一种异构多核处理器的并行流存储结构

邓让钰,陈海燕,窦 强,徐炜遐,谢伦国,戴泽福,李永进,夏 军,罗 莉,张民选
(国防科学技术大学计算机学院,湖南长沙 410073)

摘 要: 异构多核处理器可结合多种处理器体系结构的优势,既保留传统通用体系结构的灵活性,又拥有大量计算资源,可提供更高的峰值计算性能. YHFT64-3 异构多核处理器中浮点处理部件 18 套,峰值计算能力强大,设计与之相匹配的存储系统是一项重大挑战. 针对 YHFT64-3 处理器,本文提出了一种并行流层次存储结构,深入阐述了如何体现应用特点、支持并行数据流处理的存储系统的设计思想和方法,从多个层次实现对并行数据流的挖掘或捕获. 测试结果表明,这种存储结构体现了应用特点,能够较好地发挥 YHFT64-3 处理器的性能,同频情况下(500MHz),YHFT64-3 比 YHFT64-2 性能高 2—3 个数量级,与 1.6GHz 的 Itanium2 性能相当,但代价更低.

关键词: 异构多核处理器; 流体系结构; 预取; 存储调度; 优化的锁步执行模型
中图分类号: TP302.1 **文献标识码**: A **文章编号**: 0372-2112 (2009) 02-0312-06

A Parallel Stream Memory Architecture for Heterogeneous Multi-core Processor

DENG Rang-yu, CHEN Hai-yan, DOU Qiang, XU Wei-xia, XIE Lun-guo,
DAI Ze-fu, LI Yong-jin, XIA Jun, LUO Li, ZHANG Min-xuan
(*School of Computer Science, National University of Defense Technology, Changsha, Hunan* 410073, *China*)

Abstract: Heterogeneous multi-core processor can integrate merits of many types of architecture, so it can achieve peak performance as high as processors with special architecture, and keep as flexible as traditional processors at the same time. It is challenging to design a memory sub-system to suit to YHFT64-3, a heterogeneous multi-core processor with 18 float function units. In this paper, a parallel stream memory sub-system architecture is presented for YHFT64-3, its design idea is described, and the principle to find and capture parallel data streams in several levels is detailed. Testing results show that the proposed application-specific memory sub-system can improve system performance significantly. The performance of 500 MHz YHFT64-3 is 2-3 order of that of YHFT64-2 with the same working frequency, and is close to that of 1.6GHz Itanium2 with less cost.

Key words: heterogeneous multi-core processor; stream architecture; prefetch; memory access schedule; OLSM (optimized look-step execution model)

1 引言

VLSI 工艺的发展为在单一芯片中集成大量计算资源成为可能. 例如,在 0.13nm 工艺下,一个 64 位 FPU 的面积约为 $1mm^2$,$14\times14mm^2$ 的芯片上可以集成上百个 64 位 FPU,提供强大的峰值计算能力,但大量的计算单元需要与之相匹配的存储系统.

媒体应用中,同一组数据的处理方式相似甚至相同,易于并行处理,可以充分利用大量计算资源. 针对这种特点,新型可编程处理器如 VIRAM[1]、IMAGINE 等设置了大量计算单元,并采用优化的高带宽存储系统以适应流应用的特点.

科学与工程应用领域较广,典型的如科学模型建模,涉及流体力学、气象、分子动力学、有限元方法等. 这些应用和媒体应用具有许多相似之处,两者都具有计算密集性、并行性、丰富的数据局部性等特点[2,3]. 尽管如此,科学与工程应用和媒体应用还是有很多不同之处. 媒体应用具有较为规整的计算结构,能够很容易地映射成流程序,在流处理器上获取很高的性能,而科学与工程应用的计算模式具有更高的复杂度和不规整性,存储带宽与处理器速度的匹配问题更加突出[4]. 计算模式的不规整性使得应用不能很好地映射成流程序,而大量计算单元需要更高的片外存储带宽. 这种差异决定了面向媒体应用的流体系结构不能很好地适用于科学与工程应用. 另外,传统处理器结构复杂,受多方面的影响,能够管理的计算资源非常少(浮点处理单元一般不超过 2

收稿日期:2007-07-16;修回日期:2008-08-21
基金项目:国家 863 高技术研究发展计划基金(No. 2005AA110020);国家自然科学基金(No. 60573173;No. 60703074)

个),这是影响其性能进一步提高的直接障碍.

