

# 微小型计算机 开发与应用



6

1987

微小型计算机开发与应用编辑部

## DEC推出两款适用于商业及工业市场的VAX计算机系统

美国DEC计算机公司最近推出两款性能最高的VAX计算机系统,为商业及工业用户提供最先进的计算机设备。

DEC最新推出的VAX8974及VAX8978计算机系统,比VAX-11/780系统的效能高出达五十倍,为目前使用传统大型计算机的行业,如会计、电子转账、数据库管理及数据处理、办公室自动化、研究及发展等,提供了新的选择。

VAX8974及VAX8978计算机系统是根据DEC备受欢迎和非常成功的VAX簇及先进储存技术设计而成。还有全面性的客户支援服务,所以特别对那些需要广泛使用集中式及分布式资料处理的大型用户来说显然是极大的喜讯。

这两款系统与VAX系列的各种计算机完全兼容,亦可连接于DEC本地及广泛地区网络。

VAX簇的科技成就使VAX8974及VAX8978系统提供高度处理能力,价钱却与传统大型计算机相当。现全球已装置了五千六百套VAX簇系统。

DEC中国市场拓展经理黄祖麟说:「传统的大型计算机一向由中央处理系统控制计算资源,但新的系统可用性更高,能在增加容量时维持操作,可与其他VAX系统完全兼容,使这些系统成为市场上最有竞争力的大型计算机。

该两款新型系统采用VAX簇科技,将处理子系统及外存设备连成总体,假如部分设备因故障或定期维修而停用时,系统仍能操作及处理数据。这项设计可减少在计算机系统出现故障时所浪费的时间。

此外,DEC的VAX Volume Shadowing软件能够使用户在储存数据的同时,立刻自动复制一些重要的资料以作备用。

新系统可使用现有的VAX软件,而且无须重新编译及连接,节省用户在软件上的投资。和一般功能相等的大型计算机相比,新系统能为用户节省超过百分之四十的成本。

在系统管理方面,DEC的VAXperformance Advisor (VPA)可帮助系统经理找出系统的瓶颈部分,并将系统调节至最佳效能。VAX簇科技为主机用户提供了高度的实用性。

DEC同时为用户提供一套行业中最先进及全面的维修器材,其中包括一套完整的监察系统,能在系统及网络即将发生问题前提醒操作员以作防范。

VAX8974及VAX8978系统能够按客户工作需要增添处理子系统及储存子系统,而不会影响系统的操作程序。

本栏目稿件全部由美国DEC中国有限公司提供

---

编辑:《微小型计算机开发与应用》编辑部

出版:天津市电子计算机研究所

地址:天津市河西区友谊路爱民道5号

发行:天津市邮局

印刷:天津市静一胶印厂

定购处:全国各地邮局

定价:0.60



# 目 录

1987年 (双月刊)

第6期 (总第32期)

## 计算机软件

- 软件产业概述.....李凤祥 王俊铁 宋雅正 (1)  
再谈被删除文件的恢复问题.....崔秀起 (10)

## 计算机系统

- Sun-3工程工作站的系统结构 (续3) ..... (14)

## 微小型计算机应用实例

- IBM PC/XT工业控制系统硬件及软件实现.....赵育才 冯德田 (17)  
计算机符号演算软件的发展与应用.....向延育 (22)  
一个实用的A/D, D/A转换电路.....王士元 (28)

## 编译·译文

- 逻辑阵列器件的特点及应用..... (34)

## 动态与信息

- 实现我国金融技术手段现代化的支撑环境.....杭承让 杨润征 (43)  
美国DEC公司信息与产品报导.....美国DEC中国有限公司 (封四)

- 《微小型计算机开发与应用》编委会会议纪要..... (48)  
《微小型计算机开发与应用》1987年总目录..... (封三)  
征订启事..... (47)

# 微小型计算机开发与应用

## 一九八七年总目录

### 第一期

DGC32位超级小型机新产品——MV/7800  
32位机总线的实际性能比较  
微小型计算机的性能评价与选型  
一个软件工程开发估算系统的设计  
ORACLE简介  
怎样定义好激光打印机与计算机的软件接口  
OAS/724办公室自动化系统的开发步骤与方法  
化工信息管理系统HUAGON G  
在IBMPC/XT屏幕上绘制趋势图和相关图  
计算机服装辅助设计系统的应用研究  
一种控制器用的专用微机  
通用性较强的数据检索程序(四)  
用APPLE II机把Z80汇编文件写入EPROM  
的一种简便方法  
用在Z80微处理机上的8位十六进制数字显示器

### 第二期

决策支持系统体系结构初探  
32位机MICROVAX II及其工作站的特点分析  
KBS-1知识库系统  
用Apple II开发MCS-51单片机  
提高具有汉字处理功能程序系统运行速度的编程技术  
报表数据的压缩处理技术  
如何建立计算机企业管理信息系统  
模具管理子系统的设计  
电解制铝生产自动控制系统  
多用户微型机系统市场分析及主要评选标准  
82586 局域网控制器  
Ethernet网的中断程序设计  
长城0520C作为汉字以太网服务器的实现  
通用性较强的数据检索程序(五)

### 第三期

以市场为导向,发展信息产业  
日本信息产业的水平、发展历程和战略  
微机控制技术与机电一体化  
机电一体化的现状与发展

微机控制电封闭试验台

Sun-3工程工作站的系统结构

VME bus 32位微机总线

第二代32位微处理器68030

用单板机控制实现数字/交流转换

用最低配置的APPLE II实现大中型立体仓库的  
现代化管理

用8255A设计苹果机接口的一点经验

苹果机在教学评估中应用初探

通用性较强的数据检索程序(六)

美国DEC公司信息与产品报导

欧美研制光子计算机取得重大进展

### 第四期

捷克斯洛伐克的微处理机

Sun-3工程工作站的系统结构(续1)

港口外贸费收管理系统的分析与设计

产品结构优化系统

计算机在收购粮食工作中的应用

计算机在能源审计中的应用

DBASE II与高级语言关系的探讨

微机应用中被删除文件的恢复

汉字AST-PCNET宽域网

可编程控制器S5-115U的CPU硬件结构

### 第五期

生产计划管理软件包的设计

浅谈机器翻译的翻译方式

屏幕显示特大汉字的一种新尝试

用Dixon准则剔除实验数据中坏值的微机程序

Sun-3工程工作站的系统结构(续2)

微机在FUZZY性事物量化研究中的应用

单板微型计算机在岩土力学参数测量中的应用

用计算机实现时分制长话通信自动控制可行性探索

机动车及驾驶员微机管理系统

一种微机局网分布式资源共享的设计思想与实现

1M位DRAM控制器

IBM PC通用EPROM编程器的设计

美国DEC公司信息与产品报导

### 第六期(见本期目录)

敬爱的读者：

一年来，《微小型计算机开发与应用》得到了您的大力支持和关怀，在此特致谢意。为了进一步了解您的需求，现特发征求意见表，恳切希望您能提出宝贵意见，以使本刊不断改进，更好地为您服务。谢谢！

微小型计算机开发与应用编辑部  
一九八七年十二月

姓 名		专 业		文化程度	
职 务		单位及地址			
您喜欢的栏目：					
本刊哪篇文章曾帮助您解决过哪些实际问题？					
您对本刊的希望和要求：					

# 软件产业概述

## ——国内外软件产业发展状况

李凤祥 王俊铁 宋雅正

(天津市电子计算机研究所)

**摘要** 本文概述了软件工程的起源和出现的背景,叙述了以软件工程为基础的软件产业的概念和 其 展阶段,以及软件产业的几项关键问题,一些基本方法和模型。本文还评述了国内软件产业的发展概况, 有利条件和存在问题,大力发展软件产业在国内是必要的也是可行的。

### 一、软件工程和软件产业

软件工程的 概念起源于六十年代末期。1968年10月,北大西洋公约组织(NATO)的科学委员会中所设的计算机科学研究组,首次提出“软件工程”的概念。其背景如下:

#### 1. 计算机系统日趋庞大复杂

1968年7月美国IBM/360的操作系统OS/360(14版)具有100万条指令,到1978年已增加到300余万条(21版),6000多个模块。

#### 2. 软件开发的周期长,成本高

计算机硬件生产效率成百倍地增长,而软件生产率十几年才增长1~2倍,而且脱离不开手工劳动。IBM/360的操作系统前后共花了数千人年之多,投资十分可观。美国面向国防的实时系统SAGE,花了近2.5亿美元。

#### 3. 可靠性低

1962年美国飞向金星的第一个空间探测器(水手1号),因一个语句的语义错,致使其偏离航线,遭到破坏。另外,花费了巨额投资,精心设计的阿波罗登月飞行系统软件,仍然屡屡出错,如阿波罗14号的10天飞

行中,出现了18个软件错误。

总之,由于软件规模大,复杂性高,生产率低,产品质量难以保证,开发费用日益上涨,造成供求关系严重脱节。在这样的背景下,60年代末形成了“软件工程”的概念。当时还是比较模糊的概念,其想法是软件科学工程化、体系化,利用有关的生产方法和工具,降低结构的复杂性,提高管理水平和质量,争取高效。经过10多年的发展,软件工程已接近于传统生产过程的概念,并不断创造出新的方法和工具及文档标准,以适应于软件生产周期的几个阶段以及对应的规范化文档及管理技术,把其统一成软件工程开发环境。这样就暂时缓和了软件的供求紧张状况,提高了效率及可靠性、可维性等问题。同时,软件生产的“纪律化”和“规范化”又可促进软件的商品化。1969年IBM公司实行了“价格分离”政策,软件单独计价出售,也促进了软件向商品化发展。

以上情况都说明了“软件工程”是基于“软件危机”这个现实情况而产生的,软件工程所提出的方法和工具,重点解决软件生产的质量和效率。1974年提出的软件生命周期及Waterfall(瀑布)的设计方法(层次模块化的程序设计方法以及各种测试技术



等等)。以及80年代出现的throw-away型新的生命期模型,对软件生产起了划时代的影响。

直到现在,这些宝贵的的方法和手段仍然被世界范围内的软件工程项目所采用。但是,就目前来说,由于软件设计人员的缺乏,造成的新的“软件危机”尚未从根本上解决。原因是软件工具不配套,整个工程没有统一思想做指导。缺乏集成化的软件工程支撑环境,因而软件无法预测,成本的增长失控,软件新技术的发展,不能解决全部飞速增长的软件需求。据估计,美国1981年专业软件人员有40万人,缺少10万人,按每年增长4%计算,到1990年,软件人员将缺少80~100万人之多。

软件产业比软件工程具有更广泛的含义,然而,软件工程是软件产业的基础。软件产业是信息处理产业的核心部分。把软件产业简单定义为“根据用户要求而制作程序的事业”是不够全面的,因为软件的开发、维护和管理比制作要付出更多的代价。“制作”的确切含义应阐述的更清楚一些。我们说,软件产业应是软件研究、设计、开发、评价、生产、销售、安装调试、维护、管理和服务的事业。从事软件产业,目的是高质量、高效地向用户提供软件商品,并对用户加以各种服务。从世界范围内来观察,软件产业可分成三个发展时期:

#### (1) 初创时期

1954年美国通用电气公司在UNIVAC I型计算机上实现了职工工资计算;1959年出现了第一家独立于计算机制造业的软件公司,表示了软件产业的开始。

#### (2) 成长时期

1969年IBM公司推行的“价格分离”政策,软件单独计价出售;1968年出现了“软件工程”的概念,1974年的软件生命期概念及瀑布式设计方法等等,标志着大幅度地提高了软件开发的可靠性及效率,软件产业快速

成长起来。

#### (3) 成熟时期

80年代,以美国为首的软件产业趋于成熟。不但在软件开发上有一个较完美的软件开发环境,而且在管理上逐渐趋于完美。在软件开发中的文档组织方面日本已经走在前面。在软件的维护、服务、评价、标准化、成本核算和计价以及针对不同的国情制定出适合自己发展的软件发展政策,都是值得我们借鉴的。总之,80年代的今天,软件产业在世界范围内逐渐成熟,有人用四句话来综合表示这个阶段的情况,即:软件开发纪律化;软件成果商品化;软件经营企业化;软件服务社会化。

## 二、软件产业的几个关键问题

### 1. 软件的开发和维护

#### (1) 软件开发环境

软件开发到目前已经历了手工开发的初始阶段;程序设计方法和测试方法的研究阶段;软件工程开发方法(特别是软件生命期的出现)等多个阶段。随着微电子技术的飞跃发展,人们开始进入基于计算机系统和工具的开发方法的新阶段,即建立软件开发环境。软件开发环境包括以下内容:

##### 1) 计算机系统

2) 软件开发方法论;包括管理规程和技术方法。

3) 软件工具:包括分析、设计、开发、调试和维护工具。

4) 环境相关科学:包括生态学(研究人与开发环境的关系),人类工程学(研究人与机器的关系),用户心理学及哲学等。

#### (2) 软件工程的法则、公理和定理

到目前为止的软件工程,已经产生了不少普遍的法则,或是可以作为标准的公理、定理。仅举几例:

##### 1) 并发过程的同步方法: Dijkstra

〔DIJ65〕提出临界区，一个时间，只允许一个进程进入，不允许中断，解决多进程的同步问题。

2) 结构化程序设计与阶段的细分：Dijkstra提出含有GOTO的程序设计不好。应进行结构化的程序设计。Wirth〔WIR71〕更进一步将这些原则体系化，提出了从最高层次，顺序向下，逐步按阶段细分下去直到模糊问题全部解决为止。

3) 模块规格：Parnas〔PAR72〕把模块定义为：管理抽象目标（实体）的功能体。可以观察目标状态变更，改变目标的界面是公开的，除外，全部应隐含于内部，以此为基础提出了模块规格描述法，并产生了抽象数据类型。

### (3) Waterfall型生命期

图1给出了Waterfall型生命期。

生命期各阶段或者说各层次象瀑布

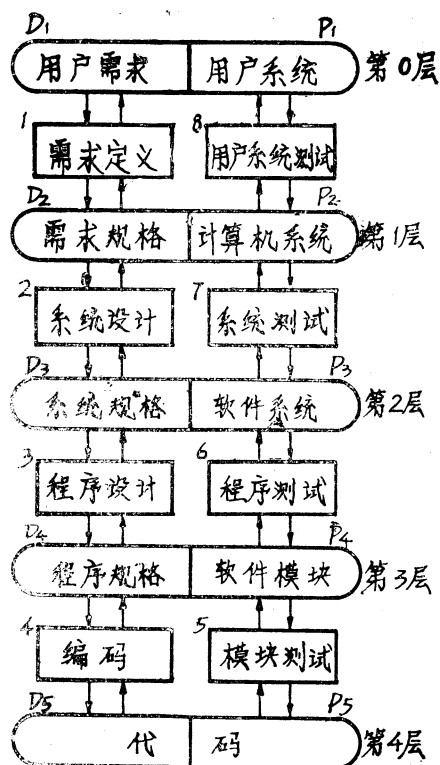


图1 Waterfall型软件生命周期模型

一样，从上向下流动。向上的箭头表示反馈。过程中的层次是在作业内容发生本质变化的地方，称为抽象层。越接近代码处抽象度越低。这种思想是70年代软件工程的共同基础。很多软件工厂，均按此方法进行。但它存在以下特点：

a) 当分析了用户需求，制定出规格书以后，没有对设计者的想法是否与用户要求一致进行检查，而是不停地向前进行，用户没有修正的机会。

b) 设计者的概念是否能通过规格书表达出来，也是一个问题。针对以上的问题，在80年代又出现了一个新的生命期模型，它补充了Waterfall生命期的不足。

### (4) Throw-away型生命期

以Throw-away型生命期模型为基础研究出来的技术有：

a) Prototyping：即在需求定义阶段，使用相当简易的方法设计出快速原型（Prototyper）来试运行，检验是否可以满足用户要求。它最好是实际系统的子集合。目的是让用户早一天看到用户实际系统是怎么样的，好早日修改。但快速原型意味着提前跨入生命期的前几步，技术难度必然较大。

b) Program generator：对某些特定的应用，已经开发出了写出需求规格后即可自动生成程序的应用程序生成软件，最近受知识工程的影响，正在形成以知识库为基础的通用生成软件。这样的软件和原型技术一起在确立Throw-away型生命期上发挥很大作用。

### (5) 集成化的软件工程支撑环境的开发

这个新课题的提出是针对“S”类生命期〔注〕特别是Waterfall型生命期模型的缺点而提出的。Throw-away型开发方法是弥补“S”类方法的缺点并和Waterfall

〔注〕“S”类生命期由按阶段写出说明书的方法而来的。



型方法结合使用的较好手段。但目前正在着手研究的方法是一种从系统整体性考虑的接近人的思维方式的抽象方法。这种集成化的软件工程支撑环境从统一概念、统一接口、统一文件和介质管理方式来着手。避免用人工分别管理生命期各阶段的中间文件造成的整体性和各阶段联系性不强的弊病。新的方法“自顶向下”和“由底向上”互相结合,互相交织的方法。也就是说,该方法是把开工前的需求概念的描述和实现领域的实体描述结合起来的方法但又不是以“工具箱”方式松散地组织在一起(象unix工作台那样)的方法。总之,集成化的软件工程支撑环境的出现,必然会大大提高软件的质量和开发效率。但由于这种方法仍属设想,还未到实用阶段,所以目前各国所采用的软件生命期模型的方法,仍将延用很长时间。

#### (6) 软件维护

软件维护是软件开发的后续工作,它占用了软件总开销的大部分费用。可维性是现代程序设计技术的新要求,它可分为程序的度量,程序正确性和可维性理论和维护工具等。为了提高可维性。应加强软件系统说明书和程序的易读性。养成结构化、模块化设计的好习惯,养成在程序行中加注释以及在程序段前加概要说明的习惯,这样开发的软件,易于以后自己 and 他人阅读和修改。

目前,国外正在致力于提高软件维护效率,美、日等国相继开发了软件维护设施,以及软件维护支撑环境,例如:维护工具箱(如定时检查磁盘环境,定时拷贝等),维护专用工作站操作系统等。

### 2. 软件的经营和保护

1980年美国的软件营业额达到了26亿美元(日本为8亿美元),1985年美国软件销售额已超过200亿美元,1987年可达500亿美元以上,软件企业数千家,从业人数达百万人。预计今后一段时间内,每年增长30%以上。80年代后,软件已成为计算机工业的主

导行业,也是经济发展的重要行业。

软件产业的发展,随之而来的问题就是软件保护问题。由于软件是一种易于复制的产品,产品一经别人复制,研制开发人员所投入的全部精力、时间和费用将轻而易举地被别人占有。这样就使得研制开发者的权益没有保障。在这样的情况下,软件很难投入市场。虽然开发单位和销售单位想出一些软件加密的措施,但加密所投入的精力和费用有时占相当的比重,而且有的很容易被解密。所以,大量投入人力去研究各种软件加密的方法尚不能从根本上解决问题。为此,发达国家大都制定了各种软件保护法,并设立了相应的机构。通常的软件保护有版权法、专利法和合同法等等。

### 3. 软件管理

从软件的层次上讲,软件管理包括国家或地区级的行业管理、部门级的企业内部管理和单个项目的管理。从软件开发和流通这个角度上讲,又可分为“软件开发生产管理”和“软件产品管理”。

