ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.4, April 2008, pp.1069−1080
DOI: 10.3724/SP.J.1001.2008.01069

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种分片式多核处理器的用户级模拟器*

黄 琨[1,2+], 马 可[3], 曾洪博[1,2], 张 戈[1], 章隆兵[1]

[1](中国科学院 计算技术研究所 系统结构重点实验室,北京 100080)

[2](中国科学院 研究生院,北京 100049)

[3](中国科学技术大学 计算机科学与技术系,安徽 合肥 230027)

A Use-Level Simulator for Tiled Chip Multiprocessor

HUANG Kun[1,2+], MA Ke[3], ZENG Hong-Bo[1,2], ZHANG Ge[1], ZHANG Long-Bing[1]

[1](Key Laboratory of Computer System and Architecture, Institute of Computing Technology, The Chinese Academy of Sciences, Beijing 100080, China)

[2](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

[3](Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)

+ Corresponding author: Phn: +86-10-62600851, E-mail: huangkun@ict.ac.cn

Huang K, Ma K, Zeng HB, Zhang G, Zhang LB. A use-level simulator for tiled chip multiprocessor. *Journal of Software*, 2008,19(4):1069−1080. http://www.jos.org.cn/1000-9825/19/1069.htm

Abstract: As the transistor resources and delay of interconnect wires increase, the tiled multi-core processor has been a new direction for multi-core processor. In order to thoroughly study new type processor and explore the design space of it, this paper designs and implements a user-level performance simulator for the tiled CMP architecture. The simulator adopts the directory-based Cache Coherence Protocol and the architecture of store-and-forward Network- on-Chip with Godson-2 CPU as the processing core model, and depicts out-of-order transacted requests and responses and conflictions of requests and their timing characteristics in detail. The simulator can be used to evaluate all kinds of important performance features of the tiled CMP (chip multiprocessor) architecture by running all kinds of sequential or parallel workloads, and thus provides a fast, flexible and efficient platform for architecture design of multi-core processor.

Key words: tiled CMP (chip multiprocessor); simulator; network on chip; performance analysis; Godson-2 processor

摘 要: 随着片上晶体管资源的增多和互连线延迟的加大,分片式多核微处理器已成为多核处理器设计的新方

* Supported by the National Natural Science Foundation of China under Grant No.60673146 (国家自然科学基金); the National Natural Foundation of China for Distinguished Young Scholars under Grant No.60325205 (国家杰出青年基金); the National High-Tech Research and Development Plan of China under Grant No.2006AA010201 (国家高技术研究发展计划(863)); the National Basic Research Program of China under Grant No.2005CB321600 (国家重点基础研究发展计划(973)); the Beijing Natural Science Foundation of China under Grant No.4072024 (北京市自然科学基金); the Knowledge Innovation Program of the Institute of Computing Technology, the Chinese Academy of Sciences under Grant No.20066012 (中国科学院计算技术研究所知识创新课题)

Received 2007-02-05; Accepted 2007-05-24

向.为了对这种新型处理器进行体系结构的深入研究和设计空间的探索,设计并实现了针对分片式多核处理器的用户级多核性能模拟器.该多核模拟器在龙芯 2 号单处理器核的基础上,完整地模拟了基于目录的 Cache 一致性协议和存储转发式片上互联网络的结构模型,详细地刻画了由于系统乱序处理各种请求应答和请求之间的冲突而造成的时序特性,可以通过运行各种串行或并行的工作负载对多核处理器的各种重要性能指标加以评估,为多核处理器的结构设计提供了快速、灵活、高效的研究平台.

关键词: 分片式 CMP(chip multiprocessor);模拟器;片上网络;性能分析;龙芯 2 号微处理器

中图法分类号: TP302 **文献标识码:** A

片上多核微处理器(single-chip multiprocessor,简称CMP)的结构设计能够有效地利用日益增多的片上晶体管资源,它通过在单芯片上使用多个处理器核运行并发的线程来充分利用程序的并行性,可在不提高频率和功耗的基础上显著提高处理器的性能.由于受到频率和功耗的限制,工业界相继将高性能处理器的设计方向转向片上多核的处理器设计[1−3].

现有的CMP设计大多数采用集中式的多核处理器结构,在这种结构设计中,每个处理器核拥有私有的一级Cache,多个处理器核通过交叉开关或共享总线共享二级Cache,其优点在于结构设计比较简单,容易实现;缺点在于处理器核与二级Cache的数据交换是集中式的,容易成为瓶颈,造成数据通路的阻塞[4].在未来的CMP设计中,随着处理器核数目、二级Cache容量以及片上连线延迟的增大,这种集中式共享的多核结构将不再适用,而必须采用新型的多核结构.以RAW(the raw architecture workstation)[5]和TRIPS(the tera-op,realible,intelligently adaptive processing system)[6]为代表的分片式CMP结构可以很好地解决上述问题.分片式CMP的结构如图 1 所示(P为包含一级Cache的处理器核,*L2* 为二级Cache块,R为片上互联网络中的Router,该图中的片上互联网络为二维Mesh的结构).

