

一种多核 SoC 中基于 Cache 机制的存储结构设计

张多利, 张 宇, 宋宇鲲, 汪 健

(合肥工业大学 微电子设计研究所, 安徽 合肥 230009)

摘 要: 本文针对一种面向高密度计算的异构多核 SoC 系统, 提出了一种层次化的共享二级存储结构(L2-Cache), 以缓解系统数据处理速度与外部存储间的速度差异, 所设计的层次化存储结构提供对象数据缓存功能, 利用计数替换策略, 减少二级存储污染, 提高有效数据命中率; 在计算时间间隙实现数据准确预读取和 L2-主存同步操作, 增加有效存储带宽. 最终测试结果表明, 采用层次化存储结构的设计兼顾了不同访存比应用的数据访存特性, 平均访存性能提高 31.1%, 不同规模的矩阵运算最高获得 1.573 的加速比, 整体任务计算时间平均减少了 27.8%.

关键词: Cache; 多核系统; 数据预取; 缓存替换

中图分类号: TN402

文献标识码: A

文章编号: 1000-7180(2017)10-0026-06

A Storage Structure Design Based on the Cache Mechanism in a Multi-core SoC

ZHANG Duo-li, ZHANG Yu, SONG Yu-kun, WANG Jian

(Institute of VLSI Design, Hefei University of Technology, Hefei 230009, China)

Abstract: A hierarchical shared secondary storage (L2-Cache) structure for a heterogeneous multi-core SoC face to high density calculation is proposed in this paper to ease the speed difference between data processing and data storage. The hierarchical storage structure provides object data caching function and reduce the pollution of secondary storage using count replacement strategy to improve valid data hit rate. The structure can also increase the effective memory bandwidth by L2 - main memory synchronous operation and accurate prefetching in data computing interval time. Final test results show that, L2-Cache design can adapt for various data's feature of different applications. The new design increase the average fetch performance of about 31.1%, and gain 1.573 speed ratio highest of different sizes of matrix, and reduce the average computing time of about 27.8%.

Key words: cache; multi-core system; prefetch mechanisms; cache replacement

1 引言

多核处理器极大提高了计算机的计算能力, 同时对存储器带宽和访存策略提出了更高要求^[1-2]. 为了解决该问题, 除了不断改善存储器的访存性能以外, 最常用也是最有效的方法就是采用层次化存储结构(Cache)^[3-4].

Cache 的原理是根据时间局部性和空间局部性, 通过数据缓存重复利用经常发生事件, 使得 CPU 能高速获取需要的数据, 提升系统性能^[5]. Cache 基本参数中关联度与行长都会对命中率产生

影响, 为 Cache 改进设计提供基本思路. 替换算法同样会对 Cache 命中率产生重要影响, LRU 是常用的经典的 Cache 替换算法. 数据预取技术是另外一种提升 Cache 性能的办法, 可以解决“Cache 污染问题”, 作为一种辅助设计, 提供另一种 Cache 改进思路.

本文将 Cache 引入面向矩阵类运算的多核系统 HMCS, 通过分析该多核系统与现有经典 CPU 系统异同, 在 Cache 的映射方法、替换算法、数据预取三个方面进行优化, 提出了一种共享二级存储(L2)的 Cache 结构. 该结构充分发挥了多核系统的优势, 利

