第34卷 第14期 计 算 机 工 程 2008年7月
Vol.34 No.14 Computer Engineering July 2008
·网络与通信· 文章编号：1000—3428(2008)14—0116—02 文献标识码：A 中图分类号：TP393

基于RapidIO和存储映射的高速互连网络

黄 亮，刘福岩
(上海大学计算机工程与科学学院，上海 200072)

摘 要：分析当前高速互连网络中同时存在的TCP/IP, GAMMA, InfiniBand, SCI等技术的实现机制，介绍RapidIO高性能总线技术。研究RapidIO协议和MPC8548处理器的相关技术，提出在RapidIO高速互连网络中实现存储映射的通信技术解决方案。
关键词：RapidIO网络；存储映射；高速互连网络

High-speed Interconnection Network Based on RapidIO and Memory Mapping

HUANG Liang, LIU Fu-yan
(School of Computer Engineering and Science, Shanghai University, Shanghai 200072)

【Abstract】This paper analyzes implementation mechanism of technologies, such as TCP/IP, GAMMA, InfiniBand, and SCI, which exist simultaneously in the current high-speed interconnection network. With RapidIO high performance bus technique introduced, this paper researches the prominent feature of the new protocol RapidIO and MPC8458 CPU, and presents a communications technology solution for RapidIO to implement memory mapping.
【Key words】RapidIO network; memory mapping; high-speed interconnection network

1 概述

传统高速互连网络采用基于客户机/服务器和消息传递的通信模型，通过消息数据的发送和接收实现客户机与服务器之间的数据传输。随着网络通信技术的发展，越来越多的网络硬件，如Dolphin公司开发的SCI网络[1]、近年出现的RapidIO网络[2]等，能支持基于存储映射的通信方式。实践结果表明，此方式具有延迟小和带宽利用率高等优点，在体系结构上优于传统网络。

RapidIO是由Motorola等公司率先提出的一种新型互连技术标准。在众多产品中，如Freescale公司推出的处理器芯片(MPC8548, MPC8641等)、Xilinx公司推出的Real RapidIO解决方案，都已集成了完整的RapidIO逻辑，通过RapidIO控制器的ATMU(Address Translation and Mapping Unit)，在硬件上已开始支持基于存储映射的通信功能[3]。但目前没有面向RapidIO、基于存储映射的网络通信技术方案，因此，仍然需要对RapidIO进行研究，开发基于存储映射的通信软件。

本文建立基于高速互连网络的存储映射通信模型，通过采用RapidIO网络互连，构建全局的、网络的存储映射通信技术解决方案。

2 建模的相关技术

2.1 TCP/IP协议

TCP/IP协议的API接口函数为Socket编程接口。Socket采用完全基于客户机/服务器和消息传递的通信模型，发送或接收数据时需要操作系统处理复杂的协议栈并执行操作系统、文件系统及存储管理的相关代码，这些操作增加了网络通信的延迟和软件开销，降低了网络带宽利用率。

2.2 GAMMA系统

在GAMMA[4]系统中，应用程序直接进入核心态(软中断号0x81)，不经过操作系统直接执行GAMMA驱动程序，从而减小了发送接收和数据时处理器的运算量和通信的延迟、提高了网络带宽利用率。Giuseppe Ciaccio等人的测试结果表明，经过优化的、配备了GAMMA千兆以太网的集群计算机性能可以超过其它互连方式；应用程序运行在GAMMA网络中自然状态下的速度和可测量性常优于InfiniBand集群。

GAMMA是基于消息传递机制的网络互连解决方案，由于发送和接收过程需要软件的参与，因此会耗费处理器的执行时间。

2.3 InfiniBand方式

InfiniBand[4]是一种混合实现方式，实现了基于客户机/服务器和消息传递的通信方案及基于存储映射实现网络通信的方案。它将复杂的I/O系统与处理器/存储设备分离，使I/O子系统独立，是一种基于I/O通道共享机制的总线互连技术。在InfiniBand通信网络中实现了存储映射后，访问远程存储器的效率更高，在存取相邻节点数据时，可以达到很低的消息延迟和很高的带宽峰值[5]。

2.4 SCI技术

基于IEEE标准的SCI(Scalable Coherent Interface)是基于存储映射的网络通信技术。在该网络中，软件只要负责建立本地和远程节点的映射关系，维护数据的本地网络节点并对网络开放本地内存段。访问数据的远程节点通过网络建立对本地共享内存段的存储映射，并通过对内存映射区域直接读写实现对数据的直接访问。一旦映射关系成功建立，应用程

