[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 12/00 (2006.01)



[12] 发明专利说明书

专利号 ZL 02805431.8

[45] 授权公告日 2006 年 11 月 22 日

[11] 授权公告号 CN 1286016C

[22] 申请日 2002.2.25 [21] 申请号 02805431.8

[30] 优先权

[32] 2001. 2.24 [33] US [31] 60/271,124

[86] 国际申请 PCT/US2002/005587 2002.2.25

[87] 国际公布 WO2002/069238 英 2002.9.6

[85] 进入国家阶段日期 2003.8.22

[71] 专利权人 国际商业机器公司

地址 美国纽约州

[72] 发明人 马赛厄斯 A·布卢姆里奇 陈 东
保罗 W·科特尤斯 艾伦 G·加拉
马克 E·贾姆帕帕
菲利普·海德伯格 德克·霍尼克
马丁·奥马彻特

审查员 乔凌云

[74] 专利代理机构 中国专利代理(香港)有限公司

代理人 王 岳 陈 霁

权利要求书 2 页 说明书 10 页 附图 1 页

[54] 发明名称

通过放/取窗口管理一致性

[57] 摘要

一种用于管理多处理器计算机系统的两个处理器节点的两个处理器之间的一致性的方法和装置。总体来说，本发明涉及一种简化并明显加快在消息传递并行计算机中的高速缓存一致性的管理的软件算法，还涉及辅助此高速缓存一致性算法的硬件装置。 该软件算法使用放/取窗口的打开和关闭来协调获得高速缓存一致性所需的行为。 硬件装置可以是硬件地址解码的扩展，它在节点的物理存储器地址空间内创建一个(a)并不实际存在的、并(b)因此能够立即响应来自处理单元的读和写请求的虚拟存储器区。

1.一种简化并加快在消息传递并行超级计算机中的高速缓存一致性管理的方法，该超级计算机包含两个或更多个非一致的处理器单元，这里的一个处理器单元主要执行计算，而另一个处理器单元执行消息传递行为，该方法包含步骤：

打开和关闭放/取窗口；

执行用于实现高速缓存一致性的行为；以及

使用放/取窗口的所述打开和关闭来协调用于获得高速缓存一致性的行为。

2.根据权利要求 1 所述的方法，其中所述使用的步骤包含确保被发送的数据不在任一处理器的高速缓存中，而且被接收的数据也不在任一处理器的高速缓存中的步骤。

3.根据权利要求 2 所述的方法，其中确保的步骤包含通过发出软件命令来将数据载入高速缓存的步骤。

4.根据权利要求 1 所述的方法，进一步包括一个辅助实现所述高速缓存一致性的步骤，所述步骤包括：

使用一个存储器控制器，以便在节点的物理存储器地址空间内创建一个（a）并不实际存在的、并（b）因此能够立即响应来自处理单元的读和写请求的虚拟存储器区。

5.根据权利要求 4 所述的方法，其中存储器控制器允许将处理器绝不会使用的垃圾数据拉进处理器的高速缓存，从而仅驱逐修改过的数据，并替换具有最佳性能的未修改的数据。

6. 根据权利要求 5 所述的方法，其中，实际上不需要从存储器中取出垃圾数据，而只需以存储器控制器对处理器单元的立即回复的方式来实现将垃圾数据拉进处理器的高速缓存的步骤。

7.根据权利要求 6 所述的方法，其中只有实际修改的数据被从高速缓存写到存储器，而未修改的数据被简单地立即抛弃。

8.根据权利要求 7 所述的方法，其中，当放/取窗口的总体大小超过处理器的高速缓存的大小时，以这种方式清理高速缓存为用于确保在高速缓存中没有遗留来自通信区的数据所需的工作总量提供了一个上限。

9.一种简化并加快在消息传递并行超级计算机中的高速缓存一致性管理的系统，其中该超级计算机包含两个或更多个非一致的处理器

单元，这里的一个处理器单元主要执行计算，而另一个处理器单元执行消息传递行为，该系统包含：

用于打开和关闭放/取窗口的装置；

用于执行获得高速缓存一致性的行为的装置；以及

用于使用放/取窗口的所述打开和关闭来协调实现高速缓存一致性的行为的装置。

10.根据权利要求9所述的系统，其中所述使用的装置包含用于确保被发送的数据不在任一处理器的高速缓存中，而且被接收的数据也不在任一处理器的高速缓存中的装置。

11.根据权利要求10所述的系统，其中确保的装置包含用于通过发出软件命令来将数据载入高速缓存的装置.