用计算时间间隙实现数据预取和 L2-主存同步操作,提高 L2 命中率,减少数据访存需要的片外存储带宽,提高了系统运算性能。

2 面向矩阵类运算的多核系统

面向矩阵类运算的多核系统 HMCS 集成了存储单元 (Memory unit, MU)、多个运算单元 (Computing unit, CU)、输入输出接口单元 (I/O)、主控制单元 (MCU) 等,单元之间通过片上网络 (NoC) 交叉互连,其结构图如图 1 所示。以一个任务的执行为例, I/O 请求导入任务配置信息以及源数据, MCU 接受请求并解析配置信息, 分配运算单元的算力, 下发运算操作命令控制 CU 执行相应操作, 同时发布存储控制命令操控 MU 存取任务源/结果数据, 任务运算完成后将结果数据回写到主存储器。其中, MU 接收 MCU 数据存储控制命令, 管理源数据、结果数据、运算中间结果数据的存取, 使用 DDR3 SDRAM 作为片外存储介质。单 CU 一次运算数据规模支持 $8 \times 64 \text{ bit} \sim 1\text{M} \times 64 \text{ bit}$, 不同 CU 具有本地存储空间不同, 以 $2 \text{ k} \times 64 \text{ bit}$ 、 $4 \text{ k} \times 64 \text{ bit}$ 的规模居多, 其本地存储空间视为一级存储 (L1-Cache)。

存储单元处理的连续数据在主存中以矩阵形式存在, 称为一级矩阵。一个一级矩阵可以切分成多个大小相同的二级矩阵, 分别给不同 CU 计算, 以此利用多核并行计算减少任务执行时间。单 CU 一次能处理一个二级矩阵数据, 将一个二级矩阵视为一个基本存储单位。假设有一批连续数据, 二级矩阵个数为 M , 参与运算的 CU 数目为 $N(1 \leq N \leq 8, \text{一般 } M = N \times C, C \text{ 为整数})$ 。那么第 1 个 CU 第一次请求第 1 个二级矩阵, 第 2 个 CU 第一次请求第 2 个

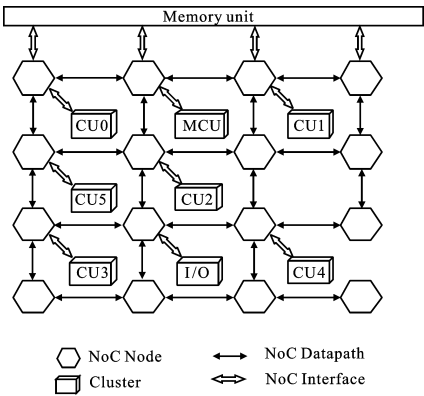


图 1 多核计算系统结构缩比图

二级矩阵, 第 N 个 CU 第一次请求第 N 个矩阵, 第 1 个 CU 第二次请求第 $N+1$ 个二级矩阵。因此, 根据每个 CU 请求次数、一级矩阵起始地址、地址跳变规律即可以计算出当前请求二级矩阵实际存储地址。

3 多核系统中共享二级存储的缓存结构

本文针对于 HMCS 多核系统提出的共享二级存储结构 (L2-Cache) 采取组相连的方式, 旁路式读出, 写回式写入, 其结构设计如图 2 所示。L2-Cache 的设计基于多通道随机分配的并行访存存储器接口 (RMAMI), 增加二级缓存以实现数据预取以及读写数据命中^[6]。L2-Cache 作为多核系统与片外主存 SDRAM 间的接口, 匹配两者间的带宽与数据格式。区别于传统缓存, L2-Cache 命中率并非由于局域性, 而是主要依靠计算时间间隙预读取以及中间结果数据写后读。

