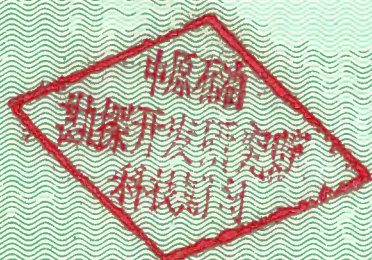




微小型计算机

开发与应用

MICRO-MINICOMPUTER
DEVELOPMENT & APPLICATION



1992

4

微小型计算机开发与应用编辑部



微小型计算机开发与应用（公开发行）

1981年创刊

编辑：《微小型计算机开发与应用》编辑部

发行：天津市邮局

出版：天津市电子计算机研究所

印刷：天津市武清县长宏印刷厂

天津市计算机学会

订购处：全国各地邮局

地址：天津市河西区友谊路宾馆南道5号

定价：0.95

邮政编码：300061

出版日期：1992年6月1日

邮局代号6—87

津工商广字0146号

国内统一刊号

ISSN1001-8786
CN12—1122/TP

《微小型计算机开发 与应用》编辑委员会

顾 问	郭平欣	
主 编	黄 侃	
副主编	王治宝	张凤枝
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
	付园明	许镇宇
	朱植松	曲庭维
	李凤祥	刘连棣
	陈力为	吴锦声
	房家国	张凤枝
	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

1992年第4期目次 (总第60期)

计算机软件

- 基于4GL开发的ORACLE DBMS数据字典
信息追踪系统……………杨晓峰 王丽东(1)
一种新的SQL * Forms菜单设计
……………李春葆(4)
ORACLE和Rdb/VMS的比较评估
……………孙海波 周 菲(6)
用C语言构造通用的下拉式菜单程序
……………许卓明 角钢(10)
介绍一种LR(K)状态集和分析表的构造方法
……………孙桂茹(14)
中西文兼容UNIX操作系统的一种汉字输入
输出实现方法……………尹竞(19)

应用实例

- CCD高速数据采集装置……………伍龙田(23)
EM—891 Ethernet监控器的实现…程湘云(27)
微机电力系统暂态稳定实验装置……………张文生(32)
微机力矩限制仪……………陈守中(34)
采用新技术(PC)改造旧磨床
……………张秀玲 张怀安(36)
分层次结构化记述方法HCP图的应用
……………王升超(39)

通讯技术

- IBM—PC系列机与多台MCS—51单片机间
的通信……………尹蓉华(41)
微机控制监测系统中IBM—PC与STD总线
间数据通讯……………张瑞祥(46)

CONTENTS

SOFTWARE

- ORACLE DBMS Data Dictionary Information, Tracing Spstem Developed
by 4GL..... Yang Xiaofeng Wang Lidong (1)
- A New Menu Designing in SQL * Forms..... Li Chunbao (4)
- The Comparison and Estimation of ORACLE and Rdb/VMS
..... Sun Haibo Zhou Fei (6)
- Writing a General Pull-down Menu Program in C Language.....
..... Xu Zhuoming Jiao Gang (10)
- An Introduction to a Method for Building LR (K) States and Parsing
Tables..... Sun Guiru (14)
- An Implementation Technique of Chinese Character I/O for Chinese Com-
patible with English UNIX System..... Yin Jing (19)

APPLICATION EXAMPLE

- CCD High Speed Data Acquisition System..... Wu Longtian (23)
- Implementation of EM-891 Ethernet Monitor..... Cheng Xiangyun (27)
- Experimental Device for Transient Stability in Power Systems.....
..... Zhang Wensheng (32)
- Microcomputer-Controlled Instrument for Limiting Moment.....
..... Chen Shouzhong (34)
- Using New Technique to Reform Old Grinder.....
..... Zhang Xiuling Zhang Huaian (36)
- The Application of Hierarchical Compact Description Chart.....
..... Wang Shengchao (39)

COMMUNICATIONS TECHNOLOGY

- An Approach for Communications between IBM-PC Series Microcomputer
and Multi-MCS 51 Single Chip Microcomputers Yin Ronghua (41)
- Data Communications between IBM-PC and STD Bus in Microcomputer
Controlled Monitoring and Testing System..... Zhang Ruixiang (46)

基于4GL开发的ORACLE DBMS 数据字典信息追踪系统

东北工学院自控系

杨晓峰

(邮 码 110006)

沈阳建设银行

王丽东

摘要 本文针对某大型钢铁企业计算机管理信息系统所使用的ORACLE数据库,运用其提供的先进的4GL工具SQL*FORMS,实现了全屏幕式ORACLE DBMS数据字典信息追踪系统(DDTS),为数据库管理员、程序开发人员及最终用户使用数据字典提供了一种方便、简洁的方法。

1 引言

随着数据库系统技术的发展,数据字典的概念也不断充实和丰富。数据字典是关于数据描述信息的集中的库,它包含数据库数据或元数据(META DATA),字典本身也形成了一个特殊的库,它是数据处理人员在数据库设计、实现、运行、维护以及扩充各个阶段中控制并管理信息的工具^[1]。所以,一些大型数据库管理系统都具备数据字典。ORACLE数据库是世界上著名的四大关系数据库之一,目前在國內得到了越来越广泛的应用。ORACLE数据字典就是由ORACLE DBMS提供的一个一体化、交互式数据字典功能程序,是ORACLE最重要的成分之一,是ORACLE DBMS一切数据活动的中心^[3]。

我们在某大型钢铁企业MIS设计开发过程中,通过对ORACLE数据字典的使用,发现它只提供一种操作方式,就是用SQL*PLUS中的SQL命令来完成的,但是,这种方法对最终用户,甚至一些较熟练的应用开发者来说,存在一定的困难,主要有以下几点:

a.数据字典中表数众多(43个),表名难于记忆;

b.用户往往不清楚要检索的信息存放在哪个表中;

c.数据字典表中字段均为英文编写,即

使信息检索出来,有的也难于辨别其含义;

d.屏幕显示信息用户难以阅读和接受。

基于上述原因,为了高效地使用数据字典,需要开发一个对用户友好的、操作方便、汉字提示的实用程序。因此,我们运用ORACLE DBMS提供的先进的4GL工具SQL*FORMS开发了ORACLE数据字典信息追踪系统(以下简称DDTS)。系统投入运行后,获得用户好评。

本文详细介绍了DDTS所基于的ORACLE数据字典信息的描述和实现技术。

2 ORACLE数据字典信息描述

2.1 ORACLE数据字典的作用

ORACLE数据字典用来存放数据定义和系统运行控制信息。

a.最终用户用来找寻其标识,其拥有的数据库客体(表,空间定义,视图,索引,聚簇,同义词)的名称;

q.程序员用来了解各实体的名称,使用特点及其格式;

c.系统设计者用以了解数据库的名称,使用和结构规则;

d.数据库管理员用以控制和监督数据库的使用和存储分配,了解系统性能,及时掌握数据库的动态。

可见,ORACLE数据字典信息为各类用户所使用。

2.2 DDTS所基于的数据字典信息描述与分类

ORACLE数据字典是由SYS和SYSTEM帐户下的一组表和视图组成并对PUBLIC用户授权,各个用户都可使用,利用如下SQL命令可得到数据字典表的清单[3]。

```
SELECT * FROM DTAB;
```

字典表共有43个,但是,正和我们前面所说的那样,用户使用很困难,因此,我们通过对数据字典信息的分析,进行了归纳和分类,去掉了不经常使用的一些信息,又考虑到每一类信息又分布在几个字典表中的情况以及DDTS的通用性,我们从中选取了适当的表作为该类信息追踪程序的基础信息表。基础信息表的描述及对应的程序名称如表1所示。

表1

表 名	信息内容	对应程序名
SYSUSERLIST	所有用户及特权信息	MYDD
CATALOG	用户自建或有权存取表的信息	CT_FORM
COLUMNS	表中的字段信息	
SYSTABAUTH SYSCOLAUTH	有关表授权及表中字段授权信息	GD_FORM
SYSINDEXES	表上所建索引信息	SL_FORM
CLUSTERS	用户建立的聚簇信息	CC_FORM
SPACES TABALLOC TABQUOTAS	表的空间存贮及空间定义信息	TS_FORM
CATALOG SYSTABAUTH VIEWS	用户建视图及可存取的视图信息	CV_FORM
PRIVATESYN PUBLICSYN	用户建同义词信息	CS_FORM
SYSPROGS	用户预编译程序信息	PG_FORM

3 DDTS设计

3.1 DDTS的总体结构

DDTS是利用ORACLE提供的先进的第四代语言工具SQL * FORMS设计完成的。一个FORM程序由一个或多个块组成,每个块对应一个数据库表或视图,也可以不对应数据库表(如菜单),一个块由一个或多个域组成,每个域对应数据库表中的一个字段,也可以是控制域(如菜单上的功能选择域)。在域级、块级、FORM级上可设置触发器来完成一定的功能[4]。

本系统由九个FORM程序组成,它们分别是:MYDD,CT_FORM,GD_FORM,SL_FORM,CC_FORM,TS_FORM, CV_FORM,CS_FORM和PG_FORM。程序与程序之间采用分级菜单调用形式,每一个FORM程序都完成一类信息的操作(见表1)。

DDTS的总体结构如图1所示

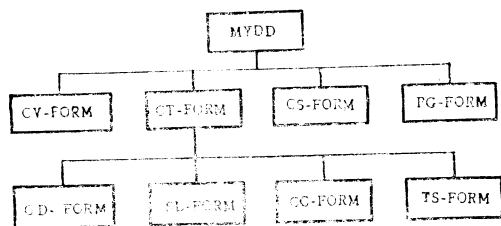


图1 DDTS总体结构

3.2 设计方法

我们在利用SQL * FORMS进行设计过程中,充分运用了其独特的触发器功能,通过光标键打功能选择,全局变量信息传递,自动按缺省条件查询等方法,给用户提供了方便,灵活的操作方式,为了防止用户的误操作,我们对其提供的功能键进行了重新定义和屏蔽。

下面,我们以光标键打和功能键重新定义及屏蔽方法为例作具体说明。详细介绍见文献[2]。

3.2.1 光标键打方法

光标键打方法是功能选择(或记录选择)的一种方法,就是把光标放在所要选择的功能项上(或记录上),按某一功能键(采用SQL * FORMS缺省定义或自行定义的)激发

所设计的触发器，则执行该触发器功能。

例如，在MYDD FORM程序中，光标进入DD1块执行查询后，则屏幕上显示ORACLE所有用户标识及其特权（见图2），这时按功能键[NEXT RECORD]（在IBM-PC/XT机上为[↓]键）把光标放在你自己的用户标识上，然后按回车键，则激发如下定义在此块上的块级触发器KEY—NXTFLD。其内容如下：

```
SELECT 'X' FROM DUAL /* 第一步 */
WHERE DD1.USERNAME=USER
#COPY : DD1 . USERNAME :
DD2.UNAME/* 第二步 */
#EXEMACRO NXTBLK; GOFLD
SE; /* 第三步 */
```

用户名	建立时间	三种特权		
		C	D	R
SYS	09-DEC-88	Y	Y	Y
PUBLIC	29-NOV-88			
SYSTEM	07-JAN-88	Y	Y	Y
SCOTT	29-NOV-88	Y	Y	Y
ADAMS	29-NOV-88	Y	Y	Y
JONES	29-NOV-88	Y	Y	Y
CLARK	29-NOV-88	Y	Y	Y
BLACKE	29-NOV-88	Y	Y	Y
SDD	29-NOV-88	Y	Y	Y
BJQPK	30-NOV-88	Y	Y	Y

图2 DD1块屏幕图

首先执行第一步，为了安全起见，要检查一下光标是否放在你自己的用户标识上（如你注册的用户名为SCOTT），如果是，则执行第二步，否则不予执行。第二步把DD1块中所显示的你的用户标识名拷贝到DD2块中的用户名数据域UNAME中，该拷贝命令为SQL*FORMS触发器中的宏拷贝命令。第三步执行进行DD2块的功能选择域SE中，#EXEMACRO命令为SQL*FORMS触发器中的宏执行命令。DD2块屏幕图

见图3。

用户名 BJQPK
0--退出。 1--查询建表情况。 2--查询建视图情况。 3--查询建同义词情况。 4--查询预编译情况。 请选择 _____

图3 DD2块屏幕图

3.2.2 功能键重新定义及屏蔽

功能键重新定义主要是利用触发器中的宏功能语句完成的，如在上述中所定义的触发器KEY—NXTFLD，它所对应的缺省功能键是[NEXT FIELD]（在IBM PC/XT机上为回车键），而我们对它做了重新定义，完成的是光标键打功能。为了防止用户的误操作，要对没有使用的SQL*FORMS缺省的功能键进行屏蔽，这一点是非常重要的。这项工作经常靠KEY—OTHERS触发器来完成，一般地，该触发器定义在FORM级上。

例如，在CT_FORM中，在FORM级上定义触发器KEY-OTHERS来屏蔽其它未使用的功能键，内容如下：

```
#EXEMACRO NULL;
```

其中，NULL功能码为不执行任何操作。

4 DDTS特点

本系统具有如下特点：

- 全部以菜单及全屏幕窗口显示方式提供给用户，易理解，易接受。
- 操作简单，用户只需按几个功能键便可完成操作。
- 自动提供帮助信息。
- 采用了功能键屏蔽技术，减少了操作者的误操作。

一种新的SQL * Forms菜单设计

武汉测绘科技大学 李睿葆

(武汉武昌珞瑜路39号 邮编430070)

摘要 本文使用SQL * Forms触发器设计一个采用光标键选择菜单项的菜单系统,并详细讨论了其设计原理和过程。

关键词 ORACLE SQL * Forms 数据库 菜单

1 引言

SQL * Forms是ORACLE数据库管理系统的应用生成器,它提供的触发器特别便于快速设计系统菜单,SQL * Forms设计菜单的过程是:把菜单块设计成控制块,它不与任何基本相对应,在该块的屏幕格式中列出所有可能的菜单选择项,然后设计一个选择字段,用户在该字段中输入选择项对应的数字进行选择,实际上是在选择字段上设计一个KEY-NXTFLD触发器,它由一个CASE语句构成,实现判定选择和转向功能。这种菜单设计虽然直观但形式过于简单,用户操作比较死板。我们下面设计的菜单是用户

用上下光标键选择到需要选择的菜单项,再按[Enter]键便作选择,后者既直观,也比较生动。

2 菜单设计

我们以一个实例讨论这种菜单设计的方法,假设菜单块为B₁块,它有4个选择项,各选择项名及转向块如表1所列。B₁块的设计如下:

2.1 B₁块的屏幕格式设计如图1所示。

图中汉字和方框是屏幕提示符,括号里的字符为字段名,SEL字段为不可显示字段。所有的OPTION字段和对应的MENU字段连接在一起放在同一行中,OPTION字段为CHAR类型且宽度为2,它们排成一列,

e.可移植性好。本系统虽是在IBM PC/XT机上开发的,但是,它是对ORACLE数据字典本身的一种加强,又因为ORACLE本身具有硬件独立性,因此,它适用于任何安装ORACLE数据库的机器上运行。

f.易于扩充。本系统采用的是分级菜单调用形式,若要进行扩充,只须设计一个完成要扩充功能的FORM程序。然后在主菜单中的CASE语句中增加一项调用命令,调用之即可。

5 结论

本文介绍的DDTS,给最终用户,应用程序开发人员和数据库管理员(DBA)使用

ORACLE数据字典提供了一条简捷的途径,是对ORACLE数据字典本身的一种加强,它具有操作方便,用户界面友好,通用性强等特点,在一定程度上满足了我们的需要,但还有待于进一步加强与完善其功能。

参考文献

- 1 萨师煊,王珊.数据库系统概论,高等教育出版社,1983
- 2 杨晓峰.ORACLE 数据库第四代语言工具SQL * FORMS 应用与研究.东北工学院硕士论文,1989
- 3 ORACLE Database Administrator's Guide, ORACLE Part No.3601
- 4 SQL * FORMS Designer's Reference, ORACLE Part No.3304

MENU字段也为CHAR类型,长度根据需要而定。该块对应的基表为空。

表1. B₁块的选择项及对应的转向块块名

序号	选择项名	对应的转向块块名
0	返回	B ₀
1	数据输入	B ₂
2	数据维护	B ₃
3	数据统计	B ₄

-----菜单块-----	
(option1)	(menu1)
(option2)	(menu2)
(option3)	(menu3)
(option0)	(menu0)
(sel)	

图1 B₁块的屏幕格式

2.2 在B₁块上设计如下块级触发器。

触发器名: PRE-BLOCK

步1: #EXEMACRO EXETRG INIT;

属性: • Abort trigger when step fails

步2: SELECT '数据输入', '数据维护', '数据统计', '返回'

INTO B₁.MENU1, B₁.MENU2, B₁.MENU3, B₁.MENU0

FROM DUAL

属性: • Abort trigger when step fails

步3: SELECT '→', '1'

INTO B₁.OPTION1, B₁.SEL
FROM DUAL

属性: • Abort trigger when step fails

触发器名: INIT

步1: SELECT ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '

INTO B₁.OPTION1, B₁.OPTION2, B₁.OPTION3, B₁.OPTION0
FROM DUAL

属性: • Abort trigger when step fails

触发器名: KEY-NXTREC

步1: SELECT MOD((: B₁.SEL+1), 4)

INTO B₁.SEL
FROM DUAL

属性: • Abort trigger when step fails

步2: #EXEMACRO EXETRG INIT;

属性: • Abort trigger when step fails

步3: #EXEMACRO EXETRG CHOOSE;

属性: • Abort trigger when step fails

触发器名: CHOOSE

步1: #EXEMACRO CASE SEL IS
WHEN '1' THEN EXETRG CHOOSE1;
WHEN '2' THEN EXETRG CHOOSE2;
WHEN '3' THEN EXETRG CHOOSE3;
WHEN '0' THEN EXETRG CHOOSE0;
END CASE;

属性: • Abort trigger when step fails

触发器名: CHOOSE1

步1: #COPY '→': B₁.OPTION1

属性: • Abort trigger when step fails

触发器名: CHOOSE2

步1: #COPY '→': B₁.OPTION2

属性: • Abort trigger when step fails

触发器名: CHOOSE3

步1: #COPY '→': B₁.OPTION3

属性: • Abort trigger when step fails

触发器名: CHOOSE0

步1: #COPY '→': B₁.OPTION0

属性: • Abort trigger when step fails

触发器名: KEY-PRVREC

步1: SELECT MOD((: B₁.SEL-1+4), 4)

INTO B₁.SEL
FROM DUAL

属性: • Abort trigger when step fails

步2: #EXEMACRO EXETRG INIT;

属性: • Abort trigger when step fails

步3: #EXEMACRO EXETRG CHOOSE;

ORACLE和Rdb/VMS的比较评估

机电部第二十八研究所 孙海波 周菲

(江苏南京1406信箱42分箱 邮码 210014)

摘要 在管理信息系统的开发过程中,数据库的选型是至关重要的。本文从技术性能、软硬件平台、国内使用情况、汉化工作和经费等五个主要的方面,阐述了国内中小型计算机上较流行的ORACLE和Rdb/VMS数据库的优劣。

1 引言

目前,国内在小型机上运行的数据库主要有两种,它们是 ORACLE 公司推出的 ORACLE 关系 DBMS 和 DEC 推出的 Rdb/VMS。这两种数据库究竟谁优谁劣一直是困扰国内开发单位和用户的一个重要问题,我们在辽河石油勘探局物资信息管理系统(简称 LHWGX)的开发过程中同样遇到了这一问题。为此,我们进行了深入的调查研究,发现简单评论谁优谁劣是不公正的,其实它们各有所长,从不同的角度看问题可以得到不同的结论。

2 ORACLE和Rdb简介

ORACLE公司创建于1977年,随后便推

出了ORACLE系统,它既是第一个关系型DBMS,又是第一个基于IBM的SQL数据库语言的产品。ORACLE产品可适用信息系统开发策略的每一环节,在ORACLE关系DBMS基础上,ORACLE公司为系统建立者、应用开发者和最终用户提供了一系列具有高生产率的第四代应用开发工具。这些工具包括:SQL * Forms,SQL * Menu,SQL * Plus,SQL * Gragh,SQL * Report-Writer和CASE * Dictionary,开发者使用这些工具可以建立复杂的事务处理、报表和菜单系统,而不需要使用高级语言编程。

