(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 115858420 A
(43)申请公布日 2023.03.28

(21)申请号 202310153451.8
(22)申请日 2023.02.23
(71)申请人 芯砺智能科技(上海)有限公司
地址 201306 上海市浦东新区中国(上海)自由贸易试验区临港新片区环湖西二路888号C楼
(72)发明人 李晓均
(74)专利代理机构 北京品源专利代理有限公司 11332
专利代理师 苏舒音
(51)Int.Cl.
G06F 12/0877(2016.01)
G06F 13/28(2006.01)
G06F 13/40(2006.01)
G06F 13/16(2006.01)
G06F 15/78(2006.01)

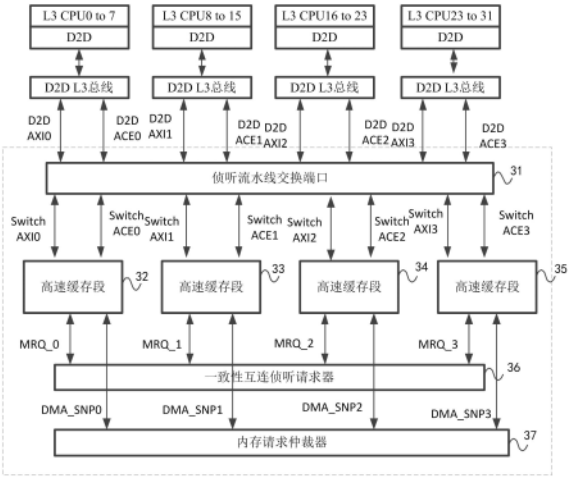
权利要求书3页 说明书11页 附图6页

(54)发明名称

用于支持多处理器架构的系统缓存架构和芯片

(57)摘要

本发明公开了一种用于支持多处理器架构的系统缓存架构和芯片。该架构包括:侦听流水线交换端口与多处理器架构的至少两个处理器的最后一级内存总线连接,将来自任一处理器的内存读写请求通过内存请求仲裁器转发至内存系统或者将内存读写请求发送至至少两个高速缓存段中的任一高速缓存段;一致性互连侦听请求器将来自DMA主机的侦听读写请求发送至至少两个高速缓存段中的任一高速缓存段;至少两个高速缓存段用于响应来自侦听流水线交换端口或一致性互连侦听请求器的并发的读写请求,并在存储的缓存数据与内存读写请求或侦听读写请求对应时反馈或更新缓存数据。本发明实施例的技术方案,对多处理器架构的高性能扩展提供了支持。



CN 115858420 A

1.一种用于支持多处理器架构的系统缓存架构,其特征在于,包括:

侦听流水线交换端口、至少两个高速缓存段、内存请求仲裁器、一致性互连侦听请求器;

所述侦听流水线交换端口与多处理器架构的至少两个处理器的最后一级内存总线连接,将来自任一处理器的内存读写请求通过所述内存请求仲裁器转发至内存系统或者将所述内存读写请求发送至所述至少两个高速缓存段中的任一高速缓存段;

所述内存请求仲裁器通过一致性互连与内存系统连接;

所述一致性互连侦听请求器通过一致性互连与直接内存访问DMA主机连接,将来自DMA主机的侦听读写请求发送至所述至少两个高速缓存段中的任一高速缓存段;

所述至少两个高速缓存段用于响应来自所述侦听流水线交换端口或所述一致性互连侦听请求器的并发的读写请求,存储与所述内存读写请求或所述侦听读写请求对应的缓存数据,并在存储的缓存数据与所述内存读写请求或所述侦听读写请求对应时反馈或更新所述缓存数据。

2.根据权利要求1所述的用于支持多处理器架构的系统缓存架构,其特征在于,所述侦听流水线交换端口还用于在将来自任一处理器的内存读写请求通过所述内存请求仲裁器转发至内存系统或者将所述内存读写请求发送至所述至少两个高速缓存段中的任一高速缓存段之前,确定所述内存读写请求是否位于可缓存周期,若是则将所述内存读写请求发送至所述至少两个高速缓存段中的任一高速缓存段,若否则将所述内存读写请求通过所述内存请求仲裁器转发至内存系统。

3.根据权利要求1所述的用于支持多处理器架构的系统缓存架构,其特征在于,若所述内存读写请求为内存读取请求,则所述侦听流水线交换端口根据来自第一处理器的内存读取请求在第一高速缓存段中查询对应的第一缓存数据,若缓存未命中则通过所述内存请求仲裁器从内存系统中获取对应的第一缓存数据发送至所述第一处理器并存储至所述第一高速缓存段,若缓存命中则从所述第一处理器以外的其他处理器中侦听所述第一缓存数据对应的脏数据,若侦听到所述脏数据则将所述脏数据发送至所述第一处理器并刷新所述第一高速缓存段中的数据,所述第一处理器为所述多处理器架构中的任一处理器,所述第一高速缓存段为所述至少两个高速缓存段中任一空闲高速缓存段。

4.根据权利要求3所述的用于支持多处理器架构的系统缓存架构,其特征在于,若来自第一处理器的内存读取请求在第一高速缓存段中查询对应的第一缓存数据时缓存命中,且从所述第一处理器以外的其他处理器中未侦听到所述第一缓存数据对应的脏数据,则所述侦听流水线交换端口将所述第一高速缓存段中的第一缓存数据发送至所述第一处理器。

5.根据权利要求1所述的用于支持多处理器架构的系统缓存架构,其特征在于,若所述内存读写请求为内存写入请求,则所述侦听流水线交换端口确定所述内存写入请求为部分写入请求时,在第一高速缓存段中存储所述内存写入请求,并向第一处理器以外的其他处理器发送侦听请求阻止其他处理器更新所述内存写入请求所在缓存行并将所述内存写入请求通过所述内存请求仲裁器转发至内存系统,若所述侦听流水线交换端口确定所述内存写入请求为全部写入请求时直接将所述内存写入请求通过所述内存请求仲裁器转发至内存系统。

6.根据权利要求5所述的用于支持多处理器架构的系统缓存架构,其特征在于,所述侦

听流水线交换端口将所述内存写入请求通过所述内存请求仲裁器转发至内存系统之后，还将所述第一高速缓存段中存储的内容写入请求标记为脏数据。

7.根据权利要求1所述的用于支持多处理器架构的系统缓存架构，其特征在于，当所述侦听读写请求为侦听读取请求时，接收到所述侦听读取请求的第一高速缓存段，用于确定所述侦听读写请求是否缓存命中，若未命中则不响应所述侦听读取请求。

8.根据权利要求7所述的用于支持多处理器架构的系统缓存架构，其特征在于，所述第一高速缓存段，还用于在接收到所述侦听读取请求时，确定所述侦听读取请求缓存命中时，确定与所述侦听读取请求对应的第二缓存数据是否为脏数据，若是则反馈所述第二缓存数据并将所述第二缓存数据标记为干净数据，若否则通过所述侦听流水线交换端口在所述至少两个处理器中查询与所述侦听读取请求对应的脏数据并将所述脏数据缓存并反馈。

9.根据权利要求8所述的用于支持多处理器架构的系统缓存架构，其特征在于，所述第一高速缓存段还用于在接收到所述侦听读取请求时，确定所述侦听读取请求缓存命中时，确定所述侦听读取请求对应的第二缓存数据为干净数据时，若通过所述侦听流水线交换端口在所述至少两个处理器中未查询到与所述侦听读取请求对应的脏数据则将反馈所述第二缓存数据。

10.根据权利要求1所述的用于支持多处理器架构的系统缓存架构，其特征在于，当所述侦听读写请求为侦听写入请求时，接收到所述侦听写入请求的第一高速缓存段，用于确定所述侦听读写请求是否缓存命中，若未命中则将所述侦听写入请求写入内存系统目录。

11.根据权利要求10所述的用于支持多处理器架构的系统缓存架构，其特征在于，所述第一高速缓存段，还用于在接收到所述侦听写入请求时，确定所述侦听写入请求缓存命中时，当确定所述侦听写入请求为脏数据命中时则将所述侦听写入请求写入内存系统目录，当确定所述侦听写入请求为干净数据命中时将所述侦听写入请求通过所述侦听流水线交换端口发送至所述至少两个处理器。

12.根据权利要求11所述的用于支持多处理器架构的系统缓存架构，其特征在于，所述第一高速缓存段，还用于在确定所述侦听写入请求缓存命中时，将所述侦听写入请求写入内存系统目录或发送至所述至少两个处理器之后，将命中的缓存行失效。

13.根据权利要求1~12任一项所述的用于支持多处理器架构的系统缓存架构，其特征在于，每个高速缓存段包括高速缓存、双端口缓存标签段、处理器缓存控制器和侦听控制器；

所述高速缓存用于存储数据，所述双端口缓存标签段用于为来自侦听流水线交换端口或来自一致性互连侦听请求器的数据分配不同的缓存标签，所述处理器缓存控制器用于控制来自所述侦听流水线交换端口的数据请求，所述侦听控制器用于控制来自所述一致性互连侦听请求器的数据请求。

14.根据权利要求1~12任一项所述的用于支持多处理器架构的系统缓存架构，其特征在于，所述多处理器架构为由多个处理器组成的处理器集群，或者所述多处理器架构为由多个CPU芯粒组成的芯粒到芯粒互连结构。