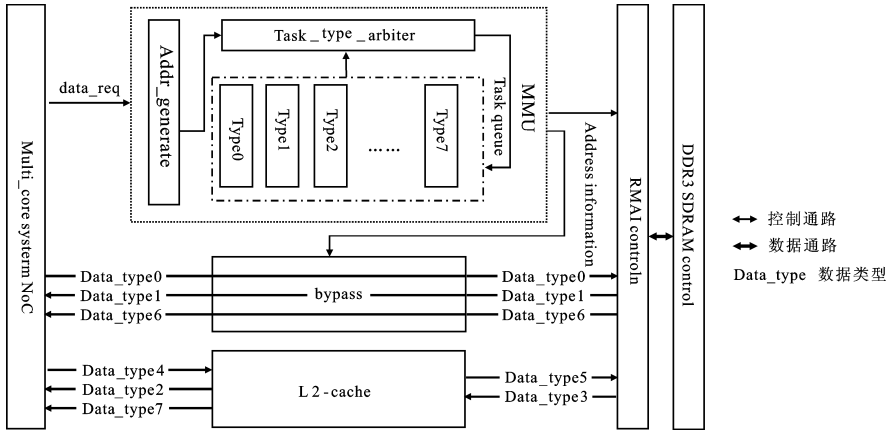


图 2 共享二级存储结构设计图

L2-Cache 中的存储空间有限,特殊运算如流运算的大批量数据不适合进入缓存中存储. 为了不影响这种运算的效率,保留主存与多核系统之间的数据交换通路,即旁路. 而旁路的存在允许 L2-Cache 采取旁路式读策略,此外,相较于经典缓存中的写入策略,可以采取写回与旁路相结合的方法.

该结构工作流程如图 3 所示. HMCS 在 MCU 的控制下,向 L2-Cache 结构发出读/写请求. 对于一个读请求,将该读请求同时送给 L2-Cache 和主存,若 L2-Cache 判断命中,切断对主存的访问;否则不做操作,多核系统直接访问主存. 对于一个写请求,首先判断 L2-Cache 写命中,若是则直接修改 L2-Cache,否则判断 L2-Cache 中是否还存在足够的未被污染的块,是则同样直接修改 L2-Cache,否则通过旁路写入主存. 在运算时间间隙,L2-Cache 未被读/写请求,将 L2-Cache 中污染的块写入主存,保证 Cache-主存一致性,其余时间均可分配给数据预取操作.

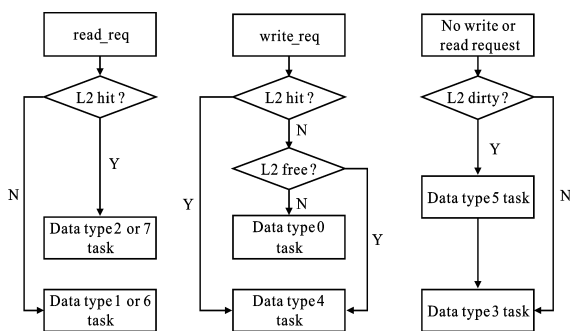


图 3 共享二级存储结构工作流程

3.1 映射方法

L2-Cache 是组相连的,这样既能灵活控制替换块的位置,又简化了设计难度. 对于一个存储控制命令来说, N 个源/目的 CU 有可能发出数据访存请求,但任一 CU 在当前请求未响应之前,不可能再次请求. 因此,L2-Cache 分为 8 组,第一组存储第 $8C$ ($C=0,1,2,\dots$) 个二级矩阵,第二组存储第 $8C+1$ ($C=0,1,2,\dots$) 个二级矩阵,以此类推,第八组存储第 $8C+7$ ($C=0,1,2,\dots$) 个二级矩阵. 由于 $N \leq 8$,可以有效避免多个 CU 请求命中同一组的情况.

与经典缓存不同,面向高密度计算的多核系统数据请求数据量大,数目不固定,且地址跳变不连续,因此,提出了一种基于对象存储的 L2-Cache 文件系统. 基于对象存储的文件系统 (Object-Based Storage File System, OBFS) 是针对对象存储体系结构设计的一种文件系统^[7]. 该文件系统针对集群

文件系统中文件读写的特点,把大文件和小文件分开存储于不同的“区域”中,每个区域由相同大小的块组成^[8]. L2-Cache 中的每一组都由相同大小的块组成,一个二级矩阵拆分成多个块数据存储在不同的块中. 例如,若一块数据大小为 $2\text{ k} \times 64\text{ bit}$,三个二级矩阵分别为 $1\text{ k} \times 64\text{ bit}$ 、 $4\text{ k} \times 64\text{ bit}$ 、 $5\text{ k} \times 64\text{ bit}$,三个二级矩阵请求分别需要 1 个、2 个、3 个块存储空间. 但需要注意的是,一个对象可能会占据多个块,其中一个块被其他对象占有,该对象的其余块都将失效.

