

基于 LonWorks 技术的通信电源监控系统的设计

李 坚^① 陈 兵 谢伟成 张世永^②

摘 要 本文介绍了一种基于 LonWorks 技术的通信电源监控系统。讨论了系统的结构及数据采集子系统的软、硬件的设计,给出了监测和信息管理子系统的软件流程图。

关键词 LonWorks Neuron 芯片 监控 数据采集 信息管理

随着我国通信事业的飞速发展,通信网的规模越来越大,其所要求的维护管理的质量和水平越来越高,为了改变传统的分散式以人工看管为主的维护管理体制,提高通信电源设备管理的水平,研制和开发通信电源设备集中监控系统具有重要的意义。近几年来,计算机工程公司、生产设备厂家、高等院校和科研院所相关专业、计算机集中监控系统专业厂商以及电信部门在电源设备集中监控系统这一课题上投入了大量的人力和物力,开发了具有不同特色的通信电源设备集中监控系统,其中主要是数据采集控制网络部分各不相同。我们与上海高校仪器设备有限公司合作,研制开发了“横浜通信综合楼通信电源监控系统”。该系统具有动态实时地对通信电源设备的运行数据进行监控、报表打印、对设备的告警监视、故障定位、运行参数的范围设置等特点。此外,当通信电源设备调整时,本系统还具有良好的组态性能。

一、系统结构

整个系统可分成二个子系统,一个是现场数据采集子系统,另一个是监测和信息管理子系统。每个子系统完成其特定的功能。见图 1 所示。

1. 现场数据采集子系统

现场数据采集子系统采用总线式拓扑结构,通信介质为双绞线。其间的通信协议采用的是 LonWorks 技术中的 LonTalk 协议,并由 N 个 LonMod01E 智能节点和 N 个 LonMod02E 智能节点组成。LonMod01E(02E)智能节点负责将电源设备中的模拟量(数字量)信号采集处理后,自动刷新 LonWorks 网络数据库(此数据库在 LonWorks 网络开发工具安装智能节点时自动生成)。

2. 监测和信息管理子系统

监测和信息管理子系统采用 Client/Server 结构,Server 运行数据采集服务器程序,Client 则利用中文 Windows95 操作系统和 Microsoft VC++5.0 编写的非常友好的人机界面

程序,为用户提供了非常便捷的下拉式菜单和树型目录,操作方式全部采用鼠标点击方式。Server 与 Client 之间的通信采用 TCP/IP 协议。

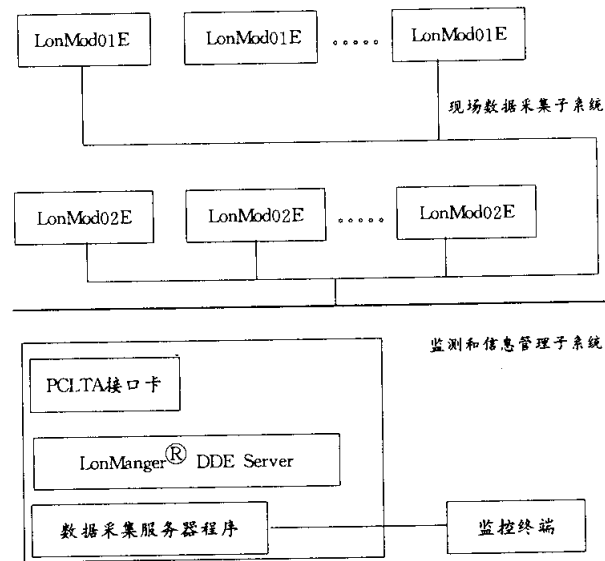


图 1 系统结构图

3. 子系统间的通信

监测和信息管理子系统与现场数据采集子系统之间的通信是通过 PCLTA-10PC LonTalk 适配器与 LonManger DDE Server 系统软件来实现的。PCLTA-10PC LonTalk 适配器装在监测和信息管理子系统的服务器中,使之成为 LonWorks 网络中的一个智能节点,进而可很容易地与该网络中的其他智能节点进行通信。