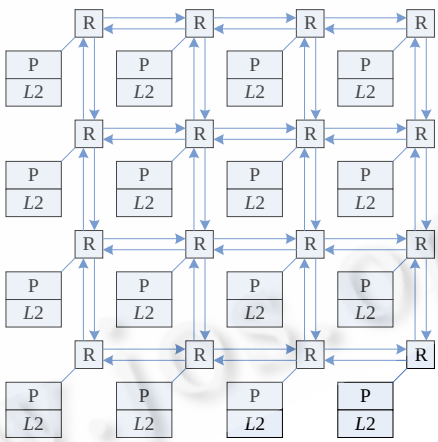


Fig.1 Architecture of tiled CMP

图 1 分片式 CMP 的结构

分片式 CMP 的结构特点如下:

(1) 各处理器核与二级Cache块均匀分布在芯片内,各二级Cache块物理上分布但逻辑上共享.每个处理器核都可以直接访问离自己最近的本地二级Cache块,除此之外,处理器核若要访问其他远程的二级Cache块都需要通过片上互联网络来访问.因此,各处理器核访问二级Cache的延时并不相同,访问本地的二级Cache块比访问远程的二级Cache块要快很多.这种二级Cache的组织结构被称为NUCA(non-uniform cache access)结构[7];

(2) 核间一般采用基于目录的 Cache 一致性协议,目录与二级 Cache 块放在一起,表示该二级 Cache 块被包含在哪些处理器核的一级 Cache 中;

(3) 拥有高效的片上互联网络和核间通信机制.片上网络通常采用二维 Mesh 或二维 Torus 等拓扑结构,这种拓扑结构比较简单,容易达到低延迟和高带宽的要求,能够快速、有效地在分布式的各处理器核和二级 Cache 块之间传输数据.

为了对这种新型的CMP结构进行有效的性能分析和设计空间的探索,我们需要快速而有效的性能评估工具对分片式CMP的各项性能指标进行评估.在这方面,结构级的灵活、快速、高效的多核性能模拟器无疑起着极其重要的作用,它使得处理器设计者在早期就能够对各种结构的处理器原型进行性能上的评估,从而在针对多核的相对庞大的探索空间中寻找适合于自己设计目标的处理器体系结构.本文以龙芯 2 号处理器[8]和对应的Sim-Godson模拟器[9]为基础,实现了针对分片式CMP的多核性能模拟器.该多核模拟器将Sim-Godson作为分片式CMP的单元处理器核模型,在此基础上,详细地模拟了片上互联网络和Cache一致性协议的时序特性,并通过模拟同步操作的执行算法为并行测试程序提供了运行环境上的支持.我们能够通过在模拟器上运行各种串行和并行的负载程序来考察目标应用程序在模拟器上展现的行为特征,从而可以对分片式CMP进行性能分析和设计空间的探索.

该模拟器的主要特点如下:

(1) 完整地模拟了Cache一致性协议和存储转发式片上互联网络的行为特性和结构模型.该模拟器详细地模拟和刻画了由于系统乱序地处理Cache一致性协议而引起的各种请求应答和由于请求之间的冲突而造成的时序特性和结构特征.该模拟器可以用作分片式多核处理器的前期性能分析和评价工具,也可为未来的龙芯 3 号多核处理器[8]的结构设计提供依据,并兼具部分设计验证的作用,具有重要的应用价值.

(2) 灵活性与扩展性好.该模拟器采用标准 C 设计,模拟器在结构上支持任意多核的扩展.模拟器的片上网络实现了多种常用的结构模型,如拓扑结构采用扩展性较好的二维 Mesh 和二维 Torus 网络结构,Router 采用简单的 XY 和 TXY 路由算法以及 Round Robin 和 Wave Front 的仲裁策略,使用基于 Credit 的流控策略,结构灵活性较高.另外,我们可以基于该模拟器方便地集成各种结构级功耗模拟的功能.该模拟器的一个设计特色是通过对每个处理器核的调度采取时钟触发的事件驱动方式,使得处理器的每个核可以运行在不同的时钟频率上,从而可以评价不同的多核功耗管理策略,如 DVFS(dynamic voltage and frequency scaling)等对多核处理器性能造成的影响.

(3) 执行速度快.该模拟器在设计上采取功能和时序分离的执行驱动方式,其中,功能部分负责保证数据运算结果的正确性,而时序部分则只需模拟 Cache 一致性协议和片上网络延迟等,因此,设计非常简单、高效.另外,该模拟器不需要运行操作系统,对系统调用直接采用本机模拟.因此,相对于详细模拟片上多核处理器结构的其他模拟器来说,本模拟器执行速度较快.据评估,本模拟器的运行速度可达 200K~300K 指令/秒.

本文第 1 节是相关工作介绍,简要介绍国际上已有的多核模拟器,并比较它们与本文描述的多核模拟器的不同点.第 2 节介绍模拟器中多核模型的实现,分别对模拟器中的处理器核模型、Cache 一致性时序模型和片上网络的模拟进行阐述.第 3 节介绍多核模拟器的运行环境.第 4 节给出模拟器的评测数据.最后是结论和未来的工作.

1 相关工作

目前,学术界已有一些模拟器可以用于多核微处理器结构的研究.Wisconsin大学开发的GEMS(general execution-driven multiprocessor simulator)[10]模拟器是一个全系统的模拟器,它借助商业化的Simics虚拟机[11]在全系统的环境下负责解释程序的执行.另外,提供了OPAL和RUBY两个时序模拟的模块,其中,OPAL负责乱序处理器的时序模拟,RUBY负责存储层次和网络互联的模拟,为了用户方便地更改和配置模拟器,GEMS专门提供了一种名为SLICC(specification language for implementing cache coherence)的脚本语言来描述Cache一致性协议.但是,由于该模拟器需要运行操作系统来做全系统模拟,因此模拟速度较低,而且由于使用了Simics作为模拟器的功能模拟的基础,使用者需要详细了解Simics提供的用户接口函数,这对于使用者灵活地使用和更改模拟器是不利的.

而另一个模拟器SESC(SuperESCalar simulator)[12]的目标是模拟目前大多数的处理器结构,包括单核乱序处理器结构、多线程处理器结构、CMP、PIM(processor in memory)以及线程级猜测(thread level speculation,简称TLS)等多种体系结构.该模拟器能够有效地模拟多线程并行的结构和线程级猜测的结构,但是对于CMP结构及其片上互联网络的模拟比较简单,不太适用于模拟分片式CMP及其片上互联网络的结构.