12.根据权利要求9所述的系统，进一步包括用于辅助实现高速缓存一致性的装置，该装置包含：

一个用于在节点的物理存储器地址空间内创建一个（a）并不实际存在的、并（b）因此能够立即响应来自处理单元的读和写请求的虚拟存储器区的存储器控制器。

13.根据权利要求12所述的系统，其中存储器控制器允许将处理器绝不会使用的垃圾数据拉进处理器的高速缓存，从而仅驱逐修改过的数据，并替换具有最佳性能的未修改的数据.

14. 根据权利要求13所述的系统，其中，实际上不需要从存储器中取出垃圾数据，而存储器控制器只需以对处理器单元的立即回复的方式来实现将垃圾数据拉进处理器的高速缓存。

15.根据权利要求14所述的系统，其中只有实际修改的数据被从高速缓存写到存储器，而未修改的数据被简单地立即抛弃。

16.根据权利要求15所述的系统，其中，当放/取窗口的总体大小超过处理器的高速缓存的大小时，以这种方式清理高速缓存提供了为确保在高速缓存中没有遗留来自通信区的数据所需的工作总量的一个上限。

通过放/取窗口管理一致性

对相关申请的交叉引用

本发明请求于2001年2月24日提交的题为“MASSIVELY PARALLEL SUPERCOMPUTER”的共同拥有、共同未决的美国临时专利申请No.60/271,124的利益,通过引用的方式将该申请的全部内容和公开直接并入到这里，就好像是在这里完整给出的一样。本专利申请还涉及下面与之同日提交的共同拥有、共同未决的美国专利申请，通过引用的方式将这些申请的每一个的全部内容和公开直接并入到这里，就好像是在这里完整给出的一样：美国专利申请No.(YOR920020027US1,YOR920020044US1(15270))，“Class Networking Routing”;美国专利申请No.(YOR920020028US1(15271))，“A Global Tree Network for Computing Structures”；美国专利申请No.(YOR920020029US1(15272))，“Global Interrupt and Barrier Networks”；美国专利申请No.(YOR920020030US1(15273))，“Optimized Scalable Network Switch”；美国专利申请No.(YOR920020031US1, YOR920020032US1(15258))，“Arithmetic Functions in Torus and Tree Networks”；美国专利申请No.(YOR920020033US1,YOR920020034US1(15259))，“Data Capture Technique for High Speed Signaling”；美国专利申请No.(YOR920020035US1(15260))，“Managing Coherence Via Put/Get Windows”；美国专利申请No.(YOR920020036US1,YOR920020037US1(15261))，“Low Latency Memory Access And Synchronization”；美国专利申请No.(YOR920020038US1(15276))，“Twin-Tailed Fail-Over for Fileservers Maintaining Full Performance in the Presence of Failure”；美国专利申请No.(YOR920020039US1(15277))，“Fault Isolation Through No-Overhead Link Level Checksums”；美国专利申请No.(YOR920020040US1(15278))，“Ethernet Addressing Via Physical Location for Massively Parallel Systems”；美国专利申请No.(YOR920020041US1(15274))，“Fault Tolerance in

a Supercomputer Through Dynamic Repartitioning”；美国专利申请No.(YOR920020042US1(15279))，“checkpointing Filesystem”；美国专利申请No.(YOR920020043US1(15262))，“Efficient Implementation of Multidimensional Fast Fourier Transform on a Distributed-Memory Parallel Multi-Node Computer”；美国专利申请No.(YOR9-20010211US2 (15275))，“A Novel Massively Parallel Supercomputer”；以及美国专利申请No.(YOR920020045US1(15263))，“Smart Fan Modules and System”。

