高技术通讯 2013 年 第23 卷 第4 期:360 ~ 367

doi:10.3772/j.issn.1002-0470.2013.04.005

面向云计算的多核处理器存储和网络子系统优化设计①

苏 文②[*,**,***,****] 王焕东[*,**,****] 台运方[*,**,***,****] 王 靖[*,**,***,****]

([*]中国科学院计算机系统结构重点实验室 北京 100190)

([**]中国科学院计算技术研究所 北京 100190)

([***]中国科学院研究生院 北京 100049)

([****]北京龙芯中科技术服务中心有限公司 北京 100190)

摘 要 针对传统多核处理器设计缺乏对虚拟机和典型云服务的支持的问题,分析了云计算数据中心和虚拟机的基本架构和特点,指出数据传输、网络性能、I/O 虚拟化是一直未被关注的影响系统性能的关键因素,进而提出了一种改进的多核处理器设计方案。该方案通过采用片上内存拷贝引擎、改进直接内存访问(DMA)设计、改进直接缓存访问(DCA)设计和采用快速地址转换和远程内存访问(RDMA)技术,来较大地提高存储系统、网络、I/O 的性能和系统的并行性。实验表明,该方案实现的单核 800MHz 处理器千兆以太网络 TCP 传输带宽较传统方案提高 48.2% 并达到峰值 800Mbps,内存拷贝操作加速比达到 14 倍以上,快速傅立叶变换(FFT)和矩阵乘法加速比达到 2 倍以上,同时系统高速缓存效率显著提高。

关键词 云计算,多核处理器,网络优化,虚拟机,计算机体系结构

0 引 言

随着云计算技术的快速发展,软硬件系统设计成为研究的热点。Amazon、Google 等公司都建立了云计算数据中心并提供相应的云服务[1,2]。云服务具有虚拟化、高并行、访存/计算密集等特点[3],这对系统的网络、数据传输、并行处理性能提出了更高的要求。与传统机群系统不同,云计算数据中心的每个物理终端上都装有虚拟机系统以支持多用户服务。由于虚拟机实现了对硬件系统资源的虚拟化,这在一定程度上改变了底层程序的行为特点。研究表明虚拟机会降低原有操作系统和硬件的性能和效率,这种现象在网络子系统上更加严重[4,5]。传统多核处理器由于没有考虑虚拟机和云服务的特点,所以不能够完全满足当前云计算数据中心的需求[3]。尽管当前设计者在芯片内集成了更多的处理器核来提供更高的并行能力,但由于在存储体系和芯片功能上没有突破性的创新,系统整体效率仍有待提升。针对这一问题,研究人员从不同角度和方法提出了优化结构和方案:Regnier 等提出将 TCP 专用处理硬件集成在多核处理器中的某一核上来提高网络性能[6];Zhao 等提出利用专用的拷贝引擎来完成系统内数据搬运[7];AMD 公司对其 12 核 Opteron 处理器在系统高速缓存(cache)和内存(memory)系统的设计上进行了多项支持高吞吐量数据中心的创新设计[8]。尽管这些方法在系统性能提高上取得了一定效果,但其与云计算系统的结合程度仍稍显单薄,没有完全发挥出多核处理器的计算能力。本文通过分析云计算数据中心基本架构和虚拟机典型特点,指出,数据传输、网络处理、虚拟化和并行性造成了云环境多核处理器设计的瓶颈,进而,基于龙芯 3 号处理器,提出了一种改进的多核处理器设计方案。实验表明,用该方案设计的多核处理器系统的性能有显著提高。

① 国家"核高基"科技重大专项(2010ZX01036-001-002, 2009ZX01028-002-003, 2009ZX01029-001-003)和国家自然科学基金(61100163, 61133004)资助项目。

② 男,1986 年生,博士生;研究方向:网络与分布式系统,计算机系统结构,操作系统;联系人,E-mail: suwen@ict.ac.cn

(收稿日期:2012-05-10)

1　云计算数据中心结构及其典型应用特点

1.1　数据中心基本架构

典型的云计算数据中心架构如图1所示。廉价刀片服务器或PC机组成了数据中心的基本计算/存储单元(端点机器),多个基本单元通过以太网(Ethernet)与交换机相连。基本单元的功能并不完全相同,如基于谷歌文件系统(Google File System, GFS)的数据中心要求在一个GFS集群中有一个主服务器和多个块服务器[9]。基于多级路由的互联网络将所有计算/存储资源相连接,并最终通过虚拟机技术形成一个面对用户的统一服务接口。

