维普资讯 http://www.cqvip.com

多核多线程处理器存储技术研究进展*)

屈文新　樊晓桠　张盛兵
(西北工业大学航空微电子中心　西安 710072)

摘　要　多核多线程技术已经成为微处理器发展的趋势,使用多核多线程技术可以使微处理器的性能得到极大的提高,但同时也对存储系统提出了更高的要求。而相对增长的存储器访问延迟已经成为影响多核多线程处理器性能进一步提高的重要因素。本文首先介绍了当前常见的几种多核多线程处理器的结构,然后介绍了目前多核多线程处理器存储系统的研究现状,在此基础上讨论了当前多核多线程处理器存储系统研究的热点,并对多核多线程处理器存储系统技术的发展趋势进行了展望。

关键词　多核多线程,存储系统,访问延迟

Research Progress in Memory Technique of the Multi-Core and Multi-Thread Processor

QU Wen-Xin　FAN Xiao-Ya　ZHANG Sheng-Bing
(Aviation Microelectronic Center, Northwestern Polytechnical University, Xi'an 710072)

Abstract　Multi-core and multi-thread technique has become the trend of the research of the micropressor. Multi-core and multi-thread technique can improve the performance of the micropressor. But in the same time, the memory system has more pressure. The latence of access the memory has become the obstacle to improve the performance of the micro-processor. This paper firstly discusses some of the common architechture of the multi-core and multi-thread processor. And secondly explains today the research of the multi-core and multi-thread processor. Finally, at the base of above, the possible diretions of the memory techniques of the multi-core and multi-thread processor are stated.

Keywords　Multi-core, Multi-thread, Memory

随着集成电路制造工艺水平的迅猛提高和计算机微体系结构设计技术的发展和创新,微处理器的发展已经迈入线程级并行的时代。线程级并行将会是下一代高性能处理器的核心技术,多线程体系结构有望在未来10年中占支配地位[1]。

在2001年MTEAC(Workshop on Multithreaded Execution, Architecture and Compilation)会议上,Intel微处理器研究实验室的研究员John P. Shen给出了微系统结构发展的趋势展望,如图1所示。

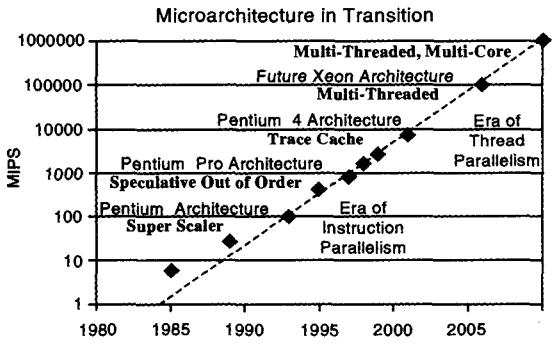

图1　微系统结构的发展趋势

本文第1部分介绍了几种常见的多核多线程处理器结构;第2部分讨论了多核多线程处理器存储系统的研究现状;第3部分讨论了当前多核多线程处理器存储系统研究的热点;最后给出了结论。

1　多核多线程处理器

在很多主流商业应用程序中,如在线数据库事务处理(Online Database Transaction)与网络服务(如Web服务器),这些程序一般都具有较高的线程级并行性(TLP, Thread Level Parallelism)。为此,研究人员曾经提出过两种不同于传统的单核处理器的结构:单片多处理器(Chip Multiprocessor, CMP)与同时多线程处理器(Simultaneous Multithreading, SMT),这两种处理器结构可以充分利用这些应用的指令级并行性和线程级并行性,从而显著提高了这些应用的性能[2]。

CMP是指由在单个芯片上的多个处理器核所构成的多处理器系统。CMP允许线程在多个处理器核上并行执行,从而利用线程级并行性来提高系统性能。CMP的优势主要是处理器核可以很简单,易于获得较高的主频,同时缩短了设计和验证时间。CMP存在的主要问题是由于单片多处理器系统的资源是采用划分方式的,当没有足够的线程时,资源就浪费了[3]。

SMT结构的基本思想是:在一个时钟周期内发射多个线程的指令到功能部件上执行,SMT的优势是结合了超标量和多线程处理器的特点。SMT允许在一个时钟周期内执行来自不同线程的多条指令,因此在一个时钟周期内,SMT能够同时利用程序的线程级并行和指令级并行来消除水平浪费。SMT允许多个活动线程的组合来发射指令,当由于长延迟操

