计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
47(4): 716-726, 2010

Flash 存储技术

郑文静[1] 李明强[1] 舒继武[1,2]
[1](清华大学计算机科学与技术系 北京 100084)
[2](清华大学信息科学与技术国家实验室(筹) 北京 100084)
(zhengwj07@mails.tsinghua.edu.cn)

Flash Storage Technology

Zheng Wenjing[1], Li Mingqiang[1], and Shu Jiwu[1,2]
[1](*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084)
[2](*Tsinghua National Laboratory for Information Science and Technology, Beijing* 100084)

Abstract Flash memory has the merits of non-volatility, solid-stateness, small volume, light weight, shock resistance, high sequential access performance, high random read performance and low power consumption. Though flash memory has been widely applied in embedded systems and various kinds of technologies have been proposed in this area, recently, with the continual increase in capacity and decrease in cost, there are strong motives from both industrial and academic world to apply flash memory in notebook computers and enterprise storage systems, which raise new challenges for building a high performance, high reliable and low power-consuming storage system based on flash memory. In this paper, a comprehensive analysis and summary of the state of the art of technologies that are related to flash memory are given in the hope of enlightening future work. First, an introduction to the characteristics of flash memory and a discussion on the role of flash memory in storage system architecture are presented. Then, several key techniques for efficient exploitation of flash memory are reviewed in detail, including address-mapping technique, garbage collection mechanism, wear-leveling strategy, flash-aware buffer cache replacement strategy, flash-aware indexing data structure and flash-based transaction technique. Finally, a panoramic view is provided and possible future research areas are given.

Key words flash; storage management; flash translation layer; buffer cache; indexing data structure; transaction; garbage collection

摘 要 Flash 存储器具有非易失性、固态性、体积小、重量轻、抗震动、高性能、低能耗等特点.近年来,随着容量的提高和价格的降低,Flash 存储器在通用计算环境中的应用技术迅速成为研究热点.研究的目的是对 Flash 存储技术研究现状进行分析、总结,以期为进一步的研究工作提供启发.首先介绍了 Flash 的存储特性,探讨了其在存储体系结构中地位,并讨论了管理 Flash 存储器的两种软件体系结构.然后重点分析、总结了 Flash 存储的各项关键技术的研究现状,包括地址映射机制、垃圾回收机制、磨损均衡策略、基于 Flash 的 buffer cache 管理策略、基于 Flash 的索引数据结构以及基于 Flash 的事务处理技术.最后对 Flash 存储技术的研究现状进行总结,提出可能的未来研究方向.

收稿日期:2009-03-19;**修回日期**:2009-06-25
基金项目:国家自然科学基金项目(60873066);高等学校博士学科点专项科研基金项目(200800030027);国家"八六三"高技术研究发展计划基金项目(2009AA01A403)

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

关键词 闪存;存储管理;闪存转换层;缓存;索引数据结构;事务处理;垃圾回收

中图法分类号 TP333.2

0 引 言

从20世纪80年代的一个概念到2007年在全球范围内产生230亿美元收益的产业[1],Flash存储器经历了最初应用于个人计算机BIOS(basic input/output system)存储、嵌入式系统的标准存储器,到目前在某些笔记本电脑中代替磁盘作为外存储器,并被引入到企业级存储的高端存储阵列中,Flash存储技术已经得到很大的发展.作为一种电可擦除可编程只读存储器,Flash存储器不但能在不移除存储芯片的情况下进行擦除和编程操作,还具有非易失性、固态性、体积小、重量轻、抗震动、高性能、低能耗等优点.由于Flash存储器在价格、访问延迟、传输带宽、密度和能耗方面弥补了RAM(random accessed memory)和磁盘之间的差异,关于Flash存储器在存储系统体系结构中的地位探讨成为研究人员关注的问题之一,主要包括主存储器层、RAM与磁盘之间的buffer cache层和持久性存储层3个层次.

和内存、磁盘最显著的不同之处在于,Flash存储器的每个存储单元只有在擦除以后才能写数据.目前擦除操作所需的时间比写操作多一个数量级,并且擦除的基本单位由几十个读、写基本单位组成.此外,每个存储单元允许的擦除次数是有限的.这些特点使得对Flash存储的有效管理变得十分重要.目前,有两种管理Flash存储器的软件体系结构:通过Flash转换层为Flash存储器提供块设备接口,使已有的文件系统不需要经过修改就能运行,为此,Flash转化层实现了地址映射、垃圾回收、磨损均衡等功能;设计并实现专门的Flash文件系统,以充分发挥Flash存储器的特点.

随着Flash存储器的容量越来越大,价格越来越低,Flash存储器相对磁盘的优势越来越明显.尤其是随着绿色存储概念的提出,在个人计算机以及服务器等通用计算环境中使用基于Flash的存储系统迅速成为应用和研究的热点.EMC公司宣布在其高端产品Symmetrix中支持SSD(solid state disk).Google公司为了缓解能耗问题,将美国总部部分服务器的硬盘替换为Flash SSD.百度公司也宣布开始使用其自行研制的SSD.然而,面向通用计算环境的Flash存储系统面临一些新的问题:一方面,和许多移动设备不同,个人计算机和服务器等系统向存储子系统发出大量的随机写请求,而由于Flash存储的独特特性,目前的随机写性能有时比磁盘低几个数量级,因此提高随机写的性能成为Flash存储管理的重要任务之一;另一方面,几十年来,操作系统、文件系统、数据库管理系统等大量上层软件的设计和实现都是基于底层采用磁盘存储系统的假设,大量的数据结构和算法的设计和实现都以优化磁盘系统的性能为目标,如B^+树、buffer cache管理策略等,将这些软件直接用于基于Flash的存储系统无疑将无法有效地发挥Flash存储系统的性能,因而修改或设计新的数据结构和算法以提高Flash存储系统的性能是另一个需要研究的问题.

1 Flash存储器

1.1 Flash存储器介绍

Flash存储器根据其内部架构和实现技术可以分为AND,NAND,NOR,NiNOR几种,目前占据主流市场的有NOR Flash和NAND Flash两大类.NOR Flash由Intel公司于1988年最初推出.为了提高容量/价格比,东芝公司于1989年推出NAND Flash.两种Flash技术各有优、缺点以及各自适用的场合.