VAX计算机上的Rdb/VMS是近年来美国DEC公司推出的一个关系型数据库管理系统,其性能不断得到大幅度的提高。DEC公司为了方便用户,提供了许多与之相配套

属性: * Abort trigger when step fails
触发器名: KEY-NXTFLD

```

步1: #EXEMACRO CASE SEL IS
WHEN '1' THEN GOBLK B2;
WHEN '2' THEN GOBLK B3;
WHEN '3' THEN GOBLK B4;
WHEN '0' THEN GOBLK B0;
      END CASE;

```

属性: * **Abort** trigger when step fails

用户操作及触发器的执行过程为：当用户操作进入B₁块时，激发PRE-BLOCK触发器的执行，它把各OPTION字段清空，把选择项送入相应的MENU字段，并把选择开关置于MENU₁字段处，当用户使用上下光标键进行选择移动时，这些按键分别激发

KEY-NXTREC和KEY-PRVREC触发器的执行，它们重新设置选择开关和SEL字段值，当用户按[Enter]键时表示选择了相应的选择项，此时激发KEY-NXTFLD触发器的执行，它用于功能转向。

这些触发器构成一个整体实现用户菜单项选择和响应。

3 结语

本文设计的菜单直观，用户操作方便，系统响应速度快，可扩充成通用菜单系统。

参考文献

1 李春葆, 马玉枫. SQL * Forms触发器设计技术计算机科学技术与应用, 1992.1

的开发应用工具,例如:DEC Forms,ACMS, DTR和Rally等,并且从3.0版开始支持SQL。迄今为止,DEC在全世界已颁发了超过15000个Rdb的许可证,VAX计算机上的Rdb用户比任何其它种类的关系数据库都多。

3 ORACLE和Rdb的比较

选择任何软件产品不仅要考虑该产品技术性能,而且要考虑经费、环境等诸方面的因素。一般地说,没有任何产品可以做到十全十美,只能权衡各方面因素后折中做出选择。影响MIS软件选型的主要因素主要有五个,即:

- 技术性能;
- 软硬件平台;
- 汉化问题;
- 国内的使用情况;
- 经费。

依据上述五个方面我们对ORACLE和Rdb进行了简单的评估。

3.1 技术性能

3.1.1 处理能力

根据美国权威的数据库测试机构(Codd and Data, Inc) 1989年的结果:在VAX6360计算机上,ORACLE 6.0版的TPS为66.0, VAX Rdb3.0版的TPS为30.6,这里的指标TPS是数据库测试中最重要的,也是最合理的指标。一般地讲,可以认为前者的处理能力为后者的两倍。另外,据我们调研,国内许多同时使用过ORACLE和Rdb用户反映,ORACLE的速度明显快于Rdb。

3.1.2 开发效率

在开发MIS的过程中,最令开发者头痛的是大量的屏幕格式和各种各样的报表。使用第三代语言编制如此多的屏幕格式和报表是非常困难的,主要的困难有两个:其一是编程效率低,其二是变动不方便,适应性差。

在大中型的MIS的开发中,一般都采用原型法(Prototyping),其方法是首先快

速产生一个基本模型。用户在试用这一基本模型的同时不断反馈修改意见,开发者则据此不断修改完善,直至用户满意为止。原型法要求开发时要能快速方便地产生一个模型,模型的结构为开放性的,可以方便地对模型进行修改、删除和增加。使用第三代语言在短时间内编制如此良好结构的程序是非常困难的,几乎是不可能的。

ORACLE为了适应这种需求,研制推出了许多第四代的应用工具。这种工具为ORACLE专用,与其数据库有紧密的联系,开发者仅要给出所需的字段和条件选择就会方便得到屏幕格式或报表,所有的数据均可重新编辑和定义,如果需要,还可以调用第三代语言编制的例程。

ORACLE提供的主要工具有:SQL * Forms, SQL * Menu和SQL * Report-Write。使用SQL * Forms可迅速有效地建立基于屏幕式的事务处理应用,只要简单地交互式给出要求,SQL * Forms便可以把你的命令与来自ORACLE数据字典的信息结合起来产生所需的应用,而不需要编制程序。SQL * Forms独特的非过程化开发方法有效地促进了应用原型法的开发工作,使开发者可不断地完善所建立的应用。使用SQL * Menu可以为所有模块设计动态菜单接口,这些模块可以用第三代语言编写的,也可以是用第四代应用开发工具开发的,并且可以给菜单选择项授保护级别。SQL * Report-Writer可以方便地编制各种报表,进行简单的算术运算,开发者只要使用简单的命令就可以把数据库中的数据填入报表中,并且可以交互式地修改和填写报表栏目。

DEC也提供了第四代应用开发工具。例如,DEC Forms, ACMS等,但与ORACLE的产品相比有两个不足:其一是品种单一。除去上面提到的屏幕格式开发工具DEC Forms外,还没有推出有关报表生成的任何第四代开发工具。其二是功能较弱,使用不方便。由于DEC的第四代应用开发工具是针

对整个DEC信息结构的,而不是Rdb数据库专用的,必须借助于第三代语言编程的方式建立与数据库的联系,可以认为这还不是一个完整的第四代应用开发工具。由于受第三代语言的制约,很难在原型法开发中担当重任,开发效率要低得多。

3.1.3 使用效率

借助于ORACLE的第四代应用开发工具。最终用户经过一定的培训就可以进行工作,选择需要的字段名和条件表达式,即可在屏幕得到简单的、组织好的数据,也可以得到实用的报表。现行Rdb支持的软件产品不可能被最终用户所利用,尤其是在国内计算机应用水平较低的实际情况,更是不可能实现。

3.2 软硬件平台

ORACLE的硬件平台非常丰富,它可以在IBM, DEC, SUN, HP, Apple等许多世界著名的计算机公司的机器上运行,基本上包括了全世界大多数种类的计算机。这些计算机的种类非常复杂,差别很大,其中有大型机、中型机和小型机,也有各种各样的微机和工作站,能在如此众多的计算机上运行也许只有ORACLE一家了。

从软件的角度上,ORACLE支持IBM公司的MVS, VM, VSE, PC-DOS和OS/2等,DEC公司的VMS, ULTRIX等,还有许多国内外流行的操作系统,例如:MS-DOS, UNIX, XENIX等。另外,ORACLE完全实现了工业标准关系数据库语言—SQL。ORACLE预编译方式支持C, Ada, COBOL, FORTRAN, Pascal和PL/I等多种高级语言。

ORACLE的分布式结构使其数据和应用能够驻留在多台计算机上,这些计算机可以是大中型机,也可以是微型计算机,相互之间的通信是透明的,通过网络可以把不同的运行ORACLE的计算机连起来,实现各节点之间方便迅速的互访。在MIS中,经常需要异种机之间互相传递数据,ORACLE对

复杂环境的适应性,使这种传递变得更为简单。

Rdb的软硬件平台都比较单一,其只能在DEC的硬件的VMS操作系统下运行。虽然DEC已经做了许多接口工作,使Rdb中的数据可以传递到异种机上,但这种方式层次较低,且使用不方便。

3.3 汉化问题

由于MIS中有大量的汉字输入输出,任何软件产品必须具有输入输出汉字和屏幕编辑汉字的功能。多数软件产品只要其不屏蔽字节高位,且使用的特殊代码不与汉字代码冲突,则只要使用汉字终端,汉字的输入输出是不会有困难的。实现汉字的屏面编辑功能相对要困难得多,厂商必须针对汉字的特殊情况做一定的开发工作,要保证可以一个汉字一个汉字地进行删除或插入工作。汉化问题在中国是非常重要的,不支持汉字的软件产品几乎无法在国内使用。尤其是在MIS的开发应用中更是如此。

RDB汉化的版本为3.0版,其最新的4.0版还没有汉化,如果配合汉化的DECForms使用,则完全可以支持汉字的输入输出和编辑功能,在MIS的开发中不会有困难。

ORACLE 5.0版有汉化版本,而现在的最新的6.0版没有完全支持汉字。通过SQL * Forms输入输出汉字尚可行,但插入或删除汉字就有困难了。据ORACLE公司透露,其6.0版在5.0版基础上重写了70%的代码,其效率是5.0版的5~10倍,也是ORACLE推荐的版本。另外,ORACLE公司正在积极开发最新的多国文字版(National Language Support),这个版本可以完全支持汉字,估计近期有望推出。

3.4 国内的使用情况

ORACLE商品化比较早,在国外市场上占有较大的份额。在国内各种微机上安装的数量也不少,但在中小型机上的用户相对较少,其成功的用户也相应较少。Rdb在中国市场的份额很大,如铁路系统、海关系统等

均为Rdb的市场,成熟的用户很多,使用经验比较丰富,开发应用成果较多。

3.5 经费

ORACLE公司是纯粹的软件公司,采用软件的License策略,其收费依据有三个:

a. 计算机的档次

同一套软件在不同档次的计算机上的收费标准是不同的,而且差距悬殊。例如,在一台VAX3100上安装数据库软件的费用约为1.3万美元,而在一台VAX4200上安装一套数据库软件则需要12万美元。

b. 用户数量

同一档次的计算机用户数量不同费用也不同。例如,40用户的VAX4200约需12万美元,20用户的VAX4200大约只有其一半,约合6万美元。

c. 与计算机台数成正比

系统中计算机的台数越多,费用也就越高,即总费用等于台数乘以单价。虽然随着数量的增大会得到相应的折扣,但数额相对较小。

DEC的Rdb的销售方式与ORACLE在前两点上是有相同之处的,但在第三点上有明显的差别。DEC允许用户在其系统中有一台计算机购买Rdb的开发权,而在其它计算机上仅花很少一点钱购买运行权即可使整个系统投入使用,这种方式使大系统用户得到巨大的好处。

在LHWGX的总体方案中,我们推荐了

两种系统配置。方案一选择了两台VAX4200和三台VAX3100通过DECnet IV网络连成一个大系统。方案二选择了两台DECSystem5500和三台DECSystem 5100通过DECnet IV网络连成一个大系统。两种方案使用ORACLE和Rdb数据库的费用如表1所示,Rdb费用仅为ORACLE费用的五分之一左右。

表1 ORACLE与RDB使用费用比较

DBMS 机型	ORACLE	RDB/VMS
方案一	\$25万/150万元	\$5万/30万元
方案二	\$19万/114万元	\$5万/30万元

4 结论

由上面比较我们可以得出这样的结论:在技术性能和软硬件平台这两个方面,ORACLE要明显强于Rdb;而在国内使用情况和经费两个方面Rdb要明显优于ORACLE;在汉化问题上,使用Rdb比较稳妥,而使用ORACLE在近期内有一定的风险。

从技术角度上看,选用ORACLE是毫无疑问的。但从系统开发的实际情况出发,经费往往是最重要的,ORACLE的费用确实太高,国内用户难以承受。因此,应在技术和经费上折中考虑,做出符合实际的选择。

用C语言构造通用的下拉式菜单程序

河海大学计算机科学系 许卓明

(南京西康路1号 邮编210024)

南京交通银行计算中心 角 钢

摘要 本文以C语言为例提出了通过建立下拉式菜单框架与驱动函数,以及通过用基于BIOS的程序设计方法来构造一组菜单支持函数,可以设计出通用的下拉式菜单程序,并给出了具体实现方法。

关键字 下拉式菜单 视屏适配器 基于BIOS的程序设计 可移植性

1 问题提出

当使用下拉式菜单(pull_down menu)的程序运行时,能给人以清晰的程序处理流程和生动的人机界面,因此,倍受用户青睐。

软件开发常伴有这种体会:用户往往以人机界面来衡量一个应用程序的质量;而对于开发者来说,人机界面的设计往往占有了他很多的时间与精力。

那么,掌握一种通用的设计技术、构造通用程序,便能既节省时间与精力、又能设计出令用户满意的程序。在此,笔者试图以C语言这种常用的软件开发工具为例,提出通过建立菜单框架与驱动函数、构造一组通用的菜单支持函数等方法来设计出通用的、可移植的下拉式菜单程序。

2 建立菜单框架和菜单系统驱动函数

一个应用程序包含许多菜单,而这些菜单往往构成一个树型系统。为了使菜单支持例程(C函数)能区分各菜单,可按一定顺序(如图1所示)给各个菜单编上号:

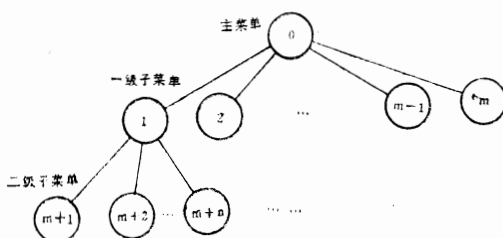


图1 树型菜单系统

另外,各个菜单又均有其自己特定的屏幕坐标参照系和特征值,因此,可定义一个统一的菜单框架(frame)来描述这些特性,以便提供给菜单支持例程。显然,这个菜单框架用全程的结构数组来表示最为合适:

```
struct menu_frame
{
    int active; /* 菜单激活标志 */
    int sx,sy,ex,ey; /* 占用屏幕的起止行列 */
    char * title; /* 菜单标题 */
    char ** menu; /* 菜单选项正文 */
    char * hotkeys; /* 选项热键 */
    int count; /* 选项个数 */
    int ncolor,rcolor; /* 正显及反显色 */
    unsigned char * p; /* 指向保存覆盖屏幕信息的空间 */
} f[MAX_F_N]; /* MAX_F_N为最大菜单号 */
```

对于一个特定应用程序的菜单系统,在激活整个菜单系统之前,需通过多次调用一个支持例程为所有菜单的框架赋值,其中激

活标志初始化为0。然后,通过一特定的驱动器来驱动整个菜单系统。例如,假设activate(n)函数用于激活第n号菜单,并返

回用户的选择响应值或当用户要求退出此菜单时返回-1,则下面的C函数就是一个示意的菜单系统驱动器:

```
void pdmenu_driver()
{int c0, c1, ...;
  while( (c0=activate(0)) != -1) /* 激活0号菜单 */
  {switch( c0)
    {case 1: while( (c1=activate(1)) != -1) /* 激活1号菜单 */
      {switch( c1)
        {case 1: /* 激活1号菜单的第一子菜单 */; ...; break;
         case 2: /* 激活1号菜单的第二子菜单 */; ...; break;
          :
         case n: /* 激活1号菜单的第n子菜单 */; ...;
        }
      }
      restore_video_info(1); /* 恢复1号菜单的覆盖屏幕 */
      break;
    case 2: /* 激活2号菜单 */; /* 恢复2号菜单的覆盖屏幕 */; break;
    :
    case m: /* 激活m号菜单 */; /* 恢复m号菜单的覆盖屏幕 */;
  }
  restore_video_info(0); /* 恢复0号菜单的覆盖屏幕 */
}
```

3 基于BIOS调用的视屏控制技术

IBM系列的视屏适配器一般工作于文本模式,这与大多数应用程序的需要是一致的。目前常用的CGA、EGA及VGA等,在DOS启动后的缺省模式为3(即80列×25行彩色文本方式),对于象MDA等单显,其缺省模式是7(即80列×25行文本方式)。因此,本文假设适配器工作于以上80列×25行文本方式,且假设显示RAM的活动显示页为通常的第0页。

要实现以一定的颜色属性显示字符,以及保存与恢复屏幕,必须对视屏进行直接控制。一般有以下两种实现方法:其一是:直

接存取显示寄存器及RAM(即直接程序设计)。用这种方法设计的程序可获得极快的显示速度,但由于其硬件依赖性,开发的程序可移植性差。特别是在汉字系统下,要确定显示RAM的地址是很困难的。其二是:通过调用BIOS 10H中断(即基于BIOS的程序设计)。各种常用的适配器不论其是否自带BIOS,还是使用ROM BIOS,功能号为0~F的10H中断调用是兼容的,而且,由于BIOS对硬件物理特性的屏蔽,故有利于开发可移植的通用性程序。

Microsoft C及Turbo C是目前常用的程序开发工具,两者均提供了INT86()函数,可用此实现10H中断调用。表1列出了笔者开发的一组菜单通用支持函数:

表1

菜单支持函数	函数功能简述	使用的功能号及支持函数
set_video_mode(m)	设置视屏适配器工作模式m	0H号(设置显示模式)
get_video_mode()	返回视屏适配器当前工作模式	0H号(读当前显示模式)
set_cursor(x, y)	光标定位于x行y列	2H号(光标定位)
save_video_info(n)	保存n号菜单的覆盖屏幕信息	8H号(读光标处字符及属性)
restore_video_info(n)	恢复n号菜单的覆盖屏幕信息	9H号(在光标处写字符及属性)
disp_ch(x, y, ch, color)	在x行y列显示字符	9H号(在光标处写字符及属性)
disp_st(x, y, st, color)	在x行y列显示字符串	disp_ch()
ccls(x1, y1, x2, y2, bkc)	在矩形区内以bkc为背景色着色	disp_ch()
draw_border(n)	画n号菜单的边框	disp_ch(), disp_st()
display_menu(n)	显示n号菜单	draw_border(), disp_st()等
get_user_resp(n)	接受并返回n号菜单的用户响应	disp_st()
activate(n)	显示并激活n号菜单,然后返回用户响应	save_video_info(), display_menu(), get_user_resp()

限于篇幅,在此对几个关键的支持函数,给出其示意程序。

```

int activate(n)
int n;
{int m; unsigned char *p;
 m=get_video_mode();
 if((m!=2)&&(m!=3)&&(m!=7))
 {printf(" video must be in 80*25 text mode"); exit(1);}
 if(!f[n].active){
  p=(unsigned char*)malloc(2*(f[n].ex-f[n].sx+1)*(f[n].ey-f[n].sy+1));
  if(!p){printf(" stack space overflow! \n"); exit(1);}
  f[n].p=p; /*申请空间*/
  save_video_info(n); /*保存菜单的覆盖屏幕信息*/
  f[n].active=1; /*置活动标志*/
  display_menu(n); /*显示菜单*/return get_user_resp(n); /*返回用户响应*/
 }
}

```

```

int get_user_resp(n)
int n;
{union inkey {char ch[2]; int i; } c; int ac=0, x, y, kc;
  x=f[n].sx+3, y=f[n].sy+2; /* ('x, y) 为选项正文起始行列 */
  disp_st(x, y, f[n].menu[0], f[n].rcolor); /* 反显 */
  for(;;)
  {while(! bioskey(1)); /* 等待用户击键 */
    c.i=bioskey(0); /* 读键的16位扫描码 */
    if(c.ch[0]) /* 是常规键 */
    {kc=is_hotkey(f[n].hotkeys, tolower(c.ch[0]));
      if(kc) /* 是热键 */
      {disp_st(x+ac, y, f[n].menu[ac], f[n].ncolor);
        /* 正显 */ ac=kc-1;
      }
      else switch(c.ch[0]) /* 是ENTER, ESC键 */
      {case '\r': return ac+1; /* 返回选项序号 */
        case 27: return -1;
      }
    }
    else switch(c.ch[1]) /* 是上下箭头键 */
    {case 72: disp_st(x+ac, y, f[n].menu[ac], f[n].ncolor);
      /* 正显 */ ac--; break;
      case 80: disp_st(x+ac, y, f[n].menu[ac], f[n].ncolor);
      /* 正显 */ ac++; break;
    }

    if(ac==f[n].count) ac=0;
    if(ac<0) ac=f[n].count-1;
    disp_st(x+ac, y, f[n].menu[ac], f[n].rcolor); /* 反显 */
  }
}

void save_video_info(n)
int n;
{unsigned int *buf_ptr;
  union REGS r; register int i, j;
  buf_ptr=(unsigned int *)f[n].p;
  for(j=f[n].sy; j<=f[n].ey; j++)
  for(i=f[n].sx; i<=f[n].ex; i++)
  {set_cursor(i, j); /* 光标定位 */
    r.h.ah=8; /* 写字符功能 */
    r.h.bh=0; /* 活动显示页 */
    *buf_ptr++=int86(0x10, &r, &r);
    putchar(' '); /* 清屏 */
  }
}

void restore_video_info(n)
int n;
{unsigned char *buf_ptr;
  union REGS r; register int i, j;
  buf_ptr=(unsigned char *)f[n].p;
  for(j=f[n].sy; j<=f[n].ey; j++)
  for(i=f[n].sx; i<=f[n].ex; i++)
  {set_cursor(i, j); /* 光标定位 */
    r.h.ah=9; /* 写字符功能 */
    r.h.bh=0; /* 活动显示页 */
    r.x.cx=1; /* 写字符次数 */
  }
}

