(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 103049422 A
(43)申请公布日 2013.04.17

(21)申请号 201210544976.6

(22)申请日 2012.12.17

(71)申请人 浪潮电子信息产业股份有限公司
地址 250014 山东省济南市高新区舜雅路 1036 号

(72)发明人 王恩东 胡雷钧 陈继承 张东 公维锋 张峰

(51)Int.Cl.
G06F 15/167(2006.01)

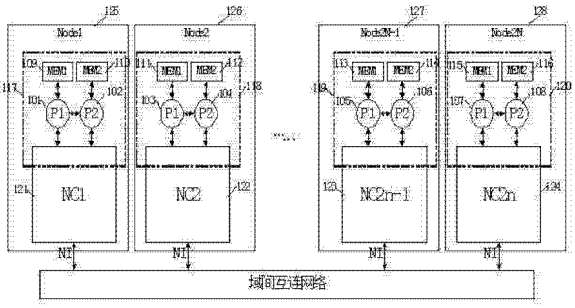
权利要求书 2 页 说明书 9 页 附图 7 页

(54)发明名称

一种具有多 cache 一致性域的多处理器节点系统构建方法

(57)摘要

本发明提供一种具有多 cache 一致性域的多处理器节点系统构建方法，在本系统中，节点控制器内建立的目录需包含处理器域属信息，该信息可通过配置节点控制器与处理器相连的各个端口的 cache 一致性域属性获得。本发明可以使一个节点控制器支持节点内多个物理 cache 一致性域。其目的在于减少多处理器计算机系统中节点控制器的数量，减小节点间互连的规模，降低节点间拓扑复杂度，以提升系统效率；同时也可以解决处理器互连端口数目和能够支持的域内处理器 ID 数量非常有限对构建大规模 CC-NUMA 系统带来的性能瓶颈问题。



CN 103049422 A

1. 一种具有多 cache 一致性域的多处理器节点系统构建方法，其特征在于在多处理器节点系统中，在每个节点的控制器内部构建一个统一的逻辑 cache 一致性域，然后由若干处理器与该节点控制器组成相互隔离的多物理 cache 一致性域；包括 cache 一致性域的组建方式、cache 一致性域划分方式和目录结构方式；其中

（1）多物理 cache 一致性域的组建方式包括：

1）一级节点间通过节点控制器直接连接或经过节点路由器连接，组成一级节点间的单一 cache 一致性域；

2）通过一级节点控制器直接连接二级节点控制器，并在二级节点控制器内部构建一个统一的逻辑 cache 一致性域，该 cache 一致性域完全包含彼此隔离的由若干一级节点控制器和二级节点控制器组成的多物理 cache 一致性域；

3）二级节点间通过二级节点控制器直接连接或经过节点路由器连接，组成二级节点间的单一 cache 一致性域；

4）基于 1）-3）的方式扩展到 n（n>2）级节点，从而实现更多级节点多处理器系统；

（2）cache 一致性域划分方式包括：

1）对于一级节点控制器，与处理器相连接的各个端口信息，作为由若干处理器和该节点控制器组成的物理 cache 一致性域域属信息来源，每个端口域信息寄存器标识该端口的物理子域，端口域信息寄存器根据系统要求配置各端口域属信息，从而实现由若干处理器和该节点控制器组成的各个物理 cache 一致性域划分；

2）对于二级节点控制器，与一级节点控制器相连接的各个端口信息作为由若干一级节点控制器和该二级节点控制器组成的物理 cache 一致性域域属信息来源，根据系统要求配置各端口域属信息从而实现由若干一级节点控制器和该二级节点控制器组成的各个物理 cache 一致性域划分；

3）采用方式 2）描述的二级节点控制器端口域属配置方式配置其 n 级（n>2）节点控制器端口域属；

（3）目录结构方式

1）一级节点控制器采用本地和远端双目录结构，对于远端数据目录，目录项记录本地节点内持有统一数据副本的处理器及其域属信息、一致性状态信息；而对于本地数据目录，目录项记录持有数据副本的远端节点及其一致性状态信息，同时设置标志位指示本地节点内不同于数据所在物理子域的其它子域是否持有该数据副本；

2）二级节点控制器采用本地和远端双目录结构，对于远端数据目录，目录项记录本地二级节点内持有统一数据副本的一级节点及其域属信息、一致性状态信息；而对于本地数据目录，目录项记录持有统一数据副本的远端二级节点及其一致性状态信息，同时设置标志位指示本地二级节点内不同于数据所在物理子域的其它子域是否持有该数据副本；

3）采用方式 2）描述的二级节点控制器目录结构方式配置其 n 级（n>2）节点控制器目录结构；

4）一级节点控制器采用本地 / 远端统一目录结构，目录项记录除本物理子域的处理器外，系统所有处理器持有统一数据副本信息及一致性状态信息；

5）二级节点控制器采用本地 / 远端统一目录结构，目录项记录除本物理子域的一级节点控制器外，系统所有一级节点控制器持有统一数据副本信息及一致性状态信息；

6）采用方式 5）的二级节点控制器目录结构方式配置其 n 级（n>2）节点控制器目录结构。

一种具有多 cache 一致性域的多处理器节点系统构建方法

技术领域

[0001] 本发明涉及计算机应用领域，具体地说是一种具有多 cache 一致性域的多处理器节点系统构建方法 .

背景技术

[0002] 对于多处理器系统，其多个处理器共享系统内存空间。当前，多处理器之间的连接方式由总线连接转变为点对点连接，内存也从挂接处理器外部桥接芯片转变为直接挂接处理器。由于内存挂接方式的变化，内存在系统中的分布也发生变化，从而导致多处理器系统中内存访问的非均一性，故当前多处理器系统多为 NUMA(Non-Uniform Memory Access——非一致内存访问) 架构系统。

[0003] NUMA 架构多处理器系统具有以下 3 个重要特点 :

1、所有内存进行统一编址，形成一个统一的内存空间 ;

2、所有处理器都能访问内存空间的全部地址 ;

3、访问远端内存比访问其本地内存慢。

[0004] NUMA 系统中有多个 cache 单元分布于系统，因而设计 NUMA 系统须解决多 cache 之间的一致性问题。满足 cache 一致性的 NUMA 系统也称为 CC-NUMA (Cache Coherent Non-Uniform Memory Access——Cache 一致性非一致内存访问)系统。如何解决 cache 一致性问题是 CC-NUMA 系统的核心问题。

[0005] 由于当今处理器直接挂接内存，其本身支持 cache 一致性协议 ;因而一种方案是将这些处理器直连构成多处理器系统，处理器之间的 cache 一致性可以由这些处理器本身的 cache 一致性协议维护引擎保证，并形成一个单一 cache 一致性域。在单一 cache 一致性域之中的各个处理器用处理器 ID 号标识和识别。但这种方式组织的多处理器系统规模有限，这是因为每个处理器在 cache 一致性域内至少占用一个处理器 ID 号，而每个处理器能够分辨的处理器 ID 号个数是受限的。例如，某款处理器能够分辨 4 个处理器 ID 号，即能够支持域内最多 4 颗处理器直接互连。再例如，某款处理器只能分辨 2 个处理器 ID，其所能支持的 cache 一致性域内的处理器数仅为两个。同时，由于物理限制和价格限制，处理器的互连端口数量同样受限，在某些情况下，即使处理器支持的单一 cache 一致性域内处理器 ID 数目能够满足要求，但直连的方式会带来跨处理器内存访问较大的跳步数和延迟，并不能构成一个高效的多处理器系统。

[0006] 处理器参数配置、互连端口数目和能够支持的处理器 ID 数量与处理器定价体系密切相关，一般来说，处理器支持的互连端口数和处理器 ID 数量越少，价格越便宜。支持域内 2 个处理器 ID 会比支持 4 个处理器 ID 的处理器价格更低廉。

