(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 110457238 A
(43)申请公布日 2019.11.15

(21)申请号 201910601175.0

(22)申请日 2019.07.04

(71)申请人 中国民航大学
地址 300300 天津市东丽区津北公路2898号

(72)发明人 李炳超

(74)专利代理机构 天津市北洋有限责任专利代理事务所 12201
代理人 李林娟

(51)Int.Cl.
G06F 12/128(2016.01)
G06F 12/0811(2016.01)
G06F 12/0842(2016.01)

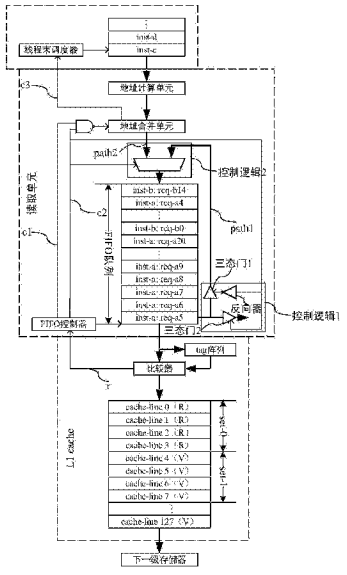
权利要求书2页 说明书6页 附图3页

(54)发明名称

减缓GPU访存请求及指令访问cache时停顿的方法

(57)摘要

本发明公开了一种减缓GPU访存请求及指令访问cache时停顿的方法，所述方法包括：位于FIFO队首的访存请求访问L1 cache，将访存请求的tag与L1 cache中的tag进行比较，若存在发生保留停顿的访存请求，将该访存请求从FIFO队首弹出，放入FIFO队尾；通过数据通路连通队首和队尾、以及第一控制逻辑控制访存请求从FIFO队首弹出之后的走向；构建第二控制逻辑及第一、三控制信号，用于对线程束调度器和读取单元之间的访存指令进行流水处理，使得当读取单元中的地址合并单元将所有的访存请求都合并完毕时便可处理下一条访存指令，并且当FIFO队列中有空闲条目时即可生成访存请求并存入。与现有的技术相比，本发明能够减少访存请求的停顿时间，提高访存请求的处理速度，同时也能够减少访存指令的等待时间，提高访存指令的处理速度。



CN 110457238 A

1.一种减缓GPU访存请求及指令访问cache时停顿的方法，其特征在于，所述方法包括：

位于FIFO队首的访存请求访问L1 cache，将访存请求的tag与L1 cache中的tag进行比较，若存在发生保留停顿的访存请求，将该访存请求从FIFO队首弹出，放入FIFO队尾；

通过数据通路连通队首和队尾、以及第一控制逻辑控制访存请求从FIFO队首弹出之后的走向；

构建第二控制逻辑及第一、三控制信号，用于对线程束调度器和读取单元之间的访存指令进行流水处理，使得当读取单元中的地址合并单元将所有的访存请求都合并完毕时便可处理下一条访存指令，并且当FIFO队列中有空闲条目时即可生成访存请求并存入。

2.根据权利要求1所述的一种减缓GPU访存请求及指令访问cache时停顿的方法，其特征在于，所述第一控制逻辑控制访存请求从FIFO队首弹出之后的走向具体为：

当访存请求在L1 cache中的访问结果为保留停顿时，第二控制信号为假，通过反向器之后为真；

第一三态门呈导通状态，第二三态门呈高阻状态，表示访存请求从FIFO队首弹出之后被传输至FIFO队尾。

3.根据权利要求1所述的一种减缓GPU访存请求及指令访问cache时停顿的方法，其特征在于，所述构建第二控制逻辑及第一、三控制信号，用于对线程束调度器和读取单元之间的访存指令进行流水处理具体为：

1)若访存请求未全部被地址合并单元合成完毕，则第三控制信号为假，通知线程束调度器不能将其他访存指令发送到读取单元；否则，第三控制信号为真，通知线程束调度器将其他访存指令发送到读取单元；

2)将FIFO队列是否满的状态通过第一控制信号发送给地址合并单元从而控制访存请求的生成。

4.根据权利要求3所述的一种减缓GPU访存请求及指令访问cache时停顿的方法，其特征在于，所述将FIFO队列是否满的状态通过第一控制信号发送给地址合并单元从而控制访存请求的生成具体为：

