

微小型计算机



开发与应用

MICRO—MINICOMPUTER
DEVELOPMENTS & APPLICATIONS

1989 4

天津市显示器厂



天通电脑技术联合服务公司

—— 集高信息 高科技 高效益於一身

以中科院、通集团高科技开发为依托，凭天津电子计算机研究所数百名工程师为后盾，实现计算机从总体设计、系统配置、软件开发、人员培训到设备维修的应用一体化。

本公司下设：

技术·发部：设计安装3以太网网络，多用户微机系统，推出最新九国文字处理系统，CAD、轻印刷等应用软件。

维 修 中 心：专修IBM系列微机和各种兼容机主机板、专修单色、彩色显示器，PC电源、各种24针打印机、修配软、硬盘驱动器、维修空调机。
备有专用维修车，小修随叫随到，大修定期修复。

经营：

· 美国太平洋公司PSI-013GW/AT机

—— 兼容长城286、浪潮0530和IBM PC/AT於一机

配置：CPU80286-13MHz零等待；内存1MB

HD：40MB、50MB、20MB、80MB（28ms）高速硬盘可选

FD：1.2M、360K（日本东芝）

640×400 14吋高分辨彩色显示器配长城汉卡，可运行长城286系统、浪潮联想汉字和IBM/AT

全部软件：图形功能强

· 两串一並可接多用户及使用MODEM远程通讯

· 美国米兰达公司ST-286H（世界银行贷款项目中标机）

CPU80286-13.2MHz，内存1MB（可扩展到16MB）

HD：20MB、40MB可选 FD：360K×2，中高分辨彩显

· BST386多用户超级微机

CPU80386ΣΣ 时钟20MHz，内存2MB（可扩展到17MB）

HD：40MB（28ms），FD：1.2M+360K

800×600高分辨彩显，四用户卡，可运行CDOS PC-MOS或UNIX操作系统

· IBM PC/XT、IBMPC/XT286、IBMPS/2系列

· AR3240双字库高速打印机，MI724、VP8624打印机

· 美国山顿全稳压U.P.S电源 500W、1000W，美国IPS625型稳压U.P.S电源

· 经销VAX II超级小型机，太极2230超级小型机

· 备有各种仪器仪表，正牌3M、JANUS软盘，进口外设及配件，机房设备和工作台、椅

· 最新AUTOCAD10.1.0版、建筑CAD（PD）、地籍设计CAD、排版印刷系统，图书馆系统

地址：天津市河西区马场道津港路5号（计算机大世界）

电话：307186

联系人：鹿典忠 李庆宇

天津市显示器厂

国家定点 专业生产 现货供应 价廉物美

1. HTX-4 型高速彩色图形显示终端

640 × 400 分辨率: 20" CRT RS232 接口适合工业控制用。 28000元/台

2. 12" 绿色屏幕监视器

HM2233 型-H: 15.2 — 16.3 KHZ 900 线分辨率

可直接配用中华学习机和Apple-II机 560元/台 520元/100台

HM2233 "A" 型-H: 18.432 KHZ 可配IBM/PC. 配单色卡/ 彩色卡 任选
760 元/ 台 600 元/ 100 台

3. (原CM376 型) TX8040- II 型彩色 14" 监视器

H: 31.5 KHZ 640 × 480 分辨率 可直接配用长城386 、浪潮386 3600元/台

4. 日本原装进口夏普SHARP CU — 20PIDK 型彩色监视器20" 大屏幕

640 × 400 分辨率 H: 24.8 KHZ 可直接配用NEC 9801, 8801 等各类PC 机
13600元/台(8色) 15000元/台(16色) 18000元/台(64色)

5. HM-2244 型平板直角高分辨率单色(绿色、琥珀色、灰色任选)监视器14"

H: 15KHZ/ 18KHZ 1000 线 分辨率

可直接配用IBM PC/AT: 286 、386 机 1050 元/ 台

全国范围内寻求代理欢迎来人来函联系!

本厂已於今年5月迁往新地址办公, 一切业务来往函电、来人均请到新地址办理。

新地址: 天津市南开区中环线红旗路北草坝(电子计算机小区)

电 话: 754472 — 5 转 2 楼

电 挂: 2036

“天鹰”系列中英文终端机



天鹰终端机是天津电子计算机研究所独立研制并批量生产的产品。其中CE-1600型、CE-2000型和CE-2400型是最新研制的产品。产品结构合理、功能齐全、通用性好、操作简便、运行速度快、稳定可靠，是各类计算机的理想中英文终端。

三种型号的相同点：

- 15吋绿色/橙色CRT，25行显示，具有多种显示属性。
- 采用105键扩充键盘。
- 备有国标一、二级字库，可配全电报码字库，五千条词库。
- 有区位、五笔、笔型、拼音、大众、全电报等汉字输入法及词组输入法。
- 备有EIA RS232C/422, 20mA 电流环三种通讯接口，CENTRONICS和RS-232打印口，可实现24×24点阵优质打印。
- 传输速率75~19200波特。
- 系统仿真有D210/D211, VT52/VT100/VT220等多种格式。
- 终端参数均为软设置，全屏幕编辑一页，可扩至四页。
- 均可配图形功能软件包、自动译报软件，无限扩充造字功能。

不同点：

- CE-1600型汉字标准字库及显示均为15×16点阵。
- CE-2400型汉字字库及显示均为24×24点阵。
- CE-2000型配备矢量汉字库，有强的图形功能，可进行平移变换，比例变换，旋转，开窗，剪裁等操作，可配绘图仪，光笔等，系CAD优选终端之一。

以上产品备有现货，欢迎用户来人来函洽谈。

地址：天津市河西区友谊路爱民道5号 电报：3913 电话：332396

《编辑委员会

(总第42期)

顾问	郭平欣		
主编	黄侃		
副主编	王治宝	邹秀凤	
委员	(以下按姓氏笔划排列)		
	于万源	于清汶	王治宝
	王 镭	王士禧	王寿松
	付国明	许镇宇	朱植松
	曲庭维	李凤祥	刘连棣
	陈力为	邹秀凤	吴锦声
	房家国	夏纪寅	夏业勋
	袁维本	曹东启	黄侃
	黄宝良	章渭臣	梅克定
	童宣明	裴少峰	薛大中

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

天津市电子计算机学会

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:天津市武清陈咀乡印刷厂

订购处:全国各地邮局

定价:0.95

邮局代号 6—87 津工商广字0146号

国内统一刊号 CN12—1122

微小型计算机应用实例

一个变电站的微机监测系统

……………戴明鑫 孙淑信 陈前臣(2)

政府机关公文管理系统的设计与实施

……………张一鸣 张燕 王晶(5)

以三次样条函数逼近非线性,用微机

实现的方法……………马广成

……………胡继国 刘锡安(10)

部件与接口

微电脑计价秤的接口电路设计

……………袁惠源(13)

ROBOT—1机械手控制器剖析

……………戴齐(15)

不添加任何硬件扩充内存的一种方法

……………庄亚伦 王振明

……………陶忠明 谢忠新(19)

计算机软件

信息系统开发的几种有效方法

……………周勇(24)

微机上矩阵求秩的一种快速算法

……………朱容村(28)

对于稀疏型数据表的压缩方法

……………周望(30)

在应用软件中执行DOS命令的方法

……………孟宪福(33)

微型机信号处理系统中软接口的研究

……………陈宝珊(35)

一种实现UNIX系统与VAX/VMS系统间

异步通讯的软件……………魏军(40)

新技术

新一代关系数据库—FoxBASE+

……………周琼(44)

STE总线介绍……………陈晓午(46)

一个变电站的微机监测系统

华中理工大学电力系 戴明鑫 孙淑信 陈前臣

摘要 本文介绍一个变电站微机监测系统。内容包括算法, 硬件、软件, 几个辅助设计, 现场调试与结论。该装置使用方便, 设计合理, 数据准确, 具有较强的抗干扰能力。

一、前言

在电力系统中, 变电站数量多、分布广, 承担给用户传递电能的职能。因此, 提高变电站的运行及自动化水平, 是工农业发展也是电力工业发展的需要。目前, 我国变电站通常采用人工监视, 手动记录, 常规保护, 常规仪表。它们一方面布置分散, 结线复杂, 占地面积大, 更主要的在事故情况下缺乏必要的故障记录, 数据分析及处理能力。

微机在变电站中的应用, 开辟了变电站自动化的新路。它的优点是设备集中, 便于管理, 一机多用, 具有很强的数据处理能力。通过修改软件, 扩大功能, 能适合于变电站的发展和变化。

我们研制的变电站微机监测系统, 是变电站微机化的一个尝试。该装置安装于湖北省江陵县滩桥变电站。该站初建于1953年, 1987年由电力部农电司从英美引进重合器, 分段器, 熔断器按模式变电站更新改建。根据双方协议, 监测系统包括模拟量、开关量、电度、频率及变压器油温的实时采集; 接地故障的选线报警; 对主接线、参数、潮流的动态显示以及对越限、事故、定时, 召唤的打印制表报警。为提高装置的可靠性, 装置具有掉电保护、人机联系, 自诊断和自恢复的能力。

二、算法

1. 模拟量采用交流数据采集方式。只采集线路电流、电压的瞬时值, 由软件得到电流、电压、有功, 无功的有效值。同过去使用的采用直流变换器的方案相比, 增加了编制软件的难度, 但减少了硬件的投资。

对正弦电流 $i = I_m \sin \omega t$ 采用 90° 采样法得到二个随机采样值:

$$i_1 = I_m \sin(\omega t_1 + \phi_1)$$

$$i_2 = I_m \sin(\omega t_1 + \phi_1 + 90^\circ)$$

$$= I_m \cos(\omega t_1 + \phi_1)$$

$$i_1^2 + i_2^2 = I_m^2 [\sin^2(\omega t_1 + \phi_1) + \cos^2(\omega t_1 + \phi_1)]$$

$$= I_m^2$$

$$\text{电流有效值 } I = \sqrt{\frac{i_1^2 + i_2^2}{2}}$$

同理

$$U = \sqrt{\frac{U_1^2 + U_2^2}{2}}$$

$$P = i_1 u_1 + i_2 u_2$$

$$Q = u_1 i_2 - u_2 i_1$$

式中 ϕ_1 —— 初相角, I_m —— 正弦波幅值。

I 、 U 、 P 、 Q 分别代表电流、电压、有功、无功的有效值。

2. 开关量的采集计算需判断三种状态:

(1) 开关目前所处的状态; (2) 开关的变位情况即开关由分→合, 还是由合→分; (3) 开关动作的前后次序。对无法采集到开关量的熔断器, 可利用模拟量的有无反映其通断。

开关量的监测采用逻辑算法, TMC-80A 是 8 位机, 每字节 8 位, 反映 8 个开关的

状态

上次采集到的开关量用A表示，本次采集到的开关量用B表示。经简单的“异或”运算 $C = A \oplus B$ 得到C，如C为0，则表明8个开关均没有动作，如不为0，为1，则为1的数位对应的开关发生了变位动作。

判断开关由合 \rightarrow 分，还是由分 \rightarrow 合则需进行“与”运算。

$$D_1 = A \oplus B \wedge A$$

$$D_7 = A \oplus B \wedge B$$

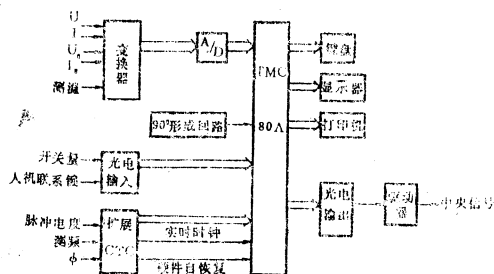
由 D_1 、 D_2 中数位为1的对应开关，分别表示开关的动作方向。

三、监测系统的构成

1. 硬件

主机TMC—80A, Z—80CPU, 内存48K显示缓冲区1K, 字符发生器2K, 监控程序和应用程序存放在4片8K的EPROM中。

硬件构成如图 1:



1

框中变换器包括电流, 电压、温度变换器。其作用是隔离, 消除干扰及完成与 A/D 接口的电平配合。

光电隔离输入输出板型号 LCB—64。
A/D板型号MS—1208,转换精度12位。
CTC扩展板, 90°形成回路板, 驱动器板。

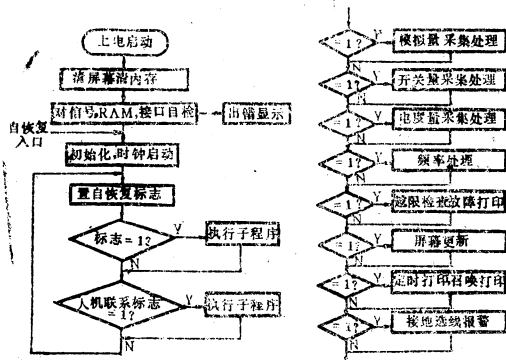
2. 软件

为保证系统的实时性,程序全部用Z-80汇编语言写成,并固化在2片8K的 EP-ROM中。

程序采用模块化结构,对每一独立功能

设计成独立模块。对于反复应用的基本子程序建立子程序库供各模块调用。

主程序框图如图 2



24 2

进入主程序首先进行清屏，自检和初始化。自检包括对内存，中央讯号，接口的自检。初始化包括对CTC，PTO，内存初始化。在正常情况下，整个程序在主程序的指挥下有条不紊的运行。只要中断开放，执行中允许中断子程序插入。当程序飞出可执行区或死锁后，装置依靠硬软件自恢复功能，进入自恢复入口，重新启动运行。

主程序通过起动中断，查询标志来调用和连接各子程序（独立模块）。图2系主程序中第一个菱形的详图。标志由实时时钟，子程序，人机联系键共同放置，由于子程序执行时间间隔不同设立标志单元增加了主程序的伸缩性。

特别应当强调指出,对重要的少数开关量采集应该由中断子程序进行。本程序中断功能模块包括:①系统实时时钟,每隔10~20ms采集重要开关量一次,同时可以置定时打印等标志;②CTC测频中断;③90°采样回路中断。

人机联系子程序包括, 修改时间电度, 检查光字牌, 强制清光牌报警, 强制返回监控程序, 切换不同屏幕等。

在RAM中, 程序开辟数据区, 工作区, 标志单元, 在主设备异常时为加强微机监视, 固化在EPROM中的整定值可以在RAM中临时修改。

四、几个辅助设计

1. 由算法可以看出, 如果不能保证模拟量两瞬时采样值 90° 的相位差, 测值会有一些的误差。为此, 该装置设有 90° 采样形成回路, 利用串联电路中电阻与电容上电压差 90° 的特点, 产生两个矩形触发脉冲, 分别用中断方式控制采样。这样, 装置的采样间隔随系统的频率变化而变化, 或者说装置对频率有自适应能力。

2. 在进入 A/D 转换以前, 模拟量先经过了硬件去耦, 滤波技术, 以滤去回路中的高频分量。为进一步滤去脉冲干扰, 在软件中又采用了中位值数字滤波技术。在具体实施中有二种方法可供选择, 一种是对几组计算结果滤波, 这方法直观, 易理解, 但运算量极大, 占机时多; 另一种方法是先找出采样值的规律, 再利用中位值滤波技术得到采样数据的中位值。实践证明, 后者满足了实时和数据精度的要求。

3. 在系统突然停电时, 为使脉冲电度表测得的数据不致在内存中丢失, 我们用二节干电池实现对RAM数据区的掉电保护, 正常时RAM由主机电源供电。当主机电源低于3V, 由于电池提供电力保存数据, 因干电池经电阻直接接到芯片供电范围小, 二节干电池可使用几个月且更换方便。

4. 利用CTC计数法测电度, 由于CTC是采用脉冲边沿触发, 易受脉冲信号的“毛刺”误触发, 使计数值无规律的跳动, 硬件

中需采用去抖整形措施。

5. TMC—80A配有12寸单色显示器, 主机有EPROM 2716作为字符发生器, 可写入256个 8×6 点阵的汉字, 符号, ASCII码。对于复杂的汉字, 我们采用8个单元写2个 8×8 点阵的汉字, 可满足显示的要求。

打印机采用点阵式, 利用位图码, 内存中建立汉字库, 打印图形字符并不困难。

五、现场调试与结论

变电站控制室仍采用屏合结构, 微机安放在台内, 显示器则位于台中央正前方, 为防止微机出万一, 两侧布置了必要的表计。模拟量、开关量用信号电缆引入, 为吸收随导线而来的干扰, 接入变换器原方的线间和对地加装了电容, 付方加装了滤波回路。参数的选择应使原付方相移一致, 以提高测量精度。开关量经光电隔离, 抗干扰能力强, 但作为引入的辅助接点易氧化, 增加了接触电阻, 我们采用重复继电器的方案。

目前变电站微机化是一个较热门的课题, 国内已取得一批成果。微机监测系统的生命力在于开发软件, 逐步的与通讯、远动结合起来。

我们研制的微机监测系统, 自今年初调试投运以来, 主机正常, 接口基本可靠。元件没有因雷雨和湖北持续高温而损坏。它的投入提高了变电站的自动化水平, 普及了微机技术, 因而得到用户的欢迎。

政府机关公文管理系统的设计与实施

天津市电子计算机研究所 张一鸣 张燕 王晶

摘要 利用计算机实现公文管理,是政府机关办公自动化的一项主要内容。本文讨论了设计政府机关办公自动化系统中应考虑的问题、解决途径和数据结构模型,并且介绍了一个成功地投入实际应用的公文管理系统。

一、概述

利用计算机实现公文管理,是政府机关办公自动化的一项主要内容。实现公文管理的计算机化,不仅体现在提高公文处理的准确性,使用效率和利用程度上,而且有更重要的政治经济意义。

1988年,我们与国务院办公厅秘书局合作,在美国DG公司的32位超级小型机MV上开发了计算机公文管理系统(简称ADS),并投入运行,实现了对各类公文进行登记、查询、统计、修改、打印和系统维护的计算机管理,效果良好。本文就是在其基础上归纳总结而成的。

二、ADS设计的指导思想

从宏观上看,主要有公文信息的输入、综合处理(检索、统计、查询等)、输出三部分构成。一个高效率的计算机公文管理系统应当在各个环节上快速有效,并尽可能地与原有的工作方式兼容,减少碰撞。因此,我们认为,建设一个政府机关公文管理系统的指导思想是:照顾兼容性,突出检索性,提高可用性,加强维护性。

三、ADS设计中的几点考虑

1. 操作的复杂性

公文管理和一般的信息管理系统不同,如档案管理或情报检索系统的数据是一次录入的,可以多次查询。而公文数据的录入要复杂的多。对收到的公文要首先进行一次登记,录入:收文号、来文日期、份数、来文单位、主题词、标题等表示公文各种特征标记的信息。登记后的公文要送去办理,为了掌握它在系统内的流向和处理情况;又要把每次送办过程进行再次登记。在送办过程中,经办人员和决策人员会给出明确的处理意见,这也是再次登记的内容。在必要时,还要录入公文摘要和最后公文分送的去向等,以供检查。整个公文处理过程除了开始输入和最后检索、统计、查询输出外,中间各次经办近似一个闭环的循环系统。如图1。