[0007] 如上文所述，按处理器直连的方式构成的多处理器系统规模有限。为实现更大规模的 CC-NUMA 多处理器系统，必须借助于节点控制器(Node Controller)。节点控制器具有扩展系统规模和维护全局 cache 一致性的功能 ;首先，每个节点控制器连接 1 至 4 颗处理器，组成一个节点和一级 cache 一致性域，域内 cache 一致性由处理器和节点控制器共同维

护。节点控制器也会占用域内的至少一个处理器 ID，因而域内处理器加节点控制器的数量不能大于处理器所能支持的域内处理器 ID 数。然后，节点控制器直接互连或通过节点路由器连接以组成大规模 CC-NUMA 系统。节点间的二级 cache 一致性由节点控制器维护，某节点内的处理器跨节点和 cache 一致性域访问另一个节点内处理器的内存时，全局 cache 一致性通过节点控制器维护。

[0008] CC-NUMA 系统使用节点控制器扩展系统规模和维护全局 cache 一致性增加了跨域处理和域间通信的开销，导致访问远端内存的显著下降，系统规模越大，下降越明显。若构建一个由 64 颗处理器组成的 CC-NUMA 系统，可采用两种方案，方案 1 是每个节点内一致性域内有 4 颗处理器，则整个系统至少需要 16 个节点控制器。方案 2 可使用仅支持域内 2 个处理器 ID 的处理器，则一个节点内 cache 一致性域只能由一颗处理器和一个节点控制器构成，这就必须使用至少 64 个节点控制器。如此多的节点控制器使节点间互连规模非常庞大，节点间拓扑愈发复杂；造成跨节点访问远端内存的速度明显恶化，进而导致系统效率的快速下降和性能的巨大损失。

[0009] 由此可见，对于多节点多处理器系统，减少节点的数目对于降低节点间互连规模、简化节点间拓扑具有直接而显著的作用，尤其是对于互连端口数目和能够支持的域内处理器 ID 数量非常有限的处理器更是如此。因此，能否有效减小节点控制器的数量是一个意义重大且又非常紧迫的技术难题。

发明内容

[0010] 本发明的目的是提供一种具有多 cache 一致性域的多处理器节点系统构建方法。

[0011] 本发明的目的是按以下方式实现的，在多处理器节点系统中，在每个节点的控制器内部构建一个统一的逻辑 cache 一致性域，然后由若干处理器与该节点控制器组成相互隔离的多物理 cache 一致性域；包括 cache 一致性域的组建方式、cache 一致性域划分方式和目录结构方式；其中：

（1）多物理 cache 一致性域的组建方式包括：

1）一级节点间通过节点控制器直接连接或经过节点路由器连接，组成一级节点间的单一 cache 一致性域；

2）通过一级节点控制器直接连接二级节点控制器，并在二级节点控制器内部构建一个统一的逻辑 cache 一致性域，该 cache 一致性域完全包含彼此隔离的由若干一级节点控制器和二级节点控制器组成的多物理 cache 一致性域；

3）二级节点间通过二级节点控制器直接连接或经过节点路由器连接，组成二级节点间的单一 cache 一致性域；

4）基于 1）-3）的方式扩展到 n（n>2）级节点，从而实现更多级节点多处理器系统；

（2）cache 一致性域划分方式包括：

1）对于一级节点控制器，与处理器相连接的各个端口信息，作为由若干处理器和该节点控制器组成的物理 cache 一致性域域属信息来源，每个端口域信息寄存器标识该端口的物理子域，端口域信息寄存器根据系统要求配置各端口域属信息，从而实现由若干处理器和该节点控制器组成的各个物理 cache 一致性域划分；

2）对于二级节点控制器，与一级节点控制器相连接的各个端口信息作为由若干一级节

点控制器和该二级节点控制器组成的物理 cache 一致性域域属信息来源，根据系统要求配置各端口域属信息从而实现由若干一级节点控制器和该二级节点控制器组成的各个物理 cache 一致性域划分；

3）采用方式 2）描述的二级节点控制器端口域属配置方式配置其 n 级（n>2）节点控制器端口域属；

（3）目录结构方式

1）一级节点控制器采用本地和远端双目录结构，对于远端数据目录，目录项记录本地节点内持有统一数据副本的处理器及其域属信息、一致性状态信息；而对于本地数据目录，目录项记录持有数据副本的远端节点及其一致性状态信息，同时设置标志位指示本地节点内不同于数据所在物理子域的其它子域是否持有该数据副本；

2）二级节点控制器采用本地和远端双目录结构，对于远端数据目录，目录项记录本地二级节点内持有统一数据副本的一级节点及其域属信息、一致性状态信息；而对于本地数据目录，目录项记录持有统一数据副本的远端二级节点及其一致性状态信息，同时设置标志位指示本地二级节点内不同于数据所在物理子域的其它子域是否持有该数据副本；

3）采用方式 2）描述的二级节点控制器目录结构方式配置其 n 级（n>2）节点控制器目录结构；

4）一级节点控制器采用本地 / 远端统一目录结构，目录项记录除本物理子域的处理器外，系统所有处理器持有统一数据副本信息及一致性状态信息；

5）二级节点控制器采用本地 / 远端统一目录结构，目录项记录除本物理子域的一级节点控制器外，系统所有一级节点控制器持有统一数据副本信息及一致性状态信息；

6）采用方式 5）的二级节点控制器目录结构方式配置其 n 级（n>2）节点控制器目录结构。

[0012] 本发明的有益效果是：本发明的目的是解决由于处理器所能支持的域内处理器 ID 数量限制而带来的节点规模受限问题，该问题会导致多处理器系统所需节点控制器数目过大，从而造成节点间互连规模庞大、拓扑结构复杂。应用本发明则可以显著减少中、大规模共享内存多处理器系统所需节点控制器数目，从而有效提升系统性能并降低系统拓扑复杂度。

附图说明

[0013] 图 1 是现有多节点多处理器系统结构示意图；

图 2 是应用本发明的多节点多处理器系统结构示意图；

图 3 是处理器访问本地节点内处于不同物理子域内存示例图；

图 4 是本地节点处理器访问远端节点内存示例图；

图 5 是处理器访问本地节点处于同一物理子域内存示例图；

图 6 是目录格式设计图；

图 7 是处理器端口物理子域属性设置示例图；

图 8 是处理器访问本节点不同物理子域内存处理流程图；

图 9 是处理器访问远端节点内存处理流程图。

具体实施方式

[0014] 参照说明书附图对本发明的方法作以下详细地说明。

实施例

[0015] 一个带多节点的多处理器系统，对于各个节点，在其节点控制器内部构建一个统一的逻辑 cache 一致性域以完全包含彼此隔离的由若干处理器和该节点控制器组成的多个物理 cache 一致性域；而其节点间可通过节点控制器直接连接或经过节点路由器(NR)连接，组成节点间单一 cache 一致性域；

另外，上述系统的一级节点控制器可接二级节点控制器，并在二级节点控制器内部构建一个统一的逻辑 cache 一致性域以完全包含彼此隔离的由若干一级节点控制器和该二级节点控制器组成的多个物理 cache 一致性域；通过其二级节点控制器直接连接或经过节点路由器(NR)连接，从而组成二级节点间单一 cache 一致性域；

更进一步，上述方案也可扩展到 n (n>2)级节点系统，从而实现更多级节点多处理器系统；

对于上述系统的节点控制器，与处理器相连接的各个端口信息可作为由若干处理器和该节点控制器组成的物理 cache 一致性域域属信息来源，每个端口的域信息寄存器标识该端口的物理子域，端口域信息寄存器可根据系统要求配置各端口域属信息从而改变由若干处理器和该节点控制器组成的各个物理 cache 一致性域划分；

而对于二级节点控制器，与一级节点控制器相连接的各个端口信息可作为由若干一级节点控制器和该二级节点控制器组成的物理 cache 一致性域域属信息来源，其可根据系统要求配置各端口域属信息从而改变由若干一级节点控制器和该二级节点控制器组成的各个物理 cache 一致性域划分；

