一种高效的混合内存布局机制与编码技术

吴 炀 付印金 陈卫卫 倪桂强
(中国人民解放军理工大学指挥信息系统学院 南京 210007)

摘 要 随着大数据和多核技术的发展,传统内存技术的发展已经远远不能满足大量数据密集型应用涌现所催生的内存计算需求。近年来,新型非易失性存储器(NVM)的兴起与发展为打破传统内存技术瓶颈提供了契机。相变存储器(PCM)作为一种典型的新型非易失性存储器(NVM),与传统内存DRAM各有优势,被认为是最有可能代替传统内存DRAM的存储器,在内存应用中具有很好的发展前景。基于DRAM和PCM的混合内存使得同时发挥DRAM与PCM各自的优势成为可能,故提出一种DRAM与PCM混合内存架构,设计针对混合内存布局的高效读写策略及数据迁移机制,并且在混合内存系统中应用纠删码来提高系统的可靠性。实验表明,此混合内存系统能够大大减少能耗,提高数据吞吐量,同时保证读写的可靠性。

关键词 混合内存,DRAM,PCM,纠删码,读写策略
中图法分类号 TP302.8 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.06.009

Efficient Mechanism of Hybrid Memory Placement and Erasure Code

WU Yang FU Yin-jin CHEN Wei-wei NI Gui-qiang
(Institute of Command Information System, PLA University of Science and Technology, Nanjing 210007, China)

Abstract With the rapid development of big data and multi-core technology, the growth of traditional memory technology, as a matter of fact, has been far away from satisfying the memory computing needs along with the emergence of a large number of data intensive applications. In recent years, the emergence and development of non-volatile memory (NVM) obviously provides an opportunity to break the bottleneck of traditional memory technology. As a typical emerging non-volatile memory, phase change memory (PCM) and traditional DRAM memory have their own advantages. What's more, it is widely considered to be most likely to replace traditional DRAM memory and has very good development prospects in the memory applications. Hybrid memory based on DRAM and PCM makes it possible to play the respective advantages of DRAM and PCM simultaneously. Therefore, in this paper, a hybrid memory architecture of DRAM and PCM was proposed, which is designed to evidently improve the system reliability of the hybrid memory system by using erasure code, based on the efficient reading, writing strategy and data migration mechanism. Experiments firmly show that the hybrid memory system can greatly reduce energy consumption, obviously improve the throughput, and ensure the reliability of reading and writing.

Keywords Hybrid memory, DRAM, PCM, Erasure codes, Reading and writing strategy

1 引言

大数据的处理与分析给计算机存储系统带来了巨大的困难与挑战,尤其是现有的内存技术已经远远不能满足大量数据密集型应用涌现所催生的内存计算需求,以及越来越快的处理器对内存速度、容量、功耗、可靠性提出的新要求。根据摩尔定律,计算速度每18个月增长一倍,然而内存技术的发展已经遇到瓶颈,形成了巨大的差距[1]。现有的以动态随机存储器(DRAM)为代表的传统内存技术由于工艺制程的限制,再加上其本身物理特性的制约,电子的微观特性越来越明显。DRAM是易失性存储器,若想保存其中的数据,必须不断地对其进行充电,而且使用时需要动态刷新,这导致其面临着时延、功耗、可靠性方面的困境。

近年来,新型非易失性存储器(Non-Volatile Memory, NVM)的兴起与发展为打破传统内存技术的系统性能与能耗瓶颈提供了契机,正推动计算机存储系统结构的变革。相变存储器(Phase Change Memory, PCM)是最具代表性的新型非易失性存储器,具有非易失、低功耗和大容量的优势,其性

到稿日期:2016-11-11 返修日期:2017-01-04 本文受国家自然科学基金项目(61402518)资助。
吴 炀(1992—),男,硕士生,主要研究方向为纠删码和云计算,E-mail:1135773457@qq.com;付印金(1984—),男,讲师,主要研究方向为大数据管理、网络存储和云计算;陈卫卫(1967—),女,教授,硕士生导师,主要研究方向为云计算和大数据管理,E-mail:njcww@qq.com(通信作者);倪桂强(1966—),男,教授,博士生导师,主要研究方向为网络管理、大数据管理和图像处理。