由于软件开发过程是智力劳动的过程,也就是说是思维密集的过程。产品一经开发出,其制作(复制)是极其简单的。所以它和硬件生产有显著的区别,因而在国外软件生产管理大都集中在软件开发生命期的各个阶段上的管理工作,而且各阶段的中间产品是以文档形式出现的。各软件企业间都有自己的一套文档推进方法和标准。互相之间很少一致性。而且这些文档形式,以及软件开发方法和开发工具以及所谓“支撑环境”的概念都在不断发展,而且有相当一部分的手工形式和人的因素。这就体现了软件管理的特殊性和复杂性。

除此之外,软件产业是一个在七十年代才开始形成的新型产业,基础性的管理工作尚需完善,完全自动化管理更是遥远的事情。由于用户在需求调查和分析方面,很难一下子把问题讲清楚,加之不同开发者水平

之间相差甚远。据SDC公司的调查统计,对同一作业,不同人员编写和调试所用的时间相差20多倍;而编出的程序执行效率又相差10多倍。其它公司的调查结果也大体类似,这说明了软件管理的难度。所以软件管理工作是一个多因素的综合管理。它既有标准化的正规部分,又有手工式的人为因素的管理部分,既有一种严格的质量控制、产品检验、分析和统计估算及各阶段的技术规格要求和约定,又有一套灵活的可调整的技术来协调各部门间不同人员之间工作的一套方法。

目前软件已渗透到各个行业,这又增加了软件行业管理的广度和深度。由于管理信息系统的发展,决策信息的发展,计算机服务业的发展和市场信息的形成,以及人才培养、技术交流、软件合作、软件出口等等都在软件行业管理范围内,所以从软件管理方面范围日趋扩大,工作日趋重要和复杂。

软件产品管理如软件登记、专利、定价、评价、发布等是软件商品化的必备条件。软件产品管理合理化可以使开发者能了解产品的行情,用户需求度,并避免重复劳动。用户可以随时了解到所需产品的功能、特性、价格等信息以及何时何地能购买该类产品。同时国家可以掌握软件产品情况和用户需求情况,便于定向投资和制定开发计划,便于软件产品评优活动。便于制定软件标准,便于软件登记,给开发者以合法性保护,避免互相抄袭,能较好地解决各种争端。有利于信息交流,掌握市场行情,促进国家软件产品发展。

#### 4. 软件质量保证

软件质量保证是用系统方法评审或检查软件产品是否满足需求说明的一系列工作。详细情况参阅软件评价部分。

#### 5. 软件标准化

所谓标准,就是对工业生产或工程建设方面的组织、加工、产品、质量以及所施加

的方法、采用的工具的统一规定,它是生产部门和流通交换部门共同遵守的依据。这种标准的形成是现代大生产的基本条件之一。软件开发标准也是软件产业中必备的条件,是软件象工业生产那样开发工程化的基本内容之一。

当前国际上有一个标准化组织ISO,计算机信息处理行业的标准组织是ISO下属的TC97技术委员会,其中的SC7是相对应的软件工程化标准,在我国已有和ISO/TC97/SC7的对口组织。

软件工程标准的建立是软件工程成熟的重要标志。它不仅关系到软件产品开发和维护的质量和效率,而且关系到软件性能价格比的提高,软件生命期的延长和应用范围的扩大(可移植性)。目前,日本软件企业的系统开发及维护业务标准有三十多种;系统开发技术标准有55种及通用的软件标准30多种。

从目前发展情况分析,在近3~5年内,软件工程标准化仍然是配合软件生命期各阶段不断完善,如用户需求说明标准,系统设计标准、系统开发标准、软件质量标准、检查测试标准等。从长期目标来说,软件工程向统一化集成化发展,那么软件工程的标准化必然要配合其发展由软件生命期各阶段分别开发的标准到整体化开发方法和工具发展。软件生产的产品象元器件一样作为标准部件来搭成大型系统。那时的软件生产流程,生产方法和工具,开发环境的标准与软件产品的标准化就融为一体了。

#### 6. 软件评价

软件评价包括系统开发前的评估和系统开发后的评价。对系统开发前的评价,我们把它放在项目的可行性分析中,在此不加详述,以下只对项目完成后的评价工作进行概述。

对于每种技术或项目,我们需要一种评价总体质量和有效性的手段,这种手段是测

试手段。所以软件评价问题,实质上是对软件产品的各种指标的度量问题。评价的目的是准确地把握产品的质量,促进软件高效生产。除了对投入费用和系统效益进行估算外,重点评价以下部分:

### (1) 评价标准

国外把软件的质量标准(品质特定)定为以下14种:

正确性:软件实现的功能达到原设计要求。

可靠性:在规定的条件下、规定的时间内不发生错误的效率。

可维性:查错纠错及在规定范围内进行变更的容易程度。

可移植性:从一个计算机系统到另一个计算机系统的容易程度。

易用性:用户掌握和使用的容易程度。

可扩充性:软件扩充功能的容易程度。

安全性:软件防止自身受到破坏的能力。

可测性:建立测试数据以及用这些数据测试评价软件的容易程度。

可理解性:对程序的理解容易程度。

有效性:满足用户要求及处理能力的程度(和正确性接近)。

完整性:冗余度的多少。

灵活性:和可维性类似,易于更改变动。

重用性:再次使用的程度或多个系统重用情况。

互通性:和可移植性等类似。

注意:以上指标是有冲突的,如灵活性提高必降低完整性等,所以实际评价时一定要抓住关键因素。

### (2) 评价方法

第一类方法是定性方法,比较容易做。

a) 用黑盒法测量系统功能强弱。

b) 用白盒法测量结构

c) 随机抽测

d) 检查研制过程的合理性

e) 检查文档的规范性

f) 同类产品比较

g) 根据应用情况、实用性、经济效益来评价。

h) 专家评审

第二类方法是定量评价法

a) 根据专家评审和测试结果采取Fisher方法(利用质量标准积累资料 and 判别函数)确定软件等级。

b) 动词法

根据语句及操作的控制及执行动词多少来评价软件性能质量(对应用软件适用)。

c) 概况法

制定一些标准测试方法,模拟任务变量,直接测出软件各项性能质量指标,综合分析出总体性能质量。

d) 化整为零法

根据软件结构,化整为零,由各组成部分的性能质量指标,使用频度综合归纳出软件的性能质量。

国外还有一种评价方法叫做FCM 分层简化法。将软件质量的判别分成:质量因素(Factor)、判别准则(Criteria)、实际测度(Metrics)三个不同的层次逐步细化展开(最后细分成11个指标进行评价)。

### (3) 评价注意事项

a) 评价人员一定是非设计和开发者。

b) 在实际评价时,各种质量指标有的有矛盾,如提高灵活性必造成整体性降低等,所以实际针对一具体系统要根据它的目标抓住关键因素进行评价。

## 7. 软件成本,成本—利润估算及成本—效果分析

到目前为止软件成本估算尚未形成一门科学,在应用方法上,大体有以下几种:

(1) CoCoMo(Constructive Cost Model) 模型,输入某些参数来估算



工程的进度和成本。

(2) 专家预测, 分别预测而不是集合在一起, 然后对过于分散的预测进行进一步的咨询。

(3) 模拟预测: 用类似的项目来比较

(4) 自顶向下或自底向上的预测由全局总成本推出各单元的成本, 或者反方向推测。

成本—利润模型:

$$V_1 = \sum_{t=1}^n (B_{1t} - C_{1t}) / (1+i)^t$$

其中:  $V_1$  为纯利润

$B_{1t}$  为毛利润

$C_{1t}$  为投资量

$t$  为利息

$(1+i)^t$  为通货膨胀因子

成本—效果(满足用户需求度)分析成本—效果关系如图2所示

1) 分析关键:

- 在给定的效能级上, 寻找最小开发成本。
- 在给定投入资金级上寻找最高效果方案。

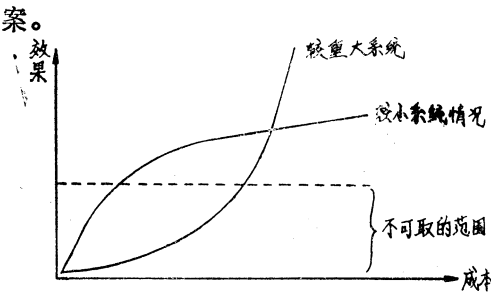


图2 成本—效果关系

2) 分析模型

$P_1 V_1$  为对1项活动的效果

$$\sum_{i=1}^n P_i V_i = n \text{项活动的总效果}$$

其中:  $P$  ~ 某个百分比。

$V$  ~ 项目的价值。

## 8. 软件政策<sup>[注]</sup>

由于原材料、能源和计算机技术是国际上极为重视的经济发展三大支柱, 所以发达国家, 特别是美国, 对软件产业的作用给以特别注意, 制订了相应的扶持政策, 大力支持计算机公司开展软件工作, 使美国成为世界上最发达的软件强国。为了维护自己的实力地位, 发达国家和发展中国家从自己的国情出发分别采取了内向型、外向型和内外兼向型的发展对策。

发达国家主要以内向型为主, 其对策有:

(1) 把发展软件产业列为国策, 并在财政金融上以投资、拨款、信贷等多种方式对软件行业予以支持, 如:

美国为了适应第五代计算机的需要, 开发Ada语言, 1983年由国防部进行了7年投资2.5亿美元的软件研究开发计划。1985年拨款1亿美元建立国家软件机构。

法国1982年7月用于软件和信息处理服务业45亿法郎。

西德于1967~1979年三个五年计划中拨款18.53亿马克用于扶持软件产业。80年代后, 把扶持新兴技术和新兴产业作为西德的主要课题。为了加强软件市场竞争力, 努力开展软件标准化。

英国的情况和法国、西德大体相同。1982~1985年拨给软件开发补助金2500万英镑。为了迎接日本的第五代计算机的挑战, 1983年9月提出五年间计划投资3.5亿英镑用于研究开发软件工程、人机接口和人工智能等。

日本的软件产业约比美国落后十年。但日本早就意识到软件产业的现实意义, 从70

[注]: 以上只是对软件政策的概述, 详情可参阅天津市科委软件管理政策课题组的“我国计算机软件商品化对策研究报告”以及《计算机世界》1986年的有关文章。

年代起耗资数百万日元进行软件产业投资，提高软件生产自动化水平，促进软件商品化。

为了解除80年代后的第二次“软件危机”，日本通产省联合其它多个公司和企业，共同赞助投资，推出SIGMA计划。根据该计划，1985~1989年投资300亿日元进行系统开发。第二期工程完成，1990年将提供带500个终端的大型系统为用户服务，其内容有：①提供丰富的系统软件和系统开发功能；②提供软件开发支援工具群；③提供软件技术情报，教育情报、程序情报等数据库服务功能；④提供特殊领域的技术服务如通讯服务、远程录入文档和消息传送，多窗口工作站等等。

(2) 设立国家软件管理机构，并颁布有关法律进行立法和建立相应的制度。

(3) 制定软件研究开发计划，设立软件研究机构，不断开发新技术。

(4) 提供大型系统为用户服务。

发展中国家，如新加坡、印度、南朝鲜等国，根据自己的国情，采取了“外向型”软件开发政策，即在资金困难、技术落后的情况下，发挥本国人才资源的优势，争取利用国际市场平台，采取与外资合作，打开软件推销局面，其对策是：

1) 把发展本国软件产业作为基本对策，设立国家管理机构和经营机构。

2) 制定经济立法，对软件产业施行收税等优惠政策，并免除法人所得税。

3) 引进外资，引进技术，建立软件产品输出加工区，对给予技术转让的外国企业以优惠措施。

4) 借助外国公司，建立相应机构培训高级技术人员。

特别值得提出的是印度，它区别于欧美。印度凭借本国人力资源的优势（印度受过高等教育的人数较多，并且由于历史原因普遍精通英语），多数受外国公司雇佣，或

采取与外资合作，打开软件推销局面很有成效。

84年10月甘地任总理以后又推行了新政策，制定了一系列政策和进出口法令。为了振兴软件产业，新政策规定，凡是以出口软件为目的而进口的计算机、软件和软件工具，消减其进口税。出口产品则在金融上给以优惠。印度从70年代开始出口软件，现在已是具有一定竞争能力的软件出口国。产品远销中东各国。1979年软件出口额为440万美元，1981年为1440万美元、1985年为3千万美元，预计1987年近1亿美元。预计在80年代末出口总值达30亿美元，现在印度纯软件咨询及服务机构有400个，有关组织500多个，且有6个软件公司在美国、澳大利亚、欧洲、新加坡及中东建立了业务基础。由48个软件出口商组成的印度出口工程促进理事会，正为软件产业寻找合适的市场。

### 三、国内情况

由于十年动乱，软件工程的思想到70年代末才开始传入我国，所以在软件产业方面，至少落后于欧美先进国家15年以上。我国的软件产业刚刚处于萌芽状态。软件商品化的方向正在被人们逐渐认识，个别单位曾做过试点，但还没有较成功的经验。可以预测，在1990年以后，我国的软件产业将有实质性的进展。就目前来说，我国软件生产有以下三个特点：

1. 软件生产处于无明显层次的低水平手工劳动，自动化生成手段比较弱，产品质量不够高。

国内的软件研制开发，绝大多数还处于小团体作坊型，或是一些小团体的联合体形式。各单位之间及单位内部的分工协作尚不够完善，不够标准化。软件生产分工不够科学、效率低、出错率高，检查手段不完善、不严格。大部分应用软件开发完没有很好地

投入运行,就送去鉴定,因而有相当多的软件系统处于病态。长期被采用的并产生较好经济效益的应用软件为数甚少。软件生产流程一般是指定专人负责全过程,没有分好层次,工作强度不均,耗费时间长,而且没有一套科学化的供传递的设计材料。生产的软件通用性差,运用面窄。再加上软件资料不标准,不完善,很难被别人理解,因而也不可能被其他人修改、移植,扩大应用范围,也没有科学的评价办法,鉴定会往往流于形式。另外,软件环境不理想,软件工具落后,软件方法不妥,软件人员技术训练不够严格以及对软件工程化认识不足也是造成上述结果的直接原因。

## 2.大量进口计算机没有发挥应有的作用

据统计,我国至少有1/3的计算机没有发挥作用。比如,我市的110种型号的机器6000余台中,40%用的很差或闲置,日开机量不足4小时。有的领导不愿意花钱购置必要的软件 and 培训软件人员。至于软件产业,多数计算机软件开发单位还只是流于口头上。

3.软件管理方面的理论研究刚刚起步,实践方面正处于探索之中,尚无具体政策和可行的办法,但是也出现了可喜的苗头。

软件产业化方向是近几年才提出来的。在教育部支持下,已投资了数百万元筹建武汉大学的软件工程研究室。1983年5月,计算机工业管理局批准在北大二分校建立北京软件工程研究中心,同年,国务院电子计算机和大规模集成电路领导小组,正式将软件工程列入国家科技攻关项目。教育部和电子工业部曾列出19项攻关项目。其中有软件工程支撑环境、软件工具、Ada语言及其支撑环境、软件维护工具、软件可靠性定量估计,异种语言自动转换、测试工具等项目。国家科委在1985年指示上海科委研究软件产业化问题,指示天津科委研究商品软件管理

政策,探讨我国软件商品化对策。由电子部和司法部组成我国软件保护法草案编辑组。国家出版局已写出软件版权草案,提交人大常委会讨论。计算机管理局组建了软件产品登记中心,中软公司及中软协会组织召开了全国软件工程经济学讨论会,讨论软件成本及计价问题。

综上所述,说明围绕软件产业出现的一系列方针政策,在我国正在逐步形成。另外,一些大型软件的设计中,不少已或多或少地采用了软件工程的方法和思路,取得了良好的效果。科学院研究所研制的XYZ/F软件开发工具也取得了显著的成果。因此,现在着手考虑软件工程方面的问题,时机已经成熟。

## 参考资料

- 1.国家经济信息系统设计与应用标准化规范  
1986·3
- 2.我国计算机软件商品化对策研究报告  
——天津市科委软件管理政策课题组
- 3.漫话软件产业 朱三元 卢祥一 应明  
《计算机世界》1986
- 4.印度计算机产业的发展与政策 陈厚云  
《计算机世界》1987.5
- 5.S类软件开发方法的变迁 何克清  
《软件产业》1986.7
- 6.国外软件工程概况 贾耀良 1984
- 7.国外软件工程动向 朱三元 1984
- 8.《MANAGEMENT INFORMATION SYSTEMS》SEON H. CHO WANG  
INSTITUTE OF GRADUATE STUDIES 1986
- 9.《ソフトウェア工学演習》  
松本吉弘著 1984·8
- 10.“软件开发技术” 潘锦平著
- 11.“软件工程” R.S.pressman著  
唐世谓 方裕译



# 再谈被删除文件的恢复问题

崔秀起

(北京大学)

**编者按** 本刊曾于1987年第4期登载了“微机应用中被删除文件的恢复”一文，其间收到马鞍山钢铁设计研究院计算机中心王德银，江苏省泗洪县统计局曹勇以及上海市中山西路1521号计算机软件实验室等同志来信，均提到了该文所介绍方法的局限性。对此，我们作了验证，并与作者及时联系，作者对此也进行了实践与探索。现发表作者的改进稿，以期能将实用的方法介绍给读者。欢迎读者与作者之间的讨论。

为发展计算机事业，活跃学术气氛，本刊愿作读者和作者相互沟通的桥梁。

本刊一九八七年第四期登载了笔者《微机应用中被删除文件的恢复》一文，介绍了一种简单易行的恢复方法，但对于大于1K的文件恢复就遇到了困难。因为FAT尚未恢复，后续簇链找不到。下面就FAT的使用方法和文件恢复步骤介绍如下。

## 一、FAT的使用方法

对于一个较长的文件，MS-DOS把两个或者更多的簇分配给它，而该文件占用的簇链被映象在FAT中。但是FAT中的实际信息并不是簇号的直接映象，而是按特殊规则分割而产生的变型映象。这里介绍两种读FAT的实用方法。

### 1. 规则法

(1) 从文件目录登记项中得到该文件的起始簇号，把该簇号乘以1.5(因为每一个FAT登记项是1.5个字节长)，所得结果取整，这个整数就是该簇号在FAT中的首字节位置。

(2) 从该位置开始取一个字，注意第一个字节为低位，第二个字节为高位。

(3) 若已知簇号为偶数，则保留该字的低12位，否则保留高12位；

(4) 若保留的12位是FF8~FFF，则表明已知簇是文件的最末簇，否则为该文件簇链中的下一个簇号。例如，盘上有一文件ERSOIO·BIN文件，目录项打印文本如下：

```
1000:0820 45 52 53 4F 49 4F 20 20-42 49 4E 27 00 00 00 00 ERSOIO BIN*****
1000:0830 00 00 00 00 00 00 2E 00-21 00 13 00 A4 19 00 00 ***** !**$**
```

由目录项可知，文件簇链的首簇号是13H，文件长度是19A4H。

文件占用簇数 =  $\min(\text{整数} > \text{文件长度} / \text{每簇字节数}) = \min(\text{整数} > 19A4H / 400H)$

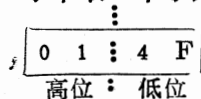
= 7

文件首簇号在FAT中对应位置是  $13H \cdot 1.5 = 1CH$

根据已求出的首簇号在FAT中的位置及文件占用簇数，打印出部分FAT，文本如下：

1000:0010 C0 00 0D E0 00 0F 00 01-11 20 01 FF 4F 01 15 80  
 1000:0020 01 17 80 01 19 F0 FF 1B-C0 01 1D E0 01 1F 00 02

从1CH字节取一个字为



本簇号是奇数，保留高12位为014，14就是该文件下一簇的簇号。

以此类推，便可推出文件簇链及其在FAT中的位置。这种方法一般不采用。

## 2. 分割法

(1) 用文件簇号乘以1.5，然后取整，得到的整数作为该簇号在FAT中的字节定位位置。

(2) 若已知簇号为偶数，则从定位字节开始向后取三个字节，设为A、B、C，其中每个字节中的高四位和低四位分别用下角标H、L表示，然后按图1所示分割法分割拼装，则可求得文件下一簇的簇号。