如果FIFO队列变满，则第一控制信号为假，通知地址合并单元暂停对访存请求进行合并，直到FIFO队列中有空闲的条目为止；

否则，通知地址合并单元继续对访存请求进行合并、并放入FIFO队尾。

5.根据权利要求1-4中任一权利要求所述的一种减缓GPU访存请求及指令访问cache时停顿的方法，其特征在于，所述方法还包括：

对地址合并单元生成的访存请求、从FIFO队首弹出的访存请求在同一周期放入FIFO队尾时的冲突处理。

6.根据权利要求5所述的一种减缓GPU访存请求及指令访问cache时停顿的方法，其特征在于，所述冲突处理具体为：

赋予从FIFO队首弹出的访存请求高优先级，通过第二控制逻辑将访存请求放入FIFO队尾；

地址合并单元暂停生成新的访存请求，直到没有从FIFO队首弹出的访存请求需要放入FIFO队尾为止。

7.根据权利要求6所述的一种减缓GPU访存请求及指令访问cache时停顿的方法，其特

征在于，所述通过第二控制逻辑将访存请求放入FIFO队尾为：

当访存请求在L1 cache中的访问结果为命中或缺失时，第二控制信号为真，选通多路选择器的输入path2，将地址合并单元生成的访存请求放入FIFO队尾；

当访存请求在L1 cache中的访问结果为保留停顿时，第二控制信号为假，选通多路选择器的输入path1，将从FIFO队首弹出的访存请求放入FIFO队尾。

减缓GPU访存请求及指令访问cache时停顿的方法

技术领域

[0001] 本发明涉及GPU(图形处理器)cache(高速缓冲存储器)体系结构领域,尤其涉及一种减缓GPU访存请求和访存指令在访问L1 cache(一级高速缓冲存储器)时发生停顿的处理方法。

背景技术

[0002] 近年来,GPU已经发展成一种多线程的高性能并行通用计算平台,并且GPU的计算能力仍然在快速提高,吸引着越来越多的应用程序在GPU上进行加速。

[0003] 在GPU软件层面,应用程序在GPU上运行时,首先需要将应用程序的任务细分为若干个可以独立运行的线程,再将多个线程组织成线程块。在GPU硬件层面,一个GPU由若干流多处理器、片内互联网络以及存储器构成。流多处理器内部具有支持多线程并行运行的寄存器文件,标量处理器,读写单元,共享存储器,cache等。线程以线程块为单位分别被发送到各个流多处理器上面,在流多处理器内部,线程块又被硬件细分为线程束,线程束是GPU的最基本的执行单元[1]。在NVIDIA公司的GPU中,一个线程束由32个线程组成,这32个线程会被并行执行。

[0004] 当线程束执行访存指令时,每个线程都会产生一个访存请求,为了减少访存请求数量,在GPU的流多处理器内部通过地址合并单元对同一线程束所产生的访存请求进行合并。如果一个线程束所产生的访存请求所访问的地址位于同一数据块(如128字节)内,则可以将这些访存请求合并成一个访存请求[2]。但是,由于某些程序的访存特征具有不规则性,即使经过地址合并之后,一个线程束的访存指令仍然具有多个访存请求,这些访存请求会被放入一个FIFO(先入先出)队列里,对cache造成突发式的访问。另一方面,由于流多处理器内部的cache容量较小(16KB-96KB),而线程数量能达到上千个,每个线程的平均cache容量只有几十字节,导致cache的缺失率非常高。当访存请求发生cache缺失时,根据相应的替换策略,会在cache中选择一个cache-line(高速缓存行)替换其中的数据,然后该访存请求会继续访问下一级存储器(L2 cache(二级高速缓冲存储器)或DRAM(动态随机访问存储器))。该cache-line从旧数据被替换到新数据从下一级存储器取回并存入该cache-line之间的这段时间所处的状态称为保留状态。处于保留状态的cache-line不能被其他发生缺失的访存请求替换。如果访存请求过多,会使cache中的cache-line全部处于保留状态,那么后续的访存请求发生cache缺失之后,就没有可以被替换的对象,导致访存请求发生停顿[3],直到cache中某一cache-line的数据返回,结束保留状态,这种现象称为保留停顿。GPU对一个线程束的访存请求按照先入先出的顺序进行处理,由于访存请求访问下一级存储器一般需要数百个周期,这样就使得读取单元中其他访存请求即使不会发生保留停顿,也必须等待数百个周期直到前面的访存请求保留停顿结束才能被处理,降低了访存请求的处理效率。

