



● 一九九二年 ● 总期第82期



電子

ISSN 1000-1077

479

與 電腦

猴年进步!



• ELECTRONICS AND COMPUTERS •



Star

智力明星

四达儿童教育电脑

深圳蛇口四达电子有限公司最新推出的人机对话式儿童教育电脑——智力明星集科学性、知识性、趣味性、实用性于一体,该机通过了国家教委中小学教材审定委员会评鉴和广东省教育厅教材编审室审订、各项指标经深圳市标准技术监督局检验合格,并获国家专利(专利号:91 2 05120.5)。在北京、广州、深圳几十所小学推广,受到老师、家长、学生的一致欢迎。

该机的主要功能有:

一、10种基础训练功能:

能对学生进行 $+$ 、 $-$ 、 \times 、 \div 混合运算,填运算符号、音乐作曲、记忆力、判断能力的训练,并具有随机出题、自动记分、自动纠错、自动保护的特点。

二、电子石英钟功能:

可作为电子石英钟使用,并可置闹。

三、人机对话功能:

按照国家教委颁布的教学大纲,该机配有一系列的小学数学、语文、英语训练指导书,并以人机对话形式帮助学生进行系统的课外学习。



欢迎您使用智力明星。

愿您的孩子成为真正的智力明星



广东深圳蛇口四达电子有限公司

地址:深圳市蛇口振兴电子大厦

信箱:深圳蛇口 114 信箱

传真:(0755)691632

电话:691056 691632

电挂:深圳 1612

邮编:518067

新年献词

我们怀着胜利的喜悦送别了1991年，迎来了1992年。值此辞旧迎新之际，请让我代表本社全体同志向《电子与电脑》杂志的广大读者和作者，向多年来一直关心、支持和帮助本刊的各级领导、国内外各界朋友致以节日的祝贺！

电子信息技术是当代新技术革命的重要标志，它已渗透到国民经济、国防及社会生活的各个领域。计算机科学技术正以其强大的生命力改变着人类社会的面貌，新概念、新技术、并以惊人的速度改变着其自身的面貌，新概念、新技术、新系统不断涌现。

90年代是实现祖国社会主义现代化建设总体目标的关键时期。计算机的应用普及和推广对促进祖国四化建设必将起着十分重要的作用。我们正是怀着为在全国普及计算机科学技术，培养初、中级计算机应用人才，为祖国社会主义现代化建设服务的热忱几年前创办了《电子与电脑》杂志。其读者对象是广大计算机和电子技术爱好者，其中绝大部分是青年人。青年是跨世纪的一代，是国家的未来。国家十分重视青年的成长，希望他们承担起继往开来振兴中华的伟大使命。

在过去的一年里，本刊编辑部在广泛征求广大读者和作者意见的基础上总结了几年来办刊的经验和不足，进一步明确了读者对象，突出了本刊的特色和风格，相应调整了栏目，经过一年的实践取得了很好的效果。几年来，我们深刻地体会到，本刊的成长在很大程度上得益于一大批技术骨干的关心、指导和帮助。他们有的是科技界的知名学者，有的是长期在第一线上从事科研、数学、新产品开发的专家和教授，有的是各个领域的革新能手，他们为办好本刊献计、献策，积极供稿，为本刊的健康成长作出了重要贡献，在此谨致以亲切的谢意！

一个刊物的生命力在于她必须具有鲜明的特色和很强的实用性，真正能为广大读者提供他们迫切需要的知识、技术和信息，切实为他们提供良好的服务，我们深感在这些方面尚有很大的差距，热切希望继续得到广大作者和读者的热心指导和帮助。我们将努力把本刊办成广大读者的良师益友，成为联系广大读者和作者的纽带和桥梁，为祖国四化建设作出应有的贡献。

敬致最美好的祝愿！

电子工业出版社

总编辑 张殿阁 谨致

《电子与电脑》 编辑委员会

顾问：孟昭英 张效祥

吴鸿适 周明德

主任：梁祥丰

副主任：宋玉升

委员：（按姓氏笔划排序）

王有春 宋东生

沈成衡 杨仲濂

陈树楷 张殿阁

张道远 顾育麒

柳维长 黄国健

谭浩强 苏 华



“8088、80286系统板故障诊断卡”使维修计算机变得容易了

8088系统板维修测试卡是根据PC—XT及其兼容机的特点设计的,80286系统板维修测试卡是根据PC—AT及其各种286兼容机的硬件特点设计的,它们具有如下共同特点:

1. 在微机的系统板上插入上述测试卡和固化的测试程序,就会对自身系统板的故障进行检测。
2. 具有单步总线测试及显示功能。在单步测试状态下,测试卡直接显示当前CPU操作的地址与数据,并保持CPU操作状态,直到手动按下单步操作键至下一步为止。实现了将快速多变的总线信号转变成静止信号并具有保护故障现场的能力。
3. 对系统板上各模块进行测试。该测试是从实际维修的角度编写的。只要CPU能正确地读取固化的测试程序,即可实现模块化测试(若不能运行测试程序,可用单步总线来检测故障)。程序模块化测试实现了对系统板的所有模块电路和器件的故障测试,并且将全部测试结果显示于屏幕。

该卡由本刊监制及经销。

测试卡价格随行就市,按市售最低价格出售。现时售价:

8088测试卡 1600元/每块
80286 测试卡 1950元/每块

卡售出后,免费保修一年。

银行汇款:

北京工商银行微路分理处

891238—72 电子工业出版社《电子与电脑》

编辑部

邮局汇款:

北京173信箱《电子与电脑》编辑部

邮编: 100036

微机维修技术培训班

本刊邀请中国人民大学信息中心有多年维修经验的专业技术人员任教,在北京不定期举办为期四周的微机原理及维修技术培训班,详细讲授并演示如何利用“诊断卡”准确、迅速地排除系统板故障。同时讲授微机实用维修技术、包括软盘、硬盘、打印机、显示器、电源、键盘。上述内容都以讲课及演示方式授课,使学员在理论及实践的结合中掌握维修技术。

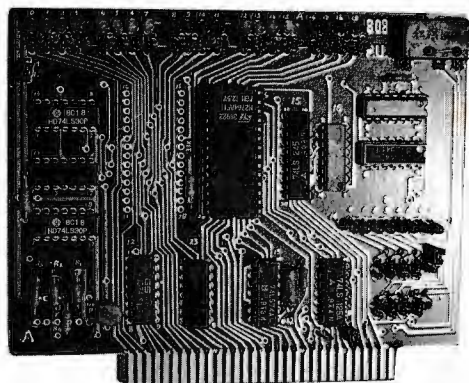
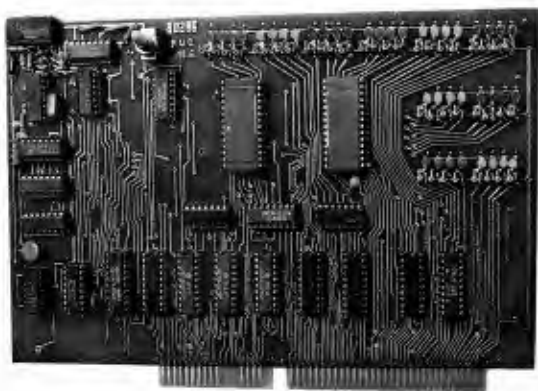
为了保证上课质量,培训班采用小班授课方式,每班不超过30人。欲参加者请复制以下所附报名表寄回。我们收到报名表后,根据报名次序向学员发出上课通知书并通知具体报到、上课时间、地点及乘车路线。

学费380元,资料费实收约40元。

本班负责安排食宿(费用自理)并代购回程车票
联系人:北京中国人民大学信息中心

胡野红 梁杰熙

邮政编码: 100872



报名表		
姓 名	性 别	详细通信地址
单 位 名 称		邮 政 编 码



電子

ISSN 1000-1077

電腦



猴年进步!

一九九二年

总期第82期

電子與電腦

目 录

• 综述 •

- 猴年进步——本刊 92 年致读者 本刊编辑部(2)
我国单片机应用技术发展趋势及展望 何立民(2)

• PC 用户 •

- 自动调平四舍五入误差的简易快速软件 ... 李燕姝(4)
新颖的用户界面——下拉式菜单的实现 李晓峰(6)
再谈硬盘无效时回收文件的方法 岳千钧(9)
IBM 计算机几个重要硬件设备的软锁程序
..... 温盛文(10)
一个实用月历程序 沈玉江(11)
用拉幕方式显示信息 刘善平(11)
拼图游戏 刘晓峰(13)
通用文件病毒防御系统 姜金友(14)

• 学习机之友 •

- 苹果 BASIC 语言容错性浅析 那履弘(16)
Apple 内存校对发声程序 连 勤(19)
ProDOS 磁盘操作系统入门(续) 廖 凯(21)

• 语言讲座 •

- 6502 机器语言程序设计讲座 朱国江(23)

• 初级程序员级水平考试辅导讲座 •

- 计算机硬件基础知识 顾育麒(27)

• 学用单片机 •

- 8031 单片机最小系统 罗明宽 车金相(31)

• 学装微电脑 •

- 附带应急开关的顺序控制 易齐干(34)

• 电脑巧开发 •

- 多功能程序移植器——一种简易的硬件工具
..... 朱立钢(37)

• 电脑游戏机 •

- 第二章 F BASIC 的基本语句 于 春(39)

• 维修经验谈 •

- 微机安装和使用中应注意的几个问题 胡野红(43)
CEC-I 中华学习机修理一例 卢光怀(44)
软盘驱动器综合故障维修一例 周凯歌(44)

• 新书与软件 •

- 电子工业出版社软件部新出版软件介绍 (45)

• 读者联谊 •

- 普及型 PC 个人用户软件交流联谊活动问题解答(一)
..... 王路敬(46)

封面 猴年进步

封二 新年献词

封三 8088、80286 系统板故障诊断

封四 智力明星——四达儿童教育电脑

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 副主编：王昌铭

责任编辑：张 丽

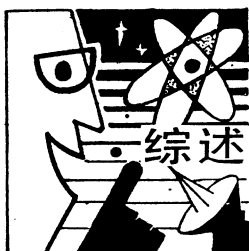
订购处：全国各地邮电局

国外总发行：中国国际图书贸易总公司

(北京 399 信箱 邮政编码 100044)

广告经营许可证：京海工商广字 147 号

定价：0.95 元



猴 年 进 步

——本刊 92 年致读者

羊年过去,猴年到来。在新的一年里开始之际,我们《电子与电脑》全体工作人员向本刊国内外的新老读者、译者、编委们、关心支持我们工作的领导和各界朋友们恭贺新春,祝大家猴年愉快,事业进步!

在过去一年里,本刊收到了许多热心朋友们的办刊建议、指导和希望。对于读者们参予办刊的热情,深受鼓舞,增加了我们办好刊物的信心和力量。我们一定努力克服主客观条件的困难,将刊物办的一年更比一年强,为我国电脑应用技术的普及,作出应有的贡献。

科技期刊作用的大小,取决于办刊的质量。在新的一年里,我们将从报道内容质量,编辑出版质量;为读者服务的质量等几大方面进一步改进工作。

一、提高办刊质量,增强刊物特色

1. 报道内容质量

本刊的读者层次多为初、中级水平的电脑应用人员。在此前提下,我们在刊物报道内容方面,不仅应注意政策性和科学性,更要突出普及性、启发性和实用性。

本刊读者群中,多数为自学电脑的爱好者,其学习的侧重点也不尽相同。为此,本刊在栏目设置上覆盖了电脑软件和硬件二方面。内容有一定的系统性,软硬件资料的透明度高,使初学者便于借鉴和移植,再辅之以必要的知识竞赛、联谊交流、函授教学和水平考试辅导、以及出版专题性的专辑,使读者通过刊物和社会活动结合起来,在实践中学习电脑应用技术。让我们共同努力把刊物办成普及电脑应用技术的开放式社会大学校。

2. 编辑出版质量

本刊主要栏目今年仍相对稳定,不拟作较大调整。编辑人员要积极组稿并认真审读来稿。不断提高文章加工水平,使文章层次结构清楚,逻辑性强,语言精炼顺畅,技术内容准确,减少文字和内容的差错率。

封面设计要富有特色,寓意和内涵确切,富有知识性和感染力。正文版面设计合理,图文协调,提高校对质量、印刷质量及出版准确率。

二、努力加强读者服务工作

科技期刊编辑部,一向把为读者提供信息、技术咨询、代购代邮等服务作为办刊的有机组成部分。去年,我们同中电总公司的北京振兴电子公司在京试办了《电子与电脑》读者服务部,开展了一些为读者邮购期刊和电脑零配件的工作,受到了读者的信赖和支持,由于人力和组织工作经验的不足,在邮购范围上,特别是技术咨询方面仍有较大差距,今后,我们将努力改进。

今年,为了搞好本刊的信息和技术咨询工作和为没有实践条件的读者开办一个具有技术实验条件并能将实验性设计转变为“产品”的《电子与电脑》技术开发服务部,正同有关单位协商具体事宜并办理申报手续。

希望有更多的读者通过刊物和我们的读者服务工作,成长为具有一定应用水平的电脑人才,为我国电脑应用技术的普及和提高贡献力量。

《电子与电脑》

全体工作人员

1992. 元旦

我国单片机应用技术发展趋势及展望

北京航空航天大学 706 教研室(100083) 何立民

在我国,单片机的开发应用已有六年之余,形成了一支庞大的技术开发队伍,为我国单片机应用积累了丰富的经验。随着电子技术、计算机芯片技术、微电子技术的飞速发展,引起了单片机应用技术一日千里的变化。面对这种形势,不断地及时地了解、调查国内外现状、总结经验、分析技术发展趋势来指导我国单片机应用技术的健康发展具有重要意义。

由于我国微电子技术、计算机技术、半导体器件工业现状以及国民经济发展的模式,决定了我国在单片机应用系统的开发、研制方面走着与国外不尽相同的道路。为了指导我国单片机应用技术的发展,除了不断分析国内外市场形势,技术发展状况及其对我国开发环境的影响外,还应分析我国的技术状态、国情,提出中肯的意见,现提出一些不成熟的看法。

一、我国单片机主流芯片的发展趋势

我国单片机应用从起步到现在走的是一条自发地追随进口市场,沿国外 Intel 公司主干芯片更新变化的路径,从 MCS-48、MCS-51 到 MCS-96 及其新系列 8098、80C196。从 MCS-48 的淘汰和 8098、80C196 的兴起引起了对单片机主流机型的思考:我国单片机主流机型能否稳定下来,究竟能稳定在什么机型上。

单片机最确切反映其含义的名称是工业测控用微控制器,其应用对象决定了它的应用领域与应用形态。目前以及今后一个较长的时间内,8 位、4 位 CPU 已能满足 80%测控对象的要求。因此,国外在单片机发展策略上十分重视 4 位、8 位机的研究,认识到单片机的位数只是单片机诸性能中的一个指标。对于工业测控对象来说,还有更重要的功能要求,如为提高系统可靠性的软件监视、电源监测、数据防改写及掉电保护;提高控制速度的高速 I/O 口;改善控制功能的可编程计数器阵列、多机通信的硬件识别以及 A/D 转换、PSW 电路、各种接口电路等。在不提高 CPU 数据总线宽度,甚至减小宽度至 4 位水平,把有限的集成度用来增加 I/O 口的数量与功能,甚至把一些模拟电路、功率电路集成到单片机中以提高其综合性能。

以上这些技术措施的基础是微电子技术的高度发展。我国现阶段盲目跟踪国外市售芯片的现象充分反映了我国微电子技术的落后状态。换言之,我国单片机技术的发展趋势在相当大的程度上决定于我国微电子技术和单片机国产化的进程。目前 MCS-48、MCS-51 系列单片机的国产化已有眉目。可以预计我国单片机的发展趋势。

1. 8 位机在相当长的时期内仍会是我国单片机的主流,根据用户不同要求会引进一些 8 位机的新产品系列。这些新型 8 位单片机完全具备或在某些方面已超过目前 16 位机的一些特殊功能。

2. 8 位机的国产化会使 8 位机在我国长期稳定,并开创我国 8 位单片机应用的崭新局面,如 8048、8051 的掩膜程序产品,典型系统的 ASIC 化,新型派生系列单片机的研制等。

3. 随着我国微电子技术的发展,ASIC 技术的成熟会导致 4 位机的应用热潮,因为 4 位机的应用只能走 ASIC 的道路。

4. 目前我国 8098、80C196 的供货及廉价开发环境为 16 位、准 16 位机的应用创造了极好的条件,但应用热中有一定的虚假现象。其中部分用户未必是非用 16 位 CPU 不可,而是看中其片内 A/D、高速 I/O 口、PSW 以及 80C196 的 CHMOS 工艺。实际上在 MCS-51 新型芯片中也具备了这些特性,甚至还有更特殊的功能模块,但苦于国内无供货渠道,只好选用国内供货方便,并有廉价开发环境的 8098 及 80C196。

二、单片机应用系统中的新器件

单片机应用系统中主要大规模集成电路依靠国外

市场的局面还会延续一个相当时期,但由于多年的努力,国内已形成较齐全、稳定的芯片供货市场,少量最新型器件也能快速引进,有利于及时跟踪国外先进技术。

1. 单片微型计算机

目前 8 位机仍是厂家重点发展的产品,包括早期的 MCS-48 及后来 MCS-51。其新型派生系列芯片沿两种模式发展:一是电子厂家用芯片,最典型的如飞利浦公司,将 MCS-48、MCS-51 为核心构成的 I²C 总线单片机,大量用于本公司的产品改造中;另一种是芯片厂家推出的新型市售单片机供用户使用。

16 位和准 16 位机也是芯片厂家注意发展的产品,以解决 8 位 CPU 无法解决的高速、复杂运算的应用场合。

国内在选用新型单片机时仍把目标盯在 Intel 系列上。MCS-51 系列新器件大多为 CHMOS 工艺,具有功耗低,抗干扰性能好、速度快、集成度高的特点。有了最早的 CHMOS 系列 80C51BH、80C31BH、87C51 以及继 8052、8352 后的 80C52、83C52。

1988 年以后 Intel 公司公布的新型单片机主要有带两个 DMA 的通用通信控制器 8XC152;带有可编程计数器阵列、可编程串行通道、增强掉电方式的 8XC51FA/FB;带 8 位 8 通道 A/D 的 8XC51GA/GB 以及带有大量 I/O 口的 8XC451。

16 位机、准 16 位机由于国内单片机开发公司的快速跟踪,无论在单片机芯片供货和开发环境都为用户创造了较好的条件,因此 8098、80C196 将会拥有不少新用户。

其它系列芯片,如 μ PD7810/11、Z86 系列单片机也是具有十分优异性能的单片机,限于国内开发环境的限制,受主流芯片的排斥,大面积推广有一定困难,但在一些地区仍有较好的市场。

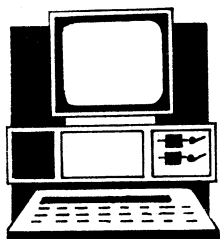
2. 存储器

存储器容量迅速增大,使 2764、27128、27256 成为当今单片机应用系统程序存储器的主流芯片,6264 成为数据存储器的主流芯片。

应用系统要求对 SRAM 的数据保护促进了形形色色掉电保护 SRAM 的发展。目前先进的掉电保护集成芯片将可充电电池、充电电路、保护电路集成在一个芯片之中,如自保护存储器组件 9964,其外形和插脚与通用存储器类似,可直接替代 6116、6264、2716、2732、2764;DS1235YH 则是一种 32KB 集成掉电保护 SRAM 芯片,可直接替代 62256。

由于可靠的掉电保护、防改写电路的日趋成熟,焊接安装型可充电电池的出现,以及掉电保护集成 SRAM 芯片组件的出现,引起了人们对 E²PROM 前途的争议,可以预料,如果不提高 E²PROM 的写入速度就很难与掉电保护 SRAM 集成组件竞争。

(待续)



PC 用户

自动调平四舍五入误差的简易快速软件

天津电气传动设计研究所(300180) 李燕妹

一、问题的提出

在计划统计一类的报表处理中,一般基层表的计量单位:值用元、量用仟克;当汇总后向上级机关报表时,值和量的单位却要转换为万元和吨。在变单位的过程中,由于要做四舍五入计算,其结果势必会产生基数和不等于合计数的误差。因此为使一个二维报表的横向和纵向的各个小小计、小计、合计和总计的合计关系保持平衡,必须对舍入误差进行调整。

二、自动调平舍入误差的设计

根据调整内容的不同,可将调整方式分为三种:横调、纵调和纵横调整。只对若干字段或若干记录进行变单位的报表,只需做横调或纵调即可。但对流向分析一类的报表,由于库文件的所有数据都需做单位换算,所以应进行纵横调整。

下面就调整舍入误差软件的设计思想和实现过程做一些说明,并将其程序清单附后。

首先由用户输入欲调整的 FOXBASE 数据库文件名、调整方式、合计数的字段号、记录号和基数的字段号(横调时指定)或记录号(纵调时指定)、以及调整位数(元变万元为 4、仟克变吨为 3)。然后通过该软件运算几秒钟即可将所有数据调平。

横向调整时,由用户输入参加合计的各个基数的字段号,字段号间用逗号隔开,当字段号连续时,可只输入首尾两个字段号并用破折号隔开来简化操作。采用子串搜索函数 AT()和截子串函数 SUBSTR()将用户输入的字段号依次截取出来,并用字段名函数 FIELD()将字段号转变为字段名,便可生成以逗号为分隔符的字段名表 F,接着用拷贝命令 COPY TO ZJK FIELD &F 就可将所选各基数字段拷贝到中间库 ZJK 中。随后打开中间库,用散布命令 SCATTER TO T 将各基数字段的数据按顺序填入数组 T 的各个元素中。

纵向调整时,用类似的方法可把参加合计的各个基数的记录号截取出来,并依次将需调整的内容存入数组 T 的各个元素中,同时用删除命令 DELETE 给这些记录加上删除标志,由过滤命令 SET FILTER TO DELETE()把打上删除标志的记录筛选出来,通过拷贝命令 COPY TO ZJK FIELDS 行次,&Z1 就可将纵调选

择的基数字段拷贝到中间库 ZJK 中。

一旦需要调整的基数存入数组 T,该软件就要调用子程序 ZTG,PRG,将各项被调数据除以变单位的换算因子,并进行四舍五入运算。用 $CAS = \text{ROUND}(&Z1 / 10^{\wedge} WS, 0)$ 把做了四舍五入的合计数存入变量 CAS 中,用 $U(J) = \text{INT}(T(J) / 10^{\wedge} WS)$ 把各基数的整数部分存入数组 U 中,用 $V(J) = \text{VAL}(\text{SUBS}(\text{STR}(T(J)), 11 - WS, WS))$ 把各基数的小数部分存入数组 V 中,然后以合计数为准调整基数。此时做过舍入的合计数必然大于或等于只取整的各基数之和,所以如为等于,则无需调整,而如为大于,则要按差值的多少,看小数部分的大小,自大到小循环给相应基数的整数部分加 1,直到合计数与基数持平。这样就可将存放调平基数的数组 V 送入数组 T 中,以便下一步将它们反送回库文件的原始位置。

横调时仍用 AT()和 SUBS()函数顺次将字段名表 F 中的基数字段抽出来,通过替换命令 REPL 将数组 T 的各个元素逐一代入原记录中;而纵调时,需先将数组 T 的各个元素代入中间库中,然后通过更新命令 UPDATE 将中间库中的数据按行次更新被调字段的内容。

选择纵横调整时,采用先纵后横或先横后纵的方法,均是两种方式的综合,所以结果是一样的。本软件选用的是先横后纵的方法,即先以最大的合计数为准进行横调,再以该记录的各项分合计数为准进行纵调。最后将各记录的基数相加,反算出相应的横向合计数,即可自动地将各种合计关系调平。

纵横调整还可采用以记录号为内循环的控制条件,以字段号为外循环的控制条件,将所有要调整的数据存入二维数组中,随之在二维数组中折算单位四舍五入,然后视纵横双向的合计关系将数调平,最后仍以字段号和记录号为控制条件,将此二维数更新库文件的内容。由于篇幅的限制,在此就不再赘述了。

本软件不仅简易实用、快速准确,而且通用性好,实用性强,可为各种变单位的报表自动调平舍入误差使用。

程序清单

```
ZDTZ. PRG
set talk off
set safe off
publ z1,ws,gs,tzjl,hj
```

```
dime t(70),u(31),v(31),r(30)
stor 0 to hj,tzjl,c,gs,z2
stor spac(30) to jsh,jlh
stor " " to f
acce "请输入库名" to km
```

```
input "请输入变大单位的调整位
数" to ws
clea
@ 2,24 say "选择调整方式"
@ 3,22 to 7,40 doub
```



```

@ 4,27 prom "1. 横向调整"
@ 5,27 prom "2. 纵向调整"
@ 6,27 prom "3. 纵横调整"
menu to x
sele 1
use &.km
if x=1. or. x=3
@ 10,23 say'请输入横向调整的;
合计字段号'get hj pict "99"
@ 11,0 say'请输入横向调整的基
数字段号,号间用逗号隔开,连续用破折
号连首尾'

```

```

@ 12,0 get jsh
@ 13,0 say'请输入纵向调整的合
计记录号';

```

```

get tzjl pict"99"
read
p=1
sele 1
jsh=trim(jsh)+','
* l=1
do while .t.
b=at(',' ,jsh)
if b<4
b1=subs(jsh,1,b-1)
gs=gs+1
else
c=at('-',jsh)
c1=subs(jsh,1,c-1)
c2=subs(jsh,c+1,b-1-c)
do while val(c1)<=val(c2)
b1=c1
gs=gs+1
if f=" "
f=field(val(b1))
else
f=f+',' +field(val(b1))
endi
f=trim(f)
c1=str(val(c1)+1)
l=l+1
endd
endi
b3=stuff(jsh,1,b,"")
if len(ltrim(b3)) # 0
b2=subs(jsh,b+1)
endi
if c=0
if f=" "
f=field(val(b1))
else
f=f+',' +field(val(b1))
endi
f=trim(f)
l=l+1