3.2 准确地址预读取策略

数据预取技术有两个优化方向. 在基于历史的预取技术中,顺序预取在 Cache 发生失效时自动预取下一块;等跨度预取可以动态检测相邻访存地址的差值,发起跨度预取;Markov 预取技术动态的检测失效地址流中地址的相关性,捕获其中重复出现的地址序列,并为彼此相关的失效发起预取^[9-11]. 而基于执行的预取技术通过执行一段具体的代码对复杂的访问模式进行精确的预取,弥补了基于历史的预取技术在处理复杂访存模式方面的不足^[12].

上述多核系统中,存储单元根据接收到的数据存储控制命令,计算出当前请求二级矩阵的实际地址,假设为第 X 个二级矩阵. 那么, $X+N$ 个二级矩阵即为预取数据. 若在计算时间间隙能提前取出 $X+N$ 个二级矩阵,可以预见到,该数据在 L2-Cache 中将准确命中. 除了某些特殊请求不支持预读取,地址管理单元接收到的每一个数据请求,在发出当前请求地址的同时,都应伴随一个准确预读取地址. 该方法类似基于执行的预取技术,因为不同代码对应二级矩阵不同,但一段代码中预取地址的计算方法类似于基于历史的等跨度预取方法. 预读请求预判如图 4 所示. 如果预取请求的数据在 L2-cache 中命中,不需重复从主存中请求数据;若预取请求的数据在 L2-cache 中缺失但在预取任务队列中命中,不做操作;两者都不命中,新增一个预读请求,写入预取任务队列.

3.3 计数替换策略

LRU 是经典的 Cache 替换算法,但它也存在一次扫描应用污染 Cache、不同访问概率共用同一个页面等诸多缺点. LIRS 替换算法为解决 LRU 存在的问题,将所有一次访问的数据块和长时间未访问的数据块都设置成 HIR 状态,优先替换 HIR 的数据,但由于 HIR 集合太小,仅识别数据块的一次访问特性和时间

特性,会将近期要访问的数据块替换掉^[13-14]. BDP 算法针对数据块识别进行算法优化,预测未来访问模式^[15]. LIRS-A 算法针对 HIR 集合大小问题优化,自动调整 HIR 集合大小^[16]. 本设计中由于准确预取机制的存在,某个数据块未来是否会被访问,以及访问次数都是已知的. 若一个数据块已经识别出未来较

长一段时间内不会被访问,可以优先替换这部分数据块,保留未来访问次数不为 0 的数据. 根据以上分析,本文设计了一种计数替换策略,利用该多核系统中预取地址可见,根据命中次数排列 LRU 队列,标识数据块替换优先级,精确判断 L2-Cache 中数据块驻留时间.