基金项目：国家自然科学基金资助项目(90612010, 60402016)；国家“973”计划基金资助项目(2003CB317000)；上海市教委基金资助项目
作者简介：黄 亮(1981－)，男，硕士研究生，主研方向：高性能计算，新一代网络技术，软件工程；刘福岩，副教授、博士
收稿日期：2007-09-24 **E-mail：**huangliang@shu.edu.cn

序不通过任何软件操作，只要像访问本地内存一样，访问已经建立起存储映射的内存段，即可直接访问远程数据。

3 基于 RapidIO 和存储映射的高速互连网络

3.1 系统结构模型

MPC8548 支持 36 位本地总线(OCN)地址空间，存储器和其他设备地址可全部映射到总线地址空间，采用存储映射方式访问各种外部设备。

存储映射 I/O 由 MPC8548 上的 ATMU(图 1 中的 X 部分)实现。ATMU 支持 34 位 RapidIO 地址空间，共有 5 个 Inbound 窗口和 9 个 Outbound 窗口。5 个 Inbound 窗口可以为外部节点开放 5 段存储区，其功能是将得到的 34 位地址转换为总线地址，并映射事务和目标接口。每个 Outbound 窗口可以再划分为 32 个子窗口，因此，可以同时访问外部 9×32＝288 个节点上的存储区，其功能是将本地 36 位总线地址转换为 34 位的 RapidIO 地址，并分配访问远程节点的事务。

在如图 1 所示的系统体系结构中，系统中的各个节点通过 RapidIO 连接，每个节点都拥有独立的 Linux 操作系统、处理器和存储器。

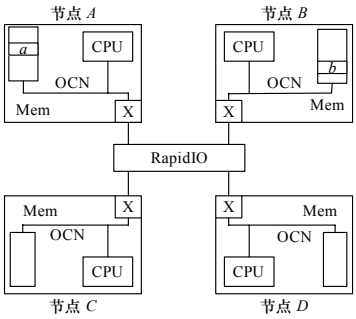

图 1 体系结构示意图

处理器处理所需数据从 OCN 总线中获得。在系统实际运行过程中，处理器 *A*(节点 *A* 上的处理器)某次执行所需数据在节点 *A* 的存储器区域 *a* 中，另外某次的数据可能在节点 *B* 的存储器区域 *b* 内。在访问区域 *b* 内数据的过程中，如同处理器访问本地节点的区域 *a* 中的数据一样，用户只从逻辑上认为是在本地节点存储的数据——这些正是基于 RapidIO 网络互连的存储映射所做的工作。

在该基于存储映射的通信网络系统的实现过程中，建立地址映射关系的工作可划分为 2 个部分：(1)为本地节点的地址建立存储映射关系，即从存储器等节点的 I/O 设备映射到本地 36 位 OCN 地址空间中供本地和远程节点使用；(2)通过建立一系列地址映射关系实现与远程节点的通信，即将远程节点数据通过 RapidIO 直接映射到本地节点中，而不必通过远程节点上的处理器处理操作和多次拷贝传输数据来完成对远程节点数据的访问。

3.2 本地节点内地址映射关系的建立

在 Linux 中，用户进程通过使用 mmap()系统功能调用，转而调用系统内核中与之对应的 mmap 方法(该方法是 file_operations 结构体的一部分，用于请求将设备内存映射到进程地址空间)。为建立节点内部的映射关系，驱动程序需要为该地址范围建立合适的页表，并将 vma->vm_ops 替换为其他操作。通过 vm_area_struct 结构提供操作，使系统在 OCN 地址空间中申请一段地址空间来建立进程空间与 OCN 的映射关系，从而使进程可以直接与通过 VMA 映射得到的 OCN 空间段进行各种操作。

该系统使用函数 io_remap_page_range()为一段物理地址创建新页表，并建立虚拟地址(virt_addr)与物理地址(phys_addr)之间的映射关系。函数中的参数 phys_addr 指向实际系统的 I/O 内存(对于远程节点则是指 RapidIO 的地址)。

内存区域由 vm_area_struct 结构体描述，它包含了用于访问设备的虚拟地址信息，指定了地址空间内连续区间上的一个独立内存范围。内核将每个内存区域作为一个单独的内存对象管理。

vm_area_struct 结构体定义部分如下：

```
struct vm_area_struct{  …
                unsigned long      vm_flags;  /* 标志 */ …  };
```

VMA 标志是位标志，包含在 vm_flags 域内，通过将 vm_flags 域内标志设置为 VM_IO，标识该区域映射在设备 I/O 空间。设置 VM_RESERVED 标志标识内存区域不能被换出。通过使用 mmap()函数建立 VMA 与 OCN 的地址映射关系，使用针对该 VMA 的 open()等函数对 VMA 进行操作。进程对 OCN 上内存区域的操作通过其进程空间内映射到 OCN 空间中的 VMA 完成。