NOR Flash和NAND Flash都将存储单元组织为块阵列.块是擦除操作的最小单位,擦除操作将块内所有的位置为"1".页是读、写操作的基本单位.在对页进行写操作(也叫编程操作)之前需要判断该页内所有的位是否为"1".如果全部为"1",则可以进行写操作;否则,需要先对整块进行擦除操作.NAND Flash的页大小通常为512 B,2 KB,4 KB,而NOR Flash能够以字节为单位进行数据访问.NAND Flash的一个块通常包括32,64或128个页.在NAND Flash中,每个页包含数据区和带外区两部分.数据区存储用户数据,带外区存储ECC(error correcting codes),地址映射信息等用于Flash存储管理的信息,对应的大小通常为512 B/32 B,2 KB/64 B,4 KB/128 B.图1给出典型NAND Flash的块和页的结构.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

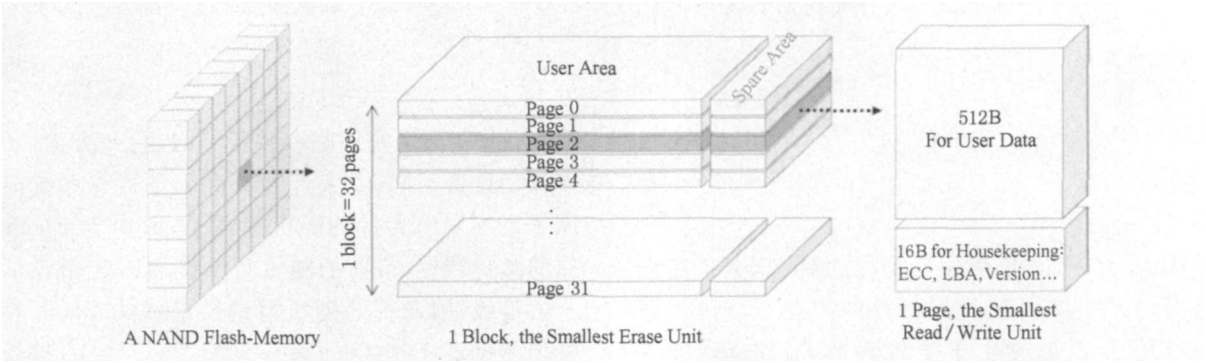

Fig. 1 NAND Flash[2].

图 1 NAND Flash[2]

NOR Flash 以并行的方式连接存储单元,具有分离的控制线、地址线和数据线,具有较快的读速度,能够提供片上执行的功能.但写操作和擦除操作的时间较长,且容量低、价格高.因此 NOR Flash 多被用于手机、BIOS 芯片以及嵌入式系统中进行代码存储.NAND Flash 以串行的方式连接存储单元,复用端口分时传输控制、地址和数据信号,并由一个复杂的 I/O 控制器为主机提供接口.由于对一个存储单元的访问需要多次地址信号的传输,且每次访问 512 B,2 KB 或 4 KB 的数据,NAND Flash 的读取速度较慢.但写操作和擦除操作相比 NOR Flash 较快,且容量大、价格较低.因此 NAND Flash 多被用于数码相机、MP3 播放器、优盘,笔记本电脑中进行数据存储.

1.2 Flash 存储器与 RAM、磁盘的比较

Flash 存储器在价格、访问延迟、传输带宽、密度和能耗等方面弥补了 RAM 和磁盘之间的差异.表 1 给出三者在能耗、读写延迟,读写吞吐率、价格/容量比、可靠性、使用环境方面的对比.

Table 1 Comparison of DRAM, NAND and HDD[3]

表 1 DRAM, NAND 与 HDD 的比较[3]

Memories	Active Power/mW	Read Latency/μs	Write Latency/μs	Read Throughput /MBps	Write Throughput /MBps	Erase Latency/ms	Price/Capacity *	MTBF/h	Operating Temperature/℃
1Gb DDR2 DRAM	878	0.055	0.055	N/A	N/A	N/A	About 100 $/MB	N/A	N/A
1Gb NAND-SLC	27	25	220	100	80	1.5	About 10-35 $/GB	$>2\times10^6$	0～70
HDD	13 000	8 500	9 500	59	60	N/A	About 1-3 $/GB	$<0.7\times10^6$	5～55

The items with * are assessed according to the typical market prices when the paper is written.

综上所述,与其他存储介质相比,Flash 存储器具有如下优点:

1. 与低读、写延迟和包含机械部件的磁盘相比,Flash 存储器的读、写延迟较低;

2. 统一的读性能,寻道和旋转延迟的消除使得随机读性能与顺序读性能几乎一致;

3. 低能耗,能量消耗显著低于 RAM 和磁盘存储器;

4. 高可靠性,MTBF(mean time between failures)比磁盘高一个数量级;

5. 能适应恶劣环境,包括高温、剧烈震动等.

1.3 Flash 存储器在存储体系结构中的地位探讨

Flash 存储技术的迅速发展为研究人员提出了新的挑战:如何设计新的存储体系结构以充分利用 RAM,Flash 存储器以及磁盘的特性.关于 Flash 存储器在新的存储体系结构中的地位,目前可以分为 3 类:代替或部分代替内存;作为磁盘存储系统的读 cache 或预写日志;作为持久性存储系统.

在代替部分内存方面,Wu 等人研究了一种主要由 Flash 存储器组成的非易失性内存存储系统[4];Kgil 和 Mudge 等人提出使用 Flash 存储器代替部分内存,实现只需要少量的 RAM,在可接受的性能损失条件下系统整体能耗的显著降低[5-6];

Flash 存储器在价格、访问延迟、传输带宽、密度和能耗等方面弥补了 RAM 和磁盘之间的差异,使得 Flash 存储器特别适合于作为介于 RAM 和磁

(C)1994-2025 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

盘之间的一级,即读 cache 或预写日志. Dushyanth 等人为使用 Flash 存储器作为读 cache 和预写日志分别作了性能和成本权衡的定量分析[7].

在作为持久性存储系统方面,有两种趋势:用 Flash 存储器完全代替磁盘;使用 Flash 存储器和磁盘的混合存储结构. 在笔记本电脑领域,用 Flash SSD 完全代替磁盘已经成为现实,并且该趋势很可能将延伸到桌面计算机领域. 此外,三星公司和微软公司联合推出的混合式硬盘 Windows ReadyBoost,内置 Flash 芯片,能大幅提高硬盘的性能,提供更快的启动和恢复速度,降低能耗,为笔记本电脑提供更长的电池续航时间[8].