LonManger DDE Server 系统通过其配置文件中的设置来建立与 LonWorks 网络数据库之间的连接。而监测和信息管理子系统的数据采集服务器程序,则采用 DDE(动态数据交换)方式与 LonManger DDE Server 系统软件之间进行通信。这样,数据采集服务器程序就可以将 LonWorks 网络中的网络变量值(即模拟量和开关量)反映在人机界面上。

二、现场数据采集子系统硬件设计

LonWorks Networks 是九十年代最新的控制网络技术,广泛应用于各行各业的自动化监控系统,为实现智能控制网

① 李 坚 上海移动通信公司 工程师 上海 200081
② 张世永 复旦大学计算机科学系 教授 上海 200433

络提供了完整的解决方案。LonWorks 技术除了总线式网络结构以外,用户还可以选用任意形式的网络拓扑结构。网络通信介质可以是双绞线、电力线、光纤、无线、红外线等,并可在用一个网络中混合使用。网络中通信与控制的功能是由 LonWorks 技术的核心元件-Neuron 芯片来完成的。该芯片有三个 8 位的 CPU,分别用于介质访问控制、过程控制和网络

通信控制,直接支持一套完整的七层协议——LonTalk 协议。保证了网络通信的安全性、可靠性和互操作性。Neuron 芯片提供了 11 个可独立配置的数字 I/O 管脚,可根据需要组织成 34 种类型的并行、串行和定时/计数接口,直接与它所连接的设备进行数据交换。LonMod01E 智能节点硬件框图如图 2 所示:

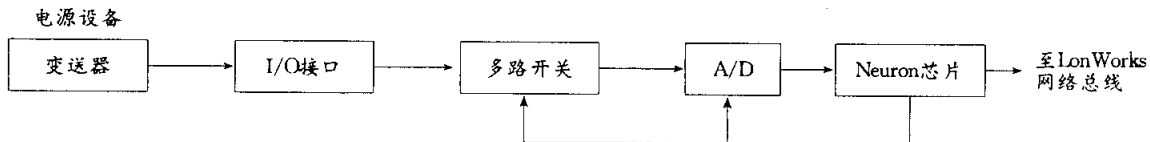


图 2 LonMod01E 智能节点硬件框图

LonMod02E 智能节点硬件框图如图 3 所示:

三、监控系统的软件设计

整个系统的软件设计分为二个部分,一是现场数据采集子系统的基于 Neuron 芯片的编程;另一个是监测和信息管理子系统的客户终端和服务器的编程。

1. 现场数据采集子系统的软件设计

Neuron 芯片的编程语言为 Neuron C,该语言以 ANSI C 为基础,经过扩展后具有 I/O 操作、事件处理、报文传送和分布数据处理等能力,同时还定义了一种称为“网络变量”的数据类型、一种“When”的语句类型和一整套 I/O 数据类型等。网络变量的使用大大简化了复杂的分布式应用的编程。网络变量提供了非常方便、灵活的观察系统中由节点操作的分布式数据。程序员不需要处理报文缓冲区、节点地址、请求/响应/重试过程,以及其他一些底层的细节。