*)国家自然科学基金(60573143)(60573107)资助。**屈文新**　博士生,主要从事计算机体系结构的研究;**樊晓桠**　教授,主要从事计算机体系结构的研究和VLSI系统设计研究工作;**张盛兵**　教授,主要从事计算机系统结构、SOC设计与测试、嵌入式系统设计等方面研究。

维普资讯 http://www.cqvip.com

作或者资源冲突只有一个活动线程时，该线程能够使用所有可获得的发射槽来消除垂直浪费。SMT 存在的主要问题是使得指令发射阶段变得比较复杂；随着指令发射宽度的增加，模块和电路的延迟越来越大，不易获得较高的主频；另外，多个线程共享同一个一级 cache、TLB 和分支预测逻辑，将会导致冲突，使得 cache 不命中和分支预测错误率增加，限制了整个处理器性能的提升[4]。

多核多线程处理器是通过支持单片多处理器(CMP)和同时多线程(SMT)的组合来实现的。多核多线程处理器由多个简单的同时多线程处理器核构成，它提供了一种更加简单有效的方法去提高集成度。它不同于超标量处理器通过硬件来提取指令级的并行，是通过编译器的支持，多核多线程处理器可以提供一种线程级的并行。由于它由多个简单的同时多线程处理器，所以它就可以拥有单片多处理器主频高、设计和验证时间短的优势，又拥有同时多线程资源利用率高的优势，从而大大提高程序的运行效率。目前，越来越多的芯片生产商和研究机构都将注意力放在了多核多线程处理器的研究上[5]。

2005 年 SUN 公司提出了片上多线程技术(Chip Multiple Threading，CMT)，即将 CMP 技术和 SMT 技术结合起来。SUN 公司同时宣称将在其新一代处理器 Niagara 中使用这种新的多核多线程技术。Niagara 处理器主要包含了 8 个处理器核，每个处理器核可以同时运行 4 个线程[6]。

Sun 公司的多核多线程处理器设计转向反映了整个业界的发展趋势[7]。IBM 于 2000 发布了双核的 POWER-4[8] 处理器并且在 2005 年发布了它的第二代多核多线程处理器 POWER-5。

AMD、富士通和 Intel 都已发布了双核片上多处理器[9~11]。虽然高性能多核多线程设计技术一开始是限制在高端服务器处理器上，但是目前多核多线程设计技术已经开始应用到通用处理器的设计上了[12]。

随着生产厂家已经开始更多地关注多核计划而不是刚开始的双核计划，多核多线程在通用处理器设计方面已经开始引领潮流了。在不久的将来，多核多线程处理器将占领整个计算机领域，从笔记本电脑和台式电脑到服务器和超级计算机[12]。

2 多核多线程处理器存储系统研究现状

提高存储系统性能的问题早已经成为提高处理器性能进一步提高的主要障碍，学术界和工业界对此都极为关注。由于多核多线程技术是最近才提出来的新技术，所以相关的研究还较少。但是，多核多线程技术是通过将 CMP(片上多处理)和 SMT(同时多线程)技术的组合来实现的，因此现有的一些解决 CMP 和 SMT 存储访问延迟的技术也可以应用到多核多线程处理器的设计上去。

目前，国际上在多核多线程系统研究领域有影响的研究单位主要包括 SUN 研究院[7]、美国的 Stanford[3]、California 大学[13,14]、Pennsylvania 大学[5]、North Carolina 大学[15]、加拿大的 Toronto 大学[16]、西班牙的 Catalonia 科技大学[17]等。

2005 年，SUN 最新的多核多线程 SPARC 处理器(内核叫做 Niagara)已经在实验室中设计成功[7]。Niagara 有 8 个处理器内核，每个内核都是一个四路并发多线程处理器，都有自己的一级 cache。所有的 8 个内核共享 3MB 的 12 路二级 cache。Niagara 是第三代多核多线程处理器，它的整体设计包括内核都专门为多核多线程设计进行了优化。因为 Niagara 的目标是使用在具有高线程级并行(TLP)、低指令级并行(ILP)的应用程序和大工作集负载的商业服务器应用程序中[6]。