能虽然远高于非易失的闪存介质，但仍不如 DRAM。表 1[2] 从工艺尺寸、能耗、读写性能等方面比较了 DRAM 与 PCM 的特性。

表 1 DRAM 与 PCM 的性能对比

参数	理论工艺制程级别/nm	读操作时间/ns	写操作时间/ns	读能耗/mW	写能耗/mW	读过程破坏性	数据保持力
DRAM	20	<10	<10	210	195	破坏性	刷新
PCM	5	10～100	20～120	78	773	非破坏	>10 年

部分研究者使用了 PCM 直接内存技术，无论通过 MLC 与 SLC 的转换[3]或者重新利用具有 bit 错误的页面[4]，还是设计计数器加密模式[5]或者研究增量式加密方法[6]，都带来了巨大的能耗，增加了开销。

为构建适应大数据分析的内存系统，研究者们提出了用部分 PCM 取代 DRAM 来共同构建混合内存的设计方法[1,7,13]，两种存储介质在相互弥补不足的同时能充分发挥各自的优势。Li 等人[1]提出了一种基于混合存储架构的页面替换算法，PCM 中的页面采取主动替换的方式将合适的页面替换到 DRAM 中，DRAM 中的页采取被动的方式调入到 PCM 中。此系统虽然能够取得较好的效果，但涉及的数据结构过多，产生的能耗比较大，而且无法保证数据的可靠性。Wang 等人[8]提出了一种基于访存热点控制的有效策略，该策略虽然能够延长 PCM 的使用寿命，但读写分布对 DRAM 区域的占用率较高，而且没有考虑到可靠性方面的内容。根据表 1，PCM 写能耗太大，因此需要尽量减少 PCM 的写操作。本文提出一种新的混合内存机制，利用简单的写缓存、日志记录表及纠删码，创造性地将原始数据、校验数据分别放入 PCM 和 DRAM，发挥 DRAM 与 PCM 各自的优势，大大降低了存储系统的能耗，减少了读写延迟，增强了数据的可靠性。

本文第 2 节介绍了混合内存领域的相关工作；第 3 节详细阐述了所提的混合内存布局及纠删码策略；第 4 节给出了实验结果与分析；最后对未来进行展望。

2 相关工作

现有的混合内存架构大致分为两类：横向混合内存和纵向混合内存。横向混合内存是对 DRAM 和 PCM 统一编址，将它们看成一个整体。在此架构中，PCM 与 DRAM 地位平等，由操作系统中的统一页表来进行地址转换。对于纵向混合内存，主存全部由 PCM 组成，而缓存由一块容量较小的 DRAM 来充当。由于 PCM 的存储密度比 DRAM 的大，因此系统运行时可以保存更多数据，并能有效减少对内存的访问。同时，DRAM 速度快，可以作为主存的缓存。图 1 示出了两种不同的存储架构。

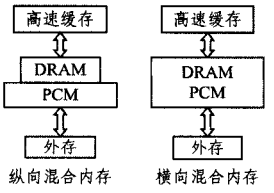

图 1 两种不同的存储架构

横向混合内存将 PCM 与 DRAM 统一编址，具有很强的扩展性[9-10]，比如 Zhang 等人[9]根据页面修改的频繁程度提出了一种混合内存系统调度数据的策略。在实现过程中，通过 MQ(Multi-Queue)算法把修改频繁的页迁移到 DRAM 中，算法维持了 16 个 LRU 队列，将页挂入队列中，并对修改次数计数，当达到阈值之后，便认为是频繁修改的页。该方案发挥了 PCM 和 DRAM 的优势，但队列开销很大，并且迁移操作影响系统的整体性能。Li 等人[1]提出的页面替换算法采取主动替换的方式将 PCM 合适的页面替换到 DRAM 中，当 DRAM 没有空闲页时调用改进的 CLOCK 算法将页面调入 PCM 中。此系统虽然能取得较好的效果，但涉及的数据结构过多，产生的能耗大，而且无法保证数据的可靠性。

