一种 TCP/IP 卸载结构的设计与实现①

Design and Implementation of TOE Architecture

陈续喜 曾文海 程扬军 （湖南大学 电气与信息工程学院 湖南 长沙 410082）

摘 要： 为了将处理器从繁重的通信任务中解脱出来，解决通信系统中的瓶颈部分，适应高速通信网络，本文研究了一种新的 TCP/IP 卸载引擎（TOE）的原理和设计方法，并提供了 TOE 网络接口卡（NIC）的一套参考实现方案，此方案以 Intel IOP310 I/O 为处理器芯片组，在 Linux 上搭建自己的软硬件平台并得出实验结果。实验证明，该技术有效地提高了网络的传输性能，降低了计算机的 CPU 占用率。

关键词： TCP/IP 卸载引擎 实时操作系统 Linux 软件系统

1 引言

在目前的以太网环境中，TCP/IP 协议的处理都是通过软件方式在中心处理器上实现。当网络速度达到 G 比特数量级时，主 CPU 就越来越繁忙，其中大部分处理负荷都是来自对 TCP/IP 协议的处理[1,2]，例如对 IP 数据包的校验处理、对 TCP 数据流的可靠性和一致性处理。大量协议数据还需要通过 I/O 中断进行操作，不断在网络接口缓冲区和应用程序内存之间进行数据交换，这些额外的负担极大地降低了主 CPU 的处理效率，增加了应用计算的平均等待时间。频繁的协议处理和内存操作使得处理器力不从心[3],人们提出 TCP/IP 卸载引擎(TCP/IP Offload Engine，TOE)技术[4,5]，将部分甚至全部 TCP/IP 协议处理任务交给网络接口卡执行，加速网络协议的处理，提高网络吞吐量；同时极大地减轻 CPU 的负担，避免网络处理消耗过多计算资源，提高了系统的总体性能。本文就采用了 Intel IOP310 I/O 处理器芯片组，在 Linux 上搭建自己的软硬件平台，实现一种新的 TOE 网络接口卡。

2 TOE 体系结构

如图 1 所示即是整个的 TOE 技术的原理框图，TOE 将 TCP/IP 协议从 CPU 中移到硬件中处理。在主机中安装和 TOE 通信的驱动后，内核和用户的应用都可以直接和 TOE 通信，这样就可以使主机 CPU 来处理别的工作。TOE 在接收到网络上的数据后，进行一系列的协议处理，将数据放在指定地址，交给上层应用。发送方向则相反，将需要处理的数据包装后通过硬件缓冲发送出去。图 1 中 TOE 将整个或部分 TCP/IP 协议栈卸载到网卡中处理，绕过了主机 CPU 处理 TCP/IP 协议的路径。在主机上安装了 TOE 的驱动后，内核空间的应用(各种网络服务)和用户空间的应用都可以直接与 TOE 通信。TOE 网卡接收到数据后，自动完成协议栈各层协议处理，发送过程相反。通过 TOE 处理，中断和数据拷贝次数都会大大降低，CPU 的负担大大降低。

3 TOE 网络接口卡的硬件设计

我们选择的方案是利用嵌入式处理器(Embedded CPU)运行实时操作系统(Real-time Operating System，RTOS)，再加上一个 MAC/PHY 来实现 TOE 网络卡，在主机上处理的协议被卸载到 RTOS 中进行处理。这种实现不仅卸载 TCP/IP 协议栈，而且卸载其它凡是能嵌入到 RTOS 中的协议。这

