基于SOC的对象存储控制器的设计与实现

郭御风 李 琼 罗 莉 刘光明
(国防科技大学计算机学院 长沙 410073)

摘 要 对象存储重新划分了传统文件系统的功能,并将存储管理功能下放到智能存储设备中。采用基于对象接口,利用智能存储设备的计算能力改善存储性能,获得了更好的可扩展性、安全性以及跨平台无缝共享能力,目前正得到广泛的研究和应用。对象存储控制器是对象存储系统的核心部件,是对象存储系统性能发挥的关键。介绍了一种新型的基于SOC的对象存储控制器的设计和实现。测试结果表明,设计的对象存储控制器在性能、可靠性、成本和功耗方面都具有巨大优势。最后介绍了几种正在研究的对象存储控制器的并行优化方法。

关键词 对象存储,对象存储控制器,文件系统,片上系统,RAID控制器,I/O调度算法
中图法分类号 TP301 **文献标识码** A

Design and Implementation of Object-based Storage Controller Based on SOC

GUO Yu-feng LI Qiong LUO Li LIU Guang-ming
(School of Computer Science, National University of Defense Technology, Changsha 410073, China)

Abstract The object-based storage repartitions the tradition file system functionalities and offloads the storage management functions to intelligent storage devices. With computing power of intelligent storage devices and based on object interface to enhance performance and security of storage system, and data sharing across platforms, Object-based storage has been widely studied and applied. Object-based storage controller is the key component of object-based storage system, and pays import effect to performance of object-based storage system. Design and implementation of an object-based storage controller based on SOC was put forward in this paper, the experimental results show that our object-based storage controller performs well for system performance, reliability, cost and power. Finally, some parallel optimize methods which we are studying on were introduced.

Keywords Object-based storage, Object-based storage controller, Filesystem, SOC, RAID controller, I/O schedule algorithm

1 引言

随着互联网技术的飞速发展以及人们模拟和解决问题的规模越来越大,且数码普及化造成数字信息生产的平民化,数据信息呈几何爆炸式增长,数据存储容量以每年3~5倍的速度急剧增长[1]。IDC(International Data Corporation)调查也发现存储的信息量从2000年的17407TB增长到2005年的308064TB[2],增加了17.7倍。数据存储已经进入了PB量级的海量信息存储时代。另一方面,相对于飞速发展的处理器速度和网络带宽,I/O和存储的性能提高则缓慢得多。按照Amdahl/case准则:一个平衡的计算机系统,其CPU每1MIPS的速度应该对应有1Mb的主存容量和1Mb/s的I/O吞吐率。根据"木桶原理",目前存储和I/O的性能仍然是整个系统的瓶颈。目前存储体系结构主要有3种:直接附加存储(DAS)、网络附加存储(NAS)和存储区域网(SAN)。这3种结构由于自身的缺陷,在实现海量信息存储时都存在不同程度的缺陷。DAS难以大容量扩展,主机往往是系统瓶颈;NAS采用文件级接口,能较好地支持跨平台共享,但文件服务器容易成为性能瓶颈,另外NAS协议开销大、带宽低、延迟长;SAN以数据存储为中心,采用可伸缩的交换网络把客户机和存储设备连接起来,它以块为单位进行数据传输,具有高吞吐率,但会带来安全性和不易实现跨平台共享等问题。

对象存储(OBS)结合了NAS基于文件和SAN基于块的优点,采用了一种新的基于对象的接口,同时具有高性能、高可靠和易跨平台共享等特性,正成为当前存储研究的热点。1999年成立了全球网络存储工业协会(SNIA)的对象存储设备工作组(OSD),并发布了ANSI的X3 T10标准,迈出了对象存储标准化的步伐[3]。对象存储对数据的访问、控制和管理功能进行了重新划分,把文件系统的逻辑结构和物理结构的映射关系交给智能存储设备,由存储设备完成对象到块的映射,增强存储设备的自管理能力。元数据服务器只管理文件的全局视图,实现文件到对象的映射。这样,90%的文件服务工作都改由存储设备完成,大大减少了集中管理的元数据服务器的负载,极大地提高了系统可扩展性。目前对象存储的研究多是集中在对象文件系统上,对象存储也多是基于对象服务器实现。随着存储规模越来越大,系统中的对象存储

