doi:10.3772/j.issn.1002-0470.2019.03.001

多核片上系统全局主动访存优化研究①

李 鹏②[*,**,***] 曾 露[****] 王焕东[****] 章隆兵[***]

([*]计算机体系结构国家重点实验室(中国科学院计算技术研究所) 北京 100190)

([**]中国科学院计算技术研究所 北京 100190)

([***]中国科学院大学 北京 100049)

([****]龙芯中科技术有限公司 北京 100095)

摘 要 本文提出了一种多核片上系统(MPSoC)全局主动访存调度优化方法(GPMS)来提升系统的访存性能。该方法利用 IP(intellectual property)核的访存局部性和延迟容忍度,通过限制访存冲突的 IP 核使其在一个调度窗口内分别连续访问内存,从而减少访存冲突次数,同时不存在访存冲突的 IP 核在调度窗口内一直保持内存的使用权,从而可以充分发挥内存控制器端访存队列调度的自由度和 DRAM 的 bank 级并行性。实验结果表明,当 IP 核间访存冲突严重时,该方法相比访存队列调度方式可以提升 1 到 2 倍的访存带宽。

关键词 多核片上系统(MPSoC), 访存调度, 访存局部性, 延迟容忍度, 服务质量

0 引 言

现代多核片上系统(multi-processor system-on-chip, MPSoC)通常集成多个针对不同应用的 IP(intellectual property)核,这些 IP 核共享内存,对存储子系统的服务质量、吞吐率和公平性等有比较严格的要求。与此同时,IP 核内规则核间混杂的特点使得单个工作的 IP 核具有很高的访存性能,而同时工作的 IP 核之间存在严重的访存冲突(页冲突和读写冲突),加大了存储子系统设计的难度。随着工艺的提升和市场需求的发展,多核片上系统将更多地从以"计算"为中心转变为以"数据搬运"为中心,因此存储子系统的性能对多核片上系统的影响越来越大,成为芯片设计的关键因素,多核片上系统的访存优化工作势在必行。

针对多核片上系统对内存子系统的压力和挑战,本文充分参考和借鉴现有的访存优化方法(global proactive memory scheduling, GPMS),在此基础上提出了一种全局主动访存调度优化方法。该方法相比传统的访存队列调度方式,更加合理地利用了 IP 核的访存特性,从而大幅提高了系统的访存性能。

1 相关工作

多核片上系统的 IP 核一般都有一定的服务质量(quality of service,QoS)需求,尤其是一些实时性很高的 IP 核,对带宽或延时有一定的需求,这在存储子系统上的体现就是需要保证一定的吞吐率和公平性,本质上就是优化访存调度。访存调度优化方法一般都是根据动态随机存取存储器(dynamic random access memory, DRAM)的访存特性,通过利用各种有利因素(bank 并行性、页打开优先等)来对访

① 国家"核高基"科技重大专项课题(2009ZX01028-002-003, 2009ZX01029-001-003, 2014ZX01020201, 2014ZX01030101)和国家自然科学基金(61521092, 61232009, 61222204, 61432016)资助项目。

② 男,1986 年生,博士生;研究方向:计算机系统结构;联系人,E-mail: lipeng-cpu@ict.ac.cn

(收稿日期:2018-07-06)

存序列进行重定序,降低访存序列的无序性,达到访存优化的目的。

文献[1]提出了先准备好-先来先服务(first ready-first come first service,FR-FCFS)的调度策略,该策略简单有效,被广泛用于以后的实际设计和性能对比方案中。文献[2]提出了一种突发(burst)调度策略(同一页访问请求组成一个 burst),根据 DRAM 时序约束兼顾了 burst 间的调度,最大化了总线利用率。文献[3]提出了基于页打开的多 bank 内存控制器,通过采用贪婪启发式的调度算法,对不同 bank 的请求分别仲裁。文献[4]提出了一种动态多优先级调度策略,根据不同线程在末级缓存(last level cache,LLC)的每千周期缺失数来分类(带宽敏感型和延迟敏感型),通过粗细两种粒度对不同线程的优先级进行控制并赋予延迟敏感型的线程高优先级来提升系统性能。文献[5]基于一种新型 DRAM 提出了一种延迟感知调度策略,通过对访存队列里的请求按照访存延迟进行优先级排序,使得总的访存时间下降。

上述方法是一系列提高存储子系统吞吐率的访存调度策略,但是没有考虑多核片上系统不同 IP 核同时访存带来的冲突加剧问题。

文献[6]提出了一种分阶段调度策略,将 IP 核访存请求分为 3 个阶段:同一页形成批量(batch),以批量为单位发送给访存调度器,根据不同 bank 发送给内存芯片。文献[7]对分阶段调度策略的前两步进行了修改,形成批量时考虑了 bank 级并行性。文献[8]进一步分析了分阶段调度策略的优势和潜力。文献[9]认为分阶段调度策略不能给延迟敏感的 IP 提供很好的服务质量,于是为每个 IP 访存地址按照 bank 分组形成一个虚队列,根据任务完成的进度和可用的 bank 从队列中选择一个请求发送给内存芯片。

上述方法通过减少访存冲突提高了多核片上系统的吞吐率,但是没有考虑多核片上系统不同 IP 核之间访存公平性问题。

文献[10]提出了一种分层次的访存调度策略,将访存调度分为应用间批处理调度、IP 核间批处理调度和 IP 核内批处理调度 3 个层次。文献[11]进一步对此方法进行了优化并加入电源门控来降低功耗。文献[12]提出一种服务质量感知的调度策略来动态调整 GPU 和 CPU 的优先级。文献[13]通过将得到内存芯片服务过多的请求 IP 核列入黑名单,使其在一段时间内不能得到服务来维护公平性。