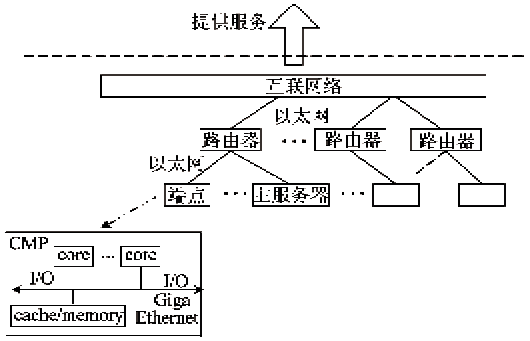

图1　云计算数据中心基本架构

由于云计算数据中心采用低成本和低功耗设计,因此端点机器之间一般采用千兆以太网连接,而路由器之间的互连采用万兆以太网。为了保证系统可靠性并达到负载均衡,端点间的进程迁移、虚拟机迁移和数据传输频繁进行[3],同时大量的数据冗余被用来保证存储系统的正确性。这些特点决定了应用于数据中心的处理器芯片应具有较高的网络数据传输能力。

1.2　虚拟机与I/O虚拟化

云计算的一个显著特点是通过虚拟化技术在廉价机群的基础上实现无限多的虚拟硬件资源。通过操作系统、虚拟机和中间件技术,云服务提供商实现多个用户共享一台物理机器上的计算、存储和I/O等资源。因此,虚拟机的性能在很大程度上决定了整个云计算系统及数据中心的性能。这里我们通过研究典型虚拟机对底层硬件系统的影响,寻求指导本文提出的多核处理器存储和网络子系统设计优化方案。

Xen是由英国剑桥大学开发的基于x86平台的开源虚拟机系统,其广泛应用于Amazon和Gogird等云服务平台[1,10]。研究表明,当前Xen虚拟机的最大性能瓶颈在于I/O设备的虚拟化,基于虚拟机的网卡传输带宽较原始操作系统网络带宽降低30%~60%[4,5]。图2给出了Xen虚拟机I/O虚拟化的基本架构和原理。在Xen中所有与物理I/O设备相关的驱动和协议被集中在驱动域(driver domain)当中,驱动域是Xen中的一个地位特殊的虚拟机,其他虚拟机均通过它来完成与物理I/O设备的交互。而完全虚拟化的一般虚拟机则被称为客户域(guest domain),该域根据自身操作系统类型的不同,集成相应的虚拟I/O设备接口。驱动域与客户域之间的数据传输通过专门的I/O通道与环形描述符所完成,多个"客户域"的I/O请求根据特定的调度算法送往驱动域。上述Xen中I/O虚拟化技术的最核心思想是将与真实设备相关的部分独立出来并对其进行特殊的管理,从而保证多个虚拟机能够安全、正确地占有"独立"的I/O设备,避免出现由于I/O设备直接内存访问(direct memory access, DMA)操作等造成的越权访问、修改等问题[11]。

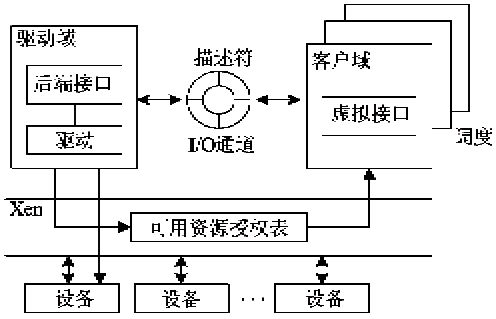

图2　Xen虚拟机I/O架构

可以看出这一过程有以下特点:(1)进程间通信频繁。多个客户域会并发地与驱动域进行数据传输,并且系统在不同域间不断切换调度,多个进程间需要频繁相互通信来实时更新当前I/O的工作状态;(2)I/O相关操作开销大。由于高速I/O设备中DMA传输的原子性和可控性较差,因此对其虚拟化操作的性能开销很大,这一问题已经成为虚拟机I/O优化的关键问题。

基于以上分析,本文方案通过对进程间通信的主要操作"数据拷贝"和I/O设备DMA传输进行优化,并改进由于虚拟机调度产生的频繁地址转换问题,进而为系虚拟机本身提供更大的优化空间。此外,改进的集成直接内存访问(integrated DMA, IDMA)设计支持多媒体程序的三种典型访存模式[12],

从而提高系统处理媒体数据流的效率。

2 面向数据中心的存储系统设计

本节中我们首先介绍了面向云计算的改进多核处理器系统互联架构，之后描述了改进数据通路、IDMA 设计、网络优化和虚拟机支持的详细方案及其技术实现。

2.1 系统互连和数据通路

本研究是基于本课题组之前的工作——Godson3 多核处理器架构[13,14]展开的，重点对其存储和网络子系统结构按本文提出的方案进行了优化设计，得到的改进的处理器系统互连结构如图 3 所示。

