软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2016,27(Suppl.(2)):25−34

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于自描述缓冲区的网络处理开发环境*

唐 路, 徐东来, 吕高锋, 李 韬, 孙志刚

(国防科学技术大学 计算机学院,湖南 长沙 410073)

通讯作者: 唐路, E-mail: yan_jinli@126.com

摘 要: 随着通用多核技术的发展,多核处理器在网络核心设备能够提供较高的 I/O 吞吐能力,并支持更复杂的网络处理,为网络处理与转发带来了前所未有的灵活性和普适性.但是,在核心网络中,多核处理器的处理与转发性能仍然面临着巨大的挑战.首先,随着网络带宽的不断提升,多核处理器需要提供越来越高的处理能力.其次,随着网络业务复杂度的提高,在网络报文处理与转发中,应用对报文的处理开销越来越大,对设备的 I/O 裸转发能力提出了更高的要求.提出了一种硬件缓冲区管理机制 Self-Described Buffer(SDB),具有硬件低开销、软件高性能等优点.基于 SDB 的机制,设计并实现了一个通用的网络处理开发环境 NPDK.NPDK 采用零中断、零拷贝方式,提供内核与用户驱动,适用于通用多核 CPU 系统.不但具有简单、灵活、易开发等特点,而且在内核态下支持用户对每个 CPU 核上进行异构报文处理编程和在用户态下支持独占式多线程编程与共享式多进程并行编程.测试结果表明,基于 SDB 的网络处理开发环境在 10G 速率报文 I/O 转发性能接近线速,特别是 64 字节小报文可达到 7.49Gbps.目前,NPDK 已经在 click 路由器、OpenFlow 交换机和网络探针等方面得到应用.

关键词: 网络处理开发套件;自描述缓冲区;零中断;零拷贝;用户驱动

中文引用格式: 唐路,徐东来,吕高锋,李韬,孙志刚.基于自描述缓冲区的网络处理开发环境.软件学报,2016,27(Suppl.(2)):25−34. http:// www.jos.org.cn/1000-9825/16015.htm

英文引用格式: Tang L, Xu DL, Lü GF, Li T, Sun ZG. Network processing development kit based on self-described buffer. Ruan Jian Xue Bao/Journal of Software, 2016,27(Suppl.(2)):25−34 (in Chinese). http://www.jos.org.cn/1000-9825/16015.htm

Network Processing Development Kit Based on Self-Described Buffer

TANG Lu, XU Dong-Lai, LÜ Gao-Feng, LI Tao, SUN Zhi-Gang

(College of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract: Incremental performance gain, especially in terms of higher I/O bandwidth and more complex packet processing capability, has made general purpose and multicore processors a preferable option building network devices in core networks, which brings unprecedented flexibility and universality into forwarding plane. However, challenges still exist because of lagging processing and forwarding performance. Firstly, with network link rate continuing booming, the multicore processor needs to offer corresponding higher processing throughput. Secondly, the increasing complexity of network services and applications will introduce more packet processing overhead inside network devices in forwarding and processing processes, indirectly resulting in higher requirements on pure I/O capabilities. Self-Described Buffer (SDB) is a hardware-based buffer management mechanism proposed in this paper which features low overhead in hardware and high performance in software. Furthermore, NPDK, a general network processing development environment has been designed and built based on SDB. It adopts zero interrupt and zero copy technology, provides kernel space and user space drivers and can be applied for general purposes and multicore systems. NPDK is not only easy to implement, flexible to program, feasible to deploy, but it also supports per-CPU-based hierarchical packet processing programmability in the kernel space, as well as monopolistic multi-threaded and shared multi-process programming in the user space. Experimental results show that NPDK can reach near line rate

* 基金项目: 国家高技术研究发展计划(863) (2015AA010201); 国家自然科学基金(61202483, 61202485)

Foundation item: National High-Tech R&D Program of China (863) (2015AA010201); National Natural Science Foundation of China (61202483, 61202485)

收稿时间: 2015-05-31; 采用时间: 2016-01-05

forwarding under 10Gbps link rate. Specifically 7.49Gbps for 64 Bytes packets. Currently NPDK has already been deployed and applied in Click routers, OpenFlow switches and network probing applications.

Key words: network processing development kit (NPDK); self-described buffer; zero interrupt; zero copy; user driver

通用多核技术的发展为网络核心设备处理更加复杂的业务提供了有效支持,在灵活性与普适性上都具有较大优势,但是通用多核处理器在处理与转发能力方面差强人意.针对通用多核处理器,从软件角度或软硬件联合的角度对性能进行优化与提升成为当前通用多核处理器性能的研究热点,如软件路由器[1]和虚拟交换机[2].

