HT-7U 极向场电源控制系统跨平台网络通信

陈飞云 翁佩德 傅 鹏 龙 风
(中国科学院等离子体物理研究所 安徽 合肥 230031)

摘 要 核聚变装置 HT-7U 对极向场电源控制实时性要求特别高,必须在一个控制周期(1ms)内完成对 12 套电源设备的通信、反馈控制、采集、测量等闭环控制环节的操作。该系统利用了 QNX 平台的微内核实时性。同时为了增强人机界面的友好性,采用 Windows 系统作为操作平台。本文描述了在 QNX 和 WINDOWS 不同操作系统之间的一种跨平台网络通信。实践表明基于 TCP/IP 协议,利用 Socket API 建立的套接字在两种操作系统之间进行通信是完全可行的,而且是高效的。

关键词 HT-7U QNX Visual C++6.0 TCP/IP 网络通信

NETWORK COMMUNICATION BETWEEN DIFFER FLATS FOR POLOIDAL FIELD POWER SUPPLY CONTROL SYSTEM OF HT-7U

Chen Feiyun Weng Peide Fu Peng Long Feng
(*Institute of Plasma Physics, Chinese Academy of Science, Hefei Anhui 230031, China*)

Abstract The nuclear fusion device—HT-7U given extremely requirements for real time response ability of poloidal field power supply control system. It must implement communication, feedback control, data acquisition, and measurement within a control cycle (1ms) on 12 set power supply sub-systems devices. It should use the QNX OS high neutrino kernel realtime, and for enforcing the friendly of man-machine interface, we use Windows as operate flat. This paper describes network communication between QNX and Windows. The practice shows that the communication on network between the two operate systems is entirely feasible and high efficiency, with the socket built on socket API based on TCP/IP.

Keywords HT-7U QNX Visual C++6.0 TCP/IP Network Communication

1 引 言

"HT-7U 超导 TOKMAK 装置"是国家九五重大科学工程。极向场电源控制系统是托卡马克主要子系统之一,它为等离子体的产生、约束、维持、加热,以及等离子体的电流、位置、形状、分布和破裂的控制,提供必要的工程基础和控制手段。对装置运行的性能与安全,物理实验的成败与效率,有着至关重要的作用。其实时性、可靠性、可扩充性要求甚高。为了满足整个控制系统的这种高要求,整个控制系统由三层网络组成:Windows 监测层、QNX 实时控制层、现场总线执行层。Windows 监测层选用 Windows 2000 平台作为操作工作站,进行测量、监控、故障分析和波形显示。而 QNX 实时控制层选用 QNX6.20 实时操作系统,担负着电源系统多变量反馈电流反馈控制、系统连锁控制、保护及各种运行方式的调配,它与 Windows 监测层的通信是通过基于 TCP/IP 协议,利用 Socket 建立的套接字进行跨系统通信来实现的。现场总线层由大量的现场总线控制器及现场总线模块组成,其部分信息通过网络传输到 QNX 子层或者直接送到 Windows 子层。

2 QNX6.20 网络体系结构

QNX6.20 是一个多任务、分布式、嵌入式、可扩展规模的符合 POSIX 标准的微内核实时操作系统;其内核提供 4 种服务:进程调度、进程间通信、底层网络通信和中断处理;网络服务程序在内核外执行,为程序员提供了一个单一化的编程接口,忽略所涉及到的网络结构和数量;QNX6.20 系统可以动态地停止和启动网络服务程序,支持 Qnet, TCP/IP 以及其它通信协议,并且可以使它们以任何方式结合在一起同时运行。

QNX 网络子系统由三个相互作用的部分组成:

- 网络可执行管理器(io-net) 可以根据配置命令负责加载协议和驱动接口。
- 网络协议接口(npm-tcpip.so, npm-qnet.so) 负责执行特定协议的具体工作。

tcpip 在 QNX 动态网络结构中,QNX 所提供的以协议模块方式作为 BSD TCP/IP 栈的一种补充。

qnet 是 QNX 内部网络协议,其主要作用为扩展 OS 消息传递功能,使得微内核网络对 IPC(进程间通信)而言是透明的。

- 网络驱动接口(devn-*.so) 负责管理特定网络适配器。

作为一个分布式实时操作系统,QNX 节点间的网络服务采用 FLEET(fault-tolerant networking Load-balancing on the fly、Effi-