```

```

endi
c=0
if len(ltrim(b3))=0
exit
endi
jsh=b2
endd
copy to zjk fields &.f
sele 2
use zjk
scat to t
do ztg
use
f=f+','
sele 1
use &.km
go tzjl
i=1
do while i<=1
b=at(',' ,f)
y1=subs(f,1,b-1)
r(i)=y1
repl &.y1 with t(i)
b3=stuff(f,1,b,"")
if len(ltrim(b3)) # 0
b2=subs(f,b+1)
else
exit
endi
f=b2
i=i+1
endd
n=gs
endi
if x=2. or. x=3
if x=2
@ 12,23 say'请输入纵向调整的字
段号';
get hj pict "99"
@ 13,23 say'请输入纵向调整的合
计记录号' get tzjl pict"99"
read
n=1
endi
@ 14,0 say'请输入纵向调整的基
数记录号,号间用逗号隔开,连续
用破折号连首尾'
@ 15,0 get jlh
read
sele 1
copy stru exte to jg
use
sele 3
use jg
loca for field.name='行次'

```

```

if .not. found()
appe blan
repl field.name with'行次'
repl field.type with'c'
repl field.len with 2
endi
use
sele 4
crea tzkl from jg
appe from &.km
repl all 行次 with str(recn(),2)
use
sele 1
use &.km
zap
appe from tzkl
jlh=trim(jlh)+','
q=1
do while q<=n
sele 1
p=2
gs=0
go top
do while .t.
b=at(',' ,jlh)
if b<4
b1=subs(jlh,1,b-1)
loca for b1 $ 行次
gs=gs+1
if x=2
z1=field(hj)
else
z1=r(q)
endi
t(gs)=&.z1
dele
else
c=at('-',jlh)
c1=subs(jlh,1,c-1)
c2=subs(jlh,c+1,b-1-c)
do while val(c1)<=val(c2)
loca for c1 $ 行次
gs=gs+1
if x=2
z1=field(hj)
else
z1=r(q)
endi
t(gs)=&.z1
dele
c1=ltrim(str(val(c1)+1))
endd
endi
b3=stuff(jlh,1,b,"")
if len(ltrim(b3)) # 0

```

```

b2=subs(jlh,b+1)
else
exit
endi
jlh=b2
endd
sele 1
set filt to dele()
go top
copy to zjk fields 行次,&z1
set filt to
sele 2
use zjk
reca all
do ztg
i=1
do while i<=gs
go i
z3=field(2)
repl &z3 with t(i)
z2=0
i=i+1
endd
sele 1
index on 行次 to tzsy
use &km index tzsy
upda on 行次 from b repl;
&z1 with b->&z3 random
use

```

```

sele 2
use
if x=3
sele 1
use &km
z2=field(hj)
if q=1
repl all &z2 with 0 for recn() #tzjl
endi
repl all &z2 with &z2+&z1;
for recn() #tzjl
endi
q=q+1
endd
endi
close all
dele file zjk.dbf
retu
ZTG. PRG
sele 1
go tzjl
if x=1. or. x=2. or. x=3. and. p=1
z1=field(hj)
cas=round(&z1/10^ws,0)
endi
if x=3. and. p=2
z1=r(q)
cas=&z1
endi

```

```

repl &z1 with cas
sele 2
stor 0 to u,v
j=1
do while j<=gs
u(j)=int(t(j)/10^ws)
v(j)=val(subs(str(t(j)),11-ws,
ws))
cas=cas-u(j)
j=j+1
endd
do while cas>0
stor 0 to sz,wz
j=1
do while j<=gs
wz=iif(sz>=v(j),wz,j)
sz=iif(sz>=v(j),sz,v(j))
j=j+1
endd
u(wz)=u(wz)+1
v(wz)=0
cas=cas-1
endd
j=1
do while j<=gs
t(j)=u(j)
j=j+1
endd
retu

```

新颖的用户界面——下拉式菜单的实现

中国人民银行沈阳市分行(110014) 李晓峰

一、概述

下拉式菜单、弹出式菜单是目前流行的编程技术之一。它操作方便、快捷、形式新颖,扩大了屏幕的有效使用面积,为系统提供了一个良好的用户界面。

所谓下拉式菜单是指:一个运行的系统在屏幕上常驻一简单提示或一菜单,当用户打某一“热键”或在菜单上选择某一选项时,在屏幕某一位置上瞬间弹出相应的下一级菜单,当结束某一级菜单上的操作,退出到上一级菜单时,该菜单立即消逝,原屏幕内容恢复。

弹出式菜单是指深度只有一层的下拉式菜单。目前,很多软件,包括系统软件都采用了这种技术,例如: Turbo C 的集成调试环境 TC 等。下面介绍以 C 语言为工具,实现下拉式菜单的方法。

二、实现方法

描述一个菜单,需要这样一些属性:

- 菜单上的显示项目。
- 菜单左上角坐标(相对于整个屏幕坐标原点)。

- 当前光标所在的选项号。
- 菜单保存区的内存首地址。
- 菜单的前景、背景色彩。

... ..

当程序中菜单较多时,最好构造一个结构数组,以便于菜单的处理。例如:

```

typedef struct struc-menu{
char * * menu;
int x1,y1,x2,y2;
int cup;
char * p;
int count;
int foratt;
int bakatt;
... ..

```

}MMENU;

数组大小由菜单多少而定,每个菜单对应一个数组元素,这样就可由下标来调用一个菜单。

为了支持下拉菜单,相应于一个菜单的弹出一操作一消失,需要构造一系列函数:

•菜单生成函数:

该函数以一个菜单的部分属性为输入数据,计算菜单面积,并申请内存区做为菜单所占区域的保存区,生成全部属性后,将其存入一个结构数组元素中。

该函数在系统初始化时调用,每调用一次生成一个菜单。

•屏幕保存函数:

在一个菜单弹出前,把菜单将要覆盖的那部分屏幕的当前内容保存起来。

我们知道,屏幕的每个显示位置,在显示缓冲区中占两个字节空间:一个是显示代码,一个是显示属性。故对一个 m 行 \times n 列的菜单,实际应申请 $2 \times m \times n$ 个字节的空间。

有三种方法可实现屏幕保存:

1. 利用 BIOS 系统调用:

getch(char *zc, char *zs)

```
{
    -AH=8;
    -BH=0;
    geninterrupt(0x10);
    *zc=-AL;
    *zs=-AH;
}
```

在一个两层循环结构中,扫描矩形区域的每个显示位置,并调用上例函数,实现屏幕保存。

2. 直接读取显示缓冲区中的数据:这种方法比前一种速度快,但由于各种机器所配的显示卡不同,其显示缓冲区的地址也不相同,所以程序的可移植性差。

3. 如果用 Turbo C 编程,可使用 gettext 函数,既快又简便,即使保存整个屏幕,视觉上也没有停留感。

•菜单显示函数:

用菜单背景色清菜单区域,在菜单边界画框,然后,用前景色显示每一选择项。

•菜单操作函数:

在菜单的第一选项上,反转显示一光条,并支持下列按键操作:

① 光标键 \uparrow 、 \downarrow 、 \leftarrow 、 \rightarrow 和“热键”(选项首字母)用于移动反转光条。

② Esc 或其它特殊键用于结束操作。

③ 回车选中一个选项,并返回该选项序号。

•屏幕恢复函数:

在结束菜单操作后,把原屏幕内容恢复。相应于屏幕保存方法,也有三种方法恢复屏幕:

1. 利用 BIOS 系统调用。

2. 直接读写显示缓冲区。

3. 用 Turbo C 的 puttext 函数。

三、菜单的汉化显示

英文版 Turbo C 2.0 的集成环境下,不支持汉字,但是,我们可以在其它汉字编辑环境下,完成汉字部分的编辑。

Turbo C 的 printf, puts 函数可以完成汉字输出,但它们不支持色彩,而控制台的 cprintf, cputs 函数虽然支持色彩,却不支持汉字输出。因此,若想以丰富的色彩显示汉字,需另写一些输出函数,如:

void cdis_ch(char c, int fore, int bak, int n)

```
{
    union REGS r;
    r.h.ah=0x9;
    r.h.al=c;
    r.x.cx=n;
    r.h.bh=0;
    r.h.bl=(fore|(bak<<4))&0x7f; /* note: not
    flash!! */
    int86(0x10, &r, &r);
}
```

以 fore 为前景、bak 为背景,显示 n 个字符 c。

四、对汉字屏幕的保存与恢复

上述关于保存、恢复屏幕的方法以及目前其它一些专业刊物上登载的有关保存与恢复屏幕的技术,在英文系统下实现是没有问题的,但当我们在汉字系统下实现时,必然会遇到一个问题:即当菜单(或窗口)弹在汉字显示区域且边界跨在一个汉字的中间时,屏幕不能被有效恢复!!

原因是:若菜单左边界跨在某行的一个汉字中间时,由于保存的原该行显示数据是以一个汉字的第2个内码为首的一串显示内码及属性,所以在恢复屏幕重写该行时,这个汉字的第2字节内码与后续的内码(可能是汉字内码,也可能是 ASCII 码)重新组合显示,结果是一些杂乱的汉字或图形符。

若菜单右边界跨在某行的汉字中间,则恢复时,该行原内容可恢复,但边界上的汉字不能恢复。原因与上述类同。

下面介绍两个解决这个问题的方法:

1. 对每个菜单都申请一块足以保存整个屏幕的内存区,弹出前,扫描整个屏幕的每个显示位置,保存其显示代码和属性。然后弹出菜单。当要消掉菜单时,再用保存的数据重写整个屏幕。

这种方法实现简单,缺点是如果程序菜单的叠加程度很深时,需要较多的内存空间,并且由于扫描的面积大,若使用 BIOS 系统调用实现,视觉上有停留感。

2. 对一个 m 行、 n 列的菜单,申请 $2 \times m \times n + m \times 4$ 个字节空间。当保存每一行最右列时,若是一个汉字的第2字节,则向左再读一个显示位置;当保存最右列时,若是汉字的第1字节,则向右再读一个显示位置。恢复屏幕时,处理逻辑类同(程序见清单)。

我用新写的 cgettext, cputtext 及其它函数,成功地用英文版 Turbo C 2.0 在 286 机上实现了一个具有全屏编辑功能的、汉化的、配有下拉菜单和弹出窗口的汇总表程序。

其中: $p = (\text{unsigned char } *) \text{malloc}(2 * (x2 - x1 + 1) * (y2 - y1 + 1))$
 $pl = (\text{unsigned char } *) \text{malloc}(2 * (x2 - x1 + 1))$


```

pr=(unsigned char *)malloc(2*(x2-x1+1))
void cgettext(int y1,int x1,int y2,int x2,char *p,char *pl,
char *pr)
{
    int i,j,n;
    char zc,zs;
    char *zpl,*zpr;
    zpl=pl;
    zpr=pr;
    for (i=x1;i<=x2;i++) {
        for (j=y1;j<=y2;j++) {
            gotoxy(j,i);
            (void)GetTCE(&zc,&zs);
            *p++=zc;
            *p++=zs;
        }
    }
    for (i=x1;i<=x2;i++){
        *zpl++='\1';
        *zpl++='\1';
        *zpr++='\1';
        *zpr++='\1';
    }
    if (y1!=1) {
        for(i=x1;i<=x2;i++){
            gotoxy(y1,i);
            n=GetCE(&zc,&zs);
            if (n==2){
                gotoxy(y1-1,i);
                (void)GetCE(&zc,&zs);
                *pl++=zc;
                *pl++=zs;
            }
            else {
                pl=pl+2;
            }
        }
    }
    if (y2!=80) {
        for (i=x1;i<=x2;i++){
            gotoxy(y2,i);
            n=GetCE(&zc,&zs);
            if (n==1) {
                gotoxy(y2+1,i);
                (void)GetCE(&zc,&zs);
                *pr++=zc;
                *pr++=zs;
            }
            else {
                pr=pr+2;
            }
        }
    }
}
/* end 'cgettext' */

```

```

void cputtext(int y1,int x1,int y2,int x2,char *p,char *
pl,char *pr)
{
    int i,j,n;
    char zc,zs;
    for (i=x1;i<=x2;i++) {
        if (*pl!='\1') {
            gotoxy(y1-1,i);
            zc=*pl;
            pl++;
            zs=*pl;
            pl++;
            PutCE(&zc,&zs);
        }
        else {
            pl=pl+2;
        }
        for (j=y1;j<=y2;j++){
            gotoxy(j,i);
            zc=*p;
            p++;
            zs=*p;
            p++;
            PutCE(&zc,&zs);
        }
    }
    if (*pr!='\1'){
        gotoxy(y2+1,i);
        zc=*pr;
        pr++;
        zs=*pr;
        pr++;
        PutCE(&zc,&zs);
    }
    else {
        pr=pr+2;
    }
}
/* end 'cputtext' */
GetCE(char *ch,char *att)
{
    union REGS r;
    r.h.bh=0; /* 0 page */
    r.h.ah=0x48;
    int86(0x10,&r,&r);
    *ch=r.h.al;
    *att=r.h.ah;
    return(r.h.bl); /* 0:ascii,1:first,2:second */
}
/* end 'GetCE' */
void PutCE(char *ch,char *att)
{
    union REGS r;
    r.h.bh=0;
    r.h.ah=9;
}

```

一个实用月历程序

沈玉江

月历程序已为广大的计算机爱好者所熟悉,但笔者迄今为止所看到的月历程序都是不带农历的。为了弥补这个缺陷,笔者试着编写并在PC机上调试成功了此程序。运行此程序(以1991年日历为例)后,能同时显示农历。

一、程序功能

只要将公元的年数、每月天数和农历每月天数输入后,即可获得如下效果:

- 1、每季度首末都打印星期标志。其中周一到周六(M1—M2)为黄色,星期天(SU)为红色;
- 2、周一至周六为公历都用蓝色打印,紧随其后的农历(除每月第一天外)都用青色打印;
- 3、星期日的公历和农历(除每月第一天外)都用亮洋红打印;
- 4、农历每月的第一天都用洋红打印,并且用“Y+该农历月份数”表示,以便识别;
- 5、在每月的左侧打印出该公历月名称(用数字表示);

二、程序说明

(一)、变量说明:

YN——公元年(本例为1991);TN 星期(初始值表示元旦是星期几);NL——农历(初始值表示元旦的农历);D 数组——公历每月天数;NAM\$ 数组——星期标志;X—农历每月天数;NX——农历月份;S——每月打印行数指针。

(二)、部分语句说明:

- 1、160和170语句是为了处理公历和农历月份不一致而设置的,每年的情况均不同;
- 2、100和250语句是为了让公历月名称打印在该月内容的第三行的位置上;
- 3、320——330语句是为显示控制语句,可改成其它形式或者不要;
- 4、390——440语句为打印星期标志子程序;
- 5、370——380语句分别存放公历和农历每月天数,本程序380语句中的前两个数据(30、30)为农历90年11月和12月的天数;
- 6、改动部分语句后可将星期日显示在每周的最前面;将程序中的色彩显示语句去掉后,可在黑白显示器上显示;稍作改动,即可在彩色打印机上打印出彩色日历。

```
10 SCREEN 0:WIDTH 80:COLOR 1,6:CLS
20 YN=1991
30 TN=2:NL=16:DIM D(12)
```

```
40 FOR K=1 TO 7:READ NAM$(K):NEXT K
50 FOR K=1 TO 12:READ D(K):NEXT K
60 PRINT TAB(32);“ * * * ”;:COLOR 4:PRINT YN;:COL
  OR 1:PRINT“ * * * ”:PRINT
70 READ X:NX=1
80 FOR I=1 TO 12
90 IF ((I-1)/3)=INT((I-1)/3) THEN GOSUB 390
100 S=1
110 FOR K=1 TO D(I)
120 IF TN=7 THEN COLOR 13
130 PRINT TAB(TN*10);USING“\ ”;STR$(K);
140 IF NL(>1) THEN COLOR 3:GOTO 190
150 COLOR 5
160 IF NX=2 THEN PRINT USING “-(Y##- )”;12;:
  GOTO 180
170 PRINT USING “-(Y##- )”;NX-2
180 COLOR 1:GOTO 210
190 IF TN=7 THEN COLOR 13
200 PRINT USING “-(##- )”;:COLOR 1
210 IF NL(>X) THEN NL=NL+1:GOTO 230
220 READ X:NL=1:NX=NX+1
230 IF TN(>7) THEN TN=TN+1:GOTO 270
240 TN=1:S=S+1:PRINT
250 IF S(>3) THEN 270
260 COLOR 4:PRINT“ * ”;I;“ * ”;:COLOR 1
270 NEXT K
280 IF TN=1 THEN 300
290 PRINT
300 IF I/3(>)INT(I/3) THEN PRINT :GOTO 340
310 GOSUB 390:PRINT :PRINT
320 A$=INKEY$:IF A$="" THEN 320
330 IF A$(<)CHR$(13) THEN BEEP:GOTO 320
340 NEXT I
350 END
360 DATA M1,T2,W3,T4,F5,S6,SU
370 DATA 31,28,31,30,31,30,31,31,30,31,30,31
380 DATA 30,30,29,30,29,29,30,29,29,30,29,30,30
390 COLOR 14:FOR J=1 TO 6
400 PRINT TAB (3+J*10);NAM$(J);
410 NTXT J
420 COLOR 4:PRINT TAB (73);NAM$(7)
430 COLOR 1
440 RETURN
```

用拉幕方式显示信息

烟台师范学院(264000) 刘善平

各种信息的显示都需要有个好的显示效果,尤其是应用程序的封面和菜单的显示。目前,各种程序显示信息多采用滚屏式或下拉式。这里,提供一种更美观、更新颖的显示方式:拉幕式。

设计思想:将需要显示的信息以文件的形式存入

磁盘,显示时从盘中取出数据装入4K长度的显示缓冲区,而后按相应顺序把缓冲区的数据显示出来。显示前,先分别从左右两边向中间显示反相的空串列,这样不论原先屏幕上有什么内容都能有一种拉闭屏幕的效果。显示时,再从中间分别向左向右依次显示被存储屏幕的每一列的信息。这样屏幕就又被拉开了,所需的画面也就呈现在眼前。

具体实现:根据上述设计思想,笔者编写了汇编语言源程序 SAVE. ASM 和 LOAD. ASM。读者可用编辑程序输入 SAVE. ASM 和 LOAD. ASM,进行编译和连接以后,再用 EXE2BIN 分别生成 SAVE. COM 和 LOAD. COM。SAVE. COM 的功能是:把要显示的信息以文件的形式存入磁盘。LOAD. COM 的功能是:将需显示的内容从磁盘上取出,再以拉幕的方式显示出来。比如:在 dBASE III 环境下,可以编程序用 TEXT—ENDTEXT 或 SAY 语句将画面制作好,用 RUN SAVE. COM 把画面存入磁盘。在需要显示时再用 RUN LOAD. COM 把画面以拉幕的方式显示出来。

SAVE. ASM 清单:

PRG1 SEGMENT PARA PUBLIC 'CODE'

ORG 100H

ASSUME CS:PRG1,DS:PRG1,ES:PRG1,SS:PRG1

PRG2 PROC NEAR

JMP NM1

DATA DB 4000 DUP(?) ;行数×列数×2

FNAM DB 'C:SCR',0 ;路径及文件名

FILE DW 0

NM1:MOV AH,3CH

XOR CX,CX

MOV DX,OFFSET FNAM

INT 21H

MOV AH,3DH

MOV AL,1

MOV DX,OFFSET FNAM

INT 21H

MOV FILE,AX

MOV SI,0

MOV DH,0

NM2:MOV DL,0

NM3:MOV AH,2

MOV BH,0

INT 10H

MOV AH,8

INT 10H

MOV [DATA][SI],AL

INC SI

MOV [DATA][SI],AH

INC SI

INC DL

CMP DL,80

JNE NM3

INC DH

CMP DH,25 ;行数

JNE NM2

MOV AH,40H

MOV BX,FILE

MOV CX,4000 ;行数×列数×2

MOV DX,OFFSET DATA

INT 21H

MOV AH,3EH

MOV BX,FILE

INT 21H

MOV AX,4C00H

INT 21H

PRG2 ENDP

PRG1 ENDS

END PRG2

LOAD. ASM 清单:

PRG1 SEGMENT PARA PUBLIC 'CODE'

ORG 100H

ASSUME CS:PRG1,DS:PRG1,ES:PRG1,SS:PRG1

PRG2 PROC NEAR

JMP NM1

DATA DB 4000 DUP(?) ;列数×行数×2

FNAM DB 'C:SCR',0 ;路径及文件名

FILE DW 0

NN DB 0

NM1:MOV NN,80

MOV DL,80

NM2:SUB DL,NN

MOV DH,0

NM3:MOV AH,2

MOV BH,0

INT 10H

MOV AL,' '

MOV BL,112

MOV BH,0

MOV AH,9

MOV CX,1

INT 10H

INC DH

CMP DH,25 ;行数

JNE NM3

DEC NN

ADD DL,NN

MOV DH,0

NM4:MOV AH,2

MOV BH,0

INT 10H

MOV AL,' '

MOV AH,9

MOV BH,0

MOV BL,112

MOV CX,1

INT 10H

INC DH

CMP DH,25 ;行数

JNE NM4

DEC NN


```

CMP NN,0
JNE NM2
MOV AH,3DH
MOV AL,0
MOV DX,OFFSET FNAM
INT 21H
MOV FILE,AX
MOV AH,3FH
MOV BX,FILE
MOV CX,4000;列数×行数×2
MOV DX,OFFSET DATA
INT 21H
MOV NN,0
MOV DL,39
NM5:SUB DL,NN
MOV DH,0
NM6:MOV AL,160
MUL DH
MOV SI,AX
MOV AH,2
MOV BH,0
INT 10H
MOV AL,2
MUL DL
ADD SI,AX
MOV AL,[DATA][SI]
MOV BL,[DATA][SI+1]
MOV BH,0
MOV AH,9
MOV CX,1
INT 10H
INC DH
CMP DH,25;行数
JNE NM6
INC NN
ADD DL,NN
MOV DH,0
NM7:MOV AL,160
MUL DH
MOV SI,AX
MOV AH,2
MOV BH,0
INT 10H
MOV AL,2
MUL DL
ADD SI,AX
MOV AL,[DATA][SI]
MOV AH,9
MOV BH,0
MOV BL,[DATA][SI+1]
MOV CX,1
INT 10H
INC DH
CMP DH,25;行数
JNE NM7

```

```

INC NN
CMP NN,80
JNE NM5
MOV AH,3FH
MOV BX,FILE
MOV CX,4000;列数×行数×2
MOV DX,OFFSET DATA
INT 21H
MOV AH,3EH
MOV BX,FILE
INT 21H
MOV AX,4C00H
INT 21H
PRG2 ENDP
PRG1 ENDS
END PRG2

```

拼 图 游 戏

浙江杭州市米山路60号(310022) 刘晓峰

目前,市场上有一种拼图板,由八块小方块组成,各个小方块上分别是图案的一部分,通过一定的规则移动它们,使它们拼成一幅完整的图案。

我在XZ-PC机上,利用GW BASIC的图形语句,用数字代替图案,实现这一游戏过程的模仿。游戏者利用I:↑、K:↓、J:←、L:→、四键来控制小方块的移动。使它们最终成为规则的顺序排列。

本程序主要是利用了PUT和GET语句,把每个小方块看成一个独立的部分,用一个数组Y(3,3)与之对应,使屏幕上小方块的排列与二维数组Y对应。由改变Y(3,3)的值来实现小方块的动态移动。

程序20—60构造小方块。80—127产生小方块的一种随机组合。500—2600通过键盘控制小方块,并由程序判断有否成功。

```

5 DIM A1(35),A2(35),A3(35),A4(35),A5(35),A6(35),
  A7(35),A8(35),A9(35)
10 SCREEN 1:CLS:KEY OFF
20 LINE (7,0)-(30,20),2,B
30 FOR I=0 TO 8:T=I+1
40 LOCATE 2,3:PRINT CHR$(49+I);
43 IF T=1 THEN GET (7,0)-(30,20),A1:GOTO 60
45 IF T=2 THEN GET (7,0)-(30,20),A2:GOTO 60
47 IF T=3 THEN GET (7,0)-(30,20),A3:GOTO 60
49 IF T=4 THEN GET (7,0)-(30,20),A4:GOTO 60
51 IF T=5 THEN GET (7,0)-(30,20),A5:GOTO 60
53 IF T=6 THEN GET (7,0)-(30,20),A6:GOTO 60
55 IF T=7 THEN GET (7,0)-(30,20),A7:GOTO 60
57 IF T=8 THEN GET (7,0)-(30,20),A8:GOTO 60