为了提高数据 I/O 转发性能,在数据处理路径上的各种优化手段层出不穷,各有侧重点.比如,Netmap[3]主要使用内存映射机制和高效的缓冲区管理机制,内核与用户共享缓冲区传输,循环使用报文缓冲区来接收报文.Han 等人[4]提出了一种使用静态预分配及大分组缓冲区的方式,有效降低了缓冲区动态分配与回收的开销.Liao 等人[5]提出了一种新的面向高速网络处理的服务器 I/O 架构,针对在传统报文收发流程中对 I/O 性能影响最明显的几个部分,如数据拷贝、缓冲区释放和处理过程中因 CACHE 失效造成内存停顿所带的来开销等方面进行优化,极大地提高了服务器的网络处理性能与 WEB 服务的吞吐量.

英特尔在 X86 平台下使用自己的 CPU 与网卡研发出数据平面开发套件(data plane development kit,简称 DPDK)[6],Intel DPDK 在数据面中进行了英特尔架构特定的优化,包括缓冲区管理、无锁队列等.提供一套完整的 Linux 用户空间库,为用户提供底层构件来创建高性能的数据平面应用程序.DPDK 使用的是标准的 Intel 以太网卡,如 82599.此类型网卡芯片在设计之初并没有考虑将来要对缓冲区进行任何卸载操作.在缓冲区管理上仍然使用软件的方式,对数据 I/O 的性能造成了一定的影响,而且 DPDK 的很多优化都基于 IA 架构的系统,其通用性存在不足.

自描述缓冲区管理(self-described buffer,简称 SDB)[7]是我们提出的一种自描述符的缓冲区管理机制.该机制通过将分组数据及相关描述信息优化压缩、合并至连续存储中,可以有效地简化缓冲区管理操作,降低缓冲管理开销.SDB 功能可以卸载到硬件完成,实现缓冲区管理的零开销.

针对目前通用多核处理器上对报文的 I/O 加速与处理流程的优化仍存在性能不足、灵活性不够和通用性不强的问题,本文基于 SDB 的缓冲区管理机制,提出并设计了一套网络处理开发套件(network processing development kit,简称 NPDK).其主要功能包括:kLibNPE、uLibNPE、资源管理、设备管理、报文收发接口和用户态驱动.

本文主要创新如下:

(1) 基于 SDB 缓冲区管理机制,开发一套通用的网络处理器开发套件 NPDK,提供核内与核外的快速 I/O 接口,支持用户快速开发与应用部署.

(2) NPDK 提供以太网驱动支持,通过报文转换模块实现与内核协议栈无缝对接.

(3) NPDK 多核转发支持动态多核线程报文处理异构编程.

(4) NPDK 在用户态支持设备独占式多线程编程和共享式多进程编程.

(5) 使用 DPDK 开发环境,设计并实现了用户态数据 I/O 转发性能测试原型系统.

本文首先阐述本文的研究背景,然后概括本文的主要工作和创新.第 1 节主要介绍 NPDK 的基本原理、设计思想和主要特点.第 2 节主要介绍 NPDK 的系统实现与接口.第 3 节使用 NPDK 环境实现一个测试原型系统 PoF(packet on fly).第 4 节介绍在 PoF 系统下的性能测试结果与分析.第 5 节是总结与下一步工作.

1 NPDK 基本原理

1.1 SDB 的基本思想

SDB 是一种不同于传统缓冲区管理方式,面向通用多核处理平台加速报文分组转发的缓冲区管理机制,如图 1 所示.该机制只需要软件固定分配一定数据的缓冲区空间,由硬件实现缓冲区的动态分配与回收,并循环使用.通过将报文体数据与对应描述符信息进行压缩、合并,将两者存储在一体式连续缓冲区中,减少了缓冲区的

管理操作,并降低了管理开销.SDB 与传统的 SKB 相比主要有如下优点:一体式的缓冲区结构简化报文缓冲区的分配与回收工作;所有缓冲区可通过报文链的形式逐一访问报文,不需要描述符环操作,去除了描述符环的管理与访存开销;直接通过访问报文字段判断报文数据是否有效,不需要中断交互机制,去除了中断处理开销,提高了整个系统的分组转发性能;缓冲区管理功能全部卸载到硬件实现,在报文收发过程中,软件不需要任何管理操作,真正实现了软件的缓冲区管理零开销.

