◎网络、通信与安全◎

基于通用以太网卡的高性能通信库设计实现

胡长军,李景祥,常晓东,李建江
HU Chang-jun,LI Jing-xiang,CHANG Xiao-dong,LI Jian-jiang

北京科技大学 信息工程学院,北京 100083
Information Engineering School,University of Science and Technology Beijing,Beijing 100083,China
E-mail:ljxtf@sohu.com

HU Chang-jun,LI Jing-xiang,CHANG Xiao-dong,et al.Design and implementation high performance communication library based on commodity Ethernet NICs.Computer Engineering and Applications,2007,43(12):112-115.

Abstract: A high performance communication library independent of hardware called HPCL/Ethernet is implemented based on commodity Ethernet NICs on the Linux.HPCL/Ethernet supports commodity Ethernet NICs by using the network device interface on the Linux kernel.Message reliable delivery is guaranteed in the software layer.Channel abstract simplify buffer management and fixed-buffer policy adopted for avoiding message copy.Both interrupt reaping technology and the policy of message processing in link layer are also used to optimize communication.The experimental results show that the HPCL/Ethernet is characteristic of low latency with short payload message and high bandwidth with long payload message.It provides 1.18E+08 bytes/s of bandwidth which is 94.4% of hardware-level bandwidth for 1 196 bytes message.

Key words: cluster;communication library;commodity ethernet;interrupt reaping;fixed-buffer policy

摘 要:采用硬件无关的设计方法,在Linux下实现了一个基于通用以太网卡的高性能集群通信库HPCL/Ethernet。该库利用Linux操作系统的网络设备接口实现了对各种以太网NIC(Network Interface Card)的支持,在软件层实现一个轻量级通信协议支持消息的可靠传输。采用通道简化缓冲区管理,固定缓冲区策略来减少消息拷贝,采用中断回收技术、链路层直接处理消息来优化通信。测试结果表明,该通信库在传输小消息时具有很低延迟,在传输大消息时具有高带宽的特点。传输消息长度为1 196 bytes时带宽高达1.18E+08 bytes/s,是千兆以太网卡硬件带宽的94.4%。

关键词:集群;通信库;通用以太网;中断回收技术;固定缓冲区策略
文章编号:1002-8331(2007)12-0112-04 **文献标识码**:A **中图分类号**:TP338;TP393

1 引言

通信系统的低延迟、高带宽是提高机群并行计算系统性能的关键。在高性能通信子系统中,按照网络互连的网卡种类划分,有基于专用网络和通用网络两种通信库。专用网络如Myrinet、Infiniband、ELAN-3等,虽然具有高带宽低延迟的性能,但是价格昂贵。而通用网络如Ethernet等,其性能虽然不如专用网络,但是价格便宜很多。在90年代中期,这两种网络速度相差可以达到十几倍。而现在,千兆以太网已经开始流行,并且10 Gbps以太网已经出现。可以预见,利用通用网络进行高性能计算将会达到很高的性价比。因此,研究基于通用网络的高性能通信技术就显得很有意义。

目前,基于通用网络开发的高性能通信库有GAMMA[1],U-Net[2],GigaE PM[3],EMP[4]等,它们都采用网络接口硬件相关的设计方法。这种方法虽然可以获得好的通信性能,但是也牺牲了通用性。不同的硬件厂商生产很多种NIC,要支持不同的NIC就必须重新开发通信库来对它支持。同时,网卡硬件更新很快,新的硬件不断在市场中涌现,这也要求以硬件无关的方法设计通信库。

采用硬件无关的方法设计的通信库通常基于TCP/IP协议,典型的基于TCP/IP协议的PC集群用户通信软件接口有MPI(ch_p4设备)和PVM。TCP/IP协议存在开销主要有硬件中断上下文切换、链路层到IP层的协议切换、数据校验、内存分配和释放等,并不适合高性能通信。