到稿日期:2010-01-20 返修日期:2010-03-29 本文受"十一五国防预研"项目——层次式海量存储技术研究(5136040301)资助。
郭御风(1979—),男,硕士,助理研究员,主要研究方向为高性能计算、高性能微处理器和海量信息存储,E-mail:yfguo21@yahoo.com.cn。

设备的数目将会非常巨大，带来的成本和功耗问题也将会越来越突出。

大规模集成电路正在飞速发展，单芯片的晶体管数目仍按照摩尔定律呈倍数增长，从而带来了片上系统(SOC)的快速发展。SOC 把处理器、存储、I/O 等都集成在一个单芯片中，构成一个完整的系统，具有高带宽、低延迟、低功耗等优点。另一方面，SOC 设计多采用成熟的 IP 核集成方式，大大缩短了芯片设计周期，并降低了芯片的设计风险，目前已成为很多芯片设计厂商采用的设计方法。

本文的主要贡献是基于 SOC 设计和实现了一款对象存储控制器芯片 OBSC，并基于 OBSC 设计了对象存储设备 OSD。通过性能评测，发现它在性能和功耗、体积等方面都具有很好的优势。另一方面，我们提出了一些对象存储控制器的优化方法，并进行了分析。下一步我们将在新的对象存储控制器中实现这些优化方法。

本文第 2 节将详细介绍对象存储控制器 OBSC 的体系结构和芯片的性能指标；第 3 节将介绍基于 OBSC 的对象存储设备 OSD；第 4 节介绍一些目前我们正在研究的对象存储控制器的并行优化方法；第 5 节是基于 OBSC 的对象存储设备的性能测试方法和实际性能测试结果；第 6 节简单介绍目前国内外有关对象存储的相关研究；最后对全文进行简单总结。

2 对象存储控制器 OBSC

2.1 OBSC 体系结构

我们采用 SOC 的设计方法设计了一款对象存储控制器 OBSC，它集成了嵌入式处理器、Cache、存控、I/O 控制器以及高速互连接口，内部采用 PLB 总线进行连接。如图 1 所示，对象存储控制器 OBSC 主要以下几个部件组成：

1)PPC440：采用 PPC440 处理器作为 OBSC 的嵌入式处理器，完成全芯片的数据处理和管理工作，支持操作系统微核、驱动程序以及处理和管理进程的运行；目前很多 FPGA 内也嵌入 PowerPC，因此采用 FPGA 进行功能验证比较容易。虽然业界多采用 Intel 的 IOP 处理器，但 IOP 过于通用，应用在应用性质比较单一的 I/O 处理上也暴露了一定缺陷。3Ware 等厂商采用 PowerPC 作 RAID 控制器的处理器，性能评测显示其表现相当突出。

2)Cache：为 PPC440 的二级 Cache，集成在 SOC 芯片内，大小为 512kB，位于 PLB 与 MCU 的数据通路之间，支持两条流水线，可以对存储器两个体并行访问。

3)MCU：DDR2 存储控制器，外接存储芯片；采用 64 位数据总线，支持 ECC，最大可以支持 32Gb 存储空间。

4)IOU：IO 处理单元，负责实现 PLB 总线和 PCI Express 接口的对接，完成 PLB 总线协议到 PCI Express 协议的转换，并完成硬件 I/O 加速和并行优化功能。

5)NI：定制高速网络接口，用于接入系统高速存储网络，构建大规模对象存储系统；自主实现互连通讯协议，支持高带宽、低延迟的串行连接。

6)PCI-Express controller：支持 PCIe1.0 协议，PCI Express 串行链路用于连接外部 I/O 设备；可以通过 PCI Express 的 Switch 进行扩展，挂接多个 PCI Express 设备。