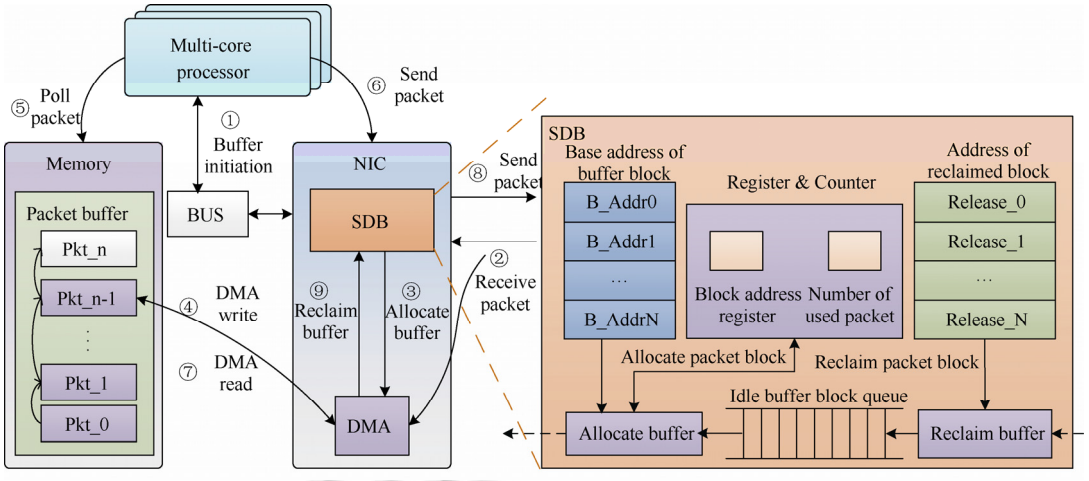


Fig.1 Hardware buffer management of SDB

图 1 SDB 硬件缓冲区管理

基于 SDB 的硬件缓冲区管理,通过硬件方式将所有报文以链的形式组织报文,不仅将软件缓冲区管理开销降至 0,还提供了两种报文加速方法.

(1) 无中断的轮询机制

由于接收报文的有效标记就存储在报文 metadata 结构中,可以使用软件轮询的方式来判断是否接收到报文,链式组织报文方法可以很灵活地获取到下一个报文地址,而无需使用中断、Mailbox 或 Doorbell 等方式来实现软硬件报文处理交互功能.而且,发送报文的回收由硬件处理,也不需要软硬件交互机制完成.

(2) 缓冲区的 CPU 亲和

由于 CPU 访问缓冲区数据时需要使用 TLB 表进行虚拟地址到物理地址的转换,而 TLB 的表项是有限的,如果缓冲区能够固定在指定的 CPU 核上访问数据,则可以有效地降低 TLB 表项的更替,提高报文的访存速度.

1.2 NPDK 的设计思想

在通用多核处理器上,网络报文 I/O 主要分为两条路径:数据核心路径与辅助路径.数据核心路径主要包括硬件与内存之间的 DMA 读写和内核态与用户态之间的数据拷贝.辅助路径主要包括:硬件中断、中断服务例程、软中断驱动处理程序(包括描述符管理和 SKB 管理)、协议处理和 SOCKET 处理.

NPDK 从缓存区管理、系统调度、中断模式与内存拷贝等方面全新考虑,结合通用多核上 Linux 系统的特性和硬件 FPGA 的可编程性,主要对数据处理核心路径与辅助路径进行优化,实现一种新的网络报文 I/O 方法,最大化优化数据 I/O 的辅助功能,给用户数据处理预留更多的时间和资源.其主要实现技术除了硬件 SDB 缓冲区管理、无中断轮询机制和缓冲区的 CPU 亲和以外,还包括以下的主要技术.

(1) 报文处理线程的 CPU 亲和

每一个硬件报文队列都由一个指定的报文处理线程进行处理,通过设计 CPU 的亲和力,可以使指定的线程只在一个 CPU 核上调度运行,减少在不同核上切换的开销,提高软件处理效率.软件线程的 CPU 亲和与缓冲区

的CPU亲和一起使用,分别从线程调度的角度和缓冲区TLB表项的开销优化性能,提高报文的访存速度与处理能力.

(2) 数据零拷贝机制

NPDK 使用内存映射机制,将硬件管理的所有缓冲区全部映射到用户空间,用户可以直接使用映射之后的地址对报文进行读写操作,消除了内核与用户之间的两次内存拷贝开销,减小了网络报文的数据搬移时间.

(3) 用户态驱动机制

根据 Linux 内存映射的原理,将硬件的 PCIE 地址访问空间映射到用户空间,可以直接获得对硬件 PCIE 空间的访问权限,可以直接在用户态读写硬件寄存器.NPDK 在用户态下对硬件环境进行初始化,并通过写寄存器方式发送报文,有效地减少了用户态报文发送的开销.

1.3 NPDK 的主要特点

NPDK主要针对通用多核CPU的I/O密集型应用开发,灵活支持多种开发模式,提供简便而有效的快速API开发接口.其主要特点如下.