2 管理 Flash 存储器的两种软件体系结构

目前有两种管理 Flash 存储器的方式:通过 Flash 转换层提供块设备访问接口;设计并实现专用的 Flash 文件系统.

Flash 转换层位于通用文件系统和 Flash 存储器之间,为文件系统提供块设备接口,如图 2 所示. Flash 转换层提供对已有的文件系统和大量应用程序的兼容性,减少了开发和测试新软件的代价,是 Flash 存储器快速得到广泛应用的原因之一. 然而,将基于磁盘存储系统设计的文件系统堆叠在模拟磁盘接口的 Flash 转换层之上,不能有效地利用 Flash 存储系统的特点,难以达到性能的最优.

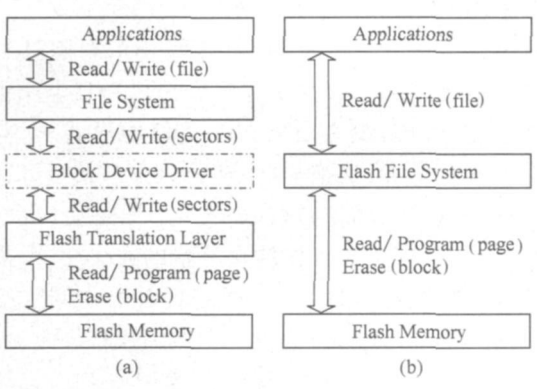

Fig. 2 Software architecture of FTL and special flash file system. (a) Flash translation layer and (b) Flash file system.

图 2 FTL 与专用 Flash 文件系统软件体系结构

专用 Flash 文件系统是指专门为 Flash 存储系统设计、实现的文件系统,它直接管理 Flash 存储器,如图 2 所示. 和 Flash 转换层相比,专用 Flash 文件系统能够充分利用文件系统层次的信息对 Flash 存储器进行存储管理,能提供更多提高性能、可靠性以及降低能耗的机会. 目前已有不少专门的 Flash 文件系统,包括 TrueFFS[9],JFFSx[10],YAFFS[11],SMXFFS[12]等,在国内也存在一些文献资料[13],不在此赘述.

3 Flash 存储关键技术

Flash 存储器独特的擦除操作、非对称的读写性能等特性要求高效的管理技术以提供高性能、高可靠性的 Flash 存储系统. 此外,为了优化基于 Flash 的存储系统的性能,需要重新研究已有的操作系统和文件系统中大量的基于磁盘系统的数据结构与算法,提供面向 Flash 的技术. 本节对这些关键技术的研究现状作全面的分析、总结.

3.1 地址映射

地址映射机制建立逻辑地址和 Flash 存储器的物理地址(物理块号和物理页号)之间的映射关系. 如第 1 节所述,数据更新的写操作包括耗时、耗能的擦除和数据复制操作. 为了提供可接受的写性能,Flash 存储管理借鉴了基于日志的文件系统的原则[14],将更新数据存储在事先已经擦除过的块中,更新地址映射信息,再把原数据标记为无效. 这种异地更新的策略分离了擦除和写操作,有效提高了写性能,但要求动态的地址映射机制的支持. 根据映射表的粒度大小,可分为基于页的地址映射、基于块的地址映射和混合式地址映射.

3.1.1 基于页的地址映射

在基于页的地址映射机制中,地址映射表以页为粒度,包含的表项数与 Flash 存储器的页数相同. 由于粒度细,基于页的地址映射机制具有灵活性高、性能好的优点,但所需的 RAM 空间较大. 在嵌入式系统中,由于价格、能耗等方面的限制,RAM 资源是宝贵的. 尤其随着 Flash 存储器的容量不断增大,基于页的地址映射所需要的 RAM 大小难以得到满足. 为了解决这个问题,基于块的地址映射机制被提出.

3.1.2 基于块的地址映射

在基于块的地址映射机制中,块内数据必须顺序存储. 因此,地址映射表仅需要维护每块的第 1 个页对应的逻辑地址,表项个数与 Flash 存储器的块数相同. 替换块策略(replacement block scheme)就是一种基于块的地址映射机制[15],它为每个需要更新的数据块维护一个或多个日志块,用来存储更新的数据. 由于块内顺序存储的限制,对某页的多次更

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

新将占用多个日志块,导致日志块的空间利用率很低.另外,随着 Flash 存储器容量的增大,目前的大块 Flash 存储器从硬件上要求对块内的所有页必须顺序地写,这进一步加剧了空间浪费问题.

3.1.3 混合式地址映射

混合式地址映射机制利用文件系统的访问局部性的特点,对频繁更新的数据维护基于页的地址映射表,而对大量的其他数据维护基于块的地址映射表,达到以较少的 RAM 空间开销提供高灵活性和高性能的目标,得到研究人员的重视,不少混合式地址映射机制被提出.

日志块策略(log block scheme)[16]放松了替换块策略中对日志块顺序存储的限制,更新数据按写操作的先后在对应的日志块中依次存储,提高了日志块的空间利用率.由于在每页的空闲区中存储对应的逻辑地址,RAM 中地址映射表的大小与基于块的地址映射机制相同.全相联块转换策略 FAST (full-associative sector translation)[17]进一步放松对日志块的限制,允许任何数据块的更新数据存储在任意日志块的任意位置.这进一步提高日志块的空间利用率,降低垃圾回收的频率,提高了性能.为通用计算环境设计的局部性感知的块转化策略(locality-aware sector translation, LAST)[18]将日志块分为顺序日志块和随机日志块.顺序日志缓冲区包含多个顺序日志块,能同时处理多个顺序写流.随机日志缓冲区采用全相联的地址映射机制.超级块策略(superblock scheme)把相邻的 N 个逻辑块合并为一个超级块,为每个超级块分配 M 个日志块.在一个超级块内,日志块和数据块都是共享的,数据块内的数据布局可随更新频率动态改变[19].当 $N=1$,超级块策略退化为日志块策略.当 N 为 Flash 存储器中的总块数时,超级块策略就退化为 FAST.