```

介绍一种LR(K)状态集和分析表的构造方法

南开大学计算机与系统科学系 孙桂茹

(邮编 300071)

摘要 LR分析已被广泛地应用于编译程序的构造中,本文介绍一种新方法构造LR(K)状态集及分析表(特别是当K大于等于1),并在此基础上给出了利用分析表进行语法分析的算法。

关键词 LR(K)文法 分析表 项目集

1 引言

LR分析是由Knuuth引入的,从而它被广泛地应用在编译程序中。本文介绍一种新型的LR(K)分析表的构造,文中称为简化表。此表的构造方法适于K大于1的情况,并在文中给出了用此表进行归约分析的过程。

1.1 术语和基本定义

为了本文的叙述,把用到的基本术语和定义回顾如下:

a.上下文无关文法采用四元式定义 $G = (N, T, P, S')$,其中N, T, P分别表示为非终极符号、终极符号、产生式的集合, S' 为开始符。

b.*和+分别表示自反传递闭包和传递闭包。 \Rightarrow 和 \Rightarrow_{rm} 表示推导和规范推导。

c. ϕ 表示空集, ϵ 表示空符号串, a, b, c 表示终极符号, A, B, C, \dots, S 表示非终极符号, X, Y, Z 表示 V (V 表示 $N \cup T$) 中符号, u, v, \dots, z 表示 T^* 中串, $\alpha, \beta, \gamma, \dots$ 表示 V^* 中串。

d. $T^*K = \{u \in T^* \mid |u| \leq K\}$

e.假定扩展文法是由原文法再加入开始产生式 $S' \rightarrow S\#$ 组成。其中S为原文法开始符, S' 和 $\#$ 不可出现在其它产生式中。

f. $FIRST_K(\alpha) = \{\omega \in T^*K \mid \alpha \Rightarrow \cdot \omega u, | \omega | = K \text{ 或 } | \omega | < K \text{ 时 } u = \epsilon\}$

g. $EFF_K(\alpha) = \{\omega \mid \omega \in FIRST_K(\alpha) \text{ 且 } \alpha \Rightarrow^* r_m \beta \Rightarrow r_m \omega Y \text{ 而 } \beta \neq A \omega Y \forall A \in N\}$

1.2 V^* 中符号串的运算

由于在某些文法中,有些非终极符号可以生成空串,为此我们定义 V^* 中符号串的运

000 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040

```
r.h.al = *buf_ptr++; /* 字符 */
r.h.bl = *buf_ptr++; /* 属性 */
int86(0x10, &r, &r);
f[n].active = 0; /* 置非活动标志 */
free(f[n].p) /* 释放空间 */
}
```

4 汉字显示问题

要在菜单中显示汉字,必须让应用程序运行于汉字操作系统下。虽然汉字系统往往修改了原BIOS的有关显示管理模块,但大多数汉字系统已在BIOS级上解决了汉字显示问题,因此用本文介绍的方法编制的C程序只要稍加注意即能在汉字系统下正常运行,尤其是在采用文本显示方式的联想汉卡下更无问题。笔者开发的程序已在联想汉卡及

UCDOS 2.01下正常运行。在有的汉字系统下,如2.13H是采用图形显示方式的,程序必须作适当调整,然而,基本设计方法不变。

参考文献

- 1 Jack purdum, C Programming Guide, Que Corporation, Indianapolis, 1983
- 2 M.I. Bolisky, The C Programmer, s Handbook, AT&T Bell Lab, prentice hall, 1985
- 3 张福炎等编著,微型计算机原理与应用(续二)——图形显示器及其程序设计,南京大学出版社, 1990
- 4 尹彦芝著, C语言常用算法与子程序, 北京科海培训中心, 1991
- 5 张怀莲编, 宏汇编语言程序设计, 电子工业出版社, 1987

算。

1.2.1 定义 已知G, 对V中每个X,

$$\text{WEIGHT}(X) = \begin{cases} \min \{ |\omega| \mid X \Rightarrow^+ \omega, \\ \omega \neq \varepsilon \} \\ 1 \quad x \in T \end{cases}$$

对于V*中符号串 $\alpha = X_1 \cdots X_n$ ($X_i \in V$) 推广为

$$\text{WEIGHT}(\alpha) = \begin{cases} \sum_{i=1}^n \text{WEIGHT}(x_i) & \alpha \neq \varepsilon \\ 0 & \alpha = \varepsilon \end{cases}$$

1.2.2 定义 已知G, 对于V*中每个 α , 定义,

$$\text{FW}_K(\alpha) = \begin{cases} \alpha & \text{WEIGHT}(\alpha) < K \\ \beta & \text{WEIGHT}(\beta) \geq K \text{ 并且 } \beta \text{ 为 } \alpha \text{ 的最小前缀} \end{cases}$$

对于V*中符号串集合 $\{\alpha_1, \dots, \alpha_n\}$ 推广为

$$\text{FW}_K(\{\alpha_1, \dots, \alpha_n\}) = \{\text{FW}_K(\alpha_1)$$

$$\cup \text{FW}_K(\alpha_2) \cup \dots \cup \text{FW}_K(\alpha_n)\}$$

1.2.3 定义 已知G, 对于V中每个X, X的副本记为 $\text{DUP}(X)$, 定义 $\text{DUP}(X)$ 的符号串集合:

$$\text{DUP}(X) = \begin{cases} \{\varepsilon, x\} & x \Rightarrow^+ \cdot \varepsilon \\ \{X\} & \text{其它} \end{cases}$$

对于V*中符号串扩展定义为

$\text{DUP}(X\alpha) = \text{DUP}(X) \parallel \text{DUP}(\alpha)$, 其中 \parallel 为并操作。

1.2.4 定义 已知G, 对于V*中每个 α .

a. 如果 $\alpha = x\alpha'$ 则

$$\text{DUPEF}(\alpha) = \begin{cases} \text{DUP}(x\alpha'), & x \in T \\ \bigcup_{x \rightarrow \beta \in P} \text{DUPEF}(\beta \end{cases}$$

$\alpha')$, $X \in N$, $\text{EFF}_1(X) \neq \emptyset$, 且 $\beta \neq \varepsilon$,

b. $\text{DUPEF}(\alpha) = \emptyset$, 对于其它情况。

1.2.5 定义 已知G和V*中串 β , $\beta = X_1 \cdots X_n$, $X_i \in N$, 定义 $\text{FIRST}_K^+(\beta) = \{\omega \mid \omega \in \text{FIRST}_K(\beta), \text{ 且 } X_i \neq \varepsilon, \forall X_i \text{ 为 } \beta \text{ 中符号 } (i=1, 2, \dots, n)\}$

2 建立简化状态的集合 (FULL)

现在我们考虑两种类型的项目集: a. 正规项目集; b. 前看项目集。

2.1 正规项目集

正规项目集是状态集中最基本项目集,

它的组成方法规定为:

2.1.1 定义 已知G和 $K > 0$, 对于P中每个 $A \rightarrow \alpha\beta$, $[A \rightarrow \alpha \cdot \beta, \gamma]$ 是一LR(K)正规项目, 在此有 $\text{FW}_K(\gamma) = \gamma$ 。

有了正规项目, 我们进一步构造正规项目集的闭包。

2.1.2 定义 已知G和 $K > 0$, 对于每个正规项目集NS, 定义NS的闭包 ($\text{NCLOSURE}_K(\text{NS})$):

a. $\text{NS} \subseteq \text{NCLOSURE}_K(\text{NS})$

b. 如果 $[B \rightarrow \alpha \cdot A\beta, \gamma] \in \text{NCLOSURE}_K(\text{NS})$, 则有 $[A \rightarrow \cdot \omega, \delta] \in \text{NCLOSURE}_K(\text{NS})$, 其中 $A \rightarrow \omega \in P$, $\delta \in \text{FW}_K(\text{DUP}(\beta\gamma))$ 。

例如文法 $G_1 = (N, T, P, S')$ (已扩展), $K=2$, 其中 $N = \{S', S, A, B\}$, $T = \{a, b, c, \#\}$, P由下列产生式组成:

$$S' \rightarrow S\# \quad (1)$$

$$S \rightarrow AabA \quad (2)$$

$$A \rightarrow aB \quad (3)$$

$$A \rightarrow \varepsilon \quad (4)$$

$$B \rightarrow c \quad (5)$$

$$\text{NCLOSURE}_2(\{[S' \rightarrow \cdot S\#, \varepsilon]\}) = \{[S' \rightarrow \cdot S\#, \varepsilon], [S \rightarrow \cdot AabA, \#], [A \rightarrow \cdot aB, ab], [A \rightarrow \cdot, ab]\}$$

下面给出正规项目集的转换函数。

2.1.3 定义 已知G和 $K > 0$, 对于每个正规集NS和V中每个符号X, 定义:

$$\text{NGOTO}(\text{NS}, X) = \text{NCLOSURE}_K(\{[A \rightarrow \alpha x \cdot \beta, \gamma] \mid [A \rightarrow \alpha \cdot x\beta, \gamma] \in \text{NS}\})$$

如上例文法 G_1 中 $\text{NGOTO}(\text{NS}, S) = \{[S' \rightarrow S \cdot \#, \varepsilon]\}$

在用上述方法构造的项目集中, 我们把满足下述条件的项目集称为NS的不安定项目集 (inadequate), 否则称为安定项目集 (adequate)。

2.1.4 定义 已知G和 $K > 0$, 一个正规项目集NS若它包含 $[A \rightarrow \beta_1 \cdot \beta_2, \gamma]$ 和 $[B \rightarrow \alpha \cdot \delta]$ 两个项目。其中 $\text{FIRST}_K^+(\text{DUPEF}(\beta_2)) \neq \emptyset$ 或 $\beta_2 = \varepsilon$, 则称NS为不安定项目集。

如对文法 G_1 所求的项目集: $\{[S' \rightarrow \cdot S\#, \varepsilon], [S \rightarrow \cdot AabA, \#], [A \rightarrow \cdot aB, ab],$

$[A \rightarrow \cdot, ab]$ ($k=2$) 满足上述条件, 我们称此项目集为不适定正规项目集。

2.2 前看项目集

在LR分析中, 为了解决分析动作的单一性, 往往有时需要朝前看K个符号, 对于这种情况, 我们定义如下:

2.2.1 定义 已知文法G和 $K>0$, 对P中每个 $A \rightarrow \alpha\beta$, 定义 $[A \rightarrow \alpha \cdot u \cdot \delta, act]$ 为前看项目, 其中 $|u| \leq K$, 且 act 为下述动作之一: $\{shift, ace\} \cup \{reduce \text{ 用 } A \rightarrow \gamma \text{ 归约} \mid A \rightarrow \gamma \in P, A \neq S'\}$ 。

一个前看项目包含两个点, 它的最右的点称为前看点, V^* 中符号串 δ 称为前看点的前看内容。

类似地, 我们定义前看项目集的闭包:

2.2.2 定义 已知G和 $K>0$, LS为前看项目集, LS的闭包($LCLOSURE_K(LS)$)为:

a. $LS \subseteq LCLOSURE_K(LS)$

b. 如果 $[A \rightarrow \alpha \cdot u \cdot B\gamma, act] \in LCLOSURE_K(LS)$ 且 $|u| = i \leq K$, 则 $[A \rightarrow \alpha \cdot u \cdot \beta, act] \in LCLOSURE_K(LS)$, $\forall B \rightarrow \eta \in P, \eta \neq \epsilon, \forall \beta \in FW_{K-i}(DUP(\eta\gamma))$

如上例文法 G_1 中, $K=2$ 有:

$LCLOSURE_2(\{[A \rightarrow \cdot a \cdot B, shift], [A \rightarrow \cdot a \cdot b, reduce-3]\}) = \{[A \rightarrow \cdot a \cdot B, shift], [A \rightarrow \cdot a \cdot b, reduce-3], [A \rightarrow \cdot a \cdot c, shift]\}$

我们也来定义LS的转换函数。

2.2.3 定义 已知G和 $K>0$, 对于每个LS和T中每个a, 则 $LGOTO(LS, a) = LCLOSURE_K(\{[A \rightarrow \alpha \cdot ua \cdot \gamma, act] \mid [A \rightarrow \alpha \cdot u \cdot a\gamma, act] \in LS\})$

如上述LS和T中b有: $LGOTO(LS, b) = \{[A \rightarrow \cdot ab \cdot, reduce-3]\}$

对于 T^* 中串v, 扩展定义为:

$LGOTO(LS, v) = LGOTO(LGOTO(LS, a), v')$, 其中 $v = av', a \in T, v' \in T^*$

同样定义LS中的不适定情况。

2.2.4 定义 已知G和 $K>0$, 若LS中包含 $[A \rightarrow \beta_1 \cdot u \cdot \delta_1, act_1]$ 和 $[B \rightarrow \beta_2 \cdot u \cdot \delta_2, ac-$

$t_2, [其中 act_1 \neq act_2, 则LS为不适定LS项目集。$

如项目集 $\{[S' \rightarrow \cdot a \cdot B, Shift], [S \rightarrow \cdot a \cdot B, shift], [A \rightarrow \cdot a \cdot B, shift], [A \rightarrow \cdot a \cdot b, reduce-3], [S' \rightarrow \cdot a \cdot c, shift], [S \rightarrow \cdot a \cdot c, shift], [A \rightarrow \cdot a \cdot c, shift]\}$ 为不适定前看项目集。

2.3 不适定正规项目集的转换

由上可知在正规项目集中, 若发现了本项目集为不适定情况 (即为本项目集中出现多个reduce, 或shift与reduce), 我们为LR分析正常进行, 就需对这种情况加工处理, 使它演变成前看项目集。

2.3.1 定义 已知G和 $K>0$, 并且NS为不适定正规项目集, 定义

$TRANSITION_K(NS) = LCLOSURE_K(V)$

其中 $V = \{[A \rightarrow \alpha \dots \xi, act] \mid [A \rightarrow \alpha \cdot \beta, \delta] \in NS, \text{ 且 } \xi \in v(\beta\delta) \neq \phi\}$

$v(\beta\delta) = \begin{cases} \{\delta\} & \beta = \epsilon \\ FW_K(DUP(\beta) \parallel \{\delta\}) & \text{其它} \end{cases}$

$act = \begin{cases} reduce & \text{用 } A \rightarrow \alpha \text{ 归约, } \beta = \epsilon \text{ 且 } A \neq S' \\ shift & \text{其它} \end{cases}$

如 $TRANSITION_K(\{[S' \rightarrow \cdot S\#, \epsilon], [S \rightarrow \cdot AabA, \#], [A \rightarrow \cdot aB, ab], [A \rightarrow \cdot, ab]\}) = \{([S' \rightarrow \cdot aB, shift], [S \rightarrow \cdot aB, shift], [A \rightarrow \cdot aB, shift], [A \rightarrow \cdot ab, reduce-3])\}$

下面定义函数运算。

2.3.2 定义 对于不适定正规项目集NS和 T^* 中符号串u, 定义

$\begin{cases} LS(NS, u) = LGOTO(TRANSITION(NS), u), & |u| > 0 \\ LS(NS, \epsilon) = TRANSITION(NS) \end{cases}$

应用上述函数来求NS (不适定的) 的项目集LOOKSET(NS), 算法:

设前看状态集为L

a. $L = LS(NS, \epsilon)$

b. 求出 $LS(NS, u)$, 检查各前看状态集是否有不适定的前看状态集, 若无 $L = L \cup LS(NS, u)$; 若有, 假定不适定前看项目集为

LS' , 这时有 $L = L \cup \{LS(NS, u)\}$, 当 $|u| < K$ 时, $u \neq u' \# \} \cup \{LGOTO(LS', a) \mid a \in T\}$

```

normal state. 0 inadequate
[S' → · S#, ε]
[S → · AabA, #]
[A → · aB, ab]
[A → ·, ab]
normal state. 1 adequate
[S' → S · #, ε]
normal state. 2 adequate
[S → A · abA, #]
normal state. 3 adequate
[A → a · B, ab]
[B → · c, ab]
normal state. 7 adequate
[S' → S# ·, ε]
normal state. 8 adequate
[S → Aa · bA, #]
normal state. 9 adequate
[A → aB ·, ab]
normal state. 10 adequate
[B → c ·, ab]
normal state. 11 inadequate
[S → Aab · A, #]
[A → · aB, #]
[A → ·, #]
normal state. 12 adequate
[S → AabA ·, #]
normal state. 13 adequate
[A → a · B, #]
[B → · c, #]
LOOKSET(NS11) = {LS14, LS15}
TRANSITION2
[S → · aB, shift]
[A → · aB, shift]
[A → · #, reduce-3]

```

注: 在求TRANSITION₂(NS₁₁)时, 在前看项目集中, 为简化表示略去了点前部分。

对于文法G, 它的状态集FULL是由两部分状态集组成, 一部分为NORMAL, 另一部分为LOOKSET(NS), 其中

c. LOOKSET(NS) = $L - \{LS(NS, \epsilon)\}$

下面以文法G₁为例构造出它的(K=2)全部项目集。

```

LOOKSET(NS0) = {LS4, LS5, LS6}
TRANSITION2
[S' → · aB, shift]
[S → · aB, shift]
[A → · aB, shift]
[A → · ab, reduce-3]
100k-state. 4 inadequate
[S' → a · B, shift]
[S → a · B, shift]
[A → a · B, shift]
[A → a · b, reduce-3]
[S' → a · c, shift]
[S → a · c, shift]
[A → a · c, shift]
100k-state. 5 adequate
[s' → ac ·, shift]
[s → ac ·, shift]
[A → ac ·, shift]
100k-state. 6 adequate
[A → ab ·, reduce-3]
100k-state. 14 adequate
[S → a · B, shift]
[A → a · B, shift]
[s → a · c, shift]
[A → a · c, shift]
100k-state. 15 adequate
[A → # ·, reduce-3]
normal state. 16 adequate
[A → aB ·, #]
normal state. 17 adequate
[B → c ·, #]

```

$NORMAL = \{NCLOSURE_K(\{S' \rightarrow \cdot S \#, \epsilon\}) \cup \{NGOTO(NS, X) \neq \phi \mid X \in V, NS \in NORMAL\}\}$

那么 $FULL = NORMAL \cup \{LOOKSET(NS) \mid NS \in NORMAL \text{ 且是不适定的}\}$

3 简化表的构造

简化表是一个函数对 (c, eg) 其中 c 是分析动作, 它是 $\{shift, acce, look\} \cup \{reduce\}$ 用 $A \rightarrow \beta \mid A \rightarrow \beta \in P, A \neq S'$ 中之一; eg 是状态转换函数, 是 V 到 $\{error\} \cup \{\text{简化表状态集}\}$ 上的映射。