① 收稿时间:2008-07-26

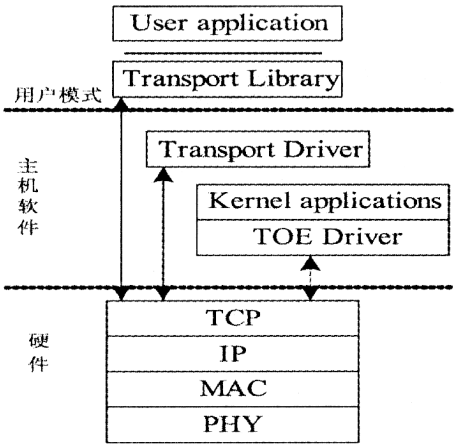

图 1　TCP/IP 卸载引擎体系结构

种方案的优点是其解决方案灵活且可扩展性强。此方案中嵌入式 CPU 采用了 Intel IOP310 I/O 处理器芯片组，该芯片组由一个 Intel 80200 CPU 和一个 Intel 80312I/O 配套芯片组成，Intel 80200 是基于 Intel XScale 微体系结构的高性能嵌入式微处理器，其主频为 600MHz，外频为 66MHz。Intel 803I2 I/O 配套芯片专用于 I/O 处理和内部及外部总线的桥芯片，具有内存/PCI/消息/DMA 控制器，与 SDRAM 和 FLASH 之间的接口速度为 100MHz。硬件的架构如图 2 所示，它完成 IP、TCP 处理，与主机之间通过 PCIE 接口进行 DMA 操作，其中 Compact Flash 里主要存放软件代码以及需要保存的参数，DDR SDRAM 用来做系统运行时的代码和数据存储。在这个结构中，服务器上 CPU 的计算或处理转移到嵌入式 CPU Intel IOP310 I/O 处理器处理单元上进行，与传统普通网卡对比，TCP/IP 卸载引擎引入了一种新的网络接口体系结构，它将 TCP/IP 协议栈的处理工作从服务器上卸载下来，交给硬件 Intel 80200 CPU 和一个 Intel 80312 I/O 来完成 TCP/IP 协议处理，简化了整个 TCP/IP 的处理路径，从而减轻了 CPU 和服务器 I/O 系统的处理负担，消除了服务器的网络瓶颈。

4　TOE 软件系统设计

根据 TOE 网卡对连接管理部分的支持方式，将 TCP 协议卸载方式分为全卸载和部分卸载两种[5]。全卸载方式下，TOE 网卡完成 TCP 协议的全部功能，而不需要 CPU 的参与，而在部分卸载情况下，TOE 网卡处理 TCP 协议的数据传输、定时器管理和错误与拥塞控制等过程，主机处理连接管理部分，还有 TOE 网卡不需

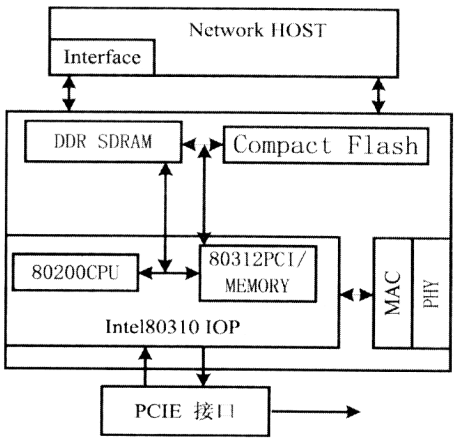

图 2　TOE 硬件系统设计

要实现完整的 TCP／IP 协议栈，它只负责连接建立后报文的发送和接收处理[6]。针对不同的应用环境，两种卸载方式各有优势。在连接持续时间较短、出错率高的环境下，使用全卸载方式更利于提高 TCP 处理的性能，降低 CPU 的占用率。但是，在可靠的大数据量传输应用环境下，使用部分卸载方式获得的性能与全卸载方式相当。由于本系统的 TOE 网卡主要应用于网络存储、文件服务、IP 存储和可视化计算，这些应用的特点是在局域网环境下进行海量数据传输。因此，与全卸载方式相比，部分卸载方式更适合于此方案。

4.1 设计的要求

绝大多数用户在使用 TOE 网卡时只希望提高性能而不希望改变现有的应用程序，因此 TOE 软件系统必须提供与现有应用程序兼容的接口。TOE 网卡对于不需要硬件 TCP 处理的报文，会像传统网卡一样直接交给 TOE 软件系统处理。TOE 软件系统不仅要处理 TOE 工作模式下的报文，还要能够处理传统的 TCP 报文、UDP 报文、ICMP 报文等。所以软件的设计必须得保证 TOE 工作模式对应用程序的兼容性和实现不同类型报文的发送、接收处理，需要或不需要硬件处理的 TCP 报文，UDP／ICMP 报文等。

4.2 TOE 的软件架构

图 3 虚线左边显示出传统的 Linux 是用一系列相互连接层的软件实现 Internet 协议地址族[7]。BSD 套接字（BSD sockets）由专门处理 BSD sockets 通用

套接字管理软件处理，它由 INET sockets 层来支持，这一层为基于 IP 的协议 TCP 和 UDP 管理传输端点，UDP（用户数据报协议）是一个无连接协议而 TCP（传输控制协议）是个可靠的端对端协议，TCP 包则被 TCP 连接两端编号以保证传输的数据被正确接收，IP 层包含了实现 Internet 协议的代码。现在我们要做的就是对这传统的 Internet 协议进行改进，进而实现与 TOE 工作模式对应用程序的兼容，将 UDP、IP、TCP 卸载到 TOE 网络接口卡上来实现 TCP 协议部分卸载方式，图 3 中虚线右边表示的是卸载后的 Linux TOE NIC 的方案。