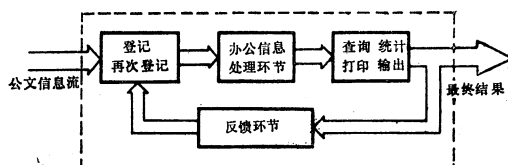


图1 公文信息流模型

正是由于公文处理的自身循环的特殊性,给计算机公文管理系统的设计和实施都增加了复杂性和困难性。对数据库的更新、修改和检索提出了很高的要求。

2. 数据库结构

由上面的叙述可知,一个公文登记的内

容包括公文特征信息（编号、日期、来文单位、主题词等）和送办过程、批示内容、公文摘要、分送记录等内容信息共五部分。相应地，一个公文数据库也由主库、送办库、批示库、摘要库、分送库五个库文件构成，按照先后次序，每份公文形成一个主记录，并在送办、批示过程中产生若干个送办、批示记录，主记录和送办记录，送办记录和批示记录之间的关系都是一对多的对应关系。在需要时，主记录还会派生出唯一的摘要记录和分送记录。如图2所示。

为了完成数据的检索，要指定数据记录中各种特征信息为索引字段，并自动抽取出建立起各种用途和格式的索引文件。如主库文件就建立了收文号、来文日期、来文文号、承办单位、发文号、领导人文号、办完日期等8个索引文件。如图3。这样，就为用户提供了多种多样灵活快速的检索途径。当然，在用户对公文数据做修改、再次登记时，若涉及到记录中的索引字段，则除了改写主库记录外，也要相应地修改其在索引文件中的内容，以保持数据的一致性和完整性，这无疑增加了系统本身和程序处理的难度。但是从整个公文系统要突出检索性这个关键要求来看，这种代价是值得的。

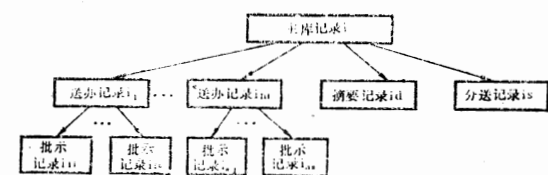


图2 五种数据记录的关系

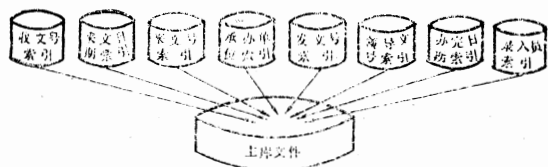


图3.主库文件和其索引文件

图2中 $m = \{1|2|\dots|N\}$ N 为一个100以内的正整数

$n = \{0|1|\dots|N\}$

s 或 $d = \{0|1\}$

3. 用户界面设计

由于公文管理的数据不仅输入频度高，而且内容多。因此用户界面是一个很重要的问题。为了使用户操作安全、方便、快速。操作使用选单来实现这些要求。一方面，我们设计了“捷径”，即按特定键后可以不经过多层选单而直接返回主选单或某些具体功能；另一方面，对命令进行合法性检查，并允许用户在执行时可以中断操作。

再有，我们专门开发了窗口编辑软件，可在一屏幕上支持最多50个窗口，数据在各自窗口内编辑或显示，在窗口内上下滚动以便查看和增、删、改。并可以对指定窗口内的数据和日期、来文单位、主题词等进行核实，不对的要求修改。否则不能退出该窗口。避免了反复问正确与否的对话。用户还可以在各窗口之间移动，随意查看。这些功能的提供，使得用户界面变得更加友好。

4. 空间的划分和保护

对于一个大型的多用户计算机系统，往往有多个应用软件和用户同时使用系统的资源，产生大量的数据。如果不采取保护措施，一旦失误，将造成不可弥补的损失。因此，在ADS设计中，我们采取了下述重要措施：

(1) 划分存储空间

按照系统资源、应用程序、数据相分离的原则，我们在保证系统资源不受影响的前提下，把应用系统空间划分为三个独立的区域。其中可执行程序、错误信息文件等存放在专门建立的：UTIL:ADS目录中，做为应用程序区；全部数据库和索引文件放入专门建立的：DB:ADS中，而每年产生的数据做为：DB:ADS下的一个独立子目录，以便于数据转贮和恢复。各个目录通过限制用户访问权限，设置文件访问表等方法来保证合法访问。

(2) 数据安全性的保护

为了保证数据安全性，实现对多用户的有效管理，ADS把所有用户分为三类，即

系统管理员, ADS 录入员, 普通用户。各个级别的人员有不同的职责权限, 权限高的可执行权限低的工作, 反之则不行。

5. 数据操作的层次结构

从对数据操作处理的观点来看, ADS 可以分为三层, 即环境层, 数据加工层和用户层。每层中由若干独立的函数构成, 见图 4。从用户发出命令到解释执行, 返回结果, 上层函数根据需要调用下层中相应的函数, 逐级推进, 直至完成任务。这也使得增、改系统功能时, 只需在高层中增加、修改新功能函数即可, 实现了系统的模块化设计。

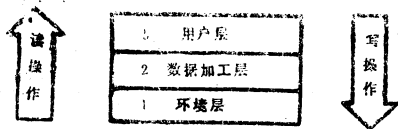


图4. 函数对数据操作的层次结构

环境层是ADS与I/O设备(如终端、磁盘), 数据库管理系统和操作系统的接口部分, 如窗口编辑软件, 读取路径、日期专用函数, 语言与数据库接口函数的参数、变量设置函数等。在这一层里, 一方面把处理层送来的内部数据和参数按接口的格式放好, 送给数据库或刷新窗口编辑缓冲区; 另一方面, 又要按要求从盘或窗口编辑缓冲区读出数据送给加工层。

数据加工层是对从用户层或环境层所得到的数据进行格式处理。当属于写操作时, 则把数据转换为内部格式, 以供环境层使用; 当属于读操作时, 则把数据转换成外部标准格式供用户显示用。

用户层是与用户打交道的, 包括各种菜单选择、数据录入、编辑、修改以及数据的格式显示、打印等。一方面, 它要接收输入信息, 如选择项、查询条件、公文数据等, 并把它送给下层处理; 另一方面, 又要把下层传来的数据在屏幕上显示出来。这一层

所处理的数据都是可显示的。

四、主要功能和结构

ADS由系统维护、数据录入和查询服务三个子系统构成。具体分为: 系统维护、登记、查询、修改、统计、打印、清单显示等七部分。如图5所示。

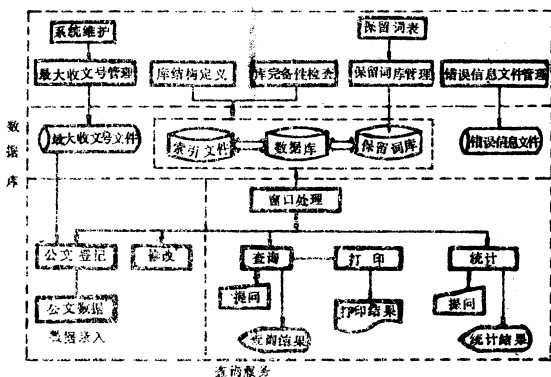


图5. 系统主要功能示意图

1. 系统维护子系统

它是整个系统运行的基础, 只能由ADS系统管理员执行。由于本系统的数据库设计为以年度为独立目录, 因而系统的备份和恢复可以直接使用AOS/VS操作系统提供的系统命令。系统维护包括:

(1) 建立库结构 为每年度各库建立全套数据库、索引文件和辅助数据文件结构。

(2) 检查数据文件完备性 核实数据库和索引文件是否齐全。

(3) 词库管理 完成主题词和来文单位索引的增、删、改、统计、输出等工作。

(4) 最大收文号管理 解决系统自动维护公文登记收文号的问题。

(5) 系统错误信息文件管理 完成出错提示信息的增、删、改、查看、打印等工作。

2. 数据录入子系统

它包括公文数据的第一次登记, 再次登记和修改三部分。主要供系统录入员执行。

3. 查询服务子系统

它包括查询、统计、打印、清单显示几部分,可以由全体ADS用户使用,也是使用频度最高的子系统。

(1) 查询

查询性能是ADS系统设计的一个重要课题。查询途径的多少,灵活程度,尤其是多用户环境下的检索速度是衡量其性能的一个关键指标。

我们在ADS系统中,提供了:

单一条件的精确查找

单一条件的近似查找。

复合条件的精确查询;

复合条件逻辑匹配的近似查询;

标题子串的查询;

共5类近20种检索途径。其中前4类是利用索引文件完成的,在十多个用户同时使用的情况下,任一种查询响应时间均小于1秒钟。后一种是用限制条件下的遍历库记录方法实现的。速度较前者稍慢。

(2) 统计

包括统计指定日期内的来文件数、办完件数、文件送办往返次数、来文单位报文次数、经办部门办理件数,以及录入员输入信息量等,用以考核各种指标,科学地统计、分析业务发展、工作量等。

(3) 打印

把查出的各种文件按规定格式和字型打印出来,以备留档或检查。

(4) 清单显示

相当于办文记录,能把前一天的新登文件显示出,以备优先处理。

五、检索策略和实现

检索在各类管理信息系统中都是非常重要的,对于公文管理系统(ADS)尤其如此。登录的文件正是为了提供查询、统计、分析工作用的;而打印、修改等工作的前提也是要检索出满足条件的公文子集。在ADS的

检索设计中,以主题词检索技术难度最大,结构最复杂,而解决的又最好。所以这里重点讨论一下主题词的检索策略和实现。

主题词检索就是从公文中抽出中心概念性词语加以规范化,并由此作为公文的主题标识和查找依据,比传统的分类法更为直观。它最初应用于情报检索领域,现在已经广泛被公文管理所采用。按规定,对每份公文要用若干条主题词来标引,如一份有关工厂企业搞改革,推行生产责任制的公文,会标有“工业”、“改革”、“责任制”等主题词。在检索时,用这些主题词一起就可以找出同时标有这些词的全部公文。这就要求有管理全部主题词的词库,能找出每条词所标引的每天的收文号个数和全部收文号。主题词本身的管理,每个主题词每天标引公文的统计数据和具体的收文号,特别是每词每天标引收文号个数的不确定性,使采用常规的索引技术无法解决。我们把主题词索引分为三级结构,即:

第一级:主题词索引 由主题词和它的代码有序地构成。主题词和其代码是按国家统一标准建立的。这样,每个词就有两个索引项,因而能提供相互独立的两种检索方式。

第二级:主题词记录 每个主题词和代码唯一地对应一个记录。记录中有367个域,前366个域与一年的366天一一对应。每个域中有该天标引收文号个数和收文号所在桶链的首桶号,第367域为全年标引收文号总和。这样,由词找到记录后,根据给定日期就可以得到某天中本词标引的收文号个数和所在地址。

第三级:桶链记录 主题词某天标引的全部收文号顺序写入桶的记录结构中。一桶放不下可申请新桶,各桶依次连接而成桶链。全部桶形成桶链文件。

当要进行查询时,要做下述四步工作:

第一步:用每个保留词到第一级中检索,找出它在第二级中的记录。

第二步：在第二级中，用给定的日期确定在相应子记录中的日域的集合，再到这些域中逐个读出每天标引数和所在首桶地址。

第三步：在第三级按每个地址读出桶链中的全部收号放入每词的收文号集合中，直至处理全部地址。

第四步：对所有的词查出的收文号进行逻辑和操作，求出交集，做为结果返回。其处理流程见图6。

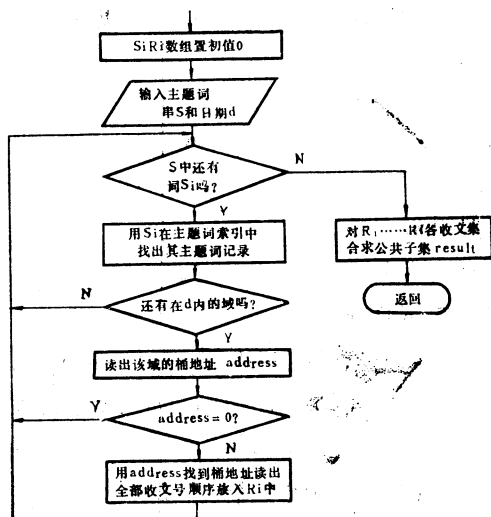


图6. 主题词检索流程

例如，要查1989年1月20日用“工业”“改革”标引的全部公文。假设该日共收六份公文，收文号为890021~890026。其中，用“工业”标引的有890021, 890023, 890024三份。用“改革”标引的有890023, 890024, 890026三份。如图7所示。在查询时，分别有“工业”标引的收文号集合 $A = \{ 896623$

890024, 890026 }，则两者的交集 $A \cap B = \{ 890023, 890024 \}$ ，查出两份。

为了实现主题词检索的速度，我们在第一级主题词索引中采用了B+树结构，所有叶子结点中包含全部关键字信息和记录指针，叶子结点本身依关键字值增序排列。非叶子结点是索引部分，其中仅含子树中最大关键字。我们选定的结点尺寸为4K，阶数（每结点关键字个数）为74。这样，当关键字小于8806项时，可保持二层结点结构，即根结点和叶结点二层，每层需一次访盘。在第二级主题词记录中，每个记录占一个结点，也只需一次访盘。

在第三级桶链中，涉及每词一天内标引的全部收文号桶链。按每天处理20份公文，（每年达5000份），每份公文用6个词标引则每天约有120个收文号进入桶链，形成大约100桶左右。它们都分布在2K的连续区域内，由于操作系统内存与盘交换数据以2K为单位，故也只需一次访盘即可读出全部收文号。

因此，每一保留词的检索，需要2（第一级）+ 1（第二级）+ 1（第三级）四次访盘。几乎是一个常数时间。对于采用虚拟分页存储的MV机来讲，读到的数据不是立即冲掉，而是看缓冲区有空就暂时存放，故当下一次检索时，若它所需的数据尚在内存，就不用访盘，无论对哪一层，都可以少一次I/O操作。因而实际上平均访盘约三次。这个速度是很理想的。

六、应用效果

ADS由于在多用户环境下其快速高效，安全可靠而受到用户欢迎，并迅速投入使用，为我国政府机关办公自动化做出了贡献。我们准备进一步完善优化，推广到更多的政府机关部门，提高政府机关的公文管理水平。

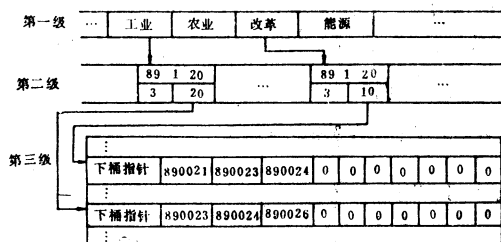


图7. 主题词库关系示意图

以三次样条函数逼近非线性, 用微机实现的方法

天津职业大学 马广成 胡继国 刘锡安

摘要 热电偶的热电势—温度 ($V-T$) 曲线, 是一条非线性曲线, 传统的热电偶测温, 一般是采用分段线性化的方法, 即将热电偶的 $V-T$ 曲线分成若干段, 将每段都用直线去近似代替曲线。分段越细精度越高。用微机处理这样的数据, 需将分段点的温度、热电势和斜率都存入计算机内, 这需要的存贮容量较大, 分段越多越是如此。并且更换热电偶时, 就更换大批数据, 显然通用性很差。

本文介绍在某冲天炉上所采用的, 以三次样条函数逼近热电偶 $V-T$ 曲线的方法。它与传统的分段线性化比较, 不仅具有分点少、精度高、占用内存少等优点, 而且具有一定通用性, 能对多种曲线进行高精度逼近。

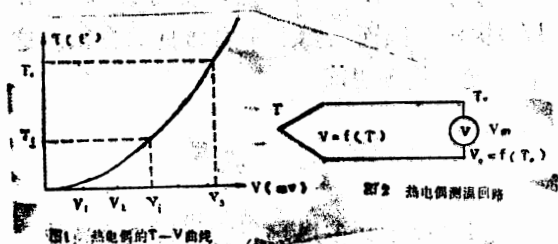
一、基本原理

热电偶的实际 $T-V$ 曲线如图1所示, 它近似为三次多项式, 所以用三次样条函数逼近它, 即设热电偶的 $T-V$ 特性为

$$T_i = \begin{cases} a + bV_i + cV_i^2 + dV_i^3 & V \geq V_i \\ 0 & V < V_i \end{cases} \quad (1)$$

式中 T_i 为热电偶的热端温度, V_i 为相应的热电势, a, b, c, d 为冷端温度为 0°C 时的一组常数。

图2是热电偶的测温回路



热电偶测温理论:

$$V_m = f(T) - f(T_0) = V - V_0 \quad (2)$$

$$\therefore V = V_m + V_0$$

上式说明 V_0 起着补偿作用, 因此 V_0 与 T_0 的关系虽可用三次样条函数逼近, 但温度却为自变量, 而热电势则为函数。即

$$V_0 = \begin{cases} a' + b'T_0 + c'T_0^2 + d'T_0^3 & T \geq T_0 \\ 0 & T < T_0 \end{cases} \quad (3)$$

由 (1), (2), (3) 式, 只要能求得最佳的常数 a, b, c, d 及 a', b', c', d' , 则可求得热电偶的热端温度, 由于最小二乘法广泛用于测量技术中, 所以, 我们求最佳常数时, 也应用了最小二乘法, 其原理为:

$$\sum_{i=1}^n e_i^2 = \min \quad (e_i \text{ 为残差}) \quad (4)$$

具体方法如下

设 $i = 1, 2, 3, \dots, i, \dots, n$ 且 $V \geq V_i$

由 (1) 式可得

$$T_1 = a + bV_1 + cV_1^2 + dV_1^3$$

$$T_2 = a + bV_2 + cV_2^2 + dV_2^3$$

$$T_3 = a + bV_3 + cV_3^2 + dV_3^3$$

$$\vdots$$

$V = f(T)$ 为热端温度相对于 0°C 的热电势值,

$V_0 = f(T_0)$ 为冷端温度相对于 0°C 的热电势值,

V_m 为电压表实际测得的热电势值, 根据

$$T_i = a + bV_i + cV_i^2 + dV_i^3$$

$$T_n = a + bV_n + cV_n^2 + dV_n^3$$

式中 T_i , V_i 为的相应真值, 可由热电偶分度表中查出。根据残差的定义:

$$e_i = T_i - (a + bV_i + cV_i^2 + dV_i^3)$$

$$\sum_{i=1}^n e_i^2 = \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)]^2 = \min \quad (5)$$

未 $\sum_{i=1}^n e_i^2$ 为最小的条件是使 $\sum_{i=1}^n e_i^2$ 的导数为零即

$$\begin{aligned} \frac{\partial}{\partial a} \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)]^2 \\ = -2 \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)] = 0 \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial b} \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)]^2 \\ = -2 \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)] V_i = 0 \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial c} \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)]^2 \\ = -2 \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)] V_i^2 = 0 \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial d} \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)]^2 \\ = -2 \sum_{i=1}^n [T_i - (a + bV_i + cV_i^2 + dV_i^3)] V_i^3 = 0 \end{aligned}$$

$$cV_i^2 + dV_i^3)] V_i^3 = 0$$

将以上四式进行整理得

$$\begin{aligned} a \sum_{i=1}^n V_i^0 + b \sum_{i=1}^n V_i^1 + c \sum_{i=1}^n V_i^2 + d \sum_{i=1}^n V_i^3 \\ = \sum_{i=1}^n T_i V_i^0 \\ a \sum_{i=1}^n V_i^1 + b \sum_{i=1}^n V_i^2 + c \sum_{i=1}^n V_i^3 + d \sum_{i=1}^n V_i^4 \\ = \sum_{i=1}^n T_i V_i^1 \\ a \sum_{i=1}^n V_i^2 + b \sum_{i=1}^n V_i^3 + c \sum_{i=1}^n V_i^4 + d \sum_{i=1}^n V_i^5 \\ = \sum_{i=1}^n T_i V_i^2 \\ a \sum_{i=1}^n V_i^3 + b \sum_{i=1}^n V_i^4 + c \sum_{i=1}^n V_i^5 + d \sum_{i=1}^n V_i^6 \\ = \sum_{i=1}^n T_i V_i^3 \end{aligned} \quad (6)$$

$$\text{令 } X_k = \sum_{i=1}^n V_i^k \quad (k=0, 1, 2, 3, 4, 5, 6)$$

$$Y_k = \sum_{i=1}^n T_i V_i^k \quad (k=0, 1, 2, 3)$$

则 (6) 式可简化为

$$\begin{cases} aX_0 + bX_1 + cX_2 + dX_3 = Y_0 \\ aX_1 + bX_2 + cX_3 + dX_4 = Y_1 \\ aX_2 + bX_3 + cX_4 + dX_5 = Y_2 \\ aX_3 + bX_4 + cX_5 + dX_6 = Y_3 \end{cases} \quad (7)$$