(1) 支持内核态和用户态分组转发

NPDK 既提供内核态编程开发接口,又也支持用户态编程开发.内核态提供用户转发模块注册接口,将报文提交给注册模块进行处理与转发.同时,NPDK 也提供标准以太网驱动,将报文转换后提交给协议栈处理,并转发协议栈向驱动发送的 SKB 报文.在用户态下,提供报文快速收发接口,支持用户部署各种报文分组转发应用.

(2) 支持 SDB 与标准 SKB 的转发

NPDK 核心缓冲区使用 SDB 格式,同时通过对 SDB 报文的无损转换,以极小的开销实现 SDB 与标准 SKB 报文格式的转换.两种报文格式的无缝转换可以使 NPDK 通过以太网驱动和内核协议栈对接,有效地支持各类以太网接口的应用开发.

(3) 支持动态多核线程报文处理异构编程

内核态开发接口在每个 CPU 核的处理线程上都提供报文转发处理函数的注册接口,用户可以动态地在指定 CPU 核的处理线程上插入或删除自己的报文处理功能函数.同时,在不同 CPU 核上,可以支持用户注册不同的报文处理功能函数,实现异构的报文处理要求.

(4) 支持用户态独占式多线程编程与共享式多进程编程

独占式多线程编程由主进程统一初始化硬件与缓冲区,每个线程处理独立的报文队列.共享式多进程编程统一由核心守护进程实现硬件与缓冲区的初始化,每个处理进程只对报文缓冲区进行进程本地化空间映射,然后由各个进程独立处理各自的报文队列.各进程之间互不影响.

(5) 多进程一致性缓冲区映射,简化硬件管理

一致性的缓冲区映射是实现多进程的关键技术,核心进程仅使用一组缓冲区映射关系下发到硬件,对硬件进行初始化,简化硬件管理复杂度.其他所有处理进程使用核心进程的缓冲区映射关系,将所有缓冲区统一映射成与核心进程一致的页表地址转换关系.使所有处理进程实现对缓冲区的共享和用户空间地址的一致性要求.

2 NPDK 系统结构与开发接口

DPDK 系统环境分为 kLibNPE 和 uLibNPE 两大部分.kLibNPE 支持基于标准以太网设备驱动开发与多核报文处理模块编程;uLibNPE 支持独占式报文处理进程编程与共享式报文处理多进程编程.具体结构如图2所示.

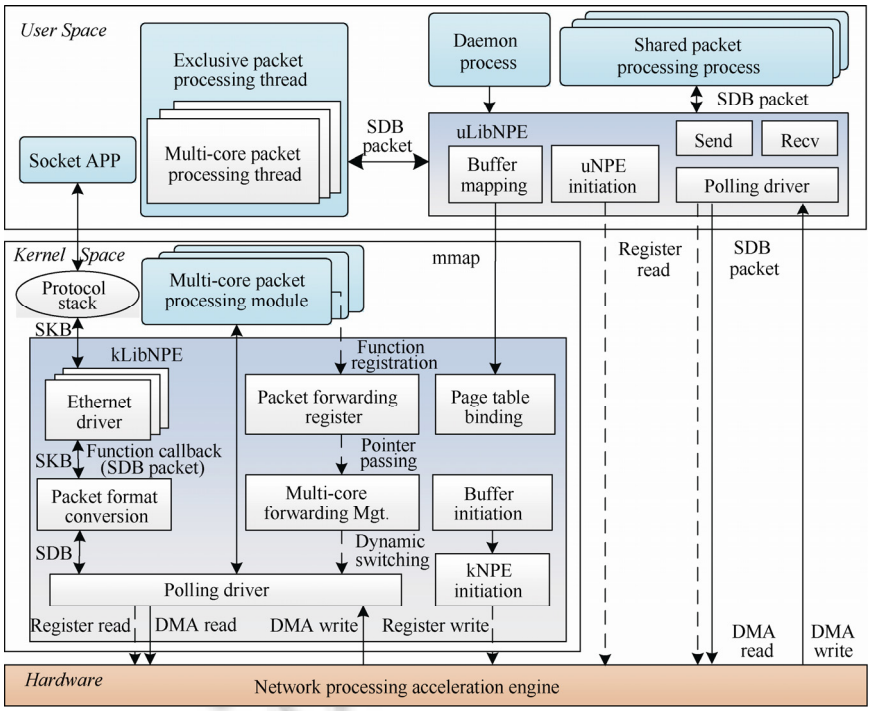


Fig.2 System architecture of NPDK

图 2 NPDK 系统结构

2.1 系统结构

kLibNPE 主要包括:内核态轮询驱动模块、报文格式转换、以太网接口驱动、报文转发注册、多核转发管理模块、缓冲区初始化模块、页表绑定模块和 kNPE 初始化模块.uLibNPE 主要包括:用户态轮询驱动模块、报文收发接口模块、缓冲映射模块和 uNPE 初始化模块.