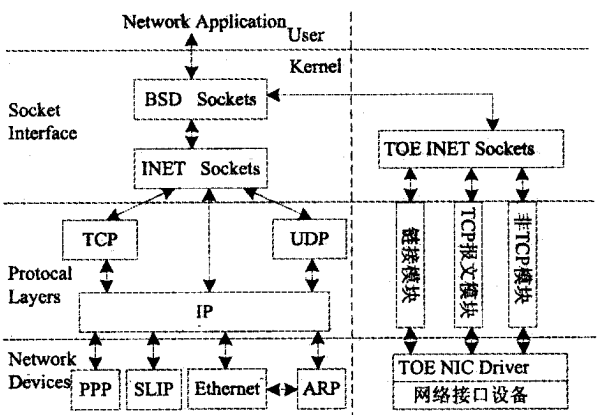

图 3　TOE 卸载后的网络协议与传统协议之间的比较

当 Network Application 要发送数据时，BSD Socket 层调用 TOE INET Socket 层的发送函数发送数据，TOE INET Socket 层调用修改后的设备驱动层即 TOE 设备驱动层的发送函数将数据发往网卡，再由网卡进行 TCP 协议处理后发往网络；同样，当网络设备收到数据报文后，先在网卡上进行 TCP 协议处理，然后将处理后的报文发往 TOE 设备驱动层，再经由 TOE INET Socket 层和 BSD Socket 层到达应用层供用户使用。此外，对于 TCP 控制报文、未卸载到网卡连接上的 TCP 数据报文以及 UDP，ICMP 等报文，它们需要经传统的 TCP/IP 协议栈处理，在主机操作系统上建立、关闭和维护连接，为此我们在主机操作系统中仍然保留传统的 TCP/IP 协议栈处理这些报文。

5　TOE 架构的主要功能模块

对于部分卸载的 TOE 架构网络协议如图 3 中所示，本文就对其主要的功能模块进行说明。

5.1 TOE INET Socket 层

由于对 TOE 报文(指需要经过 TOE 网卡处理的报文)、TCP 报文(指 TCP 控制报文、未卸载到网卡的连接上的 TCP 数据报文)和非 TCP 报文(指 UDP，ICMP 等报文)的处理所使用的 INET 层操作函数不一样，所以在 INET Socket 层与 BSD Socket 层交互时，需要 TOE Socket 层在中间屏蔽这种差异。

发送数据时，根据发送数据的类型决定走哪条通路，对于 TCP 数据，TOE Socket 层先判断它所属的连接是否被卸载到网卡上，如果已被卸载，则走 TOE 报文的通路，否则走 TCP 报文的通路。接收数据时，对不同通路交上来的报文处理后交给 BSD Socket 层，而 BSD Socket 层并不知道这些数据来自不同的通路。

5.2 TOE NIC Driver 层

TOE NIC Driver 层提供对网卡硬件的读、写、控制操作，向上层屏蔽网卡的硬件细节，处理中断，是 TOE 网络接口卡的驱动层。接收报文时，根据不同的报文类型将数据发往不同的数据通路。如果是 TOE 报文，发往连接控制模块 如果是 TCP 报文，构造 Sk_buf 报文结构，送往 TCP 报文处理模块 如果是 UDP，ICMP 等报文，发往非 TCP 报文处理模块。发送报文时，如果是非 TOE 报文，则以 MAC 帧的格式发给网卡；如果是 TOE 报文，则将该报文以及与该报文所属连接相关的一些信息一起发送给网卡，以便网卡进行 TCP 处理。这里对于 TOE 报文，网卡和驱动层之间传送的数据既不是 MAC 帧，也不是简单的 Sk_buf 结构，传送的数据中必须携带一些信息，如报文所属连接、报文所属设备号、报文类型、紧急数据指针等。驱动层和网卡通过这些信息交互后才能正确处理数据，这是一个新的接口。

5.3 TCP 报文处理模块

该模块处理不经过 TOE 网卡的 TCP／IP 协议栈处理的 TCP 报文，这些报文包括 TCP 控制报文和不能进行 TOE 处理的 TCP 数据报文，TCP 报文处理过程中的连接释放，包括主动释放与被动释放。主动释放，应用程序关闭连接时，TCP 卸载控制模块通过驱动通知硬件