纵向混合内存将 DRAM 作为 PCM 的缓存，利用空间局部性响应 CPU 的需求[11-13]。Qureshi 等人[11]描述了由小容量 DRAM 缓存和 PCM 构成的混合内存系统，使用页表管理 PCM 介质，采取 Lazy-Write Organization 和 Line-Level Writes 等策略来减少 PCM 的写操作。该系统能够隐藏 PCM 与 DRAM 之间的延迟差距，但空间开销太大，而且比较操作会带来性能延迟。Park 等人[12]提出了运行时自适应的 DRAM 功耗递减策略、绕开 DRAM 策略和保持脏数据策略，此方案从降低系统功耗出发，但忽略了实现的复杂度。

本文采取横向混合内存架构，将原始数据放入 PCM，将校验数据放入 DRAM，并且维持 DRAM 写缓存，此布局机制能够有效减少 PCM 写操作的次数。同时，本文通过维持日志记录表，降低系统的功耗，减少读写延迟，使用纠删码策略保证数据的可靠性。

3 一种高效的混合内存布局及其纠删码策略

3.1 高效的混合内存布局

由于 PCM 的写操作寿命有限，因此需要尽量减少对 PCM 的写操作。本文在内存中应用了纠删码策略，将数据分为原始数据与校验数据，考虑到校验数据的写操作频率远高于原始数据的写操作频率，将原始数据存储于 PCM，将校验数据存储于 DRAM。同时，为了应对 PCM 频繁的写请求，设计一部分 DRAM 作为 PCM 的镜像缓存，并且采取日志记录对镜像缓存的数据进行索引。图 2 展示了混合内存的布局机制。

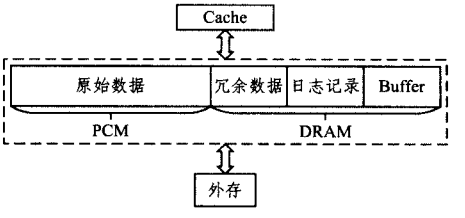

图 2 混合内存的布局机制

如图 2 所示，由于冗余数据、日志记录需频繁修改，因此两者与写缓存数据都存储于 DRAM 中，以减少 PCM 的写操作；而 PCM 作为存储原始数据的地址，与 DRAM 共同构成整个内存系统，PCM 与 DRAM 在空闲状态进行数据迁移。内存系统通过 PCM 与 DRAM 的接口与外界进行交互。

3.2 混合内存读写策略

针对 3.1 节中的数据布局，为充分发挥 DRAM 和 PCM

各自的优势,分别对读请求和写请求的读写策略进行设计。

为描述清晰,表 2 列出了策略中定义的英文名词。

表 2　本文英文名词的说明

英文名词	名词意义
D_{LR}	Log record,当前日志记录
D_{NLR}	New log record,新日志记录
D_{Addr}	Address,数据地址
P_{OD}	Original data,原始数据
$P_{OD}(n)$	Original data,某条带上第 n 个数据
D_{EC}	Erasure codes,某条带的纠删码(校验数据)
find()	查找当前日志记录,命中返回 true,否则返回 false
getaddr()	根据日志记录,取出数据地址并返回
getdata1()	根据数据地址取出数据并返回
erasure()	计算当前数据所在条带的校验码并返回
getdata2()	查找取出 PCM 中的数据并返回
delete()	删除当前日志记录
creat()	创建新的日志记录
writecode()	在 DRAM 中写入校验码
writedata()	写入原始数据
getlog()	获取一条日志记录

3.2.1　读请求

当总线上传来读请求时,内存系统响应读请求,从内存读出数据。首先,根据请求查找日志记录,若记录存在,则说明数据存储于写缓存中,从日志记录中读出数据存储地址,再根据数据存储地址从 buffer 中读取数据,最后计算数据对应的纠删码,若其与 DRAM 存储的纠删码一致,则返回数据,否则返回 error;若记录不存在,则查找 PCM 中的数据,再计算数据对应的纠删码,若与 DRAM 存储的纠删码一致,则返回数据,否则返回 error。算法的伪代码如算法 1 所示。

算法 1

```
If(find(D_LR)) {
  D_Addr = getaddr(D_LR);
  P_OD = getdata1(D_Addr);
  D_EC = erasure(P_OD);
  If(D_EC == D_EC')
    Return P_OD;
  Else
    Return error;
}
Else{
  P_OD = getdata2( );
  D_EC = erasure(P_OD);
  If(D_EC == D_EC')
    Return P_OD;
  Else
    Return error;
}
```

