOSUB 100;GOSUB 110;NEXT;GOTO 90
70 L=16-J;FOR M=1 TO L
80 FOR I=1 TO 21;PRINT "|";NEXT;PRINT;GOSUB
   100;NEXT;GOSUB 110
90 PRINT D$"STC B0";END
100 PRINT "□";FOR I=1 TO 39;PRINT "□";
   NEXT;PRINT "□";PRINT;RETURN
110 PRINT "15X20=300";SPC(15);"海宁市中医院";
   SPC(15);"第 ";K;" 页"
120DATA "          汉字文稿打印程序
130DATA "          浙江省海宁市中医院 冯惠民
140DATA "          《电子与电脑》杂志曾介绍过汉字文稿打
150DATA "印程序,但只能运行于配有 ZH/L310 卡的
160DATA "LASER310 机。且程序复杂,经移植并简化
170DATA "后的本程序,由于采用了 STC 软汉字系统,
180DATA "所以能适用于任何型号的 APPLE 兼容机。
190DATA "程序中的特殊制表符以及文稿中的标点符
200DATA "号,可以用软汉字系统的造字功能建造;第
210DATA "10 语句中 D 的赋值,由文稿中汉字数确定。
220DATA "          文稿输入在 DATA 语句中,每句 20 个
230DATA "汉字,文稿中所有非整行空格,须在汉字输入
240DATA "状态下以空字符区位码输入。输入完毕后,最
250DATA "后一条 DATA 语句,请输入制表符"—,以
260DATA "示结束。
270DATA "—"

```

ProDOS 系统内部结构剖析(续)

北京铁路局卫生防疫站(100038) 廖 凯

CLOSE:由文件的输入/输出缓冲区传送任何未写的
数据到文件,并释放文件的输入/输出缓冲区和文件控
制块,必要时,修改文件的目录项目。最后释放文件的
参考号,以便为以后打开的文件所使用。

FLUSH:从文件的输入/输出缓冲区传送任何未写数
据到文件,并在必要时修改文件的目录项目。

SET-MARK:改变文件内当前位置。这当前位置是指
下一个要被读写的字符在文件内的绝对位置。

GET-MARK:送回文件内当前位置。

SET-EOF:改变文件的逻辑长度(文件的结尾)。

GET-EOF:返回文件的逻辑长度。

SET-BUF:为打开文件的输入/输出缓冲区指定一个
新的地址。

GET-BUF:返回一个打开文件的输入/输出缓冲区的
当前地址。

3. 系统调用

系统调用既不是内务调用,也不是文件调用。它是
用于获得当前日期和时间,安装和取消中断子程序,读
和写一个磁盘上指定的块。系统调用有:

GET-TIME:若你的系统有一个计时/时钟卡,并安装
了一个可以由计时卡读取时间的例程,则它在系统日
期和时间存储单元内存入当前日期和时间。

ALLOC-INTERRUPT:放置一个指向中断处理例程
的指针到系统中断向量表中。

DEALLOC-INTERRUPT:从系统中断向量表删除一
个指向中断处理例程的指针。

READ-BLOCK:从一个磁盘读取一个指定块(512字
节)的数据,并放到一个用户指定的数据缓冲区内。

WRITE-BLOCK:将用户指定的数据缓冲区内的一
个数据块(512字节)写到磁盘上一个指定的块内。

四、内务调用

此部分将详细描述每一个内务调用。

1. CREATE(\$C0)

每个磁盘文件除卷目录文件外必须用此调用建
立。文件存储类型有两种:树型结构(存储类型=\$1),
用于标准文件;链接表(类型=\$D),用于目录文件。

路径名指定要建立的文件名和插入一个新文件项
目的目录。

分配1块(512字节)的磁盘空间,并设置指向那

个块的关键指针字段。存取标
记在大部分情况下设为\$C3
(允许全部存取)。

参数个数:此调用有7个
参数。

路径名:这两字节地址指
向一个ASCII串,这串包含一
个计数字节和路径名(最大64
个字符)。若路径名以"/"开始,
则它被当作一个完整的路径
名;否则被当作部分路径名,部
首附加在它的前面以作为完整
的路径名。路径名不被改变。

存取标记:此字节确定这
文件是否可存取。它的格式见
前面介绍。通常存取标记设为
\$C3。若文件是允许删除、改名和写入,则它是解锁
的;若所有三个是禁止的,则它是上锁的。备份位(B)
通常被此调用设置。

文件类型:此字节指示文件的类型。

辅助类型:此字节被系统程序使用,BASIC用它
存储文本文件的记录长度或二进制文件的载入地址。

存储类型:此字节描述文件的物理组织。存储类型
=\$0D表示一个链接的目录
文件;存储类型=\$01表示一
个标准文件。

建立日期和建立时间:此
两字段包含文件建立时的日期
和时间,格式见前介绍。

2. DESTROY(\$C1)

可以通过删去目录内它的
项目来删除经路径名指定的文
件,并返回它的块到卷位图来
删除文件,卷目录文件和打开
的文件不能被删除。

参数个数:此调用有1个
参数。

路径名:同前。

3. RENAME(\$C2)

7 6 5 4 3 2 1 0

0	参数个数=\$7
1	路径名 (2字节)
2	存取标记
3	文件类型
4	辅助类型 (2字节)
5	存储类型
6	建立日期 (2字节)
7	建立时间 (2字节)
8	备份位(B)
9	

7 6 5 4 3 2 1 0

0	参数个数=\$1
1	路径名 (2字节)
2	

7 6 5 4 3 2 1

0	参数个数=\$2
1	路径名 (2字节)
2	新路径名 (2字节)
3	
4	

用新路径名指定的名字来代替经路径名指定的文件的名字。路径名和新路径名的路径必须是相同的,只是最右边的文件名不同(指定文件必须在相同的目录内)。

参数个数:2个参数。

路径名:同前。

新路径名:此两字节地址指示新路径名的位置。

4. SET-FILE-INFO (\$C3)

修改指定文件的项目字段内的信息。这调用可在文件打开或关闭的时候执行。但是新的存取属性不被一个已打开的文件所使用,直到下一次这文件被打开为止(即此调用不修改目前的文件控制块)。

你应该用 GET-FILE-INFO 调用读取一个文件的属性到参数表中,按照需要修改它,然后使用相同的参数表调用 SET-FILE-INFO。

参数个数:7个参数。

路径名:同前。

存取标记:同前。

文件类型:同前。

辅助类型:同前。

空字段:为与 GET-FILE-INFO 调用保持对称而留。

修改日期和修改时间:此两字节包含修改时的日期和时间,格式见前面介绍。

5. GET-FILE-INFO (\$C4)

当需要有关一个卷目录信息时,卷上总块数被送到辅助类型字段,所有文件占用的总块数被送到使用块数字段。

GET-FILE-INFO 返回存储在指定的文件项目字段内的信息,这调用可以在文件被打开或关闭时执行。若 SET-FILE-INFO 调用被用于改变一个打开文件的存取类型时,则对于那个打开的文件返回的存取信息可能无效。

参数个数:10个参数。

路径名:同前。

存取标记:同前。

文件类型:同前。

辅助类型:若此调用用在一个卷目录文件上,则辅助类型将包含卷的总块数。

存储类型:存储类型为 \$0F 说明是一个卷目录文

7 6 5 4 3 2 1 0

0	参数个数=\$7
1	路径名(2字节)
2	存取标记
3	文件类型
4	辅助类型(2字节)
5	空字段(3字节)
6	修改日期(2字节)
7	修改时间(2字节)

7 6 5 4 3 2 1 0

0	参数个数=\$A
1	路径名(2字节)
2	存取标记
3	文件类型
4	辅助类型(2字节)
5	存储类型
6	使用块数(2字节)
7	修改日期(2字节)
8	修改时间(2字节)
9	建立日期(2字节)
A	建立时间(2字节)

件;为 \$0D 说明是一个目录文件;为 \$01、\$02 和 \$03 分别是树苗、幼树和树型文件。

使用块数:这两字节包含文件所使用的块数。如果此调用被用在一个卷目录上,则使用块数字段包含卷上所有文件所使用的块数。

修改日期和修改时间:同前。

建立日期和修改时间:同前。

6. ON-LINE(\$C5)

这调用可以用于确定当前所安装的所有 ProDOS 的卷名(例如在磁盘驱动器内的磁盘),或者它可以用于确定在一个指定的槽口及驱动器内的磁盘名。

在设备号为 0 时,它将指定卷名、槽口号和所有安装磁盘的驱动器号的列表放到由数据缓冲区指针指向的 256 字节的缓冲区。当需要一个指定的设备号时,只需 16 字节保存在缓冲区内。

参数个数:2个参数。

设备号:此字节指定一个磁盘设备的硬件槽口位置。格式是:

7	6	5	4	3	2	1	0
Dr	Slot	未	用				

对于驱动器 1,Dr=0;对于驱动器 2,Dr=1。Slot 指定设备的槽口号(1-7)。若设备号是 0,则所有装配的磁盘被扫描。

下面是设备号可能存在的值:

Slot:	7	6	5	4	3	2	1
驱动器 1:	70	60	50	40	30	20	10
驱动器 2:	F0	E0	D0	C0	B0	A0	90

数据缓冲区:这两字节地址指向一个返回数据的以 16 字节为记录的缓冲区。若设备号为 0,则缓冲区是 256 字节长度;其它情况下 16 字节是足够的。

每一个记录的首字节标识设备和它的卷名长度:

7	6	5	4	3	2	1	0
Dr	Slot	名字长度					

位元 7 指定驱动器 1(Dr=0)或驱动器 2(Dr=1)。位元 6-4 指定槽口号(1-7)。位元 3-0 指定一个有效的名字长度(如果非零)。

记录的后 15 个字节是卷名。

若名字长度为 0,则在指定的槽口和驱动器内发生错误。错误代码被放到记录的第二个字节内。若错误代码为 \$57(重名卷),则第三个字节包含重名的设备号。

在送回多个记录时,最后有效记录跟随一个设备号和名字长度为 0 的记录。

ON-LINE 送回的卷名前面没有斜线。记住你在一个路径名内使用卷名时,要在卷名前面加上一个斜线。

(未完待续)

第八讲 内部结构

北京西四北三条 10 号 FORTH 应用研究会(100034) 丁志伟

在这一讲中,将要深入 FORTH 系统内部,介绍系统的构成、系统词典的结构、词典管理及单词的结构等。

由于每个版本在设计上都有其自身特点,不可能面面俱到,所以这里主要介绍 D1.0 版本,其结构简洁、清晰,便于初学者理解。在方便的时候,也顺便介绍其他版本的一些情况。FORTH 语言在原理上是相通的,以此为基础,再理解其他版本也就容易多了。

前边曾提到过,FORTH 语言有一个特点,比起其他语言来,它透明度较高。象 BASIC 和 C 这样的语言,它们是如何运行的,其内部结构如何,用户一般无从了解,即便能了解,想改动也很困难。FORTH 语言则不然,用户可以在原有基础上对它进行扩充改造,自由度是很大的。与之相应,就要求系统透明度较高。对于较高水平的应用,也要求对系统内部情况有较清楚的了解。

系统的构成。系统包括系统本身以及为之服务的堆栈、输入缓冲区和系统变量等。请看图 1。

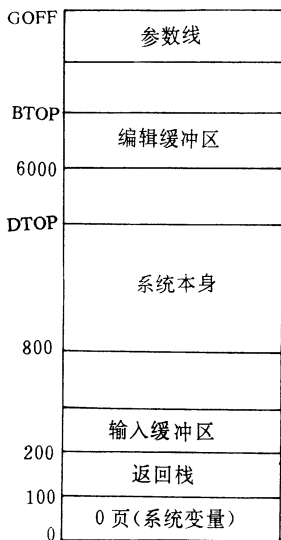


图 1 D1.0 版本的内存安排

先来看看系统本身。一般情况下,它存放在低地址区。对于 D1.0,是从 0800H 开始存放的。这是大多

数 Apple 软件存放的起始地址。

系统本身分为两个部分,请看图 2。

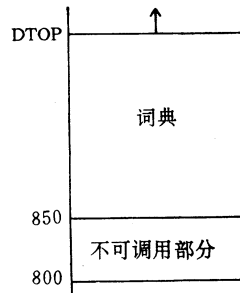


图 2 系统构成

一是不可调用部分。它包含系统起动和初始化程序,运行程序指针处理及少量系统变量等。对于 D1.0 版本,其地址位于 0800H~084FH,只有几十字节。这部分对用户来说是不可调用的。

另一部分就是词典。它是系统的主要部分。词典之中是一个个单词。系统在运行时,就始终运行着其中某些单词。系统对单词的各种操作,都通过词典进行。词典规模不是固定的,一方面,定义单词时,就把新词放进词典;另一方面,也可以删除其中一部分单词。这样,系统的规模是可以扩充和压缩的。

对于 D1.0 版本,词典中第一个词是 DROP,其地址是 850H,这是固定的。词典顶部则是可以变动的。提供给用户的版本,词典顶部一个词是 FBY。对于词典顶部的词,有两个指针,指向其起址和终址,分别存放在名为 DICT(起址)和 DTOP(终址)的两个变量中。

想知道词典顶部地址,就可以打入:

```
DTOP ? ↵
```

显示 2DFD

就是说,从 850H~2DFDH,其中的单词都可以调用,可见系统的空间效率是很高的。

接下来定义一个单词,看看词典顶部位置的变化情况。

```
: TEST! . " HELLO!" CR
```

```
  . " THIS IS TESTI" ;
```

定义之后,再看一下词典顶部地址。

```
DTOP ? ↵
```

显示 2E2B

比刚才有所增长。再定义一个。

```
: TEST2 ." THIS IS TEST2" CR ;
DTOP ? ✓
```

显示 2E4D

又有所增长。

用 LIST 显示一下词典,最先显示出的是 TEST2,随后是 TEST1。可以看出,每次定义都会把新词放在词典顶部。

如果把定义过的词删除掉,词典规模会随之减小。删除是从指定的词到词典顶部的全部单词。这里试试一个个删除。

```
FORGET TEST2 ✓
```

```
DTOP ? ✓
```

显示 2E2B

这个地址与刚才定义过 TEST1 一样。

```
FORGET TEST1 ✓
```

```
DTOP ? ✓
```

显示 2DFD

又回到最初数值。当然,还可以继续删除部分单词。

从图 1 中可以知道,内存中不止是系统本身,还有其他部分。

现在看看堆栈。FORTH 是双栈系统,它有两个堆栈。一个是参数栈,用于存放各种参加运算的参数;另一个是返回栈,用来存放单词的返回地址,也用来存放循环变量和临时存放一些数据。

D1.0 版本是在 CEC-I 中华学习机上开发的,在 CEC-I 机上,9200~94FFH 是汉字显示映射区。参数栈栈底安排在 90FFH,这个栈是向下增长的,有数据压入,栈顶就向下移动。词典中有一个词 SP?,是用来查看栈顶位置的。栈空时打入:

```
SP? ✓
```

显示 90FD

```
1 2 3 ✓
```

```
SP? ✓
```

显示 90F7

堆栈中压入了 3 个数,每个数占两个字节,由于向下增长,栈顶就变成 90F7。再打入:

```
DROP DROP ✓
```

```
SP? ✓
```

显示 90FB

丢掉两数,栈顶指针位置向上增长。

如果不断向栈中存入数据,堆栈增长过多,就有可能把其它部分冲掉;相反,如果不断从栈中取数,把栈中数取空之后,就有可能向栈底以下(这里是更高的地址)取数。这两种情况都是错误。使用 FORTH 系统时,如果能随时注意栈中状态,编写单词注意保持栈平衡,一般就不会出现严重错误。这里介绍一个技巧:打入一个错误命令,即词典中不存在的词名,系统就会显示出相应的错误信息,同时清除堆栈,把栈顶

指针重新置成 90FDH。比如连打两个分号,然后回车,分号键与回车键接近,打起来比较顺手。这种技巧在其他 FORTH 系统也适用。

返回栈用来存放单词的返回地址。这种存放是自动的,用户看不见。返回栈位于 100H~1FFH。熟悉 6502CPU 的读者会发现,这是 CPU 堆栈所用空间。实际上也正是这样,系统利用了 6502 对堆栈进行操作的指令来处理返回栈的操作。

能与返回栈打交道的单词不多。除了循环中使用的 I 和 J,还有 <R 和 R>,及几个不太常用的词,这里介绍一下 <R 和 R>。

```
<R ( n-- )把参数栈中数存入返回栈。
```

```
R> ( --n )把返回栈中数取进参数栈。
```

<R 用来临时把参数栈中数存入返回栈,便于某些处理,这个词在其他系统中,词名可能是 >R。R> 则相反。这两个词一般应该成对使用,并且在冒号定义中用。举个例子,比如要计算 (a+b)/c,可以定义如下:

```
: TEST3 ( a b c--(a+b)/c )
```

```
<R + R> / ;
```

先把栈顶的 c 放到返回栈,计算 a+b 之后,把 c 取回,再做除法。如果不用 <R 和 R>,可以这样定义:

```
: TEST4 ( a b c--(a+b)/c )
```

```
ROT ROT + SWAP / ;
```

动作会复杂一些,也不那么直观。

系统还用两个缓冲区。一个是编辑缓冲区,一个是输入缓冲区。

编辑缓冲区的起始地址是 6000H,其终址随着其中内容多少而变化。其起止、终址分别存放于 BUF 和 BTOP 两个变量中。关于编辑缓冲区,请参考第 5 讲。值得注意的是,参数栈是从 90FFH 向下增长,而编辑缓冲区是从 6000H 向上增长,二者不断增长,就有可能碰到一起,这时就要出乱子。当然,一般情况是不会碰上的。

输入缓冲区是从 0200H 开始的,这也是 Apple 机其他软件系统常用的缓冲区位置。实际上,D1.0 版本正是调用了监控输入子程序。

在 0 页中,还存放着一些系统变量。有些是系统临时变量,有些是系统指针,还有一些是用词典中常数做为指针的变量。比如 HT、VT,是词典中两个词,它们都是单字节常数,其中存放着屏幕显示横座标和纵座标,分别指向 024H 和 025H。

以上介绍了 D1.0 版本的内存安排情况。对于其他版本,其内容、位置都会有所不同,并无一定之规,要查阅说明书才能了解。

下面重点介绍一下词典结构。

已经知道,系统的主要部分是词典,用户与系统打交道也主要是通过词典进行的。其中是一个个称之为单词的功能模块,系统的性能就取决于这些单词。单词是按顺序放在词典中的,其结构非常简单,请看图 3。

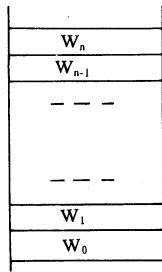


图3 词典结构示意图

词典中的单词是 W_0, W_1, \dots , 一直到 W_n , 共 $n+1$ 个单词。如果定义一个新词, 比如前边的 TEST1:

```
: TEST1 ." HELLO!" CR
  ." THIS IS TEST1" ;
```

定义之后, 这个词就放在原来的 W_n 之后内存中, 这在前边已经介绍过了。只要内存够用, 就可以继续定义下去。在 D1.0 中, 词典上限一般是 6000H, 也就是编辑缓冲区的地址。如果还想使词典大于 6000H, 在编辑缓冲区空时, 可以把 BUF 修改一下。比如可以:

```
HEX 7000 BUF ! CLR
```

这样就把编辑缓冲区起址改为 7000H 了。当然, 这样做之后, 编辑缓冲区就比原来小了。

FORTH 的词典结构有些象堆栈, 新定义的词放在其顶部, 删除也是从词典顶部进行的。从这个角度看, 词典又可以看做是一个“词栈”。在某些版本的编辑器中, 还有所谓“行栈”和“屏栈”。FORTH 系统与堆栈概念是息息相关的, 可以说, 如果没有堆栈, 就没有 FORTH。

假如只是简单地把一个个单词放在词典中, 那还不够。系统还需要对这些词进行各种操作。比如定义、删除、执行及编译等, 都要求能方便、迅速地检索到各个单词才行。

FORTH 中, 检索是通过链表进行的。下面介绍一下链表是怎么回事, 请看图 4。

每个单词中有一部分称为链域, 用英文表示是 LFA。其中存放着前一个单词的起址。在 D1.0 中, LFA 位于一个单词的起址, LFA 中的数据也就是前一个单词 LFA 的地址。 W_n 的 LFA 中存放着 W_{n-1} 的起址, W_{n-1} 的 LFA 中又存放着 W_{n-2} 的起址。这样, 一个接一个, 一直接到词典中第一个词, 而第一个词的 LFA 中存放的是 0, 表示到此为止。这种链域一个连接一个的机制, 称为链表。

这样, 只要知道最后一个词的地址, 就可以通过链表查到其他词。比如, 列出词典, 是用 LIST; 查找一个词, 是用 GETVERB; 删除一个词, 是用 FORGET, 等。还有其他直接、间接与这些词有关系的词, 都通过链表对词典进行操作。这类操作, 又可称为对词典的管理。如果没有链表, 这种管理就不可能了。因此有人

说, 链表是 FORTH 系统的生命线。

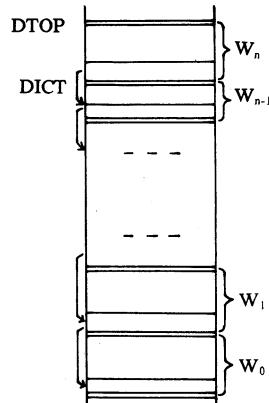


图4 链表示意

在上一讲, 介绍过一段 LIST 的简化程序, 这里再列出如下, 请读者从新的角度分析一下。

```
: LIST DICT
  BEGIN @ DUP
  WHILE DUP ID. CR
  REPEAT DROP ;
```

ID. (a--)按表中给出的单词地址, 显示出一个词的词名。

DICT 是词典中最后一个词的起址, 在这个循环中, 不断地把一个个词的链域内容取出来, 检查其是否等于 0, 显示出词名, 一旦遇到第一个词, 其 LFA 中为 0, 循环就会终止。这样, 执行 LIST, 就会把词典中各个词都显示出来。

以上介绍了 D1.0 系统的链表, 这是所谓单链结构。这样的结构, 符合 FORTH 创始人摩尔倡导的简明的风格, 很适合于较小的系统。

对于很多其他 FORTH 版本, 出于种种考虑, 设计成多链系统。把一个系统分为基本部分和若干个专用部分。每部分含有一组词, 叫做一个词汇支, 词汇支内部用一个独立的链表连在一起。另外, 还有多种其他各具特色的词典结构, 限于篇幅, 这里就无法介绍了。

接下来看看 D1.0 中单词的结构。

词典中的单词, 分为词头和词体两部分。请看图 5。词头位于低地址, 词体位于高地址。

词头包括链域 LFA 和名域 NFA 两部分。词典中每个词的词头在结构上都是一致的, 其好处是系统可以对词典中全部单词进行统一管理, 也可以使词典构成方式简化。

关于 LFA, 前边介绍过, 其作用是构成链表, 它位于一个词的起始位置, 占两个字节, 其中存放的是上一个词的起址。

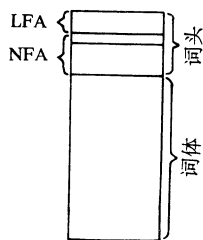


图 5 单词结构

NFA (名域) 用来存放这个词的词名, 分为两部分。第一部分占一个字节, 它的低 4 位用来表示这个词的词名长度。这个字节的最高位表示这个词的性质, 一般是 0, 如果是 1, 则说明这个词是“立即词”, 又称编译词, 在冒号定义之中它会立即执行, 供编译时做特殊用途。严格地说, 这个最高位不属于 NFA, 就不多介绍了。NFA 的后几个字节, 存放着词名各字符的 ASCII 码。它的长度是由词名长度决定的。比如 LIST 这个词是 4 个字符, 它就占 4 个字节, 加上一个长度字节, LIST 的 NFA 共占 5 个字节。

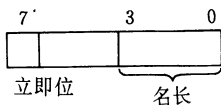


图 6 NFA 的第 1 字节

词名的最大长度是有限的。D1.0 版本中最大长度是 15, 其他系统中可能是 31。

知道了 NFA 的地址, 就可以用 ID. 把这个词名显示出来。比如 DROP 的 NFA 是 850H,