Standford 大学在 1996 年首次提出了基于 CMP 结构的 Hydra 处理器。Hydra 集成了 4 个 MIPS R10000 微处理器核[3]。处理器核间采用私有一级 cache，共享 2 级 cache。片外的 3 级 cache 和主存使用一条 128 位的总线连接。Hydra 使用了一种单片共享 cache 的结构来提高多处理器之间的通信带宽，降低通信延迟。相对于基于总线互联的多处理器体系结构来说，这种基于 cache 的互联充分地利用了片上 cache 高带宽的优势来提高片上的多个处理器之间的通信速度和降低同步延迟。这将提高并行处理器的效率，从而在多处理器上为串行程序更灵活地开发细粒度的并行处理提供支持。Hydra 的一级 cache 采用哈佛结构，指令和数据 cache 的大小各 16kB，4 路组相连，通过 64 位的数据线与处理器相连。

California 大学在 Tullsen 的带领下也展开了一系列 SMT 项目的研究，提出了基于线程的多路径执行、选择性值预测、SMT 中动态线程调度等思想。最近，他们还对单指令集异构 Multi-Core 结构进行了研究[13,14]。与传统的 CMP 相比，该结构中的多个处理器可以是不同大小、结构和性能的，但是必须是使用同样的指令系统，以便根据应用的需要，把不同的任务划分到不同的处理器上执行。

Pennsylvania 大学对 CMP 系统中 L2 cache 的组织结构进行了大量的研究。对于在 CMP 系统多个处理器核私有 L2 cache 和共享 L2 cache 的性能进行了研究[5]。通过大量的仿真实验，发现在共享 L2 cache 的情况下，多个处理器核对 L2 cache 的访问请求并不平衡，而在私有 L2 cache 的情况下多个处理器核对 L2 cache 的访问请求又会存在大量的冲突。所以，他们提出了一种被称作为基于处理器核划分的共享二级 cache 结构。这种结构拥有共享 L2 cache 和私有 L2 cache 的优点，同时避免了共享 L2 cache 和私有 L2 cache 的缺点。

North Carolina 大学对共享 L2 cache 的 CMP 系统的 cache 性能进行了深入的研究[15]，研究了 L2 cache 中的内容对内部线程的影响。通过对不同的共享 L2 cache 的 CMP 结构的模型进行仿真，提出了不同的共享 L2 cache 结构对不同的线程的影响也是不一样的，不同的 L2 cache 调度策略对线程的影响也非常大。他们还提出了两个基于启发式的评价模型和一个分析模型，使用这些模型来预测共享 L2 cache 结构的 CMP 系统对内部线程执行的影响。通过大量的研究，在一个双核的 CMP 系统中使用一个共享的 512kB 的 8 路组相连 L2 cache 的 Prob 模型可以提供准确度达到 96.1%的预测。

加拿大的 Toronto 大学对在 SMT 系统中提高单线程情况下 cache 的命中率进行了研究。提出了通过使用辅助线程进行指令预取的方法来提高指令预取准确率，从而提高了 cache 的命中率，提高了 SMT 系统的资源利用率。之后，他们提出了直接预执行和有限状态机重调度的方法来提高单线程情况下，提高指令预取准确率[16]。

西班牙的 Catalonia 科技大学(the Technical University of Catalonia)的高性能计算组在 SMT 的指令预测、取指单元等方面进行了一系列的研究[4]。改进了 Tullsen 提出的 SMT 模型，为每个线程设置了一个取指目标缓冲(FTB－Fetch Target Buffer)，代替原来的 BTB，因此可以提前形成取指流，

维普资讯 http://www.cqvip.com

提高了取指带宽的利用率和 cache 命中率。

其它关于多核多线程处理器 cache 结构的研究还有很多。例如法国的 IRISA 研究中心展开了同时多线程的研究[18]；日本的 Takanobu 大学在进行多线程处理器二进制代码转换的研究；日本的 Kyushu 大学和 Tsukuba 大学研究 FUCE 处理器[19]，目标是组织合理的 cache 结构，使得处理器内部执行和处理器间的通信相结合。

国内对线程级并行的研究中，国防科技大学进行了一系列关于多线程技术的研究[20]，所提出的前瞻性多线程系统结构 SMA 有机地结合前瞻性执行和多线程技术。SMA 将程序分为一系列线程，线程间可以存在数据和控制相关。SMA 以线程为步长，前瞻性执行程序。处理器每次预测将要执行的线程，将该线程分配给一个逻辑处理器执行。通过将指令的执行提前到其控制相关性消除之前，来获得更高的 ILP。同时，SMA 采用多线程来隐藏访存操作的长时延。处理器通过组合每个线程的指令窗口形成一个大的动态指令窗口，从而使得处理器中功能单元的利用率得到很大的提高。与同时多线程 SMT 类似，SMA 处理器中所有的线程共享取指单元、运算单元，从而能够灵活地使用程序中的线程级并行和指令级并行。SMA 模型综合了前瞻性执行和多线程的优点。通过前瞻性执行线程来实现深度的前瞻执行，将每个线程的指令窗口组合成一个大的指令窗口。另一方面，SMA 通过线程切换有效地隐藏访存操作的长时延，提高资源的利用率。