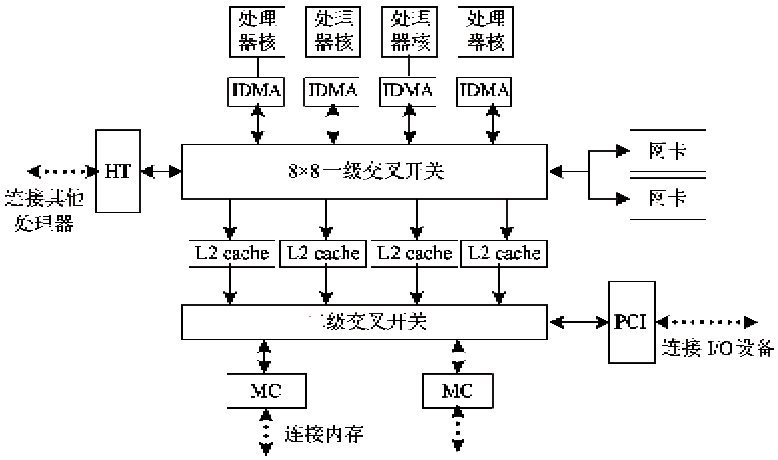

图 3 基于龙芯 3 号的改进多核处理器结构

四个同构的改进 GS464 处理器核通过 DMA 引擎与一级交叉开关相连，其中 DMA 负责完成二级 cache/内存到核内存储空间及本地存储间的大规模数据传输，同时额外的 bypass 通路支持原有处理器与片上网络的直接相连。采用共享 cache 协议组织的 4 块四写四读 cache 块并行组成总大小为 4MB 的二级 cache，其中每一个 cache 块采用四路组相连结构。

与传统方案不同，我们将两个千兆以太网卡集成在处理器内部并与一级交叉开光相连，其目的是有效减少网络数据经过 I/O 控制器和桥片的延迟[15]。同时我们在一级交叉开关内实现了 DCA 相关功能支持，网络数据将会通过网卡内 DMA 引擎被直接送往二级 cache，从而实现处理器对网络数据的快速访问。

此外 1 个 HT(HyperTransport)控制器被用来分别完成与传统 I/O 设备的连接和多片处理器间的板级互连。2 个 DDR2/3 控制器通过二级交叉开关与系统相连。同时在一级交叉开关上预留有 2 个 AXI 接口为之后的系统扩展提供便利(如集成专用的虚拟机加速、TCP 处理等模块)。

同时我们还在处理器核内集成了 128 个 256 位寄存器作为核内存储空间，并对浮点运算部件进行了 256 位向量扩展，大幅度提升了处理器核的计算能力。并通过 IDMA 设计改善系统原有访存模型，进而大大提高了向量部件数据传输的有效数据带宽。同时 IDMA 通道支持多数据流、多间隔读写等，最大程度上利用了多媒体流程序的访存特点。

2.2 IDMA 设计与系统并行

为了实现高效率、高带宽的数据传输，我们设计了集成在处理器核内的专用 IDMA 引擎，其负责实现支持缓存访问的不同物理地址间数据拷贝，以及完成核外存储到核内存储空间的批量数据传输。此外考虑到传统 DMA 部件虚拟化效率较低，我们在 IDMA 上实现了多通道并行、可控传输过程等特性，为包括网络在内的 I/O 设备提供虚拟化支持。具体来说其具有以下设计特点：

(1)支持动态配置的多通道设计：每一个 IDMA 控制器包含 3 条读通道和 1 条写通道。每条通道都有自己的配置寄存器，可以实时根据任务的需求传输任意大小的数据流。IDMA 控制器可以在不同传输任务之间来回切换，并且能够保存每个数据流的工作现场，使得各个数据流并发传输而不会产生额外的硬件开销。逻辑上支持 3 个输入流和 1 个输出流并发执行，物理上每个时钟周期支持一个输入流和一个输出流并行执行。因此，IDMA 在虚拟机调