同样，可采用系统的二级节点控制器端口域属配置方式配置其 n 级 (n>2) 节点控制器端口域属；

对于节点控制器目录构造，可采用本地和远端双目录结构，对于远端数据目录，目录项记录本地节点内持有数据副本的处理器及其域属信息、一致性状态信息；而对于本地数据目录，目录项记录持有数据副本的远端节点及其一致性状态信息，同时设置标志位指示本地节点内不同于数据所在物理子域的其它子域是否持有该数据副本；

对于节点控制器目录构造，也可采用本地 / 远端统一目录结构，目录项记录除本物理子域的处理器外系统其它所有处理器持有数据副本信息及一致性状态信息；

同样，二级节点控制器可采用本地和远端双目录结构。对于远端数据目录，目录项记录本地二级节点内持有数据副本的一级节点及其域属信息、一致性状态信息；而对于本地数据目录，目录项记录持有数据副本的远端二级节点及其一致性状态信息，同时设置标志位指示本地二级节点内不同于数据所在物理子域的其它子域是否持有该数据副本；

二级节点控制器也可采用本地 / 远端统一目录结构，目录项记录除本物理子域的一级节点控制器外系统其它所有一级节点控制器持有数据副本信息及一致性状态信息；

同理，可按照二级节点控制器目录结构方式配置 n 级 (n>2) 节点控制器目录结构。

[0016] 如图 1 所示，每个节点由若干处理器和节点控制器构成。本地节点内的各个处理器和节点控制器处于同一物理 cache 一致性域，可以直接进行节点内跨处理器访存操作。

[0017] 如图 2 所示，节点控制器可以同与其相连的处理器组成多个节点内物理 cache 一致性域(如图 2 中的节点控制器支持 2 个节点内物理 cache 一致性域)，与图 1 所示系统相比，采用图 2 方式构成相同处理器规模的 CC-NUMA 系统所使用的节点控制器数目减少一半。本图中的各节点控制器处于同一个层次。可以通过多层节点控制器级联，构建大规模系统。

[0018] 如图 3 所示，处理器 P1(301)属于该节点的物理子域 Dom1(307)，P1 处理器(301)访问本地节点物理子域 Dom2 (308) 中处理器 P2 (302) 的内存模块 MEM2 (305) 数据。一种典型处理过程为：

处理器 P1 (301)执行访存操作时在处理器内部未命中 cache 缓存，而将访问指向同一物理 cache 一致性域的节点控制器 (315)，因此处理器 P1 (301) 向节点控制器(315) 的远端内存代理引擎 RPE (313) 的根代理 HP (309) 发送访存请求；

节点控制器(315) 的远端内存代理引擎 RPE (313) 的根代理 HP (309) 接收到处理器 P1 (301) 发送的访存请求，在根代理 HP (309) 内部保存该访存请求信息，包括请求处理器处于哪个物理子域、请求的类型、访存地址等，并维护读写请求在处理过程中的状态。

[0019] 远端内存代理引擎 RPE(313)的根代理 HP(309)查询远端内存目录 RDIR(316)，判断是否需要在本地节点除目标地址所在物理子域之外的各物理子域内的做侦听(Snoop)操作。本例中假设不需要进行物理子域 Dom1 (307)的处理。则远端内存代理引擎 RPE (313) 的根代理 HP (309) 向远端内存代理引擎 RPE (313) 的缓存代理 CP (310) 转发访存请求。

[0020] 节点控制器(315)的远端内存代理引擎 RPE (313)的 cache 缓存代理 CP (310)接收到访存请求，根据访存请求地址可知其属于本节点内另一物理子域 Dom2 (308)中的内存模块，所以向本地内存代理引擎 LPE (314) 的根代理 HP (312) 转发访存请求。

[0021] 节点控制器(315) 的本地内存代理引擎 LPE (314) 的根代理 HP (312) 接收到访存请求，查询本地内存目录 LDIR (317)，判断其它节点是否持有相同地址的数据副本，如果有则需要进行节点间一致性域内的处理。本例中假设没有其它节点持有同一地址的数据副本。本地内存代理引擎 LPE (314) 的根代理 HP (312) 向本地内存代理引擎 LPE (314) 的 cache 缓存代理 CP (311) 转发请求消息。

[0022] 节点控制器(315) 本地内存代理引擎 LPE (314) 的 cache 缓存代理 CP (311) 接收到本地内存代理引擎 LPE (314)的根代理 HP (312)转发的访存请求，在 cache 缓存代理 CP (311) 内部保存该访存请求信息，包括请求的类型、访存地址等，并维护读写请求在处理过程中的状态。

[0023] 本地内存代理引擎 LPE (314) 的 cache 缓存代理 CP (311) 根据访存地址确定其属于处理器 P2 (302) 内存模块 MEM2 (305) 的数据，所以向处理器 P2 (302) 发出请求。

[0024] 处理器 P2 (302) 接收到访存请求，经过处理后，处理器 P2 (302) 向节点控制器(315) 的本地内存代理引擎 LPE (314) 返回数据及其一致性状态。

[0025] 节点控制器(315) 的本地内存代理引擎 LPE (314) 的 cache 缓存代理 CP (311) 接收到返回的数据及其一致性状态，并转发给本地内存代理引擎 LPE (314) 的根代理 HP (312)。

[0026] 本地内存代理引擎(314) 的根代理 HP (312) 接收到返回的数据及其一致性状态，在本地数据目录 LDIR (317) 中记录远端代理引擎(313) 的 cache 缓存代理 CP (310) 持有数据副本及其一致性状态。本地内存代理引擎(314)的根代理 HP (312)向本地节点(315)

的远端内存代理引擎 RPE（313）的 cache 缓存代理 CP（310）返回数据及其一致性状态。

[0027] 节点控制器（315）的远端内存代理引擎 RPE（313）的 cache 缓存代理 CP（310）接收到返回的数据及其一致性状态，向远端内存代理引擎 RPE（313）的根代理 HP（309）转发。

[0028] 节点控制器（315）的远端内存代理引擎 RPE（313）的根代理 HP（309）接收到返回的数据及其一致性状态，在远端内存目录 RDIR（316）中记录对于该请求地址，物理子域 Dom1（307）的处理器 P1（301）持有该地址数据的副本及状态。

[0029] 节点控制器（315）的远端内存代理引擎 RPE（313）的根代理 HP（309）向请求者 Dom1（307）的处理器 P1（301）返回数据及一致性状态，完成该请求的处理。

[0030] 如图 4 所示，处理器 P1（401）属于节点 Node1（433）的物理子域 Dom1（413），P1 处理器（401）访问远端节点 Node2（434）物理子域 Dom2（416）中处理器 P2（405）的内存模块 MEM2（411）数据。该处理过程为：

节点 Node1（433）的处理器 P1（401）执行访存操作时在处理器内部未命中 cache 缓存，根据访存地址，处理器 P1（401）可知其不属于本节点 Node1（433），而属于远端节点 Node2（434）。因此处理器 P1（401）向节点控制器（435）的远端内存代理引擎 RPE（427）的根代理 HP（417）发送访存请求；

节点控制器（435）的远端内存代理引擎 RPE（427）的根代理 HP（417）接收到处理器 P1（401）发送的访存请求，在根代理 HP（417）内部保存该访存请求信息，包括请求处理器处于本地节点 Node1 的哪个物理子域（Dom1 还是 Dom2），请求的类型、访存地址等，并维护缓存读写请求在处理过程中的状态。

[0031] 节点 Node1（433）远端内存代理引擎 RPE（427）的根代理 HP（417）查询远端内存目录 RDIR（425），判断是否需要本地节点各物理子域内做侦听（Snoop）操作。本例中假设不需要进行物理子域 Dom1（307）的处理。则远端内存代理引擎 RPE（427）的根代理 HP（417）根据访存请求地址可知其属于远端节点 Node2（434）的内存模块，所以向远端内存代理引擎 RPE（427）的 cache 缓存代理 CP（419）转发访存请求。

