◎博士论坛◎

基于指令 Cache 作废的多核处理器同步技术

郭建军,戴 葵,王志英
GUO Jian-jun,DAI Kui,WANG Zhi-ying

国防科技大学 计算机学院,长沙 410073
School of Computer,National University of Defense Technology,Changsha 410073,China
E-mail:jjguo@tom.com

GUO Jian-jun,DAI Kui,WANG Zhi-ying.Synchronization technology based on invalidation of instruction Cache for multi-core processor.Computer Engineering and Applications,2009,45(4):1-3.

Abstract: The "busy-wait" technology is often used to implement locks or barriers in shared memory multi-core processors. These synchronization mechanisms are restricted by the long latency and resource contention problems and are not scalable.They often need to access memory repeatedly and affect the normal memory accessing process.A synchronization technology based on the invalidation of instruction cache for the SDTA-based multi-core processor is proposed.At the synchronization point,the processor cores invalidate the corresponding instruction cachelines which cause instruction fetch miss.Then instruction fetch requests are issued to the L2 cache.The L2 cache adopts a filter mechanism to freeze those fetch requests to suspend the processor cores that need synchronization.The proposed mechanism is scalable and has a better performance.

Key words: Synchronous Data Triggered Architecture(SDTA);multi-core processor;invalidation;synchronization

摘 要:共享存储多核处理器中"忙-等待"技术常用来实现锁或栅栏等同步操作,这些典型的同步机制通常受限于较长的同步延迟和资源竞争等问题,导致扩展性较差,且需要不时进行访存操作,影响正常存储器访问操作,加剧对存储系统的带宽需求。提出了一种用于同步数据触发结构多核处理器的基于指令 Cache 作废的同步技术,同步时作废将执行的指令 Cache 行导致取指失效,向 L2 Cache 发送取指请求,L2 Cache 中设置相应的过滤机制,不服务不满足同步条件的处理器核的取指请求,使相应处理器核暂停,达到同步目的。测试表明,该方法在可扩展性和同步性能方面均具有一定的优势。

关键词:同步数据触发结构;多核处理器;作废;同步技术
DOI:10.3778/j.issn.1002-8331.2009.04.001 **文章编号**:1002-8331(2009)04-0001-03 **文献标识码**:A **中图分类号**:TP391

1 前言

多核处理器上运行的多个任务之间协同需要提供高效的同步机制,同步机制的效率对多核处理器效能的发挥起着十分重要的作用。典型的同步机制是基于"忙-等待"技术的,共享存储结构多核处理器中"忙-等待"技术常用来实现锁或栅栏等同步操作[1-5]。这些典型的同步机制受限于较长的同步延迟、资源竞争等问题,导致扩展性较差,且需要不时进行访存操作,影响正常存储器访问操作,加剧对存储系统的带宽需求,使得多核条件下"存储墙"问题表现更为突出。

2 同步数据触发结构多核处理器

图1 给出了本文的同步数据触发结构多核处理器框图,其中包含一个做控制核的通用处理器和多个同步数据触发结构(Synchronous Data Triggered Architecture,SDTA)计算核。控制核负责控制相关的任务,将计算任务分配给计算核进行运算,计算核负责计算密集型任务加速。控制核和计算核通过全局互连网络连接在一起,共享下层二级 Cache。

SDTA 结构继承了 TTA 结构[6]在指令并行性挖掘上的优势,在此基础上,对 TTA 结构的不足进行了改造与完善。首先,重新划分了流水段。TTA 结构中指令流水处理统一分为取指、译码、传输和执行段。SDTA 结构对流水线中各种操作类型区分对待,为少量功能复杂的操作设置独立的执行流水段,而大多数简单功能的执行操作则与数据传输段合并,流水段划分更加合理。其次,改进了分支处理机制,利用专门旁路机制,减小

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.60773024);国家高技术研究发展计划(863)(the National High-Tech Research and Development Plan of China under Grant No.2007AA012101);国家重点基础研究发展规划(973)(the National Grand Fundamental Research 973 Program of China under Grant No.2007CB310901)。

作者简介:郭建军(1981-),男,博士生,主要研究方向:计算机体系结构、微处理器设计;戴葵(1968-),男,博士,副教授,主要研究方向:高级计算机体系结构、微处理器设计、IC 设计;王志英(1956-),男,教授,博士生导师,主要研究方向:高级计算机体系结构、高性能微处理器设计、计算机安全。