收稿日期:2004-06-14。基金项目:国家发展计划委员会"投资(1998)1303 号项目"(子项目)。陈飞云,博士生,主研领域:实时控制,TCP/IP,实时操作系统 QNX 的应用。

cient performance、Extensible architecture、Transparent distributed processing)网络技术。消息传递构成了 QNX 体系结构的基础。通过在这个系统中最基础的层次上提供网络服务,任何在 OS 体系结构中更高层次上提供的服务都可以被在局域网中任何地方的任何进程透明地访问。使得在网络中本地调用和远程调用没什么区别,可将物理上隔离的多个机器变成一个逻辑上的超级计算机。

3 QNX 与 Windows 两种操作系统间的网络进程通信

3.1 网络通信模式

网络中存在多种类型的机器,这些不同类型的机器表示数据的字节顺序是不同的,即主机字节顺序不同。而网络协议中的数据采用统一的网络字节顺序,因为只有采用统一的字节顺序,才能在不同类型的机器和不同操作系统间进行正确地发送和接收数据。进行跨平台进程间的通信时必须要将字节序转变成网络字节序进行传递,并且套接字的建立需要双方可见的 IP 地址和端口号。在传输层可以选用 TCP 或者 UDP 协议进行通信,视具体情况而定。在 HT-7U 极向场电源控制系统中,同时选用两种协议进行交互式传输数据。

3.2 网络通信工作方式

整个系统的控制方式分为两种:程序控制和反馈控制。若采用反馈控制,由 QNX 实时控制层和现场总线执行层完成,Windows 平台接收数据采集节点发送过来的实际工作电流值,进行波形显示,起监控作用。若采用程序控制,整个系统的工作以在 Windows 平台预设电流值、置位开关状态开始,把本次工作所涉及到的所有预设值按一定方式组成一个特定格式的数据报,通过面向连接的可靠的 TCP 协议发送至 QNX 实时反馈控制节点;反馈控制节点将收到的预设值同电流回路采集到实际值进行反馈运算,输出误差补偿值到 QNX 实时控制网络;控制网络的现场总线控制器将收到的误差值进行对应的分解运算,直接控制对应的电源设备;数据采集节点将每个控制周期的实际工作电流转变成数字量分别发送到反馈控制节点和数据库备份节点(此处采用 UDP 协议)以及 Windows 平台的波形显示平台。整个数据流程过程必须控制在一个时间周期(1ms)内完成。数据流程见图 1 示。

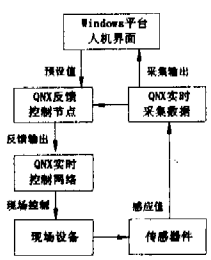

图 1 数据流程图

3.3 Windows 平台的软件实现

Windows 平台软件用 Visual C ++6.0 编程,运行在 Windows 2000 环境下。Visual C ++6.0 的最大的特色就是提供对面向对象技术的支持,它利用类把大部分与用户界面设计有关的 Windows API 函数封装起来,通过 MFC(Microsoft Foundation Class)类库的方式提供给开发人员使用,大大提高了程序代码的重用性,其功能几乎包括了 Windows 应用的各个方面。Visual C ++6.0 还支持多任务、多线程的编程技术。实际运行界面如图 2。

CBlocking Socket 是在 Visual C ++ 的 Winsock API 基础上的一个简单包装,为辅助线程中的同步编程而设计的,其优点就是能进行对错误的异常处理和发送、接收数据的超时处理。这一