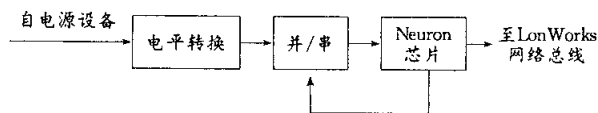


图 3 LonMod02E 智能节点硬件框图

(1) LonMod01E 智能节点的软件设计

LonMod01E 智能节点的主要功能是将 24 路模拟信号(电压、电流、有功功率、无功功率、功率因数、温度)通过电平转换、滤波及多路开关后分成 8 路信号,再进行 A/D 转换变成数字量,经过加工处理后,送至网络数据库,自动改写其中的网络变量的值。其软件的主要功能如下:

```
//编译指令
#pragma scheduler_reset //打开复位机制
#pragma num_domain_entries 1 //设置芯片内的EEPROM中的域表的大小
#pragma disable_snvt;i //删除自标识数据
//定义变量
unsigned int i,j,k; //控制字的数组下标变量及循环变量
```

```
unsigned int temp1,temp2,temp3;
//存放 A/D 转换后的中间结果
unsigned long data1,data2,data3;
//存放 A/D 转换后的最后结果
stimer repeating tmNodeHeart;//定时器
//定义输入/输出对象的初始状态
IO_0 output bit SCLK=0;
IO_1 output bit Din=0;
IO_2 input bit Dout=1;
IO_3 output bit CS1=0;
IO_4 output bit CS2=0;
IO_5 output bit CS3=0;
//定义具有 bind_info(unackd)类型的输出网络变量
network output unsigned long bind_info(unackd) AI_value[24];
//定义 A/D 芯片的控制字
const eeprom int condata [8][8]={{1,0,0,0,1,1,1,1},
{1,1,0,0,1,1,1,1},
{1,0,0,1,1,1,1,1},
{1,1,0,1,1,1,1,1},
{1,0,1,0,1,1,1,1},
{1,1,1,0,1,1,1,1},
{1,0,1,1,1,1,1,1},
{1,1,1,1,1,1,1,1}}
```

```
初始化、复位程序模块;
when(reset)
{
tmNodeHeart=30;//定时时间为 0.5 分钟
}
定时处理模块;
when(timer_expires(tmNodeHeart))
{
1)处理第一个 8 路模拟信号
I 送控制字
II 读数据
```

Ⅲ 数据变换

Ⅳ 将变换后的数据赋给网络变量 AI_value[k]

2) 处理第二个 8 路模拟信号

重复上述步骤 I 到 Ⅳ

3) 处理第三个 8 路模拟信号

重复上述步骤 I 到 Ⅳ

(2) LonMod02E 智能节点的软件设计

LonMod02E 智能节点的主要功能是将 24 路开关信号通过电平转换, 由移位寄存器将并行信号转换为串行信号, 经 Neuron C 芯片处理后, 送至网络数据库, 自动改写其中相应的网络变量的值。其软件的主要功能如下:

//编译指令

#pragma scheduler_reset //打开复位机制

#pragma num_domain_entries 1 //设置芯片内的 EEPROM 中的域表的大小

#pragma disable_snvt_si //删除自标识数据

//定义变量

unsigned int i,j,k; //循环变量

unsigned int first_temp1[8]; //存放第一次转换后的中间结果

unsigned int first_temp2[8];

unsigned int first_temp3[8];

unsigned int second_temp1[8]; //存放第二次转换后的中间结果

unsigned int second_temp2[8];

unsigned int second_temp3[8];

stimer repeating tmNodeHeart; //定时器

//定义输入/输出对象的初始状态

IO_0 output bit PullDown0=1;

IO_3 output bit PullDown3=1;

IO_6 output bit PullDown6=1;

IO_1 output bit ShiftClock1=0;

IO_4 output bit ShiftClock4=0;

IO_7 output bit ShiftClock7=0;

IO_2 input bit ShiftBit2;

IO_5 input bit ShiftBit5;

IO_8 input bit ShiftBit8;

//定义输出网络变量

network output unsigned int DI_value[24];

初始化、复位程序模块;

when(reset)

```
{
    tmNodeHeart = 30;
}
```

定时器处理模块:

when(timer_expires(tmNodeHeart))

```
{
    1) 处理第一个 8 路开关信号
    I 依次处理 8 路开关信号
    II 确认此次信号为非干扰信号
    III 将真实数据赋给网络变量 DI_value[i]
    2) 处理第二个 8 路开关信号
    重复上述步骤 I 到 III
    3) 处理第三个 8 路模拟信号
    重复上述步骤 I 到 III
}
```