收稿日期:2008-10-09 **修回日期**:2008-11-20

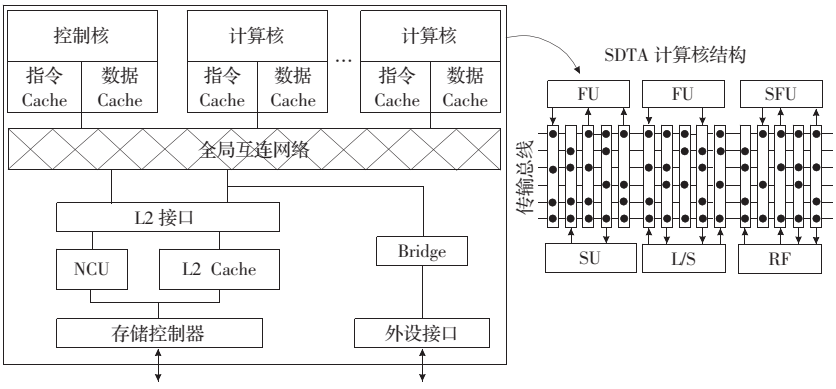

图 1　同步数据触发结构多核处理器框图

了分支延迟，更有利于编译器调度无关指令填充分支延迟槽。第三,SDTA 结构将数据与操作信息绑定传输，降低了 TTA 结构中原有译码逻辑的复杂度和延迟。最后,SDTA 内部功能单元数据宽度较宽,更有利于向子字并行[7]和向量并行两个方向进一步挖掘应用中广泛存在的数据级并行。

典型的 SDTA 计算核结构(如图 1 所示)由多个功能单元通过多条传输总线连接在一起，各功能单元之间耦合度较低，可以灵活地根据应用配置功能单元种类及个数,并且可以根据应用需求设计专门的特殊功能单元。本文将利用这一特点设计专门用于同步的功能单元用于实现计算核同步。SDTA 结构中功能单元采用数据触发机制,内部包含一个或多个操作数寄存器(O)、唯一的触发寄存器(T)与若干个结果寄存器(R)。当数据写入触发寄存器时,会触发功能单元将操作数寄存器和触发寄存器中的值作为源操作数来完成具体操作,并将结果写入结果寄存器。该结构下,各种操作均通过数据传输指令完成,比如一条典型加法指令可以分解为如下三条数据传输操作:$add\ r_3, r_2, r_1 \Rightarrow r_1 \rightarrow ADD\_O; r_2 \rightarrow ADD\_T; ADD\_R \rightarrow r_3$。首先,$r_1$ 和 r_2 的值被分别写入加法单元的 O 寄存器与 T 寄存器，触发加法操作,之后将计算结果写入 R 寄存器直接使用,也可以暂存如 r_3。

多个处理器核之间采用共享存储结构,核间的同步与通信机制都是基于共享存储结构来完成的。这种结构下传统的旋转锁同步机制[8]将一个存储单元定义为锁,锁值为“0”表示锁空闲,锁值非“0”表示锁忙。几个处理器核共享锁变量,使用软件锁算法实现对共享变量的互斥访问。初始状态下，锁值都为“0”。当某个处理器核希望获取对某个共享变量的独占访问权限时,它读出与该共享变量对应的锁值,然后进行判断。如果读出的值为“0”,表明锁空闲,当前处理器核就将自己的 ID 号写入这个锁单元,加锁成功,获得锁权限。如果读出的值非“0”,表明锁忙,加锁失败,重复上述过程去试图获得该锁。在释放锁时,处理器核将相应的锁值清零即可。

对锁变量的访问,生存周期较长,多个处理器核竞争访问时,只有一个处理器核能够获得锁,其他处理器核需要不停测试锁的忙闲,直到获得锁为止。这个过程中需要不停地进行访存操作,同时还需要维护锁变量的一致性,这些都影响了处理器核的正常访存操作。如果同步变量处于不可 Cache 区域,则每次同步变量访问都需要访问下层二级 Cache 或者直接访问存储器,这需要耗费较多的处理器周期数。即使同步变量处于可 Cache 区域,为了维护 Cache 一致性,也增加了总线访问、一致性维护等开销,对其他处理器核访存会有所影响。

3　基于指令 Cache 作废的多核同步技术

传统旋转锁同步技术中处理器核需要不断主动探测同步资源的忙闲,属于一种主动探测的机制,本文提出一种被动方式的基于指令 Cache 作废的多核同步技术。处理器核在需要同步时作废自己将要执行的指令 Cache 行,导致取指失效,继而向 L2 Cache 发出取指请求,在 L2 Cache 中设置一个过滤器来处理各个处理器核的取指请求,仅服务满足同步条件的处理器核的取指请求,使其向前执行,未能满足同步条件的处理器核的请求将不被服务,从而该处理器核将暂停,直到满足同步条件才重新取得指令向前执行。通过这种机制达到多核同步的目的。