因此,简单多核,尤其是结合传统通用体系结构和新型流处理器体系结构的异构多核处理器,并改革计算和编程模型可能是未来面向科学与工程应用的计算机体系结构发展的重要方向[5~7].探索将面向媒体计算的体系结构应用到科学与工程应用中是当前的研究热点之一[8].

2　微体系结构

YHFT64 - 3 微处理器体系结构如图 1 所示,它包含一个标量处理器核、一个流处理器核以及容量为 512 KB 的共享二级 Cache、DDR 存储控制器,并通过网络接口支持多处理器直接互连.

标量处理器核采用精简后的 YHFT64 - 1 通用微处理器[9].该核采用多发射结构,有 2 个浮点处理部件.它的主要工作除负责运行操作系统外,还参与基本运算操作,调度流处理器核的指令流和数据流,以及管理与片外网络的通信等任务.

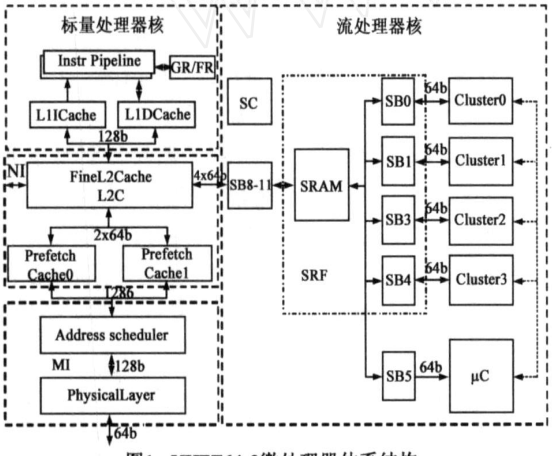

图1　YHFT64-3微处理器体系结构

流处理器核是处理器系统的加速部件,以 YHFT64-2 的核心部分为主体.它采用类似 Imagine 流处理器的体系结构[10],设置了 16 套计算单元,主要子部件有计算簇单元(cluster)、微控制器(μC,Micro Controller)、流寄存器文件(SRF,Stream Register File)和流控制部件(SC,Stream Controller).流处理器核负责执行由标量处理器发送的流指令,并对流数据进行所需的操作.

3　并行流存储结构

3.1　处理器核内的存储结构

标量处理器核内部的存储层次由通用寄存器文件和独立的一级指令和数据 Cache 构成,在结构和功能上与传统处理器内部存储层次相似.由于该标量处理器核采用了 OLSM 执行模式[9],能够体现延迟敏感型应用和带宽敏感型应用的特点,可以充分发挥科学和工程应用中的长流特点.

流处理器核采用流式存储层次结构,包括本地寄存器文件 (LRF,Local Register File)、SRF,能够体现流式应用的访存特点.LRF 采用分布式模式,直接为各计算簇提供数据,并存放运算簇内的临时数据.SRF 缓存计算簇所需要的输入流和计算输出流,是流处理器核内的主要数据存储器.

处理器核之间的数据一致性由软件实现,硬件只提供基本操作,如显式写回数据块.

3.2　共享的 L2 Cache

细粒度 L2 Cache 设计成由标量核与流处理器核共享的结构.标量核与流处理器核由于结构、数据处理方式不同,以及特定的应用模式,在访存行为和存储带宽上的要求存在较大的差异.一方面 L2Cache 能够较好地弥合这种差异;另一方面,共享结构不可避免地会引起不同处理器的存储数据相互替换的问题,因此正确分析两种处理器访存方式的特点,在结构设计上进行有针对性的调整,减轻两个处理器核访存要求的相互影响,才能充分发挥 L2Cache 的性能.标量处理器核负责任务调度工作,它的访存方式具有传统的空间和时间局部性的特点.流处理器核对数据的处理方式是:在一个 ALU 簇内完成针对一个数据记录的所有操作,然后再处理下一个记录.这种处理方式将导致处理器周期性、连续性地访问外部存储器,因此,存储系统的设计应该注重带宽的提高,减小失效开销,而不是缩短单次访存时间.