这些策略考虑了 IP 核访存的公平性需求,但是由于吞吐率和公平性有一定的矛盾性,为了提高吞吐率,就必须让地址连续的高带宽 IP 核优先访存;为了提高公平性,就必须强制对每个 IP 按照一定比例要求进行访存。现有的访存调度策略多是在吞吐率和公平性之间进行权衡,并且只能侧重某一方面。不仅如此,现有的访存调度多是对访存队列中已有的访存请求进行优化调度,这种方式受限于访存队列的大小,由于面积时序功耗等方面的限制,访存队列不可能做得非常大,因此这种低层次的访存队列调度方式是一种被动调度策略,达到的效果也是比较有限的,很难满足多核片上系统的访存需求。

文献[14]提出了一种独占式访存调度,通过窗口仲裁使得不同 IP 核在某个时间片内独占访存带宽来充分利用连续流式访存的高效性。文献[15]在此基础上进一步提出了一种电源门控方法来降低 IP 核窗口关闭期间的漏电功耗。这种方法无法充分发挥内存控制器端访存队列调度的自由度和 DRAM 的 bank 级并行性,使得内存芯片的带宽无法得到充分利用。

本文充分参考和借鉴现有的多核片上系统访存调度优化方法,利用 IP 核的访存局部性和延迟容忍度以及 DRAM 的三维特性提出了一种全局主动访存调度优化方法,该方法综合了 IP 核的访存特性和访存队列调度的优势,有效提高了多核片上系统的访存带宽和服务质量。

2 多核片上系统访存特性

2.1 IP 核访存特性

多核片上系统主要访存 IP 核除了传统的 CPU 以外,一般包括图像处理单元(GPU)、高清视频解码器、显示控制器和网络控制器等。这些 IP 核的访存序列一般都有比较好的空间局部性,而访存需求

一般都有带宽需求量大和对延迟有一定的容忍度的特点。

显示控制器具有非常规律的访存行为,比如当其工作于双屏显示模式时,它从两个帧缓冲区中以固定的时间间隔轮流读取数据,它的内部一般都有数据缓冲区,用来存储提前取来的帧数据。由于支持预取操作,它对访存延迟有比较大的容忍度,只要访存调度系统能够在它内部数据消费完之前取来新的数据,就可以保障显示控制器对服务质量的需求。

图像处理单元(GPU)内部包含非常多的线程处理器执行单元,可以同时发出多个访存请求,通过多线程技术增加数据并行性,减少了对访存延迟的依赖。图像处理单元的多个线程往往会对同一块地址空间发出连续的访存请求,而在一段时间内,会形成对多个 bank 交错访问,但是对每个 bank 的连续访问往往落在同一个页中。

由以上分析可以看出,多媒体片上系统(system of chip, SoC)中的很多 IP 核都有预取机制和内部缓冲区来对读取的数据进行缓存,工作中只需要在缓存数据使用完毕前补充新的数据即可,因此这些 IP 核对访存延迟的要求普遍不高。

从某个 IP 核具体的访存请求序列来看,这些 IP 核访存序列普遍具有较好的空间局部性,这主要是由于这些多媒体应用多是针对声音和图像的处理,而这些信息通常按照顺序存储在连续的内存地址空间中,因此在对其访问时,体现为同一内存页的连续访问命中。这些 IP 核单独工作时体现了良好的空间局部性,但是多媒体应用多是很多 IP 配合完成工作的,比如上面提到的视频解码器、GPU 和显示控制器(display controller, DC)就是高清视频播放等应用中必不可少的,当这些 IP 核同时发生访存行为时,会产生大量的访存冲突,使得总的访存带宽严重下降。

综上所述,多核片上系统中的 IP 核普遍属于带宽敏感、延迟不敏感的访存类型,虽然每个模块单独工作具有良好的空间局部性,对内存非常友好,但是多个模块同时工作时会产生严重的访存冲突,使得访存带宽显著下降。

2.2　访存序列顺序影响

为了说明不同访存序列对内存带宽和延迟的影响,图 1 列举了一个实际应用场景中的示例。该示例中有 2 个 IP 核读访存请求序列,每个 IP 核发出 2 个读请求(burst length = 8),其中 IP1 的 2 个读访存地址(bank, row, column)分别为(0,0,0)和(0,0,1),IP2 的 2 个读访存地址分别为(0,1,0)和(0,1,1),也就是说 2 个 IP 对同一个 bank 的不同页进行读访问请求。图 1(a)中来自同一个 IP 的读请求采用连续访存,图1(b)的区别是2个IP的读请求交