15.一种芯片，其特征在于，包括至少两个芯粒和如权利要求1~14任一项所述的用于支持多处理器架构的系统缓存架构，所述至少两个芯粒通过所述用于支持多处理器架构的系统缓存架构与一致性互连连接。

16.一种处理器集群组件,其特征在于,包括由至少两个处理器组成的处理器集群和如权利要求1~14任一项所述的用于支持多处理器架构的系统缓存架构,所述至少两个处理器通过所述用于支持多处理器架构的系统缓存架构与一致性互连连接。

用于支持多处理器架构的系统缓存架构和芯片

技术领域

[0001] 本发明实施例涉及芯片技术,尤其涉及一种用于支持多处理器架构的系统缓存架构和芯片。

背景技术

[0002] 高性能扩展是高性能计算中大规模多处理器架构的目标。其中,中央处理单元(Central Processing Unit,CPU)芯粒(chiplet)是一种能够有效扩展CPU性能的技术架构。Chiplet技术通过使用多个较小的芯粒进行芯片设计,既可以降低制造成本,又可以提高计算性能。采用CPU集群(clusters)同样能够有效扩展CPU性能。但CPU chiplet架构和CPU集群在实现高性能扩展时,多个CPU或多个CPU chiplet 之间的一致性互连(coherent interconnect)性能均是影响性能充分发挥的关键。

[0003] 图1示出包括32个内核的4个CPU集群的两个系统,如图1所示,左侧示出的是嵌入式CPU系统,所有CPU集群都在一个片上系统(System On Chip,SOC)中。右侧示出的是CPU chiplet系统,部分或全部CPU可以是驻留在SOC之外的CPU chiplet。从图1中可以看出,无论哪种架构,CPU集群中的各CPU以及CPU chiplet架构中的各CPU chiplet均通过一致性互连(coherent interconnect)与内存系统(Memory System)以及通过coherent interconnect和直接内存访问(Direct Memory Access,DMA)主机(master)交互(例如高速外设组件互连(Peripheral Component Interconnect express,PCIe)或计算高速连接(Compute EXpress Link,CLX)或加速的缓存一致性互连(Cache Coherent Interconnect for Accelerators,CCIX)。

[0004] 当系统执行对称对处理(Symmetrical Multi-Processing,SMP)操作系统(Operating System,OS)内核时,所有CPU之间需要保持缓存一致性。缓存一致性协议的机制将决定系统执行并发多线程任务的性能。如何能够在多处理器架构下,解决缓存一致性的问题,是实现多处理器架构的高性能扩展的关键因素。

发明内容

[0005] 本发明提供一种用于支持多处理器架构的系统缓存架构和芯片,降低了一致性互连与处理器之间的数据请求交互,对多处理器架构的高性能扩展提供了支持。

[0006] 第一方面,本发明实施例提供了一种对多处理器架构的高性能扩展提供了支持,包括:侦听流水线交换端口、至少两个高速缓存段、内存请求仲裁器、一致性互连侦听请求器;

[0007] 侦听流水线交换端口与多处理器架构的至少两个处理器的最后一级内存总线连接,将来自任一处理器的内存读写请求通过内存请求仲裁器转发至内存系统或者将内存读写请求发送至至少两个高速缓存段中的任一高速缓存段;

[0008] 内存请求仲裁器通过一致性互连与内存系统连接;

[0009] 一致性互连侦听请求器通过一致性互连与DMA主机连接,将来自DMA主机的侦听读

写请求发送至至少两个高速缓存段中的任一高速缓存段；

[0010] 至少两个高速缓存段用于响应来自侦听流水线交换端口或一致性互连侦听请求器的并发的读写请求，存储与内存读写请求或侦听读写请求对应的缓存数据，并在存储的缓存数据与内存读写请求或侦听读写请求对应时反馈或更新缓存数据。

[0011] 在第一方面一种可能的实现方式中，侦听流水线交换端口还用于在将来自任一处理器的内存读写请求通过内存请求仲裁器转发至内存系统或者将内存读写请求发送至至少两个高速缓存段中的任一高速缓存段之前，确定内存读写请求是否位于可缓存周期，若是则将内存读写请求发送至至少两个高速缓存段中的任一高速缓存段，若否则将内存读写请求通过内存请求仲裁器转发至内存系统。

[0012] 在第一方面一种可能的实现方式中，若内存读写请求为内存读取请求，则侦听流水线交换端口根据来自第一处理器的内存读取请求在第一高速缓存段中查询对应的第一缓存数据，若缓存未命中则通过内存请求仲裁器从内存系统中获取对应的第一缓存数据发送至第一处理器并存储至第一高速缓存段，若缓存命中则从第一处理器以外的其他处理器中侦听第一缓存数据对应的脏数据，若侦听到脏数据则将脏数据发送至第一处理器并刷新第一高速缓存段中的数据，第一处理器为多处理器架构中的任一处理器，第一高速缓存段为至少两个高速缓存段中任一空闲高速缓存段。

[0013] 在第一方面一种可能的实现方式中，若来自第一处理器的内存读取请求在第一高速缓存段中查询对应的第一缓存数据时缓存命中，且从第一处理器以外的其他处理器中未侦听到第一缓存数据对应的脏数据，则侦听流水线交换端口将第一高速缓存段中的第一缓存数据发送至第一处理器。

[0014] 在第一方面一种可能的实现方式中，若内存读写请求为内存写入请求，则侦听流水线交换端口确定内存写入请求为部分写入请求时，在第一高速缓存段中存储内存写入请求，并向第一处理器以外的其他处理器发送侦听请求阻止其他处理器更新内存写入请求所在缓存行并将内存写入请求通过内存请求仲裁器转发至内存系统，若侦听流水线交换端口确定内存写入请求为全部写入请求时直接将内存写入请求通过内存请求仲裁器转发至内存系统。

[0015] 在第一方面一种可能的实现方式中，侦听流水线交换端口将内存写入请求通过内存请求仲裁器转发至内存系统之后，还将第一高速缓存段中存储的内容写入请求标记为脏数据。

[0016] 在第一方面一种可能的实现方式中，当侦听读写请求为侦听读取请求时，接收到侦听读取请求的第一高速缓存段，用于确定侦听读写请求是否缓存命中，若未命中则不响应侦听读取请求。

[0017] 在第一方面一种可能的实现方式中，第一高速缓存段，还用于在接收到侦听读取请求时，确定侦听读取请求缓存命中时，确定与侦听读取请求对应的第二缓存数据是否为脏数据，若是则反馈第二缓存数据并将第二缓存数据标记为干净数据，若否则通过侦听流水线交换端口在至少两个处理器中查询与侦听读取请求对应的脏数据并将脏数据缓存并反馈。

[0018] 在第一方面一种可能的实现方式中，第一高速缓存段还用于在接收到侦听读取请求时，确定侦听读取请求缓存命中时，确定侦听读取请求对应的第二缓存数据为干净数据

时，若通过侦听流水线交换端口在至少两个处理器中未查询到与侦听读取请求对应的脏数据则将反馈第二缓存数据。

[0019] 在第一方面一种可能的实现方式中，当侦听读写请求为侦听写入请求时，接收到侦听写入请求的第一高速缓存段，用于确定侦听读写请求是否缓存命中，若未命中则将侦听写入请求写入内存系统目录。

[0020] 在第一方面一种可能的实现方式中，第一高速缓存段，还用于在接收到侦听写入请求时，确定侦听写入请求缓存命中时，当确定侦听写入请求为脏数据命中时则将侦听写入请求写入内存系统目录，当确定侦听写入请求为干净数据命中时将侦听写入请求通过侦听流水线交换端口发送至至少两个处理器。

[0021] 在第一方面一种可能的实现方式中，第一高速缓存段，还用于在确定侦听写入请求缓存命中时，将侦听写入请求写入内存系统目录或发送至至少两个处理器之后，将命中的缓存行失效。

[0022] 在第一方面一种可能的实现方式中，每个高速缓存段包括高速缓存、双端口缓存标签段、处理器缓存控制器和侦听控制器；

[0023] 高速缓存用于存储数据，双端口缓存标签段用于为来自侦听流水线交换端口或来自一致性互连侦听请求器的数据分配不同的缓存标签，处理器缓存控制器用于控制来自侦听流水线交换端口的数据请求，侦听控制器用于控制来自一致性互连侦听请求器的数据请求。

[0024] 在第一方面一种可能的实现方式中，多处理器架构为由多个处理器组成的处理器集群，或者多处理器架构为由多个CPU芯粒组成的芯粒到芯粒互连结构。

[0025] 第二方面，本申请实施例提供了一种芯片，包括至少两个芯粒和如第一方面任一种可能方式的用于支持多处理器架构的系统缓存架构，至少两个芯粒通过用于支持多处理器架构的系统缓存架构与一致性互连连接。

[0026] 第二方面，本申请实施例提供了一种处理器集群组件，包括由至少两个处理器组成的处理器集群和如第一方面任一种可能的实现方式的用于支持多处理器架构的系统缓存架构，至少两个处理器通过用于支持多处理器架构的系统缓存架构与一致性互连连接。

[0027] 本发明实施例提供的用于支持多处理器架构的系统缓存架构和芯片，通过在多处理器架构的多个处理器和一致性互连之间建立由侦听流水线交换端口、至少两个高速缓存段、内存请求仲裁器、一致性互连侦听请求器组成的架构，在多处理器架构中的处理器数量很大时，能够应对并发的读写请求，避免大量的读写请求无法被即使处理而增加的延迟，并且至少两个高速缓存段中存储与内存读写请求或侦听读写请求对应的缓存数据，并在存储的缓存数据与内存读写请求或侦听读写请求对应时反馈或更新缓存数据，可以对来自DMA主机的侦听请求实现过滤器的功能，减少DMA主机通过一致性互连直接侦听处理器的请求。