在此过程中,buffer 中所有数据均为最近更新所得。首先必须从 buffer 中获取,若数据存储于 buffer 中,检错之后即可返回数据;否则,从 PCM 中返回相应的数据。这保证了读出的数据均为最新数据,此过程使用纠删码检错纠错,同时保证了数据的可靠性。

3.2.2　写请求

当总线上传来写请求时,内存系统响应写请求,并将数据写入内存。首先,查询日志记录,判断 buffer 中是否存储了相应数据,若命中,则根据命中的日志记录取出数据的地址,并删除当前日志记录,更新一条新日志记录,根据地址将数据写入缓存,计算数据相应的纠删码,将纠删码写入 DRAM 中;若不命中,则创建一条新的记录,将数据写入缓存,再计算数据相应的纠删码,将纠删码写入 DRAM 中。

算法的伪代码如算法 2 所示。

算法 2

```
If(find(D_LR)) {
  D_Addr = getaddr(D_LR);
  delete(D_LR);
  creat(D_NLR);
  writedata(D_Addr, P_OD);
  D_EC = erasure(P_OD);
  writecode(D_EC);
}
Else{
  creat(D_NLR);
  writedata(D_Addr, P_OD);
  D_EC = erasure(P_OD);
  writecode(D_EC);
}
```

由于写缓存存储最近更新的数据,因此需要将总线上最新的数据写入缓存,分为两种情况:若缓存中已存在,则只需更新即可;否则需在写入缓存之后创建新的日志记录。最后,更新其所在条带的纠删码。由于所有的数据都写入 DRAM 镜像缓存中,针对总线频繁写请求时,由 DRAM 而非 PCM 执行,因此能大大减少 PCM 的写操作,明显降低写能耗与写延迟,延长 PCM 的使用寿命。

3.2.3　空闲状态

每一时钟周期对系统的当前状态查看一次,当系统空闲时,首先取出一条日志记录,获取其数据地址,根据地址从 buffer 中读取数据,再将数据写入 PCM 中,删除当前日志记录,接着重复执行上述回收利用操作,直到日志记录为空。算法的伪代码如算法 3 所示。

算法 3

```
While(D_LR != NULL)
  {
    getlog(D_LR);
    D_Addr = getaddr(D_LR);
    P_OD = getdata1(D_Addr);
    writedata(D_Addr', P_OD);
    delete(D_LR);
  }
```

回收利用操作仅在系统空闲状态执行,在执行的过程中当系统需要相应读写请求时,立即中断回收利用操作。在执行回收利用操作时,根据日志记录进行写回,而且写回数据之后必须删除其日志记录,以免发生后续读写错误。由于在相应写请求的过程中已经对校验码进行了更新,因此在执行回收利用操作时不需更新校验码。

3.3　内存纠删码

为了提高存储系统的可靠性,将纠删码应用到存储系统

中。与传统的镜像副本技术相比，基于纠删码的方法具有冗余度低、磁盘利用率高等优点[14]。

然而，在存储系统中，纠删码通常应用于磁盘阵列，在内存中的应用少之又少。由于上述混合内存由 PCM 存储原始数据，为了保证其数据的可靠性，可以考虑使用纠删码技术。常用的纠删码技术有 RS(Reed-Solomon)编码[15]，EVENO-DD[16]，RDP[17]，X-code[18]，P-code[19]等。

本文的混合内存采用类似 RAID4 编码[20]，即采用最简单的异或操作求出原始数据对应的校验数据，纠删码的数据信息和校验信息布局示意图如图 3 所示。

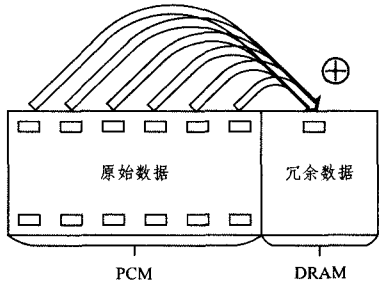

图 3 纠删码布局示意图

对同一 PCM 条带上的数据 $P_{OD}(1)$，$P_{OD}(2)$，$P_{OD}(3)$，…，$P_{OD}(n-1)$，$P_{OD}(n)$，求出的校验码如下：

$$D_{EC}=\bigoplus_{t=1}^{n}P_{OD}(t)$$

在响应读请求的过程中，求出读取数据 P_{OD} 所在条带的校验码为 D_{EC}，将其与存储的校验码 D'_{EC} 比较，若 $D_{EC}=D'_{EC}$，则返回正确数据 P_{OD}；否则返回 error。在响应写请求的过程中，每写入一次数据，就计算一次写入数据 P_{OD} 所在条带的校验码 D_{EC}，并将 D_{EC} 写入 DRAM。