通过上述一系列操作建立映射关系，完成 I/O 设备内存映射意味着将 OCN 总线地址空间中的一段内存与存储器的地址空间相关联。只要该映射关系存在，当用户进程在系统分配的地址范围内进行读、写等操作时，就是对该 I/O 设备内存直接进行访问操作，使用户进程具备直接访问本地节点上 I/O 设备内存的能力。

3.3 与远程节点映射关系的建立

如图 2 所示，当节点 *A* 上的处理器所需数据存在远程节点 *B* 的存储器中时，需要通过建立远程地址的映射关系来访问其数据。

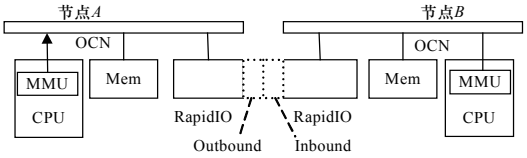

图 2 存储映射

节点 *A* 的处理器发送指令以获取数据地址，MMU(Memory Management Unit)解析该地址，在 OCN 总线地址空间查询解析得的地址，若其存在于 RapidIO 设备地址空间中，则由 ATMU 转换为 34 位的 RapidIO 地址及网络节点的标识(如节点 B)，并与节点 *B* 的 RapidIO 设备通过 Inbound 窗口通信，实现对节点 *B* 上数据的直接访问。

节点 *A* 的 RapidIO 将请求的数据地址发送到节点 *B* 的 RapidIO 中。该地址在节点 *B* 的 ATMU 中进行地址转换，ATMU 把该地址转换为节点 *B* 上的 36 位 OCN 总线地址，RapidIO 设备将根据得到的地址从 OCN 空间(已和 I/O 设备内存建立起了地址映射关系)中取得数据，并自动触发一个应答包的发送操作，将取得的数据打包传输到节点 *A* 中。在此通信过程中，不需要任何软件参与，所有操作由 RapidIO 硬件负责实施。

在系统启动之初，各个节点内都对 OCN 总线和 RapidIO 进行初始化配置，建立与远程节点的映射关系。建立的映射关系可分为 3 个部分：(1)OCN 地址空间与 RapidIO 中 Inbound 窗口的映射关系；(2)OCN 地址空间与 RapidIO 中 Outbound 窗口映射关系；(3)本地节点上的 RapidIO 中 Outbound 与远程节点上的 Inbound 窗口的映射关系。 (下转第 120 页)

曲线的吞吐率仍高于 2HOP_11G_TCP 和 3HOP_11G_TCP 曲线的吞吐率。

3.2 无线链路延迟

跳数增加对 WMN 影响的另一个表现是无线链路延迟的增加，跳数增加时无线链路延迟的变化如图 4 所示。

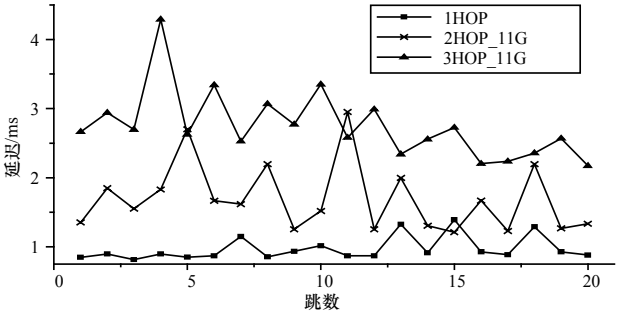

图 4 跳数增加时无线链路延迟的变化

测试时分别从无线链路两端向对端发送 PING 包，使用 ethereal 抓包工具来捕获数据包。为了保证数据的真实性，图 4 中每条曲线的数据均取自同一组实验。由图 4 可以看出，随着跳数增加，无线链路延迟的变化很明显，单跳无线链路平均延迟 0.975 ms，2 跳无线链路平均延迟 1.716 ms，3 跳无线链路的平均延迟为 2.78 ms，且 2 跳和 3 跳时延迟的抖动比单跳时剧烈。随着跳数的增多，数据包在端到端传输过程中需要的传输时间和调度时间随之增加，造成传输延迟的增加。

4 结束语

多发射多信道 WMN 架构可以保证 WMN 性能，满足用户需要，但跳数增加时 RTT 随之增加，TCP 性能会有所下降。当跳数超过 3 跳时，由于 IEEE 802.11b/g 只能提供 3 个不重叠的信道，因此如何合理分配信道，保证在无线信号的覆盖范围内不出现相同信道，是今后需要重点研究的问题。当前常用的方法是使用 IEEE 802.11a 协议作为 WMN 的骨干、以 IEEE 802.11b/g 作为 WMN 的接入，由于 IEEE 802.11a 协议可以提供 8 个不重叠的信道，因此能极大缓解信道冲突带来的问题，同时使用 IEEE 802.11b/g 协议作为接入可以保证与现有设备的兼容。但因为国内没有开放 IEEE 802.11a 工作的 5 GHz 频段，所以无法使用该方法进行实验。