```
HEX 850 ID. ✓
显示 DROP
```

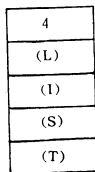


图 7 LIST 的 NFA 示意

列出词典 (LIST)、反编译 (FBY) 等都需要把词名显示出来。

再来看看词体。词体是单词的可执行部分。一个词具有什么功能, 是由词体决定的。各个单词的词体一般来说是不一样的。

词典中的单词分为几个类型, 分别是常数、变量、用汇编定义的词和用冒号定义的词。它们的词体各具特点。

常数。一个常数的词体前 3 个字节是一条 JMP 指

令, 转向名为 CON@ 的单词去执行。CON@ 的功能是把其后的数取出放进堆栈。

变量与常数结构基本一样, 所不同的是, 其中的 JMP 指令是转到 VAR@ 单词。VAR@ 与 COR@ 的功能不同, 因而变量的功能与常数不同, 它把这个变量的地址放在堆栈上。

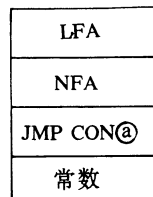


图 8 常数

汇编的词体是一段机器码。最后一条一般要转回到系统的地址指针处理程序去执行。这里以 DROP 为例, 看看汇编的程序是什么样的。这段程序很简单。

```
$ 857 LDA #2
      CLC
      ADC $ 62
      STA $ 62
      BCC $ 862
      INC $ 63
$ 862 JMP $ 808
```

程序功能是移动栈指针。多数汇编词比这个词复杂, 但规模一般不大, 读者感兴趣, 可用 FBY (反编译) 查看。

冒号定义的词。系统中多数词都是这类词, 它们也是 FORTH 中最具特色的一类词。

冒号定义词中前三字节是一条 JMP 指令, 转向 ENTER 词。之后的内容一般是一条条单词的执行地址。最后一个地址是 EXIT 词的执行地址。由于其中是一条条地址, 因此称为地址流。懂英文的读者都知道, ENTER 和 EXIT 分别是进入和退出的意思。

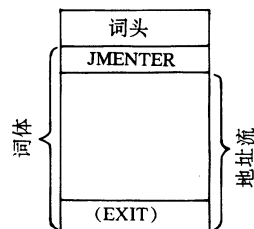


图 9 冒号定义词

比如问号词“?”, 它的功能是把一个地址中的内容显示出来。其定义是

```
: ? @ U. CR;
```

词头
JMP ENTER
G3F(@)
IA0A(U.)
1515(CR)
88A(EXIT)

图 10 问号词结构

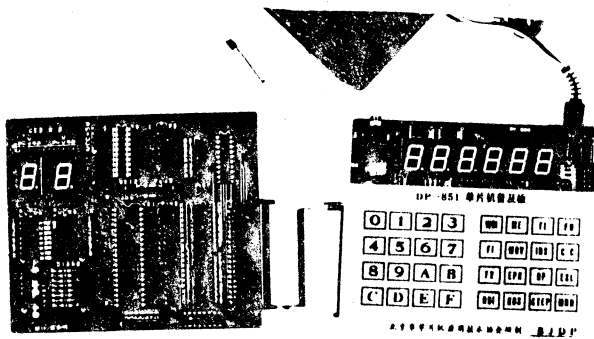
其词体的前三个字节,转向 ENTER 词,这是冒号定义词之中的例行入口词,类似于子程序。它把返回地址压入返回栈,然后取出一条条地址转去执行,这里就是问号词中的@、U.、CR 的各个执行地址,最后遇到 EXIT 这个词的执行地址,就按返回栈中的地址返回到调用它的地方。

如果冒号定义中的词是以前用冒号定义的,这个内部的冒号定义词的执行情况就会重复上述介绍的过程执行,然后返回。就象是子程序中调用子程序。

这一讲介绍了 D1.0 版本中系统的一般情况,还有很多有关内容没有涉及到。其他版本也各有其自身特点。要想深入了解,需要结合具体版本及说明书,上机反复研究才行。

DP-851 单片机普及板

结构:



特点:

1. 开发系统板:主板+键盘板(或主板+PC机)可构成单板型单片机开发器,主板上 16KB 的监控程序(键盘和 PC 机各占 8KB),具有置入,修改,调试,运行等多种开发功能。
2. 教学实验板:开发器+实验板可构成教学实验板,该板从教学实际需要出发,具备了实验内容多,覆盖范围宽,结构灵活,性能可靠,价格低廉等优点,是一种大众型的教学实验设

备。

3. 目标应用板:开发器+A/D,D/A 等扩展板可构成用户的目标系统。

4. 基本系统:薄膜键盘板;主板;实验板;说明书;电源;工具;包装盒;40 芯电缆,

5. 报价:398.00/套

培训及比赛:

由中国计算机学会和北京单片机应用技术协会面向全国开展函授教学,计算机学会将于九四年底用 DP-851 单片机普及板进行全国学装微电脑比赛,

TP-STD8000 系列工控机模板

具有可靠性高,抗干扰性强,模板种类丰富,组成灵活,易于扩展,维修方便等特点,被广泛用于过程控制、自动控制/检测、自动化仪表、生产自动化等领域。

TPμP 系列微型打印机

具有体积小,重量轻,功能强的特点,被广泛用于各种领域。

公司还经营各种开关电源,元器件,微型计算机等。

单位:北京海声工业控制系统工程公司

电话:256.2231 255.4603

地址:北京海淀区中关村路口南北京 8703 信箱

邮编:100080

联系人:程刚

开户行:工商银行海淀区双榆树信用社

帐号:003148-28

C 语言中的字符串处理及指针变量 在字符串处理中的应用

北京航空航天大学计算中心(100083) 李宁

0 前言

对字符串的基本操作一般指左对齐(内容靠左放置)、对中(内容居中放置)、右对齐(内容靠右放置)、计算字符串中字符的个数和截取一个字符串中的某一部分;对字符串的高级操作一般指对字符串进行插入,查找,删除和替换以及比较两个字符串。这些操作组成了对字符串进行处理所需要的大部分功能。

在高级程序设计语言中,一般都提供一组函数使用户对字符串的处理变得较简单。从学习程序设计方法的角度来说,若能了解用程序设计语言是怎样编写出这些函数的,则不但程序设计水平会有提高,而且对程序设计语言的掌握也会更熟练。这里介绍一下用 C 语言(使用 Turbo C 2.0 版)编写处理字符串的函数。

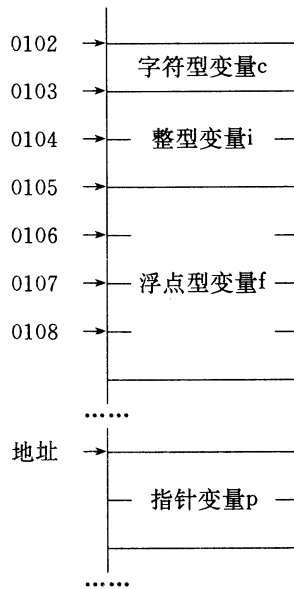
若用 C 语言编写处理字符串的函数,则最好使用指针,对字符串的操作也是表现指针优越性的重要方面。C 语言的灵魂在于指针的运用,如果编程者对指针使用得当,就能编出清晰、灵活、高效的程序;如果使用不当,其危害对运行中的程序和操作系统都将是致命的。不幸的是,初学者学习使用指针的过程通常不是一帆风顺的,所以本文作者希望在处理字符串和使用指针这两方面都对读者有所帮助。

1 指针

在 C 语言中,任何变量的值都存储在由编译程序分配的存储单元中,存储单元的大小取决于变量的类型(Turbo C 2.0 版规定,字符型变量占 1 字节的存储空间,整型变量占 2 字节的存储空间,浮点型变量占 4 字节的存储空间,等等)。所有的存储空间都有一个统一的以字节为单位的编号,这就是存储空间的地址,每一个存储单元都有相应的一个起始地址。若某存储单元存储了某个变量的值,这个存储单元的起始地址就是这个变量的存储地址,简称变量的地址。

请看图 1,上面画了某段存储空间,每一格代表一个字节,各个字节的地址分别为 0102 到 0108。若在地址为 0102 的地方存储一个字符型变量 c 的值,则 c 的地址就是 0102。若在地址为 0103 的地方存储一个整型变量 i 的值,则 i 的地址就是 0103,同时地址为 0104 的字节也被 i 占了(一个整型变量占 2 字节)。同理,若在地址为 0105 的地方存储一个浮点型变量 f 的值,则 f 的地址是 0105,它要占从 0105 到 0108 的 4 字节的空

.....



指针变量简称指针,是一种特殊的变量。作为变量,它有自己的地址,但是存储在这个地址处的内容(即指针变量的值)仍是一个地址。还看图 1,设地址 1 为指针 p 本身的地址,若地址 1 处的内容是 0102,则由于地址 0102 处存储的是一个字符型变量的值,那么说 p 是指向字符型变量的指针。同理,若地址 1 处的内容是 0105,则 p 是指向浮点型变量的指针。通常,我们并不关心指针本身的地址,而是关心它所指向的目标或指向的位置。

在 C 语言中,“int * p;”说明 p 是指向整型变量的指针,简称 p 是整型指针,* 说明 p 是指针变量,int 说明 p 是指向整型变量的;同理,“char * s;”说明 s 是指向字符型变量的指针,简称 s 是字符型指针。这些说明语句只说明了指针是指向哪种变量类型的,不能说明指针具体指向了哪个变量,同一类型的变量可以有很多,必须向指针赋以某一变量的地址才能使指针指向该变量。在 C 语言中,用单目运算符 & 作用于变量取得变量的地址。这样,“p=&i;”就把变量 i 的地址赋给了 p,使 p 指向变量 i。注意,i 的类型与 p 所指向的类型必须是一致的(有例外,这里不讨论),p 可以再指向同一类型的其它变量,但不能再指向其它类型的变量。在 C 语言中,用单目运算符 * 作用于指针取得指针所

指目标的值(指针所指地址处的内容)。设 i 的值是 7, 则用 i 和用 $*p$ 去参加运算是等价的, 都是使用 7 这个值。因此, 单目运算符 $\&$ 和 $*$ 是互逆的, 但作用对象有区别, $\&$ 可作用于各种类型的变量(包括指针变量), $*$ 仅作用于指针变量。

由于指针的值是地址, 所以一般对指针只做 4 种运算: (1) 取内容(单目 $*$); (2) 加一个整数(得另一地址); (3) 减一个整数(得另一地址); (4) 两个指针相减(得一整数)。指针加 1 的意思是使指针的值(地址)增加一个单位量, 单位量随指针所指向的变量类型的不同而不同。在 Turbo C 2.0 版中, 一个整型变量占 2 字节的存储空间, 所以一个整型指针加 1 意味着它的值(某整型变量的地址)实际上加了 2。同理, 字符型指针减 1 实际上地址就是减 1。用户在程序中只管写加 1, 实际上加了几是编译程序根据用户说明的指针所指向的变量类型自动进行的。从这里也可以理解为什么一个指针只能指向一种变量类型。若指向的目标的类型不定, 编译程序就无法确定地址的实际增减量。若指针变量不是加 1 而是加 3, 实际上地址就加了三倍的单位量。指针加上(减去)一个整数, 就是使指针所指的位置后移(前移), 所以也叫做“移动指针”。两个指针相减得到的是它们所指的位置之间相差多少个字节, 若得负的, 则前一个指针所指的位置在前, 这在处理某些问题时也是常用的。

对图 1 中的例子, 执行语句“ $p=\&i;$ ”后, 整型指针 p 的值就是 0103, 指向 i ; 再执行语句“ $p=p+1;$ ”后, p 的值就成了 0105, 现在 p 指向哪呢? 指到了一个浮点型变量上了。这时再用 $*p$ 取 p 所指处的内容就发生了错误, 它只能按整型变量的规定取出 0105 处前两个字节的內容, 这是一种动态错误。在程序执行过程中不恰当地移动指针往往造成指针没指到该指的地方, 因而取出的数据完全不对, 送入的数据则会覆盖了不应被冲掉的数据, 这种错误一般还很难查出, 因此使用指针需要较多的编程经验。

指针和数组的关系很密切, 数组是在存储空间中连续顺序排放的同一类型的一组变量, 它们有个共同的名字(即数组名), 可以通过序标来使用其中的某一个(即数组元素)。在 C 语言中, 数组的序标从 0 开始, 数组名等价于数组第一个元素的地址。“ $\text{int } a[20];$ ”说明了一个整型数组, 数组名 a 就是一个指针, 它指向存储 20 个整型变量的一段连续存储空间的起始位置, 不过它是指针常量, 试图对其赋值是不允许的。对数组元素的使用除了用序标外也可以通过数组名(首地址)来进行, 第 $n+1$ 个元素可写为 $a[n]$, 也可写为 $*(a+n)$, 只要 $0 \leq n \leq 19$, 指针 $a+n$ 就移不出数组 a 所占据的空间, 并总是指向一个整型的数组元素。

2 字符串处理

在 C 语言中, 字符串是以“空字符”(写为 ‘\0’ 或 NULL, 即 ASCII 码 00)结尾的若干个连续的字符。字符串可以用字符数组或指向字符串第一个字符的指针来实现。当用“ $\text{char } s[4];$ ”定义字符串时, 编译程序开

辟了一段最多能存储 3 个字符(加上末尾的 ‘\0’ 共 4 个)的连续存储空间, 字符数组名指针 s 指向这段空间的起始位置; 当用“ $\text{char } *t=\text{“ABC”};$ ”定义字符串时, 编译程序也开辟了一段长 4 字节的连续存储空间, 字符型指针 t 指向这段空间的起始位置, 并且这段存储空间中已经放好了 ‘A’, ‘B’, ‘C’, ‘\0’ 这 4 个字符。注意, s 与 t 的不同在于 s 不能再被赋值而 t 可以。编程者应保证在用指针对字符串进行操作的过程中, 指针所指的位置不超出字符串长度空间的限制。

例 1. 计算字符串中字符个数的函数 $\text{scnum}()$, 个数不包括末尾的 ‘\0’。

函数 $\text{scnum}()$ 的形参 s 为指向某一字符串的指针。在 $\text{scnum}()$ 中, 首先定义了一个字符型指针 p , 它先指向 s 所指的位置。while 循环用于将 P 所指的位置移到字符串末尾。当 p 已经指向了字符串末尾时, 所指处的内容等于 ‘\0’, while 循环的条件为假, 因而退出循环。此时指针 p 与 s 的差就是 p 与 s 所指位置之间的字节数。因为每个字符占 1 字节的存储空间, 所以 $p-s$ 就是字符串中字符的个数。

```
/* 计算字符串 s 中字符的个数 */
int scnum(char *s)
{ /* 个数不包括末尾的 '\0' */
  char *p;
  p=s; /* 指向 S 的第 1 个字符 */
  /* 用循环移动指针 p, 直到指向末尾 */
  while(*p != '\0') p++;
  return(p-s); /* 返回字符个数 */
}
```

例 2. 将字符串内容靠左放置的函数 $\text{slefj}()$ 。

若想去掉一个字符串开头的空格, 最省事的办法是用函数 $\text{slj}()$ 。此函数中用 while 循环使原指向字符串第 1 个字符的 s 跳过前面的所有空格而指向第一个不是空格的字符, 此时从 s 开始的字符串就是所要求的无前导空格的字符串, 返回 s 即可。但这样做有一个缺点, 即返回的字符串比原字符串的长度缩短了, 而我们不能在字符串的末尾加空格补齐, 因为多加的空格很可能超出原字符串的存储空间而写入其它数据区, 这种越界行为往往造成程序动态错误, 甚至死机。

```
char *slj(char *s)
{
  while(*s == ' ') s++;
  return(s);
}
```

函数 $\text{slefj}()$ 是对 $\text{slj}()$ 的改进, 它利用两个指针 p 和 q 将字符串中的非空格字符依次向前搬动。

```
/* 将字符串 s 的内容靠左放置 */
char *slefj(char *s)
{ /* 字符串 s 的长度保持不变 */
  char *p, *q;
  p=q=s; /* 指向 s 的第 1 个字符 */
  /* 用循环移动指针 q, 指向第 1 个非空格字符 */
  while( (*q == ' ') && (*q != '\0')) q++;
  /* 若 q 不指向第 1 个也不指向末尾, 则搬动 */
}
```

```

if((p! =q)&&(*q! = '\0'))
while(*q! = '\0')
* (p++)= *q,
* (q++)= ' ';
return(s);
}

```

例 3. 将字符串中从第 i 个字符开始的个数为 j 的子字符串复制到字符串 t 中的函数 ssubs()。

C 语言中调用函数时,参数传递的规则为“按值传送”,简称传值。即在函数内部为每一个形参分配一个存储单元,调用函数时将实参的值复制给形参,用形参去参加函数内的运算,形参值的变化不会影响到对应的实参。调用结束时不保留形参的存储单元,因而其值也被丢弃。当形参为字符指针时,复制给它的将是某字符串的地址,因此它就与对应的实参一样指向了该字符串。显然此字符串的存储单元在函数之外,调用结束时,对形参所指处的内容所做的修改就会被保留下来。在 C 语言中,使用指针是使函数的处理结果在函数调用结束后保留下来的一种方法。

在函数 ssubs() 中,指针 p 首先指向 s 的第 i 个字符,指针 q 首先指向 t 的第 1 个字符。通过 j 逐次减 1 的循环将相应的字符依次复制到 t 里。按传值规则,形参 j 的值虽然改变了,对应的实参将不受影响。

```

/* 将字符串 s 中从第 i 个字符开始的个 * /
/* 数为 j 的子字符串复制到字符串 t 中 * /
char * ssubs(s,i,j,t)
char * s, * t;
int i,j;
/* 调用前要保证 t 的长度大于 j * /
char * p, * q;
p=s+i-1; /* 指向 s 的第 i 个字符 * /
q=t; /* 指向 t 的第 1 个字符 * /
/* 用循环依次复制字符 * /
for(;j>0;j--)
* (q++)= * (p++);
* q= '\0'; /* 加上末尾的 '\0' * /
return(t);

```

```

}

```

例 4. 比较两个字符串前 i 个字符的函数 scomp() ,若两个字符串相等,则返回 0; 否则返回第一对不等的字符的差。

因为字符串的末尾有一个 '\0', 所以用比较法在某字符串中寻找一个子字符串时会发生点小麻烦(想一下为什么), 因此在定义 scomp() 时规定只比较两个字符串的前 i 个字符。

在 C 语言中没有专门的逻辑量, 所有非零值都可作为真, 而零值作为假, 所以 p 没指在字符串末尾这个条件既可用“*p! = '\0'”也可用“*p”来表示。

在 scomp() 中用循环控制逐一比较各对字符, 当既在 i 个字符之内, 又没到字符串的末尾, 且两个对应字符相等时, 就移动指针以便比较下一对字符。

```

/* 比较两个字符串 s 和 t 的前 i 个字符; 若两个 * /
/* 字符串相等, 则返回 0; 否则返回第一对不 * /
/* 等的字符的差 * /

```