(3) 若已知簇号为奇数，则定位字节作为B字节，向前取一字节A，向后取一字节C，然后按图2进行分割拼装，可求得下一簇的簇号。

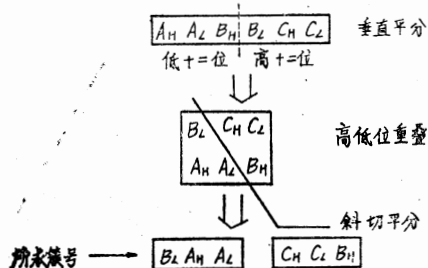


图1 偶数簇号分割法

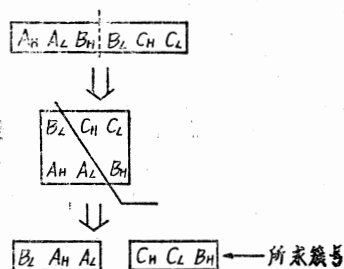


图2 奇数簇号分割法

## 3. 对被删文件空簇链的恢复

(1) 由文件目录项查得文件首簇号，根据奇偶性，在FAT中取三个字节，按分割法找出第一个空簇。

(2) 根据文件目录项中提供的文件长度，由第一空簇向后找出所需要的空簇数。

(3) 将相邻的后一空簇号填入FAT的前一空簇位置中，最后一个空簇位置填入FFF。至此被删文件的FAT已被恢复。

## 二、恢复误删文件实例

在双面双密度软盘上由于不慎，将磁盘格式化程序FORMAT.EXE误删除了，可按如下步骤恢复：

双面双密度软盘盘区分配情况如表1所列：

表 1

	引导区长度	FAT区长度	目录区长度
扇 区 数	1	4	7
逻辑扇区号	0	1~4	5~11

其中每扇区512字节，二个扇区为一簇。

由表中可知FAT区和目录区的逻辑扇区为1~11 (0H~BH)，执行如下操作：

A>DEBUG ↵

-L 1000:001B ↵；把A: 盘1~11号逻辑扇区信息装入首地址为1000:0的内存区中

-D 1000:800 1400 ↵；显示 5~11逻辑扇区的目录，查找被删除的FORMAT·EXE文件目录项

文本如下：

```

1000:0850 00 00 00 00 00 00 68 60-6A 07 1A 00 55 3E 00 00 ..... h j... U....
1000:0860 E5 4F 52 4D 41 54 20 20-45 58 45 20 00 00 00 00 eORMAT EXE.....
1000:0870 00 00 00 00 00 00 C5 05-21 00 2A 00 80 11 00 00 .....E.!.*......
1000:0880 46 44 49 53 4B 20 20 20-45 58 45 20 00 00 00 00 FDISK EXE.....
1000:0890 00 00 00 00 00 00 F3 02-21 00 2F 00 80 10 00 00 .....S.!. /.....

```

↑
↑
↑

文件删除标志
首簇号
文件长度

由目录项可知，文件簇链的首簇号是2AH，文件长度是1180H。

文件占用簇数 =  $\min(\text{整数} > 1180H / 400H) = 5$

文件首簇号在FAT中对应位置是  $2AH \cdot 1.5 = 3FH$

### 1. 恢复文件名

-E 1000:0860 ↵

1000:0860 E5 46 ↵；将删除标志“e” (E5) 恢复为“F” (46)

### 2. 恢复FAT

显示并打印出1~2逻辑扇区中包含文件FORMAT、EXE首簇号3FH及簇链数的FAT部分：

-D 1000:0000 00BF

```

1000:0000 FD FF FF 03 40 00 05 60-00 07 80 00 09 A0 00 0B 1 .....②.....
1000:0010 C0 00 0D E0 00 0F 00 01-11 20 01 FF 4F 01 15 60 ②.....O...
1000:0020 C1 17 80 01 19 F0 FF 1B-C0 01 1D E0 01 1F 00 20 .....P..②.....
1000:0030 21 20 02 23 40 02 25 60-02 27 80 02 29 F0 FF 00 1..②.%. ....
1000:0040 00 00 00 00 00 00 03 31 20 03 33 F0 FF 35 60 .....1 .....
1000:0050 03 37 80 03 39 A0 03 3B-C0 03 3D E0 03 3F 00 04 .7..9.:②.....
1000:0060 41 20 04 43 40 04 45 60-04 47 80 04 49 A0 04 4B A.C②.E.G.....

```

↑

文件簇链首字节位置3FH

由FAT中得到如图3如示：



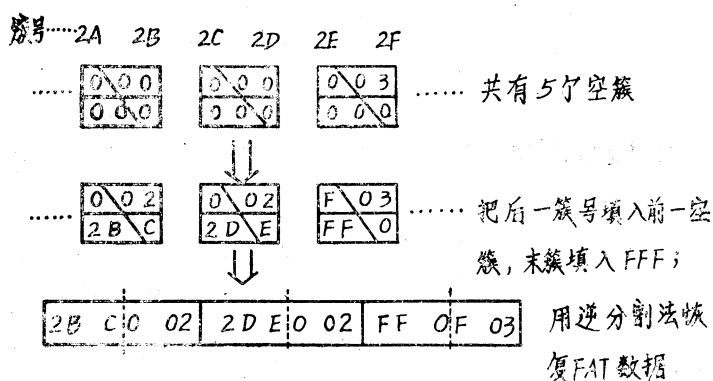


图 3

将数据写入FAT:

-E 1000:3F ✓

⋮

，将数据逐一写到FAT中

至此文件的FAT恢复工作结束。

### 3. 把恢复好的文件目录项和FAT表写入A盘。

-W 1000:0 0 1 B ✓；把内存中已恢复的信息写到A；盘的1~11号逻辑扇区

-Q ✓；退出DEBUG，返回DOS系统

到此为止，对软盘文件FORMAT、EXE的恢复工作做完了。

由于被删文件簇链的FAT项区域，可能含有n个空簇，这样有可能使恢复的文件多了n簇无用信息，而少连了n簇的有用信息。通过查看已恢复的文本，可发现这种错误。若发现错误时，可重复上述恢复步骤，把n个多余的簇再改为000，并把簇链向后延伸n个空簇，恢复FAT，写入A盘，再进行复查，直到完全恢复为止。

实际上，由于MS-DOS文件是按最近单向存放原则按簇链存放某一文件的，使得被恢复的文件中含有多余信息簇的可能性是极少的，但为以防万一，特别是对重要文件的恢复，还是慎重为好。

### 参 考 资 料

1. 《IBM-PC磁盘操作系统》
2. 《微型计算机IBM-PC的原理与应用》，张福炎等编著，南京大学出版社，1984
3. 《计算机操作系统原理》，北京大学徐联舫、柳纯录编著，机械工业出版社，1985，12
4. 《IBM-PC磁盘文件系统的特点分析和使用的技巧》，白剑华，“小型微型计算机系统”1986，12。

## Sun-3工程工作站的系统结构 (续3)

### 二、软件体系结构

#### 5. 编程环境

Sun的编程环境是建立在UNIX传统之上的,它鼓励使用小模块,以便容易地链接分立的程序块以形成新的功能块。Sun的编程环境包括C, FORTRAN-77, Pascal和Common Lisp,带有优化的编译程序,调试程序和程序库。系统接口易为应用程序员所使用;分层的模块化使得能够在各层次上访问Sun的软件。SunView图形用户接口在SunPro上增加了直观的接口及鼠标的交互作用。

(1)语言 Sun的标准语言结构见图1。Sun的语言策略是实现尽可能多的良好语言工具,并且直接支持最通用、最有影响的语言。Sun的语言可充分地访问系统,图形,网络 and 用户接口。SunPro是一种多语言间的综合程序开发环境。它为所用的许多语言提供了与Sun系统软件一致的用户接口。优化的各种语言的编译程序产生简洁而有效的

目标程序。

Sun目前提供C, FORTRAN-77, ISO Pascal, Modula-2, 和Common Lisp。C是系统语言,它“属于”UNIX。ISO Pascal可作为应用工程方面的一个标准编程语言。为系统工程的需要,Sun把系统编程的扩充加到了它的Pascal中, FORTRAN-77是为数字处理和科学编程选用的工业语言。Modula-2提供Pascal的类型检查和控制,同时通过提供对低级系统的访问功能和为大的系统设计提供模块化结构来弥补它的很多不足。人工智能(AI)的重要性使Sun采纳了Common Lisp。为了在Sun工作站上进行软件开发,除了C, Pascal, FORTRAN-77和Common Lisp以外,很多其它语言已可由第三厂家提供。这些语言包括Prolog, Ada, APL, BASIC, COBOL, PL-I及其它。

1) C, FORTRAN 和 Pascal C, FORTRAN 和 Pascal 语言是Sun软件的标准成分。Sun为这些语言实现的编译程序产生一个公共的中间代码,并使用公共代码

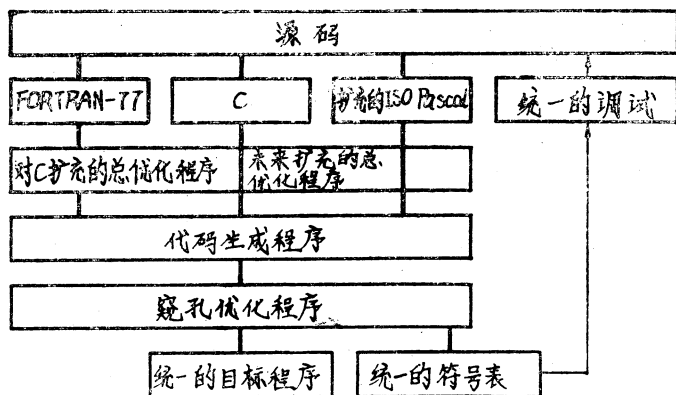


图1 Sun的标准语言结构

生成程序和窥孔优化程序。Sun已经为FORTRAN-77研制了一个总的优化程序，将来把它移植到C和Pascal编译程序上。

三个编译程序都产生为公共调试工具所用的统一的目标代码和统一的符号表。这一方法有三个主要的先进性。

- 可交叉调用 程序员可以用一种源语言写程序，它可以调用另一种语言写的模块或库程序。例如，C程序可以调用FORTRAN-77的数值计算程序库，FORTRAN-77程序能够利用C的例行程序与系统交互。

- 容易移植 当新的硬件技术出现时，Sun能够移植它的编译程序，并且通过写一个代码生成程序及窥孔优化程序使编译程序适用于新的处理器。用户能够从使用新的硬件和兼容的软件工具中迅速收到效益。

- 容易实现新的编译程序及工具 实现一个新的编译程序只需要在代码生成程序上增加一个新的前端处理程序。软件工具就更容易与新的语言结合。

Sun也提供汇编语言编程及调试实用程序。

2) FORTRAN和Sun-3 在Sun系统软件的2.0版本和3.0版本之间，对编译程序性能改进的重点主要放在有效地使用MC68020和MC68881指令系统，为FORTRAN-77编译程序建立一个新的全面的优化程序，对FORTRAN-77进行更快的编译及为语言内在操作产生更好的内码。

Sun已经全面地修改了由C，FORTRAN-77及Pascal编译程序所共享的公共代码生成程序，以便使用MC68020增加的功能（与MC68010比较而言）。这些修改包括对长字的乘除法指令，位字段上的操作指令，及提供几种新的寻址方式。代码生成程序能够产生供MC68881使用的代码，并提供与IEEE P754标准完全一致的浮点运算，以便使浮点运算具有更好性能（对于单、双

精度运算速度在700K Whetstones以上，大约80Kflops左右）。

一个全新的用于FORTRAN-77编译程序的总优化程序使代码比非优化码性能改进10%~80%。进行的优化包括常数合并，短路布尔估值，循环不变码的运动，归纳变量的消除，本地和全局公共子表达式的消除，本地和全局的复制传送，死码的消除和全局寄存器的分配。另外，窥孔优化程序执行与跨距相关的指令分解，分支链接，机器习语的使用及消除多余的存储和装入。

- Modula-2 Modula-2最初是为写操作系统和设备驱动程序而设计的一种系统编程语言，但通过把系统语言的实力与高级结构语言的控制相结合，它就适合于大的应用课题并用在教育领域及系统工作方面。正如名称所示，Modula-2为“模块化的”编程提供了一种结构。这一能力与由该语言提供的高度数据抽象一起，使Modula-2十分适用于大型的集体编程项目。

Modula-2也提供了一个高度的源程序可移植性。由于全部与系统相关部分都隔离在库程序模块中，故只需很小的努力就可以把一个程序移植到另一个环境中。在Sun的工作站上，不限于仅仅使用Modula-2的库程序。相反，Modula-2编译程序能够访问C的库程序或其它C的例行程序，包括那些带有可变长度自变量表的C的例行程序。编译程序还有特殊的模块，能对UNIX系统，Sun Window™，图形包及许多其它例行程序进行完全的访问。

因为Modula-2与Sun的C，FORTRAN-77和Pascal共享一个公共后端，所以它可利用Sun工作站上已有的全部浮点选件。这种安排给用户提供一个在竞争中取胜的高性能的IEEE标准浮点软件包。共享一个公共后端也使Sun系统的许多编程工具，例如dbx和dbxtool，调试程序和剖面图工具等得到利用。

### 3) Sun Common Lisp 象Sun

Common Lisp那样高性能、系统集成语言的进展将促使人工智能(AI)的语言及技术在商业上的使用得到迅速发展。

AI的基础是,计算机能够模仿以推论为基础的人类知识结构。灵活的数据结构对表示人类知识的多样性是必要的。符号编程的构思试图仿真人类决策的方式。象Lisp这样的语言提供这些工具。

Lisp是人工智能研究和越来越多的应用开发的基础。但至今为止,它的使用仅限于研究,缺乏标准的AI语言的定义限制了AI软件包的可移植性,因而限制了商业环境的可行性。由于缺乏与更多的传统系统和应用语言的结合,限制了Lisp访问系统调用, I/O程序库和其它语言通用的外部软件包。由于Lisp起源于一种解释语言,实际性能只可能在特殊目的的符号处理机上实现。Sun Common Lisp去掉了这些限制。

Sun Common Lisp与其它Sun语言很好地结合在一起。用Sun Common Lisp写的程序在交叉调用方面具有与其它Sun语言一样的潜力; Sun Common Lisp程序可以引用以C, Pascal和FORTRAN-77写的例行程序。Sun提供AI开发工具和Sun的编程环境,使Lisp程序能与Sun处理机,用户接口,图形和组网功能相结合。Sun Common Lisp编程环境提供了一种Lisp的解释程序和一个优化的编译程序。

后者是为得到与解释结果完全兼容的应用代码而提供的。为动态地检查和修改数据结构, Sun Common Lisp提供一个数据检查程序,此外还提供了交互调试程序,通过它,可对Lisp运行环境进行完全访问。Sun Common Lisp具有充分的功能打断,跟踪及单步的能力。Lisp程序能够与其它程序通过UNIX管道线进行通讯,外来的功能调用接口允许用户把经过编译的C, FOR

TRAN-77和Pascal 程序动态地链接到运行着的Sun Common Lisp程序中。

4) 性能分析 Sun操作系统提供一些用于监视软件性能的工具,其范围是从报告运行一个程序所需要时间的一条简单命令到一个提供程序的语句到语句分析的代码覆盖工具。

- Time Time是一条简单的系统命令,它报告执行一个程序需要多长时间。报告发布与时间有关的信息,如一个进程所占用的CPU的时间(秒),缺页数,及用千字节表示的程序所用的最大实际主存及未共享的栈空间。

- prof和gprof 通过对程序执行时间取样汇集成的统计信息, prof描绘了每个例行程序的时间耗费情况。prof也显示了子程序调用频率。

gprof是由prof升级得到的。gprof给一个程序的子程序分配时间,并且描绘一个程序的时间分布图。它详述了一个例行程序调用另一个例行程序或自我调用的时间数。

- Code Coverage tcov是一个程序覆盖工具,它增加了prof及gprof的分辨率,使之扩展到语句级,从而提供了对一个程序执行频率的极详细的分析。tcov也给出了实际正在执行的程序部分的报告,并指出还没有被测试的程序部分。

(2) 软件开发 UNIX操作系统在历史上成功的一个主要因素之一是为程序开发者提供了丰富的编程环境。各种编程的实用程序是体现这种灵活性的积木块;管道线, UNIX的Shell命令语言及容易换向的输入与输出是把这些积木块结合起来的粘合剂。这些特性在一个快速程序样品开发环境中是理想的。程序样品是由更高一层Shell命令与系统软件实用程序结合而建立的。Sun的整体软件开发工具和附加的成熟的用户接口加强了UNIX快速程序样品设计的方法。

面向Sun Pro 工具的编程环境给程序

# IBM PC/XT工业控制系统硬件及软件实现

赵育才 冯德田

(华东石油学院自动化系)

## 一、前言

随着计算机应用的推广和普及,建立以微计算为中心的工业控制系统,已经成为微计算机应用的一个极为广阔的领域。特别是IBM PC/XT个人计算机,价格低廉,以其优良的特性,完善的软件系统,作为办公计算机已被人们广泛的应用。怎样用其优点

为过程控制系统和控制理论实验研究服务呢?本文介绍的小型工业控制系统就是为了解决上述问题在IBM PC/XT上进行开发的。为使IBM PC/XT个人计算机成为工业控制机,需要解决以下问题:IBM PC/XT与过程之间连接问题——接口,以及系统软件实现问题。实时时钟控制、采样频率选择及外中断源等问题。下面介绍所做的工作。

员提供了统一的源码级的接口,它与使用的源语言无关。程序改变的快速周期可以通过dbxtool实现。dbxtool是编辑程序和以窗口为基础的接口合并成的一个符号调试程序。所有的编程工具都与SunView工具相结合,这样就简化了用户接口的结构。Sun的工具为程序员和用户提供一个公用的、直观的环境。

1) SunPro SunPro是Sun的标准语言的编程环境。对程序员来说,SunPro是不定方式和面向源码的环境的基础。在不定方式环境中,程序员不需要在程序研制期间,在编辑程序、编译程序、调试程序或其它程序开发工具之间移来移去。需要时,工作站用户可以自由地交替进行这些工作,而不必进入和退出每种工具。SunPro的dbxtool为用C,FORTRAN-77,Pascal写的程序提供综合的编辑,调试和测试功能。通过Sunpro,用户在程序开发过程中把源码理解为处理对象——调试就是处理源代码

而不是机器代码。

• dbxtool dbxtool是一个成熟的符号调试程序,它以大大扩充了功能的4.2BSD符号调试程序dbx为基础,是与SunPro独立编程功能相接的基本用户接口。通过用鼠标器代替键盘作为基本输入装置,dbxtool不需要输入变量,行号,断点和大多数命令。dbxtool的窗口给出正在调试的程序清单及有关程序状态的详细信息,这就对调试问题给出了两种性质不同的观察。

Sun有扩充dbxtool及基本的dbx作为强有力的调试工具。新功能包括多进程程序和已经运行着的进程及UNIX核本身的调试。

SunPro也支持管理着全盘软件工程项目的管理活动。这些活动包括源码控制,配置管理及项目成员之间的相互通讯。

(未完待续)