[0005] 另一方面,一个读取单元目前只能容纳一条线程束访存指令。也就是说,在当前位于读取单元中的线程束访存指令的所有访存请求未被处理完毕之前,即使FIFO中有空闲的

条目，线程束调度器也不能将其他访存指令发送到读取单元进行处理。如果当前访存指令的访存请求发生了保留停顿，下一条访存指令同样需要额外等待数百个周期，降低了线程束访存指令的处理效率。

[0006] 参考文献

[0007] [1]E.Lindholm,J.Nickolls,S.Oberman,J.Montrym.“NVIDIA Tesla:A Unified Graphics and Computing Architecture”,IEEE Micro,vol.28,no.2,pp.39-55,2008.

[0008] [2]NVIDIA Corporation.NVIDIA CUDA C Programming Guide,2019.

[0009] [3]W.Jia,K.A.Shaw,M.Martonosi.“MRPB:Memory Request Prioritization for Massively Parallel Processors”,International Symposium on High Performance Computer Architecture,pp.272-283,2014.

发明内容

[0010] 本发明提供了一种减缓GPU访存请求及指令访问cache时停顿的方法，本发明对保留停顿的访存请求重新排序，减少访存请求在读取单元中的停顿时间，提高访存请求的处理效率；并且通过对访存指令进行流水处理，减少访存指令在读取单元外的等待时间，提高访存指令的处理效率，详见下文描述：

[0011] 一种减缓GPU访存请求及指令访问cache时停顿的方法，所述方法包括：

[0012] 位于FIFO队首的访存请求访问L1 cache，将访存请求的tag与L1 cache中的tag进行比较，若存在发生保留停顿的访存请求，将该访存请求从FIFO队首弹出，放入FIFO队尾；

[0013] 通过数据通路连通队首和队尾、以及第一控制逻辑控制访存请求从FIFO队首弹出之后的走向；

[0014] 构建第二控制逻辑及第一、三控制信号，用于对线程束调度器和读取单元之间的访存指令进行流水处理，使得当读取单元中的地址合并单元将所有的访存请求都合并完毕时便可处理下一条访存指令，并且当FIFO队列中有空闲条目时即可生成访存请求并存入。

[0015] 其中，所述第一控制逻辑控制访存请求从FIFO队首弹出之后的走向具体为：

[0016] 当访存请求在L1 cache中的访问结果为保留时，第二控制信号为假，通过反向器之后为真；

[0017] 第一三态门呈导通状态，第二三态门呈高阻状态，表示访存请求从FIFO队首弹出之后被传输至FIFO队尾。

[0018] 进一步地，所述构建第二控制逻辑及第一、三控制信号，用于对线程束调度器和读取单元之间的访存指令进行流水处理具体为：

[0019] 1)若访存请求还未全部被地址合并单元合成完毕，则第三控制信号为假，通知线程束调度器不能将其他访存指令发送到读取单元；否则，第三控制信号为真，通知线程束调度器将其他访存指令发送到读取单元；