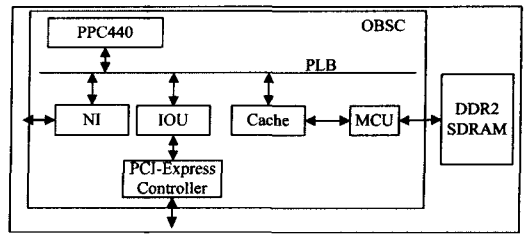

图 1 OBSC 结构框图

2.2 SOC 芯片的性能参数

OBSC 芯片采用. 13u 工艺，芯片面积为 20mm×20mm，封装为 FCBGA575，功耗为 12.4W。芯片集成了 PPC440 处理器核、定制高速互连接口 NI、Cache、DDR2 存储控制器接口、IO 控制逻辑 IOU 和 PCI Express 接口，所有模块通过 PLB 总线连接。其中 PPC440 工作在 500MHz，DDR2 控制器工作频率为 333MHz，IOU 和 PCI Express 接口工作在 250MHz，Cache 大小为 512kB，工作频率为 500MHz，NI 工作在 312.5MHz，PLB 总线为 128b×133MHz。

OBSC 对外有 3 个接口，分别为存储器接口 MCU、PCI Express 接口和高速网络互连接口 NI。其中存储器接口双向带宽为 333MHz×64b×2/8=5.3Gb/s，最大支持 32Gb 内存；PCI Express 接口支持 PCI Express Gen1 标准，串行速率为 2.5Gb/s，支持 16lane，双向带宽为 2.5Gb/s×16×2/8=10Gb/s；高速互连接口采用定制互连协议，支持 12 路串行互连，双向总带宽为 3.12Gb/s×12×2/8=9.36Gb/s。

可以看出，OBSC 芯片提供了非常大的 I/O 存储接口带宽、存储互连网接入带宽和大容量、高带宽的存储器访存接口，内部集成的 PPC440 处理器也具有很强的处理能力，另外，由于 OBSC 采用 SOC 设计，相对于传统的基于服务器方式和基于 IO 处理器方式的对象存储，OBSC 无论是在带宽和延迟方面，还是在功耗、体积和质量方面都有很好的表现。下一步，我们将采用更高带宽的设计，PCI Express 接口将支持 Gen2(5.0Gb/s)，网络接口也将采用 6.25Gb/s 设计，存储控制器将采用 DDR3，频率将会达到 533MHz，这样新的 OBSC 将会有更好的性能表现和并行优化空间。

3 基于 OBSC 的对象存储设备

我们基于 OBSC 设计了对象存储设备 OSD，目前设计的主板只支持一路 OSD。图 2 为我们基于 OBSC 构建的对象存储设备 OSD 结构框图。外接 SDRAM 提供了一个大容量的对象存储 Cache，目前我们在主板上设置了 2G 内存；OBSC 的 PCI Express 链路挂接一个 PLX 公司的 Switch，它有两路 X8 的 PCI Express 插槽，分别插一块 Areca 公司的 PCI Express 接口 RAID 卡，每块 RAID 卡最多可以挂接 8 块盘。但由于目前普通 2U 机箱最多只有 15 个盘位，因此每块 RAID 卡只接 7 块盘，构成 RAID5。由于采用 SOC 设计，OSD 主板的面积非常小。下一步我们准备一块主板上布 4 组 OSD，构成一个存储簇。由于 OBSC 非常小，主板可以放在一个普通的 2U 机箱内，集成度非常高，体积、质量和功耗相对都非常小，也会大大减小供电和散热的压力，非常适合组建超大规模的基于对象存储的海量信息存储。

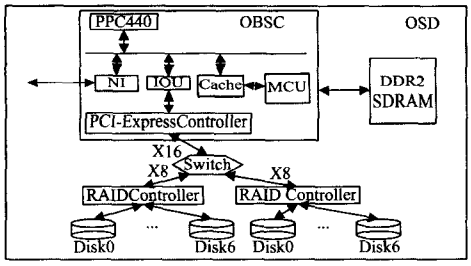

图 2　OSD 结构框图

4　OBSC 的并行优化技术

目前我们只是实现了一个基本功能的对象存储控制器芯片，正在研究采用多种并行优化方法提高对象存储的性能，从而更好地满足大规模并行 I/O 的需求。下面介绍几种我们正在研究的 OBSC 的优化技术。