度等需要频繁切换程序执行上下文的环境中能够保持较高的效率和性能。其结构如图 4 所示。

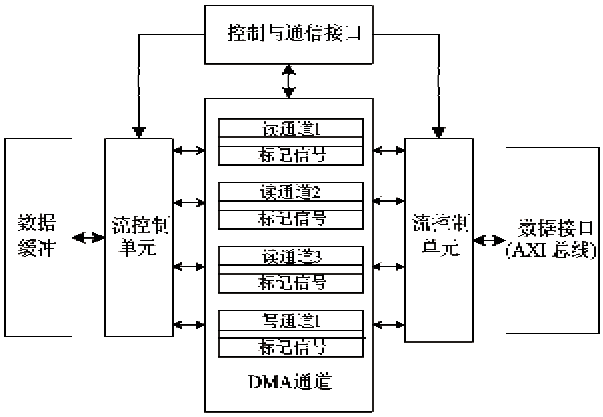

图 4　IDMA 控制器结构

(2)可控的细粒度并行设计:传统 DMA 控制器的数据传输任务一般具有原子性,即一旦开始工作就无法停止,并且其内部传输状态对外界是透明的,这一特点对于系统异常处理及虚拟化操作非常不利。为了解决该问题并提供任务级并行性,在 IDMA 控制器内部实现了一组可配置的标记信号(tags)作为与处理器的交互接口。处理器通过查询标记信号可以获得每条通道传输任务的实时状态和工作进度,并通过对其配置以达到启动和暂停各条通道的目的。相应标记信号位为 1 代表处理器可以使用相应数据,反之则由 IDMA 使用。通过不断改变这组标记信号位的状态,可以实现细粒度的数据传输与计算并行。分别利用传统 load/store 指令和 IDMA 实现高速并行内存拷贝的例子,如图 5 所示。

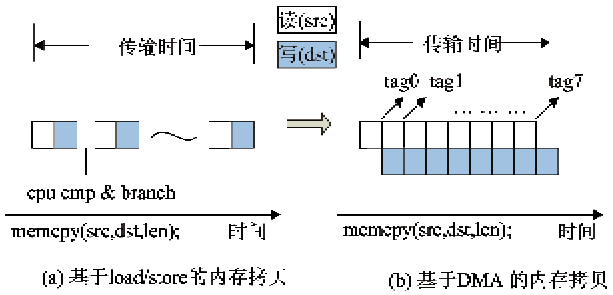

图 5　利用 IDMA 实现程序并行 memcpy 的例子

系统首先配置 IDMA 读通道和写通道的源地址、目的地址和数据长度并启动传输。读通道从源地址读取 tag0 代表的数据量,然后向 CPU 发出中断请求。CPU 如果判断这部分数据已读取完毕则开始写通道的数据传输任务,将已读取的数据写回内存的目的地址。同时,读通道继续读取 tag1 代表的数据量,从而实现数据读写的并行。通过这种流水化的处理,写通道的启动和读写通道之间的转换、暂停与重启等开销都被隐藏起来并减少了处理器的开销,同时也获得了很高的带宽利用率,改善程序局部性。

2.3　网络系统优化

网络性能对数据中心极为关键,因此本文方案对多核处理器网络系统在硬件体系结构和软件行为上进行了专门的优化设计。同时针对 I/O 虚拟化严重降低系统网络性能这一问题我们也做了相应考虑,但虚拟机自身 I/O 优化技术不在本文讨论之内。

为了有效降低网络数据在 I/O 总线上的传输延迟,我们采用了改进的直接缓存访问(direct cache access, DCA)技术。DCA 是由 INTEL 提出的一种通过将 I/O 数据直接送入缓存当中从而降低 CPU 数据访问延迟的优化技术,但研究表明简单的 DCA 会在系统负载压力较大的情况下产生大量缓存缺失(cache miss),同时传统多核处理器结构缺陷降低了 DCA 的效果[16,17]。我们认为这些问题主要由以下原因导致:(1)网卡 DCA 时,短时间内大量的 I/O 数据被连续存入高速缓存会导致原有高速缓存中的数据被彻底换出,在系统高负载的情况下这会极大降低原有程序性能,同时程序原有的访存请求会带来数据"二次替换",影响之后的网络处理过程。这种数据"交替换出"过程会显著降低系统高速缓存效率。(2)由于 DCA 的数据仍要通过 I/O 总线传输至高速缓存中,这将不可避免地花费上千时钟周期来完成 I/O 控制器上的信号转换[18],因此 DCA 的效果被显著降低。

基于以上分析,我们提出了利用创新体系结构、传输策略和处理器 cache 锁技术来解决该问题。图 6给出了传统网络处理、DCA 传输和本文方案提出的改进 DCA 传输的访存路径。可以看到,本文方案由于将网卡从 I/O 总线下移至了靠近处理器核的一级交叉开关(图 3),使得数据传输的线路大大缩短,并避免了耗时的 I/O 控制器传输协议转换。同时我们利用了现代处理器中的高级功能 cache 锁[19]来实现对网卡 DMA 描述符、skb _ buf 和中断处理函数[20]等时间开销较大的关键代码和数据的快速访问。cache 锁改变了系统高速缓存的行为,使得被锁住的数据常驻于高速缓存中,这使得操作系统频繁中断和虚拟机切换造成的高速缓存数据替换开销有效降低,并克服网络处理程序时间局部性较差的缺陷。