上式是一组线性方程组, 四个未知数四个方程, 故一般情况下有一组确定的解。

常数 a' , b' , c' , d' 也可用同样的方法求得。

二、计算常数的 BASIC 程序

图 3 是程序框图。

在执行程序前,首先要根据热电偶分度表将热电势、温度数据输入到计算机里,以铂铑₁₀-铂铑₉₀为例,测温范围为0~1800℃,每10℃取一点共取181个点,将这181个点平均分为四段,进行四次计算,得出各段常数a, b, c, d值。

三、接口电路和程序框图

1. 接口电路和地址分配表 在实时控制中一般都有一个确定的温度范围,同时为了降低控制系统的成本和充分发挥微机的效率,可选用不同的微机,本文介绍某冲天炉

Z80单板机温度采集部分的接口电路,它们温控范围在1300~1500℃.其接口如图4所示。

2. 温度计算程序框图和最终数据 因为实时控制系统中采用了冷端温度补偿措施,所以根据(2)、(3)式,首先要计算补偿电势 V_0 ,然后再计算温度 T 。在程序编制过程中采用了变形式,以加快速度和减小内存容量。所谓变形式即把式(2)、(3)分别变为:

$$T_1 = [(c + dV_1)V_1 + b]V_1 + a \quad V_1 \geq V_0$$

$$V_0 = [(c' + d'T_0)T_0 + b']T_0 + a' \quad T < T_0$$

表1为测试数据对照表

程序框图如图5。

表1 测试数据与热电偶分度表数据对照表

测试热电势 (mV)	测得温度 (°C)	分度表温度 (°C)	相对误差
4.8321	1000.0239	1000	0.00239%
5.7801	1100.0240	1100	0.00218%
...
13.5819	1800.0261	1800	0.00156%

四、结束语

由表1可见由三次样条函数逼近非线性最终结果与分度表的真值基本一样。它较传

统的分段线性化法精确的多。它不只适宜热电偶的非线性逼近,而且适用于多种非线性函数的逼近,具有广泛的推广价值。参考文献略。

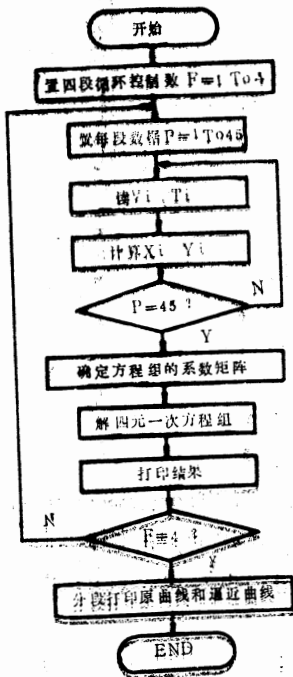


图3. 程序框图

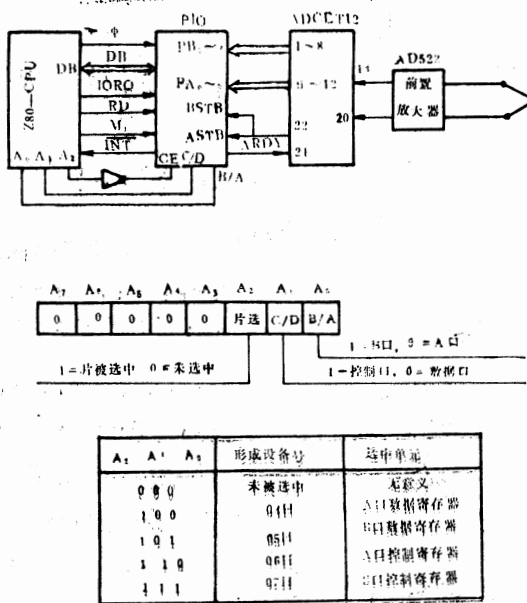


图4. 接口电路和地址分配

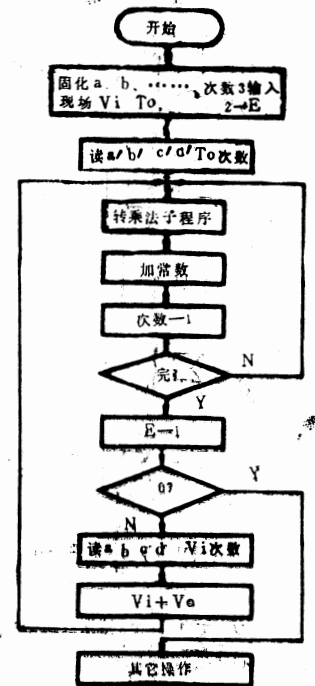


图5

微电脑计价秤的接口电路设计

上海原子核所 裘惠源

摘要 本文介绍了以Intel 8039为核心的商用计价秤的设计思想。控制电路以8039CPU的最小系统为主,扩展了一片8255并行接口。ADC采用4 1/2位BCD编码输出的7135,并以中断结合查询来采集重量数据。键盘显示电路采用扫描方式,以节省I/O口。单片微型计算机,由于在内部硬件电路及指令系统的设计上充分考虑了控制方面的要求,在国外又称微控制器。它具有直接的I/O接口能力和中断处理能力,并能进行一定的数据处理。因此,常作为精密电子秤的控制核心。本文着重介绍使用Intel公司的8039作为核心的计价电子秤的接口电路设计。

一、控制单元

它由8039CPU, 8D锁存器74LS373, EPROM2716, 扩展I/O口8255等组成。其电原理如图1所示。

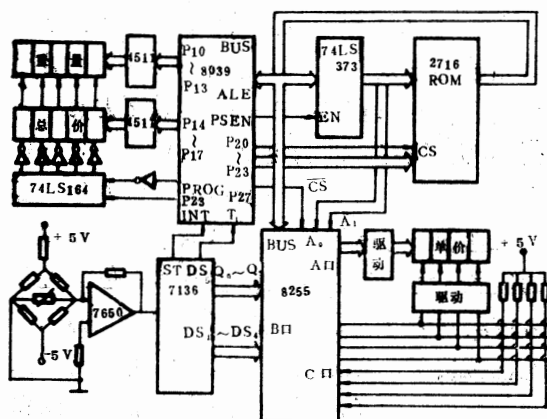


图1 单片微机电子秤电原理图

8039是MCS—48系列中无内部ROM的单片微型计算机，价格低，应用较广。但它必须采用外部扩展ROM，ROM的容量由应用系统的规模而定。由于电子秤的监控程序不长，故采用容量为2k字节的2716就足够了。8039的BUS总线是分时分出地址和数据的，所以必须在地址锁存使能信号到来时（ALE），将地址低八位打入地址锁存器74LS373中。8039本身还带有128个字节

的RAM，可以用来存放数据处理过程中数据，对于象电子秤这样的小系统，就不必再扩展RAM了。这里我们将并行I/O芯片8255作为片外RAM来扩展的（这一点与Z80明显地不同，下面还要解释），其A口作为显示单价的段选码口；C口的上半部（PC₀~PC₃）作为显示器LED的位选码口和键盘行码的输入口，C口的下半部（PC₄~PC₇）作为键盘列码的检测口；B口则作为ADC变换结果的数字输入口。

二、ADC采样接口电路

它由电阻应变片传感器检测电路, 自动
调零斩波运放电路, 4 1/2位BCD 码ADC
变换电路组成。

传感器检测电路是用电阻应变片组成的电桥电路，将重量信息转变为电压信息。称重时电桥平衡遭到破坏，在电桥输出端产生与重量成正比的电压信号。通过自动调零，斩波运放 7650 将这个微弱电压放大，送到 ADC 变换器。7135 是一种高精度（相当于二进制输出的 14 位—15 位）、低功耗，抗干扰能力强而转换速度较低（3—10 次/秒）的双积分型 CMOS 器件。因为称重过程要求

AD的采样速度不高,在称重10kg时,要求称重误差为 $\pm 5g$,因此采用7135完全可以满足电子秤的要求。7135的数据输出是分时分位的,即当DS5有效时, $Q_0 \sim Q_4$ 输出的是万位BCD码,当DS4有效时,输出 $Q_0 \sim Q_4$ 是千位的BCD码,依次类推。因此在硬件连结和AD采样中断程序的设计上都有它自己的特点。我们将7135的数字输出端 $Q_0 \sim Q_4$,数字驱动端DS₁~DS₄与8255的B口线一一对应相连,并将DS₅与8039的T₁相连。这样,就可采用中断加查询的方式来采样ADC的输入数据,也即将8039先置为可中断方式,将7135的选通端 \overline{ST} 接向8039的中断输入。当 \overline{ST} 端来一个负跳变(其宽度大于8039的三个周期),8039在完成当前指令后,即跳到003H,并由此转入外部中断服务程序。在中断服务程序中,单片机顺序查询DS₅~DS₁的状态,一旦状态为1,说明数据线上 $Q_0 \sim Q_4$ 的BCD码已经稳定有效,就将该位的数据送到显示缓冲区。这样每次中断就可采样一个重量数据。当第五次中断(个位)时,完成对数据的多次平均、判别、自动校零和其他数据处理功能。

三、显示和键盘扫描电路

商用计价秤一般要求显示5位重量,4位单价,5位总价,因此需要14位LED显示器。功能键一般设置为键入单价、清单价、去皮、累计存储、复位等功能,所以我们设置10个数字键和6个功能键。为了减少硬件电路,本机对8255A口和C口采用软件译码,动态扫描显示和扫描键盘的设计方案。并且用MOVZ指令对I/O芯片8255进行读写操作,也即是说将显示器和键盘接口电路当作单片机8039的扩充ROM使用。而对重量和总价,采用对PI口输出的BCD码进行硬件译码和利用P23移位寄存器方式动态扫描的设计方案。

单片机8039的BUS总线与8255的BUS

线一一对应相连,当 $p_{27}=0$ 时,选中8255,并将8255置成A口输出,B口输入,C口上半部输出,C口下半部输入的工作方式。8039将要显示的数字经软件译码得到的八段码从A口送出,经驱动后去点亮4位LED显示器(单价),以选择所要显示的字型(0~9数字及小数点)。8039还通过8255的C口上半部送出位选信息,经驱动后,去扫描点亮各位LED(若送出低电平,则与这位相连的LED被点亮)。这样,只要在8039的RAM区开辟一个显示缓冲区,在显示时,从左到右依次取出显示缓冲区的信息,查阅了在EPROM中的字模表后,将相应的八段码从8255的A口送出,再从8255的C口第一位送出零电平(其余各位送1),而后,使零电平在各位上按闭环右移方式循环,就可在4位LED上动态显示出放在显示缓冲区中的单价值了。

8255的C口连接由16个键盘组成的阵列(4×4)四根键盘线连接 $C_0 \sim C_3$ (行),另四根键盘线(列)一端接 $C_4 \sim C_7$,另一端通过10k电阻接在+5V。如无键按下, $C_4 \sim C_7$ 检测到的肯定是高电平。一旦有键按下, $C_4 \sim C_7$ 中某一根必定为低电平,此时根据键行号值就能查出零电平在位选线上的位置(零电平在位选线上右移一位,键行号值就加1),也就是说可以查出是哪一个键已被按下。如果是数字键,就可将对应的数字送入显示缓冲区。如果是功能键,则根据列号值算出键入程序的入口地址,从而转入相应的功能操作。

8039的PI口有八根数据线,分别与二个BCD—七段译码器4511相连、各自对应重量值和总价的BCD码(一个字节放二个BCD码,低半字节放重量,高半字节放总价),经过4511译码和驱动,将要显示的七段码作为LED的段选码。二组5位LED的位选端分别对应相连,并连到移位寄存器74LS164的五个输出端,将8039的PROG倒相后作为移位寄存器的时钟,而将8039的

ROBOT-1机械手控制器剖析

西南交通大学

戴 齐

摘要 本文对ROBOT-1机械手的控制器进行了全面剖析,着重分析了控制器的硬件配置、接口和控制软件,阐明了其控制方法。

一、ROBOT-1机械手概况

ROBOT-1机械手是日本AMUST计算机公司生产的一种小型机械手,它是以教学和科研为目的而设计的,主要用于教学演示、实验和模拟工业机械手。

该机械手具有5个自由度(5个旋转关节),并有一钳形抓手。机械手的结构如图1所示。它的驱动系统有七台步进电机和一台直流伺服电机。其中步进电机用于驱动各旋转关节,其步距角均为 0.1° ,伺服电机用于驱动抓手。

ROBOT-1机械手采用微机 and 控制器两级控制方式,即可以在微机上编程控制和示教,并由控制器接收和执行微机发出的控制指令。控制器由软件设置了5级速度控

制,使其控制方式和运动方式都可模拟现代大型工业机械手。

该机械手各关节的运动范围见表1。

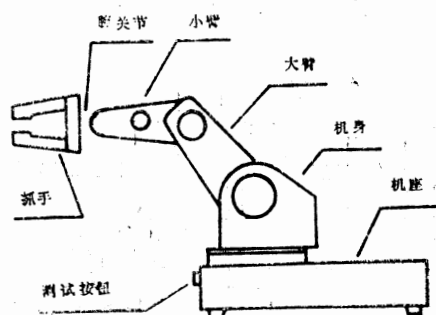


图1. ROBOT-1机械手结构图

P23端作为移位寄存器的串行输入脉冲。这时,我们借用了8039专为I/O扩展器8243设计的指令MOV_P, A, 将1电平在移位寄存器内逐位移动,可完成循环地将5位LED逐个点亮的操作,从而实现显示的动态扫

描。

参考文献

- 1.《Microcontroller Handbook》
Intel Corporation 1984
- 2.《单片机原理及接口技术》(内部教材)
上海交通大学 1987

表1 ROBOT-1机械手各关节的
运动范围

关节名称	运动范围
大臂俯仰	-45°~45°
小臂俯仰	-45°~45°
手腕俯仰	-90°~90°
手腕旋转	-360°~360°
机身旋转	-150°~150°

二、ROBOT-1机械手控制 器的硬件结构

ROBOT-1机械手的控制器安装在机座内,它是由一块Z80单板机及相应电路组成,采用全数字控制,其结构框图如图2所示。

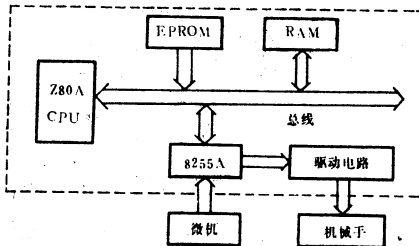


图2. ROBOT-1控制器结构框图

控制器的CPU是Z80A, CPU的任务是:

1. 与微机通讯。
2. 解释执行微机的控制指令。
3. 接收外部手动测试按钮的控制信号。
4. 控制机械手的各关节和抓手。

控制器的存储器现配备了4 KEPROM (2732) 和4 KRAM (5516), EPROM和RAM均可扩展到8 K。其存储地址如下:

内容	地址
控制程序EPROM	0000H
	0FFFH
用户RAM	3000H
	3E1EH
	3E1FH
系统RAM	3FFFH

控制器与外部的联系均由一块可编程I/O接口8255A实现。8255A被置成工作方式0, A口输出, B口输入, C口上半部(PC₁~PC₄)输出, 下半部(PC₅~PC₆)输入。从B口接收微机发出的控制指令, 用PC₅和PC₆控制数据的正确接收, 用PC₂输入外部手动测试按钮的状态。控制器对各电机的控制由A口和C口完成, 其中: 大臂的电机由PA₀和PA₁控制, 小臂由PA₂和PA₃控制, 腕关节的俯仰由PA₄和PA₅控制, 腕关节的旋转由PA₆和PA₇控制。机身由PC₀和PC₁控制。在每个关节的两个控制位中, 低位用于控制关节运动的方向, 高位用于发出控制脉冲。抓手的开/合由PC₄控制。

从8255A发出的对各电机的控制信号, 都需经过驱动电路, 根据各电机的功率大小, 对其控制信号进行功率放大。

三、ROBOT-1控制器与微机 之间的接口

1. 硬件接口

ROBOT-1机械手控制器可用电缆与微机的打印机接口相连, 这样就可以在微机上控制机械手的运动。由于打印机接口是并行接口, 数据传输的速率较高, 能够满足实时控制的要求。

控制器从打印机接口中共引出10条线, 其中包括7条数据线(传送7位ASCII码)和3条控制线(忙、认可和选通)。

控制器接收微机发出的数据采用查询方

式如图3所示。开机后,控制器通过8255A的PC₅发出一负脉冲,作为给打印机接口的认可信号,同时使D触发器Q端输出为低,表示此时微机可以发送数据。微机在发送数据时,先将数据送到7位数据线上,然后查询控制器是否忙,若不忙,则发选通信号,使D触发器Q端输出为高电平,表示此时控制器尚未接收完毕。Q端输出的低电平接8255A的PC₃,控制器通过查询该位是否为低电平,就可以知道此时有无数据待输入,若有则从B口读入,然后从PC₅发一负脉冲作接收认可信号,微机就可以发送下一待发数据。这样,控制器就能够接收微机发出的控制指令,每条指令以换行号为结束。

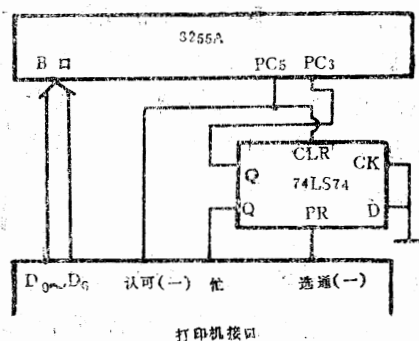


图3 控制器与微机的接口

2. 软件接口

ROBOT-1机械手向用户提供了12条控制指令。这些指令主要分为两类:控制类和示教类。

控制类指令是按用户给出的控制参数,控制机械手各关节和抓手。示教类指令是按用户的控制参数对机械手进行示教,示教后各控制参数存入控制器中,可以再现。

控制器所提供的这12条控制指令,把微机上的高级控制程序与控制器的低级控制程序联系起来,使两部分控制程序分工明确,为用户使用提供了方便。

四、ROBOT-1控制软件分析

ROBOT-1机械手控制器的EPROM中固化有4K控制程序,机械手的控制就是通过这些程序实现的。作者通过反汇编得到源程序,并对全部程序进行了详细分析。主程序框图见图4。