技术领域

本发明涉及分布式存储器消息传递并行计算机设计和系统软件的领域，例如应用于生命科学领域中的计算。

背景技术

在题为“A Novel Massively Parallel Supercomputer”的临时专利申请中，描述了一种三维环形式的大规模并行超级计算机的体系结构，该体系结构被设计为大范围应用的、在万亿OPS（每秒万亿个操作）的数量级上传递处理功率。该体系结构包含65,536个处理节点，这些处理节点被组织成一个64×32×32的三维环，其中每个处理节点连接到六个（6）相邻的节点。

超级计算机体系结构的每个处理节点都是包含两个电子处理器（此外还有其它的部件）的半导体设备。这些处理器中的一个被指定为“计算处理器”，并且该处理器在通常进行的操作中专用于应用计算。另一个处理器是“I/O处理器”，该处理器在通常进行的操作中是专用执行支持消息传递通信的行为的服务处理器。这些处理器中的每一个包含一个分离的一级高速缓存（L1），该一级高速缓存（L1）可包含在两个处理器访问的公用存储器中存储的数据的拷贝。如果一个处理器改变它的存储器位置的L1拷贝，且另一个处理器具有相同位置的拷贝，那么如果使这两份拷贝相同，则它们变得“一致”。

消息传递是计算机通信的一种众所周知的形式，其中处理器从它们自己的存储器向另一个节点的存储器清楚地拷贝数据。在上述临时专利申请No.60/271,124中公开的双处理器节点中，I/O处理器主要用

于方便在节点的公用存储器与其它节点的公用存储器之间的消息传递。因此，它既产生由计算处理器消耗的数据（当消息被接收时），又消耗由计算处理器产生的数据（以便发送消息）。结果，对于这两个处理器来说，通常在它们的L1中包含相同存储器位置的拷贝。如果传递的消息很小，并且数量很多，这个问题就会变得更严重。因此，很明显地需要一种方法来使每个处理器的L1一致，同时不需要扩展电路，并对性能的影响最小。

由于大规模并行计算机被扩张到几千个处理节点，所以典型的应用消息传输业务包含更多的消息，其中每一个这样的消息包含由计算机中的其它节点传送的信息。通常，一个节点将本地产生的消息分散到一些其它节点，同时将一些远程产生的消息接收到其本地存储器中。这些大规模计算机的总体性能通常由该系统的消息传递性能限制。

对于这种数据传输，一种在文献（见例如 http: //www.mpi-forum.org/docs/docs.html，MPI-2 下）中描述的公用的消息传递接口称为“单向通信”。单向通信使用一个“放/取”（“put/get”）消息传递的模式，在该模式中，消息携带源（用于取）或目的（用于放）存储器地址。在操作一个公共问题的并行超级计算机中，典型地，放和取是成批集合并一起发出的。这使独立运行的处理器保持大致同步，同时使性能最优化。放和取发生的时间段称为放/取窗口。此窗口在时间（它的出现时候）和存储器（在放或取消息携带的存储器地址的范围上）方面进行扩展。图 2 显示了包含许多不同消息的放/取窗口 30。

放/取窗口将一致性的概念扩展到大规模并行超级计算机的不同处理节点上的处理器。放/取窗口的实施必须确保在窗口打开期间被放到窗口的所有消息在窗口关闭之前被接收到窗口的存储器中。相似的，在窗口的存储器上的取也只有在窗口打开的期间内允许。因此，放/取窗口只不过单纯地是一种用于使节点与操作它的存储器的远程处理器同步的机构。

放/取窗口的管理目前是通过缓冲放/取消息或者通过使用清楚的同步消息来完成的。对消息的缓冲消耗存储器，而所述存储器总是有限的。每个窗口的清楚的同步受访问窗口的所有节点之间往返消息的

较长延迟的影响。因此，在例如共同未决的专利申请No.10/468993（代理卷号15275）中描述的大规模机器上，这些方法不很适合，因为缓冲存储器有限，还因为访问任一特定窗口的节点数目通常随计算机中的处理节点的数目扩张。