IEEE 802.11b/g 协议使用开放频段，对无线信道进行管理有一定难度，因此，本文 WMN 架构较适合为大学校园或小区等提供无线 Internet 接入服务。

本文未对超过 3 跳的 WMN 进行性能测试，但可以预见，在没有信道间干扰的情况下，无线链路的性能不会因跳数增加而出现明显下降。WMN 骨干部分的位置较固定，可通过合理的部署避免出现信道间的干扰现象。在无法避免信道间干扰时，应考虑将相距最远的 2 跳无线链路设置为工作在同一信道，以最大程度地减少无线信道间干扰对 WMN 整体性能的影响。

参考文献

[1] Akyildiz I F, Wang Xudong, Wang Weilin. Wireless Mesh Networks: A Survey[J]. Computer Networks, 2005, 47(4): 445-487.

[2] ANSI/IEEE. Std 802.11 Wireless LAN Medium Access Control and Physical Layer Specifications[S]. 1999.

[3] ANSI/IEEE. Std 802.11b Wireless LAN Medium Access Control and Physical Layer Specifications: Higher-speed Physical Layer Extension in the 2.4 GHz Band[S]. 1999.

[4] ANSI/IEEE. Std 802.11g Wireless LAN Medium Access Control and Physical Layer Specifications: Further Higher-speed Data Rate Extension in the 2.4 GHz Band[S]. 2002.

(上接第 117 页)

在系统引导时，为 RapidIO 设备分配连续内存页面的缓冲区，使其设备驱动程序直接链接到内核中，而不是以模块形式来获得。通过调用函数 alloc_bootmem _low()获得该缓冲区；当系统运行完毕时通过内核提供的接口函数 free_bootmem()释放。通过上述步骤完成配置 RapidIO 的 Outbound 窗口和 Inbound 窗口与 OCN 总线地址的映射关系。

在系统初始化配置时，将申请好的内存页面缓冲区与 RapidIO 设备进行地址映射，采用 mmap()设备操作来处理。对 RapidIO 按先 Outbound 窗口后 Inbound 窗口、以其编号顺序遍历，分别建立与 OCN 总线地址的地址映射关系，并将建立好的地址映射关系保存在 CCSR(Configuration Control and Status Register)中。这些配置完成后，RapidIO 设备中的各窗口都与本地节点的 OCN 总线地址建立起地址映射关系，可供远程节点访问。

通过 RapidIO 初始化硬件抽象层提供统一的软件接口配置本地和远程节点的寄存器。建立映射主要由 rioGetNumLocalPorts(), rioConfigurationRead(), rioConfigurationWrite()三个函数完成，其中主要参数为 localport(指定本地 RapidIO 窗口，从该窗口发送或者接收事务)，destid(被映射的 RapidIO 窗口)。将建立好的映射关系保存在 CCSR 中，即建立了本地节点上 RapidIO Outbound 窗口与远程节点上 RapidIO Inbound 窗口的映射关系。2

4 结束语

在 RapidIO 互连网络中，采用基于存储映射的通信方式，软件只负责建立和取消地址映射。一旦地址映射关系建立，数据传输完全由 RapidIO 硬件实现，因此，与传统基于客户机/服务器和消息传递的通信方式相比，该方式延迟较小且带宽利用率较高。

参考文献

[1] Dolphin Interconnect Solutions. Low-level SCI Software Functional Specification[EB/OL]. (1999-03-11). http://www.dolphinics.com/support/documentation.html.

[2] RapidIO Trade Association. RapidIO Interconnect Specification[EB/OL]. (2005-06-23). http://www.rapidio.org.

[3] Freescale Semiconductor Inc.. MPC8548E Power QUICCIIITM Integrated Host Processor Family Reference Manual[Z]. (2007-02-04). http://www.freescale.com.

[4] InfiniBand. Host Channel Adapter Verb Implementer's Guide[EB/OL]. (2003-03-23). http://infiniband.source forge.net.

[5] Tipparaju V, Santhanaraman G, Nieplocha J, et al. Host-assisted Zero-Copy Remote Memory Access Communication on InfiniBand[C]//Proceedings of the 18th IEEE International Conference on Parallel and Distributed Processing. Santa, USA: IEEE Press, 2004.