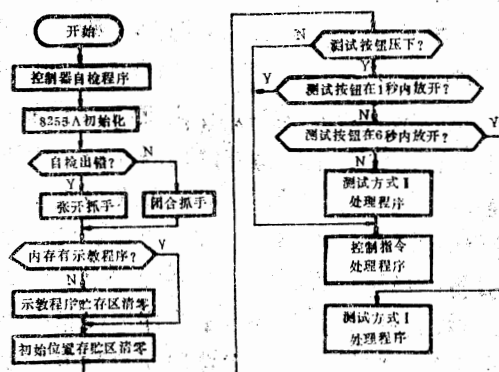


图4 主程序框图

整个程序可分成四大部分:控制器自检程序、测试方式I处理程序、测试方式II处理程序和控制指令处理程序。

控制器自检程序是在开机后对存储器进行检查,包括:1.ROM的检验和检查。2.RAM的内容是否被破坏。3.RAM容量检查。4.RAM各单元的读写检查。如果自检有错,则机械手抓手张开以告诉用户。

测试方式I处理程序用于查询外部手动测试按钮的状态。如测试按钮压下,则依次控制机械手各关节运动。这种工作方式主要用于检查各关节是否运动自如并设置初始位置。

测试方式Ⅰ处理程序是将控制器ROM中的一段数据作为控制参数,控制机械手表演若干动作。

控制指令处理程序是接收和执行微机发出的12种控制指令。当控制器接收到一行用ASCII码表示的数据后,先存放在内存的(3E1FH)~(3E9EH)单元中,然后取出前几个数据,判断是否为控制指令,若是则计算其相应的指令处理子程序的入口地址并转入该程序,否则将不执行。

在各个指令处理子程序中,若该指令带有控制参数,则要对其控制参数逐个进行语法和语义检查,并同时将其转换为16进制数,依次存放在内存的(3E9FH)~(3EA EH)单元中。对于各关节控制参数,用其最高位表示运动的方向(0为正向,1为负向)。语法检查主要检查参数的个数是否满足要求,分隔符是否正确等,语义检查要对参数的范围进行检查。若以上检查有错,控制器将拒绝执行该指令。在完成了上述参数处理后,各控制指令处理子程序将按指令的要求进行控制操作。由于控制指令较多,限于篇幅,不能在这里详细介绍。下面将分析在各控制指令处理子程序中被调用最多的一个子程序——电机控制子程序。

电机控制子程序非常重要,控制程序主要依靠它来控制各关节的运动和抓手的开/合,该程序将内存的(3ED9H)和(3EDBH)单元作为各关节的控制单元,其中(3ED-9H)单元的第0、1位控制大臂的运动,第2、3位控制小臂,第4、5位控制手腕的俯仰,第6、7位控制手腕的旋转,(3EDBH)单元的第6、7位控制机身的转动。在上述每个关节的两个控制位中,低位是关节运动的方向控制位,高位是关节运动的脉冲控制位。抓手的开/合由(3EDBH)单元的第4位控制。该程序所采用的控制方法是:首先根据各关节运动的方向置相应的方向控制位,将所有脉冲控制位置0,然后检

查各控制参数,若某项参数不为零,则将其相应的脉冲控制位置1,该参数减1。在置完各个脉冲控制位后,从8255A的A口和C口分别输出(8ED9H)和(3EDBH)单元之值,再将各脉冲控制位置0,仍从8255A的A口和C口分别输出(8ED9H)和(3EDBH)单元值。这样就相当于向相应的步进电机发出了一个控制脉冲,如此反复循环,直到所有的控制参数均为0。对抓手的控制只需置(3EDBH)单元第4位,抓手张开置1,闭合置0,并从8255A的C口输出(3EDBH)单元之值即可。

在控制器发出控制脉冲的间隙,进行不同的延时,则可控制脉冲的频率,也就可以控制关节运动的速度,控制程序设置了5级速度控制。此外,该程序还利用控制脉冲频率的方法,在机械手运动的起动阶段进行加速,制动阶段进行减速,其速度特性,分两种情况,当关节最大旋转角度大于 16° 时,关节运动有加速,匀速和减速3个阶段,否则只有加速和减速阶段。

五、评 价

以上我们主要从硬件和软件两方面对ROBOT-1机械手控制器进行了剖析。硬件结构紧凑,把主要的控制功能用软件实现,提高了可靠性,降低了成本,充分发挥软件的灵活功能,便于进行教学和实验。使用微机打印机并行接口发送控制指令,提高了数据传输速率,为实时控制提供了可能。该控制器所采用的控制方法,对于别的某些控制场合,也具有一定的实用价值。

该控制器设计的不足之处是,由于它采用开环控制,因此控制精度较差。其控制软件也尚需进一步改进完善。

参考文献

1. AMUST Computer Corp. Mini-Robot Simulator ROBOT-1 Owner's Manual, 等

不添加任何硬件扩充内存的一种方法

上海第六师范学校 庄亚伦 王振明 陶忠明 谢忠新

摘要 本文介绍了一种在Apple机上不添加任何硬件, 扩充内存的方法, 并介绍了DOS系统中与磁盘直接联系的RWTS程序及其使用方法, 本文附有具体扩充内存的实例及恢复DOS的实用程序。

一、概 述

一般标准的Apple II 系列8 位微机的内存空间为64K。其中ROM及I/O 为16K, 余下的48K为RAM空间。内存分配情况见图1。若用 Applesoft BASIC 语言编程时, 程序行从地址 \$0800 开始往上存放, 简单变量、下标变量及字符串变量紧跟程序末尾向上存放。磁盘操作系统 (DOS) 由内存地址\$9600~\$BFF连续存放, 所以, 用户真正可用空间为 \$0800~\$9600这一段空间, 约为36K。

我们在编写程序中发现, 当使用高分辨率显示区作为汉字显示时, 用户内存空间又将减少8K, 加上汉字系统一般占5~6K, 汉字小字库占4K (以100个汉字为例, 32byte 1字), 这样, 用户可用空间只剩下36-8-6-4=18K, 这对不少实用性程序来讲, 内存可用空间显然是不够的。

解决的一种办法是采取覆盖的方法, 也就是将一个大的程序分成几个片段, 每次调用一个片段, 依执行情况再调入另外某个片段, 后调入的程序与内存中当前运行的程序占用同一个贮存空间。我们在本文中介绍的方法, 实际上也属于程序覆盖的方法, 不过是采用了另外的覆盖方法。

二、DOS结构简析

在内存中从地址 \$9D00~\$BFFF 的DOS是由三大部分组成的。这三大部分分别

是命令处理接口程序 (简称DCI)、档案管理系统 (简称FMS) 及磁道扇区读写程序 (简称RWTS)。

其中DCI 的作用是接受、分析从用户传来的DOS 命令, 若命令无误, 则由它传给FMS, FMS通过RWTS与磁盘沟通。

DOS在内存中的分配如图2所示。

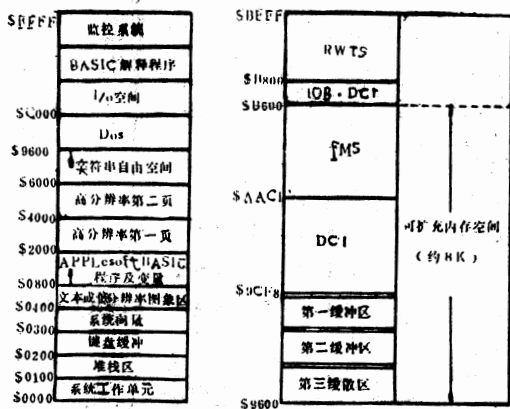


图1. 内存分配图

图2. DOS的内存安排

三、RWTS使用方法

如前所述, 10.5K长度的DOS 程序中与磁盘直接联系的实际上是长度为 2K 的RWTS, 这是DOS的内核程序, 它以一个扇区为单位对磁盘进行读写。

磁盘上每个扇区分成两个域: 地址域和数据域, 地址域包括盘卷号、磁道号、扇区号这些信息。数据域包含256个字节数据的代码。用户如果能正确调用 RWTS, 则对

磁盘上的任意磁道扇区都能进行读写。

RWTS 是一个用6502机器语言编写的子程序，其入口地址在\$03D9。调用RWTS前，应先设定两个表。它们是IOB表（I/O控制表）和DCT表（设备特征表）。

IOB 表包括如下一些信息：磁盘驱动器所在的扩充槽口号及驱动器号，磁盘的卷号，要访问的磁道和扇区号，读写缓冲区指针值，读或写盘的标志，设备特征表指针值等内容。

DCT表包括如下一些信息：设备类型，每道步进道数，马达运转时间计数的补码等内容。

两个表的具体内容如下：

1. IOB表：

字节(\$)	内 容
0	表类型，规定为\$01
1	扩充槽口号×16
2	驱动器号
3	卷号
4	要访问的磁道号
5	要访问的扇区号
6	设备特征表的低地址
7	设备特征表的高地址
8	读写缓冲区的低地址
9	读写缓冲区的高地址
A=B	未用
C	命令码（\$00=查找，\$01=读，\$02=写，\$04=格式化）
D	返回码（\$10=写保护，\$20=卷号不匹配，\$40=驱动器错，\$80=读出错，\$00=无错）
E	上次访问时用的卷号
F	上次访问时用的扩充槽口号×16
10	上次访问时用的驱动器号

2. DCT表：

字节(\$)	内 容
0	设备类型，规定为\$00
1	步进道数，规定为\$1
2-3	等候马达稳定的时间，规定为\$EF、\$D8

用户依上述要求分别设定好表内容后，应将IOB表的地址值放入Y和A寄存器中，低地址放入Y寄存器，高地址放入A累加器，

然后转RWTS入口地址\$3D9，调用RWTS。

例，用RWTS 直接读取 17 道 0 区的VTOC表（磁盘内容表，通常存放在17道0区）

假设扩充槽口号为6，驱动器号为1，IOB表从内存地址\$6000起存放，把17道0区的VTOC 表内容读到内存地址\$7000起存放。

第一步，先建立 IOB 表：（从\$6000起存放）

地址	内容	说 明
\$6000	01	设备类型，规定为 1
\$6001	60	扩充槽口号6×16
\$6002	01	驱动器号
\$6003	00	卷号
\$6004	11	17磁道
\$6005	00	0扇区
\$6006	11	设备特征表低地址
\$6007	60	设备特征表高地址
\$6008	00	读到内存 \$7000起存放
\$6009	70	
\$600A	00	未用
\$600B	00	未用
\$600C	01	命令码， \$01= 读
\$600D	00	出错码
\$600E	00	实际卷号
\$600F	60	前一次存取的扩充槽口号
\$6010	01	前一次存取的驱动器号

第二步，建立设备特征表。（紧跟IOB表，从\$6011存放）

地址	内容	说 明
\$6011	00	设备类型，规定为0
\$6012	01	每道步进道数，规定为1
\$6013	EF	} 马达稳定时间
\$6014	D8	

第三步，编写一个简单的控制程序，将IOB表的地址值传送给Y，A寄存器。（控制程序假定从\$6015起存放）

地址	内容	说 明
\$6015	A9 60 LDA \$#60IOB表高地址送A累加器	
\$6017	A0 00 LDY# \$00IOB表低地址送Y寄存器	

\$6019 20 D9 03 JSR \$0309

转RWTS程序

\$601c 60 RTS

然后在监控系统中可用6015G命令运行上述的控制程序。(BASIC中可用 CALL 24597调用), 读取完毕后, 用户可以从内存 \$7000起查看17道0区的VTOC表内容。

四、内存扩充与DOS连接

从上述RWTS使用的介绍来看, 如果用户知道自己的程序或其它信息在磁盘上存放的确切物理地址, 那么用户完全可以不要DOS系统中的其它控制程序, 而只借用RWTS来与磁盘沟通信息交流。

如前所述, DOS在RAM 48k的内存中, 是存放在 \$9600~\$BFFF这一段连续内存空间的, 假如用户程序的运行中, 没有频繁地调用磁盘的现象, 那么完全可以将内存空间的最高地址 \$9600修改为 \$B600, 这样, 剩下约2.5k的RWTS及必要的IOB表和DCT表, 从而可给用户增加8k左右的宝贵内存空间。(图2)。

附录中的程序3第40句, 将内存最高地址设定为 \$B5FF, 60句在新开辟的内存中设定一个10×300大小的二维矩阵(约16k大小), 这个数据矩阵有一部份就利用了原来DOS系统所占据的空间。

实际上, 一般用户很少用RWTS去存取每一个自己所编写的程序或数据, 因为这样实在太烦, 而且也容易出错。要完成与磁盘的信息沟通, 当然还是利用DOS要来得容易、方便。要利用DOS完成上述任务, 先将DOS从磁盘中调入原来的内存地址, 被冲掉的这部分DOS程序在磁盘上是从00磁道0c扇区一直到02磁道04扇区连续存放的。我们用RWTS将这一段DOS从盘上调入内存相应地址。然后用DOS再完成其它必要的磁盘访问任务。

从磁盘上调入DOS的程序见附录程序1, 在BASIC中调用恢复DOS的地方见附

录程序3中的第170句及第180句。附录程序2是占据DOS内存时应做的工作。

五、程序的说明

1. 程序1的说明

(1) 子程序 \$B711的功能:

主要作用是将DOS资料(除RWTS)调进内存, 放入到 \$9D00~~\$B5FF。

不过, 真正调DOS的子程序是 \$B793。此程序以反方向来读DOS资料, 由于DOS以连续格式储存在磁片上, 所以调进DOS资料时不需要重新挪动地址, 而使调入时的速度十分快。

(2) 为何在RWTS工作结束后把 \$48单元规定为0:

由于RWTS利用了 \$48与 \$49两零页地址作为存放IOB的地址, 然而 \$48同时也被APPLE的监控程式作为状态寄存器P暂存之用。所以, 当我们结束了RWTS工作后, APPLE会将地址 \$48的内容存入P暂存器中。如果我们不将它设定为0, 则一些麻烦的事情由此而产生, 使APPLE工作失常(譬如喇叭发出声音)。

(3) 子程序 \$A7D4的功能:

主要做了以下三点工作。

1) 恢复了3个缓冲区(BUFFER)中的地址指针。

2) 对各缓冲区中的档名暂存区第一位元组送0, 使DOS获得了对它们的使用权。

3) 重新设定HIMEM的值, 为 \$9600。

因为 \$9600~~\$9cF8并没作为DOS资料存入了磁片上, 但是, 我们用DOS命令时, DOS还是用到了这3个缓冲区, 而这些, 正需要一些指针来帮助它来完成命令工作, 所以要对3个缓冲区的某些地址添上数值作为指针, 子程序 \$A7D4恰好做了这步工作。

(4) 子程序 \$9E30的功能:

主要作用是把 \$BF, \$9D和\$9D及\$A5经过异或后的值送给 \$3F2~~\$3F4。

\$9DBF是DOS的热启动地址。

(5) 子程序 \$A851的功能:

主要作用是使计算机的(I/O)地址设定为DOS的(I/O)地址。

它把DOS的输入子程序入口地址 \$9E81送入\$38, \$39。把DOS的输出子程序入口地址\$9EBD送入\$36, \$37。

2. 程序2的说明

(1) 子程序\$9EE0的功能:

主要作用是恢复真正的(I/O)地址入口。

因为子程序\$9EE0具有这个功能, 所以它使我们的程序十分灵活。无论是STC

或其它汉字系统, 只要一执行子程序 \$9EE0, 它就会把(I/O)地址入口定为你所使用系统的输入/输出地址。如果你没有使用任何汉字系统, 那也没关系, 它会把输入, 输出地址定为原来BASIC的输入/输出指针即\$FDF0和\$FD1B。

参考文献

1. 施威铭 Apple I 彻底研究(之三)。
2. 郑昌 Apple I 微机及汉字系统, 同济大学出版社, 1985年。
3. 荣树勋 张开启 6502微处理器及微应用, 北京师范大学出版社, 1984年。

附录

程序1

1	ORG	\$BC57	
2	JUANHAO	EQU	\$B7EB ; 磁片卷号
3	WEIZIL	EQU	\$B7F0 ; 资料缓冲区低地址指针
4	HUIFUDOS	LDA	#\$00
5		STA	JUANHAO ; 磁片卷号为0, 即与任何卷号通用
6		STA	WEIZIL
7		LDX	#\$60
8		STX	\$B73B
9		LDA	#\$19 ; 需调进25个扇区的DOS资料
10		JSR	\$B711 ; 转调DOS资料 * (A, 1)
11		LDA	#\$A2
12		STA	\$B73B
13		LDA	#\$00
14		STA	\$48 * (A, 2)
15		LDA	\$69
16		STA	\$6B
17		STA	\$6D
18		LDA	\$6A
19		STA	\$6C
20		STA	\$6E
21		JSR	\$A7D4 ; 恢复缓冲区的指针及重新设定HIMEM值 * (A, 3)
22		LDA	#\$60
23		STA	\$9E41

```

24      JSR      $9E30      ; 恢复DOS热启动向量地址      *(A, 4)
25      LDA      #$A9
26      STA      $9E41
27      JSR      $A851      ; 恢复DOS的(I/O)地址      *(A, 5)
28      RTS

```

程序2

```

1      ORG      $BC92
2 CHONGDOS JSR      $9EE0      ; 设定真正的(I/O)地址      *(B, 1)
3      LDX      #$02
4 CS     LDA      SHUJU, X
5      STA      $03F2, X      ; 设定$3F2~$3F4的值与BASIC沟通的热启动向量地址
6      DEX
7      BPL      CS
8      RTS
9 SHUJU  DFB      $03, $E0, $45

```

程序3

```

10     PRINT     CHR$(4); "BLOAD  HUIFUDOS/2.OBJ0"
20     PRINT     CHR$(4); "BLOAD  CHONGDIAODOS/2.OBJ0"
30     CALL      48274, RAM 调冲 DOS程序
40     HIMEM, 46591; RAM 设定 HIMEM为$B5FF
50     LOMEM, 30000
60     DIM       A(10, 300)
70     FOR       I=1 TO 10
80     FOR       J=1 TO 300
90     A(I, J)=I*J
100    NEXT, NEXT
110    FOR I=1 TO 10
120    PRINT     A(I, 300),
130    NEXT, PRINT
140    PRINT     CHR$(4); "CATALOG" ; RAM 未执行 CATALOG命令
160    CALL      -198
170    CALL      48215, RAM 调恢复DOS程序
180    PRINT     CHR$(4); "CATALOG" ; RAM 执行 CATALOG命令

```

信息系统开发的几种有效方法

东北工学院

周 勇

摘要 本文列举了几种实用的信息系统开发技术,并对其优缺点作了简单的评述,旨在为信息系统的开发人员和应用部门提供一些有益的启发。

一、引言

采用实用而有效的信息系统开发技术,对于缩短软件研制周期,提高工作效率和生产效益,具有事半功倍的效果。在此,将几种具有代表性的信息系统开发方法列举如下:

- 小规模实验方法
- 自然开发方法
- 金字塔方法(或分阶段实现方法)
- 原型方法
- 结构化方法
- 自顶向下方法
- 集成方法

在信息系统的开发中,大多数信息系统研制人员综合使用这些方法。例如,一个业务系统就可能结合使用自顶向下方法、结构化方法和集成方法进行系统开发。值得注意的是,用户在整个开发过程中起着不可忽视的作用。

二、对系统开发方法的评述

1. 小规模实验方法

小规模实验方法结合采用一种或多种信息系统技术如决策支持系统和专家系统来开发系统,系统的开发和应用只限于一个单位的某个部门或某个专业领域如会计帐目管理、干部人事档案管理、医疗辅助诊断等。小规模实验法要求软件人员首先与用户反复会话,在熟悉了应用部门的业务并确立了新系统的目标后,才着手进行系统的开发与实施。