内存数据更新频繁，特别是写入数据所引起的校验码更新额外增加了一次对 DRAM 的写操作，所以必须最大限度地减少计算校验码所引起的功耗和额外时延，故选取简便易算的纠删码，以在保证数据可靠性的同时避免过大的开销。

3.4 日志记录

根据 3.2 节所述，上述算法中日志记录的创建、删除与更新是比较重要的一环。简而言之，日志记录实质上是由操作系统所维持的一个为写缓存索引数据的数据结构，其结构如图 4 所示。

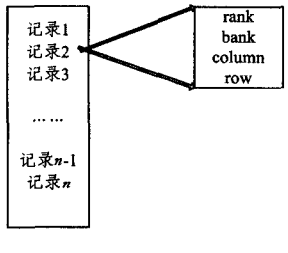

图 4 日志记录结构

如图 4 所示，日志中存储多条记录，每一记录包含 4 个字段，rank 表示数据所在的 channel 中的 rank 号，bank 表示数据所在 rank 的 bank 号，而 column 与 row 表示所在 bank 的行号和列号。

日志记录的创建、删除、修改过程虽然造成了一定的额外开销，但是其作用也是非常明显的，即读写请求时明显减少了查找数据是否存储于写缓存所带来的功耗与延迟。不仅如此，日志记录由于仅仅存储写缓存的索引，因此规模较小，降低了读写时对日志记录操作的开销。

4 实验分析与结果

4.1 实验环境

实验环境为 Linux 系统，Ubuntu 版本为 Ubuntu12.04，虚拟机硬件设置为：内存 1GB，硬盘(SCSI)20GB，处理器数量为 1。使用 DRAMsim2 模拟器进行仿真模拟。DRAMsim2 是一个主要模拟 DRAM memory 读写访问延迟和工作能耗的工具，其因模拟结果与实际运行结果非常接近而被科研工作者广泛使用。

实验过程中，通过改变 system.ini 等配置文件中的参数进行模拟，将 DRAM 存储系统、PCM 存储系统、PCM+DRAM 混合内存系统进行对比实验。图 5 是 DRAMsim2 的逻辑结构图。

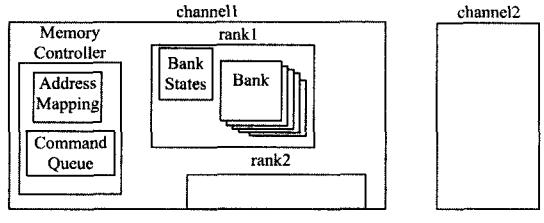

图 5 DRAMSim2 的逻辑结构

DRAMSim2 可以模拟多 channel(通道)的内存，每个 channel 内都是一个独立的内存系统。系统中的结构如下：内存控制器，其完成地址映射，记录指令序列，记录各 rank 状态；channel 分为多个 rank，对于每个 rank，bank 控制器控制 bank 状态，每个 rank 包含多个 bank，每个 bank 都是地址范围相同的行列矩阵，即读写访问的最终地址。

本文所对比的内存系统结构包括以下 6 种：1)DRAM-no code，未使用纠删码的 DRAM 内存系统；2)PCM-no code，未使用纠删码的 PCM 内存系统；3)DRAM+PCM-no code，未使用纠删码的混合内存系统；4)DRAM-code，使用纠删码的 DRAM 内存系统；5)PCM-code，使用纠删码的 PCM 内存系统；6)DRAM+PCM-code，使用纠删码的混合内存系统。本文所用到的系统文件的参数如表 3 所列。