在计算机设计领域内的长期存在的问题是如何保持这些L1高速缓存一致。典型的解决方案使用了称为“侦听”其它处理器的存储器总线的技术，这种技术可能速度很慢，并可能降低每个处理器的性能。作为替换，包含公用存储器中的数据的L1旧拷贝的处理器可请求一个新的拷贝，或者将旧拷贝标记为作废，但这需要知道该拷贝何时变为无效。有时，该知识是不完整的，这就必须执行不必要的存储操作，从而进一步降低了性能。其它计算机利用“互锁”，由此，一个处理器获准使用特定数据，而其它处理器不能使用，但是，这种准许涉及两个处理器之间的相互作用，通常需要在半导体设备中的额外复杂电路，从而降低了两个处理器的性能。

通常实际应用中的其它解决方案使希望共享的存储器区的所有高速缓存失效。这一事实使对这些区域的所有存储访问恶化，而不仅仅是对共享数据的存储访问。

发明内容

本发明的目的是提供一种用于管理并行处理计算机系统中的一致性的改进的方法。

本发明的另一个目的是在没有扩展电路并且对每个处理器影响最小的情况下获得多处理器节点的处理器的一级高速缓存之间的一致性。

本发明的再一个目的是提供一种方法和装置，所述的方法和装置与软件算法结合使用，以便在处理器或直接存储器存取（DMA）设备之间完成有效的高速消息传递通信，这可在不明显降低性能的条件下维持一致性。

使用本发明的方法和装置可获得这些目的以及其它的目的。根据第一方面，本发明提供一种简化并明显加快在消息传递大规模并行超级计算机（例如在共同未决的专利申请 No.10/468993（代理案号15275）中描述的超级计算机）中的高速缓存一致性的管理的软件算

法，其中该超级计算机包含两个或更多个非一致的处理单元（或者甚至 DMA 控制器），这里的一个处理单元主要执行计算，而另一个单元执行消息传递。在这种大规模并行超级计算机中，算法通常以一系列步骤进行，其中每个步骤包括一个计算阶段，随后是一个通讯阶段。在通信阶段中，节点交换由计算阶段产生的数据和该算法的下一步所需的数据。由于算法的性质，这些阶段通常是紧密同步的，以便使通信在一瞬间发生在整个机器上。因此，在每个通信阶段的开始和结束时，可以将管理放/取窗口的同步的成本分摊到大量节点上。简要地说，可以在通信阶段的开始时使用一个全局操作来打开很多放/取窗口，并且可以在通信阶段的结束时使用一个第二全局操作来关闭这些窗口。

因为 I/O 处理器在高速缓存一致性被保证之后才能对消息进行实际的发送或接收，所以本发明提供了一种机构以确保：被“放”（发送）的数据不在任一处理器的高速缓存中；并且被“取”（接收）的数据也不在任一处理器的高速缓存中。通过在打开和关闭“放/取窗口”时协调这些行为，本发明降低了获得一致性并允许将那个工作分摊到大量个体消息上所需的工作总量。同样，由于在节点中的两个处理单元必须执行此工作，所以本发明使这能够同时发生。而且，当需要的时候，可以通过使用共同未决的专利申请No.10/468993（代理卷号15275）中描述的全局屏障网络来在大规模并行机器中的大量独立节点上协调这些行为。

根据第二方面，本发明提供一种硬件装置，该硬件装置辅助上述的高速缓存一致性软件算法，并限制在放/取窗口上获得高速缓存一致性所需的总时间（或等待时间）。此装置是硬件地址解码器的一个简单的扩展，它在节点的物理存储器地址空间内创建一个（a）并不实际存在的、并（b）因此能够立即响应来自处理单元的读和写请求的存储器区。这进一步地加快了一致性行为，因为它允许将垃圾数据（处理器绝不会使用的数据）拉进处理器的高速缓存，从而仅驱逐修改过的数据，并以最佳性能替换未修改的数据。性能变得更快，因为此垃圾数据实际上不需被从存储器中取出，相反，存储器控制器只需要立即地进行答复。