对在局部范围采用小规模实验法实现系

统的评价,基于用户对这个开发项目的态度。当用户对此抱积极态度时,则对系统的成功起着尤为重要的作用。安装新系统的效益必须与开发和运行该系统的成本相比较。另外还要考虑到该系统的实现对其它单位或部门工作程序的影响。一旦小规模实验完成,重要的是评估这种技术在该单位的其它领域实现的方便程度。

使用小规模实验法的一些优点是便于管理,有灵活性,风险小,开发期间需要较少的人力资源,不需要掌握综合信息系统开发的技术和知识,对其它系统影响较小,在新系统开发完成之前就能得知最终的效益。

小规模实验法的缺点在于系统开发和应用的局限性,即不能充分发挥网络、共享数据库等现代信息系统技术成果的优势,在整个系统的开发和使用中也没有考虑其它部门和专业领域的应用和开发,因此难以得到必要的配合和协调,从而影响了系统的使用寿命。

2. 自然开发方法

自然开发方法允许和鼓励信息系统开发“自然地”进行。即当用户提出什么要求就去完成什么要求。这种方法在系统开发中提供了高度的灵活性,它允许用户进入信息系统的任何层次,即用户不仅在需求分析阶段起重要作用,而且在系统的设计和实施过程中不断提出自己的愿望和意见。虽然有时候用户的意见对于开发一个效率高、生命力强的系统很不利(如希望新系统不改变地模仿原始的工作方式,或者只根据当时的财力而不是系统需求来决定购买的设备

等), 但设计人员还是完全按照用户的意见和想法去实现系统, 这样做就最大限度地将用户包含在系统开发中, 即由用户对整个系统的成功负责。

尽管这种方法表面上没有得到多少专家的赞同, 但已被不少单位所采用。相当于增加了系统的限制, 对于系统分析和设计人员的要求相对低些。在我国目前缺乏计算机高级软件人员的情况下, 这也是一种行之有效的方法。

自然开发方法的缺点是难以进行有计划的系统开发和单位间的合作。一个单位的大多数主要的信息系统需求可能并不是新系统所满足的需求, 因为后者往往依赖于当时所购得的软硬件配置。

3. 金字塔方法

金字塔方法(或分析阶段实施方法)是一种有计划、分阶段地开发系统的方法。即在深度和广度上使系统各种功能逐步得以实现。该方法从局部或者某个业务范围的基本需求开始, 不断通过系统开发的各个层次, 直到建立起一个整体信息系统网络。图1表示了一个采用这种方法实现的系统。该系统从最低级的字处理、数据处理和传真等开始通过数据库管理、电子邮件等逐步向前发展直至获得一个综合的管理信息网络系统。

金字塔方法特别适宜于那些很希望实现综合管理信息系统而又一时解决不了所有资金、人力、技术以及管理上的问题的单位或部门。

这种方法的主要优点是提供了完成信息系统的某一层到该系统上一层工作的有计划的发展进程。它造成一种使用户逐步适应和了解新系统的环境。当用户从一个系统转移到其它系统时, 这种方法为他们提供了从一个系统中学习的条件。同时这种方法还提供了一种理解、研究和开发自动化管理系统的策略。也就是说, 由于系统是分阶段完成的, 所以在开发过程中就不必将所有用户同

时卷入或者向应用部门提出过多的人力资源要求。因而这是一种多数应用部门能够接受的开发方法。

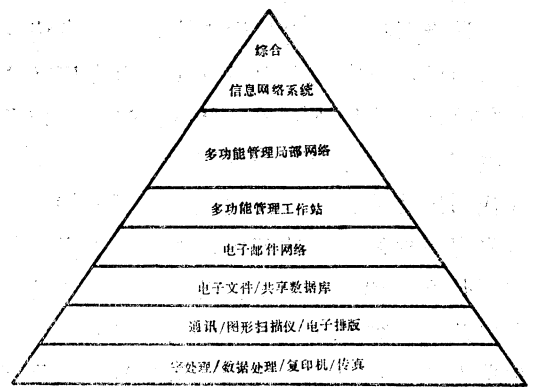


图1. 一个采用金字塔方法实现的系统

4. 原型方法

原型方法强调被开发系统的原型的作用。原型法要求在短时间内建立一个交互式应用系统来满足用户的基本需求, 并且强调开发速度而不是运行效率。原型是系统的基本部分, 原型的建立以复盖用户的最基本的业务为目的, 因而是一个简化了的系统。

可以借助于数据库管理软件、通用检索、统计和报表软件、特殊软件包或者第四代语言开发工具来实现原型系统。当前, 面向用户的非过程式软件不断问世, 为应用原型方法提供了有利的开发环境, 因此这种方法在系统开发中用得越来越多。

原型系统建立后, 需要软件人员和用户密切配合来进行系统的改进和扩充工作, 以实现最终的系统。应用原型方法来开发一个成功的系统必须以如下条件为前提;

- 有一个严格定义的调查范围
- 有限的用户目标
- 能建立起一个数据字典
- 有愿意与原型法研制者合作来回答应用方面问题的用户
- 有愿意接受原型系统作为满足他们要求的临时和适宜的结果的用户

原型法的优点在于, 它以一种快速的方

法来进行系统的开发和应用实践。通过提供一个活生生的工作模型于研制最终的系统设计之前,来作为中间学习和训练用户的媒介。在系统开发中,这种方法也提供了灵活性。

尽管原型法可能替代系统开发中传统的生命周期,可在现实中应该成为生命周期的一部分。它是增强而不是替代、增强的因素产生于对用户需求的分析和这种设计方法、在使用传统的方法开发系统时,首要的工作是决定用户需求。然而用户常常不知道他们需要什么,因为他们不知道新系统能干些什么以及能干到什么程度,所以他们对系统的创造力不能得以充分发挥。随着他们获得基于他们要求的新系统的经验时,他们关于这个系统的知识增长了,创造力产生了,从而提出更多的需求。因此,系统修改和扩充的内容是会很多的。

实质上,使用原型方法开发系统的主要目的是在软件开发周期的早期开发用户的知识库。

原型方法和小规模实验方法看起来非常相似。其基本区别在于,小规模实验方法假定系统根据用户的要求已被设计成为“最终的”形式,这种最终形式与用户需求紧密联系,结果几乎不需要什么改变。正如前面所讨论的,对使用小规模实验法的主要评价只包含学习和接受系统所提供的內容。

与之相反,原型方法的前提在于最终的设计,是根据在该原型上工作的用户的意见进行修改后的结果。因此,最终的系统可能相当不同于原型。甚至在系统已经安装并且可运行后,其设计可能还不是最终的。用户几乎立即开始提出修改。特别是随着他们对系统的熟悉更是如此。采用原型法时,是用户构造了这个系统,因此在开发系统中要重视这些修改。此外,这种方法对于开发决策支持系统时更显其特长。

原型方法有一个缺点,它需要初始投资来安装软件和硬件即原型系统。随着系统的

不断修改和加强,对原型中的软、硬件将会进行变动,有时甚至会完全抛弃这些初始投资的设备,从而造成经济上的浪费。

5. 结构化方法

结构化方法遵循规范的、严格的系统开发法则,其目的是产生出一个有效而灵活且易于维护的系统。结构化技术在近十多年来对信息系统的设计者有着特别重要的影响。结构化系统开发的生命周期如图2所示。

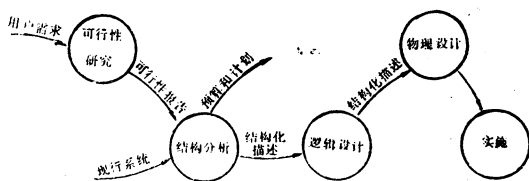


图2 结构化系统开发的生命周期

许多人在今天将结构化方法看成仅适用于大型信息系统工程,仅仅强调其在系统设计和程序设计中的使用。而当前结构化方法在系统分析中的应用正日益走向成熟,并且适用于不同类型的系统开发。

结构化设计有一种精确的组织结构,它将系统的不同部分分解成子系统模块,每一个子系统可以被独立的研制,并且能够进行修改、删除和增加子系统操作而不会影响其它子系统。

结构化方法包括三个方面的工作。首先,将系统划分或分段成易于管理的子系统模块;其次,尽量完全和准确地定义模块或子系统之间的接口,然后对每个模块或子系统的处理或操作进行详细说明。

结构化方法的主要优点是提供一种规范的、有组织的方法来开发信息系统。此外,使用结构化技术的系统总是导致较少的开发费用,使用户更为满意,并且系统更容易维护。系统某一部分的改变并不会影响系统其它部分。

今天有许多自动化软件包来帮助我们进行结构化系统分析、系统设计和文本编辑工作。这种计算机辅助软件工程势必导致软件

工业的革命。

结构化方法的两个主要缺点是：

- 在信息系统开发活动中需要高级专业技术人员
- 需要应用单位付出较多的人力物力的资源

在现实中，使用结构化方法开发系统既有许多优点又并不是处处适宜的。一种有效的技术是将结构化方法与其它方法结合使用，如小规模实验方法或原型方法。另外，结构化方法常常是与下面准备讨论的自顶向下方法结合使用的。

6. 自顶向下方法

使用自顶向下方法的系统开发人员将一个单位看成一个系统。他们调查这个单位的任务并分析其目的，将各种处理和活动分成子处理或各种功能，同时又将每个功能分成子功能并一直分下去直到便于管理为止。通过对单位的调查，设计人员研制出系统模型或决定系统中各功能间关系的模型。一个在系统开发生命周期中使用自顶向下方法的例子如图3所示。

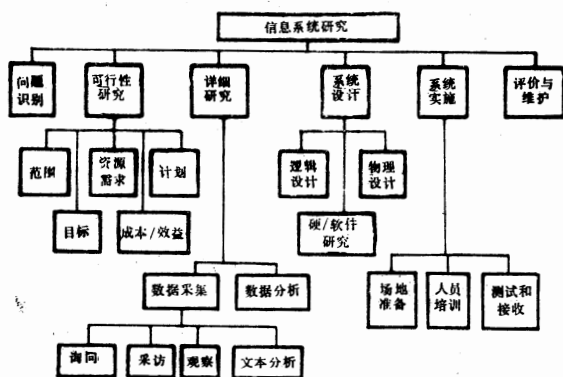


图3. 一个使用自顶向下方法的例子

7. 集成方法

与自顶向下方法紧密联系的是集成方法。集成法是一种提供系统兼容性处理的方法，即使一个系统中的软件、硬件和数据在其它系统中兼容。由于计算机硬件的更新周

期越来越短，为满足新的需求要购置新的设备，而现有的软硬件又一时不能抛弃，所以就提出了系统的兼容性要求，独立开发的系统有时也要求与别的系统有一定兼容能力，由此产生了集成开发方法。

尽管集成可以通过自底向上的开发方法（如在购买了新的系统设备后，再考虑集成）实现，但采用自顶向下方法（如按照集成的要求来配置软件和硬件设备）实现集成会更好。若采用自底向上方法实现集成，则系统设计人员就被迫在现有系统上进行兼容处理。强制的集成方法常常需要对现存系统重新加工，如重新定义现行数据库或输入输出元素。如果通过自顶向下的研究方法获得集成，就可以随着系统开发的过程，消除系统的再加工。而自底向上方法几乎总是需要这种再加工。

三、结 语

上述几种信息系统开发方法远远优于常用的“试错纠错”处理方法。而有选择地将其中几种方法结合起来使用更会得益不浅。信息系统开发人员选择这些方法时，除了权衡各种方法的利弊外，还应依据开发项目的大小，软件人员的水平，硬设备和软件的配置情况，同时还要兼顾到本单位、本系统的长远规划。最终目的是应能保证所完成的信息系统工程的质量和效益。

参考文献

- [1] Baldwin, Darrell F., "Following the Natural Path to Real Office Automation", The Office (May 1984), P.166.
- [2] Yourdon, Edward, "What Ever Happened to structured Analysis?", Datamation (June 1986), pp133-138.
- [3] 唐世渭、方裕译，《软件工程—实践者的研究途径和方法》，《小型微型计算机系统》编辑部，1984年。
- [4] 曹锦芳，《信息系统分析与设计》，北京航空学院出版社，1987年。

微机上矩阵求秩的一种快速算法

湖北省计委计算中心 朱容村

摘要 由于国内外微机应用软件中没有现成的矩阵求秩功能,而且一般求秩方法又比较烦难,因此本人通过长期摸索,总结出一种所谓“孤数化零”的矩阵快速求秩法,这种方法可大大简化求秩过程,并确保矩阵的秩不变。本文介绍了“孤数化零”矩阵求秩法的基本概念、程序设计、应用实例以及其主要特点。

一、引言

矩阵求秩在编制处理数据,特别是在编制数组数据的程序时经常碰到。国内普遍使用的微机应用软件目前尚无现成的矩阵功能,更谈不上有矩阵求秩的库函数,使用时程序员必须自己临时编制。由于编制的方法不同,实现求秩的步骤也不一样,达到这一目的所花费的机时也不相同。

我们知道,矩阵求秩的问题往往使人觉得原理简单,而实际求秩则较烦难,有时甚至感到无从下手,特别是当矩阵的行、列数较多时,求秩的过程更为复杂。另外,对矩阵施行三类初等变换时,矩阵的秩不会改变,也就是说,一般矩阵的秩,可以通过初等变换的方法求得,但是也有些难解(求秩)的矩阵,它无法通过初等变换直接决定其秩,而非要借助于计算矩阵中较低阶行列式的数值是否为零来作辅助判断不可。而且解行列式又是一个极为麻烦的事情。因此,这给程序人员在计算机上编制矩阵求秩程序带来了很大的困难。为了解决这个问题,本人通过较长时间的摸索,总结出一种所谓“孤数化零”的矩阵求秩法,它可简化求秩过程,并确保矩阵的秩不变。下面介绍“孤数化零”求秩法的程序设计与实现。

二、引入几个基本概念

1. 孤数

设矩阵的某行(或某列)中只有某一元素不为零,而其它各元素均为零,则称该唯一不为零的数为孤数。

2. 孤数阵

矩阵中任一行或任一列中至多只有一个非零元素的矩阵就叫做孤数阵。

3. 孤数阵的性质

(1) 孤数阵是任何矩阵经过求秩变换后必能达到的终极形式。

(2) 矩阵经过求秩变换所得的孤数阵中的孤数数就是原矩阵的秩。而不论孤数在孤数阵中散布形式(即位置)如何。换句话说,孤数阵中的非零元素有几,原矩阵的秩就是几。

关于孤数阵的定义和基本性质,已作过论证,为从简,在这里就不赘述了。

三、“孤数化零”矩阵求秩法的程序设计

在进行程序设计时,只要依据以上叙述的孤数阵的定义和性质,对于任意维的矩阵若要求其秩,只要将其化为孤数阵的形式就行了,不必再判断该矩阵中的较低阶行列式数值是否为零。下面是“孤数化零”矩阵求秩的程序框图1。

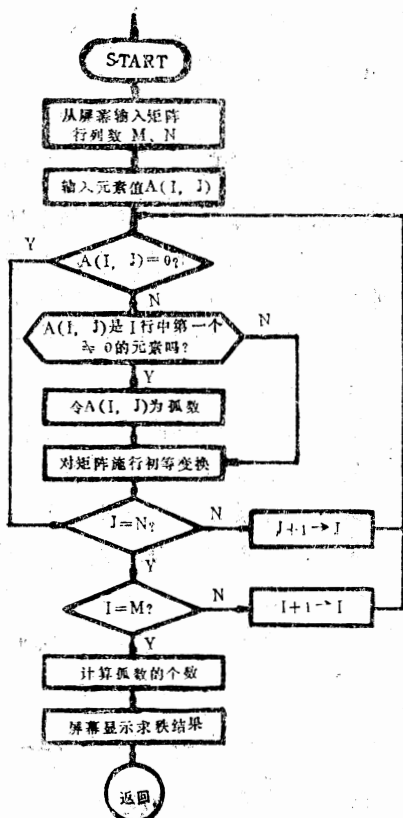


图1.求秩程序框图

根据程序框图编制矩阵求秩程序，程序如下所示。

```

10 INPUT "M="; M: INPUT "N="; N
20 DIM A(M, N)
30 FOR I=1 TO M: FOR J=1 TO N
40 INPUT "ELEMENT="; A(I, J)
50 NEXT J: NEXT I
60 FOR I=1 TO M: H=0: FOR J=1 TO N
70 IF A(I, J)=0 THEN 100
80 IF H=0 THEN K=J: H=1: GOTO 100
90 GOSUB 200
100 NEXT J
110 IF I<M AND H<>0 THEN FOR L=I+1 TO M: A(L, K)=0: NEXT L
120 NEXT I
130 FOR I=1 TO M: FOR J=1 TO N
140 IF A(I, J)<>0 THEN R=R+1: A(I, J)

```

```

=A(I, J)/A(I, J)
150 NEXT J: NEXT I
160 PRINT "A_r="; R
170 FOR I=1 TO M: FOR J=1 TO N: PRINT A(I, J);
180 NEXT J: PRINT " ", NEXT I
190 END
200 D1=A(I, K), DZ=A(I, J)
210 FOR L=I TO M: A(L, J)
  *D1-A(L, K)*DZ: NEXT L
220 RETURN

```

程序中60~150和200~220语句是矩阵求秩过程；10~50及160、170语句为控制输出部分。若将此程序作为子程序调用，只需将其中相应的输入输出语句稍作改变即可。

四、求秩的应用实例

为了进一步证明“孤数化零”矩阵求秩法的正确性，我选用了北京大学，同济大学和上海交大三校“线性代数”教材中求秩的例题，用本求秩法加以验证，结果完全正确。如果有兴趣，也可选用其他例题来验证。下面是从三校教材中摘选的几个矩阵 A_1, A_2, A_3, A_4 ， R_1, R_2, R_3, R_4 表示它们对应的秩。

应用前面程序（清单），将下列矩阵分别输入计算机，立刻可以得到其矩阵的秩。

例：

$$A_1 = \begin{pmatrix} 1 & -1 & 2 & 2 & 0 \\ 2 & -2 & 4 & -2 & 0 \\ 3 & 0 & 6 & -1 & 1 \\ 0 & 3 & 0 & 0 & 1 \end{pmatrix} R_1 = 3$$

$$A_2 = \begin{pmatrix} 14 & 12 & 6 & 8 & 2 \\ 6 & 104 & 21 & 9 & 17 \\ 7 & 6 & 3 & 4 & 1 \\ 35 & 30 & 5 & 20 & 5 \end{pmatrix} R_2 = 2$$

对于稀疏型数据表的压缩方法

苏州市计算机开发应用研究所

周 望

一、问题的提出

在研制开发微电脑产品的过程中,经常会遇到在内存中固化大量的数据表。而对于那些专用电脑来说,内存容量受到一定的限制,如果数据的存放方法不好,势必占用很大的内存空间,也增加了硬件的成本。因此此研究数据的结构,找出一种最佳的存放方式,是很有现实意义的。