本文采用硬件无关的设计方法实现了一个基于通用以太网卡的通信库HPCL/Ethernet:(1)利用Linux网络设备接口层,该库支持各种以太网NIC;(2)实现一个轻量级集群通信协议支持消息的可靠传输。因为Ethernet硬件并不保证消息传输的可靠性,硬件无关的设计方法必然要求上层通信协议保证消息的可靠传输;(3)高性能通信是HPCL/Ethernet的必然要求,并针对TCP/IP协议的开销进行了相应的改进。该库采用中断回收

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.60373008)。
作者简介:胡长军(1963-),男,博士,教授,博士生导师,主要研究方向:并行计算与并行编译技术,并行软件工程、网络存储体系结构、数据工程与软件工程;李景祥(1976-),男,硕士生,主要研究方向:高性能通信和并行计算;常晓东(1978-),男,硕士生,主要研究领域为并行软件工程;李建江(1971-),男,博士,副教授,主要研究方向:并行计算、并行编译与多线程技术。

技术、链路层直接处理消息、通道技术、固定缓冲区策略来优化通信,满足集群低延迟、高带宽通信要求。

2　HPCL/Ethernet 软件体系结构

如图 1 所示,Linux 的网络子系统被设计成完全协议无关的。这为采用硬件无关的通信库设计方法提供了基础。

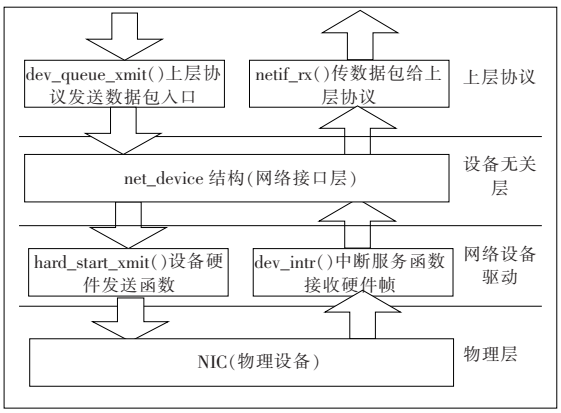

图 1　Linux 网络设备接口

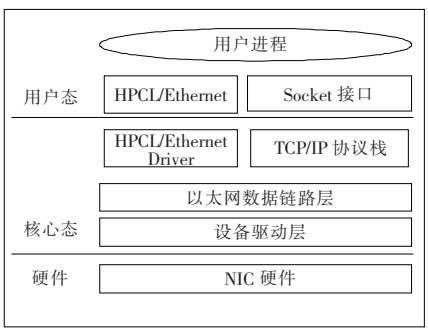

图 2　通信库软件体系结构

Linux 对所有的网络物理设备进行了抽象并定义了一个统一的接口 net_device 结构,任意一个网络接口均可以看作一个发送和接收数据包的实体。当要发送数据包时,网络子系统根据路由表选择相应的网络接口进行数据传输;而当接收到数据包时,通过驱动程序注册的中断服务程序进行数据的接收处理,所有的中断处理函数最后都调用 netif_rx()把协议数据包交给上层协议处理。

HPCL/Ethernet 通信库采用如图 2 的体系结构,它包含一个用户层的库 HPCL/Ethernet、一个进行集群协议处理的虚拟字符设备驱动 HPCL/Ethernet Driver、一个 Ethernet 数据链路层协议分发器(集群协议与 TCP/IP 协议共存),以及一个硬件相关的 Ethernet 网络设备驱动。HPCL/Ethernet 无需修改任何与网卡硬件相关的驱动代码,但需对 Linux 内核进行小部分修改:

(1)替换网络层入口函数 netif_rx()为 hpcl_ethernet_ netif_rx()函数,在该函数中实现协议分发器,设备中断处理函数会调用 hpcl_ethernet_ netif_rx()函数。

(2)在网络设备接口数据结构 net_device 中注册网卡设备中断处理函数,这样在内核中可以直接调用这个内核接收原语,实现中断回收技术。

(3)在数据链路层加入 HPCL/Ethernet 协议处理方法。实际上 hpcl_ethernet_ netif_rx()函数既实现 TCP/IP 协议和 HPCL/Ethernet 协议分发,也直接处理 HPCL/Ethernet 协议数据。