```

```

59 IF T=9 THEN GET (41,2)-(62,20),A9
60 NEXT;CLS;CS=0
80 LINE(0,0)-(190,300),2,B
100 LINE(40,40)-(119,110),2,BF
105 DIM B(9),Y(3,3):FOR I=1 TO 9:B(I)=0:NEXT
106 IF INKEY$=" " THEN B(0)=RND;GOTO 106
107 FOR I=1 TO 9
112 IF I<=3 THEN T=I;U=1
113 IF I>3 AND I<=6 THEN T=I-3;U=2
114 IF I>6 THEN T=I-6;U=3
115 B(0)=INT(1+RND(1)*9)
120 IF B(B(0))=1 THEN 115
125 Y(T,U)=B(0);GOSUB 130
127 NEXT;GOTO 500
130 IF Y(T,U)=1 THEN PUT (17+T*23+4,20+U*20
+2),A1,AND;B(1)=1;GOTO 300
150 IF Y(T,U)=2 THEN PUT (17+T*23+4,20+U*20
+2),A2,AND;B(2)=1;GOTO 300
170 IF Y(T,U)=3 THEN PUT (17+T*23+4,20+U*20
+2),A3,AND;B(3)=1;GOTO 300
190 IF Y(T,U)=4 THEN PUT (17+T*23+4,20+U*20
+2),A4,AND;B(4)=1;GOTO 300
210 IF Y(T,U)=5 THEN PUT (17+T*23+4,20+U*20
+2),A5,AND;B(5)=1;GOTO 300
230 IF Y(T,U)=6 THEN PUT (17+T*23+4,20+U*20
+2),A6,AND;B(6)=1;GOTO 300
250 IF Y(T,U)=7 THEN PUT (17+T*23+4,20+U*20
+2),A7,AND;B(7)=1;GOTO 300
270 IF Y(T,U)=8 THEN PUT (17+T*23+4,20+U*20
+2),A8,AND;B(8)=1;GOTO 300
280 IF Y(T,U)=9 THEN PUT (17+T*23+5,20+U*20
+3),A9,AND;B(9)=1;JX=T;JY=U
300 RETURN
500 LOCATE 17,3;PRINT "i."CHR$(24);" " "k."CHR
$(25);" "; "l."CHR$(26);" "; "j."CHR$(27);
LOCATE 19,8;PRINT "e:end"
510 LOCATE 21,5;PRINT "?";I$=INKEY$;LOCATE 2,2;
PRINT CS,TIME$
545 IF I$="e" OR I$="E" THEN 2600
550 IF I$="i" OR I$="I" THEN Y(JX,JY)=Y(JX,JY+
1);Y(JX,JY+1)=9;GOSUB 1300
570 IF I$="k" OR I$="K" THEN Y(JX,JY)=Y(JX,JY
-1);Y(JX,JY-1)=9;GOSUB 1300
590 IF I$="l" OR I$="L" THEN Y(JX,JY)=Y(JX-1,
JY);Y(JX-1,JY)=9;GOSUB 1300
610 IF I$="j" OR I$="J" THEN Y(JX,JY)=Y(JX+1,
JY);Y(JX+1,JY)=9;GOSUB 1300
620 GOTO 510
1300 LINE(40,40)-(119,110),2,BF;CS=CS+1
1500 FOR T=1 TO 3;FOR U=1 TO 3
1640 W=W+1;GOSUB 130
1800 NEXT U:NEXT T;GOTO 2000
1900 RETURN
2000 YSZ=0
2050 FOR W=1 TO 3;FOR V=1 TO 3

```

```

2150 YSZ=YSZ+1
2200 IF P(V,W)<>YSZ THEN 1900
2300 NEXT V:NEXT W
2400 LOCATE 21,4;PRINT "Continue?(y/n)";INPUT A$
2500 IF A$="y" THEN 70
2600 SCREEN 0;END

```

通用文件病毒防御系统

合肥矿山机器厂研究所(230011) 姜金友

文件病毒是一种行为恶劣的病毒,它的载体是扩展名为 .COM 和 .EXE 的程序。现已发现的文件病毒有:黑色星期五、扬基、维也纳、音乐、1590(1575)等,这类病毒不象引导型病毒那样容易预防(只要用无病毒的系统盘引导即可),并且破坏性大,隐蔽性强,发作时常毁坏文件。现有的免疫系统都只能预防某种具体的文件病毒。有没有一种方法能够预防各种各样文件病毒的发作和传播呢?答案是肯定的。

笔者通过分析现今出现的各种文件病毒,发现它们有一个共同之处,即都是通过修改 INT21H 的入口,使其指向病毒体而获得控制权。由此可见,只要禁止修改 INT21H 的入口地址,则病毒即使驻留内存,也无法传染给其他文件,更谈不上破坏了。

如何禁止修改 21H 中断向量?可以在系统未染上文件病毒之前,修改 INT21H 使之指向免疫程序。该免疫程序常驻内存,它首先获取 21H 中断向量的入口地址,再同正确的相比较,若 INT21H 入口被修改,则再次修改 INT21H 入口,使之重新指向免疫程序,然后再执行 INT21H 所要求的功能调用。以下是根据该原理开发的免疫软件源程序清单。

C>TYPE PFV. ASM

```

pushr      macro
push ax
push bx
push cx
push dx
push ds
push es
endm
popr        macro
pop es
pop ds
pop dx
pop cx
pop bx
pop ax

```

```

endm
call_old_int macro
    pushf
    cli
    call dword ptr cs:old_int21
    sti
endm
code segment
    assume cs:code
    org 100h
start:
    jmp install
old_int21 dw?,?
int21 dw?,?
msg1 db "Prevent V1. 0 by J. Jinyou, 6/6/
1991",24h
msg2 db 0ah,0dh,"Your computer now may
avoid files virus!!!",24h
msg3 db 0ah,0dh,"Prevent already in-
stalled!!!",24h
flag dw?
new_int21 proc far
    pushr
    push cs
    pop ds
    mov ax,3521h
    call_old_int
    cmp int21,bx
    jnz recover
    mov ax,es
    cmp int21+2,ax
    jz ok
recover:
    mov ax,2521h
    lea dx,new_int21
    call_old_int
ok:
    popr
    cli
    jmp dword ptr cs:old_int21
new_int21 endp

install:
    assume ds:code
    push cs
    pop ds
    mov ah,09
    lea dx,msg1
    int 21h
    mov ax,0
    mov ds,ax
    mov bx,182h
    cmp word ptr [bx],4a59h
    jz exit

```

```

    mov ax,4a59h
    mov [bx],ax
    push cs
    pop ds
    mov ah,09
    lea dx,msg2
    int 21h
    mov ax,3521h
    int 21h
    mov old_int21,bx
    mov old_int21+2,es
    mov ax,2521h
    lea dx,new_int21
    call_old_int
    mov ax,3521h
    call_old_int
    mov int21,bx
    mov int21+2,es
    lea dx,install
    mov cl,4
    shr dx,cl
    inc dx
    mov ax,3100h
    int 21h
exit:
    push cs
    pop ds
    mov ah,09
    lea dx,msg3
    int 21h
    ret
code ends
end start

```

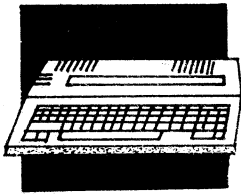
编译并连接上程序后,用 EXE2BIN.EXE 将之转化为.COM 文件。

笔者经过多次验证发现,一旦在内存中加载了 PFV.COM 程序,就可以预防迄今为止所发现的所有文件病毒,当然要在内存中尚无文件病毒时运行该程序(最好修改 AUTOEXEC.BAT 文件使之一开机就执行 PFV.COM)。

《电子电脑》1991年下半年合订本已经出版,每本定价7.8元,欲购者请附加15%邮费。由于数量有限,需要者请从速。

邮购办法:邮局汇款 北京万寿路173信箱电子工业出版社《电子与电脑》编辑部

银行汇款 开户行北京市工商银行翠微路分理处帐号891238—72



学习机之友

苹果 BASIC 语言容错性浅析

沈阳黄金学院(110015) 那履弘

苹果机浮点 BASIC (APPLESOFT) 是解释性语言, 修改原程序较为方便, 使用这一语言的人多为青少年和初学者, 因此, 对于大多数的使用者来说, 往往在编制程序时不太注意程序语法, 而希望计算机运行解释程序时能甄别出自己程序中的错误的性质和出错的行号, 这就是通常所说的“动态侦错”。

在解释性语言中, 动态侦错的完善性是容错性的一个很重要的方面。容错性具有容留错误的含义, 但是它所容留的错误不应导致程序的失误, 因此它与侦错是不可分割的。一般来说, 作为一种高级语言, 应该具有充分完善的容错性, 应能识别出用户编制的程序中的任何导致失误的错误。但是, 解释程序也是某个或是某些人编制的, 由于程序规模和思考问题等各方面的原因, 难免存在着一些错误的情况超出了预先设计的范围, 因而造成某些错误不能侦出, 有些出错处理不当的情况。

由于人们对于容错性缺乏深入的了解, 往往把通过运行的程序认为是正确的程序。这个结论不完全正确。实际上, 除了数据、公式和程序流程等方面的错误之外, 仍有一些语法上的错误能混过解释程序的侦错检查。本文重点指出一些程序运行中 APPLESOFT 不能发现的语法错误和一些侦错后处理不当的例子。在这些例子中, 有些可能导致错误的计算结果、错误的程序流向; 有些可能造成“死机”的情况。

为使读者对容错性有进一步的认识, 本文还针对一些实例, 对容错性进行粗浅的分析, 希望能对读者, 特别是编制小规模商业化软件的读者有所帮助。

为叙述方便, 文中把以行号为标志的语句称为语句行或行, 把以“:”分隔的语句称为语句或句, 并将语句中可以替换的关键字用“()”括起来。

一、行号侦错

BASIC 程序的每一条叙述称为语句, 语句的标志是行号。浮点 BASIC 语言的行号由纯数字构成, 取值范围是 0—63999, 除了写在行首作为语句行的标志之外, 还有若干个命令以行号作为参数或可选参数。

这些命令是

```
GOTO;          GOSUB;
ON (开关变量表达式) GOTO;
ON (开关变量表达式) GOSUB;
ON ERROR GOTO;
IF(表达式) THEN(GOTO)(或 GOSUB);
RUN;  DEL;  LIST.
```

APPLESOFT 有一段编辑解释程序, 每当使用者键

入一段程序并按下 RETURN 键, 这段解释程序便将键盘缓冲区的程序移入程序区。写在行首的行号在这个过程中侦错, 任何在取值范围内的, 以数字开始的连续的数码被解释成合法的行首行号。行首行号是不容错的, 如果插进了非数字代码, 那么解释程序仅将前面的数码作为行号, 非数字代码和后面的代码将作为语句行的内容, 并在程序运行时予以解释。但是上面列举的 GOTO 等命令的行号参数不在编辑时侦错, 这是解释性语言逐句解释的特点所决定的。这些非行首的行号参数随着它所依附的命令的不同, 也有着不同的容错性, 下面分别列举几种情况加以分析。

1. GOTO 语句

语法书中介绍 GOTO 语句是由 GOTO 命令和数字行号两部分构成的; 行号不能省略。请看下面例子。

```
0 PRINT " * ";
10 GOTO
```

运行这段程序, 屏幕上会布满“*”号, 实际上进入了死循环状态。

分析解释程序可知, 程序的行号无论写在哪一个命令之后都首先由一段共同的子程序来解释。这个子程序首先将存放行号的单元清零, 再去找数字, 将数字的值存入行号单元, 一旦发现非数字代码便返回主程序。这段子程序只能发现行号超出取值范围的错误, 其他的错误不能侦出。在这个例子里, GOTO 命令后面没有跟随行号参数, 因此解释程序便认为 0 是要转移的行号。很多程序不常有 0 行, 省略行号便会出错, 因此初学者认为省略行号是非法的。一些计算机语言专家认为, 类似行号问题, 解释系统不应提供缺省值, 以防止引起误解, 这个观点是很有道理的。

由于行号的解释程序遇到非数字代码便认为行号结束, 实际上引用了“非”逻辑。非数字代码不是唯一的, 因此行号参数便有了容错性。通过分析可知:

```
GOTO 10.5      等于    GOTO 10
GOTO 20+A      等于    GOTO 20
GOTO B         等于    GOTO 0
```

看起来上述语法错误是不容易出现的, 但是一些初学者可能将数字零打成字母 O; 在键入数字时, 可能夹入不显示出来的控制字符, 这些错误初学者不易发现。由此可以看出引用“非”逻辑后侦错是不够强有力的。GOTO 语句是绝对转移语句, 除了 DATA 和 REM 语句之外, 任何其他的命令写在 GOTO 语句行的后面均没有实际的意义。解释程序考虑到 GOTO 语句后面没有可执行语句, 便不继续侦错是有一定道理的, 但是类似

夹入控制符之类的错误很难由屏幕清单上反映出来,因此也具有不利因素。作为一个系统软件也不应该以含混的解释形成所谓的技巧。实际上在 GOTO 语句中再对非数字代码进行验证,看其是否为“:”号或零,那么就不能出现上述的问题,引用“是”逻辑只占用很少的字节,代价不大。

2. GOSUB 语句

GOSUB 语句的格式与 GOTO 语句的格式有相似之处,也需要有行号作为它的参数,实际上由于对行号的解释用同一段子程序,因而也会出现 GOTO 语句中出现的问题。由于 GOSUB 语句要与 RETURN 语句成对使用,而 RETURN 语句又难以写在 0 行,因此 GOSUB 语句省略行号是难以实现的。GOSUB 语句也有容错性和侦错方面的问题,请看下例。

```
10 A=20:GOSUB 60 B=A:A=0
20 PRINT B,A:END
60 RETURN
```

在第 10 语句中,显然是丢掉了 GOSUB 语句后面的“:”号。但是运行这段程序,并没有出现错误中断或出错提示;打印出的 B,A 两值均为零。通过这个现象可以看出,赋值语句 B=A 没有执行,因而 B=0。分析解释程序可知,GOSUB 语句要经 RETURN 语句返回到本行,因此要保护现场,也就是要将 GOSUB 语句当前的地址、行号和 GOSUB 命令的标志符入栈保护。为了简化解释程序;为了使行号具有容错性,解释程序在解释 GOSUB 语句时,找到行号之后,它不是把下一字符的地址入栈,而是要找到“:”后,再将地址入栈。所有夹在行号和“:”之间的其他代码统统作为废料处理。在这个原则下,就有下面的关系:

GOSUB 60.5 等于 GOSUB 60

不出错,这样容错性似乎广泛了,但是也出现了可能丢失语句的问题。如果错将 GOSUB 语句后面的“:”号打成“;”号,也会发生丢失语句的情况,这种错误是解释程序不能侦出的,人的检查也容易忽视两个符号之间的差别。全面衡量利弊,该语句的侦错功能似应加强,对这一语句的容错性应予以重新认识。

3. ON...GOTO(GOSUB) 语句

ON...GOTO(GOSUB) 可有多个行号作为参数,其他有两个以上行号参数的语句还有 LIST,DEL 等。

解释程序在解释以多个行号为参数的语句时,侦错要严格一些,解释程序不仅检查行号是否在取值范围,而且还要核对行号之间的分隔符号代码是否符合规定。请看下例。

```
10 ON A GOTO 50,60.5,,70,80.8
   当 A 值等于 1 时,相当于 GOTO 50;
   当 A 值等于 2 时,相当于 GOTO 60;
   当 A 值大于 2 时,程序将侦出错误,即行号 60
```

后面的分隔符不是所要求的逗号。如果将行号 60 后面的“.5”去掉,那么:

当 A 值等于 3 时,相当于 GOTO 0,在这里,解释程序又一次引用了缺省值。实际上,我们在 ON...GOTO

语句中用连续的逗号,往往是出于节省程序字节的考虑,并非有意转至 0 行,而对应的开关变量 A 值也是非预期的。如果不用缺省值的方法,那么对于侦出非预期值也是有帮助的。当开关变量指向最后一个行号时,ON...GOTO 语句会出现与 GOTO 语句一样的问题。

从上面的几个语句中可以看出,语法书中往往只列举正确的语法格式,而对解释系统的容错性介绍不够,在语法和解释程序之间的对应关系呈单向性;也就是说,动态侦错不能侦出所有的语法错误,通过运行的程序不一定没有错误。

二、表达式侦错

在学习浮点 BASIC 语言时,经常要用到表达式(EXPRESSION)的概念。算法语言书籍中没有为表达式作出严格的定义,一般仅称表达式是:“最终可以得出一个值的运算式”。这样的定义是较为含糊的。APPLESOFT 中规定了 3 种数据类型,即整数、实数和字符串,而没有逻辑型或布尔型量,那么最终得出的值是一个什么类型的量,初学者是很难掌握的。一些表达式的特殊用法难以在语法书籍中找出与之对应的举例。

表达式中经常出现常数、变量、函数;可以进行数值运算、关系运算和布尔运算。表达式的分析和侦错较为复杂,随着内涵的不同,表达式中可能出现的错误种类也很多,有些语法上的错误是不能依靠动态侦错的方法来发现的。下面就表达式中关系运算和布尔运算中不能由动态侦错发现的错误分述如下。

1. 关系运算

浮点 BASIC 语言没有为布尔值定义专门的数据类型,一个表达式的值究竟是整型量、实型量,还是布尔型量,要靠算式本身来确定。如果在表达式中写进了关系运算符,所得出的表达式的值就可能是布尔型量。实际上由于 APPLESOFT 没有明确的布尔型量,那么,布尔量也可以参加数值计算。从这个意义上说,有关系运算符的表达式也可以得出其他类型的量。

在表达式中有三种常用的关系运算符,分别为等量、大于和小于;与之对应的表达符号分别为=、>和<。按照语法定义,这三种关系运算符可以单独使用;或者是不同的关系运算符两两组合使用。除此之外的任何组合形式在算法语言中是没有定义的,皆应视为非法的组合,也就是出错。所有的合法组合如下:

符号组合	定义
=	等于
>	大于
<	小于
>=或=>	大于或等于
<=或=<	小于或等于
<>或><	不等于

在解释程序中,专门在零页中设置了一个单元用来记载表达式中是否存在关系运算符,以及所出现的关系运算符属于哪一种组合。该单元的最低位用来记载表达式中是否出现关系运算符>;(有此关系运

算符时,该位为1,否则为0,以下类似)这一单元的第一位和第二位分别用来记录表达式中是否出现关系运算符=号或<号。在甄别关系运算符的组合是否合法时,解释程序用两次异或的运算来侦错和记录。这种判别较为简捷,可以侦出同一个关系运算符在一次关系运算中出现两次和两次以上的错误。例如:

A>>B 或 C<=<D 等。

但是异或运算不能判别三个不同的关系运算符连写的错误,例如:

A>=<B 或 C=<>D 等。

尽管在编辑程序的过程中不容易出现这样的错误,但是就其容错性来说,它已经丧失了同语言、语法所设计的正确性之间的对应关系。作为一个容错性较好的软件,不应以错误出现的概率高低而做为是否进行侦错的依据。对于三个关系运算符同时连用的情况,要么予以定义,要么按出错处理,这样才更加完善。一般来说,如果在语句中出现了三个不同的关系运算符连接在一起使用,由于解释程序不能侦出错误,程序会继续执行,而将该运算的结果视为真,也就是1。如果这一表达式写在IF...THEN语句中,那么程序流向将是忽视判别条件而执行THEN语句。如果关系运算的各个常量和变量的值均为零,那么类似表达式的值会因得不到正确的比较而取值为0。这些复杂的关系是初学者难以记忆的,因此要注意此类错误。解释关系运算符时,解释程序要利用栈区,这种错误不仅可能导致程序的错误流向和错误的计算结果,而且在特殊的情况下,还可能出现死机的严重情况。

2. 布尔量的数值运算

在苹果机浮点 BASIC 语言中,布尔量可以通过关系运算或布尔运算得到,同时布尔量也可以参与其他的数值运算。例如:

```
10 A=5:B=5:C=(A=B)*10:PRINT C
```

运行该句,可知变量C取值为10,这句程序是符合苹果机浮点 BASIC 语言的语法的。

在一般情况下,布尔量经常用来判别条件式的真伪,进而控制程序的流向。但是语法书中并没有详细讨论布尔量的数值运算问题;也没有研究布尔量进行四则运算和其他运算的规则。从语言系统应具备的严格性上来讲,要么容许布尔量参加非布尔运算,使布尔量完全等同于数量0或1;要么不容许布尔量参加非布尔运算,一但在程序中出现此种情况,就按照出错处理。APPLESOFT 采用了前种灵活的方式,但是它在这些问题的解释上还缺乏慎重的思考,在处理稍微复杂的关系运算时,可能出现混乱的情况。下面我们用一段程序来说明这个问题。

```
10 A=10:B=5:C$="B":D$="A"
```

```
20 E=(A AND B)+(C$>D$)
```

```
30 PRINT E:END
```

这段程序是符合语法的。不难分析,式中 A AND B 和 C\$>D\$ 两种运算皆为真,两布尔量相加后,为变量E赋值为2。

如果去掉20语句中的括号,该句程序变为:

```
20 E=A AND B+C$>D$
```

依照该句中运算符的优先级进行分析,可知三个运算符的优先顺序为+,>,AND。按这一顺序,在执行这句程序时,首先要进行加法运算,不难看出,变量B与字符串C\$分别属于两种不同类型的数据,它们之间不能运算,这句程序有语法上的错误。

运行改后的程序,APPLESOFT 没有提示出错,而将变量E赋值为1。

严格来说,这种情况已经超出了容错性的范畴,是解释程序的失误。由于APPLESOFT对各种数据类型没有明确的区分,许多语法书中又介绍了类似的技巧,使得一些初学者愿意利用上面的方法编程。但是这种灵活性必须有侦错来保证,否则是容易出错的。

在上面的例子中,如果将第20语句作如下改动

```
20 E=C$>D$+A AND B
```

再运行这段程序便会得到数据类型不匹配的错误提示。为什么前者的错误没有被侦出,仅仅交换了顺序便能侦出错误?这些问题是较为复杂的,要借助汇编语言和汇编语言跟踪才便于解释清楚。由于篇幅和主题关系,本文对此不加详述。仅此可以看出APPLESOFT在表达式的解释上,既存在着比较灵活的方面;也存在着容易发生混乱的方面。很多人喜欢追求程序的技巧和简捷,经常把各种数据类型混合在同一个表达式中,这是要十分谨慎的。实际上,程序语言严谨是更好的编程指导思想,这一点对初学者尤为重要。

三、命令保留字的容错问题

苹果机浮点 BASIC 语言有99个保留字,它们已经作为64个命令、25个函数和运算符的专用字。语法书中已经强调,这些保留字不应作为变量名称和夹入变量名中使用。由于内存容量的限制,解释程序不能甄别此类错误,因此解释程序的侦错与语法书中定义的正误是不对应的,这种情况更不能动态侦错。

上述错误的识别只能靠人来进行,而初学者又很难将近百个保留字一下子全记住。因此系统解释程序仍然在一些危险的命令中加入简单的安全措施,如:

```
200 NEW A=50
```

当程序执行到这一句时,解释程序会因NEW命令后没有“:”或或语句行结束的标志而不执行NEW命令。这种作法实际上是将NEW后面的语句分隔符与命令合在一起来解释的,在核对命令时必须命令本身和命令之后的分隔符同时正确,那么此保留字才作为命令执行。事先对保留字代码之后的分隔符侦错的命令有7条,它们是:

```
CLEAR,      CONT,
END,         NEW,
POP,         RETURN,
STOP
```

这些命令或者写在语句的最后,或者有“:”跟随其后才能被执行。

另一些无参数命令的危险性要小一些,例如:

我们可以将这一个语句行看成是丢掉了分隔符“:”号,或是编者误将 HOMEA 当作了变量名称,计算机在运行这一语句时,一旦查到 HOME 命令之后,便先将字符屏幕清除,再往下执行时,才会发现语法错误。这样的无参数命令有16个。如果详细分析起来,16个命令破坏程序的危险性也不是相同的。例如 GR, HGR, HGR2等命令,在程序较长或数组较大时,如果误用这类命令,在显示区内的程序和数组会被消掉。因此,对这样的保留字也必须正确使用。

四、结束语

为一种语言设计良好的容错环境是很重要的工作,同时也是很复杂的工作。在不影响程序正确运行的前提下,应尽可能地扩大容错范围,但是由于采用了容错技术,侦错的范围会有所扩大,一些意想不到的错误会影响程序的正确性。容错技术的采用也不应该增加附加的语法,或使语法变得难于理解。从这些意义上讲,容错和侦错是对立和统一两个方面,不能单纯强调一个方面而忽视另一个方面。

本文虽然列举了一些苹果机浮点 BASIC 语言中存在的一些问题,即容错性是否适当和侦错功能是否应加强的例子,但是作者仍然给苹果机的解释程序以很高的评价。这一系统软件为我们编制小规模的系统软件提供了经验。算法语言是本世纪形成的一门知识,这一种知识目前仍在不断地深化和完善过程中,为了提高计算机知识的认识水平,深入地剖析一些系统软件是很重要的工作。尽管苹果机的内存容量较小,但是作为一个系统它是很健全的,小的系统便于分析,而从其中得到的结论可以推广到大的系统中去。

对于广大的非专业人员来说,发现解释程序等系统软件中的不完善之处也许是很困难的,但是非专业人员适当地了解一些系统软件的设计思想,对于提高计算机的应用水平是有很大大好处的。

Apple 内存校对发声程序

浙江绍兴稽山中学(312000) 连勤

机器语言程序由于占用内存小,运行速度快,深受计算机爱好者的青睐。但由于十六进制数的16个字符在键盘上分布较分散,输入时很容易出错。常用的检查方法是在监控中用地址1. 地址2 \checkmark 的方式显示内存值,然后看一眼屏幕,再看一眼原稿,这样检查,眼睛很吃力,也容易看错。我根据 Apple 讲话的原理,编了一个发声程序;使得计算机在显示内存值的同时,读出内存值,这样在核对机器语言时,只要眼看原稿,耳听计算

机就行了,即省力又准确。

初次使用本程序时,按下列步骤操作:

(1)取一台音质较好的录音机和一盒磁带,录下0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F,这16个音。注意:每个音之间有1~2秒的间歇,0音之前和F音之后也要留几秒钟的间隙。然后取一根电缆,一端插录音机的耳机(EAR)插孔或喇叭(EXT)插孔,另一端插 Apple 的录音输入插孔。

(2)分别在 BASIC 和监控状态下,正确无误地输入程序一和程序二。

(3)把磁带倒至开头,有音调旋钮的话,应开大高音,关小低音,把音量开关放在适中的位置。

(4)运行程序一,打开录音机,按一下计算机的任意键,即开始把声音信号输入计算机,同时计算机的喇叭也发出相应的音,如果计算机声音不正常,可调节录音机的音量开关。发完F音后,再等几秒钟,即可关掉录音机。

(5)等待大约4分钟后,计算机就依次发出0~F的每一个音,供你审听,如果不满意,按N键可退出程序一。在审听时,如果在某个音的位置连着发几个音,说明录音时,这几个音间的间歇太短,如果发的音和屏幕显示的数不符,说明环境噪音太大。出现这些情况,均应按步骤一重新操作。