以我们近来开发的SYP-1膳食营养配方计算机为例,这是一台专用微机。按照配方的功能要求,在内存中固化了250种食物的

57种营养元素的每百克含量。如果按照每个元素的含量分配二个字节的存放单元,那么每一种食物要占114个字节。整台机器就得安排 $250 \times 114 = 28500$ 个字节来存放全部数据。这是原来设计的存放方案,它是一个定步长的数据表。经过我们对数据表进行详细分析后(参见本文所附的食物百克含量表)发现,以57种元素组成的数据表是一个稀疏型的数据表,表中有百分之六十以上的元素含量为零。但是,为了格式统一,在安排数据表时,这些项仍占二个字节,因此,使有限的内存空间得不到充分的利用。

$$A_3 = \begin{pmatrix} 1 & 0 & 0 & 1 & 4 \\ 0 & 1 & 0 & 2 & 5 \\ 0 & 0 & 1 & 3 & 6 \\ 1 & 2 & 3 & 14 & 32 \\ 4 & 5 & 6 & 32 & 77 \end{pmatrix} R_3 = 3$$

$$A_4 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix} R_4 = 5$$

五、主要特点

矩阵求秩的功能在微机应用软件中是没有的,而“孤数化零”矩阵求秩法不仅弥补了这方面的不足,而且比一般求秩方法要简单得多。具体地说,它有如下一些特点:

1. 适用性强

“孤数化零”求秩法,不仅能在微机上

使用,也可在其它大中型机上使用,而且可用不同的语言来编写。

此种方法适用各种不同的矩阵。

在程序设计上,它是一个独立的公用模块,可作子程序而随时调用,并有较好的数据接口。

2. 运算速度快

由于该求秩方法先进,步骤简单,不需要一般求秩方法中的各种复杂的循环运算和行列式运算,而且整个求秩过程,只在本矩阵中进行,不需要开辟存放中间结果的工作矩阵,所以求秩速度极快。

3. 结果准确

在程序设计时,考虑到矩阵施行初等变换所遇到的各种可能,本程序将除法运算全部变为乘法运算,避免了特殊情况下,所产生的非法运算和数值溢出问题,不仅保证了数值的精度,而且使结果准确无误。参考文献从略。

对于这样一种稀疏型、并且元素含量值为零的规律无法遵循的数据表，我们设计了映射技术，将值为零而仍占据2个字节的情况排除掉，有效地压缩了这种稀疏型数据表。

二、设计思想

首先，为每种食物的元素百克含量表设计一张能反映含量是零值还是非零值的特征映射表。表的大小为10个字节，前8个字节共反映64个特征位，实际用57个，后二个字节安排该食物含量的数据表在整个数据存放空间的相对位移量。

规定特征映射表中的特征位为：元素含量为零，特征值为“0”；反之，元素含量不为零，则特征位为“1”。

例如原来每百克香菇的57种元素含量，在内存中的存放格式为：（同时参见食物百克含量表）。

1000— 59 30 14 40 01 90 59 30

1008— 00 00 00 00 00 00 14 40

1010— 00 00 00 00 00 00 00 00

1018— 00 00 08 50 10 50 01 24

1020— 04 15 25 30 19 60 00 00

1028— 00 00 00 00 00 00 00 00

1030— 00 07 01 13 00 30 00 00

1038— 00 53 00 00 18 90 02 10

1040— 00 00 00 00 00 00 00 00

1048— 00 00 00 00 00 00 00 00

1050— 00 00 00 00 00 00 00 00

1058— 00 00 00 00 00 00 00 00

1060— 00 00 00 00 00 00 00 00

1068— 00 00 00 00 00 00 00 00

1070— 00 00 FF FF FF FF FF FF

1078— FF FF FF FF FF FF FF FF

现在，采用前面所述的映射方法，把原来的数据表映射成特征映射表，其格式及分解如下：

地址	十六进制值	二进制值
0800	F1	1 1 1 1 0 0 0 1
0801	07	0 0 0 0 0 1 1 1
0802	E0	1 1 1 0 0 0 0 0
0803	EB	1 1 1 0 1 0 1 1
0804	00	0 0 0 0 0 0 0 0
0805	00	0 0 0 0 0 0 0 0
0806	00	0 0 0 0 0 0 0 0
0807	00	0 0 0 0 0 0 0 0

与此同时，将原来的数据表中的值为零的元素剔除，演变成下列格式：

2000— 59 30 14 40 01 90 59 30

2008— 14 40 08 50 10 50 01 24

2010— 04 15 25 30 19 60 00 07

2018— 01 13 00 30 00 53 18 90

2020— 02 10

经过压缩之后的数据表可通过映射表的最后二个字节查出其在内存中的相对位移量。在查找数据时，只需将整个数据表的首址，加上该食物映射表提供的相对位移量便可得到其数据。从而，解决了压缩后数据块长度不一的动态查找问题。

经过上述方法，一个稀疏型的数据表得到了有效的压缩。另一方面，原先定步长的数据表变成了一个变步长的数据表，并且增加了一个定步长的食物元素含量的映射表。按照实际使用，还要恢复成原来的格式，即恢复含量为零的元素的对应位置。

恢复的方法是：按照给定的食物号，查找到对应的映射表，从中取出相对位移量，然后加上表的首址，这就得到该食物的数据表首址。另一方面，开辟一个缓冲区。接着对映射表中的特征位逐一进行位测试。当被测特征位为1时，就从数据表中取出当前指针所指的2个字节的数据，将其填入缓冲区当前指针位置，然后修改缓冲区和数据区指针。反之，被测位为0时，缓冲区当前指针位置填入二字节零，同样修正指针，但是需修改缓冲区指针。按照元素的个数进行测试，最后在缓冲区内出现一个与原来定步长数据表一样的稀疏数据表。其流程图见图1。

三、压缩效率

按照原来的有效方案，每种食物共占用114个字节，现在采用映射表的压缩方法，虽然新增加了10字节的映射特征位表，但是，新数据表仅是原先的数据表长的40%，即：
 平均占用 $114 \times \frac{40}{100} = 45$ 个字节，再加上映射

表的10字节，共55个字节，节约一半以上。
 从广义上来说，只要数据表中的零值元素的个数是数据表的20%以上，采用上述压缩方法均能收到效果。对元素值占据2个字节以上的稀疏数据，尤其合适。

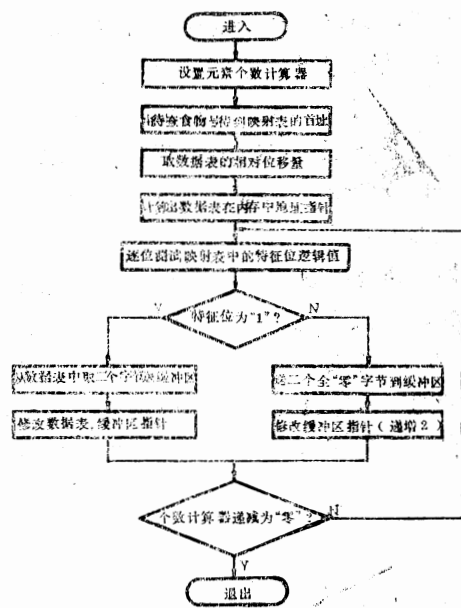


图1

表1. 食物成份表

每百克香菇含：																			
碳水化合物		59.30	g	蛋白质		14.40	g	脂肪		1.90	g								
多糖		59.30g	单糖		0	g													
动物蛋白		0	g	豆类蛋白		0	g	其它蛋白		14.40g									
动物脂肪		0	g	饱和脂肪		0	g	不饱和脂肪		0	g	单不饱和脂肪		0	g				
胆酸醇		0	mg	粗纤维		8.50	g	水份		10.50	g								
钙	124	mg	磷	415	mg	铁	25.30	mg	钾	1960	mg	钠	0	mg					
氯	0	mg	镁	0	mg														
胡萝卜素		0	mg	vitA		0	I.U	B1	0.07	mg	B2	1.13	mg						
叶酸		30	ug	vitC		0	mg	B6	0.53	mg	B12	0	ug	pp	18.90	mg			
泛酸		2.10	mg	vitD		0	I.U	vitk		0	ug	vitE		0	I.U				
缬氨酸		0	mg	亮氨酸		0	mg	异亮氨酸		0	mg	苏氨酸		0	mg	苯丙氨酸	0	mg	
色氨酸		0	mg	蛋氨酸		0	mg	赖氨酸		0	mg	精氨酸		0	mg	组氨酸		0	mg
酪氨酸		0	mg	胱氨酸		0	mg												
锌	0	mg	铜	0	mg	锰	0	mg	硅	0	mg	铝	0	mg					
硼	0	mg	碘	0	ug	钴	0	ug	硒	0	ug	碘	0	ug					

在应用软件中执行DOS命令的方法

大连理工大学力学所 孟宪福

摘要 本文给出了在一般应用软件中执行MS-DOS或PC-DOS命令的方法及其程序实现,根据这种方法,程序设计者就可以在应用程序中直接使用DOS系统的所有命令,这为开发某些大型应用程序提供了方便。

一、应用背景

众所周知,COMMAND.COM是MS-DOS系统的组成部分之一,它的主要任务是解释、执行DOS系统的所有命令,其中包括内部命令和外部命令。DOS命令一般是在系统提示符状态下执行的,但在开发某些大型应用程序时,有时我们希望在退出应用程序的前提下也能执行DOS命令,如显示当前目录,拷贝文件等,这样,就希望在应用程序中也能执行DOS命令,否则就需要程序设计者自己来编写完成这些功能的程序,其工作量是相当大的。文献[1]中提供的方法,一般来讲只能执行.EXE型文件或.COM型文件,而对.BAT型文件(批处理命令文件)及DOS内部命令则无能为力。本文将给出在应用程序中执行所有DOS命令的方法,其中包括执行.EXE型文件,.COM型文件,.BAT型文件及DOS内部命令。文中所提到的应用软件,一般是指用汇编语言写成的,而对于用高级语言写成的应用软件来讲,由于一般的高级语言文本都提供执行DOS命令的函数,如C语言中提供的System函数,所以此文中不再赘述。

二、DOS命令的执行步骤

在应用程序中执行DOS命令的过程,实际上是启动DOS子进程的过程,通过再次装入系统的COMMAND.COM文件,来解释、执行DOS命令。目前我们所使用的DOS版本,都是单任务、单用户操作系统,但它

允许象主程序调用子程序那样,在程序执行过程中调用另一程序(一个完整的可执行程序或DOS命令)。DOS系统中允许的这种调用关系(可称为父子进程关系),同多任务、多用户操作系统中的进程概念是完全不一样的。在DOS系统中,只有当子进程执行完后,父进程才能继续执行,父子进程不能并发地同时执行,如图1所示。

1. 内存区域的释放

在DOS系统中,程序装入内存时,如果被装入程序的类型是EXE型,则系统将根据EXE型文件头部信息表中的MAXALLOC的值来为其分配内存空间。在一般情况下,利用DOS的Link程序进行连接操作时,如果不附加/HIGH开关信息的话,则MAXALLOC的值将被设置为FFFFH,这样,就把DOS所能管理的最大空间分配给了被装入的EXE型程序,如果被装入的程序类型为COM型的话,则DOS系统将把剩下的所有内存空间分配给被装入的程序。已经被分配的内存,将成为相应进程的专用空间,其它进程不能使用。这样,为了在父进程中能够启动子进程,就必须先利用系统调用4AH来释放未使用的内存空间,以便为启动子进程制造条件。

4AH号功能调用的入口参数设置如下:

- (1) 将AH寄存器设置为4AH。
- (2) 将父进程执行时的PSP所在段址存入ES寄存器中。
- (3) 把要求获得的内存区域大小放入BX寄存器中。

BX寄存器中的参数很重要,如果设置

得不正确，则将影响内存的释放以及子进程的启动。在实际应用中，BX 中的参数值可用下述方法求出：

①在父进程正文的结尾处设置一个空段，把其段名设置为ZZZZZZ。

如：

```
ZZZZZZ SEGMENT
ZZZZZZ ENDS
```

②把父进程执行时的PSP所在段址存入PSPSEG单元中。

③利用下列算式求出BX 中的参数值：

$BX = ZZZZZZ - PSPSEG$

2. 启动子进程

在释放了相应的内存空间后，就可利用4BH号功能调用来启动一个子进程，以便完成DOS命令的执行。

4BH号功能调用有两种工作方式：当其入口参数AL = 00时，读入程序并执行；当AL = 03时，只读入程序而不执行，这时读入的程序由父进程进行控制和操作。

使用4BH号功能调用的关键，是要正确设置其入口参数。

下面，我们以AL = 0的情形为主，根据我们的编程经验，来设置4BH号功能调用的入口参数。

(1) 将AX寄存器设置为4B00H。

(2) 将DS:DX 设置为指向欲执行的文件名。为了在应用程序中能够执行DOS的所有命令，就需要将DS:DX指向 COMMAND.COM，以后通过COMMAND.COM来解释、执行相应的DOS命令。

(3) 把参数块指针放入ES:BX寄存器中。

参数块中的具体内容，请参阅相应的DOS手册或文献[1]。下面，我们以 dir/w 命令为例，来设置相应的参数块，如图2所示，其中环境段地址设置为0，表明子进程继承父进程的环境。

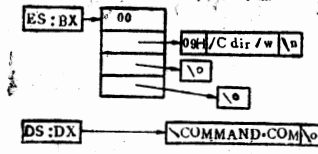


图2 参数块设置

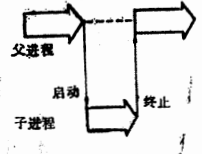


图1 进程的执行顺序

三、程序实现

我们在实现大型编辑程序CWS 时，引进了执行DOS 命令这一功能，下面，我们以简单框图的形式，给出在应用程序中执行DOS命令的过程，如图3所示。

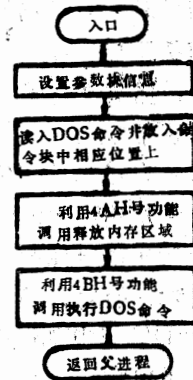


图3 程序框图

参考文献

- [1] 张汉亭, MS-DOS子过程的启动, 微机应用, 3, 1988.
- [2] MS-DOS Programming technique, アスキー出版局, 1986.

微型机信号处理系统中软接口的研究

华侨大学 陈宝珊

摘要 本文以IBM—PC微型机为基础,分析了研制信号处理系统中混合语言编程出现的若干理论与技术问题。文章以建立磁盘数据文件和连接目标模块两个角度实现了高级语言之间以及高级语言与汇编语言之间的软接口,解决了数据采集和实时控制中混合编程的关键问题。

一、问题的提出

利用微型机,配以少量的硬件设备,通过开发一系列应用软件,即可实现对静态误差信号的检测,并用动态方法进行数据处理,这比某些引进的昂贵的专用数据处理系统来得合算。我们研制的以IBM—PC机为基础的信号处理系统,包括用FORTRAN语言、BASIC语言、汇编语言以及混合语言编程的数据采集存取、统计处理、圆度评定、时序分析预测、功率谱分析、灰度预测和图形显示等一系列软件。

用汇编语言来控制的信号采集,能得到所要求的采样频率。一般说来,汇编语言编制的程序只能作为子模块,其支撑主模块通常用高级语言编写,BASIC语言有丰富的图形显示功能,但其表达算法能力及运算速度都较差;相反,FORTRAN语言则具有较高的运算速度,但它在实现图形显示方面却需用特殊的支撑软件,由此可见,在一个完善的信号处理系统中,灵活地运用各种语言可使问题简单化。

为了正确使用BASIC、FORTRAN以及汇编语言编程,必须解决各种语言之间的相互连接,即所谓软接口问题。本文深入探讨了从建立磁盘数据文件和连接目标模块两个角度实现高级语言与汇编语言之间的软接口,从而解决了数据采集和实时控制中混合编程的关键问题。

二、建立磁盘数据文件实现不同语言程序之间的软接口

这种软接口方法以磁盘作为中介,汇编语言在内存设置缓冲区,当缓冲区被采集到的数据填满后,通过DOS功能调用把缓冲区数据以文件形式装入磁盘,在磁盘上的数据文件即可供BASIC和FORTRAN语言调用。与此同时,BASIC语言在调用汇编语言数据文件进行简单处理和图形显示后,可建立BASIC数据文件供FORTRAN调用,或者FORTRAN语言调用汇编语言数据文件进行处理,然后建立FORTRAN数据文件供BASIC进行图形显示。

1. 汇编数据文件的建立及调用

汇编数据文件的建立可以通过MS-DOS 2.0的功能调用来实现,MS-DOS提供了形成汇编数据文件的两种功能调用方法,一种是建立文件控制块FCB(方法1),另一种是直接使用文件号和文件名(方法2)。

表1给出了两种方法所使用的主要功能调用情况,调用时只要预先设置好入口参数,然后把功能号放在AH寄存器,并发中断21H即可。在使用方法1进行功能调用时,事先必须填好文件控制块FCB,根据FCB的内容,MS-DOS即可用顺序方式或随机方式建立文件。对于随机文件的建立,方法2提供了一种更简便的途径。

表1. 方法1和方法2功能调用情况

功 能	方法1功能号	方法2功能号
建立文件	16H	3CH
打开文件	0FH	3DH
关闭文件	10H	3EH
读文件	顺序读一个记录	4H
	随机读一个记录	21H
	随机读若干记录	21H
写文件	顺序写一个记录	15H
	随机写一个记录	22H
	随机写若干记录	28H
置磁盘缓冲区	1AH	1AH

实质上,由汇编语言通过上述两种方法所建立的数据文件是把内存中的缓冲区数据认作为字符串的ASCII码,其在磁盘上的文件格式是ASCII码的字符表示,因此,BASIC或FORTRAN语言在调用汇编数据文件时须通过函数转换把磁盘上的字符串表示成ASCII码(内存区数据)再作进一步处理。下边分别给出BASIC和FORTRAN语言调用汇编数据文件MYDADA.DAT的实例。

例1. BASIC语言调用汇编数据文件

```

10 DIM A(3, 100), V$(100)
20 OPEN "B, MYDATA.DAT" AS #1
   LEN=100
30 FIELD #1, 100 AS B$
40 FOR I=1 TO 3
50 GET #1, I
60 FOR J=1 TO 100
70 V$(J)=MID$(B$, J, 1)
80 A(I, J)=ASC(V$(J)): PRINT
   A(I, J):
90 NEXT J
100 PRINT
120 NEXT I
130 CLOSE #1
140 END

```

例2. FORTRAN语言调用汇编数据

文件

```

1. CHARACTER A(300)
2. OPEN(1, FILE='B, MYDATA.
   DAT',
   STATUS='OLD', ACCESS=
3. 1 'DIRECT', FORM='UNFORM-
   ATTED', RECL=1)
4. DO 10 I=1, 300
5. READ(1, REC=I) A(I)
6. M=ICHAR(A(I))
7. WRITE(*, *) M
8. 10 CONTINUE
9. CLOSE(1)
10. STOP
11. END

```

例1 用了两个BASIC函数,一个是MID\$函数,另一个是ASC函数,前者把字符串分离成单独的字符,后者则实现了从字符到ASCII码的转换。例2使用了一个FORTRAN内部函数ICHAR,其目的也是为了实现从字符到ASCII码的转换。

2. FORTRAN和BASIC语言数据文件的建立及相互调用

前已述及,利用FORTRAN和BASIC语言数据文件的相互调用可弥补两种语言自身的缺陷。从理论上讲,这种调用是完全可实现的,但是,由于文件存取方式和记录格式的不同(表2给出了两种语言在不同情况下的存贮方式),其数据文件之间的调用往往必须采取一定的技术措施才能实现。

如表2所示,对于FORTRAN语言所建立的数据文件,不管是随机方式或顺序方式其无格式记录总是以二进制形式存贮的,而有格式记录则以ASCII码存贮;对BASIC语言建立的数据文件,其顺序方式以ASCII码存贮,而随机方式则以二进制码存贮。另外,由于BASIC语言建立的随机文件只允许直接存贮字符串数据,因此,数值数据在存入数据文件之前必须转换成字符

