底层通信协议中内存映射机制的设计与实现*

刘 炜 郑纬民 申 俊 鞠大鹏

(清华大学计算机科学与技术系 北京 100084)

E-mail: lw@est4.cs.tsinghua.edu.cn

摘要 在底层网络通信协议中使用内存映射机制为用户层应用提供了虚拟网络界面,使用户层能够方便地访问快速通信设备;通过减少系统软件的协议处理开销,有效地减少了网络通信的延迟.讨论了通信协议中的内存映射机制的设计思想和实现过程,提出了通信区的概念,利用通信区有效地完成核心与用户之间的数据交换.同时给出一个实例,对其实现与性能进行了分析.

关键词 内存映射,延迟,通信区,协议,底层通信.

中图法分类号 TP393

随着机群系统的迅速发展,机群系统所进行的高速度、低耗费的高性能计算越来越多地受到重视.机群系统通常用高速局域网络将高性能的工作站互连,可以实现其扩展性好、性能价格比高等优点.其中的高速局域网络广泛使用了普通的商用网络技术,如快速以太网、ATM 等等.高速局域网络技术的迅速提高,能够提供高带宽、低差错的物理链路,这就使局域网通信中的瓶颈从物理信道上转移到了系统软件协议处理上.[1]如何使高速网络的物理特性充分地在用户应用层得以体现,正是本文所探寻的目标之一.

我们设计建立了一套面向用户的底层通信协议,它利用内存映射机制,将网络界面提供给上层协议所拥有,用户层应用占有虚拟网络界面,可以为自己的特殊应用定制专门的通信协议,并且使通信协议得以在应用层而不是传统的核心层实现.研究表明[2],在应用层实现通信协议可以减少操作系统核心的干预和开销,提高通信效率.有了网络通信性能的改进(包括软件和硬件),才能使强大的物理通信能力在应用中得到体现.[3,4]我们在底层通信协议中所使用的内存映射技术,具有效率高、灵活性好、协议层次简单等优点,但也有难于管理、缺乏保护机制等缺点,这将在我们的设计中加以克服.在实际的设计中,我们提出了通信区的概念,利用通信区有效地完成核心和用户之间的数据交换.

1 目 标

在机群系统中,通信速度的高低是影响计算性能的关键因素,建立低延迟、高带宽的可靠通信协议是我们的机群系统研究的主要目标之一.充分利用现有高速网络的成熟技术,使通信协议的层次更趋简明,减少消息传递途径中经过多层协议时的额外开销,我们在面向用户的通信协议中所使用的内存映射机制正是朝着这一方向所作出的努力.充分地发挥底层网络的物理带宽,尽可能地减少消息传递途径中的冗余开销是实现低延迟、高带宽的有效方法.

内存映射机制的实现可以采取两种途径:一是利用网络硬件来支持映射功能,二是利用操作系统的协助来完成映射机制.前者要进行专门的硬件设计或是选择专门的支持硬件,而后者则可以利用现有的一些商用网络硬件,将系统核心中的网络界面部分映射到用户空间中,从而实现面向用户的通信控制.本文讨论后者的设计

---

* 作者刘炜,1974年生,博士生,主要研究领域为并行/分布计算机系统,网络通信环境.郑纬民,1946年生,教授,博士生导师,主要研究领域为并行/分布处理.申俊,1969年生,博士生,主要研究领域为并行机群系统,通信系统.鞠大鹏,1967年生,讲师,主要研究领域为并行/分布式网络计算环境.

本文通讯联系人:刘炜,北京 100084,清华大学计算机科学与技术系应用教研组

本文1997-11-04收到原稿,1998-01-23收到修改稿

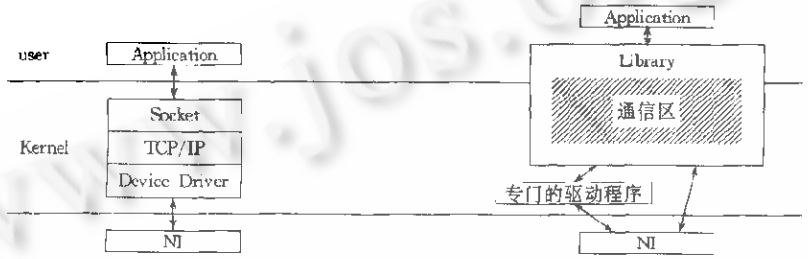
方案.