性能变得更快还因为只有修改的数据被从高速缓存写到存储器，

而干净的数据被简单地立即抛弃。而且，对于“放/取窗口”的总体大小超过（可能是大大超过）处理器高速缓存的大小的情况，以这种方式清理高速缓存为用于确保在高速缓存中没有遗留来自通信区的数据所需的工作总量提供了一个上限。可以注意到：独立于上述的软件算法，对于通常的、使用最近最少使用的高速缓存替换策略的计算机系统来说，这种硬件设备是有用的。

此外，有两种特殊软件指令可被用在本发明的优选实施中。一种称为“数据高速缓存块清除和无效”的指令可被用于将数据从第一处理器的存储器区写进共享的存储器区，并同时防止第一处理器使用被写入其存储器区的数据。称为“数据高速缓存块零”的第二软件指令可被用于将数据从第一处理器的存储器区写进共享的存储器。通过使用这些软件指令或类似的软件指令，本发明的方法和装置在与软件算法结合使用时可以获得节点之间的高速消息传递通信，同时在不明显降低性能的条件下维持一致性。

通过考虑下面参照指定和显示本发明的优选实施例的附图而给出的具体描述，可以使本发明的进一步益处和优点变得明显。

附图说明

图 1 显示了实现本发明的两个处理器节点。

图 2 图示了可在本发明的实际应用中使用的放/取窗口。

具体实施方式

本发明涉及用于管理多处理器计算机系统的一致性的方法和装置。图 1 示出可实现本发明的一个节点 10，节点 10 的处理器 12 和 14 中的每一个都具有相应的高速缓存存储器区 16 和 20，并且两个处理器共享一个第三存储器区 22。总体来说，本发明涉及一种简化并明显加快在消息传递并行计算机中的高速缓存存储器一致性管理的软件算法，还涉及辅助此高速缓存一致性算法的硬件装置。该软件算法使用放/取窗口的打开和关闭来协调获得高速缓存一致性所需的行为。硬件装置可以是硬件地址解码的扩展，它在节点的物理存储器地址空间内创建一个（a）并不实际存在的、并（b）因此能够立即响应来自处理单元的读和写请求的物理存储器区。

如上面指出的，本发明利用一个称为“放/取”数据传输的原理。由于并行的多个计算机被扩张到更多的节点，所以典型的应用消息传输包含更多的消息，其中每一个这样的消息包含由多个计算机中的其它节点执行的一项工作。通常，一个节点将本地产生的工作项目分散到许多其它节点（一次“放”），同时将许多远程产生的工作项目集合到其本地存储器中（一次“取”）。所述多个计算机的总体性能通常受该系统的消息传递性能所限。

对于这种数据传输，一种在文献（见例如 http: //www.mpi-forum.org/docs/docs.html，MPI-2 下）中描述的特殊有效的消息传递接口称为单向通信。单向通信使用一个“放/取”消息传递的模式，在该模式中，消息携带源（用于“取”）或目的（用于“放”）的存储器地址。在操作一个公共问题的并行超级计算机中，典型地，放和取是成批集合并同时发出的。这使独立运行的处理器保持大致同步，同时允许对于公共问题的良好性能。放和取发生的时间段由放/取窗口限定。此窗口在时间（它什么时候出现）和存储器（放或取中的数据存在于存储器地址的哪个范围）的方面进行扩展。图 2 显示了包含许多不同消息的放/取窗口 30。

在这种大规模并行超级计算机中，算法通常以一系列步骤进行，其中每个步骤由一个计算阶段随后接着一个通讯阶段组成。在通信阶段中，节点交换由计算阶段产生的数据和该算法的下一步所需的数据。由于算法的性质，这些阶段通常是紧密同步的，以便使通信在一瞬间发生在整个机器上。因此，在每个通信阶段的开始和结束时，可以将管理放/取窗口的同步的成本分摊到大量节点上。简要地说，可以在通信的开始时使用一个全局操作来打开很多放/取窗口。