(6)如果审听满意,则再等3分钟后,屏幕提示,让你插入磁盘,计算机会自动把程序二和刚生成的声音数据连接成一个以TALK为文件名的二进制文件存盘。因为文件较长,磁盘应有足够的空间。

经过以上步骤,以后需校对内存时,只要运行TALK文件就行了。TALK程序从\$1000开始存放,故使用时,先应把待校对的机器语言程序用监控M命令搬到TALK以外的地方,如\$1000以前,然后BRUN TALK \checkmark ,按屏幕提示输入起始地址、结束地址(4位16进制数表示),计算机就开始一边显示、一边读出内存值。如果需要修改内存单元的话,可按任意键退出TALK程序,待修改完以后,再用1000G运行之即可。

程序二的\$FB2~\$FFF是录音程序,用于把录音机放出的模拟信号变成数字信号存入内存,数据存放的起始地址低位由\$FB3、高位由\$FB8的值决定,结束地址高位存于\$FF9内。\$1000~\$1076的功能是输入并显示待校对内存的起始地址,结束地址,起始地址高位存\$3D,低位存\$3C,结束地址高位存\$3F,低位存\$3E。\$1077~\$1094是主控程序,用于显示内存地址、内存值,并转发音程序,读出内存的值。\$10AD~\$10C8的功能是把累加器A中的值分离成二个十六进制数,通过查表找到某个音的发音数据存放的起始地址和结束地址,\$10E5~\$111A是发音子程序,\$10E8、\$10E7分别存发音数据起始地址的高位和低位,\$110E、\$1116分别存结束地址的高位和低位。

为了帮助使用者迅速正确地找到每个音的有关发声数据在内存中的存放地址,我又编了程序一,它的功

能是自动找出发每个音的起始、结束地址,填入\$111B~\$115A内,并把程序二和声音数据合并后存盘,形成一个可直接运行的TALK文件。

```

7 ONERR GOTO 80
10 HOME
15 PRINT "PLEASE SWITCH ON TAPE RECORDER THEN
   HIT ANY KEY TO BEGIN"
25 IF PEEK(49152) < 128 THEN 25
30 CALL 4018
35 PRINT "RECORDING DONE, PLEASE WAIT FOR 4 MIN-
   UTES"
40 S = 36864: F1 = 260
45 DIM A(17), B(17)
50 FOR I = 8194 TO S
55 IF PEEK(I) = 0 THEN GOSUB 250: GOTO 75
60 FOR J = I + 1 TO S
65 IF PEEK(J) < > 0 THEN F1 = F1 + 1: NEXT
70 I = J - 1: W = I - F1: F0 = 0
75 NEXT I
80 A$ = "0123456789ABCDEF"
85 FOR I = 2 TO 17
90 PRINT "THE SOUND IS"; MID$(A$, I - 1, 1)
95 C = A(I): GOSUB 200: POKE 4328, A: POKE 4327, B
100 C = B(I): GOSUB 200: POKE 4366, A: POKE 4374, B
105 CALL 4325
110 INPUT "IS IT RIGHT?"; B$: IF B$ = "N" THEN
   END
115 NEXT
117 PRINT "PLEASE WAIT FOR 3 MINUTES"
120 DB = 4443: B(1) = DB - 1
125 FOR I = 2 TO 17
130 FOR J = A(I) TO B(I)
135 POKE DB, PEEK(J): DB = DB + 1
140 NEXT
145 C = A(I): A(I) = B(I - 1) + 1: B(I) = A(I) + B(I)
   - C
150 NEXT
155 DD = 4443 - 64
160 FOR I = 2 TO 17
165 C = A(I): GOSUB 200: POKE DD, A: DD = DD + 1:
   POKE DD, B: DD = DD + 1
170 C = B(I): GOSUB 200: POKE DD, A: DD = DD + 1:
   POKE DD, B: DD = DD + 1
175 NEXT
178 X = B(17) - 4096 + 1
180 S$ = "BSAVE TALK, A$ 1000,"
185 PRINT S$; "L"; X
190 PRINT CHR$(4); S$; "L"; X
195 END
200 A = INT(C / 256): B = C - 256 * A: RETURN
250 FOR J = I + 1 TO S
255 IF PEEK(J) = 0 THEN F0 = F0 + 1: NEXT
260 I = J - 1
265 IF F0 < 64 OR F1 < 260 THEN F1 = F1 + F0 + 2:

```

RETURN

```

270 N = N + 1: A(N) = W
275 M = M + 1: B(M) = I - 1 - F0
280 F1 = 0: RETURN

0FB2-- A2 00 8E C4 0F A9
0FB8-- 20 8D C5 0F AD 60 C0 29
0FC0-- 80 85 FF EE E6 34 D0 03
0FC8-- 20 DC 0F A5 FF C5 FE F0
0FD0-- EB 85 FE 8D 30 C0 20 DC
0FD8-- 0F 4C BC 0F EE C4 0F D0
0FE0-- 08 EE C5 0F A0 02 88 D0
0FE8-- FD AD C4 0F 8D F6 0F AD
0FF0-- C5 0F 8D F7 0F 8E E6 34
0FF8-- C9 90 F0 01 60 68 68 60
1000-- 20 58 FC A2 00 BD 95 10
1008-- 20 F0 FD E8 E0 0C 90 F5
1010-- A2 02 20 0C FD 20 F0 FD
1018-- 20 51 10 0A 0A 0A 0A 85
1020-- F9 20 0C FD 20 F0 FD 20
1028-- 51 10 29 0F 05 F9 85 3D
1030-- CE 2F 10 CA D0 DC A9 A1
1038-- CD 06 10 F0 2C 8D 06 10
1040-- 20 8E FD EE 2F 10 EE 2F
1048-- 10 EE 2F 10 EE 2F 10 D0
1050-- B2 49 B0 C9 0A 90 11 69
1058-- 88 C9 FA B0 0B BA E8 E8
1060-- 9A 08 20 8E FD 28 90 01
1068-- 60 A9 95 8D 06 10 A9 3D
1070-- 8D 2F 10 90 8E D0 06 A5
1078-- 3C 29 07 D0 03 20 92 FD
1080-- A9 A0 20 ED FD B1 3C 48
1088-- 20 DA FD 68 4C AD 10 20
1090-- BA FC 90 E3 60 D3 F4 E1
1098-- F2 F4 A0 C1 E4 E4 BA A0
10A0-- A4 C5 EE E4 A0 A0 A0 C1
10A8-- E4 E4 BA A0 A4 48 29 F0
10B0-- 4A 4A 20 C9 10 68 29 0F
10B8-- 0A 0A 20 C9 10 AD 00 C0
10C0-- 10 04 AD 10 C0 60 4C 8F
10C8-- 10 A8 BE 1B 11 8E E8 10
10D0-- C8 BE 1B 11 8E E7 10 C8
10D8-- BE 1B 11 8E 0E 11 C8 BE
10E0-- 1B 11 8E 16 11 18 AE 4D
10E8-- 00 86 FA A0 04 88 D0 FD
10F0-- EA CA D0 F7 A6 FA F0 08
10F8-- 8D 30 C0 A0 06 88 D0 FD
1100-- EE E7 10 B0 0D D0 DF EE
1108-- E8 10 AD E8 10 C9 00 4C
1110-- E6 10 AD E7 10 C9 00 38
1118-- D0 CC 60

```


ProDOS 磁盘操作系统入门(续)

廖 凯

4. 建立一个主菜单

下面的程序可以按照你需要的格式在屏幕上显示文字,它在屏幕四周产生一个边,这在显示菜单时很有用,你可以修改此程序以获得所期望的结果。

```
1 REM Screen output for a main menu
5 HOME
10 LL$="press ESC to Back Out"
80 GOSUB 900
89 REM MAIN MENU
90 L$="COMPUTE ASSET DEPRECIATION":V%
  =4:GOSUB 1130
100 L$="1. Straight Line Depreciation":
  H%=10:V%=10:GOSUB 1140
110 L$="2. Double Declining Balance":V%
  =12:GOSUB 1140
120 L$="3. Accelerated Method":V%=14:
  GOSUB 1140
130 GOSUB 1200
140 IF C%>48 AND C%<52 THEN 160
150 GOTO 140
160 GOSUB 910
200 REM DESCRIPTION OF PRODUCT
210 L$="PRODUCT DESCRIPTION":V%=4:GOSUB 1130
220 L$="Product Description":H%=10:V%=10:GOSUB
  1140
230 L$="Product cost":V%=12:GOSUB 1140
240 L$="Salvage value $":V%=16:GOSUB 1140
250 V%=22:L$=LL$:GOSUB 1140
260 GOSUB 1200:IF C%=27 THEN GOSUB 910:GOTO 90
890 END
899 REM BORDER ROUTINE
900 INVERSE:HOME:NORMAL
910 POKE 32,2:POKE 33,76:POKE 34,1:POKE 35,23:
  HOME:TEXT:RETURN
999 REM POSITION ROUTINE
1000 POKE 1403,H%:VTAB V%
1010 RETURN
1130 H%=INT(40-(LEN(L$)/2))
1140 GOSUB 1000:PRINT L$:RETURN
1200 REM GET A KEYBOARD COMMAND
1210 C%=PEEK(-16384):IF C%<128 THEN 1210
1220 C%=C%-128:POKE 49168,0
1230 IF PEEK(-18287)>127 THEN C%=C%+128
1240 IF PEEK(-16286)>127 THEN C%=C%+128
1250 RETURN
```

行号80将程序转到行号900的建边子程序,在屏幕四周建立一个反白的边,行号90至120用行号1130和行号1140的子程序,在屏幕上定位文字,建立主菜单,行号130转到行号1200键盘输入子程序等候用户输入。行号140检查用户是否按下1,2或3,如果符合条件,程序执行行号910,清除屏幕准备建立子菜单,在适当的子程序被执行后,子菜单被显示并等候用户输入。按 Esc 键即返回主菜单。

5. 清除屏幕

这是一个很短的子程序,它使用 POKE 屏幕指令清除屏幕而得到一种移离的效果,你可以结合这子程序来清除所有或部分屏幕。

```
10 INVERSE:HOME
20 FOR I=20 TO 0 STEP -2:POKE 32,I:POKE 33,80-I*
  2:POKE 34,I/2:POKE 35,24-I/2:NORMAL:HOME:
  NEXT
30 END
```

行号20经 FOR 循环设置 STEP-2为递减移动,这将使移动宽度变为两个字符宽度。POKE 33, 80设置屏幕宽度为整个80列,它可以用算式-I*2来改变。

四、连接打印机

有效利用打印机是程序设计的一个重要部分。连接打印机的接口有两种:并行接口和串行接口。一个并行接口通过8根线一次发送8位资料。一个串行接口经1根线一次发送1位资料。串行卡还可以用于与其它设备通信。

1. 有关指令

ON GOSUB 根据表达式的值,选择行号,使计算机转向该行号执行

PR# 发送输出到一个槽口或程序

2. 使用打印机

要发送输出到一个打印机,请用 PR# 命令,数字是用于指出打印机接口卡所在的槽口。

```
5 D$=CHR$(4)
10 PRINT CHR$(21)
20 PRINT D$;"PR#1"
30 PRINT "This is a test to send output to the printer"
40 PRINT D$;"PR#3"
50 END
```

行号10关闭80列卡。打印机可用 PR# 命令来选定或消除。在打印机与80列卡一起使用时一定要特别小心。建议在选择其它槽口之前先关闭80列卡。

3. 字符输出

大部分打印机可以打印多种格式和字符。以下程序是为 APPLE DOT MATRIX 和 IMAGE WRITER 打印机而设计的。APPLE DOT MATRIX 打印机使用一个并行接口。IMAGE WRITER 打印机使用 Super Serial Card。

```
100 REM Demo of APPLE Printer family
110 D$=CHR$(4)
120 V%=6
130 PRINT CHR$(21)
```

```

140 PRINT D$;"PR #1"
150 PRINT CHR$(9)"80N":REM send 80 columns to the
    printer
160 PRINT CHR$(27);CHR$(60):REM bidirectional
170 FOR I=1 TO 10
180 ON I GOSUB 1000,1010,1020,1030,1040,1050,1060,
    1070,1080,1090
190 VTAB(V%+2)
200 ON I GOSUB 1100,1110,1120,1130,1140,1150,1160,
    1160,1170,1180,1190
210 NEXT I
220 PRINT D$;"PR #3"
230 END

1000 L$="Here is a Demonstration":RETURN
1010 L$="of the":RETURN
1020 L$="APPLE DOT MATRIX":RETURN
1030 L$="and":RETURN
1040 L$="IMAGEWRITER PRINTERS":RETURN
1050 L$="as you can see":RETURN
1060 L$="we can control":RETURN
1070 L$="the size and":RETURN
1080 L$="overall makeup of":RETURN
1090 L$="the output":RETURN
1100 GOSUB 1280:PRINT SPC(78);L$:RETURN
1110 GOSUB 1270:PRINT SPC(46);L$:RETURN
1120 GOSUB 1220:GOSUB 1200:GOSUB 1310:
    PRINT TAB(12);L$:RETURN
1130 GOSUB 1230:GOSUB 1240:PRINT SPC(68);L
    $:RETURN
1140 GOSUB 1220:GOSUB 1310:PRINT TAB(10);L
    $:RETURN
1150 GOSUB 1230:GOSUB 1210:GOSUB 1300:
    PRINT SPC(75);L$:RETURN
1160 GOSUB 1240:PRINT SPC(64);L$:RETURN
1170 GOSUB 1310:PRINT SPC(31);L$:RETURN
1180 GOSUB 1260:PRINT SPC(48);L$:RETURN
1190 GOSUB 1290:PRINT SPC(37);L$:RETURN
1200 PRINT CHR$(27);CHR$(33):RETURN:REM
    start BOLD
1210 PRINT CHR$(27);CHR$(34):RETURN;
    REM BOLD off
1220 PRINT CHR$(14):RETURN:REM HEADLINE on
1230 PRINT CHR$(15):RETURN:REM HEADLINE off
1240 PRINT CHR$(27);CHR$(81):RETURN;
    REM ULTRA
1250 PRINT CHR$(27);CHR$(113):RETURN
    REM SEMI
1260 PRINT CHR$(27);CHR$(101):RETURN
    REM CONDENSED
1270 PRINT CHR$(27);CHR$(69):RETURN
    REM ELITE
1280 PRINT CHR$(27);CHR$(80):RETURN
    REM PROP ELITE
1290 PRINT CHR$(27);CHR$(78):RETURN
    REM PICA

```

```

1300 PRINT CHR$(27);CHR$(112):RETURN
    REM PROP PICA
1310 PRINT CHR$(27);CHR$(110):RETURN;
    REM EXTENDED

```

此程序内的大部分语句主要与打印机的格式有关。此程序运行打印的结果是10行不同字体的文字。

行号130关闭80列卡。行号140发送输出到槽口1的打印机。当发送输出到外设时，首先检查80列卡是否被启动。在80列卡启动时，发出输出到打印机会发生无法预料的结果。行号150设置打印机为80列输出，行号160设置双向打印，行号1200至1300是设置不同的字体。实际打印发生在行号1100至1190，在那儿设置间距，打印L\$，我们也可以使用POKE 36,X(X是分隔符位置)代替TAB和SPC语句，作为在页面上定位字符的另一种方法。行号220脱离打印机，设置屏幕为80列显示。

如果用户使用的是其它型号的打印机，就请参考打印机使用手册。

<未完待续>

(上接第33页)

显示2005

键入20✓

显示2006

键入00✓

显示2007

键入完毕后按下O键，便退出H命令。

②L命令

作用：以16进制数的形式显示存储单元之中的内容。

格式：L××××✓ ××××为四位地址，如列出从2000H开始的存储器单元之中的内容

键入L2000✓

显示2000 ×× ××为两位十六进制的数。

继续显示只要按下✓键，若不想再显示，则按下“O”键即可退出。

③E命令

作用：运行键入的程序

格式：E××××✓

例如：运行刚键入的由2000开始的程序

操作：E2000✓

想停止执行程序按下复位键即可。

四、操作步骤：

1. 开机，数码管显示“8031—A”，若显示的不是该字型可按下复位键。

2. 按下“W”键以后数码管不显示任何数字，此时表示监控程序已作好接收各种命令的准备。

3. 根据需要即可键入以上命令。

有需要本开发系统程序清单的可寄复制邮费3元至本文作者罗明宽、丰金相，地址：北京宣武区南横街西街宣武青少年科技馆 邮编：100052。

6502机器语言程序设计讲座

南京大学大气科学系(210008) 朱国江

苹果机和中华学习机的三大系统软件,即监控程序、浮点 BASIC 解释程序 Applesoft 和磁盘操作系统 Apple DOS,都是用6502机器语言编写的。要想知道这些系统程序怎样工作,如何有效地利用它们提供的丰富的功能以提高编程水平,就得学好6502机器语言。

在16位机和32位机日趋普及的今天,如何让相当大一部分现有的8位机资源不致闲置,引起人们的关注。除了在中小学电脑教育方面暂时还将保持一席之地外,许多有识之士呼吁将8位机改制成各种专用机,使它们在社会和家庭的各方面继续发挥作用。要搞好这一点,掌握6502机器语言也是一个重要条件。

我们根据许多读者来信表达的愿望,特邀请南京大学大气科学系朱国江副教授开设本讲座。朱老师在中华学习机的编程及应用方面有多种专著问世,深受读者欢迎。我们期望这个讲座对于想深入一层学习计算机的广大初学者有所帮助。随着讲座的开展,我们还将组织系统程序剖析和6502编辑、汇编程序使用介绍的稿件,欢迎广大读者、作者投稿支持。 编者

第一章 机器语言程序设计绪论

一、指令、指令系统

众所周知,不论计算机使用何种语言编写的程序,其本质都是按程序存储器中的内容依次动作的。这就是说,不论何种语言写成的程序,它最终都要转化为计算机程序区中的一个一个的8位二进制代码,计算机只能按这种转换好的代码动作。我们称这种最基本的代码为指令,而这种代码的集合称为该种型号计算机的指令系统。

不同型号的计算机具有不同的指令系统。中央处理单元 CPU(central processing unit)中具有指令译码器,显然送入某种计算机的指令必须是该计算机 CPU 中的指令译码器所约定的。所以一台计算机使用什么指令系统,取决于它采用何种型号的 CPU 芯片。APPLE-Ⅰ、紫金-Ⅰ及中华学习机是采用以6502微处理器为 CPU 的计算机,所以它们的指令系统是6502的。

例如,在指令提取运算时,当6502微处理器接到一个8位二进制代码“11101000”输入,意指:“将 X 寄存器内的值加1”。同样地,代码“10101001”则是:“将下一个内存单元所存之值移至累加器 A 中”。

微处理机仅“认识”二进制代码的指令或资料,它“看不懂”字组、八进制、十进制或是十六进制等数字。

二、程序及其执行过程

程序是由一系列的指令所组成,以使计算机执行特定的任务。计算机的程序不仅只包含指令,它也含有用以完成某任务的指令所需要用到的资料及存储器地址等。例如,微处理机要执行一个加法,则一定要有有两个待相加的数,且有一地方存放相加后的结果。所以,这个计算机程序除了具有执行运算的功能外,还应决定此两个相加数的来源,并且决定加完后结果存放在何处。

例如,将内存单元 \$ 60 与 \$ 61 内的值相加,结果存于 \$ 62 中,其6502机器语言程序为:

```
10100101
01100000
01100101
01100001
10000101
01100010
```

如果将上述程序,送入一个以6502为 CPU 的微电脑中,则此微电脑就能直接执行它。

程序执行的过程,就是微机工作的过程。由于程序是由指令序列组成,因此,执行程序的过程就是执行指令序列的过程,也就是不断地取指令、执行指令的过程。这个过程可简单地分为两个阶段。开始时,微处理机进入取指阶段。在取指阶段,从存储器取出指令,送指令寄存器,经译码器译码后,进入第二阶段,即执行指令阶段。在执行指令阶段,微处理机执行指令所规定的操作,指令不同,执行的操作也不一样。而在取指阶段,指令不同,取指令的操作却是相同的。总之,微机工作的过程,就是周而复始地取指令、执行指令的过程。

三、机器语言

用二进制代码编写的程序,称为机器语言程序。在机器语言中,指令是用二进制数字表示的。

显然,以二进制代码书写的程序(又称目的程序),阅读和理解都很困难,诸如:

- 二进制数目看起来都差不多相似,尤其在大量阅读数据以后,令人眼花缭乱,难以区别。

- 二进制码必须一个接一个地输入,稍不留心就会出错,并且输入速度十分缓慢。

- 无法了解计算机所要完成何种任务,执行的是何种操作。

- 程序冗长,编写麻烦,乏味,容易混淆,很难发现错误,纠正也极不方便。

- 很难记忆,因为人们无法像电脑处理二进制数那样容易、直接识别。

为了克服用二进制代码编制和书写程序的困难,人们采用十六进制,一方面因其简洁;另一方面则是它与二进制数有简单的换算关系。

例如,上面所举的两数相加的6502程序变为:A5 60 65 61 85 62

可以看出,一位十六进制数用来表示四位二进制数,如A代表1010,6代表0110,0代表0000等。这样,程序简明多了,不像二进制数那样麻烦,也容易找出错误。下表是几个基本的二进制、十进制和十六进制表的关系:

二进制	十进制	十六进制
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F
1111111111111111	65535	FFFF

前面谈到,机器语言是计算机能够直接识别的唯一语言,微处理器也只了解二进制指令代码,那么,现在改用十六进制代码,是否可行?唯一的办法是将十六进制数转换成二进制数。但这种转换是一种令人厌烦的重复劳动;人们极易疲劳。幸好,计算机却非常适合做这项工作,它绝对不会疲劳、厌烦、也不会造成愚蠢的错误。将十六进制数转换成二进制数,是许多微电脑特有的本领。

然而,用十六进制代码代替二进制代码后,并没有解决所有机器语言设计上的问题。因为用十六进制代码表示的机器语言程序,人们仍然难以阅读和了解,例如:我们区别不出何者是指令、资料或是地址;也无法知道程序在做什么,到哪里去;A5是代表什么?强记这些码,岂不是倒胃口吗?再说,不同类型的微处理机,这些码的含意也可能完全不同。因此,上述用十六进制代码表示的机器语言程序,虽比用二进制代码表示的机器语言程序有所改善,尚需要进一步改进。

四、汇编语言、汇编程序

如果把每一个指令码指定一个名称,那么机器语言程序设计就可以获得明显的改进。指令码的名称叫助记符,这是人们在不断实践过程中的一个创造。

助记符可以反映指令的功能和主要特征,用以为指令的动作做某些方面的陈述、注释。用助记符表示的

计算机语言叫做符号汇编语言,简称汇编语言(Assembly Language)。它是一种与计算机指令密切相关的计算机语言,直接用计算机指令的汇编符号来编写程序。

例如,前面的加法程序,用助记符表示则为:

```
LDA    $ 60
ADC     $ 61
STA     $ 62
```

这个程序虽然还不够明显,但至少其中某些部分已变得较易了解了。用“ADC”比“65”有了相当的改进。如第一条指令A5的助记符是LDA,它是英文Load A from memory(装入累加器A中的内容与存储器一致)的缩写。A是6502中的一个寄存器。指令LDA \$ 60的意思是取\$ 60存储单元中的数据放入寄存器A。在6502的指令系统中,采用\$号来表示后面的数字是十六进制数编址的存储单元。第二条指令65的助记符是ADC,它是英文Add memory to A with carry(取存储器中的内容连同进位标志位与寄存器A中的内容做加法,结果送入A中保存)。指令ADC \$ 61的意思是取\$ 61中的数据与进位标志C及累加器A中数据三者相加,并存入A中。第三条指令85的助记符是STA,它是store A in memory(把A中的内容存到存储器中)的缩写。指令STA \$ 62意即把寄存器A的数据存入存储器\$ 62单元中。

由此可知,上述简单程序中,我们可以知道那个部份是指令,那个部份是数据,又那个部分是地址了。

汇编语言比机器语言前进了一大步,它的优点在于:

- 用汇编符号表示指令,比较直观,容易记忆。
- 用标号代替地址,写程序时可不考虑转移地址的具体数值。
- 程序修改较为方便,不会因修改一条指令而牵动整体。
- 允许对源程序进行注释,便于阅读、理解。
- 允许操作数为简单的表达式或标号,编写程序方便。

前面已提到,计算机只能理解和执行二进制数代码的机器语言程序,因此,需要有专门的汇编程序(Assembler),将汇编语言编写的程序翻译成二进制的程序代码。常用的6502的汇编程序有EDITOR/ASSEMBLER(EDASM)和LISA等。

五、汇编语言的语句格式

下面以EDASM采用的符号来说明汇编语言的语句格式。

汇编语言编写的程序由一系列语句组成,每个语句单独占一行。程序由两类语句构成:一类是指令语句(即由指令构成的语句);另一类是伪指令语句(在程序执行过程中不产生操作)它们都必须按语句格式的规定来书写。

一个语句由四个字段组成,或者说分四个区:标号区、操作码区、操作数区和注释区。语句的一般格式为:

标 号 操作码 操作数 注 释

Label Opcode Operand Comment
 字段之间用空格作为分隔符,同一字段内不使用空格,而操作数与注释之间用分号隔开。

例如,有一段汇编语言程序:

标号	操作码	操作数	注释
START	LDA	\$ 8000	;将8000单元的内容送入累加器 A
	STA	\$ 8100	;将 A 内容送入8100单元

•标号

标号在语句中代表以字符串方式表达存储单元的地址。这种地址不是用十六进制数表示,而是用符号、文字表示的,称为符号地址。当某行指令之前冠有标号时,这个标号就代表该指令存储在程序存储区中的地址值。因此在转换为目标程序时,标号就被具体的存储单元地址所取代了,程序员不必算出程序中存储单元的具体地址值,这就使得程序单元容易查找和修改,也给编制和调试程序带来方便。如果给每个用标号的地址加一个常数,则能将整个程序段浮动或插入其它程序中,也可用于内存搬家,即用移动标号来修改程序。使用标号时应注意几点:

1. 标号必须从一行开头处开始,它必须是唯一的,即在一个源程序中一个标号只能定义一次。即使一个源程序包含多个文件,各文件间也必须遵守这一规定。
2. 标号必须以英文字母开头,以8个字符为限,以空格符作为结束。
3. 不能使用操作码助记符作为标号,也不得使用 A、X、Y、P、S 这几个寄存器的名字作标号。
4. 标号不是每个语句都必须的,可根据程序的需要选用。一般对于程序入口地址、程序转移地址、计数器单元、常数单元等都可使用标号。

例如:一个求和程序

	LDA	# 0	;和=0(初始化), #号表示后面是数,不是地址。
	TAX		;计数器(这里是 X 寄存器)置0
SUM	CLC		;清进位标志
	ADC	\$ 6000, X	;和=和+数据
	INX		;计数器加1
	CPX	# 10	;比较计数值是否等于10,即10个数是否加完
	BNE	SUM	;若不等,则未加完,继续求和
	STA	\$ 0340	;加完,送结果
	RTS		;结束

本程序是对从 \$ 6000 单元开始的十个数求累加和,最后结果存入 \$ 0340 单元,并假设和数不大于一个8位二进制数,这样可以不考虑进位。程序中 SUM 就是一个标号,当程序执行到 BNE SUM 时,若转移条件满足(上条指令执行后结果不为零, Z 标志=0),就转到标号为 SUM 相对应的存储单元,即执行 CLC 指令。

•操作码

6502微处理器有56种基本指令,一条指令包括一个字节(由八位二进制代码组成)至三个字节,不论单

字节指令或多字节指令,都包含两个部分:操作码部分和操作数部分。

操作码指明操作性质,它规定某条指令进行什么样的操作。不论单字节指令或是多字节指令,操作码都是放在指令的第一个字节中。操作码由指令助记符或伪指令构成,前者一律三个字母表示,后者由二个以上字母表示,例如:

ORG	\$ 1000	;程序起始地址是 \$ 1000
LDA	# \$ 40	;取立即数40到 A
STA	\$ 08	;再存到 \$ 08 单元中
BRK		;中断

上述小程序中,ORG、LDA、STA、BRK 都是操作码,其中 ORG 是伪指令,其余均为指令助记符。显然,操作码是每一个语句的必备字段。

•操作数

操作数字段中放的是操作码所要操作的数据,或者是被操作的数字所存储的单元的地址。操作数和操作码一起确定指令要执行的内容。

应该注意理解操作数的含意,它不仅仅是一个“数字”的概念,而是表示更为丰富的内含。例如:

LDA	# \$ 40;	将数字64装入累加器 A 中。
CMP	\$ 03F4;	比较 A 中的数与 \$ 03F4 单元中储存的数哪个大。
LDA	\$ 3F, X;	X 寄存器中的内容加上 \$ 3F
		再把该值指示的地址中的内容送入 A。

这里第一条指令的操作数 # \$ 40, 是所谓“立即数”,是一个数字(十进制数的64),不是地址;第二条指令中的操作数是一个地址,真正的操作数存放在 \$ 03F4 单元中;第三条指令中的操作数也是一个地址,不过这个地址必须通过 X 变址寄存器中的内容来变化,真正的操作数是在 (3F+X) 中。由此可见,计算机的指令中,表达操作数的方式是多样的,也很灵活,这种表达不同操作数的方式,就是我们将要介绍的寻址方式。

在汇编语言中,操作数可以使用标号、常数、表达式、字符或字符串,现分别说明如下:

1. 操作数是指定的标号

如前面介绍标号字段时所举的求累加和程序中, BNE SUM 这条指令中的操作数就是以标号 SUM 的形式出现的。又如下列指令:

```

LOOP1  LDY    # $ 64
        :
        JMP   LOOP1
  
```

在 JMP LOOP1 这条指令中,操作数是以标号 LOOP1 的形式出现的,它表示程序将转移到标号为 LOOP1 的地址去继续执行指令 LDY # \$ 64。

从这两个简单例子看出,用已有定义的标号作为操作数,在汇编语言中经常使用。此外,使用标号作为操作数,程序员如果使用带有汇编语言编译程序的计算机,就不必算出程序中存储单元的具体地址值。但如果使用手工编译,仍要人工计算存储单元地址。

2. 操作数是常数

所谓常数包括十进制常数、十六进制常数、八进制常数和字符串常数等四种。例如,前面的10个数据累加和程序中, LDA #0, CPX #10中的0和10都是十进制常数,在十进制数前面不加字符; STA \$0340中的0340是十六进制常数,它的前面必须加\$字符;字符串常数用跟在单引导之后的字符表示 ASCII 码字符,例如 LDA #'A',表示的是把字母 A 所对应的 ASCII 值 C1,送入累加器 A 中。当所表示的 ASCII 码字符不止一个时,仅对第一个字符有效,而在最后一个字符之后应再缀有单引号。如 LDA 'ABC'和 LDA 'A'是等效的。

3. 操作数以表达式出现

这时表达式由运算符+、-、*、/及>、<构成。例如:

```
LDA SECD-1      ;表示(SECD-1)→A
LDA #<10000     ;将10000的高字节→A
STA TICH        ;再送入 TICH 单元
LDA #>10000     ;将10000的低字节→A
STA TICL        ;再送入 TICL 单元
```

在上面的程序中,十进制数10000转换成十六进制数的2701,高、低字节分别为27和01。

对表达式进行运算的规则,是按自左至右的顺序进行计算。如5+COUNT-2,是先将5加上 COUNT 对值的高字节数,然后再减去2。

•注释

注释字段是一个可选字段,并不是每个语句都必须有的。注释只是为了阅读程序及编写文件方便,对指令或程序段所作的功能说明,用以了解程序当前工作或者转向等。注释必须以分号开始。其内容不属程序本身,因而不影响程序执行。

前面曾经提到汇编语句有两类,一类是指令语句,另一类是伪指令语句。这里选择常用伪指令的介绍如下。

伪指令是对汇编程序的一种命令,它在程序中执行一些特殊的操作。例如,它命令汇编程序在汇编时(即翻译汇编语言时),执行一些动作;为程序分配一定的存储区域;给符号赋值;将给定数据放入存储单元;留出存储数据用的存储区等等。伪指令构成的语句同样可以分为标号、操作码、操作数、注释四个字段。把用伪指令编写的程序输入到配有汇编程序的计算机中,汇编程序(编辑/汇编程序)与执行6502指令语言一样,同样执行伪指令所规定的各种操作。

•定义程序起始地址的伪指令 ORG

ORG,即 ORIGIN(起点)的缩写,用于规定一段程序或数据的起始地址,它的操作数字段表示的是存储单元地址值。例如:

```
ORG      $0300
START    PHA
        LSR    A
        :
```

由于使用了 ORG 的伪指令,上述 PHA 指令的代码将安排在 \$0300 单元,同时 START 标号也与 \$0300 等量。若在上述程序中加一句 ORG *+30,则将当前地址值加30。

•定义符号的伪指令 EQU

EQU 是 EQUATE(等值)的缩写。此语句用来给标号指定一个值。这个语句中必须有标号,并且不允许此标号再作其它语句的标号使用。例如:

```
DISPLAY  EQU  $FDF0
        :
        JSR  DISPLAY
        :
```

这个例子经过汇编则 DISPLAY 被赋值为 \$FDF0 这个地址值,而“JSR DISPLAY”便被汇编为“JSR \$FDF0”。

注意在写程序时,EQU 伪指令应放在程序的开始处。

•定义字节的伪指令 DFB

DFB 是 DEFINE BYTE(定义字节)的缩写。DFB 的功能是把字节或字节串存入从标号开始的连续单元中。例如:

```
ORG      $300
MNEML    DFB  $00,$11,$22,$33,
          $44,$55,'A','B'
```

伪指令 ORG \$300 规定了标号 MNEML 的地址值是 \$300,伪指令 DFB 指定字节串 00、11、22、33、44、55 以及字符 A、B 的 ASCII 码等八个字节,应该顺序地存放在标号 MNEML 开始(即从 \$300 开始)的连续单元中。其中第七、八两个地址中放的是字母 A、B 的 ASCII 码值。

•定义字的伪指令 DW

DW 是 DEFINE WORD(定义字)的缩写。此指令定义一个16位二进制数,即两个字节长度的数。其低8位在第一字节,高8位在第二字节。例如:

```
WORD DW $FBIC
```

此时,在 WORD 单元存入 \$1C,在 WORD+1 单元存入 \$FB。

伪指令还有不少,这里不再介绍。

六、汇编语言的特点

计算机的程序设计,从使用二进制数字表示的机器语言,发展到使用符号表示的汇编语言,之后,又发展到了使用接近人类语言的高级语言,是经历了很长时间的。程序设计语言的迅速发展和更新换代,极大地推动了计算机技术的普及和应用。高级语言受到广大青少年和科技工作者的欢迎,是因为对计算机本身的结构不必作详细的了解,而且可以减少程序设计时所下的功夫,同时它面向问题,面向使用者,简单易懂,易于学习和掌握,所有这些对于了解和掌握计算机的程序设计无疑是很有有效的。在学习了高级语言(例如 BA

(下转第42页)

初级程序员级水平考试辅导讲座

1992年计算机软件专业初级程序员级

技术资格考试复习自测题

计算机硬件基础知识

北京619信箱(100083) 顾育麒

自测题

从供选择的答案中选出正确答案,把相应的编号写在自我测试题的对应横线上。

1. 将十进制数125.8125转换为二进制数是_____,八进制数是_____,十六进制数是_____。

供选择的答案:

- (1)1011111.1101 (5)175.61 (9)7D.B
(2)1111101.1011 (6)175.64 (10)7D.13
(3)1111101.1101 (7)175.46 (11)D7.D
(4)111101.1101 (8)571.64 (12)7D.D

2. 将二进制数10110010.0011转换为十进制数是_____,八进制数是_____,十六进制数是_____。

是_____。

供选择的答案:

- (1)178.0625 (5)134.06 (9)B2.3
(2)178.1875 (6)544.14 (10)59.B
(3)176.125 (7)262.14 (11)B2.B
(4)178.375 (8)134.06 (12)32.3

3. 已知某微型计算机的字长是8位,则二进制数-1111111的原码表示是_____,反码表示是_____,补码表示是_____。

供选择的答案:

- (1)10000001 (2)11111111
(3)01111111 (4)10000000

4. 用补码表示的二进制数01110010的真值是_____,用补码表示的二进制数10110101的真值是_____(真值用十进制数表示)。

供选择的答案:

- (1)14 (2)114 (3)-114 (4)-14
(5)-53 (6)-75 (7)75 (8)53

5. 对于二进制数的代码10101101,若把它理解为无符号整数时,其对应的十进制数为_____;若把它理解为补码表示的有符号整数时,其对应的十进制数为_____。

供选择的答案:

- (1)-45 (2)173 (3)-83 (4)45

6. 某计算机采用定点整数格式,字长8位(包含一位符号位),当X采用原码表示时, $[X]_{\text{原}}$ 的最大正值是_____。

是_____,最小负数值是_____,当X采用补码表示时, $[X]_{\text{补}}$ 的最大正值是_____,最小负数值是_____,用十进制真值形式填写。

供选择的答案:

- (1)-1 (2)256 (3)255 (4)-127
(5)-128 (6)127 (7)-255 (8)128

7. 已知一逻辑表达式为 $F = A \overline{BC} + A \overline{BC} + ABC + A \overline{B}$,化简后可得到表达式_____。

供选择的答案:

- (1)AB (2)B (3)A (4)AC

8. 化简逻辑表达式: $AB + \overline{AC} + BC =$ _____。

供选择的答案:

- (1) $\overline{AC} + BC$ (2) $AB + \overline{AC}$
(3) $AB + BC$ (4)AB

9. 在CPU中,指令寄存器的作用是_____,程序计数器的作用是_____。

供选择的答案:

- (1)用来存放后继指令地址。
(2)保存当前正在执行的一条指令
(3)保存将被存储的下一数据字节的地址。
(4)保存当前CPU所访问的主存单元的地址。

10. 指令系统中采用多种不同寻址方式的主要目的是_____。

供选择的答案:

- (1)实现存储程序和程序控制。
(2)缩短指令长度,扩大寻址空间,提高编程的灵活性。
(3)可以直接访问外存储器。
(4)提供扩展操作码的可能性,降低指令译码的难度。

11. 变址寻址方式中,操作数的有效地址等于_____。

供选择的答案:

- (1)内存地址寄存器内容加上形式地址。
(2)数据寄存器内容加上形式地址。
(3)变址寄存器内容加上形式地址。
(4)指令寄存器内容加上形式地址。

12. 在间接寻址方式中,操作数处在_____。

供选择的答案:

- (1)通用寄存器 (2)主存单元
(3)程序计数器 (4)堆栈

13. 为了使微型计算机能正常工作,延长其使用寿命,机房的相对湿度应控制在_____范围内。

供选择的答案:

- (1)大于80%
(2)20%~80%
(3)小于20%

14. 计算机的技术性能指标 MIPS 的意义是_____。

供选择的答案:

- (1)每英寸扫描的线数。
(2)每秒钟能执行的百万条指令数。
(3)每秒钟能打印的字符数。

15. 计算机的技术性能指标 MTBF 的意义是_____。

供选择的答案:

- (1)磁带记录密度。
(2)平均无故障工作时间。
(3)机器翻译时间。

16. 半导体存储器在关机或停电后,_____芯片所存储的信息不会丢失,接通电源即可读出使用;而_____芯片,一旦断电,存储的信息全部丢失,接通电源后,其中已没有有用的信息。

供选择的答案:

- (1)RAM (2)ROM

17. 某微型计算机配备320KB/360KB 双面5.25英

寸软盘驱动器,能在这种软盘驱动器上读/写的软盘片有:_____和_____。

供选择的答案:

- (1)720KB 双面3.5英寸软盘片。
(2)160KB/180KB 单面,倍密度5.25英寸软盘片。
(3)320KB/360KB 双面,倍密度5.25英寸软盘片。
(4)1.2MB 双面,高密度5.25英寸软盘片。

18. 某磁带机,读写磁带时的走带速度为45IPS,磁带的记录密度为800BPI,则该磁带机的数据传送速度为_____。

供选择的答案:

- (1)32000BPS (2)36000BPS

19. 下列三种打印输出设备中,_____是击打式打印机;_____是非击打式打印机。

供选择的答案:

- (1)激光印字机。
(2)喷墨印字机。
(3)点阵针式打印机。

20. 计算机科技文章中,英文缩写 CAD 代表_____。

供选择的答案:

- (1)计算机辅助制造
(2)计算机辅助管理
(3)计算机辅助设计
(4)计算机辅助教学

自测题答案

题 号	答 案	题 号	答 案
1	(3)(6)(12)	11	(3)
2	(2)(7)(9)	12	(2)
3	(2)(4)(1)	13	(2)
4	(2)(6)	14	(2)
5	(2)(3)	15	(2)
6	(6)(4)(6)(5)	16	(2)(1)
7	(3)	17	(2)(3)
8	(2)	18	(2)
9	(2)(1)	19	(3)(1)(2)
10	(2)	20	(3)

自测题的解答与分析

[第1题]这道题是数制及其转换的自测题。数制间的转换,实质上是基数间的转换。如果两个

有理数相等。则两数的整数部分和小数部分一定分别相等。因此,进行各数制间的转换时,把整数部分和小

数部分分别按它的转换规律进行转换。

十进制整数转换为二进制整数,采用“除2取余”法。它的转换规律是:用2不断地去除要转换的十进制数,若余数为1,则相应位为1;若余数为0,则相应位为0,从低位到高位依次求得相应位的数字符号,直到商为0时为止。最后一次除法得到的余数,相应于最高位的数字,然后按从高位到低位的顺序写出这个转换后的二进制数。

类似地把十进制整转换成八进制整数、十六进制整数,只需要用“除8取余”法,“除16取余”法。

十进制小数转换为二进制小数,采用“乘2取整”法。它的转换规律是:用2反复去乘十进制纯小数的小数部分,每次乘上2之后,所得新数的整数部分为1,则相应位为1;整数部分为0,则相应位为0。从高位向低位依次进行,直到满足精度要求为止。最后一次乘积的整数部分为最低位的数字,然后按从高位到低位的顺序写出转换后的二进制小数。

类似地把十进制小数转换成八进制小数、十六进制小数,只需要用“乘8取整”法,“乘16取整”法。

这道题要求将十进制数125.8125转换为二进制数。

整数部分“除2取余”:

$125 \div 2 = 62$	余数	1	(低位)
$62 \div 2 = 31$	余数	0	
$31 \div 2 = 15$	余数	1	
$15 \div 2 = 7$	余数	1	
$7 \div 2 = 3$	余数	1	
$3 \div 2 = 1$	余数	1	
$1 \div 2 = 0$	余数	1	(高位)

$(125)_{10} = (1111101)_2$

小数部分“乘2取整”:

乘2	纯小数部分	整数部分
$0.8125 \times 2 = 1.6250$	0.625	1 (高位)
$0.625 \times 2 = 1.250$	0.25	1
$0.25 \times 2 = 0.5$	0.5	0
$0.5 \times 2 = 1.0$	0.0	1 (低位)

$(0.8125)_{10} = (0.1101)_2$

整数部分与小数部分用小数点连起来,得到:

$(125.8125)_{10} = (1111101.1101)_2$

在选择填空时,应选择答案(3)

在数制转换的过程中容易发生的错误:

1. 把高位到低位的数字顺序写错。供选择的答案(1),把整数部分高位与低位的数字顺序写反;供选择的答案(2),把小数部分高位与低位的数字顺序写反。这两个答案都是错误的。

2. “除2取余”还没有进行到商为0,就急于结束数制转换过程。供选择的答案(4)比其它三个答案少一个二进制数的高位数字,产生的原因是:计算到商为1时就错误地认为“除2取余”的过程已经结束,因此,在整数部分的高位少一个二进制数字。

这道题的第2个空格是要求将十进制数转换为八

进制数。

整数部分

“除8取余”:

$125 \div 8 = 15$	余数	5	(低位)
$15 \div 8 = 1$	余数	7	
$1 \div 8 = 0$	余数	1	(高位)

小数部分“乘8取整”:

$0.8125 \times 8 = 6.5$	整数部分	6	(高位)
$0.5 \times 8 = 4.0$	整数部分	4	(低位)

$(125.8125)_{10} = (175.64)_8$

这道题的第3个空格要求将十进制数转换为十六进制数。

整数部分“除16取余”:

$125 \div 16 = 7$	余数	13(写成D)	(低位)
$7 \div 16 = 0$	余数	7	(高位)

小数部分“乘16取整”:

$0.8125 \times 16 = 13.0$ 整数部分13(写成D)

$(125.8125)_{10} = (7D.D)_{16}$

当读者熟练地掌握了数制及其转换的方法以后,可以发现还有更加简单的方法。

十进制数125.8125转换成二进制数1111101.1101以后,可以利用二进制数直接转换成八进制数、十六进制数。

$2^3 = 8$,三位二进制数与一位八进制数相对应。二进制数转换为八进制数的方法:以小数点为界,小数点左边的整数部分自右至左每三位二进制数分为一组,不足三位时左边补0;小数点右边的小数部分自左至右每三位二进制数分为一组,不足三位时右边补0,然后把每一组二进制数用相应的八进制数表示。

二进制数	001	111	101	110	100
	↓	↓	↓	↓	↓
八进制数	1	7	5	6	4

$2^4 = 16$,四位二进制数与一位十六进制数相对应。二进制数转换为十六进制数的方法:以小数点为界,小数点左边的整数部分自右至左每四位二进制数分为一组,不足四位时左边补0;小数点右边的小数部分自左至右每四位二进制数分为一组,不足四位时右边补0,然后把每一组二进制数用相应的十六进制数表示。

二进制数	0111	1101	1101
	↓	↓	↓
十六进制数	7	D	D

从前面的计算中,还可以看到:十进制数转换成八进制数的计算过程短,转换成十六进制数的计算过程更短。因此可以先将十进制数转换成八进制数,再利用八进制数与二进制数的对应关系转换成二进制数,这样更节约时间。读者可以根据自己的熟练程度,灵活运用以上的几种转换方法。

[第2题]二进制数转换为十进制数的方法有两种:

1. 按权相加法:把每一位的权(2的某次幂)与数位值(0或1)的乘积项相加,所得到的和,就是相应的十进制数。

$(10110010.0011)_2 = 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4}$

$$\begin{aligned}
& 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + \\
& 0 \times 2^0 + 0 \times 2^{-1} + 0 \times 2^{-2} + 1 \\
& \times 2^{-3} + 1 \times 2^{-4} \\
& = 128 + 32 + 16 + 2 + 0.125 + \\
& 0.0625 \\
& = (178.1875)_{10}
\end{aligned}$$

2. 对应关系法：二进制数每一位的权与十进制数有一定的对应关系。例如小数点左边第2位的权，对应的十进制数为2；第4位的权，对应的十进制数为8。将二进制数中数字为1的位的权，所对应的十进制数值相加，其和就是相应的十进制数。

$$(10110010.0011)_2 = 128 + 32 + 16 + 2 + 0.125 + 0.0625 = (178.1875)_{10}$$

二进制数转换为八进制数、十六进制数的过程如下：

二进制数	010	110	010	001	100
↓	↓	↓	↓	↓	↓
八进制数	2	6	2	1	4
二进制数	1011	0010	0011		
↓	↓	↓	↓		
十六进制数	B	2	3		

[第3题] 前面提到的二进制数，没有提到符号问题，是一种无符号的二进制数。但是在计算机中运算的数，是有正负号的。人们规定用“0”表示正数符号，用“1”表示负数符号，而且规定一个数的二进制编码的最高位为符号位。这样，连同符号位在内的若干位二进制

数作为一个数，称为机器数。它是数在机器中的表示形式。它所代表的数值称为机器数的真值。

为了运算方便，在机器中负数有三种表示法——原码、反码和补码。采用补码以后，可以使正、负数的加、减运算简化为单纯的加法运算。

原码表示法：

正数的符号位用“0”表示，负数的符号位用“1”表示，数值位保持不变。

$$[-1111111]_{\text{原}} = 11111111$$

反码表示法：

正数的反码与原码相同，最高位为符号位，用“0”表示正，其余位为数值位。

负数的反码表示，即为它对应的正数按位取反（连同符号位）而形成的。

$$\begin{aligned}
& \text{二进制数} && -1111111 \\
& \downarrow && \downarrow \\
& \text{对应的正数} && 01111111 \\
& \downarrow && \downarrow \\
& \text{按位取反} && 10000000 \\
& [-1111111]_{\text{反}} = 10000000
\end{aligned}$$

补码表示法：

正数的补码与原码相同，最高位为符号位，用“0”表示正，其余位为数值位。

负数的补码表示，即为在它的反码的末位加1（简称“求反加1”）

$$[-1111111]_{\text{补}} = 10000001$$

《全国中小学计算机教育资料汇编》征订启事

为总结我国“七五”期间中小学计算机教育方面的经验，向广大教师、中小学校及有关教育管理部门提供计算机教育方面的信息，以促进计算机教育事业的发展，“全国中学计算机教研中心”和电子工业出版社编辑出版了《全国中小学计算机教育资料汇编》，内容包括：

1. 有关部委及省、市领导部门关于计算机教育方面的文件；
2. 中央领导同志的题词与讲话；
3. 国内外计算机教育工作会议纪要及学术活动；
4. 教师的优秀论文及研究报告；
5. 中学计算机课教学大纲及有关规定；
6. 全国中学计算机配置及教师队伍情况的调查；
7. 全国及各省青少年计算机竞赛与竞赛题，国际信息学奥林匹克竞赛题；
8. 全国教育软件评审机构、评审标准、优秀软件简

介；

9. 开展计算机教学的中小学名单，科技馆、青少年宫名单；

10. 有关计算机普及教育的管理部门及学会、协会等机构通讯录；

11. 全国出版的青少年计算机类图书、报刊目录。

本《汇编》是我国计算机教育方面的一本较全面的资料，可供中学师生、计算机教研工作人员、中小学校领导及各级教育管理部门参考。

本《汇编》16开本，75万字，定价13.5元，已于1991年8月出版。

（注：此书邮购需另加15%的邮费）

地址：北京市万寿路173信箱 电子工业出版社软件部

邮 码：100036 电 话：815342

联系人：谈众安 王 旭



8031单片机最小系统

北京宣武区青少年科技馆(100052) 罗明宽 车金相

我们利用 APPLE II 和 LASER310 研制出一套开发装置—EPROM 仿真器。利用这个仿真器开发出了 8031 单片机的最小系统。

一、EPROM 仿真器的工作原理

下面我们对 EPROM 仿真器的电路(见图1)做一下简单说明:该仿真器有两个系统,其中一个系统是 APPLE 机,该系统可以通过 IC₂、IC₃、IC₄、IC₅、IC₆、IC₇ 等芯片对 2K RAM 6116 进行读写操作,另一个为被开发系统,即 8031 单片机系统,这个系统通过 EPROM 插座 IC₂、IC₃、IC₄、IC₅、IC₆ 等芯片也能对 6116 进行读操作,可见 2K RAM 6116 为两个系统公共存储器。IC₂、IC₃、IC₄、IC₅ 为 74LS157,为二选一芯片,它的作用是对两个系统的地址线进行切换,由 1 脚进行控制,IC₇、IC₈ 为双向数据缓冲器,利用各自的 19 脚的低电平进行数据传送和封堵对方系统的数据传送,各自的 1 脚控制数据传送的方向,IC₆ 为 74LS138 译码器,它的作用是对 2K RAM

6116 提供片选信号和为 IC₇、IC₈、IC₂~IC₅ 提供控制信号。

当在 APPLE 机上对 RAM 6116 进行读写操作时, IC₆ 地址译码器 74LS138 的输出端 12 脚(Y₃)为 6116 提供片选信号,通过计算,6116 的地址范围是 C800~CFFF 2K 存储区,当我们在监控系统下,对 C800~CFFF 这段存储区进行读写操作时, IC₆ 的输出端 12 脚输出“L”电平,这个信号被接到:

1. IC₇ 的 19 脚上,该片处于工作状态,即三态门的闸门被打开。

2. IC_{9b} 的一个输入端上,另一输入端不论何种状态,其输出端为高电平,这个高电平又接到 IC₈ 的 19 脚上,此芯片停止工作,这就保证在 APPLE 机对 2K RAM 6116 进行读写操作时,数据不能流入被开发系统,被开发系统的数据也不能流入 APPLE 机,起到两个系统之间数据的封堵,防止系统之间的混乱。

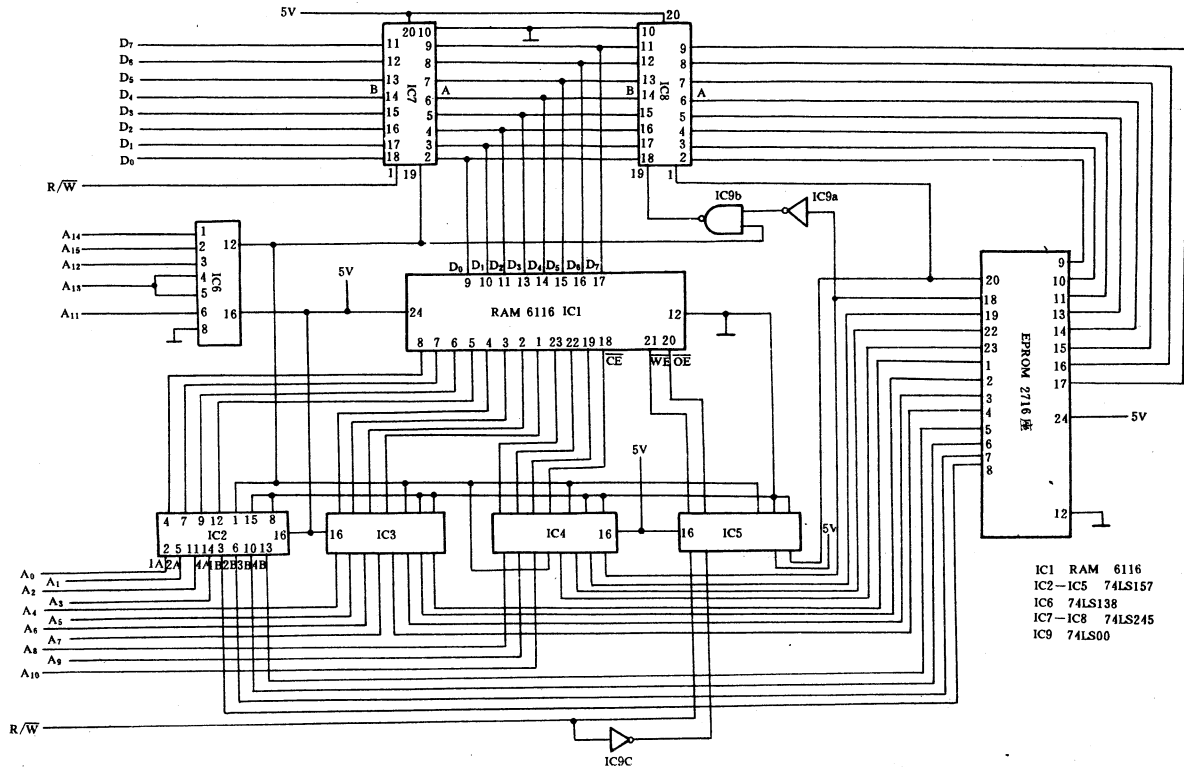


图1

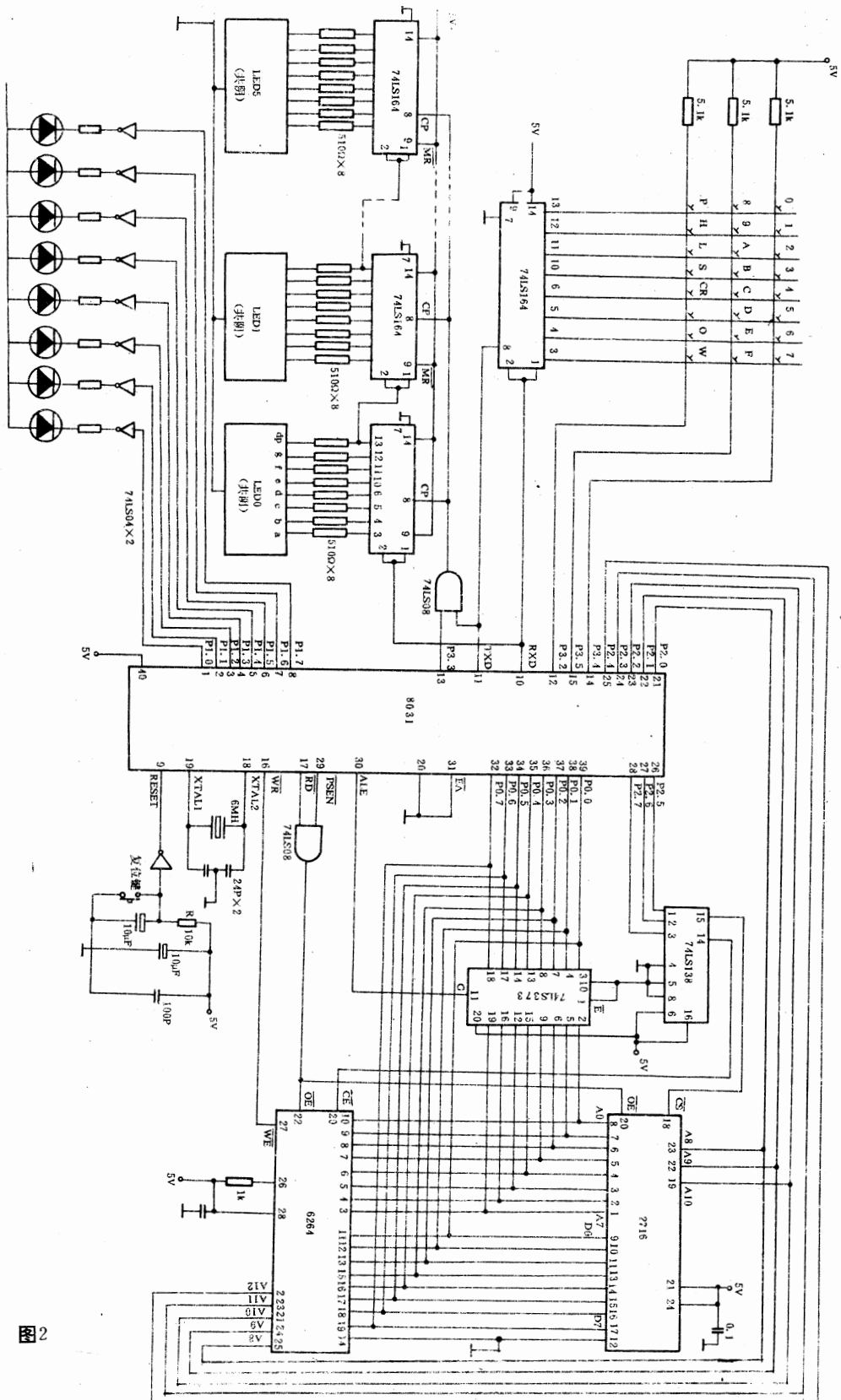


图2

3. IC₂~IC₅四个芯片所有的1脚上,则输出端按 A 口的输入状态进行输出,(A 口被接到 APPLE 机上)B 口状态不能被输出,(B 口被接到被开发系统上)这就达到被开发系统的地址线和控制线被封堵。

4. IC₄的14脚上,其对应输出端4y,将这个低电平信号传输到 RAM 6116的18脚的片选端上,只要对 C800~CFFF 这2K 存储区进行读写操作时,该片被选中。

当 CPU 读时, $R/\overline{W}=H$ (高电平)这个信号和这个信号通过反相器分别接到 IC₅的1A 和2A 上,其对应输出端1y,2y 又分别与 RAM 6116的21脚的 \overline{WR} (写控制端)和20脚 \overline{OE} (读控制端)相接,此时 $\overline{WR}=H$ 、 $\overline{OE}=L$,则 RAM 6116芯片处于被读状态,同时, $R/\overline{W}=H$ 这个电平又传输到 IC₇的1脚上,此时1脚为高电平,根据 74LS245这个芯片的特性,数据的流向由 A 到 B,即 CPU 将 RAM 中单元里的数据读回 CPU。

当 CPU 进行写入操作时, $R/\overline{W}=0$,即读写线为“L”电平。而这个电平和这个电平通过反相器又分别通过 IC₅传输到6116的21脚和20脚上,此时这两个脚的电平分别为“L”和“H”。此时该芯片为写入状态,同时 $R/\overline{W}=\text{“L”}$ 的信号又接到 IC₇的1脚上,则数据的流向为 B 到 A,即 CPU 将数据写入 RAM 6116存储器的单元之中。

当被开发系统(单片机)通过 EPROM 2716插座对 RAM 6116进行读操作时,IC₆的地址译码器的输出端12脚为 H 电平,这个电平被接到:

①IC₇的19脚上,此芯片停止工作,即内部闸门被关闭,不能传输数据,同样起到两个系统之间的数据被封堵,防止两个系统之间的混乱。

②IC₉的一个输入端上,而被开发系统的 EPROM 的片选信号,通过扁平电缆线传输到仿真器2716插座的18脚上,这个片选信号(L 电平)一路接到非门(IC_{9a})的输入端上,其输出端接到 IC₉的另一个输入端上,IC₉的两个输入端均为高电平则输出端为“L”电平,使 IC₆的19脚也变为 L 电平,该片处于工作状态,即内部的三态门被打开,另一路被接到 IC₄的4B 端口上,其对应输出端4y 接到6116的18脚的片选上,保证被开发系统读时该片被选中。

③IC₂~IC₅的四个芯片所有1脚上,均为高电平,则选择 B 口输出,A 口停止,APPLE 机的地址线和控制线被封堵。

当被开发系统“读”操作时,读信号传输到 EPROM 2716插座上(20脚),一路输入到 IC₆的1脚上,此时为“L”电平,则数据由 B 流向 A,即6116中的单元的数据通过数据总线读回被开发系统 CPU 中,另一路通过 IC₅的2B 的输入端,使其输出端2y 接到 RAM 6116的20脚读控制端上,而 IC₅的另一输入端1B 接到+5V 上,其输出端1y 接到6116的21脚写控制端上,此时 $\overline{OE}=0$ 、 $\overline{WR}=1$,则该芯片处于读的状态。这样被开发系统将 RAM 6116单元中的数据读回 CPU。

二、硬件电路几点说明

1. 该系统利用 8031 内部的振荡电路,在 XTAL1、XTAL2 引脚上外接定时元件,内部振荡电路便产生自激振荡,形成单片机的时钟电路。晶振与电容的参数如图2所示。

2. 由上电复位和按键复位结合电路为 8031 提供复位信号(RESET),CPU 便从 0000H 地址开始执行程序。

3. 地址总线由 P0 口提供低八位(A₀~A₇)地址线,P2 口提供高八位(A₈~A₁₅)地址线,由于 P0 口还要作数据总线口,因此只能分时用作地址线,故 P0 口输出低八位地址时必须用锁存器锁存在该系统中用 74LS373 八 D 锁存器为 EPROM 2716 和 RAM 6264 提供低八位地址线,锁存信号由 8031 的 ALE 管脚提供。

4. 74LS138 为地址译码器,为 EPROM 2716 和 RAM 6264 提供片选信号,其地址分别为 0000H~07FF 和 2000H~3FFF,其中 2716 存放着管理程序,6264 为用户程序区和数据区。

5. 将外部程序存储器的选通信号线 \overline{PSEN} 和外部数据存储器选能信号线 \overline{RD} 相与,其输出端与 6264 \overline{OE} 端相连,这样 6264 既可做程序存储器和数据存储器用,为用户的使用带来方便。

6. 本系统我们选用了 MCS-51 串行口(设定串行口工作在移位寄存器方式 0 状态下)外接 74LS164 构成了键盘和显示器的控制电路,其中外接 6 片 74LS164 作为 6 位静态显示口,另外接一片 74LS164 作为键盘列线扫描输出口,通过实验静态显示亮度大,键输入稳定。主程序可不必扫描显示器,从而有更多的时间处理其它事情。

三、监控程序的几点说明

1. 本监控程序只编写了最基本的三大功能,其它功能留用户在学习本系统后根据需要自行开发,本系统显示缓冲区占 68H~6DH 单元,键盘缓冲区占用 58H~5EH 单元,用户不能占用。

2. 三大功能介绍如下:

①H 命令

作用:将程序写入存储单元。

格式:H×××× \swarrow H 后是存储器的四位地址,“ \swarrow ”为回车。数码管显示××××四位地址,然后键入数据再回车。

例如:从 2000 开始输入一段点亮 P1 口的小灯程序。

操作:键入 H2000 \swarrow

显示 2000

键入 74 \swarrow

显示 2001

键入 FF \swarrow

显示 2002

键入 F5 \swarrow

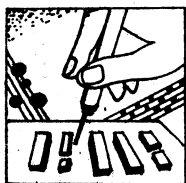
显示 2003

键入 90 \swarrow

显示 2004

键入 02 \swarrow

(转第 22 页)



学装微电脑

附带应急开关的顺序控制

易齐干

顺序控制过程中,为避免异常状态的发生,要不断地监视工作过程。如果发生应急事态,就要暂时中断正常工作,进行应急处理。本文讨论应急处理时硬件安排与软件的设计。

对应急处理的要求:应急开关为 ON,LED 灯熄;蜂鸣器蜂鸣。应急开关为 OFF,LED 灯恢复灯熄前的状态,蜂鸣器停止蜂鸣。

针对上述要求,我们可将它编制为子程序,在执行程序过程中,应急开关为 ON,或为 OFF,均执行预先安排的子程序。这在程序设计上非常方便。满足这种要求的功能称为中断。

μ p-80 套件中 CPU Z80 的 16[#] 出脚由 H 变为 L 电平,CPU 则产生中断,中断正执行的主程序,转而去执行 0038H 地址为开始的中断子程序。使用中断必须注意下述三点。

1. 为使 Z80 CPU 能接收中断程序,在程序开始时,必须添加“开中断”(EI 指令)和“1 型中断”(IM1 指

令)语句。

2、0038H 为中断程序开始地址,主程序的地址要避开中断程序地址。

3、CPU 执行完中断程序,要返回原主程序,则自动关闭中断。为能实现多次中断,应该在中断程序的最后,添加“开中断”(EI 指令)语句。

能正确满足上述要求的硬件安排如图1所示。输入输出部件的 A 口高位连接打码开关和 H/L 电平验证部件, S_0 打码开关为应急开关, S_0 为 ON,A 口第4位由 1 \rightarrow 0,同时 CPU 谋求中断。中断之后,CPU 查询 S_0 打码开关的 ON、OFF 状态,如果为 ON,LED 灯熄;蜂鸣器蜂鸣。如果为 OFF,LED 灯返回到熄灯以前的状态,蜂鸣器停止蜂鸣。这就是符合要求的应急开关处理。流程图与程序清单如图2所示。

应急开关的处理是在顺序控制过程中进行。顺序控制的流程图如图3所示。

0038H 地址开始的中断程序中,用 PUSH 指令

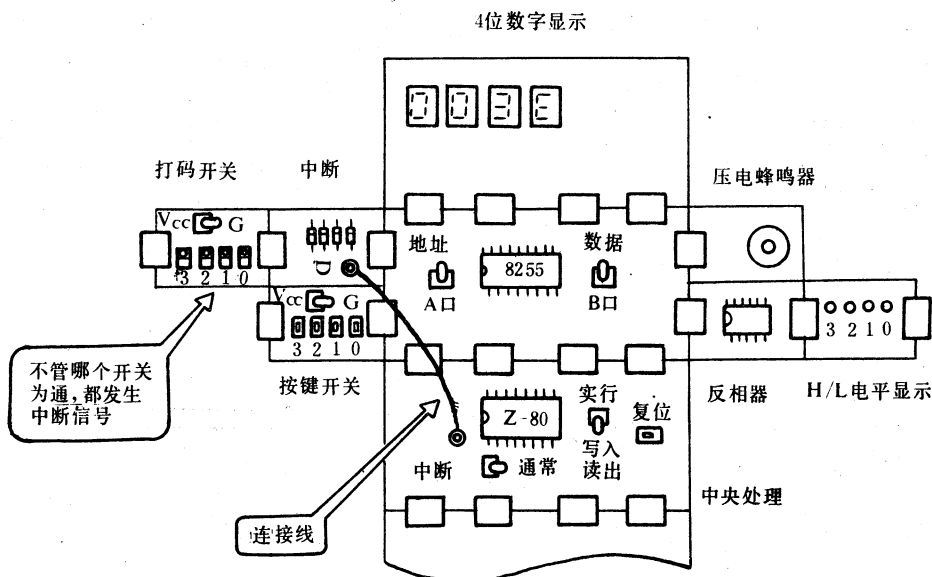


图1

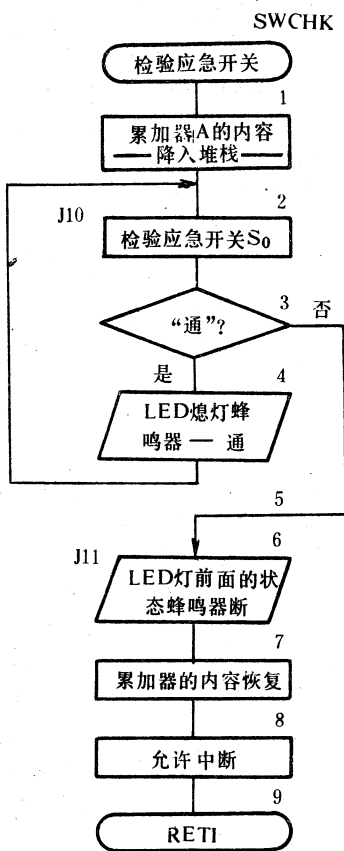
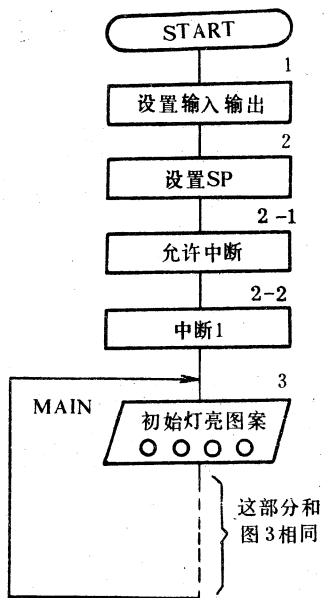
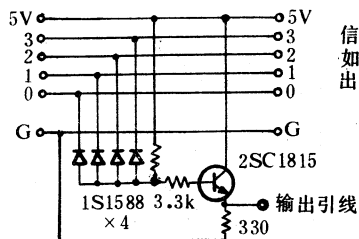
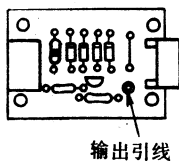


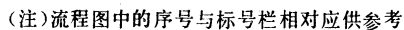
图2

标号		助记符	地址	机器语言
START	1	LD A,90H	0000	3E 90
		OUT(03H),A	0002	D3 03
	2	LD SP,0100H	0004	31 00 01
		JP MAIN	0007	C3 60 00
MAIN	3	LD HL,00F0H	0060	21 F0 00
		LD (HL),00H	0063	36 00
		LD A,(HL)	0065	7E
		OUT(02H),A	0066	D3 02
	J1 4	IN A,(00H)	0068	DB 00
		BIT 0,A	006A	CB 47
	5	JP NZ,J1	006C	C2 68 00
	6	LD HL,00F0H	006F	21 F0 00
		LD (HL),06H	0072	36 06
		LD A,(HL)	0074	7E
		OUT(02H),A	0075	D3 02
	J2 7	IN A,(00H)	0077	DB 00
		BIT 1,A	0079	CB 4F
	8	JP NZ,J2	007B	C2 77 00
	9	LD HL,00F0H	007E	21 F0 00
J3		LD (HL),09H	0081	36 09
		LD A,(HL)	0083	7E
		OUT(02H),A	0084	D3 02
	10	IN A,(00H)	0086	DB 00
		BIT 2,A	0088	CB 57
	11	JP NZ,J3	008A	C2 86 00
	12	LD HL,00F0H	008D	21 F0 00
		LD (HL),05H	0090	36 05
		LD A,(HL)	0092	7E
		OUT(02H),A	0093	D3 02
	13	IN A,(00H)	0095	DB 00
		BIT 3,A	0097	CB 5F
	14	JP NZ,J4	0099	C2 95 00
	15	JP MAIN	009C	C3 60 00



信息线哪条
如果为0,则输
出引线也为0。

图4



将累加器 A 的内容推入堆栈,最后,用 POP 指令再将内容退回累加器 A。这样应急开关不会破坏主程序中开关输入的 ON、OFF 状态。

执行上述程序,既能对按键开关 $S_0 \sim S_3$ 进行顺序控制,又能查询应急开关 S_0 的状态。

如果安排两个以上应急开关,可考虑如图4所示的部件。 $0^{\circ} \sim 3^{\circ}$ 任意一条信息线由 ON \rightarrow OFF 时,均产生中断信号。按照图5接排 $\mu\text{p}-80$ 套件,则具备4种应急开关处理功能。任何一个打码开关为 ON, CPU 均执行 0038H 地址开始的中断程序。

本实验从另一个侧面告诉我们,硬件设计得好,可以减少软件负担。读者应该努力掌握硬件、软件两方面的知识,以达到平衡。



多功能程序移植器

——一种简易的硬件工具

北京西城区青少年科技站 朱立钢

1. 电路功能

我们经常希望把电子游戏卡或其它一些 ROM 片中的程序,复制到自己的可编程 ROM 片或 RAM 片中,进行研究和利用。而在单片机应用开发过程中,要把自己在 RAM 中编好的应用程序及简单监控程序写入 ROM,这样再投入实际应用,才不至于每次断电程序都会丢失,也可以省去软盘驱动器,实现廉价和无需专业人员操作的完全自动化。虽然我们对存储芯片已不再陌生,ROM、RAM 天天都在利用,然而这些存储芯片都在计算机的内部,对其读取都是在系统软件控制下,并通过 CPU 进行,而 ROM 的写入,目前通用的方法用专门的计算机仿真系统 MDS(如 ICE-48/49),或制做一些特殊接口配合计算机,接上存储器再编制专用程序完成内存复制(如国内的一些简易系统)。这些方法的价格都在千元以上,而且要求使用者有一定的软硬件知识,使一般的爱好者难以完成,使廉价的单片机开发受到限制。下面介绍的电路可以实现:把源存储器中任意始末地址的内容写入目标存储器。这个电路由于无需使用 CPU 和键盘,可以对存储器独立进行复

制,因而成本很低而且不受机型的限制,应用广泛灵活,设计使用 74LS 系列通用元件,成本只有十几元。即使无很多硬件和专门软件知识的人,也能操作,按下启动按钮,复制自动完成,简便迅速,应用范围广,地址总线数可变,可根据你的需要确定,所以很适合业余爱好者自制。

2. 电路原理

图1为电路设计思路: D11与 D12组成一个标准 RS 触发器,控制电路动态静态的切换。D13, D14, D21组成可控振荡器,输出时钟信号。D23, D24组成单脉冲发生器产生写入脉冲,并使此脉冲在两次地址加1之间的时刻产生,控制程序的写入与末地址的判断。IC1, IC2(四位二进制同步计数器)组成地址发生器。D42, D41, D32, D33, $K_n - K_0$ 与 $K_n' - K_0'$ 组成末地址检测器,当到达末地址时,产生一个停机信号到自停电路。

工作时首先把 K 拨到预置位,拨动 $K_n' \sim K_0'$ 与 $K_n \sim K_0$ 预置始末地址,然后按一下启动按钮, A 点置成高电平,可控振荡器 D13 门打开,脉冲可以通过, C 点立即输出时钟信号,动态指示灯亮。C 点的时钟信号进入各计数器,此时 B 点为低,各计数器置位,初地址被置入。再看 D22, 它在 CP 信号与单脉冲之间,设计它的目的有两个:一是在置位时 D22 的 4 脚为 0, CP 信号无法通过,不发出写入脉冲。这可以防止在置位地址的重复写入,因为 ROM 的写入次数是有限的;二是它使写入脉冲正好在两次地址加一之间,否则地址变化时的瞬时杂信号会影响工作。K 拨至存入,再看 C 点, CP 上升沿时计数器触发地址加1,在 CP 下降沿时经 D22 变为上升沿触发单脉冲发生器产生一个大于 100 微秒的写入脉冲,原存储器的一位数据自动写入目标存储器,然后 CP 上升沿又使地址加1,下降沿再写入一位。两部分如此交替工作直到地址输出与预置的末地址相同时, E 点输出低脉冲, RS 触发器翻转, A 点变为低电平,振荡器停振,动态指示灯熄灭,这时复制就完成了。

3. 使用举例

这里举一个把一片 ROM 中的程序写入一片 RAM 的例子来具体说明程序移植器的使用。第一步,打开电源,动态指示灯可能是亮着的,这时要等几秒钟它会自动熄灭。这时进行第二步, K 拨至置入,置入要写入程