```

int scomp(char * s,char * t,int i)
{
while( (i-1)&&(*s )
&&(*t )&&(*s== *t))
i--,s++,t++;
return(*s- *t); /* 返回字符的差 * /
}

```

3 思考题

- (1) 编将字符串内容靠右放置的函数 srigi()。
- (2) 编将字符串内容居中放置的函数 smidj()。
- (3) 编在字符串 s 中查找字符串 t 的函数 ssearch()。
- (4) 编在字符串 s 中删除字符串 t 的函数 sdelete()。
- (5) 编将字符串 r 中的子字符串 s 用字符串 t 替换掉的函数 srepl()。
- (6) 编在字符串 s 中第 i 个字符之前插入字符串 t 的函数 sinse()。
- (7) 四个例子中的五个函数在不改变其功能的前提下均可再精简一下程序的写法, 请根据自己对 C 语言的掌握程度试一下。

“电子技术与维修大专函授班”首届招生

为培养具有较高理论水平和熟练技能的电子技术维修人员, 为城乡电器维修专业户和大批经过自修、培训的青年人提供深造的机会, 为培养青年人谋求职业或第二职业的本领, 天津广播电视大学、《电子维修与制作》杂志编辑部、北京现代家用电器协会联合举办电子技术与维修函授大专班。

函授班设置“工业电气设备原理与维修”、“办公自动化设备原理与维修”、“家用电子电器设备原理与维修”三个社会急需人才的热门专业, 均按大专水平的内容进行培训。学制二年, 学习 12 门课程。聘请大学教授

主讲, 辅导, 按教学计划分期指导, 批改作业与答疑, 学期末邮寄试卷考试。完成所设课程学习、考试合格者颁发盖有天津广播电视大学钢印的结业证书, 与家电维修相关专业同时颁发全国家电维修统一证书, 作为从业登记文凭。学员可跨专业选修课程。

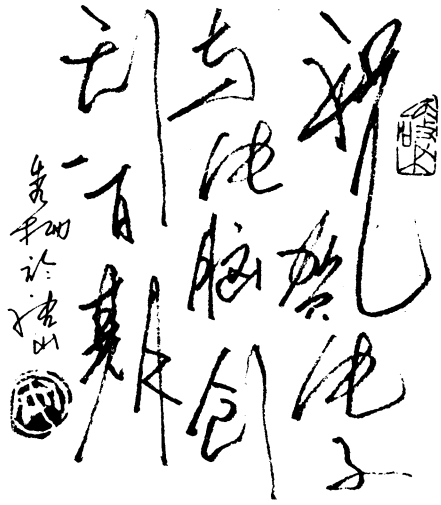
为确保教学效果和教育质量, 成立了以原电子部副部长、中国电子学会理事长为主任委员的教学指导委员会, 其中委员有陈可军(电子部系统工程发展研究中心副主任)、姚志清(电子部人事教育司副司长)、陈秋华(天津广播电视大学副校长)、荫寿琪(北京现代家

用电子电器协会秘书长)、柳维长(北京计算机技术研究所所长),陈太一(中国人民解放军军事通讯工程学院教授);同时成立了以陈太一教授为主任委员的教学委员会,其中委员有柳维长、贺子修(建工学院机电系主任、教授)、韩广兴(天津广播电视大学教授)、耿文学(北方交通大学教授)、申世璋(北京大学教授)和刘莹(电子部系统工程发展研究中心编辑)。

凡具有中等以上文化水平的电子维修专业户、企事业单位的电工及维修人员、军地两用人才、社会青年、农村青年、应届毕业生及电子爱好者均可报名。报名截止日期为93年12月31日。收费标准:报名注册费60元,每学年学费(含教学、教材、辅导教材、考卷、办公及邮寄费等)300元。第一学年共计收费360元。本部收到学费后即将收据、录取通知书、学籍号等资料邮给学员,所用教材及辅导材料分期寄发。一经注册不再办理退学手续。

报名者请通过邮局汇款到北京市海淀区永定路123号(100039)《电子维修与制作》杂志编辑部收。请

注明姓名、详细地址、邮编和学习专业。简章备索。联系人:刘莹。



模块化结构的单片微机普极型开发、应用系统 HCH—51 单片单板机

HCH 单片微机开发、应用系统是具有很强的自开发功能的单片机用户系统,系编根据用户系统中可能需要的测控技术要求,将系统主板设计成通用型模板,三总线全部引出,由用户任意扩展,并配有能适应各种不同测控系统要求的功能模块,广泛适应不同用户的要求,克服了单片单板机固定结构不能获得最佳配置的缺陷,系统配置可由用户根据需要任意调整,使系统在实际中的应用/配置比接近于1,提高了系统性/价比,这样就使本系统既适应于小批量或单台产品的开发与应用,也能适应于各企业大批量生产中微机控制系统的选型。

本系统具有很强的开发、应用功能,程序可通过通用个人微机编译后,经串口传送到单板机,键操作方便,并配有相应开发软件,具有汇编编辑、编译、反汇编、双向通讯等功能,能使用户进行高效率的软件开发。用户程序调试空间16K,还可在64K范围内扩展,系统配有强功能的监控软件,硬件诊断功能完善。开发系统在独立工作时代可靠地保护用户重要数据不丢失,即具有PAM掉电保护。用户程序在调试时可单步运行和断点运行,也可连续全速运行并可进行状态查询。系统的监控软件仅占4K地址空间,其余可通过仿真头全部引出由用户分配,系统主模板也按通用用户系统的要求设计了较齐全的外设,可供用户选择使用。

HCH 开发、应用系统的应用领域相当广泛。

系统用作开发工具时,可在线调试用户系统,然后用户系统可离线运行。

系统在用作用户系统模板时分单机和多机应用。

本系统单机应用主要在于各种智能仪表、工业测控系统、数控系统,家电产品数字化以及各大专院校的教學选型用本系统改造,开发各种测控仪表,就能在仪表的数字化、智能化、多功能化上表现出很大的优越性,能集数据采集、数据处理、控制算法、控制输出等功能于一体,能极大地扩广采集的点数,能实现多点巡检、监测,最大巡检点数为64点(附巡检卡)系统本身配有巡检通用子程序,检测点数可在64以内任意选。系统主板同时提供了模拟量的输出通道,使智能仪表具备控制功能、主板上的用户并行口可直接连微型打印机,系统配有TP系列微打的驱动器程序,故用户使用本系统作智能仪表,既能作为开发工具,又能直接使用,软、硬件二次开发工作量小,周期短。

在工业测控系统和由于系统前、后向通道硬、软件功能齐全,用户接口方便,故广泛用于交、直流调速系统,生产流水线自动控制,温湿度测控系统等等,大幅度降低了成本,提高了微机系统的抗干扰性和工作可靠性。

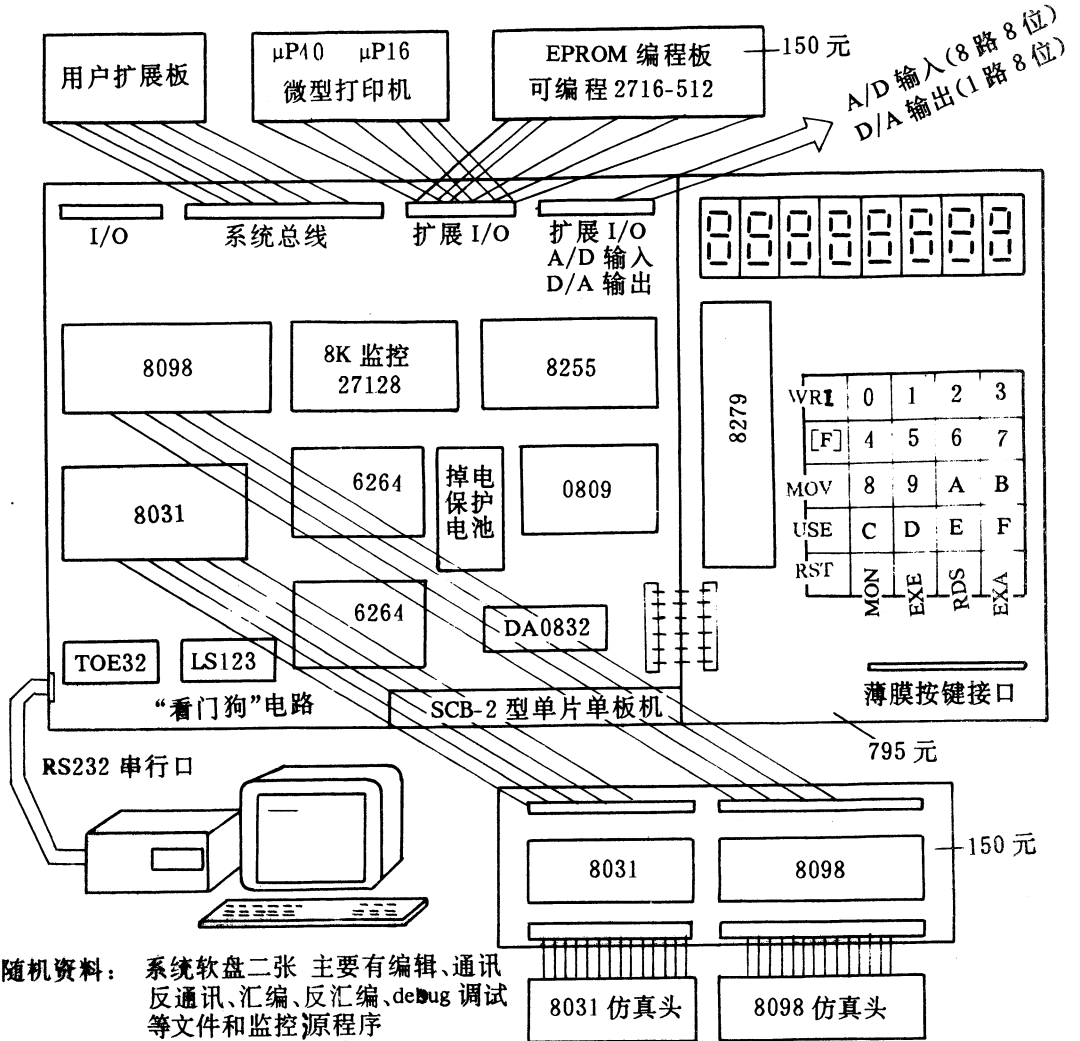
用本系统改造由Z-80单板机系统构成的简易数控系统,能有效缩短软件,提高运行速度,提高系统采集控制精度以及工作可靠性,在工业现场应用时,提高了微机系统抗干扰性,并不须再配开发装置,系统就能直接开发应用。

系统在多机应用时主要用于功能弥散系统和并行多机控制系统。

由于本系统软、硬件资料全开放,系统设计中均采用大专院校教材中通用接口器件及接(下转第21页)

CSB 系列单片单板机

应用: 数学自学仿真开发工控仪表 (51.98 两用)



随机资料: 系统软盘二张 主要有编辑、通讯反通讯、汇编、反汇编、debug 调试等文件和监控源程序

《单片机原理及制作》…… SCB-1 软硬件详解
 《SCB-2 型单片机详解》…… SCB-2 软硬件详解
 另选配《SCB 单片单板机实验教程》

产品开发单位:

产品生产单位: 武汉尚吉电子研究所

产品销售单位: 全国各地经销代理点

地址: 湖北武汉武昌珞瑜路 37-3 号

邮编: 430070

电话: 716138 716692

电挂: 4561

帐号: 2604-67-016072001

图文传真: (027) 716138

开户银行: 工商银行洪山区办

联系人: 单部

深圳办事处地址: 深南中路 75 号科技商场 47 号柜

联系人: 李秋华 罗胜辉

微机打印机共享器原理与设计

深圳大学电子系 朱明程 杜志明 李力

DAKE 微机打印机共享器是我们新近设计开发的一种实用电脑辅助用具,具有使 2 台及多台电脑全自动共享一台打印机的功能。该电路设计成熟,兼容性能好,具有自动排队功能,免去人工选择开关,使用方便,造型精巧,具有很好的性能价格比和实用价值,可广泛用于企业,事业,银行,商业等电脑室及大专院校电脑实验机房。为了进一步推广新技术,本文将微机打印机共享器原理及电路设计介绍如下,以飨读者。

一、微机打印机共享器原理

打印机与电脑的通信接口有两种,一种为串行,另一种为并行。DAKE 微机打印机共享器基于并行通信方式,其原理如下图(1)所示。

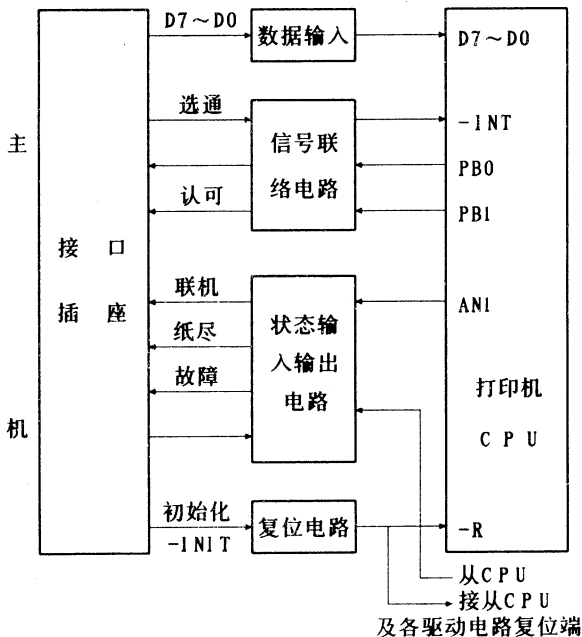
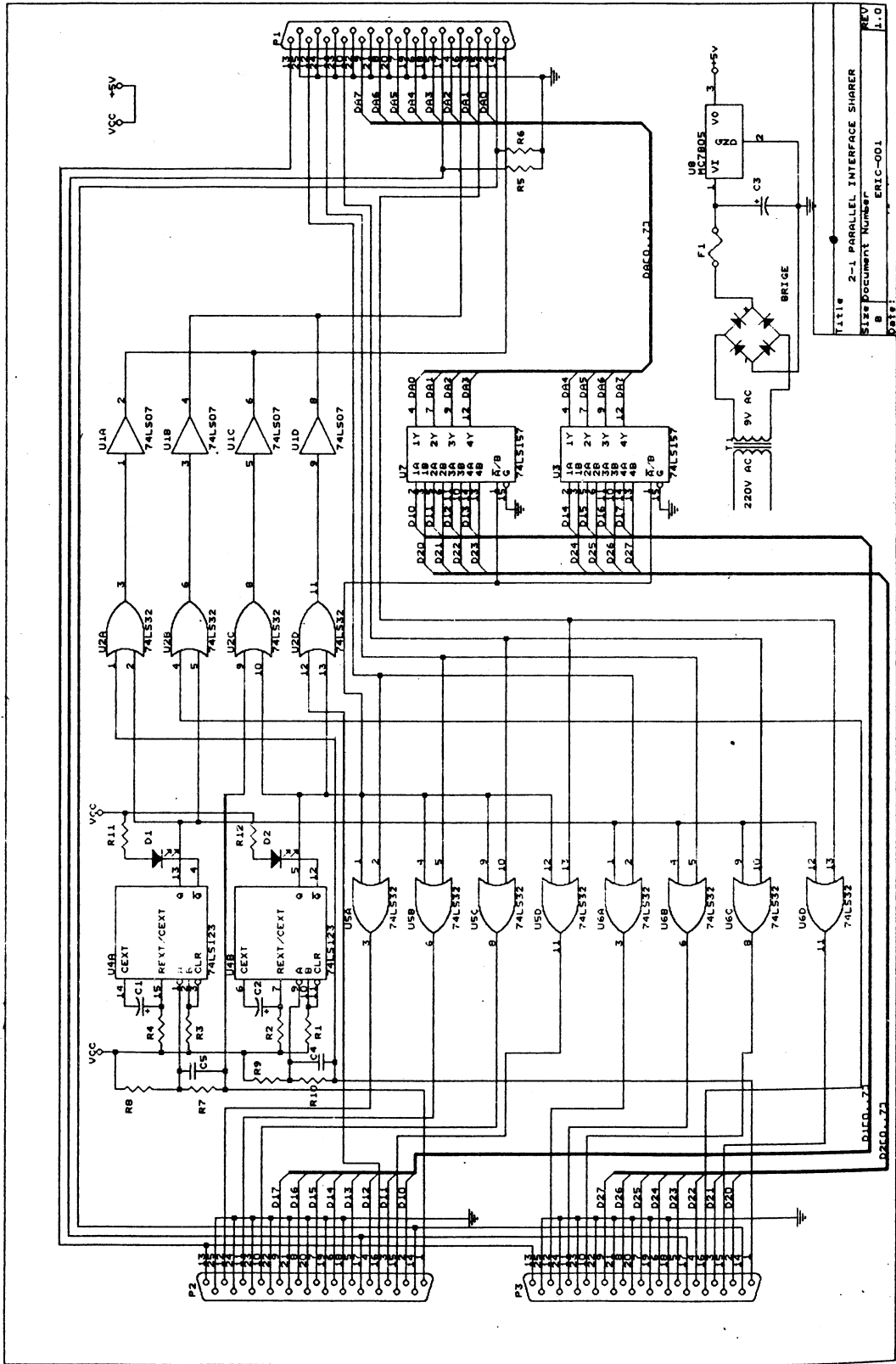


图 1 打印机通信接口电路框图

打印机并行通信接口电路主要由接口插座,复位电路,信号联络,状态输入输出电路及数据输入电路组成。电路采用标准的 D 型接口,所有通信信号用 25 针脚引出,由电缆与主机并行接口连接。复位电路与控制电路中各芯片的复位端相连,打印机通过复位电路进行初始化,使缓存清除,各触发器复位,为打印输出工作做好准备。信号联络电路在数据传输过程中起到主机与打印机之间联络作用;其由选通输入,忙输出及认可输出信号组成。STROBE 选通脉冲信号;由主机发出,通过联络电路向打印机的主 CPU 的 INT 端提出输入中断请求信号。该信号用于通知打印机读取主机送来的数据(DATA0~DATA7),当信号为低电平时有效,高电平时无效。ACK, 认可信号;由打印机主 CPU 的 PB1 端发出,通过信号联络电路通知主机,打印机已接收主机传输来的数据,并做好接受下一个数据的准备。BUSY, 忙信号。由打印机的主 CPU 的 PB0 端发出,通过信号联络电路通知主机,打印机的工作状态。BUSY 为高电平,打印机忙,不能接收数据。反之,表明打印机工作结束,可以接收主机传来的数据。状态输入输出是由联机,纸尽,故障等信号组成。SLCT, 联机信号;用于指明打印机处于联机还是脱机状态。高电平时、联机状态。反之亦反。PE, 纸尽输出信号;由打印机主 CPU 发出,向主机表明打印机是否装有纸。PE 为高电平时,打印机自动脱机停止打印,且蜂鸣器鸣叫报警。ERROR, 故障输出信号,向主机传递打印机故障信息。低电平时,打印机脱机,停止打印并报警。AUTO FEEDXT, 自动换行信号;由主机接口提供一个输入信号,用来设置是否由 CR 代码(回车)自动产生换行(LF)。当信号为低电平时,打印机收到 CR 代码的自动连行换行,不必向打印机送 LF 代码。反之,打印机收到 CR 代码时,打印一



TITLE: 2-1 PARALLEL INTERFACE SHARER
 SIZE: DOCUMENT NUMBER ERIC-001
 REV: 1.0
 DATE:

图 2 DAKE 电脑打印机共享器电路图

行后不产生换行行为。SLETIN, 为打印机选择输入信号, 由主机接口提供的数据输入类型选择信号。数据输入电路的作用是将主机传送的数据信息或控制码(控制指令), 通过数据总线馈入打印机 CPU。在通常情况下, 主机与打印机通信时, 先由 CPU 查询打印机的 ACK 或 BUSY; 若打印机“忙”, 则主机等待, 若“空”, 则把管理权交给优先级高的请求主机。打印机的服务时间等于连续信息通信所需的最长时间, 以确保当前管理主机的信息传输连续。

实际上, 主次要控制打印机, 其总是先在数据线上送出一个字节并行数据, 然后主机选通线 STROBE 输出一个脉冲, 在该脉冲的上升沿, 主机送出的数据才能真正被打印机接收。当打印机接收了一个字节的数据后, 送回一个应答信号 ACK; 如果主机要传送下一个字节, 则只有主机接收到一个 ACK 信号后才能进行, 否则只能处于等待状态。DAKE 微机打印机共享器就是采用这样的特点, 主设计而成。

二、微机打印机共享器电路设计

基于选通脉冲 STROBE 作主控信号, 配之 BUSY ACK、PE、ERROR、INIT 等状态信号, 来控制打印机自动地为不同主机服务的机理, DAKE 微机打印机共享器电路设计如图(2)所示。该电路共采用 7 块 74LS 系列的集成电路构成, 结构简单, 性能稳定可靠, 可以全自动地使两台电脑共享 1 台打印机。

在实际使用中, 两台电脑主机 A 和 B 的打印并行口分别和打印机共享器 P2 和 P3 接口通过带有标准 D25 脚插座的电缆连通, 且打印机电缆的一端接于打印机共享器的 P1 接口。见图(3)所示。

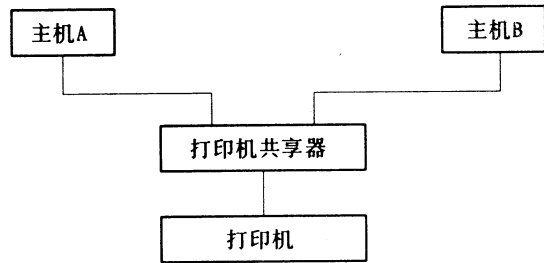


图3 使用连接框图

分别打开电脑及打印机电源, 使电脑处于正常工作状态, 打印机处于联机状态。这样, 当主机的 CPU 检测到打印机处于空闲状态时, 发出选通脉冲信号 STROBEA 或 STROBEB, 双延迟门 U4 使该脉冲信号变为电平信号, 令 U4 的 13 和 5 脚输出状态分别为“1, 0”。U4 的 5 脚打开 U5 各或门, 连通 BUSY, ACK, PE, ERROR 信号到主机 A 的通路, 同时 U4 的 13 脚使 U6 的各或门输出 1, 返回忙, 缺纸, 不确认, 设备不能操作等信号于主机 B, 从而连通主机 A 和打印机通路, 阻断主机 B 和打印机通路。事实上, 由于主机的 CPU 申请打印时, 先查询打印机状态寄存器, 如果得到打印机忙(或 I/O 出错)的信号, 就不会发 STROBE 选通脉冲。所以, 当主机 A 输出打印期间, 主机 B 不可能发出 STROBE 选通打断主机 A 正在进行的打印机工作。当主机 A 优先选通打印通路后, U4 的 5 脚的低电平控制 U3, U7 选择主机 A 的数据馈入打印机, 并阻断主机 B 的数据传递。同理, 主机 B 的打印过程亦一样。DAKE 微机打印机共享器(2×1)推出后, 受到电脑应用领域广大用户的欢迎。我们拟继续推出(3×1), (4×1)等系列打印机共享器, 欢迎交流。

电子工业出版社广州科技公司邮购部:
 通信地址: 广州市石牌华南师大北区一号 203
 邮 编: 510630
 联系人: 王丽端



· MS-DOS 操作系统结构分析系列教材之一

BIO 结构分析教程 郭嵩山编著

本书从操作系统原理和结构的角动，采用以模块主程序为主线，以数据结构为中心的系统软件分析方法，彻底剖析了 MS-DOS (PC-DOS) 3.3 版操作系统的三大模块之一——基本输入输出模块。全书共分六章：IBMBIO 模块总体概述；IBMBIO 引导；标准设备驱动程序；BIO 初始化实现原理；系统初始化实现原理；与系统配置文件的处理。

本书是由中山大学自然科学基金资助的有关操作系统研究科研课题成果之一。

本书力求从有利于教与学的角动对一般人感到难度很大的操作系统内部结构和实现原理通过深入浅出的系统论述，让读者既能建立整体的概念，又能逐步深入，一层一层地剖析，为了帮助读者能结合原理读懂程序清单，对于程序清单的注释尽量详细，力求深入到每一条指令。每章后均附有习题和思考题，以帮助读者更好地理解和掌握。

本书可作为大专院校计算机有关专业的教材和教学参考书，也是从事微型计算机系统和应用开发的工程技术人员常备的技术参考书。

全书 630 千字，408 页，16 开，定价：21.50 元
邮挂费 3.20 元

该系列教材之二三为：

COMMAND 结构分析教程

DOS 内核结构分析教程

均将在今年三、四季度出版发行

(电子工业出版社出版)

· 高频开关稳压电源 张占松编著

本书系统介绍高频开关电源的工作原理，电路构成、设计计算方法、实际使用的器件及调整方法。全书分 12 章，从降压、升压变换器讲到它们的派生、组合的变换器；从一般脉调制控制讲到准谐振、多谐振开关、零电流、零电压开关的控制；从单端变换器讲到半桥、全桥变换器；对开关器件 GTR、VMOSFET 等有专章分析、典型应用、调整方法有深入浅出的介绍。对兆赫级工作的谐振式零电压变换器，多谐振变换器均有论述。书中有例题、习题。

全书 400 千字，288 页，16 开，定价：9.80 元，
邮挂费：1.50 元 (广东科技出版社出版)

· 开关电源集成电路手册 (POWER INTEGRATED CIRCUIT DATA BOOK)

该书系美国 POWER 公司有关开关电源 IC 芯片的英文选编本，介绍的芯片有：PWR-SMP210、211、212、220、240、260 等，以及以上芯片的应用资料，包括：利用开关 IC 的电路图，印刷板图及有关高频变压器的资料。(注：请读者注意，该书为英文本)

全书 250 千字 202 页，16 开，定价：18.00 元，
邮挂费：2.70 元 (中国器材深圳公司资料)

· MICROSOFT C6.0 大全

徐为民 刘益敏等译

全书共分三篇：分别为：“安装和使用专用开发系统，MICROSOFT C6.0 使用指南和 MICROSOFT C6.0 高级编程技巧。本书是学习和掌握 MICROSOFT C6.0 的实用参考书，适合广大从事计算机教学、研究和应用开发的科技人员使用。

全书 1100 千字，723 页，定价：34.00 元，邮挂费：5.10 元 (中国科技大学出版社出版)

· NOVELL 网络及其互联技术

汤岳清编著 李智渊主审

目前，国内计算机网络大都基于微机局域网 (PCLAN)。随着 NOVELL 网络的普及和应用的深入，人们不再满足于单一的 NOVELL 网络在部门级充当文件和打印服务器的角色。特别是企业集团化使企业间通信要求大大增加，这样，把多个 NOVELL 网络和企业内的大型机、小型机或 UNIX 工作站以及与企业外的多种资源 (如 X.25) 连成一体，达到共享资源和通信的目的。这是国内网络应用的现实课题和未来网络发展方向。

本书作者有承接多项网络工程的实践经验，书中介绍了许多网络互联的技术问题，对 NOVELL 网络的工程开发具有重要的指导意义。

全书 250 千字，16 开，208 页，定价：13.00 元，
邮挂费 2.00 元 (电子工业出版社出版)

下期新书预告

- NOVELL 网络的安装、使用指南
- NOVELL 网络的故障维护技术
- 微计算机的下一场革命——多媒体技术

汇款请写明详细地址及邮编以便能准确及时的将您的邮购品寄出，谢谢合作。

电子工业出版社广州科技公司

邮购产品介绍

软件狗 (WATCH DOG) —

——软件的忠实保护神

WATCH DOG 是一种新颖的软件加密工具。它体积小 (60×50×15mm)，只需将其插入主机箱外的并行输出插口上 (即串接在主机并行口和打印机电缆之间，不占用机器扩展槽)。

当主机安装了某个 WATCH DOG (设其为 1 号狗)，即可通过 WATCH DOG 对您使用的软件进行加密，被加密的软件可以被任意拷贝但不能在没有加装 1 号狗的机器上运行。或者从另一个角度来看，当某软件用 2 号狗加密后，该软件只能在装上 2 号狗之后才能运行。也就是说，如果您的软件要加密保护，那就每一套软件必须有一个相应的 WATCH DOG。

适用范围：可对 C (BORLAND C⁺⁺、TURBO C、MSC C)、编译 BASIC、FORTRAN、PASCAL、汇编语言、dBASE (Clipper、dBASE III、Fox base、FOXRPO) 等写成的软件加密。也可对 EXE 可执行文件加密。

加密方法：只要在源程序中加入调用 WATCH DOG 函数的语句，编译时将随 WATCH DOG 提供的 OBJ 文件链接起来，形成加密的 EXE 文件，或者调用随 WATCH DOG 提供的 LINKEXE.EXE 对可执行的 EXE 文件加密。

- 特点：
1. 采用多种反跟踪，反破译，反仿制技术。完整的 WATCH DOG 代码大于 9K。WATCH DOG 的核心程序几乎不可能被破译。
 2. 能直接与 EXE 可执行文件链接，产生一个新的 EXE 可执行文件。完成对 EXE 文件的加密。
 3. 每一 WATCH DOG 及随 WATCH DOG 提供的软件都有一唯一的 ID 号，无任何信息、资料可查询到这些 ID 号。WATCH DOG 及随 WATCH DOG 提供的软件是一一对应的，其间不能互用。
 4. 对各类打印机正常工作无任何影响。
 5. 对于网络用户，可以订购一批同 ID 号的 WATCH DOG。

WATCH DOG 可能是您最好的软件加密工具。

SUNSHINE WY09-3B 开关

电源

输入电压：AC220V ± 15% 0.7A 50Hz
AC110V ± 15% 1.4A 60Hz 开关切换

输出电压：
+5V 端：恒压输出，其可调范围 4.5V ~ 6.5V，

主负载输出端

工作电流 最大值 15A
纹波电压 小于 100mV
瞬变特性 100% 负载变化时优于 ±

0.5%

+12V 端：恒压输出电压误差范围 ± 2%
工作电流，最大值 1.5A
纹波电压 小于 50mV
瞬变特性 100% 负载变化时优于 ± 2%

-12V 端：恒压输出 电压误差范围 ± 2%
工作电流 0 ~ 0.5A
纹波电压 小于 50mV
瞬变特性 100% 负载变化时优于 ± 2%

输出功率：75W 最大值 90W (需加 2mm 水柱
高压散热)

空载功率：小于 6W

工作效率：大于 82%

开机涌浪：小于 20A / 10mS

隔离电压：AC、DC、FG 间大于 1000VAC 1Min

TEM = 27° HUM = 60%

绝缘电阻：AC、DC、FG 间大于 100MΩ

TEM = 27° HUM = 60%

工作环境：TEM = -10℃ ~ +35℃ HUM = 10% ~ 90% REL

工作温升：在 +27℃ 自然无风冷散热情况下，机身温度低于 +70℃

外形尺寸：L158 × W98 × H48mm

机身重量：小于 0.5kg

本机还具有交流滤波、开机缓冲冲击、过载保护、输出输入短路保护、LED 工作指示、外壳与地隔离等功能。

主要应用于微外理机，可编程控制器、高精程控制器、精密电子仪器、开发机，显示屏供电等。

DP—851 监控程序简介

北京宣武区科技青少年馆(100053) 罗明宽 车金相

上期我们对 DP—851 单片机普及板的电路图和电路原理,已作了详细的说明。本期继续向读者介绍本系统的主要性能,并对监控程序中的主要模块,逐条向读者介绍。目的是使读者尽快掌握编程思路和技巧,或归己所用,以更充足的时间和精力,开发单片机。

DP—851 普及板性能简介。

DP—851 单片机普及板是为单片机的普及、教学、开发服务的一种专用 51 系列单片机系统,可为两种用户服务,无 PC 机用户可利用本普及板构成轻便型开发板,再加上其它接口板,可进行各种基础实验和开发。PC 机用户,可与 PC 机相连,组成联机型开发板,可以实现汇编语言的程序设计及调试。

DP—851 具有很强的调试功能,对读者的学习和使用提供了极大的方便,它包括:

1. 写入操作

可方便对片外 RAM 空间写入数据,完成程序的输入操作,同时还能对片内 RAM,工作寄存器及特殊功能寄存器内容的写入。

2. 读操作

可将读者向 RAM 空间键入的数据读出,可方便对键入的程序进行检测。

3. 修改操作

利用读写插入、填充基本操作对键入的程序进行修改。

4. 数据块的移动和比较

利用数据块移动操作,可将程序块移置到读者所需要的起始地址单元,并能对其检测显示错误信息。

5. 运行控制功能

能控制单片机单步、断点运行和连续全速运行,除连续全速运行外,其它各种运行状态,在运行过程中或运行结束后,向用户提供单片机系统的状态信息,利用此功能可向用户提供程序调试手段。

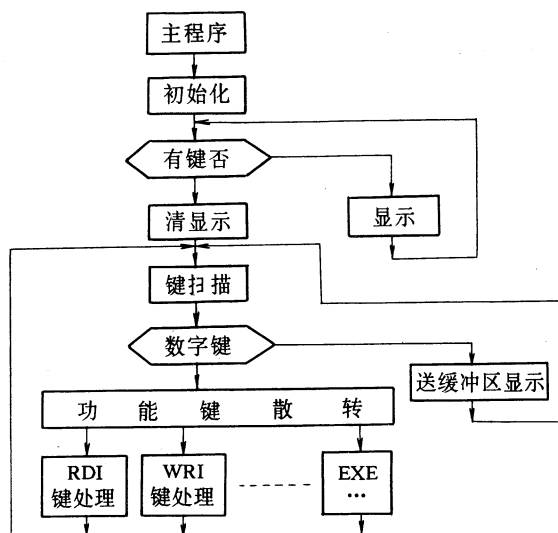
6. EPROM 编程功能

利用编程板可对 2716、2732、2764、27128 芯片进行固化、检测、读入等操作,为用户提供开发手段。

7. 本系统配有简易的 RS232 标准通信接口,可方便地与微机相连,在微机上对单片机进行调试和开发。

DP—851 监控程序简介。

监控程序是由主程序、键功能处理程序,I/O 驱动程序及通用程序几部份组成。主程序的任务是:为监控程序的运行作好准备(初始化)并监督分析处理键的状态,在遇到有键按下时,转入相应的键功能处理程序,在完成键处理程序后,返回主程序,并等待下一次键输入。监控程序框图如下:



键功能处理程序是为完成每一个键特定功能而设计的程序,共有 16 个功能键处理程序,分别与各功能键相对应,它们的入口地址、键名、键值及功能见下表:

功能键处理程序入口地址

入口地址	键名	键值	功 能
09E9H	WRI	10H	存储器写入并地址增 1
0D38H	ME	11H	将 RAM 中内容送入 EPROM 中去
105DH	FI	12H	填充操作(用于将某一机器码填充指定的地址空间)
163BH	FU	13H	功能扩展键
0AE2H	F1	14H	首地址置入
0C42H	MOV	15H	数据块传递
0C67H	INS	16H	数据插入(在连续两地址间插入数据)
0EAFH	C—C	17H	数据块比较
0B03H	F2	18H	末地址装入
0DB3H	EPR	19H	EPROM 编程
10F3H	BP	1AH	全速断点运行控制
0606H	EXE	1BH	连续运行控制
0957H	RDI	1CH	读存储器内容地址增 1
0AADH	RDS	1DH	读存储器内容地址减 1
02A0H	STEP	1EH	单步运行控制
0B35H	MON	1FH	返回监控

主程序占用 0535H~0605H,包括:

1. 初始化程序(0535H~05A8H)完成下列几项工作:

A: 对显示缓冲区(79H~7EH)初始化(055CH~056EH)

B: 数据缓冲区及系统工作区初始化(0535~055B,0572H~0596H)。用户 PC 值初始化为 8000H,系统初始化为仿真状态。

C: 系统堆栈指针设置为 60H

D: I/O 芯片 8155,8255 初始化(0597H~05A8H) 8255P_A□、P_B□、P_C□均为输出口。

2. 键判断程序从 05A9H 开始,子程序 0810H 用于判断是否有键按下,A≠0 为有键按下,否则无键按下。子程序 0772H 为显示子程序。子程序 07BAH 用于键扫描和译码,将最后结果(键值)存放于累加器 A 中。本系统编码规则为 00H~0FH 定义为数字键的键值,大于 0FH 的为功能键的键值。可用 A 累加器中 D4 位进行判断是功能键还是数字键,05C0H~05CCH 进行数字键处理。

3. 功能模块程序

功能模块程序的入口地址,由功能键处理程序从 05CDH 开始,先对键值进行处理,通过查表找到入口地址,转移到各功能模块程序进行处理,它们的入口地址见前表。

DP-851 监控程序中主要模块介绍。

在单片机开发和应用过程中,经常会遇到键盘扫描和显示软件设计问题,如果键盘和显示软件的设计和编写,等于完成了 50% 的监控程序,因此这两个程序读者是必须掌握的,不仅会调用,而且深刻理解其工作原理,下面将本系统的键盘程序和显示程序结合硬件向读者逐条介绍,有关键盘接口需要解决的问题,请读者参考有关书籍,键盘扫描子程序入口地址为 07BAH,显示子程序入口地址为 0772H。

1 键盘判断子程序清单