本发明利用这个放/取窗口提供一个的简单方法以便管理存储器的一致性。根据第一方面，提供了一种简化并明显加快在消息传递大规模并行超级计算机中的高速缓存一致性的管理的软件算法（例如在共同未决专利申请 No.10/468993（代理卷号 15275）中描述的算法），其中该超级计算机包含两个或更多个非一致的处理单元(或者甚至 DMA 控制器），这里的一个处理单元主要执行计算，而另一个单元执行消息传递行为。简要地说，这种算法使用“放/取窗口”的打开和关闭来协调获得存储器一致性所需的行为。

因为在高速缓存一致性被保证之后才能对消息进行实际的发送或接收，所以本发明提供了一种机构以确保：被“放”（发送）的数据不在任一处理器的高速缓存中；并且被“取”（接收）的数据也不在任一处理器的高速缓存中。通过在打开和关闭“放/取窗口”时协调这些行为，本发明降低了获得一致性并允许将那个工作分摊到大量个体消息上所需的工作总量。同样，由于在节点中的两个处理单元必须执行此工作，所以本发明使这能够同时发生。而且，当需要的时候，可以通过使用共同未决的专利申请 No.10/468993（代理卷号 15275）中描述的全局屏障网络来在大规模并行机器中的大量独立节点上协调这些行为。

此算法由下面描述的硬件辅助，但通常来说，即使在没有该装置的时候该算法也对消息传递计算机有益。在没有该装置时，通过为保留物理区的每个高速缓存线发出一个 DCBT（数据高速缓存块接触）指令来从此物理区向高速缓存进行载入，就可以使用一个大小等于处理器的高速缓存的物理存储器的特殊保留区，但性能会有所降低。

根据本发明的第二方面，提供一种新颖的硬件装置，该装置辅助上述的高速缓存一致性算法，并限制在放/取窗口上获得高速缓存一致性所需的总时间（或等待时间）。此装置是硬件地址解码器的一个简单的扩展，它在节点的物理存储器地址空间内创建一个（a）并不实际存在的、并（b）因此能够立即响应来自处理单元的读和写请求的虚拟存储器区。这进一步地加快了一致性行为，因为它允许将垃圾数据（处理器绝不会使用的数据）拉进处理器的高速缓存，从而仅驱逐修改过的数据，并以最佳性能替换未修改的数据。性能变得更快，因为此垃圾数据实际上不需被从存储器中取出，相反，存储器控制器只需要立即地进行答复。

性能变得更快还因为只有实际修改的数据被从高速缓存写到存储器，而干净的数据被简单地立即抛弃。而且，对于“放/取窗口”的总体大小超过（可能是大大超过）处理器高速缓存的大小的情况，以这种方式清理高速缓存为用于确保在高速缓存中没有遗留来自通信区的数据所需的工作总量提供了一个上限。例如，假设一个完全关联的高速缓存，如果通信区是 16 兆字节（通常事件），传统的高速缓存清除技术可能需要（等于每高速缓存线 16MB/32MB）524,288DCBF 指令，

而这里描述的算法可能需要最多 1,024DCBF 指令——如果处理器的高速缓存的大小为 32 千字节，并具有 32 字节高速缓存线。可以注意到：独立于上述的软件算法，对于通常的、使用最近最少使用的高速缓存替换策略的计算机系统来说，这种硬件设备是有用的。

下面描述两个具体的软件实施例。如果被接收的消息的大小小于 L1 的大小，则第一实施例可能是优选的；如果被接收的消息的大小大于 L1 的大小，则第二实施例可能是优选的。

第一实施例：如果被接收的消息的大小小于 L1 的大小。

在这种情况中，本发明使用术语上称为“数据高速缓存块清除和无效”（DCBF）的软件指令，由此，如果存储器的连续范围已经在 L1 中被修改，则将其从 L1 写回到公共存储器。DCBF 是一个 PowerPC BookE 指令；对于其它处理器存在相似的指令。同时，高速缓存中的数据被标记为无效，并且在没有重载公用存储器内容的情况下不能使用。对于地址窗口中的每个线发出一个 DCBF。