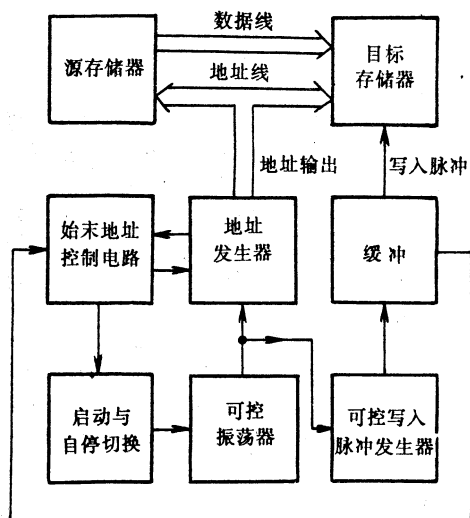


图1

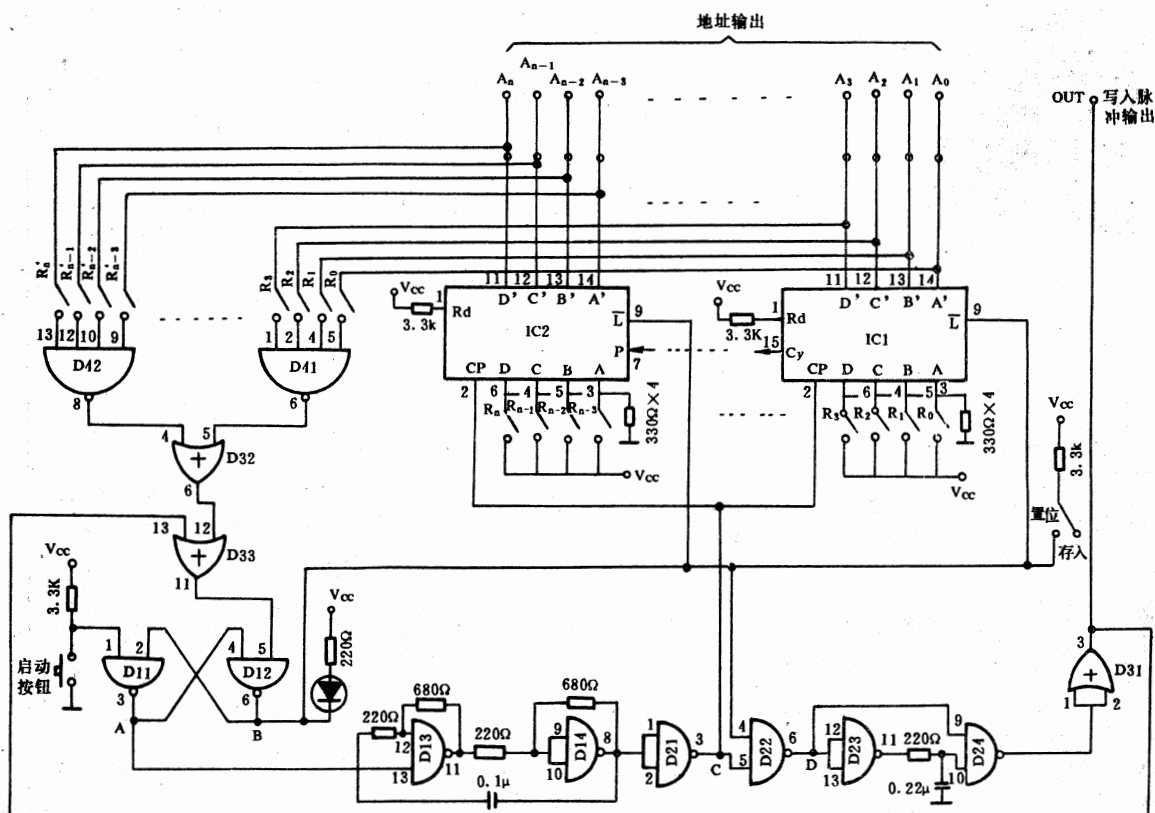
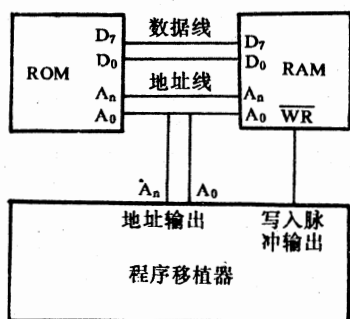


图2



WR为写入允许端

图3

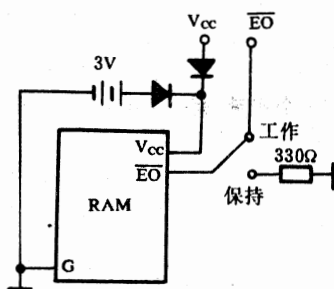


图4

序的始、末地址。第三步,连接源存储器与目标存储器如图3。第四步,按一下启动开关,动态指示灯亮,K拨至写入,经过几秒钟后,复制完成,动态指示灯熄灭。

4. 制作和使用注意事项

1. 试验电路工作可在输出逐位检验,并用手动给定C点信号,检查地址变化。
2. 如果把此电路接入你的计算机时,要注意地址线由移植器占用时其它地址线上的输出单元都要断开(CPU要中断),即高阻输出,数据线也是这样,不能同时占用总线。
3. 如果只需全部地址的移植,可以省去n位拨码开关 $K_n \sim K_0, K_n' \sim K_0'$ 用导线代替,但置位按钮不可省。

4. 为实验方便,介绍一个RAM作ROM使用的方法,见图4,开机工作时电池不耗电,关机前把开关拨至保持,这时电流很小,如我使用的5101型CMOS RAM只有20微安(工作时40毫安)两节5号电池可使用半年。

5. 元件选择

74LS00 $\times 2$; 74LS32 $\times 1$

74LS161(或139) $\times 2$; 74LS20 $\times 1$;

其它元件无特殊要求。



第二章 F BASIC 的基本语句

山东苍山机械电子化学工业局(277700) 于 春

三、数值函数语句。

1、绝对值函数

ABS 简写 AB.

该函数的功能是将数值换成绝对值。如

P. AB. (-41) 回车后,显示41

2、符号判别函数

SGN 简写 SG.

SGN 用来判别数的性质,当 X 为正数时 SG. (X) = 1; X 为负数时 SG. (X) = -1; X 为零时 SG. (X) = 0。

3、随机数函数

RND 简写 RN.

RND 语句可使计算机产生随机数,其格式为 RN. (X), X 的取值范围为 1~32767,用它可产生小于 X 的随机数。

为指导小学生进行数学练习,介绍两个实用程序,读者可以举一反三。

例14 一位数乘法练习程序

10 A=RN. (10):B=RN. (10)

20 P. A " * " B " = " ;

30 I. C

40 IF C=A * B T. P. "V":G. 10

50 P. "X":G. 20

RUN 后,显示一位数乘法算式,输入得数后,若正确显示"V",给出另一算式;若错则显示"X",重复打印算式。

例15 两位数加法练习程序

10 A=RN. (100):B=RN. (100)

20 P. A " + " B " = " ;

30 I. C

40 IF C=A+B T. P. "V":G. 10

50 P. "X":G. 20

读者可以发现,两个程序格式相同,不同的是更换了随机数上限和运算符号。因此,读者可照此编写其它位数的乘、加、减练习程序。

四、数据语句

数句语句包括读数语句(READ)、置数语句(DATA)、恢复数据区语句(RESTORE)、存数语句(POKE)和取数语句(PEEK)。

1、READ 语句和 DATA 语句

READ 简写 REA.

DATA 简写 D.

当有许多变量需要赋值,用 LET 语句就显得很繁琐,这时可采用 READ 和 DATA 语句向变量提供数据:

10 REA. A,B,C,D,E,F,G

20 D. 3,4,5,6,7,8,9

当执行 READ 语句时,变量的值依次从 DATA 语句中读出,即 A 读3,B 读4……

可以发现,当变量赋值较多时,用 READ、DATA 语句比较简单。

在使用 READ、DATA 语句时要注意两点:

(1)由于 DATA 语句是非执行语句,所以它可以放在程序中的任何位置。但若有多条 DATA 语句,它们之间必须有先后顺序。另外 DATA 提供数据的个数,不能少于 READ 中的变量数。如:

10 D. 3,4,5,6

20 REA. A,B,C

30 P. A,B,C

与

10 REA. A,B,C

20 P. A,B,C

30 D. 3,4

40 D. 5,6

两程序效果是一样的。

(2)DATA 语句中的数据可以是数值、字符串,但不能是变量、函数或表达式。DATA 语句中的数据类型应严格与 READ 语句中的变量类型一一对应。如:

10 REA. A\$,AB,CDE

20 D. "Zhong Guo",1991,8,"Computer"

READ 中有三个变量。第一个是串变量,后两个是实变量。DATA 中有四个数据,一、四为字符串,二、三为实数。两语句中一、二、三分别对应,所以程序正确。至于 DATA 中多了一个串变量"Computer"则没有关系,不影响程序的运行。

我们学习了读数、置数语句后,例13分类统计学生成绩的程序中可不使用 INPUT 语句,因每输入一个数要回车一次,也嫌麻烦。可修改程序如下,改20、120两

行:

```
20 REA. X
120 D. (学生成绩单), 110
```

读者可自己练习。

2、恢复数据区语句

RESTORE 简写 RES.

如果要3~9七个自然数分别送给变量 A~G、H~N、O~U,一般程序为:

```
10 REA. A,B,C,D,E,F,G
20 REA. H,I,J,K,L,M,N
30 REA. O,P,Q,R,S,T,U
:
100 D. 3,4,5,6,7,8,9
110 D. 3,4,5,6,7,8,9
120 D. 3,4,5,6,7,8,9
```

大家可以看到,三行 DATA 语句是重复的。为节省内存,简化程序,可使用 RESTORE 语句。当程序执行到 RESTORE 时,便使下一个 READ 语句需要的数据又从最小行号的 DATA 语句的第一个数据读起。于是上面的程序可改写为:

```
10 REA. A,B,C,D,E,F,G
20 RES. :REA. H,I,J,K,L,M,N
30 RES. :REA. O,P,Q,R,S,T,U
:
100 D. 3,4,5,6,7,8,9
```

3、三种提供数据语句的比较

到此为止,我们已经介绍了 LET、INPUT、READ/DATA 三种语句。这三种语句都可以给变量赋值,同一个问题可分别用三种不同的方法编写程序。现举一例说明。

例16 鸡兔同笼,已知鸡兔总头数为 T(Tou),总脚数为 J(Jiao),求鸡兔各有多少只?

先用代数求出鸡兔个数的数学表达式。设鸡 X 只,兔 Y 只,则有

$$X = (4 * T - J) / 2$$

$$Y = (J - 2 * T) / 2$$

今设 T=16, J=40, 编程如下:

程序 I: 用 LET 语句

```
10 T=16:J=40
20 X=(4*T-J)/2:Y=(J-2*T)/2
30 P. "JI="X, "TU="Y
```

程序 II: 用 INPUT 语句

```
10 I. "T,J", T, J
20 X=(4*T-J)/2:Y=(J-2*T)/2
30 P. "JI="X, "TU="Y
```

程序 III: 用 READ/DATA 语句

```
10 REA. T, J
20 X=(4*T-J)/2:Y=(J-2*T)/2
30 P. "JI="X, "TU="Y
40 D. 16, 40
```

由上例可见,三种语句的相同之处是都能提供原

始数据,但使用特点各有不同。LET 语句适合于赋值变量少,且编程时数据已确定的场合。另外赋值语句可用来运算。这一功能,其它两种无法取代。如果数据很多,且编程时已确定,使用 READ/DATA 语句比较方便;如参数是变化的,要随时输入变化了的数据,则宜于采用 INPUT 语句,如商业柜台,银行付息的计算等。三种语句掌握熟练后,定会给编程带来极大的方便,希望读者多加练习。

4、POKE 和 PEEK 语句

POKE 简写 PO.

PEEK 简写 PE.

POKE 和 PEEK 是两个互为逆操作的语句,POKE 的作用是把数存到指定的内存单元;PEEK 的作用是把指定内存单元的数取出来。这两个语句,随机手册中已有举例,本文不再列举。由于这两个语句在 F BASIC 的程序设计中使用不多,读者仅仅了解就行了。

五、数组及数组说明语句(DIM)

在第一章中我们介绍了简单变量。现在介绍第二种变量——下标变量。

1、下标变量

顾名思义,下标变量就是带有下标的变量。如 A(K)、B(I)等。使用下标变量的目的,是为了使一些性质相近的变量便于归类。如就 A(K)而言, K=1 时, A(K) 就是 A(1); K=2 时, A(K) 就是 A(2)……。

同一个下标变量名可以包含有两个下标,其形式为 A(I,J)。一般称一个下标的为单下标变量;两个下标的为双下标变量。

2、数组:

由同一个变量名组成的下标变量的集合叫数组。在实际应用中,人们习惯把一些具有相同性质的数据放在一个数组内,用不同的下标去区分。

由单下标变量组成的数组称一维数组;由双下标变量组成的称二维数组。当数组进入电脑后,每个下标变量占用若干连续的内存单元。因此,一个数组要占用内存中连续的一片单元。

3、数组说明语句(DIM 语句)

用一个数组说明语句可以定义一个数组或几个数组,它规定数组的名称及变量个数。如

```
10 DIM A(8)
```

此语句定义了一个实型数组,变量名为 A,下标变量的个数有 9 个(下标为 0~9),最大下标为 9。对于 A(m) 数组,变量个数为 m+1 个,对于 A(m,n) 数组,变量个数为 (m+1) × (n+1) 个。DIM 语句是非执行语句,一般放在程序的开头。

例17 从 DATA 语句中读出 10 个数据存于 A 数组中,然后将数据次序颠倒后再存于 B 数组中。程序如下:

```
10 DATA 3,4,5,6,7,8,9,10,11,12
20 DIM A(9),B(9)
30 F.I=0 TO 9:REA. A(I):N.
40 RES.
```

```
50 F.L=9 TO 0 ST. -1;REA. B(L):N.
```

```
60 F.M=0 TO 9:P. A(M),B(M):N.
```

例18 有10个杂乱无章的数,要求编一程序把这些数按大小顺序排列出来。

```
10 DIM A(9)
```

```
20 D. 38,99,26,78,32,100,27,66,95,46
```

```
30 F.K=0 TO 9:REA. A(K):N.
```

```
40 F.L=0 TO 8
```

```
50 F.N=0 TO 8-L
```

```
60 IF A(N)<=A(N+1) T. 80
```

```
70 SWAP A(N),A(N+1)
```

```
80 N. :N.
```

```
90 F.I=0 TO 9:P. A(I):N.
```

程序运行后显示从小到大排列的顺序。若欲从大到小排列,只须把90行改为:

```
90 F.I=9 TO 0 ST. -1:P. A(I):N.
```

即可。

程序中使用了变量交换语句 SWAP。它的格式为 SWAP n_1, n_2 ,其功用是使变量 n_1, n_2 的内容进行交换。这一语句是 F BASIC 所特有的,若用一般 BASIC 语句进行两变量内容的交换,需如下编程:

```
70 T=A(N):A(N)=A(N+1):A(N+1)=T
```

可以看出,使用 DIM 语句,可以对学生成绩排列名次。读者可以试编一程序,并打印出名次和成绩。

六、CLEAR 语句

CLEAR 简写 CLE.

CLEAR 语句有两个功能:

1. 规定使用内存的范围。

为了防止因用户程序太长而与 F BASIC 系统内存空间相冲突,用 CLEAR 语句来规定用户的内存空间范围。

如10 CLE.&H7600

规定程序不能使用十六进制数7600以前的内存空间。

注意——该功能只在对话进入 BASIC 状态时有效,而在直接进入 BASIC 状态时无效。

2. 清除内存中所有变量的值。

当程序执行到 CLEAR 时,程序中所有变量的值被清除。即:数字变量的值变为0;字符变量变为空。

该功能在两种 BASIC 状态中都有效。

七、程序运行控制语句(STOP、CONT、PAUSE)

1、在程序运行中,输入 STOP 指令可中断程序运行,以便于测试程序或检查程序的执行情况。然后,输入 CONT 指令,可令程序在原中断点继续往下运行。

2、暂停语句(PAUSE)

PAUSE 语句可令程序运行暂停一定时间后再继续运行,暂停时间的控制量为0~32767。示例见随机手册。

八、子程序调用(GOSUB)和返回语句(RETURN)

在一个程序中,某些部分往往要反复执行,为避免重复编写实现同一目的程序而引入了子程序。子程序为一具有某种功能的程序,它是独立的,可以放在主程

序之外。当主程序需要这一功能时,就可以调用这一子程序。

1、子程序的调用

主程序调用子程序的方法很简单,即编一个转子语句,就可以转去执行子程序,转子语句的格式为:

```
GOSUB<n> 简写 GOS. <n>
```

式中 n 为子程序第一语句的行号。

子程序的最后一条语句必须是返回语句 RETURN,简写为 RE.,当程序执行到 RETURN 时,自动返回到 GOSUB 下面的语句继续执行主程序。

例19 设 $A=4, B=5, C=6$,试编一个程序求 $S=A!+B!+C!$?

```
10 GOS. 100
```

```
20 A=P
```

```
30 GOS. 100
```

```
40 B=P
```

```
50 GOS. 100
```

```
60 C=P
```

```
70 P. "=";A+B+C:E.
```

```
80 D. 4,5,6
```

```
100 P=1:REA. M
```

```
110 F. J=1 TO M:P=P * J:N.
```

```
120 P. "+"M"!";
```

```
130 RE.
```

```
RUN +4!+5!+6!=864
```

2、ON—GOSUB 语句

这是开关转移语句的第二种形式。格式为

O. <表达式>GOS. <子程序入口行号表>

在这里有几个子程序,就有几个入口行号,根据 ON 后面表达式的值决定转向哪一个子程序。如:

```
10 ON A GOS. 100,700,1000,.....
```

A 若取值为1,则转向100行;

A 若取值为3,则转向1000行.....

执行完那个子程序后,返回到10语句的后一语句。

3、子程序的嵌套

子程序也可以再调用其它子程序,这叫子程序的嵌套。但不允许子程序调用主程序。关于子程序的嵌套,上例中120行以后程序可以改写如下:

```
120 GOS. 200
```

```
130 RE.
```

```
200 P. "+";
```

```
210 GOS. 300
```

```
220 RE.
```

```
300 P. M;
```

```
310 GOS. 400
```

```
320 RE.
```

```
400 P. "!";
```

```
410 RE.
```

当然,在实际编程中,这样转来转去没有多大意义,示例仅在于使读者了解怎样使用子程序的嵌套。

九、有关程序调试的指令

重编行号指令 (RENUM)、删除行号指令 (DELETE) 和查找指令 (FIND) 随机手册中没有介绍, FIND 指令还没有发表。由于它们在程序的调试中经常用到, 现予以分别介绍。

1、重编行号指令 (RENUM)

RENUM 简写 REM.

一个较复杂的程序编写完毕, 一般都要在运行中调试。由于调试程序将使行号变得杂乱, 不便于阅读。这时可键入 RENUM 指令, 使行号重新排列为 10, 20, 30, …… 的整齐形式。在重编行号中, 语句中的 GOTO、THEN、GOSUB 等语句的行号也随之改变, 所以不要担心打乱了原有程序。如

```
10 F. A=1 TO 55
20 B=A * A
22 IF B>35 G. 33
25 P. A,
31 N.
33 P. "B>35"
```

键入 RENUM 回车后, 键入 LIST, 列表为

```
10 FOR A=1 TO 55
20 B=A * A
30 IF B>35 GOTO 60
40 PRINT A,
50 NEXT
60 PRINT "B>35"
```

2、删除行号指令 (DELETE)

在调试程序中, 往往要删去部分行号, 可使用 DELETE 指令。

DELETE 简写 DEL.

若键入 DEL. 100 则删除 100 行。

若键入 DEL. 100-200 则删除 100-200 行。

若键入 DEL. 100- 则删除 100 及以下所有行。

若键入 DEL. -100 则删除 100 及以前所有行。

3、查找指令 (FIND)

在调试程序中, 如欲查找某一数字、某一变量、某一表达式或字符串在程序中的位置, 可使用 FIND 指令, 尤其在阅读别人编制的程序, 掌握某一变量随程序运行的变值等, FIND 指令尤为重要。

FIND 简写 FL.

FIND 指令的使用格式为:

FL. "变量或数值或表达式"

回车后, 立即打印查找内容所在程序的所有行。

注意——FIND 指令中, 查找的内容必须用双引号引起来, 否则, 给出 TM ERROR 出错提示。

作为本章的结束, 再举一例:

例 20 民间趣题——韩信点兵

有一队兵, 若三个一数则余 A, 若五个一数则余 B, 若七个一数则余 C, 求最少有多少兵?

民间解法有一首诗以加强记忆:

三人行走七十稀,
五树梅花廿一枝。
夫妻团圆半月正, (15)
减百零五便得知。

意思是说, 被三除的余数乘以 70, 被五除的余数乘以 21, 被七除的余数乘以 15, 三个积加起来减 105 便是得数。即

$$70A + 21B + 15C - 105 = X$$

民间解法存在一个问题, 即有时三个余数之和大于 105 的两倍, 如 $A=2, B=4, C=5$, 按上述公式求出的 $X=194$, 其实 X 的最小解是 89, 所以必须减两次 105。用计算机求解, 便可避免这一问题的出现。

设某数为 N , 被 A 除余 X , 被 B 除余 Y , 被 C 除余 Z , 程序如下:

```
10 I. A, B, C
20 I. X, Y, Z
30 D=A * B * C
40 F. N=1 TO D
50 IF N MOD A=X T. IF N MOD B=Y T. IF N
MOD C=Z T. 70
60 N.
70 P. N:G. 20
RUN
```

先输入三个除数, 如 3、5、7, 回车后, 再输入三个余数 X, Y, Z , 则可打印出得数 N 。这时除数一定, 每输入一组余数, 可得一个 N 。如欲改变除数, 如改为 5、9、13, 键入 STOP 中断运行, 再键入 RUN 重新运行, 依次输入 5、9、13 后便可进行变除数计算。

本程序对原题意进行了扩展, 因此, 游戏趣味更浓厚。

到此为止, 我们介绍了 F BASIC 33 条最基本的语句和指令。运用这些语句, 读者可以较容易地编制出解决一般复杂问题的程序。还有一部分语句, 将作为 F BASIC 的提高和加深另文介绍。下一章重点介绍 F BASIC 所特有的画面、卡通控制语句。

(上接 26 页)

SIC 语言)之后, 为什么还要学习汇编语言呢? 这除了我们已经在前面介绍过的优点外, 汇编语言还有以下几个特点:

- 节省内存空间和 CPU 资源。
- 执行速度快。

- 能准确掌握程序执行时间。
- 更适用于实时控制和数据采集。
- 有利于对高级语言的评估, 更清晰理解它们在机器中的执行细节。
- 方便应用软件和系统软件的开发。



微机安装和使用中应注意的几个问题

中国人民大学信息中心系统研究室(100872) 胡野红

微机在我国有众多用户,但总有些用户在安装和使用中忽略其供电要求,以致造成不必要的损失和麻烦。轻者是机器在运行时容易出现随机读写错、死机等故障。重者则损坏主机部件,甚至发生人身安全事故。据我所知且经过数年工作实践,只要注意以下几个问题,就可避免上述损失和麻烦,给工作带来方便。

一、按左零右火连接电源线

按常规,用电设备交流输入插座是按左零(零线)右火(火线)安装。微机随机所带电源线大都有明显标记。火线英文标记为 L(Live),零线标记为 N(Neutral)。其原理是当按以上要求连接电源线时,能使火线中的电流经过机内电源的保险管。当供电线路或机器本身发生过流时,保险管则迅速熔断,使负载免受其害。其示意图如图1所示。

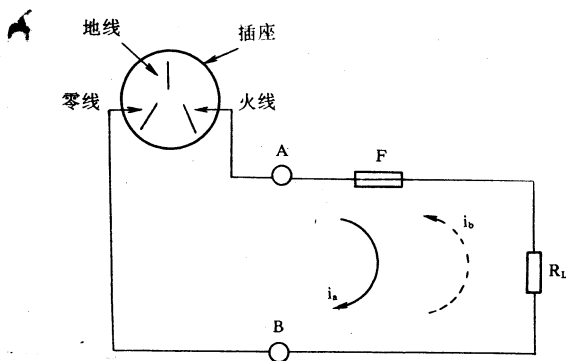


图1

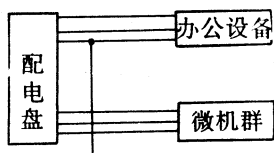


图2

正常工作时,A端是火线,电流 i_A 沿实线方向先流经保险管F再供给负载 R_L (主机)形成回路。当发生过流时, i_A 迅速超过额定值数倍或数十倍,F快速熔断切断供电回路,保护了负载 R_L 。若火线和零线接反,即B端接火线时,电流 i_B 沿虚线方向先经过负载 R_L 后再过

保险F,当发生过流时,迅速增大且大大超过额定值的 i_B 必先烧毁负载 R_L 。保险管起不到任何保险作用。所以一定要按设备要求连接电源线。

当一台微机安装完毕,加电前应仔细检查电源线连接是否正确?具体做法是:拔下电源线连接主机的一端后,合上供电电源开关,用验电笔检查该线连接主机电源插座左边一端的插孔是否是火线?若是,则再用万用表测量供电电压是否和微机标牌要求一致?否则不可给主机加电。

二、接可靠的地线

接地线的目的是当微机本身发生故障或其它原因引起微机外壳带电时电流经地线流入大地,从而保护了微机 and 操作人员的安全。其次,因主机稳压电源采用振荡频率为20kHz的开关线路,虽然在线路上采取了一定措施,但仍在其屏蔽外壳上产生数十伏的感应电压。由于该壳和主机板地线同地,若不接地线,感应电压将被加到主机板上,足以损坏仅需5伏电压供电的集成电路,或者该电压干扰主机工作,使主机出现莫名其妙的故障,如“死机”、丢失数据等。以致机器“坏了”修,修了“坏”,同一故障现象多次发生而查不到故障的真正原因,造成人力、物力、机时的浪费。所以安装微机一定要接可靠的地线。

三、忌地线、零线绞接在一起

我国大多数工、民建筑均采用三相四线制供电线路,由电工原理可知,当三相负载不平衡时,零线中将有电流通过。若微机接入这类供电线路且接地线和零线绞结在一起时,零线中有相当能量的不平衡电流将通过地线流经微机外壳,使微机外壳带电,给机器本身和操作人员的安全带来威胁。当地线和零线分开连接时,零线中的不平衡电流将流经机内电源处理。所以,安装微机千万不要把零线和地线绞接在一起。