另外,在互联网络的模拟方面,以往关于片上互联网络的研究[13,14]一般都只是单独研究互联网络在随机数据的激励下的网络性能,而没有将片上互联网络放在多核处理器这个大的评估环境中去考虑.事实上,在多核处理器的互联网络上传递的是处理器核之间的访存请求及数据响应以及一致性请求及其响应,这些网络上传输的信号显然与以往研究的随机激励不同;而大多数多核模拟器对于片上互联网络的模拟又太过简单,只是粗略地模拟总线或交叉开关等网络类型,这种网络类型只适用于集中式的体系结构,而不适用于分片式CMP的NUCA这种分布式的体系结构.本文详细地模拟了片上互联网络接受和转发处理器核之间的各种访存和一致性请求及它们的响应情况,从而可以对分片式CMP的网络通信情况进行正确的评估.

2 模拟器中多核模型的实现

模拟器的构建应该遵从准确和快速的目标,但是这两个目标实际上是互相矛盾的,我们需要在这两个目标之间巧妙地进行权衡.为了兼顾上述提到的两个目标,本文的模拟器采用了功能和时序分离的执行驱动方式:模拟器的功能部分使用独立的指令执行引擎来解释和执行指令,指令的执行效果更新处理器的状态;而模拟器的时序部分只维护指令在流水线中流动的时序,不关心实际的数据.采用这种执行驱动方式的好处是:一方面使得模拟器的基本数据处理只在功能部分中涉及,时序部分中只有控制通路,没有数据通路,不需要在流水线中将数据按级往下传递,可以减少大量的数据复制和传递的操作,从而提高模拟器的运行速度;另一方面,若用户要实现其他体系结构,只需要修改模拟器的时序部分,而无须与具体的数据通路打交道,这样做易于修改和调试,为进行体系结构的评价提供了较高的灵活性.

另外,本模拟器不运行操作系统,直接将目标应用程序装载到模拟器上运行,当遇到目标应用程序中与并行无关的系统调用代码时,将此目标机的系统调用转变成宿主机的系统调用执行,对于与并行相关的同步操作则使用相应的算法在模拟器中实现.不运行操作系统的好处在于:(1) 不需要在模拟器中实现与内核态有关的代码,这样使模拟器的代码实现简单;(2) 不需要使模拟器在遇到系统调用代码时频繁陷入到内核态执行,减少了运行时间;(3) 研究者有时只是单纯地想研究应用程序引起的多核之间的通信,想要排除操作系统的代码对核间通信的影响,此时不宜使用操作系统调用来同步各处理器核上运行的进程,以免引起干扰;(4) 若要改动模拟器以实现另外的体系结构,只需修改模拟器的相关部分,无须考虑修改操作系统的代码,这样可以使模拟器的修改非常方便,而且易于跟踪和调试.

2.1 处理器核的模拟

龙芯2号微处理器[8]是中国科学院计算技术研究所研制的高性能通用处理器,采用了四发射超标量超流水结构,实现了先进的转移猜测、寄存器重命名、动态调度等乱序执行技术,以及非阻塞的高速缓存和取数操作猜测执行等动态存储访问机制,其单核性能较高,而且它的物理设计采用ASIC的方法完成,其电路以静态电路和标准宏单元为主,单核功耗较低,所以,它很适合作为处理器核单元来组成多核处理器.Sim-Godson[9]是在SimpleScalar工具集的基础上开发的基于龙芯2号微体系结构的模拟器,它仅使用了SimpleScalar模拟器的数据统计功能,而详细模拟和实现了龙芯2号微体系结构的流水线.在本文中,我们使用Sim-Godson作为分片式CMP模拟器中的单核模拟器模型.

2.2 Cache一致性协议的模拟

CMP结构中各处理器核的访问请求主要是由Cache一致性协议引起的,详细地模拟Cache一致性协议的时序特性对于模拟器的时序模拟非常重要.目前,大部分模拟器对于Cache一致性协议的模拟都是采用消息队列的方式进行,方法如下:在某种数据请求或一致性请求发出时,将其放入消息队列等待处理,消息队列在模拟

器运行固定的拍数后被唤醒,此时,模拟器顺序地服务消息队列中的各种请求,将该请求得到响应的结果.返回给请求者.这种方法将处理器核发出的各种请求看成是顺序发出的,从请求发出到结果返回之间相隔的拍数固定.但是实际上,因为需要优化性能,大多数真实处理器的访存请求以及一致性请求是乱序发出的,加上传输Cache 一致性请求和应答的片上互联网络是存储转发式的,因此,各种请求和应答到达目的的延时也是不确定的.显然,这种队列模型与实际的 Cache 一致性协议的时序特性不一致,而且 Cache 一致性的处理过程也与实际的处理过程不一致.为了使模拟器的结果与真实的机器尽可能一致,本模拟器采用状态机的方法来模拟 CMP 中常用的目录 Cache 一致性协议.处理器中本地节点的存储层次如图 2 所示(其中,P 为不包含一级 Cache 的处理器核),模拟器中引入 Missq 和 Wtbkq 两个队列,其中,Missq 接受和处理各处理器核(包括本地的和远程的)发出的数据访问请求或一致性请求,并访问二级 Cache(本地)或 Router(远程),Wtbkq 接受和处理各处理器核(包括本地的和远程的)发出的脏数据写回请求,并访问二级 Cache(本地)或 Router(远程).模拟器把 Missq 和 Wtbkq 作为联系和控制一级 Cache、二级 Cache 和互联网络的机制,通过模拟 Missq 和 Wtbkq 的详细状态转换,精确地模拟 Cache 一致性协议的完整处理过程.