天津市电子计算机研究所

袁野 汪霞 编译 黄侃 校



## 二、硬件结构

要增强IBM PC/XT的功能，第一步是选择适当的硬件接口。要设计一接口板将该机总线驱动连到扩充机箱上，采用BCS 80系统机箱。该机与过程之间连接如图1所示。

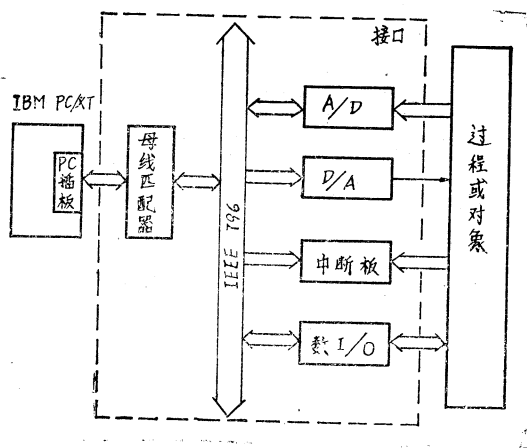


图1 系统框图

其中：

1. PC插板和总线匹配器是自己研制的，其功能是将IBM总线引出并驱动BCS 80系统接口。

2. A/D板——Inter ST711板

通道数目：32个单端输入通道

分辨率：12位（0.02%）

A/D转换速度：28KHz

三种可编程序操作方式，重复的单一通道输入，顺序通道扫描输入和随机通道输入，另外可外触发控制输入。

3. D/A板——BSBC724模拟量输出板

分辨率：12位

通道数目：共12个通道（每块板4个通道）

4. 中断量板——外中断源输入

用于实时处理外中断源请求，共16路。

5. 数I/O——数字量输入输出板

用于对象状态控制及检测，输入输出各16路。

为了提高数据转换的速率，该接口装置采用存储器对应输入输出方式。各功能板基本地址分配如下表所示。其特点是接口操作不通过I/O端口，不但数据转换速率有所提高，而且接口操作可使用IBM宏汇编语言的全部存储器指令。但接口需占用一定的内存单元。

各功能板基本地址分配表

功能板名称	基本地址（16进制）
ST711 A/D	8F70:0000
BSBC724 D/A	8F70:0008
数I/O	8F70:0040
时间定时器8253	8F70:0050
中断量输入板	8F70:0060

在控制中，实时时钟控制是必不可少的。为了解决时间控制问题，该系统在PC插板中增加了可编程的定时器8253。可通过定时器初始化程序设置8253工作方式并给各计数器加载，待到规定时间后，产生3级中断请求。从而可方便解决控制中的采样频率选择，定时控制等问题。

8253时间中断和外部中断源接在系统8259的三级中断输入上，中断向量号为0BH。其它中断源如A/D板的转换结果（EOC）接在系统8259的二级中断输入上，中断向量号为0AH。优先级排队用软件实现。

## 三、软件的实现

为了充分利用IBM PC/XT丰富的软件，该系统主程序用IBM FORTRAN 77 3.0以上版本编写。调用宏汇编语言系统子程序库来实现各类控制。接口各功能子程序用IBM宏汇编语言编写，系统功能程序（如中断处理程序）使用PC-DOS2.1版本提

供中断21H功能调用实现。下面介绍系统中断处理程序的编写及FORTRAN77调用子程序的编写。

### 1. 中断处理程序驻留及内存数据区的开辟

当计算机用于实时控制时，现场的各参数信息、故障处理需通过外中断源发出中断申请。CPU响应后转入中断处理程序加以处理，而中断处理程序必需驻留内存，不能被覆盖，并得到DOS操作系统的承认，否则中断无法响应。并需在中断处理程序中开辟一定的内存数据区供系统处理时使用。

中断程序驻留程序框图如图2所示。

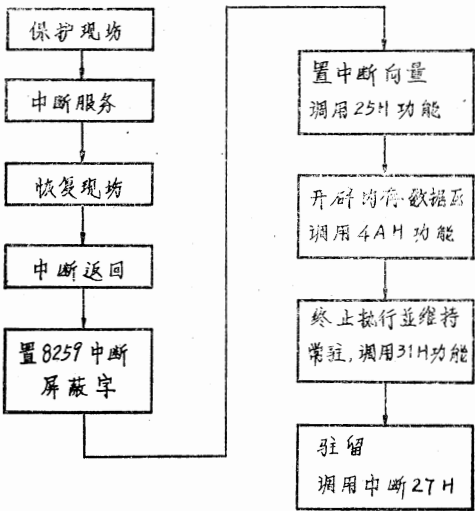


图 2

下面给出中断驻留程序。

```

START: IN    AL, 21H
      AND    AL, 0FBH
      OUT    21H, AL
      PUSH   CS
      POP    DS
      MOV    DX, 007H
      MOV    AX, 250AH
      INT    21H
  
```

} 置8259中断屏蔽字  
 } 置中断向量  
 } DS, DX含中断处理程序入口地址  
 } AL为中断处理程序所占用的中断向量号

```

PUSH CS
POP  ES
MOV  BX, 1000H
MOV  AH, 4AH
INT  21H

MOV  DX, BX
MOV  AX, 3103H
INT  21H
INT  27H
  
```

} 开辟内存区  
 } ES含有块的段号  
 } BX含请求开辟块的大小  
 } 终止处理保留驻存  
 } AL, 含出口代码  
 } 驻留

### 2. FORTRAN77语言调用宏汇编语言子程序

FORTRAN 77调用汇编语言子程序可用CALL语句。其格式为：

```
CALL Sname [(Variable list)]
```

其中：Sname是定义的汇编子程序名

Variable list是做为参数变量传递给子程序的。FORTRAN调用汇编语言子程序的关键是正确地传递参数。CALL语句传递参数的实质是将参数变量的基址偏移量地址推入堆栈，可利用宏汇编的伪指令STRUC建立栈区。

格式：

```

SSEG    STRUC
SAVEBP   DW?
SAVERET  DD? ; "段点"
          :
VAR - A  DD? ; 第一个参数的段值和偏移量
  
```

```
SSEG    ENDS
```

开辟栈区的大小决定Variable list变量的个数，汇编子程序中可利用BP寄存器在堆栈内寻址得到参数变量的地址，故在有参数传递的汇编子程序的最初指令应为：

```

PUSH BP
MOV  BP, SP
  
```

下面举实例说明参数变量在汇编子程序中的传送。大家知道，BASIC语言中有PEEK, POKE语句，使BASIC程序可直接访问指定的内存单元，使用十分方便。下面

# 介绍FORTRAN 77调用的PEEK POKE程序。

调用格式:

CALL POKE (unit, VAR)

CALL PEEK (unit, VAR)

其中: unit——地址偏移量

VAR——整型变量0~255

宏汇编程序如下:

SSEG	STRUC	
	DW ? ,	
	DD ? ,	"段点"
B	DD ? ,	第二个参数VAR的段值和偏移量
A	DD ? ,	第一个参数unit的段值和偏移量
SSEG	ENDS	
CSEG	SEGMENT	'CODE'
DGROUP	GROUP	DATA
	ASSUME	CS : CSEG, DS : DGROUP, ES : DGROUP, SS : DGROUP
POKE	PROC	FAR
	PUBLIC	POKE
	PUSH	BP
	MOV	BP, SP } ; 栈指针
	LES	BX, [BP] + A } ; 将参数unit的值传送到DI寄存器
	MOV	DI, ES : [BX]
	LES	BX, [BP] + B } ; 将参数VAR的值传送到DX寄存器
	MOV	DX, ES : [BX]
	MOV	AX, 8F70H } ; 将段基址8F70H送到ES寄存器
	MOV	ES, AX
	MOV	ES : [DI], DL ; 将VAR的值传送到指定的内存单元
	POP	BP
	RET	8 ; 清栈
POKE	ENDP	
PEEK	PROC	FAR
	PUBLIC	PEEK
	PUSH	BP
	MOV	BP, SP
	LES	BX, [BP] + A } ; 将参数unit的值传送到DI寄存器
	MOV	DI, ES : [BX]
	MOV	AX, 8F70H
	MOV	ES, AX
	MOV	DX, ES : [DI] ; 将指定的内存单元内容送DX
	LES	BX, [BP] + B } ; 将内存单元的内容传送给VAR变量, 带回主程序
	MOV	ES : [BX], DX
	POP	BP

```

    RET      8
    PEEK     ENDP
    CSEG     ENDS
            END

```

，清栈

本程序因段基址定为8F70H，在FORTRAN主程序中使用POKE, PEEK语句可检测控制本系统接口装置的各功能板。本系统的中断采样频率就是用POKE语句给8253定时器加载实现的。如想访问其它段基址单元，可改变段基址内容，用DEBUG进行调试。

调用汇编子程序传递多个变量时，进入堆栈的特定变量此时相对于BP的偏移量按下式计算：

相对BP的偏移量 =  $n \times (-4)$

其中n为CALL语句变量表中第n个指定变量。例如共传递6个参数，要获得第2个参数值，用下述语句：

```
LES BX, [BP] + [-8]
```

```
MOV AX, ES:[BX]
```

这样第2个整型变量的数值送到了AX寄存器中，如想把AX寄存器内容送回第4个参数变量时用如下语句：

```
LES BX, [BP] + [-16]
```

```
MOV ES:[BX], AX
```

(上接第27页)

## 四、结束语

前面以MAPLE和MACSYMA为例，对目前流行的计算机符号演算软件作了简要分析，并以简单的例子说明了MACSYMA的使用。人们还记得，二、三十年以前，计算尺和对数表是每一位工程师的必需品，可是曾几何时，它们已被袖珍电子计算

注意：在参数传递前，DS寄存器的内容不应改变，子程序返回前应清栈。

## 四、结束语

本系统是根据控制专业特点设计的，主要用于实验室实时控制及控制理论的研究，IBM PC/XT用于过程控制作了以上尝试，硬件和软件都不完善，有待进一步开发。我们在该系统上做了些工作，如多路连续数据采集，实时控制中的PID参数优化，系统辨识和自适应控制等。使用证明IBM PC/XT用于过程控制研究是可行的。但IBM PC/XT运算速度较慢，实时控制有时不能满足要求。在实际工作中，我们发现A/D板转换速度对控制系统影响很大，最好使用两块A/D板并行工作。另外系统软件可充分利用已有的控制软件包，根据需要加以修改进行组合链接即可成为系统软件。这样可以节省大量的编程时间。

器完全取代，成了年青大学生们眼中的老古董。电子计算机符号处理系统软件的研究和新一代强功能、32位微计算机的兴起，计算机代数的应用正在迅速推广。完全可以预期，计算机符号代数工作站在今后几年中将得到迅速普及，成为高等数学计算机辅助教学和科学研究及工程设计中重要的工具，从而大大地改变科学计算的面貌。

# 计算机符号演算软件的发展与应用

向延育

(中科院北京天文台)

**摘要** 本文对于计算机符号处理专家系统的发展和应用,作了简要的分析和展望。我们试图指出,随着MACSYMA, REDUCE等符号处理专家系统的成熟和广泛地应用于科学计算,将使新一代电子计算机在科学计算领域中的应用发生极大的变化,并且有可能进而影响传统的数值计算程序语言(例如FORTRAN)的发展。

## 一、

从现代电子计算机问世以来,科学计算就是计算机应用的一个重要领域。为了解决科学研究和生产实践中的各种问题,人们对计算机数值计算进行了长期、广泛深入的研究,发展了大量适用于不同问题的数值计算方法和数值计算软件。很多计算机语言(例如FORTRAN)的主要应用就是数值计算,乃至不少人甚至误认为电子计算机的功能就是数值计算。

随着近二十年来微电子技术的迅速发展,计算机得到了广泛的普及应用,人们认识到了大量的计算机的非数值计算及其在更广泛领域中的应用,并且力图以第五代计算机的观念,从人工智能的角度来理解和发展这类不同的应用。正是在这样一种背景下,非数值的计算机科学计算——计算机符号演算(有时人们也称为计算机代数)发展逐渐成熟,从研究走向实际应用。

所谓“计算机代数”,就是用计算机来做代数运算。计算机代数是随着现代电子计算机的发展,特别是六十年代以来人工智能和专家系统的研究发展而成长起来的。它的主要功能不再是传统的计算机数值计算,而是非数值的符号演算和公式推演。它的发

展,为计算机的应用打开了一个令人惊叹的新领域。事实上,有几种计算代数系统——即计算机符号处理软件,已经能相当成功地完成多种较基本的解析的数学演算,以至有人开始预言,计算机代数将成为明天的代数学“计算器”。人们研制成了多种计算机代数系统,而其中较广泛为人所知的有:MACSYMA, REDUCE和 muMATH, MAPLE等。它们是在早期符号演算研究,特别是在美国麻省理工学院所进行的广泛研究的基础上发展而来的。这些计算机代数系统,不仅已成功地应用于天体力学、量子物理和相对论等领域,而且也已经在聚变等离子物理,直升飞机浆叶的工程设计、电力网络分析、机器人控制,以及冶金学、声学、经济学等不同的领域中得到实际的应用。甚至有的大学(例如英国伦敦玛丽女王学院)已设置专门课程向大学生讲授计算机代数。

尽管都是在现代电子计算机上实现的程序系统,计算机符号代数却同传统的数值计算程序极不相同。它有很多独特之处。一般而言,功能较完备的计算机代数(例如MACSYMA),能对各种含有常数、变量和函数的代数表达式进行解析处理。计算机代数可以做微分、求积分、取极限、作因式分解、对符号表达式作简化和代换、将解析函数展开为泰勒级数、作拉氏变换或逆变



换、求傅里叶系数、解代数方程和常微分方程、作向量计算、张量计算和矩阵计算，也可以解线性方程组，非线性方程组以及微分方程组等等。而所有这些计算，都以精确的解析形式出现。正因为如此，计算机代数语言，同传统的主要面向数值计算的程序语言如FORTRAN，ALGOL也很不相同。

那么，计算机代数的优点在哪里呢？为什么人们在有了这样多种的程序语言之后，还要花费如此大功夫去研制新的计算机代数语言呢？一个容易看到的优点是精度方面。传统计算机语言的精度，直接受到机器字长的某种约束，不能很灵活地扩充，而计算机代数中主要进行的是精确计算，即便是数值结果原则上也可以很容易地达到几十位，几百位甚至更高位的十进制有效数字的精度，而同所使用计算机硬件字长没有直接关系。当然，计算机代数系统更主要功用是大大地节省人们的脑力劳动，使人们可以把注意力集中到更需要创造性的核心问题上，集中于研究问题的过程与算法。例如，人们常提到的一个例子是，十九世纪的天体力学家曾花费了20年时间对月球运行公式作推导，而所得到的结果在1970年人们用计算机代数只花20小时就又得出了。现有的计算机系统已经能够在许多方面使人们从重复性的繁复的解析计算中解脱出来。计算机代数还能通过其解析推演来对所研究的对象进行更深入的本质的分析，有时这是一般数值计算所不及的。例如，英国卡拉姆实验室的W·亚特等人，最近通过计算机代数系统MACSYMA进行推导和验证，分析某种等离子体不稳定性，发现在这以前用超级计算机作许多小时数值模拟所得到的结果竟然是错的。MACSYMA的符号推导，为简洁而可靠地进行这种分析，提供了很好的手段。从另一方面来看，计算机代数也能够帮助人们用不同的方式进行数值计算。例如，对于有一类在激光物理和流体力学中应用较广的含不规

则高维边界的椭圆型偏微分方程，必须将问题化为规则边界去积分；如对三维问题已经复杂到极难于手工完成相应的坐标变换并生成差分格式，美国的S·斯泰因伯格等利用MACSYMA符号处理软件，自动生成了相应的FORTRAN语言子程序，从而解决了单纯通过标准数值计算软件难于下手的问题。

目前的计算机代数软件，是在人工智能研究的强大推动下实现的，在很大程度上体现了专家系统的设计原则和特征。特别是较完备的MACSYMA，一般认为是最早的专家系统之一。在它的一些关键模块中，例如积分程序和有理式处理程序，大量运用了基于知识的推理，从而提高了符号计算的效率。由于这样地提取了专家知识，使得目前典型的计算机代数系统，一般而言超过了大学生的解析推演能力。

由于有这样一些传统计算机数值计算所难于取代的优点，使得计算机符号运算软件在最近几年开始有了更多的应用，它的研究也在迅速地扩展。在美国，在国家航空和宇宙航行局、海军、空军和能源部的支持下研制的MACSYMA，已经在许多大学和研究机关中得到推广并通过网络得到了更多的用户；许多利用计算机代数的研究计划正在开展；HP公司最近甚至顺应这种潮流推出了袖珍的具有一定演算功能的符号计算器；在欧洲许多大学的计算中心中，另一种广泛使用的计算机代数系统REDUCE已经成为提供给用户的标准软件。我国这些年来也有不少研究人员在从事这方面的工作，其中包括中科院数学所，物理所和理论物理所所做的工作。清华大学等单位所做的开发工作也正在顺利进行。

## 二、

符号代数系统的实现，要大量而自然地涉及表处理方式，所以大多数系统都由LI-

SP语言支撑。早期也有寄生在FORTRAN上的(例如FORMAC)。近年来亦有些设计者更着重考虑符号代数系统的可移植性,可维护性以及和硬件发展的协调性,而用C语言写系统的核心程序(例如MAPLE)。一般而言,除了一个较小的核心之外,系统都通过自展扩充而更完备。我们仅以MAPLE为例来说明典型计算机符号处理系统的设计与实现中的若干特点。

MAPLE是加拿大滑铁卢大学计算机系统从1981年初开始研制的一个计算机符号代数软件,它的核心只有5000行,因此有可能在典型的微型机上有效地运行。它的设计中充分地考虑了可移植性,其核心用C语言书写(或者说用BCPL类语言书写,其最早的一个版本是在Honeywell机器上用B语言写的)。MAPLE的系统设计目标是实现一种紧致而强有力的符号数学演算语言,并具备良好的可移植性和友好的用户接口。

为了提供友好的用户接口,MAPLE的文法设计从数学上看来十分自然,用户通过终端输入一维的字符串进行操作。MAPLE除了保留了ALGOL等系统语言文法中的函数调用文法,还扩充有“方程”、“范围”等数学结构。

通过下面一些例子可了解MAPLE文法的若干特点。

```
taylor(exp(3*x**2+x), x=0, 4);
```

$$1 + x + \frac{7}{2}x^2 + \frac{19}{6}x^3 + \frac{145}{24}x^4 + o(x^5)$$

这是对一个指数函数在 $x=0$ 点作泰勒展开,取前四项。

```
limit((tan(x)-x)/x**3, x=0);
```

$$1/3$$

这是求函数在 $x=0$ 点的极限值。

```
fibonacci := proc(n)
option remember;
if not type(n, integer) or n < 0 then
ERROR('invalid argument')
else
```

```
if n < 2 then n else
fibonacci(n-1) + fibonacci(n-2)
fi
fi
end;
fibonacci(101);
```

$$573147844013817084101$$

这是用MAPLE语言定义斐波那契数并计算第101个斐波那契数。remember的功能是将这个定义保留下来。请注意这里的控制结构是从ALGOL68中借来的。

```
if <表达式> then <语句>
else <语句>
fi
```

此外,MAPLE也保留了C语言中循环的break语句和Return<表达式>语句。

MAPLE语法将迭代求值作为表达式求值的标准法则。例如:

```
a := x;          x := 3;
a;
```

MAPLE给出了a等3,而不是x。MAPLE也保留了LISP语言中用到的用来禁止求值的法则。在MAPLE中表达式求值,不管是系统函数还是用户定义函数,都是从左到右进行的。