[0020] 2)将FIFO队列是否满的状态通过第一控制信号发送给地址合并单元从而控制访存请求的生成。

[0021] 其中，所述将FIFO队列是否满的状态通过第一控制信号发送给地址合并单元从而控制访存请求的生成具体为：

[0022] 如果FIFO队列变满，则第一控制信号为假，通知地址合并单元暂停对访存请求进

行合并，直到FIFO队列中有空闲的条目为止；

[0023] 否则，通知地址合并单元继续对访存请求进行合并、并放入FIFO队尾。

[0024] 优选地，所述方法还包括：对地址合并单元生成的访存请求和从FIFO队首弹出的访存请求在同一周期放入FIFO队尾时的冲突处理。

[0025] 其中，所述冲突处理具体为：

[0026] 赋予从FIFO队首弹出的访存请求高优先级，通过第二控制逻辑将访存请求放入FIFO队尾，

[0027] 地址合并单元暂停生成新的访存请求，直到没有从FIFO队首弹出的访存请求需要放入FIFO队尾为止。

[0028] 进一步地，所述通过第二控制逻辑将访存请求放入FIFO队尾为：

[0029] 当访存请求在L1 cache中的访问结果为命中或缺失时，第二控制信号为真，选通多路选择器的输入path2，将地址合并单元生成的访存请求放入FIFO队尾；

[0030] 当访存请求在L1 cache中的访问结果为保留停顿时，第二控制信号为假，选通多路选择器的输入path1，将从FIFO队首弹出的访存请求放入FIFO队尾。

[0031] 本发明提供的技术方案的有益效果是：

[0032] 1、本发明能够对发生保留停顿的访存请求重新排序，从而能够使后续的访存请求继续访问L1 cache，减少了访存请求的停顿时间；

[0033] 2、本发明中的其余访存指令无需等待读取单元中当前访存指令的全部访存请求处理完毕才能被读取单元处理，而是只需读取单元中的地址合并单元将全部线程的访存请求合并完毕，线程束调度器就能将其他访存指令发送到读取单元进行处理，从而减少访存指令的等待时间，提高访存指令的处理效率。

附图说明

[0034] 图1为本发明提供的减缓GPU访存请求和访存指令在L1 cache中停顿的处理的结构示意图；

[0035] 图2为访存请求发生保留停顿的示意图；

[0036] 图3为采用本发明之后的运行结果对比图。

具体实施方式

[0037] 为使本发明的目的、技术方案和优点更加清楚，下面对本发明实施方式作进一步地详细描述。

[0038] 实施例1

[0039] 参见图1，本发明实施例提供了一种减缓GPU访存请求及指令访问cache时停顿的方法，该方法包括以下步骤：

[0040] 101：将访存请求的tag(标签)与L1 cache中的tag进行比较，若存在发生保留停顿的访存请求，对FIFO队重新排序；

[0041] 其中，位于FIFO(先入先出)队首的访存请求访问L1 cache，首先将该访存请求的tag(标签)与L1 cache中的tag进行比较，包括以下三种情况：

[0042] 如果发生cache命中，则将该访存请求从FIFO队首弹出，进而访问所命中的cache-

line;或,

[0043] 如果发生cache缺失,则将该访存请求从FIFO队首弹出,发送到下一级存储器;或,

[0044] 如果发生保留停顿,则将该访存请求从FIFO队首弹出,放入FIFO队尾,这样在下一个周期,FIFO队列中的其他访存请求就可以继续访问L1 cache,避免了停顿,加快了访存请求的处理速度。

[0045] 为此,本发明实施例设计相应的数据通路path1来连通FIFO队列的队首和队尾、以及相应的第一控制逻辑1来控制访存请求从FIFO队首弹出之后的走向。

[0046] 其中,数据通路path1是用于传输访存请求信息的数据线,访存请求信息一般包括:地址信息、线程束索引、读写信息。

[0047] 其中,第一控制逻辑1用于控制访存请求从FIFO队首弹出后的走向:当访存请求在L1 cache中的访问结果r为命中或者缺失时,控制信号c2为真,通过反向器之后变为假。因此,三态门1呈高阻状态,三态门2呈导通状态,表示访存请求从FIFO队首弹出后被删除;当访存请求在L1 cache中的访问结果r为保留停顿时,控制信号c2为假,通过反向器之后变为真。因此三态门1呈导通状态,三态门2呈高阻状态,表示访存请求从FIFO队首弹出之后被发送到FIFO队尾。

[0048] 102:对访存指令进行流水处理;

[0049] 其中,该步骤具体包括以下流程:

[0050] 1)若当前的访存指令所产生的访存请求还未全部被地址合并单元合成完毕,则控制信号c3为假,通知线程束调度器不能将其他访存指令发送到读取单元;

