

基于共享缓冲区的多接口网络接口卡驱动设计

赵 博, 刘勤让, 李高鹏, 李向涛

(国家数字交换系统工程技术研究中心, 河南 郑州 450002)

摘 要: 文中提出一种 Linux 系统下基于共享缓冲区的网络接口卡驱动程序设计方案, 并成功应用于多接口网络接口卡的具体实现中. 介绍了 Linux 下网络接口卡驱动程序结构, 分析多接口网络接口卡驱动程序设计需求, 完成驱动程序设计和实现. 该方案减少了硬件资源占用, 降低了实现成本.

关键词: 共享缓冲区; 多接口; 网络接口卡; 驱动程序

中图分类号: TP393

文献标识码: A

文章编号: 1000-7180(2011)07-0171-04

Driver Design of Multi-ports Network Interface Card Based on Buffer Sharing

ZHAO Bo, LIU Qin-rang, LI Gao-peng, LI Xiang-tao

(National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 45002, China)

Abstract: A driver design of multi-ports network interface card based on buffer sharing technology in Linux is presented, and is implemented successfully. The architecture of network interface card driver in Linux is introduced, and the requirements of driver design are analyzed. The design esign of buffer sharing reduces the hardware resources requirement, so that reduces the implement cost.

Key words: buffer sharing; multi-ports; network interface card; driver

1 引言

多接口网络接口卡(简称多接口网卡)在路由器、网络测量和入侵检测等方面均有广泛应用. 多接口网卡具有多个独立通信的数据通道, 软硬件需维护多个数据通道资源, 实现成本较高.

文中针对多接口网卡通信需求, 设计了基于共享缓冲区的多接口网卡驱动实现方案. 该方案简化了结构组成, 降低了设计难度, 减少了多接口网卡需要维护的数据缓冲区数量, 减少了硬件资源占用, 降低了实现成本.

2 软硬件平台

操作系统为 Linux 系统, 采用 Linux 2.6 内核. 多接口网卡为自主设计, 集成 8 个千兆以太网接口, 使用高速外设部件互连标准(Peripheral Compo-

nent Interconnect Express, PCIE)通道进行数据传输, 采用自主设计的现场可编程门阵列(Field Programmable Gate Array, FPGA)硬件程序进行通信管理. 多接口网卡的结构如图 1 所示.

多接口网卡由 4 个模块组成, 分别为 FPGA 处理模块、媒体接入控制模块(Medium Access Control, MAC)、端口物理层模块(Port Physical Layer, PHY)和接口部分. FPGA 模块使用自主编写的 FPGA 硬件程序, 实现通信管理; MAC 模块、PHY 模块和硬件接口为数据传输提供硬件支持.

3 多接口网卡驱动结构

3.1 网卡驱动结构

Linux 系统的网卡驱动结构由四部分组成, 分别是网络协议接口层、网卡接口层、设备驱动功能层

收稿日期: 2010-12-01; 修回日期: 2010-12-25

基金项目: 高可信网络业务管控系统(2009AA01A346); 国家发改委专项(CNGI09-02-03)

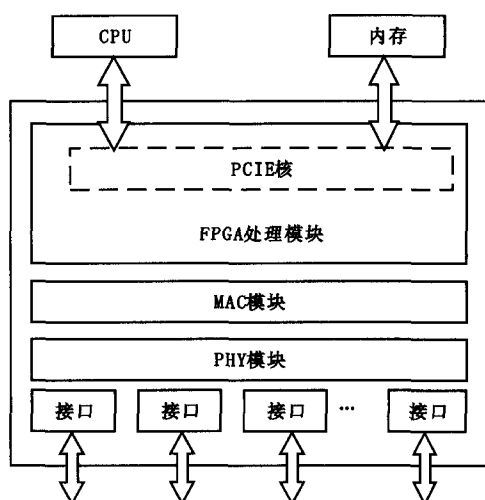


图1 多网口卡结构图

和网卡与媒介层^[1-3]。结构组成如图2所示。

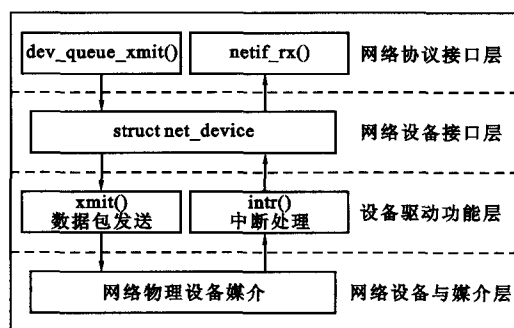


图2 网卡驱动结构

网络协议接口层向所有的网卡提供统一的数据包收发接口,系统中所有的网卡收发协议报文均通过该接口进行,上层协议和网卡驱动程序通过调用dev_queue_xmit()和netif_rx()函数完成数据发送和接收。它使得上层协议独立于具体的设备。