表 3 系统参数配置

系统	总容量/M	Rank 数	Devices per rank	Sim2 clock Frequency/Hz	CPU clock Frequency/Hz
1	4096	1	16	$6.5*10^8$	$6.5*10^8$
2	2048	1	8	$4.0*10^8$	$4.0*10^8$
3	2048	1	8	$6.5*10^8$	$6.5*10^8$
4	2048	1	16	$8.0*10^8$	$8.0*10^8$
5	2048	1	16	$6.5*10^8$	$6.5*10^8$
6	2048	1	16	$4.0*10^8$	$4.0*10^8$

在以上 6 种系统中，分别对 6 种内存结构进行实验，对其读写共测试 100 次(读 98 次，写 2 次)。

4.2 评估指标

实验以带宽(GB/s)与功耗(W)为评估指标。每一种信

号都要占据一定的频率范围,称该频率范围为带宽。为了使信号在通过信道时失真尽可能小,要求信道的带宽尽可能宽。计算机系统的带宽是指单位时间内所能执行的操作数目,通道或存储带宽是指它们的数据传输率。由于目前系统处理的数据量都是相当巨大的,因此几乎每一步操作都要经过内存,这也是整个系统中工作最为频繁的部件,从而内存的带宽就在一定程度上决定了整个计算机系统的性能。内存的带宽可由如下公式计算:

带宽=总线宽度 * 总线频率 * 一个时钟周期内交换的数据包个数

根据内存的带宽公式,同一系统中总线宽度与总线频率恒定,而一个时钟周期内交换的数据包个数可反映内存性能的优劣,故采取带宽作为评估指标是合理的。

功耗作为一个重要的指标,是单位时间内所消耗的能源数量。由于内存功耗占计算机系统功耗的比例越来越大,功耗已经成为高性能计算系统设计必须考虑的重要问题,降低内存系统功耗的需求越来越迫切。本文的内存系统由 DRAM 与 PCM 组成,对如何最大限度地减少整个内存系统的功耗具有重要意义。

4.3 实验结果与分析

4.3.1 结果分析

所有的仿真结果如图 6、图 7 所示。图 6 中横轴表示 6 种不同的系统配置,每种系统配置都有 6 种系统结构;纵轴表示功耗(单位是 W)。图 7 中横轴同图 6;纵轴表示带宽(单位是 GB/s)。

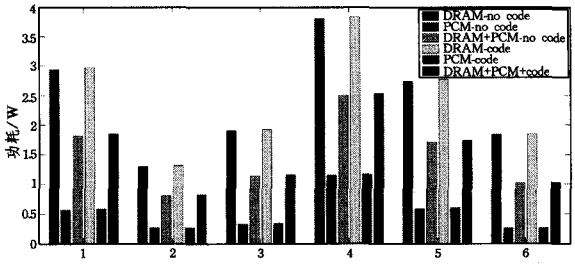

图 6　内存系统功耗比较

从图 6 可以看出,在同一系统配置中,无论是否使用纠删码,DRAM 与 PCM 的混合内存系统功耗都比 DRAM 降低了 30%～40%。这是因为当系统执行读请求时,首先查找日志记录,读出原始数据的地址,然后从存储器中读取数据,这与直接在存储器中查找数据相比大大降低了功耗。当系统执行写请求时,首先写入 DRAM 缓存,减少了对 PCM 的写操作,而 PCM 写操作的能耗是 DRAM 的 4 倍,故能够最大限度地减少写操作的功耗。

在不同的系统配置中,系统功耗各不相同,相比于另外 5 种系统,4 号系统的功耗最大,这是由于此系统的频率最高,单位时间内执行的总线请求最多。

本文的混合内存布局技术与读写策略在功耗上充分体现了优越性,提高了系统性能。

从图 7 可以看出,在同一系统配置下,无论是否使用纠删码,DRAM 与 PCM 的混合内存系统带宽都比 PCM 高 3 倍左右,接近 DRAM 的带宽,说明混合内存系统在执行读写请求时 DRAM 写缓存与日志记录几乎可以隐藏 DRAM 与 PCM 延迟的差距。当写请求到来时,先写入 DRAM 缓存,能够最大限度地利用 DRAM 高带宽的特点。