更具体地说，当为放或取而打开窗口时，软件（在通信库中）指示接收处理器（在我们的双处理器节点中的计算处理器）清除地址窗口中的 L1 的内容，就如上面所述的。这个简单的操作确保公用存储器中的数据与计算处理器的高速缓存中的数据相同，并且进一步，因为无效，所以这个简单的操作允许给 I/O 处理器一个机会来改变公用存储器的内容，因为 L1 的全部内容很快从保留区中被替换。该软件之后指示 I/O 处理器继续进行，直到所有预期的消息到达。该软件之后允许计算机处理器继续处理指令，并且使用全局同步操作（例如在共同未决的申请 10/468997（代理卷号 15272）全局中断和屏障网络中描述的全局屏障）来关闭放/取窗口。

第二实施例：如果被接收的消息的大小大于 L1 的大小。

在这种情况中，本发明在大小等于 L1 的保留的连续物理地址范围上使用术语上称为“数据高速缓存块零”（DCBZ）的指令。DCBZ 创建一个内容为零的新高速存储线。如果新的高速存储线不可用，则 L1 中的另一个高速存储线（例如，最近最少使用的线）将其数据写回到公用存储器，并且之后使用由 DCBZ 指令给定的地址置零。DCBZ 是一个 PowerPC BookE 指令；对于其它处理器存在相似的指令。软件向保留区的每个线连续地执行 DCBZ，其中保留区的线的大小等于高速缓存线

并被类对齐。这造成 L1 中的所有线被清除，即，所有修改的线被写回公用存储器，因为 L1 的全部内容很快地从保留区中被替换。该软件之后允许计算机处理器继续处理指令，并且使用全局同步操作（例如在共同未决的申请 10/468997（代理卷号 15272）全局中断和屏障网络中描述的全局屏障）来关闭放/取窗口。

可以注意到：保留的物理空间不需要存在于物理存储器中，除非到该空间的访问不可能造成访问违例。到此保留的存储器空间的所有写必须是由存储器控制器确认的。到此保留的空间的所有读必须立即返回一个任意（即“垃圾”）值到请求处理器的 L1。还需要注意，这样的装置还提供用于无特权的（又称用户空间）程序的、用于对 L1 高速存储的全部内容进行清除并使所述内容无效的最有效的方法。

还可注意到，如果 DCBF 指令慢于 DCBZ，那么操作系统可以对慢于 L1 的消息使用 DCBZ 指令，并且反之亦然。

使用本发明，如果通信存储器空间被标记为向其 L1 高速缓存进行直接写入，则 I/O 处理器根本不需要清除其高速缓存。

在相关的专利申请 No.10/468991（代理人案号：15258）中，详细地讨论了对于上述全局的“和”在允许放/取窗口暂时变窄的短时间间隔中的建立。

虽然很明显这里描述的本发明是被充分计算以达到前述目标的，但可以理解，可以由本领域技术人员设计多种修改和实施例，希望附带的权利要求书覆盖所有的这种修改和实施例，作为落入本发明的真正精神和范围内。