附图说明

[0028] 图1示出包括32个内核的4个CPU集群的两个系统；

[0029] 图2为本申请提供的用于支持多处理器架构的系统缓存架构在系统中的应用示意图；

[0030] 图3为本申请实施例提供的用于支持多处理器架构的系统缓存架构的结构示意

图；

[0031] 图4为本申请实施例提供的另一种用于支持多处理器架构的系统缓存架构的结构示意图；

[0032] 图5为本申请实施例提供的用于支持多处理器架构的系统缓存架构进行AXI内存读取的流程图；

[0033] 图6为本申请实施例提供的用于支持多处理器架构的系统缓存架构进行AXI内存写入的流程图；

[0034] 图7为本申请实施例提供的用于支持多处理器架构的系统缓存架构响应来自DMA主机的侦听读取请求的流程图；

[0035] 图8为本申请实施例提供的用于支持多处理器架构的系统缓存架构响应来自DMA主机的侦听写入读取请求的流程图。

具体实施方式

[0036] 下面结合附图和实施例对本发明作进一步的详细说明。可以理解的是，此处所描述的具体实施例仅仅用于解释本发明，而非对本发明的限定。另外还需要说明的是，为了便于描述，附图中仅示出了与本发明相关的部分而非全部结构。

[0037] 如图1所示，当系统执行SMP OS内核时，需要保持所有CPU之间的缓存一致性，缓存一致性将决定系统执行并发多线程任务的性能。缓存一致性算法有两种，一种是侦听(snooping)算法，另一种是基于目录的算法。基于目录的算法的优点是减少了总线流量，易于在一致性互连中实现。snooping算法需要侦听所有的CPU，并且需要侦听的快速响应。在许多现有CPU架构系统架构中，侦听算法的性能被证明高于基于目录的算法。本申请提出了一种具用于支持多处理器架构的系统缓存架构，具体地，为一种用于支持多处理器架构的具有侦听总线的系统缓存架构，该架构具有最后一级系统缓存(last level system cache，LLSC)，以减少由一致性互连引起的侦听冲突。本申请实施例提供的用于支持多处理器架构的系统缓存架构中的LLSC 可以避免基于目录的一致性算法中的长延迟，并且类似于基于目录的一致性算法还可以减少总线流量。

[0038] 在CPU集群或基于CPU chiplet 的芯粒到芯粒(die-to-die)结构中，最后一级系统缓存(如L3) 之后的私有CPU总线的流量很低。通过CPU集群中每个CPU中有效的本地缓存系统，仅有不到5%的CPU周期要求CPU集群向一致性互连发出内存请求。CPU集群到一致性互连的低总线流量并不是导致性能下降的主要原因，性能下降的根本原因是 CPU 集群通过一致性互连获得所需内存的延迟。侦听算法将内存侦听请求转发并广播到CPU集群中的所有CPU，来自CPU集群的内存侦听请求可能会与一致性互连中的其他内存主机发生冲突，特别是DMA内存主机，例如 PCIe、图形处理器(Graphics Processing Unit，GPU)等。DMA 内存主机的内存访问模式与CPU集群的内存访问模式不同。在一种典型的系统中，CPU集群每次通过一条64字节的高速缓存行访问内存，DMA 内存主机以一个块的形式访问内存，每次2K/4K字节。当来自CPU集群的内存请求与来自DMA内存主机的内存请求冲突时，侦听延迟会变得很长。一致性互连不能有效地支持两种不同的存储器访问模式，即快速响应和长存储器传输。

[0039] 上述内容以CPU集群为例进行了说明，但基于CPU chiplet 的D2D架构由于存在多

个芯粒,同样存在上述问题。

[0040] 为了解决上述问题,本申请提出在CPU集群(或CPU chiplet)与一致性互连之间建议最后一级系统缓存(LLSC),LLSC用于隔离两种不同的内存访问模式,即快速响应和长内存传输。如图2所示,图2为本申请提供的用于支持多处理器架构的系统缓存架构在系统中的应用示意图。图2以CPU chiplet架构为例,系统包括由32个芯粒组成的集群,其中每8个芯粒位于一个SOC中,具有一个最后一级内存总线(L3),LLSC通过D2D连接与各芯粒之间通过高级可拓展接口(Advanced eXtensible Interface,AXI)/AXI一致性扩展(AXI Coherency Extensions,ACE)总线连接。并且LLSC通过AXI/ACE总线与一致性互连连接并进一步地与内存系统以及DMA主机连接,其中AXI总线是用于读/写内存的内存总线,ACE总线是用于缓存一致性的侦听总线。图1中的AXI/ACE总线仅为示意性说明,通过其他总线同样可以实现。图2中的LLSC即本申请提供的用于支持多处理器架构的系统缓存架构,与图1所示系统架构相比,本申请就是在CPU集群与一致性互连之间增加了LLSC架构。图2中示出LLSC与一致性互连、内存系统、DMA主机位于同一SOC中,但LLSC还可以部署于其他位置,例如独立的SOC等。本申请实施例提供的用于支持多处理器架构的系统缓存架构即为LLSC。LLSC将执行侦听过滤,以防止DMA长内存访问终端CPU集群的本地缓存总线,由于一致性互连的中断较少,CPU集群之间的侦听响应将更快,那么随着CPU集群中CPU数量的增加,CPU集群的性能将能够得到高度扩展。

[0041] 图3为本申请实施例提供的用于支持多处理器架构的系统缓存架构的结构示意图。如图3所示,本申请实施例提供的用于支持多处理器架构的系统缓存架构包括侦听流水线交换端口(Snooping pipeline switch)31、至少两个高速缓存段32~35、内存请求仲裁器(Memory Request Arbiter)36、一致性互连侦听请求器(Coherent interconnect Snooping Requester)37。

[0042] 侦听流水线交换端口31与多处理器架构的至少两个处理器的最后一级内存总线(L3 bus)连接,将来自任一处理器的内存读写请求通过内存请求仲裁器36转发至内存系统或者将内存读写请求发送至至少两个高速缓存段中的任一高速缓存段。内存请求仲裁器36通过一致性互连与内存系统连接。一致性互连侦听请求器37通过一致性互连与DMA主机连接,将来自DMA主机的侦听读写请求发送至至少两个高速缓存段中的任一高速缓存段;至少两个高速缓存段用于响应来自侦听流水线交换端口31或一致性互连侦听请求器37的并发的读写请求,存储与内存读写请求或侦听读写请求对应的缓存数据,并在存储的缓存数据与内存读写请求或侦听读写请求对应时反馈或更新缓存数据。图3中以基于CPU chiplet的架构为例进行说明。

[0043] 侦听流水线交换端口31是与CPU集群连接的端口,侦听流水线交换端口31通过一对D2D流水线将CPU chiplet的最后一级内存总线完全复制到LLSC中,如图中所示,侦听流水线交换端口31与CPU chiplet通过D2D AXI总线和D2D ACE总线连接,并且侦听流水线交换端口31通过Switch AXI总线和Switch ACE总线与各高速缓存段连接。

[0044] 内存请求仲裁器36作为各高速缓存段与系统内存之间的接口,与各高速缓存段通过内存请求总线(MRQ)连接。

[0045] 一致性互连侦听请求器37作为各高速缓存段与DMA主机之间的接口,与各高速缓存段通过侦听总线(DMA_SNP)连接。

[0046] 至少两个高速缓存段用于响应来自侦听流水线交换端口31或一致性互连侦听请求器37的并发的读写请求，也就是说，设置多个高速缓存段是为了在多处理器架构中的处理器数量很大时，能够应对并发的读写请求，避免大量的读写请求无法被即使处理而增加的延迟。并且至少两个高速缓存段中存储与内存读写请求或侦听读写请求对应的缓存数据，并在存储的缓存数据与内存读写请求或侦听读写请求对应时反馈或更新缓存数据，可以对来自DMA主机的侦听请求实现过滤器的功能，减少DMA主机通过一致性互连直接侦听处理器的请求。

[0047] 本实施例提供的用于支持多处理器架构的系统缓存架构，通过在多处理器架构的多个处理器和一致性互连之间建立由侦听流水线交换端口、至少两个高速缓存段、内存请求仲裁器、一致性互连侦听请求器组成的架构，在多处理器架构中的处理器数量很大时，能够应对并发的读写请求，避免大量的读写请求无法被即使处理而增加的延迟，并且至少两个高速缓存段中存储与内存读写请求或侦听读写请求对应的缓存数据，并在存储的缓存数据与内存读写请求或侦听读写请求对应时反馈或更新缓存数据，可以对来自DMA主机的侦听请求实现过滤器的功能，减少DMA主机通过一致性互连直接侦听处理器的请求。

[0048] 图3中以高速缓存段32、高速缓存段33、高速缓存段34、高速缓存段35共四个高速缓存段为例，高速缓存段的数量可以根据实际需求设置。用于支持多处理器架构的系统缓存架构中各部件之间的连接关系如图中所示。