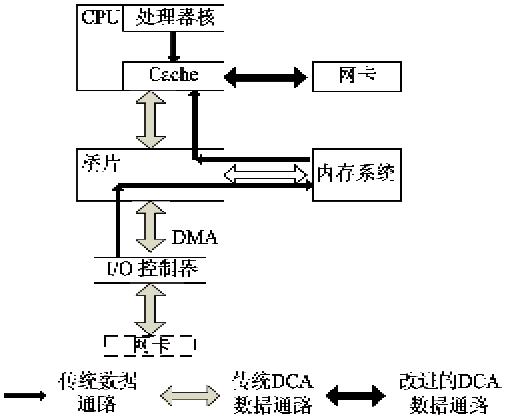

图 6 文中方案系统网络处理数据通路

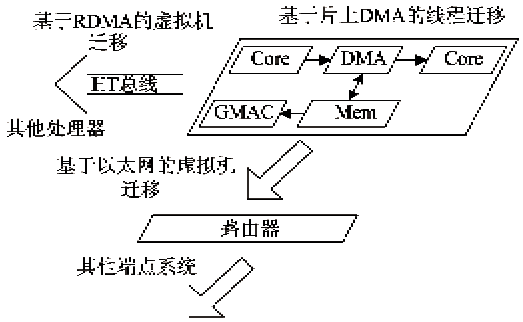

图 7 基于本文改进的系统的进程迁移与虚拟机迁移机制

需要注意的是,将不会被访问到的数据锁住会降低系统性能,因此被锁的数据大小应受到限制,并根据系统负载情况在网络传输压力较大时启用 cache 锁。

我们利用上一小节给出的并行内存拷贝方法,优化了由系统套接字缓冲区至用户程序缓冲区的数据拷贝过程,降低了数据包处理对处理器造成的负担。

2.4 虚拟机相关硬件支持

在本小节中我们将讨论硬件系统对虚拟机的支持。研究表明,虚拟机及操作系统的二次内存管理和虚拟机调度会使程序在运行时频繁产生虚实地址转换及相应缺页处理等操作,极大地降低了系统性能。为了解决该问题,我们特意在处理器核中集成了 64 项的 CAM 查找表用来记录缺页最为频繁的地址对,并对 CAM 的访问和修改进行了特殊指令编码实现。这样通过向虚拟机提供专用的地址转换函数可以极大地减少地址转换开销。该技术已经成功应用于基于本文方案实现的相关系统的 JAVA 虚拟机性能优化当中,并取得很好的效果[21]。

需要考虑的另一个重要情形是基于跨指令平台的虚拟机性能,如支持 MIPS 和 x86 体系结构转换的 QEMU 系统[22]。这时 CAM 可以为特权指令或者翻译难度较高指令提供快速查找支持,这对实现高性能的二进制翻译和跨平台虚拟机性能影响巨大[23]。

另外本文重点考虑了弹性云计算中数据中心虚拟机动态迁移频繁等特点,在实现多片互联的 HT 控制器中实现了远程内存访问(remote DMA, RDMA)功能。图 7 给出了基于本文方案系统的多核间进程迁移、同一端点机器内和不同端点系统间虚拟机迁移的过程,前者通过片内 IDMA 控制器完成,而后者通过 HT 总线上的 RDMA 控制器或高速以太网络完成。详细的多片处理器互联及数据传输方案在之前工作[24]中有具体描述,这里不再重复。

3 实验结果与分析

3.1 实验环境和实验三方法

我们采用实现了本文方案改进 Godson3 处理器单核芯片作为评测环境。尽管该芯片是单核,但其互连结构和存储系统都是基于多核芯片设计的,因此通过实验结果可以简单推算出多核环境下系统的性能。由于当前云计算系统不存在统一的标准化测试程序,我们重点测试了系统网络 TCP 传输带宽、数据拷贝和典型科学计算程序的性能。实验平台具体参数如表 1 所示。Synopsys 公司的千兆以太网控制器(GMAC)被集成在处理器内作为内集成网卡(INIC)[25],一个四核 2.66GHz Dell 服务器作为网络测试数据发送方与待测系统通过千兆以太网直接相连。整个实验基于 Linux 2.6.21 操作系统。

表 1 实验平台具体参数

CPU & Cache	
CPU	800MHz
核内存储	2kB
L1 Cache	64K + 64K, four-way associated
L2 Cache	512kB, four-way associated
Memory & I/O	
内存	250MHz, DDR2/3, 1GB
内部总线	800MHz ,128 位 AXI
外部总线	200MHz, 16 位 HT1.0
网卡	Synopsys GMAC

实验中我们采用了对比测试的方法,即在同一实验平台上分别基于传统方式和本文方案系统运行测试程序并比较性能。具体说来,针对网络性能我们选择了 netperf 测试集的 TCP 带宽测试[26](云环境网络通信主要基于 TCP/IP 协议),分别记录通过