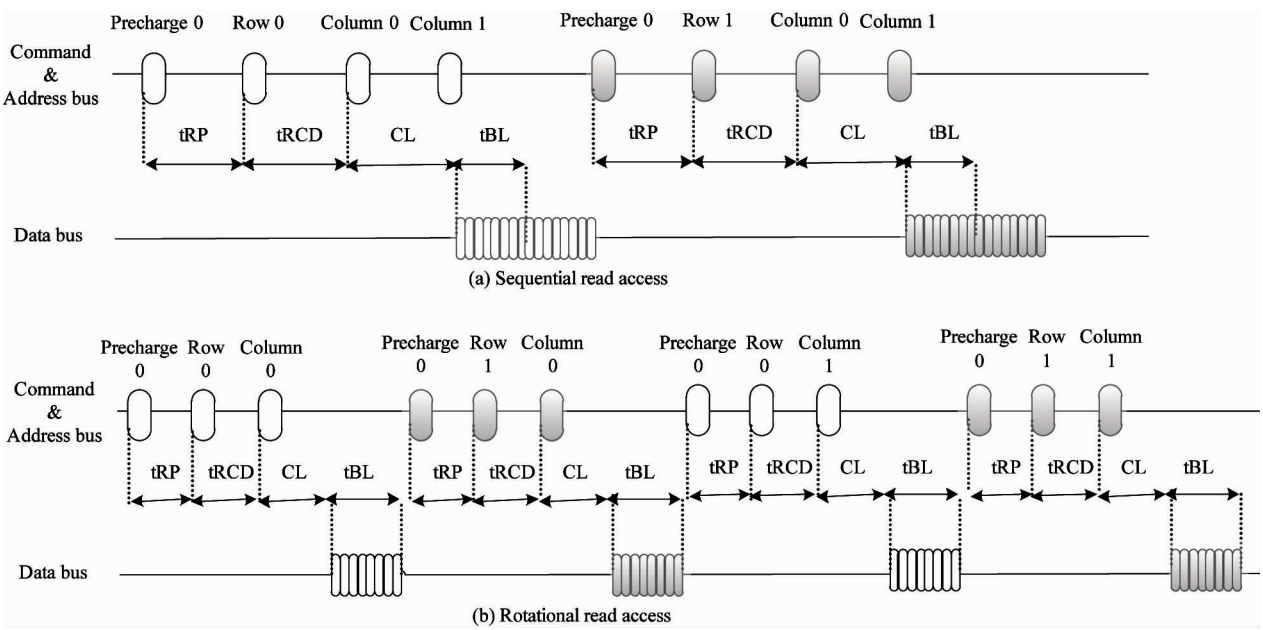

图 1　连续读访存和交错读访存对比

错访问内存。结果是由于图 1(b)中每一次访存都需要进行预充电和页激活操作,因此大大增加了访存延迟。当内存时序参数为 5/5/5/15(CL-tRCD-tRP-tRAS)时,图 1(a)和图 1(b)消耗的时钟周期分别为 46 和 79。由此可见,连续和交错读访存 2 种不同的调度方式对内存性能产生了很明显的区别,这只是 2 个读请求连续的情况,当更多的请求连续时,与交错访存相比,总的访存带宽相差超过 3 倍,将体现出更加明显的访存性能优势。

2.3 设计思路

多核片上系统中多个 IP 核共享访存通路,任何单一的 IP 核都不可能一直独占内存总线,因为任何 IP 核都不是每个时钟周期都会有访存请求。理想情况下,在一个特定的访存采样周期内,一个 IP 核获得的访存服务时间比例应该与其带宽需求比例相符,即

$$\frac{\text{访存时间}}{\text{采样周期}}=\frac{\text{需求带宽}}{\text{总带宽}} \tag{1}$$

已有的面向服务质量的访存调度策略多是以式(1)为设计目标,因为这样可以保障不同 IP 核访存的公平性。但是现实中多数访存调度方法并没有利用 IP 核对访存延迟有一定容忍度的特点,这些策略很少从源头上调整控制不同 IP 核的访存顺序,它们对访存顺序的调整多是针对已经进入访存队列或者请求端口的访存序列,而由于功耗面积等物理设计因素的限制,访存队列不可能容纳过多的访存请求,因此这些策略只能采取被动的方式,对有限的访存序列进行微调,难以应对多核片上系统中多 IP 核的复杂访存应用场景需求。

本文以使用先入先出(first input first output, FIFO)存储器作为内部数据缓存的 IP 核来说明其对延迟有一定容忍度的原因。这种 IP 核一般都是采用的贪心策略,总是趋向于将 FIFO 填满以应对随时变化的带宽和延迟,因此一旦 FIFO 中有空间可以接纳新的数据,就会发出新的访存请求。由于需求带宽小于总的访存带宽,这些请求会在时间上离散分布于整个工作周期。当存在多个 IP 核时,不同 IP 核离散的访存请求交错在一起,会产生严重的访存冲突,而且这种冲突难以在较小的时间维度进行优化调度,这也是现有访存调度方法难以直接运用的原因。另一方面,当 FIFO 存储器中的数据即将耗尽时,该 IP 核将连续发出访存请求,这些访存请求一般有很好的空间局部性,如果让其独享命中 bank 的访存带宽则会很好地提高访存性能。

基于以上的分析,如果对来自不同 IP 核的访存请求进行主动的重排序,比如将同一个 IP 核的请求连在一起访存,可以在保证服务质量的同时优化访存性能。文献[14]提出的独占式访存调度便是通过不同 IP 核在某个时间片内独占访存带宽来提高访存性能,但是没有考虑不同 IP 核间访存的 bank 级并行性。前面提到,任何一个 IP 核在一小段时间内只会访问有限的几个 bank,因此当多个 IP 核同时访存时,它们之间如果访问同一个 bank 的不同页,则显然是冲突的访问,应当采取相应的调度策略给予优化。但是如果访问的是不同的 bank,则应该充分利用内存芯片的 bank 级并行性,不应对其访存进行限制。

3 全局主动式访存调度器

本节提出的全局主动式访存调度器利用 IP 核访存的空间局部性和延迟容忍度以及 DRAM 的三维特性来优化访存性能,进而提高服务质量。下面介绍全局主动式调度方式和实现中的一些关键点。