```
07BA: 12 08 LCALL 0810H    键盘扫描
10
07BD: 70 04 JNZ 07C3H      有键按下转 07C3H
07BF: F1 72 ACALL 0772H    无键按下显示和延时
07C1: E1 BA AJMP 07BAH     继续键盘扫描
07C3: F1 72 ACALL 0772H    延时显示(消除抖动)
07C5: F1 72 ACALL 0772H
07C7: 12 08 LCALL 0810H    有键按下再扫描
10
07CA: 70 04 JNZ 07D0H     有键按下转 07D0H
07CC: F1 72 ACALL 0772H    无键按下显示延时
07CE: E1 BA AJMP 07BAH     返回继续扫描键盘
07D0: 7A FE MOV R2, #FE    R2 装发送状态
07D2: 7C 00 MOV R4, #00    R4 为列值
07D4: 90 60 MOV DPTR, #    6001 8155 的 A 口
01
07D7: EA    MOV A, R2
```

```
07D8: F0    MOVX @DPTR, A    对列扫描
07D9: A3    INC DPTR
07DA: A3    INC DPTR        指向 8155C 口
07DB: E0    MOVX A,        读 C 口状态(读键盘行
@DPTR        状态)
07DC: 20 E0 JB E0,07E3H     在当前列状态下,第一
04          行是否有键按下?
07DF: 74 00 MOV A, #00      第一行有键按下,第一
07E1: E1 F6 AJMP 07F6H     行行值取 00 转 07F6H
07E3: 20 E1 JB E1,07EAH     第一行无键按下,第二
04          行是否有键按下?
076: 74 08 MOV A, #08      第二行有键按下行值取
07E8: E1 F6 AJMP 07F6H     08 转 07F6
07EA: 20 E2 JB E2,07F1H     第二行无键按下,第三
04          行是否有键按下?
07ED: 74 10 MOV A, #10      第三行有键按下,行值
07EF: E1 F6 AJMP 07FF6     取 10 转 07F6
07F1: 20 E3 JB E3,0803H     第四行无键按下,转
0F          0803
07F4: 74 18 MOV A, #18      第四行有键按下,行值
          取 18
07F6: 2C    ADD A, R4      键值 = 行值 + 列值(A
          为行值)
          键值进栈
07F7: C0 E0 PUSH ACC      键值进栈
07F9: F1 72 ACALL 0772H    显示延时
07FB: 12 80 LCALL 0810H    键扫描
10
07FE: 70 F9 JNZ 07F9H     等待键释放
0800: D0 E0 POP ACC      键值出栈
0802: 22    RET          返回
0803: 0C    INC R4        在当前列状态下(0 状
          态)各行无键按下,对下
          一列扫描列值增 1
0804: EA    MOV A, R2      取发送状态
0805: 30 E7 JNB E7,080DH     扫描到第七列还无键按
05          下,转 080DH
0808: 23    RL A          取下列发送状态
0809: FA    MOV R2, A      下列发送状态送往 R2
080A: 02 07 LJMP 07D4H     转 07D4H 进行下一列
D4          扫描
080D: 02 07 LJMP 07BA      无键按下重新键扫描
BA
0810: 90 60 MOV DPTR, #    6001 键扫子程序指向
01          8155A 口
0813: 74 00 MOV A, #00      列线(8155PA0~PA7 输
0815: F0    MOVX @DPTR, A    出)0 电平
0816: A3    INC DPTR      指向 8155C 口
0817: A3    INC DPTR
0818: E0    MOVX A, @      DPTR 读 C 口状态送
          A
0819: F4    CPL A          取 C 口 @ 状态
081A: 54 0F ANL A, #0F      保留 C 口低四位状态
081C: 22    RET          返回
```

几点说明:

①键判断程序中调用了 0810H 子程序,0810 子程序,只用于判断是否有键按下,当 A≠0 时,为有键按

下,否则无键按下。

②07D0H~080CH 完成哪一行哪一列的键被按下同时取得按下的键值,送 A 累加器。

③07F9H~07FFH 处理按下键后不释放的处理,保证只取一个键值。

④0772H 子程序是显示程序,在本程序块中,其作用是用于消除抖动和显示。

2. 显示子程序模块清单

0772:78 79	MOV R ₀ , #79	显示子程序 79~7E 显示缓冲区
0774:7B 01	MOV R ₃ , #01	R3 存放位控码
0776:EB	MOV A, R ₃	
0777:90 60 01	MOV DPTR, #	6001H 指向 8155A 口
077A:FO	MOVX @DP	TR, A 位控码送 A 口
077B:A3	INC DPTR	指向 8155B 口
077C:E6	MOV A, @R ₀	显示缓冲区内内容送 A
077D:24 12	ADD A, #12	
077F:83	MOVC A, @A	+PC 取显示代码 送 A
0780:30 D5 02	JNB D5, 0785H	若 D5=1 点亮小数 点, D5=0 转送显示
0783:D2 E7	SETB E7	A 累加器最高位置 1
0785:FO	MOVX @DP	TR, A 送显示
0780:F1 B1	ACALL 07B1H	延时
0788:08	INC R ₀	指向下一个显示缓冲 区
0789:EB	MOV A, R ₃	位控码送 A
078A:20 E5 04	JB E5, 0791H	若显示到最高位返回
078D:23	RL A	取下一个位控码
078E:FB	MOV R ₃ , A	位控码送 R ₃
078F:E1 77	AJMP 0777H	转 0777H 继续显示
0791:22	RET	返回
0792:3F 06 5B 4F 66 6D 7D 07 7F 6F 77 7C 39 5E 79 71 73 3E 31 6E 1C 23 40 03 18 00 00 1E 79 38 5E		
07B1:7F 02	MOV R ₃ , #02	延时子程序
07B3:7E FF	MOV R ₆ , #FF	
07B5:DE FE	DJNZ R ₆ , 07B5H	
07B7:DF FA	DJNZ R ₃ , 07B3H	
07B9:22	RET	

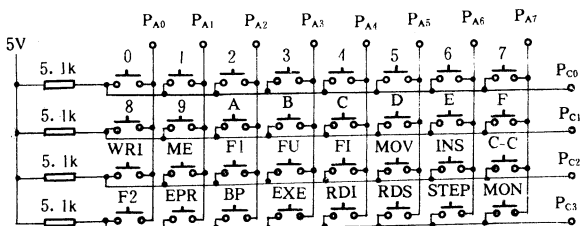
几点说明:

①79~7E 为显示缓冲区,79~7A 为显示数据,7B~7E 显示地址。

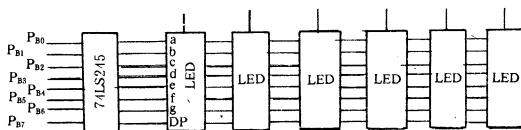
②本系统所用的是动态显示方式,哪一个数码管被点亮,其控制是由 8155 的 A 口通过 74LS06 来实现的,位控码 01,02,04,08,10,20 分别对应显示缓冲区 79,7A,7B,7C,7D,7E 各单元。

③0792H~07B0H 放的是显示代码。

④为了方便读者读键处理程序和显示子程序,将其硬件电路图附录如下:



键盘图



显示电路图

3. 调用方法

①显示子程序的调用,只要将显示的内容送入显示缓冲区,调用显示子程序即可。

②键盘处理子程序的调用读者只需明确两点:a. 键盘处理子程序中已调用了显示子程序,b. 有键按下键值存入 A 累加器中。

4. 调用范例

题目:对 RAM 空间写入数据程序的编写

8000:78 79	MOV R ₀ , #79	对显示缓冲区清零 (请显示)
8002:74 19	MOV A, #19	
8004:F6	MOV @R ₀ , A	
8005:08	INC R ₀	
8006: B8 7F FB	CJNE R ₀ , #7F, 8004H	R ₀ 装入显示缓冲区 最高地址
8009:78 7E	MOV R ₀ , #7E	
800B:C0 00	PUSH R ₀	R ₀ 值进栈
800D: 12 07 BA	LCALL 07BA	调用键处理子程序
8010:D0 00	POP R ₀	R ₀ 出栈
8012:F6	MOV @20, A	键入的数字写入显示 缓冲区
8013:E5 7A	MOV A, 7A	
8015: B4 10 03	CJNE A, #10, 801B	7A 单元中是否是 WRI 键?
8018: 02 80 22	LJMP 8022H	是 WRI 键转 8022H
801B:18	DEC R ₀	7A 单元中不是 WRI 键,取下一个显示缓 冲区地址
801C: B8 79 EC	CJNE R ₀ , #79, 800BH	R ₀ 不等于 79 转 800B 搜索键盘
801F: 02 0B	LJMP 0B35H	R ₀ 是 79 返回监控

8022:74 19	MOV A, #19	} 请显示	
8024:F5 7A	MOV 7A, A		
8026: 12 07	LCALL 0772H		
72		} 调用显示器程序	
8029:E5 7C	MOV A, 7C		
802B:C4	SWAP A		
802C:25 7B	ADD A, 7B	} 将键入的四位十六进制数合并成地址低八位送 20 号单元, 高八位送 21 号单元。	
802E:F5 20	MOV 20, A		
8030:E5 7E	MOV A, 7E		
8032:C4	SWAP A		
8033:25 7D	ADD A, 7D		
8035:F5 21	MOV 21, A		
8037:78 7A	MOV R ₀ , #7A		} R ₀ 装显示缓冲区
8039:C0 00	PUSH R ₀		
803B: 12 07	LCALL 07BAH		} 调用键处理程序
BA			
803E: B4 1F	CJNE A, #1F,	} 8044H 是否按下“MON”键? 不是, 转 8044	
03			
8041: 02 0B	LJMP 0B35H	} 是, 返回监控	
35			
8044:D0 00	POP R ₀	} R ₀ 出栈	
8046:F6	MOV @R ₀ , A		
8047:18	DEC R ₀	} 将按下的数字写入显示缓冲区	
8048: B8 77	CJNE R ₀ , #77,	} 取一下显示缓冲区地址	
EE			
804B:E5 78	MOV A, 78	} 8039H R ₀ 是否等 77, 不是转 8039 继续键搜索	
804D: B4 10	CJNE A, # 10,		
03	8053H	} 是否按下 WRI 键否 转 8053H	
8050: 02 80	LJMP 8056H		
56		} 是按下 WRI 键, 转 8056H	
8053: 02 80	LJMP 8039H		
39		} 将键入的两位十六进制数合并成数据送 22 号单元	
8056:E5 7A	MOV A, 7A		
8058:C4	SWAP A		
8059:25 79	ADD A, 79		
805B:F5 22	MOV 22, A		
805D:E5 20	MOV A, 20		} 将 20 号单元 21 号单元中内容取出送 DPTR
805D:E5 20	MOV DPL, A		
805F:F5 82	MOV A, 21		
8061:E5 21	MOV DPH, A		
8063:F5 83	MOV A, 22		
8065:E5 22		} 将键入的数据送入 DPTR	
8067:FO	MOVX @DPTR, A		
8068:C0 82	PUSH DPL	} DPTR 进栈	
806A:C0 83	PUSH DPH		
806C:74 19	MOV A, #19	} 请显示	
806E:F5 7A	MOV 7A, A		
8070:F5 79	MOV 79, A		
8072: 12 07	LCALL 0772	} 调显示器程序	
72			

8075:D0 83	POP DPH	} DPTR 出栈
8077:D0 82	POP DPL	
8079:A3	INC DPTR	} DPTR 加形成下一个地址
807A:E5 82	MOV A, DPL	
807C:F5 20	MOV 20, A	} 将新的 DPTR 值存入相应的 20、21 号两单元中
807E:E5 83	MOV 20, A	
8080:F5 21	MOV A, DPH	
	MOV 21, A	
8082:E5 20	MOV A, 20	
8084:54 0F	ANL A, #0F	} 将新的 DPTR 值送入显示缓冲区显示地址
8086:F5 7B	MOV 7B, A	
8088:E5 20	MOV A, 20	} 转 8037H 准备键入新的数据
808A:54 F0	ANL A, #F0	
808C:C4	SWAP A	
808D:F5 7C	MOV 7C, A	
808F:E5 21	MOV A, 21	
8091:54 F0	ANL A, #F0	
8093:C4	SWAP A	
8094:F5 7E	MOV 7E, A	
8096: 02 80	LJMP 8037H	
37		

示范程序使用说明

读者按上述地址键入程序后, 返回监控, 按下下述步骤进行:

1. 键入 8000, 按一下全速运行键(EXE)
2. 键入 9000, 按一下 WRI 键(写入键)
3. 键入数据例如 74, 按一下写入键, 74 便被写入 9000 号单元中, 同时地址自动增 1 并显示
4. 再键入数据, 再按一下写入键, 新的数据写入下一个地址单元中, 依次类推
5. 数据(程序)键入完毕可按一下 MON 键, 返回监控
6. 可用 RDI 键读 9000 起始地址单元, 查看刚键入的数据(程序)是否写入到相应的地址单元中去。

如果读者认真读一下上述程序, 将受益非浅。其实此程序已是监控程序中的写入操作, 稍加修改, 便可完善。

(接 54 页)

地址范围	用途
3F00~3F0F	背景页配色代码定义区, 其中 \$3F00 为底背景配色单元
3F10~3F1F	卡通页配色代码定义区。其中 \$3F10 为底背景配色单元。
3F20~3FFF	空区。映射于 \$3F00~\$3F1F。

(待续)

实用汉字库芯片的制作

陈超波

近年来,计算机领域的汉字技术得到了广泛的应用,许多类型的计算机系统及应用设备都配备了汉字显示。而更多的场合,如电脑控制的广告牌、带中文显示的专用设备,它们仅需几十至几百个汉字,如果为这样的系统配备具有一级或二级字库的汉字库芯片,显然是不经济的,也是不太现实的。解决的办法是挑选出一部分需要的汉字,制作成实用的汉字库芯片。

实用汉字库芯片的制作工作可分为字模文件的形成和 EPROM 芯片的写入两部分,其中芯片的写入可以使用 EPROM 写入卡或用 DVCC-51 单片机开发机来完成。不同的写入方法对字模文件有不同的要求。如果使用 EPROM 写入卡,则把汉字字模直接放在字模文件中,而使用开发机完成写入过程,则在字模文件的前八个字节安排 EPROM 的首末地址。一般首地址是 0000H,末地址视不同型号的 EPROM 芯片而定。如 Intel2764 芯片的末地址是 1FFFH。下面我们着重说明字模文件的形成过程。

首先,让我们来分析一下汉字库的结构。大家知道,在国家 GB2312-80 标准中有 6763 个汉字,共分为 94 个区,每区 94 个位,每个汉字对应一个区位号。其中 1-9 区为符号,字母和制表符,16-87 区为一、二级汉字,其他为空区。CCDOS 软字库结构跟 GB2312-80 稍微有所不同,它把 16-87 区的汉字往前挪了 6 个区,省略了原来 10-15 及 88-94 区。这样的字库结构可以节省一部分存储空间。

知道了字库结构之后,我们就可以来读取某个汉字的字模。只要知道一个字的区位号,就不难算出它在字库中的位置。具体关系如下:

$$\text{区号} = \begin{cases} \text{区号}, & \text{区号} \leq 15 \\ \text{区号} - 6, & \text{区号} > 15 \end{cases}$$

$$\text{位置码} = (\text{区号} - 1) \times 94 + \text{位号}$$

但是,汉字在输入计算机时,在机器内是用统一的机内码表示的。机内码是系统内部处理和存储汉字而使用的代码,CCDOS 采用变形国标码作为汉字机内码。由两个字节组成,它与区位码相差 160,即机内码的高字节和低字节分别减去 160,就是相应的区号和位号。根据这些转换关系,即可从字库中读取字模存入字模文件中。

程序开始到 370 句实现从字库中读取字模,其中 110-140 句保证输入的汉字个数是 4 的倍数,以确保字模文件的完整性;170-220 句用于查找汉字,其中 M 是区号,N 是位号,输入的汉字都能在此范围内找

到;290-330 句用于显示字模,显示 4 个字节后自动换行。

另外,在 CCDOS 中用于显示的 16 点阵软字库里字模的排列为横向点阵形式,如图 1 所示。

第 0 字节	第 1 字节
第 2 字节	第 3 字节
⋮	
⋮	
第 28 字节	第 29 字节
第 30 字节	第 31 字节

图 1

第 0 字节	第 16 字节
第 1 字节	第 17 字节
⋮	
⋮	
第 14 字节	第 30 字节
第 15 字节	第 31 字节

图 2

在实际的应用中,这种横向点阵显示常使硬件与软件设计不够合理。这时可将字模转换成如图 2 所示的纵向点阵形式。程序 500-700 句可以由用户选择是否完成这一转换。

实用汉字字模文件形成的最后一步是十六进制数到 ASCII 码的转换过程。程序 740-960 句实现这一功能。

由此可见,整个实用汉字字模文件形成是由以下几步来完成的:

1. 根据输入汉字计算汉字机内码并形成字库中的区位码。
2. 根据区位码找到该字字模的首地址,并读出字模存放到位字模文件中。
3. 由用户选择是否完成纵向点阵形式的转换。
4. 由写入形式确定是否将芯片首末地址写入字模文件中。
5. 完成十六进制数到 ASCII 码的转换。

最后,需要注意的一点是,用户应根据自己的需要选用相应的 EPROM 芯片。在完成汉字输入时,应记住存入文件时的顺序,计算其地址,以便以后使用。

本程序使用字库为 CCDOS4.0 版本,在 IBM PC XT/AT 机上运行通过。