PCI 总线外接千兆网卡和芯片内集成 GMAC 的网络带宽峰值;对于数据拷贝和典型科学计算程序,我们首先测试了基于处理器传统 load/store 指令访存的程序性能,之后对测试程序访存部分进行修改,利用 IDMA 完成大规模数据传输操作,测试相关性能并计算加速比(加速比 = T_1/T_2,其中 T_1、T_2 分别代表传统程序和改进程序的运行时间)。

3.2　结果分析

(1)网络性能

从图 8 可以看出,基于本文提出的优化方案系统带宽平均提升 48.2%,在包大小为 1024 字节时达到带宽峰值 794MB/s。考虑到配合其他优化措施如中断聚集技术(interrupts coalescing)等,则有望接近千兆以太网络的传输极限。另外从图 8 中看出,随着数据包大小的增加,带宽并没有随之增加,这与的结论有所不同[27]。我们分析认为,这是由于一方面优化方案大幅减少了中断、缓冲区维护等每包传输开销(per-packet cost),这使得系统面对频繁的小数据包处理仍能够保持较高的效率;而另一方面,因为本文方案中并未包含对包内数据传输的优化,因此随着包数据增大,其他部分优化所带来的收益占比在降低,这在一定程度上抵消了前人研究所指出的批量数据传输带来的效率提升[27]。

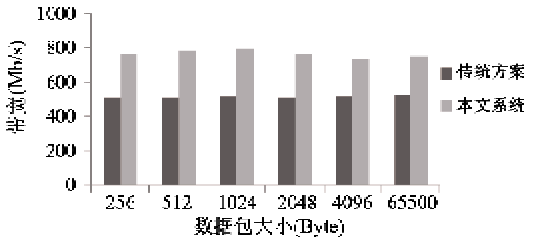

图 8　TCP 传输带宽

(2)数据传输与典型应用测试

我们根据图 6 所示的算法实现了数据拷贝操作并对其性能进行评估。数据拷贝操作广泛存在于操作系统、网络处理和虚拟机等云服务的关键路径上,其对于系统整体性能有很大影响。与此同时,我们还评估了矩阵乘法和快速傅利叶变换操作在本文方案系统上的性能,这二者是多媒体、数字信号处理中的核心运算,属于云服务科学计算中的核心操作。

实验采用 Goto _ Blas 数学库中矩阵乘法与标准 1k 和 8k 数据点下的快速傅立叶变换(FFT)运算程序,通过对其进行了汇编程序优化,并利用 IDMA 控制器完成大规模数据传输和程序任务级、数据级并行后与原程序进行性能比较。同时测试了基于核内 IDMA 以及标准 C 语言库的 memcpy 程序传输 64B、256B 和 1KB 数据的性能,运行结果如图 9 所示。

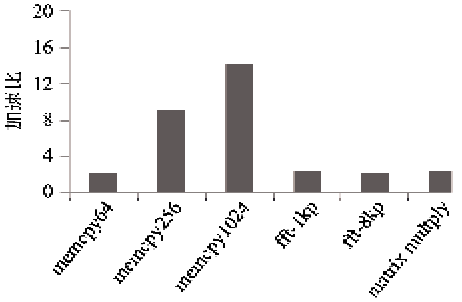

图 9　数据拷贝与典型科学计算程序运行加速比

从测试结果可以看出,基于 IDMA 进行数据传输与传统 load/store 访存方式相比,FFT 等程序整体运行加速比达到 2.2 倍以上,内存拷贝程序在数据量大于 1KB 时,加速比达到 14 倍以上。因为传输机制不同,利用 IDMA 在大规模数据传输方面性能会高出 load/store 方式许多,并且前者在一定程度上实现了程序的任务级并行,突破了串行程序加速的瓶颈,所以程序性能会有较大提升。对于规模较小的数据传输(如 64B 内存拷贝),IDMA 配置开销时间占比较大;而当数据规模大于 1kB 时,该开销基本可以忽略不计,系统传输性能提升明显。此外由于矩阵乘法和 FFT 运算访存模式固定并且任务级并行性较高,考虑到程序的计算复杂性特点和理论访存时间比例,本文方案实现的存储系统对其性能提升较大。

(3)高速缓存性能

多核环境下处理器高速缓存的组织和层次结构对应用程序性能影响很大。基于本文方案实现的 Godson3 系统为两级 cache 结构,其中一级指令和数据 cache 集成在每个处理器核内并被其所私有,而二级 cache 则被四个处理器核共享并由硬件维护一致性。根据这一特点,IDMA 选择后者作为数据交互接口。一方面,共享的 2MB 二级 cache 完全可以在不降低原有性能的基础上满足数据传输时的缓冲区开销;另一方面,其避免了将数据保存在私有的一级 cache 而产生的数据重复备份及频繁的一致性操作,进而在多线程环境下取得较高性能。此外其与 DCA 通过二级 cache 交互会降低数据传输路径的长度,提高传输效率。

