第3卷 第3期 光学 精密工程 Vol. 3, No. 3
1995年6月 OPTICS AND PRECISION ENGINEERING June, 1995

分布式控制系统的软件同构化设计

汤建华 白超光 姜 弢
(中国科学院长春光学精密机械研究所,长春 130021)

摘要 结合工程实际从系统结构、数据结构、信息接口,软件运行控制与调度逻辑,处理时序等方面介绍了一个分布式控制系统的软件同构化设计方法。

关键词:同构;软件设计;多重循环网络;分布式控制系统

1 引 言

从程序设计发展主流和历史趋势来看,大致经历了三个阶段,即终止于60年代的非结构程序设计;发展于70年代的结构化程序设计;发端于90年代初的同构化程序设计[1]。软件同构化程序设计目前正处于发展阶段,其含义尚无共识。本文从多机分布式控制系统整体出发,在充分考虑各单机性能的基础上,将各单机软件的系统结构、数据结构、信息接口、运行控制与调度逻辑,处理时序等五个主面设计成相同(似)的,从而进行多机软件的同构化设计。这种方法的优点是软件总体结构清晰、多机资源共享,研制开发周期短,便于系统故障时的软件功能实时重组,软件总体联调容易。

2 系统简介

本文所说的分布式控制系统如图1所示,它由四台结构相同的486/33机组成,任意两机之间采用双向并行通讯方式进行信息传递,是一个全互连的分布式控制系统。四台主机在时统中断信号的作用下按20Hz的频率协调同步工作,每个50ms处理周期内,各单元机需实时完成外部信息串行接收、数据分析处理,内部信息并行传递,控制决策输出,信息实时显示与记录等任务。

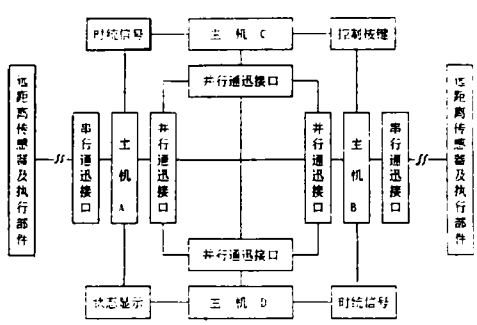


图1 分布式控制系统结构框图

该分布式控制系统采用故障冗余设计,在多机硬件同构的基础上,可实时进行系统的软

收稿日期:1994年9月10日

件、硬件功能重组，提高系统工作的可靠性，针对这一特点，我们采用了如下的软件同构化设计方法。

3 系统结构同构化设计

3.1 单元主程序同构设计

系统结构同构化设计要针对各单元机的主程序结构而言，其同构化设计如图 2 所示。

采用相同的主程序结构，使得整个系统的软件总体结构清晰，利于多机软件同步协调执行，方便系统联调。在程序设计上各机可以结构共享，即四个单元机共用一个同构的主程序，功能上的差异由所调度的功能模块决定，而调用逻辑是相同的(功能模块调用网络的同构化设计在后详述)，从而可用相同的程序结构完成不同的软件功能，拓宽程序设计的开发利用、缩短研制周期。

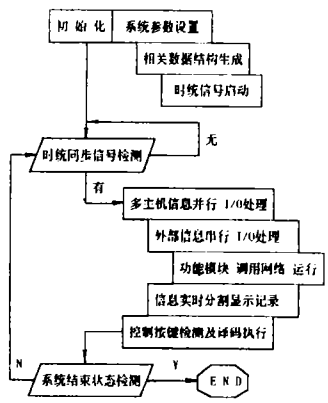


图 2 主程序结构同构化设计

3.2 多机软件帧处理时序设计

由于本系统是以 20Hz 的频率进行实时帧信息处理的，信息交换采用定时查询工作方式，为此进行系统整体工作时序同步设计是保证系统主程序同构设计和正常运行的前提，提供 20Hz 和 100Hz 同步脉冲的时统信号是时序同步的硬件支持，各处理机均在时统中断信号作用下协调同步工作。系统帧处理时序如图 3 所示。

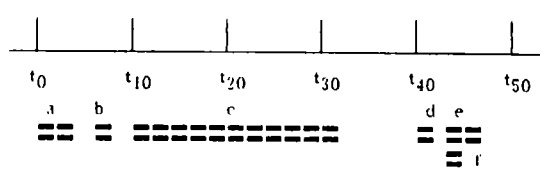


图 3 系统帧处理时序

其中 *a* 为并行 I/O 处理，约为 6ms；

b 为串行 I/O 处理，约为 3ms；

c 为功能模块调用网络，小于 25ms；

d 为信息高速记录，小于 3m；

e 为 CRT 屏幕实时分割显示处理，小于 5ms；

f 为单控制按键查询处理，小于 3ms；

g 和 *f* 由软件控制进行帧交替工作，整个系统的占空比约为 4∶1。

4 数据结构同构化设计

数据结构同构化设计是整个系统软件同构化设计的基础。

4.1 同构化数据的组织形式与层次划分

同构化数据的组织形式及与内存的结构关系如图 4 所示，从层次划分上分为四层：1. 信息义换通讯层(串行、并行)；2. 数据处理帧结构(动态链表)；3. 网络运行控制层；4. 信息显示