网卡接口层向网络协议接口层提供统一的用于描述设备属性和操作的接口,并封装于net_device结构体中。该结构体包含设备名称、状态等属性内容及打开、关闭等功能操作接口。设备初始化时,将设备信息及各功能函数注册至net_device结构体中,上层通过该结构体获取设备属性信息,并调用设备驱动功能层具体功能函数进行各项操作。网卡接口层屏蔽了设备间的差异,提供了统一的接口。

设备驱动功能层包含了设备的具体功能函数,如初始化、打开、关闭操作及数据发送、中断处理等,此外包括设备的数据信息如发送缓冲区、接收缓冲区的起始地址、大小,以及各指针等。该层函数完成与具体设备有关的数据和控制操作。设备的数据信息和部分操作函数被封装于private结构体中,并指向上层的net_device结构体。该层内容与设备密切

相关。

在设备初始化及进行数据收发时,通常根据设备接口数量初始化接收和发送缓冲区,用于存放收发的数据。

网卡与媒介层是完成数据包收发的物理实体,包括网络适配器及具体的传输媒介,被设备驱动功能层所驱动。对于操作系统而言,该层内容可以是虚拟的。

3.2 多接口网卡功能

多接口网卡驱动程序应具备以下功能^[4]:

- 在设备打开时完成所有接口初始化工作;
- 应用层能够通过指定接口收发数据;
- 多个接口同时收发数据,互不干扰;
- 接收数据时,识别自定义报头,并将原始报文送交指定的应用程序;
- 发送数据时,判断其发送接口,并添加自定义报头。

多接口网卡通过一个统一的网络接口与上层协议交互配置和数据信息。驱动程序向系统注册多个net_device结构体,与每一个物理接口相对应,用于与上层应用程序和协议栈交互。同时,驱动程序注册多个私有结构体,用于储存硬件接口相关信息和参数。每个私有结构体维护一套输入输出缓冲区,通过轮训、中断或者DMA(Direct Memory Access,直接内存存取)方式与硬件接口交互数据。相应的,网卡硬件也维护多个缓冲区,用于与驱动程序的数据交互。驱动程序结构如图3所示。

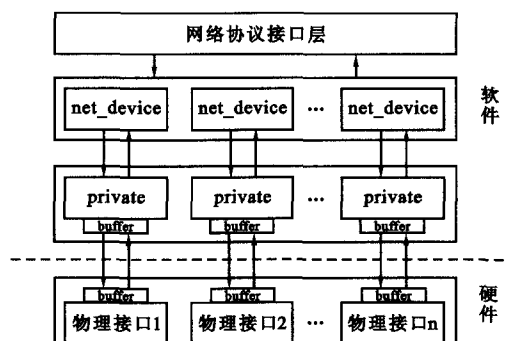


图3 多接口网卡驱动程序结构图

然而,在采用FPGA硬件程序进行时,这样的结构存在两个问题。首先,由于负责通信的PCIE核的技术限制,硬件必须为一个设备,而不能是多个设备(接口)。其次,当物理接口较多时,缓冲区资源消耗较大,采用FPGA实现硬件功能时,需要采用较高级别的器件,增加了成本。为了解决这两个问题,提出了共享缓冲区的设计方案。

4 共享缓冲区方案

共享缓冲区方案中,在硬件设计上,网卡的多个物理接口虚拟为一个逻辑接口,共享一套输入输出缓冲区。相应的,软件驱动设计上,注册一个私有结构体与虚拟的硬件接口通信,维护包括发送缓冲区、接收缓冲区的起始地址、大小,以及各指针等在内的设备数据信息。与通常的单接口网卡不同的是,private 结构体还需要维护一个 net_device 列表,用于设备初始化和数据收发时的接口识别。对上层应用程序和协议栈来说,能够识别及区分网卡的所有硬件接口,因此驱动程序仍向系统注册多个 net_device 结构体。驱动程序结构如图 4 所示。