[0049] 图4为本申请实施例提供的另一种用于支持多处理器架构的系统缓存架构的结构示意图。本实施例提供的用于支持多处理器架构的系统缓存架构的结构在图3的基础上，进一步地示出了各高速缓存段的具体结构。每个高速缓存段包括高速缓存、双端口缓存标签段、处理器缓存控制器和侦听控制器。每个高速缓存段的结构相同。

[0050] 如图4所示，高速缓存段32包括高速缓存321、双端口缓存标签段322、处理器缓存控制器323和侦听控制器324；高速缓存段33包括高速缓存331、双端口缓存标签段332、处理器缓存控制器333和侦听控制器334；高速缓存段34包括高速缓存341、双端口缓存标签段342、处理器缓存控制器343和侦听控制器344；高速缓存段35包括高速缓存351、双端口缓存标签段352、处理器缓存控制器353和侦听控制器354。

[0051] 以高速缓存段32为例，高速缓存321用于存储数据，包括来自侦听流水线交换端口31的数据或者来自一致性互连侦听请求器37的数据。双端口缓存标签段322用于为来自侦听流水线交换端口31或来自一致性互连侦听请求器37的数据分配不同的缓存标签，为来源不同的数据请求分配不同的缓存标签，可以实现来自侦听流水线交换端口31的数据请求和来自一致性互连侦听请求器37的数据请求的并发，避免一致性互连和处理器本地总线之间的冲突，从而可以缩短内存读写延迟。处理器缓存控制器323用于控制来自侦听流水线交换端口31的数据请求，侦听控制器324用于控制来自一致性互连侦听请求器37的数据请求。

[0052] 图4中以CPU chiplet架构的每个CPU具有8M字节(byte)的L3缓存总线为例。每个高速缓存段中的高速缓存为32M byte的静态随机存取存储器(Static Random-Access Memory，SRAM)。假设每个CPU的L3缓存总线是一个8M byte的4路关联缓存，则L3标签(TAG)条目将为A[20:6]，缓存TAG为A[39:21]。若每个缓存段是32M byte的4路关联缓存，则TAG条目将为A[22:6]，高速缓存段条目为A[24:23]，缓存TAG为A[39:25]。

[0053] 侦听流水线交换端口31的性能将会影响多处理器架构中多个处理器的性能扩展。

下面将对本申请实施例提供的用于支持多处理器架构的系统缓存架构进行具体的读写请求时的处理流程进行进一步详细说明。

[0054] 首先，本申请实施例提供的用于支持多处理器架构的系统缓存架构可以对来自侦听流水线交换端口31的内存读写请求进行处理，内存读写请求包括内存读取请求或内存写入请求。

[0055] 侦听流水线交换端口31在将来自任一处理器的内存读写请求通过内存请求仲裁器36转发至内存系统或者将内存读写请求发送至至少两个高速缓存段中的任一高速缓存段之前，确定内存读写请求是否位于可缓存周期，若是则将内存读写请求发送至至少两个高速缓存段中的任一高速缓存段，若否则将内存读写请求通过内存请求仲裁器36转发至内存系统。也就是说，侦听流水线交换端口31在获取来自任一处理器的内存读写请求之后，首先判断该内存读写请求周期是否为可缓存周期，若不是可缓存周期那么将不通过高速缓存段而直接将内存读写请求通过内存请求仲裁器36转发至内存系统。若内存读写请求周期为可缓存周期，那么选择任一高速存储段进行该内存读写请求的处理。

[0056] 若内存读写请求为内存读取请求，那么侦听流水线交换端口31根据来自第一处理器的内存读取请求在第一高速缓存段中查询对应的第一缓存数据。第一处理器为多处理器架构中的任一处理器，第一高速缓存段为至少两个高速缓存段中任一空闲高速缓存段，空闲高速缓存段表示没有被占用的高速缓存段。若内存读取请求在第一高速缓存段中缓存未命中，也就是内存读取请求的数据未缓存在第一高速缓存段中，则通过内存请求仲裁器36从内存系统中获取对应的第一缓存数据发送至第一处理器并存储至第一高速缓存段。若内存读取请求在第一高速缓存段中缓存命中，则表示第一高速缓存段中缓存有内存读取请求对应的第一缓存数据，而此时还需要考虑第一高速缓存段中缓存的数据是否为脏数据。脏数据表示最新的数据，由于多处理器架构中的各处理器可能进行并发的数据读取请求，那么多个处理器可能并发请求读取相同的数据，此时数据读取请求可能具有多个对应的数据，因此需要从第一处理器以外的其他处理器中侦听第一缓存数据对应的脏数据。若在第一处理器以外的其他处理器侦听到第一缓存数据对应的脏数据则将脏数据发送至第一处理器并刷新第一高速缓存段中的数据。这样就能够使第一处理器发送的数据读取请求能够读取到最新的数据。

[0057] 若来自第一处理器的内存读取请求在第一高速缓存段中查询对应的第一缓存数据时缓存命中，且从第一处理器以外的其他处理器中未侦听到第一缓存数据对应的脏数据，则侦听流水线交换端口31将第一高速缓存段中的第一缓存数据发送至第一处理器。也就是说，在第一高速缓存段中缓存命中，且没有在第一处理器以外的其他处理器中侦听到对应的脏数据，那么第一高速缓存段中的存储的第一缓存数据就是脏数据，也就是最新数据，可以直接发送给第一处理器。

[0058] 若内存读写请求为内存写入请求，内存写入请求包括部分写入请求或全部写入请求，表示该内存写入请求是否占据整个内存周期。若侦听流水线交换端口31确定内存写入请求为部分写入请求时，在第一高速缓存段中存储内存写入请求，并向第一处理器以外的其他处理器发送侦听请求阻止其他处理器更新内存写入请求所在缓存行并将内存写入请求通过内存请求仲裁器36转发至内存系统。也就当第一处理器发送的内存写入请求为部分写入请求，未占满一个内存周期时，为了避免并发的其他内存读写请求对该内存写入请求

产生冲突，侦听流水线交换端口31向其他处理器发送侦听请求，避免该内存写入请求所在缓存行被其他处理器所使用。若侦听流水线交换端口31确定内存写入请求为全部写入请求时直接将内存写入请求通过内存请求仲裁器36转发至内存系统，若内存写入请求为全部写入请求则不会被其他处理器占用内存写入请求所在缓存行。

[0059] 另外，侦听流水线交换端口31将内存写入请求通过内存请求仲裁器36转发至内存系统之后，还将第一高速缓存段中存储的内容写入请求标记为脏数据。

[0060] 另外，本申请实施例提供的用于支持多处理器架构的系统缓存架构还可以对来自一致性互连侦听请求器37的侦听读写请求进行处理，侦听读写请求包括侦听读取请求或侦听内存写入请求。

[0061] 当侦听读写请求为侦听读取请求时，接收到侦听读取请求的第一高速缓存段，用于确定侦听读写请求是否缓存命中，若未命中则不响应侦听读取请求。也就是说，侦听读写请求对应的数据若在第一高度缓存段中未缓存，则第一高速缓存段不响应该侦听读写请求，也不会中断多处理器架构中各处理器的本地缓存总线。

[0062] 第一高速缓存段在接收到侦听读取请求时，若确定侦听读取请求缓存命中时，也就是第一高速缓存段中存储有侦听读取请求对应的数据，则还需要确定与侦听读取请求对应的第二缓存数据是否为脏数据，若是则反馈第二缓存数据并将第二缓存数据标记为干净数据。也就是说，若第一高速缓存段中存储有侦听读取请求对应的第二缓存数据且为最新数据，则将改数据反馈给侦听读取请求的请求端，并且由于该数据已经被读取需要将其标记为干净数据。若第一高速缓存段中存储的第二缓存数据不为脏数据，那么需要通过侦听流水线交换端口31在至少两个处理器中查询与侦听读取请求对应的脏数据并将脏数据缓存并反馈。这样能够最大限度地避免来自一致性互连的数据请求对多处理器架构的各处理器的本地缓存总线产生影响。

[0063] 第一高速缓存段在接收到侦听读取请求时，确定侦听读取请求缓存命中时，确定侦听读取请求对应的第二缓存数据为干净数据时，若通过侦听流水线交换端口31在至少两个处理器中未查询到与侦听读取请求对应的脏数据则将反馈第二缓存数据。也就是说，虽然第一高速缓存段中存储的第二缓存数据为干净数据，但各处理器中也并未查询到与侦听读取请求对应的脏数据，则直接反馈第二缓存数据。

[0064] 当侦听读写请求为侦听写入请求时，接收到侦听写入请求的第一高速缓存段，首先确定侦听读写请求是否缓存命中，若未命中则将侦听写入请求写入内存系统目录。

[0065] 第一高速缓存段在接收到侦听写入请求时，确定侦听写入请求缓存命中时，当确定侦听写入请求为脏数据命中时则将侦听写入请求写入内存系统目录，当确定侦听写入请求为干净数据命中时将侦听写入请求通过侦听流水线交换端口31发送至至少两个处理器。侦听写入请求可能也是并发的，因此侦听写入请求同样可能存在脏数据，若侦听写入请求为脏数据意味着是一个独占缓存行，那么第一高速缓存段将在不干扰各处理器本地缓存总线的情况下直接将其写入内存系统目录并使缓存行失效。而若侦听写入请求为干净数据，那么该侦听写入请求可能为共享的缓存行，此时将侦听写入请求通过侦听流水线交换端口31发送至各处理器。