由于芯片上很难准确观察到系统内部 cache 行为等细节信息,本文选择通过在用 xtreme Ⅲ 硬件加

速仿真的环境中运行 1k 点下的 FFT 程序来观察存储系统的性能，结果如图 10 所示。其中 xtreme Ⅲ系统是由 Cadence 公司提供的系统仿真及硬件加速系统，具有很高的精确度和仿真速度。

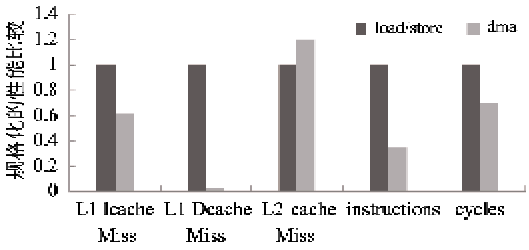

图 10 基于文中系统 1k 点下 fft 程序性能分析

可以看出，基于本文方案系统，程序运行时指令执行条数减少 60% 以上，运行时间减少 50% 以上，一级指令 cache 和一级数据 cache 缺失次数大大降低，二级 cache 缺失次数增加 20% 左右。利用 IDMA 进行数据传输会大大减少程序运行时的访存指令条数，从而降低一级指令 cache 缺失。由于大规模的数据传输通过 IDMA 命中系统二级 cache，使得一级数据 cache 缺失几乎没有，但会使得二级 cache 缺失次数有一定上升。

4 结 论

本文对云计算环境下数据中心架构和虚拟机的典型特点的分析表明，多核处理器数据传输、并行性、网络和 I/O 性能是决定系统性能的关键因素。本文基于龙芯 3 号处理器提出的面向云计算的多核处理器改进设计方案，通过基于核内 IDMA 和 RDMA 机制实现了处理器内部不同存储空间及多个直连处理器间的高效数据传输，而且通过体系结构创新并结合改进 DCA 和 cache 锁等策略实现了网络性能的大幅改进，此外还通过利用 CAM 表实现快速地址转换、可控并行 DMA 设计等对虚拟机性能优化提供支持。该方案有以下创新点：(1)通过设计适合于 I/O 虚拟化的多通道 IDMA 引擎来增强系统访存能力，实现对核内数据的批量读取，并支持快速高效的数据传输，提高系统并行处理能力；(2)利用改进 DCA 策略和 cache 锁技术并结合体系结构优化，有效降低网络数据在 I/O 总线上的数据延迟，并改善网络处理程序的局部性；(3)针对虚拟机相关特点，通过在处理器内集成 IDMA、内容可寻址存储器(content addressable memory, CAM)表和 RDMA 等技术加速相关地址转换和进程/虚拟机迁移等过程。实验表明，基于本方案实现的多核处理器系统千兆以太网络 TCP 传输带宽提高约 48.2%，内存拷贝性能提高 14 倍以上，典型科学计算程序 FFT、矩阵乘法等性能提高 2 倍以上。该系统对应的龙芯 3 号相关处理器已经应用于国内多款云计算系统方案中。

参考文献

[1] Amazon ec2. http://aws.amazon.com/ec2/

[2] Barroso L, Dean J, Hölzle U. Web search for a planet: The Google cluster architecture. *IEEE Micro*, 2003, 23(2): 22-28

[3] Armbrust M, Fox A. Above the Clouds: A berkeley view of cloud computing. technical Report No. UCB/EECS-2009-28, University of California at Berkley, USA, Feb. 10, 2009

[4] Menon A, Cox A, Zwaenepoel W. Optimizing network virtualization in Xen. In: Proceedings of the ACM/USENIX Annual Technical Conference, Boston, USA, 2006

[5] Menon A, Santos J, Turner Y, et al. Diagnosing performance overheads in the Xen virtual machine environment. In: Proceedings of the ACM/USENIX Conference on Virtual Execution Environments, Chicago, USA, 2005

[6] Regnier G, Makineni S, Illikkal R, et al. TCP onloading for data center servers. *IEEE Transactions On Computer*, 2004, 37(11): 48-58

[7] Zhao L, Bhuyan L, Iyer R, et al. Hardware support for accelerating data movement in server platform. *IEEE Transactions On Computer*, 2007, 56(6):740-753

[8] Conway P, Kalyanasundharam N, Donley G, et al. Cache hierarchy and memory subsystem of the AMD Opteron processor. *IEEE Micro*, 2010, 30(2):16-29

[9] Ghemawat S, Gobioff H, Leung S. The Google file system. In: Procedings of the 19th ACM Symposium on Operating Systems Principles, New York: ACM Press, 2003. 29-43