3.1　同步实现技术

为了支持上述同步方案,处理器核指令 Cache 需要支持作废操作,处理器核需要能够发出指令 Cache 作废(invic)命令和释放(rlsic)命令。在每个计算核内部设置了一个专门的用于同步的功能单元 SyncUnit(SU),实现 invic 和 rlsic 指令,前者作用是作废相应的指令 Cache 行并且通知 L2 过滤器过滤相应的取指请求,后者作用是释放相应的指令 Cache 行,即通知 L2 过滤器放行相应的取指请求。

支持所提同步技术的一个关键模块是位于 L2 Cache 中的过滤器,用于过滤那些来自需要暂停服务的处理器核的取指请求,具体的结构如图 2 所示。过滤器中包含一个表,有一定数量的入口。每个表入口包含同步标识 T、作废指令地址 A、服务状态标识 S。过滤器根据不同的策略服务取指请求,使相应的处理器核获得访问权限。锁同步中,当一个处理器核退出临界区、释放锁之后,它需要通知过滤器释放参与同步的其中一个处理器核的取指请求,使其获得锁权限。只要有相同标识的行被服务了，就说明已经有一个处理器核取得锁权限进入临界区,再来的其他请求就要被阻塞,直到获得权限的处理器核释放锁权

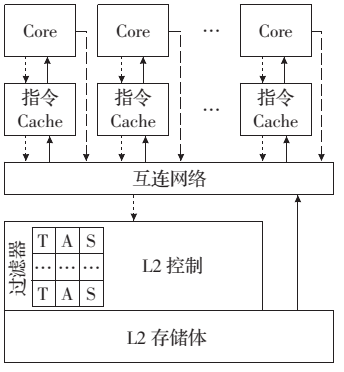

图 2　同步支持结构

限，选择一个请求进行服务。具体的处理机制如下：

（1）初始化时表为空。

（2）当接收到带特定标识（锁地址）的作废时，查找过滤表，比较标识 T 和地址 A，如果 T 不命中，说明当前不存在对同一锁变量的请求，插入一个表项，设置 T、A、S 等相关域。如果有 T 命中，说明存在对同一锁变量的请求，若地址 A 不相同，则选择一个表项，设置 T、A、S 等相关域；若地址 A 相同，则更新表项内容，设置 T、A、S 等相关域。

（3）接收到取指请求，比较 A，如果没有地址匹配，则该取指请求是正常取指请求，过滤器不做处理。若有地址匹配且与匹配项 T 值相同的项中有 S 值为 1，说明已经有处理器核获得锁权限，暂存该取指请求。若有地址匹配且与匹配项 T 值相同的项中 S 均不为 1，说明没有处理器核获得锁权限，则从这些 T 值相同的项中选择一个取指请求进行服务，将 S 置 1。

（4）接收到释放请求，比较 T，将 T 相同且 S 为 1 的项的 S 置 0，说明前次获得锁权限的处理器核已经释放锁权限，此时，如有即到的取指请求或挂起的取指请求，则按照（3）中选择一个请求进行服务。

3.2　同步机制

应用程序中包含对锁的调用，锁同步机制的代码如图 3 所示，其中 fu27 是同步功能单元，r1 是锁变量地址，r2 是要作废的指令地址，使用该地址作废相应的指令。同步代码内部都包含指令 Cache 作废指令，执行同步代码后将相应的指令 Cache 行作废，当处理器核继续执行时将试图取得该行，如果发生取指失效，流水线暂停直到失效处理完毕。该 Cache 行将被 L2 过滤器阻塞，由过滤器控制以不同的服务策略满足这些取指请求。

```
//加锁过程 Lock
r1->fu27.invic_o1,r2->fu27.invic_t,…,…,…,…,…,…;//作废指令 Cache 行通知
…,…,…,…,…,…,…,…,…;                    //L2 过滤相应的取指请求
…,…,…,…,…,…,…,…,…;
//解锁过程 Unlock
r1->fu27.rlsic_t,…,…,…,…,…,…,…,…;   //释放对取指请求的过滤
```