L2 Cache 将访问过程流水化,以较小的失效开销换取更高的时钟频率和数据吞吐率.L2 Cache 采用二路组相联结构,设置 64 字节的 Cache line,以获取更多的数据局部性.操作策略上采用了可配置的写回或写穿透策略,同时又针对应用的特点及流处理器核的双流缓冲结构采用了双流水线结构,可以挖掘可能的多条并行流,以提高整体性能.L2Cache 与标量核和流处理器核之间均使用独立的数据和请求总线连接,处理器发出的请求由请求队列统一接收,请求队列管理器根据请求的地址决定访问 0 号或 1 号 Cache 流水线.L2Cache 的结构如图 2 所示.

L2Cache 每条流水线将访问过程划分为四个流水站:请求接收站(RP,Request Prepare)、数据访问站(RA,RAM Access)、数据流控制站(FC,Flow Control)和存储器更新站(RU,RAM Update).结构如图 3 所示.

第一站 RP 负责接收处理器核的访存请求以及 Cache 内部流水线对 Cache RAM 数据的更新请求.多个请求发生冲突时,优先处理 L2Cache 的内部请求.第二站 RA 完成对数据和标记存储器的访问.由于存储器的访问时间相对较长,可根据不同的主频要求,这一站可

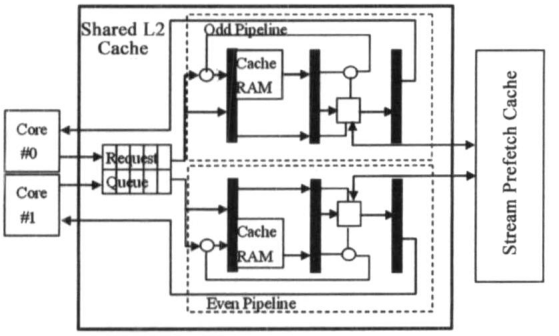

图2 L2 Cache结构

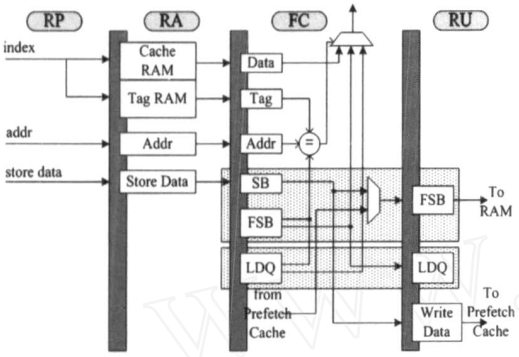

图3 L2Cache流水线

进一步划分成多站,即:通过采用具有多个读、写端口的存储器,以及与每个端口相对应的多相时钟信号发生器,可实现对 RAM 存储器的流水化访问,从而保证每个时钟周期都可进行新的数据访问. 第三站 FC 对每种数据流向进行控制,具体工作包括:命中判别、失效处理和数据输出. 处理器的请求地址与 Tag 标记进行比较,判断是否命中,请求不命中时,L2Cache 将该请求传递给流预取 Cache,流预取 Cache 返回的数据先写入填充数据缓冲 FSB 中,再由 FSB 更新数据体和标记. FSB 中的数据还需要传送到更新数据队列 LDQ 中,以解决内部流水线的数据相关性问题. 由于 FSB 和 LDQ 中都可能包含有效数据,因此,命中判别还需要包含与 FSB 和 LDQ 的比较. 命中或失效数据从下一级返回时,请求数据将立刻从 FC 流水站输出,以尽可能减小访存延时. 第四站 RU 负责更新 Cache 数据和标记 RAM. 如果更新请求与处理器访问请求相冲突,则优先处理更新请求.