Flash 存储器特殊的异地更新机制使得设计恰当的地址映射机制成为保证系统性能的重要方面之一.随着 Flash 存储器的容量越来越大以及应用对 Flash 存储器性能的要求越来越高,地址映射机制从简单的单粒度映射到灵活的多粒度映射,有了很大的发展.然而,进一步设计具有自适应能力的地址映射机制仍是值得研究的课题.此外,针对通用计算环境的 Flash 地址映射机制还没有得到充分的研究.

3.2 垃圾回收

由于 Flash 存储采用异地更新的机制,随着系统运行,Flash 存储器中的空闲块越来越少,无效数据越来越多,需要擦除这些包含无效数据的块以得到新的空闲块,这就是垃圾回收机制的功能.垃圾回收过程首先选择一个进行回收的块,将其中的有效页复制到空闲块中,更新地址映射信息,擦除该块,最后把它加入空闲块列表中.由于垃圾回收过程涉及数据复制以及耗时、耗能的擦除操作,还会降低可靠性,因此被认为是 Flash 存储系统的性能瓶颈,成为研究的焦点之一.

3.2.1 垃圾回收的时机

垃圾回收通常根据阈值触发垃圾回收.阈值可以是静态的,也可以是动态的.Kawaguchi 等人提出动态的阈值策略,综合考虑了剩余空闲块个数和垃圾回收的效益[20].为垃圾回收的效益设定阈值.当剩余的空闲块较多时,阈值为无穷大,不进行垃圾回收.当空闲块个数少于一定值后,阈值为有限值并随空闲块数目的减少而降低,最终为 0.动态阈值的方法和静态阈值相比,一方面考虑了垃圾回收的效益,提高了垃圾回收的性能,另一方面也使垃圾回收的启动较慢,避免了存储性能的突然降低.

3.2.2 回收块的选择

对回收块的选择与具体的地址映射以及日志块的管理策略紧密相关,但总的原则是以尽可能少的代价回收尽可能多的页面.为了最大化垃圾回收的效益代价比,通常选择包含无效页个数最多的块进行擦除[21].另一方面,Flash 存储器擦除次数有限的限制,使得垃圾回收中选择擦除的块需要考虑其磨损程度的因素,这在 3.3.1 节中详述.

3.2.3 垃圾回收的合并操作

在混合式地址映射机制(超级块策略除外)中,垃圾回收需要通过合并操作将数据块和日志块中的有效数据重新组织为块内顺序存储的形式.合并操作分为 3 种类型:交换合并、部分合并以及全合并.

1. 当数据更新为顺序进行时,日志块(也叫更新块)中的数据可以直接转化为新的数据块,只需要擦除原来的数据块即可,这样的合并操作叫作交换合并.交换合并只需要进行一个擦除操作即可.2. 当数据更新是顺序的,但在进行垃圾回收时日志块还没有被写满,需要把剩下的有效数据复制到日志块中,再擦除原来的数据块,这样的合并操作叫作部分合并.3. 在一般情况下,对数据的更新不是顺序进行的,需要将数据块和日志块中的有效数据复制到一个空块中,再擦除原来的数据块和日志块,这样的合并操作叫作全合并.3 种合并操作都只产生一个空闲块,但代价却有很大差别,交换合并的代价最小,全合并的代价最高.随机写会导致大量的全合并操

(C)1994-2025 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

作,只有顺序写时才会出现交换合并和部分合并.这就是 Flash 存储对顺序写的性能好,而随机写的性能很差的原因.

表 2 给出几种地址映射机制的垃圾回收代价比较.从中可以看出,提高日志块的空间利用率,降低垃圾回收的频率是提高垃圾回收性能的有效途径,此外,各种机制的设计还应该考虑到为垃圾回收过程提供更多的交换合并的机会.

Table 2 Cost of GC(Garbage Collection) of the Five Strategies.
表 2 五种策略的垃圾回收代价

Strategies	Frequency of GC	Switch Merge	Partial Merge	Full Merge	Total Cost of GC
Replacement Block Scheme	high	many	many	none	high
Log Block Scheme	medium	few	few	many	medium
FAST Scheme	low	few	few	many	low
Super Block Scheme	low	few	none	none	low
LAST Scheme	low	medium	few	few	low

综上所述,作为 Flash 存储系统的性能瓶颈,降低垃圾回收过程的代价成为设计高性能 Flash 存储系统设计的关键技术之一.为此,需要结合系统各个层次上的技术,包括地址映射机制,甚至上层软件的 buffer cache 管理策略、索引数据结构与算法等.关于这些技术将在 3.4 节,3.5 节详述.

3.3 磨损均衡

Flash 的每个存储单元能够进行的写/擦除操作次数有限,超过一定次数以后,数据的可靠性将不能得到保证.通常,SLC(single level cell) Flash 存储器的擦除次数上限为 1×10^5,而 MLC(multiple level cell) Flash 存储器则只有 1×10^4.因擦除次数过多而被磨损的块,一方面减少了 Flash 存储器的可用容量;另一方面可能造成用户数据的丢失.磨损均衡机制的目标是将擦除操作均匀分布在 Flash 存储器中所有的块上,最大化 Flash 存储器第 1 个磨损块出现的时间,提高可靠性.

3.3.1 动态磨损均衡

擦除操作直接在垃圾回收过程中产生,为了磨损均衡,动态磨损均衡在垃圾回收过程中综合考虑磨损程度来选择进行擦除的块.动态磨损均衡也叫第 1 层次的磨损均衡[9].

Kawaguchi 等人提出基于加权的效益/代价比的回收策略[20].效益为块中包含的无效页个数,代价为有效页的复制工作,权重为该块的年龄.这种机制为较长时间不更新而包含较多有效数据的块提供了被回收的机会.由 Chiang 等人提出的回收策略 CAT(cost age times)考虑了块的擦除次数[21],在当一个块的擦除次数接近上限时,将它的数据和擦除次数最少的块交换.Wells 等人提出的回收机制在剩余空闲块较少时给效益分配较大的权重,而在空闲块足够多时给磨损均衡分配较大的权重,但该权重值是预先设定的[22].Kim 等人提出自适应的回收策略 CICL(cleaning integrated with cycle leveling)[23],其权重是磨损不均衡程度的单调函数,随系统运行动态变化.