串, 而从数据文件读回时又要转换回来。

表2.FORTRAN和BASIC语言的存贮方式

语言 存取方式 记录形式	FORTRAN语言		BASIC语言	
	顺序文件	随机文件	顺序文件	随机文件
有格式记录	ASCII码	ASCII码	ASCII码	二进制码
无格式记录	二进制码	二进制码		

针对BASIC和FORTRAN语言建立数据文件的不同点, FORTRAN 语言建立的有格式数据文件和BASIC 语言顺序方式建立的数据文件相互调用比较好掌握, 因为它们都是以ASCII码存贮的(如表2所示), 下边给出FORTRAN以有格式随机方式调用BASIC 顺序数据文件的实例。

例3

```

10 DIM A(1, 9)
20 OPEN 'B, SBAS.DAT' FOR OUTPUT
   AS #1
30 FOR I=0 TO 1
40 FOR J=0 TO 9
50 B$=' '
60 A(I, J)=10*I+J, B$=B$+STR$(A(I, J))
70 L=INSTR(B$, ' ')
80 IF L=0 THEN B$=B$+'.0'
90 B$=RIGHT$(B$, 6)
100 PRINT #1, B$,
110 NEXT J, B$=CHR$(13)+CHR$(10)
120 NEXT I
130 CLOSE #1
140 END

```

(a). BASIC建立顺序数据文件 SBAS.DAT

```

DIMENSION A(20)
OPEN(1, FILE='B, SBAS.DAT',
      STATUS='OLD',
1 ACCESS='DIRECT', FORM='FORMATTED', RECL=120)
READ(1, 30, REC=1)(A(I), I=1, 20)
30 FORMAT(20F6, 1)

```

CLOSE(1)

WRITE(*, *) (A(I), I=1, 20)

STOP

END

(b) FORTRAN有格式随机方式调用 SBAS.DAT

例3(a)中语句110用以以“回车一换行”来结尾记录, 这主要是为了与例3(b)中有自动结尾标志的FORTRAN 有格式记录相对应。对于表2中存贮方式为ASCII码的其它数据文件的调用, 其基本原理均与之相同。

从理论上讲, 由于BASIC 语言随机方式建立的文件和FORTRAN 语言无格式方式建立的文件都是以二进制码存贮的, 所以它们之间的相互调用也应该容易实现, 但实质上, 实型数据的文件调用还容易出错。作者发现: 对于整型数据, 表2中存贮方式为二进制码的文件调用较易实现。

此外, FORTRAN数据文件的无格式记录是以二进制码存贮的, 其与BASIC顺序文件的 ASCII码存贮方式不同, 相互调用较难。从表2可看出, BASIC 随机文件与FORTRAN的有格式文件的相互调用似乎也是不可能的, 但实质上, 只要利用BASIC的某些函数转换, 它们之间的调用就很容易实现。例4给出BASIC随机方式调用FORTRAN建立的有格式顺序文件的实例。在图中, BASIC语言VAL函数的正确使用是实现调用成功的关键。对于FORTRAN有格式文件调用BASIC的随机文件, 其原理相同, 只不过这时要正确使用BASIC语言的STR\$函数, 同时注意在BASIC程序中加上记以结尾语句(如例3)。

例4

```

DIMENSION A(10)
OPEN(1, FILE='B, SFOR.DAT',
* STATUS='NEW', ACCESS=
* 'SEQUENTIAL', FORM=
* 'FORMATTED') DO 10 I=1, 10
10 A(I)=10*I+1

```

```

WRITE(1, 30) (A(I), I=1, 10)
30  FORMAT(10F12.3)
CLOSE(1)
STOP
END

```

(a). FORTRAN建立有格式顺序文件 SFOR.DAT

```

10 DIM BB$(10)
20 DIM A(10)
30 OPEN#B, SFOR.DAT# AS#1 LEN=120
40 FIELD#1, 120 AS B$
50 GET#1, 1
60 J=1
70 FOR I=1 TO 10
80 BB$(I)=MID$(B$, J, 12)
90 A(I)=VAL(BB$(I))
100 PRINT A(I)
110 J=J+12
120 NEXT I
130 CLOSE #1
140 END

```

(b). BASIC调用SFOR.DAT

三、连接目标模块实现高

级语言与汇编语言的软接口

建立磁盘数据文件来实现高级语言与汇编语言的软接口解决了大容量数据采集的存储问题,但是,由于这种方法要经过磁盘中间介质,实质上是一种离线处理,处理时间较长,这在实时性强的信号处理中是不满足要求的。

如果用高级语言作为信号处理系统的主导语言,在信号处理时通过调用汇编语言对信号采样,并将采样到的数据存入指定的内存空间,然后对数据进行分析与计算,这就可完成信号的在线处理。

我们分别用BASIC和FORTRAN语言作为主导语言调用汇编语言。在正确解决其参数传递的基础上,只要把高级语言和汇编语言程序分别编译,通过LINK命令把其目

标文件连接在一起,生成一个统一的可执行文件。运行过程中用高级语言的CALL语句即可实现对汇编子程序的调用。

1. BASIC语言与汇编语言软接口的实现

BASIC语言调用汇编语言有两种常用的方法,一种是把汇编程序翻译成机器码并把它当作为BASIC程序的数据输入,另一种是把汇编程序作为文件的形式输入,后一种方法虽比前一种方法方便,但它却存在着程序修改、调试困难等缺点。

例5是使用连接目标模块实现的BASIC语言调用汇编语言求和的,该例汇编子程序SURTE有三个参数A和B为加数,C为和数,当BASIC语言调用汇编子程序时,参数的传递通过堆栈来进行,例6给出了调用前后堆栈指针的变化情况。

例 5

```

1: 10  A%=2
2: 20  B%=3
3: 30  CALL SURTE(A%, B%, C%)
4: 40  PRINT C%
5: 50  END

```

(a) BASIC语言主程序

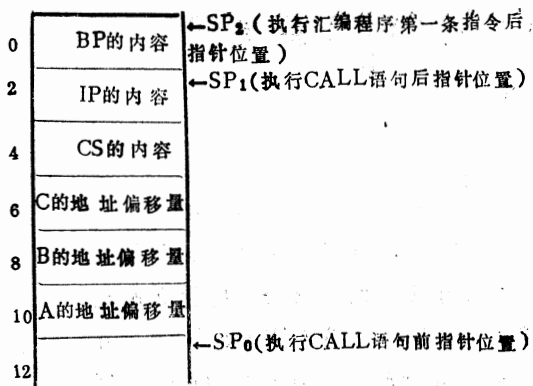
```

1: CODE SEGMENT * CODE*
2:     ASSUME CS, CODE
3:     PUBLIC SURTE
4: SURTE PROC FAR
5:     PUSH BP
6:     MOV BP, SP
7:     MOV SI, [BP]+10
8:     MOV AX, [SI]
9:     MOV SI, [BP]+8
10:    ADD AX, [SI]
11:    MOV DI, [BP]+6
12:    MOV [DI], AX
13:    POP BP
14:    RET 6
15: SURTE ENDP
16: CODE ENDS
17:    END

```

(b) 汇编语言子程序

例 6



BASIC程序调用汇编子程序后堆栈变化

为了正确传递参数，通常利用BP寄存器在堆栈内寻址得到参数变量的地址，这时候，汇编程序的最初指令就要求为：

PUSH BP

MOV BP, SP

若传递的参数有m个，则第i个参数就在存贮单元： $[BP] + 2(m - i) + 6$

中，例5中的第二个参数B的地址就存贮在 $[BP] + 8$ 单元中。

汇编语言子程序的返回语句为RET 2m，只有这样才能使堆栈指针指向调用前位的位置。

2. FORTRAN语言与汇编语言软接口的实现

为了实现FORTRAN 20与汇编语言的软接口，可以通过目标模块实现其参数的正确传递。

例7是连接目标模块实现FORTRAN语言与汇编语言的软接口。汇编程序通过不断修改缓冲区数据地址指针和数组区地址指针来实现把缓冲区长为二个字节的128个整型数传送给数组。同BASIC语言调用汇编语言一样，被传递参数的地址指针也是通过堆栈来设定的。例8是FORTRAN程序调用汇编程序前后堆栈的变化情况，从例中可看出，FORTRAN语言执行CALL语句以后压入堆栈的参数地址不但包括偏移量同时

例 7

```
1,  $STORAGE, 2
2,      DIMENSION KDT (128)
3,      CALL SURTE(KDT
4,      (1))
5,      WRITE(*, *) (KDT
6,      (I), I=1, 128)
7,      STOP
8,      END
```

(a) FORTRAN语言主程序

```
1,  DATA      SEGMENT
2,  DTA        DW      128 DUP
3,  DATA      ENDS
4,  CODE      SEGMENT
5,  DGROUP    GROUP  DATA,
6,  ASSUME    CS, CODE,
7,  DS, DGROUP
8,  SURTE    PROC    FAR
9,  PUBLIC    SURTE
10, PUSH     BP
11, MOV      BP, SP
12, PUSH     DS
13, MOV      AX, DATA
14, MOV      DS, AX
15, MOV      SI, OFFSET
16, DTA
17, LES      BX, [BP]
18,          +6
19, MOV      CX, 128
20, CON:     MOV      AL, [SI]
21, MOV      ES, [BX],
22, AL
23, INC      SI
24, ADD      BX, 2
25, DEC      CX
26, JNZ      CON
27, POP      DS
28, POP      BP
29, RET      4
30, SURTE    ENDP
31, CODE     ENDS
32, END
```

(b) 汇编语言子程序

一种实现UNIX系统与VAX/VMS系统间 异步通讯的软件

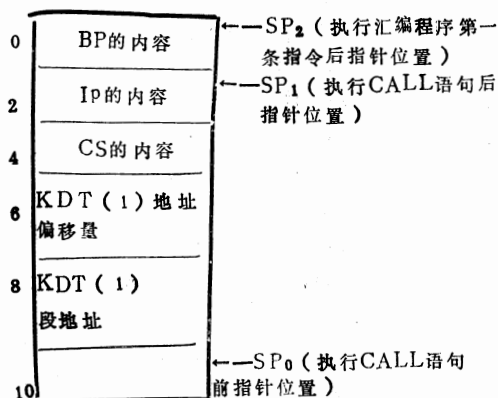
兰州电机厂计算站 魏 军

摘要 本文提出一个在UNIX多用户操作系统与VAX/VMS系统之间实现异步通讯的软件。介绍了软件的功能和实现方法以及程序框图。此软件的特点是无需附加的硬件支持，UNIX上任一终端用户均可进入或退出VAX/VMS系统。两个系统之间的文件可双向传送，其中汉字代码可相互自动转换。该软件用C语言写成，可容易地移植到其它多用户系统上。

一、引言

UNIX是国内外广为流行的一种分时多用户操作系统，它的结构性能好，便于修改

例 8



FORTRAN调用汇编子程序后堆栈变化

也包含段地址，也即参数地址在堆栈占四个字节，这一点与BASIC语言调用汇编语言的情况大不一样，因此使用时必须注意。另外，如果汇编程序也用BP寄存器在堆栈内寻址，则其最初两条指令同前一样。此时若传递的参数有 m 个，则第 i 个参数的地址就在存贮单元：

$$[BP] + 4(m - i) + 6$$

中。子程序返回指令的格式应为RET 4 m 。

例7中传递参数有一个，其段地址和偏移量通过LES指令来传递，而返回指令为RET 4。

上面讨论了FORTRAN、BASIC与汇编语言的软接口，为了正确使用这种软接口，

我们还要说明几点。

- (a) 汇编子程序终结语句不能有标号。
- (b) 汇编子程序必须保护用到的寄存器。
- (c) 汇编子程序的过程说明属性应为FAR，过程名本身要有PUBLIC说明。

四、结论

1. 汇编语言可用两种方法建立磁盘数据文件，其文件格式为字符串，BASIC和FORTRAN调用汇编数据文件时必须正确使用ASC函数和ICHAR函数。

2. BASIC和FORTRAN语言建立的任意格式任意存取方式的文件只要正确利用转换函数，其相互调用都可实现。

3. 连接目标模块实现高级语言与汇编程序的软接口简便、实用，但在进行参数传递时要注意两种语言调用时其堆栈内容的差异。

4. 通过高级语言程序与汇编程序的参数传递，可直接利用高级语言为汇编语言指定的缓冲区建立磁盘数据文件，这在大数据量的实时检测中既可解决存贮问题，又可解决速度问题。（参考文献从略）

和移植。VAX/VMS则是DEC公司专为其VAX系列产品而配置的、功能很强的操作系统。这两种系统在国内外都拥有相当广泛的市场和用户。在这两种不同的系统之间,建立较完备的信息交换功能,是一件有普遍意义的事情。

二、软件功能

1. 终端仿真

配有UNIX的多用户计算机作为VAX机的下层机,通过其上面的串行智能终端接口板和RS232-C电缆与VAX机的终端接口相连接,也可以与VAX机的以太网上的终端服务器DECserver连接。下层机上的任一终端用户可通过与VAX相连的接口,成为VAX/VMS系统的一个终端用户,享用VAX系统的软硬件资源,如VAX/VMS, DCL命令,各种编译以及硬盘,磁带和行式打印机等。

2. 文件传送

在两系统之间双向传送文件,主控权在下层机。通过菜单选项,分别调用发送文件,接收文件,发送汉字文件,接收汉字文件等功能。

3. 汉字代码转换

UNIX支持的汉字终端常采用四字节汉字代码,而CVMS的汉字则是两字节代码。该软件可实现这两种汉字代码的相互转换,使下层机与上层机的汉字文件也达到共享。

三、实现方法与编程

1. 通讯参数设定

通讯参数设定的正确与否,对于异步通讯的成败是至关重要的。对于UNIX下层机来说,有两个端口的参数需要设定,一个是当前用户终端,另一个则是与上层机相连的终端接口。对上层机,仅需设定与下层机相连的那个终端接口。除此以外,就是重新定

义双方发送文件和接收文件状态下的通讯规程。

在UNIX下层机的用户终端仿真VAX机的终端状态下,该用户终端本身的接口可通过UNIX命令

```
stty -echo
```

定义之。即取消主机将键盘输入的信息,回送到CRT的功能。

对通讯口,则不论是在仿真状态,还是在发送与接收文件状态,均以下列命令设定:

```
stty -echo cbreak >/dev/ttyA
```

ttyA是通讯端口设备名,cbreak的含义是不用等待出现换行符或回车符,一次传送一个字符。其余参数或选项均可按普通终端处理。

需要做的另一件事是修改UNIX文件系统中/etc目录下的ttys文件。做法非常简单,用edit或vi命令,编辑ttys文件,将文件中对应于通讯端口ttyA记录行的首字符,由来的“1”,改为“0”。数值“0”意味着UNIX操作系统,将不发送注册信息到此端口。否则,当该端口与VAX终端接口相连后,会将对方看成是本系统的一个终端用户,不断向对方发送注册信息,要求对方注册,使UNIX机不可能成为VAX的下层机。

对于VAX机的通讯端口,首先要修改的参数是字符位数char_bits和奇偶校验parity。UNIX对于字符的表示,较严格地定义为7bits,这是难以改动的。VAX机上字符表示是8bits,无奇偶校验。但所连接的终端可以是8bits,也可以是7bits。使用一条VMS/DCL命令,便可轻而易举地将通讯口的这两个属性设定的与下层机完全一致。

```
$ set term/noeight/parity=odd(or even).
```

其他属性则需根据仿真,发送文件和接收文件三种情况而定。

在仿真状态下,仅需将VAX对应的通

讯端口改定为noreadsync即可。在VAX接收文件状态下, 则用下列命令设定:

```
$ set term/noecho/notty/readsync
```

在VAX发送文件状态下, 将上一条命令中限定词readsync改成noreadsync。其余参数同VT220终端。

不言而喻, 双方通讯口的波特率必须一致。一般为了提高传输速度应选较高一些, 但应视具体机型而定。通常采用试验的办法确定。本程序在传送文件时采用9600波特较为适宜, 但顾及到终端仿真状态, 选定为4800波特。

2. 通讯规程问题

多用户计算机系统的智能终端接口板, 绝大部分用来连接终端、打印机等外部字符设备。此类设备I/O速度很低。所以许多智能接口板上的输出缓存区设有自动响应外部设备所发出的 CTR/Q、CTR/S (XON, XOFF) 信号的功能, 而它的输入缓存区, 常常不具备向外设发送CTR/Q、CTR/S的功能。

Dual 68000 83/80超级微机的SIO-4串行智能接口板便是一例。当这类接口板与其它多用户机的接口板相连时, 如果对方I/O速度较快, 就会在它接收对方发送的数据时, 丢失数码。因为在发送方发送速度快于接收方处理字符的速度时, 当接收方缓存区已接近溢出, 而该口又不能发CTR/S使对方停止发送, 因此在传输过程中丢失数码是必然的。解决此问题的途径之一, 就是取消原接口上自有的XON、XOFF功能, 重新建立双方默认的通讯协议, 使XON、XOFF的发出与响应完全处在通讯程序的控制之下。以保证传送数据与文件的完整性。

本通讯程序的文件发送与接收就是采用了这种方法。

3. 终端仿真程序

终端仿真的主要问题是如何实现处理仿真终端随机读键盘信息和随时接收上层机发来的信息, 二者之间程序并发矛盾。利用

UNIX所提供的FORK()系统调用, 可以解决上述矛盾。

在仿真程序中, 通过调用FORK()生成一个新的进程, 它和仿真程序并发。FORK()生成的进程专用于处理接收通过指定接口由上层机发来的信息。它循环地读通讯口, 只要有信息, 便立即将其送屏幕显示。而仿真程序本身的进程则负责读键盘和向上层机发送键盘信息。只有当按下事先规定的特殊键ESC时, 仿真程序将调用KILL()函数, 将FORK()进程杀死, 并使程序返回UNIX系统和程序的菜单。仿真程序框图如图1所示。

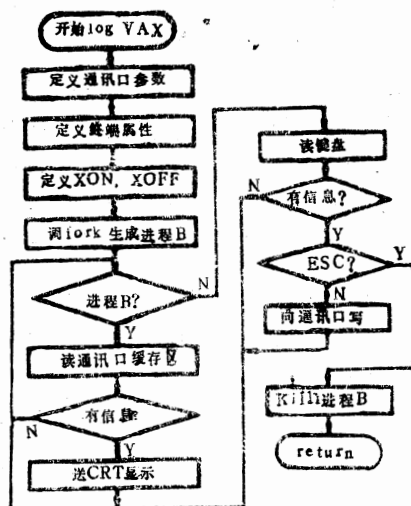


图1 UNIX用户终端仿真VAX/VMS终端程序框图

4. 发送文件程序

从下层机向上层机上发送文件, 是通过下层机上发送程序和上层机上的接收程序, 相互协调配合而实现的。发送程序一经调用, 首先向上层机传送一命令行, 激活上层机上对应的接收程序, 接收程序根据对方发来的命令行参数, 创建并打开由命令行指定的文件。此后, 双方建立通讯协议, 正式传送文件。文件的发送与接收以行为单位。由于VAX机为终端设置的缓冲区为255个字节, 故传送行长不宜超过255个字节。当传送的文件中含有汉字时, 可根据主菜单选项,

指定发送汉字文件的功能选项,调用同一个发送程序。该程序激活上层机的另一个接收程序,此接收程序与第一个接收程序的唯一不同之处,仅在于它可将四字节汉字代码转换成CVMS支持的二字节汉字代码。发送文件程序框图如图2所示。