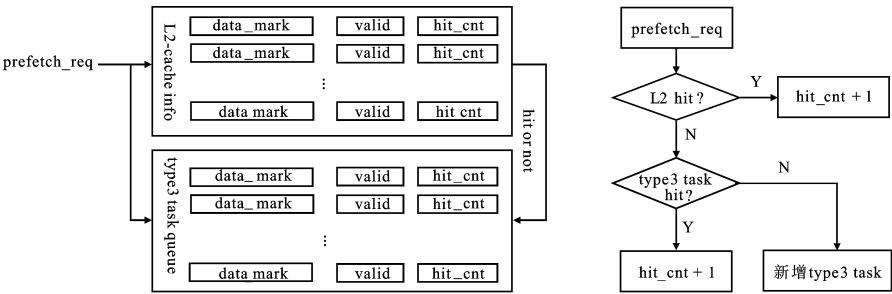


图 4 预取请求预判示意图

定义概念命中次数 hit_cnt:一批数据数据未来被重复请求的次数. 一批数据被预读请求命中,可以预测到该批数据即将被一个读请求命中,一个预读

请求对应一个读请求,若被预读请求命中 T 次,那么一定会被读出 T 次. L2 中数据失效伪代码如下:

```
If prefetch_addr hit // 预取请求命中
    data_valid 置 1;
else if request_addr hit 且 hit_cnt 为 0 // 读请求命中时 hit_cnt 为 0
    data_valid 置 0; // 数据失效,短时间内不会再被请求命中
If prefetch_addr hit 且 data_valid 为 1 // 预读请求命中
    hit_cnt + 1;
else if request_addr hit 且 hit_cnt 为 0 // 读请求命中时 hit_cnt 为 0
    hit_cnt 保持 0 不变;
else if request_addr hit // 读请求命中
    hit_cnt - 1;
```

4 性能评估

为验证多级存储优化设计的功能,并进行性能评估,本文在 Xilinx Virtex7 2000T FPGA 上搭建了集成 L2-Cache 的多核系统. 本文定义数据传输并行度 DTP——存储器接口支持并行传输数据通道的平均数目. 同一时间段内,同时传输数据的通道数目越多,多核并行计算度越高,相应的,任务执行的总体时间越少.

(1)资源消耗

本文设计的共享二级存储结构中,设置数据缓存中块大小为 $2\text{ k}\times 64\text{ bit}$,每组缓存包含 8 个读数据缓存块,4 个写数据缓存块. 旁路提供 6 路数据通道,数据缓存提供 8 路数据通道. 资源消耗情况如表 1 所示.

表 1 共享二级存储结构资源消耗

	Slice register	Slice LUTs	Block RAM
L2-cacheUtilization	56 067	60 658	66
Utilization/%	2. 27	4. 99	5. 12

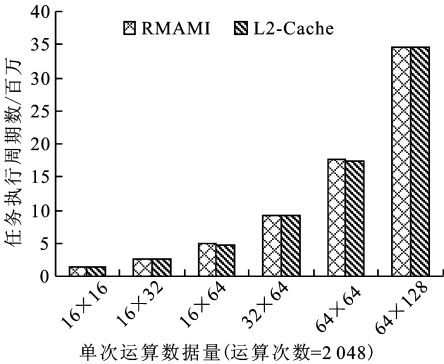
(2)共享二级存储结构性能评估

相较于主存与系统直接相连的情况,共享二级存储结构对存储性能的提升主要体现在两个方面. 一方面是利用计算时间间隙准确预取运算源数据以及同步二级存储中的脏块数据写入主存. 另一方面是二级存储中的脏块数据被其他任务读命中. 针对上述提升存储性能的两个方面,本文设计了两种类型的实验,验证共享二级存储性能提升幅度.

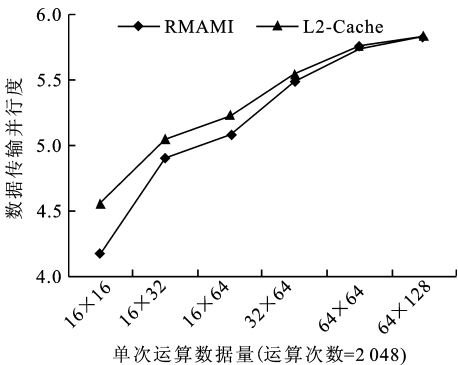
实验一包含实验 1. 1、1. 2 和 1. 3. 实验 1. 1 以仅执行数据访存操作的矩阵转置任务为对象;实验 1. 2 以计算访存比小于 1 的矩阵复数加任务为实验对象;实验 1. 3 则以计算访存比大于 1 的 FFT 运算

任务为对象. 三组实验执行任务均使用 8 个 CU 参与运算过程,不同运算量下的每个运算单元均运算 2 k 次. 实验二设计了一个任务流水线,执行不同数据量的矩阵乘运算,首先将其中一个矩阵转置,转置后的结果矩阵再与另外一个矩阵相乘. 同样使用了 8 个 CU 参与运算过程.