3.1 调度器结构

图 2 所示为全局主动式访存调度器结构示意图。该调度器主要由 4 部分构成:访存请求仲裁器、DDR 状态记录表、IP 软件配置寄存器和辅助仲裁器。

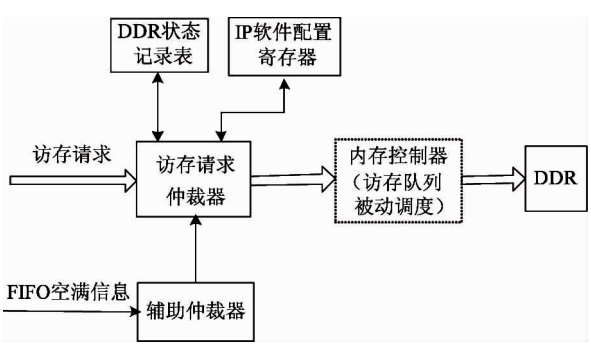

图 2 全局主动式访存调度器结构示意图

访存请求仲裁器是完成访存调度的核心模块,用来接收来自不同 IP 核的访存请求,并且通过其他模块提供的信息决定让哪些 IP 核的访存请求通过或者等待,相当于一个开关的作用。具体仲裁的方法是,当一个 IP 核发出访存请求时,首先提取该请求地址中包含的 bank 和 row 信息,然后与 DDR 状态记录表中记录的 bank 和 row 信息进行比较,当满足以下条件之一时允许该请求通过:

(1) 状态记录表中该 bank 没有记录任何 IP 核编号(identification, ID)值,也就是说当前没有其他 IP 核对该 bank 进行访存,该 bank 处于空闲状态;

(2) 状态记录表中该 bank 记录的 ID 值与该 IP 的 ID 不一致,但是该 bank 记录的打开 row 信息与该访存请求一致,这意味着其他 IP 核正在对该 bank 进行集中式访存,但是该 IP 恰好也对相同 bank 相同的页进行访存,此时这两个 IP 核同时访存并没有任何冲突,这种情况比较少见;

(3) 状态记录表中该 bank 记录的 ID 值与该 IP 的 ID 不一致,但是根据调度策略,此时需要对该 bank 使用权轮转,轮到该 IP 使用;

(4) 状态记录表中该 bank 记录的 ID 值与该 IP 的 ID 一致,且根据调度策略,此时未到对该 bank 使用权轮转时;

(5) 辅助仲裁器发出对该 IP 核的强制访存指示,此时需要强制中断其他 IP 核对该 bank 的使用权,令该 IP 核获得该 bank 的使用权。

当不满足以上条件时,说明该 IP 核的访存请求与其他 IP 核存在访存冲突,需要等待直到获得该 bank 的使用权。

由于不排除某些 IP 核会在某些特定场景中发出地址离散的请求,为了排除这种干扰,本文在访存请求仲裁器内部为每个访存的 IP 核设置了一个 4 项的访存请求地址记录表,用来对每个 IP 核的请求地址进行实时监测。只有最近 4 项请求为同一个 bank 页内的请求时才对该 IP 核应用主动调度,否则认为该 IP 核访存地址不连续,则不对其访存请求进行主动访存调度,让其自由通过仲裁器,通过后端常规的访存队列被动调度对其进行相应优化。

DDR 状态记录表记录了不同 bank 此时打开的页信息以及此时使用该 bank 的 IP 核的 ID 值,当没有任何 IP 核访问某一 bank 时,需要将该 bank 记录的 row 和 ID 值复位清零,表示该 bank 处于空闲状态,可以随时接受访问。当有新的 IP 核获得该 bank 的使用权时,需要记录对应的 row 和 ID 值,用来给访存请求仲裁器的仲裁提供指导。

IP 软件配置寄存器可以通过软件配置,它里面保存的是不同 IP 核正常工作时的访存带宽和访存周期参考值。这些值可以在 IP 核单独工作时获取,通过软件写入寄存器,在多个 IP 核同时工作时可以为访存请求仲裁器提供参考,具体的过程将在下一小节中介绍。

辅助仲裁器用来接收 IP 核提供的 FIFO 空满信息,用来对访存请求仲裁器的仲裁提供更加准确的依据。很多 IP 核都可以输出内部 FIFO 的使用情况信息,比如显示控制器、GPU 等。这些信息的提供可以不用依赖 IP 软件配置寄存器中提供的带宽值和访存周期来推测 FIFO 的空满和 IP 核的饥饿程度,因为依据 IP 软件配置寄存器做出的推测会随着时间变得不准确。因此对于可以提供内部 FIFO 缓存使用信息的 IP 核,可以利用这些信息更加准确地指导访存请求仲裁器,同时对 IP 软件配置寄存器中配置的值进行实时修正。

3.2　调度器实现关键点

前面提到访存请求仲裁器是该全局主动式访存调度器中最核心的部分,因为冲突访存请求的仲裁是在该模块完成的,而如何实现仲裁则变得非常关键。在这个访存请求仲裁器中采用了 3 种策略,这 3 种策略有机结合,互相配合,共同完成访存请求仲裁器的精确仲裁工作。

(1) 实时带宽仲裁。当某个 IP 核刚刚获得访存权限时,由于它之前由于冲突已经被阻塞了一段时间,所以内部 FIFO 相对比较空,需要请求大量的数据,所以会发出大量连续的访存请求。此时开始计算 IP 核在整个调度时间窗口内的访存带宽,当其带宽值等于 IP 软件配置寄存器中配置的需求值时,说明该 IP 核在开启的服务窗口时间片内获得的服务相当于整个调度时间窗口内的需求,此时说明该 IP 核内部 FIFO 已经填充比较满,恢复成正常工作

状态。此时如果有其他 IP 核需要对该 bank 访存，则可以终止该 IP 核的访存权限，使它利用 FIFO 中已有的数据继续工作。需要说明的是，这里的实时带宽统计的是已经发出的请求带宽，并非实际收到的数据带宽，这是因为根据请求可以得到返回的数据量，而等待数据返回需要一定的访存延迟，无法满足该调度方法实时性的要求。