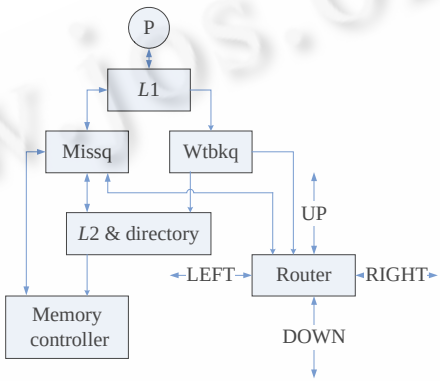


Fig.2 Architecture of host node

图 2 本地节点结构图

Missq/Wtbkq 的状态转换和 Cache 一致性协议的具体处理过程如下:

1. 处理器核 P 发出访存请求到 *L*1,模拟器根据访存地址查找 *L*1 相应块的状态,若不命中,则根据相应的失效请求类型发送请求到 Missq,并在 Missq 中占据一项,将该队列项的状态由 EMPTY 改为 *L*1_MISS.具体的请求类型为:若处理器发出的是读请求且 *L*1 块状态为 INVALID,则发送 READ_SHARED 请求到 Missq;若处理器发出的是写请求且 *L*1 块状态为 INVALID,则发送 READ_EXCLUSIVE 请求到 Missq;若处理器发写请求且 *L*1 块状态为 SHARED,则发送 UPGRADE 请求到 Missq;
2. 根据 Missq 队列项的地址确定是本地的失效请求还是远程的失效请求,若为本地的失效请求,Missq 向本地的 *L*2 和目录发出查找请求,若 *L*2 接受请求(*L*2 端口数有限,每拍只能接受有限的读写请求),则将 Missq 状态由 *L*1_MISS 改为 READ_*L*2,转步骤 3(a);若为远程的失效请求,Missq 向 Router 发出请求,若 Router 接受请求,也将 Missq 状态由 *L*1_MISS 改为 READ_*L*2,转步骤 3(b);
3. (a) 本地的失效请求根据 Missq 发出的请求地址查找本地的 *L*2 和目录状态,若 *L*2 命中且不需要改变目录中其他处理器核对该块的状态,则返回数据,且将 Missq 队列项状态改为 REPLACE_*L*1,转步骤 5;若 *L*2 命中,且需要改变目录中其他处理器核对该块的状态(需要向其他处理器核发出相应的 INTERVENTION 请求),则将队列项状态改为 MODIFY_*L*1,转步骤 4(a);若 *L*2 不命中,则将队列项状态改为 *L*2_MISS,转步骤 4(b);

   (b) 远程的失效请求通过本地的 Router 向片上互联网络发出,互联网络负责将请求发送到对应的目

标处理器核,该请求对于目标处理器核是本地请求,转步骤 2 向下处理;

4. (a) 向其他处理器核发出 INTERVENTION 请求,等待其他处理器核返回响应,若响应均已返回,则将 Missq 队列项状态改为 REPLACE_*L*1,转步骤 5;
   (b) 向内存控制器发出请求,在等待返回数据的同时,根据要替换出的二级 Cache 块状态和目录的情况,决定是否要向其他处理器核发出 INTERVENTION 请求,若数据返回且发出的 INTERVENTION 请求的响应均已返回,则将 Missq 队列项状态改为 REPLACE_*L*1,转步骤 5;
5. Missq 通知 *L*1 请求的数据已返回,让 *L*1 选择替换的块,若替换的块是脏块,则将脏块写入 Wtbkq,然后将 Missq 队列项的状态改为 REFILL_*L*1,转步骤 6;Wtbkq 中的脏块按照地址等待 *L*2 或 Router 有空闲时写入,本地的脏数据写入 *L*2,远程的脏数据通过 Router 写入远程处理器核的 Wtbkq,最终写入远程处理器核的 *L*2;
6. 将返回的数据写入 *L*1,将 Missq 队列项的状态改为 EMPTY,结束该次失效请求.

如上所述,本模拟器通过状态机的方法模拟 Missq 和 Wtbkq 状态的转换,从而详细地模拟了 Cache 一致性协议的完整处理过程,模拟的细节程度与真实机器的 Cache 一致性协议很相近.状态机的效率稍低,但与消息队列的固定延时相比,状态机考虑了由于请求相互竞争而造成等待的时序信息,而且请求得到服务的时机也是乱序的,这样得到的延时数据显然更加可信.

另外,考虑到现今的多核处理器中的片上互联网络的特性(包括网络带宽、网络延时、丢包情况等)与传统的采用CC-NUMA(cache coherence non-uniform memory access)结构的机器(如Origin2000[15]等)很不一样,本文中的Cache一致性协议在处理多个请求相互冲突的情况时,按照先到先服务的方法处理,即先到达的请求先被服务,后到达的冲突请求需要继续在Missq队列中等待,直到先到达的请求返回数据,系统中不再存在冲突的请求时再服务等待的请求.这种方法不需要向冲突请求的发起者返回NACK消息,因而减少了网络带宽的需求,而且简化了控制逻辑.

2.3 片上互联网络的模拟

片上互联网络是分片式 CMP 用以连接各处理器核和二级 Cache 块的互联结构,其网络延时、带宽等重要结构参数决定了互联网络的性能,从而在很大程度上影响了分片式 CMP 的性能.具体来说,片上互联网络包括图 2 中的 Router 以及 Router 之间的互连线,负责给处理器核与二级 Cache 块之间的数据和一致性协议的请求及响应提供低延时、高带宽、无死锁的数据、地址和控制通路,Router 负责接受本地处理器核的各种请求或响应,然后将请求或响应放在互连线上驱动,使之在有限拍数内转发到下行网络的 Router 中.Router 的结构如图 3 所示.从图中可以看出,每个 Router 拥有 5 个 FIFO(first in first out)作为 5 个方向(上、下、左、右、本地)的输入队列,用来暂存网络间传输的数据,中间通过 Arbiter 和 Crossbar 将输入请求向 5 个方向的输出端口转发.这种存储转发式的互联网络能够高效地在功能单元之间传输数据.不过,这也造成了 Cache 一致性协议的各种请求和应答在网络中传输的乱序特性,使 Cache 一致性协议的处理复杂化了.