四、微机不要和其它办公设备同用一供电线路

大多数办公设备如复印机、空调机、吸尘器、电扇、日光灯均为感性负载,当这些设备启停时,会产生数倍于(一般为10倍)电网电压的过电压,如果把微机接入有这类负载的供电线路中,该过电压足以损坏微机。所以,微机供电线路一定要和办公设备分开连接。正确的连接如图2所示。

CEC-I 中华学习机修理一例

广西来宾铁路中学(546138) 卢光怀

我使用的 CEC-I 中华学习机,半年多来出现下列故障现象:主机与电视机连接,开机启动,操作正常运行。约过20分钟左右,电视机屏幕上图象出现上下翻滚,且跳出杂乱无规则的字符,主机发出笛声报警,然后出现死机。关机过一会儿再开机启动,屏幕显示正常,键入 CTRL-RESET-TEST 机器进入自检,其显示结果也正常。操作一段时间又出现上述故障现象。

修理:开盖首先检查电源,各路电压均正常,待片子出现故障时,电源电压也未发生变化,由此否定是由电源引起的故障,洗手或抚摸金属泄放手上电荷,仔细检查主机板上印刷线路有无断裂,各分立元件有无虚焊松动,各集成块插件有无松动,均无发现异常。重新开启主机,趁未出现故障时,触摸分立元件或集成块插件,当轻微敲击视频输出盒时,故障出现。于是关机,小

心焊下视频输出盒,打开盒盖仔细检查,发现视频线插口中心焊脚稍有松动,焊脚与印刷线路板焊接处有裂纹。由此可认为是引起故障的可疑之处。用烙铁加焊锡焊牢,尔后装机。经数日长时间开机操作使用,均未再出现故障。

大家知道,启动电脑都要求先开外设电源,最后再开主机,以免外设干扰冲击主机。同样,视频输出插座接触不良,造成时断时通,干扰脉冲必然影响主机正常工作。同理,视频输出线内的接触不良,电视机或监视器输入接口接触不良,都会造成这类故障。外部设备连线的插头,经常插入和拔出主机的插座,很容易损坏接口。这提醒我们,作为使用者,在插入或拔出插头时,要格外小心,位置应正确、用力要适度;作为修理者,首先要注意接口部位的检查。

软盘驱动器综合故障维修一例

建设银行黄冈地区中支电脑科 周凯歌

我单位有一台 AST-286微机上的 1.2M 软盘驱动器(NEC FD1157C),在使用过程中突然出现三种不正常现象:1. 工作时发出很大噪声。2. 不能正常读写软盘。3. 严重划伤软盘。

根据上述故障现象,初步分析为:故障现象1与灰尘和润滑条件有关;故障现象3可能是磁头上有灰尘或1面与0面磁头不平行造成的;故障现象2可能产生的原因有很多,有待深入细致的检查。在作了以上初略分析之后,我是这样处理的:首先打开主机箱,取下 1.2M 软盘驱动器,发现里面堆积的灰尘的确很多,这与初步判断相吻合。用毛刷小心地将软盘驱动器上下表面的灰尘清扫干净后重新将其装到主机上,再开机观察,发现噪音依然存在,仔细辨听,发现噪音是从电机转动轴心位置传出的,而且噪音象是机械噪声。关机后,卸下软盘驱动器上的浮动夹紧装置,发现其压紧轮表面有一层灰尘,而且下面主轴电机轴心的杯形结构上也有很多灰尘,故先用毛刷清扫之,再用软布蘸上磁头清洁剂将其表面擦洗干净,然后装上浮动夹紧装置,重新开机后,噪音消除了。下一步再来解决划盘问题。用小镊子夹住软布蘸上磁头清洁剂小心擦洗上下两个磁头,然后再插入软盘,开机检查仍有划盘现象,关机后再认真观察上下两个磁头是否平行,当然凭直观很难看出,

我采取的办法是,不管平行与否,先作一次平行校正。具体作法是:将一块磁头清洗盘插入该软盘驱动器中,然后反复开关驱动器的小门,并在关上小门后用手指均匀下压1面磁头(注意,这些步骤是在没开机的情况下进行的),作了上述处理之后,划盘问题也得以解决。剩下的就是集中精力解决不能读写的问题。如前所述,造成不能读写的可能性很多,比如无索引信号,读写电路故障,读写磁头断线,磁头定位不准等等,都会造成读写不正常。使用一台 CQC-A 型磁盘驱动器测试仪和一台 V-1065型双踪示波器,接上该软盘驱动器,先用一张空白软盘插入该驱动器,关上小门后,发现有索引脉冲,“00”道信号和 RDY 信号,再检查其寻道功能也正常,然后进行自读自写,从示波器显示的读写波形来看是正常的。以上检查说明该驱动器的电路部分和两个磁头都是好的,可以断定是机械定位问题。把空白盘取出,换上 CE 盘,利用示波器和驱动器测试仪,反复调校“猫眼”信号,方位角信号,“00”道信号和索引信号,直到上述四种信号的波形都合乎精度要求为止。这时取下该驱动器,将其安装到主机上,先测试其读写和格式化功能都正常,再检查其互换性也很好,至此,该软盘驱动器修复。

电子工业出版社软件部新出版软件介绍

一、PC 及其兼容机软件

1. 汉字复杂报表自动生成系统 软盘:2片

定价:150元

本软件采用菜单选择方式,能自动生成多层表头汉字报表,表格修改方便,还能被其它管理系统调用,使已用 dBASE 进行事务管理的系统得到扩充。它在 DOS 系统下直接运行,操作简单,兼容性好。

2. 中西文辞书编纂系统 软盘:2片

定价:180元

本软件用 Turbo BASIC 编成。它能对用 EDLIN 或 Wordstar 输入的无顺序中西文词条,按设计文件格式,进行自动排序。排序方法有四种,即汉字按“纯笔顺”、“音序十笔顺”和“偏旁十笔顺”排序,英文按“字母”顺序排序。本软件可在 DOS 下直接运行,适合辞书作者和编辑以及其它需要进行字、词、人名与地名排序的工作人员使用。

3. 多方案联想式长城系列机 DOS 软盘:3片

定价:200元

本系统是在长城系列机及其兼容机的硬件基础上研制的新一代汉字磁盘操作系统。它沿用了原 GW BIOS 的扩充 ROM-BIOS 10H 外部显示管理模块的功能,新设计了 16H 中断管理程序,使之可支持 O14、CEGA 和 CMGA 等显示系统;屏幕提示区扩大,提出格式优化;它支持多种汉字输入法且具有联想功能;它有字词两种编码,可重复输入,能自动进行大小写转换,自动显示内码、区位码和外码,提供窗口式帮助。

4. 彩色屏幕界面程序自动生成工具 软盘:1片

定价:260元

本软件通过选择菜单,利用上下左右键即可确定其在屏幕上的位置,自动生成用户设计的、可在 dBASE 或 FoxBASE 环境下运行的程序,该程序能并入用户所开发的系统。

5. 通用工资管理系统 软盘:1片 定价:300元

本软件的功能是建立工资数据库,修改库结构和内容,任意设定和修改工资计算表达式,对工资数据进行查询和打印,完成任意单位的工资核算处理。它通用性强,结构灵活,操作简便,适用于 IBM PC/XT, AT 及其兼容机,要求 640K 以上内存,显示 25 行以上的汉字。软件环境:CCDOS, UC DOS, NCDOS 和北大 DOS 及西山 DOS 等。

6. 清华中西文多语种操作系统 软盘:12片

定价:600元

本软件主要是汉字操作系统,也能处理其它多国文字,它不仅本身自成系统,也可挂靠在其它汉字系统。汉字有繁简两种字库,输入方法有:双拼字词联想码、混合拼音码、形声码、方言代码、短语码、五笔字形码、区位码、电报码、英语码和德语码等。多国外文输入语种包括:日语、俄语、希腊语、德语、法语、意大利语、西班牙语和葡萄牙语等,而且还可以混合输入这些外文。

7. 手写汉字输入系统(HGD-9200-1)

软盘:3片(另含书写版、写字笔和专用卡)

定价:2000元

该系统通过接口卡和标准 RS-232C 串行接口与主机连接并启动后,即可用写字笔在书写板上手写汉字、绘图和制表等,实时地将汉字和图形信息输入计算机。凡是会写汉字的人,无需记忆任何编码,均可按手写汉字的习惯进行信息输入。

二、中华学习机及其兼容机软件

1. 电路实物连接图练习 软盘:1片 定价:15元

本软件用来练习连接中学物理课中常见的 56 种电路(简单电路、串联电路和并联电路),练习者可自选器材和电路接法。当练习者遇到困难时,机器能予以指导并绘出正确的连接图。本软件适合课堂教学演示和课外自学练习。

2. 美术字幕生成编辑系统 软盘:1片

定价:35元

本系统在超级汉字文章编辑系统 V3.0 支持下工作。它可将中华学习机硬汉字加工处理成空心、粗体、细体和立体美术字,字型可放大成 13 种不同的尺寸。经处理的美术字幕能生成独立的显示模块,存盘后可在其它系统下调用,还提供有打印字幕功能。

3. CEC—I 联想汉字输入系统 软盘:1片

定价:50元

本系统与拼音方式配合,具有多级联想方式,能自动转换。它有良好的扩展功能,用户可建立与系统字词库合并使用的专用联想词库,该词库在使用中享有优先权。

4. CEC—I 通用字词输入文章编辑系统

软盘:1片 定价:72元

该系统是中华学习机扩充汉字输入法的良好环境和编辑工具。用户可利用该系统定义五笔、双拼、声形等输入法,可建立自己的字、词输入法和联想字库,它配有五笔字形字、词库;用户可利用该系统象编制 BASIC 程序一样的方法来编写文章。

5. ML BASIC-3.0(程序设计语言) 软盘:2片

定价:75元

ML BASIC 语言是一种扩展 BASIC 语言,它比 CEC BASIC、APPLESOFT 多 78 个新语句,是软件设计的优选新语种。它可用于数值计算、符号、音乐与图形处理和程序自扩张方面的程序设计。它的图形功能不仅兼有 LOGO 语言的作图优点,甚至在某些方面还超过 LOGO 语言。

欲购软件,请汇款到:

北京万寿路 173 信箱(邮编:100036)

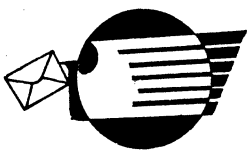
电子工业出版社软件部

联系人:谈众安 王旭

开户银行:北京工商银行翠微路分理处

户名:电子工业出版社杂志编辑部

帐号:8913333-59



普及型 PC 个人用户软件交流联谊活动

问题解答(一)

读者联谊

北京中国农科院计算中心(100081) 王路敬

自从九一年一月份《电子与电脑》杂志组织“普及型 PC 机个人用户软件交流联谊活动”以来,得到社会各界人士,尤其是《电子与电脑》读者的大力支持和充分肯定,作为联谊点的挂靠单位,我们中国农科院计算中心培训部全体同志能有这样一个与广大用户直接联系,直接交流应用体会的机会,能为 PC 机个人用户获得自己需要的软件,提高用户应用水平作一些力所能及的工作感到十分高兴。

到目前为止,我们先后收到来自全国各行各业 PC 机个人用户要求参加该项活动的已近200人,尤其《电子与电脑》今年第7期刊登第一批“常用交流软件清单”以来,收到大量用户来信。大家积极主动的提供了自己的软件清单;反映了希望通过联谊活动得到自己所缺乏的软件的愿望和要求;提出了在使用中碰到的困难和问题;对搞好联谊活动提出了殷切的希望。针对大家的来信、来函、来电,以及直接到我们这里联系的,我们已组织人员将有关材料进行了计算机管理。归纳大家的普遍要求:一是由于机器硬件资料缺乏,对其性能了解甚少,进行基本操作尚有一定困难。希望提供有关硬件方面的资料信息;二是由于缺乏软件,机器的功能不能很好的发挥,利用率低,希望交流交换软件;三是有了软件由于缺乏必要的使用性说明材料,致使软件不能得到利用,所以希望提供有关软件的使用说明或有关资料的来源;四是使用中碰到一些困难和问题,不能及时得到解决,借此活动交流使用经验、体会和实践中遇到的问题解决方法与途径。针对上述问题和大家的普遍要求,从今年第1期起在《电子与电脑》杂志上,以问答的形式,对普及型 PC 机从硬件系统,常用交流软件以及在使用中碰到的困难和问题,经过归纳、整理,奉献给参加 PC 机软件交流联谊会的朋友以及 PC 机用户,以起抛砖引玉的作用,把我们这一“联谊活动”搞的更好,达到不断提高应用水平之目的。

PC 机硬件系统

1. 国内目前普及型 PC 机的代表机型有哪些?如何考虑 PC 机的配置问题?

继 IBM-PC 系列机、长城系列机以及中华学习系列机应用推广以来,在计算机知识日益普及,计算机已逐渐被家庭、被社会接受的时候,近年来在我国微机市场上又先后推出了大众化普及微型计算机。其代表产品如北京海华公司,航天 CAD 公司,北工大电子厂开发的海华—航天个人电脑 HH-PC,北京北方电脑公

司开发的 BFPC-BOY,四川的新潮 XC-PC,上海长江集团公司推出的东海0520SD 小博士电脑,天津通博电子技术联合公司推出的 TH-MINI-PC,陕西省推出的长安 SUPER-PC 等都颇受用户的欢迎。这些机型与 IBM-PC 相比较运算速度更快,内存容量更大,系统扩充能力更强。可根据用户的不同需要进行组合配置,除少量集成电路外,大部分元器件均可在国内解决,维修方便。由于与 IBM-PC 系列和长城等系列兼容,所以系统软件,应用软件来源多而广,为发挥系统的功能提供了非常便利的条件。

这类微机适宜于各类学校、家庭、个人作为教学、学习和家庭事务处理使用,还可以作为企事业单位一般文字处理,信息管理等小型事务的微机处理以及计算机网络的结点机或多用户系统的智能终端机使用。

一台 PC 机系统组成应包括二大部分,硬件和软件。所谓硬件是指组成一台 PC 机所有固定装置的总称。主机,显示器,键盘打印机等都是系统的硬设备,统称 PC 机系统的硬件。主机包括微处理器和内存储器,微处理器是 PC 机的核心,它要完成数据的处理,加工和控制功能,区分 PC 机档次高低首先看 PC 机的微处理器性能。内存储器主要存储程序和数据,它们是通过键盘输入到系统的内存储器,然后再执行。显示器,打印机是 PC 机系统的输出设备,显示器显示 PC 机处理的信息,打印机在打印纸上打印处理的结果,包括字符,汉字,图表等。所谓 PC 机的软件是指提供 PC 机能够工作的各种程序的集合。PC 机系统的软件包括系统软件和应用软件两大类。系统软件是指使用管理 PC 机的软件。这一类软件包括磁盘操作系统软件,各种高级语言例如 BASIC 语言, dBASE I, dBASE II 等编译或解释程序,各种服务性的程序,例如自检程序,诊断程序及工具软件 PC-TOOLS 等。应用软件是指根据 PC 机硬件资源、系统软件资源编制的解决具体问题的程序。使用 PC 机首先要了解和掌握硬件系统的配置有哪些,系统软件配置有哪些,在这种硬件和系统软件的环境支持下,能够运行的应用软件又有哪些,怎样进行系统扩展可以运行更多的应用软件等等。其次是掌握正确的操作使用方法。

以 HH-PC 和 BFPC-BOY 机为例,说明如何考虑 PC 机的配置问题。

HH-PC 型微机与 IBM-PC/XT 完全兼容,系统硬件基本配置:

CPU: Intel 8088-2

时钟主频: 4.77/10.7MHz

内存储器: 256KB

软盘驱动器: 360KB×1

显示器: 12英寸高分辨率单色显示器, 分辨率720×350

键盘: 101键

主机接口: 显示器接口、电视机接口、软盘驱动器接口、光笔接口、并行打印机接口、RS-232异步通信接口、协处理器8087插座、PC总线扩展槽一个

硬件选配件:

黑白双频14英寸直角平面单色显示器, 分辨率720×350, 640×200, 高分辨率彩色显示器, 分辨率为640×350, 800×600(EGA), 1024×768(VGA)

内存扩展卡, 由256KB可扩展到640KB

5. 25英寸20M硬盘

3. 5英寸20M硬盘

5. 25英寸360KB软盘驱动器

打印机, 各类24针、9针打印机

CAD设备, 鼠标器, 光笔, 扫描仪, 数字化仪, 绘图

仪

16×16点阵汉字库

网络卡, 3+网卡, AT&T网卡, NOVELL网卡

软件基本配置:

PC-DOS(支持不同版本的DOS系统)

BFPC-BOY机

系统软件基本配置:

CPU: NEC V20与8086/8088兼容

8087插座: 配8087协处理器

主频: 4.77/10MHz

内存储器: 256KB

软盘驱动器: 360KB×1

硬盘驱动器: 可选

软/硬卡: 软卡

单色显示器: 720×350/640×200

显示卡: MGP/CGA

接口: 并行口

键盘: 101键

主机板插槽: 标准的PC/XT插槽

电源: 功率100W

软件基本配置:

PC-DOS(支持不同版本的DOS系统)

MKF(图形转换软件)

HH-PC或BFPC-BOY主机采用Intel 8088(或V20)CPU, 这是一个16位的微处理器, 地址总线20条, 最大可寻址可达1MB字节, 主板内存可由256KB扩至512KB或640KB。

所配置的键盘为标准的PC/XT键盘, 其功能和使用方法与IBM-PC/XT完全相同, 其几个主要的组合控制键的功能如下:

Ctrl+Alt+Del: 重新启动系统(热启动)

Ctrl+Break: 终止当前执行中命令

Ctrl+NumLock: 暂停屏幕滚动显示

Ctrl+PrtSc: 将屏幕显示同时输出到打印机

Num Lock: 光标移动状态/数字状态转换

Caps Lock: 字符大小写转换

所配置绿色显示器, 分辨率为720×350时, 文本状态下80×25行显示, 可进行西文信息处理。若玩游戏时, 首先要使用“图形转换软件”MKF或MKB将显示器的方式模拟成为图形转换方式。方法是先将“图形转换软件”盘插入软盘驱动器内, 然后执行MKF或MKB即可。即执行:

A>MKF

或 A>MKB

若使用CGA卡和640×200绿显时则可不用“图形转换软件”。

所配置的软盘驱动器及软磁盘与标准的IBM-PC机上使用的完全相同, 即5.25英寸双面双密度360K软盘。每张软盘分为40个磁道, 两面共80个磁道。每个磁道又分成9个扇区, 每个扇区内存放512个字节, 故每张软盘的密度为512×9×80=360640字节, 为360KB。

系统的安装与启动可以这样进行: 将主机放置在水平桌面上, 将键盘联于主机的键盘插座上, 显示器的信号线联于主机的显示卡端口上, 然后把主机及显示器的电源线分别联在交流电源插座上。

启动时将随机系统盘即DOS操作系统盘插入软盘驱动器中, 然后开机。开机时先开显示器电源(如有打印机, 再开打印机), 然后开主机电源。开机加电的过程称之为“冷启动”。关机时则顺序相反, 先关主机电源, 然后关显示器电源(如有打印机, 再关打印机电源)。当有非法键盘操作时, 有时会出现机器“死锁”, 这时可同时按Ctrl+Alt+Del键, 或按前面板上的“RESET”钮重新启动系统, 这一过程称之为“热启动”。

在HH-PC或BFPC-BOY等个人电脑上使用的操作系统DOS, 可以用低版本如DOS 2.0或2.10, 也可使用DOS 3.0以上的高版本, 但根据系统的硬件资源, 建议使用DOS 2.0或2.10就可以了, 不要追求高版本, 否则有些应用软件的使用会带来一些困难。

DOS为用户提供了一种工作环境, 它主要负责文件管理, 内存管理, 外设备管理和CPU管理, 而用户使用时则是透明的。

DOS的构造是复杂的, 主要层次为IBMBIO.COM(基本输入/输出系统), DOS核心IBMDOS.COM和命令处理程序COMMAND.COM。在DOS系统盘中我们只能见到COMMAND.COM, 而DOS的前两个层次IBMBIO.COM的IBMDOS.COM的接口是COMMAND.COM, 它接收用户从键盘上发出的命令并解释执行之。

DOS能够接收的用户命令分为三类:

(1)内部命令——驻留在内存中的如DIR、TYPE、COPY、ERASE、RENAME、CLS、MD、CD、RD等。

(2)外部命令——不驻留在内存中或称暂驻程序,

如 FORMAT, DISKCOPY, DISKCOMP, CHKDSK 等。

(3) 批处理命令——是由内部命令, 外部命令和批命令组成的文本文件, 是一类可执行文件, 其文件的扩展名为 .BAT。

当加电启动后, 机器的 ROMBIOS 开始执行自检和引导装入程序。ROM 引导装入程序从 DOS 系统盘上的第一扇区读入内存, 并由磁盘引导装入程序将两隐含的文件 IBMBIO.COM 和 IBMDOS.COM 读入内存, 然后分配磁盘缓冲区和文件控制块的地址空间, 最后装入 COMMAND.COM 的驻留部分, 便显示系统提示符 A>, 此时 DOS 进入正常工作状态, 等待用户的作业。

在系统内存容量基本配置 256KB, 360K 软盘驱动器的情况下作为家庭、学校进行智力开发, 语言启迪, 游戏等是完全胜任的, 若要进行家庭事务处理、企事业单位一般的文字处理、信息管理等工作要增加汉字处理的功能。HH-PC 或 BFPC-BOY 或兼容机实现汉字处理功能有以下几种途径:

(1) 扩充专用汉卡。这类机器系统主板上都有一个或几个扩展插槽, 将汉卡直接插入扩展槽即可, 专用汉卡不占主机内存。这样相当于扩充了内存的容量。例如陕西省计算机公司推出的长安 SUPER 汉卡, 该汉卡采用了 7 个 4MBIT 芯片制作 48 点阵宋体、仿宋、黑体、楷体精密字库和 10 种 128 点阵西文字库。它支持 24 针打印机, 支持多种的汉字输入方法, 编辑打印集于一体, 兼容 WS 和 MS2401 功能, 全屏幕多窗口, 弹出选择, 系统帮助, 一看就懂, 一学就会。汉字字体美观, 文字可任意大小, 而且具有在屏幕上模拟打印输出功能, 使用起来很方便。航天汉卡, 王码高速字符型汉卡都具有类似的功能。

(2) 将主机内存由 256KB 扩充到 512KB 或 640KB 后即可使用市面上流行的中文操作系统 CC-DOS。例如 CC-DOS 2.0/2.1, 或 CC-DOS 4.0。在中文操作系统支持下可运行 IBM-PC/XT, 长城 0520A 型机的汉字软件。例如 WS, dBASE II, dBASE III, 五笔字型以及高级语言 BASIC、FORTRAN、COBOL 等软件。

(3) 利用压缩字库 CC-DOS 可以在基本配置即 256KB 内存的 HH-PC 或 BFPC-BOY 或兼容机上实现汉字处理功能。但是由于基本内存太小, 仅能做一般简单的文字处理, 象 dBASE III, 五笔字型等软件仍不能运行。

若将内存扩充至 512KB 或 640KB, 扩充一个 20MB 的硬盘, 其功能相当于 IBM-PC/XT, 所有在 IBM-PC/XT 机上运行的软件均可在个人电脑上运行。

无论采用上述何种方法实现汉字功能, 其汉字输入方法和操作都是相同的或类似的, 区别仅在于实现汉字的形式和途径不同。

个人电脑上启动 CC-DOS 进入中文状态后功能的定义如下:

ALT+F1	选择区位码输入方式
ALT+F2	选择首尾码输入方式

ALT+F3	选择拼音输入方式
ALT+F4	选择快速输入方式
ALT+F6	选择 ASCII 码输入方式即英文输入方式
CTRL+F7	纯西文/纯中文方式转换
CTRL+F8	建立自动光标/取消自动光标转换
CTRL+F9	建立纯中文方式/取消纯中文方式转换
CTRL+F10	选择打印机字型和行宽方式

对个人电脑连有打印机的使用者, 在输出汉字时, 可先在 CC-DOS 状态下挂打印机驱动程序。不同类型的打印机使用其相应的打印驱动程序。运行驱动程序的方式有两种: 一种在 CC-DOS 启动后, 当提示符出现, 在提示符后从键盘上打入相应打印机驱动程序的文件名并回车即可。另一种方法就是驱动程序的自动引导。也就是说打印驱动程序放在 AUTOEXEC.BAT 自动执行批处理文件中, 这样在 CC-DOS 启动的时候同时自动执行批处理文件的命令, 从而达到驱动程序的自动引导。但是有一点需要注意, 打印驱动程序要在 CC-DOS 的系统盘上。建立自动执行批处理文件方法可以利用 COPY CON: 命令, 也可用 EDLIN 行编辑程序或其他的编辑软件。例如在 CC-DOS 2.0 系统盘上建立 AUTOEXEC.BAT 文件, 所用打印机是 3070, 其操作如下:

(1) 把 CC-DOS 系统盘插入软盘驱动器 A 中, 并启动;

(2) 待系统提示符 A> 出现后从键盘打入如下信息:

```
A>COPY CON:AUTOEXEC.BAT
ECHO OFF
CLS
FILE1
CCCC
ALL24P
^Z
```

这样 AUTOEXEC.BAT 就建好了, 再重新启动系统即可达到打印驱动程序的自动引导。

在 CC-DOS 状态下使用 CTRL+F10 可以选择打印字型和行宽。能变换多少种字型取决于驱动程序的功能。可以使用 SHIFT+PrtSc 拷贝屏上的内容, 在打印机上输出要打印的内容。如果要打印一个文本文件的内容, 可以使用 TYPE 命令将显示的信息送到打印机, 其操作如下:

```
A>TYPE 文件名 > PRN
```

普及型 PC 个人电脑是性能价格比较好的微机系统, 可以根据工作的需要, 在基本配置的基础上适当的组合配置, 以满足不同层次的需要, 运行各种流行中西文软件, 或开发自己满意的软件产品。

(待续)