可见,动态磨损均衡机制朝着动态、自适应的方向发展.然而,由于动态磨损均衡机制在垃圾回收的过程中进行,无法在 Flash 芯片上所有的块范围内实现磨损均衡,同时不可避免地降低了垃圾回收的性能,这些局限促进了对静态磨损均衡机制的研究.

3.3.2 静态磨损均衡

静态磨损均衡机制通过每隔一定时间,将 Flash 存储器上的冷、热数据进行交换以达到磨损均衡的目的.静态磨损均衡也叫第 2 层次的磨损均衡[9].根据触发的机制,静态磨损均衡机制可以分为随机磨损均衡机制和基于擦除次数的磨损均衡机制.

随机磨损均衡机制每隔一定数目的擦除操作,随机地选择一个块进行擦除.Woodhouse 等人提出每 100 次擦除,系统随机地回收一个只包含有效页面的块[10].Ban 提出在系统中保留一个空闲块,每隔 1 000 次擦除操作,系统随机地选择一个块,将它的数据复制到空闲块上,之后该块被擦除,成为新的空闲块[24].随机的磨损均衡策略不需要为每个块维护擦除次数信息.另外,这种技术的开销是可预测并且在时间上是均匀分布的.

基于擦除次数的磨损均衡机制依据块的擦除次数的差别大小触发冷、热数据交换.Lofgren 等人提出保留一个空闲块,当所有块擦除次数的最大值和最小值之差超过一定阈值时,将擦除次数最少的块

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

的数据复制到空闲块上,擦除次数最大的块的数据复制到擦除次数最小的块上,将擦除次数最大的块回收,作为新的空闲块[25].为了降低擦除次数信息的开销,基于群的磨损均衡机制把连续 k 个块组织为一个群,仅在 RAM 中维护每个群的擦除次数信息.如何有效地设计、维护并使用基于群的擦除次数信息成为基于群的磨损均衡机制的主要研究内容[26-27].

经过几十年的研究,多种磨损均衡算法被提出.目前的磨损均衡机制基本能够保证 Flash 存储器为用户提供足够长时间的服务.随着 Flash 存储器的容量越来越大,如何减小磨损均衡机制的空间和时间复杂度,降低系统开销是进一步研究的方向.

3.4 基于 Flash 的 buffer cache 管理策略

buffer cache 广泛地存在于多层存储体系结构中,以弥补不同层次存储器之间性能和价格的差异,实现以较低的价格得到高性能的存储系统的目标.在基于磁盘的存储系统中,已有大量的工作对 buffer cache 的替换算法进行研究[28].然而这些替换算法在基于 Flash 的存储系统上难以实现性能、能耗、可靠性的最优化.原因是它们没有考虑 Flash 和磁盘相比迥然不同的特性,如不对称的读、写性能,不对称的读、写能耗,异地更新策略,磨损均衡问题等.

3.4.1 基于页的替换策略

基于页的替换策略以页为替换的基本单位.考虑到 Flash 存储器写操作耗时为读操作的数倍,能耗大,降低可靠性,还会引起更加昂贵的擦除操作,基于页的替换策略大多对已有的 buffer cache 替换策略进行修改,通过延迟替换脏数据的方法减少写操作.

与传统的以命中率为主要评价指标的替换算法不同,CFLRU(clean-first least recently used)综合考虑了命中率和替换代价两个因素[29],将 LRU(least recently used)队列的 MRU(most recently used)端的 w 个页面叫作 clean-first 区域.当 clean-first 区域中存在干净的页面时,CFLRU 替换最靠近 MRU 端的干净的页面;否则,CFLRU 退化为 LRU 替换策略.可见,CFLRU 通过优先选择干净页面进行替换来延迟对脏数据的替换,提高写命中的机会,同时通过调整 w 的大小限制命中率降低的程度.CFLRU/C,CFLRU/E 和 DL-CFLRU/E 是在 CFLRU 的基础上,以减少对 Flash 存储器的擦除次数以及提高磨损均衡水平为目标的替换策略[30].LIRS-WSR(low inter-reference set-write sequence reordering)在 LIRS 的基础上,延迟替换热的脏数据,以减少写和擦除操作,同时防止命中率的严重降低[31].

3.4.2 基于块的替换策略

基于块的替换策略结合 Flash 存储器擦除操作以块为基本单位的特点,以块为替换单位,力求降低垃圾回收代价,提高性能.

FAB(flash-aware buffer management policy)是针对 PMP(portable media players)的 Flash 感知的 buffer 管理策略[32].FAB 以块为单位对缓冲数据进行 LRU 排队,即当某页被访问时,该页所在的块被移动到 LRU 队列头部.替换页面时,FAB 选择包含页面数最多的块.BPLRU(block padding least recently used)是为 SSD 内的写缓冲区所设计的替换算法,旨在提高 SSD 的随机写性能[33].BPLRU 在 FAB 的基础上,同时使用了页填充技术.页填充技术将随机的写序列填充为顺序写序列,增加垃圾回收中交换合并的机会,降低垃圾回收的代价[34].

可见,与对基于磁盘存储系统的 buffer cache 的大量深入研究相比,对基于 Flash 的 buffer cache 管理策略的研究还处于起步阶段,主要思想是减少对 Flash 存储器的写操作,以及为 Flash 转换层提供更多交换合并的机会,降低垃圾回收的代价.这些替换算法为提高 Flash 存储系统的性能、降低能耗作了初步的尝试,但灵活性不高,可以预见更多针对 Flash 存储系统的自适应 buffer cache 替换算法的出现.此外,目前尚未见到关于基于 Flash 的预取算法的研究.考虑到 Flash 存储器与磁盘截然不同的结构,原有的基于磁道、柱面概念的预取算法不再适用于 Flash.

3.5 基于 Flash 的索引数据结构

索引数据结构提供对大量数据的检索、插入和删除等操作,被广泛应用于数据库管理系统和文件系统中.其中,B 树和 B^+ 树是最重要的索引数据结构之一,其性能对系统的整体性能有重要的影响,在磁盘存储系统中已经得到深入研究.然而,基于 Flash 存储的 B^+ 树面临一些新的问题:对 B^+ 树结点的更新往往很小,可能仅仅修改一个指针值,简单地将整个结点读入 RAM,修改后再写入的方法不能达到性能的最优;B^+ 树在 Flash 文件系统中成为"漫游树",这引起的性能损失是不可容忍的.