(4)设计一个虚拟字符设备,通过该字符设备驱动 HPCL/Ethernet Driver 支持集群通信协议和用户接口。HPCL/Ethernet driver 以模块的方式加载进内核。

结合图 1、图 2,下面是 HPCL/Ethernet 发送、接收包的过程:

(1)向网卡发送包

用户调用 HPCL/Ethernet 接口发送函数 hpclSend(),该函数通过系统调用操作虚拟字符设备。虚拟字符设备的驱动 HPCL/Ethernet Drive 响应用户的发送操作,对发送数据加一个集群通信协议头和以太网硬件包头后,调用 dev_queue_xmit()函数发送。该发送函数最终会调用到和每个网卡硬件相关,但每个 net_device 接口都会实现的函数 hard_start_xmit()。

(2)从网卡接收包

网络包的接收有两种方式:中断接收或用户主动接收。

中断接收是网络设备在接收到包后产生设备中断,中断处理程序处理接收帧,最后调用网络层入口函数 hpcl_ethernet_ netif_rx(),在该函数中识别 HPCL/Ethernet 协议头并把数据处理后拷贝到用户接收缓冲区。另一个接收过程是通过用户直接调用 net_device 中注册的中断处理函数接收,后面提到的中断回收技术详细阐述了它的接收处理过程。

3　可靠的轻量级集群通信协议

TCP/IP 协议主要是为了支持低带宽、高差错率、多结点、且距离较远的广域网,并不适合高性能通信。以太网硬件不支持消息的可靠传输,网卡硬件无关的设计方法必须在软件协议层来保证消息的可靠传输。

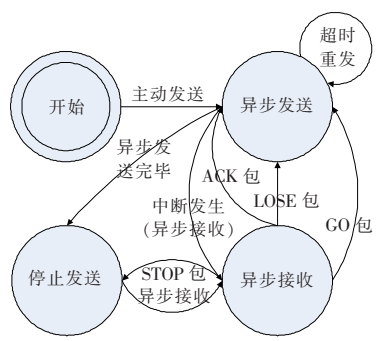

图 3　发送端状态转换图

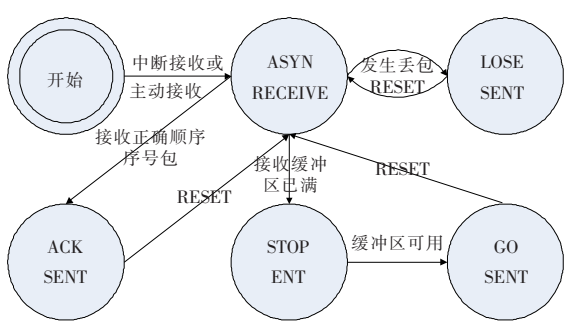

图 4　接收端状态转换图

HPCL/Ethernet 通信库采用 send/receive 双边通信模型。该协议保证消息的可靠传输和 FIFO 接收顺序。发送方可以异步发送 Nasy 个消息给接收方,即发送方接收到 i 序号消息的确认后,就可以异步发送 i+1 到 i+Nasy 序号的消息,而接收方接收到 i+1 序号消息后需马上向发送方发送该消息的确认消息。图 3 中发送方异步发送消息后,可能进入异步接收状态,接收到接收方发来的四种消息:确认消息 ACK、丢包消息 LOSE、停止发送消息 STOP、重新发送消息 GO。如图 4,接收方收到正确序号的消息后发送确认消息 ACK 以确保消息的 FIFO 接收。接

收方如果接收到的消息序号不是刚接收到的消息的下个序号的消息,且消息序号大于应该接收的序号,表明发生丢包,则发回 LOSE 消息驱动发送端重发。如果消息序号正确,但接收缓冲区已满,则发送“停止发送”消息 STOP 给发送方,进行简单的流量控制。当重新有缓冲区可用时,发送“继续发送”GO 消息给接收方,重新驱动发送过程。此外,发送方也用超时重传机制保证消息可靠传输。