中国科技大学提出了一种用 cache 存放线程现场的多线程实现技术[21]，其核心思想是改造多机系统中的 cache 系统，将其分割成两个部分：一部分仍用作原来意义上的 cache，另一部分则用来存放数个线程的现场。当然，用 cache 存放现场，速度可能比不上专门设计的多线程处理器芯片。不过研究表明，在线程个数为 2～4 之间时，系统的性能还是有很大的提高。

中科院计算所进行了低功耗 SMT 体系结构的研究[22,23]，围绕高性能、低功耗这两个目标讨论和探究 SMT 体系结构的基本思想、设计技术、低功耗考虑以及编译器和操作系统设计应注意和对待的新问题。多线程处理器在很多低功耗和功率受限设备的设计和应用中具有很大的吸引力。首先，它通过多线程提供了额外的并行性，使得处理器并不太依赖于大量的指令预测，因此它在预测和从不提交指令上消耗很小。其次，当运行多线程时，由于它较高而且更多的并行性，使得没有被充分利用的资源的能耗也减少了。另外，多线程结构允许线程选择机制和指令动态分配机制有更大的设计调整和权衡空间，以适应降低功耗的需求目标。

西北工业大学提出了双路径执行的置信度估计算法 DCR 算法，提出了线程预构的思想，将取指部件制约 SMT 资源利用率的两个主要因素分开来考虑[24]。线程预构的思想就是根据预测的控制流，提前从指令 cache 取指，为每个线程形成顺序的指令流。在基本 SMT 结构的基础上，增设一个线程预构 cache (TPC)，指令从指令 cache 中取出来，经过译码以后，加入到线程预构 cache，然后再从线程预构 cache 中取出，经过寄存器重命名以后，进入指令队列(整数指令队列或浮点指令队列)。引入线程预构的思想之后，将线程取指调度的目标分解为两个子目标，采用两级调度策略，分别针对不同的目标，采用不同的线程调度策略。从 cache 取指到线程预构 cache，主要针对取指带宽的利用问题；从线程预构 cache 到指令队列主要是考虑减少指令队列阻塞。

3 多核多线程处理器存储系统的研究热点

使用多核多线程技术可以使处理器的性能得到了极大的提高，但同时也对存储系统提出了更高的要求。而相对增长的存储器访问延迟必将成为影响多核多线程处理器性能进一步提高的重要因素，因此存储系统已经成为进一步提高多核多线程处理器性能的一个瓶颈。

目前研究多核多线程处理器存储系统的热点主要有：通过分析多核多线程处理器的动态指令流特性，研究多核多线程处理器结构的线程调度策略和指令调度策略；多核多线程处理器指令预取技术的研究；L1 cache、L2 cache、专用 cache 技术研究；多核多线程处理器存储系统仿真模型的研究等。

1)在多核多线程处理器中，将多个线程分配给多个同时多线程处理器核，在任一时钟周期都允许来自不同线程的指令共享所有的处理器资源，能够在指令级并行和线程级并行之间做互补和折衷，充分地利用处理器的资源，提高处理器的吞吐量。然而，线程的数目可能有很大的差别，有时线程数目多，有时线程数目少。选用恰当的线程调度策略，使得无论线程数目多少，也无论线程是来自多个不同的应用程序还是一个应用程序的不同程序段时，都能够获得很高的资源利用效率，是多核多线程处理器研究的主要内容之一。

对于多核多线程处理器中的每个同时多线程处理器核，为了增加指令执行的并行性，提高资源的利用率和吞吐率，必然要引入分支预测技术和乱序发射的推测执行技术。采用分支预测、推测发射执行的方法，必然就可能将错误路径的指令取进到指令队列，也可能会将与执行周期长的指令或者引起数据 cache 缺失的取数指令数据相关的后续指令取进到指令队列而引起指令队列阻塞，这些都会导致处理器资源利用率下降。