```

10 CLS:ON ERROR GOTO 420
20 CLOSE
30 OPEN "A:CCLIB" AS #1 LEN=32
40 IF LOF(1)<>0 THEN 80
    
```

```

50 PRINT:PRING“请在 A:驱准备好 CCLIB 汉字库,按任
   意键继续……”
60 SS$=INKEY$:IF SS$="" THEN 60
70 GOTO 20
80 FIELD #1,32 AS A$
90 OPEN "CCM.DOT" AS #2 LEN=32
100 H=LOF(2);P=INT(H/32)
110 PRINT:INPUT“请输入 4 的倍数个汉字:”,B$
120 L=LEN(B$);IF L-INT(L/8)*8=0 THEN 150
130 PRINT:LL=4-(L-INT(L/8)*8)/2:PRINT“请再
   输入”;LL;“个字:”;:INPUT BB$
140 B$=B$+BB$:GOTO 120
150 FOR I=1 TO L STEP 2
160 C$=MID$(B$,I,2)
170 FOR M=1 TO 87
180 FOR N=1 TO 94
190 D$=CHR$(160+M)+CHR$(160+N)
200 IF D$=C$ THEN 230
210 NEXT N
220 NEXT M
230 PRINT:PRINT C$;“:”;:PRINT“区位码:”,M*100
   +N
240 X$=HEX$(160+M).Y$=HEX$(160+N)
250 PRINT“      机内码:”,X$;Y$
260 IF M>15 THEN M=M-6
270 R=(M-1)*94+N:GET #1,R
280 PRINT:PRINT“      字模:(16 进制)”
290 FOR J=1 TO 32
300 BB$=MID$(A$,J,1):DD=ASC(BB$):FF$=
   HEX$(DD)
310 IF J-INT(J/4)*4=0 THEN PRINT FF$:PRINT:
   GOTO 330
320 PRINT FF$,
330 NEXT J
340 FIELD #2,32 AS Q$
350 LSET Q$=A$:P=P+1:PUT #2,P
360 NEXT I
370 CLOSE
380 PRINT:INPUT“继续(Y/N)?”,K2$
390 IF K2$="N" OR K2$="n" THEN 470
400 CLEAR:PRINT
410 GOTO 10
420 IF ERL=30 OR ERL=40 THEN
   GOTO 50 ELSE GOTO 450
430 RR$=INKEY$:IF RR$="" THEN 430
440 GOTO 20
450 PRINT:PRINT“有一些错误!”
460 GOTO 980
470 CLOSE
480 PRINT:INPUT“是否将字模转换成纵向点阵(Y/
   N):”;ZZ$
490 IF ZZ$="N" OR ZZ$="n" THEN 720
500 DIM BB$(32),CC$(32)
510 OPEN "CCM.DOT" AS #1 LEN=32
520 FIELD #1,32 AS A$
530 L=LOF(1);L=L/32
540 OPEN "ZMZ.DOT" AS #2 LEN=32
550 P=0:FIELD #2,32 AS Q$
560 PRINT:PRINT“正在转换成纵向点阵!请等待……”
570 FOR I=1 TO L
580 GET #1,I
590 M=1:FOR J=1 TO 31 STEP 2
600 BB$(J)=MID$(A$,J,1):CC$(M)=BB$(J):M
   =M+1
610 NEXT J
620 FOR J=2 TO 32 STEP 2
630 BB$(J)=MID$(A$,J,1):CC$(M)=BB$(J):M
   =M+1
640 NEXT J
650 W$="" :FOR N=1 TO 32
660 W$=W$+CC$(N)
670 NEXT N
680 LSET Q$=W$:P=P+1:PUT #2,P
690 NEXT I
700 CLOSE
710 PRINT:PRINT“转换完毕!”
720 PRINT:INPUT“是否将字模转换成 ASCII 码存贮?
   (Y/N):”;YY$
730 IF YY$="N" OR YY$="n" THEN 1010
740 OPEN "ZMZ.DOT" AS #1 LEN=1
750 IF LOF(1)<>0 THEN 770
760 OPEN "CCM.DOT" AS#1 LEN=1
770 OPEN "ASCM.DOT" FOR OUTPUT AS #
280 FIELD #1,1 AS U$
790 L=LOF(1)
800 PRINT:INPUT“需要首末地址吗?(Y/N):”;XX$
810 IF XX$="N" OR XX$="n" THEN 880
820 LL$=HEX$(L)
830 IF LEN(LL$)=4 THEN 870
840 N=4-LEN(LL$):FOR I=1 TO N
850 LL$="0"+LL$
860 NEXT I
870 PRINT #2,"0000";LL$
880 PRINT“正在转换 ASCII 码!请等待……”
890 FOR I=1 TO L
900 GET #1,I
910 U=ASC(U$):A$=HEX$(U)
920 IF LEN(A$)>1 THEN 940
930 A1$="0":A2$=A$:GOTO 950
940 A1$=MID$(A$,1,1):A2$=MID$(A$,2,1)
950 PRINT #2,A1$;A2$;
960 NEXT I
970 PRINT:PRINT“转换完毕!”
980 CLOSE
990 PRINT:INPUT“退出吗(Y/N)?”;WW$
1000 IF WW$="N" OR WW$="n" THEN 10
1010 END

```


烟台大学计算机应用开发中心 让您的微电脑一展其音乐才能

由我中心开发成功的 SFSD—1 型微电脑乐曲播放系统,是集微电脑技术与电子音乐技术于一身的最新产品。

当您久坐在电脑屏幕前紧张工作感到单调乏味时,一曲美妙动听的音乐将顿使您一扫疲倦,精神焕发,效率倍增。

系统构成

* 硬件

主机—IBM PC 机或各种兼容机一台

外设—SFSD 微电脑放音机一台

* 软件

存放音乐驱动程序的软盘若干套

性能简介

* 构成微电脑彩色音乐系统,能播放各种中外名曲(流行乐、轻音乐、舞曲及背景音乐)

* 具备卡拉 OK 伴奏功能

* 演奏曲目及演奏顺序可由用户自编辑

* 网上用户可互相点播名曲进行音乐交流

* 安装使用方便(无须装卡,菜单选曲)

* 价格便宜,面向各界、服务大众

购货须知

* 与我部取得联系,索取订单

* 告知贵方主机类型(如 286/16M、386/20M 等)

* 说明音乐曲目要求,如发行盘、定制盘(用户点曲)等

* 商谈交货、付款等事宜

联系地址

* 邮编:264001

* 地址:烟台市迎祥路 11 号付 3 号

烟台大学计算机应用开发中心

联系人

* 经营部:肖万春 刘元晓(电话:0535—226252 213726)

* 技术部:李凯里 李纲民(电话:0535—248995—746)

51 单片机学习及应用装置

北京工业大学 裴 巍

51 单片机学习及应用装置是适用于高等院校单片机教学、单片微机技术培训和小型单片机化产品研制开发的一种通用装置,其结构简练,功能完整,操作容易方便,且加上+5V 稳压电源即可正常工作,不受其余条件限制。

整机原理见图 1,它由 8031 单片机系统、数据存储 6116、可编程并行 I/O 芯片 8255、键盘及数码管显示器等组成。

监控程序存放在 EPROM 2732 之中,占有 0000H~0FFFH 的地址空间。

用户程序或数据均存放在 6116 之中,其地址为 1000H~17FFH。其中最后的 48 个 RAM 单元 17D0H~17FFH 为监控程序所占用,它们是一些系统标志单元、断点地址及操作码存放单元等,用户不应在这些单元中存放自己的程序和数据。

17F8H~17FDH 是六个数据显示缓冲单元。

图中可见,当 P 2.4 为“1”时,经反相以后选通 6116 芯片。由于 \overline{OE} 线是 \overline{RD} 和 \overline{PSEN} 相“与”以后提

供,无论 \overline{RD} 有效还是 \overline{PSEN} 有效都能使 6116 的 \overline{OE} 有效,这样,6116 就能够既作为程序存储器、也作为数据存储器来使用。

8255 的 PA 口作为字型输出口,输出的字型码经锁存和驱动后加到数码管的各显示段上。PB 口作为字位输出口(同时也是键盘的行扫描输出口),输出的字位信号也经锁存以后,再反相驱动输出到数码管的字位端。键盘信息是通过一只总线缓冲器 74LS125 在 PC_i 有效时送到 PC 口的低四位来获取的。

显示器由六只 LED 数码管构成,工作时与键盘操作配合显示各种状态,如左边一位可以标示某些特殊功能寄存器;A 累加器、B 寄存器、D 数据指针等;二位可构成片内 RAM 单元地址显示;四位可构成程序或数据存储地址显示。右边二位通常为十六进制数据 display;四位可以是 DPTR 数据指针、PC 程序计数器的显示等。

除复位键外,键盘部分共含有 24 个按键。其中有 16 个是数字键为 0H~FH,用于向 CPU 输入存储单

元地址、寄存器地址或代码、指令码以及数据。

8个功能键分别为：

- MEM 存储单元读/写键
- REG 寄存器及内部RAM读/写键
- LAST 上一存储单元读/写键
- NEXT 下一存储单元读/写键
- BP 设置断点键
- MON 监控程序键
- STEP 单步执行程序键
- EXEC 连续执行程序键

此外，还有单字节插入键和 EPROM 2732 写入键，它们均为复合键。

需要写入的 EPROM 2732，其数据线、地址线均由 8255 的 PA、PB 和 PC 口提供。按动相应功能键可以将用户存储区 1000H~1700H 的 2K 数据写入 2732 芯片的 0000H~0700H 存储单元中。

本机直接运行 MCS-51 单片机机器语言。系统只占用 $\overline{INT0}$ 一根线，其余资源用户均可使用。在监控程序中固化有 12 个常用子程序可以调用。为了方便实

验和使用，还例举了十个实验程序如电子时钟、计数器、子程序调用、仿真调试等以利参考。

配备的 40 芯线仿真插头可便于连线样机进行仿真调试。

51 单片机学习及应用装置体积小，性能稳定，精巧美观，价格低廉，能够满足各种需要，它是广大师生和工程技术人员学用单片机、研制开发单片机产品实用有效的工具。

有关“51 单片机学习及应用装置”咨询及邮购请与北京海声工业控制系统工程公司卞雨池同志联系。

价格：460 元/台 邮费：15 元

开户行：工商银行海淀区双榆树信用社

帐号：003148—28

电话：255.4603

信箱：北京 8703 信箱

邮编：100080

地址：北京中关村路南口

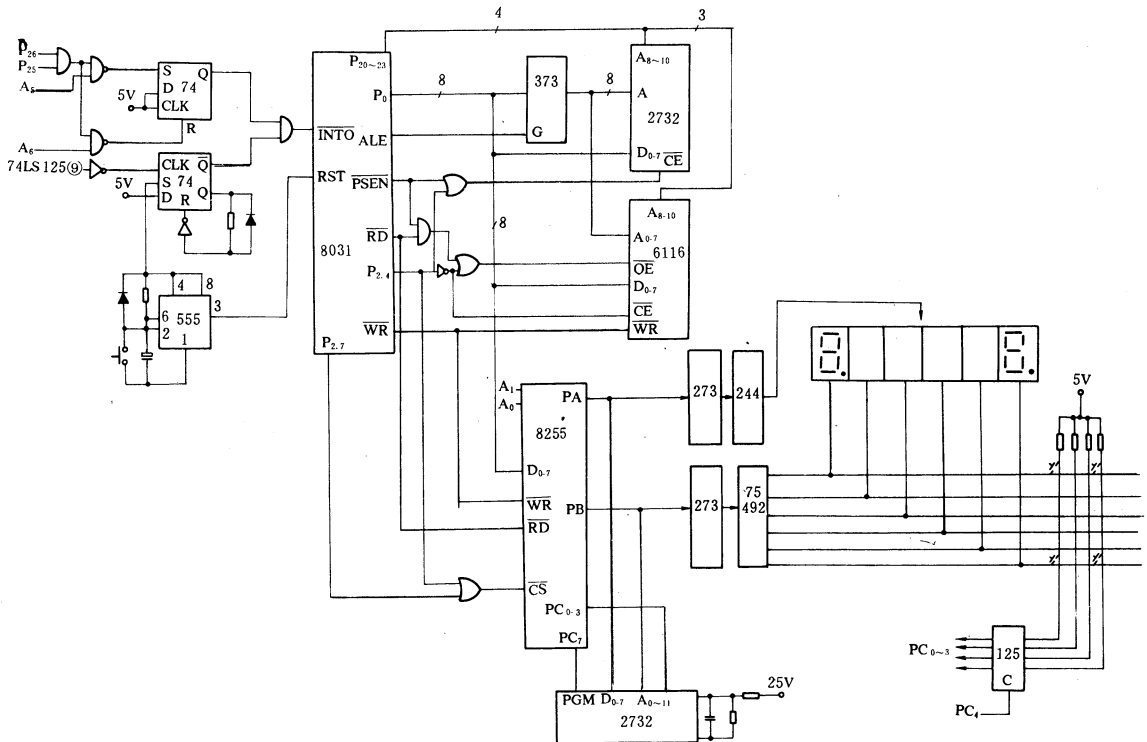


图 1

IBM—PC 总线扩展技术和开放工作台

合肥中国科技大学(230026) 朱世鸿

微处理器的出现是 70 年代微电子技术发展的重大成果之一,由于 IBM—PC 个人计算机除具有十分丰富和功能很强的软件资源外,IBM 公司还将其精心设计、使用十分方便的总线结构全部资料公开,这即等于邀请全体用户利用总线设计各种应用卡。虽然目前总线标准很多,但实践证明用 IBM—PC 微机解决实际问题的能力远高于 80 年代盛兴一时的 STD 等总线,随着 PC 机性能/价格比的提高,利用 PC 机完成各种处理和控制在一种最佳方案,因此国家已将推广使用 PC 机作为八五规划的首要任务。

PC 总线包括 PC/XT 总线和 PC/AT 总线,它们之间存在差异。PC/XT 总线是 62 总线结构,而 PC/AT 总线是 98 总线结构。IBM—PC 微机内只有五个总线插槽,IBM—PC/XT 有八个总线插槽。在上述总线插槽中,由于已被显示适配器、软、硬盘适配器、打印适配器和串行通信适配器占用,因而在实际使用中

户的仍感由微机自身提供的插槽不足。另外,目前市场虽然有较多标准的应用卡,但由于用户广泛性,因而用于总线插槽中应用卡的功能相差很大,致使许多用户必须自己设计专用功能的应用卡,这样用户即遇到一个普遍、实际的问题,即每次对应用卡的调试和检查都要打开主机外壳,并且频繁的关闭电源,使主机损坏可能性增大,特别是在高校微机教学实验中,其问题更加突出,为此我们在总结十余年微机教学和科研工作基础上,并荟萃众人智慧结晶研制出 KD IBM—PC 开放工作台,它可充分利用微机丰富的软件和硬件资源,使用户可随心所欲地进行软硬件设计,特别是工作台提供单独电源,避免因频繁地关闭主机电源和学生实验及科研调试造成对主机的损坏,设计和生产人员可将在工作台上调试好的“应用卡”直接插入微机内部的总线扩展槽中即可工作,为生产和实验提供了方便,可靠的检测调试装置。

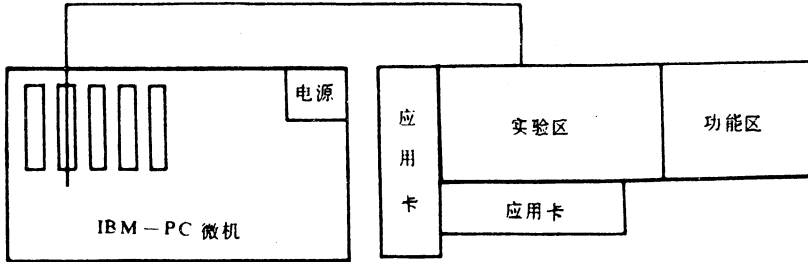
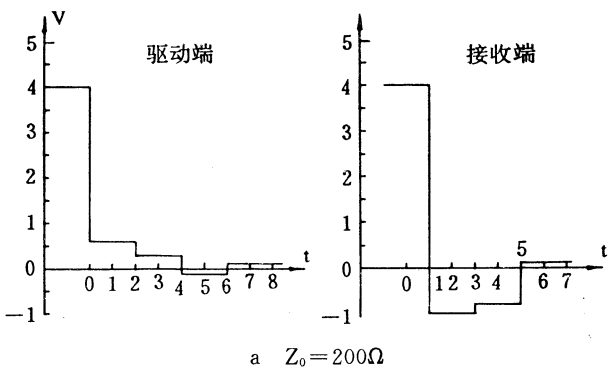


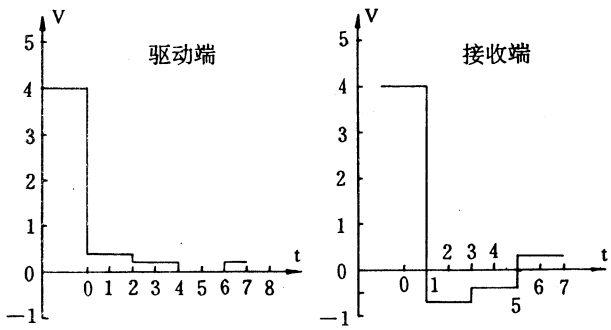
图 1 KD IBM—PC 开放工作台原理图

图 1 是开放工作台的基本原理,它将 PC 机内的总线功能用全开放模式提供给用户使用。根据传输线理论,当用多芯圆导体的带状电缆使传输信号的延迟时间等于或大于信号的转换时间时,必须考虑信号的反射影响。另外,由于用标准 IC 构成的数字逻辑电路,因器件自身阻抗变化的非线性和各种器件型号的离散性,在工程上它们之间的互连线很难严格按照它们自身的特性阻抗进行匹配连接,因而产生反射,使信号衰减,时间延迟往往可同信号的上升沿相比拟,及信号在终端、始端多次产生的尖峰电流使传输信号波形产生“振荡”和“过冲”,并使噪声容限减少。

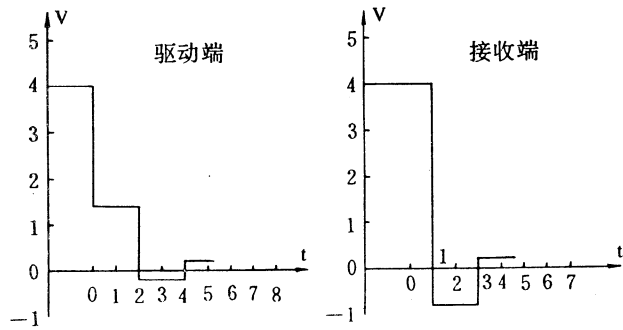
图 2 和图 3 是 TTL 电路,它们之间互连线的线阻抗为 200Ω 、 100Ω 和 50Ω 时,逻辑状态由“1”变为“0”或由“0”变为“1”时驱动端和接收端的波形图。从图中可看出线阻抗减小不能有效地改变由逻辑“1”变为逻辑“0”时的转换特性,但可改变由逻辑“0”变为逻辑“1”时的转换特性。

另外,需要注意传输电缆长度增加,其分布电容也随之增加,当其分布电容值大于某一数值时,该分布电容使前级电路的负载加重,如图 4 所示,它使总线信号严重失真,造成系统内逻辑竞争,导致系统工作紊乱。



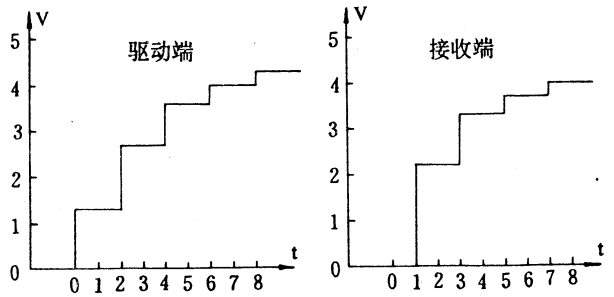


b $Z_0 = 100\Omega$



c $Z_0 = 50\Omega$

图2 逻辑由“1”到“0”转换时电压和时间关系图



c $Z_0 = 50\Omega$

图3 逻辑由“0”到“1”转换时电压和时间关系图

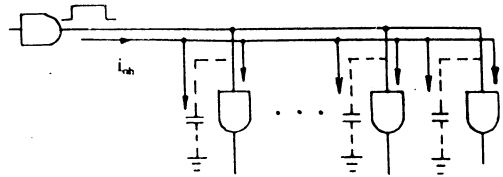
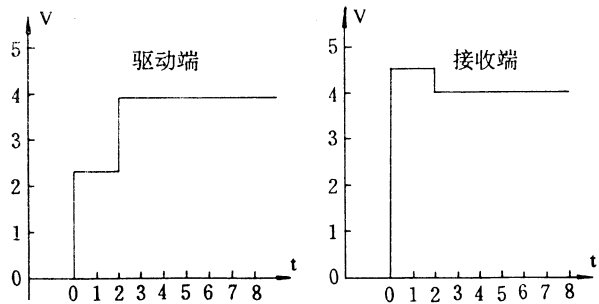


图4 交流负载图

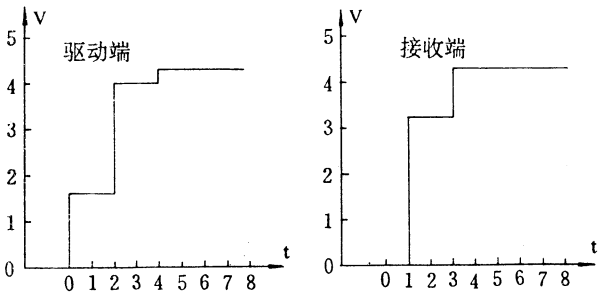
例: 设 t_i 时刻的电压 $U(i)$ 为“0”, 当逻辑由“0”向“1”转换时, $U(i)$ 为:

$$U(i) = \frac{1}{C_{分}} \int_0^T I_i(t) dt$$

若总线中某位驱动电流 I_i 为 2mA 时, 分布电容为 5Pf, 则



a $Z_0 = 200\Omega$



b $Z_0 = 100\Omega$

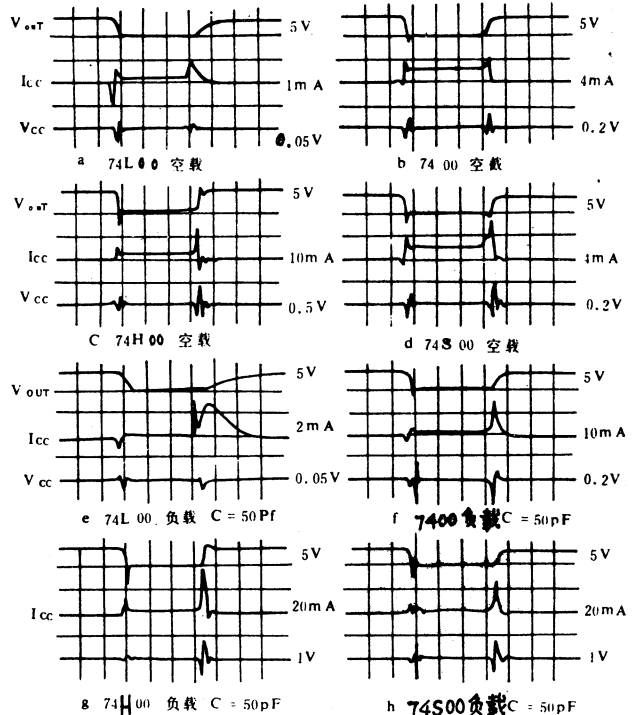


图5 7400系列瞬态 ICC 比较图

注: $R_L = 500\Omega$ 1v/div 2ns/div

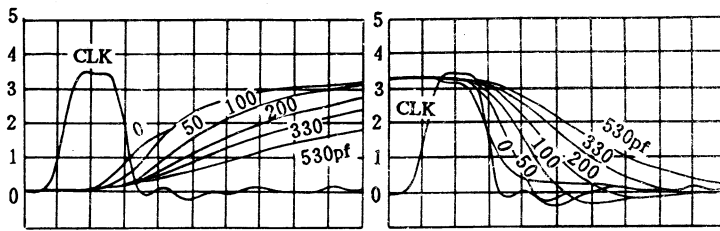


图 6 74AS74 CLK-Q 延迟曲线

$$U(i) = \frac{I_i T}{C_{分}} = \frac{2 \times 10^{-3}}{5 \times 10^{-12}} T = 4 \times 10^8 T$$

根据 TTL 输入状态为“1”时的电平为 $\geq 2V$ ，“0”电平 $\leq 0.8V$ ，输出状态为“1”时电平 $\geq 2.7V$ ，“0”电平 $\leq 0.4V$ ，在留有合适余量后，

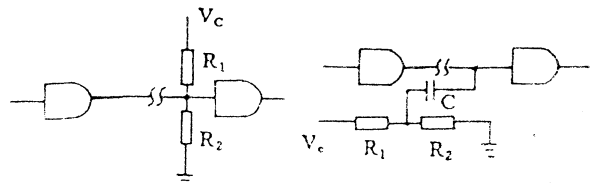
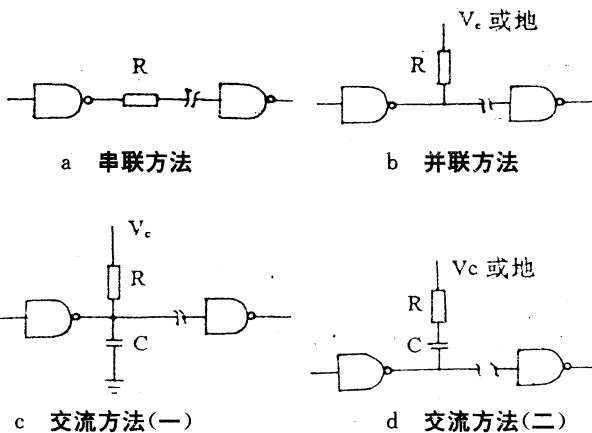
$$T = \frac{U(i)}{4 \times 10^8} = \frac{3.0}{4 \times 10^8} = 7.5 \times 10^{-9} = 7.5ns$$

图 5 是通过示波器实测 7400 系列的 I_{CC} 结果。图 6 是 74AS74 在不同负载电容情况下实测时钟 CLK 和输出延迟曲线 T。

另外，传输总线信号的电缆，当导线中通过电流或电压时，邻线之间的相互串扰也是重要的噪声源，它们是通过静电场、磁场、互感和分布电容产生。

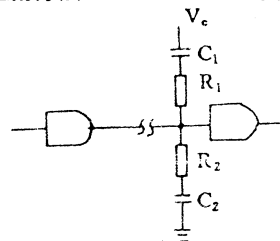
通过上述分析和工程中的实际经验，当传输线工作频率在 MHz 时，长度小于 30cm 时，其影响可忽略不计。但超过 30cm 时，这时由传输线阻抗不匹配造成的反射，通常是有源的匹配方法将传输线上的反射波吸收。图 7 是现在一般采用的匹配技术。分布电容造成前级负载加重，可使用驱动电路减少由传输线产生的信号失真和畸变。

采用上述技术生产的 KD IBM-PC 总线开放工作台，经实践特别适用于各高校微机接口实验，它可完成多种规模的单元和综合实验，是真正提高教学质量、培养学生具有分析、解决问题的实际能力，这是传统教学仪器无法比拟的。

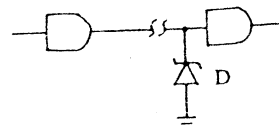


e 戴维南方法

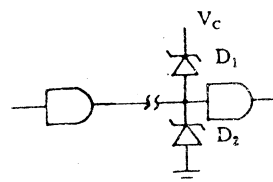
f 交流戴维南方法(一)



g 交流戴维南方法(二)



h 稳压法(一)



i 稳压法(二)

该工作台另外具有 5V/3A、-5V/300mA，±12V/300mA、2~25V/100mA 电源，8421 计数器，12 级二进制分频器，八位 LED，标准信号源，单脉冲发生器，二极管、三极管在线测试仪，实验面包板等多种功能，因而同时也可用于数字逻辑实验和设计，欢迎选购。需要者可同电子工业出版社广州科技公司联系。

联系地址：广州五山路华南师大科技服务楼 215 室

联系人：周青峰 邮编：510630

电话：7504448 转 3872

微型计算机遥控键盘

济南市职工大学 顾正平

一、引言

目前微机键盘,体积大,与主机有连线,只能放在操作台上操作。操作人员长期固定一种姿势工作,容易产生疲劳和得职业病。为此我们研制了一种遥控键盘。这种键盘,体积小,与主机无连线,使微机操作方便、灵活。这种遥控键盘借用了彩电遥控器的发射与接收电路。此类集成电路经功能扩展后可以发射 256 条指令,与目前标准键盘的键值种类相同。而且其抗干扰性能和可靠性很高,很适合作微机控制。以此集成电路为基础,加上功能扩展,单片机数据处理,可组成遥控键盘。

二、遥控键盘系统原理

本系统原理如图 1。第一部分是遥控键盘及红外

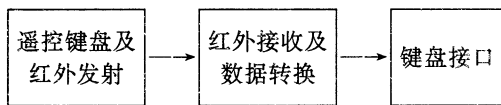


图 1

发射部分。主要由按键和集成发射电路组成。本电路使用了东芝公司的 M50560 集成发射电路。集成发射电路的外电路包括一般彩电遥控器电路:使用了 4×8 键矩阵以及扩展电路。实际电路见图 6,原电路加扩展电路后可以把发射的指令码扩展到 256 条。第二部分是红外接收及转换部分。它把红外发射器发出的以一定频率调制的红外线信号经选频接收、滤波、放大后变为脉冲信号,此脉冲信号送一单片机进行处理和转换为标准键盘信号,然后送微机键盘接口。