3.3 流预取 Cache

预取缓冲的思想就是在处理器真正需要数据之前进行预取来有效提高 Cache 命中率,因而被广泛采用. 但如何在异构多核处理器系统中实现适当的预取策略却缺乏深入研究. 由于科学与工程应用具有特殊的数据局部性,设置流预取 Cache 可以更加有效地开发这种局部性,发掘潜在的流. 在每条 L2Cache 流水线的失效路径上设置一个独立的流预取 Cache,它根据失效请求的信息决定预取策略,保存预取数据,从而捕获更多的数据局部性. 预取 Cache 的结构如图 4 所示.

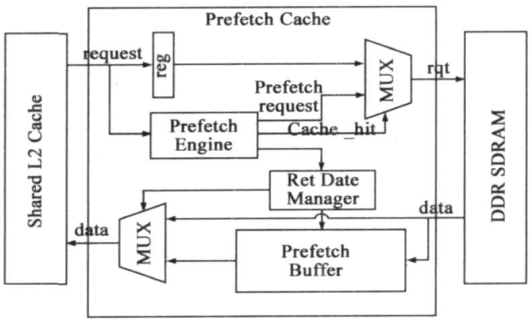

图4 预取Cache结构

首先,预取引擎(PE,Prefetch Engine)检查 L2Cache 的失效请求在预取 Cache 中的命中情况:当请求命中时,通知数据返回缓冲器(RDM,Return Data Manager)直接从预取缓冲区中读出数据,并返回给 L2Cache,阻止该请求向存储控制器传递;若不命中,PE 将请求发送至存储控制器的同时,根据设定的预取策略生成预取地址并发送预取请求. 然后,RDM 判断从下一级存储器返回的数据是否为预取数据,若为预取数据,则将数据存储到预取缓冲中,否则直接将数据返回给 L2Cache.

预取缓冲不命中时,将增加两个时钟周期的失效开销,而命中时可免去访问片外存储器的大量延时,因此,提高预取缓冲的命中率能够有效提高存储系统的性能. 预取缓冲的命中率大小跟多种因素相关,针对不同应用的访存方式的特殊性,仔细调整预取缓冲区的大小、预取策略和替换算法,能够以最小的代价获得最大的性能提升.

(1) 根据 BlueGene/L 的研究结果[3],对于科学计算和工程应用,当预取缓冲区容量大于 2KB 以后,新增加的存储容量对命中率已无多大贡献. 因此,实际设计中预取缓冲区的容量非常小,只有 1KB,这得益于支持存储调度的 DDR2 存储控制器.

(2) 高效的替换算法总是保留更为有用的数据,从而提高有效带宽. 传统的 LRU 替换算法并不一定能获得高性能,并且实现复杂度也较大,而相对简单的轮转替换算法更能体现流应用的特点,所以本设计中采用了这种算法.

(3) 预取策略与具体应用紧密相关,与应用不相适应的预取策略甚至可能降低存储系统性能. 在异构系统中,由于不同处理器的访存方式具有较为明显的差异,因此,对不同处理器的请求设置一个标志位,根据请求的不同采取相应的预取策略:对于流处理器的访存请求,可以采取乐观式的固定步长预取策略,并且进行连续预取,即对每个失效请求同时产生多个连续的预取请求,而对于标量处理器的访存请求,则采用流检

测器减少预取请求数量,甚至不预取.

相对于L2Cache,预取Cache具有结构简单、面积和功耗小、额外失效开销小等特点.

3.4 存储控制器调度策略

DDR2存储器和标准SDRAM一样,也采用多体技术(multibanking)和行缓冲(row buffer)技术,允许多个访存事务并发处理.如果同体不同行的串行访问不能够流水化,将导致极高的访问延迟;而不同体或者同体同行不同单元的访问可以利用多体并发和行缓冲技术,隐藏行选和行预充时间,从而实现较低的访问延迟.