因此，第一要减少错误路径指令取指，分支预测是必不可少的，关键是减少错误路径的取指。一方面要提高分支预测的准确率，另一方面在分支预测错误时，能暂停该线程的取指。理想情况是，每次预测不准确时，都暂停该线程取指。第二要减少指令队列阻塞，指令之间的数据相关或者功能单元的冲突，都可能导致进入指令队列的指令无法发射到功能单元。关键是不能让指令长时间占用指令队列资源，而且不能简单地增加指令队列大小(因为增加指令队列大小，会显著增加发射逻辑的复杂度，降低处理器的主频)。只能通过合理选择指令进入指令队列，使得指令进入指令队列能尽快地发射出去。

2)在超标量处理器中比较适用的预取策略在多核多线程处理器中不一定适用。要想达到超标量处理器中预取的准确率，必须结合多核多线程处理器的特点，研究新的适合于多核多线程处理器结构的预取策略。

对各种预取策略进行分析，发现大部分误预取都是由使用的历史信息不合适所致。当某个转移指令需要某种历史信息(全局或者局部)，而预取器提供的是不同类型的历史信息时，就会引起历史信息的不恰当使用，使得预取的结果不准确。在多核多线程处理器中，由于多个线程的乱序交叉，使得历史信息可能更混乱，因此新的预取策略中应采用一种新的记录和使用历史信息的方法。为了能够更好地适应不同转移指令的特点，把全局历史信息和局部历史信息结合起来进行预测是一种有效的方法。

3)cache 的引入是存储系统中最基本也是最有效的解决

维普资讯 http://www.cqvip.com

这个问题的方法。cache 利用指令、数据流的空间局部性和时间局部性来提高系统性能。提高 cache 效率最直接的方法是增加 cache 的容量，然而片内 cache 容量的增加受到制造工艺的限制。为此，人们提出各种方法来提高 cache 的效率，例如改进 cache 替换算法、减少 cache 命中时间等。

cache 预取也是提高 cache 效率的有效方法。无论是指令预取还是数据预取、软件预取还是硬件预取，都是通过提前发出存储器访问请求，从而提高了 cache 的命中率。目前，已经有许多预取策略被用来提高 cache 的效率。然而，大部分预取策略都属于单核单线程结构 cache 预取，对于多核多线程结构的 cache 预取策略研究还较少。其中历史预测表结构是一种有效提高 cache 命中率的方法。对于多核多线程结构，需要设计一种全新结构的历史预测表。

cache 在 CMT 系统占用和消耗了相当大的空间和能量。可以使用多级的 cache 去最大限度地开发程序的局部性原理。如何组织这些多级的 cache，使它们发挥最大的效率，是一个重要的问题。

L1 cache 是离处理器数据通路最近的，处理了大多数的处理器请求，它的访问时间是最短的。将 L1 cache 做得更大，并且有多个端口使得所有的处理器都能够访问它，并不能有效地减少访问延迟。因此，私有小的 L1 cache 是一种有效的做法。私有的意思是处理器核不需要通过内部总线就可以访问到它所分配 L1 cache 中的数据。

4)对于 L2 cache，通常有两种结构，私有 L2 cache 和共享 L2 cache。

对于私有 L2 cache，将 L2 cache 的空间平均分配给每一个处理器。每一个处理器核访问自己的 L2 cache 空间不需要通过内部总线。对于 L2 cache 不命中，处理器核要通过内部总线访问其他的处理器 L2 cache 中是否有所要访问的数据。如果没有，再访问片外存储器。

对于共享 L2 cache，共享 L1 cache 和 L2 cache 之间的总线，在 L1 cache 之间要保证一致性。L1 cache 不命中，查找 L2 cahce。如果 L2 cache 也不命中，就要访问片外存储器。

这两种方法都有利有弊。在私有 L2 cache 时，cache 单元离处理器很近，在大多数情况下不需要请求总线访问，减少了访问延迟和总线争夺。它也有不利的方面：首先，一个数据块在多个处理器的 L2 cache 单元存在副本，这就减少了整个芯片上 cache 中所能保存的数据；另一个不利的方面是，预先给每个处理器划分 L2 cache 的大小，会导致 Load 不平衡问题。例如：一个处理器核的 L2 cache 利用率非常高，而另一个处理器核的 L2 cache 利用率非常低。共享的 L2 cache 方案中就不存在这些问题，共享的 L2 cache 可以看作是一个大的 cache，其中不存在数据块的副本，同样也不存在 Load 不平衡问题。而共享 L2 cache 的结构的缺点是：当 L1 cache 不命中时，处理器通过内部总线访问 L2 cache 存在较大的延迟和总线争用。

传统的共享 L2 cache 结构是将 L2 cache 作为一个整体。这样，每次 cache 访问的动态功耗就非常大。通常，解决这个问题的方法是将 L2 cache 分成多个独立编址块，这些块进行交叉编址，我们称这种结构为交叉编址的共享 L2 cache 结构。