3.5.1 基于 Flash 转换层的实现

Wu 等人提出在文件系统和 Flash 转换层之间增加一层软件 BFTL(B-tree flash translation layer)[35].BFTL 在 RAM 中维护保留缓冲区,将对结点的更

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

新以索引单元的形式存储在保留缓冲区中,当保留缓冲区写满以后将索引结点打包成页写入 Flash.一个写操作被平摊在多次更新中,提高了性能.但对易失性 RAM 的使用导致意外断电情况下可能的数据丢失.

Nath 等人在 BFTL 的基础上把 B^+ 树的结点分为两种模式:1.磁盘结点模式在更新时读入整个结点的数据,修改要更新的数据,再写回整个结点,写性能很低,读性能较好,适合于读密集型的结点;2.日志结点模式采用 BFTL 技术,写性能好,读性能低,适合于写密集型的结点,并提出以结点为粒度的自适应 B^+ 树,使每个结点根据自身的工作负载在磁盘结点和日志结点两种模式之间动态转化,达到性能的优化[36].

3.5.2 基于 Flash 文件系统的实现

在基于 Flash 的文件系统中,由于采取异地更新的机制,对叶子结点的更新被存储到新的位置,引起对父亲结点中物理指针值的更新,以此类推,导致从叶子到根结点路径上所有结点的更新,这样的树被称为"漫游树"[11]."漫游树"导致的性能损失是不可容忍的.

JFFS3 通过使用在 RAM 中维护的日志树来合并多次结点更新,减少写操作.但 RAM 的使用导致在断电情况下较长时间的启动[37].Kang 等人提出一种针对 NAND Flash 的有序索引结构 μ 树[37].μ 树的逻辑结构与 B^+ 树相同,但重新设计了物理存储结构.μ 树的结点大小由所在的层次决定.叶结点为第 1 层,占页的一半.离根结点越近,所占的空间逐级减半.这样的设计使得 μ 树能够将从根到叶结点路径上的所有结点存储在一个页当中,更新操作只需要写一个页即可,解决了"漫游树"的问题.

综上所述,目前的研究都着眼于对已有的 B^+ 树的设计和实现进行优化.然而 B^+ 树是为充分提高磁盘性能而设计的,在此基础上进行改进的空间有限.为基于 Flash 的大容量、高性能存储系统设计新的索引数据结构是一种值得考虑的解决方案.

3.6 基于 Flash 的事务处理技术

数据库管理系统、文件系统以及许多面向数据的应用都不同程度地要求对事务处理的支持[38].传统地,这些系统在上层软件中采用写时复制 CoW(copy-on-write)或其变种技术支持事务处理[39].然而,在这些上层软件是软件错误的一个重要来源.另外,为每个应用都单独实现事务处理支持导致大量冗余工作,而在存储层次上提供事务处理接口能显著减少上层软件的复杂性,提高系统的可靠性.实际上,已有一些研究尝试在磁盘存储系统上提供这些接口[40-41].

然而,在磁盘上实现事务处理支持有如下几个缺点:CoW 机制在更新数据时将数据分散在磁盘上.读取分散的数据导致磁盘寻道、旋转延迟.为了弥补这个缺陷,系统通过检查点和清理机制将数据组织回原来的位置.但清理机制的代价是很高的;在磁盘控制器上实现 Cow 引入了显著的复杂性.而在 Flash 存储系统上提供事务处理接口具有以下几个优点:异地更新机制使得 Flash 转换层已经采用了 CoW 的原则,扩展 Flash 转换层以支持事务处理只需引入很少的开销;和磁盘不同,CoW 所导致的数据分片问题在 Flash 存储器中不存在;由于 Flash 存储相对来说较新,提供了制定新的接口规范的可能性[39].

在基于 Flash 的事务处理技术方面,出现了 Atomic Write FTL 和 TxFlash 技术.Atomic Write FTL 通过修改日志块策略增加了原子写接口,使得文件系统能安全地更新元数据[42].它为每个原子写分配一个事务号,在页的带外区写入事务号,通过在一个特定区域中写入提交记录来表示该事务已经提交.系统重新启动时扫描该特定区域判断事务是否提交,并执行必要的回滚擦作,使数据恢复到一致的状态.另外,Atomic Write FTL 还修改对日志块的垃圾回收策略,防止合并操作在事务提交之前破坏数据.TxFlash 技术在页的带外区存储该事务所写的下 1 个页的逻辑地址和版本号,最后一个页指向该事务所写的第 1 个页,使一个事务的所有页形成一个环[39],以是否存在环为依据判断事务是否已经提交.这种环形提交机制避免了写提交记录所需的时间和空间开销,尤其对小的事务而言,显著提高了性能.

Atomic Write FTL 和 TxFlash 都仅设计和实现了原子写接口,这在一定程度上满足了文件系统对元数据更新的原子性要求,但没有提供通用的事务处理支持,无法有效地满足数据库管理系统和其他应用对事务处理的要求.如何在可接受的开销下,提供对通用事务处理的支持仍然是需要进一步研究的课题.

4 总结与展望

本文对目前 Flash 存储技术的研究现状进行了

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

全面而较为深入的分析总结,包括 Flash 的存储特性介绍以及设计和实现高效、高可靠性、低能耗 Flash 存储系统所涉及的几个关键技术.通过以上分析可以看出,Flash 存储相比传统的磁盘存储有很多优势,有望显著提高存储系统的性能,降低能耗,但同时也存在一些缺陷,亟需进一步研究.我们认为进一步的研究需要重点解决两个问题:一是针对特定的存储应用,尤其是对企业级存储系统,如何设计新的硬件和软件体系结构,以充分利用 Flash 存储器的特性,达到最优的性能价格比;二是如何进一步提高 Flash 存储器的随机写性能,降低 Flash 存储器的垃圾回收开销,这是使 Flash 存储器在通用计算环境中得到充分应用需要解决的主要问题之一.要完善地解决这两个问题,需要采用系统的方法,综合考虑硬件、设备驱动以及上层软件中数据结构与算法的设计与实现.