4.1　多核并行加速策略

OBSC 目前只有一个处理器核。我们可以采用多个处理器核进行并行处理，提高 I/O 带宽。目前 PCI Express 链路带宽远远高于 I/O 存储设备带宽，下一步将采用 PCIe2.0 协议，单路速率为 5.0Gb/s。这样总带宽将提高到 20Gb/s，可以挂接更多的 RAID 卡，从而单 OSD 的存储容量和性能都将得到大大提高。但是处理器核个数和 I/O 存储带宽之间要匹配。设处理器核个数为 N_{cpu}，单 RAID 卡的最大带宽为 W_{raid}，I/O 总线带宽为 W_{io}，RAID 卡的个数为 N_{raid}，从带宽匹配的角度，系统中合理的 RAID 卡个数为

$$N_{raid}=[W_{io}/W_{raid}]$$

假设有一个处理器核专门负责执行各种管理工作，则系统总 CPU 核的个数为

$$N_{cpu}=N_{raid}+1$$

4.2　层次化 RAID 技术

如果采用 PCIe2.0 协议，I/O 带宽单向将达到 10Gb/s。如果每块 RAID 卡的 I/O 带宽按最大 800Mb/s 计算，则一个 OBSC 可以同时支持十几个 RAID 控制器。这样，为了高效地控制下属的 RAID 控制器，提高系统的可靠性和优化 I/O 性能，可以把多个 RAID 控制器组织成一个或多个 RAID，从而构成层次化的 RAID。层次化的 RAID 首先可以使系统的可靠性大大提高，单个 RAID 的不可恢复失效可以通过上一层的 RAID 恢复。随着存储容量的海量化，可靠性越来越被重视。通过层次化可以实现对象存储设备的高可靠；另一方面，层次化 RAID 技术使得对象可以在不同子 RAID 间并行分配和访问，大大提高了访问的并行化，I/O 性能也将得到很大改善。

4.3　对象 Cache 优化技术

传统的 Cache 都是以块为单位组织，并且采用单一的 Cache 策略，很难根据应用的访问特性进行灵活处理，造成虽然采用了预取等方法，Cache 的失效率仍然很高。对象存储中，数据对象除了包含传统的数据外，还带有很多相关属性信息，因此可以根据这些属性所揭示的访问特性灵活管理 Cache，所以我们提出了对象 Cache 的概念。Cache 针对不同的对象属性进行分区，每个分区可以采用不同的块大小、组织形式和独立的 Cache 策略，且不同分区之间互不干扰，可以避免 Cache 预取造成的 Cache 污染问题，从而可以大大降低 Cache 失效率。大量 I/O 访问无需到内存甚至磁盘，大大减小了 I/O 访问延迟，提高了 I/O 性能。我们准备提出一种对象 I/O Cache 的体系结构；定义软硬件接口；并基于对象 Cache 优化预取算法和 Cache 管理算法。

4.4　基于机器学习的并行 I/O 作业调度算法

在大规模并行系统中，对于一个并行作业而言，其执行时间为执行最慢的 Client 完成相关任务的时间。当多个 Client 共享 OSD 时，在 I/O 处理方面，下面两种原因会导致并行作业的执行性能受到严重影响：1）同一个并行作业，单个 Client 的 I/O 访问相对集中，常用的 FCFS 的 I/O 请求调度策略使得其它 Client 的 I/O 请求滞后，从而使得整个并行作业的执行时间延长；2）当有多个并行作业同时运行时，由于其它作业的干扰，使得某个作业中存在的 I/O 请求需要等待较长时间才能调度执行，从而影响作业的整体性能。我们可以采用机器学习方法把从网络收来的对象访问请求重新调度，然后再把对象访问请求转换成块请求，发往 RAID 控制器，使得同一并行作业的 I/O 请求能够在连续的时间段内得到处理，在保证 RAID 中的各个磁盘尽量并行工作的同时，提高整个系统的实际运行性能。

5　测试和性能评测结果

5.1　测试方法

我们的测试系统采用上述第 4 节介绍的 OSD 主板，插入两块 RAID 卡，每块 RAID 卡带 7 块 400GB 的 SATA 盘，构成两个 RAID5。首先使用裸设备测试工具 DD 对单个 RAID 的读写性能进行测试；然后把 OSD 通过高速网络接口连到存储网络上，接一台商用服务器做元数据服务器，并且连接多个客户端，通过文件标准测试程序 IOR，测试多个客户端并发访问 OSD 时的持续读写性能。