(1) kLibNPE

kLibNPE 是 NPDK 的内核编程环境与用户编程的支持环境,在内核支持基于标准以太网设备驱动开发与多核报文处理模块编程.缓冲区初始化主要负责申请和 DMA 映射所有的报文缓冲区.kNPE 初始化模块主要负责硬件的复位操作,SDB 缓冲地址信息下载和 DMA 通道等参数的配置.

a) 内核态轮询驱动

轮询驱动是 kLibNPE 的核心驱动,主要由多个内核态轮询线程组成,每个线程独立在对应的硬件缓冲区队列上轮询报文状态,判断报文是否有效.如果报文有效,则将报文提交上层以太网接口或者用户定义的内核转发模块处理.同时,驱动提供以寄存器写的方式向硬件发送报文.内核轮询驱动提供两路数据收发通路,支持标准以太网驱动或支持多核报文处理模块.

b) 以太网接口驱动实现

kLibNPE 根据设备物理端口或虚拟化端口数量,向系统申请并注册标准以太网设备.每一个以太网设备都与轮询驱动的报文队列绑定.轮询驱动接收的报文,先通过报文格式转换模块,将 SDB 格式转换为标准的 SKB 格式,然后再由以太网驱动处理 SKB 的两层数据,最后调用 netif_receive_skb 函数将报文送入协议栈处理.协议栈处理完成的 SKB 报文同样通过报文格式转换.将 SKB 格式转换为 SDB 格式,最后调用轮询驱动的报文发送接口函数进行发送.

SDB 报文格式与 SKB 格式的转换是一种无数据拷贝的转换方式.首先,从 SDB 转换到 SKB,只需要给 SDB 在内核申请一个 SKB 的数据控制块,重置控制块中的相关指针即可.同理,将 SKB 转换为 SDB,只需要释放 SKB

控制块,将数据块内容以 SDB 形式发送即可.由操作系统内核产生的 SKB,可以通过数据拷贝的方式实现到 SDB 的转换.

c) 多核异构报文转发实现

kLibNPE 支持用户在内核编写报文快速处理转发模块,内核模块通过调用 kLibNPE 的报文转发注册函数,将自己的报文处理接口函数指针注册到转发管理模块,管理模块再将此报文接收函数的指针动态更新到轮询驱动,由轮询驱动回调用户函数实现快速将驱动报文回送给用户在内核模块函数处理.

d) 页表绑定模块

此模块的主要功能是支持实现用户态下对所有报文缓冲区的映射功能.用户态的应用开发支持同时多进程的方式,在对缓冲区进行映射时,所有进程必须形成一致的缓冲区映射关系,此模块主要就是保证所有用户态进程的缓冲区映射的一致性,可以共享式访问硬件所有缓冲区报文.

(2) uLibNPE

uLibNPE 是 NPDK 的用户态编程环境,用户态编程还需要部分 kLibNPE 的功能,如缓冲区的页表绑定,实现用户态地址的映射.uNPE 初始化模块也与内核对应功能相似,实现在用户态下初始化硬件设备.用户态轮询驱动与内核态相似,只是轮询的报文队列地址为用户态地址,发送则是在用户态直接访问硬件寄存器方式发送.uLibNPE 在用户态下支持独占式的报文处理多线程编程和共享式的报文处理多进程编程方式.

a) 独占式多线程编程

独占式多线程编程是指由一个进程独占使用硬件设备.由此进程完成整个硬件的初始化,并启动多个报文处理线程,轮询每个队列上的报文进行处理.独占式多线程编程适合整体项目集中编程使用.

b) 共享式多进程编程

共享式多进程编程是指由多个进程共享使用硬件设备.每个进程负责处理硬件不同队列上的报文,各进程之间虽然共享所有硬件报文缓冲区,但只处理自己的数据,互不影响.共享式多进程必须启动一个守护进程,专门用来初始化硬件设备和负责处理非正常退出进程的报文队列回收工作.共享式多进程共享硬件缓冲的关键是由内核提供报文缓冲区的一致性映射关系,确保所有进程都实现对报文缓冲区的相同映射.共享式多进程适合多任务,多应用的分布式开发.

2.2 开发接口

NPDK 内核态环境支持用户自定义的转发模块 APP 和基于标准以太网的内核态网络转发应用.在用户态环境支持标准 SOCKET 的网络应用开发与 uLibNPE 提供的零中断与轮询、零拷贝和旁路内核的用户态快速报文 I/O 开发.开发接口函数见表 1.