下面给出构造简化表的算法。

设 $\tau \in FULL$ 的一个状态集, 用 S, A, L 分别表示 $shift, acce, look$ 的缩写, 用空表示 $error$ 。

3.1 如果 τ 是适定正规项目集, 则

a. $c=N$, 如果 τ 包含一个项目 $[A \rightarrow \beta \cdot]$, 且 $A \rightarrow \beta \in P$, 它的编号为 N 。

b. $c=A$, 如果 τ 包含项目 $[S' \rightarrow S \# \cdot \varepsilon]$

c. $c=S$, 其它

eg 函数构造:

a. $eg(x) = R_j$, 如果 $\tau_j = NGOTO(J, X) \neq \phi$

e. $eg(x) = \text{空}$, 其它

3.2 如果 τ 是一个不适定正规集, 或 J 是一个前看项目集 LS , 则

a. $c=L$, 如果 τ 是一个不适定项目集。

b. $c=act$, 如果 $\tau \in LS$ 是适定的, 那么 act 与 LS 中相联系项目 act 相同。

eg 函数构造

对于 NS :

a. $eg(A) = R_j$, 如果 $NS_j = NGOTO(NS, A)$ 且 $NS_j \neq \phi$

b. $eg(a) = R_j$, 如果 $LS_j = LS(NS, a) \neq \phi$

c. $eg(x) = \text{空}$, 其它

对于 LS :

a. $eg(a) = R_j$, 若 LS 是不适定的, 其中 $LS_j = LGOTO(LS, a) \neq \phi$

b. $eg(a) = R_j$, 如果 $LS = LS(NS, a\omega)$ 是适定的, 其中有 $NS_j = NGOTO(NS, a) \neq \phi$

c. $eg(x) = \text{空}$, 其它

利用上述算法, 构造出上例中简化如表1:

在简化表中, 我们把项目15和17分别统一到相应状态6和10。

表1

c		eg								
		S'	S	A	B	a	b	c	#	ε
R0	L			R1	R2		R4			
R1	S								R7	
R2	S						R8			
R3	S				R9				R10	
R4	L							R6	R5	
R5	S					R3				
R6	3									
R7	A									
R8	S								R11	
R9	2									
R10	4									
R11	L				R12	R14			R6	
R12	1									
R13	S					R16			R10	
R14	S							R13		
R16	2									

4 用简化表进行归约分析

我们用三元组来描述用简化表进行归约分析的过程。三元组的形式为: $(R_0 X_1 R_1 X_2 \dots X_n R_n, w, \pi)$ 其中 $R_0 X_1 R_1 X_2 \dots X_n R_n$ 为下推表的当前内容。顶元素为 R_n ; w 是输入串停止分析部分; π 是已识别的输入串所应用的产生式序列串。

假定 z 是 T^* 中符号串, 对 z 进行分析过程用算法描述如下:

三元组初始状态 (R_0, z, ε) ; 对于中间某一步三元组状态为 $(R_0 X_1 \dots R_n, aw, \pi)$, 下一步动作:

4.1 若 $C_n = S$ 且 $eg_n(a) = R_m$, 则三元组状态变成 $(R_0 X_1 \dots R_n a R_m, w, \pi)$; 若 $eg_n(a) = \text{空}$, 则分析停止。

4.2 若 $C_n = N$ 且 N 对应的产生式为 $A \rightarrow X_{i+1} \dots X_n$, $A \neq S'$, 且 $eg_i(A) = R_m$, 则三元

中西文兼容UNIX操作系统的一种汉字输入输出实现方法

国家科委管理学院科技统计室 尹竞

(武汉市武昌东湖路58号 邮编430077)

摘要 本文根据UNIX系统结构上的层次关系及UNIX文件系统特有的pipe功能,提出了一种通过在UNIX外层添加模块,来实现UNIX汉字输入输出的新的方法。

1 引言

西文系统要实现汉字信息处理。这涉及汉字编码方案、汉字内码选择、汉字输入输出设备和输入输出实现方法诸方面,这些方面在实现时相互关联而又互相独立。国内现有的汉化系统(如CCDOS)多采用修改操作系统的I/O模块的方法来实现汉字的I/O。这种方法的一个主要问题是实现起来较困难,难于确保与西文系统的高度兼容性。

本文提出了一种方法,它根据UNIX操作系统结构上的层次关系及UNIX文件系统特有的pipe功能,通过在UNIX外层添加一

个汉字命令解释程序ccshell及in, out两程序来实现汉字的I/O。由于这是在操作系统的外层实现的,对其内核不做大的改动,因而可确保中西文兼容。

2 UNIX操作系统浅析

首先,对于应用程序员来说:

- UNIX在多厂家的各种硬件结构上可实现源级程序移植,即UNIX独立于厂家,独立于硬件。

- UNIX允许某个程序的输出成为另一个程序的输入,这样原有的程序模块可用于

组变成 $(R_0 X_1 \dots X_j R_j AR_m, aw, \pi \parallel A \rightarrow X_{j+1} \dots X_n)$;若 $eg_j(A) = \text{空}$,分析停止。

4.3 若 $C_n = A$,三元组为 $(R_0 S' R_n, \varepsilon, \pi)$,则 z 被接收,且 π 是它的分析过程中应用产生式的顺序序列。

4.4 若 $C_n = L$,下推表不变,假定此时 $aw = a_1 \dots a_j t$,其中 $j \leq K$,令 $m = 1$ 且 $q = n$,查看 eg_j :

a.若 $eg_j(a_m) = \text{空}$,分析停止。

b.若 $eg_j(a_m) = R_\eta$,而 R_η 行 C_η 和 eg_η 为:

(i)若 $C_\eta = L$,则 m 增加1, q 置成 η ,执行a或b

(ii)若 $C_\eta = S$,执行(4.1)。

(iii)若 $C_\eta = N$,且 N 相应产生式 $A \rightarrow \beta$, $A \neq S'$,执行(4.2)。(注意 $C_\eta \neq A$)

在以上的分析算法中,分析停止,均表

示出错情况。

对于 $LR(K)$ 文法 G , $L(G)$ 中任一句子,我们用简化表进行归约分析与规范分析效果是一样的。它们都应用了相同的产生式序列。对于一个非 $L(G)$ 中句子,简化表进行归约分析比规范分析报错时间上要晚一些。

参考文献

- 1 ANCONA., M, AND GIANUZZIV. Implementation of reduced $LR(K)$ tables when $K > 1$. Tech. Rep. 47, IMACNR, Genova, Italy, 1980
- 2 ANCONA, M., AND GIANUZZ V. A new method for implementing $LR(K)$ tables. Inf. process. Lett. 13, 4 and 5 (1981), 171—176

建立新的应用。

- UNIX的实用程序可再次使用, 便于开发新的程序。

- UNIX是多用户和多任务操作系统, 比其它开放结构的DOS和OS/2应用更广, 适应性更强。

正是由于以上特性, UNIX系统已成为最受欢迎, 最广泛应用的小型计算机操作系统, UNIX将是90年代发展最快的产品。

让我们再从结构上来看一看, UNIX在结构上可分为核心和外层两部分。核心部分包含了操作系统的主要功能, 如存储管理, 进程和处理机管理, 设备管理和文件管理等, 在核心的最外层是系统调用, 它是UNIX核心的对外接口, 是用户程序与核心之间仅有的接口。也是用户程序取得操作系统服务的唯一途径, 核心外面的shell是UNIX操作系统的命令程序设计语言和命令程序解释程序的统称, 是用户与UNIX系统间的接口。应当注意在一般操作系统中, 命令解释语言是作为操作系统核心的一部分, 而在UNIX系统中, 它已不属于UNIX核心, 而是在核心外以用户态运行。shell根据用户输入的命令, 找到相应的模块中的程序。并为之建立进程并执行之。中西文兼容UNIX结构图如图1所示。

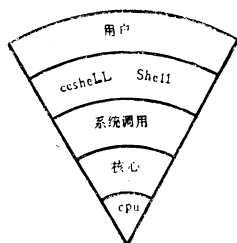


图1 中西文兼容UNIX结构图

3 UNIX汉字I/O实现基本思想及数据结构

对操作系统进行汉化, 首先应确保与原西文系统的兼容性, 即西文系统提供的所有命令和系统功能调用以及它支持的所有软件, 均应在汉化后系统的运行环境中能正确

地被执行。同时, 应具备汉字I/O功能和汉字信息处理; 这就要求具备一套汉字信息处理软件, 这套软件包括两部分: a. 汉字信息 I/O 软件; b. 汉字信息处理支持软件。即汉字库的建立和维护软件, 词库的生成和维护软件, 各种输入编码方案的处理软件以及中文编辑软件等, 这些汉字信息处理支持软件, 提供了汉字信息I/O服务一级的功能。为便于不同层次的用户使用, 一般将其汇集成实用程序的形式, 根据不同的需求选择。相应地加载至内存常驻或存储在硬盘上。

传统的汉化方法, 实践证明也是一种直观有效的方法, 如对西文PC—DOS汉化完成的CCDOS系统即是一个成功的系统例子。但这种方法因为牵涉到系统内层(模块级的修改), 牵涉面大, 产生的副效应也多, 为确保中西文兼容, 因而工作量是很大的。根据上一章我们对UNIX结构的剖析, 我们如果根据UNIX系统结构的层次关系, 及UNIX文件系统特有的pipe功能。通过在UNIX外层添加一个汉字命令解释程序及相应模块是可以实现汉字输入输出的。由于这是在操作系统的外层实现的。对其内核不做重大的改动, 因而完全证明与原西文系统的兼容性, 实现起来也极为简捷。

汉字处理系统一般首先应设计一个完整的汉字编码系统, 确定汉字信息在汉字处理的各个环节中具有的不同编码, 并通过内部相应的程序模块对其编码作出适当的变换, 以适应不同的信息处理要求。

在汉化的UNIX中也分有几种编码; 输入码(如区位码、拼音码、五笔字型码等), 机内码(一般采用国标码)和输出码(汉字点阵码)。在计算机处理时, 因汉字信息的编码形式多, 规则复杂, 输入码→机内码, 机内码→汉字点阵码之间的变换不能用简单的映射关系, 应采用依据换码对照表转换的方法。

从键盘输入的汉字码由输入换码程序(即本文下面将提到的in模块)凭借输入换

码表转换成机内码。换码表按字典排列法构成, 整个表可采用索引—线性表结构, 所有汉字分成若干子线性表, 每一子表放相应个对照项, 如图2所示。

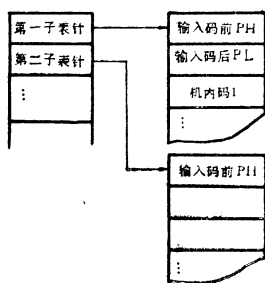


图2 输入换码表数据结构

输入换码程序对索引表可采用折半法检索, 对子线性表采用平方根法检索, 整个换码表存于磁盘空间, 其索引表常驻内存。

机内码由输出换码程序(即后面将提到的out模块)借助于输出换码表换成字形点阵(当然也可以采用固化的字形点阵库, 但采用软件方法更方便), 换码表以机内码值排序, 是一个线性表, 它可作为磁盘上的一个库文件。

4 UNIX 汉字I/O处理过程及程序流程

UNIX 系统初启后, 只有进程“0”和“1”。进程“0”负责程序对换和分配处理机, 进程“0”在核心态下运行。进程“1”的任务是为终端用户建立进程, 它是除进程“0”外系统内所有进程的祖先。

首先, 由进程“1”建立一个汉字命令解释进程ccshell, 该进程与进程“1”建立起来的其它进程(如shell)等同地处于系统之中, 由ccshell又生成输入进程in和输出进程out, 并在ccshell, in, out之间建立输入管道pipein和输出管道pipeout。in的职能

是读键盘, 凡由键盘输入的汉字外码(即汉字输入码)由它转变成汉字机内码。in把汉字内码和其它键盘输入放入pipein, ccshell从pipein中取得in的输出(即机内码)及操作员的键盘命令, ccshell检查解释并通过生成新的子进程cexec来执行键盘命令, cexec也从pipein中获得键盘输入, ccshell和cexec的输出送入pipeout文件。输出子进程out负责处理pipeout中的数据, 例如把其中的汉字机内码转换成汉字点阵码送给显示或打印驱动程序, cexec结束后, 通知ccshell, ccshell又继续从pipein中读取键盘命令并生成新的cexec执行之, 如此不断循环, in进程由键盘中断驱动, out进程在pipeout取空时被挂起, 任何对pipeout的重新写入都将使之重新活动。

这样, 添加模块ccshell, in, out就达到了与修改操作系统的输入输出程序相同的目的与修改操作系统的输入输出模块比较起来, 本方法仅增加了写入读出两个pipe文件的开销, 但因pipe文件存于内存, 不会增加对换的麻烦, 所以开销不是很大。由于添加的ccshell与系统原有的shell一样是在UNIX的外层, 对UNIX内核没有改动, 从而也完全而简便地保证了与西文UNIX的兼容性。

图3为系统内进程示意图。

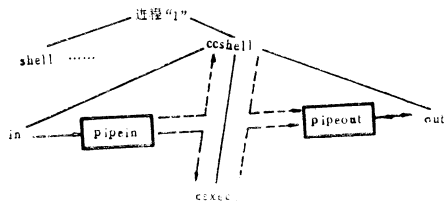


图3

ccshell, in, out模块的功能及框图

这三个模块是三个独立的模块, 各自有各自的功能。设计时, 可作为三个独立的程序进行编制。下面分别说明它们各自的功能并给出其粗略框图。

4.1 ccsHELL汉字命令解释程序

功能：首先要生成两个子进程in, out, 并建立三者间的通讯 pipein 和pipeout. ccsHELL. 进程与in, out 三进程共行执行, ccsHELL从pipein 取得数据及键盘命令, 并生成新的子进程cexec执行键盘命令. cexec结束后通知ccsHELL. ccsHELL又循环从pipein中读取键盘命令并生成新的cexec执行之, 如此不断循环, ccsHELL及cexec的输出放入pipeout文件. 如图4.

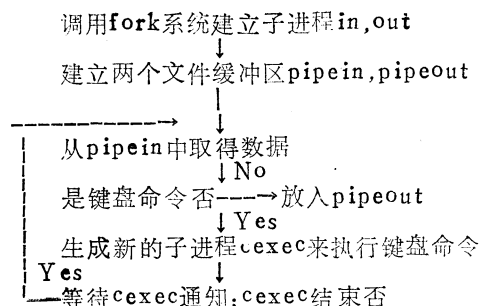


图4 ccsHELL程序框图

4.2 in 输入换码程序

功能：读键盘，将键盘输入的汉字外码转变成汉字内码，把汉字内码和其它键盘输入放入pipein, 框图见图5。

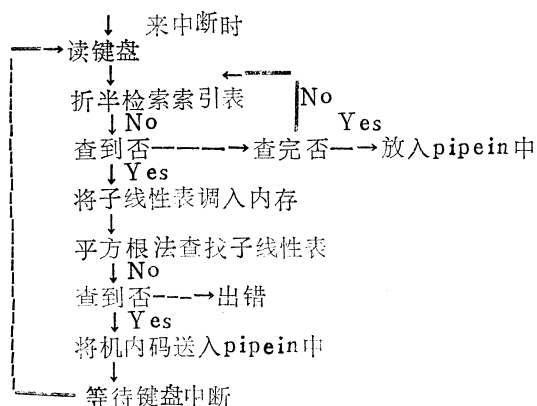


图5 in程序框图

4.3 out输出换码程序

功能：负责处理pipeout中的数据, 把汉

字内码转换成汉字点阵码, 送给显示或打印驱动程序, 框图见图6。

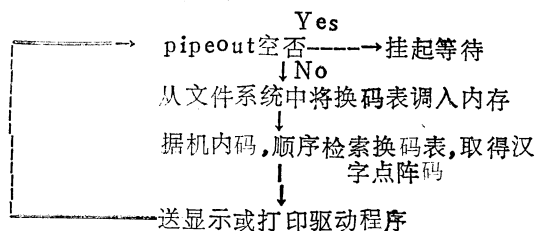


图6 out程序框图

5 结束语

这种方法是建立在UNIX支持的多道程序, 进程通信工具等之上的, 与具体的输入输出设备及其驱动程序无关, 因而它是一种UNIX操作系统通用的汉字处理技术, ccsHELL及in, out程序最好也用c语言编写, 以利用这种技术移植。

ccsHELL, in, out是三个独立的实体, 在运行时, 是以三个独立的进程共存于系统进程键之中; 在设计时, 是分为三个单独的进程进行编制的, 由于这一特点, 使这种输入输出技术与汉字输入输出的其它方面如汉字编码方案、汉字内码选择、汉字输入输出设备间的关系比较松散, 因此, 这种汉字处理技术可以很容易地支持各种不同的编码方案、机内码及各种输入输出设备。

在UNIX上具体实现时, 此方法还有很多的具体问题需要解决。由于能力有限, 作者也不可能考虑得很完备。另外还必须注意的是, 由于Pipe文件是UNIX系统的独特之处, 其它的操作系统皆不具备此特性, 所以本方法仅适用于UNIX操作系统。

参考文献

- 1 UNIX操作系统 武汉大学计算机科学系翻印, 1983

CCD高速数据采集装置

重庆建工学院机电系 伍龙田

(邮码 630045)

摘要 为了实现对视频、超声、雷达等信号的采集。必需要求10MHZ以上的采样速度, 这要求有特高速($0.1\mu\text{s}$)的A/D转换器, 以及相同速度的存贮缓冲器(1KB以上的容量), 这在技术上和经济上都要花很高的代价。本文提出了一种替代方案, 采用新型的CCD芯片(R5103)利用它高速(15MHZ的采样速度)和大的存贮容量(910字节)较便宜地实现了对超声回波的采集接口。并给出了 CCD和A/D转换部份的线路和工作原理。

关键词 CCD电荷耦合装置 采样时钟 读取时钟 二次采样

1 概 述

高速采集视频信号、雷达、超声信号时, 由于信号频率高达几兆赫, 根据香农定理, 采样频率至少为信号频率两倍, 所以一般采样频率都要求大于10MHZ。对于这样高的采样频率使信号采样并转变为数字信号, 再送入计算机中, 不可避免地面对两大矛盾。一是高速的A/D转换器, 如(ADC0809)它的转换时间是 $100\mu\text{s}$ 左右, 即它的采样频率在 10^4 次/秒以下, 比这里要求的采样频率相差 10^3 倍以上。目前已有采样频率大于10MHZ的A/D芯片出售, 但它们价格很高(几千元以上)并很难买到。另一大矛盾是计算机的I/O输入速度, 常规的查询I/O输入速度一般只有每秒几十KB即使使用DMA技术, (如PC机中使用的8237DMA控制器)也只能达到每秒1MB以下。对于高速数据采样频率大于10MHZ时, 计算机的输入速度也是明显低于数据采集速度(相差3个数量级)。所以高速数据采集装置必须有高速($0.1\mu\text{s}$)数据存贮(缓冲)器, 然后用较低速向CPU传送, 所以价格是很高的(从5000美元到20,000美元)。这里介绍一种使用CCD装置来实现高速数据采集, (而价格相对又便宜得多)。此装置成功地用于超声回波采集, 用10MHZ采样频率每次可采

集910字节, 然后以95KB/秒速度向CPU传送。现在已有作成集成电路芯片的CCD装置出售, 它们有的能工作在15MHZ的采样频率, 我们正是利用了这个特点实现了高速数据的采样和转换。

2 装置工作原理

整个装置的框图如图1所示。

2.1 组成

装置实际上分成三个部份, 分别装在三块接口板上, 其中第一部份为CCD及其附属电路。第二部份为A/D转换及与CPU的接口。第三部份为控制器, 分别对CCD及A/D转换发出有关的控制命令。

2.2 装置的设计思想

装置的设计思想: a. 利用 CCD 实现对模拟信号的高速(10MHZ)采样和存贮。 b. 然后用较慢的频率(95KHZ)从CCD读出采样数据, 实现模拟量转换成数字信号, 并对CPU实现同步传输。每次采样存贮或读出都控制在910个采样脉冲。由于采集系统对计算机来说是一个异步系统, 因此协调各部分之间工作, 并保持与CPU 同步必须单独设计一个控制器。这里将重点介绍CCD部份及A/D转换部分及其同步部分。