DDR2控制器利用SDRAM的上述特性,采用了一种非最优搜索的启发式调度策略.访存调度问题归结为:给定 n 个访存请求 $M_1, \cdots, M_n$,在满足DDR2时序规范约束条件下,给出这 n 个访存请求的一个调度序列,以使其总执行时间最短.在调度策略确定的情况下,不同的体地址、行地址映射方式对不同应用的性能影响很大.

4 核心算法的测试与分析

在验证过程中,我们选取了科学与工程应用中的两个核心算法作为测试激励,它们是fft和gemm.根据实际结构的特点,两个算法的不同问题规模分别进行了多种优化,如减少访存操作、增加规则访问、减少Cache失效率等:

(1) fft1024:1024点fft;

(2) fft512:二维512点fft;

(3) fft512_O:优化后的二维512点fft;

(4) gemm256:采用分块算法的256x256矩阵乘;

(5) gemm1024:采用分块算法的1024x1024矩阵乘.

4.1 FFT算法

快速FFT算法是一个常用算法,属典型的数据访问局部性较差的运算密集型算法,运算过程包括三部分:系数计算、蝶型变换和位反操作.由于系数计算不能够通过编译直接流化,本次流化的部分只包括蝶型变换和位反操作两个部分.对于一维的 N 点FFT变换来讲,其理论计算访存比为:

$$\frac{N \times \log_2^N \times 12}{N} = 12 \times \log_2^N$$

可见,随着输入规模的增大其计算访存比呈对数增长.但是由于每计算一个结果都需要访问所有的输入数据,因此它对数据的访问局部性也随着输入规模的增大线性下降.FFT的算法可根据硬件资源的设置,针对不同的运算规模有不同的实现.

4.2 矩阵乘算法

矩阵乘算法的核心计算是求向量的内积 $A_{n \cdot m} \times B_{n \cdot m}$,至少需要 $O(m \cdot n^2)$ 个浮点操作,计算访存比为 $O(n)$,属于典型的数据局部性较好的计算密集型算法.

在Intel Itanium2处理器上,按传统策略组织数据,编译器能够较好地管理Cache,充分发挥两条浮点流水线的性能.

在YHFT64-3中,如果按照一般方式来组织数据流,即A按行组织,B按列组织,每次循环计算A的一行乘以B的一列,则内积操作分散在各个cluster上进行,会导致大量的簇间通信,而且流级的两重循环调用kernel的次数很多,存在大量重复从SRF向LRF传输数据的操作,另外每次循环A和B都要重复访存.为了提高内积操作的效率,将A的每一行和B的每一列作为一个记录,放在一个cluster中,使所有乘和加的计算都在一个运算簇本地完成.

4.3 算法测试与结果分析

从访存、计算过程来看,fft和gemm有很大的不同.fft算法的访存和计算不能够很好重叠,因此访存延迟难以隐藏,性能提升依赖于降低绝对访问延迟,而gemm算法通过分块,可以在计算的同时,取下一分块的数据,达到访存延迟隐藏的目的.

我们将上述两个核心算法在Itanium2(1.6GHz)上测试多遍,所得性能取平均值,其中对二维fft算法只测试了512点规模.YHFT64-2采用原型芯片作为测试平台(核心工作频率为500MHz),而YHFT64-3处理器则使用硬件描述语言模拟平台(模拟的核心频率为500MHz,DDR2存储控制器频率为333MHz).受宿主机的模拟速度限制,YHFT64-3上只测试了gemm256算法.

表1和表2是两个核心算法的测试结果,除列出了YHFT64-3在L2Cache、流预取Cache和存储调度都开启(正常配置)情况下的测试结果外,还列出了fft1024和gemm256在关闭L2Cache和流预取Cache下的测试结果.

表1 fft算法性能(单位:秒)

	fft1024	fft512	fft512_O
Intel Itanium2	0.00045763	——	0.000818
YHFT64-2	0.00599	5.74	0.0116
YHFT64-3 (正常配置)	0.000070776	0.067825664	0.000132472
YHFT64-3 (Cache off)	0.000806360	——	——

就所测试的问题规模而言,YHFT64-3中L2Cache对性能的影响非常显著,有10倍以上的性能提升.总体上看,YHFT64-3的性能与Itanium2(1.6G Hz)相当(Itanium2上的测试存在一定误差,与操作系统调度有关),比YHFT64-2快2～3个数量级.考虑到Itanium2的片内Cache容量远大于YHFT64-3,因此当问题规模增大时,

其优势将逐步得到发挥.