重新编写的驱动程序将充分支持内存映射机制的实现.我们要达到两个目标：(1) 灵活方便地访问底层通信网络；(2) 实现高效的通信区管理机制.

2　内存映射机制的设计与实现

2.1　原　理

传统的设备驱动程序总是通过操作系统内核来完成所有的硬件支持与协议软件处理的任务，操作系统要干预所有消息传递中的发送与接收处理，并且负责它们的缓冲管理和通信协议的实现.在面向用户的驱动程序中，我们利用了内存映射技术，使用户层可以拥有一定的网络界面，减少操作系统的干预过程，使用户层可以有更大的灵活性，并且在消息的发送与接收处理中有效地减少了数据拷贝.图 1 中的模型给出了两者的比较.



(a) 传统的TCP/IP协议通信模型　　(b) 利用通信区实现的面向用户的通信协议模型

图1　两种通信模型的比较

在传统的通信协议中，消息发送过程要经过应用程序到系统核心的数据拷贝，系统核心中消息数据的各种处理过程(包括数据封装、顺序控制等等)，最后将封装的报文分组发向通信网络中；消息的接收过程则基本上是上述过程的逆过程.消息在不同的系统空间中要进行多次拷贝，在协议的不同层次之间也要进行数据的拷贝.在面向用户的通信协议模型中，应用程序可以直接控制通信区的数据存取，而通信区域与 NI(network interface)之间是可以直接传递数据的.

专门的驱动程序负责网卡硬件的初始化、通信区的建立与维护等工作.作为核心进程的驱动程序，它建立了网络接口与通信区的连接.网络接口与通信区能够直接交换数据，从而减少了消息传递的环节，降低了系统开销.

2.2　通信区的结构设计

处于内存映射机制的实现核心地位的是通信区的设计.通信区是一块由用户层和核心层共享的物理内存，它采用由用户空间与系统核心空间共享内存的方法，避免了在不同的系统空间之间的数据拷贝.通信区的设计要考虑其共享特性，在用户层与核心层之间均可以方便地存取数据.通信区与网络界面之间的数据交换依赖于网络控制器的支持，所以，选择能够支持 DMA 方式的网卡是必要条件之一.通信区是连接网络界面与用户层应用的桥梁，它的实现就像缓存一样，用来存放用户数据和网络数据.接收过程中，核心进程从网络界面中取得数据，放入通信区的适当区域；用户进程从通信区中取走接收到的数据；发送过程中，用户进程把待发送数据放入通信区，由核心进程从通信区中取得数据，发送入网络界面.通信区通过 UNIX 的系统调用 mmap 由核心空间映射至用户进程空间，为两者所共有，通信区中包含了消息发送与接收的缓冲区，在其中维护了发送、接收和空闲三个循环队列，图 2 给出了通信区的结构示意.

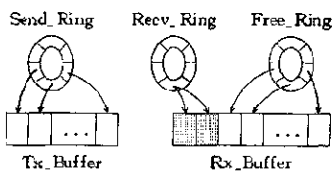


图2　通信区结构示意图

2.3　通信机制实现的实例

我们在 Linux2.0.0 的操作系统平台上，使用 DECchip21140 芯片的网络适配器进行了实例设计.DECchip21140 网络适配器支持 DMA 传送方式，它将接收到的网络以太帧通过 DMA 通道直接送到主机内存

中;相应地,它将内存中指定区域的待发送数据通过 DMA 通道送到网络接口中.通信区的具体形式如图 3 所示.其中 Send_Ring,Free_Ring,Recv_Ring 中每一项描述符的数据结构如下:

```
struct comm_send_desc{
    u_int    buf_offset;          /* 发送缓冲区相对于通信区的偏移地址 */
    int      len;                 /* 待发送数据的长度 */
    char     header[HEADER_LEN];  /* 用户通信协议的协议头 */
}
struct comm_free_desc{
    u_int    buf_offset;          /* 接收缓冲区中的空闲块 */
}
struct comm_recv_desc{
    u_int    buf_offset[PAYLOAD]; /* 接收缓冲区的偏移地址 */
    int      len;                 /* 接收到的数据长度 */
}
```