片上互联网络的平均延迟决定了访问远程二级 Cache 的延迟,因此,互联网络是位于处理器的关键路径上的.例如,当处理器核由于读失效发出共享读请求,在发出后希望尽快收到应答的数据,否则就只能堵在流水线中等待;又例如,处理器核由于写失效发出升级的请求,需要由目录向其他拥有这一块的处理器核发出一致性请求,只有在其他处理器核收到该一致性请求并给予目录应答后才能完成升级请求.以上两个例子的请求都表示片上互联网络的延迟需要尽可能地小,这要求 Router 的流水线结构和路由算法尽可能简单,因此,在本文的模拟器中,片上互联网络的流水线分为较少的 3 级:arbitration,crossbar_switch,flit_traverse,它们的功能描述如下:

(1) arbitration:使用指定的路由算法和仲裁策略,结合下一跳的 Router 的 Credit 的情况为数据包选择可用通道;
(2) crossbar_switch:在所有输出端口中并行执行以下操作:从所有针对输出端口的请求中选择一个出来(一般为输入队列头节点中的数据包),赋以输出端口的使用权,并向上一跳的 Router 发送 Credit 信息;
(3) flit_traverse:把 Flit 放到输出线上,经过 Wire_Delay 的时间,下一跳的 Router 的输入队列就可以接收

到这个 Flit.

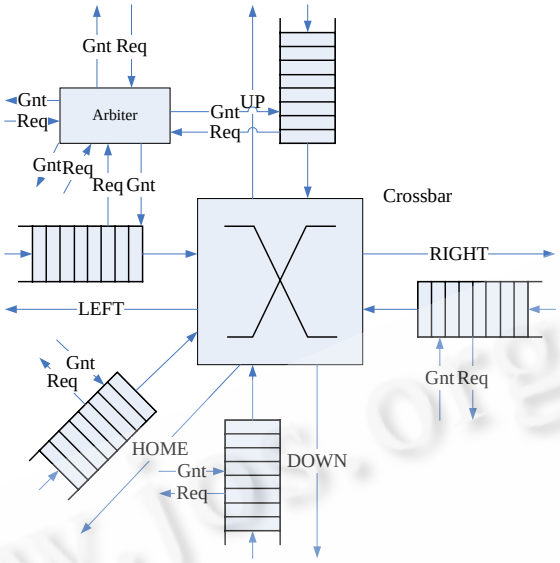


Fig.3 Architecture of router

图 3 Router 的结构图

本模拟器实现的片上网络模型具有良好的灵活性和扩展性,片上网络的拓扑结构、路由算法、仲裁策略和流控策略等均可方便地进行配置,现简述如下:

拓扑结构:本模拟器的片上网络结构实现了两种拓扑结构:二维 Mesh 和二维 Torus,可以通过配置选择哪一种网络结构.

路由算法:片上网络的路由算法有很多种,为了达到延时小的要求,不可能使用非常复杂的路由算法.因此,本模拟器没有实现复杂的自适应性的路由算法,而仅仅实现了确定性路由算法中的 XY 路由算法(二维 Mesh)和 TXY 路由算法(二维 Torus),即简单地先走 *X* 方向的路由,待 *X* 方向走完后再走 *Y* 方向的路由.这两种算法都很简单,Router 可以在 arbitration 阶段在一拍内就计算出数据包的下一跳的地址,实现的硬件复杂度低.

仲裁策略:模拟器实现了Round Robin和Wave Front两种仲裁策略[16].

流控策略:模拟器实现了基于Credit的流控策略[16],即当下行网络中有空余的输入队列项时,会向上行网络发送一次Credit信号,通知上行网络能够发送数据包.上行网络接收到Credit信号后,才会给申请这个方向的数据包赋以输出端口的使用权.

本模拟器可以轻松地通过配置来决定选择上述的片上互联网络中的哪一种网络结构、路由算法、仲裁策略和流控策略,而且由于模拟器的模块化较好,用户也可以较容易地添加其他结构和算法策略.

3 运行环境

本模拟器的指令执行引擎大部分借用Sim-Godson模拟器的指令执行引擎,但是,Sim-Godson模拟器只能运行单进程且静态编译的程序映像文件,这在运行诸如SPEC CPU2000 等串行程序时没有问题.但是,为了使模拟器能够运行SPLASH-2[17]等并行程序集,我们还需要提供支持并行线程执行和同步的系统调用,而且由于不运行操作系统,我们需要手工地安排并行程序的装载运行以及并行线程在模拟的目标机内存中的布局.

3.1 同步系统调用

以 SPLASH-2 中的 OCEAN 为例,并行程序的结构一般如图 4 所示.

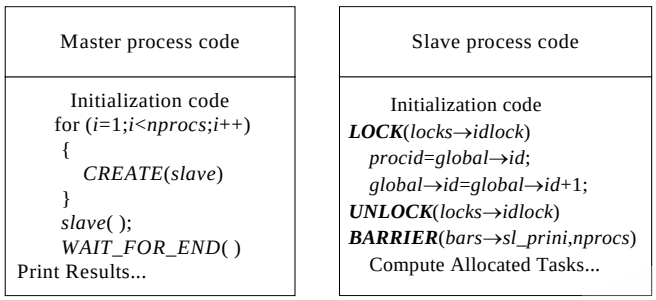


Fig.4 Structure of parallel program OCEAN

图 4 并行程序 OCEAN 的结构

从图 4 中可以看出,并行程序都会从一个固定入口运行,此时,系统中只有一个主进程运行在主处理器上,在主进程初始化完成后会调用 ***CREATE*** 之类的库函数或系统调用生成从进程;然后,主进程和从进程分别由系统调度到不同的处理器上运行,在计算过程中会调用 ***LOCK***,***UNLOCK***,***BARRIER*** 等库函数或系统调用来实现进程间的同步;最后,当所有进程的计算任务完成后由主进程规约所有的进程,此时,所有的从进程被结束,只剩下主进程运行,在统计计算的结果之后结束主进程.由于这些并行的系统调用与具体的体系结构的实现相关,因此,我们需要在模拟器中实现上述系统调用.