图 3　计算核基于指令 Cache 作废的锁同步代码

本文方法还可以用于实现栅栏，需要在过滤器中设置同步计数。处理器核在执行栅栏代码中作废相应的指令 Cache 行，导致取指失效，向 L2 Cache 发取指请求，L2 过滤器冻结取指操作，相应的处理器核暂停。当同步计数值达到参与同步的处理器核数时，说明所有处理器核已经到达栅栏，此时过滤器释放冻结的取指操作，各个处理器核才可以向前执行。

4　性能测试及分析

性能测试使用了两个程序：分块流水模型（Partitioned Pipelined Model，PPM）和视频编码程序 H.264[9]。前者进行大规模矩阵运算（4 096×4 096），将矩阵分块后，按小块以流水方式通过处理单元阵列进行处理，它是从各类应用程序中抽象出来的矩阵处理模型；后者是一个实际应用程序，实现了 QCIF 格式视频图像压缩。

L2 过滤器可以采用几种服务策略处理取指请求：全随机（RANDOM0）服务、部分随机（RANDOM1）服务和先到先服务（FIFO）。其中全随机服务在所有到达的请求中随机选择一个进行服务，部分随机服务中当前服务的请求被放到一个链表结尾，不参与下一次随机选择过程，先到先服务则是按照请求到达的先后顺序进行服务。这里比较了传统基于共享 Cache 结构多核处理器中的旋转锁同步机制（同步方法 1，SYN1）和本文同步机制（同步方法 2，SYN2）的性能。

图 4 给出了 PPM 在不同同步核数量情况下分别使用两种同步机制进行同步的平均同步开销。从图中可以看出，本文同步方法的同步开销要小，并且随同步核数量增加同步开销变化幅度要小，说明其可扩展性要比同步方法 1 要好。其中，采用 FIFO 服务策略的同步性能要优于采用其他两种服务策略的同步性能，最后只采用 FIFO 服务策略用于该同步技术实现。

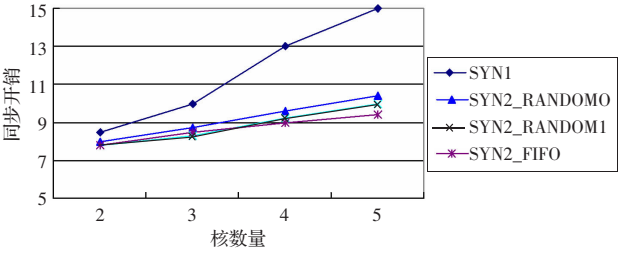

图 4　不同核数条件下的锁同步开销

图 5 是程序 PPM 和 H.264 使用同步方法 2 进行同步相对于使用同步方法 1 进行同步获得的程序性能加速比。从图中可以看出，这两个程序使用同步方法 2 带来的程序加速比都要大于 1，并且随着同步核数量增加，性能加速比开始增大，进一步说明了采用同步方法 2 进行同步的可扩展性和性能优势。

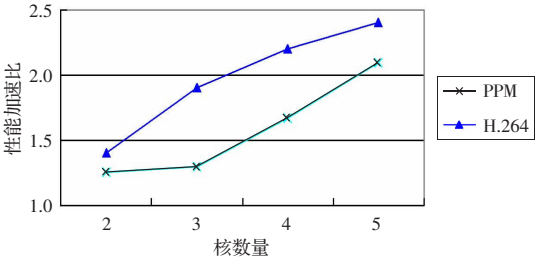

图 5　不同核数条件下的程序性能比较

5　结论

同步机制对于多核处理器性能发挥起着十分重要的作用，典型的同步技术包括锁同步和栅栏同步，根据底层原子指令的不同，实现的锁和栅栏的性能也有所差别。共享存储结构多核处理器下的同步操作涉及到资源竞争、Cache 一致性等问题，会影响正常访存过程，加剧对下层存储器的带宽需求，妨碍了处理器整体性能的发挥。

为提高 SDTA 多核处理器同步性能，利用 SDTA 结构易于加入功能单元的特性，提出了被动方式的基于指令 Cache 作废的多核同步技术，设计了用于同步的功能单元。所提同步技术利用了处理器中本来的访存通路，通过作废将要执行的指令 Cache 行达到同步的目的。在取指失效时向 L2 Cache 发送取指请求，在 L2 Cache 中设置相应的过滤机制，通过不同的服务策略达到不同的同步目的。测试表明，该方法在可扩展性和同步性能方面均具有一定的优势。

参考文献：

[1] Mellor-Crummey J M，Scott M L.Algorithms for scalable synchronization on shared-memory multiprocessors[J].ACM Transactions on Computer Systems，1991，9(1)：21-65.

（下转 63 页）