该协议采用固定发送窗口和接收窗口,比 TCP/IP 协议中复杂的滑动窗口机制要简单的多。此外,集群环境下千兆以太网的通信传输的差错率小于 10^{-10},协议去掉了类似 TCP 协议中复杂的差错控制,减轻协议处理开销。

4 HPCL/Ethernet 通信优化技术

由于采用硬件无关的设计方法,HPCL/Ethernet 的性能受网卡驱动程序的影响。为了减少通信延迟,针对 TCP/IP 的主要开销,HPCL/Ethernet 采用下面几种技术优化通信。

4.1 中断回收技术实现

中断需要保存软硬件现场,并且中断上下文切换是很耗时的。采用中断回收技术来避免硬件中断开销。用户程序需要等待一个消息到来的时候,首先关中断(从而避免硬件中断,即中断回收),然后通过直接调用 NIC 硬件中断处理程序来得到消息,最后再开中断。现在网络速度已经很高,让用户进程循环等待一个消息的到来而无需挂起可以获得更高的性能。

定义一个虚拟字符设备的设备驱动 HPCL/Ethernet Driver(如图 5)来支持集群通信协议,对该字符设备的操作集定义如图 3。

```
static struct file_operations hpcl_ethernet_fops={
        poll:       hpcl_ethernet_poll,
        ioctl:      hpcl_ethernet_devioctl,
        mmap:       hpcl_ethernet_mmap,
        open:       hpcl_ethernet_open,
        release:    hpcl_ethernet_release
};
```

图 5 HPCL/Ethernet Driver 虚拟字符设备操作集

当用户调用 hpclReceive()时,该函数会调用系统调用 ioctl()进入 HPCL/Ethernet Driver 中响应所有 I/O 控制命令的 hpcl_ethernet_devioctl()函数。图 6 是该函数响应接收命令的代码片断。

```
case HPCLRECV: /* 接收命令 */
{struct ehpcl_recv recv;
 ret=hpcl_ethernet_recvframe(ctx,&recv);
/*  该函数从通道上下文 ctx 获得当前已经接收到的
* 消息描述符 recv,该描述符指明接收缓冲区地址、长度、
* 消息源结点号,若 ctx 中有消息存在,返回 ret 为 0
*/
 if(ret){
/* 接收缓冲区中还没有消息,用中断回收技术接收 */
  interrupt_diable();/* 关中断 */
  dev_intr_handler();/* 调用 NIC 设备中断函数 */
  hpcl_ethernet_recvframe(ctx,&recv);
  interrupt_enable();/* 开中断 */
 }
  copy_to_user((structehpcl_recv*)addr,&recv,sizeof(struct ehpcl_recv));
  /* 把接收到的消息描述符从内核空间传给用户空间 */
}
```

图 6 hpcl_ethernet_devioctl 函数响应接收命令代码片断

接收过程首先调用函数 hpcl_ethernet_recvframe(ctx,&recv),从接收缓冲区中取得已经到达消息的描述符。若通道上下文接收缓冲区已有消息,用户调用接收函数并不会进行关中断下的接收。在中断处理函数 dev_intr_handler()中最后调用网络层的入口函数 hpcl_ethernet_netif_rx(struct sk_buff *skb)。该函数判断帧载荷的协议类型,如果是高性能协议帧,直接处理该帧。如果是其它帧,调用 Linux 操作系统原来的网络层入口函数 netif_rx(skb),仍然通过软中断机制交给其它上层协议处理。

实际上,中断回收时机上只发生在用户主动调用 hpclReceive()接收函数且接收缓冲区没有消息到达时,其它情况下网络帧的接收仍然是通过中断激发,由中断处理函数接收的。虽然如此,接收进程在等到消息的情况下主动关中断接收,依然可以减少中断造成的开销。同时,通过 hpcl_ethernet_netif_rx()函数实现协议分发器,实现了高性能协议和 TCP/IP 协议族在系统中共存。