[0051] 2)若当前的访存指令所产生的访存请求已全部被地址合并单元合成完毕,则控制信号c3为真,通知线程束调度器可以将其他访存指令发送到读取单元;

[0052] 3)持续检测FIFO队列的状态;

[0053] 如果FIFO队列变满,则控制信号c1为假,通知地址合并单元暂停对访存请求进行合并,直到FIFO队列中有空闲的条目为止;

[0054] 如果FIFO未满,则控制信号c1为真,通知地址合并单元可以继续对访存请求进行合并、并放入FIFO队尾。

[0055] 为此,FIFO控制器将FIFO队列是否满的状态通过控制信号c1发送给地址合并单元从而控制访存请求的生成。

[0056] 103:地址合并单元生成的访存请求和从FIFO队首弹出的访存请求在同一周期放入FIFO队尾时的冲突处理。

[0057] 若地址合并单元生成的访存请求和从FIFO队首弹出的访存请求需要在同一周期放入FIFO队尾,此时赋予从FIFO队首弹出的访存请求高优先级,将其优先放入FIFO队尾。此时,地址合并单元暂停生成新的访存请求,直到没有从FIFO队首弹出的访存请求需要放入FIFO队尾为止。

[0058] 为此,需要在FIFO队尾设计相应的控制逻辑2,并将访存请求的tag比较结果r作为控制信号控制FIFO队尾的输入选择。当访存请求在L1 cache中的访问结果r为命中或者缺失时,控制信号c2为真,选通控制逻辑2中多路选择器的输入path2,表示将地址合并单元生成的访存请求放入FIFO队尾;当访存请求在L1 cache中的访问结果r为保留停顿时,控制信号c2为假,选通控制逻辑2中多路选择器的输入path1,表示将从FIFO队首弹出的访存请求

放入FIFO队尾。

[0059] 实施例2

[0060] 下面对比现有技术中的对访存请求保留停顿的处理方式，对本发明实施例1作进一步地介绍、以及对比性验证，详见下文描述：

[0061] 访存请求的FIFO队列具有32个条目，GPU流多处理器个数：15；DRAM通道数：6；流多处理器最大线程个数：1536；流多处理器寄存器文件容量：128KB；共享存储器容量：48KB；L1 cache：4路组关联，cache-line大小为128字节，32组，总容量为16KB；L2 cache(第二级高速缓冲存储器)：8路组关联，cache-line大小为128字节，总容量为128KB。L1 cache访问延迟：1周期；L2 cache访问延迟：120周期；DRAM访问延迟：220周期。

[0062] 如图2所示，假设初始状态时，L1 cache中所有的cache-line都能被访问(cache冷启动)，FIFO队列中存储的访存请求是来自访存指令inst-a的req-a0、req-a1、req-a2・・・req-a20。根据地址映射，req-a0・・・req-a4将会访问L1 cache中的set-0，req-5・・・req-a9将会访问L1 cache中的set-1。

[0063] 按照先入先出的访问顺序，先用req-a0访问L1 cache，发生cache缺失，set-0中的一个cache-line分配给req-a0，处于保留状态(R)，然后req-a0被发送去往下一级存储器，同时FIFO控制器将req-a0从FIFO队首弹出。之后的三个周期，set-0中的剩余三个cache-line分别分配给req-a1，req-a2，req-a3。此时，set-0中所有的cache-line都处于保留状态(R)。当req-a4继续访问set-0并发生cache缺失后，由于此时set-0中已经没有可分配的cache-line，因此req-a4发生保留停顿，FIFO控制器不会将req-a4从FIFO队首弹出，而是等待req-a0，req-a1，req-a2或者req-a3从下一级存储器返回，将其所对应的cache-line保留状态取消。虽然req-a5等访存请求不需要访问set-0，即不需要等待req-a0，req-a1，req-a2，req-a3返回，但是由于req-a4在前面堵塞，req-a5等其余访存请求也不得不等待，极大的降低了访存请求的处理效率。

[0064] 另一方面，此时FIFO队列中存储的访存请求全部为inst-a的访存请求，虽然FIFO队列此时仍然有空闲的条目，但是FIFO控制器会通过控制信号c1通知线程束调度器目前读取单元依然不能处理其他的访存指令inst-b等，导致inst-b等其他访存指令也要受到inst-a访存指令保留停顿的影响，极大的降低了访存指令的处理效率。