[0032] 节点 Node1（433）的远端内存代理引擎 RPE（427）的 cache 缓存代理 CP（419）接收到访存请求，在 cache 缓存代理 CP（419）内部保存该访存请求信息，包括请求的类型、访存地址等，并维护读写请求在处理过程中的状态。

[0033] 远端内存代理引擎 RPE（427）的 cache 缓存代理 CP（419）根据访存请求地址可知其属于节点 Node2（434）管理的内存模块，故向节点 Node2（434）发出请求；

节点间互连网络将节点 Node1（433）发出的请求消息发送到目的节点 Node2（434）；

节点 Node2（434）的本地内存代理引擎 LPE（432）的根代理 HP（424）接收到该访存请求，在根代理 HP（424）中保存其请求信息，包括请求发自哪个远端节点，请求的类型、访存地址等，并维护读写请求在处理过程中的状态。

[0034] 本地内存代理引擎 LPE（432）的根代理 HP（424）查询本地内存目录 LDIR（430），判断其它节点是否持有相同地址的数据副本，如果有则需要进行节点间一致性域内的处理。本例中假设没有其他节点持有同地址数据副本而无需进行节点间一致性域的处理。

[0035] 本地内存代理引擎 LPE（432）的根代理 HP（424）向本地内存代理引擎 LPE（432）的 cache 缓存代理 CP（422）转发请求；

本地内存代理引擎 LPE（432）的 cache 缓存代理 CP（422）根据访存地址确定其属于节点 Node2（434）的物理子域 Dom2（416）内处理器 P2（405）管理的内存模块 MEM2（411）数据，所以向处理器 P2（405）发出请求。

[0036] 节点 Node2（434）的处理器 P2（405）接收到访存请求，经过处理后，处理器 P2（405）向节点控制器（436）的本地内存代理引擎 LPE（432）返回数据及其一致性状态。

[0037] 节点控制器（436）的本地内存代理引擎 LPE（432）的 cache 缓存代理 CP（422）接收到返回的数据及其一致性状态，并转发给本地内存代理引擎 LPE（432）的根代理 HP（424）。

[0038] 节点控制器（436）的本地内存代理引擎 LPE（432）的根代理 HP（424）接收到返回的数据及其一致性状态，在本地内存目录 LDIR（430）中记录节点 1（433）持有该数据副本及其一致性状态。节点控制器（436）的本地内存代理引擎 LPE（432）的根代理 HP（424）向节点 1（433）的远端内存代理引擎 RPE（427）返回数据及其一致性状态。

[0039] 节点控制器（435）的远端内存代理引擎 RPE（427）的 cache 缓存代理 CP（419）接收到返回的数据及其一致性状态，向远端内存代理引擎 RPE（427）的根代理 HP（417）转发。

[0040] 节点控制器（435）的远端内存代理引擎 RPE（427）的根代理 HP（417）在远端数据目录 RDIR（425）中记录 Dom1 的处理器 P1 持有数据副本及其持有状态，然后向请求者 Dom1 的处理器 P1（401）返回数据及其一致性状态，完成该请求的处理。

[0041] 如图 5 所示，其中包括数据侦听（snoop）消息的处理。图 5 中节点 1（533）的处理器 P2（502）属于节点 Node1（533）的物理子域 Dom2（514），P2 处理器（502）访问同一物理子域 Dom2（514）中处理器 P3（503）的内存模块 MEM3（509）数据。一种典型的处理过程为：

节点 Node1（533）的处理器 P2（502）执行访存操作时在处理器内部未命中 cache 缓存，根据访存地址，处理器 P2（502）可知该地址属于本地节点 Node1（533）的 P3 处理器（503）管理的内存模块 MEM3（509）。由于 P3 处理器（503）与 P2（502）同处于一个物理子域 Dom2（514），所以处理器 P2（502）可以直接访问内存模块 MEM3（509）。处理器 P2（502）向处理器 P3（503）直接发送访存请求；

处理器 P3（503）接收到访存请求，发现节点控制器 1（535）代表本物理子域外其它处理器持有数据副本；根据接收到的访存请求类型，处理器 P3（503）向节点控制器 1（535）的本地内存代理引擎 LPE（528）的 cache 缓存代理 CP（518）发送数据侦听（Snoop）消息。

[0042] 节点控制器 1（535）的本地内存代理引擎 LPE（528）的 cache 缓存代理 CP（518）接收到数据侦听消息，向节点控制器 1（535）的本地内存代理引擎 LPE（528）的根代理 HP（520）转发该消息。

[0043] 节点控制器 1（535）的本地内存代理引擎 LPE（528）的根代理 HP（520）接收到数据侦听消息，查询本地数据目录 LDIR（526）。本例中假设本地数据目录 LDIR（526）显示节点 1（533）的其它物理子域中的处理器（本例中为物理子域 Dom1 中的处理器）和远端节点 2（534）内的处理器持有该数据副本。则本地代理引擎 LPE（528）的根代理 HP（520）向本节点控制器（535）远端代理引擎 RPE（527）的 cache 缓存代理 CP（519）转发数据侦听消息，以处理本地节点 Node1（533）内 Dom1（513）中的数据副本；另外，本地代理引擎 LPE

(528)的根代理 HP (520)向远端节点 Node2 (534)远端代理引擎 RPE (531)的 cache 缓存代理 CP (523)发送数据侦听消息,以处理远端节点 Node2 (534)内的数据副本。

[0044] 节点控制器 1 (535)的远端内存代理引擎 RPE (527)的 cache 缓存代理 CP (519)接收到发送的数据侦听消息,将该消息转发给远端内存代理引擎 RPE (527)的根代理 HP (517);

节点控制器 1 (535)的远端内存代理引擎 RPE (527)的根代理 HP (517)接收到该消息,查询远端数据目录 RDIR (525),以判断本地节点(533)内物理子域 Dom1 (513)中的处理器是否持有数据副本。本例中假设本地节点(533)物理子域 Dom1 (513)中的处理器 P1 (501)持有该数据副本,则节点控制器 1 (535)的远端内存代理引擎 RPE (527)的根代理 HP (517)向物理子域 Dom1 (513)内的处理器 P1 (501)发送数据侦听消息。

[0045] 处理器 P1 (501)接收到数据侦听消息,根据其类型更新数据副本状态,然后向节点控制器 1 (535)内远端数据引擎 RPE (527)的根代理 HP (517)返回应答消息。

[0046] 节点控制器 1 (535)内远端内存代理引擎 RPE (527)的根代理 HP (517)在收集全本地节点内所有应答消息后(在本例中,仅有物理子域 Dom1 (513)内的处理器 P1 (501)持有数据副本),向远端数据引擎 RPE (527)的 cache 缓存代理 CP (519)转发数据侦听消息的应答。

[0047] 节点控制器 1 (535)内远端内存代理引擎 RPE (527)的 cache 缓存代理 CP (519)向节点控制器 1 (535)内本地内存代理引擎 LPE (528)的根代理 HP (520)发送应答消息。

[0048] 节点控制器 1 (535)内本地内存代理引擎 LPE (528)的根代理 HP (520)接收到远端内存代理引擎 RPE (527)的 cache 缓存代理 CP (519)发送的应答消息,则本地节点 Node1 (533)的侦听过程完成。

[0049] 在处理本地节点 Node1 (533)的侦听过程中,节点控制器 2 (536)的远端内存代理引擎 RPE (531)的 cache 缓存代理 CP (523)接收到节点控制器 1 (535)的本地内存代理引擎 LPE (528)的根代理 HP (520)发送的数据侦听消息,将该消息转发给节点控制器 2 (536)的远端内存代理引擎 RPE (531)的根代理 HP (521);