[10] Gogrid. http://www.gogrid.com/

[11] Fraser K, Hand S, Neugebauer R, et al. Safe hardware access with the Xen virtual machine monitor. In: Proceedings of the 1st Workshop on Operating System and Architectural Support for the on demand IT Infrastructure (OSASIS)/Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2004

[12] Lee J, Chanik P, Soonhoi H. Memory Access Pattern Analysis and Stream Cache Design for Multimedia Applications. In: Proceedings of the Asia and South Pacific Design Automation Conference, 2003, 22-27

[13] Hu W W, Wang J, Gao X, et al. Godson-3: a scalable multicore RISC processor with x86 emulation. *IEEE Micro*, 2009, 29(2): 17-29

[14] Gao X, Chen Y J, Wang H D, et al. System architecture

of Godson-3 multi-Core processors. *Journel of computer science and technology*, 2010, 25(2): 181-191

[15] Binkert N, Saidi A, Reinhardt S. Integrated Network Interfaces for High-Bandwidth TCP/IP. In: Proceedings of the 11th Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2006. 315-324

[16] Huggahalli R, Iyer R, Tetrick S. Direct Cache Access for High Bandwidth Network I/O. In: Proceedings of the 32nd annual International Symposium on Computer Architecture, 2005. 50-59

[17] Kumar A, Huggahalli R, Makineni S. Characterization of Direct Cache Access on Multi-core Systems and 10GbE. In: Proceedings of the IEEE 15th International Symposium on High Performance Computer Architecture, 2009. 341-352

[18] Miller D, Watts P, Moore A. Motivating Future Interconnects: A Differential Measurement Analysis of PCI Latency. In: Proceedings of the 5th ACM/IEEE Symposium on Architectures for Networking and Communications Systems, 2009. 94-103

[19] Liang Y, Mitra T. Instruction cache locking using temporal reuse profile. In: Proceedings of the 47th ACM/IEEE Design Automation Conference, 2010, 344-349

[20] Wu W J, Crawford M. Potential performance bottleneck in Linux TCP. *International Journal of Communication Systems*, 2007, 20(11), 1263-1283

[21] Cai S S, Yang Y Q, Lin Q W, et al. JVM virtual method invoking optimization based on CAM table. In: Proceedings of the 6th IEEE International Conference on Networking, Architecture and Storage, 2011. 122-129

[22] Bellard F. QEMU: a Fast and Portable Dynamic Translator. In: Proceedings of the ACM/USENIX Annual Technical Conference, Anaheim, USA, 2005. 41-46

[23] Hu W W, Liu Q, Wang J, et al. Efficient binary translation system with low hardware cost. In: Proceedings of IEEE International Conference on Computer Design, 2009. 305-312

[24] 王焕东, 高翔, 陈云霁等. 龙芯3号互联系统的设计与实现. 计算机研究与发展, 2008, 45(12): 2001-2010

[25] Synopsys GMAC IP. http://www.synopsys.com/dw/dwtb.php? a = ethernet _ mac

[26] Jones R. NetPerf: a network performance benchmark. http://www.netperf.org

[27] Foong A, Huff T, Hum H, et al. TCP Performance Revisited. In: Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software, 2003. 70-79

Optimization of memory and network subsystems in multi-core processors for cloud computing

Su Wen[*,**,***,****], Wang Huandong[*,***,****], Tai Yunfang[*,**,***,****], Wang Jing[*,**,***,****]

([*] Key Laboratory of Computer System and Architecture, Chinese Academy of Sciences, Beijing 100190)

([**] Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

([***] Graduate University of Chinese Academy of Sciences, Beijing 100049)

([****] Loongson Technology Corporation Limited, Beijing 100190)

Abstract

To solve the problem that traditional multi-core processer designs rarely consider to give support to virtual machines and typical cloud services, the paper analyzes the typical architecture and characteristics of data-centers for cloud computing and virtual machines and points out that data transfer, network performance and I/O virtualization are the key but neglected factors affecting system performance, and based on the analysis, proposes an improved multi-core processor design scheme. The scheme uses the innovative technologies of on-chip memory copy engine, improved direct memory access (DMA), improved direct cache access (DCA), fast address translation and remote DMA (RDMA) to increase the performance of memory system, network, I/O and parallelism. The experimental results show that when the scheme is used, the TCP bandwidth can be increased by 48.2%, a peak bandwidth 800Mbps can be achieved in a Gigabit Ethernet, the memory copy can reach a speed up of 14 times compared with the traditional methods, the speeds of fast fourier transform (FFT) and matrix multiplication can be doubled, while the cache efficiency is significantly improved.

Key words: cloud computing, multi-core processor, networking optimization, virtual machine, computer architecture