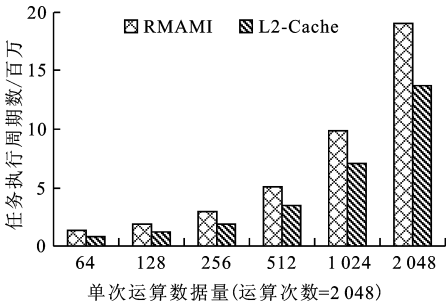
图 5 给出了实验 1. 1、1. 2、1. 3 的实验结果. 实验 1. 1 任务执行周期加速比和平均 DTP 加速比接近 1,这是由于该运算执行几乎没有计算时间间隙,共享二级存储结构能利用的空闲带宽很少,当 L2 被脏块数据占满后,仅存旁路可以数据交互,任务执行情况非常接近 RMAMI. 实验 1. 2 任务执行周期加速比最大 1. 573,平均任务执行周期加速比为 1. 479;数据传输并行度 DTP 最大 1. 558,平均 DTP 为 1. 476. 随着数据量的增加,L2-Cache 性能提升占比减少,加速比整体上呈下降趋势. 实验 1. 3 任务执行周期加速比最大 1. 449,平均任务执行周期加速比为 1. 247;数据传输并行度 DTP 最大 2. 049,平均 DTP 为 1. 969. 由于任务执行时,计算周期大于数据访存周期,对访存周期的优化在整体任务执行周期中占比并不明显,而 DTP 的优化效果正好相反,数据计算时间长,有足够的时间预取命中数据. 由图可知,共享二级缓存的存储性能优于旁路 RMAMI 的性能.



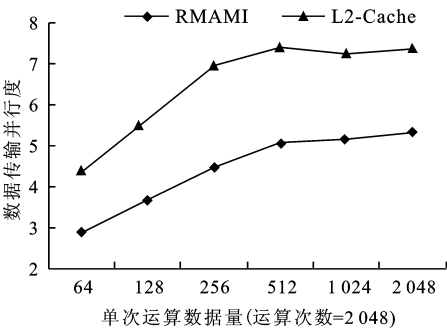
(a) 实验 1. 1 任务执行周期数



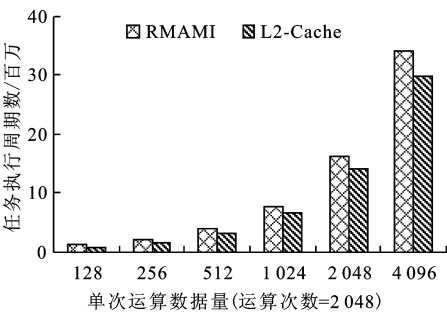
(b) 实验 1. 1 数据传输并行度



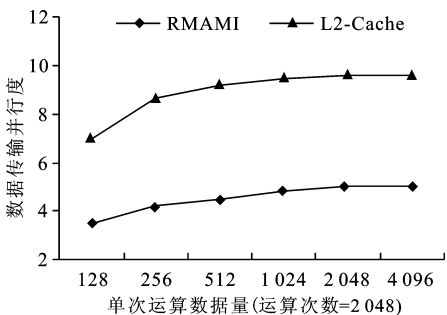
(c) 实验 1. 2 任务执行周期数



(d) 实验 1. 2 数据传输并行度



(e) 实验 1. 3 任务执行周期数



(f) 实验 1. 3 数据传输并行度

图 5 实验一测试结果

表 2 总结了不同数据量下 RMAMI 和 L2-Cache 的任务执行周期、数据传输并行度 DTP 以及它们各自的加速比. 任务执行周期加速比最高可达到 1. 178,DTP 加速比最大值为 1. 799. 当相乘矩阵

数据量达到 64 K 时,任务执行周期和数据传输并行度的加速比均接近于 1.