总的来讲，对于多核多线程处理器，私有 L1 cache 是通常的做法。而对于 L2 cache 而言，有两种做法：私有 L2 cache 和共享 L2 cache。现在大多数研究表明，共享比私有好。而共享又有多种做法，其中整个 L2 cache 作为一块共享，将 L2 cache 分成多个小块，将块分配给多个处理器核来使用，使用一种表的结构来记录每个处理器对 L2 cache 块的使用情况，动态地进行分配。

5)在多核多线程处理器中，为了提高处理器资源利用率，减少访存压力，需要设置一些专用 cache。例如在多核多线程处理器中引入 Trace cache 的思想，将指令流(物理上)不连续的块转变为连续的块，从而加快指令的预取，为多核多线程处理器的指令译码提供更大的指令流量，减少存储层次的带宽压力，加快译码的速度。使用转移地址 cache 来提高指令 cache 的预测命中率。引入线程预构的思想，设计线程预构 cache，不但可以实现线程中指令的复用，缩短取指的延迟，还可以减少对指令 cache 取指带宽的要求，降低指令 cache 的复杂度。

6)对多核多线程处理器动态指令流特性分析、线程调度策略、指令调度策略、预取策略研究、cache 结构的研究，都要采用多核多线程仿真器对 benchmarks 程序进行仿真，通过仿真结果对提出的策略进行分析和评估，以便进行优化。目前有关多核多线程处理器研究的文章中，绝大多数研究者是通过修改 SimpleScalar 仿真器搭建一个多核多线程的处理器来进行仿真，运行 SPEC 程序来验证和评测他们的设计方案。

结论 使用多核多线程技术可以使处理器的性能得到极大的提高，但同时也对存储系统提出了更高的要求。而相对增长的存储器访问延迟已经成为影响多核多线程处理器性能进一步提高的重要因素。存储系统瓶颈问题已经成为提高微处理器性能的主要障碍，学术界和工业界对此都进行了大量的研究。使用恰当的指令调度策略、线程调度策略、组织合理的多级 cache 结构，使它们发挥最大的效率，从而提高多核多线程处理器存储系统的效率，提高整个多核多线程处理器的性能。

参 考 文 献

1 Sohi G, Roth A. Speculative Multithreaded Processors. IEEE Computer, 2001, 34(4): 66～71

2 Chaudhry S, Caprioli P, Yip S, et al. High-performance throughput computing. IEEE Micro, 2005, 25(3):32～45

3 Hammond L, Hubbert B A, Siu M, et al. The Stanford Hydra CMP. IEEE Micro, 2000, 20(2):71～84

4 Fal con A, Santana O J, Ram yrez A, et al. Tolerating branch predictor latency on SMT. In: Proceedings of the 5th International Symposium on High Performance Computing(ISHPC-V), 2003. 86～98

5 Liu C, Sivasubramaniam A, Kandemir M. Organizing the last line of defense before hitting the memory wall for CMPs. In: Proceedings High Performance Computer Architecture, 2004. HPCA-10. 10th International Symposium, 2004. 176～185

6 Kongetira P. A 32-way Multithreaded SPARC Processor. In Hot Chips 16. http://www. hotchips. org/archive/,2004

7 Spracklen L, Abraham S G. Chip Multithreading: Opportunities and Challenges. High-Performance Computer Architecture, 2005. HPCA-11. In: 11th International Symposium on, Feb. 2005. 248～252

8 Moore C. POWER4 System Microarchitecture. In: Microprocessor Forum, 2000

9 Advanced micro Devices. AMD Demonstrates Dula Core leadership. http://www. amd. com/,2004

10 Maruyama T. SPARC64 VI: Fujitsu's Next Generation Processor. In: Mictrprocessor Forum, 2003

11 Mcnairy C, Bhatia R. Montecito -the Next Product in the Itanium Processor Family. In Hot Chips 16. http://www. hotchips. org/archive/,2004

12 Otellini P. Intel Keynote. In: Intel Developer Forum, 2004

(下转第 23 页)