四、监测和信息管理子系统的软件设计

监测和信息管理子系统的软件设计分为服务器编程和客户终端的编程。

服务器的程序流程框图如图 4、5 所示。

客户终端的功能框图如图 6 所示。

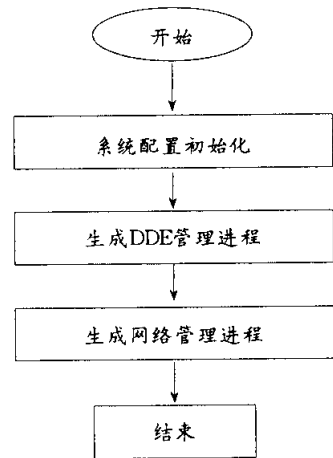


图 4 服务器主程序流程框图

五、结束语

该系统经过近一年时间的运行和测试, 证明了现场数据采集节点质量可靠, 采集精确度高, 每个节点均采用了 LonWorks 网络技术, 使现场的布线连接非常灵活自由。通信传输突破了传统主/从方式的限制, 具有响应速度快、实时性强等特点。

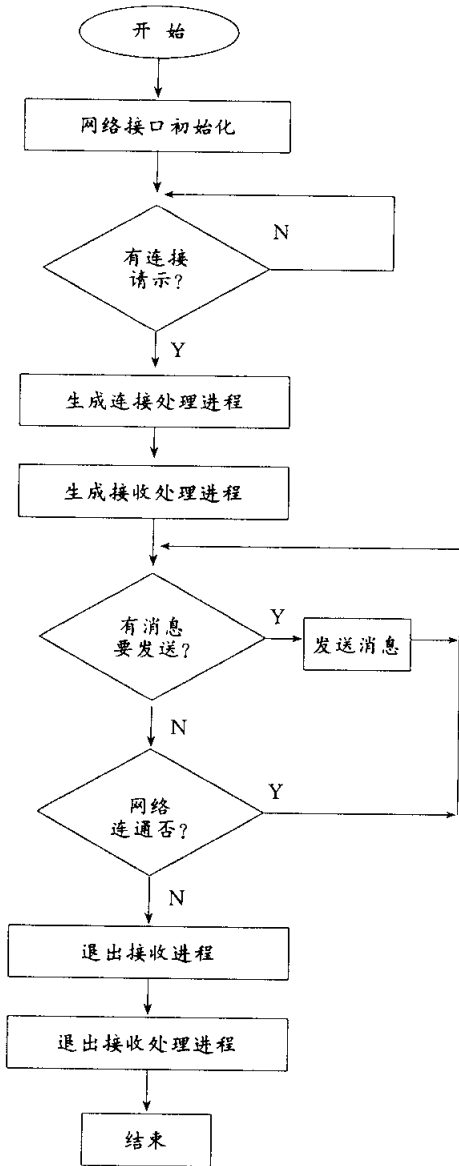


图5 服务器网络管理流程框图

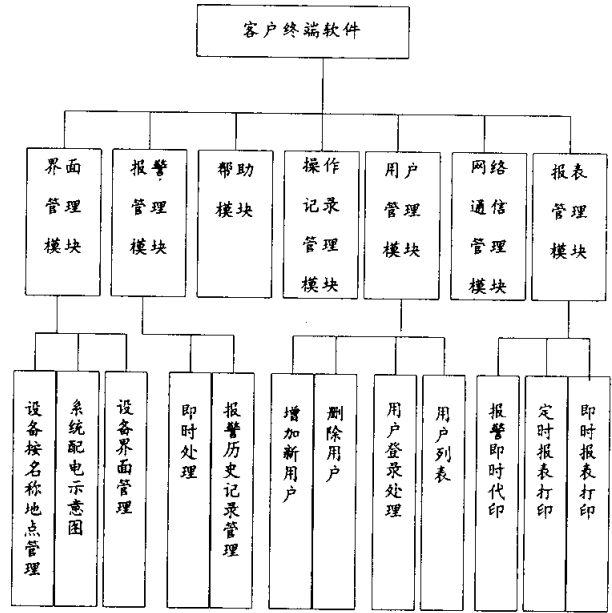


图6 客户终端软件功能框图

参考文献

[1] Echelon Co. Neuron C 参考手册 第二版
 [2] Echelon Co. Neuron C 程序员指南 第四版
 [3] Echelon Co. LonManager DDE Server User's Guide Vision 1.5
 [4] David Benne 等著, 徐军等译, Visual C++ 5.0 开发人员指南, 北京, 机械工业出版社和西蒙与舒斯特国际出版公司。

(收稿日期: 99年12月10日)