节点控制器 2 (536)的远端内存代理引擎 RPE (531)的根代理 HP (521)接收到数据侦听消息,查询远端数据目录 RDIR (529),以判断本地节点(534)内各个物理子域中的处理器是否持有数据副本。本例中假设节点 2 (534)物理子域 Dom1 (515)中的处理器 P1 (504)持有该数据副本,则节点控制器 2 (536)的远端内存代理引擎 RPE (531)的根代理 HP (521)向物理子域 Dom1 (515)内的处理器 P1 (504)发送数据侦听消息。

[0050] 处理器 P1 (504)接收到数据侦听消息,根据其类型更新内部数据副本状态,然后向节点控制器 2 (536)内远端数据引擎 RPE (531)的根代理 HP (521)返回应答消息。

[0051] 节点控制器 2 (536)内远端内存代理引擎 RPE (531)的根代理 HP (521)在收集全本地节点内所有应答消息后,向远端数据引擎 RPE (531)的 cache 缓存代理 CP (523)转发应答消息。

[0052] 节点控制器 2 (536)内远端内存代理引擎 RPE (531)的 cache 缓存代理 CP (523)向节点控制器 1 (535)内本地内存代理引擎 LPE (528)的根代理 HP (520)发送应答消息。

[0053] 节点控制器 1 (535)本地内存代理引擎 LPE (528)的根代理 HP (520)接收到节点控制器 2 (536)远端内存代理引擎 RPE (531)的 cache 缓存代理 CP (523)发送的应答

消息。

[0054] 节点控制器 1（535）本地内存代理引擎 LPE（528）的根代理 HP（520）收集完所有应答消息，然后更新本地数据目录 LDIR（526），并向本地内存代理引擎 LPE（528）的 cache 缓存代理 CP（518）转发应答消息。

[0055] 节点控制器 1（535）本地内存代理引擎 LPE（528）的 cache 缓存代理 CP（518）向物理子域 Dom2（514）内的处理器 P3（503）发送应答消息。

[0056] 处理器 P3（503）收到应答后，更新其内部目录信息，并向请求者处理器 P2（502）发送数据。

[0057] 目录格式设计如图 6 所示：

远端内存代理引擎 RPE 中远端内存目录 RDIR 记录的信息按访存地址分为两部分：

（1）远端地址在本地节点各物理子域的数据共享者信息：记录本地节点内各个物理子域中各处理器是否持有数据副本及其一致性状态；

（2）本地地址在本地节点各物理子域的数据共享者信息：假设本地节点由 Dom1~Dom n 共计 n 个物理子域组成，则对于物理子域 Dom i（i = 1,2,..,n）中处理器管理的内存数据，RDIR 记录本地节点内物理子域 Dom j（j = 1,2,…,n 且 j!=i）内的处理器是否持有该数据副本及其一致性状态。

[0058] 本地内存代理引擎 LPE 中本地内存目录 LDIR（如图 6(b) 所示）记录的信息表示本地节点内某一物理子域的内存数据是否被远端节点或本节点其它物理子域缓存。

[0059] 本地、远端目录也可整合为统一的目录结构（UDIR），如图 6（c）所示。UDIR 目录项记录除本物理子域的处理器外系统其它所有处理器持有数据副本信息及一致性状态信息。

[0060] 如图 7 所示，节点控制器（712）可以支持多个物理子域。节点控制器（712）的各处理器接口属于哪个物理子域可以进行配置。对于每个处理器接口，通过为接口配置物理子域标识寄存器（709,710,711）来指明该处理器接口所属的物理子域。节点控制器的处理器接口物理子域标识寄存器（709,710,711）既可以在系统启动时配置，也可以在系统运行过程中配置。