[0066] 第一高速缓存段在确定侦听写入请求缓存命中时，将侦听写入请求写入内存系统目录或发送至至少两个处理器之后，将命中的缓存行失效。

[0067] 上述图2-图4所示用于支持多处理器架构的系统缓存架构,可以服务于由多个处理器组成的处理器集群,也可以服务于由多个CPU芯粒组成的D2D互连结构。

[0068] 下面以几个具体实施例对本申请实施例提供的用于支持多处理器架构的系统缓存架构进行进一步说明。

[0069] 图5为本申请实施例提供的用于支持多处理器架构的系统缓存架构进行AXI内存读取的流程图,如图5所示,对于来自处理器的内存读取请求,首先侦听流水线交换端口会检查内存读取请求周期是否为可缓存周期。如果不是可缓存周期,侦听流水线交换端口将直接将内存读取转发到一致性互连而不检查 高速缓存段。如果内存读取请求周期是可缓存周期,则侦听流水线交换端口将执行下一个总线操作。当内存读取请求周期为可缓存周期时,该周期是来自多处理器架构最后一级内存总线的缓存未命中周期。多处理器架构最后一级内存总线须获取一个新的缓存行,无论它是读未命中还是写未命中。因此,侦听流水线交换端口向多处理器架构的其他处理器断言一个侦听周期 ReadNotShareDirty。同时侦听流水线交换端口也向任一高速缓存段断言一个高速缓存读取周期。根据缓存包含原则,至少两个高速缓存段中应该包括所有处理器当前驻留的缓存。侦听流水线交换端口首先检查高速缓存段中双端口缓存标签的结果。如果是缓存未命中,则侦听流水线交换端口将AXI内存读取周期断言到一致性互连,以从系统内存中获取所需的缓存行。这是因为高速缓存段缓存未命中意味着由于缓存包含原理,其他处理器将没有所需的缓存行。如果高速缓存段命中缓存,侦听流水线交换端口接下来会检查侦听响应是否存在其他处理器中存在的任何脏缓存行。如果其他处理器中不存在脏缓存行,侦听流水线交换端口将缓存行从高速缓存段返回到启动内存读取请求的处理器并完成循环。如果其中一个处理器返回脏缓存行存在的侦听响应,侦听流水线交换端口将等待处理器刷新脏缓存行。当侦听流水线交换端口收到脏缓存行时,侦听流水线交换端口将脏缓存行返回给启动内存读取请求的处理器。侦听流水线交换端口同时断言高速缓存写入以更新高速缓存段中的高速缓存行。执行完这两个总线周期后,侦听流水线交换端口完成处理器的内存读取。当高速缓存段发生缓存未命中时,需要替换缓存行。当高速缓存段替换现有的缓存行时,高速缓存段必须向被替换的缓存行发出读取唯一通知(readUnique)。在这种情况下,所有其他处理器将不会包含被替换缓存的数据(如果有的话)。这就是缓存包含原理的执行方式,如果缓存行中有脏数据,各处理器会刷掉脏数据。

[0070] 图6为本申请实施例提供的用于支持多处理器架构的系统缓存架构进行AXI内存写入的流程图,如图6所示,侦听流水线交换端口会检查内存写周期是否为可缓存周期。如果不是可缓存周期,侦听流水线交换端口将直接将内存写入转发到一致性互连而不检查高速缓存段。如果请求写周期是可缓存的读周期,则侦听流水线交换端口将执行下一个总线操作。当内存写周期是可缓存周期时,该写周期是缓存替换时的脏缓存行刷新或者由于共享状态上的写入命中而产生的写入命中广播。脏缓存行刷新须是具有完整高速缓存线的写入周期,通常为 64 字节。对于共享高速缓存行上的写命中,写周期必须是部分写周期,对于 64 位处理器,通常为 8 字节。一旦侦听流水线交换端口检测到可缓存写入,接下来就是检测是否是部分缓存行写入周期。如果是部分缓存行写入,侦听流水线交换端口将向所有其他处理器上的共享缓存行发出监听请求MakeInvalid。这使得其他共享缓存无法更新,并且启动内存写入请求的处理器保持最新数据并将缓存行标记为脏且唯一。部分高速缓存

行写入时，不需要对高速缓存段做任何循环。高速缓存段仍然包含缓存行，但最新的更新数据在 处理器中。如果是全部缓存写入，一定是由于刷新了脏缓存行。在缓存算法中，多处理器架构中的所有处理器只能存在一条脏缓存行。因此，脏缓存行刷新不应涉及其他出全力中的任何侦听。因此，侦听流水线交换端口将断言对高速缓存段的内存写入并将高速缓存段中的高速缓存行标记为脏。

[0071] 图7为本申请实施例提供的用于支持多处理器架构的系统缓存架构响应来自DMA主机的侦听读取请求的流程图，如图7所示，当 DMA内存主机访问可缓存内存空间时，总线传输周期应基于每次传输一个缓存线。例如，对于 DMA主机传输1K byte，DMA内存主机应为每次传输断言16次64字节传输。这是为了确保一致性互连、高速缓存段和DRAM之间的有效交互。一致性互连应将来自DMA主机的侦听读取请求转换为与64字节边界高速缓存行读取对齐。当高速缓存段接收到侦听读取请求时，一致性互连可以通过其双端口缓存标签侦听高速缓存段，而不会中断处理器本地总线。高速缓存段将通过侦听读取请求检查其双端口缓存比较结果。如果是侦听未命中，高速缓存段将响应一致性互连作为侦听未命中，然后没有操作并且一致性互连不会中断处理器本地总线。 如果是侦听命中，高速缓存段会检查这是否是一个脏缓存行命中。由于系统只允许在整个缓存系统内复制一份脏数据，因此具有相同缓存标签的处理器的其余部分应该处于无效状态。 高速缓存段会将脏缓存行刷新到一致性互连并将缓存行标记为干净，此操作也不会中断处理器本地总线。如果这是对高速缓存段 的干净命中，则高速缓存段需要检查处理器上是否存在脏缓存行。高速缓存段会将 ReadNotSharedDirty 的侦听转发到侦听流水线交换端口。侦听流水线交换端口将进一步对所有处理器进行侦听查询。如果所有处理器在其本地缓存上都没有响应脏缓存行，则高速缓存段将其缓存行返回到一致性互连。如果一个 处理器响应脏缓存行刷新，则侦听流水线交换端口在更新高速缓存段缓存行时将脏缓存行返回到一致性相干互连，从而完成侦听读取请求。

[0072] 图8为本申请实施例提供的用于支持多处理器架构的系统缓存架构响应来自DMA主机的侦听写入读取请求的流程图，如图8所示，当高速缓存段接收到侦听写入请求时，一致性互连可以通过其双端口高速缓存段缓存标签侦听 高速缓存段，而不会中断处理器本地总线。如果高速缓存段缓存上侦听未命中，则高速缓存段会响应侦听未命中，并且一致性互连将写入系统内存目录，而不会干扰处理器本地总线。如果高速缓存段上侦听命中，则高速缓存段响应侦听命中。高速缓存段会进一步识别是否是脏缓存行命中。由于一致性互连将所有部分写入或未对齐写入转换为对齐和高速缓存行突发写入，因此一致性互连将侦听并使高速互联段中的高速缓存无效。如果高速缓存段中的侦听命中是脏缓存行，则意味着这是一个独占缓存行。所以高速缓存段会在不干扰处理器本地总线的情况下使缓存行失效。如果和高速缓存段是一个干净的缓存行，那么其他处理器中可能存在共享缓存行。因此高速缓存段将向侦听流水线交换端口断言 MakeInvalid 并转发到所有处理器。高速缓存段也会使缓存行失效。当DMA 主机将DMA写周期置为一致性互连时，这将完成侦听写操作。

[0073] 图5-图 8 中，用于支持多处理器架构的系统缓存架构可以提供多个处理器的最小中断，每个处理器中有多个处理器内核，侦听流水线交换端口可以实现对称多处理任务执行的短内存延迟。

[0074] 本申请实施例还提供一种芯片，该芯片包括至少两个芯粒和用于支持多处理器架

构的系统缓存架构,至少两个芯粒通过用于支持多处理器架构的系统缓存架构与一致性互连连接,用于支持多处理器架构的系统缓存架构为如图2-图4所示实施例的用于支持多处理器架构的系统缓存架构。

[0075] 本申请实施例还提供一种处理器集群组件,该处理器集群组件包括由至少两个处理器组成的处理器集群和用于支持多处理器架构的系统缓存架构,至少两个处理器通过用于支持多处理器架构的系统缓存架构与一致性互连连接,用于支持多处理器架构的系统缓存架构为如图2-图4所示实施例的用于支持多处理器架构的系统缓存架构。

[0076] 一般来说,本申请的多种实施例可以在硬件或专用电路、软件、逻辑或其任何组合中实现。例如,一些方面可以被实现在硬件中,而其它方面可以被实现在可以被控制器、微处理器或其它计算装置执行的固件或软件中,尽管本申请不限于此。

[0077] 本申请的实施例可以通过计算机装置的数据处理器执行计算机程序指令来实现,例如在处理器实体中,或者通过硬件,或者通过软件和硬件的组合。计算机程序指令可以是汇编指令、指令集架构(Instruction Set Architecture,ISA)指令、机器指令、机器相关指令、微代码、固件指令、状态设置数据、或者以一种或多种编程语言的任意组合编写的源代码或目标代码。