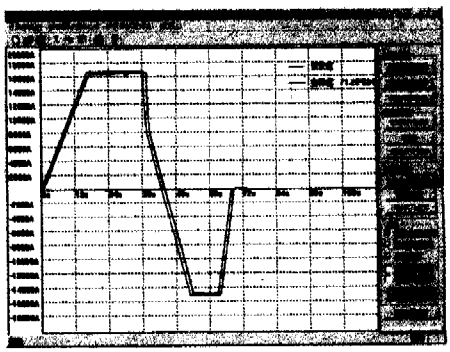

图 2 程序运行界面

点对极向场电源的实时控制反应能力尤为重要,因此 Windows 平台软件网络部分程序采用 CBlockingSocket 编写。对于接收数据采集节点发送过来的数据采用 UDP 服务器模式,同时利用辅助线程进行具体的接收和处理数据;而对于发送预设电流值、置位开关状态采用 TCP 客户机模式。部分 TCP 客户机模式代码如下:

```
char singal[4]
singal[0] = 'p';//设置标志位
//在此设置开关状态值
CBlockingSocket sClient;
CSockAddr saServer;
saServer = (CSockAddr(ip,port2);
try{
    sClient. Create( );//建立套接字
    sClient. Connect(saServer);//连接服务器
    sClient. Write(singal,4,10);//给服务器发送 singal 值
}
//异常处理
catch( CBlockingSocketException * e){
        e- > Delete( );
        AfxMessageBox("connection error!");
        return;
        }
    sClient. Close( );//关闭套接字
```

3.4 QNX 平台的软件实现

QNX 平台软件采用 ANSI C 语言、利用 QNX6.20 特有的函数和库函数来编写,运行在 QNX6.20 环境下。采用 BSD Socket API 作为套接字应用程序接口(Socket API)进行 TCP/IP 通信,在 Windows 环境中,Winsock API 是基于和共享于大部分 BSD Socket API,这就使得在这两种操作系统间进行数据互送相当容易。在 HT-7U 极向场电源控制系统中,QNX 实时平台与 Windows 平台间的通信主要有两步组成,即发送和接收数据。同 Windows 平台相配合,发送数据进程每 50ms 向 Windows 平台发送一次当前控制系统最新的状态信息,采用 UDP 协议封装当前系统运行时间值和系统电流、系统电压等需要实时监测和监控的状态信息;其接收进程采用 TCP 服务器模式,并始终处于等待客户机连接请求阻塞状态。部分 TCP 服务器模式代码如下:

```
struct sockaddr_in servaddr;
listenfd = socket( AF_INET,SOCK_STREAM,0);//建立套接字
//在此处给 servaddr 赋初值,给定 IP 地址和端口号
bind(listenfd,(SA*) & servaddr,sizeof(servaddr));        //绑定套接字
//设置套按字为非阻塞式
ioctl(listenfd,FIONBIO,& nonblock);
```

(下转第 62 页)

$$m = \alpha \frac{u_i}{\sum_i u_i} + (1-\alpha)\frac{N_i}{\sum_i N_i}$$

其中 $0 \leqslant \alpha \leqslant 1$，我们可以看出，第一部分实际上是结点在某一时间内来自某结点的发送数据速率。第二部分是该结点在整个子数中拥有的子结点及孙接点和的数目在该树中的比值。一般来说，子结点越多占的比值就大，分配到的速率也将相对大一些。该方法充分体现了在拥塞控制过程中的公平性。

5 结 论

基于矩阵的动态拥塞控制，适合分布式拥塞控制、能够很好地处理持续性拥塞和局部临时性拥塞。从文中的仿真结果可以看出基于建立在路由路径上的动态矩阵信号机制，可以快速检测拥塞并反馈给后继结点，从而迅速缓解拥塞。

参 考 文 献

[1] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," In Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies(INFOCOM 2002), New York, NY, USA, June, 2002, pp. 1 ~ 10.

[2] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. Infrastructure tradeo_s for sensor networks. In Proc. of First ACM International Workshop on Wireless Sensor Networks and Applications(WSNA 2002), pp. 49 ~ 58. Atlanta, September 2002.

[3] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed di_usion: A scalable and robust communication paradigm for sensor networks. In Proc. of the 6th Annual International Conference on Mobile Computing and Networking(Mobicom 2000), pp. 56 ~ 67, August 2000.