表 2 gemm 算法性能(单位:秒)

	256 ×256	512 ×512	1024 ×1024
Intel Itanium2	0.008	0.066	0.553
YHFT64-2	1.396	5.406	20.538
YHFT64-3 (正常配置)	0.008080962	0.031283775	——
YHFT64-3 (Cache off)	0.094478478	——	——

我们在 Cache 打开的情况下,进一步对 YHFT64-3 还进行了流预取 Cache 关/开结构调整情况下的性能测试,测试结果见表 3～5.

表 3 fft1024 的性能分析

	流预取关闭	流预取开
访存	39490ns	24718 ns
计算	23234ns	23234ns
写回结果	27888ns	21350ns
其它	1446ns	1474ns
总和	92058ns	70776ns
计算访存时间比	1∶1.7	1∶1.06
性能提升	23.1 %	

表 4 fft512_O 的性能分析

	流预取关闭	流预取开
时间总和	202698ns	132472ns
性能提升	34.6 %	

表 3 和表 4 表明,流预取 Cache 开/关只影响了访存时间,对计算时间基本没有影响.fft 算法中由于访存取数、计算和写结果是串行完成的,性能直接由花费最大的取数环节决定.在流预取 Cache 关/开前后,不考虑写结果的时间,计算/访存时间比从 1∶1.7 提升到 1∶1.06,访存性能得到了极大的改善,总体性能分别提升了 23.1 %和 34.6 %.

表 5 中,预取 Cache 关闭后,gemm256 的计算访存时间比接近 1∶3,而在打开流预取 Cache 后,数据准备阶段的计算访存时间比提升到 1∶2,数据准备阶段后的计算访存时间比甚至提升到 1∶0.65.将块处理过程流水化,隐藏了绝大部分访存时间,性能提升了 24.3 %.

表 5 gemm256 的性能分析

	流预取关闭	流预取开	
		数据准备阶段	数据就绪
分块的访存时间	5452ns	3630ns	1202ns
分块的计算时间	1856ns	1852ns	1860 ns
总和	10675890ns	8080962ns	
计算访存时间比	1∶2.94	1∶2	1∶0.65
性能提升	24.3 %		

5 结构性问题讨论

表 2 的测试结果表明 Itanium2 的浮点运算部件的利用率可达 80 %以上,而 YHFT64-3 的浮点运算部件的利用率约为 30 %.这暴露了 YHFT64-3 体系结构的不足之处.

另外,尽管 L2Cache 和预取 Cache 的存在极大地改善了访存特性,但我们还应该看到如下问题值得思考:

(1) 在 Cache 发生大规模块替换时,数据准备阶段和数据就绪阶段的计算访存时间比为 1∶3,有变坏的趋势,说明问题规模对 Cache 的容量比较敏感.

(2) 在 gemm256 算法中,第一轮子块的处理过程计算/访存时间约为 1∶2,流指令的通信延迟能被隐藏.第一轮以后的子块处理中,尽管计算/访存时间已经大于 1∶1,表明访存本可以完全被计算隐藏,但实际情况并非如此,分析原因是由于流指令发送时机太晚,说明此类应用中流指令极易成为影响性能的瓶颈,减少流指令将成为未来衡量流程序质量的一个重要指标.

(3) DDR2 控制器可能将大量的连续地址映射到同体中,而不同体的请求优先,这导致调度策略、地址映射方式与流应用的访存模式不甚协调,会损失一部分性能.

(4) 观察表明,L2Cache 中两条流水线的利用率很低,请求通常只发送到一条流水线,算法、编译还不能充分体现 L2Cache 的结构.