先把 M50560 遥控发射电路的功能介绍一下:(1)振荡电路:片内有振荡电路,在 IN(4 脚)和 OUT(5 脚)两端接石英晶体,就可得基准信号。(2)发射指令码:指令码以载波形式发送,其格式如下图 2。在指令码的前面是维持 8ms 的高电平和 4ms 的低电平,然后是 8 位用户码,一种接法固定一种用户码。然后是 4ms 的间隔,最后是 8 位指令码。这是根据按键的不同而不同的。用户码和指令码的“0”码和“1”码的格式见图 3。分别是周期为 1ms 和 2ms 的脉冲。(3)用户码:用户码

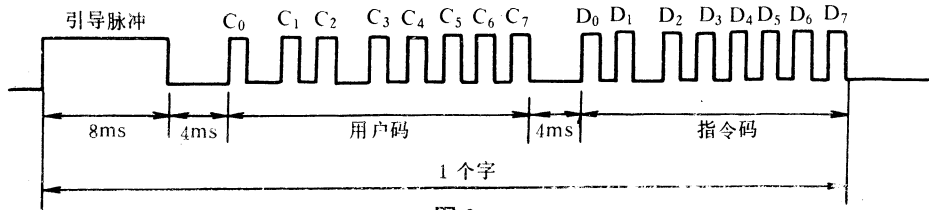


图 2

是由 D 输入端与 $F_0 \sim F_4$ 扫描输出端之间以不同的连接方式可确定不同的用户码。图 4 是扫描输出与输入之间的一种连接和相应的用户码,用户码中的 C_5, C_6, C_7 均固定为“0”。(4)指令码:指令码是通过扫描输入端 $E_0 \sim E_3$ 和扫描输出端 $F_0 \sim F_7$ 之间的不同连接而产生 4×8 种指令码。4 个输入端和 8 个输出端组成 4×8 键矩阵。每次只能按下一个键,按二个无效。因 $32 = 2^5$ 所以产生了 $D_0 \sim D_4$ 5 位指令码。(5)扩展码:D 输入端与 $F_5 \sim F_7$ 扫描输出端之间的各种连接可产生 D_5, D_6, D_7 三位指令扩展码。图 5 是 D 端与 $F_5 \sim F_7$ 端之间的连接方式和相应的扩展码。扩展后的遥控发射电路见图 6,图的上部是一般彩电遥控器的 32 键发射电路,下部的 $T_1 \sim T_7$ 按键和 $D_1 \sim D_{12}$ 二极管是扩展电路。二极管是起隔离作用。256 条指令产生方法是这样的,图 6 中上部,32 个按键每按一个产生一条指令码。这是第一组 32 键。然后按下 T_1 再按 32 键之一,又产生 32 条指令。这是第二组 32 键。这样 32 键配合 $T_1 \sim T_7$ 共可产生 256 指令码。M50560 电路的 C 端是脉冲输出端;输出脉冲经三极管放大后送红外发射管,把指令发射出去。

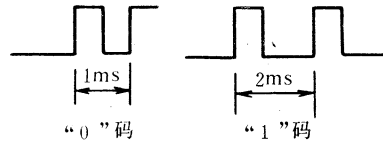


图 3

四、信号转换原理

接收电路是用的 $\mu\text{pc}1490\text{HA}$ 集成接收电路。其外电路接法见图 7,图中 D 是红外接收管,接收红外发射管发出的信号,然后滤波,放大后变为脉冲信号从 2 脚输出,送单片机进行转换。单片机使用 8031,再加一片地址锁存 74LS373 和一片 EPROM2716,组成一单片机系统,见图 8。单片机的软件原理见图 9 程序框图。现对框图作以说明。本流程图分两部分:1. 键盘复位,一般使用软件复位。主机启动时先进入自测试,对键盘

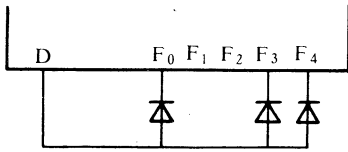


图 4

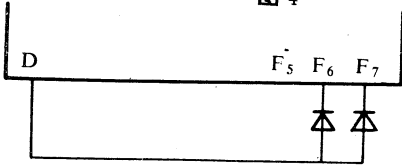


图 5

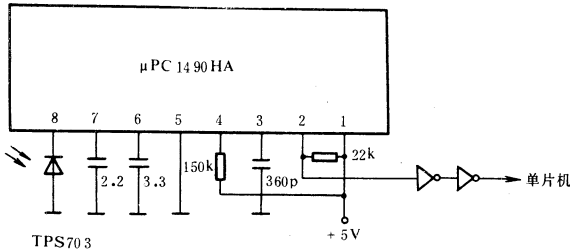


图 7

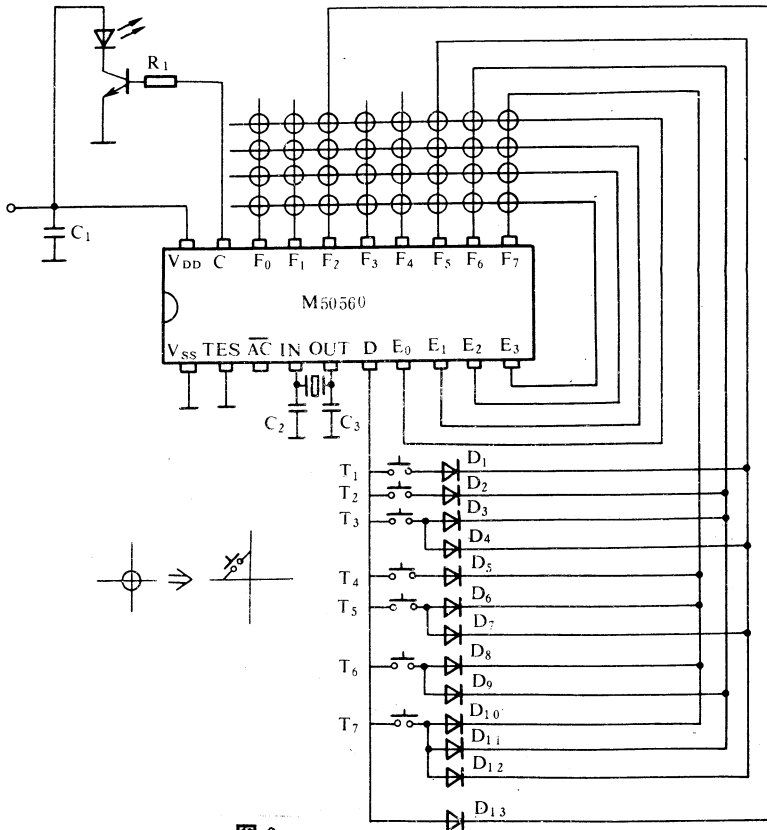


图 6

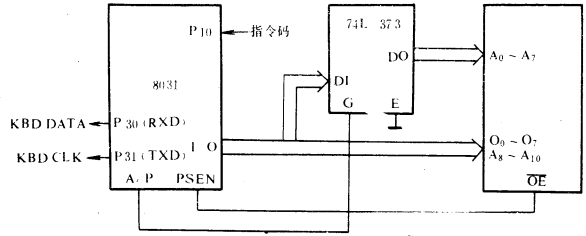


图 8

扩展指令码:
D₅ D₆ D₇
0 1 1

的测试是向键盘时钟线 KBD CLK 送一低电平, 维持用户码:

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

20ms, 键盘接口以此判断为复位命令。键盘应答返回码“AA”。2. 把遥控器发出的指令码转换为标准键盘编码。标准键盘送给主机键盘接口的信号是由 KBD CLK 线发出的同步信号和 KBD DATA 线发出的串行数据信号。这种串行传送方式与 8031 单片机的串行口 0 方式相同, 所以要把单片机的 P3 口设置为串行输出 0 方式。在这种方式时 P3.0 线为串行输出的 RXD 信号, 输出串行数据, 它作为键盘输出数据线 KBD DATA。而 P3.1 线为串行输出的 TXD 信号, 输出同步信号, 它作为键盘输出线 KBD CLK, 见图 9。把指令码转换为键盘输出码的软件的编制方法, 要根据图 2 遥控器发出的指令码的格式。先搜索到引导码, 即: 8ms 的高电平, 4ms 的低电平。然后接收用户码, 最后才是指令码。这就是这部分程序的逻辑。这部分流程图先搜索大于 7ms 的高电平, 通过后再检测是否是大于 3.5ms 的低电平, 然后是一双重循环, 外循环 2 次, 是为了接收 2 个字节数据, 内循环 8 次, 是为了接收 8 位数据。第一次外循环是接收用户码, 把接收的用户码与定义的用户码相比较, 如相同, 再进入第二次外循环, 接收指令码, 然后将接收到的指令码转换为标准键盘码, 从 P3 口以 0 方式串行输出。

结束语

本遥控键盘抗干扰性能好, 可靠性高, 而且各遥控器可定义不同用户码, 可以搞成多用户系统, 很有发展前途。

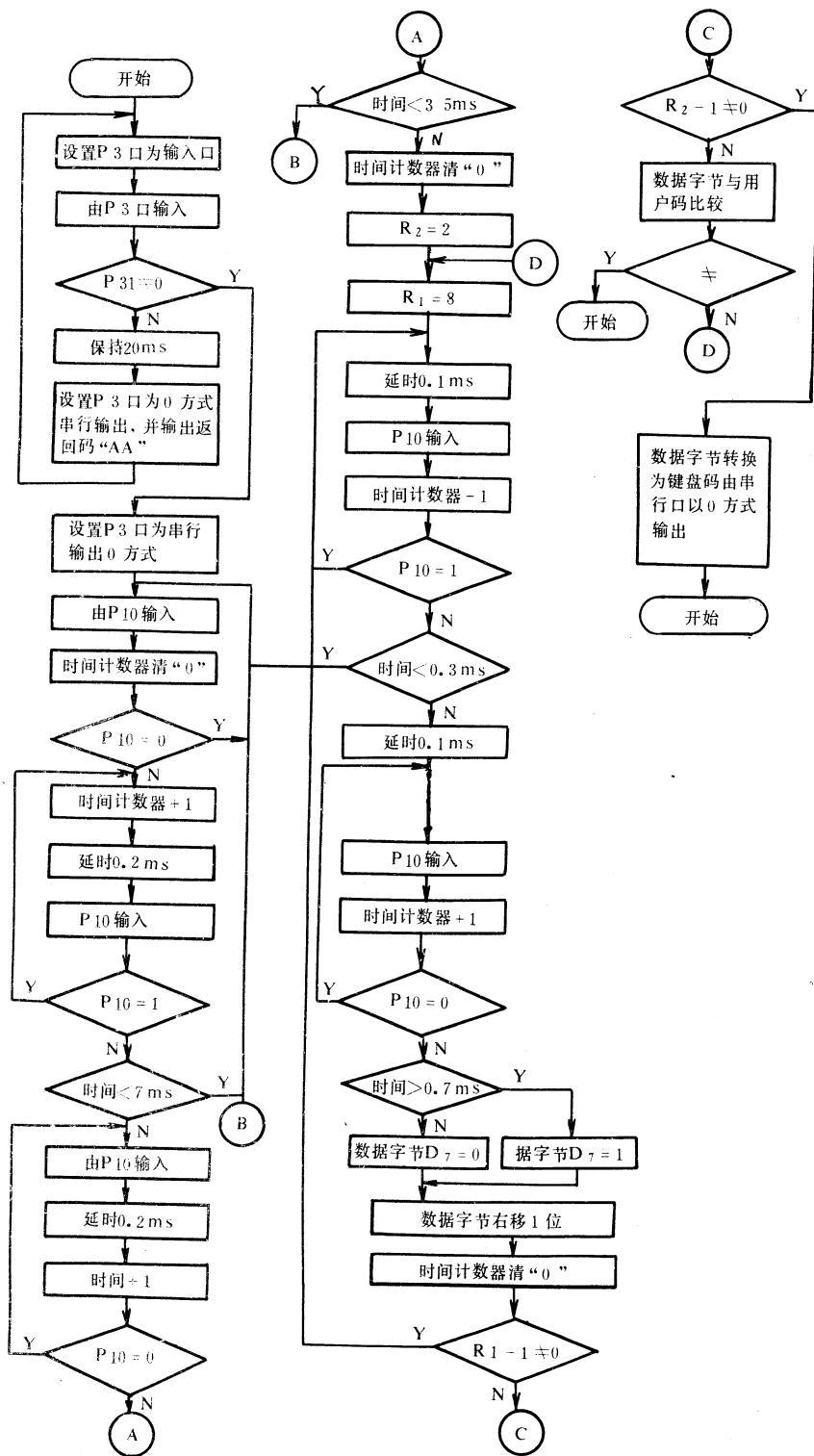


图 9

F BASIC 语言编程特技的深入研究

山东苍山机械电子化学工业局(277000) 于 春

序言

自拙作“电脑游戏机的 F BASIC 语言”及“F BASIC 语言的游戏程序编写技巧”刊出以来,相继收到大量的读者来信。信中提出了各种各样的问题,归纳起来,大体有以下几个方面:

1. F BASIC 的背景画面显示分辨率为 28 列×24 行,而任天堂游戏中的背景画面显示分辨率却为 32 列×30 行(与底背景画面一样大)。相比之下,F BASIC 的显示画面仅为任天堂游戏画面的十分之七。因此,用 F BASIC 语言设计的游戏画面远不如任天堂游戏画面的内容丰富。既然 F BASIC 是一种面向游戏的语言,那么能否在 F BASIC 状态下编写显示 32 列×30 行的画面程序呢?

2. 在大多数任天堂游戏中,如《魂斗罗》、《绿色兵团》等,音响、卡通(或称为角色)可以同步控制,即在背景音乐伴奏的同时,卡通可以做着各种复杂的动作。但用 F BASIC 语言编写的游戏程序,音响的发出和卡通的动作却是分步进行的,即在卡通动作时不能发声;发出音响时角色的动作要停下来,直到音响结束后才能继续动作,那么,能否使用 F BASIC 语言编写出音响、角色同步动作的程序呢?

3. 在任天堂游戏中,游戏画面可以连续左、右卷动或上、下滚动。这种游戏语言编写的游戏程序却不能进行背景画面的卷动、滚动,只能一场一场地更换背景,从而使游戏单调、呆板。再如在《淘金者》二代的游戏中,每一局的画面有 60 列×48 行,可按暂停键(START 起动键)使游戏暂停后,通过方向键上、下、左、右移动画面截取任一部分 32 列×30 行的画面。为观察整个游戏画面提供了方便。既然任天堂游戏和 F BASIC 都使用同一个 cpu,那么能否用 F BASIC 语言实现上述功能?

4. 任天堂游戏中有着丰富逼真的效果音响,如飞机起飞、降落、射击、爆炸、开门、关门等声音。还有自然界中听不到的“宇宙音”。使用 F BASIC 语言虽然也能模仿部分效果音响,但总不如任天堂游戏中的音响逼真动听。尤其是模仿诸如警报声、引擎声等类音响,使用 PLAY 指令则很难实现。那么,任天堂游戏中的这些音响是如何发出的?能否在 F BASIC 系统中实现?

5. F BASIC 系统中,卡通角色最大为 16×16 点阵。而任天堂游戏中的角色大多是 32×32 点阵的,有的甚至是 32×64 点阵。它们是怎样定义的?任天堂游

戏中有各种各样的卡通角色和背景图形,F BASIC 系统中为什么没有这么多?

以上问题一方面集中反映了朋友们强烈的求知欲望,另一方面也暴露了 F BASIC 语言存在的种种不足与缺陷。

为满足朋友们的愿望,弥补 F BASIC 语言目前存在的种种不足,本文拟从简要介绍任天堂游戏机 6527 cpu 工作系统的特点入手,讨论在 F BASIC 工作状态下,以最小的程序量实现:32 列×30 行背景图形的绘制;64 列×30 行图形的显示;背景画面的连续卷动和滚动;音响、角色同步工作以及各种效果音响的发出等等,并简要介绍各种卡通块、背景图形块的结构及定义方法。使读者能够使用 F BASIC 语言编写出与任天堂游戏相同效果的游戏程序片断,进而通过扩展内存,自己亲手编写出完整的游戏软件。

另外,任天堂游戏机中的 6527 cpu 既是中央处理单元,又是一个八位单片机,它独特的工作系统固化在 cpu 内部,一般情况下很难读出研究,又加之有关 6527 cpu 的资料较少,因此人们对 6527 cpu 的了解尚处于探索研究、逐步认识阶段。本文的目的并不在于全面系统的介绍 6527 cpu 的工作系统(目前也不可能),而是给出研究它的一般方法,提供一条研究 6527 cpu 的捷径。所以仅以特点的方式介绍 6527 cpu 的工作系统,文中所述内容则是近年来国内同仁的研究成果和笔者的一点研究心得,错误与不当在所难免,欢迎广大同仁验证、斧正。

第一章 6527 cpu 工作系统的特点

任天堂游戏机实际上是一台电脑裸机。由游戏机电路原理图(《电子与电脑》91 年第三期封底)中可以看到游戏机的主控电路板主要由中央处理器 cpu、图像处理器 ppu、两片 6116 动态随机存储器 RAM 和门电路等组成。游戏机工作时,要靠插入 60 脚扩展槽上的游戏软件支持;同样,当插入学习软件 BS 卡时,就可以通过键盘进行电脑学习。可见扩展槽中插入不同的软件就可使游戏机具备不同的功能。所以任天堂游戏机是一台使用极为灵活的裸电脑。

任天堂系列游戏机有多种型号,所使用的 cpu、ppu 型号也由于产生厂家的不同而差异较大。但是它们的功能都是相同的。因此,我们把任天堂系列游戏机的 cpu 以最有代表性的型号 6527(或 6527P)命名,并把由 6527 cpu 和 6528(或 6528P) ppu 组成的硬件系

统称为 6527 cpu 系统,相应地把支持它们协调工作的内部软件系统称为“6527 cpu 工作系统”。

一、6527 cpu 的基本特征

6527 cpu——电脑游戏机的中央处理器,它是一个八位 NMOS cpu,也是一种八位单片机。它具有八位数据线,十六位地址线,直接寻址 64K 字节,时钟频率为 26.601712MHz,每秒钟可执行约 500 万次八位数据运算。由于 6527 cpu 仍属 65 系列微处理器,故它具有 65 系列 cpu 的基本特征,但又具有独特的特点。与中华学习机的 6502 cpu 相比,6527 cpu 的运算速度快(6502 cpu 的时钟频率仅有 1MHz)、工作效率高。另外,由于 6527 cpu 又是一个八位单片机,故它的操作解释程序较简洁。因此,6527 cpu 的性能远优于 6502 cpu。

1. 6527 cpu 的指令系统

6527 cpu 的指令系统与 6502 cpu 的指令系统完全兼容,故凡学习过苹果机、中华学习机的 6502 机器语言的朋友,完全可以驾驭 6527 机器语言。没有学习 6502 机器语言的朋友可参阅《电子与电脑》1992 年 1 月号连载的朱国江老师撰写的 6502 机器语言讲座,定可很快领会掌握,因为在以后的讨论中要经常涉及机器语言程序的设计,故掌握 6527 机器语言将有助于编程特技的学习和提高。

2. 6527 cpu 的内部寄存器

同 6502 cpu 一样,6527 cpu 内部也有六个寄存器(A、X、Y、S、P、PC),其中除程序计数器 PC 是十六位寄存器之外,其它均为八位寄存器。各寄存器的功能、作用、特点可参阅《6502 机器语言讲座》中的介绍。

3. 6527 cpu 的复位

6527 cpu 的复位地址存储在 \$FFFC、\$FFFD 两个单元中,开机后,系统自动执行由 \$FFFC、\$FFFD 指定地址的复位程序。一般复位地址入口指向零页的 \$00ED。BS. 2A 版本的复位程序由 \$00ED、\$00EE、\$00EF 三个单元控制转向 \$C400 复位程序入口。

4. 6527 cpu 的中断

6527 cpu 的中断处理程序的入口地址放在 \$FFFA、\$FFFB 两个单元中。它只有不可屏蔽中断 \overline{NMI} 一种中断方式。中断请求信号由 6528ppu 的第 19 脚发出,在场回扫的消息期间触发。中断频率为 50HZ,中断间隔为 0.02 秒。

5. 6527 cpu 管理的内存分布

6527 cpu 在 F BASIC 系统中地址分布如表一。

表一 6527 cpu 管理的内存分布表

地址范围	用途
0000~00FF	系统零页
0100~01FF	系统堆栈区
0200~02FF	卡通页显示映射区(卡通定义区)
0300~03FF	F BASIC 解释程序暂存区

0400~047F	工作数据暂存区
0480~04FF	功能键定义区
0500~05FF	键盘输入暂存区
0600~06FF	键盘扫描、定义卡通、屏幕显示标志工作区
0700~07FF	屏幕显示缓冲区
0800~1FFF	空区。映射于 \$0000~\$07FF
2000~3FFF	与 ppu 相连的 I/O 区
4000~5FFF	cpu 的 I/O 区
6000~7FFF	8K 字节空区。BS. 2A 使用 \$703E~\$7FFF 为用户程序存储区。
8000~FFFF	软件 ROM 区,共 32K 字节。对于容量大于 32K 的软件,在此区间有存储体切换。

二、6528 ppu 的基本特征

6528 ppu——电脑游戏机图像处理,它在电脑游戏机中主要用于处理图像视频信号。任天堂游戏机中就因为引进了 ppu,才实现了声、像同步的多种功能,使一个 cpu 完成两个 cpu 的工作。

ppu 有八位数据线,十四位地址线,可直接寻址 16K 字节空间,时钟频率一般为 21~27MHz,故图像处理速度极快。

ppu 管理的内存分布如表二所示。

表二 6528 ppu 管理的内存分布表

地址范围	用途
0000~0FFF	图形库第 I 区,共 4K 字节空间,存储 256 个 8×8 点阵的卡通图块数据。
1000~1FFF	图形库第 II 区,共 4K 字节。存储 256 个 8×8 点阵背景图形块。
2000~23BF	背景第一页(00 页)显示区。每个单元存储一个 8×8 点阵图形块。
23C0~23FF	背景第一页配色区。每一个单元存储 16 个图形块的配色代码。
2400~27BF	背景第二页(10 页)显示区。
27C0~27FF	背景第二页配色区
2800~2BBF	背景第三页(01 页)显示区。
2BC0~2BFF	背景第三页配色区。
2C00~2FBF	背景第四页(11 页)显示区。
2FC0~2FFF	背景第四页配色区。
3000~3EFF	空区。映射于背景显示区。

(转 42 页)

用电流环路实现远程快速数据传输

袁学文

一、引言

随着微型计算机应用领域的日益广泛,计算机之间的通信业务也越来越多。微型计算机本身配置的异步串行通信卡,在电压串行通信接口方式下,最大传输距离小于 100m。为了实现局部地区(如矿山、机场、学校、医院等单位)范围内的数据通信,一般都需要借助于基带传输器或调制解调器,而这些设备一般每台需要几百元、甚至几千元。如果采用 RS-232C 接口卡上已经设置的电流环路接口,或对异步串行卡进行适当改进,或者干脆制作专用的电流环路通信卡,进行电流环路数据传输,不仅可以大大降低成本,而且可以实现 10km 范围内,最高达 115200bit/s 的高速数据传输,并且可以保证足够低的误码率,满足许多通信业务的实际要求。

二、用通用异步串行卡实现电流环路通信

IBM PC 或 IBM PC/XT 型微机配置的异步串行通信卡上,大多本身带有电流环路接口,通过跳线组开关供用户选择。它们一般是为了与电传打字机等设备连接而设置的,但将其用于计算机之间的点对点通信同样是合适的。图 1、图 2 分别为电流环路接口的数据发送部分和数据接收部分的电原理图。图中省略了 INS8250 可编程通用异步串行通信芯片及其控制电路部分,它们主要完成将计算机并行数据变成可供异步串行传输的脉冲序列,或者将接收到的脉冲序列,变成计算机能识别的数据供计算机处理。按图 3 所示连接,并将接口卡上 16 脚跳线组缺口朝下,便可以实现电流环路通信。

三、通信软件的设计方法:

电流环路通信程序与电压异步串行通信程序基本相同。为了获得高速数据传输,最好采用宏汇编语言编写通信程序。首先需要根据通信协议对 INS8250 各控制寄存器进行初始化;对 3F8H(波特率因子低位字节)、3F9H(波特率因子高位字节)赋对应波特率值(此时,要求 3FBH 的 $D_7=1$);对 3FBH 赋值决定数据位数、停止位数;对 3F9H 单元赋值约定是否采用中断方式(此时要求 3FBH 寄存器的 $D_7=0$)等。波特率因子值由下列公式给出:

$$BDVH = \text{INT}(BDV/256)$$

$$BDVL = BDV - BDVH \times 256$$

其中 $BDV = 115200/\text{BAUD}$, BAUD 为波特率数。

如果程序采用询问方式,其数据发送程序流程图如图 4 所示。接收程序同发送程序对应,故省略。

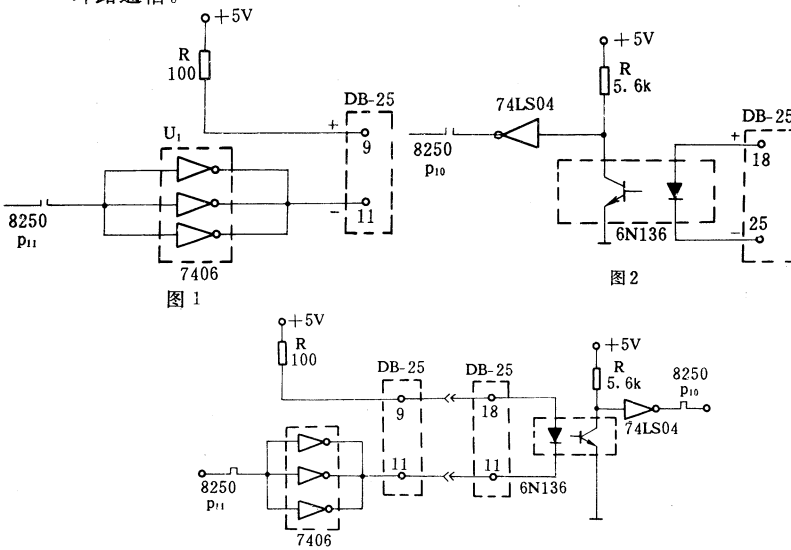


图 3

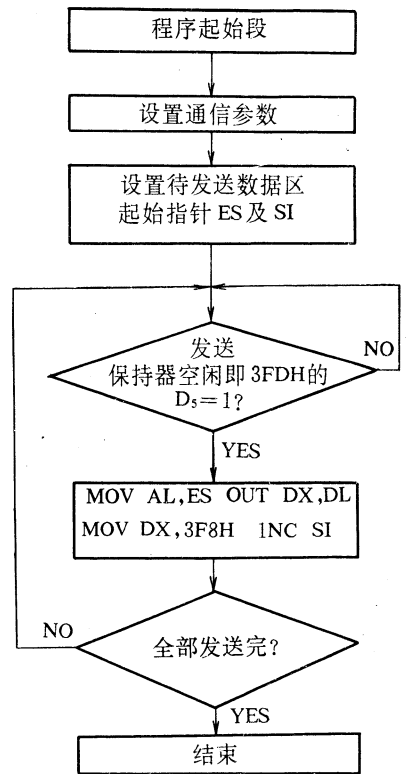


图 4 发送程序流程图

INS8250 有十个可寻址寄存器供 CPU 读写, 实现与外界的数据通信、制订通信协议和提供通信状态信息。各寄存器的详细介绍请参阅《IBM PC/XT 硬件手册》。

四、通用异步串行通信卡的改进

按图 3 所示连接的线路, 只能实现二线单工或者四线双工通信, 对于相距较远而又需要双工通信的通信业务来说, 势必多增设一对线路, 因而大大增加了成本。而且不管有没有数据发送, 只要双方计算机在工作, 发送端与接收端都构成了电流环回路, 影响了光电耦合器件的使用寿命, 并且也不利于实现一对多点的通信。解决的办法是对接口卡进行改进, 即引出 INS8250 芯片第 32 脚 $\overline{\text{RTS}}$ 信号, 来控制一小型继电器 (如 JRX30F 型等), 让数据接收电路与发送电路通过继电器触点分时接入通信外线, 便可实现 2 线半双工通信, 从而节省一对线路。接收电路接常闭点, 发送电路接常开点。只有当要进行数据发送时, 才控制继电器吸合, 使发送电路通过触点接入外线, 数据发送完毕, 立即断开。在通信程序中, 通过对 INS8250 芯片的 MODEM 控制寄存器 3FCH 赋值 03H (发送时) 或 01H (接收时) 来控制继电器触点动作。电原理图如图 5 所示。

如果通信线路是普通双绞线, 其阻值可能较大, 你可以短接图 1 电路中的电阻 R, 或将电源电压改接到 +12V 上, 并相应提高 U1 的输出脉冲电平。

如果你只有不带电流环路接口的异步串行卡, 也可以在 DB-25 芯扁平电缆接头外接一电流环路驱动和接收电路, 而实现电流环路通信, 其电原理如图 6 所示。不过这种通信方法, 因为通过了多次电平转换, 增大了信号延时, 故不能获得太高的传输速率。

五、结束语

用电流环路实现计算机之间的点对点、一对多点的传输, 不仅可以实现较远距离, 而且可以获得很高的传输速率, 因此, 它越来越受人们重视。

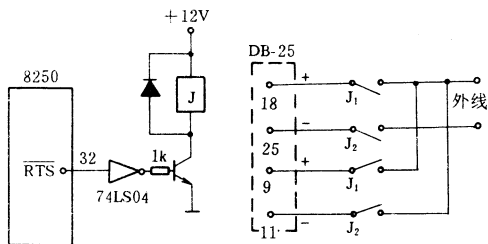


图 5

加速 WPS 的实现

邓海白奉礼

WPS 文书处理软件以其强大的功能, 操作简便赢得广大微机用户的青睐。但使用软汉字版 WPS 存在的主要缺陷为速度较慢, 换页及选用下拉菜单时存在闪烁现象, 不利于视力。以前也有些尝试, 但多过于复杂。今经多次试用, 找到了一种简便的加速方法。

我在微机上安装了一套 Windows 3.0 操作系统及一套 WPS2.1 文书处理系统, 意外发现 Windows 的系统配置可使 WPS 加速。CONFIG.SYS 文件中调用了两个 SYS 文件: HIMEM.SYS 及 SMARTDRV.SYS。在 CONFIG.SYS 文件中加入如下两条语句:

```
DEVICE=HIMEM.SYS
DEVICE=SMARTDRV.SYS 320
```

可使 WPS 运行时换屏速度明显加快, 接近于装有汉卡的机器。

由于我对微机原理及 Windows 操作系统不太熟悉, 不能够对其内部机理做一详述。今以本文抛砖引玉, 希望广大微机爱好者给予指教。

我是一名电脑迷, 无奈条件有限, 想通过“读者联谊”这个栏目求购旧的 PC 机或 286 机、打印机各一台, PC 机要求内存 640K 以上, 10M 以上硬盘及单显, 价格在 1.000 元左右, 286 机要求内存 1M、20M 以上硬盘, 1.2M 软驱及高分单显, 101 键盘, 价格在 1.500 元左右, 最高不超过两千元, 打印机要求 9 针或 24 针打印机, 1.000 元以内。

通讯地址: 大庆石油管理局第二机械厂财务科李春浩
邮编: 163411

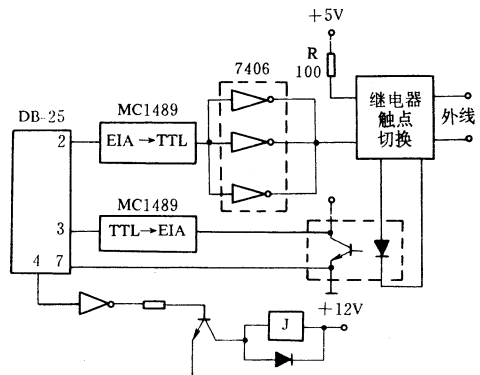


图 6

求购二手便携机或笔记本 (286 或 386, 带 20-40M 硬盘)。

来函请寄: (610072) 四川省社会科学院新闻传播研究所林之达转沈云刚。

传真机的安装和使用

一、安装环境要求

- (一)传真机应避免安装在阳光直射的地方;
- (二)不要安放在空调设备旁边,应与有电磁幅射的设备,如电视机、无线电发射机等保持一定的距离。
- (三)不要安装在潮湿、灰尘多以及振动剧烈的环境中。

(四)安装时,机器距墙 10 厘米或更远。保证传真机通气孔的正常通风,切勿放在高低不平或有斜度的桌上。

(五)传真机上不应搁置东西,如花瓶、水杯等,忌置于高温处。

(六)记录纸应妥善保管,不要放在潮湿,高温的地方,不要让阳光直射,以免影响记录质量。

(七)请勿与瞬间功耗大的机器使用同一电路。

二、安装

(一)确认传真机的电源电压交流 220V 还是交流 110V,避免损坏机器。

(二)根据传真机所需电源电压接入电源。

(二)安装附件、原稿托盘和原稿退出托盘及记录文件托盘。

(三)安装记录纸。

(四)外线及电话连接。一般机器的侧面或背面标有外线和电话连接处,“LINE”接外线,“TEL”接电话。

(五)机器附件安装完毕,接通电源进行设置和自检。根据传真机的功能进行功能参数设置,不同型号的传真机有不同的设定方法,有改变 RAM 的参数方式,有采用开关设置机器参数的方式。安装时根据所安装的传真机的参数设定方法操作。

(六)自检。当参数设置完后,就可以自检。自检一般通过复印、对通的方式进行。

1. 复印

打开机器电源,取一张标准样张,放在原稿托架上,按下复印键“COPY”就可复印。复印完毕,检查样张,然后改变对比度,扫描线密度进行复印,检查其复印件,清晰度应达到 90% 以上,可读度应达 99% 以上。

2. 联机试验,与标准机接通。没有标准机,其它正常机器也可以。然后双方收发若干张,来检查机器的显示、发报、接收功能。

三、对稿件的要求

(一)下列所示的稿件请复制之后再发送或用运载片发送。

1. 压纸或背面复写纸等化学处理的稿件。
2. 有破损,有皱纹卷起和折弯厉害的稿件。
3. 有墨水浆糊未完全干的稿件。

4. 两面稿件均有涂覆层。

5. 太薄或太厚的稿件

(二)安装稿件的注意事项。

1. 自动连续发送时,一次可装到 30 张。请把同样尺寸、质量的稿件排整齐装上。

2. 若夹子、钉书针阻塞在机器内部,会造成故障原因,请把它取出。

3. 挠起的稿件和弯曲的稿件,请把它摆平。特别是比较严重的请用运载片发送。

4. 在发送范围外的文字、图形、记号等不能进行传送,请在规定的有效范围内发送。

四、发送、接收方法

(一)发送

1. 装好稿件

2. 选择扫描线密度。标准 3.85 线/mm,精细 7.7 线/mm 或超精细 15.4 线/mm。

3. 选择原稿对比度。标准、浓、淡。

4. 给收方打电话。当听到对方的回答声“2100Hz”的信号时按下“START”键,确认传真开始,放下听筒。以上操作完毕,装上的稿件完全发送结束时,机器就自动的恢复到待机状态。

(二)接收

接到发方打来的电话,通话后,接收方按下“START”键,接收开始。在双方通信过程中,收发方根据接收发送情况及时和对方联系时,可按下电话预约键或停止键“STOP”。

五、举例 UF—200

(一)基本发送方式的设定:

正常发送之前,用户可临时对分辨率(线密度)、反差(原稿浓淡)、灰度(中间色调)、发送标记、总页数和单独日志打印等六种基本设定项目进行更改。这些更改可以在把文件安放在 ADF 上之前,也可在其之后进行。发送结束时,它们将恢复其事先调定值(只有单独日志打印除外,它将保留于新调定状态)。用户还可以在发送过程中改变这类调定内容,但只有在该页稿件被发送出去后,新调定内容方能生效。

1. 线密度选择:

UF—200 预先调定的分辨率为标准 STD 3.85 线/mm 等级,它适合于正常的稿件发送。发送极为细密的稿件时,请使用精细 FINE 7.7 线/mm 或超精细 SFINE 15.4 线/mm。

(1)按线密度选择键,液晶显示日、月、年、时、分
RESOLUTION=STD

(2)按线密度选择键,液晶显示日、月、年、时、分

RESOLUTION=FINE

(3)按线密度选择键,液晶显示日、月、年、时、分

RESOLUTION=SFINE

如果想要返回标准状态,请再次按动线密度键。线密度等级一旦调定,用户可以按动停键,使机器恢复待机状态。或按动中间色调键,原稿浓淡选择键,发送标记键、模式+键或功能键进行其它系列内容的调定。

2. 原稿浓淡选择

UF-200 预先调定的反差为普通(NORMAL)等级。如果用户必须以黑反差发送稿件,则最好把反差等级调至浓(DARK)。反之,若是浅调发送,则调至淡(LIGHT)。

按下原稿浓淡键,液晶显示日、月、年、时、分

ORIGINAL=NORMAL

原稿浓为普通等级。再按下原稿浓淡键,液晶显示

ORIGINAL=DARK

原稿浓淡为浓等级。第三次按下原稿浓淡键液晶显示

ORIGINAL=LIGHT

原稿浓淡为淡等级。要想返回普通状态,请再次按动原稿浓淡键。原稿浓淡等级被调定,用户可以按动停止键,使机器恢复待机状态,或按动复印键,复制一页文件,以检测调定后的效果,或按动中间色调、发送标记、模式+功能键,进行其它系列内容的调定。

3. 中间色调选择

发送灰调的照片及绘画时,可启用中间色调功能。因为采用此功能发送比较费时,所以 UF-200 的中间色调功能预先调定为非工作状态(OFF)。要使此功能转为工作状态(ON),操作步骤如下:

(1)按下中间色调选择键,液晶显示日、月、年、时、分

HALF TONE=OFF

显示板显示出此功能处于非工作状态。

(2)按下中间色调选择键,液晶显示日、月、年、时、分

HALF TONE=ON

而且分辨率也将被自动调至精细(FINE)状态。要使中间色调功能返回不工作状态,只需再次按动中间色调键即可。

(3)此时,可利用电话/拨号键加完整传真号码、单角键、缩位键加上两个数码键进行拨号、或进行重拨。

4. 发送标记

UF-200 的发送标记,出现在被成功发送稿件的底部,为一粉红色的小印章。这种标记用户可根据需要和不需要,依照以下步骤把发送标记打印调至非工作状态(OFF)和工作状态(ON)。

(1)按下中间色调选择键,液晶显示日、月、年、时、分

STAMP=ON

按下中间色调选择键,液晶显示日、月、年、时、分

STAMP=OFF

要使发送标记功能重新生效,只需再次按动发送

标记键即可。

5. 单独发送日志

若用户想记录所有传真件的单独发送情况,UF-200 可于每次发送结束后打印一份单独的发送日志,以解除手写记录的麻烦。可通过下述方法得到日志。

(1)反复按动模式+或-键,使显示面板液晶显示日、月、年、时、分

XMT JRNL=OFF

(2)按下选择键,显示面板显示日、月、年、时、分

XMT JRNL=ON

要使该功能处于非工作状态,再次按动选择键。

(6)改错再发方式(ECM)

把 ECM 调至工作状态,即可保证传真内容正确。在发送速度的重要性高于接收记录质量的情况下,用户可以把 ECM 调至非工作状态。

(1)反复按动模式键-或+,显示面板显示日、月、年、时、分

ECM=ON

(2)按下选择键,显示面板显示日、月、年、时、分

ECM=OFF

要使该功能恢复工作状态,可再次按动选择键。

7. 直接拨号方法

用户可以使用若干方法进行传真,这里介绍的是两种最基本的方法,即键盘键钮拨号和与 UF-200 联机的电话机拨号。

(1)键盘键钮拨号。将稿件正面朝下安放于 ADF 之上,左右调整两侧的稿件导板,显示面板显示 DOCUMENT SET,按下电话/拨号键,显示面板显示 * DIALING *,利用键盘键钮进行拨号,例:电话号码:5124321,在显示面板显示 * DIALING * 5124321 后将会拨出这个号码。如果线路有空,稿件即会被发送出去,若对方站有识别编码,它将出现在显示板上。如果占线,将以 3 分钟间隔,重拨两次这个号码。发送结束时,被发送的稿件页数将在显示板上出现。显示:

COMPLETED

TOTAL PAGES=05

(2)电话机拨号

将稿件正面朝下放在 ADF 上,左右调整两侧的稿件导板。显示面板显示 DOCUMENT SET。拿起听筒拨出接收方传真机的号码,如果线路有空,且对方机器又有应答时,即可听到“啵-”之长蜂音。然后按下传真机的启动键,显示面板显示:“ON LINE * XMT * 挂上听筒”。这时,UF-200 将把稿件发送出去。如对方站有识别编码,它将在显示板上出现。发送结束后,显示面板上将显示出发送的稿件页数。

COMPLETED

TOTAL PAGES=03

(二)基本接收方式

1. 自动接收

若用户选择了自动接收方式,那么,电话铃鸣响也无需应答,机器会自动进行接收。(下转 64 页)

MODEM 卡的使用——PC—PC 远程通信(二)

自从本栏目开设以来,申老师收到了不少电脑通信爱好者的来信。对于读者提出的有代表性的问题,申老师将选择一些在以后的栏目中给予解答。

河北邯郸市小郑:申老师,我对电脑通信有极大的兴趣,可是我用 MODEM 卡与我爸爸单位的电脑通信,使用了一个通信软件,可是怎么也连不通,这是怎么回事?

申老师:这两次正在讲这个问题。你可以对比讲的内容,自己查一查。上一次已经介绍了要选购合用的 MODEM 卡;要对自己机器的通信口情况调查清楚。这一次介绍 MODEM 的硬件设置,以及用最简单的 BASIC 语句测试它的好坏的方法。同时也就学习了 MODEM 卡的使用方法。

上次说过,MODEM 卡使用的通信口,要避开机器上已经存在的通信口。不管机器上的通信口是用了还是没用(比如鼠标器接上了或没接上),只要电路已经存在,就要回避。这些通信口,可能是一个独立的“串行通信卡”,也可能在“二串一并”I/O 卡上,还可能在“多功能卡”上。为了我们的远程通信实验能够做通,不妨先将原有的通信口撤掉,比如把“二串一并”I/O 卡拔掉。但如果是在多功能卡上,该卡还带有软 硬盘接口或显示接口,那就比较麻烦了,因为不能拔掉。如果该卡上只含一个串行口,今天介绍的 BASIC 程序仍可使用;如果有二个串行口,就只得借助于下一次介绍的通信软件的使用了。

小李:我用的机器已经查过了,只有一个串行口。虽然 I/O 卡上标有 ASYN1 和 ASYN2,但对应 ASYN2 的少两块 IC 片,实际不起作用。

申老师:就按你这种情况来看如何设置。由于 COM1 已被占用,所以 MODEM 卡应选 COM2。中断请求号与通信口有固定的对应关系如下:

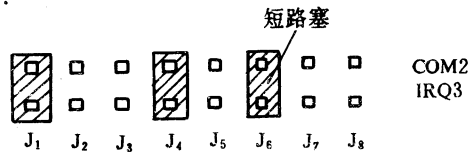
- COM1 对应 IRQ4
- COM2 对应 IRQ3
- COM3 对应 IRQ4 或 IRQ5
- COM4 对应 IRQ3 或 IRQ2

所以,COM2 中断号应选 IRQ3。不同类型的 MODEM 卡设置方法不大一样,有的用开关(DIP),有的则使用跳线(jumper)。以型号为 DM—2400H 的 MODEM 卡为例,它全使用跳线,说明书上查到的跳线表如下:

通信口	J1	J2	J3	J4	中断号	J5	J6	J7	J8
COM1:	*	*			IRQ4			*	
COM2:	*			*	IRQ3		*		
COM3:		*	*		IRQ4 IRQ5			*	*
COM4:			*	*	IRQ3 IRQ2	*			

注:J1~J8 为跳线引脚号 * 表示短路脚

对于 COM2 及 IRQ3,短路脚应为 J1、J4、J6,如下所示:



MODEM 卡硬件设置完成后,就可以插入到 PC 机插座中。将从墙上引来的电话线,通过 RJ11 小插头,联至 MODEM 卡“LINE”插座上。这样,联接就已经完成。如果你还想用电话机监听铃声,可用另一个 RJ11 插头将电话机联至 MODEM 卡的“phone”插座上。

小李:我工作的地方,电话机离电脑比较远,隔一个房间,电话机的位置不能改变,因为经理经常用。能不能从经理室电话接线盒上并联二根线出来,拉到电脑这边 MODEM 卡上?

申老师:可以的。但一定要征得你们经理同意,而且接线时千万不能短路,否则电信局可能来找你的麻烦。

一切准备就绪,就可以打开 PC 机的电源。DOS 启动后,进入 BASIC 或 GWBASIC 命令状态。在“OK”提示符下,按顺序键入如下三个无标号“语句”:

OPEN “COM2: 1200,N,8,1” AS #1 初始化通信口,它表示打开通信口 COM2:,选择传输速率为 1200bps,无校验位,8 位数据位,1 位停止位。如果执行正常,应出现“OK”。如果出现“设备超时”(Device Timeout)信息,或“设备不可用”、“坏的文件名”等英文信息,说明 MODEM 卡的硬件设置还不对,或者 OPEN 语句中的 COM 编号与硬件设置不一致,要查清原因,直至正常执行。否则无法再往下进行。

OPEN 语句执行正常后,执行如下拨号命令:

PRINT #1,“ATDP××××” 其中,×××× 是要拨的电话号码,例如 2123。该语句从通信口输出字符串“ATDP2123”。而“AT”是 MODEM 常用的“AT”命令集的字头,表示引起 MODEM 注意的意思(英文 Attention),“D”是拨号(Dial),“P”是用脉冲方式(pulse)拨号。大家知道,电话有脉冲式拨号的和双音多频拨号(Tone)的两种。老式的都是脉冲式,每拨一个数字都听见“喀喀喀……”响,速度很慢;新式的能进

行音频拨号,拨号时只听见不同音调的几声,速度很快。如果你所用的电话系统支持音频拨号,则可以将脉冲拨号命令 ATDP 改为 ATDT 音频拨号命令。例如 ATDT2123。

发出该拨号命令后,你要仔细听 MODEM 卡上小喇叭发出的声音。如果正常,应该听到如下声音系列(也称呼叫过程):

相当于拿起话机听到的“电流”声→脉冲(或音频)拨号声→电话总机向对方电话拨号的回铃音→如果对方处于自动应答状态,按规定次数响铃后也相当于拿起话机→双方发出的“沙沙……”检测载波声→连接完成,进入信号交换状态。

如果 OPEN 语句执行为 OK,而 ATDP 拨号听不到拨号音和以上呼叫过程的系列声音,多半说明该 MODEM 卡有故障。当然要排除某些 MODEM 卡带有音量旋钮,它被置于音量最小的情况。

当然,在对方没有准备好的情况下,只能听见拨号音,而不能实现连通。你可以键入以下语句使电话“挂机”:

PRINT #1,“ATH” 表示请 MODEM 注意执行 H 命令,即“挂机”(Hang up)。执行此语句,你可听见“咔嚓”一声继电器响,电话被挂断了。

小李:在电话的另一端,即应答方,又应该如何配合才能连通?

申老师:在应答方,同样在 BASIC 或 GWBASIC 提示符“OK”下,先执行一个 OPEN 语句,其中的通信参数要与呼叫方一致,例如 1200,N,8,1,而通信口 COM 的编号则要根据应答方自己机器的串行口配置情况而确定,比如 COM1:,同样要直至正确执行,出现“OK”:

```
OPEN “COM1: 1200,N,8,1” AS #1
```

OK 表示已正确执行 OPEN,紧接着键入如下语句:

```
PRINT #1,“ATS0=1” 令 MODEM 执行 AT 命令,设置寄存器 S0 的值为 1。S0 是响铃次数寄存器,设为 1 表示响铃一次后即转入自动回答状态:“拿起话机”→测试载波→连通。
```

小李:连通以后,是不是进行“交谈”式通信和传送文件都可以?

申老师:都可以,就看你程序怎么写了。下面会给出一个“交谈”方式的程序,实际上与以前介绍过的 PC—PC 直接串行通信差不多,只多了关于拨号的部分。

小李:通信结束后,用什么命令退出?

申老师:数据传送状态的退出,并不是一件容易的事情。因为通信链路一旦建立,COM 口→MODEM→电话线是直通的,一般的字符或信息都被直接传送到对方。连字符串“AT”,MODEM 也不再把它当作命令,而且当成字符传到对方去了。为此,专门设定了一个“退出系列”,即:

停止发送 1 秒以上→发送“+++”,不带回车→

再静止 1 秒钟以上。

这样就能引起 MODEM 注意,使它从数据传送状态退回到命令接收状态,再发 AT 命令(比如挂机 ATH),它就能执行了。

在应答方,由于对方挂机而再也检测不到载波,几秒钟之后,也会自动挂机。

以下给出二个小程序供大家试用。它们经过调试在多种 BASIC 版本下均能运行。祝你们成功,并愿意分享你们第一次与对方连通时的喜悦。

一、AT 命令试验程序

1. 运行 BASIC 或 GWBASIC,至出现“OK”提示。

2. 输入如下便于试验 AT 命令的程序:

```
800 REM AT test
```

```
810 CLOSE
```

```
820 OPEN “COM2: 1200,N,8,1” AS #1
```

打开通信口 COM2,选择速率 1200bps,,无校验,数据位 8 位,停止位 1 位。

```
830 INPUT “AT$=”,AT$ 从键盘输入 AT 命令。
```

```
840 PRINT #1,AT$ 送至 MODEM 执行。
```

```
850 GOTO 830 试下一个,直至用 Ctrl Break 打断。
```

```
860 END
```

注意:对于某些 BASIC 版本,在 COM1 未用的情况下,应将语句中 COM2 改为 COM1 才可运行,MODEM 卡上的设置不必改变。

可试用的 AT \$ 有:呼叫方为 ATDP(或 ATDT)拨号命令、ATH 挂机命令;应答方为 ATS0=1。应答方这命令要先于呼叫方的 ATD 命令之前发出。

二、远程通信小程序

读懂并输入、调试、运行以下远程通信小程序,向你约定的对方拨号并发送或接收信息。(Call—呼叫方,Answer—应答方)

```
1 REM MODEM XFER DEMO
```

```
2 INPUT “Select:1—Call 2—Answer 3—Exit”,N
```

```
3 ON N GOTO 5,500,660;GOTO 2
```

```
5 REM Modem send demo
```

```
10 OPEN “COM2: 1200,N,8,1” AS #1(打开通信口)
```

```
15 INPUT “phone No. :”,T$(输入电话号码)
```

```
20 PRINT #1,“ATDP”;T$(脉冲拨号)
```

```
25 PRINT “Waiting for connected...,then strike a key begin sending”
```

```
30 K$=INKEY $:IF K$=”” THEN 30(等待连通)
```

```
100 PRINT “Sending...('end' for exit)”
```

```
110 INPUT K$
```

```
120 PRINT #1,K$
```

```
125 IF K$=”end” GOTO 170
```

```
130 GOTO 110
```

```
170 SOUND 8000,30;SOUND 2000,1(空闲一秒)
```

```
180 PRINT #1,“+++”;(<向 MODEM 发“+++”,不带回车)
```

```
190 SOUND 8000,30;SOUND 2000,1(空闲一秒)
```

(下转 58 页)

模拟乘法器 IC 及使用方法(一)

李兰友

在测量信号处理系统中,模拟乘法器 IC 可实现两个模拟输入电压相乘的运算。一般模拟乘法器的传递函数可写为:

$$V_{OUT} = V_x \cdot V_y / V_{REF}$$

其中, V_{OUT} 是乘法器 IC 的输出, V_x 、 V_y 是模拟输入信号, V_{REF} 是标度因子, 通常取 10V。

本文介绍美国 AT 公司的模拟乘法器集成电路 AD534, AD535, AD538, AD734, AD834 及其使用方法。

一 AD534

1. 引脚及参数

AD534 是一种具有在片微调的高精度模拟乘法器 IC。其引脚排列如图 1 所示。

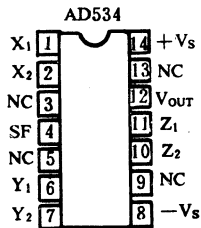


图 1 AD534 引脚

引脚说明如表 1:

表 1 AD534 引脚

引脚序号	名称	功能
1	X ₁	} V _x 输入
2	X ₂	
3	NC	空
4	SF	标度因子
5	NC	空
6	Y ₁	} V _y 输入
7	Y ₂	
8	-V _s	负电源
9	NC	空
10	Z ₁	} V _z 输入
11	Z ₂	
12	V _{OUT}	输出
13	NC	空
14	+V _s	正电源

AD534 主要参数如表 2 所示

表 2

类别 型号	传递函数	总误差 (%)	标度因子 误差 (%)	标度因子 温度系数 (%/°C)	X 输入 非线性 (%)	Y 输入 非线性 (%)	小信号 带宽 (MHz) 0.1V _{rms}	转换 速率 (V/μs)	电源电压/ (V/mA)
AD534J	$\frac{(X_1 - X_2)(Y_1 - Y_2)}{10V + Z_2 - Z_1}$	1(max)	0.25	0.022	0.4	0.01	1	20	(±8~±22)/6 (max)
AD534K		0.5(max)	0.1	0.015	0.2	0.01			
AD534L		0.25(max)	0.1	0.008	0.1	0.005			

动态参数:

$$f_{1\%} = 50\text{kHz (读数相对误差为 1% 的频率)}$$

$$f_{-3\text{dB}} = 1\text{MHz (读数相对误差为 -3dB 的频率)}$$

2 AD534 的工作方式

AD534 的工作原理简述如下。图 2 是 AD534 内部构成简图。X 输入和 Y 输入经 V/I 转换电路送入可变跨导乘算电路, 乘算输出 X · Y 与 Z 输入送至增益为 A 的输出放大器放大后输出。输出 V_{OUT} 为:

$$V_{OUT} = A \{ (X_1 - X_2)(Y_1 - Y_2) / SF - (Z_1 - Z_2) \}$$

由于输出放大器的增益 A 很大(70dB), SF 亦由内部调整为 10V, 这样, AD534 的各输入有以下关系:

$$(X_1 - X_2)(Y_1 - Y_2) = 10(Z_1 - Z_2)$$

AD534 除可做乘法运算外, 通过改变反馈方式, 还可实现除算、平方根运算及平方差运算等。

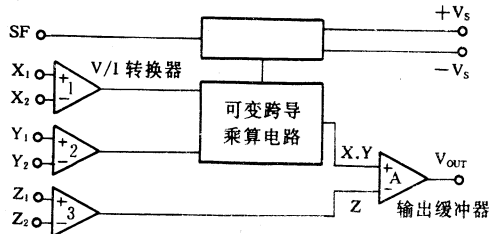


图 2 AD534 内部构成

(1) 乘法运算电路

用 AD534 实现乘法运算的接法如图 3 所示。

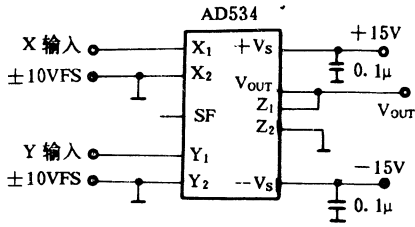


图3 乘法运算

由于 $Z_1 = V_{OUT}$ (Z_1 和 V_{OUT} 接在一起), $Z_2 = 0$ (接地), 因而

$$V_{OUT} = (X_1 - X_2)(Y_1 - Y_2)/10V$$

从而实现 X 输入和 Y 输入的乘法运算。

当 $Z_2 \neq 0$ 而用做其他输入时, 则

$$V_{OUT} = \{(X_1 - X_2)(Y_1 - Y_2)/10V\} + Z_2$$

(2) 除算电路

实现除法运算的电路如图 4 所示。 V_{OUT} 反馈至 Y_2 , 即 $Y_2 = V_{OUT}$, 因而

$$V_{OUT} = 10V(Z_1 - Z_2)/(X_1 - X_2)$$

从而实现 Z 输入和 X 输入的除算。 Y_1 可以做为其他输入, 这时

$$V_{OUT} = \{10V(Z_1 - Z_2)/(X_1 - X_2)\} + Y_1$$

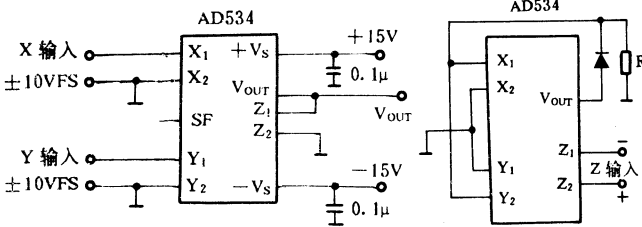


图4 除法运算电路

图5 平方根电路

(3) 平方根运算

用 AD534 做平方根运算的电路接法如图 5

在本电路中, $X_1 = V_{OUT}$, $X_2 = 0$, $Y_1 = 0$, $Y_2 = V_{OUT}$ 因而可求得

$$V_{OUT} = \sqrt{(Z_1 - Z_2)10V}$$

实现对 Z 输入的开平方运算。

根号内的值必需为正值, 因此应注意输入的符号。即若为负值时, 交换 Z_1 和 Z_2 。

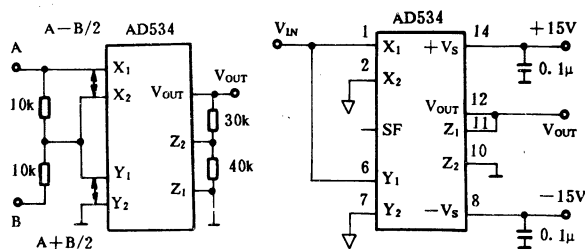


图6 平方差运算 图7 平方电路

(4) 平方差运算

平方差运算电路如图 6 所示。根据接法, 设 $X_1 = A$, $X_2 = (A+B)/2$, $Y_1 = (A+B)/2$, $Y_2 = 0$, $Z_1 = V_{OUT}/4$, $Z_2 = 0$, 则

$$V_{OUT} = (A^2 - B^2)/10V$$

实现平方差运算。

(5) 平方运算

平方运算电路如图 7 所示。输出 V_{OUT} 为

$$V_{OUT} = V_{in}^2/10V$$

3 AD534 实测特性

以图 3 所示的乘法电路为例。 V_x 输入满量程为 $\pm 10V$, 输出 V_{OUT} 为

$$V_{OUT} = (X_1 - X_2)(Y_1 - Y_2)/10V$$

(1) 实测频率特性 (小信号频率特性)

当输入 $V_y = 10V$, $V_x = 1V_{rms}$ 时, 实测频率特性如图 8 所示。此时, $V_{OUT} = 1V_{rms}$ 。从图中可看到, $-3dB$ 频率 $f_{-3dB} = 1.3MHz$ 。

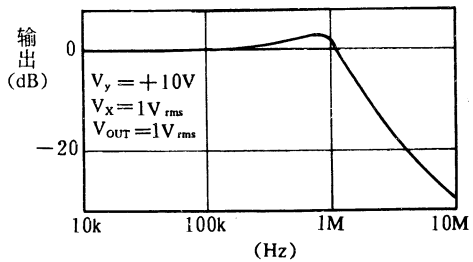


图8 AD534 实测频率特性

(2) 非线性

AD534 的 X 输入非线性约为 0.4%, Y 输入非线性约为 0.1% (见表 2)。实测亦表明, Y 输入的非线性优于 X 输入。因此, 对于大信号, 应尽量使用 Y 输入。小信号用 X 输入, 以使非线性减到最小。(待续)

(上接 56 页)

```

200 PRINT #1, "ATH" (挂机)
210 CLOSE
220 GOTO 2
500 REM Modem receive demo
510 OPEN "COM2: 1200, N, 8, 1" AS #1
520 PRINT #1, "ATS0=4" (响铃 4 次后接通)
600 INPUT #1, A$: IF A$ = "end" GOTO 630
610 PRINT A$;
620 GOTO 600
630 PRINT #1, "ATH"
640 CLOSE
650 GOTO 2
660 END

```

如果你所用的 BASIC 是支持“全双工”的版本; 可增加如下语句, 进行“交谈”式通信。

```

126 INPUT #1, A$
127 PRINT A$;
615 INPUT K$;
616 PRINT #1, K$ (完)

```

DOS 应用技巧集锦

李 际

本人在查阅一些外文计算机资料时发现了一些国内鲜为介绍的 DOS 使用技巧,现介绍给读者。

一 DOS 通配符的简化使用。

DOS 通配符“*.*”常用于指某一目录下的所有文件。其实,在需要键入“*.*”之处,往往键入“.”可得到同样的效果。比如,

```
C:\>COPY A: *.* 可简化为 C:\>COPY A:.  
A:\>DEL *.* 可简化为 A:\>DEL .
```

读者不妨一试。

二 在 DOS 下让扬声器发声。

通过如下方法,用户可在命令行中或批文件中让扬声器发出“嘟”声,以达到提示或表示任务完成等目的。在 DOS 提示符下键入:

```
A:\>COPY CON BEEP  
<Alt>007  
F6<Enter>
```

<Alt>007 的输入方法是:一只手按住 Alt 键不放,另一只手在数字小键盘上键入 007,然后放开 Alt 键。<Alt>007 输入完毕后按下功能键 F6 并回车,这样就在当前目录下建立了一个名为 BEEP 的文件。要想使扬声器发声,可在命令行上键入 TYPEBEEP 或在批文件内需发声处加上 TYPE BEEP 的语句。若想延长发声时间,可在 BEEP 文件中多加几组“<Alt>007”。

三 在批文件中获取按键值。

批文件的应用是非常广泛的,如可用来制作一个菜单,以便使用户,特别是不熟悉 DOS 命令的计算机使用者更方便地使用各种常用程序。

若想用批文件建立一个使用方便的菜单,可利用批文件的 IF ERRORLEVEL 子命令。ERRORLEVEL 是 DOS 的一个变量名,常用于判别系统返回的错误代码值。由于 DOS 没有提供用 ERRORLEVEL 来判断用户按键值的方法,可先用 DEBUG 建立一个能读取键值的小程序,然后用 ERRORLEVEL 语句根据用户的按键值调用所需的程序。以下是建立该程序的方法:

```
C:\>debug getkey.com  
File not found  
-a100  
xxxx:0100 mov ah,00  
xxxx:0102 int 16  
xxxx:0104 mov ah,4c  
xxxx:0106 int 21
```

```
xxxx:0108  
-rcx  
CX 0000  
:8  
-w  
Writing 0008 bytes  
-q
```

有了这个名为 GETKEY.COM 的程序,即可在批文件中用 ECHO 子命令建立一个菜单,然后执行 GETKEY 读取键值。与按键数值 1-9 相对应的返回代码值为 49-57,可用 IF ERRORLEVEL 子程序判别。下面列出一个有四项选择的批文件菜单作为范例。

```
A:>TYPE MENU.BAT  
@ECHO OFF  
:START  
ECHO START MENU  
ECHO 1 PCSHELL  
ECHO 2 WORDPERFECT 5.1  
ECHO 3 NORTON DISK DOCTOR  
ECHO 4 EXIT  
ECHO Your choice? (1-4)  
:CONTINUE  
GETKEY  
IF ERRORLEVEL 53 GOTO CONTINUE  
IF ERRORLEVEL 52 GOTO EXIT  
IF ERRORLEVEL 51 GOTO THIRD  
IF ERRORLEVEL 50 GOTO SECOND  
IF ERRORLEVEL 49 GOTO FIRST  
GOTO CONTINUE  
:FIRST  
C:\PC6\PCSHELL  
GOTO START  
:SECOND  
C:\WP51\WP  
GOTO START  
:THIRD  
C:\NU\NDD  
GOTO START  
:EXIT  
CLS
```

用户只需在显示菜单处填上适当的选项,并在相应的标号(如:FIRST)下面填入所要执行的程序名及盘符路径,一个批文件菜单就完成了。以菜单选项为一例。若用户想执行 PCSHELL 程序可按“1”键,此时返

回代码为 49,计算机执行 .FIRST 标号下的 C:\PC6\PCSHLL。当用户退出 PCSHELL 后,由于该标号下的最后一句为 GOTO START,所以计算机又回到: START 标号下显示菜单。

四 利用批文件在屏幕上显示方框。

以下介绍如何将特殊的 ASCII 码从键盘直接输入到批文件中,从而在屏幕上显示框线。其方法是按住 ALT 键不放,然后用数字小键盘输入这些代码的十进制等效值(不可用键盘上方的数字键)。下面列出与方框线有关的 ASCII 等效值与十进制值的对应表。

十进制值	ASCII 等效值
186	竖线
187	右上角
188	右下角
200	左下角
201	左上角
205	横线

比如,要画出一个长方型的小框,可键入:

```
COPY CON BOX
<Alt>201<Alt>205<Alt>205<Alt>187
<Alt>186<Alt>032<Alt>032<Alt>186
<Alt>186<Alt>032<Alt>032<Alt>186
<Alt>200<Alt>205<Alt>205<Alt>188
F6<Enter>
```

读者可根据需要设置框的大小,或在方框中写上一些文字,如前面介绍的批文件菜单,可将 ECHO 语句中的菜单选项放在方框中,使之成为一个更美观的菜单。

五 利用批文件向打印机发送以 Esc 开头的打印控制代码。

多数打印机的控制码是以换义码 Esc 开头的。Esc 的 ASCII 码值为 27,由于 DOS 已将(Esc)作为控制键而无法从键盘直接输入(当 DOS 从键盘接收到 Esc 信号后立刻停止接收数据并清零,以准备接收新的命令)。然而,我们可以从键盘输入一个经过伪装的 Esc 代码,方法是给 27 加上 128,使之成为 ASCII 155。其实 ASCII 155 与 ASCII 27 的差别仅在其代码高位第八位,前者为 1,后者为 0,这样一来,DOS 不认为它是 Esc 码,而打印机则只考虑该码的前七位而忽略其第八位。所以对于打印机来说,ASCII 27 与 ASCII 155 是等效的。假设要向打印机发送 Esc G 的控制码,可建立这样一个批文件:

```
COPY CON SEND. BAT
ECHO <Alt>155G>LPT1 (打开指定的打印模式)
COPY %1 LPT1 (%1 指需打印的文件名)
ECHO <Alt>155H>LPT1 (关闭该打印模式)
<F6><Enter> (功能键 F6 及回车键)
```

若需打印一个名为 TEXT 的文件,只需在命令行上键入 SEND TEXT 即可。读者可参照上例将有关代码组合起来自行建立批文件。有关打印控制码则可从打印机手册中获得。

六 在 DOS 提示符下利用 PROMPT 命令向打印

机直接发送以 Esc 开头的打印控制码。

DOS 的 PROMPT 命令和拷屏命令(Ctrl-Prts)可结合使用,以直接向打印机发送控制码。其实现方法是,用 PROMPT 命令将系统提示的字符串定义为替代串 Esc,然后利用拷屏功能将 Esc 及后面的其它字符送至打印机,然后关闭拷屏功能并恢复原提示符。下面仍以输送 EscG 打印代码为例给出按键顺序:

```
prompt $e <Ctrl><Prts><Enter>
G<Ctrl><Prts><Backspace><Enter>
prompt <Enter>
```

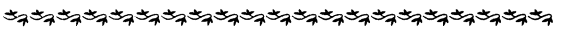
第二行中的 <Backspace>是指按退格键。G 是打印控制码的后半部分,按退格键是为了将 G 删去。一般说来,说该行前面键入过几个字符(打印控制码)就在后面按几次回车键,将其全部删除,免得在按回车键后 DOS 以为有命令输入而试图执行(在上面的例子中需按一次回车键以删除 G)。读者可根据上例举一反三、灵活运用。

七 将计算机当打字机用。

若想打印少量字符(如打印一个信封),那就不一定启动字处理程序,因为可将计算机当打字机用。具体方法如下:

```
COPY CON PRN<Enter>
<TEXT>
<F6><Enter>
```

<TEXT>就是需要打印的内容,输入完毕后按 F6 功能键和回车键就可将其输送到打印机进行打印,操作十分简便。



(上接 58 页)

2. 手动接收

用户想要亲自接收重要文件,这时只需将接收方式由自动(RCV=AUTO)调至手动(RCV=MANUAL)即可。

操作方法如下:

(1)按模式键+或-一直到显示面板上出现 RCV=AUTO 为止。

(2)按下选择键,显示面板显示日、月、年、时、分

RCV=MANUAL

如果返回自动接收状态,再次按动选择键。

(3)机器处于手动接收状态,电话铃鸣响时,拿起话筒,表明对方有传真件发给您,或听到对方准备发送传真的应答,这时,收方机器 ADF 上没有放稿件,按下启动键显示面板显示:“ON LINE * RCV * 挂上电话。”