4.2 固定缓冲区策略

固定缓冲区策略是将系统缓冲区和应用缓冲区合起来成为统一的通信缓冲区。也就是说,内核空间和用户空间共享物理页面缓冲区。在初始化通信库通道上下文时,预分配发送和接收缓冲区,并把它们配置成环形缓冲队列、锁定缓冲区所在页面使其常驻内存,利用系统调用 mmap 把通信缓冲区映射到用户虚拟地址空间。固定缓冲区有两个好处:一是用户直接使用通信缓冲区可以避免消息内存拷贝,而内存拷贝是开销很大的操作;二是由于通信缓冲区在程序运行过程当中是固定的,避免了传统机制中对系统缓冲区频繁的申请和释放操作,简化了缓冲区的管理,为简化协议提供了条件。

4.3 通道技术

HPCL/Ethernet 用“通道”来抽象一个虚拟的通信网络。在并行应用分布在各个结点的进程通信之前,首先进行通道的初始化,即初始化并行应用的通信上下文和接收、发送缓冲区。并行应用的每一个进程只能利用具有相同通道号的通道通信,并且这个并行应用独占这个通道。实际上,基于通道的通信能带来两个主要的好处:一是上层的并行应用只要在指定的通道上接收和发送消息,而无需象面向连接的通信(如 TCP/IP)那样,在上层应用等待消息时,需要在每个连接上等待;二是有利于实现轻型的通信协议。因为在面向连接的通信中需要为每个连接分配通信的缓冲区,而在 HPCL/Ethernet 中,所有的通信缓冲区都只和一个通道相关联,所有这个通道对应的消息都放在这个通道缓冲区中管理,大大简化了缓冲区的管理和协议实现的复杂度。

4.4 消除协议切换开销

利用 TCP/IP 协议进行通信,由链路层协议切换到上层网络协议处理采用 Linux 软中断机制。该机制实现一种延迟执行策略,它基于网络速度比较慢这种假设。但这种假设不再符合现在的 Gigabit 以太网以及目前已经出现的 10 Gbps 带宽的以太网卡。

HPCL/Ethernet 通过在链路层直接调用集群协议处理函数,可以减少软中断驱动的协议切换延迟。在数据链路层,通过识别硬件帧负荷协议头,可以区别出到达的是集群通信协议还是 TCP/IP 协议族的协议。如果是集群通信协议头,直接调用协

议处理函数来处理,消除了协议切换开销。

5 性能测试

在 16 个节点规模的 USTB SMP-Cluster 集群环境下测试了该高性能通信库,用 netperf-2.4[7]的 TCP_RR 测试(TCP 建立一次连接,在这一连接上进行多次数据传输)TCP/IP 的延迟,用 pingpong 和 burst 程序来分别测试 HPCL/Ethernet 的延迟和带宽。测试环境如表 1。

表 1 HPCL/Ethernet 测试环境

计算节点 CPU 及缓存	2 x Intel Xeon 2.8 GHz/512 K L2 Cache
网卡	Intel EtherExpress/1000
交换机	24 口千兆以太网交换机
操作系统	Redhat 9.0 Linux(2.4.21)