与记录层。

4.2 系统软件接口数据同构化设计

(1)系统与外界(传感器、执行部件等)以同构的帧格式进行信息串行 I/O 交换；

(2)分布式系统各处理机间以并行传输的同构数据文件格式进行实时帧信息交换；

(3)各功能模块间以公用变量进行信息传递(公用变量的读写逻辑由功能模块的节点调用逻辑控制)；

(4)网络运行控制及节点调用逻辑由网络标志变量、节点标志变量及网络连通表，路径迭通表等网格数据结构及相应的系统运行状态决定。

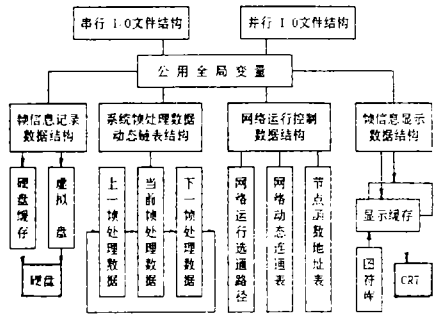
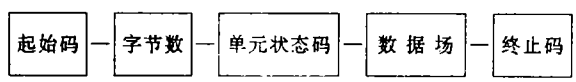


图 4　系数据的组织形式与层次化分

4.3 通讯信息的同构化设计

系统的通讯信息(串行、并行)以同构的数据文件格式进行实时的帧传输，其格式如下：

(1)单元状态码是一个 4 字节的数组，以位方式记录各单元的实时自检信息；



(2)数据信息记录在数据场中，是一个定格式变长度的二维数组，例如 pio－in[4][500]和 pio－out[4][500]角标 1 为处理机并行通讯端口，角标 2 为数据信息，如 A 机发往 B、C、D 机的数据按上述格式存入 pio－out[?][500]中，A 机从 pio－in[?][500]中接收来自 B、C、D 机的信息。

(3)同构的通讯数据格式使得该分布式控制系统可以设计出同构的通讯程序和系统状态检测程序(12 个并行输入程序＋12 个并行输出程序＋12 个并行检测程序＋6 个串行输入程序＋6 个串行输出程序＋6 个串行检测程序＝54)，所需的 54 个通讯检测程序，只需设计出 6 个(并入、并出、并检、串入、串出、串检)即可，其余 48 个可同构设计，程序设计量减少了 88%。

4.4 多重网络数据结构的同构化设计

本文功能模块的运行调度控制网络是在 ATN 网络和下推自动机的基础上，结合工程实际设计构造的一种多重循环网络，它分为主网络和子网络两大部分，每个网络由节点和路径两个基本要素组成，节点表达系统分析状态，路径控制系统分析进程，是一个有路径约束条件的多重有向图。网络间可递归调用，各节点函数调用相应的软件功能模块完成特定的功能，系统分析从主网络的起点开始，到终点结束。

网络数据结构的同构化设计是实现网络运行控制程序同构化设计的基础，百后者又是实现多机主程序同构的前提，进而实现整个系统的软件同构化设计。

网络数据采用 2 维数组，包括以下三种主要结构：

(1)网络运行迭通路径 phr－i－road[NETNUM][NETWIDE]

网络的拓扑结构由连接各节点间的路径唯一确定，用二维数组同构设计，其中

NETNUM 为 PHR－I 网络的节点数；

NETWIDE 为 PHR－I 网络中单一节点可转移的最大节点数；

phr－i－road[n][0]为节点 n 的节点号；

phr－i－road[n][i]为节点 n 第 i 条转移路径的节点号；

不同网络的结构关系由这种同构的二维数组唯一确定，包含了全部可能的网络运行状态。

(2)网络动态连通表 phr－i－table[NETNUM][NETWIDE]

网络的动态运行状态由该表唯一确定，它也用同构的二维数组表示，其中 NETNUM 为 PHR－I 网络的节点数，角标 1 与角标 2 均为该网络的节点号，例如：

phr－i－table[A_i][B_j]＝1 表示该网络的 A_i 与 B_j 节点间有一条动态可选通路径；

phr－i－table[A_i][B_j]＝0 表示该网络的 A_i 与 B_j 节点间此刻无选通路径；

不同网络的动态连通表由其网络运行选通路径数组唯一确定。

(3)节点函数地址表 phr－i－abd[NETNUM][2]