[0078] 本申请附图中的任何逻辑流程的框图可以表示程序步骤,或者可以表示相互连接的逻辑电路、模块和功能,或者可以表示程序步骤与逻辑电路、模块和功能的组合。计算机程序可以存储在存储器上。存储器可以具有任何适合于本地技术环境的类型并且可以使用任何适合的数据存储技术实现,例如但不限于只读存储器(Read-Only Memory,ROM)、随机访问存储器(Random Access Memory,RAM)、光存储器装置和系统(数码多功能光碟(Digital Video Disc,DVD)或光盘((Compact Disc,CD))等。计算机可读介质可以包括非瞬时性存储介质。数据处理器可以是任何适合于本地技术环境的类型,例如但不限于通用计算机、专用计算机、微处理器、数字信号处理器(Digital Signal Processing,DSP)、专用集成电路(Application Specific Integrated Circuit,SAIC)、可编程逻辑器件(Field-Programmable Gate Array,FGPA)以及基于多核处理器架构的处理器。

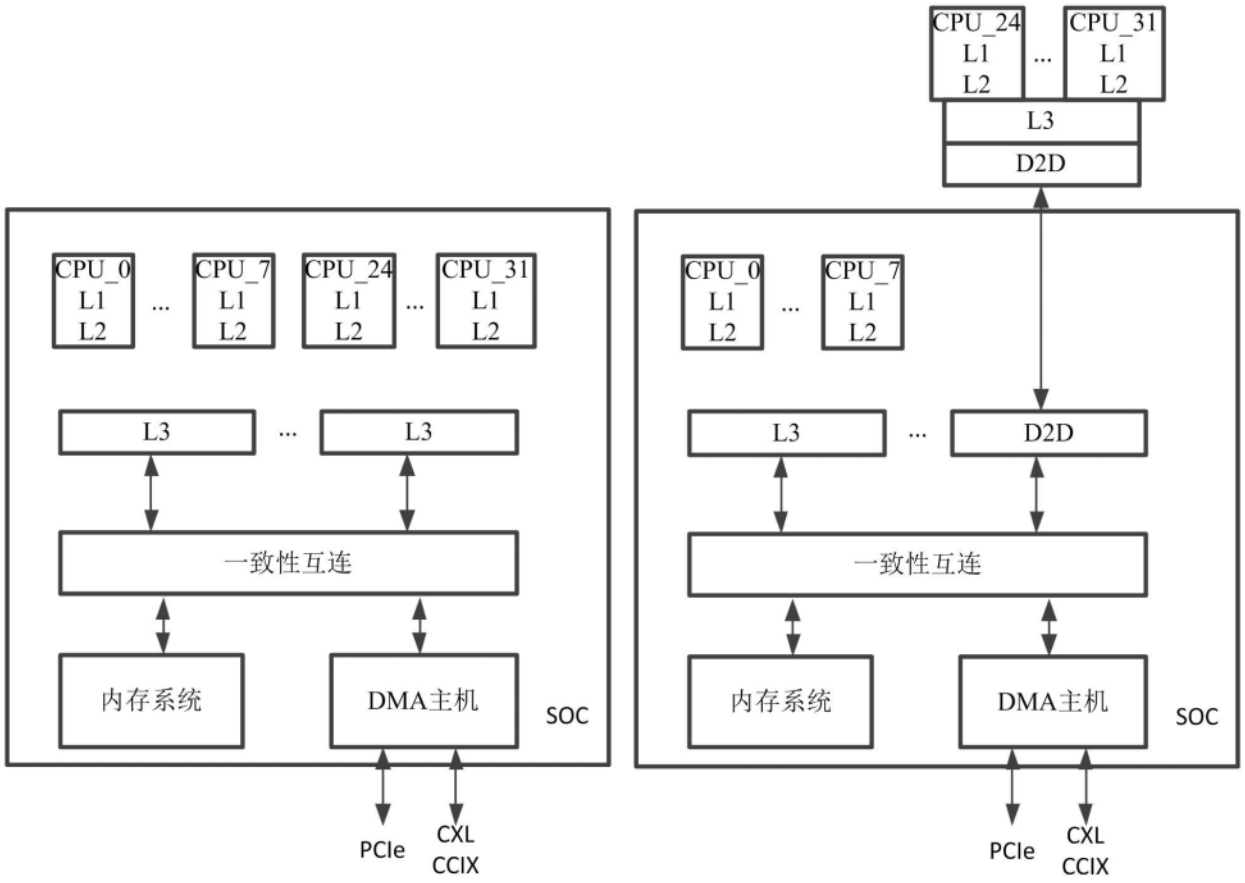

CPU_0
L1
L2
CPU_7
L1
L2
CPU_24
L1
L2
CPU_31
L1
L2
L3
L3
一致性互连
内存系统
DMA主机
SOC
PCIe
CXL
CCIX
嵌入式CPU系统
D2D
CPU chiplet系统

图1

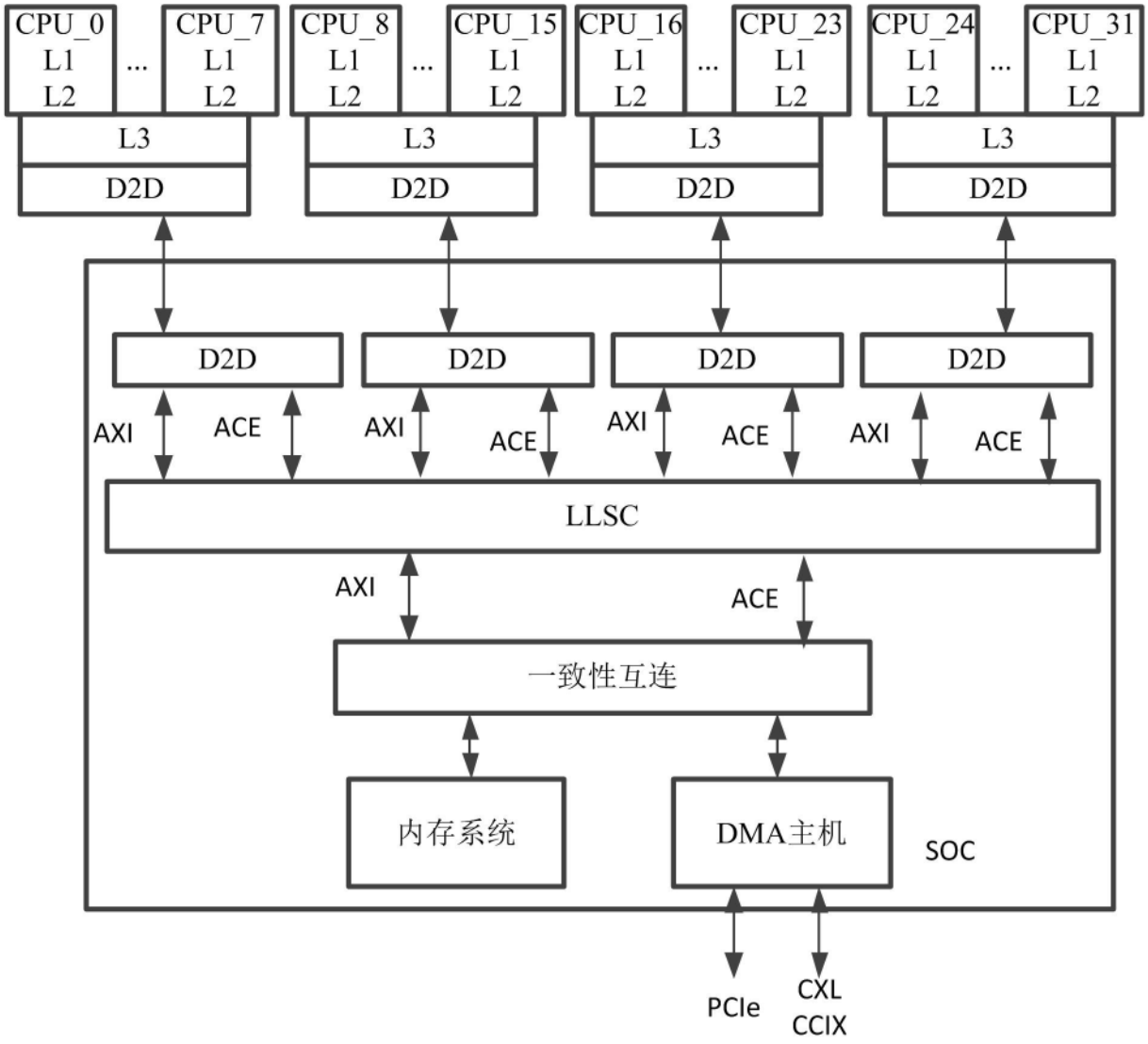

CPU_0
L1
L2
...
CPU_7
L1
L2
CPU_8
L1
L2
...
CPU_15
L1
L2
CPU_16
L1
L2
...
CPU_23
L1
L2
CPU_24
L1
L2
...
CPU_31
L1
L2
L3
D2D
L3
D2D
L3
D2D
L3
D2D
D2D
D2D
D2D
D2D
AXI
ACE
AXI
ACE
AXI
ACE
AXI
ACE
LLSC
AXI
ACE
一致性互连
内存系统
DMA主机
SOC
PCIe
CXL
CCIX