在MAPLE中,每一对象都有其类型,这不但对一般定义下的整数和表是如此。对一些数学对象,如求和(SUM)、数组(arrays)、过程(procedures)也是如此。例如MAPLE中的绝对值函数的定义为

```
abs := proc(x)
if not type(x, rational) and
not type(x, real) then
ERROR('invalid argument to
procedure abs')
else
if x < 0 then -x else x fi
fi
end;
```

这里过程具有proc...end结构,它被当作

合法的MAPLE表达式而赋给名字abs。

MAPLE中数据结构是通过动态向量方法实现的。它对于整个系统的紧凑和效率是非常重要的。这种数据结构的表达形式为：

头1	数据1	数据2	...	数据n
----	-----	-----	-----	-----

例如，有理分数的内部表示为：

有理分数	↑ <正整数或负整数>	↑ <正整数>
------	-------------	---------

其中第二和第三部分分别为二个指针，指向二个整数，而第二个整数为不等于1的正整数，这两个整数互质。

在MAPLE中，表的检索是使用散列算法进行的。其中入口结构是一个指针表：

散列表	↑ <HASH>	...	↑ <HASH>
-----	----------	-----	----------

散列链入口为：

HASH	键	值	...
------	---	---	-----

链取0值意味着该散列链结起来。

效率对于计算机代数系统是至关重要的。稍不留心，结果可以差之甚远。例如，斐波那契数的如下定义：

f := proc(h)

(c34)  $x^3 - 1$ ;

(d34)  $x^3 - 1$

(c35) SOLVE(%, X);

(d35)  $\left[ x = \frac{\sqrt{3} \%i - 1}{2}, x = -\frac{\sqrt{3} \%i + 1}{2}, x = 1 \right]$

(c36) FACTOR( $x^6 + 1$ );

(d36)  $(x^2 + 1)(x^4 - x^2 + 1)$

下面几个例子中，DIFF是微分，EXPAND是展开，INTEGRATE是积分，SUM为求和：

(c38) EXPAND( $(x+1)^3$ );

(d38)  $x^3 + 3x^2 + 3x + 1$

(c39) EXPAND( $(x+1)^7$ );

(d39)  $x^7 + 7x^6 + 21x^5 + 35x^4 + 35x^3 + 21x^2 + 7x + 1$

(c40) DIFF(SIN(X) +  $x^3 + 2 \cdot x^2$ , X);

(d40)  $\cos(x) + 3x^2 + 4x$

if n < 2 then n else

f(n-1) + f(n-2) fi end

其计算时间为指数增长，而通过remember功能来改进定义：

f := proc(n) option remember;

if n < 2 then n else

f(n-1) + f(n-2) fi end;

便只要求线性增加的计算时间。

MAPLE相对而言，只占用了较小的计算空间。它在VAX上的实现大约为100K字节。用户所需的更多的功能，则通过调入相应函数库来实现。因此，它所需要的硬件资源可以由MC68000类的微机来充分提供。

### 三.

我们以著名的符号演算专家系统MACSYMA为例来说明计算机解析演算的方式。其中的表达式符号大多同通常计算机语言是一致的，%表示上一个表达式，分号；表示输入结束，%E表示自然对数的底，%Pi表示园周率，等等。

简单的解方程（例如  $x^3 - 1 = 0$ ）用SOLVE命令，分解因式用FACTOR：

(c41) INTEGRATE (%E^X/(%E^X+2), X) ;

(d41)  $\log(\%e^x + 2)$

(c42) INTEGRATE(1/(X\*LOG(X)), X;

(d42)  $\log(\log(x))$

(c43) INTEGRATE(SIN(2\*X+3), X) ;

(d43)  $-\frac{\cos(2x+3)}{2}$

(c44) SUM(I1, I, 1, 4) ;

(d44) 33

求阶乘和分解质因数 (其中井号表示接下行) :

(c1) 120! ;

Time = 150msec.

(d1) 6689502913449127057588118054090372586752746333138029810295671352301\*

633557244962989366874165271984981308157637893214090552534408589408121859\*

89848111438965000596496052125696000000000000000000000000000000

(c2) factor (%) ;

Time = 933msec.

(d2)  $2^{110} 3^{58} 5^{28} 7^{19} 11^{10} 13^9 17^7 19^6 23^5 29^4 31^3 37^3 41^2 43^2 47^2 53^2 59^2 61 67 71 73 79$

83 89 97 101 103 107 109 113

下例中求一个无穷乘积并给出其泰勒级数的前五项:

(c23) product((1-x^n)^24, n, 1, inf)\*x;

Time = 600msec.

(d23) 
$$\begin{array}{c} \text{inf} \\ \diagup \quad \quad \quad \diagdown \\ \begin{array}{ccc} & ! & \\ & ! & \\ x & ! & (1-x^n)^{24} \\ & ! & \\ & ! & \end{array} \\ n = 1 \end{array}$$

(c24) /\* we should be able to find the first five terms by taylor expanding, \*/  
taylor(%, x, 0, 5) ;

Time = 700msec.

(d24) /T/  $x - 24x^2 + 252x^3 - 1472x^4 + 4830x^5 + \dots$

下面例子中, 用MACSYMA函数ALGSYS求解代数方程组F1=0和F2=0的未知数x, y:

(c6) F1: X\*\*2-Y\*\*2;

Time = 66msec.

(d6)  $x^2 - y^2$

(c7) F2: X\*\*2-X+2\*Y\*\*2-Y-1;

Time = 50msec.

(d7)  $2y^2 - y + x^2 - x - 1$

(c8) ALGSYS([F1, F2], [X, Y]);

Totaltime = 12166msec. GTime = 2950msec.

(d8)  $\left\{ \left[ x = -\frac{1}{\sqrt{3}}, y = \frac{1}{\sqrt{3}} \right], \left[ x = \frac{1}{\sqrt{3}}, y = -\frac{1}{\sqrt{3}} \right] \right\}$

$$\{x=1, y=1\}, \left\{x=-\frac{1}{3}, y=-\frac{1}{3}\right\}$$

最后, 我们用MACSYMA来求解一个简单电路的微分方程问题。考虑如下电路图1:

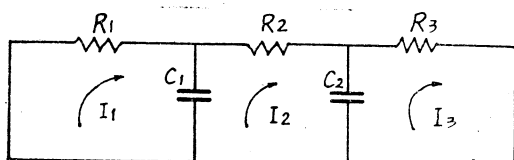


图1

这里三个回路, 其中回路中电流与电荷关系为

$$I_n = \frac{dQ_n}{dt} \quad (n=1, 2, 3)$$

为简化讨论, 选择参数为

$$R_1 = R_3 = C_1 = C_2 = 1$$

$$R_2 = 1/P$$

这样就得到与电路相应的微分方程

$$\frac{dQ}{dt} = A Q$$

其中Q和A分别为

$$Q = \begin{pmatrix} Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}, \quad A = \begin{pmatrix} -1 & 1 & 0 \\ P & -2P & P \\ 0 & 1 & -1 \end{pmatrix}$$

为了解出这一微分方程组, 可设

$$Q(t) = u e^{\lambda t}$$

其中u为待解的本征向量, 方程组化为如下本征值问题

$$A u = \lambda u$$

要解出本征值 $\lambda$ , 只须解出使矩阵 $A - \lambda I$ 行列式值为零的 $\lambda$ 。让我们来看怎样用MACSYMA来解这个例题。

首先输入本征问题软件

```
(C2) LOADFILE("DUAI, [MAC.
      SHARE] EIGEN, MAC");
```

```
(D2)      DONE
```

输入矩阵A:

```
(C3) A: MATRIX([[-1, 1, 0],
[P, -2*P, P], [0, 1, -1]]);
```

$$\begin{bmatrix} -1 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} \end{bmatrix}$$

```
(D3)      [P -2P P]
```

$$\begin{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & -1 \end{bmatrix}$$

求本征值和本征向量:

```
(C4) EIGENVECTORS(A);
```

系统给出本征值 (第一部分) 和三个本征向量:

```
(D4) [[[-2P-1, -1, 0], [1, -2P, 1]
```

```
[1, 0, -1], [1, 1, 1]]
```

这就原则上给出了微分方程的解。为了检验这个结果, 我们下面用相似变换将A对角化。用上面给出的本征向量为列向量组成矩阵M:

```
(C5) M: ADDCOL(TRANPOSE
(PART(%, 2)), TRANPOSE(PART
(%, 3)), TRANPOSE(PART(%, 4));
```

求出其逆矩阵:

```
(C6) INVERT(M), DETOUT;
```

$$\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} 2P+1 & 0 & -2P-1 \end{bmatrix}$$

$$\begin{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} 2P & 2 & 2P \end{bmatrix}$$

```
(D6)      4P+2
```

最后, 用相似变换将A对角化( $M^{-1}AM$ ):

```
(C7) FACTOR(EXPAND(%, A, M);
```

$$\begin{bmatrix} -(2P+1) & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} \end{bmatrix}$$

```
(D7)      [ 0      -1    0 ]
```

$$\begin{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

得到的矩阵的对角元素 $-(2P+1)$ ,  $-1$ ,  $0$ 的确是前面D4中第一部分所给本征值。将求得的本征向量和本征值代入 $Q(t) = u e^{\lambda t}$ , 便得到电荷Q随时间的变化关系。

(下转第21页)



# 一个实用的A/D、D/A转换电路

王士元

(南开大学)

**摘要** 本文介绍一个自控项目中采用的A/D、D/A转换电路。它经济实用,稳定可靠,可满足一般的精度要求。编程也容易。D/A部分为一位八位数字量输入,可选择的单极性或双极性输出,转换时间约 $1\mu s$ 。笔者根据实践,还列出了A/D在中断方式下和查询方式下的示范程序清单,A/D、D/A联合使用时的程序清单。用本文介绍的电路制成的插件式A/D、D/A印制电路板已成功的用于项目中。

A/D电路可八路模拟量输入( $0\sim 5$ 伏),八位二进制输出。D/A电路可一路八位数字量输入, $0\sim -5V$ 或 $0\sim -10V$ 单极性输出,或者 $-5V\sim +5V$ , $-10V\sim +10V$ 双极性输出。在TP—801单板机控制下,能较好地完

成数据采集和数字控制量模拟输出。A/D电路既可在中断方式下工作,也可在查询方式下工作,适用性强。电路如图1所示。

现将A/D和D/A 电路部分的特点和编程简介如下:

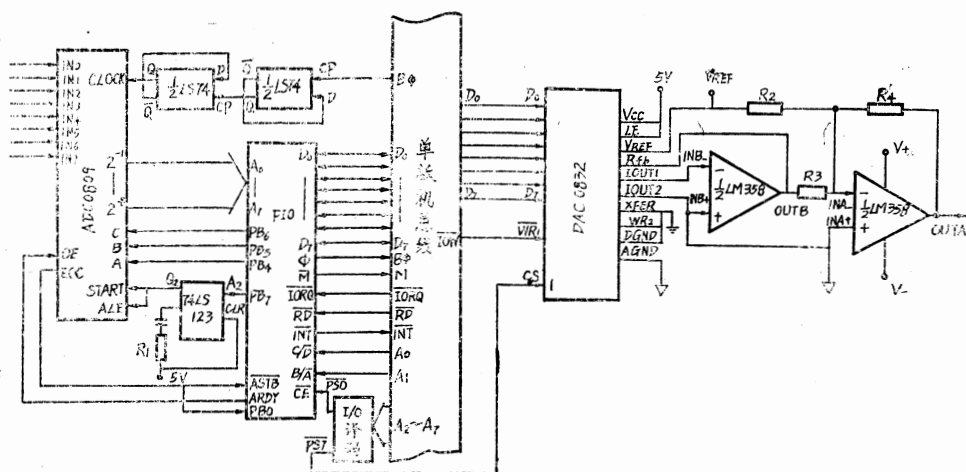


图1 A/D、D/A电路图

## 一、A/D转换电路

A/D采用ADC0809,其时钟频率是将单板机时钟 $B\phi$ 用74LS74 四分频后得到,而其控制信号和转换后的数据锁存,是由一片PIO集成电路完成(文中所述,是采用TP

—801上的PIO,不再另加)。其中B口用于产生控制信号和输入A/D转换完成(EOC)信号,故工作在位控方式。A口用于接收转换后的数据,工作在输入方式。

当初始化时,PIO的ARDY为低电平(无效),为此执行一条形式IN A, (80H)指令,使ARDY变为高电平,表示等待接收数

据, 程控使PB7产生一个0→1跳变, 其后沿触发74LS123单稳产生约1μs宽的脉冲。启动A/D开始转换, 当转换完成, 0809产生EOC信号(高电平有效)送至PIO的 $\overline{\text{ASTB}}$ 脚, 同时转换后的数据就锁存到A口的数据输入寄存器中, EOC信号的后沿引起PIO中断请求, 紧接着时钟 $\phi$ 的下降沿使ARDY信号变低, 即不再接收数据。当CPU响应中断, 就转入中断服务程序, 当执行IN A, (80H) 指令后, 就将锁存在A口的转换数据取走, ARDY又变高, 等待下一次转换后的数据。此时再次启动A/D, 并返回主程序, 当A/D转换完成后, 又重复上述过程。

注意: 在第一次启动A/D时, 必须执行一条IN指令, 以使ARDY变高, 否则第一次启动后, A/D转换的数据由于PIO ARDY为低, 故不能接收而丢失!

A/D八个输入通道的选择, 由PIO B口的PB4、PB5、PB6来决定。

PIO和AD0809相互协调的工作时序如图2所示。

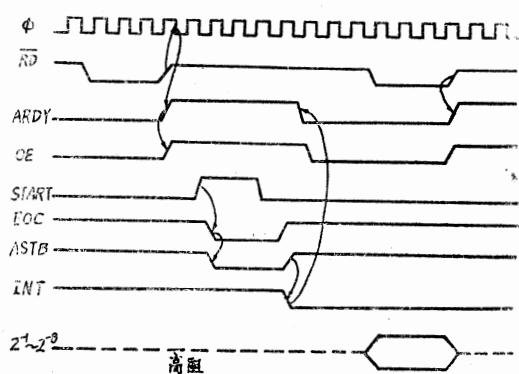


图2 PIO与A/D时序图

A/D电路在上述中断方式下工作时, 可提高CPU的工作效率。即当启动A/D开始转换后, CPU大约仍可执行几十条指令。当转换完成, 并产生中断时, CPU才转入A/D中断服务程序。

中断方式下的示范程序如下:

Tektronix			Z80 ASM V4.0B			
00001	2000	>	ORG	2000H		
00002	2000	3E27	START LD	A, 27H	} PIO A口中断矢量 27B2H	
00003	2002	ED47	LD	I, A		
00004	2004	3EB2	LD	A, 0B2H		
00005	2006	D382	OUT	(82H), A	} PIO A口输入方式 允许中断	
00006	2008	3E4F	LD	A, 4FH		
00007	200A	D382	OUT	(82H), A		
00008	200C	3E87	LD	A, 87H	} PIO B口位控方式 PA0为输入	
00009	200E	D382	OUT	(82H), A		
00010	2010	3ECF	LD	A, 0CFH		
00011	2012	D383	OUT	(83H), A		
00012	2014	3E01	LD	A, 01H		
00013	2016	D383	OUT	(83H), A		
00014	2018	1680	LD	D, 80H		
00015	201A	1E00	LD	E, 00H		
00016	201C	216721	>	LD	HL, BUF	缓冲区首地址
00017	201F	0608	LD	B, 08H		八个通道
00018	2021	7A	LOOP1 LD	A, D	} 启动A/D, 打开0通道	
00019	2022	D381	OUT	(81H), A		
00020	2024	7B	LD	A, E		
00021	2025	D381	OUT	(81H), A		

00022	2027	DB80		IN	A, (80H)	} 使PIO A口ARDY变高
00023	2029	ED5E		IM	2	
00024	202B	FB	Q1	EI		} 等待中断
00025	202C	76		HALT		
00026	202D	C32B20 >		JP	Q1	
00027		27B2 >		ORG	27B2H	} 中断入口地址
00028	27B2	4621		BYTE	46H, 21H	
00029		2146 >		ORG	2146H	
00030	2146	DB80		IN	A, (80H)	} 取出转换结果并存入缓冲区
00031	2148	77		LD	(HL), A	
00032	2149	23		INC	HL	} 八个通道采样完否?
00033	214A	05		DEC	B	
00034	214B	C25721 >		JP	NZ, CN	
00035	214E	1680		LD	D, 80H	} 是, 重新从0通道开始
00036	2150	1E00		LD	E, 00H	
00037	2152	0608		LD	B, 08H	
00038	2154	CZ5E21 >		JP	Q2	} 否, 通道号加1
00039	2157	7A	CN	LD	A, D	
00040	2158	C610		ADD	A, 10H	
00041	215A	57		LD	D, A	} 送出启动信号
00042	215B	D680		SUB	80H	
00043	215D	5F		LD	E, A	
00044	215E	7A	Q2	LD	A, D	} 送出启动信号
00045	215F	D381		OUT	(81H), A	
00046	2161	7B		LD	A, E	
00047	2162	D381		OUT	(81H), A	} 开中断并返回
00048	2164	FB		EI		
00049	2165	ED4D		RETI		
00050	2167	00FF	BUFFER	BLOCK	0FFH	
00051						

在许多应用项目中, 需对监测的信号进行定时采样, 这可用CTC来实现。在这种方式下工作时, 可初始化PIO, 令A口为输入方式, 并禁止中断, 设B口为位控方式, 禁止中断, 通过PB4~PB6选择输入通道, 并由PB7产生启动A/D转换的信号, 然后不断的对PBO查询。当它为1时 (即EOC有效), 表明转换完成, 这时可将数据取走, 然后返回主程序, 下一次定时时间到, 又重复此过程。

查询方式下的示范程序如下:

Tektronix		Z80 ASM V4.0B				
00001		2000	>	ORG	2000H	
00002	2000	3E4F		LD	A, 4FH	} PIO A口输入禁止
00003	2002	D382		OUT	(82H), A	
00004	2004	3E07		LD	A, 07H	} 中断
00005	2006	D382		OUT	(82H), A	

00006	2008	3ECF		LD	A,0CFH	B口位控, PBO为 输入
00007	200A	D383		OUT	(83H), A	
00008	200C	3E01		LD	A,01H	
00009	200E	D383		OUT	(83H), A	
00010	2010	3E27		LD	A,27H	
00011	2012	ED47		LD	I, A	CTC中断矢量字
00012	2014	3E9B		LD	A,9BH	
00013	2016	D384		OUT	(84H), A	
00014	2018	3EB5		LD	A,0B6H	CTC定时及中断方式
00015	201A	D384		OUT	(84H), A	
00016	201C	3E08		LD	A,08H	装时间常数
00017	201E	D384		OUT	(84H), A	
00018	2020	210024		LD	HL,BUF	
00019	2023	3600		LD	(HL),00H	
00020	2025	DB80		IN	A,(80H)	PIO A口ARDY变高
00021	2027	ED5E		IM	2	
00022	2029	FB		EI		
00023	202A	76	0Q	HALT		
00024	202B	C32920	>	JP	Q0	等待定时时间到
00025		279B	>	ORG	279BH	
00026	279B	1024		BYTE	10H,24H	中断服务入口
00027		2410	>	ORG	2410H	
00028	2410	0608		LD	B,08H	
00029	2412	1680		LD	D,80H	
00030	2414	1E00		LD	E,00H	
00031	2416	7A		LD	A,D	启动A/D
00032	2417	D381		OUT	(81H), A	
00033	2419	7B		LD	A,E	
00034	241A	D381		OUT	(81H), A	转换完成否? 检测PBO
00035	241C	DB81	Q1	IN	A,(81H)	
00036	241E	CB47		BIT	0, A	未完, 等待
00037	2420	28FA		JR	Z, Q1	
00038	2422	DB80		IN	A,(80H)	完, 数据存入存贮单元
00039	2424	77		LD	(HL), A	
00040	2425	23		INC	HL	
00041	2426	3600		LD	(HL), 0	
00042	2428	3E10		LD	A,10H	算出下一启动的通道号
00043	242A	82		ADD	A,D	
00044	242B	57		LD	D,A	
00045	242C	D680		SUB	80H	
00046	242E	5F		LD	E,A	
00047	242F	05		DEC	B	
00048	2430	C23624	>	JP	NZ,CN	八路采集完返主程序
00049	2433	FB		EI		
00050	2434	ED4D		RETJ		