[0061] 除说明书所述的技术特征外，均为本专业技术人员的已知技术。

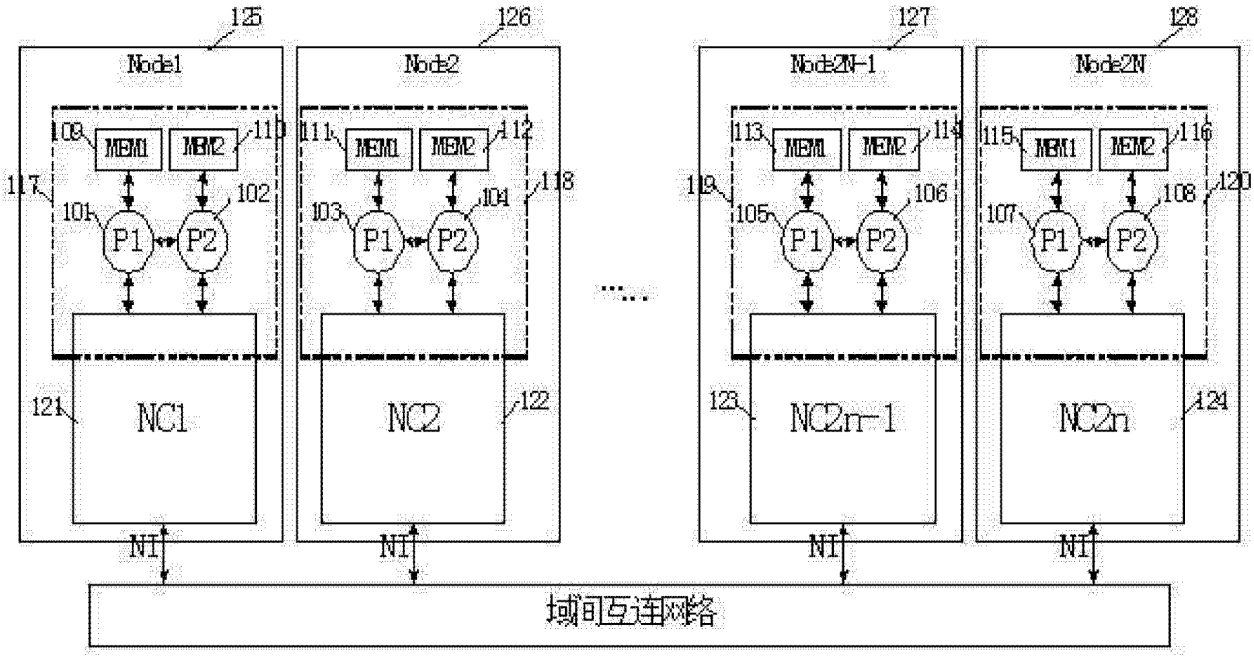


图 1

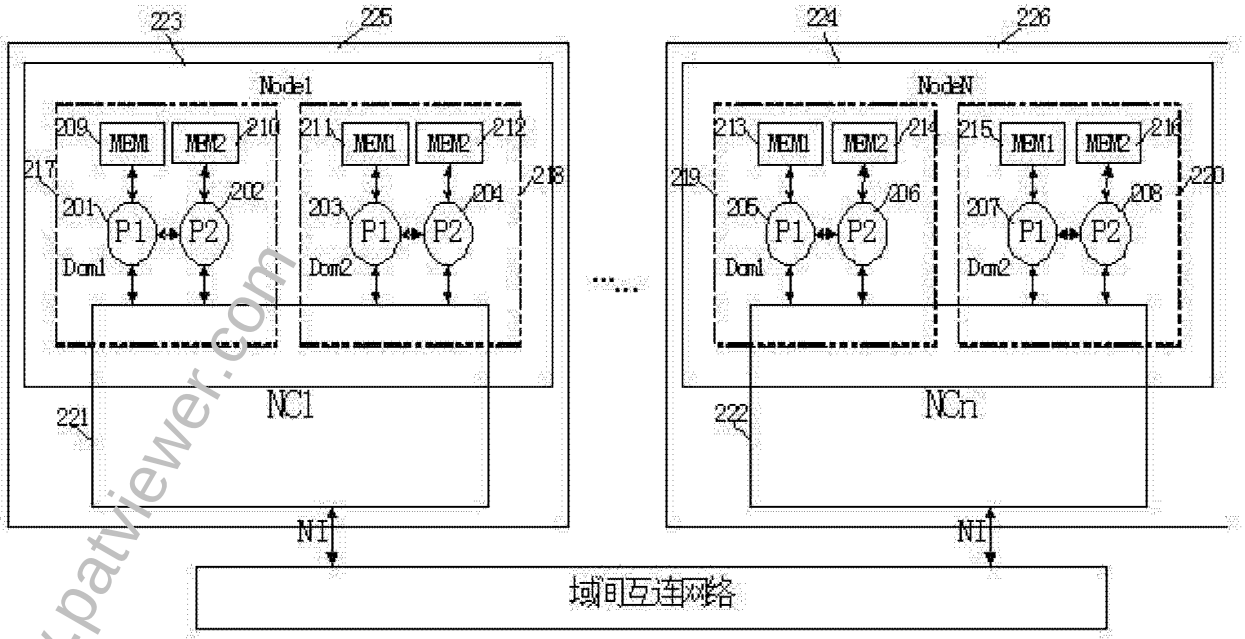


图 2

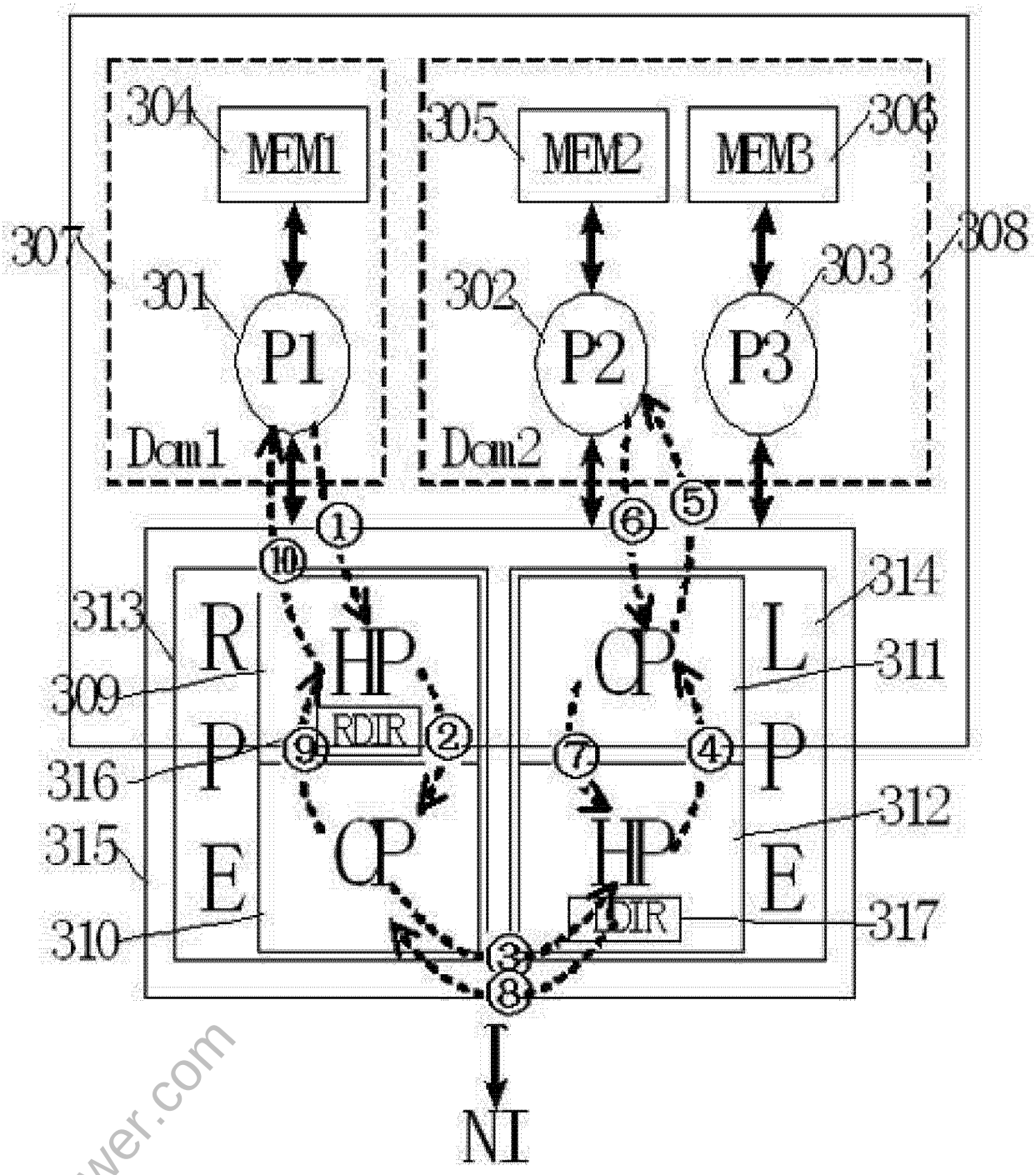