图2

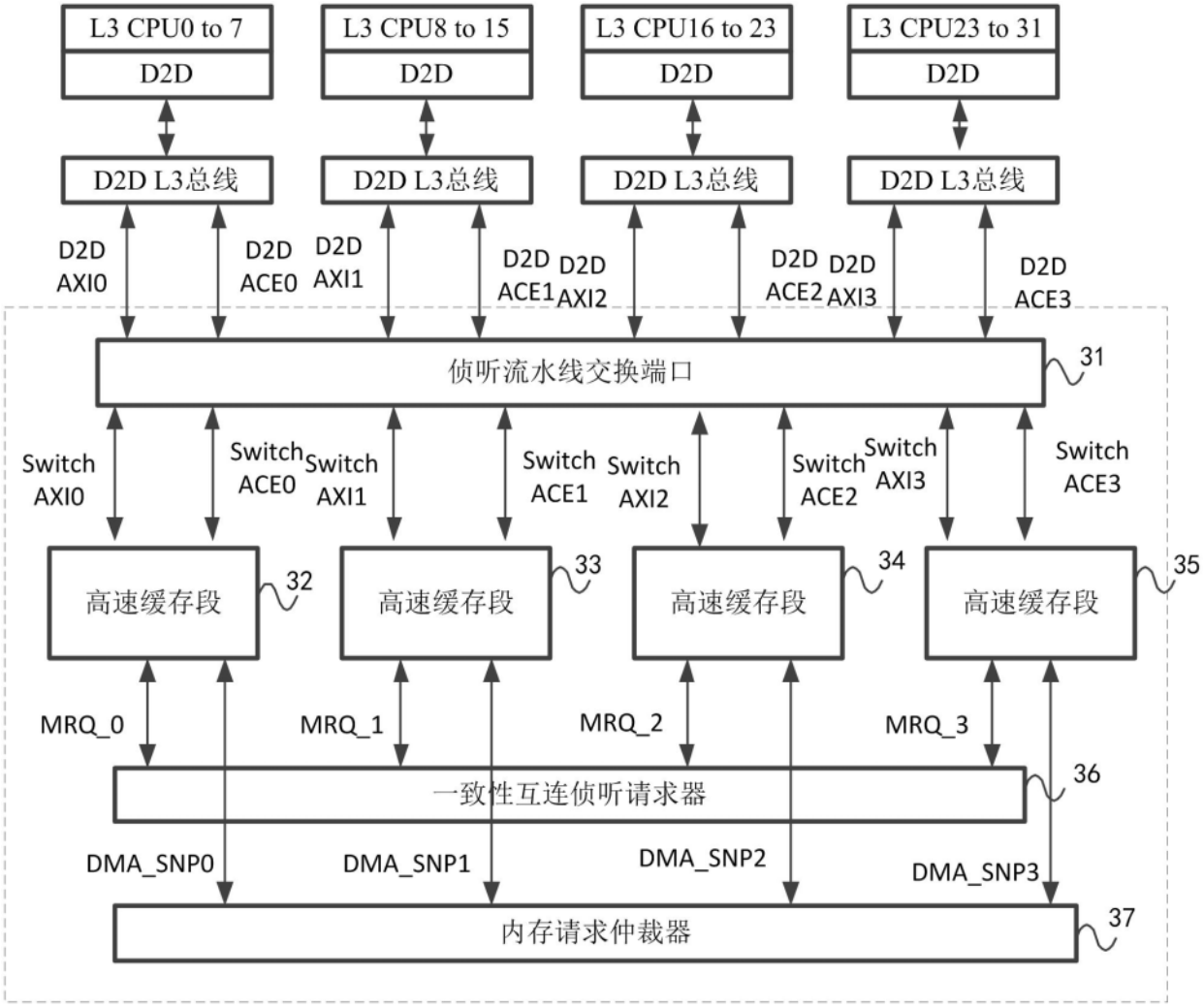

L3 CPU0 to 7
D2D
L3 CPU8 to 15
D2D
L3 CPU16 to 23
D2D
L3 CPU23 to 31
D2D
D2D L3总线
D2D L3总线
D2D L3总线
D2D L3总线
D2D AXI0
D2D ACE0
D2D AXI1
D2D ACE1
D2D AXI2
D2D ACE2
D2D AXI3
D2D ACE3
侦听流水线交换端口
31
Switch AXI0
Switch ACE0
Switch AXI1
Switch ACE1
Switch AXI2
Switch ACE2
Switch AXI3
Switch ACE3
高速缓存段
32
高速缓存段
33
高速缓存段
34
高速缓存段
35
MRQ_0
MRQ_1
MRQ_2
MRQ_3
一致性互连侦听请求器
36
DMA_SNP0
DMA_SNP1
DMA_SNP2
DMA_SNP3
内存请求仲裁器
37

图3

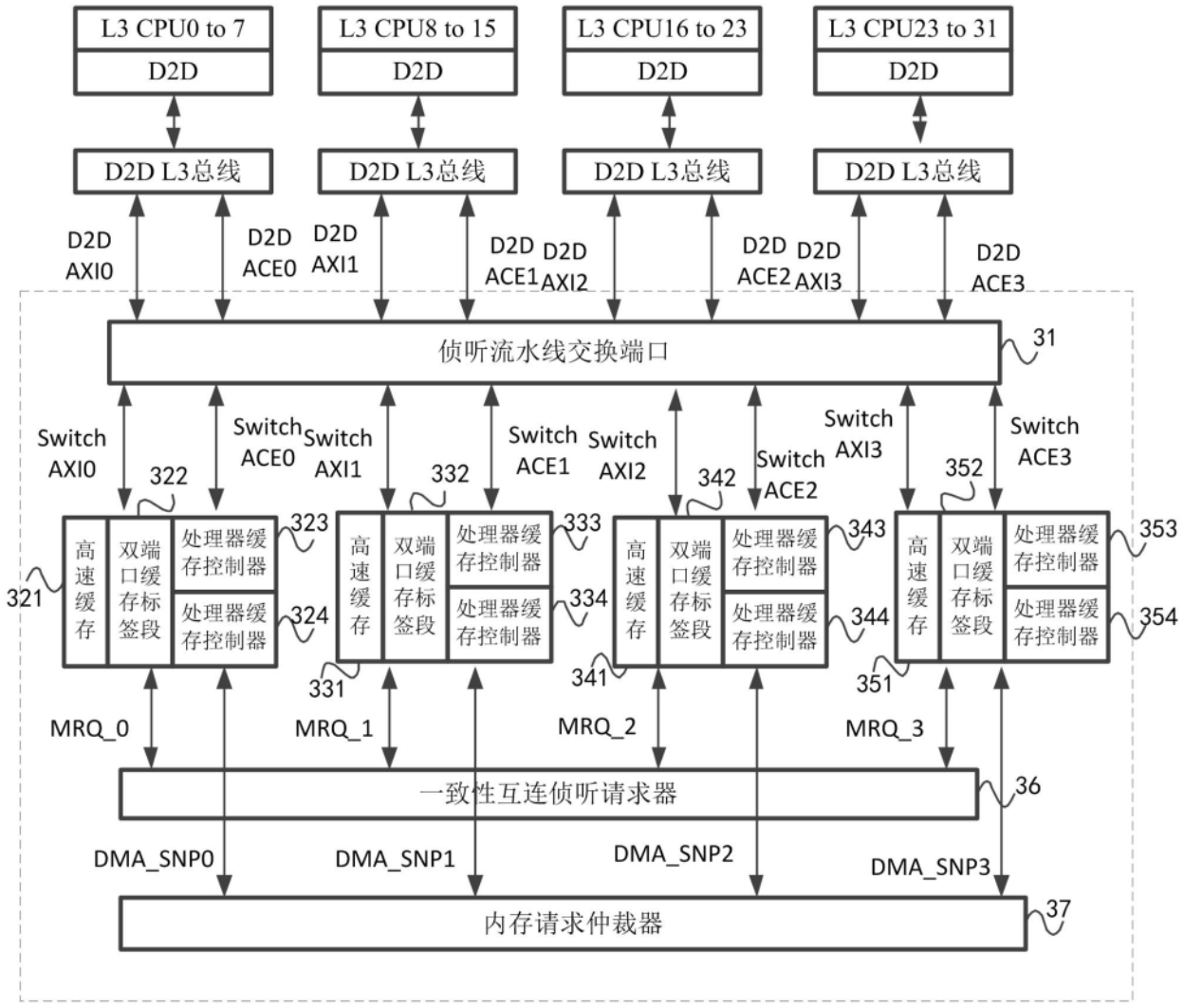

L3 CPU0 to 7
D2D
L3 CPU8 to 15
D2D
L3 CPU16 to 23
D2D
L3 CPU23 to 31
D2D
D2D L3总线
D2D L3总线
D2D L3总线
D2D L3总线
D2D AXI0
D2D ACE0
D2D AXI1
D2D ACE1
D2D AXI2
D2D ACE2
D2D AXI3
D2D ACE3
侦听流水线交换端口
31
Switch AXI0
Switch ACE0
Switch AXI1
Switch ACE1
Switch AXI2
Switch ACE2
Switch AXI3
Switch ACE3
322
332
342
352
高速缓存
双端口缓存标签段
处理器缓存控制器
处理器缓存控制器
321
323
324
333
334
343
344
353
354
331
341
351
MRQ_0
MRQ_1
MRQ_2
MRQ_3
一致性互连侦听请求器
36
DMA_SNP0
DMA_SNP1
DMA_SNP2
DMA_SNP3
内存请求仲裁器
37