3 CCD采样器

驱动电路、CCD器件及其偏置、频率特性补偿三部分)。

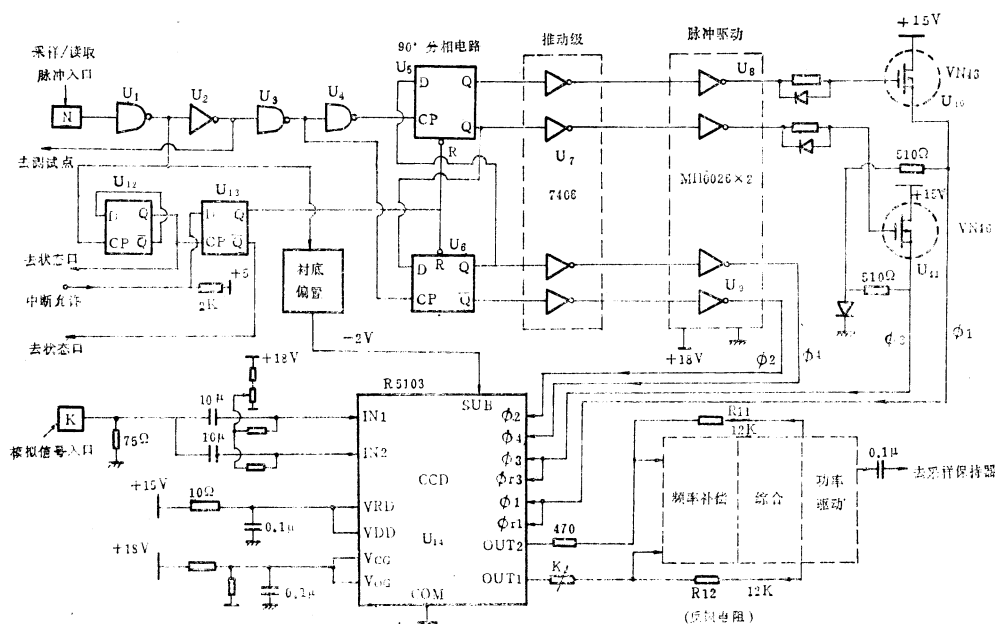
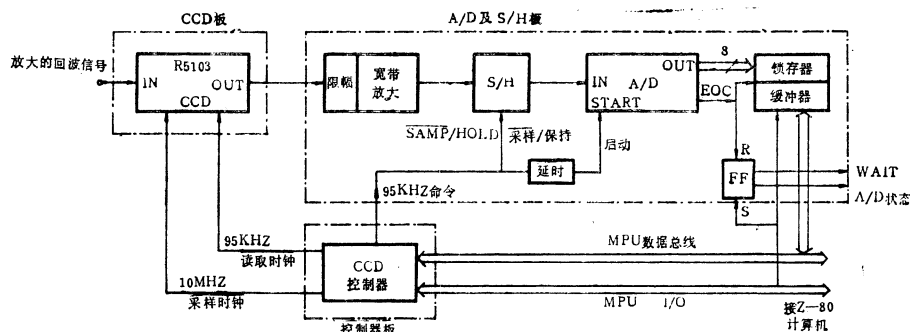


图2 CCD采样器逻辑框图

其中CCD器件由R5103组成，它是由两个910级的视频延时线（错开 $\frac{1}{2}$ 级）及多路开关构成，封装在双列16腿的集成芯片中。它的逻辑框图结构如图3所示、它的工作波形如图4所示。

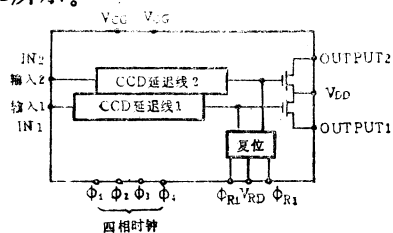


图3 R5103 (CCD) 逻辑框图

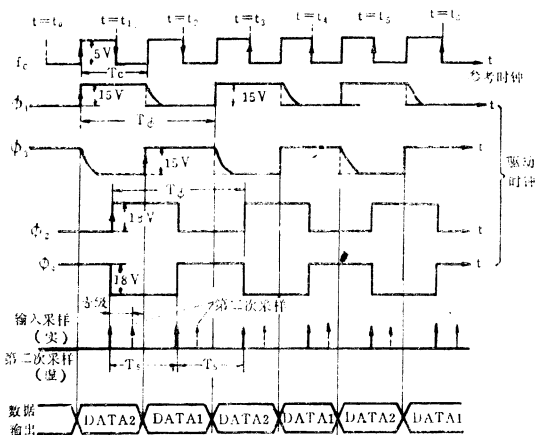


图4 CCD工作波形

图中, $\phi_1\phi_3$ 为奇数时钟驱动电压, 幅度为15伏。

$\phi_2\phi_4$ 为偶数时钟驱动电压, 幅度为18伏。

$\phi_{r1}\phi_{r3}$ 为复位时钟, 幅度为15伏。

V_{CG} , V_{OG} 为控制偏置, 典型值为+5伏 (V_{OG} 为输出偏置)

V_{DD} V_{RD} 为漏极电压典型值为+15伏 (V_{DD} 为输出漏压)

V_{SUB} 衬底偏置典型值为-2伏

另外 IN_1 , IN_2 也要加上电压偏置, 典型值为+8伏一般使用中, IN_1 IN_2 是接受同一模拟输入信号, V_{CG} V_{OG} 是直接短接。 V_{DD} V_{RD} 短接至+15伏电源上。从图4中可以看出有以下特点: a. 四相驱动时钟 $\phi_1-\phi_4$ 由参考时钟 f_c 分相得来, f_c 每跳变一次产生一相驱动时钟, 按1—2—3—4顺序产生。b. 奇数时钟的下降沿是缓变下滑的 (拖尾), 而偶数时钟却是陡变的下降。这是为了保证少数载流子顺利地向前转移所必须的。c. 驱动时钟是MOS电平 (15伏) 而参考时钟 f_c 是TTL电平 (5伏), 因此需要相应的变换电路。d. 模拟信号的采样是在 $\phi_2\phi_4$ 的上升沿处进行, 因此 $T_s = 2T_s = 2T_c$ 。相应地有 $f_s = f_c = 2f_\phi$ (f_ϕ = 驱动时钟频率) 在这里我们采用的是 $f_s = 10\text{MHz}$ 。e. 复位时钟 ϕ_{r1} , ϕ_{r3} 必须与 $\phi_1\phi_3$ 同步, 因为数据 $DATA1$ 出现在 ϕ_1 为低电平的时候, ϕ_{r1} 与 ϕ_1 同步才不致使 $DATA1$ 控制门复位。我们这里是使 ϕ_{r1} 与 ϕ_1 直接短接, ϕ_{r3} 与 ϕ_3 短接。

根据以上特点, 我们设计了图2所示的电路其中CCD的输入偏置由电位器 W_1 设定, 应使R5103的腿2及腿3在无输入信号时, 整定为8伏。衬底偏置由 D_1D_2 整流参考时钟 f_c 得到-2伏电压加至CCD的腿1上, 并需加上 $0.1\mu\text{F}$ 电容旁路接地。控制偏置 V_{OG} V_{CG} 直接由电源电压+18伏分压得到, 调整分压电阻使CCD腿12和腿16得到+5伏电压。 V_{DD} 及 V_{RD} 直接由稳压电源+15伏提供。漏极电压必须经稳压 (7815芯片) 并加以

$0.1\mu\text{F}$ 电容滤波旁路, 线路中 $U_4U_5U_6$ 用于对参考时钟分成相互相差 90° 的四相时钟。注意 U_5 的 Q 、 \bar{Q} 端, U_6 的 Q 、 \bar{Q} 端分别构成 $\phi_1\phi_3\phi_4\phi_2$, 任何一个位置颠倒将导致CCD不能正常工作。并分别接入CCD的相应时钟输入端。为了实现从TTL→MOS电平的转换, 线路中采用了专门的时钟驱动器 U_8U_9 。(NS.公司的MH0026) 它可以工作在20伏电压, 工作频率达10MHZ, 最大输出电流达1.5安, 由于它的逻辑“1”输入电流达10mA, 分相电路74LS74D触发器不能直接提供, 故增加了 U_7 (7406) 用以驱动 MH0026 驱动器。 U_7 是集电极开路型驱动门, 它的驱动电流直接由外接上拉电阻 (680Ω) 决定。为了实现 $\phi_1\phi_3$ 的下降沿拖尾, U_8U_9 后面加接了 $U_{10}U_{11}$, 在下降沿时, $\phi_1\phi_3$ 直接经 D_4D_5 驱动, 而在逻辑1时经 $U_{10}U_{11}$ 组成的场效应管源跟随器驱动。并使它们的高电压保持在+15伏左右, 而 $\phi_2\phi_4$ 直接驱动, 可使高电压保持在+18伏左右。CCD模拟视频信号在读出时由脚9和脚11经 $U_{15}U_{16}U_{17}$ 组成的频率补偿电路加以补偿使其输出电压的频率响应补偿到5MHZ左右 (而不加补偿时只有1.5MHZ)。其频率特性如图5所示。由于CCD器件本质上是一个模拟信号采样装置, 所以它的频率特性依赖于采样频率, 在这里 $f_c = 10\text{MHz}$ 条件下, 把频率补偿网路设计成5MHZ带宽是必要的, 大于 $f_c/2$ 成份应该被滤掉。其补偿的深度主要由反馈电阻 R_{11} 及 R_{12} 决定。输出数据 $DATA1$, $DATA2$ 在CCD内部自动分相 (多路开关分相) 后经脚9脚11两路输出并经各自频率补偿后

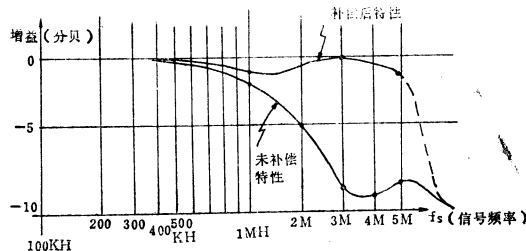


图5 R5103频率响应

在电位器 W_2 上综合平衡后再送入末级推挽输出级放大,再经A脚送至A/D转换板中采样保持器,最后经A/D转换后送入计算机(如图1所示)CCD的输入脚2和脚3对视频信号等效于短接并接入 75Ω 匹配阻抗,通过同轴电缆和视频信号源相联接。

4 A/D转换电路

A/D转换板电路逻辑框图如图6所示,调整好的CCD波形如图7所示。图6中 U_1 为采样保持器, U_2 为模数转换芯片, U_3 为8位锁存器, U_4 为总线驱动器。它们的型号和管脚接线都已标注在图中。从CCD读出的(以95KHZ频率时钟驱动)模拟信号送入A/D板的N端输入经二极管限幅和三级 μA 741组成的宽带放大后,送入S/H进行再一次采样保持,最后经A/D转换后送入8T97缓冲器供Z-80 CPU读取。为了保持与CPU的读取同步,这里选取CCD的读取频率为95 KHZ。这是本电路特点之一。一般常用CPU查询方式进行I/O,每读取一字节大约需30-35 μs 。用中断方式也差不多化相同的时间(视CPU不同而不同),只有Z-80 CPU有

特殊的INIR指令,它可以以一种“准DMA”方式进行块数据输入,它每读取一个字节只需10.5 μs (相当于95KHZ),这是除了DMA方式外最快的I/O输入方式了。所以本电路采用Z-80作CPU以求得最快的输入速度。在这个数据链上所有器件都必须大于(或等于)10.5 μs 的速度工作。为此我们选取了高速的A/D转换器ADC EH8B1,它完成一次模数转换只需4 μs 。也选取了高速采样保持器SHM-1(DATEL公司)它的捕捉时间(即一次采样时间)只有2 μs (保持电容 $CH=1000PF$)。而不能选取市场常见的低速A/D芯片如ADC0804、ADC0809等,也不能采用低速的采样保持器AD582或LF398等。仅仅使CCD的读取速度和CPU的输入速度大致相等(不可能绝对相等)并不能真正保持与CPU的完全同步。为此电路专门设置了 U_7 (74LS73),用以完成同步工作。每当CPU读取一次A/D转换数据,都将使 U_7 置“1”。下一次欲再读时,通过M点的I/O命令(IN A,12H)将使CPU陷入WAIT“等待状态”,直到再完成一次新的A/D转换由EOC脉冲使 U_7 复位为“0”才能解除CPU的“等待状态”,从而自动地完

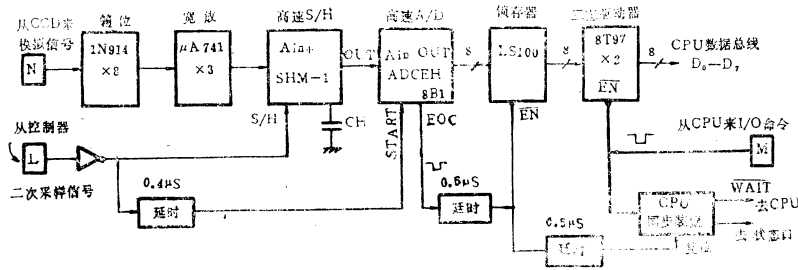


图6 A/D及S/H电路

成了A/D转换与CPU的同步工作。

本电路的第二个特点就是第二次采样脉冲的设计(见图4)。从CCD控制器发来的第二次采样脉冲是一个宽度为2.2 μs 的负极性脉冲,并且距离CCD采样时刻滞后约1.5 μs 。这样安排的目的在于获取真实的采样值。如果我们从示波器观察一下CCD输入、输出波形(及放大), (见图7)从放大波形

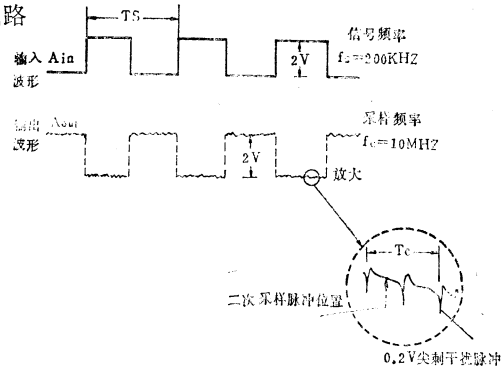


图7 CCD采样波形(输入输出)

EM—891 Ethernet监控器的实现

吉林工业大学 程湘云

摘要 目前局部网的应用已越来越广泛,它的性能和稳定性已成为整个网络系统的关键。然而网络管理人员对运行中的网络上的情况却所知甚少,因此在网络软件开发,网络调试及网络维护等过程中,网络监控器就成了一个非常有用的工具。本文实现的 Ethernet 监控器能清楚地观察到以太网上的情况。

关键词 以太网网络监控器 网络调试 网络维护

1 概述

EM-891 Ethernet监控器是一种便携式网络分析监控器。它具有监视,统计,模拟负载,永久性地保存必要报文及分析网络报文的能力。它是建立,维护及开发以太网的有效工具。该监控器尤其适用于DNA局部网的维护和开发。

该监控器主要功能:

· 数据显示

EM—891 Ethernet监控器可以联机不间断地显示网络上传输的数据，并可通过保持屏幕随时停止后续数据的CRT显示。字符显示形式有两种选择，即ASCII码和十六进制形式。CRT共有25行，每行80列。两个数据行中间为一低亮度显示的虚线分隔行。最后两行用于功能提示区。

· 数据记录

[illegible]

图中可以发现,输入波形经CCD采样输出后,波形引来了干扰(尖刺状脉冲),它集中在每次采样的开始,这是由于驱动时钟直接耦合至输出级的驱动门带来的,另一处是芯片内的多路开关在DATA1和DATA2的换转时产生。主要的两个尖刺脉冲在时间上相距大约一个采样周期(图7中为 $0.1\mu\text{s}$)。因此为了使从CCD读出的模拟信号能正确进行A/D转换,必须对模拟信号进行第二次采样,并躲开这两个尖刺干扰脉冲。为此,电路中作了下述安排,在控制器中,一方面在向

该监控器在联机方式下,可在数据显示的同时不间断地提供报文记录的存盘。一片软盘的数据存储容量为720k字节。

• 统计功能

它包括对整个网络情况的统计和对每一个节点收、发报文情况的统计。对整个网络情况的统计包括开机后网络上总的接收报文字节数, 帧数, 发送报文字节数, 帧数, 发生单次及多次发送碰撞的次数等等。

• 协议分析

该监控器对 DNA 网络上传输的报文进行了链路层、路由层和端端通信三层协议的分析。

• 模拟负载

为了能较真实地模拟端节点发送情况,该监控器按泊松分布规律发送报文。

• 系统结构

该监控器包括系统处理器 (8兆赫8086 CPU), 一个 3.5 英寸的软盘驱动器, 一

CCD发出读出的95KHZ 的参考脉冲的同时又把这参考脉冲用单稳态延时 $1.5\mu\text{s}$ 以躲开采样开始的第一干扰尖峰,另一方面再用第二单稳把前一单稳输出(已延时)的脉冲定宽成 $2.2\mu\text{s}$ 后发向A/D板作为采样命令(见图7),这样对应了CCD“阶梯”信号中间段真实的模拟值,经S/H采样固定在保持电容 C_H 上供A/D转换成数字量,锁存后供CPU读取存入内存。因此采样保持器必须选择捕捉时间 T_{AC} 在 $2\mu\text{s}$ 的高速采样保持器 SHM-1(DATEL公司产品)。

(下转第31页)

2.2.2 路由层

路由层报文分为:

a.分组路由报头 { 长格式
短格式

b.路由选择控制报文 { 在非广播电路中 { 初始化报文
验证报文
呼叫和测试报文
在广播电路中 { 一级路由选择报文
二级路由选择报文
以太网路由节点呼叫报文
以太网端节点呼叫报文

显示举例如下:

Frame,70	Bytes=60
ROU ,Len=34	PadLen=1 Control Message,Ethernet Endnode Hello Message
ROU ,Tiver=020000h.	Id=AA0004005204. linfo=03. Biksize=DA05h. Area=00
ROU ,Seed=0000000000000000.	Neighbour=AA0004005404. Timer=0F00h. Mpd=00
Frame,138	Bytes=60
R1OU ,Len=29	PadLen=1 Flags=Long.Intra-Ethernet
14042001200D80	
Frame,222	Bytes=240
ROU ,Len=224	Padlen=1 Control Message,Level 1 Routing Message
ROU ,Srcnode=5404h	

2.2.3 端端通信层 NSP报文分为:

a.数据报文 { 数据段报文
中断报文
链路服务报文
b.应答报文 { 数据应答报文
其它数据应答报文
链接应答报文
c.控制报文 { 无操作报文
链接初始化报文
链接确认报文
拆链初始化报文
拆链确认报文

显示举例:

Frame,336	Bytes=60
ROU ,Len=33	PadLen=1 Flags=Long.Intra-Ethernet
NSP ,Msgflg,Data.Link Service	Dstaddr=0420h. Srcaddr=0120h.

Frame,337 Bytes=60

ROU ,Len=30 PadLen=1 Flags=Long,Intra-Ethernet

NSP ,Msgflg,Control,Disconnect Initiate. Dstaddr=0120h. Srcaddr=0420h.

NSP ,Reason=0000h.

00

Frame,338 Bytes=60

ROU ,Len=29 PadLen=1 Flags=Long,Intra-Ethernet

NSP ,Msgflg,Control,Disconnect Complete. Dstaddr=0420h. Srcaddr=0120h.

NSP ,Reason=2A00h.