Table 1 Programming interface functions

表 1 开发接口函数

Name	Function
npe_lookup_register	Registering forwarding function in kLibNPE
base_npe_lookup	Prototype definition of forwarding function interface
create_net_device	Creating NPE device in user space
read_pkt_from_thread	Acquring packets form a thread
send_pkt	Sending a packet

内核态开发接口还支持所有与标准以太网设备相关的所有内核开发工作,如标准的网桥转发和路由转发等.用户态开发接口还包括标准的 SOCKET 应用、libpcap 和 libnet 等应用开发.

3 NPDK 的应用开发示例

根据 NPDK 原理,使用 NPDK 环境提供了的用户态库模块,实现一个用户态数据 I/O 裸转发性能测试系统——PoF.PoF 的系统主要包括:NPE 加速引擎、NPDK 环境和 PoF 系统.

PoF 实现了一个数据 I/O 裸转发性能测试系统,可以测量用户态模式数据 I/O 的最大性能.业务能力测量模

块可以测量出在有效完成数据 I/O 的前提下,允许用户业务进行 CPU 计算与内存访问的次数.

(1) 初始化 NPDK

NPDK 环境将所有软硬件的初始化函数全部都封装在环境库内部,用户只需调用初始化函数 *create_net_device*(*dma_cnt*,*disp_mode*)即可,*dma_cnt* 表示硬件启动多少个 DMA 通道,与软件处理线程数对应,每个 DMA 通道对应一个处理线程,绑定在一个指定的 CPU 核上运行;*disp_mode* 指定报文分派模式,可以是循环分派,也可以是端口绑定分派.NPDK 的其他初始化工作都在此函数内部完成.

(2) 创建业务处理线程

根据用户要求,在不超过硬件最大支持 DMA 通道数情况下,用户通过重复调用 *pthread_create*(*&p_t*, *&attr*,*start_npe_thread*,*tp*)函数,选择自由创建多个处理线程,创建线程数要求与前面的 *dma_cnt* 数相同,*start_npe_thread* 即为业务处理线程.线程创建后由用户显示指定绑定到哪个 CPU 核上运行,线程的创建与 CPU 亲和设置都使用标准的 libpthread 库函数.

(3) I/O 裸转发

处理线程启动完成后,可以调用 NPDK 提供的接收报文函数对此线程上的报文队列进行轮询 *read_pkt_from_thread*(*threadid*).参数 *threadid* 说明是从哪个线程上读,此线程编号与硬件 DMA 通道号一致,每个处理线程都对应从一个 DMA 通道的队列上读取报文.

PoF 系统主要用来测量数据 I/O 转发性能,并不对报文进行任何操作.处理流程非常简单,线程轮询到报文后,立即调用报文发送函数进行发送,完成一个报文的转发操作.发送函数 *send_pkt*(**pkt*,*outport*,*pkt_len*),*pkt* 表示报文指针,*outport* 表示输出端口号,*pkt_len* 表示发送长度.具体算法如算法 1 所示.

算法 1. Raw I/O forwarding.

```
Input: thread_id, type, num.
Output: pkt.
while(1){
  pkt=read_pkt_from_thread(thread_id);
  if (pkt){
    switch(type){
    case IO:
      test_IO();
      break;
    case MEM:
      test_mem(num);
      break;
    case MD5:
      test_md5(num);
      break;
    }
    send_pkt(pkt,outport,pkt_len);
  }
}
```

(4) 业务能力测量

对报文的所有业务操作可以总结为两类:CPU 计算和访存,其中 CPU 计算使用标准的 MD5 计算为参考标准,访存以对非报文体内存的非 CACHE 访问为标准.在 I/O 转发流程中插入一个业务能力测量模块,可以通过转发性能的变化可以测量出在不同硬件与系统环境下,用户使用 NPDK 进行数据 I/O 在不丢包情况下,允许用户最多可以执行的访存操作次数与 CPU 计算次数.以此可以衡量用户业务代码在不同的硬件平台与操作系统环

境下是否满足业务性能要求,指导用户选择或改进系统平台的某些专项性能.比如,CPU 计算性能不够可以提升 CPU 频率,访存速率不够可以提升内存频率.相反,当业务处理性能足够时,可以通过降频实现节能.

4 实验分析

使用 PoF 系统对不同长度报文进行转发性能测试,同时在相同参数下对 DPDK 的转发性能进行相同测试.

4.1 测试环境

本系统的测试环境主要包括 4 个部分:测试仪、服务器、支持 DPDK 的万兆网卡和 NPE 的万光网卡.两块万兆网卡分别插在服务器的 PCIE 槽中,加载对应的设备驱动和测试程序.测试仪负责进行数据测试与结果统计.各设备详细配置见表 2.