00051	2436	7A	CN	LD	A,D	八路未采集完, 再次启动 下一通道
00052	2437	D381		OUT	(81H), A	
00053	2439	7B		LD	A,E	
00054	243A	D381		OUT	(81H), A	
00055	243C	C31C24	>	JP	Q1	
00056		2400	BUF	EQU	2400H	
00057						

## 二、D/A转换电路

D/A电路采用DAC0832集成电路和双运算放大器LM358, DAC0832将数字量转换成差动电流输出, 经运算放大器变成电压输出。当从358第7脚引出时(断开R3电阻)形成单极性电压输出,  $0 \sim -5V$  (当 $V_{REF}$ 接 $+5V$ 时)或 $0 \sim -10V$  ( $V_{REF}$ 接 $+10V$ 时)。若需双极性电压输出时, 可将后级反相求和电路联上, 这时输出电压将在 $-5V \sim +5V$ 或 $-10V \sim +10V$ 范围内。电路转换时间约 $1\mu s$ 。

示范程序如下:

```

Tektronix      Z80 ASM V4.0B
00001          2000 >      ORG 2000H
00002 2000 06FF      N1 LD B,0FFH
00003 2002 78        N2 LD A,B
00004 2003 D39C      OUT (9CH),A
00005 2005 10FB      DJNZ N2
00006 2007 18F7      JR N1
00007

```

输出端OUTA接示波器Y轴, 运行此程序, 将会看到连续不断的锯齿波。

由于D/A电路有数字和模拟之分, 因此在设计印制电路时, 模拟的汇集一点, 数字的汇集一点, 然后两点再联结在一起, 否则数字信息将会串入模拟回路, 形成干扰, 造成转换时的很大误差。

由于运算放大器采用双运放LM358, 输

入失调电压 $\pm 2mV$ , 有自调零功能, 因而省去了用精密电位器调零的线路, 只要调整 $R_2$ 、 $R_3$ 、 $R_4$ , 使 $R_2 = R_4$ ,  $R_2/R_3 = 2$ , 阻值精确到 $0.4\%$ 以上即可。本电路采用 $R_2$ 、 $R_4$ 为 $15K$ ,  $R_3 = 7.5K$ 。可以发现一旦转换失调电压大, 肯定是电阻很不成比例造成的。LM358可用LM158或LM258代替, 也可用国产CF158、CF258和CF358代替。

## 三、A/D, D/A的联合使用

一般控制电路经常需要A/D, D/A电路联合工作, 如在闭环自动控制中的PID控制, 用A/D电路采集被控量, 然后与设定值比较, 进行PID运算, 给出控制量, 由D/A电路进行电压或电流输出, 以控制执行机构。为了验证A/D, D/A两电路联合使用的效果, 可用信号发生器输出一个峰峰值为 $2.5V$ 的正弦信号或别的信号(设周期为 $5ms$ ), 用 $50\mu$ 左右的电容将其接至A/D通道口0上, D/A的输出端OUTA接至示波器Y轴输入(用高阻抗探头, 如 $10:1$ 的 $10M\Omega$ 探头), 启动文中所示示范程序后, 就可在示波器上看到一个连续不断的用点构成的正弦波, 其形状周期完全和输入信号一致, 而构成波形的点之间的时间差, 就表示了采样转换然后反转换成模拟量的时间(当然严格地说, 还应包括一些指令执行的时间)。

A/D, D/A联合使用示范程序如下:



## Tekronix

## Z80 ASM V4.0B

00001	2000	>	ORG	2000H	
00002	2000	3E27	START LD	A, 27H	
00003	2002	ED47	LD	1, A	PIO A口中断矢 量27B2H
00004	2004	3EB2	LD	A, 0B2H	
00005	2006	D382	OUT	(82H), A	
00006	2008	3E4F	LD	A, 4FH	
00007	200A	D382	OUT	(82H), A	A口输入方式 允许中断
00008	200C	3E87	LD	A, 87H	
00009	200E	D382	OUT	(82H), A	
00010	2010	3ECF	LD	A, 0CFH	
00011	2012	D383	OUT	(83H), A	B口位控方式 0A0位为输入
00012	2014	3E01	LD	A, 01H	
00013	2016	D383	OUT	(83H), A	
00014	2018	1680	LD	D, 80H	
00015	201A	1E00	LD	E, 00H	
00016	201C	7A	LD	A, D	
00017	201D	D381	OUT	(81H), A	启动A/D 0号通道
00018	201F	7B	LD	A, E	
00019	2020	D381	OUT	(81H), A	
00020	2022	DB80	IN	A, (80H)	使ARDY变高
00021	2024	ED5E	IM	2	
00022	2026	FB	Q1 EI		等待中断
00023	2027	76	HALT		
00024	2028	C32620	> JP	Q1	
00025		27B2	> ORG	27B2H	
00026	27B2	4621	BYTE	46H, 21H	中断入口
00027		2146	> ORG	2146H	
00028	2146	DB80	IN	A, (80H)	取出转换结果并 送D/A 转换输出
00029	2148	D39C	OUT	(9CH), A	
00030	214A	1680	LD	D, 80H	
60031	214C	1E00	LD	E, 00H	
00032	214E	7A	LD	A, D	
00033	214F	D381	OUT	(81H), A	重新启动A/D转换
00034	2151	7B	LD	A, E	
00035	2152	D381	OUT	(81H), A	
00036	2154	FB	EI		
00037	2155	ED4D	RETI		中断返回
00038					

## 注释:

该电路采用TEKTRNIX 8550微机开发系统进行了软硬件调试, 文中所附各程序清单均是在仿真状态下由8550 MDS打印。

当将此电路和TP-801结后, 键入上述各程序由TP-801执行时, 文中所述结果全一样。该电路用PC机辅助设计成印制板, 已成功地用于项目中。

# 逻辑阵列器件的特点及应用

**摘要** 随着电路设计的复杂化,印制板上器件密度愈来愈高,为实现有效面积内容纳更详尽复杂的逻辑功能,人们已经注意应用逻辑阵列器件。特别是近几年,逻辑阵列器件品种日益齐全,自动编程软件日趋完善,装备逻辑阵列器件的功能板越来越多。我们曾在S-100总线功能板上使用了它,拟在设计VME总线功能板时进一步应用该类器件简化电路设计。本文将就该器件基本功能和使用,以及ABEL编程“语言”加以介绍。

## 一、基本概念及结构

### 1. 逻辑阵列器件的逻辑表达符号

用简化逻辑符号画逻辑图使逻辑图和逻辑电路之间建立一一对应的关系,读起来方便,易于理解。如图1所示。图中“×”表示完整的熔丝,以实现逻辑“与”、“或”功能。(应注意,输入端的“×”并非将全部输入端本身连接。)

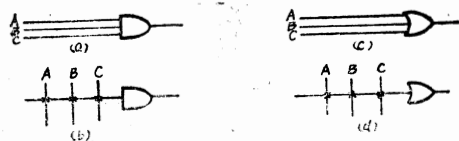


图 1

图1中(a)、(c)为正常画法,  
(b)、(d)为简化逻辑符号。

使用简化逻辑符号,我们可以画出任何一个组合逻辑图。以异或逻辑为例其逻辑式为:

$$O = I_1 \bar{I}_2 + \bar{I}_1 I_2$$

简化逻辑符号逻辑图由图2所示。

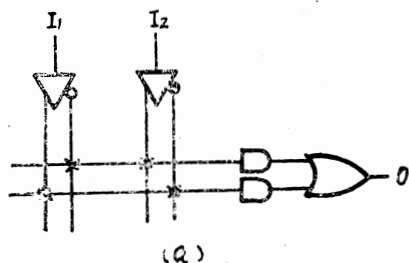


图 2

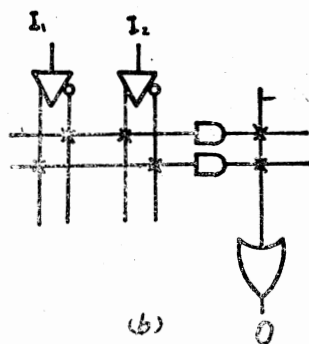


图 2

图2(a)和(b)在电路形式上有所不同,但其完成的功能是一样的。实际上,(a)和(b)分别代表了两种不同的逻辑阵列电路,前者为PAL电路,后者为PLA电路。

### 2. PLA, PAL及PROM比较

用简化逻辑符号,将PLA, PAL和PROM进行比较,便于了解其间的联系及差别(见图3)

图3(a)为PROM电路,它由一个可编程的“或”阵列和固定连接的“与”阵列组成。固定的输入端作为计算机存储器的地址输入,而输出端则是存储器内容。当作为逻辑器件使用时,PROM的输入线为N,“与”阵列的积项输出线为 $2^N$ 根,积项总数满足这一要求时才能使用。与逻辑阵列相比,PROM中“与”阵列的利用率不高。

图3(b)为PAL电路,由可编程的“与”阵列和固定的“或”阵列组成,与PROM相比,可以更有效的利用“与”阵列。但是,用PAL电路与图3(c)所示的

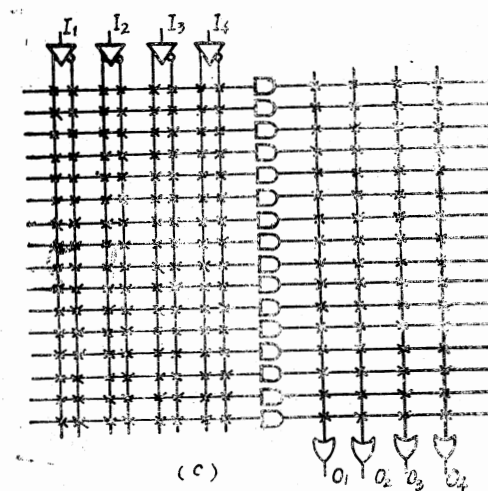
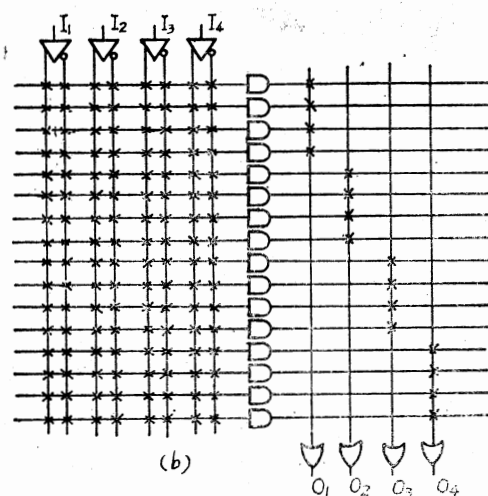
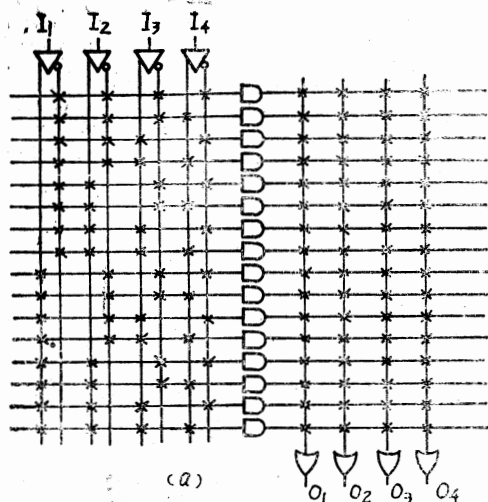


图 3

PLA 电路相比,可以看出具有一定的局限性,使用起来不如PLA灵活。PLA 电路的“与”、“或”阵列完全都是可编程的,可以充分利用“与”、“或”阵列。PAL “或”阵列的线数是固定的,如图所示,输入1个或门的积项线固定为4根,(这是PAL 电路设计时决定的),当组合逻辑式中或项超过4个时,使用该器件将不能满足要求。另外,如果逻辑式中的或项不足4个,即在1个PAL 或门上没有用满全部积项线,剩下的几根也不能再做它用。

在使用逻辑阵列进行逻辑电路设计时,PLA 电路要比PAL 电路的适应能力强一些。假定所设计的逻辑如下式所示:

$$O_1 = I_1 \cdot I_2 + I_2 \cdot I_3, \quad O_3 = I_1 \cdot I_2 + I_4 \cdot I_1$$

$$O_2 = I_3 \cdot I_4 + I_4 \cdot I_1, \quad O_4 = I_3 \cdot I_4 + I_2 \cdot I_3$$

实现这一逻辑式的逻辑图如图4所示,可以看出用PAL 电路需要8个积项,而用PLA 电路只需4个积项。

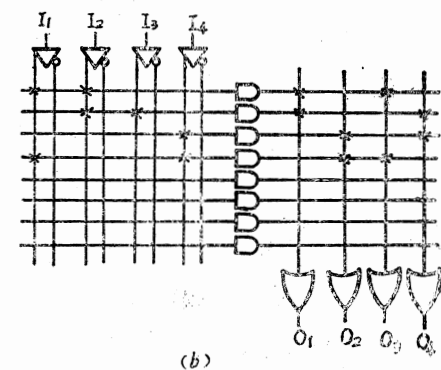
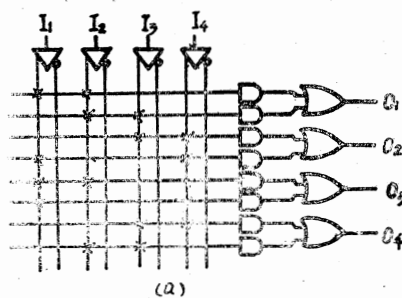


图 4

从选择使用的角度来说,小规模、低功耗、高速且无设计工具时可用PAL电路;大规模、复杂,有可能使用设计工具时用PLA电路较适宜。

### 3. 逻辑阵列器件的基本类型

#### (1) 基本逻辑阵列及有效输出电平

图5(a)为PAL基本电路,当输出用“或非”门时有效输出电平为“低”;当用“或”门时,有效输出电平为“高”。图5(b)为PLA基本电路,它比较巧妙地使用了异或门解决输出有效电平问题。当异或门一端接地时,有效输出电平为高,否则为低。这一功能给应用带来很大方便,如逻辑式 $F = A + B + C + D$ ,实现这一逻辑功能时需要4个积项,若将此式转变为 $\overline{F} = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$ 则只需要1个积项。对于PLA电路来说,可以节约几个积项另做它用。

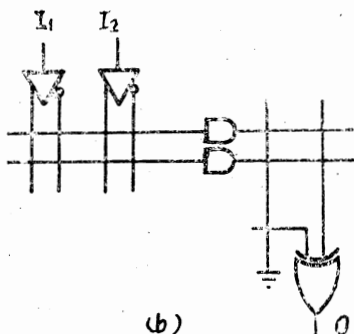
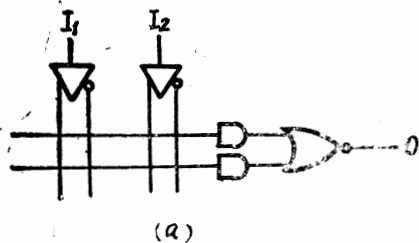


图 5

#### (2) 反馈及可编程I/O

如图6所示,在基本逻辑阵列的输出端加上反馈门,将或门输出反馈到与阵列,和其它输入形成积项。这样一来,可以在逻辑阵列内部实现较复杂的逻辑功能。三态输出

缓冲器由“与”阵列的一个积项输出直接控制。当三态缓冲器有效时,I/O端作为输出使用;而当三态缓冲器无效时,I/O端可用作输入端。这一功能可以提高电路使用的灵活性。

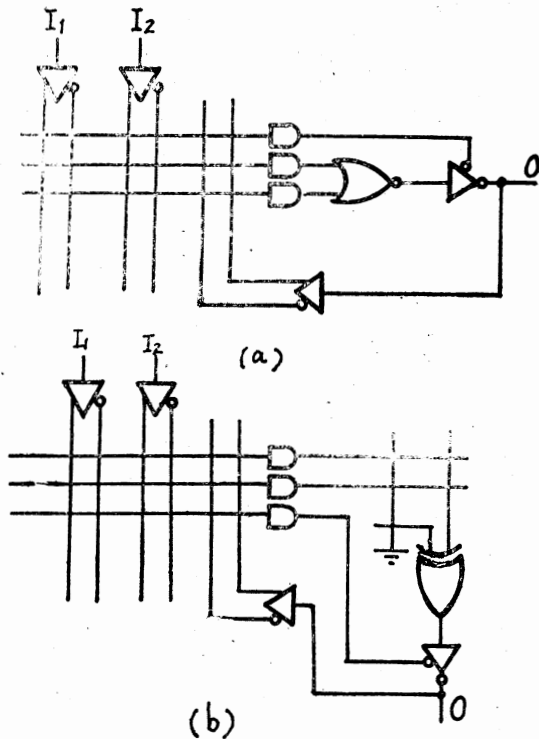


图 6

#### (3) 触发器型

如图7所示,每一个“或”门输出端与一个触发器相连,触发器的一个输出端与三态缓冲器相连作为逻辑阵列输出使用,而触发器的另一个输出端则与一个反馈门相连。一般情况下,触发器型逻辑阵列都带有反馈。这样的逻辑结构可使电路用于时序电路设计,状态记忆等等,可完成计数加,计数减,跳跃,移位,转移等逻辑功能。

在PAL电路中,多用D型触发器,引出端设有CK和OC端。PLA电路的触发器类型种类较多,有J-K型, D型和R-S型等等。如82S159,可程序设计多种触发器类型,根据需要可将触发器变形为J-K型, D型, T型,由积项输出编程控制,以实现各种逻辑

辑功能。

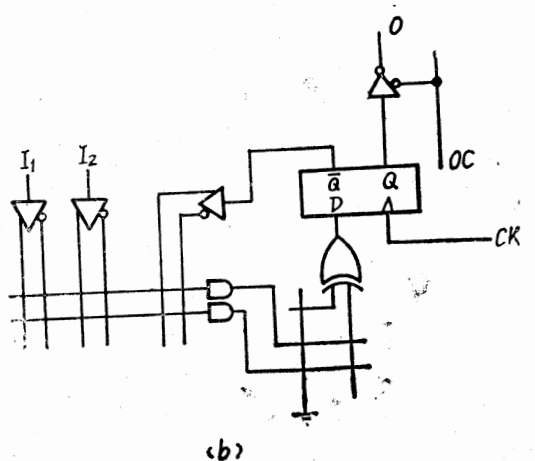
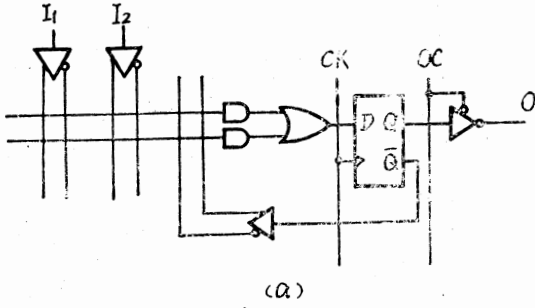