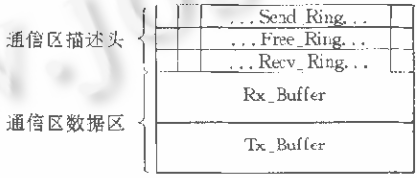
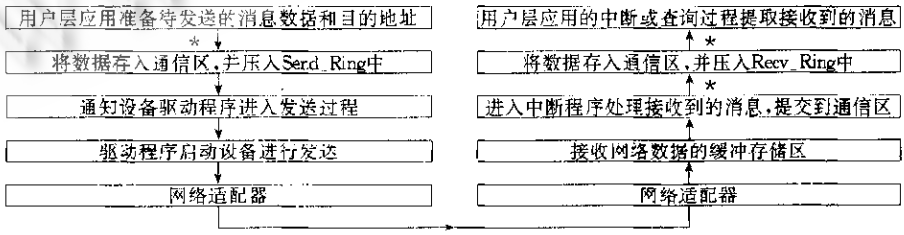


图3 通信区的分配

发送队列中保存着发送数据以及其协议头数据,这些数据将在网络接口中封装成以太帧形式发送到物理网络中去;而接收队列的每一个描述符则对应了一个接收到的消息,通过其 *buf_offset*[*PAYLOAD*]数组,可以将不同块中的数据重新组成消息本身,作为优化过程,小于 *PAYLOAD* * 4 字节的消息会直接存于其中而避免了缓存分配的开销.

在通信区描述头中维护了 3 个队列,其中的 buf_offset 分别指向了各自的数据缓冲区的偏移地址,这就保证了无论是核心程序还是应用程序都可以方便地将其视为自身存储空间的一部分而进行操作.

在发送过程中,用户层应用将准备好的待发送数据和目的地址复制到通信区中的适当区域,并填写 Send_Ring 中的相应描述符.发送过程紧接着就通知设备驱动程序进行发送,驱动程序在进行了发送过程的管理手续之后,通过 DMA 方式把数据发送到网络中去.在接收过程中,DEC21140 控制器会通过 DMA 方式主动地将接收到的数据存放到驱动程序中所指定的缓冲区中,然后发出中断信号通知驱动程序进行接收处理.在驱动程序的接收过程中,驱动程序将接收到的数据提交到通信区中的适当区域,并填写 Recv_Ring 中的相应描述符.对于用户而言,他可以选择中断方式或者查询方式来接收数据.在中断方式中,驱动程序负责向用户进程发送一个数据已接收到的信号,请求进入用户接收处理过程;在用户的查询方式下,用户每隔一段时间就查询接收队列的状态,当接收队列不为空时,用户主动地从通信区中提取数据.发送与接收过程中的消息流如图 4 所示.



注: * 表明数据在此处发生拷贝

图4 发送与接收过程的消息流

在发送过程中要进行一次数据拷贝,在接收过程中要进行两次数据拷贝.而在传统的驱动程序的设计中,存

在着不可避免的多次数据拷贝：发送过程中，用户到通信接口（如 Socket）、通信接口到设备驱动程序，至少发生了两次数据拷贝；接收过程中，接收缓存到设备驱动程序、设备驱动程序到通信接口、通信接口到用户空间，至少发生了 3 次数据拷贝. 从这一点上看，内存映射机制在数据拷贝次数的减少上是有其优越性的.

3　实验结果分析

为了对内存映射机制的性能作进一步的分析，我们对所实现的实例进行了性能测试，性能测试是在 Linux2.0.0 的操作系统上，使用 100 兆快速以太网卡来进行的，并且进行了内存映射机制与 TCP/IP 和 UDP/IP 的性能比较. 以下是我们的实验测试结果.

3.1　通信往返延迟时间

在用于并行计算的机群系统中，通信延迟是一个十分重要的性能指标，从图 5 中可以看出，采用了内存映射机制的通信系统的通信延迟远远低于 TCP/IP 和 UDP/IP 协议的通信延迟. 其主要原因是：前者在数据拷贝和缓冲管理等方面作了简化，从而实现了较好的通信性能；而后两者的多次的数据拷贝和复杂的缓冲管理，占用了较多的系统开销，影响了通信性能. 由于在并行计算中，小消息包用作控制信息的传递，它的延迟性能直接影响系统的计算速度，在我们使用了内存映射机制的通信中，有 64 字节的消息包的用户层往返延迟约为 58μs. 这比 TCP/IP 协议所提供的 270μs 要好得多.