此外,考虑到能耗问题的日益加剧和对绿色存储的强烈需求,以及其优越的性能,Flash SSD 对企业级存储系统具有强大的吸引力.企业级 Flash SSD 要求在降低成本的同时,必须在容量、性能和可靠性方面满足企业级负载的要求.目前,仍有许多问题需要解决:哪种类型的 Flash SSD 适合于企业级存储系统? Flash SSD 带来的性能提升、能耗节约等能否弥补其目前仍然较高的价格劣势? 各种应用怎样将 Flash SSD 引入企业级存储系统中? 考虑到 Flash SSD 的昂贵价格和高性能,Moshayedi 等人认为高端 SSD 适合于系统性能严重依赖于 I/O 性能的场合:

1. 要求大量高速硬盘的应用.为了提供快速的 I/O 性能,这些系统将数据限制在磁盘的外部磁道上,同时将数据条带化在大量的磁盘中.在这样的某个系统中,企业级 SSD 能够以 30∶1 的比例代替硬盘,同时能够减少机架、控制器、电缆、交换机等设备的数量,带来 50%的最初成本降低以及 90%的日后维护和能耗费用降低[43].

2. 系统整体性能的瓶颈是 I/O 子系统性能的应用.在这些应用中,使用相对较低成本,但仍能提供 10 倍于目前硬盘的性能的 SSD 能够在总体性能、能耗和成本之间得到较好的平衡[43].

可见,SSD 在企业级存储领域的应用前景与效益、成本之间的平衡密切相关.随着 SSD 价格的持续降低,其在企业级存储领域中无疑将扮演越来越重要的角色.

综上所述,在通用计算环境中,从短期来看,Flash 存储还不可能完全代替磁盘.但从长期来看,随着 Flash 存储器容量的不断增大、价格的不断降低以及研究的不断深入,Flash 存储将发挥越来越重要的作用,成为主要的存储设备之一.

参 考 文 献

[1] Lai S. Flash memories: Successes and challenges [J]. IBM Journal of Research and Development, 2008, 52(4/5): 529-535

[2] Chang Lipin, Kuo Teiwei. Efficient management for large-scale flash-memory storage systems with resource conservation [J]. ACM Trans on Storage, 2005, 1(4): 381-418

[3] Park C, Seo J, Bae S, et al. A low-cost memory architecture with NAND XIP for mobile embedded systems [C] // Proc of the 1st IEEE/ACM/IFIP Int Conf on Hardware-Software Codesign and System Synthesis. New York: ACM, 2003: 138-143

[4] Wu M, Willy Z. eNVy: A non-volatile main memory storage system [C] //Proc of the 6th Int Conf on Architectural Support for Programming Languages and Operating Systems. New York: ACM, 1994: 86-97

[5] Taeho K, Trevor M. FlashCache: A NAND flash memory file cache for low power Web servers [C] //Proc of the 2006 Int Conf on Compilers, Architecture and Synthesis for Embedded Systems. New York: ACM, 2006: 103-112

[6] Kgil T, Roberts D, Mudge T. Improving NAND flash based disk caches [C] //Proc of the 35th Int Symp on Computer Architecture. New York: ACM, 2008: 327-338

[7] Dushyanth N, Eno T, Austin D. Migrating server storage to SSDs: Analysis of tradeoffs [C] //Proc of the 4th ACM European Conf on Computer Systems. New York: ACM, 2009: 145-158

[8] Microsoft Corp. Explore the features: Performance [EB/OL]. [2008-12-05]. http://www.microsoft.com/windows/windows-vista/features/performance.aspx

[9] Shmidt D. Trueffs wear-leveling mechanism [R]. Newark, CA: M-System, 2002

[10] Woodhouse D. JFFS: The journaling flash file system [EB/OL]. 2001 [2008-12-05]. http://sources.redhat.com/jffs2/jffs2.pdf

[11] Aleph One Ltd. YAFFS: A flash file system for embedded use [EB/OL]. [2008-12-05]. http://www.aleph1.co.uk/yaffs/index.html

[12] Micro Digital Corp. SMXFFS [EB/OL]. [2008-12-05]. http://www.smxinfo.com/rtos/fileio/smxffs.htm

[13] Gu Baogen, Gu Ximei. Research for mechanism of journal filesystem used in embedded environment [J]. Computer Engineering and Application, 2004, 25(6): 915-917 (in Chinese)

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

(顾宝根,顾喜梅. 日志结构的嵌入式文件系统研究[J]. 计算机工程与应用, 2004, 25(6): 915-917)

[14] Rosenblum M, Ousterhout J. The design and implementation of a log-structured file system [C] //Proc of the 13th ACM Symp on Operating Systems Principles. New York: ACM, 1991: 26-52

[15] Ban A. Flash file system: USA, US005404485A [P]. 1995-04-04

[16] Kim J, Kim J M, Noh S H, et al. A space-efficient flash translation layer for compact flash systems [J]. IEEE Trans on Consumer Electronics, 2002, 48(2): 366-375

[17] Lee S, Park D, Chung T, et al. A log buffer based flash translation layer using fully associative sector translation [J]. ACM Trans on Embedded Computing Systems, 2007, 6(3): 1-27

[18] Lee S, Shin D, Kim Y, et al. LAST: Locality-aware sector translation for NAND flash memory-based storage systems [J]. ACM SIGOPS Operating Systems Review, 2008, 42(6): 36-42

[19] Kang J, Jo H, Kim J, et al. A superblock based flash translation layer for NAND flash memory [C] //Proc of the 6th ACM & IEEE Int Conf on Embedded Software. New York: ACM, 2006: 161-170

[20] Kawaguchi A, Nishioka S, Motoda H. A flash-memory based file system [C] //Proc of 1995 USENIX Technical Conf. Berkeley: USENIX Association, 1995: 155-164

[21] Chiang M, Chang R. Cleaning policies in mobile computers using flash memory [J]. Journal of System Software, 2007, 48(3): 213-231

[22] Wells S. Methods for wear leveling in a flash EEPROM memory: USA, US005341339A [P]. 1994-08-23

[23] Kim H, Lee S. An effective flash memory manager for reliable flash memory space management [J]. IEICE Trans on Information System, 2002, 85(6): 950-964

[24] Ban A. Wear leveling of static areas in flash memory: USA, US006732221B2 [P]. 2004-05-04

[25] Lofgren K, Norman R. Wear leveling techniques for flash EEPROM systems: USA, US007353325B2 [P]. 2008-04-01