（2）软件配置参考。由于多 IP 核访存的不可控性，当实时带宽经过很长时间仍未达到正常访存带宽值时，此时可能是因为系统突然的访存压力增大导致的，如果继续给予该 IP 核访存权限则很可能其他等候的 IP 核会出现数据饥饿的问题，因此当该 IP 核访存时间超过 IP 带宽配置寄存器提供的访存周期参考值时，需要中断该 IP 核对内存的使用权，防止其他 IP 核出现饥饿导致不能正常工作的情况出现。访存周期参考值的设置需要整体考量，它反映的是访存调度的粒度，设置较大值可以使得同一个 IP 核获得较长时间的内存权限，更好地利用空间局部性，但是很可能会使得某些 IP 核出现饥饿问题。设置较小值可以加速访存冲突 IP 核的轮转，有效避免数据饥饿问题，但是对访存的空间局部性有一定损害。这就需要设计者对整个系统全盘考虑，满足缓存延迟忍耐度最低的 IP 核的访存需求。

（3）辅助仲裁控制。辅助仲裁器可以获得多个 IP 核精准的 FIFO 空满信息，因此它可以为访存仲裁提供最后一道精确的保障，当某个 IP 核显示内部缓存 FIFO 数据即将消耗殆尽时，需要该辅助仲裁器对访存请求仲裁器发出强制访存命令信号，立刻终止其他 IP 核对该 bank 的使用权，满足该 IP 核的访存需求。

当 CPU 出现在访存调度层次时，由于其对访存延迟非常敏感，因此不对其做任何仲裁，直接让其穿过访存调度器访问内存，也不对其访存地址信息做任何记录。

对于如何从多个冲突 IP 核中选择一个给予访存权限，本文采用了轮转的策略，当辅助仲裁器没有给出强制访存命令信号时，轮转策略给予等待最久的 IP 核优先访存权限，相当于让饥饿程度可能最高的 IP 核优先访存。

本方法实现的硬件开销非常小，只是在顶层对来自不同 IP 核的访存请求进行仲裁调度，虽然会增加访存冲突 IP 核等待期间的延迟，但是由于 IP 核的延迟容忍度，后续连续访存完全能够弥补等待期间的损失，因此不会造成服务质量下降。由于不需要增加任何访存队列，也不会增加后续访存通路的延迟，硬件实现方法非常简单。

本方法与内存控制器访存队列被动调度策略完全具有正交性，可以与其配合使用，由于本方法已经在宏观层面对访存序列进行了优化，可以提供给内存控制器非常友好的访存序列，因此可以达到很好的访存性能。

4　实验平台和实验结果

4.1　实验平台

本文的实验平台基于龙芯 2K1000 处理器的仿真环境，龙芯 2K1000 处理器是 2017 年发布的龙芯 2 号处理器系列的最新产品，是一款主要面向网络应用的高性能多媒体 SoC，结构如图 3 所示。

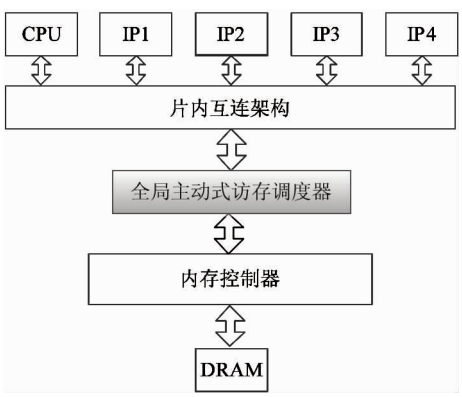

图 3　龙芯 2K1000 仿真环境结构

图 3 中的互联架构使用了 AMBA AXI 总线协议；内存控制器为龙芯自主研发的高性能内存控制器，读写命令队列分别有 16 项，访存队列调度策略采用经典的 FR-FCFS 策略；DRAM 颗粒采用了 DDR3-800 内存标准，时序参数为 5/5/5/15，数据总线宽度为 64 bit，提供的最大带宽为 6.4 GB/s。

本实验平台共设置了 5 个仿真 IP 核：1 个模拟 CPU 访存行为，该核在随机的访存间隔内发出随机

地址访存请求,平均访存带宽为 400 MB/s;其他 4 个模拟多核片上系统中其他访存的 IP 核,每个 IP 核的访存地址按照顺序发出,用于模拟访存空间局部性,使得同一个 IP 核的访存请求尽量在同一个页中命中。每个仿真 IP 核均可对访存地址、带宽需求以及 outstanding 数量进行配置。

为了研究不同应用场景下访存流带宽和冲突对多核片上系统的性能影响,本实验设置了 5 组不同的带宽需求,如表 1 所示,其工作负载量逐渐增加,并且每组都进行不同数量访存冲突的测试,以便研究访存流存在冲突时使用不同调度策略对性能的影响。

表 1　IP 核的带宽需求(MB/s)

工作负载	IP1	IP2	IP3	IP4
BW1	100	100	100	100
BW2	200	200	200	200
BW3	400	400	400	400
BW4	800	800	800	800
BW5	1200	1200	1200	1200

4.2　实验结果

图 4 展示了使用 3 种不同的主动访存调度策略在不同冲突数量下所能达到的最大 IP 访存带宽,其中使用了 3 种主动调度策略进行对比:In turn 表示使用龙芯 2K1000 自带的轮转调度策略,该策略实际上并未进行有效的主动调度,只是在 IP 核同时存在访存请求时采用轮流调度策略;EMS 表示使用文献[14]中提出的独占式访存调度策略;GPMS 表示使用本文中的全局主动式访存调度策略。与3种主动调度策略配合使用的访存队列被动调度策略均为 FR-FCFS 策略。横坐标表示存在访存冲突的 IP 核数,由于使用了 4 个访存 IP 核,所以一共测试了 4 种访存冲突的情况。