5.2　测试结果

测试结果如图 3 所示，单个 RAID 最大读带宽可以达到 775Mb/s，最大写带宽可以达到 615Mb/s；写性能随着数据块大小的增大而提高，由于没有打开读预取，读性能随着数据块大小的增大而略有下降。

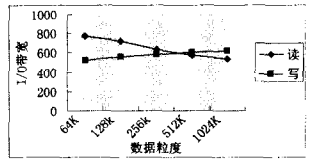

图 3　DD 测试结果

图 4(a)为 IOR 测试的读带宽随着 Client 数增大而变化的情况，图 4(b)为写带宽随 Client 数增大的变化情况。从图中可看出，读写带宽随着 Client 数增大，多个 Client 并发访问，性能都略有下降，但变化比较平缓，说明我们设计的 OSD 能较好地支持多个 Client 的并发访问，并且文件系统下获得的实际 I/O 带宽可达到裸设备 I/O 带宽的 60%～80%左右，性能损失比较小。下一步我们将构建一个多 OSD 的测试环境，获取多个 Client 对多个存储设备进行并发访问时的性能。

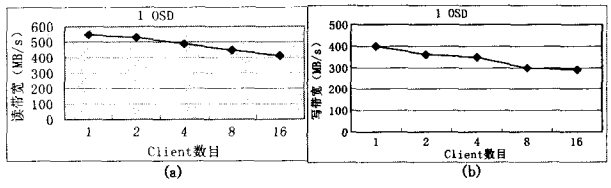

图 4　IOR 测试不同 Client 数并发访问性能

6 相关研究

对象存储源于卡内基梅隆大学(Carnegie Mello University)并行数据实验室(Parallel Data Lab,PDL)的 NASD(Network-Attached Security Disks)项目[4]。NASD 的基本思想是将处理器集成到磁盘驱动器,使它具有一定的智能,能够独立管理其自身的安全、存储和网络通信。University of California,Santa Cruz(UCSC)的存储系统研究中心(SSRC)开发了 OBS 的原型系统 Ceph[5],Ceph 提供了可挂接在 Linux VFS 之下的客户端文件系统,用户使用该客户端可以透明地访问整个存储系统。

目前对象文件系统已经得到了广泛的研究和应用。著名的有 Cluster File Systems 公司的 Lustre[6]、Panasas 公司的 ActiveScale 文件系统[7]、IBM 的 zFS[8]和 Storage Tank[9],Intel 的 iSCSI/OSD 参考原型[10]等。Lustre 是高性能的 Linux 集群文件系统,目前已经得到应用。ActiveScale 文件系统来源于卡耐基梅隆大学的 NASD(Network Attached Secure Disks)项目,目前已是业界比较有影响力的对象存储文件系统。

由于 OSD 是基于对象存储的基础,一些学术机构对之进行了研究。IBM Haifa Research Laboratory 的 Antara[11]是 OSD 最早的一个原型系统。ObjectStone[12]是 IBM 实现的另一个基于对象的控制器原型,它把对象作为文件存储在传统的文件系统之上,以块设备作为存储介质,其主要特点是它实现了标准的 T10 SCSI OSD 协议,并使用 iSCSI 作为 SCSI 命令的传输层。而在 Luster 系统中,存储服务器(Object Storage Target,OST)用商用的 PC 机或服务器实现。OST 对外为对象接口,由内部的过滤器(OBD Filter)把对象的读写转化为对后端文件系统(EXT2/3,ReiserFS,XFS,JFS)的读写,不同的后端文件系统需要不同的过滤器支持。加州大学 Santa Cruz 分校存储系统研究中心(SSRC)提出的 OBFS[13],针对对象负载特点在特定的对象分配策略下对对象的存储作了优化。

国内华中科技大学等也对对象存储进行了较深入研究,他们基于 Intel IOP315 处理器芯片组开发了对象存储设备 OSD,采用交换网络支持多个 I/O 通道的并行数据传输,并实现了 OSD 对象文件系统 HustOSDFS。

结束语 对象存储以其很好的性能优势而成为目前海量信息存储研究的热点,学术界和工业界都投入了极大的热情对其进行研究和开发。但目前这些研究和开发主要集中在基于对象存储服务器基础上的对象文件系统的开发和设计。本文提出的基于 SOC 的对象存储控制器的设计对构建 PB 级海量信息存储系统在成本、性能、功耗和结构等方面具有巨大优势,值得进一步深入研究。下一步我们将对本文中提出的几种对象存储控制器的优化方法进行深入研究,并将在下一款芯片中实现;另一方面,我们也将优化对象文件系统,使得对象存储发挥更大的性能优势。