该连接释放,硬件将发送缓存中的报文发送完毕后通知软件,TCP 卸载控制模块继而通过 TOE 设备核心层的接口调用网卡驱动程序提供的设备方法释放连接，将连接的控制信息读回 Linux 原始协议栈。被动释放,若网卡接收到控制报文(FIN 或 RST)，触发中断通知 TOE 网卡驱动程序，驱动程序构造一个主机报文格式(Sk_buff) 的控制报文，通过 TOE 网络核心层提交 IP 层,并最终经过 Linux 原始协议栈的协议处理,然后通知 TCP 卸载控制模块释放连接。TCP 卸载控制模块通过 TOE 网络核心层调用网卡驱动程序释放连接，将连接控制变量读回 Linux 原始协议栈。

6 性能测试与结论

我们采用嵌入式芯片设计开发出的 TOE 网卡，在 Linux 操作系统环境下做了性能测试。硬件环境是两台 AMD 2200+ 1.79GHz PC 机，512MB 内存，主板集成 64bit/67MHz PCI 以接插 TOE 网卡；软件环境是 RedHat Linux；测试工具是 iperf。两台测试机分别为 Server 和 Client，Client 端安装的千兆网卡是 Intel PRO/1000，Server 端则先后安装了 TOE 网卡和 Intel PRO/1000 作性能对比测试。主要测试其网络吞吐率和 CPU 占用率，测试结果如图 4。

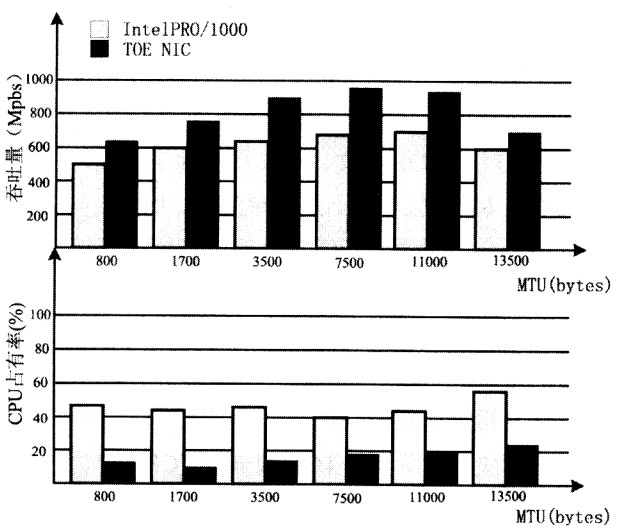

图 4　吞吐率与 CPU 率占有对比测试结果

测试结果表明，在相同 MTU 的测试下 T0E 网卡的网络吞吐率高于普通千兆网卡，而 CPU 占用率却低得多，各种 MTU 的测试都说明 TOE 网卡保持在高吞吐率水平，而 CPU 占用率一般都较低。

本文深入研究了以 Intel IOP310 I/O 处理器和 Linux 构成的软硬件平台，且以部分卸载方式实现 TOE 网卡的软件架构的设计。测试证明，此种新的 TOE 卸载技术可以更有效地降低主机 CPU 利用率，提高传输效率。

参考文献

1 Tak S W，Son J M，Kim T K．Experience with TCP/IP Networking Protocol S/W over Embedded OS for Network Appliance//Proc．of International Workshops on Parallel Processing.1999:557–561．

2 Camarda P，Pipio F，Piscitelli G．Performance Evaluation of TCP/IP Protocol Implementations in End Systems．IEEE Computers and Digital Techniques，1999,146(1):32–40．

3 Chase Js，Gallatin AJ，Yocum KG．End-System Optimizations for High-speed TCP．IEEE Communications Magazine，2001，39(4)：68–74．

4 Schuehler D V Lockwood J W.A Modular System for FPGA-based TCP Flow Processing in High-speed Network．Berlin Heidelberg：Springer-Verlag，2004．

5 Eric Yeh,Herman Chao,Venu Mannem,eta1. Introduction to TCP/IP Offload Engine．http://www.techonline.com/communjty/related_content/21208．2005-07-09．

6 TCP/IP Offload for High-speed Ethernet Networks．http//whitepapers.zdnet.co.uk/0，39025945，60038177p-39000380q,00.htm.2005-07-01．

7 Ang BS.An Evaluation of an Attempt at Offloading TCP/IP ProtocolProcessing onto an i960RN-based iNIC.http://www.hp1.hp.com/techreports/2001/HPL-2001-8.pdf,2005-05-09．