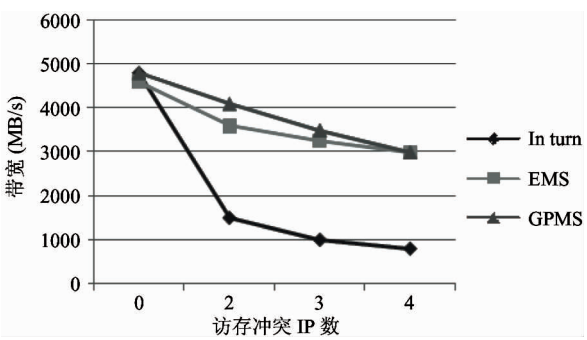

图 4　不同场景下的 IP 最大访存带宽

从图中可以看出,当 IP 核之间不存在访存冲突时,3 种访存调度策略能够达到的最大访存带宽几乎相同。

当 2 个 IP 核间存在访存冲突时,3 种调度策略所能达到的最大访存带宽出现显著区别:简单轮转调度策略出现剧烈下降,下降幅度达到了将近 70%,EMS 和 GPMS 策略也出现了下降,但是下降的幅度明显缓和了许多,分别达到 22% 和 14%。由此可见在顶层进行访存调度的重要性,虽然本文系列的控制器调度队列有多达 16 项,但是在仅仅有 2 个 IP 核存在访存冲突时则会导致访存性能的剧烈下降。虽然本文的内存控制器也在访存队列层次进行了重排序等一系列的调度,但是这种微观层次的调度是比较低效的,实际取得的效果远远不如在顶层进行访存调度。同时可以看到,GPMS 策略也优于 EMS 策略,这一方面是因为 EMS 策略使用简单的时间窗口轮转调度策略,在一个时刻只有一个 IP 能够访存,这牺牲了内存控制器层面利用非冲突的访存请求进行性能优化的机会,而 GPMS 策略则只是对存在访存冲突的 IP 核进行轮转调度,而非冲突的 IP 核则不受影响;另一方面是因为 EMS 策略将一个调度周期分配给所有的访存 IP 核,每个 IP 核获得的访存时间片相对较少,因此冲突 IP 间的轮转次数相对较多,而 GPMS 策略则只是将一个调度周期分配给存在访存冲突的 IP,因此冲突 IP 核能够获得的访存时间片更长,而非冲突 IP 核能够获得整个调度周期,因此访存的连续性较好,访存性能也更高。

随着访存冲突 IP 数的进一步增加,3 种调度策略所能达到的最大访存带宽进一步下降,但是 EMS 和 GPMS 策略仍然显著优于简单轮转调度策略。同时可以看到,EMS 和 GPMS 策略之间的差别逐渐缩小,在访存冲突 IP 核为 4 时两者的实际效果几乎相同。这是因为随着冲突 IP 核的增加,GPMS 策略中需要进行访存调度的 IP 核增加,每个冲突 IP 核在 1

个调度周期内能够获得的访存时间减少,因此实际访存性能越来越接近 EMS 策略。在访存冲突 IP 核为 4 时,GPMS 策略退化为 EMS 策略,因此两者的实际带宽几乎相同。

图 5 展示了使用龙芯 2K1000 自带的简单轮流仲裁调度方式在不同应用场景下的 CPU 访存延迟。

从图中可以看出,在 IP 核带宽需求很低时(BW1),CPU 访存延迟随着冲突 IP 数量的增加逐渐增大,但是并未出现剧烈变化。但是从 BW2 开始,CPU 访存延迟随着冲突 IP 数量的增加而迅速增加,在冲突 IP 数量为 4 时,此时的带宽需求已经接近内存能够提供的最大带宽,在 CPU 访存延迟上已经超过 200 个时钟周期。由此可见,随着负载的增加,访存冲突导致访存队列迅速增长,访存延迟也剧烈增大。当负载进一步增加时,此时的内存已经不能满足存在冲突时的带宽需求,CPU 访存延迟在不同冲突数量时稳定在一个相对很大的值,此时系统的访存性能也维持在一个非常低的状态。

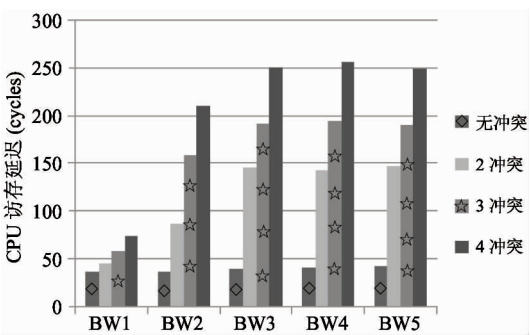

图 5　简单轮转调度策略下的 CPU 访存延迟

图 6 展示了使用独占式访存调度方式在不同应用场景下的 CPU 访存延迟。

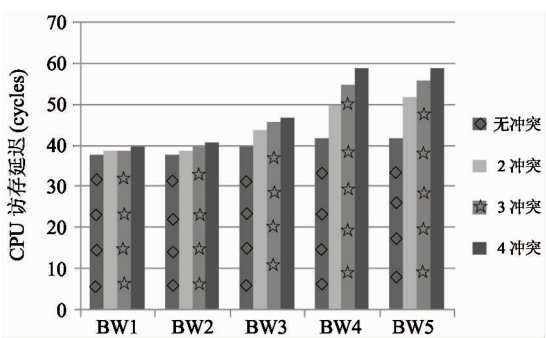

图 6　独占式访存调度策略下的 CPU 访存延迟

从图中可以看出,当负载较低时(比如 BW1 和 BW2),CPU 访存延迟随着冲突 IP 数量的增加并没有明显变化,由此可见连续地址访存的高效性,此时 CPU 能够提供的带宽远大于需求,因此访存请求可以得到及时响应。当负载进一步增加时,CPU 访存延迟随着冲突 IP 数量的增加开始出现变化,这主要是因为带宽需求逐渐接近和超过内存能够提供的最大带宽,因此访存队列也迅速增长,访存延迟也相应增加。但是与图 5 明显不同的是,CPU 访存延迟在不同冲突数量时稳定在一个相对较低的值,均不超过 60 个时钟周期,此时系统的访存性能维持在一个较高的状态。