该二维数组记录 PHR－I 网络中所有 NETNUM 个节点函数的内存地址；

phr－i－add[n][0]为 PHR－I 网络第 n 节点函数的段地址；

phr－i－add[n][1]为 PHR－I 网络第 n 节点函数的偏移量；

4.5 同构化设计数据的 C 语言实现

该系统软件由多个文件及功能模块组成，任一单机软件都含有十几个文件，上百个功能模块，因此数据结构同构化设计除结构上合理外，还需要在整体上进行把握，数据同构化设计应满足 *a*. 可为多个文件方便调用；*b*. 避免重复定义；*c*. 公用变量共享。为此在用 C 语言实现时，根据 C 的特点和功能，对数据结构的定义作了如下处理；

- 所有的公用变量均采用标题文件(·h)方式定义；
- 标题文件分初始化和非初始化文件两大类；
- 初始化文件均在各主程序文件中一次调用；
- 非初始化文件在各程序文件中可多次调用；
- 四台主机间同名变量进行重名定义；
- 采用的主要 C 数据结构有数组、结构，指针及由它们组成的链表结构等。

5 系统软件运行控制与调度逻辑同构化设计

本系统的控制、调度逻辑采用多重网络技术实现，用相同的网络运行策略完成多机整体控制逻辑的同构化设计，在网络数据结构同构的基础上，网络运行策略同构化设计主要包括以下几个方面。

5.1 网络调度方式同构化设计

设网络调用标志变量为 phr，被调用的网络标志常量为 PHR－A，PHR－B，……，PHR－N，其中 PHR－A 为主网络；各网络调用结束标志为 PHR－A－E，PHR－B－E，…，PHR－N－E，则同构化设计如下：

```
while(PHR－A－E 元效)
    {while(phr＝PHR－A 且 PHR－A－E 无效)
        {PHR－A 网络节点调度逻辑函数 ttt();}
     ……
     while(phr＝PHR－N 且 PHR—n—E 无效)
```

{PHR－N 网络节点调度逻辑函数 ttt();}

5.2 网络节点调度逻辑函数同构化设计

网络节点调度逻辑函数 ttt(0)同构化设计是实现网络调度方式同构的基础。

设 mark 为网络节点调用标志变量；

phr－i－add[mark][0]为 PHR－I 网络第 mark 节点函数的段地址；

phr－i－add[mark][1]为 PHR－I 网络第 mark 节点函数的偏移量；

则网络节点调度逻辑函数 ttt()同构设计如下：

```
ttt(phr－i－add[mark][0],phr－i－add[mark][1])
{利用入口参数,得到 PHR－I 网络的第 mark 节点函数地址;改变程序指针,调用节点函
 数执行}
```

通过改变节点调用变量 mark，可使 PHR－I 网络遍历其内部的相关节点函数，从而完成该网络的一次调用。

5.3 节点函数同构化设计

系统的全部功能模块均通过相应的节点函灵敏来调用，为此设计了如下同构的节点函数结构。

```
phr－i－net－n()
{if(phr－i－table[n][phr－i－road[n][1]]==1)
    {调相应的功能模块函数 phr－i－nl()处理;
  mark=phr－i－road[n][1];return;}
……
if(phr－i－table[n][phr－i－road[n][NETWIDE]]==1)
  {调相应的功能模块函数 phr－i－nW()处理;
  mark=phr－i－riad[n][NETWIDE];return;}
```

通过 PHR－I 网络的动态连通表 phr－i－table[][];和运行迭通路径 phr－i－road[][]等网络数据结构，找出该网各节点 n 的相关路径，选择最优路径执行相应功能模块，并使节点控制变量 mark 指向该路径的另一端点，完成一次节点函数调用。

6 结　论

本系统所用到的软件同构化设计是以功能模块的网络调度运行策略为核心的，在数据结构同构的基础上，设计出了同构的网络运行控制程序，结合同构的时序设计完成了多机主程序的同构设计，进而实现了整个分布式控制系统总体的软件同构化设计。

软件的同构化设计应以系统结构，数据、程序等资源可共享为前提。采用上述的软件同构化设计方法使得本系统多机软件间可最大限度地资源共享，加快程序开发周期，也为软件在系统故障情况下的实时功能重组提供了可能。工程实践表明该方法是成功的。

参 考 文 献

[1]周启海，计算机同构化程序设计原理及其应用导论．北京：清华大学出版社，1993

[2]W. A. Woods, Transition Network for Language Analysis. Communication of the ACM, 1970, 13: 551～586

[3]汤建华，除近需，利用句法、语义循环递归网络实现汉语拼音→汉字转换．中文信息学报，1989, 4: 50～59

Homogeneous Software Design in Distributed Control System

TangJianhua, Bai Chaoguang and Jiang Tao

(*Changchun Institute of Opticas and Fine Mecanics, Chinese Academy of Sciences, changchun* 130021)

Abstract

In this paper, we introduce the application of homogeneous software design in a distributed control system. It is based on the following system structure, data structure, information interface, software running and scheduling, processing time sequence.

Key words: Homogeneous, Software design, Multi-cycle-metwork, Distributed control system