协议分析有助于了解网络上某时刻报文的传输方向及类型。使原本不可见的网络内部情况展示在我们面前。同时对网络协议的剖析还有助于网际互连的实现。显示与协议分析子程序框图见图2。

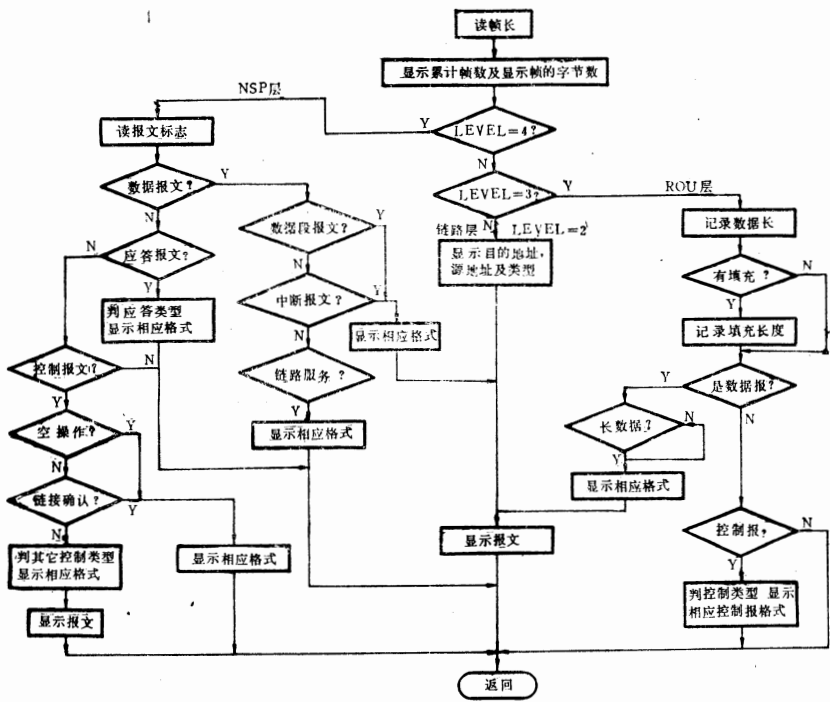


图 2

2.3 模拟负载功能

在新网络投入运行前,用模拟负载功能可以对网络的性能进行评估。同时通过模拟负载功能还可预测增加站点对现存网络的影响。由于模拟负载发送的帧的长度及频率均

可选择,因此测量起来非常方便。由此可以了解所增负载可能产生的传输时延及碰撞情况。为了能较真实地模拟负载,该监控器按泊松分布规律实现发送。具体数表略。选择发送方式子程序框图如图3所示。

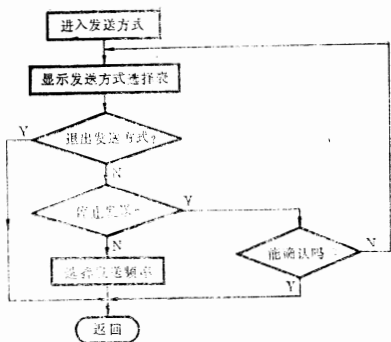


图3 选择发送方式子程序框图

3 结束语

除上述功能外,本监控器还具有存盘等诸多功能,在此不再一一例举。用户主程序框图如图4所示。目前该监控器运行情况效果较好,基本达到了原目的要求。本监控器虽是在DEC net支持的以太网路上实现的,但它同样适用于其它以太网网络。

000 040 060 080 100 120 140 160 180 200 220 240 260 280 300 320 340 360 380 400 420 440 460 480 500

(上接第27页)

5 控制器

控制器的任务主要是保证第一次采样时需要的10MHz参考时钟,以及读取时95KHZ的参考脉冲并保证它们的个数都是910个。并发出精确定时的第二次采样命令脉冲。由篇幅原因不再赘述。

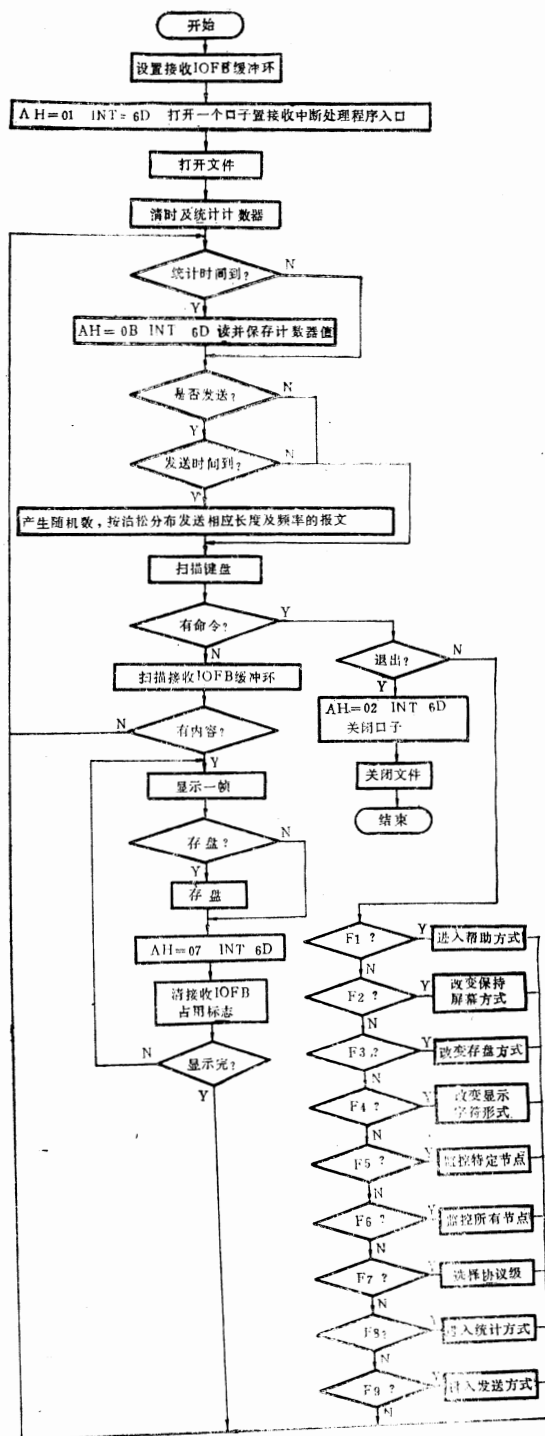


图4 用户主程序框图

微机电力系统暂态稳定实验装置

东北电力学院 张文生

吉林市长春路169号 邮码132012)

1 问题的提出

在电力系统动模实验中,当研究电力系统暂态稳定问题时,要求模拟系统能够方便、灵活地改变运行方式,并能随意地在给定点形成各种形式的故障(三相短路、两相短路接地等),同时,要求实验装置能够精确地测定出对应于某一种故障形式,能保持该系统暂态稳定的极限切除时间,为达此目的,笔者设计了一种用于电力系统暂态稳定实验研究的装置,投入实际应用以后,达到了预期的效果。

2 实验对装置的要求

以下以最简单的单机对无穷大系统为例,说明实验过程及对装置的要求。

实验的模型系统如图1所示,该系统中模拟发电机、模拟变压器、模拟输电线路的标么参数及时间常数与原型系统一致。

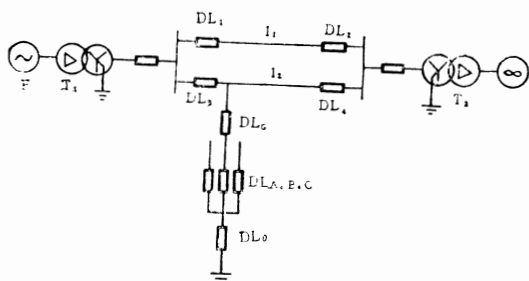


图1 实验系统原理接线图

图中

F——模拟发电机

T₁, T₂——模拟变压器

l₁, l₂——模拟输电线路

DL_{1~4}——模拟输电线路始、末端断路器

DL_G——故障形成开关

DL_{A, B, C, O}——分相控制开关,用以设定不同的故障形式。

实验步骤如下:

a.启动发电机组,经双回线与无穷大系统并列,并向系统输送一定的有功、无功功率,建立起正常的运行方式。

b.通过故障形式设置开关设置故障形式,如要形成两相短路接地故障,则合DL_A, DL_B, DL_O开关,如要形成三相短路故障,合DL_A, DL_B, DL_C。

c.在t=0秒时合三相开关DL_G形成故障(故障点在输电线路的始端)。

d.经过时间Δt,跳开DL₃, DL₄,即切除故障线路l₂,使系统单回线运行。

e.观察此时系统是否失去暂态稳定,如发电机经过几次摇摆后恢复了稳定,则断开故障开关DL_G,重合线路开关DL₃, DL₄。

f.增加故障持续时间Δt,再次合故障开关DL_G形成故障,并经时间Δt后,切除故障线路l₂。

g.如此不断反复进行,只要系统不失去稳定,就不断增大短路持续时间Δt,直至发电机失步为止,记下此时的Δt值,此值即为此系统对应于某一故障形式,某一运行水平,保持系统暂态稳定下的极限切除时间t_{cr}。

分析以上实验过程可以看出:

a.实验中要频繁操作故障开关DL_G及线路l₂开关DL₃, DL₄,为了精确测定Δt之值,对于每一种故障形式,各开关往往需要操作多次。

b.实验中Δt的增加值,希望每次愈小愈好,每次设置的Δt值都要准确、可靠。

3 装置原理

考虑到对实验装置的上述要求,笔者采用TP—801单板计算机控制故障开关 DL_G 、线路 l_2 开关 DL_3 、 DL_4 的通、断,并通过软件中的延时程序设置故障持续时间 Δt ,在实验中可以方便地改变 Δt 值,以下为装置原理见图2。

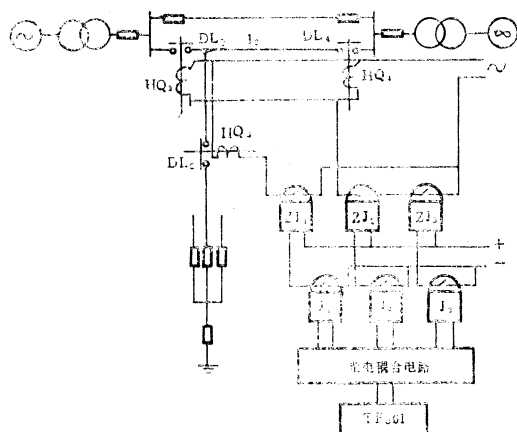


图2 装置原理图

图中:

DL_3, DL_4 ——线路 l_2 始末端开关

DL_G ——故障形成开关

HQ_3, HQ_4 —— DL_3, DL_4 的合闸线圈

ZJ_1 —— DL_G 的合闸中间继电器

ZJ_2 —— DL_3, DL_4 的跳闸中间继电器

ZJ_3 —— DL_3, DL_4 的合闸中间继电器

J_1, J_2, J_3 ——灵敏继电器

线路开关 DL_3 、 DL_4 及故障开关 DL_G 均用交流接触器来实现,其操作电源为交流220V,中间继电器 ZJ_1 、 ZJ_2 、 ZJ_3 的作用是增大接点的容量,其操作电源为220V直流。

J_1, J_2, J_3 为灵敏继电器,受光电耦合电路输出电压的控制,光电耦合电路用于将被控系统与单板计算机隔离开来。

装置通过并行输入输出接口 Z80—PIO与计算机连接,用其输出的开关量来控制灵敏继电器,当某一位为高电平时,其对应的灵敏继电器接通,从而使中间继电器励磁,

当该位为低电平时,其对应的灵敏继电器失磁,中间继电器亦失磁,以此来控制各开关的分与合,本装置使用了Z80—PIO的三位,一位用于合故障开关 DL_G ,一位用于同合线路 l_2 开关 DL_3 、 DL_4 ,一位用于同断线路 l_2 开关 DL_3 、 DL_4 。

4 程序框图

图3为程序框图。

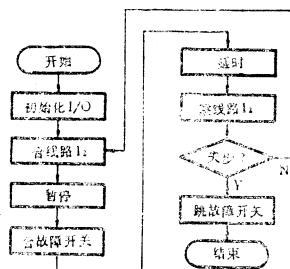


图 3

程序首先初始化PIO,将其置成输出方式,线路 l_2 的原始状态为断开,程序初始化PIO之后自动将线路 l_2 投入运行并进入等待状态,在选定了故障形式之后,给出指令使程序继续运行,合故障开关,经过时间 Δt 后,跳开故障线路 l_2 ,如此时发电机未失步则重合 l_2 开关,调整 Δt 值,如失步则切除故障。

时间 Δt 是通过延时程序设定的,其值可以通过改变B寄存器及DE寄存器对中的内容随意给定,其计算公式为:

$$\Delta t = (24 \times T16 + 23) \times \text{TIME} \times 0.5 (\mu s)$$

式中T16—DE寄存器对中的内容

TIME—B寄存器中的内容

可在B寄存器中先设定一个数值,借助于上述公式,就可计算出当 $\Delta t=0.1s, 0.12s, 0.13s \dots$ 时的T16值,程序运行前只要改变T16的值,就可知 Δt 之值,修改非常方便。

5 结论

本实验装置具有如下特点:

a. 结构简单,造价低廉,运行灵活,可反复多次操作。

b. 由于利用了微机技术测定极限切除时间,取代了传统的利用时间继电器的测定方

微机力矩限制仪

杭州二轻科研所 陈守中

(杭州市中山路甘泽坊巷11号 邮编 310002)

摘要 本文介绍了单片微机起重力矩限制仪的工作原理,应用A/D芯片14433组成的多路数据采集器的设计思路,软件中如何建立、查寻表格的方法以及提高仪表自身抗干扰能力的措施。

1 概述

起重机械在工作过程中承受的力矩是有限的。也就是力臂越长,允许起吊的重物越轻。当起重机的主电机提升或降下重物,小车电机带动小车在横臂上作水平方向的运动时,有可能出现力矩过载的情况。为了防止起重机因过载发生倾翻、折臂的意外事故,国家劳动部等有关部门作出规定:起重机必须安装力矩限制器。本文介绍的微机力矩限制仪主要用于臂架式起重机的力矩保护,对起重机提升的重量,小车的工作幅度,风速以及提升速度四个参数连续检测,及时显示允许起重量及参数的检测值。一旦遇到越限,仪表自动采取保护措施并且发出声光报警信号,保护作业人员和设备的安全。微机力矩限制仪集检测、控制、保护功能于一体,充分显示了智能化仪表的优点。装备了该仪表的起重机已向东南亚地区的国家出口创汇。表1列出了仪表的功能。

2 硬件电路

起重机械力矩限制仪硬件电路由四个部分组成,硬件框图如图1所示,下面逐一加以介绍。

2.1 模拟电路

为了完成表格1的各项功能,单片机需

要对实际提升重量、工作幅度、风速三个模拟信号及重物提升速度的开关量进行检测。设计时选用双积分A/D芯片14433作三路模拟信号共用的A/D转换器。该芯片备有与计算机通讯的接口,能输出 $3^{1/2}$ 位BCD码。根据资料介绍将A/D芯片的转换速度定为25次/秒。单片机采用分时扫描的方法逐一接通模拟开关,对三路模拟信号分别进行A/D转换,待每路信号转换结束后,利用 V_R 端从外部使A/D芯片强迫复零。具体的方法是将MC14433的基准端 V_R 通过模拟开关接入-5V电压,保持8个以上的时钟周期即可完成复零。A/D转换的结果以BCD码的形式通过 P_2 口送入单片机内存。由于三路模拟信号的数量级不同,所以预处理的方法也

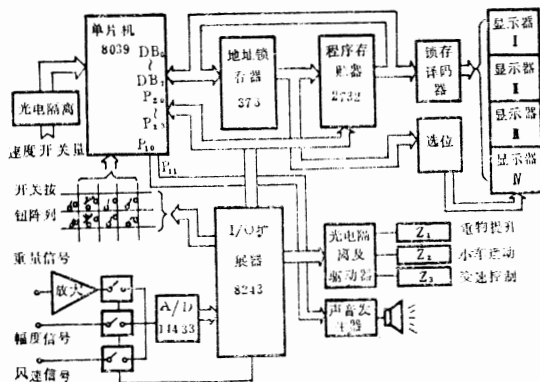


图 1

法,省去了在实验中多次整定时间继电器的麻烦,减小了误差,提高了实验水平。

c.利用本装置,还可方便地模拟电力系统的自动重合闸装置,硬件不改变,只需另行编

制程序,就可实现线路中某一点发生故障→瞬间切除故障线路→重合线路开关这一复杂过程,重合时间也可方便调整,这样简化了实验设备。

表 1

保护类别	保护条件	报警方式	保护功能
超载保护	起重量 ≥ 1.05 倍允许起重量	高音连续报警, 实际起重量数值闪烁	不允许重物提升, 不允许变速, 不允许小车前进
预警	1.05倍允许起重量 $>$ 实际起重量 ≥ 0.9 倍允许起重量	高音断续报警	
提升超速保护	起重量 ≥ 2.8 吨, 提升速度为45米/分 起重量 ≥ 1.19 吨, 提升速度为100米/分		不允许重物提升, 不允许变速
风速超速保护	风速 ≥ 20 米/秒	低音断续报警, 风速数值闪烁	
幅度保护	小车行程 \geq 设定工作极限	工作幅度数值闪烁	不允许小车前进
变速保护	提升重量 $\geq 100\text{kg}$		不允许变速

不同。重量传感器产生的毫伏级电压信号经自稳零斩波运放IC7650放大变成伏级信号, 幅度信号由小车电机经多级齿轮变速带动位移传感器取得, 风速信号由风杯的测速发电机产生, 经整流分压后方能送入A/D转换器。

2.2 单片机系统

起重量力矩限制仪以MC—48系列芯片8039为核心, 配以地址锁存器, 4k程序存储器组成最小系统。在P₂口低4位上经I/O扩展芯片8243扩充16个I/O口供外设使用。仪表设有清零、复位按键以及多位DIP开关, 主机频率选用6MHZ。

2.3 显示译码

显示器共有四组, 分别为风速/提升速度、幅度、允许提升重量、实际提升重量显示器。显示方式采用静态锁存, 使显示数值清晰稳定。为了节省端口, 从地址锁存器74LS373的输出端D₀—D₂经74LS138译码作为显示器的位选信号。

2.4 执行驱动

I/O扩展芯片的P₇口输出继电器动作指令, 经光电隔离, 达林顿电路反向驱动带动继电器动作。报警指令由P₁端口输出, 启动双时基电路NE556推动扬声器发出相应的报警声。

3 软件设计

程序按功能模块化, 图2给出了主程序的流向图。编程中根据国标的要求设计了自检

程序, 便于操作人员监视仪表自身的工作状况。由于篇幅缘故, 下面只叙述程序中的特点。

3.1 力矩表的建立和查寻 根据起重机制造厂提供的起重特性表的有关数据, 运用数学回归法计算出幅度与允许提升重量间一一对应的数据, 在程序存储器适当的页面内建立表格, 以供查寻。表格的容量约占2k空间, 查寻的步骤是将BCD码表示的幅度值变成二进制数, 作为查表地址。从EPROM的地址中取出对应的允许起重量送数据区暂存。利用查表法使单片机避开了复杂费时的数学运算, 提高了单片机的运行速度。

3.2 计算方法 程序运行中要计算0.9倍和1.05倍允许起重量, 若用二进制数做定点运算, 结果是很费时的。为了简化运算, 改用

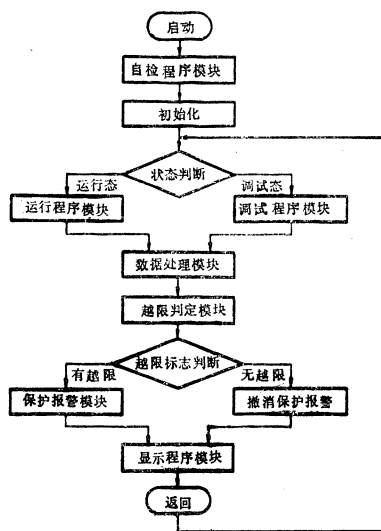


图 2

采用新技术(PC)改造旧磨床

河北大学 张秀玲 张怀安

(河北省保定市合作路1号)

摘要 本文以改造3MZ-1310自动磨床为例,介绍了可编程控制器(PC)的工作原理和特点,并给出了程序流程图。

1 引言

目前机械行业仍有许多继电器控制的各种机床,其控制箱庞大,耗电量高。工作效率低易出现故障。寿命短且不利于维修。有的设备虽经改造采用了单板机单片机控制,但其抗干扰能力差,不适于在潮湿、高温的恶劣环境下工作,并且程序不容易修改,改造周期长,不利于推广应用。