[4] C. -Y. Wan, A. T. Campbell, and L. Krishnamurthy. Psfq: a reliable transport protocol for wireless sensor networks. In Proc. of First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002), pp. 1 ~ 11. Atlanta, September 2002.

[5] A. Woo and D. Culler. A transmission control scheme for media access in sensor networks. In Proc. of the 7th Annual International Conference on Mobile Computing and Networking(Mobicom 2001), pp. 221 ~ 235, July 2001.

[6] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz. Event-to-sink reliable transport in wireless sensor networks. In Proc. of the 4th ACM Symposium on Mobile Ad Hoc Networking & Computing.

(上接第 18 页)

```
listen(listenfd,1);//等待客户机连接
ioctl(connfd,FIONBIO,& nonblock);
connfd = accept(listenfd,(SA*) & cliaddr,&len);
if(connfd >=0)
{
    bzero(buf,sizeof(buf));
    //接收数据
    if(read(connfd,buf,sizeof(buf)) <=0)
        {
        continue;//继续接收
        }
  //在此处理所接收的数据
  }
 close(connfd);关闭连接套接字
```

4 QNX 平台控制周期及同步性问题的解决

在 HT-7U 极向场电源控制系统中，QNX 实时层在一个控制周期(1ms)内不止要完成同 Windows 平台、现场总线数据互传，还要保证各节点间进程合理有序地实时通信，如反馈节点通过实时控制子网周期组播出控制信息报文，alpha 节点接收到此数据报并对之进行解析换算，将控制晶闸管控制角度通过 D/A 卡直接输出控制极向场电源。这就要考虑控制周期的设定和各个子进程的同步性问题。在本控制系统中，每个节点建立定时器进程，设定时间周期为 1ms，担负着整个节点的定时任务。QNX 定时器时间是由系统内核产生的，这样各个子节点的定时时间基本一致，达到全局同步的目的。在一个节点内，当 1ms 周期时间到时，定时器进程则向各个其它子进程发送 pulse 信号，当其它子进程收到一个 pulse 信号，并检查其 code 内容，若由定时器发送过来的，则运行程序。这样周而复始地进行，达到 1ms 定时周期的目的。定时器进程部分程序如下：

```
struct sigevent timer_event;
struct itimerspec itime;
timer_chid = ChannelCreate(0);
//给 timer_event 赋值，确定其属性、优先级及 code 值
timer_create(CLOCK_REALTIME,&timer_event,imer_id);//建立定时器
//在此设定 1ms 时间周期，即给 itime 赋值
timer_settime(timer_id,0,&itime,NULL);
for(;;){
timer_rcvid = MsgReceivePulse(timer_chid, &timer_pulse, sizof(timer_
pulse),NULL);
//收到定时器发来的 pulse 信号，检查其 code 值
if((timer_rcvid ==0)&&(timer_pulse.code == TIMER_PULSE_CODE))
    {
//给 init 进程发送 pulse 信号
MsgSendPulse(init_coid,20,TIMER_PULSE_CODE,0);
//给 ps_manage 进程发送 pulse 信号
MsgSendPulse(ps_manage_coid,23,IMER_PULSE_CODE,0);
.....
    }
```

5 结 论

本系统在 HT-7 试验装置多次运行后，满足各方面的要求，具有良好的性能，达到设计要求。利用 BSD Socket 采用 API 建立的套接字，使得 QNX 平台能方便地同 Windows 平台在基于 TCP/IP 协议的网络上进行通信，满足系统提出的实时性和可靠性要求。本系统的建立及运行成功，为 EAST 装置的建立打下坚实的基础。

参 考 文 献

[1] Kruglinski DJ，潘爱民、王国印，Visual C ++ 技术内幕(第四版)，北京：清华大学出版社.

[2] Rob krten. QNX Operating System Architecture. QNX Software system Itd. 2000. 2.

[3] QNX Neutrino Realtime Operating System QNX Software system Itd. 2002. 4.

[4] 张斌、高波，Linux 网络编程，北京：清华大学出版社.

[5] 刘予学、边信黔、熊华胜，"基于 QNX 与 Windows 运行的 PC 机之间的网络进程通信"，《微计算机信息》，2003，19(7).