[0065] 如图1所示，采用本发明实施例之后，当req-a4发生保留停顿之后，FIFO控制器会将req-a4从FIFO队首弹出，同时控制信号c2控制数据通路path1开通，将req-a4放入FIFO队尾，req-a5变成了FIFO队首，因此避免了保留停顿。

[0066] 下一个周期，req-a5访问set-1，从而提高了访存请求的处理速度。另外，此时地址合并单元已将inst-a的所有访存请求合并完毕，地址合并单元将会通知线程束调度器目前读取单元可以接收inst-b。假设inst-b共能生成24个访存请求(req-b0・・・req-b23)，req-b0和req-a4需要同时放入FIFO队尾，发生了访问冲突。

[0067] 本发明实施例给予发生保留停顿的req-a4较高的优先级，因此req-a4先被放入FIFO队列，此周期中，控制信号c2控制读取单元中的地址合并单元暂停对inst-b的访存请求进行合并。当FIFO队尾的访问冲突结束之后，控制信号c2控制地址合并单元继续对inst-b的访存请求进行合并。因此，本发明实施例也同时提高了对访存指令的处理速度。

[0068] 如图3所示，采用本发明实施例之后，GPU的平均性能(GM)提高了23％。

[0069] 本发明实施例对各器件的型号除做特殊说明的以外，其他器件的型号不做限制，只要能完成上述功能的器件均可。

[0070] 本领域技术人员可以理解附图只是一个优选实施例的示意图，上述本发明实施例序号仅仅为了描述，不代表实施例的优劣。

[0071] 以上所述仅为本发明的较佳实施例，并不用以限制本发明，凡在本发明的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