[26] Jung D, Chae Y, Jo H, et al. A group-based wear-leveling algorithm for large-capacity flash memory storage systems [C] //Proc of the 2007 Int Conf on Compilers, Architecture, and Synthesis for Embedded Systems. New York: ACM, 2007: 160-164

[27] Chang Y, Hsieh J, Kuo T. Endurance enhancement of flash-memory storage systems: An efficient static wear leveling design [C] //Proc of the 44th Annual Conf on Design Automation. New York: ACM, 2007: 212-217

[28] Sun Guozhong, Yuan Qingbo, Chen Mingyu, et al. An improved adaptive buffer replacement algorithm used for second level buffer [J]. Journal of Computer Research and Development, 2007, 44(8): 1331-1338 (in Chinese)
(孙国忠,袁清波,陈明宇,等. 用于二级缓存的一种改进的自适应缓存管理算法[J]. 计算机研究与发展, 2007, 44(8): 1331-1338)

[29] Park S, Jung D, Kang J, et al. CFLRU: A replacement algorithm for flash memory [C] //Proc of the 2006 Int Conf on Compilers, Architecture and Synthesis for Embedded Systems. New York: ACM, 2006: 234-241

[30] Yoo Y, Lee H, Ryu Y, et al. Page replacement algorithms for NAND flash memory storages [C] //Proc of the 2007 Int Conf on Computer Science and Its Applications. Berlin: Springer, 2007: 201-212

[31] Jung H, Yoon K, Shim H, et al. LIRS-WSR: Integration of LIRS and writes sequence reordering for flash memory [C] //Proc of the 2007 Int Conf on Computer Science and Its Applications. Berlin: Springer, 2007: 224-237

[32] Jo H, Kang J, Park S, et al. FAB: Flash-aware buffer management policy for portable media players [J]. IEEE Trans on Consumer Electronics, 2006, 52(2): 485-493

[33] Kim, H, Ahn, S. BPLRU: A buffer management scheme for improving random writes in flash storage [C] //Proc of the 6th USENIX Conf on File and Storage Technologies. Berkeley, CA: USENIX Association, 2008: 239-252

[34] Kim H, Kim J, Choi S, et al. A page padding method for fragmented flash storage [C] //Proc of the 2007 Int Conf on Computer Science and Its Applications. Berlin: Springer, 2007: 164-177

[35] Wu Chinhsien, Kuo Teiwei, Chang Liping. An efficient B-tree layer for flash-memory storage systems [J]. ACM Trans on Embedded Computing Systems, 2007, 6(3): 19-es

[36] Nath S, Kansal A. FlashDB: Dynamic self-tuning database for NAND flash [C] //Proc of the 6th Int Conf on Information Processing in Sensor Networks. New York: ACM, 2007: 410-419

[37] Kang D, Jung D, Kang J, et al. μ-tree: An ordered index structure for NAND flash memory [C] //Proc of the 7th ACM & IEEE Int Conf on Embedded Software. New York: ACM, 2007: 144-153

[38] Sears R, Brewer E. Stasis: Flexible transactional storage [C] //Proc of the 7th Symp on Operating Systems Design and Implementation. Berkeley, CA: USENIX Association, 2006: 29-44

[39] Prabhakaran V, Rodeheffer T, Zhou L. Transactional flash [C] //Proc of the 8th USENIX Symp on Operating Systems Design and Implementation. Berkeley, CA: USENIX Association, 2008

[40] Chao C, English R, Jacobson D, et al. Mime: A high performance parallel storage device with strong recovery guarantees HPL-92-44 [R]. Palo Alto, CA: Hewlett-Packard Laboratories, 1992

[41] Jonge W, Kaashoek M, Hsieh W. The logical disk: A new approach to improving file systems [C] //Proc of the 14th ACM Symp on Operating Systems Principles. New York: ACM, 1994: 15-28

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

[42] Park S, Yu J, Ohm S. Atomic write FTL for robust flash file system [C] //Proc of the 9th Int Symp on Consumer Electronics. Washington: IEEE Computer Society, 2005: 155-160

[43] Moshayedi M, Wikison P. Enterprise SSDs [J]. Queue, 2008, 6(4): 32-39

Zheng Wenjing, born in 1984. Master candidate. Her main research interests include network storage, flash memory technology and hierarchical storage system.

郑文静, 1984 年生, 硕士研究生, 主要研究方向为网络存储、Flash 存储技术和分级存储系统.

Li Mingqiang, born in 1984. PhD candidate. He is currently working on high-reliability, high-performance, and low-cost storage systems.

李明强, 1984 年生, 博士研究生, 主要研究方向为高可靠、高性能和低成本的存储系统研究.

Shu Jiwu, born in 1968. PhD, professor, and PhD supervisor. Senior member of China Computer Federation. His main research interests include network storage, storage security, parallel process technologies, and so on.

舒继武, 1968 年生, 博士, 教授, 博士生导师, 中国计算机学会高级会员, 主要研究方向为网络存储、存储安全和并行处理技术等.

Research Background

Flash memory has the merits of non-volatility, solid-stateness, small volume, light weight, shock resistance, high sequential access performance, high random read performance and low power consumption. Though flash memory has been widely applied in embedded systems and various kinds of technologies have been proposed in this area, recently, with the continual increase in capacity and decrease in cost, there are strong motives from both industrial and academic world to apply flash memory in notebook computers and enterprise storage systems, which raise new challenges to building a high performance, high reliable and low power-consuming storage system based on flash memory. This paper aims at giving a comprehensive analysis and summary of the state-of-the-art of technologies that relate to flash memory in the hope of enlightening future work. First, an introduction to the characteristics of flash memory and a discussion on the role of flash memory in storage architecture are presented. Then, several key techniques for efficient exploitation of flash memory are reviewed in detail, including address-mapping technique, garbage collection mechanism, wear-leveling strategy, flash-aware buffer cache replacement strategy, flash-aware indexing data structure and flash-based transaction technique. Finally, a panoramic view is provided and possible future research areas are given. This study on flash memory is supported by the National Natural Science Foundation of China under grant No. 60873066; the National Grand Fundamental Research 973 Program of China under grant No. 2004CB318205; the Program for New Century Excellent Talents in University under grant No. NCET-05-0067; and the Specialized Research Fund for the Doctoral Program of Higher Education under grant No. 200800030027.

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net