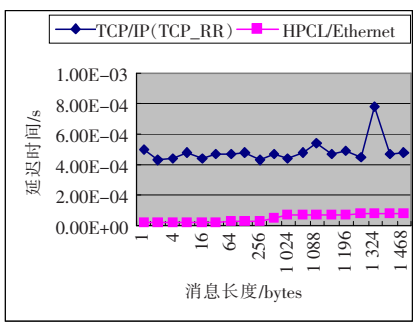

图 7 RTT(Round Trip Time)比较

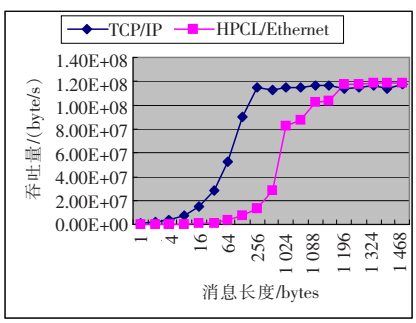

图 8 HPCL/Ethernet 和 TCP/IP 带宽比较

图 7 是 HPCL/Ethernet 和 TCP/IP 的延迟时间比较。可以看出 HPCL/Ethernet 发送相同大小消息的延迟比 TCP/IP 要小得多,TCP/IP 延迟时间是 HPCL/Ethernet 延迟时间的一个数量级左右。如在传输 128 bytes 节的消息时,HPCL/Ethernet 的延迟时间为 2.54E-05 s,而 TCP/IP 为 4.82E-04 s。原因有两方面:一方面是因为 TCP/IP 协议是个功能复杂的协议,尤其是数据校验的计算非常费时,而在系统区域网(System Area Network)的集群环境中数据差错率非常低,没有必要。另外 IP 是个路由协议,而局域网中的 PC 结点完全可以直接通过网卡 MAC 地址直接通信。另一方面,HPCL/Ethernet 中采用的优化技术如中断回收等都是为了减少延时时间。

图 8 是 HPCL/Ethernet 和 TCP/IP 带宽比较,可以看出,在传输 128 bytes 的小消息时,TCP/IP 协议比 HPCL/Ethernet 具有更高的带宽,这是由于每个 TCP/IP 包一次可以传输多个消息,效率较高。HPCL/Ethernet 由于一次只传输一个数据包,在传输小消息时效率并不高。从图 8 可以看出,随着消息长度的不断增长,HPCL/Ethernet 的带宽持续稳定上升。在传输 1 196 bytes 的消息时,带宽达到 1.18E+08 bytes/s(994 Mb/s),是千兆以太网带宽的 94.4%。另外,HPCL/Ethernet 作为上层并行软件接口(如 MPI[5])的底层通信原语,当传送大消息时,MPI 的消息分片功能将使大消息分成多个底层 HPCL/Ethernet 的最大传输单元进行传输(当然,最后一个消息分片片断除外),这使得 HPCL/Ethernet 能满足高性能通信要求的高带宽。多通道技术利用多道网络链路通信可以提高带宽,这一技术使以后进一步提高通信库带宽成为可能。

6 结论

随着千兆以太网和 10 Gbps 以太网的出现,利用通用的以太网卡互连进行高性能通信可以获得很高的性价比。采用硬件无关的设计方法实现了一个支持通用以太网卡的高性能集群通信库。利用 Linux 操作系统的网络设备无关层,HPCL/Ethernet 实现了对通用以太网卡的支持;实现一个轻量级通信协议支持消息的可靠传输,并支持简单的流量控制。另外,采用多种软件技术优化通信来减少通信延迟时间。测试结果表明,HPCL/Ethernet 是具有低延迟和传输大消息时的高带宽的高性能通信库。(收稿日期:2006 年 12 月)

参考文献:

[1] Chiola G,Ciaccio G.Efficient parallel processing on low-cost clusters with GAMMA active ports[J].Parallel Computing,2000-02.

[2] Basu A,Buch V,Vogels W,et al.U-Net:A user-level network interface for parallel and distributed computing[C]//Proc of the Third International Symposium on High Performance Computer Architecture(HPCA),Feb 1997.

[3] Shinji Sumimoto,Hiroshi Tezuka,Atsushi Hori.High performance communication using a commodity network for cluster systems[C]//the Ninth International Symposium on High Performance Distributed Computing(HPDC-9),IEEE,August 2000.

[4] Piyush Shivam,Pete Wyckoff,Dhabaleswar Panda.EMP:Zero-copy OS-bypass NIC-driven gigabit ethernet message passing[C]//International Conference on Supercomputing 2001.ACM SIGARCH,2001.

[5] The Message Passing Interface(MPI) Standard[S/OL].http://www.mpi-forum.org.

[6] Tezuka H,Hori A,Ishikawa Y,et al.PM:An operating system coordinated High performance communication library[C]//LNCS 1225.High-Performance Computing and Networking.[S.l.]:Springer-Verlag,1997:708-717.

[7] The Public Netperf[EB/OL].http://www.netperf.org/.