接下来,本节的其他内容说明如何在模拟器中实现上述系统调用.由于 ***CREATE*** 和 ***WAIT_FOR_END*** 等系统调用与应用程序在目标机内存中的布局有着很密切的关系,因此把它放在下一节中加以说明.对于其他系统调用,它们涉及的都是关于并行程序之间的同步操作.

以锁同步操作为例,程序首先会调用 ***LOCKINIT*** 声明某锁变量,若进程 *A* 要访问某一全局变量,为了不引起混乱,进程 *A* 需要对该全局变量拥有排他的所有权,做法是,*A* 进程调用 ***LOCK*** 对声明过的锁变量加锁,然后对此全局变量进行访问,此时,若有 *B* 进程要对全局变量访问,会检测到锁变量已经被加锁,进程 *B* 在原地等待,直到进程 *A* 调用 ***UNLOCK*** 将锁变量释放,进程 *B* 才能够对锁变量加锁,从而访问全局变量.

为了模拟这种获取锁和释放锁的过程,模拟器在内部维护一个全局的同步变量队列,队列中的每一项表示系统内的同步变量以及同步变量的状态,模拟器通过访问该全局队列来模拟不同进程对锁变量的竞争情况.当锁变量初始化时,应用程序首先会调用 ***LOCKINIT*** 来通知系统有某一锁变量被声明,当模拟器执行 ***LOCKINIT*** 系统调用时,会在全局的同步变量队列中占用一项作为该锁变量的内部状态,并且用该队列项在同步变量队列中的位置来索引该锁变量.进程在调用 ***LOCK*** 时,模拟器首先查看锁变量对应的队列项的状态,若状态为已经被其他进程加锁,则当前进程等待锁变量被释放;否则,改变队列项的状态为已经加锁,然后继续往下执行程序代码,直到调用 ***UNLOCK*** 释放锁.在调用 ***UNLOCK*** 时,模拟器将该锁变量对应的队列项的状态改为未被加锁,以供其他进程获得锁变量.对于 ***BARRIER*** 等其他同步操作,模拟器采用类似的内部全局同步变量队列来模拟对应的同步算法.

3.2 应用程序的装载及其在目标机内存中的分布

由于模拟器不运行操作系统,因此,需要模拟器提供装载程序手动地将应用程序装载到模拟的目标机的内存中.在运行操作系统的真实的多核系统上,操作系统在调度进程的运行时,会尽量将一个进程调度到同一个处理器核上运行,这样可以重用处理器核的 Cache 中的内容,从而使 Cache 失效率尽量降低.为了模拟实际操作系统的这种亲核调度的特性,本模拟器仔细安排应用程序在目标机内存中的分布情况,在应用程序装载进目标机内存时,按照此分布情况进行装载,同时,在执行 ***CREATE*** 系统调用时,按照此布局情况生成从进程的执行代码.分布情况如图 5 所示(其中,*ld_brk_point* 指示每个进程的堆地址,*regs_R*[29]为每个进程的栈指针寄存器的值,*thread_size* 为每个进程的分配的进程空间大小,*pid* 为每个进程的进程号,主进程号为 0).

如图 5 所示,模拟器在初始化时,会先申请一些本地机的内存,用这些内存空间模拟生成目标机的一个初始 0 号进程,模拟器中的装载程序会将命令行中指定的测试程序装载到目标机的 0 号进程的内存中(将测试程序文

件中的代码段、数据段装载到如图 3 所示的 0 号进程相应的地址空间,堆空间指针使用模拟器中的全局变量 *ld_brk_point* 表示,堆的起始地址为数据段的顶端位置,当程序中遇到 ***MALLOC*** 系统调用时,会动态扩充堆的大小, *ld_brk_point* 会随之增加,*regs_R*[29]为进程中栈指针寄存器,其初始值为每个进程的顶端地址,当程序中遇到函数调用等时,会在代码中改变栈指针寄存器的值,栈空间大小则随之发生变化).当 0 号进程执行到 ***CREATE*** 系统调用时,则按照图 3 中从进程相应的地址空间分布情况初始化从进程的堆和栈(堆起始地址为在每个从进程的起始空间向上加上 4MB 空间,用以隔开上一进程,栈空间则从每个从进程的顶端向下扩展,从进程的起始地址和顶端地址由进程号 *pid* 计算得到).从进程所需的代码和数据是全局数据,因此,在模拟器中采用共享 0 号主进程的代码段和数据段的内容的方法.事实上,正是这些共享内容的读写引起了进程间的数据共享,从而造成处理器核之间的通信.当各个从进程完成分配到的计算任务后,等待退出执行,0 号主进程通过调用 ***WAIT_FOR_END*** 系统调用来规约各从进程的计算结果.从上面的说明中可以看出,模拟器中的这种地址空间布局方法较好地模拟了测试程序在真实系统上运行时的情况,而且在模拟了同步操作的系统调用后,不需要运行操作系统来进行同步,模拟器的运行速度大大加快了.

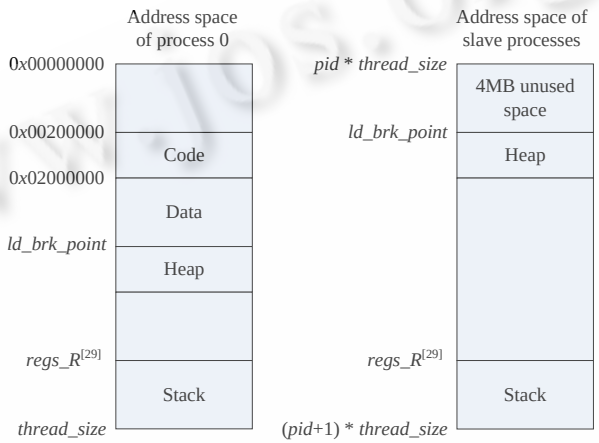


Fig.5 Application's memory allocation

图 5 应用程序内存分布情况

4 评测数据

4.1 速度评测

使用者对于模拟器最直观的感受来源于模拟器的速度,模拟器运行的快慢程度是用户选择模拟器的重要因素.本文的模拟器由于非常详细地模拟了处理器的行为特性(包括详细的乱序处理器核、完整的 Cache 一致性协议以及片上互联网络),因此需要占用大量的 CPU 计算时间;而评测数据表明,模拟器的运行速度可达 200K~300K 指令/s.从这一点来看,本模拟器的速度与常用的用户级模拟器相差不大.