参 考 文 献

[1] Butler R, Lusk E. Monitors, Message, and Clusters: The P4 Parallel Program System[J]. Parallel Computing, 1994, 20: 547-564

[2] 周蕾. 把握飞速成长的存储市场[EB/OL]. http://cnw2005.cnw.com.cn

[3] Weber R O. SCSI Object-based Storage Device Commands (OSD)[C]// Document Number: ANSI/INCITS 400-2004. International Committee for Information Technology Standards (formerly NCITS). http://www.t10.org/drafts.htm, December 2004

[4] Gibson G A, Nagle D F, Courtright II W, et al. NASD Scalable Storage Systems[C]// Procee-dings of 1999 USENIX Annual Technical Conference, Extreme Linux125 Workshop, 1999

[5] Brandt S A, Ethan M L, Long D E D, et al. Efficient metadata management in large distributed storage systems[C]// Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST2003). 2003: 290-298

[6] Braam P J. The Lustre storage architecture [EB/OL]. Cluster File Systems, Inc. http://www.lustre.org/docs/lustre.pdf, 2002

[7] Tang Hong, Gulbeden A, Zhou Jingyu, et al. The Panasas ActiveScale Storage Cluster-Delivering Scalable High Bandwidth Storage[C]// Proceedings of the ACM/IEEE SC2004 Conference on Supercomputing. 2004: 53-62

[8] Rodeh O, Teperman A. zFS-a scalable distributed file system using object disks[C]// Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies(MSST 2003). 2003: 207-218

[9] Menon J, Pease D A, Rees R, et al. IBM Storage Tank--A heterogeneous scalable SAN file system[J]. IBM Systems Journal, 2003, 42(5): 250-267

[10] Intel Corporation. Intel iSCSI Reference Implementation [EB/OL]. http://www.intel.com/ technology/computing/storage/iscsi/index.htm

[11] Gray R, North B, Turner V. Storage Network Management and Virtualization[C]// IDC. August 2002: 1-5

[12] Factor M, Meth K, Naor D, et al. Object Storage: The Future Building Block for Storage Systems[C]// Proceedings of the 2nd International IEEE Symposium on Mass Storage Systems and Technologies. 2005: 119-123

[13] Wang Feng, Brandt S A, Miller E L, et al. OBFS: A File System for Object-based Storage Devices[C]// Proceedings of the 21st IEEE/12th NASA Goddard Conference on Mass Storage Systems and Technologies(MSST2004). 2004: 101-118

[14] He Shuibing, Feng Dan. Implementation and Performance Evaluation of an Object-based Storage Device[J]. Storage Network Architecture and Parallel I/O, 2007: 129-136

(上接第 246 页)

[12] Hung Ming-chuan, Yang Don-lin. An efficient Fuzzy C-means clustering algorithm[C]// Proceedings IEEE International Conference. 2001: 225-232

[13] 唐旭东,等. 水下机器人光视觉目标识别系统[J]. 机器人,2009,31(2)

[14] Balasuriya A, Ura T. Vision-based underwater cable detection and following using AUVs[A]// Proceedings of the oceans 2002 Conference and Exhibition[C]. Piscataway, NJ, USA: IEEE, 2002: 1582-1587

[15] Nikhil R P, Bezdek J C. On cluster validity for the fuzzy C-means Clustering Algorithm [C]// Proceedings of IEEE International Conference on Data Mining. SanJose, 2001: 225-232

[16] 杨淑莹. 图像模式识别——VC++技术实现[M]. 北京:清华大学出版社,北京交通大学出版社,2005: 161-162

[17] Gonzalez R C, Woods R E. 数字图像处理(2 版)[M]. 北京:电子工业出版社,2002: 98

[18] Bezdek J C. Numerical Taxonomy with Fuzzy Se[J]. J Math Biol, 1974, 1(1): 57-71

[19] Bezdek J C. Cluster Validity with Fuzzy Sets[J]. Journal of Cybernetics, 1974, 3(3): 58-73