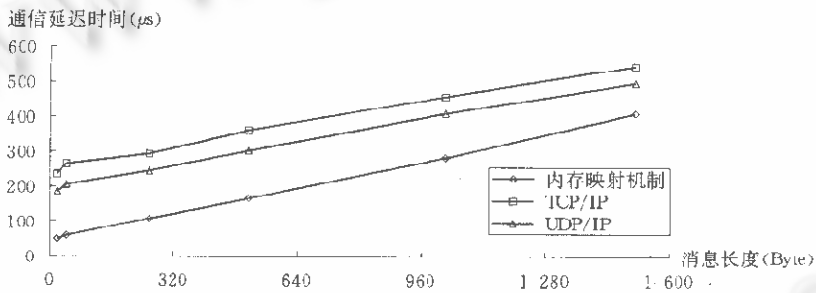


图5　几种协议通信延迟时间的比较

3.2　网络峰值带宽

图 6 中所示为内存映射机制与 TCP/IP 的带宽比较. 在我们实现的发送－接收（Send－Recieve）简单通信接口上，采用滑动窗口协议进行流量测试，本机制也取得了较好的性能. 对于 1K 字节的消息包，可以达到 86.6Mbps 的传输率，高于 TCP/IP 协议所取得的 77.1Mbps 的传输率. 在这里，我们可以看出，带宽性能的改进幅度不如通信延迟的改进幅度大，这是因为带宽性能主要受限于缓冲区的大小和流控协议的性能. 我们的底层通信协议主要针对通信延迟的性能作出了一些改进，对小消息包进行优化处理，减少了数据的拷贝，而没有在流量控制方面作出改进. 在带宽的测试中，我们仍然采用了传统的滑动窗口协议，其性能的改进主要是由于通信延迟减少而带来相应的带宽的改善. 在改善网络流量方面，我们仍需作进一步的研究工作.

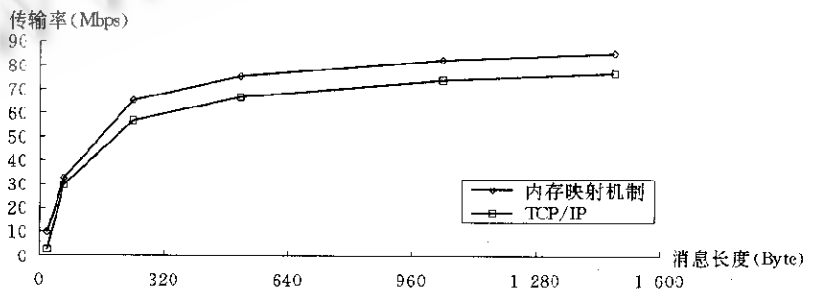


图6　内存映射机制与TCP/IP的流量性能比较

4 结束语

在底层通信协议的设计中采用内存映射技术，可以提高访问网络界面的灵活性和效率，实现了灵活高效的网络接口，降低了消息传递中的处理环节，减少了软件开销，提高了通信效率，这对于设计低延迟、高带宽的通信系统有着重要的意义.

参考文献

1 Blumrich M, Dubnicki C, Felten E W *et al*. Virtual-memory-mapped network interfaces. IEEE Micro, Feb. 1995, 15(1): 21～28

2 Eicken T V, Basu A, Bush V *et al*. U-Net: a user-level network interface for parallel and distributed computing. In: Proceedings of the 15th ACM Symposium on Operating Systems Principles. December 1995. 40～53. http://WWW.cs.cornell.edu/u-net/papers/sosp.pdf

3 Welsh M, Basu A, Eicken T V. Low-latency communication over fast Ethernet. In: Proceedings of EuroPar'96. Lyon, France, August 1996. http://WWW.cs.cornell.edu/u-net/papers/europar.pdf

4 Chang Sheue-ling, David Hung-Chang Du, Jenwei Hsieh *et al*. Enhanced PVM communications over a high-speed LAN. IEEE Parallel and Distributed Technology, 1995, 3(3): 20～32

Design and Implementation of Memory Map Mechanism in Underlying Communication

LIU Wei　ZHENG Wei-min　SHEN Jun　JU Da-peng

(*Department of Computer Science and Technology　Tsinghua University　Beijing*　100084)

Abstract　In underlying communication, using memory map provides user-level applications with a virtual network interface to enable user level access high speed communication devices, efficiently reduces the overhead of system protocol software and the latency in communication. In this paper, the design idea, the implementation process and the concept of communication area are discussed, by which the data can be exchanged between the kernel and user space efficiently. A sample and performance analyses are provided at last.

Key words　Memory map, latency, communication area, protocol, underly