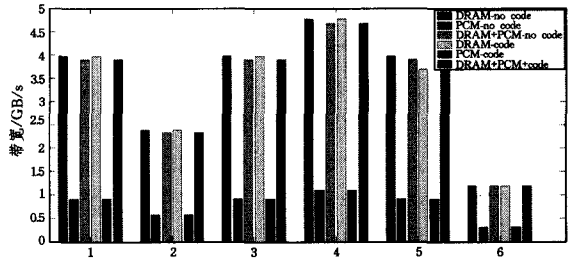

图 7　内存系统带宽比较

在不同的系统配置中,系统带宽各不相同,4 号系统的带宽比另外 5 种系统都大,这是由于此系统的频率最高,总线的频率最高,而且在一定的时间内,执行读写请求数更多,从而交换的数据包数量最多。

本文的混合内存布局技术与读写策略能够最大限度地提高系统带宽,达到充分利用资源的目的。

4.3.2 性能分析

(1)功耗分析

当内存系统加入纠删码之后,带宽相差不大,功耗额外增加额度如表 4 所列。

表 4　功耗增加额度/%

系统	1	2	3	4	5	6
DRAM	1.1	1.6	1.1	1.0	1.1	1.0
PCM	2.6	3.0	3.0	1.5	2.6	2.0
DRAM+PCM	1.8	2.0	1.8	1.4	1.8	1.0

从表 4 可以看出,加入纠删码之后,功耗增加的额度都在 1%～3%之间,虽然造成了少量功耗的增加,但是理论和实验已经证实,RAID4 纠删码能够减少读写错误,提高系统可靠性[13]。因此,在本文提出的混合内存系统中应用 RAID4 纠删码是可行的。

(2)可靠性分析

本文采用 RAID4 编码来保证数据读写的可靠性。假设每一条带上某一原始数据出错的概率为 $1/p$,则出现不能恢复错误的概率是 $\sum_{k=2}^{m} 1/p^k$(m 为每一条带的原始数据量),比不使用纠删码的出错概率减少了 $1/p$,保证了整个系统数据在读写过程中的可靠性。

(3)空间冗余度分析

本文所述的类似 RAID4 编码存储的校验数据量是原始数据量的 $1/n$(n 为条带的数量),与镜像副本相比,具有较高的冗余度,节省了内存空间。当读取数据发生错误时,能够利用校验码对数据进行重构。当执行写请求时,对同一条带上的数据进行简单的异或操作能够最大限度地减少时延与功耗。

结束语　随着计算机科技的进步和存储系统海量化的发展,相变存储器被广泛应用于未来计算机内存子系统的开发。在未来,以 PCM 为代表的非易失性内存将进一步在计算机存储系统中崭露头角,研究并发展新型的、更加节能且有效的

存储器是整个计算机领域将来能够持续而快速发展的重中之重。相对于 DRAM,PCM 具有非易失性、密度高、容量大等优点。但是 PCM 的写操作次数非常有限,能够承受的写次数为 $10^6 \sim 10^8$,具有相对较大的延迟。采用混合内存结构能够使得存储系统兼具 PCM 与 DRAM 的优点,本文在内存中应用了纠删码策略,将原始数据存储于 PCM,将校验数据存储于 DRAM;同时,为了应对 PCM 的频繁写请求,将一部分 DRAM 作为 PCM 的镜像缓存,并且采用日志记录对镜像缓存的数据进行索引。实验表明,该机制能够有效降低系统功耗,提高系统带宽,延长 PCM 器件的寿命,保证内存系统读写的可靠性。

本文的工作还处于初步阶段,还在需要进一步探讨和研究的内容:

1)为了保证 PCM 中原始数据的可靠性,采用的是最简单的 RAID4 的纠删码,这需要进一步完善,需要研究更加适合混合内存系统的纠删码技术。

2)对于使用的 DRAMsim2 模拟器,已经有了一定的研究成果,并且人们能够根据自己的需要进行源代码的修改。后续可以将其放入 GEM5 模拟器并结合 NVMain 进行仿真,以取得更好的效果。

参 考 文 献

[1] LI X. Efficient page replacement algorithm based on hybrid memory[D]. Jinan: Shangdong University, 2015. (in Chinese)
李骁. 基于混合架构的高效页面替换算法的分析[D]. 济南:山东大学,2015.

[2] MAO W, LIU J N, TONG W, et al. A Review of Storage Technology Research Based on Phase Change Memory[J]. Chinese Journal of Computers, 2015, 38(5): 944-960. (in Chinese)
冒伟,刘景宁,童薇,等. 基于相变存储器的存储技术研究综述[J]. 计算机学报,2015,38(5):944-960.