Table 2 Configuration parameters of test equipment

表 2 测试设备配置参数

	Tester	Server	DPDK	NPDK
Device	Ixia	IBM	Intel 82599	NPE
CPU	2.0G	2.8G/40core	无	无
Memory	2GB	64GB	无	无
Network	2×10G	–	2×10G	4×10G

4.2 实验与数据

所有实验数据中的参考线表示测试仪的发送数据.DPDK 使用标准的 1.6.0 版本代码,重新编写了一个基于端口转发的测试程序,其转发处理流程部分与 PoF 系统完全相同,只是报文接收与发送使用 DPDK 对应接口函数.

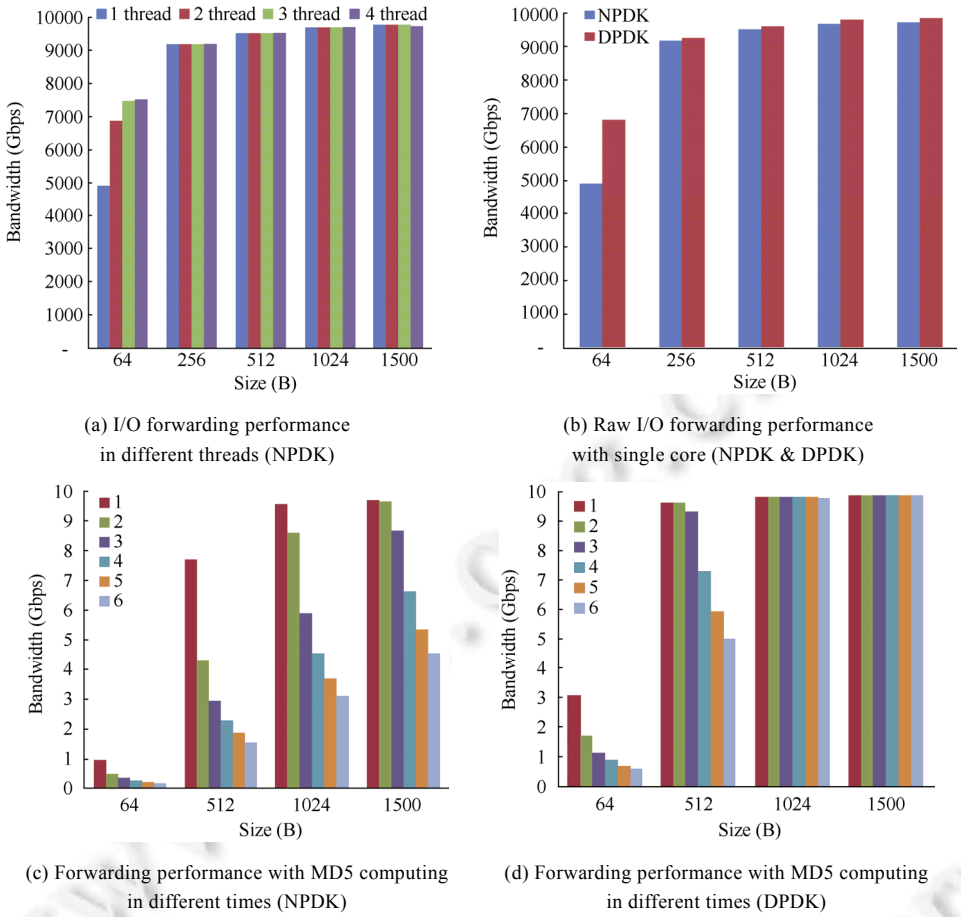
通过使用 PoF 系统对 NPDK 进行不同长度报文的 I/O 转发能力进行测试,NPDK 运行不同转发线程的报文 I/O 转发性能,如图 3(a)所示.测试使用报文循环分派方式,利用多核并行转发,性能有明显的提升.超过 64 字节长度报文仅需 1 个转发线程就可以达到最大转发性能,转发速率接近测试仪发送速率.64 字节长度报文在单线程转发时性能仅为 4.9Gbps,4 线程基本可达到性能转发最大值 7.49Gbps.

如图 3(b)所示,DPDK 与 NPDK 在不同字节的单核(线程)转发能力基本接近,但在 64 字节报文转发时,DPDK 性能要高出 1.89Gbps.其他各字节转发性能 DPDK 平均高出 0.119Gbps.

NPDK 与 DPDK 中插入业务能力测量模块后,可以测量在不同报文长度转发过程中,插入不同次数的 MD5 计算或非 CACHE 访存次数对转发性能的影响.

如图 3(c)所示,为 NPDK 在报文转发中加入不同次数的 MD5 计算后的性能转发结果,其中参考项为 0 次 MD5 计算.图 3(d)是 DPDK 的测试结果.从两图比较可看出,NPDK 在加入 MD5 计算后性能损失较大.而 DPDK 只在小于 512 字节报文或 MD5 计算次数超过 6 次后才有较大的性能变化.NPDK 的 CPU 性能消耗较大应该与其驱动的无中断轮询方式有关,在无报文处理时,轮询报文会使 CPU 的利用率很高,只有当有报文处理时才真正使用 CPU 进行报文处理,其他时间全在对报文进行轮询操作.