图 7 展示了使用全局主动式访存调度方式在不同应用场景下的 CPU 访存延迟。

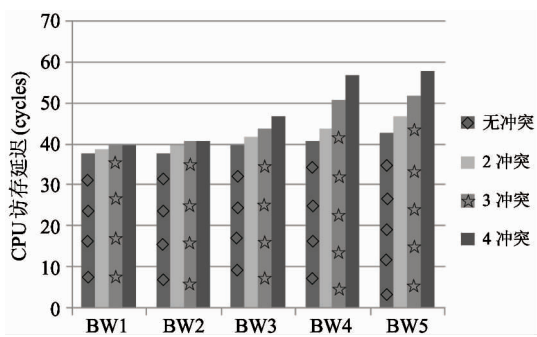

图 7　全局主动式访存调度策略下的 CPU 访存延迟

从图中可以看出,当负载较低时,CPU 访存延迟与使用 EMS 策略的结果几乎相同,也没有明显的变化。随着负载的进一步增加,两种调度策略的区别开始体现,主要是在冲突 IP 数量为 2 和 3 时,使用 GPMS 策略的 CPU 访存延迟略小于 EMS 策略的访存延迟,这主要是由于使用 GPMS 策略可以更好地利用内存芯片的 bank 级并行性,给予内存控制器更大的调度优化空间,同时也增加了冲突 IP 核单独连续访存的时间,减少了访存冲突导致的 DRAM 中页开关的次数。

为了进一步研究在顶层进行全局主动访存调度相比在内存控制器访存队列层次进行被动调度的优势,本文进一步设置了对比实验。由于所使用的内存控制器读访存队列为 16 项,理论上 FR-FCFS 调度策略可以对 16 个访存请求进行优化调度,为了进

行对比,将 GPMS 策略中的调度窗口也设置为 16 项,为了排除其他非冲突访存请求的干扰,只设置 IP 核发出冲突的访存请求,对不同冲突 IP 数量进行了测试(比如 2 冲突时我们只让 2 个 IP 核工作,每个 IP 核在调度窗口内发出 8 个访存请求;3 冲突时则让 3 个 IP 核工作,每个 IP 核在调度窗口内发出 5 个访存请求(近似 16 项);最多 4 冲突时每个 IP 核在调度窗口内发出 4 个访存请求)。为了消除内存控制器调度优化的干扰,在使用 GPMS 策略时将内存控制器调度关闭,直接采用按序访存。为了研究增大访存队列对性能的影响,进行了另一个对比实验,实验中将访存队列增加为 24 项,其他配置参数完全相同,进行与上面类似的实验。两种访存调度方式下的最大访存带宽如图 8 所示(由于当所有 IP 核都存在访存冲突时 GPMS 策略会退化为 EMS 策略,也即此时无法利用 DRAM 的 bank 级并行性,因此两种策略性能相同)。

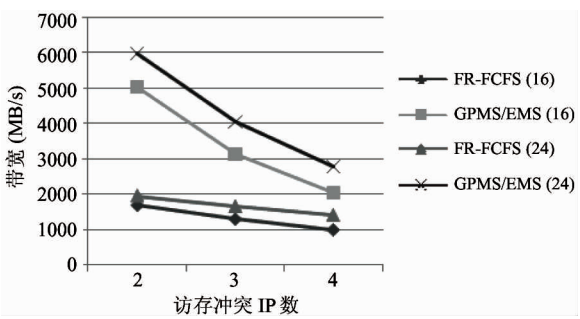

图 8　不同冲突 IP 数量在两种访存调度方式下的最大访存带宽

从图中可以看出,虽然理论上两种调度方式可以达到相同的效果,但是实际访存带宽却相差很大。比如在访存队列为 16 项时,当访存冲突 IP 数为 2 时,GPMS 策略达到的实际带宽是 FR-FCFS 策略的 3 倍,而冲突 IP 数为 4 时,前者也是后者的 2 倍之多。由此可见内存控制器端被动访存调度的低效性,虽然两种策略理论上能够达到相同的效果,但是由于访存队列内的调度实际上是动态进行的,并不能做到获得一个完整的访存序列再进行决策,而且无法获得 IP 核的相关信息,因此实际效果是非常差的。即使增加访存队列的长度,这只能从一定程度上缓解访存冲突的问题,但是效果比较有限。比如在使用 24 项的访存队列时,虽然将访存队列增大了 50%,但是实际增加的访存带宽平均只有 23%,用了很大的硬件代价换来了很小的收益。而当通过 GPMS 策略进行访存调度时,只需要将调度的窗口变大就可以达到相同的效果,比如 3 冲突时将每个访存调度窗口从 16 项增加到 24 项,实际访存带宽也增加了 26%,而这种改变并不需要增加硬件开销,因此是非常高效的。