图 3

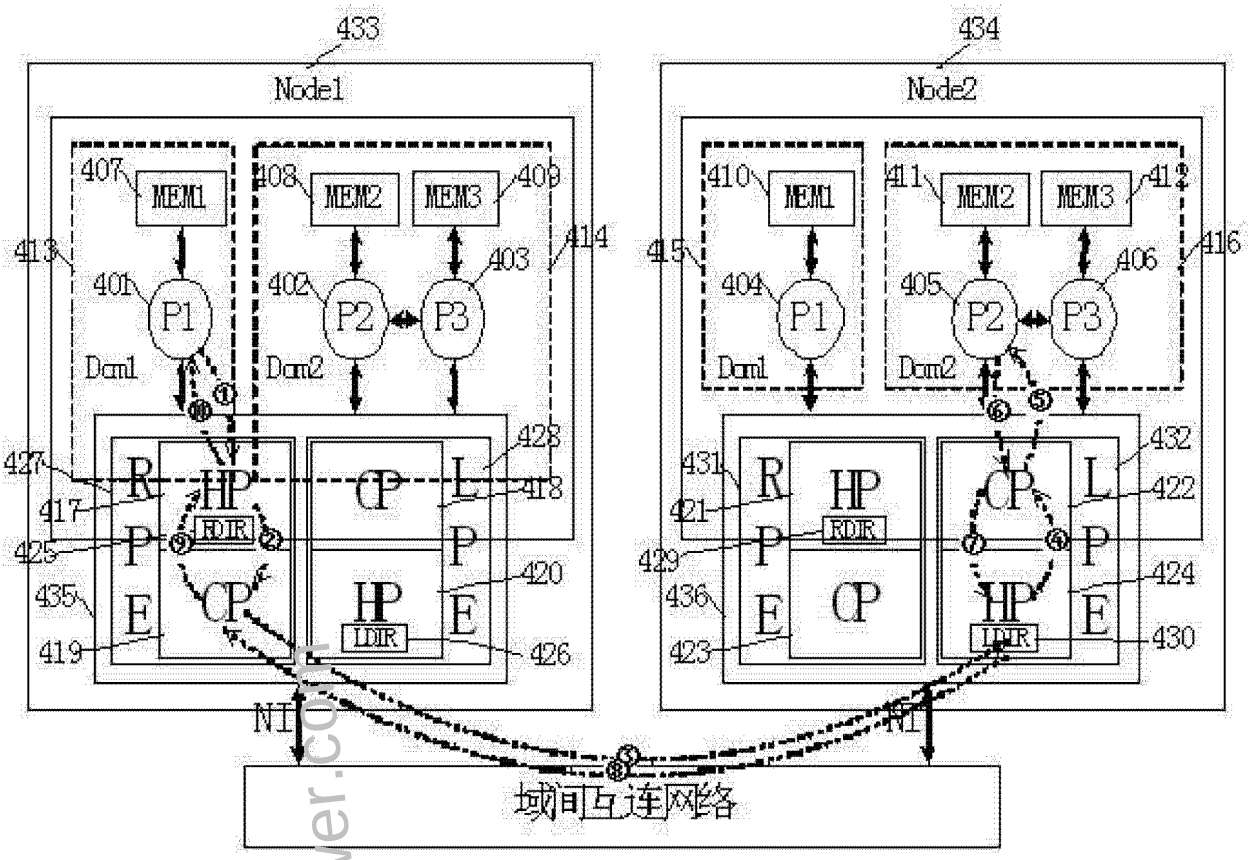


图 4

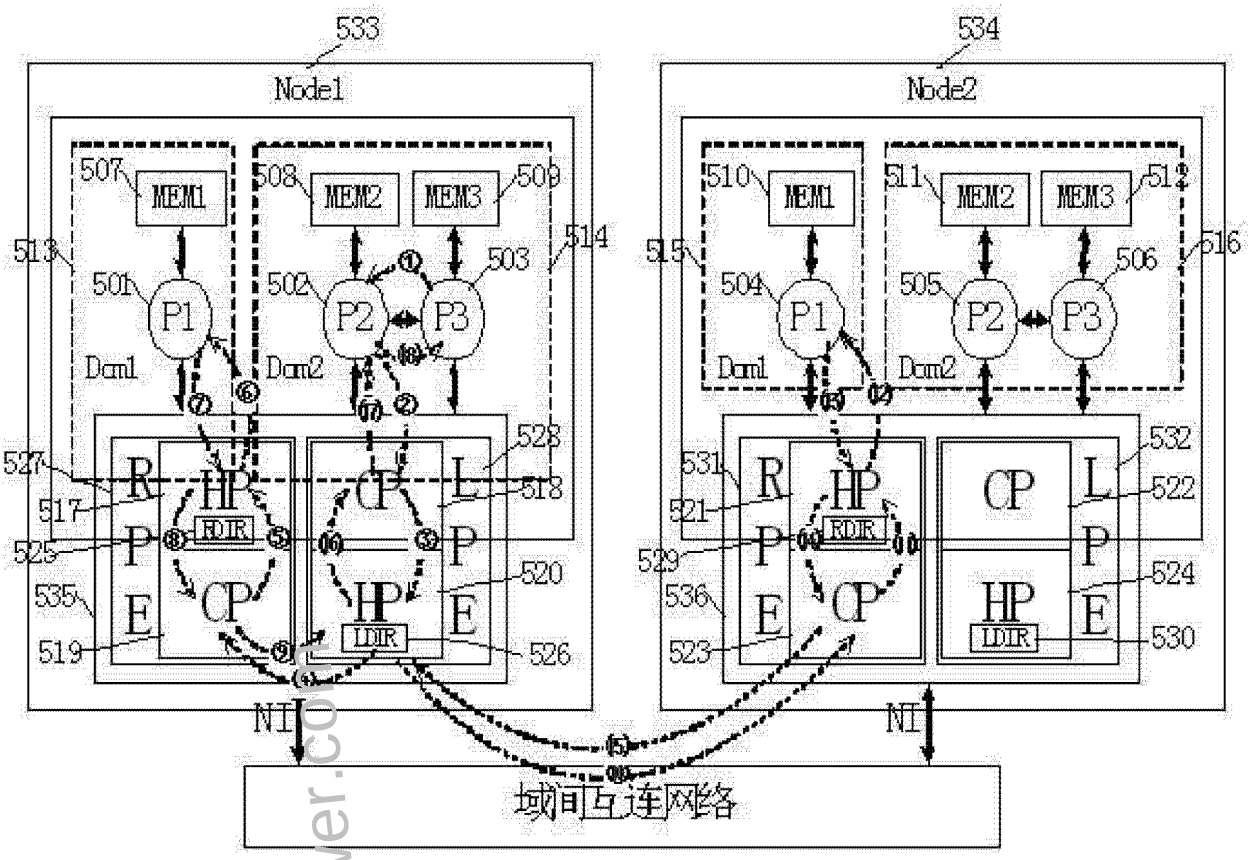


图 5

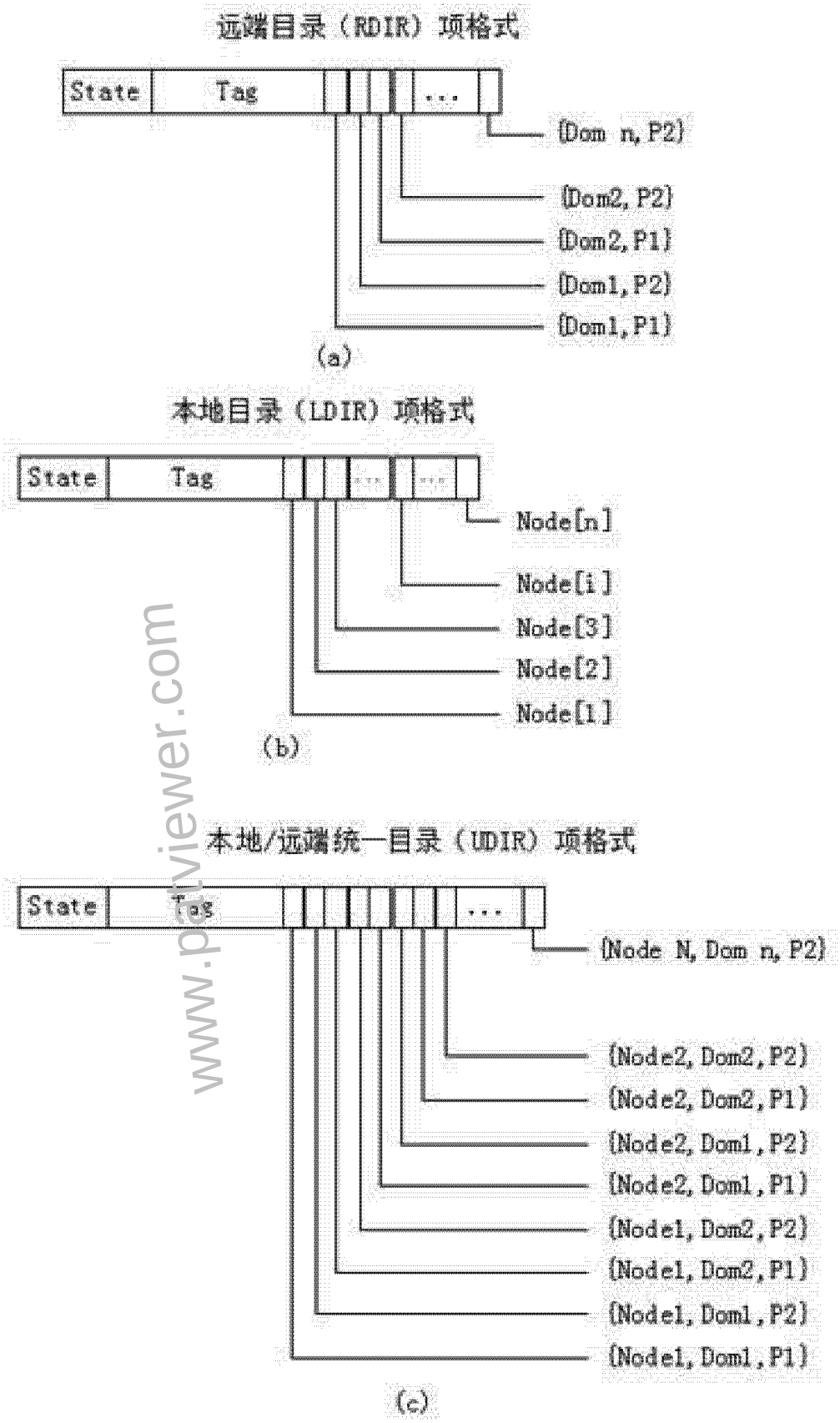