在报文转发中加入非 CACHE 访存后的性能变化,当加入次数小于 500 时,各性能变化不大.但是,在 1 500 字节报文转发时,随着非 CACHE 访存次数的增加,DPDK 受影响,变化比 NPDK 要大.



(a) I/O forwarding performance in different threads (NPDK)

(b) Raw I/O forwarding performance with single core (NPDK & DPDK)

(c) Forwarding performance with MD5 computing in different times (NPDK)

(d) Forwarding performance with MD5 computing in different times (DPDK)

Fig.3 Experimental results (NPDK & DPDK)

图 3 DPDK 与 NPDK 实验结果

4.3 测试结果分析

通过 PoF 系统对 NPDK 的性能测试发现,在相同的硬件平台与操作系统环境下,NPDK 的性能在 64 字节大小报文时比 DPDK 要高.而在其他字节报文,DPDK 基本上可以接近线速,但 NPDK 的转发速率要比 DPDK 略小.其主要原因是,在 NPDK 在报文转发出来后,有部分报文 CRC 校验时有错,影响了 NPDK 转发速率.从两者的报文业务能力测量可知,在 DPDK 报文转发中可以加入更多的 CPU 计算功能,而在 NPDK 中可以加入更多的访存操作.从对网络报文的处理分析,要对报文进行更深入的解析或基于应用层内容的分析,主要偏向对报文的访存,NPDK 更适用于此类型操作.如网络探针、防火墙、IDS 和 DPI 等应用.

5 结束语

通用多核处理器的研究越来越多,也越来越深入,通过对数据 I/O 路径上辅助功能的优化可以在很大程度上提高数据 I/O 性能.NPDK 与 DPDK 都可以取得较好的效果.NPDK 的主要优势是缓冲区的硬件管理下载,在小字节报文转发速率上比 DPDK 更有优势.而且 NPDK 在缓冲区的大页使用上还未进行优化,下一步的工作主要朝着这一方向努力.另外,为了减少 CPU 轮询的浪费,需要设计一种新的方式轮询报文.最后,为了兼容更多的系统应用,NPDK 的缓冲区管理需要再进一步优化,进一步提升 NPDK 的系统处理与转发性能.

References:

[1] Kohler E, Morris R, Chen BJ, Jannotti J, Kaashoek MF. The click modular router. In: Proc. of the ACM Symp. on Operating Systems Principles (SOSP). 1999. 217−231.

[2] Koponen T, Amidon K, Balland P, Casado M, Chanda A, Fulton B, Ganichev I, Gross J, Gude N, Ingram P, Jackson E, Lambeth A, Lenglet R, Li SH, Padmanabhan A, Pettit J, Pfaff B, Ramanathan R, Shenker S, Shieh A, Stribling J, Thakkar P, Wendlandt D, Yip A, Zhang RH. Network virtualization in multi-tenant datacenters. In: Proc. of the 11th USENIX Symp. on Networked Systems Design and Implementation (NSDI). 2014. 203−214.

[3] Rizzo L. Netmap: A novel framework for fast packet I/O. In: Proc. of the USENIX Annual Technical Conf. 2012. 101−112.

[4] Han SJ, Jang K, Park KS, Moon S. Packet shader: A GPU-accelerated software router. In: Proc. of the ACM SIGCOMM. 2010. 195−206

[5] Liao G, Znu X, Bnuyan L. A new server I/O architecture for high speed networks. In: Proc. of IEEE the 17th Int'l Symp. on High Performance Computer Architecture (HPCA). 2011. 255−265.

[6] Intel. Intel data plane development kit (Intel DPDK). http://www.dpdk.org

[7] Tang L, Sun ZG, Li T. Self-Described buffer management method for software high-speed packet. In: Proc. of the High Performance Computing. 2014 (in Chinese with English abstract).

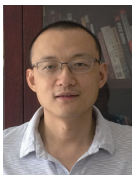
附中文参考文献:

[7] 唐路,孙志刚,李韬.面向高速软件分组转发的自描述缓冲区管理技术.见:2014 全国高性能计算学术年会论文集.2014.

唐路(1988－),男,湖南衡阳人,博士生,主要研究领域为高性能路由器,网络处理器.

李韬(1983－),男,博士,助理研究员,CCF 专业会员,主要研究领域为计算机网络,网络处理器.

徐东来(1982－),男,硕士,主要研究领域为新型互联网体系结构,高性能路由与交换技术.



孙志刚(1973－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为计算机网络体系结构,高性能路由器.

吕高锋(1980－),男,博士,助理研究员,主要研究领域为新型互联网体系结构,高性能路由与交换技术.