表 2 实验二测试结果

数据量	任务执行周期数		任务执行周期加速比	数据传输并行度		DTP 加速比
	RMAMI	L2-Cache		RMAMI	L2-Cache	
A:16 × 16 B:16 × 16	6 708	5 694	1.178	3.827	4.557	1.799
A:16 × 32 B:32 × 16	13 429	11 591	1.159	4.568	7.400	1.620
A:16 × 64 B:64 × 16	26 861	23 195	1.158	4.831	7.445	1.541
A:32 × 64 B:64 × 32	85 233	77 037	1.106	5.169	7.949	1.538
A:64 × 64 B:64 × 64	300 296	283 885	1.059	5.399	8.055	1.492
A:64 × 1 024 B:1 024 × 64	4 894 128	4 893 097	1.000	5.482	5.482	1.000

5 结束语

本文基于支持多通道随机分配的并行访存存储器接口(RMAMI),提出了一种针对面向高密度运算的多核系统所设计的共享二级存储结构,以缓解片上多核计算与片上访存之间的速度差异.该结构提供了与经典缓存类似的功能,在此基础上,实现数据准确预取,进一步提高缓存命中率,提高存储单元访存性能.最终实验结果表明,采用层次化存储结构的设计平均性能提高 31.1%,不同规模的矩阵运算最高获得 1.573 的加速比,整体任务计算时间平均减少了 27.8%.

参考文献:

[1] 董立平,周丽. 多核多线程是未来处理器的发展方向[J]. 高性能计算发展与应用,2008,6(3):29-32.

[2] 晏沛湘. 片上多核处理器二级 Cache 结构及资源管理技术研究[D]. 长沙:国防科学技术大学,2012.

[3] 王庆. 面向嵌入式多核系统的并行程序优化技术研究[D]. 哈尔滨:哈尔滨工业大学,2013.

[4] 肖瑞瑾. 多核处理器层次化存储体系研究[D]. 上海:复旦大学,2012.

[5] John L Hennessy, David A Patterson. Computer architecture: a quantitative approach[M]. 4th ed. Netherland,Elsevier Inc, 2007.

[6] 孙玲. 多通道并行访存接口设计与实现[D]. 合肥:合肥工业大学,2015.

[7] Wang Feng. Storage management in large distributed object-based storage system[J]. Journal of Parallel & Distributed Computing , 2009, 69(4):349-359

[8] 李柱. 分布式文件系统小文件性能优化技术研究与实践[D]. 长沙:国防科学技术大学,2008.

[9] Joseph D,Grünwald Dirk. Prefetching using markov

predictors [J]. IEEE Transactions on Computers, 1999, 48(2): 121-133.

[10] Vander S P,Lilja D J. Data prefetch mechanisms[J]. ACM Computing Surveys, 2000, 32(2): 174-199.

[11] 谢学军. 数据值预测与存储器预取技术研究[D]. 哈尔滨:哈尔滨工业大学,2006.

[12] 方娟,张红波. 一种改进的多核处理器硬件预取技术[J]. 计算机科学,2012,39(10):48-50.

[13] Lee D,Choi J,Kim J H,et al. LRFU: A spectrum of policies that subsumes the least recently used and least frequently used policies[J]. IEEE Trans on Computers,2001,50(12):1352-1360

[14] Megiddo N,Modha D. ARCl A self-tuning, low overhead replacement cache[C]//Proc of the 2nd USENIX Syrup on File and Storage Technologies. USA, San Francisco,2003:115-130

[15] 朱旭东,柯剑,那文武,等. 一种数据块关系指导的缓存替换方法[C]//第 15 届全国信息存储技术学术会议论文集. 北京,2008:8.

[16] 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, 2007,49(8):1331-1338.

作者简介:

张多利 男,(1976-),博士,研究员.研究方向为多核处理器体系结构与设计方法、多媒体专用集成电路设计实现等.

张宇(通讯作者) 女,(1994-),硕士研究生.研究方向为数字集成电路设计. E-mail:920337859@qq.com.

宋宇鲲 男,(1975-),博士,副研究员.研究方向为面向数据密集与计算机密集应用的 SoC/MPSoC 体系结构与实现.

汪健 女,(1969-),研究员级高级工程师.研究方向为 SoC 及大规模数字集成电路.