维普资讯 http://www.cqvip.com

28 Gilbert A C, Levchenko K. Compressing network graphs. LINKKDD, 2004
29 Feder T, Motwani R. Clique partitions, graph compression and speeding-up algorithms. Journal of Computer And System Sciences, 1995,51:261～272
30 Adler M, Mitzenmacher M. Towards compressing web graphs. In: Proceedings of the IEEE Data Compression Conference, 2001. 203～212
31 Gansner E, Koren Y, North S. Topological Fisheye Views for Visualizing Large Graphs. IEEE Transactions on Visualization and Computer Graphics, 2005,11(4):457～468
32 Teoh Soon Tee, Zhang Ke, Tseng Shih-Ming, et al. Combining Visual and Automated Data Mining for Near-Real-Time Anomaly Detection and Analysis in BGP. VizSEC/DMSEC'04, October 29, 2004
33 Erbacher R F, Walker K L, Fincke D A. Intrusion and misuse Detection in Large-Scale System. IEEE Computer Graphics and Applications, 2002,22(1):38～48
34 Nyarko K, Capers T, Scott C, et al. Network Intrusion Visualization with NIVA, an Intrusion Detection Visual Analyzer with Haptic Integration. In: Proceeding of the 10th Symp, (HAPTICS'02), 2002
35 Ishmael J, Race N J P. Visawin: Visualizing a Wireless Network. 0-7803-8255-2/04 IEEE, 2004
36 Erbacher R F, Christensen K, Sundberg A. Designing Visualization Capabilities for IDS Challenges
37 Freeman L. Visualizing Social Networks. Journal of Social Structure, 2000(1)
38 Heer J, Boyd D. Vizster: Visualiziing Online Social Networks, 2005
39 Liben-Nowell D, Kleinberg J. The Link Prediction Problem for Social Networks. In: Proceedings of the Twelfth Annual ACM International Conference on Information and Knowledge Management(CIKM'03), November 2003. 556～559
40 Chen Hsinchun, Atabakhsh H, Tseng Chunju, et al. Visualization in Law Enforcenment. CHI, Apr. 2005
41 eNetViz- Gene Network Visualization. http://genetviz.sourceforge.net/, 2005
42 Ahmad S, Gromiha M M. Prediction of DNA Binding in Proteins from Composition, Sequence and Structure. Genome Informatics, 2002, 13: 308～309
43 Sharan R, Maron-Katz A, Shamir R. CLICK and EXPANDER: a system for clustering and visualizing gene expression data. Bioinformatics, 2003, 19(14)
44 Barnes J E, Hut P. A hierarchical o(n log n)force calculation algorithm. Nature, 1986,324(4):446～449
45 Quigley A, Eades P. Graph drawing, clustering, and visual abstraction. In: Marks J, ed. Proc. 8th Int Symp Graph Drawing, Springer LNCS 1984, 2000. 197～210
46 Newman M E J. Fast algorithm for detecting community structure in networks. Phys Rev E, 2004 (in press)
47 Ghoniem M, Fekete J D, Castagliola P. A comparison of the readability of graphs using node-link and matrix-based representations. In: INFOVIS'04: Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'04) (Washington, DC, USA), IEEE Computer Society, 2004. 17～24
48 Purchase H C, Cohen R F, James M I. An experimental study of the basis for graph drawing algorithms. J Exp Algorithmics, 1997, 2(4)
49 Brandes U, Eiglsperger M, Herman I, et al. GraphML Progress Report: Structural Layer Proposal. In: Proc. 9th Intl Symp Graph Drawing (GD'01), LNCS 2265, Springer-Verlag, 2002. 501～512
50 Wu A Y, Michael G, Han Jiawei. Mining scale-free networks using geodesic clustering. KDD, 2004. 719～724

(上接第 16 页)