[3] QURESHI M K, SRINIVASAN V, RIVERS J A. Scalable high performance main memory system using phase-change memory technology[J]. ACM SIGARCH Computer Architecture News, 2009, 37(3): 24-33.

[4] CHEN J, WINTER Z, VENKATARAMANI G, et al. rPRAM: Exploring redundancy techniques to improve lifetime of PCM-based main memory[C]// Proc of the Parallel Architectures and Compilation Techniques (PACT) International Conference. Galveston TX, 2011: 201-202.

[5] KONG J, ZHOU H. Improving privacy and lifetime of PCM-based main memory[C]// Proc of the IEEE/IFIP International Conference on Dependable Systems and Networks(DSN). Chicago, 2010: 333-342.

[6] CHHABRA S, SOLIHIN Y. i-NVMM: A secure non-volatile main memory system with incremental encryption[C]// Proc of the 38th Annual International Symposium on Computer Architecture(ISCA). San Jose, 2011: 177-188.

[7] KWON S, KIM D, KIM Y, et al. A case study on the application of real phase-change RAM to main memory subsystem[C]// Proc of the Conference on Design, Automation and Test in Europe. San Jose, 2012: 264-267.

[8] WANG Q, CHEN L, HAO X R. Efficient Management with Hotspots Control for Hybrid Memory System[J]. Microelectronics & Computer, 2014, 31(1): 1-4. (in Chinese)
王强,陈岚,郝晓冉. 一种混合内存系统访存热点控制方法[J]. 微电子学与计算机,2014,31(1):1-4.

[9] ZHANG W, LI T. Exploring phase change memory and 3D die-stacking for power/thermal friendly, fast and durable memory architectures[C]// Proc of the 18th International Conference on Parallel Architectures and Compilation Techniques. Raleigh, 2009: 101-112.

[10] RAMOS L E, GORBATOV E, BIANCHINI R. Page placement in hybrid memory systems[C]// Proc of the International Conference on Supercomputing. New York, 2011: 85-95.

[11] QURESHI M K, FRANCESCHINI M M, LASTRAS-MONTANO L A, et al. Morphable memory system: A robust architecture for exploiting multi-level phase change memories[C]// Proc of the 37th Annual International Symposium on Computer Architecture. New York, 2010: 153-162.

[12] PARK H, YOO S, LEE S. Power management of hybrid DRAM/PRAM-based main memory[C]// Proc of the 48th Design Automation Conference. New York, 2011: 59-64.

[13] MLADENOV R. An efficient non-volatile main memory using phase change memory[C]// Proc of the 13th International Conference on Computer Systems and Technologies. New York, 2012: 45-51.

[14] LUO X H, SHU J W. Summary of Research for Erasure Code in Storage System[J]. Journal of Computer Research and Development, 2012, 49(1): 1-11. (in Chinese)
罗象宏,舒继武. 存储系统中的纠删码研究综述[J]. 计算机研究与发展,2012,49(1):1-11.

[15] REED I S, SOLOMON G. Polynomial codes over certain finite fields[J]. Journal of the Society for Industrial and Applied Mathematics, 1960, 8(2): 300-304.

[16] BLAUM M, BRADY J, BRUCK J, et al. EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures[J]. IEEE Trans on Computer, 1995, 44(2): 192-202.

[17] CORBETT P, ENGLISH B, GOEL A, et al. Row-diagonal redundant for double disk failure correction[C]// Proc of the 3rd USENIX Conf on File and Storage Technologies. Berkeley, CA: USENIX Association, 2004: 2-15.

[18] XU L, BRUCK J. X-code: MDS array codes with optimal encoding[J]. IEEE Trans on Information Theory, 1999, 45(1): 272-276.

[19] JIN C, JIANG H, FENG D, et al. P-code: A new RAID-6 code with optimal properties[C]// Proc of the 23rd Int Conf on Supercomputing. New York: ACM, 2009: 360-369.

[20] PATTERSON V, GIBSON V, KATZ R. A case for redundant arrays of inexpensive disks(RAID)[C]// Proc of the ACM SIGMOD International Conference on Management of Data. 1988: 109-116.