评测环境是一台 AMD64 3200+的机器,内存为 512MB.图 6 给出了模拟器运行 SPLASH-2 并行程序集时的速度,从图中我们可以看出,大部分程序在模拟器分别模拟 2 核、4 核、8 核和 16 核时的运行速度大致上是线性下降的,而 OCEAN 程序由于程序结构中数据共享模式的特殊性,导致进程间的同步操作比较多而需要频繁等待.因此,在模拟 16 核运行时,模拟器速度相比于分别模拟 2 核、4 核、8 核时速度下降得很快,说明 OCEAN 程序在模拟 16 核运行时由进程间的同步操作导致的延迟大为增加.

本模拟器不仅可以运行并行程序,还可以将多个串行程序绑定在各处理器核上运行.我们在 SPEC2000 程序集中随机地选出 4 个程序(2 个定点程序,2 个浮点程序)组合在一起,模拟器模拟的处理器核数目设置为 4,将各组程序分别绑定在各处理器核上运行.图 7 给出了模拟器运行 SPEC2000 测试程序集时的数据.从图中数据可

以看出,由于运行串行程序时各处理器核之间没有同步通信的开销,因此,模拟器的运行速度要比模拟相同处理器核数时运行并行程序的速度要高一些.

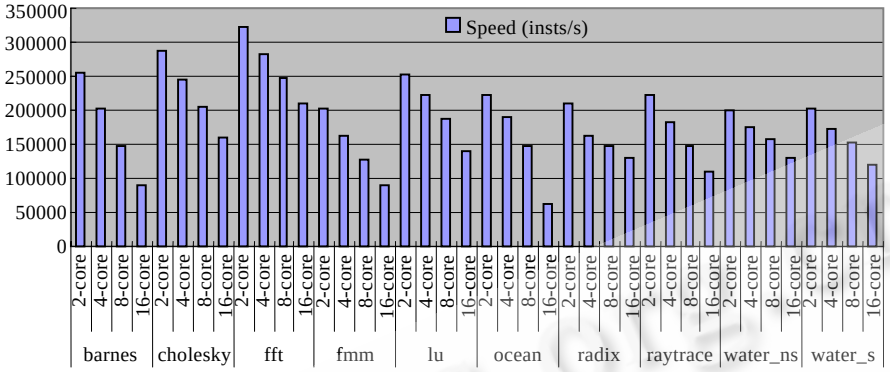


Fig.6 Simulator speed of running SPLASH-2

图 6 模拟器运行 SPLASH-2 时的速度评测

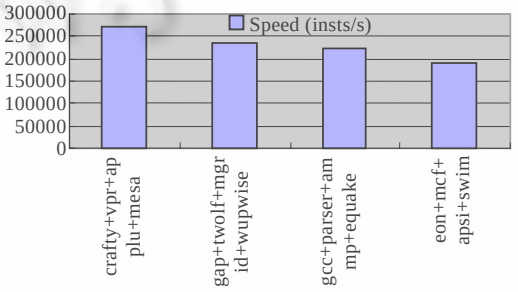


Fig.7 Simulator speed of running SPEC2000

图 7 模拟器运行 SPEC2000 时的速度评测

表 1 给出了本模拟器与其他同类型模拟器运行速度的比较.从表中数据可以看出,本模拟器的运行速度相对较快.

Table 1 Speed comparison of similar simulators

表 1 与同类型模拟器的运行速度的比较

GEMS	SESC	SimOS-Goodson[18]	Simulator of this paper
~100K instructions/s	~200K instructions/s	~300K instructions/s	~300K instructions/s

4.2 性能评测

本文的多核模拟器可以灵活地评测出在不同结构参数下的处理器性能指标,因此可以方便地评估多核处理器的结构特性并进行设计空间的探索.图 8 给出了模拟器运行 SPLASH-2 的某些程序时,当网络的线延迟(wire_delay)为 1 拍和 2 拍时片上互联网络的平均延迟.由图中数据可以看出,线延迟的增加大大增加了片上互联网络的平均延迟.

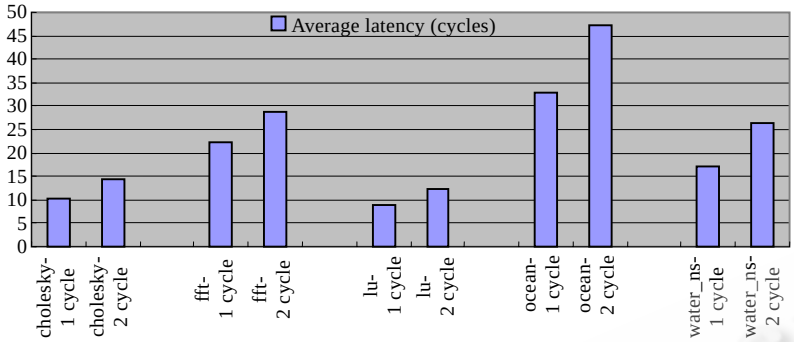


Fig.8 Average latency of network-on-chip when wire delay are 1 cycle and 2 cycles respectively

图 8 线延迟为 1 拍和 2 拍时片上互联网络的平均延迟

5 结论和未来工作

针对目前处理器设计的新方向——分片式多核微处理器,本文以龙芯 2 号处理器和对应的 Sim-Godson 模拟器为单处理器核模型,实现了用户级的多核性能模拟器,详细刻画了乱序的 Cache 一致性协议的处理过程和存储转发式的片上互联网络结构以及它们的时序特性,能够通过在模拟器上运行串行或并行的工作负载来评估分片式多核微处理器的结构特性并进行设计空间的探索,可以为将来的龙芯 3 号多核处理器的设计提供依据.在未来的工作中,我们计划将结构级功耗评估的功能加入到多核模拟器中,并针对片上互联网络使用该模拟器评价不同路由算法和仲裁策略的功耗和性能,从而在此基础上研究平衡性能和功耗的片上互联网络的结构设计.