引入PC新技术改造旧机床对现实生产有着非常重要的意义。它具有编程方便可靠

BCD码定点运算来代替。思路是将乘数扩大 $10N$ 倍,两数相乘后将积缩小 $10N$ 倍,这样就得到了带小数的乘积。

3.3 数据传送与处理

A/D转换器与单片机之间的数据传送采用外中断服务程序来完成。由于运用多路数据采集电路,所以就存在模拟电子开关、运算放大器的切换速度与A/D器件速度的匹配问题。为了达到正确测量的目的,除了软件上使A/D器件强迫复零之外,在数据处理上将四次采样值舍去最大最小值后取平均作为测量值,在实地调试时数据相当稳定。为了使单片机对各个保护要求作出有条不紊的反应,在保护执行模块中安排了保护报警处理的优先级别,由单片机按级别的高低顺序查询,统一执行保护动作。

4 结束语

最后还要提一下如何解决仪表的抗干扰

性高等特点，特别适用于传统工业改造和新产品的设计。目前在国内外已拥有广阔的市场。

2 PC机简介

可编程控制器 (Programmable Controller) 简称PC是以计算机技术为基础专为工业环境下运用而设计的自动控制设备。由输入、逻辑控制、输出三大部分构成。输入部分用来采集并保存现场信息, 为了提高抗干扰能力, 采用了光电隔离措施。逻辑控

问题。由于起重机的主电机在起动中电流值的变化很大,建筑工地上电源线往往又长,因此电网电压大幅度跌落以及各种交流继电器、接触器频繁动作给共网运行的仪表造成很强的干扰。为了解决仪表的抗干扰问题,采取了几个措施。信号输入回路设计了一个二阶有源低通滤波器,滤除从信号端引入的干扰。主机与显示箱之间采用了专用接口电路,引线加以屏蔽。主机电源侧加了LC滤波器,铁制机壳可靠接地。模拟地与数字地远离且在一点处汇合。程序设计时在各模块间安排软件陷阱。采取了这些措施后仪表在现场运行中性能稳定、工作可靠。

参考文献

- 1 徐爱卿等著. 单片微型计算机及其应用. 北京航空学院出版社
- 2 [英国] 克莱顿著. 数据转换器. 上海科技文献出版社

制部分处理取得的信息,并判断是否需要输出。输出部分根据主机的“命令”控制外设哪些需要操作。PC机的工作是靠事先编制的程序控制的。程序设计采用了不同于其它计算机的梯形图语言,其控制符号(指令)的定义与传统的继电器表示符号完全一致,其程序与继电器控制逻辑互相对应,便于广大工程技术人员掌握。同时PC机还具有很强的编辑与调试功能,即当需要改变被控对象的内容时,只改变程序即可,无需重新配线,这是继电器控制无法比拟的。

PC机工作采取程序存储、顺序扫描执行方式,其过程见图1所示。

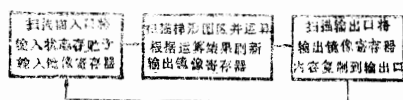


图 1

为了提高工作的可靠性,PC机采用了硬件与软件措施,设计了很完整的自诊断功能,它可随时检查机器本身有无故障,一旦出现

故障立即停机,不会产生误动作。综上所述,采用PC机控制各种机床,实现生产自动化是必然趋势。

3 磨床控制要求和控制系统的构成

3.1 3MZ—1310磨床根据磨削工艺要求,具有自动、半自动、调整三种工作方式。在自动和半自动时,又有记数整修和无整修功能可供选择。

3.1.1 自动状态

工作方式选择开关置于自动位置时,按压循环启动按钮,磨床开始自动循环,连续加工工件,直到按压复位按钮为止。其动作顺序由行程开关状态控制,其循环过程如表1。机床位置与行程开关状态的对应关系见表2。

3.1.2 半自动状态

工作方式选择开关置于调整位置时,按压各调整按钮,则完成相应的单步动作,其

表1 自动磨床循环工步表

工作程序 电磁顺序号		用途		开 车	上 下 料	磨 架 进	磨 架 超 进	粗 磨			精 磨	无 进 给 磨	磨 进 架 给 快 退	上 下 料
								机械手 跳出	上下料 复位	机械手 插入				
10CT	方向	×	×	×	×	—	—	—	—	—	—	—	×	×
11CT	无进给	—	—	—	×	×	×	×	×	×	×	—	—	—
12CT	粗精磨	—	—	—	—	×	×	×	×	×	—	—	—	—
1CT	工作台补偿	—	—	—	—	—	—	—	—	—	—	—	—	—
2CT	上下料	—	×	×	×	×	×	×	—	—	—	—	—	×
3CT	磨架进退	—	—	×	×	×	×	×	×	×	×	×	—	—
4CT	修整	—	—	—	—	—	—	—	—	—	—	—	—	—
5CT	机械手出入	—	—	—	—	—	—	×	×	—	—	—	—	—

按钮与单步动作关系如表3所示。

综上所述 3MZ—1310 磨床动作流程如

图2所示。

3.2 控制系统的构成

整机控制系统由EX—40PC机和外部按钮、调整开关、行程开关电磁阀等设备组成，

共24个开关量输入与PC机输入口数目相等，需要控制的有电磁阀、接触器等共13个。PC机有输出口16个这样能满足要求。

表2 行程开关代号和机床位置对照

1xk	2xk	3xk	4xk	5xk	6xk	8xk	9xk	14xk	15xk	18xk	19xk
上下料机构复位时压	上下料机构上料时压	磨架退极压	工作台移到左面压	磨床至前压	修整器正行程极限压	上下料机构跳出时压	工作台在工作位压	粗进给完压	精进给完压	进给油缸复位时压	快速超进保护

表3 按钮与单步动作关系

控制按钮代号	1AN	3AN	5AN	1K		2K	3K	4K	5K	6K	7K	8K
机床动作	长停	电机启动	循环启动	方式选择		上下料	粗进给	上磁	磨架进	工作台左程	修整	修整选择
				自动	调整							
相应PC机入口	X ₂₇	X ₀	X ₁₅	X ₁₆	X ₁₇	X ₂₀	X ₂₁	X ₂₂	X ₂₃	X ₂₄	X ₂₅	X ₂₆

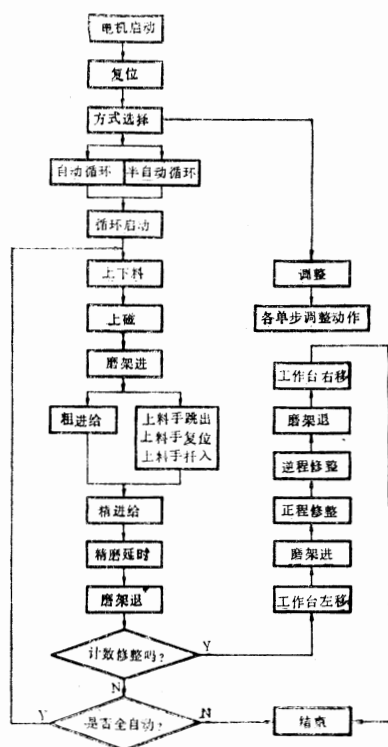


图2 3MZ—1310磨床工作流程图

4 程序设计

程序是控制部分的核心，是指挥机床动作的司令部。PC机程序设计采用的是梯形图语言程序，所谓梯形图程序就是以计算机软

件来代替传统的继电器方式所形成控制系统，其设计方法有以下几种：a.逻辑设计法。b.继电器控制电路翻译法。c.步进顺序控制法等。3MZ—1310控制方式采用了步进顺序程序，条理清楚，与实际动作相接近，易于掌握，且调试方便（梯形程序图略）。

5 结束语

PC机控制的磨床已在河北轴承厂投入运行。一年多的实践表明：整机性能稳定可靠，抗干扰能力强，便于维修和故障处理，由于程序设计合理使生产效率提高了10%左右，这些是原传统的继电器控制所无法比拟的，显示了PC机的强大生命力。

现在世界上每年都有新的PC产品推出，且销售额成倍增长。PC机不仅能控制开关量、数字量、还能控制模拟量甚至能连网和通讯。因此在机械、化工、冶金、纺织等各个领域有着广泛的推广价值，将对推进我国工业现代化进程起到巨大的作用。

参考文献

- 1 EX—40型PC机使用手册。
- 2 朱绍祥，张宏生编译可编程序控制器PC原理与应用等。

分层次结构化记述方法HCP图的应用

哈尔滨电子计算技术研究所 王升超

摘要 分层次结构化记述方法HCP(Hierachical Compact description chart)图,在许多层次结构化程序设计中起着重要的作用。本文就如何正确使用HCP图的方法作了详细介绍。

1 前言

在以往的程序设计中,程序设计人员经常使用的一种古老的程序设计流程图,这种古老的程序设计流程图简单易懂。已被广大的程序设计人员所接受,但随着计算机应用的发展。这种古老的程序设计流程图已适应不了分层次结构化程序设计的思想,无论在程序结构化的设计和层次化的描述,以及在程序模块之间的连接和程序测试方面,都难以胜任。而分层次结构化记述方法HCP图,则恰恰在这方面显示出应有的生命力。HCP图的程序设计记述方法,在日本电气电讯有限公司(简称日本NTTI公司)使用多年,已被该公司定为标准的程序设计记述方法。HCP图特别是在大型工程设计的系统分析和系统测试中起着重要的作用。笔者在一些大型的管理信息系统的程序设计中使用了HCP图的方法,受益非浅。现将使用方法和应用实例做以介绍。

2 HCP图的记述方法

HCP图的记述方法简单易懂,如表1所示。

3 HCP图与程序流程图表现方法比较

为了使程序设计人员更好的理解HCP图的使用方法,现将HCP图与程序流程图表现形式做一下比较,如表2所示。

4 应用实例比较

例题对于输入数据的奇偶累计的程序设计,见图1,图2。

5 结束语

从上述实例比较可以看出,HCP图的结构层次清晰,并且画法简单,上述实例只是一个简单的例子,如果描述更为复杂的程序,HCP图则将显示出应有的优势,特别是在大型的信息系统的系统设计与系统测试中,HCP图将起着重要的作用。

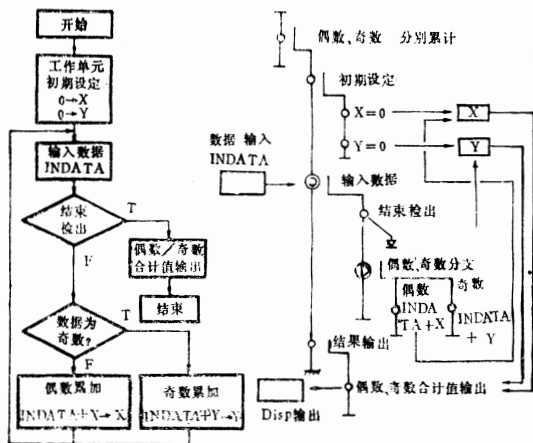


图1 程序流程图

图2 HCP图

表1















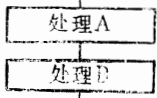
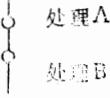
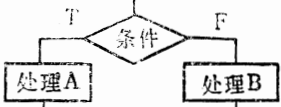

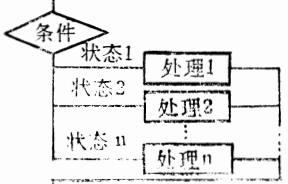
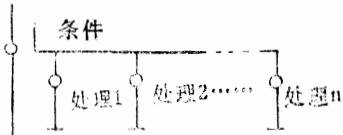
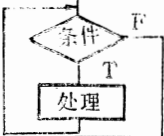
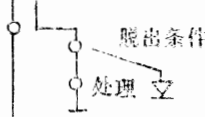
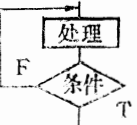

分类	用途	记号	功能	注释
处 理	一 般 的 处 理	 XXX	通常处理	XXX为功 能说明
		 XXX	循环处理	XXX为功能说明
		 XXX () ()	分支处理	“() ”为分支条件
		 XXX	检查出错处理	XXX为功能说明
	模 块 化 处 理	 XXX	调用外部模块	XXX 为外部模块名
		 XXX	调用宏指令	XXX为宏指令名
		 XXX	分页符号	XXX为功能说明
控 制 处 理	条件控制	 ()	条件判别	“() ”为条件记述
	开 始		程序开始符号	
			实现方法开始	
	终 了		分支结束返回	
		 9	返回数字所标层次	“9” 为返回的层次
			程序结束符号	
			错误出口	

表2

分类	程序流程图	HCP图
顺序		
判断		
选择		
控制 返回		
		

IBM-PC 系列机与多台MCS-51 单片机间的通信

天津冶金局职工大学 尹蓉华

(邮编 300210)

1 引言

应用IBM-PC系列微型计算机和多台单片机构成小型集散控制系统在一定的应用范围内是最经济的可行方案,已被广泛采用。不过,在众多的这类应用系统中,多是作为上位机的IBM-PC系列机(以下简称主机)定时扫描以单片机为核心的智能化控制器(以下简称从机)以采集数据或发送控制信息。而我们遇到的问题是,要求用一台主机控制几十台从机(即从机数量较多),并且主机除了定时呼叫从机、向从机发送信息外,更多的工作是处理大量从机频繁、随机的中断,接收各从机发来的信息,并在从机可以容忍的等待时间内发送给从机其所索要的信息。因为从机数量太多,如何提高主机的响应时间就成了一个必须解决的问题。

按照多CPU并行工作可以提高工作速率的设想,我们在主机和从机之间增加了一个由MCS-51单片机构成的中间控制器(以下简称中控站)。这样构成的系统如图1所示。

其各部份的功能如下:

从机独立完成一些控制、数据处理、报警等任务,定时接收主机传送来的控制信息,随时将现场信息不定时地传送给主机并向主机索要为完成控制和数据处理等各种任务所必须的参数和数据。

中控站分担主机的部份工作,主要接待大量从机频繁、随机的中断申请,将其存入先进先出存储器。按照先联络先通信、后联

络后通信的原则,组织从机和主机的通信。

主机除定时向各从机发送控制信息外,更重要的是按照中控站的安排为从机完成各种控制和数据处理任务提供必要的数据和控制命令;将从机传送来的信息形象地显示在CRT上,并加工处理和打印成各种报表。

本文主要介绍中控站的硬、软件设计。

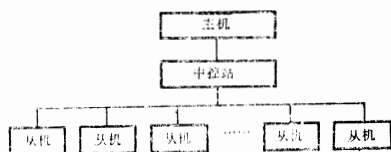


图1 由IBM-PC系列计算机集中管理的小型集散控制系统

2 通信控制器(中控站)的硬件设计

从抗干扰能力和经济性考虑,整个系统的通信采用了串行方法、基带传输方式。

中控站的核心部件是一片8031单片计算机芯片,它与主机的8088(或80286)芯片构成双CPU,8031分担和组织主机与各从机间的通信。

中控站的一端(8031芯片的串口)要与多个从机相接,因而另一端通过8251A芯片来扩展8031串口与主机相连。中控站工作原理如图2所示。

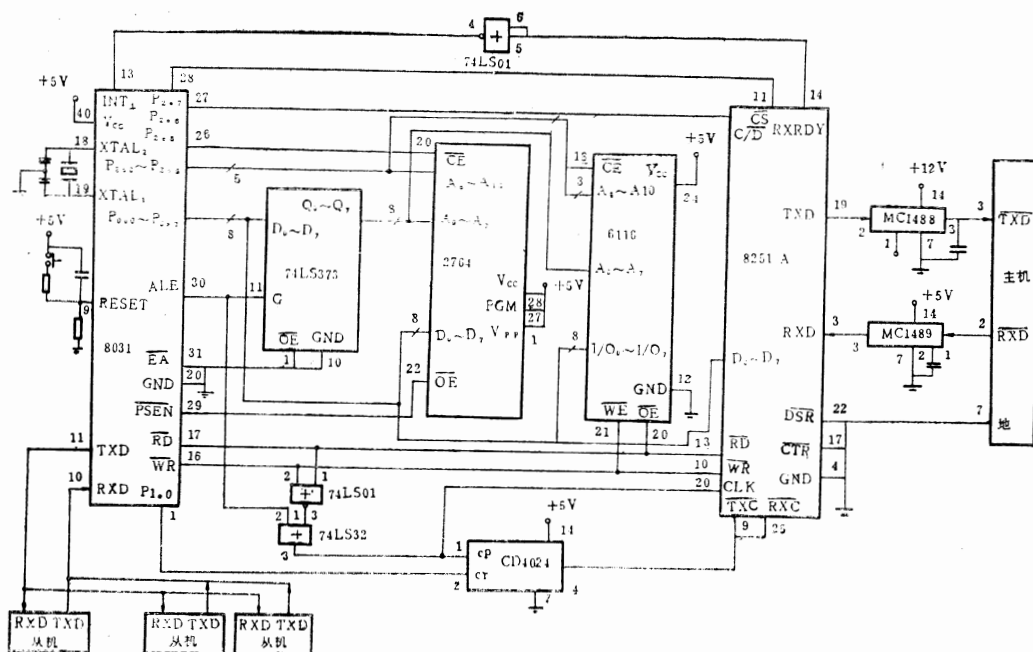


图2

8251A芯片的接收数据线RXD(3脚), 及发送数据线TXD(19脚)通过1488和1489芯片与主机相连。这是因为8251A的输入、输出均为TTL电平, 故需通过MC1488将TTL电平转换成RS-232C标准电平、用MC1489把RS-232C标准电平转换成TTL电平才能与主机的RS-232C标准串口连接。

8251A芯片的时钟输入线CLK(20脚)为8251A提供定时信号。在异步方式时要求CLK的频率至少要大于8251A内接收器或发送器输入频率的4.5倍。引脚RXC(25)为接收器时钟, 用于控制字符接收速率。引脚TXC(9)为发送器时钟, 用于控制字符发送速率。在异步方式中RXC及TXC(即接收、发送时钟)为传输波特率的1、16或64倍。

现设定传输波特率为1.2KHZ, 设置方式控制指令中波特率因子: D1D0为10, 故要求接收器和发送器时钟信号为传输波特率的16倍, 即19.2KHZ。因设置8031的时钟为7.3728MHZ, 故由8031的ALE、WR、RD

引线组合后产生1.2288MHZ的脉冲信号, 该信号作为8251A的CLK时钟信号。此信号经64分频后为19.2KHZ, 作为8251A接收器和发送器的时钟。CD4024为分频器。8251A的C/D引脚接8031的P2.6, 片选线CS接P2.7。这样, 8251A的控制字和状态寄存器地址为7FFFH, 数据缓冲器地址为03FFH。

通信控制器(中控站)中还配置了一片2K×8位的静态随机存储器芯片(6116), 芯片的容量是根据实际传输的数据量计算确定的。如果传输的数据量较少, 只用8031内部RAM即可, 而无需再用此扩展RAM芯片。

在扩展RAM芯片上开辟了三个容量不同的先进先出存储区。当多台从机几乎同时呼叫主机要求通信时, 若主机正闲, 则与最早呼叫的从机通信。而中控站则将其余从机的编号和发送过来的信息按其呼叫的先后次序及不同的数据标帜分别存入应该存入的先进先出存储区; 主机一旦空闲, 便按照中控站的安排将相应存储区中的信息取走, 处理

其余从机的中断请求。这样不仅提高了主、从机通信的响应速率,也实现了从机先呼叫先通信、后呼叫后通信的要求。

3 通信控制器(中控站)与从机之间、主机之间通信的有关约定

通信有一定的规范,不同的应用场合有不同的通信协议。中控站与从机及主机间通信的约定如下:

3.1 每台从机的编号即为该机地址。

3.2 无论是中控站呼叫从机还是从机呼叫中控站,联络前均置SM2=1,并且传送数据前必须先发送地址帧。地址帧前八位是从机地址,第九位数据(SCON中的TB8)为1;发送数据或控制帧时,第九位数据(SCON中的TB8)为0。

3.3 从机接收到的地址与本机地址相符时,则令SM2=0,不相符时保持SM2=1。中控站发出地址帧且确认某从机已接收后,或接收到某从机发来的地址帧又返送回去之后,置SM2=0,并要对以后接收到的从机数据的第九位(即RB8)作判断,当RB8=0时,接收该数据;RB8=1时丢弃该数据。