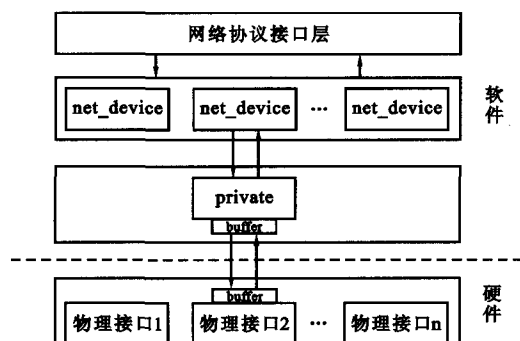


图4 共享缓冲区方案的驱动结构

多个物理接口的数据通过统一的通道与驱动顶层交互,为了区分数据来源,文中重定义了报文格式。在普通数据报文的头部添加了自定义报头,长度2字节,内容为物理接口标识。驱动程序通过自定义报头标识和区分报文归属接口。报头格式如图5所示。

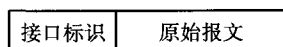


图5 自定义报头格式

4.1 多接口网卡驱动的注册与注销

网卡驱动的注册通过 register_netdev() 函数实现,该函数需重复执行,执行次数对应物理接口的个数。在此之前需要重复执行 alloc_etherdev() 函数,为每一个物理接口分配 net_device 结构体,并进行必要的初始化,包括挂载所有的功能函数等。修改所有 net_device 结构体的 priv 成员指针,指向同一个 private 结构体,即所有接口使用同一个发送和接收缓冲区。

网卡驱动的注销通过 unregister_netdev() 函数实现,类似地,该函数也需要重复调用,直至所有接口对应的 netdev 结构体均被注销完毕。

4.2 多接口网卡的打开与关闭

当设备被激活时,需要执行打开操作。打开操作

通过调用 open() 函数进行,主要完成资源分配、中断申请与挂载、看门狗启动等工作。由于本例中8个接口共享收发缓冲区及中断资源,因此 open() 函数执行一次即可。

资源释放、中断释放等工作在设备关闭时进行。类似地,关闭操作对应的函数 close() 也仅应被调用一次。

4.3 数据收发流程

驱动程序通过中断获知有数据包到达后,从接收缓冲区中取出数据包,根据自定义报头中有关设备接口的相关信息识别其来源接口,修改 sk_buff 结构体的 dev 成员为接口对应的 net_device 结构体,使得数据报文可被送至正确的应用程序接口。使用 skb_pull() 函数去除自定义报头后,向协议栈提交。

发送数据时,具体的发送函数 xxx_xmit() 被协议栈调用。从发送函数传递的参数中得到发送接口对应的 net_device 结构体,在 private 结构体中的 net_device 列表中查找,判断数据报文的发送接口,为数据报文添加自定义报头后发送。自定义报头的空间通过 skb_push 获得。在内存中存储的原始报文前面有2字节的空余空间,用于字节对齐,因此使用 skb_push 函数不会引起错误异常。

5 仿真分析

多接口网卡的每个接口对应一个双向数据通道,每通道单方向拥有2048个描述符,每个描述符占用128比特的空间,则每个通道需要的缓存空间为比特512K。每个Block RAM大小为32K比特,则每个通道需要16块Block RAM。在共享缓冲区设计中,需要1个通道的资源,即16块Block RAM;在非共享缓冲区设计中,需要8个通道的资源,即128块Block RAM。共享缓冲区的设计节省了112块Block RAM,资源利用率明显提高。Block RAM资源占用对比如图6所示。