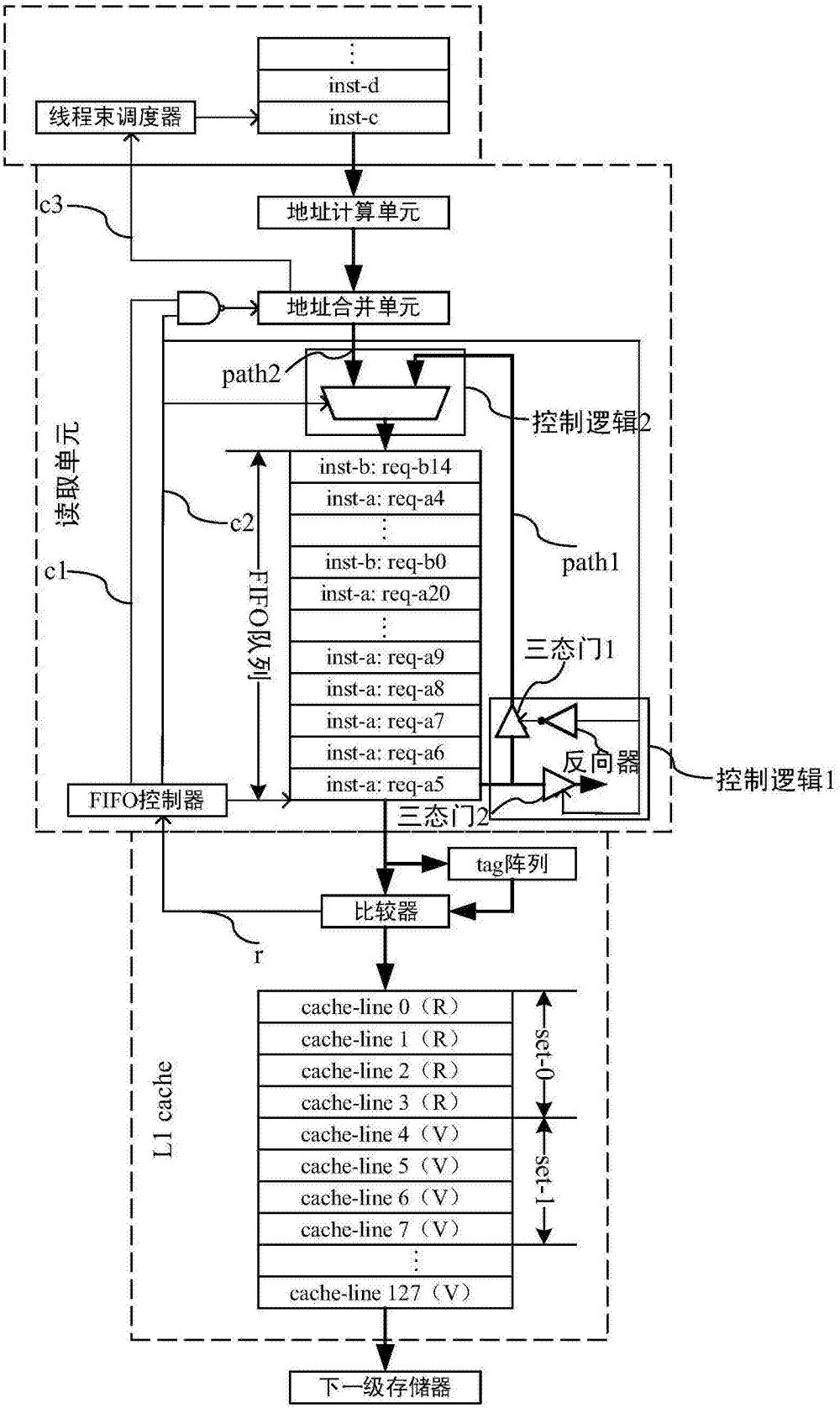

线程束调度器
inst-d
inst-c
地址计算单元
c3
地址合并单元
path2
控制逻辑2
读取单元
inst-b: req-b14
inst-a: req-a4
c2
path1
c1
FIFO队列
inst-b: req-b0
inst-a: req-a20
三态门1
inst-a: req-a9
inst-a: req-a8
inst-a: req-a7
反向器
控制逻辑1
inst-a: req-a6
inst-a: req-a5
FIFO控制器
三态门2
tag阵列
比较器
r
cache-line 0（R）
cache-line 1（R）
cache-line 2（R）
cache-line 3（R）
set-0
cache-line 4（V）
cache-line 5（V）
cache-line 6（V）
cache-line 7（V）
set-1
L1 cache
cache-line 127（V）
下一级存储器