图 7

#### (4) 异或PAL

在触发器PAL电路中，将原来属于一个或门的积项分两部分，然后将这个或门再进行异或，送到触发器。异或PAL包括了触发器型PAL的全部功能，且在完成计数和其它状态序列操作功能时也较方便（见图8）

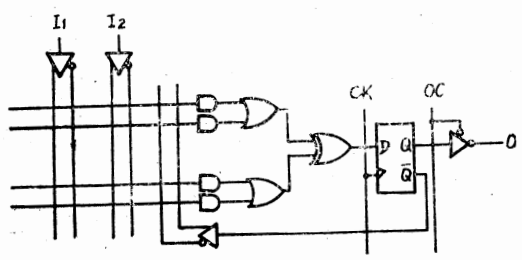
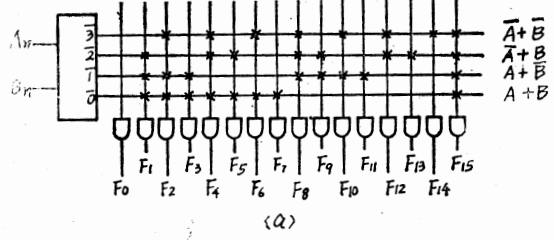


图 8

#### (5) 输入译码型。

在输入端附加2输入端译码器，如图9所示，这个功能实现了两个变量之间算术运算能力且有效地利用了积项。

通常，逻辑阵列的输入是将变量I及其补 $\bar{I}$ 送至与阵列的输入端，若将全部N个变量送入有N个输入端的译码器则输出有 $2^N$ 个，这样一来“与”阵列的积项可用译码器输出代替了，实际上这相当于PROM。输入译码型逻辑阵列在结构上吸取了PROM的优点。MMI公司出售的 $16 \times 4$ 输入译码型PAL，将异或PAL的触发器输出反馈到2输入端译码器做为一个输入使用，具有更大的灵活性。



F0	0	F4	$\bar{A} \cdot B$	F8	$(A+B)$	F12	$\bar{A}$
F1	$A \cdot B$	F5	B	F9	$A \oplus B$	F13	$\bar{A+B}$
F2	$A \cdot \bar{B}$	F6	$A \oplus B$	F10	$\bar{B}$	F14	$(A \cdot B)$
F3	A	F7	A+B	F11	A+B	F15	1

注： $\oplus$  - 同或  
 $\ominus$  - 异或

图 9

#### 4. 逻辑阵列器件的新发展

近年来，为适应电路设计的需求，一些公司相继生产了功能较强的PAL电路，如MMI公司的24RS系列。它的特点是积项线共享，即一根积项线连至两个或门；可编程选择输出有效电平；加电清“0”及触发器预置等。大规模的PAL有64R32，共有64个输入端和32个输出端，还有32个触发器，可构成约5000个门的逻辑。该器件也具有积项线共享，编程选择输出的有效电平及预置触发器等特点。

此外，CMOS型阵列逻辑电路及紫外线可擦除的逻辑阵列器件也相继问世，该类器

件的使用范围必将进一步扩大。

## 二、逻辑阵列的图形设计

### 1. 组合电路设计

使用逻辑阵列器件进行逻辑设计与以往用SSI、MSI器件的不同之处仅在于逻辑阵列器件的“与”“或”阵列结构。逻辑设计过程是将逻辑变量之间的函数关系变形为阵列结构的基本“与”“或”关系,即积和式,进而得到器件内部熔断器状态分布图形,这就是阵列的图形设计过程。

若更有效地使用逻辑阵列,必须注意积项最小、或项共有部分及适当选择输出极性。其过程为图10所示

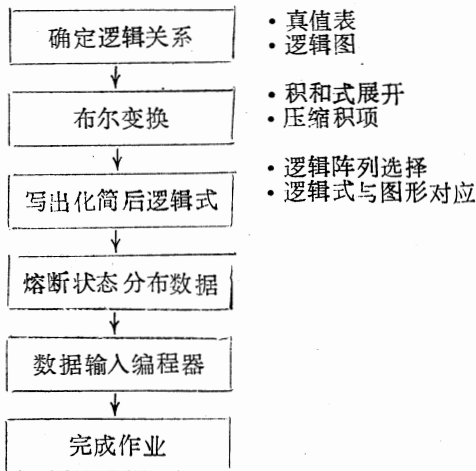


图 10

第一步: 确定逻辑关系, 即输入, 输出信号之间的函数关系, 可用真值表和逻辑图表示。

第二步: 进行布尔代数式的变换, 将复杂的逻辑关系式变成逻辑阵列器件所能接受的“积和式”形式。

第三步: 用卡诺图简化逻辑, 尽量减少积项线数, 化简后的表达式用于图形变换, 形成逻辑阵列器件内部熔断器状态分布图。

第四步: 将熔断器状态分布图变换为分布

数据, 例如, 第00行至07行的分布图为:

× × - - × × - -

(“×”表示熔断器完好, “-”表示熔断器烧断)。则分布数据为00110011, 16进制数表示为33。

第五步: 在编程器上完成分布数据输入, 将数据打入与熔断器相对应的地址中, 如00-07的值是33, 08-15的值是5A, 则在00-07相应的地址内写入33, 在下一地址写5A。

第六步: 完成熔断作业。

以图11所示的电路为例。电路为4输入端, 2输出端, “异或”、“与非”、“或非”的组合逻辑, 设计其逻辑阵列图形。

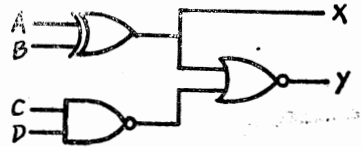


图 11

布尔代数式为  $Y = ((A \oplus B) + \overline{C \cdot D})$

$$X = A \oplus B$$

若使用负逻辑, 其布尔代数式为

$$\overline{Y} = (A \oplus B) + \overline{C \cdot D}$$

$$\overline{X} = \overline{A + B}$$

将上式展开为积和式:

$$\overline{Y} = \overline{A} \cdot B + A \cdot \overline{B} + \overline{C} + \overline{D}$$

$$\overline{X} = \overline{A} \cdot \overline{B} + A \cdot B$$

使用输出低有效的PAL电路, 设计结果为6积项的PAL电路图形, 图12为设计图形。

若将原式按正逻辑展开, 则可得到下式:

$$Y = A \cdot B \cdot C \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot D$$

$$X = \overline{A} \cdot \overline{B} + A \cdot B$$

使用输出高有效的PAL电路只用4个积项就能完成其逻辑功能。从卡诺图上可以直接看出使用正逻辑Y只有两项, 而使用负逻辑则需要4项。见图13。

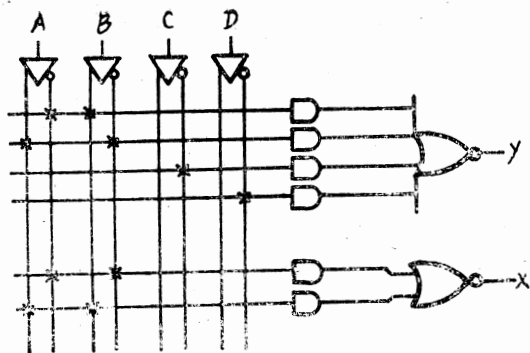


图 12

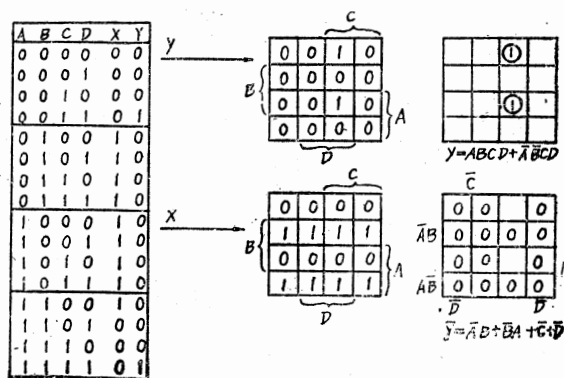


图 13

进一步变化上式，可以得到。

$$Y = (AB + \overline{A}\overline{B})C \cdot D$$

而  $AB + \overline{A}\overline{B}$  正是  $\overline{X}$ ，这样我们在 Y 和 X 之间找到了共有“或”项。使用异或输出的 PLA 电路，只用 4 个积项也同样实现了其逻辑功能，设计图形如图 14 所示。

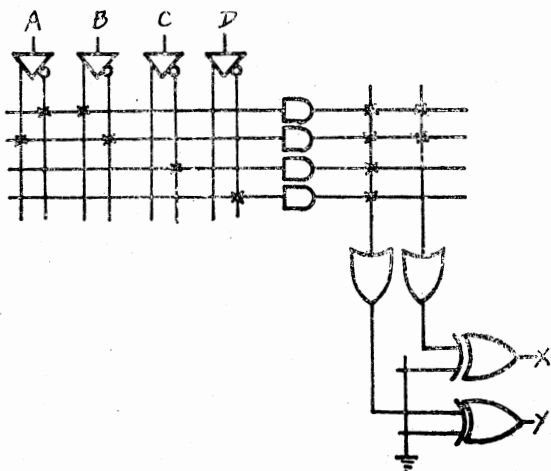


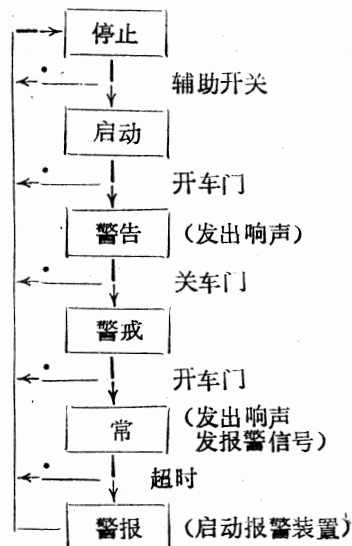
图 14

## 2. 时序电路设计

时序电路分同步型和非同步型两种，内部触发器用于状态记忆，电路动作与外部时钟同步为同步型，而利用或门输出反馈至与阵列的器件为非同步型。非同步型与同步型相比，状态转移只依赖于器件本身的延迟时间，因而速度较快。市场上出售的大部分用于状态记忆的器件为同步型，我们以同步型器件为例说明设计过程。其顺序为：

- 确定输入、输出信号
- 分配内部状态
- 制作状态转移图
- 制作状态转移表
- 写出积和逻辑式
- 压缩积项

图 15 为报警装置各状态的转移过程。当小偷撬车门、前盖、行李箱时，报警器启动。安装在该部位的辅助开关将报警器从“停止”状态转移至“启动”状态。小偷打开车门则使报警器处于警告状态，发出响声。此时若关上车门，响声暂时停止，立即转入警戒状态继续监视。小偷若庆幸报警器不响而再次打开车门时，则报警器转入非常状态，发



注“.”车主开车锁

图 15



出报警信号，继而转移到报警状态，启动报警装置发出警报。不论在任何一种状态下，车主用钥匙打开车锁，发动马达，报警器将返回到停止状态。

图16为报警器的逻辑图

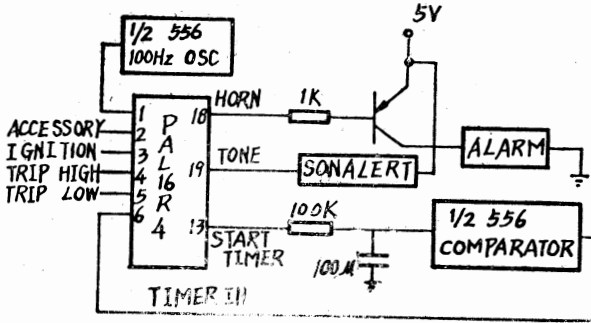


图 16

该报警器共有6个状态，只需3个触发器（ $2^3 = 8$ ）即可实现其功能，选用 PAL 16R4器件已经完全可以了。下面逐步说明其设计过程：

#### (1) 确定输入，输出信号

第一脚为时钟输入；第二脚为辅助开关输入；第三脚为车锁输入线；第四、五脚为车门打开输入线，分高有效和低有效；第六脚为时间延迟输入信号，由十三脚启动时间延迟电路而来；输出一方，十八脚、十九脚用于报警。该设计实例中，由4个触发器输出的信号，14，15，16，17脚均未使用，触发器完全用于记忆内部状态，该电路输出完全由内部状态和输入条件决定。

#### (2) 分配内部状态

设计时序电路时，保证其稳定可靠的绝对条件是内部状态分配。如图17所示，将6个报警器状态分配给内部触发器， $S_2, S_1, S_0$ 各位在进行状态转移时，只能有一位变化，否则将有可能进入未定义状态而又无退出条件，出现“死锁”。

#### (3) 制作状态转移图

图18为状态转移图，为便于阅读，图中

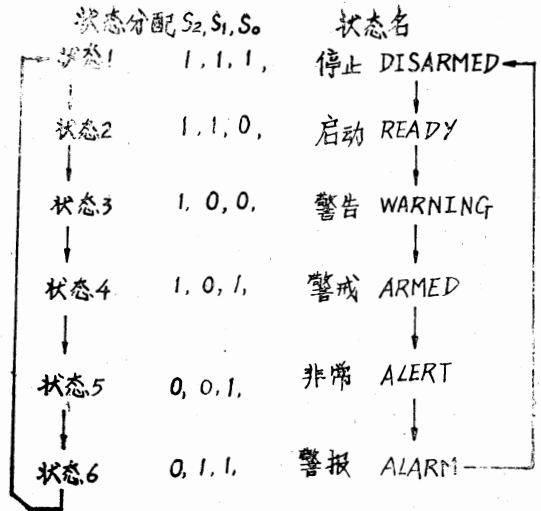


图17

状态转移采用宏定义。

$$\text{TRIP} = \text{TRIP-HIGH} + \overline{\text{TRIP-LOW}} \cdot \text{IGNITION}$$

$$\text{UNTRIPPED} = \overline{\text{TRIP-HIGH}}$$

$$\cdot \text{TRIP-LOW} \cdot \text{IGNITION}$$

$$\text{WFT} = \text{TIMER-IN} \cdot \text{IGNITION}$$

$$\text{TIMEOUT} = \overline{\text{TIMER-IN}}$$

$$\cdot \text{IGNITION}$$

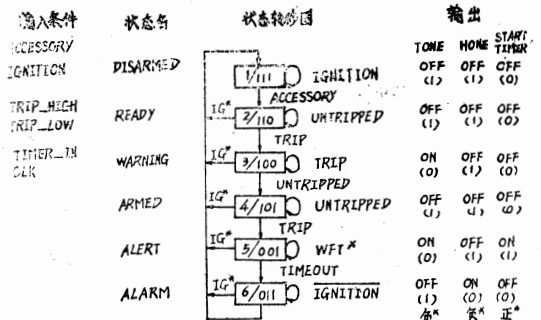


图18

注：IG·即IGNITION

WFT·即WAIT-FOR-TIMEOUT

#### (4) 制作状态转移表

表1（见42页）为状态转移表。左边记录当前状态，中间为转移条件，右边为下一状态。例如①状态 DISARMED 只要有

ACCESSORY条件存在就转移到②状态READY。处于②状态READY只要有UNTRIPED条件就回到READY,有TRIP条件就转移到③状态,有IGNITION条件就转移到①状态DISARMED。该表记录了全部状态。

#### (5) 写出积和逻辑式

因为选用的PAL16R4为负有效输出,故逻辑式用负逻辑。从状态转移表中右边E栏中取出触发器 $S_3, S_2, S_1, S_0$ 为0的情况写出触发器状态逻辑式。写逻辑式时,将B栏触发器状态与输入条件取积,然后全部“或”起来即可。化简后得到: ( $S_3$ 永远为1)

$$\begin{aligned}\overline{S_2} &= S_3 \cdot \overline{S_1} \cdot S_0 \cdot \overline{\text{TRIP-LOW}} \\ &\quad \cdot \overline{\text{IGNITION}} \\ &\quad + S_3 \cdot \overline{S_1} \cdot S_0 \cdot \text{TRIP-HIGH} \\ &\quad \cdot \overline{\text{IGNITION}} \\ &\quad + S_3 \cdot \overline{S_2} \cdot S_0 \cdot \overline{\text{IGNITION}} \\ \overline{S_1} &= S_3 \cdot S_2 \cdot \overline{S_1} \cdot \overline{\text{IGNITION}} + S_3 \cdot S_2 \cdot \overline{S_0} \\ &\quad \cdot \overline{\text{IGNITION}} \cdot \text{TRIP-LOW} \\ &\quad + S_3 \cdot S_2 \cdot \overline{S_0} \cdot \overline{\text{IGNITION}} \cdot \\ &\quad \text{TRIP-HIGH} \\ &\quad + S_3 \cdot \overline{S_1} \cdot S_0 \cdot \overline{\text{IGNITION}} \cdot \\ &\quad \text{TIMER-IN} \\ \overline{S_0} &= S_3 \cdot S_2 \cdot S_1 \cdot \overline{S_0} \cdot \overline{\text{IGNITION}} \\ &\quad + S_3 \cdot S_2 \cdot S_1 \cdot S_0 \cdot \text{ACCESSORY} \\ &\quad + S_3 \cdot S_2 \cdot \overline{S_0} \cdot \overline{\text{IGNITION}} \cdot \\ &\quad \overline{\text{TRIP-LOW}} \\ &\quad + S_3 \cdot S_2 \cdot \overline{S_0} \cdot \overline{\text{IGNITION}} \cdot \\ &\quad \text{TRIP-HIGH}\end{aligned}$$

如法炮制TONE、HORN、STAR-TIMER的逻辑式。如TONE有效时应是 $Y_3$ 为0的情况,也就是③状态WARNING或⑤状态ALERT。

$$\overline{\text{TONE}} = S_3 \cdot S_2 \cdot \overline{S_1} \cdot \overline{S_0} + S_3 \cdot \overline{S_2} \cdot \overline{S_1} \cdot S_0$$

同样有:

$$\overline{\text{HORN}} = S_3 \cdot \overline{S_2} \cdot S_1 \cdot S_0$$

$$\overline{\text{STAR-TIMER}} = S_3 \cdot S_2 + S_3 \cdot \overline{S_2} \cdot S_1 \cdot S_0$$

依照上述逻辑式就可在相应设备上编程16R4器件。

### 三、ABEL简介

ABEL是英语Advanced Boolean Expression Language的缩写,为美国Data I/O公司近年发表的逻辑阵列编程系统软件,源程序用C语言写成,共3万条,运行环境主要有UNIX,MS-DOS,CP/M-68等,随着不断完善,现在已能在个人机上高速运行。

设计者在使用ABEL进行设计时,要用ABEL格式描述逻辑电路的功能,对复杂的布尔代数式,真值表,时序电路的状态转移图可原样输入,ABEL均能处理。再根据该设计式进行仿真,检查设计错误。然后,将仿真器通过的逻辑式变换成逻辑阵列熔断状态图形。该图形以文件形式存于软盘上,送入编程器。编程器按图形进行作业,最后产生测试向量进行测试以完成全部工作。