图4

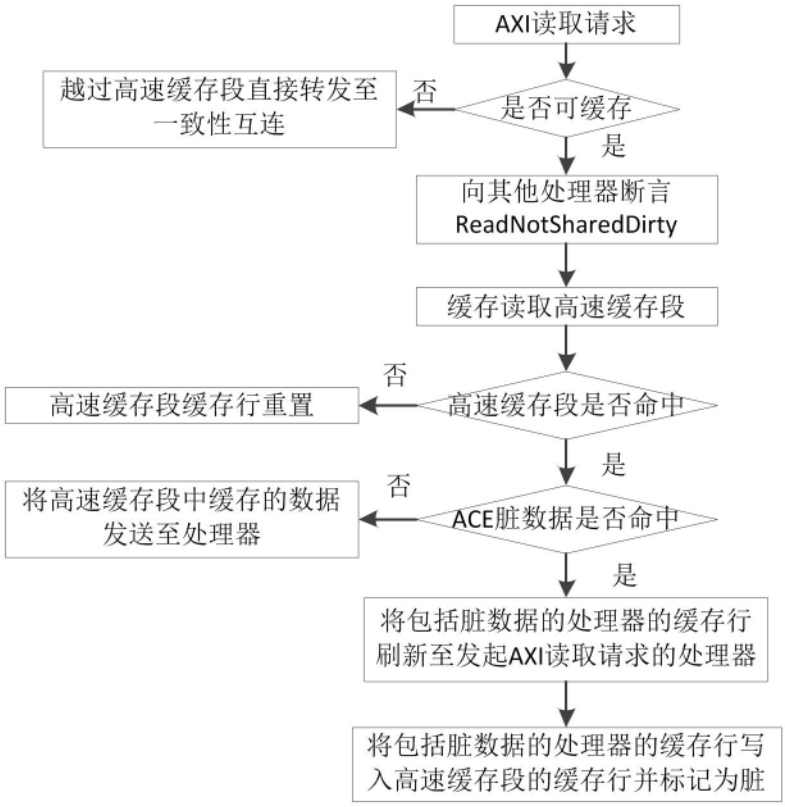

AXI读取请求
是否可缓存
否
越过高速缓存段直接转发至一致性互连
是
向其他处理器断言ReadNotSharedDirty
缓存读取高速缓存段
高速缓存段是否命中
否
高速缓存段缓存行重置
是
ACE脏数据是否命中
否
将高速缓存段中缓存的数据发送至处理器
是
将包括脏数据的处理器的缓存行刷新至发起AXI读取请求的处理器
将包括脏数据的处理器的缓存行写入高速缓存段的缓存行并标记为脏

图5

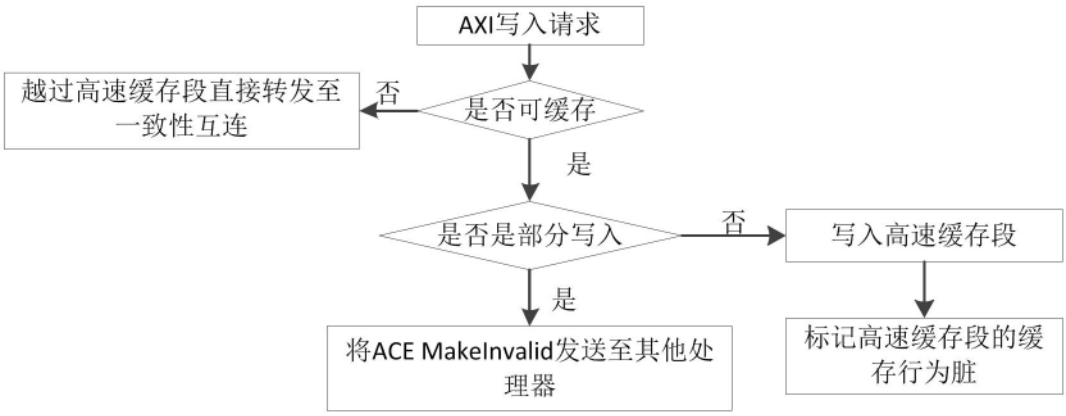

AXI写入请求
是否可缓存
否
越过高速缓存段直接转发至一致性互连
是
是否是部分写入
否
写入高速缓存段
是
将ACE MakeInvalid发送至其他处理器
标记高速缓存段的缓存行为脏

图6

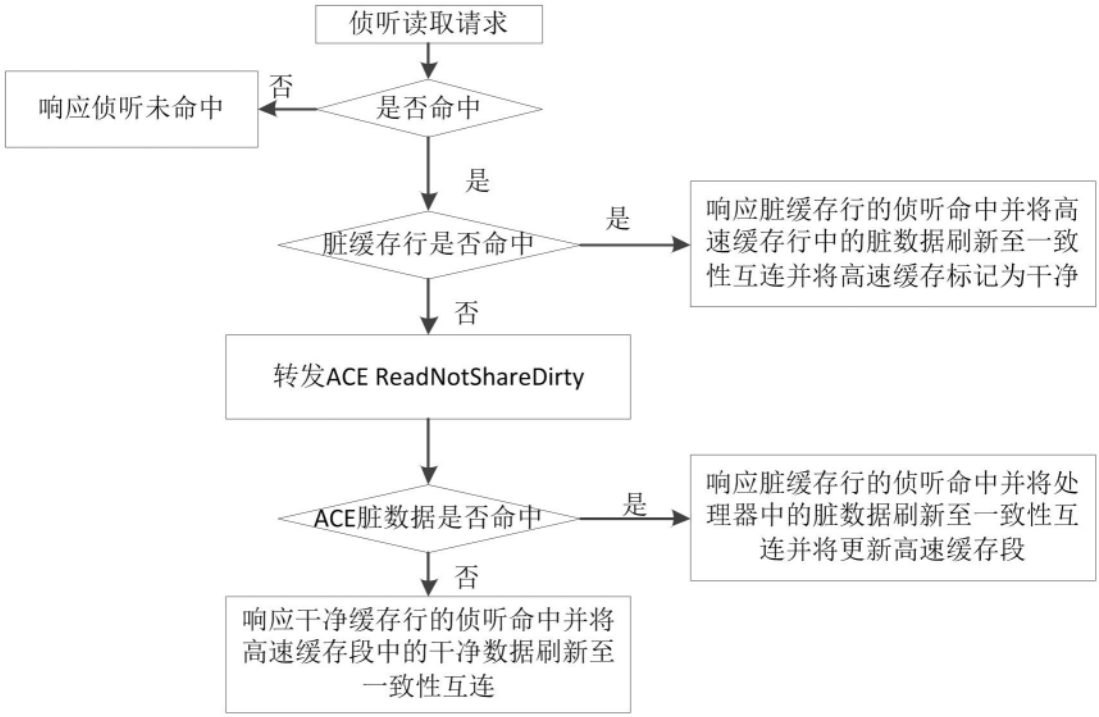

侦听读取请求
是否命中
否
响应侦听未命中
是
脏缓存行是否命中
是
响应脏缓存行的侦听命中并将高速缓存行中的脏数据刷新至一致性互连并将高速缓存标记为干净
否
转发ACE ReadNotShareDirty
ACE脏数据是否命中
是
响应脏缓存行的侦听命中并将处理器中的脏数据刷新至一致性互连并将更新高速缓存段
否
响应干净缓存行的侦听命中并将高速缓存段中的干净数据刷新至一致性互连

图7

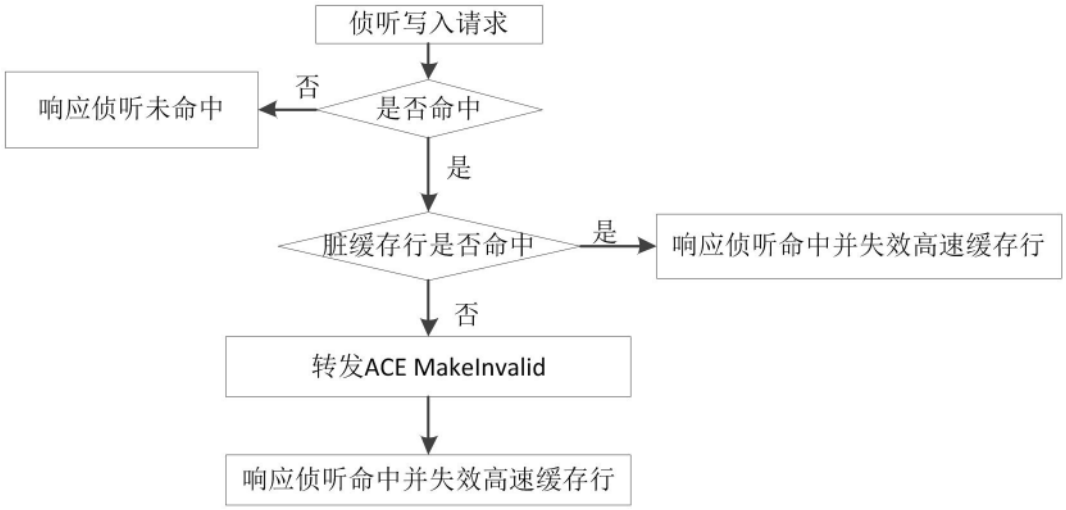

侦听写入请求
是否命中
否
响应侦听未命中
是
脏缓存行是否命中
是
响应侦听命中并失效高速缓存行
否
转发ACE MakeInvalid
响应侦听命中并失效高速缓存行

图8