图1

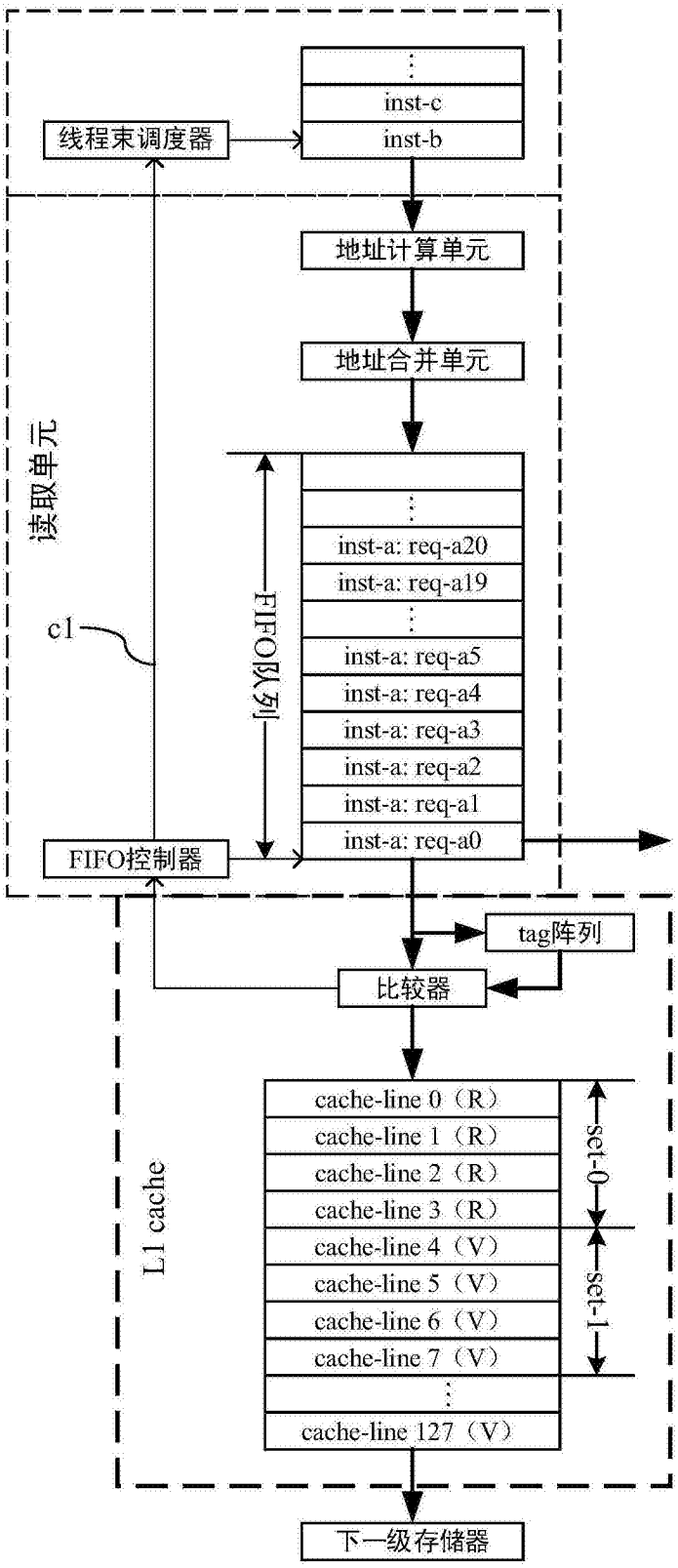

线程束调度器
inst-c
inst-b
地址计算单元
地址合并单元
读取单元
c1
FIFO队列
inst-a: req-a20
inst-a: req-a19
inst-a: req-a5
inst-a: req-a4
inst-a: req-a3
inst-a: req-a2
inst-a: req-a1
inst-a: req-a0
FIFO控制器
tag阵列
比较器
L1 cache
cache-line 0（R）
cache-line 1（R）
cache-line 2（R）
cache-line 3（R）
cache-line 4（V）
cache-line 5（V）
cache-line 6（V）
cache-line 7（V）
cache-line 127（V）
set-0
set-1
下一级存储器

图2

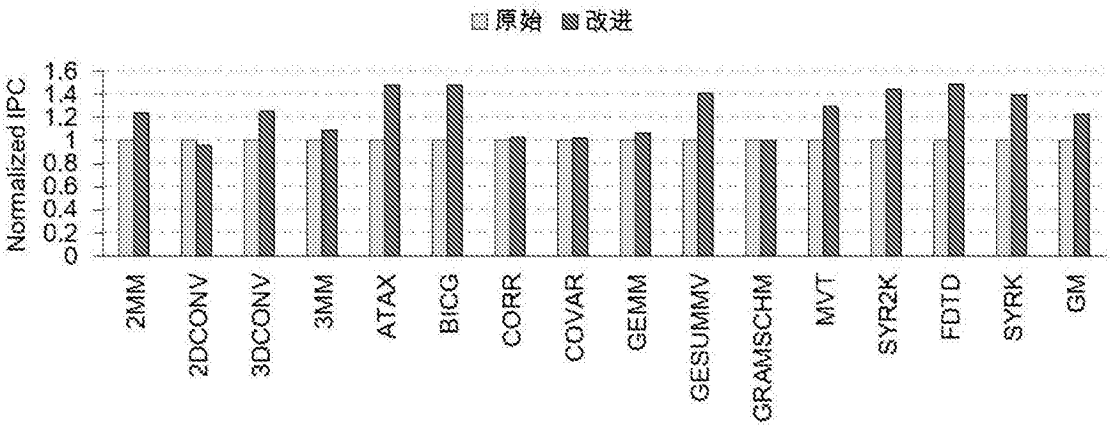

原始
改进
Normalized IPC
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
2MM
2DCONV
3DCONV
3MM
ATAX
BICG
CORR
COVAR
GEMM
GESUMMV
GRAMSCHM
MVT
SYR2K
FDTD
SYRK
GM

图3