由于篇幅所限,有关ABEL软件详细内容请读者参阅其它资料。

表1

状 态 转 移 表

当前状态			输入条件 (状态转移条件)	转移状态			输出			行
号	状态名	S <sub>3</sub> , S <sub>2</sub> , S <sub>1</sub> , S <sub>0</sub>		号	状态名	S <sub>3</sub> , S <sub>2</sub> , S <sub>1</sub> , S <sub>0</sub>	Y <sub>3</sub>	Y <sub>2</sub>	Y <sub>1</sub>	
1	DISARMED	1, 1, 1, 1	IGNITION	1	DISARMED	1, 1, 1, 1	1	1	0	0
		1, 1, 1, 1	ACCESSORY	2	READY	1, 1, 1, 0				1
2	READY	1, 1, 1, 0	UNTRIPPED	2	READY	1, 1, 1, 0	1	1	0	2
		1, 1, 1, 0	TRIP	3	WARNING	1, 1, 0, 0				3
		1, 1, 1, 0	IGNITION	1	DISARMED	1, 1, 1, 1				4
		1, 1, 0, 0	TRIP	3	WARNING	1, 1, 0, 0				5
3	WARNING	1, 1, 0, 0	UNTRIPPED	4	ARMED	1, 1, 0, 1	0	1	0	6
		1, 1, 0, 0	IGNITION	1	DISARMED	1, 1, 1, 1				7
		1, 1, 0, 1	UNTRIPPED	4	ARMED	1, 1, 0, 1				8
4	ARMED	1, 1, 0, 1	TRIP	5	ALERT	1, 0, 0, 1	1	1	0	9
		1, 1, 0, 1	IGNITION	1	DISARMED	1, 1, 1, 1				10
		1, 0, 0, 1	WFT*	5	ALERT	1, 0, 0, 1				11
5	ALERT	1, 0, 0, 1	TIMEOUT	6	ALARM	1, 0, 1, 1	0	1	1	12
		1, 0, 0, 1	IGNITION	1	DISARMED	1, 1, 1, 1				13
		1, 0, 1, 1	IGNITION	6	ALART	1, 0, 1, 1				1
6	ALARM	1, 0, 1, 1	IGNITION	1	DISARMED	1, 1, 1, 1	15			
A		B	C	D		E	F			

注: WFT\* 即WAIT—FOR—TIMEOUT

 $Y_1$  = START—TIMER(正逻辑) $Y_2$  = HONE(负逻辑) $Y_3$  = TONE (负逻辑)

(参考资料略)

天津市电子计算机研究所

孙 晓 陈荣华 编译

黄 侃 校

# 实现我国金融技术手段现代化的支撑环境

杭承仁 杨润征

(电子工业部计算机与信息发展研究中心)

## 一、金融电子化系统是一个多层次、网络型的复杂系统

世界各国金融技术手段的现代化无不是以计算机技术为基础的。据有关方面统计,美国金融界装备的大中小型计算机超过 16500 台,日本也有5000台以上的这类计算机在金融网中运行。为了实现这些装备,所花的费用是惊人的。以日本为例,金融界的计算机装备费用占全国计算机装备总费用的17.6%,为各行业之首。金融技术手段的现代化就是计算机化,或广义地称电子化。

我国的金融业是一个庞大的社会实体。除了作为中央银行的人民银行,还设有工商银行、中国银行、建设银行、农业银行、交通银行等专业银行以及保险公司,它们一般都有着全国、省市、县、区、乡镇等多层次的机构,在全国各城市、乡镇分布了13万个金融点,30万个信用社,它们在纵向和横向都有着大量信息交换。我国的金融系统是一个多层次、网络型的庞大而复杂的系统。

七十年代以来,我国的金融界陆续装备了一些进口的和国产的计算机,在一些大城市首先开展了一些以计算机为手段的业务,但大部分工作还处于手工操作状态。

为了解决金融电子化问题,首先应对银行的基本业务和对环境的基本要求做一些分析。

银行业务基本可分为以下三类:

1.柜台业务。这是直接面向顾客的数量巨大业务,包括储蓄、支票、信贷等。

2.资金清算。包括资金划拨、汇兑、结算等。

3.金融信息处理。包括金融信息动态分析、综合、预测、宏观控制决策等。

为了实现上述业务的电子化,需要下列四类产品:

1.各种金融终端,包括由银行人员使用的柜员工作站,以微机为基础的智能终端和相应的终端控制器,以及直接面向顾客的自动存取款机(ATM)和由非银行业务人员使用的销售点终端(POS)等。

2.不同档次的计算机系统。

3.通信线路及交换设备。

4.各种金融软件。

下面将就装备的引进和国产化,通信设施等问题进行分析。

## 二、金融电子化装备的引进和国产化

国外金融电子发展迅速,不引进部分国外装备,缩小同世界水平的差距是不可能的。但是,引进的目的不只是为了解决银行当前的装备,还要通过引进取得必要的技术,提高国内计算机研究和生产的水平,达到提高自给率的目的。忘记这一点就不可能真正

实现我国金融电子化的目标，因为要国家拿出巨额外汇支持全部金融电子设备的进口是不可能的。

据银行方面预计，“七五”期间金融电子化装备进口需外汇3.77亿美元，配套人民币35亿元。同期国产设备占进口设备金额的40%左右。如何提高国产装备的比例，如何通过进口取得必要的技术以提高自给率，哪些装备主要依靠进口，哪些主要立足国内，哪些产品要优先开发，哪些可以暂时放一放是我们关心的问题。同时，需要制定正确的政策来解决这些问题。

技术和贸易相结合的原则是我国引进先进技术产品的一贯原则，也是引进金融电子化装备的原则。应当规定，在引进这些产品时工业部门要同银行一道参加同国外厂商的技术谈判和商务谈判，一道派人参加出国考察和培训，并把国外厂商向我方提供必要的制造技术和关键部件，公开某些我们必要的终端和主机的接口技术，作为我方购买他们的产品不可缺少的交换条件，达到以市场换技术的目的，要在这一原则指导下制定相应的政策。

把市场大、技术难度较低或经过努力可以突破的产品放在国内优先发展的地位，尽快实现国产化，是我国金融电子化装备的另一项重要原则。

据银行方面预测，“七五”期间金融系统需装备中大型机126台，14,600万美元；小型机(含中型低档机)56台，2,650万美元；微型机6000套，9,474万美元；各类终端3072套，22,302万美元。图1为各类产品的总额、数量和平均单价。

由图1可知，按所占金额，各类终端第一(45.5%)，中大型机第二(29.8%)，微型机第三(19.3%)，小型机第四(5.4%)。

每台产品的平均价格在一般情况下可作为生产难易程度的指标。根据这个条件，各类产品由易到难依次为：微型机、各类终

端、小型机、中大型机。

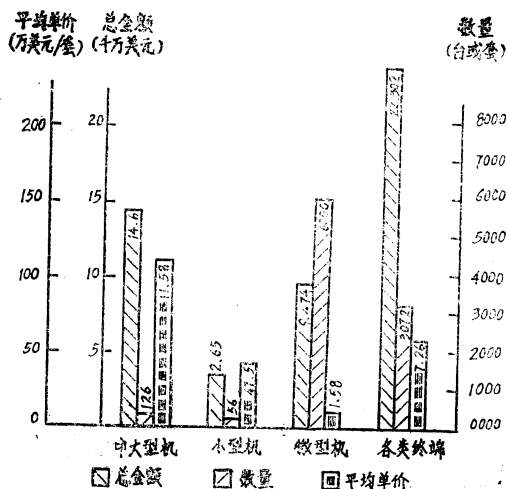


图1. “七五”期间银行所需的各类计算机产品的金额、数量和平均单价

各类金融终端占有最大的总金额，自然会吸引我们最大的注意。这些终端包括终端控制器、柜员工作站(含显示器、键盘、存折打印机、磁条读书机、磁卡机等)、自动存取款机(ATM)和销售点终端(POS)等，当前我们可以到手的材料，只提供了ATM和POS的需求量和总金额，仅根据这些材料要做出详细分析是不可能的。但可大体了解，在各类终端中ATM的难度较高，POS的数量比ATM多，难度也较低；柜员工作站当中虽然包含的品种较多，却是终端设备中数量最多、难度最低的部分。

金融终端的研制可以国外产品为蓝本。不同公司的产品，在功能相似的情况下，应以是否容易实现作为取舍的标准。表1对国外五家公司的终端产品在配置灵活性、价格、联机适应性、软件的多少和制造的难易程度方面进行了比较。很显然ISC的PIN-NACLE系列和PHILIPS的PIS6000系列具有较好的综合指标。

金融系统所需的中大型机虽占有较多的金额，但由于难度较高，在相当时期内需依

表1

各厂商金融终端的比较

厂 商	系 统	配置灵活性	价 格	联机适应性	软 件	制造的难易
ISC	PINNACLE	较高	低	好	一般	易
PHILIPS	PTS6000	高	低	好	一般	较易
优利	FSA	高	中	较好	一般	一般
IBM	4700	一般	高	一般	多	难
日立	FBT	较差	高	一般	少	难

靠进口是无疑的。但 2708、2760、2730 在 1986-1987 年分别在北京、上海、广州投入生产,表明小型机生产在国内已有基础,应立足国内。

八十年代以来,由于超级小型机的崛起和群集(Clusler)技术的发展,国外金融系统中的中大型机有被超级小型机替代的趋势。在金融电子化的体系设计中,应充分考虑这种因素。性能相当于 IBM4361 的 8060 计算机已通过鉴定,表明我国已有自己的中型机种。鼓励用国产超级小型机加上群集技术以及积极采用国内已经有的中型机,代替进口的中大型机应成为我们的一项政策。

根据 1986 年底的统计,我国已引进或自建并投入运行的微型机生产装配线 21 条,年设计生产能力 194,800 台;打印机生产装配线 6 条,年设计生产能力 6.7 万台;显示终端生产装配线 6 条,年设计生产能力 8.6 万台;磁盘机生产装配线 9 条,年设计生产能力 35 万台;小型机生产装配线 10 条,年设计生产能力 1600 台。可见,我国已经具有相当规模的计算机生产能力。这说明以上关于微机、终端及小型机等应立足于国内的分析是具有物质基础的。只要能根据银行电子化的具体要求,在技术上作一定的调整和补充,就可将这种分析转变为现实。

综上所述,可以得出以下结论:

1. 中大型机在相当时期内主要依靠进

口,小型机以国产为主。在体系设计时,要研究以小型机和群集技术代替中大型机,以减少中大型机进口的可能性,并在政策上予以鼓励。

2. 微型机应立足国内,一般情况下不得进口。国产微机要形成单用户、多用户、金融智能化终端型系列。

3. 金融终端要以国产为主。当前,可进口一部分解决急需;进口时,要认真贯彻技贸结合的原则;进口要为国内生产创造条件。

4. 慎重而有决断的选型,对于银行电子化的发展和国内计算机产业发展都是至关重要的。金融终端的仿制以 ISC 的 PINNACLE 系列和 PHILIPS 的 PTS6000 系列为目标,仿制的重点是终端控制器、设备控制器以及包括显示器、键盘、存折打印机、磁条读写机、磁卡机等在内的柜员工作站,力争早形成国产的成龙配套系列,同时抓紧销售点终端 POS 的研制。自动存取款机 ATM 的研制作作为下一步的目标,目前先依靠进口。

### 三、金融电子化系统的通信设施

金融电子化系统的网络化目标分为三个实施阶段:

1990 年底前在北京、上海等 36 个城市建立全市规模的计算机网;

1995年底前在累计196个大中城市建立计算中心并全部联网；

力争到2000年实现全国累计约300个大中小城市全部联网。

这个目标对计算机通信、通信线路和交换设备提出了严峻的要求。

在我国，公共交换电话网是当前最主要的通信网，除拉萨、乌鲁木齐外，所有省、市、自治区首府都安装了具有自动拨号功能的交换机或终端设备，可通过自动拨号电话网实现传输速率为300—1200bps的市内或城市间的数据传输，误码率为 $(0.26-4.23) \times 10^{-4}$ （步进制交换网，传输速率为300bps时）到 $(7.4-8.7) \times 10^{-5}$ （纵横制交换网，传输速率为600/1200bps时）。采用纠错控制后，误码率可降低到 $10^{-5}-10^{-6}$ 以下。安装了程控交换机的城市，传输速率可提高到1200—2400bps，误码率也显著降低。

用户电报及低速数据网是我国当前的另一个主要通信手段。到目前为止，除乌鲁木齐和呼和浩特外，所有省会、自治区首府及十四个开放城市，均已安装了中小容量的数据交换机，开放了以用户电报为主兼有少量数据用户的50bps的业务，而300bps的业务即将开放。“七五”期间，中心城市以上地区都安装用户电报和低速数据交换机，开放300bps的用户，并在县的范围开放50bps的用户。在经济发达的县，300bps的用户可通过调制/解调器挂到中心城市交换机上，50—300bps传输的误码率为 $10^{-4}$ （无纠错控制）或 $10^{-5}$ （有纠错控制）。

国家公用分组交换网是已列入计划的项目。“七五”期间将首先在北京、上海、广州三个城市建立三个结点，在沈阳、天津、南京、武汉安装远程集中器；在此基础上，天津、武汉、沈阳、西安、成都（或重庆）将增建五个结点，在各省会、自治区首府和个别中心城市将装设远程集中器。这样，即可初步形成全国范围的分组交换网。

分组交换数据网可实现误码率为 $10^{-6}-10^{-10}$ 的高速数据传输。

分组交换数据网可与公共交换电话网、用户电报和低速数据交换网等互相交换，是今后要发展的综合业务数字网的重要组成部分。

金融电子化系统对传输速率和误码率都有很高的要求。公共交换电话网、用户电报和低速数据网在一定条件下可以是金融电子化系统的传输手段，但难以满足它的全部要求。

分组数据交换网可以满足传输速率和质量的要求，但远水解不了近渴。

此外，以公共网作为金融电子化系统的通信手段使计算机犯罪的可能性增加，这是利用上述三种公共通信网存在的共同问题。

租用专线实现金融电子化系统的市内和城市之间的通信，一般情况下，传输速率可达2400bps，线路均衡后可达4800—9600bps。租用的线路可构成群路通道供传输速率达64—72Kbps的数据流传输。

租用专线的优点是传输质量高，但效率低、费用高，要求线路复盖全国。按照这种办法，最终将导致我国的天气预报系统、民航管理系统、国家经济信息系统、科技情报检索系统等成为各个独立的网络，形成管线组织混乱、调配困难、信息互通不易的局面。此外，许多建立在用户通过公用通信网直接访问金融电子化系统基础上的金融服务项目将难以实现。

在选择通信方案时，以下原则应当得到确认：

1. 在建立金融电子化通信网时，应采取立足现实、区别对待、分步实现、远近结合的方针。

2. 在可以接受的传输速率条件下，保证传输的正确性，是应该满足的最低要求。

3. 顾客可以通过公共通信网络取得银行提供的多种金融服务，应是系统的一个长远



目标。

4.应当尽可能利用公共的通信资源,避免重复建设。

5.金融信息网络最终应可能同我国的其它信息系统一样,成为国家信息通信网的一个组成部分,而不应成为各个独立的网络系统。

6.要确保系统运行的安全,能防止计算机犯罪活动。

根据以上分析我们建议:

1.在总体方案设计时,应充分考虑通信的要求。网络结构、编号制度、通信规程、编码方式应符合国家分组数据网的要求,向国际标准靠拢。在置机、建库时应注意满足这些要求。

2.在公共通信网能满足要求的地区,如安装了程控交换机城市,应尽可能利用公共通信网。

3.在通信条件不能满足要求的地区,先租用专用线路,包括市内和城市间的线路。

4.根据实际情况的不同,可采用联机通信,脱机数据传送,和磁盘邮递等多种并存的形式,实现点之间的数据传输,随着通信条件的改善,逐步向目标过渡。

5.改造市内和城市间的通信网状况,为金融电子化系统提供一个良好的通信环境。

6.加强防止计算机犯罪的技术手段和立法的研究,为系统运行的安全提供物质、技术和法律的保证。

## 《软件产业》月刊 1988年征订启事

《软件产业》是中国软件行业协会和中国软件技术公司主办的、由《软件产业》编辑部编辑出版的计算机软件产业中级技术性和技术管理性刊物,全国公开发行,以实用和迅速反映新动态为基本特点。

本刊的宗旨在于推动计算机在我国的应用,推动我国计算机软件产业的建立和发展,促进软件应用社会化、软件开发工程化、软件成果商品化和软件经营企业化。

本刊宣传我国计算机软件政策,评价国内外优秀商品化软件和优秀软件工具,介绍实现软件开发工程化、标准化的经验、软件产品质量评价方法,以及开发成本的核算方法等,并辟有《青协园地》、《软件法律案例》、《软件信息》、《软件资料题录》等栏目。本刊多为国内计算机软件名流和具有实践经验的软件科技人员撰稿。可供从事计算机软件研究、开发和经营单位、计算机用户,以及有关高等院校和中等技术学校师生参考和使用。

本刊由承德市邮局发行,全国各地邮局均可订阅。代号18—37。月刊,16开本,正文56页,每月15日出版,每期单价0.5元,全年12期订费6元。

《软件产业》编辑部

信箱:北京市945信箱

地址:北京市海淀区学院路37号

## 《微小型计算机开发与应用》编委会会议纪要

〔本刊讯〕1987年9月16日,《微小型计算机开发与应用》编辑部在天津市电子计算机研究所举行了本刊编委会会议。来自中外合资企业、大专院校、科研单位、工厂的编委委员和有关领导以及编辑部的同志共十七人出席会议。外地及本市部分因工作原因未能出席的编委,如电子工业部计算机工业管理局总工程师陈力为,清华大学计算机系教授房家固,以及天津市电子计算机研究所付总工程师夏业勋均送交了极有价值的书面发言。中国计算机行业协会秘书长、中国软件行业协会DG应用协会理事长夏纪寅,中国科学院软件所主任曹东启也表示了对会议的关注并请人转达了他们的意见。

会议由本刊主编邹秀凤同志主持。本刊总编、天津市电子计算机研究所总工程师黄侃同志首先作了讲话,他代表本刊向应邀参加会议的各位计算机专家表示感谢并宣布了会议的中心议题。

编辑部的同志汇报了该刊1987年的工作概况及下一步的设想,提出了关于1988年本刊出版方针及专题重点的初步意见。

编委们进行了热烈认真地讨论。中外合资企业——天津天芝通讯公司总经理童宣明同志谈到了国际上计算机技术的状况和市场导向问题,强调计算机的应用领域是十分广阔的,我们的思路应该放得更开些。天津大学教授王镭,天津计算机联合公司总工程师刘连棟,天津市电子计算机厂总工程师朱植松以及天津市电子计算机研究所付总工程师吴锦声、徐庭桂、黄宝良等同志都发表了颇有见地的讲话,从不同角度分析了计算机技术的发展状况,提出了《微小型计算机开发与应用》1988年的报导重点及报导方式。编委们一致表示愿意支持本刊工作,愿为办好它发挥作用。

会议最后确定:

- 1.由各位专家分别负责撰写和组织有关“计算机与通讯”,“专家系统”,“微电子技术改造传统产业”以及“CAD综述性报导”等方面的专题文章作为本刊1988年的出版重点。

- 2.“专题连载”是一种有利于中级以上水平读者深入探讨技术问题的好形式,可以采用,以满足计算机专业读者的需求。

- 3.从普及角度,将以非计算机专业出身的计算机应用工作者作为主要读者对象,刊登有经济效益的应用成果实例和经验。继续保持本刊“技术性”和“实用性”的风格。特别欢迎那些介绍设计思想及分析解决关键技术问题思路的好文章,以真正达到技术交流之目的,而不是一般的信息交流或成果报导。

会议结束时,本刊编委、天津市电子计算机研究所所长裴少峰同志讲了话。他感谢各位专家在百忙中出席会议,殷切希望大家共同努力,把《微小型计算机开发与应用》办得更加出色。