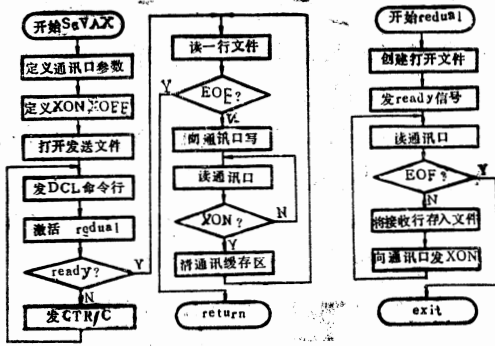


图2 UNIX发送文件与VMS接收文件程序框图

5.接收文件程序

与发送文件相仿,UNIX接收VAX的/VMS文件,也是由两个系统上的一对接收与发送程序实现的。下层机上的接收程序运行起来之后,也是首先通过传送VAX/DCL命令,激活VAX上的发送程序,然后转入等待接收文件状态。VAX上的发送程序打开由对方指定的文件,成行的读入,然后成行地发送。其中汉字代码的处理方法与前面所述方法基本相同,只不过此次是将二字节汉字代码转换成四字节代码而已。

稍有麻烦之处在于,当被传送的VAX/VMS数据文件行长大于80列时,下层机收到的该行数据会自动地以80列为界,分成若干行。这是因为VAX/VMS向终端口传送数据时,在80列处自动加上一个换行符,该符号不能用修改VAX通讯口参数的办法取

消之。这样一来,当传送数据的行长大于80列时,程序中所规定的通讯协议便不能保证一致性,会发生丢行丢字现象。

解决的办法有两种,其一是将VAX机上对应下层机的通讯口设置成132列行宽,这样当被传送的文件行不超过132列时,便能保证通讯协议的一致性,不会发生数据紊乱。其二是在文件传送之前,将大于80列的行分为小于80列的若干行,截断处标记一特殊字符,然后依次发送。对方接收后,对指定列进行检测,若是上一行的续行,则删除标记列,并将此行与上行接成一行,最后再写入接收文件中。这样便可保证所接收的文件与原文件完全相同。第一种方法在行长超过132列时便不宜使用,第二种方法的缺点是传送速度稍有降低。但对任一行长的文件均可正确地传送。本程序采用第二种方法。

接收程序框图如图3所示。

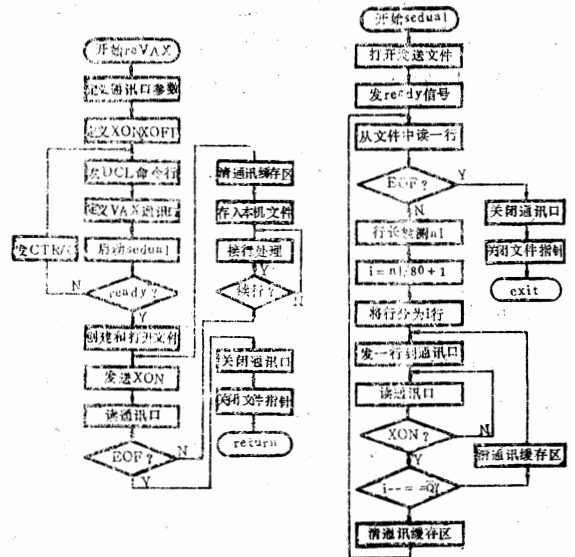


图3 UNIX接收文件和VAX/VMS发送文件程序框图

新一代关系数据库—FoxBASE +

近年来随着越来越多的以80286, 80386芯片为CPU的IBM PC/AT 及各种286、386系列微机的推出, 关系数据库dBASE III越来越不能满足用户的需要。为进一步发挥286, 386系列微机的潜力, 提高关系数据库系统管理数据的能力, 美国SCO公司于1987年6月推出了新一代数据库系统—FoxBASE +。

FoxBASE + 硬件环境为IBMPC/AT及兼容机, 并可在国产长城系列微机、浪潮系列机等以80286, 80386为CPU 微型机上运行。它的操作系统环境即可是XENIX也可可是DOS。由于FOXBASE + 与当今广为流行的dBASE III 系统完全兼容, 因此, 它的问世受到了广大用户的热烈欢迎。用户在dBASE III上开发的程序、数据库文件、经验与习惯都得到了充分的尊重。dBASE III上的程序可以原封不动地移到FoxBASE + 上

正确运行, 且数据处理速度大大提高。

同时FoxBASE + 较dBASE III 更加完善增加了一部分命令, 函数和系统调用功能。由于它是个多用户系统, 因此具有完善的文件, 记录锁定等保护措施。下面简述一下FoxBASE + 的特点。

1. FoxBASE + 在dBASE III 命令的基础上增加了一些命令。它们是:

(1) DIMENSION 该命令可产生一维、二维内存变量数组。

(2) SUSPEND, RESUME 该命令为一主要的程序调试工具。SUSPEND使正执行的程序暂停以便检查内存变量。RESUME将继续程序的运行。

(3) 增设HISTORY功能

为方便用户调试程序, FoxBASE + 提供了一种将已执行过的命令存储起来的功
能。可用LIST, DISP命令显示出最近执行

四、使 用

在下层机上装入主程序DVCMM, 并将其安装在UNIX文件系统/USR/BIN 之下, 使每一个UNIX用户均可运行。将上层机上的接收与发送程序以及汉字转换程序装在VMS系统的SYS\$SYSEXEC目录下。在下层机任一终端上运行主程序, 屏幕便给出主菜单, 共有七个选项。HELP选项为1, 用来说明怎样使用这个程序; LOGVAX选项为2, 用来注册到VMS系统中, 作为VMS的终端用户; 选项3, 4分别为发送和接收ASCII码文件; 选项5, 6则是发送和接收汉字文件; 选项7为退出程序。只要键入对应的选项编号, 便可实现前面所述的各

种功能。

本软件使用后, 使DUAL 68000微型机成为VAX 8350超级小型机的一个子系统。68000机上的用户可方便地利用VAX机的软硬件资源。而68000机则作为VAX前置机, 做一些数据录入, 编辑和作一些管理系统运行的预处理工作。对于需要内存大。磁盘空间大的生产管理信息系统的运行, 可通过通讯程序, 将经过下层机预处理过的数据送至上层机, 由上层机运行信息管理系统程序, 最后将结果返送至下层机。

经实际使用证明, 该软件使用方便, 可靠, 数据传送速度较快。不失为一种实现异型机之间信息交换与资源共享的经济有效的手段。(参考文献略)

的20条命令。

2. FoxBASE+ 还增加了一些函数

(1) 系统函数

DISKSPACE() 返回默认10录中当前自由空间的字节数。

ERROR() 返回由ON ERROR条件引起的错误号。

OS() 返回当前操作系统名。

SYS() 该函数根据各种自变量的值返回各种有用的系统信息。

MESSAGE() 返回当前的错误信息串。

NDX() 返回当前工作区中正工作的索引文件名。

(2) 字符串操作

INKEY() 返回输入键的ASCII码值。

READKEY() 返回自全屏幕命令出口时的按键。

LSALPHA() 返回逻辑值。字母为T否则F。

ISLOWER() 字符串为小写字母，返回T。

ISUPPER() 字符串为大写字母，返回T。

LEFT, RIGHT() 从左(右)截取一个字符串。

LTRIM() 删除字符串的前空格。

RTRIM() 删除字符串的尾空格

MAX(), MIN() 返回两个表达式的最大(小)值。

STUFF() 修改字符串。

REPLICATE() 重复字符串。

(3) 其它函数

MOD() 返回除法的余数。

RECCOUNT() 返回当前工作区中

记录数目。

RECSIZE() 返回当前工作区中记录的大小。

3. 多用户的特性

单用户和多用户编程的主要区别就是是否允许多个用户分享数据，同时为了管理不可避免的冲突而采取一系列的措施。

(1) 数据共享

许多FoxBASE+要求全部或部分数据库具有唯一使用的特性，能够修改数据的命令都要求唯一地存取数据(即独家占有数据库)。此外FoxBASE+包括一些出错处理子程序处理多个用户同时调用一个库时出现的错误。

(2) 直接加锁函数

FoxBASE+有三种加锁函数，FLOCK()用来锁定文件，RLOCK()，LOCK()用来锁定记录。FoxBASE+还有二个解锁命令UNLOCK用来释放当前库中所上的锁，UNLOCK ALL命令用来释放所有工作区中全部记录锁和文件锁，

(3) “ON ERROR”操作

可以用ON ERROR命令来设置错误陷阱，用ERROR()和MESSAGE()函数来识别错误并用RETRY命令重复操作。

4. FoxBASE+ 程序文件的编译

FoxBASE+不仅可以直接执行所有的dBASE III程序(·PRG文件)还可对这些程序编译后执行。

可以在系统命令级使用foxCOMP命令把数据库原程序(·PRG文件)编译成目标文件(·fox文件)。编译过的程序不但可以避免出现语法错误，而且可以达到最大程序装入速度。

天津联合软件公司 周 琼

STE总线介绍

北京航空航天大学工业部303所

陈晓午

摘要: STE bus是已被IEEE正式承认的开放式结构总线系列的最新成员。在欧洲,该总线已被广泛用于智能化仪器仪表及工业控制设备中。在美国,它也受到越来越多的制造商的关注。本文简要介绍了STE bus的起源及它的主要特性。

一、概述

1987年12月10日, IEEE标准委员会正式批准STE总线技术规范为IEEE1000本文将简要地评述STE总线的发展历史, 介绍这一标准的主要特性, 并探讨在低/中档嵌入式系统中采用STE总线的益处。

众所周知, VME总线是一个 开放式结构的总线, 它有广泛的电路板产品支持。32位VME总线计算机的处理能力及价格对于一些低/中档任务是不合适的, 许多任务八位微处理机就足以胜任。这些低目标、低代价的计算任务可以由STE总线(以下简称为STE bus)很好地解决。

二、STE bus的基本原理

设计

在设计STE bus中, 本着实现 价格效用与性能之间的平衡这个原则, 以 VME bus为模式并简化而实现之。STE bus提供了许多与八位总线相同的特点: 可带较小的板子、简单的协议等等。STE bus特点包括

- 单独的(半高)Eurocard格式
- 处理器无依赖性
- 八位数据总线和二十位地址总线
- 异步数据传递
- 有力而简单的中断及DMA机理
- 有力而简单的多重处理能力

STE bus技术规范的经济及精美体现在STE bus板的简单性上。一个典型的总线接口仅需要6或8个IC芯片, 给处理器及I/O留出了最大的板上空间。板子设计相对来讲较为容易, 并且造价低廉, 克服了不同厂商的板子间的不兼容性。

宽范围的处理器及I/O卡适用于STE bus。处理器包括Z80系列, 8088系列、68000系列, 以及6809、32016、8052等。从属卡——存储器、串并I/O、磁盘控制器、模拟量I/O、图形及特殊用途卡, 均可适用。

三、总线信号

STE bus卡使用一个64脚DIN 41612C插座与基板相连。图1示出了插座各脚功能定义。信号可分为八组(见表1)。

表1

信号分组	定义
数据总线	$D_0 \sim D_7$
地址总线	$A_0 \sim A_{19}$
命令信号	$CM_0 \sim CM_2$
握手应答信号	\overline{ADRSTB} , \overline{DATSTB} $\overline{DATAACK}$, \overline{SYSERR}
引起注意请求信号	$\overline{ATNRQ_0}$, $\overline{ATNRQ_7}$
总线请求/认可信号	$\overline{BUSRQ_0}$, $\overline{BUSACK_0}$ $\overline{BUSRQ_1}$, $\overline{BUSACK_1}$
总线公用信号	\overline{SYSRST} , \overline{SYSCLK}
电源	+5V, +12V, -12V +VSTBY, 0V

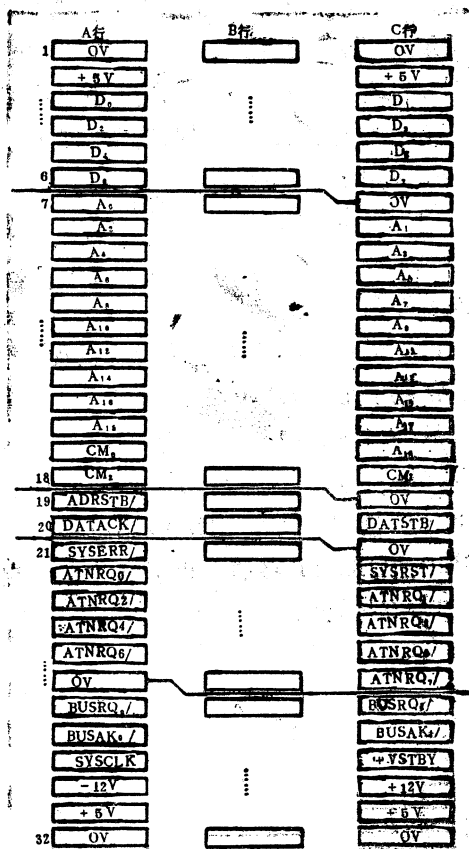


图1 STE bus各脚功能定义

四、基本总线周期

STE bus 采用五种基本的数据传递类型：存储器及I/O读、写周期，中断矢量获取周期。总线定时是异步的。

图2是读周期的例子。一个地址被放在总线上，同时一个地址选通（ $\overline{\text{ADR STB}}$ ）信号有效。周期类型（本例中为读情况）由三条命令线 $\text{CM}_0 \sim \text{CM}_2$ 确定。数据在总线上传递使用了一对相互联系的异步握手信号——数据选通（ $\overline{\text{DATSTB}}$ ）及数据响应（ $\overline{\text{DATAACK}}$ ）信号。数据选通由总线主控设备发出，而数据响应由总线从属设备发出。图2示出了CPU卡从由 $\text{A}_0 \sim \text{A}_{19}$ 地址位确定的存储器单元中获得数据的时序。在

这一读过程中，需设置 $\text{CM}_0 \sim \text{CM}_2$ 位，使 $\overline{\text{DATSTB}}$ 有效。总线从属卡提供数据，当数据在 $\text{D}_0 \sim \text{D}_7$ 线上稳定时，使 $\overline{\text{DATAACK}}$ 有效。在这期间，CPU接收数据并使 $\overline{\text{DATSTB}}$ 无效。然后，从属卡使 $\overline{\text{DATAACK}}$ 无效，一个读周期便完成了。

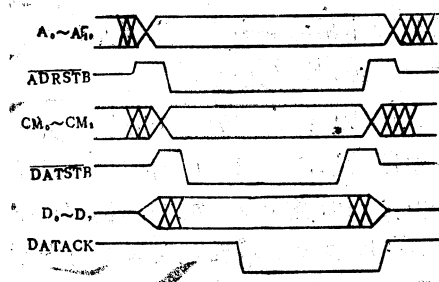


图2 读周期

写周期类似于读周期，所不同的是：在写周期，总线上的数据是由CPU卡提供的。在整个 $\overline{\text{DATSTB}}$ 为低期间，CPU卡保持住数据有效。被寻址的存储器或I/O卡在接收数据时使 $\overline{\text{DATAACK}}$ 有效。随着 $\overline{\text{DATSTB}}$ 恢复为高，则 $\overline{\text{DATAACK}}$ 也变为高，一个写周期完成。

五种周期类型由三条命令线的内容确定。三条命令线共有八种组合，其中常用的有五种，见表2。还有两种组合也被定义：一个是读一改一写周期，这个周期对于多重处理很有用。另一个是突发式传递周期，这个周期用于高速的数据块传递。

表2 命令线信号

CM_2	CM_1	M_0	周期类型
1	1	1	存储器读
1	1	0	存储器写
1	0	1	I/O读
1	0	0	I/O写
0	1	1	矢量获取

五、引起注意请求信号

引起注意信号体现了STE bus 某些最有力的特点。八条引起注意请求信号或用作中断请求、或用作DMA请求,还可按系统设计者的要求使用。这八条线是 $\overline{\text{ATNRQ}}_0 \sim \overline{\text{ATNRQ}}_7$ 。如果需要, $\overline{\text{ATNRQ}}$ 线可以由集电极开路驱动器驱动,以便允许多个卡使用同一个 $\overline{\text{ATNRQ}}$ 信号。

$\overline{\text{ATNRQ}}$ 线可作为一个I/O设备向处理器发中断请求信号线。如何响应中断请求,还要由处理器类型及产生中断的I/O设备来决定。STE bus技术规范定义了对中断的两种响应方式:隐含响应与明确响应。在对中断的“隐含响应”方式中,处理器通过登记适当的卡去查找中断源。然后,处理器必须确保从属卡停止驱动 $\overline{\text{ATNRQ}}$ 信号,例如:向从属卡写一个控制字。

在对中断的“明确响应”方式,处理器在STE bus上运行一个中断矢量获取周期。这是一个特殊的总线引导周期,在这期间处理器从产生中断的从属卡中读一个中断矢量。处理器用这个矢去检索中断服务程序入口地址表。STE bus这一矢量获取周期可以改善中断驱动系统的性能。

$\overline{\text{ATNRQ}}$ 信号的第二个用途是作为DMA请求线。如磁盘控制器芯片或类似芯片的一个DMA请求经某一 $\overline{\text{ATNRQ}}$ 线发往处理器的DMA请求管脚上。接着,处理器的DMA控制器将向磁盘控制器发出一个字节或从磁盘控制器接收一个字节数据。

上面已叙述了使用 $\overline{\text{ATNRQ}}$ 信号的模式,下面举例说明。考虑一个实际目标系统:一个CPU卡,一个磁盘控制器卡,一个采用DSP(数字信号处理)设计技术的

八端口串行通讯卡。图3示出了配置情况。该系统应用于某无线电公司,处理这些信号的计算机通过调制解调器与无线电发射机相连。信息被记录在一个温彻斯特磁盘中,磁盘还可以用于存贮顾客资料。

处理器卡SX180是以日立64180处理器为基础构成的。64180芯片运行280指令,并包括许多外围口:两路DMA通道、两个定时器、两个串行口、四个中断请求线及一个存储器管理单元。68140芯片非常适宜STE bus。具体地说:中断请求线及DMA请求线可直接馈至 $\overline{\text{ATNRQ}}$ 。SX180具有完整的STE bus接口、两个RS232串行口、最大256KB的DRAM备件。由后备电池供电的32KB静态RAM。

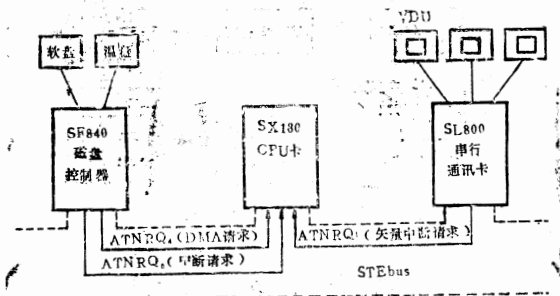


图3 ATNRQ应用

在应用中,计算机终端由中断驱动代码控制。SL800八端口串行通讯卡支持SX180处理器的矢量中断。通讯卡使 $\overline{\text{ATNRQ}}$ 有效并且在处理器卡的矢量获取周期送出一个矢量。对于八个端口中的每个端口,送出不同的矢量,这样可简化中断服务程序的任务。

该应用表现了STE bus设计的简易性与先进性。三个独立的Eurocard板,均采用标准技术在双层印制板上实现,提供了多个串行口、硬软盘控制器及一个带有大存贮阵列的快速处理器。