图 6

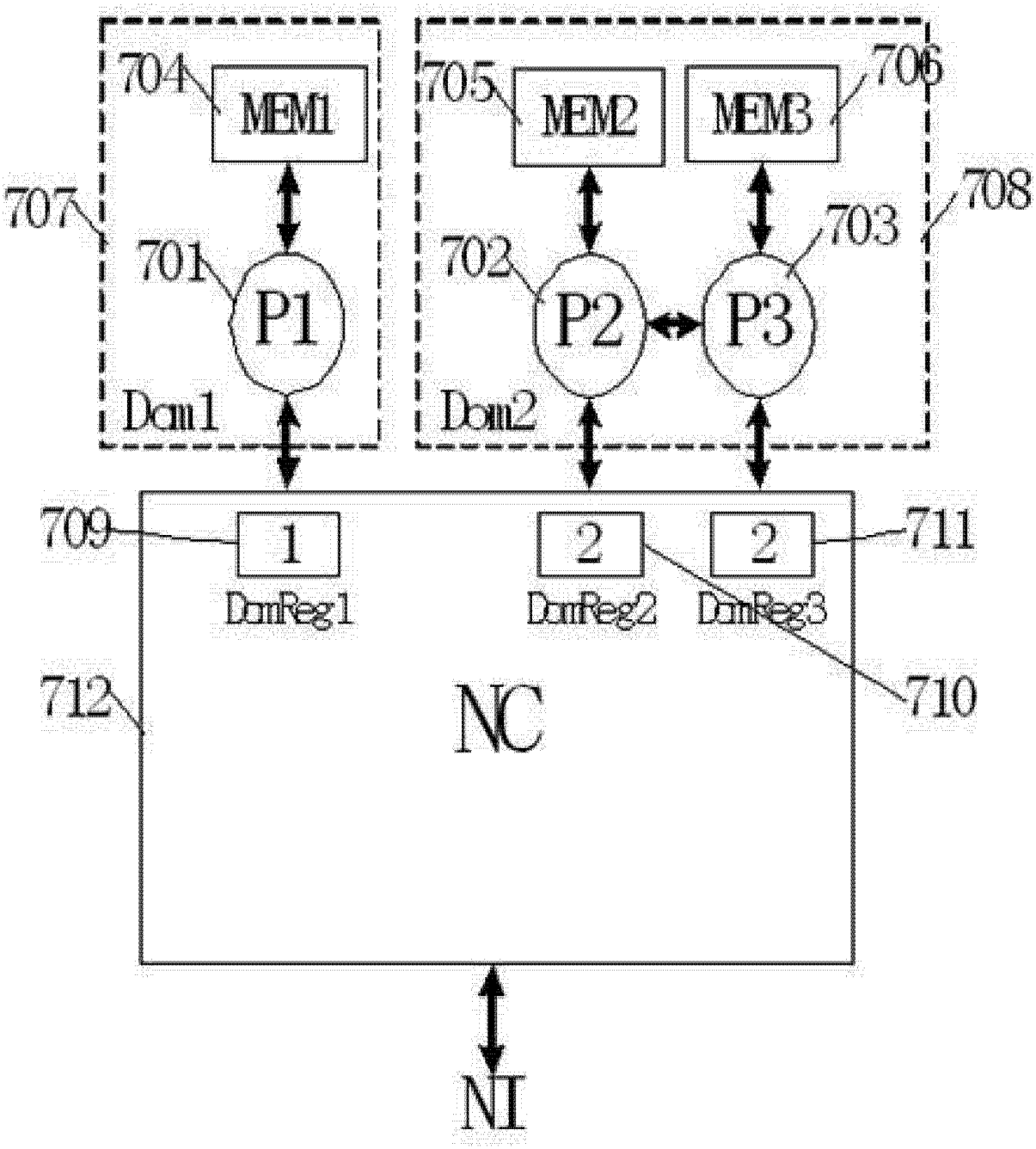


图 7

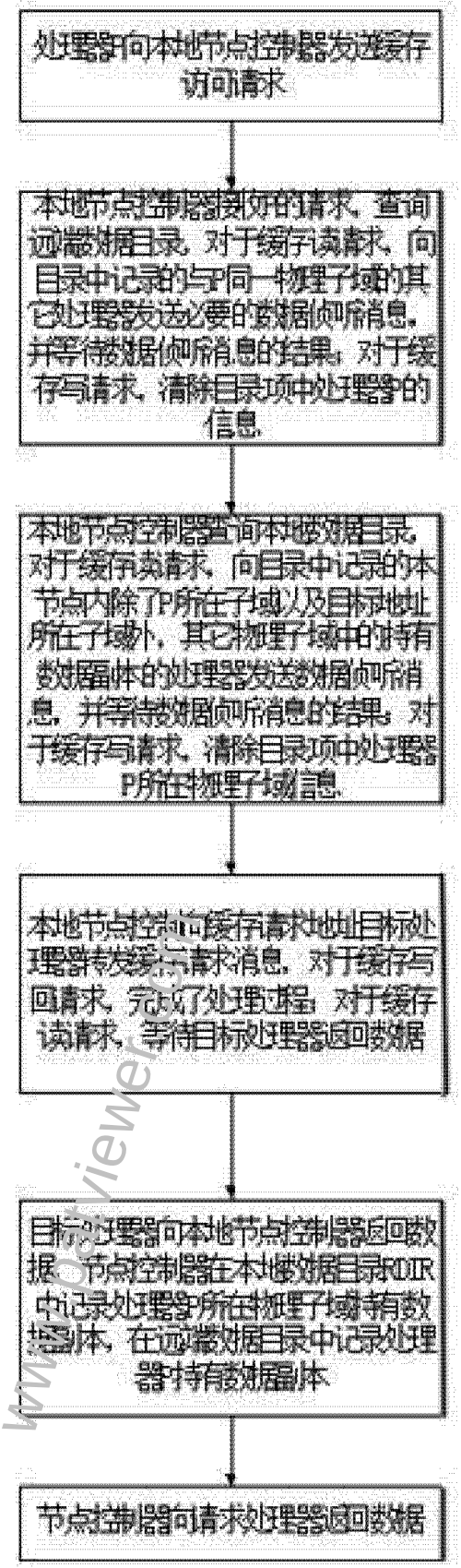


图 8

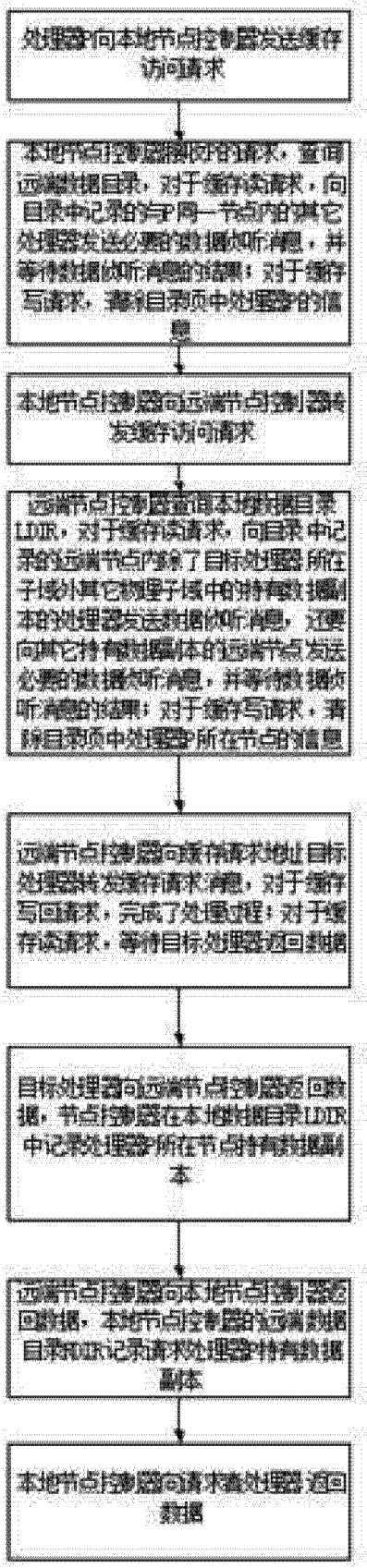


图 9