13 Kumar R, Tullsen D M, Ranganathan P, et al. Single-ISA Heterogeneous Multi-Core Architectures for Multithreaded Workload Performance. In: 31st International Symposium on Computer Architecture, June 2004
14 Kumar R, Farkas K, Jouppi N P, et al. Single-ISA Heterogeneous Multi-Core Architectures: The Potential for Processor Power Reduction. In: 36th International Symposium on Microarchitecture, December 2003
15 Chandra D, Guo Fei, Kim S, et al. Predicting inter-thread cache contention on a chip multi-processor architecture. High-Performance Computer Architecture, 2005. HPCA-11. In: 11th International Symposium on, Feb. 2005. 340～351
16 Aamodt T M, Chow P, Hammarlund P, et al. Hardware Support for Prescient Instruction Prefetch. In: Proceedings High Performance Computer Architecture, 2004. HPCA-10. 10th International Symposium on, Feb. 2004. 84～84
18 Hily S, Seznec A. Branch prediction and simultaneous multithreading. In: Proceedings of Parallel Architectures and Compilation Techniques (PACT'96), Boston, October 1996
19 Matsuzaki T, Tomiyasu H, Amamiya M. Basic Mechanisms of Thread Control for On-Chip-Memory Multi-threading Processor. In: Proceedings of Workshop on Multithreaded Execution, Architecture and Compilation, Austin, Texas, December 2001
20 黄光奇,周兴铭. 单芯片多处理器的性能优势. 计算机工程与科学,2001,23(1):35～38
21 袁俊立,高原,陈新,等. 一种新的多线程实现技术. 小型微型计算机系统,1998,19(11):7～11
22 赵荣彩,唐志敏. 低功耗 SMT 体系结构研究. 计算机工程与设计,2002,23(8):7～12
23 赵荣彩,唐志敏,张兆庆,等. 编译指导的多线程低功耗技术研究. 计算机研究与发展,2002,39(12):1572～1579
24 李瑛. 同时多线程结构指令流特性及取指技术研究:[博士论文]. 西北工业大学,2004
25 胡伟武,唐志敏. 龙芯 1 号处理器结构设计. 计算机学报,2003,26(4):385～396
26 Peir J K, Lai S C, lu S L. Bloom Filtering cache Misses for Accurate Data Speculation and Prefetching. ICS'02, June 22-26, 2002, New York, USA. ACM. 1-58113-483-5, 2002
27 Li Yingmin, Brooks D, Hu Zhigang, et al. Performance, Energy, and Thermal Considerations for SMT and CMP Architectures. In: Proceedings of the 11th Int'l Symposium on High-Performance Computer Architecture, 2005
28 Kalla R, Sinharoy B, Tendler J. IBM POWER5 chip: a dual-core multithreaded processor. IEEE Micro, 2004,24(2):40～47
29 Tremblay M, et al. The MAJC Architecture: A synthesis of Parallelism and Scalability. IEEE Micro, 2000,20(6):12～25
30 Tremblay M, et al. MAJC-5200: A VLIW Convergent MPSOC. In: Mictoprocessor Forum 1999, 1999
31 Kapil S. UltraSPARC Gemini: Dual CPU Processor. in Hot Chips 15. http://www.hotchips.org/archive/, 2003
32 Jacobson Q. UltraSPARC IV Processors. In: Microprocessor Forum 2003, 2003
33 Wong W A, Baer J-L. Modified LRU Policies for Improving Second level cache Behavior. In: 6th International Symposium on High-Performance Computer Architecture, January 2000. 49～60
34 Srinivasan V. Hardware solutions to reduce effective memory access time: [PhD thesis]. Department of Computer Science and Engineering, University of Michigan, 2001
35 Wong W A, Baer Jean-Loup, The Impact of Timeliness for Hardware-based Prefetching from Main Memory: [Technical Report]. Department of Computer Science and Engineering, University of Washington, 2002
36 Marty M R, Bingham J D, Hill M D, et al. Improving multiple-CMP systems using token coherence. High-Performance Computer Architecture, 2005. HPCA-11. In: 11th International Symposium on, Feb. 2005. 328～339
37 Hennessy J L, Patterson D A. Computer Architecture: A Quantitative Approach. 3nd. San Mateo: Morgan Kaufmann Publishers, 2002
38 Hao E, Chang P-Y, Patt Y N. The Effect of Speculatively Updating Branch History on Branch Prediction Accuracy. In: Proc. 27th Int'l Symp on Microarchitecture(MICRO-27), Nov. 1994
39 Hammond L, Nayfeh B, Olukotun K. A single-Chip Multiprocessor. IEEE Computer, September 1997, 30(9):79～85
40 Haungs M, Sallee P, Farrens M. Branch transition rate: A new metric for improved branch classification analysis. In: Proc. HPCA, 2000. 241～250
41 He Li-Qiang, Liu Zhi-Yong. An Effective Instruction Fetch Policy for Simultaneous Multithreaded Processors. In: Proceedings of the 17th International Conference on High Performance Computing and Grid in Asia Pacific Region(HPCAsia'04), 2004
42 Hennessy J L, Patterson D A. Computer Architecture - a Quantitative Approach. second edition. Morgan Kaufmann Publishers Inc, 1996
43 Heil T, Smith J. Selective Dual Path Execution: [Technical Report]. ECE Department, University of Wiscosin-Madison. November 1996
44 Hammond B A, Hubbert M, Siu M K, et al. The Stanford Hydra CMP. IEEE Micro, 2000, 20(2): 71～84