综上所述,全局主动访存调度策略可以有效减少多核片上系统中存在的多 IP 核访存冲突,有效提高系统的访存性能,保障了系统的服务质量。

5　结　论

本文总结了多核片上系统访存调度方法,并在此基础上提出了一种全局主动访存调度方法。该方法通过限制访存冲突的 IP 核使其在一个调度窗口内分别连续访问内存,从而减少访存冲突次数,降低内存控制器端访存调度的压力;同时不存在访存冲突的 IP 核在调度窗口内一直保持内存的使用权,从而可以充分发挥内存控制器端访存队列调度的自由度,也充分利用了 DRAM 的 bank 级并行性。实验表明,当 IP 核间访存冲突严重时,该方法相比访存队列调度方式可以提升 1 到 2 倍的访存带宽,CPU 访存延迟在不同冲突数量和负载下也有不同程度的大幅下降。

全局主动式访存调度方法要求设计者全盘考虑片上系统各个 IP 核的配置对性能的影响。未来的工作包括进一步研究各访存设备的访存行为,比如从软件层面获取更多的信息,从而更加精确地指导调度策略;此外还要对带宽需求变化剧烈以及地址离散的访存 IP 核做进一步研究,对访存调度实时性做出更加精确的判断,进一步提升系统的访存性能。

参考文献

[1] Rixner S, Dally W J, Kapasi U J. Memory access scheduling[C]. In: Proceedings of the 27th International Symposium on Computer Architecture, Vancouver, Canada, 2000. 128-138

[2] Shao J, Davis B T. A Burst scheduling access reordering

mechanism[C]. In: Proceedings of the 2007 IEEE 13th International Symposium on High Performance Computer Architecture, Scottsdale, USA, 2007. 285-294
[3] Guo T P, Li L, Guo D O. Design and implementation of a DDR3-based memory controller[C]. In: Proceedings of the 3rd International Conference on Intelligent System Design and Engineering Applications, Hong Kong, China, 2013. 540-543
[4] Xiong D, Huang K, Jiang X, et al. Memory access scheduling based on dynamic multilevel priority in shared DRAM systems[J]. *ACM Transactions on Architecture & Code Optimization*, 2016, 13(4):42
[5] Liu W, Huang P, Tang K, et al. LAMS: a latency-aware memory scheduling policy for modern DRAM systems [C]. In: Proceedings of the 2016 IEEE 35th International Performance Computing and Communications Conference, Las Vegas, USA, 2016. 1-8
[6] Ausavarungnirun R, Chang K W, Subramanian L, et al. Staged memory scheduling: achieving high performance and scalability in heterogeneous systems[C]. In: Proceedings of the 2012 39th Annual International Symposium on Computer Architecture (ISCA), Portland, USA, 2012. 416-427
[7] Lu C H, Chao H L, Song Y C, et al. Interference-aware Batch Memory Scheduling in heterogeneous multicore architecture[C]. In: Proceedings of the 2016 International Symposium on VLSI Design, Automation and Test, Hsinchu, China, 2016. 1-4
[8] Ausavarungnirun R, Loh G H, Subramanian L, et al. High-performance and energy-effcient memory scheduler design for heterogeneous systems[J]. *arXiv*: 1804.1103, 2018
[9] Song Y, Samadi K, Lin B. Single-tier virtual queuing: an efficacious memory controller architecture for MPSoCs with multiple realtime cores[C]. In: Proceedings of the 2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC), Austin, USA, 2016. 1-6
[10] Lin Y J, Yang C L, Lin T J. Hierarchical memory scheduling for multimedia MPSoCs[C]. In: Proceedings of the 2010 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San Jose, USA, 2010. 190-196
[11] Lin Y J, Yang C L, Huang J W, et al. System-level performance and power optimization for MPSoC: a memory access-aware approach[J]. *ACM Transactions on Embedded Computing Systems*, 2015, 14(1):1-26
[12] Jeong M K, Erez M, Sudanthi C. A QoS-aware memory controller for dynamically balancing GPU and CPU bandwidth use in an MPSoC[C]. In: Proceedings of the DAC Design Automation Conference 2012, San Francisco, USA, 2012. 850-855
[13] Subramanian L, Lee D, Seshadri V. BLISS: balancing performance, fairness and complexity in memory access scheduling[J]. *IEEE Transactions on Parallel and Distributed Systems*, 2016, 27(10): 3071-3087
[14] Liu S, Su M, Wu R. Exclusive memory scheduling for multimedia MPSoC[C]. In: Proceedings of the 2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing, Zhangjiajie, China, 2013. 2022-2026
[15] 刘苏, 苏孟豪, 苏文. 基于独占式访存调度的片上系统电源门控方法[J]. 高技术通讯, 2014, 24(3): 256-262

Global proactive memory scheduling on multi-processor system-on-chip

Li Peng [*] [**] [***], Zeng Lu [****], Wang Huandong [****], Zhang Longbing [***]
([*] State Key Laboratory of Computer Architecture (Institute of Computing Technology, Chinese Academy of Sciences), Beijing 100190)
([**] Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)
([***] University of Chinese Academy of Sciences, Beijing 100049)
([****] Loongson Technology Corporation Limited, Beijing 100095)

Abstract

A global proactive memory scheduling method is proposed for multi-processor system-on-chip (MPSoC) to improve memory performance. This method uses the memory access locality and delay tolerance of IP (intellectual property) cores. By restricting the conflicting IP cores, they continuously access the memory within a scheduling window, thereby reducing the number of memory conflicts. At the same time, IP cores that do not have memory access conflicts maintain the memory usage rights within the scheduling window. In this way, the freedom of memory scheduling queue on memory controller side and bank-level parallelism of DRAM can be fully utilized. The experiment results show that when the IP core memory access conflict is serious, the method can increase the memory bandwidth by 1 to 2 times compared with the memory scheduling queue method.

Key words: multi-processor system-on-chip (MPSoC), memory scheduling, memory access locality, delay tolerance, quality of service