(5) 标量处理器核比流处理器核更依赖 L2Cache,而后者更依赖流预取 Cache.但 L2Cache 的存在缓解了计算密集、数据局部性差的应用对存储带宽的需求压力,且变相提高了流的长度,因此 L2Cache 对流处理器核也是很重要的.测试结果也表明,没有 L2Cache 的 YHFT64-2 的性能比 YHFT64-3 相差甚远.

6 结论

本文针对 YHFT64-3 异构多核处理器的应用特点,提出了一种并行流层次存储结构,强调了存储系统对并行数据流的多层次挖掘作用.测试结果表明,这种面向特定应用的并行流存储结构能使处理器以较低代价获得高性能,同频情况下(500MHz),YHFT64-3 比 YHFT64-2 性能高 2～3 个数量级,与 1.6GHz 的 Itanium2 性能相当,但代价更低.

尽管测试结果表明,YHFT64 - 3 在某些科学与工程应用方面具有一定的优势,但也存在计算部件利用率低的问题,计算/通信模式和存储有待进一步改进.目前,该设计已经成功投片,我们将在原型芯片系统上进一步对各种问题规模和计算/访存特性的科学与工程应用进行深入测试分析.

种种迹象表明,类似 YHFT64-3 的异构多核处理器有望成为未来某类科学与工程应用的加速器,但还有很多问题需要解决.

参考文献:

[1] Christoforos Kozyrakis David Patterson. Scalable vector processors for embedded systems[J]. IEEE Micro, 2003, 23(6): 36 - 45.

[2] Junhee Lee, Chanik Park, Soonhoi Ha. Memory access pattern analysis and stream cache design for multimedia applications [A]. Asia and South Pacific Design Automation Conference [C]. New York: ACM, 2003. 22 - 27.

[3] Jose R. Brunheroto etc. Data cache prefetching design space exploration for BlueGene/L supercomputer[A]. SBAC-PAD '05 [C]. Washington, DC: IEEE Computer Society ?, 2005. 201 - 208.

[4] J Weinberg, M o Mcracken, A Snavely, E Strohmaierm. Quantifying locality in the memory access patterns of HPC applications[A]. SC2005[C]. Washington, DC: IEEE Computer Society, 2005. 50.

[5] John McCalpin, Chuck Moore, Phil Hester. The role of multicore processors in the evolution of general-purpose computing [J]. CTWatch, 2007, 3(1): 18 - 30.

[6] DAVE Turek. High performance computing and the implications of multi-core architecture[J]. CTWatch, 2007, 3(1): 31 - 33.

[7] Jack Dongarra, Dennis Gannon, Geoffrey Fox, Ken Kennedy. The impact of multi-core on computational science software [J]. CTWatch, 2007, 3(1): 3 - 10.

[8] James Irwin, Simon McIntosh-Smith. The best of both world: delivering aggregated performance for high-performance math libraries[Z]. ISC2007.

[9] 邓让钰,陈海燕,邢座程,谢伦国,曾献君. EPIC 微体系结构的存储级并行执行模型的研究[J]. 计算机学报, 2007, 20(1): 74 - 80.
Deng Rangyu, Chen Haiyan, Xing Zuocheng, Xie Lunguo, Zeng Xianjun. The research on memory-level parallelism execution model in EPIC architecture[J]. Chinese Journal of Computers, 2007, 20(1): 74 - 80. (in Chinese)

[10] Yan Xuejun, Yan Xiaobo, Xing Zuocheng, Deng Yu, Jiang Jiang and Zhang Ying. A 64-bit stream processor architecture for scientific applications[A]. ISCA2007[C]. New York: ACM, 2007. 210 - 21.

作者简介:

邓让钰　男,1972 年生于湖南新田,国防科技大学计算机学院副教授,主要研究方向为高性能计算机、微处理器体系结构和集成电路设计. **通信作者**. E-mail: rydeng@21cn.com

陈海燕　女,1967 年生于四川南充. 国防科技大学计算机学院副研究员,主要研究方向为微处理器体系结构及 VLSI 设计.

窦　强　男,1973 年生于湖南长沙,国防科技大学计算机学院副教授,主要研究方向为高性能计算机体系结构.