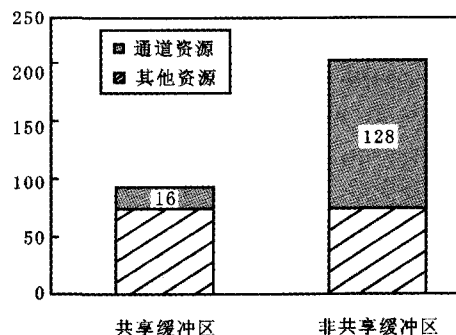


图6 Block RAM资源占用对比

设计并实现了一块多接口网卡,有 8 个千兆位以太网接口,采用本文提出共享缓冲区方案设计数据通道,使用自主编写的 FPGA 硬件程序实现通信管理. 在 Xilinx Integrated Software Enviroment 10.1 上查看综合报告(Synthesis Report),使用 88 块 Block RAM,其中 16 块为数据通道占用,剩余 72 块为网卡其它模块以及连接冗余. 如使用非共享缓冲区方案,则至少需要 200 块 Block RAM. Block RAM 的需求增加了 127%. XC5VLX110T 的已不能满足要求,需要更换为 XC5VLX330T^[5]. 由此可见,共享缓冲区方案明显节省了硬件资源,降低了实现复杂度.

6 结束语

文中提出了一种基于共享缓冲区技术的多接口网卡驱动实现方案. 多个物理接口被虚拟为一个逻辑接口,与协议栈进行数据交换,通过自定义报头实现物理接口的识别. 该方案简化了硬件结构,降低了设计难度,减小了硬件 FPGA 资源占用,降低了实现成本.

参考文献:

- [1] Jonathan Corbet, Alessandro Rubini & Greg Kroah-Hartman. Linux 设备驱动程序[M]. 魏永明, 译. 北京: 中国电力出版社, 2006.
- [2] 郑魁, 杨志义, 曾小芸. 基于 Linux 的光纤通道网卡驱动程序开发. 微电子学与计算机[J]. 2007(1)
- [3] 宋宝华, Linux 设备驱动开发详解[M]. 北京: 人民邮电出版社. 2008.
- [4] 赵利娜, 贾豫东, 欧攀. 嵌入式 Linux 系统中的多层次驱动程序[J]. 计算机工程, 2009, 35(9): 69-71.
- [5] Xilinx Support Web Site. Virtex5 series parameter data sheet [EB/OL]. [2010-12-01]. <http://www.xilinx.com/support/documentation/virtex-5.html>

作者简介:

赵 博 男, (1981-), 博士研究生, 研究方向为业务流分类.

刘勤让 男, (1975-), 博士, 副教授, 研究方向为片上系统, 网络安全.

李高鹏 男, (1985-), 助理工程师, 研究方向为嵌入式系统驱动程序开发.

(上接第 170 页)

念及其性质研究;(2)最小代价用户输入集的高效精确求解算法研究.

参考文献:

- [1] Abrehet Mohammed Omer, Alexander Schill. Web services composition using input/output dependency matrix [C]// Proceedings of the 3rd workshop on agent-oriented software engineering challenges for ubiquitous and pervasive computing. United Kingdom, London: IEEE, 2009: 21-26.
- [2] Ivo Calado, Heitor Barros, Ig Ibert Bittencourt. An approach for semantic web services automatic discovery and composition with similarity metrics[C]// Proceedings of the 2009 ACM symposium on Applied Computing. Honolulu, Hawaii: ACM, 2009: 694-695.
- [3] 李增智, 董少然. 基于 Petri 网和 WS-BPEL 的自动服务组合研究[J]. 微电子学与计算机, 27(9): 100-102.
- [4] 李琦, 夏阳, 束长军, 等. 一种基于与或关系的 Web 服务

组合算法研究[J]. 微电子学与计算机, 2010, 27(8): 222-225.

[5] 邓水光, 吴健, 李莹, 等. 基于回溯树的 Web 服务自动组合[J]. 软件学报, 2007, 18(8): 1896-1910.

[6] Michael DiBernardo, Rachel Pottingera, Mark Wilkinson. Semi-automatic web service composition for the life sciences using the BioMoby semantic web framework [J]. Journal of Biomedical Informatics, 2008, 41(5): 837-847.

[7] 许斌, 李涓子, 王克宏. Web 服务语义标注方法[J]. 清华大学学报: 自然科学版, 2006, 46(10): 1784-1787.

[8] 叶恒舟, 罗晓娟, 牛秦洲. 基于与或图的 Web 服务自动组合[J]. 计算机工程与设计, 2010, 31(11): 2645-2647.

作者简介:

叶恒舟 男, (1980-), 硕士, 讲师. 研究方向为 Web 服务、网络构架等.