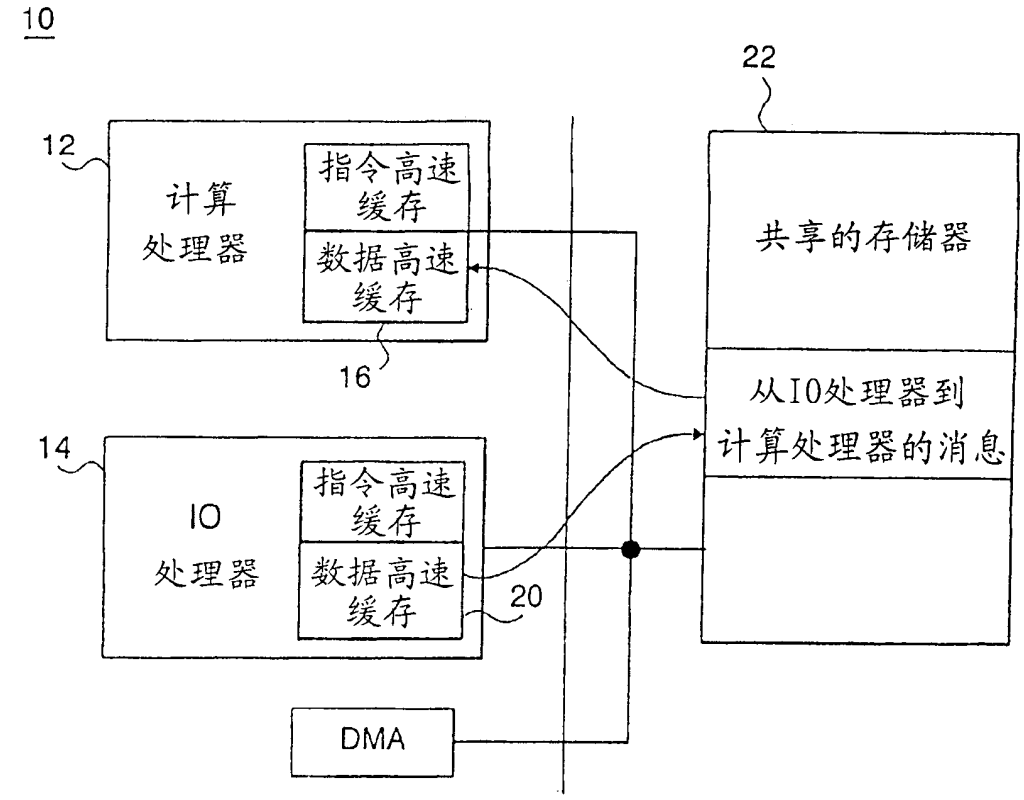

10
22
12
计算
处理器
指令高速
缓存
数据高速
缓存
16
共享的存储器
从IO处理器到
计算处理器的消息
14
IO
处理器
指令高速
缓存
数据高速
缓存
20
DMA

图 1

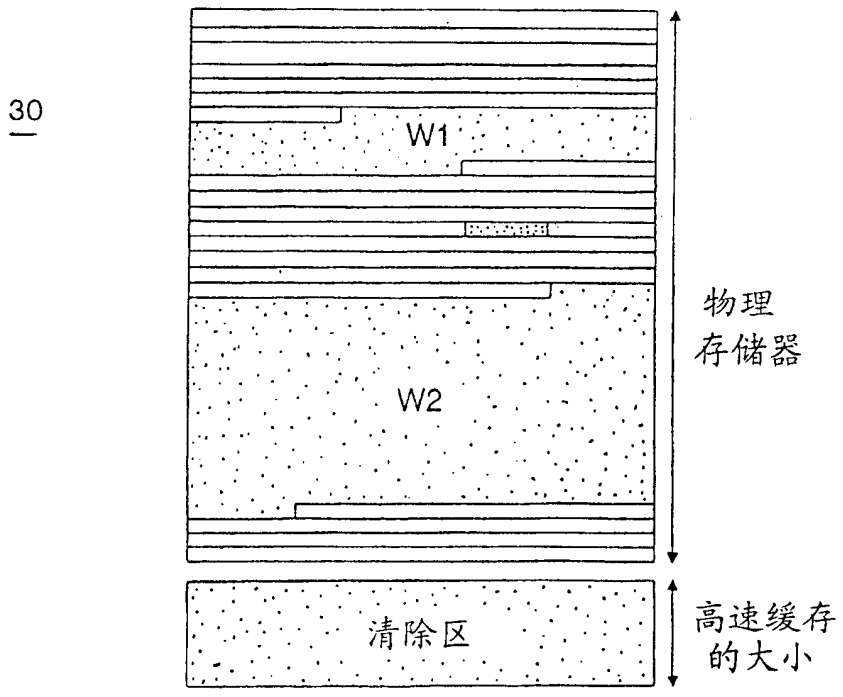

30
W1
W2
物理
存储器
清除区
高速缓存
的大小

图 2