3.4 约定FFH为复位信号、00H为正确信

号。

3.5 主机向中控站发00H信号表示主机空闲,可以从中控站存储区中取信息;发01H信号表示主机要通过中控站向从机发送数据。

3.6 中控站向主机发11H信号表示请主机取存储区1中数据;发12H信号表示请主机取存储区2的数据;发13H信号表示请主机取存储区3中的数据。

4 通信控制器(中控站)的程序功能及流程图

中控站中8031芯片的全双工串行口用于与多台从机的串行口相连,如图2所示。

8031的串行通信有四种工作方式,由串行口控制器SCON设置。方式0发送/接收数据为八位,主要用于I/O扩展及同步方式的数据通信。方式1属异步通信规程,通信波特率可变,发送/接收数据为十位,主要用于点对点的单机通信。方式2和方式3发送/接收数据为十一位,发送信息的附加第九位数据是串行口控制器SCON中的TB8(D3位),属异步通信规程:常用于多机分支线路的通信系统。SCON寄存器是位可寻址的寄存器,其格式为:

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
-----	-----	-----	-----	-----	-----	----	----

↑

SCON.7

↑

SCON.0

SM0、SM1用于方式选择。2用于多机通信,可根据需要设置。REN为允许串行接收位,允许接收时由软件置位,否则用清零来禁止中断。TB8和RB8用于方式2和方式3。TI为发送中断标帜,由硬件自动置位,但需由软件清零。RI为接收中断标帜。串行中断由硬件产生。CPU执行一条写入SBUF的指令就启动发送器发送信息。当一帧数据发送完成后,产生发送中断,标帜位被置位。在方式3时,接收完一帧数据后,若RI=0,且

SM2=0或接收到的第九位为1,则标帜位RI被置位,产生接收中断;否则接收到的信息将丢失,RI不置位。所以,通过判断RI是否被置位,就可以随时监听从机是否发送来数据。

必须对中控站的串行口进行初始化,根据8031的主振频率和通信波特率确定定时器/计数器T1的计数器初值。中控站的 $f_{osc}=7.3728\text{KHZ}$,波特率 $B=1200$,采用定时器/计数器T1的工作方式2,取SMOD(波特率

选择位) = 0, 则T1计数器初值 (装重载值) 为F0H。串行口采用工作方式3, 关闭串行口中断。根据工作要求和通信协议, 中控站串行口的初始化程序编制如下:

```
MOV PCON, #00H
MOV TL1, #0F0H
MOV TH1, #0F0H
```

DSR	SYNDET	FE	OE	PE	TXE	RXRDY	TXRDY
-----	--------	----	----	----	-----	-------	-------

收数据前必须对它初始化。即由8031CPU装入方式控制指令和命令指令。

命令字中RXE=1 (接收开放) 时, 接收器接收到字符后, 状态寄存器的RXRDY位置1, 表明接收器已从RXD端接收到一个字符, 并已准备好将其发送给CPU。此时, RXRDY引脚由低电平变为高电平, 即为接收器中断请求信号。

发送缓冲器空且发送准备好时, 状态寄存器的TXRDY位置1, 命令字中TXEN=1 (允许发送) 并CTS=0, 引脚TXRDY由低电平变为高电平, 发送中断请求信号。

中控站通过查询8251A状态寄存器中D1位的状态, 或根据有无中断请求信号随时监听主机是否有信息发来。8251A芯片与主机的通信采取异步方式, 波特率B=1200, 传送字符长度为八位、奇校验、两位停止位。其方式控制字为DEH, 命令指令为15H。

8251A的初始化程序为:

```
MOV DPTR, #7FFH
MOV A, #0DEH
MOVX @DPTR, A
MOV A, #15H
MOVX @DPTR, A
SETB IT1
SETB EA
```

用软件设置先进先出存储器的思想是: 开辟一定容量的存储区, 设置两个指针, 分别指向存储区中所存数据的队头和队尾。指针RP指向所存数据最后一个数据的下一个空位置, 表示下一个要存入数据的存储位

```
MOV TMOD, #20H
MOV SCON, #0F8H
SETB TR1
```

中控站中8251A的内部有四个寄存器, 它们的工作状态寄存于其状态寄存器中, 可由8031CPU读取。状态字的格式如下:

8251A的工作方式由程序设定, 在发送/接

置。指针FP指向存储区最前面的一个数据, 表示要取出的数据位置。存入一数据时, 是存在RP当前指向的位置, 而后RP加1。而取出一数据时, 是从FP指针当前指向的位置取, 然后FP加1。初始时, (RP) = (FP) = 0, 它们都设置在存储区的起始端。存入一数据时, RP加1, FP不动。当RP指针指向存储区末端时, 即RP增加到存储区长度时, 将指针值置到初始位置处。同理, 每从存储区取走一数据, FP加1, 当FP增加到存储区长度时, 也将其置到初始位置处。RP指针每次移动都做一次判断, 若RP指针移动后, (RP) = (FP), 表明存储区已存满, 此时该区的溢出标帜置位。溢出标帜置位时只能从存储区取数据而不能存入数据。本通信控制器根据实际需要扩展了一片2K的RAM芯片, 设置了三个这样的先进先出存储区, 分别存放三种数据。

中控站还可以代替主机做部分工作, 如检查各从机工作是否正常等。这类工作是以中控站顺序地向各从机发地址帧的查询方式进行的, 其流程如图3所示。

本系统中无论是主机主动呼叫从机进行通信, 还是从机主动呼叫主机进行通信, 都要通过通信控制器。

我们的课题中, 主机主动呼叫从机时, 容许的响应时间较长, 故采用主机对从机查询的方式。通信控制器一旦接收到主机要查询的从机地址就立即呼叫相应的从机。被呼叫的从机接到呼叫信号后立即返送它的本机地址给中控站, 并置其SM2=0。中控站

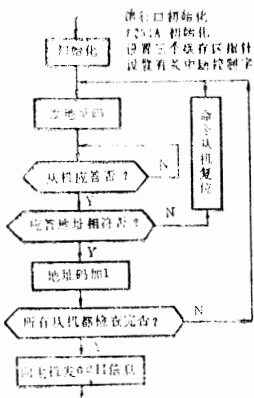


图3 中控站检查各从机是否正常的流程
比较收到的地址与发出的地址相符时,置TB8=0、SM2=0,而后才开始传送数据。从主机处接收到一帧数据就传送给从机一帧数据,直到传送完全部数据并检查代码和正确为止。这部份的流程图见图4。(每传送一批数据都要计算各帧的数据和—称为代

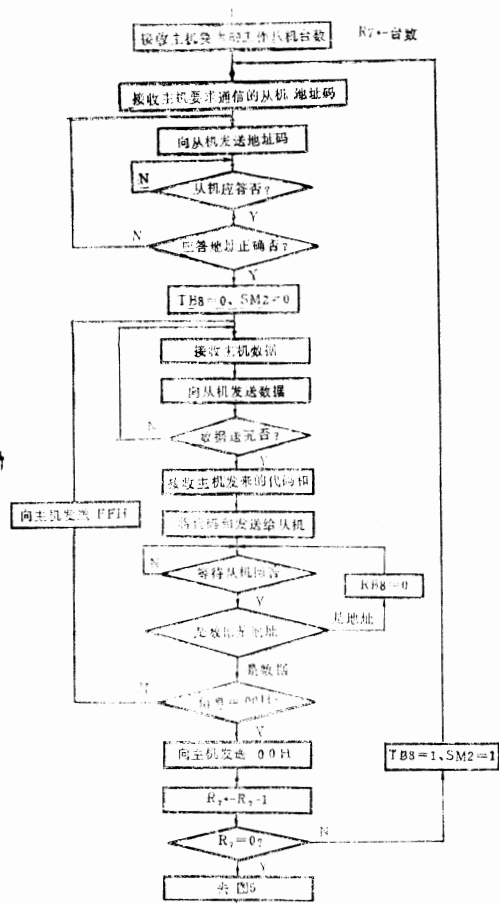


图4

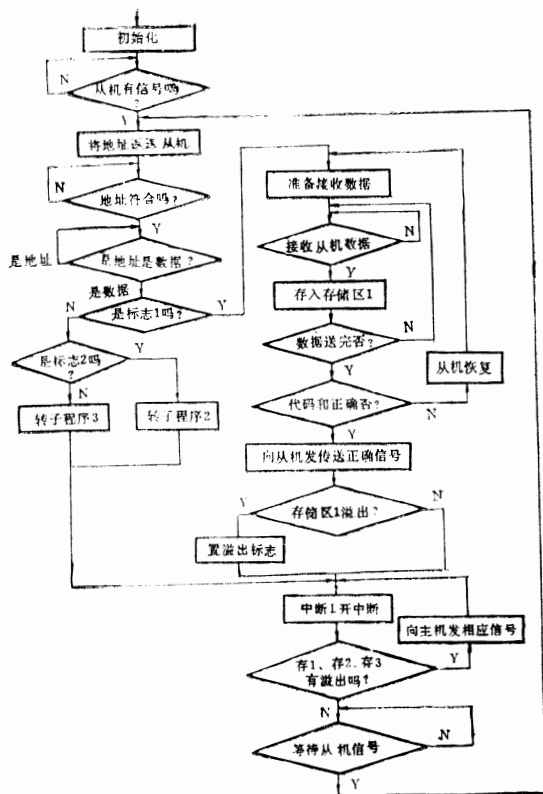


图5

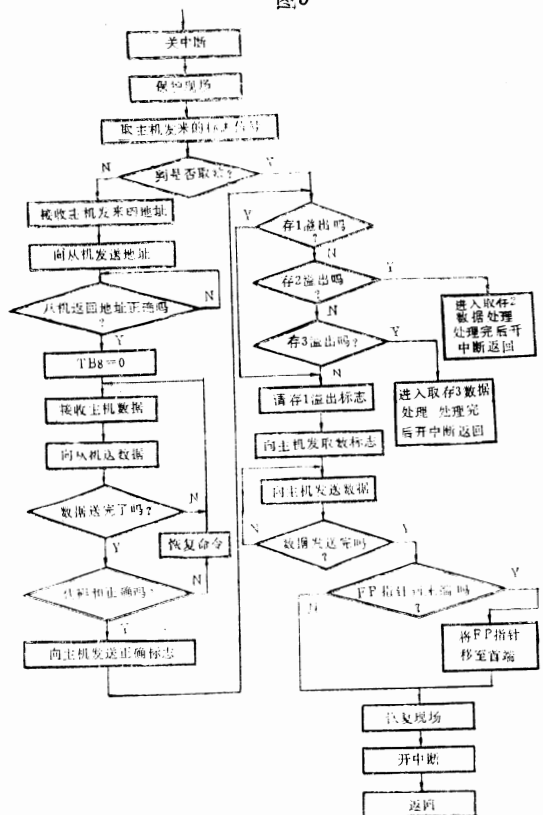


图6

微机控制监测系统中IBM—PC与STD总线间数据通讯

郑州纺织工学院 张瑞祥

(郑州中原西路41号 邮编 450007)

摘要 本文主要介绍用微机监测冷冻站的温度、流量和压力等参量的控制系统中,完成作为上位机的IBM—PC与下位机STD总线工控机之间的数据信息交换,而实施的异步串行通讯的应用。

STD总线是一种工业标准微机总线。它具有高可靠性,适应在环境恶劣的现场工作;又实时性强,对目标的采集和控制简单直接,减少了不必要的中间层次和转换界面;且采用标准模块化,灵活性好,易于进行功能扩展和更新。我们在对湖北云梦棉纺厂冷冻站进行微机监测的研究中,考虑到该

厂地处山区,工作环境恶劣,而采用STD总线工控机作为下位机完成对站中各参变量的巡回监测任务,它与IBM—PC上位机组成多级系统,以完成控制系统的微机智能化。很多生产信息管理系统是运行在PC机上的,为提高系统功能,用户希望能把生产数据如温度、压力等通过传感器直接送到计算

码和,代码和正确表示传送无误,否则要重新传送)。

相反,从机主动呼叫主机时,却希望响应越快越好。为使多台从机呼叫主机时各从机都能在可以容忍的等待时间得到主机的响应,就必须严格按照各从机呼叫的先后次序排队。通信控制器即是把各从机发送来的数据按照其数据标帜存入不同的先进先出存储区,等待主机处理。中控站的主程序流程示于图5。主机一旦空闲就向中控站申请中断,索要待处理的从机信息。通信控制器的中断服务程序框图示于图6。

5 从机通信部份程序框图

从机要完成控制、数据处理、报警等多项工作,本文从略。这里仅画出与中控站通信部份的程序框图,示于图7。

6 主机的通信程序设计

主机与通信控制器的通信程序框图见图8。主机要做的工作也很多,与通信无关的部份从略。

7 结束语

本设计已用于实际系统。运行情况良好并改善了系统功能,使多台从机几乎同时呼叫主机时各从机都能够有比较合理且可以接收的等待时间。

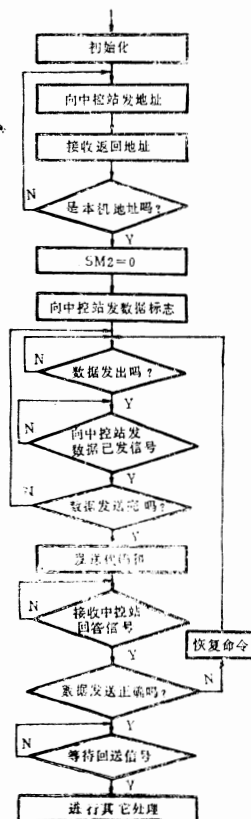


图7

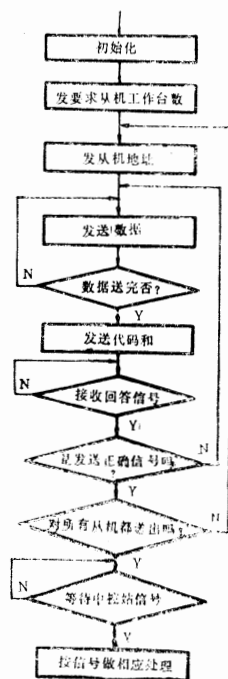


图8

机内,而不通过键盘人工键入。我们采用把传感器的信号先送给STD总线及有关模块,它们做些简单运算,存贮等工作,待PC机需要这些信息时,再由STD总线送给PC。这样做是将监测传感器的任务交给了STD总线,PC机在不需要数据时,可做其它工作,甚至可以关机。此方法提高了PC机使用效率,增强了系统的灵活性;其次STD总线扩展方便,易于扩充连接的传感器数量和类型。较将传感器直接与PC机相连的方法更显得优良。

1 系统软硬件设计

STD总线与PC机间的通讯沟通了系统所需数据的获得,而这里的异步串行通讯需满足下列基本条件:

a. 通讯过程中,人不需要干预STD总线模块的工作。如通讯过程中不需要通过人工启动STD总线的通讯程序。

b. STD总线和PC机间通讯连线简单可靠。

c. STD总线和PC机通讯至少要半双工,因为PC机与STD总线之间要互传信息。

基于上述条件,我们选用了北京计算机配件五厂生产的SC—13030主机板作为STD总线的控制处理中心。该板采用HD64180C MOS八位多功能MPU,自身拥有2通道异步串行通讯器,并开发一个RS—423A 通讯口。这两个通道是完全独立全双工的,每个通道具有完全独立的可编程能力。其中有两个控制寄存器用来完成操作方式选择,奇偶校验和波特率等通讯协议的设定;状态寄存器反映的则是通讯过程和出错情况等状态标志,CPU可根据对标志的判断,来保证通讯的顺利进行。通讯口RS—423A 与 RS—232C全兼容,因此STD总线和PC机无需进行硬件改动,就可直接相连。根据我们对通讯操作的要求,其简单连线如图1。

由于控制系统要求频繁进行上、下位机间数据传送,且对巡回监测周期实时性要求

不太高,所以我们在对STD总线通讯程序的编制中以一段时间的查询作为与PC机的联络手段,如果其间PC机有通讯要求,则STD转入对应通讯程序中执行;反之进行巡回监测等工作。这种方法较中断法易实现且简单。STD总线通讯程序框图如图2所示。

RS—423A信号 RS—232C信号

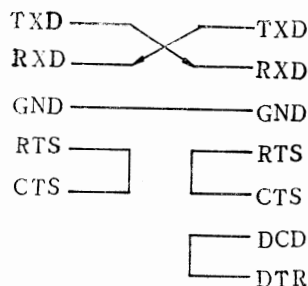


图1

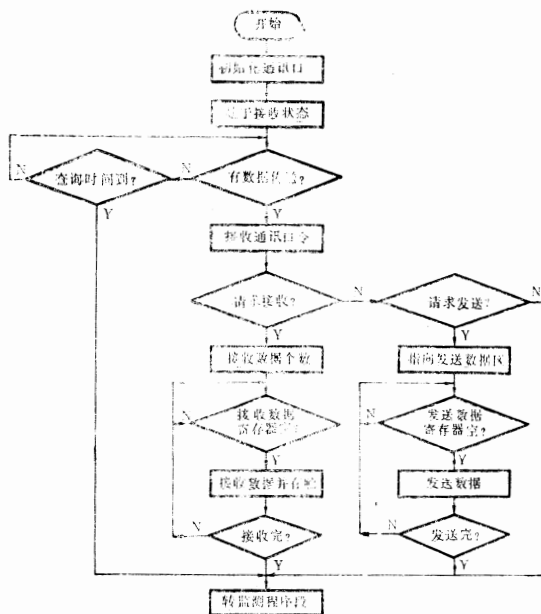


图2

上位机PC的通讯程序可采用两种语言编制,即a.用BASIC语言进行的高级编程方法,它对PC机串行口初始化以文件形式打开通信信道:

OPEN "COM1; ..." AS #1

用这一条程序就可完成波特率、奇偶校验和数据位等的设置;高级编程还能用数组形式为接受或发送的数据提供栈区。b.用8086汇编语言的低级编程方法,它初始化8250芯片

时,至少要对其中的4个寄存器顺序设定,才能确定波特率、数据位等通信协议。虽然两种语言的编程各有特点,但其编程依据却是一样的,PC机的通讯程序框图如图3所示。

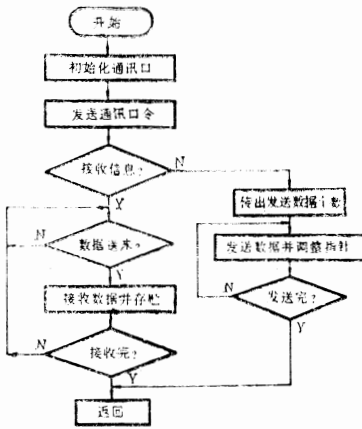


图3

在通讯程序中要加口令检验是因为要避免由于通讯线路中有些干扰,它们会引起误动作,而对特定的传送字符串口令进行核对,从而判定是否为真的发送或接收数据,以保证通讯顺利进行。

2 调试运行

通讯程序的编制虽不易,但对编好的程序联机调试也不是一蹴而就的。首先遇到

是由于上、下位机设定的通信协议不匹配,造成通讯数据误码率很高,而使通讯失败;所以通信协议的一致是保证异步串行通讯成立的先决条件。用GW BASIC版本单独运行PC机通讯程序中的接收或发送功能段时可顺利执行,同时由数据接收转为发送也能通过;但由发送转为接收数据时却无法运行。后改用BASICA版本运行时就没有了上述情况。单纯将数据接收和发送功能联接循环运行时,会出现下位机发送一串数据而立即转为接收状态时,上位机总有一个或几个数据无法收到。后查出原因是下位机将数据写入发送数据寄存器后还未移入发送移位寄存器就转为接收状态,致使发送数据暂停,上位机接收不够。于是在发送功能段后加一段延时,就保证了数据全部发送出去。经过不断完善,最后我们使上、下位机通讯按设计要求运行,保证了数据的传送。

参考文献

- 1 “HD64180/Z64180用户手册”北工大出版社
- 2 “SC-13030主机板说明书”.北京计算机配件五厂研究所
- 3 陈学谦等译.“IBM-PC编程指南”.电子工业出版社
- 4 高传善等编.“接口与通信”复旦大学出版社