References:

[1] McNairy C, Bhatia R. Montecito: A dual-core, dual-thread itanium processor. IEEE Micro, 2005,25(2):10−20.

[2] Kongetira P, Aingaran K, Olukotun K. Niagara: A 32-way multithreaded sparc processor. IEEE Micro, 2005,25(2):21−29.

[3] Kahle JA, Day MN, Hofstee HP, Johns CR, Maeurer TR, Shippy D. Introduction to the cell multiprocessor. IBM Journal of Research & Development, 2005,49(4-5):589−604.

[4] Patterson D, Hennessy J. Computer Architecture: A Quantitative Approach. 4th ed., San Francisco: Morgan Kauffman Publishers, 2006.

[5] Taylor MB, Lee W, Miller J, Wentzlaff D, Bratt I, Greenwald B, Hoffmann H, Johnson P, Kim J, Psota J, Saraf A, Shnidman N, Strumpen V, Frank M, Amarasinghe S, Agarwal A. Evaluation of the raw microprocessor: An exposed-wire-delay architecture for ILP and streams. In: Proc. of the Int'l Symp. on Computer Architecture. Munich: IEEE Computer Society, 2004. 2−13.

[6] Sankaralingam K, Nagarajan R, Liu H, Huh J, Kim CK, Burger D, Keckler SW, Moore CR. Exploiting ILP, TLP, and DLP using polymorphism in the TRIPS architecture. In: Proc. of the 30th Annual Int'l Symp. on Computer Architecture. New York: ACM Press, 2003. 422−433.

[7] Kim CK, Burger D, Keckler SW. An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. In: Proc. of the Int'l Conf. on Architectural Support for Programming Languages and Operating Systems. New York: ACM Press, 2002. 211−222.

[8] Hu WW, Zhao JY, Zhong SQ, Yang X, Guidetti E, Wu C. Implementing a 1GHz four-issue out-of-order execution microprocessor in a standard cell ASIC methodology. Journal of Computer Science and Technology, 2007,22(1):1−14.

[9] Zhang FX, Zhang LB, Hu WW. Sim-Godson: A godson processor simulator based on SimpleScalar. Chinese Journal of Computers, 2007,30(1):68−73 (in Chinese with English abstract).

[10] Martin MMK, Sorin DJ, Beckmann BM, Marty MR, Xu M, Alameldeen AR, Moore KE, Hill MD, Wood DA. Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset. Computer Architecture News (CAN), 2005. http://www.cs.wisc.edu/gems

[11] Magnusson PS, Christensson M, Eskilson J, Forsgren D, Hållberg G, Högberg J, Larsson F, Moestedt A, Weruer B. Simics: A full system simulation platform. IEEE Computer, 2002,35(2):50−58.

[12] http://sesc.sourceforge.net. 2005.

[13] Shang L, Peh LS, Jha NK. Dynamic voltage scaling with links for power optimization of interconnection networks. In: Proc. of the 9th Int'l Symp. on High-Performance Computer Architecture. Anaheim: IEEE Computer Society, 2003. 91−102.

[14] Wang HS, Peh LS, Malik S. Power-Driven design of router microarchitectures in on-chip networks. In: Proc. of the 36th Int'l Symp. on Microarchitecture. San Diego: IEEE Computer Society, 2003. 105−116.

[15] Laudon J, Lenoski D. The SGI origin: A ccNuma highly scalable server. In: Proc. of the 24th Annual Int'l Symp. on Computer Architecture. Denver: ACM Press, 1997. 241−251.

[16] Dally WJ, Towles B. Principles and Practices of Interconnection Networks. San Francisco: Morgan Kaufmann Publishers, 2003.

[17] Woo SC, Ohara M, Torrie E, Singh JP, Gupta A. The SPLASH-2 programs: Characterization and methodological considerations. In: Proc. of the 22nd Int'l Symp. on Computer Architecture. Santa Margherita Ligure: ACM Press, 1995. 24−36.

[18] Gao X, Zhang FX, Tang Y, Zhang LB, Hu WW, Tang ZM. SimOS-Goodson: A goodson-processor based multi-core full-system simulator. Journal of Software, 2007,18(4):1047−1055 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/18/1047.htm

附中文参考文献:

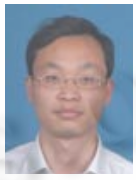
[9] 张福新,章隆兵,胡伟武.基于 SimpleScalar 的龙芯 CPU 模拟器 Sim-Godson.计算机学报,2007,30(1):68−73.

[18] 高翔,张福新,汤彦,章隆兵,胡伟武,唐志敏.基于龙芯 CPU 的多核全系统模拟器 SimOS-Goodson.软件学报,2007,18(4):1047−1055. http://www.jos.org.cn/1000-9825/18/1047.htm

黄琨(1979－),男,湖北公安人,博士,主要研究领域为微处理器系统结构设计,性能评估和功耗评估.

张戈(1983－),男,博士,助理研究员,主要研究领域为高性能处理器设计,低功耗设计.

马可(1980－),男,博士,主要研究领域为微处理器系统结构设计,性能评估.



章隆兵(1974－),男,博士,副研究员,主要研究领域为微处理器设计,机群计算.

曾洪博(1981－),男,博士,主要研究领域为微处理器系统结构设计,缓存一致性协议.